eman ta zabal zazu

Universidad del País Vasco
Euskal Herriko Unibertsitatea

INFORMATIKA FAKULTATEA
FACULTAD DE INFORMÁTICA

Informatika Ingeniaritzako Gradua

Software Ingenieritza

Alzheimer gaixoentzat musika bidezko terapiarako ariketak dituen mugikorrerako aplikazioa

Mikel Armendariz Alberdi

Zuzendariak: Ekaitz Jauregi eta Iñigo Perona

Donostia, 2020ko otsaila

Laburpena

Gradu Amaierako Lan honen helburua alzheimerra jasaten duten gaixoei laguntzeko musikoterapian oinarritutako aplikazio bat sortzea izan da.

Aplikazioak musikan oinarritutako ariketa desberdinak eskainiko ditu, eta gainera, nahi izanez gero, taldean eta denbora errealean egin ahalko dira ariketa hauek.

Bestalde, zaintzailearen rola ere sortu da, eta hauek gaixoen jarraipena eramateko, ariketetan lortuko dituzten emaitzak ikusi ahalko dituzte. Horrez gain, gaixoei gauzak erraztu ahalko dizkie, izan ere, zaintzaile hauek izango baitira taldeko ariketak prestatuko dituztenak, gaixoek soilik ariketa gauzatu beharko dute.

Gainera, melodia berriak dinamikoki sortuko dira, hau da, ariketa bakoitzerako melodia originaletatik eratorritako melodia berriak sortuko dira, honela ariketak horren errepikakorrak ez izateko.

Gaien aurkibidea

Irudien zerrenda

Taulen zerrenda

1

Sarrera

Garunari eragiten dioten endekapenezko gaixotasun progresiboei dementzia deritze eta gaixotasun hauek memoria, ikaskuntza, portaera eta emozioei eragiten die. Alzheimerra da dementzia mota ohikoena. Gaixotasun hauen kausa eta tratamenduaren inguruan aurkikuntzak egiten ari dira, baina orain arte sortu diren sendagaiek oso efektu mugatua dute.

Dementziaren etapa aurreratuan hitz egiteko gaitasuna gutxitzen joaten da pixkanaka, erabat galtzera iritsi arte. Kasu hauetan musikoterapiak bestelako komunikazio mota bat eskain diezaieke gaixoei, izan ere hitz egin ezin duten gaixoak gai izaten baitira musika jarraitu eta ahapetik kantatzeko.

Terapia musikalaren aplikazioak eragin positiboa izan du alzheimerra duten gaixoengan, hauen memoria, orientazioa, umorea eta bizi kalitatea hobetu baititzake [1].

Musikan oinarritzen diren terapia hauek bi talde handitan banatzen dira: terapia hartzaileak eta terapia aktiboak. Terapia hartzaileetan gaixoak bere bizitzako etapa desberdinekin lotutako kantuak entzuten ditu, eta honek, sentsazio onak sentiarazten dizkio eta alzheimerrari aurre egiten laguntzen dio. Aldiz, terapia aktiboetan gaixoek musikaren sorkuntzan parte hartzen dute kantuen zati txikiak interpretatuz.

Proiektu honetan, musikoterapiako ariketak landu ahal izateko mugikorreko aplikazio bat garatuko da.

1.1 Motibazioa

Terapia musikalean ariketa desberdinak landu daitezke: abestiak abestu, musika entzun, musikaren jarraipen erritmikoa egin txaloen edo perkusio instrumentuen bidez, etab. Gipuzkoako foru aldundiagatik finantzatuta dagoen Remembering Reminding proiektuak musika terapian erabili daitezkeen zenbait ariketa berri garatzea du helburu. Besteak beste:

- Asmatu melodia originala: melodia original bat emanda honi zenbait aldaketa burutuko zaizkio eta ondoren bertsio berri hau entzunda gaixoak melodia originala asmatu beharko du.
- Asmatu noiz aldatu den melodia: gaixoak ezagutzen duen melodia bati amaiera aldatuko zaio eta helburua gaixoak melodia zein puntutatik aurrera aldatu den asmatzea izango da.

Ariketa hauek burutu ahal izateko melodia sortzeko gai den programa bat behar da. Lan honetaz Robotika eta Sistema Autonomoa Ikerketa Taldea arduratu da [2]. Proiektu honen helburua, sortutako ariketa berri hauekin lan egin ahal izatea baimenduko duen mugikorrerako Android aplikazioa garatzea da.

Esan bezala, aplikazioa gaixoei laguntzeko sortu da, baina nola jakin benetan laguntzen ari zaien? Hori dela eta, gaixoen estatistikak gordeko dira eta gaixoa bera edo bere zaintzaileak gai izango dira mugikorrean bertan bere estatistikak ikusteko eta horrela jarraipen bat eramateko.

Lehen aipatu bezala, ariketa hauetarako eraldatutako kantak beharko dira, eta hauek sortzeko algoritmo bat erabiliko da. Honela, kanta originaletik abiatuta nahi adina bertsio sortu ahal izango dira eta gehien gustatutakoak gorde eta ondoren ariketetan erabili.

Gaur egun mugikorretako aplikazioak oso erabiliak dira arlo desberdinetako hainbat gauza egiteko, eta honelako tresna batentzat ere mugikorra oso gailu aproposa izan daiteke. Izan ere edozein une eta lekutan Android aplikazioa eskura izan ahalko da eta ez da beharko inolako ekipo berezirik.

1.2 Helburuak

Proiektuaren helburu nagusia alzheimerra duten gaixoei laguntzeko tresna bat sortzea da. Hori dela eta, sortuko den aplikazioaren erabilpena oso erraza izan beharko da. Horretaz gain, dementziako lehendabiziko etapetan dauden gaixoei lagundu nahi zaie, eta honetarako, tresna honek ariketak ahalik eta modu entretenigarrienean eskainiko ditu. Ariketa hauek banaka edo taldean egin ahalko dira, betiere zaintzailearen aholkularitzapean. Ariketak taldean egitean, taldeko gaixo guztien mugikorretan denbora errealen erreproduzituko dira kantak. Modu honetara musikologoaren dependentzia gutxitzen da, nahiz eta musikologoarekin lan egitea gomendagarria izan, gaixoak terapia musikaleko ariketak burutu ahal izango ditu musikologoa aurrean egon gabe.

Bestalde, aplikazio honekin, garatutako ariketak egokiak diren edo ez probatu ahal izango da. Hau da, ariketak oso modu errazean jarri ahalko dira martxan, eta horrek, ariketa hauen eraginkortasuna neurtzea erraztuko du.

Horrez gain, gaixoei jarraipen bat ematea oso garrantzitsua izango da. Hori dela eta, gaixoen puntuazioak eta estatistikak gordeko dira, eta gaixoa bera edo zaintzaileak gai izango dira estatistikak ikusteko.

Bestalde, gaixoei plan pertsonalizatuak eskaini nahi zaizkie, beraien gaixotasunaren mailaren arabera gehien egokitzen zaizkien ariketa jartzeko. Hori dela eta, nahi adina kanta pertsonalizatu desberdin automatikoki sortu ahalko dira eta gaixoei egokitzen zaizkienaren arabera gorde edo alde batera utzi.

Laburbilduz, alzheimerra duten gaixoentzat tresna bat sortuko da, gaixo desberdinentzat, gaixotasun maila edo beharren arabera, egokitzen zaizkien ariketak eskaini ahal izateko. Honela gaixotasunaren lehen faseetan dauden gaixoei alzheimerra atzeratzen edo geldiarazten lagundu nahi zaie.

1.3 Aurrekariak

1.3.1 Testuingurua

Garuneko gaixotasun desberdin ugari daude, eta hauetako bat alzheimerra da. Alzheimerrak gaixoen memoriari eragiten die, hau da, gaixoak memoria galtzen joaten dira pixkanaka. Gaur egun, jada hainbat teknika desberdin ezagutzen dira alzheimerraren eraginak atzeratzeko, eta hauetako bat musikoterapia da. Hain zuzen, Android aplikazio hau teknika horretaz baliatuko da gaixoei laguntzeko.

Horrez gain, mugikorretako aplikazioak indar handia hartzen ari dira azken urteetan. Dagoeneko mota guztietako aplikazioak sortu dira, bideo jokoetatik hasi eta osasunarekin lotutako aplikazioetaraino. Izan ere, mugikorrek jasan duten eboluzioak eta hauetarako aplikazioak sortzeko dauden plataforma desberdinek eskaintzen dituzten tresnek, aplikazio hauen garapena asko errazten dute.

Hori dela eta, proiektu honetan musikoterapian oinarritutako mugikorreko aplikazio bat garatzea erabaki da.

1.3.2 Antzeko aplikazioak

Lehendik badira musikoterapian oinarritutako mugikorretako aplikazioak, esate baterako, hauetako batzuk pertsona bakoitzaren egoera sentimentalaren arabera kanta desberdinak aholkatzen eta erreproduzitzen dituzte. Adibidez, MoodAgent [3] aplikazioari zure sentimenduen mailak eman ezkero, honek hainbat kanta gomendatuko dizkizu eta bertan erreproduzitu ahalko dituzu.

Bestalde, badira alzheimerra edo bestelako dementziaren bat duten gaixoei laguntzeko aplikazioak, eta hauetako batzuk forma geometriko eta koloreetan baliatzen dira ariketa desberdinak eskaintzeko, memoria lantzeko aproposak baitira. Esaterako, Game Show [4] mota honetako aplikazio bat izango da. Proiektu hau bereziki alzheimerra duten pertsonei laguntzeko aplikazio bat izango den arren, eta laguntzeko tresna nagusia musika izango den arren, garbi dago antzeko aplikazioak lehendik badirela, eta ez dela musika bakarrik erabiltzen memoria edo garuna lantzeko.

Hala ere, proiektu honetan garatu den aplikazio honek denbora errealean taldean ariketak egitea ahalbidetzen du, eta zaintzaileari emango zaio ariketen kontrol guztia. Izan ere, giza faktorea oso garrantzitsua izan daiteke dementzia jasaten duten gaixoekin lan egiterakoan. Gainera, automatikoki musika sortzeko algoritmoak, hainbat ariketa eta kanta berezi sortzeko ahalmena ematen du. Zentzu honetan, mota honetako aplikazioen artean, hau nahiko berritzailea dela esan daiteke.

1.4 Memoriaren egitura

Dokumentu hau, hainbat zati desberdinetan banatuta dago. Hasteko, sarrera txiki baten ondoren, proiektua gauzatzeko motibazioa, bere helburuak eta aurrekariak azaldu dira (1 kapitulua). Bigarrenik, proiektuaren kudeaketari buruzkoa azalpenak emango dira, hau da, atazak, arriskuak, etab. (2 kapitulua). Jarraitzeko, aplikazioaren eskakizunen bilketa azalduko da, aktore eta erabilpen kasu desberdinen azalpenekin (3 kapitulua). Ondoren, aplikazioaren diseinua (4 kapitulua) egingo da, eta gero erabilitako tresnak (5 kapitulua) eta garapena (6 kapitulua) nola egingo den azalduko da. Gero, aplikazioari egin zaizkion proba desberdinak azalduko dira (7 kapitulua), eta azkenik, proiektuaren ondorioak eta hausnarketa (8 kapitulua) azalduko dira.

2

Proiektuaren kudeaketa

Atal honetan proiektuaren planifikazioa eta proiektuaren garapenean zehar eramango den kudeaketa azalduko dira. Horretarako, hainbat zati desberdinetan banatuko da atal hau: proiektuaren helburuak, denboraren estimazioak, desbiderapenak, arriskuak, etab.

2.1 Deskribapena eta helburuak

Proiektuaren helburu nagusia alzheimerraren lehen etapan dauden gaixoei laguntzeko Android aplikazio bat garatzea da, gaixotasunaren eraginak mantsotzeko musikoterapia erabiliz. Honetarako, mugikorreko aplikazio bat sortu nahi da, honek musikarekin lotutako ariketa desberdinak eskainiko dituelarik.

Aplikazioa garatu aurretik proiektuaren kudeaketa prozesua gauzatu behar da, hau da, proiektuaren estimazioak, aurrekontua eta plangintza egin behar dira. Gainera, proiektuan zehar gerta daitezkeen ezustekoak ere identifikatu behar dira, horrela, proiektuaren garapenean egon daitezkeen arazoak aurreikusteko eta arriskuak ekiditeko.

2.2 Irismena

Proiektua hainbat azpiatazetan banatuko da eginbeharrak hobeto identifikatu ahal izateko. Horrez gain emangarriak ere zerrendatuko dira, egin eta entregatu beharreko guztia zerrenda batean argi eta garbi ikusi ahal izateko.

Hasiera batean, proiektua guztiz amaitzea lortu nahiko da, hau da, proposatuko diren erabilpen kasu guztiak izatea, diseinu egoki bat izatea, etab. Baina denboraren arabera erabaki batzuk hartu beharko dira, beraz, hasteko helmuga minimoak aukeratuko dira, eta behin horietara iritsita, denbora baldin badago gauza gehiago egingo dira.

Lehenengo urratsa bakarkako ariketak egitera iristea izango da. Honekin batera, ariketetako melodiak erreproduzitzerakoan, notak modu desberdinetan bistaratu ahal izan beharko dira. Horrez gain, midi originalak entzuteko aukera izan beharko da, eta egindako ariketen datuak jaso beharko dira, ondoren estatistikak erakusteko.

Behin horra iritsita, ahal bada, taldeka ariketak egitera iritsi beharko da, eta horretarako hainbat mugikorren artean denbora errealeko komunikazioa egon beharko da. Horrez gain, ahal den neurrian erreproduzitzerakoan sinkronizatu beharko dira, eta hau agian ez da hain erreza izango.

Bestalde denbora badago, ariketak alde batera utzi eta midi berriak sortu eta hauek gordetzeko aukera eskaini nahi da.

Behin hori eginda aplikazioari diseinu eder bat eman nahi izango zaio, eta noski ahal den neurrian noten errepresentazioa hobetu nahiko da, izan ere, hasiera batean oso modu sinplean egingo baita. Baina hau ez da hain garrantzitsua izango, hasiera batean funtzionaltasunari emango baitzaio lehentasuna.

2.2.1 Emangarriak

- Memoria: proiektu guztia barne hartzen duen dokumentua idatziko da. Bertan, informatika ingeniari batek, proiektu hau zerotik hasita, berriz ahalik eta berdinen garatu ahal izateko beharko lituzkeen zehaztapen guztiak emango dira.
- Aplikazioa: aplikazioaren kodea eskuragarri ipiniko da, bertsio kontrola ahalbidetzen duen GitHub [5] plataforman.

2.2.2 Atazak eta denboraren kudeaketa

Proiektua bost zatitan banatuko da: formakuntza, kudeaketa, garapena, dokumentazioa eta aurkezpena.

Lehen pausua formakuntza izango da, izan ere, proiektuan ezagutzen ez diren teknologia berri batzuk erabili beharko dira. Hain zuzen, proiektuaren helburuak finkatu ondoren ikerketa bat egin beharko da, eta honela, tresna erabilgarriak bilatu eta hauetan trebatzea izango da atal honen funtsa.

Formakuntzaren zatiaren ondoren, proiektuaren kudeaketa landuko da. Ataza honetan proiektuaren plangintza, irismena, emangarriak, atazak eta denboraren kudeaketa, eta arriskuak baloratuko dira eta hauen inguruko informazioa bilduko da.

Garapenaren atalean midi fitxategien zerbitzariaren eta Android aplikazioaren sorkuntza eta garapena sartuko dira. Android aplikazioa garatzeko Android Studio [6] ingurunea erabi-

liko da eta midi fitxategien zerbitzaria sortzeko berriz, Eclipse [7] ingurunean Dynamic Web Project [8] motako proiektu bat garatuko da Java-z, REST API [9] zerbitzua emateko.

Dokumentazioaren funtsa memoria lantzea izango da. Hain zuzen, proiektuaren xehetasun guztiak azalduko dituen txosten bat garatu beharko da.

Aurkezpenaren zatian berriz, memoria laburtuko duen aurkezpen bat prestatu beharko da.

Beraz, 2.1 irudiko diagraman ikus daitekeen moduan, hainbat ataza eta azpi-ataza izango ditu proiektu honek.

Atal bakoitzaren azpi-atazak eta hauen deskribapenak bilduko dira 2.1 taulan. Bestalde, 2.2 taula erabiliko da, ataza bakoitzean pasatuko den denboraren estimazioa egiteko. Azpi-ataza guztien estimazioa egin eta gero, hauen batura kalkulatuko da, eta azkenik batura totala.

