UniRide - Hub

Software Engineering Project Report

Amani Merhi	202304693
Jad El Moghrabi	202303028
Moamen Hamdan	202302491
Sandi ElZein	202301685

Table of Contents

Abstract

Our project revolves around developing a mobile application UniRide Hub that aims to revolutionize bus transportation for university students by enhancing campus transportation efficiency, student safety, and overall user experience.

Chapter 1. Introduction - Amani

A. Main Project Description

Transportation is an essential aspect of every university's student life, yet those less fortunate who use the university's bus transportation system often face a variety of challenges. The students must wait for extended periods on the sides of the roads to avoid the risk of missing the bus, and often should wave down the bus or rely on calling the driver to make sure they are picked up from the streets. Evidently, these methods pose a safety threat to the bus passengers and to the student waiting. And if the student is lucky enough to be picked up and steps onto the bus, he/she may not have a place to sit and will end up standing the whole ride; as the current system has no way of gathering insights on student transportation patterns, and peak usage times. Thus, more students are relying on their private vehicles to avoid the inconvenience of the more eco–friendly transportation option. Therefore, our team decided to tackle these problems by developing UniRide Hub, a user-friendly mobile application that provides students with real-time visibility into bus locations, estimated arrival times, and the ability to communicate with drivers seamlessly. Moreover, our app enhances passenger safety by eliminating the need for physical signaling and enabling location sharing between passengers and drivers which reduces waiting times and minimizing the likelihood of missed pickups. Furthermore, the app collects valuable data that can be utilized by the university to optimize bus schedules, by presenting the admin with necessary information such as bus usage patterns, popular routes, and peak usage times. Overall, our bus shuttle app aims to improve the whole campus transportation experience for students, which will promote sustainability and simultaneously highlight the institution's commitment to innovation and technological advancement.

B. Problem Statement

Our application aims to solve various problems faced by students who use the university's bus transportation system to get to and leave from the university. Primarily, the students often must guess when the bus will arrive while also waiting for extended periods of time in potentially unsafe environments, especially on the sides of the roads, to avoid the risk of missing it. Using UniRide Hub students gain real-time visibility into bus locations and estimated arrival times, which reduces wait times and minimizes the likelihood of missing the bus. Secondly, the unsafe communication between students and drivers, who respond to their calls while driving, is eliminated as our software offers a special map for bus drivers to view student locations and manage pick-ups efficiently. Thirdly, the current system used by the university has no process of gathering valuable data insights on student transportation habits to improve overall service quality. UniRide Hub tackles this issue by leveraging necessary data to optimize bus schedules, allocate resources efficiently, and organize more efficient route planning for future improvements. Lastly, more students are letting go of the existent bus transportation system and are now relying on individual car transportation which contributes to environmental pollution and traffic congestion on campus. Our app, UniRide Hub encourages the use of the university-provided eco-friendly transportation option that supports sustainability efforts. In brief, UniRide Hub is the golden savior of the current bus transportation system.

C. Project Goal

Our software aims to revolutionize bus transportation for university students by enhancing campus transportation efficiency, student safety, and overall user experience. Our goal is to provide university students with a reliable app that tackles several key challenges faced by students including long wait times, being overlooked by the bus-drivers, road safety concerns, and the right for a seat on the bus. UniRide Hub is intended to facilitate the lives of university students, faculty, and staff who rely on campus shuttle services for daily transportation. As, our user-friendly and feature-rich app ensures real-time access to bus locations and estimated arrival times, as well as the ability to book and unbook a seat on the bus and the privilege to add your current location to the driver's map. Encouraging the university's community to be part of a much more sustainable and eco-friendlier transportation approach.

D. System and Domain Review / Background

Existing systems related to campus transportation typically include traditional bus schedules that provide a general idea of bus timings but are often static and do not account for real-time delays or changes, while our app offers real-time bus tracking for students. Furthermore, students typically rely on manual methods such as phone calls or messages to inform the bus-drivers about their presence on the road. Therefore, we plan to offer a special map for bus drivers to view the locations added by the students. Also, existing bus transportation apps often fail to gather and analyze data on student transportation habits effectively, hindering the ability of universities to optimize bus schedules and allocate resources efficiently. While UniRide Hub can collect data on student pick-up locations, bus usage patterns, popular routes, and peak usage times. Ensuring that no student is left stranded on the streets or without a seat on the bus. Not to forget, our app's exclusive features include personalized notifications for both the drivers and students, and student bus requests. The app can send personalized notifications to students, such as reminders when the bus takes off and in case of any delays; moreover, the app alerts the driver once he approaches a student nearby on the road to avoid missed pick-ups. Also, the software allows the users to request a bus and depending on the number of requests, a bus trip can be scheduled. On the other hand, alternatives such as private transportation services (taxis) and personal vehicles exist and although they may offer more flexibility and convenience, they can be expensive for students, especially those on tight budgets. Moreover, using these alternatives contradicts sustainability efforts and contributes to traffic congestion on campus.

Chapter 2. Project Plan - Jad/Moamen/Sandy

A. SDLC Model - Sandy

A software development life cycle (SDLC) model helps divide projects and properly assign tasks, while also assisting in managing risks by correctly planning the project ahead of time. To implement our application, we will use the waterfall model as an SDLC model; where each phase should be completed fully before we can begin with the next phase. There are multiple reasons why we decided upon the waterfall model as the best suit for our project. Firstly, the requirements of our system are well-defined and stable. Secondly, our project's milestones are well understood as we have planned all the phases. Thirdly, we have set a great management plan and figured out all the technologies that will be needed. Furthermore, our initial deployment of the app would have all the major features, and then we can build a-top the stretch features and additional user functionalities that are customized based on our client's preferences, since each university has its own system and regulations. Thus, while working on delivering the final product we plan to finish each phase fully before going onto the very next phase, exactly how the waterfall model works. Finally, our chosen model also offers multiple advantages including predictability, efficiency, and clearly defined stages.

B. Project Organization - Sandy

1. Amani (Project Lead and Content Specialist):
   a) Introduction Section (Chapter 1):
      - Main Project Description
      - Problem Statement
      - Project Goal
      - System and Domain Review / Background:

b) Software Requirement Specification (Chapter 3):
   - Product Functions
   - User characteristics
   - Use case description

c) Methodology (Chapter 5):
   - Testing

2. Sandy (Design and Documentation Lead):
   a) Project Plan Section (Chapter 2):
      - SDLC Model
      - Project Organization
      - Ethical Standards and Guidelines

   b) Project Design (Chapter 4):
      - User Interface Prototype
      - Database Diagram: EER Diagram
      - Class Diagram: UML Class Diagram.

3. Jad (Technical Lead):
a) Project Plan Section (Chapter 2): - Feasibility Study: Risk Management, Technical Feasibility, Economic Feasibility, and the Delivery of the Application.
b) Software Requirement Specification (Chapter 3): - Domain Requirements
c) Project Design (chapter 4)
   - Data Flow Diagram
   - Sequence Diagram

4. Moamen (Development Lead):
A) Project Plan Section (Chapter 2): - Schedule/Timeline:
B) Methodology (chapter 5):
   - Implementation:
   - Maintenance.

5. Team Effort:
   - Non-functional Requirements
   - functional Requirements
   - Use Case Diagrams

C. Ethical standards and guidelines - Sandy

At UniRide Hub, the students' well-being will always be the main concern of ours. This is something we believe is important in the whole business and will be doing accountability in all aspects of our business. Also, the organization should not only follow user privacy policies but get confirmation from users to know their location. Moreover, the transparency of data is an achievement being that any user must have the assurance that his/her data are only collected from specific sources and has the freedom to manage his/her data sharing process. Also, we must not forget the genre inclusiveness: we offer language widely, and we plan on testing with people characterized by different user types. In security and safety, the issues should be anticipated through the implementation of control measures. Regardless of their socioeconomic status, everyone should have fair access to our system. Policy can range from data security, collection, and users to privacy and, if any modifications are required, it will be communicated. Additionally, the ways of applying the tools are covered from the time they are installed on the computers till when data can be accessed by the users. Evidently, by adhering to these ethical principles and guidelines, the UniRide Hub project aims to not only address the transportation challenges faced by university students but also prioritize user privacy, safety, inclusivity, fairness, and empowerment throughout the development and deployment process.

D. Schedule/Timeline -Moamen

1. **The preliminary requirements and planning Getting together:**
   - Describe the project's scope:
   - Hold stakeholder meetings to clarify the project's goals, parameters, and deliverables.
   - Draft a project charter that outlines the objectives, stakeholders, and purpose of the project.
   -Acquire Necessities:
   - Gather functional and non-functional requirements by conducting workshops, questionnaires, and interviews with stakeholders.
   - Use methods like use cases, user stories, and requirements traceability matrices to document requirements.
   -Determine the First Dangers:
   -Make a list of all the possible risks and uncertainties that can affect the project's outcome.
   - Sort hazards into priority groups according to likelihood and influence on project goals.
   - Deliverables for Document Planning:
   - Create a project plan that includes duties, deadlines, materials, and dependencies.
   - To provide a baseline for development activities, create a requirements document.

Timeline: 1 – 2 months

2. Architecture and Design:

- System design:

- Specify the mobile application's general design, including its layers, components, and communication protocols.
- To enable the required functionality and scalability, use the right technologies and frameworks.

- User Interface Design:

- To envision the user interface design, create wireframes, mockups, and prototypes.
- To ensure usability and user happiness, get input from stakeholders and iterate on designs.

- Database Schema Design:

- Create a database schema that will effectively store and retrieve the data that the application needs

- Consider elements like performance optimization, normalization, and data integrity.

Timeline: 2 – 4 weeks

3. Implementation:

-Frontend Development:

- Develop frontend components of the mobile application using Flutter framework and Dart programming language.

