eman ta zabal zazu
Universidad del País Vasco
Euskal Herriko Unibertsitatea
INFORMATIKA FAKULTATEA
FACULTAD DE INFORMÁTICA

Informatika Ingeniaritzako Gradua

Software Ingeniaritza

Gradu Amaierako Lana

Akta Gardena: Akta multimediak hartzeko, editatzeko eta dokumentatzeko sistema

Egilea

Adrian Artola Korta

2019

eman ta zabal zazu
Universidad del País Vasco
Euskal Herriko Unibertsitatea
INFORMATIKA FAKULTATEA
FACULTAD DE INFORMÁTICA

Informatika Ingeniaritzako Gradua

Software Ingeniaritza

Gradu Amaierako Lana

Akta Gardena: Akta multimediak hartzeko, editatzeko eta dokumentatzeko sistema

Egilea

Adrian Artola Korta

Zuzendaria(k)

Rosa Arruabarrena Santos

Laburpena

Gradu amaierako proiektu honetan, Iametzak dagoeneko sortuta duen tresna gainditukoduen udaletxeen akta multimediak grabatzeko, editatzeko eta dokumentatzeko tresna berri bat garatuko da zerotik.

Iametza enpresak azken urtean udaletxeetan aktak grabatzeko tresna bat sortu zuen. Hala ere, tresna honek muga handiak ditu eta orokorrean nahiko aukera gutxi eskaintzen ditu. Egoera honi aurre egiteko eta produktu hau gainditukoduen produktu berri bat sortzeko erabaki da proiektu hau sortzea.

Sistema honekin, udaletako biltzarretan akta multimedia hartuko da. Ondoren, akta multimedia honekin lan desberdinak egin nahiko dira: interbentzioak editatu, transkribapenak sortu, datuak publiko egin, aktak dokumentatu eta beste hainbat aukera.

Proiektu hau hasieratik EHUko Informatika Ingeniaritzako Graduko ikasle batek egina izango da Iametza enpresarentzako.

Eskerrak

Proiektu hau garatzea oso baliagarria izan da nire formakuntza pertsonalarentzako. Azken 4 urte hauetan asko ikasi dut eta lan hau, urte hauetan guztietan jaso dudan ezagutzaren islada da. Horregatik, eskerrak eman nahiko nizkieke lau urte hauetan unibertsitatean egin ditudan lagunei eta izan ditudan irakasleei.

Iametzako lankideei ere eskerrak eman nahiko nizkieke, batez ere *Julen*i nire tutorea izateagatik, bai bertan egindako praktiketan eta bai GrAL honetan. Lankide paregabeak izan dira eta izan dugun harreman bikaina asko eskertzen dut.

Eskerrak eman nahiko nizkioke *Rosa*ri, berak gidatu bainau proiektu guztian zehar; alegia, bere tutoretza beharrezkoa izan da proiektuaren inplementaziorako eta memoriaren garapenerako.

Gaien aurkibidea

Irudien aurkibidea

Taulen aurkibidea

1. KAPITULUA

Sarrera

Kapitulu honetan proiektuaren nondik norakoak eta aukeratu izanaren zergatiak azalduko dira ahalik eta modu sinplean. Horretarako, lehenik eta behin, Iametza enpresaren deskribapenaz gain izandako aurrekariak azalduko dira.

1.1 Iametza

Gradu amaierako lan hau Iametza enpresan garatu den proiektu bat da, dagoeneko existitzen zen proiektu baten ideiatik abiatuz, baina kodea baztertuz.

Iametza Interaktiboa Ametzagaina taldeko enpresa bat da, Adur, Antza eta Argia enpresekin batera. 10 urte bete ditu aurten eta komunikazioa eta teknologiaren arloan lan egiten du.

1.2 Iametzan praktikak

Iametza enpresan egindako praktikak 2018ko maiatzean hasi ziren. Iametzarekin kontaktua Adur enpresaren bitartez eman zen, Adur enpresak Iametzan aurrera eraman nahi zen proiektu baten berri zutelako. Praktika hauek onartu nituenean lana egiten hasi nintzen Iametzan arlo desberdinetan jorratzen zuten lankideekin eta nire aurrerapenak kontrolatuko zituen tutore batekin.

Praktiken helburua web-aplikazioen garapenean sakontzea izan zen. Horretarako, Farapi aholkularitza enpresarentzako, enpresa desberdinen erantzunkidetasun soziala neurtzen duen tresna bat garatu nuen. Honetaz gain, beste proiektu batzuetan ere lan egin nuen produkzioaren parte gisa, esaterako, Etxegiroan izeneko audio liburuak entzuteko web orri batean eta Errenteriako udalarentzat herritarren arretarako HAZ tresnan.

Iametzan egin nituen udako praktikak oso baliagarriak izan ziren web teknologietan sakontzeko. Praktikan zehar erabili ziren teknologiak HTML, CSS, JS, JQUERY, Laravel, PHP eta MySQL izan ziren, eskaintzan esaten zenarekin bat etorriz.

1.3 Farapi tresna

Farapi tresna gisa erreferentzia egin arren, tresnaren izena Erakide da. Tresna honen helburua aholkularitza enpresa desberdinek, bezero dituzten enpresen erantzunkidetasun soziala neurtzea da.

Tresna honek ere aurrekariak zituen eta helburua lehenago erabiltzen ziren tresnak web-aplikazio gisa garatzea zen. Tresna honetaz gain, alde publiko bat eta administrazio alde bat garatu ziren tresna aurkezteko eta kudeatzeko.

Proiektu honen garapena izan zen praktiken helburu nagusia eta proiektu honi esker barneratu nintzen web teknologiaren munduan bere osotasunean. Graduko hirugarren urtean landu ziren irakasgai desberdinez baliatuta (Web sistemak, Pertsona eta Konputagailuen arteko Elkarrekintza eta Datu-Baseen Diseinua batez ere) praktiken prozesua errazagoa izan zen.

1.4 Akta Gardena Proiektua

Akta Gardena udaletxeetako biltzarren akta multimedia formatuan eratzeko erreminta bat da. Tresna hau, nahiz eta modu sinplean garatuta, dagoeneko garatuta dagoen tresna bat da eta hainbat udalek erabiltzen dute. Tresna C++ lengoaian idatzia dago eta sistema eragile desberdinetarako garatuta dago. Mahai-gaineko plataforma anitzeko aplikazio bat izateak izugarrizko buruhausteak sortzen ditu tresnaren mantentze-lanak, horregatik, hau hedatu ordez web teknologietan oinarritzea erabaki zen hasieratik, GrAL honen proiektua sortuz.

Izenak argi esaten duen bezala, Akta Gardenaren helburua udaletxeen gardentasuna handitzea da eta herritarrei zinegotzien eta udaletxearentzako lana egiten dutenen lana gerturatzea da.

Dagoeneko funtzionamenduan dagoen aplikazio honek dituen mugetatik abiatuta, hainbat hobekuntza pentsatu dira produktu honen erabilgarritasuna zabaltzeko, esaterako hizkuntza teknologien integrazioa.

2. KAPITULUA

Proiektuaren kudeaketa

Proiektua modu arrakastatsuan amaitu ahal izateko plangintza bat garatu da proiektuak jarraituko duen bidea ezartzeko, arriskuak aurreikusteko eta hauei nola aurre egin jakiteko, lanaren deskonposaketa bat egiteko eta ordu-kostua kalkulatzeko eta kontrolpean mantentzeko.

2.1 Proiektuaren irismena

Akta Gardena proiektuaren irismena, udal biltzarren aktak audioz eta bideoz jasotzeko eta hauen post-ediziorako balio duen software baten garapena da. Software honetan multimediaren tratamenduaz gain, hizkuntza-teknologiak barneratuko dira bai transkribapenerako bai etorkizun batean itzulpen automatikorako teknologietara zabalketa posiblea izan dadin. Software hau ez da jostailuzko prototipo bat izango, baizik eta esplotaziorako garatuko da.

Proiektuaren hasieran erreminta hau mahai-gaineko aplikazioarekin konbinatu nahi izan zen, aplikazioari post-ediziorako funtzionalitatea bakarrik emanez. Orain helburua dena tresna bakarrean integratzea da.

2.2 Proiektuaren helburuak

Proiektu honen helburua, Iametza enpresarekin batera produktu bat garatzen da. Produktu hau biltzarren aktak era errazean jasotzeko aplikazioa izango da, eta hasiera batean, udalei zuzenduta egongo da.

Produktu honek dagoeneko badu mahai gaineko aplikazioa (hain zuzen C++ lengoaian garatua dagoena), baina helburua web-aplikazio bat bihurtzea da. Horretarako, teknologikoki egon daitezkeen mugak aztertu behar dira. Hauek dira zerrendatzen diren helburu orokorrak, bai produktuaren inguruan eta bai kudeaketaren inguruan:

- Akta Gardena zaharreko mahai gaineko aplikazioak muga handiak dituenez, web-aplikazio baten bidez Akta Gardena proiektuaren funtzionalitateak zabaldu behar dira. Bilatzen dena ondorengoa izanik:
  - Biltzarren grabaketarako erreminta bat, denbora-markak sortzeko eta multimedia grabatzeko.
  - Akta gisa jaso den audioa transkribatzeko aukera izatea.
  - Biltzarren zehar hartzen diren denbora-markak editatzeko post-edizio zatia izatea.
  - Akta osatzen duten dokumentu guztiak digitalki sinatu ahal izateko aukera barne izatea.

- Iametzak ikertu dituen multimedia aktak hartzeko beste softwareak gainditzea hizkuntza-teknologien integrazioarekin.
- Multimedia-zerbitzuen ezaugarrien inguruan ikertzea eta hauen integrazioa egin interfazean (audio grabaketak egiteko esaterako).
- Sinadura digitalen inguruan ikertzea etorkizunean Iametzak beharko dituen proiektuetan barneratzeko. Besteak beste, jasotako informazioaren babes bermea egon dadin.
- Plangintzan egindako ordu-estimazioa errealista izatea eta ahal den heinean hau betetzea.
- Jarraipen eta kontrol jarrai baten ostean denboraren desbideraketa txikia izatea.

2.3 LDE diagrama

Proiektuaren garapenak inplikatzen dituen lan eta ataza guztien deskonposaketa da 2.1 irudian eta 2.1 taulan ikusi daitekeen LDE diagrama. Diagrama honetan ikus daitekeen bezala hiru zati nagusitan banatu dugu proiektua: Produktua, kudeaketa eta dokumentazioa. Hiru ardatz nagusi hauen inguruan beste ataza eta lan-pakete batzuk identifikatu dira.

Produktua bi zatitan banatzen da: formakuntza eta garapena. Formakuntza lan honen aspektu garrantzitsu bat da, Laravel ikastea lehen puntua da, produktua Laravel Frameworkean oinarritzen baita. Enpresaren azken garapenetan hartu den erabakiaren ondorioz, garapenak framework honekin egiten dira. Hala ere, Laravel ikastea formakuntza pertsonalerako ere garrantzitsua da. Multimedia zerbitzuei buruz eta sinadura digitalei buruz ikertzeak ordea oraindik, pisu gehiago du enpresarentzat, aplikazioak erabiliko dituen zerbitzuez gain, Iametzaren etorkizuneko proiektuetan puntu garrantzitsuak izango direlako. Produktuaren bigarren zatiari dagokionez, hau da, garapenari dagokionez, lau puntutan banatzen da. Alde batetik produktuaren analisia, diseinua eta probak ditugu eta bestetik hiru atazetan banatu dezakegun inplementazioa dugu. Lehenengo ataza maketazioa da, hau da, ikusgarri izango den interfaze grafikoaren maketazioa. Bigarren ataza post-edizioaren garapena izango da, hau izanik garapenari dagokionez lortu nahi den ezaugarri oinarrizkoena. Azkenik grabaketaren ataza dugu, oso lotua dagoena multimedia-zerbitzuei buruzko ikerketarekin.

Kudeaketan hiru ataza nagusi nabarmendu daitezke. Alde batetik bilerak ditugu, bai fakultateko tutorearekin eta bai enpresako beste kideekin. Beste batetik, plangintzari dagokionez, proiektuaren hasieran idatziko da proiektuaren garapenaren gida-plana izanik. Ulergarria da noski bertan idatzitako plana alda daitekeela proiektua aurrera doan heinean, kontuan hartu ez diren gertakarien ondorioz. Jarraipen eta kontrola azken ataza da eta, proiektua aurrera doan heinean, ataza hau ere aurrera joango da.

Dokumentazioaren zatia da azkena eta bi ataza ditu, memoria idaztea eta lanaren defentsa prestaketa. Lanaren defentsa proiektuan egingo den azken ataza izango da.

2.4 Balioespenak

2.2 taulan ikus dezakegunaren arabera, nabari da denbora handiena produktuari ematen zaiona dela. Produktuko lan-paketeak, denbora asko behar duten bi lan-pakete ditu bere barnean, formakuntza eta garapena. Formakuntza barruan ikerketa prozesua ere bildu da eta honek ordu kopuru handia gehitzen dio lan-paketeari. Produktuari dagokion denbora-estimazioa nahikoa dela pentsatzen da; hala ere, posiblea da luzatzea, formakuntzaren ondorioz, garapena atzeratu eta luza daitekeelako.

Estimazioaren arabera, dokumentazioa da hurrengo lan-pakete luzeena. Memoriaren idazketa ataza luzea da ukitu behar diren puntu guztien ondorioz, eta ataza honek ematen dio dokumentazioari duen pisua proiektuan. Defentsaren prestaketak ordea ez du hain denbora handia esleitua, aurretik sartuko diren ordu guztien ondorioz proiektua ondo ezagutuko delako.

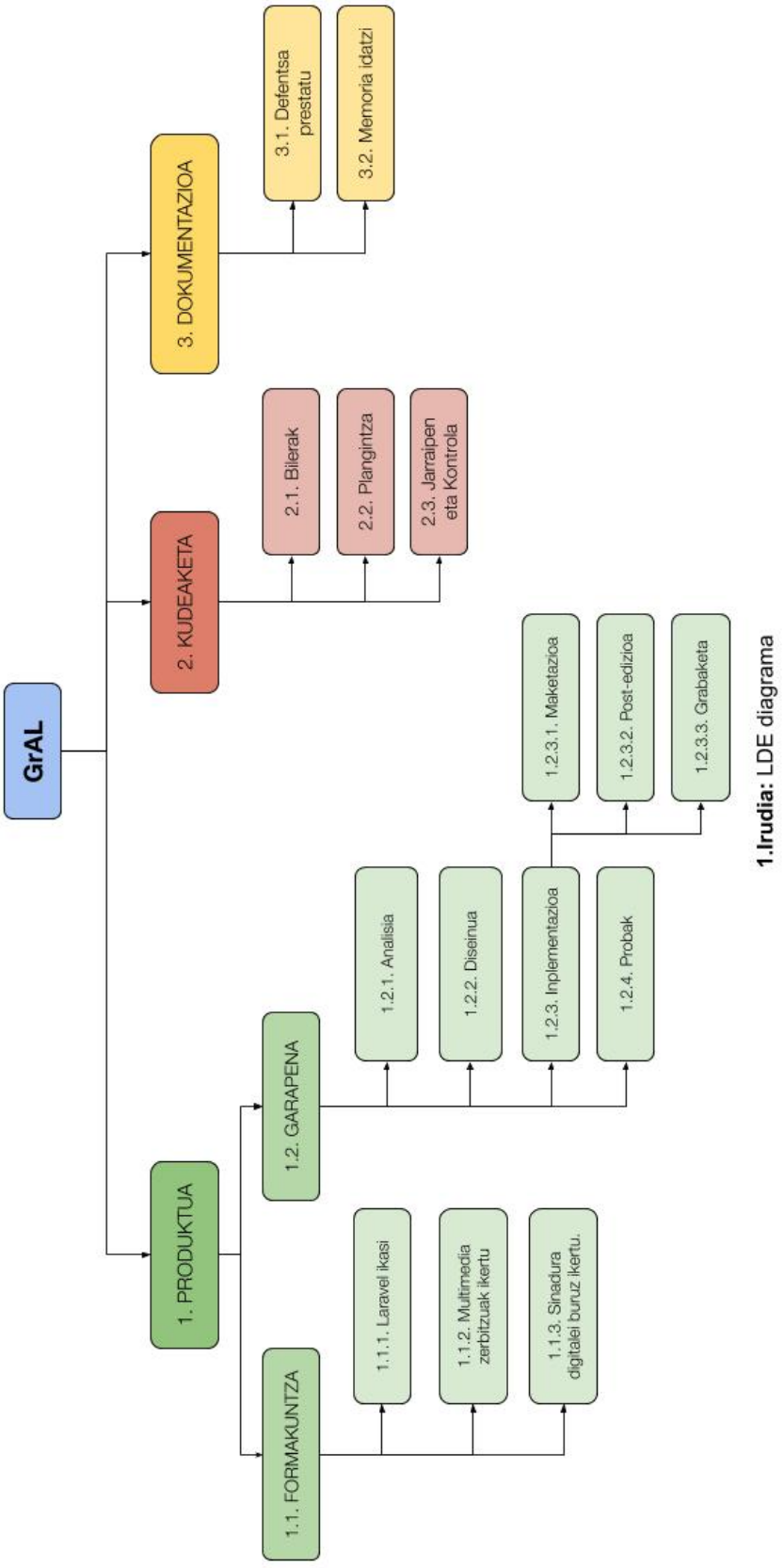
Ordu gutxien beharko duen lan-paketea kudeaketarena da. Bileren iraupena luzatu daitekeen arren, estimatu den denbora egokia dela deritzogu. Plangintza hasieran egingo den dokumentu hau bera da eta jarraipen eta kontrola ordea proiektuaren bizian zehar idatziko den dokumentua da.

2.5 Emangarriak

Emangarrien zerrenda zati desberdinetan bana dezakegu beraien arteko erlazioan oinarrituta. Emangarri desberdinek mugarri desberdinak dituzte eta hauen ordenak garrantzia du ataza desberdinek elkarrekin duten erlazioaren ondorioz. Erlazio hauetan oinarrituta, 3 zatitan banatu dugu emangarrien zerrenda.

Kodea	Ataza	Azalpena
Produktua		
1.1.1	Laravel ikasi	Laravel PHP framework zabalak eskaintzen dituen abantailak erabiltzen ikasteaz gain, ezartzen dituen mugak ikasi.
1.1.2	Multimedia zerbitzuak ikertu	Front-end teknologietaz baliatuz, multimedia-zerbitzu egokienak ikertu audio grabaketak egin ahal izateko.
1.1.3	Sinadura-digitalei buruz ikertu	Akta multimediak sinadura-digitala jasotzeko egin behar den prozesuari buruz ikertu.
1.2.1	Analisia	Akta Gardenaren analisia, erabilpen kasuak definitu eta domeinua finkatu.
1.2.2	Diseinua	Akta Gardenaren interfaze grafikoen eta sistemaren diseinua.
1.2.3.1	Maketazioa	Aplikazioaren Interfaze grafikoa inplementatu.
1.2.3.2	Post-Edizioa	Akten post-ediziorako erreminta guztiak inplementatu eta hizkuntza-teknologiekin konbinatu transkribapenak egiteko.
1.2.3.3	Grabaketa	Akten grabaketa egiteko erremintak inplementatu eta multimedia zerbitzuak integratu.
1.2.4	Probak	Multimedia zerbitzuen eta sinadura digitalen probak egin.
Kudeaketa		
2.1	Bilerak	GrAL-a gidatzeko tutorearekin adostutako bilerak egin eta enpresan proiektuaren garapenean zehar behar diren bilerak egin.
2.2	Plangintza	Proiektuak behar duen plangintza egitea: lanaren deskonposaketa, orduen estimazioak, arriskuen analisia, lan bakoitzaren egutegia, irismena etab.
2.3	Jarraipen eta kontrola	Plangintzan idatzitakoa bete dela bermatu eta honi buruz kontuak emantea, ordu estimazioaren desbiderapena etab.
Dokumentazioa		
3.1	Defentsa prestatu	Epaimahaiaren aurrean defentsa egin aurretik proiektua biltzen duen aurkezpena prestatu.
3.2	Memoria idatzi	Proiektuari buruzko dokumentazio guztia biltzen duen dokumentu hau idatzi.

2.1 Taula: Lanaren deskonposaketa taula.



1.Irudia: LDE diagrama

2.1 Irudia: LDE diagrama.

Izena	Estimazioa
Produktua	250 or
Formakuntza	80 or
Laravel ikasi	20 or
Multimedia zerbitzuak ikertu	30 or
Sinadura Digitalei buruz ikertu	30 or
Garapena	170 or
Analisia	10 or
Diseinua	10 or
Inplementazioa	140or
Maketazioa	30 or
Post-edizioa	70 or
Grabaketa	40 or
Probak	10 or
Kudeaketa	50 or
Bilerak	10 or
Plangintza	20 or
Jarraipen eta kontrola	20 or
Dokumentazioa	70 or
Defentsa prestatu	10 or
Memoria idatzi	60 or
Guztira	370 or

2.2 Taula: Denbora estimazioak.

2.5.1 Proiektuaren inguruko formakuntzaren emangarriak

Proiektuaren inguruko formakuntzatik eratorritako emangarriei buruz hitz egiten dugunean, multimedia-zerbitzuen inguruko ikerketari buruz eta sinadura-digitalei buruzko ikerketari buruz hitz egiten dugu.

- Aktak izan ditzaketen multimedia-zerbitzuei buruzko eta honen lokaleko tratamenduari buruzko ikerketa, 2019ko martxoaren 4erako
- Sinadura digitalei buruzko ikerketari dagokionez, 2019ko martxoaren 13rako.

2.5.2 Produktuaren emangarriak

Produktuaren emangarriei buruz hitz egiten dugunean, formakuntzatik aurrera dagoen zatiari buruz hitz egiten dugu.

- Akta Gardena softwareak izango dituen eskakizun eta zehaztasuna guztiak bilduko dituen analisia eta diseinua egun bererako dira, 2019ko martxoaren 21erako.
- Akta Gardena softwarearen garapena eta aplikazio honi egingo zaizkion proba guztiak 2019ko maiatzaren 22rako egongo dira.

2.5.3 Kudeaketaren emangarriak

Kudeaketaren inguruan identifikatu diren emangarriak plangintza eta jarraipen eta kontrola dira.

- Irismena, helburuak, lanaren deskonposaketa, denbora estimazioak, emangarriak, Gantt diagrama, etab. elkartzen dituen plangintza 2019ko otsailaren 13rako.
- Egindako plangintzaren eta benetan burutu den denbora errealean arteko jarraipen eta kontrolaren dokumentua 2019ko ekainaren 17rako.

2.5.4 Dokumentazioaren emangarriak

- Proiektuaren memoria idatzi osoa 2019ko ekainaren 17rako. Memoria bidaltzeko azken eguna ekainaren 23a izango da.

2.6 GANTT diagrama

2.2 irudian Gantt Diagrama azaltzen da, bertan, lanaren banaketa ageri da egunetan zehar. Proiektuaren hasiera 2019ko otsailaren 4a da eta proiektuaren bukaera, 2019ko ekainaren 14a izango da. Hala ere, proiektuaren defentsa prestakuntza 2019ko uztailaren 1ean bukatuko da. Proiektuaren plangintza, 2019ko otsailaren 13an bukatuko da, jarraipen eta kontrola proiektuaren zehar garatuko den bitartean.

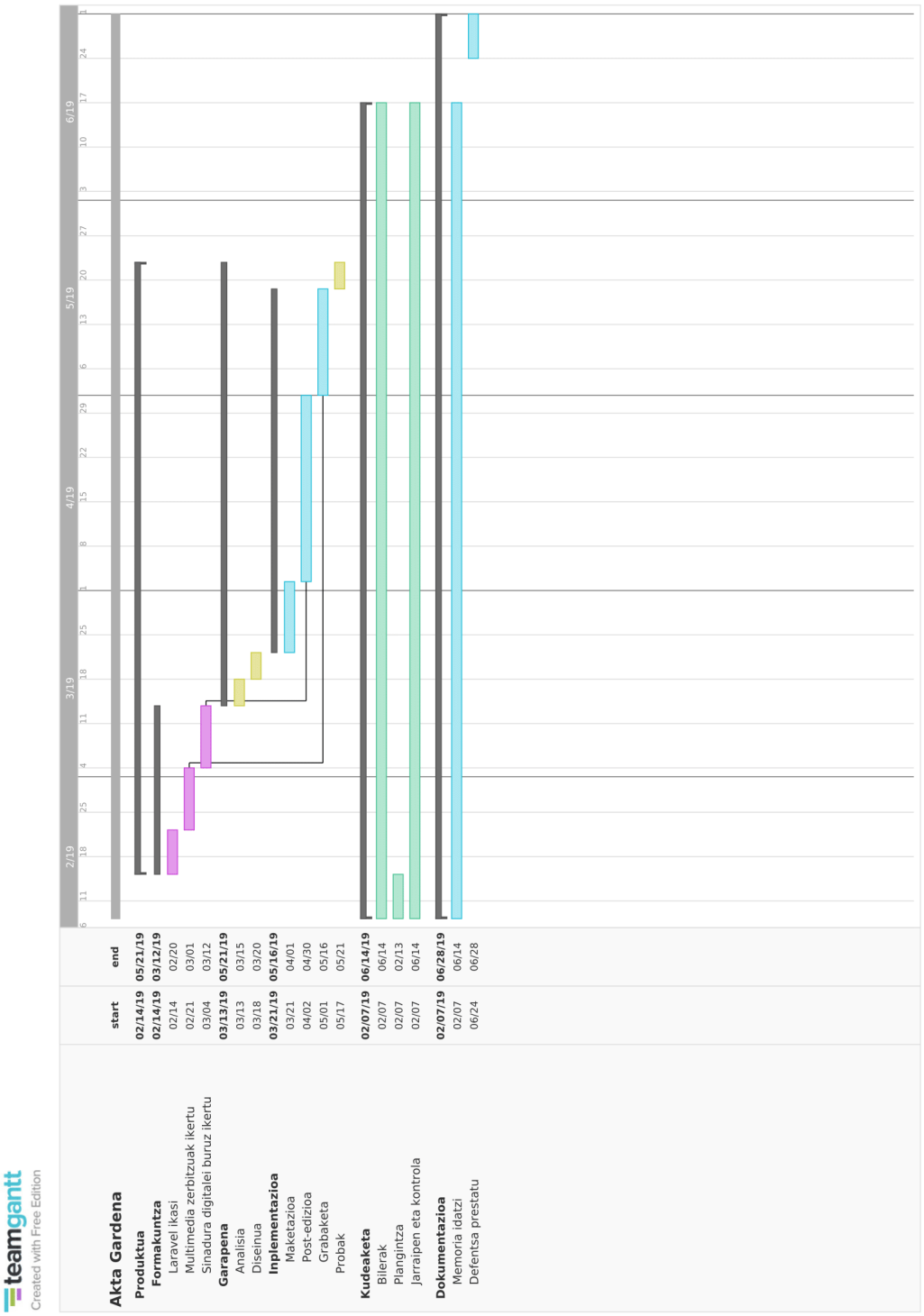
Plangintza da proiektuaren abiapuntua eta honek erabakitzen du proiektuaren prozesua nola aurrera joango den. Ondorioz, produktua landu aurretik plangintza egingo da. Ondoren produktura joango da prozesua, formakuntzara hain zuzen ere. Formakuntzari ia hilabete bat emango zaio, 2019ko otsailaren 14tik 2019ko martxoaren 12ra. Formakuntza lehenago egiteak arrazoi logikoa du, garapenera pasatu aurretik beharrezkoa baita honi buruzko informazioa biltzea eta lantzea. Garapena hastean egingo den lehenengo gauza proiektuaren analisia da eta, hau bukatzean, diseinura joko da. Behin analisia eta diseinua bukatuta, inplementaziora joko da, honek ia bi hilabete iraungo du 2019ko martxoaren 21etik 2019ko maiatzaren 16ra. Azkenik probak egingo dira eta garapen prozesuarekin amaituko da.

