LONDON METROPOLITAN UNIVERSITY

islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title

CS5002NI SOFTWARE ENGINEERING

Assessment Weightage & Type

35% Individual Coursework

Year and Semester

2022-23 Spring

Student Name: Arbaaz Alam

London Met ID: 22015655

College ID: NP01CP4S220108

Assignment Due Date: 04/27/2023

Assignment Submission Date: 04/27/2023

Word Count: 3985

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1 Background

“Allgemein” is an establishment that has been ruling the entertainment industry for a decade. Its source of income is from projects related to the entertainment industry like movie making, documentary filming, music and game production etc. In order to grow the company, ‘Allegmein ’wants to established the transportation system. In order to support their business, they have purchased a sizable number of vehicles. The cars range from regular cabs to transport and construction vehicles like bulldozers and cargo trucks. They aim to concentrate especially on two services. The first step is to offer taxi services, where customers can reserve a taxi to take them from one location to another. This is valid for both traveling within Kathmandu and between states. Second, they intend to offer automobile rentals for machines like bulldozers and freight trucks. Customers can rent the automobiles for a specified amount of time after making a deposit of a particular amount of money and providing identity document. Customer can join training course to learn various course of driving cab to heavy vehicles. So that the necessary information may be extracted whenever needed, it is necessary to track every record of transaction, client information, and vehicle information.

In the end of this coursework project a proto type of this project will be created will help to know test out the feature required. In order to create prototype, Gantt Chart must be created to allocated the task according to time required. Use case diagram is created of the system to know the object and actors which will used in the system. Sequence and collaboration diagram is created to show how message will communicated between actors and objects. Expanded and high-level description of use cases will be written to know about each use case.

