

THRIDER

UCS503 Software Engineering Project Report End-Semester Evaluation

Submitted by :

(101903069) Paritosh Arora (Offline)
(101903420) Dhruv Mehta (Offline)
(101903418) Kunal Khullar (Offline)
(102083039) Agrima Malhotra (Offline)

BE Third Year, CoE

Group 3COE16

Submitted to: Dr Aashima



THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

Computer Science and Engineering Department TIET, Patiala

December 2021

Table of Contents	i
1. Project Selection Phase	1
1.1 Software Bid	1
1.2 Project Overview	4
2. Analysis Phase	5
2.1 Use Cases	5
2.1.1 Use-Case Diagrams	5
2.1.2 Use-Case Templates	6
2.2 Activity diagram and Swimlane Diagrams	7
2.3 Data Flow Diagrams (DFDs)	9
2.3.1 DFD level 0	9
2.3.2 DFD Level 1	10
2.3.3 DFD Level 2	10
2.4 Software Requirement Specification in IEEE Format	11
3. Design Phase	25
3.1 Class Diagram	25
3.2 Sequence Diagram	26
3.3 Collaboration Diagram	27
3.4 State Chart Diagrams	28

4. Implementation	29
4.1 Component Diagrams	29
4.2 Deployment Diagrams	30
4.3 Screenshots	31
5. Testing	35
5.1 Test Plan	35
5.2 Test Cases	36
5.2 Test Reports	44

1. Project Selection Phase

1.1 Software Bid

Team Name: Aztecs

Team ID (will be assigned by Instructor): 3

Please enter the names of your Preferred Team Members.

You are required to form **a three to four person** teams

Choose your team members wisely. You will not be allowed to change teams.

Name	Roll No	Project Experience	Programming Language used
Agrima Malhotra	102083039	4	React, Html, CSS, ROS,ML
Paritosh Arora	101903069	App Dev Internship + Hackathons, Web Dev	React Native, MERN
Kunal Khullar	101903418	Web Dev Internship + Hackathon experience	MERN, Html,JS
Dhruv Mehta	101903420	Full Stack WebDev, Robotics (ROS)	ROS,Firebase,Express,Node

Programming Language / Environment Experience

List the languages you are most comfortable developing in, **as a team**, in your order of preference. Many of the projects involve Java or C/C++ programming.

1. React
2. React Native
3. Firebase
4. Node

Choices of Projects:**1. Thapar Genie**

1. Swiggie genie+Uber
2. Mobile apps React Native Firebase
3. Goods/Items delivery from peer to peer via e-rikshaws already present on campus
4. If the user goes and buys something and requires extra carrier/help to put the goods then e rickshaw can be called
5. USP: passive/alternate source of income for e-rikshaws as they can pick and drop students while delivering the items from one location to another. When they have nothing to do they'll have something to do.
6. Extra Feature: Map of campus for convenience

2. Doctor Patient Appointment

1. Web MERN
2. Video Calling and scheduling appointment with Doctors
3. Doctors will get weekly/monthly reports of patients and will be displayed graphically.
4. Extra Feature : Chat Bot

3. Thapar Mart

1. Web+App
2. User orders, displayed on Sellers portal and delivery is done via seller only
3. Provides more organised way of online order fulfilment

4. Schedule Planner

1. Web+App
2. Todo list
3. Attendance manager
4. Calendar
5. Time Table + Alerts

Additional Remarks/ Inputs

Please tell us about any other factors that we should take into consideration (e.g., if you really would like to work on a project for some particularly convincing reason).

1. **Thapar Genie** among these projects seems to be the most ideal one as any such concept is not implemented as of now and moreover it would be a boon for both the rickshaw drivers as well as the consumers due to its uniqueness and being an passive source of income in these troubling times.
2. **Thapar Mart** is also another viable option as it streamlines the flow of online ordering without affecting any core working during the process. Also such concept has not been implemented as of now.

1.2 Project Overview

The Thrider is an 'on demand e-rickshaw and door to door delivery application' software. People are reluctant to get out of their hostel rooms as they are already preoccupied with all the college work and moreover the weather most of the time isn't ideal from going anywhere in the university just to get an item from a shop or from a friend. Apart from this, getting an e-rickshaw on campus right on time which requires vacant slots is difficult as it's completely on the basis of luck of the user. This portal provides a platform in the form of an application to everyone present on the campus. The users can sign up at the application. After logging in the users would see two broad options of either scheduling a door to door delivery or if they want any e-rickshaw on demand. Based on the result of the selected option, the user will be asked to enter all the relevant details. These details would be sent to all the e-rickshaw drivers in the nearby proximity and if anyone accepts the job then the user will be provided with relevant tracking details. To make the whole process secure, authentication would be required for both the e-rickshaw drivers and the users while logging in the application.

The e-rickshaw drivers across the campus can register themselves on our application specifying the necessary details. Once logged in then each driver will receive notifications regarding nearby rides and jobs. After going through the details, it's up to the driver if he/she wants to take up the job. This will help overcome the challenge of increased drop of passengers during weekends and during off hours as the application will allow e-rickshaw drivers to take up door to door delivery jobs also.

2. Analysis Phase

2.1 Use Cases

2.1.1 Use-Case Diagrams

A use case diagram is used to represent the dynamic behaviour of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships. In **Figure 1** we have 3 actors and 8 use cases.

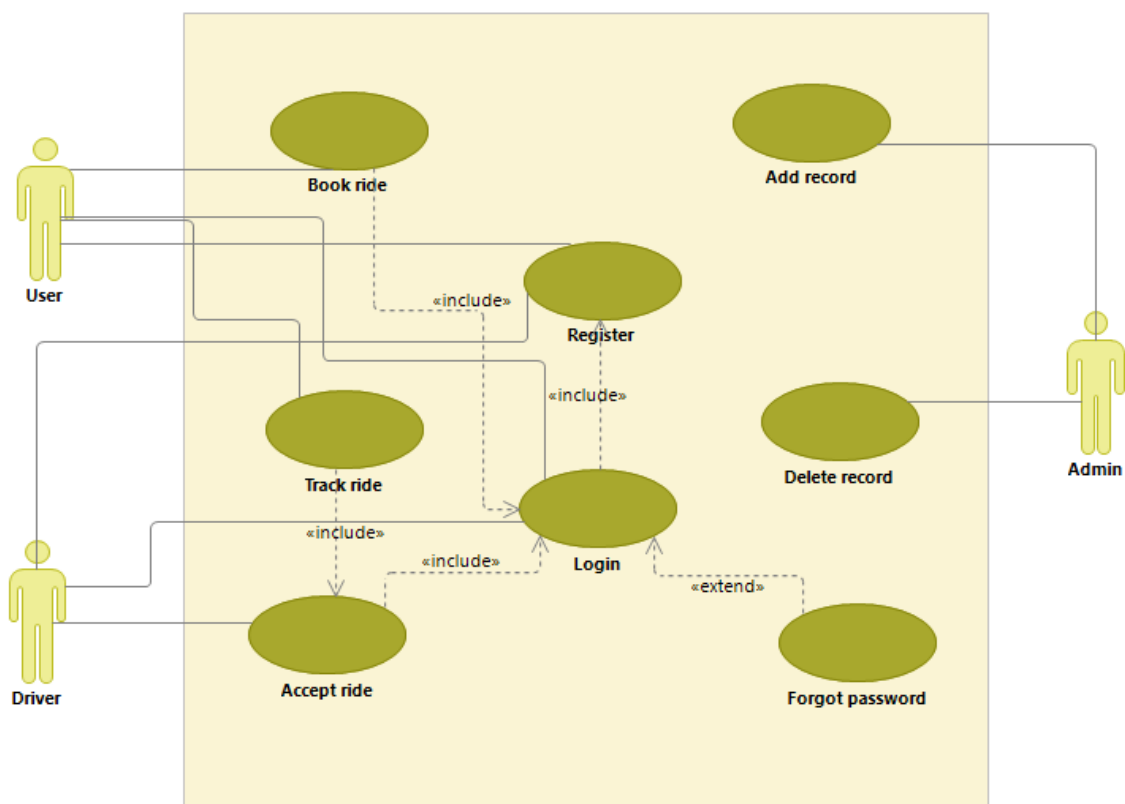


Figure 1

2.1.2 Use-Case Templates

This is the use case template corresponding to the use case diagram in **figure 1** .