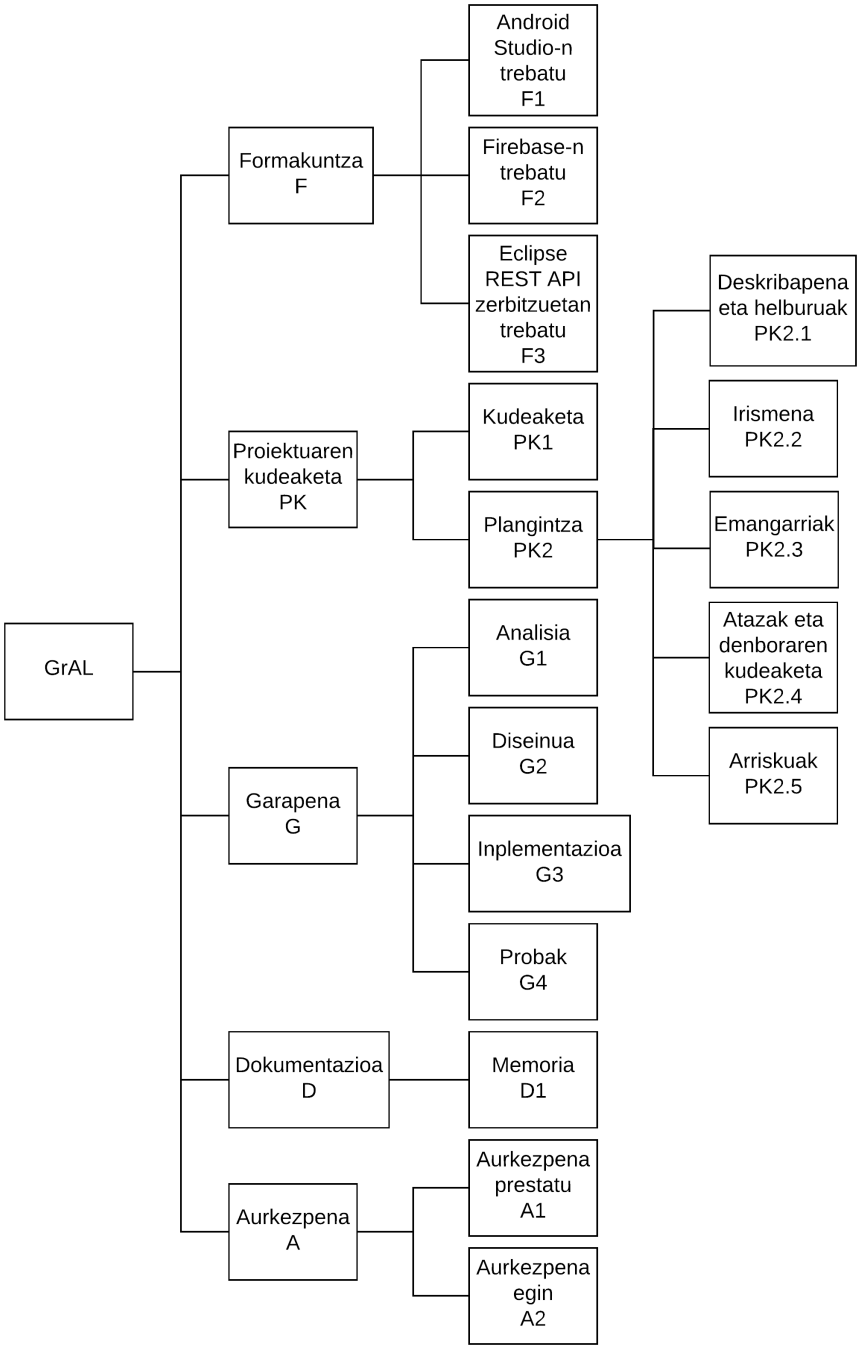
Behin sartutako orduak kalkulatuta, hasieran estimatutako orduekin konparatuko dira eta honela egindako estimazioaren desbiderapenaren ideia bat izango da. Hain zuzen horri buruzko informazioa erakutsiko da 2.3 taulan. Hausnarketa bat egiteko eta etorkizunera begira estimazioak egiten hobetzeko balioko digu.

Ikus daitekeen moduan, ia eremu guztietan egon da desbiderapena, ia beti behar izan dena baino denbora gutxiago estimatu delako. Kasu batzuetan desbiderapenak nahiko nabarmenak izan dira.

Esate baterako, kudeaketaren zatian 20 ordu estimatu dira eta 36 behar izan dira, hau da, estimatutako denboraren %80a gehiago. Desbiderapen honen arrazoia bilerak izan dira, izan ere, esperotakoa baino bilera gehiago egin dira.

Bestalde, diseinuan ere 30 ordu estimatzen ziren eta 48 behar izan dira, estimatutako denboraren %60a gehiago. Kasu honetan, diseinuan esperientzia gutxi izatea izan da arrazoi nagusia, gauza batzuetan beharrezkoa baino denbora gehiago behar izan baita. Honek inplementaziorako ordu gutxiago izatea ekarri du. Hori dela eta estimatzen zirena baino 10 ordu gutxiago sartu dira, aplikazioaren interfazean eragina izanik.

Memoriaren kasuan ere 15 orduko desbiderapena egon da, baina atazari eskaini behar izan zaion ordu kopurua askoz ere handiagoa denez ez da hain esanguratsua.



Irudia 2.1. Atazak eta azpi-atazak erakusten diren diagrama.

Ataza	Deskribapena
F Formakuntza	
F1 Android Studio-n trebatu	Android Studio garapen ingurunearen ikasketa. Kodea Java-z idatziko den arren, ingurune berria izango da eta hemen trebatu beharko da.
F2 Firebase-n trebatu	Firebase teknologiako trebaketa prozesua. Hau guztiz ezezaguna zaigunez dokumentazio irakurri eta hainbat tutorial ikusi beharko dira.
F3 Eclipse-ko REST API zerbitzuetan trebatu	Eclipse-k eskaintzen dituen REST API zerbitzarien inguruko informazioa bildu eta zerbitzari hauetan trebatzea.
PK Proiektuaren kudeaketa	
PK1 Kudeaketa	Proiektuaren kudeaketa orokorra, hau da, erabakiak hartu, eman beharko diren pausuak aukeratu, etab.
PK2 Plangintza	Proiektuan zehar jarraituko den plangintza egin, eremu desberdinak kontuan hartuta.
PK2.1 Deskribapena eta helburuak	Proiektuaren deskribapen orokorra eta helburu nagusiak emango dira.
PK2.2 Irismena	Proiektuaren mugarriak zein diren azalduko da, hau da, proiektuarekin noraino iritsi nahi den, bere norainokoak.
PK2.3 Emangarriak	Entregatu beharreko dokumentu eta fitxategien zerrenda.
PK2.4 Atazak eta denboraren kudeaketa	Proiektua gauzatzeko bete beharko diren atazen deskribapenak eta hauetan sartu beharko den denbora kopuruaren aurreikuspena emango da.
PK2.5 Arriskuak	Arrisku potentzialak zerrendatuko dira.
G Garapena	
G1 Analisia	Garapena hasi aurretik proiektuaren bezeroaren eskaerak jasoko dira, eta ondoren analisi bat egingo da, ea proiektua bideragarria den ikusteko.
G2 Diseinua	Inplementazioa errazteko, lehenago aplikazio guztiaren diseinua egingo da. Honela, ondoren inplementazioa asko erraztu eta azkartuko da.
G3 Inplementazioa	Implementazioa azalduko da xehetasun guztiekin eta edonork ulertu ahal izateko moduan.
G4 Probak	Behin aplikazioa inplementatuta, gauzatuko diren probak azalduko dira. Honela, aplikazioaren erabilgarritasuna eta eraginkortasuna kalkulatu eta sortu diren akatsak zuzentzeko.
D Dokumentazioa	
D1 Memoria	Proiektuaren azalpenak ematen diren dokumentua idaztea. Atal desberdinetan zatituko da, eta bakoitzean informazio zehatza emango da.
A Aurkezpena	
A1 Aurkezpena prestatu	Gradu amaierako lana (GrAL) tribunal baten aurrean defendatzeko prestatuko den aurkezpena.
A2 Aurkezpena egin	Prestatu den aurkezpena gauzatzea eta egindako galderei erantzutea.

Taula 2.1. Atazak eta hauen deskribapenak biltzen dituen taula.

Ataza	Estimatutako orduak	Guztira
F Formakuntza		
F1 Android Studio-n trebatu	10	20
F2 Firebase-n trebatu	5	
F3 Eclipse-ko REST API zerbitzuetan trebatu	5	
PK Proiektuaren kudeaketa		
PK1 Kudeaketa	30	50
PK2 Plangintza	20	
G Garapena		
G1 Analisia	20	140
G2 Diseinua	30	
G3 Inplementazioa	80	
G4 Probak	10	
D Dokumentazioa		
D1 Memoria	90	90
A Aurkezpena		
A1 Aurkezpena prestatu	10	10.5
A2 Aurkezpena egin	0.5	
Guztira		**310.5**

Taula 2.2. Ataza bakoitzean eta guztira sartutako orduen estimazioen taula.

Ataza	Estimatutako orduak	Sartutako orduak	Desbiderapena
F Formakuntza			
F1 Android Studio-n trebatu	10	15	+5
F2 Firebase-n trebatu	5	7	+2
F3 Eclipse-ko REST API zerbitzuetan trebatu	5	5	0
PK Proiektuaren kudeaketa			
PK1 Kudeaketa	30	46	+16
PK2 Plangintza	20	25	+5
G Garapena			
G1 Analisia	20	30	+10
G2 Diseinua	30	48	+18
G3 Inplementazioa	80	70	-10
G4 Probak	10	20	+10
D Dokumentazioa			
D1 Memoria	90	105	+15
A Aurkezpena			
A1 Aurkezpena prestatu	10	15	+5
A2 Aurkezpena egin	0.5	0.5	0
Guztira	**310.5**	**386.5**	**+76**

Taula 2.3. Atazen denbora estimazioen eta desbiderapenen taula.

2.3 Arriskuak

Proiektuaren garapenean hainbat ezusteko aurki daitezke eta hauek ahal den neurrian aurreikusten eta ekiditen saiatu beharko da. Izan ere, ezusteko hauek, kontuan hartzen ez diren atzerapenak eta gastuak sor ditzakete, eta hauek galerak dakartzate. Hori dela eta, arrisku hauek aurreikusiko dira eta hauek gertatuko balira ere, badaezpada arazo hauen aurrean jarraituko liratekeen pausuak zehaztuko dira.

2.3.1 Arriskuen zerrenda

- Teknologikoak
  - Teknologiaren ikaskuntzan zailtasunak. Teknologia berriak ikasteak arazoak eta atzerapenak ekar ditzake.
  - Ordenagailua izango da erabiliko den tresna eta tresna hau apur daiteke.
- Pertsonekin lotutakoak
  - Pertsonok akatsak egin ohi ditugu eta akats hauek ondorio desberdinak izan ditzakete.
  - Bakarkako proiektua izatean pertsona baten menpe dago proiektuaren garapen guztia. Hori dela eta, pertsona horri ezustekoren bat gertatzen bazaio edota gaixotzen bada atzerapenak sortuko ditu.
  - Proiektua hainbat zuzendariren begiztapean egongo da, eta zuzendariak ere atzerapenak sortu ahalko dituzte.
- Plangintzarekin lotutakoak
  - Gerta daiteke plangintza ez izatea errealista edota objektiboa eta hori dela eta ez iristea benetan planifikatu den punturaino.
  - Proiektu errealetan aurrekontuak daude, beraz proiektuaren plangintza zuzena ez bada aurrekontuak gaindi daitezke eta honek ondorio larriak ekar ditzake.

2.4 Arriskuei aurre egiten

Aipatutako arriskuei aurre egiteko estrategia desberdinak pentsatu beharko dira.

Alde batetik, arrisku teknologikoek denboran atzerapenak ekarri ahalko dituzte, hori dela eta, erreferentzia onak izaten saiatu beharko da, eta bestela foroetan galdetu beharko da. Horrez gain, ordenagailua apurtzen bada, proiektuaren azken bertsioaren kopia izatea komeniko da, beraz sarri egin beharko dira kopia hauek.

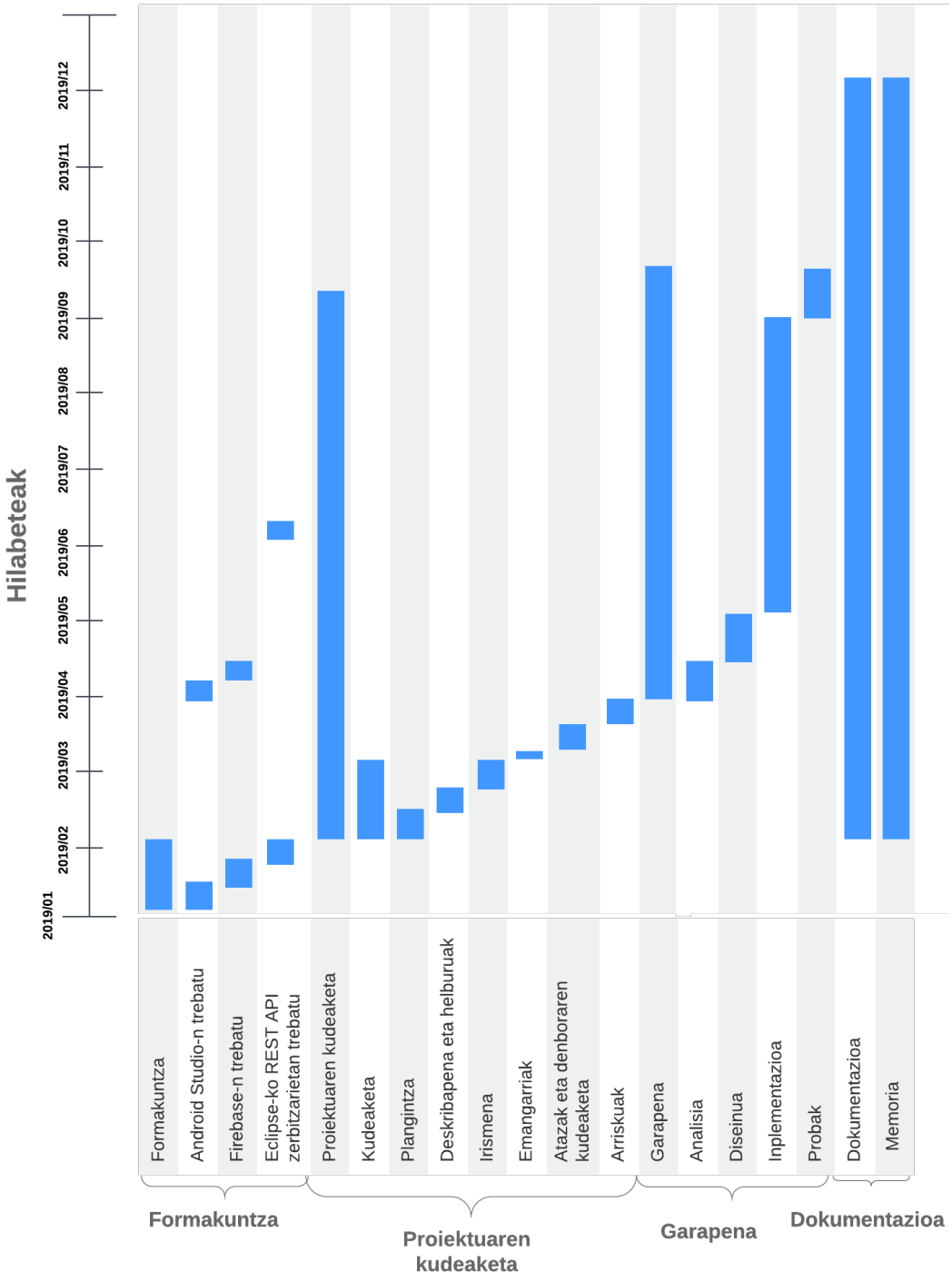
Bestalde, pertsonekin lotutako atzerapen batzuk ezin izango dira ekidin, baina beste batzuk minimizatzen saiatu beharko da. Esate baterako, pertsonei gerta dakizkien ezustekoak

ezin izango dira ekidin, eta atzerapen hauek onartu egin beharko dira. Baina pertsonek proiektuan egin ditzaketen akatsak zuzentzeko denbora minimizatzen saiatu beharko da. Hori dela eta, kopia eta bertsioak gordetzen joan beharko da, akatsen edo erabaki txarren aurrean "backup-ak izateko.

Azkenik, plangintzarekin lotutako arriskuei aurre egiten saiatzeko, ahalik eta plangintza errealistena egiten saiatu beharko da, eta proiektua entregatzeko epe nahiko malguak jarriko dira.

2.5 Gantt diagrama

2.2 irudian ikus daiteke proiekturako sortu den Gantt diagrama. Bertan grafikoki azaltzen da proiektuan zehar egingo den ataza bakoitzak hartuko duen denboraren estimazioa.



Irudia 2.2. Gantt diagrama.

3

Eskakizunen bilketa

Atal honetan garatuko den aplikazioaren deskribapen osoa azalduko da, hau da, aplikazioak izango dituen erabiltzaileak, funtzionalitateak, hauen deskribapenak, etab. Horretarako erabilpen kasuak, gertaeren fluxuak eta domeinuaren ereduak erabiliko dira.

3.1 Erabilpen kasuak

Aplikazioak izango dituen jardueren eta ekintzen deskribapenak azalduko dira. Horretarako, lehendabizi aktore desberdinen azalpenak emango dira.