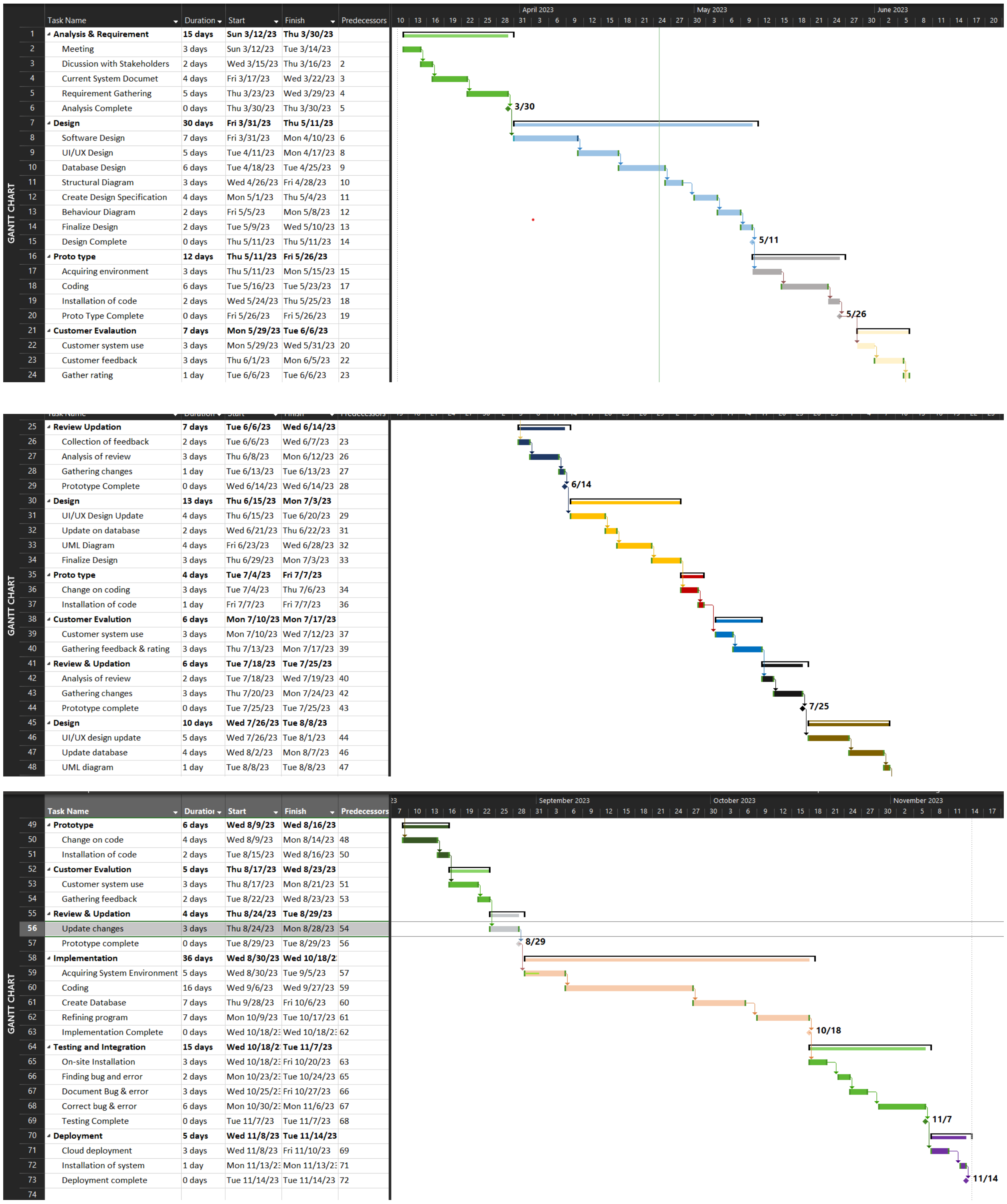
1.2 Aim and Objectives

In order to make ‘Allegmein’s’ transportation industry well establish, the main aim of this project is to reflect on our learning and create best software design for transportation industry using idea, creativity and knowledge as well as know business requirement of business. The main objectives of this projects are listed below:

- To reflect on our learning and create Gantt Chart to allocated task with deadline.
- To address the requirement and create use case diagram to show relation between use case and actors.
- To write expanded and high-level description to know about use case's purpose.
- To create collaboration and sequence diagram to know how actors and object communicate.
- To choose best architecture, methodology for the software and create proto type of the software.

2. Gantt Chart

Gantt charts are one of the many project management tools. They present in one chart all the tasks in a project. A Gantt chart is a horizontal bar chart that was created in 1917 as a production management tool by American engineer and social scientist Henry L. Gantt. A Gantt chart, which is frequently used in project management, offers a pictorial representation of a timetable that may be used to plan, organize, and track work in a project (Lutkevich, 2021).