Use case title	Thrider
Abbreviated title	Thrider
Use case id	1
Actors	User , driver , admin
Description With this facility users can book rides / drivers can accept as per convenience . Can be used to transport goods / travel from one place to another .	
Pre conditions	Users must be logged in .
Task sequence 1 . select book ride on screen . 2 . enter pickup / drop location . 3 . Driver accepts the request . 4 . Users can track the driver .	
Post conditions 1 . After driver reaches user pickup location he can pick/drop goods/user to the required destination . 2 . After task is done , user can book another ride / driver can accept another ride .	
Modification History	8-10-2021
Author	Dhruv Mehta, Paritosh Arora, Kunal Khullar, Agrima Malhotra

Use Case Template

2.2 Activity diagram and Swimlane Diagrams

An activity diagram is a behavioral diagram i.e. it depicts the behavior of a system. An activity diagram portrays the control flow from a start point to a finish point showing the various decision paths that exist while the activity is being executed. Figure 2 shows an activity diagram for thrider. The Figure 3 shows corresponding swimlane diagram

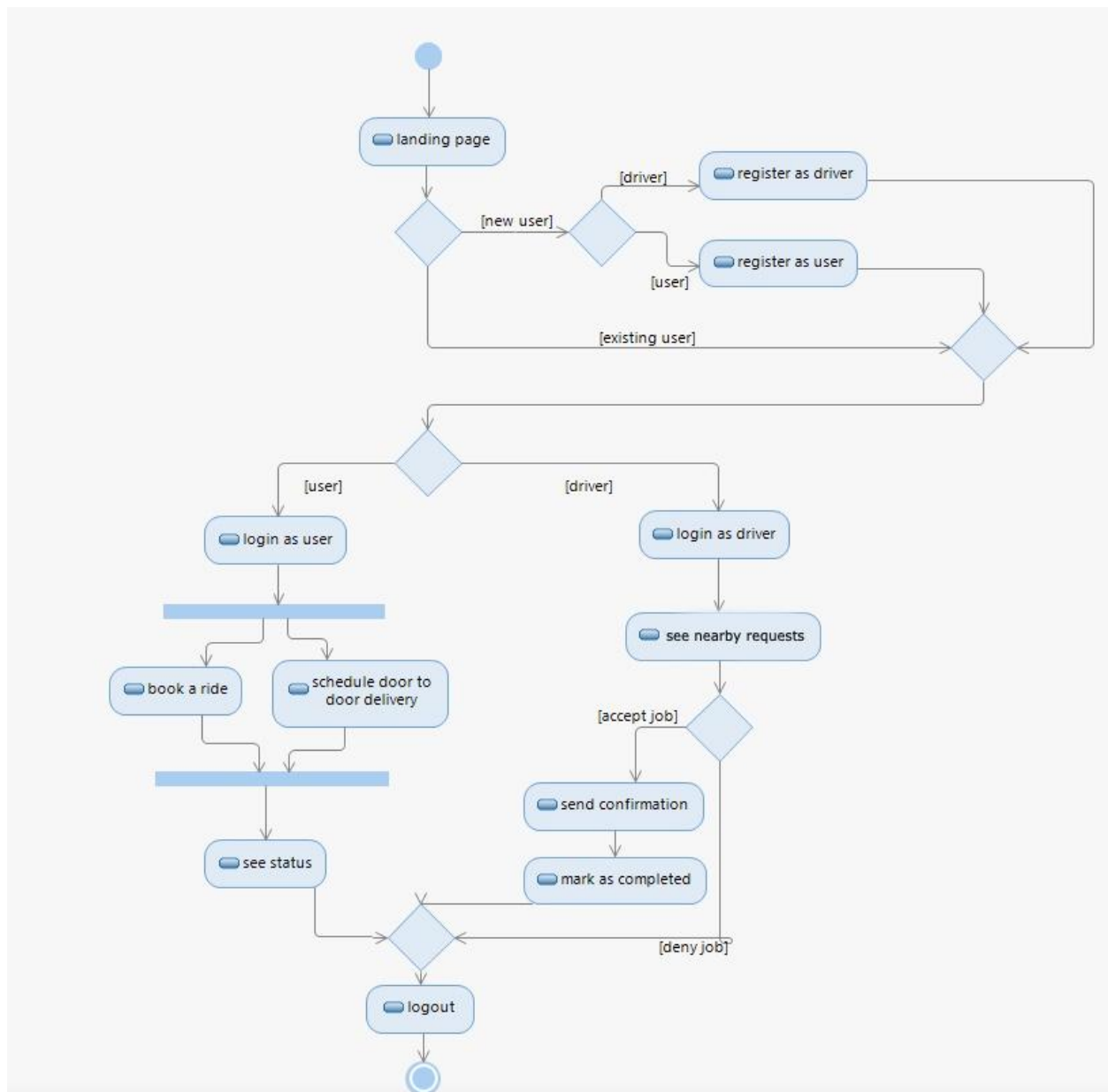


Figure 2

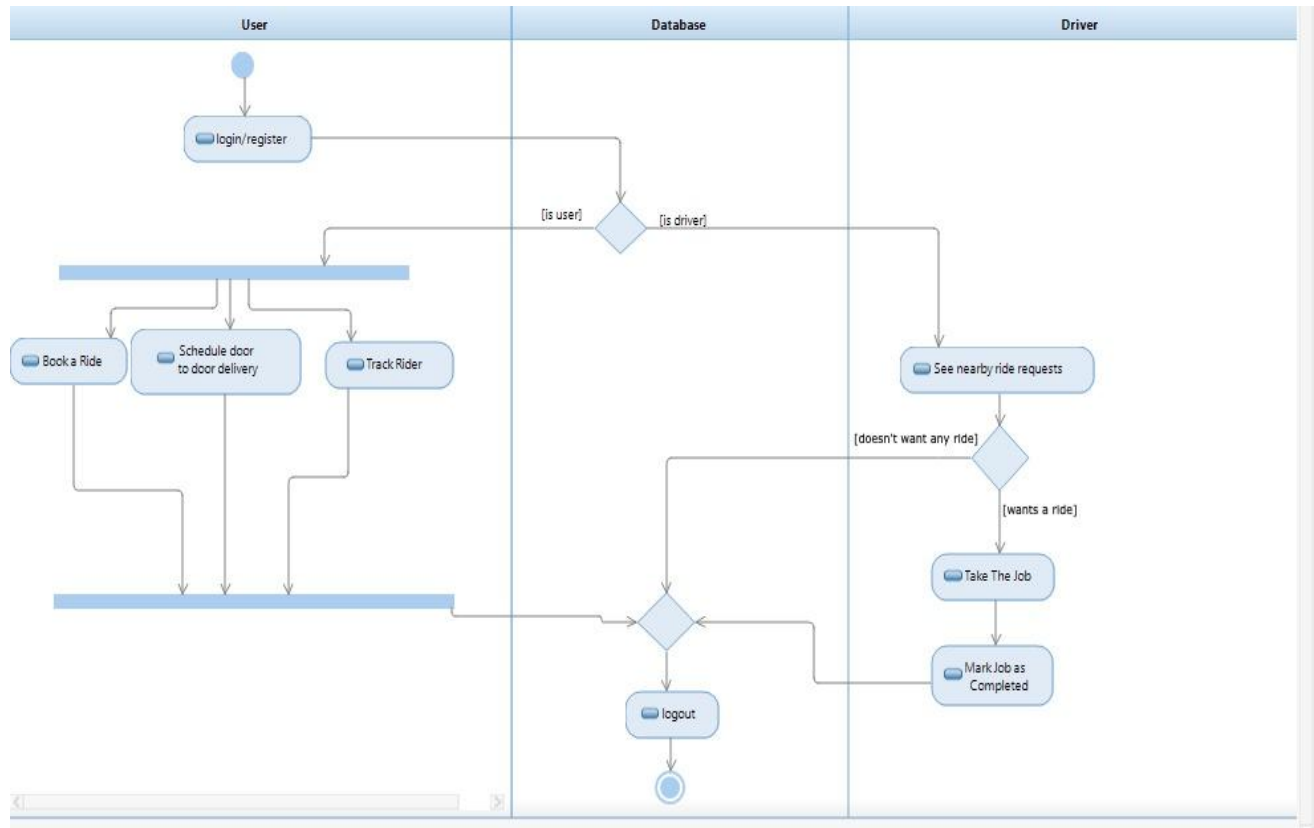


Figure 3

2.3 Data Flow Diagrams (DFDs)

DFD(data flow diagram) is drawn to represent the system of different levels of abstraction. Higher-level DFDs are partitioned into low levels-hacking more information and functional elements. Levels in DFD are numbered 0, 1, 2 or beyond. In **Figure 4, 5, 6** we will see mainly 3 levels in the data flow diagram, which are: 0-level DFD, 1-level DFD, and 2-level DFD.