Kudeaketako jarraipen eta kontrolarekin gertatzen den bezala, memoriaren dokumentuarekin eta bilerekin berdina gertatzen da. Proiektua aurrera doan heinean idazten eta osatzen joango dira. Memoriaren dokumentua 2019ko ekainaren 14rako amaitzea estimatzen da, baina honen defentsa beranduago egingo da, aurkezpenak egingo diren astearen aurreko astean hain zuzen ere, 2019ko ekainaren 24tik 2019ko ekainaren 28ra.

2.7 Arriskuak

Hainbat arrisku identifikatu dira proiektuaren izaeragatik honen garapen egokia baldintzatu dezaketenak. Arrisku hauek bere kalte potentzialengatik eta gertatzeko probabilitatengatik sailkatu dira.

1. **Informazioaren galera:** Informazioaren galera arazo erabat kritikoa izan daiteke, kontuan hartzen badugu proiektu honen zati guztiak digitalak direla. Proiektua interneten kokatuta dagoenez arrisku hau ekiditeko hainbat neurri hartu dira: Garapena GitLab-en egoteaz gain, Iametza-ko zerbitzarietan gordetzen da, beraz galera



2.2 Irudia: Gantt diagrama.

arriskua baxua da. Beste dokumentazioa Drive plataforman gordetzen da eta kontutan hartuta Google enpresaren fidagarritasuna altua dela, arriskua oso baxua bezala jotzen da.

2. **Ikerketa prozesuan arazoak:** Dagoeneko aipatu den bezala, proiektu honen aspektu nagusi bat ikerketa prozesua da. Esanguratsuak dira ikerketa oker batek izan ditzakeen ondorioak beste atazekin duen loturagatik eta produktuaren inplementazioan izan dezakeen kaltearengatik.

3. **Osasun arazoak:** Gaixotzeko arriskua nahiko ohikoa da eta modu batean edo bestean proiektuaren epeak betetzea baldintza dezake. Kontuan hartu beharreko da proiektuan bi zuzendari daudela, bata fakultatean eta bestea enpresan eta hauen osasunak ere proiektuaren garapen egokia baldintza dezake. Har daitekeen neurri bakarra osasuna ondo zaintzea da eta lan-istripuen aurrean protokolo egokia izatea. Gainera lanera egunero kotxez joatea arriskutsua da osasunarentzako gerta daitezkeen trafiko istripuengatik.

4. **Garapenean arazoak:** Garapenean arazo ugari gerta daitezke, atzerapenak dira esanguratsuenak, baina arazo teknikoak ere gerta daitezke garapenaren funtzionamendu egokia arriskuan jarriz. Lehen puntu hori plangintza egoki batekin saihes daiteke estimazio hobea emanez, bigarren puntua ordea Git teknologia bitartez ekidin daiteke.

5. **Denbora-epeak ez betetzea:** Arazo nahiko ohikoa da hau, eta gertatzeko probabilitatea ere altua da egiten den denbora estimazioa ez denez ehuneko ehunean fidagarria nahiz eta honetan oinarritzen diren epeak. Honek izan ditzakeen ondorioen larritasuna desberdina da, desbiderapen hau ataza desberdinei lotuta baitago eta ez baitie modu berean eragiten.

6. **Estimazio irrealak:** Denbora estimazio irrealak nahiko ohikoak dira, eta batez ere ataza asko dauden proiektuetan. Estimazio irreal batek izan dezakeen eragina modu desberdinetan gerta daiteke, motz geratzea, gertu geratzea baina atazei ordu dedikazioa aldatzea eta luze geratzea estimazio optimistegiak erabiltzeagatik. Arrisku hau saihesteko, berriz, plangintza on bat da onena.

2.3 taulan ikus daitekeen bezala arrisku nagusia 6. arriskua da. Estimazio irreal batek eragin handia baitu proiektuarengan. Nahiz eta 5. arriskua kalte handia dela idatzi den, kontuan izan behar dugu arazoaren puntuan azaltzen dena. Larritasun maila handiagoa

Probabilitatea Kaltea	%15	%30	%45	%60	%75	%90
Baxua						
Ertaina						
Altua	3	2, 4, 5			6	
Oso altua	1					

2.3 Taula: Arriskuen eraginaren eta probabilitatearen taula.

du memoria epean ez emateak, emangarri bat berandu emateak baino. Azkenik, larritasun handiena duen puntua, informazioaren galera litzateke. Hala ere, hau gertatzeko probabilitatea oso baxua da.

2.8 Kalitate plana

Proiektuaren kalitatea bermatzea oso garrantzitsua da arrazoi askorengatik. Esaterako, proiektuaren mantentzea ahalbidetzea oso garrantzitsua da kontuan badugu proiektuaren garapenak etorkizunean jarraituko duela eta erosle desberdinek eskaerak eta gomendioak egingo dituztela. Gainera, proiektu honek jasotzen duen nota oso garrantzitsua izango da egilearen bizitza-akademikoan.

- **Dokumentuen kalitatea:** Proiektua aurrera doan heinean sortzen diren dokumentu guztiek kalitate maila altua dutela bermatzea garrantzitsua da: Tipografia, ortografia, formatuak eta estiloak zainduko dira.

- **Kodearen kalitatea:** Inplementazioaren mantentze-lanetan laguntzeko, garrantzitsua da kalitatezko kodea idaztea, hau ez da mugatzen iruzkinen idazketara noski, softwarearen garapenean existitzen diren patroi desberdinak errespetatzea eta Laravel Framework-ak inposatzen dituen arauak errespetatzean oinarritzen da, etorkizunean proiektuarekin jarraitu behar duen pertsonak erraz jarrai ahal izan dezan.

- **Jarraipen eta kontrolaren kalitatea:** Proiektuaren bizi-zikloari buruz ondorioztatzen ditugun datuak egiazkoak izateko, egin behar den jarraipena egunerokoa da. Jarraipen eta kontrol txostena prozesu guztian zehar idatziko da, eta orduen kalkulu aproposerako bi teknika erabiliko dira. Alde batetik Excel taula batean, lanean inbertitutako orduak gehitzen joango dira; eta bestetik, enpresan erabiltzen den ordu inputaketa softwareaz baliatuko gara.

- **Denboraren inbertsio egokia:** Denbora ondo banatzea bezain garrantzitsua da banatutako denbora hori ondo erabiltzen jakitea. Ondorioz, garrantzitsua da lan-giro aproposa eta erosoa izatea produktibitatea handitzeko. Lankideekin egiten diren jarduerak laguntzen dute giro egokia izaten eta ondorioz produktibitatea handitzen, atsedena behar denean atseden hartzeak ere laguntzen du.

2.9 Komunikazio eta informazio sistemak

Komunikazio eta informazio sistemak erabakitzeko, testuingurua ulertzea garrantzitsua da. Komunikatzeko arrazoiaren izaerak komunikazio mota bat edo bestea aukera dezake.

Dagoeneko aipatu den bezala, Iametza-ko zerbitzariez gain eta konputagailu pertsonalaz gain, informazioa Google Drive bitartez gordetzen da. Tresna hau aukeratzeko arrazoia argia da, Googlek eskaintzen duen lan-ingurumenak ahalbidetzen duen integrazioa (Google Drive, Gmail, Google Docs etab). Bertsio-kontrolerako Git teknologia ere erabiltzen da segurtasunerako. Honetara Ubuntu 18.04 gainean exekutatzen den edozein nabigatzaileren bitartez iritsi ahalko da.

Komunikazioari dagokionez, bi komunikazio mota desberdin bereiz ditzakegu, formala eta ez formala. Komunikazio ez formala konstantzirik utzi behar ez denean erabiliko da, galdera arruntak adibidez edo eguneroko gauzak. Horrelako komunikazioa enpresako tutorearekin egingo da. Komunikazio formala normalean erabiliko den komunikazioa da, normalean fakultateko tutorearekin erraz atzitzeko informazio zehatz bat.

2.10 Interesatuak

Hainbat interesatu identifika daitezke proiektu honen inguruan, interesatu bakoitza bere erara noski. Alde batetik, proiektuaren egilea bera da lehen interesatua, proiektuaren garapen egokiaren ardura berea baita eta aukera paregabea baita etorkizun baterako arlo honetan esperientzia izateko. Proiektuaren egilearekin erlazioa duten bi pertsona nabarmedu ditzakegu:

- Rosa Arruabarrena, Informatika fakultateko irakaslea eta proiektuaren garapen egokia gainbegiratu eta gidatuko duen tutorea.

- Julen Irazoki, Iametza enpresako garapeneko langilea eta enpresak izendatutako tutorea, garapenean eta formazioan lagunduko duena.

Bigarren interesatua Iametza enpresa da. Enpresaren helburua Akta Gardena proiektua udaletxe askotan zabaltzea da eta dagoeneko eginda dagoen proiektua gainditzea eta hobetzea da.

Hirugarren interesatuak Iametzaren lana iritsi daitekeen udaletxe desberdinak dira. Hauek gardentasunarengatik apustua egin ahal izateko eta herritarren aurrean kontuak emateko aukera paregabea baitute tresna honi esker.

Azken interesatua lan hau ebaluatu eta epaituko duen epaimahaia da. Epaimahaiaren eginkizuna aldez aurretik ezarritako irizpideez baliatuz lan hau ebaluatzea izango da.

3. KAPITULUA

Teknologiak

Atal honetan web-eko zatia eta mahai gaineko zatia garatzeko dauden aukera desberdinen analisi bat egingo da egokiena aukeratuz[1] (ikusi 4 kapitulua).

Mahai gaineko zatia garatzeko dauden teknologia desberdinen analisi bat egingo da, lehenagoko Akta Gardenaren fasean izandako esperientzian oinarrituta.

Azkenik, garapenean erabiliko diren teknologiak eta tresnak aipatuko dira bai eta zertarako erabiltzen diren.

[1] Dokumentuan aurrerago azaltzen den bezala, enpresa erabaki bat hartu zen aplikazioak bi zati izateko. Horregatik, 4 kapitulua irakurtzea gomendatzen da

3.1 Web teknologiak

Web-teknologiaren munduan eskaintza oso zabala da gaur egun, aukera gehienak informazio ugarirekin sarean. Programazio-lengoaia asko erabiltzen dira gaur egun web-aplikazioak garatzeko: **Python**, **C#**, **PHP**, **Java** eta beste lengoaia gehiago.

Analisi honetan, aipatutako lengoaien gainean garatutako framework-ak aztertuko dira, kasu honetan: **Django**, **ASP.NET core**, **Laravel** eta **JSF**.

3.1.1 Django

Django[4] Python gainean sortutako framework bat da, web-aplikazioak garatzeko balio du eta Model-View-Template patroia errespetatzen du. Gaur egun oso erabilia da garapen azkarra promesten baitu. Datu-base askorekin elkarekintza onartzen duenez aukera ona da web-aplikazioak garatzeko.

3.1.2 ASP.NET core

ASP.NET teknologiak dagoeneko 17 urte ditu gure artean, *.NET framework*-eko zati gisa jaio zen eta C# edo VB.NET lengoaiak erabiliz web-aplikazioak garatzea ahalbidetzen digu. Gaur egun mundu osoan bigarren teknologiarik erabiliena da web-aplikazioak garatzeko eta aukera oso ona da.

Hala ere, ASP.NET klasikoa, hau da, *.NET framework*-aren muina Microsoften teknologia denez, mugatuta dago Windows sistema eragileetara. Microsoftek 2016an muga honi aurre egiteko, *.NET core* abiarazi zuen.

.NET core bere bigarren bertsioan dago dagoeneko, eta 2019ko irailean hirugarren bertsioa abiaraziko du. Framework berri honek abantaila asko ditu, baina plataforma anitzeko sistemaren arlotik, *.NET core*-ren abanataila nagusia edozein sistema eragile gainean erabili daitekeela da.[5]

3.1.3 Laravel

PHP web-aplikazioak sortzeko lengoaia nagusiena da, eta Laravel[7] PHP gainean garatutako framework askotako bat da. Laravel-en inguruan dagoen dokumentazio zabalak asko

laguntzen du edozein arazoren aurrean. Gainera, komunitate handia dago honen inguruan eta gehigarri ugari dago edozein gauzetarako.

Abantaila asko ditu baina datu-basearen kudeaketa, garapen azkarra eta dokumentazio zabala dira nagusiak.

3.1.4 JSF

JavaServer Faces (JSF) Java lengoaian oinarritutako framework bat da. Teknologia honi esker Javaren liburutegi estandarrak eskaintzen dizkigun abantailak web mundura eraman ditzakegu. Model-View-Controller (MVC) patroia jarraitzen du eta Javarako beste teknologia askorekin konbinatu ohi da, esaterako: Hibernate.

ASP.NET-en kasuan bezala JSF ez da teknologia berria eta bere merkatuaren zatia du web-aplikazioen munduan, oso erabilia baita kudeaketarako softwarean.

Abantaila nagusietako bat Javatik datorren garatzaile klasiko orok zailtasun maila txikia izango duela da.

3.2 Mahai gainerako teknologiak

Mahai gaineko aplikazioez hitz egiten badugu, eskaintza amaigabea da. Ikertuko ditugun teknologiak aukeratzeko erabiliko ditugun irizpideak ondorengoak izango dira:

- Plataforma anitzetarako izatea.
- Interfaze grafikoa izatea.
- Dokumentazio egokia izatea.

Aurreko irizpideak erabilita, iragazkia pasa ondoren bi aukera nagusi aztertu dira: **Java** eta **Qt**.

3.2.1 Java

Java programazio-lengoaia, lengoaiarik ospetsuenetarikoa da programazio munduan. Edozein Java aplikazio, Java Virtual Machine (JVM)[2] softwarea duen edozein ordenagailuk exekutatu ahal izateak eman dio ospe hori.

Arrazoi hau nahiko teorikoa da, sistema guztietan ez baitu portaera berdina, baina orokorrean ez du arazorik ematen. Esaterako, leiho bidez interfaze grafiko bat sortzeko erabiltzen den liburutegiak, Java Swing izenekoak, ez du portaera berdina sistema eragile guztietan, baina orokorrean ez da arazo gehigarria.

Sarean Java lengoaiaren inguruan dagoen informazioa oso zabala da, eta gainera liburutegi asko daude edozein arazori aurre egiteko.

3.2.2 Qt

Liburutegi honek C++ lengoaiari falta zaion plataforma anitzeko sistema izaera gehitzen dio eta interfaze grafikoen garapena eskaintzen du. Liburutegi nahiko zabala denez interfaze grafikoez gain datu-basera sarbidea, XML-rekin lan egiteko aukera eta funtzio gehiago eskaintzen ditu.[1]

3.3 Aukeratutako teknologiak

Web-ari dagokionez aukeratu den teknologia **Laravel** izan da. Aukeratzeko lehen arrazoia, arrazoi enpresariala da, enpresako aurreko garapenak Laravelen egin direlako. Proiektu hau ez denez amaituko aurkezpenaren ostean, garapen jarraia eta mantentzea behar du eta beraz komeni da enpresako garapen guztiak Laravelen eginak egotea. Gainera, proiektuaren dimentsiorako eta Laravel erabiltzen ikasteko aukera paregabea da hau.

Mahai gainekoari dagokionez aukeratu den teknologia **Java** izan da. Akta Gardenaren lehen bertsioa **Qt** gainean egin zen, baina hainbat gabezia eta arazo topatu ziren, garrantzitsuena mantenugarritasunarena da. Bertsio berri bat egin behar den bakoitzean eguneraketa gisa, exekutagarri berri bat sortu behar da sistema eragile bakoitzeko eta **Javak** aurre egiten dio arazo horri, JVM duen edozein makinak exekuta dezakelako.

[2]Java programazio-lengoaia edozein plataforman konpilatzerakoan, *bytecode* izeneko pila lengoaiako kodea sortzen da, denetan *bytecode* lengoaia berdina. JVM exekuzio denboran *bytecode* hau prozesadoreak ulertuko duen kodera itzultzen duen softwarea da.

3.3.1 FFmpeg

Garapena aurrera eramateko teknologiaz gain, bitarteko softwarea ere erabili da, FFmpeg kodegailiuaren kasua da hau. Kodegailu honen zeregina aplikazioan *.wav* audio fitxategiak, *.mp3* fitxategietan kodetzea da. Horregatik, kodeketa hori egin ahal izateko mahai gainekoak dei egin behar dio FFmpeg kodetzaileari.

3.4 Garapenerako teknologiak

Garapena bera oinarrituko den teknologiaz gain, atal honetan garapena aurrera eramateko erabiliko diren tresnak aipatuko dira.

3.4.1 Visual Studio Code

Microsoften aplikazio hau, kode editore arina eta boteretsua da. Oso programa erosoa denez web-aplikazioak garatzeko, web-eko zatia garatzeko erabiliko da. Eskaintzen dituen abantailak ondorengoak dira:

- Sintaxia nabarmentzen du kolore desberdinekin.
- Terminalera sarbidea eskaintzen du.
- Kodea automatikoki identatzen du.
- Bertsio kontrolerako Git zerbitzuekin elkarrekintza eskaintzen du.

3.4.2 Eclipse

Mahai gaineko zatia garatzeko erabiliko den plataforma Eclipse izango da. Java programazio lengoaian garatzeko erabili izan ohi da. Ahalmen handiko tresna da eta abantaila nagusienak ondorengoak dira:

- Refaktorizaziorako erraztasunak.
- Sintaxia nabarmentzen du kolore desberdinekin.
- Konpilatzailea barnean dakar.

- Gehigarri desberdinekin zabaldu daiteke.
- Kodea automatikoki identatzen du.

3.4.3 Chromium

Chromium proiektutik hainbat nabigatzaile jaio dira, famatuenak Google Chrome, Microsoften Edge berria eta noski Chromium bera. Web nabigatzaile bat izateaz gain, erreminta oso boteretsua da web-garapenean, denbora errealean aldaketak egin ditzakegulako webaren interfazean simulazio moduan.

Eskaintzen duen kontsolak gainera JavaScript lengoaiak izan ditzakeen arazoak erakusten dizkigu eta behar izanez gero aginduak eta funtzioak exekutatu ditzakegu.

3.4.4 phpMyAdmin

Laravel softwareak datu-basearen kudeaketa ona egiten du *migration*-ei[3] eta *seeder*-ei[4] esker. Hala ere, datu-basera sarbide zuzena izatea komenigarria da kasu batzuetan, batez ere probak egiten direnean sortzen diren ilarak ezabatzeko.

Datu-basera sarbide hori lortzeko erabiliko den erreminta phpMyAdmin da. Php lengoaian idatzitako software honek datu-base baten kudeaketa osoa ahalbidetzen digu modu bisualean.

[3]Datu-basearen egitura (taulak sortu, zutabeak aldatu, gakoak gehitu etab) kode bidez egiteko sistema bat da. Datu-basea *migration*-ekin eraikitzen bada, web-aplikazioa beste zerbitzari batean montatzen badugu, *migration*-ak exekutatu ditzakegu eta datu-basea egitura berdinarekin sortuko da.

[4]Datu-basean aurredefinitutako ilarak gehitzeko balio duen baliabidea da. *Migration*-ak bezala, zerbitzari berri batean montatzen badugu web-aplikazioa, datu-base berria ilara zehatz batzuekin betetzeko balioko du, esaterako: erabiltzaileak, textu itzuliak eta bestelakoak

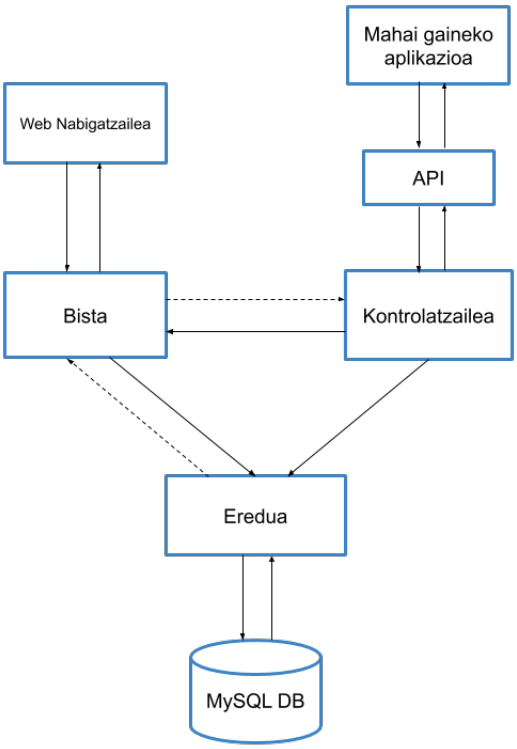
4. KAPITULUA

Arkitektura

Atal honetan aplikazioaren arkitektura globala azalduko da. Gainera, mahai gaineko aplikazioaren eta web-aplikazioaren arteko erlazioa eta komunikazioa azalduko da.

4.1 Arkitektura

Akta Gardena aplikazioak mahai gaineko aldea eta webeko aldea du. Banaketa honen arrazoia segurtasuna da, grabaketa baten zehar eskatzen dena segurtasuna eta fidagarritasuna baita. Lehenik eta behin web-aplikazioaren arkitektura azalduko da, ondoren mahai gaineko aplikazioaren arkitektura eta amaitzeko elkarren arteko komunikazioa azalduko da.



4.1 Irudia: Sistemaren arkitektura globala.

4.1.1 Sistema hibrido baten beharra

Akta Gardena aplikazioa, esan bezala, bi zatitan banatzen da, mahai gainekoa eta web zatia. Bi zatien existentzia justifikatzea beharrezkotzat jo da ondo ulertzeko.

Akta Gardenaren bertsio berri honekin, dagoeneko existitzen zen bertsioa web-aplikazio batekin ordezkatu nahi izan zen. Hasierako planteamenduaren arabera sistema osoa nabigatzaile batetik atzigarria izango zen, arazo ugari ekidinez (mantentzea batez ere). Hala ere, proiektuan aurrera joan hala, arrisku eta arazo tekniko batzuk aurkeztu ziren.

Topatu zen arrisku nagusia akten integritatearekin lotuta zegoen, web-aplikazio batean internet konexioa existitzea beharrezkoa baita. Interneteko konexioa galduko balitz, akta horrek ez luke inongo baliorik izango, audioa galduko litzatekeelako. Topatu zen beste arazoetako bat estandar digitalak ziren. Bestalde, gaur egun, hainbat udaletxetan, mikrofonia sistema analogikoak sistema digitalengatik ordezkatzen ari dira. Dante[6] bezalako protokolo digitalek abantaila nabarmenak eskaintzen dituzte modu honetako aplikazio batentzat. Esaterako, mikrofonoen detekzioa eskaintzen dute eta honela zein hizlari hitz egiten ari den jakiteko erabili litekeelarik.

Arazo eta arrisku hauei aurre egiteko, mahai gaineko aplikazio bat eraginkorragoa da, eta udal biltzarren iraupen osoan zerbitzua bermatu behar den kasuetan beharrezkoa ere. Horregatik, hain kritikoa den grabaketa prozesua mahai gainera eramatea erabaki zen eta beste guztia web-aplikazioan egingo da.

4.1.2 Web-aplikazioaren arkitektura

Web-aplikazioa aldiz, Laravel framework-aren bitartez garatu da. PHPko framework honek MVC patroia erabiltzen du. Web-aplikazioa da sistema guztiaren bihotza, bertan egiten baita datuen tratamendua eta kudeaketa. MVC patroia jarraitzen duenez, 3 zatitan banatu dezakegu aplikazioa: eredua, bista eta kontrolatzailea 4.1 irudian ikusten den bezala.

- Eredua: Datuen geruza irudikatzen du. Hau da, erabiliko den informazioa irudikatuko duen zatia da. Datuak aldatzeko eskaera kontrolatzailetik egingo da beti, ereduaren helburua informazio hau irudikatzea baita.

- Kontrolatzailea: Datuen prozesaketaz arduratzen den zatia da hau. Ereduak eraldatzeaz gain, erabiltzaileak jasoko duen bista prozesatzeaz ere arduratuko da. Erabiltzaileak egindako eskaerei erantzuna ematen dio.

- Bista: Eredua aurkezteaz arduratzen da, erabiltzaileak elkarekintza egokia izan dezan.

Diseinurako patroi honen bidez ardurak modu apropos batean antolatzea bilatzen da. Oso eredu erabilia da, batez ere web sistemak garatzeko, webean lan egiteko erara gehien urbiltzen den patroia baita.

Laravel-ek, MVC patroiaz gain, abantaila izugarri eskaintzen ditu web-aplikazio bat garatzeko. Esaterako, middleware-en erabilera edo azpitik dituen beste zerbitzuak garatzailearri kodeketa errazteko. Horretaz gain, gehigarri ugari topa ditzakegu beste funtzionalitate asko egiteko.

4.1.3 Mahai gaineko aplikazioaren arkitektura

Esan bezala, mahai gaineko aplikazioaren zeregina audioaren grabaketa prozesua aurrera eramatea da. Mahai gainekoa plataforma anitzeko aplikazioa izan dadin, honen garapena hasieratik Java-z egingo da. Honi esker, jasoko ditugun abantaila nagusiak ondorengoak dira:

- Javan idatzita egotean JVM duen edozein makinak exekuta dezake, beraz behin idatzita edonon exekutatu daiteke.
- Mantentzean lagunduko du konpilatu bakarra atera behar baita eta ez bat sistema eragile bakoitzeko.

Mahai gaineko aplikazioaren dimentsioak txikiak dira. Hala ere, mahai gaineko aplikazioaren arkitektura kontuan izan da gakoa diseinatzerako unean. mahai gaineko aplikazioa geruza desberdinetan garatu da, interfaze grafikoa, negozio logika eta datuen sarbidea banatuz, etorkizunean mantentze-lanak erraztuz.

