eman ta zabal zazu

ESCUELA UNIVERSITARIA DE INGENIERIA TECNICA INDUSTRIAL DE BILBAO

GRADO EN INGENIERIA INFORMATICA DE GESTION Y SISTEMAS DE INFORMACION

TRABAJO FIN DE GRADO

2014 / 2015

GESTION WEB DE UN TALLER MECANICO

MEMORIA TFG

DATOS DE LA ALUMNA O DEL ALUMNO	**DATOS DEL DIRECTOR O DE LA DIRECTORA**
YASMIN CABEZA LARRAZABAL	MIKEL VILLAMAÑE GIRONES
	DEPARTAMENTO: LENGUAJES Y SISTEMAS INFORMÁTICOS
FDO.:	FDO.:
FECHA:	FECHA:

RESUMEN

El proyecto de fin de carrera "Gestión web de un taller mecánico" se ha desarrollado para la empresa Talleres Tauro y Richard, con sede en Galdakao.

El objetivo del proyecto es implantar una aplicación web, con un diseño elaborado, para que los clientes puedan acceder a toda la información referente a esta empresa. Esta aplicación cumplirá totalmente con la legislación española, puesto que posee aviso legal, política de privacidad y aviso de cookies, con su correspondiente política de cookies.

Para la realización de una web de tal magnitud, es necesario el uso de gestores de contenidos y se ha optado por la utilización de WordPress, cuyo número de usuarios que lo utiliza aumenta exponencialmente, debido a las numerosas prestaciones que tiene y a lo sencillo que es acceder a su código.

Una de las peculiaridades de esta aplicación web es que, además de mostrar toda la información referente a la empresa, posee un módulo, que se diseñará y desarrollará en este trabajo fin de grado, que permitirá a todos los clientes de la empresa poder saber, en cada instante, el estado de su automóvil y pagar la factura de la reparación desde casa, con total confianza y comodidad, y todo ello mediante PayPal o por transferencia bancaria.

Otro de los módulos que se va a desarrollar va a ser el panel de administración, a través del cual los administradores podrán realizar numerosas gestiones, así como, introducir un nuevo vehículo, modificar datos de vehículos o reparaciones existentes, ver el historial de todas las reparaciones que se han realizado en dicho taller, pudiendo ordenarlas por fecha o nombre y gestionar los servicios de mano de obra que irán incluidos en las facturas, pudiendo insertar, modificar o eliminar los que se desee.

Además el administrador también podrá visualizar las cámaras web del taller desde esta aplicación y podrá insertar ofertas que aparecerán visibles en la web para todos los usuarios, las cuales se publicarán, a su vez, en el muro del Facebook del taller.

Por último, el administrador podrá añadir o eliminar avisos, que servirán para estar conectados con la App móvil para Android, que también se va a desarrollar a lo largo de este trabajo fin de grado. Esta App permitirá a los clientes, que dispongan de ella, saber si la reparación de su vehículo ha finalizado, pedir cita a través de la misma e, incluso, saber con cuantos kilómetros debe cambiar el aceite o las pastillas de frenos o cual es la fecha de la próxima ITV, entre otras funcionalidades.

ÍNDICE DE CONTENIDOS

1. INTRODUCCION 1
1.1. Origen del proyecto 1
1.2. Razones de elección del TFG 2
1.3. Planteamiento del problema 2
2. PLANTEAMIENTO INICIAL 5
2.1. Objetivos del proyecto 5
2.2. Alcance del proyecto 6
2.2.1. Estructura de descomposición del proyecto 6
2.2.1.1. Captura de requisitos y gestión 8
2.2.1.2. Análisis 8
2.2.1.3. Diseño 9
2.2.1.4. Implementación 9
2.2.1.5. Pruebas 10
2.2.1.6. Creación de aplicación móvil 10
2.3. Planificación temporal 11
2.4. Arquitectura 13
2.4.1. Arquitectura aplicación web 13
2.4.2. Arquitectura aplicación móvil 15
2.5. Herramientas utilizadas 16
2.5.1. Lenguajes de programación 16
2.5.2. Tecnologías 17
2.5.3. Gestor de contenidos 20
2.5.4. Programas utilizados 20
2.6. Gestión de riesgos / planes de contingencia 21
2.7. Evaluación económica 26
2.7.1. Coste hardware 26
2.7.2. Coste software 26
2.7.3. Coste personal 27
2.7.4. Cálculo de los honorarios totales 27
2.7.5. Amortización de la herramienta 28
3. ANÁLISIS DE ANTECEDENTES 29
3.1. Historia 29
3.2. Situación actual 29
3.3. Estudio de diferentes alternativas 30
3.3.1. ERPs 30
3.3.2. Gestores de contenidos 31

3.3.3. Sistemas operativos para la aplicación móvil 32

4. CAPTURA DE REQUISITOS 35
4.1. Requisitos previos 35
4.2. Casos de uso 36
4.2.1. Jerarquía de actores de la aplicación web 36
4.2.2. Jerarquía de actores de la aplicación móvil 37
4.2.3. Modelo de casos de uso 37
4.2.3.1. Modelo de casos de uso de la aplicación Web 38
4.2.3.2. Modelo de casos de uso de la aplicación móvil 40
4.3. Modelo de dominio 41

5. ANÁLISIS Y DISEÑO 43
5.1. Diseño de la base de datos 43
5.1.1. Modelo relacional 44
5.1.2. Descripción detallada de la base de datos 44
5.2. Estructura de la aplicación web 47
5.2.1. Diseño de los módulos desarrollados 47
5.2.1.1. Módulo Administrar_datos_vehiculos 48
5.2.1.2. Módulo Pago_online_factura 50
5.3. Estructura de la aplicación móvil 52
5.3.1. Diagrama de clases 52

6. DESARROLLO 55
6.1. Implementación de la aplicación web 55
6.1.1. Introducción 55
6.1.2. Instalaciones previas 56
6.1.3. Gestor de contenidos WordPress 57
6.1.3.1. Estructura de WordPress 58
6.1.3.2. Los ganchos (Hooks) de WordPress 59
6.1.4. Diseño e implementación de la aplicación web 60
6.1.4.1. Diseño de la interfaz gráfica 60
6.1.4.2. Plugins instalados 66
6.1.5. Implementación de los módulos Panel de Administración y Pago online de la factura 68
6.1.5.1. Estructura de los archivos principales de los módulos 68
6.1.5.2. Llamadas entre archivos php 71
6.1.5.3. Conexiones con la base de datos 73
6.1.5.4. Inserción de imágenes en el servidor 75
6.1.5.5. Conexión de la aplicación web con Facebook 76
6.1.5.6. Creación de facturas en PDF 78
6.1.5.7. Envío de emails desde archivos php 81
6.1.5.8. Envío de notificaciones al servidor GCM 83

6.1.5.9. Generación de tablas dinámicas 84
6.1.5.10. Validación de los formularios 85
6.1.5.11. Suma de variables 87
6.1.5.12. Pagos a través de la aplicación web 88
6.1.5.13. Integración de cámaras web en la aplicación 89
6.2. Implementación de la aplicación móvil 91
6.2.1. Sistema operativo Android 92
6.2.2. Componentes de la aplicación 93
6.2.2.1. Estructura de la aplicación 94
6.2.2.2. Activity 95
6.2.2.3. Services 96
6.2.2.4. Broadcast Receiver 96
6.2.2.5. AndroidManifest.xml 97
6.2.3. Detalles de implementación 99
6.2.3.1. Llamadas entre actividades 99
6.2.3.2. Servicios web 100
6.2.3.3. Notificaciones Push 102
6.2.3.4. Realización de tablas dinámicas 110
6.2.3.5. Calendario dinámico 112
6.2.4. Diseño de la interfaz gráfica 114
7. VERIFICACIÓN Y EVALUACIÓN 121
7.1. Objetivos 121
7.2. Pruebas realizadas 121
7.2.1. Pruebas relacionadas con el servidor 121
7.2.2. Pruebas relacionadas con la aplicación web 122
7.2.3. Pruebas relacionadas con la aplicación móvil 144
8. CONCLUSIONES Y TRABAJO FUTURO 149
8.1. Conclusiones finales 149
8.2. Revisión de los objetivos 150
8.3. Desviaciones de la planificación inicial 153
8.3.1. Problemas encontrados 153
8.3.2. Planificación final 154
8.4. Posibles ampliaciones y mejoras 156
9. BIBLIOGRAFÍA 159
10. ANEXO I . – CASOS DE USO EXTENDIDOS 167
11. ANEXO II. – DIAGRAMAS DE SECUENCIA 257

ÍNDICE DE FIGURAS

Ilustración 1. - Estructura de descomposición del proyecto (EDT) 7
Ilustración 2. - Diagrama de Gantt 12
Ilustración 3. - Arquitectura de la aplicación web 14
Ilustración 4. - Arquitectura App móvil 16
Ilustración 5. - Jerarquía de actores de la aplicación web 37
Ilustración 6. - Jerarquía de actores de la App Tauro & Richard. 37
Ilustración 7. - Modelo de casos de uso de la aplicación web 38
Ilustración 8. - Modelo de casos de uso de la App Tauro & Richard 40
Ilustración 9. - Modelo de dominio 41
Ilustración 10. - Estructura del módulo Administrar datos vehículos 49
Ilustración 11. - Estructura del módulo pago online de la factura 51
Ilustración 12. - Diagrama de clases de la aplicación móvil 53
Ilustración 13. - FTP Filezilla 57
Ilustración 14. - Estructura de una página web [43] 58
Ilustración 15. - Funcionamiento de WordPress [44] 60
Ilustración 16. - Escritorio de WordPress 61
Ilustración 17. - Página Empresa de la aplicación web 62
Ilustración 18. - Página principal de la aplicación 63
Ilustración 19. - Páginas que forman la aplicación web 64
Ilustración 20. - Entorno de administración y de clientes 65
Ilustración 21. - Aviso legal y política de cookies 66
Ilustración 22. - Banner de cookies 66
Ilustración 23. - Página de plugins de la aplicación web 67
Ilustración 24. - Parte inicial del archivo principal.php 68
Ilustración 25. – Parte inicial del archivo índice.php 68
Ilustración 26. - Introducir ofertas en Facebook 78
Ilustración 27. - Facturas en pdf 81
Ilustración 28. - Visualización de las cámaras web 91
Ilustración 29. - Estructura de Android [45] 92
Ilustración 30. - Estructura de la aplicación móvil 95
Ilustración 31. - Servicios web 101
Ilustración 32. - Tráfico de una notificación push [46] 103
Ilustración 33. - Notificaciones en el dispositivo móvil 110
Ilustración 34. - Calendario de la App móvil 114
Ilustración 35. - Creación de archivos XML de Android 115
Ilustración 36. - App con rotación de imagen 117
Ilustración 37. - Mapa de navegación de la aplicación móvil 119

Ilustración 38. - Diagrama final de Gantt 155
Ilustración 39. - Caso de uso Identificación admin 167
Ilustración 40. - Entorno de administración 168
Ilustración 41. - Menú del administrador 169
Ilustración 42. - Contraseña del administrador incorrecta 169
Ilustración 43. - Caso de uso Introducir nueva reparación 170
Ilustración 44. - Entorno de administración 172
Ilustración 45. - Menú del administrador 172
Ilustración 46. – Selección de matrícula del vehículo a reparar 173
Ilustración 47. - Selección de nuevo vehículo 173
Ilustración 48. - Formulario de inserción de una reparación para vehículo existente 174
Ilustración 49. – Formulario de Inserción de reparación para vehículo nuevo 175
Ilustración 50. - Mensaje de error de campo nombre 175
Ilustración 51. - Mensaje de error de campo email 176
Ilustración 52. - Mensaje de error de campo teléfono 176
Ilustración 53. - Caso de uso Modificar reparación de vehículo 177
Ilustración 54. - Entorno de administración 179
Ilustración 55. - Menú de administración 180
Ilustración 56. - Selección de matrícula al modificar datos 180
Ilustración 57. - Elegir fecha de la reparación del vehículo 181
Ilustración 58. - Formulario de modificación de datos de la reparación 181
Ilustración 59. - Mensaje de error de campo nombre 182
Ilustración 60. - Mensaje de error de campo email 182
Ilustración 61. - Mensaje de error de campo teléfono 182
Ilustración 62. - Caso de uso Gestionar servicios mano de obra 183
Ilustración 63. - Entorno de administración 184
Ilustración 64. - Menú del administrador 185
Ilustración 65. - Gestión de servicios de mano de obra 185
Ilustración 66. - Inserción de nuevo servicio 186
Ilustración 67. - Mensaje de error de campo nombre 186
Ilustración 68. - Mensaje de error de campo precio 186
Ilustración 69. - Modificación de los datos de un servicio 187
Ilustración 70. - Eliminar un servicio 187
Ilustración 71. - Caso de uso Historial de reparaciones 188
Ilustración 72. - Entorno de administración 190
Ilustración 73. - Menú del administrador 190
Ilustración 74. - Historial de reparaciones 191
Ilustración 75. - Ordenar historial 191
Ilustración 76. - Ver factura de la reparación 192
Ilustración 77. - Factura de la reparación sin pagar 193

Ilustración 78. - Factura de la reparación pagada 194
Ilustración 79. - Modificar datos de la reparación 194
Ilustración 80. - Formulario de modificar datos de la reparación 195
Ilustración 81. - Mensaje de error de campo nombre 195
Ilustración 82. - Mensaje de error de campo email 196
Ilustración 83. - Mensaje de error de campo teléfono 196
Ilustración 84. - Eliminar datos de la reparación 196
Ilustración 85. - Eliminar historial de un vehículo 197
Ilustración 86. - Caso de uso Ver cámaras web 198
Ilustración 87. - Entorno de administración 199
Ilustración 88. - Menú del administrador 199
Ilustración 89. - Cámaras web 200
Ilustración 90. - Caso de uso Adminsitrar avisos 201
Ilustración 91. - Entorno de administración 203
Ilustración 92. - Menú del administrador 204
Ilustración 93. - Menú de administrar avisos 204
Ilustración 94. - Introducir nuevo aviso 205
Ilustración 95. - Introducir nuevo aviso - Escoger aviso 205
Ilustración 96. - Introducir nuevo aviso - Finalización de la reparación 205
Ilustración 97. - Introducir nuevo aviso - Formulario finalización de la reparación 205
Ilustración 98. - Introducir nuevo aviso - fechas 206
Ilustración 99. - Introducir nuevo aviso - Kilometrajes 206
Ilustración 100. - Mensaje de error Fecha inválida 206
Ilustración 101. - Mensaje de error de valor numérico incorrecto 207
Ilustración 102. - Mensaje de error de falta campos por rellenar 207
Ilustración 103. - Mensaje de error de falta campos por rellenar 207
Ilustración 104. - Eliminar aviso 207
Ilustración 105. - Eliminar aviso seleccionado 208
Ilustración 106. - Caso de uso Adminsitrar ofertas 209
Ilustración 107. - Entorno de administración 211
Ilustración 108. - Menú del administrador 211
Ilustración 109. - Menú de administrar ofertas 212
Ilustración 110. - Introducir nueva oferta 212
Ilustración 111. - Mensaje de error de campos sin rellenar 212
Ilustración 112. - Página de Ofertas 213
Ilustración 113. - Publicación de oferta en Facebook 214
Ilustración 114. - Modificar oferta 214
Ilustración 115. - Formulario de modificación de oferta 215
Ilustración 116. - Eliminar oferta 215
Ilustración 117. - Eliminar oferta seleccionada 215

Ilustración 118. - Caso de uso Ver ofertas 216
Ilustración 119. - Menú clientes 217
Ilustración 120. - Página de Ofertas 217
Ilustración 121. - Detalles de la oferta 218
Ilustración 122. - Caso de uso identificación cliente 219
Ilustración 123. - Entorno de clientes 220
Ilustración 124. - Mensaje de error de los datos incorrectos 220
Ilustración 125. - Mensaje de error del campo matrícula 220
Ilustración 126. - Menú de clientes 221
Ilustración 127. - Caso de uso identificación cliente 222
Ilustración 128. - Entorno de clientes 223
Ilustración 129. - Formulario de recuperación de contraseña 223
Ilustración 130. – Caso de uso Modificar datos personales 224
Ilustración 131. - Entorno de clientes 225
Ilustración 132. - Menú de clientes 226
Ilustración 133. - Formulario de modificación de datos personales 226
Ilustración 134. - Mensaje de error de campo nombre 226
Ilustración 135. - Mensaje de error de campo email 226
Ilustración 136. - Mensaje de error de campo teléfono 227
Ilustración 137. – Caso de uso Ver historial del vehículo 228
Ilustración 138. - Entorno de clientes 229
Ilustración 139. - Menú de clientes 230
Ilustración 140. - Historial de reparaciones 230
Ilustración 141. - Ver factura 231
Ilustración 142. - Factura de la reparación sin pagar 232
Ilustración 143. - Factura de la reparación pagada 233
Ilustración 144. - Pagar factura de la reparación 233
Ilustración 145. - Pagar y ver la factura 234
Ilustración 146. - Pago con transferencia bancaria 235
Ilustración 147. - Pago con Paypal 235
Ilustración 148. - Página de agradecimiento 236
Ilustración 149. –Caso de uso Pagar mis facturas 237
Ilustración 150. - Entorno de clientes 239
Ilustración 151. - Menú de clientes 239
Ilustración 152. – Pagar mis facturas 240
Ilustración 153. - Ver factura 240
Ilustración 154. - Factura de la reparación sin pagar 241
Ilustración 155. - Factura de la reparación pagada 242
Ilustración 156. - Pagar factura de la reparación 242
Ilustración 157. - Pagar y ver la factura 243

Ilustración 158. - Pago con transferencia bancaria 244
Ilustración 159. - Pago con Paypal 244
Ilustración 160. - Página de agradecimiento 245
Ilustración 161. - Caso de uso Identificación 245
Ilustración 162. – Identificación clientes 246
Ilustración 163. - Menú cliente 247
Ilustración 164. – Error por datos incorrectos 247
Ilustración 165. – Error por campos vacíos 248
Ilustración 166. - Caso de uso Pedir Cita 249
Ilustración 167. - Menú cliente 250
Ilustración 168. - Pedir cita 250
Ilustración 169. - Calendario 251
Ilustración 170. - Mensaje de éxito 251
Ilustración 171. - Mensaje de error 252
Ilustración 172. - Caso de uso Consultar revisiones 253
Ilustración 173. - Menú cliente 254
Ilustración 174. - Revisiones 254
Ilustración 175. - Caso de uso Consultar revisiones 255
Ilustración 176. - Alerta notificación 255
Ilustración 177. - Aviso de finalización de la reparación 256
Ilustración 178. - Diagrama de caso de uso de Identificación Admin 258
Ilustración 179. - Diagrama de secuencia de caso de uso Introducir nueva reparación 265
Ilustración 180. - Diagrama de secuencia de caso de uso Modificar reparación de vehículo 272
Ilustración 181. - Diagrama de secuencia de caso de uso Gestionar Servicios 275
Ilustración 182. - Diagrama de secuencia de caso de uso Historial de reparaciones 288
Ilustración 183. - Diagrama de caso de uso Ver cámaras web 289
Ilustración 184. - Diagrama de secuencia del caso de uso Administrar avisos 297
Ilustración 185. - Diagrama de secuencia de caso de uso Administrar Ofertas 303
Ilustración 186. - Diagrama de secuencia de caso de uso Ver ofertas 305
Ilustración 187. - Diagrama de secuencia de caso de uso Identificación cliente 307
Ilustración 188. - Diagrama de secuencia del caso de uso Recuperar contraseña 309
Ilustración 189. - Diagrama de secuencia de caso de uso Modificar datos personales 312
Ilustración 190. - Diagrama de secuencia de caso de uso Ver historial del vehículo 320
Ilustración 191. - Diagrama de secuencia de caso de uso Pagar mis facturas 328
Ilustración 192. - Diagrama de secuencia App - Identificación 331
Ilustración 193. - Diagrama de secuencia de caso de uso Pedir Cita 334
Ilustración 194. - Diagrama de secuencia de caso de uso Consultar revisiones 337
Ilustración 195. - Diagrama de secuencia de caso de uso Ver nuevo aviso 339

ÍNDICE DE TABLAS

Tabla 1. - Tabla de gestión de riesgos 25
Tabla 2. - Tabla de evaluación económica 28

1. INTRODUCCION

Este documento pretende materializar el trabajo realizado durante el desarrollo del Trabajo Fin de Grado "Gestión web de un taller mecánico". Este capítulo es el punto de entrada al resto del documento, proporcionando una visión general sobre el tema, incluyendo motivaciones por las cuales se ha llevado a cabo este proyecto.

1.1. Origen del proyecto

Hoy en día, todo el mundo habla de fenómenos globales, o de globalización, y de lo que se trata es de la condición de acceso irrestricto a bienes y servicios de naturaleza diversa para el consumo mundial. En consecuencia, las condiciones técnicas que exige la globalización implican una vinculación informática y tecnológica. Estar globalizado implica entonces, estar a la vanguardia o bien conocer la tecnología punta de una época determinada. Lo contrario sería rechazar la evolución significaría, en resumidas cuentas, lo que esboza la siguiente frase: dejar de tener relevancia, caer en el olvido, dejar de existir.

Es por esto que se ha producido un aumento de la competitividad entre los mercados y cada vez más empresas intentan buscar mejoras en su actividad. Por consiguiente, se intenta ofrecer mejores servicios con el fin de fidelizar clientes.

Talleres Tauro y Richard es una pequeña empresa especializada en la reparación de chapa, pintura y mecánica. La evolución positiva de esta empresa, a lo largo de los años, ha ido incrementando su número de clientes y proveedores. Por ello, desean invertir en nuevos canales de marketing, trayendo consigo las garantías de la satisfacción de sus clientes, a la vez de captar nuevos.

Cabe destacar, la importancia de tener una aplicación web con la que poder llevar a cabo la gestión o administración de la empresa, utilizando cualquier ordenador y desde cualquier lugar. Además, se brinda a los clientes una herramienta cómoda, mediante la cual pueden estar al corriente de todas las novedades sobre dicha empresa y ofrecerles numerosas prestaciones para su máxima satisfacción.

Por último, mencionar que, hoy en día, las aplicaciones móviles pueden convertirse en un buen escaparate de la marca, por la posibilidad que ofrecen para llegar a un gran número de usuarios. Gracias a este rasgo, las empresas pueden utilizar las aplicaciones móviles como canales de venta, ofreciendo a través de las mismas sus productos y servicios a los clientes.

1.2. Razones de elección del TFG

La motivación principal de la realización de este proyecto ha sido ver cómo mis conocimientos adquiridos durante la carrera, más los adquiridos durante las prácticas realizadas, me permiten ejecutar correctamente este trabajo, desde el inicio hasta el final.

Otra de las motivaciones para la elaboración de este trabajo fin de grado, ha sido poder crear una aplicación web para una pyme real y que además se vaya a instaurar en su empresa. Este trabajo fin de grado es el primer proyecto complejo al que me voy a enfrentar, tras terminar los estudios de Grado en Ingeniería Informática de Gestión y Sistemas de Información. Esto supone el luchar contra un gran reto, tanto en el plano personal como profesional.

Siento una gran inclinación por el mundo de las páginas web, posicionamiento SEO/ SEM y marketing online y, en un futuro, me gustaría dedicarme profesionalmente a él. Otra de las causas, es el inmenso interés por todo lo relacionado con el mundo empresarial, en el que poco a poco espero hacerme un hueco y poder dedicarme a lo que me gusta en cuerpo y alma.

Asimismo, también tengo gran curiosidad por el mundo de las aplicaciones Android, que ahora mismo, son casi tan importantes como una aplicación web. Por consiguiente, al no tener ningún conocimiento en este ámbito, he decidido aprender a desarrollarlas, mediante este Trabajo de Fin de Grado, y llevar a cabo su implementación e implantación en la empresa.

1.3. Planteamiento del problema

Este proyecto parte de la necesidad de la Pyme de implantar una página web, con la que desean generar una imagen moderna y tecnológica de la empresa para sus clientes, potenciando y actualizando así su imagen corporativa y difundiéndola por las redes sociales.

Por otro lado, la empresa necesita centralizar todos los datos de las reparaciones de los vehículos en una sola aplicación.

La empresa desea ofrecer a los clientes nuevas mejoras y comodidades, como la opción de poder realizar el pago de la factura a través de internet o poder descargarse la factura mediante la aplicación web.

Por otra parte, la empresa inicialmente no cuenta con apariciones en redes sociales, lo cual ahora mismo es primordial para darse a conocer.

Destacar que la empresa tiene unas cámaras web en su edificio y solo es posible visualizarlas por medio de un monitor de la oficina. Esta sería una buena oportunidad, para dar la posibilidad a la empresa, de poder acceder a esas cámaras web desde cualquier parte, mediante el uso de la aplicación.

En última instancia, comentar que la compañía desea tener una herramienta, con la que pueda avisar a sus clientes de novedades, directamente en su dispositivo móvil.

2. PLANTEAMIENTO INICIAL

La finalidad de este capítulo es indicar cuales son los objetivos y el alcance del proyecto, así como realizar una estimación económica y temporal del mismo. También se lleva a cabo la gestión de riesgos, la especificación de la arquitectura de las aplicaciones a realizar, además de describir las herramientas con las cuales se llevará a cabo su desarrollo.

2.1. Objetivos del proyecto

Los objetivos que debe cumplir este trabajo fin de grado son los siguientes:

- Herramienta intuitiva y potente, para que la experiencia de usuario sea lo más gratificante posible.
- Aplicación accesible desde cualquier lugar donde haya conexión a internet.
- Gestión cómoda y fácil de todas las reparaciones.
- Darse a conocer en las redes sociales.
- Visualización de las cámaras IP desde cualquier lugar.
- Facilitar a los clientes el pago de las facturas sin necesidad de acudir al taller.
- Conexión completa entre clientes y empresa y que ésta pueda comunicarles la finalización de la reparación.

Los objetivos personales de este TFG son los siguientes:

- Aprender a desarrollar aplicaciones Android y afianzar los conocimientos de Java adquiridos hasta ahora.
- Lograr una buena formación en el desarrollo de módulos, para el gestor de contenidos WordPress, poniendo en práctica todos los conocimientos adquiridos, durante toda la carrera, de PHP, JavaScript y AJAX.
- Estudiar la programación de pasarelas de pago como PayPal.
- Instrucción en la integración de cámaras IP en aplicaciones web, mediante redes.
- Aprender a vincular WordPress con Facebook mediante programación.

Para ello, será necesario realizar una buena captura de requisitos y un estudio de viabilidad, para detectar las posibles carencias funcionales que pueda poseer el gestor de contenidos, con el que se va a realizar la web, y poder amoldarlo exactamente a lo que la empresa necesita.

2.2. Alcance del proyecto

Una de las primeras tareas de todo proyecto, es la definición de su alcance, o dicho de otra forma, delimitar las funcionalidades a desarrollar (en el caso de un producto software) para poder cumplir con todos sus objetivos al finalizar el proyecto.

2.2.1. Estructura de descomposición del proyecto

En la siguiente imagen se expone la Estructura de Descomposición del proyecto (EDT), indicando las partes en las que se divide y, a continuación, se detalla el número de horas dedicadas a cada apartado del EDT. En la siguiente sección, planificación temporal, se detalla el número de días que se va a dedicar a cada apartado de este EDT.

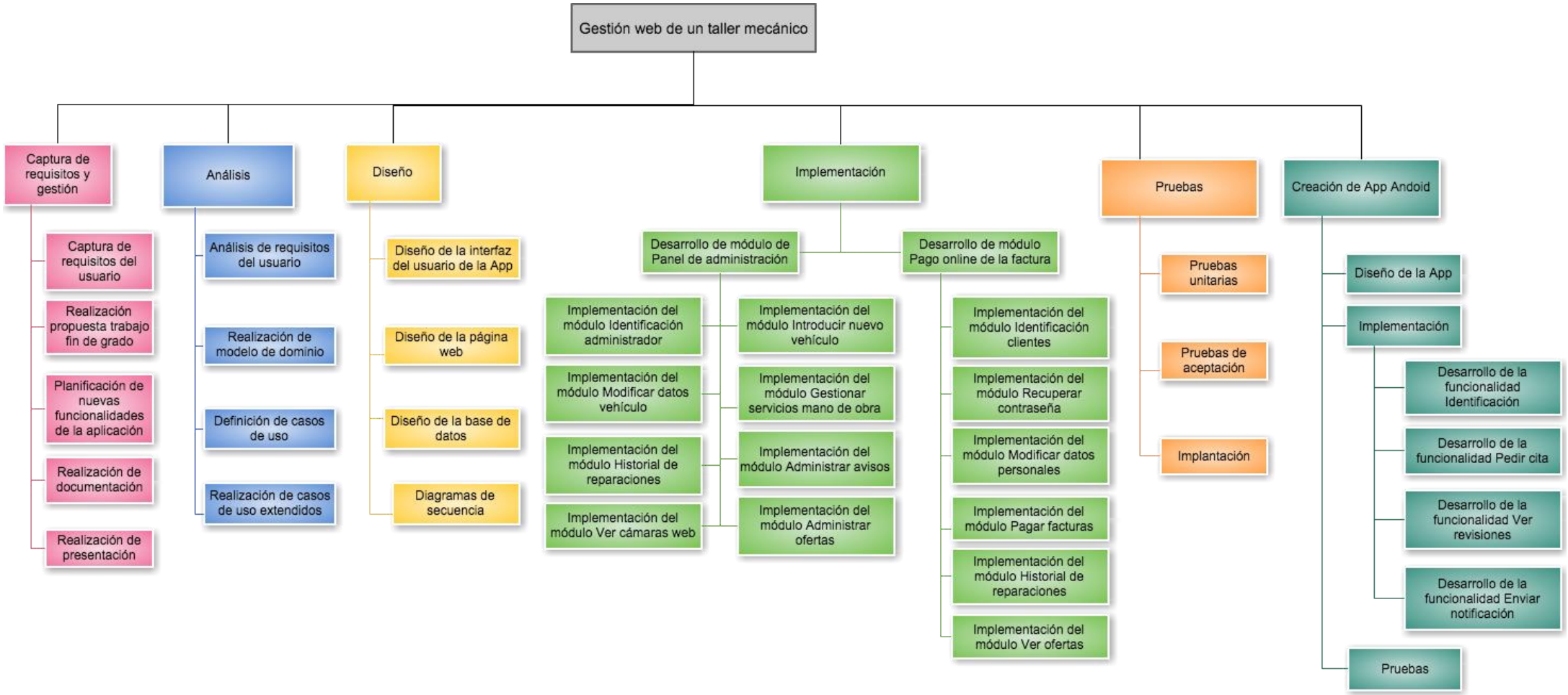


Ilustración 1. - Estructura de descomposición del proyecto (EDT)

A continuación , se van a detallar las horas dedicadas a cada tarea de este EDT.

2.2.1.1. Captura de requisitos y gestión

- **Captura de requisitos del usuario:** Extracción de requisitos del usuario para, más adelante, definir las funcionalidades de la aplicación →12 horas.
- **Realización de propuesta de trabajo fin de grado:** El tiempo estimado para la redacción del documento propuesta de trabajo fin de grado → 10 horas.
- **Planificación de nuevas funcionalidades de la aplicación:** Tiempo dedicado a planificar y detallar las nuevas funcionalidades que tendrá la aplicación web → 50 horas.
- **Realización de documentación:** Realización de toda la memoria del proyecto → 120 horas.
- **Realización de presentación:** Documento para la realización de la presentación del trabajo fin de grado frente al tribunal → 30 horas.

Total horas de fase captura de requisitos y gestión: 222 horas.

2.2.1.2. Análisis

- **Análisis de requisitos de usuario:** Análisis exhaustivo de los requisitos obtenidos en la captura de requisitos →12 horas.
- **Realización de modelo de dominio:** Estudio de las necesidades de la base de datos y la relación entre las diferentes tablas → 4 horas.
- **Definición de casos de uso:** Realización del diagrama de casos de uso, tanto de la aplicación web como de la móvil, donde se especifica el papel que juegan los actores con los casos de uso relacionados → 1 hora.
- **Realización de casos de uso extendidos:** Realización de documentación de todos los casos de uso, tanto de la aplicación web como de la aplicación móvil, de manera detallada, explicando en profundidad la funcionalidad de cada uno →50 horas.

Total horas de fase análisis: 67 horas.

2.2.1.3. Diseño

- **Diseño de la interfaz del usuario de la App:** Diseño en papel de todas las interfaces de la App móvil → 30 horas.
- **Diseño de la página web:** Diseño gráfico de la interfaz y diseño de los contenidos de cada página de la aplicación web → 45 horas.
- **Diseño de la base de datos:** Diseño del modelo de datos a utilizar a partir del modelo de dominio → 5 horas.
- **Diagramas de secuencia:** Realización de los diagramas de secuencia, tanto de la aplicación móvil como de la aplicación web →90 horas.

Total horas de fase diseño: 170 horas.

2.2.1.4. Implementación

Desarrollo de módulo panel de administración:

- Implementación del **módulo Identificación administrador** → 5 horas.
- Implementación del **módulo Introducir nuevo vehículo** → 24 horas.
- Implementación del **módulo Modificar datos vehículo** → 32 horas.
- Implementación del **módulo Historial de reparaciones** → 55 horas.
- Implementación del **módulo Ver cámaras web** → 7 horas.
- Implementación del **módulo Gestionar servicios mano de obra** →29 horas.
- Implementación del **módulo Administrar avisos** → 20 horas.
- Implementación del **módulo Administrar ofertas** → 24 horas.

Desarrollo de módulo pago online de la factura:

- Implementación del módulo **Identificación clientes** → 10 horas.
- Implementación del módulo **Recuperar contraseña** → 11 horas.
- Implementación del módulo **Historial de reparaciones** → 25 horas.
- Implementación del módulo **Modificar datos personales** → 18 horas.
- Implementación del módulo **Pagar facturas** → 20 horas.

- Implementación del módulo **Ver ofertas** → 2 horas.

Total horas fase de implementación: 282 horas.

2.2.1.5. Pruebas

- **Pruebas unitarias:** Pruebas de cada uno de los módulos y de cada una de las funcionalidades → 70 horas.
- **Pruebas de aceptación:** Realización de pruebas por un grupo de usuario finales para asegurarse de que el sistema cumple los requisitos. → 12 horas.
- **Implantación:** Puesta en marcha de la aplicación web →9 horas.

Total horas de la fase de pruebas: 91 horas.

2.2.1.6. Creación de aplicación móvil

Diseño de la App: Implementación de todas las interfaces de la App → 35 horas.

Implementación:

- Desarrollo de la **funcionalidad identificación** → 50 horas.
- Desarrollo de la **funcionalidad Pedir cita** → 15 horas.
- Desarrollo de la **funcionalidad Ver revisiones** → 10 horas.
- Desarrollo de la **funcionalidad Enviar notificación** → 18 horas.

Pruebas: Realización de todas las pruebas de cada funcionalidad de la App e implantación de la App en un dispositivo móvil → 15 horas.

Total horas de la fase de Creación de App para Android: 143 horas.

Después de la realización de la descomposición del proyecto, se puede afirmar que dicho proyecto conlleva 975 horas de trabajo.

2.3. Planificación temporal

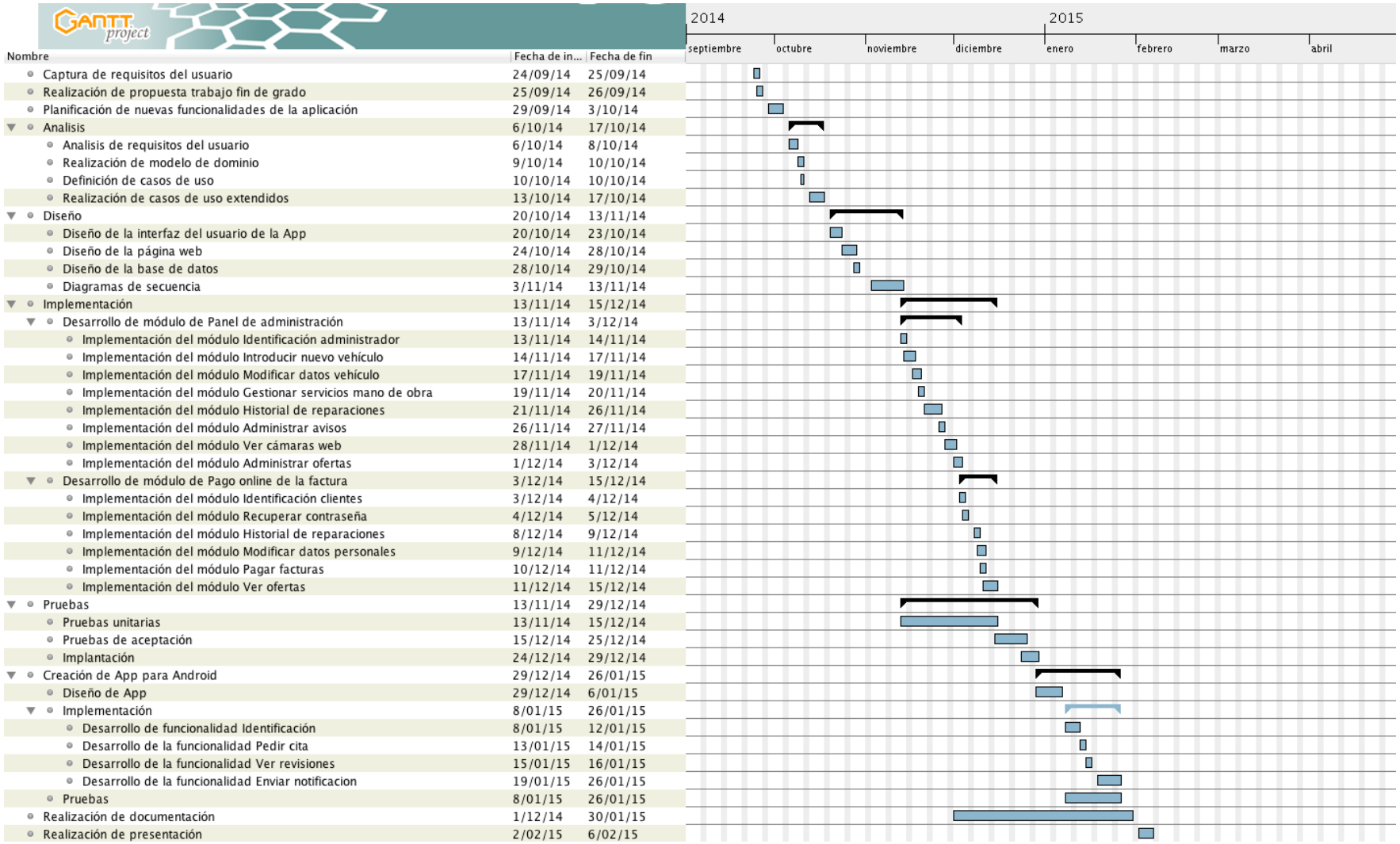
A este proyecto se le va a dedicar una media de 8 horas al día, como se ha visto en el apartado del alcance del proyecto, y comprenderá los meses de Septiembre a Febrero. La planificación de este trabajo fin de grado se ha estructurado en seis fases:

- **Fase 1 (Septiembre - Octubre):** Primeras reuniones con el tutor del proyecto, búsqueda de ideas, captura de requisitos, concreción de objetivos a alcanzar y planificación a seguir. Se realizará la propuesta de trabajo de fin de grado y una vez aceptada se comenzará con toda la parte introductoria de la memoria.
- **Fase 2 (Octubre):** Se realizará la planificación de nuevas funcionalidades de la aplicación y se comenzará con el análisis de requisitos del usuario, modelo de dominio, definición de casos de uso y realización de casos de uso extendido.
- **Fase 3 (Noviembre) :** Se deberá realizar el diseño de la interfaz de usuario de la App, diseño de la página web, diseño de la base de datos y los diagramas de secuencia.
- **Fase 4 (Diciembre):** Implementación de todos los módulos y funcionalidades de la página web. A su vez, se van a realizar todas las pruebas unitarias y de integración.
- **Fase 5 (Enero - Febrero):** Realización de la App para Android y finalización de la memoria del trabajo fin de grado. Se realizarán las conclusiones del mismo, la bibliografía, los diferentes índices y los anexos.
- **Fase 6 (Febrero):** Preparación de la presentación que se realizará para presentar el proyecto ante del tribunal.

Todas las fases serán supervisadas por el tutor del proyecto, corrigiendo cada vez que no se estén realizando las tareas correctamente.

En la siguiente figura se observa un diagrama de Gantt, cuyo objetivo es mostrar el tiempo de dedicación previsto para las diferentes fases, a lo largo del tiempo total que va a ocupar la elaboración de este proyecto.

Esta planificación es teórica, puesto que es el primer proyecto de tal magnitud que se realiza, por lo que no se ajustará al tiempo empleado en realidad, por lo tanto, en el apartado de conclusiones aparecerá el tiempo real empleado en cada fase, pudiéndose haber modificado respecto a este.



Nombre	Fecha de in...	Fecha de fin
Captura de requisitos del usuario	24/09/14	25/09/14
Realización de propuesta trabajo fin de grado	25/09/14	26/09/14
Planificación de nuevas funcionalidades de la aplicación	29/09/14	3/10/14
Analisis	6/10/14	17/10/14
Analisis de requisitos del usuario	6/10/14	8/10/14
Realización de modelo de dominio	9/10/14	10/10/14
Definición de casos de uso	10/10/14	10/10/14
Realización de casos de uso extendidos	13/10/14	17/10/14
Diseño	20/10/14	13/11/14
Diseño de la interfaz del usuario de la App	20/10/14	23/10/14
Diseño de la página web	24/10/14	28/10/14
Diseño de la base de datos	28/10/14	29/10/14
Diagramas de secuencia	3/11/14	13/11/14
Implementación	13/11/14	15/12/14
Desarrollo de módulo de Panel de administración	13/11/14	3/12/14
Implementación del módulo Identificación administrador	13/11/14	14/11/14
Implementación del módulo Introducir nuevo vehículo	14/11/14	17/11/14
Implementación del módulo Modificar datos vehículo	17/11/14	19/11/14
Implementación del módulo Gestionar servicios mano de obra	19/11/14	20/11/14
Implementación del módulo Historial de reparaciones	21/11/14	26/11/14
Implementación del módulo Administrar avisos	26/11/14	27/11/14
Implementación del módulo Ver cámaras web	28/11/14	1/12/14
Implementación del módulo Administrar ofertas	1/12/14	3/12/14
Desarrollo de módulo de Pago online de la factura	3/12/14	15/12/14
Implementación del módulo Identificación clientes	3/12/14	4/12/14
Implementación del módulo Recuperar contraseña	4/12/14	5/12/14
Implementación del módulo Historial de reparaciones	8/12/14	9/12/14
Implementación del módulo Modificar datos personales	9/12/14	11/12/14
Implementación del módulo Pagar facturas	10/12/14	11/12/14
Implementación del módulo Ver ofertas	11/12/14	15/12/14
Pruebas	13/11/14	29/12/14
Pruebas unitarias	13/11/14	15/12/14
Pruebas de aceptación	15/12/14	25/12/14
Implantación	24/12/14	29/12/14
Creación de App para Android	29/12/14	26/01/15
Diseño de App	29/12/14	6/01/15
Implementación	8/01/15	26/01/15
Desarrollo de funcionalidad Identificación	8/01/15	12/01/15
Desarrollo de la funcionalidad Pedir cita	13/01/15	14/01/15
Desarrollo de la funcionalidad Ver revisiones	15/01/15	16/01/15
Desarrollo de la funcionalidad Enviar notificacion	19/01/15	26/01/15
Pruebas	8/01/15	26/01/15
Realización de documentación	1/12/14	30/01/15
Realización de presentación	2/02/15	6/02/15

Ilustración 2. - Diagrama de Gantt

2.4. Arquitectura

En este apartado se va a realizar una descripción detallada de la arquitectura en la que se van a basar tanto la aplicación web, como la aplicación móvil. Servirá de referencia a la hora de realizar el diseño e implementación de ambas aplicaciones.

2.4.1. Arquitectura aplicación web

La aplicación web se basa en una arquitectura cliente/servidor: Por un lado, se encuentra el cliente (el navegador, explorador o visualizador) y, por otro lado, el servidor (el servidor web). Existen diferentes variantes de la arquitectura básica, según cómo se vayan a implementar las diferentes funcionalidades de la parte servidor.

La utilización de esta arquitectura se justifica por las siguientes razones:

- **Escalabilidad:** Esta arquitectura otorga la posibilidad de aumentar la capacidad de clientes y servidores por separado. Cualquier elemento se puede aumentar o mejorar o se pueden añadir más nodos a la red, ya sean clientes o servidores.
- Esta arquitectura se basa en la existencia de **una única lógica de negocio,** que se encuentra en el servidor. Esto sumado a **la centralización de datos** origina la innecesaria replicación de datos en cada una de las máquinas cliente. Por otro lado, también permite la realización de cambios y actualizaciones de datos con gran facilidad.

Las aplicaciones web se modelan mediante lo que se conoce como modelo de capas. Una capa representa un elemento que procesa o trata información. En esta aplicación se va a utilizar el modelo de 3-capas, en el cual, la información atraviesa tres capas. En la Ilustración 3 se puede apreciar la estructura del sistema.

Las capas de este modelo son:

Capa clientes

- Recoge la información del usuario.
- Manda información a la capa de proceso para su procesado.
- Recibe los resultados de la capa de proceso.
- Generan la presentación.
- Visualizan la presentación al usuario.

Capa de proceso (Capa servidor web)

- Recibe la entrada de datos de la capa de clientes.
- Interactúa con la capa de datos para realizar operaciones.
- Manda los resultados procesados a la capa de clientes.

Capa de datos (Base de datos)

- Almacena los datos.
- Recupera datos.
- Mantiene los datos.
- Asegura la integridad de los datos.

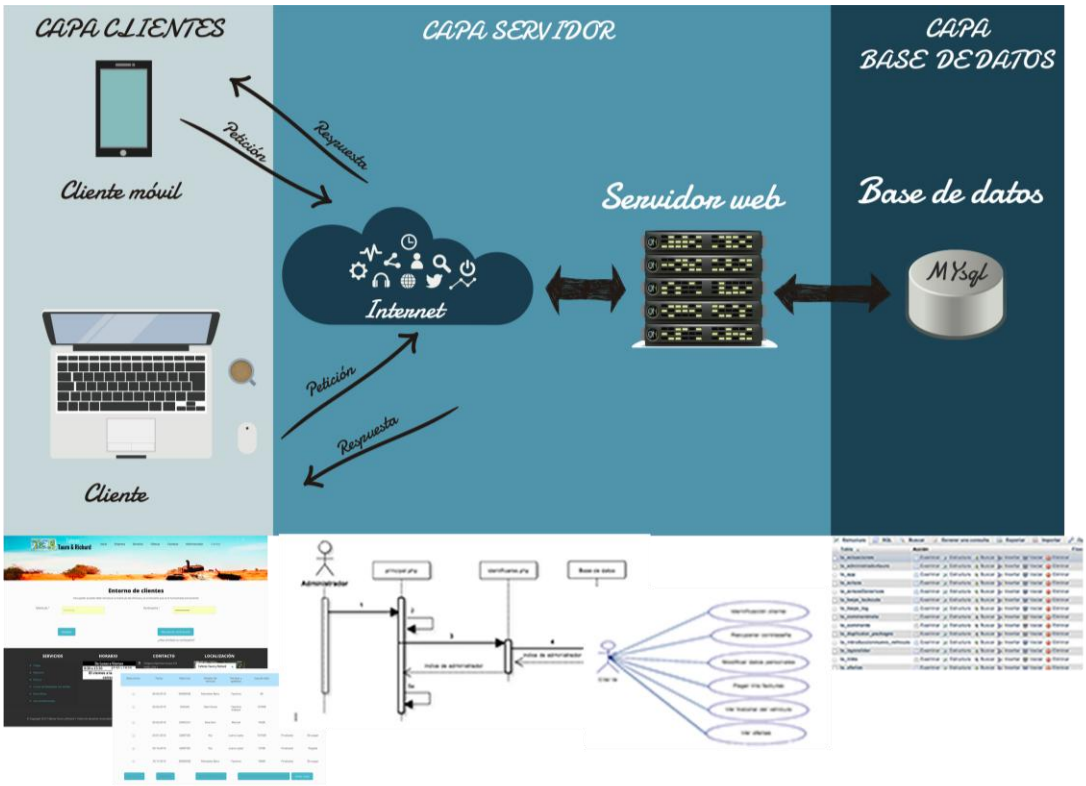


Ilustración 3. - Arquitectura de la aplicación web

En la capa de clientes se encuentran todas las máquinas cliente de la aplicación. Los clientes no contienen nada de la lógica de negocio de la aplicación y realizan una serie de peticiones al servidor. El servidor es el elemento central de la aplicación, ya que incorpora toda la lógica de negocio y el servidor de datos contiene todos los datos.

Debido a esto, las máquinas cliente solamente contendrán el software necesario para que el usuario pueda invocar la funcionalidad del servidor y pueda visualizar el resultado de la ejecución. Debido a que se ha utilizado el protocolo HTTP, este software será un navegador.

La arquitectura en tres niveles permite:

- Un mayor grado de flexibilidad.
- Mayor seguridad, ya que la seguridad se puede definir independientemente para cada servicio y en cada nivel.
- Mejor rendimiento, ya que las tareas se comparten entre servidores.

La aplicación web se ha diseñado con una arquitectura en la que relacionan varias tecnologías y lenguajes de programación, como son PHP, JavaScript, HTML, CSS, MySQL y AJAX. Esta arquitectura posibilita un acceso eficaz y rápido a los datos guardados en la base de datos.

2.4.2. Arquitectura aplicación móvil

Para el desarrollo de la aplicación móvil se va a utilizar el lenguaje de programación Java de Android. Esta aplicación no tiene una base de datos individual, si no que utiliza la misma base de datos utilizada en la aplicación web. La conexión de la aplicación móvil con el servidor web y el servidor de datos (base de datos) es mediante servicios Web.

Para el envío de avisos entre la aplicación web y la aplicación móvil se realiza mediante notificaciones push. Las notificaciones push son avisos que llegan al dispositivo cuando se produce un evento en el servidor web. De esta manera, se evita tener un servicio que esté periódicamente conectándose al servidor para realizar comprobaciones. Para ello Google proporciona el servicio Google Cloud Messaging (GCM) para realizar dichas notificaciones. Es por esto, que el dispositivo deberá registrarse en el GCM, como se indica en la Ilustración 4, para posteriormente poder recibir las notificaciones push. Una vez registrado el dispositivo móvil en el GCM, la aplicación móvil deberá enviarle al servidor web el número de registro para que, posteriormente, cuando el servidor web mande un aviso al GCM le indique a los dispositivos que desea mandar dicho aviso.

La conexión de la aplicación web con el servidor GCM se realiza mediante la conexión HTTP POST con Curl. Curl es una biblioteca que permite conectarse y comunicarse con diferentes tipos de servidores y diferentes tipos de protocolos. Cada vez que el administrador, a través de la aplicación web, desee enviar una notificación a los dispositivos móviles, el servidor web se conectará con GCM y le dirá a que dispositivos debe enviarlos. Por último, se configura la App móvil para poder recibir las notificaciones push.

En la siguiente imagen se refleja la arquitectura de la aplicación móvil, así como la interacción entre todos sus elementos.

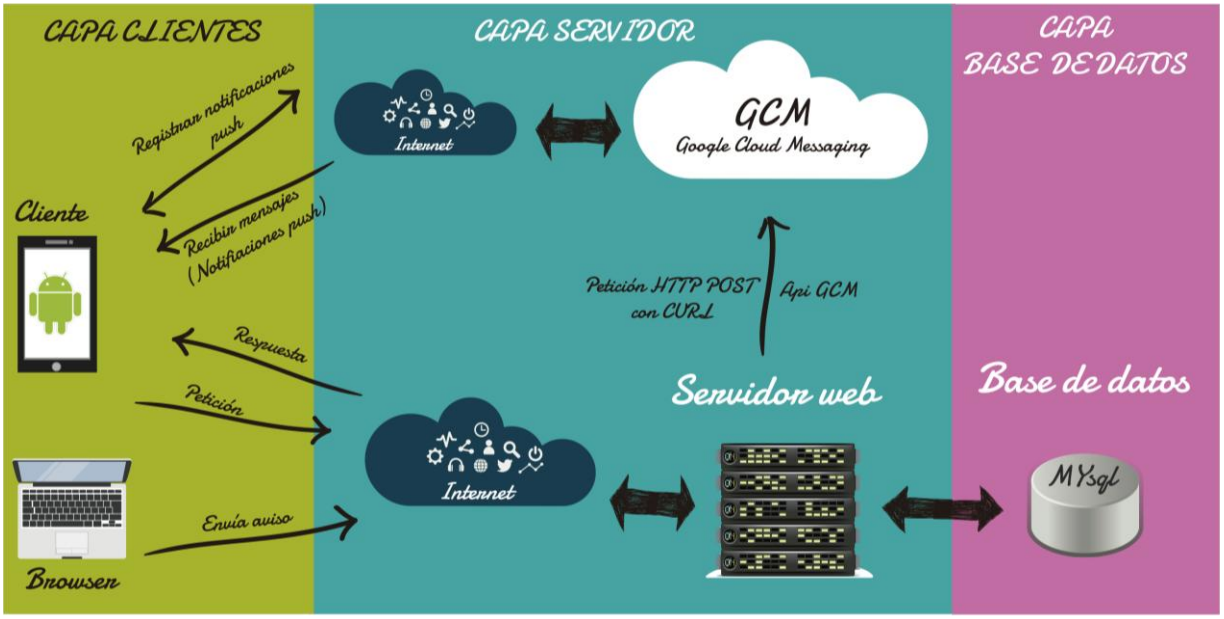


Ilustración 4. - Arquitectura App móvil

2.5. Herramientas utilizadas

En este apartado, se van a recoger todas aquellas tecnologías, lenguajes, herramientas y programas utilizados para la implementación del proyecto.

2.5.1. Lenguajes de programación

PHP (Hypertext Pre-Processor) : "*Es un lenguaje de programación de uso general de código del lado del servidor, originalmente diseñado para el desarrollo web de contenido dinámico*". Fue creado originalmente por Rasmus Lerdorf en 1995, sin embargo, la implementación principal de PHP es producida ahora por "The PHP Group". Publicado bajo la PHP License, la Free Software Foundation considera esta licencia como software libre. Este lenguaje está diseñado especialmente para el desarrollo web y puede ser incluido dentro de código HTML. Generalmente se ejecuta en un servidor web, tomando el código en PHP como su entrada y creando páginas web como salida. [1]

Lenguaje JavaScript : JavaScript es un lenguaje interpretado, utilizado principalmente en páginas web, con una sintaxis semejante a la del lenguaje Java. Sin embargo, al contrario que Java, JavaScript no es un lenguaje orientado a objetos propiamente dicho, ya que no dispone de herencia. Es más bien un lenguaje basado en prototipos, ya que las nuevas clases se generan clonando las clases base (prototipos) y extendiendo su funcionalidad. [2]

JAVA: "*Es un lenguaje de programación de propósito general, concurrente, orientado a objetos que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible*." Este lenguaje fue desarrollado en sus inicios por James Gosling de Sun Microsystems (la cual fue adquirida por la compañía Oracle).

La sintaxis de java procede en gran parte de C y C++, pero con menos utilidades de bajo nivel que ellos. Las aplicaciones de Java son normalmente compiladas a bytecode (clase Java). Esto significa que puede ejecutarse en cualquier máquina virtual Java (JVM), no teniendo importancia la arquitectura de la computadora subyacente.[3]

2.5.2. Tecnologías

HTML (HyperText Markup Language) : Lenguaje de Marcas de Hipertexto. "*Hace referencia al lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia para la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, entre otros.*" [4]

XHTML (Extensible Hypertext Markup Language) : Lenguaje extensible de Marcado de Hipertexto. "*Es básicamente HTML expresado como XML válido. Es más estricto a nivel técnico, pero esto permite que posteriormente sea más fácil al hacer cambios o buscar errores entre otros*." [5]

AJAX : "*Acrónimo de Asynchronous JavaScript And XML (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications). Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, lo que significa aumentar la interactividad, velocidad y usabilidad en las aplicaciones*." [6]

Hoja de estilo CSS (Cascade Style Sheets): Lenguaje que describe cómo deben ser presentadas por pantalla u otros dispositivos de salida, las diferentes páginas de un Website. Permite la separación forma-contenido de los documentos escritos en HTML. El WWWC12 es el encargado de formular la especificación de las hojas de estilo, para que sirvan de estándar para los distintos navegadores. El

objetivo que se encuentra detrás del desarrollo de las CSS, es el de separar el contenido de un documento de su presentación. [7]

SQL : "*El lenguaje de consulta estructurado (SQL Structured Query Language) es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones sobre las mismas. Una de sus características es el manejo del álgebra y el cálculo relacional permitiendo lanzar consultas con el fin de recuperar de una forma sencilla información de interés de una base de datos, así como también hacer cambios sobre la misma*." [8]

MySQL: Sistema de gestión de base de datos, desarrollado bajo la filosofía del software libre.[9]

Base de Datos o BD: "*Una base de datos es una herramienta para recopilar y organizar información. En las bases de datos, se puede almacenar información sobre personas, productos, pedidos, o cualquier otra cosa. Muchas bases de datos empiezan siendo una lista en un programa de procesamiento de texto o en una hoja de cálculo.* " [10]

Servicios web: (en inglés, Web Service o Web Services) "*Es una tecnología que utiliza un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes de ordenadores como Internet. La interoperabilidad se consigue mediante la adopción de estándares abiertos*." [11]

Tecnología Push, o servidor push: "*Describe un estilo de comunicaciones sobre Internet donde la petición de una transacción se origina en el servidor.*" [12]

PhpMyAdmin: "*Es una herramienta escrita en PHP con la intención de manejar la administración de MySQL a través de páginas web, utilizando Internet. Actualmente puede crear y eliminar bases de datos, crear, eliminar y alterar tablas, borrar, editar y añadir campos, ejecutar cualquier sentencia SQL, administrar claves en campos, administrar privilegios, exportar datos en varios formatos y está disponible en 50 idiomas*." .[13]

Apache: "*Servidor web HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras. Apache presenta entre otras características altamente configurables, bases de datos de autenticación y negociado de contenido, pero fue criticado por la falta de una interfaz gráfica que ayude en su configuración*." [14]

E-mails: "*Servicio de red que permite a los usuarios enviar y recibir mensajes rápidamente mediante sistemas de comunicación electrónicos*." [15]

FTP (File Transfer Protocol): Protocolo de red para intercambiar archivos entre sistemas que están conectados a una red TCP (Transmission Control Protocol) y está basado en la arquitectura cliente-servidor. Se utiliza principalmente para descargar un archivo de un servidor o para subir un archivo a un servidor a través de Internet.[16]

Interfaz de usuario: Representa la forma y disposición visual de los elementos en pantalla (texto, imágenes, objetos gráficos) que posibilitan la interacción de un usuario en un programa o aplicación.[17]

Servidor: "*Es una aplicación en ejecución (software) capaz de atender las peticiones de un cliente y devolverle una respuesta en concordancia. Los servidores se pueden ejecutar en cualquier tipo de computadora, incluso en computadoras dedicadas a las cuales se les conoce individualmente como "el servidor". En la mayoría de los casos una misma computadora puede proveer múltiples servicios y tener varios servidores en funcionamiento.* " [18]

HTTP (Hypertext Transfer Protocol): Protocolo usado en cada transacción de la World Wide Web. Está orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor.[19]

HTTPS (Hypertext Transfer Protocol Secure): "*Más conocido por sus siglas HTTPS, es un protocolo de aplicación basado en el protocolo HTTP, destinado a la transferencia segura de datos de Hipertexto, es decir, es la versión segura de HTTP*." [20]

Web, WWW o World Wide Web: "*Es un sistema de distribución de documentos de hipertexto o hipermedios interconectados y accesibles vía Internet. Con un navegador web, un usuario visualiza sitios web compuestos de páginas web que pueden contener texto, imágenes, vídeos u otros contenidos multimedia, y navega a través de esas páginas usando hiperenlaces*." [21]

Navegador Firefox: Navegador muy útil para realizar proyectos, puesto que permite ir viendo lo implementado y facilita la labor de corrección, en caso de fallo, con su consola de errores. Ésta indica en que línea de código se encuentra el error y una breve descripción del mismo.[22]

Navegador Google Chrome: "*Es un navegador web desarrollado por Google y compilado con base en varios componentes e infraestructuras de desarrollo de aplicaciones (frameworks) de código abierto*". Navegador que nos permite ver si el estilo de web implementado es adecuado para más de un navegador o si por el contrario se modifica el diseño. [23]

Android: "*Es un sistema operativo basado en el kernel de Linux. Fue diseñado principalmente para dispositivos móviles con pantalla táctil, como teléfonos inteligentes o tablets; y también para relojes inteligentes, televisores y automóviles*." [24]

Desarrollo de software Android: "*Es el proceso por el que las nuevas aplicaciones se crean para el sistema operativo Android. Las aplicaciones se desarrollan por lo general en Java, lenguaje de programación utilizando el Android Software Development Kit (SDK), pero otros entornos de desarrollo también están disponibles*."[25]

2.5.3. Gestor de contenidos

WordPress: Es un gestor de contenidos o CMS (*Content Management System*) orientado a la estética, los estándares web y la usabilidad. WordPress es libre y, a su vez, gratuito.

Ha sido desarrollado en PHP para entornos que ejecuten MYSQL y Apache bajo licencia GPL y código modificable, y su fundador es Matt Mullenweg.

WordPress fue creado a partir del desaparecido b2/cafelog y se ha convertido en una de las plataformas de diseño web más populares de la red. Las causas de su enorme crecimiento son, entre otras, su licencia, su facilidad de uso y sus características como gestor de contenidos.[26]

2.5.4. Programas utilizados

Eclipse: Es un programa informático compuesto por un conjunto de herramientas de programación de código abierto multiplataforma para desarrollar lo que el proyecto llama "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores.

Eclipse fue desarrollado originalmente por IBM como el sucesor de su familia de herramientas para VisualAge. Eclipse es ahora desarrollado por la Fundación Eclipse, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto y un conjunto de productos complementarios, capacidades y servicios. [27]

Sublime: Es un editor de texto sofisticado para el código, marcado y prosa. Su interfaz es limpia e intuitiva y soporta el uso de snippets, plugins y sistemas de construcción de código (Build Systems).[28]

Microsoft Word: "*Microsoft Word es un software destinado al procesamiento de textos. Fue creado por la empresa Microsoft, y actualmente viene integrado en la suite ofimática Microsoft Office1*" [29]

Microsoft Power Point: "*Microsoft PowerPoint es un programa de presentación desarrollado por la empresa Microsoft para sistemas operativos Microsoft Windows y Mac OS, ampliamente usado en distintos campos como la enseñanza, negocios, etc. Según las cifras de Microsoft Corporation, cerca de 30 millones de presentaciones son realizadas con PowerPoint cada día. Forma parte de la suite Microsoft Office*."[30]

Filezilla: "*Es un cliente FTP multiplataforma de código abierto y software libre, licenciado bajo la Licencia Pública General de GNU*". Esta herramienta soporta los protocolos FTP, SFTP y FTP sobre SSL/TLS (FTPS).

En sus inicios, fue diseñado para actuar solamente en Microsoft Windows, pero desde la versión 3.0.0, gracias al uso de wxWidgets, es multiplataforma. Ahora está disponible para los siguientes sistemas operativos: GNU/Linux, FreeBSD y Mac OS X.[31]

Adobe photoshop: "*Es un editor de gráficos rasterizados desarrollado por Adobe Systems. Usado principalmente para el retoque de fotografías y gráficos*." Esta aplicación líder mundial del mercado de las aplicaciones de edición de imágenes.[32]

Adobe Illustrator (AI): "*Es un editor de gráficos vectoriales en forma de taller de arte que trabaja sobre un tablero de dibujo, conocido como «mesa de trabajo» y está destinado a la creación artística de dibujo y pintura para ilustración (ilustración como rama del arte digital aplicado a la ilustración técnica o el diseño gráfico, entre otros)*."

Adobe Illustrator tiene opciones creativas, un acceso más simple a los componentes y una gran versatilidad para generar rápidamente gráficos flexibles cuyos usos se dan en (maquetación-publicación) impresión, vídeo, publicación en la Web y dispositivos móviles. [33]

2.6. Gestión de riesgos / planes de contingencia

En toda realización de un proyecto hay ciertos riesgos que pueden entorpecer su correcta ejecución y, por consiguiente, retrasar la fecha de entrega. En la siguiente tabla se van a analizar dichos riesgos, con el fin de definir la ocurrencia y se van a estudiar las diferentes maneras de actuar, para compensar los contratiempos y minimizar el impacto en la realización del proyecto.

Problema	Prevención	Probabilidad	Plan de contingencia	Impacto
Mala estimación de los plazos de entrega.	Para evitar que esto suceda, se realiza una planificación con amplios plazos, dejando márgenes de tiempo para cada tarea.	Muy alta.	En caso de que se produjera este riesgo, habría que realizar una modificación de la planificación original inmediatamente, aumentando las horas de trabajo desde el momento hasta la entrega.	Alta. Lo más probable es que se cumpla, obligando a trabajar más los últimos días, pero sin llegar al extremo de no poder cumplir con la fecha de entrega.
Baja por accidente o enfermedad.	No existe mucha prevención posible para este riesgo, aparte de realizar una planificación con márgenes de tiempo entre tareas, como se ha dicho previamente.	Baja.	Se debería replantear la planificación y la fecha de entrega o buscar la manera de compensar las horas de trabajo.	Media. Este riesgo causaría un impacto de diferente magnitud dependiendo del tiempo que se esté enfermo y del trabajo asignado para esos días.
Posibles pérdidas parciales o totales de documentos.	Realizar copias de seguridad más a menudo y alojarlas en un disco duro virtual (DropBox, Google Drive…etc.) disponible y accesible desde cualquier terminal conectado a internet, o en un disco duro externo.	Media.	En caso de que se produjera este riesgo, se debería seguir trabajando con otro documento obtenido de alguna copia de seguridad.	Baja. No sería importante debido a que se realizan numerosas copias de seguridad, por lo que no afectaría casi en absoluto a la realización del proyecto.

Problema	Prevención	Probabilidad	Plan de contingencia	Impacto
Falta de conocimiento sobre este área de trabajo.	Formación y asesoramiento sobre este tema, mediante cursos online o clases presenciales.	Alta.	Realización de cursos para lograr un mayor conocimiento sobre el área a tratar.	Alta. La falta de conocimiento provocaría la mala realización del proyecto y gran retraso en su finalización.
Problemas en la utilización o implantación del software.	Utilización de software conocido y manejado previamente a la realización de este trabajo.	Alta.	Consulta de manuales, búsqueda de información en la red o solicitud de asistencia a personas especializadas en el ámbito.	Media. Este riesgo es casi seguro que se produzca, dando lugar a un retraso en los plazos de entrega.
Caída del servidor en el que está la web.	Utilización de un servidor bastante estable y utilizado con anterioridad.	Baja.	Cambio de servidor, restaurando una copia de seguridad de la página web.	Media. Este riesgo no tiene por qué ocurrir. Si se produjera, teniendo una copia de seguridad, no retrasaría en exceso el proyecto.
Utilización de herramientas de desarrollo inadecuadas.	Realizar una búsqueda exhaustiva de las herramientas idóneas para el desarrollo de una página web y una aplicación.	Baja.	Revisar tutoriales, documentación o pedir ayuda a personas expertas en el ámbito.	Media. Este riesgo es improbable que ocurra, pero en tal caso, conllevaría un atraso de la finalización del trabajo fin de grado.

Problema	Prevención	Probabilidad	Plan de contingencia	Impacto
Implementación errónea de alguna de las funcionalidades.	Realizar pruebas durante toda la fase de implementación.	Media.	Se debería revisar todos los módulos programados en busca de los errores.	Alta. Produciría un retraso importante del trabajo.
Diseño erróneo o incompleto de la base de datos.	Se debe realizar un modelo de dominio que se adapte a las necesidades de las funcionalidades de la aplicación.	Media.	Se debería incluir en la BD las tablas que faltan y revisar todas las funcionalidades de la aplicación.	Muy Alta. Retrasaría el trabajo porque se tendrían que rehacer todas las funcionalidades que interactúan con la BD.
Mala captura de requisitos.	Es necesario hacer las reuniones necesarias, hasta que se realice una buena captura de requisitos, para comenzar con el proyecto.	Alta.	Se tendría que volver a hacer reuniones con el cliente, para clarificar los requisitos del proyecto.	Alta. Retrasaría la ejecución del trabajo y los plazos de entrega.
Incompatibilidad de herramientas para la programación con el SO del ordenador.	Se debe realizar el proyecto con un ordenador que sea compatible con todas las herramientas con las que se va a desarrollar el proyecto.	Baja.	Habría que adquirir otro ordenador que fuera compatible con las herramientas de programación.	Media. No tendría un impacto muy grande si se dispone con facilidad de otro ordenador compatible.

Problema	Prevención	Probabilidad	Plan de contingencia	Impacto
Incompatibilidad del software de las cámaras web para poder insertarlo en la pagina web.	Asegurarse de que el software de las cámaras sea compatible con la aplicación.	Media.	Se debería adquirir otras cámaras y grabador compatibles para poder insertarlos en la web.	Alta. Si ocurriese no se podría realizar la funcionalidad de ver cámaras web desde la aplicación.
Incompatibilidad de versión Eclipse con Android SDK.	Se debe asegurar, previamente al desarrollo, la compatibilidad de la versión de eclipse con Android SDK.	Alta.	Se debería descargar la versión de eclipse compatible con Android SDK.	Muy baja. Si ocurriera, se solucionaría rápidamente descargando la versión compatible.
Versión de Android en el móvil anterior a la necesaria para ejecutar la aplicación.	Se debe descargar la última versión de Android en el dispositivo móvil que va a servir para probar la App móvil.	Muy baja.	Sería necesario descargar la última versión de Android en el terminal móvil.	Alta. No se podría probar la aplicación en el móvil hasta tener la nueva versión de Android.

Tabla 1. - Tabla de gestión de riesgos

2.7. Evaluación económica

Una vez establecido el coste temporal del proyecto en el apartado alcance del proyecto, se va a poner de manifiesto el cálculo de cual sería su coste económico.

2.7.1. Coste hardware

Para la elaboración del proyecto se ha adquirido un PC portátil Mac OSX 10.7.5 nuevo, cuyo coste ha sido de 1.300€. Esta herramienta no va a formar parte del producto final, puesto que no se le va a entregar al cliente. Es por ello que se va a realizar una estimación de la amortización, suponiendo que la vida útil que le queda, a pleno rendimiento y sin verse afectado, es de 5 años (60 meses).

Amortización: 1300 / 60 = 21,6 €/mes

El trabajo dura 4 meses y medio, por tanto: 4,5 x 21,6 =97,2 €.

Coste del hardware: 97,2 €.

2.7.2. Coste software

La aplicación está basada en la utilización del gestor de contenidos WordPress gratuito para que sea asequible para cualquier investigador, por lo que el coste de la plataforma es de 0€. Para la realización de la memoria y la posterior presentación se va a emplear el paquete Office, que viene ya con el ordenador. En cuanto al desarrollo de la aplicación Android y al desarrollo de los plugins de la aplicación web, se va a utilizar el Sublime Text editor, el Eclipse Indigo y SDK Android, por requerimiento de ciertas aplicaciones. Todas estas herramientas han tenido coste 0 €.

La aplicación está subida a un servidor de la nube, por tanto, hay que tener en cuenta el coste del servidor al año que serían 100 € y añadirle el coste del dominio que son 8 € / año.

Coste de software: 100 + 8 = 108 €/año -> 9 €/mes

El trabajo dura 4 meses y medio, por tanto, 4,5 x 9 = 40,5 €.

Coste del software: 40,5 €.

2.7.3. Coste personal

Como se ha observado en el apartado de planificación temporal, el proyecto se realizará en un periodo de tiempo de unos 4 meses y medio aproximadamente, eliminando los fines de semana. En este tiempo están incluidas las reuniones con el cliente.

En el apartado del alcance del proyecto se han definido las horas que durará el mismo. El número de horas dedicadas al proyecto serán 975 horas.

Considerando la categoría profesional de Ingeniero recién salido de la carrera, cuyo salario es de 15 € la hora, se obtendría el siguiente coste:

975 x 15 = 14.625 €.

Coste personal: 14.625 €.

2.7.4. Cálculo de los honorarios totales

En este punto, entran las contingencias acarreadas por los gastos que puedan ir apareciendo a lo largo del periodo de realización del proyecto (Impresión de la memoria, materiales necesarios, viajes a la universidad…). Teniendo en cuenta esto, estos serán los gastos extras durante la realización del proyecto:

Transporte: 150 €.

Papelería : 50 €.

Se le añaden los gastos comunes como es la luz (con su correspondiente partida de contingencia del 5% de los gastos comunitarios que serían aproximadamente 110€/mes):

Gastos comunes: 110x5%=5,5 €/mes x 4 meses y medio=24,75 €.

97,2 + 40,5 + 14.625 + 150 + 50 + 24,75 = 14.987,45 € $\cong$ 14.987 €.

Coste total del trabajo: 14.987 €.

2.7.5. Amortización de la herramienta

El taller se ahorraría una media de 10 min/reparación, debido al trabajo que evitaría la disposición de esta herramienta.

	Tiempo de la reparación	Tiempo para papeleo y gestión	Salario por reparación (60 min = 35€)
Sin herramienta	60 min	20 min	40 min a facturar=23,33€
Con la nueva herramienta de gestión	60 min	10 min	50 min a facturar= 29,1€

Tabla 2. - Tabla de evaluación económica

Por ende, se puede concluir que serían 5,77 € (29,1€ – 23,33€) lo que ganaría la empresa por reparación, con la incorporación de la nueva herramienta.

Suponiendo que se realizan una media de 8 reparaciones al día, de 1 hora cada reparación, serían 46,16 € a ganar por día.

¿Cuanto tiempo tiene que pasar para que se rentabilice el precio de la nueva herramienta de gestión?

14.987/ 46,16 = 325 días

Siendo aproximadamente 230 días de trabajo al año, 325 / 230 = 1,4 años ≅ 1 año y medio.

Se puede ratificar que la herramienta será rentable en 1 año y medio aproximadamente desde el momento de la compra.

3. ANÁLISIS DE ANTECEDENTES

En este capítulo se pretende dar a conocer la situación actual de la empresa, la razón por la que se ha decidido implantar una herramienta de las características mencionadas, así como explicar las diferentes alternativas existentes a dicho proyecto, sus ventajas e inconvenientes.

3.1. Historia

La pyme en la que se va a realizar la implantación de la aplicación web es la empresa llamada "Talleres Tauro y Richard".

Es una empresa fundada en 1.994 con tan solo 4 trabajadores que necesita una aplicación web para poder mostrar toda la información de la misma, así como mostrar a los clientes un seguimiento de sus vehículos y contar con la comodidad de no tener que preocuparse del coche, desde que lo depositan en el taller hasta que llega a su domicilio.

3.2. Situación actual

La empresa ha trabajado con otras consultoras informáticas que le han diseñado aplicaciones web estándares y no han quedado satisfechos porque no se ajustaban a sus necesidades.

Las aplicaciones presentadas al cliente han sido de carácter generalista, no se han adecuado a sus necesidades. Como consecuencia, si los administradores querían introducir los datos de un vehículo debían hacerlo introduciéndose en los entresijos del gestor de contenidos, por lo que el desconocimiento de la informática y el uso de un lenguaje excesivamente técnico han abrumado a las personas que gestionaban la aplicación.

Por consiguiente, el objetivo es desarrollar una web 2.0 para esta empresa y, a su vez, utilizar la web como canal de comunicación para generar feed-back con los clientes, adaptarse a los nuevos hábitos de consumo y demanda del mercado. Esto se va a lograr con los módulos nuevos, que permitirán al administrador de este negocio poder introducir los datos de las reparaciones desde la interfaz de la aplicación web, sin tener que acceder al backend del gestor. Igualmente, permitirá al taller ponerse en contacto con sus clientes a través de la App móvil para Android, que se va a desarrollar a lo largo de este trabajo.

3.3. Estudio de diferentes alternativas

Hoy en día, el número de herramientas de gestión para implantar en una empresa es muy amplio. Existe una cantidad inmensa de aplicaciones, algunas de ellas muy factibles y con facilidad de uso, mientras que algunas otras difíciles de gestionar, con interfaces poco amigables o con usabilidad escasa.

3.3.1. ERPs

Los sistemas de planificación de recursos empresariales (ERP, por sus siglas en inglés, Enterprise Resource Planning) "*Son sistemas de información gerenciales que integran y manejan muchos de los negocios asociados con las operaciones de producción y de los aspectos de distribución de una compañía en la producción de bienes o servicios*." [34]

Los sistemas ERP manejan la producción, logística, distribución, inventario, envíos, facturas y contabilidad de la compañía de forma modular. No obstante, este sistema puede realizar el control de muchas actividades de negocios como entregas, pagos, ventas, calidad de administración, producción, entre otros.

Las aplicaciones ERP son sistemas de gestión global para la empresa y están formados por diferentes módulos. Algunos de los módulos son: producción, ventas, compras, logística, contabilidad, gestión de proyectos, inventarios y control de almacenes, pedidos, nóminas, etc.

Los objetivos primordiales de los sistemas ERP son:

- Mejora de los procesos empresariales.
- Acceso a la información.
- Capacidad de compartir información entre todos los departamentos de la empresa.
- Supresión de datos y operaciones innecesarias de reingeniería.

El propósito fundamental de un ERP es suministrar apoyo a los trabajadores del negocio, un eficaz manejo de los datos e información que posibilite la toma de decisiones apropiada y descenso de los costos de operación, así como breves tiempos de respuesta a sus problemas.

En contrapartida, al ser esta empresa una pyme, todas las ventajas que ofrece la implantación de un ERP no son compensadas por las siguientes desventajas:

- La instalación del sistema ERP es muy costosa.

- El éxito de los ERPs depende de las aptitudes y la experiencia de la fuerza de trabajo, incluyendo la educación y cómo hacer que el sistema trabaje correctamente. Muchas empresas disminuyen costos mediante la reducción de preparación. Los propietarios de pequeñas empresas están menos cualificados, lo que denota una clara inexperiencia en el manejo del sistema ERP por ser gestionado por personal poco preparado.
- Los sistemas pueden ser difíciles de usar.
- Los vendedores del ERP pueden cobrar sumas de dinero para la renovación de sus licencias anuales, que no está vinculado ni con la dimensión del ERP de la compañía ni con sus ganancias.
- Cuando el sistema está implantado, las futuras modificaciones conllevan gastos muy altos.
- Los sistemas pueden tener demasiada ingeniería respecto a las necesidades reales de la empresa.

Muchas de las contrariedades que producen los ERPs en las empresas se deben a la incorrecta inversión para la educación constante del personal importante, abarcando las modificaciones de implementación y de prueba, y una carencia de políticas corporativas que perjudican a la forma de obtener los datos del ERP y a la manera de mantenerlos actualizados.

La implantación de los sistemas ERPs es complicada debido a la necesidad de una parametrización a medida para cada compañía, partiendo de la configuración inicial de la aplicación, que es común. Las personalizaciones y desarrollos específicos para cada empresa conllevan un gran esfuerzo en tiempo y dinero, para amoldar todos los procesos de negocio de la vida real en la aplicación.

Asimismo, analizando las ventajas e inconvenientes, tanto el aprendizaje de su utilización como el coste de su realización e implantación, la harían muy poco rentable. Es una herramienta orientada a grandes empresas, para cubrir todas sus necesidades, que puedan perfectamente acarrear con los gastos que supone la implantación de esta herramienta.

3.3.2. Gestores de contenidos

Para la implementación de esta aplicación web se va a utilizar el gestor de contenidos WordPress. En su lugar, se podría haber utilizado otro, como puede ser Joomla, Magento o Drupal. Sin embargo, estos gestores son de más difícil acceso a su código que WordPress, además de su complicada estructura e implantación.

Por todas las razones mencionadas previamente, se ha decidido utilizar el gestor de contenidos WordPress para la realización de la aplicación web, porque contiene todas las prestaciones

necesarias y cubre perfectamente los requisitos del proyecto. También se ha optado por realizar la programación de módulos en PHP, JavaScript y Ajax para aumentar su funcionalidad y cubrir todos los requisitos que debe tener el sistema. Estos lenguajes de programación son relativamente sencillos y de gran utilidad para el desarrollo de módulos y el aumento de funcionalidad de las páginas web.

3.3.3. Sistemas operativos para la aplicación móvil

A la hora de desarrollar la App móvil, se ha decidido que el sistema operativo, para el cual se debe implementar, debe ser Android por su carácter de libre distribución y el desarrollo que ha experimentado en tan poco tiempo, llegando a ser un contrincante fuerte en el mercado de los sistemas operativos móviles.

Existen otras alternativas a este sistema operativo. No obstante, con las explicaciones y razones comentadas a continuación, se ratifica que la mejor opción es el SO Android.

- IOS

El iOS es el sistema operativo utilizado por Apple en el iPhone. Este sistema se caracteriza por la simplicidad y usabilidad, proporcionando una experiencia gratificante a sus usuarios. El IPhone es uno de los terminales móviles más integrado entre la sociedad. Por esta razón, desarrollar una aplicación para esta plataforma supondría un gran alcance en la sociedad.

Aunque en un primer momento puede parecer que las propiedades de este sistema operativo son idóneas para la realización de la App, es justo a la hora de desarrollar cuando encontramos los principales inconvenientes:

El SDK está disponible para su descarga, pero es necesario tener un ordenador MAC para que éste funcione y estar registrado en la web de desarrolladores de Apple.

Se precisaría disponer de un IPhone para poder comprobar el funcionamiento de la aplicación.

La aplicación solo es posible probarla en el emulador que viene con el SDK. Si se desearía probarla en un iPhone o subirla al App Store para su distribución, sería necesario pagar una licencia de 99$ (aunque la aplicación fuera gratuita).

Los inconvenientes aquí expuestos son suficientes para descartar esta opción.

- Symbian

El sistema operativo Symbian es propiedad de Nokia. Es uno de los sistemas operativos más extendidos en todo el mundo. Esto se debe al auge que vivió Nokia en términos de ventas de terminales durante varios años en todo el mundo, consiguiendo así una inmensa distribución de su sistema operativo.

En cambio, tras la aparición de Android, muchas empresas fabricantes de móviles han creado nuevos dispositivos provistos con un hardware mejor que el de los móviles Symbian. A raíz de lo mencionado, Nokia está intentando equiparar su nueva versión de sistema operativo con los actuales estándares, pero aún no es capaz de ofrecer tantas prestaciones como lo hace Android.

- BlackBerry OS

RIM (Research In Motion) fue uno de los precursores en la incorporación del correo electrónico en un terminal móvil, el BlackBerry. Una de las características que destacan es el teclado físico de tipo QWERTY que es seña de identidad en todos los dispositivos móviles del fabricante. Este sistema operativo está enfocado al uso profesional como un gestor de emails y agenda.

BlackBerry OS posibilita a programadores independientes desarrollar sus aplicaciones, pero si éstas necesitarían tener acceso a ciertas funcionalidades con acceso limitado deberán ser asociadas a una cuenta de desarrollador de RIM.

El motivo de descarte de este sistema operativo ha sido por las razones que se acaban de mencionar y por ser un sistema operativo menos extendido en la sociedad.

4. CAPTURA DE REQUISITOS

La captura de requisitos es el primer paso para poder realizar una buena aplicación web. Este análisis podría considerarse como uno de los pasos más importante en el proceso de realización de una web, para una empresa. Un buen análisis va a permitir la elección de un gestor de contenidos que se adapte a los requerimientos actuales y futuros, aspecto muy importante para plantear el diseño de una manera u otra, de forma que las modificaciones futuras no supongan ningún tipo de problema.

4.1. Requisitos previos

En las conversaciones mantenidas con la empresa, se transmite el deseo de poder realizar una herramienta que cumplan los siguientes requisitos:

- Aplicación cercana al usuario, de fácil manejo tanto para clientes como administradores (La oficinista que actualiza la información del vehículo).
- Aplicación accesible desde cualquier lugar donde haya una conexión a internet. Se debe encontrar en "la nube".
- Aplicación web que se ajuste totalmente a la legalidad española, además de adaptarse perfectamente a los dispositivos móviles (Responsive design).
- Una herramienta en la que los clientes puedan identificarse y ver el estado del vehículo en cualquier momento, pagar el importe de la reparación a través de la web y recibir la factura. Igualmente, deberían poder modificar sus datos personales.
- Una plataforma para que los administradores del taller puedan gestionar todas las reparaciones, así como generar las facturas de las mismas.
- Disponer de un apartado en la aplicación web, para que los administradores del taller puedan visualizar las cámaras web del taller.
- Una web que se de a conocer en las redes sociales.
- Disponer de un gestor de avisos, en el que el administrador pueda introducir avisos, como finalización de la reparación, cambios de aceite, cambios de pastillas de frenos, pasar la ITV, entre otros.

- Aplicación móvil para que los clientes puedan recibir los avisos en sus móviles, puedan pedir cita previa, a través de la misma, o puedan consultar las revisiones necesarias de su vehículo.

4.2. Casos de uso

Una vez realizada la captura de requisitos funcionales de la aplicación a desarrollar, es necesario profundizar en alguna técnica que permita la detección de aquellas funciones generales que guardan una relación directa con esos requerimientos.

En esta sección, se van a exponer los casos de uso extraídos a partir de las especificaciones de usuario. En primer lugar, se describe la jerarquía de actores junto a una pequeña descripción de los actores implicados. A continuación, el modelo de casos de uso de la aplicación web y el de la aplicación móvil. Por último, aparecerá el modelo de dominio. Solo aparece un único modelo de dominio porque se utiliza la misma base de datos, tanto para la aplicación web como para la aplicación móvil.

Los casos de uso extendidos, tanto de la aplicación web como de la aplicación móvil, aparecen en el Anexo I. En ellos se detallan en profundidad los casos de uso, explicando la funcionalidad de cada uno de ellos.

4.2.1. Jerarquía de actores de la aplicación web

La aplicación web tiene 2 actores:

Usuario Administrador : Es el administrador o administradores de la empresa y pueden realizar todas las funciones del panel de administración, como son Introducir nuevas reparaciones, modificar datos de las reparaciones existentes, ver el historial de reparaciones, gestionar los servicios de mano de obra, ver cámaras web, gestionar las ofertas y gestionar los avisos.

Usuario cliente : El usuario cliente son todos los clientes de la aplicación, que mediante la matrícula de su vehículo y una contraseña, suministrada previamente, pueden acceder al entorno de clientes para modificar sus datos personales, ver el historial de reparaciones, ver todas las facturas de las reparaciones y pagar las facturas pendientes.

La siguiente imagen refleja la jerarquía de actores de la aplicación web:

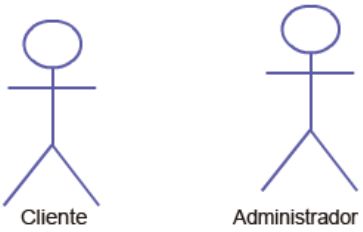


Ilustración 5. - Jerarquía de actores de la aplicación web

4.2.2. Jerarquía de actores de la aplicación móvil

La App Tauro y Richard tiene un actor :

Usuario cliente : El usuario cliente son todos los clientes de la aplicación que mediante el DNI y una contraseña, suministrada previamente, pueden acceder a la App móvil y van a poder pedir cita, ver las revisiones y, cuando la reparación haya finalizado, recibir el aviso directamente en su terminal móvil.

La siguiente imagen refleja la jerarquía de actores de la aplicación móvil:

Ilustración 6. - Jerarquía de actores de la App Tauro & Richard.

4.2.3. Modelo de casos de uso

Los casos de uso se emplean como una representación gráfica de las funcionalidades del sistema, para su correcta comprensión y asimilación. Un caso de uso representa una acción del sistema, por consiguiente, el modelo de Casos de Uso mostrará cómo debería interactuar el sistema con el usuario o con otro sistema, para conseguir su objetivo. También permite definir y representar los requerimientos de necesario cumplimiento por la aplicación. Se trata, por ende, de una representación de la secuencia de acciones entre un sistema y sus actores, respondiendo a un evento que inicia uno de los actores principales.

El uso de este tipo de diagramas está ampliamente propagado debido a la notación relajada de la que hace uso, resultando muy cercano para el usuario final.

4.2.3.1. Modelo de casos de uso de la aplicación Web

En este apartado se mostrará el modelo de casos de uso de la aplicación web.

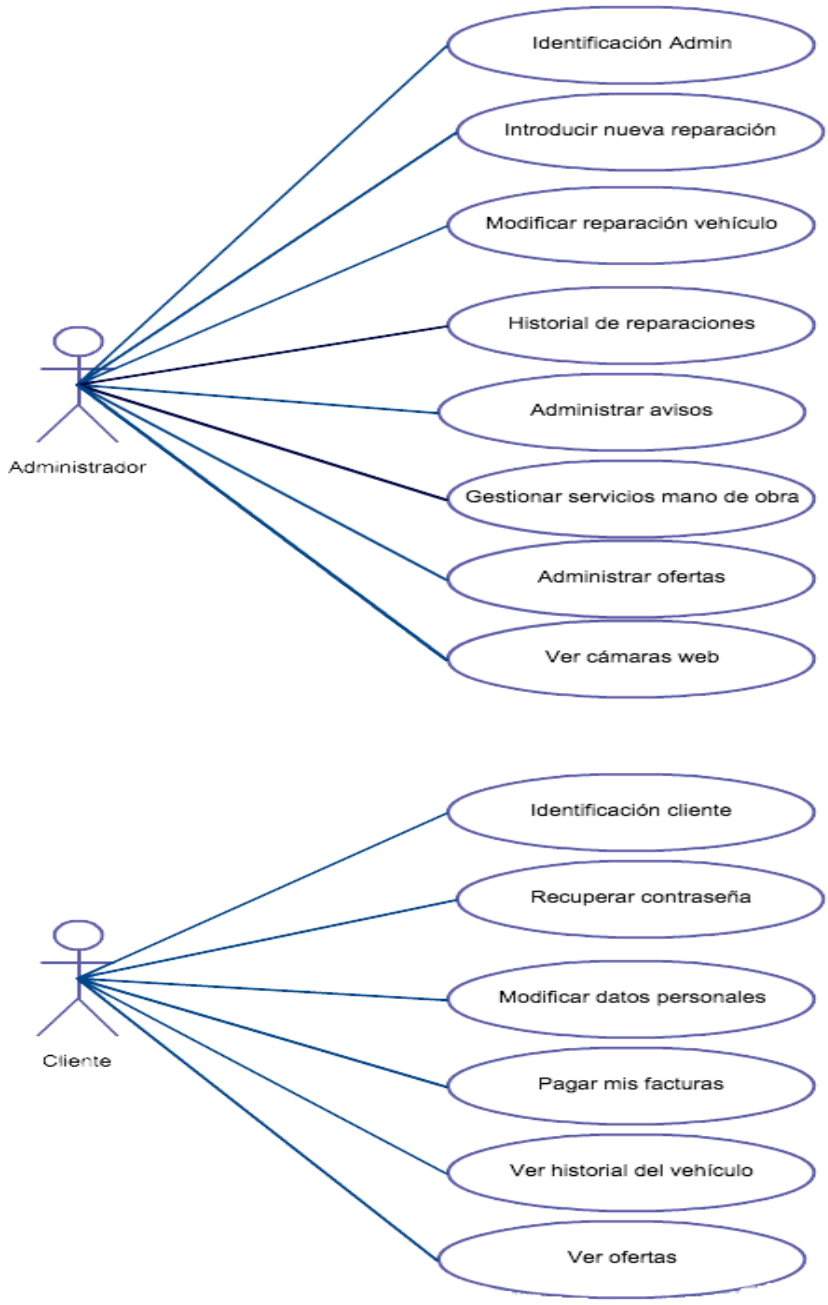


Ilustración 7. - Modelo de casos de uso de la aplicación web

A continuación, se va a dar una pequeña descripción de la función de cada caso de uso.

Caso de uso Identificación Admin: Este caso de uso servirá para que el administrador pueda acceder al panel de administración con una contraseña y gestionar todas las reparaciones.

Caso de uso Introducir nueva reparación: El administrador podrá introducir todos los datos relacionados con la reparación de un vehículo nuevo o que ya estuviera registrado en la web.

Caso de uso Modificar reparación de vehículo: El administrador podrá modificar, si así lo desea, cualquier dato relacionado con la reparación de un vehículo o con los datos del propio vehículo que previamente hayan sido insertados en el sistema.

Caso de uso Historial de reparaciones: El administrador podrá ver todo el historial de reparaciones realizadas en el taller. Podrá ver la factura, modificar los datos y eliminar la reparación o todo el historial de un vehículo.

Caso de uso Administrar avisos: El administrador podrá insertar o eliminar un aviso para el vehículo que desee. El aviso que se programe servirá para avisar a los clientes a través de la App móvil.

Caso de uso Gestionar servicios mano de obra: El administrador podrá gestionar los servicios de mano de obra. Podrá añadir, modificar o eliminar el que desee.

Caso de uso Administrar ofertas: El administrador podrá insertar, modificar o eliminar la oferta que desee. La oferta que inserte, automáticamente se insertará en el muro de Facebook del taller Tauro & Richard.

Caso de uso Ver cámaras web: El administrador podrá visualizar las cámaras web del taller desde la página web.

Caso de uso Identificación cliente: El cliente podrá acceder al menú del entorno de clientes identificándose en esta página mediante su matrícula y contraseña, suministrada previamente.

Caso de uso Recuperar contraseña: El cliente, en caso de que no recuerde la contraseña podrá recuperarla.

Caso de uso Modificar datos personales: El cliente podrá modificar cualquier dato personal o del vehículo que desee.

Caso de uso Pagar mis facturas: El cliente podrá ver los datos de las reparaciones que aún le faltan por abonar las facturas correspondientes y realizar el pago a través de la aplicación.

Caso de uso Ver historial del vehículo: El cliente podrá ver todo el historial de su vehículo, ver las facturas de las mismas, así como pagar las facturas pendientes.

Caso de uso Ver ofertas: El cliente podrá ver las ofertas introducidas por el administrador y que han sido publicadas en Facebook.

4.2.3.2. Modelo de casos de uso de la aplicación móvil

En este apartado se va a mostrar el modelo de casos de uso de la App móvil.

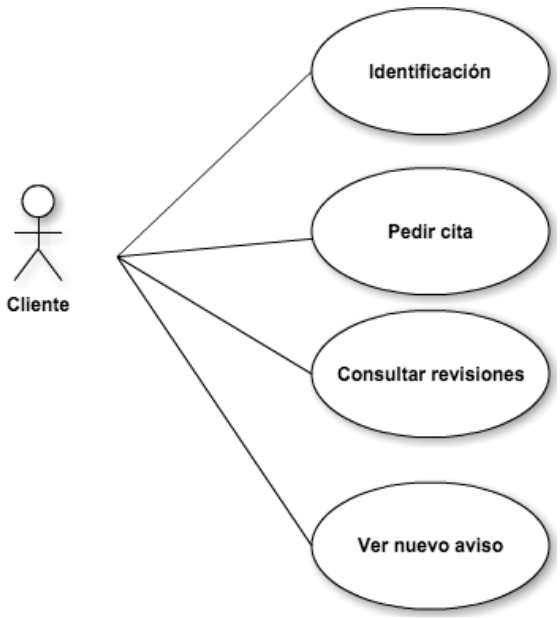


Ilustración 8. - Modelo de casos de uso de la App Tauro & Richard

A continuación se expondrá una breve explicación de cada caso de uso de la aplicación móvil.

Caso de uso Identificación: El cliente desde la App Tauro & Richard con su DNI y contraseña podrá acceder a la interfaz Menú del cliente.

Caso de uso Pedir cita: El cliente desde la aplicación móvil podrá pedir cita, eligiendo el día en el calendario e insertando el mensaje que desee.

Caso de uso Consultar revisiones: El cliente desde la App podrá consultar las próximas revisiones necesarias del vehículo.

Caso de uso Ver nuevo aviso: El cliente, que previamente se haya identificado al menos una vez en la aplicación móvil, podrá recibir notificaciones del taller Tauro & Richard directamente en su móvil.

4.3. Modelo de dominio

El modelo de dominio es un modelo conceptual de todos los temas relacionados con un problema específico. En él se describen las distintas entidades, sus atributos y sus relaciones entre ellas.

