

RIDES24

Software Ingeniaritzaritza I

2024-2025 Ikastureta

Egileak:

Eneko Pardavila Barquilla (Scrum Master)

Oihane Pereira Juanenea

Aroa Rodríguez Villa

Itxaso Urkoitia Agirre

AURKIBIDEA

1. Sarrera.....	2
2. Eskakizunen Bilketa.....	3
2.1. Domeinuen ereuda.....	3
2.2. Erabilpen kasuak.....	4
3. Diseinua.....	6
3.1. Sekuentzia diagramak.....	6
3.2. Klase diagrama.....	8
4. Implementazioa.....	9
5. Ondorioak.....	11
Aurkitutako arazoak.....	11
Taldearen dinamika.....	11
Lortutako emaitza.....	11
Irakasgaiaren balorazioa.....	11
Etorkizunerako gomendioak.....	12
6. Bideoaren URL-a.....	12
7. Kodearen URL-a.....	12

1. Sarrera

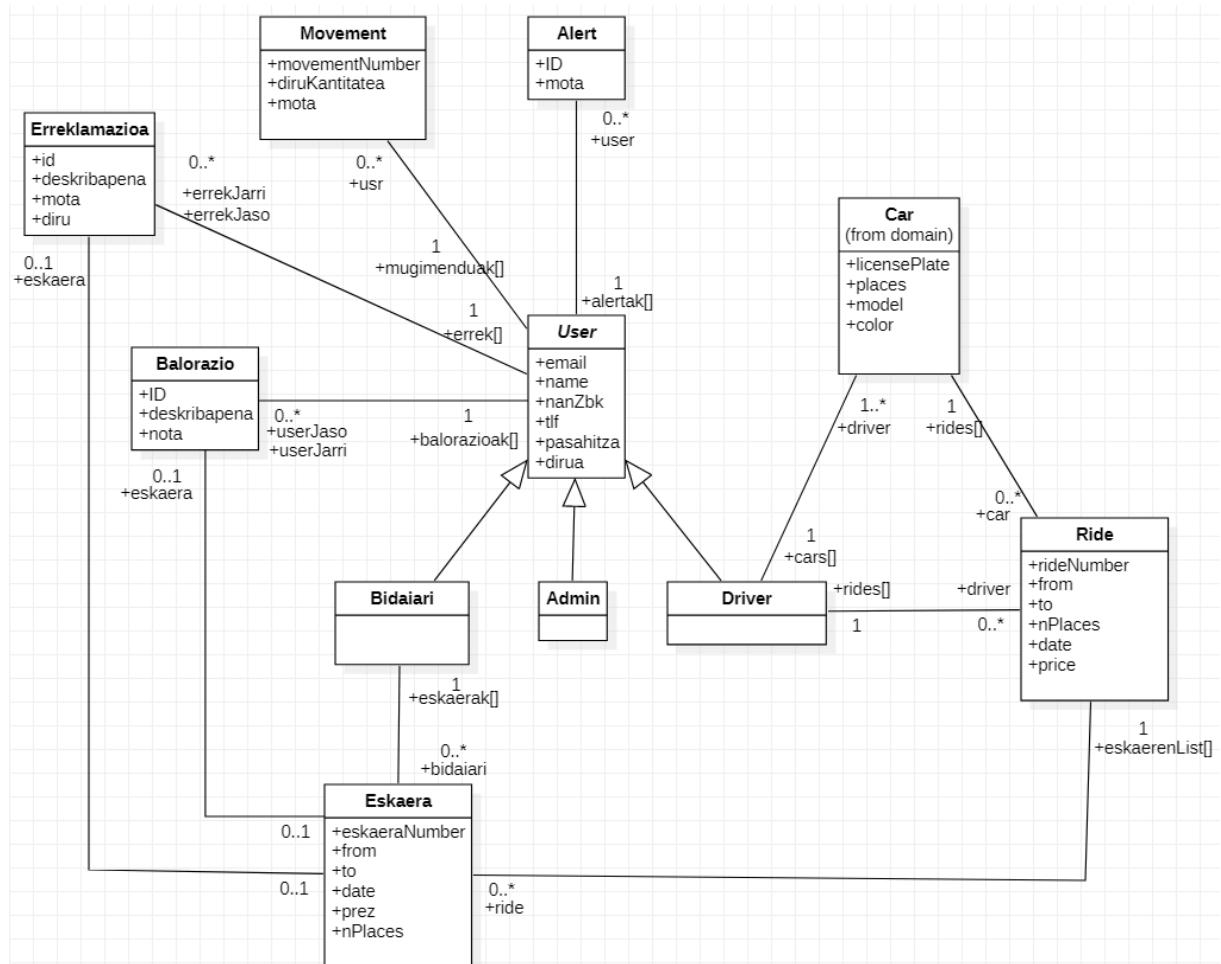
Rides24 bidaiaik partekatzeko plataforma digitala da, erabiltzaileek bidaiai edo gidari gisa jardun dezaketen aplikazio bat. Gidariak bidaia berriak sor ditzake, eta bidaiariek eskaerak egin ditzakete. Gidariak eskaerak onartu edo ukatu ditzake, eta bidaia amaitzean, bidaiaiak egindako bidaia baieztatu behar du diru-transferentzia burutu dadin.

Aplikazioak hainbat funtzionalitate eskaintzen ditu: eskaeren jarraipena, wallet mugimenduen konsulta, jakinarazpen automatikoak, balorazioak eta erreklamazioak egiteko aukera. Erabiltzaileek bere kontua ezabatu dezakete, tramite penderenterik ez badute.

Aplikazioa internazionalizatuta dago, eta euskaraz, gaztelaniaz eta ingelesez erabil daiteke, hizkuntza-aniztasuna bermatuz. Azkenean, ez ditugu web zerbitzuak implementatu.

2. Eskakizunen Bilketa

2.1. Domeinuen ereuda



User

Edozein erabiltzaile izan daiteke (bidaaria, gidaria edo administratzailea). Erabiltzaileak honako elementuekin erlazionatuta egon daitezke:

- Erreklamazio bat edo gehiago (1:N)
- Balorazio bat edo gehiago (1:N)
- Mugimendu ekonomiko bat edo gehiago (1:N)
- Alerta bat edo gehiago (1:N)

Driver

User motako azpimota da. Bere erlazioak hauek dira:

- Gidari batek kotxe bat edo gehiago izan ditzake (1:N)
- Gidari batek bidaia (Ride) bat edo gehiago eskaini ditzake (1:N)
- Kotxe bakoitzak bidaia bat baino gehiago egin ditzake (1:N)

Bidaiai

Hau ere User motako azpimota da. Honako erlazioak ditu:

- Bidaiai batek eskaera bat edo gehiago egin ditzake (1:N)
- Eskatutako bidaia bakoitzarekin:
- Erreklamazio bat egin dezake (0:1)
- Balorazio bat eman dezake (0:1)

Car

Kotxe bakoitza gidari bati dagokio eta kotxe batek bidaia bat edo gehiago egin ditzake (1:N)

Ride

Bidaia bat gidari batek eskaintzen du eta:

- Bidaia batean eskari (Eskera) bat edo gehiago egon daitezke (1:N)
- Bidaia bakoitza kotxe batean egiten da

Eskaera

Bidaiai batek egindako eskaera bat da. Erlazioak:

- Eskera bat bidaia bati dagokio (N:1)
- Eskera batek balorazio edo erreklamazio bat izan dezake (0:1 bakoitza)

Balorazio

Eskaera bati edo erabiltzaile bati dagokio eta nota bat eta deskribapena dauzka.

erreklamazioa

Erabiltzaileak bidaiaaren edo eskaeraren inguruan aurkeztutako kexa da. Mota, deskribapena eta diru kopurua jasotzen ditu.

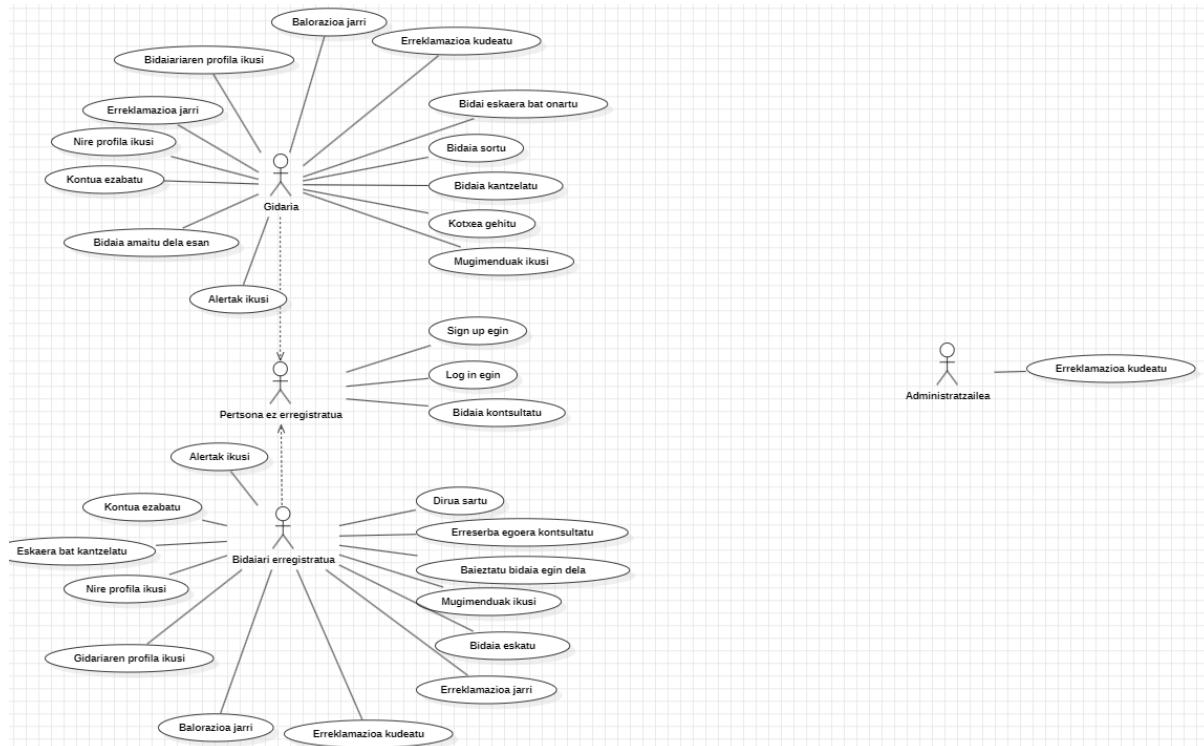
Movement

Diru mugimendu bat da. Erabiltzaile bati lotuta dago eta mugimendu bakoitzak zenbatekoa, mota eta identifikatzaila bat dauzka.

Alert

Sistemak erabiltzaileari bidalitako abisua da. Mota bat eta identifikatzaila bat jasotzen ditu.