- Implement navigation, layout, and user interaction patterns to create a seamless user experience.

- Backend Development:

- Set up backend infrastructure using MySQL database and develop server-side logic using appropriate programming languages (e.g., Node.js, Python).

- Implement APIs to enable communication between frontend and backend components of the application.

- Integration of Plugins:

- Integrate third-party plugins and libraries to enhance the functionality of the application, such as geolocation services, push notifications, and authentication mechanisms.

- Test integrations to ensure compatibility and reliability.

Timeline: 3 – 4 months

4. **Testing and Quality Assurance:**

- Unit Testing:

- Write unit tests to validate individual components and functions of the application.

- Use testing frameworks such as JUnit (for Java) to automate testing processes.

- Integration Testing:

- Test the interaction between frontend and backend components of the application to ensure seamless integration and data flow.

- Conduct end-to-end testing to verify the application's behavior across different devices and platforms.

- User Acceptance Testing:

- Involve stakeholders and end-users to test the application against predefined acceptance criteria.

- Gather feedback and make necessary adjustments to address any issues identified during testing.

Timeline: 1 – 2 months

5. **Marketing and Deployment Preparation:**

- Marketing Strategy Development:

- Define target audience, positioning, and messaging for the marketing campaign.

- Identify key marketing channels and tactics to reach potential users, such as social media, email marketing, and influencer partnerships.

- App Store Preparation:

- Prepare assets and metadata required for app store submission, including app description, screenshots, and promotional materials.

- Ensure compliance with app store guidelines and requirements to facilitate smooth approval process.

- User Guide Creation:

- Develop comprehensive user guides and documentation to help users understand how to use the application effectively.

- Provide tutorials, FAQs, and troubleshooting tips to address common user queries and issues.

Timeline: 1 month prior to completion

6-Deployment and Maintenance:

-App Deployment:

-Deploy the application on app stores, such as Google Play Store and Apple App Store, following approval from app store review process.

-Monitor app performance and user feedback to identify areas for improvement and optimization.

-Maintenance Procedures Setup:

-Establish procedures for monitoring application performance, handling user feedback, and releasing updates.

-Implement bug tracking systems and version control mechanisms to streamline maintenance activities.

-User Support:

-Provide ongoing support to users through various channels, such as email, chat, and forums.

-Address user inquiries, troubleshoot issues, and escalate unresolved issues to development team for resolution.

Timeline: After completion

7-Ongoing Support and Updates:

-Bug Fixes and Enhancements:

-Continuously monitor the application for bugs and performance issues reported by users.

-Release regular updates to address user feedback and add new features or enhancements.

-User Engagement:

-Engage with users through social media, newsletters, and in-app notifications to keep them informed about updates and encourage usage.

-Encourage user feedback and reviews to foster a sense of community and loyalty.

-Feedback Collection:

-Solicit feedback from users through surveys, ratings, and reviews.

-Use feedback to prioritize future development efforts and improve user satisfaction and retention.

Timeline: Continuous

E. Feasibility study - Jad

Risk Management

Risk management is critical for our project, particularly considering our adoption of the Waterfall model. Especially because the waterfall model has a high amount of risk. We try to prevent these problems by preparing and recognizing potential risks. First, we fear that our software will not be accepted by universities and companies, because although it has a lot of features and is user-friendly, some companies prefer sticking to their existing system. After all, it's easier and they are used to it. Secondly, after the pandemic the Lebanese economy declined, and some companies might not have extra cash in their budget to afford our software. Third, someone's job is at risk, since our software can plan trips and automate multiple functions this eliminates the need for multiple supervisors and only one manager is enough. Fourth, since we are still 2nd year students, we fear that we will not have the expertise to implement this project since we have no experience and are still not familiar with some APIs and tools.

Finally, to minimize any upcoming risks we will pitch our idea to multiple bus system managers so they can guide us and help us with their knowledge, if we believe that we lack the necessary expertise, we will seek professional assistance for guidance. We take risk management extremely seriously in our development process, so, by following these procedures, we will not run into issues down the road.

Technical feasibility:

We think that our software is technically feasible since software with some of our features already exists in the market and all the added features are technically feasible and can be implemented using existing software and programs. Initially, to have a functional application we need to use Flutter to achieve a cross-platform app, Flutter is a cross-platform development tool powered by Google. Based on the Dart programming language, it offers fast and fluid performance thanks to its integrated widgets. In addition, we will be using Visual Studio Code as our IDE for Flutter because it is a lightweight, free code editor with great Flutter compatibility.

Also, we need a database to hold data as using our software requires users to be able to create accounts. We are going to utilize MySQL for that.

For our app to function correctly, it requires location access; to achieve this, we will utilize the "geolocator" Flutter plugin. This plugin allows us to inform the driver of the user's location to guarantee a safe pickup. It will also be used to calculate the distance between the user and the driver to improve communication and the user experience. Additionally, we will employ a version control system called "Git" to enable remote work among all team members.

To have an easier time managing the project we will be using "Trello", Trello is a project management tool that helps teams coordinate by using boards, lists, and cards to distribute work, manage deadlines, measure progress, and organize tasks. Lastly, we will be using Microsoft Word to list ideas and functionalities in addition to Canva which will help in creating a neat and user-friendly interface for our application.

Economic Feasibility:

According to our estimations building UniRide Hub will take approximately 9 months,

The use of the Waterfall model helps to deliver a functioning product that is fully functional and bug-free on delivery, so we will be focusing on analysis during the first 2 months to make sure of every step since the waterfall model does not work well with changing requirements, the time is divided as follows:

a. Requirements and analysis:1-2 months
b. Design: 2-4 weeks
c. Implementation: 3-4 months
d. Testing: 1-2 months

According to these calculations, we need a maximum of 38 weeks (about 8 and a half months) to complete the project and be ready to be sold.

Our application will not cost a lot of money to build. Still, since our development team consists of 4 2nd year students with no experience, we will have to allocate some money to their training in addition to some software tools and some other fees listed below:

- Dev Training: $388
- Marketing: $600
- Graphic design: $200
- Tutorial: $200
- Software tools: $97/month

We will need $1661 with an additional $600 for marketing to get this project implemented in 38 weeks (about 8 and a half months).

Note: this budget is only for the base software and not for the final product.

> The project will be sold as is for $2700, any extra features will cost extra and will increase the price accordingly

Delivery:

After finalizing the project, we will start our marketing campaign to try to attract the greatest number of universities and companies possible and make sure to deliver the software's pros in an eye-catching way.
After we sign an agreement with a university and based on the extra features that they would like to add they will be charged extra (between $2700 and $3500) and the delivery date will move accordingly, ideally between 1-2 months.
This agreement will also contain the maintenance fees, which will be $500, and the length of the contract, which can be renewed if both parties agree on the same terms.
After completing the final software we will post a user guide for the users and will buy a secure physical database to link to our software and make sure it's placed in a secure location in the main building of the company/university and will deploy our app on the play store and pay a monthly fee of $20 per month to be able to deploy the app on the apple store.

Chapter 3. Software Requirement Specification - Amani / Jad

A. Product Functions - Amani

UniRide Hub offers a variety of unique features tailored to facilitate the student's transportation experience to the university. Firstly, our app provides students with real-time visibility into bus locations, estimated arrival times, and the ability to communicate with drivers seamlessly through a special map. While the users can use this map to track the bus, the drivers can use a similar map to view student locations and manage pick-ups efficiently. Secondly, the students can guarantee themselves a spot to sit on the bus by booking a seat beforehand. Moreover, this procedure ensures that the students are saving money by paying per ride and not for the whole semester, as when the student books a seat the system deducts the seat price from the user's account. Not forgetting that the student can unbook a ticket and get a full or partial refund depending on the bus departure time. Thirdly, the students can request a bus by entering the date and time of the requested trip and if the number of requests is adequate, a trip will be added by the admin to the schedule. Fourthly, although the current system used by the university has no process of gathering valuable data insights on student transportation habits to improve overall service quality. UniRide Hub tackles this issue by leveraging necessary data that is presented to the admin such as bus usage patterns, popular routes, and peak usage times, average trip times, student pick-up locations. In addition, the first of the two stretch features that we plan to implement later is to provide the user with personalized notifications, such as reminders of when the bus takes off and alerts in case of delays. Also, alerts are sent to the driver once he approaches a student to help manage pick-ups efficiently. While the second one is to allow connections among students who can send a friend request, and if the added friend accepts the request, they become friends and can see each other's booked bus seats.

B. User characteristics - Amani

The users of our bus shuttle app primarily consist of university students, faculty, and staff seeking efficient and safe transportation solutions to their campus. Unlike other transportation apps, our system does not require any specialized training or prior experience in navigating similar platforms. A quick tutorial at the outset highlights the key functionalities of the application, ensuring ease of use for all students. However, it is essential for users to have basic familiarity with smartphone technology and mobile applications. Yet, this will not be a problem as most if not all students and staff nowadays are comfortable with mobile apps and devices in general. Also, our user-friendly and intuitive app interface will primarily be in the English language as it is the language commonly used by the university's community.

C. Non-functional Requirements - Team Effort

Security	R1: User data, including personal and payment information, should be encrypted, and stored securely in our database, to secure it from unauthorized access and data breaches
Scalability	**R2:** Without compromising performance or user experience, our app should be able to scale up to accommodate our growing user base and increasing interactions.
Reliability	**R3:** UniRide Hub should have a prominent level of reliability ensuring that it does not crash frequently and operates smoothly and flawlessly.
Performance	**R4:** For outstanding performance, our system should be able to quickly respond to user interactions with minimal latency in the processing of data.
Maintainability	**R5:** Our app's code infrastructure should be well-structured and well-documented to facilitate future maintenance and updates.

