Table of Contents

Table of figures

Table of Tables

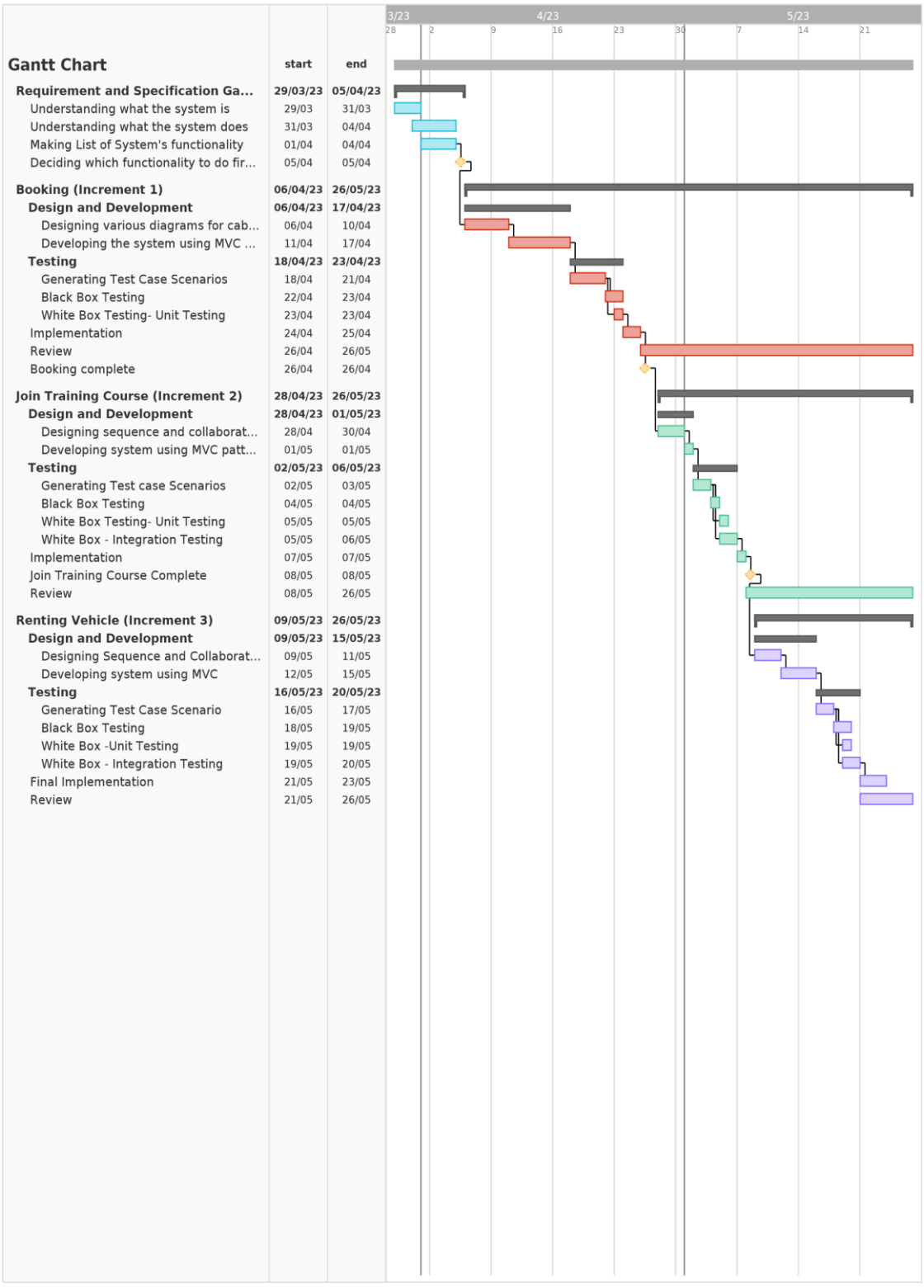
1. Introduction

For this coursework,we are supposed to create a detailed framework/outline for a web application that focuses on delivering transport services like cab services and vehicle rental services for vehicles like cargo trucks, bulldozers, etc for a company called Allgemein who wants to enter in the Transport Industry. As software engineers, we're supposed to design the entire software, how it looks, how the system interacts with the user, how the system works, relationship between different classes in class diagram and finally a gantt chart to show the methodology used for coding.
All this is achieved by creating a number of diagrams to help us better understand the working of the system and it's interactions. The use case diagram is developed to sum up the entire functionality of the system. Then to further visualize each interaction objects and actors, we create a sequence diagram and collaboration diagram based on the expanded use-case description and then as a blue print for the whole system, we create a class diagram.

2. Gantt Chart

A Gantt chart is a project management tool that shows relationship between work completed over a period of time and the time that was specified or planned for the task. The left side of the gantt chart oulines a list of tasks, while the right side has a timeline with bars that visualize work done. Gantt chart contains every detailed planning steps of the project including the start and end dates of tasks, milestones, dependencies between tasks, etc (Atlassian, 2023).

Here's the gantt chart for Allgemein transport system. The below gantt chart shows all the work done and the time duration for each work. For developing this coursework, I followed Incremental model. In the first stage of Incremental model, requirements are gathered and SRS is developed. Then work on any one of the selected function begins. First, the functions design is completed and then it is developed followed by testing, implementation and review. This process occurs for each of the increment. The customer can review the software and give feedback at any stage of system development.



Gantt Chart	start	end
Requirement and Specification Ga...	**29/03/23**	**05/04/23**
Understanding what the system is	29/03	31/03
Understanding what the system does	31/03	04/04
Making List of System's functionality	01/04	04/04
Deciding which functionality to do fir...	05/04	05/04
Booking (Increment 1)	**06/04/23**	**26/05/23**
Design and Development	**06/04/23**	**17/04/23**
Designing various diagrams for cab...	06/04	10/04
Developing the system using MVC ...	11/04	17/04
Testing	**18/04/23**	**23/04/23**
Generating Test Case Scenarios	18/04	21/04
Black Box Testing	22/04	23/04
White Box Testing- Unit Testing	23/04	23/04
Implementation	24/04	25/04
Review	26/04	26/05
Booking complete	26/04	26/04
Join Training Course (Increment 2)	**28/04/23**	**26/05/23**
Design and Development	**28/04/23**	**01/05/23**
Designing sequence and collaborat...	28/04	30/04
Developing system using MVC patt...	01/05	01/05
Testing	**02/05/23**	**06/05/23**
Generating Test case Scenarios	02/05	03/05
Black Box Testing	04/05	04/05
White Box Testing- Unit Testing	05/05	05/05
White Box - Integration Testing	05/05	06/05
Implementation	07/05	07/05
Join Training Course Complete	08/05	08/05
Review	08/05	26/05
Renting Vehicle (Increment 3)	**09/05/23**	**26/05/23**
Design and Development	**09/05/23**	**15/05/23**
Designing Sequence and Collaborat...	09/05	11/05
Developing system using MVC	12/05	15/05
Testing	**16/05/23**	**20/05/23**
Generating Test Case Scenario	16/05	17/05
Black Box Testing	18/05	19/05
White Box -Unit Testing	19/05	19/05
White Box - Integration Testing	19/05	20/05
Final Implementation	21/05	23/05
Review	21/05	26/05

Figure 1: Gantt Chart of the System

3. Use Case Model

A Use case model is the overview of the system from user's prespective. The use case defines the interaction between the system and the user and the system's behavior required to meet these objectives. A use case diagram contains of three main elements such as actors, use-cases and the relation between them (javatpoint.com, 2008). Here's a detailed explanation of them:

Actor

An actor is basically a user who uses the system. A system can have many users for example a transport system can have actors such as Drivers, Staff, Employee, Customers, Admin, etc.

Figure 2: Actor Use-Case

Use-Case

Each actor performs a specific role in the system. The role performed by the actor in the system is called a Use-Case. There are many use-cases in a Use-Case diagram. They depict everything that a user can do in a system. In a use case diagram, there can be seen two types of use cases, one connected to the actor directly which is called base use case while the other connected to the base use-case via include or extend. The difference is that while the base use-case can be defined independently and is meaningful by itself, the extension use case is not meaningful on its own. The extension use case can be payment option after booking a ride or rating option once the ride is complete. The base use case gives extension use case meaning.

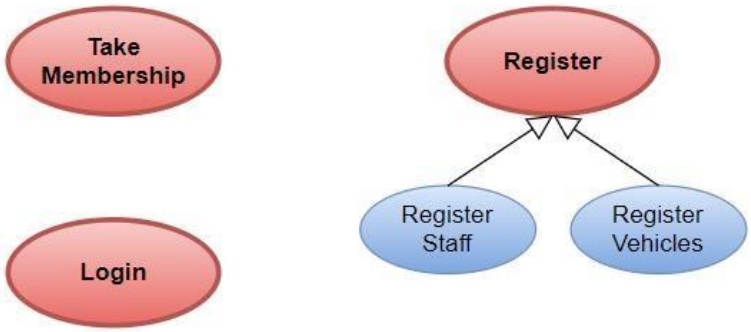


Figure 3: Use-Cases

Relation

The relationship between use-cases and actors is called association. The use-cases are connected to the actors that contribute in it and the relation is called Association. In a use-case diagram, other relations can be seen as well such as include, extend and generalization. The include relation joins base use-case to extended use-cases. Include means that when the functionality of base use-case is being performed, functionality of extended use-case gets performed as well where in extend relation, the functionality of extended use-case is optional and depends on the actor if he wants to use it or not. Generalization shows relationship between usecases as child and parent.