2.2. Erabilpen kasuak



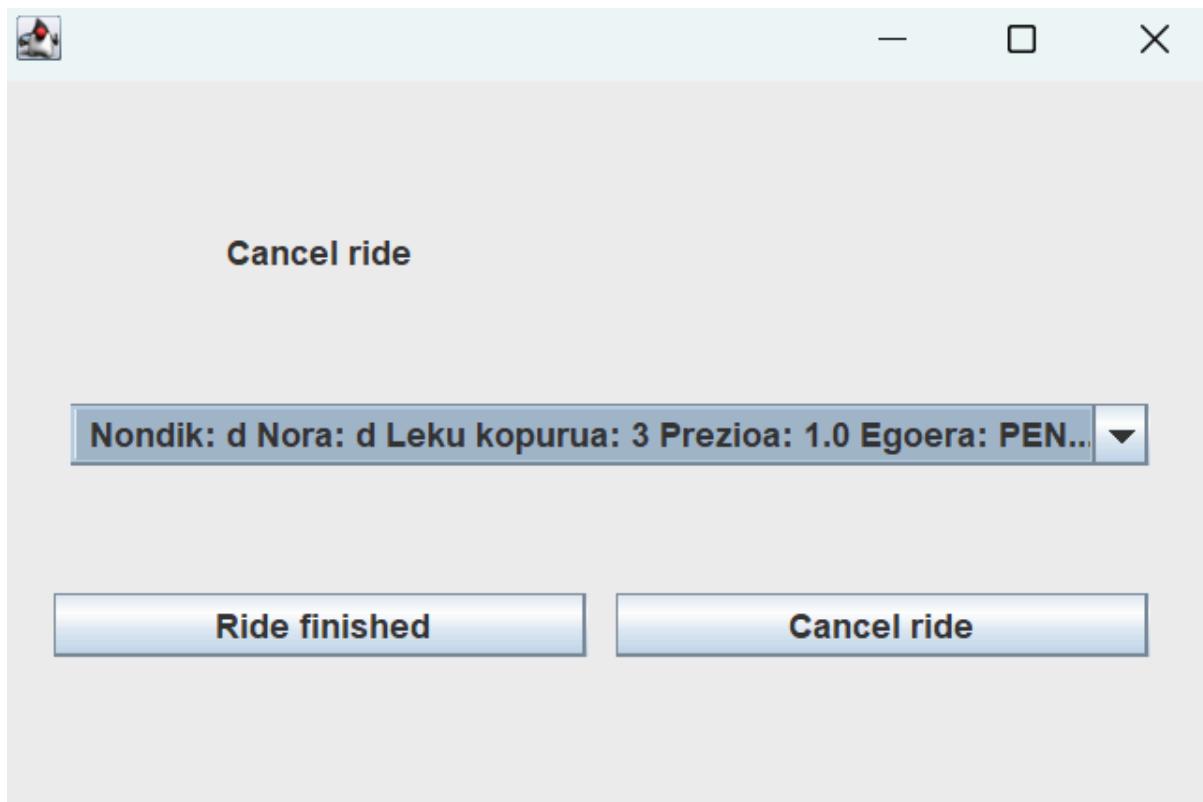
Bidaia amaitu dela esan

Hau gidariak egiten du.

Bidaia amaitu dela esan Flow of events

Basic Flow

- *Gidariak* ride bat aukeratuko du eta btnFinished sakatuko du.
- *Sistemak* ride horren egoera FINISHED bezala jarriko du.
- *Sistemak* ride horren onartutako eskaera guztiak FINISHED-en jarriko ditu.
- *Sistemak* alerta bat bidaliko die ride horretan eskaera egin duten bidaiariei.



Kontua ezabatu

Hau gidariak zein bidaiaiak egiten du

Kontua ezabatu Flow of events

Basic Flow

1. *Userrak* kontua ezabatzeko botoia klikatuko du.
2. *Sistemak* Driver edo Bidaiai den begiratu behar du.
3. Driver bada, *Sistemak* bere bidai guztiak kantzelatuko ditu.
4. Bidaiai bada, *Sistemak* bere eskaera guztiak kantzelatuko ditu.

Alternative Flow

1. Bidaiai bat ezabatu behar bada eta rideenbat FINISHE-en badago, ezin da ezabatu bidaia ordaindu behar duelako.



Nire profila ikusi

Hau gidariak zein bidaiaiak egiten du

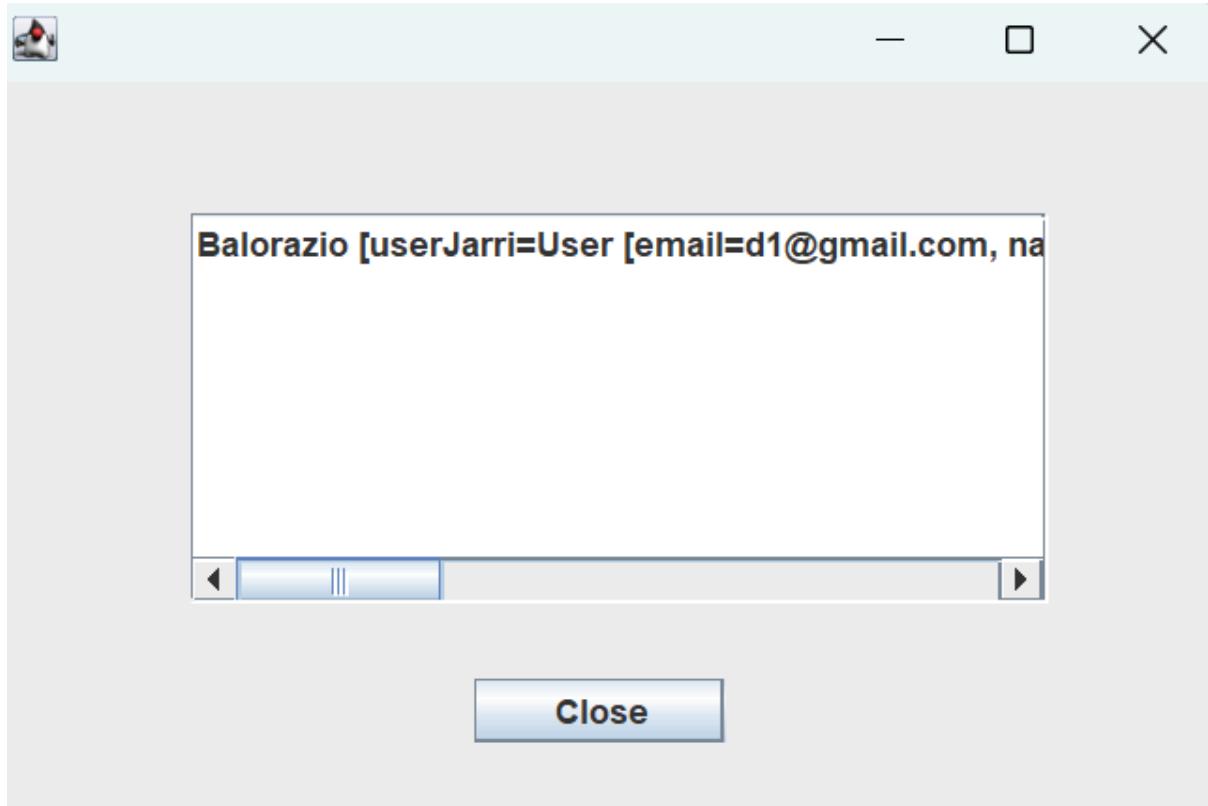
Nire profila ikusi Flow of events

Basic Flow

1. *Userra* nireBalor botoia klikatuko du.

2. *Sistemak* ProfilaGUI lehioa irekiko du eta bertan userraren balorazio guztiak agertuko dira.

3. *Userrak* balorazio guztiak ikusiko ditu.



Erreklamazioa Jarri

Hau gidariak eta bidaiaiak egiten du. Interfase grafikoa desberdina da.

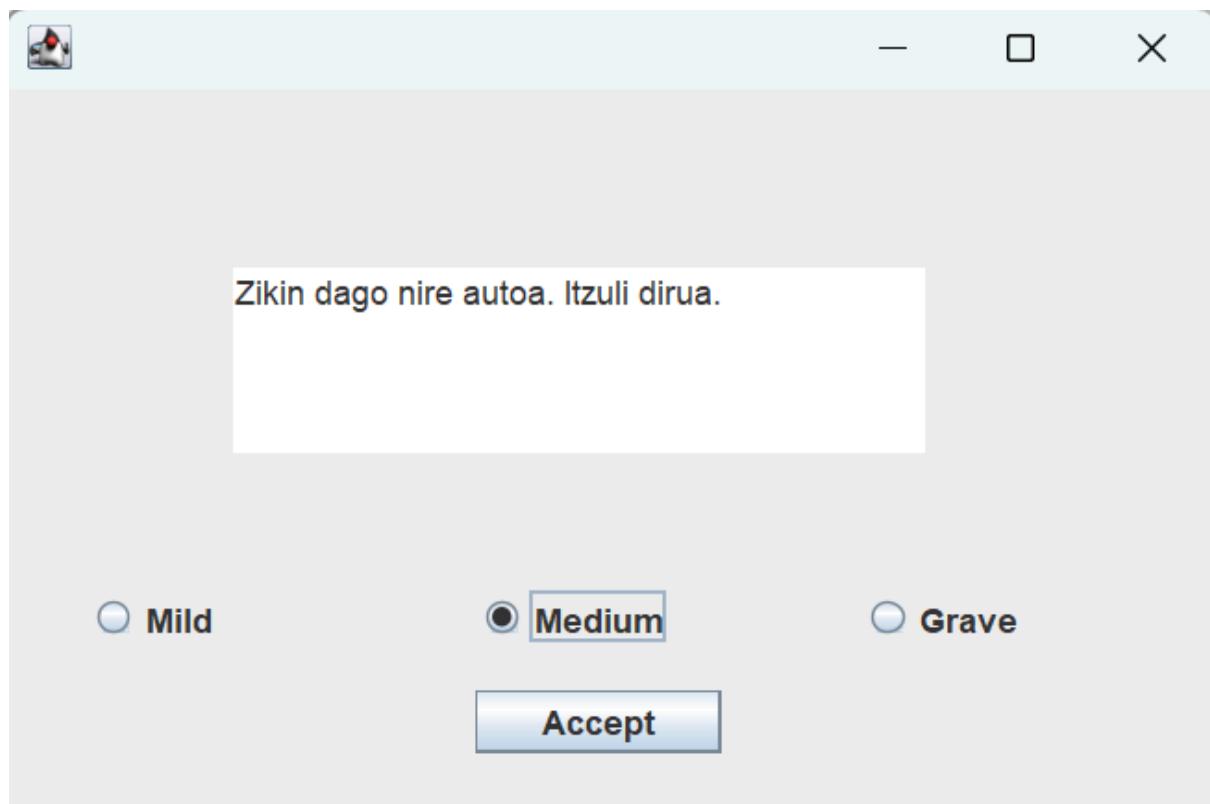
Erreklamazioa Jarri Flow of events

Basic Flow

1. *Gidariak* **erreklamazioa jartzeko GUI-a** irekitzen du, erlazionatutako eskaera eta beste erabiltzailea hautatuz.
2. *Gidariak* **erreklamazioaren deskribapena** idazten du.
3. *Gidariak* erreklamazioa **aurkezten du**.
4. *Sistemak* erreklamazioa sortzen du, dagokion informazioarekin eta egoera **PENDING**-ean, eta datu-basean gordetzen du.
5. *Sistemak* **alerta bat** sortzen du erreklamazioa jaso duen erabiltzailearentzat.
6. *Sistemak* erabiltzaileari baieztapena erakusten dio/reklamazioaren GUI-a ixten du.

Alternative flow

1. Ez dira derrigorrezko datu guztiak sartu (deskribapena, erlazionatutako eskaera...). Sistemak erabiltzaileari abisua ematen dio eta ez da erreklamazioa sortzen.
2. Erreklamazioa jartzen saiatzen denean errore bat gertatzen da. Sistemak errore-mezua bistaratzen dio erabiltzaileari.