Se muestra, a continuación, el modelo de dominio obtenido tras la captura de requisitos:

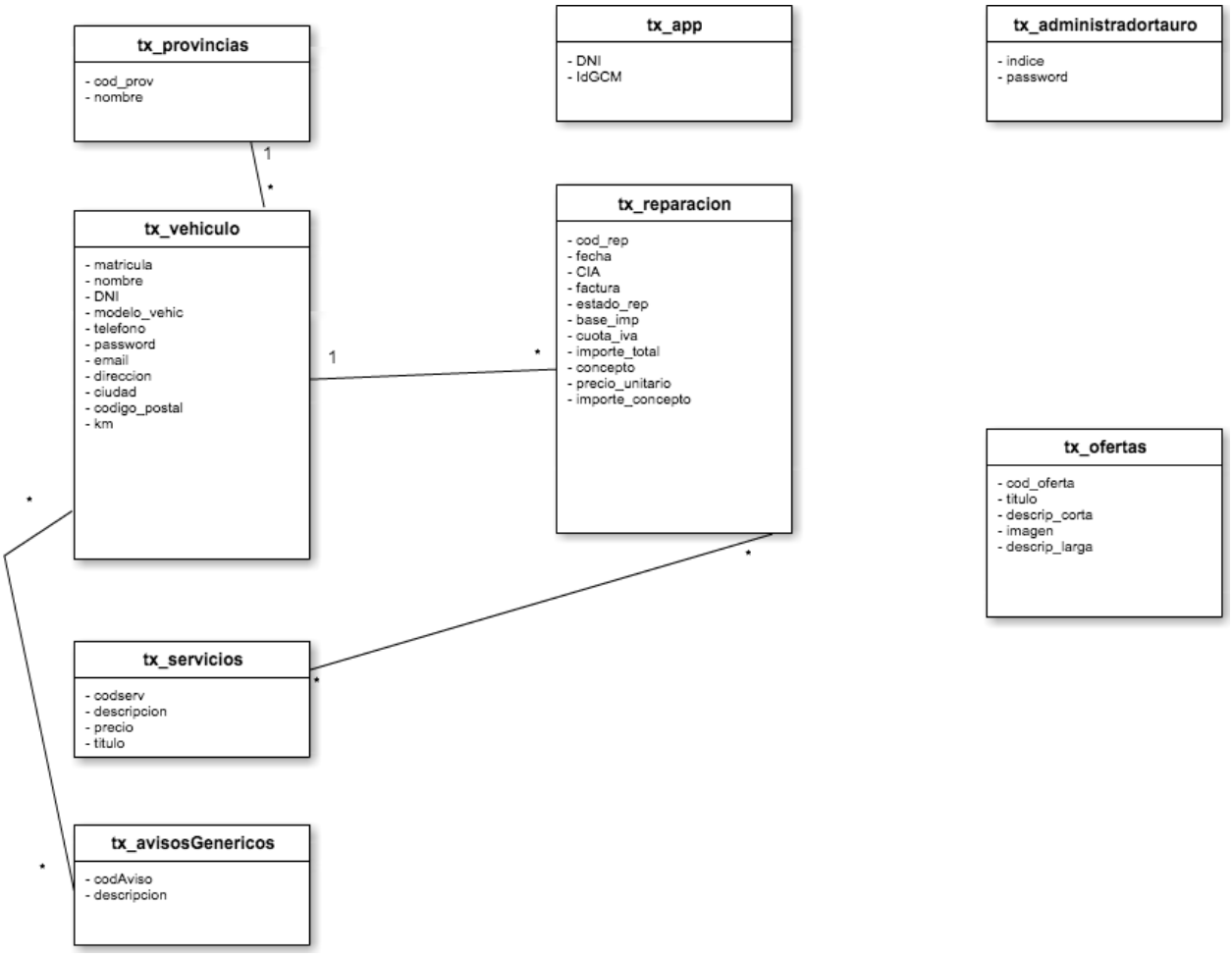


Ilustración 9. - Modelo de dominio

En este modelo de dominio se puede observar la creación de varias tablas y sus relaciones entre ellas:

tx_administradortauro: En esta entidad se guarda la clave del administrador, con la que el administrador podrá acceder al panel de administración.

tx_ofertas: En esta tabla se almacenarán todos los datos de las ofertas introducidas por el administrador en el sistema.

tx_provincias: Se guardarán los nombres de todas las provincias españolas junto con un código que se le asignará a cada una de ellas.

tx_vehiculo: En dicha entidad se van a recoger todos los datos del vehículo y del propietario del mismo.

tx_reparacion: En esta tabla se guardarán todos los datos de cada una de las reparaciones que se realizan en dicho taller.

tx_servicios: En esta entidad se van a almacenar los servicios de mano de obra, guardándose la descripción, precio y un código asociado a cada servicio.

tx_avisosGenéricos: En esta tabla se van a guardar los tipos de avisos que va a haber, almacenándose de cada uno de ellos, una descripción y un código asociado a cada aviso.

tx_app: Se va a almacenar el id que proporciona el GCM (Google Cloud Messaging) para asociar cada dispositivo móvil al DNI de la persona que pertenece.

La relación entre tx_provincias y tx_vehículo es de 1..* porque cada vehículo, que pertenece a una persona, está asociado a una única provincia y varias personas, dueñas de los vehículos, pueden pertenecer a la misma provincia.

La relación entre tx_vehiculo y tx_reparacion es de 1…* porque cada reparación pertenece a un único vehículo pero, sin embargo, un vehículo puede tener múltiples reparaciones.

La relación entre tx_reparacion y tx_servicios es *…* porque una reparación puede haber necesitado varios servicios de mano de obra y un mismo servicio de mano de obra puede haberse realizado en varias reparaciones.

La relación entre tx_avisosGenéricos y tx_vehiculo es de *…* porque a un vehículo se le pueden haber insertado varios avisos y a varios vehículos se les puede haber introducido el mismo aviso.

5. ANÁLISIS Y DISEÑO

La fase siguiente a la captura de requisitos, en un proyecto de desarrollo software, es la fase de análisis y diseño. En esta fase se van a desarrollar las pautas de funcionamiento del sistema, tomando como pilar fundamental su arquitectura, definida en el apartado 2, con el fin de definir un sistema lo suficientemente preciso como para permitir su construcción física (Apartado desarrollo).

Se analizan y diseñan todos los aspectos imprescindibles para el desarrollo de la aplicación, especificando el diseño de la base de datos y la estructura de la aplicación web y de la aplicación móvil.

Para una mejor comprensión de las funcionalidades de la aplicación web y de la aplicación móvil, se adjuntan los diagramas de secuencia de todos los casos de uso en el Anexo II.

5.1. Diseño de la base de datos

La generación de la base de datos se va a realizar mediante el modelo relacional. Este modelo presenta un grupo de datos que están guardados en tablas, con sus diferentes atributos en forma de columnas. Entre dichas tablas se van a crear unas relaciones para poder operar con los datos de manera eficiente y segura. Para usar y gestionar una base de datos relacional se usa el lenguaje estándar de programación SQL.

El gestor de bases de datos utilizado en la aplicación es MySQL. Es un sistema gestor de bases de datos OpenSource, es decir, la licencia de este gestor es gratuita.

En el momento de desarrollar este proyecto, se ha elegido esta SGBD por diferentes causas:

- La licencia libre.
- La flexibilidad del sistema en el momento de trabajar con diferentes lenguajes de programación.
- La facilidad y velocidad de manipulación y extracción de los datos almacenados.

La realización del diseño de las tablas de la base de datos de la aplicación se ha desarrollado con el programa phpMyAdmin. Es una herramienta gratuita de la misma empresa que la base de datos utilizada, que facilita la elaboración del diseño de la base de datos de manera visual, para posteriormente generar el código de manera automatizada. Esto permite una mayor flexibilidad y

velocidad a la hora de realizar cualquier tipo de modificación sobre el diseño original de la base de datos, durante la implementación de la aplicación.

Esta base de datos será utilizada tanto por la aplicación móvil como por la aplicación web.

5.1.1. Modelo relacional

Para la realización del diseño de la base de datos se han considerado las entidades lógicas indispensables para el adecuado almacenamiento de la información. A continuación, se muestran las diferentes tablas con sus atributos y claves que describen con detalle el diseño de la base de datos:

tx_vehículo (matrícula, nombre, DNI, modelo_vehic, teléfono, password, email, dirección, ciudad, codigo_postal, cod_prov, km)

tx_reparación (cod_rep, matrícula, fecha, CIA, factura, estado_rep, base_imp, cuota_iva, importe_total, concepto, precio_unitario, importe_concepto)

tx_provincias (cod_prov, nombre)

tx_servicios (codserv, descripción, precio, titulo)

tx_actuaciones (codserv, cod_rep)

tx_avisosGenéricos (codAviso, descripción)

tx_avisos (id, codAviso, matricula, km, kmanterior, fecha, fechaprox)

tx_app (DNI, IdGCM)

tx_administradortauro (índice, password)

tx_ofertas (cod_oferta, titulo, descrip_corta, descrip_larga, imagen)

5.1.2. Descripción detallada de la base de datos

En este apartado se va a comentar cada aspecto del modelo relacional, describiendo cada tabla con sus campos y sus tipos de datos.

- Tabla tx_vehículo

En dicha entidad se van a recoger todos los datos del vehículo y del propietario del mismo.

Se encontrarán campos como nombre, DNI, teléfono, email, dirección, ciudad, codigo_postal y cod_prov que serán todos los datos personales del propietario del vehículo. También se almacenará una contraseña para que el cliente pueda acceder con ella al entorno de clientes de la web y a la App móvil Tauro & Richard.

De la relación 1…*, que se ha visto en el modelo de dominio entre la tabla tx_provincias y tx_vehículo, se decide almacenar el cod_prov , que es el código de la provincia obtenido de la tabla tx_provincias y la clave primaria de la entidad, en la tabla tx_vehículo.

Referente al vehículo, se guarda el modelo del vehículo en el campo modelo_vehic, en el campo km se guardan los kilómetros del vehículo y, por último, en el campo matrícula se almacenarán las matrículas de los vehículos y servirá de identificador de esta tabla, porque nunca se pueden introducir tuplas con dos matrículas iguales.

- Tabla tx_reparación:

En esta tabla se guardarán todos los datos de cada una de las reparaciones que se realizan en dicho taller, cuyos datos serán necesarios para la realización de la factura.

De la relación 1…*, existente entre la tabla tx_vehículo y tx_reparacion, se decide que la clave de la tabla tx_vehículo, siendo ésta el campo matricula, será almacenada en la tabla tx_reparacion como clave extranjera y con la opción "ON DELETE CASCADE" que indicará que borrar una fila en la tabla referenciada (tx_vehículo), con un valor determinado de clave, implica borrar las filas de la tabla tx_reparación con el mismo valor de clave foránea.

Se almacena la fecha de la reparación en el campo fecha, en el campo CIA la compañía de seguros del vehículo, en el campo factura se almacenará si la factura ha sido pagada o no, siendo los valores de ésta "Pagada" o "Sin pagar". En el campo estado_rep se guardará el estado de la reparación del vehículo, pudiendo tomar el valor "Finalizada" o "Sin finalizar".

También se almacenan el concepto de la reparación, el precio unitario de cada parte de la reparación y el importe total de cada concepto en los campos concepto, precio_unitario y importe_concepto, respectivamente.

Por último, comentar que se guardan en los campos base_imp, cuota_iva y importe_total el importe de la reparación desglosado en base imponible (importe sin IVA), la cuota de IVA de la base imponible y el importe total con IVA para poder incluirlo en la factura de la reparación, respectivamente.

Cada reparación será identificada por un código de reparación que se almacenará en el campo cod_rep y será de tipo incremental.

- Tabla tx_provincias

Se guardarán los nombres de todas las provincias españolas en el campo nombre y vendrán identificadas por un índice llamado cod_prov, que será de tipo incremental.

- Tabla tx_servicios

En esta entidad se van a almacenar los servicios de mano de obra, guardándose el titulo, descripción y precio. Cada servicio será identificado por un código que se almacenará en el campo codserv.

- Tabla tx_actuaciones

De la relación *…* entre la tabla tx_servicios y tx_reparación surge una nueva tabla llamada tx_actuaciones que servirá para guardar los servicios utilizados por cada vehículo en reparación. Asimismo, se almacena como claves de dicha entidad, las claves primarias de cada una de las entidades que intervienen en la relación. Esto supone la creación de dos campos codserv y cod_rep, siendo el código del servicio y el código de reparación respectivamente, y siendo ambos la clave de la entidad.

- Tabla tx_avisosGenéricos

En esta tabla se van a guardar los tipos de avisos que va a haber, almacenándose de cada uno de ellos, una descripción del aviso y un código asociado a cada aviso, llamado codAviso. Este último servirá de identificador de la entidad.

- Tabla tx_avisos

De la relación *…* entre las tablas tx_vehículo y tx_avisosGenéricos se crea una nueva tabla llamada tx_avisos cuya clave principal de tipo incremental es el campo id. En esta tabla se almacenarán las relaciones entre las dos tablas mencionadas previamente. Por consiguiente, se van a almacenar las claves primarias de cada una de las tablas, siendo éstas el codAviso y matricula. Igualmente, se almacenarán la fecha en la que se introduce el aviso en el campo fecha y la fecha de la próxima revisión en el campo fechaprox. Además, se guardan los km que tiene el vehículo en el momento de introducción del aviso en el campo kmanterior y en el campo km se almacenarán los kilómetros con los que el vehículo debe hacerse la próxima revisión.

- Tabla tx_app

Se va a almacenar el id que proporciona el GCM (Google Cloud Messaging) en el campo IdGCM para asociar cada dispositivo al DNI de la persona que pertenece, guardándose este último en el campo DNI. La clave primaria de esta entidad es el DNI del cliente.

- Tabla tx_administradortauro

En esta entidad el campo password contiene la contraseña para acceder al panel de administración. Se utiliza el campo índice como clave de la entidad.

- Tabla tx_ofertas

Entidad lógica en la que se almacenarán todos los datos de las ofertas en los siguientes campos:

- Título: En el que se almacenará el título de la oferta.
- descrip_corta: Es la pequeña descripción debajo de la oferta que aparece en la página donde se encuentran todas las ofertas.
- descrip_larga: Es la descripción extensa que se muestra al entrar dentro de cada oferta.
- Imagen: Se guarda el path absoluto del lugar donde se encuentra almacenada la imagen en el servidor.

Cada oferta vendrá identificada unívocamente por un identificador (cod_oferta) que se generará de forma incremental al guardarse la oferta.

5.2. Estructura de la aplicación web

El objetivo del presente apartado pretende ofrecer una visión lo más generalizada posible de la aplicación web. Esta aplicación funcionará de forma autónoma sobre cualquier plataforma, valiéndose para ello de cualquier navegador web de actualidad y obviamente, de una conexión a Internet.

5.2.1. Diseño de los módulos desarrollados

Las tecnologías principales empleadas en la implementación de los módulos serán:

- Lenguaje PHP dado el carácter eminentemente dinámico.

- Plantillas CSS para optimizar la legibilidad de la información mostrada.
- AJAX para mejorar la usabilidad y velocidad de las aplicaciones, puesto que con la utilización de esta herramienta es posible realizar cambios sobre las páginas sin necesidad de recargarlas.
- JavaScript para moldear la forma en que se muestran algunas páginas web.
- MYSQL como sistema de gestión de base de datos.

Por tanto, se utilizará un servidor donde alojar la página web, que soporte dichas tecnologías.

Los módulos o plugins desarrollados se encuentran en dos carpetas llamadas "Administrar_datos_vehiculos" y " Pago_online_factura".

5.2.1.1. Módulo Administrar_datos_vehiculos

Este módulo es el panel de administración del administrador del sistema. A través del mismo se podrán insertar nuevas reparaciones al sistema, así como modificar sus datos, se podrán gestionar los servicios de mano de obra (Insertar, modificar y eliminar), se podrán gestionar las ofertas, los avisos, ver las cámaras web del taller y ver todo el historial de reparaciones del taller.

El módulo está compuesto por un archivo principal llamado "principal.php" y cada funcionalidad se encuentra dentro de su carpeta correspondiente.

Por ejemplo, la funcionalidad de introducir datos vehículo se encuentra dentro de la carpeta "introducir-datos-vehículo", de esta misma forma la funcionalidad gestionar servicios se encuentra dentro de la carpeta "gestionar-servicios". La única excepción es la funcionalidad de identificarse, que al formarse por un único archivo llamado "identificarse.php" se encuentra, junto a principal.php , en el directorio raíz del módulo.

En la siguiente imagen quedan reflejados todos los archivos y carpetas del módulo.

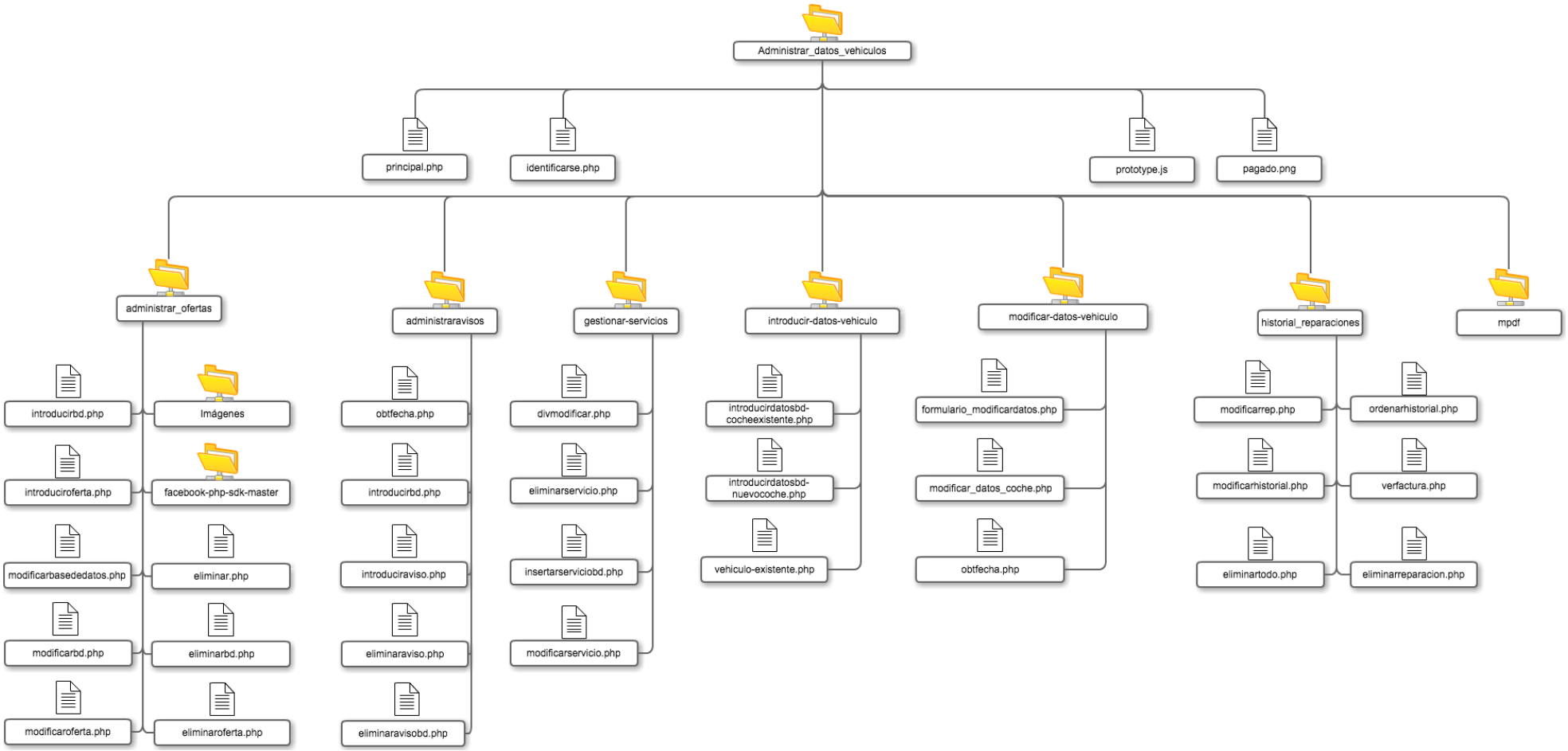


Ilustración 10. - Estructura del módulo Administrar datos vehículos

En el directorio principal se puede apreciar la existencia del archivo prototype.js. Este archivo se define así mismo como un framework, escrito en javascript, para desarrollar aplicaciones web dinámicas. Si bien, más que un framework, se trata de una librería que proporciona:

- Una extensión del lenguaje.
- Un acceso sencillo al árbol DOM, con independencia del navegador del cliente.
- Un soporte claro y sencillo de Ajax.
- Un soporte para JSON.

De todas las prestaciones que ofrece, la utilizada en este módulo es la de soporte de Ajax.

Por otro lado, se puede ver en el directorio principal una carpeta llamada mPDF. Esta carpeta está formada por archivos que se encargan de generar pdfs. Esta librería se va a utilizar para la generación de las facturas en pdf a través de la aplicación.

Por último, mencionar que en el directorio administrar_ofertas se encuentra la carpeta llamada "facebook-php-SDK-master". En dicha carpeta se encuentran los archivos necesarios para llevar a cabo la vinculación de la aplicación web con Facebook, para insertar ofertas tanto en la web como en Facebook.

5.2.1.2. Módulo Pago_online_factura

Este módulo será el entorno de clientes. Gracias a este módulo los clientes podrán identificarse en la aplicación, modificar sus datos personales, ver el historial de todas las reparaciones que han realizado en dicho taller y pagar las facturas pendientes, si así lo desean.

Está formado por un archivo principal llamado "índice.php" y cada funcionalidad se encuentra dentro de su carpeta correspondiente, como en el módulo anterior.

En la siguiente imagen quedan reflejados todos los archivos y carpetas del módulo.

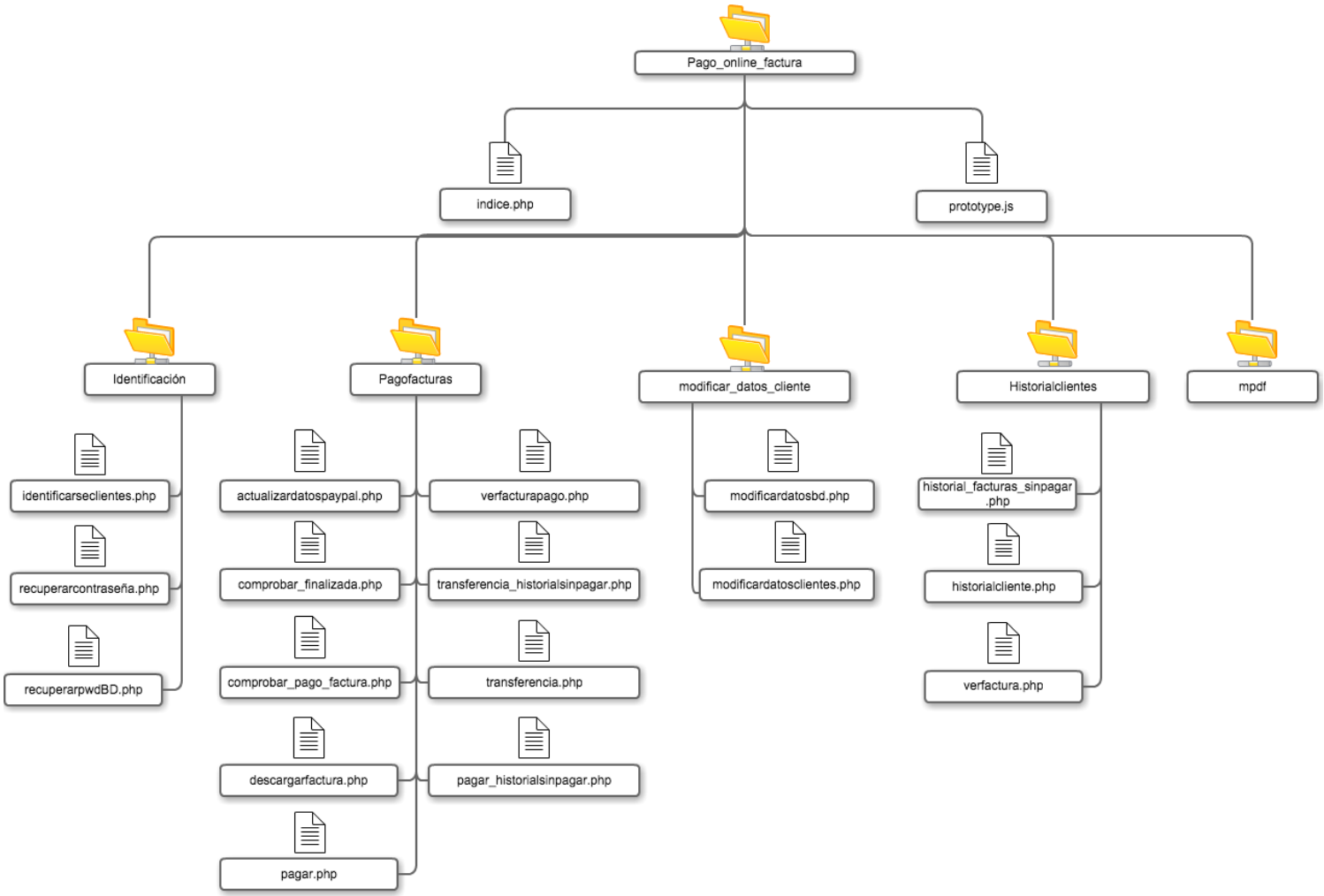


Ilustración 11. - Estructura del módulo pago online de la factura

Como se puede observar también utiliza la librería javascript.js y mPDF. Las utiliza por las mismas razones que el módulo anterior.

5.3. Estructura de la aplicación móvil

La aplicación móvil está creada en el lenguaje de programación Java, que es un lenguaje orientado a objetos. El sistema operativo para el que se va a desarrollar esta App es para Android.

5.3.1. Diagrama de clases

Este diagrama es un tipo de diagrama estático que describe la estructura de un sistema o aplicación mostrando sus clases, orientadas a objetos.

El diagrama de clases contiene información, como la relación entre un objeto y otro, la herencia de propiedades de otro objeto y conjuntos de operaciones que son implementadas para una interfaz gráfica.

Una clase define los atributos y métodos de una serie de objetos, de manera que todos los objetos de una clase determinada tienen los mismos atributos y el mismo comportamiento. Este tipo de diagramas también reflejan el concepto de herencia, esto es, la posibilidad de crear sub-clases o clases-hijas más especializadas, ya que pueden incorporar nuevos atributos y métodos, además de conservar los que heredan de la clase-padre.

La aplicación móvil en dicho diagrama debe mostrar el conjunto de objetos de dominio con las relaciones indicadas. Se ha creado una App bastante simple y esto se ha de ver reflejado en su lógica de negocio.

A continuación, se muestra el Diagrama de Clases (Ilustración 12) que representa el sistema objeto de estudio, indicando atributos y los métodos que lo conforman. En el apartado de desarrollo se entrará en mayor detalle en lo referente a la implementación de métodos y la forma que tienen de comunicarse unas clases con otras.

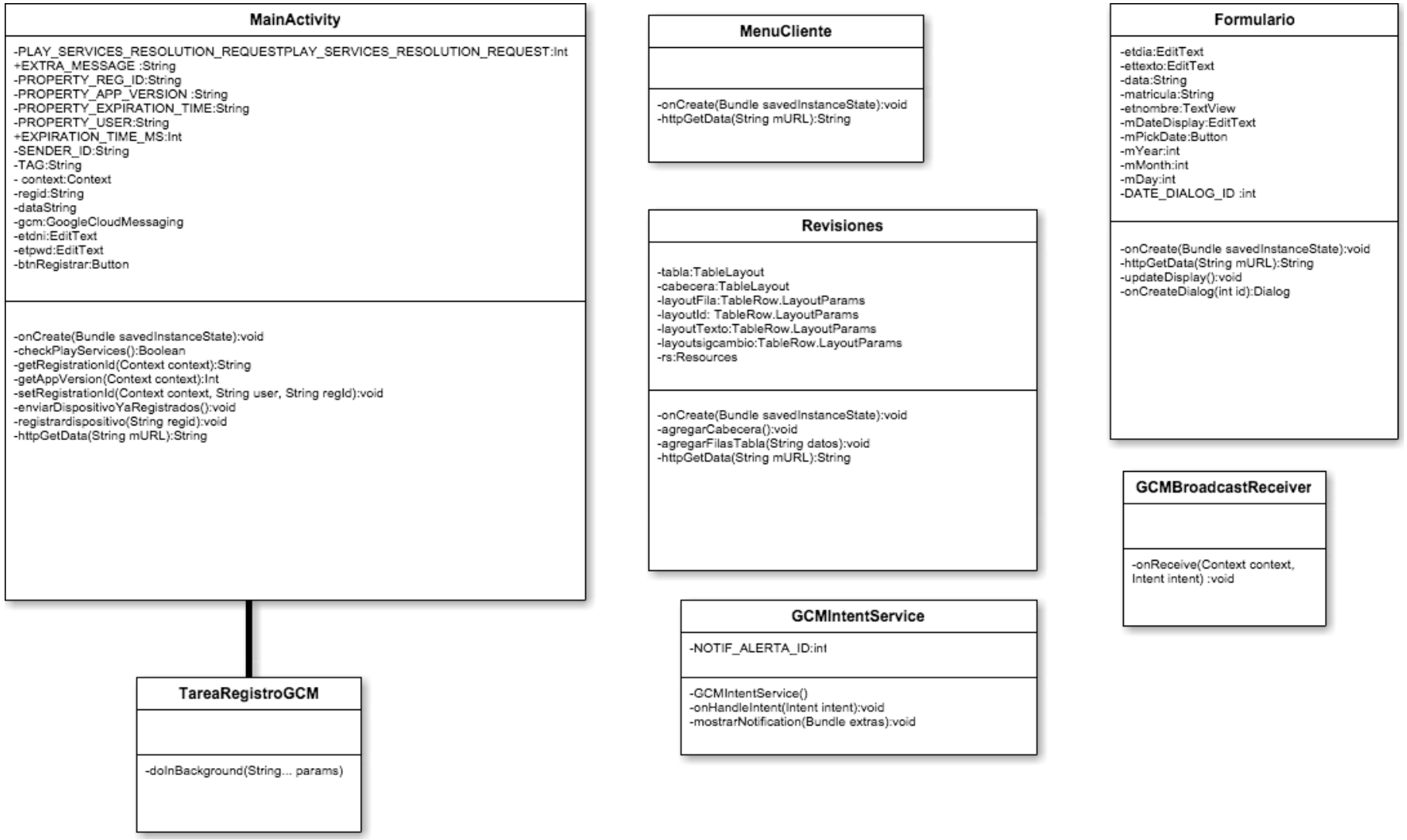


Ilustración 12. - Diagrama de clases de la aplicación móvil

La clase Activity es la clase base para todas las actividades. Por tanto, habrá algunas clases de Java que sean subclases de la misma.

Se pueden visualizar las clases como **MainActivity**, que es la clase que se ejecuta al poner en funcionamiento la aplicación. Al corresponder con la pantalla de identificación de la App contendrá métodos para asegurar la correcta identificación del usuario en el sistema. Además, dispone de los métodos necesarios para el correcto registro del dispositivo móvil en el GCM (Google Cloud Messaging), para más adelante poder recibir notificaciones desde la aplicación web en el terminal móvil. Esta clase es una subclase de Activity.java.

Para la realización del registro del dispositivo en el GCM se crea una clase en su interior, llamada TareaRegistroGCM, que extiende de AsyncTask. Esta clase corresponde a las tareas asíncronas. Todos los componentes de una aplicación se ejecutan en el mismo hilo de ejecución, llamado hilo principal. Es por esto que cualquier operación larga que se realice en este hilo bloquearía la ejecución del resto de componentes. Por tanto, las tareas asíncronas son las tareas que se realizan en segundo plano por su larga duración.

También se pueden encontrar clases como **MenuCliente** que es la clase que se ejecuta tras la identificación del cliente en la App. Contendrá métodos para crear el menú de la aplicación, además de las acciones que deben realizarse en caso de pulsar cualquiera de los botones del menú. Esta clase es descendiente de Activity.java.

Otra de las clases que se encuentran es la de **Revisiones**. Esta clase posee atributos y métodos con los que se creará una tabla dinámica con todas las revisiones necesarias del vehículo del cliente, identificado en la App. Estas revisiones se obtienen de la base de datos y es por esto que contendrá los métodos necesarios para conectarse con el servidor web y obtener los datos necesarios de la base de datos. Esta clase desciende de Activity.java.

La clase **Formulario** es subclase de Activity.java. Representa el apartado de Pedir cita de la App. A través de ella se generará un formulario con un campo para introducir la fecha de la cita, mediante un calendario que se genera dinámicamente, y otro campo para introducir el texto deseado. Mediante esta clase se enviarán mensajes de las citas a los administradores de la empresa.

La clase **GCMBroadcastReceiver** es la clase encargada de recibir los mensajes. Esta clase extenderá de WakefulBroadcastReceiver y tendrá el método necesario para procesar las notificaciones que reciba.

La clase **GCMIntentService** va a ser la encargada de procesar los mensajes recibidos. En este caso, esa tarea es crear una notificación como consecuencia del mensaje recibido. Esta clase extenderá de la clase IntentService.

6. DESARROLLO

En el capítulo anterior, las aplicaciones fueron organizadas en diferentes niveles y se puso de manifiesto el diseño de ambas, especificando la estructura de los archivos que lo forman, evitando entrar en detalles sobre tecnologías o recursos de implementación.

En el capítulo que nos ocupa, serán revisadas las tecnologías Web recurridas para implementar la aplicación, explicando con detalle la implementación de cada funcionalidad. Se expondrán algunas secciones de código que implementan las funciones o procesos que forman parte del nivel de Negocio. También se especificará con detalle el diseño e implementación de las interfaces de las aplicaciones. Por cuestiones obvias de extensión, únicamente se plasman aquéllas cuya relevancia sea destacada.

6.1. Implementación de la aplicación web

Para acceder a la aplicación es necesario abrir un navegador de Internet y teclear la siguiente url: http://proyecto.tauroyrichard.com/

6.1.1. Introducción

Los recursos de implementación o tecnologías webs a los que se ha recurrido son:

- Servidor de la nube para alojar la aplicación web con base de datos MySql 5, con PHP 5.6, Apache y PhpMyAdmin.
- Gestor de contenidos WordPress, para la creación de la web.
- Lenguaje PHP, para todo el apartado de programación de módulos y lógica de control.
- Plantillas CSS, con la finalidad de crear una interfaz moderna y atractiva acorde a las últimas tendencias.
- JavaScript, para la implementación de los módulos a desarrollar.
- MySql como base de datos. Para manejarla se ha utilizado PhpMyAdmin por su facilidad de uso y de interacción con la base de datos.

- Navegadores web (Mozilla Firefox, Google Chrome o Safari), para poder visualizar la aplicación web.
- Editor de texto Sublime Text Editor, para realizar la implementación de los módulos.
- FTP Filezilla, para el intercambio de archivos con el servidor.

6.1.2. Instalaciones previas

En primer lugar, se comienza realizando la instalación del gestor de contenidos WordPress en el servidor. Para ello se ha necesitado el paquete WordPress versión 3.8. y el theme a instalar, en este caso se llama Be Theme.

Los pasos a seguir son los siguientes:

- **Descargar WordPress:** Se procede a realizar la descarga de WordPress desde su sitio oficial en español.
- **Crear una base de datos MySQL :** Se crea una base de datos donde WordPress almacenará toda la información relacionada con la página web. Se realiza a través del servidor web donde está alojada la aplicación web.
- **Subir archivos al servidor:** Para subir los archivos al servidor se utiliza un protocolo especial llamado FTP (File Transfer Protocol o Protocolo de Transferencia de Archivos). Los datos de acceso FTP se obtienen en el panel de control del servidor.
- **Instalación de WordPress desde el navegador:** Se instala WordPress a través del navegador, siguiendo los pasos indicados.

Después de completar estos pasos, el WordPress ya estará instalado y se comienza a realizar el diseño y desarrollo de la aplicación web.

Para la transmisión de archivos entre el servidor y el ordenador se ha utilizado el FTP Filezilla. En la imagen de la izquierda, se puede apreciar la configuración de Filezilla para acceder al servidor y en la de la derecha, tras una correcta conexión con el servidor, aparecen las carpetas que componen la aplicación.

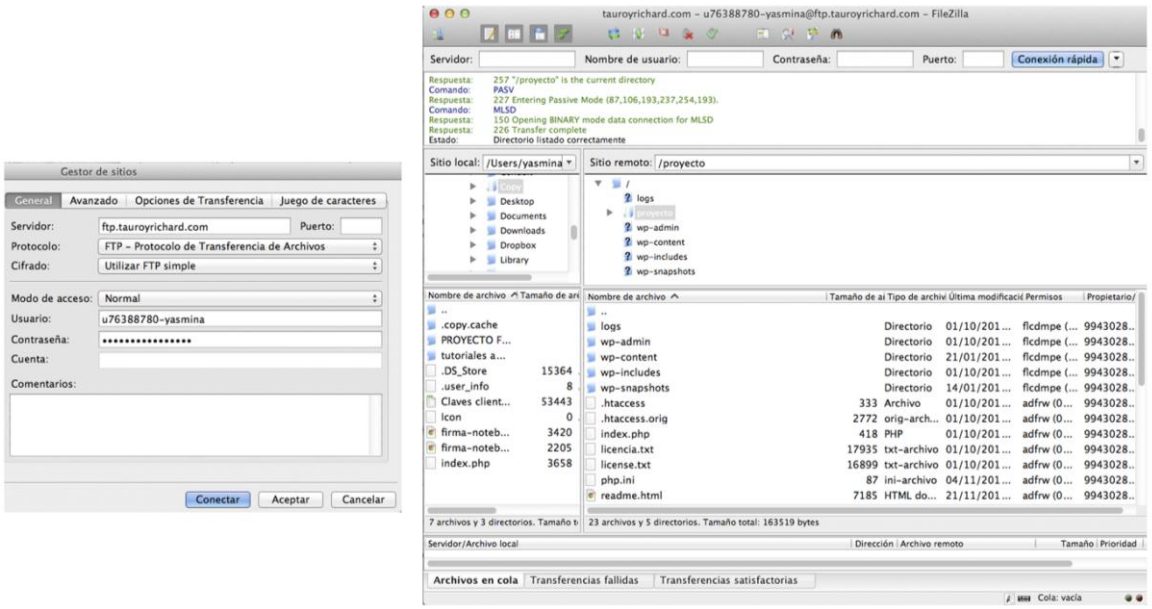


Ilustración 13. - FTP Filezilla

6.1.3. Gestor de contenidos WordPress

El hecho de que uno de los requerimientos no funcionales es el desarrollo de una aplicación web con WordPress significa que la arquitectura base del sistema sea la que utiliza WordPress.

WordPress es un CMS o un sistema de gestión de contenidos orientado a la generación de cualquier tipo de página web, especialmente, a la generación de blogs. Se ha desarrollado en PHP para entornos que ejecuten MySQL y Apache, bajo licencia GPL y código modificable.

La filosofía de WordPress se basa en la elegancia y la sencillez. Igualmente, se basa en las recomendaciones del W3C (World Wide Web Consortium), que es un consorcio internacional que produce recomendaciones para la World Wide Web. Al respetar este código, hace que la aplicación web sea compatible con todos los navegadores (Mozilla Firefox, Google Chrome, Internet Explorer, etc.).

Este CMS también genera pings automáticos, esto es, en cuanto se actualiza cualquier información en la página, WordPress emite un ping avisando a los buscadores. También mejora el posicionamiento del sitio web: permite tener URLs amigables, añadir etiquetas y categorías y utilizar plugins que ayuden a mejorar la visibilidad del sitio para los buscadores.

6.1.3.1. Estructura de WordPress

El diseño visual y la estructura del sitio web depende de un sistema de plantillas, independiente del contenido, que pueden tener varias alternativas de personalización. Los themes o plantillas, de las que hace uso WordPress, se componen de varios archivos a la vez para formar una página. Estos archivos son llamados por funciones determinadas de la plataforma, para hacerlo más eficiente. Estas funciones se asimilan a la función include que posee PHP y ayudan evitando tener que escribir la misma porción de código en varias plantillas.

Un ejemplo es la sección del encabezado del sitio. Este encabezado es igual para todas las páginas de un sitio web. Se compondrá de un logo y un menú que se desea que sea siempre igual. Si se construyera un Website en HTML, se debería copiar este código en cada una de las páginas. Sin embargo, en WordPress bastará con introducir el código del encabezado dentro del archivo header.php y luego hacer la llamada a dicho archivo en cada sección que se necesite. Igualmente se realizará para la barra lateral (sidebar.php) y el pie de página (footer.php).

En la siguiente imagen se refleja la estructura de WordPress:

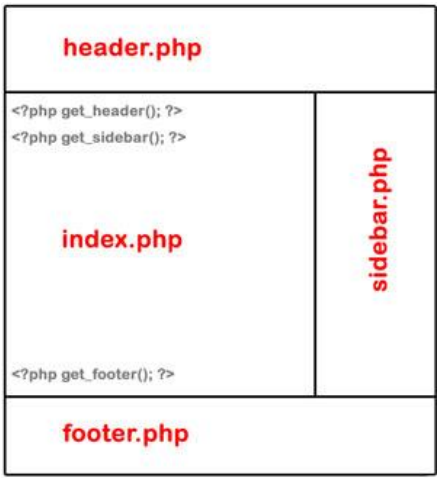


Ilustración 14. - Estructura de una página web [43]

De manera que cuando se carga una página web, realizada por WordPress, la página principal estará formada por los archivos index.php, header.php, sidebar.php y footer.php. El resto de páginas del sitio estarán compuestas por los archivos page.php, header.php, sidebar.php y footer.php. La única diferencia se encuentra en que en la página principal se carga el archivo index.php y en el resto de páginas el archivo page.php.

Para la realización de esta aplicación web, se ha escogido un theme y se ha comenzado con el diseño de la aplicación, introduciendo contenidos e imágenes. Se ha realizado un diseño elaborado de cada página del sitio, utilizando los recursos proporcionados por WordPress y por el theme escogido.

6.1.3.2. Los ganchos (Hooks) de WordPress

WordPress posee una funcionalidad, llamada Hooks o Ganchos, que le hace ser tan modular y flexible. Gracias a los hooks se podrá cambiar virtualmente cualquier aspecto del sitio, ya sea del contenido o del diseño.

Los ganchos o hooks forman parte de la manera en que WordPress está implementado. Su principal cometido es el de permitir transformar prácticamente cualquier aspecto de la plataforma, sin tener que tocar su código fuente original.

Los ganchos son una forma diferente de personalizar el diseño de un sitio realizado con WordPress. Éstos funcionan sobre todo por medio de PHP. Para el desarrollo de los módulos de la aplicación se han utilizado estos hooks, puesto que son una herramienta imprescindible para poder extender las funcionalidades de WordPress.

WordPress posee un hook llamado the_content. Este gancho se usa cada vez que se origina el contenido de una página, empleándose la función PHP del mismo nombre. Con este gancho se creará una función propia que añada texto adicional en el bloque de contenidos. Además, se sirve de la función condicional is_page, para que este código solo se ejecute en una página determinada.

En la siguiente imagen queda reflejado la manera en que WordPress carga las páginas, preguntando a cada momento en que página se encuentra, para saber qué tipo de archivos ha de ejecutar.

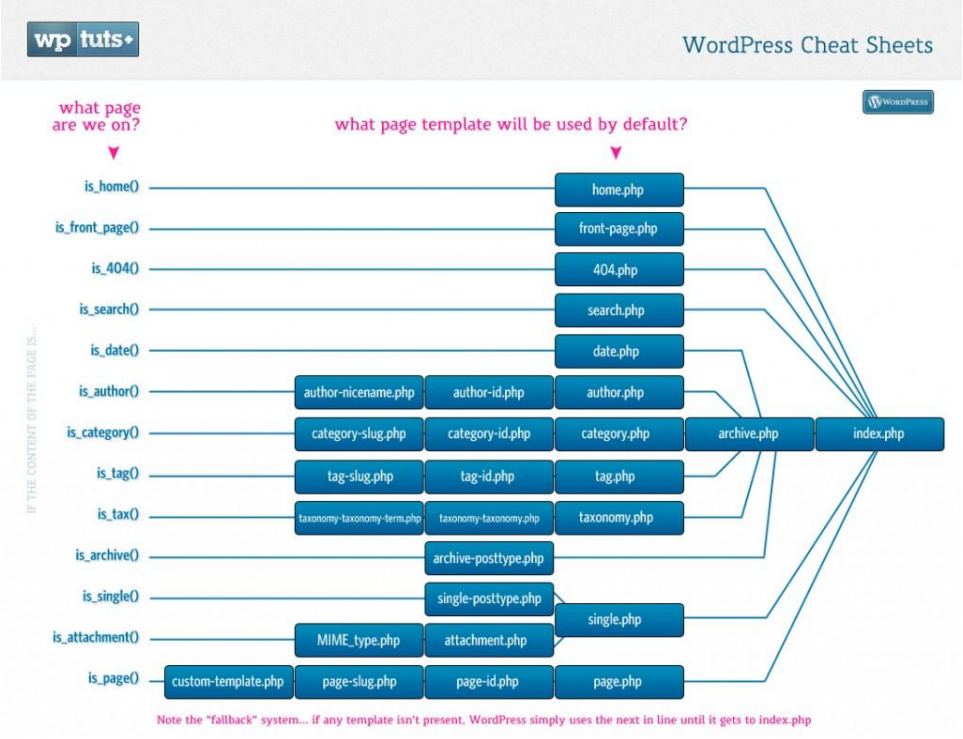


Ilustración 15. - Funcionamiento de WordPress [44]

6.1.4. Diseño e implementación de la aplicación web

El diseño de la aplicación responde a la necesidad de tener la información de manera clara y ordenada, y a su vez gozar de un aspecto trabajado y original. El diseño se realiza buscando constantemente la simplicidad en el uso de la aplicación, a la vez de llamar la atención del usuario.

6.1.4.1. Diseño de la interfaz gráfica

Tras la instalación de WordPress, se comienza con el diseño de la interfaz gráfica de la aplicación web, confeccionando la página principal y subsecciones, que son Empresa, Servicios, Ofertas y Contacto.

Toda la aplicación se controla desde un menú superior que siempre es visible, desde que el usuario entra en la aplicación. La estructura de la aplicación está dividida en bloques o módulos accesibles desde el menú, que hacen fácil e intuitiva su uso.

Para la implementación de la página web se emplean los recursos proporcionados por WordPress. En la imagen siguiente, se puede apreciar el menú de la parte de administración de WordPress, a través del cual se realiza la implementación de toda la interfaz gráfica de la web.

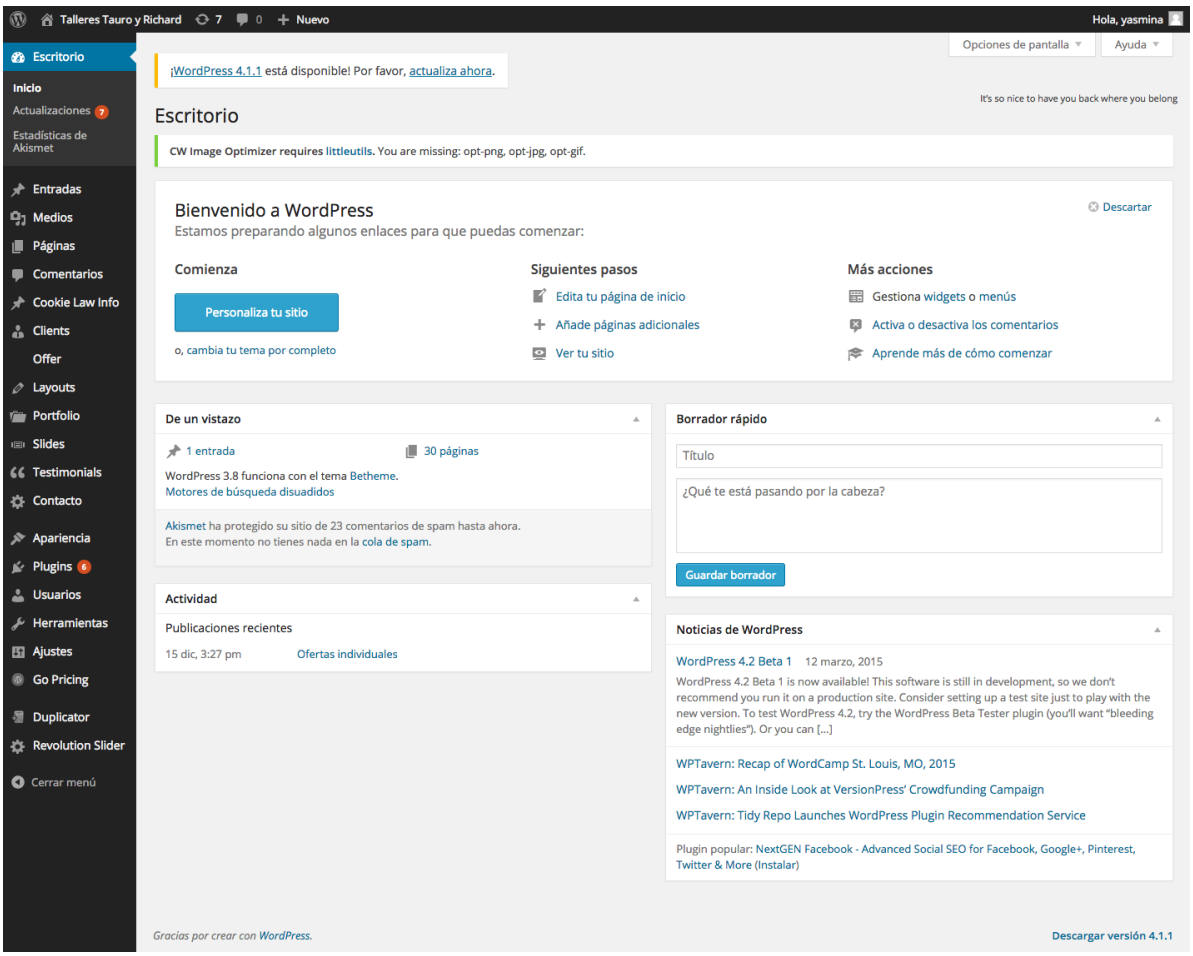


Ilustración 16. - Escritorio de WordPress

Cada página se elabora, a través del menú de administración de WordPress, en la opción Páginas. En la imagen siguiente, se puede observar que, la imagen de la izquierda, es la página a través de la cual se desarrolla el contenido y, la página de la derecha, es la página que el usuario final visualiza. De la misma manera se realizan el resto de páginas de la aplicación web.

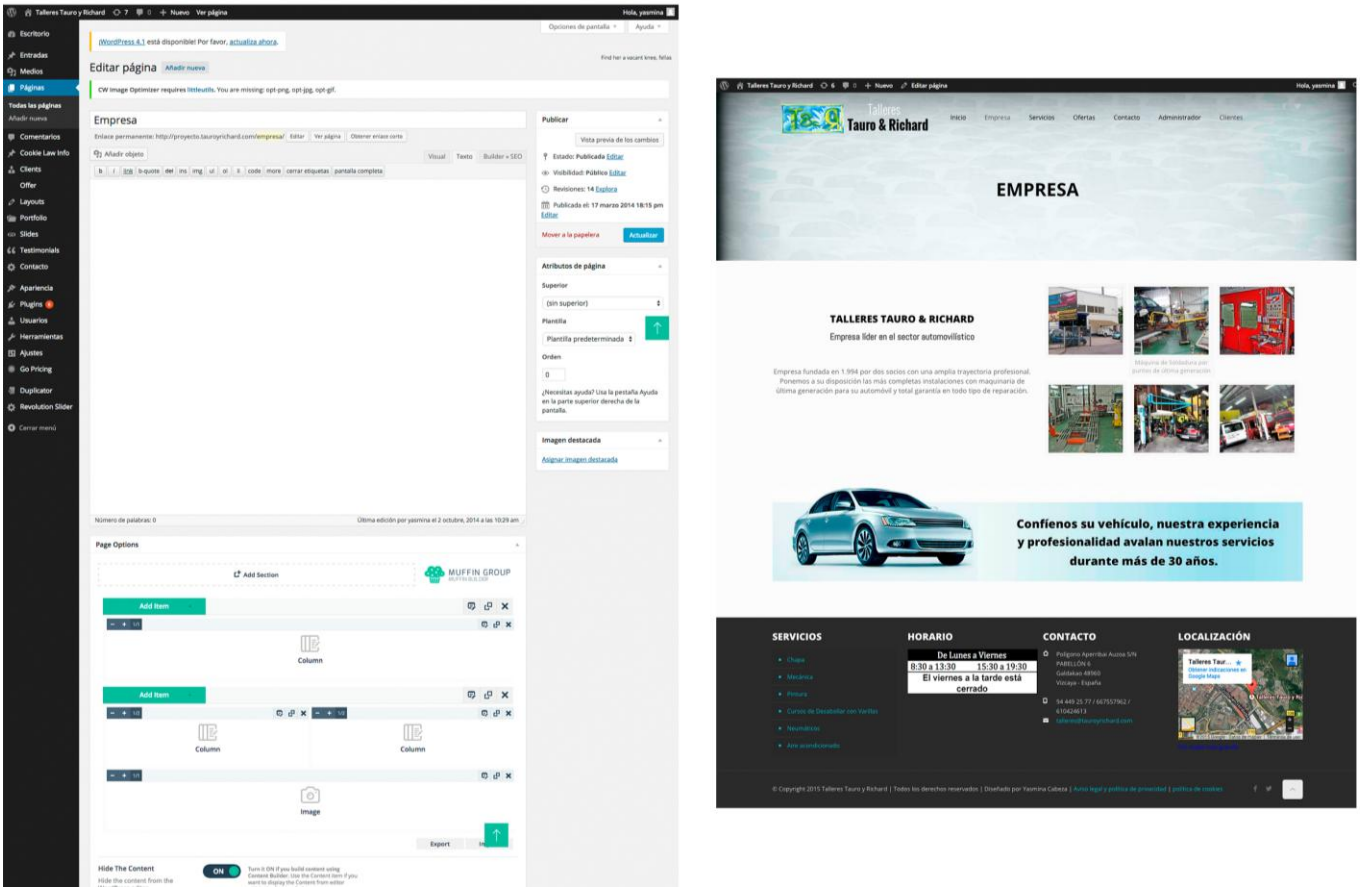


Ilustración 17. - Página Empresa de la aplicación web

La página web tiene una página principal (Ilustración 18) y en su parte superior posee el menú mediante el cual se navegará a través de la misma y que estará presente en todas las páginas de la aplicación. Igualmente, tiene un pie de página en el que se informa de los servicios que ofrece el taller, el horario, los datos de contacto y localización.

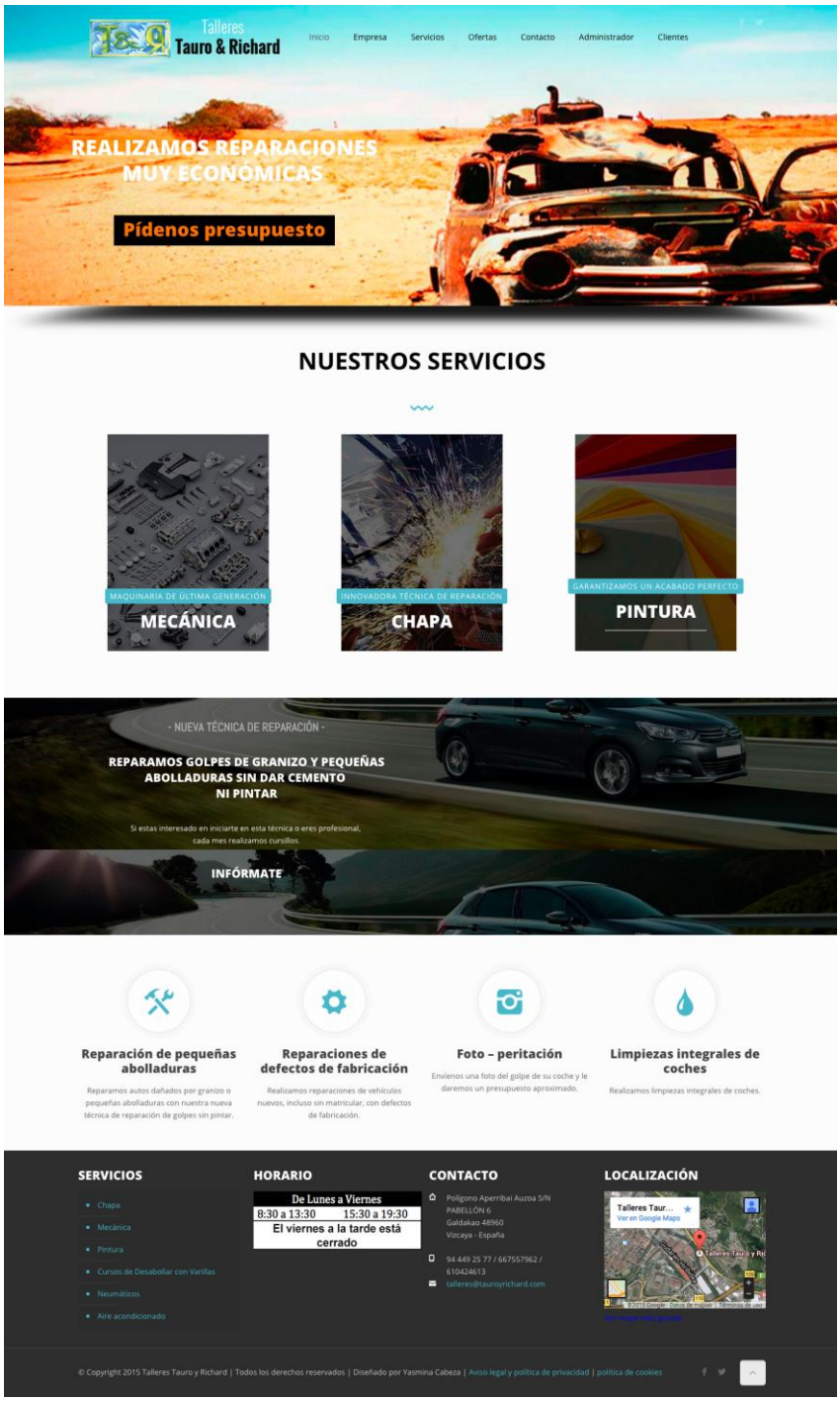


Ilustración 18. - Página principal de la aplicación

Esta aplicación web dispone de varios apartados explicando los diferentes servicios que ofrece el taller. A continuación, se muestran algunas de las páginas de la aplicación.

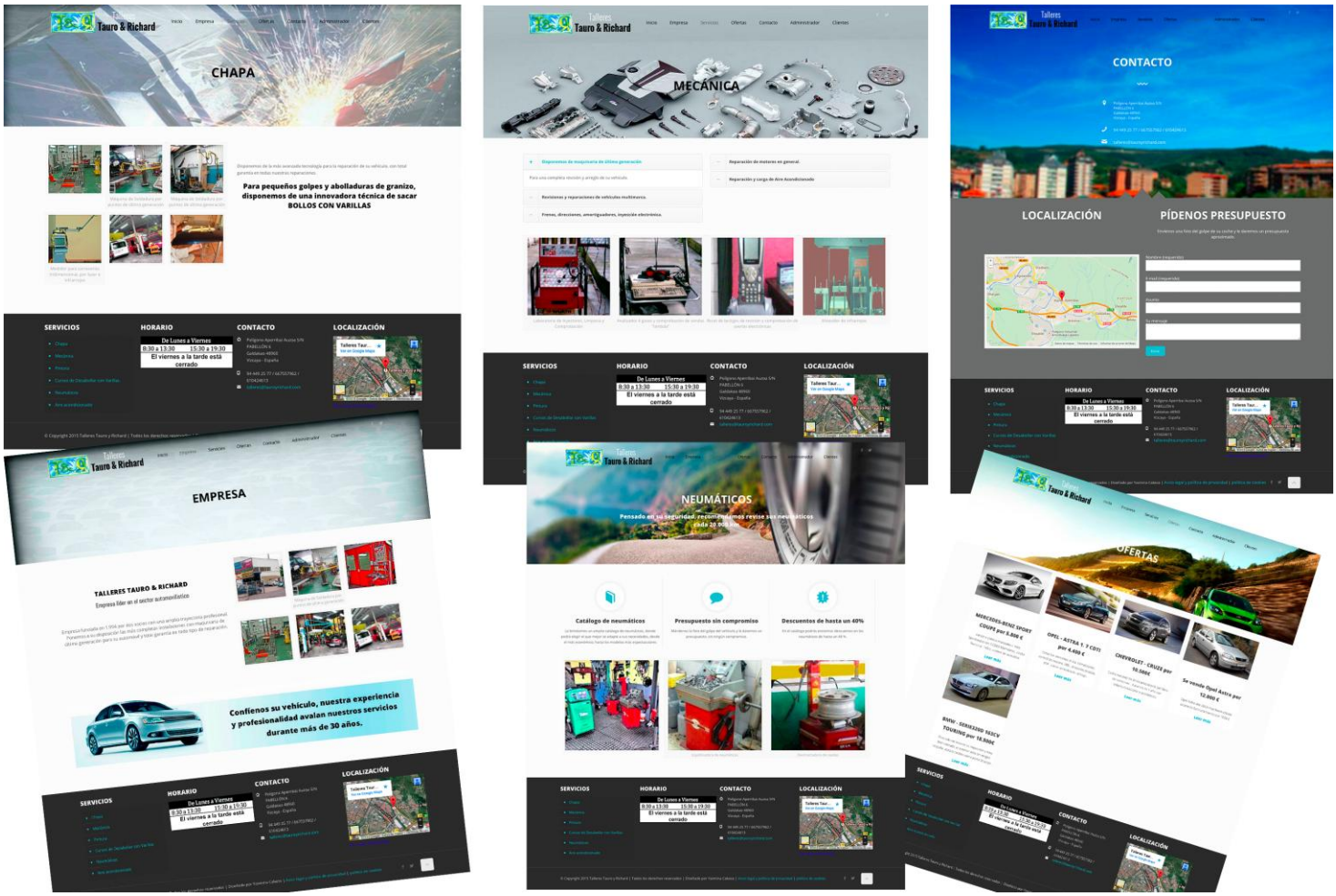
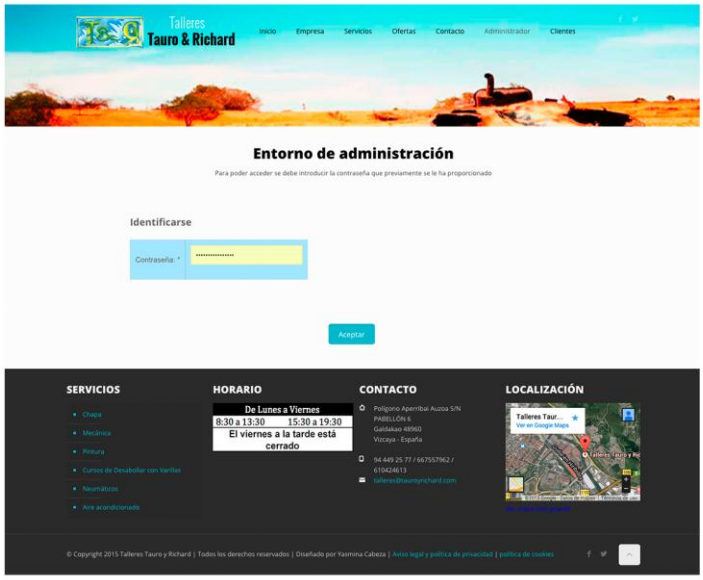


Ilustración 19. - Páginas que forman la aplicación web

En la siguiente ilustración aparecen la página principal del entorno de administración y la del entorno de clientes.



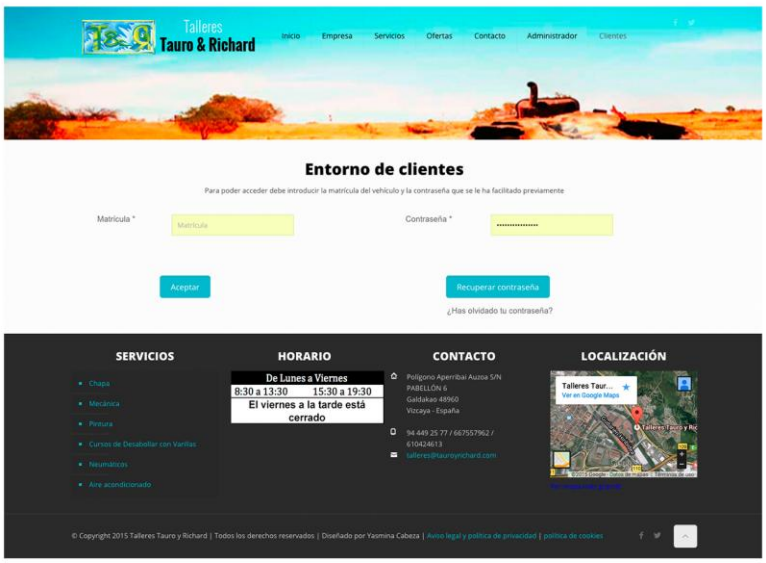


Ilustración 20. - Entorno de administración y de clientes

Para cumplir de una manera correcta la LOPD, se ha optado por la inserción en la página Web de un "Aviso Legal y política de privacidad", cuyo enlace aparece en el pie de página, en el que se proporciona a los visitantes toda aquella información relevante. Igualmente, aparece un banner informativo anunciando las cookies que presenta el sitio web. A pie de página se encuentra el enlace a la página que contiene la política de cookies.

La Ilustración 21 muestra, a la derecha, el aviso legal y, a la izquierda, la política de cookies.

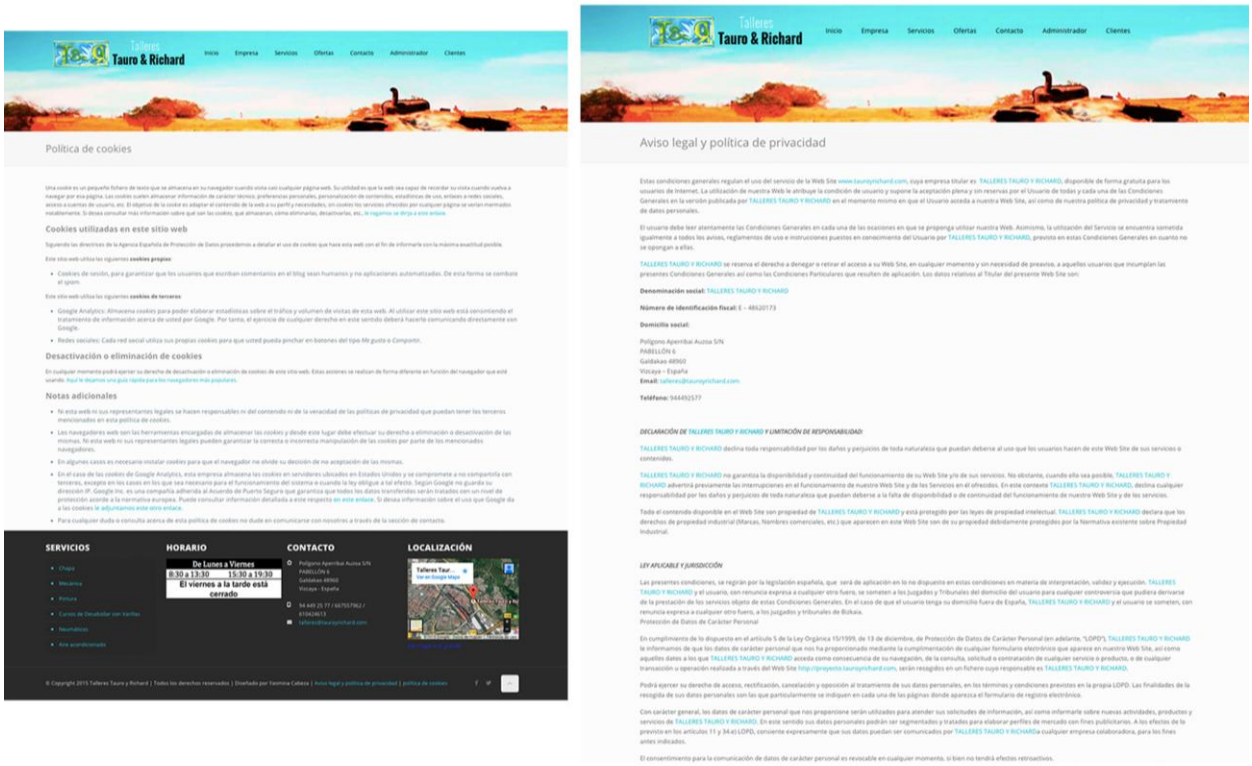


Ilustración 21. - Aviso legal y política de cookies

La Ilustración 22 muestra el banner informativo sobre las cookies de la página web. Pulsando la opción de "Leer más" se le reconducirá a la página en la que se encuentra la política de cookies.

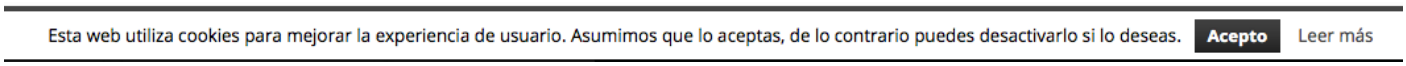


Ilustración 22. - Banner de cookies

6.1.4.2. Plugins instalados

Para añadirle más funcionalidad a la aplicación web se han instalado módulos o plugins. Ahora se van a detallar los módulos instalados más importantes:

- **Akismet:** Módulo que sirve para proteger el sitio de spam en comentarios y trackbacks.
- **Formulario de contacto:** Módulo que crea el formulario de contacto, que se encuentra en la sección de contacto de la página web. Sirve para enviar mensajes a la empresa a través de la aplicación web.
- **Cookie Law Info:** Con este módulo se crea el banner informativo de existencia de cookies.
- **Duplicator:** Sirve para la realización de copias de seguridad en la aplicación web.
- **Revolution Slider:** Con este módulo se crean los sliders o galerías de imágenes que se encuentran en la página web, como por ejemplo, en la página principal.

En la imagen siguiente, se puede ver la página de administración de módulos de la aplicación web, entre los que se encuentran los módulos de Panel de Administración y Pago online de la factura, cuya implementación se va a comentar en la siguiente sección.

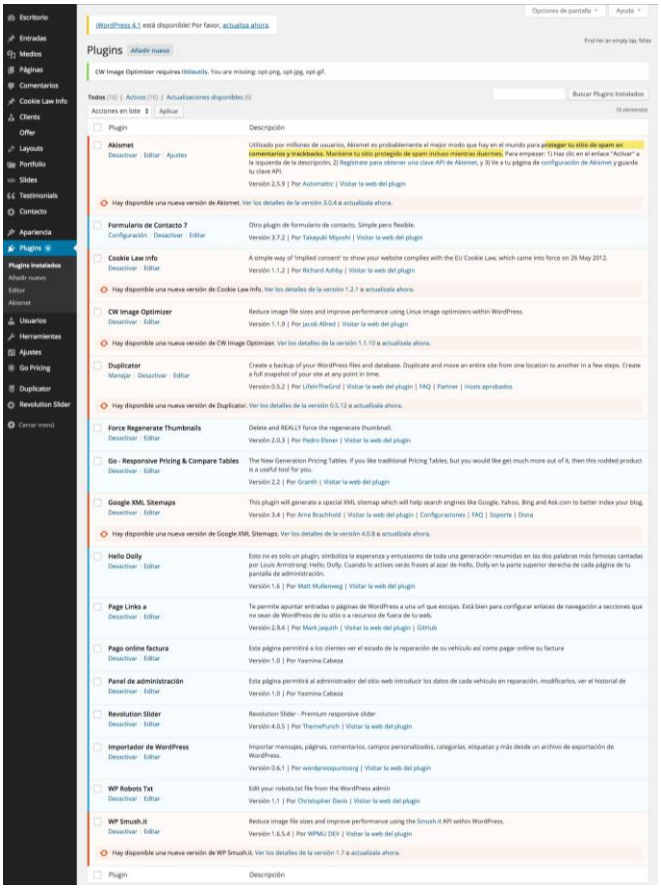


Ilustración 23. - Página de plugins de la aplicación web

6.1.5. Implementación de los módulos Panel de Administración y Pago online de la factura

En este apartado, se va a explicar en profundidad la manera en la que se han programado ambos módulos, dando una explicación detallada de la implementación de las funcionalidades principales.

6.1.5.1. Estructura de los archivos principales de los módulos

La estructura de los archivos principal.php e índice.php es similar. Estos archivos son los index de los módulos desarrollados. El archivo principal.php pertenece al módulo Panel de administración y el archivo índice.php pertenece al módulo Pago online de la factura. Es por ello, que cada uno de ellos debe llevar al principio unas líneas de código, indicando el nombre del paquete, la versión y una pequeña descripción del funcionamiento del módulo.

En las dos siguientes imágenes, aparece el comienzo tanto de principal.php como de índice.php, indicando el nombre del paquete, la versión y el nombre y descripción del plugin.

```php
<?php
/**
 * @package Panel de administración
 * @version 1.0
 */
/*
Plugin Name: Panel de administración
Description: Esta página permitirá al administrador del sitio web introducir los datos de cada vehículo en reparación,
modificarlos, ver el historial de las reparaciones, administrar noticias, añadir avisos y ver las cámaras web.
Author: Yasmina Cabeza
Version: 1.0

*/
```

Ilustración 24. - Parte inicial del archivo principal.php

```php
<?php
/**
 * @package Pagar online factura
 * @version 1.0
 */
/*
Plugin Name: Pago online factura
Description: Esta página permitirá a los clientes ver el estado de la reparación de su
vehículo así como pagar online su factura y recibirla por email.
Author: Yasmina Cabeza
Version: 1.0

*/

```

Ilustración 25. – Parte inicial del archivo índice.php

Como se ha mencionado antes, WordPress está creado mediante ganchos para poder efectuar cambios en el contenido de la página web, sin alterar el código fuente original.

Add_filter es el principal instrumento a la hora de utilizar los ganchos de WordPress, ya que permite unir una función creada a uno de los hooks disponibles. En ella, se debe indicar que al gancho the_content se quiere enlazar la función "nombre_de_la_funcion" y queda de la siguiente manera: add_filter ('the_content', 'nombre_de_la_funcion').

Sirva de ejemplo el siguiente fragmento de código del archivo principal.php, similar a índice.php, en el que aparece la función add_filter ('the_content', 'nombre_de_la_funcion') y debajo se encuentra la función correspondiente:

```
add_filter ('the_content', 'identificarse');
```

```
function identificarse ($content) {
      if (is_page()){
            if (get_the_ID()==13356){
         ?>
         <script type="text/javascript" src="prototype.js"></script>
         <script type="text/javascript">
function insertarbd(){
  if ( obligatorios()== true){
document.getElementById("formularionuevo").submit();
   }
  }
   . . . (Resto del código)
         </script>
         <body>
        <div id="tabla3">
        <h2>Introduzca los datos del vehículo</h2>
        <h6>Los campos con asterisco * son obligatorios</h6>
        </div>
        . . . (Resto del código)
         </body>
   <?php
}
}
return $content;}
```

Prosiguiendo con el análisis del código, después de declarar el add_filter, se procede a declarar la función 'identificarse'. El nombre de la función debe ser el mismo que se especifica en la anterior línea, de lo contrario ocurriría un error que supondría un riesgo para la integridad de la aplicación.

En el interior de cada función, se emplea la función is_page(). El uso de esta función tiene como objetivo determinar que el usuario se encuentra en una página. Esto es necesario porque al añadir

una acción al gancho the_content quiere decir que se utilizará cada vez que el tema WordPress haga uso de dicha función, por lo que se necesita filtrar según el tipo de página en el que se encuentra el usuario, para tener mayor control. A su vez, cada función se quiere mostrar en una página concreta de las creada en WordPress, las cuales llevan asociado un id. Para ello, se hace uso de la función get_the_ID(), para saber si la página en la que se encuentra el usuario es la misma en la que se debería insertar el código de la función. En caso de ser afirmativo, se introduciría en la página el código del interior de la función.

Con referencia a la función 'Identificarse', en su interior se debe introducir el código que generará el contenido a visualizar en la página deseada. El código de la función se divide en 2 partes. La primera parte hace referencia a la sección en la que se implementan las funciones JavaScript y la segunda parte al lugar donde se implementa el resto de código. En la sección JavaScript se especifica la función insertarbd(), que es una función que insertará los datos en la base de datos y, después, aparece el resto de código indicando los títulos que aparecerán en la página junto con sus etiquetas.

Por otro lado, indicar que ambos archivos (principal.php e índice.php) tienen la función estilo_css, en cada uno de ellos implementada de manera distinta. Es una función para indicar el código CSS que se quiere introducir en cada página de WordPress.

A continuación, aparece un pequeño fragmento dicha función.

```
function estilo_css($content) {
  if(get_the_ID()==13356 || get_the_ID()==13333){
  echo "
  <style type='text/css'>
  . h2{
    color: #000000;
    text-align: center;
    padding-top: 40px;
    font-size:30px;
  }
</style>
 ";}
return $content;}
```

Esta función solo introduce el código CSS de su interior en las páginas cuyo ID sea el indicado. Para ello, utiliza la función get_the_ID(), para saber si en la página en la que se encuentra el usuario es una de las que debería introducir el código de su interior. Dentro de las etiquetas <style> aparece un ejemplo de código CSS introducido en la función. Indica las propiedades que debe tener las etiquetas h2, que son el color, la alineación del texto, el tamaño del texto, entre otras.

6.1.5.2. Llamadas entre archivos php

Las llamadas entre archivos php se realizan de tres maneras posibles. La primera manera es mediante el uso de funciones AJAX. Dentro de cada función, se especifica la ruta del archivo prototype.js, que es el encargado de las funciones AJAX. La estructura de dicha función es la siguiente:

```
new Ajax.Updater (container, url[, options]); //Estructura de Ajax
```

```
<script type="text/javascript" src="prototype.js"></script>
```

Todas las llamadas que se realizan a través de Ajax se hacen de esta manera. A través de esta llamada se pueden enviar parámetros a los nuevos archivos mediante la etiqueta postBody.

A continuación, se muestra una función AJAX con envío de parámetros:

```
new Ajax.Updater('formulario', 'vehiculo-existente.php',{
asynchronous:false, postBody: 'matricula='+valor });
```

La siguiente función es una función AJAX sin envío de parámetros:

```
new Ajax.Updater('formulario', 'vehiculo-existente.php',{
asynchronous:false});
```

En este ejemplo se hace una petición al PHP 'vehiculo-existente.php', con los parámetros indicados en el postBody o sin ellos, que sustituirá el contenido del elemento div llamado 'formulario' con la respuesta que reciba.

La otra manera, en la que se realizan las llamadas entre archivos php, es mediante la opción Submit de los formularios. Para ello, el formulario necesita un botón que le indique que la edición del formulario ha concluido y que puede enviar la información. En la cabecera del formulario, en la etiqueta ACTION, se indica el path del archivo php al que se quiere llamar.

```
<FORM  name="formularionuevo" id="formularionuevo"
ACTION="eliminar.php" METHOD="post" enctype="multipart/form-data" >
```

Los botones de envío se crean con la etiqueta "Input" de tipo "Button". Además, se define el texto que aparecerá en el botón, mediante la etiqueta Value.

El código del botón es el siguiente:

```
<input type="button" name="modificar" value="Modificar"
onclick="insertarbd();">
```

La función insertarbd() está definida en la sección JavaScript indicando el formulario al que se quiere hacer Submit.

```
function insertarbd(){
…(Resto de código)
document.getElementById("formularionuevo").submit();
}
```

En este ejemplo, cuando el usuario pulsa el botón "Modificar" se ejecuta la función insertarbd(), que a su vez realiza el Submit del formulario, redireccionando al php indicado en la cabecera del formulario, que en este caso es eliminar.php.

La última manera es mediante redireccionamientos URLS y, con ello, se puede pasar parámetros de un archivo a otro. Este ejemplo basta para ilustrar la última opción de llamada entre archivos php.

```
location.href="http://proyecto.tauroyrichard.com/administracion-
clientes/?matricula="+val2;
```

```
function administracionclientes($content){
if(is_page()){
      if(get_the_ID()==13407){
$matricula = $_GET['matricula'];
. . . (Resto de código)}}}
```

Desde el archivo deseado se ejecuta la primera sentencia del código expuesto, indicando la url a la que se quiere redireccionar seguido de la variable o variables que se quieran enviar. En la función que se ejecuta cuando esa url es llamada, mediante la función $_GET, se recupera el valor o valores de las variables enviadas.

6.1.5.3. Conexiones con la base de datos

Cuando se realiza una llamada a un archivo php en la que se le pasan parámetros, en dicho archivo se recuperan dichas variables mediante $_REQUEST['nombre_de_la_variable']. La manera de realizar una conexión del servidor con la base de datos queda reflejado a continuación. A su vez, aparecen las sentencias que se ejecutan para obtener, eliminar o modificar información en la base de datos. Todos los archivos que realizan conexiones con la base de datos las realizan de esta manera.

Se comienza dando un ejemplo de cómo se recuperan los datos del formulario o de la función AJAX, con la función $_REQUEST.

```
$matricula=$_REQUEST['matricula'];
$fecha=$_REQUEST['fecha'];
```

La conexión con la base de datos se realiza con la función mysql_connect, que necesita, en primer lugar, el nombre del servidor, en segundo lugar, el usuario de la base de datos y, en tercer lugar, la contraseña de la base de datos. Por último, se utiliza la función mysql_select_db en la que se debe indicar el nombre de la base de datos. El siguiente código sirve para mostrar un ejemplo de conexión.

```
$conexion0=mysql_connect("db546010264.db.1and1.com", "dbo546010264",
"proyecto2014taller") or die ("No se puede conectar al servidor");


mysql_select_db("db546010264") or die ("No se puede seleccionar la
base de datos");
```

La aplicación web realiza interacciones con la base de datos. Para ello, se introduce la consulta en una variable. A continuación, con la función mysql_query se realiza la petición a la base de datos. Se muestra un mensaje de error, en caso de que la petición no se haya realizado correctamente. Una vez obtenidos los datos de la consulta, mediante la función mysql_fetch_array se procede a introducir los datos de la consulta en variables.

Para la selección de datos de la base de datos se utiliza la consulta SELECT. La sentencia para seleccionar datos en la base de datos es la siguiente:

```
$consulta= "SELECT cod_prov FROM tx_provincias WHERE nombre =
'".$provincia."' ";
$consulta=mysql_query($consulta,$conexion0);
```

```
$my_error = mysql_error($conexion0);
if(!empty($my_error)) { echo "Ha habido un error al consultar los
valores".$my_error;}
$codigo_provincia=mysql_fetch_array($consulta);
```

En esta consulta se obtiene el código de la provincia de la tabla tx_provincias, cuyo nombre de provincia sea igual al especificado en la variable $provincia. Por último, en la variable $codigo_provincia se guarda el valor obtenido en la consulta SELECT.

Para la inserción de datos en la base de datos se ejecuta la siguiente sentencia, en la cual se inserta en los campos código del servicio (codserv) y código de la reparación (cod_rep), de la tabla tx_actuaciones, el valor de las variables $codigoservicio y $codigoreparacion, respectivamente.

```
$consulta= "INSERT INTO tx_actuaciones (codserv,cod_rep) VALUES
('$codigoservicio','$codigoreparacion)";
```

Para la actualización de los datos en la base de datos se realiza la siguiente petición, en la cual se actualizarán los valores nombre y DNI de la tabla tx_vehículo, cuya matrícula sea igual a $matricula, con los valores de las variables $nombre y $dni, respectivamente.

```
$consulta ="UPDATE tx_vehiculo SET nombre = '".$nombre."', DNI =
'".$dni."'
WHERE matricula ='".$matricula."' ";
```

El borrado de datos de la base de datos se efectúa de la siguiente manera. En esta consulta se borran las tuplas, de la tabla tx_actuaciones, cuyo código de reparación (cod_rep) sea igual al valor de la variable $cod_rep.

```
$consulta= "DELETE FROM tx_actuaciones WHERE cod_rep
='".$cod_rep."'";
```

Además de las sentencias mostradas previamente, también se utiliza en determinadas conexiones con la base de datos la sentencia INNER JOIN. Esta sentencia es la sentencia JOIN por defecto, y consiste en combinar cada fila de una tabla con cada fila de la otra tabla, seleccionando aquellas filas que cumplan una determinada condición.

Para ilustrar mejor se muestra la sentencia join que se ha utilizado. En ese caso, se utiliza para obtener las matrículas de los vehículos que tengan al menos una reparación en el taller. Para ello, combina la tabla tx_vehículo con tx_reparacion unidos por el campo con el que coinciden, que es matrícula. También se utiliza la palabra DISTINCT que sirve para que no se repitan valores.

```
$consulta= "SELECT DISTINCT tx_vehiculo.matricula FROM tx_vehiculo
inner join tx_reparacion on
tx_vehiculo.matricula=tx_reparacion.matricula ";
```

En ocasiones, es necesario ordenar los datos que se reciben de la consulta utilizando de referencia algún campo de los obtenidos. Esto se realiza con la función ORDER BY. Un ejemplo se encuentra en la funcionalidad de Ver el historial de reparaciones, cuya tabla generada se puede ordenar por fecha, matrícula o nombre. El siguiente código sirve para ejemplificar las tres consultas diferentes necesarias para ordenar según esos tres campos.

```
$sqlrep="select * from tx_vehiculo inner join tx_reparacion on
tx_vehiculo.matricula=tx_reparacion.matricula ORDER BY
tx_vehiculo.nombre ASC"; //Ordena por nombre de manera ascendente
$sqlrep="select * from tx_vehiculo inner join tx_reparacion on
tx_vehiculo.matricula=tx_reparacion.matricula ORDER BY
tx_reparacion.fecha DESC"; //Ordena por fecha de manera descendente

$sqlrep="select * from tx_vehiculo inner join tx_reparacion on
tx_vehiculo.matricula=tx_reparacion.matricula ORDER BY
tx_vehiculo.matricula ASC"; //Ordena por matrícula de manera
ascendente
```

6.1.5.4. Inserción de imágenes en el servidor

La aplicación web tiene la posibilidad de insertar, modificar o eliminar ofertas. Esta funcionalidad está implementada en el módulo Panel de administración. Para llevarla a cabo, se necesita la posibilidad de insertar o eliminar imágenes del servidor.

Sirva de ejemplo el siguiente código para ilustrar mejor la manera de insertar imágenes en el servidor.

```
$archivo=$HTTP_POST_FILES['archivo']['tmp_name'];
$nombre=$_FILES['archivo']['name'];
$directorio="imagenes/";
$url_absoluta=$directorio.$nombre;
move_uploaded_file($archivo, $url_absoluta);
```

Con respecto a este código, en primer lugar, se obtiene la imagen introducida a través del formulario, mediante la función $HTTP_POST_FILE y se introduce en la variable $archivo. En segundo lugar, en la variable $nombre se introduce el nombre de la imagen subida. En tercer lugar, en la variable $url_absoluta se concatena el directorio donde se va a guardar la imagen en el servidor con la variable $nombre, por tanto, en esta variable queda guardado el path absoluto de la

ubicación de la imagen en el servidor. Por último, con la función move_uploaded_file se almacena la imagen en el path indicado.

Para eliminar la imagen del servidor se realiza de la siguiente manera, indicando el path de la imagen.

```
unlink(realpath($url_imagen));
```

6.1.5.5. Conexión de la aplicación web con Facebook

Como ya se ha mencionado, a través del módulo panel de administración, el administrador puede añadir ofertas y que a su vez, éstas se publiquen en Facebook. Para poder publicar artículos en Facebook desde la aplicación web es necesario, lo primero, descargar el SDK de PHP que contiene dos archivos "facebook.php" y "base_facebook.php", que se encuentran en la carpeta de src. Esta carpeta se encuentra dentro del plugin Administrar_datos_vehiculos y, a su vez, dentro de la carpeta Administrar_ofertas.

Se prosigue creando una aplicación en facebook, en la página Facebook Developer (https://developers.facebook.com/apps). Se rellenan todos los datos que se piden y una vez realizados todos los pasos, Facebook proporciona la AppId y Secret Key. Por último, para poder utilizar la aplicación es necesario adquirir un token y configurar los permisos que requiere la aplicación al usuario y los permisos que se le dan a la aplicación para publicar.

Una vez que se adquieren todos los datos necesarios, se comienza a realizar la implementación en el archivo de inserción de una nueva oferta (Administrar_datos_vehiculos > administrar_ofertas > introducirbd.php). El código necesario para la inserción de las ofertas en Facebook aparece a continuación:

```
require('facebook-php-sdk-master/src/facebook.php');
$app_id = '864598686918184';
$app_secret = '897948bf9bb920cb3bfc3833c6425970';
$token =
'CAAMSWRGbZAigBANAOdrBhj18Tml2JG2yxRC3XGyLgSuegCStOqbuC8N2wW86pcoac4ZCfzRAh
UbgpNwZCNV8ZC5Jy3YWPZCVnot6JWDMlJJsCJu1I5gnh8XdQq49plLBuXpuliGGouXZCxlN9z6n
WgkuMWqMAFvXTB9bjzslS0fAHkCojTkC5iseYgI9HyVJ6L234GqJW10PxxqZCFQQSI3';

$facebook = new Facebook(array(
    'appId' => $app_id,
    'secret' => $app_secret,
    'cookie' => false));

$req =  array(
```

```
        'access_token' => $token,
        'message' => 'Nueva oferta de http://proyecto.tauroyrichard.com',
        'name' => $titulo,
        'link' => $link,
        'description' => $descripcion,
        'picture' => $img);

    $res = $facebook->api('/me/feed', 'POST', $req);
```

En la primera línea aparece la función require() para introducir el código del archivo de Facebook.php en el archivo actual. Luego, se crean tres variables ($app_id, $app_secret y $token) en las que se introducen los datos obtenidos de la página de Facebook Developers. Con las siguientes líneas de código se especifican los datos de la oferta que se publicará en Facebook (La ruta de la imagen, el título, la descripción, el enlace, entre otros). Las últimas líneas de código permiten la inserción de la oferta en Facebook.

La siguiente imagen desvela la forma en la que la aplicación genera la oferta en Facebook. Cuando el administrador rellena todos los campos del formulario, desde el panel de administración, se crea una oferta que se guarda en la base de datos y se inserta la oferta en la página de Facebook del taller. Igualmente, la nueva oferta puede ser visualizada en la aplicación web a través de la pestaña "Ofertas", pudiendo pulsar dicha oferta y ver sus características de manera más detallada.

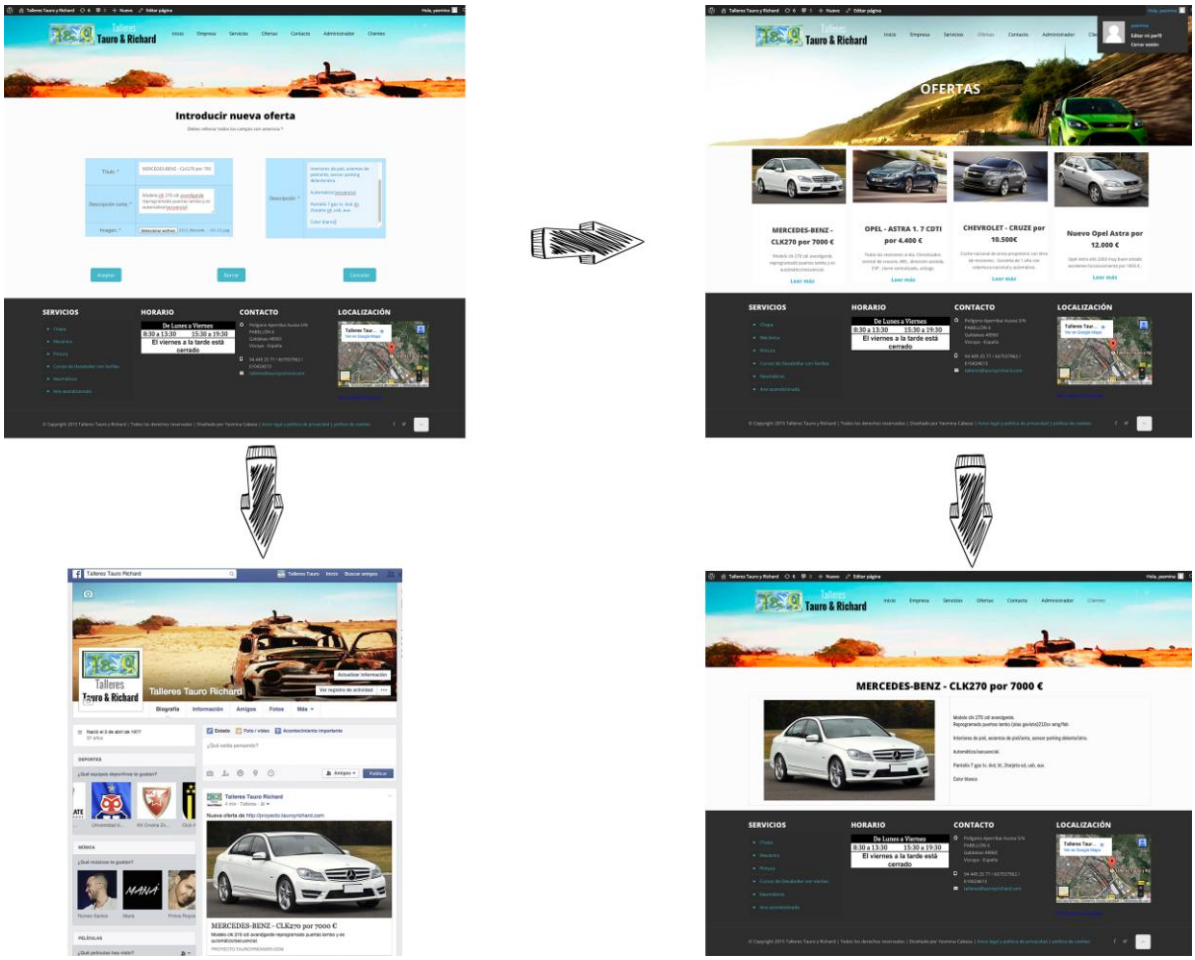


Ilustración 26. - Introducir ofertas en Facebook

6.1.5.6. Creación de facturas en PDF

En la aplicación web, tanto en el módulo de Panel de administración, como en el Pago online de la factura, hay funcionalidades que generan facturas en pdf. Para la creación de facturas en PDF se ha utilizado la librería mPDF. Esta librería está basada en la librería FPDF, pero permite pasar una plantilla de HTML, con sus hojas de estilos, imágenes y cualquier otro elemento a PDF con unos sencillos pasos.

Esta librería permite varios casos:

- Abrir el archivo pdf generado.
- Guardar el pdf en el servidor con el nombre y la ruta que se le indique.
- Se puede transformar en cadena y enviarlo por correo electrónico.

Lo primero que se realiza es crear el objeto de la librería MPDF. Luego, se invoca a su método WriteHTML y se le pasa la cadena HTML que se desea convertir a PDF. Por último, se indica que cree el archivo mediante el método Output.

En la siguiente imagen aparece cómo se crea el objeto mPDF, habiendo puesto previamente las líneas de require_once de mPDF. A continuación, en las variables $html, $html2, $html3 se introduce todo el contenido de la factura.

```
define('__ROOT__', dirname(dirname(__FILE__)));

require_once(__ROOT__.'/mpdf/mpdf.php');

$mpdf = new mPDF('utf-8', 'A4');
$mpdf=new mPDF();
$html='. . .';
$html2='. . . ';
$html3='. . .';
$mpdf->WriteHTML($html.$html2.$html3);
```

Cabe destacar el método WriteHTML que pasándole como argumentos las tres variables html se forma la factura. Por último, se invoca al método Output. Este método tiene diferentes parámetros de configuración, sin embargo, en este proyecto se han utilizado los siguientes:

- I: Se utiliza en la funcionalidad de Ver historial de reparaciones, en la opción "Ver factura".
- D: Se utiliza cuando un cliente realiza el pago de la factura a través de internet, utilizando el método de pago Transferencia bancaria, puesto que en la pantalla de agradecimiento aparece la opción "Descargar factura".
- S: Esta opción se utiliza cada vez que el cliente realiza el pago de la factura por Internet, con cualquiera de los dos métodos de pago. Cuando se le envía al cliente un email con todos los detalles del pago se le envía, igualmente, la factura como archivo adjunto.

El código necesario para que la factura se abra directamente en le navegador es el siguiente:

```
$modo="I";
$nombre_archivo="Factura_numero"factura.pdf";
       $mpdf->Output($nombre_archivo,$modo);
```

Para forzar la descarga de archivos hay que realizar lo siguiente:

```
$modo="D";
$nombre_archivo="Factura_numero"factura.pdf";
	$mpdf->Output($nombre_archivo,$modo);
```

Para poder adjuntar una factura en pdf en un email, se debe escribir el siguiente código:

```
	$content = $mpdf->Output('','S');
```

En la imagen siguiente se puede apreciar la factura que se genera a través de la aplicación, pudiendo haber dos variantes de la misma, según si la factura está abonada o no.

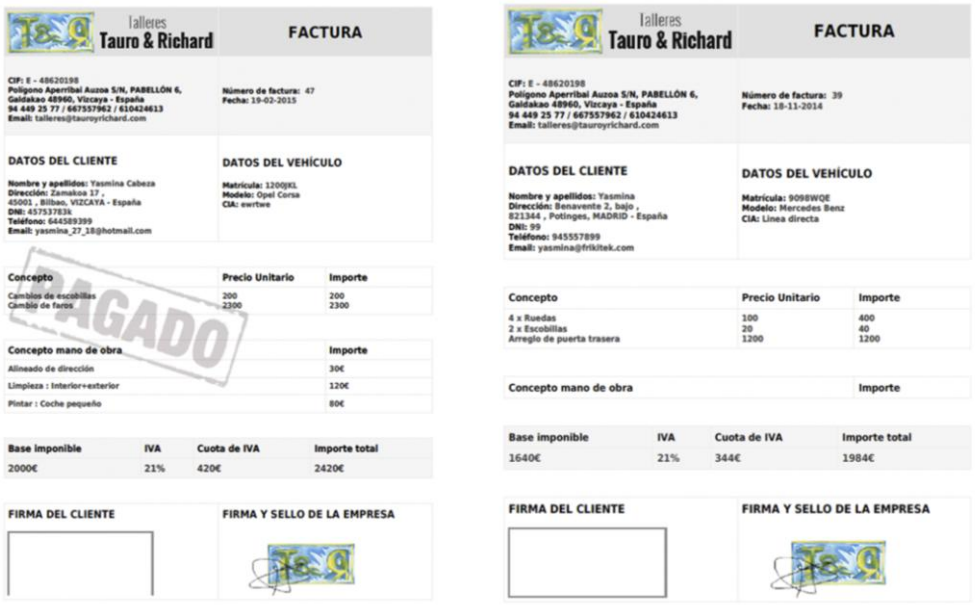


Ilustración 27. - Facturas en pdf

6.1.5.7. Envío de emails desde archivos php

Cuando el administrador introduce los datos de una reparación de un vehículo nuevo, es decir, que no ha realizado ninguna reparación en el taller antes, el sistema crea una nueva contraseña para ese cliente y se la envía por correo electrónico. Dicha contraseña sirve para que el cliente pueda ingresar en el entorno de clientes de la aplicación web y en la App móvil.