	Task Name	Duration	Start	Finish	Predecessors
1	**Analysis & Requirement**	**15 days**	**Sun 3/12/23**	**Thu 3/30/23**	
2	Meeting	3 days	Sun 3/12/23	Tue 3/14/23	
3	Dicussion with Stakeholders	2 days	Wed 3/15/23	Thu 3/16/23	2
4	Current System Documet	4 days	Fri 3/17/23	Wed 3/22/23	3
5	Requirement Gathering	5 days	Thu 3/23/23	Wed 3/29/23	4
6	Analysis Complete	0 days	Thu 3/30/23	Thu 3/30/23	5
7	**Design**	**30 days**	**Fri 3/31/23**	**Thu 5/11/23**	
8	Software Design	7 days	Fri 3/31/23	Mon 4/10/23	6
9	UI/UX Design	5 days	Tue 4/11/23	Mon 4/17/23	8
10	Database Design	6 days	Tue 4/18/23	Tue 4/25/23	9
11	Structural Diagram	3 days	Wed 4/26/23	Fri 4/28/23	10
12	Create Design Specification	4 days	Mon 5/1/23	Thu 5/4/23	11
13	Behaviour Diagram	2 days	Fri 5/5/23	Mon 5/8/23	12
14	Finalize Design	2 days	Tue 5/9/23	Wed 5/10/23	13
15	Design Complete	0 days	Thu 5/11/23	Thu 5/11/23	14
16	**Proto type**	**12 days**	**Thu 5/11/23**	**Fri 5/26/23**	
17	Acquiring environment	3 days	Thu 5/11/23	Mon 5/15/23	15
18	Coding	6 days	Tue 5/16/23	Tue 5/23/23	17
19	Installation of code	2 days	Wed 5/24/23	Thu 5/25/23	18
20	Proto Type Complete	0 days	Fri 5/26/23	Fri 5/26/23	19
21	**Customer Evalaution**	**7 days**	**Mon 5/29/23**	**Tue 6/6/23**	
22	Customer system use	3 days	Mon 5/29/23	Wed 5/31/23	20
23	Customer feedback	3 days	Thu 6/1/23	Mon 6/5/23	22
24	Gather rating	1 day	Tue 6/6/23	Tue 6/6/23	23
25	**Review Updation**	**7 days**	**Tue 6/6/23**	**Wed 6/14/23**	
26	Collection of feedback	2 days	Tue 6/6/23	Wed 6/7/23	23
27	Analysis of review	3 days	Thu 6/8/23	Mon 6/12/23	26
28	Gathering changes	1 day	Tue 6/13/23	Tue 6/13/23	27
29	Prototype Complete	0 days	Wed 6/14/23	Wed 6/14/23	28
30	**Design**	**13 days**	**Thu 6/15/23**	**Mon 7/3/23**	
31	UI/UX Design Update	4 days	Thu 6/15/23	Tue 6/20/23	29
32	Update on database	2 days	Wed 6/21/23	Thu 6/22/23	31
33	UML Diagram	4 days	Fri 6/23/23	Wed 6/28/23	32
34	Finalize Design	3 days	Thu 6/29/23	Mon 7/3/23	33
35	**Proto type**	**4 days**	**Tue 7/4/23**	**Fri 7/7/23**	
36	Change on coding	3 days	Tue 7/4/23	Thu 7/6/23	34
37	Installation of code	1 day	Fri 7/7/23	Fri 7/7/23	36
38	**Customer Evalution**	**6 days**	**Mon 7/10/23**	**Mon 7/17/23**	
39	Customer system use	3 days	Mon 7/10/23	Wed 7/12/23	37
40	Gathering feedback & rating	3 days	Thu 7/13/23	Mon 7/17/23	39
41	**Review & Updation**	**6 days**	**Tue 7/18/23**	**Tue 7/25/23**	
42	Analysis of review	2 days	Tue 7/18/23	Wed 7/19/23	40
43	Gathering changes	3 days	Thu 7/20/23	Mon 7/24/23	42
44	Prototype complete	0 days	Tue 7/25/23	Tue 7/25/23	43
45	**Design**	**10 days**	**Wed 7/26/23**	**Tue 8/8/23**	
46	UI/UX design update	5 days	Wed 7/26/23	Tue 8/1/23	44
47	Update database	4 days	Wed 8/2/23	Mon 8/7/23	46
48	UML diagram	1 day	Tue 8/8/23	Tue 8/8/23	47
49	**Prototype**	**6 days**	**Wed 8/9/23**	**Wed 8/16/23**	
50	Change on code	4 days	Wed 8/9/23	Mon 8/14/23	48
51	Installation of code	2 days	Tue 8/15/23	Wed 8/16/23	50
52	**Customer Evalution**	**5 days**	**Thu 8/17/23**	**Wed 8/23/23**	
53	Customer system use	3 days	Thu 8/17/23	Mon 8/21/23	51
54	Gathering feedback	2 days	Tue 8/22/23	Wed 8/23/23	53
55	**Review & Updation**	**4 days**	**Thu 8/24/23**	**Tue 8/29/23**	
56	Update changes	3 days	Thu 8/24/23	Mon 8/28/23	54
57	Prototype complete	0 days	Tue 8/29/23	Tue 8/29/23	56
58	**Implementation**	**36 days**	**Wed 8/30/23**	**Wed 10/18/2**	
59	Acquiring System Environment	5 days	Wed 8/30/23	Tue 9/5/23	57
60	Coding	16 days	Wed 9/6/23	Wed 9/27/23	59
61	Create Database	7 days	Thu 9/28/23	Fri 10/6/23	60
62	Refining program	7 days	Mon 10/9/23	Tue 10/17/23	61
63	Implementation Complete	0 days	Wed 10/18/2	Wed 10/18/2	62
64	**Testing and Integration**	**15 days**	**Wed 10/18/2**	**Tue 11/7/23**	
65	On-site Installation	3 days	Wed 10/18/2	Fri 10/20/23	63
66	Finding bug and error	2 days	Mon 10/23/2	Tue 10/24/23	65
67	Document Bug & error	3 days	Wed 10/25/2	Fri 10/27/23	66
68	Correct bug & error	6 days	Mon 10/30/2	Mon 11/6/23	67
69	Testing Complete	0 days	Tue 11/7/23	Tue 11/7/23	68
70	**Deployment**	**5 days**	**Wed 11/8/23**	**Tue 11/14/23**	
71	Cloud deployment	3 days	Wed 11/8/23	Fri 11/10/23	69
72	Installation of system	1 day	Mon 11/13/2	Mon 11/13/2	71
73	Deployment complete	0 days	Tue 11/14/23	Tue 11/14/23	72
74					