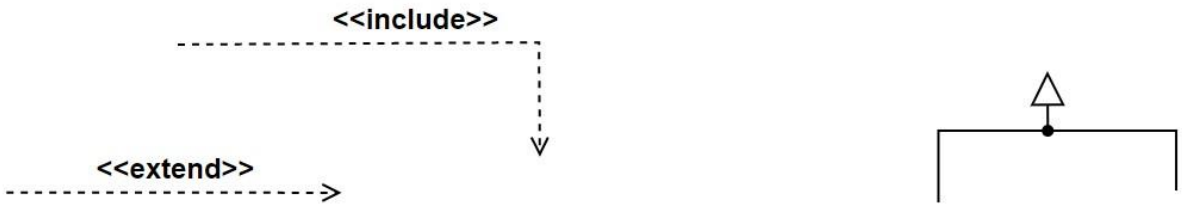


Figure 4: Relations of Use-Case

3.1. Use Case Diagram

Here is the Use-Case diagram for the transport system of Allgemein.

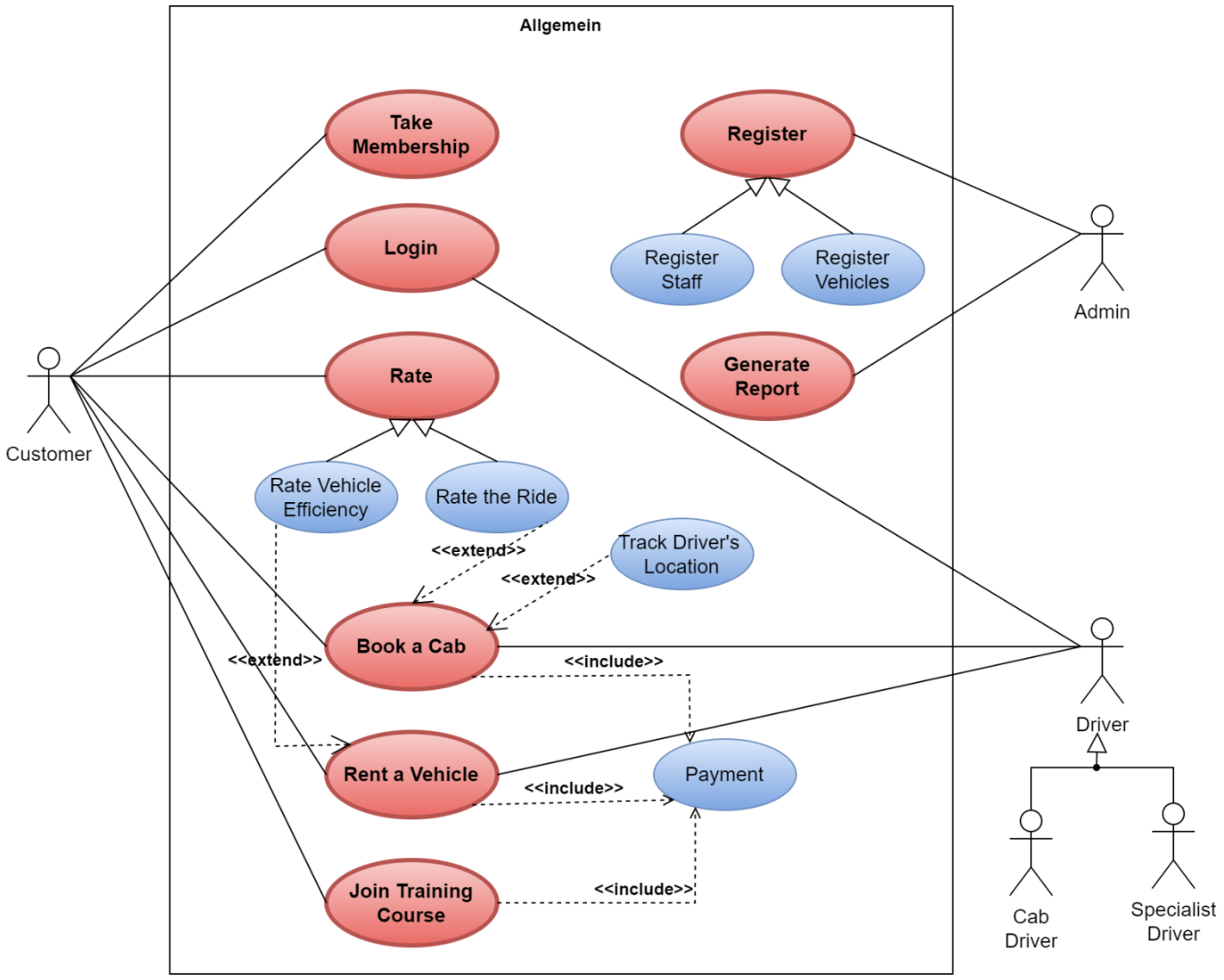


Figure 5: Use-Case diagram

3.2. High Level Use Case Description

Use Case	Take Membership
Actor	Customer

Description	When a new Customer wishes to become a member of our application. He/She enters their personal details and becomes a member.

Table 1: Take membership high level use-case

Use Case	Login
Actor	Customer
Description	When a member wishes to use our service. He/She have to login to the system by the providing their username and password. If the login details match that of the database, the member is logged into our system.

Table 2: Login high level use-case

Use Case	Book a Cab
Actor	Customer, Driver
Description	When a member wishes to book a cab. He/She enters the necessary informations like pickup location, drop off locations and sends a request. A driver then accepts the request and a Cab is booked. Once the ride is over, option for payment is shown to the member. The member can also track driver's location and rate the ride.

Table 3: Book a Cab high level use-case

Use Case	Rent a Vehicle
Actor	Customer,Driver
Description	When a member wishes to rent a vehicle, he/she enters the necessary information, pays the fare and a vehicle is rented. Once renting duration is over, the member gets an optional option to rate the vehicle efficiency.

Table 4: Rent a Vehicle High level description

Use Case	Join Training Course
Actor	Customer
Description	When a member wishes to join a training course, he/she enters the necessary details of the training course, pays the fare and the course is unlocked.

Table 5: Join Training Course high level desc

Use Case	Rate
Actor	Customer

Description	Once the service ends, the customer has the option to rate the ride of the cab or rate the efficiency of the vehicle rented.

Table 6: Rate high level desc

Use Case	Register
Actor	Admin
Description	When a new staff is hired or vehicle is bought, the admin registers their details into the system.

Table 7: Register high level desc

Use Case	Generate Report
Actor	Admin
Description	When the admin, wants to look at the reports of cab rides, or vehicle renting, revenue generation, the system will generate a report using the datas collected from the users.

Table 8: Generate Report high level desc

3.3. Expanded Use Case Description

Use Case	Book a Cab

Actor	Customer,Driver
Description	When a member wishes to book a cab. He/She enters the necessary informations like pickup location, drop off locations and sends a request. A driver then accepts the request and a Cab is booked. Once the ride is over, option for payment is shown to the member. The member can also track driver's location and rate the ride.
Actor Action	**System's Response**
1.The Customer enters the necessary informations and sends a pickup request. 3. A driver accepts the request. 5.Once ride is over, customer makes the payment. 8. The Customer either rates the ride or skips the process.	2.The request is forwarded to all nearby drivers. 4.Displays Driver's details,ride fare and payment option. 6. System validates the payment. 7. After payment is made, the option to rate the ride is displayed to the user.
9. The ride ends.	

Table 9: Book Cab Expanded Desc **Alternative**

Course of Action:

Line 3: No driver is available nearby or no driver accepts the request. Appropirate message is displayed to the user and Use Case ends.

Line 5: Online Payment fails. Customer pays the driver physically. Use-Case continues from Line 7.

Use Case	Join Training Course
Actor	Customer
Description	When a member wishes to join a training course, he/she enters the necessary details of the training course, pays the fare and the course is unlocked.
Actor Action	**System's Response**
2. The customer choses a training course. 4. Customer makes the payment.	1.The Customer is displayed a number of trainign courses along with details. 3. A message asking user to pay for the selected training course is displayed. 5. System validates the payment and a message "Training Course Joined" is displayed along with course details.

6. Customer enrolls in the training course and can access all the materials online. 8. The Customer either rates the ride or skips the process. 9. The ride ends.	7. Once the training course is over, the option to rate the course is displayed to the Customer.

Table 10: Join Training Course Expanded desc

Alternative Course of Action:

Line 2: Customer doesn't find the desired course. Use case ends.

Line 4: Payment fails. Message "Please Try Again" is displayed. If Payment fails three times, message to 'try again later' is displayed. Use case ends.

4. Sequence Diagram

Sequence diagram shows interaction between objects in a sequential manner. It shows the entire interaction between actors and objects for a particular use-case. Sequence diagram has a number of elements like:

Activation Bar: A thin rectangle above the lifeline of the object represents the period of time for which the object is performing an operation. The top and bottom of the rectangle are aligned with the initiation and the completion time respectively (visual-paradigm.com, 2019).

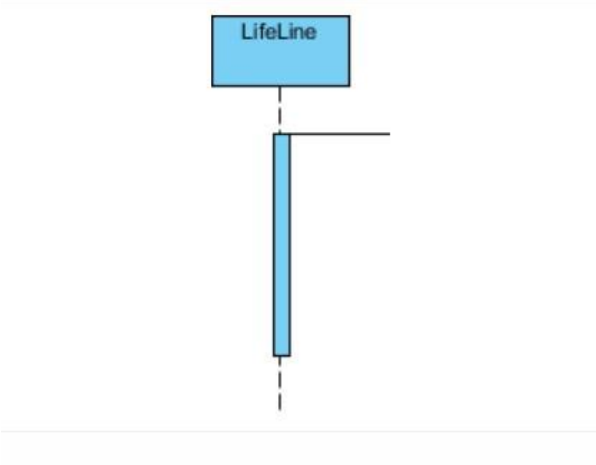


Figure 6: Activation Bar

Object Lifeline: A lifeline represents an individual participant throughout the interaction.