La contraseña se genera de manera aleatoria y para ello se utiliza un script generador de contraseñas en php. El script utilizado aparece en la siguiente imagen.

```
$str =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890";
$pwd = "";
for($i=0;$i<6;$i++) {
$pwd.= substr($str,rand(0,62),1);
}
```

Como ya se ha mencionado, esta contraseña será enviada al cliente mediante correo electrónico.

El código necesario para enviar un email desde un archivo php aparece a continuación. En primer lugar, se visualiza la variable $mail en la que se introduce el cuerpo del mensaje a enviar, la variable $titulo será el asunto del email, $headers es la variable donde debe ir la codificación del mensaje y el remitente del mismo. Por último, en la variable $to se indica a qué email se desea mandar el correo electrónico. Con la función mail() de PHP se envía el email correspondiente.

```
$mail = "
<div id='titulo'>
<h5>Estimado/a $nombre ,<br><br>
Nos alegra que haya confiado en nosotros para realizar la reparación
de su vehículo.
.. .. ..";
$titulo = "Contraseña asignada en Talleres Tauro & Richard";
$headers .= 'Content-type: text/html; charset=utf-8' . "\r\n";

//dirección del remitente
$headers .= "From: Talleres Tauro & Richard
<talleres@tauroyrichard.com>\r\n";
$Msj.= "$mail\n";
$to = $email;
mail($to,$titulo,$Msj,$headers);
```

Cuando los clientes realizan el pago de la factura de la reparación a través de la aplicación web se genera una factura en PDF y se envía a los clientes a través de un email. Es por esto que, en primer lugar, se genera la factura correspondiente, como se ha indicado en el apartado anterior. En segundo lugar, se añade como adjunto al mensaje.

En la siguiente imagen aparece la variable $content que es donde se encuentra la factura guardada.

```
$mpdf->WriteHTML($html.$html2.$html3);
$content = $mpdf->Output('', 'S');//En esta variable se almacena la
factura
```

En la imagen siguiente aparece el código con el que se realiza el envío de un email con un archivo adjunto a través de PHP. Las cabeceras son iguales que al enviar un email normal. La variable $content tiene la factura en PDF y para poder mandarla adjunta al email debe estar codificada con la función base64_encode(). Asimismo, se utiliza la función chunk_split() para dividir el archivo en pequeños trozos.

```
$content = chunk_split(base64_encode($content)); //En la variable
$content se encuentra la factura en pdf


$filename = 'factura.pdf';


$header .= "Content-Type: multipart/mixed;
boundary=\"".$uid."\"\r\n\r\n";
$header .= "This is a multi-part message in MIME format.\r\n";
$header .= "--".$uid."\r\n";
$header .= $message."\r\n\r\n";
$header .= "Content-Type: application/pdf;
name=\"".$filename."\"\r\n";
$header .= "Content-Transfer-Encoding: base64\r\n";
$header .= "Content-Disposition: attachment;


filename=\"".$filename."\"\r\n\r\n";


$header .= $content."\r\n\r\n";
$header .= "--".$uid."--";


$is_sent = @mail($mailto, $subject, "", $header);
```

6.1.5.8. Envío de notificaciones al servidor GCM

Cuando el administrador introduce un aviso a través de la aplicación web o cuando modifica los datos de una reparación y cambia los valores del estado de la reparación y factura a "Finalizada" y "Sin pagar", respectivamente, se introduce un aviso en la base de datos y se manda una notificación al servidor GCM para que envíe la notificación a los dispositivos móviles correspondientes. Estas funcionalidades son del plugin Panel de administración.

Para ello se utilizan las peticiones http POST con CURL. En la siguiente imagen se puede apreciar el código de programación para enviar dicha notificación.

En primera instancia, se selecciona el IdGCM del dispositivo móvil, que es el id obtenido en el registro del dispositivo móvil en el GCM, almacenado en la tabla tx_app.

```
$consultagcm= "SELECT IdGCM from tx_app where DNI='$dni' ";
$idGCM = $row['IdGCM'];
```

```
$regId=$idGCM;
$regArray[]=$regId;
```

```
$url = 'https://android.googleapis.com/gcm/send';

$fields = array('registration_ids' => $regArray, 'data' =>
array("nombre" => "$nombre" , "matricula" => "$matricula" ),);

$headers = array( 'Authorization:
key=AIzaSyDBuPa7gbmgSK8Ejo2ro3UnqUTkeVqZwIc','Content-Type:
application/json');

$ch = curl_init();

curl_setopt($ch, CURLOPT_URL, $url);
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_HTTPHEADER, $headers);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
curl_setopt($ch, CURLOPT_POSTFIELDS, json_encode($fields));

$result=curl_exec($ch);
```

Por último, se envía la notificación mediante http POST con CURL, indicando en la cabecera ($headers) el API key que se obtiene al registrarse en Google Developers Console. Por otro lado, se indica la url del GCM, que es a donde se envía la notificación. También se envían datos adjuntos en la notificación y se introducen en el atributo $fields que es un array JSON. En dicho array se introducen las variables nombre y matrícula, para que cuando llegue la notificación al dispositivo móvil pueda tratar esos datos y realizar las operaciones pertinentes. En esa misma variable también se inserta el id del dispositivo al que se quiere enviar la notificación, obtenido del registro del dispositivo en el GCM.

6.1.5.9. Generación de tablas dinámicas

En la sección historial de reparaciones, del módulo panel de administración, para crear la tabla con todas las reparaciones se genera una tabla de manera dinámica. Esta funcionalidad es empleada, igualmente, en el módulo de Pago online de la factura, en la funcionalidad de mostrar el historial de reparaciones de un vehículo.

Al principio aparece la consulta a la base de datos con INNER JOIN para sacar los datos necesarios de las tablas tx_vehiculo y tx_reparacion. A continuación, se recorren todas las filas de la consulta generando así la tabla dinámica con todos los datos.

La consulta a la base de datos con INNER JOIN es la siguiente:

```
$sqlrep="SELECT * from tx_vehiculo inner join tx_reparacion on
tx_vehiculo.matricula=tx_reparacion.matricula ORDER BY
tx_reparacion.fecha DESC";
```

El bucle para recorrer todas las filas de la consulta y la generación de la tabla aparece a continuación:

```
while($fila=mysql_fetch_array($consultarep)
    echo "
  <table border='1' align='center' style='table-layout: fixed;'>
  <tr>
 <td><input type='radio' value='".$fila['cod_rep']."'
onclick='habilitar_boton();'/> </td>

  <td>".$fila["fecha"]."</td>

            . . . . (El resto del código)
 </tr>
</table>";
```

6.1.5.10. Validación de los formularios

En todos los formularios de la aplicación web se realiza la validación de los datos introducidos por los usuarios. De esta manera, si el usuario ha cometido algún error al rellenar el formulario, se le puede avisar de forma instantánea, sin tener que esperar una respuesta del servidor. La notificación de errores de forma inmediata, mediante funciones JavaScript, mejora la experiencia de usuario y ayuda a reducir la carga de procesamiento en el servidor.

Para la validación de los campos de texto obligatorios se utiliza la siguiente función, que comprueba que el valor introducido sea válido, que el número de caracteres introducido sea mayor que cero y que no se hayan introducido solo espacios en blanco. En caso de que alguna de estas tres condiciones no se cumpla devuelve un error al usuario.

```
if( valor == null || valor.length == 0 || /^\s+$/.test(valor) ) {
    alert("Falta de rellenar el campo NOMBRE");
    return false;
}else{
    return true;
```

Para la validación de una dirección email se comprueba que la dirección parezca válida, ya que no es posible la comprobación de si es una cuenta real y operativa. Para ello, se verifica mediante expresiones regulares. En caso de que no cumpla dicha expresión, aparece un cuadro de error.

La condición Javascript consiste en:

```
var filter=/^[A-Za-z][A-Za-z0-9_]*@[A-Za-z0-9_]+.[A-Za-z0-9_.]+[A-
za-z]$/;
if (filter.test(valor)){
  return true;
}else{
   alert("Formato de email no válido");
   return false;}
```

Una vez más, para la validación de un número de teléfono se hace uso de las expresiones regulares, que comprueban si el número introducido es una sucesión de nueve números consecutivos, sin espacios ni letras. En caso de que no se cumpla, se produce un error.

```
if( !(/^\d{9}$/.test(valor)) ) {
   alert("Tiene que escribir un teléfono de 9 dígitos");
   return false;
}else{
   return true;
```

Para la validación de una fecha se utiliza la función Date (año, mes, día) que permite construir fechas a partir de los datos introducidos. Como se puede ver, el número de mes se indica de 0 a 11, es por esto que a la variable mes hay que restarle siempre 1 porque el usuario introducirá un valor entre 1 - 12. La validación consiste en intentar construir una fecha con los datos proporcionados por el usuario. Si los datos del usuario son incorrectos, la fecha no se puede construir y se crea un mensaje de error.

```
aho = fechaArr[0];
mes = fechaArr[1];
dia = fechaArr[2];

plantilla = new Date(aho, mes - 1, dia);//mes empieza de cero Enero
= 0

if(!plantilla || plantilla.getFullYear() == aho &&
plantilla.getMonth() == mes-1 && plantilla.getDate() == dia){
    return true;
}else{
     return false;}
```

6.1.5.11. Suma de variables

En la funcionalidad de administrar avisos, en el panel de administración, el administrador puede introducir diferentes avisos, entre los cuales se encuentran, por ejemplo, cambio de aceite o carga de aire acondicionado.

Al elegir, por ejemplo, cambio de aceite aparecerán dos campos de texto para introducir los valores. Cuando el administrador introduce el primer valor, automáticamente se genera el siguiente valor en función del aviso escogido.

- En el caso de cambio de aceite cuando el administrador introduce los km del vehículo en la actualidad, el sistema suma 15.000 km al valor introducido porque el cambio de aceite se realiza cada 15.000 km.
- En el caso de cambio de ruedas, al valor introducido se le suma 25.000 km.
- Cuando se introduce un aviso de cambio de embrague, se le suma al valor introducido 150.000 km.
- En el aviso cambio de pastillas de frenos se le suma 50.000 km.

Para el cálculo de dichas sumas se utiliza la función parseInt () que devuelve un valor entero. Se utiliza para garantizar que la suma se realice correctamente.

```
resultado=(parseInt(kmactual) + parseInt('50000')); //Cambio de
frenos
```

De la misma forma, hay otros dos tipos de avisos que en lugar de contabilizarse mediante kilómetros, se realiza mediante fechas.

- Para la carga de aire acondicionado, a la fecha actual o última fecha de carga, se le suman 5 años.
- Para el paso de la próxima ITV, se le suman 4 años a la fecha introducida.

Para generar la siguiente fecha de manera automática se utiliza la siguiente función:

```
sumarDias=parseInt(1825); //5 años = 1825 días
fecha= new Date(fecha);
fecha.setDate(fecha.getDate()+sumarDias);
var anio=fecha.getFullYear();
var mes= fecha.getMonth()+1;
var dia= fecha.getDate();
```

```
if(mes.toString().length<2){
mes="0".concat(mes);
}

if(dia.toString().length<2){
    dia="0".concat(dia);
}
resultado=anio+"-"+mes+"-"+dia;
```

En la primera variable se guardan los días que hay que sumarle a la fecha introducida. A continuación, le suma los días a la fecha introducida con la función fecha.setDate(). Realiza las comprobaciones pertinentes y, por último, guarda en la variable resultado la fecha generada.

6.1.5.12. Pagos a través de la aplicación web

La aplicación web, con su módulo de Pago online Factura, permite a los clientes realizar los pagos de sus facturas a través de la aplicación. En ella existen dos métodos de pago: Transferencia bancaria y PayPal.

Para realizar el pago por transferencia bancaria, la aplicación simplemente se encarga de enviar un email tanto al administrador, de que el cliente desea realizar el pago mediante una transferencia, como al cliente, indicando en el email los datos bancarios de la empresa para que pueda realizar la transferencia. En este mismo email viene adjunta la factura en pdf, además de todos los datos relacionados con la reparación, así como el importe de la misma.

El otro método de pago es mediante PayPal. En primer lugar, se crea una cuenta de empresa en PayPal. Antes de poner activo este método de pago, se realizan las pruebas en un entorno de pruebas, llamado Sandbox Test. Puesto que este proyecto todavía no está implantado en la empresa, el método de pago sigue en modo test y, por tanto, el email necesario para llevar a cabo la prueba de la compra es yasminacabeza-buyer@gmail.com, con su correspondiente contraseña.

Como email del propietario de la cuenta, también se ha creado uno de prueba: yasminacabeza-facilitator@gmail.com

Para la implementación de este método de pago, es necesario introducir el siguiente código:

```
<form id='formulariopypal' name='formulariopypal'
action='https://www.sandbox.paypal.com/cgi-bin/webscr'
method='post'>
<input type='hidden' name='cmd' value='_xclick'>
```

```
<input type='hidden' name='business' value='yasminacabeza-
facilitator@gmail.com'>
<input type='hidden' name='item_name' value='Pago de la factura del
coche '".$row['matricula']."'>
<input type='hidden' name='currency_code' value='EUR'>
<input type='hidden' name='amount'
value='".$row['importe_total']."'>
<input type='hidden' name='cod_rep' id='cod_rep'
value='".$cod_rep."'>
<input type='hidden' name='return'
value='http://proyecto.tauroyrichard.com/gracias-por-confiar-en-
nosotros/'>
<input class=boton1 type='button' name='paypal' value='PayPal'
onclick='pagop();'>
```

Como se puede ver, es un formulario cuyo ACTION es la url de la página de PayPal a la que se le va a redireccionar al cliente para proceder con el pago de la factura, en este caso le redirecciona al usuario al entorno de pruebas. En los diferentes campos del formulario se especifica, entre otras cosas, el email del propietario de la cuenta, el título del pago, la moneda a utilizar, la cantidad a la que asciende el pago y la página a la que se le redirecciona, una vez haya concluido el pago.

Para pasar de entorno de pruebas a entorno real, solamente se debería cambiar la url de destino del ACTION del formulario, indicando la que PayPal nos comunique como entorno real.

En todos los pagos realizados la aplicación envía un email al administrador comunicando el pago de la factura y al cliente con todos los detalles del pago, junto con la factura en pdf. Una vez que ha concluido el pago, PayPal redirecciona al usuario a la página de agradecimiento de la aplicación.

6.1.5.13. Integración de cámaras web en la aplicación

Otra de las funcionalidades del panel de administración es que el administrador pueda ver las cámaras IP desde la aplicación. Para llevarlo a cabo, es necesario conocer las IPs del router y del grabador de las cámaras.

Para poder ver las cámaras web desde un servidor remoto, es necesario configurar la apertura de puertos del router para poder derivar el tráfico del grabador que llega al router. Para visualizar las cámaras a través de la red local, basta con poner en un navegador la dirección IP del grabador. Sin embargo, para poder tener acceso remoto, no hay que utilizar la dirección IP que se ha establecido para el grabador, sino utilizar la dirección IP externa del router (IP pública).

En primer lugar, se realizan las configuraciones del router y la operación básica consiste en abrir los puertos que utiliza el vídeo grabador. Por tanto, se procede a abrir los siguientes puertos desde la configuración del router:

- 37777 para el protocolo TCP
- 37778 para el protocolo UDP
- 80 para el puerto http y se abre con protocolo TCP y UDP

Una vez que los puertos del grabador se han abierto, para poder incluirlo en la aplicación web se emplea el código definido a continuación, indicando la IP pública del router.

```
<iframe style="width: 247px; height: 153px;" src="http://
XX.X.XXX.XXX" frameborder="0" height="128" scrolling="no"
width="313"> </iframe>
```

De esta manera queda embebida la página de acceso a las cámaras web como se puede observar en la siguiente imagen. En primer lugar, sale una pantalla que requiere datos de acceso (Usuario y contraseña) y una vez introducidos se podrá visualizar las cámaras web, así como realizar cualquier tipo de configuración en las mismas o ver las grabaciones de días pasados a través de la aplicación.

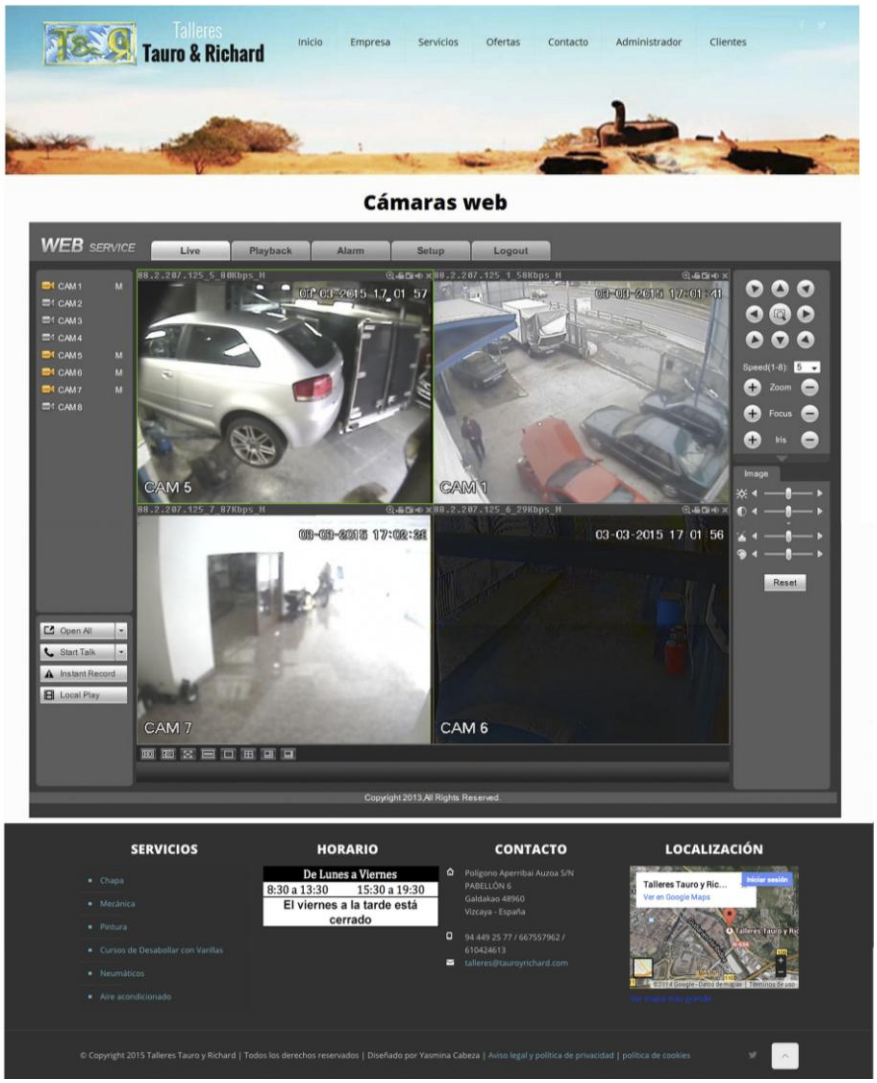


Ilustración 28. - Visualización de las cámaras web

6.2. Implementación de la aplicación móvil

En esta sección se va a realizar una descripción exhaustiva de la implementación de la aplicación móvil. Esta aplicación ha sido desarrollada para el sistema operativo Android.

Para lograr este objetivo, en alguno de los apartados se recurrirá a la inclusión de fragmentos de código fuente para ilustrar las explicaciones, advirtiendo que dicho código no es una copia literal del implementado, ya que incluirlo en su totalidad no sería posible por problemas de espacio.

6.2.1. Sistema operativo Android

Android es un sistema operativo basado en Linux y realizado, sobretodo, para dispositivos móviles y tablets. Es una plataforma de código abierto distribuida bajo la licencia Apache 2.0 por lo que su distribución es libre y posibilita el acceso y modificación de su código fuente.

En sus inicios fue desarrollado por Google, para más tarde unirse a la Open Handset Alliance (de la cual, Google también forma parte) que está integrada por T-Mobile, Intel, Samsung, HTC o Nvidia entre otros. No obstante, Google ha sido la compañía que ha publicado la mayor parte del código fuente bajo la licencia Apache.

Con respecto al ámbito del desarrollo de software para esta plataforma, a los desarrolladores se les facilita de manera gratuita el SDK y un plugin que se integra dentro del entorno de desarrollo de Eclipse, donde se incluyen todas las APIs necesarias. Igualmente, se incorpora un potente emulador integrado para la supervisión de las pruebas de la aplicación.

Esta plataforma está diseñada mediante capas y utiliza el kernel de Linux 2.6 que le da acceso a la parte hardware de los dispositivos a la vez que le permite ser compatible con muchos de los drivers creados para Linux. Esta arquitectura se refleja perfectamente en la imagen que viene a continuación y debajo se van a detallar cada una de las capas.

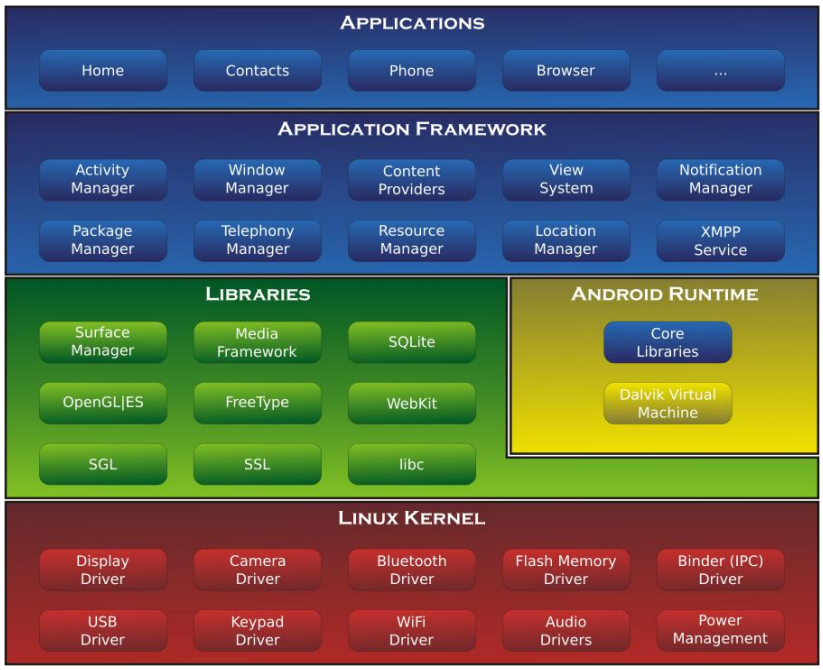


Ilustración 29. - Estructura de Android [45]

- **Marco de aplicaciones (Framework):** Los programadores tienen acceso completo al SDK de Android para facilitar la implementación de aplicaciones. La estructura de Android está diseñada para simplificar la reutilización de funciones o componentes. Todo esto es posible si las normas de seguridad impuestas por el framework se cumplen, de manera que se logra cierto grado de reutilización del código.
- **Aplicaciones:** Todas las aplicaciones están escritas en lenguaje de programación Java. Estas aplicaciones pueden estar compuestas por 5 bloques, de los cuales se utilizarán los necesarios para la aplicación a desarrollar: Activity, Intent, Broadcast, Services y Content Providers. Las aplicaciones previamente instaladas incluyen un cliente de correo electrónico, programa de SMS, calendario, mapas, navegador, contactos y otros.
- **Bibliotecas:** Android incorpora un grupo de bibliotecas de C/C++ utilizadas por varios componentes del sistema. Estas bibliotecas se ponen de manifiesto a los programadores mediante el marco de trabajo de aplicaciones de Android. Algunas son: System C library (implementación biblioteca C estándar), Media libraries (Librería de medios), Surface Manager (bibliotecas de gráficos) y SQLite, entre otras.
- **Núcleo Linux:** El Kernel de Linux actúa como una capa de abstracción entre el hardware y el resto de la pila de software. Android también necesita a Linux para los servicios base del sistema como seguridad, gestión de memoria, gestión de procesos, pila de red y modelo de controladores.
- **Runtime de Android (Android Runtime):** En la imagen anterior de la arquitectura de Android se puede apreciar que al mismo nivel que las librerías de Android se encuentra el Runtime de Android. Está formado por librerías que otorgan la mayoría de las funcionalidades disponibles en las bibliotecas del lenguaje Java, incluyendo la máquina virtual. Cuando una App se ejecuta crea su propio proceso con su propia instancia de la máquina virtual. Esta máquina virtual se ha escrito de tal manera que permite el lanzamiento de varias instancias de sí misma de manera eficiente.

6.2.2. Componentes de la aplicación

En este apartado se van a especificar los componentes o bloques que conforman la aplicación, así como la estructura de carpetas de la misma.

6.2.2.1. Estructura de la aplicación

En este apartado, se va a explicar cómo estructura Android los recursos necesarios para crear la aplicación.

En primer lugar, para crear la interfaz de la aplicación, cuando se crea un proyecto se genera una carpeta llamada res que contiene varias carpetas dentro de ella. Las tres carpetas principales dentro de res son Drawable, Layout y Values.

- **Drawable** contiene las imágenes de fondo utilizadas por la aplicación, el icono de la aplicación, así como ficheros XML utilizados para crear el estilo de las tablas dinámicas, de la clase Revisiones.
- **Layout** contiene las vistas encargadas de crear la interfaz, es decir, los ficheros XML a partir de los cuales se crea la interfaz. Existen diversos tipos de carpetas Layout, como por ejemplo layout-land. Esta característica se utiliza para crear distintas interfaces según la orientación de la interfaz. El proyecto hace uso de la carpeta Layout para establecer la interfaz básica para dispositivos móviles con una resolución de tamaño normal. Así pues, para la orientación horizontal se ha utilizado la carpeta layout-land.
- **Values**, utilizada para almacenar valores de la forma clave valor. Se utiliza principalmente para almacenar cadenas de caracteres y acceder a ellas posteriormente haciendo uso del id. Dentro de Values encontramos los archivos string y styles. El primero de ellos se usa para almacenar simples cadenas y el segundo permite crear estilos.

En cuanto al código que implementa las funcionalidades puede encontrarse en la carpeta src. Aquí se hace una distinción entre las actividades, los servicios y Broadcast. Todos ellos se encuentran en la misma carpeta. Las actividades de la aplicación corresponden con los archivos MainActivity.Java, MenuCliente.java, Revisiones.java y Formulario.java. El archivo Service es GCMIntentService.Java y el Broadcast es GCMBroadcastReceiver.Java.

Por último, aparece el archivo AndroidManifest.xml, encargado de realizar una descripción exhaustiva de los componentes de la aplicación.

En la siguiente imagen queda reflejado la estructura de la aplicación móvil.

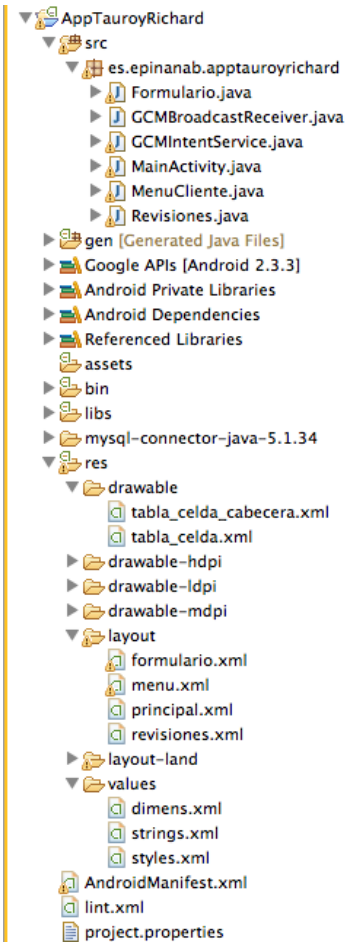


Ilustración 30. - Estructura de la aplicación móvil

6.2.2.2. Activity

Las actividades representan el componente principal de la interfaz gráfica de una aplicación Android. Se deduce, por tanto, que una actividad podría ser el elemento análogo a una ventana en cualquier otro lenguaje visual.

Las actividades tienen la misión de mostrar las interfaces de usuario e interactuar con él. Para ello tienen asociada una definición de pantalla, en formato XML, y son las encargadas de responder a los eventos de usuario (Pulsaciones de botones…etc.)

Una actividad refleja una determinada acción llevada a cabo por una aplicación. Lleva asociada la interfaz de usuario representada por la clase View y sus derivados. La aplicación móvil está compuesta de varias pantallas con diferentes objetivos, es por esto, que estará compuesta de varias actividades. Todas las actividades creadas en Java son una subclase de Activity.java.

Las actividades de la aplicación son las siguientes:

- **MainActivity.Java** es la clase encargada de la identificación del cliente. Para ello, realiza también el registro del dispositivo en el GCM para, posteriormente poder recibir notificaciones.
- **Formulario.java** es la clase encargada de crear el formulario para pedir cita. En esta clase se crea el calendario dinámico para elegir fecha y se envía un email a la empresa con la fecha para la cita.
- **Revisiones.java** es la clase encargada de generar una tabla dinámica con los datos de las revisiones del cliente (Cambio de aceite, carga de aire acondicionado…etc.).
- **MenuCliente.Java** es la clase que se encarga de generar los botones del menú y comprobar si la reparación del vehículo del cliente ha sido finalizada para poder indicárselo en la página del menú.

6.2.2.3. Services

La App también está formada por servicios. Son componentes sin interfaz gráfica que se ejecutan en segundo plano. Los servicios tienen la capacidad para actualizar datos o lanzar notificaciones, entre otras. Todos los servicios creados serán una subclase de la clase Service.java.

La clase servicio de la App es la clase GCMIntentService.java que se encarga de recibir las notificaciones que van a llegar al terminal móvil.

Esta clase extiende de la clase Intent Service que es un tipo particular de servicio Android cuya misión es la creación y gestión del nuevo hilo de ejecución y detenerse a sí mismo una vez terminada la tarea.

Esta clase contiene los métodos necesarios para poder manejar la notificación y realizar la acción deseada.

6.2.2.4. Broadcast Receiver

Este componente tiene la función de detectar y reaccionar ante determinados mensajes o eventos globales que se generan en el sistema. Este componente tampoco tiene interfaz gráfica asociada pero puede utilizar la API Notification Manager para avisar de manera no intrusiva al usuario del

evento que se ha producido, a través de la barra de notificaciones presente en el sistema. La clase Broadcast creada es una subclase de la clase BroadcastReceiver.java

Se ha utilizado un tipo específico de Broadcast receiver, WakefulBroadcastReceiver, que va a asegurar que el dispositivo esté "despierto" el tiempo que sea necesario para que termine la ejecución del servicio que se va a lanzar para procesar los mensajes. Si se utilizase un Broadcast receiver tradicional el dispositivo podría entrar en modo de suspensión (sleep mode) antes de terminar de procesar el mensaje.

Por estas razones, la clase implementada **GCMBroadcastReceiver** extenderá de WakefulBroadcastReceiver y tendrá el método necesario para procesar la notificación.

6.2.2.5. AndroidManifest.xml

El archivo AndroidManifest.xml es esencial en cualquier aplicación Android. Este archivo se genera de manera automática durante la creación de un nuevo proyecto. Su formato es XML y realiza una descripción exhaustiva de los componentes de la aplicación. Esta descripción consta de aspectos como las clases que los implementan, sus requisitos, los datos que pueden manejar o cuando deben ser ejecutados (si al iniciar la aplicación o bajo orden de algún Intent).

Asimismo, es en este archivo donde se especifican los permisos que son necesarios para la aplicación. Algunos de estos permisos son acceso a los datos de contactos, acceso a internet o realizar llamadas de teléfono, así como controlar el hardware del dispositivo.

En este manifiesto se declara, igualmente, la actividad principal y las diferentes actividades que forman la aplicación. Para las notificaciones push, se declara un Broadcast receiver y el servicio GCMIntentService.

Un fragmento del archivo mencionado de la aplicación se puede apreciar en la siguiente imagen, dando las explicaciones pertinentes en cada parte:

```
<manifest>

    <!-- Esto es para darle permisos de conexion a internet -->
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE" />

    <!-- Aqui se indica que se va a permitir usar el servicio GCM --
>
<uses-permission android:name="android.permission.GET_ACCOUNTS" />
```

```
<uses-permission android:name="android.permission.WAKE_LOCK" />
<uses-permission android:name="android.permission.VIBRATE"/>
<uses-permission android:name="android.permission.GET_TASKS"/>
<uses-permission
android:name="com.google.android.c2dm.permission.RECEIVE" />
<permission android:name="es.epinanab.apptauroyrichard.permission.
C2D_MESSAGE" android:protectionLevel="signature" />
<uses-permission android:name="es.epinanab.
apptauroyrichard.permission. C2D_MESSAGE"/>

<uses-permission android:name="android.permission.READ_PROFILE" />
<uses-permission android:name="android.permission.READ_CONTACTS" />

<application
        android:icon="@drawable/icon"
        android:label="@string/app_name" >

<meta-data android:name="com.google.android.gms.version"
android:value="@integer/google_play_services_version" />


<!-Aquí se indica la actividad principal de la aplicacion->
<activity
android:name=".MainActivity"
android:label="@string/app_name" >
   <intent-filter>
      <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
     </intent-filter>
</activity>

<!-- Esta es la actividad menú del cliente -->
<Activity android:name="es.epinanab.apptauroyrichard.MenuCliente" />
<!-- Esta es la actividad de revisiones -->
<activity android:name="es.epinanab.apptauroyrichard.Revisiones" />
<!-- Esta es la actividad de pedir cita -->
<Activity android:name="es.epinanab.apptauroyrichard.Formulario" />

<!—Se declara que la app va a interceptar los intents enviados por
GCM -->
<receiver android:name=".GCMBroadcastReceiver"
android:permission="com.google.android.c2dm.permission.SEND" >

<intent-filter>
<action android:name="com.google.android.c2dm.intent.RECEIVE" />
<action android:name="com.google.android.c2dm.intent.REGISTRATION"
/>
<category android:name="es.epinanab.apptauroyrichard" />
```

```
</intent-filter>

</receiver>
<service android:name=".GCMIntentService" />
</application>
</manifest>
```

6.2.3. Detalles de implementación

En este apartado se va a dar una explicación detallada de la implementación de la aplicación móvil, mostrando trozos de código Java para su mejor comprensión.

6.2.3.1. Llamadas entre actividades

Las llamadas entre actividades se realizan a través de los Intent. Un Intent es un elemento esencial de comunicación entre los diferentes componentes que forman Android. Se entienden como las peticiones que se envían entre los distintos elementos de una aplicación. A través de un Intent se puede mostrar cualquier actividad, iniciar un servicio, enviar un mensaje Broadcast, entre otras.

En el siguiente código, por un lado, se muestra el código para pasar de una actividad a otra y, por otro, el código que se introduce dentro del método OnCreate de la actividad a la que se ha llamado.

```
<!—Inicio de una nueva actividad-->
Intent i = new Intent(); //Creamos el intent
  i.putExtra("Nombre", result[0]);
  i.putExtra("matricula",result[1]);
  i.setClass(MainActivity.this, MenuCliente.class);
  startActivity(i);
```

```
public void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.menu);//Se cambia el layout
<!—Aquí se recuperan las variables enviadas desde la otra actividad-
->
String nombre = getIntent().getStringExtra("Nombre");
String matriculavehiculo = getIntent().getStringExtra("matricula");

. . . (Resto de código)
```

```
Button btn_pedircita = (Button)findViewById(R.id.cita); //Se
localiza el botón en la vista
btn_pedircita.setOnClickListener(new OnClickListener(){
public void onClick(View v) {
. . . (Resto de código)
}
```

Cada actividad cuando llama a otra crea un nuevo Intent. Mediante la propiedad putExtra() del Intent se envían los datos necesarios a la nueva actividad introduciéndolas en variables. Para lanzar la nueva actividad se utiliza el método SetClass, indicando la clase desde donde se lanza y la actividad que se va a lanzar. Por último, se utiliza el comando startActivity para pasar de una actividad a otra.

Todas las actividades tienen el método OnCreate, que es el método que se ejecuta primero cuando se inicia la actividad. Para ello, dentro de dicho método se meten todas las instrucciones que se van a ejecutar cuando se cree dicha actividad. Por tanto, dentro de este método se cambia de Layout con el método setContentView (R.layout.menu) indicándole, en este caso, que se quiere cambiar al Layout menu.xml. A continuación, se recogen las variables que se pasan al crear la nueva actividad con el método getIntent().getStringExtra("nombre_variable").

Igualmente, dentro de este método también se define el Listener de los botones que estén en el Layout de la actividad. El Listener es el encargado de escuchar al botón e indicará a la aplicación cuándo lo ha pulsado el usuario. Para ello, primero se localiza el botón en la lista, como se puede ver en el anterior código, introduciéndolo en una variable. Por último, se define el Listener del botón introduciendo dentro de la función Onclick el código que se debe ejecutar una vez pulsado el botón.

6.2.3.2. Servicios web

Para poder conectar la aplicación móvil con el servidor web se emplean servicios web. Estos servicios ofrecen mecanismos de comunicación estándares entre diferentes aplicaciones, que interactúan entre sí para mostrar información dinámica al usuario. Para otorgar interoperabilidad y extensibilidad entre estas aplicaciones, y que al mismo tiempo sea posible su combinación para realizar operaciones complejas, es necesaria una arquitectura de referencia estándar.

Por ende, los servicios web llaman a unas funciones desarrolladas en PHP que realizan la comunicación entre el servidor y la aplicación. En estas funciones se realiza la petición correspondiente al servidor y devuelve los datos a la aplicación móvil.

Por tanto, en la aplicación web se utiliza la tecnología http para poder comunicar la aplicación web con el servidor web.

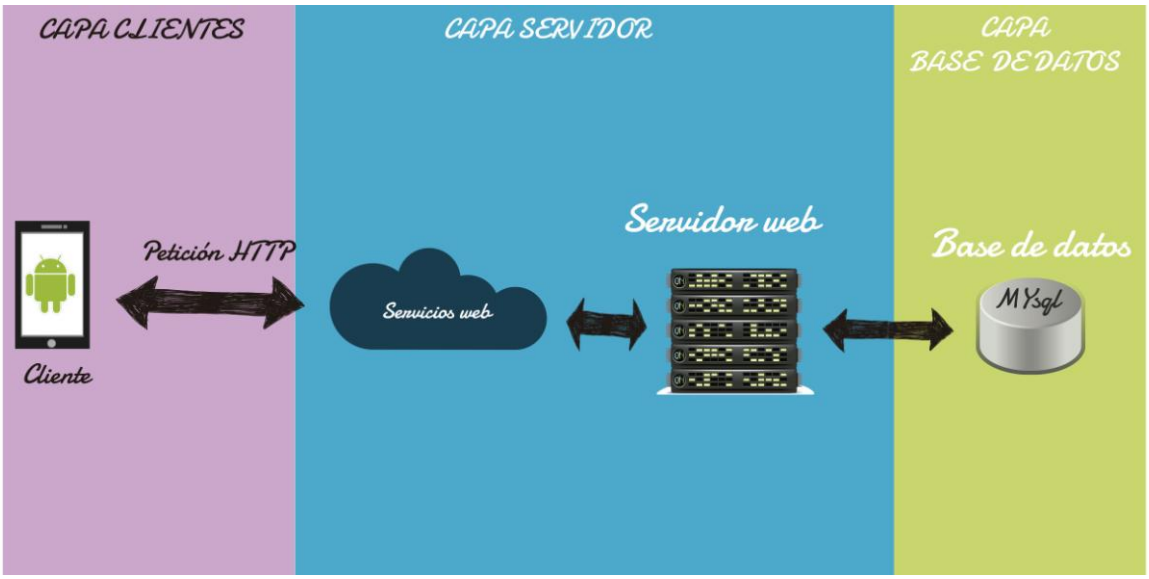


Ilustración 31. - Servicios web

Estos servicios web son utilizados sobretodo para acceder a la base de datos del servidor web.

El acceso a la base de datos se utiliza en la clase MainActivity para guardar en la base de datos del servidor el ID del GCM obtenido. Igualmente, también en esta misma clase se utiliza para comprobar que el DNI y la contraseña , introducidos por el usuario, son correctos.

También se utiliza en la clase Revisiones para obtener los datos de todas las revisiones que aparecen en la tabla de este apartado. En la clase MenuCliente se utiliza para comprobar si la reparación se ha terminado e indicarlo en la parte inferior del menú.

Por otro lado, también se han utilizado los servicios web para poder mandar emails desde la aplicación, como es en el caso de la clase Formulario, que se utilizan los servicios web para conectar con un archivo php del servidor y que éste realice el envío del email.

El código utilizado en las llamadas de acceso a la base de datos es el siguiente:

```
public String httpGetData(String mURL) {
   String response="";//Aqui se guarda la respuesta que se va a
devolver
mURL=mURL.replace(" ","%20"); // Es para cambiar los espacios de la
```

```
url por %20
HttpClient httpclient = new  DefaultHttpClient();
HttpGet httpget = new HttpGet (mURL);


try {
ResponseHandler<String> responsehandler = new
BasicResponseHandler();
response = httpclient.execute(httpget, responsehandler);


} catch (ClientProtocolException e) {
      …(Resto de código)
}
return response; //Devuelve lo obtenido de la llamada al php
}
```

```
data=httpGetData("http://proyecto.tauroyrichard.com/wp-content/App/
identificarseregistrados.php?dni="+dni+"&pwd="+pwd);
```

Como se puede apreciar en el código anterior, por un lado, se encuentra la función httpGetData (String) y en el siguiente bloque aparece la manera de invocar dicha función. Para llamar a esa función, se debe introducir la URL del archivo PHP y en esa URL añadir las variables que se desean pasar a dicho archivo PHP.

La función httpGetData, en primer lugar, crea la variable string en la que se devuelve la información obtenida del PHP. En segundo lugar, sustituye los posibles espacios que pueda haber en la url por %20. Por último, mediante la tecnología http conecta con el archivo PHP del servidor.

Los archivos PHP que conectan con la base de datos, lo hacen de igual manera que la indicada en el apartado 6.1.5.3 y el archivo php que realiza el envío del email de la aplicación móvil lo hace, también, de igual manera que lo indica el apartado 6.1.5.7.

6.2.3.3. Notificaciones Push

La aplicación móvil necesita contar con la capacidad de notificar a los usuarios determinados eventos, que ocurren fuera del dispositivo móvil, en este caso en la aplicación web (Cuando el administrador introduce un aviso de finalización de la reparación).

Como ya se comentó en la arquitectura de la App móvil, para poder realizar estas notificaciones Google proporciona la posibilidad de implementar notificaciones de tipo push.

Una notificación push se basa en que el servidor web inicie el proceso de notificación, pudiendo realizarlo en el mismo momento que se produce el evento, mientras que el cliente (App móvil) se dedica a "esperar" los mensajes, sin estar continuamente consultando al servidor para ver si hay alguna novedad, y sin tener que permanecer en conexión constante con éste.

Todo esto se logra introduciendo un nuevo actor en el proceso, un servidor de mensajería push (GCM), que hará de intermediario entre la aplicación web y la aplicación móvil. La misión de este servidor intermedio es la de recibir las notificaciones enviadas desde las aplicaciones web y hacerlas llegar a las aplicaciones móviles instaladas en los dispositivos correspondientes. Para ello, es necesario que el servidor conozca la existencia de ambas aplicaciones. Esto se consigue mediante un "protocolo" bien definido de registros y autorizaciones entre todos los actores que intervienen en el proceso.

En primer lugar, se crea una cuenta en Google Developers Console. En dicha cuenta se crea un proyecto y Google proporciona el número de proyecto que será el llamado "Sender ID", necesario para el registro de la aplicación móvil en GCM. Por último, se obtiene una Server API Key que será utilizada por la aplicación web cuando envíe una notificación Push al GCM (Capítulo 6.1.5.8).

Como ya se ha mencionado, para asegurar una correcta comunicación entre los tres actores que forman parte de la notificación push es necesario un protocolo de registros y autorizaciones que asegure la seguridad y calidad de todo el proceso.

En la siguiente imagen se resume el tráfico de la notificación push.

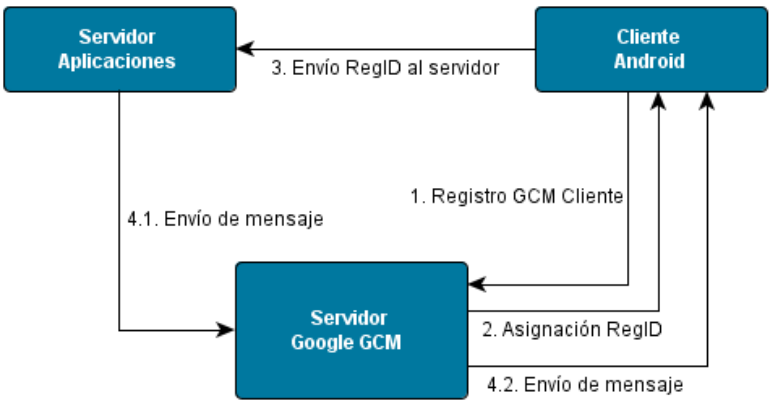


Ilustración 32. - Tráfico de una notificación push [46]

Las acciones del diagrama se podrían agrupar en tres grandes bloques. El primero sería el registro del dispositivo en el GCM, que serían el paso 1, 2 y 3. El segundo sería el envío de notificaciones por

parte del servidor web, que sería el paso 4.1, que se ha comentado en el apartado 6.1.5.8. El tercer bloque es el paso 4.2. que es el envío del mensaje del servidor GCM al dispositivo móvil. A continuación, se van a explicar los pasos indicados en el diagrama.

- Registrar el dispositivo en el GCM

El bloque primero es el de los pasos 1, 2 y 3, indicados en el diagrama. La aplicación Android se registra en el GCM como cliente capaz de recibir mensajes de dicho servicio. Esto se realiza en la actividad MainActivity.java porque es la actividad de identificación del usuario. Es por esto que el código mostrado en esta sección está introducido dentro de dicha actividad.

```
//Comprobar si está instalado Google Play Services
if(checkPlayServices()){
gcm = GoogleCloudMessaging.getInstance(MainActivity.this);

//Obtener el Registration ID guardado
regid = getRegistrationId(context);

//Si no se dispone de Registration ID comenzar el registro
if (regid.equals("")) {
TareaRegistroGCM tarea = new TareaRegistroGCM();
tarea.execute(etdni.getText().toString());
}else{
enviarDispositivoYaRegistrados();}
```

En primer lugar, se verifica que el dispositivo tenga Google play services, de lo contrario las notificaciones no funcionarán. En segundo lugar, se revisa si está almacenado el código de registro debido a que se ha realizado una ejecución anterior. Si no se dispone de código, se realiza el nuevo registros de la aplicación.

Para comprobar si se tenía un id de registro del GCM se utiliza el método getRegistrationId(), indicado en el código siguiente.

```
private String getRegistrationId(Context context) {
SharedPreferences prefs = getSharedPreferences(
MainActivity.class.getSimpleName(),Context.MODE_PRIVATE);

<!—Se obtiene el registration id-->
String registrationId = prefs.getString(PROPERTY_REG_ID, "");

<!--Si no hay id de registro, la aplicación se sale del método-->
if (registrationId.length() == 0){
Log.d(TAG, "Registro GCM no encontrado.");
return "";}
```

```
<!—Recuperar las preferencias compartidas-->
String registeredUser =prefs.getString(PROPERTY_USER, "user");
int registeredVersion = prefs.getInt(PROPERTY_APP_VERSION,
Integer.MIN_VALUE);
long expirationTime =prefs.getLong(PROPERTY_EXPIRATION_TIME, -1);
sdf.format(new Date(expirationTime));
int currentVersion = getAppVersion(context);

<!—Tres comprobaciones-->
if (registeredVersion != currentVersion) {
Log.d(TAG, "Nueva versión de la aplicación."); return "";}

else if (System.currentTimeMillis() > expirationTime) {
Log.d(TAG, "Registro GCM expirado."); return "";   }

if (!etdni.getText().toString().equals(registeredUser)) {
Log.d(TAG, "Nuevo nombre de usuario."); return "";  }

return registrationId; }//Se devuelve el id de registro, en caso de
que lo haya.
```

Lo primero que se hace es recuperar la preferencia PROPERTY_REG_ID, guardada en shared preferences. Si no está se sale inmediatamente del método para proceder a un nuevo registro. Se ha optado por guardar el registro id del dispositivo en las preferencias compartidas (Shared preferences). Shared Preferences es una clase que representa una colección de preferencias. Android proporciona este método diseñado específicamente para administrar este tipos de datos.

Si ya hay un id de registro almacenado se puede seguir usando sin necesidad de registrar el dispositivo de nuevo (se devuelve como resultado para que no se vuelva a realizar el registro). Sin embargo, hay tres situaciones diferentes en las que se debe volver a realizar el registro para garantizar que la aplicación pueda seguir recibiendo mensajes sin ningún problema, que son las sentencias condicionales contempladas en el anterior código:

- Si el nombre de usuario ha cambiado (If (!etdni.getText(). toString(). equals(registeredUser))).
- Si la versión de la aplicación ha cambiado (if (registeredVersion != currentVersion)).
- Si se ha sobrepasado la fecha de caducidad del código de registro. (if (System.currentTimeMillis() > expirationTime)).

Para verificar esto, el método recuperará cada una de las preferencias compartidas, realizará las comprobaciones indicadas y, en caso de que alguna de ellas se cumpla, saldrá del método sin devolver el antiguo registration_id para que se vuelva a realizar el registro.

Si se ha comprobado que no existe ningún registration_id se procede al registro de la aplicación móvil en el GCM. Este registro corresponde el paso 1 del diagrama. Este procedimiento se realiza, como se indica en el código siguiente, mandando el SENDER_ID, que es el ID del proyecto creado previamente en Google Developers Console. El paso dos consiste en recibir un código de registro (Regid), si el registro se ha completado satisfactoriamente.

```
private class TareaRegistroGCM extends
AsyncTask<String,Integer,String>
{
protected String doInBackground(String... params) {
   String msg = "";
try {
    ...(Resto del código)
//Registro en los servidores de GCM
 regid = gcm.register(SENDER_ID);

//Guardar en la aplicación web los datos del registro
registrardispositivo(regid);

 //Guardar el ID en Shared preferences
setRegistrationId(context, params[0], regid);

            }
            catch (IOException ex)
            {
              . . .(Resto de código)
            }
              . . .(Resto de código)
```

El paso 3 consiste en enviar, desde la aplicación móvil a la aplicación web, el código de registro GCM recibido. Éste hará las veces de identificador único del dispositivo móvil en el servidor, de forma que la aplicación web pueda indicar más tarde el ID del dispositivo móvil concreto al que desea enviar un mensaje. La aplicación web, por tanto, almacenará en la base de datos el id de registro del dispositivo móvil en GCM. Esto lo realiza la función registrardispositivo(regid) cuya implementación es mediante servicios web, conectando con un archivo PHP del servidor y almacenando el regid en el mismo.

Por último, si todo ha ido bien, se guardan los nuevos datos de registro (usuario, registration_id, versión de la aplicación y fecha de caducidad) como preferencias compartidas. Esto se realiza mediante el método setRegistrationId(), comentado a continuación.

```
private void setRegistrationId(Context context, String user, String
regId)
{
SharedPreferences prefs = getSharedPreferences( MainActivity.class.
getSimpleName(), Context.MODE_PRIVATE);
  int appVersion = getAppVersion(context);
  SharedPreferences.Editor editor = prefs.edit();
  editor.putString(PROPERTY_USER, user);
  editor.putString(PROPERTY_REG_ID, regId);
  editor.putInt(PROPERTY_APP_VERSION, appVersion);
  editor.putLong(PROPERTY_EXPIRATION_TIME,
  System.currentTimeMillis() + EXPIRATION_TIME_MS);
    editor.commit();}
```

- Recibir y procesar notificaciones Push

El último bloque del diagrama es el paso 4.2, correspondiente al envío de mensajes por parte del servidor GCM a la aplicación móvil. Para poder recibir mensajes en la aplicación móvil se ha implementado un Broadcast Receiver que se encarga de recibir los mensajes, llamado GCMBroadcastReceiver y un servicio, llamado GCMIntentService, que se encarga de procesar dichos mensajes.

Como ya se ha mencionado, el Broadcast receiver extiende del WakefulBroadcastReceiver y se implementa el método onReceive para llamar al servicio de procesamientos de mensajes que es el GCMIntentService.

```
public class GCMBroadcastReceiver extends WakefulBroadcastReceiver{
public void onReceive(Context context, Intent intent)     {
ComponentName comp = new ComponentName(context.getPackageName(),
GCMIntentService.class.getName());
startWakefulService(context, (intent.setComponent(comp)));
setResultCode(Activity.RESULT_OK);}
}}
```

Para el servicio se ha creado la clase GCMIntentService que extiende de IntentService. Se implementa el método onHandleIntent y en él se comprueba el tipo de mensaje recibido y los extras del mismo. Por último, se ejecuta el método mostrarNotificacion() pasándole los extras obtenidos del mensaje. La última línea de código llama al método completeWakefulIntent() del

Broadcast receiver, indispensable para que el dispositivo pueda volver a entrar en modo sleep, para no consumir más batería de la necesaria.

```
public class GCMIntentService extends IntentService {
...(Código)
protected void onHandleIntent(Intent intent){
GoogleCloudMessaging gcm = GoogleCloudMessaging.getInstance(this);
String messageType = gcm.getMessageType(intent);
Bundle extras = intent.getExtras();
if (!extras.isEmpty())
        {
  if (GoogleCloudMessaging.MESSAGE_TYPE_MESSAGE.equals(messageType))
            {
      mostrarNotificacion(extras);
}
}
GCMBroadcastReceiver.completeWakefulIntent(intent);
}
...(Resto de código)
}
```

El método mostrarNotificacion se utiliza para crear el aviso en la barra de tareas del dispositivo móvil. La clase utilizada para la generación de las notificaciones en la barra de tareas es la NotificationCompat.Builder y se le han asignado todas las propiedades necesarias a través de su método set (El icono, el texto, la vibración, el sonido, entre otras).

En segundo lugar, se establece la actividad a la cual se dirige al usuario cuando éste pulse la notificación. Para ello, se ha construido un objeto PendingIntent, que es el que contiene la información de la actividad asociada a la notificación y que se lanza al pulsar sobre ella.

Una vez configuradas todas las opciones de la notificación, ésta se genera llamando al método notify() de Notification Manager pasándole como parámetro un identificador único de la notificación y el resultado del builder, que se ha construido previamente y que se obtiene llamando a su método build().

```
public void mostrarNotificacion(Bundle extras)  {

Uri alarmSound =
RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);

NotificationManager mNotificationManager = (NotificationManager)
getSystemService(Context.NOTIFICATION_SERVICE);

NotificationCompat.Builder mBuilder = new
```

```
NotificationCompat.Builder(this)
.setSmallIcon(android.R.drawable.ic_menu_manage)
.setContentTitle("Alerta Talleres Tauro & Richard")
.setContentText("La reparación de tu vehículo está finalizada")
.setTicker("Notificación de Talleres Tauro y Richard")
.setVibrate(new long[] { 1000, 1000, 1000, 1000, 1000 })
.setLights(Color.RED, 3000, 3000)
.setSound(alarmSound)
.setAutoCancel(true);

Intent notIntent =  new Intent(this, MenuCliente.class);
notIntent.putExtra("Nombre",extras.getString("nombre"));
notIntent.setAction(extras.getString("nombre"));
notIntent.putExtra("matricula",extras.getString("matricula"));
notIntent.setAction(extras.getString("matricula"));

PendingIntent contIntent = PendingIntent.getActivity(this, 0,
notIntent, 0);

mBuilder.setContentIntent(contIntent);
mNotificationManager.notify(NOTIF_ALERTA_ID, mBuilder.build());
    }
}
```

A continuación, se muestra una imagen de la llegada de la notificación al dispositivo y de la actividad que se abre cuando se pincha en la misma.

Cuando se manda una notificación de finalización de la reparación desde la aplicación web al GCM, éste manda una notificación al dispositivo móvil. Cuando el usuario pulsa sobre dicha notificación se le redirige al menú cliente de la App móvil, sin necesidad de volver a registrarse, indicando en la parte inferior de la pantalla un aviso de que la reparación del vehículo ha sido finalizada.

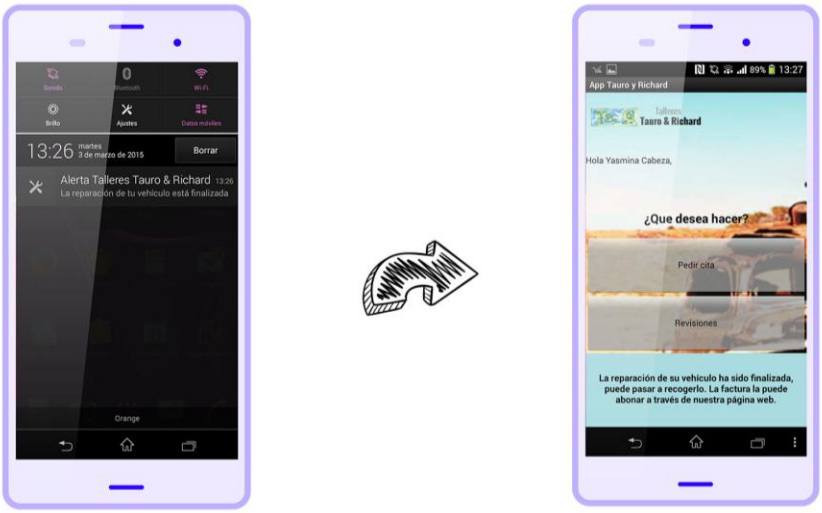


Ilustración 33. - Notificaciones en el dispositivo móvil

6.2.3.4. Realización de tablas dinámicas

En la clase Revisiones.Java se obtienen los datos necesarios de la base de datos mediante servicios web y se genera una tabla dinámica para mostrarlos al cliente.

```
<!—Servicios web -->
datos=httpGetData("http://proyecto.tauroyrichard.com/wp-
content/App/revisiones.php?matricula="+matricula);


tabla = (TableLayout)findViewById(R.id.tabla);
cabecera = (TableLayout)findViewById(R.id.cabecera);
layoutFila = new TableRow.LayoutParams(…);
layoutId = new
TableRow.LayoutParams(200,TableRow.LayoutParams.MATCH_PARENT);


layoutTexto = new
TableRow.LayoutParams(200,TableRow.LayoutParams.MATCH_PARENT);


layoutsigcambio= new
TableRow.LayoutParams(200,TableRow.LayoutParams.MATCH_PARENT);
agregarCabecera();
agregarFilasTabla(datos);
```

```
    public void agregarCabecera(){


. . .  . (Resto de código)


txtultcambio.setText(rs.getString(R.string.ultcambio));
```

```
txtultcambio.setGravity(Gravity.CENTER_HORIZONTAL);
txtultcambio.setTextAppearance(this,R.style.etiqueta);

txtultcambio.setBackgroundResource(R.drawable.tabla_celda_cabecera);
txtultcambio.setLayoutParams(layoutTexto);

fila.addView(txtultcambio);
cabecera.addView(fila);

. . .  . (Resto de código)
}
```

```
public void agregarFilasTabla(String datos){

int n=0;
while(n <result.length){

fila = new TableRow(this);
fila.setLayoutParams(layoutFila);
txtultcambio = new TextView(this);

. . .  . (Resto de código)
txtultcambio.setGravity(Gravity.CENTER_HORIZONTAL);
txtultcambio.setTextAppearance(this,R.style.interiortabla);
txtultcambio.setBackgroundResource(R.drawable.tabla_celda);
txtultcambio.setLayoutParams(layoutTexto);
fila.addView(txtultcambio);
tabla.addView(fila);
 . . . (Resto de código)
}
 }
```

En el código anterior aparecen fragmentos de código utilizados para generar la tabla dinámica. Por un lado, hay dos funciones agregarCabecera y agregarFilasTabla que se usan para agregar la cabecera de la tabla y agregar las filas a dicha tabla, respectivamente.

En el código del inicio se genera la tabla y los Layout de la cabecera y filas y luego se van llamando a las dos funciones indicadas previamente para ir agregando dinámicamente filas a la tabla, en función de los datos obtenidos de la base de datos.

6.2.3.5. Calendario dinámico

La aplicación móvil tiene un botón que permite a los clientes pedir cita a través de la misma. Esta funcionalidad se encuentra dentro de la actividad Formulario.java. En el formulario correspondiente hay una opción para escoger el día de la cita, utilizando un calendario que proporciona la App.

Para realizar esta funcionalidad se ha utilizado un componente llamado Datepicker. Este componente crea cuadro de diálogo en el que el usuario podrá escoger la fecha que desee.

Para su implementación se ha utilizado el siguiente código.

```
mDateDisplay=(EditText) findViewById (R.id.dia);//Casilla de la fecha
mPickDate = (Button)findViewById(R.id.botoncalend);//Boton mostrar calendario
//Listener del botón de mostrar calendario
 mPickDate.setOnClickListener(new OnClickListener(){
   public void onClick(View v) {

       showDialog(DATE_DIALOG_ID);}//Muestra el cuadro de dialogo
});

final Calendar c=Calendar.getInstance();
mYear=c.get(Calendar.YEAR);
mMonth=c.get(Calendar.MONTH);
mDay=c.get(Calendar.DAY_OF_MONTH);
```

```
//Clase Dialog que crea el cuadro de diálogo
protected Dialog onCreateDialog(int id){//Clase dialog creada dentro de la actividad
switch (id){
case DATE_DIALOG_ID:
return new
DatePickerDialog(this,mDateSetListener,mYear,mMonth,mDay);
}return null; }
```

```
//Método para actualizar la fecha en la casilla de la fecha
public void updateDisplay() {
mDateDisplay.setText(new StringBuilder()
.append(mDay).append("-")
.append(mMonth + 1).append("-")
.append(mYear).append(" ")); }
```

```
//El Listener de cuando se ha seleccionado una fecha en el
calendario
DatePickerDialog.OnDateSetListener mDateSetListener=
      new DatePickerDialog.OnDateSetListener() {
public void onDateSet(DatePicker view, int year, int monthOfYear,
int dayOfMonth) {

mYear=year;
mMonth=monthOfYear;
mDay=dayOfMonth;
updateDisplay();
}
};
```

En primer lugar, dentro del método OnCreate se introduce el Listener del botón Mostrar calendario. Cuando el usuario pulse dicho botón se ejecuta la línea "showDialog (DATE_DIALOG_ID)" que mostrará el cuadro de diálogo del calendario. Para ello, dicha función invoca a la clase Dialog que se crea dentro de la actividad. Esta clase se define indicando las instrucciones a realizar en su método OnCreate.

En última instancia, una vez que el usuario ha elegido la fecha, se introducen en las variables mYear, mMonth y mDay, el año, mes y día escogidos cuando el usuario escoge la fecha. Por tanto, se ejecuta el Listener que se produce cuando el usuario escoge una fecha. El Listener es DatePickerDialog.OnDateSetListener() y se encarga de modificar la fecha en el campo de texto asociado, llamado mDateDisplay. Para ello utiliza la función UpdateDisplay() definida previamente, que se encarga de construir la fecha en formato AA-MM-DDDD.

En la imagen siguiente aparece el formulario de pedir cita con el calendario y cómo la casilla correspondiente cambia de valor de fecha al escoger el usuario una fecha.

Ilustración 34. - Calendario de la App móvil

6.2.4. Diseño de la interfaz gráfica

La interfaz gráfica es la parte del sistema que se presenta al usuario, comunicándole información y capturando datos introducidos por el usuario en un mínimo proceso. Es primordial que la interfaz tenga características amigables para el usuario final.

A pesar de la posibilidad del diseño e implementación de una interfaz gráfica con el propio lenguaje de Java resultaría extremadamente costoso para el desarrollador. No obstante, Android se apoya en el uso de los ficheros XML para este propósito proporcionando dos beneficios:

- Simplificar la labor de los programadores lo cual conllevará la realización de mejores interfaces que mejoren la usabilidad y la experiencia final.
- La separación de la representación gráfica del código propio de la aplicación, facilitando la labor de futuras actualizaciones de la aplicación, ya sean realizadas por el autor como posibles futuros desarrolladores que tengan que reinterpretar y modificar el código.

Para el diseño de estas interfaces, mediante el lenguaje XML, se encuentran algunas herramientas disponibles en el mercado para su desarrollo. Sin embargo, a pesar de que estas herramientas en un principio pueden parecer viables y que facilitarían la labor, se ha decidido no utilizar ninguna de ellas. Es conveniente llevar a cabo la implementación de los ficheros XML desde el inicio porque

darán como resultado una interfaz gráfica mucho más depurada y cuyos elementos estén perfectamente definidos, puesto que deben guardar una estrecha relación con el código Java.

Los archivos XML se utilizan para la declaración de Layouts y otros elementos de los que va a hacer uso la aplicación, para su correcto funcionamiento. En Android los archivos de Layout codificados en XML se consideran recursos y se guardan dentro del directorio res/Layout del proyecto.

Cada archivo XML está formado por un árbol de elementos que especifican la manera en la que los elementos de la interfaz y contenedores se acomodarán para definir la parte visual de un objeto. Los atributos de cada elemento en el XML se llaman propiedades y describen cómo deben verse los elementos y cómo debe comportarse un contenedor, así como características que se quieran añadir a ese elemento visual (Tipo de letra, tamaño, el ancho y largo de cada elemento o contenedor…etc.).

En la siguiente imagen se muestra cómo se desarrollan las interfaces mediante los archivos XML en eclipse. La imagen de la izquierda es el "Graphical Layout" en la que eclipse nos permite diseñar la interfaz al gusto del desarrollador y luego genera el código, como se puede apreciar en la página derecha. De la misma manera, se puede generar primero el código en la parte derecha y, posteriormente, visitar la pestaña Graphical Layout para poder visualizar la interfaz final.

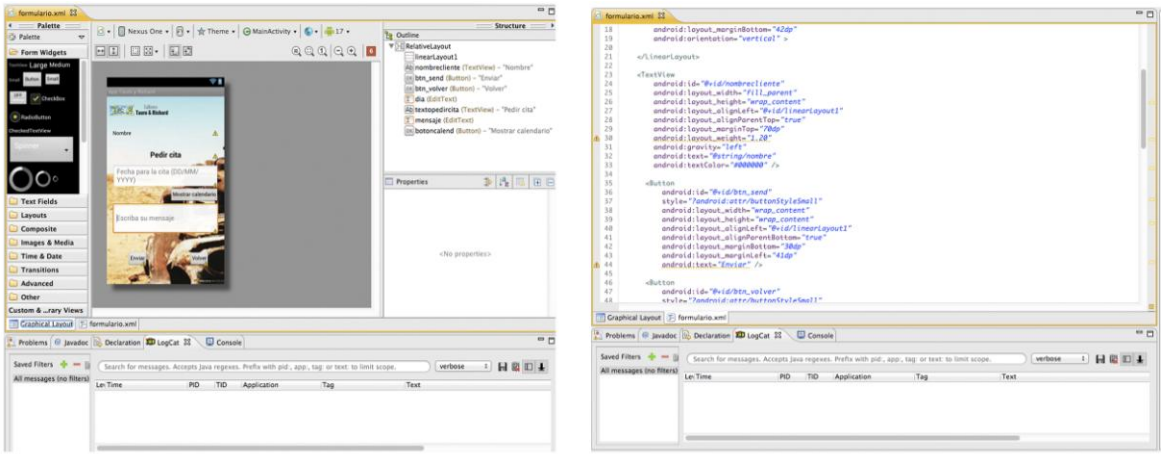


Ilustración 35. - Creación de archivos XML de Android

Una de las particularidades del desarrollo e implementación de aplicaciones destinadas a dispositivos móviles Android es que el usuario puede cambiar la orientación del dispositivo (vertical-portrait/horizontal-landscape). Esto conlleva que la App realizada se pueda visualizar de dos formas distintas en un mismo dispositivo.

Puesto que ambos formatos son muy distintos, para que la aplicación pueda explotar al máximo las posibilidades de cada formato, es necesario tener dos versiones de interfaz gráfica.

Para definir la vista en landscape (vista horizontal) es necesario crear una carpeta llamada layout-land dentro del directorio res. Dentro de esta carpeta se van a crear los mismos archivos XML que en la carpeta res/Layout, comentada previamente. Por tanto, se van a tener dos carpetas res/Layout y res/layout-land con archivos que se van a llamar igual pero el contenido de cada uno será diferente.

En el siguiente código se puede observar el código del archivo principal.xml guardado en la carpeta res/Layout y el de principal.xml guardado en la carpeta res/layout-land.

```
<!—principal.xml guardado en res/Layout-->
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:background="@drawable/background"
android:orientation="vertical" >
<TextView
        android:id="@+id/display"
        android:layout_width="match_parent" />
. . .(Resto del código)
```

```
<!—principal.xml guardado en res/Layout-land -->
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:background="@drawable/fondolandscape"
android:orientation="horizontal" >
<TextView
        android:id="@+id/display"
        android:layout_width="match_parent"/>
. . .(Resto del código)
```

En la siguiente imagen se puede apreciar cómo la App móvil desarrollada se adapta a la vista landscape.

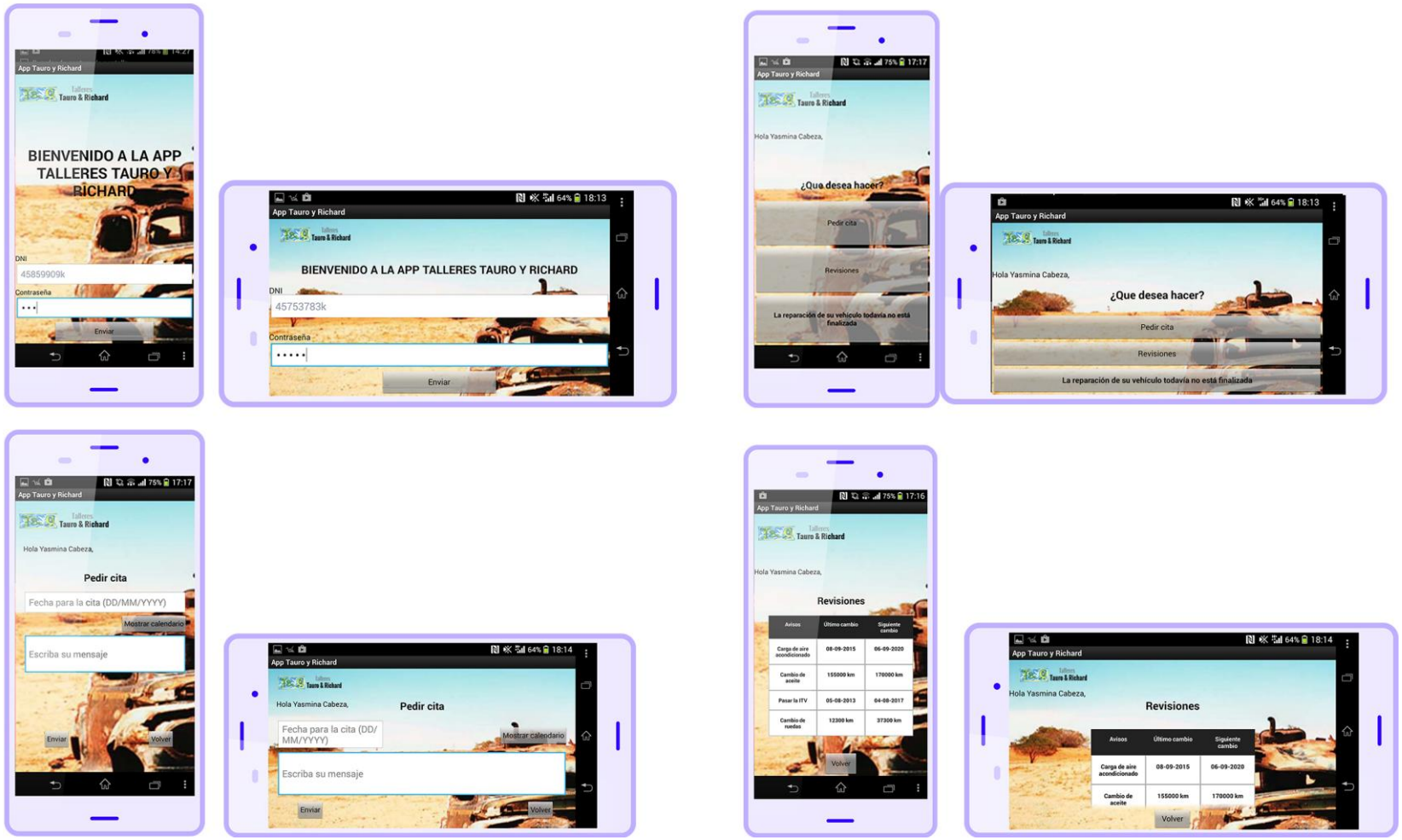


Ilustración 36. - App con rotación de imagen

A continuación, se va a presentar el diseño de la interfaz mediante diagramas de navegación genéricos, y las debidas capturas de pantalla de la App (Ilustración 37).

Los mapas de navegación sirven para visualizar de forma rápida y simple cómo será la navegación a través de la aplicación. Esto es, permiten observar las diferentes interfaces que conforman el software, además de saber cómo se pasa de unas a otras y poder ver la información mostrada en cada una de ellas.

En este mapa de navegación se contemplan las diferentes pantallas a las que podrá acceder el cliente de la App. La pantalla principal sirve para identificarse y poder pasar al menú principal. Desde el menú principal se podrá acceder a la página de pedir cita y a la de consulta de revisiones.

A través de la pantalla de pedir cita, el usuario podrá escoger el día de la cita, a través de un calendario, rellenar el campo del mensaje y enviarlo a la empresa.

Mediante la pantalla de revisiones, el cliente puede conocer los kilómetros o la fecha de la siguiente revisión de su vehículo.

Por último, cuando el administrador desde la aplicación web mande un aviso, se genera una notificación en el dispositivo móvil. Cuando se pulsa sobre ella, reconduce al menú de la aplicación, indicando en él que la reparación de su vehículo ha sido finalizada.

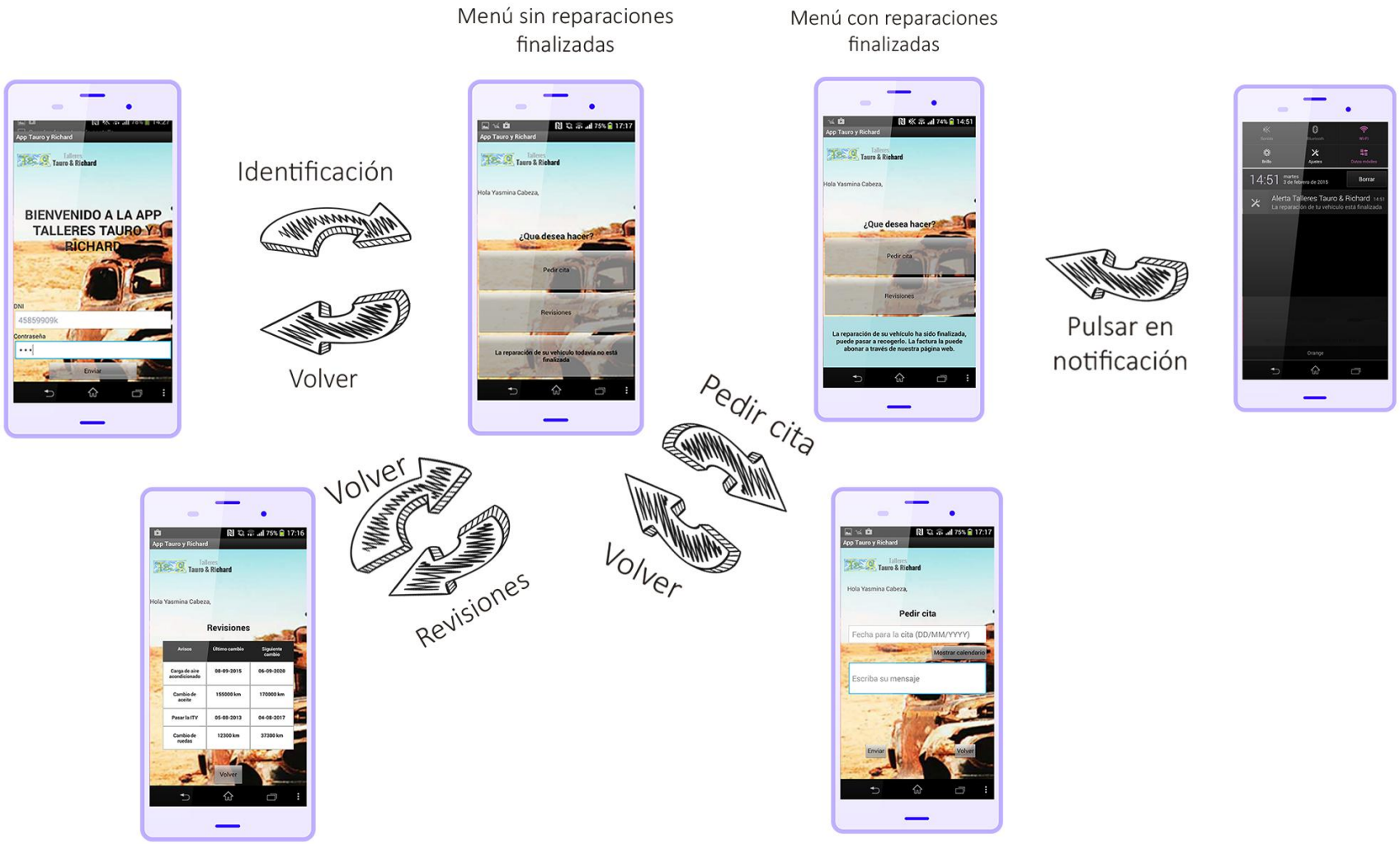


Ilustración 37. - Mapa de navegación de la aplicación móvil

7. VERIFICACIÓN Y EVALUACIÓN

La realización de una correcta integración y ejecución de pruebas va a permitir obtener un producto de gran calidad.

7.1. Objetivos

El objetivo de las pruebas, que serán comentadas a continuación, es el de verificar que todos los puntos que han sido implementados durante el proceso de diseño e implementación del proyecto, funcionan conforme a las especificaciones que han sido dadas en la captura de requisitos.

7.2. Pruebas realizadas

En esta parte se va a comentar las distintas pruebas que se han llevado a cabo en las diferentes partes implementadas, además de los resultados obtenidos durante la ejecución de las pruebas.

7.2.1. Pruebas relacionadas con el servidor

Una vez instalado el servidor en la nube, se verifica que la página funcione correctamente conectándose a la misma desde el propio ordenador y desde otros ordenadores.

De esta forma se verifica que la aplicación funciona correctamente y muestra la página alojada por defecto.

Una vez realizada esta prueba, se procede a subir la aplicación web al servidor. Una vez cargada se vuelve a realizar la prueba anterior para verificar que la página ha sido subida correctamente. Si ha ido bien se muestra la página de inicio de la aplicación.

7.2.2. Pruebas relacionadas con la aplicación web

Una vez llegados a este punto, la página está funcionando y se procede a probar todos y cada uno de los distintos módulos implementados, para verificar que cumplen con las especificaciones dadas.

Prueba nº 1.	Usuario y contraseña correctos (CUU1)
Descripción	Realizar la autenticación del usuario administrador con contraseña correcta.
Resultado esperado	Se espera que la lectura de la contraseña funcione correctamente y que la aplicación de acceso a la administración.
Resultado obtenido	Se le da acceso al sistema al administrador.

Prueba nº 2.	Usuario y contraseña incorrectos (CUU1)
Descripción	Realizar la autenticación del usuario administrador con contraseña incorrecta.
Resultado esperado	Se espera que la lectura del password sea incorrecta y por tanto no de acceso al sistema al administrador.
Resultado obtenido	No se cumple el resultado esperado. El sistema da acceso al administrador, a pesar de que la password sea incorrecta
Solución	Se corrige una sentencia condicional, cuya condición se había puesto que si la contraseña era incorrecta también se le permitía el acceso.

Prueba nº 2.1	Usuario y contraseña incorrectos (CUU1)
Descripción	Realizar la autenticación del usuario administrador con contraseña incorrecta.
Resultado esperado	Se espera que la lectura del password sea incorrecta y, por tanto, no de acceso al sistema al administrador.
Resultado obtenido	Se cumple el resultado esperado

Prueba nº 3.	Cargar matrículas (CUU2)
Descripción	El administrador pulsa el botón "Introducir nueva reparación" y se abre una nueva página con un desplegable con todas las matrículas de los vehículos que hay en el sistema.
Resultado esperado	Se espera que la aplicación muestre un desplegable con todas las matrículas de los vehículos del sistema.
Resultado obtenido	No se cumple el resultado esperado, puesto que aparecen las matrículas repetidas.
Solución	La consulta a la base de datos se había realizado a la tabla tx_reparacion, en lugar de a la tabla tx_vehiculo. Cambiando el valor de la consulta se soluciona el problema de repetición de campos.

Prueba 3.1.	Cargar matrículas correctamente (CUU2)
Descripción	El administrador pulsa el botón "Introducir nueva reparación" y se abre una nueva página con un desplegable con todas las matrículas de los vehículos que hay en el sistema.
Resultado esperado	Se espera que la aplicación muestre un desplegable con todas las matrículas de los vehículos del sistema, sin repetirse ninguna.
Resultado obtenido	Se cumple el resultado esperado.

Prueba nº 4.	Inserción de email incorrecto (CUU2)
Descripción	Se realiza la inserción de la reparación de un vehículo ya registrado o nuevo vehículo en la web y se insertan los datos relacionados y el email con un formato incorrecto
Resultado esperado	Se espera que la aplicación produzca un error de formato de email incorrecto y no realice la inserción del vehículo en la base de datos.
Resultado obtenido	Se cumple el resultado esperado. La aplicación da error de formato de email no válido

Prueba nº 5.	Inserción de teléfono incorrecto (CUU2)
Descripción	Se realiza la inserción de la reparación de un vehículo ya registrado o nuevo vehículo en la web y se insertan los datos relacionados y el teléfono con un formato incorrecto
Resultado esperado	Se espera que la aplicación produzca un error de formato de teléfono incorrecto y no realice la inserción del vehículo en la base de datos.
Resultado obtenido	La aplicación da error de formato de teléfono no válido

Prueba nº 6.	Falta de campos del formulario por rellenar (CUU2)
Descripción	Se realiza la inserción de la reparación de un vehículo ya registrado o nuevo vehículo en la web y no se rellenan todos los campos del formulario.
Resultado esperado	Se espera que la aplicación produzca un error de falta de campos por rellenar y no realice la inserción del vehículo en la base de datos.
Resultado obtenido	La aplicación da error de falta de campos por rellenar.

Prueba nº 7.	Inserción de reparación de vehículo (CUU2)
Descripción	Se realiza la inserción de la reparación de un vehículo ya registrado o nuevo vehículo en la web y se rellenan todos los datos correctamente.
Resultado esperado	Se espera que la aplicación realice la inserción correctamente en la base de datos.
Resultado obtenido	No se cumple el resultado esperado, ya que no se insertan correctamente los datos en la BD.
Solución	Algunos de los nombres de los campos de la tabla se han colocado incorrectamente en la sentencia INSERT. Corrigiendo el error cometido y colocando el nombre de los campos tal y como están en la base de datos se soluciona el error.

Prueba nº 7.1	Inserción de reparación de vehículo (CUU2)
Descripción	Se realiza la inserción de la reparación de un vehículo ya registrado o nuevo vehículo en la web y se rellenan todos los datos correctamente.
Resultado esperado	Se espera que la aplicación realice la inserción correctamente en la base de datos.
Resultado obtenido	La aplicación inserta correctamente los datos en la base de datos.

Prueba nº 8.	Carga de campo fecha (CUU3)
Descripción	Se escoge una matrícula del desplegable y se cargan todas las fechas de las reparaciones de ese vehículo.
Resultado esperado	Se espera que la aplicación cree un desplegable con todas las fechas de las reparaciones del vehículo seleccionado.
Resultado obtenido	No se cumple el resultado esperado. El sistema carga todas las fechas de las reparaciones de todos los vehículos que hay en el sistema, en vez de solamente del vehículo seleccionado.
Solución	Se corrige la sentencia select en la que no se había indicado en el campo Where la matrícula para la que se debían sacar las fechas de las reparaciones. Colocando en la sentencia select "Where matricula='".$matricula."' " se soluciona el error.

Prueba nº 8.1	Carga de campo fecha correctamente (CUU3)
Descripción	Se escoge una matrícula del desplegable y se cargan todas las fechas de las reparaciones de ese vehículo.
Resultado esperado	Se espera que la aplicación cree un desplegable con todas las fechas de las reparaciones del vehículo seleccionado.
Resultado obtenido	Se cumple el resultado esperado, cargándose solamente las fechas de las reparaciones del vehículo escogido.

Prueba nº 9.	Se realiza la modificación de los datos de una reparación a través del formulario que proporciona la aplicación (CUU3) y (CUU5)
Descripción	Se realiza la modificación de los datos de la reparación de un vehículo a través del formulario de la web
Resultado esperado	Se espera que la aplicación realice la modificación correctamente en la base de datos.
Resultado obtenido	No se cumple el resultado esperado porque no se consigue modificar correctamente los servicios.
Solución	Los bucles realizados no eliminaban bien los servicios deseleccionados e insertaban bien los servicios seleccionados. Corrigiendo los bucles que realizan esta operación se consigue obtener el resultado esperado.

Prueba nº 9.1	Se realiza la modificación de los datos de una reparación a través del formulario que proporciona la aplicación (CUU3) y (CUU5)
Descripción	Se realiza la modificación de los datos de un la reparación de un vehículo a través del formulario de la web
Resultado esperado	Se espera que la aplicación realice la modificación correctamente en la base de datos.
Resultado obtenido	La aplicación modifica los datos correctamente en la base de datos.

Prueba nº 10.	Falta de campos del formulario por rellenar (CUU3) y (CUU5)
Descripción	Se realiza la modificación de la reparación de un vehículo ya registrado y no se rellenan todos los campos del formulario.
Resultado esperado	Se espera que la aplicación produzca un error de falta de campos por rellenar y no realice la inserción del vehículo en la base de datos.
Resultado obtenido	No se cumple el resultado esperado. La aplicación inserta los datos en la base de datos sin comprobar que todos los campos estén rellenos.
Solución	Se descubre que la función encargada de verificar que todos los campos estén rellenos le faltan algunos campos por comprobar. Se

	añaden las comprobaciones de todos los campos del formulario y ya se consigue obtener el resultado esperado.

Prueba nº 10.1	Falta de campos del formulario por rellenar (CUU3) y (CUU5)
Descripción	Se realiza la modificación de la reparación de un vehículo ya registrado y no se rellenan todos los campos del formulario.
Resultado esperado	Se espera que la aplicación produzca un error de falta de campos por rellenar y no realice la inserción del vehículo en la base de datos.
Resultado obtenido	La aplicación da error de falta de campos por rellenar.

Prueba nº 11.	Inserción de teléfono incorrecto (CUU3) y (CUU5)
Descripción	Se realiza la modificación de la reparación de un vehículo en la web y se insertan los datos relacionados y el teléfono con un formato incorrecto
Resultado esperado	Se espera que la aplicación produzca un error de formato de teléfono incorrecto y no realice la inserción del vehículo en la base de datos.
Resultado obtenido	La aplicación da error de formato de teléfono no válido

Prueba nº 12.	Cargar servicios (CUU4)
Descripción	El administrador pulsa el botón "Gestionar servicios mano de obra" y se le conduce a otra interfaz con un desplegable en el que deben aparecer todos los servicios que hay en la base de datos y los que tienen el título igual, agrupados por título.
Resultado esperado	Se espera que la aplicación cargue correctamente los datos en el desplegable.
Resultado obtenido	No se cumple el resultado esperado porque se cargan todos los servicios seguidos, sin título.
Solución	Se introduce en el desplegable múltiple creado la etiqueta <optgroup> para poder agrupar y poner un mismo título a aquellos

	servicios con igual título. Realizando esta operación se consigue el resultado esperado.

Prueba nº 12.1.	**Cargar servicios correctamente (CUU4)**
Descripción	El administrador pulsa el botón "Gestionar servicios mano de obra" y se le conduce a otra interfaz con un desplegable en el que deben aparecer todos los servicios que hay en la base de datos y los que tienen el título igual, agrupados por título.
Resultado esperado	Se espera que la aplicación cargue correctamente los datos en el desplegable.
Resultado obtenido	Se cumple el resultado esperado, cargándose los servicios correctamente y agrupados por títulos.

Prueba nº 13.	**Añadir servicio (CUU4)**
Descripción	El administrador inserta un nuevo servicio a través de un pequeño formulario que proporciona la aplicación.
Resultado esperado	Se espera que la aplicación inserte correctamente los datos en la bd y, por tanto, se carguen en la web en la sección de concepto mano de obra.
Resultado obtenido	No se cumple el resultado esperado, puesto que el servicio no se inserta en la base de datos y, por lo tanto, no se carga en el desplegable.
Solución	En el archivo PHP encargado de insertar el servicio no se obtienen correctamente las variables del formulario porque se ha puesto el nombre incorrectamente. Poniendo el nombre de las variables del formulario correctamente en el archivo php se consigue obtener el resultado esperado.

Prueba nº 13.1	**Añadir servicio (CUU4)**
Descripción	Se insertar un nuevo servicio a través de un pequeño formulario que

	proporciona la aplicación.
Resultado esperado	Se espera que la aplicación inserte correctamente los datos en la bd y, por tanto, se carguen en la web en la sección de concepto mano de obra.
Resultado obtenido	Se cumple el resultado esperado

Prueba nº 14.	**Falta de campos por rellenar (CUU4)**
Descripción	Se inserta un nuevo servicio a través de un pequeño formulario sin rellenar todos los campos obligatorios
Resultado esperado	Se espera que la aplicación produzca un error de falta de campos por rellenar y no realice la inserción del servicio en la base de datos.
Resultado obtenido	La aplicación produce un error por falta de campos por rellenar y no se realiza la inserción del servicio en la base de datos.

Prueba nº 15.	**Modificar servicio (CUU4)**
Descripción	Se modifican los datos de un servicio a través de un formulario proporcionado por la web.
Resultado esperado	Se espera que la aplicación modifique en la BD los datos del servicio modificado y, por tanto, se actualicen en la web.
Resultado obtenido	Se cumple el resultado esperado.

Prueba nº 16.	**Eliminar servicio (CUU4)**
Descripción	Se elimina el servicio o servicios deseados por el administrador desde la web.
Resultado esperado	Se espera que la aplicación elimine todos los servicios seleccionados en la base de datos.
Resultado obtenido	No se cumple el resultado esperado, puesto que todos los servicios seleccionados no se consiguen eliminar.

Solución	Se modifica la condición de salida del bucle encargado de recorrer todos los servicios seleccionados y eliminarlos. Una vez modificada la condición se obtiene el resultado esperado.

Prueba nº 16.1	**Eliminar servicio (CUU4)**
Descripción	Se elimina el servicio o servicios deseados por el administrador desde la web.
Resultado esperado	Se espera que la aplicación elimine todos los servicios seleccionados en la base de datos.
Resultado obtenido	Se cumple el resultado esperado.

Prueba nº 17.	**Ver factura (CUU5)**
Descripción	Se genera la factura de la reparación seleccionada en formato pdf.
Resultado esperado	Se espera que la aplicación genere una factura pdf con los datos obtenidos desde la base de datos.
Resultado obtenido	No se cumple el resultado esperado porque se genera una factura pero sin ningún dato rellenado, puesto que la llamada a la base de datos ha fallado.
Solución	Se soluciona modificando la sentencia select realizada con INNER JOIN e indicando correctamente el código de reparación en la misma.

Prueba nº 17.1	**Ver factura (CUU5)**
Descripción	Se genera la factura de la reparación seleccionada en formato pdf.
Resultado esperado	Se espera que la aplicación genere una factura pdf con los datos obtenidos desde la base de datos.
Resultado obtenido	Se cumple el resultado esperado.

Prueba nº 18.	Eliminar reparación (CUU5)
Descripción	Se elimina la reparación deseada por el administrador de la base de datos.
Resultado esperado	Se espera que la aplicación elimine correctamente todos los datos de la reparación seleccionada de la base de datos.
Resultado obtenido	Se cumple el resultado esperado.

Prueba nº 19.	Eliminar historial vehículo (CUU5)
Descripción	El administrador elimina todos los datos relacionados con una reparación.
Resultado esperado	Se espera que la aplicación elimine todos los datos del vehículo y todas las reparaciones de ese vehículo.
Resultado obtenido	No se cumple el resultado esperado, puesto que no se eliminan los servicios de mano de obra asociados a las reparaciones del vehículo.
Solución	Se coloca la sentencia encargada de borrar los servicios asociados a todas las reparaciones del vehículo.

Prueba nº 19.1.	Eliminar historial vehículo correctamente (CUU5)
Descripción	El administrador elimina todos los datos relacionados con una reparación.
Resultado esperado	Se espera que la aplicación elimine todos los datos del vehículo y todas las reparaciones de ese vehículo.
Resultado obtenido	Se cumple el resultado esperado.

Prueba nº 20.	Introducir oferta (CUU6)
Descripción	Se realiza la inserción de todos los campos del formulario correctamente.

Resultado esperado	Se espera que la oferta se introduzca en la base de datos y se vea en la pestaña Ofertas.
Resultado obtenido	No se cumple el resultado esperado porque la oferta no se introduce en Facebook.
Solución	Se corrige el código asociado a la inserción de la oferta en Facebook, introduciendo correctamente el Token proporcionado por Facebook Developers. De esta manera se cumple el resultado esperado.

Prueba nº 20.1.	**Introducir oferta (CUU6)**
Descripción	Se realiza la inserción de todos los campos del formulario correctamente.
Resultado esperado	Se espera que la oferta se introduzca en la base de datos y se vea en la pestaña Ofertas.
Resultado obtenido	La oferta se ha introducido correctamente en la base de datos y se puede visualizar en la pestaña Ofertas y en Facebook.

Prueba nº 21.	**Introducir incorrectamente formato imagen (CUU6)**
Descripción	Se realiza la inserción o modificación de una oferta y el formato de imagen no es el adecuado.
Resultado esperado	Se espera que la aplicación produzca un error de formato incorrecto de imagen y no realice la inserción o modificación de la oferta en la base de datos.
Resultado obtenido	Se produce un error de formato de imagen no válido y la oferta no se introduce ni se modifica y, por tanto, en la pestaña Ofertas no aparece. Tampoco se introduce en Facebook.

Prueba nº 22.	**Falta de campos por rellenar en el formulario (CUU6)**
Descripción	Se realiza la inserción de una oferta pero no se rellenan todos los campos obligatorios ya sea al añadir una nueva oferta o al

	modificarla.
Resultado esperado	Se espera que la aplicación produzca un error de falta de campos por rellenar y no realice la inserción o modificación de la oferta en la base de datos.
Resultado obtenido	Se produce un error de faltan campos por rellenar y la oferta no se introduce ni se modifica y, por tanto, en la pestaña Ofertas y en Facebook no aparece.

Prueba nº 23.	**Cargar datos en Modificar oferta (CUU6)**
Descripción	El administrador pulsa el botón "Modificar oferta" y al elegir una oferta deberían cargarse todos los datos de la oferta seleccionada.
Resultado esperado	Se espera que la aplicación cargue todos los datos de la base de datos de la oferta seleccionada.
Resultado obtenido	No se cumple el resultado esperado porque la imagen no se carga correctamente.
Solución	La ruta de la imagen que se guardaba en la base de datos era incorrecta y por ello la imagen no se cargaba. Modificando el fichero encargado de introducir la ruta de la imagen en la base de datos se soluciona el problema.

Prueba nº 23.1.	**Cargar datos en Modificar oferta (CUU6)**
Descripción	El administrador pulsa el botón "Modificar oferta" y al elegir una oferta deberían cargarse todos los datos de la oferta seleccionada.
Resultado esperado	Se espera que la aplicación cargue todos los datos de la base de datos de la oferta seleccionada.
Resultado obtenido	No se cumple el resultado esperado porque la imagen no se carga correctamente.

Prueba nº 24.	Modificar oferta (CUU6)
Descripción	Se realiza la modificación de una oferta.
Resultado esperado	Se espera que la aplicación modifique correctamente la oferta en la base de datos y en la pestaña Ofertas aparezca la oferta modificada.
Resultado obtenido	Se realiza la modificación correctamente y en la pestaña de Ofertas aparece la oferta modificada.

Prueba nº 25.	Eliminar oferta (CUU6)
Descripción	Se realiza la eliminación de una oferta.
Resultado esperado	Se espera que la aplicación elimine correctamente la oferta de la base de datos y en la pestaña Ofertas no aparezca la oferta.
Resultado obtenido	Se realiza la eliminación de la oferta y en la pestaña de Ofertas no aparecerá.