3.1.1 Aktoreak

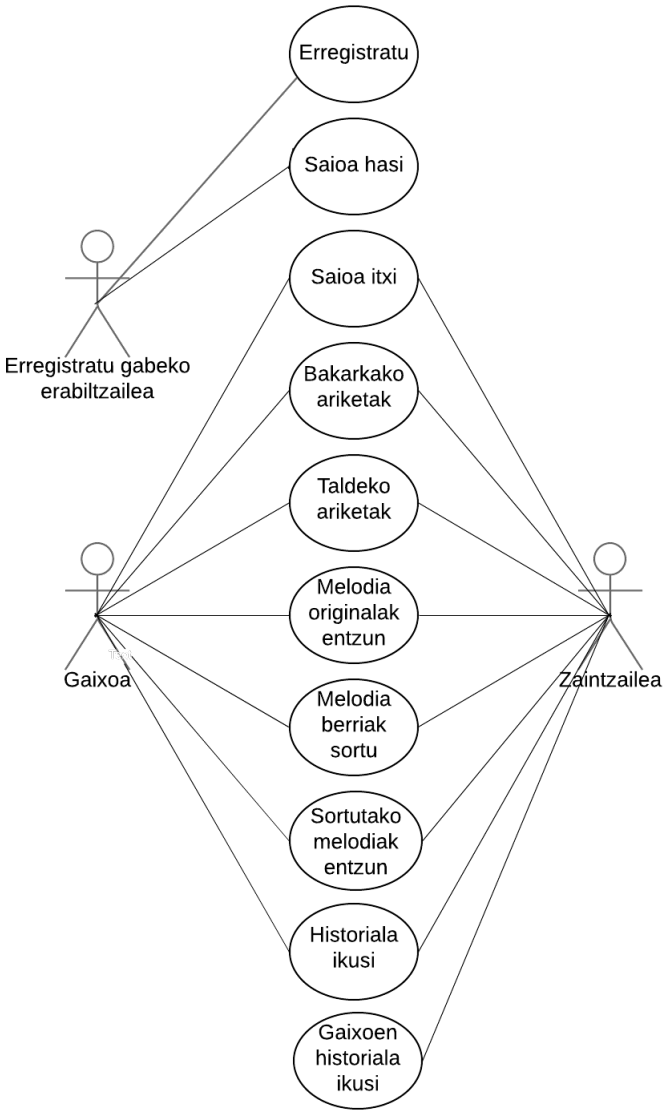
Hiru aktore mota desberdin egongo dira: alde batetik, erregistratu gabeko erabiltzailea izango da, eta beste aldetik, erregistratutako erabiltzaileen barruan, zaintzailea eta gaixoa egongo dira. Aktore mota bakoitzak erabilpen kasu desberdinak erabili ahalko ditu, baina zaintzaileak eta gaixoak hauetako hainbat partekatuko dituzte.

Hona hemen aktoreen deskribapenak:

- Erregistratu gabeko erabiltzailea: aplikazioa erabili ahal izateko erabiltzailea erregistratu egin beharko da. Hori dela eta, oraindik saioa hasteko edo erregistratzeko funtzioak soilik erabili ahalko ditu.
- Erregistratutako erabiltzailea: jada erregistratuta egongo den erabiltzailea. Hemen barruan beste bi motatan banatuko dira:
  - Zaintzailea: nolabaiteko administratzailea izango dela esan daiteke, baina ez du hainbeste indar izango. Taldeak sortu ahalko dituen erabiltzaile mota bakarra izango da, eta gainera beste erabilpen kasu gehienak erabili ahalko ditu.
  - Gaixoa: aktore mota ohikoena izango da eta erabilpen kasu gehienak erabili ahalko ditu.

3.1.2 Aktoreak eta erabilpen kasuak

Aktore motak definituta, eta gutxi gorabeherako ideia bat eginda, 3.1 irudian erakutsiko dira zehatz mehatz aktore bakoitzak egin ditzakeen erabilpen kasuak.



Irudia 3.1. Aktoreak eta erabilpen kasuak.

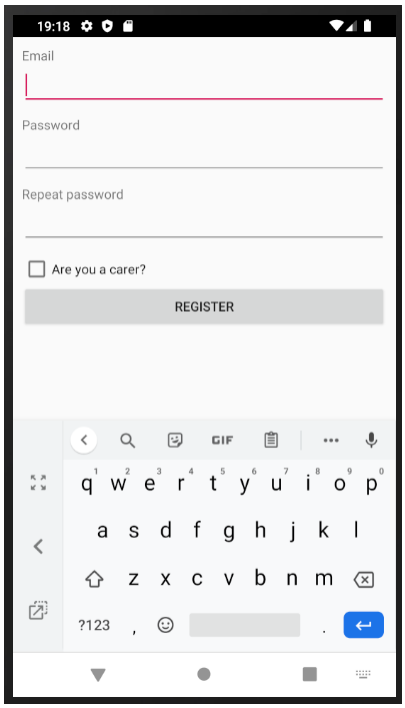
3.2 Gertaera fluxuak

Atal honetan erabilpen kasuen gertaera fluxu nagusiak eta alternatiboak azalduko dira. Hau da, erabilpen kasu bakoitzerako, hauen funtzionamendua hobeto ulertu ahal izateko, urratsez urrats azalduko dira erabiltzaileak eta sistemak jarraituko dituzten pausuak. Pauso hauei zenbakiak jarriko zaizkie, horrela, fluxu alternatiboetan, berriro pausu guztiak idatzi beharrean zenbaki horretatik aurrerakoa aldatuko dela adierazteko.

Horrez gain, paperezko prototipoak ere sortu ohi dira ideia bat egin eta ondoren aplikazioaren bistak sortzeko, baina txukunago eta ulerkorrago geratzeko asmoz, hemen gehituko diren irudiak jada aplikazioaren amaierako diseinukoak izango dira.

3.2.1 Erregistratu

Erabiltzaileak aplikazioan erregistratzeko erabilpen kasuaren gertaera fluxua azalduko da. Ikusi 3.2 irudia hobeto ulertu ahal izateko.



Irudia 3.2. Erregistratu gertaera fluxuaren bista.

3.2.1.1 Fluxu nagusia

1. **Erabiltzailea:** e-posta, pasahitza, pasahitz errepikatua eta zaintzailea den edo ez bete eta erregistratzeko botoiari klik egiten dio.
2. **Sistema:** datuak jaso eta guztia zuzen dagoela ikusita, erabiltzailea sisteman erregistratzen du eta saioa hasteko orrialdera bidaltzen du.

3.2.1.2 Fluxu alternatiboa: erabiltzailea jada existitzen da

2. **Sistema:** datuak jaso eta erregistroa gauzatzean, e-posta hori duen erabiltzailea jada existitzen dela ohartzen da eta erabiltzaileari jakinarazten dio.

3.2.1.3 Fluxu alternatiboa: e-posta okerra

2. **Sistema:** datuak jaso eta egiaztatzean e-postaren formatua okerra dela ikusten du eta erabiltzaileari adierazten dio.

3.2.1.4 Fluxu alternatiboa: pasahitz desberdinak

2. **Sistema:** datuak jasotzean, pasahitzak desberdinak direla ikusi eta erabiltzaileari jakinarazten dio.

3.2.1.5 Fluxu alternatiboa: pasahitz motzegia

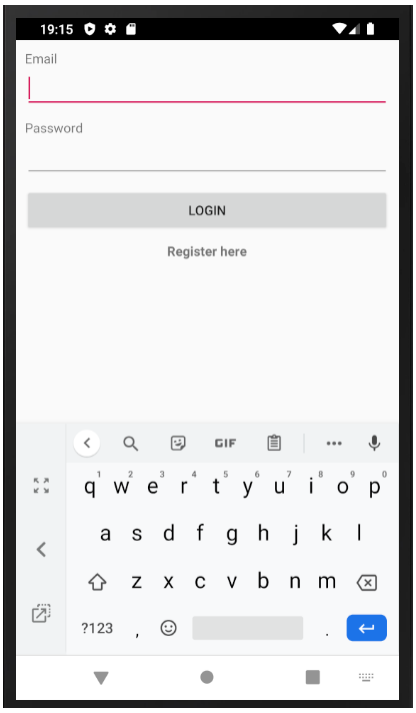
2. **Sistema:** datuak jaso eta egiaztatzean pasahitza motzegia dela ikusten du eta erabiltzaileari adierazten dio.

3.2.2 Saioa hasi

Erabiltzaileek aplikazioaren funtzionalitate gehienak erabili ahal izateko saioa hasi beharra daukate, eta gertaera fluxu honetan erabilpen kasu hori azalduko da. Ikusi 3.3 irudia.

3.2.2.1 Fluxu nagusia

1. **Erabiltzailea:** e-posta eta pasahitza sartzen ditu eta saioa hasteko botoia klikatzen du.
2. **Sistema:** erabiltzailearen datuak zuzenak direla egiaztatu eta orrialde nagusira bidaltzen du.



Irudia 3.3. Saioa hasi gertaera fluxuaren bista.

3.2.2.2 Fluxu alternatiboa: emaila edota pasahitza hutsik daude

2. **Sistema:** jasotzen dituen datuak edo datuetako bat hutsik dago, beraz erabiltzaileari jakinarazten dio.

3.2.2.3 Fluxu alternatiboa: emaila edota pasahitza ez dira zuzenak

2. **Sistema:** jasotako datuak egiaztatzean, okerrak direla ohartzen da eta erabiltzaileari jakinarazten dio.

3.2.3 Saioa itxi

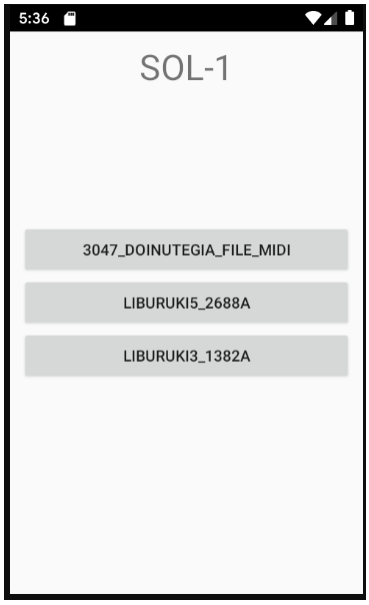
Erabiltzaileek saioa ixteko erabilpen kasuaren gertaera fluxua azalduko da jarraian.

3.2.3.1 Fluxu nagusia

1. **Erabiltzailea:**: erabiltzailea orrialde nagusian dago eta saioa itxi botoia klikatzen du.
2. **Sistema:** erabiltzailearen saioa ixten du eta saioa hasteko orrialdera bideratzen du.

3.2.4 Bakarkako ariketak: melodia originala asmatu

Gaixoek nahiz zaintzaileek egin ahalko dituzte bakarkako ariketak, eta hemen melodia originalak asmatzearen gertaera fluxua azalduko da. Aplikazioan nolakoa izango den ikusteko, ikusi 3.4 irudia.



Irudia 3.4. Bakarkako ariketak: melodia originala asmatu gertaera fluxuaren bista.

3.2.4.1 Fluxu nagusia

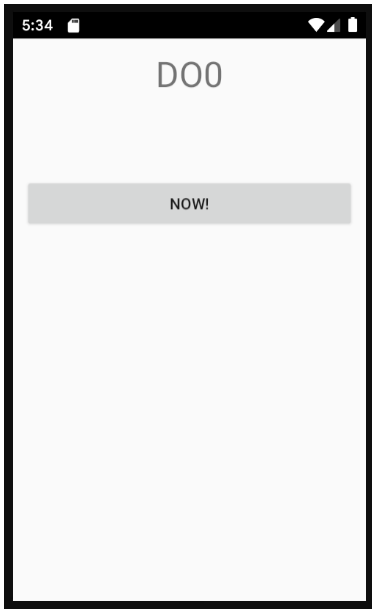
1. **Erabiltzailea:** erabiltzaileak melodia asmatzeko ariketa mota aukeratuko du.
2. **Sistema:** ariketa mota jaso eta midi-en zerrenda eskuratuko du eta REST API zerbitzaritik midi-en zerrenda lortuko du. Ondoren ausaz bat aukeratuko du eta erreproduzitzeko orrialdera eramango du. Hemen REST API zerbitzaritik dagokion midi-aren eraldatutako melodia lortu eta erreproduzitzailea prestatuko du.
3. **Erabiltzailea:** bisualizatzeko modua aukeratu eta PLAY botoia klikatuko du.
4. **Sistema:** melodia erreproduzitzen hasiko da eta erabiltzaileari ausaz aukeratutako melodia desberdinen botoiak erakutsiko zaizkio. Hauetako batek eraldatutako melodia originalaren midi-aren izena izango du.
5. **Erabiltzailea:** midi-en izena duten botoietako bat klikatuko du.
6. **Sistema:** erabiltzaileari ariketa asmatu duen edo ez adieraziko zaio eta Firebase-ko datu basean gordeko da emaitza.

3.2.4.2 Fluxu alternatiboa: erabiltzaileak ez du botoirik klikatu

6. **Sistema:** erreprodukzioa amaitzean gaizki eginda dagoela bezala hartzen du eta Firebase-ko datu basean gordetzen du.

3.2.5 Bakarkako ariketak: melodia aldatzen hasten den unea asmatu

Beste bakarkako ariketa mota melodia aldatzen hasten den unea asmatzearena izango da, eta jarraian bere gertaera fluxua azalduko da. Bestalde ikusi 3.5 irudia ondoren aplikazioan izango duen itxura ikusi ahal izateko.



Irudia 3.5. Bakarkako ariketak: melodia aldatzen hasten den unea asmatu gertaera fluxuaren bista.

3.2.5.1 Fluxu nagusia

1. **Erabiltzailea:** erabiltzaileak melodiaren aldaketa asmatzeko ariketa mota aukeratuko du.
2. **Sistema:** ariketa mota jaso eta midi-ak aukeratzeko orrialdea erakutsiko du. Ondoren REST API zerbitzaritik midi-en zerrenda lortu eta bistaratuko du.
3. **Erabiltzailea:** midi-a aukeratuko du.
4. **Sistema:** erreprodukzio orrialdera eramango du eta bertan REST API zerbitzaritik midi eraldatua eskuratuko du eta erreproduzitzailea prestatuko du.

5. **Erabiltzailea:** bisualizatzeko modua aukeratu eta PLAY botoia klikatuko du.
6. **Sistema:** melodia erreproduzitzen hasiko da eta erabiltzaileari NOW botoia erakutsiko zaio.
7. **Erabiltzailea:** NOW botoia klikatuko du.
8. **Sistema:** erabiltzaileak izan duen atzerapena erakutsiko du eta Firebase-n gordeko du emaitza.

3.2.5.2 Fluxu alternatiboa: erabiltzaileak ez du NOW botoia klikatu

8. **Sistema:** erreprodukzioa amaitu arte zain egongo da eta ondoren ariketa gaizki eginda dagoela bezala hartzen du eta Firebase-ko datu basean gordetzen du.

3.2.6 Taldeko ariketak

Taldeko ariketa erabilpen kasuaren gertaera fluxua azalduko da hemen. Ikusi 3.6 irudia.

Irudia 3.6. Taldeko ariketak gertaera fluxuaren bista.

3.2.6.1 Fluxu nagusia

Bi erabiltzaile mota desberdinek hartuko dute parte erabilpen kasu honetan, beraz bereizi egingo dira:

A. Zaintzaileak

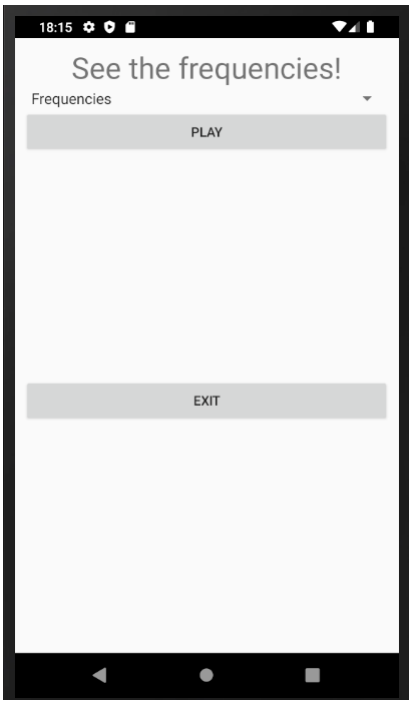
1. **Sistema:** online dauden gaixoen zerrenda erakutsiko da.
2. **Zaintzailea:** nahi dituen gaixoak aukeratu eta bere taldean sartuko ditu. Behin gaixoak aukeratuta CREATE ROOM botoia klikatuko du.
3. **Sistema:** gaixoen izenekin talde bat sortu eta Firebase-n gordeko du. Ondoren ariketa mota aukeratzeko pantailara bidaliko du.
4. **Zaintzailea:** nahi duen ariketa mota aukeratuko du.
5. **Sistema:** ariketa mota Firebase-n gorde eta midi-en zerrenda erakutsiko den orrialdea bistaratuko du. Bertan, REST API zerbitzaritik midi-ak eskuratu eta erakutsiko ditu.
6. **Zaintzailea:** zerrendako midi bat aukeratuko du.
7. **Sistema:** midi-a ere Firebasen gordeko du eta erreproduzitzeko orrialdea erakutsiko du. Firebase-n gordetako datuekin, dagokion midi eraldatua lortuko du REST API zerbitzaritik eta erreproduzitzailea prestatuko du.
8. **Zaintzailea:** PLAY botoia klikatuko du.
9. **Sistema:** gaixoen mugikorrak sinkronizatu eta segundu gutxi batzuetara erreproduzitzen hasiko da.

B. Gaixoak

1. **Sistema:** gaixoa talde batera gehitzean, itxaron gela bat erakutsiko da. Ondoren jada zaintzaileak ariketa mota eta midi-a aukeratzen dituenean erreprodukzioa orrialdera bidaliko da. Hemen Firebase-tik eskuratuko diren datuekin midi bat lortuko da REST API zerbitzaritik eta erreproduzitzailea prestatuko da.
2. **Gaixoa:** bisualizazio modua aukeratuko du.
3. **Sistema:** zaintzaileak PLAY ematean, gaixoen mugikorrak sinkronizatuko dira eta segundu gutxi barru, ariketa motaren arabera, aukera desberdinak bistaratuko dira eta erreprodukzioa hasiko da.
4. **Gaixoa:** ariketa gauzatuko du.
5. **Sistema:** gaixoaren emaitzak pantailaratuko dira eta Firebase-ko datu basean gordeko dira.

3.2.7 Melodia originalak entzun

Melodia originalen gertaera fluxua azalduko da azpian. Aplikazioan izango duen itxura ikusi ahal da 3.7 irudian.