4.1.4 Elkarren arteko komunikazioa

Mahai gaineko aplikazioaren eta web-aplikazioaren artean komunikazioa sortzeko Laravelen garatutako API bat erabiliko da, hau da, HTTP protokoloaren bitartez komunikatuko dira. Zerbitzu honen bidez mahai gaineko erabiltzaileak login-a egin ahalko du eta grabaketa egiteko prest dauden biltzar guztien zerrenda jasoko du. Behin informazio hau jaso duela, mahai gaineko aplikazioak ez du interneteko konexiorik behar grabaketa amaitu arte.

Audio grabaketa amaitu ostean, API-ra joko da berriz eta HTTP bidez audio fitxategia eta denbora-markak web-aplikaziora bidaliko dira, "exportaketa"izendatu den prozesuan.

5. KAPITULUA

Analisia eta Diseinua

5.1 Analisia

Analisiaren zati honetan proiektuak izango dituen funtzionalitate desberdinak biltzen dira. Akta Gardena bilkuren akta, multimedia formatuan jasotzeko erreminta bat denez, lan hau errazteko funtzionalitate desberdinetan pentsatu da. Funtzionalitate hauek aukeratzeko, antzeko beste erremintetan pentsatu da, bai erreferentzia gisa izateko, bai eta besteak gainditzeko, eman diezaiokegun balio gehigarriak finkatzeko.

Akta Gardena softwareak behar dituen **funtzionalitateak** hainbat eremu teknologikotan bana ditzakegu: hizkuntza, multimedia eta web-teknologiak. Hauetaz baliatuta, bi zati nagusi izango ditu softwareak: **post-ediziorako erreminta bat akten inguruko informazioa kudeatzeko eta grabaketak egiteko erreminta.**

Proiektu honen garapenean **bi rol garatuko dira, idazkariarena eta langilearena**. Erabiltzaile guztiak udaletxe bati lotuta egongo dira eta honi lotutako eragiketak bakarrik egin ahalko dituzte. Idazkariak lotutako udaletxearen gainean botere guztia du; hau da, sistemak bere udalean egin daitezkeen funtzionalitate guztiak egitea ahalbidetzen dio. Segurtasun kontuengatik eta informazioaren egiazkotasunarengatik idazkariak bakarrik egin ditzakeen funtzionalitateak egongo dira (Dokumentuak ezabatzea, aktak ezabatzea, langile berriak sortzea, idazkari berriak sortzea etab). Langilearen rol-ak grabaketa tresnara sarbidea emango du eta post-ediziorako tresnara ere emango du.

Udaletxetako biltzarren izaeraren ondorioz garrantzitsua da biltzarraren prozesua argi uz-

tea tresnaren bizi-prozesua ondo aztertzeko. Lehenik eta behin, deialdi bat egiten da eguneko gai-zerrenda publiko utzita batzarra baino egun batzuk lehenago. Eguna iristen denean grabaketa prozesua has daiteke eta bertan multimedia edukia grabatzeaz gain, gaiak eta interbentzioak noiz hasten diren erregistratzen da. Behin grabaketa amaitu denean, honen post-ediziora pasa daiteke.

Publiko egiteaz hitz egiten dugunean, kontuan izan behar dugu, etorkizunaz hitz egiten ari garela, ez baita ataria garatuko. Akta Gardena proiektuaren ataria sortzen denan bertan publikatuko dira.

Post-edizioan denbora-markak editatzeaz gain transkribapenak zuzendu, dokumentuak gehitu, akta kontsultatu etab. ahalko dira. Esan den bezala prozesu honetan zehar, aktaren egiazkotasuna baldintza dezaketen funtzionalitateak idazkariaren eskuan bakarrik geratzen dira. Dena amaitu denean eta akta publiko egiteko moduan dagoenean denean akta publika daiteke. Hala ere onartu aurretik aktari lotutako eduki guztiak sinadura digitala jaso beharko du.

Proiektu honek izango duen irismenaren ondorioz ez da administrazio zatia egingo, **administratzailearen rol-a etorkizunean garatuko baita**. Ondorioz ez da idazkarien eta udalen sorrera erakusten. Alde publikoaren garapena ere etorkizunerako utziko den zerbait izango denez ez da bisitariaren papera islatzen. Dagoeneko aipatu den bezala, proiektuaren irismenaren ondorioz ere, ez da ataria egingo. Atariarekin hiritarrek kontsultak egiteko aukera izango duten webaz hitz egiten dugu.

5.1.1 Erabiltzaileen erregistroa

Idazkari batek nahi dituen langile guztiak sor ditzake. Horretarako, erregistro menu bat izango du eta bertatik langile berriei profila sortu ahalko die. Langile bakoitza, sortu duen idazkariaren udaletxea berari egongo da lotuta.

5.1.2 Erabiltzaileen kudeaketa

Idazkariak erabiltzaileak sortzen ditu eta, hauen kudeaketa ere idazkariaren eskuetan dago. Horretarako taula moduan erakutsiko dira erabiltzaile desberdinak eta aukerak eskuin aldean.

5.1.3 Erabiltzailearen ezarpenak

Erabiltzaile mota guztiek hainbat ezarpen izango dituzte aldagai, pasahitzak adibidez. Horregatik ezarpen hauek aldatu ahal izateko menua sortuko da.

5.1.4 Udala kudeatu

Udalaren profila erabiltzaileekin lotutako elementu bat izan arren, ez dira elementu berdinak, hortaz, udalaren profileko ezarpenak aldatzeko aukera egongo da: kontakturako datuak eta beste informazio gehigarriak

5.1.5 Aktako 'dokumentuak' ezabatu

Segurtasun kotuen ondorioz hainbat gauza ezin dira grabaketatik aldatu. Hala ere, aktari lotuta bileran zehar aurkeztu diren hainbat dokumentu egon daitezke eta hauek alda daitezke akatsak dituztelako edo beste arrazoiengatik. Aktaren edizio hau nahiko arriskutsua denez idazkariaren eskutan uzten da.

5.1.6 Bilkurak grabatu

Erabiltzaileak bileraren sesioa grabatu baino lehen hainbat ezarpen bete beharko ditu, hala nola izenburu bat, deskribapen bat, bileran jorratuko diren gaiak eta partaide garrantzitsuenak hautatu. Ezarpenak prest daudenean grabaketa prozesura pasako da beste interfaze bat erakutsiz. Interfaze honetan, aplikazioaren uneko erabiltzaileak erabakiko du noiz hasi nahi duen bileraren grabaketa eta, hau hastean, beste kontrolak aktibatuko zaizkio.

Grabaketa prozesuan aktibatuko diren kontrolak, bileran ematen diren gertaerak erregistratzeko balio dute, nagusiki, nor ari den hizketan une zehatz batean edo zein gai ari den lantzen bileran une zehatz batean. Grabaketa amaitzean, grabatu den guztia gorde egingo da eta postediziorako prest egongo da.

5.1.7 Bilkurak ikuskatu

Post-ediziorako atalaren hasieran aktak bilatzeko eta hauek ikuskatzeko menua izango da. Bilaketa honetan akta guztiak edo akta zehatz batzuk bilatzeko aukera izango da eta ondoren akta bat ikuskatu ahal izateko aukera. Bilaketako emaitza taula batean agertuko da eta taula honetan aktaren post-edizioaren egoera ageriko da.

5.1.8 Denbora-markak editatu

Post-edizioko zatian denbora-markak modu bisualean aldatu ahalko dira. Honen helburua zera da, bileran zehar jaso ahal izan ez diren hitz-txanda aldaketak zuzentzea.

Hitz-txanda bakoitza zerrenda bateko elementu gisa agertuko da eta erabiltzaileak zerrenda honetako txandak aldatu ahalko ditu edo berriak sortu momentu zehatza eta hizlaria aukeratuta. Denbora-marka aukeratzeko kronometro moduko bat ageriko zaio erabiltzaileari denbora modu errazean aukeratzeko.

5.1.9 Gai-zerrenda editatu

Nahiz eta hasiera batean gai-zerrenda post-edizioan aldatzea pentsatu zen, ondoren honen beharra zalantzan jarri zen. Bilkura baten inguruko akta jaso denean ez du zentzurik gai-zerrenda editatzeak, horregatik bilkura baten grabazioaren aurretik bakarrik aldatu ahalko da.

5.1.10 Transkribapena

Transkribapenaren arazoa multimedia akta baten arazo nagusia da. Eskuz transkribatzea izugarri luza daiteke eta lan hau modu automatikoan egitea oso puntu garrantzitsua litzateke. Horregatik, post-edizioan eta akta grabatzen den bitartean transkribapen teknologiatara joko da.

Erabiliko den transkribapen teknologia dagoeneko Iametza enpresan erabili den, Vicomtech enpresaren teknologia bat da. Transkribapenerako erreminta honek transkribapenak oso modu zehatzean egiten ditu eta puntuazio markak ere kontrolatzen ditu (komak, puntuak, puntu-komak etab). Transkribapen baten zehaztasuna oso altua ez bada, erreminta hau akatsa identifikatzeko gai da, erabiltzailea hura non gertatu den adieraziz.

Hala ere erreminta honen koxka, behar duen denboran dago, audioaren luzapenaren heren bat batez bestean irauten baitu transkribapen prozesuak, eta ondorioz, orduak irauten dituzten audioak ezin dira transkribatu zerbitzariak ezin duelako hori jasan. Horregatik, grabaketaren zehar audioa zatitzen joango da eta zati hauek transkribatzera bidaliko dira. Horrela, bilera bukatzean, transkribapena eginda egongo da eta post-edizioan egin beharko dena zera da: transkribapen prozesuak izan dituen ziurgabetasunak zuzentzea interfaze grafiko baten bitartez.

5.1.11 Firma digitala lortu

Bileran egiten den grabaketa osoak sinadura jasotzen du. Sinadura honek aktari lotutako informazioaren integritatea bermatzen du eta oso garrantzitsua da modu honetako dokumentuentzako. Sinadura digitala lortzeko beste menu bat egongo da eta hortatik sinadura prozesua gauzatu ahal izango da.

5.1.12 Hizlariak taldeka kudeatu

Bilera batean ez dute beti partaide berdinek parte hartzen, batzuetan batzordetan egiten da lan, beste batzuetan, udaletxeko bilkura bat dagoenez zinegotzi guztiak daude; horregatik, hizlariak taldeka sailkatu ahalko dira. Talde hauek sailkatzeko balio izango dute eta ez dira talde estatikoak izango. Helburua bilkuraren grabaketa egin aurretik, banan-banan hizlariak deialdian sartu ordez, automatikoki talde bateko partaide guztiak agertzea da, hizlari asko dauden bilkuretan lana errazteko.

5.2 Erabilpen-kasuak

Software aplikazio honek izan ditzakeen erabilpen-kasuen azterketa egin ondoren, aplikazioaren erabilpen kasuak eta domeinuaren eredua azaltzera pasa gaitezke.

5.2.1 Aktoreak

Proiektuan bi aktore mota izango dira hasiera batean. Dagoeneko analisian esan den bezala, etorkizunean administrazio alde bat gehituko zaio proiektuari eta bertan administra-

tzailearen rol-a agertuko da. Beraz, momentuz, idazkaria eta erabiltzailea identifikatzen dira.

- Langilea: Langileak grabaketa tresna erabili ahalko du eta ondorengo bilkurak editatzeko aukera mugatu bat izango du (lotutako dokumentu gehigarriak, denboramarkak eta besteak). Nahiz eta langilearen profila erregistroa ez den publikoa eta mundu instituzionala kontrolatutako testuinguru bat den, segurtasun kontuengatik ezingo ditu ezabaketa prozesuak egin eta aldaketa garrantzitsuenak egin.
- Idazkaria: Idazkariaren rol-ak udaletxe bateko idazkariaren papela beteko luke. Honek du botere maximoa udal zehatz bateko tresnaren gainean. Langileak egin ditzaken gauzez gain, eragiketa arriskutsuenak egin ditzake.

5.2.2 Erabilpen-kasuen diagrama

Aplikazioak izango dituen erabilpen-kasuak, erabilpen-kasuen-eredu batean bildu dira. Bertan aplikazioak izango dituen funtzionalitateak biltzen dira eta bakoitza erabili dezakeen aktoreekin lotzen da. Erabilpen kasu batzuk hainbat erabilpen-kasuren multzo izan daitezke eta diagramaren txukuntasuna bermatzeko multzokatu dira. Esaterako, kudeaketa bat inplikatzen duten erabilpen kasuek sorrera, edizioa, ezabaketa eta bilaketa inplikatzen dute. Bestalde, idazkariak eta langileak osatzen duten loturak herentzia esan nahi du; hau da, langileak egin dezaken oro egin dezake idazkariak.

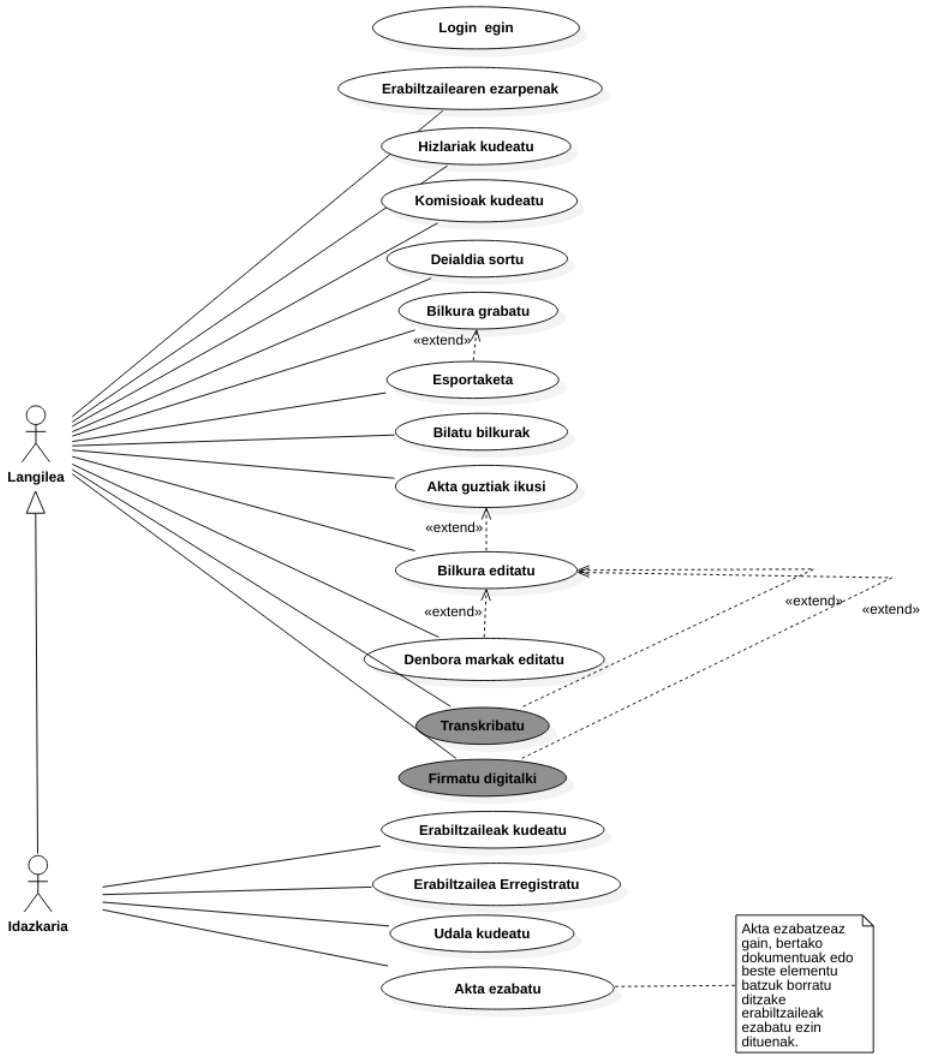
5.2.3 Gertaera-fluxuak

Gertaeren fluxuetan erabilpen-kasuek daramaten prozesua deskribatzen da. Gertaeren fluxu interesgarrienak gehitzea erabaki da, kudeaketako erabilpen-kasuak berdinak baitira (ikuskatu, editatu, ezabatu eta sortu). Bestalde, erabilpen-kasu gehiago daude.

Login Egin

Aktoreak: Langilea eta idazkaria

Azalpena: Erabiltzaile desberdinek saio bat hasteko eta kautotuta egoteko modua izango da. Erabiltzaile bakoitzeko posta elektroniko (eposta aurrerantzean) bat egongo da eta kontu honek baimen desberdinak izango ditu lotuta, nahiz eta modu berean saioa ireki.



5.1 Irudia: Erabilpen-kasuen diagrama.

Saioa irekitzeko erabiltzailearen eposta eta pasahitza eskatuko da. Erabilpen kasu hau, mahai gaineko aplikazioan eta web-aplikazioan berdina denez, atal honetan ez da desberdinduko.

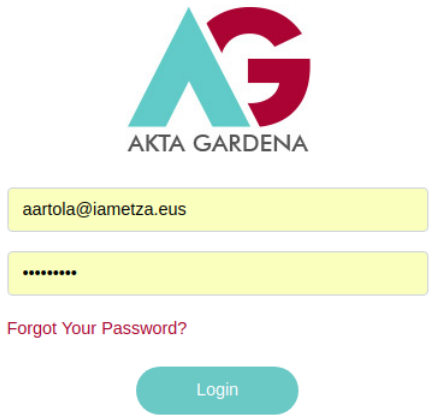
Gertaera-fluxu normala:

- Sistemak erabiltzailearen eposta eta pasahitza eskatzen ditu.
- Erabiltzaileak datuak sartzen ditu.
- Sistemak datu-basean eposta eta pasahitzaren konbinaketa bilatzen du. Aukera bakarra topatu duenean konbinaketa egokia dela esan nahiko du eta aurrera egingo du.

Gertaera-fluxu alternatiboa:

- Datu-basean konbinaketarik ez bada topatu, orria eguneratuko da eta mezu bat agertuko da kredentzialak gaizki idatzi direla dioena.

Interfazearen prototipoa:



5.2 Irudia: Login pantailaren prototipoa.

Bilkura grabatu

Aktoreak: Langilea eta idazkaria

Azalpena: Multimedia akta grabatzeko tresna bera da. Bertan audioaren grabaketaz gain denbora-markaka jartzeko botoiak egongo dira. Denbora-markak bilkurako gertaera bat erregistratzen dute, hizlari baten txanda hasiera eguneko ordenako gai batez hitz egiten hastea.

Gertaera-fluxu normala:

- Sistemak grabaketaren interfazea erakutsiko du bere kontrol desberdinekin. Bertan, grabaketako botoia egongo da gaituta.
- Erabiltzaileak grabatzeko botoia sakatuko du.
- Sistema, audio seinalea jasotzen hasiko da eta denbora-marketako botoiak gaituko ditu.
- Erabiltzaileak, gai berri bat hasten denean, gaiari dagokion botoia sakatuko du, eta hizlari zehatz baten botoia sakatuko du hizketan hasten den unean.
- Grabaketa bukatzean erabiltzaileak geratzeko botoiari sakatuko dio eta gordeta geratuko da.

Interfazearen prototipoa:

5.3 Irudia: Bilkura grabaketaren pantailaren prototipoa.

Hizlaria sortu

Aktoreak: Langilea eta idazkaria

Azalpena: Biltzar batean parte hartuko duen hizlari orok sisteman erregistratuta egon behar du. Hizlari bera hainbat aktetan ageri daiteke eta aldi berean udaletxe bakar bati lotuta egongo da.

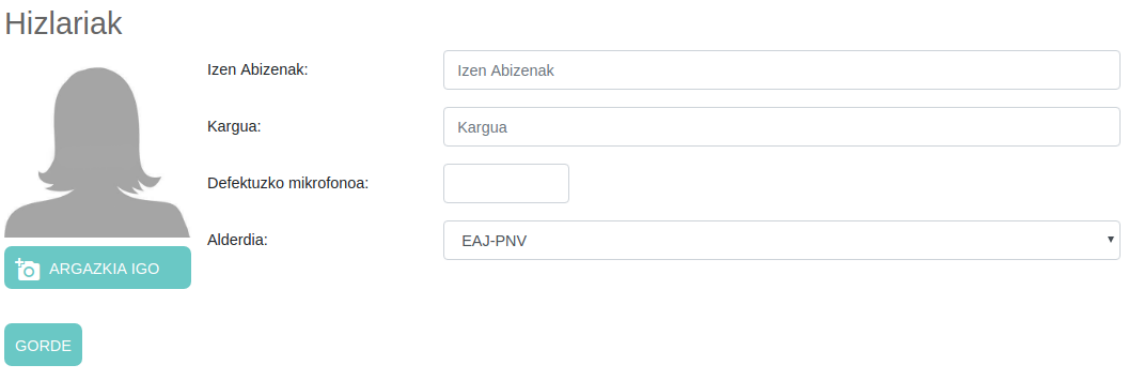
Gertaera-fluxu normala:

- Sistemak hizlari bat sortzeko menua erakutsiko dio erabiltzaileari.
- Erabiltzaileak izen-abizenak, hizlariaren kargua eta alderdia idatziko ditu derrigorrez.
- Erabiltzaileak hizlariaren argazkia igoko du.
- Hizlaria gordetzeko “Berria” sakatuko du.

Gertaera-fluxu alternatiboa:

- Erabiltzaileak ez badu hizlariaren argazkirik igotzen bere alderdiaren logoa emango zaio.
- Hizlari bat ez bada alderdi baten parte, ’alderdia’ zatian ’Beste bat’ aukeratuko da.

Interfazearen prototipoa:



5.4 Irudia: Hizlaria sortzeko pantailaren prototipoa.

Denbora-markak editatu

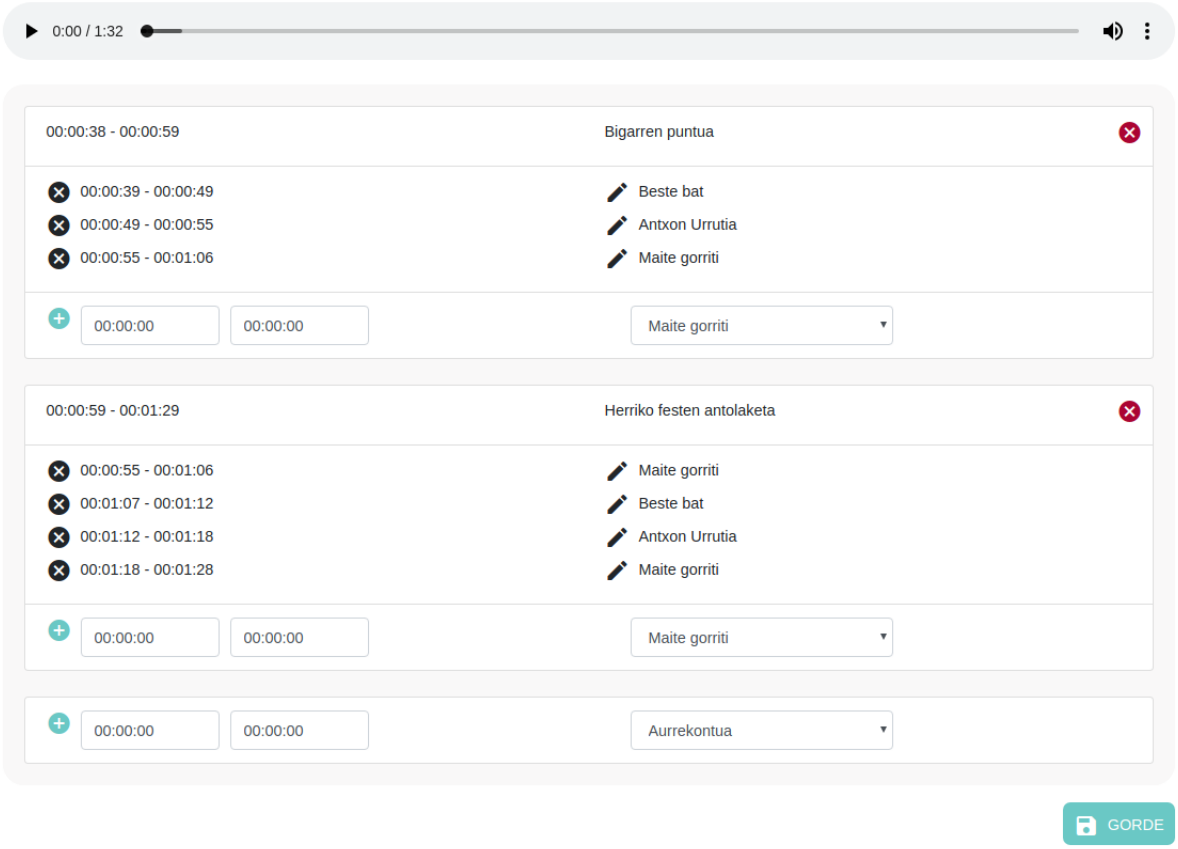
Aktoreak: Langilea eta idazkaria

Azalpena Nahiz eta denbora-markak grabaketaren zehar hartzen diren, ondoren marka hauek editatzeko aukera izan behar da. Denbora-markak editatu ahal izateko bi denbora-marka mota kudea daitezke: hizlarien txanda eta gaien markak.

Gertaera-fluxu normala:

- Sistemak, grabaketaren zehar hartu diren markak erakutsiko dizkio erabiltzaileari menuan.
- Erabiltzaileak aktako audioa entzun dezake, nahi duen unean, nahi duen alditan.
- Denbora-marka bat gehitzean edo editatzean, hasierako eta bukaerako denborak idatziko ditu eta hizlaria edo gaia aukeratuko du.
- Behar den guztietan errepikatuko du prozesua.
- Amaitzean gorde botoiari sakatuko dio.

Interfazearen prototipoa:



5.5 Irudia: Denbora-markak editatzeko pantailaren prototipoa.

Deialdia sortu

Aktoreak: Langilea eta idazkaria

Azalpena: Bilkura bat gertatu aurretik, sisteman, bilkura horrek existitu behar du, eta ez badago, sortu egin behar da. Bilkura gertatu aurretik bilkuraren deialdia ere egin behar da, deialdia bakarra izanik. Hori horrela, bileraren datuen artean izenburua eta deskribapenaz gain, gaiak, hizlariak eta dokumentuak sartuko dira. Bilkura bukatzean ere erabiltzaileak dokumentuak gehitu ahalko ditu.