Figure 1 Gantt Chart

Using Prototype model this Gantt chart was created. Prototype model was used in this system because this system requires a lot of interaction with the end users. Since in this methodology a working model of the system is provided, the users get a better understanding of the system being developed. Using it prototype of system are created and if the prototype gets approved them finally development/implementation stage is started.

3. Behavioural Diagram

The components of a behavior diagram are depicted as being time-dependent and communicating notions that are dynamic. It resembles the links between the verbs used in the English language to express time passing. These diagrams' components mimic verbs in real languages, and the connections between them frequently show how time has passed (greg, 2021). It is part of Unified Modeling Language(UML) which helps project teams communicate, explore potential design, and validate the architecture design of the software.

3.1. Use Case Diagram

Use case diagrams are used to represent the proposed system's functional requirements. Use case diagrams show the functional requirements, functions, operations, or capabilities that a system must be able to accomplish. The nonfunctional need is not covered by the use case diagram.

3.2.1. Components of Use Case Diagram:

- Actor: It interacts with the functionality provided by system. It can be person, system, office. For example, Customer, Admin.
- Use Case: Function or Operation to be performed by the system. Example: Make Order, Make Payment.
- System Boundary Box: All use cases are included within system boundary box and actors are placed around system boundary box.
- Relationship Between Actor and Use Case: include, extend, generalization.