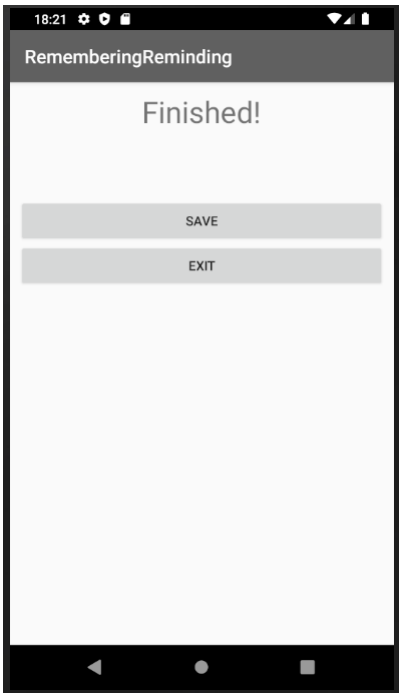
Irudia 3.7. Melodia originalak entzun gertaera fluxuaren bista.

3.2.7.1 Fluxu nagusia

1. **Sistema:** REST API zerbitzaritik melodia originalen zerrenda eskuratu eta pantailaratzen du.
2. **Erabiltzailea:** melodia originalen zerrendatik bat aukeratzen du.
3. **Sistema:** erreprodukzioa orrialdea erakutsiko du, eta REST API zerbitzaritik dagokion midi-a eskuratzean, erreproduzitzailea prestatuko du. Behin erreproduzitzailea prestatuta PLAY botoia erakutsiko du.
4. **Erabiltzailea:** bisualizatzeko modua aukeratu eta PLAY-ri emango dio.
5. **Sistema:** midi-a erreproduzitzen hasiko da.

3.2.8 Melodia berriak sortu

Melodia berriak sortzeko erabilpen kasuaren gertaera fluxua azalduko da jarraian. Ikusi 3.8 irudia.



Irudia 3.8. Melodiak berriak sortu gertaera fluxuaren bista.

3.2.8.1 Fluxu nagusia

1. **Sistema:** REST API zerbitzaritik melodia originalen zerrenda eskuratuko du eta erabiltzaileari erakutsiko dio.
2. **Erabiltzailea:** midi-en zerrendatik bat aukeratuko du.
3. **Sistema:** erreprodukzio orrialdera joango da. Hemen REST API zerbitzariari, dagokion eraldatutako melodiaren eskaera egiten dio. Orduan REST API zerbitzariak melodia berria sortuko du eta aplikazioari bidaliko dio. Ondoren, aplikazioak, eskuratutako midi horrekin erreproduzitzailea prestatuko du, eta behin prestaketa amaituta PLAY botoia bistaratuko du.
4. **Erabiltzailea:** PLAY botoia klikatuko du.

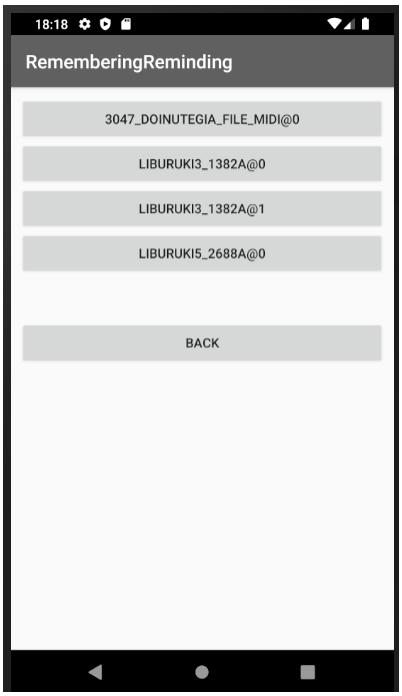
5. **Sistema:** melodiaren erreprodukzioa hasiko da. Ondoren, erreprodukzioa amaitzen denean, SAVE botoia bistaratuko da.
6. **Erabiltzailea:** SAVE botoia klikatzen du.
7. **Sistema:** erreproduzitu berri duen melodia REST API zerbitzarian gordeko du.

3.2.8.2 Fluxu alternatiboa: erabiltzaileak ez du melodia gordetzen

6. **Erabiltzailea:** ez zaio sortutako melodia gustatu eta BACK botoia klikatzen du SAVE klikatu beharrean.
7. **Sistema:** midi-a aukeratzeko orrialdea erakutsiko da beste saiakerarik egin nahi badu.

3.2.9 Sortutako melodiak entzun

Sortutako melodiak entzuteko erabilpen kasuaren gertaera fluxua. 3.9 irudia aplikazioan izango duen itxura ikusi ahal da.



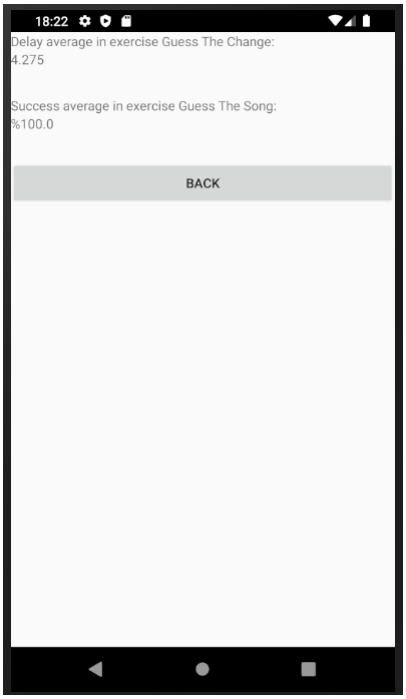
Irudia 3.9. Sortutako melodiak entzun gertaera fluxuaren bista.

3.2.9.1 Fluxu nagusia

1. **Sistema:** REST API zerbitzaritik sortutako melodien zerrenda eskuratu eta pantailaratzen du.
2. **Erabiltzailea:** sortutako melodien zerrendatik bat aukeratzen du.
3. **Sistema:** erreprodukzioa orrialdea erakutsiko du, eta REST API zerbitzaritik dagokion midi-a eskuratzean, erreproduzitzailea prestatuko du. Behin erreproduzitzailea prestatuta PLAY botoia erakutsiko dio.
4. **Erabiltzailea:** bisualizatzeko modua aukeratu eta PLAY-ri emango dio.
5. **Sistema:** midi-a erreproduzitzen hasiko da.

3.2.10 Historiala ikusi

Historiala ikusi gertaera fluxua azalduko da azpian. Ikusi 3.10 hobeto ulertu ahal izateko.



Irudia 3.10. Historiala ikusi gertaera fluxuaren bista.

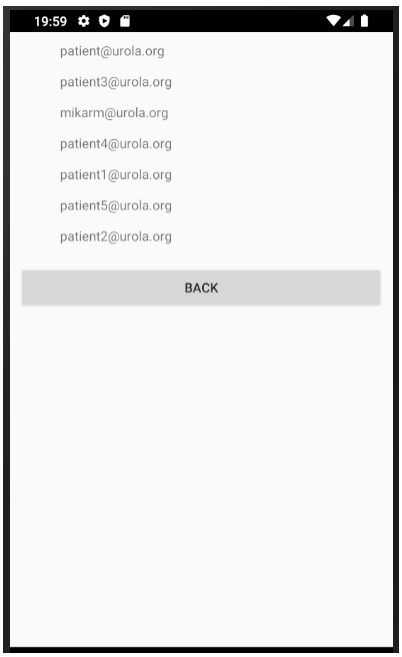
3.2.10.1 Fluxu nagusia

1. **Erabiltzailea:** estatistikak ikusteko aukera klikatuko du.

2. **Sistema:** erabiltzailearen estatistikak eskuratuko dira Firebase-ko denbora errealeko datu basetik. Ondoren bataz bestekoak kalkulatu eta pantailaratuko dira.

3.2.11 Gaixoen historiala ikusi

Gaixoen historiala ikusi gertaera fluxua azalduko da. 3.11 irudian ikus daiteke gaixoen zerrenda nola erakusten den.



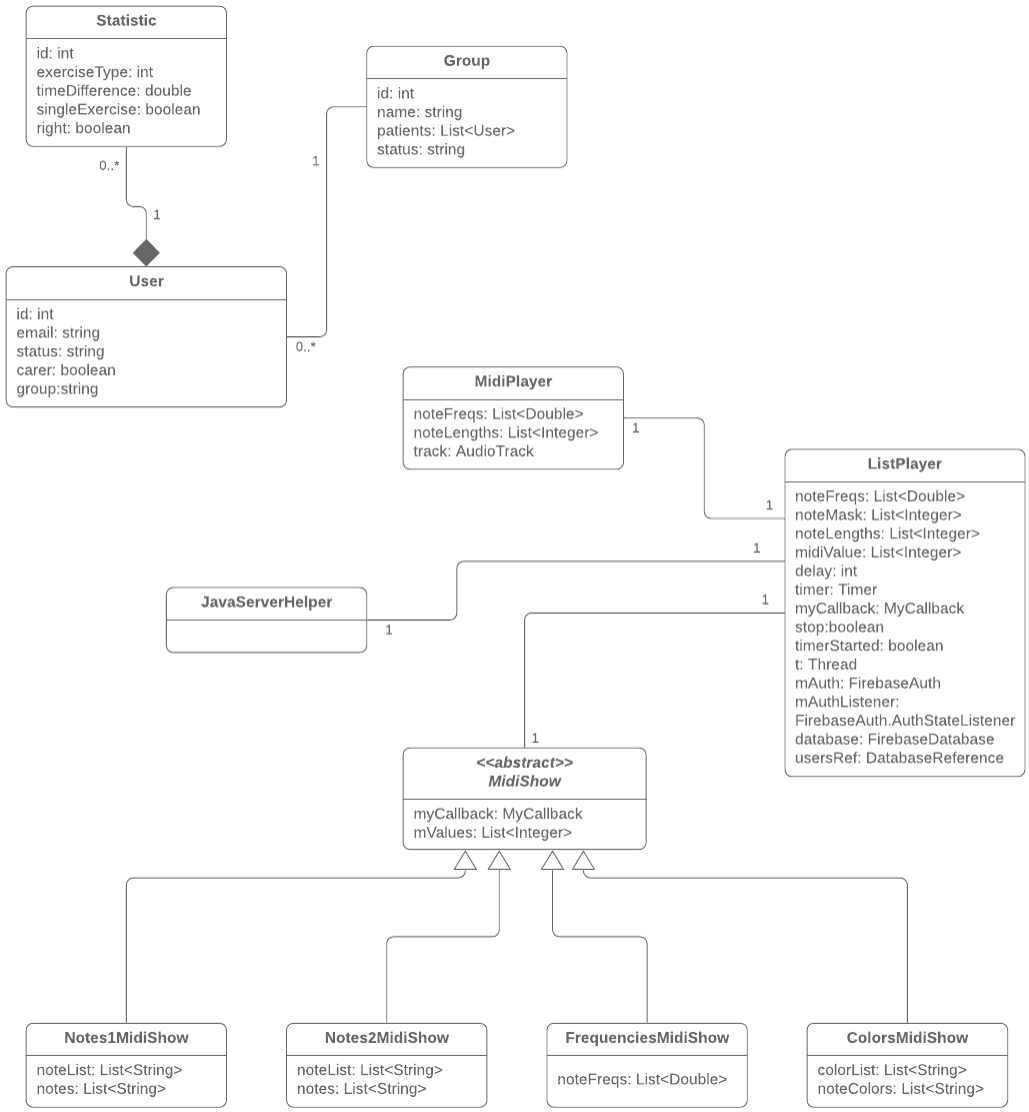
Irudia 3.11. Gaixoen historiala ikusi gertaera fluxuaren bista.

3.2.11.1 Fluxu nagusia

1. **Zaintzailea:** estatistikak ikusteko aukera klikatuko du.
2. **Sistema:** Firebase-tik gaixoen zerrenda lortuko du eta zaintzaileari erakutsiko dio.
3. **Zaintzailea:** gaixoren baten izena klikatuko du.
4. **Sistema:** aukeratutako gaixoaren estatistikak eskuratuko dira Firebase-ko denbora errealeko datu basetik. Ondoren bataz bestekoak kalkulatu eta pantailaratuko dira.

3.3 Domeinu eredua

Azpi atal honetan, entitateen eta hauen arteko harremanen azalpenak emango dira, besteak beste, hauen atributuak eta loturak azalduz. 3.12 irudia erabiliko da funtsezko kontzeptuak eta entitateak irudikatzeko eta hauek hobeto ulertu ahal izateko.



Irudia 3.12. Domeinuen eredua.

Jarraian, domeinu ereduko klase bakoitzaren azalpen labur bat emango da errazago ulertu ahal izateko:

3.3.1 Statistic

Statistic klaseak User klasearekin izango du lotura, izan ere, erabiltzaile bakoitzak bere estatistikak izango baititu. Klase honek erabiltzaileek ariketetan lortutako emaitzak gordetzeko balioko du.

3.3.2 User

Erabiltzaileak gordetzeko erabiliko da klase hau eta Statistic motako hainbat elementu gordeko ditu bere barnean. Horrez gain, Group motako elementuren batekin lotura izango du, izan ere erabiltzaileei taldeak esleituko baitzaizkie.

3.3.3 Group

Taldeko ariketetan taldeak gordetzeko erabiliko da, eta talde hauek User motako elementuak izango dituzte bere barruan. Klase hau denbora errealean taldeko ariketak egiteko erabiliko da.

3.3.4 ListPlayer

ListPlayer klasea aplikazioko erreproduzitzailea izango da. Klase honek MidiPlayer klasearekin izango du lotura, izan ere bigarren hau erabiliko baita midi-ko tonuak erreproduzitzeko. Horrez gain, JavaServerHelper motako elementu bat ere izango du REST API zerbitzaritik datuak lortzeko. Azkenik, midi-en melodien notak erakusteko MidiShow izeneko klasearekin izango du lotura.

3.3.5 MidiPlayer

Goian aipatu bezala, MidiPlayer klaseak ListPlayer klasearekin izango du lotura, hau da, ListPLayer-ak MidiPlayer klasearen instantzia bat izango du. Klase honek sortuko ditu melodiaren tonuak.

3.3.6 MidiShow

MidiShow klasea erabiliko da erreprodukzioan melodiak bisualizatzen laguntzeko. Klase hau abstraktua da, eta ondorengo Notes1MidiShow, Notes2MidiShow, FrequenciesMidiShow eta ColcorsMidiShow klaseak, klase abstraktu honen herentziak dira, horrela klase honen atributuak heredatuz. Horrez gain, lehen aipatu bezala ListPlayer klasearekin lotura izango du.

3.3.7 Notes1MidiShow

MidiShow klasearen herentzia izango da, hau da, bere atributu eta metodo guztiak izango ditu. Klase honek notazio sistema latinoan erakutsiko ditu notak.

3.3.8 Notes2MidiShow

MidiShow klasearen metodo eta atributu guztiak heredatuko ditu. Klase honek midi-aren notak notazio sistema ingelesean erakusteko.

3.3.9 FrequenciesMidiShow

MidiShow klasearen herentzia honek bere atributu eta metodo guztiak heredatuko ditu. Midi-aren notak frekuentzietan bistaratzeko.

3.3.10 ColorsMidiShow

MidiShow klasea heredatuko du bere metodo eta atributu eta guzti, eta midi-en notak bistaratuko ditu koloreetan.

3.3.11 JavaServerHelper

Klase hau ListPlayer-ekin lotuta egongo da eta REST API zerbitzaritik datuak eskuratzea ahalbidetuko dio.

4

Diseinua

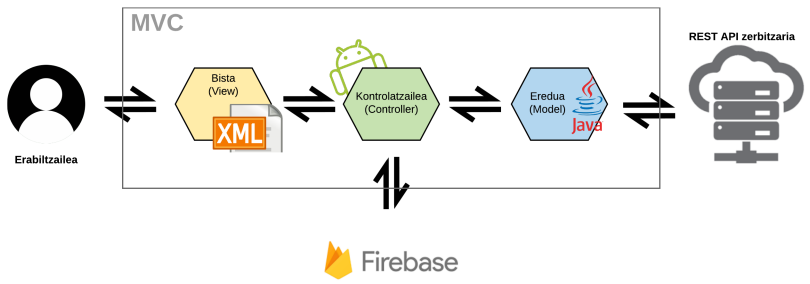
Diseinuaren zatian, aplikazioaren eta zerbitzarien azalpen zehatzak emango dira. Hasteko, arkitektura orokorra azalduko da eta, ondoren aplikazioaren zati zehatzagoak azaltzeaz gain, zerbitzarien zehaztapenak ere emango dira. Bestalde, atal honetan egongo dira erailpen kasu batzuen sekuentzia diagramak ere.

4.1 Arkitektura

Mugikorretarako aplikazio gehienak bezala, garatuko den aplikazioak MVC (ingelesez, Model-View-Controller, eta euskaraz, Eredua-Bista-Kontrolatzailea) [10] arkitektura jarraituko du. Hori horrela izanda, zati bakoitzaren azalpen bat emango da, baina lehendabizi, aplikazioaren funtzionamendua nondik norakoak ulertu ahal izateko, hona hemen azalpen labur bat: erabiltzailea bistaren zatiarekin komunikatuko da eta hau kontrolatzailearekin lotuta egongo da. Ondoren, kontrolatzaileak jasotako informazioarekin erabaki desberdinak hartuko ditu, eta batzuetan eredu geruzatik datuak eskuratu beharko ditu. Izan ere, eredu geruzaren arduretako bat datuak eskuratzea izango baita. Gure kasuan hiru zati horiez gain, beste bi gehitu beharko ditugu, Firebase [11] zerbitzaria eta Eclipse-n [7] garatutako REST API [9] zerbitzaria. Hauetan datuak eta fitxategiak gordeko ditugu, baina REST API zerbitzaria midi berriak sortzeko gai ere izango da liburutegi batzuei esker. Ikusi 4.1 irudia hobeto ulertzeko.