Prueba nº 26.	Usuario y contraseña del cliente (CUU7)
Descripción	Realizar la autenticación del usuario cliente con usuario y contraseña correctas.
Resultado esperado	Se espera que la lectura del usuario y contraseña funcione correctamente y que la aplicación de acceso al entorno de clientes.
Resultado obtenido	No se le da acceso al entorno de clientes al usuario, a pesar de que el usuario y contraseña sean correctos.
Solución	El nombre del campo contraseña de la base de datos estaba mal colocado en la sentencia select y, por tanto, la sentencia no se ejecutaba correctamente. Cambiando el nombre de dicho campo se consigue obtener el resultado esperado.

Prueba nº 26.1.	Usuario y contraseña del cliente correctos (CUU7)
Descripción	Realizar la autenticación del usuario cliente con usuario y contraseña correctas.
Resultado esperado	Se espera que la lectura del usuario y contraseña funcione correctamente y que la aplicación de acceso al entorno de clientes.
Resultado obtenido	Se le da acceso al entorno de clientes al usuario.

Prueba nº 27.	Usuario y contraseñas del cliente incorrectas (CUU7)
Descripción	Realizar la autenticación del usuario cliente con usuario y contraseña incorrectas.
Resultado esperado	Se espera que la lectura del usuario y contraseña produzca un error de autenticación y que la aplicación no de acceso al entorno de clientes.
Resultado obtenido	El sistema produce un error de acceso al sistema y no permite al usuario acceder al entorno de clientes.

Prueba nº 28.	Recuperar contraseña (CUU10)
Descripción	El usuario, que no recuerda su contraseña, introducirá la matrícula de su vehículo y recibirá en el email una nueva contraseña para poder acceder al sistema.
Resultado esperado	Se espera que el usuario introduzca correctamente la matrícula y le llegue el email correctamente.
Resultado obtenido	El sistema no envía el email al cliente y, por tanto, no puede recuperar la contraseña.
Solución	El código del email estaba escrito erróneamente. Corrigiendo dicho código el email sí le llega al cliente y se cumple el resultado esperado.

Prueba nº 28.1.	Recuperar contraseña (CUU10)
Descripción	El usuario, que no recuerda su contraseña, introducirá la matrícula de su vehículo y recibirá en el email una nueva contraseña para poder acceder al sistema.
Resultado esperado	Se espera que el usuario introduzca correctamente la matrícula y le llegue el email correctamente.
Resultado obtenido	El sistema envía el email correctamente al cliente y, con la nueva contraseña, el cliente ya puede acceder al entorno de clientes.

Prueba nº 29.	Modificación de datos personales (CUU8)
Descripción	El cliente cambiará los datos personales que desee.
Resultado esperado	Se espera que los datos modificados se graben en la base de datos.
Resultado obtenido	El sistema modifica los datos en la base de datos.

Prueba nº 30.	Falta de campos por rellenar en el formulario (CUU8)
Descripción	Se realiza la modificación de los datos de un vehículo pero no se rellenan todos los campos.
Resultado esperado	Se espera que la aplicación produzca un error de falta de campos por rellenar y no realice la modificación de los datos del vehículo en la base de datos.
Resultado obtenido	Se produce un error de faltan campos por rellenar y los datos no se modifican en la base de datos.

Prueba nº 31.	Inserción de email incorrecto (CUU8)
Descripción	El cliente realiza la modificación de sus datos personales en la web y se insertan todos los datos pero el email con un formato incorrecto.
Resultado esperado	Se espera que la aplicación produzca un error de formato de email incorrecto y no realice la modificación de datos en la base de datos.

Resultado obtenido	La aplicación da error de formato de email no válido

Prueba nº 32.	**Inserción del campo contraseña (CUU8)**
Descripción	El cliente, en caso de querer, puede modificar la contraseña para acceder al entorno de clientes.
Resultado esperado	Se espera que la aplicación modifique la contraseña en la base de datos y le envíe un email al cliente con la nueva contraseña.
Resultado obtenido	La aplicación modifica la contraseña del cliente en la base de datos y envía el email correctamente.

Prueba nº 33.	**Ver factura (CUU9) y (CUU11)**
Descripción	Se genera la factura de la reparación seleccionada en formato pdf
Resultado esperado	Se espera que la aplicación genere una factura pdf con los datos obtenidos desde la base de datos.
Resultado obtenido	Se cumple el resultado esperado.

Prueba nº 34.	**Pagar factura con transferencia bancaria (CUU9) y (CUU11)**
Descripción	El cliente que desea pagar la factura, seleccionará la opción transferencia bancaria.
Resultado esperado	Se espera que la aplicación envíe un email al cliente con todos los datos del pago, los datos bancarios de la entidad y la factura de la reparación adjunta en el email.
Resultado obtenido	La aplicación le muestra al cliente la pantalla de agradecimiento por realizar la reparación en el taller, junto con todos los datos del pago y envía un email al cliente con lo mismo.

Prueba nº 35.	Pagar factura PayPal con datos correctos (CUU9) y (CUU11)
Descripción	El cliente que desea pagar la factura, seleccionará la opción pagar con PayPal.
Resultado esperado	Se espera que la aplicación redirija al cliente a la página de PayPal para poder pagar y envíe un email al cliente con todos los datos del pago y la factura de la reparación adjunta en el email y, a continuación, le devuelva a la página de agradecimiento de la aplicación.
Resultado obtenido	Después de realizar el pago a través de PayPal, no se redirige al usuario a la página de agradecimiento de nuestra aplicación.
Solución	Se introduce la URL a la que se le debe redireccionar al usuario en la página oficial de PayPal, en un subapartado que indica un campo para introducir dicha URL. De esta manera se cumple el resultado esperado.

Prueba nº 35.1.	Pagar factura PayPal con datos correctos (CUU9) y (CUU11)
Descripción	El cliente que desea pagar la factura, seleccionará la opción pagar con PayPal.
Resultado esperado	Se espera que la aplicación redirija al cliente a la página de PayPal para poder pagar y envíe un email al cliente con todos los datos del pago y la factura de la reparación adjunta en el email y, a continuación, le devuelva a la página de agradecimiento de la aplicación.
Resultado obtenido	La aplicación le permite al cliente pagar a través de PayPal y le envía un email con los detalles del pago realizado, redireccionándole a la página de agradecimiento de la aplicación.

Prueba nº 36.	Pagar factura PayPal con datos incorrectos (CUU9) y (CUU11)
Descripción	El cliente que desea pagar la factura, seleccionará la opción pagar con PayPal y al introducir los datos, lo hará de forma incorrecta.
Resultado esperado	Se espera que la aplicación redirija al cliente a la página de PayPal

	para poder pagar y éste introduce de manera errónea los datos y se produce un error.
Resultado obtenido	La aplicación genera un error por datos incorrectos de PayPal y el cobró no se realiza y la aplicación no mandará el email correspondiente.

Prueba nº 37.	**Realizar cálculos (CUU13)**
Descripción	Cuando el administrador añade un aviso e introduce los kilómetros del vehículo, la aplicación le calcula automáticamente los siguientes kilómetros de revisión.
Resultado esperado	Se espera que la aplicación realice el cálculo correctamente dependiendo del aviso escogido.
Resultado obtenido	No se cumple el resultado esperado, puesto que hay avisos en los cuales el resultado del cálculo es incorrecto.
Solución	Se corrigen las operaciones y se consigue el resultado esperado.

Prueba nº37.1.	**Realizar cálculos (CUU13)**
Descripción	Cuando el administrador añade un aviso e introduce los kilómetros del vehículo, la aplicación le calcula automáticamente los siguientes kilómetros de revisión.
Resultado esperado	Se espera que la aplicación realice el cálculo correctamente dependiendo del aviso escogido.
Resultado obtenido	Se cumple el resultado esperado.

Prueba nº 38.	**Insertar aviso de cambio (Aceite, pastillas de frenos...) (CUU13)**
Descripción	El administrador rellena todos los campos del formulario necesarios y pulsa aceptar.
Resultado esperado	Se espera que la aplicación al introducir el kilometraje actual se autorrellene el siguiente campo con el resultado correspondiente.

	También se espera que cree el aviso en la base de datos y que cambie los kilómetros actuales del coche de la tabla de datos personales del vehículo.
Resultado obtenido	Realiza lo esperado y la aplicación genera un aviso que genera una notificación en la App Tauro & Richard.

Prueba nº 39.	**Insertar aviso de cambio (Aceite, pastillas de frenos...) con datos incorrectos (CUU13)**
Descripción	El administrador introduce erróneamente los datos en los campos del formulario, introduciendo letras en lugar de números y pulsa aceptar.
Resultado esperado	Se espera que la aplicación genere un mensaje de error y el campo de kilometraje siguiente de revisión no se rellene y que no inserte los datos en la base de datos.
Resultado obtenido	La aplicación genera un error por datos incorrectos y no inserta el aviso en la base de datos.

Prueba nº 40.	**Insertar aviso de cambio (Aceite, pastillas de frenos...) con campos vacíos (CUU13)**
Descripción	El administrador pulsa el botón aceptar del formulario sin haber rellenado todos los campos correctamente.
Resultado esperado	Se espera que la aplicación no inserte los datos en la base de datos y genere un mensaje de error.
Resultado obtenido	La aplicación genera un error por campos incompletos y no inserta el aviso en la base de datos.

Prueba nº 41.	**Insertar aviso de carga de aire acondicionado o pasar la ITV (CUU13)**
Descripción	El administrador introduce la fecha de la ultima revisión y pulsa aceptar.
Resultado esperado	Se espera que la aplicación autorrellene correctamente el campo de

	siguiente fecha y que se genere un aviso en la base de datos.
Resultado obtenido	La aplicación rellena correctamente el campo de siguiente fecha y genera un aviso en la base de datos.

Prueba nº 42.	**Insertar aviso de carga de aire acondicionado o pasar la ITV con datos incorrectos (CUU13)**
Descripción	El administrador introduce la fecha de la ultima revisión incorrectamente, es decir, con formato incorrecto o introduciendo letras en lugar de números o introduciendo una fecha que no existe, y pulsa el botón aceptar.
Resultado esperado	Se espera que la aplicación no rellene el campo de siguiente fecha, generando un mensaje de error.
Resultado obtenido	La aplicación no rellena el campo de siguiente fecha por ser un formato incorrecto, generando un mensaje de error y no introduciendo el aviso en la base de datos.

Prueba nº 43.	**Insertar aviso de carga de aire acondicionado o pasar la ITV con campos vacíos (CUU13)**
Descripción	El administrador deja alguno de los dos campos del formulario vacíos y pulsa el botón aceptar.
Resultado esperado	Se espera que la aplicación genere un mensaje de error y no introduzca el aviso en la base de datos.
Resultado obtenido	La aplicación genera un mensaje de error de campos vacíos y no crea un nuevo aviso en la base de datos.

Prueba nº 44.	**Insertar aviso de finalización de la reparación (CUU13)**
Descripción	Se añade un aviso de finalización de la reparación en la base de datos
Resultado esperado	Se espera que la aplicación añada el aviso correctamente en la base de datos.

Resultado obtenido	No se cumple el resultado esperado, puesto que al añadir un aviso de finalización de la reparación, no llega la notificación al móvil del usuario correspondiente.
Solución	Se introduce la sentencia select para obtener el id de registro del GCM del dispositivo de la tabla correspondiente para poder enviar la notificación al cliente correspondiente. Introduciendo esta sentencia se consigue obtener el resultado esperado.

Prueba nº 44.1.	**Insertar aviso de finalización de la reparación (CUU13)**
Descripción	El administrador escoge la fecha de la reparación que desea finalizar y selecciona Finalizada y pulsa aceptar.
Resultado esperado	Se espera que la aplicación cree un aviso de finalización de reparación.
Resultado obtenido	La aplicación genera un aviso de finalización de reparación en la base de datos, cambia el estado de la reparación a finalizada en la base de datos de reparaciones y envía la notificación al terminal asociado.

Prueba nº 45.	**Insertar aviso de finalización de la reparación pulsando Sin finalizar(CUU13)**
Descripción	El administrador escoge la fecha de la reparación que desea finalizar y selecciona Sin finalizar y pulsa aceptar.
Resultado esperado	Se espera que la aplicación no cree un aviso de finalización de reparación.
Resultado obtenido	La aplicación no genera un aviso de finalización de reparación en la base de datos y tampoco cambia el estado de la reparación a finalizada en la base de datos de reparaciones.

Prueba nº 46.	**Pulsar el botón cancelar (CUU13)**
Descripción	El administrador decide no insertar ningún aviso y pulsa el botón cancelar.

Resultado esperado	Se espera que la aplicación no genere ningún aviso y lleve al administrador a la página de administración de avisos.
Resultado obtenido	La aplicación no genera ningún aviso y redirige al administrador a la página principal de administración de avisos.

Prueba nº 47.	**Eliminar aviso – botón eliminar (CUU13)**
Descripción	El administrador pulsa "Eliminar aviso" y aparece un desplegable para escoger el aviso deseado.
Resultado esperado	Se espera que no se muestre el botón eliminar hasta no haber escogido un aviso.
Resultado obtenido	La aplicación muestra el botón eliminar incluso sin haber escogido ningún aviso, por tanto, si no se escoge aviso y se pulsa el botón se produce un error en el sistema.
Solución	Se introduce una instrucción en el desplegable indicando que solamente cuando se seleccione un aviso aparezca el botón eliminar. Introduciendo estas líneas de código se cumple el resultado esperado.

Prueba nº 47.1.	**Eliminar aviso – botón eliminar (CUU13)**
Descripción	El administrador escoge un aviso.
Resultado esperado	Se espera que la aplicación muestre el botón eliminar.
Resultado obtenido	La aplicación muestra el botón eliminar, solamente tras escoger un aviso, para que el administrador pueda eliminar el aviso.

Prueba nº 48.	**Eliminar aviso (CUU13)**
Descripción	El administrador escoge un aviso y pulsa el botón eliminar.
Resultado esperado	Se espera que la aplicación elimine ese aviso de la aplicación.
Resultado obtenido	La aplicación elimina de la base de datos el aviso seleccionado.

Prueba nº 49.	Ver cámaras web (CUU14)
Descripción	El administrador pulsa el botón Ver cámaras web.
Resultado esperado	Se espera que la aplicación muestre las cámaras web del taller.
Resultado obtenido	La aplicación no muestra al administrador las cámaras web.
Solución	Se corrige el código necesario para la visualización de las cámaras, introduciendo en el mismo la IP pública del router necesaria.

Prueba nº 49.1.	Ver cámaras web (CUU14)
Descripción	El administrador pulsa el botón Ver cámaras web.
Resultado esperado	Se espera que la aplicación muestre las cámaras web del taller.
Resultado obtenido	La aplicación muestra al administrador las cámaras web.

7.2.3. Pruebas relacionadas con la aplicación móvil

Una vez terminadas las pruebas de la aplicación web, se procede a realizar las pruebas de la App móvil para poder constatar que se cumplen las especificaciones dadas.

Prueba nº 50.	Identificación cliente (CUU1)
Descripción	El administrador introduce el DNI y la contraseña y pulsa el botón enviar.
Resultado esperado	Se espera que la App compruebe los datos con la base de datos y que le conduzca a la interfaz de menú del cliente.
Resultado obtenido	La App muestra error de "La App ha dejado de funcionar" por algún error de programación en el código.
Solución	Se corrige la sentencia SELECT del php encargado de realizar las

	comprobaciones en la base de datos.

Prueba nº 50.1.	Identificación cliente (CUU1)
Descripción	El administrador introduce el DNI y la contraseña y pulsa el botón enviar.
Resultado esperado	Se espera que la App compruebe los datos con la base de datos y que le conduzca a la interfaz de menú del cliente.
Resultado obtenido	Se cumple el resultado esperado.

Prueba nº 51.	Identificación cliente con datos erróneos (CUU1)
Descripción	El administrador introduce el DNI y la contraseña incorrectos y pulsa el botón enviar.
Resultado esperado	Se espera que la App compruebe los datos con la base de datos y que no le conduzca a la nueva interfaz.
Resultado obtenido	Se cumple el resultado esperado.

Prueba nº 52.	Cargar datos en Menú cliente (CUU1)
Descripción	Cuando el administrador se identifica y se le reconduce al Menú, si el cliente tiene la reparación finalizada debe aparecer el aviso.
Resultado esperado	Se espera que la App, después de la identificación, compruebe los datos con la base de datos y aparezca el aviso de finalización de reparación, en caso de que lo haya.
Resultado obtenido	No se cumple el resultado esperado porque, aunque haya un aviso de finalización de la reparación, en la App no aparece.
Solución	Se corrige la sentencia encargada de comparar los datos obtenidos de la consulta con la cadena de caracteres que se espera obtener y tras esa corrección se cumple el resultado esperado.

Prueba nº 52.1.	Cargar datos en Menú cliente (CUU1)
Descripción	Cuando el administrador se identifica y se le reconduce al Menú, si el cliente tiene la reparación finalizada debe aparecer el aviso.
Resultado esperado	Se espera que la App, después de la identificación, compruebe los datos con la base de datos y aparezca el aviso de finalización de reparación, en caso de que lo haya.
Resultado obtenido	Sí se cumple el resultado esperado.

Prueba nº 53.	Pedir cita (CUU2)
Descripción	Si se pulsa el botón "Pedir cita" aparece un formulario que el cliente debe rellenar y si pulsa el botón "Mostrar calendario" debe abrirse una nueva ventana con un calendario.
Resultado esperado	Se espera que la App cargue el calendario correctamente.
Resultado obtenido	No se cumple el resultado porque el botón de mostrar calendario no funciona y, por consiguiente, no se carga el calendario.
Solución	Se corrige el código encargado de cargar el formulario y se consigue obtener el resultado esperado.

Prueba nº 53.1.	Pedir cita (CUU2)
Descripción	Si se pulsa el botón "Pedir cita" aparece un formulario que el cliente debe rellenar y si pulsa el botón "Mostrar calendario" debe abrirse una nueva ventana con un calendario.
Resultado esperado	Se espera que la App cargue el calendario correctamente.
Resultado obtenido	Se cumple el resultado esperado.

Prueba nº 54.	Pedir cita (CUU2)
Descripción	Si se pulsa el botón "Pedir cita" aparece un formulario que el cliente debe rellenar y pulsar el botón Enviar.

Resultado esperado	Se espera que la App mande correctamente el email al administrador.
Resultado obtenido	Sí se cumple el resultado esperado.

Prueba nº 55.	**Consultar revisiones (CUU3)**
Descripción	Si se pulsa el botón "Revisiones" aparece una tabla con las revisiones del vehículo correspondiente.
Resultado esperado	Se espera que la App genere una tabla con todas las revisiones del vehículo.
Resultado obtenido	Sí se cumple el resultado esperado.

Prueba nº 56.	**Ver nuevo aviso (CUU4)**
Descripción	El cliente recibe una notificación en su móvil de finalización de la reparación y, al pulsar en ella, debe abrirse la App con los datos del cliente y apareciendo el aviso de finalización de la reparación.
Resultado esperado	Se espera que la App genere el aviso y que al pulsar en él recoja los datos correspondientes de la base de datos y conduzca al cliente al Menú de la aplicación.
Resultado obtenido	No se cumple el resultado esperado porque el cliente al pulsar en la notificación generada no le conduce al menú, si no que aparece error "La App se ha detenido".
Solución	Se revisa la clase GCMIntentService encargada de crear el aviso y las acciones a realizar tras pulsarlo, se corrige el código y se obtiene el resultado esperado.

Prueba nº 56.1.	**Ver nuevo aviso (CUU4)**
Descripción	El cliente recibe una notificación en su móvil de finalización de la reparación y, al pulsar en ella, debe abrirse la App con los datos del cliente y apareciendo el aviso de finalización de la reparación.
Resultado esperado	Se espera que la App genere el aviso y que, al pulsar en él, recoja los

	datos correspondientes de la base de datos y conduzca al cliente al Menú de la aplicación.
Resultado obtenido	Se cumple el resultado esperado.

Prueba nº 57.	**Ver nuevo aviso (CUU4)**
Descripción	El cliente recibe una notificación en su móvil de finalización de la reparación y, al pulsar en ella, debe abrirse la App con los datos del cliente y apareciendo el aviso de finalización de la reparación
Resultado esperado	Se espera que la App genere el aviso y que al pulsar en él recoja los datos correspondientes de la base de datos y conduzca al cliente al Menú con su correspondiente aviso de finalización de la reparación.
Resultado obtenido	No se cumple el resultado esperado. Al pulsar el aviso le reconduce al menú de la App pero no recoge los datos de la base de datos y, en consecuencia, no aparece el aviso de finalización de la reparación.
Solución	Se revisa como se obtienen las variables recibidas en la notificación y se corrigen los errores de programación. Tras la corrección se cumple el resultado esperado.

Prueba nº 57.1.	**Ver nuevo aviso (CUU4)**
Descripción	El cliente recibe una notificación en su móvil de finalización de la reparación y, al pulsar en ella, debe abrirse la App con los datos del cliente y apareciendo el aviso de finalización de la reparación.
Resultado esperado	Se espera que la App genere el aviso y que al pulsar en él recoja los datos correspondientes de la base de datos y conduzca al cliente al Menú con su correspondiente aviso de finalización de la reparación.
Resultado obtenido	Sí se cumple el resultado esperado.

8. CONCLUSIONES Y TRABAJO FUTURO

Tras la realización del proyecto presentaré las conclusiones extraídas sobre lo que considero más relevante de este proyecto y daré una breve explicación sobre los objetivos logrados. Además realizaré un análisis de la desviación de la planificación con respecto a la planificación primera e indicaré algunas de las ampliaciones que podría tener el proyecto.

8.1. Conclusiones finales

El aspecto más relevante de la realización de este proyecto es el aprendizaje de las fases a seguir y la estructura necesaria para el desarrollo y finalización de un proyecto. Estos conocimientos adquiridos los podré aplicar a cualquier proyecto que tenga que desarrollar en un futuro.

El desarrollo de un proyecto de este calibre requiere mucho tiempo, esfuerzo y dedicación, debido básicamente a que se están demostrando los conocimientos adquiridos y aplicando todos los conceptos que he ido aprendiendo durante los años de estudio de la carrera.

Además, la elaboración del proyecto conlleva un aprendizaje que quedará reflejado en el mismo proyecto realizado. En este caso, el conocimiento adquirido durante las prácticas realizadas en empresas me ha ayudado en la realización de este proyecto. Sin embargo, a menudo he tenido que acudir a foros o blogs para poder solventar dudas que surgían de los lenguajes utilizados en el proyecto, tales como HTML, JavaScript y PHP. También he necesitado consultar tutoriales para la realización de la aplicación móvil para Android.

Al tratarse de un proyecto real, que se implantará en la empresa, he aprendido a tratar con los trabajadores de la empresa, manteniendo reuniones y conversaciones a lo largo del periodo de creación. Modificando y agregando información desde el principio hasta el final.

He intentado realizar el proyecto de manera profesional y procurando en todo momento realizar una codificación clara y entendedora con tal de facilitar que otro programador que tenga que realizar un módulo de la aplicación, lo pueda llevar a cabo sin gran dificultad.

Aunque siempre existen aspectos que podrían ser mejorados, considero que los objetivos establecidos inicialmente, se han podido cumplir en gran medida.

8.2. Revisión de los objetivos

Tras concluir de manera definitiva el desarrollo del presente trabajo fin de grado, es el momento de hacer balance del resultado final repasando la trayectoria recorrida para obtener conclusiones, tanto en el aspecto técnico y profesional como en el ámbito personal, además de las habilidades adquiridas durante este tiempo en diversas tecnologías y herramientas.

Es importante comprobar si se han cumplido los objetivos iniciales del proyecto. Es por esto que voy a volver a nombrar cada uno de ellos junto a una breve explicación sobre el resultado final obtenido.

- Herramienta intuitiva y potente, para que la experiencia de usuario sea lo más gratificante posible.

Se ha realizado una aplicación web con un gestor de contenidos WordPress y para comprobar la satisfacción del usuario se han realizado pruebas con usuarios no informáticos. Tras la ejecución de dichas pruebas se ha llegado a la conclusión que es una herramienta de fácil manejo, que algunas prestaciones no son tan accesibles como creía pero, en cómputo global, se puede decir que la experiencia de usuario ha sido de un 9 sobre 10.

- Aplicación accesible desde cualquier lugar donde haya una conexión a internet.

Este objetivo ha sido logrado de manera satisfactoria puesto que la aplicación web está en un servidor de la nube, en vez de en un servidor local, proporcionando la accesibilidad que se quería porque desde cualquier navegador de cualquier parte del mundo se puede acceder a ella.

- Gestión cómoda y fácil de todas las reparaciones.

Para lograr este objetivo se ha realizado una aplicación web en la que se recogen todas las necesidades de gestión que tenía la empresa. En una sola herramienta es posible la gestión completa, por parte del administrador, de todas las reparaciones, así como de introducción de ofertas, visionado de cámaras web…etc.

- Darse a conocer en las redes sociales.

Se ha conseguido este objetivo añadiendo una funcionalidad en la aplicación web que permite al administrador introducir ofertas en la aplicación y que éstas se publiquen en Facebook. De esta manera, gracias a esta aplicación la empresa es capaz de darse a conocer a través de las redes sociales, en concreto a través de Facebook.

- Visualización de las cámaras IP desde cualquier lugar.

Otra de las funcionalidades de la aplicación web es la de permitir al administrador visualizar las cámaras web desde la misma. De esta manera, los trabajadores pueden vigilar la empresa sin necesidad de estar en ella, con tan solo un ordenador con acceso a internet y sin necesidad de instalación de software adicionales.

- Facilitar a los clientes el pago de las facturas sin necesidad de acudir al taller.

Este objetivo se ha logrado añadiendo una funcionalidad a la aplicación web a través de la cual los clientes de la empresa pueden realizar el pago de sus facturas a través de la misma, sin necesidad de moverse de su casa, ya sea mediante transferencia bancaria o con Paypal.

- Conexión completa entre clientes y empresa y que ésta pueda comunicarles la finalización de la reparación.

Para cumplir este objetivo ha sido necesario realizar una aplicación para móvil y que la aplicación web tenga una funcionalidad que permita a la empresa mandar notificaciones a la aplicación móvil. De esta manera, la empresa puede estar en contacto directo con los clientes y puede avisarles de la finalización de la reparación de su vehículo en cualquier momento.

En resumen, la aplicación web final integra un correcto diseño, manteniendo un estilo sencillo, claro y útil, pero a la vez original y creativo, siempre con un toque de distinción. Además, con las funcionalidades añadidas a la misma se han conseguido lograr todos los objetivos de este proyecto. Asimismo, se ha realizado una aplicación móvil con las prestaciones que se exigían en los objetivos y añadiendo algunas funcionalidades extra como la posibilidad de pedir cita a través de la misma o poder ver las revisiones necesarias del vehículo.

Los objetivos personales que se pretendían cumplir en este trabajo fin de grado son los siguientes:

- Aprender a desarrollar aplicaciones Android y afianzar los conocimientos de Java adquiridos hasta ahora.

Como resultado de la realización de la aplicación móvil para el sistema operativo Android he logrado obtener un amplio conocimiento de la plataforma Android, de su funcionamiento interno, de la manera que tiene de gestionar los recursos de un dispositivo móvil, además de aprender a desarrollar aplicaciones para dicha plataforma. Todo esto ha sido posible gracias a la extensa documentación que Google pone a disposición de los desarrolladores, aunque cabe destacar, el importante trabajo de investigación que he realizado al respecto.

- Lograr una buena formación en el desarrollo de módulos, para el gestor de contenidos WordPress, poniendo en práctica todos los conocimientos adquiridos durante toda la carrera de PHP, JavaScript y AJAX.

Gracias a la realización de los módulos implementados para la aplicación web he mejorado mis conocimientos en lenguajes de programación como HTML (para la creación de la interfaz de la Web), MySQL (conexiones con la Base de Datos del proyecto), JavaScript (validación y corrección de formularios) y AJAX (interactuar con la aplicación).

- Estudiar la programación de pasarelas de pago como PayPal.

Debido a que una de las funcionalidades de la aplicación web es proporcionar a los clientes la opción de poder pagar mediante el método de pago PayPal, he podido estudiar la programación de la pasarela de pago PayPal, así como poner en práctica su funcionamiento.

- Instrucción en la integración de cámaras IP en aplicaciones web, mediante redes.

Otra de las funcionalidades de la aplicación web era el poder ver las cámaras web desde la misma. Para ello he tenido que aprender técnicas de redes para poder realizar esta funcionalidad. Entre ellas, he tenido que aprender a abrir puertos en el router, a embeber una página web en otra y aprender el funcionamiento de las cámaras web, así como localizar los puertos necesarios para su visionado.

- Aprender a vincular WordPress con Facebook mediante programación.

Uno de los objetivos de este proyecto era que la empresa se diera a conocer en las redes sociales. Debido a que una de las redes sociales que está más en auge hoy en día es Facebook, se decidió que todas las ofertas que se incluyeran en la aplicación web a su vez se introdujeran en el muro de Facebook del taller. Para ello he tenido que formarme y visitar tutoriales de cómo se programa la conexión de WordPress con Facebook.

Todos los objetivos descritos en este proyecto han sido cumplidos con las expectativas establecidas, manteniendo reuniones periódicas con el tutor del proyecto. Han sido cumplidos, no solo por el hecho de realizar de una forma completa e individual la aplicación, sino por la experiencia adquirida al ser un proyecto real, cambios de requisitos, modificaciones a última hora, reuniones constantes, etc.

A estas alturas puedo decir que se han conseguido cumplir los objetivos perseguidos lo que me provoca una gran satisfacción tanto a nivel personal, porque sé que con entusiasmo y esfuerzo todo se supera, como a nivel profesional, por todo el trabajo que ha supuesto la realización del proyecto durante todo este tiempo sin olvidarme, por supuesto, de todas y cada una de las

asignaturas de la carrera, que, unas más que otras, han contribuido a que este proyecto sea hoy una realidad.

8.3. Desviaciones de la planificación inicial

En el planteamiento inicial de proyecto, se realizó una planificación orientativa de la realización del proyecto, habiendo dividido las tareas en base a los objetivos y el alcance de proyecto. Como en algunas materias no tenía mucha experiencia, se han encontrado algunos problemas difíciles de solucionar, que han llevado a una desviación de la planificación inicial.

En este apartado describiré los principales problemas que hicieron que el proyecto se retrasara, seguido de la planificación final realizada.

8.3.1. Problemas encontrados

Como resumen de estos, se exponen los principales problemas aparecidos durante el desarrollo de la aplicación.

Por un lado, la creación de un plugin para el gestor de contenidos WordPress no es una tarea sencilla puesto que he tenido que aprender cómo se integra código en el código principal de WordPress. Esto se realiza a través de los ganchos (llamados hooks) que son parte de la forma en que WordPress está programado, y su principal función es la de permitir cambiar prácticamente cualquier aspecto de la plataforma sin necesidad de tocar su código fuente original.

Por otro lado, ha sido una tarea ardua aprender a usar el PHP mail para poder adjuntar archivos al enviar emails al usuario.

También me ha resultado muy complicado la integración de cámaras web en la aplicación, puesto que es un tema de redes y en nuestra carrera son pocas las asignaturas que nos lo enseñan. Por tanto, he tenido que acudir a manuales para poder solventarlo. También ha resultado complicado por la incompatibilidad de las cámaras con el sistema operativo Macintosh, por lo que he tenido que recurrir a un ordenador Windows para probar su funcionamiento.

De la misma manera, he tenido que acudir a códigos de programación para integrar PayPal en mi aplicación. También ha sido complicada la inserción de las noticias en Facebook desde la aplicación y lo he logrado acudiendo a manuales y páginas orientadas a Facebook Developers.

Por último, la tarea que más esfuerzo ha supuesto ha sido el desarrollo de la App para Android por mi falta de conocimientos en este ámbito. Esto me ha hecho requerir de gran cantidad de consultas a documentación y realizar muchas pruebas de codificación. También ha resultado complicada la instalación del SDK de Android en Eclipse, por problemas de compatibilidad.

Todos estos problemas causaron un retraso importante en la planificación realizada inicialmente, por lo que se tuvo que ajustar en la medida que fue necesaria.

8.3.2. Planificación final

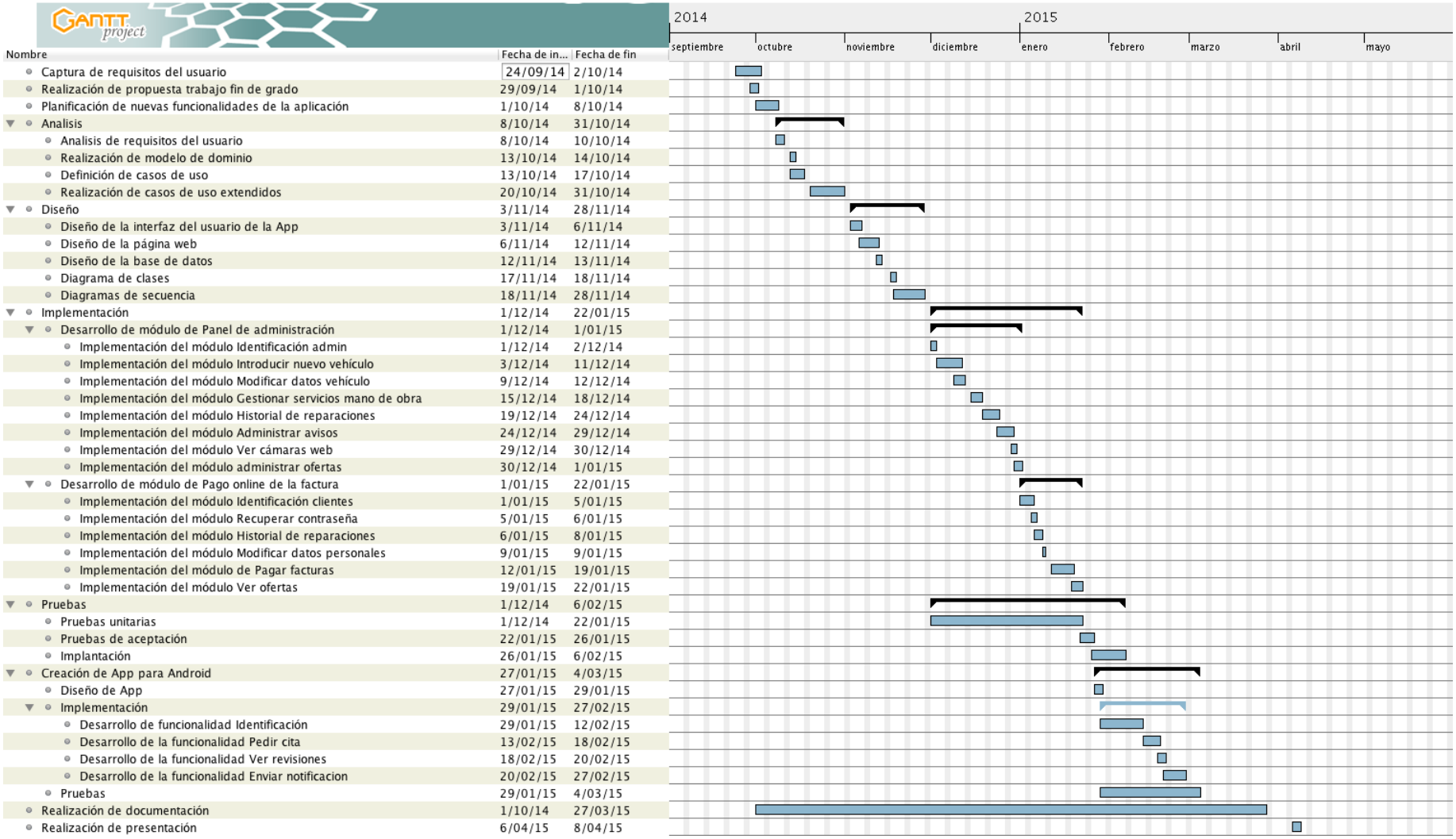
En este apartado se va a comentar el tiempo real dedicado al trabajo fin de grado que dista bastante del tiempo estimado, especificado en el planteamiento inicial. A continuación, muestro el diagrama de Gantt real (Ilustración 38) y comentaré las causas de las desviaciones respecto de la planificación inicial.

Entre la planificación inicialmente propuesta y la final hay una desviación de 2 meses, siendo unos 40 días de estos laborables.

Si se analiza en profundidad el diagrama, se puede observar que ya desde el inicio se empieza a atrasar todo el proyecto. Esto es porque pensaba que la captura de requisitos no me iba a suponer tanto tiempo. La etapa de análisis y diseño también me llevó muchísimo más tiempo del esperado. Una de las razones de retraso fue una mala implementación de la base de datos porque el modelo de dominio era erróneo. Esto supuso tener que cambiar la base de datos y algunas de las funcionalidades ya implementadas.

En general, todas las etapas al final han llevado más tiempo del indicado al principio y esto es porque nunca había realizado un trabajo de esta magnitud y la implementación de los módulos resulto ser más costosa de lo que esperaba. De igual manera, la implementación de la aplicación Android me ha supuesto mucho más trabajo del que pensaba, puesto que ha habido funcionalidades que han sido costosas de implementar y, sobretodo, han sido costosas de detectar los fallos del código.

Concluyendo y con total convicción, puedo afirmar que todo lo aprendido hasta el día hoy, ha superado ampliamente mis expectativas iniciales y me siento afortunada de haber seleccionado esta temática y no otra para mi Trabajo de Fin de Grado. Por otro lado, mi satisfacción se ve reforzada sólo con la idea de que un amplio abanico de opciones se me abren en estos momentos, tanto para mi desarrollo personal como profesional.



Nombre	Fecha de in...	Fecha de fin
Captura de requisitos del usuario	24/09/14	2/10/14
Realización de propuesta trabajo fin de grado	29/09/14	1/10/14
Planificación de nuevas funcionalidades de la aplicación	1/10/14	8/10/14
Analisis	8/10/14	31/10/14
Analisis de requisitos del usuario	8/10/14	10/10/14
Realización de modelo de dominio	13/10/14	14/10/14
Definición de casos de uso	13/10/14	17/10/14
Realización de casos de uso extendidos	20/10/14	31/10/14
Diseño	3/11/14	28/11/14
Diseño de la interfaz del usuario de la App	3/11/14	6/11/14
Diseño de la página web	6/11/14	12/11/14
Diseño de la base de datos	12/11/14	13/11/14
Diagrama de clases	17/11/14	18/11/14
Diagramas de secuencia	18/11/14	28/11/14
Implementación	1/12/14	22/01/15
Desarrollo de módulo de Panel de administración	1/12/14	1/01/15
Implementación del módulo Identificación admin	1/12/14	2/12/14
Implementación del módulo Introducir nuevo vehículo	3/12/14	11/12/14
Implementación del módulo Modificar datos vehículo	9/12/14	12/12/14
Implementación del módulo Gestionar servicios mano de obra	15/12/14	18/12/14
Implementación del módulo Historial de reparaciones	19/12/14	24/12/14
Implementación del módulo Administrar avisos	24/12/14	29/12/14
Implementación del módulo Ver cámaras web	29/12/14	30/12/14
Implementación del módulo administrar ofertas	30/12/14	1/01/15
Desarrollo de módulo de Pago online de la factura	1/01/15	22/01/15
Implementación del módulo Identificación clientes	1/01/15	5/01/15
Implementación del módulo Recuperar contraseña	5/01/15	6/01/15
Implementación del módulo Historial de reparaciones	6/01/15	8/01/15
Implementación del módulo Modificar datos personales	9/01/15	9/01/15
Implementación del módulo de Pagar facturas	12/01/15	19/01/15
Implementación del módulo Ver ofertas	19/01/15	22/01/15
Pruebas	1/12/14	6/02/15
Pruebas unitarias	1/12/14	22/01/15
Pruebas de aceptación	22/01/15	26/01/15
Implantación	26/01/15	6/02/15
Creación de App para Android	27/01/15	4/03/15
Diseño de App	27/01/15	29/01/15
Implementación	29/01/15	27/02/15
Desarrollo de funcionalidad Identificación	29/01/15	12/02/15
Desarrollo de la funcionalidad Pedir cita	13/02/15	18/02/15
Desarrollo de la funcionalidad Ver revisiones	18/02/15	20/02/15
Desarrollo de la funcionalidad Enviar notificacion	20/02/15	27/02/15
Pruebas	29/01/15	4/03/15
Realización de documentación	1/10/14	27/03/15
Realización de presentación	6/04/15	8/04/15

Ilustración 38. - Diagrama final de Gantt

8.4. Posibles ampliaciones y mejoras

Todo sistema informático evoluciona y mejora con el tiempo, introduciendo nuevas características y funcionalidades. Estas modificaciones o ampliaciones se suelen originar debido a la aparición de nuevas necesidades dentro del proyecto o aspectos no contemplados previamente. Además tanto las aplicaciones móviles como las aplicaciones web son tecnologías con un gran potencial de desarrollo.

Actualmente, ya se pueden predecir algunas líneas de desarrollo de cara a una futura versión ampliada.

Creo que la ampliación más notable sería el aumento de funcionalidades en la App móvil creada, puesto que la funcionalidad que tiene ahora es la básica pero a la vez de gran utilidad para los clientes de esta empresa. Sin embargo, considero que sus funcionalidades podrían aumentar notablemente si incluiríamos en la funcionalidad de pedir cita la opción de mostrar el calendario del taller, mostrando los días ocupados y los días libres del taller. Otra posibilidad sería que a través de la App se pudiera realizar el pago de la factura de la reparación, al igual que se hace a través de la aplicación web y, también, que en la App aparecieran las ofertas que aparecen en la aplicación.

La aplicación web podría tener muchas ampliaciones, se le podrían añadir numerosos plugins. Por ejemplo, en métodos de pago se le podría añadir la forma de pago a través del TPV virtual. Otra ampliación importante es que las cámaras web se puedan visualizar desde cualquier navegador y con cualquier sistema operativo.

Otra posible mejora sería que ambas aplicaciones fueran multilingüe ya que generalizaría aún más la aplicación, aspecto que interesa si lo que queremos es obtener un sistema lo más flexible y eficaz posible.

De dichas mejoras o ampliaciones comenzaría realizando la de insertar en la App móvil la posibilidad de realizar el pago de la factura de la reparación a través de la misma. Esta funcionalidad conllevaría varias horas de trabajo, aproximadamente 40 horas de trabajo, puesto que desconozco la manera de realizarlo y debería consultar varios tutoriales. Esta funcionalidad beneficiaría a los clientes facilitándoles el pago de la factura desde cualquier lugar, solamente con disponer del teléfono móvil.

Como segunda mejora añadiría la funcionalidad descrita previamente sobre pedir cita mostrando un calendario con los días ocupados y los días libres. Esta funcionalidad ahorraría mucho tiempo a los administradores del taller puesto que no sería necesario revisar las citas por si acaso el día elegido estaría ocupado, puesto que los clientes a la hora de elegir la cita

sabrán qué días estará libre el taller. Esta funcionalidad conllevaría varias horas de trabajo por el desconocimiento de su realización, aproximadamente 30 horas.

Por último, añadiría el método de pago TPV virtual en la aplicación web que facilitaría el pago a los clientes que no dispongan de una cuenta PayPal o que no deseen pagar mediante transferencia bancaria. Esta funcionalidad conllevaría 10 horas de trabajo, aproximadamente.

Otra de las mejoras a realizar a largo plazo sería la realización de una página web multilingüe. Esta funcionalidad es la de menor prioridad puesto que al ser la aplicación web sobre un taller mecánico es bastante improbable que personas de otras nacionalidades consulten esta web. Conllevaría muchísimas horas, 60 horas aproximadamente.

9. BIBLIOGRAFÍA

Durante la realización de este trabajo se han realizado visitas a blogs y webs y en este apartado queda reflejado algunos de los enlaces que se consideran más interesantes, ya que son los que han permitido descargar los distintos componentes de WordPress, y han ayudado a resolver dudas.

- Referencias web

1. PHP

The PHP Group.

(http://docs.php.net/manual/es/intro-whatis.php). Última modificación en Marzo 2015.

Accedido en Enero 2015.

2. JavaScript

Manish Sharma - WebDevelopersNotes.com

(http://www.webdevelopersnotes.com/manishsharma.php3).

Accedido en Enero 2015.

3. Java

J. Steven Perry, Consultor Director, Makoto Consulting Group, Inc.

(http://www.ibm.com/developerworks/ssa/java/tutorials/j-introtojava1/).Última modificación en Diciembre 2012.

Accedido en Enero 2015.

4. HTML

RI5

(http://www.ri5.com.ar/ayuda03.php).

Accedido en Enero 2015.

5. XHTML

W3C (Consorcio World Wide Web)
(http://www.w3c.es/Divulgacion/GuiasBreves/XHTML).

Accedido en Enero 2015.

6. AJAX

Jesse James Garret

(http://www.uberbin.net/archivos/internet/ajax-un-nuevo-acercamiento-a-aplicaciones-web.php).

Accedido en Enero 2015.

7. CSS

W3C (Consorcio World Wide Web)

(http://www.w3.org/TR/CSS1/).

Accedido en Enero 2015.

8. SQL

1Keydata

(http://www.1keydata.com/es/sql/).

Accedido en Enero 2015.

9. MySQL

W3Schools

(http://www.w3schools.com/php/php_mysql_intro.asp).

Accedido en Enero 2015.

10. Base de datos

Microsoft Office

(https://support.office.com/es-hn/article/Conceptos-b%C3%A1sicos-sobre-bases-de-datos-a849ac16-07c7-4a31-9948-3c8c94a7c204?ui=es-ES&rs=es-HN&ad=HN).

Accedido en Enero 2015.

11. Servicios web

W3C (Consorcio World Wide Web)

(http://www.w3c.es/Divulgacion/GuiasBreves/ServiciosWeb).

Accedido en Enero 2015.

12. Tecnología push

Adolfo Plasencia

(https://aprendizajeubicuo.wordpress.com/2012/04/23/hasta-donde-tecnologia-pull-o-push/). Última modificación en Abril 2012.

Accedido en Enero 2015.

13. PhpmyAdmin

PhpMyAdmin contributors

(http://www.phpmyadmin.net/home_page/).

Accedido en Enero 2015.

14. Servidor Apache

Ing Gustavo Manfredi

(http://www.monografias.com/trabajos93/web-server-ingenieros/web-server-ingenieros2.shtml).

Accedido en Enero 2015.

15. Correo electrónico

Boletín Electrónico Bit 1213." El inventor del Correo Electrónico no pensó en patentar su hallazgo".

(http://www.ecured.cu/index.php/Correo_electr%C3%B3nico).

Accedido en Enero 2015.

16. FTP (File Transfer Protocol)

Fernando Bolo

(https://managefiletransfer.wordpress.com/2011/08/02/mft-b2bconsulting/).Última modificación en Agosto 2011.

Accedido en Enero 2015.

17. Interfaz de usuario

Alberto Lacalle

(http://albertolacalle.com/diseno-interfaz.htm). Última modificación en Julio 2006.

Accedido en Enero 2015.

18. Servidor

Marlnx

(http://es.scribd.com/doc/254518503/Introduccion-a-los-Servidores#scribd).

Accedido en Enero 2015.

19. HTTP

Bligoo

(http://seguridadenlainformacionroberto.bligoo.com.mx/http-https#.VSTfhzusUwg).

Accedido en Enero 2015.s

20. HTTPs

Bligoo

(http://seguridadenlainformacionroberto.bligoo.com.mx/http-https#.VSTfhzusUwg).

Accedido en Enero 2015

21. WWW (World Wide Web)

Berners-Lee, Tim; Bray, Tim; Connolly, Dan; Cotton, Paul; Fielding, Roy; Jeckle, Mario; Lilley, Chris; Mendelsohn, Noah; Orchard, David; Walsh, Norman; Williams, Stuart (December 15, 2004). Architecture of the World Wide Web, Volume One. Version 20041215. W3C.

(http://www.w3.org/TR/webarch/)

Accedido en Enero 2015.

22. Mozilla Firefox

Mozilla Firefox

(http://recursos.fundacionesplai.org/intranet/dinamizadores/linux/firefox/index.htm).

Accedido en Enero 2015.

23. Google Chrome

«Chromium Terms and Conditions». Google Code

(https://www.chromium.org/). Última modificación en Septiembre 2008.

Accedido en Enero 2015.

24. Android

«Industry Leaders Announce Open Platform for Mobile Devices» (en inglés). Open Handset Alliance.

(http://www.openhandsetalliance.com/press_110507.html). Última modificación en Noviembre 2007.

Accedido en Enero 2015.

25. Android software Development

Ed, Burnette Hello, Android: Introducing Google's Mobile Development Platform (3rd ed.). Pragmatic Bookshelf. ISBN 978-1-934356-56-2.

(https://pragprog.com/book/eband3/hello-android). Última modificación en Julio 2010.

Accedido en Enero 2015.

26. WordPress

Wordpress.org

(https://wordpress.org/about/).

Accedido en Enero 2015.

27. Software Eclipse

The Eclipse Foundation

(http://www.eclipse.org/).

Accedido en Enero 2015.

28. Sublime Text

Roxana Falasco (Guía definitiva Sublime Text 2)

(http://falasco.org/guia-definitiva-sublime-text-2). Última modificación en Julio 2012.

Accedido en Enero 2015.

29. Microsoft Word

Alejandra Pereira (Monografías.com)

(http://www.monografias.com/trabajos82/trabajo-word/trabajo-word.shtml).

Accedido en Enero 2015.

30. Microsoft Power Point

Omar Chirinos (Monografías.com)

(http://www.monografias.com/trabajos100/sobre-power-point/sobre-power-point.shtml).

Accedido en Enero 2015.

31. Filezilla

José Juan Fuentes Lorca

(http://dis.um.es/~lopezquesada/documentos/IES_1112/SAD/curso/UT7/ActividadesAlumnos/grupo3/post/post3.html).

Accedido en Enero 2015.

32. Adobe Photoshop

GeoActio

(http://www.geoactio.com/index.php/es/tecnologias/137-adobe-photoshop).

Accedido en Enero 2015.

33. Adobe Illustrator

Edward Trujillo

(https://prezi.com/o23tpp9iluqz/adobe-illustratreitor/). Última modificación en Mayo 2014.

Accedido en Enero 2015

34. ERP (Sistema de planificación de recursos empresariales)

Florencia Chiesa

Metodología para selección de sistemas ERP Reportes Técnicos en Ingeniería de Software Vol. 6 N° 1 (2004), pág. 17-37. ISSN: 1668-3137

(http://www.ucla.edu.ve/dac/departamentos/informatica-II/metodologia-para-seleccion-de-sistemas-erp.PDF).

Accedido en Enero 2015.

35. Tutorial para instalar el plugin de Android en Eclipse.

Android

(http://developer.android.com/sdk/installing/installing-adt.html). Última modificación en Marzo 2015.

Accedido en Enero 2015.

36. Tutorial para hacer la app Android en Eclipse

Kike

(http://www.aprendeandroid.com/l1/instalacion.htm). Última modificación en Marzo 2015.

Accedido en Enero 2015.

37. Tutorial WordPress

WordPress

(https://es.wordpress.org.) Última modificación en Marzo 2015.

Accedido en Octubre 2014.

38. Foro Wordpress

ForoBeta

(http://forobeta.com/wordpress/). Última modificación en Marzo 2015.

Accedido en Octubre 2014.

39. Manual de PHP

The PHP Group

(http://php.net/manual/es/). Última modificación en Marzo 2015.

Accedido en Noviembre 2014.

40. Cómo implementar el pago por Paypal en un sitio web

Blog OpenAlfa

(http://blog.openalfa.com/como-implementar-el-pago-por-paypal-en-un-sitio-web). Última modificación en Marzo 2015.

Accedido en Noviembre 2014.

41. Cómo publicar en Facebook como página mediante su API con PHP

El internauta León

(http://el-internauta-de-leon.blogspot.com.es/2012/12/como-publicar-en-facebook-como-pagina.html). Última modificación en Diciembre 2012.

Accedido en Noviembre 2014.

42. Filas dinámicas en TableLayout Android

Meison Chirinos

(http://www.ameison.net/2012/09/filas-dinamicas-en-tablelayout-android.html).Última modificación en Septiembre 2012.

Accedido en Noviembre 2014.

- Imágenes:

43. La estructura de WordPress

José Aguilar - Experto programador Prestashop y Wordpress.

(http://www.jose-aguilar.com/blog/la-estructura-de-wordpress/). Última modificación en Febrero 2013.

Accedido en Febrero 2015.

44. Estructura de un theme para WordPress

Juan Ignacio Alberola Colomo

(http://www.juanignacioalberola.com/estructura-de-un-theme-para-wordpress/).Última modificación en Febrero 2014.

Accedido en Febrero 2015.

45. Android

Smieh

Fuente original:Anatomy Physiology of an Android

Arquitectura del sistema Android (http://es.wikipedia.org/wiki/Archivo:Android-System-Architecture.svg). Última modificación en Junio 2012.

Accedido en Febrero 2015.

46. Notificaciones Push Android: Google Cloud Messaging (GCM). Introducción

Sgoliver

(http://www.sgoliver.net/blog/notificaciones-push-android-google-cloud-messaging-gcm-introduccion/). Última modificación en Julio 2012.

Accedido en Febrero 2015.

10. ANEXO I . – CASOS DE USO EXTENDIDOS

En este apartado se van a definir de manera detallada los casos de uso del modelo de casos de uso, especificado en el apartado de captura de requisitos. Esta técnica sirve para detallar interacciones entre los actores y las aplicaciones, como respuesta a un evento producido por un actor sobre el sistema, proporcionando un resumen de la tarea realizada.

Los casos de uso extendidos estarán divididos en dos apartados debido a los requerimientos funcionales de cada una de las aplicaciones.

10.1. Aplicación web

En este apartado se van a describir los casos de uso extendidos pertenecientes a la aplicación web. En primer lugar, se encuentran todos los casos de uso del actor "Administrador" y, en segundo lugar, se encuentran los casos de uso pertenecientes al actor "Cliente".

10.1.1. Caso de uso **Identificación admin**

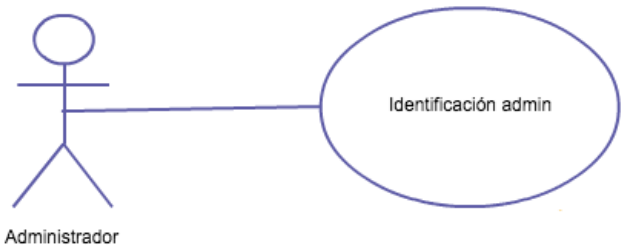


Ilustración 39. - Caso de uso Identificación admin

Nombre: Caso de uso Identificación Admin
Descripción: El administrador deberá introducir su contraseña y pulsar “Entrar” y comprobando que la contraseña sea correcta se le enviará al panel de administración.
Actores: Administrador.

Precondiciones: Ninguna.
Requisitos no funcionales: Ninguno.
Flujo de eventos: 1) El administrador se encuentra en la interfaz (Ilustración 40) en la que aparecerán el campo contraseña y deberá rellenarlo y pulsar "Aceptar" y se comprobará con la base de datos si la contraseña es correcta y se le conducirá a otra interfaz (Ilustración 41). 2) En caso de que la contraseña sea incorrecta se mostrará un mensaje de error (Ilustración 42).
Pos condiciones: Se habrá identificado el administrador y se le llevará a la página de inicio del panel de administración para que elija la acción que desee realizar.
Interfaz gráfica:

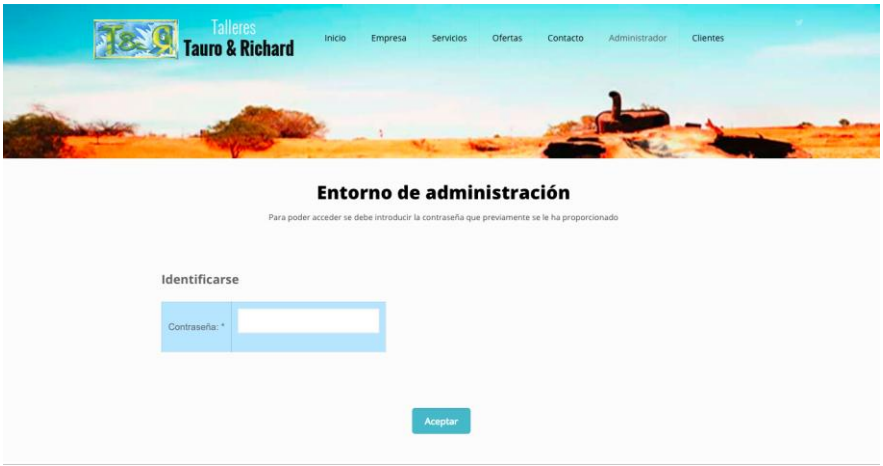


Ilustración 40. - Entorno de administración

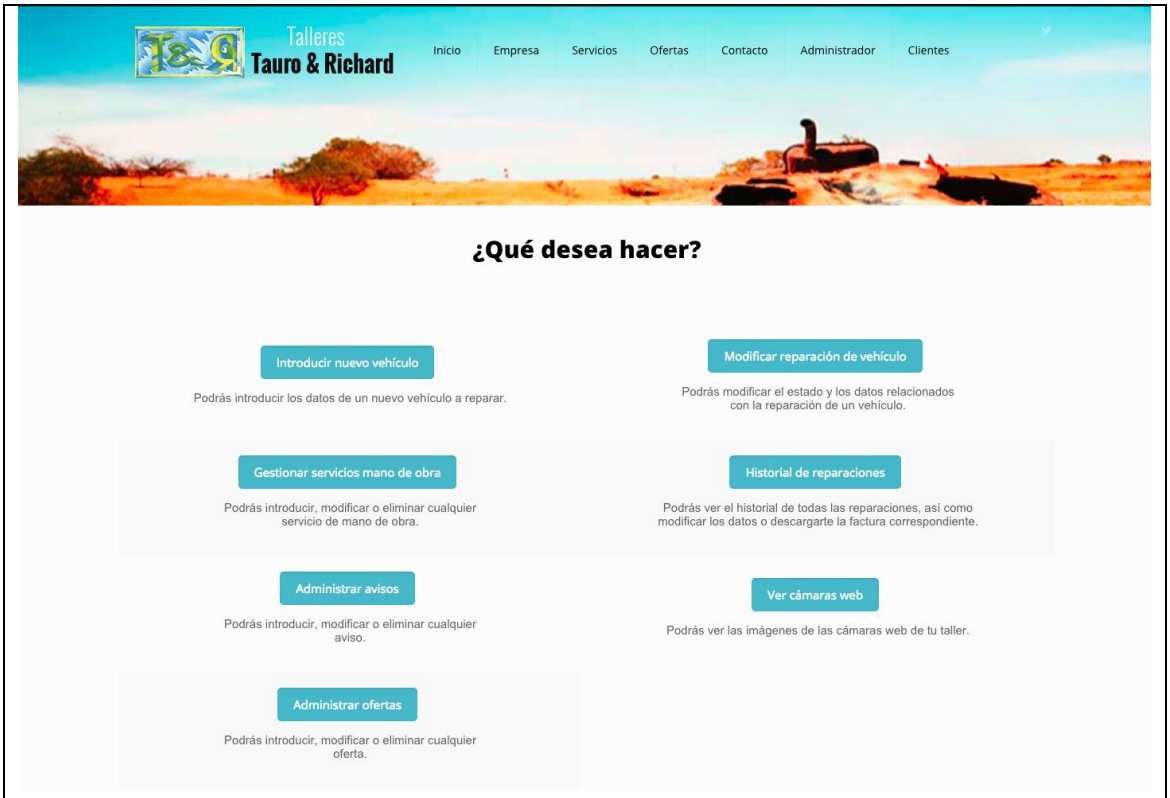


Ilustración 41. - Menú del administrador

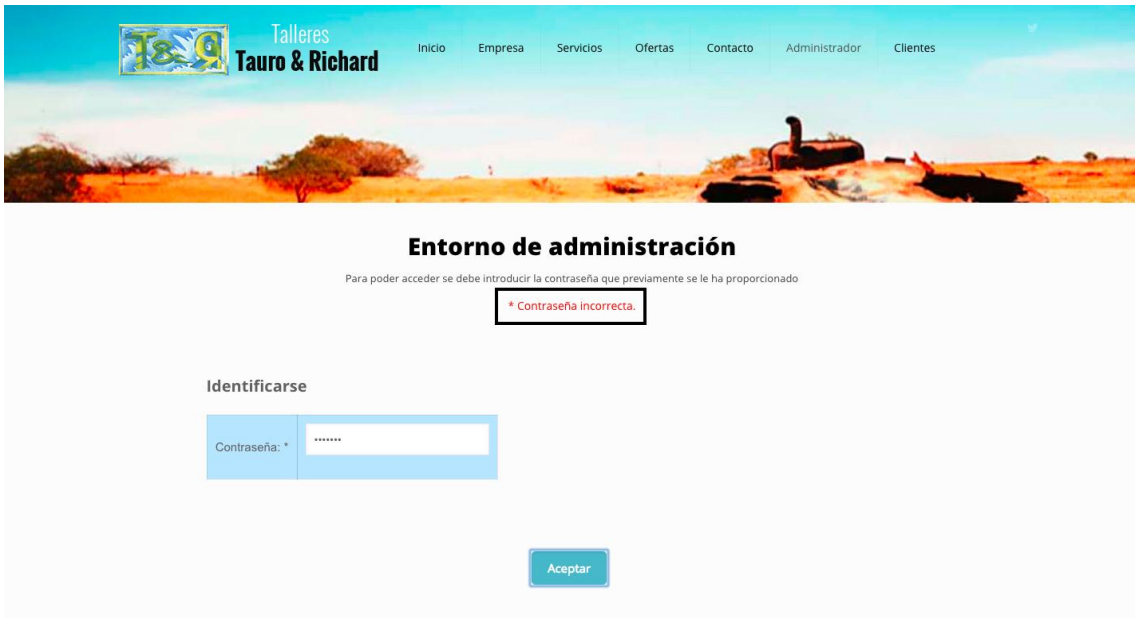


Ilustración 42. - Contraseña del administrador incorrecta

10.1.2. Caso de uso **Introducir nueva reparación**

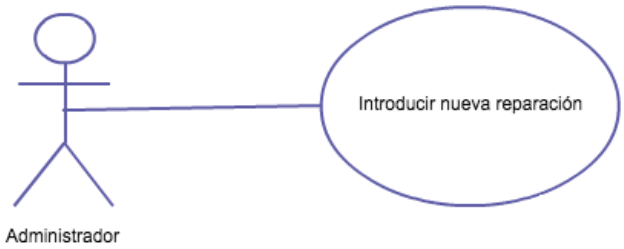


Ilustración 43. - Caso de uso Introducir nueva reparación

Nombre: Caso de uso Introducir nueva reparación

Descripción: El administrador podrá introducir todos los datos relacionados con la reparación de un vehículo nuevo o que ya estuviera registrado en la web.

Actores: Administrador.

Precondiciones: Estar identificado como administrador.

Requisitos no funcionales: Ninguno.

Flujo de eventos:

1) El administrador, que previamente ha introducido una contraseña en la página "Entorno de administración" (Ilustración 44), pulsará el botón "Introducir nueva reparación" (Ilustración 45) que le enviará a otra interfaz (Ilustración 48) que será la interfaz de introducción de todos los datos de la reparación del vehículo.

2) En dicha interfaz, en primer lugar, aparecerá un desplegable en el que pondrá "Escoja la matrícula" (Ilustración 46). En dicho desplegable aparecerán todas las matrículas de los coches que ya han sido insertados en otras ocasiones porque ya se ha hecho alguna reparación del mismo con anterioridad. Por tanto, si la reparación que quiere introducir es de un coche que ya ha sido reparado en este taller antes, se deberá escoger la matrícula correspondiente. De lo contrario se deberá pulsar en el desplegable la opción "Insertar nuevo vehículo" (Ilustración 47).

3) **En caso de escoger una matrícula,** aparecerá una nueva página (Ilustración 48) en la que aparecerán los campos de nombre y apellidos, dirección, código postal, ciudad, provincia, teléfono, DNI, email, modelo del vehículo y kilometraje ya rellenados por los datos que ya se habían insertado con anterioridad, porque ese vehículo ya había realizado alguna

reparación antes en el taller.

3.1.) Por otro lado, aparecerán los campos de CIA y el importe de la reparación (Sin IVA), concepto piezas, precio unitario y importe sin rellenar. En concepto deberá rellenar con el tipo de reparación realizada, así como piezas puestas al vehículo, en precio unitario será el precio por pieza puesta al coche y en importe será el importe total de todas las piezas cambiadas. Estos datos los deberá rellenar el administrador con los datos de la reparación en cuestión.

3.2.) Por último, aparecerá el campo concepto mano de obra en el que aparecerán algunos de los precios estándar de la mano de obra de algunas reparaciones. En este desplegable se podrá elegir más de una opción.

3.3.) Se deberán rellenar todos los campos con asterisco y los campos que previamente aparecían ya rellenados se podrán modificar si así lo desea el administrador.

4) **En caso de escoger la opción "Insertar nuevo vehículo"** aparecerán todos los campos, comentados previamente, pero vacíos para que los rellene el administrador (Ilustración 49).

5) Una vez rellenados todos los campos con asterisco, en cualquiera de las dos opciones, se deberá seleccionar el botón "Aceptar" y si todo es correcto el administrador volverá a la página principal de administrador (Ilustración 45).

5.1.) En caso de que no se hayan rellenado todos los campos, excepto "CIA" o "Concepto mano de obra" que son opcionales, saldrá un aviso de que faltan campos por rellenar (Ilustración 50).

5.2.) En caso de que el email no se introduzca con el formato correcto, saldrá un error de formato de email incorrecto (Ilustración 51).

5.3.) En caso de que el teléfono no se introduzca con un formato adecuado, saldrá un mensaje de error de número de teléfono incorrecto (Ilustración 52).

6) Si el administrador quisiera borrar todos los datos existentes y rellenar el formulario entero, deberá pulsar el botón "Borrar".

Pos condiciones: El administrador introducirá los datos de los vehículos a reparar y estos se almacenarán en la base de datos.

Interfaz gráfica:

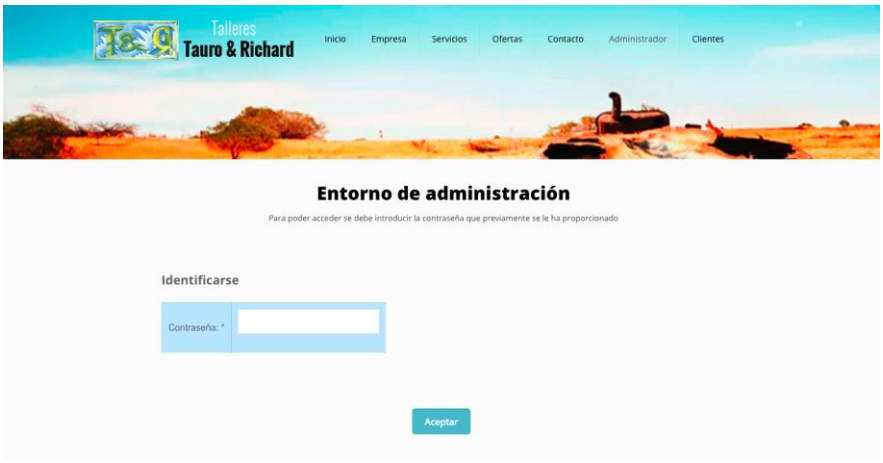


Ilustración 44. - Entorno de administración

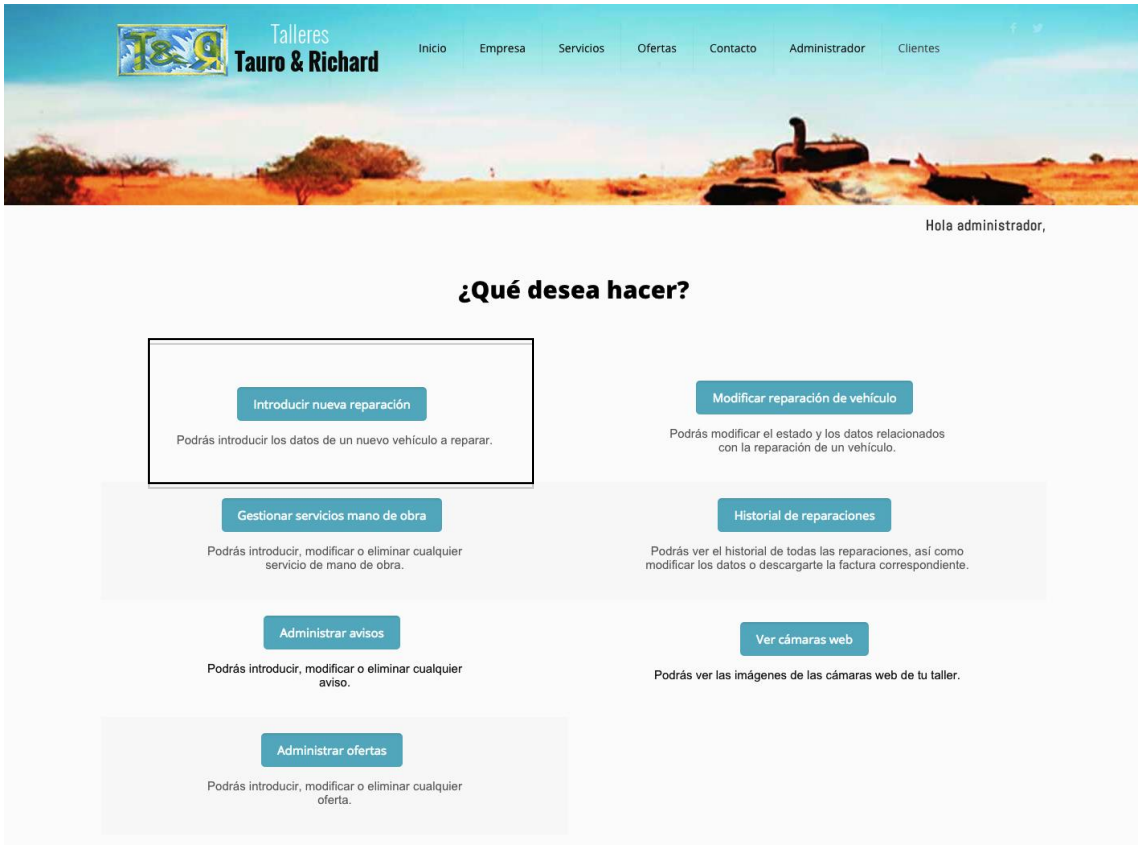


Ilustración 45. - Menú del administrador

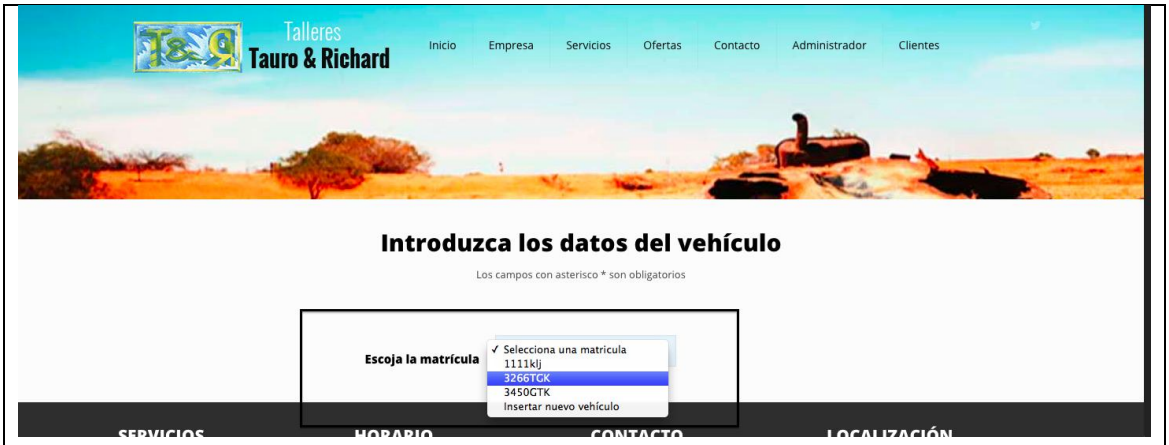


Ilustración 46. – Selección de matrícula del vehículo a reparar

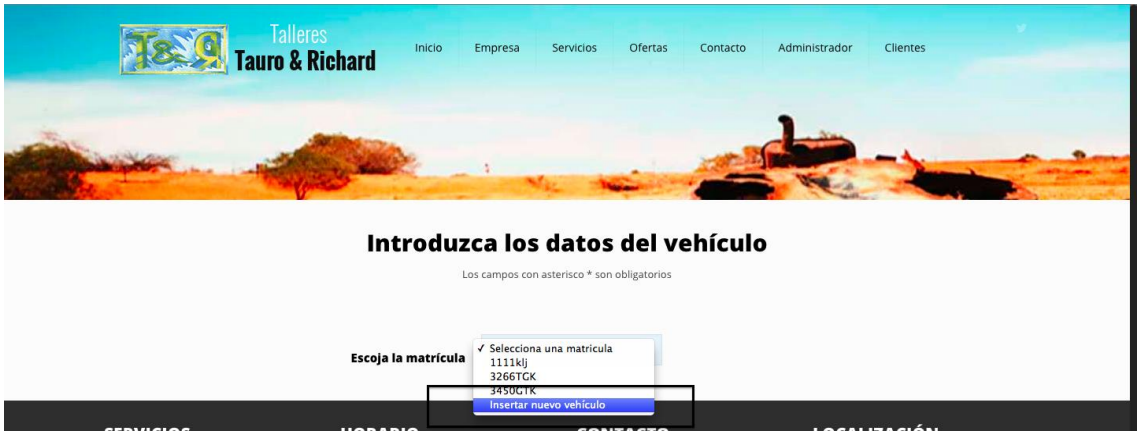


Ilustración 47. - Selección de nuevo vehículo

Ilustración 48. - Formulario de inserción de una reparación para vehículo existente

Ilustración 49. – Formulario de Inserción de reparación para vehículo nuevo

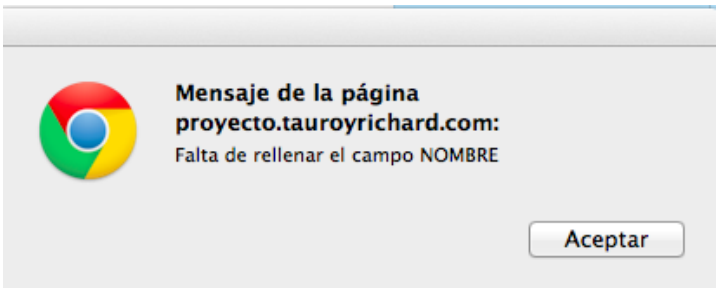


Ilustración 50. - Mensaje de error de campo nombre

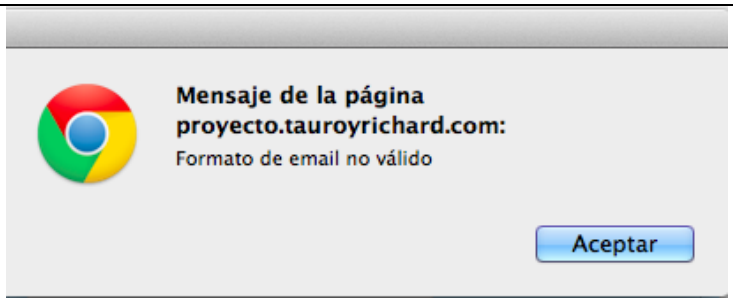


Ilustración 51. - Mensaje de error de campo email

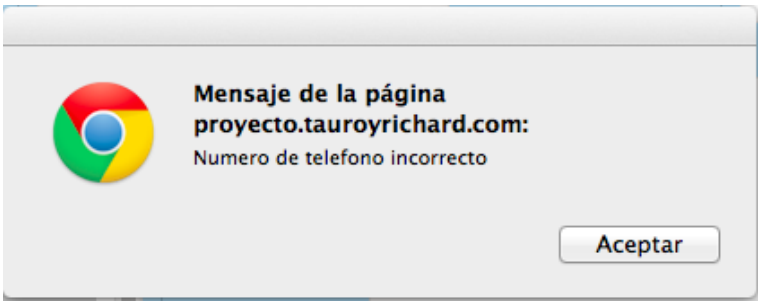


Ilustración 52. - Mensaje de error de campo teléfono

10.1.3. Caso de uso **Modificar reparación de vehículo**

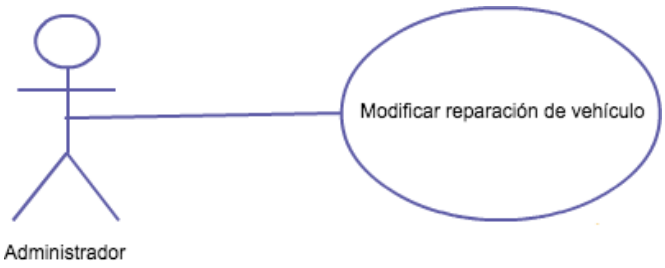


Ilustración 53. - Caso de uso Modificar reparación de vehículo

Nombre: Caso de uso Modificar reparación de vehículo
Descripción: El administrador podrá modificar, si así lo desea, cualquier dato relacionado con la reparación de un vehículo o con los datos del propio vehículo que previamente haya sido insertado en el sistema.
Actores: Administrador.
Precondiciones: Estar identificado como administrador.
Requisitos no funcionales: Ninguno.
Flujo de eventos: 1) El administrador, que previamente ha introducido una contraseña en la página "Entorno de administración" (Ilustración 54), pulsará el botón "Modificar reparación del vehículo" (Ilustración 55) que le enviará a otra interfaz (Ilustración 58) que será la interfaz en la que aparecerán todos los campos del formulario para poder cambiar cualquier dato relacionado con una reparación de un vehículo. 2) En dicha interfaz aparecerá, en primer lugar, un desplegable en el que pondrá "Escoja la matrícula" (Ilustración 56). En dicho desplegable aparecerán todas las matrículas de los coches que ya han sido insertadas en otras ocasiones porque ya se ha hecho alguna reparación del mismo anteriormente. 3) Una vez elegida la matrícula del vehículo del cual se quiere modificar los datos, aparecerá otro desplegable que pondrá "Escoja la fecha de la reparación" (Ilustración 57). El administrador deberá elegir la fecha en la que se realizó la reparación para poder modificar los datos de dicha reparación. 4) Una vez elegida la matrícula del vehículo y la fecha en la que se realizó la reparación

aparecerá un formulario con todos los campos ya rellenos con los datos del vehículo y de la reparación en cuestión (Ilustración 58). En este formulario, el administrador podrá cambiar el valor de cualquier campo.

4.1.) Aparecerán dos campos nuevos que servirán al administrador para indicar el estado de cada reparación.

4.1.1) Se encuentra el campo "Estado de la reparación" cuyos valores serán "Sin finalizar" o "Finalizada". Permitirá que los clientes cuando lo deseen y la reparación esté "Finalizada" puedan proceder al pago de la factura a través de la web.

4.1.2) También hay un campo llamado "Factura" cuyos valores son "Sin pagar" o "Pagada" y servirá para indicar si los clientes han pagado o no la factura de la reparación del vehículo.

4.1.3) Por tanto, el administrador deberá cambiar el valor de "Estado de reparación" a "Finalizada", cuando la reparación haya concluido en el taller, para que los clientes puedan pagar la factura a través de la web. Cuando se cambie "Estado de reparación" a "Finalizada", y el valor "Factura" sea "Sin pagar", el sistema creará un aviso de "Finalización de la reparación" que avisará al cliente correspondiente a través de la App Tauro & Richard que el vehículo está listo para recoger y que podrá abonar la factura a través de nuestro sistema o directamente en el taller.

4.1.4) También, deberá cambiar el valor de "Factura" a "Pagada" en caso de que la factura haya sido abonada en el taller y no a través de la web.

4.2.) En el campo "Concepto mano de obra" aparecerán seleccionadas las opciones que el administrador seleccionó cuando introdujo la reparación por primera vez o en la última modificación realizada. Se podrán deseleccionar o seleccionar todas las opciones que se deseen.

5) Una vez modificados todos los campos deseados, se deberá seleccionar el botón "Aceptar".

5.1.) En caso de que no se rellene alguno de los campos obligatorios aparecerá el siguiente cuadro de error (Ilustración 59).

5.2.) En caso de que el campo email haya sido rellenado pero no se introduzca con

el formato correcto, saldrá un error de formato de email incorrecto (Ilustración 60).

5.3.) En caso de que el teléfono no se introduzca con un formato adecuado, saldrá un mensaje de error de número de teléfono incorrecto (Ilustración 61).

6) Si el administrador quisiera borrar todos los datos existentes y rellenar el formulario entero, deberá pulsar el botón “Borrar”.

Pos condiciones: El administrador modificará todos los datos relacionados con la reparación de un vehículo, así como indicará si la reparación ha sido finalizada o si la factura ha sido pagada.

Interfaz gráfica:

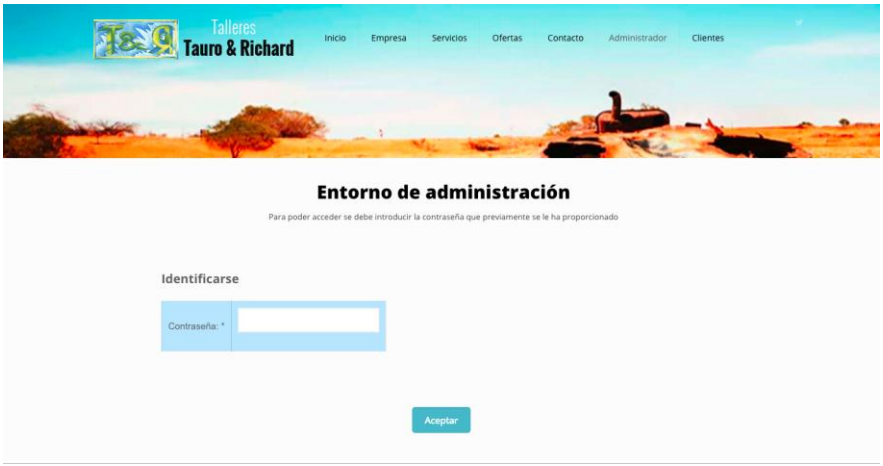


Ilustración 54. - Entorno de administración

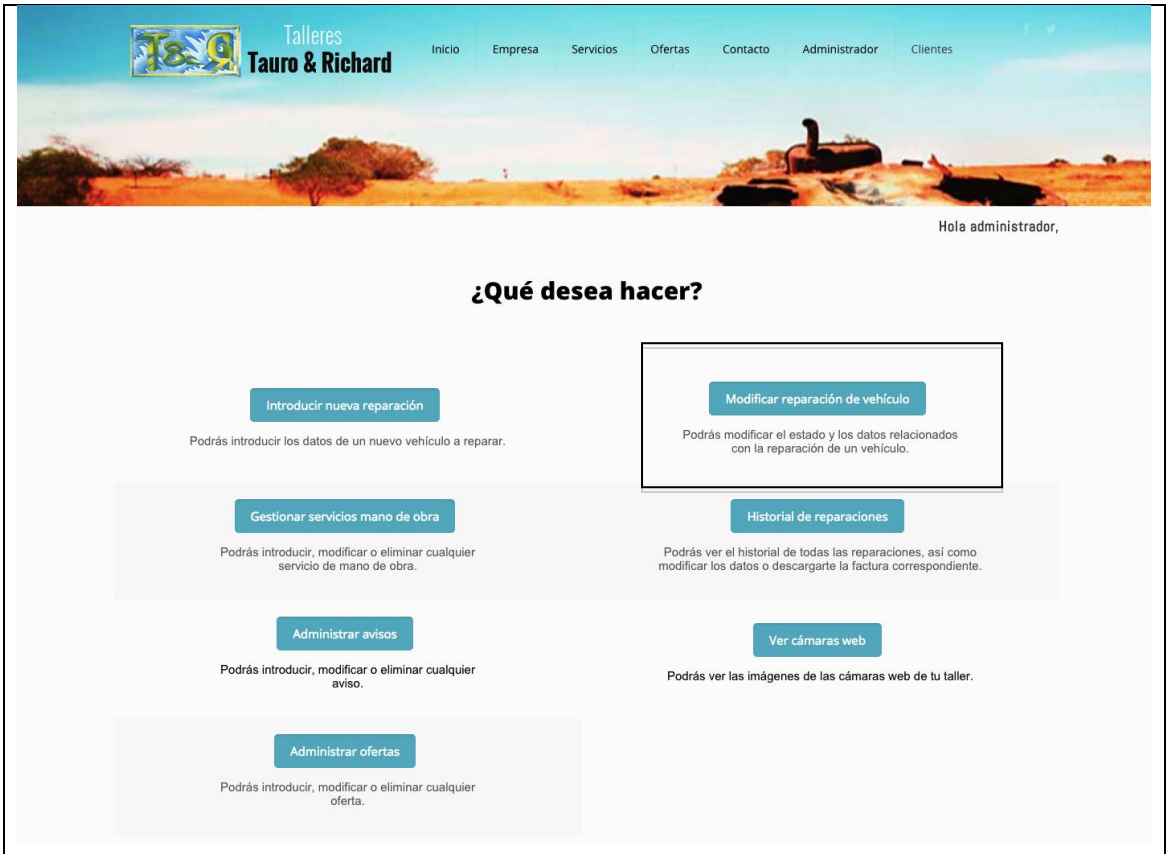


Ilustración 55. - Menú de administración

Ilustración 56. - Selección de matrícula al modificar datos

Ilustración 57. - Elegir fecha de la reparación del vehículo

Ilustración 58. - Formulario de modificación de datos de la reparación

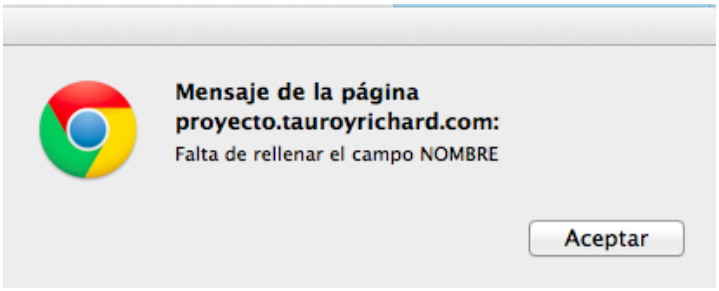


Ilustración 59. - Mensaje de error de campo nombre

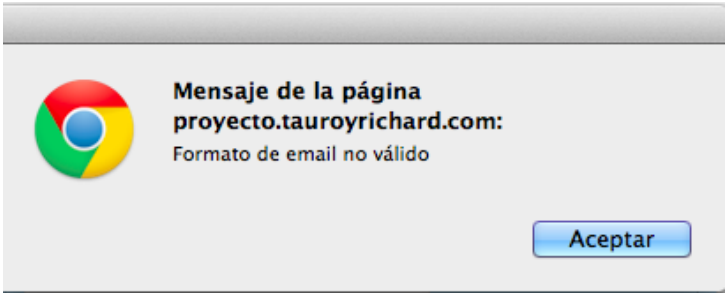


Ilustración 60. - Mensaje de error de campo email

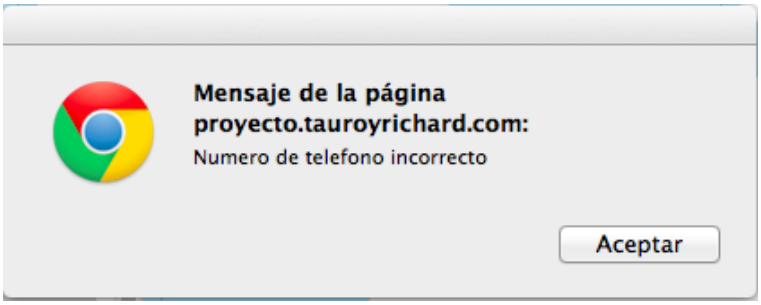


Ilustración 61. - Mensaje de error de campo teléfono

10.1.4. Caso de uso **Gestionar servicios mano de obra**

Ilustración 62. - Caso de uso Gestionar servicios mano de obra

Nombre: Caso de uso Gestionar servicios mano de obra

Descripción: El administrador podrá gestionar los servicios de mano de obra. Podrá añadir, modificar o eliminar el que desee.

Actores: Administrador.

Precondiciones: Estar identificado como administrador.

Requisitos no funcionales: Ninguno.

Flujo de eventos:

1) El administrador, que previamente ha introducido una contraseña en la página "Entorno de administración" (Ilustración 63), pulsará el botón "Gestionar servicios mano de obra" (Ilustración 64) que le enviará a otra interfaz (Ilustración 65) que será la interfaz en la que aparecerán todos los servicios de mano de obra, que podrá modificarlos, eliminarlos o insertar nuevos. Los botones "Modificar" o "Eliminar" estarán deshabilitados hasta que se seleccione un servicio.

2) Si pulsa "Insertar nuevo" aparecerá debajo un formulario (Ilustración 66) en el que aparecerán los campos nombre, descripción y precio. Los campos nombre y precio son obligatorios.

    2.1.) En caso de no rellenar alguno de los campos obligatorios aparecerá un mensaje de error (Ilustración 67 y Ilustración 68).

    2.2.) Una vez rellenados todos los datos, se deberá pulsar el botón "Insertar".

3) Si pulsa "Modificar", seleccionando previamente algún servicio, aparecerá un pequeño formulario de modificación (Ilustración 69) con los campos de nombre, descripción y

precio ya rellenados con los datos correspondientes. El administrador podrá modificar todo lo que desee.

3.1.) En caso de no rellenar alguno de los campos obligatorios aparecerá un mensaje de error (Ilustración 67 y Ilustración 68).

3.2.) Una vez rellenados todos los campos obligatorios, se deberá pulsar el botón "Modificar".

4) Si desea eliminar uno o más de un servicio, se deberán seleccionar todos los servicios que se deseen eliminar y pulsar "Eliminar" (Ilustración 70). De esta manera quedarán eliminados todos los servicios no deseados.

Pos condiciones: El administrador insertará, modificará o eliminará todos los servicios de mano de obra que desee y como consecuencia de ello se actualizará también en la base de datos.

Interfaz gráfica:

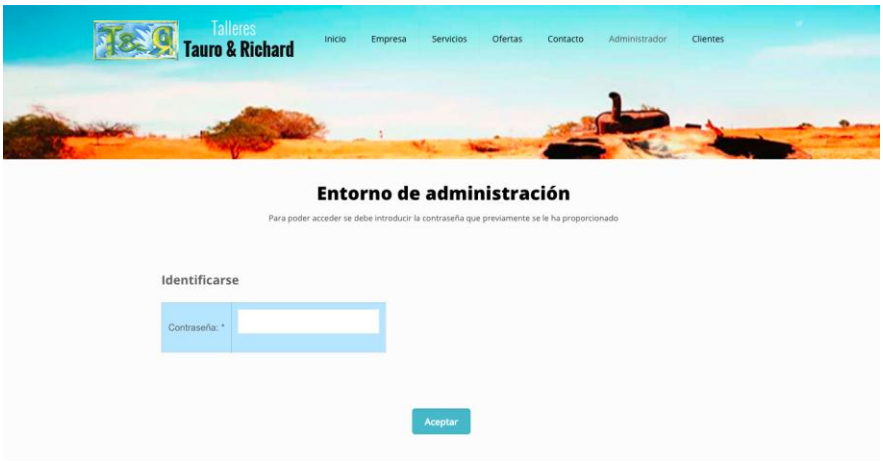


Ilustración 63. - Entorno de administración

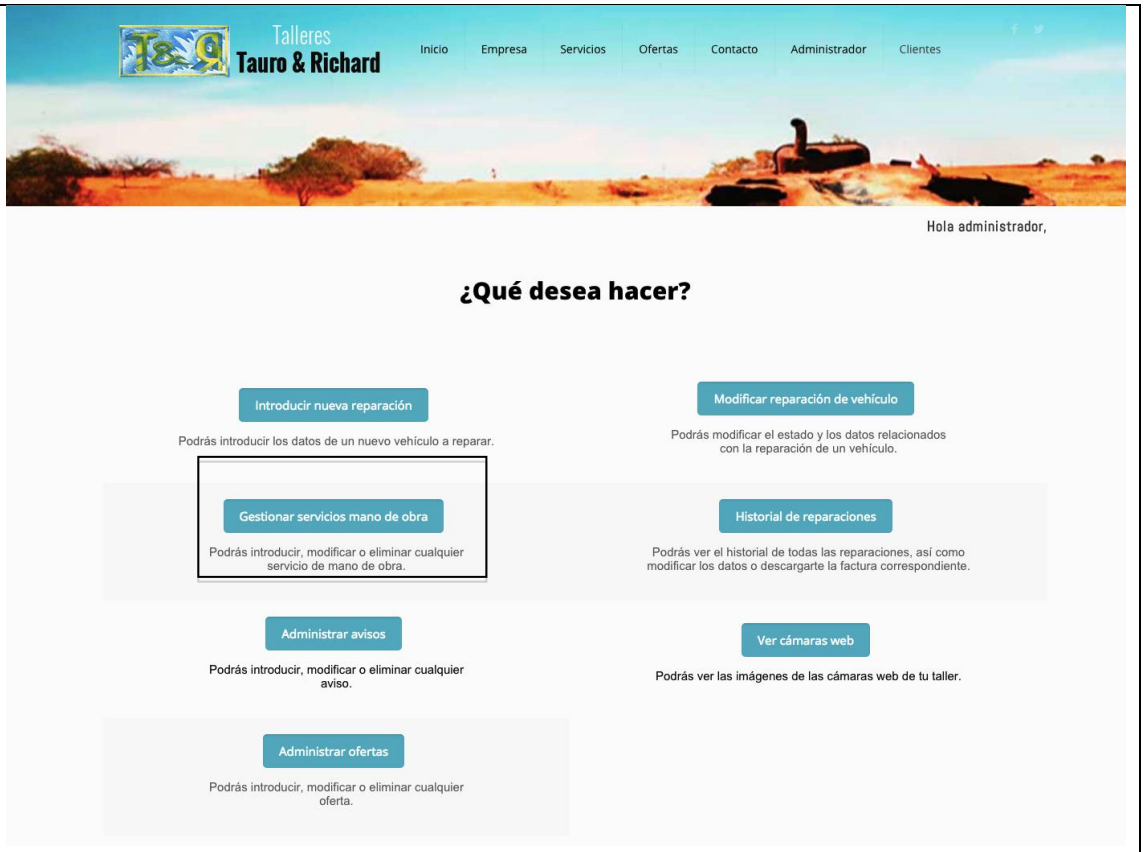


Ilustración 64. - Menú del administrador

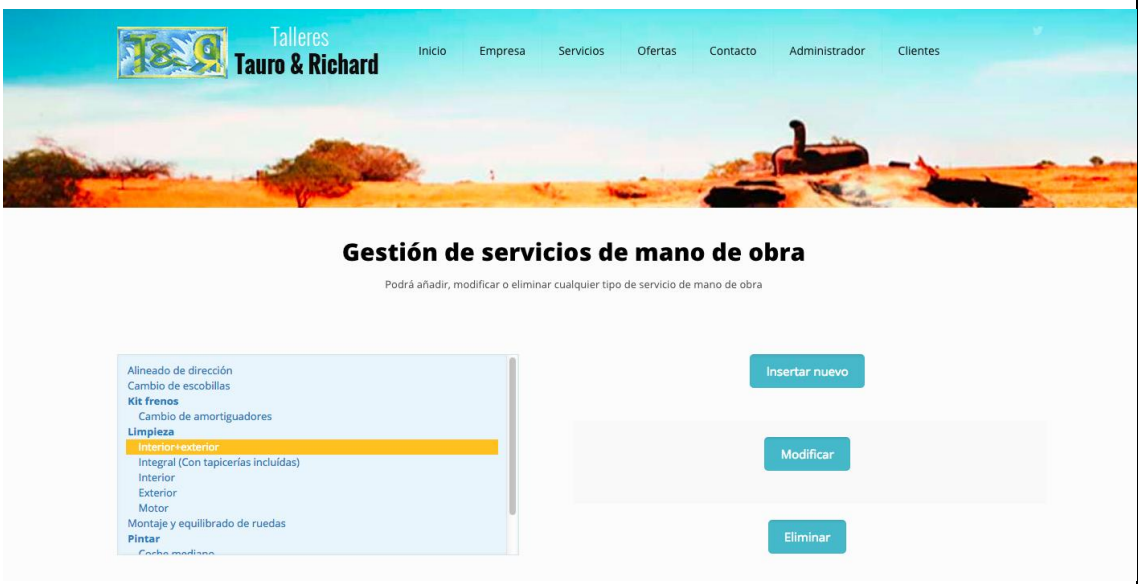


Ilustración 65. - Gestión de servicios de mano de obra

Ilustración 66. - Inserción de nuevo servicio

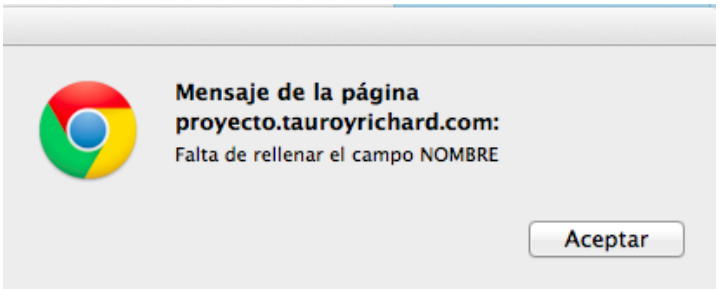


Ilustración 67. - Mensaje de error de campo nombre

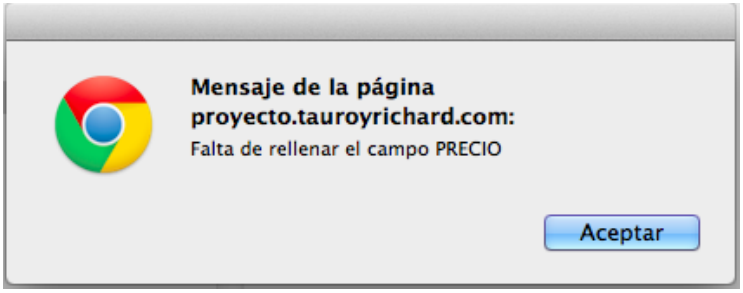


Ilustración 68. - Mensaje de error de campo precio

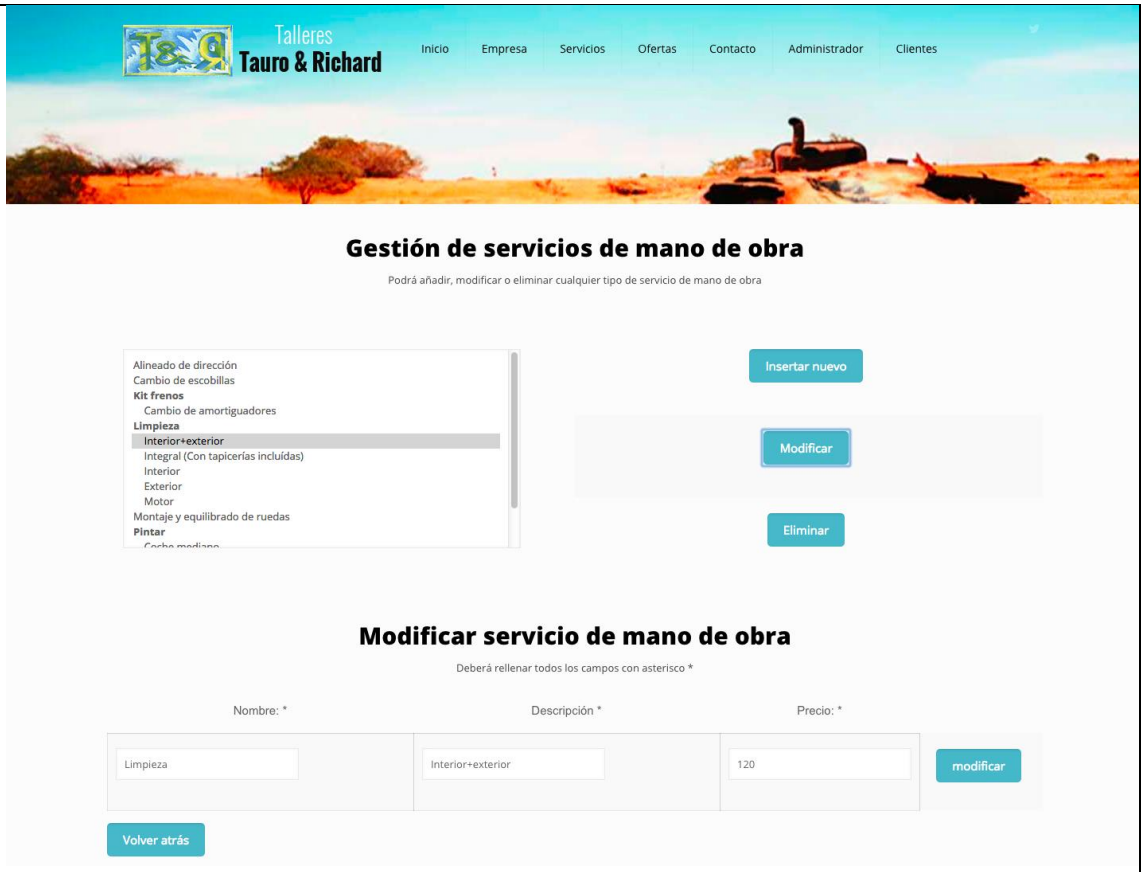


Ilustración 69. - Modificación de los datos de un servicio

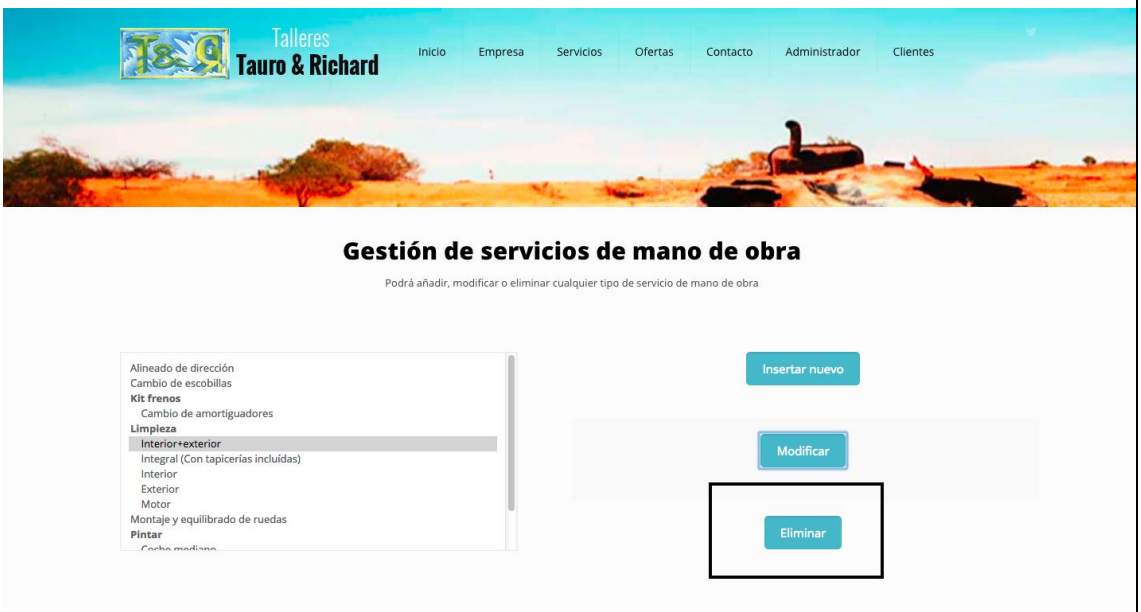


Ilustración 70. - Eliminar un servicio

10.1.5. Caso de uso **Historial de reparaciones**

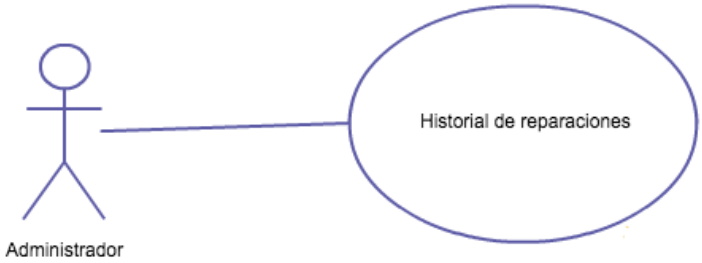


Ilustración 71. - Caso de uso Historial de reparaciones

Nombre: Caso de uso Historial de reparaciones

Descripción: El administrador podrá ver todo el historial de reparaciones realizadas en el taller. Podrá ver la factura, modificar los datos y eliminar la reparación o todo el historial de un vehículo.

