Cairo University
Faculty of Computers and Artificial intelligence
Department of Computer Sciences

DivideRide

Supervised by

Dr. Hanaa Bayomi

TA. Ahmed Hassan

Implemented by

20196012	*AyaTullah Elsayed Abdelfattah Shabana*
20196016	*George Magdy Waheeb Attia*
20196093	*Habiba Ahmed Mohmed El-Amir*

Graduation Project
Academic Year 2022-2023
Final Documentation

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ABBREVIATION	MEANING
API	APPLICATION PROGRAMMING INTERFACE
GUI	GRAPHICAL USER INTERFACE
ERD	ENTITY RELATIONSHIP DIAGRAM
PK	PRIMARY KEY
FK	FOREIGN KEY

Table 1: List of Abbreviations

Chapter 1: Introduction

1.1 Motivation:

Due to the prevailing issues of inflation, pollution, and traffic congestion plaguing our world, many individuals face challenges in owning a car or affording the cost of gasoline. In light of these circumstances, our motivation stems from a desire to address these problems through the implementation of a carpooling Android application.

Our goal is to facilitate car sharing among individuals belonging to the same company. We firmly believe that by encouraging carpooling, we can make a positive impact on the challenges posed by inflation, pollution, and traffic congestion. Carpooling not only helps alleviate the financial burden of car ownership and gasoline expenses for individuals but also contributes to reducing traffic congestion and minimizing carbon emissions, promoting a more sustainable and eco-friendly transportation system.

Our application fosters a sense of community among employees of the same company. By connecting colleagues who share the same workplace, we not only provide a convenient transportation solution but also strengthen relationships and create a more connected company community. Through carpooling, employees can reduce traffic congestion, minimize pollution, and enjoy the social aspects of shared rides. Together, we can improve the commuting experience and build a better workplace community.

To ensure the safety and reliability of our platform, we prioritize implementing robust measures. User verification, and ratings, ensuring that users can trust and rely on the carpooling experiences provided. We aim to create a secure and trustworthy environment for all users, enabling them to have a positive and satisfactory carpooling experience.

Our carpooling Android application focuses exclusively on users from the same company, ensuring safety, reliability, and a sense of community. By connecting colleagues, we create a secure environment for trusted carpooling experiences. We prioritize user verification and ratings to further enhance the safety measures and build a community of trusted individuals within the company. We foster a strong sense of colleague bonding and mutual trust among participants, ensuring a positive and secure carpooling experience.

1.2 Problem definition

The problem we aim to address is the challenges faced by individuals in owning a car or affording the cost of gasoline due to inflation, pollution, and traffic congestion. Specifically, we focus on employees within the same company who encounter these difficulties. Our goal is to provide a solution through a carpooling android application that enables users to share rides, whether they own a car or not, and share the expenses of gasoline. By connecting users from the same organization, we promote cost savings, reduce traffic congestion, minimize carbon emissions, and foster a sense of community among colleagues.

1.3 Project Objective

The objective of our project is to develop a carpooling Android application that facilitates convenient and cost-effective ride-sharing among employees of the same company. The application will allow car owners to offer rides and passengers, on the other hand, can search for rides. The application will match passengers with car owners, providing them with the option to choose their preferred carpooling partner. By implementing this solution, we aim to promote efficient transportation, reduce individual transportation costs, alleviate traffic congestion, and contribute to a more sustainable and environmentally-friendly commuting experience for employees within the same company.

1.4 Gantt chart of Project Time Plan

Task	Task Title	Start Date	End Date	Duration	State
1	Idea Brainstorming	2/7/2022	1/8/2022	30	Completed
2	Analysis	2/8/2022	20/8/2022	18	Completed
3	Learning Flutter	15/10/2022	10/2/2022	119	Completed
4	Requirements	11/11/2022	18/11/2022	7	Completed
5	Design and Architecture	1/2/2022	9/2/2023	8	Completed
6	UI Prototyping Screen	12/2/2023	16/2/2023	4	Completed
7	Mid-year Documentation	1/2/2022	17/2/2023	16	Completed
8	Mid-year Presentation	7/2/2023	17/2/2023	10	Completed
9	Application Implementation	25/2/2023	7/7/2023	133	Completed
10	Unit Testing	16/6/2023	6/7/2023	20	Completed
11	Integration Testing	26/6/2023	6/7/2023	10	Completed
12	System Testing	2/7/2023	7/7/2023	5	Completed
13	Acceptance Testing	8/7/2023	8/7/2023	1	Completed
14	Final Presentation	21/6/2023	8/7/2023	17	Completed
15	Final Documentation	21/6/2023	8/7/2023	17	Completed

Table 2: Time plan

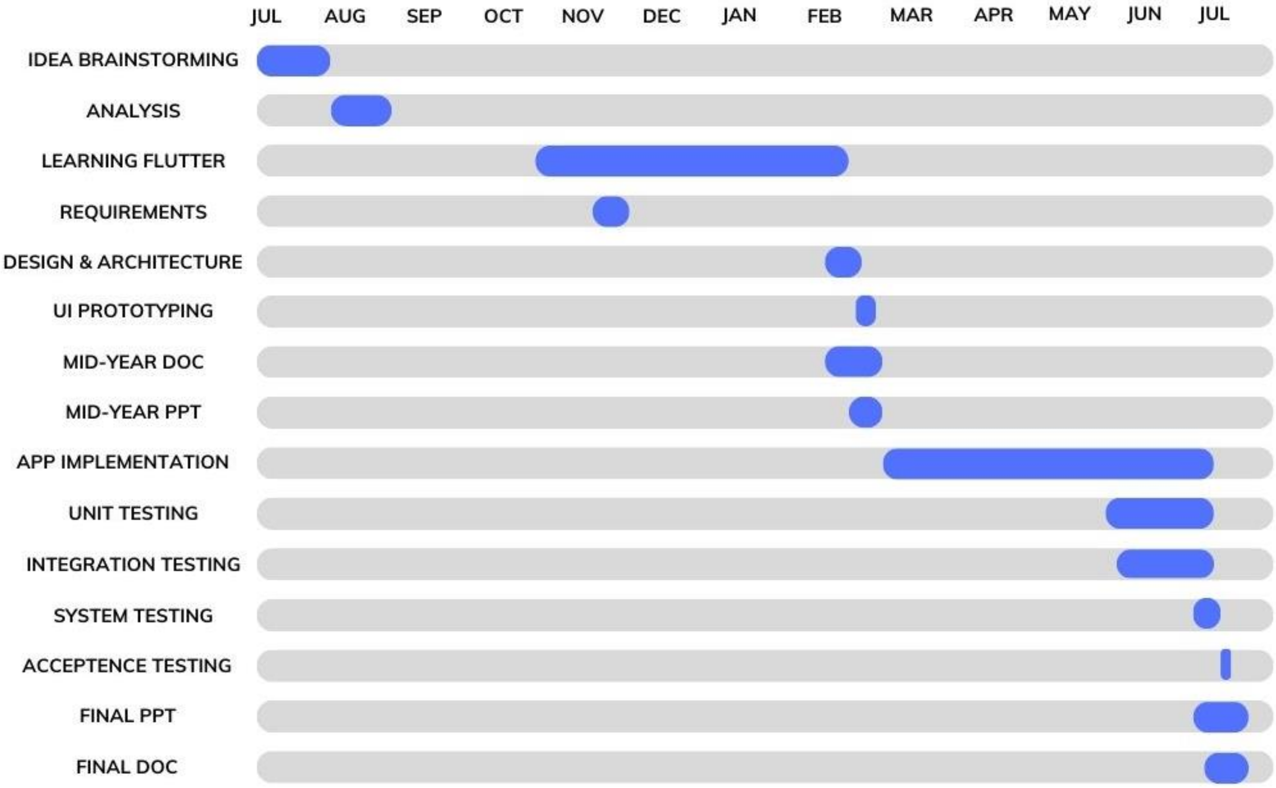


Figure 1: Gantt chart

1.5 ***Project Development Methodology***

We adopted an agile methodology, for the development of this project. Agile methodology emphasize flexibility, collaboration, and iterative improvement. This approach allowed us to adapt to changing requirements, incorporate feedback from users, and deliver incremental updates throughout the development process. By dividing the work into manageable sprints, we ensured efficient progress, effective communication, and the ability to quickly respond to changing needs. The Agile methodology enabled us to deliver a high-quality carpooling application that meets the expectations and needs of our users.

1.6 ***Used Tools in the project***

The carpooling Android application project utilizes Flutter for cross-platform UI development, Dart for efficient coding, Google Firebase for backend services, GitHub for version control, Android Studio for development, and testing on laptops, Android devices, and emulators.

1.7 *Report Organization*

Chapter 2: W will talk about Related, you will explore the closest example to our project and highlight the main differences between them and DivideRide.

Chapter 3: This chapter focuses on analyzing the carpooling Android application system. It begins with a project specification section (3.1), which includes the description of the project's functional requirements (3.1.1) and non-functional requirements (3.1.2). Use case diagrams (3.2) will be used to visually represent the interactions between system actors and the application.

Chapter 4: In this chapter, the system design of DivideRide will be discussed. It includes the following sections: system component diagram, system class diagrams, sequence diagrams, project entity-relationship diagram (ERD), and system GUI design. These design elements provide a clear understanding of the system architecture and its visual representation.

Chapter 5: This chapter focuses on the implementation and testing phase of the project. It covers the system running, providing details on how the application is executed and its functionality. Additionally, samples of the applied test cases, specifically system test cases, will be presented to demonstrate the effectiveness and reliability of the application.

References: This section lists the references used throughout the report.

Chapter 2: Related Work

In this chapter, we will explore existing application that are closely related to our DivideRide application. Among the notable examples is the "Foorera" application, which shares similarities with our application for being a carpooling application. By examining Foorera and highlighting the main differences between our project and theirs, we gain valuable insights into the competitive landscape and identify our unique value proposition.

Points of Comparisons	**DivideRide**	**Foorera**
Users	Customized for companies and their employees.	Generalized for all users.
Security	It is more Secured, as It provides tailored access control, data protection measures, & customization options, complying with industry standards.	It is less secured, since it is a generic application designed for everyone.
Payment Timing	User can ride first then pay after the ride is ended.	User pay first when he requests to join ride.
Payment Way	There is the option to pay by cash.	There is no option to pay by cash.
Application Service Fees	Application take less fees for service.	Application take more fees for service.

Table 3: DivideRide VS Foorera

Chapter 3: System Analysis

3.1 Project Specifications

3.1.1 Functional Requirements

Drivers Functionalities:

Sign up:

- Driver should authenticate his/her phone number to sign up.
- Driver should provide his username, mobile number, service location he wants to register, vehicle details (model, year, color) and driving license should be provided. Then he will be waiting for the approval.
- Verification status should be pending for the Driver until his registration has been approved.
- Driver should be able to show himself on google maps as an available driver when the verification is approved.

Log in:

- Driver should authenticate his/her phone number to Log in.
- If driver signed up before and has a profile in the system, he will be logged in automatically

Log out:

- Driver can log out easily.

Create ride:

- Each ride should have driver name, start time, pickup location, drop location, price per km per seat.
- Driver should be able to choose the repeated ride days.

View ride:

- Driver should be able see his rides info, and can edit or cancel the ride.

