Entsegu Klinikoak Kudeatzeko Web-Sistema

Javier Fernández Anakabe
Kudeaketaren eta Informazio-Sistemen Ingeniaritza Informatikoko Gradua
Bilboko Industria Ingeniaritza Teknikoko Unibertsitate Eskola

2015-08-25

Gaien Aurkibidea

1 Proiektuaren laburpena

Gurutzetako ospitaleko medikamentuen*, medikuntza-entseguen* eta medikuntza-errezeten* kudeaketa eta kontrola automatizatzen lagunduko duen Web aplikazio bat da.

Gurutzetako ospitaleak 140 medikuntza-entsegu* sortzen ditu urte natural batetan zehar bataz bestez, eta horietako bakoitza 15 urteren bitartean gorde beharra dute, entsegu mota hauen historial bat gordeta izateko eta edozein momentutan kontsultaren bat egin nahi izanez gero, modu sinple, erraz eta eraginkor batetan egiteko aukera izateko.

Entsegu bakoitzeko ondorengo informazioa gordetzea da garrantzitsua, besteak beste: zein medikamentu dauden esleituta; zein paziente/gaixo dagoen entseguarekin lotuta; paziente/gaixo bakoitzarekin egin diren medikamentuen dispentsazioak*, noiz egin diren, eta nork.

Hori horrela, aplikazio honek baimen bereziak eskainiko ditu erabiltzaile motarentzat, non, erabiltzaile motak ospitalean duten lanpostu desberdinen arabera egokituko diren. Honen arrazoia garbia da: aplikazioa erabiltzen duen erabiltzaile-kontu bakoitzak bere baimen bereziak izatea eragiketa desberdinak egiteko (adibidez: entsegu edota errezeta berri bat sortzeko aukera administratzailea den erabiltzaileak bakarrik izango du; edo, farmaziako departamentuan lan egiten ez duen baina aplikazioan erregistratuta dagoen medikuren batek, errezeta bateko informazioa ikusi ahalko du, baina ez errezeta horri dagokion dispentsazioa onartu).

Beraz, web-zerbitzari baten bitartez gorde eta kudeatuko da beharrezkoa den medikamentu*, entsegu eta errezetei buruzko informazioa, modu bizkor eta erabilgarri batetan kontrolatu ahal izateko.

Gainera, osasungintzari dagokion informazioa dela eta, informazio hori ondo gordeta egon beharra dago, paziente/gaixoen datu-pertsonalak agerian egon ez daitezen (LOPD-k dioena betez), eta informazio konfidentziala agerian egon ez dadin.

Aplikazio honen funtzionalitateak ondorengoak dira (paziente/gaixo bakoitzari dagokion zenbaki edo gako bat erabiliz, soilik):

-Entsegu bakoitzeko, momentuan Stock-ean dagoen, heldu den eta dispentsatu den (eta zein paziente/gaixori) medikazioaren informazioa kontsultatu.

-Entsegu berri bat gehitu.

-Paziente/gaixo bakoitzeko medikamentu-dispentsazioak kontsultatu, eguneratu eta gehitu.

-Unean dauden entseguak kontsultatu, bilaketarako filtro desberdinen erabileraz baliatu ahal izanik.

-Denbora-tarte batetan zehar egon diren entsegu kopurua aztertu.

-Denbora-tarte batetan irekita dauden entsegu kopurua aztertu.

-Existitzen den entsegu batetatik abiatuz, errezeta berri bat sortzea.

-Medikamentuen Stock orokorra kontrolatzea eta kudeatzea.

2 Sarrera

2.1 Glosategia

Medikuntzarekin lotutako arlo baten gaineko aplikazio bat egin denez gero, termino asko azaltzea komeni da, modu argiago batetan ulertu daiten proiektuaren garrantzia.

Horretarako, memorian zehar adierazten diren zenbait kontzepturen azalpena jartzea pentsatu da.

- Entsegu kliniko: Gizaki baten gainean aplikatutako medikamentu(ar)en edozein ebaluazio esperimentali deitzen zaio, betiere ondorengo helburuetara bideratuta egonik: bere efektu farmakomediko edo farmakozinetikoak aztertu eta ikertu; bere eraginkortasuna ezarri; eta medikamentu horen/horien aplikazioak izan ditzakeen kontrako efektuak ezagutu eta erabilerarako segurtasuna definitu. Entsegu bat medikamentu bat edo gehiagotaz egon daiteke osatuta.
- Errezeta: Entsegu baten gainean, medikamentu bat edo gehiago paziente konkretu bati esleitzea, data eta dosi konkretu batzuetan. Errezetak medikuek eskatzen dituzte paziente bat gaixo dagoenean eta berak adierazten dituen sintomak entsegu baten gaineko medikamentuekin tratatu daitezkeenean. Farmaziako departamentuko langileek, medikuak egindako errezetaren eskaera hori aztertu eta onartzeaz arduratzen dira, beharrezko medikamentuak emanez mediku horri.

  Errezeta onartu dela adierazteko, medikuaren eta farmazeutikoaren sinadura eskatzen da.
- Dispentsazioa: Entsegu baten gaineko medikamentu bat edo gehiago paziente konkretu bati egindako errezetaren konstantzia adierazten duen elementua.
- Medikamentua: Entsegu kliniko batetara esleituta dagoen farmako bakoitza, esperimentala edo ezaguna.
- Pazientea/Gaixoa: Mediku batek errezeta batetan esleitzen duen gizabanako bakoitza, entsegu konkretu batetako medikamentuen tratamendua behar duena.

2.2 Erabiltzaile Eskuliburua

Aplikazioa erabiltzeko laguntza bezala, erabiltzaile eskuliburu bat prestatu dut, funtzionalitate bakoitza non aurkitzen den, eta nola lan egin behar den bakoitzarekin azalduz.

Bideo-demo bat uzten da eskuragarri, funtzionamendua hobeto ikusteko:

http://ttiki.com/341192

SARRERA.

"*FARMAZIAPP*"entsegu klinikoak kudeatzeko helburuarekin erabili daitekeen web-aplikazio bat da. Ondoren, aplikazioak eskaintzen dituen funtzionalitateekin lan egiteko jarraitu beharreko argibideak azalduko dira, erabiltzaile-rol bakoitzak dituen baimenak kontutan harturik.

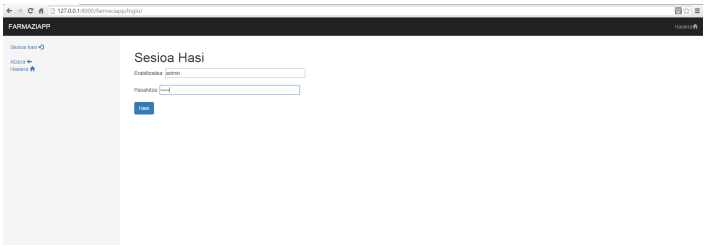
HASIERAKO PANTAILA.

Aplikazioa abiarazterakoan ondorengo pantaila erakutsiko da, sesioa irekita eduki ezean. Hemen, sesioa irekitzeko aukera eskainiko da, bai ezkerraldeko menu bertikalean dagokion aukera klikatuz, bai goiko aldeko menu horizontalean aukeratuz.

Irudia 1: Hasierako pantailaren adibidea.

SESIO HASIERAKO PANTAILA.

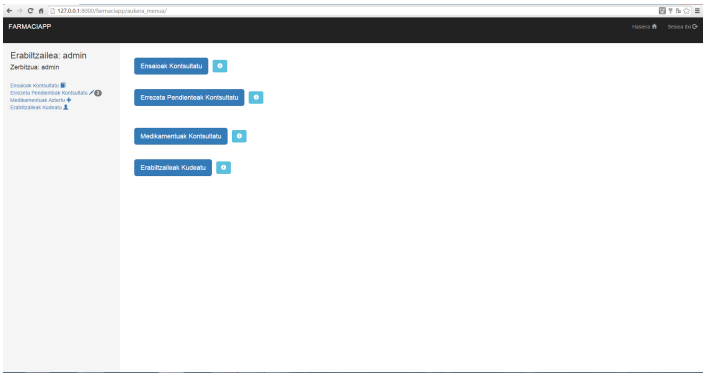
Sesioa irekitzeko, erabiltzaileak erabiltzaile-izen bat eta dagokion pasahitza jarri beharko ditu markatuta dauden testu-kaxetan. Sarbidea zuzena bada, sesioa ondo ireki duen erabiltzailearen sarrerako orrialdera berbideratuko zaio erabiltzaileari; horrela ez bada, sarrera saiakera ez dela zuzena izan jakinaraziko da.



Irudia 2: Sesioa hasieratzeko pantailaren adibidea.

ERABILTZAILEAREN HASIERAKO PANTAILA.

Pantaila honetan erabiltzailearen rolaren arabera aukera desberdinak eskaintzen dira; beraz, rol bakoitzeko erabiltzaileek dituzten aukeren gaineko nabigazioa azalduko da, baimen-maila baxuena duen erabiltzailetik hasita, mailarik altueneko erabiltzaile motaraino. Aipatu beharra dago, gainera, erabiltzailearen hasierako pantaila honetan ageri diren aukera nagusiak ezkerraldean dagoen menu bertikalean ere agertzen direla, edozein pantailatik atzitu ahal izateko, bai eta 'Atzera' botoi bat aurreko pantailara joateko, eta 'Hasiera' botoi bat erabiltzailearen hasierako pantailara bueltatzeko. Horretaz gain, aukera horien guztien goikaldean erabiltzaile-izena eta rola adierazten dira. Goian dagoen menu horizontalean, ordea, 'Hasiera' aukera egongo da, menu bertikaleko funtzionamendu berbera duena, eta 'Sesioa itxi' botoi bat ere agertzen da, erabiltzaileak sesioa itxi eta hasierako pantailara bueltatu daiten.



Irudia 3: Erabiltzailearen hasierako pantailaren adibidea.

'Medicina' rola (erabiltzaile arrunta) duten erabiltzaileek 'Ensaioak Kontsultatu' eta 'Errezeta Pendienteak' aukera izango dute atzigarri. 'Farmazia' rola duten erabiltzaileek, ordea, beste aukera bat izango dute agerian: 'Medikamentuak Aztertu'; eta administratzaile den erabiltzaileak(ek), horietaz gain, 'Erabiltzaileak Kudeatu' aukerarekin lan egiteko baimena ere izango dute.

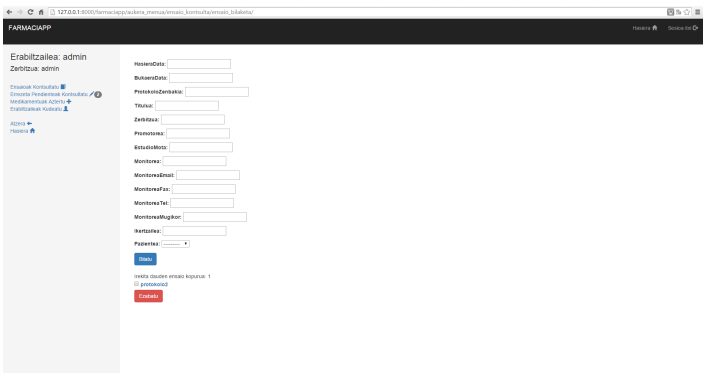
APLIKAZIOAK ESKAINTZEN DITUEN AUKERAK: Web-aplikazioak dituen hasierako pantaila nagusiak azalduta daudela, erabiltzaileak sesioa irekitzen duenean eskaintzen diren aukerak azalduko dira jarraian:

'ENSAIOAK KONTSULTATU' AUKERA.

Bi azpi-aukera eskainiko dira erabiltzaile arruntaren eta 'Farmazia' motako erabiltzailearen kasuan: 'Ensaioak Bilatu' eta 'Ensaioen Historikoa Ikusi'. Administratzaileak, hauetaz gain, 'Ensaioa Sortu' aukera ere izango du.

'ENSAIOAK BILATU' AUKERA.

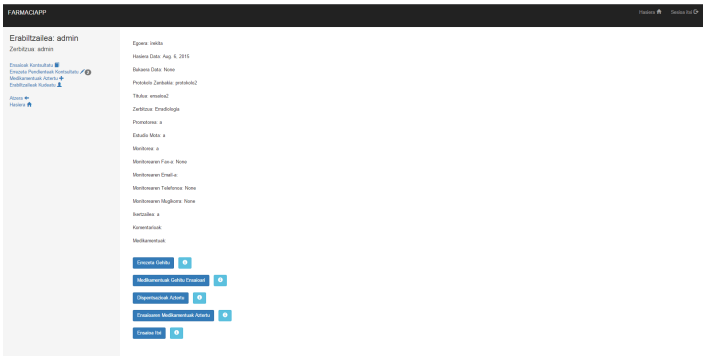
Pantaila honetan aplikazioan erregistratuta dauden eta 'irekita' egoeran aurkitzen diren entsegu guztiak bilatzeko aukera eskaintzen da. Horretarako,erabiltzaileak bete ditzakeen hainbat bilaketa-filtro agertzen dira, erabiltzeko derrigortasunik ez dagoelarik, noski. Bete daitezkeen bilaketa-eremu horietan ez da zertan bilatu nahi den terminoa osorik idatzi behar, idazten denaren arabera koinzidentziak bilatzen saiatuko baita. Hau da, adibidez, 'entsegua1' titulua duen entsegua bilatu nahi baldin bada, 'ens' jarri ezkero eta 'Bilatu' botoiari sakatu ezkero, hitz multzo hori tituluan duten entseguen multzoa agertuko da emaitza gisa.



Irudia 4: Entseguak bilatzeko aukeraren pantailaren adibidea.

Pantaila hori erabiltzaile mota guztientzat izango da berdina, baina administratzailearen kasuan, bilaketa emaitza bakoitzaren alboan, markatu dezakeen kaxetatxo bat agertuko da, aldiberean hainbat kaxeta markatzeko aukera izanik, eta 'Ezabatu' botoi bat ere izango du behekaldean, markatutako aukera guztiak ezabatzeko aukera eskaintzen diona. Esan beharra dago, entsegu bat ezabatzerakoan, entsegu horrekin erlazionatutako beste informazio guztia ezabatzen dela; hala nola, dispentsazioak, eta errezetak. Medikamentuak eta pazienteak ez dira ezabatzen, nahiz eta entsegu konkretu batekin erlazioa izan, beste entsegu batzuetara ere lotuta egon daitezkeelako.

Bilatutako entsegu batetan klikatuz gero, entsegu horren informazio pantailara eramango zaio erabiltzaileari. Pantaila horretan, entsegu horren informazio guztia agertuko da; eta erabiltzaile motaren arabera, beste hainbat aukera desberdin izango dira pantailaren behekaldean. Erabiltzaile arruntaren kasuan 'Errezeta Gehitu' eta 'Dispentsazioak Aztertu' aukerak egongo dira; berriz, beste kasuetan, 'Medikamentuak Gehitu Ensaioari', 'Ensaioaren Medikamentuak Aztertu' eta 'Ensaioa Itxi' aukerak eskainiko dira baita ere. Kasu honetan, 'Farmazia' rola eta administratzailearen rola duten erabiltzaileek eskubide eta baimen berdinak dituzte.



Irudia 5: Entseguaren informazioa erakusten duen pantailaren adibidea.

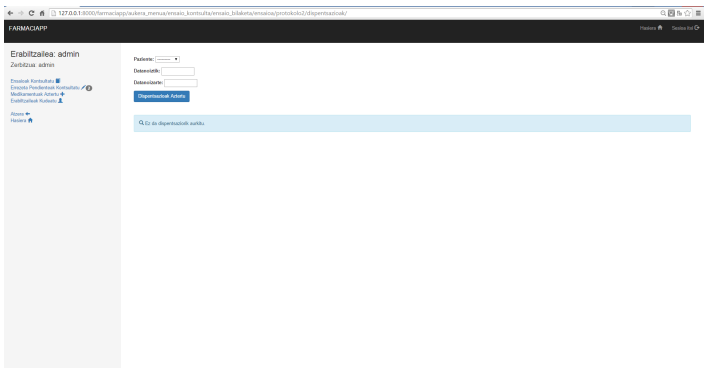
- **<u>'ERREZETA GEHITU' AUKERA.</u>**

  'Farmazia' rola duten erabiltzaileek ezingo dute errezeta berririk sortu, baimen hori mediku batek (erabiltzaile arrunta) edota administratzaileak bakarrik burutu baitezakete. Kasu horietan, aurretik aukeratuta dagoen entseguaren gainean errezeta bat sortzeko argibideak azalduko dira. Zein pazienterentzat egin nahi den errezeta adierazi beharra dago; horretarako, sisteman erregistratuta dauden paziente guztiak erakusten dituen kantseguxa bat dago, eta paziente berri bat erregistratu nahi izanez gero, 'Paziente Berria Erregistratu' aukera ere badago, paziente berri bat erregistratzeko beste pantaila batetara eramaten duena erabiltzailea beharrezko eremuak bete ditzan bere sorrerarako. Errezetaren preskripzio data zein den ere adierazi beharra dago, derrigorrez, ondorengo formatuan: UUUU-HH-EE.

  Ordea, hurrengo preskripzio data ez da derrigorrez bete beharreko atala. 'Gainontzeko eremuak' atalean, idazkera askeko kaxa bat eskaintzen da, errezeta bakoitza desberdina baita, eta medikuak edo administratzaileak, eremu bakar eta berri konkretu batzuk jartzeko beharra izan ditzaketelako. Adibidez, izan daiteke errezeta konkretu baterako 'dispentsatu beharreko ibuprofeno dosia' garrantzia duen eremu bat izatea, baina gainontzeko errezeta guztietan ez. Hori dela eta, idazketa askeko eremu bat baliatu da, derrigorrez bete beharra ez dagoelarik. Behin bete direla bete beharreko eremuak, 'Sortu Errezeta' botoia sakatzen da errezeta berria sortzeko, eta errezeta pendiente bezala gordeko da, oraindik onarpena jaso ez duelako.

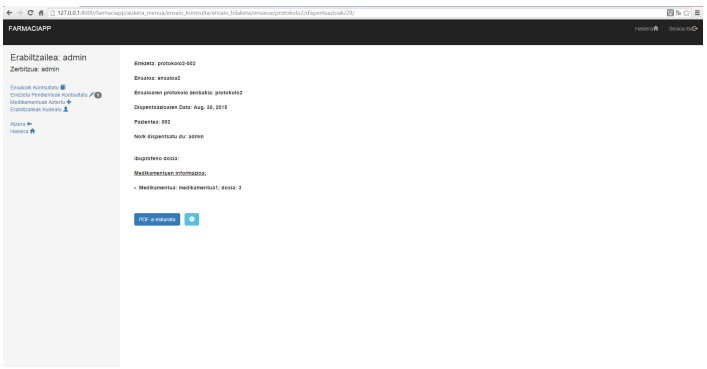
- **<u>'DISPENTSAZIOAK AZTERTU' AUKERA.</u>**

  Aurretik aukeratutako entseguari dagozkion dispentsazioak bilatzeko erabiliko da pantaila hau; hau da, entsegu horrekiko onartuta dauden errezeten informazioa ikusteko. Horretarako, hiru bilaketa-filtro desberdin eskaintzen zaizkio erabiltzaileari: pazientea (paziente konkretu baten gaineko dispentsazioak bilatu nahi baldin baditu), zein datatatik aurrerako dispentsazioak bilatu nahi diren eta zein datarainokoak (formatua UUUU-HH-EE izan behar da). Hiru eremu hauek betetzea ez da derrigorrezkoa, eta aukeraketarik ez baldin bada egiten, entsegu horren gaineko dispentsazio guztiak aurkeztuko dira bilaketan.



Irudia 6: Dispentsazioak aztertzeko aukeraren pantailaren adibidea.

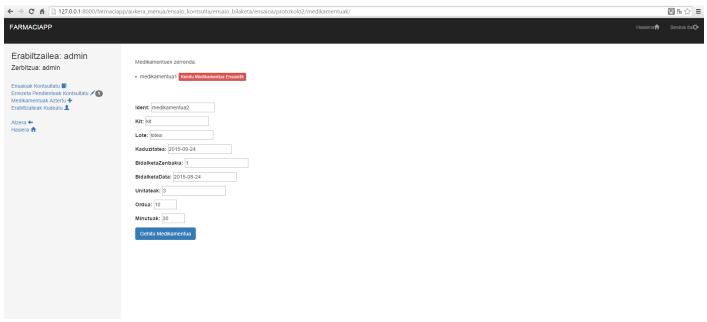
Bilaketa-emaitza bat aukeratuz gero, dispentsazio horren gaineko informazioa erakutsiko da beste pantaila batetan, eta informazio horren PDF fitxategia eskuratzeko aukera egongo da.



Irudia 7: Dispentsazioaren informazioa erakusten den pantailaren adibidea.

- **<u>'MEDIKAMENTUAK GEHITU ENSAIOARI' AUKERA.</u>**

  Entsegu bakoitzak medikamentu bat edo gehiago izan ohi ditu esleituta, eta menu honetan, aukeratutako entsegu horri medikamentu bat edo gehiago gehitzeko aukera egongo da. Horretarako, bete beharra dagoen hainbat eremu ageri dira pantaila horretan, hala nola: medikamentuaren identifikatzailea, kit, lotea, kaduzitatea (formatua UUUU-HH-EE izanik), bidalketa zenbakia, bidalketa data (formatua UUUU-HH-EE izanik) eta medikamentu horren unitate kopurua. Orduak eta minutuek medikamentua erregistratu den orduari egiten diote erreferentzia, baina ez dago derrigorrez bete beharra eremu hauek.

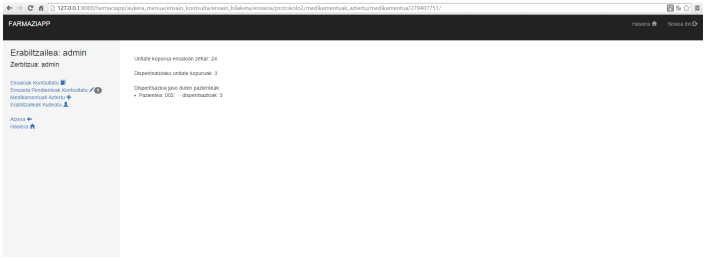


Irudia 8: Entsegu bati medikamentuak gehitzeko pantailaren adibidea.

Medikamentua gehitzerakoan, pantailaren goiko aldean okatutako lista batetara gehituko da, entsegu horretan dauden gainontzeko medikamentuekin batera. Medikamentu bakoitzaren alboan, entsegutik medikamentua ezabatzeko aukera eskaintzen duen botoi bat egongo da, gaizki sartu bada medikamentua aldaketak desegiteko, baina ez da medikamentua sistematik ezabatuko, izan daitekeelako medikamentu hori beste entsegu batzuk erabiltzea. Beraz, medikamentuak kontsultatzeko pantailara joan beharko da erabiltzailea gaizki gehituriko medikamentu hori ezabatzen edo unitateak kentzen.

- **<u>'ENSAIOAREN MEDIKAMENTUAK AZTERTU' AUKERA.</u>**

  Entsegu batetara esleituta dauden medikamentuen gaineko informazioa erakusteko erabiliko da aukera hori. Informazio horretan jakin ahalko da medikamentu konkretu bakoitzeko zenbat unitate egon diren entseguari esleituta, eta zenbat dispentsatu diren bakoitzetik eta zein paziente edo gaixori. Aukera hau oso interesgarria izateaz gain, beharrezkoa da Farmaziako departamentuarentzat, urtero jakin beharra baitute ensaio bakoitzean zenbat medikamentu erregistratu diren, zenbat unitate, eta zenbat dispentsazio egon diren eta nori.



Irudia 9: Entsegu bateko medikamentuaren dispentsazio- eta stock-informazioa.

- **<u>'ENSAIOA ITXI' AUKERA.</u>**

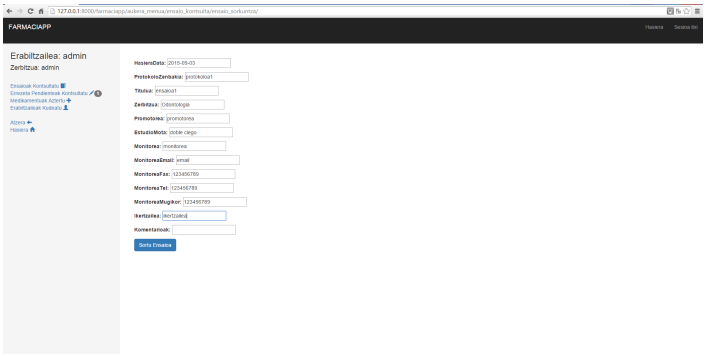
  Entsegua ixteko aukera dago hemen. Konfirmazio mezu bat aterako da, ziurtatzeko ea erabiltzaileak behin betiko itxi nahi duen entsegua edo ez. Baiezkoa aukeratuz gero, entsegua itxi egingo da eta entseguen historikoan erregistratzera pasatuko da, berriz ezin delarik ireki, eta ezingo da operazio gehiago egin entsegu horrekin.

- **<u>'ENSAIOEN HISTORIKOA IKUSI' AUKERA.</u>**

  Entseguen bilaketa pantailaren funtzionalitate berdina dauka, baina oraingo honetan, irekita dauden entseguen gaineko bilaketa izan ezin, itxita dauden entseguen gaineko bilaketa egingo da, eta entsegu horien informaziora atzitzean, bakarrik dispentsazioak ikusteko aukera egongo da; ez da, beraz, errezetarik sortzeko ezta medikamenturik gehitzeko aukerarik.

- **<u>'ENSAIOA SORTU' AUKERA.</u>**

  Bakarrik administratzaileak burutu dezakeen funtzionalitate hau, entsegu kliniko bat sortzeko erabiliko da. Horretarako, betetzea derrigorrezkoa den hainbat eremu desberdin daude; besteak beste: hasiera data (zehazteko entsegua noiz ireki den), protokolo zenbakia, titulua, zein zerbitzuri dagokion entsegua, promotorea, estudio mota, monitorea eta ikertzailea. Berriz, hautazkoak diren eremuen artean hauek daude: monitorearen email-a, monitorearen fax zenbakia, monitorearen mugikorreko zenbakia, monitorearen telefono zenbakia eta idatzi nahi diren bestelako komentarioak.

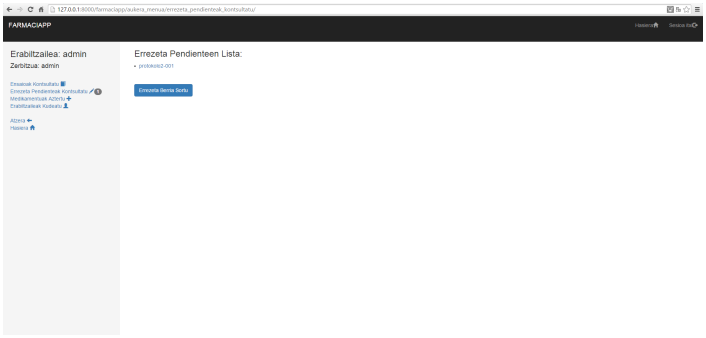


Irudia 10: Entsegu berri bat sortzeko pantailaren adibidea.

Behin eremu horiek guztiak behar bezala bete direnean, adieraziko da entsegua ondo sortu dela, eta entsegu berri horretara medikamentuak gehitzeko aukera eskainiko da ('Ensaio Berria Ikusi Medikamentuak Gehitzeko'). Aukera hori sakatuz gero, entseguaren informazio orrira eramango gaitu, eta puntu horretatik, aurretik azaldu dugun bezala, medikamentuak gehitzeko aukera egongo da, bai eta errezetaren bat sortu nahi bada egiteko ere, adibidez.

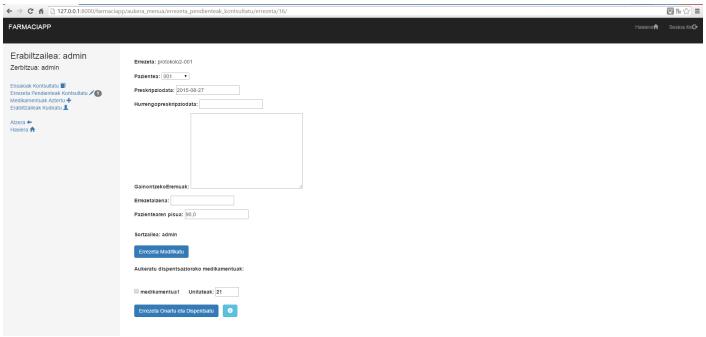
'ERREZETA PENDIENTEAK KONTSULTATU' AUKERA.

Pantaila honetan, sortu diren baina onartuta ez dauden errezeten lista bat agertuko da; 'pendiente' egoeran dauden errezetak egongo dira, alegia. Bi aukera desberdin egongo dira nabigazioaren atal honetan: errezeta horietako batetan klikatzea, edo errezeta berri bat sortzea 'Errezeta Berria Sortu' botoia sakatuz.



Irudia 11: Errezeta pendienteak aztertzeko pantailaren adibidea.

Azken hau egitea erabakiz gero, irekita dauden entseguen bilaketa-orrialdera eramango zaio erabiltzaileari, entsegu bat bila dezan eta errezeta sortzeko aukerarekin jarrai dezan, aurretik azaldu bezala. Berriz, onartu gabe dagoen errezetaren batetan klik eginez gero, errezeta horren informazio-orrialdera berbideratuko da.



Irudia 12: Errezetaren informazioa erakusten den pantailaren adibidea.

Orrialde horretan, informazioa bi era desberdinetan adierazi daiteke: editatzea ahalbidetzen duen eran edo editatzea ezinezkoa den eran. Lehenengo adierazpen modua bi kasutan gerta daiteke: errezetaren informazioa kontsultatzen ari dena errezetaren sortzailea denean, edo administratzailea denean. Bigarren adierazpen era, beste kasu guztietan emango da.

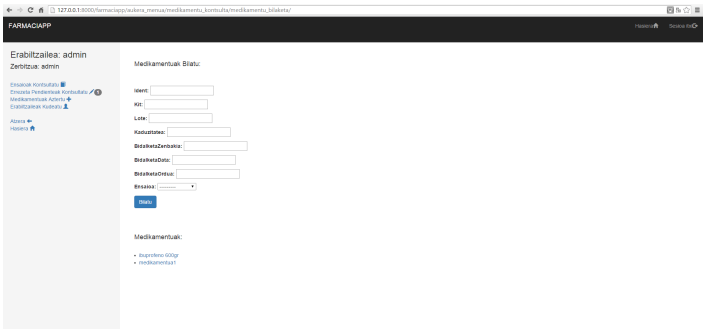
Beraz, zein pazienteri dagokion errezeta agertuko da, bai eta zein datatarako dagoen finkatuta errezeta, hurrengo preskripzio data noiz den, hautazkoak diren beste eremu guztiak, pazientearen pisua eta errezetaren sortzailea nor den. Errezeta editatzeko gaitasuna duen erabiltzaileak eremu hauetako bakoitza (sortzailearen eremua izan ezik) aldatzeko gai izango da, nahi izandako aldaketak eginez eta 'Errezeta Modifikatu' botoia sakatuz.

Beheko atalean, dispentsazioa burutu aurretik egin behar den medikamentuen aukeraketa da. Bakarrik entsegu horretara lotuta dauden medikamentuak agertuko dira, derrigorra izanik gutxienez medikamentu bat aukeratzea eta unitate bat dispentsatzea. Hau egiten ez bada, errezetaren onarpena ez da burutuko, eta beraz, dispentsazioa ez da erregistratuko. Errezeta onartzeko 'Errezeta Onartu eta Dispentsatu' botoia sakatu beharko da.

Horretaz gain, baliagarria den interfazearen atal bat dago lotuta errezeta pendienteekin. Onartzeko pendiente dauden errezetek, batzuetan, premia nagusia izaten dute, eta horretaz ohartzeko, aplikazioaren ezkerreko menuan, zenbat errezeta dauden pendiente adieraztea pentsatu da. Horrela, atentzioa deitzen duen zenbaki baten bitartez, administratzaileak edo 'Farmazia' rola duen erabiltzaileak modu argiago batetan ikusi ahalko du errezetarik dagoen onartzeko eskaerarekin eta horren gainean jokatu.

'MEDIKAMENTUAK AZTERTU' AUKERA.

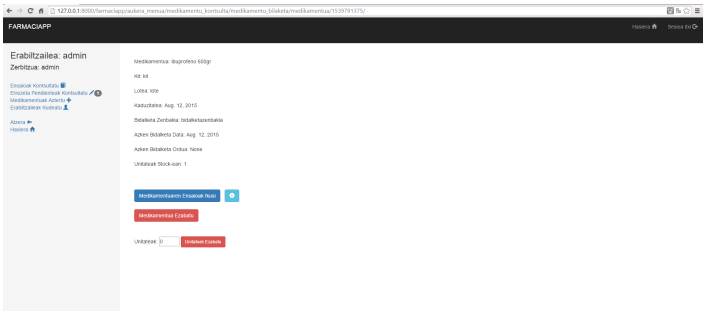
Pantaila honetan soilik funtzionalitate bat eskaintzen da: medikamentuak bilatzekoa. 'Medikamentuak Bilatu' botoiari sakatzerakoan bilaketarako filtro desberdinak agertuko dira, aurretik azaldu bezala entseguen bilaketaren pantailan, eremuetan espezifikatutako karaktere kateekin koinzidentziak izatea begiratzen duena, eta ezer idatzi ezean sisteman erregistratuta dauden medikamentu guztiak bilatuko dituena.