D. Domain Requirements - Jad

Work Email	R1: User must have a work provided email to be able to create an account.
Real-time Bus Tracking	**R2:** Users can see the buses' current location on the map in real time.
Location Sharing	**R3:** Users should share their current location on the driver's special map to facilitate pick-up.
Seat Reservation	**R4:** Our app should allow our users to book a seat in advance for them to be guaranteed a spot.
Email Verification	**R5:** Send the User an Email to their work-provided email to verify their account
User Notifications	**R6:** Send The user an email when their account is verified
Security	**R7:** Send users an email each time they sign in on a new device

E. Functional Requirements – Amani / Team Effort

1. Requirement Description - Team Effort

Type	Functional **Requirements**
Registration	**R1:** Students should be able to create accounts, providing their name, university email, and a password.
	R2: Drivers should be able to create accounts, providing their name, phone number, and a password.
	R3: Students and drivers should be able to login once they register and access their account
Booking Seat	**R4: The** user should be able to book a seat
	R5: The user should be able to unbook a ticket and get a full or partial refund depending on the bus departure time.
Bus Scheduling	**R6**: The software allows the driver to add a trip including details such as date, time, number of seats and price of seat.
	R7: the admin shall have access to all scheduled trips and can edit, delete and view each trip.
	R8: The software allows the users to request a bus and depending on the number of requests a bus trip will be scheduled
Bus Tracking	**R9:** The user will be able to see the exact location of the bus on the map and how far away it is.
	R10: The app offers a special map for bus drivers to view student locations and manage pick-ups efficiently.
Personalized Notifications	**R11:** The app can send personalized notifications to students, such as reminders when the bus is approaching their location and about delays.
	R12: The system will send an alert once the driver approaches a student nearby to avoid missed pick-ups.
Check-In	**R13:** Facial recognition technology allows registered students to check-in when they board the bus. This ensures that only authorized students are using the service.
Payment	**R14:** When booking a seat occurs, the cost will be deducted from the student's account
Data	**R15:** The app can collect data on student pick-up locations and bus usage patterns, allowing for more efficient route planning for future improvements
Connection	**R16:** Users can connect with their friends
	R17: Friends can see each other's booked seats
Privacy	**R18:** Allow users to disable the above features
Stopping	**R19:** The user can request using the app stopping the bus