4.1.1 Eredua

Ereduaren geruza datuak eskuratzeaz, gordetzeaz eta moldatzeaz arduratuko da. Kasu honetan, REST API zerbitzari batetik eskuratuko dira midi-ak, eta hain zuzen, ereduen geruzako JavaServerHelper klase izango da honetaz arduratuko dena. Horretaz gain, geruza honetan Java klase desberdinak egongo dira lortutako objektuak gordetzeko balioko dutenak.



Irudia 4.1. Aplikazioaren arkitektura azaltzen duen irudia.

Kontrolatzailearen geruzatik instantziatuko dira hemengo klaseak, eta bete beharreko aginduak ere bertatik jasoko dira.

4.1.2 Bista

Geruza hau aplikazioaren interfazeaz arduratuko da, eta aplikazioa Android Studio-n garatuko denez, bista XML fitxategiz osatuta egongo da. Fitxategi hauek estatikoak izango dira, baina aplikazioaren hainbat zatietan datuak dinamikoki erakutsi beharko direnez, fitxategi hauetan elementu dinamikoak sartuko dira. Kasu honetan, RecyclerView [12] motako elementuak erabili beharko dira horretarako, hauetan elementuak dinamikoki gehituko edo kenduko dira, eta aldaketa hauek interfazean azalduko dira.

4.1.3 Kontrolatzailea

Kontrolatzailearen geruzako klaseak, bistatik jasotakoa interpretatzeaz eta erantzun bat emateaz arduratzen dira. Bistako gertaerak jasotzeko "Listener"motako funtzio batzuk egongo dira uneoro entzuten, eta esate baterako, funtzio hauek identifikatu ahalko dute ea botoiren bat sakatua izan den. Erantzun hauek bistak aldarazi ahalko dituzte, edo datu geruzako datuekin eragiketak egin ahalko dituzte.

Horrez gain, kasu honetan, Firebase-rekin [11] lan egingo denez, maila hau arduratuko da Firebase-rekin komunikatzeaz. Hau da, geruza honetan eskuratuko dira Firebase-ko datuak, eta hemendik gorde edo eguneratuko dira. Ardura hau eredu geruzan ere enkapsulatu zitekeen, baina kontrolatzailetik zuzenean atzitzea erabaki da Android [13] aplikazioetan ohiko eginbidea baita, alegia, Firebase hain da erabilia ezen modulu bereizi moduan hartu ohi dela (ikus 4.1 irudia).

4.2 Datuen egitura

Proiektu honetan bi datu-base desberdin erabiliko dira. Alde batetik, Firebase-ko datu basea erabiliko da objektu desberdinak gordetzeko. Bestalde, Firebase-n zerbitzariaren aldean koderik ezin denez idatzi, fitxategiak gordetzeko Eclipse-n REST API zerbitzari bat jarriko da martxan.

4.2.1 Firebase

Firebase-k [11] eskaintzen duen datu base honetan hainbat datu desberdin gordeko dira. Alde batetik erabiltzaileak kudeatzeko funtzio desberdinak eskaintzen ditu Firebase Authentication [14] zerbitzuaren bidez, eta proiektu honetan funtzio horiek erabiliko dira erabiltzaile berriak erregistratzeko, saioak hasteko eta saioa ixteko. 4.2 irudian ikus daiteke erabiltzaile bat erregistratzean automatikoki sortuko diren datu motak.

Identificador	Proveedores	Creado	Accediste a tu cuenta	UID de usuario ↑
patient@urola.org	✉	3 jul. 2019	2 dic. 2019	4aKkUHZ0blOdPpgKP3UOEhrC6vy2
patient3@urola.org	✉	4 jul. 2019	4 jul. 2019	I1PSUH79QLdK9GX4s3IpQnT9jrB2
patient4@urola.org	✉	4 jul. 2019	4 jul. 2019	Zh7wzWMiDoX4qgyC4jBYtBmCEq...
patient1@urola.org	✉	3 jul. 2019	3 jul. 2019	fYqyyOWoqkWLBGvf7ZQKRQ81IP...
patient5@urola.org	✉	4 jul. 2019	4 jul. 2019	njqn2TtDKtd7oZX39XtcbaZkIxm1
carer@urola.org	✉	3 jul. 2019	12 nov. 2019	t3Bg6IgnJVXoHTi4wY9k5ucrMoA2

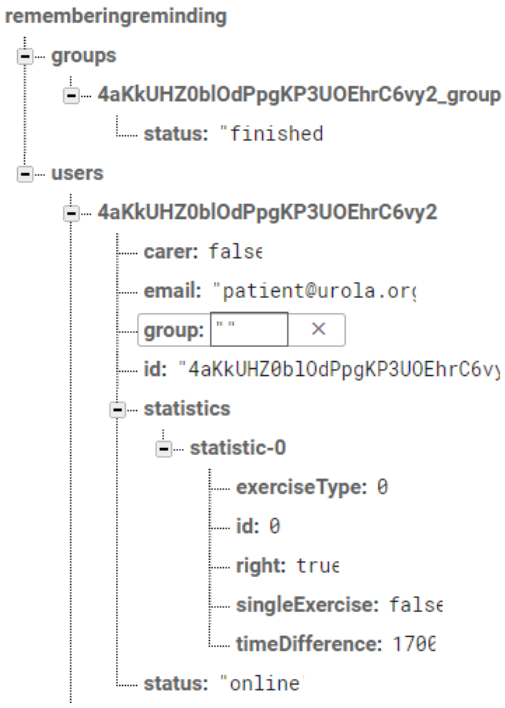
Irudia 4.2. Firebase-ko erabiltzaileei automatikoki gordetzen zaizkien datuak.

Bestalde, Firebase Realtime Database [15], denbora errealeko datu basea, erabiliko da erabiltzaileen datu gehigarriak (estatistikak, zaintzailea den edo ez...) gordetzeko eta taldeko ariketen datu desberdinak gordetzeko.

Datu base hau NoSQL motakoa izango da eta bertan datuak JSON [16] moduan gordeko dira. 4.3 irudian ikus daiteke nolako formatua izango duen proiektu honen datu baseak.

4.2.2 REST API zerbitzaria

Eclipse-n garatutako Dynamic Web Project motako REST API zerbitzaria fitxategiak gordetzeko erabiliko da, eta hemen liburutegi bat erabiliko da melodia berriak sortzeko.



Irudia 4.3. Firebase-ko Realtime Databaseko datuen egitura.

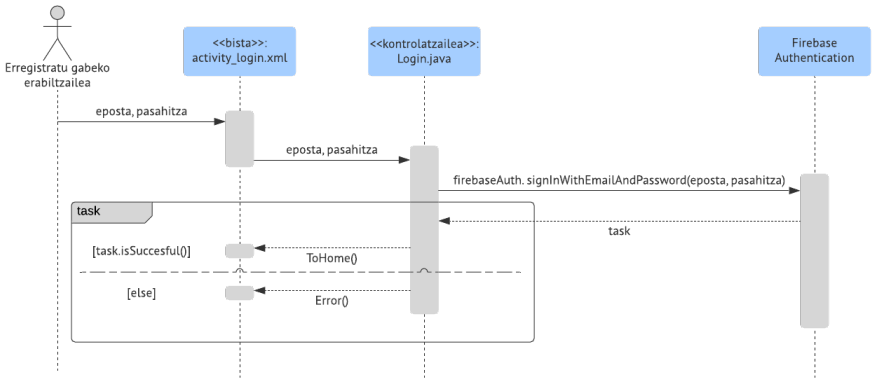
Zerbitzari honek REST API motako hainbat zerbitzu eskainiko ditu, eta beti bertan dauden fitxategietatik eskuratuko ditu datuak edo melodiak. Hau da, esan daiteke ez duela benetako datu base bat izango, fitxategiak bakarrik gordeko baitira bertan.

4.3 Sekuentzia diagramak

Erabilpen kasu guztien sekuentzia diagramak egitea gehiegizkoa litzateke, beraz, garrantzitsuenak aukeratu dira: *saioa hasi*, *melodia berriak sortu*, *bakarkako ariketak* eta *taldeko ariketak* . Jarraian erakutsiko dira aukeratutako erabilpen kasuen sekuentzia diagramak:

4.3.1 Saioa hasi

Saio hasi funtzionalitatea hobeto ulertu ahal izateko, erabilpen kasu honen sekuentzia diagrama ikusi ahal da 4.4 irudian.



Irudia 4.4. Login gertaera fluxuaren sekuentzia diagrama.

4.3.2 Melodia berriak sortu

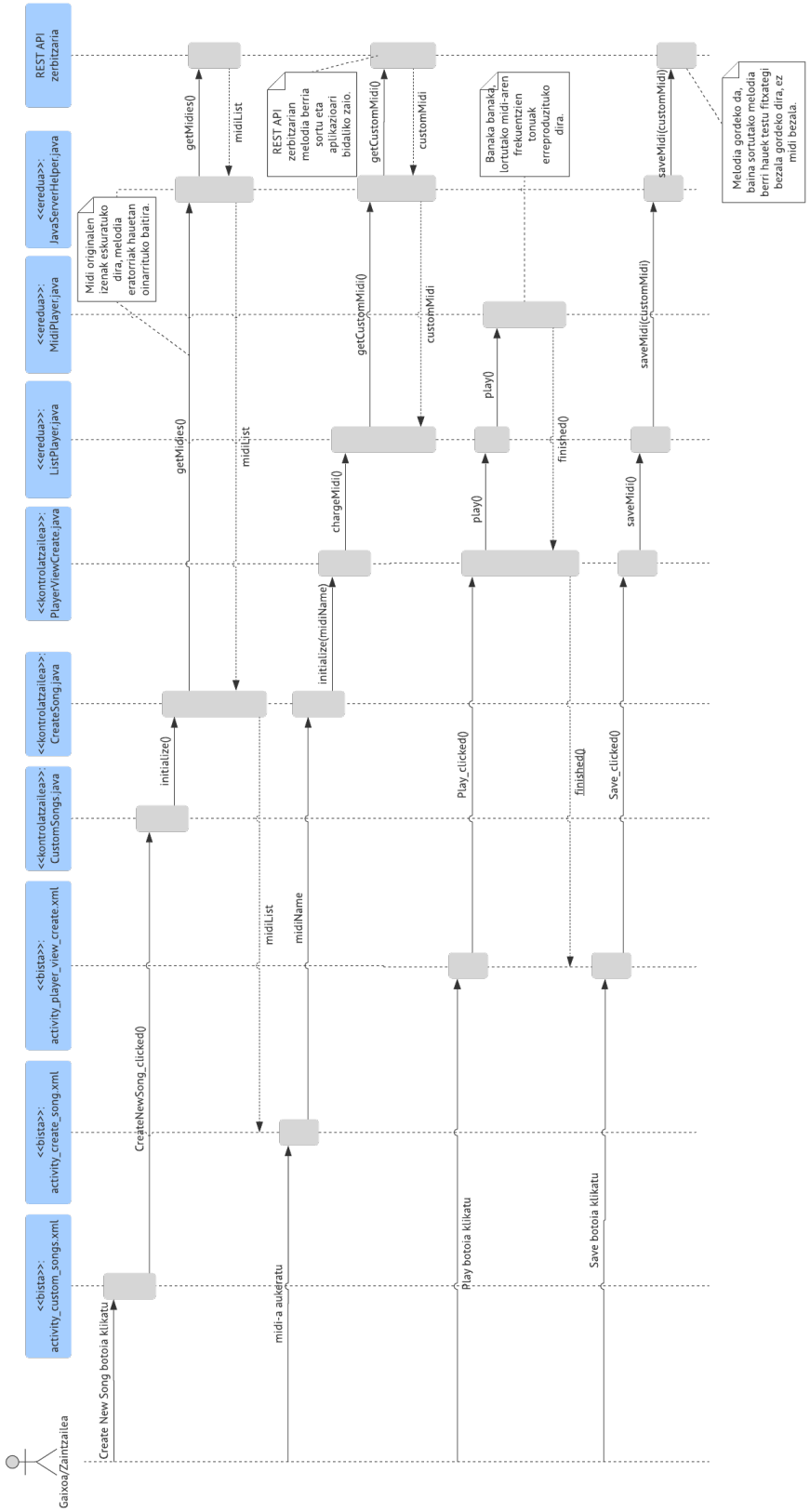
4.5 irudian melodia berriak sortu funtzionalitatearen sekuentzia diagrama ikus daiteke. Bertan, melodia berria sortzeko prozesua azalduko da pausoz pauso.

4.3.3 Bakarkako ariketak

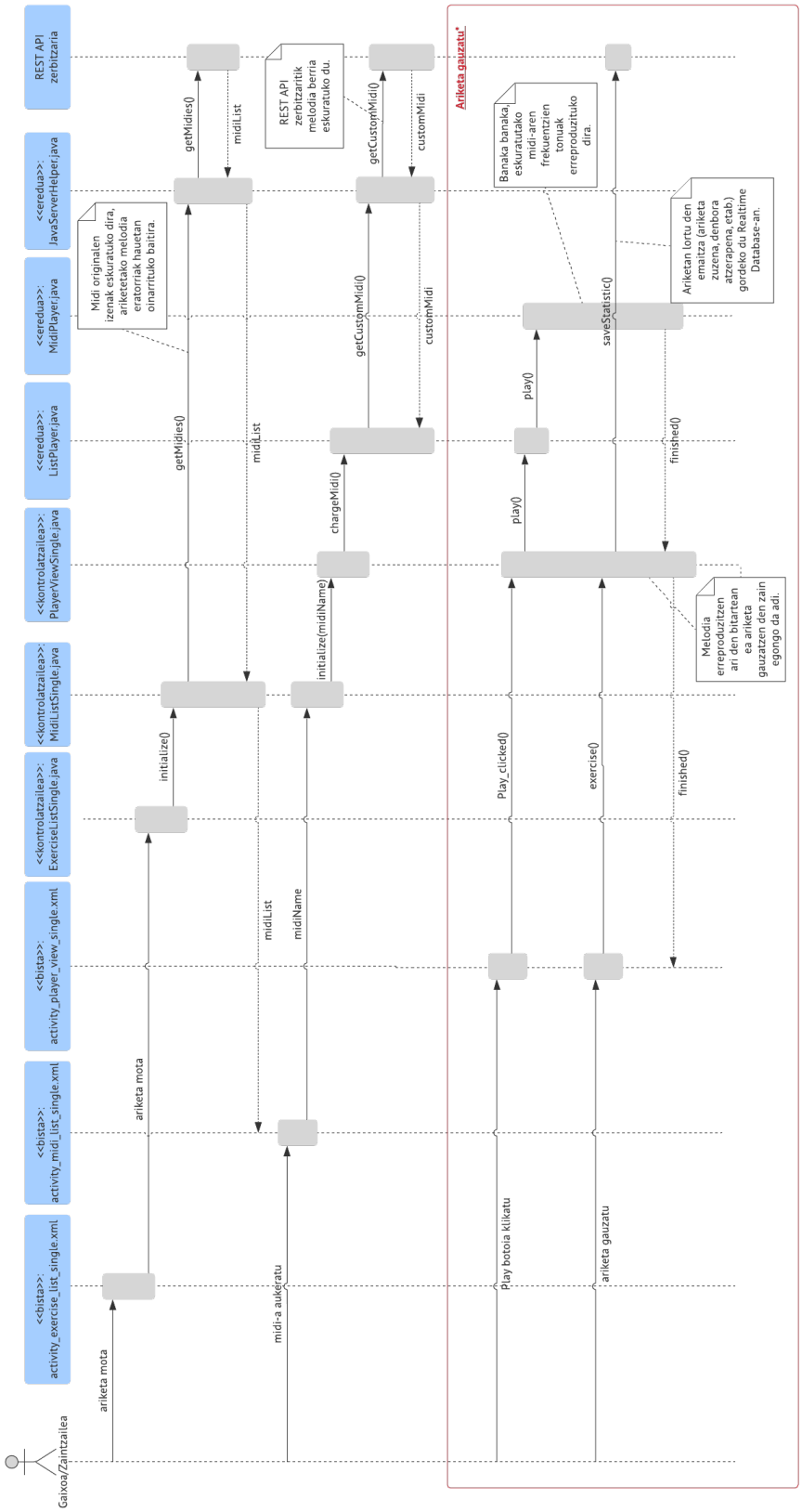
Bakarkako ariketak gertaera fluxuaren sekuentzia diagrama ikus daiteke 4.6 irudian. Bakarkako ariketen funtzionamendua deskribatzeko balioko du.