View ride requests:

- Driver should be able to see requests sent from users to join a specific ride, he/she can accept or reject each request.

View User's profile:

- Driver should be able to see user's profile.

Call User:

- Driver can call user when he starts the ride.

Cancel Ride:

- Driver should be able to cancel the ride and the users who joined it should be notified.

Notify User:

- When driver cancels or starts the ride the user that joined this ride should be informed.

My Rides:

- Driver should be able to see Ongoing, Upcoming and History Rides and see each ride details (start time, pickup location, drop location, car info (type, color) , price for a seat , total money earning , users profiles), and ride state (on going, upcoming, canceled, or ended).

Start Ride:

- When ride start time comes the driver can start the ride.
- He can check who was picked up and who was not.

End Ride:

- After reaching the destination the driver should end the ride.

<u>Users Functionalities:</u>

Sign up:

- User should be able to register to the app by entering his mobile number.
- App expects valid mobile number because we will be sending an OTP (one-time password) to this provided number.
- After entering a valid mobile number and receiving the code, user should be able to enter the code that has been sent.
- User should provide his username, home address, business address, shopping center, profile photo.

Log in:

- User should authenticate his/her phone number to Log in.
- If user signed up before and has a profile in the system, he will be logged in automatically

Log out:

- User can log out easily.

Search For ride:

- User should be able to search for ride with a destination place and specific date, then the available rides near to his pickup location are arranged from nearest to furthest.
- User should be able to see for each ride driver name, driver's profile, start time, pickup location, drop location, car info (type, color), and price for a seat.

View ride:

- User should be able see all Ride info, and can send a request to join the ride.

Send a ride request (Join Ride):

- User should be able to send request for ride's driver from the available rides he searched for.

View Driver's profile:

- User should be able to see driver's profile.

Call User:

- Driver can call user when he starts the ride.

Cancel Ride:

- User should be able to cancel a ride he/she joined.

Pay ride:

- User can pay using cash or Vodafone cash upon arriving to destination.

My Rides:

- User should be able to see Ongoing, Upcoming and History Rides and see each ride details: start time, pickup location, drop location, car info (type, color), ride state (on going, upcoming, canceled, or ended).

3.1.2 Non-functional Requirements

Usability:

- All application clients can navigate through the system, each action in system takes an average of 3 strokes.

Recoverability:

- The system should be able to recover from a crash or a failure in the system and returning to full operations as it is hosted on the cloud.

Availability:

- The application and the host features in Google Firebase should be available 24/7.

Performance:

- The application's load time should not more than 7 seconds.

Scalability:

- The application is scalable as it is hosted on the cloud and it can be scaled the way it is needed (pay as you go).

Authentication:

- The application will not accept any user/driver to access the application without a valid phone number.

Portability:

- The software can be made available for both Android and iOS devices.

Compatibility:

- The application should be compatible with a wide range of different android devices with different screen sizes, ensuring consistent of functionality and user experience.