Actores: Administrador.

Precondiciones: Estar identificado como administrador.

Requisitos no funcionales: Ninguno.

Flujo de eventos:

1) El administrador, que previamente ha introducido una contraseña en la página "Entorno de administración" (Ilustración 72), pulsará el botón "Historial de reparaciones" (Ilustración 73) que le enviará a otra interfaz (Ilustración 74) que será la interfaz en la que aparecerán todo el historial de reparaciones realizadas en el taller Tauro & Richard. Este historial mostrará de cada reparación la fecha, matrícula, modelo del vehículo, nombre y apellidos del dueño del vehículo, importe total de la reparación, estado de la reparación y la factura (si ha sido pagada o no).

2) Este historial se podrá ordenar por "Fecha", "Nombre" o "Matrícula" (Ilustración 75) para que el administrador pueda encontrar lo más rápido posible la reparación deseada.

3) El administrador podrá ver la factura de cada reparación pulsando el botón "Ver factura" (Ilustración 76), habiendo seleccionado alguna reparación previamente. Se abrirá una nueva pestaña mostrando la factura en formato pdf (Ilustración 77), en la que se podrá visualizar correctamente y descargarla si así lo desea. En caso de que la factura esté pagada, aparecerá la factura con un sello de pagado (Ilustración 78).

4) Pulsando el botón “Modificar” (Ilustración 79), aparecerá una nueva interfaz (Ilustración 80) en la que aparecerán todos los campos rellenos con los datos de la reparación del vehículo. En ella el administrador podrá modificar cualquier dato del vehículo o de la reparación, así como el estado de la reparación (Si está finalizada o sin finalizar) o si la factura ha sido pagada.

    4.1.) Una vez modificados todos los campos que se deseen, se deberá seleccionar el botón “Aceptar”.

    4.2.) En caso de que no se rellenen alguno de los dos campos obligatorios aparecerá el siguiente cuadro de error (Ilustración 81).

    4.3.) En caso de que el campo email haya sido rellenado pero no se introduzca con el formato correcto, saldrá un error de formato de email incorrecto (Ilustración 82).

    4.4.) En caso de que el teléfono no se introduzca con un formato adecuado, saldrá un mensaje de error de numero de teléfono incorrecto (Ilustración 83).

    4.5.) Si el administrador quisiera borrar todos los datos existentes y rellenar el formulario entero, deberá pulsar el botón “Borrar”.

5) Si el administrador desea eliminar una reparación concreta deberá pulsar el botón “Eliminar reparación” (Ilustración 84). De esta manera se eliminará de la base de datos la reparación deseada.

6) Si el administrador desea eliminar todo el historial de un vehículo, es decir, todos los datos correspondientes al vehículo y todos los datos de todas las reparaciones realizadas, deberá pulsar el botón “Eliminar todo el historial del vehículo” (Ilustración 85).

Pos condiciones: El administrador podrá modificar o eliminar los datos de las reparaciones introducidas en el sistema o ver la factura de cualquiera de ellas.

Interfaz gráfica:

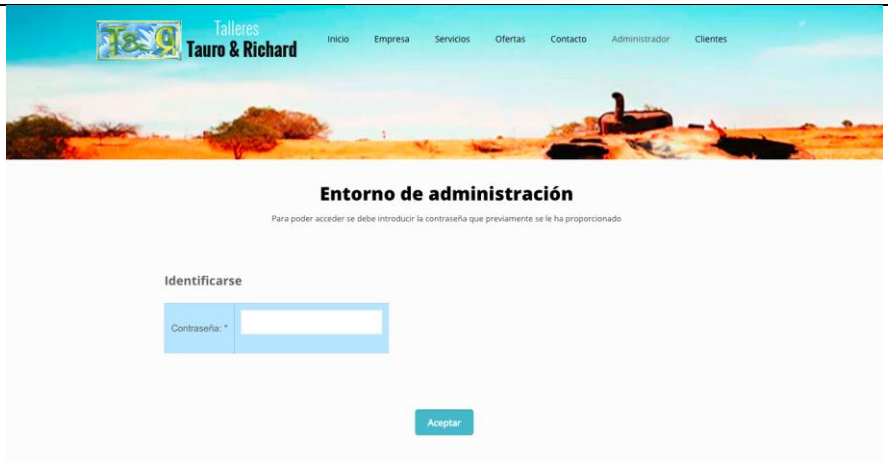


Ilustración 72. - Entorno de administración

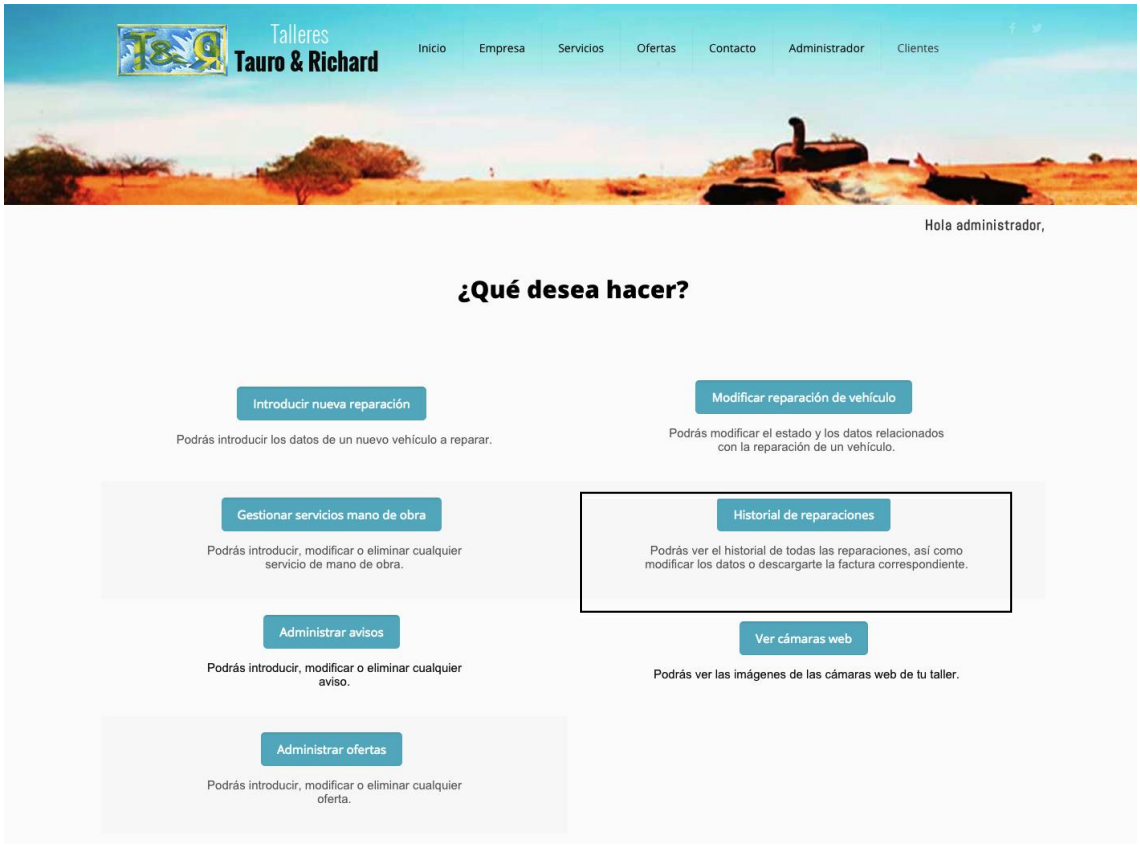


Ilustración 73. - Menú del administrador

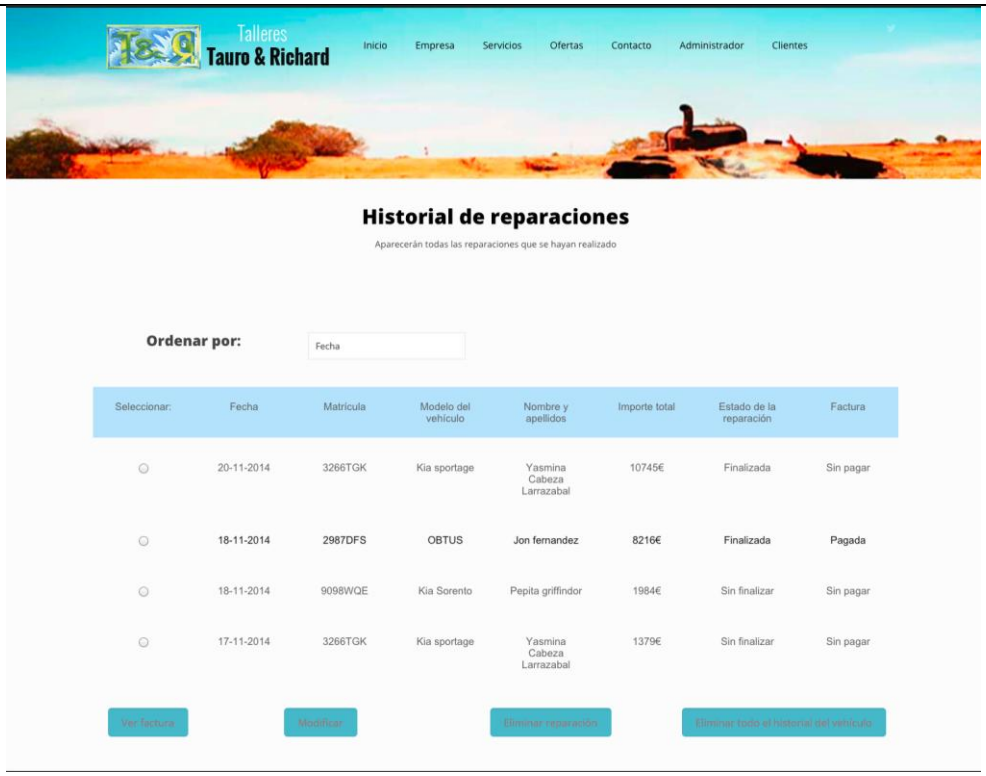


Ilustración 74. - Historial de reparaciones

Ilustración 75. - Ordenar historial

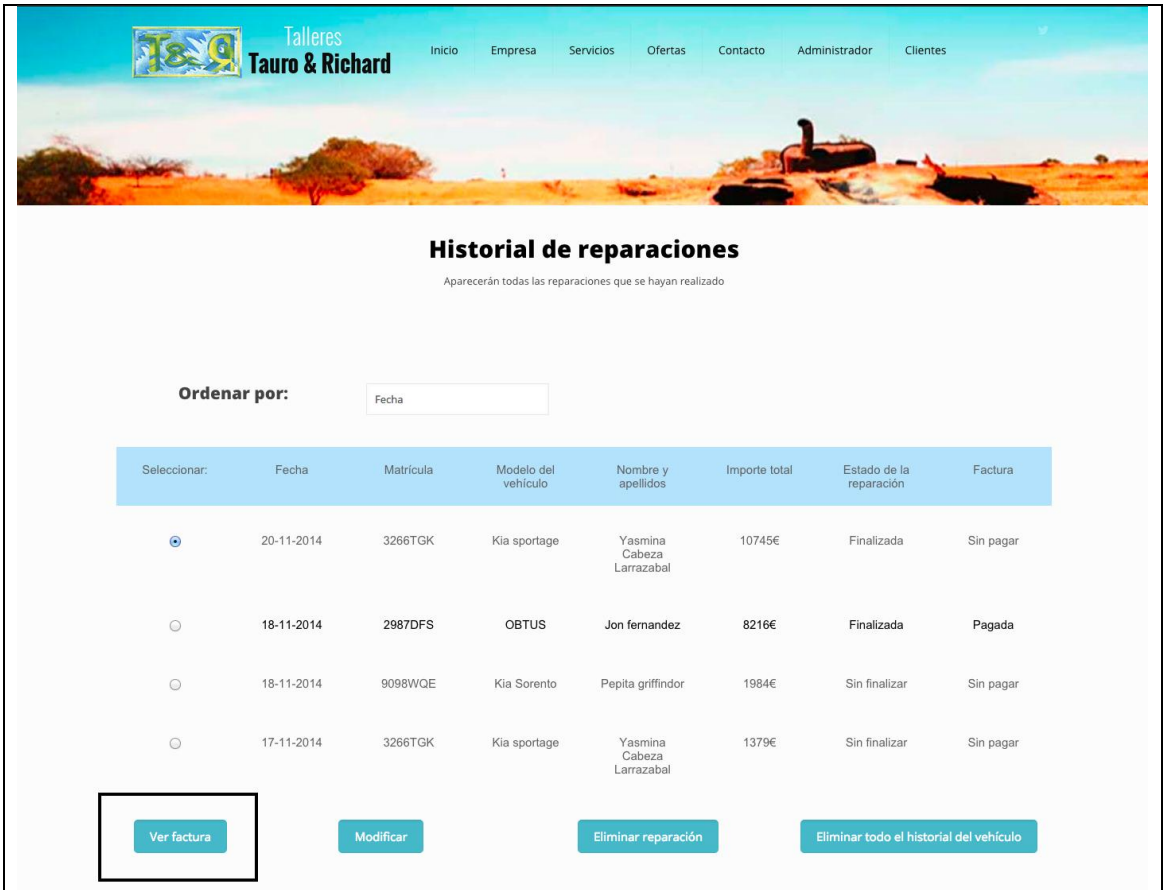


Ilustración 76. - Ver factura de la reparación

Talleres Tauro & Richard

FACTURA

CIF: E - 48620198
Polígono Aperribai Auzoa S/N, PABELLÓN 6, Galdakao 48960, Vizcaya - España
94 449 25 77 / 667557962 / 610424613
Email: talleres@tauroyrichard.com

Número de factura: 39
Fecha: 18-11-2014

DATOS DEL CLIENTE

Nombre y apellidos: Pepita griffindor
Dirección: Benavente 2, bajo , 43990 , CUENCA - España
DNI: 45879906E
Teléfono: 943336789
Email: yasmina@notebuk.com

DATOS DEL VEHÍCULO

Matrícula: 9098WQE
Modelo: Kia Sorento
CIA: Linea directa

Concepto	Precio Unitario	Importe
4 x Ruedas	100	400
2 x Escobillas	20	40
Arreglo de puerta trasera	1200	1200

Concepto mano de obra	Importe

Base imponible	IVA	Cuota de IVA	Importe total
1640€	21%	344€	1984€

FIRMA DEL CLIENTE

FIRMA Y SELLO DE LA EMPRESA

Ilustración 77. - Factura de la reparación sin pagar

Talleres Tauro & Richard

FACTURA

CIF: E - 48620198
Polígono Aperribai Auzoa S/N, PABELLÓN 6, Galdakao 48960, Vizcaya - España
94 449 25 77 / 667557962 / 610424613
Email: talleres@tauroyrichard.com

Número de factura: 38
Fecha: 18-11-2014

DATOS DEL CLIENTE

Nombre y apellidos: Jon fernandez
Dirección: San miguel de salinas, 21 2ª B , 8900 , A CORUÑA - España
DNI: 222222L
Teléfono: 678675282
Email: jon@gmail.com

DATOS DEL VEHÍCULO

Matrícula: 2987DFS
Modelo: OBTUS
CIA: Axa seguros

PAGADO

Concepto	Precio Unitario	Importe
Porton trasero	349	349
Luz delantera	120	120
2x Escobillas	20	40
2x Ruedas traseras	80	160

Concepto mano de obra	Importe
Limpieza : Integral (Con tapicerías incluídas)	80€
Limpieza : Interior+exterior	120€

Base imponible	IVA	Cuota de IVA	Importe total
6790€	21%	1426€	8216€

FIRMA DEL CLIENTE

FIRMA Y SELLO DE LA EMPRESA

Ilustración 78. - Factura de la reparación pagada

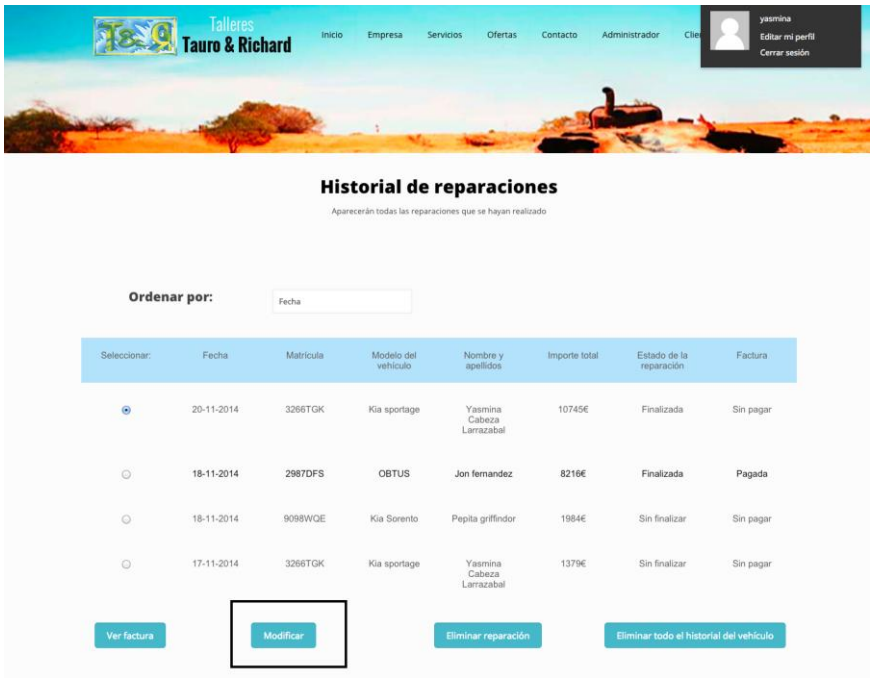


Ilustración 79. - Modificar datos de la reparación

Ilustración 80. - Formulario de modificar datos de la reparación

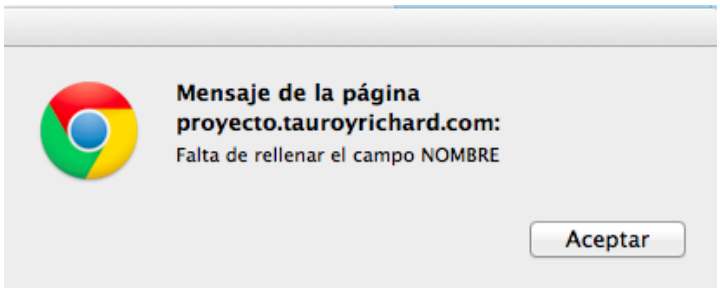


Ilustración 81. - Mensaje de error de campo nombre

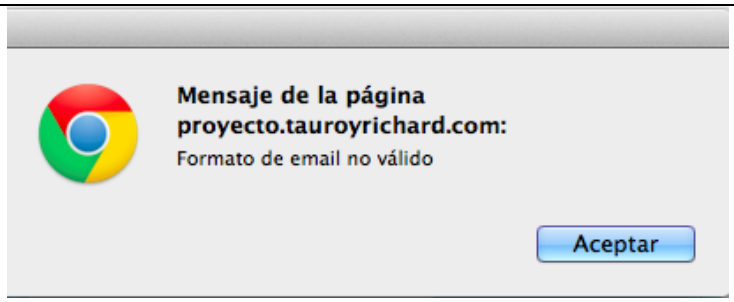


Ilustración 82. - Mensaje de error de campo email

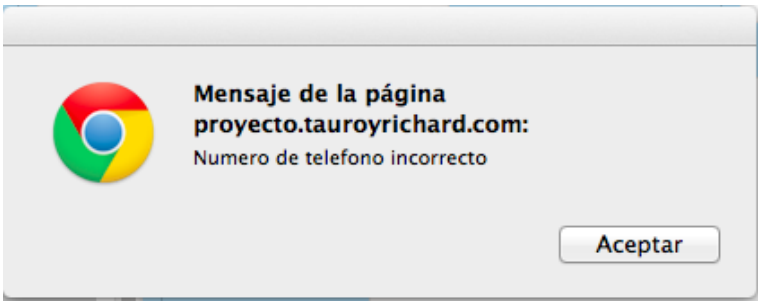


Ilustración 83. - Mensaje de error de campo teléfono

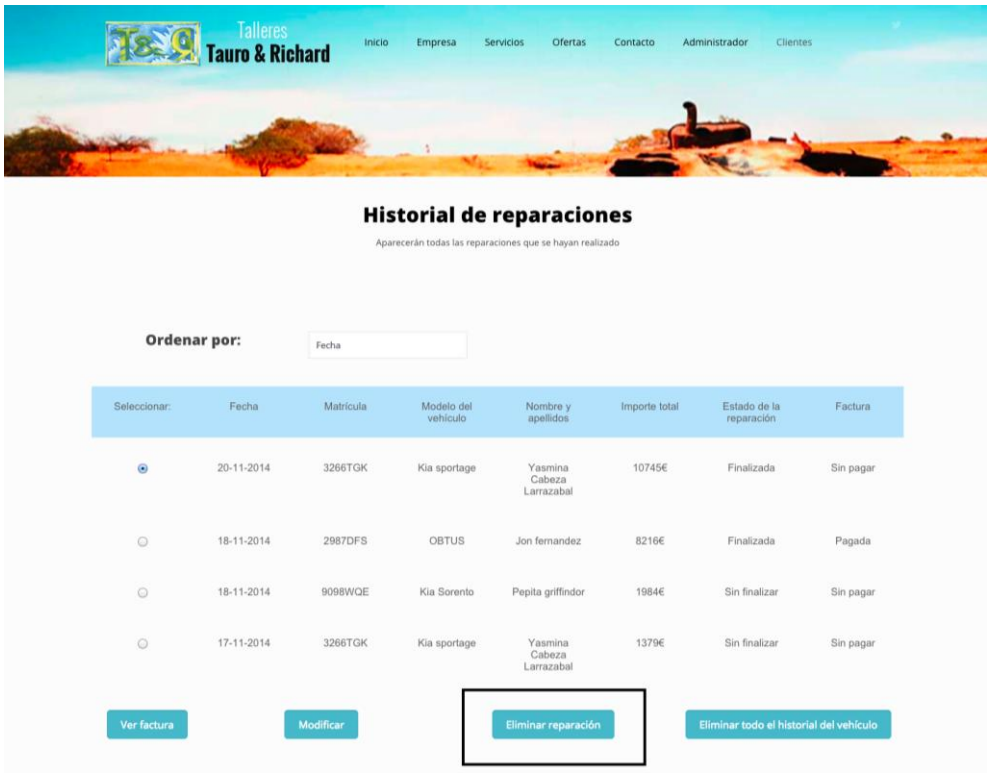


Ilustración 84. - Eliminar datos de la reparación

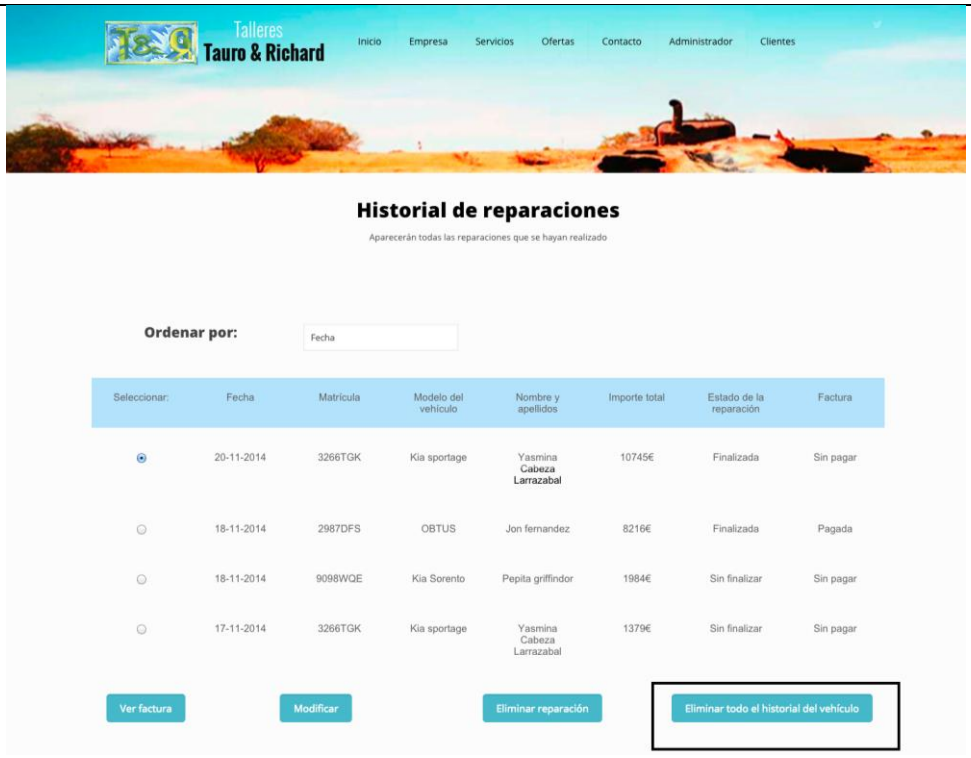


Ilustración 85. - Eliminar historial de un vehículo

10.1.6. Caso de uso **Ver cámaras web**

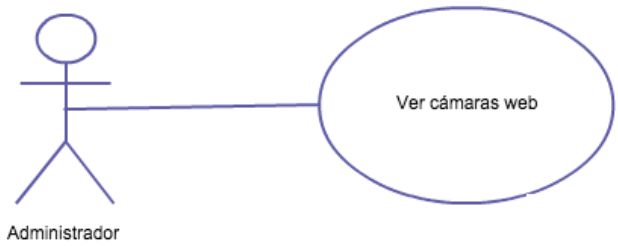


Ilustración 86. - Caso de uso Ver cámaras web

Nombre: Caso de uso Ver cámaras web
Descripción: El administrador podrá visualizar las cámaras web del taller desde la página web.
Actores: Administrador.
Precondiciones: Estar identificado como administrador.
Requisitos no funcionales: Ninguno.
Flujo de eventos: 1) El administrador, que previamente ha introducido una contraseña en la página "Entorno de administración" (Ilustración 87), pulsará el botón "Ver cámaras web" (Ilustración 88) que le enviará a otra interfaz (Ilustración 89), en la que introduciendo usuario y contraseña, se le mostrarán las cámaras web del taller.
Pos condiciones: El administrador podrá ver las cámaras web del taller a través de la aplicación.
Interfaz gráfica:

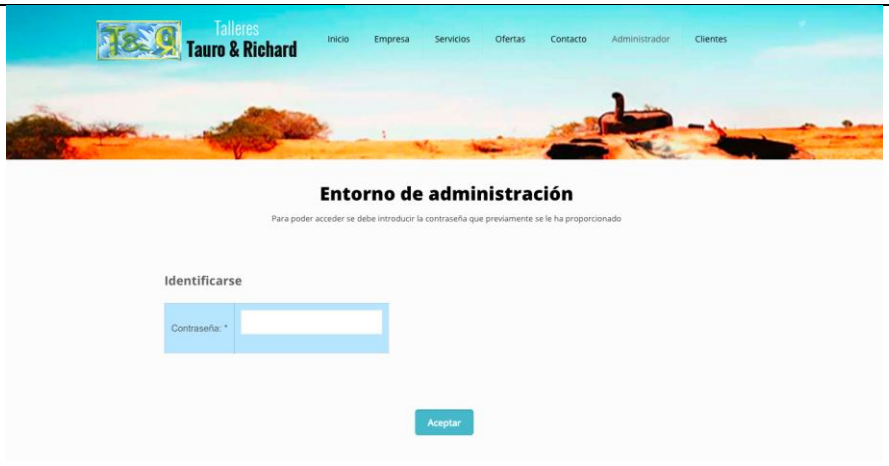


Ilustración 87. - Entorno de administración

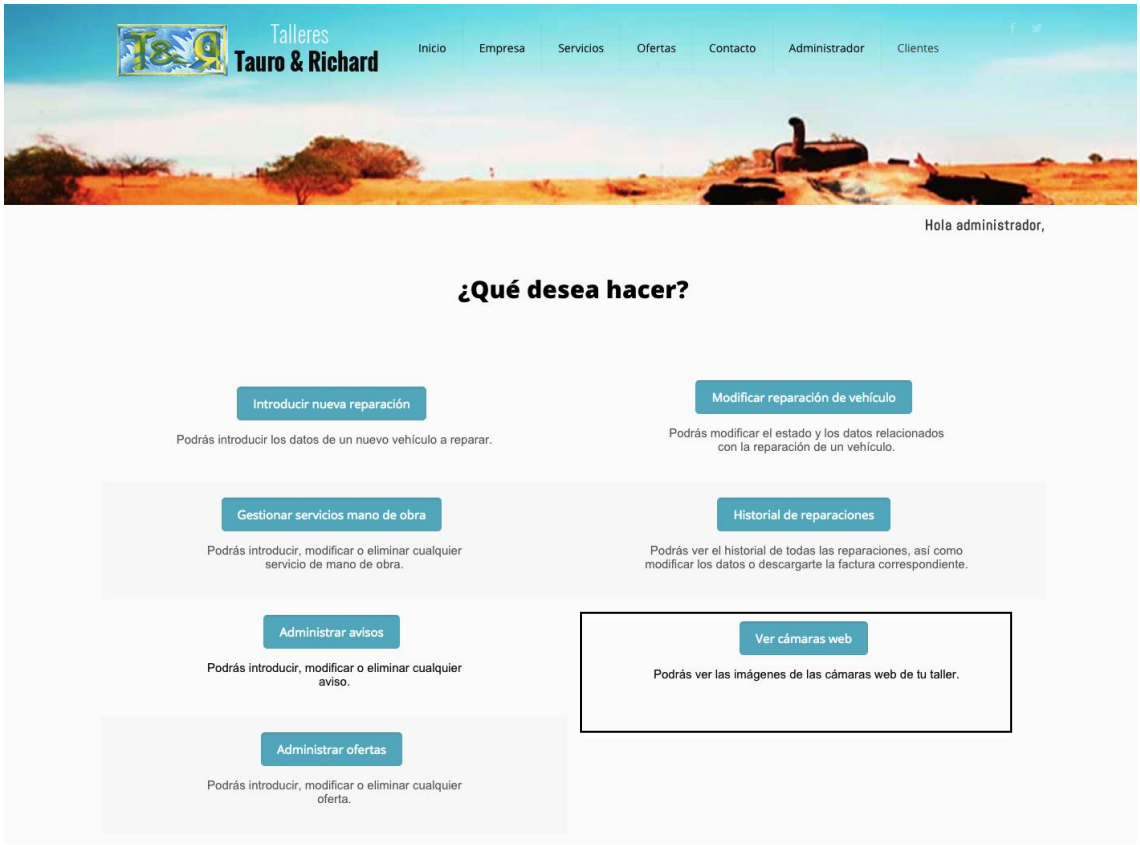


Ilustración 88. - Menú del administrador

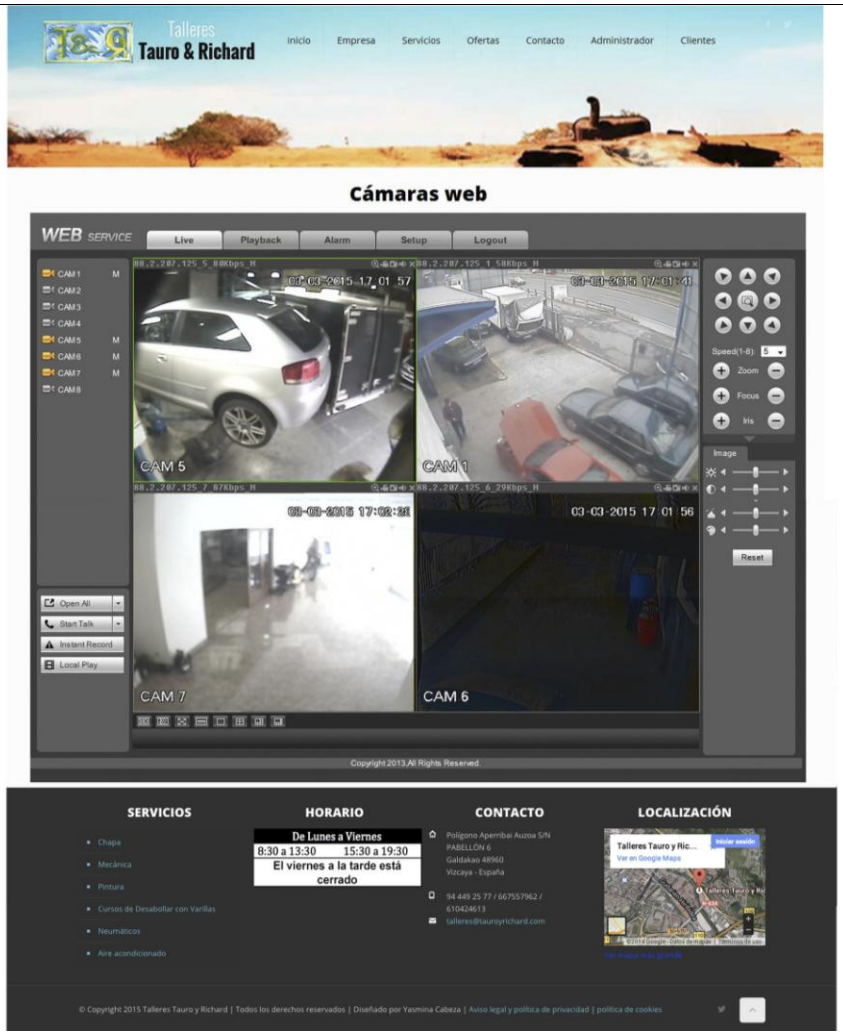


Ilustración 89. - Cámaras web

10.1.7. Caso de uso **Administrar avisos**

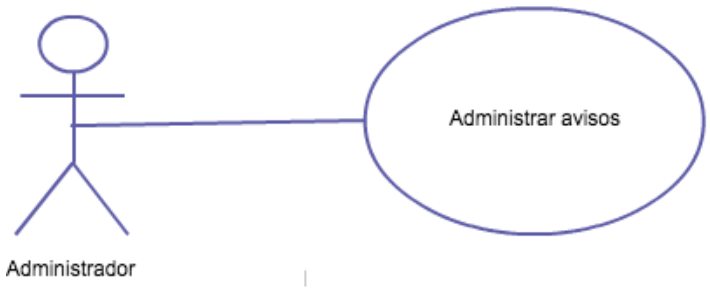


Ilustración 90. - Caso de uso Adminsitrar avisos

Nombre: Caso de uso Administrar avisos

Descripción: El administrador podrá insertar o eliminar un aviso para el vehículo que desee. El aviso que se programe servirá para avisar a los clientes a través de la App Tauro & Richard.

Actores: Administrador.

Precondiciones: Estar identificado como administrador.

Requisitos no funcionales: Ninguno.

Flujo de eventos:

1) El administrador, que previamente ha introducido una contraseña en la página "Entorno de administración" (Ilustración 91), pulsará el botón "Administrar avisos" (Ilustración 92) que le enviará a otra interfaz (Ilustración 93) que será la interfaz en la que aparecerá un menú para escoger la acción que desee realizar en relación con la administración de avisos, pudiendo ser insertar o eliminar aviso.

2) Si selecciona el botón "Introducir nuevo aviso" aparecerá una nueva interfaz (Ilustración 94) con un campo desplegable "Escoja matrícula" en el que se deberá escoger la matrícula del vehículo al que se desea añadirle el aviso. A continuación, aparecerá otro desplegable "Escoja el tipo de aviso" (Ilustración 95) para escoger el tipo de aviso que se desea insertar para el vehículo seleccionado. Una vez escogido el aviso, podrán aparecer tres tipos diferentes de formularios:

    2.1.) Si se escoge "Finalización de la reparación" aparecerá un nuevo desplegable "Escoja la fecha" (Ilustración 96) para escoger la fecha de la reparación que ya se ha finalizado y para la cual se quiere crear un aviso de finalización de reparación, que aparecerá en la App Tauro & Richard para que el cliente pase a recoger su vehículo.

Por último, aparece un desplegable en el que se debe escoger si la reparación está Finalizada o Sin finalizar (Ilustración 97) y se debe pulsar el botón "Aceptar" que le devolverá a la página de administración de avisos (Ilustración 93). Ya estará, por tanto, creado el aviso de pase a recoger su vehículo y aparecerá en la App del cliente correspondiente al vehículo con matrícula escogida previamente.

2.2.) Si se escoge "Carga de aire acondicionado" o "Pasar la ITV" aparecerá un nuevo formulario con dos campos a rellenar (Ilustración 98). La casilla en la que pone "Última fecha" se deberá introducir la última fecha de carga de aire acondicionado o de paso de la ITV, en el formato indicado. Una vez introducida dicha fecha, en el caso de carga de aire acondicionado, en el campo "Siguiente fecha" aparecerá automáticamente la fecha siguiente de revisión, que será al de 5 años y en el caso de Pasar la ITV aparecerá también automáticamente la fecha siguiente de revisión, que será al de 4 años. Por último, se pulsa el botón "Aceptar" y ya estará creado el aviso correspondiente.

2.2.1) Si alguna de las fechas no tiene el formato indicado o no se ha rellenado el campo, al pulsar "Aceptar" aparecerá un cuadro de error (Ilustración 100).

2.3.) Si se escoge cualquiera de los otras opciones, aparecerá una nueva interfaz (Ilustración 99) con dos campos a rellenar. El primer campo es el de "Kilometraje actual" y en él se deberá introducir los kilómetros que tenía el vehículo cuando se le realizó el último cambio. El siguiente campo "Kilometraje siguiente de revisión" será un campo que se rellene automáticamente en función del aviso escogido.

2.3.1) Si se escoge "Cambio de aceite", una vez rellenado el primer campo con los kilómetros del vehículo, el siguiente campo se autorrellenará y aparecerá el número de kilómetros con el que se deberá realizar el siguiente cambio. En este caso se sumará 15.000 km a los introducidos y esos serán los kilómetros con los que el vehículo deberá realizar el siguiente cambio de aceite.

2.3.2) En el caso de "Cambio de pastillas de frenos" al kilometraje actual se le sumará 50.000 km y el resultado será el que aparezca en la casilla "Kilometraje siguiente de revisión".

2.3.3) En el caso de "Cambio de ruedas" la cantidad a sumar será de 25.000 km y ese valor final serán los kilómetros con los que el vehículo deberá realizar el siguiente cambio de ruedas.

2.3.4) En el caso de “Cambio de embragues” se deberá sumar 150.000 km al kilometraje actual.

Una vez rellenados los dos campos se deberá pulsar el botón Aceptar.

2.3.5) Si alguno de los dos campos no son numéricos, al pulsar el botón Aceptar, aparecerá un mensaje de error de formato incorrecto (Ilustración 101).

2.3.6) Si no se rellena alguno de los dos campos aparecerán los siguientes mensajes de error (Ilustración 102) y (Ilustración 103).

2.4.) En cualquier caso de todos los anteriores, si el administrador desea volver al menú de administración de avisos, sin introducir ningún aviso nuevo, deberá pulsar el botón “Cancelar”.

3) Si selecciona la opción “Eliminar aviso”, aparecerá una nueva interfaz (Ilustración 104) con un desplegable para elegir el aviso que se desea eliminar. Los avisos aparecerán por nombre de aviso, matrícula del vehículo y fecha, en el caso de cambios de aceite, pastillas de frenos, ruedas o embragues, será la fecha del último cambio. En los demás casos, la fecha será la fecha de introducción del aviso al sistema.
Una vez seleccionado el aviso aparecerá el botón de eliminar (Ilustración 105). Se debe pulsar dicho botón y el aviso quedará eliminado y volverá al menú de administración de avisos (Ilustración 93).

Pos condiciones: El administrador podrá insertar o eliminar un aviso.

Interfaz gráfica:

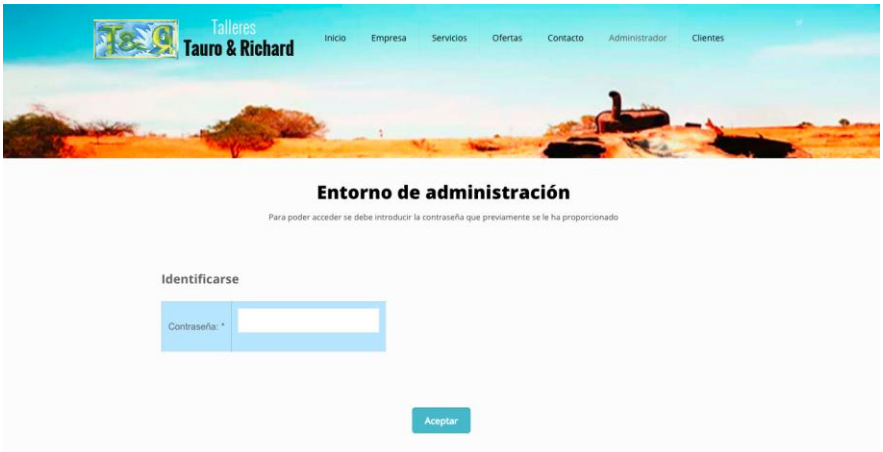


Ilustración 91. - Entorno de administración

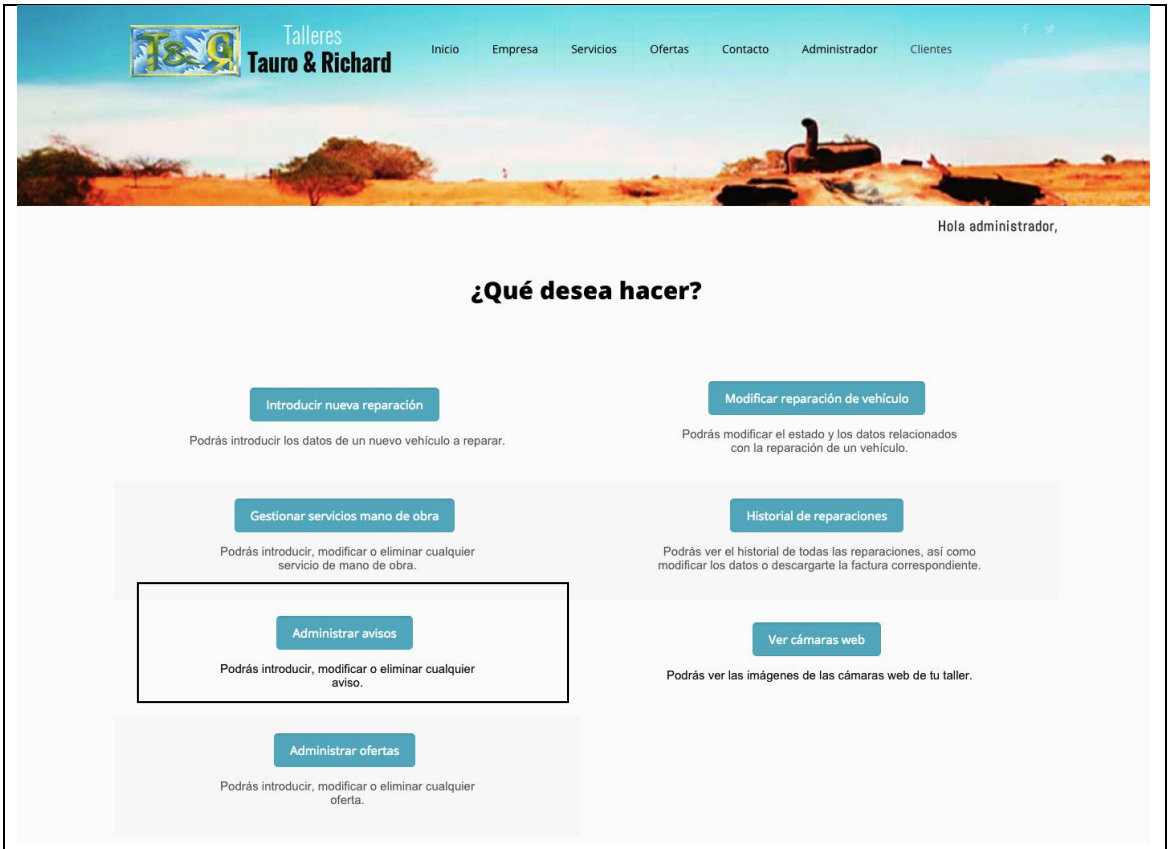


Ilustración 92. - Menú del administrador

Ilustración 93. - Menú de administrar avisos

Ilustración 94. - Introducir nuevo aviso

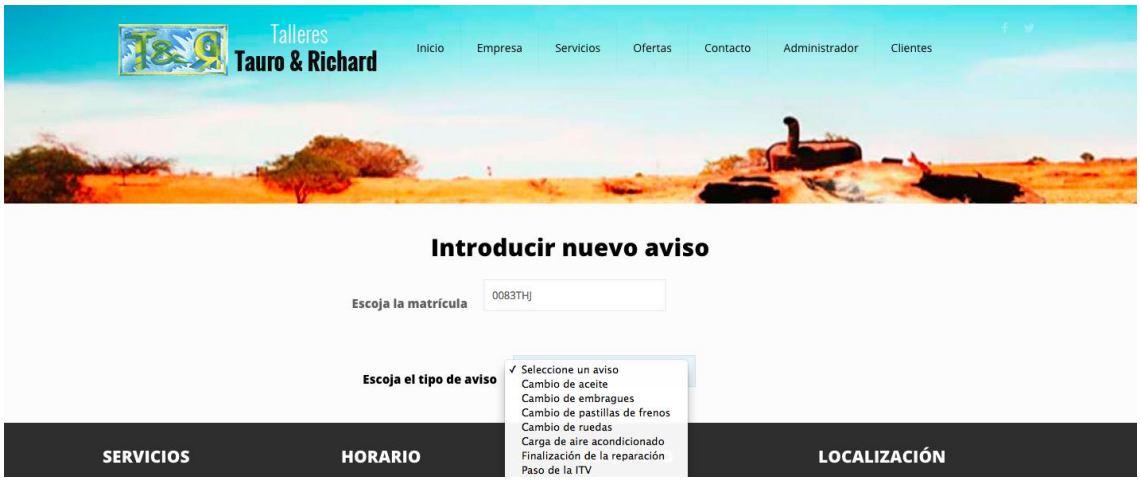


Ilustración 95. - Introducir nuevo aviso - Escoger aviso

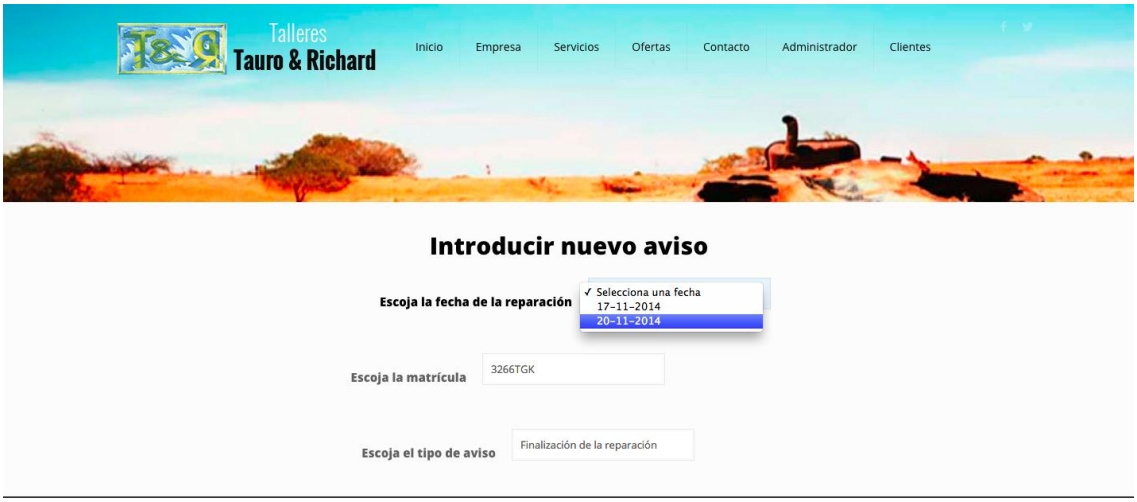


Ilustración 96. - Introducir nuevo aviso - Finalización de la reparación

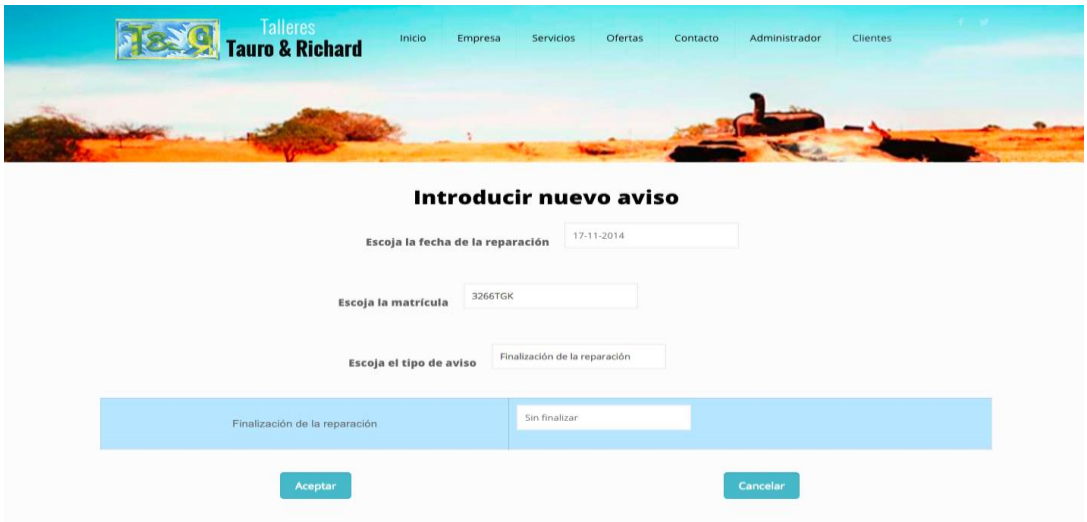


Ilustración 97. - Introducir nuevo aviso - Formulario finalización de la reparación

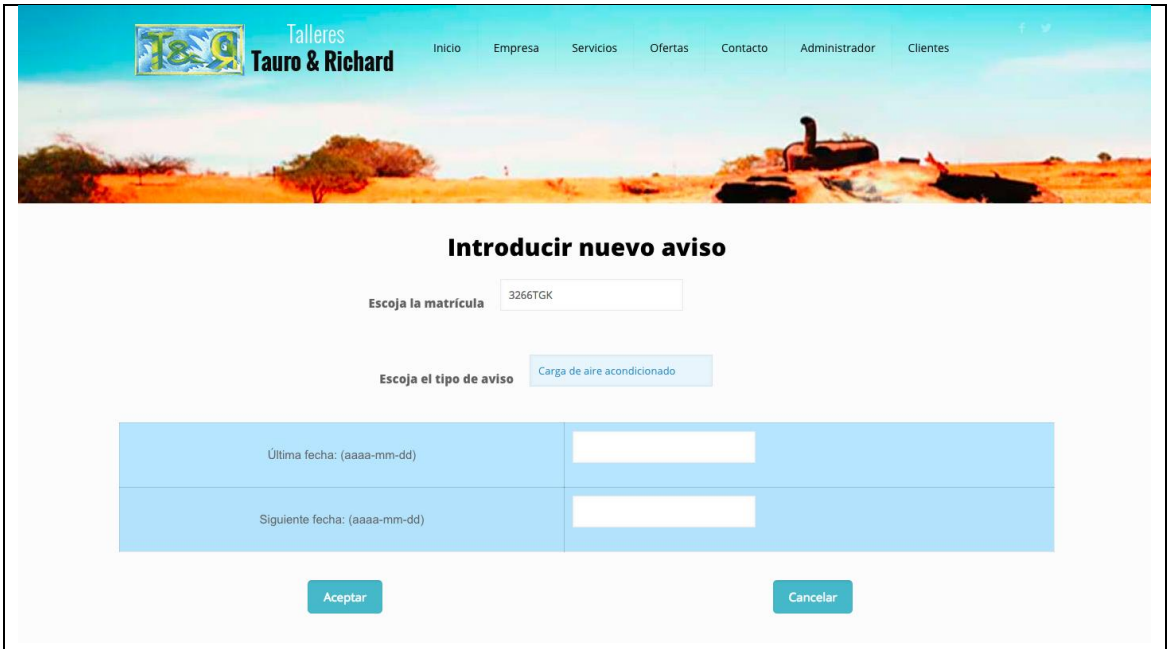


Ilustración 98. - Introducir nuevo aviso - fechas

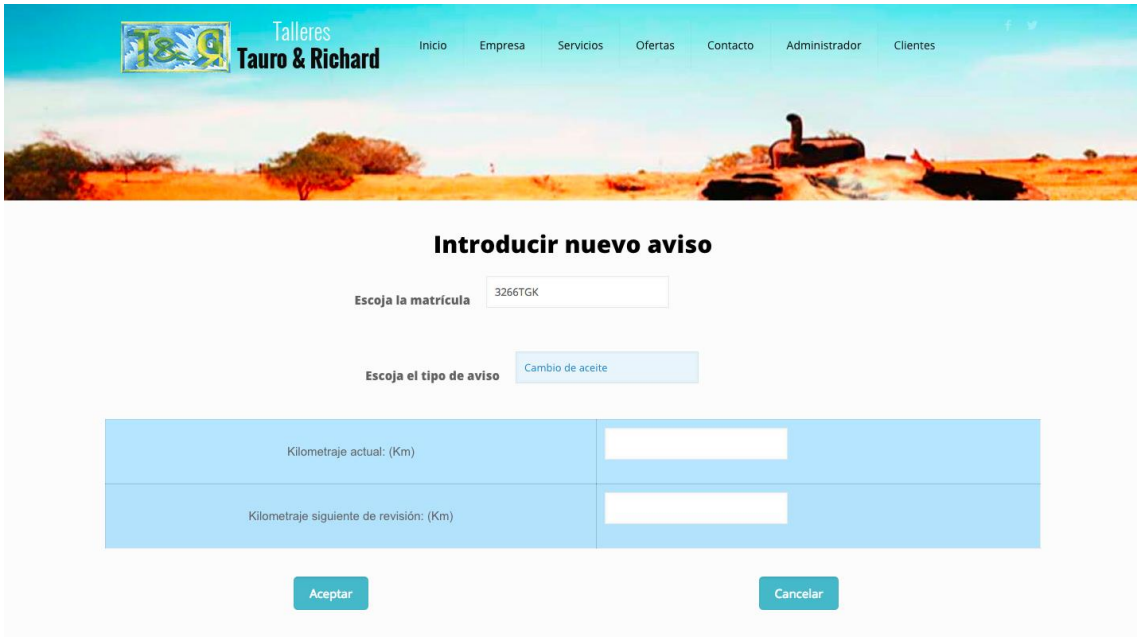


Ilustración 99. - Introducir nuevo aviso - Kilometrajes

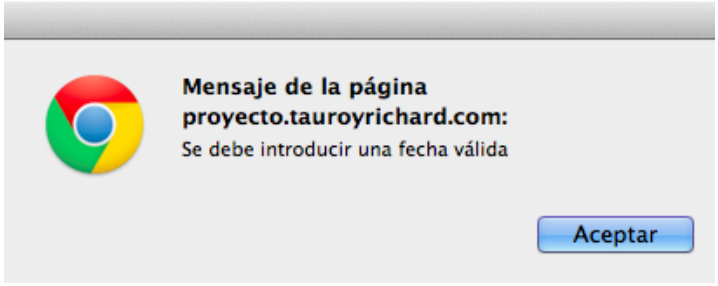


Ilustración 100. - Mensaje de error Fecha inválida

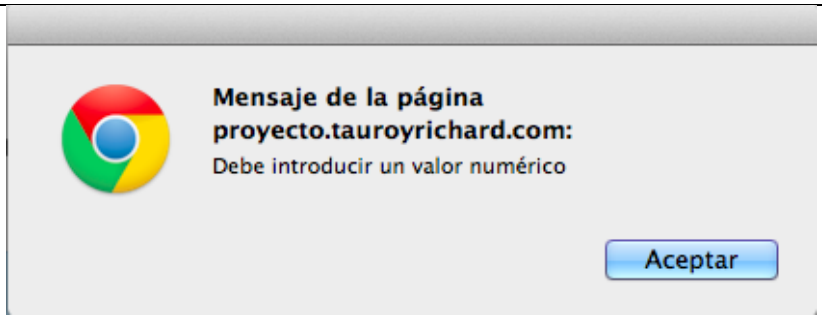


Ilustración 101. - Mensaje de error de valor numérico incorrecto

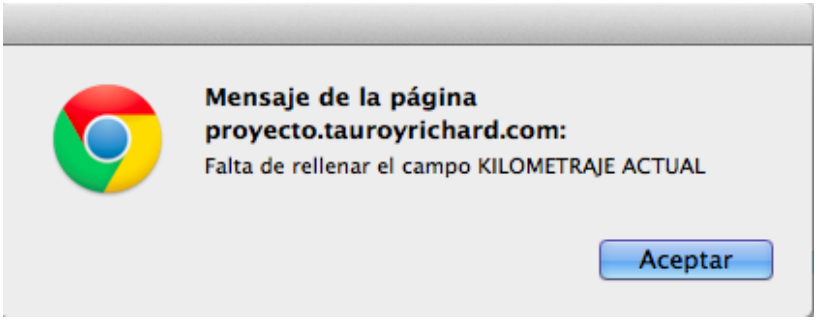


Ilustración 102. - Mensaje de error de falta campos por rellenar

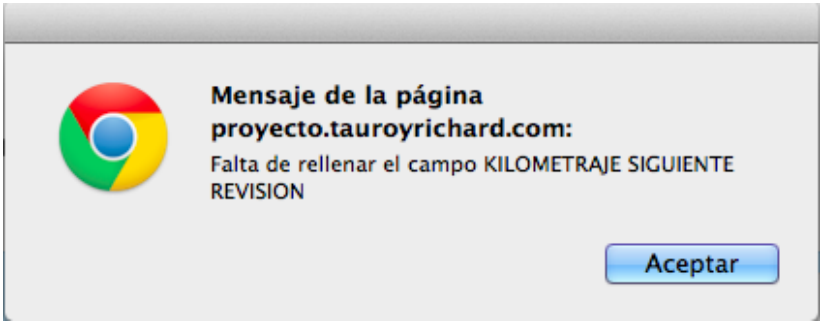


Ilustración 103. - Mensaje de error de falta campos por rellenar

Ilustración 104. - Eliminar aviso

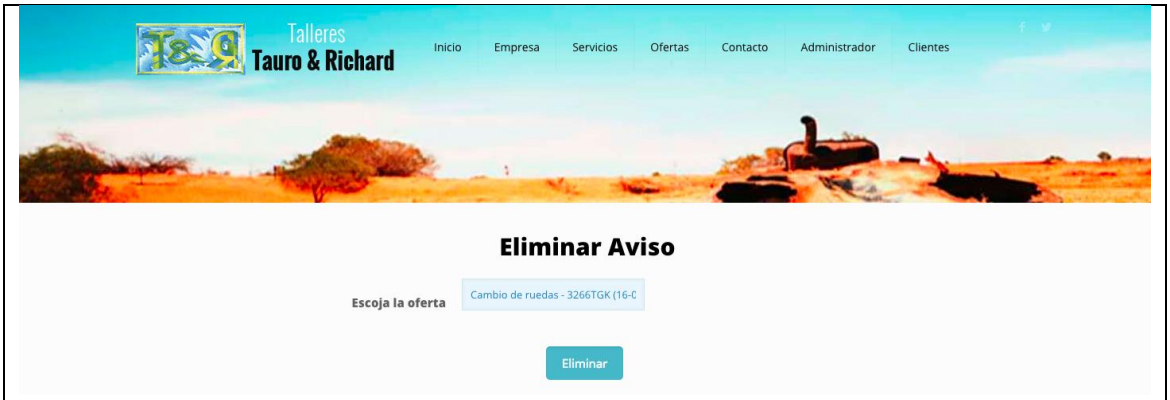


Ilustración 105. - Eliminar aviso seleccionado

10.1.8. Caso de uso **Administrar ofertas**

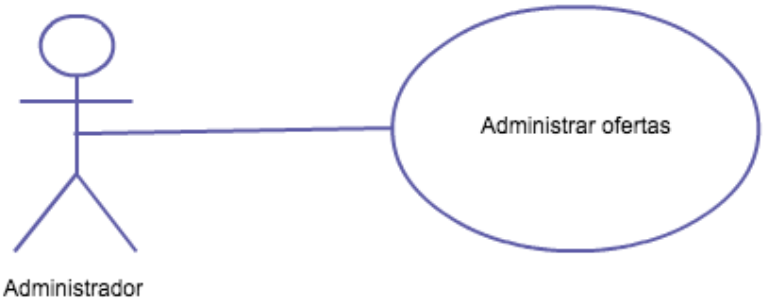


Ilustración 106. - Caso de uso Adminsitrar ofertas

Nombre: Caso de uso Administrar ofertas
Descripción: El administrador podrá insertar, modificar o eliminar la oferta que desee. La oferta que inserte, automáticamente se insertará en el muro de Facebook del taller Tauro & Richard.
Actores: Administrador.
Precondiciones: Estar identificado como administrador.
Requisitos no funcionales: Ninguno.
Flujo de eventos: 1) El administrador, que previamente ha introducido una contraseña en la página "Entorno de administración" (Ilustración 107), pulsará el botón "Administrar ofertas" (Ilustración 108) que le enviará a otra interfaz (Ilustración 109) que será la interfaz en la que aparecerá un menú para escoger la acción que desee realizar en relación con la administración de ofertas. 2) Si selecciona el botón "Introducir nueva oferta" aparecerá una nueva interfaz (Ilustración 110) con un formulario con los campos "Título", "Descripción corta", "Descripción" e "Imagen" que se deberán rellenar. 2.1.) En caso de que no se rellene algún campo aparecerá un cuadro de error (Ilustración 111). 2.2.) Si el administrador quisiera borrar todos los datos existentes y vaciar el formulario, deberá pulsar el botón "Borrar". 2.3.) Si el administrador desea volver al menú de administración de ofertas, sin

introducir ninguna oferta nueva, deberá pulsar el botón “Cancelar”.

2.4.) Una vez rellenados todos los campos se deberá pulsar el botón “Aceptar” y volverá al menú de administración de ofertas (Ilustración 109). Esta oferta aparecerá en la sección “Ofertas” de dicha web (Ilustración 112). La aplicación también insertará dicha oferta automáticamente en el muro de Facebook de Talleres Tauro y Richard (Ilustración 113).

3) Si el administrador selecciona el botón “Modificar oferta” aparecerá una nueva interfaz (Ilustración 114) con un desplegable para elegir la oferta que desee modificar. Una vez elegida la oferta aparecerá otra interfaz (Ilustración 115) con el mismo formulario que el de inserción de nueva oferta pero los campos estarán rellenos con los datos de la oferta seleccionada. El administrador podrá modificar todos los datos que desee. Si desea subir una nueva imagen, solamente deberá insertar una imagen en el campo “Subir nueva imagen” y esta nueva imagen reemplazará a la anterior.

   3.1.) En caso de que se deje algún campo sin rellenar la aplicación mantendrá los datos anteriores de la oferta.

   3.2.) Si el administrador quisiera restaurar los valores antiguos de la oferta, deberá pulsar el botón “Borrar”.

   3.3.) Una vez rellenados todos los campos se deberá pulsar el botón “Aceptar” para hacer efectiva la modificación y volverá al menú de administración de ofertas (Ilustración 109).

4) Si selecciona la opción “Eliminar ofertas”, aparecerá una nueva interfaz (Ilustración 116) con un desplegable para elegir la oferta que se desea eliminar. Una vez seleccionada la oferta aparecerá el botón de eliminar (Ilustración 117). Se debe pulsar dicho botón y la oferta quedará eliminada y volverá al menú de administración de ofertas (Ilustración 109).

Pos condiciones: El administrador podrá insertar, modificar o eliminar una oferta.

Interfaz gráfica:

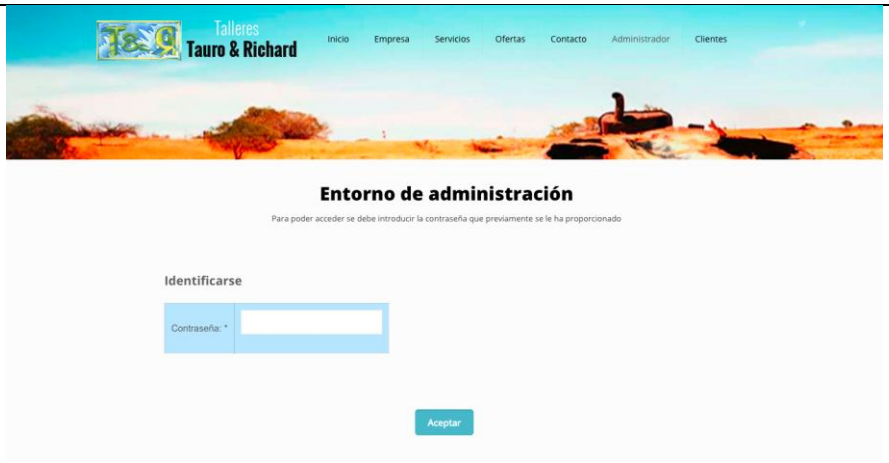


Ilustración 107. - Entorno de administración

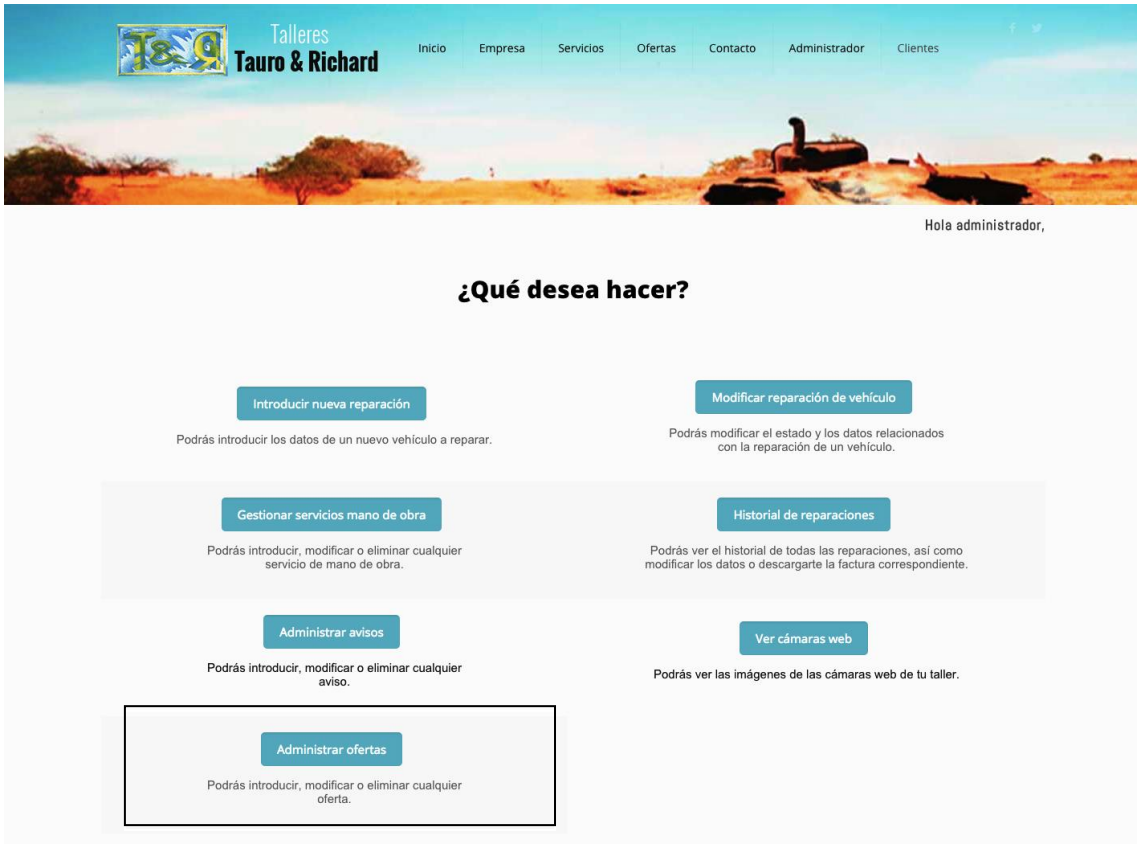


Ilustración 108. - Menú del administrador

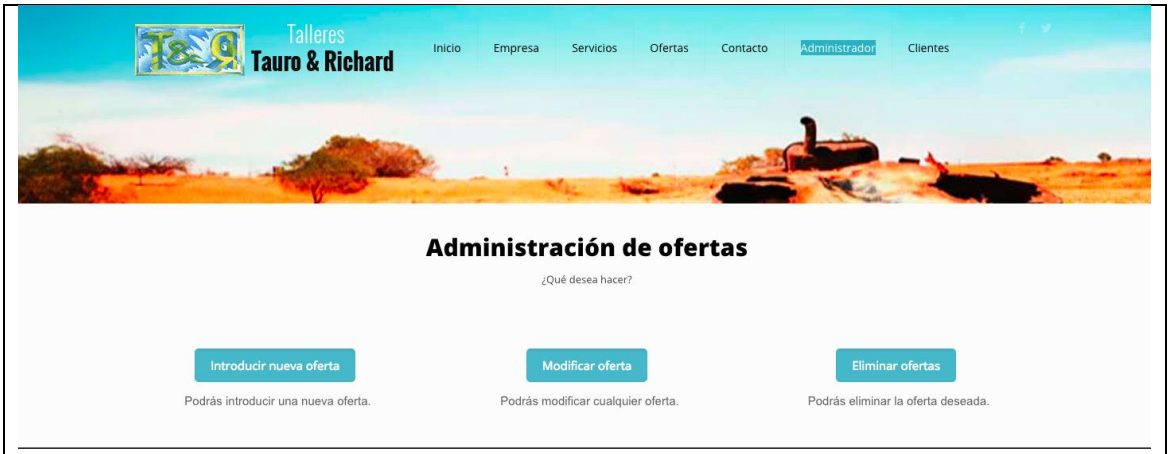


Ilustración 109. - Menú de administrar ofertas

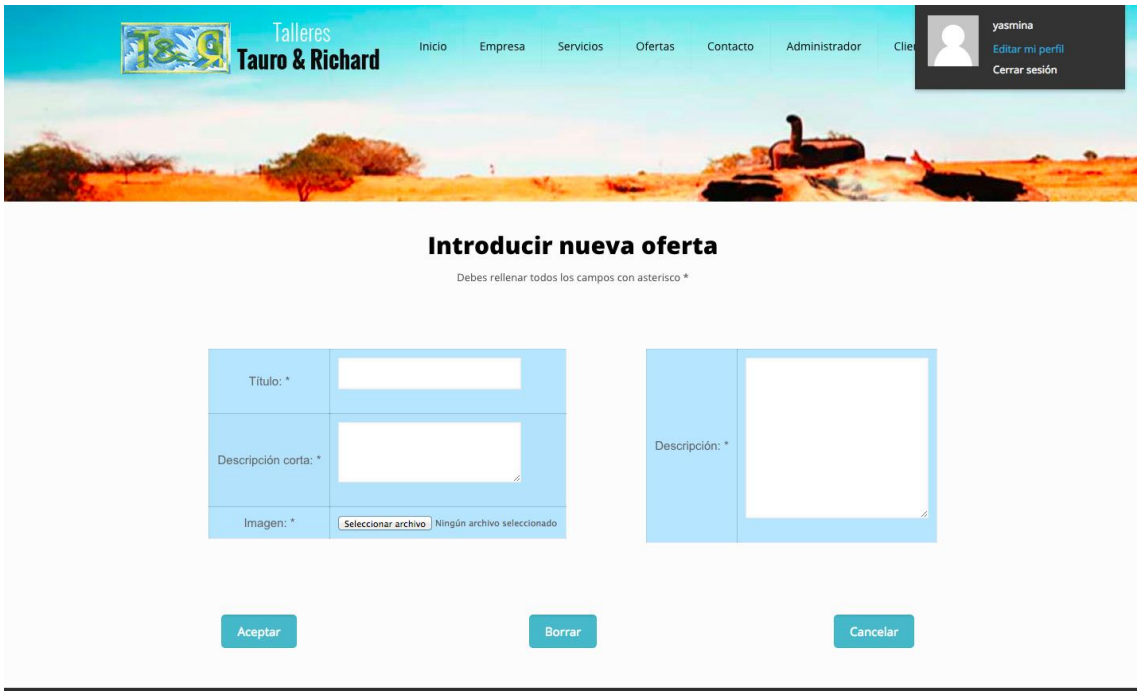


Ilustración 110. - Introducir nueva oferta

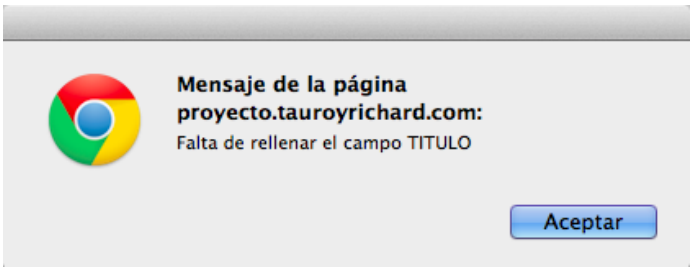


Ilustración 111. - Mensaje de error de campos sin rellenar

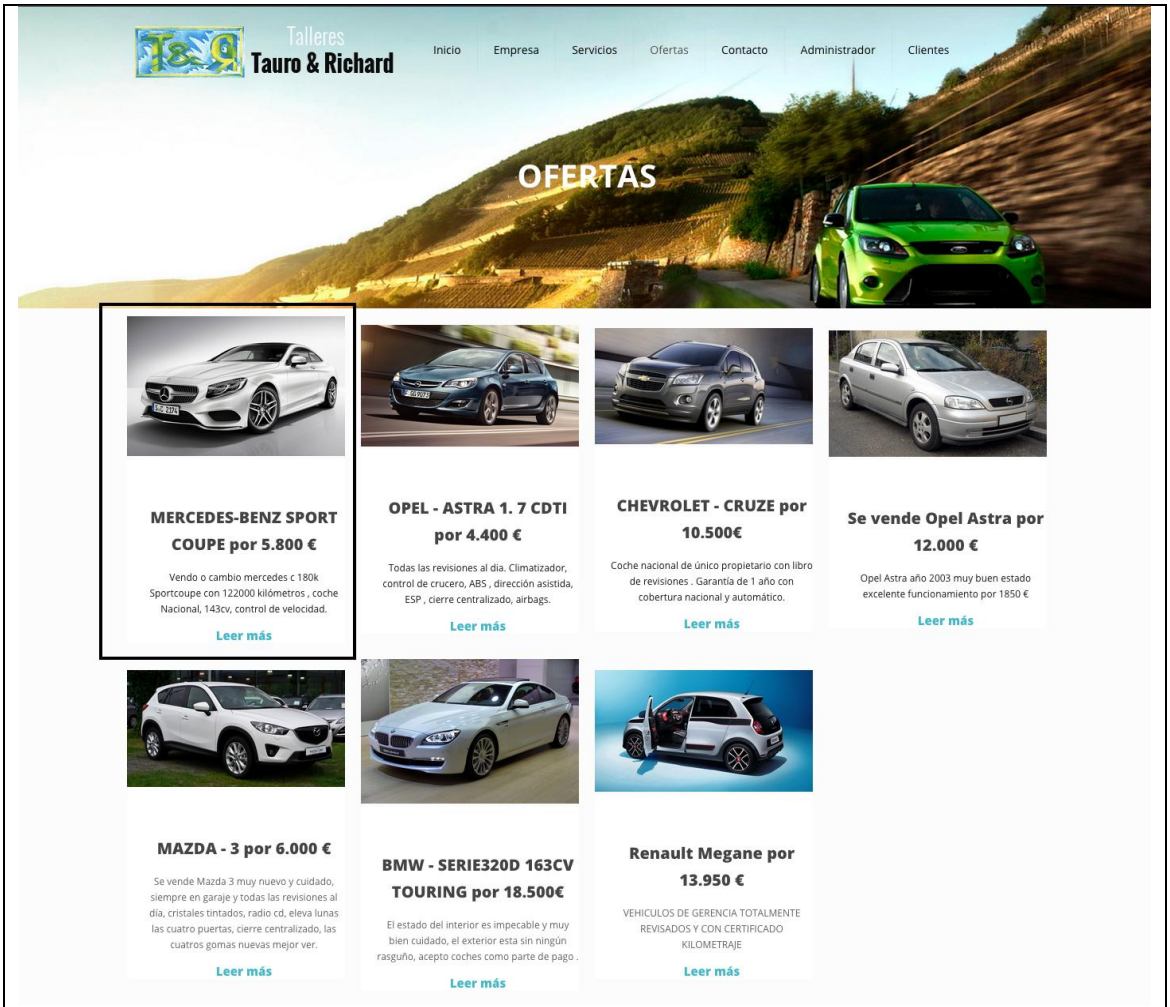


Ilustración 112. - Página de Ofertas

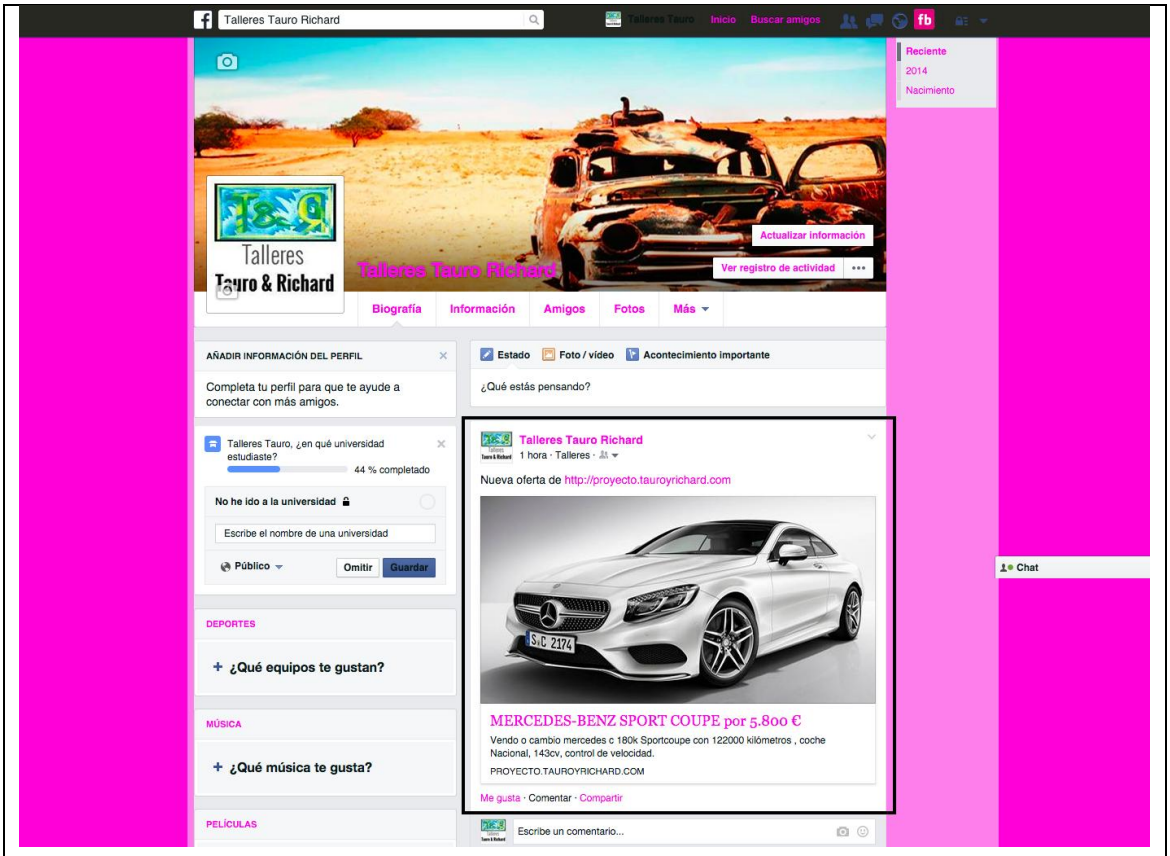


Ilustración 113. - Publicación de oferta en Facebook

Ilustración 114. - Modificar oferta

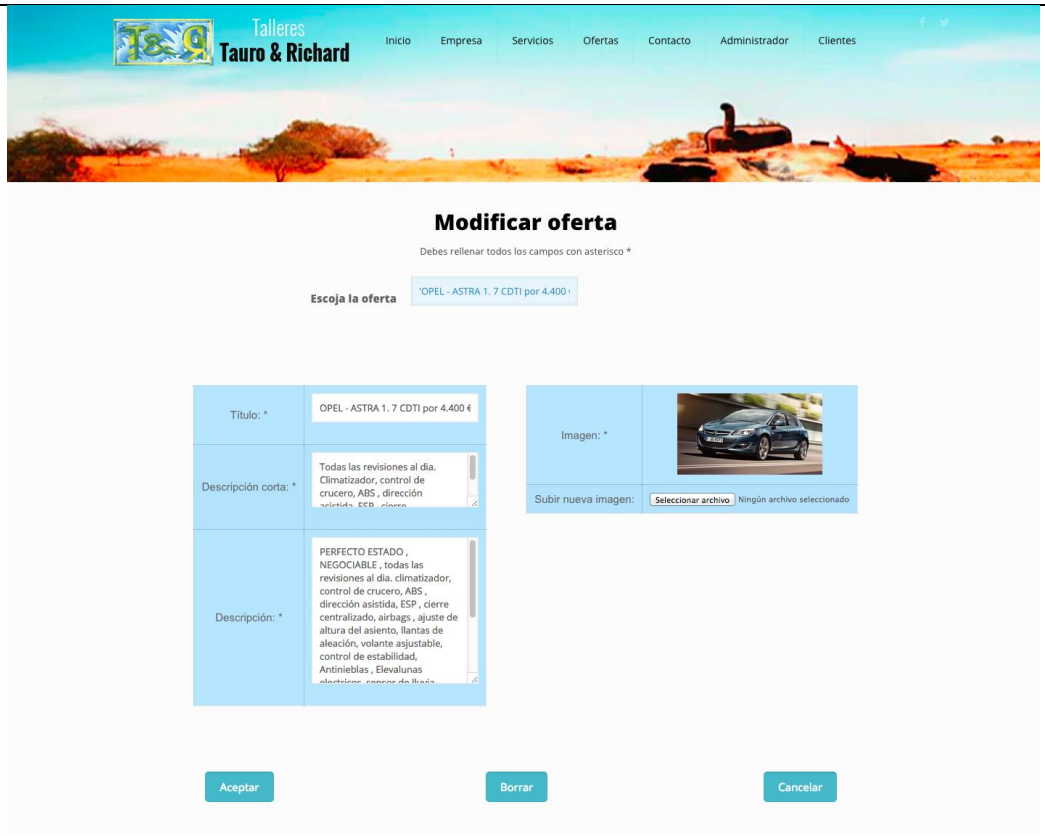


Ilustración 115. - Formulario de modificación de oferta

Ilustración 116. - Eliminar oferta

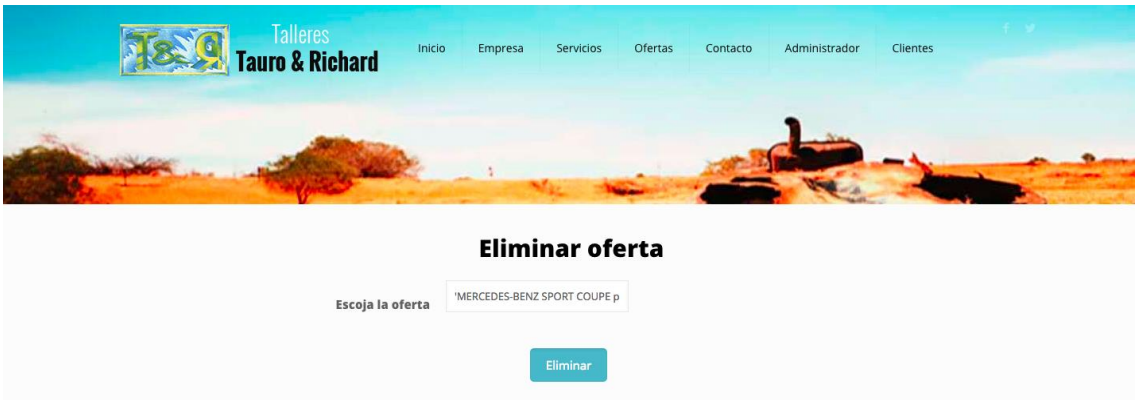


Ilustración 117. - Eliminar oferta seleccionada

10.1.9. Caso de uso **Ver ofertas**

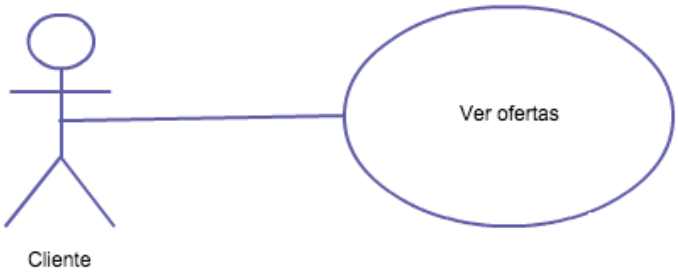


Ilustración 118. - Caso de uso Ver ofertas

Nombre: Caso de uso Ver ofertas
Descripción: El cliente podrá ver las ofertas introducidas por el administrador y que han sido publicadas en Facebook.
Actores: Cliente.
Precondiciones: ninguna.
Requisitos no funcionales: ninguno.
Flujo de eventos: 1) El cliente deberá pulsar el botón "Ofertas" del menú superior (Ilustración 119) y accederá a una nueva interfaz que se llamará "Ofertas" (Ilustración 120) y aparecerán todas las ofertas introducidas por el administrador y publicadas en Facebook. 2) Si el cliente desea visualizar los detalles de una oferta, deberá pulsar encima de ella o pulsar el enlace "Leer más" y se abrirá una nueva interfaz (Ilustración 121) en la que aparecerá el título de la oferta, la descripción y la imagen de la misma.
Pos condiciones: El cliente podrá visualizar todas las ofertas publicadas por el administrador.
Interfaz gráfica:

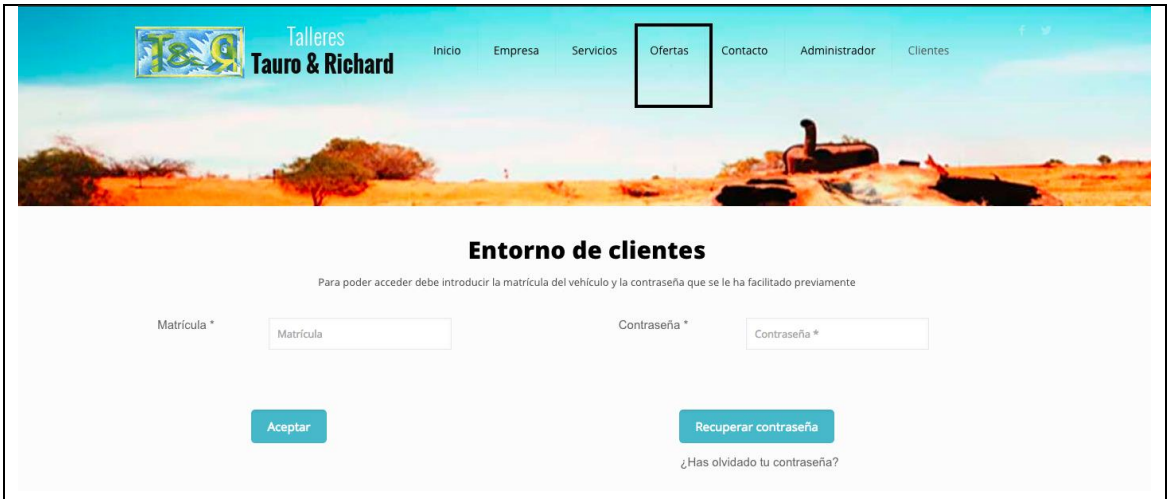


Ilustración 119. - Menú clientes

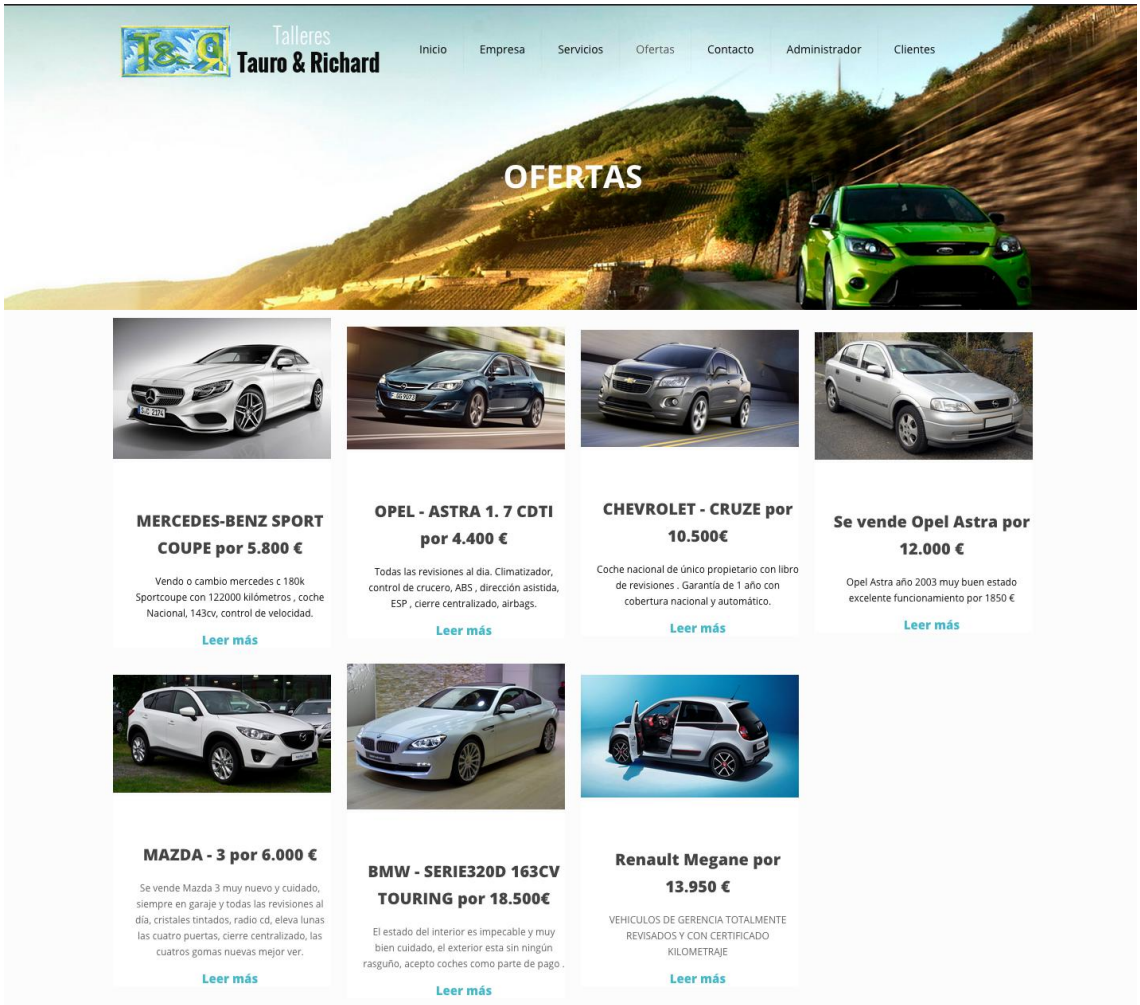


Ilustración 120. - Página de Ofertas

MERCEDES-BENZ SPORT COUPE por 5.800 €

Vendo o cambio mercedes c 180k Sportcoupe con 122000 kilómetros , coche Nacional, 143cv, control de velocidad, encendido automático de luces , climatizador bizona, volante multifunción, ESP, manos libres, ect , recién cambiado aceite y todos los filtros, itv pasada hasta noviembre del 2014 , kilómetros reales se acepta comprobación casa Mercedes, se acepta prueba mecánica , bateria y xenon recien puestos, itv recien pasada el dia 7 de noviembre. acepto cambio por diesel de 5 puertas.