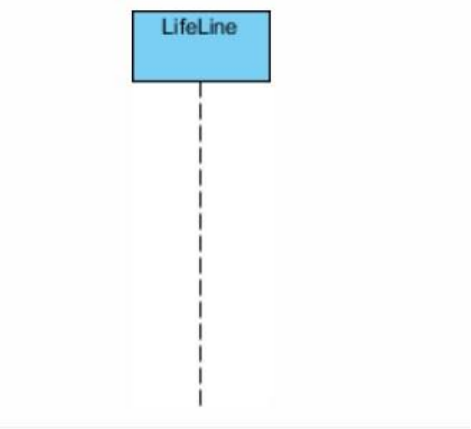


Figure 7: Object Lifeline

Messages: A message defines communication between actors, ui and objects. It shows the communication between lifeline of those interaction. Messages are of many types like synchronous, asynchronous, create, return, destroy. Their basic job is to simplify the communication between interactions.

Figure 8: Messages sequence diagram

Here's a detailed sequence diagram of the book a cab use-case :

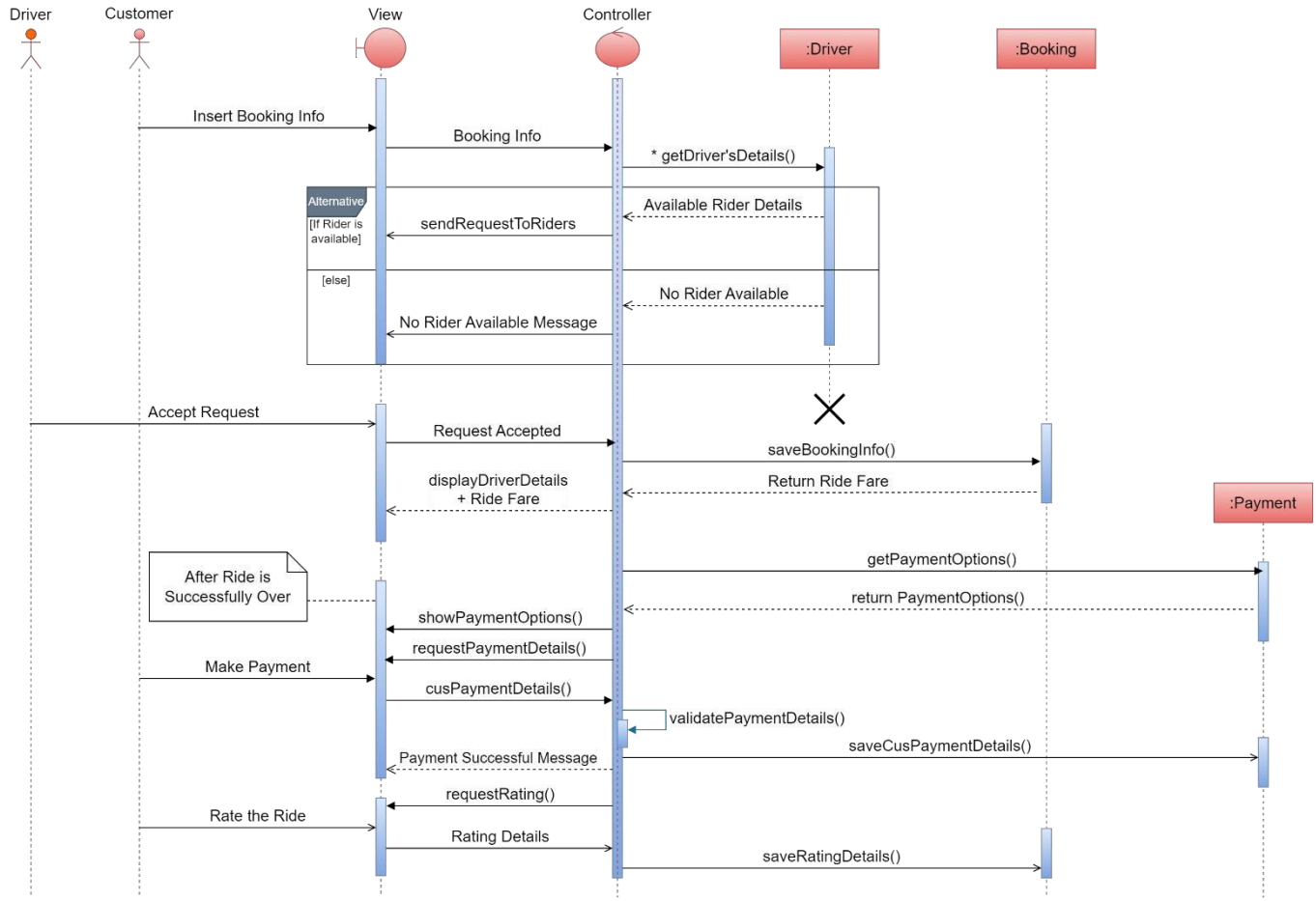


Figure 9: Sequence Diagram

5. Communication Diagram

A communication diagram is a behavioral UML diagram that shows how object interacts with each other. Unlike sequence diagram, communication diagram emphasizes the messages exchanged between objects in an application. Here's a communication diagram of the book a cab use-case showing all the object interaction.

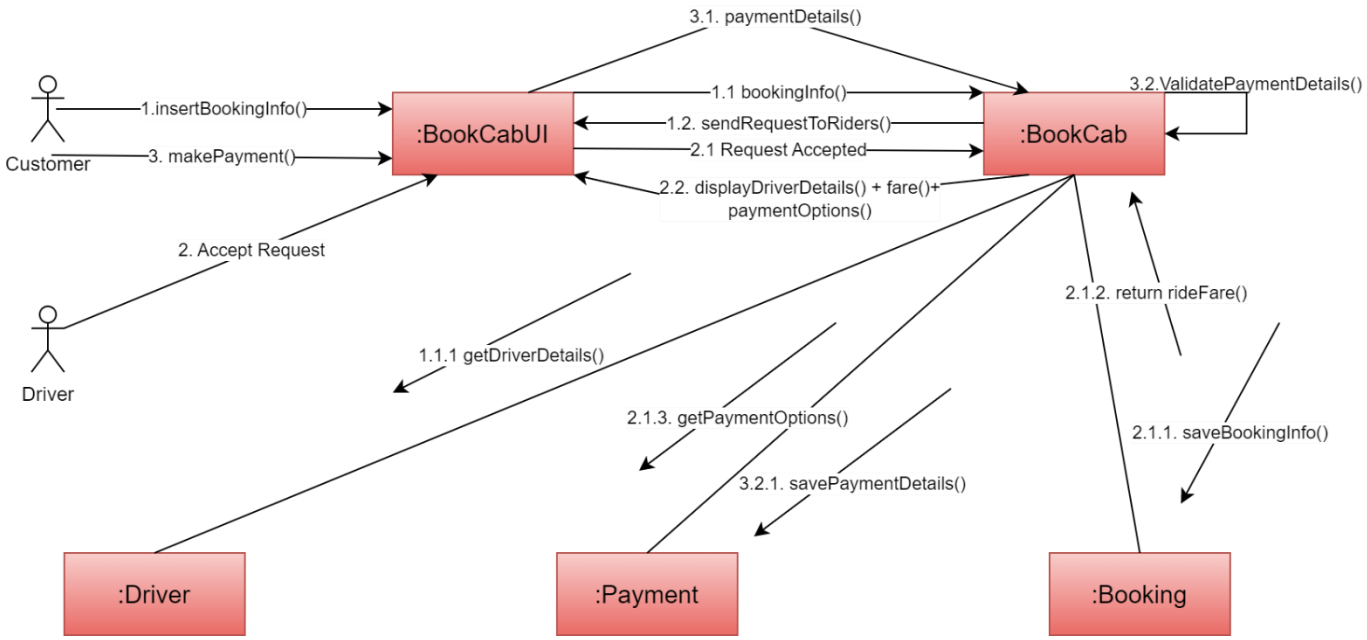


Figure 10: Communication Diagram

6. Class Diagram

A class diagram is a structural UML diagram that describes the structure of a system by plotting their classes, their attributes, operations and the relationships among the objects (en.wikipedia.org, 2023). It provides a sense of orientation, detailed insight into the structure of the system. The relationship among class diagram can be complicated. In this class diagram, we're supposed to show association relation, aggregation relation and composition relation between classes. Although not compulsory, we can show other relations as well like dependency. In SDLC, based on the principle of Object Orientation, class diagram can be implemented in various phases like analysis, design or implementation. Here's the class diagram of the Allgemein transport system.