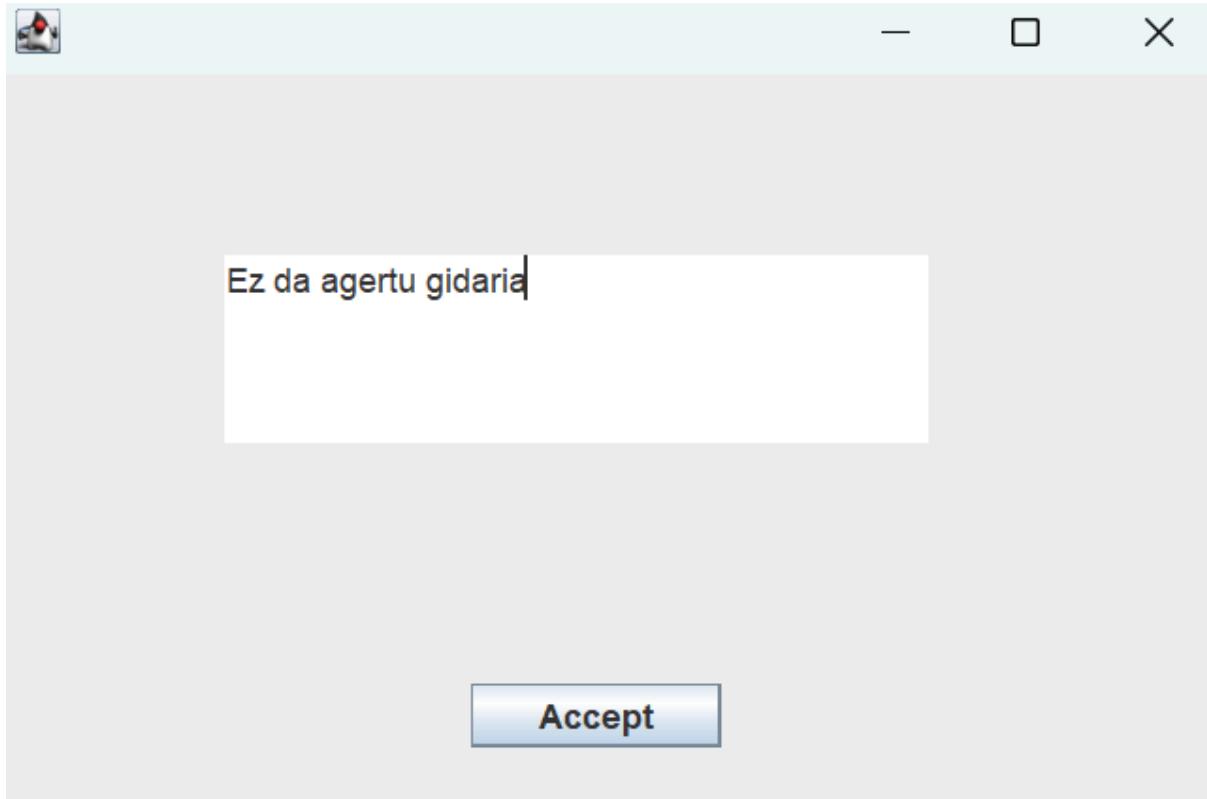
Erreklamazioa Jarri Flow of events

Basic Flow

1. *Bidaia* **erreklamazioa jartzeko GUI-a** irekitzen du, erlazionatutako eskaera eta beste erabiltzailea hautatuz.
2. *Bidaia* **erreklamazioaren deskribapena** idazten du.
3. *Bidaia* erreklamazioa **aurkezten du**.
4. *Sistemak* erreklamazioa sortzen du, dagokion informazioarekin eta egoera **PENDING**-ean, eta datu-basean gordetzen du.
5. *Sistemak* **alerta bat** sortzen du erreklamazioa jaso duen erabiltzailearentzat.
6. *Sistemak* erabiltzaileari baieztapena erakusten dio/reklamazioaren GUI-a ixten du.

Alternative flow

1. Ez dira derrigorrezko datu guztiak sartu (deskribapena, erlazionatutako eskaera...). Sistemak erabiltzaileari abisua ematen dio eta ez da erreklamazioa sortzen.
2. Erreklamazioa jartzen saiatzen denean errore bat gertatzen da. Sistemak errore-mezua bistaratzen dio erabiltzaileari.



Balorazioa jarri

Hau gidariak eta bidaiaiak egiten du. Interfase grafikoa desberdina da.

Balorazioa jarri Flow of events

Basic Flow

1. *Userrak* egindako bidaia baloratuko du JRadioButton bat aukeratuz.
2. *Userrak* textArean berak nahi duen deskribapena idatziko du.
3. *Userrak* baloratu botoia klikatuko du.
4. *Sistemak* balorazioa gehituko du baloratua izan den userraren listan.
5. *Sistemak* baloratua izan den eskaera VALUED bezala jarriko du.
6. *Sistemak* alerta bat bidaliko dio baloratua izan den userrari.

Alternative flow

1. Itxi botoia klikatzen badu ez da balorazioa egingo.
2. Ez badago JRadioButton-ik klikatuta errore bat pantailaratuko da.



Bidaia eskaera bat onartu

Hau gidariak egiten du.

Bidaia eskaera bat onartu Flow of events

Basic Flow

1. *Gidariak* bidaia eskaera bat jasotzen du.
2. *Sistemak* jasotako eskaeren lista erakusten du.
3. *Gidariak* listako eskaera bat onartzen du.
4. *Sistemak* eskaera baiezatzen du eta egoera eguneraztzen du.

Alternative flow

1. Ez daude bidaia eskaerarik. Eskaeren atala hutsik egongo da.
2. Gidariak eskaera ukatzen du.



Create Ride

Hau gidariak egiten du.

Create Ride Flow of events

Basic Flow

1. *Driver* insert the **departing city**, **arrival city**, **number of seats** and **price**
2. *Driver* selects a **ride date**
3. *System* creates a **ride** with inserted data and assigns to the **Driver**.

Alternative flow

1. Some of fields are empty. Final.
2. **number of seat** or **price** is not a number. System informs the user.
3. **Ride date** is before today. System informs the user.
4. the **ride** already exist for this **driver**. Ride is not created. System informs the user.

Create Ride

Ride date

Depart City	Arrasate	May	2025						
Arrival City	Donostia	Sun	Mon	Tue	Wed	Thu	Fri	Sat	
		18	1	2	3				
		19	4	5	6	7	8	9	10
		20	11	12	13	14	15	16	17
		21	18	19	20	21	22	23	24
		22	25	26	27	28	29	30	31

Choose car

[licensePlate=1, places=3, model=a]

Price

Create Ride **Close**

Bidaia kantzelatu

Hau gidariak egiten du.

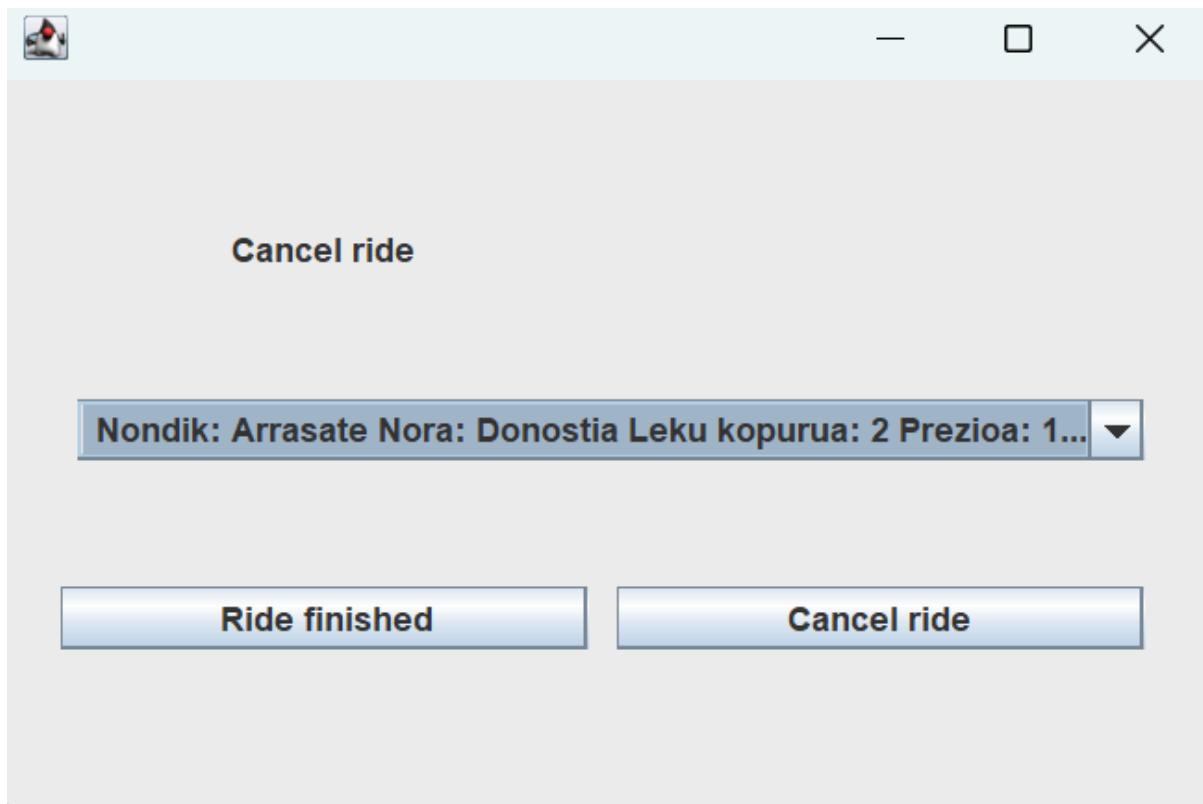
Bidaia kantzelatu Flow of events

Basic Flow

1. *Gidariak* bidaia bat aukeratzen du bere bidai-zerrendatik.
2. *Gidariak* aukeratutako bidaia kantzelatzen du.
3. *Sistemak* bidaia egoera **kantzelatuta** bezala markatzen du.

Alternative flow

1. Gidariak ez du bidaik. Bidai-zerrenda hutsik egongo da.
2. Bidaia dagoeneko hasi edo egin da, ezin da kantzelatu.



Kotxea gehitu

Hau gidariak egiten du.

Kotxea gehitu Flow of events

Basic Flow

1. *Gidariak* kotxearen datuak sartzen ditu: **marka**, **modelo** eta **eserleku kopurua**.

2. *Sistemak* kotxe berriaren datuak gordetzen ditu.

Alternative flow

1. Sartutako daturen bat ez da zuzena edo kotxeaa dagoeneko existitzen da.

The screenshot shows a Windows application window titled "Add Car". The window has a standard title bar with minimize, maximize, and close buttons. Inside, there are four input fields and a central button. The first field is labeled "License Plate:" with the value "1234". The second field is labeled "Model:" with the value "audi". The third field is labeled "Number of seats:" with the value "3". The fourth field is labeled "Color:" with the value "black". Below these fields is a large blue button labeled "Add Car".