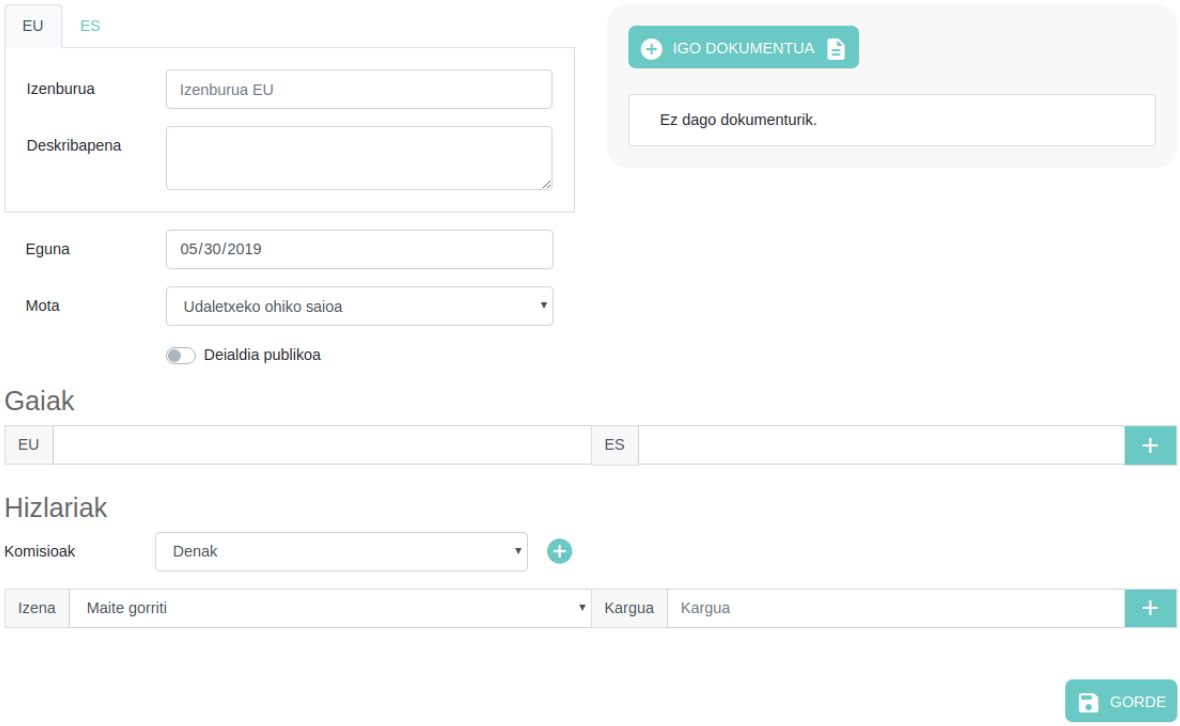
Gertaera-fluxu normala:

- Sistemak formulario bat erakutsiko dio erabiltzaileari.
- Erabiltzaileak deialdiko datuak sartuko ditu bilkura fisikoki gertatu baino egun batzuk lehenago. Izenburua, deskribapena eta bilkuraren eguna bete beharko ditu. On-

doren bilkuraren gai-zerrenda eta hizlarien zerrenda bete beharko ditu. Gainera, aktari lotutako dokumenturen bat atxikitzeko aukera izando du.

- Erabiltzaileak deialdirako behar diren datuak sartu ostean 'deialdia publikoa' izeneko botoia sakatuko du publikatzeko.
- Sistemak deialdia argitaratuko du.

Interfazearen prototipoa:



5.6 Irudia: Deialdia sortzeko pantailaren prototipoa.

Esportaketa

Aktoreak: Langilea eta idazkaria

Azalpena: Bilkura baten grabaketa bukatu denean, mahai gaineko aplikaziotik web-aplikaziora, grabaketarekin lotutako datu eta fitxategi guztiak pasatzen dira.

Gertaera-fluxu normala:

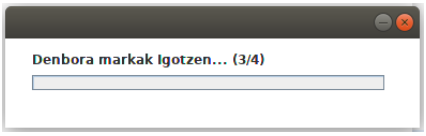
- Erabiltzaileak grabaketa bat amaitzen du.

- Mahai gaineko Sistemak esportaketa leihoa erakusten dio eta ez dio lihoa ixten uzten erabiltzaileari.
- Mahai gaineko Sistemak, audio grabaketatik sortu berri den fitxategia kodetzen du.
- Mahai gaineko Sistemak audio kodetuaren laburpena sortzen du eta ondoren zifratu egiten du.
- Mahai gaineko sistemak denbora-markak web-aplikaziora bidaltzen ditu.
- Web-aplikazioak denbora-markak gordetzen ditu
- Mahai gaineko Sistemak audio fitxategia web-aplikaziora bidaltzen du.
- Web-aplikazioak audio fitxategiaren laburpena eta fitxategia bera gordetzen ditu.
- Esportaketa leihoa ixten du sistemak.

Gertaera-fluxu alternatiboa:

- Internet konexioa prozesuan joaten bada, edo ordenagailua edo programa ustekabean ixten bada, sistemak berriz esportatzeko aukera emango dio erabiltzaileari.
- Sistemak grabaketako audioa berriz kodetu eta laburpena sortuko du.
- Sistemak grabaketa osteko laburpena, sortu berri duen laburpenarekin alderatuko du eta berdina bada jarraitzen utziko dio, bestela moztu egingo du gertaera.
- Web-aplikazioak denbora-markak gordetzen ditu
- Mahai gaineko Sistemak audio fitxategia web-aplikaziora bidaltzen du.
- Web-aplikazioak audio fitxategiaren laburpena eta fitxategia bera gordetzen ditu.
- Esportaketa leihoa ixten du sistemak.

Interfazearen prototipoa:



5.7 Irudia: Esportatu pantailaren prototipoa.

5.3 Datu-basearen diseinua

Aplikazioaren datu-basea beharrezkoa da akta guztiei lotutako informazioa gordetzeko. Taulen egituran sistemaren funtzionamendu egokirako taula gehigarri batzuk ageri dira, baina bestetik, akta inguratzen duten elementuak islatzeko balio duten taulak daude.

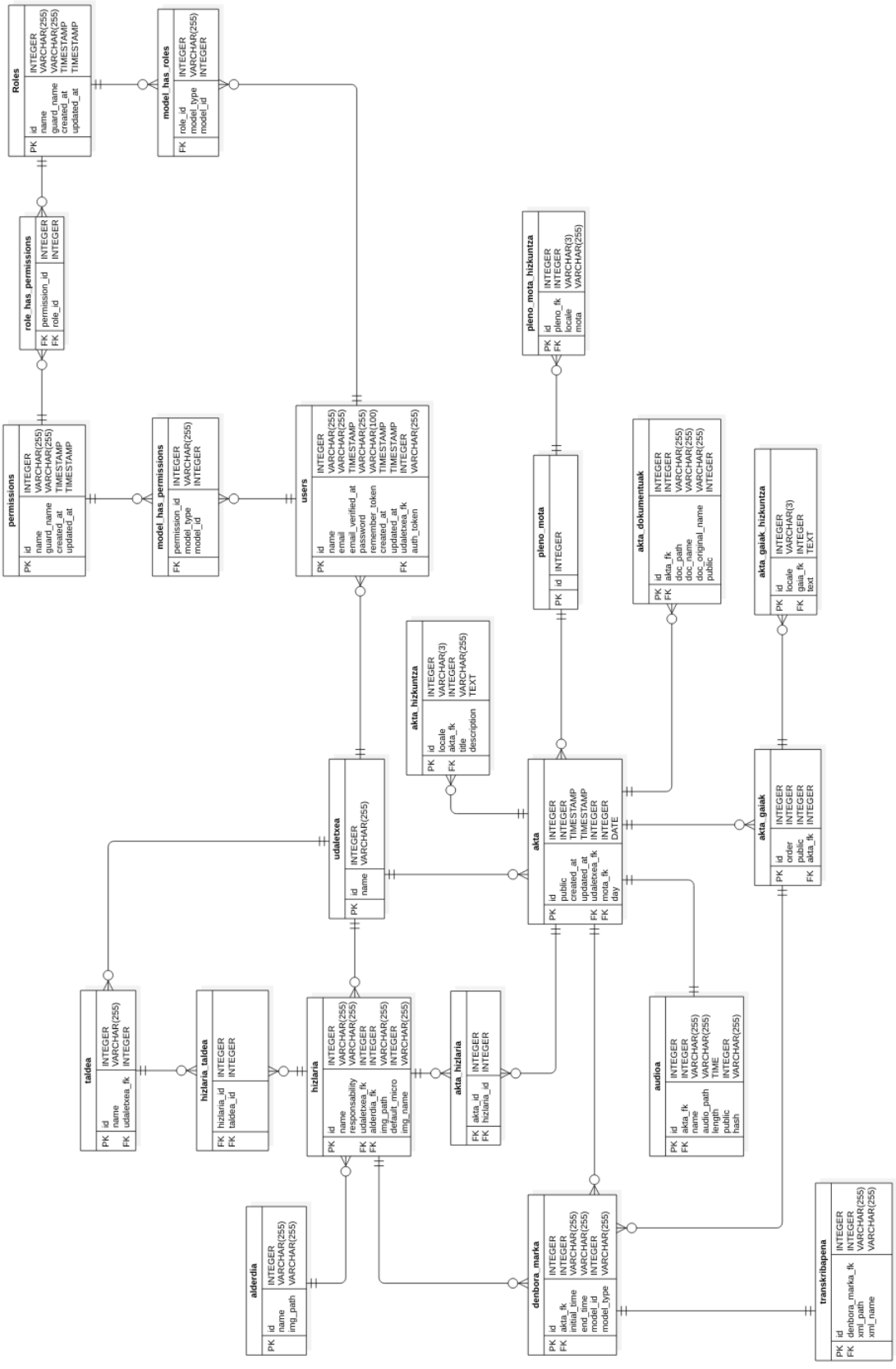
Proiektu honetako datu-base osoa hasieratik diseinatu da MySQLn. Akta Gardenaren hasierako bertsioak ez baitzuen ezertarako datu-base baten beharrik. 5.8 irudian ikusten duguna entitate-erlazio-diagrama bat da, taula desberdinen artean existitzen den erlazioa adierazteko.

Jarraian, diagraman ageri diren taulen azalpen bat emango da atalka. Datu-basearen egitura ongi ulertzeko 3 zatitan banatu dezakegu bera: erabiltzailearekin zerikusia duena, udalarekin zerikusia duena eta aktekin zerikusia duena.

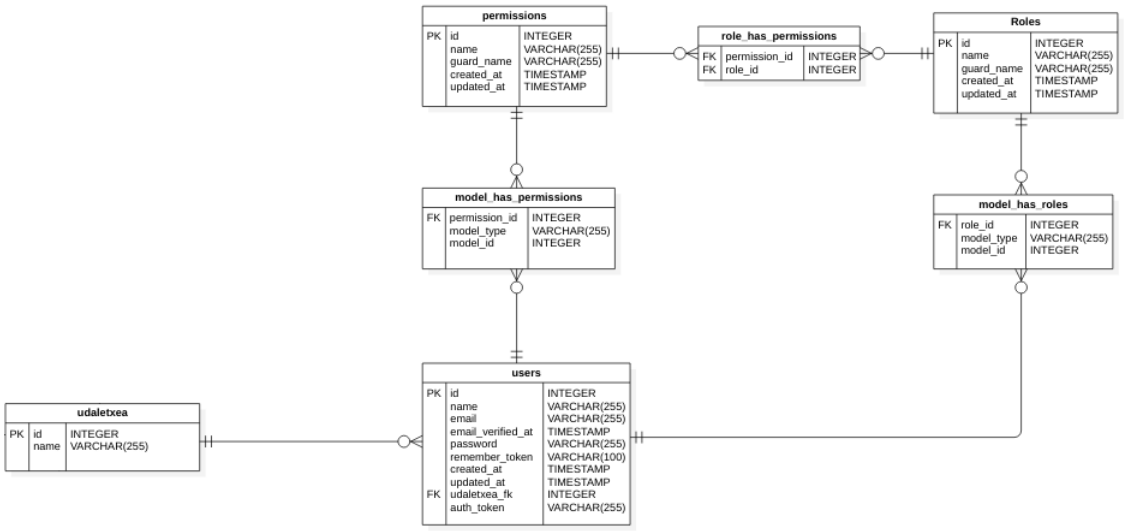
5.3.1 Erabiltzailearen taulak

Erabiltzailearen datuak gordetzeko eta bakoitzak izango duen rola eta izango dituen baimenak definitzeko 6 taula ditugu.

- users: Erabiltzaile guztien datuak gordetzeko balio duen taula da hau. Erabiltzaile bakoitza udaletxe batekin lotuta egongo da (udaletxea_fk) eta honen inguruko operazioak bakarrik egin ahalko ditu.
- permissions: Aplikazioan ekintza zehatzak egiteko baimenak definitzen dituen taula da hau.
- roles: Sistemaren rol desberdinak definitzeko balio duen taula da.
- model_has_permissions: hainbat erabiltzaile hainbat baimenekin erlazionatzeko erabiltzen den taula da hau.



5.8 Irudia: Datu-base osoaren entitate-diagrama.



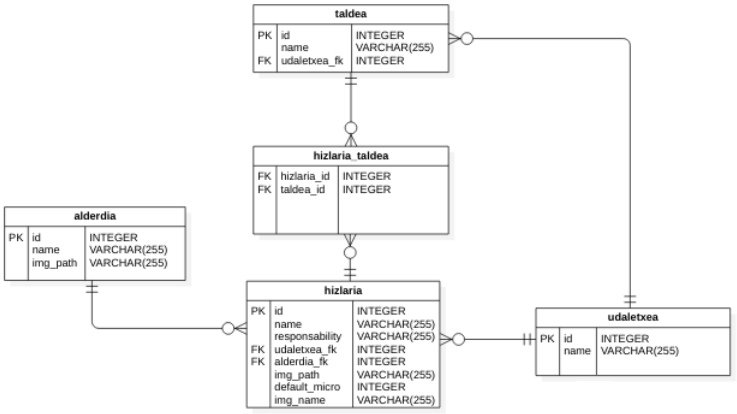
5.9 Irudia: Erabiltzaileari lotutako datu-basearen zatia.

- role_has_permissions: hainbat baimen hainbat rol-ekin erlazionatzeko erabiltzen den taula da hau.
- model_has_roles: erabiltzaile bakoitzak izango dituen rol-ak definitzeko balio duen taula.

5.3.2 Udalaren taulak

Udaletxe bakoitzak lotutako informazioa izango du nahiko estatikoa izango dena. Udalari lotutako informazio hau 5 taula artean banatzen da.

- udaletxea: Udaletxea bera errepresentatzen duen taula da hau. Sistema hau erabiltzen duen udaletxe bakoitzak bere sarrera izango du taula honetan.
- alderdia: Udaletxean ordezkaritza duten alderdiak biltzen dituen taula. Hizlari bakoitzari alderdi bat lotuko zaio.
- taldea: Hizlari desberdinak multzokatzeko balio duen taula da hau. Talde bat (berez, lan-taldea, komisio, edo dena delakoa adierazteko entitatea) udaletxe bati lotuta egongo da (udaletxea_fk) eta udaletxe batek hainbat talde izango ditu.
- hizlaria: biltzarretan parte hartzen duten hizlariak gordetzeko balio du honek. Normalean alderdi zehatz bateko zinegotziak izango dira. Gonbidaturik balego, taula honetan gordeko lirateke beste hizlariak bezala.



5.10 Irudia: Udalari loturako zatia datu-basean.

- Hizlaria_taldea: hainbat hizlari hainbat talderekin lotzeko balio duen taula.

5.3.3 Akten taulak

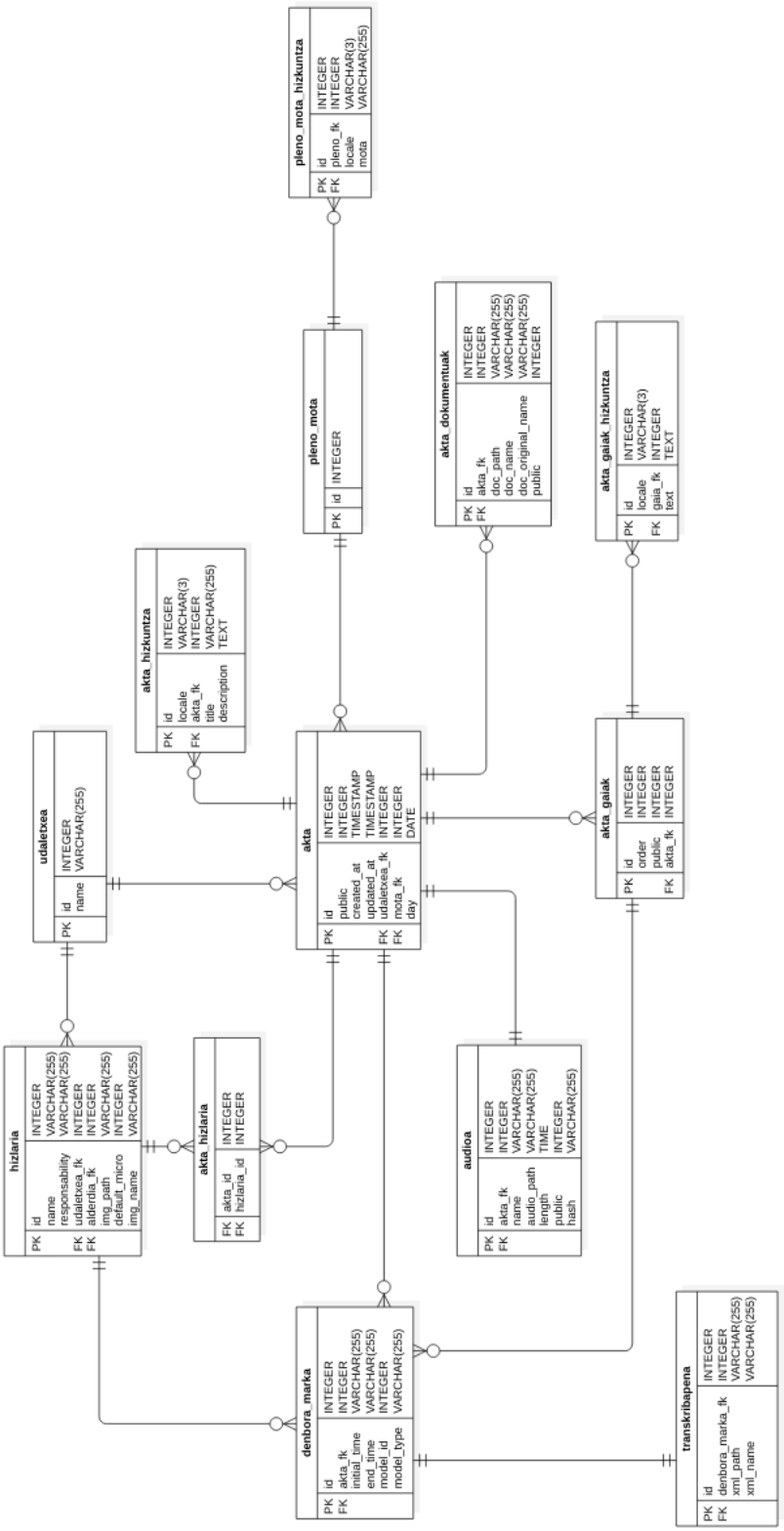
Biltzar bateko akta bat irudikatzeko balio duen taula multzoa da hau. Guztira 8 taulak osatzen dute akta bakar bat.

- akta: Akta baten informazio orokorra gordeko duen taula da hau. Gainera akta bati lotutako informazioaren muina izango da. Akta bakoitza udaletxe bati lotuta egongo da (udaletxea_fk) eta udaletxe batek hainbat akta izango ditu. Akta bakoitzak public izeneko flag bat du, kasu honetan deialdia publikoa den edo ez adierazten du.

- akta_hizkuntza: Internazionalizaziorako derrigorrezkoa da taula hau, hemen gordeko baitira itzulitako testuak.

- audioa: Akta batek audio bat lotuta izango du eta audio honi buruzko informazioa gordetzeko dago taula hau. Bertan audio fitxategiaren kokapena gordetzeaz gain (audio_path, name) lotutako aktaren erreferentzia ere gordeko da (akta_fk). Audioak public flag bat du, kasu honetan audioa publikoa egin edo ez adierazteko balio du.

- akta_dokumentuak: akta_dokumentuak taularen zeregina aktari lotutako dokumentu desberdinei buruzko informazioa gordetzea da. public flag-ak dokumentu hau publikoa den edo ez definitzen du.



5.11 Irudia: Akten zatia datu-basean.

- akta_gaiak: Akta batek izango dituen gaiak gordeko dira taula honetan. Taula honetan erreferentzia egingo dion aktaren loturaz gain (akta_fk) gaien ordena ere gordeko da (order).

- akta__gaiak__hizkuntza: Taula hau beharrezkoa da internazionalizaziorako. Bertan gaiaren testua hizkuntza desberdinetan gordeko da. Horretarako, taulak testuaz gain (text) testua bera zein hizkuntzatan idatzita dagoen (locale) eta zein gairi lotuta dagoen gordeko du (gaia_fk).

- akta_hizlaria: Aktaren eta hizlariaren arteko erlazioa ezartzen du. Hau da, hainbat hizlari hainbat aktetan ageri direla adierazteko balio du.

- denbora_marka: Biltzar baten zehar gertatzen diren interbentzioak gordetzeko denbora-markak erabiltzen dira. Hau da, una zehatz batean gai bat hasi da edo beste une batean hizlari bat hasi da. Taula honetan hizlariaren edo gaiaren erreferentzia gordeko da (hizlaria_fk edo gaia_fk), eta noski erreferentzia egingo dion aktaren erreferentzia (akta_fk).

- transkribapena: transkribapenak modu egokian sailkatu ahal izateko, hizlarien interbentzioen arabera sailkatzea erabaki da, modu honetan transkribapen bat denbora-marka bati lotuta egongo da (denbora_marka_fk). Taula honek transkribapena gordeko duen XML fitxategiari erreferentzia egingo dio.

- pleno_mota: Aktaren izaera definitzen duen akta mota (Ohiko bilkura, ezohikoa, gobernu batzarrak eta beste motak) gordetzen duen taula.

- pleno_mota_hizkuntza: Aktaren mota testua internazionalizatzeko balio duen taula.

5.4 Sekuentzia-diagramak

Atal honetan, kodeak jarraituko duen bidea eta egingo diren deiak azalduko dira. Gertaera fluxuetan bezala, erabilpen kasu garrantzitsuenak soilik jorratuko dira. Gainera, irakurgarritasunean laguntzeko, sekuentzia-diagramak kapituluaren amaieran egongo dira, atal honetan errefentzia egingo zaie soilik.

5.4.1 Login egin (Mahai gaineko aplikazioan)

Nahiz eta kautotzea web-aplikazioan eta mahai gainekoan egin daitekeen, web-aplikazioko kautotze sistema Laravel-ek azpitik kudeatzen duenez, mahai gaineko login-a aztertuko dugu.

5.12 irudian ageri den sekuentzia-diagramaren alde interesgarria, mahai gaineko aplikazioaren eta web-aplikazioan dagoen API-aren arteko komunikazioa ikustea da. Hau da, informazioa *IFacadeDataAccess*-en (mahai gaineko aplikazioan) eta *WebServiceController*-en (API-a) artean nola mugituko den ikustea; hots, funtzio dei bat izango balitz bezala.

5.4.2 Bilkura grabatu

Bilkura bat grabatu behar denean, egin behar den lana ez da audio grabaketa soil bat, denbora-markak ere erregistratu behar dira. Horregatik, prozesua hainbat hari desberdinetan banatuta egongo da, bakoitzak zeregin desberdinarekin.

5.13 irudiko diagramak erabiltzailearen 4 ekintza biltzen ditu: grabaketaren hasiera, hizlari baten txanda, gai baten txanda eta grabaketaren amaiera biltzen dira. Hizlari baten txanda eta gai baten txanda hainbat aldiz errepikatuko dira grabaketa prozesuan zehar.

Aipatu den bezala, prozesu hau hainbat haritan banatuko da. Hari nagusiak programaren exekuziarekin jarraitzen du; hau da, interfaze grafikoa (*frmGrabatu* klasea) kontrolatzen, grabaketa geratu behar bada esaterako edo denbora-marka bat erregistratzeko. Bestalde, audioa grabatzeaz arduratuko den haria legoke, *JavaSoundRecorder*-ek eta Javaren liburutegi natiboak, *AudioSystem* klaseak, osatuta.

Kontrolatzailea klasera egingo diren deiak sinkronoak izango dira. Horrela, interfaze grafikotik datorren informazioaz gain, beste informazio-iturriak jasotzera irekita geratuko da (etorkizunean mikrofonia-sistema digitalek ematen duten informazioa esaterako).

5.4.3 Hizlaria sortu

5.14 irudian ikusten dugunez, deiek Laravel-ek ezartzen duten MVC patroia jarraitzen dute. Erabiltzaileak bistarekin elkar eragingo du, bistak ostean kontrolatzaileari bidaliko dio informazioa. Kontrolatzaileak bistatik jaso duen informazioaz baliatuta *Hizlaria* ereduko

objektu berri bat sortuko du eta amaitzean beste bista bat erakutsiko dio eredu berriak bistaratzeko.

Hizlari bati profila sortzean, irudi bat atxikitzeko aukera izango dugunez, Laravel-ek eskaintzen duen *Storage* fatxada erabiliko dugu.

5.4.4 Denbora-markak editatu

Denbora-markak editatzen ditugunean, nabigatzailean exekutatuko denez tresna, 5.15 irudiko diagraman, nabigatzailearen JavaScript motoreak egingo duena ere ageri da.

Diagrama laburtzeko, 3 edizio aukera bildu dira: gaia gehitu, hizlaria gehitu eta hizlaria ezabatu. Hala ere, beste guztiek bide bera jarraituko dute:

- Lehenengo denbora-markak gordeko dituen aldagaia legoke, JSON fitxategiaren itsurarekin.
- Egin nahi den edizio mota bakoitzeko, JSON aldagai hau editatuko duen funtzio bat (denbora-marka bat gehitu, denbora aldatu, ezabatu edo ebste edozein).
- JSON aldagai hau aldatu dugula, nabigatzaileko web-aren itsura aldatuko dugu, *draw()* izeneko funtzio bati deituta.

Edizio guztiak amaitu ditugunean, JSON aldagaia, JSON formatua duen testu batera serializatuko dugu *serialize()* funtzioari deituta. Azkenik, JSON hau bidaliko dugu eta hau tratatu ostean denbora-markak gordeko dira datu-basean.

5.4.5 Deialdia sortu

5.16 irudian bilkura baten deialdia sortzeko jarraituko den prozesua ageri da. Akta bat sortzen den unean hainbat gauza sortuko dira:

- Lehenik eta behin, eta garrantzitsuena, akta bera izango da.
- Akta sortu dugula, Laravel-en *Storage* fatxadarekin lotutako dokumentuak gordeko ditugu.
- Dokumentuak gorde ditugunean, hauek helbideratzeko, bakoitzarentzako *Dokumentua* eredu bat sortuko dugu.

- Bilkuran parte hartuko duten hizlariak lotuko dizkiogu sortu berri dugun deialdiari.
- Gai guztiak sortuko ditugu eta sortu berri dugun deialdiari lotuko dizkiogu.