Ilustración 121. - Detalles de la oferta

10.1.10. Caso de uso **Identificación cliente**

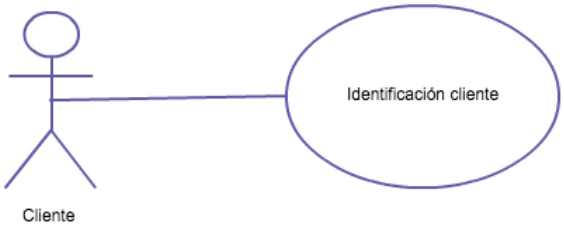


Ilustración 122. - Caso de uso identificación cliente

Nombre: Caso de uso Identificación cliente
Descripción: El cliente deberá introducir su matrícula y contraseña, proporcionada por el administrador, y pulsar "Aceptar" y comprobando que la matrícula y contraseña sea correcta se le enviará a la interfaz de pago online de la factura.
Actores: Cliente.
Precondiciones: ninguna.
Requisitos no funcionales: ninguna .
Flujo de eventos: 1) Aparecerá la interfaz (Ilustración 123) en la que aparecerán el campo Matrícula y Contraseña y deberá rellenarse y pulsar "Aceptar" y se comprobará con la base de datos si la contraseña es correcta y se le conducirá a la siguiente interfaz (Ilustración 126). 2) En caso de que la matrícula o contraseña sean incorrectas se mostrará un mensaje de error (Ilustración 124). 3) En caso de no rellenarse algunos de los campos, se indicará un mensaje de error (Ilustración 125).
Pos condiciones: Se habrá identificado el cliente y se le llevará a la página de inicio del cliente para que elija la acción que desee realizar, ya sea modificar sus datos personales o ver las facturas o pagarlas.
Interfaz gráfica:

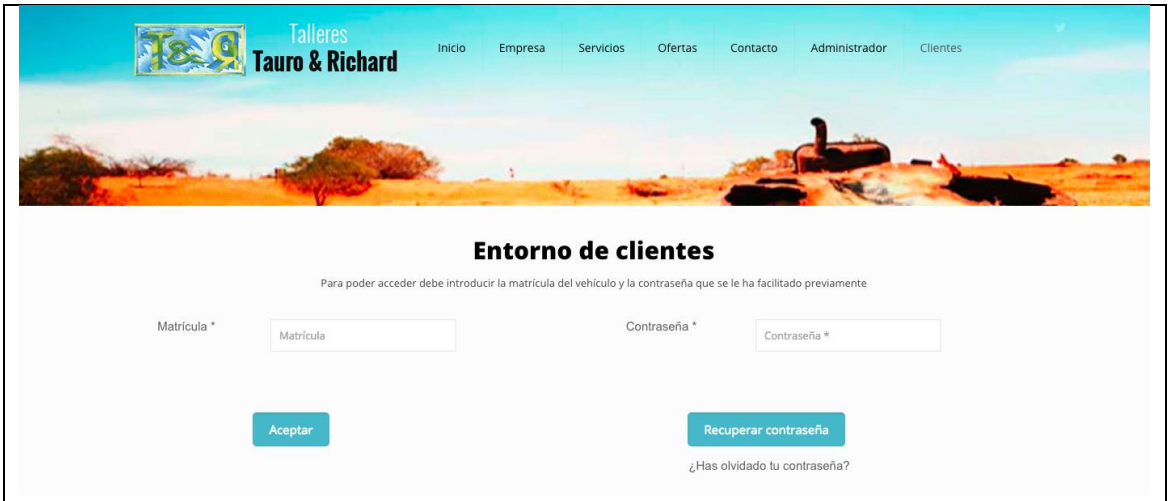


Ilustración 123. - Entorno de clientes

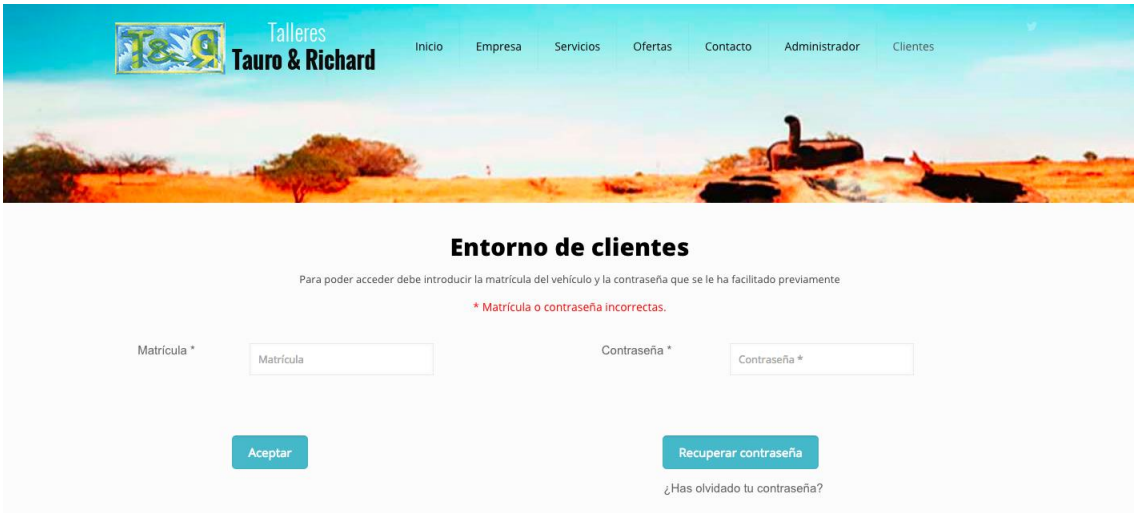


Ilustración 124. - Mensaje de error de los datos incorrectos

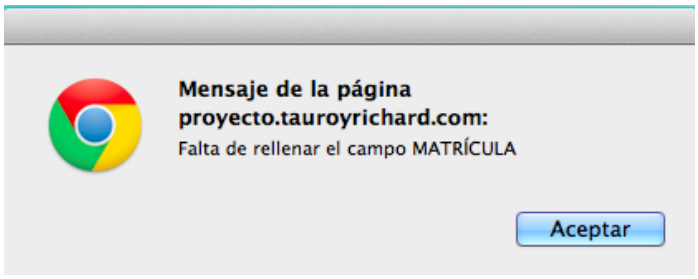


Ilustración 125. - Mensaje de error del campo matrícula

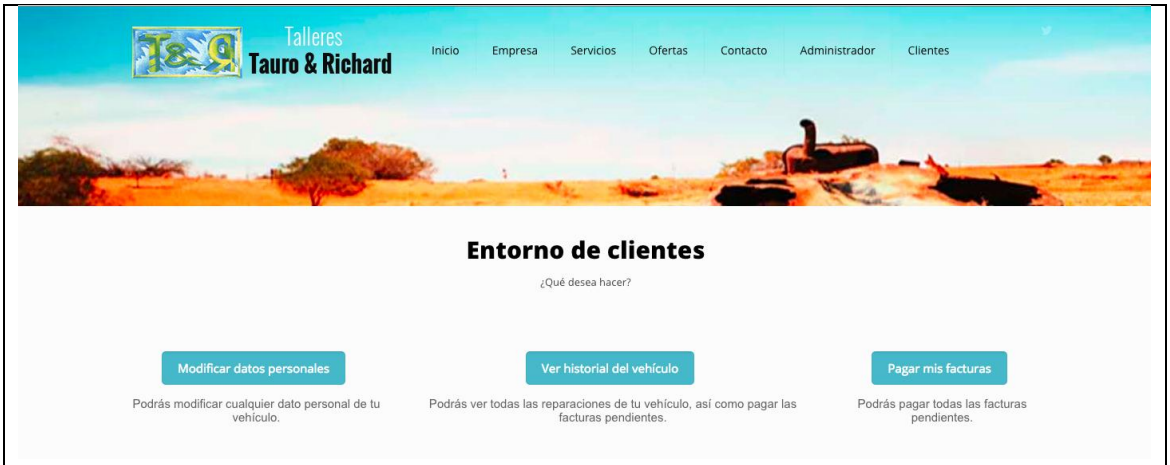


Ilustración 126. - Menú de clientes

10.1.11. Caso de uso **Recuperar contraseña**

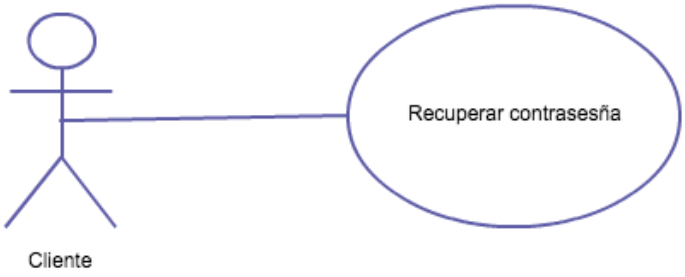


Ilustración 127. - Caso de uso identificación cliente

Nombre: Caso de uso Recuperar contraseña

Descripción: El cliente, en caso de que no recuerde la contraseña o no tenga el email recibido por la web con la contraseña, podrá recuperarla.

Actores: Cliente.

Precondiciones: Ninguna.

Requisitos no funcionales: Ninguno.

Flujo de eventos:

1) El cliente, que se encontrará en el "Entorno de clientes" (Ilustración 128), deberá pulsar el botón de "Recuperar contraseña".

2) Aparecerá una pantalla (Ilustración 129) con una casilla en la que el cliente deberá introducir la matrícula del vehículo y deberá pulsar el botón "Aceptar". Esta página le reconducirá a la página de entorno de clientes (Ilustración 128) para poder introducir la nueva contraseña.

3) Finalmente, el sistema le enviará un email con una nueva contraseña al cliente con la que podrá identificarse.

Pos condiciones: El cliente dispondrá de una nueva contraseña para poder identificarse en la aplicación.

Interfaz gráfica:

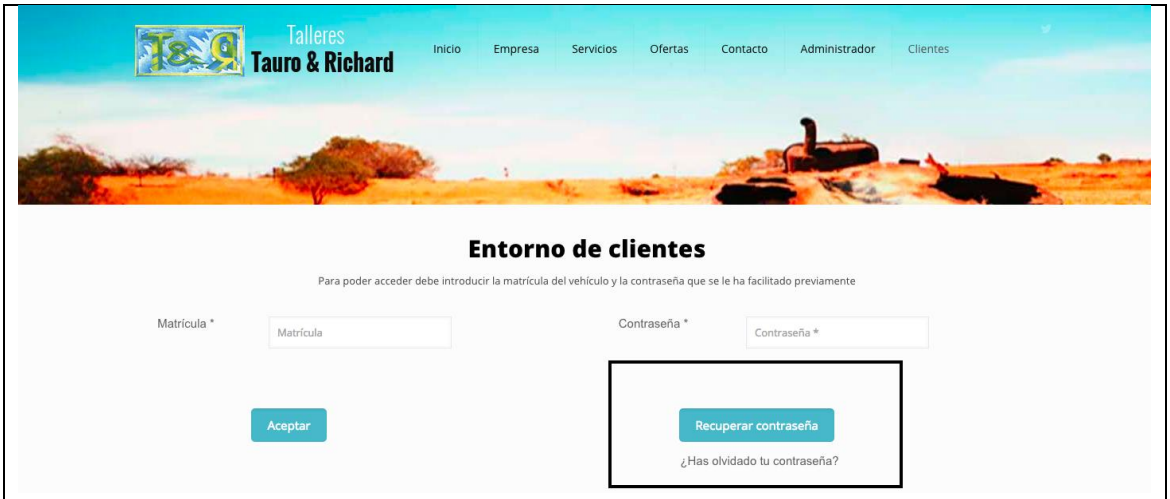


Ilustración 128. - Entorno de clientes

Ilustración 129. - Formulario de recuperación de contraseña

10.1.12. Caso de uso **Modificar datos personales**

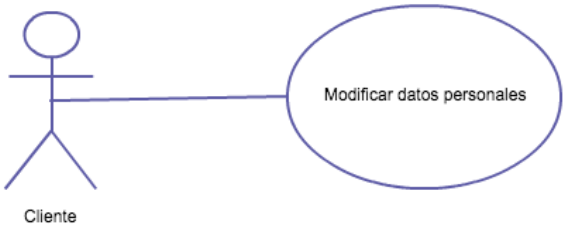


Ilustración 130. – Caso de uso Modificar datos personales

Nombre: Caso de uso Modificar datos personales

Descripción: El cliente podrá modificar cualquier dato personal que desee.

Actores: Cliente.

Precondiciones: Estar identificado como cliente.

Requisitos no funcionales: Ninguno.

Flujo de eventos:

1) El cliente, que previamente ha introducido la matrícula y la contraseña en la página "Entorno de clientes" (Ilustración 131), pulsará el botón "Modificar datos personales" (Ilustración 132).

2) Aparecerá una nueva interfaz (Ilustración 133) que será la interfaz en la que aparecerán todos los campos del formulario para poder cambiar cualquier dato relacionado con una reparación de un vehículo. Los campos aparecerán rellenados con los datos que ya se tienen de ese cliente registrado en la base de datos.

3) En el formulario aparecerán los campos de Nombre y apellidos, dirección, código postal, ciudad, provincia, DNI, teléfono, email, modelo del vehículo y kilometraje ya rellenos. El cliente podrá hacer la modificación que quiera en cada uno de ellos.

    3.1.) Aparecerá otro campo que se llamará "Nueva contraseña" que solo será necesario rellenarlo en caso de que se quiera cambiar la contraseña actual.

4) Una vez modificados todos los datos que el cliente desee deberá pulsar el botón "Aceptar."

    4.1.) En caso de que no se rellenen alguno de los campos obligatorios aparecerá el

siguiente cuadro de error (Ilustración 134).

4.2.) En caso de que el campo email haya sido rellenado pero no se introduzca con el formato correcto, saldrá un error de formato de email incorrecto (Ilustración 135).

4.3.) En caso de que el teléfono no se introduzca con un formato adecuado, saldrá un mensaje de error de numero de teléfono incorrecto (Ilustración 136).

5) En caso de que el cliente no quiera modificar nada y volver al menú de clientes (Ilustración 132), deberá pulsar el botón “Cancelar”.

6) Si el cliente quisiera borrar todos los datos existentes y rellenar el formulario entero, deberá pulsar el botón “Borrar”.

Pos condiciones: El cliente habrá podido modificar cualquier dato personal que desee.

Interfaz gráfica:

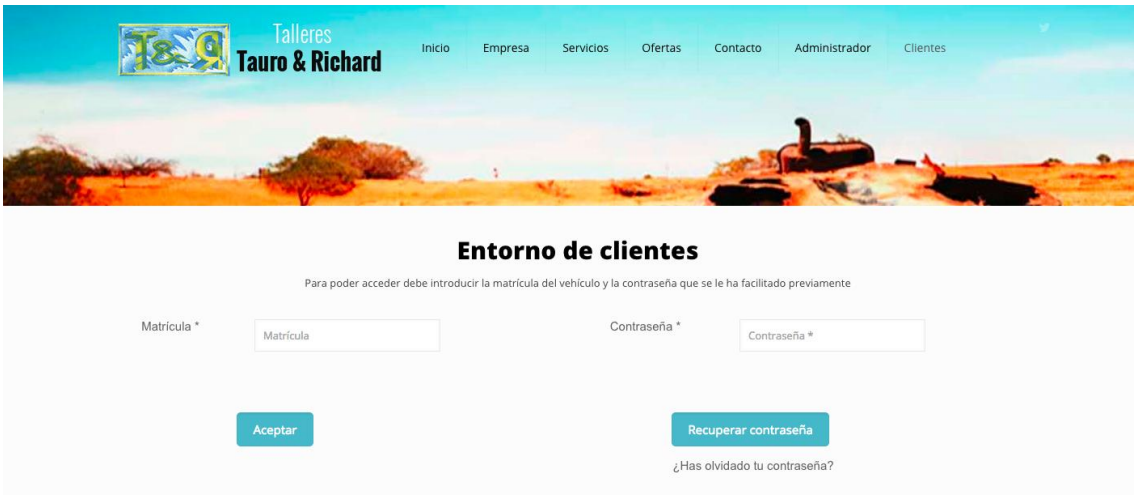


Ilustración 131. - Entorno de clientes

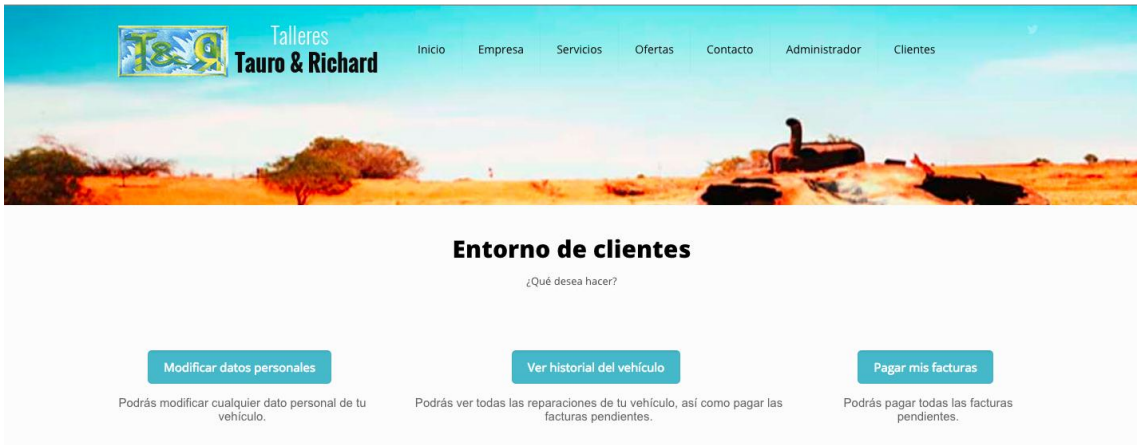


Ilustración 132. - Menú de clientes

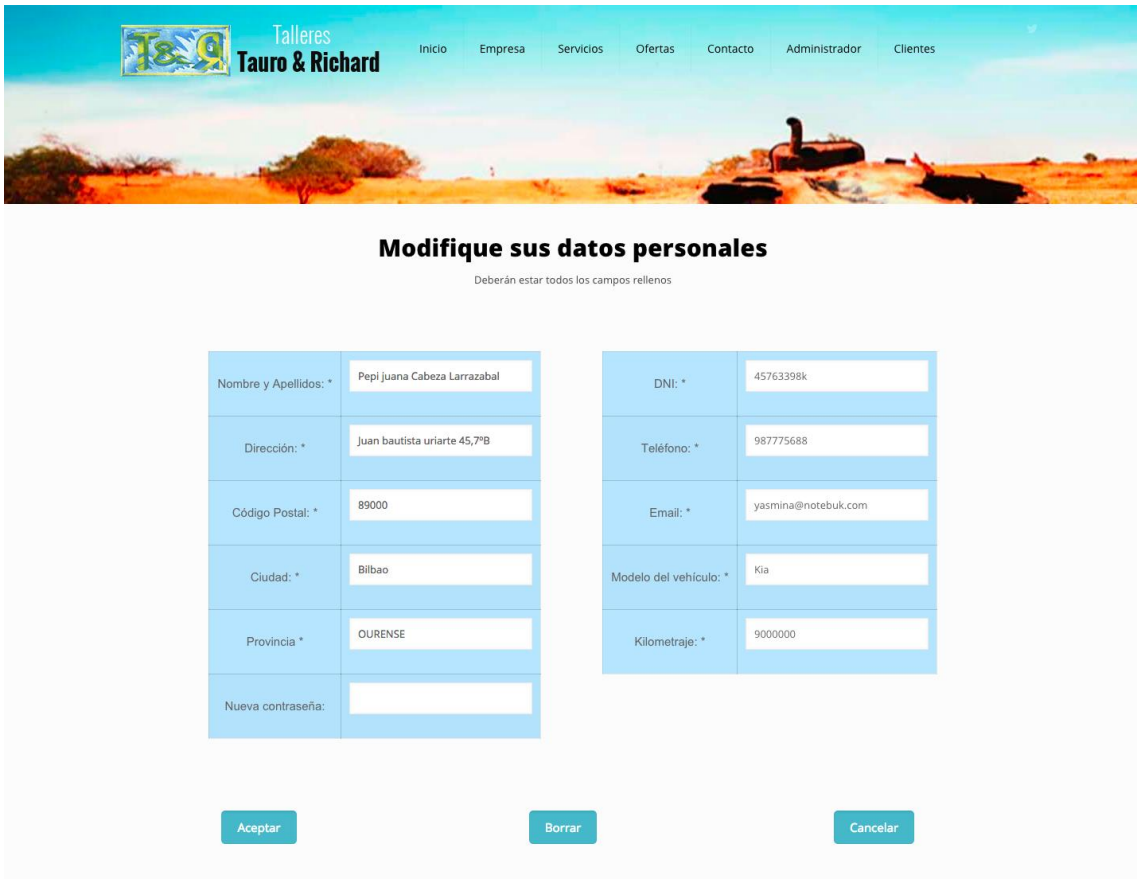


Ilustración 133. - Formulario de modificación de datos personales

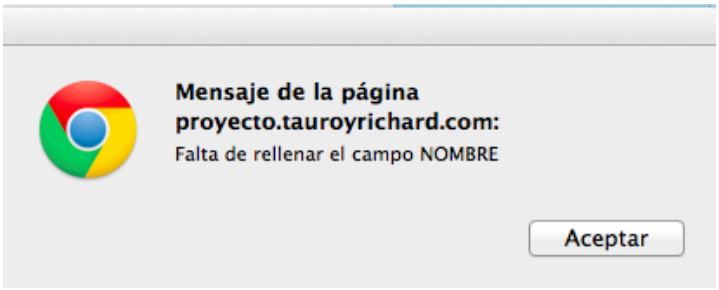


Ilustración 134. - Mensaje de error de campo nombre

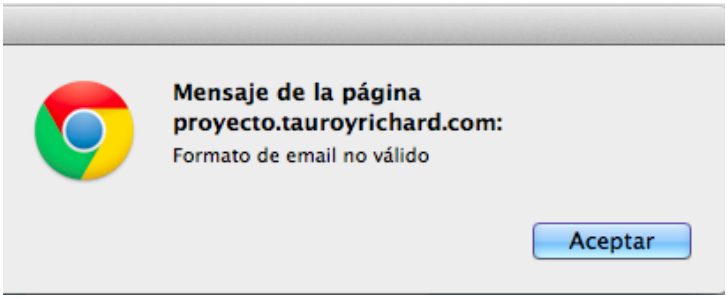


Ilustración 135. - Mensaje de error de campo email

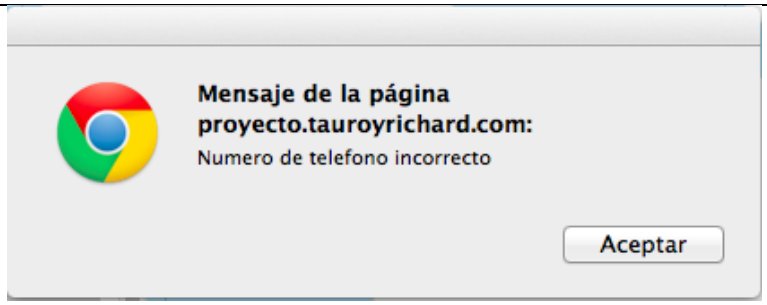


Ilustración 136. - Mensaje de error de campo teléfono

10.1.13. Caso de uso **Ver historial del vehículo**

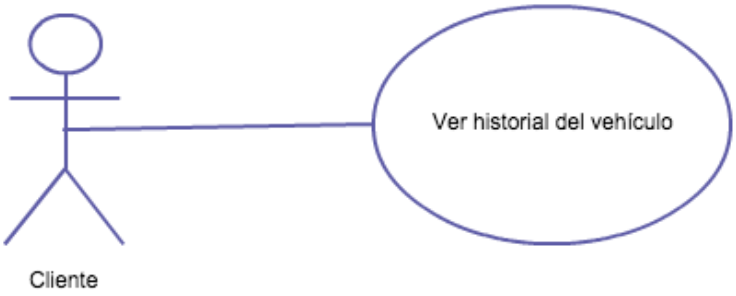


Ilustración 137. – Caso de uso Ver historial del vehículo

Nombre: Caso de uso Ver historial del vehículo

Descripción: El cliente podrá ver todo el historial de su vehículo, así como pagar las facturas pendientes.

Actores: Cliente.

Precondiciones: Estar identificado como cliente.

Requisitos no funcionales: Ninguno.

Flujo de eventos:

1) El cliente, que previamente ha introducido la matrícula y la contraseña en la página "Entorno de clientes" (Ilustración 138), pulsará el botón "Ver historial del vehículo" (Ilustración 139).

2) Aparecerá una nueva interfaz (Ilustración 140) en la que aparecerán todas las reparaciones de ese vehículo en el taller y el cliente podrá realizar las siguientes acciones:

   2.1.) El cliente podrá ver la factura de cada reparación pulsando el botón "Ver factura" (Ilustración 141), habiendo seleccionado alguna reparación previamente. Se abrirá una nueva pestaña mostrando la factura en formato pdf (Ilustración 142), que se podrá visualizar correctamente y descargarla si así lo desea. En caso de que la factura esté pagada, aparecerá la factura con un sello de pagado (Ilustración 143).

   2.2.) Si el cliente desea no realizar ninguna acción deberá pulsar el botón "Volver atrás" que le devolverá a la página de entorno de clientes (Ilustración 139).

   2.3.) El cliente podrá pagar la factura de la reparación que desee, siempre y cuando la factura no haya sido abonada previamente y la reparación esté finalizada. Para

ello deberá seleccionar una reparación y pulsar el botón “Pagar ahora” (Ilustración 144). Aparecerá una nueva interfaz (Ilustración 145) en la que se podrá visualizar la factura al momento, o si lo desea descargarla, pulsando el botón “Descargar”. Se deberá seleccionar con que método de pago se desea realizar el abono de la factura, siendo las dos opciones “Transferencia bancaria” o “Paypal”.

2.3.1) Si selecciona la opción “Transferencia bancaria” aparecerá una nueva interfaz de agradecimiento (Ilustración 146) en la que aparecerán los datos de la fecha, nombre y apellidos, importe de la factura y matrícula del vehículo. También hay un botón que permitirá al cliente descargarse la factura. Por último, aparecerán los datos bancarios para que el cliente pueda realizar la transferencia. La aplicación también le enviará al cliente un email con todos los datos anteriores detallados en él. Si el cliente desea volver al historial de reparaciones deberá pulsar “Volver atrás”.

2.3.2) Si selecciona la opción “Paypal” se le reconducirá automáticamente a la plataforma de paypal (Ilustración 147) para proceder a realizar el pago a través de la misma. Una vez terminado el pago se le reconducirá a una página de agradecimiento de la aplicación (Ilustración 148). El cliente recibirá un email con los datos detallados de su pago.

2.3.3) Si el cliente quisiera volver al menú sin realizar ninguna acción deberá pulsar el botón “Volver atrás”.

Pos condiciones: El cliente habrá podido ver el historial de las reparaciones de su vehículo, ver todas las facturas de las mismas, así como abonar las facturas pendientes a través de la aplicación.

Interfaz gráfica:

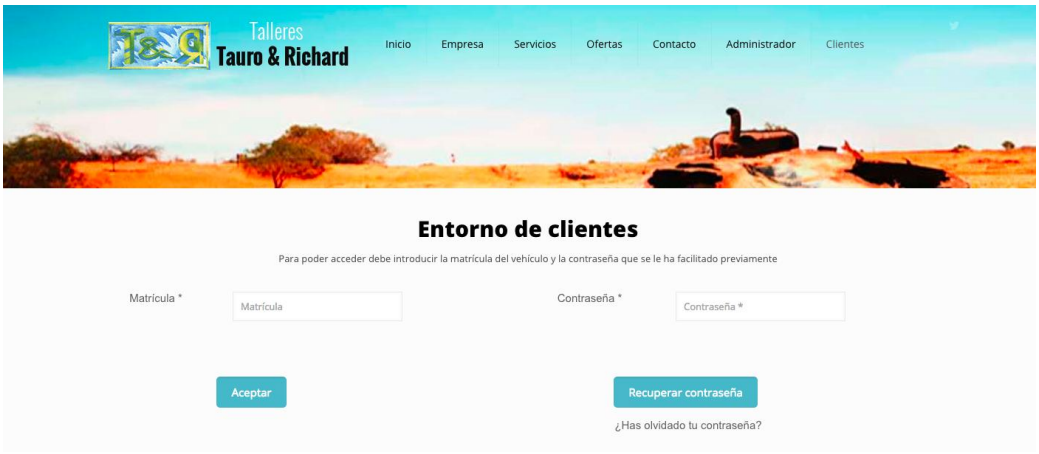


Ilustración 138. - Entorno de clientes

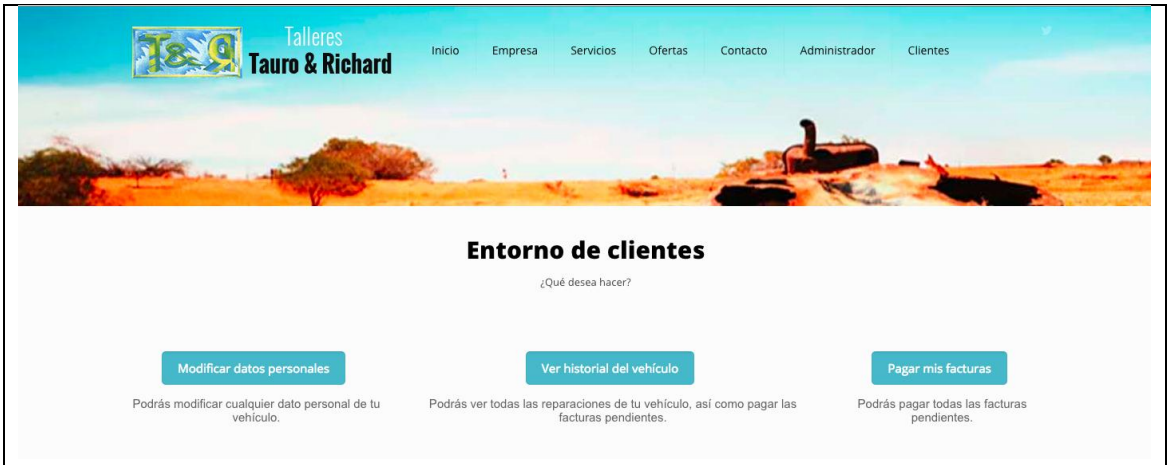


Ilustración 139. - Menú de clientes

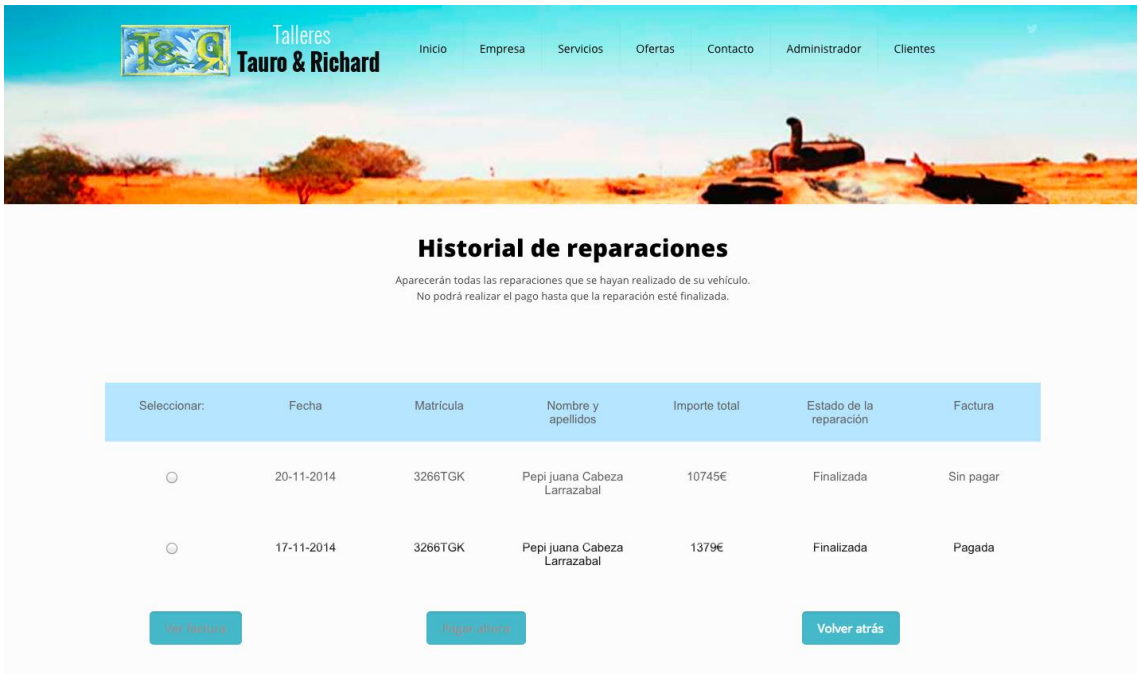


Ilustración 140. - Historial de reparaciones

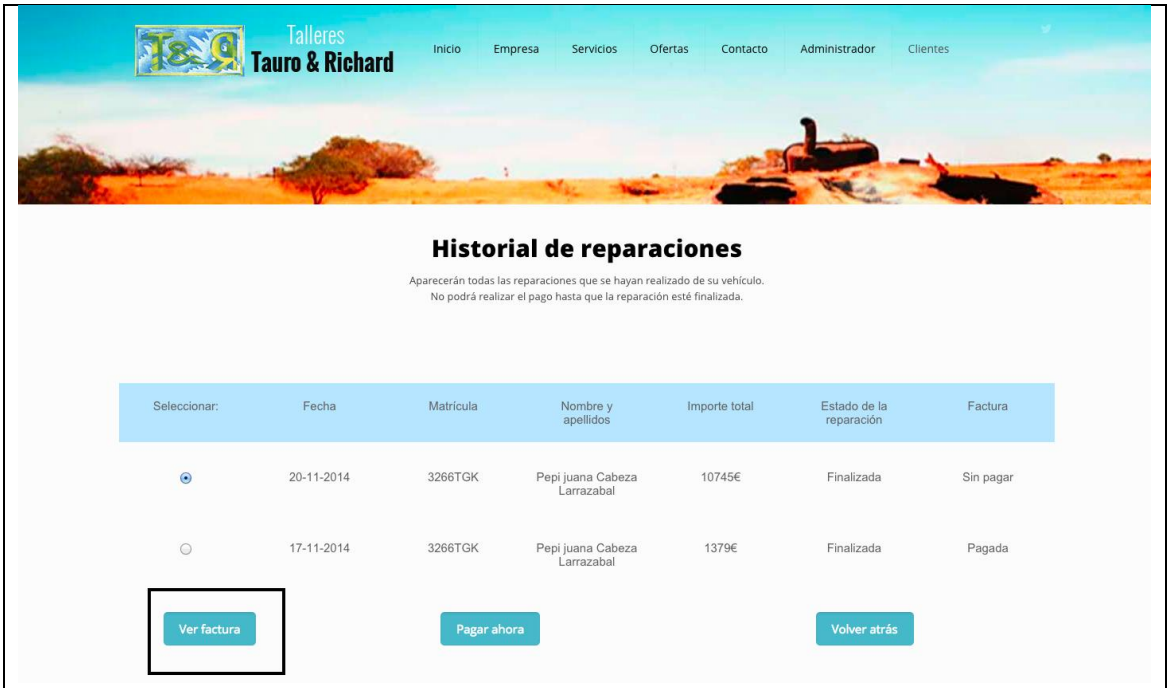


Ilustración 141. - Ver factura

Talleres Tauro & Richard

FACTURA

CIF: E - 48620198
Polígono Aperribai Auzoa S/N, PABELLÓN 6, Galdakao 48960, Vizcaya - España
94 449 25 77 / 667557962 / 610424613
Email: talleres@tauroyrichard.com

Número de factura: 39
Fecha: 18-11-2014

DATOS DEL CLIENTE

Nombre y apellidos: Pepita griffindor
Dirección: Benavente 2, bajo , 43990 , CUENCA - España
DNI: 45879906E
Teléfono: 943336789
Email: yasmina@notebuk.com

DATOS DEL VEHÍCULO

Matrícula: 9098WQE
Modelo: Kia Sorento
CIA: Linea directa

Concepto	Precio Unitario	Importe
4 x Ruedas	100	400
2 x Escobillas	20	40
Arreglo de puerta trasera	1200	1200

Concepto mano de obra	Importe

Base imponible	IVA	Cuota de IVA	Importe total
1640€	21%	344€	1984€

FIRMA DEL CLIENTE

FIRMA Y SELLO DE LA EMPRESA

Ilustración 142. - Factura de la reparación sin pagar

Talleres Tauro & Richard

FACTURA

CIF: E - 48620198
Polígono Aperribai Auzoa S/N, PABELLÓN 6, Galdakao 48960, Vizcaya - España
94 449 25 77 / 667557962 / 610424613
Email: talleres@tauroyrichard.com

Número de factura: 38
Fecha: 18-11-2014

DATOS DEL CLIENTE

Nombre y apellidos: Jon fernandez
Dirección: San miguel de salinas, 21 2ª B , 8900 , A CORUÑA - España
DNI: 222222L
Teléfono: 678675282
Email: jon@gmail.com

DATOS DEL VEHÍCULO

Matrícula: 2987DFS
Modelo: OBTUS
CIA: Axa seguros

PAGADO

Concepto	Precio Unitario	Importe
Porton trasero	349	349
Luz delantera	120	120
2x Escobillas	20	40
2x Ruedas traseras	80	160

Concepto mano de obra	Importe
Limpieza : Integral (Con tapicerías incluídas)	80€
Limpieza : Interior+exterior	120€

Base imponible	IVA	Cuota de IVA	Importe total
6790€	21%	1426€	8216€

FIRMA DEL CLIENTE

FIRMA Y SELLO DE LA EMPRESA

Ilustración 143. - Factura de la reparación pagada

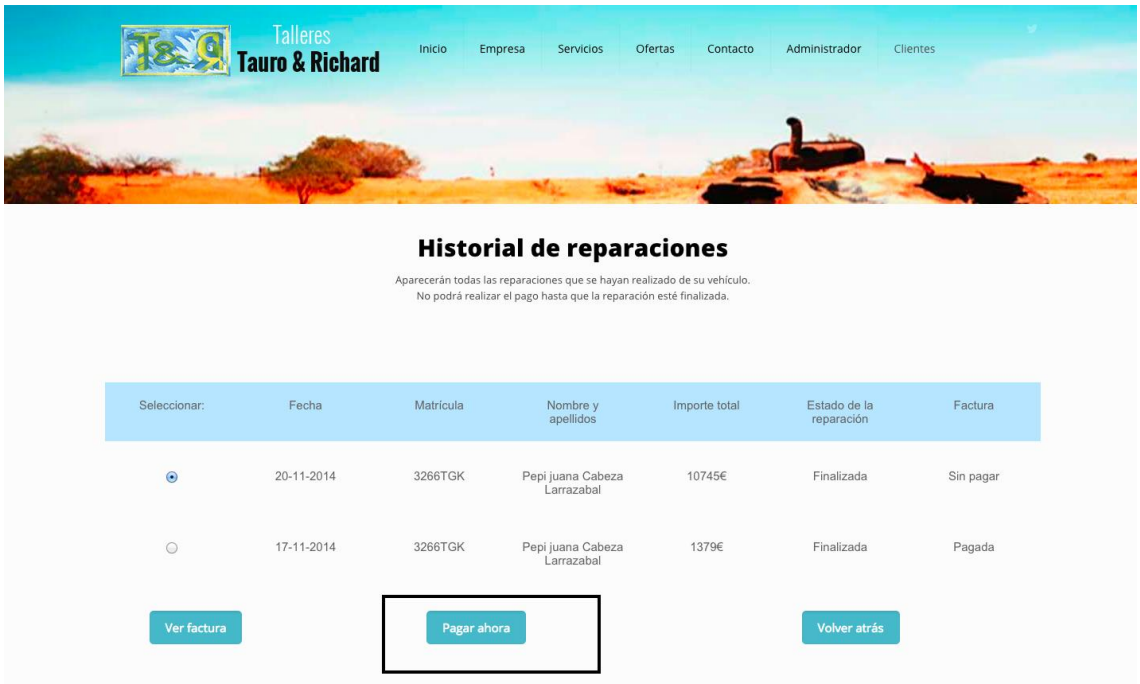


Ilustración 144. - Pagar factura de la reparación

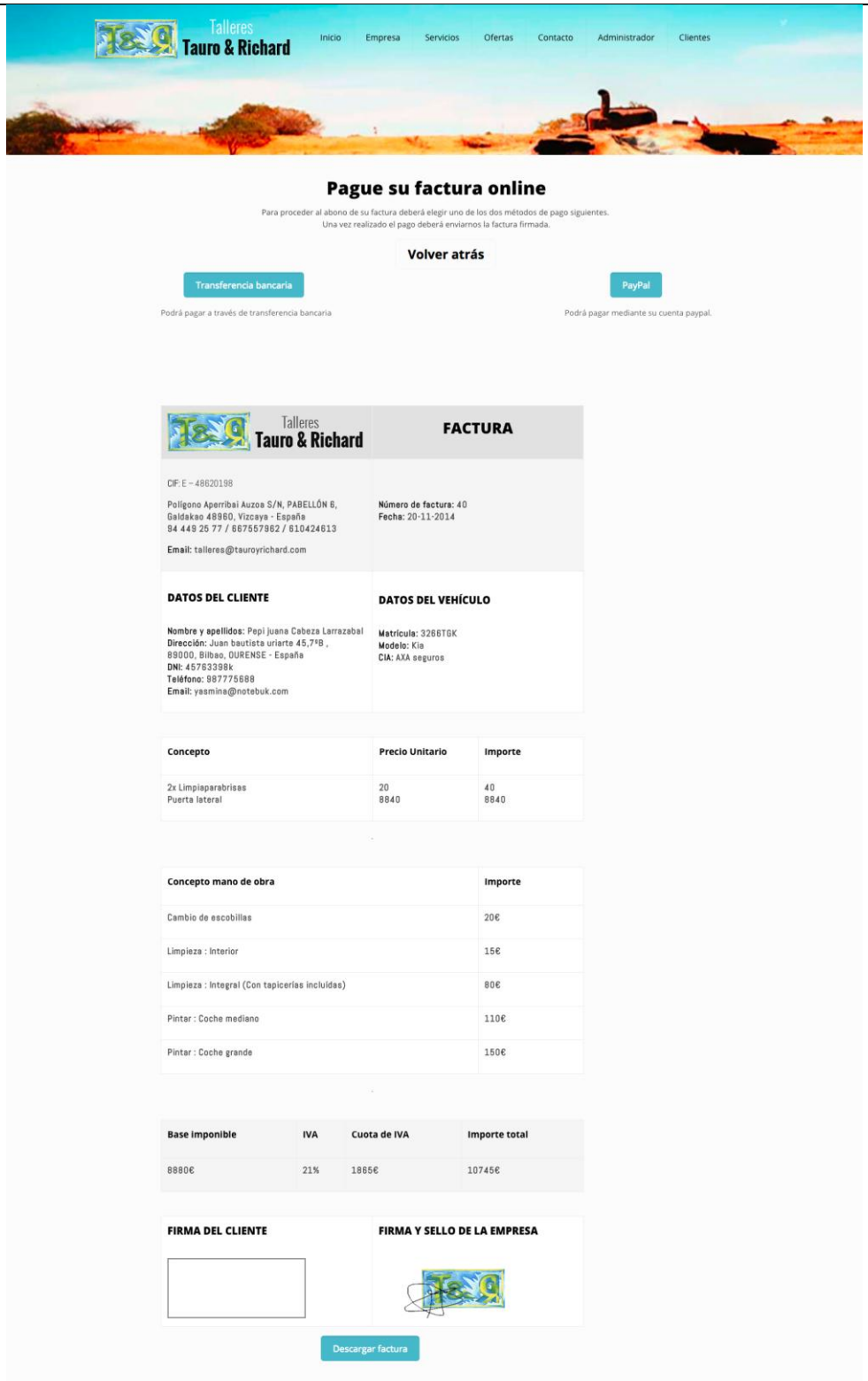


Ilustración 145. - Pagar y ver la factura

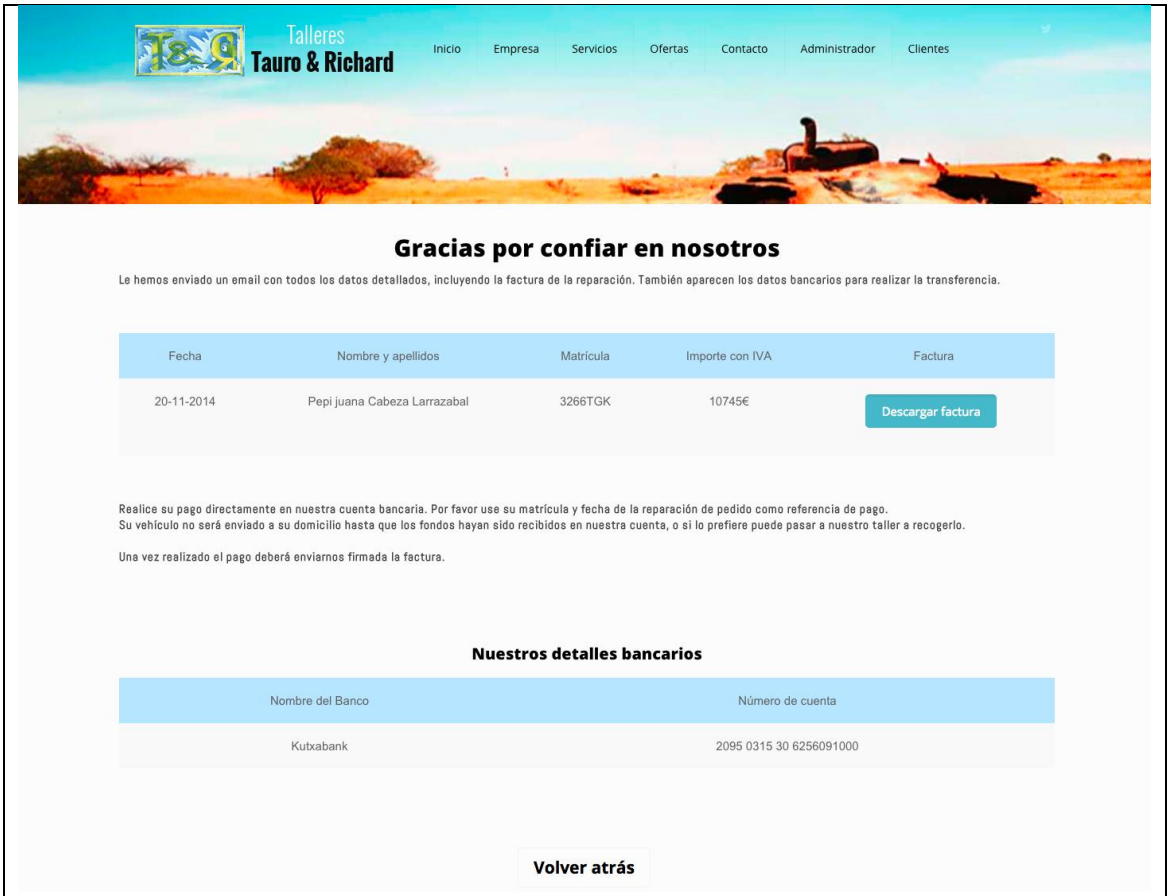


Ilustración 146. - Pago con transferencia bancaria

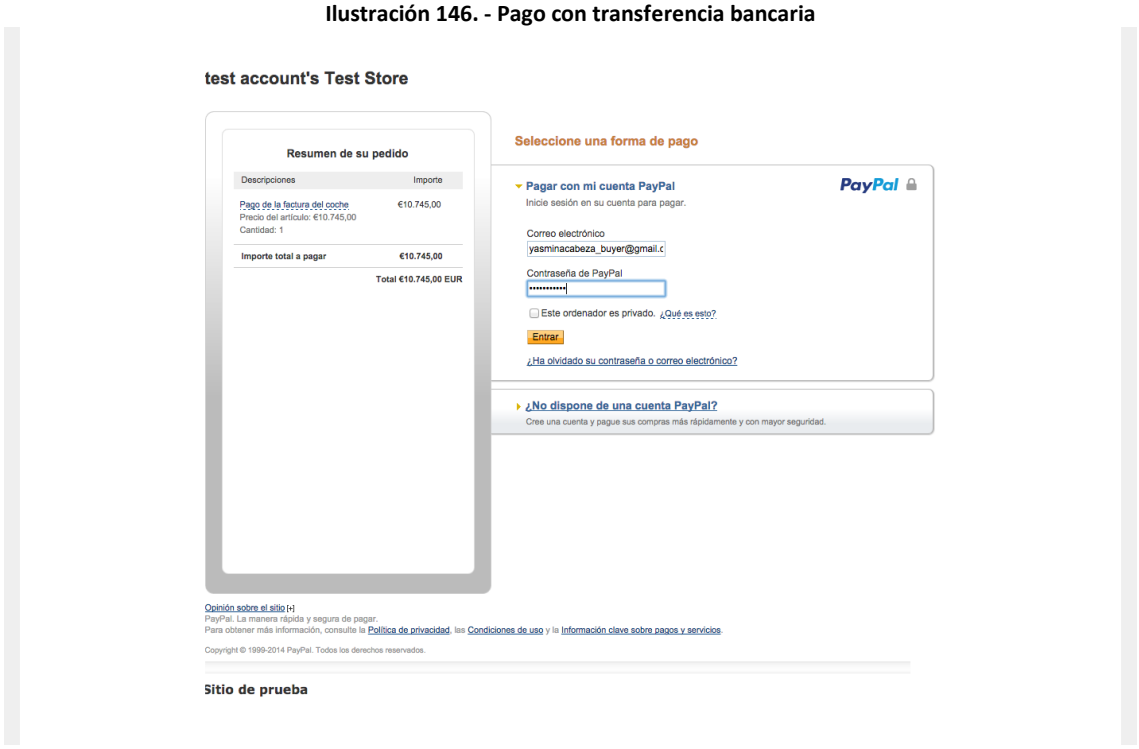


Ilustración 147. - Pago con Paypal

Ilustración 148. - Página de agradecimiento

10.1.14. Caso de uso **Pagar mis facturas**

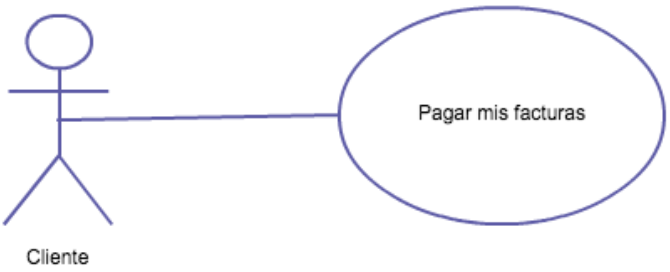


Ilustración 149. –Caso de uso Pagar mis facturas

Nombre: Caso de uso Pagar mis facturas

Descripción: El cliente podrá ver los datos de las reparaciones que aún le faltan por abonar las facturas correspondientes.

Actores: Cliente.

Precondiciones: Estar identificado como cliente.

Requisitos no funcionales: Ninguno.

Flujo de eventos:

1) El cliente, que previamente ha introducido la matrícula y la contraseña en la página "Entorno de clientes" (Ilustración 150), pulsará el botón "Pagar mis facturas" (Ilustración 151).

2) Aparecerá una nueva interfaz (Ilustración 152) en la que solamente aparecerán los datos de las reparaciones de las cuales su factura todavía no ha sido abonada, y en la que el cliente podrá realizar las siguientes acciones:

    2.1.) El cliente podrá ver la factura de cada reparación pulsando el botón "Ver factura" (Ilustración 153), habiendo seleccionado alguna reparación previamente. Se abrirá una nueva pestaña mostrando la factura en formato pdf (Ilustración 154), en la que se podrá visualizar correctamente y descargarla si así lo desea. En caso de que la factura esté pagada, aparecerá la factura con un sello de pagado (Ilustración 155).

    2.2.) Si el cliente desea no realizar ninguna acción deberá pulsar el botón "Volver

atrás" que le devolverá a la página de entorno de clientes (Ilustración 151).

2.3.) El cliente podrá pagar la factura de la reparación que desee, siempre y cuando la factura no haya sido abonada previamente y la reparación esté finalizada. Para ello deberá seleccionar una reparación y pulsar el botón “Pagar ahora” (Ilustración 156). Aparecerá una nueva interfaz (Ilustración 157) en la que se podrá visualizar la factura al momento, o si lo desea descargarla, pulsando el botón “Descargar”. Se deberá seleccionar con que método de pago se desea realizar el abono de la factura, siendo las dos opciones “Transferencia bancaria” o “Paypal”.

2.3.1) Si selecciona la opción “Transferencia bancaria” aparecerá una nueva interfaz de agradecimiento (Ilustración 158) en la que aparecerán los datos de la fecha, nombre y apellidos, importe de la factura y matrícula del vehículo. También hay un botón que permitirá al cliente descargarse la factura. Por último, aparecerán los datos bancarios para que el cliente pueda realizar la transferencia. La aplicación también le enviará al cliente un email con todos los datos anteriores detallados en él. Si el cliente desea volver al historial de reparaciones deberá pulsar “Volver atrás”.

2.3.2) Si selecciona la opción “Paypal” se le reconducirá automáticamente a la plataforma de paypal (Ilustración 159) para proceder a realizar el pago a través de la misma. Una vez terminado el pago se le reconducirá a una página de agradecimiento de la aplicación (Ilustración 160). El cliente recibirá un email con los datos detallados de su pago.

2.3.3) Si el cliente quisiera volver al menú sin realizar ninguna acción deberá pulsar el botón “Volver atrás”.

Pos condiciones: El cliente podrá ver las facturas pendientes de pago de las reparaciones realizadas en su vehículo y abonar el importe de las mismas.

Interfaz gráfica:

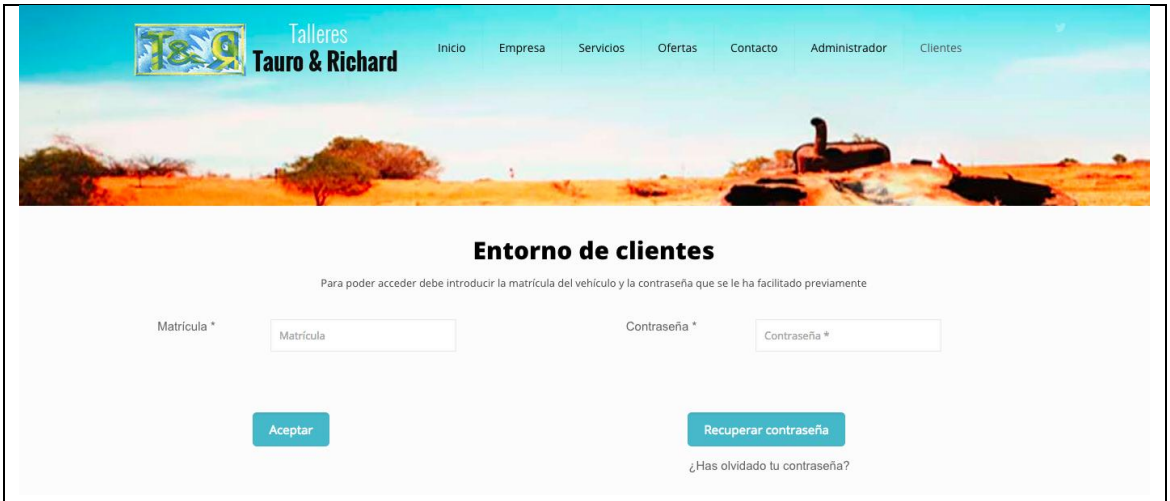


Ilustración 150. - Entorno de clientes

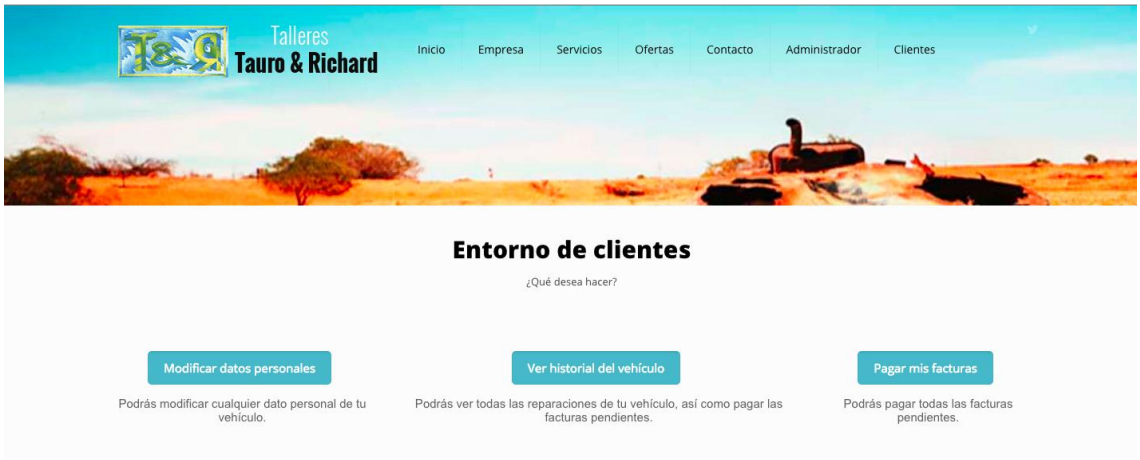


Ilustración 151. - Menú de clientes

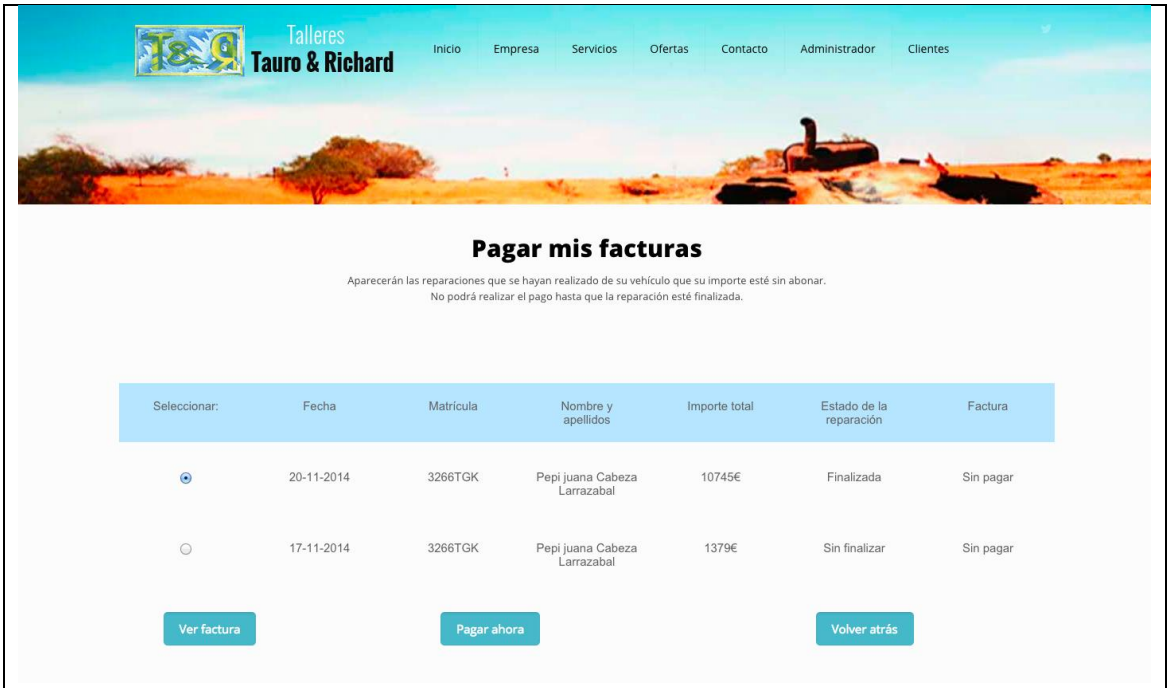


Ilustración 152. – Pagar mis facturas

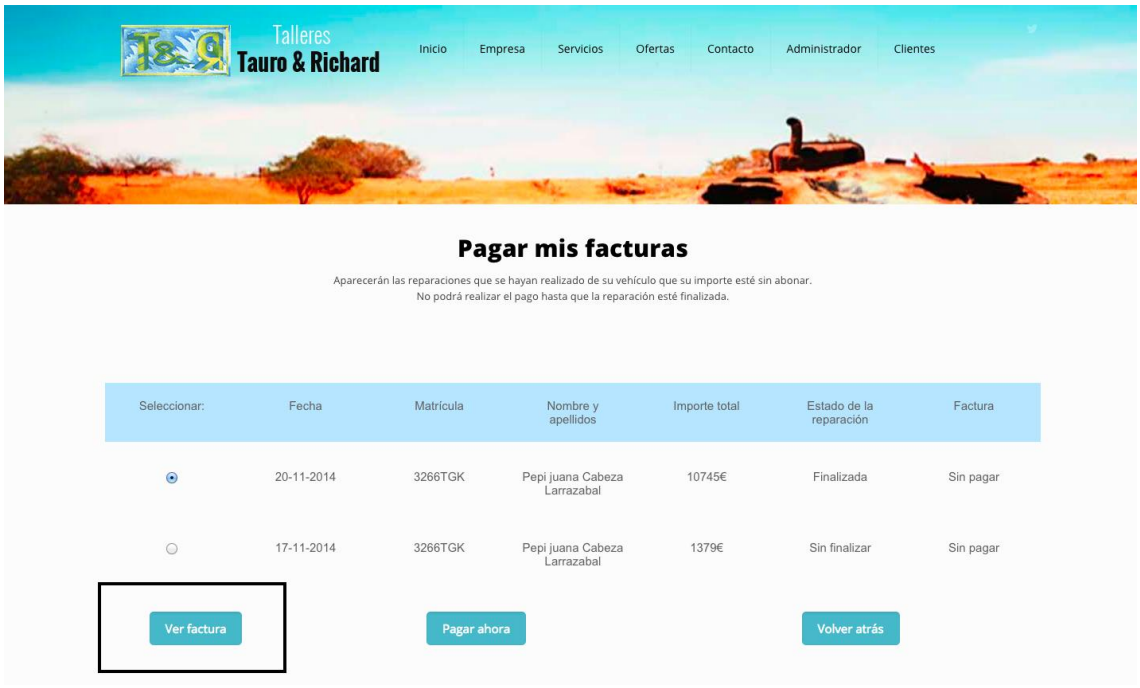


Ilustración 153. - Ver factura

Talleres Tauro & Richard

FACTURA

CIF: E - 48620198
Polígono Aperribai Auzoa S/N, PABELLÓN 6, Galdakao 48960, Vizcaya - España
94 449 25 77 / 667557962 / 610424613
Email: talleres@tauroyrichard.com

Número de factura: 39
Fecha: 18-11-2014

DATOS DEL CLIENTE

Nombre y apellidos: Pepita griffindor
Dirección: Benavente 2, bajo , 43990 , CUENCA - España
DNI: 45879906E
Teléfono: 943336789
Email: yasmina@notebuk.com

DATOS DEL VEHÍCULO

Matrícula: 9098WQE
Modelo: Kia Sorento
CIA: Linea directa

Concepto	Precio Unitario	Importe
4 x Ruedas	100	400
2 x Escobillas	20	40
Arreglo de puerta trasera	1200	1200

Concepto mano de obra	Importe

Base imponible	IVA	Cuota de IVA	Importe total
1640€	21%	344€	1984€

FIRMA DEL CLIENTE

FIRMA Y SELLO DE LA EMPRESA

Ilustración 154. - Factura de la reparación sin pagar

Ilustración 155. - Factura de la reparación pagada

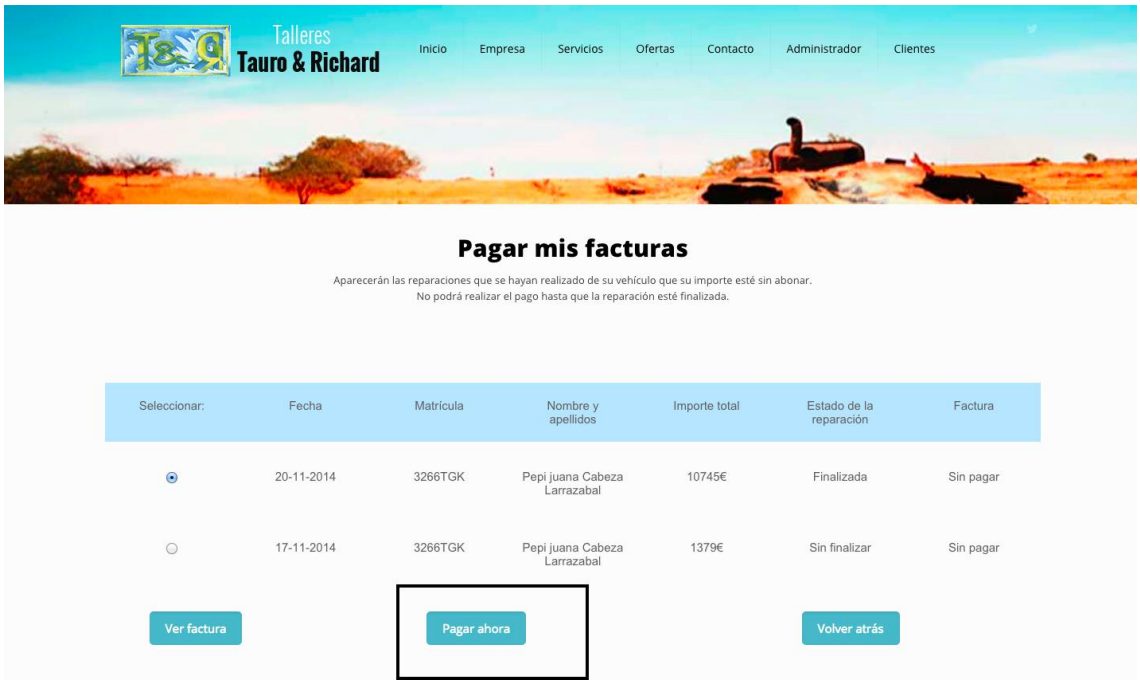


Ilustración 156. - Pagar factura de la reparación

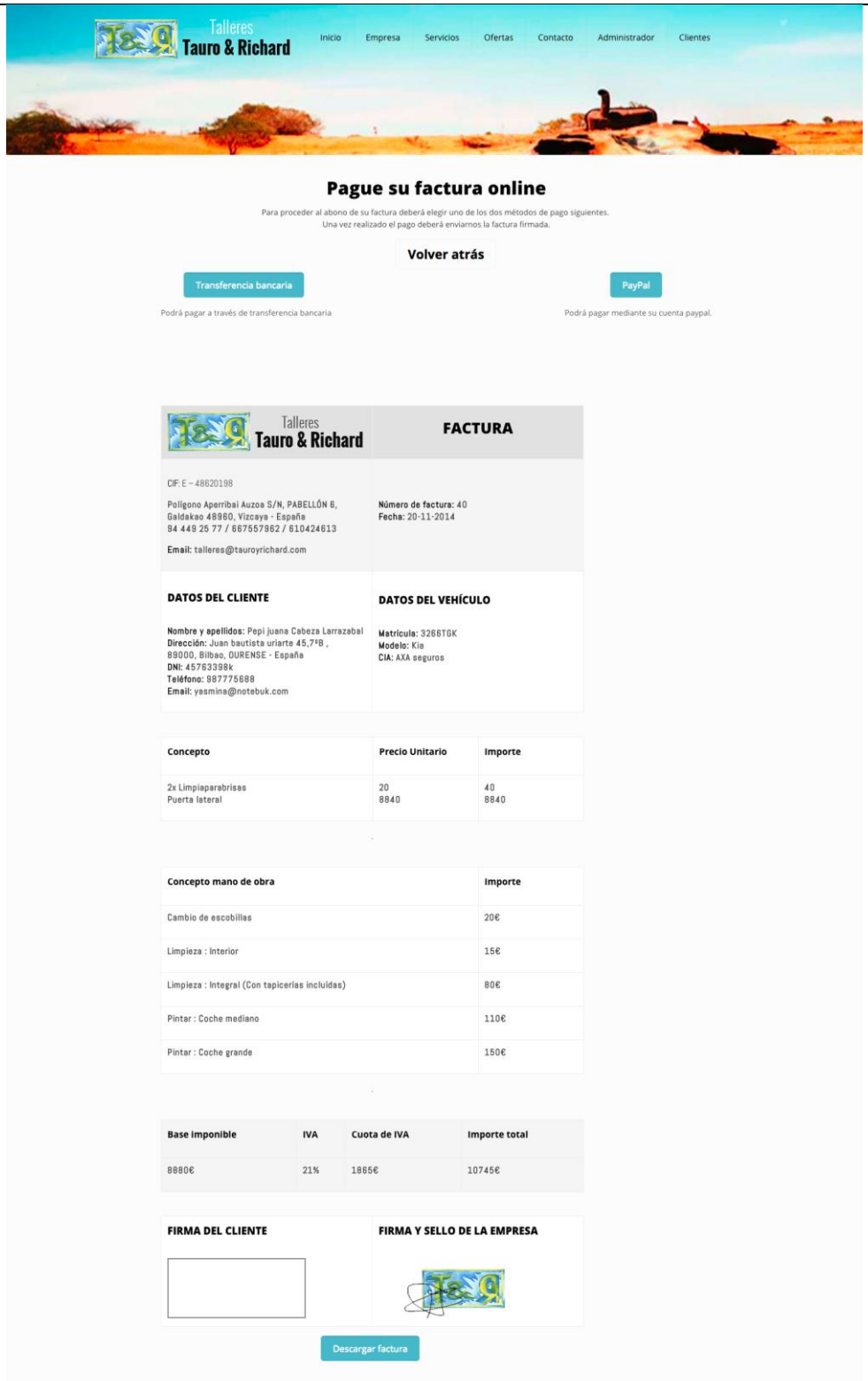


Ilustración 157. - Pagar y ver la factura

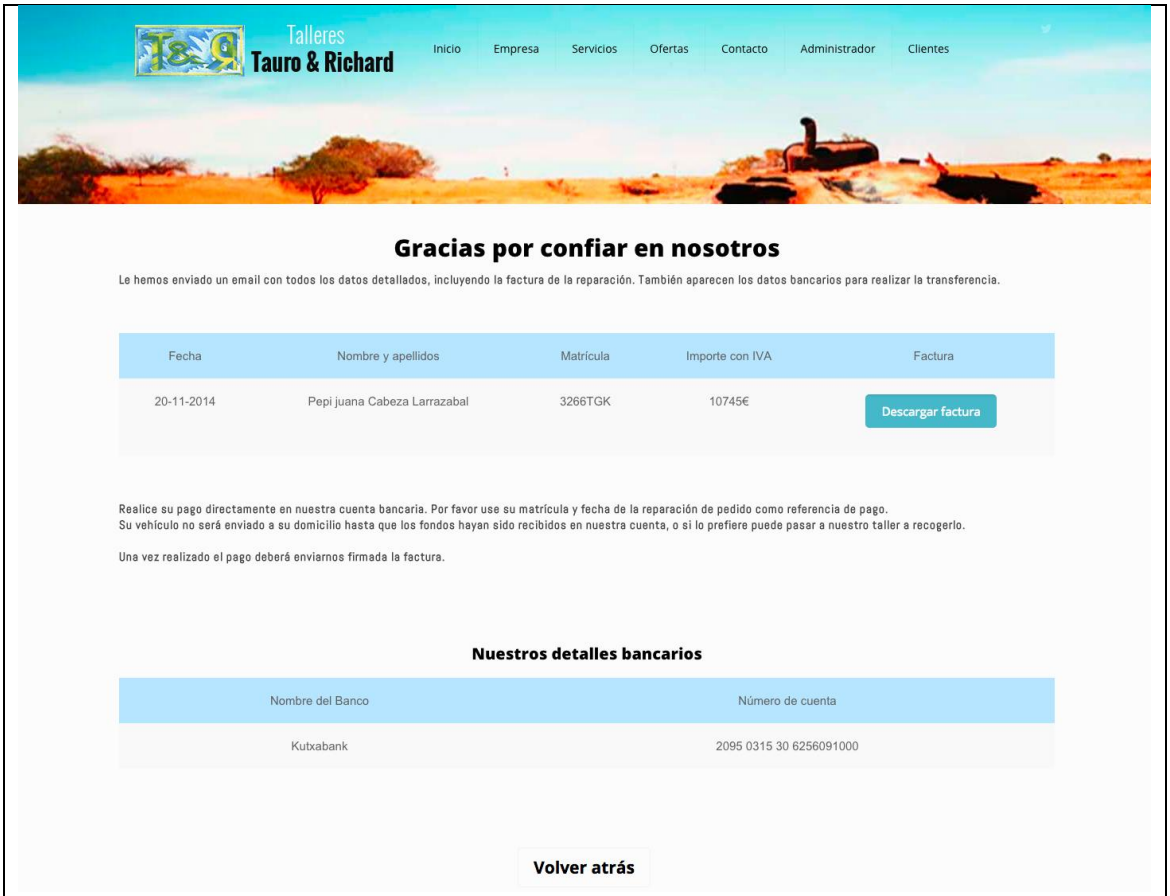


Ilustración 158. - Pago con transferencia bancaria

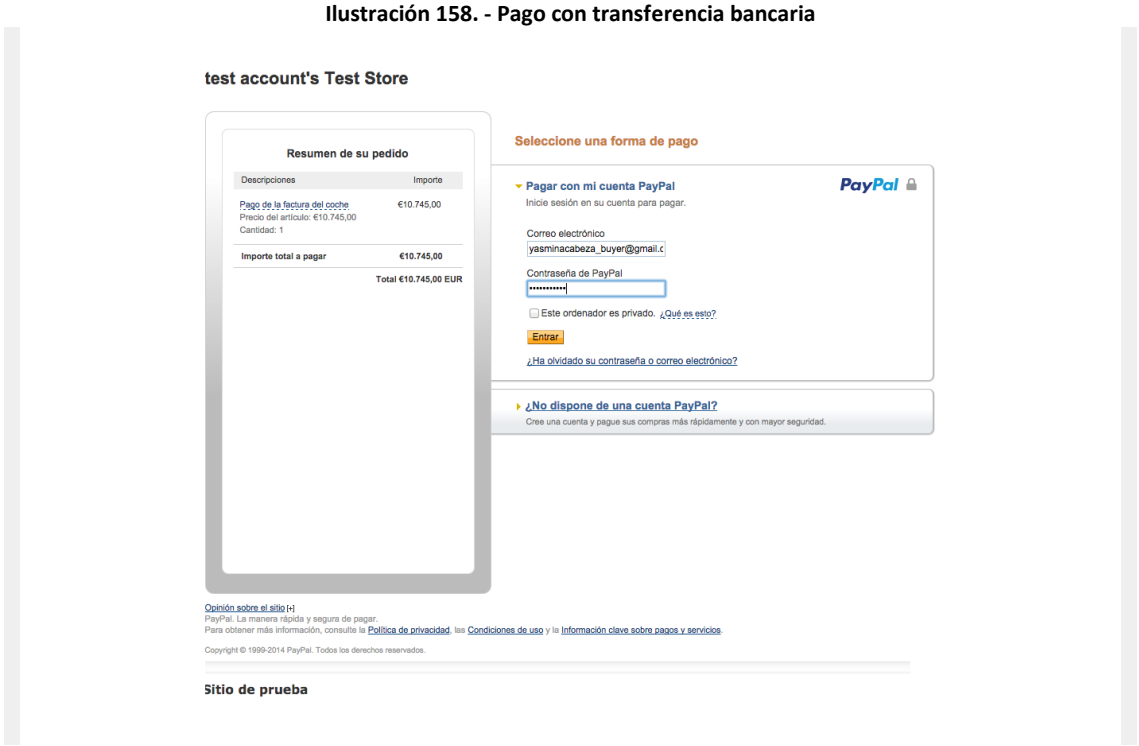


Ilustración 159. - Pago con Paypal

Ilustración 160. - Página de agradecimiento

10.2. Aplicación móvil

En esta sección se van a detallar los casos de uso extendidos de la aplicación móvil.

10.2.1. Caso de uso **identificación**

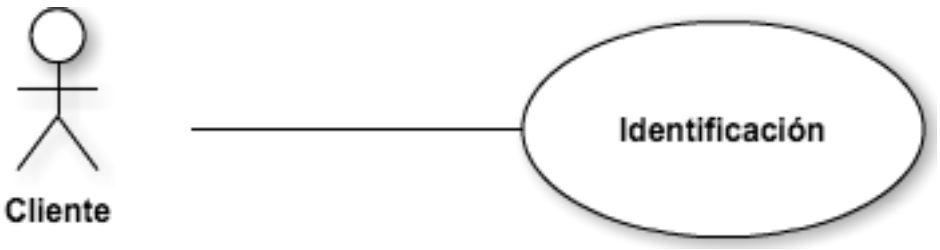


Ilustración 161. - Caso de uso Identificación

Nombre: Caso de uso Identificación
Descripción: El cliente desde la App Tauro & Richard deberá introducir su DNI y contraseña y podrá acceder a la interfaz Menú del cliente.
Actores: Cliente.
Precondiciones: Ninguna.
Requisitos no funcionales: Ninguno.
Flujo de eventos:

1) Aparecerá la interfaz (Ilustración 162) en la que aparecerán el campo DNI y Contraseña y deberá rellenarse y pulsar “Aceptar” y se comprobará con la base de datos si la contraseña es correcta y se le conducirá a otra interfaz (Ilustración 163).

2) En caso de que la contraseña sea incorrecta, se mostrará un mensaje de error (Ilustración 164).

3) En caso de que no se rellene alguno de los campos, se mostrará un mensaje de error (Ilustración 165).

Pos condiciones: Se habrá identificado el cliente y podrá realizar cualquiera de las operaciones que la App permite.

Interfaz gráfica:

Ilustración 162. – Identificación clientes

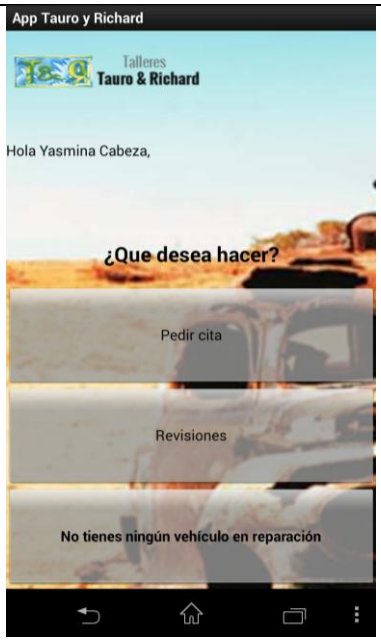


Ilustración 163. - Menú cliente

Ilustración 164. – Error por datos incorrectos

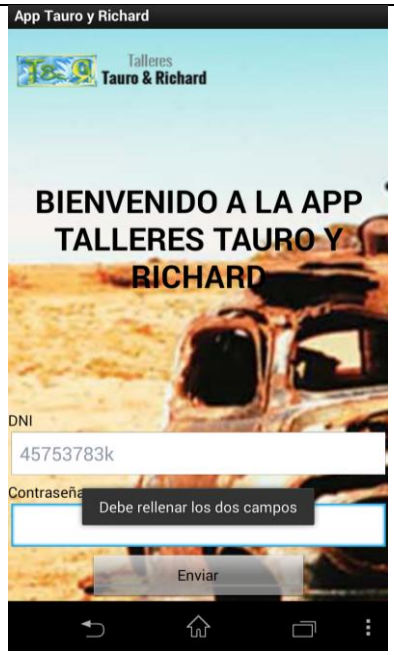


Ilustración 165. – Error por campos vacíos

10.2.2. Caso de uso **Pedir cita**

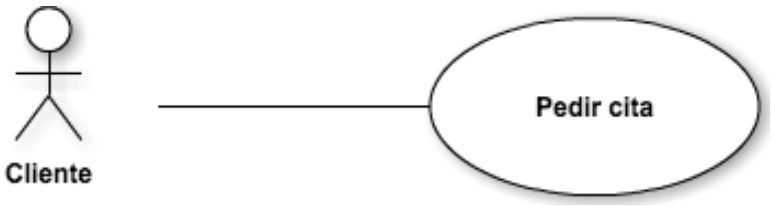


Ilustración 166. - Caso de uso Pedir Cita

Nombre: Caso de uso Pedir cita
Descripción: El cliente desde la App Tauro & Richard podrá pedir cita, eligiendo el día en el calendario y mandando el email a la empresa.
Actores: Cliente.
Precondiciones: Estar identificado en la App.
Requisitos no funcionales: Ninguno.
Flujo de eventos: 1) Aparecerá la interfaz de menú de la App (Ilustración 167) en la que aparecerán tres botones. Si el cliente pulsa “Pedir cita” se le reconducirá a otra interfaz(Ilustración 168). 2) En la interfaz de pedir cita, aparecerán dos campos a rellenar. Uno de ellos será el del día de la cita y el otro es un área de texto para escribir el mensaje que se quiera. Para elegir el día de la cita se debe pulsar el botón “Mostrar calendario” y aparecerá una ventana emergente con un calendario(Ilustración 169) para poder elegir el día de la cita. 3) Una vez rellenos los dos campos se debe pulsar el botón “Enviar” y aparecerá un mensaje de que el mensaje se ha enviado correctamente (Ilustración 170) 4) En caso de que no se rellene alguno de los campos, se mostrará un mensaje de error(Ilustración 171). 5) Si se quiere volver a la interfaz de menú de la App sin pedir cita, solo se debe pulsar el botón “Volver” y se volverá a la interfaz anterior(Ilustración 167).
Pos condiciones: El cliente habrá podido pedir cita previa a través de la App para acudir al taller de reparación.

Interfaz gráfica:

Ilustración 167. - Menú cliente

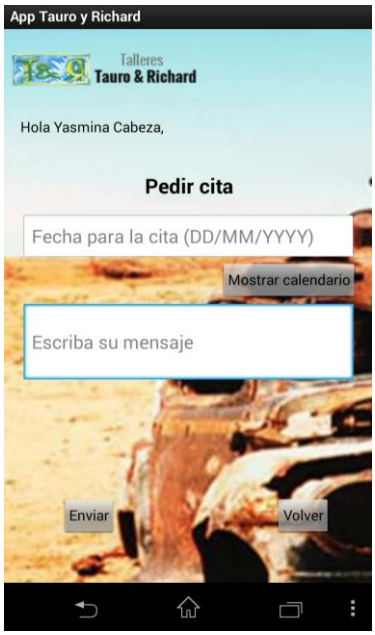


Ilustración 168. - Pedir cita

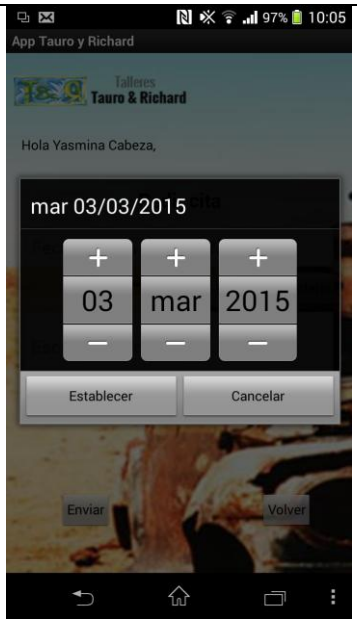


Ilustración 169. - Calendario

Ilustración 170. - Mensaje de éxito

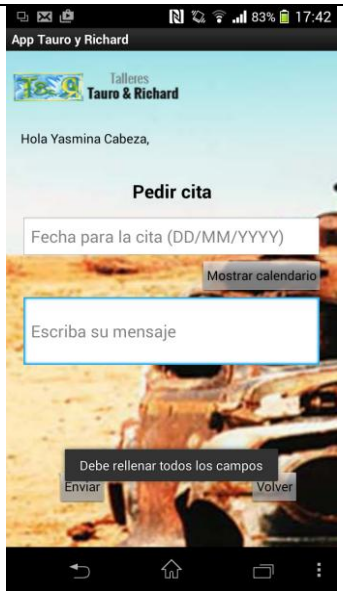


Ilustración 171. - Mensaje de error

10.2.3. Caso de uso **Consultar revisiones**

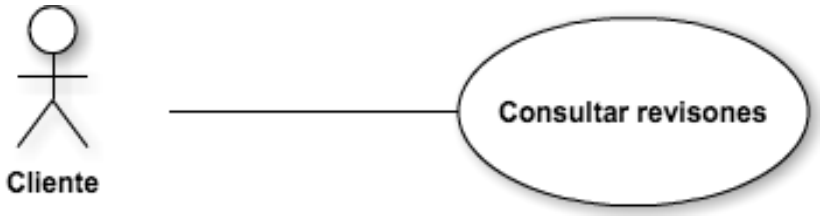


Ilustración 172. - Caso de uso Consultar revisiones

Nombre: Caso de uso Consultar revisiones
Descripción: El cliente desde la App Tauro & Richard podrá consultar las próximas revisiones que debe hacer del vehículo.
Actores: Cliente.
Precondiciones: Estar identificado en la App.
Requisitos no funcionales: Ninguno.
Flujo de eventos: 1) Aparecerá la interfaz de menú de la App (Ilustración 173) en la que aparecerán tres botones. Si el cliente pulsa “Revisiones” se le reconducirá a otra interfaz (Ilustración 174). 2) En la interfaz de Revisiones, aparecerá una tabla con todas las revisiones que tiene que realizarle al coche, ya sea la fecha de revisión o el número de kilómetros con los que se tendrá que realizar dicha revisión. 3) En caso de que quiera volver al menú de la App, solo debe pulsar el botón “Volver” (Ilustración 174).
Pos condiciones: El cliente habrá podido visualizar la fecha o kilometraje de las próximas revisiones necesarias de su vehículo.
Interfaz gráfica:

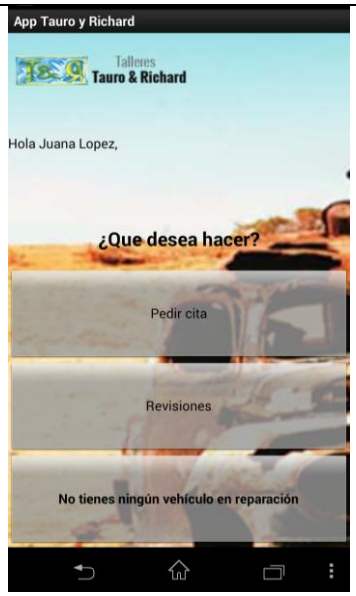


Ilustración 173. - Menú cliente

Ilustración 174. - Revisiones

10.2.4. Caso de uso **Ver nuevo aviso**

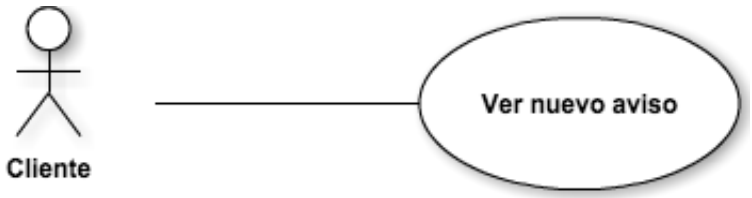


Ilustración 175. - Caso de uso Consultar revisiones

Nombre: Caso de uso Ver nuevo aviso

Descripción: El cliente, que previamente se haya identificado al menos una vez en la aplicación, podrá recibir notificaciones del taller Tauro & Richard directamente en su móvil.

Actores: Cliente.

Precondiciones: Haberse identificado al menos una vez en la App.

Requisitos no funcionales: Ninguno.

Flujo de eventos:

4) Aparecerá una nueva notificación en el dispositivo móvil del cliente que indicará “La reparación de tu vehículo está finalizada” (Ilustración 176).

5) Si pulsa dicha notificación se abrirá directamente la App, sin necesidad de volver a identificarse, y aparecerá el menú de la App (Ilustración 177) con el último botón iluminado y con el siguiente mensaje ”La reparación de su vehículo ha sido finalizada. La factura la puede abonar a través de nuestra página web”.

Pos condiciones: El cliente habrá recibido la notificación de finalización de la reparación de su vehículo, enviada por el administrador de la aplicación web.

Interfaz gráfica:

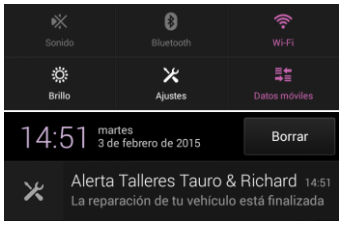


Ilustración 176. - Alerta notificación

Ilustración 177. - Aviso de finalización de la reparación

11. ANEXO II. – DIAGRAMAS DE SECUENCIA

Los objetos del sistema interaccionan entre ellos mediante la llamada de operaciones a otros objetos.

En este apartado se van a definir los diagramas de secuencia, tanto de la aplicación móvil como de la aplicación web. Estos diagramas permiten mostrar las acciones que se producen entre los actores y el sistema, permitiendo reconocer las operaciones del mismo. Igualmente, permiten visualizar los efectos que pueden conllevar dichas operaciones, desde cambios en la base de datos hasta las salidas que produce el sistema cuando se invocan dichas operaciones.

Es importante mencionar que el diagrama muestra los mensajes en orden cronológico, a medida que van surgiendo, desde arriba hacia abajo, a lo largo de unas líneas discontinuas verticales, las cuales están asociadas a los objetos que forman el diagrama.

11.1. Aplicación web

A continuación, se detallan cada uno de los diagramas de secuencia de la aplicación web junto con las explicaciones correspondientes del diagrama, concernientes a cada caso de uso de la aplicación web, definidos en el apartado anterior.