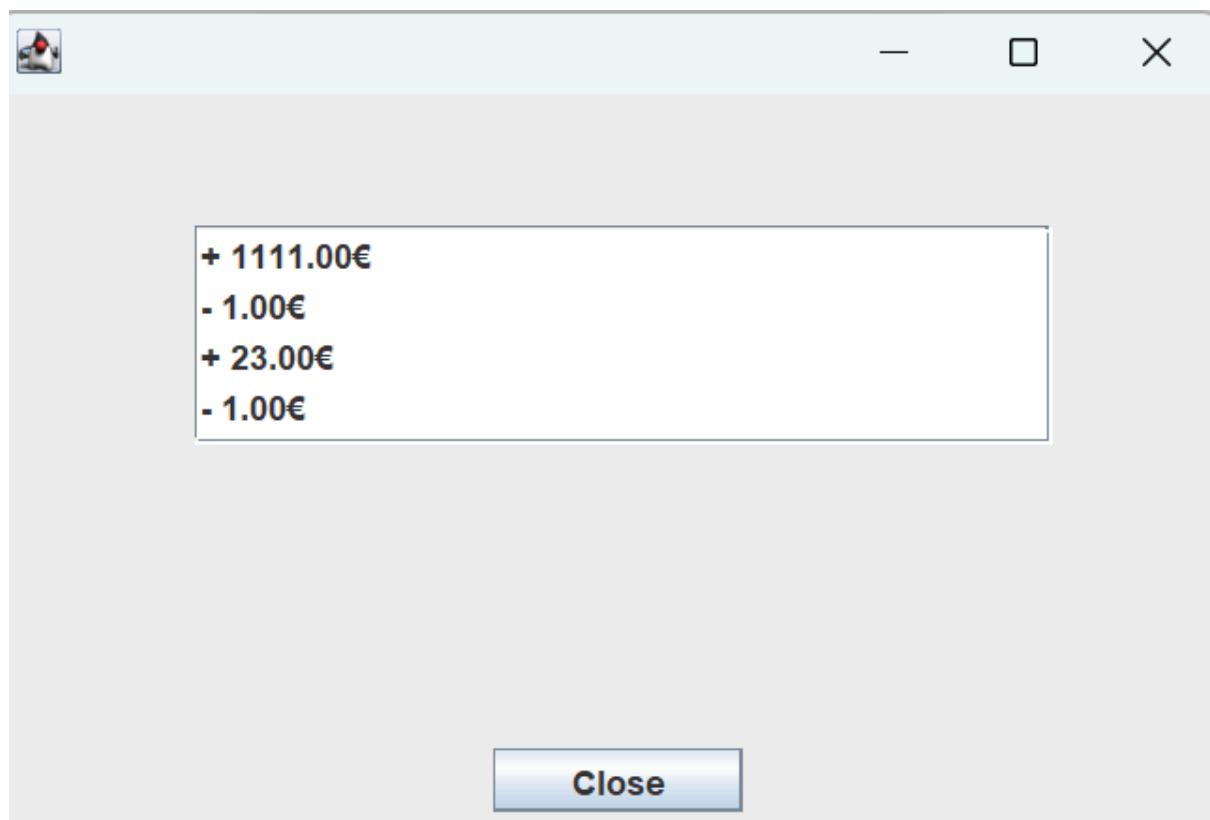
Mugimenduak ikusi

Hau gidari zein bidaiaiak egiten du

Mugimenduak ikusi Flow of events

Basic Flow

1. *Gidariak* diru-zorroko mugimenduak ikusteko eskaera egiten du.
2. *Sistemak* transkazioko edo mugimendu guztiak erakusten ditu, bere xehetasunekin (dirua sartu edo atera den).



Sing up

Hau erregistratu gabeko pertsonak egiten du

Sing up Flow of events

Basic Flow

1. **Pertsona ez erregistratutak** bere datuak sartuko ditu pantailan.
2. Gidaria edo bidaiaia den adieraziko du.
3. **Sistemak** bere datuak gordeko ditu datu-basean.

Alternative flow

1. Atal batzuk hutsik daude. Final.
2. Email hori datu-basean gordeta dago jada.

 Sing up

Sing up

Name-Surname:	a
ID number:	1
Tel. number:	1
Email:	d1@gmail.com
Password:	1

What will you be? Driver Passenger

Enter

Log in

Hau erregistratu gabeko pertsonak egiten du

Log in Flow of events

Basic Flow

1. **Pertsona ez erregistratutak** bere emaila eta pasahitza sartuko ditu pantailan.
2. **Sistemak** bere datuak kontsultatuko ditu datu-basean.

Alternative flow

1. Atal batzuk hutsik daude. Final.
2. Email hori ez da existitzen datu-basean.
3. Pasahitza ez da zuzena. Sistemak adieraziko du.

 Log in

Log in

Email:	d1@gmail.com
Password:	1

Enter

Query Rides

Hau erregistratu gabeko pertsonak egiten du

Query Rides Flow of events

Basic Flow

1. *System* displays all **cities** where **rides** depart from
2. *Driver* selects a departing **city**
3. *System* displays all destination **cities** for **rides** that depart from a **selected city**.
4. *Driver* selects an arrival **city**
5. *System* highlights in a Calendar the days where **rides** exist from the **depart** to destination **cities** in a selected month
6. *Driver* selects a **date** in a Calendar
7. *System* displays the **rides** from the selected **departin city** to the selected **arrival city** on that **date**.

Alternative flow

1. There are no **rides** on the selected **date**. **Rides** display is empty.

The screenshot shows a software window titled "Find rides". On the left, there are two dropdown menus: "Depart City" set to "Arrasate" and "Arrival C..." set to "Donostia". To the right of these is a "Ride date" section containing a calendar for May 2025. The calendar highlights specific dates: May 18th is red, May 19th is blue, May 21st is red, and May 22nd is blue. The date "May 21, 2025" is also explicitly labeled below the calendar. Below the calendar is a table titled "Viajes: May 21, 2025" with one row showing "Driver" "a", "Seats" "2", and "Price" "1.0". At the bottom of the window is a "Close" button.

Bidaia erreserbatu

Hau bidaiaiak egiten du

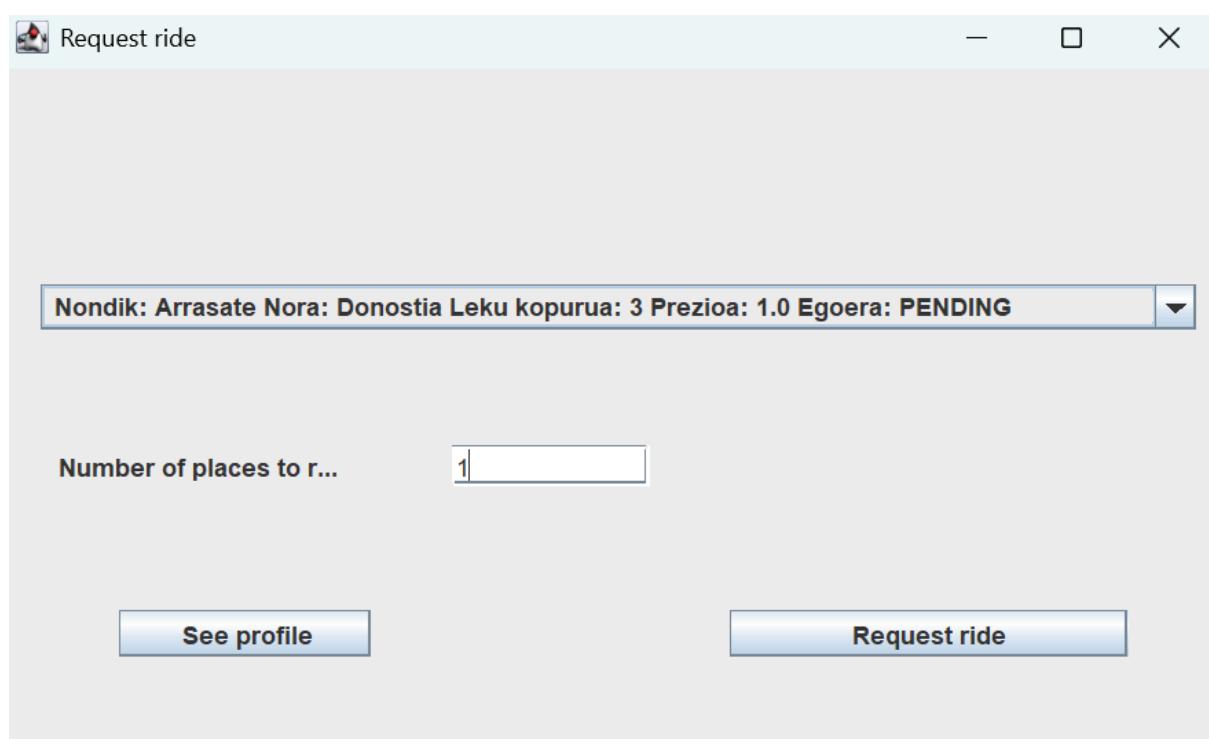
Bidaia erreserbatu Flow of events

Basic Flow

1. *Bidaiaiak* **jatorriko hiria**, **helburuko hiria** eta **data** bat jarriko du.
2. *Sistemak*, *bidaiaiaren* beharretara hoheren egokitzten diren bidaiaiak pantailaratuko ditu.
3. *Bidaiaiak* hoheren datorkion bidaia erreserbatuko du.
4. *Sistemak* erreserba bat sortuko du bidaiai horrentzako aukeratutako bidaian eserleku kopuruarekin eta egoera **pending**.

Alternative flow

1. Ez dago bidaiaiaren nahiak betetzen dituen bidairik. Sistemak jakinaraziko dio erabiltzaileari.
2. **data** ez da egokia. Sistemak erabiltzaileari jakinarazi.
3. Ez daude datu guztiak sartuta. Final



Dirua sartu

Hau bidaiaiak egiten du

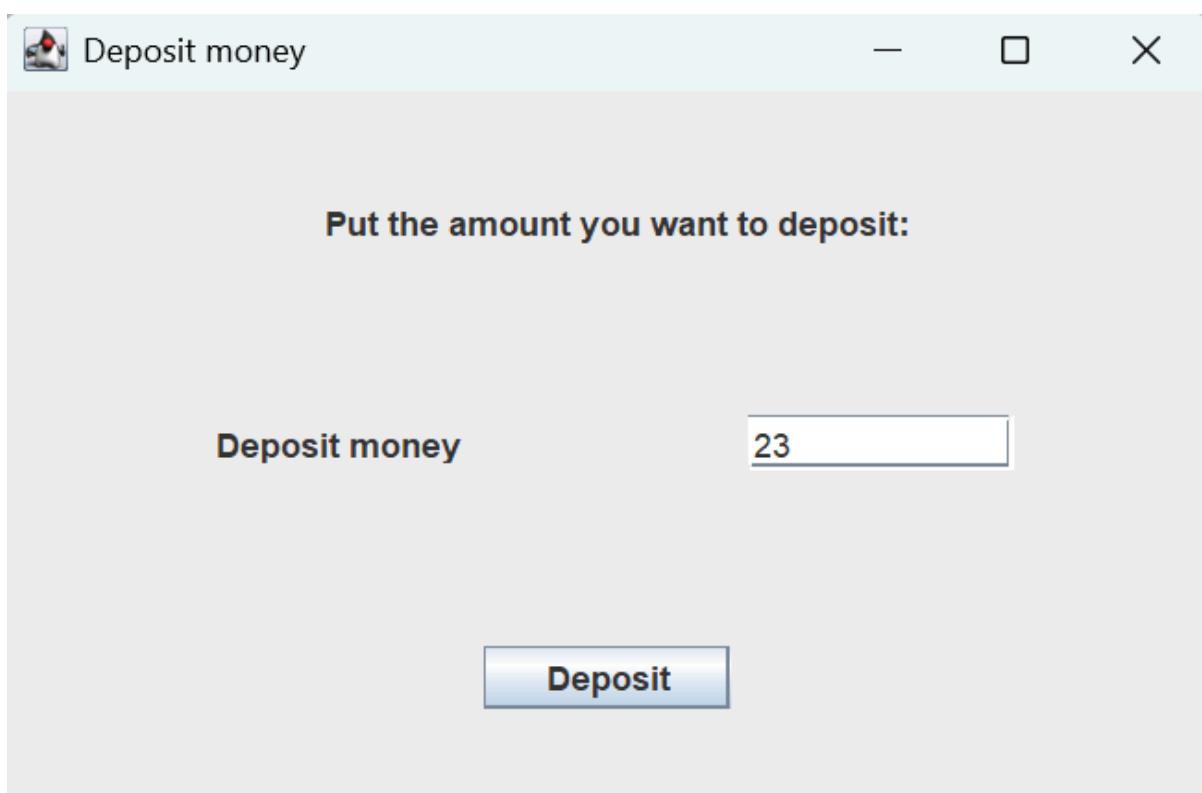
Dirua saru Flow of events

Basic Flow

1. **Bidaiaiak** bere kontuan dirua sartzen du bidaia ordaintzeko.
2. **Sistemak** bidaiai horren saldoa eguneratuko du.