5.4.6 Esportatu

5.17 irudeiko diagramaren ulergarritasunean laguntzeko eta konplexutasuna txikitzeko, hainbat gertaera laburtu egin dira. Kodeketak lau pauso izango ditu: Audioaren kodeketa, audioaren hashing-a, denbora-marken igoera eta audioaren igoera.

Sekuentzia-diagrama honetan ere mahai gaineko aplikazioaren eta web-aplikazioan dagoen API-aren arteko komunikazioa ikus dezakegu. Kasu honetan, informazioa bidali egingo zaio API-ari; hain zuzen ere, audioa eta denbora-markak bidaliko zaizkio.

5.5 Domeinuaren eredua

Mahai gaineko aplikazioak eta web-aplikazioak ez dutenez domeinu berdina, atal honetan bi zatitan banatuko dugu: mahai gaineko aplikazioaren domeinuaren eredua eta web-aplikazioaren domeinuaren eredua.

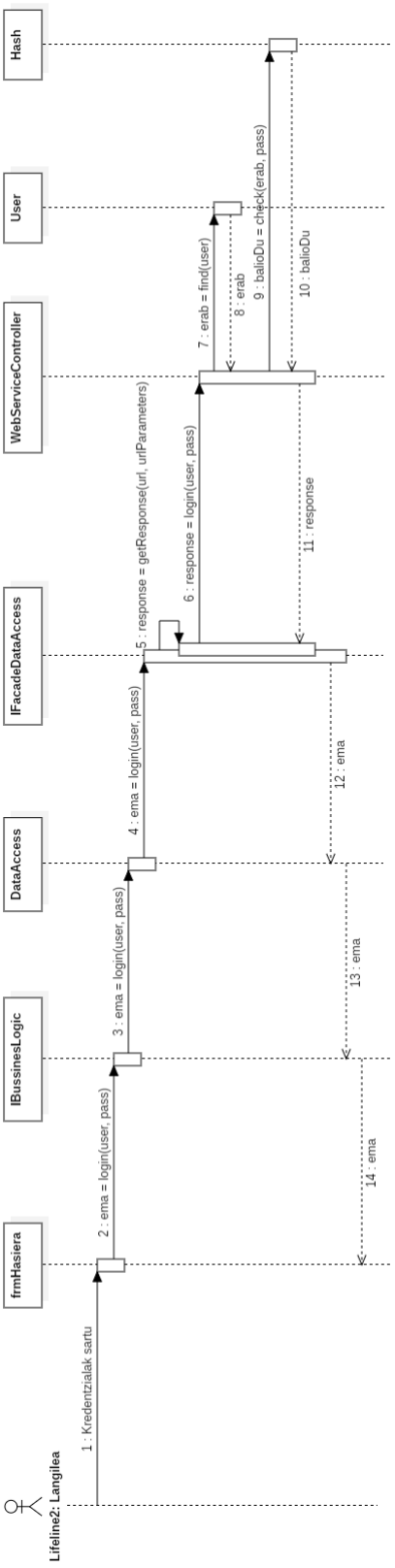
5.5.1 Mahai gaineko aplikazioaren domeinua

Mahai gaineko aplikazioak ez duenez informazioaren kudeaketa handia egiten, domeinu nahiko mugatua du.

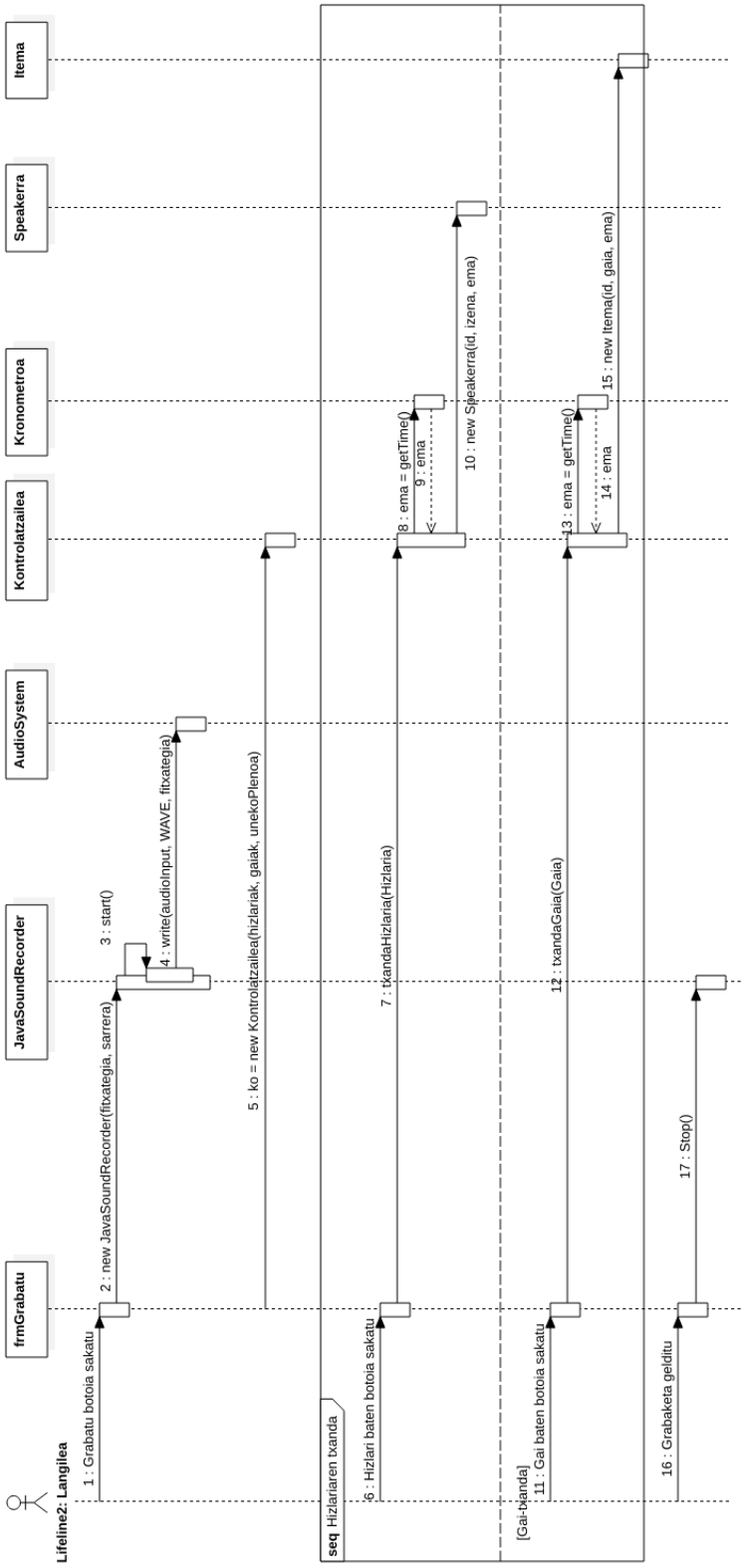
5.18 irudiko diagraman (hau da, mahai gaineko aplikazioaren domeinuan) 6 klase ageri dira. Mahai gaineko aplikazioan informazio-jarioa, arkitekturako atalean ikusi den bezala (ikusi 4 kapitulua) goranzkoa eta beheranzkoa da; hau da, web-aplikaziotik mahai gainekora eta alderantziz. Domeinuaren eredu honek, informazio-jario hori irudikatzen du.

Alde batetik, mahai gaineko aplikazioak informazioa jasotzen duenean sortzen duen domeinua dugu. Domeinuaren zati hau hiru klasez osatuta legoke:

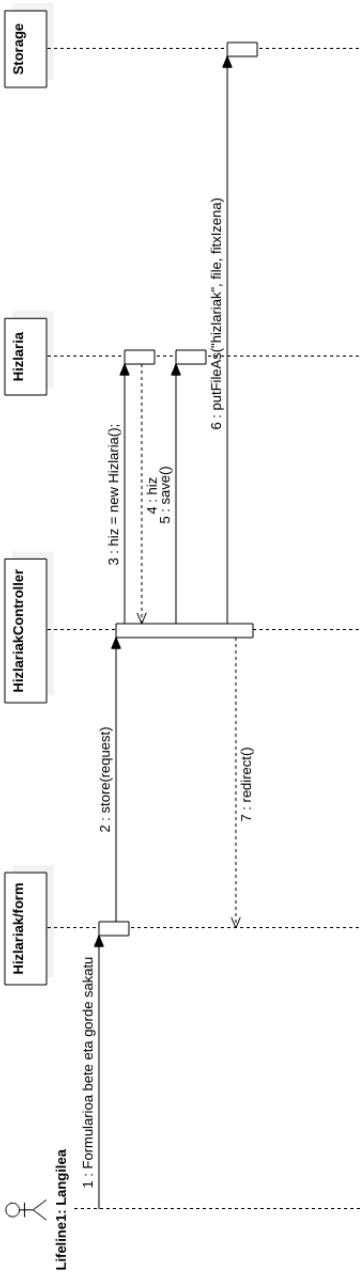
- **Plenoa** klaseak bilkura bat irudikatzen du. Bilkura honek izenburu, deskribapen eta identifikaziorako gako bat lukete.



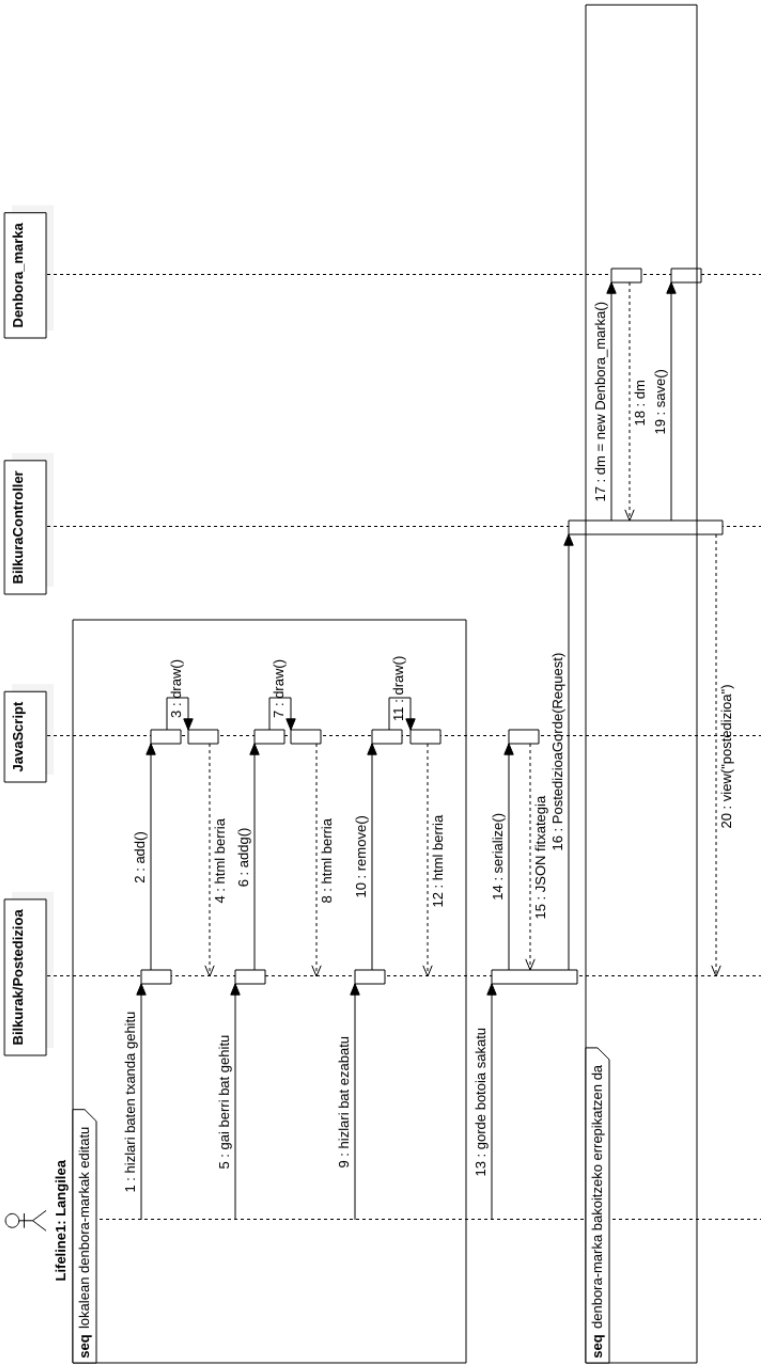
5.12 Irudia: "Login egin"erabilpen kasuaren sekuentzia diagrama mahai gaineko apliakzioan.



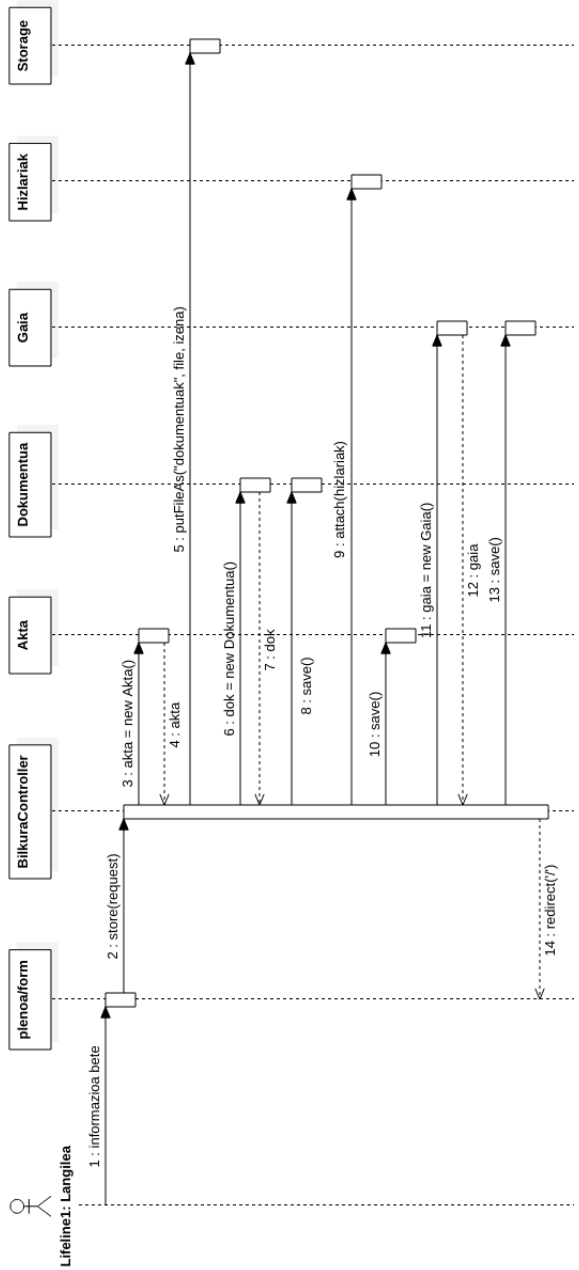
5.13 Irudia: "Bilkura grabatu"erabilpen kasuaren sekuentzia diagrama.



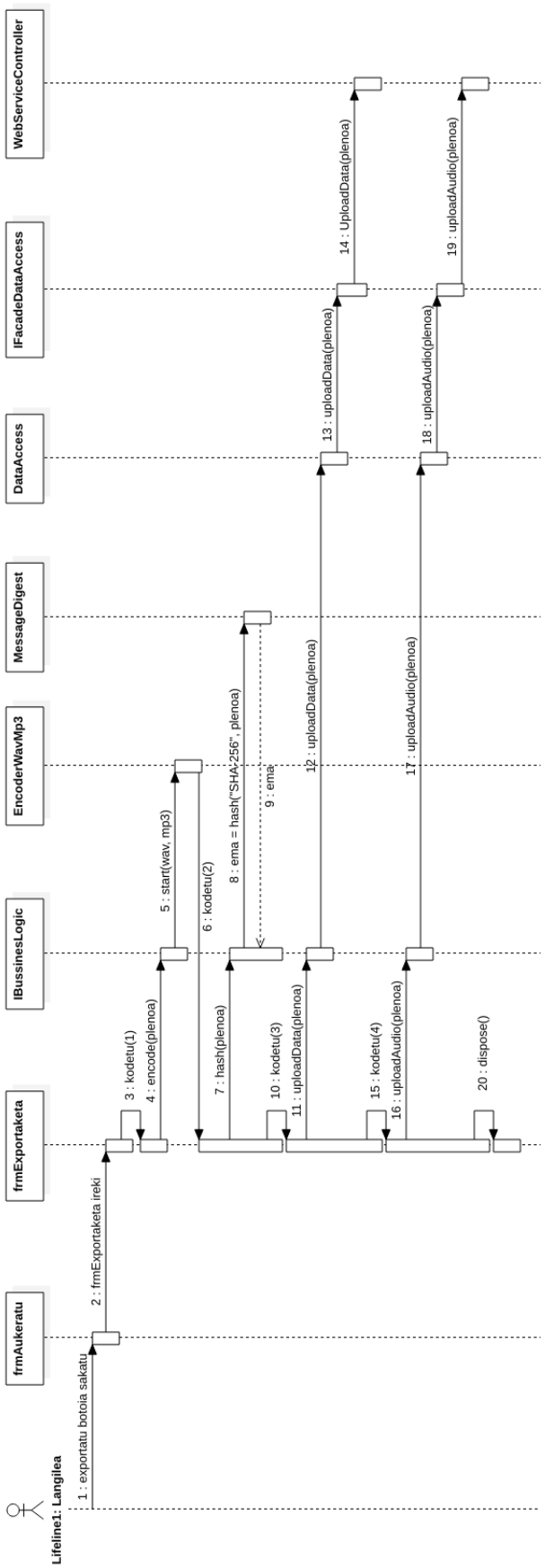
5.14 Irudia: "Hizlaria sortu"erabilpen kasuaren sekuentzia diagrama.



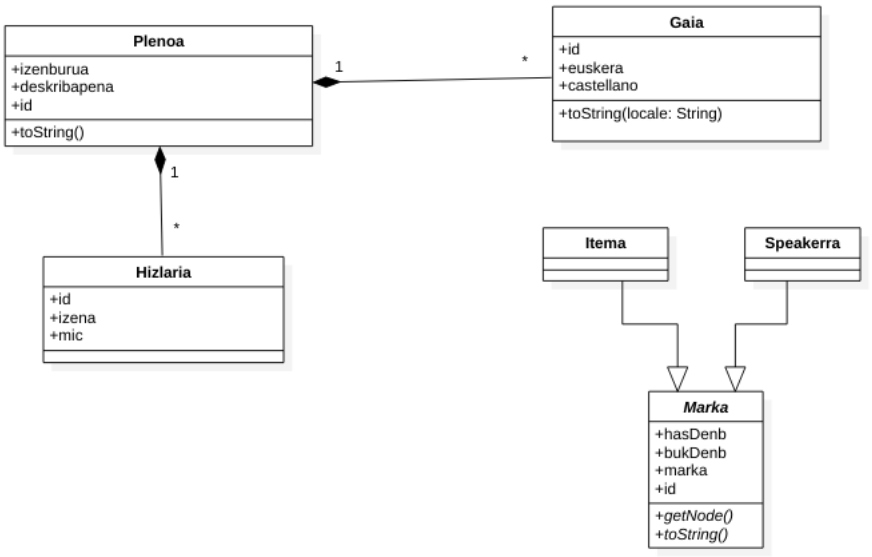
5.15 Irudia: "Denbora-markak editatu"erabilpen kasuaren sekuentzia diagrama.



5.16 Irudia: "Deialdia sortu"erabilpen kasuaren sekuentzia diagrama.



5.17 Irudia: "Esportatu"erabilpen kasuaren sekuentzia diagrama.



5.18 Irudia: Mahai gaineko aplikazioaren domeinuaren eredua.

- **Gaia** klaseak bilkura baten gai bat irudikatzen du. Bilkura bati lotutako gaiak euskaraz eta gazteleraz leudeke. Gainera, gaia identifikatuko duen gako bat gordeko da.
- **Hizlaria** klaseak bilkura batean parte hartuko duen hizlari bat irudikatzen du. Partaide honek identifikazio gako bat izateaz gain, izen bat eta defektuzko mikrofono bat izango du.

Bestalde, mahai gaineko aplikazioak informazioa bidali behar duenean, bilkura batean egin den audio grabaketaz gain, denbora-markak ere bidali behar ditu. Grabaketaren zehar sortu diren denbora-markak irudikatzeko, hiru klase leudeke:

- **Marka** klase abstraktuak denbora-marka bat irudikatzen du. Denbora-marka honek hasiera bat eta bukaera bat izateaz gain, identifikatuko duen gako bat eta lotutako gertaeraren marka bat izango du. Klase abstraktu honek funtzio nagusi bat izango luke, *getNode()* izenekoa, denbora-markak bilduko dituen fitxategiaren serializazio[1] prozesuan laguntzeko.
- **Itema** klaseak gai bati lotutako denbora-marka irudikatuko du.

[1]Serializazioaz hitz egitean, datuak ulertzeko modu batetik beste batera bihurtzeaz hitz egiten dugu. Esaterako, software batek aplikazio barnean duen informazioa, fitxategi batean gorde nahi badu, XML fitxategi batean serializa dezake.

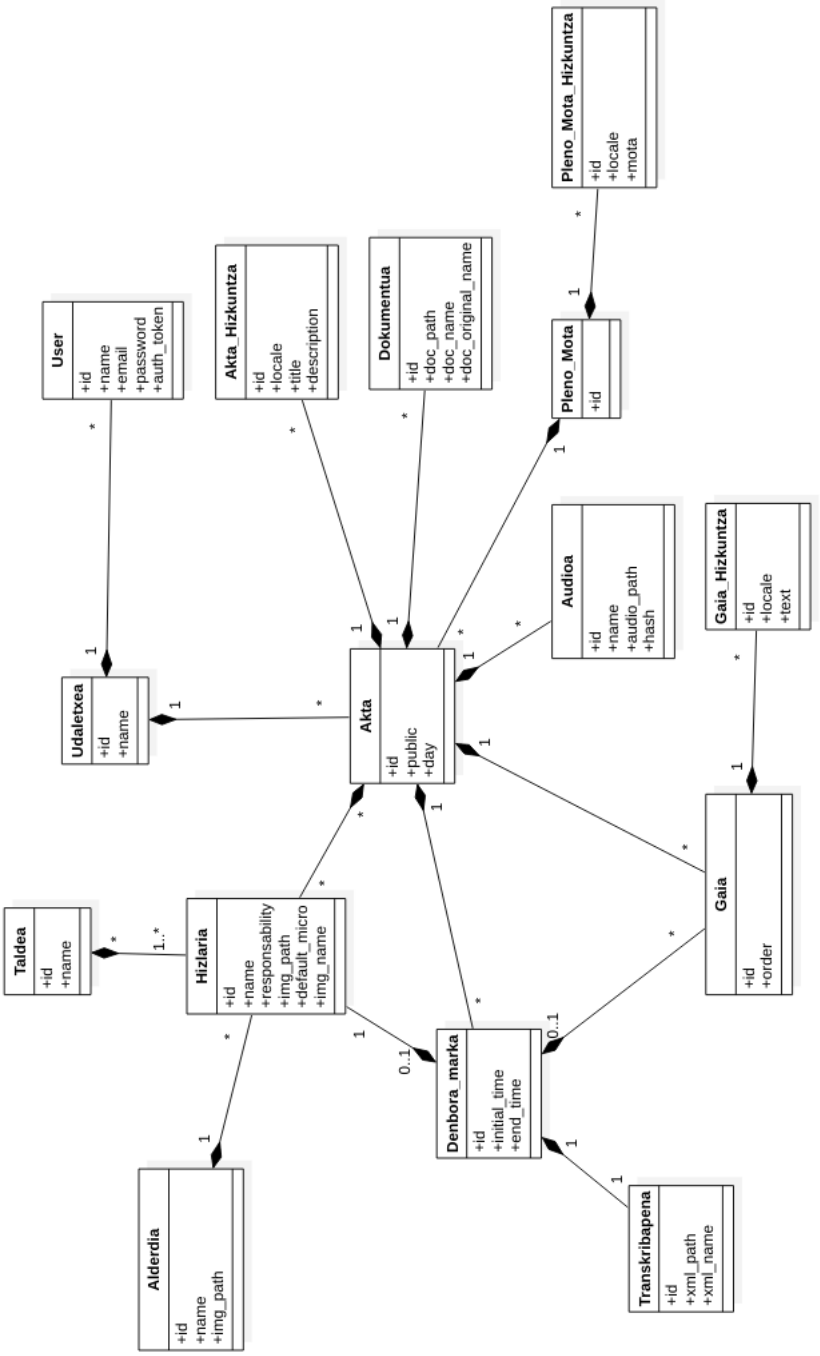
- **Speakerra** klaseak hizlari bati lotutako denbora-marka irudikatuko du.

5.5.2 Web-aplikazioaren domeinua

Web-aplikazioko domeinua aztertzen dugunean, Laravel-ek MVC patroia erabiltzea derrigortzen digunez, ereduen arteko erlazioa aztertuko dugu. Domeinuaren eredu hau, datu-basearen diseinuan oinarritu da; hala ere, Laravel-en azpiegitura (inposatzen dituen herentziak esaterako) alde batera utzi da domeinuaren diseinua sortzeko.

5.19 irudiko domeinuaren eredua diseinatzeko unean datu-basearen diseinuan oinarritu garenez (ikusi 5.8 diagrama), klase bakoitzak datu-basean homologoa izango den taula bat du. Gainera, eredua osatzen duten klaseen atributuak ere taula hauetako atributuen homologoak dira.

Hortaz, domeinuaren ereduko klase bakoitza, datu-basearen diseinuko taula bakoitzaren homologoa denez, 5.3 atalean azaltzen da klase bakoitzaren eginkizuna.



5.19 Irudia: Web-aplikazioko domeinuaren eredua.

6. KAPITULUA

Inplementazioa

Kapitulu honetan produktuaren garapen-prozesuaren gorabeherak eta erabakien zergatia azalduko da. Sistemaren arkitekturan azaldu den bezala (ikusi 4 kapitulua), aplikazioak zati bat baino gehiago dituenez, zati bakoitza eta elkarren arteko komunikazioa jorratuko da.

6.1 Web-aplikazioa

Sistema guztiaren bihotza zein den esan beharko bagenu, web-aplikazioa litzateke bihotz hori. Izan ere, bertan egiten da informazioaren kudeaketa nagusia eta hau litzateke sistemaren oinarria.

Web-aplikazioa Laravel bidez garatu denez, MVC patroia jarraitu da inplmentazio osoan. Atal honetan, Laravel-en erabili diren gehigarri garrantzitsuenetan jarriko da arreta eta Laravel-eko MVC sistemari aipamen txiki bat egingo zaio.

6.1.1 MVC

Atal honetan Laravel framework-ak MVC patroia errespetatzeko erabiltzen duen sistema azalduko da.