2.3.1 Level 0 Data Flow Diagram

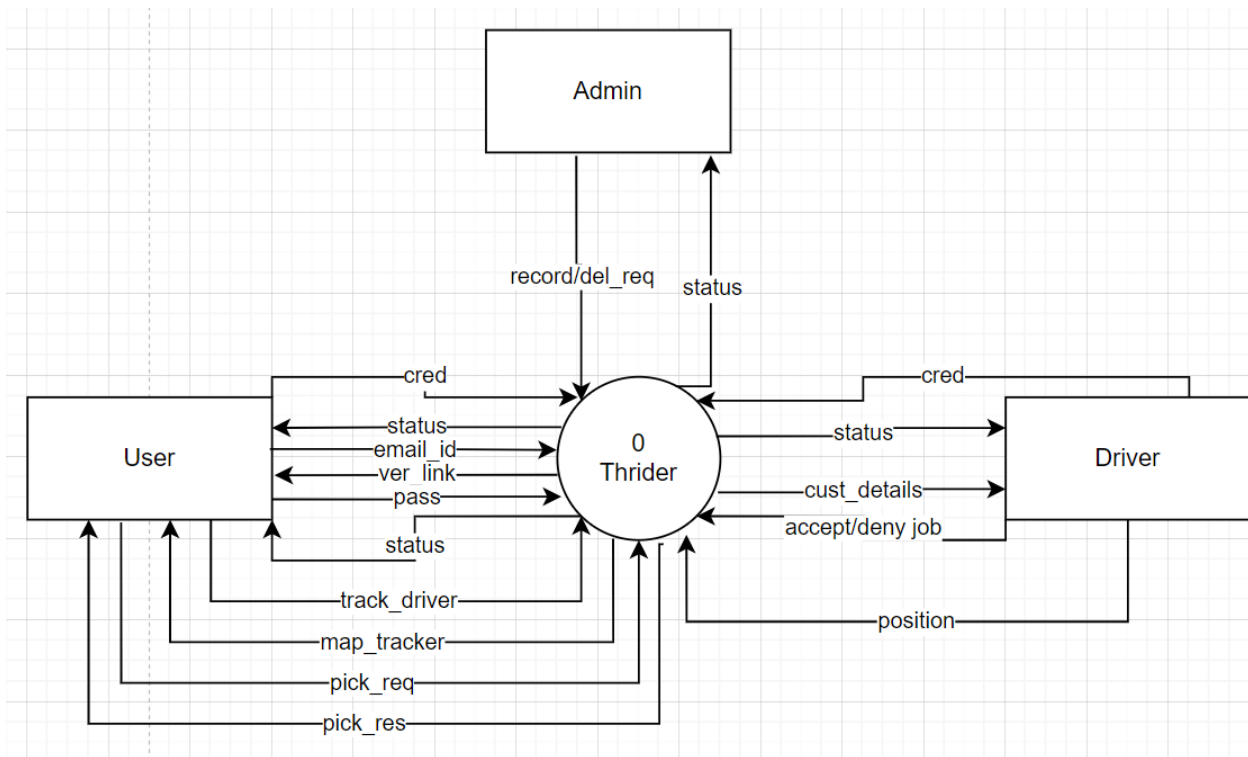


Figure 4

2.3.2 Level1 Data Flow Diagram

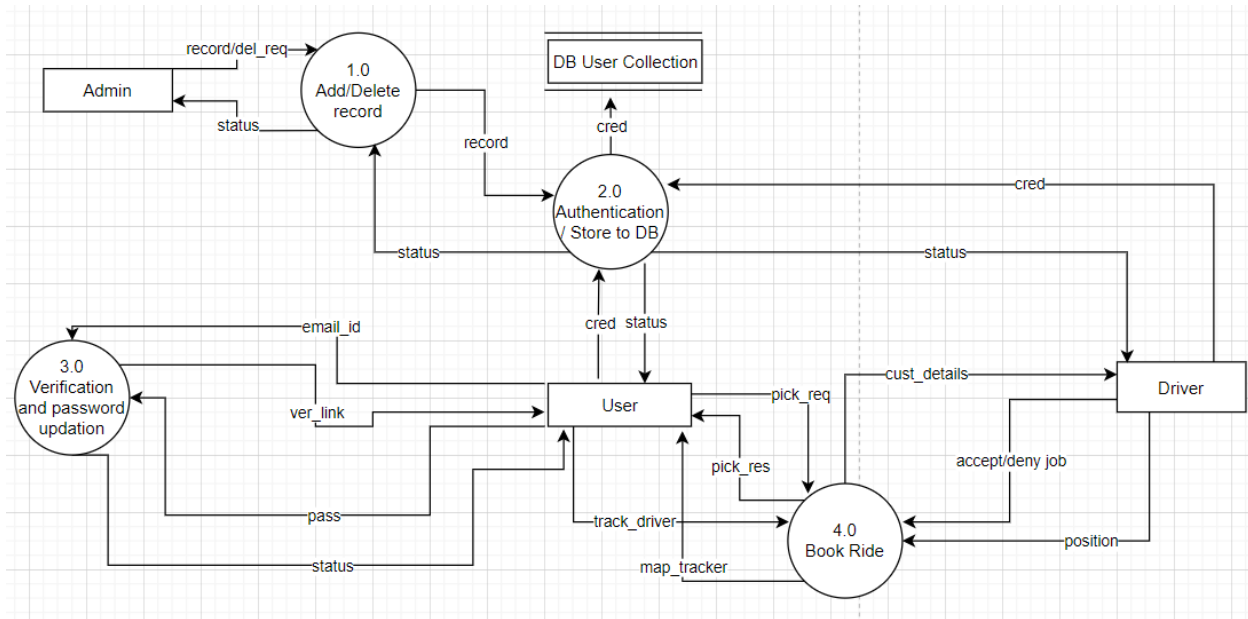


Figure 5

2.3.3 Level 2 Data Flow Diagram

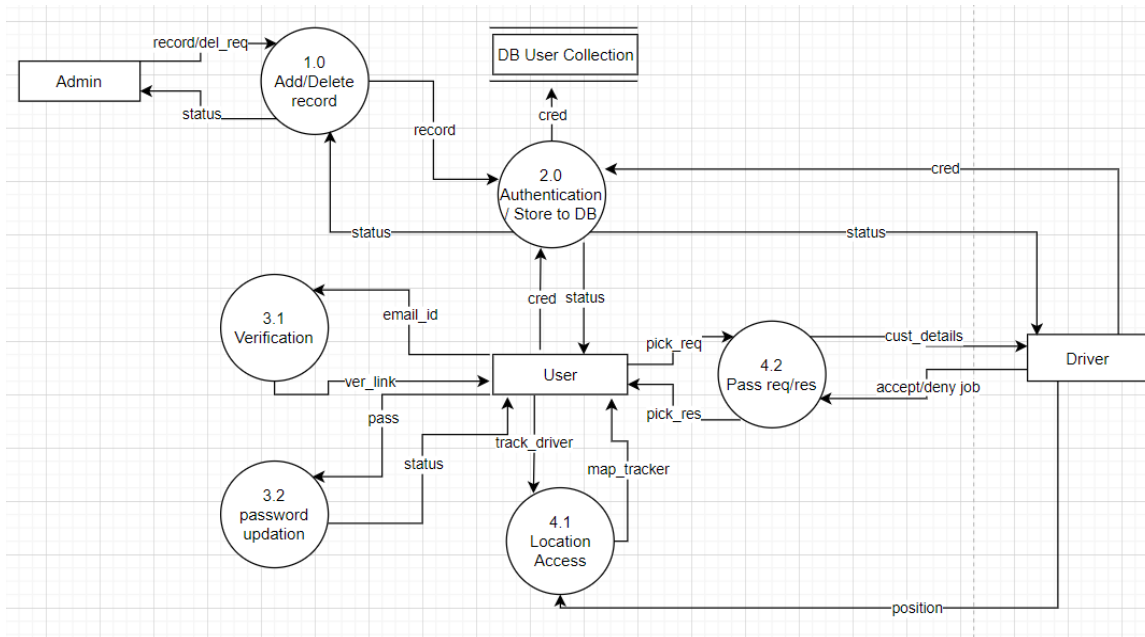


Figure 6

2.4 Software Requirements Specification

1. Introduction

1.1 Purpose

The purpose of this SRS document is to provide a detailed overview of our software product, its parameters and goals. This document describes the project's target audience and its user interface, hardware and software requirements. The goal of this project is to design and develop a mobile application which makes day to day life for everyone present in the Institute easy and convenient. With the beautiful UI from React Native, database powers of Firebase and server side data fetching using GraphQL and Apollo, we are going to develop a robust application. The idea is completely unique combining functionalities of door-to-door delivery of items as well as on demand e-rickshaws.

1.2 Document Conventions

This document was created based on the IEEE template for System Requirement Specification Documents.

1.3 Intended Audience and Reading Suggestions

The document is intended for developers, project managers, marketing staff, users, testers, stakeholders and documentation writers. The document has been divided into sections and further sub-sections depending on the aspects that need to be covered while developing the product so as to meet the required specifications. The first section gives an overview of the project- the scope and the intended audience, the conventions that are to be followed throughout the document. The second section examines the various perspectives of the product which includes the features included, viewpoints of different users to be considered, the limitations and the challenges to be faced while developing the product, the dependencies required and the

assumptions to be kept in mind. This section needs to be studied in detail by the developers since it covers each and every detail which will be required while developing the product. The third section covers the detailed description of each system feature. The users and testers must carefully read this section. These sections dealt with the functional requirements of the project. The fourth section deals with External Interface Requirements which will be needed for the product to function properly. The last section tells about parameters which should be at par with industry standards and will help evaluate the product in a better way.

1.4 Project Scope

The goal here is to develop a software making day to day commute and activities easy and convenient for users as well as generating a secondary source of income for e-rickshaw drivers. Currently there is a broad scope of this product as there is no such facility available on the campus and moreover the need will increase in the near future as the population present on campus increases. On the other hand e-rickshaw drivers face a hard time when there is immense heat out in the open or there are weekends during which the on foot crowd on the campus is pretty low in numbers, thus door to door delivery would become another source of income for them. Moreover the feature of ride along allows drivers to earn passive income by carrying people and delivering items at the same time, thus the overall effort required doesn't increase incredibly.

- Reaching from one place to another within a campus can be tiring as the campus is quite big, this is where the e-rickshaws come in action but getting an e-rickshaw right at the moment can be tough this is where this application helps anyone present on the campus. The user can book an e-rickshaw beforehand or when required using the application thus making the process of travelling from one place to another quite convenient and user friendly.
- The application makes the process of moving into a hostel or getting laundry done quite convenient, while moving in a hostel lots of shifting is required from one place to another consisting of heavy duty items like mattress, curtains, etc. Instead of carrying these alone, the app could be used to call an e-rickshaw and load the items in that.

- Door to door delivery of items inside the campus can be easily facilitated using this product. The students residing in the campus may require certain items from a place far from their respective hostel so to get these items easily an e-rickshaw can be booked for the same which would act as a door to door delivery vehicle.
- It is going to be an alternative source of income for the e-rickshaw drivers. When they are free, they can utilise their time to earn some more money as per their own convenience.

The scope of this system is not just limited to the university campus only as the same mechanism can be reused in other campuses as well. The application can be scaled to different universities and institutes after being properly implemented in Thapar.