Alternative Flow

1. Sartutako kantitatea ez da baliozkoa.



Erreserba egoera kontsultatu

Hau bidaiaiak egiten du

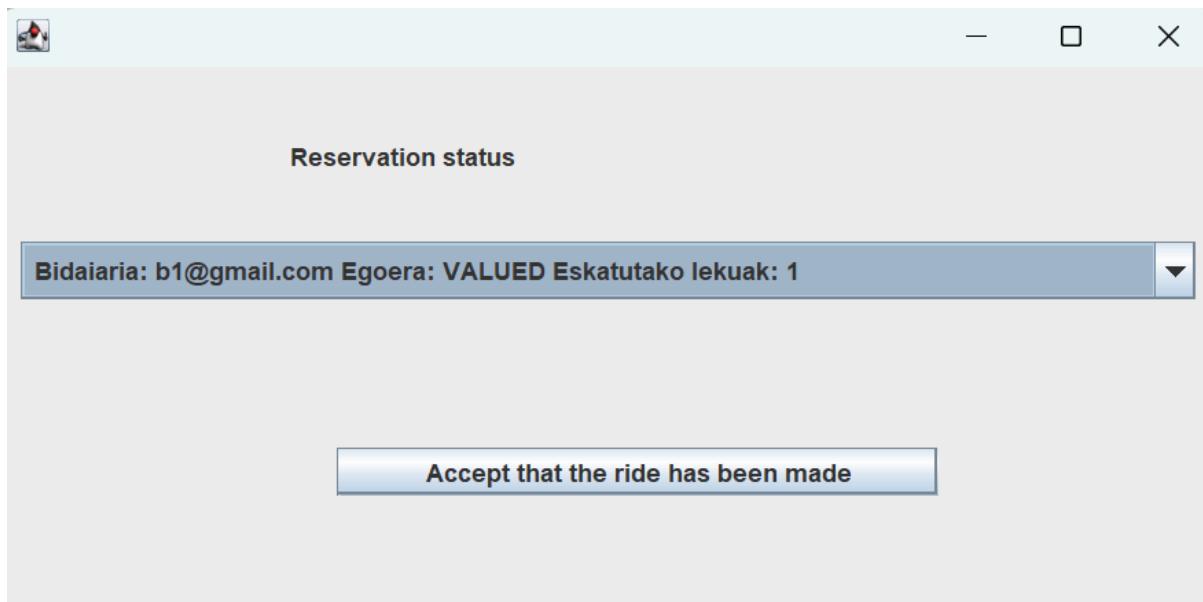
Erreserba egoera kontsultatu Flow of events

Basic Flow

1. *Bidaiaiak* bere erreserbaren egoera ikusteko eskaera egiten du.
2. *Sistemak* erreserba horren momentuko egoera erakusten du: **pending**, **baieztatuta** edo **ezeztatuta**.

Alternative flow

1. Bidariariak ez du erreserbarik egin, ez dago erreserbarik.



Baiezztatu bidaia egin dela

Hau bidaariak egiten du

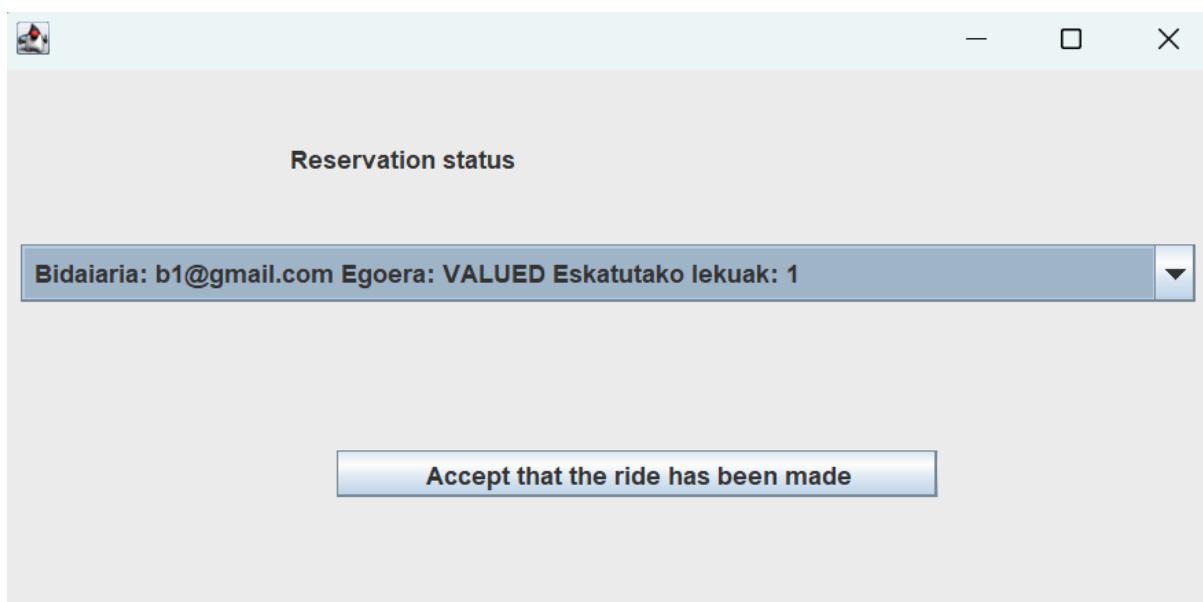
Baiezztatu bidaia egin dela Flow of events

Basic Flow

1. *Bidaariak* bidaia egin dela baieztatzen du.
2. *Sistemak* erreserba **eginda** egoerara aldatzen du.

Alternative flow

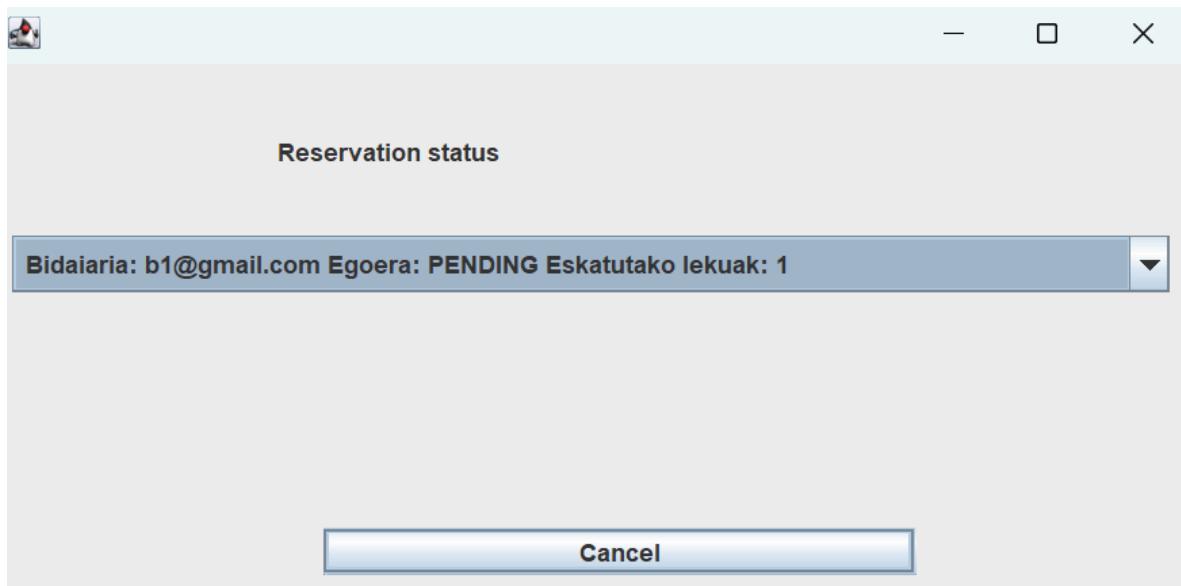
1. Bidaariak ez du bidaia egin.
2. Sistemak erreserba **ezeztatuta** egoerara aldatzen du.



Eskaera bat kantzelatu

Hau bidaiaiak egiten du

- ```
Eskaera bat kantzelatu Flow of events
Basic Flow
1. *Userra* ErrEgKonGUI-n egongo da eta JComboBoxetik berak egindako erreserba bat aukeratuko du. Erreserba hori kantzelatu nahi badu kantzelatu botoia klikatuko du.
2. *Sistemak* eskaera PENDING egoeran badago CANCELLED jarriko du.
3. Eskaera aldiz ACCEPTED badago, *Sistemak* dirua itzuliko dio bidaiaiari.
4. *Sistemak* kantzelatu den bidaian leku bat gehituko du.
5. *Sistemak* ACCEPTED egotetik CANCELLED egotera pasatuko du eskaera.
```



## Erreklamazioa kudeatu

Hau administratzaleak egiten du.

```
Erreklamazioa kudeatu Flow of events
```

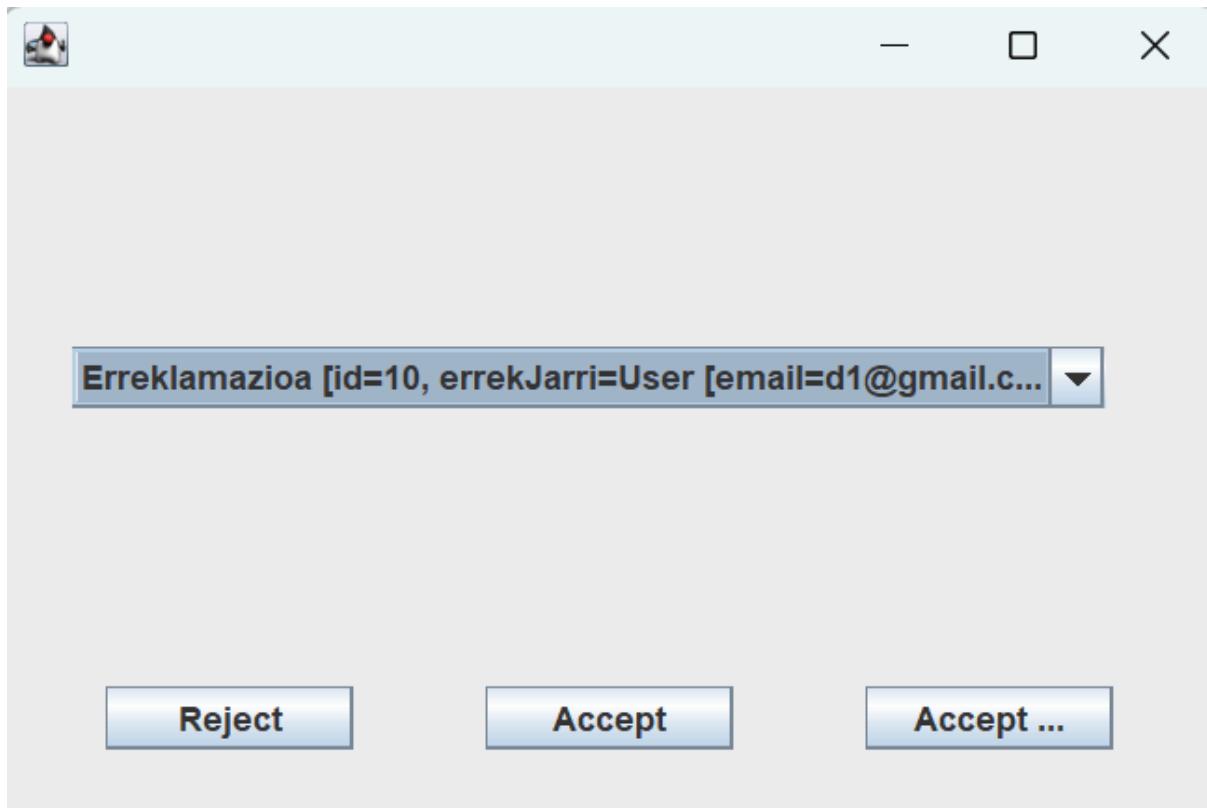
```
Basic Flow
```

1. \*Bidaiaiak\* jasotako \*\*erreklamazioen zerrenda\*\* kontsultatzen du.
2. \*Bidaiaiak\* erreklamazio bat hautatzen du eta \*\*erabaki bat hartzen du\*\*: onartu (accept), aldaketekin onartu (accept with change) edo deuseztatu (reject).
3. Onartuz gero, \*sistemak\* diru-mugimenduak eta egoera-aldeketa egiten ditu, eta \*\*alerta bat\*\* bidaltzen du inplikatuei.