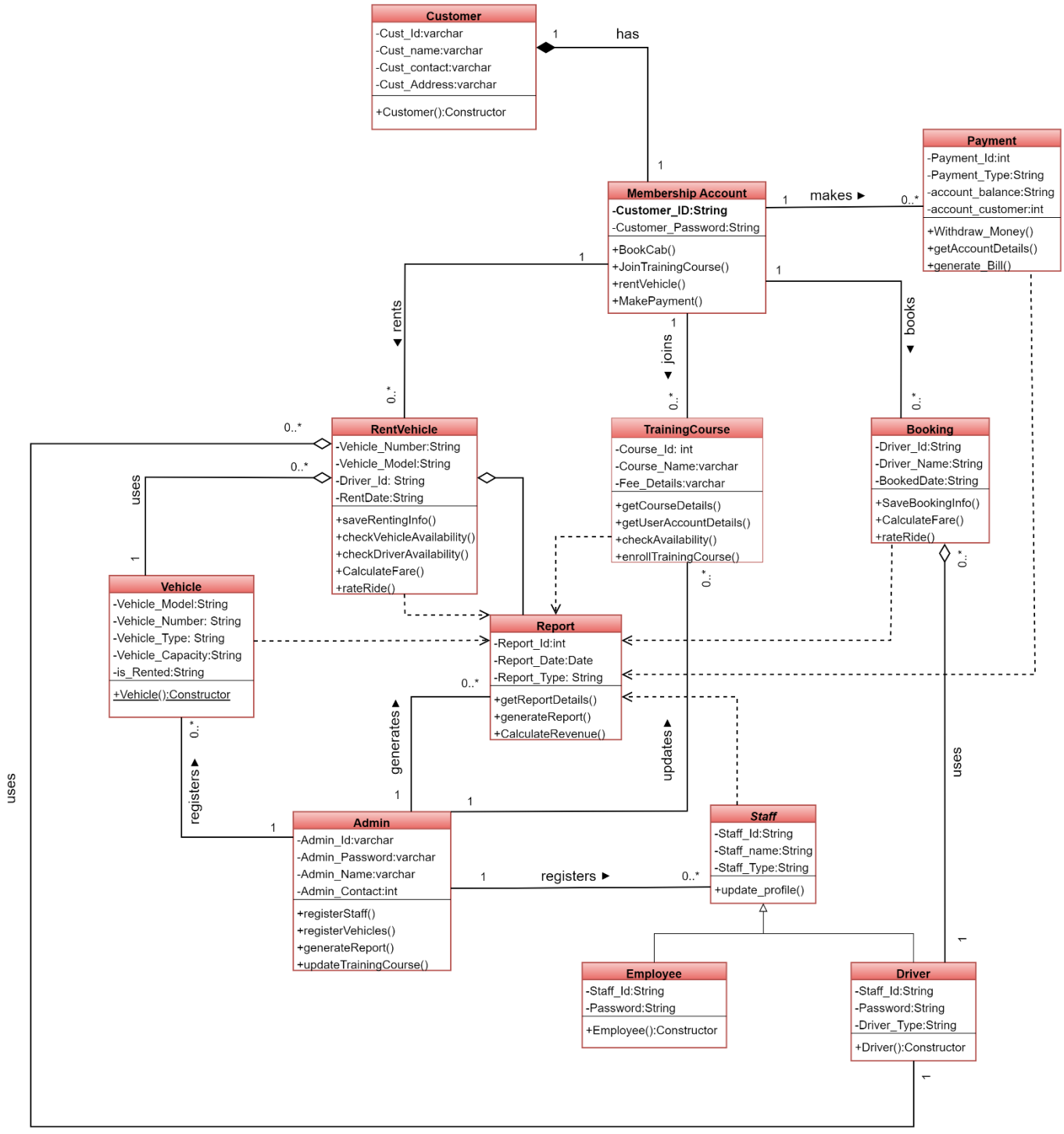


Figure 11: Class diagram

7. Further Development

7.1. Methodology Used

In order to complete this coursework, we first needed to select a methodology for our software development. After doing a lot of research on different software development methodologies and my case scenario, I decided to use Incremental Model for my software scenario as it is easy to test and debug and we deliver module by module to the customer instead of all at once. As a result, the review process goes on throughout the software development lifecycle. And it is also cost effective since we can change the product at any stage of development cycle according to customer's review. A detailed explanation of various phases of Incremental Model are as follow:

1. **Requirement analysis:** The product analysis expertise identifies the requirements in the first phase of the incremental model. The requirement analysis team also comprehends the system functional requirements. This phase is extremely important for the incremental paradigm of software development. In this phase, SRS document is completed to study about the overall requirement of the software.

2. **Design & Development:** Using the SRS document, the product architect creates an optimal product architecture at this phase. Based on the requirements supplied by SRS, the product architect generates a preliminary design, working models, describes how the software works, how the new design appears, how the control flows from one screen to another, and so on (www.interviewbit.com, 2001).

3. **Testing:** The testing step in the incremental approach evaluates the performance of each current function as well as added capabilities. Various approaches are utilized to test the behavior of each task throughout the testing process.

4. **Implementation:** The implementation step supports the development system's coding phase. It includes the final coding that is designed during the designing and developing process and tested during the testing phase. Following the conclusion of this phase, the

number of working products is increased and updated all the way to the final system product.

Here's a detailed architecture for the incremental model:

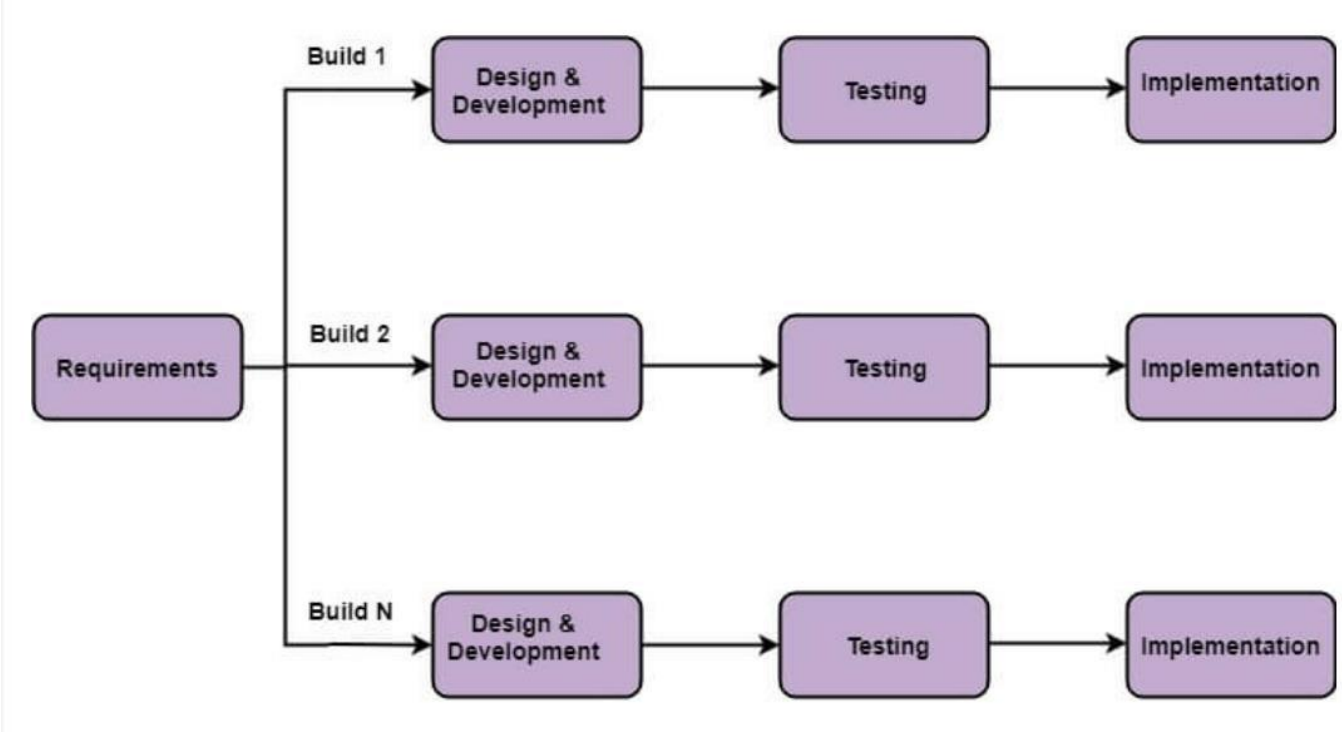


Figure 12: Incremental Model Architecture

Since, some of the work has been done beforehand, we begin from Design and Development phase.

7.2. Design and Development Process

During the design process, me and my team developed a variety of UML diagrams to visualize the indepth working of the system. First we created a Use-Case diagram to visualize how the actor interacts with the system. Then we created a sequence diagram and collaboration diagram of the Cab Booking (Increment-1) to visualize how the objects interacts with one another and to see the sequence of flow of message between objects for Cab Booking functionality.

The code is developed from scratch according to the requirements in SRS as well as the design diagrams made. The code must be written in an organized and detailed manner so to not impact the performance of software. Hence, taking all these things into consideration, we decided to follow MVC pattern to develop our software.

Model-View-Controller(MVC) is an architectural pattern that seperates the code of an application into three distinct layers based on their functionality: model, view and controller. Each of these layer handle specific development aspects of an appilcation. The controller contains the business logic while the view is the UI part of the application and model is the database. Following MVC pattern makes the code reusable and helps keep the code clean.