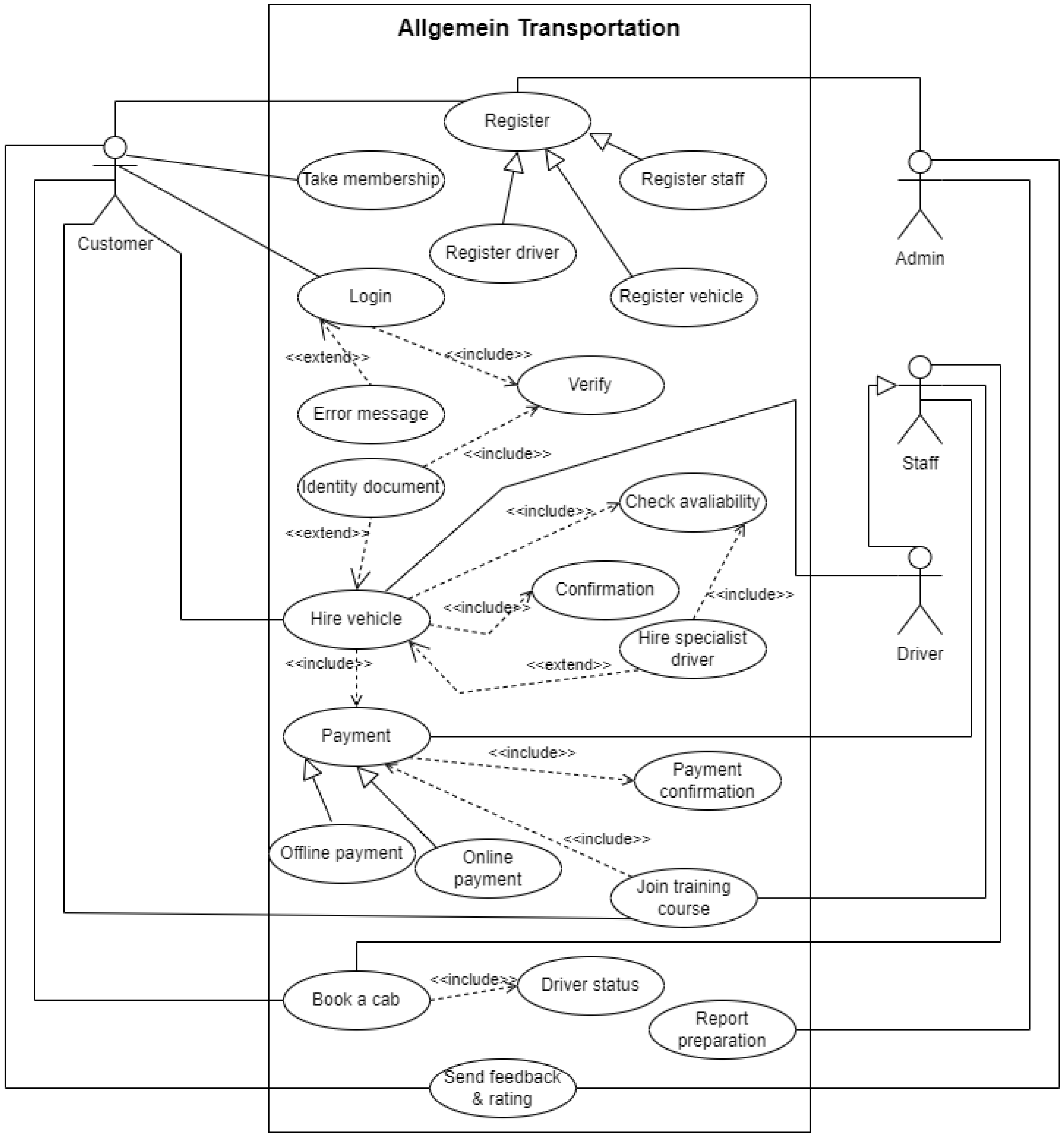


Figure 2 Use Case Diagram

In the above figure use case diagram of Allgemein Transportation. There are four actors in this use case including customer, admin, driver and staff. There are 21 uses cases including main functionality like hire a vehicle, book a cab, take a membership, payment, join training course and many others.

3.2. High Level Use Case Description

For each Use Case, it concentrates on giving a short description of each process as it actually happens. The high-level description is simply a summary description of the task, written as unstructured text a paragraph or two in length.

- Use case: Register

Actor: Customer, Admin

Description: Customer, Admin, Driver are register to system using it. Customer using GUI interface register themselves whereas admin register vehicle, staff and driver is also registered by admin.

- Use case: Register Staff

Actor: Admin

Description: When staff is chosen to work in the company the admin register them to the system to maintain records of them.

- Use case: Register Driver

Actor: Admin

Description: When driver is selected to work for the company the admin registers them to the system or they can register themselves to maintain records of them.

- Use case: Register Vehicle

Actor: Admin

Description: When Vehicle is brought to work for the company the admin registers them to the system or they can register themselves to maintain records of them.

- Use case: Take membership

Actor: Customer

Description: To use the features of the app the customer must be a member of the system and take membership by registering themselves to the system.

- Use case: Login

Actor: Customer: Customer

Description: Member user must login to system before using any feature.

- Use case: Verify

Actor: System

Description: The member user credential is verified if the credentials are right, they are logged in whereas if it wrong then error message is shown. It also verifies whether the license document provide right.