Irudia 13: Medikamentuak bilatzeko pantailaren adibidea.

Bilaketa burutzeko botoiari sakatuz gero, definitu diren ezaugarriak betetzen dituzten medikamentuen zerrenda bat agertuko da, eta elementu bakoitzaren gainean sakatzeko aukera egongo da.

Hori eginez gero, medikamentu bakoitzaren informazioa adieraziko duen pantailara eramango zaio erabiltzaileari. Bertan, beharrezkoa den informazioa agertuko da; hala nola: medikamentuaren izena, kit, lotea, iraungitze-data, bidalketa zenbakia, azkenengo unitatea bidali den data, azkenengo unitatea bidali den ordua eta zenbat unitate dauden Stock-ean.



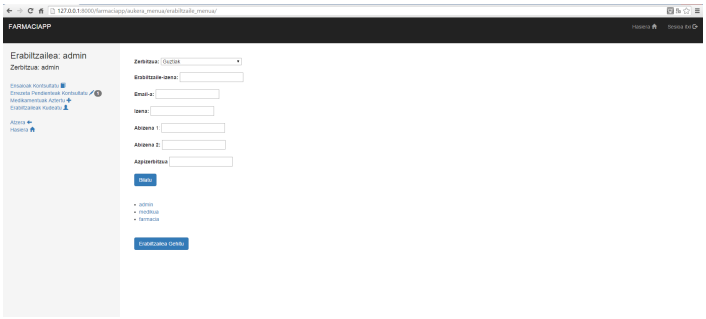
Irudia 14: Medikamentuaren informazioa erakusten den pantailaren adibidea.

Medikamentu hori zein entsegutara lotuta dagoen ikusteko 'Medikametuaren Ensaioak Ikusi' botoia dago. Hor sakatuz gero, medikamentu horrekin lotuta dauden ensaioen zerrenda bat agertuko da, entsegu bakoitzean klikatzea ahalbidetzen duena, entsegu horren informazio-pantailara joan nahi izanez gero.

Oharra*: Gurutzetako ospitaletik aipatu zuten bezala, medikamentu bakoitzaren izen edo identifikatzaileak zerikusia izango du medikamentu horren dosiarekin. Hau da, adibidez, "600mg ibuprofeno"medikamentuaren kaxa multzo bat heltzen bada, ez da "500mg ibuprofeno"medikamentuaren berdina dela suposatuko.

'ERABILTZAILEAK KUDEATU' AUKERA.

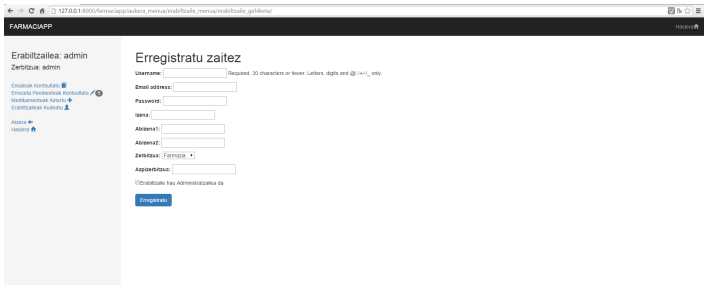
Aplikazioak eskaintzen duen azkeneko aukera administratzailetik atzigarria izango da soilik. Pantaila honetan, sisteman erregistratuta dauden erabiltzaile guztien zerrenda bat agertuko da, bi aukera desberdin emanik: erabiltzaile baten gaineko informazioa kudeatzea edo erabiltzaile berri bat sortzea 'Erabiltzailea Gehitu' botoia sakatuz.



Irudia 15: Erabiltzaileen menuaren pantailaren adibidea.

- **'ERABILTZAILEA GEHITU' AUKERA.**

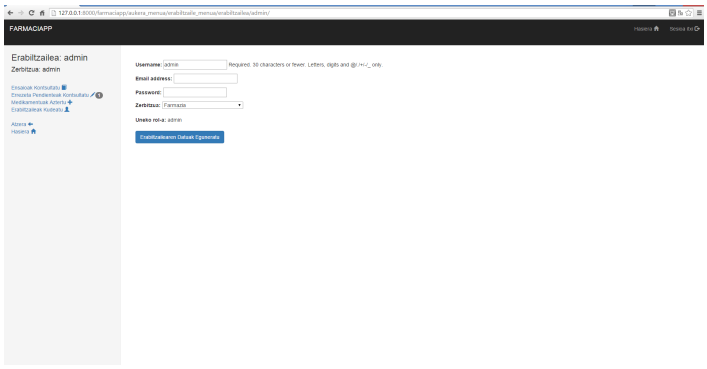
  Azken aukera hau eginez gero, erabiltzailea erregistratzeko pantailara eramango zaio administratzaileari, nahi duen informazioko erabiltzailea sortzeko, baina kasu honetan, beste aukera desberdin bat izango du eskuragarri: erabiltzaile hori administratzailea den edo ez adierazteko eremu bat. Eremu hori markatuz gero, erabiltzaile berria administratzailea dela adierazten da.



Irudia 16: Erabiltzaile bati administratzailearen rol-a esleitzeko eremuaren adibidea.

- **'ERABILTZAILEA MOLDATU' AUKERA.**

  Berriz, sortuta dagoen erabiltzaile batean sakatuz gero, erabiltzailea, email-a, pasahitza eta rol-a aldatzeko aukera egongo da, bai eta erabiltzailea ezabatzeko aukera ere.



Irudia 17: Erabiltzailearen informazioa moldatzeko pantailaren adibidea.

Oharra*: Gurutzetako ospitaleko Farmaziako departamentuarekin adostu zen erabiltzaile baten datu-aldaketa prozesatzeko aukera bakarrik administratzaileak izatea, horrela, segurtasuna handiagoa izango baitzen.

2.3 Entsegu Klinikoen Kudeaketarako Sistemaren Egungo Egoera

Gurutzetako opitalean, Farmaziako departamentuan, entsegu klinikoak eta errezeta klinikoak kudeatzeko erabiltzen duten sistemari dagokionez, gaur egungo egoeraren azalpen labur bat adieraziko dut, aplikazio honen sorrerarako ideiaren arrazoia azaltzeko:

Gurutzetako ospitalean, entsegu eta errezeta klinikoen informazioari buruzko 15 urteren bitarteko erregistro guztiak gorde behar dituzte legez, haien baliozkotasuna urte horien barruan mantentzen dela eta; eta arazoa da, hain justu, informazio hori guztia paperean dutela gordea. Horrek hainbat arazo eta denbora-galera dakartza, argi eta garbi.

Horietako lehenengoa eta garbiena, kontsultak egiterako orduan edo dokumentu konkretu bat aztertu behar izaterako orduan, fisikoki bilatu beharra egotea, eta zer esanik ez antzerako informazioa duten dokumentu multzo bat bilatu nahi denean; hor denbora galera garbi bat egoten da, lan-eraginkortasuna murriztuz.

Beste alde batetik, biltegiratze-espazioaren arazoa dago. Informazio hori guztia fisikoki gorde behar denez gero (15 urtetako informazioa, 140 entsegu kliniko izanik urte bakoitzeko, bataz bestez) Farmaziako departamentuan baliagarri dagoen espazioa gaizki aprobetxatzea ekartzen du. Lan-tokien erosotasun-kalitatea murrizten da dokumentu horiek guztiak gorde beharra dagoen heinean, eta horrek, baita ere, lan-eraginkortasuna murriztea ekartzen du.

Jarraian, gaur egun Gurutzetako ospitalean entsegu klinikoak almazenatzeko eta kudeatzeko erabiltzen duten metodologia eta sistema azaltzen duten argazki batzuk azalduko dira, espazio-falta eta gorde beharreko informazioaren kontrol eta kudeaketarako arazoak (bai eta lan-baldintza kaxkarrak) argi erakusten dituztenak.

Irudia 18: Entsegu Klinikoen gordeketa gaur egun; Gurutzetako ospitalea

Irudia 19: Entsegu Klinikoen gordeketa gaur egun; gurutzetako ospitalea

2.4 Proiektuaren Garapenari buruzko argibideak

Argi dagoenez, bezero konkretu batentzat garatuko den mota honetako proiektu batetan oso garrantzitsua da kontaktu hurbil eta konstantea izatea bezeroaren eta garatzailearen artean. Hori dela eta, Gurutzetako ospitalean Farmaziako departamentuan lanean diharduten langileekin hainbat batzar egiten joan naiz proiektuaren garapena aurrera joan den heinean. Batzar horietan Alazne Bustinza, Gurutzetako Farmaziako departamentuko entsegu eta errezeta klinikoen arduradun nagusia, izan da proiektuaren gorabeherak, itxura eta funtzionamendua probatzeaz, aztertzeaz eta ikuskatzeaz arduratu den pertsona.

Proiektuarekin ofizialki lanean hasi nintzenean, batzar horiek astean behin egiten ziren gutxi gorabehera, laneguna izan zein jai-eguna izan. Bilera horiek fisikoki egiten ziren, Gurutzetako ospitalean, berak lehen pertsonan ikus zedin nola zihoan garatzen aplikazioa, eta proposamenak eta zuzenketak modu argi eta errazago batetan proposatzeko. Hau oso garrantzitsua izan da izan ere, kontzeptu asko hobeto finkatzen edota zuzentzen lagundu baitu, eta horrela, lantzen egon naizen aplikazioa berak behar zuen aplikazio horretatik hurbilago egoten lagundu duelako. Argi geratzea garrantzitsuak ziren kontzeptuen artean hauek egon daitezke, adibidez: entsegu batetan gaixo edo paziente bat baino gehiago egon daitezkeela, baina entsegu bakoitzari dagokion errezeta bakoitza gaixo bati bakarrik dagokiola; edo medikamentu bakoitza zein entsegutara esleituta dagoen jakin beharra dagoela.

Korreo eta telefono bidezko kontaktu zuzena izaterakoan, batzarrak fisikoki ezin baziren gauzatu ere, bazegoen posta elektronikoz edo ahoz zalantzak galdetzeko aukera ere. Horrek asko lagundu du aplikazioaren garapen zuzenean, eta funtzionalitate batzuei dagokienez gaizkiulertuak egon badira ere, kontaktu zuzen eta etengabekoa egoterakoan, zuzenketak epe labur bateko tartean egitea lortu dira.

Proiektuaren onarpena gauzatu aurretik hainbat batzar egin ziren baita ere, Gurutzetako ospitaleko Farmaziako departamentuko hainbat kiderekin, aplikazioaren helburuak zer nolakoa izan behar zuen adosteko, eta proiektu bezala kontsideratu behar zenez zein muga jarri zintezkeen (bai denbora zein lan-kargari zegokienez) komentatzeko. Batzar horiek bukatu ostean adostu zen proiektuaren lehen mailako helburua Farmaziako departamentutik eskatzen zuten aplikazioaren prototipo funtzional bat egitea zela, eta ez produktua bera bukatzea ezartzeko. Adostasun horretara heltzearen arrazoiak garbiak ziren:

Hasteko, Gurutzetako ospitaleko Informatikako departamentuak ez zuen argibiderik eman erabili zintezkeen teknologia edo programazio-lengoaia eta euskarrien inguruan. Hori dela eta, eta emaitzarik gabe informazio hori jakiteko hainbat batzar egitea proposatu baziren ere, garatu nahi zen aplikazioaren softwarea Gurutzetako ospitalean erabiltzen ziren teknologiekin bateragarria ez izateko arriskua zegoela ikusita, azkeneko produktu bat ez egitea adostu zen; horren ordez, guk aukeratutako teknologia berriekin haiek eskatzen zuten aplikazioaren prototipo funtzional bat egitea adostu genuen, Farmaziako departamentuan lantzen zen entsegu eta errezeta klinikoen arloko eragiketa horiek gauzatzeko beharrezkoak ziren eragiketa eta funtzio guztiak egiteko aukera izango zituena. Horrela, adostasun batera heldu arte eta proiektua egokia zela onartu arte denbora nahikoa igaro zenez (hilabete bat, gutxi gorabehera), Gurutzetako ospitalean ezartzearen helburua bigarren mailako helburu bezala utzi genuen.

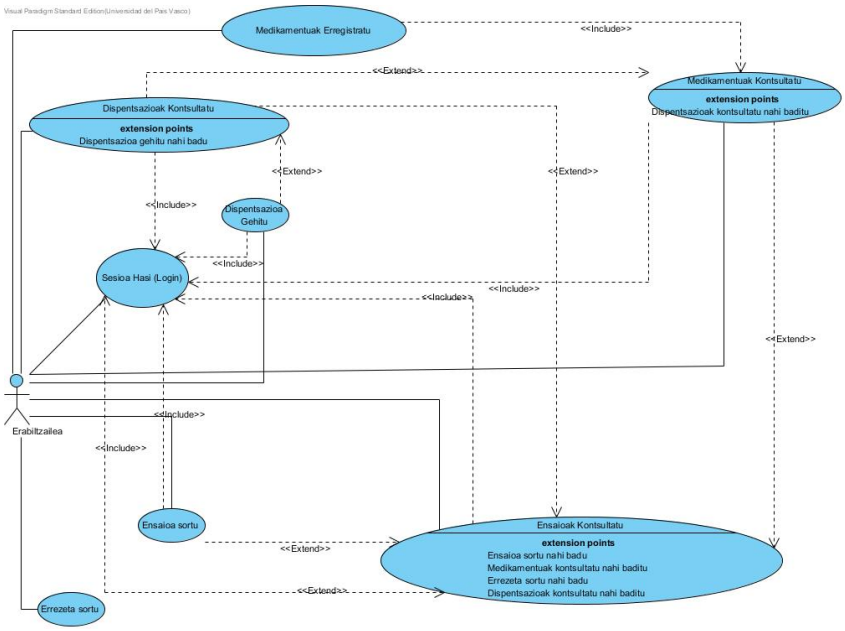
Denboran aurrera eginez, ekainean, eta aplikazioaren garapena amaierako fasean zegoela jakinik, Gurutzetako ospitaleko Informatikako saileko arduradun nagusiarekin bilera bat egitea lortu genuen, ospitalean erabiltzen ziren teknologiak zeintzuk ziren esan ziguten arte. Noski, aplikazioa aurretik aipatutako teknologiekin lantzen hasita zegoen, eta ez zegoen, beraz, aukerarik, jada, berriz hasteko; baina konprobatu genuenez, aplikazioa lantzeko erabiliko ziren teknologia berrien aukeraketa oso zuzena izan zen, plataforma anitzak eta irekiak zirenez gero arazorik gabe akoplatu baitzitezkeen ospitaleko sare informatikoan.

Arazo eta oztopo horiek guztiak kontutan izanik joan da garatzen proiektua, egindako etengabeko bilerak oso garrantzitsuak izan direla erakutsirik. Gainera, behin lehenengo prototipo funtzionala guztiz amaituta neukala eta jakinik aplikazioak zerbitzari bat behar zuela prozesatzeko, Interneteko web-zerbitzari batetan ezartzea pentsatu zen, Alazne Bustinza, Gurutzetako ospitaleko Farmaziako departamentuan entsegu eta errezeta klinikoen jarduerez arduratzen zen langileak, benetako entsegu eta errezeta klinikoekin frogak egin ahal izateko aplikazioa bera atzigarri izanik saretik, fisikoki aplikazioa bere ordenagailuan instalatuta edukitzearen beharrik izan gabe. Hori oso interesgarria izan da benetako kasuekin frogatu ahal izan delako, eta aplikazioaren eraginkortasuna, baliozkotasuna eta onespena zilegizkoak direla baieztatzen lagundu duelako, benetako kasuekin gerta daitezkeen akatsen informeak eginik eta konponbide errealak bilatzen saiatzera lagundurik.

3 Analisia

Aplikazioarekin hasi baino lehen, hark izango dituen funtzionalitateak eta erabiltzaile desberdinak islatuko zituen diagrama egin beharra zegoen; eta behin hori adostu eta gero, aplikazioa garatzeko beharrezkoak izango ziren entitateak eta haien arteko erlazioak definitu beharra. Ondoren, Erabilpen Kasuen Diagrama eta Domeinu-ereduaren Diagramak azalduko dira, diagrama mota bakoitzeko bi bertsio erakutsirik: lehenengo egindako bertsioa, inplementazioarekin hasi aurretik; eta azkeneko bertsioa, inplementazioarekin hasita egonik egindako konponketa eta moldaketekin. Beraz, diseinuaren alderdi garrantzitsuenak azalduko dira, bai eta proiektuaren garapenean izandako eboluzioa nolakoa izan den aspektu horretan baita ere.

3.1 Erabilpen Kasuen Diagrama



Irudia 20: Erabilpen Kasuen Diagramaren Hasierako planteamendua

Proiektuarekin hasi baino lehen, Gurutzetako ospitaleko Farmaziako departamentutik jasotako informaziotik eta aplikazioarentzako eskatzen zituzten betekizunetatik abiatuz, Erabilpen Kasuen Diagrama egitea izan zen lehenengo pausua.

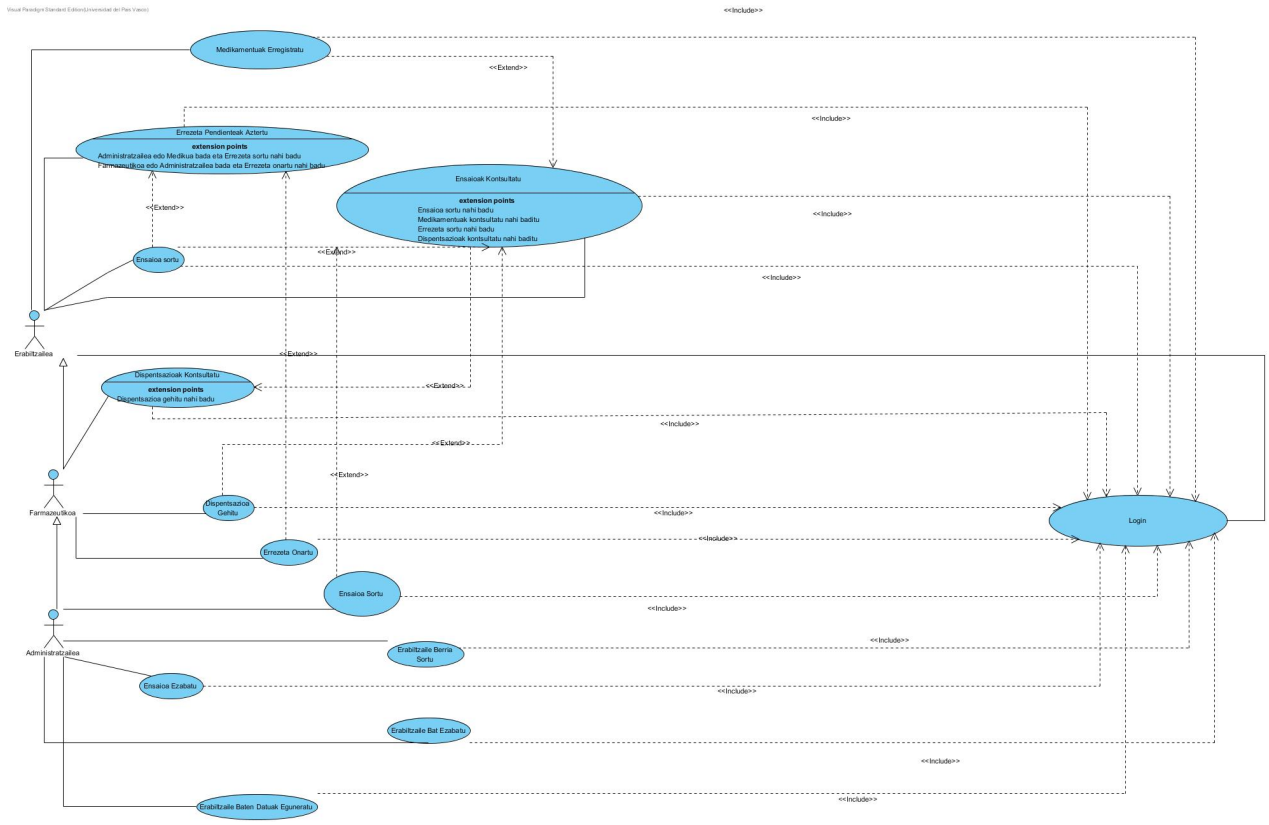
Hasierako planteamenduan, aktore bakar bat egongo zela pentsatu nuen, eragiketak egiteko baimen guztiak izango zituena. Izan ere, hasiera batetan, uste zen aplikazioa soilik Farmaziako departamentuan entsegu kliniko eta errezeten arduradunak erabiltzeko bideratua izango zela. Modu horretan, erabiltzaile horrek bakarrik izango zituen aplikazioak eskainiko zituen funtzionalitateak betearazteko aukera.
Lehenik eta behin, aplikazioarekin lanean hasteko erabiltzaileak sesioa hasi behar zuela argi zegoen gutxieneko segurtasun maila bat izateko, eta horregatik, edozein funtzio edo eragiketa egiteko, lehenik, 'login' aukera burutu beharra zegoen. Sesioa irekita zuela, erabiltzaileak hainbat aukera desberdinen artean aukeratu zitzakeen lan egiteko, hala nola:

Entsegu klinikoak kontsultatu, zeintzuk entsegu zeuden erregistratuta (irekita zein itxita) jakiteko; aukera horretatik abiatuta, nahi izanez gero, erabiltzaileak entsegu berri bat sortzeko aukera zeukan, bai eta menu nagusitik eta entseguen kontsultako aukeran sartu gabe ere entsegu bat sortu zezakeen. Entseguen kontsultako aukeratik beste hainbat funtzionalitatetara atzitu zitekeen: entsegu horri zegokion dispentsazioen azterketa edo kontsulta egin; entsegu horretara lotuta zeuden medikamentuen kontsulta egin; eta baita ere, kontsultatzen ari zen entsegu horren gainean, errezeta berri bat sortzeko aukera zeukan erabiltzaileak. Noski, aukera hauek ere, entseguak kontsultatzeko funtzionalitatean sartu gabe egitea zegoen.

Medikamentu berriak sisteman erregistratzeko aukera ere bazegoen; gainera, nahi izanez gero, medikamentuen kontsulta burutzeko pantailarik ere egin ahal zen.

Komentatzeko falta den azkeneko funtzionalitatea dispentsazio berri bat gehitzearen aukera da. Hemen egon zen, izan ere, aplikazioaren garapen-prozesuan gertatu zirenen arteko lehenengo gaizki ulertua. Hasiera batetan egindako batzarraren ostean errezeta eta dispentsazioaren artean desberdintasunak zeudela uste nuen, pentsatuz dispentsazioak paziente eta medikamentu baten arteko erlazioa adierazten zuela, eta berriz, errezeta bat paziente bati emaniko entsegu baten gaineko medikamentu multzo batzuen preskripzioa zela.

Ondorengo batzarretan ikusi zen hori ez zela horrela, hurrengo diagramaren azalpenean adieraziko den bezala; eta horrekin batera, hasierako funtzionalitateen eta aktorearen diseinuan egindako hainbat elementu eta ezaugarri ere zuzendu edo konpontzera jo zen.



Irudia 21: Erabilpen Kasuen diagramaren azkeneko planteamendua

Azkeneko diagrama honetan ikusten den bezala, lehenengoz eginiko funtzionalitate eta erabiltzaileen diseinutik hainbat aldaketak edota zuzenketak egon dira. Hasteko, aktoreen kopurua oraingoan hirukoa da. Honen arrazoia oso argia da: ospitaleko Farmaziako departamentuko lan-produktibitatea eta eraginkortasuna handituko zela aplikazioa honen erabiltze-zabalkuntza bultzatzen bazen pentsatu zen; hau da, aplikazioa departamentuko entsegu kliniko eta errezeten arduradunak bakarrik erabiltzearen ordez, arduradun horri laguntzeko asmoarekin, gainontzeko langileek ere erabiltzeko aukera bazuten. Jarraian, erabiltzaile desberdinen ezaugarriak eta egin ditzaketen funtzionalitateak azalduko dira:

Hiru aktore mota daude, herentzia erlazio batekin loturik: goi mailan, 'Erabiltzaile Arrunta' dago, Farmaziako departamentutik at dauden erabiltzaileei erreferentzia egiten diena (adibidez, gaixo batentzako entsegu kliniko batetan oinarritutako errezeta bat eskatu nahi duen medikua); erdiko mailan, erabiltzaile arruntak egin ditzakeen funtzionalitateak heredatuz, 'Farmaziako' langileak daude; eta azkenik, aurreko funtzionalitate guztiak egiteko ahalmena izateaz gain, baimen berezi batzuk dituen 'Administratzaile' erabiltzailea dago.

Erabiltzaileak modu honetan banatzearen arrazoia proiektuaren garapeneko lanean hasi eta gutxira adostu zen. Alazne Bustinzak, entsegu eta errezeta klinikoen arduradun nagusiak, argi utzi zuen bera dela sail horren arduradun bakarra, eta lan-karga handia suposatzen diola lan hori guztia berak bakarrik egitea; horrenbestez, arduradun bezala dituen baimen edota ezagutzak behar beharrezkoak ez diren eragiketak multzo batetan jartzea eta baimen edota ezagutza bereziak behar zituztenak beste multzo batetan banatzea pentsatu zen, eragiketen banaketa posible bat proposatzeko. Behin banaketa hori egin zela, aplikazioan parte hartu zezaketen aktore desberdinak zeintzuk izan zintezkeen pentsatzera jo zen.

Bai erabiltzaile arruntek (edo kasu gehienetan ospitaleko medikuek) zein gainontzeko erabiltzaileek ondorengo funtzionalitateak betetzeko aukera izango zutela adostu zen, erabiltzaileen herentzia mailan behera joan heinean, funtzionalitate gehiagorekin lan egiteko aukera izanik: sesioa irekitzeko aukera egon beharra dago (aurreko diseinu eredutik ere argi zegoen), baina Erabilpen Kasuen Diagramaren bertsio berri honetan beste erabilpen kasu berri bat gehitu da: erregistratzeko aukera ematen duena. Izan ere, aurreko diagraman ez zen eragiketa hau kontutan hartzen, erabiltzaile bakar batentzako pentsatuta zegoenez, erabiltzaile berririk erregistratzeko aukerarik ez zelako pentsatu. Baina oraingoan erabiltzaile mota anitz dagoenez gero, erregistroa burutzea beharrezkoa da (bakarrik 'Erabiltzaile Arrunt' edo 'Farmazia'-ko bezala erregistratu daitezkeelarik).

Entseguak kontsultatzeko aukera ere badauka, hainbat bilaketa-filtroren laguntzarekin, baina ezingo du inolako entsegurik moldatu ezta ezabatu ere. Erabiltzaile arruntak soilik entsegu horien informazioa kontsultatu dezake, eta behin kontsulta-orriaren barruan dagoela, entsegu horren gainean egin diren dispentsazioak zeintzuk diren aztertu, bai eta entsegu horren gaineko errezeta berri bat sortu ere.

Errezeta berri bat sortzearen aukera, beraz, edozein erabiltzailek egin dezake; erabaki hau, Gurutzetako ospitaleko Farmaziako departamentuko ordezkariarekin hitz egin eta gero adostu zen, edozein momentutan mediku batek sortuta zegoen entsegu kliniko batetan oinarritutako errezeta bat eskatu zezakeela argitu baitzuen. Horrela, errezeta bat eskatzen denean, "Errezeta Pendienteak"deritzon atalean pilatzen da, non dispentsazio bezala prozesatzeko helburuarekin onartu gabe dauden errezeta eskaerak pilatzen diren. Era honetan kudeatuta errezeten eskaera medikua ez litzateke oinez joan beharko Farmaziako departamentura edo ez luke dei baten beharrik izango errezeta bat eskatu nahi dela adierazteko. Gainera, entseguen kontsulta menuaren barruan aukeratzen den entseguaren informazioaren barrutik, entsegu horren gaineko errezeta berri bat sortzeko aukera ere badago.

Farmaziako langile den erabiltzaileak ('Farmazia' rol-eko erabiltzailea), erabiltzaile arruntak dituen baimen eta funtzionalitateak betetzeko aukera izateaz gain, ondorengo eragiketak egiteko gaitasuna ere badu: medikamentuak kontsultatu. Funtzionalitate honekin, aplikazioan erregistratuta dauden medikamentuak kontsultatu ahalko dira, hainbat bilaketa-filtro desberdinen laguntzarekin. Pantaila honetan, medikamentuen ezaugarriak ikusi ahalko ditu, bai eta zein entsegutara esleituta dauden ere. Gainera, aztertzeko aukeratutako medikamentua zein entsegutara esleituta dagoen ikusterako momentuan, entsegu bakoitzaren informazioa ikusteko aukera ere izango du.

Horretaz gain, errezeta bat onartzeko aukera ere emango zaio; hau da, medikuren ('Erabiltzaile Arrunta') batek errezeta-eskaera bat egin baldin badu, 'Farmazia' rola duen erabiltzaileak errezeta hori onartu ahalko du , dispentsazioa prozesatu dadin, eta dispentsazio horretarako beharrezkoak diren medikamentuen aukeraketa eginez. Entsegu eta errezeta klinikoen arduradunak adierazi zuena kontutan

hartuta, operazio hau Farmaziako departamentuko edozein langilek egitea ahalbidetu nahi da, bere lankarga txikiagoa izateko eta departamentuko lanaren gaineko eraginkortasuna handitzeko, operazio hori aurrera eramateko ez dela baimen berezirik behar argudiatuz.

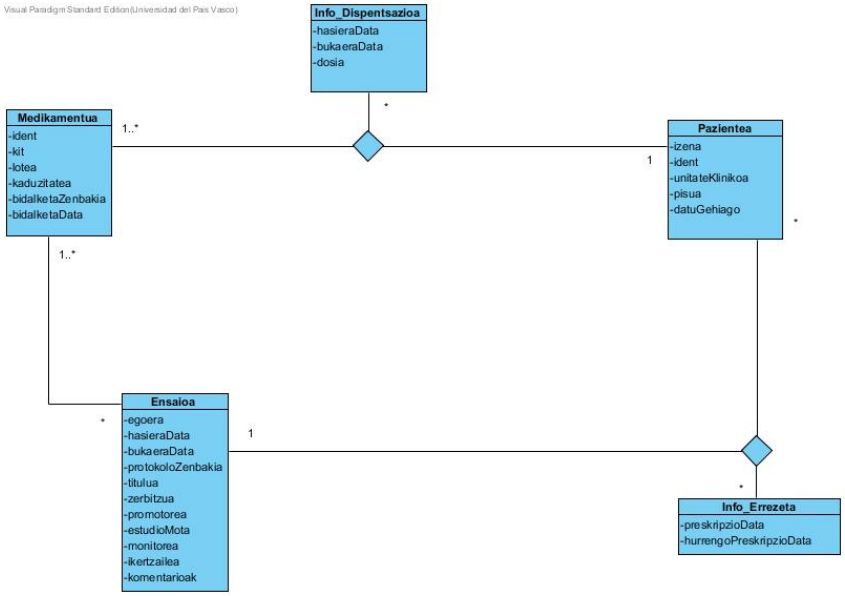
Azkenik, entsegu bat ixteko aukera izango du mota honetako erabiltzaileak. Modu honetan, entsegu baten gainean ez denean errezeta gehiago eskainiko eta egin behar diren froga eta eginbehar guztiak eginda daudenean, entsegua itxi eta historikoan gordetzeko aukera izango du.

Administratzaileak, aurretik komentatutako funtzionalitateak egiteko ahaleraz gain, beste hainbat aurrera eramateko aukera dauka; aplikazioak eskaintzen dituen aukera guztiak burutzeko baimena dauka, alegia. Funtzionalitate horietako batzuk aplikazioa erabiliko duten erabiltzaileen gainekoak dira: erabiltzaile berri bat sortzeko, existitzen den erabiltzaile bat ezabatzeko, eta erregistratuta dagoen erabiltzaile baten datuak eta baimenak aldatzeko gaitasuna izango du. Horrela, lehen aipatu dugunaren gainean, nahiz eta aplikazioan erregistratu nahi den edozein erabiltzailek 'Erabiltzaile arrunt' eta 'Farmazia' arloen arteko edozein rol aukeratu ditzakeen, administratzaileak rol horiek aldatu ditzake edozein momentutan, erabiltzaileen erabilera desegokiren bat gertatzearen arriskuari aurre egiteko aukera izanik. Gainera, aukera hau interesgarria da administratzaileak bakarrik ahalko duelako 'Administratzaile' rola esleitu beste erabiltzaile bati edo erabiltzaile berri bati. Honela, beste 'Administratzaile' erabiltzaile baten laguntza behar duen kasuan, adibidez, beste erabiltzaile bati esleitu diezaioke modu horretan jokatzeko aukera.

Beste alde batetik, administratzailea izango da entsegu kliniko bat sortzeko zein ezabatzeko gaitasuna izango duen erabiltzaile bakarra. Nahiz eta entseguak ezabatzearen aukera beharrezkotzat ez zuen jo hasiera batetan Farmaziako departamentuko entsegu eta errezeta klinikoen enkargatuak, egia da garrantzitsua dela aukera hau existitzea aplikazioan, hanka sartzeren bat egon ezkero entsegu hori erregistroan gordeta egon ez dadin edo ongi sortu ez den entsegu baten gaineko errezeta eskaerarik sortu ez daitezen. Beraz, entseguak sortzeko aukera ere edonork ezingo duenez izan, bere garrantzia eta eragiketa hau aurrera eramateko eduki beharreko ezagutza eta erantzukizuna handiak direlako, entseguak ezabatzeko aukera ere bakarrik administratzaileari esleitzea adostu zen.