11.1.1. Diagrama de secuencia **Identificación Admin**

1. El administrador introduce la contraseña requerida y pulsa “Aceptar”.

2. Obligatorios(): Boolean

[Si la obligatorios() devuelve true]

3a. identificar(); **new ajax.updater('espacio', 'identificarse.php',{ asynchronous:false, postbody: 'pwd='+val1});**
4a. execSQL(sql1)
sql1: SELECT indice FROM tx_administradortauro WHERE password='".$pwd."'
(Comprueba que la contraseña sea correcta)

5a. new()

6a. next()

[Si hay tuplas]

7aa. getInt(Indice)

8a.close()

[Si la consulta devuelve el valor índice]

9aa. panel_administracion()

10aa. New() (Irá a esta url http://proyecto.tauroyrichard.com/panel-administracion/)

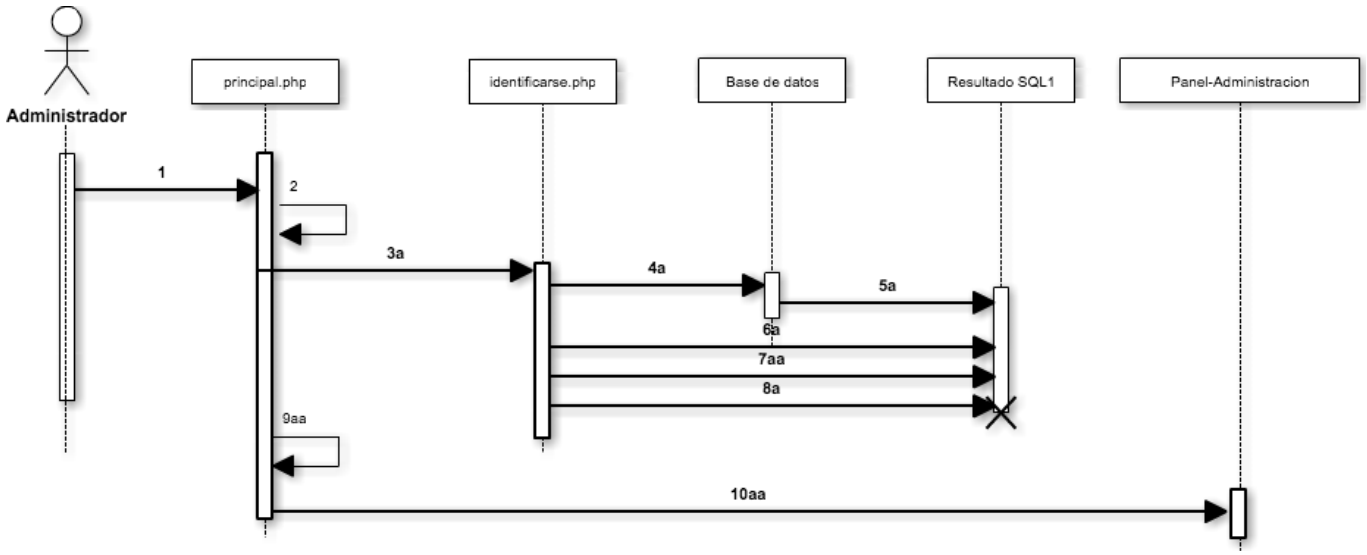


Ilustración 178. - Diagrama de caso de uso de Identificación Admin

11.1.2. Diagrama de secuencia **Introducir nueva reparación**

1.El administrador pulsa "Introducir nuevo vehículo"

2.formulario_nueva_reparacion() (Irá a esta url http://proyecto.tauroyrichard.com/formulario-reparacion-nueva/)

3.execSQL(sql1)

sql1: "Select matricula from tx_vehiculo"(Cargar todas las matrículas)

4. new()

5. next()

[Si hay tuplas]

6a. getString ("matricula"):String;

7.close()

8.El administrador selecciona un elemento del desplegable

[Si ha seleccionado “Insertar nuevo vehículo” se abrirá un nuevo formulario]

9a. Insertarnuevo():void (Insertará el formulario correspondiente)
10a. execSQL(sql2) (Cargará todas las provincias en el formulario)
sql2: "Select * from tx_provincias"
11a.new()
12a. next()
[Si hay tuplas]
13aa. getInt("cod_prov"): Int;
14aa. getString("nombre"): String;
15a.close()
16a. execSQL(sql3) (Cargar lo servicios de mano de obra)
Sql3: "Select codserv,titulo,descripcion from tx_servicios ORDER BY titulo ASC"
17a.new()
18a. next()
[Si hay tuplas]
19aa. getInt("codserv "): Int;
20aa. getString("titulo"): String;
21aa. getString("descripcion"): String;
22a.close()
23a. El administrador rellena todos los campos del formulario y pulsa “Aceptar”

24a. obligatorios():boolean
[Si obligatorios es true]
25aa. validaremail():boolean
[Si validar email es true]
26aaa. validartfno():boolean
[Si validar tfno es true]
27aaaa. insertarbd():submit
28aaaa. execSQL(sql4) (Obtener el código de la provincia)
Sql4: "Select cod_prov from tx_provincias where nombre='".$provincia."' "
29aaaa.new()
30aaaa. next()
[Si hay tuplas]
31aaaaa. getInt("cod_prov "): Int;
32aaaa.close()
33aaaa. execSQL(sql5) (Insertamos los datos del vehiculo en la tabla tx_vehiculo)
sql5:"INSERT INTO tx_vehiculo (modelo_vehic, nombre, matrícula, DNI,telefono, password,email,direccion,codigo_postal,cod_prov,ciudad,km)VALUES ('$modelo','$nombre','$matricula','$dni','$telefono','$pwd','$email', '$direccion','$codigo','$provincia', '$ciudad','$km')"
34aaaa.execSQL(sql6) (Insertamos los datos de la reparación en la tabla tx_reparacion)
sql6:"INSERT INTO tx_reparacion (matrícula, base_imp, cuota_iva,importe_total, concepto,precio_unitario,importe_concepto,fecha,estado_rep,factura)VALUES ('$matricula','$baseimponible','$cuotaiva','$importetotal','$concepto','$preciouni', '$importe_concepto','$fecha','$estado','$factura')"
35aaaa. execSQL(sql7) (Obtengo el código de la reparación)
sql7: "select cod_rep from tx_reparacion where matricula='".$matricula."' and fecha= '".$fecha."' "
36aaaa.new()
37aaaa. next()
[Si hay tuplas]
38aaaaa. getInt("cod_rep "): Int;
39aaaa.close()
40aaaa. execSQL(sql8) (Guardo los servicios de mano de obra de la reparación en la tabla tx_actuaciones)
sql8: "INSERT INTO tx_actuaciones (codserv,cod_rep) VALUES ('$manode', '$cod_rep')"
41aaaa. Mail() (Enviará un email al cliente)

[Si el administrador ha seleccionado una matrícula]

9b. Insertarnuevo():void (Cargará un formulario con los datos del vehículo) N**ew Ajax.Updater('formulariomatriculaexistente1','vehiculo-existente.php',{ asynchronous : false, postBody: 'matric='+valor });**

10b. execSQL(sql9) (Cargar todos los datos personales asociados al vehículo de la tabla tx_vehiculo)

sql9: "select * from tx_vehiculo where matricula='".$matriculaexist."' "

11b.new()

12b. next()

[Si hay tuplas]

13ba. getString("nombre"): String;

14ba. getString("matricula"):String

15ba. getString("DNI"):String

16ba. getString("modelo_vehic"):String

17ba. getInt("telefono"): Int;

18ba. getString("email"):String

19ba. getString("direccion"):String

20ba. getString("ciudad"):String

21ba. getInt("código_postal"): Int;

22ba. getInt("km"): Int;

23ba. getInt("cod_prov"): Int;

24b. close()

25b. execSQL(sq10) (Cargar las provincias)

sql10: 'select cod_prov,nombre from tx_provincias ORDER BY nombre ASC'

26b. new()

27b next()

[Si hay tuplas]

28ba. getInt("cod_prov"): Int;

29ba. getString("nombre"): String;

30b.close()

31b. execSQL(sql11) (Cargar servicios de mano de obra)

sql11: "select codserv,titulo,descripcion from tx_servicios ORDER BY titulo ASC"

32b.new()

33b. next()

[Si hay tuplas]

34ba. getInt("codserv "): Int;

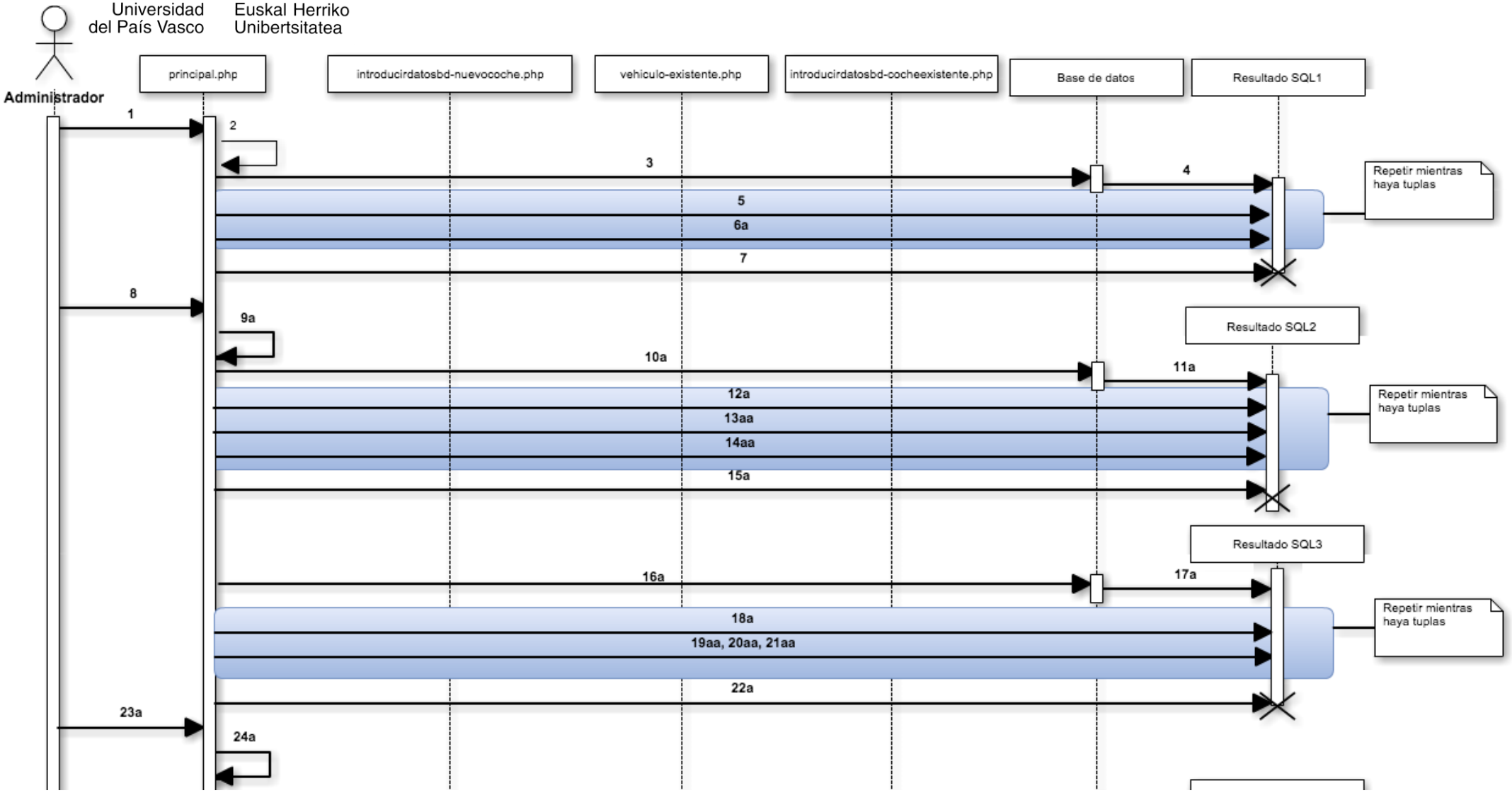
35ba. getString("titulo"): String;

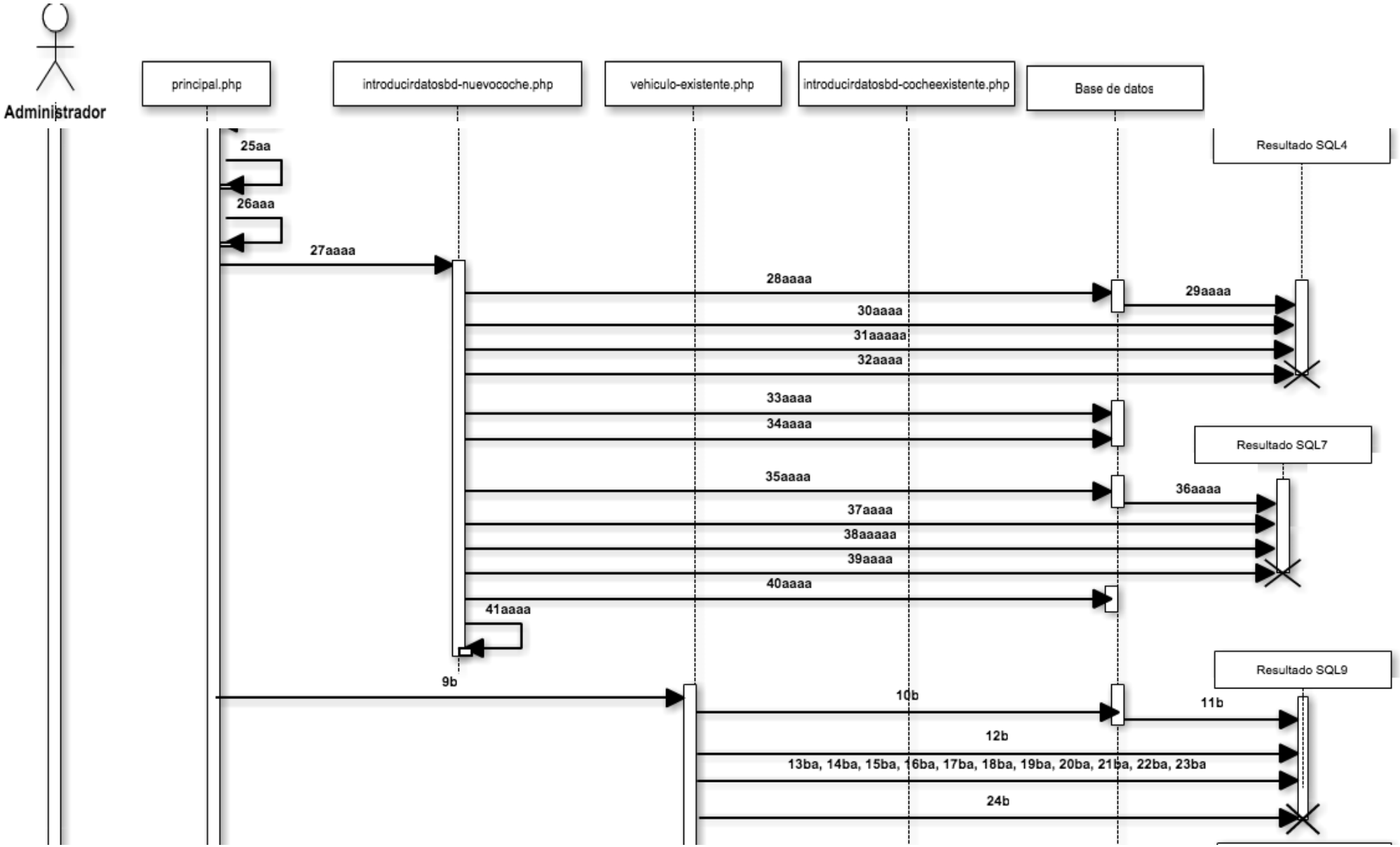
36ba. getString("descripcion"): String;

37b.close()

38b. El administrador rellena todos los campos del formulario y pulsa “Aceptar”

39b. Obligatoriosexist():boolean
[Si obligatoriosexist() es true]
40ba. validaremailexist():boolean
[Si validar email es true]
41baa. Insertarbdexistente(): Submit
42baa. execSQL(sql12) (Sacamos el código de la provincia)
sql12:"Select cod_prov from tx_provincias where nombre='".$provincia."' "
43baa. new()
44baa. next()
[Si hay tuplas]
45baaa. getInt("cod_prov "): Int;
46baa.close()
47baa. execSQL(sql13) (Actualizamos los datos de la tabla tx_vehiculo)
sql13: "UPDATE tx_vehiculo SET nombre = '".$nombre."', DNI = '".$dni."' , modelo_vehic = '".$modelo."', telefono = '".$telefono."', email = '".$email."',direccion = '".$direccion."', ciudad = '".$ciudad."' , codigo_postal = '".$codigo."' ,cod_prov ='".$provincia."', km = '".$km."' WHERE matricula ='".$matricula."' "
48baa. execSQL(sql14) (Insertamos los datos de la reparación en tx_reparación)
sql14: "INSERT INTO tx_reparacion (matricula,base_imp, cuota_iva,importe_total, concepto,precio_unitario,importe_concepto,fecha,estado_rep,factura) VALUES ('$matricula','$baseimponible','$cuotaiva','$importetotal','$concepto','$preciouni','$importe_concepto','$fecha','$estado','$factura')"
49baa. execSQL(sql15) Select SQL (Sacamos el código de la reparacion)
sql15: "select cod_rep from tx_reparacion where matricula='".$matricula."' and fecha= '".$fecha."'
50baa. new()
51baa. next()
[Si hay tuplas]
52baaa. getInt("cod_rep "): Int;
53baa. close()
54baa. execSQL(sql16) (Insertamos los servicios de mano de obra de la reparación en la tabla tx_actuaciones)
Sql16: "INSERT INTO tx_actuaciones (codserv,cod_rep) VALUES ('$manode' ,'$cod_rep')"
55baa. New () (Irá a esta url http://proyecto.tauroyrichard.com/panel-administracion)


Administrador
principal.php
introducirdatosbd-nuevococche.php
vehiculo-existente.php
introducirdatosbd-cocheexistente.php
Base de datos
Resultado SQL1
Resultado SQL2
Resultado SQL3
Repetir mientras haya tuplas
Repetir mientras haya tuplas
Repetir mientras haya tuplas
1
2
3
4
5
6a
7
8
9a
10a
11a
12a
13aa
14aa
15a
16a
17a
18a
19aa, 20aa, 21aa
22a
23a
24a


Administrador
principal.php
introducirdatosbd-nuevococche.php
vehiculo-existente.php
introducirdatosbd-cocheexistente.php
Base de datos
Resultado SQL4
Resultado SQL7
Resultado SQL9
25aa
26aaa
27aaaa
28aaaa
29aaaa
30aaaa
31aaaaa
32aaaa
33aaaa
34aaaa
35aaaa
36aaaa
37aaaa
38aaaaa
39aaaa
40aaaa
41aaaa
9b
10b
11b
12b
13ba, 14ba, 15ba, 16ba, 17ba, 18ba, 19ba, 20ba, 21ba, 22ba, 23ba
24b

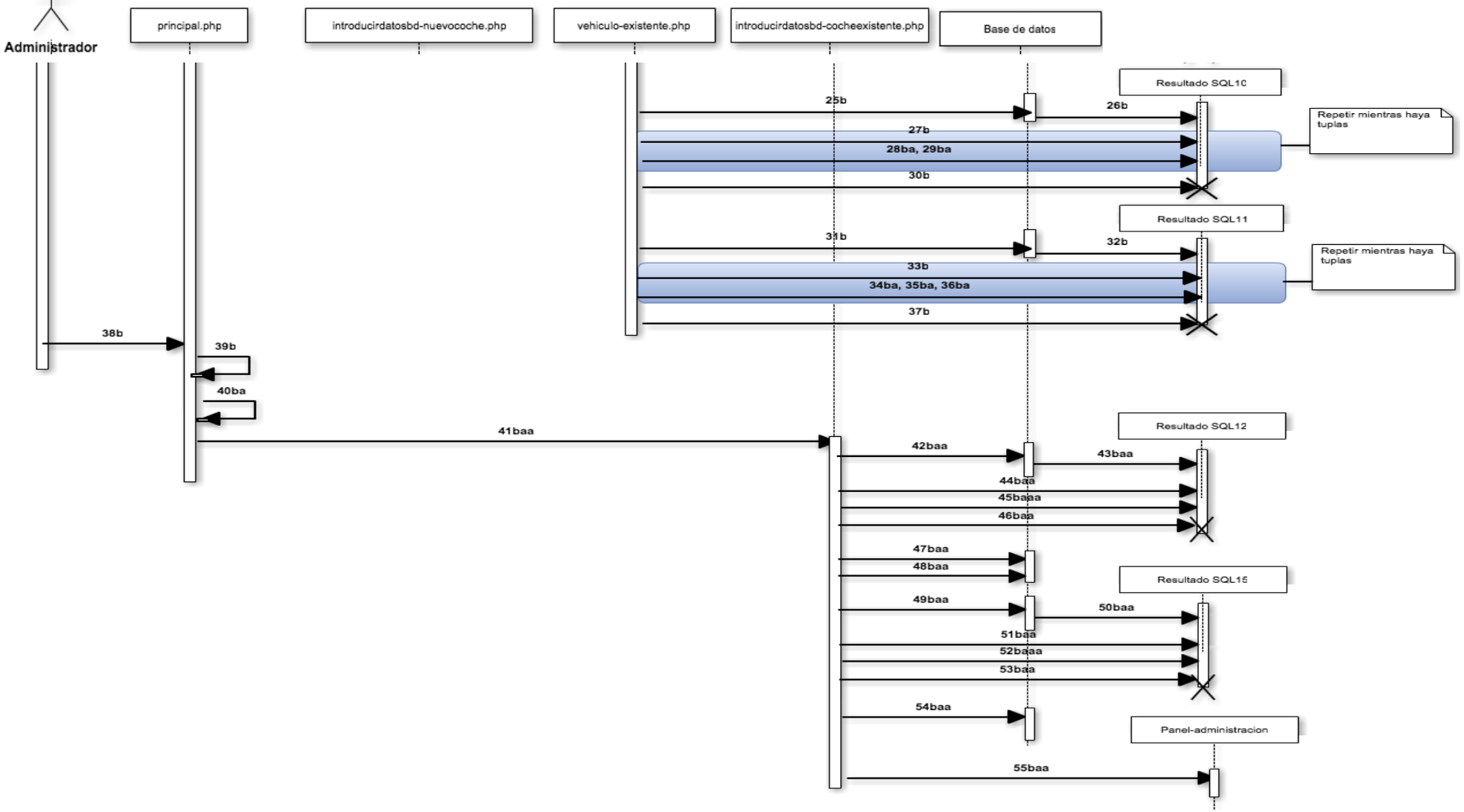


Ilustración 179. - Diagrama de secuencia de caso de uso Introducir nueva reparación

11.1.3. Diagrama de secuencia **Modificar reparación de vehículo**

1. El administrador pulsa “Modificar reparación de vehículo”.

2.Formulario_modificar_reparacion()(Irá a esta url http://proyecto.tauroyrichard.com /formulario-actualizacion-de-campos/)

3. ExecSQL(sql1) (Cargar datos de las matriculas cuyos vehículos tengan reparaciones en el taller)

sql1: select DISTINCT tx_vehiculo.matricula from tx_vehiculo inner join tx_reparacion on tx_vehiculo.matricula=tx_reparacion.matricula "

4. new()

5. next()

[Si hay tuplas]

6a. getString ("matricula"):String;

7.close()

8.El administrador selecciona una matrícula de un vehículo

9.Buscarmatricula():void new Ajax.Updater('verfecha', 'obtfecha.php',{ asynchronous:false, postBody: 'matric='+valor });

10.execSQL(sql2) (Obtiene las fechas de cada reparacion)

sql2: "select fecha,cod_rep from tx_reparacion where matricula='".$matricula."' "

11. new()

12. next()

[Si hay tuplas]

13a. getDate ("fecha"):Date;

14a. getInt("cod_rep"):Int

15.close()

16.El administrador selecciona una fecha de reparación

17.Buscarfecha():void (Se insertará un formulario con todos los datos cargados) new Ajax.Updater('formulariomodificar','formulario_modificardatos.php', {asynchronous:false, postBody: 'fecha='+valfecha+'&matric='+valmatric });

18.execSQL(sql3) (Carga los datos del vehiculo con esa matrícula de la tabla tx_vehiculo)

sql3: "select * from tx_vehiculo where matricula='".$matriculaexist."' "

19. new()

20. next()

[Si hay tuplas]

21a. getString("nombre"): String;
22a. getString("matricula"):String
23a. getString("DNI"):String
24a. getString("modelo_vehic"):String
25a. getInt("telefono"): Int;
26a. getString("email"):String
27a. getString("direccion"):String
28a. getString("ciudad"):String
29a. getInt("código_postal"): Int;
30a. getInt("km"): Int;
31a. getInt("cod_prov"): Int;

32.close()

33. execSQL(sql4) (Carga los datos de la matrícula cogidos de la tabla tx_reparacion)

sql4: "select * from tx_reparacion where matricula='".$matriculaexist."' and fecha='".$fecha."' "

34. new()

35. next()

[Si hay tuplas]

36a. getDate ("fecha"):Date;
37a. getInt("cod_rep"): Int;
38a. getString("CIA"): String;
39a. getString("factura"):String
40a. getString("estado_rep"):String

41a. getString("concepto"):String
42a. getString("precio_unitario")String
43a. getString("importe_concepto"):String
44a. getInt("importe_total"): Int;
45a. getInt("base_imp"): Int;
46a. getInt("cuota_iva"): Int;

47.close()

48.SQL(sql5) (Carga los datos de las provincias)

sql5: 'select cod_prov,nombre from tx_provincias ORDER BY nombre ASC'

49.new()

50. next()

[Si hay tuplas]

51a. getInt("cod_prov "): Int;
52a.getString("nombre"): String;

53.close()

54.execSQL(sql6) (Selecciona los servicios asociados a esa reparación en la tabla tx_actuaciones)

sql6: "select codserv from tx_actuaciones where cod_rep='".$reparacion['cod_rep']."'"

55.new()

56. next()

[Si hay tuplas]

57a. getInt("codserv"): Int;

58.close()

59.execSQL(sql7) (Selecciona todos los datos de la tabla tx_servicios)

sql7: "select codserv,titulo,descripcion from tx_servicios ORDER BY titulo ASC"

60.new()

61. next()

[Si hay tuplas]

62a. getInt("codserv "): Int;

63a. getString("titulo"): String;
64a. getString("descripcion"): String;
65.close()

66.El administrador realiza todos los cambios que desee en el formulario y pulsa un botón

[Si el botón es “Borrar”]

67a.Reset() (El formulario volverá a tener los valores cargados al principio)
[Si el botón es “Aceptar”]

67b. obligatoriosmodif():boolean
[Si obligatoriosmodif() es true]
68ba. validaremail():boolean
[Si validar email es true]
69baa. modificarbd():Submit
70baa. Execsql(sql8)(Selecciona el código de la provincia seleccionada)
Sql8:Select cod_prov from tx_provincias where nombre='".$provincia."'
71baa. new()
72baa. next()
[Si hay tuplas]
73baaa. getInt("cod_prov"): Int;
74baa.close()
75baa. Execsql(sql9) (Actualiza la tabla tx_vehiculo)
Sql9: "UPDATE tx_vehiculo SET nombre = '".$nombre."', DNI = '".$dni."',
modelo_vehic = '".$modelo."',telefono = '".$telefono."',email = '".$email."',direccion
= '".$direccion."',ciudad = '".$ciudad."' ,codigo_postal = '".$codigo."' ,cod_prov
='".$provincia."',km = '".$km."' WHERE matricula ='".$matricula."' ";
76baa. Execsql(sql10) (Selecciona el código de esa reparacion)
Sql10: "select cod_rep from tx_reparacion WHERE matricula ='".$matricula."' and
fecha= '".$fecha."' "
77baa. new()
78baa. next()
[Si hay tuplas]
79baaa. getInt("cod_rep"): Int;
80baa.close()
81baa. Execsql(sql11) (Actualiza los campos de la reparación seleccionada)
Sql11: "UPDATE tx_reparacion SET CIA = '".$cia."', factura = '".$factura."' ,
estado_rep = '".$estado."',base_imp = '".$baseimponible."',cuota_iva =
'".$cuotaiva."',importe_total = '".$importetotal."',concepto = '".$concepto."'
,precio_unitario = '".$preciouni."' ,importe_concepto= '".$importe_concepto."'
WHERE cod_rep ='".$cod_rep."' ";

[Si el administrador ha puesto el estado de la reparación ="Finalizada" y factura = "Sin pagar"]

82baaa. Execsql(sql12) (Inserta un aviso en la tabla tx_avisos)

sql12: "INSERT INTO tx_avisos (matricula,fecha,codAviso) VALUES ('$matricula','$fechaa','5')";

83baaa. Execsql (sql13) (Selecciona el idGCM de la tabla tx_app)

sql13: "SELECT IdGCM from tx_app where DNI='$dni' ";

84baaa. new()

85baaa. next()

[Si hay tuplas]

86baaaa. getInt("idGCM"): Int;

87baaa.close()

88baaa. Mandar_notificacion_push(): void;

89baa. Execsql(sql14) (Selecciona los servicios que se han deseleccionado)

Sql14:select codserv from tx_actuaciones WHERE cod_rep ='".$cod_rep."'

90baa. new()

91baa. next()

[Si hay tuplas]

92baaa. getInt("codserv"): Int;

93baa.close()

94baa. Execsql(sql15)(Borra los servicios deseleccionados)

Sql15: DELETE FROM tx_actuaciones WHERE cod_rep ='".$cod_rep."' AND codserv= '".$actuaciones['codserv']."'

95baa. Execsql(sql16) (Selecciona los nuevos servicios seleccionados)

Sql16: select codserv from tx_actuaciones WHERE cod_rep ='".$cod_rep."'

96baa. new()

97baa. next()

[Si hay tuplas]

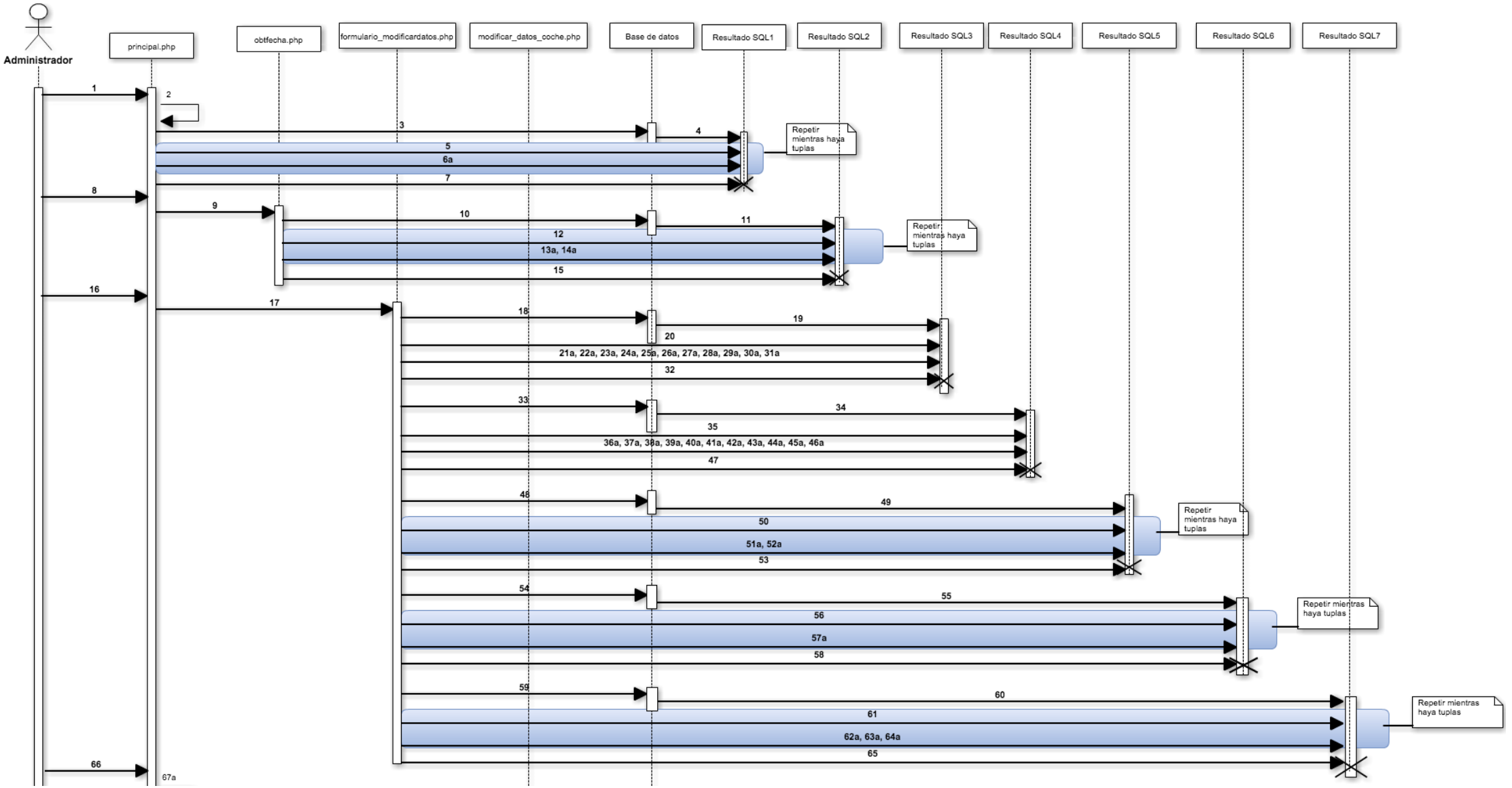
98baaa. getInt("codserv"): Int;

99baa.close()

100baa.Execsql(sql17)(Inserta los servicios seleccionados)

Sql17: INSERT INTO tx_actuaciones (codserv,cod_rep) VALUES ('$manode','$cod_rep')

101baa. New() (Irá a esta url http://proyecto.tauroyrichard.com/panel-administracion/)


Administrador
principal.php
obtfecha.php
formulario_modificardatos.php
modificar_datos_coche.php
Base de datos
Resultado SQL1
Resultado SQL2
Resultado SQL3
Resultado SQL4
Resultado SQL5
Resultado SQL6
Resultado SQL7
1
2
3
4
5
6a
7
Repetir mientras haya tuplas
8
9
10
11
12
13a, 14a
15
Repetir mientras haya tuplas
16
17
18
19
20
21a, 22a, 23a, 24a, 25a, 26a, 27a, 28a, 29a, 30a, 31a
32
33
34
35
36a, 37a, 38a, 39a, 40a, 41a, 42a, 43a, 44a, 45a, 46a
47
48
49
50
51a, 52a
53
Repetir mientras haya tuplas
54
55
56
57a
58
Repetir mientras haya tuplas
59
60
61
62a, 63a, 64a
65
Repetir mientras haya tuplas
66
67a

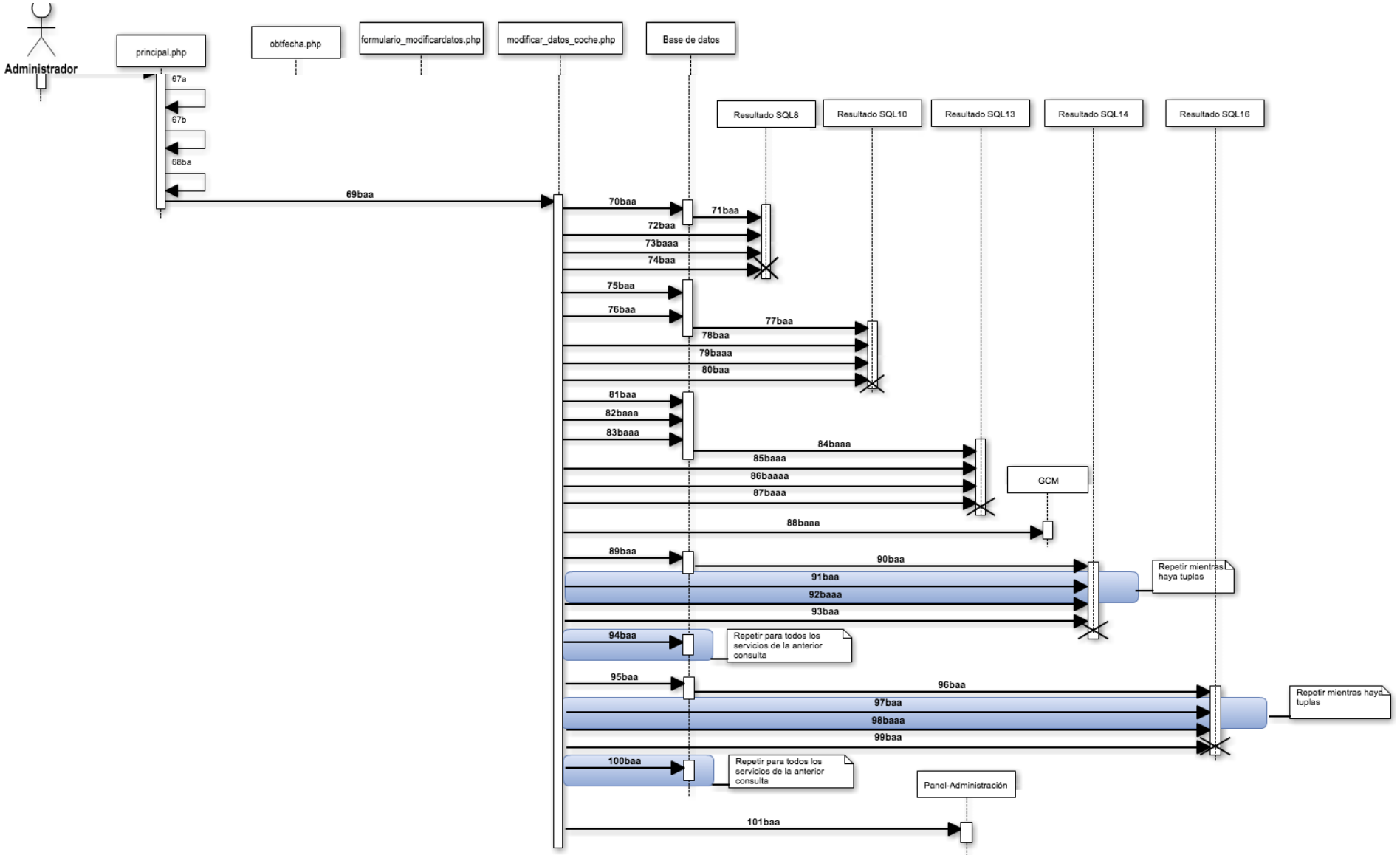


Ilustración 180. - Diagrama de secuencia de caso de uso Modificar reparación de vehículo

11.1.4. Diagrama de secuencia **Gestionar servicios mano de obra**

1.El administrador pulsa "Gestionar servicios mano de obra"

2. gestionservicios () (Irá a esta url http://proyecto.tauroyrichard.com/gestion-servicios/)

3.execSQL(sql1)

sql1: "select codserv,titulo,descripcion from tx_servicios ORDER BY titulo ASC"(Cargar todos los campos de los servicios)

4. new()

5. next()

[Si hay tuplas]

6a. getInt("codserv"):Int;
7a. getString ("titulo"):String;
8a. getString ("descripcion"):String;

9.close()

[Si el administrador pulsa Volver atrás]

10a. El administrador pulsa "Volver atrás"
11a. volveradministracion(): void;
12a. new() (Irá a esta URL http://proyecto.tauroyrichard.com/panel-administracion/)

[Si el administrador pulsa Insertar nuevo]

10b. El administrador pulsa "Insertar nuevo"
11b. insertarnuevo(): void (Aparecerá el formulario de insertar nuevo servicio)
12b. El administrador rellena los datos y pulsa insertar
13b. obligatoriosserv(): Boolean
[si obligatoriosserv es true]
14ba. insertarservicioenbd(): submit
15ba. execSQL(sql2)
sql2: INSERT INTO tx_servicios (titulo,descripcion,precio) VALUES ('$nomserv', '$descserv', '$precserv')
16ba. New() (Irá a esta url http://proyecto.tauroyrichard.com/gestion-servicios/)
[Si el administrador selecciona uno o varios servicios]
10c. El administrador selecciona uno o varios servicios

11c. habilitarboton(): void (Se habilitan los botones de modificar y eliminar que, hasta que no se selecciona un servicio, aparecen deshabilitados).

[El administrador pulsa el botón eliminar]

12ca. El administrador pulsa el botón "Eliminar"

13ca. eliminarservicio(): submit

14ca. execSQL(sql3) (Elimina las tuplas que contienen el servicio seleccionado en la tabla tx_actuaciones)

sql3: "DELETE FROM tx_actuaciones WHERE codserv= '".$manode."' ";

15ca. execSQL(sql4) (Elimina de la tabla tx_servicios el servicio seleccionado)

sql4: "DELETE FROM tx_servicios WHERE codserv= '".$manode."' ";

16ca. New() (Irá a esta url http://proyecto.tauroyrichard.com/gestion-servicios/)

[El administrador pulsa el botón modificar, habiendo seleccionado un solo servicio]

12cb. El administrador pulsa el botón "Modificar"

13cb. obtenerseleccionado(): Int (obtendrá el código del servicio seleccionado)

14cb. modificarserv():void (Cargará el formulario de modificar servicio con los datos del servicio obtenidos de la base de datos)

new Ajax.Updater('modificarzona', ' divmodificar.php',{ asynchronous:false, postBody: 'seleccionad='+valor});

15cb.execSQL(sql5)

sql5: "select * from tx_servicios WHERE codserv='".$seleccion."'"; (Cargar todos los campos del servicio seleccionado)

16cb. new()

17cb. next()

[Si hay tuplas]

18cba. getInt("codserv"):Int;

19cba. getString ("titulo"):String;

20cba. getString ("descripcion"):String;

21cba. getInt ("precio"):Int;

22cb.close()

23cb. El administrador realiza los cambios que desee y pulsa el botón "Modificar"

24cb. obligatoriosmodif(): Boolean

[Si obligatoriosmodif es true]

25cba. modificarservicioenbd(): Submit

26cba. execSQL(sql6)

$sql6="UPDATE tx_servicios SET titulo = '".$nomserv."', descripcion = '".$descserv."' , precio = '".$precserv."' WHERE codserv ='".$codigo."' "; (Se actualizan los datos del servicio modificado)

27cba. New() (Irá a esta url http://proyecto.tauroyrichard.com/gestion-servicios/)

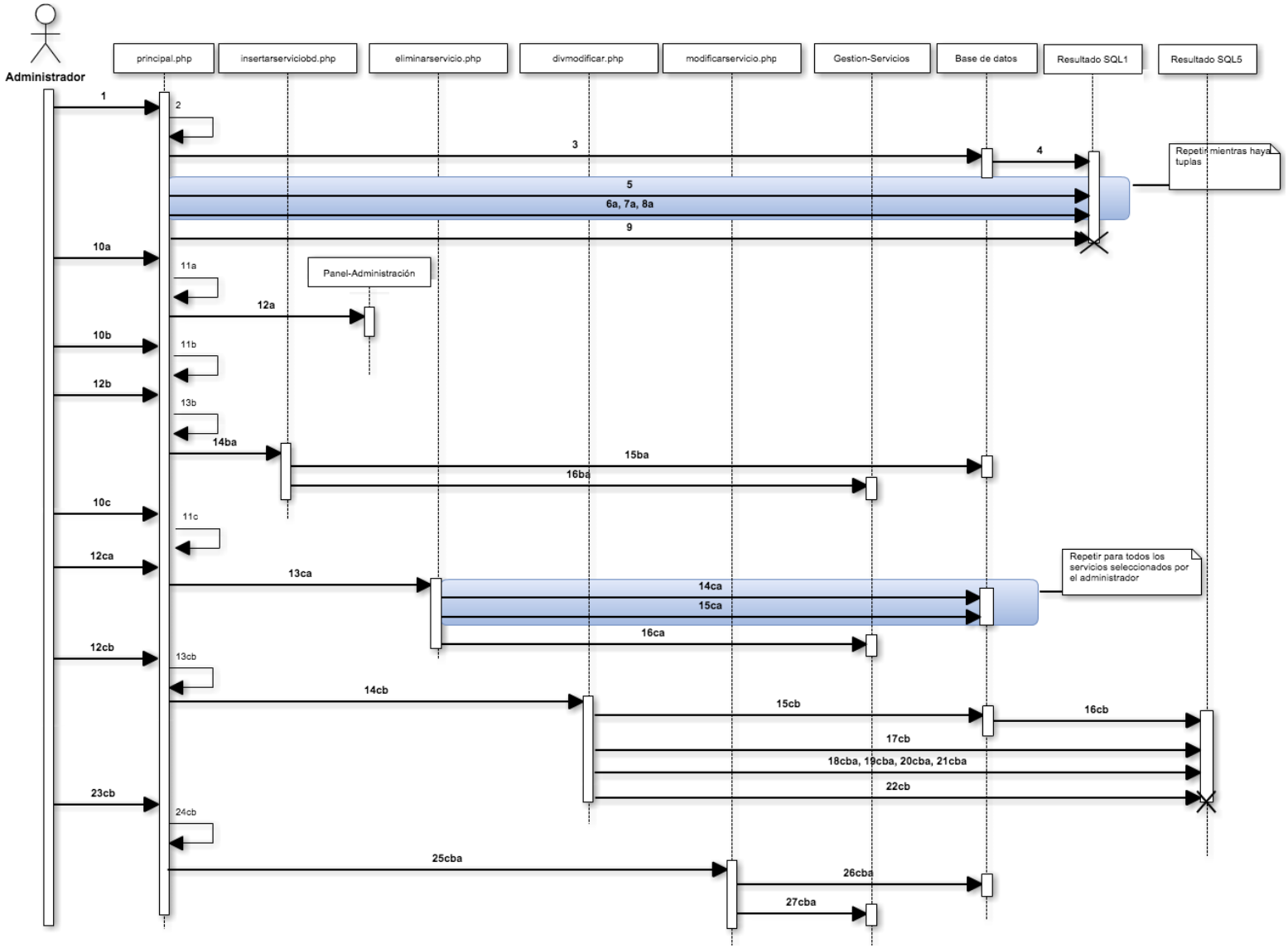


Ilustración 181. - Diagrama de secuencia de caso de uso Gestionar Servicios

11.1.5. Diagrama de secuencia de caso de uso **Historial de reparaciones**

1.El administrador pulsa "Historial de reparaciones"

2.historialclientes()

(Irá a esta url http://proyecto.tauroyrichard.com/historial-de-reparaciones/)

3.execSQL(sql1) **(**Cargar todos los datos de las reparaciones junto con los datos de los vehículos asociados a las mismas)

sql1: "select tx_vehiculo.nombre,tx_vehiculo.matricula,tx_vehiculo.modelo_vehic, tx_reparacion.fecha,tx_reparacion.cod_rep,tx_reparacion.estado_rep,tx_reparacion.factura, tx_reparacion.importe_total from tx_vehiculo inner join tx_reparacion on tx_vehiculo.matricula = tx_reparacion.matricula ORDER BY tx_reparacion.fecha DESC"

4. new()

5. next()

[Si hay tuplas]

6a. getString("nombre"): String;
7a. getString("matricula"):String
8a. getString("modelo_vehic"):String
9a. getDate ("fecha"):Date;
10a. getInt("cod_rep"): Int;
11a. getString("estado_rep"):String
12a. getString("factura"):String
13a. getInt("importe_total"): Int;

14.close()

[Si el administrador pulsa Volver atrás]

15a. El administrador pulsa "Volver atrás"
16a. volveradministracion(): void;
17a. new() (Irá a esta URL http://proyecto.tauroyrichard.com/panel-administracion/)

[Si el administrador decide ordenar el historial]

15b. El administrador pulsa el desplegable de "Ordenar por:"
16b. ordenarpor(): void;
new Ajax.Updater('historialrep', 'ordenarhistorial.php',{ asynchronous:false, postBody: 'orden='+valor });

[Si el administrador ha pulsado ordenar por nombre]

17ba.execSQL(sql2)

sql2: "select tx_vehiculo.nombre,tx_vehiculo.matricula, tx_vehiculo.modelo_vehic, tx_reparacion.fecha , tx_reparacion.cod_rep, tx_reparacion.estado_rep, tx_reparacion.factura, tx_reparacion.importe_total from tx_vehiculo inner join tx_reparacion on tx_vehiculo.matricula = tx_reparacion.matricula ORDER BY tx_vehiculo.nombre ASC" (Cargar todos los datos ordenados por Nombre por orden alfabético)

[Si el administrador ha pulsado ordenar por Fecha]

17bb.execSQL(sql3)

sql3: "select tx_vehiculo.nombre,tx_vehiculo.matricula, tx_vehiculo.modelo_vehic ,tx_reparacion.fecha,tx_reparacion.cod_rep,tx_reparacion.estado_rep, tx_reparacion.factura, tx_reparacion.importe_total from tx_vehiculo inner join tx_reparacion on tx_vehiculo.matricula = tx_reparacion.matricula ORDER BY tx_reparacion.fecha DESC" (Cargar todos los datos ordenados por Fecha, poniendo arriba la más actual y abajo la más antigua)

[Si el administrador ha pulsado ordenar por Matrícula]

17bc.execSQL(sql4)

sql4:"select tx_vehiculo.nombre,tx_vehiculo.matricula, tx_vehiculo.modelo_vehic, tx_reparacion.fecha,tx_reparacion.cod_rep,tx_reparacion.estado_rep, tx_reparacion.factura, tx_reparacion.importe_total from tx_vehiculo inner join tx_reparacion on tx_vehiculo.matricula = tx_reparacion.matricula ORDER BY tx_vehiculo.matricula ASC" (Cargar todos los datos ordenados por Matrícula, en orden ascendente)

18b. new()

19b. next()

[Si hay tuplas]

20ba. getString("nombre"): String;

21ba. getString("matricula"):String

22ba. getString("modelo_vehic"):String

23ba. getDate ("fecha"):Date;

24ba. getInt("cod_rep"): Int;

25ba. getString("estado_rep"):String

26ba. getString("factura"):String

27ba. getInt("importe_total"): Int;

28b.close()

[Si el administrador selecciona una reparación]

15c. El administrador selecciona una reparación

16c. habilitar_boton():void (Se habilitan todos los botones que, hasta que no se selecciona una reparación, aparecen deshabilitados).

[Si el administrador pulsa el botón Ver factura]

17ca. El administrador pulsa el botón "Ver factura"

18ca. obtenerRadioSeleccionado():int; (Obtiene el código de la reparación seleccionada de la tabla)

19ca. verfacturaonline():Submit;

20ca. execSQL(sql5)

sql5: " select * from tx_vehiculo inner join tx_reparacion on tx_vehiculo.matricula = tx_reparacion.matricula Where tx_reparacion.cod_rep ='".$cod_rep."' "; (Cargar todos los campos de la reparación seleccionada junto con los datos del vehículo)

21ca. new()

22ca. next()

[Si hay tuplas]

23caa.getDate ("fecha"):Date;

24caa. getInt("cod_rep"): Int;

25caa. getString("CIA"): String;

26caa. getString("factura"):String

27caa. getString("estado_rep"):String

28caa. getString("concepto"):String

29caa. getString("precio_unitario")String

30caa. getString("importe_concepto"):String

31caa. getInt("importe_total"): Int;

32caa. getInt("base_imp"): Int;

33caa. getInt("cuota_iva"): Int;

34caa. getString("nombre"): String;

35caa. getString("matricula"):String

36caa. getString("DNI"):String

37caa. getString("modelo_vehic"):String

38caa. getInt("telefono"): Int;

39caa. getString("email"):String

40caa. getString("direccion"):String

41caa. getString("ciudad"):String

42caa. getInt("código_postal"): Int;

43caa. getInt("cod_prov"): Int;

44ca.close()

45ca.execSQL(sql6)

sql6: "Select nombre from tx_provincias where cod_prov='". $row['cod_prov']."' " (Cargar el campo nombre de la tabla tx_provincias, de la provincia seleccionada)

46ca. new()

47ca. next()

[Si hay tuplas]

48caa. getString ("nombre"):String;

49ca.close()

50ca. execSQL(sql7)

sql7: "select titulo,descripcion,precio from tx_servicios inner join tx_actuaciones on tx_actuaciones.codserv=tx_servicios.codserv Where tx_actuaciones.cod_rep = '".$cod_rep."' ORDER BY titulo ASC " (Cargar todos los campos de los servicios de la tabla tx_actuaciones)

51ca. new()

52ca. next()

[Si hay tuplas]

53caa. getInt("codserv"):Int;

54caa. getString ("titulo"):String;

55caa. getString ("descripcion"):String;

56caa. getInt ("precio"):Int;

57ca.close()

58ca. Crear_factura_pdf(); (Generará la factura pdf online)

[Si el administrador pulsa el botón Eliminar reparación]

17cb. El administrador pulsa el botón " Eliminar reparación "

18cb. obtenerRadioSeleccionado():int; (Obtiene el código de la reparación seleccionada de la tabla)

19cb. eliminarreparacion()

new Ajax.Updater('historialrep', 'eliminarreparacion.php',{ asynchronous:false, postBody: 'seleccionad='+seleccionado });

20cb. execSQL(sql8) (Eliminar la reparación de la tabla tx_reparacion)

sql8: "DELETE FROM tx_reparacion where cod_rep= '".$cod_rep."'"

21cb. execSQL(sql9) (Eliminar Todos los servicios asociados a esa reparación de la tabla tx_actuaciones)

sql9: "DELETE FROM tx_actuaciones where cod_rep= '".$cod_rep."'"

22cb. execSQL(sql10) (Volver a cargar todos los datos de la tabla del historial reparaciones, sin la tupla que se acaba de eliminar)

sql10: "select tx_vehiculo.nombre,tx_vehiculo.matricula, tx_vehiculo.modelo_vehic, tx_reparacion.fecha, tx_reparacion.cod_rep, tx_reparacion.estado_rep, tx_reparacion.factura, tx_reparacion.importe_total from tx_vehiculo inner join tx_reparacion on tx_vehiculo.matricula = tx_reparacion.matricula ORDER BY tx_reparacion.fecha DESC"

23cb. new()

24cb. next()

[Si hay tuplas]

25cba. getString("nombre"): String;

26cba. getString("matricula"):String

27cba. getString("modelo_vehic"):String
28cba. getDate ("fecha"):Date;
29cba. getInt("cod_rep"): Int;
30cba. getString("estado_rep"):String
31cba. getString("factura"):String
32cba. getInt("importe_total"): Int;
33cb.close()

[Si el administrador pulsa el botón Eliminar todo el historial del vehículo]

17cc. El administrador pulsa el botón "Eliminar todo el historial del vehículo "
18cc. obtenerRadioSeleccionado():int; (Obtiene el código de la reparación seleccionada de la tabla)
19cc. eliminartodohistorial(): void;
new Ajax.Updater('historialrep', 'eliminartodo.php',{ asynchronous:false, postBody: 'seleccionad='+seleccionado });
20cc. execSQL(sql11) (Selecciona la matrícula del vehículo de la reparación seleccionada)
sql11: " Select matricula from tx_reparacion where cod_rep= '".$cod_rep."'"
21cc. new()
22cc. next()
[Si hay tuplas]
23cca. getString("matricula"):String
24cc.close()
25cc. execSQL(sql12) (Selecciona todos los códigos de las reparaciones del ese vehículo)
sql12: "Select cod_rep from tx_reparacion where matricula= '".$matricula."'"
26cc. new()
27cc. next()
[Si hay tuplas]
28cca. getInt ("cod_rep"):Int
29cc.close()
30cc. execSQL(sql13) (Borrará todas las tuplas con el mismo cod_rep)
sql13: "DELETE FROM tx_actuaciones where cod_rep= '".$fila['cod_rep']."'"
31cc. execSQL(sql14) (Borrará la tupla de la tabla tx_vehiculo cuya matrícula sea la indicada. Además se eliminarán todas las tuplas de tx_reparación con la misma matrícula por ser el campo matrícula clave extranjera en la tabla tx_reparacion)
sql14"DELETE FROM tx_vehiculo where matricula= '".$matricula."'"
32cc. execSQL(sql15) (Volver a cargar todos los datos de la tabla del historial reparaciones, sin la tupla que se acaba de eliminar)
sql15: "select tx_vehiculo.nombre,tx_vehiculo.matricula, tx_vehiculo.modelo_vehic, tx_reparacion.fecha, tx_reparacion.cod_rep, tx_reparacion.estado_rep,

tx_reparacion.factura, tx_reparacion.importe_total from tx_vehiculo inner join tx_reparacion on tx_vehiculo.matricula = tx_reparacion.matricula ORDER BY tx_reparacion.fecha DESC"

33cc. new()

34cc. next()

[Si hay tuplas]

35cca. getString("nombre"): String;

36cca. getString("matricula"):String

37cca. getString("modelo_vehic"):String

38cca. getDate ("fecha"):Date;

39cca. getInt("cod_rep"): Int;

40cca. getString("estado_rep"):String

41cca. getString("factura"):String

42cca. getInt("importe_total"): Int;

43cc .close()

[Si el administrador pulsa el botón Modificar]

17cd. El administrador pulsa el botón "Modificar "

18cd. obtenerRadioSeleccionado():int; (Obtiene el código de la reparación seleccionada de la tabla)

19cd. modificarrep():void;

new Ajax.Updater('pagina', 'modificarrep.php',{ asynchronous:false, postBody: 'seleccionad='+seleccionado });

20cd. .execSQL(sql16) (Selecciona la matricula y la fecha de la reparación seleccionada)

sql16: "select matricula,fecha from tx_reparacion where cod_rep ='".$cod_rep."' ""

21cd. new()

22cd. next()

[Si hay tuplas]

23cda. getString("matricula"):String

24cda. getDate ("fecha"):Date;

25cd.close()

26cd.execSQL(sql17) (Carga los datos del vehiculo con esa matrícula de la tabla tx_vehiculo)

sql17: "select * from tx_vehiculo where matricula = '".$matriculaexist."' "

27cd. new()

28cd. next()

[Si hay tuplas]

29cda. getString("nombre"): String;

30cda. getString("matricula"):String

31cda. getString("DNI"):String

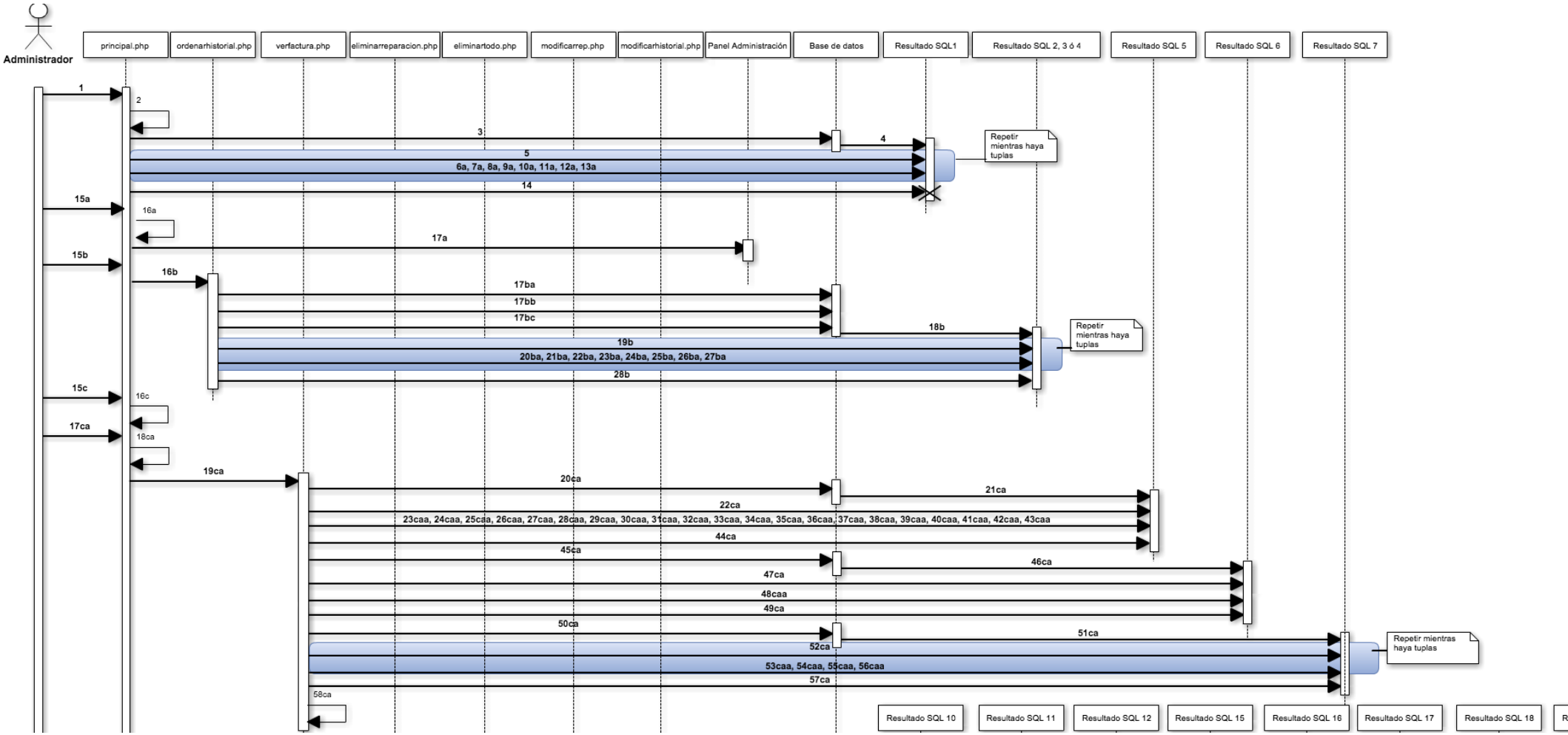
32cda. getString("modelo_vehic"):String

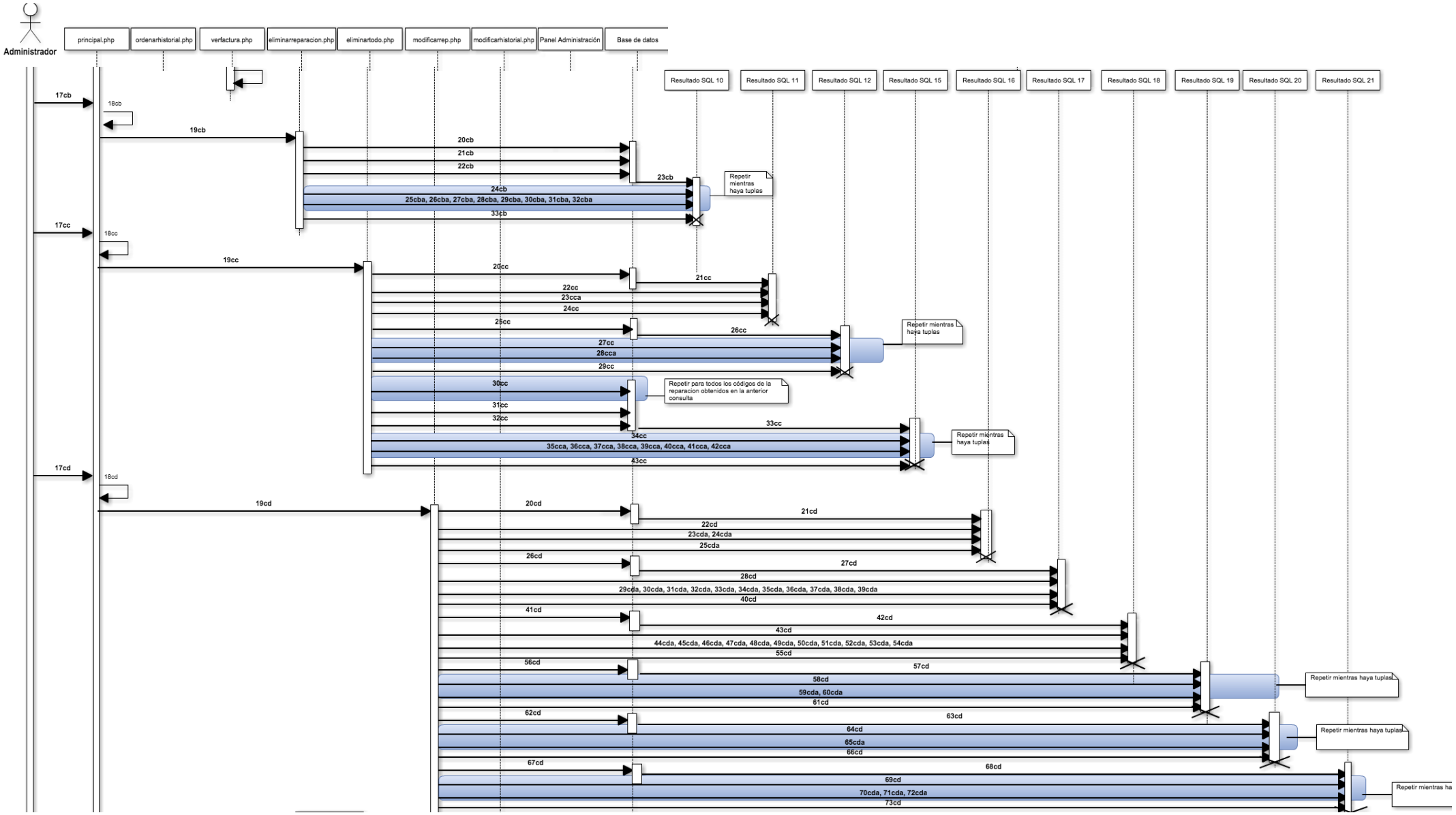
33cda. getInt("telefono"): Int;
34cda. getString("email"):String
35cda. getString("direccion"):String
36cda. getString("ciudad"):String
37cda. getInt("código_postal"): Int;
38cda. getInt("km"): Int;
39cda. getInt("cod_prov"): Int;
40cd.close()
41cd. execSQL(sql18) (Carga los datos de la matrícula cogidos de la tabla tx_reparacion)
sql18: "select * from tx_reparacion where matricula='".$matriculaexist."' and fecha='".$fecha."' "
42cd. new()
43cd. next()
[Si hay tuplas]
44cda. getDate ("fecha"):Date;
45cda. getInt("cod_rep"): Int;
46cda. getString("CIA"): String;
47cda. getString("factura"):String
48cda. getString("estado_rep"):String
49cda. getString("concepto"):String
50cda. getString("precio_unitario")String
51cda. getString("importe_concepto"):String
52cda. getInt("importe_total"): Int;
53cda. getInt("base_imp"): Int;
54cda. getInt("cuota_iva"): Int;
55cd. close()
56cd.SQL(sql19) (Carga los datos de las provincias)
sql19: "select cod_prov,nombre from tx_provincias ORDER BY nombre ASC"
57cd.new()
58cd. next()
[Si hay tuplas]
59cda. getInt("cod_prov "): Int;
60cda.getString("nombre"): String;
61cd.close()
62cd.execSQL(sql20) (Selecciona los servicios asociados a esa reparación en la tabla tx_actuaciones)
sql20: "select codserv from tx_actuaciones where cod_rep='".$reparacion['cod_rep']."'"
63cd.new()
64cd. next()

[Si hay tuplas]
65cda. getInt("codserv"): Int;
66cd.close()
67cd.execSQL(sql21) (Selecciona todos los datos de la tabla tx_servicios)
sql21: select codserv,titulo,descripcion from tx_servicios ORDER BY titulo ASC
68cd.new()
69cd. next()
[Si hay tuplas]
70cda. getInt("codserv "): Int;
71cda. getString("titulo"): String;
72cda. getString("descripcion"): String;
73cd.close()
[Si el administrador pulsa el botón Borrar]
74cda.El administrador pulsa el botón "Borrar"
75cda.Reset() (El formulario volverá a tener los valores cargados al principio)
[Si el administrador pulsa el botón Cancelar y volver atras]
74cdb. El administrador pulsa "Cancelar y Volver atrás"
75cdb. volveratras(): void;
76cdb. new() (Irá a esta URL http://proyecto.tauroyrichard.com/historial-de-reparaciones/)
[Si el administrador pulsa el botón Aceptar]
74cdc. El administrador pulsa el botón Aceptar
75cdc. obligatoriosmodif():boolean
[Si obligatoriosmodif() es true]
76cdca. validaremail():boolean
[Si validaremail() es true]
77cdcaa. modificarbdhistorial():submit;
78cdcaa.Execsql(sql22)(Selecciona el código de la provincia seleccionada)
Sql22:Select cod_prov from tx_provincias where nombre='".$provincia."'
79cdcaa. new()
80cdcaa. next()
[Si hay tuplas]
81cdcaaa. getInt("cod_prov"): Int;
82cdcaa.close()
83cdcaa. Execsql(sql23) (Actualiza la tabla tx_vehiculo)
Sql23: "UPDATE tx_vehiculo SET nombre = '".$nombre."', DNI = '".$dni."', modelo_vehic = '".$modelo."',telefono = '".$telefono."',email = '".$email."',direccion = '".$direccion."',ciudad = '".$ciudad."' ,codigo_postal = '".$codigo."' ,cod_prov ='".$provincia."',km = '".$km."' WHERE matricula ='".$matricula."' ";

84cdcaa. Execsql(sql24) (Selecciona el código de esa reparacion)
Sql24: "select cod_rep from tx_reparacion WHERE matricula ='".$matricula."' and fecha= '".$fecha."' "
85cdcaa. new()
86cdcaa. next()
[Si hay tuplas]
87cdcaaa. getInt("cod_rep"): Int;
88cdcaa.close()
89cdcaa. Execsql(sql25) (Actualiza los campos de la reparación seleccionada)
Sql25: "UPDATE tx_reparacion SET CIA = '".$cia."', factura = '".$factura."' , estado_rep = '".$estado."',base_imp = '".$baseimponible."',cuota_iva = '".$cuotaiva."',importe_total = '".$importetotal."',concepto = '".$concepto."' ,precio_unitario = '".$preciouni."' ,importe_concepto= '".$importe_concepto."' WHERE cod_rep ='".$cod_rep."' ";
[Si el administrador ha puesto el estado de la reparación ="Finalizada" y factura = "Sin pagar"]
90cdcaaa. Execsql(sql26) (Inserta un aviso en la tabla tx_avisos)
sql26: "INSERT INTO tx_avisos (matricula,fecha,codAviso) VALUES ('$matricula',' $fechaa','5')";
91cdcaaa. Execsql (sql27) (Selecciona el idGCM de la tabla tx_app)
sql27: "SELECT IdGCM from tx_app where DNI='$dni' ";
92cdcaaa. new()
93cdcaaa. next()
[Si hay tuplas]
94cdcaaaa. getInt("idGCM"): Int;
95cdcaaa.close()
96cdcaaa. Mandar_notificacion_push(): void;
97cdcaa. Execsql(sql28) (Selecciona los servicios que se han deseleccionado)
Sql28:select codserv from tx_actuaciones WHERE cod_rep ='".$cod_rep."'
98cdcaa. new()
99cdcaa. next()
[Si hay tuplas]
100cdcaaa. getInt("codserv"): Int;
101cdcaa.close()
102cdcaa. Execsql(sql29)(Borra los servicios deseleccionados)
Sql29: DELETE FROM tx_actuaciones WHERE cod_rep ='".$cod_rep."' AND codserv= '".$actuaciones['codserv']."'
103cdcaa. Execsql(sql30) (Selecciona los nuevos servicios seleccionados)
Sql30: select codserv from tx_actuaciones WHERE cod_rep ='".$cod_rep."'
104cdcaa. new()

105cdcaa. next()
[Si hay tuplas]
106cdcaaa. getInt("codserv"): Int;
107cdcaa.close()
108cdcaa.Execsql(sql31)(Inserta los servicios seleccionados)
Sql31: INSERT INTO tx_actuaciones (codserv,cod_rep) VALUES ('$manode','$cod_rep')
109cdcaa. New() (Irá a esta url http://proyecto.tauroyrichard.com/historial-de-reparaciones/)


Administrador
principal.php
ordenarhistorial.php
verfactura.php
eliminarreparacion.php
eliminartodo.php
modificarrep.php
modificarhistorial.php
Panel Administración
Base de datos
Resultado SQL1
Resultado SQL 2, 3 ó 4
Resultado SQL 5
Resultado SQL 6
Resultado SQL 7
Resultado SQL 10
Resultado SQL 11
Resultado SQL 12
Resultado SQL 15
Resultado SQL 16
Resultado SQL 17
Resultado SQL 18
1
2
3
4
5
6a, 7a, 8a, 9a, 10a, 11a, 12a, 13a
14
Repetir mientras haya tuplas
15a
16a
17a
15b
16b
17ba
17bb
17bc
18b
19b
20ba, 21ba, 22ba, 23ba, 24ba, 25ba, 26ba, 27ba
28b
Repetir mientras haya tuplas
15c
16c
17ca
18ca
19ca
20ca
21ca
22ca
23caa, 24caa, 25caa, 26caa, 27caa, 28caa, 29caa, 30caa, 31caa, 32caa, 33caa, 34caa, 35caa, 36caa, 37caa, 38caa, 39caa, 40caa, 41caa, 42caa, 43caa
44ca
45ca
46ca
47ca
48caa
49ca
50ca
51ca
52ca
53caa, 54caa, 55caa, 56caa
57ca
Repetir mientras haya tuplas
58ca


Administrador
principal.php
ordenarhistorial.php
verfactura.php
eliminarreparacion.php
eliminartodo.php
modificarrep.php
modificarhistorial.php
Panel Administración
Base de datos
Resultado SQL 10
Resultado SQL 11
Resultado SQL 12
Resultado SQL 15
Resultado SQL 16
Resultado SQL 17
Resultado SQL 18
Resultado SQL 19
Resultado SQL 20
Resultado SQL 21
17cb
18cb
19cb
20cb
21cb
22cb
23cb
24cb
25cba, 26cba, 27cba, 28cba, 29cba, 30cba, 31cba, 32cba
33cb
Repetir mientras haya tuplas
17cc
18cc
19cc
20cc
21cc
22cc
23cca
24cc
25cc
26cc
27cc
28cca
29cc
Repetir mientras haya tuplas
30cc
Repetir para todos los códigos de la reparacion obtenidos en la anterior consulta
31cc
32cc
33cc
34cc
35cca, 36cca, 37cca, 38cca, 39cca, 40cca, 41cca, 42cca
43cc
Repetir mientras haya tuplas
17cd
18cd
19cd
20cd
21cd
22cd
23cda, 24cda
25cda
26cd
27cd
28cd
29cda, 30cda, 31cda, 32cda, 33cda, 34cda, 35cda, 36cda, 37cda, 38cda, 39cda
40cd
41cd
42cd
43cd
44cda, 45cda, 46cda, 47cda, 48cda, 49cda, 50cda, 51cda, 52cda, 53cda, 54cda
55cd
56cd
57cd
58cd
59cda, 60cda
61cd
Repetir mientras haya tuplas
62cd
63cd
64cd
65cda
66cd
Repetir mientras haya tuplas
67cd
68cd
69cd
70cda, 71cda, 72cda
73cd

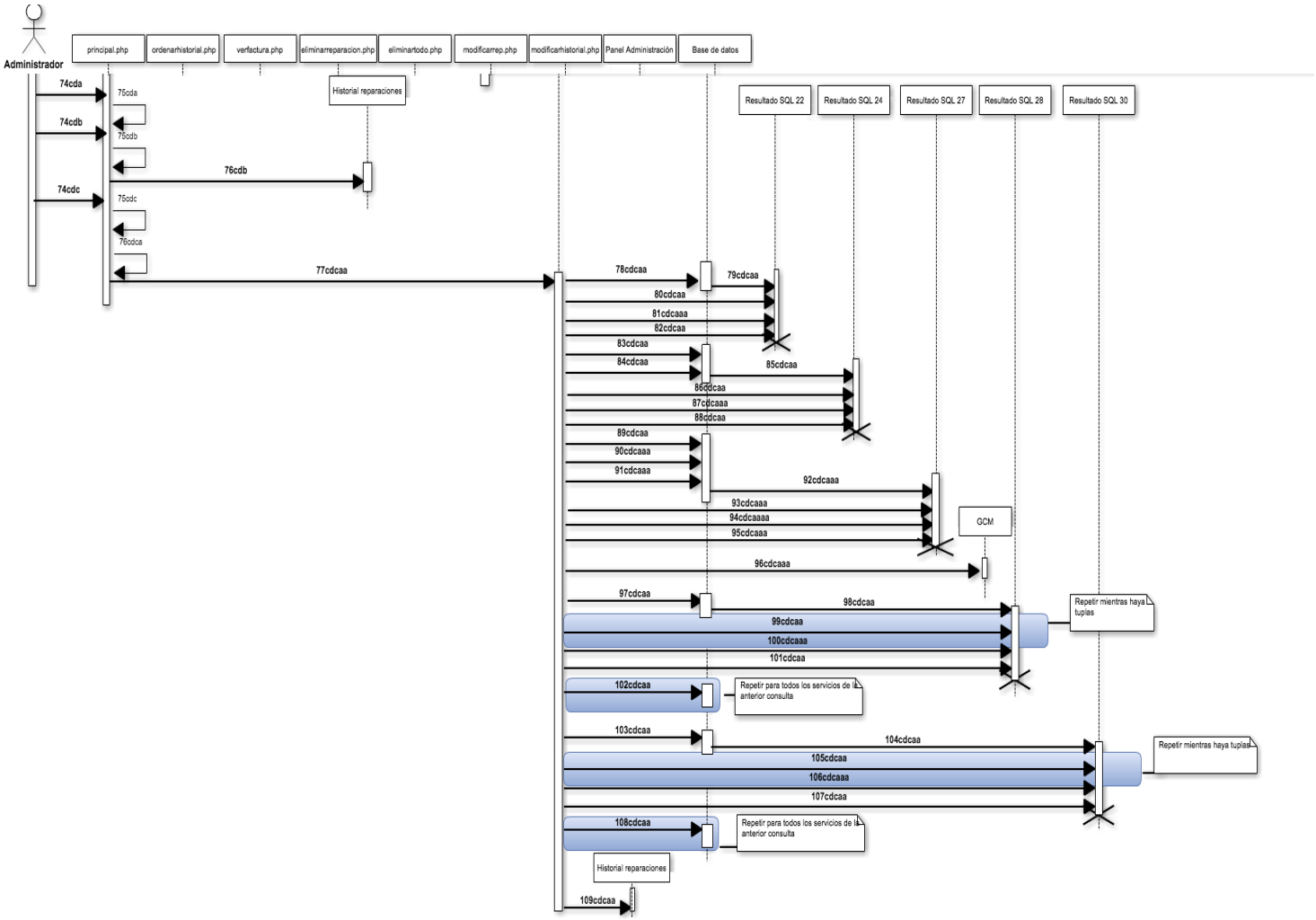


Ilustración 182. - Diagrama de secuencia de caso de uso Historial de reparaciones

11.1.6. Diagrama de secuencia **Ver cámaras web**

1.El administrador pulsa "Ver cámaras web"

2. vercamarasweb() (Irá a esta url http://proyecto.tauroyrichard.com/ver-camaras-web/)

3. El administrador introduce usuario y contraseña y visualiza las cámaras web

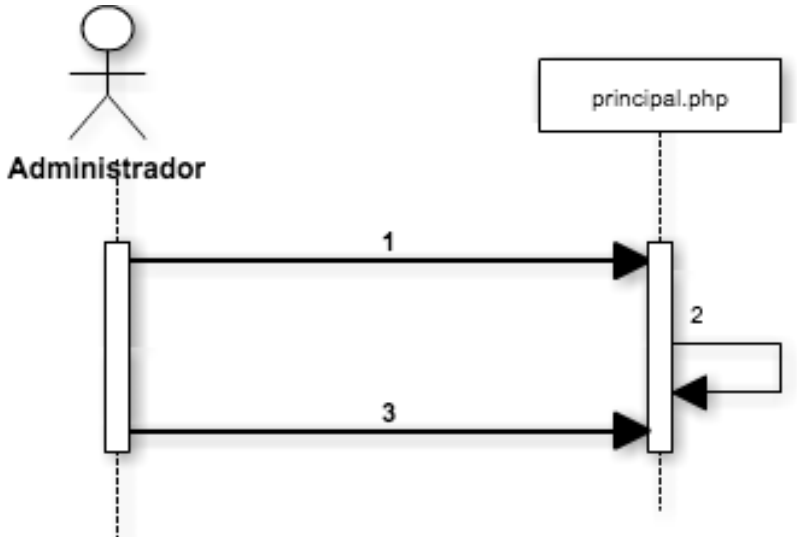


Ilustración 183. - Diagrama de caso de uso Ver cámaras web

11.1.7. Diagrama de secuencia de caso de uso **Administrar avisos**

1.El administrador pulsa "Administrar avisos"

2. administrar_avisos() (Irá a esta url http://proyecto.tauroyrichard.com/administrar-avisos/)

[Si el administrador pulsa Eliminar aviso]

3a. El administrador pulsa "Eliminar aviso"
4a. eliminaraviso(): void (Generará un desplegable con todas los avisos que hay en la base de datos)
new Ajax.Updater('espacio', 'eliminaraviso.php',{ asynchronous:false }});
5a. execSQL(sql1)
sql1:"select tx_avisos.id, tx_avisos.codAviso,tx_avisos.matricula, tx_avisos.fecha, tx_avisosGenericos.descripcion from tx_avisos inner join tx_avisosGenericos on tx_avisos.codAviso = tx_avisosGenericos.codAviso ORDER BY tx_avisosGenericos .descripcion " (Cargará todos los datos de los avisos que se encuentran en la tabla tx_avisos)
6a. new()
7a. next()
[Si hay tuplas]
8aa. getDatos():Datos
9a.close()
10a. El administrador escoge un aviso del desplegable
11a. botoneliminar(): void; (Generará el botón de eliminar)
12a. El administrador después de seleccionar un aviso pulsa el botón eliminar.
13a. Submit();
14a. execSQL(sql2)
sql2: "DELETE FROM tx_avisos WHERE id='".$cod_aviso."' " (Elimina el aviso seleccionado de la tabla tx_avisos)
15a. new() (Irá a esta URL http://proyecto.tauroyrichard.com/administrar-avisos/

[Si el administrador pulsa Introducir nuevo aviso]

3b. El administrador pulsa el botón "Introducir nuevo aviso"
4b.introduciraviso():void (Generará un desplegable con todas las matrículas)
new Ajax.Updater('espacio', ' introduciraviso.php',{ asynchronous:false}); (Generará un desplegable con todas las matrículas, también generará el desplegable de los avisos en un div oculto)
5b. execSQL(sql3)
sql3: "select matricula from tx_vehiculo " (Cargará todas las matrículas)

6b. new()
7b. next()
[Si hay tuplas]
8ba. getDatos():Datos
9b.close()
10b. execSQL(sql4)
sql4: "select * from tx_avisosGenericos ORDER BY descripcion " (Cargará todos los datos de la tabla tx_avisosGenericos)
11b. new()
12b. next()
[Si hay tuplas]
13ba. getDatos():Datos
14b.close()
15b. El administrador selecciona una matrícula del desplegable
16b. abrirformularionuevo():void (Mostrará el desplegable de los avisos, extraídos en la anterior consulta)
[Si el adminsitrador pulsa "Carga de aire acondicionado" o "Pasar la ITV"]
17ba. El administrador selecciona un aviso de los mencionados en el desplegable
18ba. abriravisos() (Se abrirá un nuevo formulario con dos campos, el primero es Última Fecha y el segundo Siguiente Fecha)
19ba. El administrador rellena el primer campo
20ba. Validarfecha():boolean
[Si validarfecha() es true]
21baa. cambiarfecha() (Se cambiará la fecha del siguiente campo, dependiendo cual de los dos avisos se haya elegido)
[Si el administrador pulsa "Cancelar"]
22baa. El administrador pulsa "Cancelar"
23baa. volveratras(): void;
24baa. new() (Irá a esta URL http://proyecto.tauroyrichard.com/administrar-avisos/)
[Si el administrador pulsa "Aceptar"]
22bab. El administrador pulsa "Aceptar"
23bab. validarfecha(fecha actual, fecha proxima): boolean
[Si devuelve true]
24baba. obligatoriosavisos():boolean
[Si devuelve true]
25babaa. insertarbdofer():Submit
26babaa. execSQL(sql5)
sql5: " DELETE FROM tx_avisos WHERE matricula='".$matricula."' and codAviso='".$aviso."'"; (Elimina el aviso anterior introducido para esa matricula)
27babaa. execSQL(sql6)

sql6: " INSERT INTO tx_avisos (matricula, fecha, codAviso, fechaprox) VALUES ('$matricula',' $fechaactual', '$aviso', '$fechaprox')" (Inserta el nuevo aviso)

[Si el adminsitrador pulsa "Finalización de la reparación"]

17bb. El administrador selecciona el aviso.

18bb. abriravisos() (Se cargará el desplegable con las fechas de las reparaciones del vehículo elegido)

new Ajax.Updater('fecharep', 'obtfecha.php',{ asynchronous:false, postBody: 'matric='+matricula});

19bb. execSQL(sql7)

sql7: "select fecha,cod_rep from tx_reparacion where matricula='".$matricula."' and estado_rep='Sin finalizar'"(Cargará las fechas de las reparaciones del vehículo elegido, cuyo campo estado de la reparación tenga el valor sin finalizar)

20bb. new()

21bb. next()

[Si hay tuplas]

22bba. getDatos():Datos

23bb.close()

24bb. El administrador escoge una fecha del desplegable

25bb. abrirformulfinalizado() (Abre el formulario correspondiente)

[El administrador pulsa cancelar]

26bba. El administrador rellena los datos y pulsa cancelar

27bba. volveratras(): void;

28bba. new() (Irá a esta URL http://proyecto.tauroyrichard.com/administrar-avisos/)

[El administrador pulsa aceptar]

26bbb. El administrador rellena los datos y pulsa Aceptar.

27bbb. insertarbdofer():Submit

28bbb. execSQL(sql8) (Elimina el anterior aviso)

sql8: "DELETE FROM tx_avisos WHERE matricula='".$matricula3."' and codAviso='".$aviso."'";

29bbb. execSQL(sql9) (Inserta el nuevo aviso)

sql9 "INSERT INTO tx_avisos (matricula,fecha,codAviso) VALUES ('$matricula3',' $fecha','$aviso')";

30bbb. execSQL(sq10) (Actualiza el estado de la reparacion)

sql10 ""UPDATE tx_reparacion SET _rep='".$finaliz."' where matricula='".$matricula3."' and fecha='".$fech."'";

31bbb. execSQL(sql11)

sql11: "select DNI,nombre from tx_vehiculo Where matricula='".$matricula3."'

"(Saca el nombre y dni del dueño del vehículo)

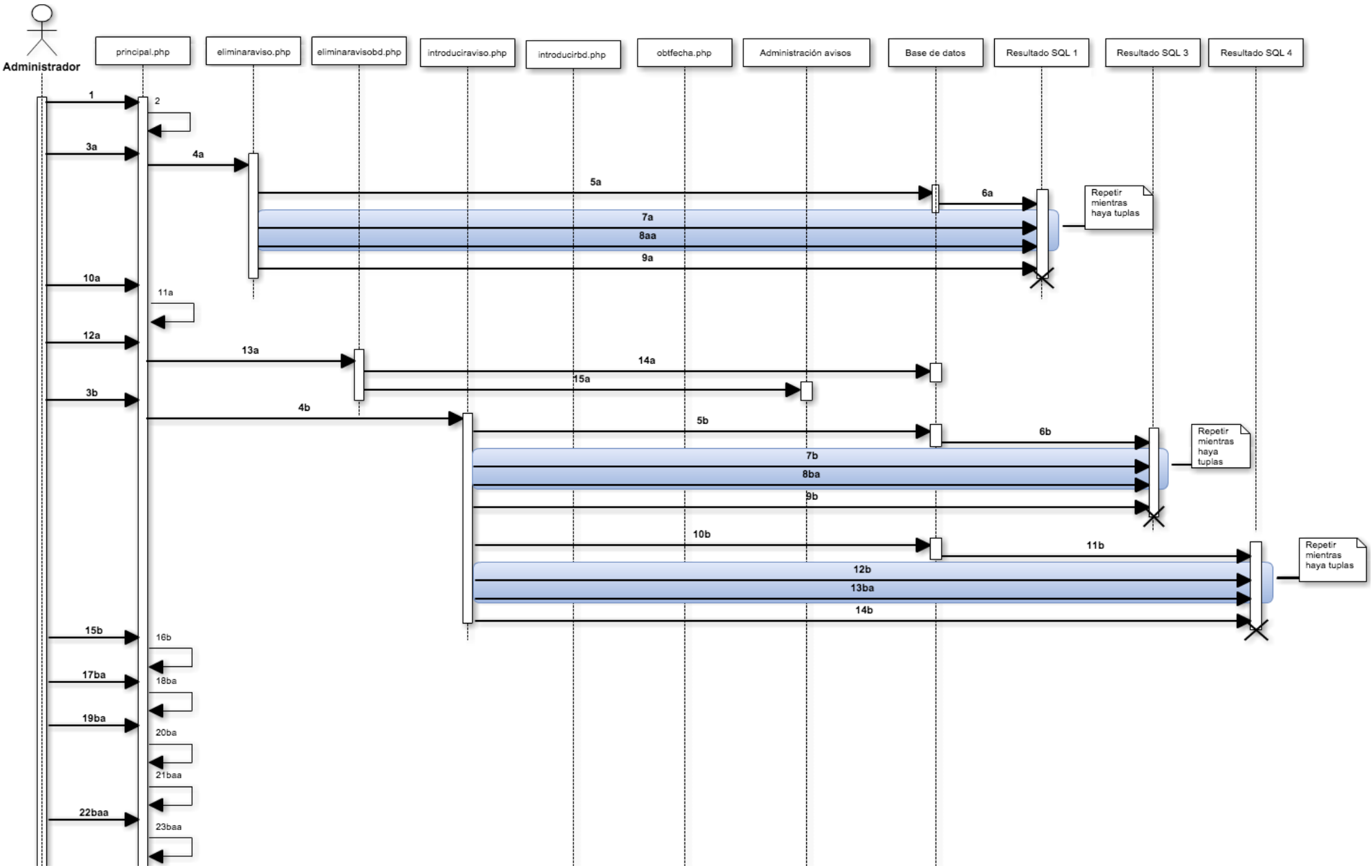
32bbb. new()

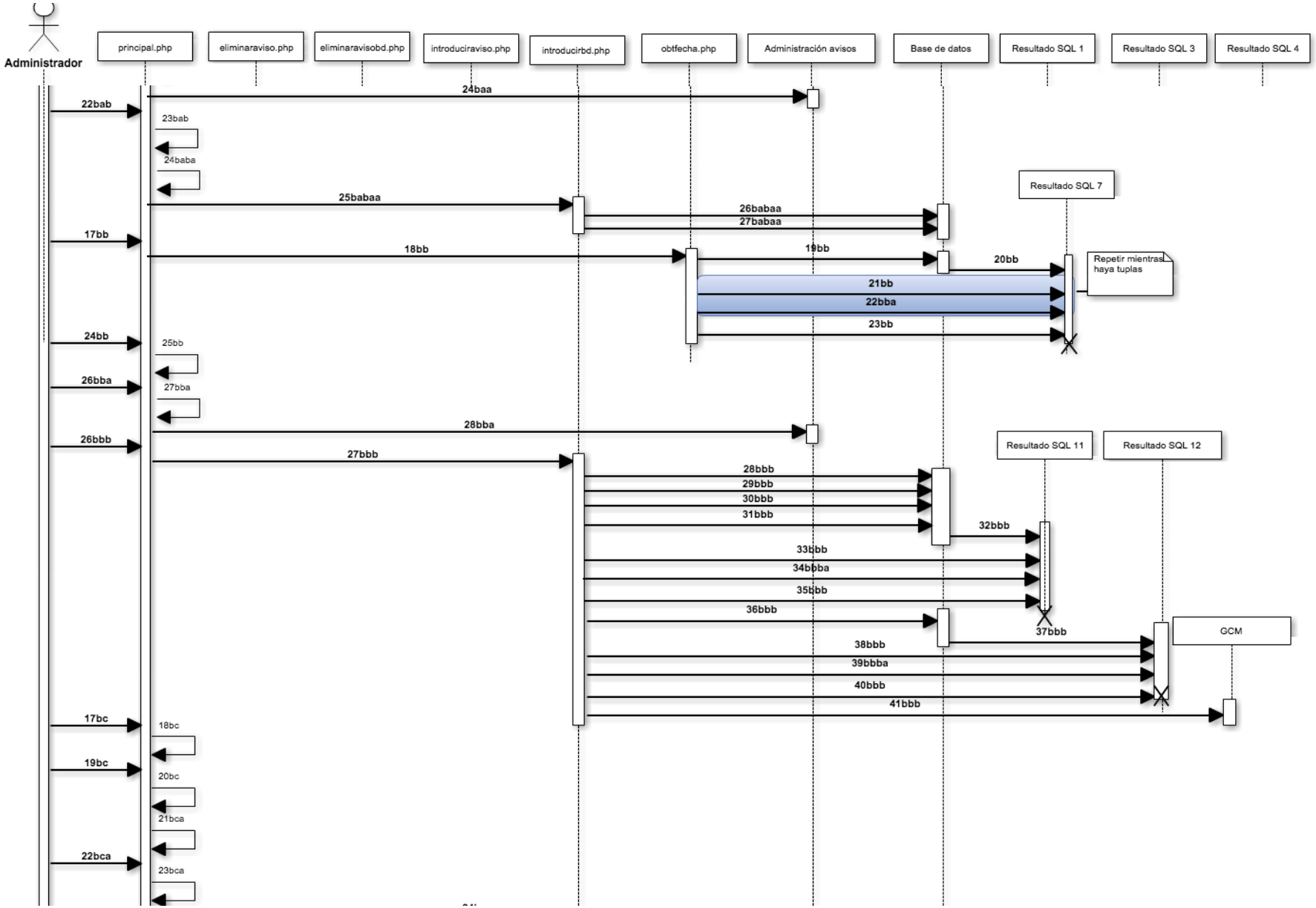
33bbb. next()
[Si hay tuplas]
34bbba. getDatos():Datos
35bbb.close()
36bbb. execSQL(sql12)
sql12 "SELECT IdGCM from tx_app where DNI='$dni' "; (el id de registro del dispositivo en el GCM del propietario del vehículo)
37bbb. new()
38bbb. next()
[Si hay tuplas]
39bbba. getDatos():Datos
40bbb.close()
41bbb. Enviaraviso()

[Si el adminsitrador pulsa cualquiera de los otros avisos]
17bc.El administrador selecciona un aviso.
18bc. abriravisos() (Se cargará un formulario con dos campos a rellenar)
19bc. El administrador rellena el primer campo
20bc. esletra():boolean
[Si esletra() es false]
21bca. cambiarvalor() (Se cambiará el valor del campo kilometraje siguiente de revisión en función del aviso escogido)
[Si el administrador pulsa "Cancelar"]
22bca. El administrador pulsa "Cancelar"
23bca. volveratras(): void;
24bca. new() (Irá a esta URL http://proyecto.tauroyrichard.com/administrar-avisos/)
[Si el administrador pulsa "Aceptar"]
22bcb. El administrador pulsa "Aceptar"
23bcb esletra (kmactual, kmsig): boolean
[Si devuelve false]
24bcba. obligatoriosavisos():boolean
[Si devuelve true]
25bcbaa. insertarbdofer():Submit
26bcbaa. execSQL(sql13)
sql13: " DELETE FROM tx_avisos WHERE matricula='".$matricula."' and codAviso='".$aviso."'"; (Elimina el aviso anterior introducido para esa matricula)
27bcbaa. execSQL(sql14)
Sql14: "INSERT INTO tx_avisos (matricula,fecha,codAviso,km,kmanterior) VALUES ('$matricula','$fecha','$aviso','$kmsig','$kmactual')" (Inserta el nuevo aviso)
28bcbaa. execSQL(sql15)

sql15: "UPDATE tx_vehiculo SET km ='".$kmactual."'where matricula='".$matricula."' " (Actualiza los kilómetros actuales del vehículo en la tabla tx_vehiculo)

42b. new() (Irá a esta URL http://proyecto.tauroyrichard.com/administrar-avisos/)


Administrador
principal.php
eliminaraviso.php
eliminaravisobd.php
introduciraviso.php
introducirbd.php
obtfecha.php
Administración avisos
Base de datos
Resultado SQL 1
Resultado SQL 3
Resultado SQL 4
1
2
3a
4a
5a
6a
7a
8aa
9a
Repetir mientras haya tuplas
10a
11a
12a
13a
14a
15a
3b
4b
5b
6b
7b
8ba
9b
Repetir mientras haya tuplas
10b
11b
12b
13ba
14b
Repetir mientras haya tuplas
15b
16b
17ba
18ba
19ba
20ba
21baa
22baa
23baa


Administrador
principal.php
eliminaraviso.php
eliminaravisobd.php
introduciraviso.php
introducirbd.php
obtfecha.php
Administración avisos
Base de datos
Resultado SQL 1
Resultado SQL 3
Resultado SQL 4
Resultado SQL 7
Resultado SQL 11
Resultado SQL 12
GCM
Repetir mientras haya tuplas
22bab
23bab
24baba
24baa
25babaa
26babaa
27babaa
17bb
18bb
19bb
20bb
21bb
22bba
23bb
24bb
25bb
26bba
27bba
28bba
26bbb
27bbb
28bbb
29bbb
30bbb
31bbb
32bbb
33bbb
34bbba
35bbb
36bbb
37bbb
38bbb
39bbba
40bbb
41bbb
17bc
18bc
19bc
20bc
21bca
22bca
23bca

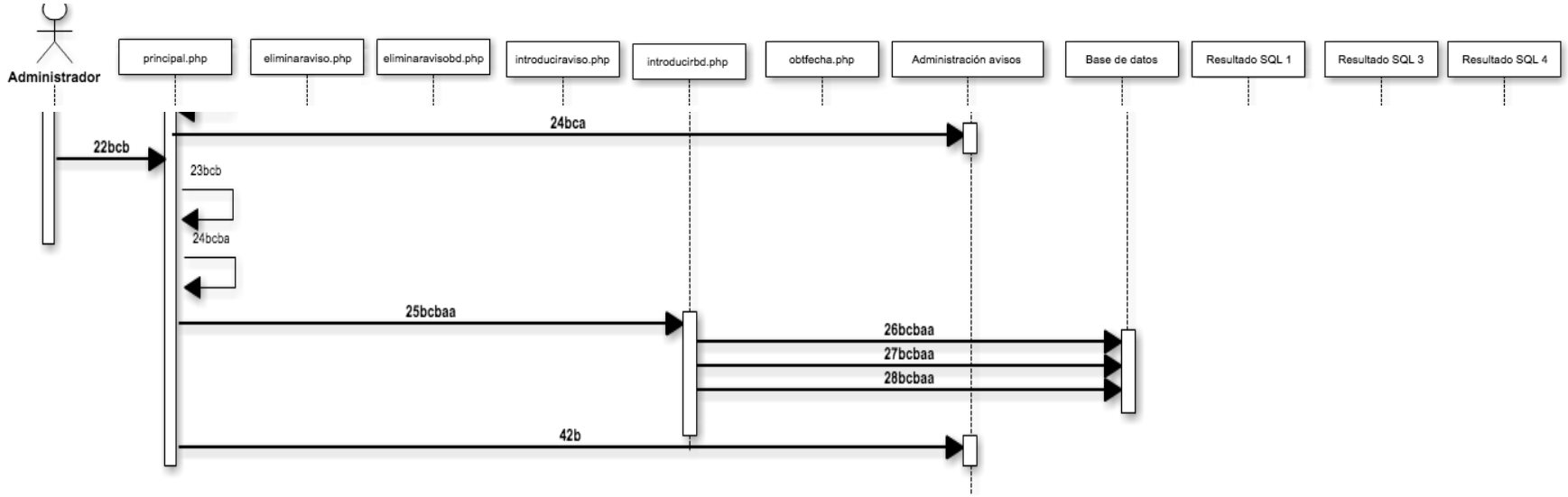


Ilustración 184. - Diagrama de secuencia del caso de uso Administrar avisos

11.1.8. Diagrama de secuencia **Administrar ofertas**

1.El administrador pulsa "Administrar ofertas"

2. administraroferta() (Irá a esta url http://proyecto.tauroyrichard.com/administrar-ofertas/)

[Si el administrador pulsa Introducir nueva oferta]

3a. El administrador pulsa "Introducir nueva oferta"

4a. introduciroferta():void (Se cargará el formulario de introducción de una oferta nueva)

new Ajax.Updater('espacio', 'introduciroferta.php');

[Si el administrador pulsa el botón cancelar]

5aa. El administrador pulsa "Cancelar"

6aa. volveratras(): void;

7aa. new() (Irá a esta URL http://proyecto.tauroyrichard.com/administrar-ofertas/)

[Si el administrador pulsa el botón Reset]

5ab. El administrador pulsa "Reset"

6ab. Reset() (El formulario volverá a todos los campos en blanco)

[Si el administrador pulsa el botón Aceptar]

5ac. El administrador pulsa "Aceptar"

6ac. obligatoriosofer(): boolean

[Si obligatoriosofer es true]

7aca. insertarbdofer(): submit

8aca.move_uploaded_file($_FILES['archivo']['tmp_name'],'imagenes/'
.$nombre) (Guarda la imagen en el servidor)

9aca. execSQL(sql1); (Se inserta la oferta en la tabla tx_ofertas)

sql1: "INSERT INTO tx_ofertas (titulo,descrip_corta,descrip_larga,imagen) VALUES ('$titulo','$descorto','$desclarga','$archivo_subir')";

10aca. Insertar_oferta_en_facebook(): void

11aca. new() (Irá a esta URL http://proyecto.tauroyrichard.com/administrar-ofertas/)

[Si el administrador pulsa Eliminar ofertas]

3b. El administrador pulsa "Eliminar ofertas"

4b. eliminaroferta(): void (Generará un desplegable con todas las ofertas que hay en la base de datos)

new Ajax.Updater('espacio', 'eliminaroferta.php',{ asynchronous:false });

5b. execSQL(sql2)

sql2: "select * from tx_ofertas" (Cargará todos los datos de las ofertas)

6b. new()

7b. next()

[Si hay tuplas]
8ba. getInt("cod_oferta"):Int;
9ba. getString ("titulo"):String;
10ba. getString ("descrip_corta"):String;
11ba. getString ("descrip_larga"):String;
12ba. getString ("imagen"):String;
13b.close()
14b. El administrador escoge una oferta del desplegable
15b. abrirboton(): void; (Generará el botón de eliminar)
new Ajax.Updater('botoneliminar', 'eliminarbd.php',{ asynchronous:false, postBody: 'cod_oferta='+cod });
16b. El administrador después de seleccionar una oferta pulsa el botón eliminar.
17b. Submit();
18b. execSQL(sql3)
sql3: "Select imagen from tx_ofertas WHERE cod_oferta ='".$cod_oferta."' ";
(Seleccionar el path de la imagen de la oferta seleccionada)
19b. new()
20b. next()
[Si hay tuplas]
21ba. getString ("imagen"):String;
22b.close()
23b. unlink(realpath($fila[0])); (Eliminará la imagen del servidor)
24b. execSQL(sql4)
sql4: "DELETE FROM tx_ofertas WHERE cod_oferta ='".$cod_oferta."' "; (Elimina la oferta seleccionada)
25b. new() (Irá a esta URL http://proyecto.tauroyrichard.com/administrar-ofertas/)