Eredua

Beste MVC sistemetan bezala, ereduaren zeregina, sistemak erabiltzen duen informazioaren irudikapena izatea da. Irudikapen hau egiteko, Laravel-ek *Eloquent ORM*[3] erabiltzen du. *Eloquent ORM* zatiak hainbat zerbitzu eskaintzen dizkigu ereduen artean erlazioak ezarri ahal izateko. Gainera, eskaintzen dizkigun funtzioen multzoek asko laguntzen dute datu-basean egin behar diren bilaketak egiteko.

Eloquent ORM-ri esker, gure datu-baseko diseinua zuzenean eraman dezakegu ereduaren domeinura, ereduen artean dauden erlazioak ezarrita. Ondorengo kodean bi ereduren artean, erlazio bat ezartzeko adibidea ikusiko dugu:

```
namespace App;

use Illuminate\Database\Eloquent\Model;

class Akta extends Model
{

    // Datubasean taularen izena
    protected $table = "akta";
```

```
    // Beste Ereduekin erlazioak ezartzen ditugu
    public function udaletxea(){
        return $this->belongsTo(Udaletxea::class, "udaletxea_fk");
    }
    public function mota(){
        return $this->belongsTo(Pleno_mota::class, "mota_fk");
    }
    public function dokumentuak(){
        return $this->hasMany(Dokumentua::class, "akta_fk");
    }
    ...
}
```

Adibide honetan ikusi daitekeen bezala, eredu bat, *Model* klasea heredatzen duen klase bat da. Taularen izena idaztea aukerazkoa izanik, derrigorrezkoa da laravel-ek eskatzen duen hitzarmena ez bada errespetatzen.

Adibidearen zati garrantzitsua ondoren datozen funtzioak dira. Ikusi daitekeen bezala, funtzio bakoitzak erlazio bat ezartzen du *belongsTo()* edo *hasMay()* bezalako funtzioekin.

Kode honi esker, objektuen arteko erlazioa izugarri erraztu da. Esaterako, Akta motako objektu bat badugu eta akta horren udaletxea lortu nahi badugu, egin behar duguna zera da:

```
    // Lehenengo akta objektua lortuko dugu
    $akta = Akta::findOrFail($id);

    // Ez da funtzio bat bezala deitzen, barruko ezaugarri bat
    // bezala baizik
    $udala = $akta->udaletxea;
```

Bista

Bistaren zeregina, eredua formatu egokian aurkeztea da. Laravel-ek *Blade* izeneko erreminta bat du bere barnean bistekin lan egin ahal izateko. *Blade* txantiloiak denbora errea-

lean konpilatzeko erreminta bat da, eskaeraren arabera, txantiloian informazioa txertatu eta erantzuna prestatuko duena.

Blade-k PHP-ren antza duen sintaxi bat erabiltzen du[2], etiketa sistema batekin funtzionatzen duena. *Blade*-en konpilatzaileak eskaera prozesatu eta gero, bistak exekutatuko ditu eta zerbitzariak itzuliko duen erantzuna sortuko du.

Web-aplikazioa garatzerako unean bista guztietarako erabili da *Blade* sintaxia. Sintaxi honi esker, bistaren zatia txukunago geratzen da eta ez da PHP-ko kode txertatua erabiltzen, *Blade* arduratzen baita lan hau egiteaz.

Kontrolatzailea

Kontrolatzaileak gertaera bati erantzuten dio, gertaera honek eskaera bat izango du eta kontrolatzailea eskaera hau erantzuteaz arduratuko da. Horretarako, eredua eguneratu dezake eta lotutako bistari aginduak bidali diezazkioke.

Helbide atseginak izateko, Laravel-ek URL bideratzailea deritzon sistema bat du. Sistema honen zeregina eskaera zehatz bat (URL bat eta HTTP eskaera mota bat) kontrolatzaile batekin lotzea da. Esaterako, demagun gure webaren helbidea http://www.aktagardena.eus/ dela eta sesioa irekitzeaz arduratzen den kontrolatzaileari deitzeko, GET eskaera bati erantzutea nahi dugula http://www.aktagardena.eus/login helbidera. Eskaera honi erantzuteko URL bideratzailean ondorengo kodea jarriko litzateke:

```
Route::get('login', 'LoginController@index');
```

Goiko kodean *LoginController* izeneko klase batean dagoen *index* funtzioari dei egiten diogu http://www.aktagardena.eus/login helbidera iristen diren GET eskaera guztietan.

Kontrolatzaile bat definitzeko *Controller* klasea heredatzearekin nahikoa da. Gure adibidera eramanda, *LoginController* klaseak *Controller* klasea heredatuko du eta *index* izeneko funtzio bat definituko zaio:

```
namespace App\Http\Controllers;

use App\Http\Controllers\Controller;
```

```
class LoginController extends Controller
{
    // Laravelen hitzarmenaren arabera, funtzio nagusia index() funtzioa da.
    public function index(){
        ...
        // view() funtzioak bistaz gain, bistara pasa nahi ditugun
        // aldagaiak pasatzen uzten digu.
        return view('bista', []);
    }

}
```

Kontrolatzaileko funtzioak eragiketa guztiak egin dituenean, bista bat itzuliko du adierazten diogun aldagaiekin, *Blade*-ek prozesatu dezan.

6.1.2 Laravel-Translatable eta internazionalizazioa

Internazionalizazioa elementu garrantzitsua da aplikazio guztian, dena bai gazteleraz eta bai euskaraz egon behar baita. Internazionalizazioari aurre egiteko, Laravelek itzulpen sistema sinple bat du eta funtzio multzo batzuekin itzulpen batzuk egin daitezke. Hots, hizkuntza arteko espresio baliokideak, bata besteagatik ordezkatzen ditu.

Dinamikoki datu itzuliak sortu nahi ditugunean ordea, zailtasunak izango ditugu, ezin delako itzulpen literalen sistema erabili.

Arazo honi aurre egiteko erabiltzen da Laravel-Translatable gehigarria, modu erosoan datu-baseko egitura bat jarraituta itzulpenak dinamikoki jasotzeko.

Gehigarri honen funtzionamentua ulertzeko, Laravelek natiboki internazionalizazioa nola kudeatzen duen ulertu behar da. Ulertu behar dugun lehen puntua *Locale*-aren ezarpena da:

```
// sistemaren hizkuntza euskarara ezarri
App::setLocale("eu");
```

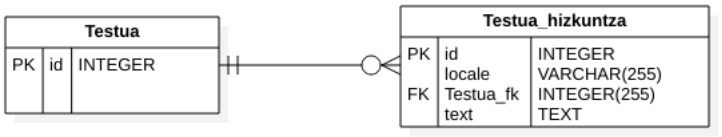
setLocale() funtzioarekin sistemaren hizkuntza ezartzen dugu. Hemendik aurrera hizkuntzarekin zerikusia duten funtzio guztiek, *setLocale()* funtzioan adierazi diogun hizkuntza

erabiliko dute defektuz. Esaterako, literal bat itzuli nahi badugu, *trans()* funtzioa erabiliko dugu:

```
$itzulpena = trans('literalak.gakoa');
```

trans funtzioak jasotzen duen gakoa, itzulpenaren kokapena da. Itzulpenak moten arabera sailkatzeko, fitxategi desberdinetan banatzen dira. *Locale* bakoitzak, bere karpeta du itzulpenen fitxategi sistemaren kopia batekin, hizkuntza zehatz horretara itzulita. Horrela, *trans()* funtzioak *locale*-a duenez, bilaketa huts bat egiten du fitxategi sistema honetan.

Beraz, behin ulertuta *locale*-a aplikazioaren hizkuntza dela, sistema hau datu-base batera eraman dezakegu. Hau da, gako batek eta *locale* batek identifikatzen duen sarrera multzo bat osa dezakegu 6.1 irudiko diagraman ezarritako patroian bezala.



6.1 Irudia: Datu-basean sarrerak hizkuntzarekin gordetzeko egitura.

6.1 irudiko diagraman, bi taula ikus ditzakegu: Testua eta Testua_hizkuntza. Testua taulak hainbat hizkuntzatan itzulitako testu bat irudikatuko du, baina ez du testua gordeko, gakoa soilik. Bestetik, Testua_hizkuntza taula, gako batek, *locale*-ak eta erreferentzia egiten dion Testuaren gakoak identifikatzen du. Azkenik, text izeneko zati bat du, bertan itzulpena joango da.

Nahiz eta eman den deskribapenak diseinuarekin lotura handiagoa duen, garrantzitsutzat jo da atal honetan azaltzea, Laravel-Translatable gehigarriak modu honetan funtzionatzen baitu.

Laravel-Translatable ez da datu-basearen egitura bat ezartzera mugatzen, gehigarri honek ereduekin egiten du lan. Hau da, demagun gure *Akta* ereduak itzulgarriak diren atributuak dituela. Datu-basean aipatu dugun lotura egin ostean, *Akta_hizkuntza* litzatekeen taulan itzulgarriak izango liratekeen atributuak gehituko genituzke. Ondoren, gehigarri honi esker, *Akta* ereduari itzuli nahi diogun atributua eskatuko diogu, nahiz eta ez den ereduaren parte.

Hau da, 6.1 irudiko diagramaren egituran oinarrituta, eredu bat sortzen badugu taula bakoitzeko, Laravel-Translatableri esker ondorengoa egin ahal dugu:

```
$testu1 = Testua::findOrFail($id);
// gogoratu text atributua ez dela Testua taulakoa, Testua_hizkuntza
// taulakoa baizik.
$itzulia = $testu->text;
```

Adibideko kodean nabari den bezala, sistema hau oso erosoa da datu-basetik itzulpenak jasotzeko. Erraztasun honekin lan egin ahal izateko, itzulgarriak diren eredu guztiek ondorengo patroia jarraitu behar dute:

```
namespace App;

use Illuminate\Database\Eloquent\Model;
use Dimsav\Translatable\Translatable;

class Testua extends Model
{
    use Translatable;
    // Taularen izena
    protected $table = "Testua";
    // Itzulgarriak diren atributuak, Testua_hizkuntza eredukoak direnak
    public $translatedAttributes = ['text'];
    protected $translationForeignKey = 'Testua_fk';
    ...
}

class Akta_Hizkuntza extends Model
{
    // Taularen izena
    protected $table = "Testua_hizkuntza";
}
```

6.1.3 Spatie

Gure aplikazioaren rol sistema eraikitzeko, Laravelek Spatie izeneko gehigarri bat eskaintzen digu. Gehigarri honek bi elementu eskaintzen dizkigu: baimenak eta rolak. Gure kasura ekarrita, rol sistema bakarrik erabili da eta baimenen sistema alde batera utzi da.

Gehigarri honekin lan egiteko modu bat baino gehiago dago. Gure kasuan rol desberdinek dituzten baimenak kontuan hartuta, ez da beharrezkoa ikusi ekintza bakoitza baimen batekin lotzea eta ondoren baimen bakoitza rol batekin edo hainbatekin.

Gehigarriaren instalazioaren prozesuaren parte da datu-basearen zatiak beharko dituen taulak gehitzea. Nahiz eta esan den bezala lan egiteko modu asko eskaintzen dituen, aplikazio honetan rol-ekin zerikusia duten taulak bakarrik erabili dira.

Datu-base honetan rol-ak definitu ditugunean, erabiltzaileak sortzerakoan rol hauetako batekin lotuko ditugu. Lotura hau sortzeko eta ezabatzeko spatiek *assignRole()* eta *removeRole()* funtzioak eskaintzen ditu. Hortaz, *langilea* izeneko rola ezarri nahi badiogu kautotutako erabiltzaileari ondorengoa egingo dugu:

```
$user = Auth::user();
$user->assignRole('langilea');
$user->save();
```

Behin erabiltzaileari rol bat ezarri diogunean, honen ekintzak mugatu nahi baditugu Spatie gehigarriak beste funtzio batzuk eskaintzen dizkigu, esaterako *hasRole()*. Hau da, kautotutako erabiltzailearen ekintzak mugatu nahi baditugu kontrolatzailetik muga ditzakegu funtzio honi esker:

```
if(Auth::user()->hasRole('langilea')){
    ...
}
```

6.2 Mahai gaineko aplikazioa

Dokumentuan lehenago azaldu den bezala arkitekturako atalean, mahai gaineko aplikazioa geruza desberdinetan garatuta dago eta, orain, atal honetan garapen hori nola eman

den azalduko da. Gainera, mahai gainekoak dituen beste aspektu batzuk azalduko dira: grabaketa prozesua nola egiten den, audioaren kodeketa eta hashing prozesua.

Aplikazioa geruza desberdinetan garatu dela esaten denan, datuen sarbidea, negozio logika eta interfaze grafikoa banatuta daudela esan nahi da. Kasu honetan, geruza guztiak aplikazioan bertan daude eta geruza desberdinak ez dira elkarren artean sarearen bidez komunikatzen. Atal honetan nola garatu diren azalduko da.

6.2.1 Interfaze grafikoa

Mahai gaineko aplikazioaren interfaze grafikoa Javako *Swing* liburutegia erabilita garatu da. Liburutegi honek interfaze grafiko bat garatzeko klase ugari eskaintzen dizkigu: botoiak, testu kutxak, zerrendak, leihoak etab.

Interfaze grafikoari lotuta dauden klase guztiak **GUI** izeneko karpetan bildu dira. Bertan, nagusiki, interfaze grafiko gisa erakutsiko diren leihoak sortu dira. Izendapenarekin laguntzeko, leiho klaseak *frm* aurrizkia dute, adibidez, *frmHasiera*.

Java *Swing* erabiltzen badugu, interfaze grafikoak diseinatzeko bi aukera ditugu. Alde batetik eskuz kodea idatz dezakegu eta bestetik Eclipsek eskaintzen duen *WindowBuilder* gehigarria erabili dezakegu. *WindowBuilder* gehigarri honek interfazeak modu bisualean egiteko aukera ematen digu, azpitik kodea automatikoki sortzen duelarik.

Leiho bat sortzen dugun bakoitzean, *JFrame* izeneko klase bat heredatuko dugu. Ondoren, klase honi eraikitzaile bat egingo diogu eta, printzipioz, *initializeComponents()* izeneko funtzio bat sortuko dugu. Eraikitzailearen lehen agindua *initializeComponents()* funtzioari deitzea izan behar da, funtzio honetan interfaze grafikoak izango dituen elementu guztiak kargatuko ditugulako.

```
public class frmAdibidea extends JFrame{

    private JPanel contentPane;
    private JLabel lblKaixo;

    public frmAdibidea(){
        // Lehen ekintza elementuak hasieratzea izango da
        initializeComponents();
```

```
        // Hemen gehitu nahi dugun guztia kargatuko dugu
    }
    private void initializeComponents(){
        setResizable(false);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setBounds(100, 100, 387, 113);
        contentPane = new JPanel();
        contentPane.setBackground(Color.WHITE);
        contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
        setContentPane(contentPane);
        contentPane.setLayout(null);

        lblStatus = new JLabel("Kaixo mundua!");
        lblStatus.setBounds(25, 12, 329, 15);
        contentPane.add(lblStatus);
    }
}
```

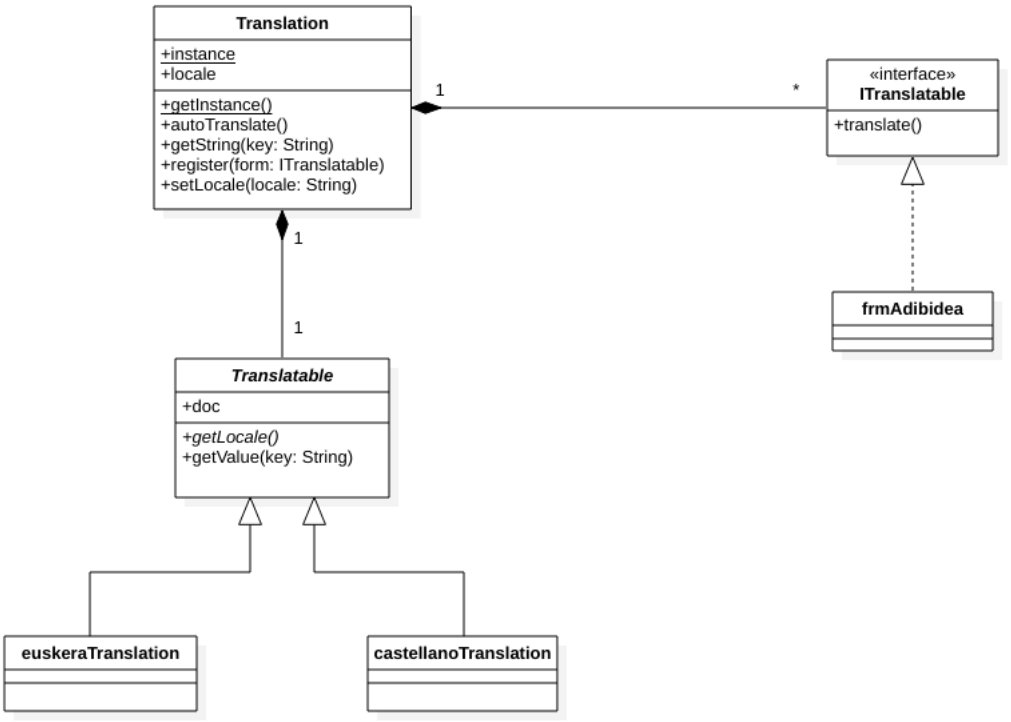
Arkitektura errespeta dadin, interfaze grafikotik egiten diren deiak, negozio logikara egiten dira. Interfaze grafikoaren ardura datuak erakustea eta erabiltzailearekin elkarekintza bat izatea da, horregatik geruza honek lan hau bakarrik egitera saiatuko gara.

Internazionalizazio sistema

Sistema hau derrigorrez internazionalizatuta egon behar du, kasu honetan Euskaraz eta Gazteleraz. Internazionalizazio hori ahalbidetzeko pentsatu den sistema, zerotik diseinatu da eta 6.2 diagraman ikusi daiteke.

Egitura hau ulertzeko, hiru elementu nagusitan pentsatu behar dugu: *Translatable* klase abstraktua, *Translation* klasea eta *ITranslatable* interfazean. *Translatable* klase abstraktuaren zeregina, itzulpenak, gordeta dauden XML fitxategitik jasotzea da, horretarako bi gauza behar ditu:

- Itzulpen zehatz baten gakoa, XML fitxategian gordeta egongo dena itzulpena berarekin.



6.2 Irudia: Internazionalizazioaren egitura.

- Localea, klase abstraktu hau osatuko duen klaseak emango diona. Euskarazko itzulpenak egiteko, *euskeraTranslation* klasean *getLocale()* funtzioa inplementatuko du, eta honen helburua hizkuntzaren localea itzultzea izango da.

XML fitxategi honetan itzulpenak gako batekin batera gordeko dira eta *locale* batekin sailkatuko dira ondorengo egituraren arabera:

```
<translations>
    <lang locale="eu">
        <term key="kaixo">Kaixo mundua!</term>
        <term key="agur">Agur mundua!</term>
    </lang>
    <lang locale="es">
        <term key="kaixo">¡Hola mundo!</term>
        <term key="agur">¡Adiós, mundo!</term>
    </lang>
</translations>
```

Itzulpenak jasotzeko, *Translatable* klase abstraktuak dokumentuan XPath bidezko bilaketa bat egingo du. Hain zuzen ere, erabiltzen duen XPath kontsulta ondorengoa da:

```
String expression = "//lang[@locale='"+getLocale()+"']/term[@key='"+key+"']";
```

getLocale() funtzioa, dagoeneko aipatu den herentziak garatu behar duten funtzioa da. Hau da, *euskeraTranslation* klaseak inplementatzen duen *getLocale()* funtzioak "eu"itzuli behar du. *key* aldagaia itzulketari lotutako gakoa izango da eta bilaketaren emaitza, itzulketa bera izango da.

Translation klasea hizkuntza kontrolatzeaz eta interfaze grafikotik eskatzen diren itzulketak emateaz arduratzen da. Klase honek *Singleton* patroia betetzen du, hau da, instantzia bakarra existizen denez, dei guztiak instantzia berera egiten dira. Software patroi honek duen egituraren puntu garrantzitsuenak hauek dira:

- Eraikitzaile pribatua.
- *getInstance()* funtzio estatikoa, klasearen instantzia itzultzeaz eta instantzia hori bakarra izateaz arduratzen dena.

- Klasearen instantzia estatikoa.

Aplikazioa abiaraztean, *Translation* klaseak eskaintzen duen *setLocale()* funtzioari deitzen zaio. Funtzio honek aplikazioak erabiliko duen hizkuntza ezartzen du, eta jasoko diren itzulpen guztiak, *locale* honen araberakoak izango dira. Hau da, *locale*-a aldatzen dugun unean, itzulpenak beste hizkuntza batean jasoko ditugu.

Gainera, itzulpenak dinamikoak izan daitezen, hirugarren elementua erabiltzen da, *ITranslatable* interfazea. Interfaze honek *translate()* funtzioa inplementatuarazten die hau inplementatzen duten leihoei. Dinamikoki itzuli ahal izateko, leihoetan inplementatzen diren *translate* funtzioek, leihoetan dauden kontrolak berriz marraztea eskatzen dute.

Hala ere, honekin ez da azaltzen nola egiten den konexio hau ezartzeko leihoen eta itzulpen sistemaren artean. Konexioa ezartzeko, *Translation* klaseak bi funtzio ditu: *register()* eta *autoTranslate()*. *register()* funtzioak *ITranslatable* interfazea garatzen duten leihoak jasotzen ditu (leihoko *initializeComponents()* funtzioan egiten zaio deia *register()* funtzioari eta norberaren instantzia ematen dio) eta zerrenda batean gordetzen ditu *Translation* klaseak.

```
// Itzulgarria izango den edozein leihoren initializeComponents()
// funtzioa
private void initializeComponent() {
 // hizkuntzak kargatu
 trans = Translation.getInstance();
 trans.register(this);
 ...
}
```

Ondoren, *Translation* klaseko *setLocale()* funtzioari deitzen zaion bakoitzean, funtzio honek bukaeran *autoTranslate()* funtzioari dei egingo dio. *autoTranslate()* funtzioak erregistratu diren instantzia guztiei dei egingo die eta leiho guztiak batera margotuko dira, hizkuntza berrian margotuz.

```
    public void setLocale(String lo) {
...
// autoTranslate() funtzioari deia hizkuntza aldatzen denan
autoTranslate();
```

```
}
```

6.2.2 Negozio logika

Aplikazioaren bigarren geruza hau, datu sarbidearen eta Interfaze grafikoaren arteko geruza litzateke. Negozio logikaren zeregina, alde batetik, jasotzen diren datuak tratatu behar badira, hauek tratatzea da eta, ondoren, emaitza interfazean erakustea da.

IBusinessLogic

Elkarekintza hau emateko erabiltzen den klase nagusia *BusinessLogic* klasea da. Klase honek *Singleton* patroia jarraitzen du eta *Facade*[1] gisa jokatzen du. Klase honek, *IBusinessLogic* interfazea implementatzen du, eta dei guztiak interfaze honen bitartez egiten dira. Interfaze honek eskatzen dituen metodoak ondorengoak dira:

```
public interface IBusinessLogic {
    // Datuen sarbidea behar duten metodoak
    public int login(String user, String pass);
    public ArrayList<Plenoa> getPlenoak();
    public String getUdaletxeak();
    public int uploadData(String plenoa);
    public int uploadAudio(Plenoa plenoa);

    // Datuen sarbidea behar ez duten metodoak
    public void serialize(Plenoa unekoPlenoa, LinkedList<Itema> gaiak,
                          LinkedList<Speakerra> hizlariak);
    public void encode(frmExportaketa form, Plenoa unekoPlenoa,
                          JProgressBar progressBar);
    public void hash(JProgressBar progressBar, Plenoa unekoPlenoa);
}
```

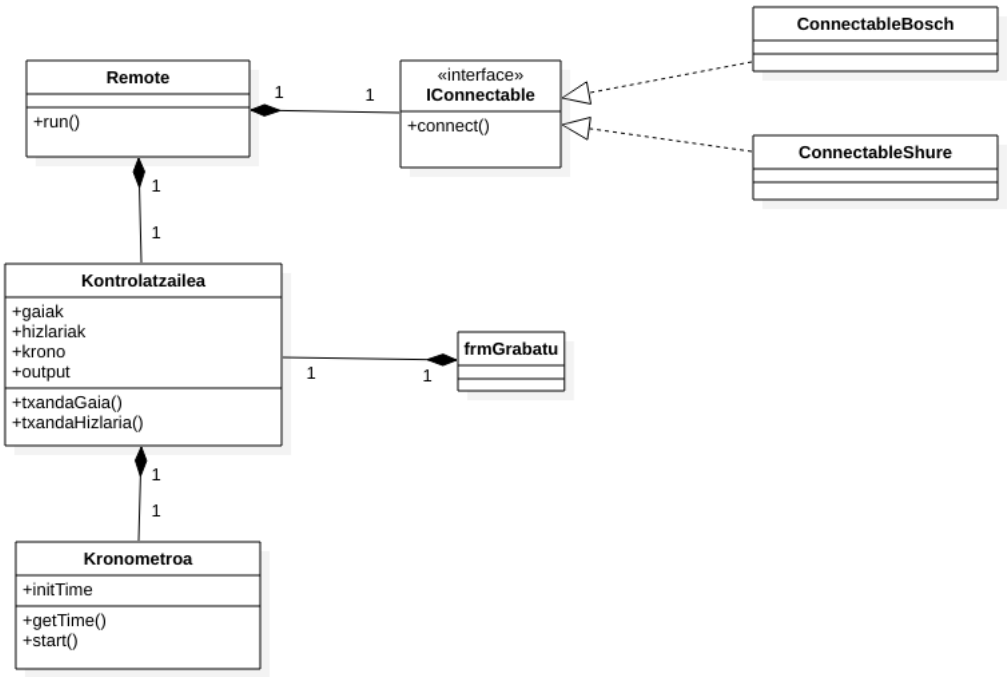
[1] Azpi-sistema konplexu bati interfaze sinple bat eman nahi diogunean, *Facade* patroia erabiltzen da. Oso erabilia izaten da aplikazio baten geruza arteko interfazean. Hitzak fatxada esan nahi du, azken finean besteek fatxada bat bakarrik ikusten dutelako, eta ez atzean dagoen sistema konplexuagoa.

Grabaketa prozesua eta mikrofonia digitala

Grabaketa prozesua ere negozio logikaren parte da. Hala ere, *IBusinessLogic* interfazetik urruntzea erabaki da eta sistema konkurrente bat egitea erabaki da hari desberdinekin, hari hauek kontrolatzen dituen kontrolatzaile batekin.

Java programazio lengoaiak duen abanataila ezagun bat harien kudeaketa ona da. Grabaketa prozesurako oso garrantzitsua da harien kudeaketa hau, mahai gaineko aplikazioa estandar digitalak betetzen dituzten mikrofonia sistemekin elkarekintza izateko diseinatu baita. Mikrofonia sistema hauek interbentzioak kontrolatzeko ahalmena dute (gure kasuan bilkuretako partaideak) eta horregatik informazioa bide desberdinetatik etorri daiteke modu asinkronoan.