1.5 References

- *IEEE Template for System Requirement Specification Documents:*
- *IEEE Computer Society, 1998*

2. Overall Description

2.1 Product Perspective

Our product is based on the idea of delivery of items from one hostel to another via E-rickshaws. It is an independent mobile platform which uses Google Maps API for information about routes. It is an easy to use application which just requires a working internet. It is user friendly and does not involve any complex functionality. It will also have multiple language support for the convenience of the E-rickshaw drivers. Lastly, it is completely software based and does not require any kind of hardware to function. The admins of an organization can look over different rides and ensure that everything is going how it is supposed to.

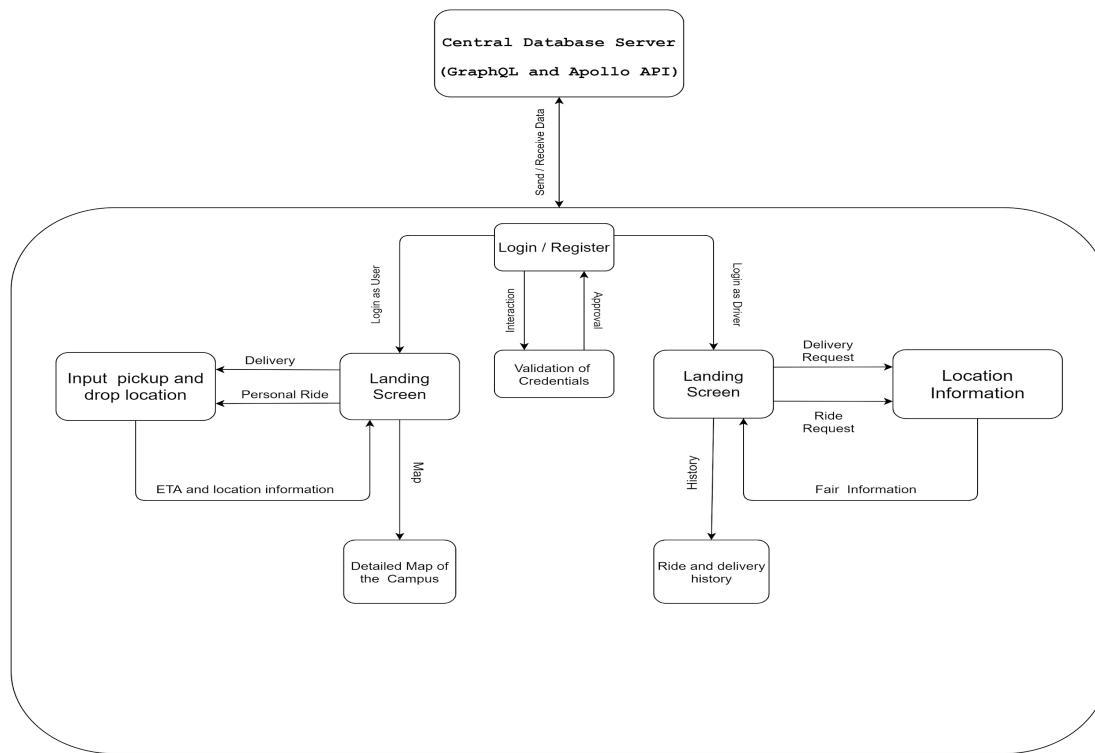


Figure 7

2.2 Product Features

1. Authentication: Users will first authenticate either as a user or an E-rickshaw driver. Users will have to upload a scanned picture of their ID proof while signing up as a security measure.

2. Door to Door delivery: Users who want to send items just have to request a pickup and enter the drop location. All the e-rickshaws in the proximity will be notified and have an option to take up the delivery. The charge for the trip will depend on the distance between the pickup location and the destination.

3. Ride Along: E-rickshaw drivers can drop users while delivering the items provided that the destination lies on the way of the delivery.

4. E-rickshaw on demand: Users can request a personal ride as well. They just have to enter the pickup and drop location and the e-rickshaw will reach there. The cost will depend on the

distance between the pickup and drop point and the number of passengers or if there is any luggage or extra load with the user.

2.3 User Classes and Characteristics

- Anyone present at the campus be it students, workers, teachers can use this application to their convenience. They can request a personal ride or a door to door delivery of items.
- E-rickshaw drivers who want to earn an extra buck can sign up on this platform.
- The user and e-rickshaw drivers have to sign up and login after which the session would start
- The user can then choose if he/she requires an e-rickshaw on demand or requires door to door delivery
- After user chooses type of task pickup and drop address along with time would be asked from user
- Estimated Price would be shown to the user depending on the distance and type of service chosen
- After getting confirmation all the e-rickshaws in proximity will receive a notification regarding what is the task along with all the relevant details
- When a driver would accept the job the user would be notified and after successful completion another notification would be passed to the user

2.4 Operating Environment

The project is an android application which can work on any of these devices with internet connections. Devices with the latest hardware are optimal as compared to the old hardware for accurate results. The device must have sufficient memory and should have functional tracking hardware. The device must be compatible with the android version 6.0(Marshmallow) and above.

1. **Front-End:** React Native
2. **Backend:** GraphQL and Apollo
3. **Database:** Firebase
4. **Mobile Operating System:** Android Marshmallow, v6.0 or above
5. **Mobile hardware:** ARM Android devices.

2.5 Design and Implementation Constraints

1. The software requires an Android device running version 6.0 (Marshmallow) or later with a touch screen and simulated keyboard.
2. The software must be able to access the device's storage and location in order to function.
3. The front-end interface should be easy enough for the users to understand and operate.
4. If no internet connection is available, users will be unable to log in or access any functionality of the software.
5. There must be enough storage on the mobile device.

2.6 User Documentation

This project is developed such that the total appearance of it is more user friendly. General users with basic computer skills and knowledge of accessing the internet can use this software. This project will include a user manual. The user manual will include a complete overview of the system, and basic guide on how to use the various features of the software such as the to-do list and the various schedule management features step by step. It will also contain the contact information which will include contact number, phone number and email address.

2.7 Assumptions and Dependencies

1. Since we are using google maps API for location data, the location is real time and does not have any lag.
2. The latency time of the database is normal, amidst the number of requests made to it.
3. All the external libraries used in the project are not deprecated in the future.

4. As we are using hosting from third party websites so the availability and speed of connection will depend upon the services provided by them.

3. System Features

3.1 Registering User

3.1.1 Description and Priority

The given feature will create an account for the user on our mobile app using which their records can be stored on our database and they can then use all the intended features. For privacy reasons we cannot let users use the other functionalities of the app without an account so their data is first required to be stored in our database so that they can use this application. As without it anything else cannot be used so this feature is a high priority. The user would be prompted regarding what type of account is to be created as an e-rickshaw driver or generic user.

3.1.2 Stimulus/Response Sequences

1. User will Click on the register button.
2. The user will be presented with a form in which they will fill all the required details.
3. The System will check the validity of the password and email followed by whether the data is valid or not.
4. A user account will be created if all the details are valid.

3.1.3 Functional Requirements

1. Navigation to Landing Screen:

- a. Purpose:** By default for first time users a landing page will be shown as the user won't be authenticated. The screen would prompt the user to either login or register using credentials.
- b. Input:** User will choose to register as a new user.

- c. Output and Errors:** A modal view will be displayed prompting users to enter details and create an account.

2. User Details:

- a. Purpose:** The user needs to input his/her credentials and all the relevant information asked.
- b. Input:** User provides with the information wherever prompted
- c. Output and Errors:** The user will get a prompt of successfully registering if the details are valid and their account will be created. If a user misses any important field then the form will not submit and the system will prompt the user to fill that detail. If the details are invalid or the password doesn't match then the user will be prompted with an error message that the password is wrong or the email is already registered depending on the condition.

3.2 Login User

3.2.1 Description and Priority

The given feature will login an existing user to the app and will grant him full functionality of the app. For privacy reasons we cannot let users use the other functionalities of the app without an account so their data is first required to be stored in our database so that they can use this application. As without it anything else cannot be used so this feature is a high priority.

3.2.2 Stimulus/Response Sequences

1. User will click login button
2. The user will enter credentials in the modal form
3. The System will check the validity of the password and email followed by whether the data is valid or not and on the basis of which the user will be presented with the output.

3.2.3 Functional Requirements

1. Navigation to Landing Screen:

- a. Purpose:** If the user is not is not authenticated, the first screen visible will be the Authentication screen, which prompts the users to either login or register to the app.
- b. Input:** The user needs to click on the login button.
- c. Output and Errors:** The user will be presented with a modal, with auth details to fill.

2. User Details:

- a. Purpose:** The user needs to input his/her credentials and all the relevant information asked.
- b. Input:** User provides with the login credentials
- c. Output and Errors:** The user will be logged in if the details are valid and their account details will be displayed and they can use the functionality of the website. If the password or email does not match to the records in the database then the user will be prompted to fill the form again with correct details or they will be given an option to register if they haven't done it already.

3.3 On Demand E-Rickshaw

3.3.1 Description And Priority

The given feature will enable users to get an e-rickshaw at a scheduled time or at the same instant of booking one. The user, if wants to travel or wants to travel with some items, can schedule a booking using this feature, thus making the process of travelling way more convenient and easy. There would be a button saying “Book a ride” on the home screen itself when the user will choose it, he/she will be prompted to fill out the required details.