4. Ezetsiz gero, \*sistemak\* erreklamazioa \*\*ADMIN\*\* egoerara pasatzen du, eta \*\*alerta bat\*\* sortzen du kexaren jarritakoarentzat.

#### ## Alternative flows

1. \*\*Erabiltzaileak ez du erabaki bat hartzen edo prozesua uzten du\*\*: Sistemak ez du aldaketarik egiten.
2. \*\*Errore tekniko bat gertatzen da (datu-basea, sarrera baliogabea...)\*\*: Sistemak errore-mezua erakusten dio erabiltzaileari eta ez du aldaketarik egiten.



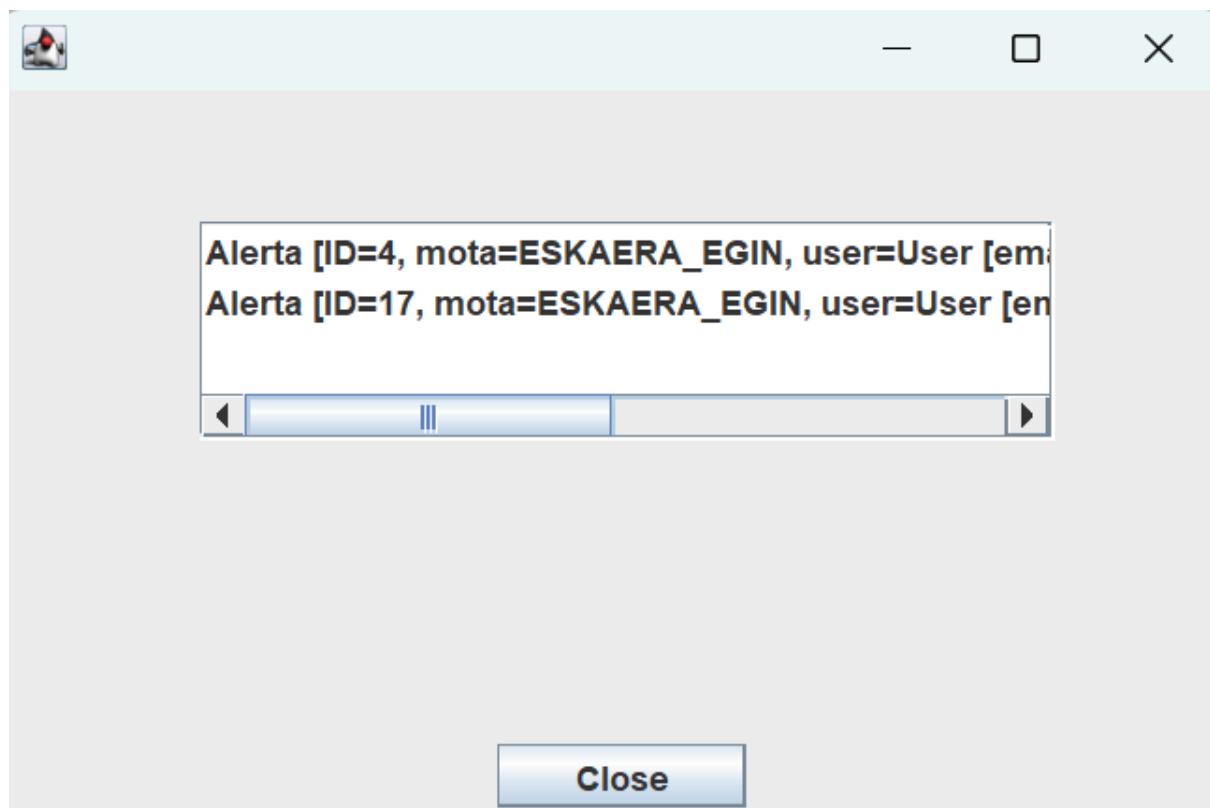
#### Alertak ikusi

Bidaia zein gidariekin egiten dute hau

# Alertak ikusi Flow of events

#### ## Basic Flow

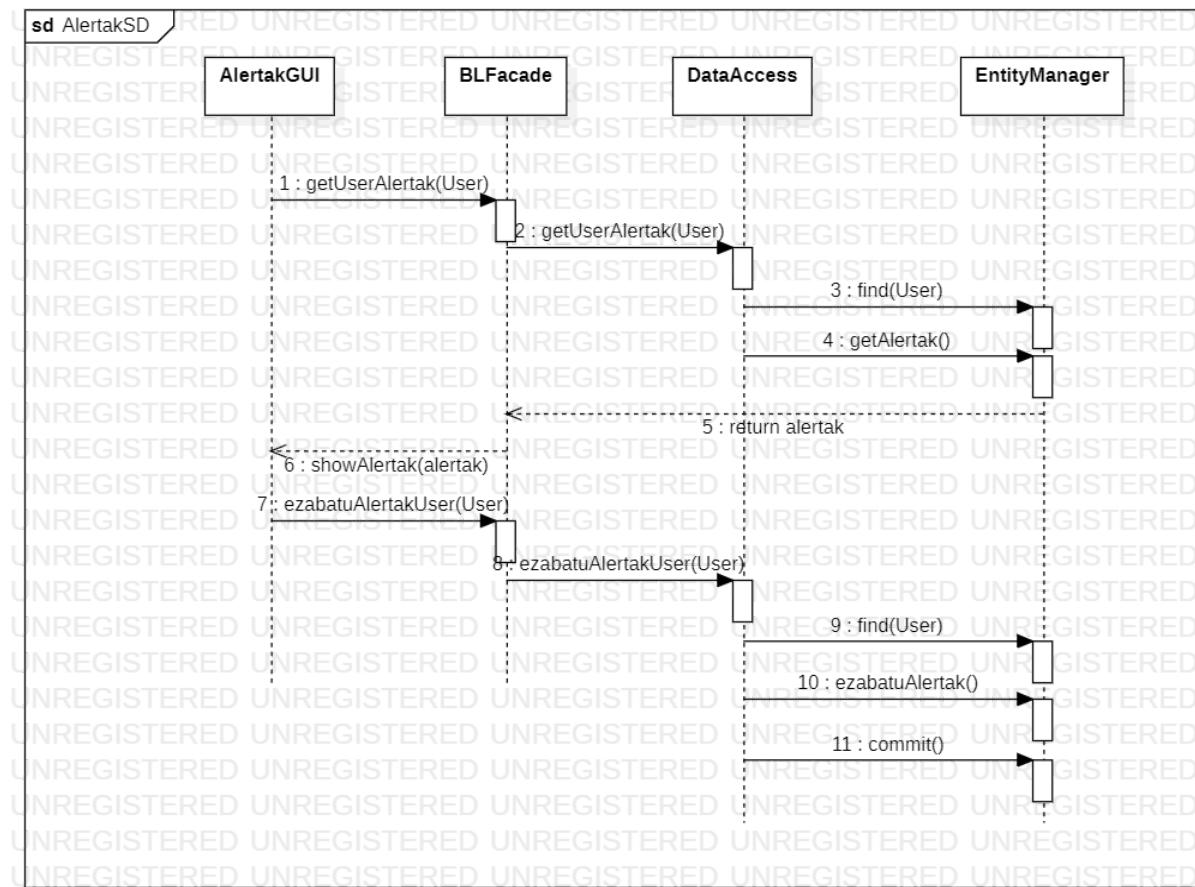
1. \*Userrak\* alertak ikusteko eskaera egiten du.
2. \*Sistemak\* userrak jaso dituen alerta guztiak erakusten ditu, bere xehetasunekin.