Aipatzekoa da, baita ere, 'Ensaioak Kontsultatu' funtzionalitatearen eginkizun berdina betetzen duen beste funtzionalitate bat ere badagoela, 'Ensaioen Historikoa Ikusi' deitzen dena. Funtzionalitate honek bakarrik itxita dauden entseguak ikusteko aukera emango du, baina ez du utziko errezeta berriak sortzen entsegu horren gainean. Berriz, 'Ensaioak Bilatu' aukera arruntak, irekita dauden entseguen gaineko bilaketak eta eragiketak egiteko aukera eskainiko du.

3.2 Domeinu-ereduaren Diagrama

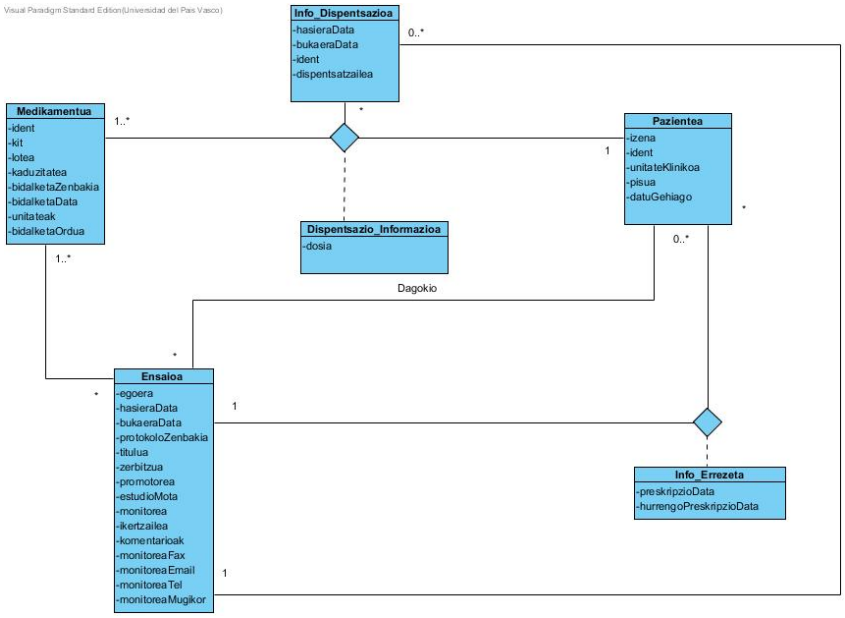


Irudia 22: Domeinu-ereduaren Diagramaren Hasierako Planteamendua

Behin Erabilpen Kasuen Diagrama eginda zegoela, Domeinu-ereduaren Diagrama egitera jo nuen. Kasu honetan, Erabilpen Kasuaren Diagramarekin ez bezala, ez da aldaketa nabarmenik egon diseinuaren lehenengo eta azkeneko bertsioen artean, baina egia da egon direla moldaketa txiki batzuk. Aipatu behar da hainbat erlazio lehenengo begiradan ulerterrazak ez badira ere, datu-basearekin erlazioa duten kontsultak eta bestelako eragiketak egitea errazteko helburuarekin definitu direla.

Ikus daitekeen bezala, 6 erlazio-entitate proposatu nituen hasiera batetan: Medikamentua, Pazientea, Ensaioa, Info_Errezeta eta Info_Dispentsazioa, bakoitzari dagozkion atributuekin. Gainera, ikus daitekeen bezala, hiru erlazio desberdin ageri dira entitate hauen artean:

'Medikamentu - Ensaio' erlazioan adierazten den bezala, argi zegoen hasieratik entsegu kliniko bakoitzeko hainbat medikamentu desberdin egongo zirela (gutxienez bat), eta medikamentu bakoitza hainbat entsegu desberdinetara lotuta egon zitekeela. 'Medikamentu - Paziente - Info_Dispentsazio' erlazioa dispentsazioei dagokien informazioa adierazteko eta gordetzeko definitu zen. Adierazpen horretan, dispentsazio bakoitzeko eta paziente bakoitzeko gutxienez medikamentu bat izan behar duela esleituta adierazten da; paziente bakoitzeko eta medikamentu bakoitzeko izan daitekeela zero, bat edo hainbat dispentsazio egotea esleituta; eta dispentsazio bakoitzeko eta medikamentu bakoitzeko instantzia, bakarrik paziente bati egokituko zaiola. Azkenik 'Paziente - Ensaio - Info_Errezeta' erlazio hirutarra ageri da; paziente bakoitzeko eta entsegu bakoitzeko ez dagoela derrigortasunik errezetarik egotera adierazten da, eta egotekotan, nahi beste egon daitezkeela; paziente bakoitzeko eta errezeta bakoitzeko bakarrik entsegu bakar bati egingo diela erreferentzia; eta, entsegu eta errezeta bakoitzeko hainbat pazienteri egokituta egon daitekeela. Azken hau, azkeneko diagramaren adierazpenean ikusiko denez, dispentsazio eta errezeta kontzeptuen gainean izan nituen kontzeptuen gaizki ulertuen ondorioz egondako adierazpen okerra da, entsegu bateko errezeta bakoitzak bakarrik paziente bati egin behar baitio erreferentzia.



Irudia 23: Domeinu-ereduaren Diagramaren azkeneko planteamendua

Domeinu-ereduaren Diagramaren azkeneko bertsio honetan, 4 erlazio-entitate daude 'Errezeta' erlazioa bigarren mailako edo informazioa gehitzeko entitate bezala erabili dudalako; gainera, beste bigarren mailako entitate bat gehiago ere badago: 'Dispentsazio_Informazioa'. Argi erakusten denez, hasiera batetan egindako lana eta pentsatutako adierazpena zuzenak izan ziren ia bere osotasunean. Entitate batzuen atributuak ere aldatu egin dira lehenengo bertsiotik azkenekoraino.

Bertsio berri honetan ikusten den bezala, erlazio kopurua handiagoa da, hasierako diseinuan oinarrituta lanean jarduten joan nintzen heinean, aplikazioaren inplementazioan zehar hainbat oztopori konponbideak aurkitzeko beharrezkoak izan baitziren; bai eta aplikazioak izan behar zituen betekizunen gaineko zuzenketa kontzeptual batzuen ondorio izan ere.

'Medikamentua - Pazientea - Dispentsazioa' erlazioa berdin mantendu da bertsio honetan ere, baina oraingoan informazioa gehitzeko erabiltzen den bigarren mailako entitate bat gehitu diot. 'Medikamentua - Ensaioa' erlazioan aldatu den aspektu bakarra bi entitateen arteko erlazioan ageri diren elementu-mugak (edo multiplizitatea) dira, kasu honetan entsegu batetan gutxienez medikamentu bat egotera behartuta, eta medikamentu bat, gutxienez, entsegu batetakoa izatera mugatuta.

Beste erlazio pare bat 'Paziente - Ensaio' erlazioak dira; ikus daitekeenez, bi erlazio mota daude, batetan, informazio gehigarria eskainiko duen entitate laguntzaile bezala 'Errezeta' dagoelarik, eta bestean ez. 'Errezeta' erlazio laguntzailearen erlazioak entsegu batetan dagoen paziente batek izan dezakeen errezeta bati egoten dio erreferentzia; beraz, entsegu batetan hainbat pazientek izan dezakete errezeta bat, inolako derrigortasunik ez egonik, eta paziente batek hainbat entsegutako errezetak izan ditzake, inolako derrigortasunik egon gabe hemen ere. Bi entitate hauen arteko beste erlazioa, 'Dagokio' izenduna, paziente bat hainbat entsegutara lotuta egon daitekeela adierazteko definituta dago, eta entsegu batetan hainbat paziente egon daitezkeela atzituta adierazteko, erlazio honetan ere, bi noranzkoetan, inolako derrigortasunik ez dagoelarik.

Azkeneko erlazioa 'Dispentsazioa - Ensaioa' da; entsegu batetan hainbat dispentsazio egon daitezkeela adierazten du, baina ez du derrigortzen bat izatera, eta dispentsazio batek, bakarrik eta nahitaez, entsegu bati egin behar diola erreferentzia ageri da.

Modu honetan, nahiz eta hasieran egindako diagramarekin alde askorik ez dagoen, ikusten da entitateen arteko erlazio kopurua unitate batean handitu dela, bai eta entitate nagusi kopurua unitate kopuru berdinean txikitu dela ere, diseinu orokorra mantendu dela erakutsiz. Aipatu beharra dago, egindako aldaketak, gehienbat aplikazioaren inplementazioan, datu-basean gordetako elementuen edo entitateen gaineko erlazioei dagokienez aurkitutako zailtasunei konponbideak aurkitzeko helburuarekin egin direla, kodea ulerterrazagoa bihurtzeko eta inplementazioa intuitiboagoa egiteko.

4 Diseinua

4.1 Prototipoak

Behin Erabilpen Kasuen Diagrama eta Domeinu-ereduaren Diagrama eginda zeudela (lehenengo bertsioa) eta aplikazioak izango zituen funtzionalitateak adostuta zeudela, paperezko prototipo azkar bat egitera jo nuen. Honen helburua, aplikazioaren pantaila desberdinek izango zuten itxuraren zirriborro bat egitea zen, eta pantaila horien arteko nabigazioa nolakoa izango zen eta nola egingo zen adostea.

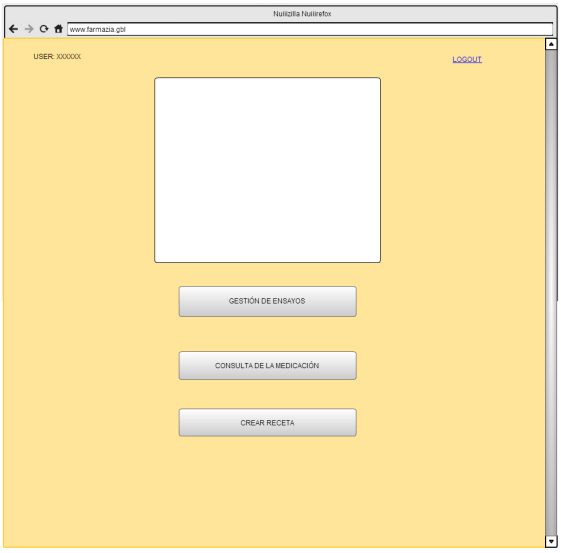
Ez nion denbora gehiegi eskaini paperezko prototipoari, esan bezala, bizpahiru pantaila marraztu bainituen, aplikazioaren interfazearen ideia orokor bat izateko eta funtzionalitateak nola sartu zitezkeen pentsatzeko.

Hori egin eta batera, prototipo digital bat egitea pentsatu nuen. Horretarako Cacoo web bidezko softwarea erabiltzea pentsatu nuen, web bidezko aplikazio baten pantailen interfazea egitea erreza baitzen, ez zuelako zailtasun- ez ikasketa-maila alturik eskatzen, eta pantailen arteko nabigazioaren simulazioa diseinatzeko oso aproposa baitzen.

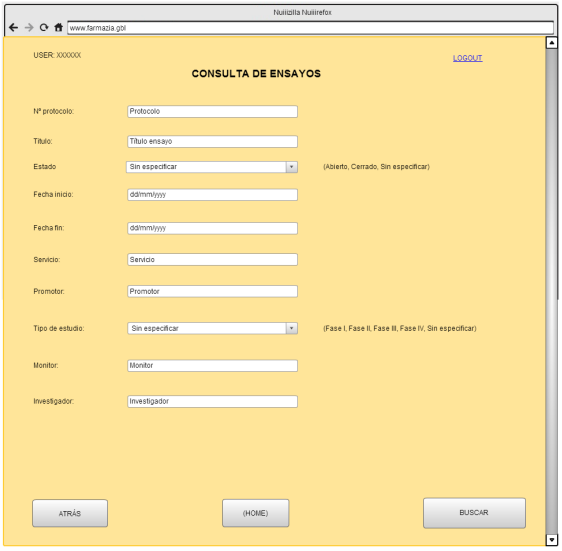
Denbora eraman zuen prototipo digitala egituratzea planteatutako erabilpen kasu guztiak barne izango zituena, eta erabilterraza eta intuitiboa izango zelaren helburuak beteko zituena. Gainera, prototipoa egituratzen nindoan heinean, Gurutzetako ospitaleko Farmaziako departamentuarekin bi batzar izan nituen diseinua eta funztionalitateetarako atzipen modua gustukoa zuten edo ez jakiteko. Behin Alazne Bustinzak, entsegu eta errezeta klinikoen arduradunak, ontzat eman zuen prototipo digital ez-funtzionala, aplikazioa garatzen hasi nintzen, bakarrik kodeak funtzionatzeko logika nolakoa izan behar zenaren ardura bakarrarekin, entitateen, funtzionalitateen eta aspektu bisualaren egituraketa definituta baitzeuden.

Jarraian, martxoan egindako prototipo digitalaren pantaila desberdinen adibide batzuk jarriko ditut, azkeneko aplikazioarekin dituen antzekotasunak ageriak direlarik, eta proiektuaren amaierako emaitzaren oinarri sendo bat dela erakutsirik.

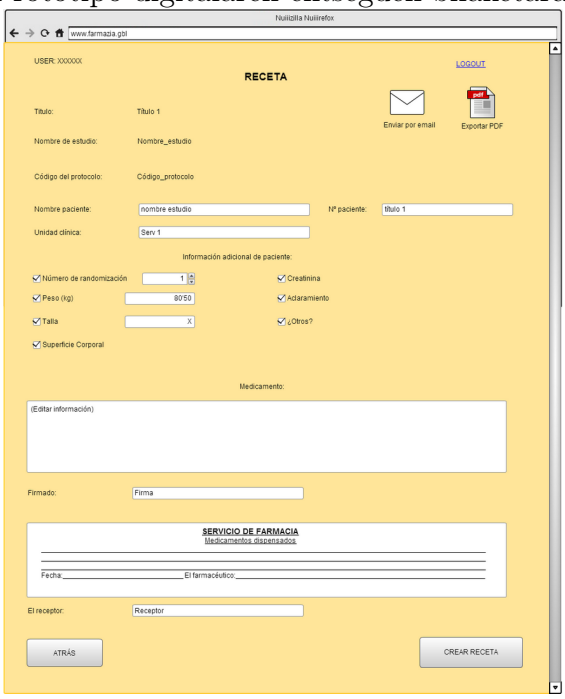
Prototipo Digitalaren esteka: http://ttiki.com/341141



Irudia 24: Prototipo digitalaren orrialde nagusiaren pantaila



Irudia 25: Prototipo digitalaren entseguen bilaketarako pantaila



Irudia 26: Prototipo digitalaren errezetaren sorkuntzarako pantaila

4.2 Hausnarketa

Proiektuaren garapenean zehar argi eta garbi ikusi dudanez, diseinu-diagramak egitea oso garrantzitsua eta lagungarria da, eta nahiz eta hasiera batetan egindako diagramen bertsioaren eta bukaeran lortutako bertsioaren artean desberdintasunak eta aldeak izan, esan beharra daukat hasiera batetan diagramak egin ez banituen ez nituzkeela egin beharra zeuden aldaketa horiek hain garbi ikusiko.

Erabilpen Kasuen Diagramak asko lagundu dit aplikazioak izango dituen funtzionalitateak aztertzeko. Modu horretan, aplikazioarekin interakzioa izango zuten erabiltzaile desberdinen rolak definitzea, baimenak esleitzea eta funtzionalitate nagusienak orokortzea eta nabarmentzea lortu dut, inplementazioan zehar argi ikusten nuelarik zer nolako bloketan banatu behar nituen eginkizunak; gainera, itxura eta interfaze bisuala nola diseinatu ahal nituen ikusten lagundu dit.

Domeinu-ereduaren Diagramak, beste alde batetik, datu-basea diseinatzerako orduan erraztasun asko eman dizkit, entitate desberdinak zeintzuk izango ziren eta haien arteko erlazioak nola egin behar nituen adieraztea errazagoa eginik. Eta ez bakarrik alde horretan; aplikazioa garatzerako orduan, inplementazioa errazteko eta ulerterrazagoa egiteko ere oso baliozkoa izan zait.

Egindako prototipoei dagokienez, paperezko prototipoak aplikazioaren interfazearen hasierako ideia orokor bat egitera lagundu zidan, gero, Cacoo erabiliz egindako prototipo digitalean, diseinua egitea errazagoa izan zelarik. Baina, hau guztia, esan bezala, Erabilpen Kasuen eta Domeinu-ereduaren Diagrametan lan egitera eskainitako denborari esker izan da, lana errazago egiten eta egin beharrekoa errazago ikusten lagundu baitidate. Gainera, aplikazioaren inplementazioan (batez ere interfaze bisualari dagokion ataletan) lanean nenbilela, prototipo digitala eginda izatea ondo etorri zitzaidan, diseinua nolakoa izan behar zen guztiz pentsatu beharrik ez nuelako izan.

5 Erabilitako Teknologien Azalpen Orokorra

Aplikazio hau egiterako orduan teknologia berrien erabileraz baliatu naiz, web-aplikazio bat egitea baitzen helburua. Web bidezko aplikazio bat egitearen arrazoia, aplikazioa lokalki instalatu beharra ekiditea zen, batez ere. Aplikazio hau zerbitzaritik prozesatuko litzateke, zerbitzari horren kontra egindako *HTTP* eskaera desberdinak eginez, erabiltzaileak agerian duen eta bisualki erabili dezakeen aplikazioaren interfazean erantzunak agertzea posible eginik.

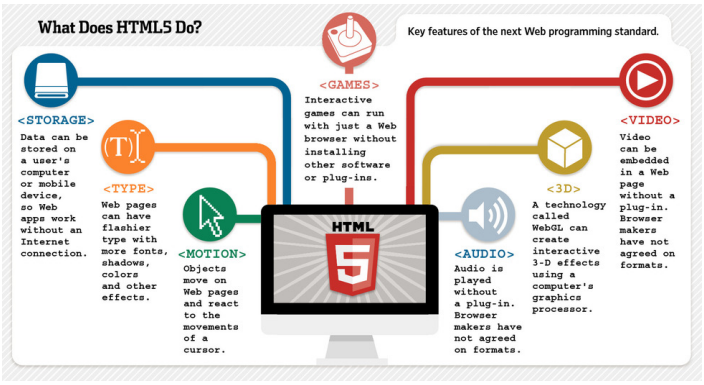
Esan bezala, zerbitzari eta aplikazioaren arteko elkar-komunikaziorako HTTP (*HyperText Transfer Protocol*) protokoloa erabili da, hain justu World Wide Web edo Internet bidezko transakzioetan erabili den protokoloa delako; software web-elementuek erabili ohi duten sintaxi eta semantika definitzen dituelako, besteak beste. Horrela, esan bezala, zerbitzari baten eta erabiltzaile-aplikazio baten arteko pakete transakzio edo eskaera-erantzunen bidez funtzionatuko duenez aplikazio honek, protokolo honen erabilera argia da.

Ideia hau aurrera eramateko, aplikazioa zein ingurunetan garatu, datu-baseekin lan egiteko zein lengoaia aukeratu eta zein programazio-lengoaiarekin landuko zen pentsatu beharra zegoen. Aukeratutako teknologia berrien ezaugarri nagusienak honako hauek dira:

- *Python*: Ezagunak diren beste hainbat programazio-lengoaien antzera, *Python* agindu bidezko programazio lengoaia da; hau da, funtzio edo instrukzio batzuk interpretatzearen bitartez, dagokion programak portaera bat edo beste izatera eramaten duena. Programazio lengoaia hau, azken urteetan, gero eta erabiliago eta ezagunagoa egiten joan da, bere sintaxi argi eta ulerterraza dela eta, bai eta eskaintzen dituen aukera sendo eta desberdinengatik; besteak beste, objektuetara bideratutako programazioan lan egiteko ahalmena erakusten duelako eta plataforma anitza delako.

- *HTML5*: *HTML* (*HyperText Markup Lenguage*) web-dokumentu baten edukian adierazten den egitura eta semantika adierazteko eta errepresentatzeko lengoaia erabiliena eta ezagunena da, eta hain justu *HTML5* bere bertsio berriena da, elementu, atributu eta portaera-mota desberdinak dituelarik.

  Open Web edo Web Irekia diren garatzaile guztiek erabiltzeko diseinatua dago, eta hainbat abantaila berri gehitu ditu lehendik zeuzkanei, besteak beste: zerbitzariarekin modu berri eta eraginkorretan komunikatzeko bideak eskaini ditu; hainbat web-guneri datuak lokalki gordetzea ahalbidetzen die, bezeroaren aldetik; audioa eta bideoa erabiltzeko euskarri hobetsi eta zabalduak eskaintzen ditu; API berriak eskaintzen duten abantailak aprobetxatzeko aukera (drag-drop, offline lan egiteko aukera, web workers, web sockets, etab.) eta Javascript lengoaiarekin batera lan egiteko aukera; etab. Gainera, edozein gailurekin bateragarria izateak, gaur egungo merkatuan indar handia duen eta oso erabilia den teknologia izatera eraman du. Izan ere, esana da *HTML5* *Flash* eta *Silverlight* lengoaien ondorengoa izango dela, eta gutxinaka-gutxinaka indarrean jartzen joango dela.



Irudia 27: *HTML5*-en elementu erabilgarrien irudia. Iturria: http://ttiki.com/341137

- *JavaScript*: Lengoaia interpretatua deritzon objektuetara bideratutako programazio-lengoaia dinamiko bat da, web-orrialdeak garatzerako orduan erabili ohi dena. Baina, nahiz eta web-orrialdeen garapenean erabiltzen den script programazio lengoaiatzat hartzen den, erabili daiteke baita ere nabigaziorik behar ez duten inguruneetan (adibidez node.js). Lengoaia honen sintanxia *C++* eta *Java* lengoaietatik ez da oso desberdina, bere sortzaileen helburua programatzerako orduan ahalik eta kontzeptu berri gutxien ikasi behar izatea baitzen, eta aipatutako bi lengoaia horiek ezagunak eta erabiliak izaterakoan, sintaxia egituratzeko oinarri bezala erabiltzea pentsatu zen, baina ez du zerikusi zuzenik haiekin.

  JavaScript lengoaia dinamikoa izateak objektuak exekuzio denboran eraikitzea ahalbidetzen du, bai eta funtzioak izan ditzaketen aldagaien eta script dinamikoen sorrera, etab. Programazio-lengoaia hau, urteetan zehar, gero eta gehiago erabiltzen joan da, eta bere eboluzioa eta zabalkuntza nabariak izan dira. *HTML5* web-orrialdeak programatzeko lengoaiarekin bateragarria izaterakoan, gainera, askoz ere ahalmen handiagoa lortzea eragin du.

- *jQuery*: *JavaScript* lengoaiarekin lotura duen liburutegi berezi bat da, HTML lengoaiarekin interakzio zuzena eskaintzeko gaitasuna errazten edo sinplifikatzen duena. Kode irekiko software librea da, eta beraz, edonork erabiltzeko gaitasuna eta baimena dauka. Bere ezaugarririk nabarmenena, *JavaScript*-en oinarritutako kodea izanik, liburutegi hau ez erabiltzearekin konparatuz kode-lerro kopurua murrizten laguntzen duela da, funtzionalitate konkretu bat inplementatzerako orduan.

- *Django*: Web-orrialdeak kudeatzeko eta garatzeko erabiltzen den lan-ingurune bat da. Kode irekikoa da eta maila altuko *Python* programazio-lengoaian kodetzeko prestatuta dago. Defektuz, SQLite 3 motako datu-base lengoaia erabiltzen du datuak almazenatu, atzitu eta lan egiteko, baina MySQL eta PostgreSQL erabiltzea ere ahalbidetzen du; eta *HTML5* eta *JavaScript*-en erabilera onartzen du web-orrialdeen diseinua eta interfazea lantzeko. *Python*, berriz, aplikazioak izango dituen funtzionalitateak betetzeko beharrezkoa den logika inplementatzeko erabiltzen da. Horrela, Model-View-Controller patroia errespetatzen du *Django*k, atal bakoitzari, dagokion gunea esleituz fitxategiak gorde eta kudeatzeko, eta haien arteko banaketak eta loturak errespetatuz eta tratatuz.

  Framework edo lan-ingurune honen helburua web-orrialdeen diseinua eta egituraketa-prozesua bizkortzea eta erraztea da. Nahiz eta bere sorrera garaian ez zegoen bideratuta edozein web-orrialde garatzeko ingurunea izatera, gaur egun oso tresna erabilia izaten hasi da, jakinik *Python* indarra irabazten ari dela, eta *HTML5* eta *JavaScript* web-orrialdeak egiterako orduan oso hedatuta dauden teknologiak direla.

  Django-k, lan egiteko, *Python* lengoaiaren 2.5 bertsioa eskatzen du gutxienez, eta ez dira beharrezkoak lengoaia horretako beste liburutegi batzuk gehitzea. Gainera, *Django*-k berak web-zerbitzari bat dakar lan egiteko, bakarrik erabiltzaile bat konektatuta egotea ahalbidetzen duena, erabiltzaileak web-zerbitzari bat instalatuta eduki beharra ez egonez.

- *SQLite* 3: *Django*-k defektuz erabiltzen duen datu-baseak kudeatzeko sistema da. *SQLite*-k, beste datu-baseak kudeatzeko sistema batzuek ez bezala, programarekin zuzenean lotuta egoten den datu-basea eraikitzen du. Horrela, kontsultak egiteko erabiltzen dituen funtzioak behar duten logika sinpletzea lortzen du, eta datu-basearekiko deien latentzia murrizten du.

- Visual Paradigm: Gaur egun hedatuta dagoen UML diagramak egiteko software bat da. Mota askotako diagramak egitea eskaintzen ditu. Horrenbestez, ordainpeko software bat bihurtuta dago, hainbat baliabide desberdin eskainten dituelarik eta hainbat enpresa eta erkaundek bere zerbitzua kontratatu dutelarik.

- DigitalOcean: Web-zerbitzari pribatuak eskaintzen dituen enpresa estatubatuar bat da, denboraepeka ordaindu daitezkeelarik zerbitzari horien kontratazioak. Bizi-denbora laburra daukan arren oso tresna erabilia bihurtu da azken boladan, web-aplikazioak abian jartzen laguntzeko medio eta laguntza garrantzitsu bat baita.

- Overleaf: Zientzia-artikulu eta -dokumentuak idazteko eta garatzeko erabiltzen den doaneko plataforma bat da. Bere abantaila nagusienetako bat sarean kokatuta dagoela da, eta beraz, ez dago lokalean ezer instalatu beharra bere erabilerarako. Oso tresna erabilgarria da LaTeX-ekin lan egiten duten erabiltzaileentzat, denbora errealean erakusten baitu, pantaila bikoiztu baten laguntzarekin, zein den LaTeX-eko testua prozesatzearen emaitza. Gaur egun, oso arrakasta handia irabazi duen tresna da eta gero eta handiagoa da editatzeko erreminta hau erabiltzen duten pertsona kopurua.

- Cacoo: Sarean dagoen eta dohainik erabili daitekeen edozein motatako diagramak egiteko teknologia bat da. Hedatuta dagoen tresna bat da, hainbat erabiltzaile desberdinek proiektu berdinaren gainean lan egitea ahalbidetzea delarik eskaintzen duen abantaila nagusietako bat. Prototipo digitalak egiteko erabili daitekeen softwarea da, erabilterraza, eta konplexutasun maila baxua daukana, baina aplikazio baten hasierako maketa bat egiteko oso baliozkoa dena.

- *Stack Overflow*: Teknologia berrien (eta batez ere programazio-lengoaia desberdinen) gaineko zalantzak galdetu eta argitzeko helburuarekin sortu zen web-gune bat da. Bertan, erregistratuta dauden erabiltzaileek zalantzak proposatu edota argitu dezakete, beti ere modu zuzen eta egoki batetan planteatuz.

  Gero eta ezagunagoa egiten ari da orrialde hori, izan ere, galderak erantzuten dituzten edo galdera egokiak planteatzen dituzten erabiltzaileek puntuak lortzen baitituzte, eta hor agerian geratzen da erabiltzaile bakoitzaren dedikazioa eta ezagutza. Hori dela eta, enpresa batzuk, bere lanpostuetan jendea kontratatzeko asmoa dutenean, pertsona desberdinen *Stack Overflow*-ko puntu kopurua begiratzen dute, adibidez.

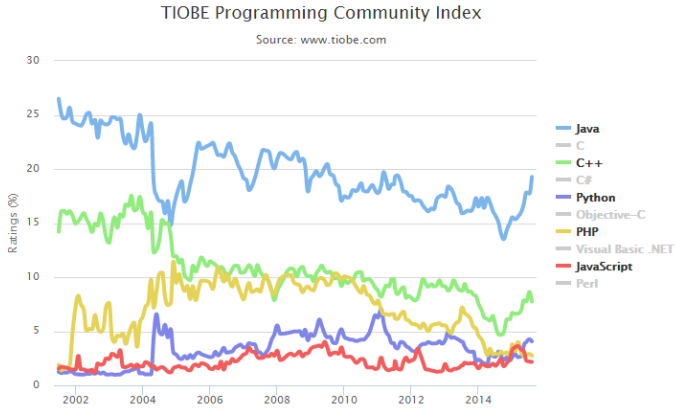
Bukatzeko, gaur egun gehien erabiltzen diren teknologien eboluzioa nolakoa izan den azken urteetan errepresentatzen duen grafika pare bat adierazten dira, erakutsirik aukeratutako teknologiak gorakada dutela eta izatea espero dutela etorkizunean ere.

Grafika hontan ikusten da *Python* programazio-lengoaia azken urteetan programatzaileen erabileran ezagunagoa eta erabiliagoa izaten joan dela, eta beraz, gorakadan dabilela, gehien erabiltzen diren lengoaien artean bostgarren postuan kokatuz 2015eko abuztuan; ikus daiteke, baita ere, *JavaScript* lengoaiak gorakada izan duela eta hemeretzigarren postuan aurkitzen dela. Hori bai, aipatu beharra dago, *HTML5* lengoaiarekin guztiz erabilgarria dela, eta *HTML5* dela gaur egunean web-guneak diseinatzeko lengoaiarik garrantzitsuena.

Aug 2015	Aug 2014	Change	Programming Language	Ratings	Change
1	2	↑	Java	19.274%	+4.29%
2	1	↓	C	14.732%	-1.67%
3	4	↑	C++	7.735%	+3.04%
4	6	↑	C#	4.837%	+1.43%
5	7	↑	Python	4.066%	+0.95%
6	3	↓	Objective-C	3.195%	-6.36%
7	8	↑	PHP	2.729%	-0.14%
8	12	⇈	Visual Basic .NET	2.708%	+1.40%
9	10	↑	JavaScript	2.162%	-0.01%
10	9	↓	Perl	2.118%	-0.10%
11	11		Visual Basic	1.781%	-0.23%
12	24	⇈	Assembly language	1.760%	+1.11%
13	13		Ruby	1.416%	+0.17%
14	18	⇈	Delphi/Object Pascal	1.407%	+0.49%
15	21	⇈	MATLAB	1.232%	+0.50%
16	14	↓	F#	1.232%	+0.14%
17	23	⇈	Swift	1.179%	+0.51%
18	15	↓	Pascal	1.138%	+0.09%
19	20	↑	PL/SQL	1.137%	+0.35%
20	30	⇈	R	1.010%	+0.49%

Irudia 28: Gaur egungo teknologien erabileraren ranking-a. http://ttiki.com/341138

Ondorengo irudian, lengoaia ezagunenen erabileraren eboluzio bat adierazten duen grafika bat ageri da, urteetan zehar, ikusirik, esan bezala, *Python* eta *JavaScript* hazten dabiltzala, nahiz eta Java edo C++ bezalako lengoaien aldean ez den hainbeste erabiltzen. Bai, ordea, web bidezko aplikazioak programatzeko.



Irudia 29: Programazio-lengoaia desberdinen erabileraren eboluzioa. http://ttiki.com/341139
Irudiak *Tiobe*-ren web-gunetik aterata daude, Software-kodearen analista eta aztertzaile ezaguneneta-riko orrialdetik.

6 Helburuak eta Baliabideak

6.1 Helburua

Web-aplikazio hau garatzea aukeratzearen arrazoiak hainbat dira, eta guztiak garrantzitsuak, nire ustez. Bilboko Industria Ingeniaritzako Unibertsitate Eskolak Gurutzetako ospitalearekin duen hitzarmen bat dela medio, Gurutzetako ospitalean hamar lan-egun egon nintzen bertan lantzen ziren alor desberdinetako arazoak aztertzen, hobekuntzak proposatzen eta konponbideak pentsatzen, Farmazia arloan zegoen benetako beharrizan bat aurkitu nuen arte.

Proiektu hau aurrera eramatea interesgarria iruditzearen arrazoi nagusia hain justu hori da: benetako beharrizan bat dela, eta ez dela bakarrik Gurutzetako ospitaletik eskatzen ari diren aplikazio bat. Estatu mailan dauden beste hainbat ospitaletatik deiak egon direla Gurutzetako ospitaleko Farmaziako departamentura ziurtatu didate, entsegu eta errezeta klinikoak kudeatzeko aplikaziorik duten edo ez jakiteko, eta erantzuna beti izan da berdina: ezetz.