3.3.2 Stimulus/Response Sequences

1. Users will click on the “book a ride” button.
2. The user will be shown a modal view where the pick up location, drop location and number of passengers will be taken as input.
3. A price estimate would be shown.
4. Users will have two options either of confirming booking or going back.
5. Upon confirmation the notification will be sent out to e-rickshaws in the proximity

3.3.3 Functional Requirements

1. **Book a Ride button:**
 - a. **Purpose:** This will prompt users to enter details regarding the ride and journey.
 - b. **Input:** The user needs to press the button
 - c. **Output and Errors:** The user will be shown estimated price for the ride and upon confirmation a notification will be sent to nearby e-rickshaws

3.4 Door to Door Delivery

3.4.1 Description And Priority

The given feature will enable users to get an e-rickshaw at a scheduled time or at the same instant of booking one. The user, if wants to get some item picked up or send some item to another place within the campus, can schedule a booking using this feature, thus making the process of sending/getting the item way more convenient and easy. There would be a button saying “D2D Delivery” on the home screen itself when the user will choose it, he/she will be prompted to fill out the required details.

3.4.2 Stimulus/Response Sequences

1. Users will click on the “D2D delivery” button.
2. The user will be shown a modal view where the pick up location, drop location will be taken as input.
3. A price estimate would be shown.
4. Users will have two options either of confirming booking or going back.

5. Upon confirmation the notification will be sent out to e-rickshaws in the proximity

3.4.3 Functional Requirements

1. Book d2d button:

- a. **Purpose:** This will prompt users to enter details regarding the ride.
- b. **Input:** The user needs to press the button
- c. **Output and Errors:** The user will be shown estimated price for the ride and delivery and upon confirmation a notification will be sent to nearby e-rickshaws

3.5 Accepting Tasks

3.5.1 Description And Priority

The feature enables e-rickshaw drivers to see what jobs are active right now in the proximity and gives the user an option to either accept the job or decline the job. The screen will display a modal popup containing every information regarding the ride like pickup location, drop location, etc.

3.5.2 Stimulus/Response Sequences

1. Driver will see a prompt giving out details of the ride
2. The driver will have an option to either reject the job or accept it
3. Upon accepting the job concerned user will be notified

3.5.3 Functional Requirements

1. Prompt Model

- a. **Purpose:** This will act as a prompt display showing the driver all the necessary information regarding the ride.
- b. **Input:** The driver needs to either accept or reject the job.
- c. **Output and Errors:** The user will be shown estimated price for the ride and all the necessary information regarding the ride, upon confirmation a notification will be sent to nearby e-rickshaws.

3.6 Tracking ongoing task

3.6.1 Description

The feature enables the users to track their ongoing task giving out information regarding how far the e-rickshaw allotted to them is and how much more estimated time would be required for the completion of the task.

3.6.2 Stimulus/Response Sequences

1. Users will be able to see a screen showing all the relevant information.
2. Estimated time and distance will also be shown to the user.

3.6.3 Functional Requirements

1. Prompt Model

- a. Purpose:** This screen will show all the necessary details to the user after a task order has been placed.
- b. Input:** The user needs to navigate to the tracking screen.
- c. Output and Errors:** The user will be shown estimated time and distance for the task completion along with the details of the driver.

4. External Interface Requirements

4.1 User Interfaces

Mobile applications will act as a GUI(Graphical User Interface) for the user.

4.2 Hardware Interfaces

The user should have access to any iOS or android phone .

4.3 Software Interfaces

The application should be available for android, iOS, and web-based ecosystems. Anyone using an iPhone, iPad, an android device should be able to use the application.

4.4 Communications Interfaces

HTTPS Protocol is the standard protocol for communicating on the web and it will be used to access the website so that the information that is personal to the user is not intercepted by any attacker or any unauthorized personnel. HTTPS encrypts the data so that even if it is intercepted the attacker will still not be able to decipher any information. POST requests are used to send form data so that the data is not directly visible so that the privacy of the user is maintained and no information is leaked.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- It is a mobile application and works on all compatible devices.
- Minimal performance requirements on the user end, most of the processing is done on the server side. Just a mobile device is required.
- A stable internet connection will ensure efficient and fast performance.

5.2 Safety Requirements

- Book when required. Users should not play pranks with the driver, i.e., making a false request to call the e - rickshaw when it is not required.
- Cancelling should be avoided. As a good practise both riders and drivers should avoid cancelling booked rides at end moments. These things may lead to loss of time / frustration among users.
- Theft can happen. There are both good and bad people in society. Drivers may steal stuff as well. So, avoid delivering precious items like jewellery, expensive gadgets, etc through them.

5.3 Security Requirements

- Users should log out of the app when they are done with the work. If we leave the session open, someone else can extract sensitive information.
- When a user will perform any action, proper authorization will be made for him/her to perform allowed action and an error message will be displayed- Action is unauthorized.
- App will have a timeout functionality. This means if users are logged in and not using the app for 10 - 15 days, they will be automatically logged out. Next time they want to use the app, they would have to login again. This can prevent loss of private data, i.e., if someone except the user(logged in) is using the app.

5.4 Software Quality Attributes

- As developers, we will try to minimise bugs in code of application as much as possible. We will try to deliver a reliable, robust application for our users after doing a decent amount of testing.
- Application works across all compatible mobile devices.
- It is maintenance free. Once deployed, it does not need any big changes. Although, after adding new features (if required), a newer version can be released anytime later on.

3. Design Phase

3.1 Class Diagram

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. In **Figure 8** we have 8 classes.

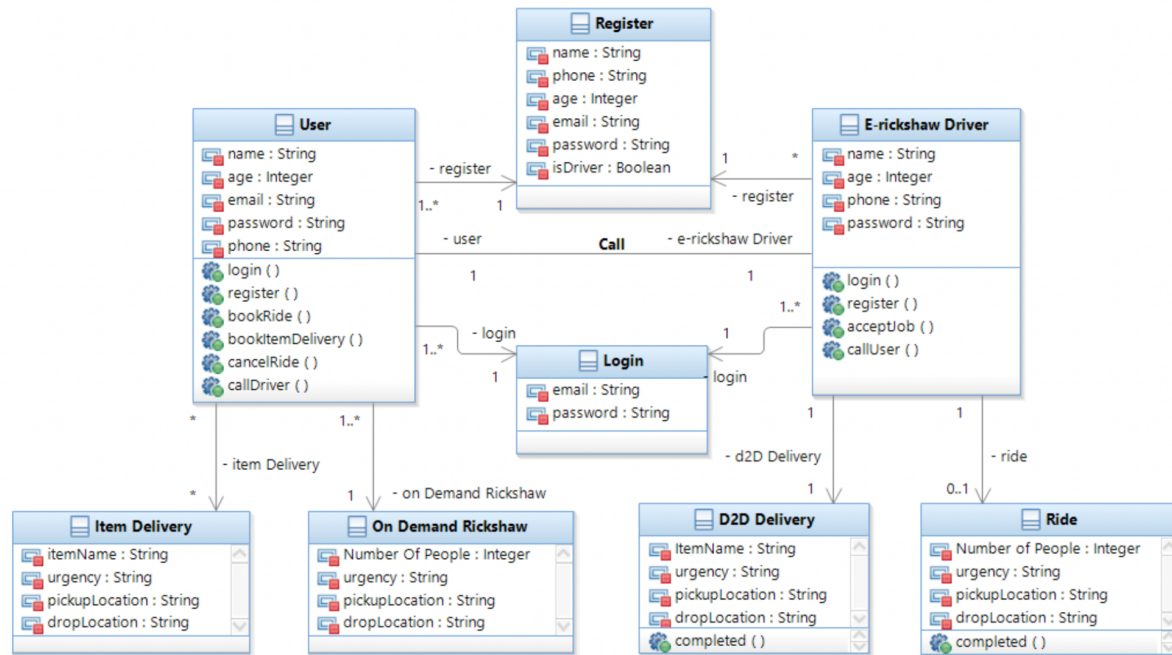


Figure 8

3.2 Sequence Diagram

A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. Sequence diagrams describe how and in what order the objects in a system function. In **Figure 9** we have 7 objects and various instances.