2. *Use Case Diagrams: - Team Effort*

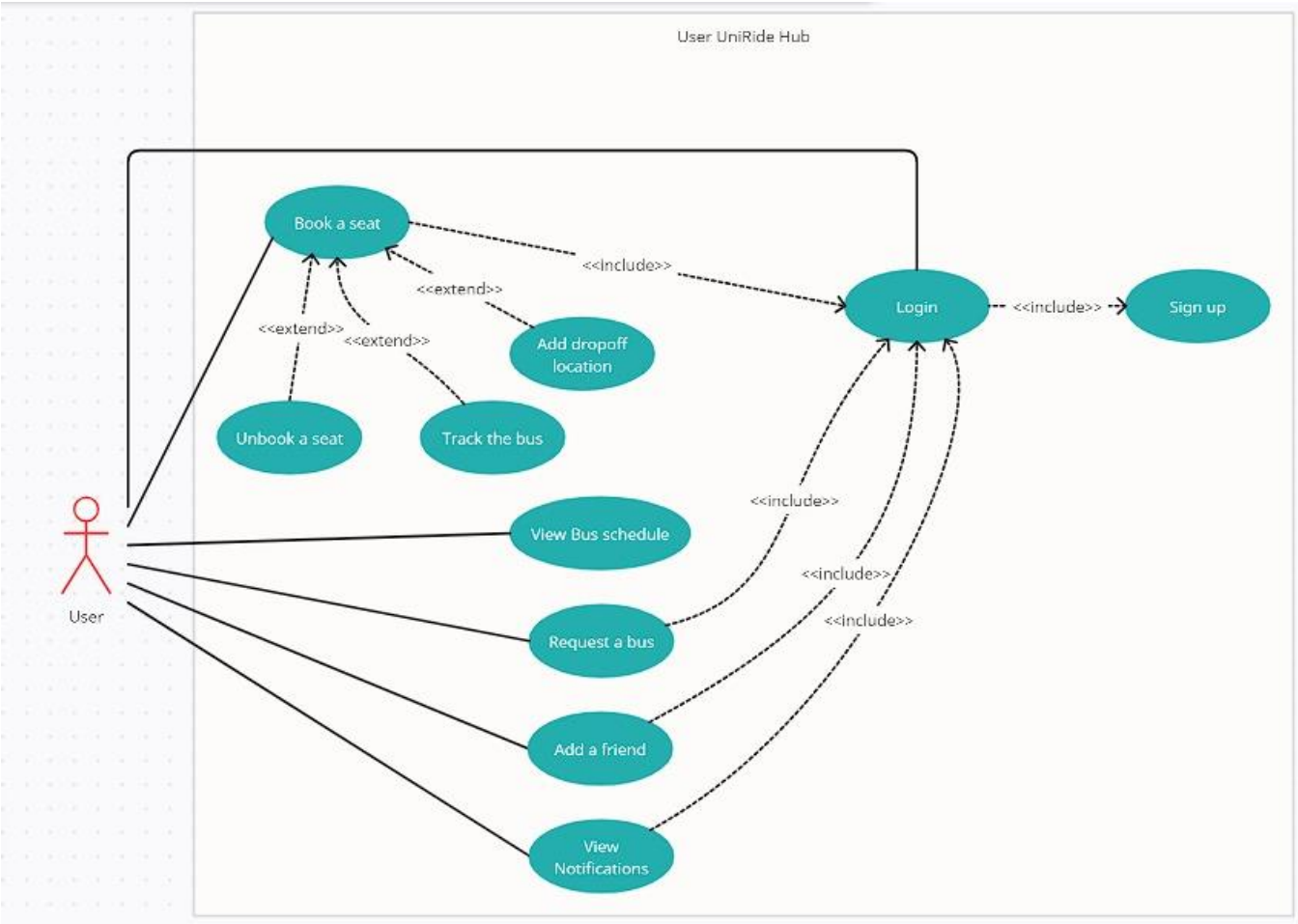


Diagram 1: Use Case Diagram for the User

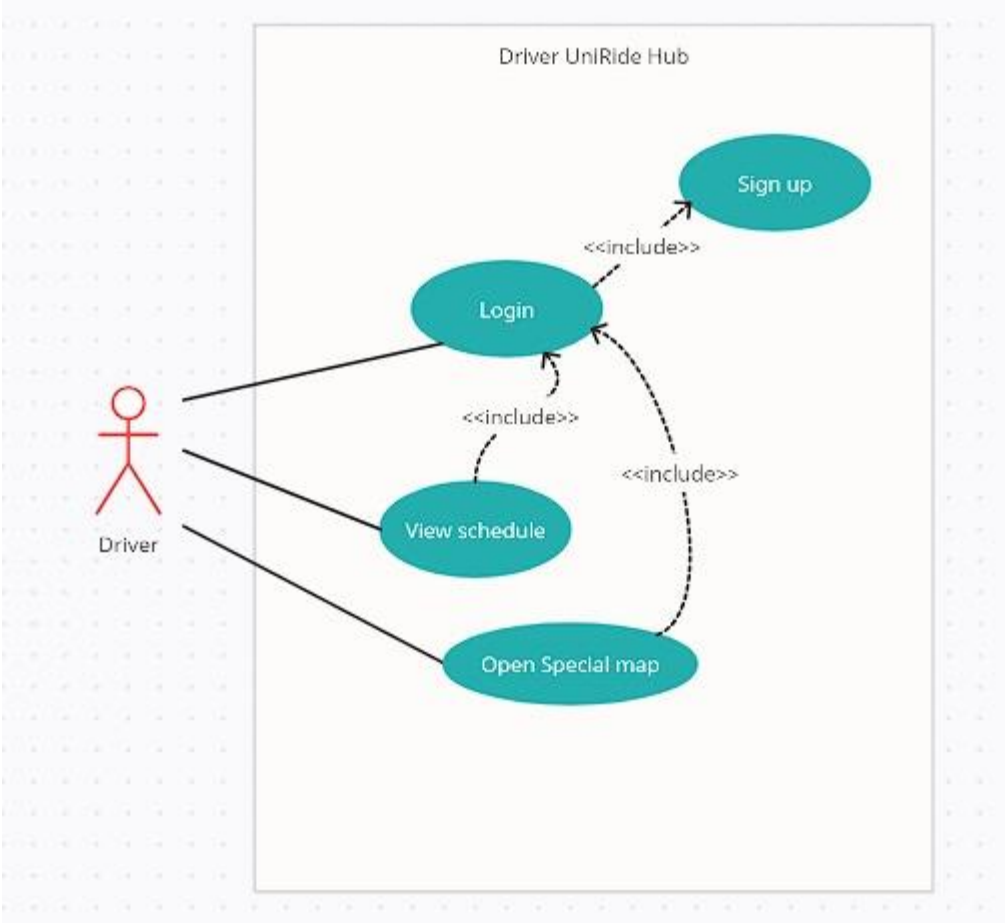


Diagram 2: Use Case Diagram for the Driver

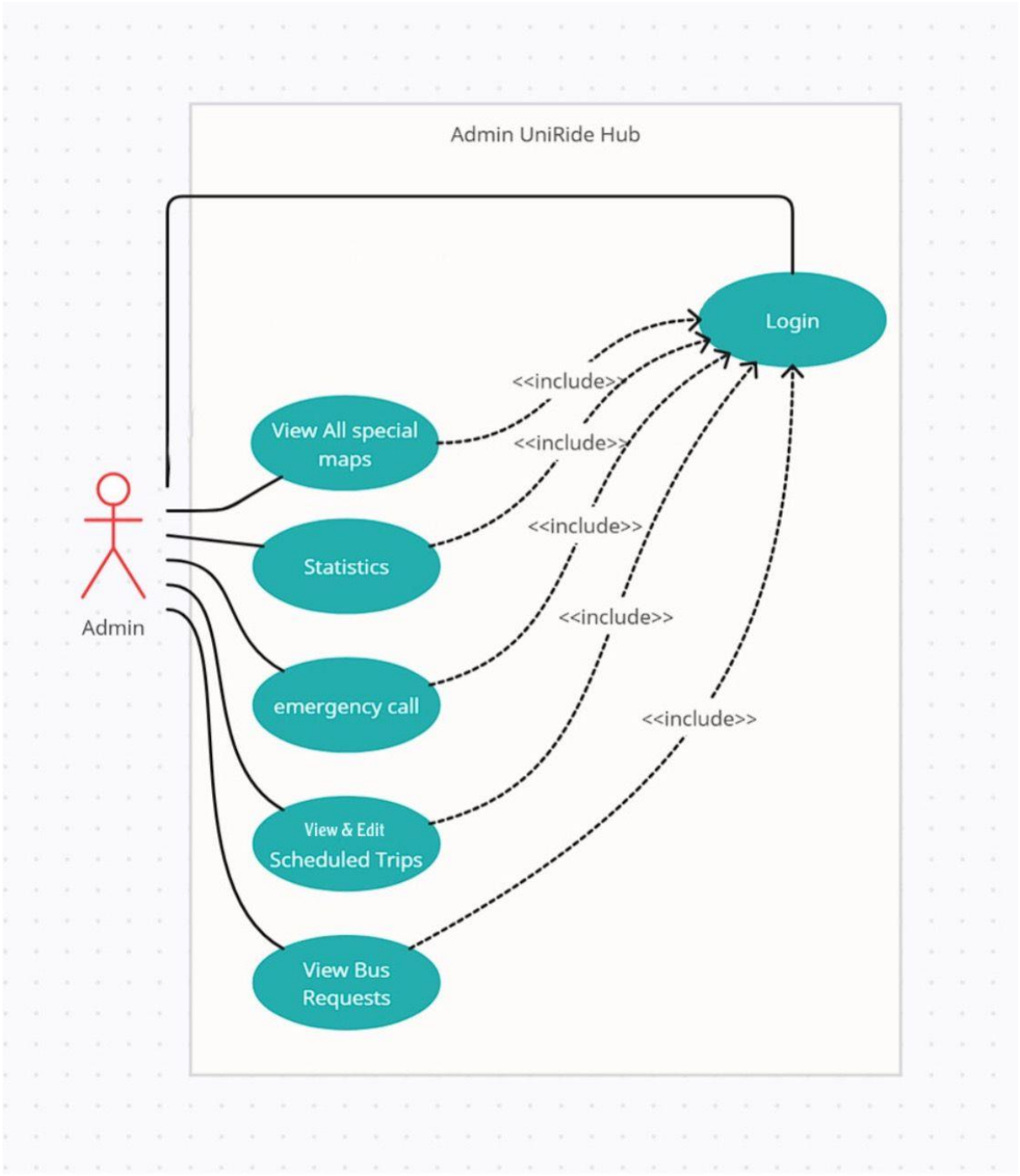


Diagram 3: Use Case Diagram for the Admin

3. *Use case description – Amani*

A. *User Use Case Diagram Description*

1-Use Case Name: Login

Actor: User

Pre-Conditions: The user should be registered

Related Use Cases: Includes Sign-Up

Steps:

Actor Actions	System Responses
1. Press on Login Button **2.** Enter Required Credentials	0. Returns if login is successful or not

Post-Conditions: The user will be able to use all app features

2- Use Case Name: Sign Up

Actor: User

Pre-Conditions: No pre-conditions

Related Use Cases: Included by Login

Steps:

Actor Actions	System Responses
1. Press on Sign Up Button 2. Enter Required Credentials	0. Returns if sign up is successful or not

Post-Conditions: The user will be able to login and use all app features

3-Use Case Name: Book a Seat

Actor: User

Pre-Conditions: User should be Logged In, User should have sufficient funds in their account, Seats are available for booking

Related Use Cases: Includes Login, Extends Unbook a Seat, Track the Bus, Add Dropoff Location

Steps:

Actor Actions	System Responses
1. Press on Book a Seat Button 2. Click on Choose Bus field 3. Enter the date, time, and city 4. Choose a bus from the list 5. Choose the preferred seat 6. Enter the pick-up location	0. Prompts the user to enter the date, time, and city of the trip 1. Will mark the seat as booked 2. Adds the user's pick-up location to the driver's special map 3. Deducts the seat price from the user's account 4. Sends to the user a confirmation notification indicating successful booking

Post-Conditions: The user will have booked a seat which he/she can un-book. Also, they would have added their pick-up location to the driver's map

4- Use Case Name: Un-Book a Seat

Actor: User

Pre-Conditions: User should be Logged In, User should have booked a seat

Related Use Cases: Extended by Book a Seat

Steps:

Actor Actions	System Responses
1. Press on the booked seat 2. Select un-book seat	1. Will mark the seat as un-booked 2. Will give the user a full or partial refund based on the bus's departure time

Post-Conditions: The user will have un-booked a seat and he/she can book another seat

5- Use Case Name: Track the Bus

Actor: User

Pre-Conditions: User should be Logged In, User should have booked a seat

Related Use Cases: Extended by Book a Seat

Steps:

Actor Actions	System Responses

1. Press on Bus Tracking Button	1. Displays a map with the bus's location and estimated arrival time

Post-Conditions: The user will be able to track the bus's location in real time

6- Use Case Name: Add Dropoff Location

Actor: User

Pre-Conditions: User should be Logged In, User should have booked a seat

Related Use Cases: Extended by Book a Seat

Steps:

Actor Actions	System Responses
1. Press on Add Drop-off Location Button	1. Displays the road's map where the user can add the drop-off location 2. Adds the drop-off location on the driver's special map

Post-Conditions: The user will be able to add his/her drop-off location

7- Use Case Name: View Bus Schedule

Actor: User

Pre-Conditions: No pre-conditions

Related Use Cases: None

Steps:

Actor Actions	System Responses
1. Press on Bus Schedule Button	1. Displays the bus's schedule for all days of the week

Post-Conditions: The user will be able to see all bus's schedules for the whole week

8-Use Case Name: Request A Bus

Actor: User

Pre-Conditions: User should be Logged In

Related Use Cases: Includes Login

Steps:

Actor Actions	System Responses
1. Press on Bus Request Button **2.** Enters the date, time, and destination	1. Prompts the user to enter the date, time, and destination of the requested trip 2. The system records the user's request for a bus trip

Post-Conditions: If the number of requests is adequate, a trip will be added by the admin to the schedule

9-Use Case Name: Add a Friend

Actor: User

Pre-Conditions: User should be Logged In

Related Use Cases: Includes Login

Steps:

Actor Actions	System Responses
1. Press on Add Friend Button 2. Click on Search for a Friend Field 3. Enter the friend's username 4. Click on Send a Friend Request	1. Sends a friend request to the selected friend

Post-Conditions: If the added friend accepts the friend request, they become friends and can see each other's booked bus seats

10- Use Case Name: View Notifications

Actor: User

Pre-Conditions: User should be Logged In

Related Use Cases: Includes Login

Steps:

Actor Actions	System Responses
1. Press on Alerts Button	1. Displays the received notifications

Post-Conditions: The user will be able to see personalized notifications, such as reminders of when the bus takes off and alerts in case of delays.

B. Driver Use Case Diagram Description

11- Use Case Name: Login

Actor: Driver

Pre-Conditions: Driver should be registered

Related Use Cases: Includes Sign Up

Steps:

Actor Actions	System Responses
1. Press on Login Button **2.** Enter Required Credentials	1. Returns if login is successful or not

Post-Conditions: The driver will be able to use all app features

12-Use Case Name: Sign Up

Actor: Driver

Pre-Conditions: No pre-conditions

Related Use Cases: Included by Login

Steps:

Actor Actions	System Responses
1. Press on Sign Up Button 2. Enter Required Credentials	1. Returns if sign up is successful or not

Post-Conditions: The driver will be able to login and use all app features

13- Use Case Name: View Schedule

Actor: Driver

Pre-Conditions: Driver should be logged in

Related Use Cases: Included by Login

Steps:

Actor Actions	System Responses
1. Press on My Schedule Button	1. Displays the driver's schedule for all days of the week

Post-Conditions: The driver will be able to see all his scheduled trips for the whole week

14- Use Case Name: Open Special Map

Actor: Driver

Pre-Conditions: Driver should be Logged In

Related Use Cases: Included by Login

Steps:

Actor Actions	System Responses
1. Press on Special Map Button **2.** Can Press on Emergency Call Button	1. Displays a map with all the locations added by the students waiting on the road 2. Sends an alert once the driver approaches a student nearby 3. If driver presses on emergency call button the admin will be called immediately

Post-Conditions: The driver will be able to view student locations and manage pick-ups efficiently with the help of the alerts sent by the system once he approaches a student. And in case of emergencies, the driver can call the admin with the press of a button!