4.3.4 Taldeko ariketak

Taldeko ariketak gertaera fluxuaren sekuentzia diagrama ikus daiteke 4.7 irudian. Baina, diagrama horretan zaintzaileak taldea sortzeko prozesua bakarrik azaltzen da, izan ere, gaixoak



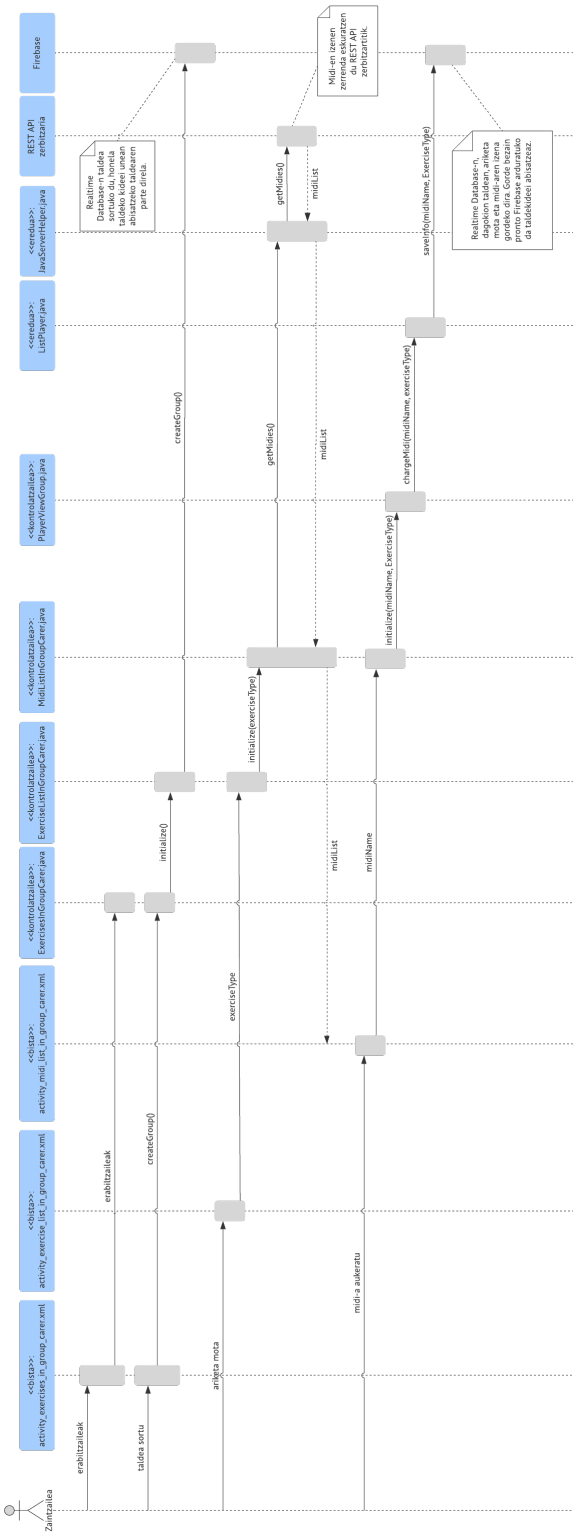
Irudia 4.5. Melodia berria sortu gertaera fluxuaren sekuentzia diagrama.



Irudia 4.6. Bakarkako ariketak gertaera fluxuaren sekuentzia diagrama.

ariketa gauzatzen duen zatia bakarkako ariketakoaren berdina baita. Hain zuzen, 4.6 irudian lauki gorri batean sartu da bakarkako ariketek taldeko ariketekin konpartitzen duten zatia.

Bestalde, gaixoak ariketa egiten hasi aurretik hauen sinkronizazioa gauzatuko da, eta horretarako NTP bezero bat erabiliko da, guztiak aldi berean erreproduzitzen hasteko.



Irudia 4.7. Taldeko ariketak gertaera fluxuaren sekuentzia diagrama.

5

Erabilitako tresnak

Proiektuaren zati honetan, proiektuan zehar erabili diren tresna eta teknologia nagusiak azalduko dira. Tresna hauek gaur egun oso erabiliak dira, eta zati honetan, hauek hobeto ulertzeko, besteak beste, beraien historia eta ezaugarriak azalduko dira.

5.1 Teknologiak

Lehendabizi hartu behar izan den erabakia garapen ingurune bat hautatzea izan da. Honetarako nahi beste aukera dago, bai garapen inguruneen artean, baita bertan erabili nahi diren baliabideen artean ere. Proiektua gaur egun oso erabiliak diren teknologiekin garatu da, eta jarraian emango dira hauen inguruko azalpenak.

5.1.1 Android Studio

Android Studio [6], Android aplikazioak sortzeko erabiltzen den garapen ingurune ofiziala da, eta jarraian bere historia eta ezaugarriak azalduko dira.

5.1.1.1 Historia

Android Studio, 2013ko maiatzaren 16an anuntziatu zen garapen ingurunea da. Android aplikazioak garatzeko orduan, garapen ingurune ofizial gisa Eclipse ordezkatzeko sortu zen eta lehen bertsio egonkorra 2014ko abenduan publikatu zuten.

JetBrains-en [17] IntelliJ IDEA [18] softwarean oinarritzen da eta Apache 2.0 [19] lizentziapean doan publikatuta dago. Azken bertsio egonkorra, 3.0 bertsioa, 2017an publikatu zuten.

5.1.1.2 Ezaugarriak

Android Studio, hiru sistema eragile nagusietan (Microsoft Windows, macOS eta GNU/Linux) erabilgarria da eta doan eskuragarri dago.

Bere izenak garbi adierazten duen bezala, Android aplikazioak sortzeko soilik balio duen IDEa da.

Aplikazioen interfazeei dagokienez, oso modu errazean lan egin daiteke, lehenetsitako txantiloiak daude eta objektuak (botoiak, testuak, etab.) layout-era arrastatu daitezke saguaren bidez, koderik idatzi gabe.

Bestalde, kodea idazterakoan laguntza ugari eskaintzen ditu eta hainbat kode modu automatikoan sor daiteke.

Bi programazio lengoaia nagusi erabil daitezke Android Studio-n, Java [20] eta Kotlin [21]. Garatutako aplikazioa Java-z idatzi da.

5.1.2 Eclipse

Eclipse, ordenagailu programazioan erabiltzen den garapen ingurune integratua (IDE) da. Oinarrizko "workspace"bat eta ingurunea pertsonalizatzeko plugin hedagarrien sistema bat dauka. Eclipse-n nagusiki Java-z idazten da eta bere erabilera nagusia Java aplikazioak garatzea da, baina beste lengoaia desberdinetako aplikazioak garatzeko ere erabil daiteke plugin desberdinak erabiliz. Esaterako, C++ [22] edo ADA [23] lengoaiak erabil daitezke.

5.1.2.1 Historia

Eclipse, Smalltalk-n oinarritutako VisualAge [24] familiako garapen ingurune integratuetan inspiratuta dago. VisualAge-k arrakasta izan zuen arren, bertan sortutako kodea ez zen osagaietan zimendutako software ingeniaritza ereduan oinarritzen. Horren ordez, kode guztia .dat izeneko fitxategi batean konprimitzen da eta gainera klase indibidualak ez dira errez atzigarriak.

Hori dela eta, talde batek, nagusiki IBM Cary NC laborategian, produktu berri hau garatu zuen Java-n oinarritutako ordezkapen gisa. 2001eko azaroan, partzuergo bat sortu zen eta honela garatu zuten Eclipse, kode irekiko software gisa.

2012ko apirilaren 26an Informatika makinariaren elkarteak (ACM) [25] Eclipse saritu zuen ACM Software Systems sariarekin.

5.1.2.2 Ezaugarriak

Eclipse-k, analizatzaile sintaktiko bat duen testu editore bat dauka eta denbora errealean konpilatzeko gai da.

JUnit [26] bidez proba unitarioak egiten ditu, CSV bidez bertsioen kontrola eramaten du eta Ant [27] integratuta dauka. Bestalde proiektuak, klaseak, testak, etab. sortzeko eta errefaktorizatzeko laguntzaileak dauzka.

Horrez gain, plugin bidez hainbat gauza desberdin gehi daitezke.

5.1.3 Firebase

Firebase, web aplikazioak eta mugikorreko aplikazioa garatzeko plataforma bat da. Plataforma hau, hodeian kokatzen da (Google Cloud Platform-ekin [28] integratuta) eta hainbat tresna eskaintzen ditu proiektu desberdinak sortu eta sinkronizatzeko.

5.1.3.1 Historia

James Tamplin eta Andew Lee-k, Envolve izeneko proiektu bat hasieratu zuten 2011n. Proiektu honek garatzaileei API bat eskaintzen zien beraien web orrian txat bat integra zezaten.

Txat zerbitzua eskaini ondoren, garatzaile askok zerbitzu hau aplikazioetan datu paketeak bidaltzeko erabiltzen zutela ohartu ziren Tamplin eta Lee. Orduan, 2012ko apirilean, bi zerbitzuak bereizi zituzten, hau da, denbora errealean informazioa bidaltzeko zerbitzu berria sortu zuten, Firebase hain zuzen.

2014an Google-k erosi zuen eta orduz geroztik Firebase asko hazi da, izan ere Google-k beste hainbat zerbitzu gehitu baitizkio.

5.1.3.2 Ezaugarriak

Esan bezala, Firebase-k denbora errealean datuak sinkronizatzeko balio du, eta ez dago konexioak kudeatu beharrik, ezta kode zati luzeak idatzi beharrik ere.

Hainbat plataformetarako balio duten tresna ugari eskaintzen ditu. Hain zuzen, honako plataforma hauetan erabil daiteke: Android [13], IOS [29], web aplikazioak, Unity [30] eta C++ [22].

Googleren azpiegitura erabiltzen du eta proiektuak sor daitezke inolako zerbitzariren beharrik izan gabe, tresnak SDK-ren barnean baitoaz.

Esan bezala hainbat tresna desberdin eskaintzen ditu, hona hemen bakoitzaren azalpen labur bat:

A. Firebase Analytics

Firebase-rako Google Analytics [31] aplikazioaren erabileraren inguruko estatistikak eta erabiltzaileen erabilerari buruz informazioa eskaintzen duen tresna bat da.

B. Firebase Cloud Messaging

Mezularitzarako erabiltzen den doako plataforma bat da eta gaur egun Android, IOS eta web aplikazioetan erabil daiteke.

C. Firebase Auth

App-an idatzitako kodearekin soilik erabiltzaileak autentifikatzen dituen zerbitzua da. Facebook, GitHub, Twitter, Google, Yahoo eta Microsoft-eko kontuen bidezko autentifikazioa eskaintzen du, baina nahi izanez gero e-posta eta pasahitz bidezko autentifikazio klasikoa ere erabil daiteke. Zerbitzu honekin autentifikazio sistemen sorkuntza erraztu nahi da.

Gainera, bestelako zerbitzu batzuk ere eskaintzen ditu, esaterako, kontuak berreskuratzea eta egiaztapena edota erabiltzaileen kuoten erregistroa.

D. Realtime Database

Firebase-k denbora errealeko "back-end"datu base bat eskaintzen du, eta hau JSON moduko zuhaitz gisa antolatzen da. Zerbitzuak informazio gorde edota lortzeko API bat eskaintzen du, baina REST API bidez ere irisgarria da.

Denbora errealeko sinkronizazioari esker, erabiltzaileek informazioa denbora errealean lor dezakete. Izan ere, erabiltzaile batek datu basean aldaketaren bat egiten duenean, aldaketa gorde eta erabiltzaile guztiei jakinarazten zaie aldi berean.

E. Firebase Storage

Sareko kalitatea kontuan hartu gabe, Firebase aplikazioentzat fitxategien igoera eta deskarga seguruak eskaintzen dituen zerbitzua da. Garatzaileak irudiak, audioa, bideoak edota erabiltzaileak sortutako bestelako edozein fitxategi gorde ditzake bertan.

F. Firebase Firestore

Realtime Database-k bezala, NoSQL motako datu base bat eskaintzen duen arren, hainbat desberdintasun ditu. Bildumetan gordetako dokumentuen gisa antolatzen da eta hauetan azpi-bildumak edo mota desberdinetako datuak sar daitezke.

6

Garapena

Proiektuaren garapenaren oinarria alzheimerraren hasierako fasean dauden pazienteei laguntzea da. Hori dela eta alzheimerra eta bestelako dementziak dituzten gaixoekin lan egiten duen AFAGI [32] elkartearekin harremana mantendu da.

Aplikazioak musikan oinarritutako terapiak erabiliko ditu pazienteei laguntzeko, hau da, musikan oinarritutako jolasak eskainiko dira.

Hortaz, behin eskakizunen bilketa eta diagramen diseinua gauzatuta, aplikazioaren inplementazioa izan da eginbehar nagusia.

Hasteko Android aplikazioaren garapena azalduko da, eta batez ere, bere hiru mailetan (Eredua-Bista-Kontrolatzailea) zentratuko da. Ondoren sortu behar izan den REST API zerbitzariaren garapena azalduko da.

6.1 Android aplikazioa

Esan bezala, aplikazioa hiru mailatan bereiziko da, eta jarraian maila bakoitzaren inplementazioa bere aldetik azalduko da.

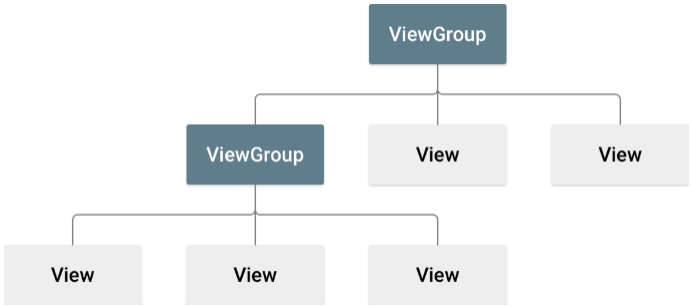
6.1.1 Bista

Android Studio-n garatutako aplikazioen bistak XML fitxategien bidez definituko dira. Fitxategi hauek sortzeko orduan, defektuzko txantiloiak edo txantiloi hutsak aukeratu ahalko ditugu.

Proiektu honetan txantiloi hutsekin egingo da lan, eta elementuak beharren arabera sartuko dira.

XML fitxategien elementu nagusiari "layout"deitzen zaio, eta hau izango da bistari egitura emango diona. "layout"-eko elementuek View eta ViewGroup objektuez osatutako hierar-

kia bat jarraituko dute 6.1 irudian ikus daitekeen moduan. ViewGroup-ak edukiontzi ikusezinak izango dira eta View-ak izan ahalko dituzte beren barnean. Aldiz, View-ei "widgets"ere deitu ohi zaie, eta hauek elementu interaktiboak izan ohi dira, esate baterako, botoiak edota testu sarrerak. Elementu guzti hauek hainbat atributu izan ahalko dituzte, eta horietako bat id-a izango da. Hain zuren, id-aren bitartez bereiziko dira elementuak elkarren artean.



Irudia 6.1. ViewGroup eta View-en arteko hierarkia.

Aplikazio gehienetan bezala, honetan ere elementu horiek erabiliko dira. Baina horiez gain, zerrendak dinamikoki kargatu ahal izateko RecyclerView motako elementuak ere erabiliko dira. Esate baterako, 6.2 irudian ikusi ahalko da nola gehituko den bistetan RecyclerView motako elementua.

```
<android.support.v7.widget.RecyclerView
    android:id="@+id/rvPatients"
    android:layout_width="match_parent"
    android:layout_height="240dp"
    android:layout_alignParentStart="true"
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:scrollbars="vertical" />
```

Irudia 6.2. RecyclerView motako elementua.

RecyclerView-ek askatasun handia ematen dute hainbat objekturekin denbora errealean lan egiteko orduan. Baina mota honetako elementuak erabili ahal izateko, lehendabizi v7 bateragarritasun liburutegiak gehitu beharko ditugu build.gradle moduluko dependecies atalean. 6.3 irudian ikus daiteke gehitu beharreko dependentzia.

```
dependencies {
    implementation 'com.android.support:recyclerview-v7:28.0.0'
}
```

Irudia 6.3. v7 bateragarritasun liburutegiak gehitu.

6.4 irudian ikus daitekeen bezala, bista nahiko sinpleak izan ohi dira, eta gainera, 6.5 ikus daitekeen bezala, Android Studio tresnak testu bidez edo interfaze bidez uzten digu diseinua egokitzen.

6.1.2 Kontrolatzailea

Kontrolatzailearen zeregin nagusia bista eta ereduaren arteko bitartekari lanak egitea izango da. Erabiltzailearen sarrera datuak jaso, eta hauekin ereduei edo bistei eskaerak egingo dizkie. Baina proiektu honetan, horrez gain Firebase-rekin konexioa gauzatu beharko du, aurretik esan den bezala, Android aplikazioak kontrolatzaileetatik konektatu ohi baitira Firebase-ra.

Hori dela eta, kontrolatzaileen zatia bitan banatuko da, lehendabizi kontrolatzaileen ohiko garapena azalduko da, eta ondoren Firebase-rekin egin behar izan diren konexioak.

6.1.2.1 Kontrolatzaileen ohiko garapena

Android Studio-n, kontrolatzaile guztiak Activity motako klaseak izango dira, eta klase hauek beti bista bati lotuta joango dira.

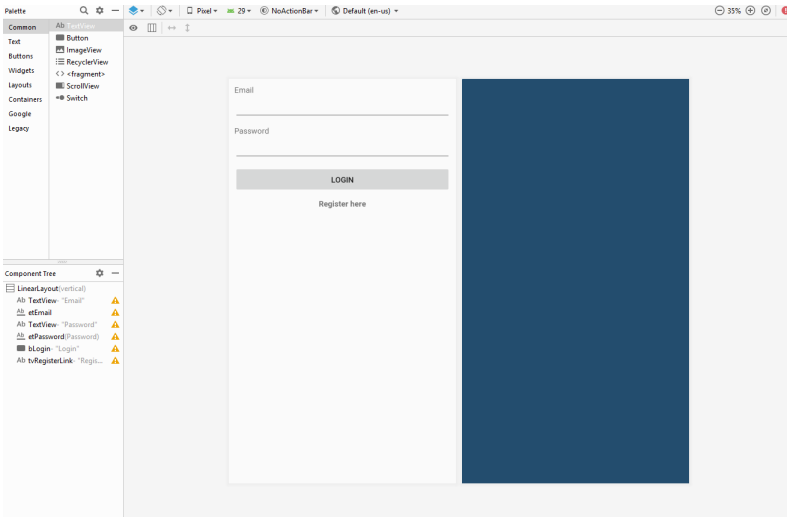
Bistako elementuak instantziatu ahal izateko, kontrolatzaileko klaseek AppCompatActivity klasea heredatu beharko dute. Klase hau heredatzean, hainbat metodo sortu beharko dira, eta hauetako batzuk derrigorrezkoak izango dira, esate baterako, onCreate() metodoa. Metodo hau, Activity bakoitza hasieratzean automatikoki exekutatuko da. Horrez gain, onCreate() metodoan adierazten da zein den Activity horri dagokion bista, eta bistako elementuen instantziak eta gertaerak ere hemen esleituko dira. Adibide bezala, ikusi 6.6 irudia.