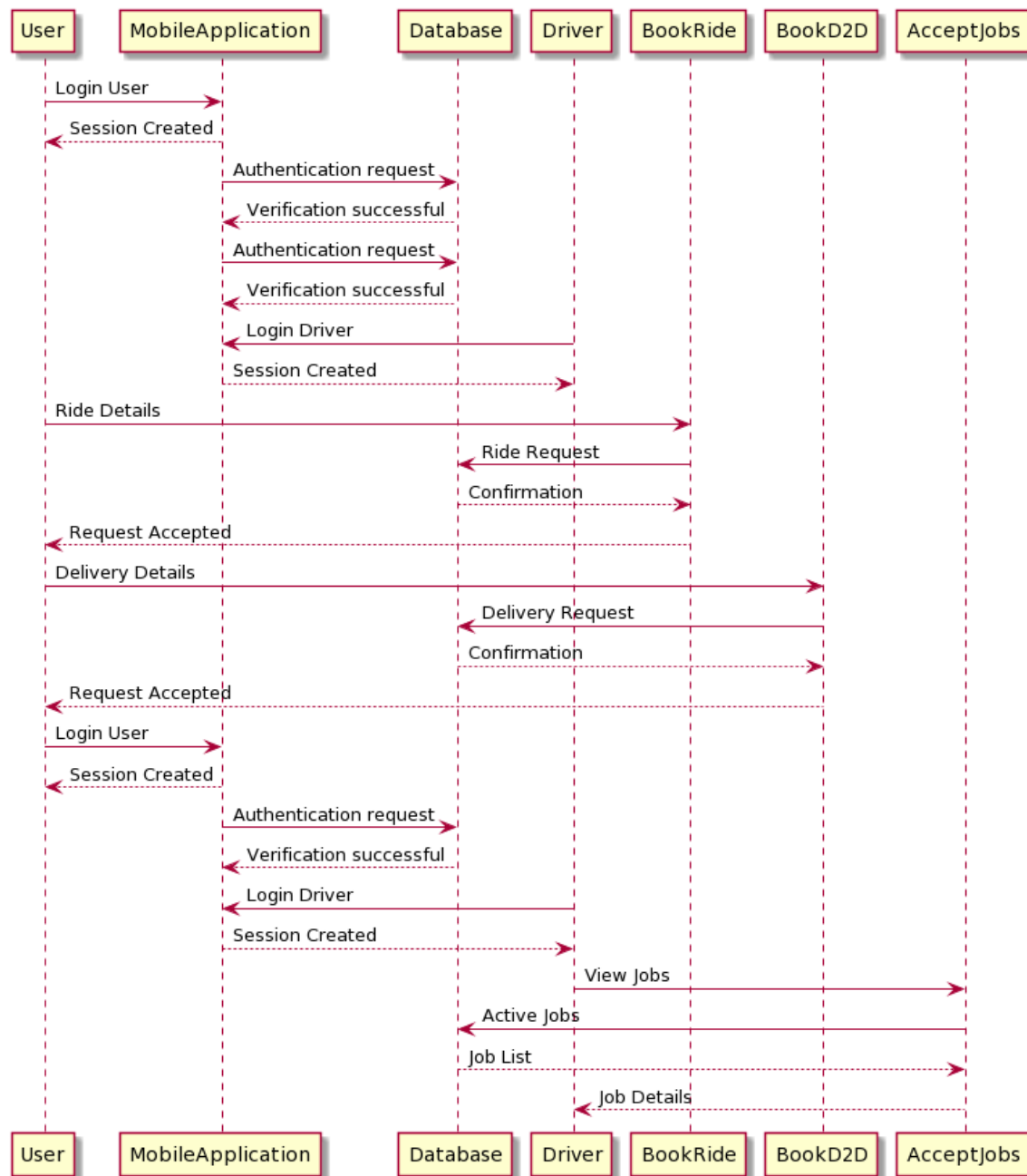


Figure 9

3.3 Collaboration Diagram

The collaboration diagram is used to show the relationship between the objects in a system. Both the sequence and the collaboration diagrams represent the same information but differently. Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming. An object consists of several features.

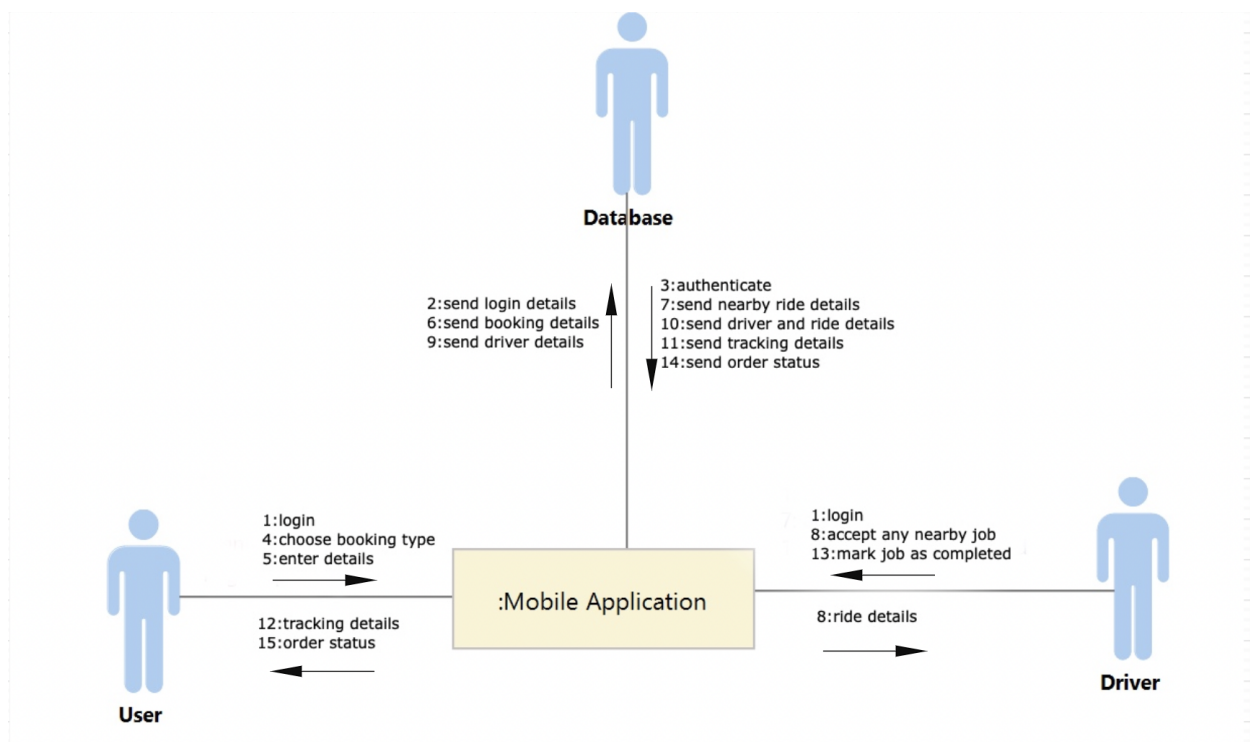


Figure 10

3.4 State Chart Diagram

Statechart diagram is used to describe the states of different objects in its life cycle. Emphasis is placed on the state changes upon some internal or external events. These states of objects are important to analyze and implement them accurately.

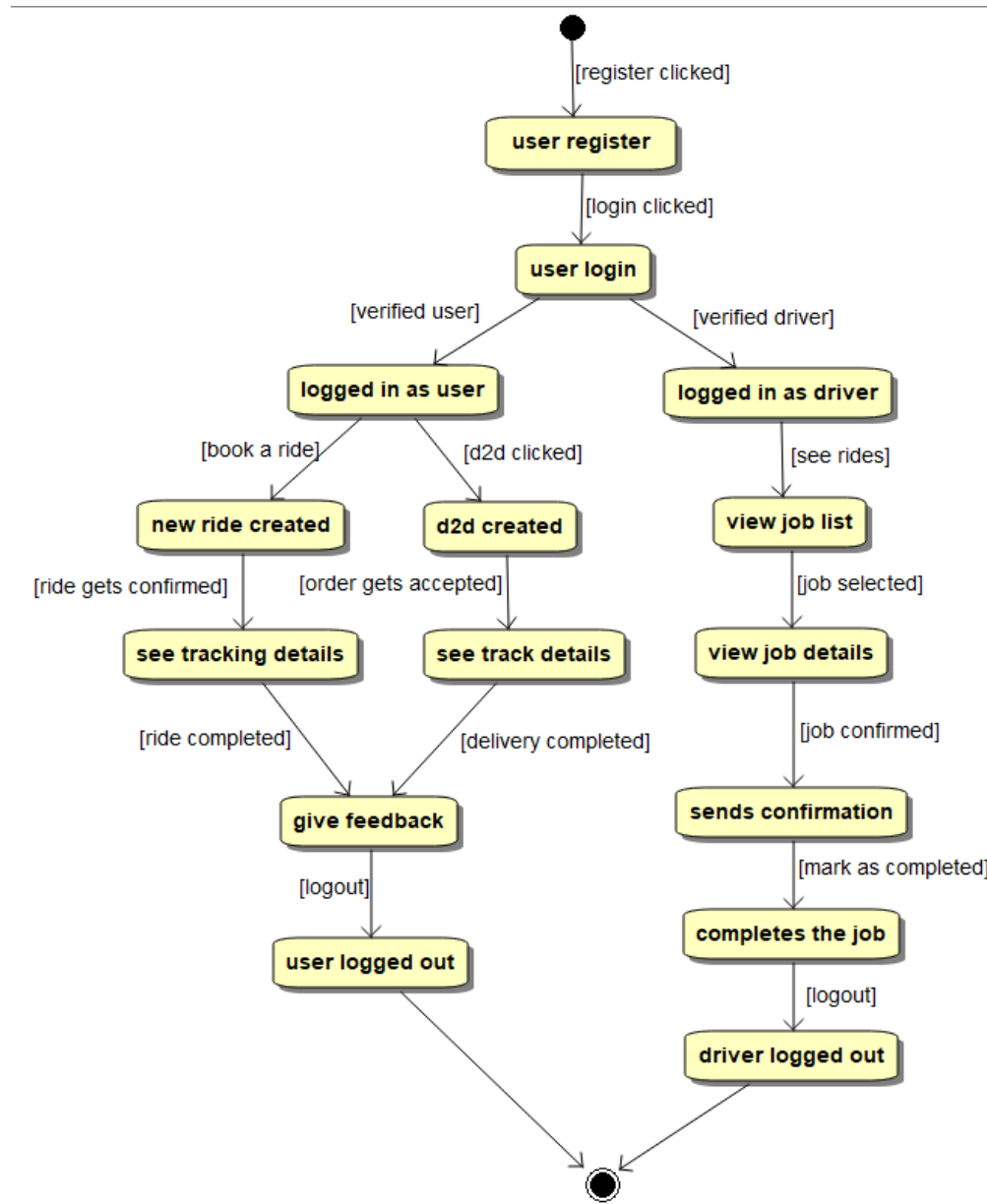


Figure 11

4.2 Deployment Diagram

A deployment diagram is a UML diagram type that shows the execution architecture of a system, including nodes such as hardware or software execution environments, and the middleware connecting them.