3.2 Use Case Diagram

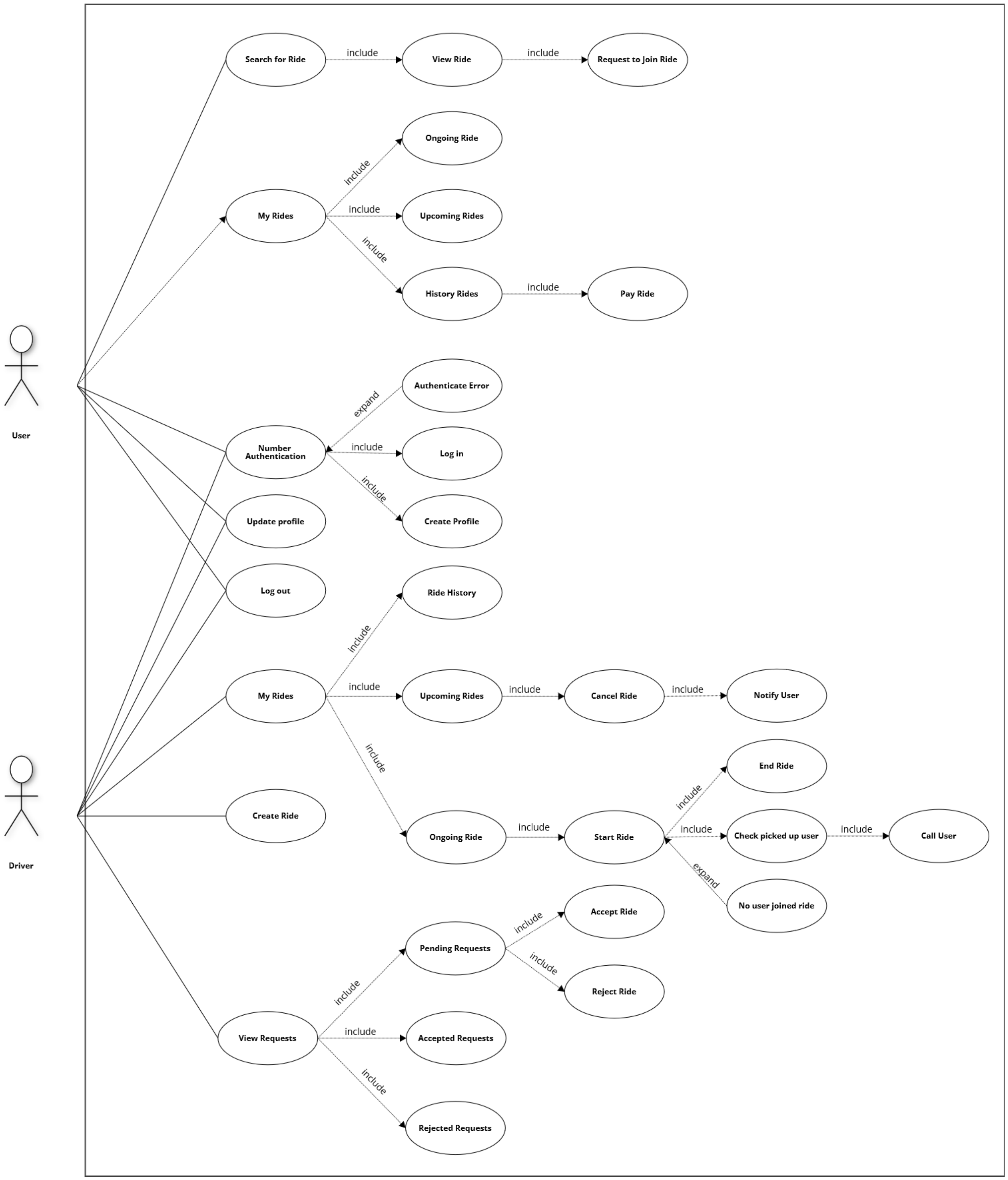


Figure 2: Use Case Diagram

Chapter 4: System Design

4.1 System Component Diagram

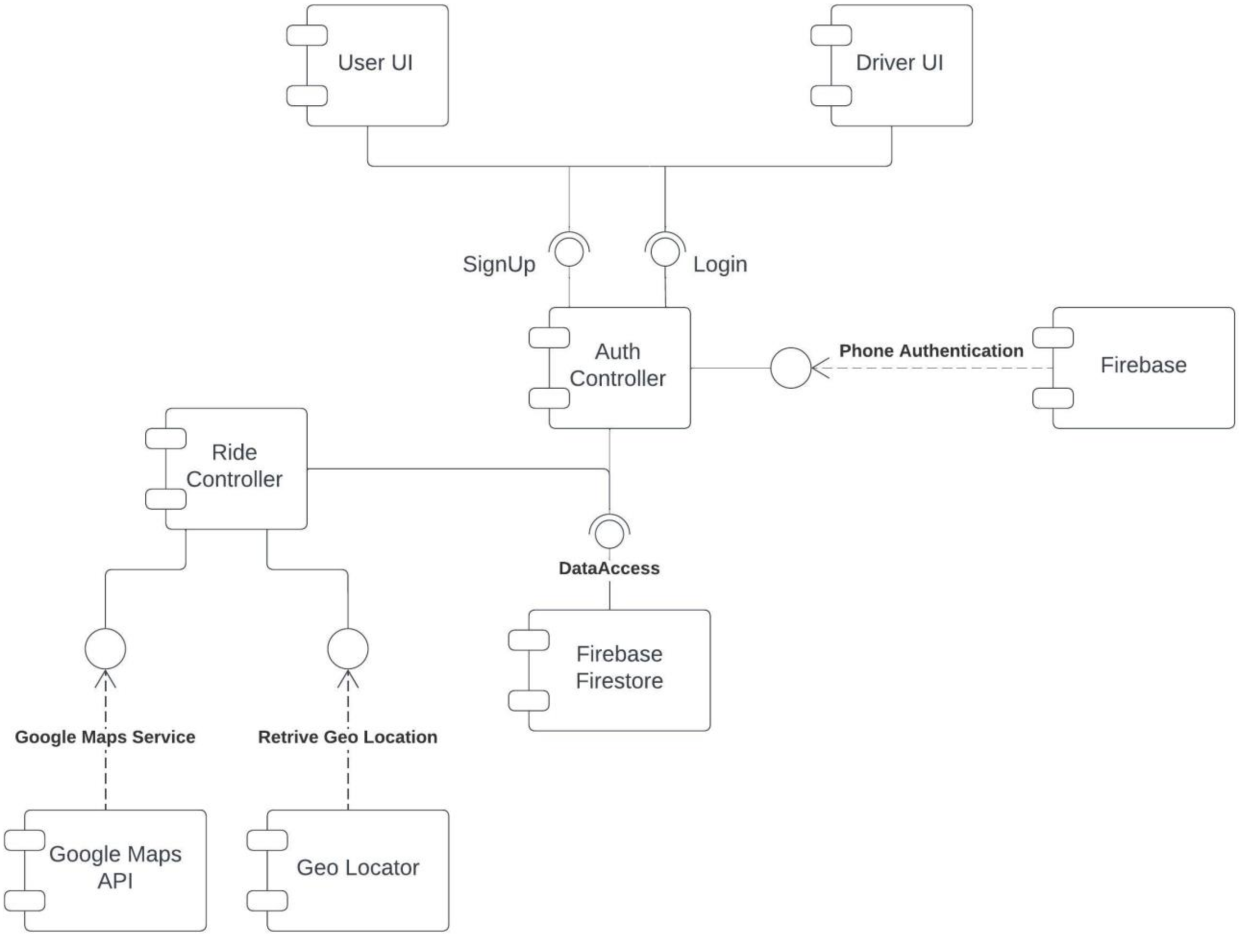


Figure 3: Component Diagram

4.2 System Class Diagram

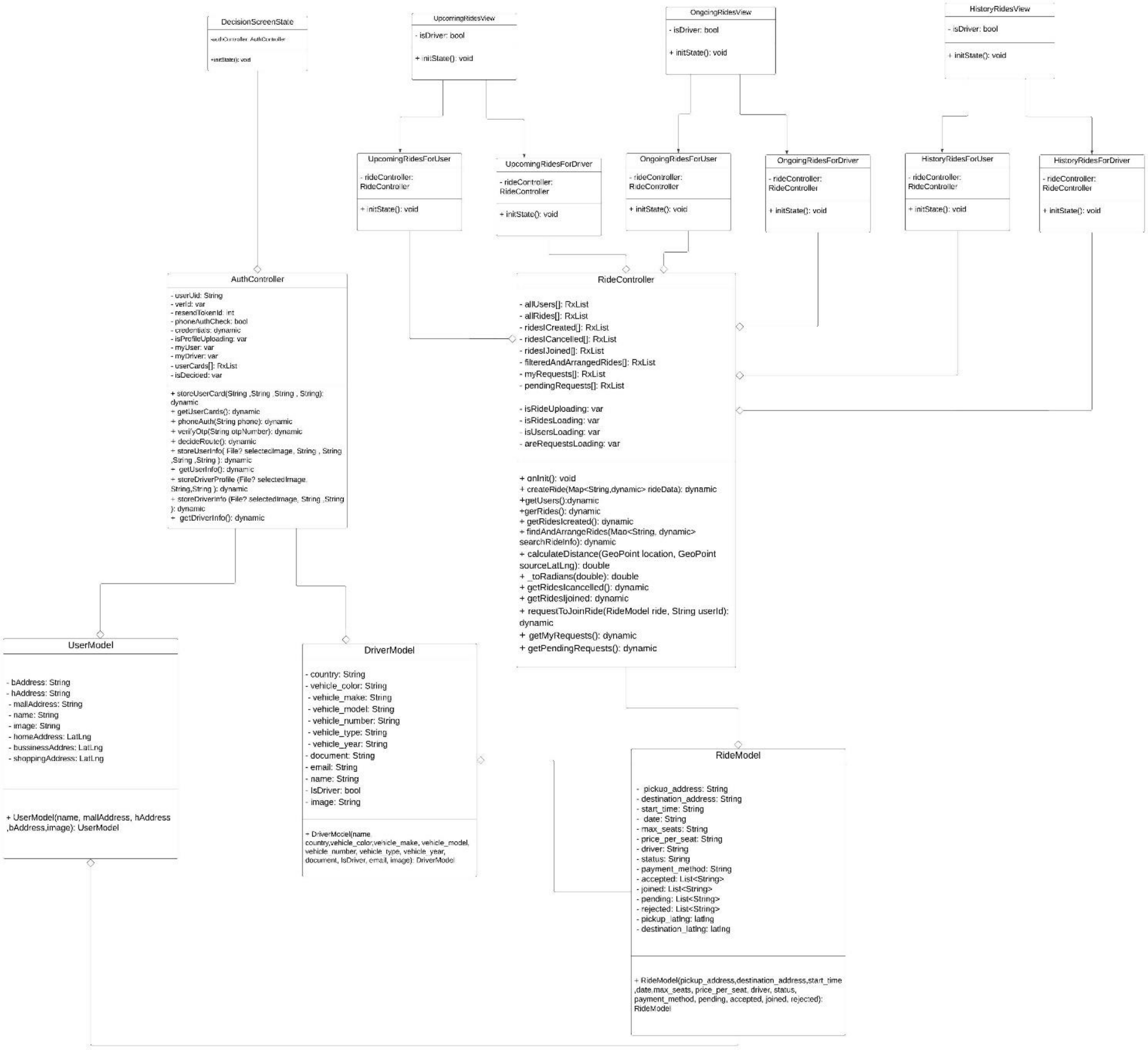

DecisionScreenState
-authController: AuthController
+initState(): void
UpcomingRidesView
- isDriver: bool
+ initState(): void
OngoingRidesView
- isDriver: bool
+ initState(): void
HistoryRidesView
- isDriver: bool
+ initState(): void
UpcomingRidesForUser
- rideController: RideController
+ initState(): void
UpcomingRidesForDriver
- rideController: RideController
+ initState(): void
OngoingRidesForUser
- rideController: RideController
+ initState(): void
OngoingRidesForDriver
- rideController: RideController
+ initState(): void
HistoryRidesForUser
- rideController: RideController
+ initState(): void
HistoryRidesForDriver
- rideController: RideController
+ initState(): void
AuthController
- userUid: String
- verId: var
- resendTokenId: int
- phoneAuthCheck: bool
- credentials: dynamic
- isProfileUploading: var
- myUser: var
- myDriver: var
- userCards[]: RxList
- isDecided: var
+ storeUserCard(String ,String ,String , String): dynamic
+ getUserCards(): dynamic
+ phoneAuth(String phone): dynamic
+ verifyOtp(String otpNumber): dynamic
+ decideRoute(): dynamic
+ storeUserInfo(File? selectedImage, String , String ,String ,String): dynamic
+ getUserInfo(): dynamic
+ storeDriverProfile (File? selectedImage, String,String): dynamic
+ storeDriverInfo (File? selectedImage, String ,String): dynamic
+ getDriverInfo(): dynamic
RideController
- allUsers[]: RxList
- allRides[]: RxList
- ridesICreated[]: RxList
- ridesICancelled[]: RxList
- ridesIJoined[]: RxList
- filteredAndArrangedRides[]: RxList
- myRequests[]: RxList
- pendingRequests[]: RxList
- isRideUploading: var
- isRidesLoading: var
- isUsersLoading: var
- areRequestsLoading: var
+ onInit(): void
+ createRide(Map<String,dynamic> rideData): dynamic
+getUsers():dynamic
+gerRides(): dynamic
+ getRidesIcreated(): dynamic
+ findAndArrangeRides(Mao<String, dynamic> searchRideInfo): dynamic
+ calculateDistance(GeoPoint location, GeoPoint sourceLatLng): double
+ _toRadians(double): double
+ getRidesIcancelled(): dynamic
+ getRidesIjoined: dynamic
+ requestToJoinRide(RideModel ride, String userId): dynamic
+ getMyRequests(): dynamic
+ getPendingRequests(): dynamic
UserModel
- bAddress: String
- hAddress: String
- mallAddress: String
- name: String
- image: String
- homeAddress: LatLng
- bussinessAddres: LatLng
- shoppingAddress: LatLng
+ UserModel(name, mallAddress, hAddress ,bAddress,image): UserModel
DriverModel
- country: String
- vehicle_color: String
- vehicle_make: String
- vehicle_model: String
- vehicle_number: String
- vehicle_type: String
- vehicle_year: String
- document: String
- email: String
- name: String
- IsDriver: bool
- image: String
+ DriverModel(name country,vehicle_color,vehicle_make, vehicle_model, vehicle_number, vehicle_type, vehicle_year, document, IsDriver, email, image): DriverModel
RideModel
- pickup_address: String
- destination_address: String
- start_time: String
- date: String
- max_seats: String
- price_per_seat: String
- driver: String
- status: String
- payment_method: String
- accepted: List<String>
- joined: List<String>
- pending: List<String>
- rejected: List<String>
- pickup_latlng: latlng
- destination_latlng: latlng
+ RideModel(pickup_address,destination_address,start_time ,date,max_seats, price_per_seat, driver, status, payment_method, pending, accepted, joined, rejected): RideModel