Izan ere, gaur egungo egoera Gurutzetako ospitalean zera da: entsegu eta errezeta klinikoen 15 urtetako erregistro guztiak gorde behar dituztela legez, haien baliozkotasuna urte horien barruan mantentzen delako; eta arazoa da informazio hori guztia paperean dutela gordea. Horrek hainbat arazo eta denbora-galera dakartza, argi eta garbi. Horietako lehenengoa eta garbiena, kontsultak egiterako orduan edo dokumentu konkretu bat aztertu behar izaterako orduan, fisikoki bilatu beharra egotea, eta zer esanik ez antzerako informazioa duten dokumentu multzo bat bilatu nahi denean; hor denbora galera garbi bat egoten da, lan-eraginkortasuna murriztuz. Beste alde batetik, biltegiratze-espazioaren arazoa dago. Informazio hori guztia fisikoki gorde behar denez gero (15 urtetako informazioa, 140 entsegu kliniko izanik urte bakoitzeko, bataz bestez) Farmaziako departamentuan baliagarri dagoen espazioa gaizki aprobetxatzea ekartzen du. Lan-tokien erosotasun-kalitatea murrizten da dokumentu horiek guztiak gorde beharra dagoen heinean, eta horrek, baita ere, lan-eraginkortasuna murriztea ekartzen du.

Ondorioz, esan daiteke ez dela soilik Gurutzetako ospitaletik datorren beharrizan erreal bat; estatu mailako beste hainbat ospitalerentzako interesgarria izan daitekeen aukera bat ere badelako, eta horrek, orokorrean existitzen den benetako beharrizan bat dela erakusten du, eta ez kasu partikular bat.

Beraz, ikus daitekeen bezala, informazio hori guztia gordeko duen, eta, batez ere, kudeatzen lagunduko duen aplikazio bat egotearen garrantzia eta beharrizana nabarmena da. Eskaria oso handia izan da Farmaziako alorretik, eta gogotsu parte hartzen ari dira proiektu hau aurrera eramatearen ahaleginetan.

Web-aplikazio honekin lor daitezkeen abantailak nabariak dira, beraz: lan-eraginkortasuna handitzea bilaketak eta kontsultak azkarragoak eta intuitiboagoak izanik, eta lan-karga berdina aurrera eramateko beharrezko denbora murriztea; eta beste alde batetik, Farmaziako departamentuko lan-espazio baldintzak hobetzea. Monike de Miguel, Farmaziako departamentuko zuzendariordeak, adierazi duenez, gainera, aplikazio hau aurrera eramaten bada proiektu bezala eta funtzional egitea lortzen bada haiek eskatzen duten baldintza eta eskaera-puntuekin, Gurutzetako ospitalean erabilera zuzenerako ezartzea eskatuko da, aplikazioarekin ahalik eta bizkorren lan egiten hasteko. Gainera, Espainiar estatutik heltzen zaizkien kontsulta-deiak direla eta, espero daiteke web-aplikazioa estatu mailara zabaltzea etorkizunean.

Beste alde batetik, aplikazio hau web bidezkoa izatearen helburua dago. Honen arrazoia, aurretik azaldu bezala, aplikazioa erabili daitekeen ordenagailu bakoitzean lokalki instalatu beharra ekiditea da. Horretarako, web-zerbitzari bat izango da aplikazioa prozesatzeaz arduratuko dena, eta Gurutzetako ospitaleak abisatu zigun bazutela zerbitzari propioak hau gertatzea ahalbidetu zitzaketenak. Horrenbestez, Gurutzetako ospitaleko sarean konektatuta dauden ordenagailu guztiek aplikaziorako atzipena izatea ahalbidetzea lortuko litzateke, modu praktikoago batetan.

Aplikazioak berak izan behar zituen helburuak aztertzerako orduan, hauek dira nagusienak edo garrantzitsuenak:

- *User-friendly*: Gurutzetako ospitaleko Farmaziako departamentutik hainbatetan adierazi zutenez, aplikazio honekin bilatu nahi den helbururik garrantzitsuena lan-bizkortasuna da; hots, heldu berri dagoen entsegu bat eta horri dagozkion medikamentuak azkar erregistratzeko gaitasuna izatea, bai eta errezeta bat onartu behar denean eta paziente bati dispentsazio bat egin behar zaionean azkartasunez egitea. Horretarako, aplikazioak izan behar duen interfazea intuitiboa izatea behar beharrezkoa da; bere erabilera erraz ikasi daitekeena eta ez duena interpretazio anitzik eskaintzen izan behar da, beraz.

  Helburu hori betetzeko oso garrantzitsuak izan dira aurrez aurre eginiko batzarrak Gurutzetako ospitaleko Farmaziako departamentuan aplikazio hau erabili dezaketen langileekin, haiek adostu behar baitute interfaze grafikoari dagozkien elementuek zein egitura, kokapen edota forma izan behar duten haien larra errazagoa eta intuitiboagoa izan dadin.

  Noski, aplikazioak tratatuko dituen datu multzoa handia izan daitekeenez gero, bilaketak, kontsultak eta bestelako eragiketak (elementuen ezabaketak edo gehikuntzak) ahalik eta azkarrenak eta eraginkorrenak izatea espero da.

- Datuen babesa edo erabiltzaile-lehentasunen kudeaketa: Oso garrantzitsua da datu konkretu batzuk ez egotea atzigarri edozein erabiltzailerentzat horien kontsultarako; adibidez, paziente edo gaixo bakoitzaren izen-abizenak ezingo dira kontsultatu zuzenean, identifikazio-kode bakar bat izango baitute esleituta haien bereizpenerako, eta informazio-pertsonal hori bakarrik errezeta bat idazten denean ikusi ahalko da, ez edozein unetan. Gainera, aplikazio honek Gurutzetako ospitalean erabiltzen diren beste aplikazio desberdinekin loturarik izango ez duenez, (independentea denez beste aplikazioekiko, alegia) ez da egongo paziente edo gaixo konkretu baten informazio pertsonal gehigarririk lortzeko arriskurik.

  Horrekin lotuta, oso garrantzitsua da erabiltzaile desberdinek izango duten baimenak kudeatzea. Kasu honetan hiru erabiltzaile mota egongo dira: 'Administratzailea', 'Farmazeutikoa' eta 'Erabiltzaile Arrunta', bakoitzak eragiketa konkretu batzuk egiteko aukera izango duelarik soilik. Hau egiteko, bilera bat organizatu zen Gurutzetako ospitaleko Farmaziako departamentuan baita ere,

argi geratzeko erabiltzaile bakoitzaren baimen eta murriztapenak zeintzuk izatea komeni zen. Modu horretan, gertatu behar ez litzatekeen eragiketa eta operazioak aurrera eramatea kontrolatzea espero da.

Lehen mailako helburu hauek guztiak azaldu eta gero, bigarren mailakotzat jo ziren helburuak zeintzuk diren azaldu beharra dago.

Hasteko, aplikazioa Gurutzetako ospitalean ezartzearen helburua azaldu beharra dago. Nahiz eta helburua, hasiera batetan eta proiektua onartu aurretik egin ziren batzarretan komentatzen zen bezala, ospitalean bertan ezarri eta erabiliko zen aplikazio bat garatzea zen, hainbat arazo eta oztopo egon ziren helburu hori betetzea konplikatzera eraman zutenak.

Jakina da ospitaleko programa eta aplikazioak erabiliak izatea ahalbidetzeko daukaten softwarea eta hardwarea ezin direla edozein motatakoak izan, eta haien zuzendaritzak baimenduta egon beharrak daudela. Horrenbestez, ospitalean erabiltzeko helburua betetzeko, aplikazioa garatzen hasi aurretik, zein teknologiekin lan egin beharko genukeen jakin nahi genuen, proiektuaren fin hori bertan behera geratu ez zedin. Hori jakinda, Gurutzetako ospitaleko Informatikako sailarekin kontaktatzeko ahaleginak egon zela aipatu behar da. Baina nahiz eta esfortzuak egin ziren haiekin elkarrizketa bat izateko eta erabili zintezkeen teknologiei buruz gauzak argitzeko, ez zen ez erantzun azkarrik ez konkreturik egon.

Horrek, proiektuaren inplementazioaren garapenean atzerapenak egotea eragin zuen, nahiz eta aplikazioarekin berarekin erlazioa zuten beste hainbat ataletan lanean nenbilen (betekizun-bilera, diseinua, paperezko prototipoak, prototipo digitalak edo mock-ups, etab.). Azkenik, proiektuaren zuzendari Juanan Pereirarekin adostu genuen, hoberena, proiektua Gurutzetako ospitaleko Informatika sailaren erantzunari luzaroan itxaron gabe guk nahi genituen teknologiekin lanean hastea zela. Momentu horretan, aplikazioaren garapenerako erabiliko ziren teknologien aukeraketa oso garrantzitsua izan zen.

Beraz, une horretan, hasiera batetan geneukan bukaerako helburua aldatzera jo behar izan genuen, eta hor adostu zen guk aukeratutako teknologiekin garatutako prototipo funtzional bat egitea izango zela helburu nagusi berria, eta bigarren mailako helburutzat jo genuen aplikazioa ospitalean ezartzearen ahalegina, ez baikenekien ziur aplikazioa garatzeko teknologia baimenduen zerrenda zein izango zen. Azkenik, ospitaleko Informatika sailetik espero genuen erantzuna ekainean heldu zen, eskaera martxoan eginda zegoelarik.

Beste bigarren mailako helburu bat entsegu kliniko bakoitzeko heltzen diren medikamentuak erregistratzeko modua zein izango den adostea izan da. Medikamentuen erregistro hori ahalik eta bizkorrena izatea nahi da, lan-prozesua azkartzeko betiere. Komentatu zigutenez, medikamentu bakoitzak barra-kode bat zuen esleituta, farmako horren informazio guztia kodetua zuena. Horrenbestez, proposamen bezala, medikamentu horiek erregistratzeko barra-kode irakurgailu baten erabileraz baliatzea posible zen edo ez galdetu ziguten. Gure erantzuna garbia izan zen: posible zela hori egitea, baina bigarren mailako helburu bezala uztea erabaki zela. Erabaki horren arrazoiak hausnarketa sakon bat du ondorio:

Hasteko, proiektuaren inplementazioarekin hasi nintzenerako jarri zituzten oztopo guztiak eta gero, (batez ere informatikako departamentutik) lana egiteko denbora murriztea suposatu zuen, eta ez genuen zentzuzkoa ikusten barra-kode teknologia bezalako ikasi beharra zegoen teknologia berri baten erabilera gehitzea helburu nagusietara. Gainera, azterketa bat egin eta gero, ikusi genuen ez zela bakarrik Gurutzetako ospitalean erabiltzen zen barra-kode irakurgailu hori aztertu behar, baizik eta medikamentu horietan barra-kodea inprimatzen duen makinaren funtzionamendua ere aztertu behar genuela, ondo jakiteko zein informazio ezartzen zen etiketa horietan, eta zein protokolo erabiltzen zen hauek irakurri eta deskodetzeko.

Hori, ospitaleko Informatikako departamentutik jaso behar genuen erabili ahal ziren teknologien zerrendaren erantzunik ez izateari gehituta, proiektuaren lan-karga eta -denborari dagokienez, gehigarri pisutsua izango zela suposatu genuen; ahalegin gehigarri hori egin eta gero, guk erabilitako teknologiak

ospitalearentzako ezar ezinak zirela esateko aukera zegoela kontutan bageneukan baita ere; eta, horretaz gain, bertan erabiltzen zituzten kode-barra irakurgailu eta inprimatzaileak erabiltzeko baimenari itxaroten egotearen arriskua ere gehitu behar zaio, ikusirik teknologien erabileraren erantzuna jasotzeko hilabeteak itxaroten egon behar izan ginela.

Beraz, medikamentuen erregistroa kode-barra irakurgailu baten bitartez kudeatzearen helburua bigarren mailakotzat jo zen; edota proiektua bukatu ostean eta aplikazio funtzional bat eskaini ostean, proiektuaz kanpo egin daitekeen hobekuntza edo lan bat bezala kontsideratu zen.

Ondorioz, eta laburbiltzeko, adostu zen lehen mailako bukaerako helburua aplikazio funtzional bat egitea izan da, haiek eskatzen zuten baldintza eta betekizunak betetzen zituena, eta ez ezartzeko prest dagoen aplikazio komertzial baten garapenari ekitea. Horrela, aplikazio funtzional hori onartuz gero, bukaerako aplikazio komertzial hori egiteko aukera egongo litzateke, konpentsazio ekonomiko baten truke, agian. Bigarren mailako helburu bezala, beraz, aplikazioa Gurutzetako ospitalean ezartzeko ahaleginetan dihardutea pentsatu zen, bai eta medikamentuen erregistrorako barra-kode irakurgailuak erabiltzea ere.

6.2 Erabilitako Baliabideen Azalpena

6.2.1 Hardwarea

Proiektua osotzeko erabili diren baliabideen artean, Hardware elementu bakarra ordenagailua izan da, hain justu inplementazioa, diagramak eta memoria egiteko erabili den tresna baita.

6.2.2 Softwarea

Aplikazioan erabili diren software-teknologia desberdinen aukeraketak ez dira ausaz egin, eta badituzte hauen hautapenerako arrazoiak.

- *Python*: programazio-lengoaia hau aukeratzearen zergatia gaur egunean bere erabilera areagotzen ari den lengoaia bat izatearena da, batez ere. Beste alde batetik, hasiera batetan ez genekienez ziurtasunez zein teknologia erabiliko genituen aplikazioa garatzeko, ospitaleko Informatika sailetik ez gintuztelako erantzun bat eman, ahalik eta zabalenak ziren teknologiak erabiltzea adostu genuen. Beraz, *Python* lengoaia plataforma anitza izaterakoan, *Python* lengoaia-interpretea instalatu daitekeen edozein sistematan lan egitea ahalbidetzen du, besteak beste: *Linux, Windows, Mac, OpenSolaris*, etab. Horrela, etorkizunari begira, ospitalean ezartzeko baimena duen software bat garatzeko aukera gehiago zeudela suposatu genuen.

  Python-ek daukan sintaxi ulerterraz eta intuitboa dela gehitzen badiogu aipatutako guztiari, gaur egun gero eta gehiago erabili ohi den programazio-lengoaia bat dela argi dago, eta oso interesgarria da lengoaia hau ezagutzea eta menperatzea, etorkizunean egon daitezkeen lan-aukera desberdinak aprobetxatzea errazago egiteko helburuarekin.

- *HTML5*: *Flash* eta *Silverlight* teknologiak jasaten ari diren beherakada nabaria denez gero, esan ohi da *HTML*-k gero eta indar gehiago hartuko duela, eta bi lengoaia horien ondorengoa izango dela. Arrazoi horrengatik, eta eskaintzen dituen aukera anitzengatik hautatu dut lengoaia honen erabilera.

- *JavaScript*: *HTML5* lengoaiaren aukeraketa egin nuenez, *JavaScript* erabiltzearen aukeraketa ere eskutik etorri zen. Izan ere, *JavaScript HTML5*-ekin erabili daitekeen lengoaia bat da, bezeroaren aldeko programazioa egiteko erabiltzeko egokia dena web-bidezko aplikazioak egiten direnean.

Gainera, *jQuery* bezalako liburutegiak *JavaScript*-en oinarrituak daudela jakina da eta oso erabiliak dira gaur egun, kodea aurrezteko eta aukera erabilgarri eta erosoak eskaintzen dutelako. Hori dela eta, lengoaia honen erabakia *HTML5* lengoaiaren hautaketarekin batera etorri zen.

- *Django*: aukeratutako lengoaiak bateratzen dituen web-bidezko aplikazioak diseinatzeko laningurune egokia denez, erabakia bide horretatik joan zen. *Python, HTML5, JavaScript* eta *jQuery* lotzen ditu web-aplikazioak garatzeko. Esan behar da, *Django* ezagutzen ez nuen teknologia bat zela, *Python*-ekin gertatzen zen bezala, eta bere erabilera nolakoa zen, nola lan egin behar zen eta nola kudeatu behar ziren lanerako erabiltzen zituen fitxategi desberdinak ikasi behar izan nuela. Horrek bere denbora eraman zidan, eta banekien, baina gaur egun erabilera areagotzen ari diren teknologiak erabiltzen zituela jakinik, aukera ezin hobea iruditu zitzaidan *Django*-rekin lan egiten ikastea; batez ere, etorkizunera begira zenbat abantaila eskaini ahal dizkidan ikusita.

  Horretaz aparte, *Model-View-Controller* patroia erabiltzen zuela ikusi nuen, interfaze bisualeko aplikazioetan oso erabilia den patroi bat delarik. Horrek ere animatu zidan *Django* erabiltzen ikastera.

- *SQLite* 3: Datu-baseak kudeatzeko sistema hau aukeratzearen zergatia da bezero-zerbitzari erakoak diren datu-baseak kudeatzeko sistemekin ez bezala, *SQLite 3* programarekin barneratuta dagoen datu-base bat izaterakoan, egiten dituen eragiketak eraginkorragoak eta azkarragoak direla (*MySQL* edo *PostgreSQL*-rekin konparatuz, adibidez). Gainera, datu-basearen eramankortasuna ere aipatzekoa da, plataforma anitzetan lan egin dezakeelako, eta datu-basearen beraren eramankortasuna inolako konfigurazio edo administraziorik gabe egin daitekeelako. Horretaz gain, domeinu publikokoa izaterakoan, edonork erabiltzeko aukera dauka, edozein delarik bere erabileraren helburua ere.

  Esan beharra dago, *Django*-k erabiltzen duen datu-base lehenetsia izanik *SQLite 3*, eta inolako instalaziorako konfiguraziorik behar ez duela abantaila bezala jarrita, lehenengo prototipo funtzionalerako datu-base hori erabiltzea pentsatu dela. Baina, etorkizunera begira, *PostgreSQL* datu-baseak kudeatzeko sistema aplikatzea jarri dudala helburu gisa.

  PostgreSQL etorkizunean erabiltzearen zergatia:

  Datu-baseak kudeatzeko sistema honen etorkizunerako aukeraketaren zergatia, aplikazioan erabiliko den datu karga handia izango dela eta kontsultak konplexuak izango direla izan da. Hain justu, datuen kudeaketarako sistema honek ziurtasun eta eraginkortasun handia eskaintzen du datu asko gordeta duten datu-baseekin lan egiteko, bai eta datuak eskuratzeko egin behar diren kontsultak konplexuak direnean.

  MySQL datu-baseak kudeatzeko sistema kontutan izana bazen ere, esan beharra dago egokiagoa dela kontsulta sinpleak egin behar diren datu-baseetarako.

  Gainera, *PostgreSQL*-k eskaera asko dituzten web-aplikazioetan erabiltzeko aproposa dela diote adituek, eta jakinik entsegu klinikoak kudeatzeko aplikazioa jende askok aldiberean erabiltzen dutela, kontsulta asko eginez, aproposena datu-baseen kudeaketarako sistema hori erabiltzea dela iruditu zait.

  Datuen integritatearen ziurtapenerako eta kontsulta konplexuen exekuziorako eraginkorra eta fidagarria da, eta denboran zehar izan dituen aurrerapenei esker, kontsultak egiterako orduan, itxarote-denbora murriztea lortu da, eskaera azkarrak egitea lortuz.

- *Visual Paradigm*: Hain justu mota askotako diagramak egiteko aukera eskaintzen duelako, modu argi eta intuitibo batetan, izan da software hau aukeratzearen zergatia. Erabilpen Kasuen Diagramak eta Domeinu-ereduaren Diagramak egiteko erabilterraza denez gero, aplikazio hau izan da aukeratua horien diseinua burutzeko.

- *DigitalOcean*: Frogak egiterako orduan, aplikazioa online uztea pentsatu zen, edozein lekutatik atzigarri edukitzeko. Horrekin lortu nahi zena zera zen: Gurutzetako ospitaleko Farmaziako departamentuan benetako kasuekin frogak egin ahal izatea. Horretarako, departamentuko ordenagailuetatik aplikaziorako atzipena izatea ahalbidetzeko, sareko zerbitzari bat erabiltzea pentsatu zen. *DigitalOcean* erabiltzearen arrazoi nagusia, erraz ulertzekoa den interfaze bat izateaz gain, bi hilabeteko froga egiteko dohaineko aukera ematen zuela izan zen. Hain justu, ezartzerako orduan Gurutzetako ospitaleko zerbitzari batetan instalatuko denez, ez genuen behar denbora gehiago frogak egiteko. Horregatik, besteak beste, aukeratu zen DigitalOcean erabiltzea.

- *Overleaf*: Dokumentuak lantzeko LaTeX erabiltzen duen web-gune bat da. Gune hau erabiltzea pentsatu zen memoria editatzeko, ez zegoelako LaTeX erabiltzeko ezer instalatu beharrik, eta editatzen ari den dokumentuaren unean uneko emaitza erakusten duelako, erdibitutako pantaila baten bitartez (alde batetan kodea, eta bestean dokumentuaren emaitza). Erabilterraza izateagatik eta ezer instalatu beharra ez egoteagatik aukeratu zen Overleaf.

- *Cacoo*: Prototipo digitala egiteko web-bidezko software erabilterraz eta eraginkorra dela iruditu zitzaidan. Gainera, pantailen arteko nabigazioa simulatzeko egokia dela ikusi nuen, ezer instalatu beharra ez zegoela, eta interfaze bisuala diseinatzeko ere hainbat aukera eskaintzen zituela. Horretaz aparte, prototipoaren esteka publikoa edo pribatua jartzeko aukera zuen, nahi bezalako pribatutasuna ezartzea posible izanik. Horregatik aukeratu nuen Cacoo.

- *Stack Overflow*: Oso aukera egokia iruditu zait web-gune horretan parte hartzea, proiektuan zehar izan ditudan zalantzei dagokienez, arazoen gainean tutoreari zuzenean galdetu beharrean, *Stack Overflow*-ko erabiltzaileei zalantzak proposatuz, eta erantzun posibleak haiekin eztabaidatuz.

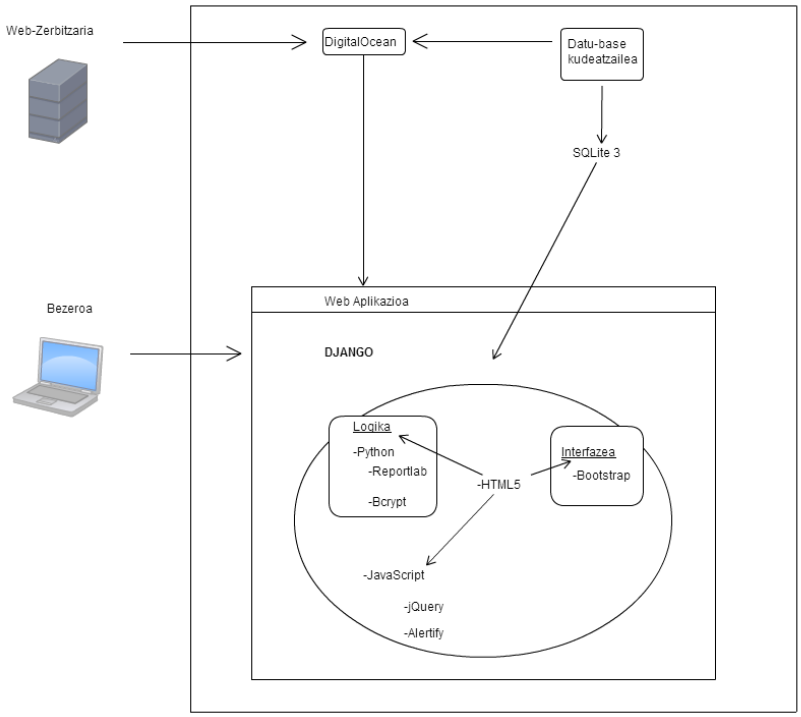
  Erabaki horrekin, esan beharra daukat hainbat puntu lortu ditudala, bai niri emandako erantzunen gainean komentarioak egiteagatik, bai galdera asko proposatu eta beste erabiltzaileen gustukoak izateagatik. Modu horretan, ikasketa autonomoa landu dut.

Gurutzetako ospitaleko Informatikako departamentuarekin ekainean izandako batzarrean, zein teknologia mota erabili zitezkeen azaldu zen etorkizunean softwarea ospitalean instalatu nahi izanez gero. Batzar hori teknologien aukeraketa egin eta gero gertatu zen, baina ikusi zen lantzeko adostutako teknologien aukeraketa egokia izan zela.

Adierazten zuten betekizunetako bat izan zen softwarea ahalik eta irekiena izan behar zuela, *Windows* sistema-eragilean erabiltzeko gaitasuna izateko, eta *Internet Explorer 8* edo *Firefox* nabigatzaileetan erabiltzea posible egiten zuena. Gainera, datu-baseak kudeatzeko sistemari dagkionez, Gusutzetako ospitalean *Oracle* eta *SQL Server* lizentziak zutela adierazi zuten. Baldintza hauek betetzen zirela ikusi zen, edozein nabigatzailerekin konpatibilitatea erakusten baitu aplikazioak, bai eta datu-basearen lengoaia *SQL* izaterakoan, ospitaleak eskatzen zuen datu-baseak kudeatzeko mekanismoekin bateragarria zela ikusi zintekeen.

Programazio-lengoaiei zegokienez, *Python, PHP* edo *HTTPS* izan zitekeela esan zuten, software librea erabiltzea izanik adierazpidea. Beste alde batetik, birtualizaziorako teknologiaren bat erabiltzekotan, *VMWare* zegoela erabilgarri adierazi zuten. *Python, HTML* eta *JavaScript* lengoaiak software librekoak direnez gero, baita ere alor horren gainean haiek ezarritako betekizunak betetzen zirela ikusi zen.

Horretaz gain, Gurutzetako ospitalean erabiltzen den *e-Osabide* lan egiteko sarearekin integratzeko beharrik ez izatea eskatzen zuten. Noski, aplikazioa beste edozein software-arekiko independentea izango zenez gero, baldintza hau betetzen zuen.



Irudia 30: Aplikazioan erabilitako teknologia eta baliabideen arkitekturaren diagrama

Irudian ikusten da aplikazioa garatzeko erabili diren teknologia eta baliabideen arteko loturak eta erlazioak. Alde batetik *DigitalOcean*, web-zerbitzari bat, erabili dut aplikazioa instalatzeko eta martxan jar daiten urrutitik, inolako makina fisiko batetan instalatuta egoteko beharra ez izanik.

Aplikaziorako *SQLite 3* datu-baseak kudeatzeko sistema erabili dut; eta aplikazioaren beraren garapen zuzenerako: alde batetik, *Python* erabili da funtzionalitateen, pantaila desberdinen estekarako kudeaketaren, eta aplikazioaren gaineko konfigurazio orokorraren inplementazioa egiteko. *Reportlab* erabili da errezeta klinikoen informazioa PDF formatuan gordetzeko, eta *Bcrypt* izeneko paketea pasahitzak zifratzeko.

Horretaz gain, interfazearen eta funtzionalitateen koderen arteko lotura eratzeko, *HTML5* lengoaia erabili da, *JavaScript* lengoaiak eskaintzen dituen abantailen laguntzarekin. *jQuery* liburutegia erabili da hainbat funtzio laburbiltzeko, eta, batez ere, *Alertify* izeneko liburutegia informazio- eta konfirmazio-mezuak adierazteko.

Aplikazioaren interfazea dotoretzeko eta itxura aldatzeko *Twitter*-ek eskaintzen duen *Bootstrap* erabili da, hainbat ikono desberdin eta gainontzeko interfazerako elementu desberdinentzako itxura-aldaketarako aukera eskaintzen dituena.

7 Kudeaketa

Proiektua aurrera eramateko erabilitako prozesuari daokionez, bi bloke eta metodologia nagusi bereizi dira, proiektuaren garapenean suertatu diren oztopoak direla eta.

Lehenengo blokea, proiektuaren betekizunen bilketari, analisiari eta diseinuari dagokio. Kasu honetan, 'urjauziaren' modeloa (*Waterfall*) erabili dut; hau da, atal bat bukatu aurretik ez nintzen hurrengo planifikatuta neukan atalarekin hasten. Fase honetan metodologia hau jarraitzea garrantzitsua izan da, izan ere, pausu bat aurrekoarekin lotuta zihoalako, eta aurreko eginkizuna bukatzeak asko laguntzen zuelako hurrengoa egitera.

Hori horrela, lehenik eta behin Gurutzetako ospitaleko Farmaziako departamentutik eskatzen ziren betekizunak jaso nituen, batzar luze batetan; jarraian, Erabilpen Kasuen Diagrama egiten hasi nintzen, jasotako betekizun horien artean zeuden funtzionalitateak errepresentatzeko. Pausu hori egin eta gero, Juanan Pereira tutorearen zuzenketak jaso eta bukatutzat jota zegoenean Domeinu-ereduaren Diagramarekin hasi nintzen lanean; hemen ere, zuzenketa guztiak aplikatu arte diagramarekin lanean jarraitu nuen.

Betekizunak eta analisiko diagramak eginda eta adostuta zeudela, diseinuarekin hasi nintzen. Kasu horretan, egindako diagramak asko lagundu zuten, paperezko prototipo edo zirriborro bat egiteko, bai eta ondoren etorri zen prototipo digitala egiteko. Izan ere, prototipo digitalaren azkeneko bertsioa adostu arte, Alazne Bustinza, ospitaleko Farmaziako departamentuko arduradunarekin bilera batzuk izan nituen, aspektu bisual, nabigazio eta funtzionalitateen gainean adostasun batetara heltzeko.

Nahiz eta banekien prototipoan landutako interfazean aldaketak egongo zirela azkeneko aplikazioari zegokienez, oso garrantzitsua izan zen hori egitea, eta horretarako, betekizunen bilketa, analisia eta diseinua eginda izatea, pausuz pausu joan eta gero, hasierako planteamendurako helburu hori betetzen lagundu zuelako.

Bigarren fasean aplikazioaren inplementazioa egon zen. Lehenik eta behin, *Python*-i buruzko gidaliburu batzuk begiratu eta sarean eskaintzen ziren tutorial batzuk egin nituen, programazio-lengoaiaren erabilera ikasteko eta menperatzen hasteko. Hori egindakoan, *Django* instalatzeko eta web-guneak eraikitzeko ingurunearekin lan egiten ikasteko tutoriala jarraitu nuen.

Hori bukatutakoan, inplementazioarekin hasi nintzen, baina aurreko ataletan erabilitako urjauziaren lan-metodologia erabili beharrean, modelo 'iteratibo' bat jarraitu nuen; hau da, astero batzar bat egiten nuen Gurutzetako ospitaleko Farmaziako departamentuan, aplikazioaren garapena nola zihoan komentatzeko, eta horretarako, ez nuen atal bat bukatu arte atal horretan lanean diharduteko lan-metodoa jarraitzen, baizkik eta eginda neukanaren gainean, batzarrean proposatutako hobeuntzak aplikatzen saiatu eta atal berriak inplementatzen jarraitzen nuen.

Modu horretan, kodetuta nituen atalak ez nituen bere horretan uzten, baizik eta batzarrean erakutsi eta hitzegindako aspektuen gaineko hobekuntzak edo gehikuntzak tratatzen nituen egindakoaren gainean, atal berri batzuk inplementatzen jarraitzearekin batera. Metodo honekin, aplikazioak, bukatzerakoan izango zuen sendotasuna handiagoa izatea lortu nahi nuen.

Esan beharra dago, ospitalean izandako batzarretaz eta aipatutako beste guztiaz gain, proiektuaren eboluzioan jarraitasun bat eramatea garrantzitsua izan dela. Horretarako, *GitHub* tresna erabili dut, proiektuaren gaineko guztia gordetzen joateko. Diagramak eta memoriarako txantiloiak aldatzen nituenean igtzen nituen *Git*-era, bai eta kodearen gainean inplementazio-karga dexentea nuen bakoitzean ere. Hau da, kodean egiten nuen aldaketa bakoitzeko ez nuen *Git*-era igotzen, baina bai funtzionalitate berri batzuk gehitu, ezabatu edota moldaketa nabariak egiten banituen.

Modu horretan lortu nahi nuena zen aplikazioaren bertsio desberdinak izatea eskuragarri *GitHub*-en, unean lantzen nenbilen kodea ordenagailuaren memoria fisikoan ere baneukalarik.

Gainera, Juanan Pereira proiektuaren tutorearekin kontaktuan egon naiz, epe ertainen buruan proiektuaren garapeneko hasierako fasean, aldaketa nabarienak edo fase bakoitzeko atalen baten amaieran iritzia eskatzeko, eta epe laburragoan proiektuaren azken txanpan, azkenengo hobekuntzak, komentarioak eta laguntzarako irizpideak eskatzeko.

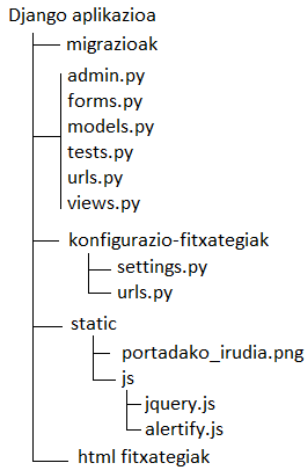
Bukatzeko, esan beharra daukat, proiektuaren kudeaketa eta garapenerako jarraitutako metodologiak asko lagundu didala, eta aldez-aurretik egindako plangintza sakon baten ondorio izan dela, oztopoak oztopo plangintza horretan aldaketak edo atzerapenak egon badira ere, antolakuntza orokorra ez baita asko aldatu, eta bukaerako helburuak lortzen lagundu dit.