C. Admin Use Case Diagram Description

15. Use Case Name: Login

Actor: Admin

Pre-Conditions: No pre-conditions

Related Use Cases: Includes by all other admin use cases

Steps:

Actor Actions	System Responses
1. Press on Login Button **2.** Enter Required Credentials	1. Returns if login is successful or not

Post-Conditions: The admin will be able to use all app features

16. Use Case Name: View & Edit Scheduled Trips

Actor: Admin

Pre-Conditions: Admin should be logged in

Related Use Cases: Included by Login

Steps:

Actor Actions	System Responses
1. Press on View & Edit Scheduled Trips Button	1. Displays all the scheduled trips with their corresponding drivers for all days of the week 2. Enables the admin to alter and update the drivers' schedules for all days of the week

Post-Conditions: The admin view all the scheduled trips with their corresponding drivers for the whole week and will be able to change and update all the drivers' scheduled trips for the whole week

17. Use Case Name: View All Special Maps

Actor: Admin

Pre-Conditions: Admin should be Logged In

Related Use Cases: Included by Login

Steps:

Actor Actions	System Responses
1. Press on View Special Maps Button	1. Displays all the driver's maps with all the locations added by the students waiting on the road

Post-Conditions: The admin will be able to view the location of all buses as well as all the student locations

18. Use Case Name: View Bus Requests

Actor: Admin

Pre-Conditions: Admin should be Logged In

Related Use Cases: Includes Login

Steps:

Actor Actions	System Responses
1. Press on View Bus Requests Button	1. Displays all demands made by the users for requested trips

Post-Conditions: The admin can see all the demands made by the users for a certain trip and if the number of requests is adequate, a trip will be added by the admin to the schedule

19. Use Case Name: View Statistics

Actor: Admin

Pre-Conditions: Admin should be Logged In

Related Use Cases: Includes Login

Steps:

Actor Actions	System Responses
1. Press on View Statistics Button	1. Displays all the bus trips taken along with all the trips' details

Post-Conditions: The admin can see all the data collected by the application such as bus usage patterns, popular routes, and peak usage times, average trip times, student pick-up locations.

20. Use Case Name: Use Emergency Call

Actor: Admin

Pre-Conditions: Admin should be Logged In

Related Use Cases: Includes Login

Steps:

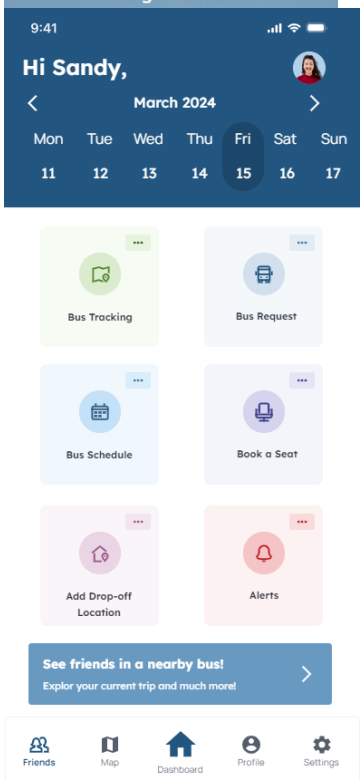
Actor Actions	System Responses
1. Press on Emergency Call Button	1. Displays all drivers currently on the road who the admin can call

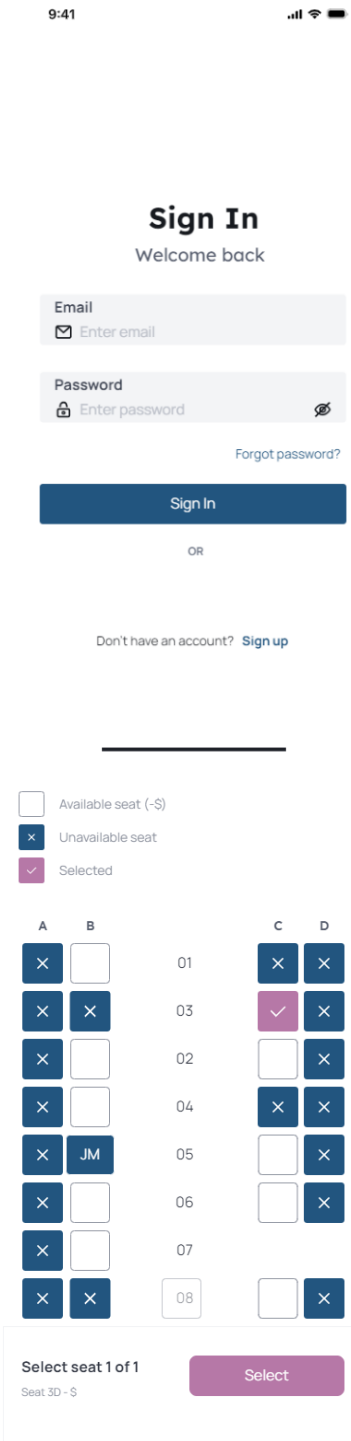
Post-Conditions: The admin can see all the drivers currently on the road and can call any of them in case of an emergency