Figure 4: Class Diagram

4.3 Sequence Diagram

4.3.1 Authentication Sequence Diagram

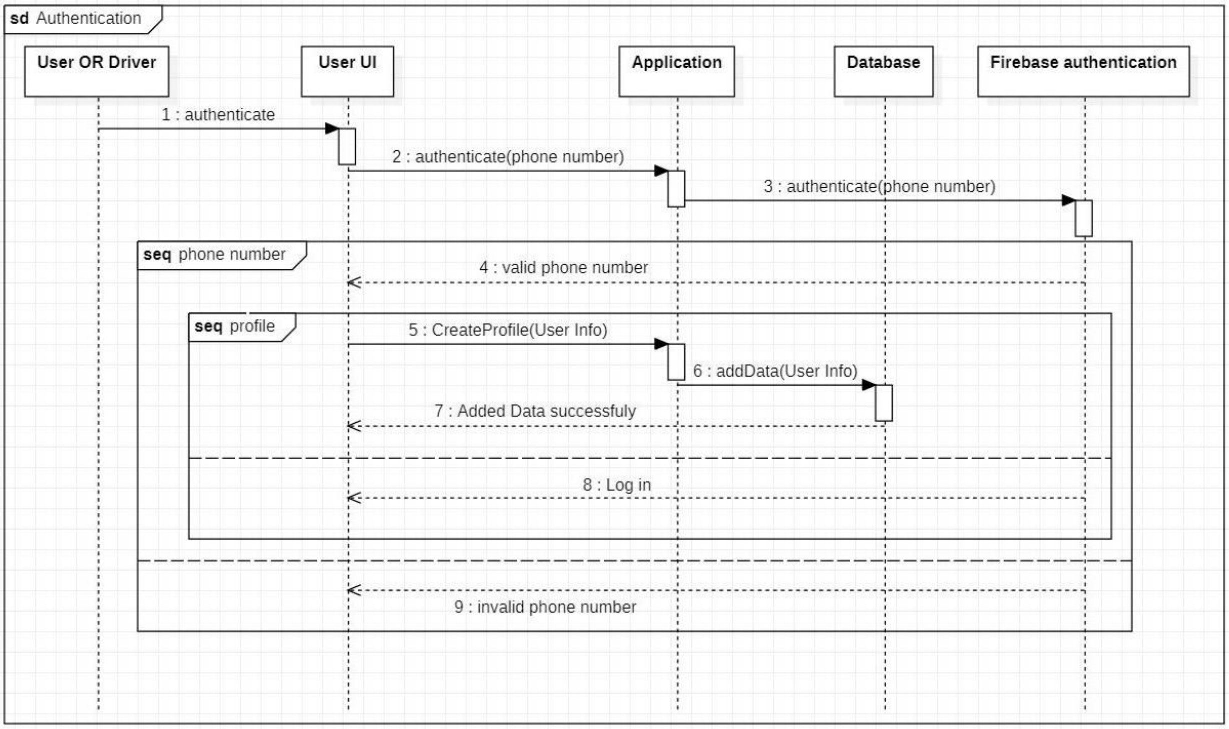


Figure 5: Authentication Sequence Diagram

4.3.2 Search for Rides Diagram

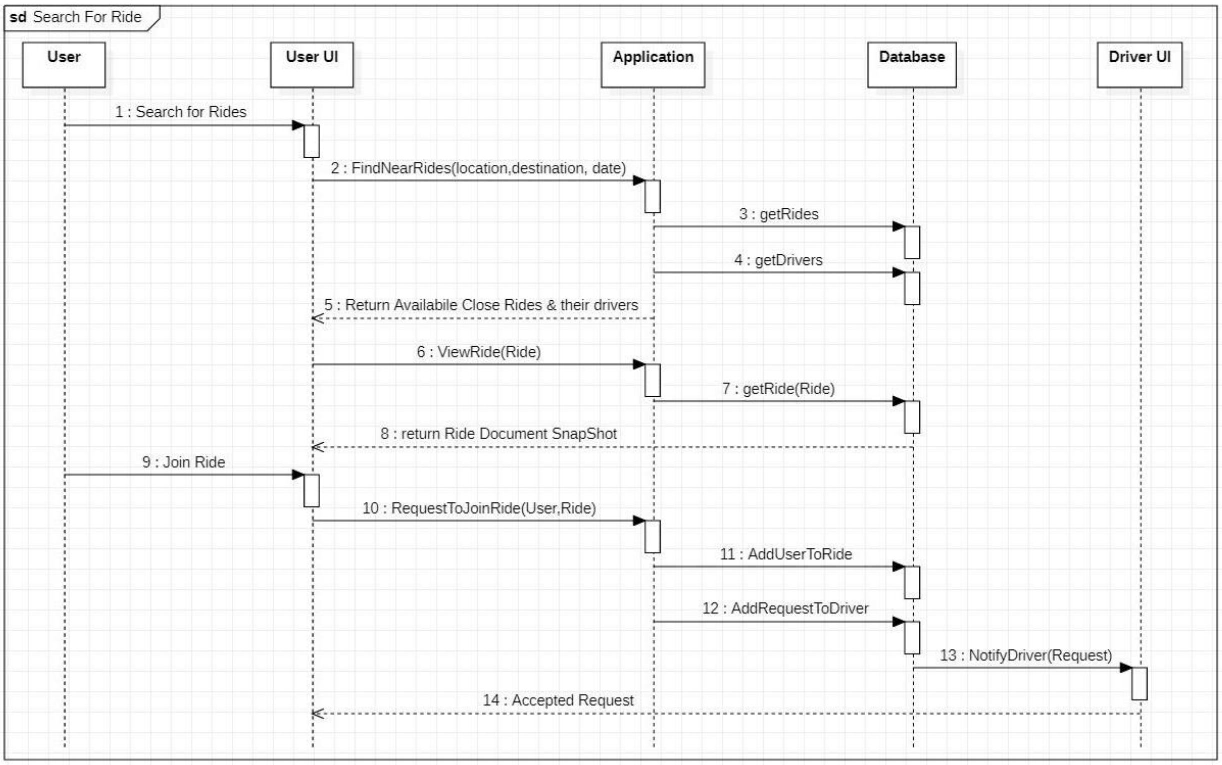


Figure 6: Search for Rides Sequence Diagram

4.3.3 Create Ride & View Requests Sequence Diagram

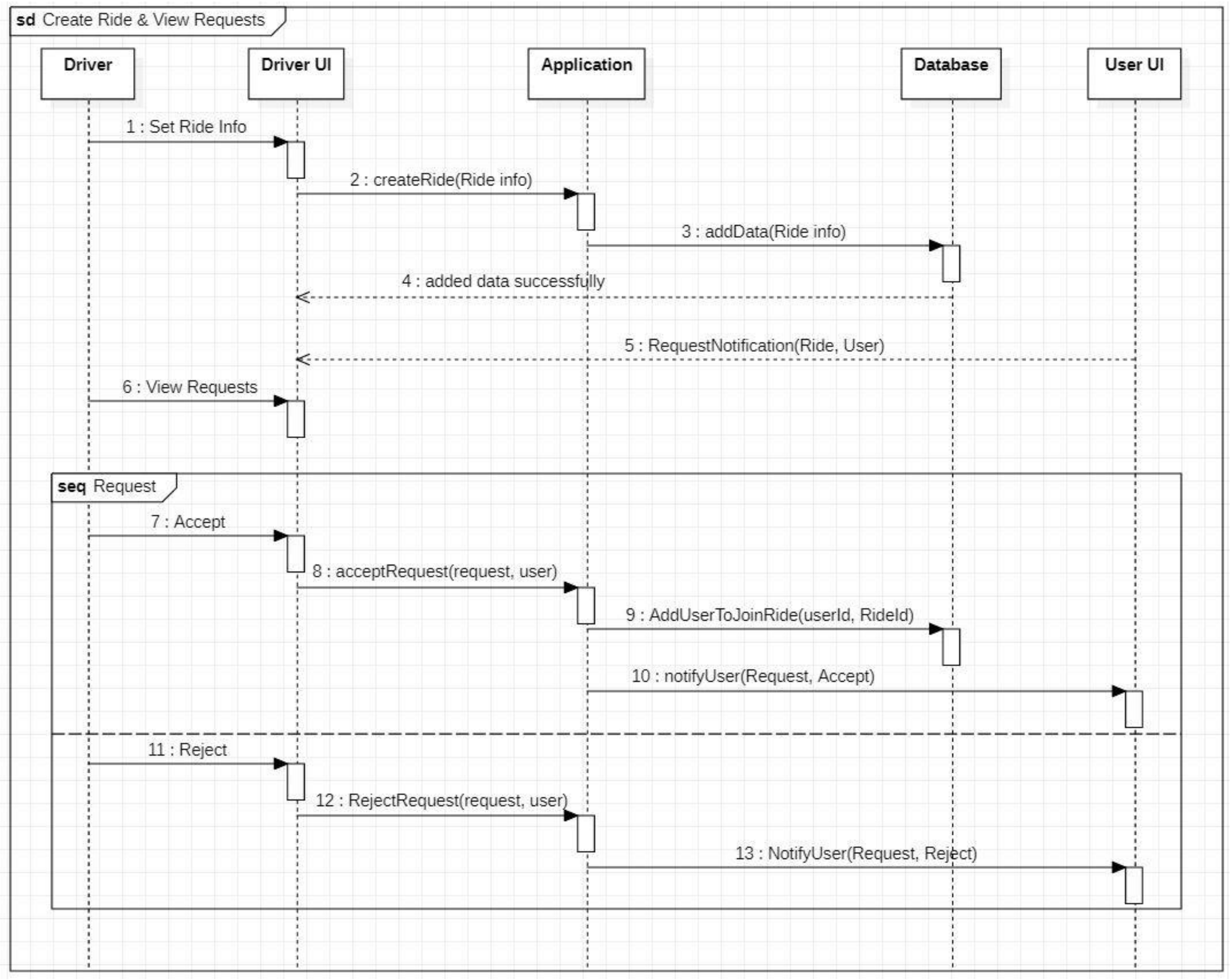

sd Create Ride & View Requests
Driver
Driver UI
Application
Database
User UI
1 : Set Ride Info
2 : createRide(Ride info)
3 : addData(Ride info)
4 : added data successfully
5 : RequestNotification(Ride, User)
6 : View Requests
seq Request
7 : Accept
8 : acceptRequest(request, user)
9 : AddUserToJoinRide(userId, RideId)
10 : notifyUser(Request, Accept)
11 : Reject
12 : RejectRequest(request, user)
13 : NotifyUser(Request, Reject)

Figure 7: Create Ride & View Requests Sequence Diagram

4.3.4 All Rides Sequence Diagram

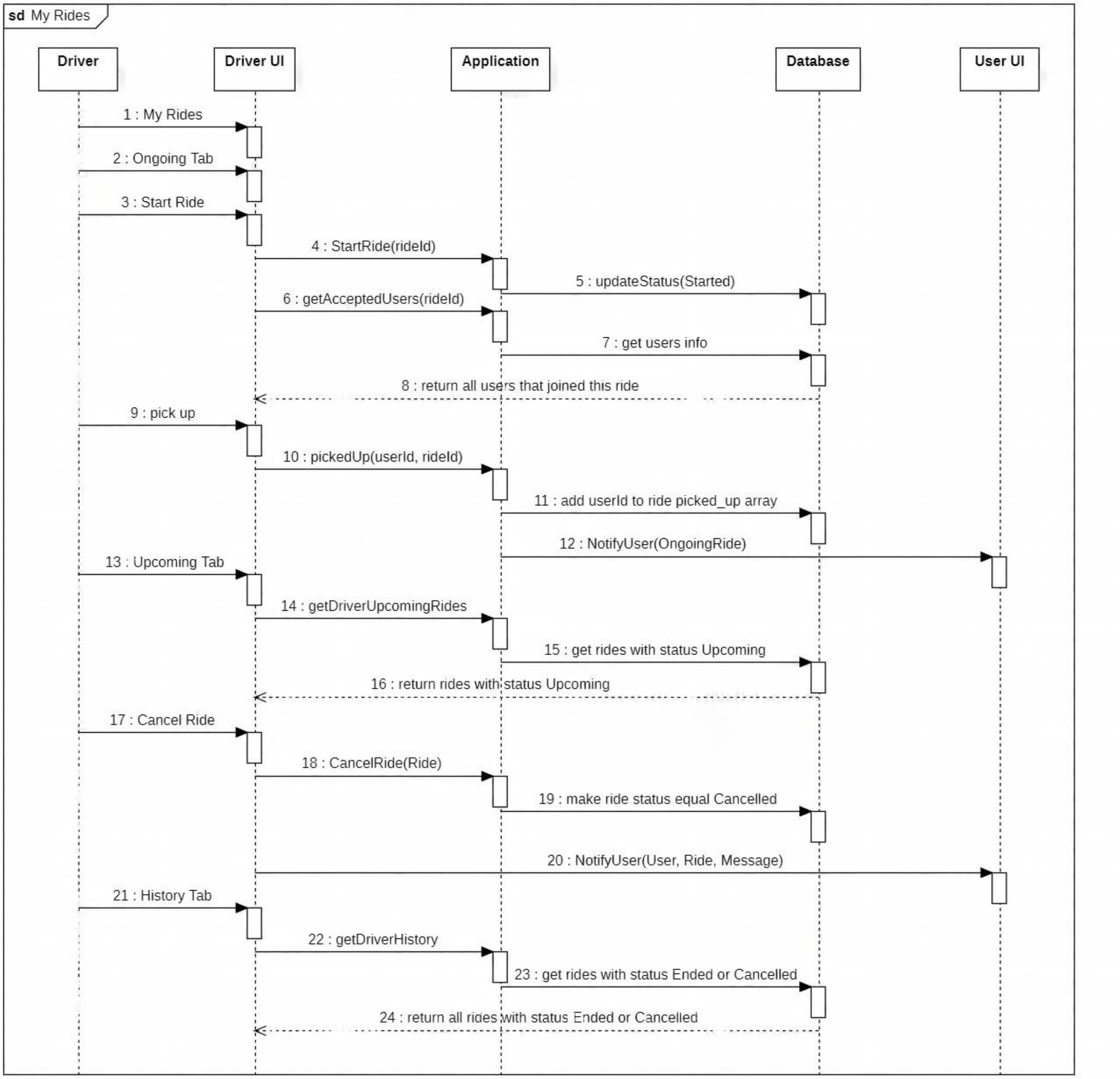
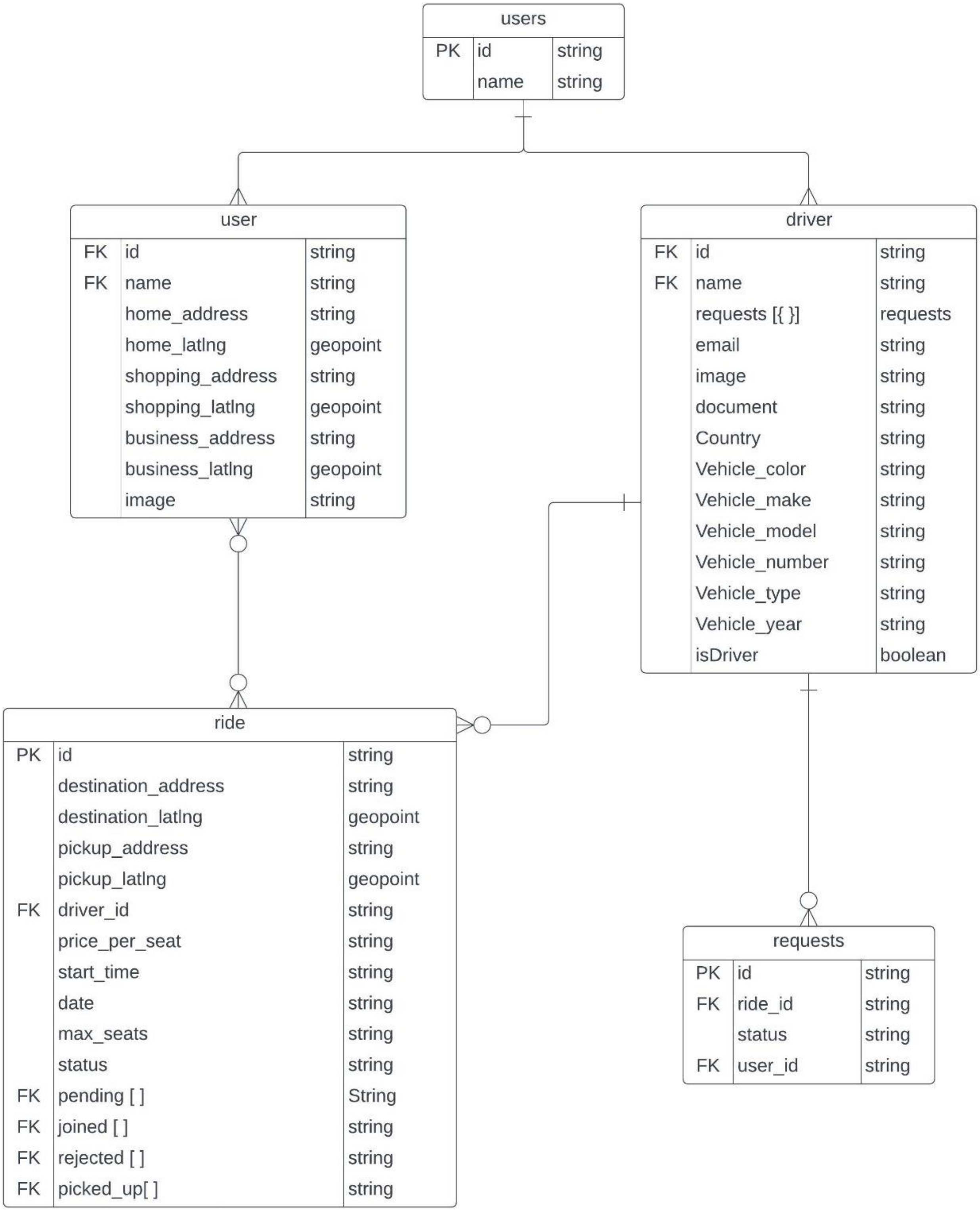