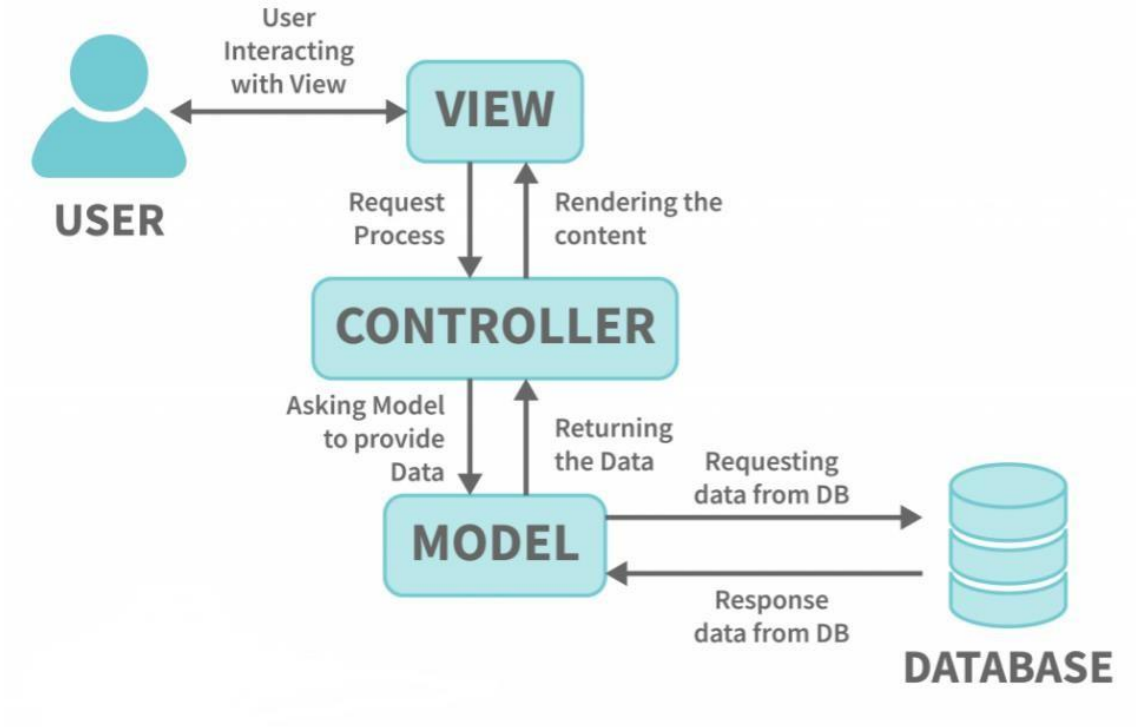


Figure 13: MVC pattern architecture

In Increment-1, we coded the Cab booking functionality along with Account UI and all the functions required for booking a cab like rating, payment,etc.

For Increment-2, we again developed Sequence and Collaboration diagram for Join Training Course. Developed the system using MVC pattern and added it to the existing Incremental Process.

For Increment-3, we developed Sequence and Collaboration diagram for Vehicle Rental Process and it was coded using MVC pattern and added to the existing Incremental process.

7.3. Testing

After finishing each Incremental processes, we tested each of the models first individually and then simultaneously by first generating Test Case Scenarios for Black box testing and white box testing. Here are some of the tests that we did throughout our project:

i) Boundary Value Analysis: Boundary Value Analysis is a software test design technique that involves determintaion of boundaries for input values and selecting values that are at the boundaries and just inside/outside of the boundaries as test data. In maximum cases, the software behaves unexpectedly towards the boundaries of input doman rather than in the center of input. So, to remove the complexity of testing and decrease the number of test cases, we went with boundary value analysis.

ii) Unit Testing: In Unit testing, the smallest testable parts of an application, called units, are tested individually and independently. This testing is performed by the software developers and sometimes QA staff during the development process. The main objective of unit testing is to isolate written code to test and determine if it works as intended (www.techtarget.com, 2021). According to our incremental model, unit testing is practical since we develop small chuncks of software in every increment and unit testing makes debugging process easier.

iii) Integration Testing: A type of software testing, where software modules are integrated logically and tested as a group. In my incremental model of software development, many programmers create multiple software modules which are integrated together at the end. This testing's objective is to find how various software components interact when they are combined together and the issues that arise.

7.5. Implementation and Review

Since, we followed Incremental model approach to design this software, after implementation stage of each increment, the review stage goes on. Unless the project is fully complete, the customer can give review and feedback anytime. Many changes were made in the software according to user's review. When we delivered the first increment, the user was not satisfied with the UI and so we tweaked it in second increment and added some missing functionalities that were not specified in the SRS hereby delivering quality product at low cost with user satisfaction.

8. Prototype

For this Coursework, we were supposed to develop prototype of our application including all important features like Booking, renting or joining a training course. Here are some of the prototypes for those features:

8.1. Take Membership/Login Page

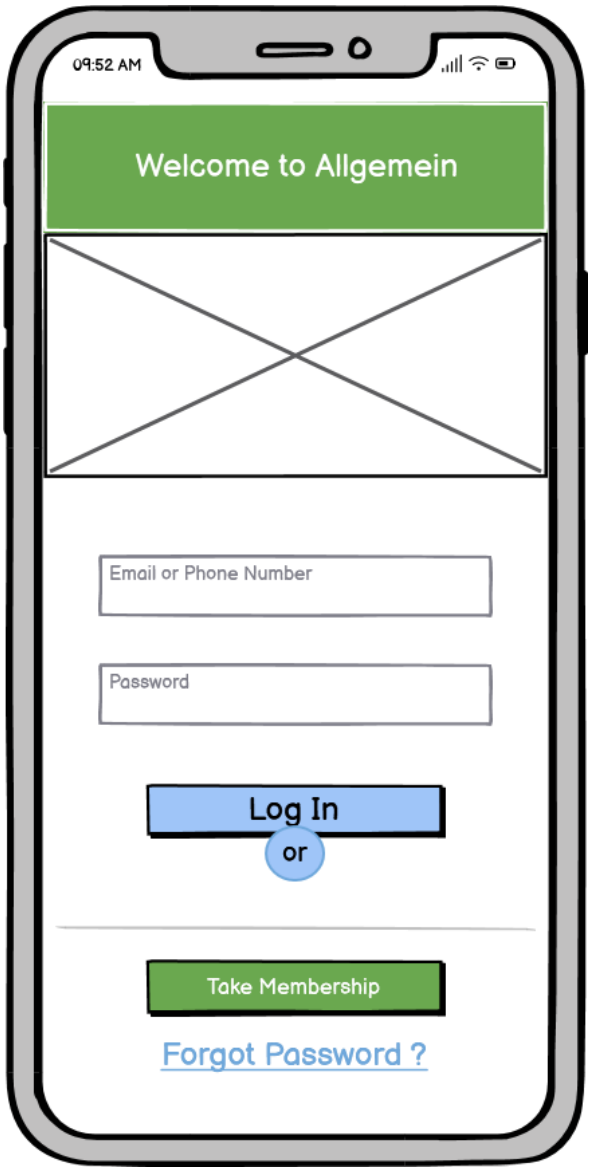
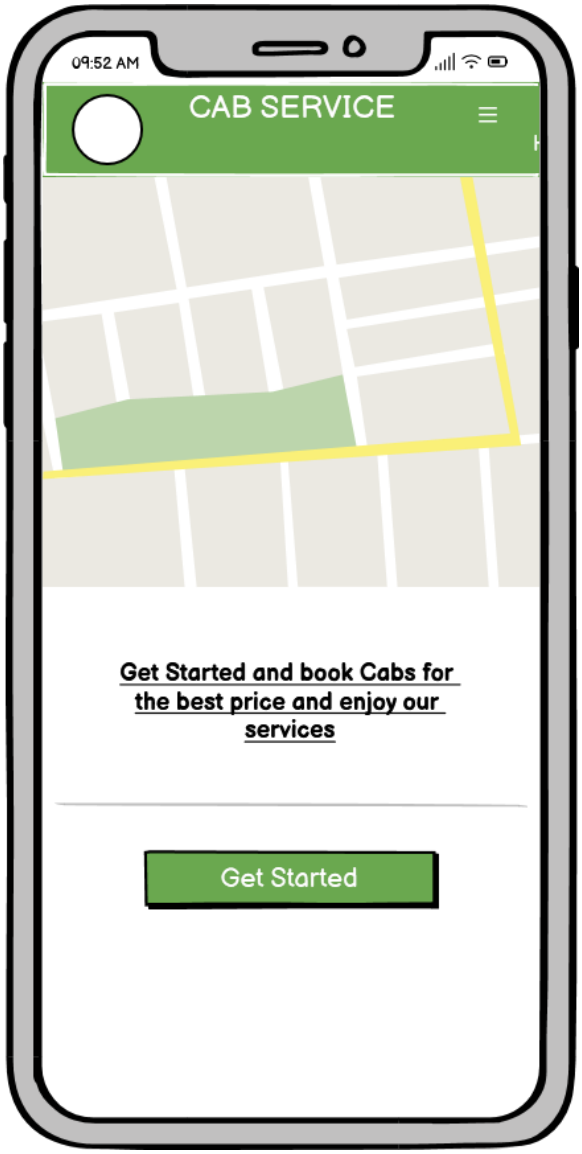


Figure 14: Login Prototype

8.2. Book Cab


09:52 AM
CAB SERVICE
Get Started and book Cabs for the best price and enjoy our services
Get Started