Horrez gain, proiektu honetan hainbat botoi egongo dira bista askotan, beraz, kontrolatzaileetan hauen gertaerak hobeto kudeatzeko View.onClickListener klase inplementatuko da. Honela, 6.7 irudian ikus daitekeen moduan, onClick() metodo orokor bat sortuko da eta bertan switch bitartez kudeatuko dira gertaerak.

Android aplikazio honek hainbat bista desberdin izango ditu, eta noski bista bakoitzeko kontrolatzaile bat, beraz, batetik bestera joaten jakitea oso garrantzitsua izango da. Android Studio-n nahiko modu sinplean egin daiteke hau, izan ere, nahikoa izango baita Intent motako elementu bat sortzea uneko eta hurrengo klasearekin, eta ondoren, Intent motako elementu

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".Login"
    tools:showIn="@layout/activity_login">

    <TextView
        android:layout_width="wrap_content"
        android:text="Email"
        android:layout_height="wrap_content" />

    <EditText
        android:id="@+id/etEmail"
        android:layout_width="match_parent"
        android:layout_marginBottom="10dp"
        android:layout_height="wrap_content" />


    <TextView
        android:layout_width="wrap_content"
        android:text="Password"
        android:layout_height="wrap_content" />

    <EditText
        android:id="@+id/etPassword"
        android:layout_width="match_parent"
        android:layout_marginBottom="10dp"
        android:layout_height="wrap_content"
        android:inputType="textPassword" />
    <Button
        android:id="@+id/bLogin"
        android:text="Login"
        android:layout_marginBottom="10dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

    <TextView
        android:id="@+id/tvRegisterLink"
        android:layout_width="wrap_content"
        android:text="Register here"
        android:textStyle="bold"
        android:layout_gravity="center_horizontal"
        android:layout_height="wrap_content" />

</LinearLayout>
```

Irudia 6.4. Adibidea: Login bistaren XML-a.



Irudia 6.5. Adibidea: Login bistaren diseinua.

```
public class Login extends AppCompatActivity implements View.OnClickListener {

    private Button bLogin;
    private EditText etEmail, etPassword;
    private TextView tvRegisterLink;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        etEmail = (EditText) findViewById(R.id.etEmail);
        etPassword = (EditText) findViewById(R.id.etPassword);
        bLogin = (Button) findViewById(R.id.bLogin);
        tvRegisterLink = (TextView) findViewById(R.id.tvRegisterLink);

        bLogin.setOnClickListener(this);
        tvRegisterLink.setOnClickListener(this);
```

Irudia 6.6. Login Activity-aren onCreate funtzioa.

```
@Override
public void onClick(View v){
    switch(v.getId()){
        case R.id.bLogin:
            if(!(etEmail.getText().equals("")) && !(etPassword.getText().equals(""))){
                Login();
            }else{
                Toast alert = Toast.makeText( context: Login.this,  text: "Email or password is empty!", Toast.LENGTH_SHORT);
                alert.show();
            }

            break;

        case R.id.tvRegisterLink:
            startActivity(new Intent( packageContext: this, Register.class));
            break;
    }
}
```

Irudia 6.7. Login Activity-ko onClick() metodo orokorra.

hori parametro gisa hartuta, startActivity() metodoari deitzea. Ikusi 6.7 irudiko switch-eko bigarren kasuan egiten dena.

6.1.2.2 Kontrolatzaileak eta Firebase

Firebase-k eskaintzen dituen zerbitzuak Android Studio-n garatutako aplikazioetan erabiltzeko, lehendabizi hainbat pausu jarraitu beharko dira.

Hasteko, Firebase-n proiektu berri bat hasi beharko da, eta ondoren app-a Firebase-ko proiektuan erregistratu beharko da. Horretarako, Firebase-ko kontsolan "Agregar app" botoia klikatu eta eskatzen dituen datuak bete beharko dira (kontsolak oso garbi adierazten du zein diren sartu beharreko datuak). Behin datuak sartuta google-service.json izeneko fitxategi bat deskargatu ahalko da, eta hau aplikazioaren karpetan kopiatuko da. Azkenik, 6.8 irudian ikusten diren dependentziak gehitu beharko dira aplikazioaren Gradle fitxategian.

```
implementation 'com.google.firebase:firebase-core:16.0.8'
implementation 'com.google.firebase:firebase-database:16.1.0'
implementation 'com.google.firebase:firebase-auth:16.1.0'
```

Irudia 6.8. Firebase-ren dependentziak.

Behin Firebase prest dagoela, eskaintzen dituen zerbitzuak kontsumitu ahalko ditugu. Proiektu honetan Firebase-ko Authentication eta Realtime Database erabiliko dira.

A. Firebase Authentication

Authentication-ek eskaintzen dituen zerbitzuak erabiltzaileak gordetzeko eta hauen autentifikaziorako erabiliko dira. Oso metodo sinpleak eskaintzen ditu, eta gai da guk e-posta eta pasahitza emanda saioen hasieraketa edota erabiltzaile berrien erregistroa gauzatzeko. Horrez gain, uneoro konektatuta dagoen erabiltzailea identifikatu ahalko du, eta horrek lan handia kentzen du gainetik.

Horretarako, lehendabizi Firebase-ko erabiltzailearen instantzia bat sortu beharko da, 6.9 irudian ikusten den bezala.

```
private FirebaseAuth mAuth = FirebaseAuth.getInstance();
```

Irudia 6.9. Firebase Authentication-eko erabiltzailearen instantzia.

Ondoren instantzia horri erabiltzaile bat esleitzeko saioa hasteko metodoa erabiliko da. Ikusi 6.10 irudia.

```
mAuth.signInWithEmailAndPassword(email, pass)
```

Irudia 6.10. Firebase Authentication-eko erabiltzaileak saioa hasteko metodoa.

Bestalde, beste hainbat metodo ere eskaintzen ditu, saioa ixteko, erregistratzeko, etab. Baina aurrekoak bezala oso metodo sinpleak dira. Proiektu honetan metodo hauek erabiliko dira erabiltzaileen kautoketarako.

B. Firebase Realtime Database

Datuak gordetzeko Firebase erabiltzeko arrazoi nagusia, taldeko ariketetan denbora errealeko komunikazioaren beharra izango da. Izan ere, hainbat mugikor sinkronizatu beharko dira eta Firebase-k eskaintzen duen Realtime Database-k asko laguntzen du horrekin.

Datu base honetan bi motatako datuak gordeko dira: erabiltzaileak eta taldeak. Hori dela eta, 6.11 irudian ikus daitekeen moduan, datu horietara iristeko bi instantzia beharko ditugu.

```
mAuth.signInWithEmailAndPassword(email, pass)
```

Irudia 6.11. Firebase Realtime Database-ren instantziak.

Firebase-rekin oso erraza izango da datu base honetan datuak sartzea (setValue()), eguneratzea (updateChildren()) eta ezabatzea (removeValue()).

Bestalde, datuak eskuratzeko orduan hainbat berezitasun ditu, datuak eskuratzeko "Listener"batez baliatuko baita. Hain zuzen, datu baseko nahi den elementuaren instantzia bat lortuko da eta honi "Listener"bat gehituko zaio, 6.12 irudian ikus daitekeen bezala. Honela, hasieran datu baseko balioak eskuratzeaz gain, datu basean aldaketarik badago erabiltzaileari jakinaraziko zaio unean bertan. Hori dela eta, esan daiteke denbora errealean lan egiten duela datu base honek.

```
groupsRef.child(group.getName()).child("patients").addListenerForSingleValueEvent(
        new ValueEventListener() {
    @Override
    public void onDataChange(@NonNull DataSnapshot dataSnapshot) {...}
    @Override
    public void onCancelled(@NonNull DatabaseError databaseError) {}
});
```

Irudia 6.12. Firebase Realtime Database-ko "Listener-ak.

6.1.3 Eredua

Android aplikazioaren azken zatia eredua izango da, eta hau datuen arduraduna izango da. Normalean datu baseekin konexio egin eta datuak eskuratzeaz arduratzen den arren, lehen esan bezala proiektu honetan Firebase-ko Realtime Database erabiliko da datu base bezala, eta honekin, kontrolatzaileetatik egingo da konexioa.

Proiektu honetan Firebase-tik lortutako datuen edukiontzi lana egingo dute ereduko elementuek. Izan ere, Firebase-k denbora errealean abisatzen baitu datuak noiz aldatu diren, eta orduan, abisatzen duen arte, ereduko elementuetan gorde diren datuak zuzenak izango baitira.

Android Studio-n, ereduko elementu hauek Java klase arruntak izango dira, beraiek behar dituzten metodo eta guzti. Ikusi 6.13 irudia adibide gisa.

```
public class User {

    private String id;
    private String email;
    private String status = "offline";
    private boolean carer;
    private String group = "";
    private HashMap<String, Object> statistics;

    public User(){}

    public User(String id, String email, boolean carer){
        this.id = id;
        this.email=email;
        this.carer=carer;
        this.statistics=new HashMap<String, Object>();
    }

    public String getId() { return id; }
    public void setId(String id) { this.id = id; }
    public String getStatus() { return this.status; }
    public String getGroup() { return group; }
    public void setGroup(String group) { this.group = group; }
    public boolean isCarer() { return carer; }
}
```

Irudia 6.13. Erabiltzaileen Java klasea.

6.2 REST API zerbitzaria

Zerbitzari hau arduratuko da fitxategiak gordetzeaz, ariketetarako midi berriak sortzeaz, midi originalak eskuratzeaz, etab. Hau da, midi eta melodiekin zerikusia duen guztia zerbitzari honetan gauzatuko da.

Bere garapena Eclipse ingurunean gauzatuko da, eta Dynamic Web Project motako proiektu bat izango da. Beraz, lehendabiziko pausua mota honetako proiektu bat sortzea izango da.

Ondoren, proiektu hau Maven proiektu bilakatuko da, eta hau egitean pom.xml fitxategia sortuko da. Hemen proiektuak izango dituen dependentziak gehitu beharko dira, baina Maven plugina instalatuta izanez gero, hauek automatikoki gehituko dira.

Gero, web.xml fitxategia konfiguratu beharko da 6.14 irudian bezala. Izan ere, REST API zerbitzari gisa soilik erabiliko den arren, azken finean, web aplikazio bat izango da.

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
id="WebApp_ID" version="2.5">
  <display-name>RESTAPI</display-name>
    <servlet>
        <servlet-name>jersey-serlvet</servlet-name>
        <servlet-class>com.sun.jersey.spi.container.servlet.ServletContainer</servlet-class>
            <init-param>
                <param-name>com.sun.jersey.config.property.packages</param-name>
                <param-value>com.java4s</param-value>
            </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>

    <servlet-mapping>
        <servlet-name>jersey-serlvet</servlet-name>
        <url-pattern>/rest/*</url-pattern>
    </servlet-mapping>

</web-app>
```

Irudia 6.14. web.xml fitxategia.

Azkenik, 6.15 irudian ikusten den moduan, Java klase bat sortuko beharko da zerbitzuak definitzeko. Hemen eskaera mota desberdinetako metodoak sartu ahalko dira, POST, PUT, GET, etab. 6.15 irudian ikusten den moduan metodoaren gainean definitzen dira eskaera mota, bidea, itzuliko duen testu formatua eta eskaera motaren arabera, jasoko duen testu formatua ere.

Maven dependentziak ondo gehitu badira jada prest egon beharko luke REST API zerbitzariak.

```java
@Path("/MidiService")
public class MidiService {

    @GET
    @Path("/midi_files")
    @Produces({MediaType.APPLICATION_JSON})
    public String getMidies() {

        System.out.println("MIDI LIST PREPARED!");

        Gson gson = new Gson();

        File folder = new File("./resource/doinuak/originalak");
        File[] listOfFiles = folder.listFiles();
        String[] midies = new String[listOfFiles.length];

        int k = 0;
        for (int i = 0; i < listOfFiles.length; i++) {
          if (listOfFiles[i].isFile() && listOfFiles[i].getName().endsWith(".mid")) {
            midies[k] = listOfFiles[i].getName();
            k++;
          }
        }

        System.out.println("MIDI LIST SENT!");

        return gson.toJson(midies);

    }
```

Irudia 6.15. REST API zerbitzuak definitzen diren klasea.

6.3 Garapenean izan diren arazoak

Proiektuaren garapenean hainbat zailtasun gainditu behar izan dira, eta jarraian arazo horien zergatiak eta irtenbideak azalduko dira.

6.3.1 Musika erreproduzitzailea

Android aplikazioko musika erreproduzitzailea inplementatzeko orduan hainbat arazo izan dira.

Hasteko, melodia erreproduzitzeko AudioTrack motako elementu bat erabiliko da eta honi balio desberdinak esleituko zaizkio nota desberdin bakoitza erreproduzitu ahal izateko. Baina AudioTrack motako elementu hori hasieratzeko orduan, oso garrantzitsua izango da 6.16 irudian ikusten den bezala egitea. Bestela melodia moztu egiten da eta ez da ondo entzuten.

Bestalde, AudioTrack motako elementua erreproduzitzeko orduan oso garrantzitsua izango da 6.17 irudiko ordenak jarraitzea, bestela aplikazioa erroreak ematen eta bertan behera geratzen hasten baita.

```
private AudioTrack track = new AudioTrack(
        (new AudioAttributes.Builder()).setLegacyStreamType(AudioManager.STREAM_MUSIC).build(),
        (new AudioFormat.Builder()).setChannelMask(AudioFormat.CHANNEL_OUT_STEREO).setEncoding(
                AudioFormat.ENCODING_PCM_16BIT).setSampleRate(44100).build(),
         bufferSizeInBytes: (int)(44100.0 * 2.0 * (5000 / 1000.0)) & ~1 * (Short.SIZE / 8),
        AudioTrack.MODE_STREAM,  sessionId: 1);
```

Irudia 6.16. AudioTrack motako elementuaren hasieratzea.

```
public void generateTone(double freqHz, int durationMs){
    int count = (int)(44100.0 * 2.0 * (durationMs / 1000.0)) & ~1;
    short[] samples = new short[count];
    for(int i = 0; i < count; i += 2){
        short sample = (short)(Math.sin(2 * Math.PI * i / (44100.0 / freqHz)) * 0x7FFF);
        samples[i + 0] = sample;
        samples[i + 1] = sample;
    }
    track.write(samples,  offsetInShorts: 0, count);
}
```

Irudia 6.17. Tonuak sortzeko erabiliko den funtzioa.

Horrez gain, aplikazio honetan, erreproduzitzaileak notak erakusteko orduan, dinamikoki aldatu beharko da pantaila, eta bistaren kontrolatzailea ez den beste klase batzuetatik aldatu beharko da gainera. Hori egiteko hainbat saiakera eta hutsegite egin ondoren, azkenik Callback motako elementuak erabiltzea erabaki da, eta hauekin lortu da nahi zena.

Hain zuzen, Callback motako elementuak klaseen arteko komunikaziorako erabiltzen dira. Honela erreprodukziorako sortu den klaseak Activity-aren bista aldatzeko gaitasuna izango dute.

7

Probak

Zati honetan, aplikazioaren funtzionamendu egokia egiaztatzeko egin diren probak azalduko dira. Erabilpen kasu bakoitzari proba desberdinak egin zaizkio eta lortu diren emaitzak azalduko dira. Ulergarriagoa izateko, pausu desberdinetan banatuko dira proba hauek, hau da, hasteko probaren deskribapen bat emango da, ondoren esperotako emaitza azalduko da, gero lortu den emaitza, eta azkenik ondorio bat aterako da.

7.1 Erregistratu

1. – **Deskribapena:** eremuak zuzen betetzea.
   – **Esperotako emaitza:** zuzen erregistratu eta orrialde nagusira joatea.
   – **Lortutako emaitza:** zuzen erregistratu eta orrialde nagusira joan da.
   – **Ondorioa:** zuzena.
2. – **Deskribapena:** eremuren bat hutsa uztea.
   – **Esperotako emaitza:** eremuak bete behar direla adieraztea.
   – **Lortutako emaitza:** eremuak bete behar direla adierazi du.
   – **Ondorioa:** zuzena.
3. – **Deskribapena:** jada badagoen e-posta bat erabiltzen saiatzea.
   – **Esperotako emaitza:** erregistroa gauzatzen ez uztea.
   – **Lortutako emaitza:** ez du erregistroa gauzatzen utzi.
   – **Ondorioa:** zuzena.
4. – **Deskribapena:** pasahitz desberdinak sartzea.
   – **Esperotako emaitza:** pasahitzak desberdinak direla adieraztea.
   – **Lortutako emaitza:** pasahitzak desberdinak direla adierazi du.
   – **Ondorioa:** zuzena.