Mikrofonia sistema hauetatik etortzen den informazioa hari batetik kontrolatuko da socket bidezko konexio bat erabiliz, eta erabiltzaileak interfaze grafikotik ematen duen informazioa bestetik. Informazio guztia kontrolatzaile sinkrono batekin kudeatuko da, egin behar diren deiak eginez. Grabaketa sistemaren interbentzioen kontrolaren itsura 6.3 diagraman ikus daiteke.



6.3 Irudia: Interbentzioen kontrol konkurrentearen itsura.

6.3 diagrama bi zatitan bana daiteke: zati sinkronoa eta zati asinkronoa. *Kontrolatzailea*

eta *Kronometroa* zati sinkronoak lirateke eta *Remote* eta *frmGrabatu* leihoa zati asinkronoak. Zati asinkronoak hari batean exekutatzen dira eta bilkura baten grabaketako edozein unetan interbentzio bat sor dezakete. Interbentzio bat sortzeak, zati sinkronoari interbentzio bat egitea eskatzea inplikatzen du, hau da, *Kontrolatzailea* klaseari dei egitea.

Zati sinkronoaz hitz egitean esan nahi dena zera da, klase honetako metodoek *synchronized* etiketa izango dutela, ondorengoak bezala:

```
public synchronized void txandaHizlaria(String i, txanda e) {
    ...
}
```

Etiketa honek egiten duena beharrezkoa da hariekin lan egiten dugunean eta hauen arteko elkarekintza behar dugunean. Honi esker, etiketa hau duten funtzioak ezin izango dira deitu une berdinean eta hariek elkar errespetatu beharko dute eta beste harietako deia bukatu arte itxaron beharko dute.

Aipatu diren hariez gain, audioa grabatzen duen beste hari bat dago eta interfaze grafikoan audioaren anplitudea erakusteaz arduratuko den beste hari bat dago.

Audioaren grabaketa egiteko Java lengoaiak eskaintzen duen liburutegia erabili da. Liburutegi honek ordenadorearen soinu sistemara sarbidea ematen digu eta grabaketa amaitzean, soinua *.wav* fitxategi batean gordetzen du.

FFmpeg kodeketa

Grabaketaren emaitzatzat jasotzen dugun *.wav* fitxategiaren tratamentuaz ere, *IBusinessLogic* arduratu behar da. Audioaren kodeketa, grabaketa prozesuaren zati garrantzitsua da biltegiratze eta sare bidezko igoera arazoak aurrezteko. Kontuan hartzen badugu udaletxe bateko bilkura batek 2 ordutik gora iraun dezakeela, sortuko den *.wav* fitxategiaren pisua izugarria izango da.

Arazo hau gainditzeko, *FFmpeg* bidezko kodeketa erabiltzea erabaki zen. Honekin lan egiteko, *FFmpeg* kodetzaileari deiak egiten dizkion *Java Audio Video Encoding* (JAVE)[2] liburutegia erabili da. Java programazio lengoaia erabili arren, *FFmpeg* softwarea ez da

[2]JAVE java lengoaian idatzitako liburutegi bat da. Liburutegi honek, bere barruan, Linux eta Windows sistema eragileetarako *FFmpeg* softwarearen konpilatuak ditu. Liburutegi honen lan bakarra, *FFmpeg* programari dei egitea da.

plataforma-anitzekoa; horregatik, JAVE liburutegiak bere barnean sistema eragile bakoitzeko FFmpeg softwarearen konpilatu bat dakar. Hemen sortu zen plataforma desberdinen arteko bateragarritasun arazoa, JAVE liburutegiak ez baitzuen *macOS* sistema eragilerako FFmpeg konpilatua.

JAVE liburutegiak kodetzaile desberdin bat erabiltzeko aukera ematen digu, *FFmpegLocator* klase abstraktua heredatzen badugu. Klase honek *getFFmpegExecutablePath()* funtzioa eskaintzen du eta bertan fitxategi sisteman dagoen *FFmpeg* exekutagarriaren helbidea itzuliko da. Honek aukera paregabea dirudi *macOS* sistema eragilean erabiltzeko, plataforma honetarako exekutagarriari dei eginda nahikoa litzatekeelako.

Bateragarritasun arazo hau gainditzeko, *macOS* sistema eragilerako ez da JAVE erabili eta zuzenean programari dei egiten zaio. Identifikatu zen arazoaren arrazoia ondorengoa da:

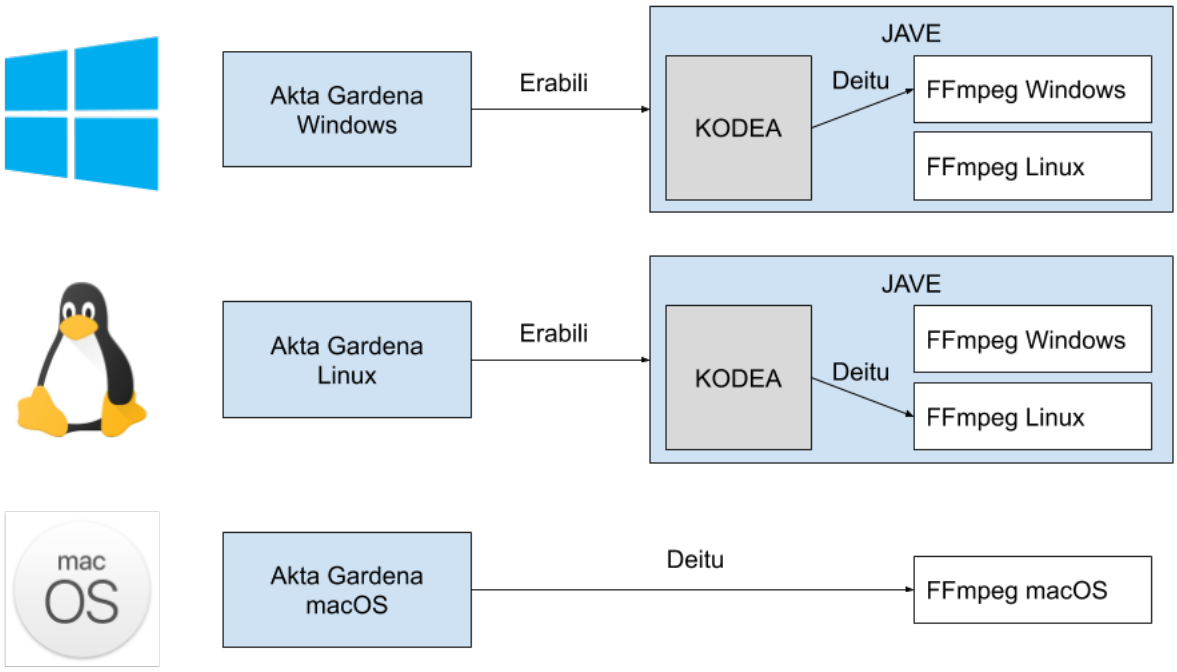
- Lehen arazoa, dagoeneko aipatu den konpilatuaren falta da. JAVE liburutegiak ez du *FFmpeg* konpilatua *macOS* sistema eragilerako bere barnean. Horregatik, macOS sistemako kasuetan, FFmpeg fitxategi sisteman zuzenean biltegiratzen da beste alde batetik.
- Bigarren arazo honetan Javaren plataforma-anitzen arteko bateragarritasun arazoa ageri da, terminaleko deiak ez baitira berdin egiten sistema eragile artean. Dirudienez, *macOS* sisteman Java erabiltzen dugunean *FFmpeg* deitzeko, lehenengo *bash*-ari deitu behar diogu:

```
new String[]{"bash", "-l", "-c", FFmpegArgumentuak}
```

Arazo hauek konpondu ostean, kodeketak, nahiz eta sistema eragile guztietan ez den bide beretik egiten, sistema eragile nagusi guztietan funtzionatzen du.

Hashing-a

Bilkura baten egiazkotasuna bermatzea garrantzitsua denez, segurtasun neurri bezala, grabatzen diren audioen laburpena gordetzea eta zifratzea garrantzitsua da. Audioen egiazkotasuna bermatzeko eta egon daitezkeen maltzurkeriak ezagutzeko, sistemak daraman prozesua ulertu behar dugu.



6.4 Irudia: *FFmpeg* eta JAVE-ren erabilera eta egitura sistema eragileka.

Lehenik eta behin audioa grabatzean *.wav* fitxategi bat sortuko dugu, ondoren fitxategi hau *.mp3* fitxategi batera kodetuko dugu *FFmpeg* erabilita, kodeketa egin ostean, kodetu berri dugun fitxategiaren laburdura sortu, laburdura RSA bidez zifratu eta beste fitxategi batean gordeko dugu. Laburdura lortu dugunean, prozesua bukatzeko, audioa web-aplikaziora igoko dugu.

Prozesu hau grabaketa bat bukatzean egiten da, baina arazoren bat egon bada igoera prozesuan, bide alternatibo bat existitzen da. Esportaketa prozesuak berriro igotzea ahalbidetzen du ez bada informazio guztia garraiatu.

Hemen ikusten dugun arazoa ondorengoa da. Erabiltzaile batek, maltzurkeriaz, grabaketa osteko esportaketa prozesua mozten badu, ondoren *.wav* fitxategia modifikatzen badu, esportaketa egiteko aukera izango du. Fitxategien integritatea apurtze hau saihesteko dator dagoeneko aipatu den hashing prozesua, esportaketa prozesuan berriz ere prozesu berdina jarraitzen denez, audio originalaren eta audio faltsuaren laburdurak lortzean, alderatu egiten dira eta ez badira berdinak programak ez du igotzen uzten.

Nahiz eta segurtasun neurri honek fitxategien egiazkotasuna bermatzen duen neurri batean, saihestu nahi dituen erabiltzaile maltzur batek gaindi ditzake. Hala ere, kontuan izanda sistema hau erabiliko den testuingurua ez dela arriskutsua, tolerantzia maila nahiko altua da.

Denbora-marken serializazio prozesua

Garrantzitsua da nabarmentzea aplikazio hau ez dela grabaketa aplikazio hutsa, eta interbentzioak jasotzeko balio duen tresna ere badela. Denbora-marka hauek grabaketa prozesuan sortzen dira bide desberdinetatik (interfaze grafikotik eskuz edo mikrofonia sistematik etorkizunean) eta amaitzean XML gisa gordetzen dira ondorengo formatuan:

```
<LOG>
    <SessionDescription id='1'>
        <SessionName><![CDATA[Bilkuraren izenburua]]></SessionName>
        <Description><![CDATA[Bilkuraren deskribapena]]></Description>
        <DateTime>2019-05-09T10:30:01.252049</DateTime>
    </SessionDescription>
    <Data>
        <speaker>
```

```
            <id>8</id>
            <speakerName>Maite gorriti</speakerName>
            <initTime>00:03:56</initTime>
            <endTime>00:04:00</endTime>
        </speaker>
        <speaker>
            <id>10</id>
            <speakerName>Antxon Urrutia</speakerName>
            <initTime>00:04:00</initTime>
            <endTime>00:04:02</endTime>
        </speaker>
        ...
        <item>
            <id>25</id>
            <itemName>Lehenengo gaia</itemName>
            <initTime>00:03:54</initTime>
            <endTime>00:04:03</endTime>
        </item>
        <item>
            <id>27</id>
            <itemName>Bigarren gaia</itemName>
            <initTime>00:04:03</initTime>
            <endTime>00:06:37</endTime>
        </item>
        ...
    </Data>
</LOG>
```

Formatu hau erabili izanaren arrazoia Akta Gardenaren lehen fasean erabili zen formatua errespetatzeko izan da. Nahiz eta hasieratik diseinatutako proiektu bat izan den hau, aurreko fasean erabili zen formatua aproposa denez berreskuratzea erabaki da.

6.2.3 Datuen sarbidea

Datuen sarbidea mahai gaineko aplikazioaren komunikazioaz arduratzen den geruza da. Datuen mugimendua gorakakoa eta beherakakoa da, web-aplikaziotik mahai gainekora eta alderantziz hain zuzen ere. Atal honetan ez da komunikazioaren testuinguru orokorraz hitz egingo, egituran eta kodean zentratuko baikara.

DataAccess eta IFacadeDataAccess

Datuen sarbidea izateko *DataAccess* klasea erabiltzen da. *DataAccess* klaseak dagoeneko azaldu den *Singleton* patroia jarraitzen du eta klase honek konexioa deritzon objektu bat gordetzen du. Konexio hau, *DataAccess* objektu bakarra eraikitzean sortuko da konfigurazioaren arabera. Konexio honek *IFacadeDataAccess* interfazea jarraitu behar du, inplementazio bakoitza datu jatorri desberdin bat izanik.

Konexioaren helmuga ezartzeko konfigurazio fitxategian ezartzen den konexio mota erabiliko da, hiru modu desberdin existitzen direlarik: *test*, *local* eta *online*. *test* eta *local* probetarako moduak dira eta normalean *online* moduan egongo da.

Aipatu ditugun modu bakoitzak *IFacadeDataAccess* interfazearen inplementazio propio bat du, ondorengo funtzioak inplementatuz:

```
public interface IFacadeDataAccess {

    public ArrayList<Plenoa> getPlenoak();
    public int login(String user, String pass);
    public String getUdaletxea();
    public int uploadData(String plenoa);
    public int uploadAudio(Plenoa plenoa);

}
```

Informazioaren normalizazioa

IFacadeDataAccess konexioaren ardura garrantzitsu bat, bere informazio jatorritik datozen datuak normalizatzea da. Normalizazio prozesu honetan, interfazeak eskatzen digun

emaitzara egokitu behar gara, esaterako, *IFacadeDataAccess* interfazeak eskatzen digun *login()* funtzioak, informazio jatorria dena dela, emaitza beti bera izan behar da:

- '1' sartutako kredentzialak huts egiten badute.
- '0' dena ondo joan bada.
- '-1' akatsik izan bada eskaeran.

Prozesu hau, mahai gaineko aplikazioak sareko informazioa jasotzeko erabiltzen dituen metodo guztietan egiten da.

6.3 Elkarren arteko komunikazioa

Arkitekturako atalean azaldu den bezala, webaren eta mahai gaineko aplikazioaren artean komunikazioa beharrezkoa da sistemak funtziona dezan.

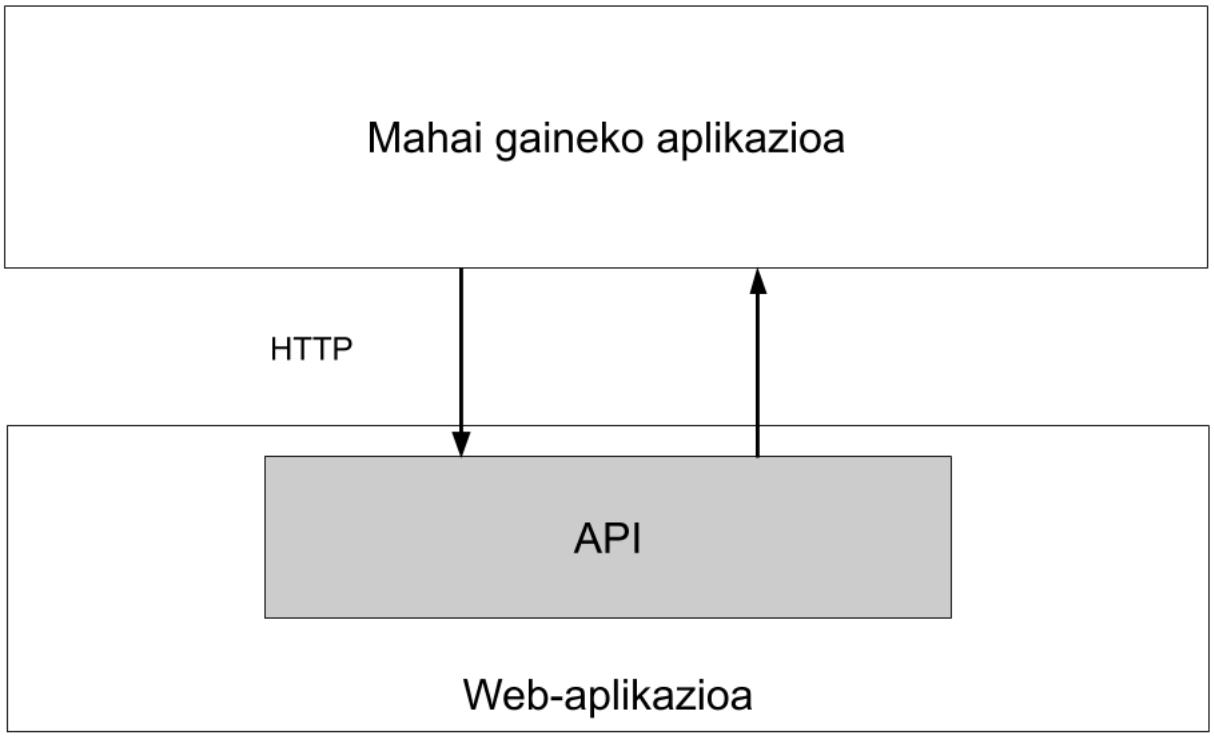
6.3.1 API

Erabiltzaileei buruz, bilkura desberdinetako aktak eta mahai gaineko aplikazaioak behar dituen beste informazio ugari datu-basean gordetzen da. Datu-basea web-aplikazioaren parte denez, bi atalen artean behar den komunikazioa bermatzeko API bat garatu da. API honen zereginak ondorengoak izanik:

- Mahai gaineko aplikazioak eskatzen dituen datuak itzultzea.
- Mahai gaineko aplikazioak bidaltzen dituen fitxategiak eta eskaerak prozesatzea.

API-arekin egiten den komunikazio oro HTTP protokoloaren bidez eta POST deiak bidalita egiten da eta prozesua beti berdina da: lehenengo mahai gainekoak eskaera bat egiten dio API-ari, API-ak eskaera prozesatzen du eta azkenik API-ak erantzun bat ematen dio.

Nortasuna mantenduko duen sesio bat egon dadin mahai gaineko atalean, API-ak autentikazio sistema bat eskaintzen du. Autentikazio prozesua gainditzen denean mahai gaineko aplikazioan, API-ak autentikazio gako bat sortzen du eta mahai gainekoari itzultzen dio. API-an egingo diren beste dei guztiak derrigorrez autentikazio gako hau erabili behar dute erabiltzailea identifikatzeko. Identifikazio gako hau sesio bakarrekoa da eta sesioa irekitzen den bakoitzean gako berri bat sortzen da.



6.5 Irudia: Garatutako APIaren egitura.

Mahai gainekoak informazioa bidali

Bilkura bati lotutako audio eta denbora-marka fitxategiak, eta audio fitxategiaren laburpena sortu direnean, mahai gaineko aplikazioa prest dago informazioa API-ari bidaltzeko.

API-ak bi funtzio ditu lan hau egiteko: *uploadData* eta *uploadAudio*. Laburki funtzio bakoitzak egiten duena ulertu dadin:

- *uploadData* funtzioaren zeregina denbora-markak jasotzea da. Mahai gainekoak dei hau egiten duenean bi elementu gehitzen ditu POST eskaeran: denbora-markak gordetzen dituen XML fitxategia eta autentikazio gakoa. API-ak, XML fitxategi hau irakurtzean, datu-basean sarrerak sortzen ditu baina ez du XML fitxategia bera kontserbatzen.
- *uploadAudio* funtzioaren zeregina, web-aplikazioan audio fitxategia gordetzea da. Mahai gainekoak deia egitean hiru elementu gehitzen ditu: audio fitxategia, audio fitxategiaren laburpena (hashing-a) eta autentikazio gakoa. API-ak fitxategia web zerbitzariaren biltegiratze sisteman gordetzen du eta datu-basean honen kokapena gordetzen du fitxategiaren laburpenarekin batera.

Mahai gainekoak informazioa jaso

Mahai gainekoak informazioa jasotzeko egiten duen eskaera nagusia bilkuraren guztien zerrenda jasotzeko eskaera da. Informazio hau jasotzeko, mahai gaineko aplikazioak *plenoak* funtzioari dei egiten dio, autentikazio gakoa erabiliz identifikatu ahal izateko.

API-ak eskaera jasotzen duenean, grabatzeko prest dauden bilkurei lotutako informazio bilatuko du datu-basean. Bilaketa honek iragazki bat pasatzen du:

- Bilkura bat agertzeko, bilkuraren ospatze eguna bilaketa egin den eguna baino zazpi egun goizago edo beranduago izan behar da. Muga honen iraupena arbitrarioa da, helburua erabiltzaileak gertu dauden bilkurak ikustea da.
- Bilkura zehatz batek denbora markak eta audioa baditu bilaketaren unean, bilkura hau bilaketatik baztertuko da.

Iragazki hau pasa duten bilkurak XML formatua duen fitxategi batean itzuliko dira. Hona hemen XML fitxategi honen formatuaren adibidea:

```
<plenoak>
    <plenoa id="5">
        <izenburua>Ohiko bilkura</izenburua>
        <deskribapena>Ohiko bilkura baten deskribapena</deskribapena>
        <gaiak locale="eu">
            <gaia id="14">Aurreko aktaren onarpena</gaia>
            <gaia id="15">Aurrekontuak</gaia>
            ...
        </gaiak>
        <gaiak locale="es">
            <gaia id="14">Aceptación del acta anterior</gaia>
            <gaia id="15">Presupuestos</gaia>
            ...
        </gaiak>
        <hizlariak>
            <hizlaria id="1" mic="1">Maite Gorriti</hizlaria>
            <hizlaria id="1" mic="1">Antxon Urrutia</hizlaria>
            ...
        </hizlariak>
    </plenoa>
    <plenoa id="6">
        ...
    </plenoa>
</plenoak>
```

7. KAPITULUA

Jarraipen eta kontrola

Kapitulu honetan garapen prozesuan egon diren gorabeherataz hitz egitean, plangintzarekiko egon den desbiderapenaz ere hitz egiten da, noski.

7.1 Proiektuaren garapena eta plangintzaren aldaketa

Proiektuaren garapenean gertatu diren gorabeherak ulertzeko, garrantzitsua da ulertzea proiektuaren definizioan eta forman aldaketak egon direla proiektua garatzen joan den heinean.

Akta Gardena proiektuaren garapenean zehar hainbat bilera egin dira enpresa mailan. Bilera hauetan, enpresako langileez gain, proiektuan lagundu zezaketen enpresa eta pertsonak parte hartu zuten (udaletxeetako mikrofoniak instalatzen dituzten enpresak, udaletxe eta instituzio publikoetako langileak eta besteak). Jorratutako gaiak, batez ere, gerta daitezkeen arazo teknikoak eta erabilgarritasun arazoak dira.

Proiektua sortu zen unean, enpresaren helburua proiektu guztia web-aplikazio bat izatea zen. Hau da, proiektu honetan garatu den mahai gaineko aplikazioan garatu diren funtzionalitate guztiak, web-aplikaziotik egin nahi ziren. Hala ere, proiektuaren garapenean hilabete bat pasa zenean, bilera baten ostean, proiektuaren helburu nagusia (web-aplikazio batean egitea guztia) baztertu egin behar izan zen.

Bilera honetan hartu zen erabakia ez zen arbitrarioa izan. Hain zuzen ere, proiektuak orain duen arkitektura izateko arrazoiak ondorengoak dira (ikusi 4 kapitulua):

- Web-aplikazioen abantaila nagusia plataformaren independentzia da, estandarrak betetzen badira edozein web-nabigatzailetan exekutatu ahalko baita. Hala ere, honek duen kostua internet konexioa da, beharrezkoa baita ekintza ia guztietarako (batez ere, audio fitxategia eta denbora-markak gordetzea gure kasuan). Orduan, internet konexiorik gabe (udaletxe batzuen errealitatea) ezin da web-aplikazioa erabili.

- Mikrofonia-sistema digitalek bitarteko softwareak behar dituzte, internet bidez jasotzen duten soinua deskodetzeko. Gainera, mikrofonia-sistema honek hainbat gertaera jaurtitzen ditu (mikrofono bat piztu da, beste bat itzali da, eta beste gertaera ugari) internet bidez, eta hauek jasotzeko nabigatzaileek eskaintzen ez duten socket interfazeekin lan egin behar da.

Aldaketa hauen ostean erabaki zen gaur egun erabiltzen den arkitektura ezartzea, nahiz eta plangintzaren irismen eta helburuak ez dituen errespetatzen.

Laburbilduz, proiektuaren plangintzan web-aplikazio bakarra egitea pentsatu zen, eta gaur egun, mahai gaineko aplikazioan egiten diren funtzionalitate guztiak, web-aplikazioan

egingo ziren. **Enpresak hartu zuen erabakiaren ostean, ikusegi berri bat eman zitzaion proiektuari eta arkitekturako atalean biltzen den sistema erabiltzea erabaki zen.**

Aldaketa honi, aurretik egindako aldaketak gehitzen badizkiogu, plangintzaren zati handi bat eraldatzea eragin du. 2019ko martxoaren 1an bilera bat izan genuen enpresaren eta udaletxe bateko langile baten artean. Bilera honetan, udaletxeko langilearen inpresioa neurtu nahi zen, Akta Gardena berriaren maketa erakutsiz. Bilera honetarako maketa bat prestatuta eraman behar izan genuenez, hemen gertatu zen lehen aldaketa plangintzan, plangintza egin zen unean bilera hau deitu gabe zegoelako.

Maketa aurreratzeak izan duen eragina, formakuntzan nabari izan da. Nahiz eta Laravel plangintzan adierazten zen bezala ikasi den (maketa egiteko beharrezkoa), multimedia-zerbitzuak eta sinadura digitalak ikertzeari ez zaio modu egokian denbora tartea utzi. Ondorioz, hauei lotutako emangarriak ez dira idatzi eta produktuaren inplementazioan zehar egin behar izan da ikerketa prozesua.

Aipatu diren gertaerak, batez ere, orduen estimazioa, formakuntzako emangarrien sorrera eta egunen banaketa desorekatu dute. Zorionez, ataza banaketa mantendu egin da eta aldaketa guztien ondorioak ez dira katastrofikoak izan proiektuarentzako.

7.2 Burututako lana

Proiektuaren plangintzan 370 orduko proiektua izango zela estimatu zen, proiektuaren hasiera 2019ko otsailaren 4a izanda eta bukaera 2019ko ekainaren 28a. Atal honetan, egindako estimazioaren zehaztasuna neurtuko dugu orduen eta egunen desbideraketa kalkulatuta.