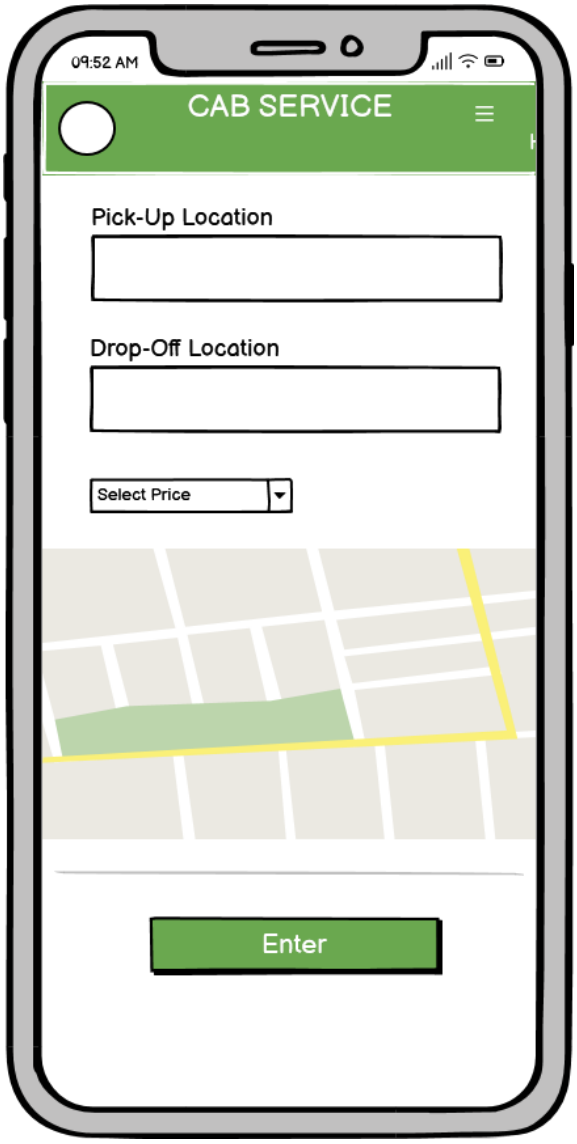


Figure 15: Book Cab Prototype

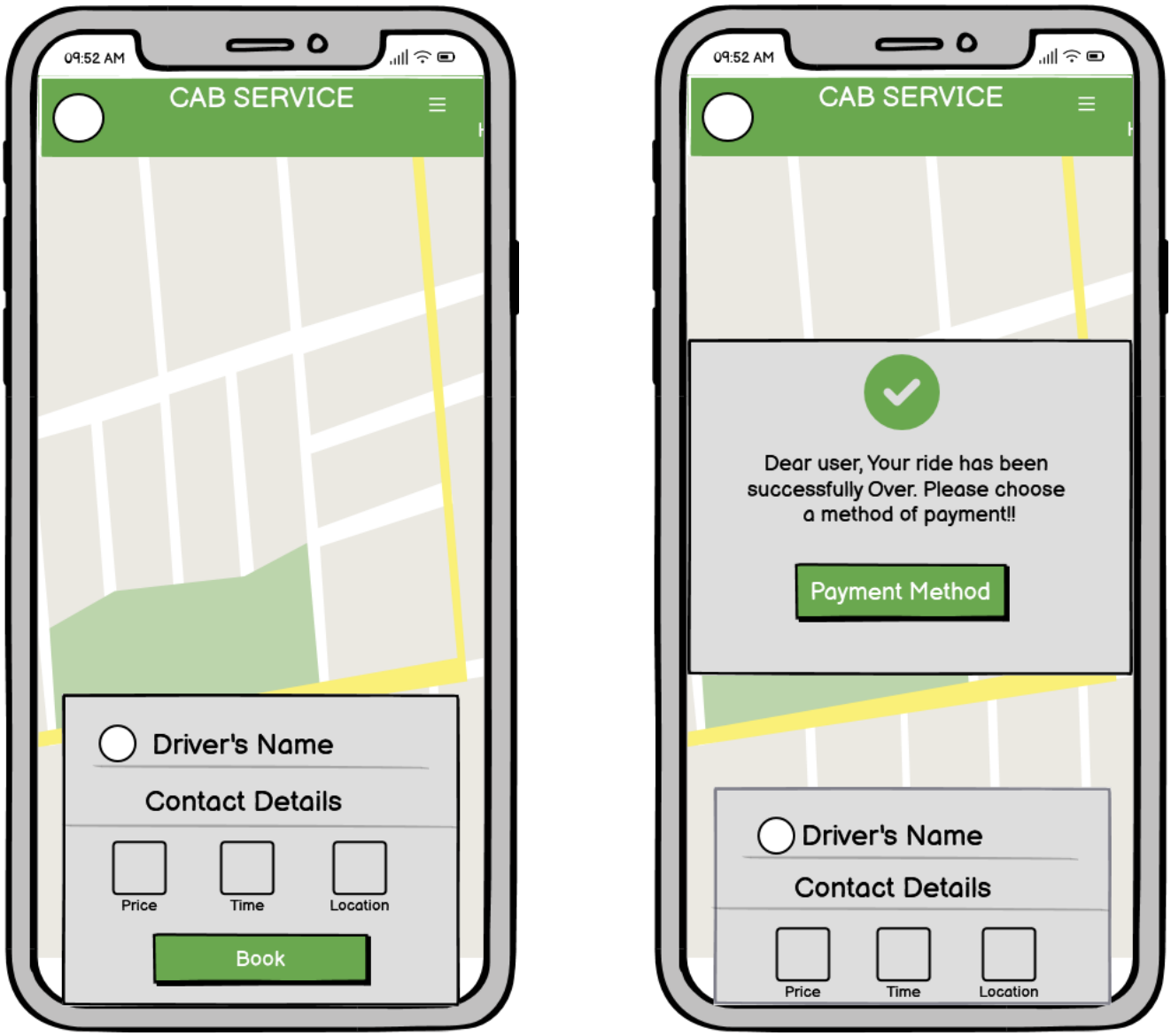


Figure 16: Book Cab Prototype

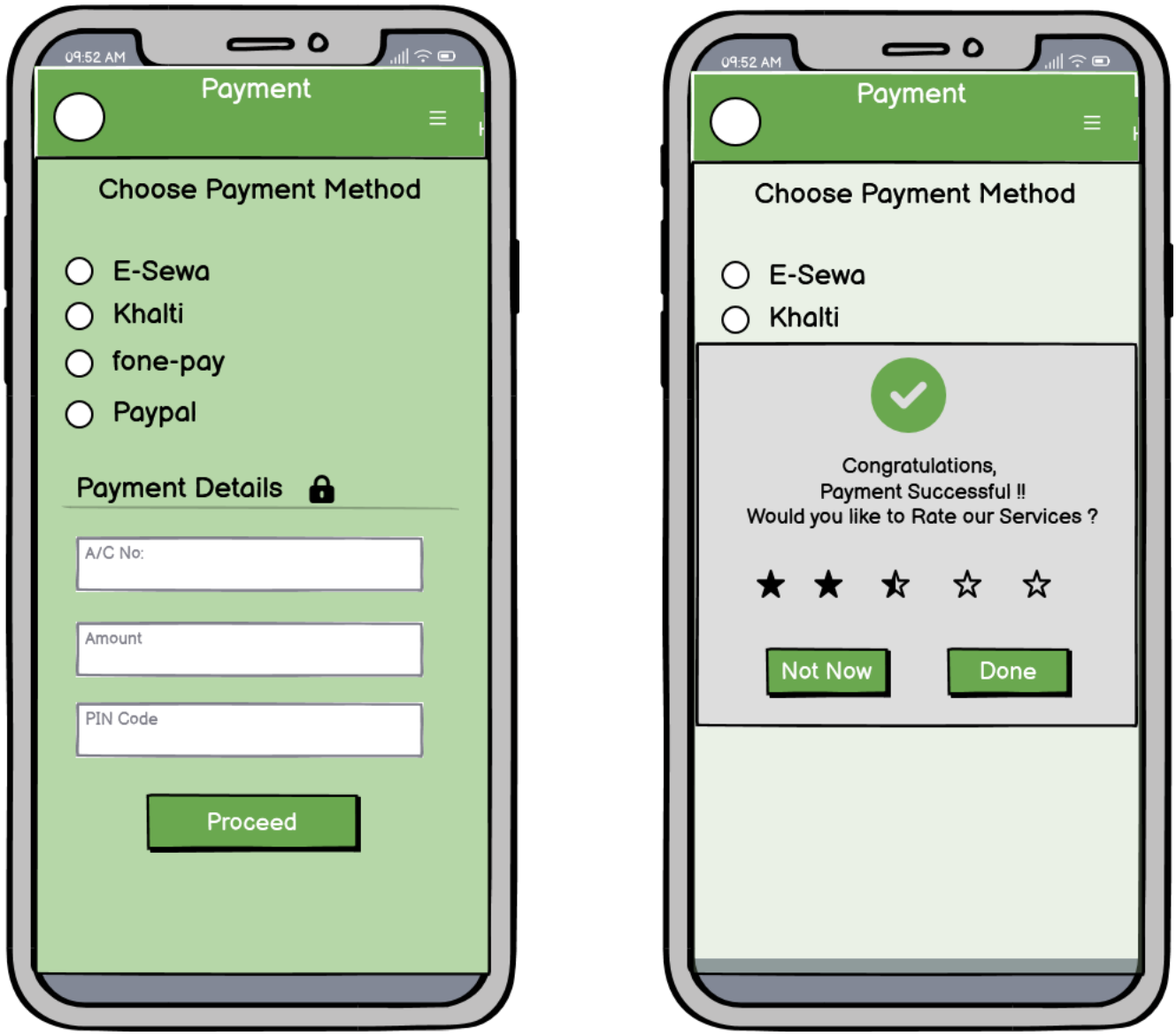


Figure 17: Book Cab Prototype

8.3. Rent Vehicles

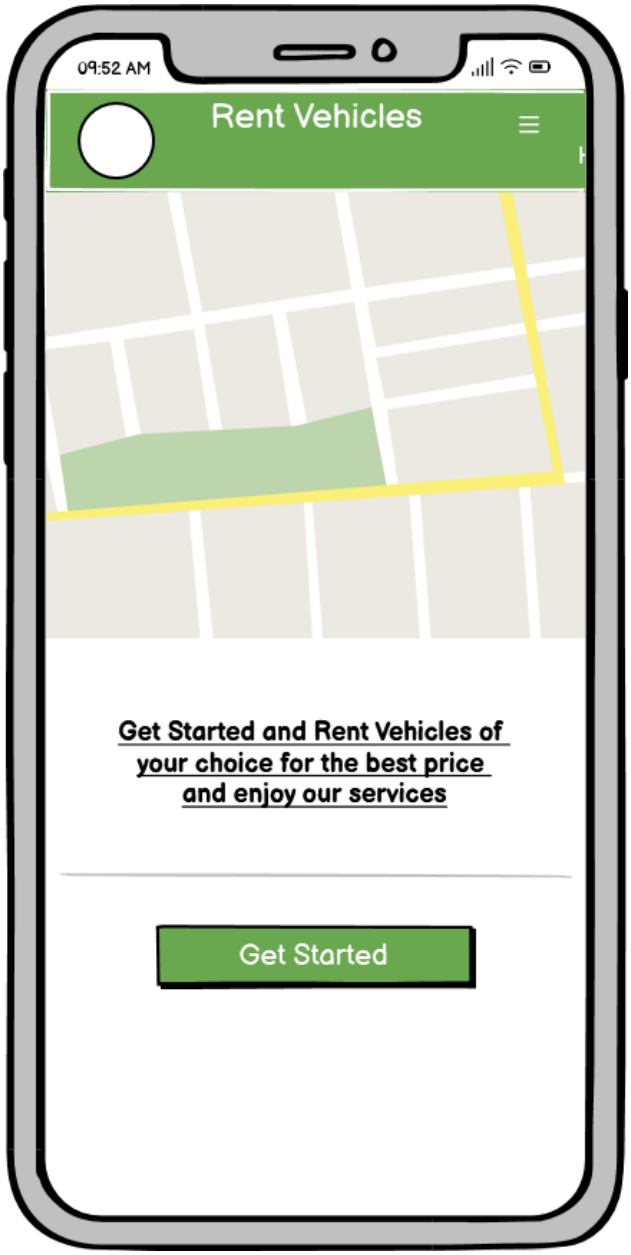

09:52 AM
Rent Vehicles
Get Started and Rent Vehicles of your choice for the best price and enjoy our services
Get Started