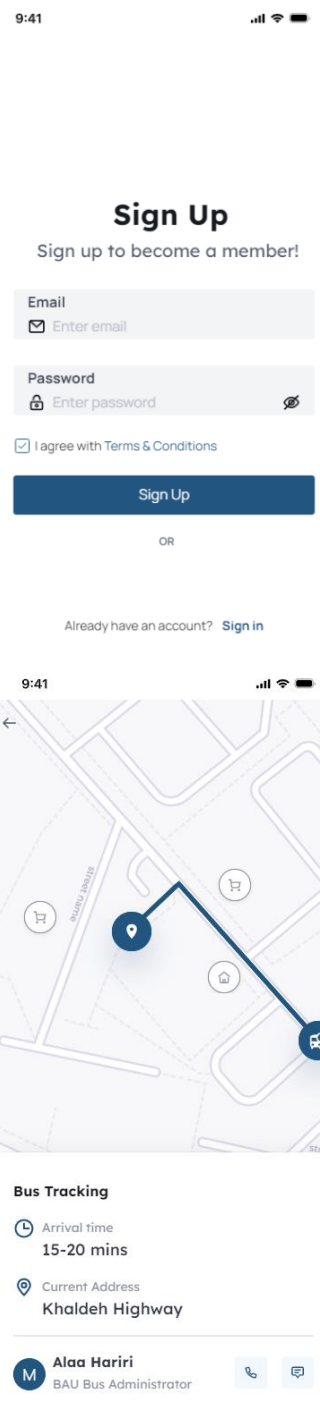
Chapter 4: Project Design – Sandy / Jad

A. Prototype - Sandy

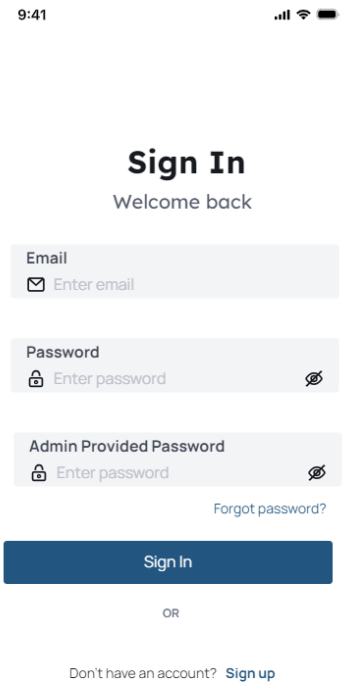
User Interface

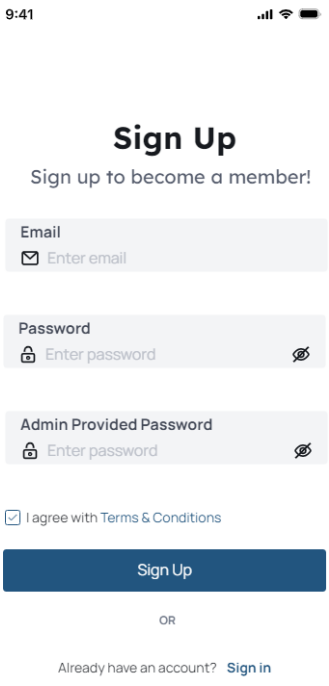


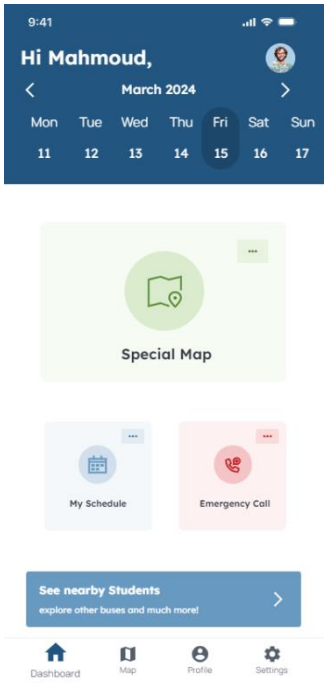


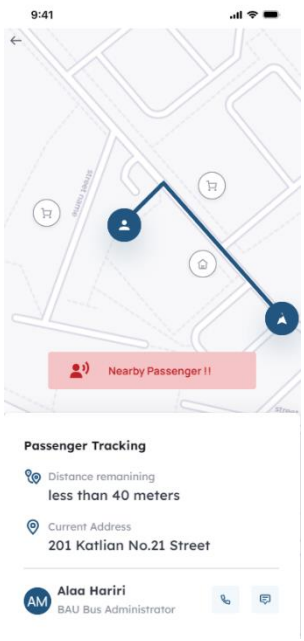


Driver Interface

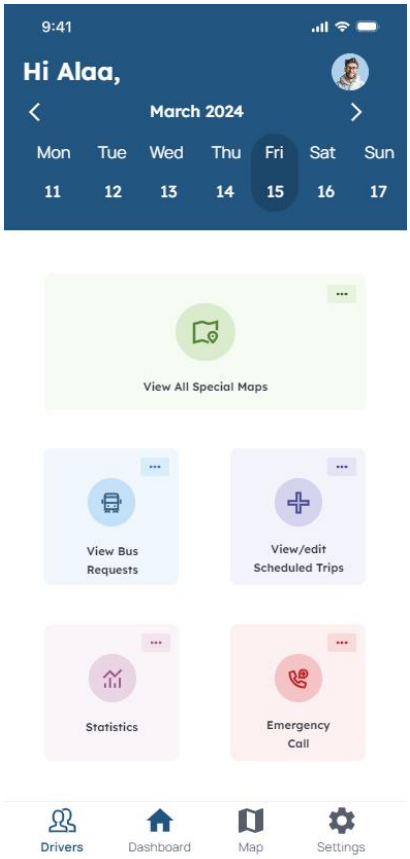


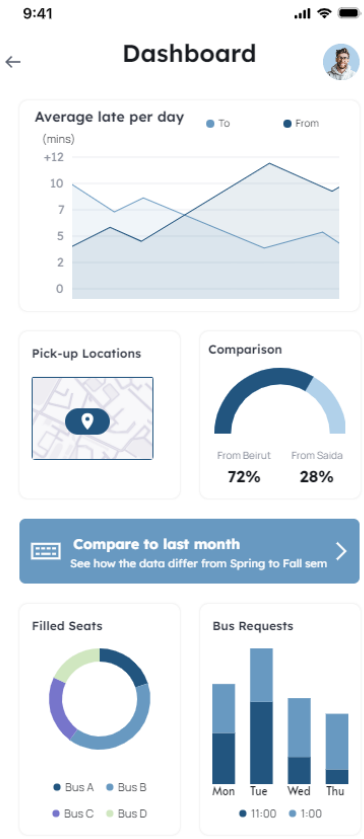






Admin Interface





B. Data Flow Diagram - Jad/Moamen

Level 0

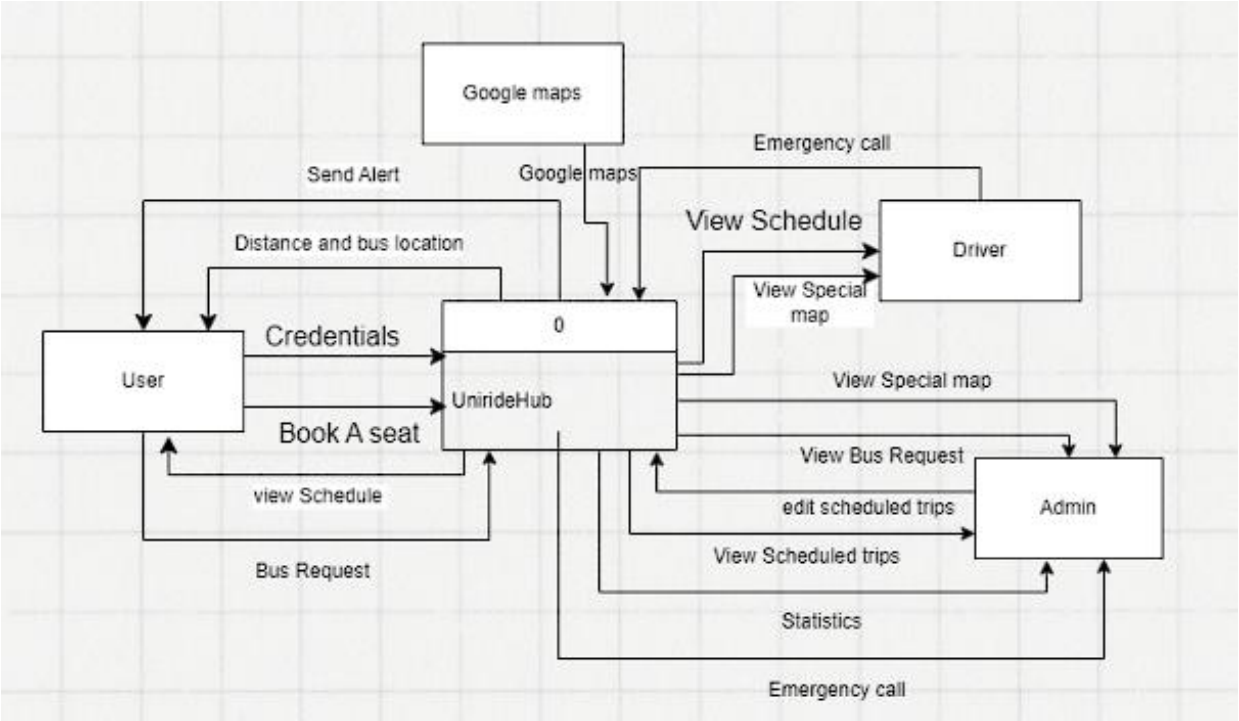


Diagram 4: Data Flow Diagram – Level 0

Level 1

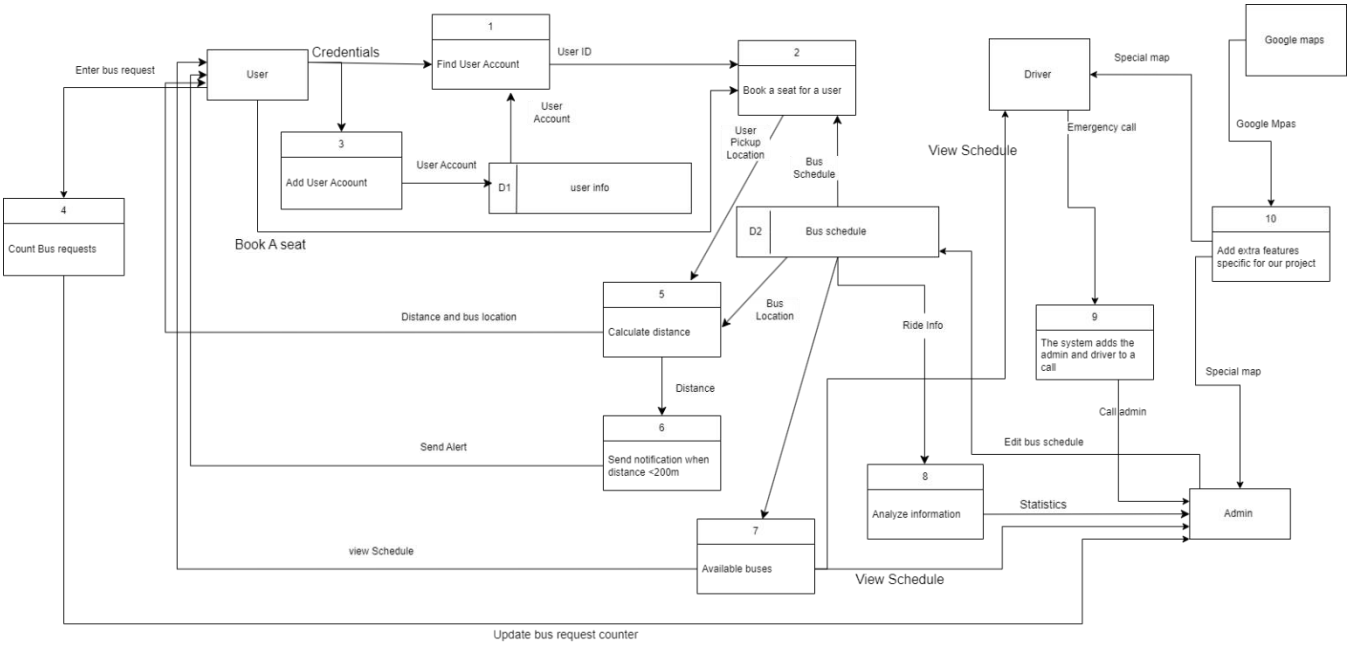


Diagram 5: Data Flow Diagram – Level 1

C. Database Diagram - Sandy

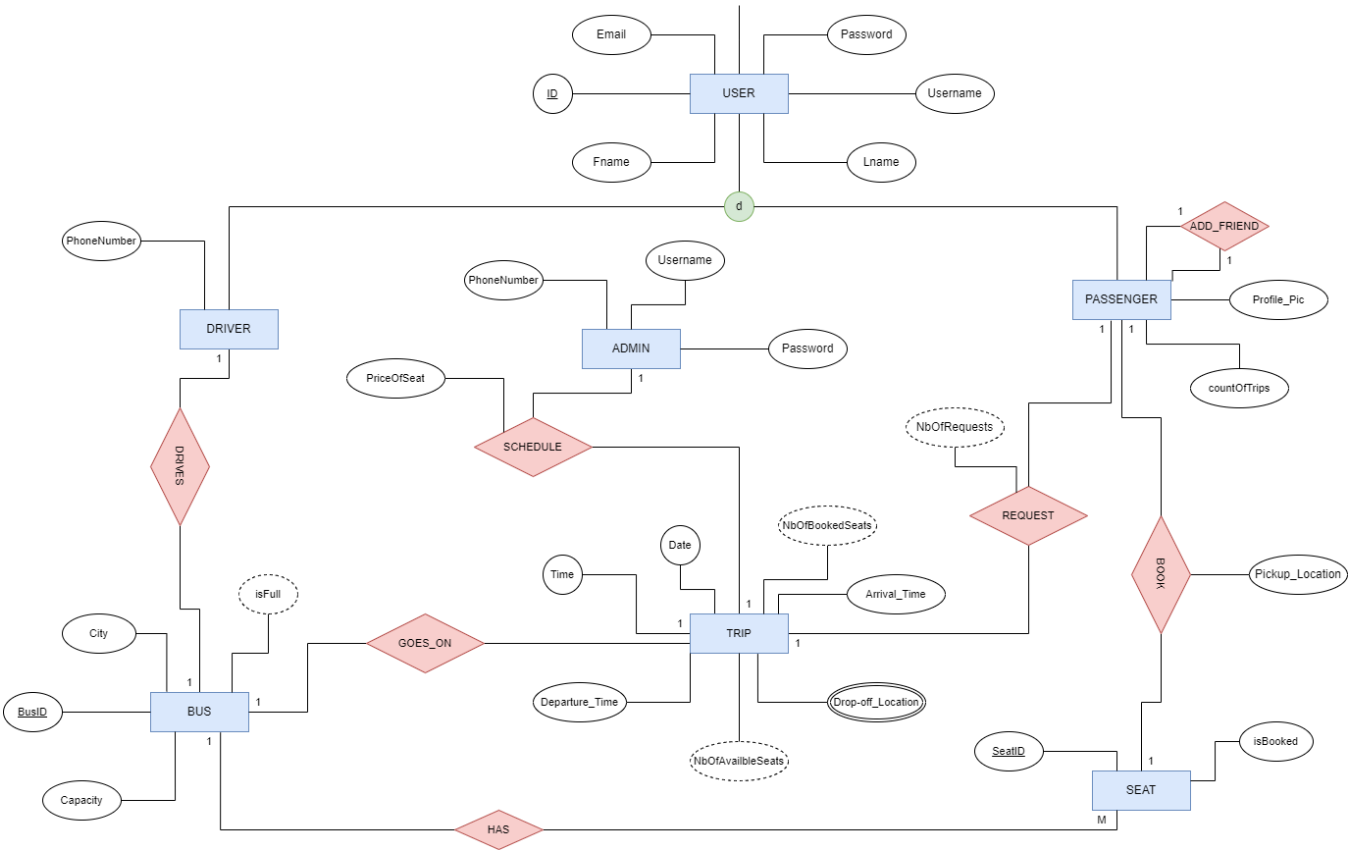


Diagram 6: ER Diagram

D. UML Class Diagram - Sandy

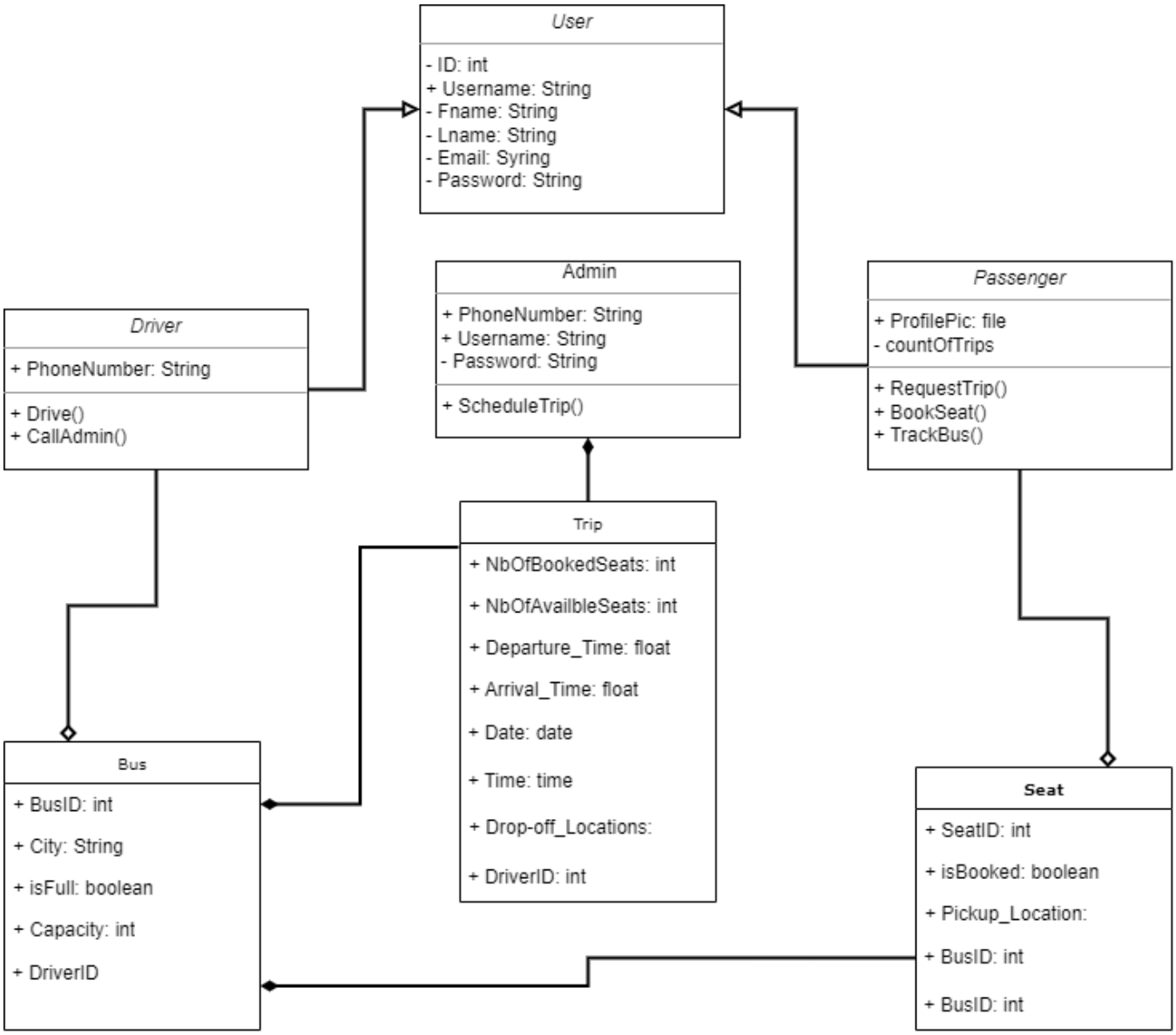


Diagram 7: UML Class Diagram

E. Sequence Diagrams - Jad

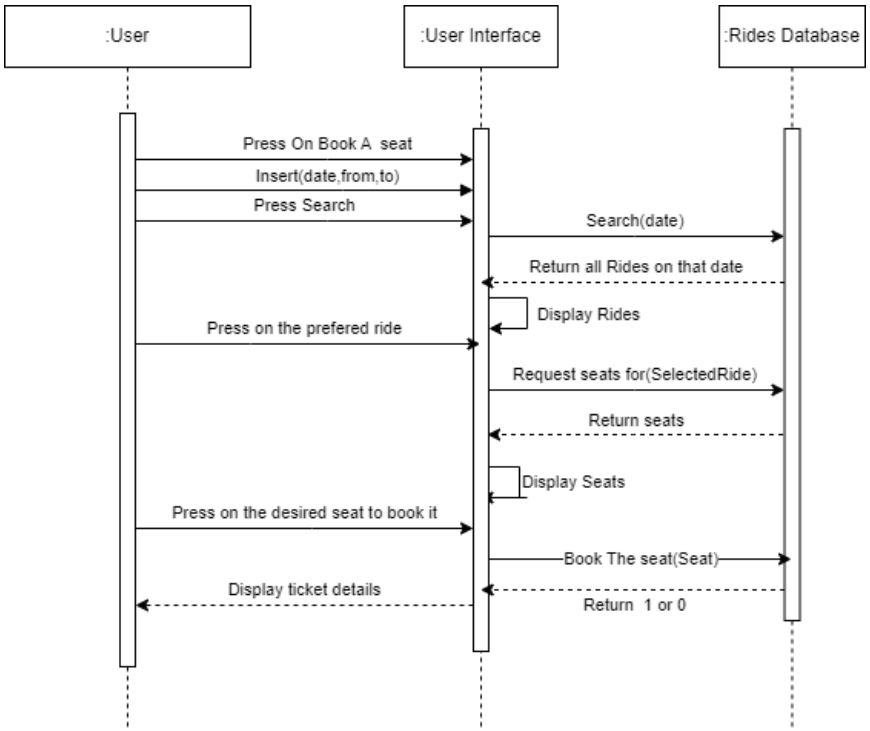


Diagram 8: Sequence Diagram – Booking a Seat

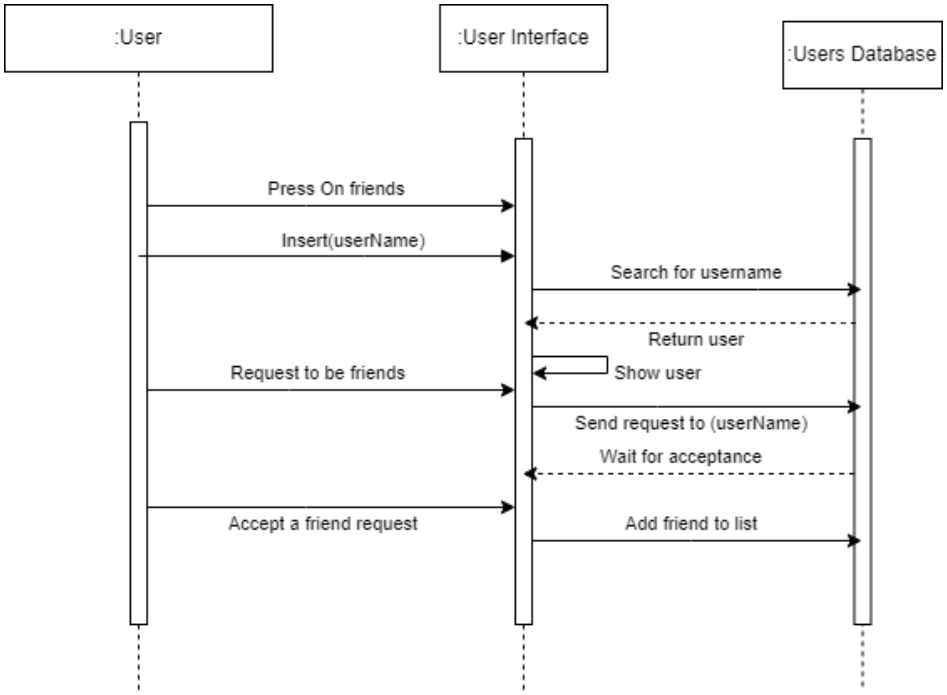


Diagram 9: Sequence Diagram – Adding a Friend

Chapter 5: Methodology – Amani / Moamen

A. Implementation - Moamen

1- Software Components:

Off-the-Shelf Components:

To speed up development and improve functionality throughout the implementation phase, we will make use of several off-the-shelf software components:

a) A multi-platform framework for creating mobile apps is called Flutter. With its extensive library of pre-built UI elements, it facilitates quick development and ensures a unified user experience for both iOS and Android platforms.

b) One popular relationship database management system is MySQL. MySQL is appropriate for storing user data, travel information, as well as additional application data since it provides strong functionality for data administration, retrieval, and storage.

c) Git is a system for distributed version control. Git promotes teamwork, allows codebase versioning, and guarantees code stability and integrity through the development process.

d) Trello: A management of projects application that facilitates teamwork, task organization, and progress monitoring. Trello is the best tool for organizing development activities and setting priorities because of its user-friendly design and adaptable workflow management features.

e) Canva and Microsoft Word: These programs will be utilized for graphic design and documentation, respectively. Canva offers design tools and templates for producing aesthetically pleasing interfaces and marketing materials, while Microsoft Word has powerful features for preparing project documentation.