7.2 Saioa hasi

1. – **Deskribapena:** e-posta eta pasahitz zuzenak sartzea.
   – **Esperotako emaitza:** saioa zuzen hastea eta orrialde nagusira joatea.
   – **Lortutako emaitza:** saioa zuzen hasi da eta orrialde nagusira joan da.
   – **Ondorioa:** zuzena.
2. – **Deskribapena:** e-posta edo pasahitz okerra sartzea.
   – **Esperotako emaitza:** saioa ez hastea eta datuak okerrak direla adieraztea.
   – **Lortutako emaitza:** saioa ez da hasi eta datuak okerrak direla adierazi du.
   – **Ondorioa:** zuzena.

7.3 Saioa itxi

1. – **Deskribapena:** saioa normal ixtea.
   – **Esperotako emaitza:** saioa itxi eta saioa hasteko orrialdera joatea.
   – **Lortutako emaitza:** saioa zuzen itxi du eta saioa hasteko orrialdera joan da.
   – **Ondorioa:** zuzena.

7.4 Bakarkako ariketak

1. – **Deskribapena:** ariketa zuzen gauzatzea.
   – **Esperotako emaitza:** ariketa lortutako emaitza datu basean gordetzea.
   – **Lortutako emaitza:** ariketan lortutako emaitza datu basean gorde du.
   – **Ondorioa:** zuzena.
2. – **Deskribapena:** ariketa oker gauzatzea.
   – **Esperotako emaitza:** ariketan lortutako emaitza datu basean gordetzea.
   – **Lortutako emaitza:** ariketan lortutako emaitza datu basean gorde du.
   – **Ondorioa:** zuzena.
3. – **Deskribapena:** ariketa ez gauzatzea.
   – **Esperotako emaitza:** ariketa gauzatu ez duen arren ariketa oker egin balu bezala gordetzea.
   – **Lortutako emaitza:** ariketa oker egin izan balu bezala gorde da datu basean.
   – **Ondorioa:** zuzena.
4. – **Deskribapena:** ariketa hasi aurretik atzera egitea.
   – **Esperotako emaitza:** ariketak aukeratzeko orrialdera itzultzea.
   – **Lortutako emaitza:** ariketak aukeratzeko orrialdera itzuli da.

- **Ondorioa:** zuzena.

7.5 Taldeko ariketak

1. - **Deskribapena:** taldeko ariketa gauzatu talde txiki batekin.
   - **Esperotako emaitza:** zaintzaileak ariketa prestatu eta gaixoek taldeko ariketa gauzatzea.
   - **Lortutako emaitza:** zaintzaileak ariketa prestatu du eta gaixoek taldeko ariketa gauzatu dute.
   - **Ondorioa:** zuzena.
2. - **Deskribapena:** taldeko ariketa gaixorik gabe gauzatuko da.
   - **Esperotako emaitza:** zaintzaileak ariketa prestatzea eta ariketa martxan jartzea bere mugikorrean bakarrik.
   - **Lortutako emaitza:** zaintzaileak ariketa prestatu du eta ariketa martxan jarri da bere mugikorrean bakarrik.
   - **Ondorioa:** zuzena.
3. - **Deskribapena:** zaintzaileak taldera gaixo bat gehitzea.
   - **Esperotako emaitza:** zaintzaileak gaixo bat gehitzea taldera eta gaixoaren egoera "okupatuta"izatera pasatzea.
   - **Lortutako emaitza:** zaintzaileak gaixo bat gehitu du taldera eta gaixoaren egoera "okupatuta"izatera pasa da.
   - **Ondorioa:** zuzena.
4. - **Deskribapena:** zaintzaileak gaixo bat taldetik botatzea.
   - **Esperotako emaitza:** zaintzaileak gaixo bat taldetik botatzea eta gaixoaren egoera "online"izatera pasatzea.
   - **Lortutako emaitza:** zaintzaileak gaixo bat taldetik bota du eta gaixoaren egoera "online"izatera pasa da.
   - **Ondorioa:** zuzena.

7.6 Melodia originalak entzun

1. - **Deskribapena:** melodia originalak entzutea.
   - **Esperotako emaitza:** aukeratutako melodia originala zuzen entzutea.
   - **Lortutako emaitza:** aukeratutako melodia originala zuzen entzun da.
   - **Ondorioa:** zuzena.

2. – **Deskribapena:** melodia entzun aurretik atzera egitea.
   – **Esperotako emaitza:** melodiak aukeratzeko orrialdea erakustea.
   – **Lortutako emaitza:** melodiak aukeratzeko orrialdea erakutsi da.
   – **Ondorioa:** zuzena.

7.7 Melodia berriak sortu

1. – **Deskribapena:** melodia bat zuzen sortzea.
   – **Esperotako emaitza:** melodia zuzen sortzea eta REST API zerbitzarian gordetzea.
   – **Lortutako emaitza:** melodia zuzen sortu da eta REST API zerbitzarian gorde da.
   – **Ondorioa:** zuzena.
2. – **Deskribapena:** melodia entzun eta gero ez gordetzea.
   – **Esperotako emaitza:** melodiak aukeratzeko orrialdera joatea.
   – **Lortutako emaitza:** melodiak aukeratzeko orrialdera joan da.
   – **Ondorioa:** zuzena.

7.8 Sortutako melodiak entzun

1. – **Deskribapena:** erabiltzaileek sortu dituzten melodiak entzutea.
   – **Esperotako emaitza:** melodiak ondo entzutea.
   – **Lortutako emaitza:** melodia ondo entzun dira.
   – **Ondorioa:** zuzena.
2. – **Deskribapena:** melodiarik sortu gabe dagoenean erabilpen kasu hau gauzatzen saiatzea.
   – **Esperotako emaitza:** zerrenda hutsa erakustea.
   – **Lortutako emaitza:** errorea eman du eta aplikazioa itxi da.
   – **Ondorioa:** okerra.
   – **Zuzenketa:** sistema ez zegoen zerbitzaritik erantzun huts bat jasotzeko preparatuta, beraz baldintza bat sartuta, erantzun hutsen tratamendua egin da.
     * **Zuzenketaren ondoren lortutako emaitza:** zerrenda hutsa erakusten du.
     * **Zuzenketaren ondoren lortutako ondorioa:** zuzena.
3. – **Deskribapena:** melodia kargatu eta gero entzun aurretik atzera egitea.
   – **Esperotako emaitza:** melodiak aukeratzeko orrialdera joatea.
   – **Lortutako emaitza:** melodiak aukeratzeko orrialdera joan da.
   – **Ondorioa:** zuzena.

7.9 Historiala ikusi

1. – **Deskribapena:** ariketetan lortu dituen emaitzen batezbestekoak ikustea.
   – **Esperotako emaitza:** ariketetan lortutako emaitzen batezbestekoak erakustea.
   – **Lortutako emaitza:** ariketetan lortutako emaitzen batezbestekoak erakutsi ditu.
   – **Ondorioa:** zuzena.
2. – **Deskribapena:** ariketarik gauzatu ez duen erabiltzaileak historiala ikustea.
   – **Esperotako emaitza:** leku guztietan %0ko puntuazioa azaltzea.
   – **Lortutako emaitza:** leku guztietan %0ko puntuazioa agertzen da.
   – **Ondorioa:** zuzena.

7.10 Gaixoen historiala ikusi

1. – **Deskribapena:** gaixoek ariketetan lortu dituzten emaitzen batezbestekoak ikustea.
   – **Esperotako emaitza:** gaixoek ariketetan lortutako emaitzen batezbestekoak erakustea.
   – **Lortutako emaitza:** gaixoek ariketetan lortutako emaitzen batezbestekoak erakutsi ditu.
   – **Ondorioa:** zuzena.
2. – **Deskribapena:** ariketarik gauzatu ez duen gaixoaren historiala ikustea.
   – **Esperotako emaitza:** leku guztietan %0ko puntuazioa azaltzea.
   – **Lortutako emaitza:** leku guztietan %0ko puntuazioa agertzen da.
   – **Ondorioa:** zuzena.

8

Ondorioak

Atal honetan, proiektuari buruzko hausnarketa bat egin eta ondorioak aterako dira. Hiru zatitan banatuko da, alde batetik, proiektuan egindako lanari buruzko hausnarketa bat egingo da eta ondorio batzuk aterako dira. Bestalde, proiektuan zehar ikasitakoaren balorazio bat egingo da, eta azkenik, etorkizunera begira proiektuak izan dezakeen eraldaketari buruz hausnartuko da.

8.1 Proiektuari buruzko hausnarketa eta ondorioak

Proiektuaren hasierako helburuak bete direla esan daiteke, izan ere, dementzia duten gaixoei lagundu ahalko dien tresna erabilgarri bat sortu baita. Erabilpen kasu sinpleez gain, proposatutako beste erabilpen kasu guztiak inplementatzea lortu da, eta aplikazioa gai da, besteak beste, denbora errealean hainbat gaixok taldean ariketak gauzatzeko, gaixoen estatistikak ikusteko, gaixoei jarraipen bat emateko eta melodia berriak sortzeko.

Bestalde, bi ariketa mota desberdin inplementatu dira, melodiak asmatzearena eta melodia noiz aldatzen hasten den asmatzearena. Ariketa mota desberdinak gauza ahal izatean, neurri batean, aplikazioa entretenigarria eta dinamikoa izatea lortu da, honela aplikazioa ez baita hain monotonoa izango.

Horrez gain, aplikazioaren erabiltzaileen zati handi batek alzheimerra izango duenez, aplikazio oso sinplea eta erabiltzeko erraza egin nahi zen, eta lortu da. Izan ere, ez dago despistatu dezakeen botoi arrarorik, une bakoitzean garbi adierazten da egin beharrekoa, eta gainera ariketak taldean egiterakoan guztia zaintzaileen esku uzten da, gaixoek soilik ariketa gauzatu beharko dute.

Esan bezala, melodia berriak sortzen dira REST API zerbitzari batean, eta honek lan zama handia izan dezakeen arren, aplikazioak segundu gutxi batzuk beharko ditu melodia berriak eskuratu eta erreproduzitzen hasi ahal izateko.

Hala ere, sortu den proiektua demo bat besterik ez da oraindik, aurrerago aipatuko dugun bezala aplikazioa borobiltzea falta baita, eta hobekuntza posible asko baititu oraindik. Esate baterako, aplikazioaren garapenean ideia berriak ere izan dira, adibidez, taldeko ariketetan, ariketak amaitzean zaintzaileak parte hartu duten gaixoen emaitzak ikustea.

Bestalde, probak nahiko mugatuak izan dira eta ezin izan da gaixo zehatz batzuen jarraipena eman. Hain zuzen, egin diren probak solteak izan dira, eta ezin izan da frogatu ea aplikazioa benetan lagungarria izan zaien.

8.2 Proiektuan zehar ikasitakoa

Honelako proiektu bat kudeatzea eta garatzea beti da aberasgarria, eta nire kasuan, esperientzia gutxi izanda, oraindik gehiago. Gauza berri ugari ikasteko aukera izan dut, izan ere, Android-erako egiten dudan lehen aplikazioa baita, eta Android Studio-n garatutako proiektuek hartzen duten arkitektura orain arte ikusitakoekin konparatuta oso desberdina baita. Horrez gain, bi zerbitzari sortu dira, eta zerbitzari mota hauek ere berriak dira niretzako.

Alde batetik Firebase zerbitzaria edo tresna erabiltzen ikasi dut, eta oso interesgarria iruditzen zait. Izan ere, gauza asko probatzera iritsi ez naizen arren, aukera ugari eskaintzen du eta erabilera oso erraza da.

8.3 Etorkizunera begira egindako hausnarketa

Aplikazioa oraindik demo bat da, izan ere, oraindik hainbat aspektu baititu hobetzeko.

Alde batetik, lehen esan bezala, proba gehiago eta sakonagoak egitea falta da, aplikazioan errorerik hauteman ez den arren, oso talde txikitan probatu baita. Horrez gain, alzheimerra duten pertsona talde bat bildu eta hainbat astetan zehar hauen datuak jasotzea komeniko litzateke ea aplikazioa eraginkorra den jakiteko.

Bestalde, aplikazioaren funtzionamenduari eman zaio garrantzi gehien, diseinu grafikoaren zatia ez da ia ukitu ere egin, eta diseinu grafikoa ere oso garrantzitsua da. Aplikazioa jendearentzat erakargarria izan behar da, hau da, dinamikoa eta ikusgarria izan behar da. Bestela jendeari aspergarria gerta dakioke eta hauek erabiltzeari utzi diezaiokete.

Azkenik, aplikazioari aipatutako hobekuntza eta probak egindakoan, aplikazioa AFAGI-ri eskaini nahi zaio. Bertako gaixoentzat erabilgarria izan baitaiteke, eta gainera, horrela ikusi ahal izango da ea benetan tresna baliagarria den.

Erreferentziak

1. R. M. Rodríguez Fernández, "Musicoterapia y enfermedad de Alzheimer. kNOWAlzheimer: Respuestas concretas a dudas reales." kNOW Alzheimer `https://knowalzheimer.com/musicoterapia-y-enfermedad-de-alzheimer/`. 2019ko apirila.
2. I. Goienetxea, I. Mendialdua, I. Rodriguez eta B. Sierra, "Statistics-based music generation approach considering both rhythm and melody coherence," *IEEE Access*, vol. 7, pp. 183365–183382, 2019. ISBN: 978-84-8438-686-5.
3. MoodAgent, "Sentimendu eta umorearen arabera musika erreproduzitzen duen aplikazioa." `https://moodagent.com/`.
4. A. Molinero Crespo, "¿Una App para el deterioro cognitivo y la enfermedad de Alzheimer leve? kNOWAlzheimer: Respuestas concretas a dudas reales." kNOW Alzheimer `https://knowalzheimer.com/una-app-para-el-deterioro-cognitivo/`. 2019ko apirila.
5. GitHub, "Taldean lan egitea ahalbidetzen duen plataforma bat da, eta horrez gain bertisoen kontrola ere eramaten du." `https://github.com/`.
6. Android Studio, "Android plataformaren garapen ingurune ofiziala." `https://developer.android.com/studio`.
7. Eclipse, "Aplikazioak garatzeko, kode irekiko tresna multzo batez osatutako softwarea." `https://www.eclipse.org/`.
8. Dynamic Web Project, "Web-eko proiektu mota hauetan PHP, ASP, JSP, Servlet, etab. bezalako kodea konplexuagoa erabiltzen da." `https://help.eclipse.org/2019-12/index.jsp?topic=%2Forg.eclipse.wst.webtools.doc.user%2Ftopics%2Fccwebprj.html`.
9. REST API, "Web zerbitzuk sortzen uzten duen estandarra logikoa eta eraginkorra." `https://bbvaopen4u.com/es/actualidad/api-rest-que-es-y-cuales-son-sus-ventajas-en-el-desarrollo-de-proyectos`.
10. MVC, "Software arkitektura eredua da." `https://codigofacilito.com/articulos/mvc-model-view-controller-explicado`.

11. Firebase, “Web aplikazioak eta aplikazio mugikorrak garatzeko plataforma.” `https://firebase.google.com/`.
12. RecyclerView, “ListView-ren bertsio aurreratuagoa.” `https://developer.android.com/guide/topics/ui/layout/recyclerview`.
13. Android, “Google-k garatutako mugikorrerako sistema eragile.” `https://www.android.com/`.
14. Firebase Authentification, “Backend zerbitzu errazak, SDKak eta UI liburutegiak eskaintzen ditu erabiltzaileak autentifikatzeko.” `https://firebase.google.com/docs/auth`.
15. Firebase Realtime Database, “Hodeian dagoen eta denbora errealean lan egiten duen datu basea.” `https://firebase.google.com/docs/database`.
16. JSON, “Datuak trukatzeko testu formatu sinplea.” `https://www.json.org/`.
17. JetBrains, “Software garapeneko enpresa.” `https://www.jetbrains.com/`.
18. IntelliJ IDEA, “Garapen ingurune integratua.” `https://www.jetbrains.com/es-es/idea/`.
19. Apache, “HTTP zerbitzaria garatzeko eta mantentzeko erabiltzen da.” `https://httpd.apache.org/`.
20. Java, “Programazio lengoaia bat eta ordenagailuetarako plataforma bat.” `https://docs.oracle.com/javase/tutorial/index.html`.
21. Kotlin, “Programazio lengoaia.” `https://kotlinlang.org/`.
22. C++, “Programazio lengoaia.” `http://www.cplusplus.com/`.
23. ADA, “Programazio lengoaia.” `https://www.adacore.com/`.
24. VisualAge, “Aplikazioen garapenerako tresna orokorra.” `https://www.ibm.com/support/knowledgecenter/es/ssw_ibm_i_71/rzahg/rzahgrpgvisage.htm`.
25. ACM, “Association for Computing Machinery.” `https://www.acm.org/`.
26. JUnit, “Java-n proba unitarioak egiteko liburutegiak.” `https://junit.org/`.
27. Ant, “Zeregin mekanikoak eta errepikakorrak egiteko programazio tresna.” `https://ant.apache.org/`.
28. Google Cloud Platform, “Googlek eskaintzen zituen web garapen aplikazio guztiak bildu dituen plataforma.” `https://cloud.google.com/`.

29. iOS, “Apple multinazionalaren sistema eragilea mugikorra da.”
`https://www.apple.com/es/ios`.
30. Unity, “Bideojokoen multiplataforma motorra.”
`https://unity.com/`.
31. Google Analytics, “Google enpresaren web analisi tresna.”
`analytics.google.com/analytics/web`.
32. AFAGI., “Asociación de familiares, amigos y personas con alzheimer y otras demencias de guipuzcoa.”
`http://afagi.eus`.