Figure 18: Rent Vehicle Prototype

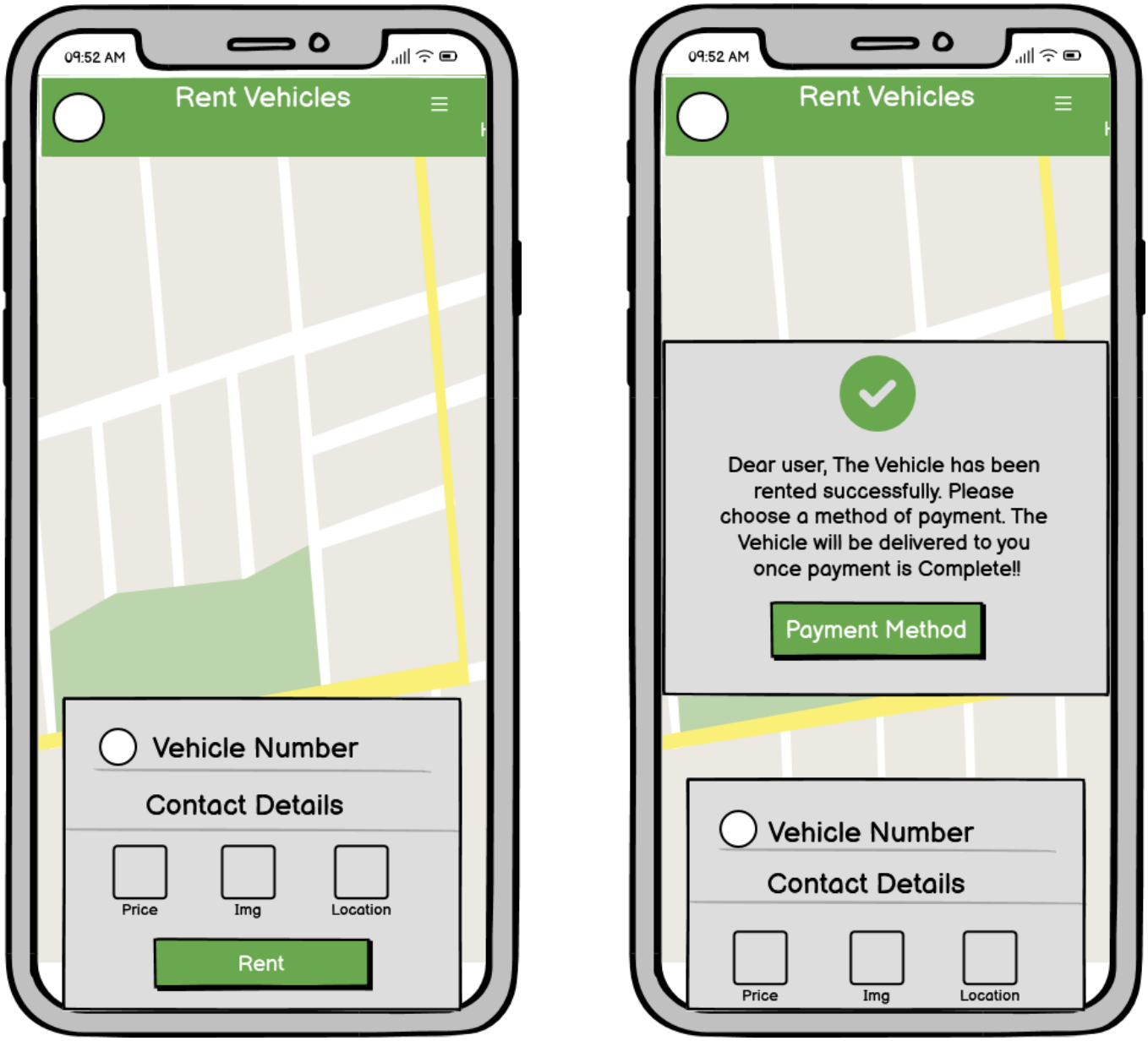


Figure 19: Rent Vehicle Prototype

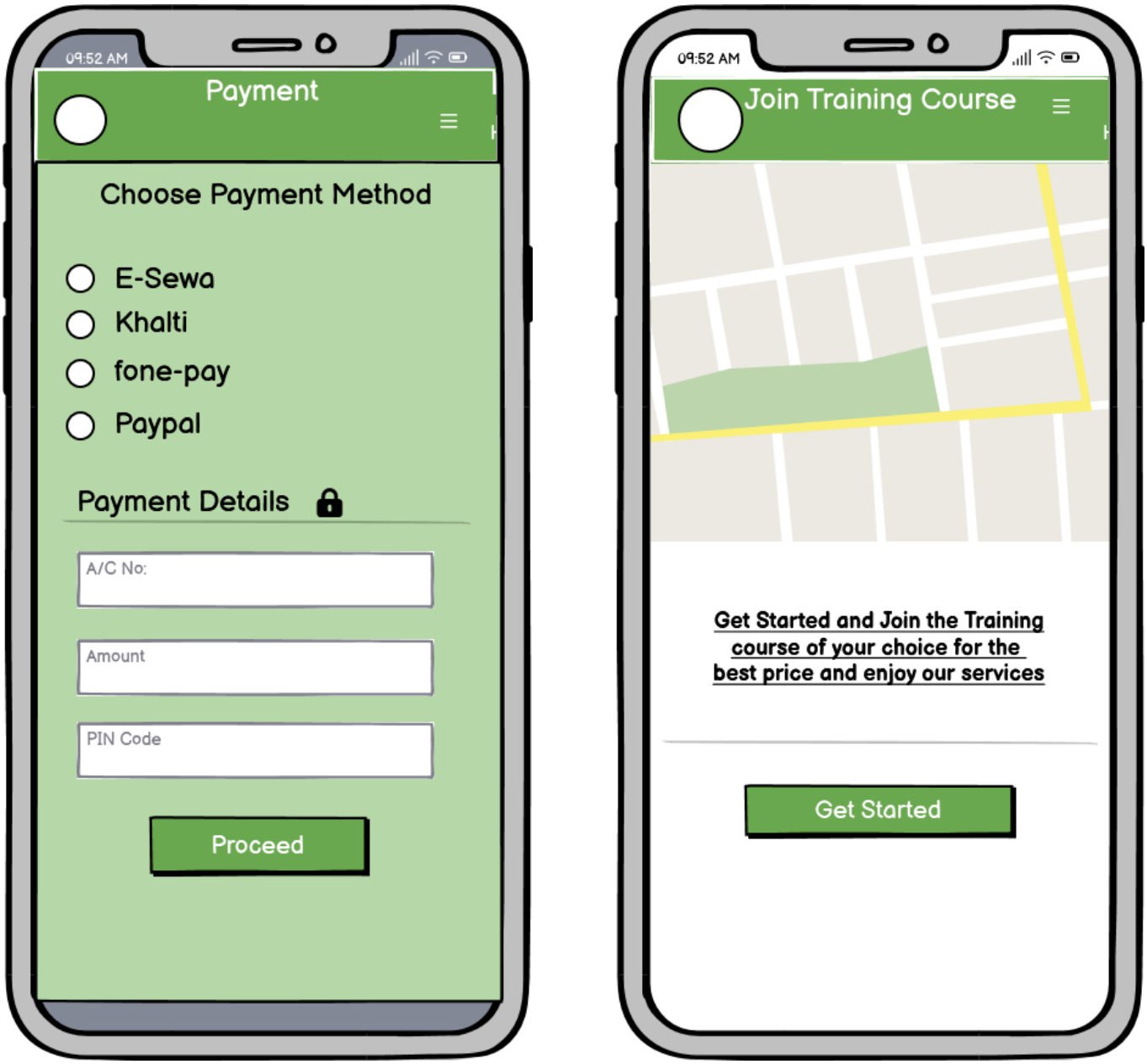


Figure 20: Rent Vehicles Prototype

8.4. Join Traning Course

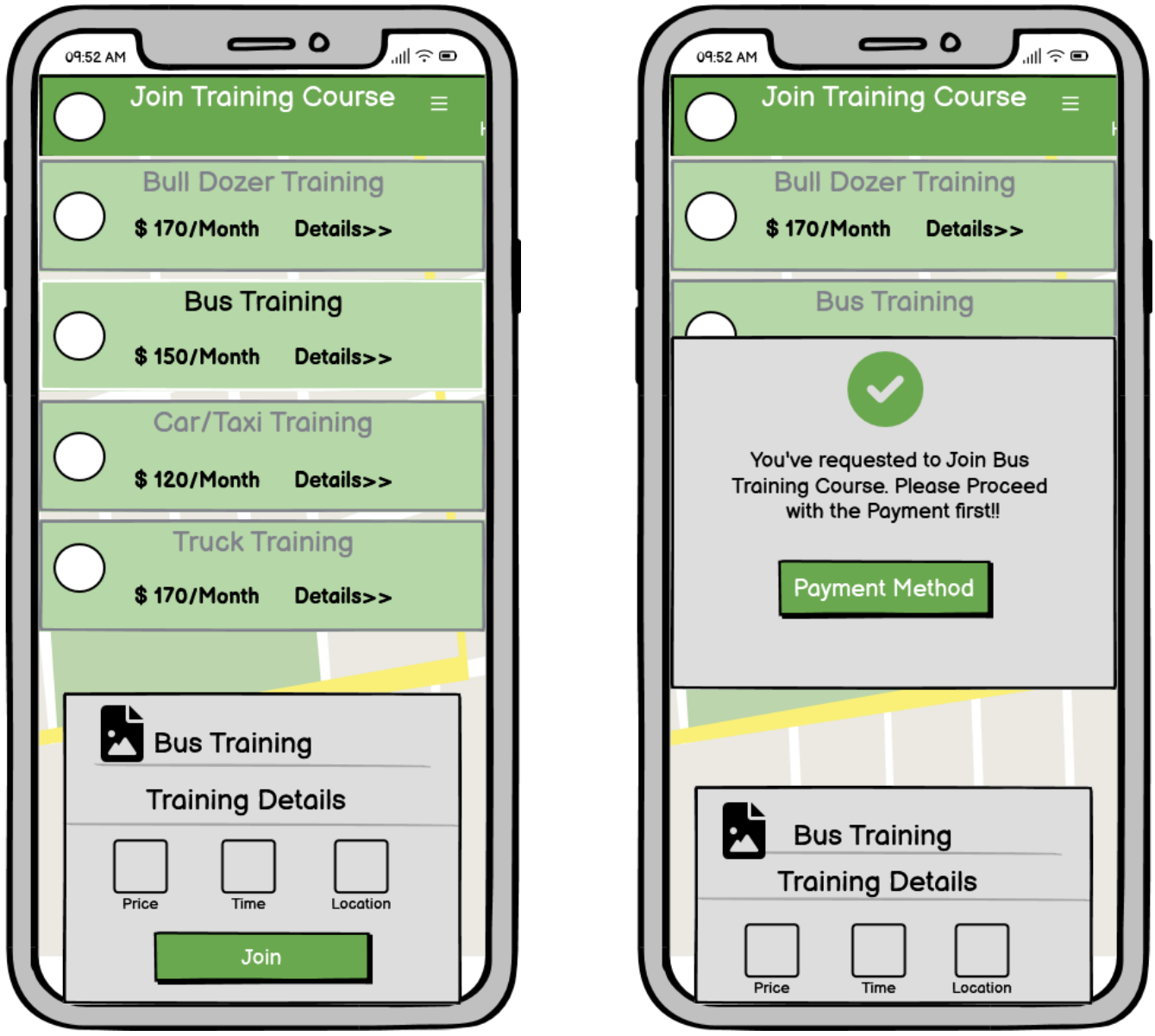


Figure 21: Join Training Course Prototype

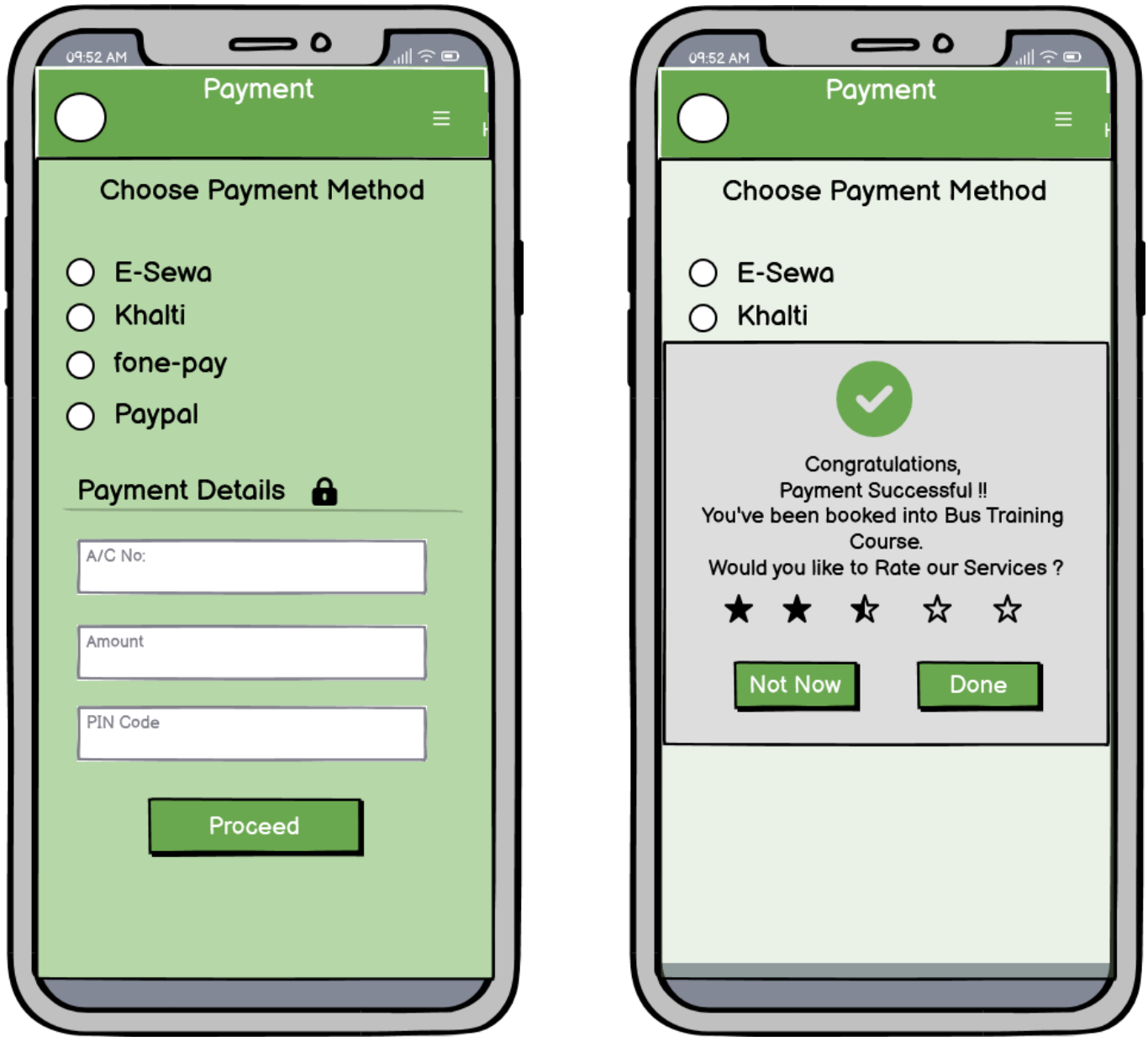


Figure 22: Join Training Course Prototype

9. Conclusion

The goal of this assignment was to further design the “Allgemein” transportation system from where we had left off before. In the last semester, we made SRS document, Project charter, DFD, Structure chart and module Specification. This time we made many UML diagrams to visualize the internal working of the system more clearly. To visualize the structural part of the software, we created use-case diagram and class-diagram and to visualize the behavioural part of the software, we created sequence diagram and collaboration diagram for one use-case.

In this coursework, we were supposed to design system for “Allgemein” transportation application which encompasses three essential services like Cab Booking, Vehicle Hiring and Join Training Course and works throughout every city of Nepal. I created different diagrams to visualize working of the system over a period of time. Once that was done, I created gantt chart explaining the software development process and developed a case-scenario.