Reasoning:

a) Using off-the-shelf components has several benefits, such as a shorter development time, cheaper development expenses, and availability of tried-and-true technology. We can expedite the creation of mobile apps while maintaining a native-like experience for users on all platforms by utilizing Flutter. Git promotes effective collaboration and version control, while MySQL offers a dependable and scalable database solution for storing application data. Trello facilitates efficient project administration, while Canva and Microsoft Word assist with design and documentation chores, respectively.

2-Software Toolset:

Programming Languages:

a) Dart (for Flutter development)

b) MySQL database is one of the data sources.

c) Reasoning: Because of its smooth interaction with the Flutter framework and support for contemporary language features like asynchronous programming and robust typing, we decided to use Dart as the main programming language for Flutter development. Due to its simplicity and ease of learning, Dart is appropriate for developers with different degrees of experience. Moreover, Flutter's hot reload function facilitates quick debugging and iteration, which raises developer productivity.

d) We chose MySQL for data storage because of its industry-wide adoption, performance, and dependability. Strong features are provided by MySQL for relational for our project. Additionally, MySQL is appropriate for managing massive volumes of data produced by the mobile application because of its flexibility as well as support for available configurations.

B. Testing - Amani

After implementing our application, we shall go into the testing phase. This phase is of paramount importance as it helps in catching and eliminating bugs and mitigating risks of software failures; thus, ensuring that the software reaches the highest quality standards. Firstly, unit testing will focus on testing each method individually, making sure that each of the application's components and features are fully functional on their own and performs as expected. Although testing of only three of the most complicated test cases was planned below, the unit testing will cover all the 20 use cases of the students, driver, and admin. Then, we will dive a little deeper into integration testing that will focus on testing and verifying how well distinctive features work together not just on their own, such as the interactions of related procedures within the UniRide Hub application. In our software, functionalities such as booking a seat, unbooking a seat, tracking a bus, and adding drop-off location are inter-connected and thus integration testing will be much needed for these methods. To enhance the user experience and ensure ease of use through flawless interface design of the UniRide Hub application, usability testing will be applied. By completing all the previously mentioned steps we will have an application that is robust, reliable, user-friendly, and that meets the requirements outlined in the SRS (software requirements)

Test Plan:

Purpose	The purpose of the UniRide Hub application is to enhance the campus transportation experience for students
Features to be Tested	1. Book a Seat Feature 2. Request a Bus Feature 3. Add a Friend Feature
Approach	First, we will unit testing each of the methods individually, then we will verify the interaction between distinctive features using integration testing, after that we will utilize Black box testing to validate user interactions and system responses, and finally we will do acceptance testing
Suspension Criteria & Resumption Requirements	Suspension Criteria: 1. Critical system failure 2. Unavailability of necessary resources for testing Resumption Requirements: 1. Resolution of critical system failures 2. Availability of required testing resources
Environmental Needs	1. Hardware: Compatible with all mobile devices operating on Android or iOS operating systems. 2. Networking: Stable internet connection for real-time updates 3. Software: Latest version of our UniRide Hub application
Schedule	Testing can be done by a few of the university's students along with the drivers and developers on a weekday (Monday through Friday) as the bus service is available only during these days
Acceptance Criteria	1. All test cases pass without critical failures
Roles & Responsibilities	1. Testers: In our case a group of students and drivers will be our testers, responsible for executing test cases and reporting defects and bugs 2. Developers: Responsible for fixing reported defects and ensuring application functionality, for our app our team are going to be the developers

Test Case 1:

Name	TC1: Verify Booking a Seat Functionality
Requirements	R4: Booking a Seat
Pre-conditions	1. User is logged into the bus shuttle app 2. Buses are available at the requested date & time 3. Seats are available for booking
Steps	1. Press on the 'Book a Seat' button 2. Click on 'Choose Bus' field 3. Enter the date, time, and city 4. Choose a bus from the list of buses available 5. Choose the preferred seat and click on 'Select' 6. User is then directed to a map 7. Choose the pick-up location on the given map and click on 'Enter'
Expected Results	1. The selected seat is marked as booked in the system 2. The user's pick-up location is accurately displayed on the driver's special map 3. The seat price is deducted from the user's account balance 4. The user receives a confirmation notification indicating successful booking

Test Case 2:

Name	TC2: Verify Request a Bus Functionality
Requirements	R6: Bus Request
Pre-conditions	1. User is logged into the bus shuttle app
Steps	1. Press on the 'Bus Request' button 2. User is directed to a form to input the date, time, and location of the requested trip 3. User enters the date, time, and destination of the requested trip and clicks 'Enter'
Expected Results	1. The system records the user's request for a bus trip with the specified date, time, and location 2. If the number of requests meets the threshold set by the admin, a trip is added by the admin to the schedule 3. If the number of requests is inadequate, the user receives a notification indicating the need for more requests or a delay in scheduling

Test Case 3:

Name	TC3: Verify Add a Friend Functionality
Requirements	R16: Connections
Pre-conditions	1. User is logged into the bus shuttle app
Steps	1. Press on the 'Add a Friend' button 2. Click on 'Search for a Friend' Field 3. Enter the friend's username 4. Click on 'Send a Friend Request' button under the friend's username
Expected Results	1. The system successfully sends a friend request to the selected friend 2. If the added friend accepts the friend request, they become friends 3. Upon becoming friends, both users gain access to view each other's booked bus seats

C. Maintenance - Moamen

Version Control:

Git:

Software development projects frequently employ Git, a distributed version control system. It lets developers work together as a team, keep track of code changes, and effectively manage several codebase versions.

Usage Rationale:

To encourage team members to work together on projects, we will use Git for version control. After cloning the project repository, each developer will work on features branches. Git's branch and merging features will make it easy for us to handle code changes and keep track of past changes over time. Furthermore, Git's ability to offer branching methods like release branching and feature branching will help us better organize our development efforts and integrate fresh capabilities into the main codebase more quickly.

Bug Tracking:

Jira:

Jira is a popular project management and issue-tracking tool that helps teams plan, monitor, and oversee software development tasks, including bug fixes and enhancements. Jira is used for bug tracking.

Usage Rationale:

Jira will be used for bug tracking so that problems that users report or that are found during testing may be methodically found, ranked, and fixed. Every bug report will be recorded as a Jira issue, complete with the bug's description, severity, reproducibility instructions, and assigned developer. We will monitor each bug's progress from discovery to resolution using Jira's configurable workflows, assuring prompt and efficient bug fixes. Jira's interface with continuous integration platforms and Git, among other development tools, will also expedite the bug-resolution process and give visibility into the state of ongoing maintenance tasks.

Code Review:

GitHub Pull Requests:

GitHub Pull Requests (PRs) are a tool that developers may use to examine and collaborate on code by proposing changes, reviewing existing code, and providing comments prior to merging it into the main repository.

Use Rationale:

To guarantee code quality, maintenance, and compliance with coding standards, we will use GitHub Pull Request for code reviews. Programmers will create pull requests, which will initiate automated tests and let other teammates review, comment on, and offer improvements on the code changes before integrating them into the main branch. Code reviews carried out via GitHub PRs will assist in locating potential problems, guarantee uniformity throughout the codebase, and promote cooperation and knowledge exchange among team members.

Continuous Integration and Deployment:

Jenkins:

Jenkins is an open-source automation server that facilitates pipelines for continuous integration and continuous delivery, or CI/CD. Teams can use it to automate build, test, and deployment procedures, which results in software that is delivered more quickly and reliably.

Use Rationale:

To automate the build and testing of code changes and speed up the delivery of new releases into production environments, we will use Jenkins for continuous integration and deployment. Jenkins pipelines will be set up such that whenever changes are made to the project repository, automated builds and tests are started. Jenkins will automatically deploy the application to staging or production environments if the tests are complete, guaranteeing that users receive new features and issue fixes in a timely and dependable manner. Jenkins' reporting and monitoring features will also offer insights into the progress of builds and deployments.

Chapter 6: Conclusion - Amani

In brief, we wanted to develop an application for those who spend their mornings waiting on the road and praying that they get noticed and picked up by the drivers. And for those who get excited to go on the bus but then are disappointed by the unavailability of seats and end up standing the whole trip. UniRide Hub was made to make every student's life so much easier, as our user-friendly and feature-rich app ensures real-time access to bus locations and estimated arrival times, as well as the ability to book and unbook a seat on the bus and the privilege to add a location to the driver's map. Not to forget, the opportunity presented for everyone to expand their social circle and make new friendships, tightening the bonds between the university's students. We also plan in the future to enhance and improve our app with more special features that will help encourage the university's community to be part of a much more sustainable and eco-friendly transportation approach.