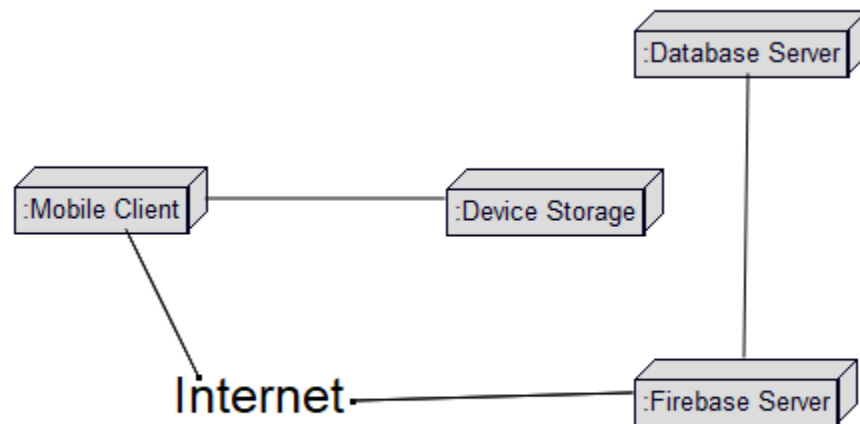


Figure 13

4.3 Screenshots

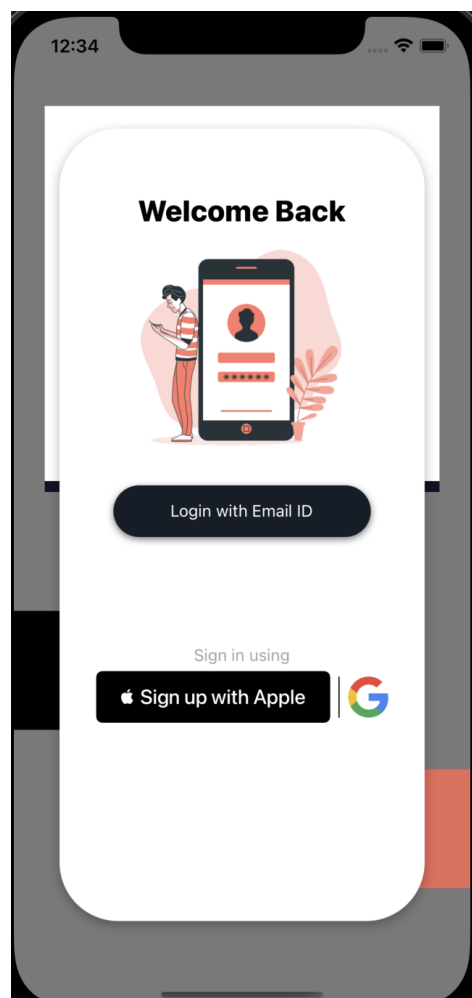



Figure 14,15



Enter your Email ID

Enter your password

Next

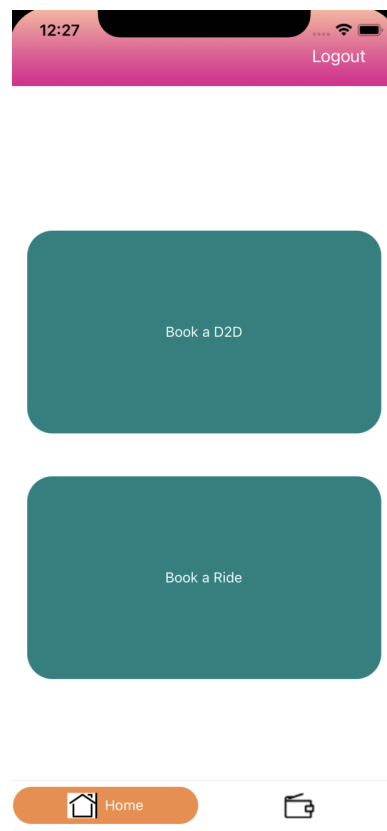


Figure 16,17

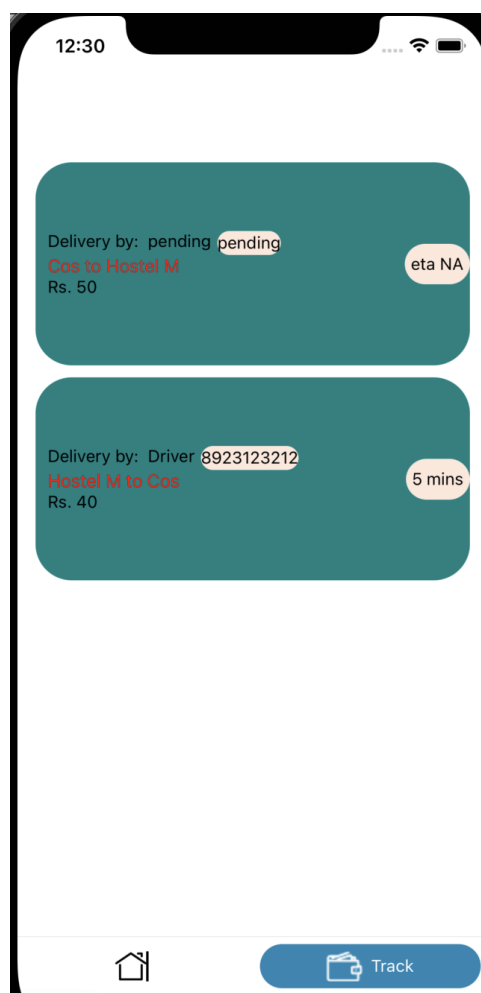


Figure 18,19

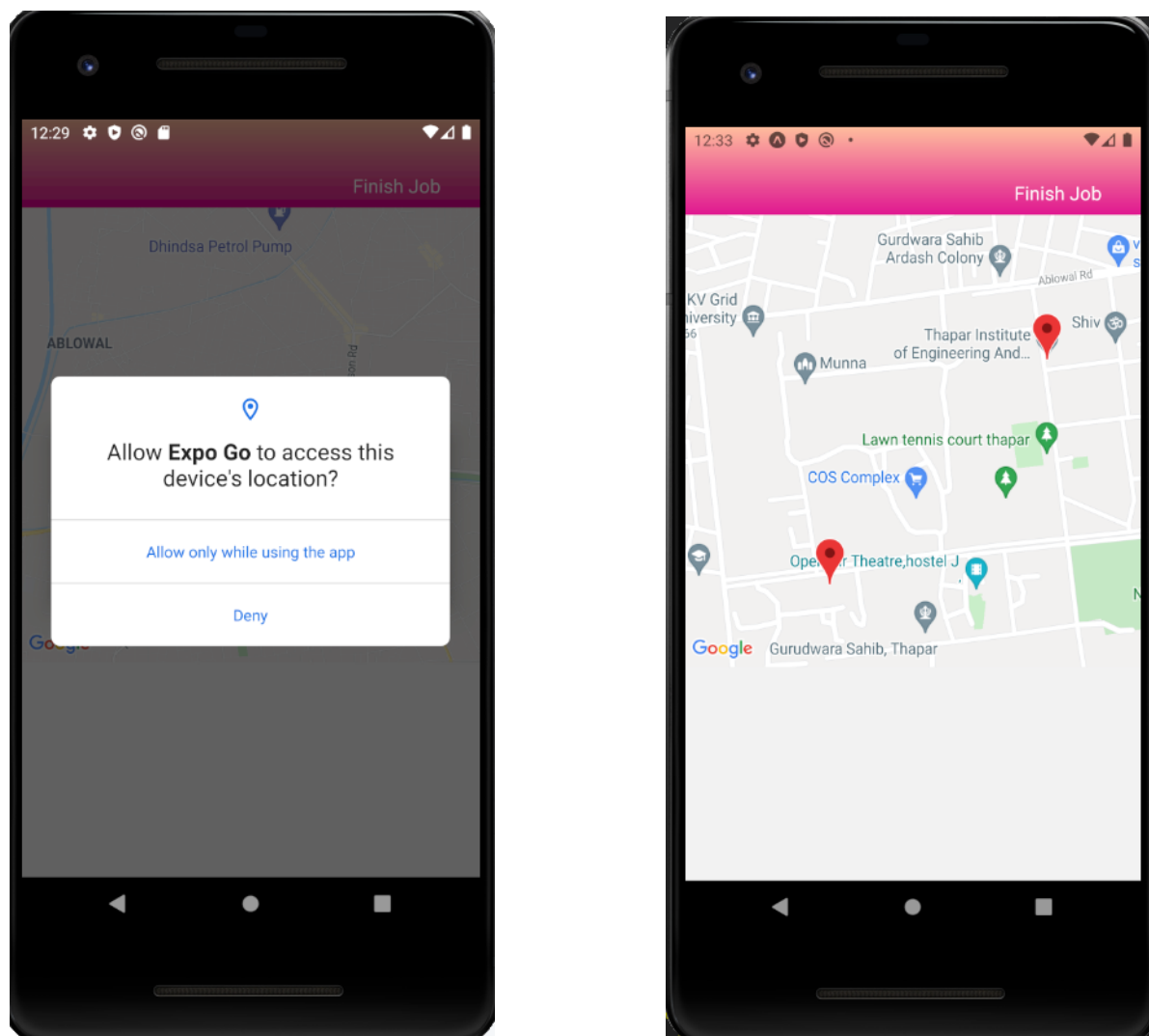


Figure 20,21

5. Testing

5.1 Test Plan

We performed the first level of testing i.e. unit testing and api testing. This testing is done to ensure that all the components of the software are functional and work according to the way they were designed for. We wrote the test cases and made them run. We conducted the unit testing manually. Unit testing makes debugging easier because finding issues earlier means they take less time to fix than if they were discovered later in the testing process.