7.2.1 Orduen estimazioa

Aipatu den bezala, proiektuaren plangintza osatzean, orduen estimazio bat finkatu zen. Estimazio honen arabera, 370 ordu iraungo zituen proiektuak. Atal honetan, ataza bakoitzari eman zaion denbora erreala eta plangintzan egindako estimazioaren artean sortzen den desbideraketa azalduko da. Proiektuaren plangintza egin zenean, orduen estimazioa ez zen ausazko zenbaki bat bezala ulertu, muga bat duen ordu poltsa bat bezala baizik. Orduetan dedikazioa neurtzeko, egunero jaso da lanean emandako ordu kopurua bi lekutan: Google Driven eta enpresako ordu inposaketarako erremintan.

Izena	Estimazioa	Erreala	Desbiderapena
Produktua	250 or	242 or	-8
Formakuntza	80 or	60 or	-20
Laravel ikasi	20 or	21 or	+1
Multimedia zerbitzuak ikertu	30 or	29 or	-1
Sinadura Digitalei buruz ikertu	30 or	10 or	-20
Garapena	170 or	182 or	+12
Analisia	10 or	11 or	+1
Diseinua	10 or	15 or	+5
Inplementazioa	140or	146 or	+6
Maketazioa	30 or	36 or	+6
Post-edizioa	70 or	68 or	-2
Grabaketa	40 or	42 or	+2
Probak	10 or	10 or	0
Kudeaketa	50 or	48 or	-2
Bilerak	10 or	16 or	+6
Plangintza	20 or	18 or	-2
Jarraipen eta kontrola	20 or	14 or	-6
Dokumentazioa	70 or	64 or	-6
Defentsa prestatu	10 or	0 or	-10
Memoria idatzi	60 or	64 or	+4
Guztira	370 or	354 or	-16

7.1 Taula: Ordu estimazioaren desbiderapen taula.

Proiektuaren plangintzan egin zen estimazioaren zehaztasuna neurtzeko, 7.1 taula begiratu beharko genuke. Begirada orokor bat emango bagenio, proiektuak uste baino gutxiago iraun duela konturatuko ginateke. Hala ere, ordu kopurua ez da erabat esanguratsua eta horregatik desbideraketa nagusiez gain, beste hainbat kasu ere azalduko dira.

Desbiderapen nagusia formakuntzan eman da. Garapen prozesuan egon diren arazoen erruz, eta proiektua hasi berria zegoenean, plangintzak pairatu zituen aldaketen ondorioz, sinadura digitalei buruz egin behar zen ikerketa ezin izan da egin eta horregatik espero zena baina 20 ordu gutxiago jaso ditu.

Maketazioarentzako estimatu ziren orduak gehiegizkoak izan ziren printzipioz. Hala ere, proiektuak jasan zituen aldaketen ondorioz, mahai gaineko aplikazioa ere maketatu behar izan zen. Mahai gaineko aplikazioa maketatu behar izateak, leiho guztiak eta internazionalizazio sistema bat sortzea ere eragin zuen.

Defentsa prestatzea azken lana denez, eta jarraipen eta kontrola amaitu ostean egingo denez, 7.1 taulan ez da jasotzen emango zaion denbora.

7.2.2 Gantt digramaren betearaztea

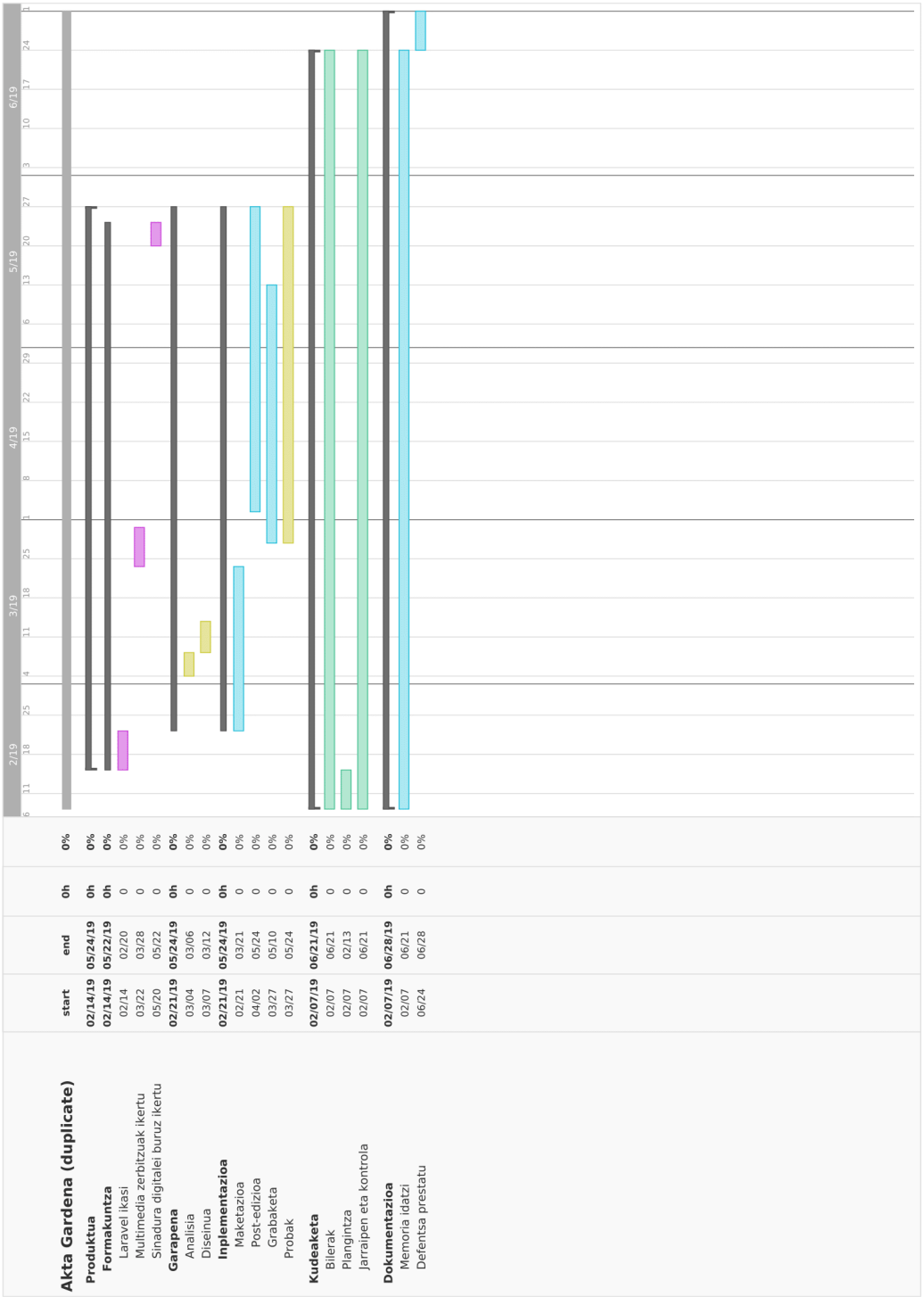
Proiektuaren hasiera 2019ko otsailaren 4a izan zen eta proiektuaren amaiera 2019ko ekainaren 28rako espero zen. Atal honetan, egun horien artean egin den lan banaketa erreala azalduko da.

Amaierako gantt diagrama ikusita (7.1 irudia) eta hasierako gantt diagrama ikusita (2.2 irudia), plangintzarekiko aldaketa handiak egon direla nabari da. Hain zuzen ere, aldaketa hauek bi egunen inguruan eman dira: 2019ko martxoaren 1a eta 2019ko martxoaren 14a.

2019ko martxoaren 1erako maketazioa prest egon behar zenez, ikerketa lanak alde batera utzi ziren eta maketazioa izan zen egin zen lehen gauza. Web-aplikazioko maketa prest zegoela, diseinua eta analisia egin ziren, plangintzarekiko aurreratu zenean.

Hurrengo egun erabakigarria 2019ko martxoaren 14a izan zen, bertan erabaki baitzen proiektuak gaur duen arkitektura izatea. Hori gertatu ondoren, mahai gaineko aplikazioaren maketa egin eta multimedia zerbitzuei buruz ikertu zen. Bai probak, bai grabaketa eta baita post-edizioa batera egin ziren, orain duten erlazioa dela eta.

Memoriaren idazketa eta jarraipen eta kontrola aste bat atzeratu luzatu dira dokumentua osatuago bat lortzeko.



7.1 Irudia: Amaierako Gantt diagrama.

7.3 Arriskuak

Arriskuen kudeaketan proiektuan zehar, gertatu daitezkeen arazoak identifikatzen saiatu gara. Arriskuen identifikazio hau estimazio hutsa da, eta nahiz eta arrisku nabarienak identifikagarriak diren, beste batzuk ez daude hain argi plangintza egiterako unean.

1. **Informazioaren galera:** Ez da informazio galerarik egon.

2. **Ikerketa prozesuan arazoak:** Ikerketa prozesuan bertan ez da arazorik egon, baina garrantzia galdu du proiektuak aurrera egin duen heinean. Hasiera batean, epe batzuk ezarri ziren ikerketarako. Enpresak, proiektuan erlazioa duen pertsona bati, aplikazioaren maketa erakutsi nahi izan zion. Proiektua erakutsi behar zatzionez, maketazio prozesua aurreratu egin zen, ikerketarako eperik gabe geratuz.

3. **Osasun arazoak:** Zorionez, proiektuaren garapena oztopatuko duen osasun arazorik ez da gertatu.

4. **Garapenean arazoak:** Ez da garapen arazorik izan.

5. **Denbora-epeak ez betetzea:** 2. arazoan azaldu denaren arabera, ikerketa prozesuak desagertu egin ziren plangintzaren aldaketaren ondorioz. Horregatik, formakuntzari lotutako emangarrien epeak ez errespetatzeaz gain, emangarriak berak ez dira egin denborarik ez zegoelako.

6. **Estimazio irrealak:** Nahiz eta desbideraketak ez diren oso altuak izan, benetan proiektuaren hasiera bateko dimentsioak txikitu direlako izan da. Funtzionalitate batzuk ezin izan direnez garatu, orduen desbiderapena nahiko txikia da. Hala ere, proiektuaren dimentsioa handiegia da 370 ordutan egiteko.

Arriskuen kudeaketan identifikatu ez zen arazo bat sortu da garapenean zehar. Arazo hau proiektuaren izaera eta forma aldatzea da, behin proiektua hasita zegoenean. Proiektuaren garapena atalean azaldu den bezala, arrazoi teknikoengatik proiektuak azpiegitura aldatu du, eta aldaketa honek izan dituen ondorio nagusiak ondorengoak dira:

- Ezin izan dira transkribapenerako sistema sartu, sinadura digitala ikertu eta mikrofonia-sistema digitalekin elkarekintzarako sistema garatu.

- Hasierako planteamenduan grabaketa prozesua web-aplikazioan parte zen eta orain mahai gaineko aplikazioaren parte da. Audio grabaketak egiteaz arduratzen den mahai gaineko aplikazio bat existitzeak, lan gehigarri bat suposatzen du, web-aplikazioaren eta mahai gaineko aplikazioaren artean komunikazio bat sortu behar delako.

7.4 Kalitatea

Proiektuaren kalitateari dagokionez, ahalik eta kalitate maila altuena mantentzen saiatu da. Hasieratik argi zegoen Akta Gardenren bertsio berria ez zela desagertuko proiektu honen garapena amaitzerakoan. Horregatik, etorkizunera begira, bi puntutan jarri da arreta: Kodearen kalitatean eta dokumentazioaren kalitatean.

Inplementazioa atala (ikusi 6 kapitulua), kodea osatzeko balioko duen dokumentu gisa idatzi da, proiektu hau hartuko duen pertsonak zehaztasunez uler dezan, kodea idazterakoan hartu diren erabakien zergatia jasoz. Testu hau, noski, ez da garatu den kodearen kalitatea bermatzen duen elementu bakarra, kodea bera, objektuan orientaziorako teknikak errespetatuta idatzi da eta softwarea garatzeko diseinu patroiak erabili dira.

Kodearen kalitate altuena mahai gaineko aplikazioan mantentzen saiatu da, enpresan ez baita mahai gaineko garapenik egiten normalean Java erabiliz.

Jarraipen eta kontrolaren kalitatea bermatzeko, zenbaketa egunero eraman da. Nahiz eta dagoeneko aipatu den arazoa gertatu zen plangintzaren aldaketarekin, atazak mantendu direnez, egin den jarraipena egunerokoa izan da. Jarraipen hau excel taula bat erabilita eta enpresaren orduen aitorpenerako tresna erabiliz egin da.

7.5 Komunikazioa

Bai enpresa barruan eta bai tutorearekin eman den komunikazio formal oro eposta bidez eman da. Eposta komunikazio formalak egiteko erabiltzen den sistema izaten da mezuen konstantzia gordetzen delako.

Fakultateko tutorearekin eman den komunikazioa ez da berdina izan une guztietan, proiektuaren hasieran (plangintzarekin, diseinuarekin eta analisiarekin) eta bukaeran (memoriarekin) izan da komunikazio jarraiena. Proiektuaren inplementazioaren zehar komunikazioa, nagusiki, enpresako tutorearekin izan da.

Enpresan eman den komunikazioa bi motatakoa izan da: formala eta informala. Bileretara deialdiak eta eskaera formalak eposta bidez eman dira, eta eguneroko beste gauzak mugikorraren bidez eman da.

8. KAPITULUA

Ondorioak

Atal honetan proiektutik ondorioztatzen ditugun puntuez gain, etorkizunean proiektua nora joango den ikusiko dugu. Behin proiektua amaitu dela, gure emaitza, hasieran espero genuen proiektuarekin aldera dezakegu; horregatik, atal honetan desberdintasun hori ere azalduko da.

8.1 Proiektuaren egoera

Proiektuak jasan dituen eragozpenak, hasieran finkatutako helburuetatik desbideratu dute emaitza. Jarraipen eta kontrolean ageri diren arrazoien ondorioz, proiektua aurrera zihoan hala, proiektuan hasieran egin zen analisia ez zen guztiz zehatza izan. Izan ere, aztertu ziren hainbat erabilpen kasu ezin izan dira inplementatu, eta bestetik aztertu ez ziren beste erabilpen kasu batzuk diseinatu behar izan dira.

Arkitekturako atalean (ikusi 4 kapitulua), proiektuaren azken forma ageri da. Arkitektura hau ez da hasieran pentsatu zena eta arkitektura hau osatzeak denbora asko eskatu du, batez ere, bertan definitzen diren bi zatien arteko komunikazioa.

Aldaketa honetatik abantailak eta desabantailak jaso ditzakegu. Alde batetik, desabantailei dagokionez, arazoa nahiko frustragarria da proiektuaren garatzailearentzat. Proiektua bereizten zuen ezaugarrietako bat hizkuntza teknologien integrazioa zen, transkribapen teknologia hain zuzen ere. Transkribapen teknologiaren integrazioak ez zuen teknologia beraren garapena eskatzen, garatutako teknologia baten integrazioan oinarritzen zen, baina proiektuan aurreikusita ez zeuden beste funtzionalitate batzuk behar handiagoa zuten. Sinadura digitalarekin eta mikrofonia-sistema digitalen integrazioarekin ere berdina gertatu da, arkitektura berriak eskatzen zituen funtzionalitateek behar handiagoa dutenez denbora hauek garatzen eman da. Hau da, proiektua desberdinduko zuten ezaugarri asko ezin izan dira garatu denbora faltagatik.

Beste aldetik, abantailetaz hitz eginda, oso positibotzat jotzen da egin den guztia. Arkitektura berriak eskatzen zituen funtzionalitateak, proiektuari konplexutasun maila igo diote. Esaterako, proiektuaren hasieran aurreikusi gabe zegoen API bat eta komunikazio protokolo bat sortu behar izan dira. Gainera, aurreikusi gabeko teknologia berriak erabili behar izan dira (Java) mahai gaineko aplikazioa garatzeko teknika ugarirekin: konkurrentzia, softwarea garatzeko patroiak, objektu orientazioko teknikak, multimedia liburutegiak, sarearen bidez komunikatzeko liburutegiak, hashing teknikak, zifraketa teknologiak eta audioen kodeketarako softwarea esaterako. Mahai gaineko aplikazioa aurreikusi gabe zegoenez, bertan egin dena ulertzeko, inplementazioa atala irakurtzea gomendatzen da (ikusi 6 kapitulua), bertan dokumentatuta baitago egin dena.

Orokorrean, eta optimistagia izatean erori gabe, amaieran lortu den proiektua lan egokia dela pentsatzen da. Proiektuaren amaierarako erabili diren teknologia guztiak kontuan baditugu (aurreikusita zeuden teknologiak baino gehiago izan dira azkenean) eta lortu den erabilgarritasun maila kontuan izanda (web-aplikazioaren eta mahai gaineko aplikazioa-

ren komunikazioa oso naturala da eta erabiltzaileak ez du ezer egin beharrik bere kaxa, aplikazioak azpitik egiten baitu dena), software aplikazio dotorea dela esan genezake.

Gaur egungo proiektuaren egoera azaltzeko, momentuz garatuta dauden erabilpen kasuak errepasatuko ditugu:

- Bai mahai gainekoan eta bai web-aplikazioan kautotzeko aukera existitzen da (Login egin erabilpen kasua).
- Rol desberdinetako erabiltzaileak (Langileak eta idazkariak) sortzeko eta kudeatzeko aukera existitzen da.
- Alderdi desberdinetako hizlariak sortzeko eta kudeatzeko aukera dago, eta hauek lan-taldeetan (komisioetan, adibidez) antolatu daitezke.
- Bilkuren deialdiak sortzeko aukera dago, bertan, gaiak sortu, hizlariak gehitu, dokumentu gehigarriak gehitu eta kendu daitezke.
- Mahai gaineko aplikazioan dagoen grabaketa tresnak audioa grabatu eta denbora-markak erregistratzen ditu.
- Mahai gaineko aplikazioaren eta web-aplikazioaren arteko komunikazioak bi norabidetan funtizonatzen du: mahai gainekoak informazioa jasotzen du bilkurei buruz eta informazioa bidaltzen dio bueltan web-aplikazioari audio eta denbora-marka fitxategiekin.
- Denbora-markak editatzeko post-edizio tresna garatu da web-aplikazioan.

Egin gabe geratu diren erabilpen kasuetara joko bagenu, konturatuko gara, aplikazioaren funtzionalitate gehigarriak direla. Hala ere, honek ez du esan nahi ez direnik garrantzitsuak, goi-mailan dauden funtzionalitateak direla esan nahi da. Garatu ezin izan diren funtzionalitatean eta egin izan diren lanak ondorengoak dira:

- Transkribapenak egiteko zerbitzuaren integrazioa ezin izan da egin.
- Sinadura digitalei lotutako formakuntza eta funtzionalitatea ezin izan da egin, plangintzaren aldaketarengatik eta denbora faltagatik.
- Mikrofonia-sistema digitalekin elkarekintza eta honi lotutako, multimedia-zerbitzuen formakuntza (Dante eta Onvif[8] estandar digitalei buruz ikertzea) ere ezin izan da

egin plangintzaren aldaketarengatik eta denbora faltagatik. Mikrofonia-sistema digitalek, audio eta bideo seinalea estandar digital batzuen arabera garraiatzen dituzte. Hala ere, mikrofonia-sistemak jaurtitzen dituen gertaerak (mikrofono bat piztu, beste bat itzali eta beste hainbeste) ez dira estandarrak. Marka bakoitzak bere protokoloa erabiltzen duenez eta protokolo hau ikasteko eman behar den denboraren ondorioz ez da ataza hau egin.

8.2 Proiektuaren etorkizuna

Aplikazioak izan dituen gorabeherak aztertu ostean, nahiko argi dago proiektuaren etorkizuna garatu ez diren funtzionalitate eta erabilpen kasuen garapenean dagoela. Akta Gardena proiektuaren helburu nagusia (Akta Gardena proiektua, proiektu honen gainetik legokeen proiektu nagusiari erreferentzia eginez) udaletxeen eta interesatuta leudekeen beste instituzioen prozesura moldatzea da. Hau da, falta diren funtzionalitate hauek, azken finean udaletxeetan lana erraztuko duten funtzionalitateak dira.

Helburu nagusi hori jarraituta, aukera ugari aztertu dira etorkizunari begira. Aukera batzuk besteak baino lehenago egingo dira, horregatik atal honetan epe laburrean eta epe luzean banatuko ditugu etorkizunari begira sortu diren ideiak.

8.2.1 Epe laburrean

Proiektuaren etorkizuna epe laburrean, proiektu honetan garatu ezin izan diren funtzionalitateak garatzetik hasten da. Proiektuaren egoeran azaldu diren funtzionalitateak itxiko zuten proiektu honen lehen fase hauek lirateke:

- Transkribapenak egiteko zerbitzuen integrazioa gehitzea. Honi esker, hiritar batek edo langile publiko batek gai guztien artean hitz-gako bat bilatu nahi badu, transkribapenari esker erraz bilatu ahalko du. Gainera, transkribapenek irisgarritasunean laguntzen diete entzumenean ezgaitasuna duten pertsonei.
- Sinadura digitala behar duten fitxategiak (Dokumentu gehigarriak eta grabaketatik lortzen diren multimedia fitxategiak) sinatzeko aukera gehitzea.
- Estandar digitalen eta mikrofonia-sistemen protokoloen integrazioa (grabaketa prozesuan mikrofonia-sistema digitalei buruz). Honi esker, bilkura baten zehar pro-

gramarekin egin beharko den elkarekintza minimoa izango da, sistema arduratuko baita asmatzen nor ari den hizketan.

- Bideo grabaketaren integrazioa, gaur egun bilkura asko zuzenean ematen baitira. Honi esker, audio-akta izateaz gain, bideo-akta izango litzateke formatua.
- Nahiz eta ez zen proiektu honen irismenean islatu proiektuak hartuko zuen pisuarengatik, administazio atala epe laburrean garatuko da. Bertan, udal berrien profilak sortu eta bestelako administrazio lanak egingo dira.

8.2.2 Epe luzean

Epe luzera begiratuko bagenu, epe laburrean lortutako hobekuntzei esker, hainbat funtzionalitate garatzea aurreikusi da. Alde batetik, arlo instituzionaletik jarraituta lortu daitezken hobekuntzak eginez eta, bestetik, hizkuntza-teknologia desberdinen integrazioarekin.

- Transkribapen-teknologiatik abiatuta, interbentzio baten transkribapena badugu, funtzionalitate oso ahaltsuak egin ditzakegu. Esaterako, interbentzio baten itzulpena lortu dezakegu interbentzio guztiak euskaraz eta gazteleraz eskuragarri egoteko.
- Itzulitako transkribapenekin, bideo grabaketarekin eta irisgarritasunarekin lotuta, transkribapena eta itzulpena lortuta, bideo bat azpititulatu dezakegu hainbat hizkuntzatan. Iametza enpresak dagoeneko lan egin du transkribapenak egiteko zerbitzu batekin, eta zerbitzu honek duen abantailetako bat, denbora-marka batzuk jasotzen ditugula da; hau da, hitz bat segundu batetik bestera esan da. Honi esker, azpititulu fitxategi bat automatikoki sortu ahalko genuen.
- Mahai gaineko aplikazioak duen grabaketa tresnari bozketa tresna gehitzea etorkizunean egingo den funtzionalitate bat da. Biltzarretan ospatzen diren bozketen emaitzei esker, aplikazioak erabakien akta automatikoki sortuko luke, ondoren sinatu eta publikatu.
- Proiektuaz hitz egin dugun une askotan, 'publikatzeaz' hitz egin dugu. Termino hau erabiltzean, akta bat hiritarrentzako eskuragai uzteaz hitz egiten dugu. Hala ere, horretarako, hiritarrek eskuragai izango duten atari bat behar da. Ez dago argi atari hau nola bideratuko den, atari orokor bat izango den udaletxeka sailkatuta edo udaletxe bakoitzak bere ataria izango duen. Argi dagoena da, atari hau beharrezkoa dela proiektuari zentzua emateko, trensa honen muina gardentasuna zabaltzea baita, eta horretarako, beharrezkoa da hiritarrek eskuragai izatea informazio hau.

8.3 Ikasitako lekzioak

Informatika ingeniaritza graduan zehar, proiektuen kudeaketa landu dugunean (Proiektuen kudeaketa irakasgaian) izan ditugun proiektuak jostailuzko proiektuak izan dira. Jostailuzko proiektu hauek, ez badute 'jolasaren' parte bezala nahita garapenean zehar aldaketa bat jasaten, abantaila argi bat dute: Proiektua estatikoa izango da prozesu guztian, edo behintzat horrela izan da ikasi dugun guztietan. Errealitatea beste bat da, enpresa munduan egiten diren garapen komertzialek garapen bat aldatu dezakete eta.

Garapen batean ezartzen diren helburuak ez dira itxuraz diruditen bezain finkoak eta denbora pasatzen den heinean, arrazoi desberdinengatik, proiektu baten helburuak eta irismena alda daitezke. Horregatik, oso garrantzitsua da arrisku hauek plangintza on batean aurreikustea.

Adi, honek ez du esan nahi enpresa guztietan egiten diren garapenak ez doazenik helburu baterantz. Edozein proiektutan gerta daitekeen bezala, proiektu berri bat jaiotzen den unean, jasan ditzakeen aldaketak kontuan izan behar dira. Garapenak aurrera doazen heinean, helburuak finkoagoak eta zehatzagoak dira eta, normalean hobekuntza konkretu bat izango da, ez garapen orokor bat.

Laburbilduz, oso garrantzitsua da plangintza on batean jasotzea, agian aurreikusitako proiektua eta amaieran izango duguna desberdinak badira ere. Horrela, arrisku hau aurreikusten badugu, arazoari aurre egiteko erraztasun handiagoa izango dugu proiektua bertan behera utzi gabe. Zorionez, proiektu honen kasuan, helburuen aldaketa honi esker, erabili behar ez ziren hamaika teknologia integratu dira, eta nahiz eta proiektuaren aurreikuspena eta emaitza ez datozen bat, emaitza oso egokia da, bai garatzaile, enpresa eta zuzendariak asebete direla esan dezakegu.

Bibliografia

[1] (2019). About qt. https://wiki.qt.io/About_Qt.

[2] (2019). Blade templates. https://laravel.com/docs/5.8/blade.

[3] (2019). Eloquent: Getting started. https://laravel.com/docs/5.8/eloquent.

[4] (2019). ¿qué es django? https://tutorial.djangogirls.org/es/django/.

[5] C, P. (2016). .net core: .net se convierte en multiplataforma con .net core. https://msdn.microsoft.com/es-es/magazine/mt694084.aspx.

[6] G, E. (2016). Dante, el protocolo de audio en red. https://www.hispasonic.com/reportajes/dante-protocolo-audio-red/42068.

[7] S, K. (2018). ¿qué es laravel? ventajas del desarrollo a medida para tus proyectos. https://www.synergyweb.es/blog/laravel-desarrollo-medida/.

[8] Á, R. (2017). ¿qué significa el protocolo onvif? https://www.nivianhome.com/es/que-es-onvif/.