8 Inplementazioko Atal Garrantzitsuenak

Web-aplikazioa garatzerako orduan inplementazioan garrantzitsuenak eta beharrezkoenak izan diren atalak azalduko dira jarraian, Domeinu-ereduaren Diagraman aipatzen diren entitate-erlazioak adierazten dituen modeloen egituraketa eta Erabilpen Kasuen Diagraman adierazten diren funtzionalitateei erreferentzia egiten dizkien kode atalak, hain zuzen.

Horretarako, lehenik eta behin, *Django* web-aplikazioak garatzeko inguruneak lan egiteko jarraitzen duen estrategia eta modulu-banaketa azaldu beharra dago. Model-View-Controller patroia jarraitzen duela komentatu dudan heinean, *Django*-k hainbat direktorio erabiltzen ditu modelo horretako atal bakoitzeko elementuak multzokatzeko eta haiekin lan egiteko.

Jarraian, *Django*-k moduluak nola antolatzen dituen azaltzen duen grafiko bat adieraziko dut, aplikazio honetarako erabilitako moduluak erakusten direlarik.



Irudia 31: *Django*-ren aplikazioaren barneko egitura

'settings.py' deritzon fitxategi bat erabiltzen da *Django*-rekin garatutako aplikazio bakoitzean, aplikazio horren konfigurazioa egituratzeko. Konfigurazio-fitxategi honetan, besteak beste, aspektu hauek tratatzen dira: erabiliko den datu-base lengoaia; pasahitzak gordetzeko eta kodetzeko erabiliko den zifratze-metodo; aplikazioaren garapenerako instalatu diren aplikazioen zerrenda zein den; aplikazioa "debug"(edo frogak egiteko bertsio) moduan abiarazi nahi den edo ez; aplikaziorako beharrezkoak diren fitxategi batzuen path-ak; etab.

Aipatu beharra dago, fitxategi desberdinen path-ak, lan egiten ari den makina fisikoaren menpe egon ez daitezen, *Django*-k aldagai berezi batzuk jartzen dituela eskuragarri, helbide horien orokortze bat egiteko, eta proiektua beste edozein makinatara migratuz gero, helbideak aurkitzean arazorik eman ez dadin.

Jarraian, beste fitxategi garrantzitsu baten erabilera azaldu beharra daukat: 'urls.py' fitxategia. Bi 'urls.py' fitxategi mota bereiztu behar dira hainbat aplikazio gorde ditzakeen *Django* karpetan: aplikazio desberdin guztientzat orokorra dena, eta aplikazio konkretu bati erreferentzia egiten diona.

Proiektuarentzat orokorra den fitxategian, aplikazio bakoitza abiarazteko zein url jarri behar den adierazten da.

Berriz, aplikazio konkretu bati erreferentzia egiten dion "urls.py"fitxategian, aplikazioari dagozkien url guztiak agertzen dira adierazita, url bakoitzak funtzio konkretu bat esleituta duelarik eragiketa bat edo bestea egitea ahalbidetzeko, eta horretaz gain url horri esleitzen zaion izen bat. Modu honetan, zerbitzaritik aplikaziora konektatzerakoan, bakarrik esteka edo url horietatik nabigatzea izango da posible. Jarraian ikus daitekeen "urls.py"fitxategiaren kodean agertzen den bezala, kasu batzuetan, url-a bistaratzerakoan exekutatu behar den kodea abian jartzen denean parametro gisa balio batzuk bidaltzea beharrezkoa izan da; hori egiteko ondorengo sintaxia erabiltzen da: '(?P<parametroa>\w+)'.

```
from django.conf.urls import patterns, url
from farmaciapp import views

#Web-aplikazioko orrialdeen url-ak definitzen dira hemen
urlpatterns = patterns('',
        url(r'^$', views.index, name='index'),

        url(r'^erregistratu/$', views.erregistratu, name='erregistratu'),

        url(r'^login/$', views.erabiltzailea_login, name='login'),

        #...Aplikazioaren funtzionalitateei erreferentzia egiten
        #dien gainontzeko estekak...

        url(r'^aukera_menua/erabiltzaile_menua/erabiltzailea/
        (?P<erabiltzailea>\w+)/erabiltzailea_eguneratu/$',
        views.erabiltzailea_eguneratu, name="erabiltzailea_eguneratu"),

        url(r'^aukera_menua/erabiltzaile_menua/erabiltzailea/
        (?P<erabiltzailea>\w+)/erabiltzailea_ezabatu/$',
        views.erabiltzailea_ezabatu, name="erabiltzailea_ezabatu"),

        url(r'^aukera_menua/erabiltzaile_menua/erabiltzaile_gehiketa/$',
        views.erabiltzailea_gehitu, name="erabiltzaile_gehiketa"),

        url(r'^aukera_menua/erabiltzaile_menua/erabiltzaile_bilaketa/$',
        views.erabiltzaileak_bilatu, name="erabiltzaile_bilaketa"),

)
```

Aplikazio konkretu baten barruan dagoen beste fitxategi interesgarri bat "admin.py"fitxategia da. Fitxategi honetan, aplikazioarekin frogak egiterako orduan administratzaile bezala kudeatzeko aukera eskaintzen duen beharrezko kodea dago. Hots, fitxategi honetan "models.py"fitxategian dauden eta nahi diren entitateak ezartzen dira, aplikazioarekin zerikusirik ez daukan beste pantaila batetan, aplikazioan zehar sortu diren elementuak kudeatu daitezkeelarik (datuak aldatu, ezabatu eta sortu). Horrela, aplikazioa bera probatzerakoan erroreen gaineko kontrol hobea bermatzen da.

```
from django.contrib import admin
from farmaciapp.models import Medikamentua, Pazientea, Dispentsazioa,
Ensaioa, ErabiltzaileProfila, PazienteEnsaio, EnsaioErrezeta,
PazienteDispentsazio, MedikamentuEnsaio

#Hemen erregistratzen dira gure modeloak,
#administratzailearen orrian kudeatu ahal izateko

admin.site.register(Medikamentua)
admin.site.register(Pazientea)
admin.site.register(Dispentsazioa)
admin.site.register(Ensaioa)
admin.site.register(ErabiltzaileProfila)
admin.site.register(PazienteEnsaio)
admin.site.register(EnsaioErrezeta)
admin.site.register(PazienteDispentsazio)
admin.site.register(MedikamentuEnsaio)

)
```

Jarraian, web-aplikazioaren interfazea, funtzionalitate eta entitate-erlazioekin zerikusia duten atalik garrantzitsuenak azalduko ditut.

Hasteko, modeloarekin nola egiten duen lan eta hau nola egituratu dudan aplikazioa garatzeko azalduko dut. "models.py"fitxategia erabiltzen du datu-basean gordeko diren entitateak eta bere atributuak adierazteko, bai eta entitateen arteko erlazioak definitzeko ere. Domeinu-ereduaren Diagraman aurkeztu diren entitate eta erlazioez gain, beste modelo desberdin bat ere definitu dut, *Django*-k defektuz beharrezkotzat jotzen duena aplikazioaren erabilerarako: 'ErabiltzaileProfila'.

```
#Erabiltzailearen profilaren informazioa gordeko duen entitatea da
class ErabiltzaileProfila(models.Model):
    # Lerro hau beharrezkoa da. ErabiltzaileProfila linkatzen du
    #User modeloaren instantziarekin.
    erabiltzailea = models.OneToOneField(User)

    # Gehitu nahi diren atributuak
    #website = models.URLField(blank=True)
    #picture = models.ImageField(upload_to='profile_images', blank=True)
    izena = models.CharField(max_length=128)
    abizena1 = models.CharField(max_length=128)
    abizena2 = models.CharField(max_length=128)

    #Bi aukera egongo dira eskuragarri: farmaziako langilea
    #edo erabiltzaile arrunta.
    zerbitzua = models.CharField(max_length=128, default=1,
    choices=(('Farmazia', 'Farmazia'), ('Medicina', 'Medicina')))

    #Adibidez, onkologia, radiologia, etab.
    azpizerbitzua = models.CharField(max_length=300, null=True,
    blank=True)

    # modelo honen instantzia bat atzitzen denean zein izenekin
    #definituko den instantzia hori adierazten du
    def __unicode__(self):
        return self.erabiltzailea.username
```

Azpitik, *Django*-k, erabiltzaile desberdinak kudeatzeko modelo bat du definituta, "User"modeloa, hain zuzen, erabiltzaile-izena, pasahitza eta email helbidea gordeko dituena; "ErabiltzaileProfila"modeloak, aldiz, erabiltzaile baten gainontzeko informazio pertsonala gordeko du: izena, lehenengo abizena, bigarren abizena, zein zerbitzutakoa den erabiltzailea eta zein azpizerbitzutakoa; azken eremu hau jartzearen arrazoia ondorengoa da: erabiltzaile arrunta edo 'Medicina' zerbitzukoa izatekotan, konkretuki zein alorrekoa den esan beharra dago (adibidez, 'erradiologia'), mota honetako erabiltzaileek entseguen bilaketak egiten dituztenean soilik bere azpizerbitzuarekin zerikusia duten entseguak agertuko baitzaie.

Eremu hauek guztiak beharrezkoak direla kontsideratu ditut, erabiltzailearen gaineko gutxieneko informazioa gordeta izateko kontrol hobeago batentzat, eta lan egiten duen zerbitzua ezartzea beharrezkoa delako baimenak esleitzerako orduan. Ikus daitekeenez, 128 karaktereko muga ezarri da karaktere-testu bakoitzarentzat; eta "zerbitzua"atributua ez da izango idazkera askeko eremu bat, aukera-eremu bat baizik ("Farmazia"eta "Medicina"izatea gustatuko litzatekeela adostu da, Gurutzetako ospitaleko Farmaziako departamentutik).

Horretaz gain, Domeinu-ereduaren Diagraman aipatutako entitate eta erlazioei dagozkien modeloak definitu dira, dagozkien atributuekin. Jarraian, modelo horien kode atalik esanguratsuenak daude, modu horretan sortzearen arrazoiak justifikatuz:

```
#Pazienteari dagokion entitatea eta bere erlazioak
class Pazientea(models.Model):
    #ident izango da Pazientearen gako nagusia, idensaioarekin batera
    ident = models.AutoField(primary_key=True)

    #entseguaren barruan zein id duen adierazteko;
    #gako nagusiaren parte ere izango da
    idensaioan = models.CharField(max_length=128, blank=True)
    izena = models.CharField(max_length=128)

    #zein unitate klinikori dagokion paziente hau adierazteko
    unitateKlinikoa = models.CharField(max_length=128, blank=True)

    #pazienteari dagozkion beharrezko datuak:
    pisua = models.FloatField()
    #datu gehiago behar badira hemen jartzen dira



    def __unicode__(self):
        return unicode(self.idensaioan)
```

Paziente baten entitatea adierazten duen entitate horretan, atributu autoinkrementala bat erabiltzen da gako nagusi bezala. Horren justifikazioa da hainbat paziente desberdin egoteaz gain, entsegu desberdinetara lotuta egon daitezkeen paziente berdinak egon daitezkeela, eta horrenbestez, izena ez zela aukera egokiena gako bezala jartzeko. Beste modu batzuetara egin badaiteke ere, kodean informazioaren eskuraketa errazteko hartutako neurri bat da, entitate desberdinen arteko erlazio konplexuak ahalik eta gehien ekiditu nahirik.

```
#Pazienteak eta Dispentsazioak lotzen dituen entitatea da
class PazienteDispentsazio(models.Model):
    #identifikatzaile autoinkrementala hau izango da gako nagusia
    identBi = models.AutoField(primary_key=True)
    #identifikatzaile hau Dispentsazioaren identifikatzailearen
    #berdina izango da

    ident = models.IntegerField(default=1)
    medikamentua = models.ForeignKey(Medikamentua, null=True)
    dispentsazioa = models.ForeignKey(Dispentsazioa, null=True,
    related_name="dispentsazioa_pazientearekiko")
    paziente = models.ForeignKey(Pazientea, null=True,
    related_name="pazientea_dispentsazioan")
    dosia = models.IntegerField(null=True, blank=True)

    #PazienteDispentsazioren instantzia eskuratzen denean
    #identifikatzailearen bidez izendatuko da;
    #hau da, dispentsazioaren identifikatzailearen bitartez
    def __unicode__(self):
        return unicode(self.ident)
```

Kode-zati honetan ikus daitekeen atributuen etiketa garrantzitsu bat 'related_name' da. Horrek adierazten duena da etiketa hori duen atributuak erreferentzia egiten dion objektu batetatik, etiketa hori jarrita dagoen atributua duen entitateko atributuetara atzitu daitekeela. Hau oso garrantzitsua izan da, entitateen artean gehiegizko erlazioak ez jartzeko, eta nahi ziren emaitzak lortu ahal izateko. Adibide horretan, 'Paziente' entitatearen objektu batetik, 'PazienteDispentsazio' entitateko atributuera atzitu daiteke, dosia lortu nahi baldin bada, adibidez.

```
#Ensaioak eta Errezetak lotzen dituen entitatea da
class EnsaioErrezeta(models.Model):
    #identifikatzaile autoinkremental bat izango da gako nagusia
    ident = models.AutoField(primary_key=True)
    ensaioa = models.ForeignKey(Ensaioa, null=True)
    pazientea = models.ForeignKey(Pazientea, null=True)
    preskripzioData = models.DateField(null=True)
    hurrengoPreskripzioData = models.DateField(null=True, blank=True)

    #Aldagai honek kontrolatuko du zein erabiltzailek sortu duen errezeta
    sortzailea = models.CharField(max_length=128, blank=True, null=True)

    #Aldagai honek esango du ea errezeta 'Pendiente' egoeran dagoen edo ez;
    #hau da, onartuta dagoen edo ez
    pendiente = models.CharField(max_length=128, default='Pendiente')

    #Eremu zabal eta ireki bat izango da nahi diren eremuak adierazteko
    gainontzekoEremuak = models.TextField(null=True, blank=True)

    #Errezetaren izena zein izango den adierazteko
    errezetaIzena = models.CharField(max_length=300, blank=True, null=True)


    #EnsaioErrezetaren instantzia eskuratzen denean identifikatzailearen
    #bidez izendatuko da
    def __unicode__(self):
        return unicode(self.ident)
```

Erlazio horretan, entsegu baten gainean egiten den errezeta baten hasierako egoera adierazten da. Errezeta berri bat sortzen denean, defektuz 'Pendiente' egoeran ezarri behar da, eta onartzen denean, egoera hori 'Dispentsatuta'-ra pasatzen da.

"models.py"fitxategian definitu diren entitate eta erlazioen instantziak sortzeko eta haien gaineko bilaketak egiteko, *Django*-k beste tresna erabilgarri bat eskaintzen du: formularioak definitzeko balio duen "forms.py"fitxategia. Fitxategi horretan, modelo horien gaineko formularioak definitu daitezke, zein eremu agertzea nahi diren adieraziz, zein ez, eremu berriak gehituz, etab. Oso baliagarria da, abantaila garrantzitsu bat eskaintzen duelako: entitate konkretu baten modeloaren inplementazioan definitutako derrigorrezko eta ez-derrigorrezko eremuak, eremu horien gordetze motak (zenbakiak, karaktere-kateak, datak, etab.), eta gainontzeko ezaugarriak betetzen dituela erabiltzaileak kontrolatzeaz arduratzen da. Modu horretan, ez da funtzionalitateen kodea konprobaketaz arduratzen den inplementazioaz bete behar, eta garbiago geratzen da.

Jarraian, bilaketarako erabili dudan formulario batzuen eta entitateen instantziak sortzeko erabili ditudan formulario batzuen adibideak daude, bere azalpenarekin, eta formularioen egituraketarako

erabili ditudan elementu nagusien azalpenekin.

- ErabiltzaileFormularioa eta ErabiltzaileProfilFormularioa: Bi formulario hauek erabiltzaileari buruzko informazioaren gainekoak dira. Esan bezala, *Django*-k berak 'User' objektu bat dauka, erabiltzaile bat erregistratzeko logika gordetzen duena eta 'username', 'password' eta 'email' atributuak dauzkana. Berriz, erbailtzailearen gaineko informazio gehiago sartzeko aukera izateko, ErabiltzaileProfilarentzako formulario bat egitea pentsatu zen.

```
class ErabiltzaileFormularioa(forms.ModelForm):
    password = forms.CharField(widget=forms.PasswordInput())

    class Meta:
        model = User
        fields = ('username', 'email', 'password')


#Hau bakarrik izango litzateke erabiltzaileak bere profila
#izan nahiko balu.
class ErabiltzaileProfilFormularioa(forms.ModelForm):

    def __init__(self, *args, **kwargs):
        super(ErabiltzaileProfilFormularioa, self)
        .__init__(*args, **kwargs)
        self.fields['azpizerbitzua'].required = False


    class Meta:
        model = ErabiltzaileProfila
        fields = ('izena', 'abizena1', 'abizena2', 'zerbitzua',
        'azpizerbitzua')
```

'class Meta:' lerroa jartzea derrigorra da, formulario horrek zein Modelo erreferentziali egoten dion erreferentzia zehazteko.

'def __init__(self, *args, **kwargs):' lerroaren bitartez formulario horren instantziazio bati pasatu nahi zaizkion argumentuak jasotzeaz arduratzen da, bai eta modeloan definitutako atributu edo eremuen gainean aldaketaren bat egon nahi baldin bada (adibidez, ikusgarri ez egotea, edo bete beharreko eremua dela definitzea, besteak beste).

- EnsaioBerriFormularioa: Entseguak sortzeko erbailtzen den formulario honetan, hasierako eta bukaerako datari erreferentzia egiten dien atributuei klase-balio bat esleitu diet, HTML orrialdean (interfaze bisualari dagokionez) kalendarioaren widget-a agertzea posible egiteko, gero azalduko dudan bezala. Ikus daitekeen bezala, baita ere, 'bukaeraData' eta 'egoera' atributuei HiddenInput widget-a esleitu diet, ez egoteko ikusgarri, ez direlako entsegu baten sorreran erabiltzaileak idatzi beharreko eremuak.

```python
#Ondorengo formularioa entsegu berri bat sortzeko erabiliko da
class EnsaioBerriFormularioa(forms.ModelForm):


    def __init__(self, *args, **kwargs):
        super(EnsaioBerriFormularioa, self).__init__(*args, **kwargs)
        self.fields['bukaeraData'].required = False
        self.fields['bukaeraData'].widget.attrs={'class':'vDateField'}
        self.fields['hasieraData'].widget.attrs={'class':'vDateField'}
        self.fields['bukaeraData'].widget = forms.HiddenInput()
        self.fields['egoera'].widget = forms.HiddenInput()
        self.fields['egoera'].required = False

    class Meta:
        fields = '__all__'
        model = Ensaioa
```

- EnsaioBilaketaFormularioa: Kasu honetan, bilaketak egiteko formulario bat denez, ez dago eremu bakar bat betetzeko beharrik erabiltzailearen aldetik; hautazkoa da eremu horiek betetzea, bilaketa zehatzago bat egin nahiko baluke, adibidez. Horregatik, atributuek required = False dute esleituta, balio bat definitzea beharrezkoa ez dela adierazteko. Horretaz gain, data atributuek klasea esleituta daukate eta 'egoera' eta 'komentarioak' atributuak ez dira ikusgarri egongo bilaketan, ez direlako bete beharreko eremuak izango; hau da, irekita dauden entseguak bilatuko direnez, erabiltzaileak ezingo du definitu zein egoeratako entseguak bilatu nahi dituen, eta komentarioen eremua hautazkoa denez, ezta ere.

```
#Ondorengo formularioa entseguen bilaketak egiteko erabiliko da
class EnsaioBilaketaFormularioa(forms.ModelForm):

    def __init__(self, *aEnsargs, **kwargs):
        super(EnsaioBilaketaFormularioa, self).__init__(*args,**kwargs)
        self.fields['egoera'].required = False
        self.fields['egoera'].widget = forms.HiddenInput()
        self.fields['hasieraData'].required = False
        self.fields['bukaeraData'].widget.attrs={'class': 'vDateField'}
        self.fields['hasieraData'].widget.attrs={'class': 'vDateField'}
        self.fields['protokoloZenbakia'].required = False
        self.fields['titulua'].required = False
        self.fields['zerbitzua'].required = False
        self.fields['promotorea'].required = False
        self.fields['estudioMota'].required = False
        self.fields['monitorea'].required = False
        self.fields['monitoreaFax'].required = False
        self.fields['monitoreaEmail'].required = False
        self.fields['monitoreaTel'].required = False
        self.fields['monitoreaMugikor'].required = False
        self.fields['ikertzailea'].required = False
        self.fields['komentarioak'].widget = forms.HiddenInput()

    class Meta:
        fields = '__all__'
        model = Ensaioa
```

- ErrezetaBerriEnsaiotikFormularioa: Komentatu beharreko azkeneko atala formulario honetan agertzen da. Ikus daitekeenez, filtro bat gehitu da atributu baten gainean; honen arrazoia da filtrorik egon ezean aplikazioan erregistratuta dauden paziente guztiak agertuko litzatekeela hautapen-lista batetan, eta bakarrik erabiltzailea aztertzen ari den entseguaren gainekoak interesatzen direla. Gainera, ikus daitekeenez ensaioa_protkolo_zenb parametroa pasatu zaio, filtroa definitzeko beharrezkoa den balioa gordetzen duena.

```
class ErrezetaBerriEnsaiotikFormularioa(forms.ModelForm):

    preskripzioData = forms.DateField(widget=forms.DateInput())
    hurrengoPreskripzioData = forms.DateField(widget=forms.DateInput())
    def __init__(self, ensaioa_protokolo_zenb, *args, **kwargs):
        super(ErrezetaBerriEnsaiotikFormularioa, self)
        .__init__(*args, **kwargs)

        self.fields['ensaioa'].required = False
        self.fields['pazientea'].required = True
        #Bilatzen ari den entseguko pazienteak agertzea bakarrik
        #ahalbidetu behar da
        self.fields['pazientea'].queryset = Pazientea.objects.filter
        (pazientea_pazienteensaion__ensaioa__protokoloZenbakia
        =ensaioa_protokolo_zenb)

        self.fields['preskripzioData'].required = True
        self.fields['preskripzioData'].widget.attrs
        ={'class': 'vDateField'}

        self.fields['hurrengoPreskripzioData'].widget.attrs
        ={'class': 'vDateField'}

        self.fields['hurrengoPreskripzioData'].required = False
        self.fields['pendiente'].required = False
        self.fields['pendiente'].widget = forms.HiddenInput()
        self.fields['ensaioa'].widget = forms.HiddenInput()
        self.fields['sortzailea'].widget = forms.HiddenInput()
        self.fields['errezetaIzena'].required = False
        self.fields['errezetaIzena'].widget = forms.HiddenInput()



    class Meta:
        fields = '__all__'
        model = EnsaioErrezeta
```

Ondoren, aplikazioak izango dituen funtzionalitateen kodea gordetzen duen fitxategia azalduko dut: "views.py". Fitxategi horretan url konkretu batetarako *HTTP* eskaera bat egiten denean zein funtzio egin behar den zehazteaz arduratzen den kodea dago, eskaera horretan bidalitako parametroak jasotzen eta interpretatzen dituena, eta zein orrialdetara berbideratu behar den kodea interpretatu eta gero adierazten duena.

"urls.py"fitxategiaren azalpenean esan bezala, esteka edo url bakoitzak bista edo view konkretu bat dauka esleituta, eskaera prozesatzeko. "views.py"fitxategian da, hain zuzen, formularioetatik jasotzen den informazioa jasotzeaz, *HTTP* eskaerak eta erantzunak kudeatzeaz, eta datu-basea behar bezala kudeatzeaz arduratzen den elementua. Horregatik da hain garrantzitsua. Beste modu batetara azalduta, aplikazioak behar dituen beste fitxategi guztien informazioa bateratzeaz, koordinatzeaz eta kudeatzeaz arduratzen den fitxategia da "views.py".

Funtzionalitateen inplementaziorako, hainbat elementu inportatzea izan da garrantzitsua, esan

bezala, formularioak, modeloak eta bestelako elementuak konbinatzen dituen fitxategia baita.

```
from django.shortcuts import render
from django.contrib.auth import authenticate, login, logout
from django.contrib.auth.decorators import login_required
from django.http import HttpResponseRedirect, HttpResponse

#Formularioak
from farmaciapp.forms import ErabiltzaileFormularioa,
ErabiltzaileProfilFormularioa, EnsaioBerriFormularioa,
EnsaioBilaketaFormularioa, EnsaioBilaketaFormularioa2,
MedikamentuBilaketaFormularioa, MedikamentuBilaketaFormularioa2,
Medikamentua, ErrezetaBerriFormularioa, DispentsazioFormularioa,
ErrezetaBerriEnsaiotikFormularioa, MedikamentuBerriFormularioa,
ErrezetaModifikatuFormularioa, PazienteBerriFormularioa

#Modeloak
from farmaciapp.models import Ensaioa, ErabiltzaileProfila,
PazienteEnsaio, Pazientea, EnsaioErrezeta, MedikamentuEnsaio,
Dispentsazioa, PazienteDispentsazio

#Django-ren defektuzko modeloak
from django.db.models import Q, Sum
from django.contrib.auth.models import User
import datetime
from dateutil import parser
import time

from random import randint
import sys


#PDF dokumentuen sorrerarako beharrezkoak
from reportlab.pdfgen import canvas
from django.http import HttpResponse
from io import BytesIO
```

Gainera, hainbat bistatan '@login_required' lerroa ageri da. Lerro horrek adierazten duena da erabiltzaileak sesioa hasita izan behar duela derrigorrez funtzionalitatea aurrera eramateko. Hori, *Django*-k automatikoki kudeatzen duen sesio-hasierarako Cookie-en bitartez kontrolatzen da.

Hori aipatuta, esan beharra dago orain arte azaldutakoarekin aplikazioaren modeloa eta kontrolatzaileari egiten diogula erreferentzia, datu-basean informazioa nola gordeko den eta kudeatuko den azalduz, bai eta funtzionalitate desberdinak nola egituratzen diren eta interfaze bisualarekin nola komunikatzen diren azalduz. Baina, hain justu, bista edo interfazearen informazioa azaltzea ere falta da, *Model-View-Controller* patroia jarraituz.

Horretarako, "templates"direktorio bat dauka *Django*-k, interfaze bisuala konposatuko duten *HTML* fitxategiak gordeko dituena. Lehen aipatu bezala, bista edo view bakoitzaren amaieran, *HTML* orrialde batetarako berbideraketa dago, informazioa adierazteko zein pantaila erakutsi behar den adierazten duena, eta hain justu, fitxategi horiek guztiak "templates"karpetan daude gordeak.

Django-k, "views.py"fitxategian idatzitako *Python* kodea "templates"direktorian gordetzen diren *HTML* fitxategietan interpretatzeko, giltzak ('{}') erabiltzeko aukera eskaintzen du. Giltza bikoitzak erabiltzen baldin badira *HTML* kodearen barruan, orrialde horretara berbideratu duen view-ak erantzunean bidali dituen elementuak eskuratu daitezke.

Ondorengo adibidean azalduko den bezala, bistatik 'erregistratu.html' fitxategira erantzunean hainbat balio bueltatzen dira, eta 'erregistratu.html' fitxategian, jasotzen dira:

```
#Bistatik html fitxategira berbideratu parametroak bidaliz
return render(request,
              'farmaciapp/erregistratu.html',
              {'pendienteak':errezeta_pendienteak.count,
              'erabiltzaile_form': erabiltzaile_form,
              'erabiltzaile_profil_form': erabiltzaile_profil_form,
              'erregistratuta': erregistratuta} )

 {{ erabiltzaile_form.as_p }}

 #HTML orrialdean jasotako emaitza:
 {{ erabiltzaile_form.as_p }}
```

Baina ez hori bakarrik; dagokion bista edo view-tik jasotzen den informazioa tratatu daiteke baita ere; modu hauetan:

```
{% if user.is_authenticated %}

        <form id="erabiltzaile_formularioa1" method="post"
        action="/farmaciapp/erregistratu/" enctype="multipart/form-data">

            {% csrf_token %}

            {{ erabiltzaile_form.as_p }}
            {{ erabiltzaile_profil_form.as_p }}
            <input type="checkbox" name="admin" value="admin">
            Erabiltzaile hau Administratzailea da<br><br>

            <input class="btn btn-primary" type=button
            onclick="konfirmazioa('erabiltzaile_formularioa1')"
            value="Erregistratu"/>
        </form>

{% endif %}
```

Ikus daitekeen bezala, baldintzako sententziak ('if', 'else') jartzea ahalbidetzen du *Django*-k *HTML* kodean zuzenean.

Django-k *HTML* fitxategiekin lan egiteko duen moduan, formularioei buruzko atal bat aipatu beharra dago. *Django*-ren formularioekin lan egiteko {% csrf_token %} kode-zatia gehitzea derrigorrezkoa dela, goiko adibidean adierazten den bezala. Kode-zati hori oso garrantzitsua da, 'POST' bidez bidaltzen diren datuak modu seguru batetan gordetzeaz gain, datu horien jatorria babesten duelako; horrela,

beste erabiltzaileren batek *JavaScript* kodean erasoren bat egin nahi izanez gero datuak eskuratzeko edo orrialde konkretu batetako informazioa eskuratzeko, babestuta geratuko litzateke.

Beste alde batetik, *Twitter*-ek libre eskaintzen duen Bootrstrapping metodoak ere erabiltzen dira botoien, menuen eta gainontzeko elementuen forma eta kolorea definitzeko. Ikonoak ('glyphicon' balioa duten klase-etiketadun elementuak) eta botoien formak ('btn' erakoak diren botoien klase-atributuaren balioak) adierazteko erabili dira kodean zehar batez ere.

**HTML* fitxategiei buruz komentatu beharreko atal bat da '<body>' etiketaren hasieran ia fitxategi guztiek jarraitzen dutela patroi berdina, elementu berdinekin konposatuta. Nahiz eta orokortzea pentsatu zen hasiera batetan, ikusi zen ez zela eraginkorra izango, orrialde bakoitzean badaudelako konpartitzen den patroi honen barruan orrialde bakoitzari dagokien elementu propioak; adibidez, atzera joateko estekak.

Horretaz aparte, aipatu behar da *HTML* fitxategiek amankomunean izan ditzaketen atal bakarra *Twiter Bootstrap*-en *script*-en deklarazioak direla, eta horiek orokortuz *HTML* fitxategi batetan, ez direla interfazearen elementuak behar bezala kargatzen.

```
<head>
    <link rel="icon" href="http://getbootstrap.com/favicon.ico">
    <link href="http://getbootstrap.com/dist/css/bootstrap.min.css"
    rel="stylesheet">
    <link href="http://getbootstrap.com/examples/dashboard/dashboard.css"
    rel="stylesheet">

    {% load staticfiles %}

    <link rel="stylesheet"
    href="//code.jquery.com/ui/1.11.4/themes/smoothness/jquery-ui.css">
    <script src="//code.jquery.com/jquery-1.10.2.js"></script>
    <script src="//code.jquery.com/ui/1.11.4/jquery-ui.js"></script>
    <script src="{% static "js/jquery-farmaciapp.js" %}"></script>
    <script src="{% static "js/alertify.min.js" %}"></script>
    <link rel="stylesheet" href="{% static "js/alertify.core.css" %}" />
    <link rel="stylesheet" href="{% static "js/alertify.default.css"%}" />

    <title>Titulua</title>


</head>
```

Beraz, atal hau horrela uztea pentsatu da html orrialde bakoitzean pertsonalizatuz, kode zati handiegia ez dela kontutan harturik.

Aipatutako *HTML* fitxategietako script desberdinak '/static/js/' direktorian aurkitzen dira. Bertan 'jquery-1.11.1.js' fitxategia eta 'jquery-farmaciapp.js' fitxategiak arduratzen dira *HTML* fitxategian definituta dauden datari erreferentzia egiten dioten atributu aukeragarrietan egutegien widget-ak jartzeaz. Horretarako garrantzitsua izan da 'forms.py' fitxategian datari erreferentzia egiten dien atributuei klase konkretu bat jartzea, script orokor horretatik atzitzeko. Gainera, 'alertify.js' fitxategi librea erabili da konfirmazio- eta informazio-mezu desberdinen adierazpenerako.

Fitxategi horietarako atzipena adierazteko, *HTML* fitxategietako {% load staticfiles %} lerroa era-

biltzen da, 'settings.py' fitxategian adierazten den estekari erreferentzia egiteaz arduratzen dena.