### 3. Diseinua

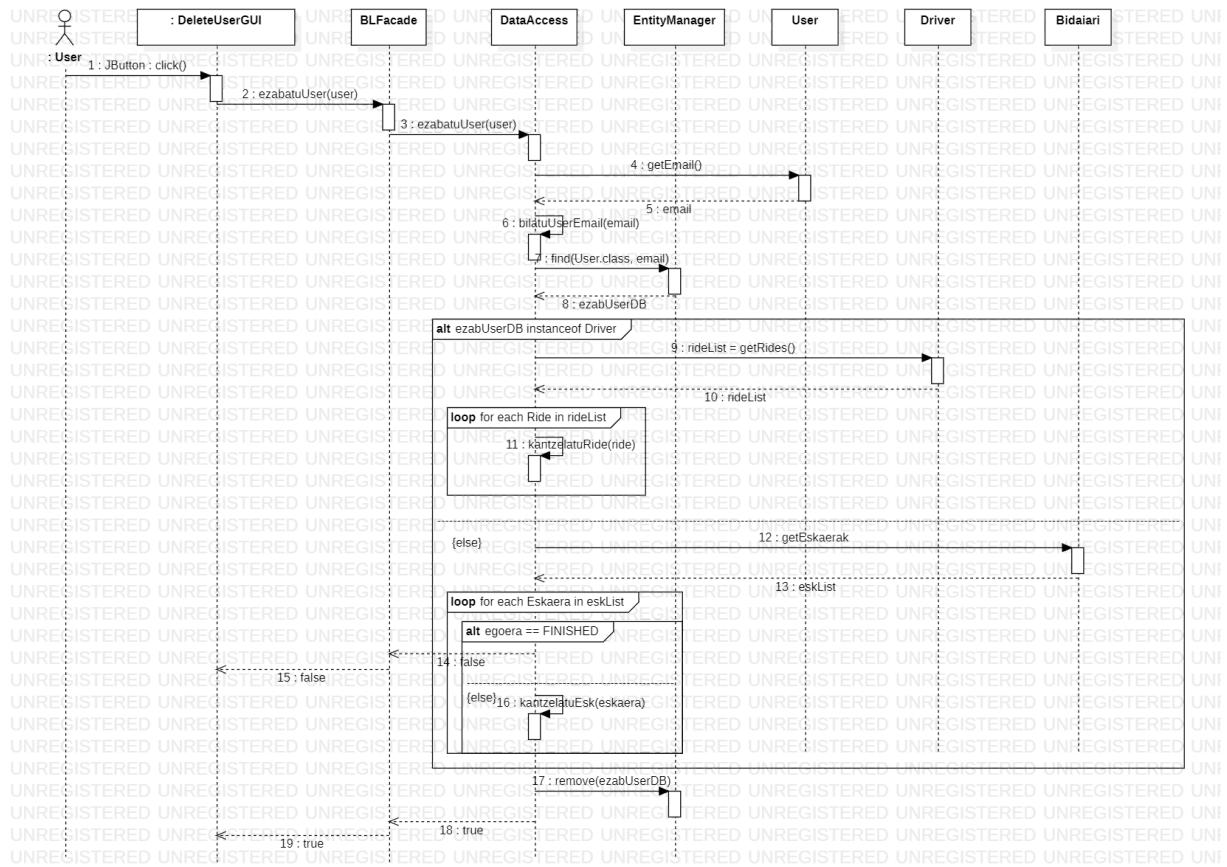
#### 3.1. Sekuentzia diagramak

##### AlertakSD



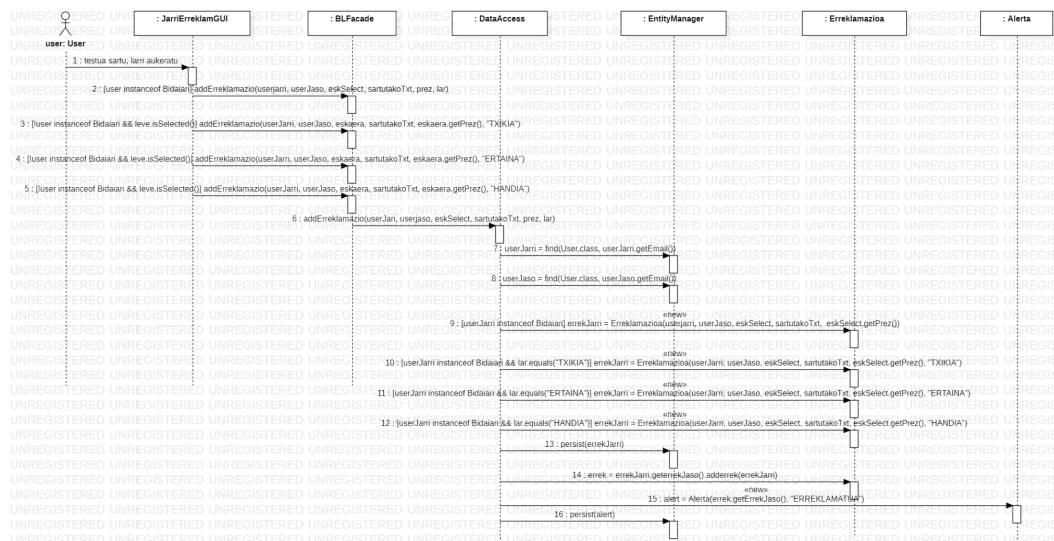
Alertak sistema erabiltzailearen alertak kudeatzeko prozesua erakusten du. AlertakGUI interfazeak erabiltzaileari alertak erakusten dizkio eta ezabatzeko aukera ematen dio. Erabiltzailearen alertak lortzen eta erakusten dira, behin ikusita, alertak ezabatzen dira.

## EzabatuUserSD



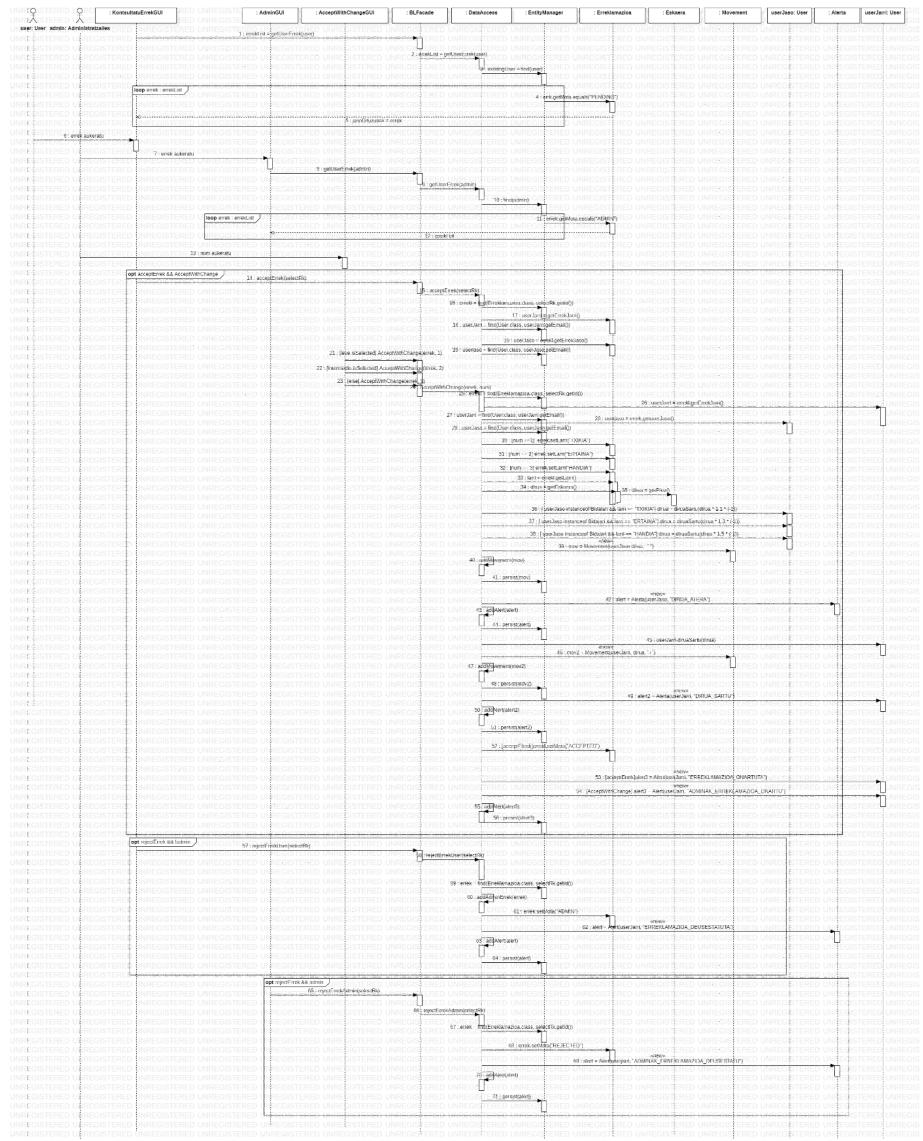
Erabiltzaile ezabaketa sistema erakusten du. DeleteUserGUI interfazeak erabiltzaile bat ezabatzeko prozesua abiarazten du. Erabiltzailea datu-basean bilatzen da datu-basean. Erabiltzailea Driver (gidaria) bada, bere bidaien zerrenda prozesatzen da. Bestela, ssaikerak prozesatzen dira. Azkenik, erabiltzailea datu-baseatik ezabatzen da.

## ErreklamazioaJestiSD



Erreklamazio sistema bidaia baten inguruan erreklamazio bat egiteko prozesua erakusten du. Bi erabiltzaile lotzen ditu (jarri eta jaso) eta larritasunaren arabera sailkatzen da. Erreklamazioa datu-basean gordetzen da eta alertak sortzen dira.

## ErreklamazioaKudeatuSD

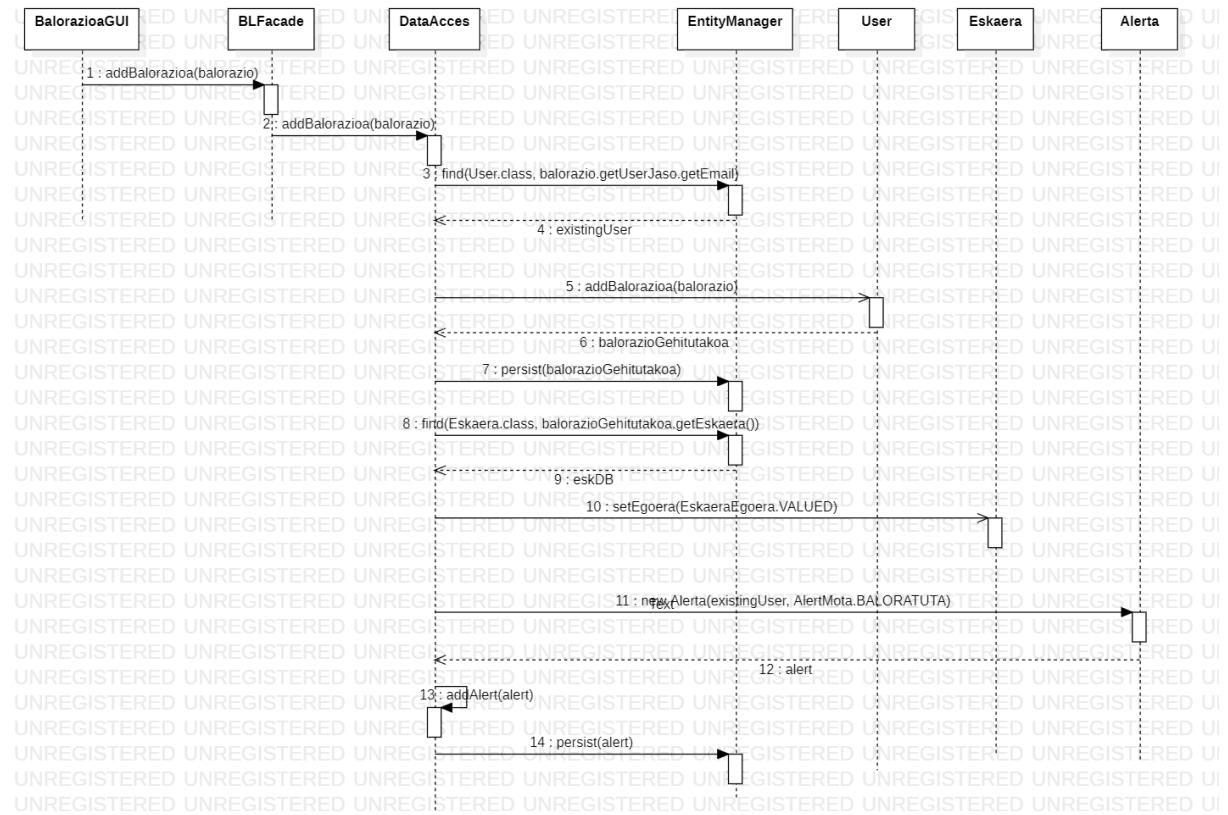


Erreklamazioa kudeatzean hiru aukera daude:

1. Bidaaria eta gidaria haien artean konpontzen dira eta erreklamazioa onartzen dute. Hau gertatzen bada, erreklamazioa jarri duenari diru-transferentzia bat egingo zaio, bidaiaaren prezioa gehi larritasunaren araberako gehikuntza.
2. Ez badira konpontzen, erreklamazioa administratzailarei goratuko zaio.
3. Administratzailak erreklamazioa onartu, aldatu eta deuseztatu dezake. Aldatzen badu, automatikoki onartuko da, larritasun berriari egokituz.