Throughout this coursework, I learned many things like how to visualize entire system of the software. I also gained expertise in project management and creating UML diagrams. As a result, the effort to finish this cousework was a success.

10. References

Atlassian. (2023, April 25). *What are Gantt Charts?* Retrieved from https://www.atlassian.com/: https://www.atlassian.com/agile/project-management/ganttchart#:~:text=a%20Gantt%20chart%3F-,A%20Gantt%20chart%20is%20a%20project%20management%20tool%20that%20illustrates,schedule%20bars%20that%20visualize%20work.

en.wikipedia.org. (2023, February 4). *Class diagram*. Retrieved from en.wikipedia.org: https://en.wikipedia.org/wiki/Class_diagram

javatpoint.com. (2008, May 18). *Use-Case Model*. Retrieved from www.javatpoint.com: https://www.javatpoint.com/use-case-model

visual-paradigm.com. (2019, January 11). *What is Sequence Diagram ?* Retrieved from www.visual-paradigm.com: https://www.visual-paradigm.com/guide/uml-unifiedmodeling-language/what-is-sequence-diagram/

www.interviewbit.com. (2001, March 14). *Incremental Model in Software Engineering*. Retrieved from www.interviewbit.com: https://www.interviewbit.com/blog/incremental-model/

www.techtarget.com. (2021, January 11). *Unit Testing*. Retrieved from www.techtarget.com: https://www.techtarget.com/searchsoftwarequality/definition/unit-testing