Jarraian, aplikazioaren funtzionalitate nagusien bistak, horietan erabilitako errekurtso garrantzitsuenak eta horiei lotuta dauden *HTML* orrialdeak nolakoak diren eta nola egin diren azalduko dira:

- Login:

```
<form id="login_form" method="post" action="/farmaciapp/login/">
      {% csrf_token %}
      Erabiltzailea: <input type="text" name="username"
      value="" size="50" autofocus/>
      <br /><br />
      Pasahitza: <input type="password" name="password"
      value="" size="50" />
     <br /><br />

          <input class="btn btn-primary" type="submit" value="Hasi" />
</form>
```

'login.html' fitxategiak egiten dio erabiltzailearen sesio-hasiera pantailari erreferentzia. Orrialde horren konposaketa sinplea da, formulario bakar bat baino ez duelako. Bertan erabiltzailea eta pasahitza sartzeko baino ez da eskatzen. 'action' etiketak adierazten duen url-a, aurretik aipatutako aplikazioaren 'urls.py' fitxategian definituta dago, eta esteka horrek adierazten duen bistara bidaliko da *HTTP* 'POST' eskaera, berri dagozkion parametroekin.

```
#Erabiltzaileak login egiteko beharreko bista
def erabiltzailea_login(request):

    #POST bada erabilitako HTTP metodoa
    if request.method == 'POST':
        #Erabitzailearen erabiltzaile-izena eta pasahitza
        #jasotzen dira eskaeratik
        username = request.POST.get('username')
        password = request.POST.get('password')

        #Django-k konprobatzen du ea autentikazioa zuzena den
        user = authenticate(username=username, password=password)

        #Erabiltzaile objektua zuzena bada, ongi hasi du sesioa
        if user:
            #Kontua irekita badago
            if user.is_active:
              #Sesioa hasterakoan erabiltzailearen menu nagusira
              #eramango zaio erabiltzaileari
              login(request, user)
              return HttpResponseRedirect('/farmaciapp/aukera_menua/')
            else:
                #Kontua ez badago aktibatuta
                return HttpResponse("Zure kontua desgaituta dago...")
        else:
            #Ez badu ongi hasi sesioa
            print "Sesio-irekiera desegokia: {0}, {1}".format(username,
            password)

            return HttpResponse("Erabiltzailea edo pasahitza ez dira
            egokiak!<br/><br/><a href='/farmaciapp/login/'>Atzera</a>")

   #Ez bada POST eskaera bat
    else:

        return render(request, 'farmaciapp/login.html', {})
```

Horrela, kode-zati honetan ikusten den bezala, aplikazioko bista edo view guztiek jarraituko duten patroia ondorengoa da: 'POST' eskaera bat egin den edo ez konprobatu, parametroak jasotzeko; kodearen interpretazioa egin; eta dagokion *HTML* orrialdera berbideratu. Kasu honetan, *Django*-k automatikoki kudeatzen du erabiltzaile baten sesio irekiera, erabiltzaile-izen eta pasahitz batekin. Zuzena bada, aukeratzen den orrialdera berbideratuko da erabiltzailea, eta zuzena ez bada web orrialde sinple batetan adieraziko da.

- Erabiltzailea Gehitu:

  Administratzaileak erabiltzaile bat gehitu nahi duenerako, 'erregistratu.html' fitxategia erabiltzen da. Orrialde honetan, erabiltzailea autentikatuta dagen eta administratzaile motakoa den konprobatzen da funtzionalitateen zerrenda adierazteko, segurtasun aldetik hobea izateko eta babestuago egoteko. Erabitzailearen motaren azterketa ia orrialde guztietan egingo da, zein esteka eta zein funtzionalitate agertu behar diren jakiteko eta kudeatzeko.

```
<body>
<!-- ... -->
 <div class="container-fluid">
  <div class="row">
   <div class="col-sm-3 col-md-2 sidebar">
     {% if user.is_authenticated %}

       {% if admin == mota %}

         <h3>Erabiltzailea: {{ user.username }}</h3>
         <h4>Zerbitzua: {{mota}}</h4><br/>

         <a href="/farmaciapp/aukera_menua/ensaio_kontsulta/">
         Entseguak Kontsultatu  <span class="glyphicon
 glyphicon-book" aria-hidden="true"></span></a><br/>

         <a href="/farmaciapp/aukera_menua/
 errezeta_pendienteak_kontsultatu/">
         Errezeta Pendienteak Kontsultatu
         <span class="glyphicon glyphicon-pencil"
         aria-hidden="true"></span><span
         class="badge">{{pendienteak}}</span></a><br/>

         <a href="/farmaciapp/aukera_menua/
 medikamentu_kontsulta/">
         Medikamentuak Aztertu <span class="glyphicon
 glyphicon-plus" aria-hidden="true"></span></a><br/>

         <a href="/farmaciapp/aukera_menua/
 erabiltzaile_menua/">Erabiltzaileak Kudeatu
         <span class="glyphicon glyphicon-user"
         aria-hidden="true"></span></a><br/>

        {% endif %}

       {% endif %}

       <br/>

       <a href="/farmaciapp/">Atzera
       <span class="glyphicon glyphicon-arrow-left"
       aria-hidden="true"></span></a><br/>

       <a href="/farmaciapp/">Hasiera <span class="glyphicon
 glyphicon-home" aria-hidden="true"></span></a>

        </div>

       <div class="col-sm-9 col-sm-offset-3
 col-md-10 col-md-offset-2 main">
```

```
<h1>Erregistratu zaitez</h1>

{% if erregistratuta %}
        <strong>Eskerrik asko erregistratzeagatik!</strong>
        <a href="/farmaciapp/">Bueltatu zaitez orrialde nagusira.
        </a><br/>
    {% else %}

    {% if user.is_authenticated %}

      <form id="erabiltzaile_formularioa1"
      method="post" action="/farmaciapp/erregistratu/"
      enctype="multipart/form-data">

       {% csrf_token %}

       {{ erabiltzaile_form.as_p }}
       {{ erabiltzaile_profil_form.as_p }}
       <input type="checkbox" name="admin"
       value="admin">Erabiltzaile hau Administratzailea da
       <br/><br/>

       <input class="btn btn-primary" type=button
       onclick="konfirmazioa('erabiltzaile_formularioa1')"
       value="Erregistratu"/>
      </form>

        {% endif %}
  {% endif %}
          </div></div></div>
</body>
```

Kasu honetan, bi aukera desberdin daude: erregistroa burututa egotea (erregistroa egiten denean orrialde berdinera berbideratzen baitzaio erabiltzaileari) edo erregistroa eginda ez egotea. Erabiltzailea ongi sortu bada, mezu bat bistaratuko da hori adierazteko; bestelako kasuan, formulario bat adieraziko da. Kasu horretan, bistatik jasotzen ditu erabiltzaile_form eta -erablitzaile_profil_form aldagaiak, formularioak bistaratzeko, automatikoki.

```
@login_required
def erabiltzailea_gehitu(request):

    #Erabiltzaile berri bat gehitzeko pantaila erakutsiko da

    erabiltzaile_form = ErabiltzaileFormularioa()
    erabiltzaile_profil_form = ErabiltzaileProfilFormularioa()

    #Jakiteko zein motako erabiltzailea den
    erabiltzaile_mota
    = ErabiltzaileProfila.objects.filter(erabiltzailea
    =request.user)[0].zerbitzua
    #aldagai hau erabiliko da html-an konprobazioa egiteko
    admin = 'admin'

    erregistratuta = False

    #Pendiente dauden Errezetak aterako dira
    errezeta_pendienteak
    = EnsaioErrezeta.objects.filter(Q(pendiente='Pendiente'))


    return render(request,
            'farmaciapp/erregistratu.html',
            {'pendienteak':errezeta_pendienteak.count, 'admin':admin,
            'mota':erabiltzaile_mota,
            'erabiltzaile_form':erabiltzaile_form,
            'erabiltzaile_profil_form': erabiltzaile_profil_form,
            'erregistratuta': erregistratuta} )
```

Erabiltzailea gehitzeko bistan aldagai baten bitartez kudeatzen da erregistroa burutu den edo ez jakitea, eta 'POST' motako eskaera egin baldin bada, *HTML* orrialdean definitu diren formularioetan bidalitako balioak jasotzen dira beste bi aldagaitan. Horrela, konprobatu daiteke ea formulario horiek ondo bete diren edo ez (betetzea derrigorrezkoa den eremuren bat bete ez baldin bada, behar bezalako karaktereak sartu badira, etab., adibidez). Zuzena baldin bada formularioaren betetzea, erabiltzailearen informazioa gordetzen da datu-basean, pasahitzaren hash-transformazioa eginik.

Erabiltzailea autentikatuta dagoen jakiteko, *Django*-k *Python* bidezko ondorengo kodea eskaintzen du, sisteman sesio-irekiera gordetzen duen elementua jasotzeko:

```
request.user.is_authenticated()
```

Horrela jakin daiteke erabiltzailea autentikatuta dagoen edo ez, eta kasu honetan administratzailea izan beharra dauka erabiltzaileak erregistratzeko, beraz, horregatik konprobatzen da administratzailea den edo ez. Formularioa zuzena ez bada, *Django*-k eskaintzen dituen errore-mezu automatikoak pantailaratzeko kodea ezartzen da:

```
print erabiltzaile_form.errors, erabiltzaile_profil_form.errors
```

Bukatzeko, render metodoaren bitartez, dagokion web orrialdera berbideratzen da erabiltzailea, hiztegi batetan gordetzen direlarik bidali nahi diren parametroak, izena:balioa formatuan.

*Aipatu beharra dago errezeta_pendienteak aldagaia bista gehienetan eskuratzen eta bidaltzen den parametroa dela, interfazean etengabe agerian egongo den balio bat delako, eta beraz, orrialde bakoitzera berbideratzen den bakoitzean orrialde horretan agertu nahi diren elementuak bidali behar direnez gero, etengabe eskuratu beharra dago balio hau.

- Erabiltzailea Modifikatu:

  Erabiltzaile baten datuak modifikatzeko 'erabiltzaile_info.html' fitxategia dago, erabiltzailearen informazio nagusia formulario batetan kargatzen delarik, aldaketak egitea ahalbidetuz. Hemen ere, informazioa zuzen aldatzen denerako mezua erakusteko kode zati bat dago, eta eguneratu ez denerako formulario batetan erakusten da informazioa. Zerbitzua aukeratzeko eta uneko rola erakusteko eremuak eskuz gehitu dira formulariotik kanpo dauden eremuak direlako, baina erabiltzailearen informazio nagusiko formularioa bistatik atera da.

```
{% if eguneratuta %}

  <div class="alert alert-success" role="alert">
  <span class="glyphicon glyphicon-ok" aria-hidden="true"></span>
  <span class="sr-only">Informazioa: </span>
  Informazioa ondo eguneratu da!
  </div>
  <br/><br/>

  <input class="btn btn-primary" type=button name="botonAtzera"
  onClick="location.href='/farmaciapp/aukera_menua/
  erabiltzaile_menua/'" value='Atzera'>

  {% else %}

   {% if mota == admin %}
   <form id="erabiltzailea_moldatu" action="/farmaciapp/aukera_menua/
   erabiltzaile_menua/erabiltzailea/{{erabiltzailea}}/
   erabiltzailea_eguneratu/" method="POST">
      {% csrf_token %}

      {{erabiltzaile_form.as_p}}

      <b>Zerbitzua:</b>
      <select id="zerbitzua" name="zerbitzua">
       <option value = "Farmazia">Farmazia</option>
       <option value = "Medicina">Medicina (Erabiltzaile Arrunta)
       </option>
       <option value = "admin">Administratzailea</option>
      </select>

      <br/><br/>

      <b>Uneko rol-a:</b> {{ikusteko_erabiltzaile_mota}}

      <br/><br/>

        <input class="btn btn-primary" type=button
    onclick="konfirmazioa('erabiltzailea_moldatu')"
    value="Erabiltzailearen Datuak Eguneratu">
    </form>

            <br/><br/>
```

```
    {% if erabiltzailea != user.username %}

      <form id="erabiltzailea_ezabatu"
      action="/farmaciapp/aukera_menua/erabiltzaile_menua/
_______erabiltzailea/{{erabiltzailea}}/erabiltzailea_ezabatu/"
      method="POST">

       {% csrf_token %}

       <input type="hidden" name="erabiltzailea"
       value="{{erabiltzailea}}">

                <input class="btn btn-danger" type=button
       onclick="konfirmazioa('erabiltzailea_ezabatu')"
       value="Erabiltzailea Ezabatu">
                </form>

     {% endif %}

   {% endif %}

{% endif %}
```

*Aipatzea garrantzitsua den aspektu bat da formularioetan *submit* balioa duten jarri beharrean type etiketari, 'button' balioa esleitzea erabaki dela, konfirmazio mezuak jartzeko beharrezkoa delako. Horrela, *onclick* etiketa baten bitartez, eta orrialdean kargatuta dagoen script baten bitartez, konfirmazioa() metodo bati deia egiten zaio, 'static/js/' direktorian definituta dagoen 'jquery-farmaciapp.js' fitxategian definituta dagoena, eta 'alertify.js' fitxategian definitzen den konfirmazio mezuari deia egiten diona.

```
@login_required
def erabiltzailea_eguneratu(request, erabiltzailea):
    #Aldagai hau erabiliko da adierazteko ea eguneraketa
    #ondo joan den edo ez
    eguneratuta = False


    if request.method == 'POST':
     erabiltzaile_form = ErabiltzaileFormularioa(data=request.POST)

     #erabiltzaile izen bat izan behar duela konprobatzen da
     if len(request.POST['username'])>0:

         erabiltzailea_profila=
         ErabiltzaileProfila.objects.get
         (erabiltzailea__username=erabiltzailea)

         erabiltzailea_info=User.objects.get(username=erabiltzailea)

         #pasahitzaren luzeera 0 karaktere baino gehiagokoa
         #baldin bada
         if len(request.POST['password'])>0:
             erabiltzailea_info.set_password(request.POST['password'])
             #pasahitza aldatuko diogu

         if len(request.POST['username'])>0:
             #Lehenik konprobatzen da ea existitzen den
             #erabiltzaile hori

             #Eta existitzen bada,
             #ea erabiltzaile honen username zaharra den
             if request.POST['username']!=erabiltzailea:
                 try:
                   erabiltzailea_konprobatu=
                   User.objects.get(username=request.POST['username'])
                   mezua = 'Erabiltzaile hori existitzen da jada!'

                 except:
                   erabiltzailea_info.username=request.POST['username']
                   #erabiltzailea aldatuko diogu

          erabiltzailea_info.email = request.POST['email']
          erabiltzailea_info.save()

          erabiltzailea_profila.username = erabiltzailea_info
```

```
            #Ikusten dugu ea zerbitzuaren balio berria admin,
            #Farmazia edo Medikua den

            #Horietako bat baldin bada, balioa aldatuko diogu

            #Bestela, mantendu egingo diogu
            motaBerria = request.POST['zerbitzua']

            if motaBerria == 'admin' or
            motaBerria == 'Farmazia' or
            motaBerria == 'Medicina':
                #Aldatu egingo diogu balioa
                erabiltzailea_profila.zerbitzua = motaBerria

            erabiltzailea_profila.save()
            erabiltzailea_info.save()
            eguneratuta = True

        else:
            print erabiltzaile_form.errors

    else:
        erabiltzaile_form = ErabiltzaileFormularioa()

    #Jakiteko zein motako erabiltzailea den
    erabiltzaile_mota=
    ErabiltzaileProfila.objects.filter
    (erabiltzailea=request.user)[0].zerbitzua

    #aldagai hau erabiliko da html-an konprobazioa egiteko
    admin = 'admin'

    #Aplikazioan dauden erabiltzaile guztiak bilatuko dira
    bilaketa_emaitzak = ErabiltzaileProfila.objects.all()

    #Pendiente dauden Errezetak aterako dira
    errezeta_pendienteak=
    EnsaioErrezeta.objects.filter(Q(pendiente='Pendiente'))



    return render(request, 'farmaciapp/erabiltzaile_info.html',
    {'pendienteak':errezeta_pendienteak.count,
    'eguneratuta':eguneratuta,
    'bilaketa_emaitzak':bilaketa_emaitzak,
    'erabiltzailea':erabiltzailea,
    'erabiltzaile_form':erabiltzaile_form, 'admin':admin,
    'mota':erabiltzaile_mota})
```

Bista honetan ere aurretik azaldutakoen patroi berdina jarraitzen da. Aldagai baten bitartez kontrolatzen da eguneraketa zuzena izan den edo ez, eta formularioaren bidalketa zuzena bada,

jasotako informazioa tratatzera jotzen da. Kasu honetan, aipatu behar dena zera da: erabiltzaileak idazten duen erabiltzaile-izena datuen eguneraketarako ez dela hutsa, ez eta pasahitza ere ez, derrigorrezko eremuak baitira. Gainera, ikusi behar da ea erabiltzaile izen hori beste erabiltzaile-kontu bateko pertsona baten erabiltzaile-izenaren berdina den, horrela bada, ezingo litzatekeelako aldaketa onartu.

Informazio-bidalketa zuzena izan bada, datuak datu-basean gorde dagokion orrialdera berbideratuko da.

*Komentatu behar den beste atal garrantzitsu bat, bista gehienetan errepikatzen den beste kodezati bat da: erabiltzaile-mota zein den jakin beharra dago, baimenak esleitzerako orduan, *HTML* orrialdeetan kudeatzen baita egungo erabiltzaileak baimena duen edo ez orrialdeko eragiketa batzuk egiteko edo ikusteko. Horregatik bidali beharra dago parametro bezala *HTML* orrietarako berbideraketetan erabiltzaile_mota atributuaren balioa.

- Erabiltzailea Ezabatu:

  'erabiltzaile_info.html' fitxategi berdinean aurkitzen da aukera hau, eta egiten den frogapen bakarra da aztertzen ari den erabiltzailea eta sesioa irekita duen administratzaile-erabiltzailea ez direla berdinak. Horrela, erabiltzaile batek ezingo luke bere burua ezabatu sesioa irekita duela, arazoak ekiditeko eta segurtasuna handitzeko.

```
@login_required
def erabiltzailea_ezabatu(request, erabiltzailea):

    #Erabiltzailearen profila eta informazioa ezabatuko dira

    #Lehenik, erabiltzailea aurkitzen da
    erabiltzailea_profila=
    ErabiltzaileProfila.objects.get
    (erabiltzailea__username=erabiltzailea)

    erabiltzailea_profila.erabiltzailea.delete()
    erabiltzailea_profila.delete()

    #Jakiteko zein motako erabiltzailea den
    erabiltzaile_mota=
    ErabiltzaileProfila.objects.filter
    (erabiltzailea=request.user)[0].zerbitzua

    #aldagai hau erabiliko da html-an konprobazioa egiteko
    admin = 'admin'

    #Aplikazioan dauden erabiltzaile guztiak bilatuko dira
    bilaketa_emaitzak = ErabiltzaileProfila.objects.all()

    #Erabiltzailea ezabatu den edo ez jakiteko flag bat
    ezabatuta = True

    #Pendiente dauden Errezetak aterako dira
    errezeta_pendienteak=
    EnsaioErrezeta.objects.filter(Q(pendiente='Pendiente'))


    return render(request,
    'farmaciapp/erabiltzaile_kudeaketa_menua.html',
    {'pendienteak':errezeta_pendienteak.count,
    'ezabatuta':ezabatuta, 'bilaketa_emaitzak':bilaketa_emaitzak,
    'erabiltzailea':erabiltzailea, 'admin':admin,
    'mota':erabiltzaile_mota})
```

Bistari dagokionez, funtzionamendua sinplea da, erabiltzailea aurkitzen baitu, url-an doan eta bistari bidaltzen zaion parametroaren bidez, eta datu-basetik ezabatzeaz arduratzen baita.

- Ensaioak Bilatu:

```
<form id="ensaioak_bilatu" method="POST"
action="/farmaciapp/aukera_menua/ensaio_kontsulta/ensaio_bilaketa/"
enctype="multipart/form-data">

    {% csrf_token %}

    {{ ensaio_bilaketa_form.as_p }}
    {{ ensaio_bilaketa_form2.as_p }}

    <input class="btn btn-primary" type="submit"
    name="bilatu_ensaioak" value="Bilatu"/>

    </form>
    <br/>

    {% if mota == admin %}

      {% if bilaketa_emaitzak %}

      <!--Entzeguen bilaketaren emaitza lortzen da, baina -->
      <!--ezabatzeko aukera ere egonik -->

     {% else %}

         <!--Emaitzarik aurkitu ez dela adierazten duen kodea-->
       {% endif %}


       {% for e in ensaio_lista %}
          <div class="alert alert-success" role="alert">
          <span class="glyphicon glyphicon-ok"
          aria-hidden="true"></span>
          <span class="sr-only">Informazioa: </span>
          {{e}} ezabatu da
          </div><br/>
        {% endfor %}
```

```
        {% else %}

          {% if bilaketa_emaitzak %}
              Irekita dauden entsegu kopurua: {{zenbatEnsaio}}<br/>
              {% for e in bilaketa_emaitzak %}
                <li>
                <a href="/farmaciapp/aukera_menua/ensaio_kontsulta/
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣ensaio_bilaketa/ensaioa/{{␣e.protokoloZenbakia␣}}">
                {{ e.protokoloZenbakia }}</a><br/>
                </li>
              {% endfor %}

            {% else %}

              <!--Emaitzarik aurkitu ez dela adierazten duen kodea-->
            {% endif %}

    {% endif %}
```

Entseguak bilatzeko fitxategia 'ensaioak_bilatu.html' da, eta bertan, entseguak bilatzeko formularioa erakusten duen formularioa definitzen da, edozein erabiltzailerentzat. Baina, erabiltzailea administratzailea baldin bada, bilaketaren ondorioz sortutako emaitzak combo box batzuen bitartez aukeragarriak izango dira, eta entseguak ezabatzeko botoi bat ere egongo da. Hori kudeatzeko, bilaketa-emaitzak eta combo box-ak dauden kode-zatia formulario batetan sartu da. Gainera, behin ezabaketa egin eta gero, ensaio_lista izeneko parametro batetan bidaltzen denez ezabatu diren entseguen zerrenda, hori pantailaratuko da mezu baten bitartez.

```
@login_required
def ensaioak_bilatu(request):

    #kodearen inplementaziorako lagunduko duten aldagai laguntzaileak
    #...

        #Begiratzen da ea pazientea espeifikatu den edo ez
    #Pazientearen arabera bilatu nahi izan bada
    if 'pazientea' in request.POST:
        pazientea_id = request.POST['pazientea']
        try:
            #Definitutako Pazientea existitzen bada
            pazientea = Pazientea.objects.get(ident=pazientea_id)
        except:
            #Ez baldin bada existitzen pazientea ez dela definitu
            #jakina da
            pazientea_id = -1
            pazientea = None
```

```
if request.method == 'POST':
    ensaio_bilaketa_form=
    EnsaioBilaketaFormularioa(data=request.POST)

    ensaio_bilaketa_form2=
    EnsaioBilaketaFormularioa2(data=request.POST)

#Baldintza hauek betetzen baldin badira,zuzena da informazio bidalketa

     if ((ensaio_bilaketa_form.is_valid()
     or request.POST['titulua'] != '')
     and ensaio_bilaketa_form2.is_valid()):
       if(pazientea!=None):
        #Medicina arlokoa bada, bakarrik bere
        #azpizerbitzuko entseguak ikusi ahalko ditu

        #Erabiltzaile mota aztertu eta medikua ez denez,
        #entsegu guztiak lortzen dira
         erabiltzaile_mota=
         ErabiltzaileProfila.objects.filter
         (erabiltzailea=request.user)[0]
          #Bilaketak egiteko paziente bat espezifikatu
          #duen edo ez; eta hasierako data edota bukaerakoa
          #definitu diren edo ez begiratzen dira
            bilaketa_emaitzak = #...bilaketarako-kodea...

         else:
         #Medicina arlokoa bada, bakarrik bere
         #azpizerbitzuko entseguak ikusi ahalko ditu
           erabiltzaile_mota=
           ErabiltzaileProfila.objects.filter
           (erabiltzailea=request.user)[0]
           if erabiltzaile_mota.zerbitzua == 'Medicina':
             #Bilaketak egiteko paziente bat espezifikatu
             #duen edo ez; eta hasierako data edota bukaerakoa
             #definitu diren edo ez begiratzen dira
             #medikua denez, bakarrik bere azpizerbitzuko
             #entseguak eskuratzen dira
                  bilaketa_emaitzak = #...bilaketarako-kodea...
       else:
            bilaketa_emaitzak = []
    else:
        ensaio_bilaketa_form = EnsaioBilaketaFormularioa()
        ensaio_bilaketa_form2 = EnsaioBilaketaFormularioa2()
        bilaketa_emaitzak = []

   #Emaitza bidali
```

HTML fitxategi hori kudeatuko duen bistan, ordea, hainbat atal azaldu behar dira. Bilaketak egiteko, lehenik eta behin, paziente bat aukeratu den edo ez ikusi beharra dago, jakiteko bilaketa

egiteko filtroan pazientea kontutan hartu behar den edo ez. Pazientea ez baldin bada existitzen, dagokien aldagaiei balio batzuk ematen zaizkie hori adierazteko. Jarraian informazioaren bidalketa zuzena izan den edo ez ikusten da, definitutako titulua hutsa bada ere ontzat emanik bilaketa.

Behin informazioaren bidalketa zuzena dela jakinda, bi alderdi ditu kodeak: lehenengo alderdian begiratzen da ea erabiltzailea 'erabiltzaile arrunta' edo 'Medicina' motakoa den, horrela izanez gero, bakarrik erabiltzailearen zerbitzukoak diren entseguak agertuko direlako emaitzan. Bigarren alderdian, ordea, edozein zerbitzutako entseguak eskuratuko dira bilaketa-emaitzan. Gainera, alderdi bakoitzean, paziente konkretu bat definitu den edo ez bilaketa eremuetan begiratu behar da, emaitza lortzeko bilaketa-eskaeran kontutan hartzeko edo ez; azkenik, bloke hauetako bakoitzarekiko, definitutako data-eremuen araberako bilaketa guztia egiten dira, zein eremu bete diren bilaketan jakin beharra baitago bilaketa egiteko.

- Ensaioa Itxi:

```
@login_required
def ensaioa_itxi(request, ensaioa_protokolo_zenb):
    #Entsegua itxiko da, bere itxiera data eguneko data jarriz

    #Eguneko data lortzen da
    eguneko_data = time.strftime("%Y-%m-%d")

    #Entsegua eskuratu eta egoera 'itxita' balioarekin ezartzen da
    ensaioa = Ensaioa.objects.get(protokoloZenbakia=
    ensaioa_protokolo_zenb)
    ensaioa.bukaeraData = eguneko_data
    ensaioa.egoera = 'itxita'
    ensaioa.save()

    #Jakiteko zein motako erabiltzailea den
    erabiltzaile_mota = ErabiltzaileProfila.objects.filter
    (erabiltzailea=request.user)[0].zerbitzua

    #aldagai hau erabiliko da html-an konprobazioa egiteko
    farmazia = 'Farmazia'
    admin = 'admin'

    ensaio_bilaketa_form = EnsaioBilaketaFormularioa()
    ensaio_bilaketa_form2 = EnsaioBilaketaFormularioa2()

    #Pendiente dauden Errezetak aterako dira
    errezeta_pendienteak = EnsaioErrezeta.objects.filter
    (Q(pendiente='Pendiente'))


    return render(request, 'farmaciapp/ensaioak_bilatu.html',
    {'pendienteak':errezeta_pendienteak.count, 'admin':admin,
    'mota':erabiltzaile_mota, 'farmazia':farmazia,
    'ensaio_bilaketa_form':ensaio_bilaketa_form,
    'ensaio_bilaketa_form2':ensaio_bilaketa_form2})
```

Entsegua ixteko botoia 'ensaioa_info_especial.html' fitxategian aurkitzen da. Botoi honek eskaintzen duen funtzionalitatearen logikari dagokionez, entseguaren protokolo zenbakiaren laguntzaz, entsegua eskuratzen da, eta bere egoera 'itxita' balioarekin ezartzen da, 'bukaeraData'-ren balioa uneko data izanik.

- Errezeta Berria Sortu:

```
{% if mota != farmazia %}
   {% if sortuta %}
      <div class="alert alert-success" role="alert">
      <span class="glyphicon glyphicon-ok"
      aria-hidden="true"></span>
      <span class="sr-only">Informazioa: </span>
      Errezeta ondo sortu da!
      </div>

      <br/><br/>

      <input class="btn btn-primary" type=button
      name="botonAtzera" onClick="location.href='..'"
      value='Atzera'>

      <br/><br/>

    {% else %}

       Entsegua: {{titulua}}
       <br/><br/>

       <form id="errezeta_sortu_ensaiotik" method="post"
       action="/farmaciapp/aukera_menua/ensaio_kontsulta/
        ensaio_bilaketa/ensaioa/{{titulua}}/
        errezeta_berria/" enctype="multipart/form-data">

          {% csrf_token %}

          {{ errezeta_form.as_p }}

          <input class="btn btn-primary" type=button
          onclick="konfirmazioa('errezeta_sortu_ensaiotik')"
          name="sortu_errezeta_berria_ensaiotik"
          value="Sortu Errezeta"/>

       </form>

       <br/><br/>
```

```
        <form id="paziente_berria_erregistratu" method="post"
        action="/farmaciapp/aukera_menua/errezeta_sorkuntza/
␣␣␣␣␣␣␣paziente_berria_botoia/" enctype="multipart/form-data">

            {% csrf_token %}

            <input type="hidden" name="eratorpena"
            value="errezeta_ensaiotik">

            <input type="hidden" name="ensaioa"
            value="{{titulua}}">

            <input class="btn␣btn-primary" type="submit"
            name="botonPazienteBerria"
            value="Paziente␣Berria␣Erregistratu" />  

            <button type="button" class="btn␣btn-info"
            onClick="informazioa('Sisteman␣paziente␣berria
␣␣␣␣␣␣␣␣␣␣␣erregistratzeko␣aukera,␣paziente␣horri␣dagokion
␣␣␣␣␣␣␣␣␣␣␣informazioa␣definiturik.')">

                <span class="glyphicon␣glyphicon-info-sign">
                </span>
            </button>

        </form>

    {% endif %}

  {% else %}

    <div class="alert␣alert-warning" role="alert">
      <span class="glyphicon␣glyphicon-ban-circle"
      aria-hidden="true"></span>

      <span class="sr-only">Oharra: </span>
      Ez duzu baimenik errezetarik sortzeko.
    </div>

    <br/>

    <a href="..">Atzera <span class="glyphicon␣glyphicon-
␣␣␣␣␣arrow-left" aria-hidden="true"></span></a>

    <br/>


  {% endif %}
```

'errezeta_ensaiotik.html' fitxategia da errezetak sortzeko pantailaren kodea gordetzen duena. Kasu horretan, begiratzen da ea farmaziako den edo ez erabiltzailea, kasu horretan, 'Farmazia' motakoa izanez gero, ezingo duelako errezetarik sortu. Beste edozein motatakoa denean, ordea, hainbat atal hartu behar dira kontutan: sortuta dagoen (errezeta ongi sortu delako mezua pantailaratzeko) edo ez dagoen (errezeta sortzeko beharrezko formularioa agertarazteko).

Bigarren egoera horretan, bi formulario desberdin proposatu dira: bata errezetaren sorrera aurrera eramango duen estekari deia egiten diona, eta bestea, paziente berri bat erregistratzeko behar den estekari deitzen diona. Formularioak sortzearen zergatia, konfirmazio mezuak erabiltzeko behar den kodea errazagoa bihurtzen dela da, botoi sinpleak erabiliko balira baino.

Erabiltzailea 'Farmazia' motakoa balitzateke, baimenik ez duela adierazten duen mezu bat erakutsiko da pantailan.

Hori kudeatzeko kodeari dagokionez, atal bat da garrantzitsua eta aipatu beharrekoa. Errezeta sortzeko formularioaren instantzia sortzen denean ondorengo lerroan, ensaioa_protokolo_zenb parametroa pasatu beharra dago, errezeta horri dagokion entseguaren protokolo zenbakiaren balioarekin:

```
errezeta_form = ErrezetaBerriEnsaiotikFormularioa(data=request.POST,
ensaioa_protokolo_zenb=ensaioa_protokolo_zenb)
```

Horrela, 'forms.py' fitxategian definituta dagoen formularioan balio hori jaso daiteke, eta errezetarako pazientea aukeratzeko eremuan bakarrik entsegu horretakoak diren pazienteak agertzea lortzen da, entseguaz kanpo dauden pazienteak hautatzea ezinezkoa eginik. Hori garrantzitsua da, ez bailitzateke hori kontrolatuko, edozein pazienteren gainean definitu ahalko bailitzateke errezeta, eta hori ez litzateke zuzena izango.