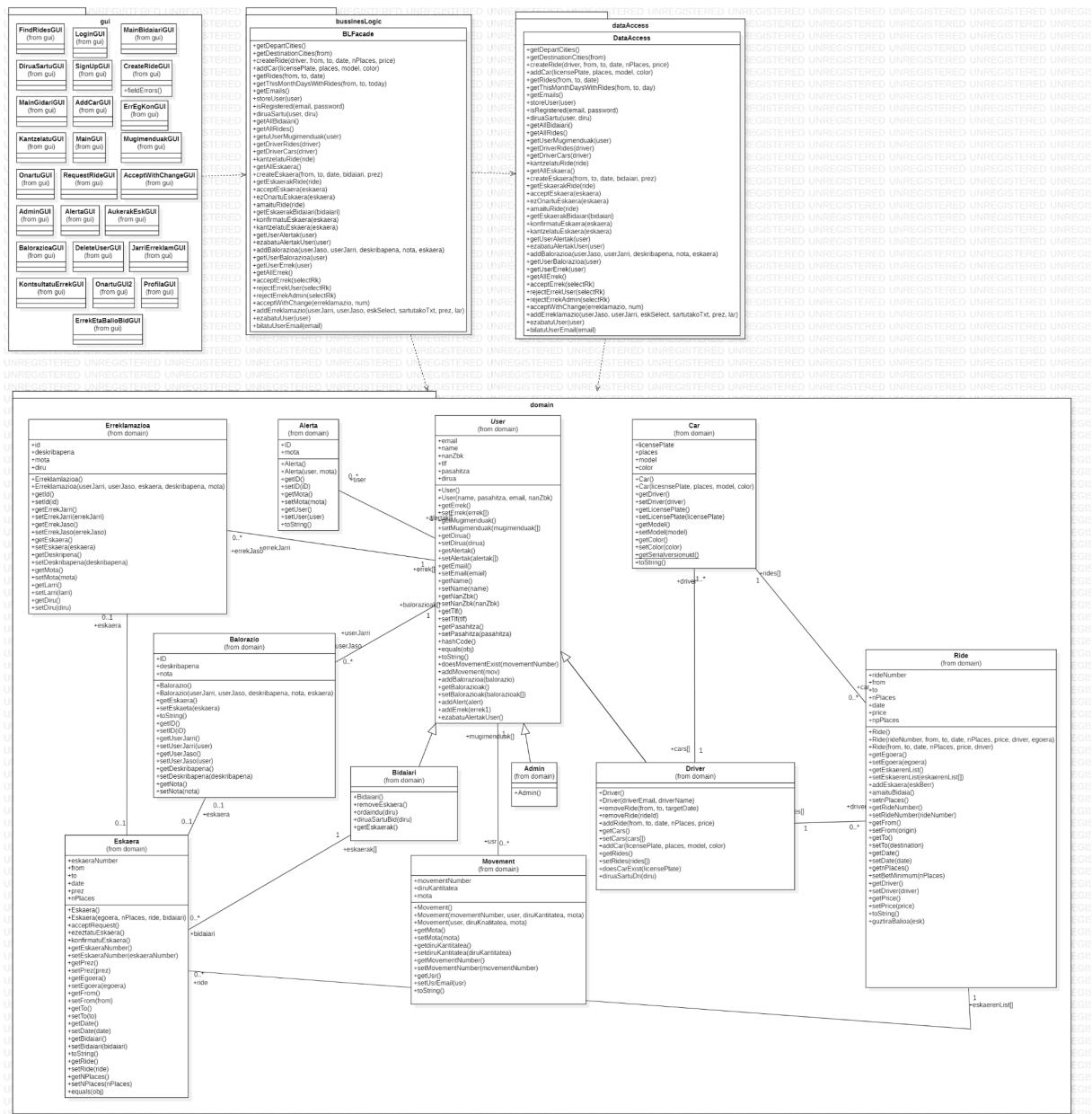
Behin erresoluzioa eginda dagoela, alertak sortuko dira.

## BalorazioakSD



Balorazio sistema erabiltzaileek bidaien balorazioak egiteko prozesua erakusten du. Balorazoa sortzen da eta lotutako eskaeraren egoera eguneratzen da. Ondoren, balorazioaren alerta sortzen da.

### 3.2. Klase diagramma



## 4. Implementazioa

Hauek dira Negozio Logikako interfaze klasean implemantatu ditugun metodoak:

### Hiri eta bidaien informazioa

1. `getDepartCities()`. Abiapuntu gisa erabilitako hiriak Itzultzen ditu.
2. `getDestinationCities(String from)`. Hiri batetik abiatuta, nora heltzen den itzultzen du.
3. `getRides(String from, String to, Date date)`. Emandako ibilbide eta dataren arabera dauden bidaiaitxak itzultzen ditu.
4. `getThisMonthDatesWithRides(String from, String to, Date date)`. Emandako ibilbidean bidaia duten egunak ematen ditu hilabete horretan.
5. `createRide(driver, from, to, date, nPlaces, price)`. Bidaia berri bat sortzen du.
6. `getAllRides()`. Bidaia guztien zerrenda itzultzen du.
7. `amaituRide(Ride ride)`. Bidaia amaitu egin dela adierazten du.

### Kotxeak eta gidariak

8. `addCar(String licensePlate, int places, String model, String color, String driverEmail)`. Gidariari kotxa bat gehitzen dio.
9. `getDriverCars(Driver driver)`. Gidari batek dituen ibilgailuak bueltatzen ditu.
10. `getDriverRides(Driver driver)`. Gidari batek dituen bidaiaiak bueltatzen ditu.

### Erabiltzaileak eta kontuak

11. `storeUser(User user)`. Erabiltzailea datu-basean gordetzen du.
12. `isRegistered(String email, String password)`. Erabiltzaile bat datu-basean existitzen den edo ez egiaztatzen du.
13. `diruaSartu(User user, float diru)`. Dirua sartzen du erabiltzaile baten kontuan.
14. `ezabatuUser(User user)`. Erabiltzaile bat ezabatzen du sistematik.
15. `bilatuUserEmail(String email)`. Erabiltzailearen email-aren bidez, erabiltzailea datu-basean bilatzen du.
16. `getEmails()`. Datu-basean gordeta dauden erabiltzaile guztien email-en zerrenda itzultzen du.
17. `getAllBidaiai()`. Bidaiai guztien zerrenda itzultzen du.

### Eskerak eta mugimenduak

18. `createEskaera(User user, Ride ride, int nPlaces)`. Eskaera berri bat sortzen du bidaia baterako.

19. acceptEskaera(Eskaera eskaera). Eskaera onartzen du.
20. ezOnartuEskaera(Eskaera eskaera). Eskaera deuseztatzen du.
21. konfirmatuEskaera(Eskaera eskaera). Eskaera konfirmatzen du.
22. kantzelatuEskaera(Eskaera eskaera). Eskaera bertan behera uzten du.
23. getEskaerakRide(Ride ride). Bidaia baterako eskaerak lortzen ditu.
24. getEskaerakBidaiai(Bidaiai bidaiai). Bidaiai baten eskaerak itzultzen ditu.
25. getAllEskaera(). Egindako eskaera guztien zerrenda itzultzen du.

### **Mugimendu ekonomikoak**

26. getUserMugimenduak(User user). Erabiltzaile baten mugimendu ekonomikoak itzultzen ditu.

### **Alertak, balorazioak eta erreklamazioak**

27. getUserAlertak(User user). Erabiltzaile baten alertak itzultzen ditu.
28. ezabatuAlertakUser(User user). Erabiltzaile baten alertak ezabatzen ditu.
29. addBalorazioa(Balorazio balorazio). Balorazio berri bat gehitzen du.
30. getUserBalorazioa(User user). Erabiltzaile baten balorazioak itzultzen ditu.
31. adderreklamazio(User userJarri, User userJaso, Eskaera eskaera, String testua, float prezioa, ErrekLarri larritasuna). Erreklamazio berri bat sortzen du.
32. getUserErrek(User user). Erabiltzailearen erreklamazioak itzultzen ditu.
33. acceptErrek(Erreklamazioa erreklamazioa). Erreklamazioa onartzen du.
34. rejectErrekUser(Erreklamazioa erreklamazioa). Erabiltzaileak erreklamazioa ez onartzen du.
35. rejectErrekAdmin(Erreklamazioa erreklamazioa). Administratzialeak erreklamazioa ez onartzen du.
36. AcceptWithChange(Erreklamazioa erreklamazioa, int kopurua). Erreklamazioa onartzen du zenbateko aldaketarekin.
37. getAllErrek(). Jarri diren erreklamazio guztien zerrenda itzultzen du.

## **5. Ondorioak**

### **Aurkitutako arazoak**

Proiektuan zehar hainbat erronka izan ditugu, besteak beste kodearen elkarbanaketa, denboraren kudeaketa eta lanaren antolaketa. Hala ere, arazo nagusiena datu-basearen diseinuari eta kudeaketari lotuta egon da. Hasieran ez genuen oso argi nola egituratu behar genituen datuak, eta horrek hainbat aldaketa, frogaketa eta egokitzapen ekarri ditu aurrerapen-prozesuan zehar.

Bestalde, proiektuaren hasierako jarraibideak ez genituen guztiz ulertu. Ondorioz, norabide aldaketak egin behar izan ditugu eta gure aplikazioa ikuspegi berri batetik berrantolatu. Aurreko iterazioan funtzionalitateak modu batean integratuta bazeuden ere, oraingo bertsioan funtzionalitate berak mantendu ditugu baina logika berri batean oinarrituta garatu ditugu, erabilera hobetzeko helburuarekin.

Adibidez, lehen sistema batean bidaiariek bidaiaiak eskatzen zitzuten eta gidariek onartu behar zitzuten eskaera horiek. Gaur egungo bertsioan, berriz, gidariek sortzen dituzte bidaiaiak eta bidaiariek bidaia horien artean bat aukeratu eta parte hartzeko eskaera egiten dute. Gidariak, ondoren, eskaera onartu edo baztertu dezake. Aldaketa honek aplikazioa errazago erabiltzeko moduan egituratu du eta erabiltzaileentzako logika intuitiboagoa eskaintzen du.

Aurkitu dugun beste arazo bat izan da web zerbitzua implementazioa, aurkezpenaren egunerako lortu genuen ondo joatea, baina hainbat hobekuntza egin eta gero web zerbitzuak arazoak ematen hasi zen eta bukaeran ezin izan dugu implementatzea.

### **Taldearen dinamika**

Taldeko lana, oro har, positiboa eta elkarlanean oinarritua izan da. Nahiz eta iterazio bakoitzean Scrum Master bat izendatu dugun, lanaren zatirik handiena elkarrekin egin dugu, eta taldekide guztiak parte hartu dute aktiboki. Scrum Master-ak ardura handiagoa izan du bideoen grabaketa eta edizioa bere gain hartzen.

### **Lortutako emaitza**

Emaitzarekin oso pozik gaude. Izen ere, hasieran galdua geunden eta ez genekien nola hasi, baina pixkanaka argitzen joan gara eta azken aplikazioarekin oso gustura gaude. Hasierako zalantzak gainditzea lortu dugu, eta funtzionalitate erabilgarri eta errealistak integratu ditugu. Gure ustez, erabilgarritasunari eta egiturari dagokionez aurrerapauso handia eman dugu.

### **Irakasgaiaren balorazioa**

Irakasgaiaren balorazioa, oro har, positiboa da. Taldean lan egiteko esperientzia aberasgarria izan da. Praktikan oinarritutako metodologia honek eta iterazioka lan egiteak gure gaitasun teknikoak eta taldeko koordinazioa garatzen lagundi digu.

Hala ere, irakasgaiaren hasierako fasean hainbat zalantza izan genituen. Aurretik aipatu bezala, proiektuaren jarraibideak ez genituen guztiz ulertu eta horrek norabidea aldatzera eraman gintuen. Hasierako informazioa ez zen nahikoa zehatza izan, ondorioz hainbat

erabaki gure intuizioaren eta irudimenaren arabera hartu genituen. Horregatik, uste dugu azalpenetan sakontasun handiagoa behar dela. Informazio zehatzagoak eta egituratuagoak izateak hasieratik lanaren norabidea argitu eta taldearen ahalegina eraginkorrago bihurtuko luke.

### **Etorkizunerako gomendioak**

Etorkizunera begira, esperientziatik ikasitako hainbat gomendio proposatzen ditugu etorkizuneko ikasleentzat:

- Proiettua ahalik eta lehenago hastea. Garrantzitsua da lana ez atzeratzea eta lehen momentutik lanean jartzea. Horrela, denbora gehiago izango duzue arazoak detektatzeko eta konpontzeko.
- Hasieratik antolaketa eta diseinu sendoa egitea. Garrantzitsua da proiektuaren oinarrizko alderdi guztiak lehen momentutik zehaztea. Behin diseinua ondo definituta dagoenean, kodea idaztea askoz errazagoa delako eta aldaketa handiak saihestu daitezke prozesuaren erdian.

## **6. Bideoaren URL-a**

Bideoaren link-a: <https://youtu.be/r5jJfwXZT38>

## **7. Kodearen URL-a**

GitHub-eko esteka: <https://github.com/ltxasoUrgoitia/Rides24/tree/oihane>

**\*GARRANTZITSUA!! “oihane” DEITZEN DEN BRANCH-A DA AZKEN ALDAKETAK DITUENA\***