sd My Rides
Driver
Driver UI
Application
Database
User UI
1 : My Rides
2 : Ongoing Tab
3 : Start Ride
4 : StartRide(rideId)
5 : updateStatus(Started)
6 : getAcceptedUsers(rideId)
7 : get users info
8 : return all users that joined this ride
9 : pick up
10 : pickedUp(userId, rideId)
11 : add userId to ride picked_up array
12 : NotifyUser(OngoingRide)
13 : Upcoming Tab
14 : getDriverUpcomingRides
15 : get rides with status Upcoming
16 : return rides with status Upcoming
17 : Cancel Ride
18 : CancelRide(Ride)
19 : make ride status equal Cancelled
20 : NotifyUser(User, Ride, Message)
21 : History Tab
22 : getDriverHistory
23 : get rides with status Ended or Cancelled
24 : return all rides with status Ended or Cancelled

Figure 8: All Rides Sequence Diagram

4.4 Project ERD



	users	
PK	id	string
	name	string

	user	
FK	id	string
FK	name	string
	home_address	string
	home_latlng	geopoint
	shopping_address	string
	shopping_latlng	geopoint
	business_address	string
	business_latlng	geopoint
	image	string

	driver	
FK	id	string
FK	name	string
	requests [{ }]	requests
	email	string
	image	string
	document	string
	Country	string
	Vehicle_color	string
	Vehicle_make	string
	Vehicle_model	string
	Vehicle_number	string
	Vehicle_type	string
	Vehicle_year	string
	isDriver	boolean

	ride	
PK	id	string
	destination_address	string
	destination_latlng	geopoint
	pickup_address	string
	pickup_latlng	geopoint
FK	driver_id	string
	price_per_seat	string
	start_time	string
	date	string
	max_seats	string
	status	string
FK	pending [ ]	String
FK	joined [ ]	string
FK	rejected [ ]	string
FK	picked_up[ ]	string

	requests	
PK	id	string
FK	ride_id	string
	status	string
FK	user_id	string

Figure 9: Entity Relationship Diagram

4.5 __System GUI Design__

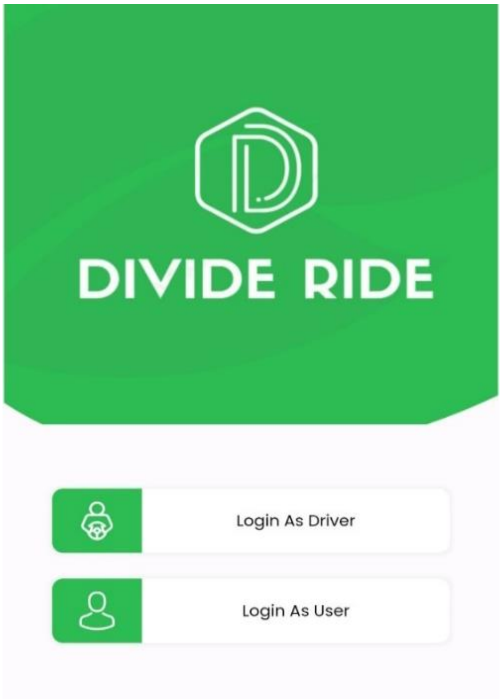
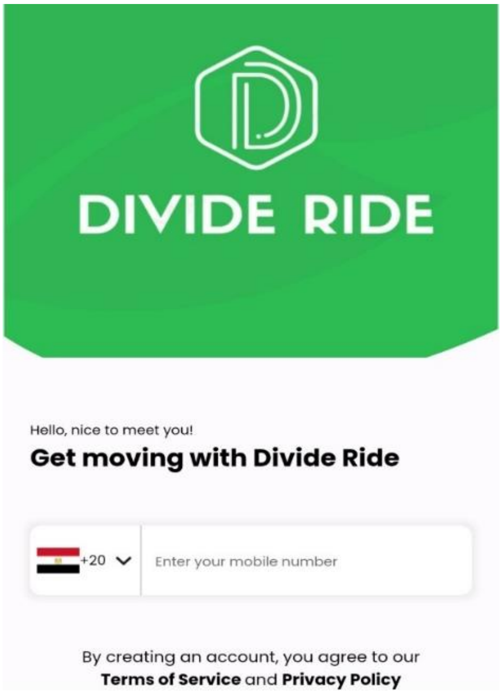


Figure 10: Decision page



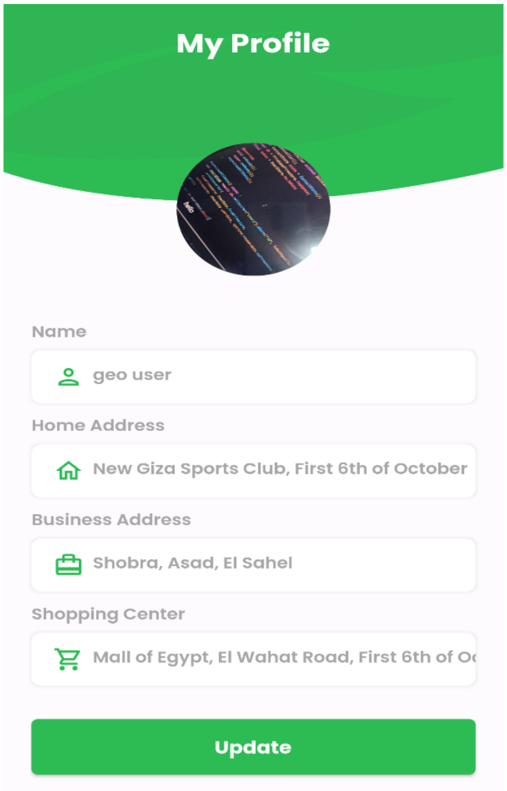
Hello, nice to meet you!

Get moving with Divide Ride

By creating an account, you agree to our **Terms of Service** and **Privacy Policy**

Figure 11: Authentication page

User UI:



My Profile

Name

geo user

Home Address

New Giza Sports Club, First 6th of October

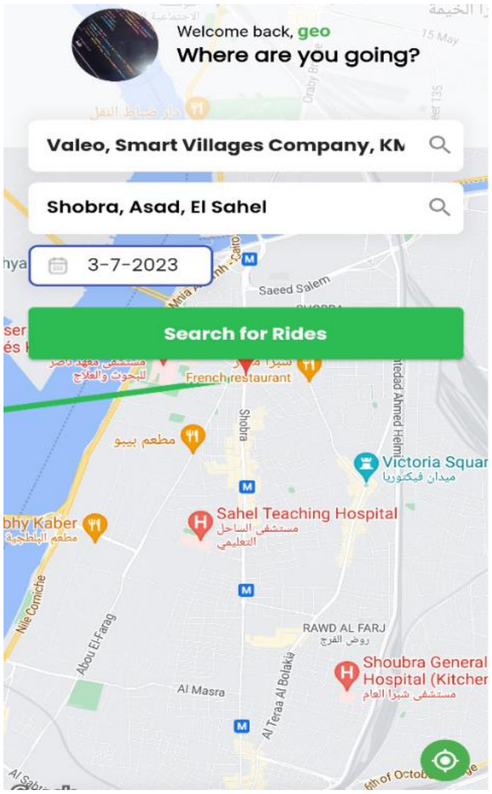
Business Address

Shobra, Asad, El Sahel

Shopping Center

Mall of Egypt, El Wahat Road, First 6th of Oc

Figure 12: User update profile page



Welcome back, geo

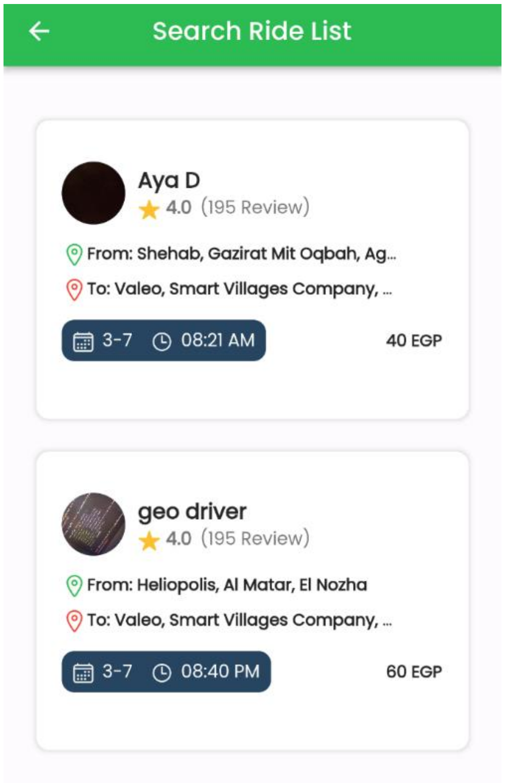
Where are you going?