- Paziente Berria Gehitu:

```
{% if erregistratuta %}

        <div class="alert alert-success" role="alert">
     <span class="glyphicon glyphicon-ok" aria-hidden="true">
     </span>
     <span class="sr-only">Informazioa: </span>
        Pazientea ondo erregistratu da!
   </div>

{% else %}
   <form id="paziente_berria_sortu" method="post"
   action="/farmaciapp/aukera_menua/errezeta_sorkuntza/
   paziente_berria/" enctype="multipart/form-data">

    {% csrf_token %}

    {{ paziente_form.as_p }}
    <input type="hidden" name="ensaioa" value="{{titulua}}">

    <input class="btn btn-primary" type=button
    onclick="konfirmazioa('paziente_berria_sortu')"
    name="sortu_paziente_berria" value="Erregistratu Pazientea"/>

   </form>

   <br/><br/>

{% endif %}
```

Paziente berri bat gehitzearen aukeran, *HTML* kodea 'paziente_berria_sortu.html' fitxategian aurkitzen da, eta bi bloke desberdin ditu: erregistroa burutu bada, pazientea sisteman erregistratu delako mezua adierazten duena; eta pazientea sortzeko formularioa erakusten duena. Bigarren horretan, entseguaren identifikatzailea zein den bidaltzen da baita ere formularioan, beharrezkoa izango delako pazientea entsegura esleitzeko.

```
@login_required
def paziente_berria_erregistratu(request):
    #Paziente berria erregistratzeko logika

    #Pendiente dauden Errezetak aterako dira
    errezeta_pendienteak = EnsaioErrezeta.objects.filter
    (Q(pendiente='Pendiente'))

    #pazientea ondo erregistratu den edo ez adieraziko duen aldagaia
    erregistratuta = False

    #Jakiteko zein motako erabiltzailea den
    erabiltzaile_mota = ErabiltzaileProfila.objects.filter
    (erabiltzailea=request.user)[0].zerbitzua

    #aldagai hau erabiliko da html-an konprobazioa egiteko
    farmazia = 'Farmazia'
    admin = 'admin'

    mezua = 'Pazientea ez da ondo sortu'
    if request.method == 'POST':
        #paziente berria sortzen da emandako datuekin

        paziente_form = PazienteBerriFormularioa(data=request.POST)
        if paziente_form.is_valid():

            paziente_berria =
            Pazientea(idensaioan=request.POST['idensaioan'],
            izena=request.POST['izena'],
            unitateKlinikoa=request.POST['unitateKlinikoa'],
            pisua=request.POST['pisua'])

            paziente_berria.save()

            ensaioa = Ensaioa.objects.get
            (protokoloZenbakia=request.POST['ensaioa'])

            pazienteensaio =
            PazienteEnsaio(pazientea=paziente_berria,
            ensaioa=ensaioa,
            idensaioan=request.POST['idensaioan'])

            pazienteensaio.save()

            mezua = 'Pazientea ondo sortu da'
            erregistratuta = True
```

```
        else:
            print paziente_form.errors
            return render(request,
            'farmaciapp/paziente_berria_sortu.html',
            {'pendienteak':errezeta_pendienteak,
            'mota':erabiltzaile_mota, 'farmazia':farmazia,
            'admin':admin, 'titulua':request.POST['ensaioa'],
            'erregistratuta':erregistratuta,
            'paziente_form':paziente_form, 'mezua':mezua})

    else:
        paziente_form = PazienteBerriFormularioa()
        return render(request,
        'farmaciapp/paziente_berria_sortu.html',
        {'pendienteak':errezeta_pendienteak,
        'mota':erabiltzaile_mota, 'farmazia':farmazia,
        'admin':admin,'titulua':request.POST['ensaioa'],
        'erregistratuta':erregistratuta,
        'paziente_form':paziente_form, 'mezua':mezua})


    #Errezetarako pazienteen zerrenda egiteko erabiliko da
    ensaioa_protokolo_zenb = request.POST['ensaioa']

    errezeta_form =
    ErrezetaBerriEnsaiotikFormularioa
    (ensaioa_protokolo_zenb=ensaioa_protokolo_zenb)

    return render(request,
    'farmaciapp/paziente_berria_sortu.html',
    {'pendienteak':errezeta_pendienteak.count,
    'mota':erabiltzaile_mota, 'farmazia':farmazia,
    'admin':admin,'titulua':request.POST['ensaioa'],
    'errezeta_form':errezeta_form,
    'mezua':mezua,'erregistratuta':erregistratuta})
```

Pazientea erregistratzeko kodean, berriz, pazientearen sorrerarako bete behar ziren eremuak ondo bete direla konprobatzen da, eta hori zuzena baldin bada, pazientea sortzera jotzen da. Kasu honetan, ez da konprobatu behar pazientea existitzen den edo ez, instantzia bakoitza bereiziko duen gako nagusien balioa autoinkrementala delako, eta erabaki hori, izen berdineko paziente bat hainbat entsegutan egon daitekeela eta entsegu berdinean hainbat errezetatan egon daitekeela jakina delako da. Pazientea sortutakoan, entsegua eskuratzen da formularioan bidalitako entseguaren identifikatzailearekin, eta 'PazienteEnsaio' motako instantzia berri bat sortzen da, paziente horrek entseguan duen id-a esleituz.

- Errezeta Onartu:

```
<form id="errezeta_onarpena"
action="/farmaciapp/aukera_menua/errezeta_pendienteak_kontsultatu/
errezeta/{{errezeta_ident}}/errezeta_onartu/" method="POST">

    {% csrf_token %}

    <input type="hidden" name="ensaioa" value="{{ensaioa}}">
    <input type="hidden" name="pazientea" value="{{pazientea_ident}}">
    <input type="hidden" name="preskripzioData"
    value="{{preskripzioData}}">
    <input type="hidden" name="hurrengoPreskripzioData"
    value="{{hurrengoPreskripzioData}}">
    <input type="hidden" name="gainontzekoEremuak"
    value="{{gainontzekoEremuak}}">

    Aukeratu dispentsaziorako medikamentuak:

    <br/><br/>

    {% if not ensaioaren_medikamentuak %}

        <div class="alert␣alert-warning" role="alert">
        <span class="glyphicon␣glyphicon-exclamation-sign"
        aria-hidden="true"></span>
        <span class="sr-only">Informazioa: </span>
        Entseguak ez dauka medikamenturik esleituta.
        </div>


     {% else %}

        {% for em in ensaioaren_medikamentuak %}

            <br/><input type="checkbox" name="medikamentua"
            id="medikamentua" value="{{em.medikamentua.ident}}" />

            <label for="medikamentua"><b>{{ em.medikamentua.ident }}
            </b></label>
                
            Unitateak: <input type="number"
            name="{{em.medikamentua.ident}}"
            value="{{em.medikamentua.unitateak}}" min="1"
            max="{{em.medikamentua.unitateak}}">

        {% endfor %}

    {% endif %}
```

```
    <br/><br/>

    <input class="btn btn-primary" type=button
    onclick="konfirmazioa('errezeta_onarpena')"
    value="Errezeta Onartu eta Dispentsatu"> 

   <button type="button" class="btn btn-info"
   onClick="informazioa('Errezeta onartuko da eta dispentsazio bezala
   erregistratuko da. Onartua izateko derrigorrezkoa da gutxienez
   medikamentu baten unitate bat aukeratzea.')">

   <span class="glyphicon glyphicon-info-sign"></span>
   </button>
  </form>
```

Errezeta onartzeko interfazearen kodea 'errezeta_info.html' fitxategian aurkitzen da. *HTML* fitxategi hau errezeta baten informazioa erakusteaz arduratzen da, bai eta errezeta horren gainean egin daitezkeen aukerak pantailaratzeaz ere, erabiltzailearen baimenen arabera; orrialde hau kudeatzeko, beraz, hainbat alderdi izan dira kontutan.

Hasteko, erabiltzailearen rol-a begiratu beharra dago, zein aukeren botoiak pantailaratu behar diren jakiteko. Gainera, ondo edo gaizki joan baldin bada errezetaren gaineko operazioren bat, atzera joateko estekak ere desberdinak dira. Horretaz gain, begiratu behar da baita ere sesioa irekita duen erabiltzailea errezetaren sortzailea den, horrela izanez gero, errezeta hori aldatzeko aukera baitu. Administratzailea baldin bada, edozein errezeta alda dezake, baina desberdintasuna da, administratzaileak errezetak onartu ditzakeela, beraz hori ere kodean kontutan hartu da. Ikusten denez, hainbat baldintzazko sententzia daude adierazita, aukera posible guztiak kontutan hartzeko, segurtasuna eta aukeren gaineko babesa sendoa izan dadin.

Errezeta onartzearen atalari dagokionez, bai 'Farmazia' motako erabiltzaileak bai Administratzaileak egin dezakete, eta horren alderdi bisuala kudeatzeko formulario bat egitea pentsatu da, ondorengo ezaugarriekin: formularioaren eremu bakoitzean, datu-basean aurretik gordeta egon den informazioa ezartzen da administratzailearen edo farmazeutikoaren kasuan, errezeta aldatzeko eta onartzeko baimena baitute; horretarako,errezetaren informazioa kargatzeaz arduratu den bistan lortu da informazioa eta parametro bezala bidali da, *HTML* orrialdean eskuragarri egon dadin. Horretaz gain, dispentsatu daitezkeen medikamentuen zerrenda egon beharra dago; medikamenturik egon ezean mezu bat adieraziko da medikamenturik ez dagoela esanez, eta kontrako kasuan, combobox batzuekin markatu daitezkeen medikamentuen zerrenda bat agertuko da, bai eta bakoitzaren unitateak aukeratzeko beharrezkoa den zenbakizko eremu bat, esleitu daitekeen balio minimoa 1 izanik, eta maximoa medikamentuaren unitate kopurua.

```
@login_required
def errezeta_onartu(request, errezeta_ident):

    #Aldagai laguntzaileen definizioa
    #aldagaien deklarazioa...

    #CheckBox-ean aukeratuta dauden elementuak kargatuko ditu
    aukeratutako_medikamentuak = []

    #Errezeta onartu den edo ez jakiteko aldagaia
    onartuta = False

    #Entseguaren protokolo zenbakia lortzen da
    errezeta_protokolo_zenb=
    EnsaioErrezeta.objects.get(ident=errezeta_ident)

    #errezetako pazienteen zerrenda kudeatzeko erabiliko da
    ensaioa_protokolo_zenb=
    errezeta_protokolo_zenb.ensaioa.protokoloZenbakia

    if request.method == 'POST':

        #Checkbox-ean markatutako medikamentuak jasotzen dira
        dispentsatu_beharreko_medikamentu_lista=
        request.POST.getlist('medikamentua')

        #Lista hutsa ez bada eta medikamentu bat aukeratu baldin bada
        if dispentsatu_beharreko_medikamentu_lista:

                        #Dispentsazioa eta Ensaioa lorzen dira lehenik
            #kodea...

            #Errezetaren egoera 'Pendiente'-tik
            #'Dispentsatuta'-ra pasatu behar da

            #Errezeta dispentsatuta bezala jartzen da
            ensaio_errezeta.pendiente = 'Dispentsatuta'
            ensaio_errezeta.save()
```

```
                #Dispentsatu diren medikamentuak lortzen dira
                dispentsatu_beharreko_medikamentu_lista=
            request.POST.getlist('medikamentua')

            #Orain, medikamentuak dispentsatzera jo behar da
            for dispentsatzeko_medikamentua in
            dispentsatu_beharreko_medikamentu_lista:

                #Orain aukeratutako medikamentu bakoitzaren
                #nitateak hartzen dira
                medikamentuaren_unitateak=
                int(request.POST[dispentsatzeko_medikamentua])

                #Meidkamentua eta Pazientea lortu
                #Eta erabiltzaileak aukeratutako dosia esleitu
                #kodea...

                #Eguneraketak gordetzen dira
                paziente_dispentsazio.save()

                #stock-ean dagoen medikamentuari
                #uitate horiek kentzen zaizkio

                medikamentua.unitateak=
                medikamentua.unitateak - medikamentuaren_unitateak

                medikamentua.save()

                #Bakarrik onartuko da
                #dispentsazioa medikamenturen bat dispentsatu bazaio

                onartuta = True

        if not onartuta:
            mezua = "Medikamentu bat dispentsatu behar zaio gutxienez"
            gaizkiJoanDa = True

            #Interfazeko formularioan agertu behar
            #den eremuen informazioa eskuratzen da
            #Eta emaitza bidaltzeko prestatzen da
```

Hau lantzen den bistan, hori kudeatzeko, lehenik eta behin checkbox-aren bidez aukeratu diren medikamentuak (id etiketaren balioa 'medikamentua' izan dutelarik, guztiek berdina) lista batetan gordetzen dira, guztiek id balio berdina izan dutela aprobetxatuz. Markatutako medikamentuen zerrenda hutsik ez baldin badago, errezetaren bukaerako dataren atributurako, 'bukaeraData' atributuaren balioa jaso eta dagokion formatura bihurtzen da, errezetaren informazioan. Jarraian, dispentsazioa sortzen da eta gorde egiten da datu-basean. Errezetaren egoera 'Pendiente'-tik 'Dispentsatuta'-ra pasatzen da, errezeta pendienteen listan ez agertzeko gehiago.

Hori egin eta gero, aukeratutako pazientea entseguarekin lotuta dagoen edo ez begiratzen da, lotuta ez baldin badago, entseguaren eta pazientearen arteko erlazioa egin beharra baitako (hau

badaezpada hartu den neurri bat da, erlazioak ondo finkatuta egoteko).

Jarraian, medikamentuen dispentsazioa kudeatzera jotzen da. Horretarako, markatutako medikamentuen zerrendako elementu bakoitzeko, aukeratu diren unitateak jasotzen dira (int-era aldatuz balioa), *HTML*-an medikamentu bakoitzaren unitateen eremuaren name etiketaren balioa medikamentuaren identifikatzailea jarri delarik, unitateen eta medikamentuen esleipena argiagoa izateko; eta Medikamentua beraren objektua ere bai, dispentsazioaren informazioan medikamentuaren instantziaren eta dispentsatutako unitateen informazioa lotzeko.

Azkenik, medikamentu horrentzat, Stock-ean dauden unitateak eguneratzen dira, lehen zeudenak dispentsatu diren kopuruarekin kenduz. Errezeta ondo onartu denaren aldagaia eguneratzen da.

Jarraian begiratzen da ea aldagai hori 'True' balioarekin dagoen edo ez; hots, medikamentuen zerrenda hutsa izan ez baldin bada eta behintzat medikamentu bat dispentsatu baldin bada edo ez. Horrela ez baldin bada, errezeta ondo onartu dela adierazten duen mezua prestatzen da, eta gaizki joan dela adierazten duen aldagaiaren balioa ezartzen da. Berriz ere, errezetaren informazioaren pantailara berbideratuko denez, eremuetan agertu behar den informazioa eskuratzen da.

- Medikamentuak Gehitu Ensaioari:

```
{% if mota == farmazia or mota == admin %}
    Medikamentuen zerrenda:
    <br/><br/>

    {% for m in medikamentuen_lista %}
      <li>
        {{ m.medikamentua.ident}} <input class="btn btn-xs btn-danger"
        type=button name="kenduMedikamentuaEnsaiotik"
        onClick="location.href='/farmaciapp/aukera_menua/
        ensaio_kontsulta/ensaio_bilaketa/ensaioa/{{titulua}}/
        medikamentuak/kendu_medikamentua/
        {{m.medikamentua.identKodetua}}/'"
        value='Kendu Medikamentua Ensaiotik'>
      </li>
      <br/>
    {% endfor %}

    <br/><br/>
```

```
      <form id="ensaioa_sortu2" method="post"
      action="/farmaciapp/aukera_menua/ensaio_kontsulta/
␣␣␣␣␣ensaio_bilaketa/ensaioa/{{titulua}}/medikamentuak/"
      enctype="multipart/form-data">

        {% csrf_token %}

        {{ medikamentua_form.as_p }}

        <input class="btn␣btn-primary" type=button
        onclick="konfirmazioa('ensaioa_sortu2')"
        name="gehitu_medikamentu_berria_ensaiora"
        value="Gehitu␣Medikamentua"/>

      </form>

    {% endif %}

<br/><br/>

    {% if mezua %}
      <div class="alert␣alert-success" role="alert">
         <span class="glyphicon␣glyphicon-ok" aria-hidden="true"></span>
         <span class="sr-only">Informazioa: </span>
           {{mezua}}
       </div>
    {% endif %}
```

Entseguari medikamentu bat gehitzeko interfazearen kodea 'medikamentuak_gehitu_ensaioari.html' fitxategian dago. Kasu horretan, entseguak gehituta dituen medikamentuen zerrenda bat agertzen da, medikamentu hori ezabatzeko botoi batekin batera, eta jarraian, medikamentua gehitzeko formularioa agertzen da.

```
@login_required
def medikamentuak_gehitu_ensaioari_botoia
(request, ensaioa_protokolo_zenb):

    #aldagai laguntzaileak
    mezua = ''

    medikamentua_form = MedikamentuBerriFormularioa()

    #Behintzat medikamentuaren titulua espezifikatzen bada,
    #medikamentuaren gehikuntza tratatuko da
    if(request.method == 'POST'):

     if(request.POST['ident']!=''):

       medikamentua_form=MedikamentuBerriFormularioa(data=request.POST)
       #Betetako formularioa zuzena bada
       if medikamentua_form.is_valid():
         if request.POST['ident'] != '':
           try:
            #Medikamentua existitzen bada,
            #informazioa eguneratuko zaio
            medikamentua_konprobatu=Medikamentua.objects.get
            (ident=request.POST['ident'])

            #Unitate kopuruari gehituko zaizkio unitate berriak
            zenbat_unitate=int(medikamentua_konprobatu.unitateak)
            zenbat_unitate=zenbat_unitate+int(request.POST['unitateak'])
            medikamentua_konprobatu.unitateak=zenbat_unitate

                  #medikamentuak izan dituen
          #unitateen historikoa eguneratzen da
            unitate_totalak=
            int(medikamentua_konprobatu.unitateak_historikoa)
            + int(request.POST['unitateak'])
```

```
            medikamentua_konprobatu.unitateak_historikoa=unitate_totalak
            #Gainontzeko informazioa eguneratzen da

            #Ordua esleitu badu erabiltzaileak, formatua eman behar zaio
            #Ordua espezifikatu badu
            ordua = request.POST['ordua']
            minutuak = request.POST['minutuak']
            ordu_berria_string = ordua + ':' + minutuak
            bidalketa_ordu_berria=parser.parse(ordu_berria_string).date()

            bidalketa_ordu_berria=bidalketa_ordu_berria.strftime('%H:%M')
            medikamentua_konprobatu.bidalketaOrdua=bidalketa_ordu_berria
            medikamentua_konprobatu.save()

        except:
        #Medikamentua ez bada existitzen,
        #medikamentu berria erregistratuko da

        medikamentua = medikamentua_form.save(commit=False)

        #identKodetua zenbaki bat izango da,
        #random bidez lortua, baina bakarra izango dena
        idkod = randint(0,sys.maxint)
        #idkod egoki bat lortzeko kodea...

        #behin aurkitu dela balio duen zenbaki bat,
        #medikamentuari esleituko zaio zenbaki hori
        medikamentua.identKodetua = idkod

        #medikamentuak izan dituen unitateen
        #historikoa eguneratzen da
        #...kodea...
        #Aurrean bezala, ordua sartu duen begiratu eta formatua eman

    #Medikamentua ondo gehitu dela adierazten da
    sortuta = True
```

```
    else:
    #Medikamentua aurretik gehituta dago sisteman
    #Begiratu behar da ea uneko ensaioak daukan edo ez
    medikamentua_konprobatu=
    Medikamentua.objects.get(ident=request.POST['ident'])

    #Entsegu horrek medikamentu hau daukan konprobatzen da
    try:
    ensaioa_titulua = Ensaioa.objects.get
    (protokoloZenbakia=ensaioa_protokolo_zenb).titulua

    ensaioak_badu_medikamentua= MedikamentuEnsaio.objects.get
    (ensaioa=ensaioa_titulua,medikamentua=request.POST['ident'])

     except:
        #Ez badauka medikamentu hau, gehitu egin behar zaio
        ensaioa= Ensaioa.objects.get
        (protokoloZenbakia=ensaioa_protokolo_zenb)

        ensaioak_badu_medikamentua= MedikamentuEnsaio
        (ensaioa=ensaioa, medikamentua=medikamentua_konprobatu)
        ensaioak_badu_medikamentua.save()
      #Informazioa eguneratzen da eta ordua sartzen bada
      #formatua ematen zaio

      #Azkenik emaitza prestatu eta dagokion orrira berbideratu
```

Hau kudeatzeko kodeari dagokionez, kontutan hartzen da medikamentuaren identifikatzailearen eremua bete baldin bada, existitzen den medikamentu bat gehitu baitaiteke berriz ere, unitateak gehituz erregistratuta zegoenari.

Lehenik eta behin, medikamentua lehendik existitzen den edo ez begiratzen da 'try' sententziaren barruan. Existitzen ez baldin bada 'catch' sententziaren barruan dagoen kodea prozesatuko da. Existitzen baldin bada, unitate kopurua jasotzen da int-erako konbertsioa eginez, eta unitateen historikoa eguneratzen da baita ere (historikoa eta uneko medikamentuak ez dira berdinak; historikoak betidanik heldu diren unitateen kopuru totala gordetzen du, eta uneko unitateak, dispentsatzeko prest dauden unitate kopurua). Erabiltzaileak bidalketa-ordua espezifikatu badu ere, informazioa gehitzen zaio medikamentuari.

Medikamentua existitzen ez baldin bada, ordea, ausazko zenbaki bat esleitzen zaio medikamentu horri, bakarra izango dena. Horrela egitearen erabakia modeloa ez aldatzearena izan zen, proiektua aurreratuta zegoela Farmaziako departamentutik komentatu zen atal bat zelarik. Hau da, modeloan unique balioa duen aldagaia jartzeko aukera zegoen arren, medikamentua existitzen bazen eta unitateak gehitzea bazen burutu nahi zen aukera, errorea emango luke, zenbaki bat espezifikatu beharra egongo zelako. Baina ausazko zenbaki bat eskuz lortuz, *Django*-ren formularioek jaurti zezaketen erroreak ekiditea lortzen da, nahiz eta erabakirik txukunena ez izan. Arrazoi nagusia, beraz, eginda zegoen kodea eta modeloa gutxien aldatzearena izan zen.

Formularioa zuzena izan ez baldin bada (kasu honetan, medikamentuaren identifikatzailearen eremuan espezifikatzen den balioa existitzen bada, modeloaren gako nagusia denez gero, errorea jaurtiko du esanez ezin dela medikamentu berdina erregistratu, baina kasu horretan, unitateak handitzea kudeatu beharko litzateke), medikamentua existitzen dela esan nahi du, beraz, medikamentua datu-basetik eskuratu eta unitate kopurua handitzera jotzen da, aurretik azaldu bezala.

- PDF:

```
@login_required
def pdf_eskuratu(request, ensaioa_protokolo_zenb, dispentsazioa_ident):
    #PDF-a eskuratzeko kodea

    #Beharrezko informazioa lortzen da lehenik

    #Dispentsazioaren/Errezeta informazio osoa eskuratzen da
    paz_dis = PazienteDispentsazio.objects.filter
    (dispentsazioa__ident=dispentsazioa_ident)

    #Pazientea lortzen da
    paziente_id = paz_dis[0].paziente

    #Dispentsatu diren medikamentuak eskuratuko ditugu
    medikamentuak = #kodea...

    #Errezetaren gainontzeko informazioa lortzen da
    errezeta = #errezeta lortzeko kodea
    gainontzekoEremuak = errezeta.gainontzekoEremuak

    #Entseguaren informazioa aterako da
    ensaioa = #informazioa lortzeko kodea...

    #PDF-a sortzeko gainontzeko informazioa eskuratzen da...

    #HttpResponse objektua sortzen da.
    response = HttpResponse(content_type='application/pdf')
    response['Content-Disposition'] = 'attachment;
    filename="'+dispentsazioa_ident+'.pdf"'

    #Canvas objektua lortzen da eta beharrezko lerroak idazten dira
    c = canvas.Canvas(response)

    c.setFont('Helvetica-Bold', 18) #Formatua, letra mota eta tamaina
    c.drawString(50,770,'Dispensacion: ' + errezeta.errezetaIzena)

    #Errezetaren gaineko informazioa jartzen da dagokion formatuan
    #kodea...
```

```
    #Medikamentuak zerrendatuko dira
    altuera = 580
    for med in medikamentuak:
        #Dagokion medikamentuaren dosia kalkulatuko da:
        dosia = PazienteDispentsazio.objects.get
        (dispentsazioa=dispentsazioa_ident,
        medikamentua=med.get('medikamentua')).dosia

        c.drawString(120, altuera, '- ' +
        med.get('medikamentua') + '        Unidades: ' + str(dosia))
        altuera = altuera - 30


#Falta den informazioa ezartzen da
#kodea...

#Azkenik koadro bat egiten da medikamentuen zerrendaren
#onarpena jartzeko
c.line(30,altuera,580,altuera)
 c.line(30,altuerakuadro1,30,altuera)
 c.line(580,altuerakuadro1,580,altuera)

 c.showPage()
 c.save() #PDF-a gordetzeko

 return response
```

Azkenik, PDF-ak bistaratzeko erabili den kodea azaltzea falta da. Horretarako, ez dago interfaze bisualaren erabilerarik. Dena egin da bistaren bitartez eta reportlab pakete gehigarriaren bitartez. Pakete horrek PDF-ak sortzea eta bistaratzea ahalbidetzen du, *Python* kodearen bitartez.

Inplementazioaren eta, azken finean, erabilitako eta garatutako software-aren atalarekin bukatzeko, aplikazioa exekutatzeko garrantzitsuak izan diren azkeneko atalak komentatzea falta da.

Proiektuaren '/farmaciapp/' direktorioaren barruan, 'migrations' izeneko beste iderktorio bat dago. Bertan, datu-basearekin zerikusia duten entitate-erlazional desberdinen gainean aldaketak egiten diren bakoitzean, aplikazioa 'migratu' beharra zegoen, hain justu datu-basean aldaketa horiek finkatu daitezen. Migrazio horien guztien 'log' fitxategiak gordetzen dira direktorio horretan, aztertzeko zein aldaketa egin diren datu-basean. Horrek dakartzan alde ona da edozein momentutan aztertu daitezkeela egindako migrazioak, bai eta aldaketa horietakoren bat gaizki eginez gero, migrazio horiek ezabatzeko aukera ere badago, karpetako dagokion *Python* fitxategia ezabatuz.

9 Segurtasuna eta Legedia

Datu klinikoak direnez gero, datuen babesa kontutan izan beharreko alor bat da, eta proiektuan zehar oso presenta izan da alderdi hori.

Hasteko, esan beharra dago, LOPD (*Ley Orgánica de Protección de Datos*)-k dioena betetzeko arazorik ez dela egon, ez baitira Pazienteen datu-pertsonalak erakusten edo konpartitzen aplikazioaren

erabilieran. Eta ez bakarrik hori; erabiltzaileek, edozein rol eta baimen dituztela ere, ezingo dute erregistratutako pazienteen informazioa ikusi edo aztertu, eta soilik errezeta bat PDF formatuan inprimatzeko eskatzerakoan agertuko dira pazientearen izena eta abizena, errezetetan agertu beharreko informazioa delako.

Horretaz gain, aplikazioan erregistratzen diren gainontzeko elementuen informazioa ez da konfidentziala, baina medikamentuen gaineko informazio sakona soilik Farmaziako departamentuko langileek edota adminitrsatzaileak aztertu ahalko dute.

Segurtasunari dagokionez, erabiltzaile-izen eta pasahitzak hash-kode bihurturik gordetzen dira datubasean, sesio-irekiera maltzurren baten saiakerak ekiditeko, eta erabiltzen den hash-kodeketa metodoa *bcrypt* da, *Rainbow Table* deritzon erasoen kontra oso eraginkorra dena; *blowfish* enkriptazio-metodoa erabiltzen du gainera, eta adituek diotenez, orain arte ez da aurkitu teknika hau desenkriptatzeko metodo eraginkorrik. Horrenbestez, kodeketa segurua ziurtatzen da sesio-irekierako datuentzako.

Aipatu bezala, aplikazioaren erabilerarako erabiltzaile desberdinen arabera baimen bereziak daude, funtzionalitate desberdinak erabiltzea ekidin edo ahalbidetzen dituztenak. Hau kontu handik tratatu den atal bat da, orrialde guztietan aztertzen baita zein den sesioa irekita duen erabiltzailea, eta zein rol daukan atzituta, aplikazioaren atal batzuetara sartzeko baimena eskaintzeko edo ez.

10 Egindako Frogak eta Test-ak

Aplikazioaren inplementazioa bukatu ostean, bere funtzionamendu zuzena egiaztatzeko hainbat test egitea pentsatu zen, bai eta aplikazioaren gainean ezarri ziren beste helburuak bete zirela egiaztatzeko beste motatako froga batzuk egitea ere.

Aplikazioa hiru modu desberdinetan frogatzea eta testeatzea pentsatu nuen, hala nola: datu-baseko entitateen instantzien sorrera zuzenerako, aplikazioan gehien erabiliko diren formulario garrantzitsuenen funtzionamendu zuzenerako, eta funtzionalitate nagusien (view edo bista nagusien) funtzionamendu zuzenerako hainbat test unitario egituratu; aplikazioaren erabilera eta aspektu bisualen gaineko hainbat galdetegi pasatu nituen aplikazioa inoren laguntzarik gabe probatu zuten hainbat erabiltzaileri; eta, azkenik, Alazne Bustinza, Gurutzetako ospitaleko Farmaziako departamentuko arduradunari, benetako entsegu eta errezeta klinikoekin lan egitea ahalbidetzen dion sareko zerbitzari batetan jarri nuen aplikazioa, benetako erabileran ondo ibiliko litzatekeen aztertzeko, eta hobekuntzarik proposatu daitekeen aztertzeko.

10.1 Test unitarioak

Django-k test unitarioak egiteko aukera bat eskaintzen du, 'tests.py' fitxategi baten bitartez. Fitxategi honetan, kodearen gaineko froga unitarioak egiten dira, izan beharko lukeen bezalako funtzionamendua duela ziurtatzeko helburuarekin. Bi bloke nagusitan banatu dut test-fitxategia. Lehenengo blokean, models.py fitxategian (datu-basean gordeko diren entitateak eta haien arteko erlazioak definituta dauden fitxategian) dauden entitateen instantziak ondo sortzen direla konprobatu dut, kasu kritikoak kontutan hartuz.

Eta bigarren blokean, aplikazioaren kodean instantziak sortzeko erabiltzen diren formularioak ondo dabiltzala egiaztatu dut, eta bete beharrezko eremuak bete behar direla konprobatzera jo dut.

Horrela, aplikazioaren funtzionamenduaren elementu nagusienak frogatuta gelditu dira.

```
#Aplikazioaren funtzionalitateak ondo dabiltzala frogatzeko Test-ak

#Modeloko entitateen sorkuntza-testak:

class PazienteaTest(TestCase):

        def test_paziente_sorkuntza(self):

      pazientea=Pazientea(idensaioan=1,izena='pazientea_proba1',
                unitateKlinikoa='Oncologia', pisua=80)
            pazientea.save()
            self.assertEqual((pazientea.ident >= 0), True)

            pazientea2=Pazientea(idensaioan=1, izena='pazientea_proba1',
                unitateKlinikoa='Oncologia', pisua=80)
            pazientea2.save()
            pazienteKopurua = Pazientea.objects.all().count()
            self.assertTrue(pazienteKopurua == 2)



class EnsaioaTest(TestCase):

        def test_ensaio_sorkuntza(self):
                data = time.strftime('%Y-%m-%d')
                ensaioa = Ensaioa(egoera='irekita', hasieraData=data,
                protokoloZenbakia='prot1', titulua='ensaioa1',
                zerbitzua='z', promotorea='p', estudioMota='em',
                monitorea='m', monitoreaEmail='em',
                monitoreaFax=123456789, monitoreaTel=123456789,
                monitoreaMugikor=123456789, ikertzailea='i',
                komentarioak='')

                ensaioa.save()
                ensaioKopurua = Ensaioa.objects.all().count()
                self.assertTrue(ensaioKopurua == 1)

                ensaioa2 = Ensaioa(egoera='irekita', hasieraData=data,
                protokoloZenbakia='prot2', titulua='ensaioa2',
                zerbitzua='z', promotorea='p', estudioMota='em',
                monitorea='m', monitoreaEmail='em',
                monitoreaFax=123456789, monitoreaTel=123456789,
                monitoreaMugikor=123456789, ikertzailea='i',
                komentarioak='')

                ensaioa2.save()
                ensaioKopurua = Ensaioa.objects.all().count()
                self.assertTrue(ensaioKopurua == 2)
```

```
                ensaioa = Ensaioa(egoera='irekita', hasieraData=data,
                protokoloZenbakia='prot1', titulua='ensaioa1',
                zerbitzua='z', promotorea='p', estudioMota='em',
                monitorea='m', monitoreaEmail='em',
                monitoreaFax=123456789, monitoreaTel=123456789,
                monitoreaMugikor=123456789, ikertzailea='i',
                komentarioak='')

                ensaioa.save()
                ensaioKopurua = Ensaioa.objects.all().count()
                self.assertTrue(ensaioKopurua == 2)

#Gainontzeko modeloen test-ak...
```

```
#Aplikazioan zehar erabiliko diren sorkuntza-formulario test-ak:

class PazienteSorkuntzaTest(TestCase):

        def test_paziente_sorkuntza_form(self):
                form_data = {'izena':'izena', 'pisua':80, 'idensaioan':1}
                form = PazienteBerriFormularioa(data=form_data)
                self.assertTrue(form.is_valid())

                form_data = {'pisua':80}
                form = PazienteBerriFormularioa(data=form_data)
                self.assertFalse(form.is_valid())

                form_data = {'izena':'izena'}
                form = PazienteBerriFormularioa(data=form_data)
                self.assertFalse(form.is_valid())
```

```
class EnsaioSorkuntzaTest(TestCase):

        def test_ensaio_sorkuntza_form(self):
                data = time.strftime('%Y-%m-%d')
                form_data = {'hasieraData':data,
        'protokoloZenbakia':'protzenb1', 'titulua':'ensaioa1',
        'zerbitzua':'z', 'promotorea':'p', 'estudioMota':'em',
        'monitorea':'mon', 'ikertzailea':'ik'}

                form = EnsaioBerriFormularioa(data=form_data)
                self.assertTrue(form.is_valid())

                form_data = {'hasieraData':data, 'protokoloZenbakia':'1',
        'titulua':'ensaioa1',  'promotorea':'p', 'estudioMota':'em',
        'monitorea':'mon', 'ikertzailea':'ik'}

                form = EnsaioBerriFormularioa(data=form_data)
                self.assertFalse(form.is_valid())

                form_data = {'hasieraData':data, 'protokoloZenbakia':'1',
        'titulua':'ensaioa1', 'zerbitzua':'z', 'estudioMota':'em',
        'monitorea':'mon', 'ikertzailea':'ik'}

                form = EnsaioBerriFormularioa(data=form_data)
                self.assertFalse(form.is_valid())

                form_data = {'hasieraData':data, 'protokoloZenbakia':'1',
        'titulua':'ensaioa1', 'zerbitzua':'z', 'promotorea':'p',
        'monitorea':'mon', 'ikertzailea':'ik'}

        form = EnsaioBerriFormularioa(data=form_data)
                self.assertFalse(form.is_valid())

                form_data = {'hasieraData':data, 'protokoloZenbakia':'1',
        'titulua':'ensaioa1', 'zerbitzua':'z', 'promotorea':'p',
        'estudioMota':'em',  'ikertzailea':'ik'}

                form = EnsaioBerriFormularioa(data=form_data)
                self.assertFalse(form.is_valid())

                form_data = {'hasieraData':data, 'protokoloZenbakia':'1',
        'titulua':'ensaioa1', 'zerbitzua':'z', 'promotorea':'p',
        'estudioMota':'em', 'monitorea':'mon'}

                form = EnsaioBerriFormularioa(data=form_data)
                self.assertFalse(form.is_valid())

        #...
```

10.2 Galdetegiak

Galdetegi bat egitea ere proposatu nuen, aplikazioaren lehenengo bertsio funtzionala bukatu nuenean. Honen erabakia argia zen: hasiera batetan jasotako betekizunen gainean adostu ziren helburuak, bai erabilpenaren eta bai aspektu bisual intuitiboen gainekoak, betetzen ziren edo ez jakiteko. Horretarako, aplikazioa adin desberdinetako eta sexu desberdinetako hainbat pertsonari eskaini nien, gero, galdera itxi zein irekiko galdetegi bat betetzea proposatuz. Galdetegia bete zuten erabiltzaile horien artean Alazne Bustinza zegoen, azken finean, bere iritzia baita gehien inportatzen duena.