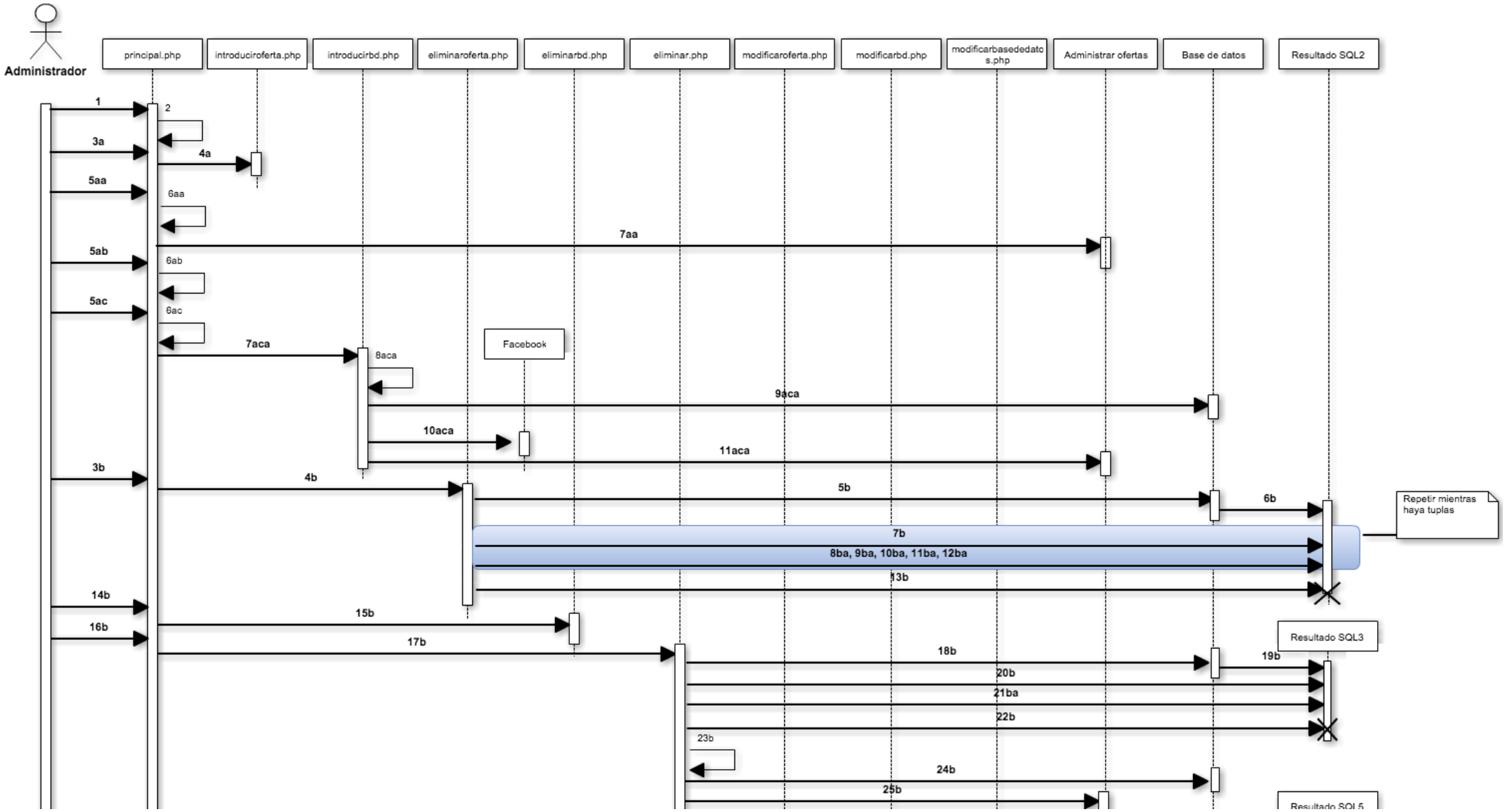
[Si el administrador pulsa Modificar oferta]

3c. El administrador pulsa "Modificar oferta"
4c. modificarofertas(): void; (Generará un desplegable con todas las ofertas)
new Ajax.Updater('espacio', ' modificaroferta.php',{ asynchronous:false});
5c. execSQL(sql5)
sql5: "select * from tx_ofertas" (Cargará todos los datos de las ofertas)
6c. new()
7c. next()
[Si hay tuplas]
8ca. getInt("cod_oferta"):Int;
9ca. getString ("titulo"):String;
10ca. getString ("descrip_corta"):String;
11ca. getString ("descrip_larga"):String;

12ca. getString ("imagen"):String;
13c.close()
14c. El administrador escoge una oferta del desplegable
15c. abrirformul():void; (Creará un formulario con todos sus campos cargados con los datos de la oferta seleccionada)
new Ajax.Updater('formularioa', 'modificarbd.php',{ asynchronous:false, postBody: 'cod_oferta='+cod });
16c. execSQL(sql6)
sql6: " select * from tx_ofertas where cod_oferta='".$cod_oferta."' "(Cargará todos los datos de la oferta seleccionada)
17c. new()
18c. next()
[Si hay tuplas]
19ca. getInt("cod_oferta"):Int;
20ca. getString ("titulo"):String;
21ca. getString ("descrip_corta"):String;
22ca. getString ("descrip_larga"):String;
23ca. getString ("imagen"):String;
24c.close()
[Si el administrador pulsa el botón cancelar]
25ca. El administrador pulsa "Cancelar"
26ca. volveratras(): void;
27ca. new() (Irá a esta URL http://proyecto.tauroyrichard.com/administrar-ofertas/)
[Si el administrador pulsa el botón Reset]
25cb. El administrador pulsa "Reset"
26cb. Reset() (El formulario volverá a todos los campos en blanco)
[Si el administrador pulsa el botón Aceptar]
25cc. El administrador pulsa "Aceptar"
26cc. obligatoriosofer(): boolean
[Si obligatoriosofer es true]
27cca. modficbdofer(): submit
[Si la imagen no se ha cambiado]
28ccaa. execSQL(sql7); (Se actualizan los datos de la oferta en la tabla tx_ofertas)
Sql7: "UPDATE tx_ofertas SET titulo='".$titulo."',
descrip_corta='".$descorto."' , descrip_larga='".$desclarga."' WHERE cod_oferta ='".$cod_oferta."' ";
[Si no]
28ccab. move_uploaded_file($_FILES['archivo']['tmp_name'], 'imagenes/' .$nombre) (Guarda la nueva imagen en el servidor)
29ccab. @unlink(realpath($imgantigua)); (Borra la imagen antigua del servidor)

30ccab. execSQL(sql8); (Se actualizan los datso de la oferta en la tabla tx_ofertas) Sql8: "UPDATE tx_ofertas SET titulo='".$titulo."', descrip_corta='".$descorto."', descrip_larga='".$desclarga."', imagen= '".$archivo_subir." ' WHERE cod_oferta ='".$cod_oferta."' ";

31 cca. new() (Irá a esta URL http://proyecto.tauroyrichard.com/administrar-ofertas/)


Administrador
principal.php
introduciroferta.php
introducirbd.php
eliminaroferta.php
eliminarbd.php
eliminar.php
modificaroferta.php
modificarbd.php
modificarbasededatos.php
Administrar ofertas
Base de datos
Resultado SQL2
Facebook
Resultado SQL3
Resultado SQL5
1
2
3a
4a
5aa
6aa
7aa
5ab
6ab
5ac
6ac
7aca
8aca
9aca
10aca
11aca
3b
4b
5b
6b
7b
8ba, 9ba, 10ba, 11ba, 12ba
13b
14b
15b
16b
17b
18b
19b
20b
21ba
22b
23b
24b
25b
Repetir mientras haya tuplas

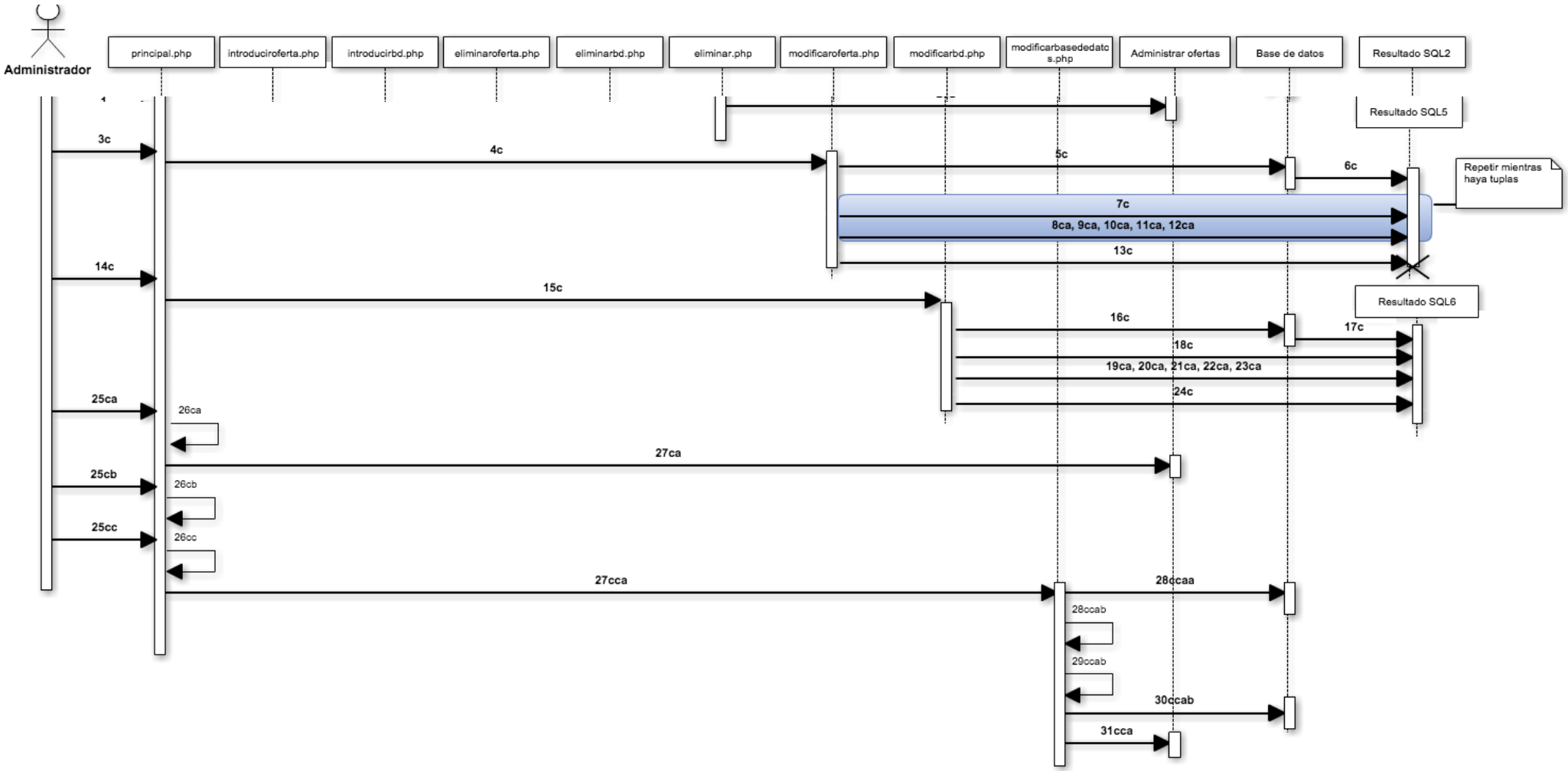


Ilustración 185. - Diagrama de secuencia de caso de uso Administrar Ofertas

11.1.9. Diagrama de secuencia **Ver ofertas**

1.El cliente pulsa el botón "Ofertas" del menú de la página web

2. ofertas()(Irá a esta url http://proyecto.tauroyrichard.com/ofertas/)

3.execSQL(sql1)

sql1: "select * from tx_ofertas ORDER BY cod_oferta DESC "(Cargar todas las ofertas)

4. new()

5. next()

[Si hay tuplas]

6a. getDatos ():Datos;

7.close()

[Si el cliente pulsa en una oferta]

8a. El cliente pulsa una oferta

9a. oferta_ind():void (Irá a esta url http://proyecto.tauroyrichard.com/oferta-ind/?cod_ofer=$cod_ofer)

10a.execSQL(sql2)

sql2: "Select * from tx_ofertas WHERE cod_oferta ='".$cod_oferta."' "(Carga todos los datos de la oferta seleccionada)

11a new()

12a. next()

[Si hay tuplas]

13aa. getDatos ():Datos;

14a.close()

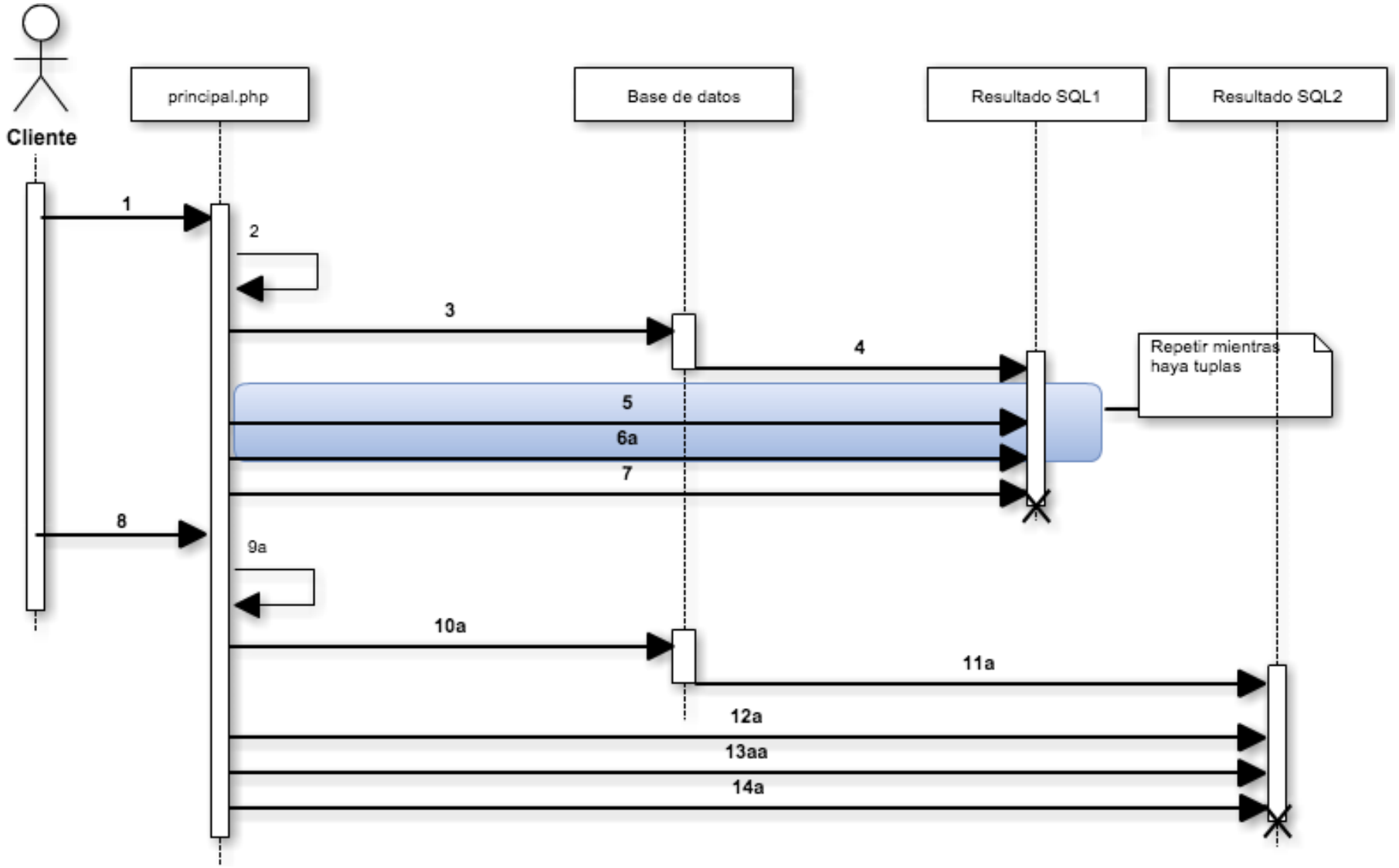


Ilustración 186. - Diagrama de secuencia de caso de uso Ver ofertas

11.1.10. Diagrama de secuencia **Identificación cliente**

1.El cliente introduce la matrícula y la contraseña requerida y pulsa “Aceptar”.

2.Obligatorios(): Boolean

[Si obligatorios() devuelve true]

3a. Identificar()

new Ajax.Updater('espacio', 'identificarseclientes.php',{asynchronous:false, postBody: 'pwd='+$pwd+'&matricula='+$matricula});

4a.execSQL(sql1)

sql1: "select nombre from tx_vehiculo Where matricula='".$matricula."' and password='".$pwd."' (Comprueba que la matrícula y contraseña sea correcta y saca el nombre del cliente)

5a. new()

6a. next()

[Si hay tuplas]

7aa. getString ("Nombre"): String

8a.close()

[Si la consulta devuelve el valor nombre]

9aa. Administracionclientes()

10aa. New() (Irá a esta url http://proyecto.tauroyrichard.com/administracion-clientes/?matricula = +matricula)

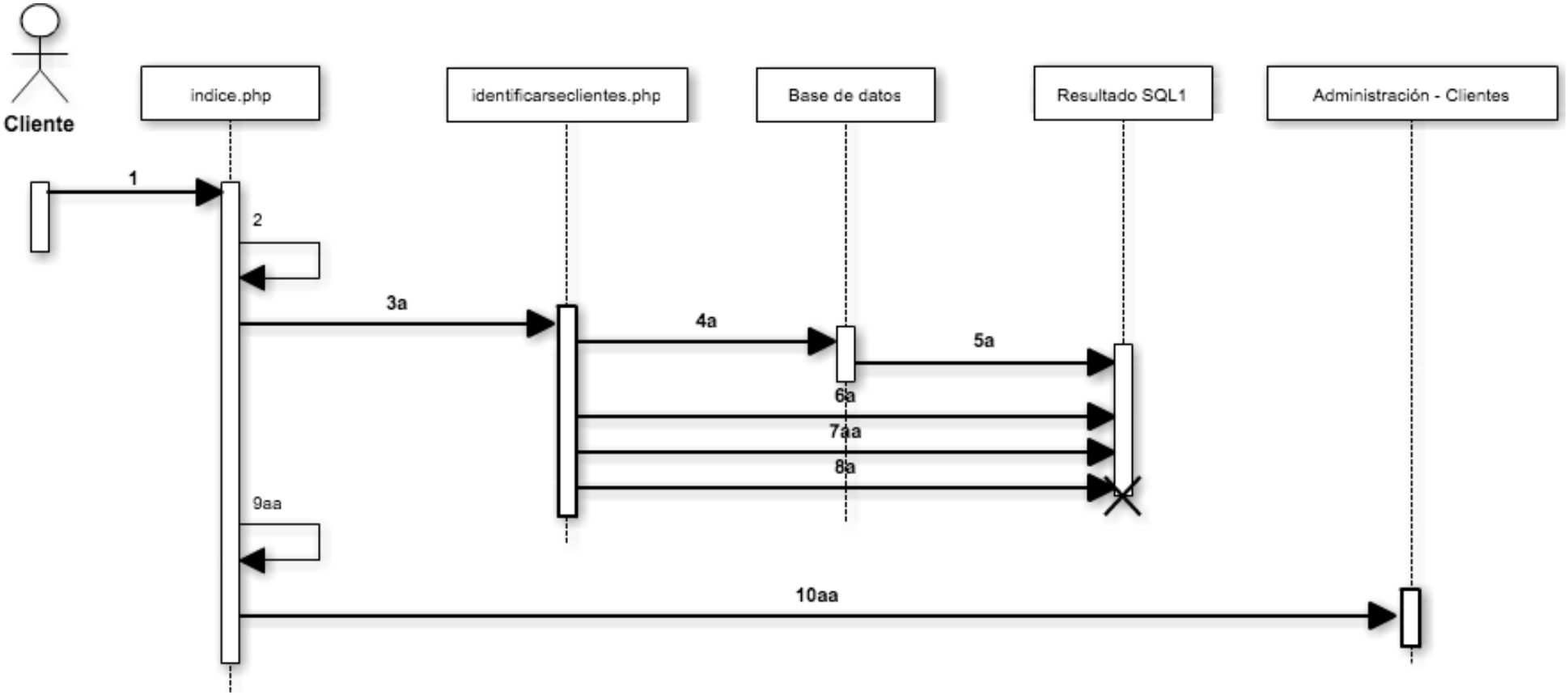


Ilustración 187. - Diagrama de secuencia de caso de uso Identificación cliente

11.1.11. Diagrama de secuencia **Recuperar contraseña**

1.El cliente pulsa el botón "Recuperar contraseña"

2. recuperarpwd()

new Ajax.Updater('espacio', 'recuperarcontraseña.php',{asynchronous:false, postBody: 'matricula='+val2});

(Se genera el formulario correspondiente a recuperar contraseña)

3. El cliente introduce la matrícula de su vehículo y pulsa "Aceptar"

4. Submit()

5.execSQL(sql1)

sql1: "select email,nombre from tx_vehiculo WHERE matricula ='".$matricula."' '' (Extrae el email y el nombre de la base de datos)

6. new()

7. next()

[Si hay tuplas]

8a. getString ("Nombre"): String
9a. getString ("Email"): String
10.close()

[Si la consulta devuelve algún valor]

11a. execSQL(sql2)
sql2: " UPDATE tx_vehiculo SET password ='".$pwd."' WHERE matricula ='".$matricula."' '' (Se actualiza la contraseña del vehículo en la tabla tx_vehiculo)
12a. enviaremail():void;
13a. identificarcliente()
14a. New() (Irá a esta url http://proyecto.tauroyrichard.com/identificar-clientes/)

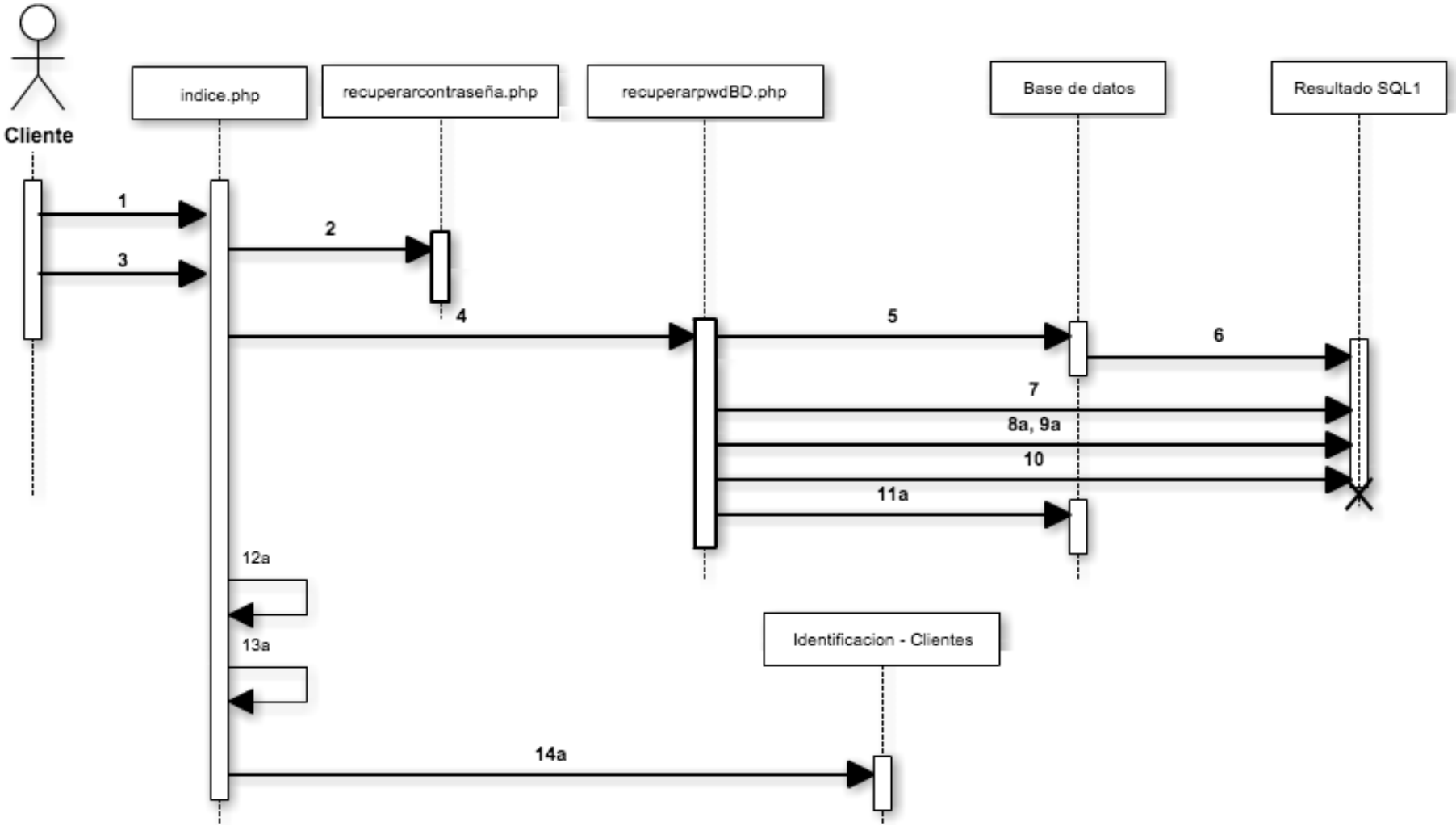


Ilustración 188. - Diagrama de secuencia del caso de uso Recuperar contraseña

11.1.12. Diagrama de secuencia **Modificar datos personales**

1. administracionclientes()(El cliente se encuentra en esta url http:// proyecto. tauroyrichard. com /administracion-clientes/?matricula="+matricula;)

2. ExecSQL(sql1) (Carga el nombre del cliente que se ha identificado previamente)

sql1: "select nombre from tx_vehiculo Where matricula='".$matricula."' "

3. new()

4. next()

[Si hay tuplas]

5a. getString ("nombre"):String;

6.close()

7.El cliente pulsa “Modificar datos personales”

8. modificardatoscliente():void (Cargará el formulario correspondiente)

new Ajax.Updater('espacio', 'modificardatosclientes.php',{ asynchronous:false, postBody: 'matricula='+valor });

9.execSQL(sql2) (Selecciona los datos personales de un vehiculo para cargar en el formulario)

sql2: "select * from tx_vehiculo where matricula='".$matricula."' "

10. new()

11. next()

[Si hay tuplas]

12a. getDatos():Datos

13.close()

14.execSQL(sql3) (Carga los datos de las provincias)

sql3: "select cod_prov,nombre from tx_provincias ORDER BY nombre ASC';

15. new()

16. next()

[Si hay tuplas]

17a. getDatos():Datos

18.close()

19.El cliente realiza todos los cambios que desee en el formulario y pulsa un botón

[Si el botón es “Borrar”]

20a.Reset() (El formulario volverá a tener los valores cargados al principio)

[Si el botón es "Cancelar"]

20b. volver()

21b. new() (Irá a esta URL http://proyecto.tauroyrichard.com/administracion-clientes/?matricula="+matricula;)

[Si el botón es “Aceptar”]

20c. obligatoriosmodif():boolean

[Si obligatoriosmodif() es true]

21ca. validaremail():boolean

[Si validar email es true]

22caa. modificarbd():Submit

23caa. Execsql(sql4)(Selecciona el código de la provincia seleccionada)

Sql4: Select cod_prov from tx_provincias where nombre='".$provincia."' "

24caa. new()

25caa. next()

[Si hay tuplas]

26caaa. getInt("cod_prov"): Int;

27caa.close()

28caa. Execsql(sql5) (Actualiza la tabla tx_vehiculo)

Sql5: "UPDATE tx_vehiculo SET nombre = '".$nombre."', DNI = '".$dni."', modelo_vehic = '".$modelo."',telefono = '".$telefono."',email = '".$email."',direccion = '".$direccion."',ciudad = '".$ciudad."' ,codigo_postal = '".$codigo."' ,cod_prov ='".$provincia."',km = '".$km."', password = CASE WHEN '".$pwd."' =' ' THEN password ELSE '".$pwd."' END WHERE matricula ='".$matricula."' ";

[Si el cliente ha rellenado el campo contraseña, es decir, ha cambiado la contraseña]

29caaa. Enviar_email() (Se le envía un email con la nueva contraseña)

30caa. New() (Irá a esta url http://proyecto.tauroyrichard.com/ administracion-clientes/?matricula=$matricula

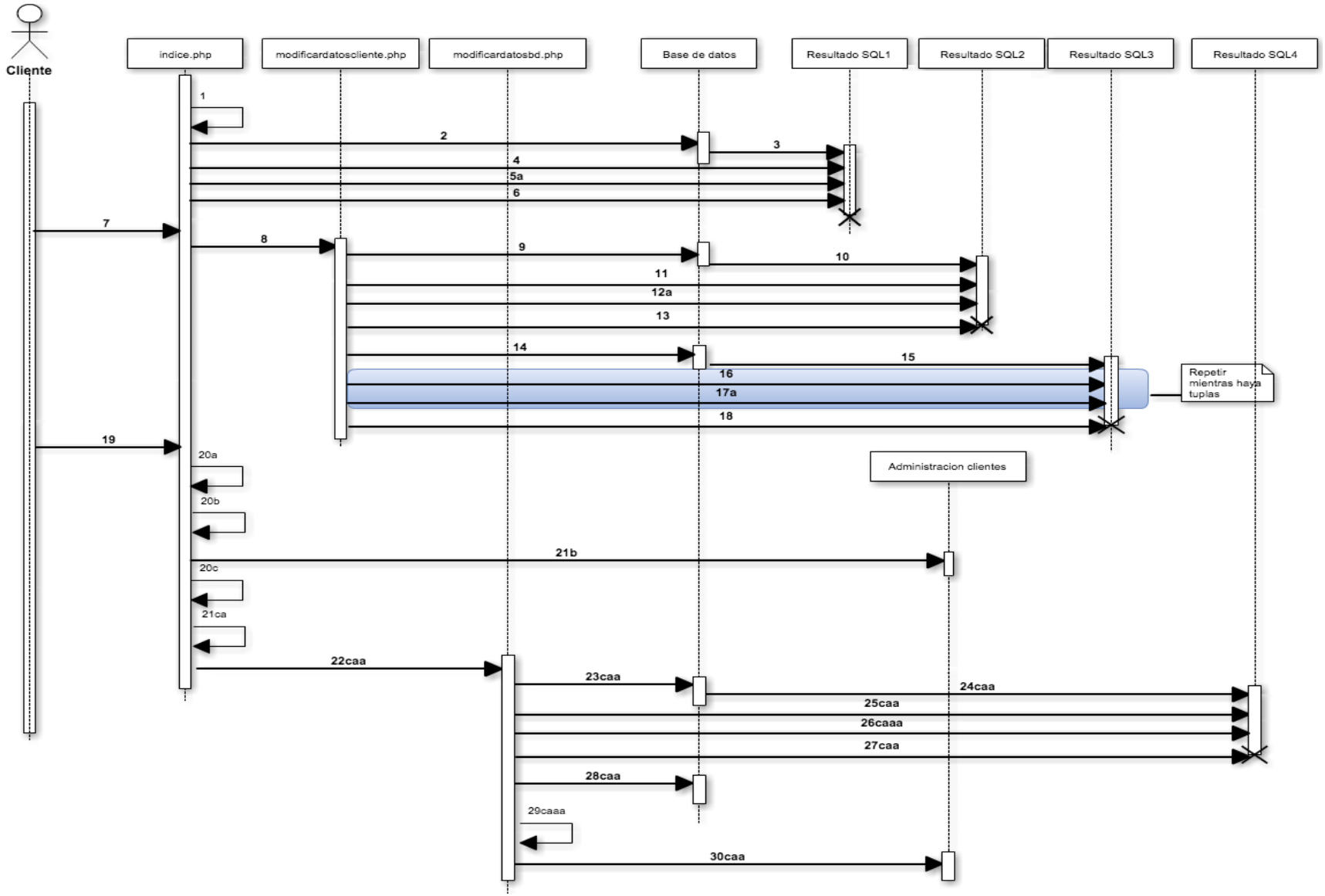


Ilustración 189. - Diagrama de secuencia de caso de uso Modificar datos personales

11.1.13. Diagrama de secuencia de Caso de uso **Ver Historial del Vehículo**

1.administracionclientes()(El cliente se encuentra en esta url http://proyecto.tauroyrichard.com/administracion-clientes/?matricula="+matricula;)

2. ExecSQL(sql1) (Carga el nombre del cliente que se ha identificado previamente)

sql1: "select nombre from tx_vehiculo Where matricula='".$matricula."' "

3. new()

4. next()

[Si hay tuplas]

5a. getString ("nombre"):String;

6.close()

7.El cliente pulsa "Ver historial del vehículo"

8. verhistorialcliente(): void (Cargará todos los datos de las reparaciones del vehículo)

new Ajax.Updater('espacio', ' historialcliente.php',{ asynchronous:false, postBody: 'matricula='+val2 });

9. ExecSQL(sql2) (Carga todos los datos de las reparaciones del vehículo)

sql2:"select tx_vehiculo.nombre,tx_vehiculo.matricula, tx_vehiculo.modelo_vehic, tx_reparacion.fecha, tx_reparacion.cod_rep, tx_reparacion.estado_rep, tx_reparacion.factura, tx_reparacion.importe_total from tx_vehiculo inner join tx_reparacion on tx_vehiculo.matricula = tx_reparacion.matricula where tx_vehiculo.matricula='".$matricula."' ORDER BY tx_reparacion.fecha DESC";

10. new()

11. next()

[Si hay tuplas]

12a. getDatos():Datos

13.close()

[Si el cliente pulsa Volver atrás]

14a. El cliente pulsa "Volver atrás"

15a. volverdehistorial() (): void;
16a. new() (Irá a esta URL http://proyecto.tauroyrichard.com/administracion-clientes/?matricula="+matricula;)

[Si el cliente selecciona una reparación]

14b. El cliente selecciona una reparación
15b. habilitar_boton_verfactura():void (Se habilita el botón ver factura que aparece deshabilitado hasta que se escoge una reparación).

[Si el cliente pulsa el botón Ver factura]

16ba. El cliente pulsa el botón "Ver factura"
17ba. obtenerRadioSeleccionado():int; (Obtiene el código de la reparación seleccionada de la tabla)
18ba. verfacturaonline():Submit;
19ba. execSQL(sql3)
sql3: " select * from tx_vehiculo inner join tx_reparacion on tx_vehiculo.matricula = tx_reparacion.matricula Where tx_reparacion.cod_rep ='".$cod_rep."' "; (Cargar todos los campos de la reparación seleccionada junto con los datos del vehículo)
20ba. new()
21ba. next()

[Si hay tuplas]

22baa.getDatos():Datos

23ba.close()
24ba.execSQL(sql4)
sql4: "Select nombre from tx_provincias where cod_prov='". $row['cod_prov']."' " (Cargar el campo nombre de la tabla tx_provincias, de la provincia seleccionada)
25ba. new()
26ba. next()

[Si hay tuplas]

27baa. getString ("nombre"):String;

28ba.close()
29ba. execSQL(sql5)
sql5: "select titulo,descripcion,precio from tx_servicios inner join tx_actuaciones on tx_actuaciones.codserv=tx_servicios.codserv Where tx_actuaciones.cod_rep ='".$cod_rep."' ORDER BY titulo ASC " (Cargar todos los campos de los servicios de la tabla tx_actuaciones)
30ba. new()
31ba. next()

[Si hay tuplas]

32baa. getDatos():Datos

33ba.close()
34ba. Crear_factura_pdf(); (Generará la factura pdf online)

35b. Ver_si_factura_esta_finalizada():boolean (Se comprueba si la reparación seleccionada está finalizada)

new Ajax.Updater('finaliz', 'comprobar_finalizada.php',{ asynchronous:false, postBody: 'codrep='+codrep});

36b.ExecSQL(sql6) (comprueba el estado de la reparación del vehiculo)

sql6: " SELECT estado_rep FROM tx_reparacion WHERE cod_rep='".$cod_rep."' ";

37b. new()

38b. next()

[Si hay tuplas]

39ba. getDatos():Datos

40b.close()

[Si Ver_si_factura_esta_finalizada() devuelve true]

41ba. ver_si_factura_esta_pagada():boolean

new Ajax.Updater('deshabilitarb', 'comprobar_pago_factura.php',{ asynchronous:false, postBody: 'codrep='+codrep});

42ba.ExecSQL(sql7) (comprueba si la factura de la reparación del vehículo está pagada)

sql7: "SELECT factura FROM tx_reparacion WHERE cod_rep='".$cod_rep."' ";

43ba. new()

44ba. next()

[Si hay tuplas]

45baa. getDatos():Datos

46ba. close()

[Si ver_si_factura_esta_pagada() es false]

47baa. habilitar_boton_pagarahora():void (Se habilita el pagar ahora).

[Si el cliente pulsa el botón Pagar ahora]

48baaa. El cliente pulsa el botón "Pagar ahora"

49baaa. obtenerRadioSeleccionado():int; (Obtiene el código de la reparación seleccionada de la tabla)

50baaa. pagarahora():void

new Ajax.Updater('espacio', 'pagar.php',{ asynchronous:false, postBody: 'cod_rep='+val1});

51baaa. execSQL(sql8)

sql8: "SELECT importe_total,matricula FROM tx_reparacion WHERE cod_rep='".$cod_rep."' "; (Obtiene el importe de la factura de la base de datos)

52baaa. new()

53baaa. next()

[Si hay tuplas]

54baaaa. getDatos():Datos

55baaa.close()

56baaa. Cargar factura(); (Carga la factura como lo realiza en el archivo verfactura.php). [Ver sección "Ver factura online" de este documento]
57baaa. El cliente pulsa un botón
[Si el cliente pulsa "Descargar factura"]
58baaaa. Descargarfactura():void(Realiza lo mismo que el fichero verfactura.php)
[Ver sección "Ver factura online"]
[Si el cliente pulsa Volver atrás]
58baaab. volveralhistorial(): void;
new Ajax.Updater('espacio', 'historialcliente.php',{ asynchronous:false, postBody: 'matricula='+val2});
[Si el cliente pulsa "Transferencia bancaria"]
58baaac. transferenciabancaria() (Se redirige al cliente a la pantalla de agradecimiento de pago a través de transferencia bancaria)
new Ajax.Updater('espacio', transferencia.php',{ asynchronous:false, postBody: 'codrep='+codrep });
59baaac. execSQL(sql9)
sql9:"SELECT tx_vehiculo.nombre, tx_vehiculo.matricula , tx_reparacion. importe_total, tx_reparacion.fecha, tx_vehiculo.email FROM tx_vehiculo inner join tx_reparacion on tx_vehiculo.matricula =tx_reparacion.matricula WHERE tx_reparacion.cod_rep='".$cod_rep."' ";
(Obtiene los datos para rellenar la tabla de la página "Gracias por confiar en nosotros" y el email)
60baaac. new()
61baaac. next()
[Si hay tuplas]
62baaaca. getDatos():Datos
63baaac.close()
64baaac. Generarfactura():void (Realiza lo mismo que el fichero verfactura.php) (Genera la factura para enviársela por email al cliente)
[Ver sección "Ver factura online"]
65baaac. Enviar_email_cliente():void (El email de confirmación de pago de la factura)
66baaac. Enviar_email_administrador():void (El email de que una factura ha sido pagada)
67baaac. execSQL(sql10)
sql10:"UPDATE tx_reparacion SET factura='Pagada' WHERE cod_rep='".$cod_rep."' " (Se actualiza el valor de factura en la base de datos)
[Si el cliente pulsa el botón "Descargar factura"]

68baaaca. Descargarfactura():void (Realiza lo mismo que el fichero verfactura.php)

[Ver sección "Ver factura online"]

[Si el cliente pulsa el botón "Volver atrás"]

68baaacb. volveralhistorial(): void;

new Ajax.Updater('espacio', 'historialcliente.php',{ asynchronous:false, postBody: 'matricula='+val2});

[Si el cliente pulsa "Paypal"]

58baaad. pagopaypal()

new Ajax.Updater('espacio', 'actualizardatospaypal.php',{ asynchronous:false, postBody: 'cod_rep='+val1 });

59baaad. execSQL(sql11)

sql11:"UPDATE tx_reparacion SET factura='Pagada' WHERE cod_rep='".$cod_rep."' " (Se actualiza el valor de factura en la base de datos)

60baaad. execSQL(sql12)

sql12:"SELECT tx_vehiculo.nombre,tx_vehiculo.matricula,tx_reparacion.importe_ total ,tx_reparacion.fecha, tx_vehiculo.emailFROM tx_vehiculo inner join tx_reparacion on tx_vehiculo.matricula=tx_reparacion.matricula WHERE tx_reparacion.cod_rep='".$cod_rep."' ";

(Obtiene los datos para insertarlos en el email).

61baaad. new()

62 baaad. next()

[Si hay tuplas]

63baaada. getDatos():Datos

64 baaad.close()

65baaad. Generarfactura():void (Se realizará de igual manera que el archivo verfactura.php, genera la factura para enviársela por email al cliente)

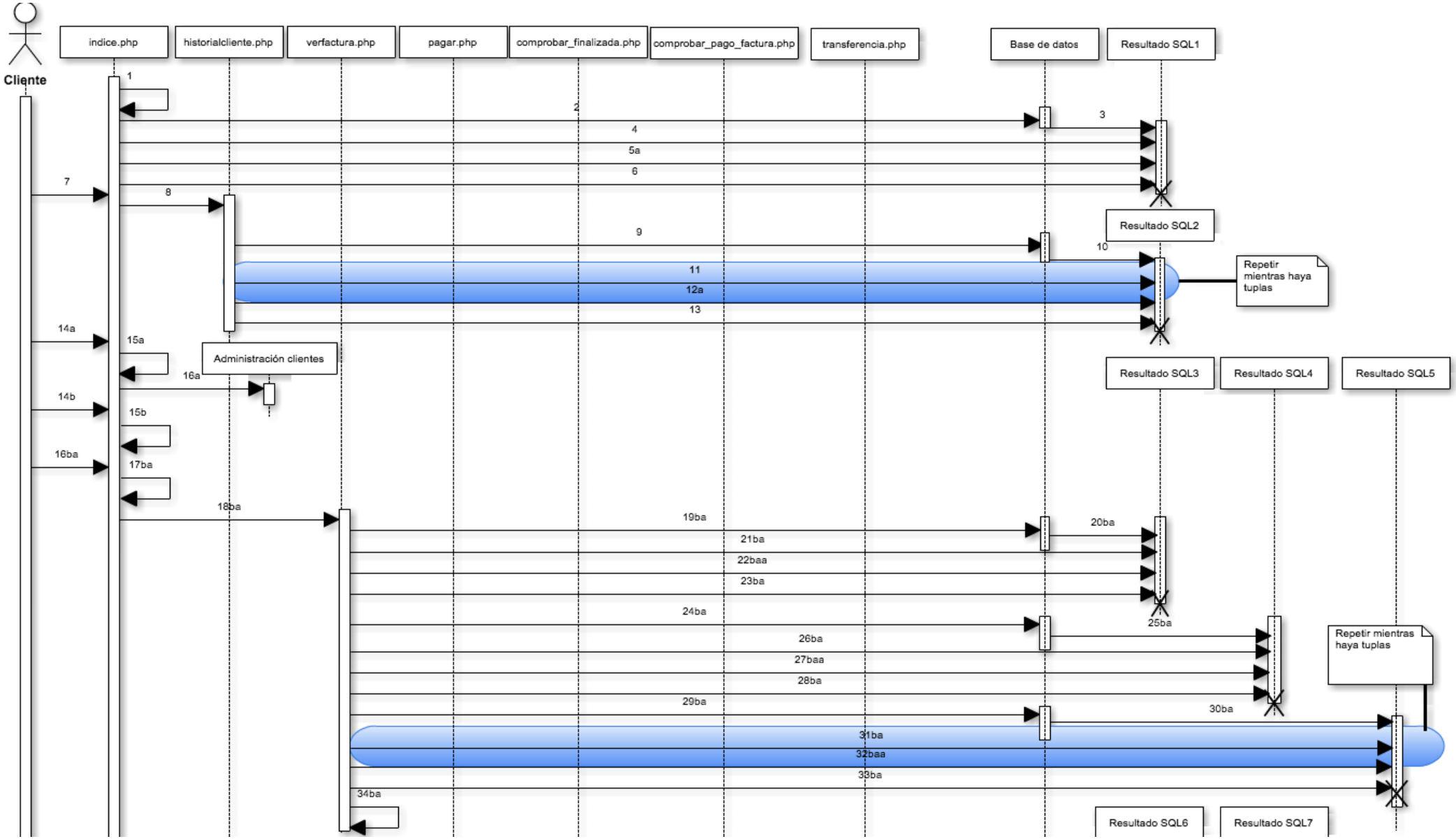
[Ver sección "Ver factura online"]

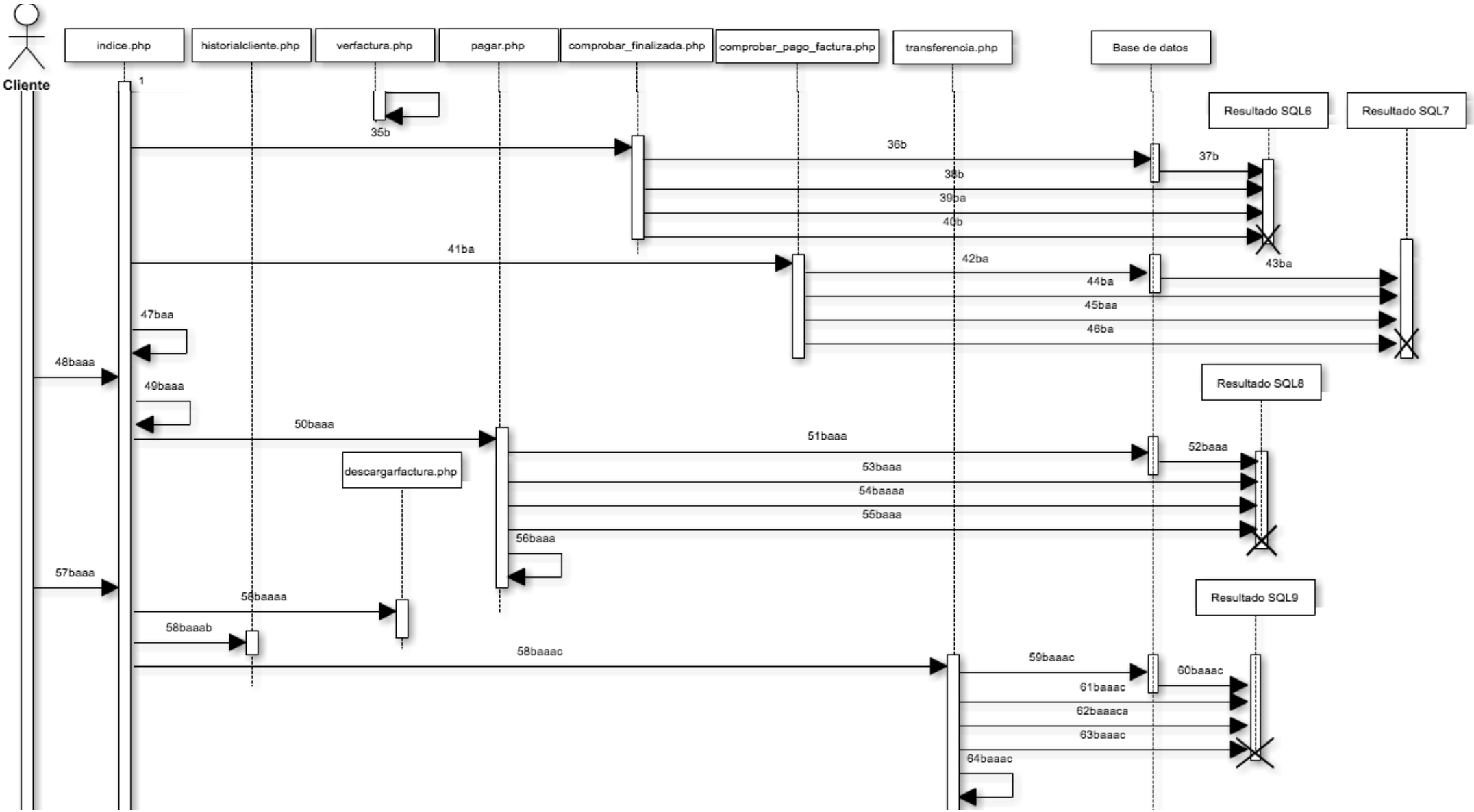
66baaad. Enviar_email_cliente():void (El email de confirmación de pago de la factura)

67baaad. Enviar_email_administrador():void (El email de que una factura ha sido pagada)

68baaad. Submit(); (Le redirigirá al cliente a la página de PayPal para proceder al pago). (Irá a esta url https://www.sandbox.paypal.com/cgi-bin/webscr)

69baaad. New() (Le redirigirá al cliente a la página de agradecimiento de la aplicación web, una vez finalizado el pago).(Irá a esta url http://proyecto.tauroyrichard.com/gracias-por-confiar-en-nosotros/)


Cliente
indice.php
historialcliente.php
verfactura.php
pagar.php
comprobar_finalizada.php
comprobar_pago_factura.php
transferencia.php
Base de datos
Resultado SQL1
Resultado SQL2
Resultado SQL3
Resultado SQL4
Resultado SQL5
Resultado SQL6
Resultado SQL7
Administración clientes
Repetir mientras haya tuplas
Repetir mientras haya tuplas
1
2
3
4
5a
6
7
8
9
10
11
12a
13
14a
15a
16a
14b
15b
16ba
17ba
18ba
19ba
20ba
21ba
22baa
23ba
24ba
25ba
26ba
27baa
28ba
29ba
30ba
31ba
32baa
33ba
34ba


Cliente
indice.php
historialcliente.php
verfactura.php
pagar.php
comprobar_finalizada.php
comprobar_pago_factura.php
transferencia.php
Base de datos
Resultado SQL6
Resultado SQL7
Resultado SQL8
Resultado SQL9
descargarfactura.php
1
35b
36b
37b
38b
39ba
40b
41ba
42ba
43ba
44ba
45baa
46ba
47baa
48baaa
49baaa
50baaa
51baaa
52baaa
53baaa
54baaaa
55baaa
56baaa
57baaa
58baaaa
58baaab
58baaac
59baaac
60baaac
61baaac
62baaaca
63baaac
64baaac

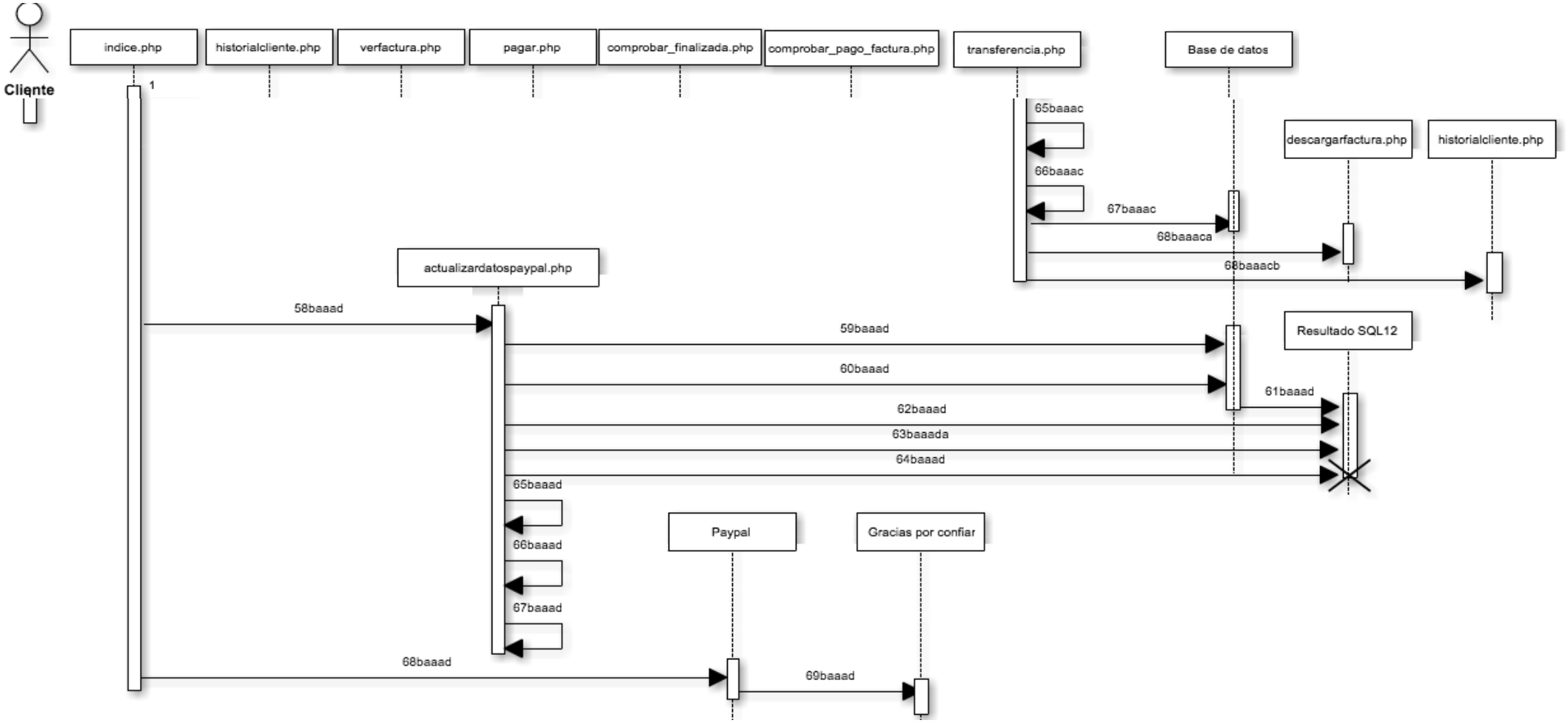


Ilustración 190. - Diagrama de secuencia de caso de uso Ver historial del vehículo

11.1.14. Diagrama de secuencia de caso de uso **Pagar mis facturas**

1.administracionclientes()(El cliente se encuentra en esta url http://proyecto.tauroyrichard.com/administracion-clientes/?matricula="+matricula;)

2. ExecSQL(sql1) (Carga el nombre del cliente que se ha identificado previamente)

sql1: "select nombre from tx_vehiculo Where matricula='".$matricula."' "

3. new()

4. next()

[Si hay tuplas]

5a. getString ("nombre"):String;

6.close()

7.El cliente pulsa "Pagar mis facturas"

8. pagarmisfacturas():void (Cargará los datos de las reparaciones que faltan por pagar)

new Ajax.Updater('espacio','historial_facturas_sinpagar.php',{asynchronous:false, postBody: 'matricula='+val2 });

9. ExecSQL(sql2) (Carga los datos de las reparaciones del vehículo cuya factura no está pagada)

sql2:"select tx_vehiculo.nombre,tx_vehiculo.matricula,tx_vehiculo.modelo_vehic,tx_reparación.fecha,tx_reparacion.cod_rep,tx_reparacion.estado_rep, tx_reparacion.factura, tx_reparacion.importe_total from tx_vehiculo inner join tx_reparacion on tx_vehiculo.matricula =tx_reparacion.matricula where tx_vehiculo.matricula= '".$matricula."' and tx_reparacion.factura='Sin pagar' ORDER BY tx_reparacion.fecha DESC";

10. new()

11. next()

[Si hay tuplas]

12a. getDatos():Datos

13.close()

[Si el cliente pulsa Volver atrás]

14a. El cliente pulsa "Volver atrás"
15a. volverdehistorial() (): void;
16a. new() (Irá a esta URL http://proyecto.tauroyrichard.com/administracion-clientes/?matricula="+matricula;)

[Si el cliente selecciona una reparación]

14b. El cliente selecciona una reparación
15b. habilitar_boton_verfactura():void (Se habilita el botón ver factura que aparece deshabilitado hasta que se escoge una reparación).

[Si el cliente pulsa el botón Ver factura]

16ba. El cliente pulsa el botón "Ver factura"
17ba. obtenerRadioSeleccionado():int; (Obtiene el código de la reparación seleccionada de la tabla)
18ba. verfacturaonline():Submit;
19ba. execSQL(sql3)
sql3: " select * from tx_vehiculo inner join tx_reparacion on tx_vehiculo.matricula = tx_reparacion.matricula Where tx_reparacion.cod_rep ='".$cod_rep."' "; (Cargar todos los campos de la reparación seleccionada junto con los datos del vehículo)
20ba. new()
21ba. next()

[Si hay tuplas]

22baa.getDatos():Datos

23ba.close()
24ba.execSQL(sql4)
sql4: "Select nombre from tx_provincias where cod_prov='". $row['cod_prov']."' "
(Cargar el campo nombre de la tabla tx_provincias, de la provincia seleccionada)
25ba. new()
26ba. next()

[Si hay tuplas]

27baa. getString ("nombre"):String;

28ba.close()
29ba. execSQL(sql5)
sql5: "select titulo,descripcion,precio from tx_servicios inner join tx_actuaciones on tx_actuaciones.codserv = tx_servicios.codserv Where tx_actuaciones.cod_rep='".$cod_rep."' ORDER BY titulo ASC " (Cargar todos los campos de los servicios de la tabla tx_actuaciones)
30ba. new()

31ba. next()
[Si hay tuplas]
32baa. getDatos():Datos
33ba.close()
34ba. Crear_factura_pdf(); (Generará la factura pdf online)
35b. ver_si_factura_esta_pagada():boolean
new Ajax.Updater('deshabilitarb', 'comprobar_pago_factura.php',{ asynchronous:false, postBody: 'codrep='+codrep});
36b. execSQL(sql6) (comprueba si la factura de la reparación del vehículo está pagada)
sql6: "SELECT factura FROM tx_reparacion WHERE cod_rep='".$cod_rep."' ";
37b. new()
38b. next()
[Si hay tuplas]
39ba. getDatos():Datos
40b. close()
[Si ver_si_factura_esta_pagada() es false]
41ba. habilitar_boton_pagarahora():void (Se habilita el pagar ahora).
[Si el cliente pulsa el botón Pagar ahora]
42baa. El cliente pulsa el botón "Pagar ahora"
43baa. obtenerRadioSeleccionado():int; (Obtiene el código de la reparación seleccionada de la tabla)
44baa. pagarahora_historialsinpagar():void
new Ajax.Updater('espacio', 'pagar_historialsinpagar.php' ,{ asynchronous:false, postBody: 'cod_rep='+val1});
45baa. execSQL(sql7)
sql7: "SELECT importe_total,matricula FROM tx_reparacion WHERE cod_rep='".$cod_rep."' "; (Obtiene el importe de la factura de la base de datos)
46baa. new()
47baa. next()
[Si hay tuplas]
48baaa. getDatos():Datos
49baa.close()
50baa. Cargar factura(); (Carga la factura como lo realiza en el archivo verfactura.php). [Ver sección "Ver factura online" de este documento]
51baa. El cliente pulsa un botón
[Si el cliente pulsa "Descargar factura"]
52baaa. Descargarfactura():void(Realiza lo mismo que el fichero verfactura.php) [Ver sección "Ver factura online"]
[Si el cliente pulsa Volver atrás]
53baab.volveralhistorialsinpagar() :void;

new Ajax.Updater('espacio', 'historial_facturas_sinpagar.php ',{
asynchronous:false,postBody: 'matricula='+val2});

[Si el cliente pulsa "Transferencia bancaria"]

53baac. transferencia_historialsinpagar():void (Se redirige al cliente a la pantalla de agradecimiento de pago a través de transferencia bancaria)

new Ajax.Updater('espacio','transferencia_historialsinpagar.php',{asynchronous:false,postBody: 'codrep='+codrep });

54baac. execSQL(sql8)

sql8:"SELECT tx_vehiculo.nombre,tx_vehiculo.matricula,tx_reparacion.importe_total, tx_reparacion.fecha, tx_vehiculo.email FROM tx_vehiculo inner join tx_reparacion on tx_vehiculo.matricula=tx_reparacion.matricula WHERE tx_reparacion.cod_rep='".$cod_rep."' ";

(Obtiene los datos para rellenar la tabla de la página "Gracias por confiar en nosotros" y el email)

55baac. new()

56baac. next()

[Si hay tuplas]

57baaca. getDatos():Datos

58baac.close()

59baac. Generarfactura():void (Realiza lo mismo que el fichero verfactura.php) (Genera la factura para enviársela por email al cliente)

[Ver sección "Ver factura online"]

60baac. Enviar_email_cliente():void (El email de confirmación de pago de la factura)

61baac. Enviar_email_administrador():void (El email de que una factura ha sido pagada)

62baac. execSQL(sql9)

sql9: " UPDATE tx_reparacion SET factura='Pagada' WHERE cod_rep='".$cod_rep."' " (Se actualiza el valor de factura en la base de datos)

[Si el administrador pulsa el botón "Descargar factura"]

63baaca. descargarfacturatransferencia(): void (Realiza lo mismo que el fichero verfactura.php)

[Ver sección "Ver factura online"]

[Si el administrador pulsa el botón "Volver atrás"]

63baacb. volveralhistorialsinpagar():void;

new Ajax.Updater('espacio', 'historial_facturas_sinpagar.php',{
asynchronous:false, postBody: 'matricula='+val2});

[Si el cliente pulsa "Paypal"]

53baad. pagopaypal()

new Ajax.Updater('espacio','actualizardatospaypal.php',{asynchronous:false, PostBody: 'cod_rep='+val1 });

54baad. execSQL(sql10)

sql10: " UPDATE tx_reparacion SET factura='Pagada' WHERE cod_rep='".$cod_rep."' " (Se actualiza el valor de factura en la base de datos)

55baad. execSQL(sql11)

sql11:"SELECT tx_vehiculo.nombre, tx_vehiculo.matricula, tx_reparacion. importe_total,tx_reparacion.fecha, tx_vehiculo.email FROM tx_vehiculo inner join tx_reparacion on tx_vehiculo.matricula=tx_reparacion.matricula WHERE tx_reparacion.cod_rep='".$cod_rep."' ";

(Obtiene los datos para insertarlos en el email).

56baad. new()

57baad. next()

[Si hay tuplas]

58baada. getDatos():Datos

59baad.close()

60baad. Generarfactura():void (Se realizará de igual manera que el archivo verfactura.php, genera la factura para enviársela por email al cliente)

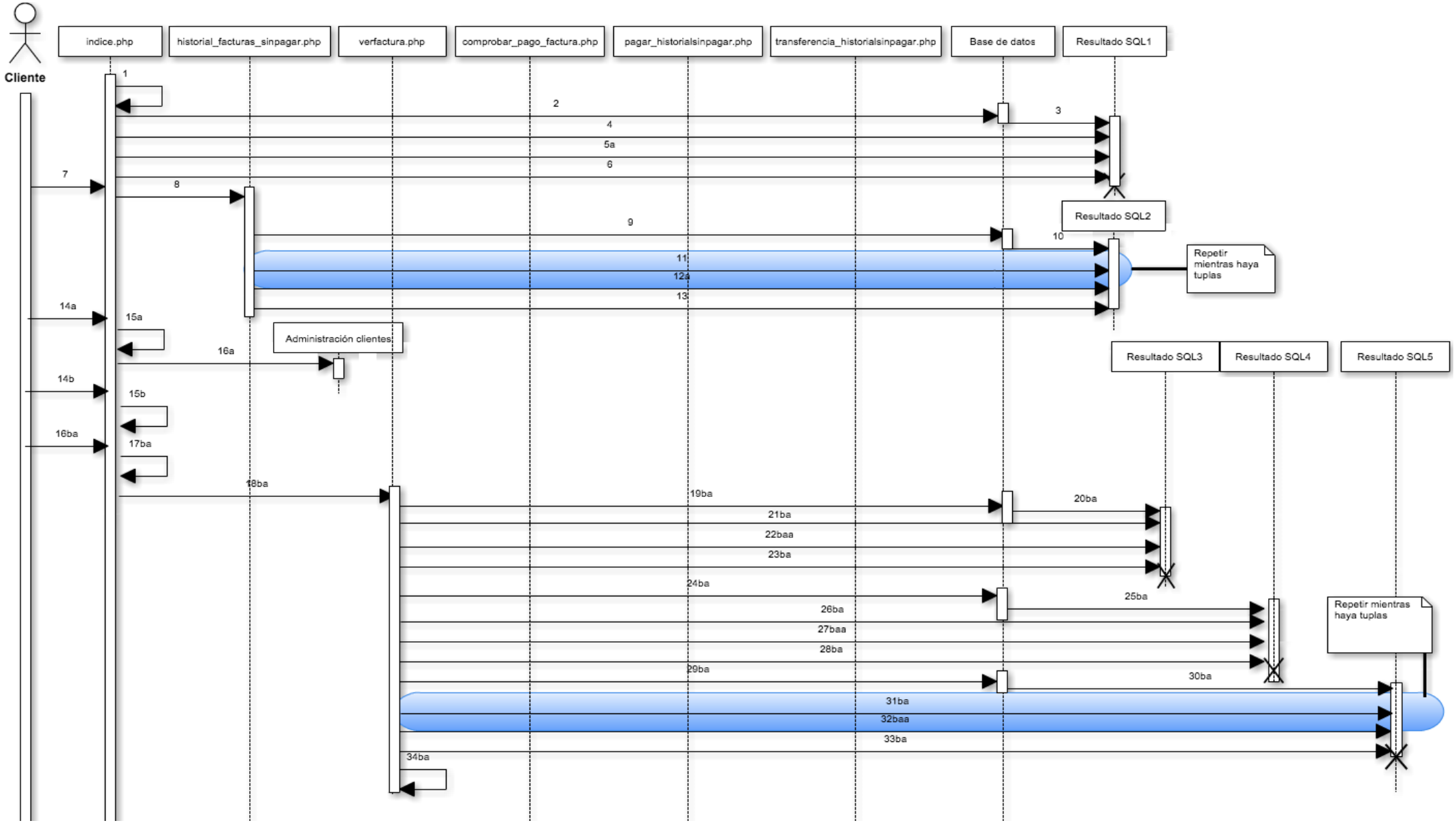
[Ver sección "Ver factura online"]

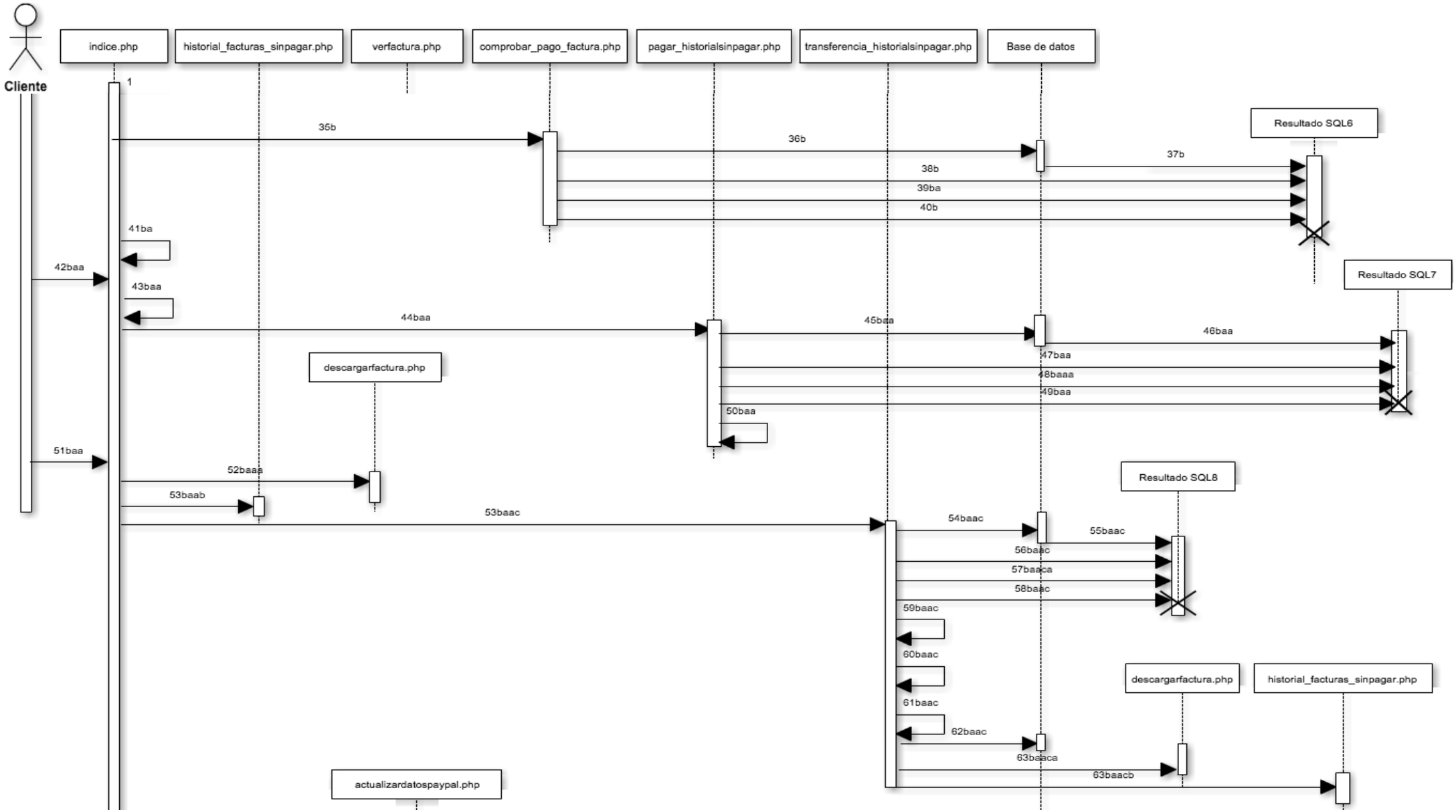
61baad. Enviar_email_cliente():void (El email de confirmación de pago de la factura)

62baad. Enviar_email_administrador():void (El email de que una factura ha sido pagada)

63baad. Submit(); (Le redirigirá al cliente a la página de PayPal para proceder al pago). (Irá a esta url https://www.sandbox.paypal.com/cgi-bin/webscr)

64baad. New() (Le redirigirá al cliente a la página de agradecimiento de la aplicación web, una vez finalizado el pago). (Irá a esta url http://proyecto.tauroyrichard.com/gracias-por-confiar-en-nosotros/)


Cliente
indice.php
historial_facturas_sinpagar.php
verfactura.php
comprobar_pago_factura.php
pagar_historialsinpagar.php
transferencia_historialsinpagar.php
Base de datos
Resultado SQL1
Resultado SQL2
Resultado SQL3
Resultado SQL4
Resultado SQL5
Administración clientes
Repetir mientras haya tuplas
Repetir mientras haya tuplas
1
2
3
4
5a
6
7
8
9
10
11
12a
13
14a
15a
16a
14b
15b
16ba
17ba
18ba
19ba
20ba
21ba
22baa
23ba
24ba
25ba
26ba
27baa
28ba
29ba
30ba
31ba
32baa
33ba
34ba


Cliente
indice.php
historial_facturas_sinpagar.php
verfactura.php
comprobar_pago_factura.php
pagar_historialsinpagar.php
transferencia_historialsinpagar.php
Base de datos
Resultado SQL6
Resultado SQL7
Resultado SQL8
descargarfactura.php
actualizardatospaypal.php
historial_facturas_sinpagar.php
1
35b
36b
37b
38b
39ba
40b
41ba
42baa
43baa
44baa
45baa
46baa
47baa
48baaa
49baa
50baa
51baa
52baaa
53baab
53baac
54baac
55baac
56baac
57baaca
58baac
59baac
60baac
61baac
62baac
63baaca
63baacb

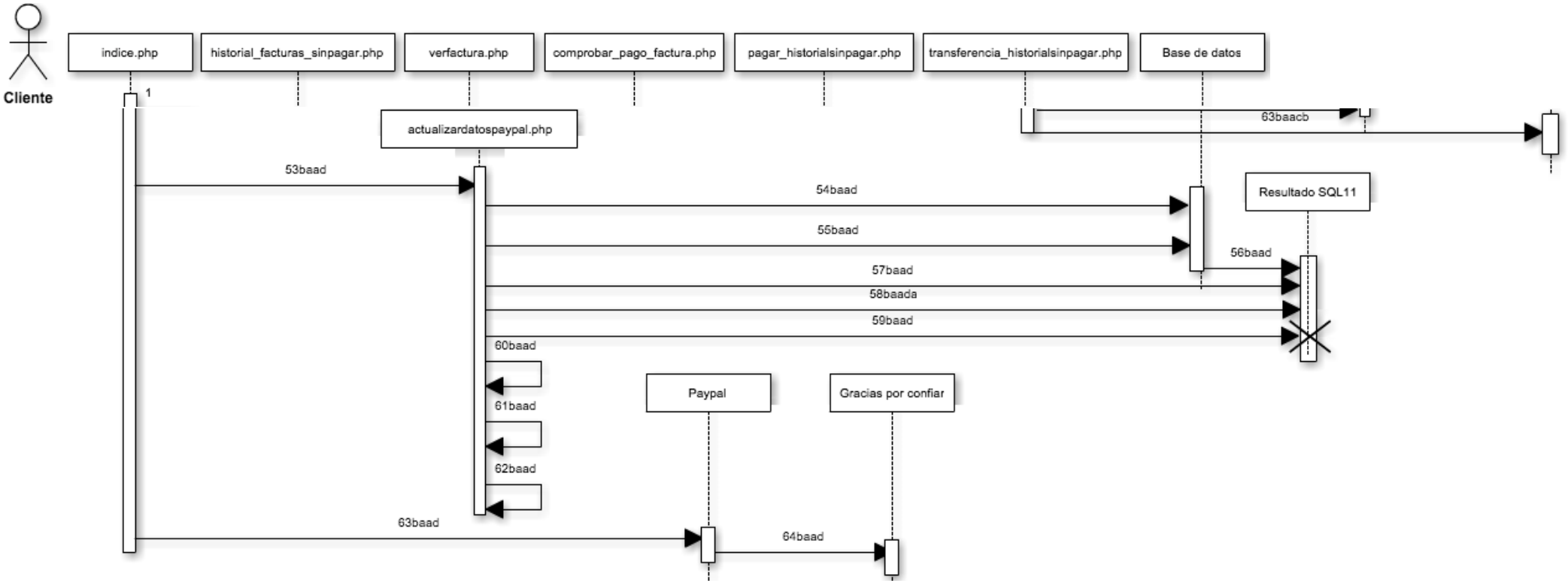


Ilustración 191. - Diagrama de secuencia de caso de uso Pagar mis facturas

11.2. Aplicación móvil

Seguidamente, se muestran los diagramas de secuencia de la aplicación móvil junto con las explicaciones correspondientes de cada uno de ellos. Estos diagramas corresponden a los casos de uso de la aplicación móvil, especificados en el apartado anterior.

11.2.1. Diagrama de secuencia de Caso de uso **Identificación**

1. El cliente rellena los campos DNI y Contraseña y pulsa "Aceptar"

2. httpGetData("identificarseregistrados.php?dni="+dni+"&pwd="+pwd):String;

3. execSQL(sql1)

sql1: " select nombre,matricula from tx_vehiculo Where DNI='".$dni."' and password='".$pwd."'"(Selecciona el nombre y la matricula de la persona cuyo dni y pwd son los introducidos)

4.new()

5. next()

[Si hay tuplas]

6a. getDatos ():Datos;

7.close()

[Si la consulta devuelve la matricula y el nombre]

8a. checkPlayServices():boolean
[Si checkPlayServices() devuelve true]
9aa. getAppVersion(context):int
10aa. getRegistrationId(context):String;
[Si getRegistrationId devuelve un ID de registro]
11aaa. enviarDispositivoYaRegistrados():void
12aaa. New(nombre, matricula)
[Si no devuelve nada]
11aab. New()
12aab. GCM.register(SENDER_ID): String;

13aab. setRegistrationId(context, params[0], regid):void;

14aab. getAppVersion(context): int

15aab. registrardispositivo(regid):void

16aab. httpGetData("introduciridgcm.php?dni= "+dni+"&regid="+regid):void;

17aab. execSQL(sql2) (Insertamos el id de registro que ha devuelto el GCM en la tabla tx_app)

sql2: " INSERT INTO tx_app (DNI,IdGCM) VALUES('$dni','$regid')"

18aab. New(nombre, matricula)

Los pasos siguientes se realizan tanto después de la instrucción 13aab como 18aab

19aa. httpGetData("comprobarreparacion.php?matricula="+matricula):String; (El usuario se encuentra en el menú de la App y cada vez que el cliente se encuentra en el menú se comprueba con la base de datos si la reparación está finalizada, para mostrarle el aviso en el menú).

20aa. execSQL(sql3)

sql3: "select estado_rep,factura,fecha from tx_reparacion Where matricula='".$matricula."' ORDER BY fecha DESC";

Selecciona el estado de la reparación y el valor de factura de todas las reparaciones ordenadas por fecha, para poder mirar esos datos de la última reparación del vehículo.

21aa. new()

22aa. next()

[Si hay tuplas]

23aaa. getDatos ():Datos;

24aa.close()

25aa. mostrarMensaje(Mensaje); (Muestra diferentes mensajes en función de lo que devuelva la consulta) EJ: ("No tienes ningún vehículo en reparación", "La reparación de tu vehículo ya está finalizada, puedes pasar a recogerlo" o "La reparación de tu vehículo todavía no está finalizada")

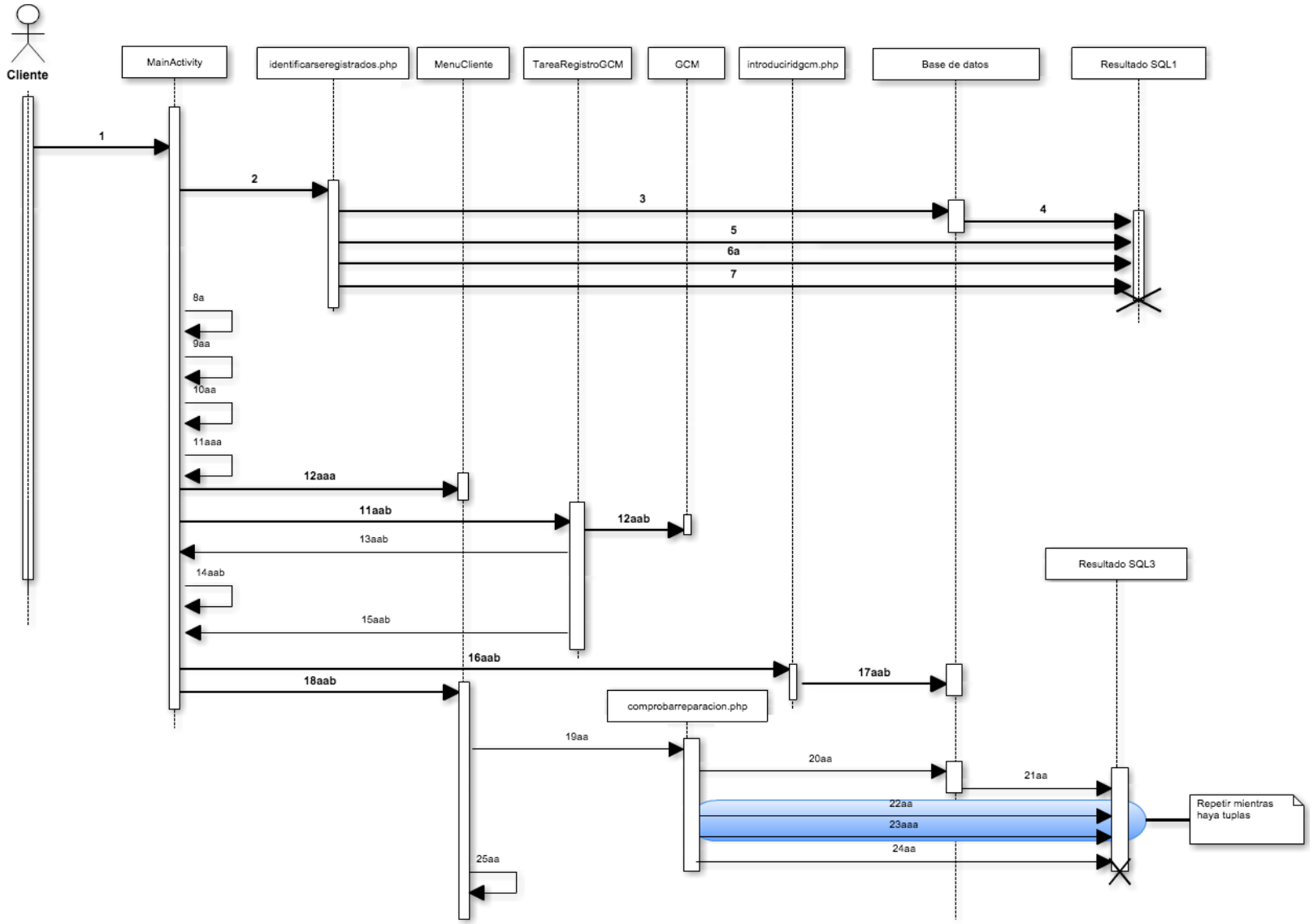


Ilustración 192. - Diagrama de secuencia App - Identificación

11.2.2. Diagrama de secuencia de caso de uso **Pedir Cita**

1. El cliente pulsa el botón "Pedir cita"

2. new(nombre, matrícula)

[Si el cliente pulsa "Volver atras"]

3a. El cliente pulsa "Volver atrás"
4a. finish();
5a. httpGetData("comprobarreparacion.php?matricula="+matricula):String; (El usuario se encuentra en el menú de la App y cada vez que el cliente se encuentra en el menú se comprueba con la base de datos si la reparación está finalizada, para mostrarle el aviso en el menú).
6a. execSQL(sql1)
sql1:"select estado_rep,factura,fecha from tx_reparacion Where matricula ='".$matricula."' ORDER BY fecha DESC";
(Selecciona el estado de la reparación y el valor de factura de todas las reparaciones ordenadas por fecha, para poder mirar esos datos de la última reparación del vehículo.)
7a.new()
8a next()
[Si hay tuplas]
9aa. getDatos ():Datos;
10a.close()
11a. mostrarMensaje(Mensaje); (Muestra diferentes mensajes en función de lo que devuelva la consulta) EJ: ("No tienes ningún vehículo en reparación", "La reparación de tu vehículo ya está finalizada, puedes pasar a recogerlo" o "La reparación de tu vehículo todavía no está finalizada")

[Si el cliente pulsa "Mostrar calendario"]

3b. El cliente pulsa "Mostrar calendario"
4b. mostrarCalendario():Dialog; (Muestra la ventana emergente del calendario)
5b. El cliente escoge una fecha
6b. modificarCasillaCalendario():void; (Modifica la casilla de la fecha del calendario por la fecha escogida por el cliente).
[Si el cliente pulsa enviar]
7ba. El cliente pulsa enviar
8ba. httpGetData(" enviarmail.php?dia= "+mDateDisplay.getText() +"&contenido="+ texto+"&matricula="+matricula):String; (Llama a la función php para el envío del email)
9ba. execSQL(sql2)
sql2: " select nombre,email from tx_vehiculo Where matricula='".$matricula."'";

(Selecciona el nombre y el email del cliente dueño del vehículo)
10ba.new()
11ba.next()
[Si hay tuplas]
12baa. getDatos ():Datos;
13ba.close()
14ba. Enviaremail(); (Enviará un email a los administradores del taller)
15ba. finish();
16ba. httpGetData("comprobarreparacion.php?matricula="+matricula):String; (El usuario se encuentra en el menú de la App y cada vez que el cliente se encuentra en el menú se comprueba con la base de datos si la reparación está finalizada, para mostrarle el aviso en el menú).
17ba. execSQL(sql3)
sql3: "select estado_rep,factura,fecha from tx_reparacion Where matricula='".$matricula."' ORDER BY fecha DESC";
(Selecciona el estado de la reparación y el valor de factura de todas las reparaciones ordenadas por fecha, para poder mirar esos datos de la última reparación del vehículo.)
18ba.new()
19ba next()
[Si hay tuplas]
20baa. getDatos ():Datos;
21ba.close()
22ba. mostrarMensaje(Mensaje); (Muestra diferentes mensajes en función de lo que devuelva la consulta) EJ: ("No tienes ningún vehículo en reparación", "La reparación de tu vehículo ya está finalizada, puedes pasar a recogerlo" o "La reparación de tu vehículo todavía no está finalizada")

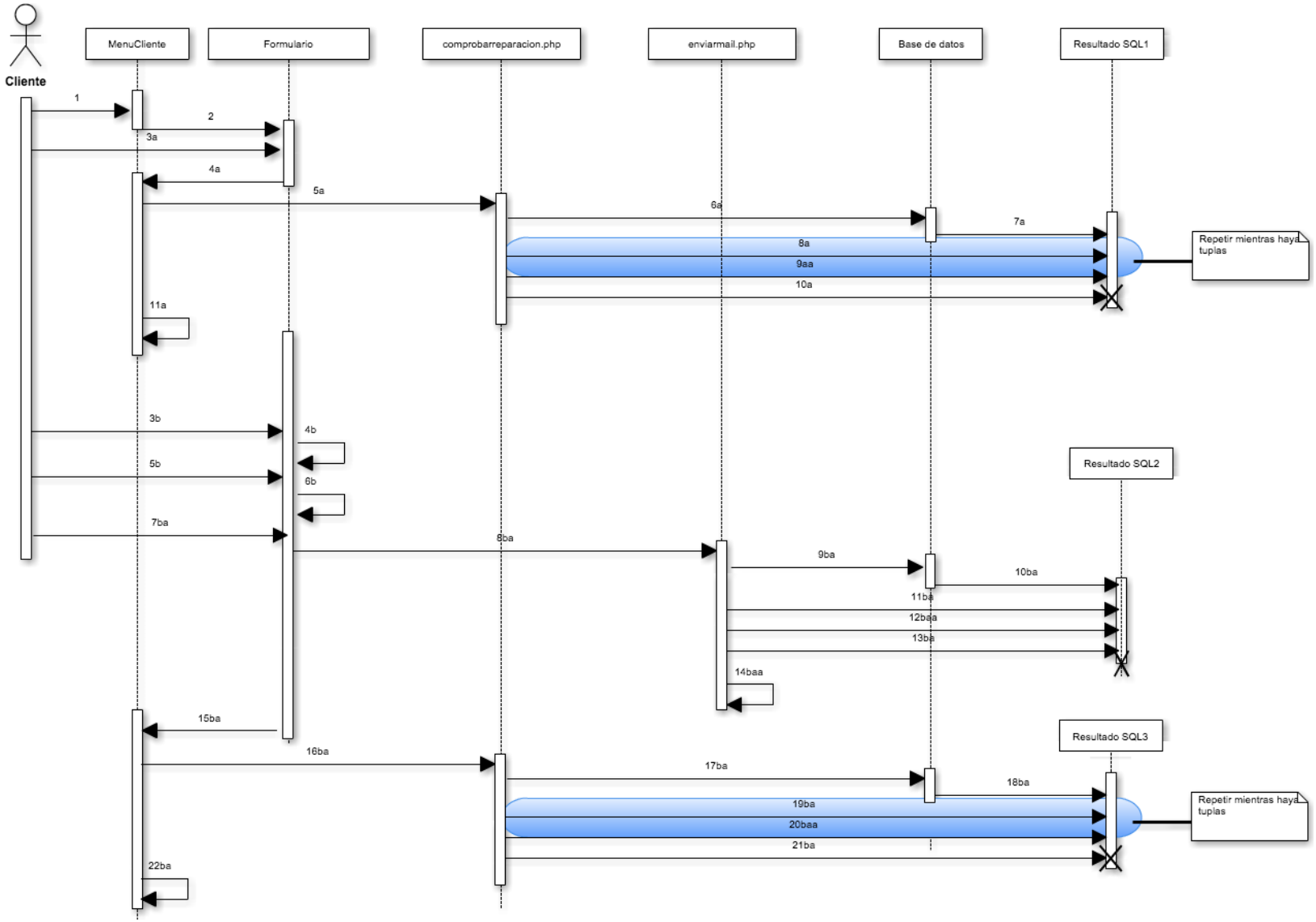


Ilustración 193. - Diagrama de secuencia de caso de uso Pedir Cita

11.2.3. Diagrama de secuencia de Caso de uso **Revisiones**

1. El cliente pulsa el botón "Revisiones"

2. new(nombre, matrícula);

3. httpGetData("revisiones.php?matricula="+matricula): Datosrevisiones;

4. execSQL(sql1)

sql1"select * from tx_avisos Where matricula='".$matricula."' ORDER BY codAviso ASC";

(Selecciona todos los avisos del vehículo correspondiente.)

5.new()

6 next()

[Si hay tuplas]

7a. getDatos ():Datos;

8.close()

9. execSQL(sql2)

sql2:" select descripcion from tx_avisosGenericos Where codAviso='".$row['codAviso']."' ";

(Selecciona la descripción en la tabla tx_avisosGenericos del aviso indicado.)

10.new()

11. next()

[Si hay tuplas]

12a. getDatos ():Datos

13.close();

[Si devuelve algún dato la llamada a revisiones.php]

14a. agregarCabecera():void;

15a.agregarFilasTabla(DatosRevisiones):void;

(Con los dos métodos anteriores se crea la tabla de las revisiones)

[Si el cliente pulsa "Volver atras"]

16a. El cliente pulsa "Volver atrás"

17a. finish();
18a. httpGetData("comprobarreparacion.php?matricula="+matricula): Datos; (El usuario se encuentra en el menú de la App y cada vez que el cliente se encuentra en el menú se comprueba con la base de datos si la reparación está finalizada, para mostrarle el aviso en el menú).
19a. execSQL(sql3)
sql3: "select estado_rep,factura,fecha from tx_reparacion Where matricula ='".$matricula."' ORDER BY fecha DESC";
Selecciona el estado de la reparación y el valor de factura de todas las reparaciones ordenadas por fecha, para poder mirar esos datos de la última reparación del vehículo.
20a.new()
21a next()
[Si hay tuplas]
    22aa. getDatos ():Datos;
23a.close()
24a. mostrarMensaje(Mensaje); (Muestra diferentes mensajes en función de lo que devuelva la consulta) EJ: ("No tienes ningún vehículo en reparación", "La reparación de tu vehículo ya está finalizada, puedes pasar a recogerlo" o "La reparación de tu vehículo todavía no está finalizada").

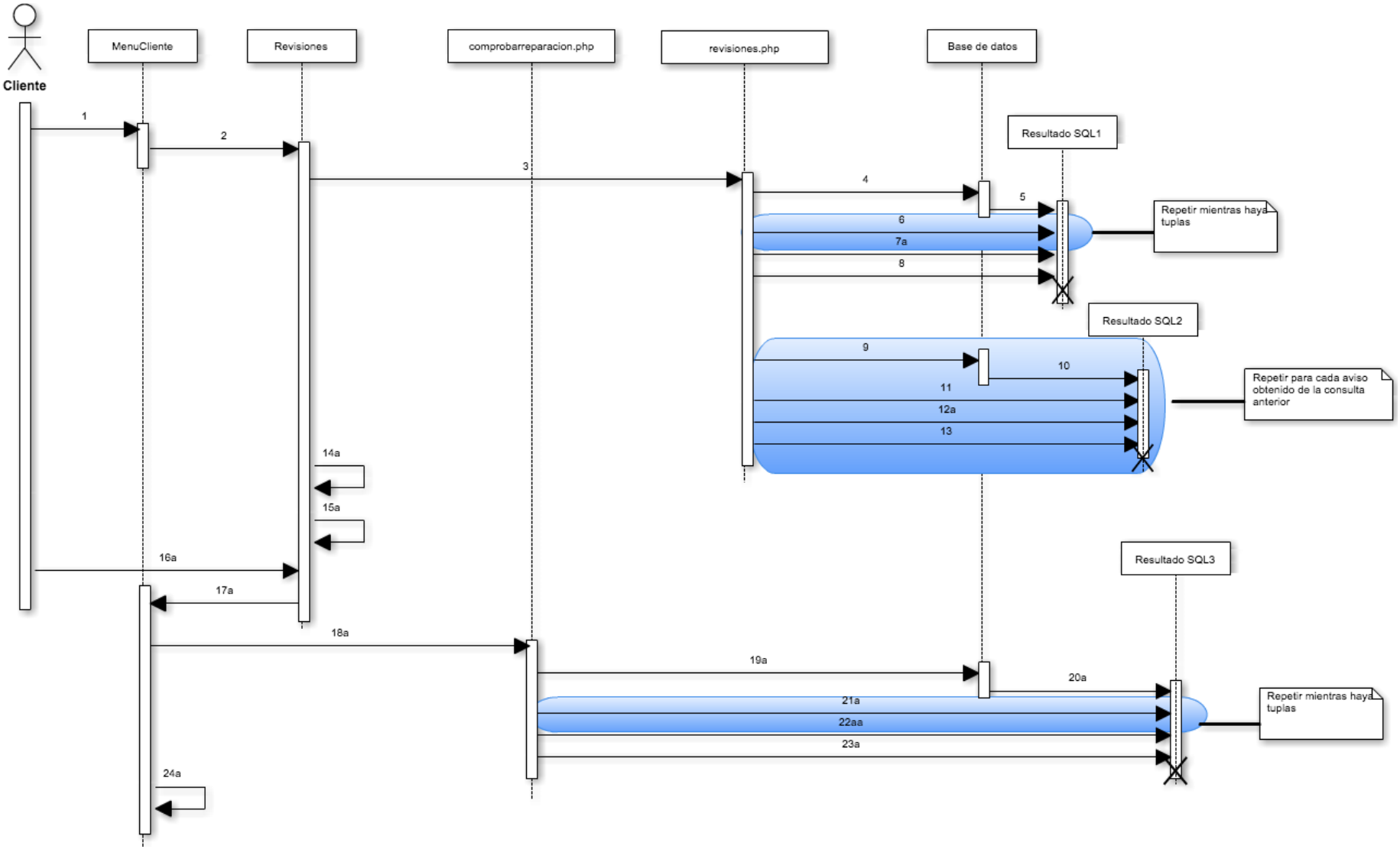


Ilustración 194. - Diagrama de secuencia de caso de uso Consultar revisiones

11.2.4. Diagrama de secuencia de Caso de uso **Ver nuevo aviso**

1. mostrarNotificacion():void; (GCM envía un aviso al dispositivo móvil de la persona que se le ha enviado un aviso desde la aplicación web.)

2. El cliente pulsa la notificación

3. new (nombre, matricula);

4. httpGetData("comprobarreparacion.php?matricula="+matricula):String; (El usuario se encuentra en el menú de la App y cada vez que el cliente se encuentra en el menú se comprueba con la base de datos si la reparación está finalizada, para mostrarle el aviso en el menú).

5. execSQL(sql1)

sql1: "select estado_rep,factura,fecha from tx_reparacion Where matricula='".$matricula."' ORDER BY fecha DESC";

Selecciona el estado de la reparación y el valor de factura de todas las reparaciones ordenadas por fecha, para poder mirar esos datos de la última reparación del vehículo.

6.new()

7. next()

[Si hay tuplas]

8a. getDatos ():Datos;

9.close()

10. mostrarMensaje(Mensaje); (Muestra diferentes mensajes en función de lo que devuelva la consulta) EJ: ("No tienes ningún vehículo en reparación", "La reparación de tu vehículo ya está finalizada, puedes pasar a recogerlo" o "La reparación de tu vehículo todavía no está finalizada")

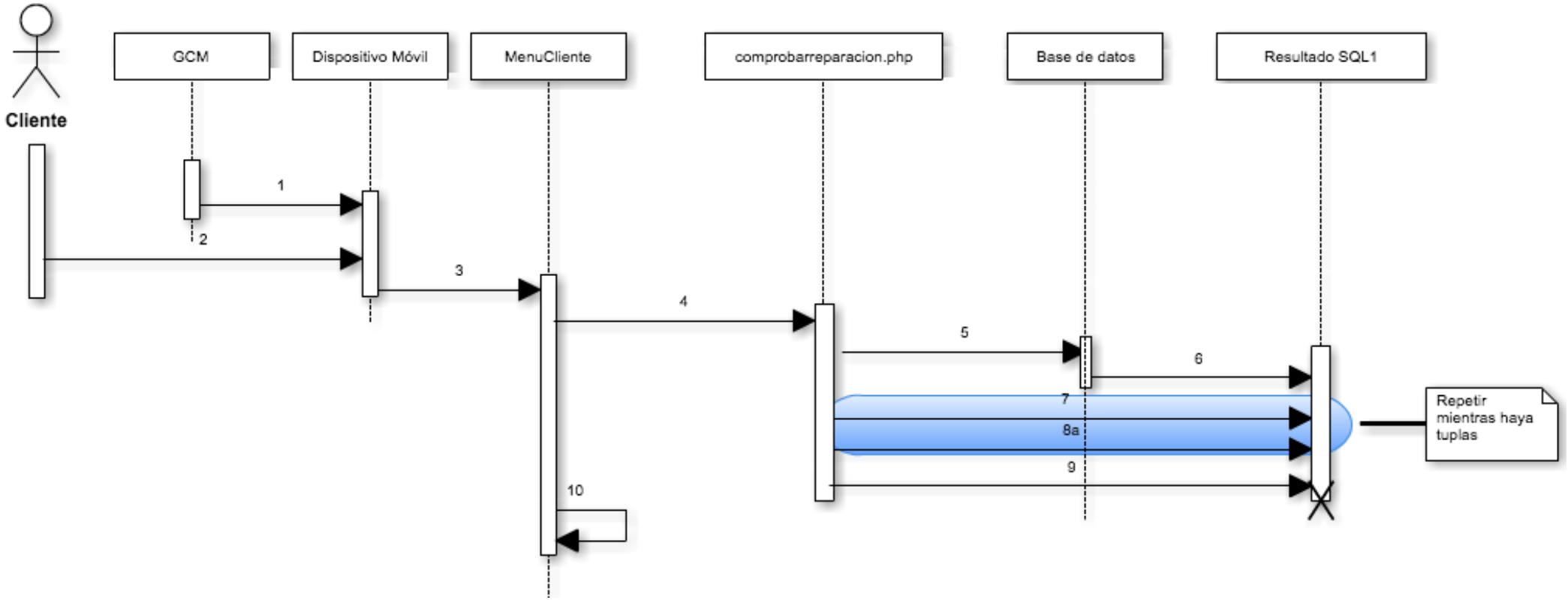


Ilustración 195. - Diagrama de secuencia de caso de uso Ver nuevo aviso