Valeo, Smart Villages Company, KM

Shobra, Asad, El Sahel

3-7-2023

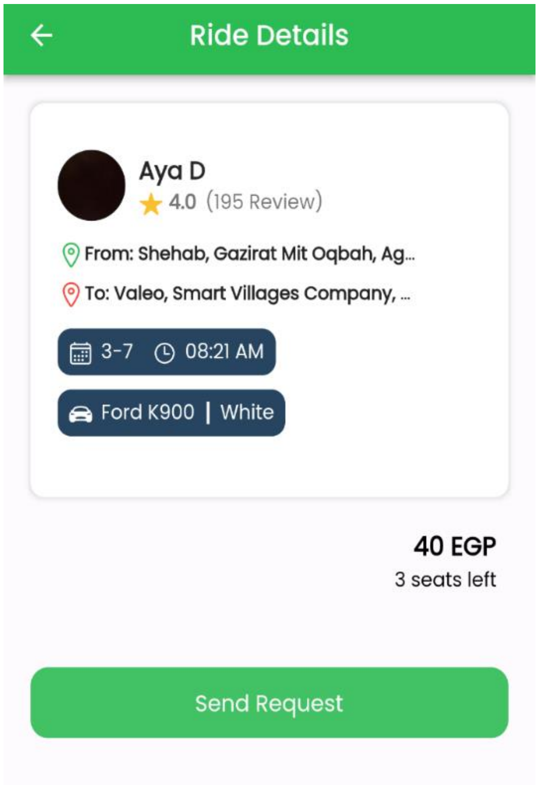
Figure 13: Search for



Aya D
4.0 (195 Review)
From: Shehab, Gazirat Mit Oqbah, Ag...
To: Valeo, Smart Villages Company, ...
3-7 08:21 AM
40 EGP

geo driver
4.0 (195 Review)
From: Heliopolis, Al Matar, El Nozha
To: Valeo, Smart Villages Company, ...
3-7 08:40 PM
60 EGP

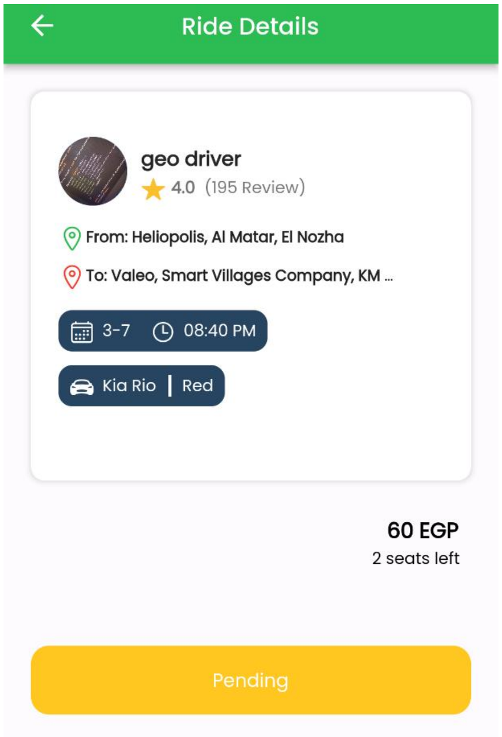
Figure 14: Rides found after searching



Aya D
4.0 (195 Review)
From: Shehab, Gazirat Mit Oqbah, Ag...
To: Valeo, Smart Villages Company, ...
3-7 08:21 AM
Ford K900 | White

40 EGP
3 seats left

Figure 15: View ride details & send request to join ride button



geo driver

4.0 (195 Review)

From: Heliopolis, Al Matar, El Nozha

To: Valeo, Smart Villages Company, KM ...

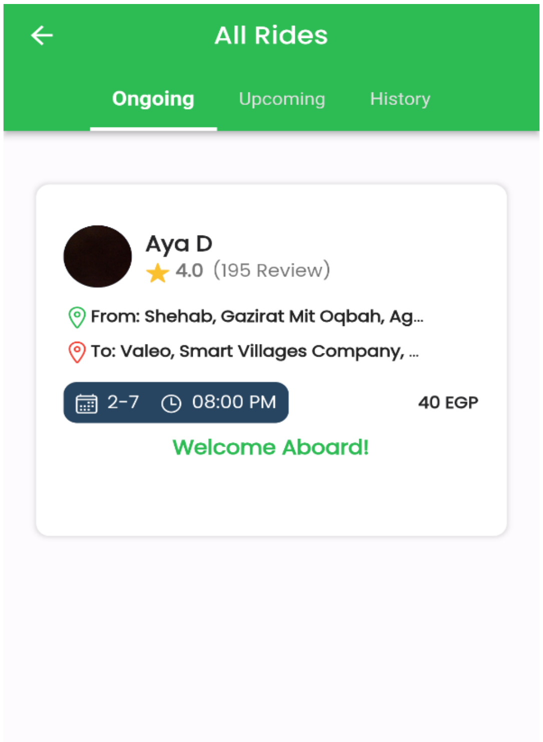
3-7 08:40 PM

Kia Rio | Red

60 EGP

2 seats left

Figure 16: Ride detail after sending the request



Aya D

4.0 (195 Review)

From: Shehab, Gazirat Mit Oqbah, Ag...

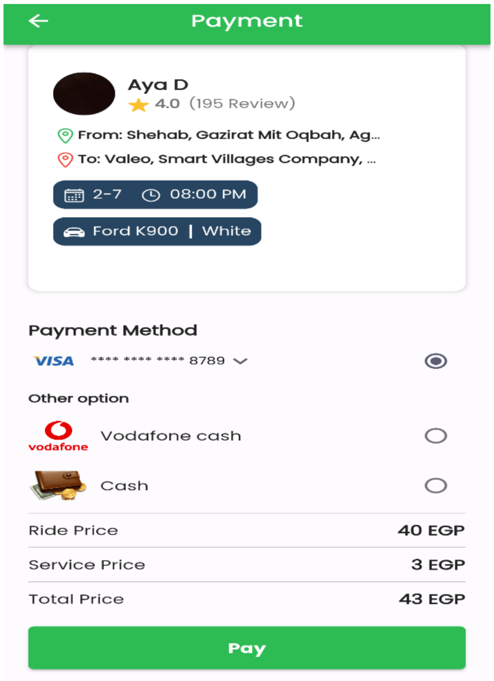
To: Valeo, Smart Villages Company, ...

2-7 08:00 PM

40 EGP

Welcome Aboard!

Figure 17: Ride started



Payment

Aya D
4.0 (195 Review)

From: Shehab, Gazirat Mit Oqbah, Ag...
To: Valeo, Smart Villages Company, ...

2-7 08:00 PM

Ford K900 | White

Payment Method

VISA **** **** **** 8789 ●

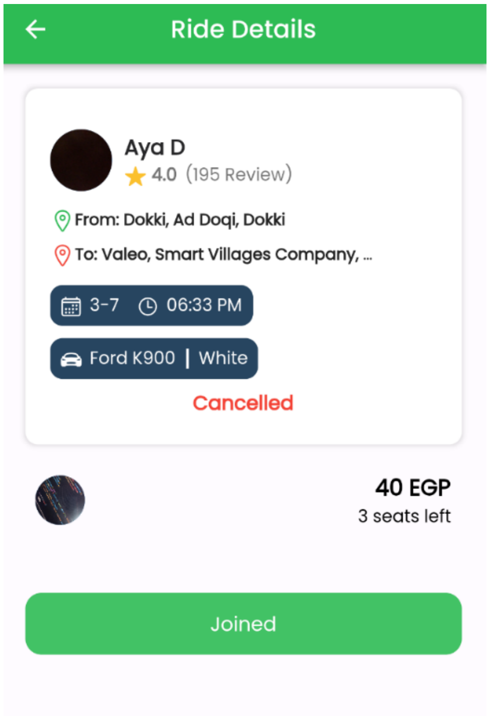
Other option

Vodafone cash ○

Cash ○

Ride Price	40 EGP
Service Price	3 EGP
Total Price	43 EGP

Figure 18: Pay ride after it ended



Ride Details

Aya D
4.0 (195 Review)

From: Dokki, Ad Doqi, Dokki
To: Valeo, Smart Villages Company, ...

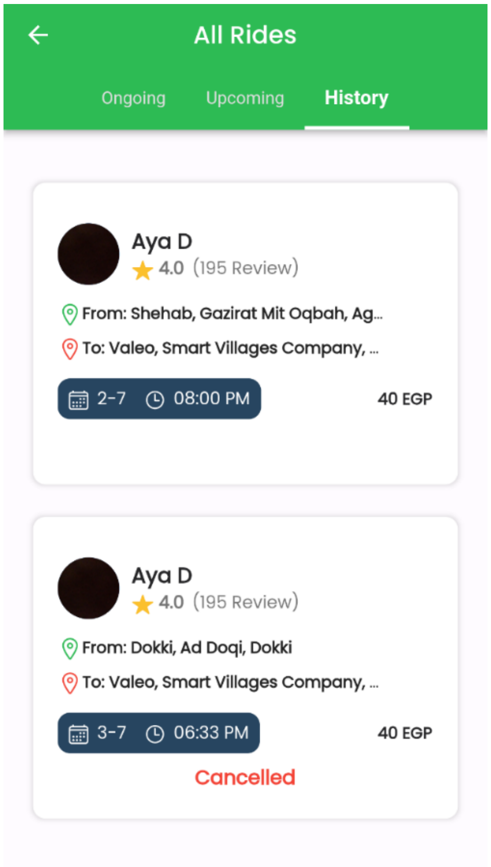
3-7 06:33 PM

Ford K900 | White

Cancelled

40 EGP
3 seats left

Figure 19: Ride details after it ended



All Rides

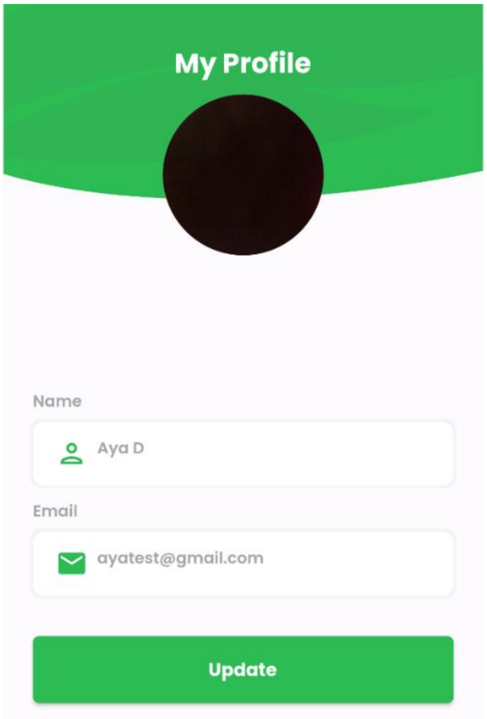
Ongoing Upcoming **History**

Aya D
★ 4.0 (195 Review)
From: Shehab, Gazirat Mit Oqbah, Ag...
To: Valeo, Smart Villages Company, ...
2-7 08:00 PM
40 EGP

Aya D
★ 4.0 (195 Review)
From: Dokki, Ad Doqi, Dokki
To: Valeo, Smart Villages Company, ...
3-7 06:33 PM
40 EGP
Cancelled

Figure 20: Ride History

Driver UI:



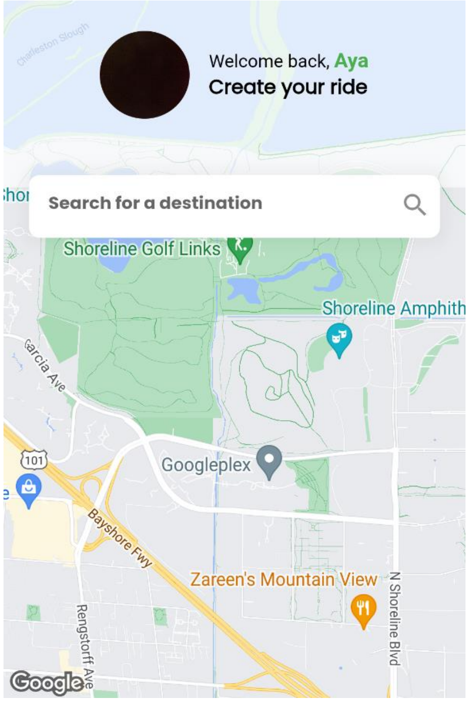
Name

Aya D

Email

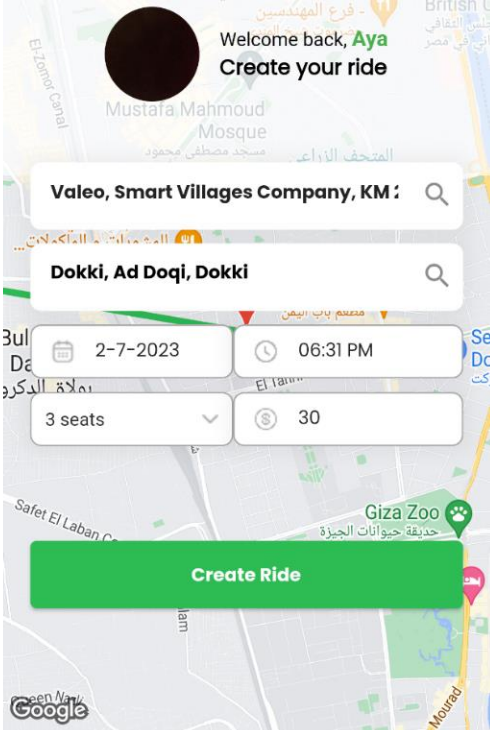
ayatest@gmail.com

Figure 21: Update profile page


Welcome back, Aya

Create your ride

Figure 22: Home page



Welcome back, Aya

Create your ride

Valeo, Smart Villages Company, KM 2

Dokki, Ad Doqi, Dokki

2-7-2023

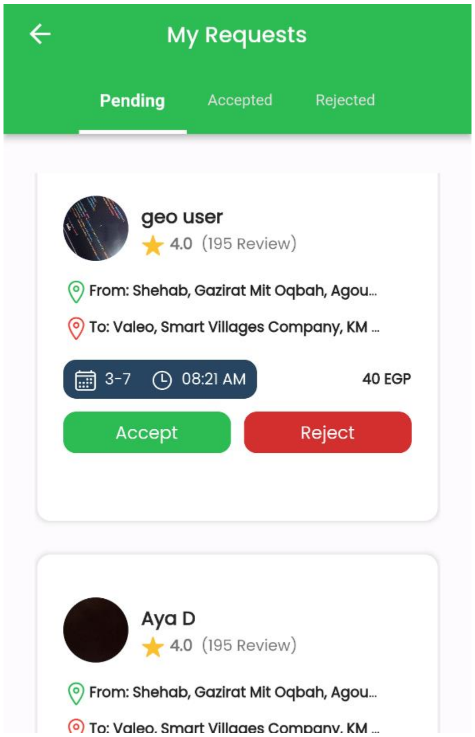
06:31 PM

3 seats

30

Create Ride

Figure 23: Create ride page



My Requests

Pending | Accepted | Rejected

geo user
4.0 (195 Review)

From: Shehab, Gazirat Mit Oqbah, Agou...

To: Valeo, Smart Villages Company, KM ...

3-7 08:21 AM

40 EGP

Accept

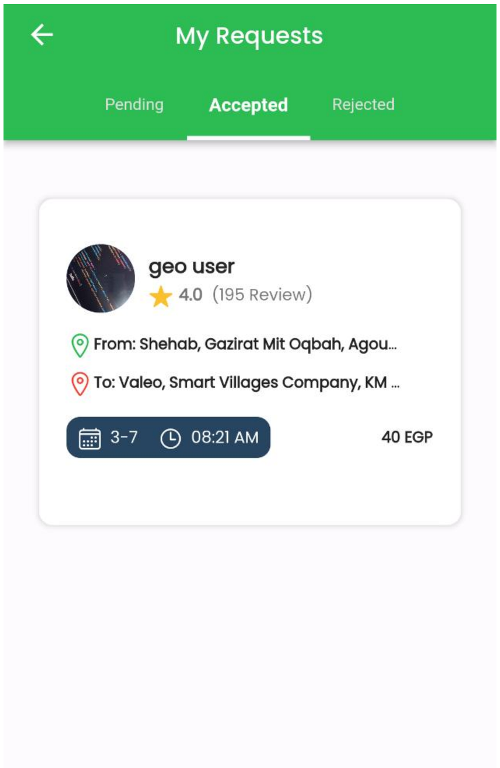
Reject

Aya D
4.0 (195 Review)

From: Shehab, Gazirat Mit Oqbah, Agou...

To: Valeo, Smart Villages Company, KM ...

Figure 24: Pending requests sent to user



geo user

4.0 (195 Review)

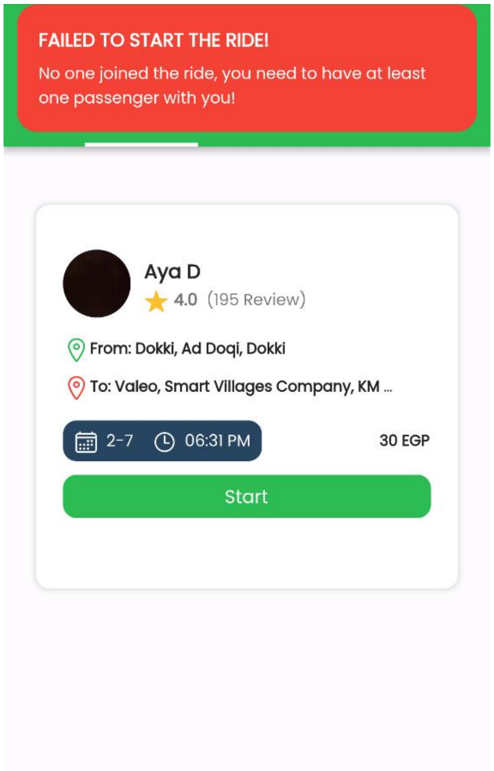
From: Shehab, Gazirat Mit Oqbah, Agou...

To: Valeo, Smart Villages Company, KM ...

3-7 08:21 AM

40 EGP

Figure 25: Accepted requests



FAILED TO START THE RIDE!

No one joined the ride, you need to have at least one passenger with you!

Aya D

4.0 (195 Review)

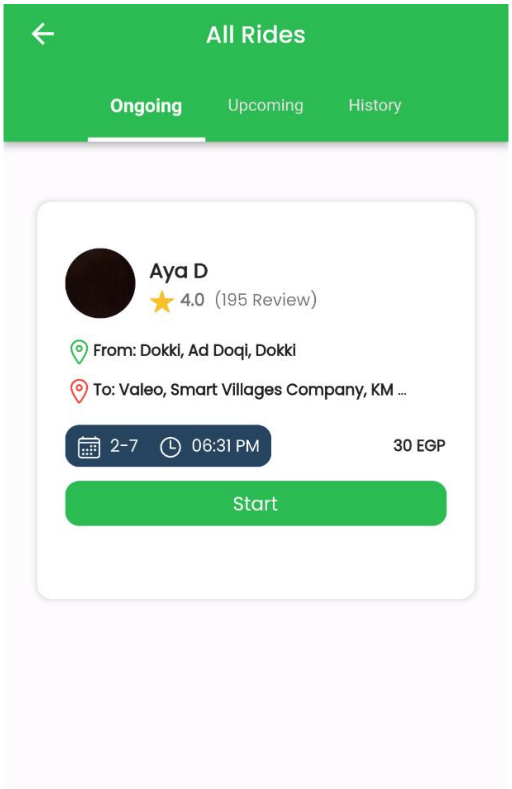
From: Dokki, Ad Doqi, Dokki

To: Valeo, Smart Villages Company, KM ...

2-7 06:31 PM

30 EGP

Figure 26: Error massage after trying to start ride when no one has joined the ride



Aya D

★ 4.0 (195 Review)

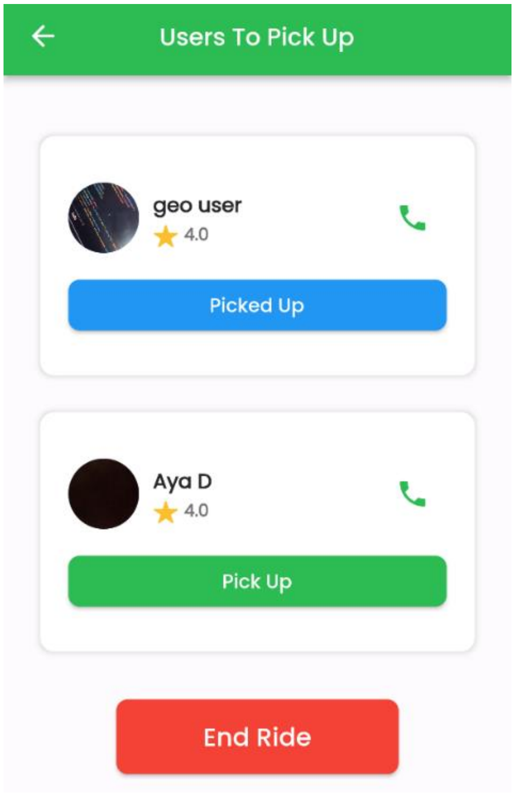
From: Dokki, Ad Doqi, Dokki

To: Valeo, Smart Villages Company, KM ...