Erabilitako galdetegiaren txantiloia ondorengoa da:

Ebaluazioaren Azalpena.

Ondorengo dokumentuaren bitartez azkeneko prototipo digitalaren gaineko ikuspegia zein den jakitea lortu nahi da. Erabiltasunaren eta interfazearen itxuraren gaineko galdera itxi sorta bat egingo da, erantzun ponderatu batzuk jasotzeko, eta galdera ireki gutxi batzuk ere egingo dira erabiltzaile potentzialaren iritzi zabalago eta zehatzago bat eskuratzeko. Bukatzeko, aplikazioaren puntu sendo eta ahulak aipatzea eskatuko da.

Edozein hobekuntza proposatzeko edo aipamen berezi egiteko aukera ere egongo da.

Galdera itxiak.

Hainbat irizpide desberdin ebaluatzea proposatzen da, eta horretarako ondorengo puntuazio-mailen artean dagoen nota bat jarri beharko da:

1: Oso gaizki; 2: Gaizki; 3: Nahiko ondo; 4: Oso ondo; 5: Bikain.

Erabilpena ebaluatzeko irizpideak	Nota
Aplikazioko pantailen arteko nabigazioa intuitiboa da eta erraz ulertzen da.	
Aplikazioko botoi, menu eta beste elementuetan erabilitako testua egokia eta lagungarria da.	
Erabiltzaileak badaki edozein momentutan aplikazioko zein nabigazio-ataletan dagoen.	
Menu desberdinetatik nola irten eta sartu jakitea erraza da.	
Aplikazioko eragiketak egiteko prozesua erraz ulertzen da.	
Ezkerreko menu bertikala lagungarria da nabigazioan zehar.	
Fitxategiak PDF-an ateratzeko aukera egokia eta gustukoa da.	
Konfirmazio-mezuak ("Ziur al zaude...? Bai/Ez") nabigazioko puntu egokietan agertzen dira.	
Interfaze bisuala ebaluatzeko irizpideak	Nota
Aplikazioaren pantailetan ez dago gehiegizko elementurik.	
Erabilitako koloreak eta formak egokiak dira eta ez dute begietara molestatzen.	
Erabilitako letra mota eta tamaina egokiak dira	
Argi ikusten da zein aukera diren interakzioa eskaintzen dutenak eta zeintzuk ez.	
Ensaio, medikamentu edo errezeten informazioa adierazteko modua egokia da.	

Galdera Irekiak.

Jarraian hainbat galdera ireki proposatuko dira erabiltzaileak nahi duen zabaltasunez erantzun ditzan.

- Erabilera eta Itxuraren Gaineko Galderak.
  - Nabigazioa egokia dela uste duzu? Zer nolako zailtasunak/erraztasunak izan dituzu aplikazioa erabiltzerako orduan?

  - Aplikazioan erabilitako botoi, testu, konfirmazio-mezu eta gainontzeko elementuak egokiak direla uste duzu? Aspergarria edo erakargarria iruditzen zaizu?

- Aplikazioaren Hobekuntzarako Proposamenak.
  - Alde Baikorrak. Aplikazioari dagokionez aipatu zeintzuk diren gehien gustatu zaizkizun atalak.

  - Alde Ezkorrak. Aplikazioari dagokionez aipatu zeintzuk diren gutxien gustatu zaizkizun atalak.

Jaso diren emaitzen artean, ondorengo aipamenak izan dira interesgarrienak edo atentzioa gehien deitu dutenak:

- "Hizkuntza aldatzeko aukera egon beharko litzateke, erabiltzaile askok gaztelera bakarrik baitakite, baina beste hainbatek euskara nahiago baitute. Ingelesa sartzea interesgarria izango litzateke, baina ez da beharrezkotzat jotzen."

- "Oso egokia da zenbat errezeta dauden 'pendiente' egoeran ikusi ahal izatea ezkerreko menuan momentu guztietan, horrela, errezetaren bat onartu behar bada, jarraian jakiten delako onartzeko listan gehitu dela."
- "Ekintza konkretu batzuek egiterakoan, mezu bat agertzea komeniko zen adierazteko ekintza hori egin nahi den edo ez, batez ere atzera bueltatzeko aukerarik ez duten horietan."
- "Oso erabilgarria da administratzaileak erabiltzaileak kudeatu ahal izatea. Modu eroso batetan sortu eta moldatzen dira erabiltzaileak, eta segurtasuna handitzen da bakarrik administratzaileak egin dezakeelako."

Ikusten den bezala erabiltzaile desberdinek betetako galdetegien emaitza hauetan, aipatu diren hobekuntza-proposamenak kontutan izan ditut, eta, adibidez, konfirmazio-mezuak gehitzeko aukera inplementatzea lortu dut, baina hizkuntzarena, momentuz, epe luzeago batetarako helburu bezala jo dut.

Bestalde, agerian dago egindako lana gustukoa izan dela, aplikazioak izan behar zituen funtzionalitateak komentatu zirenean aipatu ez ziren balio-erantsiko aukerak gehitu baitira; adibidez, zenbat errezeta dauden pendiente adierazten duen etengabeko ikonoa, eta erabiltzaileak kudeatzeko funtzionalitatea.

Jarraian, galdetegiak bete eta gero lortu diren bataz-besteko puntuak adierazten dira, 'erabilpena' eta 'interfaze bisuala' alorrentzako:

Erabilpenaren atala: 4.75/5

Interfaze bisualaren atala: 4.34/5

Emaitza oso onak direla ikusi da, batez ere erabilpenaren atalari dagokionez. Izan ere, erabilera erraz eta intuitibo bat izatea lortu nahi izan da, eta horregatik eman zaio garrantzi gehiago atal horri interfazearen diseinuari baino, nahiz eta interfazea zaintzen ere saiatu naizen, intuitiboa izateko garrantzi handia daukalako, baita ere.

Galdetegi guztien emaitzak dauden esteka: http://ttiki.com/341177

10.3 Benetako kasuen erabileraren frogapena

Hau aurrera eramateko, behin galdetegia pasatu zela eta proposatutako hobekuntzak edo arazoak konpondu zirela lehenengo bertsio funtzionalean, sareko zerbitzari batetan jartzea erabaki nuen aplikazioa, Alazne Bustinzak eguneroko lanean dauzkan eginkizunetan erabiltzeko eta frogak egiteko. Abuztuan zehar frogatu da hau, lan-karga gutxien izan duen hilabetea izan delako, eta astebeteko frogak egin eta gero, azkenengo feedback edo iritzia eman zidan aplikazioaren gaineko azken ukituak egiteko.

Froga hau oso garrantzitsua eta baliozkoa izan da, eta agerian geratu da aplikazioaren garapenean zehar egindako bilerak eta batzarrak ere bere pisua eta balioa izan dutela, oso gauza gutxi aldatzeko proposamenak egon direlako; adibidez: entsegu baten titulua luzea izan daitekeenez, idatzi daitezkeen karaktere kopurua handitzea; edo entsegu baten 'lotea' esparruak edozein karaktere izan ditzakeela, eta ez zenbakiak bakarrik, hasieran planteatu zen bezala.

Laburbiltzeko, egindako froga desberdinek amaierako helburua lortzen lagundu dute. Alde batetik, aplikazioaren funtzionamenduan kode-errorerik ez egotea ziurtatu da test unitarioen bitartez, kasurik kritikoenak modeloko entitateen instantzien sorrera eta ezabaketa delarik, bai eta formularioak betetzeko eta bidaltzeko irizpideak zuzenak direla; bizpahiru funtzio nagusi (bilaketak eta errezeten onarpenak, adibidez) ere arazorik eta errorerik ematen ez dutela konprobatu da.

Atal hori ondo zegoela konprobatu zenean, hainbat erabiltzaileri aplikazioa probatzeko aukera emateak eta haien proposamen subjektiboak jasotzeak, aplikazioaren aparientzia (koloreak, botoiak, ikonoak eta irudiak, eta abar) eta erabilterreaztasunari (nabigazioa, batez ere) dagokionez zeintzuk gauza aldatu edo hobetu zintezkeen ezagutzea ekarri zuen. Eta ez hori bakarrik, zein elementu gehitzea izango litzatekeen interesgarria eta zein kentzea ere; adibidez: erabiltzaileak kudeatzeko menuan, hasiera batetan, ez zegoen bilaketak egiteko aukerarik, eta horren falta sumatu zuten erabiltzaileak; edo bilaketa batetan emaitzarik aurkitzen ez bada mezu baten bidez jakinaraztea.

Bukatzeko, proposamen guztiak kontutan hartu eta gero egindako aldaketekin, berriz ere Alazne Bustinzari lan-giro erreal batetan probatzeko aukera eskaintzeak aste beteko epe labur-ertain batetan erakutsi zuen (bigarren mailako helburuak kontutan hartu gabe) haiek bilatzen zuten aplikazioa egitea lortu zela, eta ospitalean software hau ezartzeko lehenengo pausu handia emanda zegoela.

11 Planifikatutakoa vs Errealitatea

Hasiera batetan planifikatutakoaren eta azkenean lortutakoaren arteko aldeak ez dira nabariak izan inplementazioaren aldetik, baina bai proiektua planifikatu zen momentuarekin alderatuz.

Ospitalean ezartzearen helburua eta horren gaineko itxaropenak bertan behera geratzen joan dira proiektuaren garapenean zehar, ospitaleko Informatikako departamentutik jarri diguten oztopoak direla medio. Gainera, kode-barra irakurgailu bat erabiltzearen helburua ere ezin izan da bete, denbora epeak ez digutelako teknologia hori aztertzen utzi, proiektuan sartzeko aukera izateko beste; eta nahiz eta hori horrela izatea espero zen, denbora aldetik ez nuen espero hainbeste luzatzea garapena.

Izan ere, nahiz eta kontsultak egiterako orduan, aplikazioan gordeko zen informazio-elementuen arteko loturak zailak izango zirela buruan neukan, azkenean, batzar batzuetan gertatu izan den bezala, egindakoaren gainean argibideak egin dira, eta hori zuzentzen edo moldatzen joanak, lotura horiek konpondu edo sakontzea eragin du, zaila izatea espero zen lana, pixkat konplexuagoa bihurtuz.

Ez nuen espero, beraz, aplikazioa lantzeko beharreko kontzeptuak erlazionatzeko hain zailak izango zirenik; hau da, entsegu, medikamentu edo pazienten artean egon behar ziren erlazioak ezartzea hain zaila izango zenik.

Beraz, ondorioz, esan beharra daukat, hasieran egitea espero nuen aplikazioa (bigarren mailako helburuak alde batera utzita) egitea lortu dudala, baina eskaintzea espero nuen denbora baino gehiago eskainita, batzar askotan kontzeptu batzuen ulermena sendotzeak eta argitzeak, kodean aldaketak egitean eragin duelako, hasieratik kontzeptu horiek ezaugarri guztiekin ez sakontzearen ondorioz.

Jarraian, hasiera batetan egindako plangintzaren eta errealitatean egindakoaren arteko aldea adierazten duen taula ageri da:

Fasea	Plangintza estimatua	Plangintza erreala
Betekizun-bilketa	**2 ordu**	**3 ordu**
Analisia	**12 ordu**	**21 ordu**
Erabilpen Kasuen diagrama.	6 ordu	12 ordu
Domeinu-ereduaren diagrama.	6 ordu	9 ordu
Diseinua	**9 ordu**	**12 ordu**
Paperezko prototipoa	1 ordu	1 ordu
Prototipo digitala	8 ordu	11 ordu
Teknologien erabileraren ikaskuntza	**50 ordu**	**55 ordu**
Python	15 ordu	10 ordu
Django	35 ordu	45 ordu
Bilerak ospitalean	**40 ordu**	**42 ordu**
Inplementazioa	**360 ordu**	**429 ordu**
1.prototipo funtzionala	**290 ordu**	**337 ordu**
Modeloa eta datu-basea	15 ordu	21 ordu
Logika (funtzionalitateak)	225 ordu	250 ordu
Interfazea	50 ordu	66 ordu
2.prototipoa (kontzeptu-zuzenketak)	**50 ordu**	**60 ordu**
Modeloa eta datu-basea	10 ordu	12 ordu
Logika (funtzionalitateak)	30 ordu	42 ordu
Interfazea	10 ordu	6 ordu
3.prototipoa (Errore zuzenketak)	**15 ordu**	**20 ordu**
Modeloa eta datu-basea	3 ordu	5 ordu
Logika (funtzionalitateak)	11 ordu	12 ordu
Interfazea	1 ordu	3 ordu
Azkeneko prototipoa (azken gehikuntzak)	**5 ordu**	**12 ordu**
Modeloa eta datu-basea	1 ordu	1 ordu
Logika (funtzionalitateak)	2 ordu	6 ordu
Interfazea	2 ordu	5 ordu
Probak	**10 ordu**	**10 ordu**
Memoria	**50 ordu**	**51 ordu**

Hasiera batetan proiektua egiteko planifikatutako orduak: 533 ordu.

Errealitatean proiektua garatzeko behar izan diren orduak: 623 ordu.

Taulan ikus daitekeen bezala, hasiera batetan planifikatu zenaren eta bukaeran egin denaren lan-orduen arteko aldea 90 ordukoa izan da. Hor bi gauza ikus daitezke: benetan, espero zena baino ordu gehiago lan egin dela, batzarretan zehar komentatzen joan diren zuzenketak eta aurretik egindako lanaren gaineko hobekuntzak anitzak izan direlako, eta, aurretik aipatu bezala, Gurutzetako ospitaleko Informatikako sailetik jarri zizkiguten oztopoak nabaritu egin zirelako; eta beste alde batetik, diferentzia oso handia ez dela izan aplikazioaren garrantzizko esparruetan, nahiz eta orokorrean aldea handia izan, hasiera batetan egindako plangintzak benetan balio izan duela lanaren bideraketarako erakutsiz.

Planifikatutakoaren eta errealitatean lan egindako orduen alde nagusienak analisiko atalean, teknologien ikaskuntzan eta inplementazioko atalean eman dira. Analisiko atalean hasiera batetan pentsatutako ordu kopurua labur gelditu izanaren arrazoia da aplikazioaren prototipo desberdinak egiten joan ahala, bai Erabilpen Kasuen diagraman eta bai Domeinu-ereduaren diagraman aldaketak egin direla; hori gertatu da aplikazioan funtzionalitateen aldakuntzak egiten joan direlako, edo baita ere funtzionalitateak ezabatzen joan direlako (adibidez, hasiera batetan, erabiltzaileak bere kabuz erregistratu zitezkeen aplikazioan, baina azkenean adostu zen bakarrik administratzaileak erregistratzea nahi zituen horiek); eta baita ere, aldaketa horiek aurrera eramateko datu-basean eta modeloan aldaketak egin behar izan direlako.

Inplementazioan eskainiko ziren orduei dagokien desbiderapena espero zen, hasiera batetik. Izan ere, teknologiak ikasterako orduan, *Django*-k nola lan egiten zuen ondo menperatu beharra zegoenez, eta azkenean, espero baino alor gehiago kontutan hartu behar zirenez, ikasketarako estimatutako denbora baino gehiago dedikatzea gertatu zen. Web-aplikazioaren garapenean, berriz, bilera desberdinetara joaterakoan aurretik eginda zegoena zuzendu, moldatu eta hobetzeko teknika egitea aukeratu nuenez, espero izandakoa baino denbora gehiago eskaini zitzaion. Izan ere, akatsen zuzenketarako egin beharreko ikasketak eta ikerkuntzak ian ziren ordu horien gehikuntza suposatu zituztenak.

Azkenean, ikusten da nahiz eta planifikatutako eta benetan landutako ordu horien arteko diferentzia nabaria den, aplikazioa lantzeko beharrezkoak diren oinarri sendoen gaineko plangintza fidela izan dela, eta desbiderapen horiek bakarrik ikasketa-maila gehigarri bat suposatu zuten eginkizunetan gertatu zirela.

12 Etorkizunerako lana

Proiektu hau bukatu ostean aplikazioa garatzeko motibazioa piztu ninduen helburua edo ideia ez dela oraindik bete ikusi dut: Gurutzetako ospitalean aplikazioa ezartzea, Farmaziako departamentuko eguneroko lanean erabiltzeko. Hori dela eta, etorkizun baterako zein helburu potentzial dauden azaltzea gustatuko litzaidake, hasiera batetan pentsatu zena aurrera eraman ahal izateko.

Lehenik eta behin, esan behar da Gurutzetako ospitaleko Farmaziako departamentutik ados daudela garatutako softwarearekin, haien eguneroko lanerako diseinatutako aplikazio batek izan behar dituen betekizun guztiak eta gehiago dauzkalako. Baina egia da hainbat aukera landu barik geratu direla, lan-karga eta denbora arazoak izan direla medio.

Horregatik, etorkizun batetan, aplikazio hau barra-kode irakurgailu batekin bateratzea izango litzateke egokiena. Horren arrazoia da medikamentuen erregistroa eta medikamentu horiek paziente bati dispentsatzeko prozesua azkarra izatea garrantzitsua dela. Aplikazio honekin, datuak eskuz sartu behar dira medikamentu bat erregistratzerakoan, eta dispentsatzerakoan medikamentuak aukeratu behar dira. Aldiz, barra-kodeak ezartzerakoan informazioa nola esleitzen den eta nola irakurtzen den ikasiz gero, prozesua azkartzea lortu zitekeen. Hain justu, hori aztertzea izango litzateke etorkizuneko betebehar bat.

Egin beharreko beste gauza bat da ospitaleko Informatikako departamentuarekin bildu eta eginda dagoen proiektua erakutsi, landu diren teknologiak, barneratu diren ideiak eta erabilitako herramintak zeintzuk izan diren azaltzeko. Eta behin hori eginda, ospitalean ezartzeko beharrezkoak diren eginkizunak, gehikuntzak edo aldaketak zeintzuk diren adostu. Horrela, hasiera batetan pentsatu zen helburu nagusia betetzea lortu zitekeen.

Beraz, nahiz eta aplikazioa eginda dagoen, barra-kode irakurgailuen teknologia aztertzea eta bateratzea izango litzateke betekizun nagusi bat, bai eta ospitaleko Informatikako sailarekin komentatzea

zer nolako moldaketak egin behar diren aplikazioa ezarri ahal izateko.

13 Ondorioak

Proiektua egin ondoren hainbat dira ikasi ditudan gauzak eta atera ditzakedan ondorioak.

Lehenik eta behin, aurretik banekien ere, mota eta dimentsio honetako proiektu bat garatzen aritzean, aldez aurretik egindako planifikazio eta organizazioaren garrantzia nabarmendu nahiko nuke. Bezeroak aplikazioak izateko nahi dituen betekizunen bilketa egiteak eta maila desberdinetako helburuak finkatzeak egin beharreko aplikazioak izango zuen lan-karga eta lortu zintezkeen limiteak zeintuk ziren hobeto definitzen lagundu izan dute.

Horretaz gain, aldez aurretik egindako Erabilpen Kasuen Diagramak eta Domeinu-ereduaren Diagaramak, inolako software-aren aukeraketarik egin ez zela ere, aplikazioak izango zituen funtzionalitateen, erabiltzaile moten eta eredu erlazionalaren ideia sendo eta konkretua izaten lagundu dute.

Beste alde batetik, bai paperezko prototipoak zein prototipo digitalak izan duten garrantzia, diagramak egin eta gero, nabarmena izan da. Definitutako funtzionalitateak eta jasotako betekizunen listak zer nolako pantailak egongo ziren eta haien arteko nabigazioa nolakoa izango zer errazago eta argiago ikustera eraman dute, aplikazioaren lehen ideia orokor bat eduktzera eramanez.

Eta diagrametan aldaketak egon direla eta prototipo digitalak erakusten duen interfazearen eta aplikazioaren interfazearen artean aldea dagoela argi badago ere, esan beharra dago helburua ez zela hasierako planteamenduan definitu zen emaitza berbera lortzea, bukaerako emaitza helburuak betetzeko gai den aplikazio bat egiteko behar bezalako laguntza eskaintzea baino. Horrenbestez, inplementazioarekin hasi aurreko lan hori guztia egiteko dedikatutako ordu kopurua ez dela alperrikakoa izan esan beharra dago, asko lagundu baitit amaierako helburua lortzen.

Beste alde batetik, proiektu hau garatzen joan naizen heinean egindako batzar guztien ondorioz ikusi dut benetan bezeroa asetzeko premia ez dela hain erraza. Izan ere, astero Gurutzetako ospitalean Alazne Bustinzarekin egiten nituen batzarrak aplikazioaren gaineko hainbat kontzeptu argitzeko eta beste hainbat kontzeptu berriren jakinaren gainean jartzeko balio izan dute. Horregatik, proiektuaren inplementazioaren garapenaren planifikazioa egitea oso garrantzitsua izan da, aplikazioak eskaini behar zituen funtzionalitate desberdinak indibidualki osotzen joatea saiatu bainaiz, astero izaten nuen batzarrean komentatzeko bezeroaren erosotasuna lanarekiko.

Metodologia hori izan da, hain justu, lan ordenatu eta eroso bat egiten utzi nauena; eta nahiz eta azkeneko asteetan, aplikazioa bukatutzat emanda zegoela eta aplikazioaren erabileraren frogapenak egiten ari zirela, aldaketa edo zuzenketak proposatu diren, esan beharra dago ez direla nabarmenak izan, eta horiek konpontzea erraza izan dela (adibidez, erabiltzaileak bete beharreko eremu batek idazteko baimentzen zuen karaktereen luzeera moldatzea).

Lanaren garapenean egon diren arazo edo oztopoei dagokienez, nabarmenena eta helburuen gainean eraginik handiena izan duena, Gurutzetako ospitaleko Informatikako departamentuaren partetik izandako kontsiderazio falta izan da. Izan ere, proiektuaren inplementazioarekin hasi aurretik ospitaleko Informatikako sailarekin kontaktuan jartzeko ahaleginak ez zuten erantzunik izan, eta horrenestez eta proiektuaren luzapena ez handiagotzeko asmoarekin, departamentuko arduadunen emaitza jaso gabeko erabakiak hartu behar izan genituen, aplikazioaren garapenerako software-teknologien aukeraketari dagokienez.

Esan beharra dago bilera bat egon zela Rubén García Fernández, Gurutzetako ospitaleko Farmaziako departamentuko informatikari arduradunarekin, baina bakarrik gure proiektuaren ideiaren azalpena jasoteko eta gure helburuak jakiteko izan zen, eta ez zigun inolako argibide teknikorik eskaini proiektuaren garapenari zegokienez.

Horrenbestez, bere departamentuko zuzendariarekin kontaktuan jartzeko ahalegina egingo zuela adierazi zigun, guk eskatutako informazioa lortzeko helburuarekin. Baina gure proiektuaren gaineko ezjakintasuna dela medio, batzarra ez zen ekainera arte egin, proiektuaren inplementazioa, jada, aurreratuta zegoenean.

Gurutzetako ospitalean erabiltzen diren teknologien konfidentzialtasuna ezin zutela jakitera eman argudiatuz, ez genuen nahi izandako informazioa lortu, baina behintzat argibide batzuk bai jarri ziren mahai gainean: alde batetik, ospitalean erabltzen ziren teknologiak ezin baziren jakitera eman ere, gure proiekturako erabili beharreko softwarea ahalik eta irekiena izan behar zela eta ez zuela inolako kanpoko enpresekin loturarik izan behar; eta beste alde batetik, ospitalean software hori ezartzeko helburua ez zela ziurra, haien ezagutzaren gainean ez zegoen proiektu bat zelako eta guk egindako lanari buruzko memoria bat nahi zutelako aplikazioaren ezartzeko aukerak baloratzeko.

Oztopo horiek izan ziren, batez ere, web-aplikazioaren ezarpenaren helburua epe luzeago baterako planteatzea eragin zutenak.

Erabilitako softwareari dagokionez, esan beharra daukat aurretik ezagutzen ez nituen hainbat tresna erabiltzen ikasteak asko aberastu nauela, eta etorkizunari begira hainbat ate zabaldu nituela. *Python* programazio lengoaia ikasteko hainbat tutorial egin izanak proiektua lantzen hasi aurretik, eta *Django*-k nola egiten duen lan ikasteak eta ulertzeak denbora eraman badit ere, asko eskertu dut denbora aurrera joan ahala. Egia izan bada ere beste teknologia ezagunago batzuk erabili ahal nituela proiektua garatzeko, egia da baita ez nindutela egin dudan aukeraketa egin izanak ekarri didan ezagutza eta esperientzia ekarriko.

Lan-ingurune desberdinak ezagutzea, teknologia berriak erabiltzen ikastea eta ordutegiak eta helburuak finkatzen ikastea asko eskertu dut, epe luzera. Gainera, gaur egungo merkatua aztertzera ere eraman nau, *Python* lengoaiaren erabilera gero eta hedatuagoa dagoela ikustera eramanik, eta web-aplikazioak egiteko tresna oso egokia dela *Django* irakatsirik.

Horrenbestez, eta bukatzeko, esan beharra daukat asko ikasi dudala proiektu honekin. Ez bakarrik, erabilitako teknologiei daokioenez, baizik eta lan-munuan benetako erabilera izan dezakeen proiektu bat garatzearen garrantziari, bezeroaren betekizunak asetzeko ahaleginak egiteari, planifikazio egoki bat egiteari, eta, batez ere, gertatzen diren zailtasunei aurre egin behar izateari dagokionez.

Graduan zehar barneratutako ezagutzak praktikan jartzea lortu dut, eta horien bidez Gurutzetako ospitaleko Farmaziako departamentuan, nik garatutako softwarearen gaineko konformitate eta adostasun bat lortzea ere. Etorkizunari begira asko erakutsi didan proiektu bat izan da, ospitalean ezartzeko eta eguneroko bizitzan erabiltzeko aukera daukan software bat garatu dudala kontutan izanik.

14 Eskerrak Ematea

Proiektu honen eboluzioan zehar behar bezalako garrantzia izan duten pertsona batzuk aipatzea gustatuko litzaidake, haiek gabe, lan hau aurrera eramatea ez baitzen posible izango, gertatu den bezala.

Lehenik eta behin, nire eskerrak Juanan Pereirari, bera izan baita lan hau zuzentzen lagundu nauen pertsona nagusia, bai eta, errespetuz eta prestutasunez, hobekuntzak proposatu, akatsak zuzentzeko argibideak bilatzen lagundu eta epe luzean eskertu dudan jarraitasun batekin lan egiten lagundu nauena. Berari zor diot, baita ere, teknologia berriekin lan egitera bultzatu izana. Hasiera batetan ezagutzen ez nituen teknologiekin lanean hastea interesgarria iruditu zitzaidan, baina esan beharra daukat ez zela erraza izan graduan zehar sakondu ez dugun lengoaia eta teknologiekin lanean dihardutea. Ikasketakarga handitu da, bai, baina Juanan izan da, nigan, teknologia horien erabileran trebatzera motibatu nauen pertsona, eta hori, gaur, asko eskertzen dut.

Larraitz Zubeldiari eta Maider Azanzari ere aipamen bereziak zor nahi dizkiet. Larraitzek euskararen arloari dagokionez, artikulu zientifiko-teknikoak idazteko beharrezko oinarriak ezagutzen lagundu eta irakatsi didalako, proiektuaren memoria egiterako orduan hain garrantzitsuak izan diren kontzeptuak barneratzen lagunduz. Eta Maiderri, Gradu Bukaerako Lanari dagozkien kontutan izan beharreko elementuak zein diren eta nola landu behar diren ezagutzen lagundu didalako.

Jarraitzeko, ez nintzateke justua izango ez banitu graduan zehar izan ditudan maisu guztiak aipatuko, guztietatik ikasi baitut zerbait; eta ez bakarrik komunikazio-teknologien edota programazio-lengoaien inguruko kontzeptu berriei buruz, lan-metodologia, bikaintasunaren bilaketa eta etengabeko hobekuntzaren gaineko ezagutzari buruz ere haiek irakatsi baitizkidate kontzepturik garrantzitsuenak.

Nire ikaskideak ere aipatu nahiko nituzke, alor eta momentu askotan, maisuak irakatsi ezin duten kontzeptuak ezagutzen lagundu didatelako: adiskidetasuna, talde-lana eta hainbat pertsonako proiektu bat aurrera eramateko beharrezkoa den lan-metodologia baten ikasketa, besteak beste. Eta beharbada garrantzi gehien ematen diodan aspektua: nik berezkoa dudan etengabeko bikaintasunaren bilaketa horren kontzientzia izaten laguntzea, ez baita beti komenigarria gorputzarentzat lan batetan perfekzioa lortzen saiatzea, baina bai helburu batzuk finkatzea eta horien bila joaten trebatzea. Diferentzia hori bereizten lagundu didatelako.

Gurutzetako ospitaleko Farmaziako departamentuko kide guztiei ere esker bereziak zor dizkiet. Oso harrera bikaina eman didate ospitalera joan naizen bakoitzean, departamentuan lan egiten den metodologiari buruzko eskatzen nituen datu guztiak erraztuz. Oso gustora egon naiz lanean haiekin; eta batez ere, eskerrik asko entsegu eta errezeta klinikoen arduradunari, Alazne Bustinzari, telefonoz, email-ez zein pertsonalki lagundu bainau, benetako entsegu eta errezeten gaineko informazioa eskainiz eta bere eguneroko laneko proba kasu errealak egiteko denbora ateraz. Eskerrik asko nire lana erosoagoa izatea laguntzeagatik.

Eta bukatzeko, nire ingurugiro hurbilean izan ditudan horiei; nire bikotekideari; gurasoei eta anaiari; betiko lagunei; guzti horiei eta aurretik aipatutakoei, eskerrik asko bere laguntza eta irakaskuntzagatik, beti pertsona ekina eta ikastuna izan banaiz ere, limiteak zeintzuk diren ikusten eta aukerak eta posibilitateak hobeto baloratzen erakutsi didatelako.

15 Kontsultatutako Webguneak

Memoria idazten laguntzeko baliabideak eta erreferentziak:

- LaTex ikasteko tutorialak:
  http://www.latex-tutorial.com/tutorials/
  http://www.fing.edu.uy/ canale/latex.pdf
- OverLeaf eta Git bitarteko elkar-komunikaziorako baliabide-orria:
  http://ttiki.com/341130
  http://ttiki.com/341131
- Euskarazko idazkerarako baliabideak:
  Elhuyar hiztegia: http://ttiki.com/341132
- Aplikazioaren inplementaziorako baliabideak:
  - Python-en programatzeko dokumentazioa eta laguntza:
    https://docs.python.org/3/
  - *Python*-en bidez PDF fitxategiak sortzeko laguntza orrialdeak:
    https://docs.djangoproject.com/en/1.8/howto/outputting-pdf/
    https://www.reportlab.com/docs/reportlab-userguide.pdf
  - *Django* ikasteko tutoriala:
    http://ttiki.com/341133
  - Django-rekin sakontzeko orrialdea:
    https://docs.djangoproject.com/en/1.7/intro/overview/
  - Kodean konfirmazio-mezuak jartzeko erreferentzia-kodea:
    http://alertifyjs.com/
- Interfazaren itxura lantzeko baliabideak:
  - "Twiter Bootstrap"orrialdea:
    http://ttiki.com/341134
- DigitalOcean-en aplikazioa abiarazteko laguntza-orrialdea:
  http://ttiki.com/341135
- Portadako irudi probisionala: http://ttiki.com/341136