Test environment:

OS: Android, iOS

Software: Android Emulator (Pixel 3), iOS emulator (iPhone 12)

Test Resources:

Continuous Integration: We plan to use GitHub Actions to check the quality of the code. It can run different types of tests and if all the tests succeed then only the code gets merged in the codebase. With this approach we can maintain code quality. Common tests that run on GitHub Actions are Linting checks where it makes sure the code written is according to the standardized style guide. If linting is not up to mark, then it automatically formats the documents. We also run unit tests, integration tests and api tests

Type of tests

- Unit Testing: Unit testing technique using which individual modules are tested to determine if there are any issues by the developer himself. It is concerned with functional correctness of the standalone modules. Python has a unit test library which will be used to test Django functions.
- Api testing: We plan to perform using firebase, so that we can get http code of api request and compare with all valid codes.
- Quality Assurance: We plan to perform tests according to test cases given below. These tests will be performed by developer and alpha, beta users and domain experts to ensure the software is practical in real world scenarios.

5.2 Test Cases

Test Case Template (Doc:T_01)

Test Case #: 1.1	Test Case Name : User Registration	Page: 1 of 2
System: Mobile	Subsystem: Login	
Designed by: Kunal	Design Date: 12/10/2021	
Executed by: Paritosh	Execution Date: 28/11/2021	
Short Description: Test system's User Registration service		

Pre-conditions-

The user is not registered.

Step	Action	Expected System Response	Pass/Fail	Comment
1	Click the Sign Up button.	System displays a form containing different fields required for registration.	Pass	Form displayed correctly
2	Enter email id as " abcd@gmail.com " Enter password as "daisy@1234" After filling all other details press the Register button.	System will load a new page which will display a message asking users to validate their account by clicking on the link sent on the registered email id. After clicking on the link, users will get registered.	Pass	Screen redirected successfully
	Check the post condition			
3	Repeat step 1 and then enter email as "abcd" and then fill all other details and then click the register button.	System asks to enter valid email id.	Pass	Systems shows correct error
4	Repeat step 1 and then enter email id as " abcd@gmail.com ". and then fill other details and then click the Register button.	System will load a new page which will display a message asking users to enter details. After entering details, users will get registered.	Pass	Screen redirected successfully
	Check the post condition			
5	Repeat step 1 and then enter email id as " abcd@gmail.com " and password as "ram" and after filling all the other details, click the register button	System pops an error stating that password is too common and short. The minimum length should be of 8 characters.	Pass	Systems shows correct error

6	Repeat step 1 and then enter email id as " abcd@gmail.com " and password as "daisy@1234" and after filling all the other details, click the register button.	System will load a new page which will display a message asking users to validate their account adding correct details.	Pass	Screen redirected successfully
---	--	---	------	--------------------------------

Post-conditions

User is registered successfully after clicking the link recieved on the registered email id.

Test Case Template (Doc:T_02)

Test Case #: 1.2	Test Case Name : Login	Page: 1 of 2
System: Mobile	Subsystem: Login	
Designed by: Agrima	Design Date: 12/10/2021	
Executed by: Paritosh	Execution Date: 28/11/2020	
Short Description: Test system's login functionality		

Pre-conditions-

- 1.The user has registered on the portal
- 2.The email for the current user is abcd@gmail.com .
3. The password is daisy@1234.

Step	Action	Expected System Response	Pass/Fail	Comment
1	Click the login tab	System displays a form with email and password fields.	Pass	Form displayed correctly
2	Enter the email ' abcd@gmail.com ' Enter the password 'daisy@1234' Click the login button	System directs the user to the home page for the patient.	Pass	Screen redirected successfully
	Check the post condition			
3	Repeat steps 1,2,3 using email ' abcd1@gmail.com ' and password 'daisy@1234' and click on the login button.	System displays an error message prompting to enter valid credentials.	Pass	Systems shows correct error
4	Re-enter the email-id ' abcd@gmail.com ' and click on the login button.	System directs the user to the home page.	Pass	Screen redirected successfully
	Check the post condition			
5	Repeat steps 1,2,3 using email ' abcd@gmail.com ' and password 'daisy@123' and click on the login button.	System displays an error message prompting to enter valid credentials.	Pass	Systems shows correct error
6	Re-enter the password 'daisy@1234' and click on the login button.	System directs the user to the home page .	Pass	Screen redirected successfully
	Check the post condition			
7	Repeat steps 1,2 with email ' abcdgmail.com '.	A prompt appears telling the user to enter an email address	Pass	Systems shows correct error
8	Re-enter the email id ' abcd@gmail.com ' and repeat steps 3 and 4.	System directs the user to the home.	Pass	Screen redirected successfully
	Check the post condition			

Post-conditions

The user is logged into the portal and thus, he/she can access the other functionalities depending on whether he/she is logged in as user or driver.

Test Case Template (Doc:T_03)

Test Case #: 2.1 System: Mobile Designed by: Dhruv Executed by: Paritosh Short Description: Test system's book ride functionality	Test Case Name : Add Ride Subsystem: Booking Design Date: 12/10/2021 Execution Date: 28/11/2021	Page: 1 of 1
--	--	---------------------

Pre-conditions-

1. The user is logged in with verified credentials.

Step	Action	Expected System Response	Pass/Fail	Comment
1	Click the 'Ride' button.	System displays a form with pickup, drop and number of pax fields.	Pass	System displays Form correctly
2	Fill All details and confirm booking	System redirects the user to the tracking page	Pass	System loads Screen correctly
	Check the post condition			

Post-conditions

The recently uploaded ride is added to the rides section and saved in the database.

Test Case Template (Doc:T_04)

Test Case #: 2.2	Test Case Name : Book D2D	Page: 1 of 1
System: Mobile	Subsystem: Booking	
Designed by: Kunal	Design Date: 12/10/2021	
Executed by: Paritosh	Execution Date: 28/11/2021	
Short Description: Test system's Book D2D service		

Pre-conditions-

The User is logged in

Step	Action	Expected System Response	Pass/Fail	Comment
1	Click the D2D Button.	System displays form and below with fields pickup, drop, name of item and quantity	Pass	System displays Form correctly
2	Fill Details and Confirm booking	System redirects user to tracking screen	Pass	System loads Screen correctly

Post-conditions

The D2D is uploaded to database

Test Case Template (Doc:T_05)

Test Case #: 3.1	Test Case Name : Tracking of job	Page: 1 of 1
System: Mobile	Subsystem: Tracking	
Designed by: Paritosh	Design Date: 12/10/2021	
Executed by: Kunal	Execution Date: 28/11/2021	
Short Description: Test system's track screen		

Pre-conditions-

The user is a valid user- The user logged in with verified credentials.

Step	Action	Expected System Response	Pass/ Fail	Comment
1	Navigate to track screen from bottom tab	System displays tracking screen	Pass	System Displays list correctly
2	View Cells	Driver details along with ETA and cost are displayed	Pass	System Displays details correctly

Post-conditions

User is able to see driver details and ETA with cost

Test Case Template (Doc:T_06)

Test Case #: 4.1	Test Case Name : Accept Job	Page: 1 of 1
System: Mobile		
Designed by: Agrima	Subsystem: Driver	
Executed by: Kunal	Design Date: 12/10/2021	
Short Description: Test system's ride screen	Execution Date: 29/11/2021	

Pre-conditions-

The Driver is logged in.

Step	Action	Expected System Response	Pass/ Fail	Comment
1	Go to home screen.	System displays a list of all the available jobs with details such as phone, destination etc.	Pass	System displays List correctly
2	Select one job from list	System loads modal view displaying fields of ETA and Confirmation.	Pass	Systems loads Page correctly
3	Enter details and accept job	System redirects to a new page	Pass	System redirects successfully

Post-conditions

The Driver is redirected to map screen.

Test Case Template (Doc:T_07)

Test Case #: 4.2

Test Case Name : View Map

Page: 1 of 2.

System: Mobile

Subsystem: Driver

Designed by: Dhruv

Design Date: 29/10/2021

Executed by: Paritosh

Execution Date: 1/12/2021

Short Description: Test system's ability to show Map

Pre-conditions-

The Driver is logged in and has accepted a job.

Step	Action	Expected System Response	Pass/ Fail	Comment
1	View the map and markers	System displays a map view .	Pass	Map displayed correctly
2	Click finish job	System will pop an alert showing "Job Completed Successfully".	Pass	System displays Alert correctly

Post-conditions

Driver is redirected to home screen

5.3 Test Report

S No.	Test Cases	Sub Levels	Responsible Tester	Date of execution	Pass/Fail
1.	Authentication Testing	Login User Registration	Kunal	28/11/2021	Pass
2.	User Booking	Add Ride Book D2D	Kunal	28/11/2021	Pass
3.	User Tracking	Tracking	Paritosh	28/11/2021	Pass
4.	Driver Interface Testing	Rides Screen Map Screen	Paritosh	29/11/2021	Pass

Executed		
	Passed	7
	Failed	0
	Total executed	7
Pending	0	
In Progress	0	
Blocked	0	

Test Planned		7
Total		7

Functions	Description	%TCs Executed	%TCc Passed	%TCs pending	Priority
Authentication Testing	Testing auth	100%	100%	0%	High
User Booking	Testing Booking	100%	100%	0%	High
User Tracking	Tracking orders	100%	100%	0%	High
Driver Interface Testing	Testing Ride Screen and Map Screen	100%	100%	0%	High