2-7 06:31 PM

30 EGP

Figure 27: Time for ride to start



geo user

★ 4.0

Aya D

★ 4.0

Figure 28: Ride page after starting it

Chapter 5: Implementation and Testing

<table>
<tr><th colspan="7">User sign up positive scenario</th></tr>
<tr><td>Test Scenario</td><td colspan="4">Sign up</td><td>Test case ID</td><td>T-A1</td></tr>
<tr><td>Test Case</td><td colspan="4">Sign up positive scenario</td><td>Test Priority</td><td>1</td></tr>
<tr><td>Pre-condition</td><td colspan="6">None</td></tr>
<tr><td>Test case description</td><td colspan="6">User Sign up in the application with full name, image, address and phone number</td></tr>
<tr><td>Test steps</td><td>Test data</td><td>Expected output</td><td>Actual output</td><td>Test Browser</td><td>Test Result</td><td>Test comment</td></tr>
<tr><td>1-Decision screen</td><td rowspan="2">Personal information</td><td rowspan="2">Navigate to home screen</td><td rowspan="2">Navigate to home screen</td><td rowspan="2">Mobile android emulator</td><td rowspan="2">Pass</td><td rowspan="2"></td></tr>
<tr><td>2-User enter personal information</td></tr>
</table>

Table 4: User sign up positive scenario

<table>
<tr><th colspan="7">User sign up negative scenario</th></tr>
<tr><td>Test Scenario</td><td colspan="4">Sign up</td><td>Test case ID</td><td>T-A2</td></tr>
<tr><td>Test Case</td><td colspan="4">Sign up negative scenario</td><td>Test Priority</td><td>1</td></tr>
<tr><td>Pre-condition</td><td colspan="6">None</td></tr>
<tr><td>Test case description</td><td colspan="6">User Sign up in the application with full name, image, address and phone number</td></tr>
<tr><td>Test steps</td><td>Test data</td><td>Expected output</td><td>Actual output</td><td>Test Browser</td><td>Test Result</td><td>Test comment</td></tr>
<tr><td>1-Decision screen</td><td rowspan="2">Request information</td><td rowspan="2">Personal information</td><td rowspan="2">Display error authentication message</td><td rowspan="2">Display error authentication message</td><td rowspan="2">Pass</td><td rowspan="2"></td></tr>
<tr><td>2-User enter personal information</td></tr>
</table>

Table 5: User sign up negative scenario

Driver sign up positive scenario						
Test Scenario	Sign up			Test case ID	T-A3	
Test Case	Sign up positive scenario			Test Priority	1	
Pre-condition	None					
Test case description	Driver Sign up in the application with full name, image, email, car registration information and phone number					
Test steps	Test data	Expected output	Actual output	Test data	Expected output	Actual output
1-Decision screen	Personal information, car registration information	Navigate to home screen	Navigate to home screen	Personal information, car registration information	Navigate to home screen	Navigate to home screen
2-Driver enter personal information, car registration information						

Table 6: Driver sign up positive scenario

Driver sign up negative scenario						
Test Scenario	Sign up			Test case ID	T-A4	
Test Case	Sign up negative scenario			Test Priority	1	
Pre-condition	None					
Test case description	Driver Sign up in the application with full name, image, email, car registration information and phone number					
Test steps	Test data	Expected output	Actual output	Test Browser	Test Result	Test comment
1-Decision screen	Personal information	Display error authentication	Display error authentication	Mobile android	Pass	

2- Driver enter personal information, car registration information		message	message	emulator		

Table 7: Driver sign up negative scenario

<table>
<tr><th colspan="7">Update profile positive scenario</th></tr>
<tr><td>Test Scenario</td><td colspan="4">Update profile</td><td>Test case ID</td><td>T-A5</td></tr>
<tr><td>Test Case</td><td colspan="4">Update profile positive scenario</td><td>Test Priority</td><td>2</td></tr>
<tr><td>Pre-condition</td><td colspan="6">User signed in successfully and navigate to profile</td></tr>
<tr><td>Test case description</td><td colspan="6">Signed-in user should be able to update any information in profile at any given time such as name, image, address</td></tr>
<tr><td>Test steps</td><td>Test data</td><td>Expected output</td><td>Actual output</td><td>Test Browser</td><td>Test Result</td><td>Test comment</td></tr>
<tr><td>1-User profile screen</td><td rowspan="2">User’s name, image and address</td><td rowspan="2">Navigate to Home screen</td><td rowspan="2">Navigate to Home screen</td><td rowspan="2">Mobile android emulator</td><td rowspan="2">Pass</td><td rowspan="2"></td></tr>
<tr><td>2-User change any information</td></tr>
</table>

Table 8: Update profile positive scenario

Update profile negative scenario						
Test Scenario	Update profile			Test case ID		T-A6
Test Case	Update profile positive scenario			Test Priority		2
Pre-condition	User signed in successfully and navigate to profile					
Test case description	Signed-in user should be able to update any information in profile at any given time such as name, image, address					
Test steps	Test data	Expected output	Actual output	Test Browser	Test Result	Test comment
1-User profile screen	User's name, image and address	Display error validation message	Display error validation message	Mobile android emulator	Pass	
2-User change any information						

Table 9: Update profile negative scenario

Create ride positive scenario						
Test Scenario	Create ride				Test case ID	T-A7
Test Case	Create ride positive scenario				Test Priority	1
Pre-condition	Driver signed in successfully and navigate to home screen					
Test case description	Driver should be able to create ride with information such as pickup address, destination address, date, time, maximum number of seats and price per seat					
Test steps	Test data	Expected output	Actual output	Test Browser	Test Result	Test comment
1-Driver navigate to home screen	Ride information	Ride is added to the available rides for users	Ride is added to the available rides for users	Mobile android emulator	Pass	
2-Driver chooses the ride information						

Table 10: Create ride positive scenario

Create ride negative scenario						
Test Scenario	Create ride				Test case ID	T-A8
Test Case	Create ride negative scenario				Test Priority	1
Pre-condition	Driver signed in successfully and navigate to home screen					
Test case description	Driver should be able to create ride with information such as pickup address, destination address, date, time, maximum number of seats and price per seat					
Test steps	Test data	Expected output	Actual output	Test Browser	Test Result	Test comment
1-Driver navigate to home screen	Ride information	Display error validation message	Display error validation message	Mobile android emulator	Pass	
2-Driver chooses the ride information						

Table 11: Create ride negative scenario

User Filter rides all/joined/history positive scenario						
Test Scenario	Filter rides				Test case ID	T-A9
Test Case	Filter rides positive scenario				Test Priority	1
Pre-condition	User signed in successfully and navigate to All rides screen					
Test case description	User should be able to filter rides the different rides such as all rides or joined or the ended rides					
Test steps	Test data	Expected output	Actual output	Test Browser	Test Result	Test comment
1-User navigate to All rides screen	Ride information	Rides are filtered as if the user will	Rides are filtered as if the user will send	Mobile android emulator	Pass	

		send request or joined or end the ride	request or joined or end the ride			

Table 12: User Filter rides all/joined/history positive scenario

User Filter rides all/joined/history negative scenario						
Test Scenario	Filter rides			Test case ID	T-A10	
Test Case	Filter rides negative scenario			Test Priority	1	
Pre-condition	User signed in successfully and navigate to All rides screen					
Test case description	User should be able to filter rides the different rides such as all rides or joined or the ended rides					
Test steps	Test data	Expected output	Actual output	Test Browser	Test Result	Test comment
1-User navigate to All rides screen	Ride information	Rides are filtered as if the user will send request or joined or end the ride	Rides are not filtered as if the user will send request or joined or end the ride	Mobile android emulator	Fail	

Table 13: User Filter rides all/joined/history negative scenario

Driver Filter rides upcoming/history positive scenario			
Test Scenario	Filter rides	Test case ID	T-A11
Test Case	Filter rides positive scenario	Test Priority	1
Pre-condition	Driver signed in successfully and navigate to My rides screen		

Test case description	Driver should be able to filter rides the different rides such as rides he creates or the ended rides					
Test steps	Test data	Expected output	Actual output	Test Browser	Test Result	Test comment
1-Driver navigate to My rides screen	Ride information	Rides are filtered as if Driver will make it or end the ride	Rides are filtered as if Driver will make it or end the ride	Mobile android emulator	Pass	

Table 14: Driver Filter rides upcoming/history positive scenario

Driver Filter rides upcoming/history negative scenario						
Test Scenario	Filter rides			Test case ID	T-A12	
Test Case	Filter rides positive scenario			Test Priority	1	
Pre-condition	Driver signed in successfully and navigate to My rides screen					
Test case description	Driver should be able to filter rides the different rides such as rides he creates or the ended rides					
Test steps	Test data	Expected output	Actual output	Test Browser	Test Result	Test comment
1-Driver navigate to My rides screen	Ride information	Rides are filtered as if Driver will make it or end the ride	Rides are not filtered as if Driver will make it or end the ride	Mobile android emulator	fail	

Table 15: Driver Filter rides upcoming/history negative scenario

User send request positive scenario						
Test Scenario	Send request				Test case ID	T-A13
Test Case	Send request positive scenario				Test Priority	2
Pre-condition	User signed in successfully, navigate to All rides screen, select specific ride					
Test case description	User should be able to send request to specific driver on a ride					
Test steps	Test data	Expected output	Actual output	Test Browser	Test Result	Test comment
1-User navigate to All rides screen	Ride information	Request is sent to the driver as pending request	Request is sent to the driver as pending request	Mobile android emulator	Pass	

Table 16: User send request positive scenario

User send request negative scenario						
Test Scenario	Send request				Test case ID	T-A14
Test Case	Send request positive scenario				Test Priority	2
Pre-condition	User signed in successfully, navigate to All rides screen, select specific ride					
Test case description	User should be able to send request to specific driver on a ride					
Test steps	Test data	Expected output	Actual output	Test Browser	Test Result	Test comment
1-User navigate to All rides screen	Ride information	Request is not sent to the	Request is not sent to the driver as pending	Mobile android emulator	Pass	

		driver as pending request	request			

Table 17: User send request negative scenario

Driver accept request positive scenario						
Test Scenario	Accept request				Test case ID	T-A15
Test Case	Send request positive scenario				Test Priority	1
Pre-condition	Driver signed in successfully, navigate to My requests screen, accept specific request					
Test case description	Driver should be able to accept request from specific user on a ride					
Test steps	Test data	Expected output	Actual output	Test Browser	Test Result	Test comment
1-Driver navigate to My requests screen	Request information	Request status is accepted and user join the ride	Request status is accepted and user join the ride	Mobile android emulator	Pass	

Table 18: Driver accept request positive scenario

<table>
<tr><th colspan="7">Driver reject request positive scenario</th></tr>
<tr><td>Test Scenario</td><td colspan="4">Reject request</td><td>Test case ID</td><td>T-A16</td></tr>
<tr><td>Test Case</td><td colspan="4">Reject request positive scenario</td><td>Test Priority</td><td>1</td></tr>
<tr><td>Pre-condition</td><td colspan="6">Driver signed in successfully, navigate to My requests screen, reject specific request</td></tr>
<tr><td>Test case description</td><td colspan="6">Driver should be able to reject request from specific user on a ride</td></tr>
<tr><td>Test steps</td><td>Test data</td><td>Expected output</td><td>Actual output</td><td>Test Browser</td><td>Test Result</td><td>Test comment</td></tr>
<tr><td>1-Driver navigate to My requests screen</td><td rowspan="2">Request information</td><td rowspan="2">Request status is rejected and user not join the ride</td><td rowspan="2">Request status is rejected and user not join the ride</td><td rowspan="2">Mobile android emulator</td><td rowspan="2">Pass</td><td rowspan="2"></td></tr>
<tr><td></td></tr>
</table>

Table 19: Driver reject request positive scenario

References

1. Foorera Application:

https://foorera.com/

2. Flutter Documentation:

https://docs.flutter.dev/

3. Google Firebase Documentation:

https://firebase.google.com/docs

4. External Packages:

- ***Firebase Authentication:*** https://firebase.google.com/docs/auth
- ***Cloud Firestore:*** https://firebase.google.com/docs/firestore
- ***Google Maps API:*** https://developers.google.com/maps/documentation
- ***Firebase Analytics:*** https://firebase.google.com/docs/analytics
- ***Firebase Core:*** https://pub.dev/packages/firebase_core