- Use case: Error message

Actor: System

Description: If the credentials are wrong, error message is shown.

- Use case: License document

Actor: Customer

Description: In order to hire a vehicle, the customer must provide their correct document to system.

- Use case: Book cab

Actor: Customer, Staff

Description: Customer can book a cab near them, in order to book cab, they chose vehicle and driver sends driver status.

- Use case: Driver status

Actor: Driver

Description: Customer can book a can so driver shares it location and vehicle type to the customer so that they can locate the vehicle.

- Use case: Hire vehicle

Actor: Customer, Driver

Description: Customer can hire vehicle if they have the correct document to drive it and document is not needed if the specialist driver is needed.

- Use case: Hire Specialist driver

Actor: Customer

Description: Customer can hire specialist driver to ride vehicle as customer hires a vehicle.

- Use case: Confirmation

Actor: System

Description: Customer can hire vehicle and special driver as they want and confirmation is sent to customer.

- Use case: Check availability

Actor: Staff

Description: Customer can hire specialist driver and vehicle whereas system checks availability of the vehicle and driver.

- Use case: Payment

Actor: Staff

Description: Customer to hire a vehicle or join a training course they have to pay certain amount whereas when they book a cab they pay to the driver after the service is completed.

- Use case: Payment confirmation

Actor: System

Description: Customer pays when a service is completed the payment confirmation is sent to customer.

- Use case: Join training courses

Actor: Customer, Staff

Description: Customer can join a course as they wish to learn, but before joining a course they have fill their details and send certain payment.

- Use case: Send feedback

Actor: Customer, Admin

Description: Customer sends feedback as well as rating when service is completed.

- Use case: Report Preparation

Actor: Customer: Admin

Description: Admin create a report based on rating and the how the service are used for the future reference.

- Use case: Offline Payment

Actor: Staff

Description: Customer to hire a vehicle or join a training course they have to pay certain amount whereas when they book a cab they pay to the driver after the service is completed using offline payment or either by online payment.

- Use case: Online Payment

Actor: Customer

Description: Customer to hire a vehicle or join a training course they have to pay certain amount using the online payment options.

3.3. Expanded Use Case Description

For each Use Case. Concentrate on giving a description of each process as it actually happens. Expanded use case description is one or several behavior sequences (segments) that describe additional behavior that can incrementally augment the behavior of the base use case.

- Use case: Hire vehicle

Actor: Customer: Customer, Driver

Purpose: To hire a vehicle

Overview: Customer can hire vehicle using company GUI. Customer have to provide correct document to drive themselves or they can hire specialist driver if needed.

Type: Primary, Essential

Typical Course of Events:

Actor Action	System Response
1) Choose hire vehicle option in GUI	
	2) Shows all available vehicle
3) Choose vehicle type	
	4) Ask for license document if driver is not selected
5) Provides license document	
	6) Provides cost of hiring vehicle
7) Pays the total cost	
	8) Payment confirmation is shown

Table 1 Expanded use case description of hire a vehicle

Alternative Courses:

Line 4: Show available specialist driver list

Line 5: If specialist driver is selected then license document details are not asked

Line 11: Payment gets denied, if payments don't reach to company

- Use case: Join training courses

Actor: Customer

Purpose: To join training course

Overview: Customer can join a course as they wish to learn, but before joining a course they have fill their details and send certain payment using the GUI.

Type: Primary, Essential

Typical Course of Events:

Actor Action	System Response
1) Choose join training courses option	
	2) Presents courses available
	3) Display joining course form
4) Fills form by choosing type of training and time	
	5) Provides payment details
6) Pays the certain amount of course.	
	9) Payment confirmation is shown
	10) Provides training details

Table 2 Expanded use case description of join training course

Alternative Courses:

Line 6: If training course is already totally booked shows another time.

Line 9: Payment gets denied, if there is a problem.

3.4. Collaboration Diagram

A collaboration diagram describes a pattern of interaction among objects, it shows the objects participating in the interaction by their links to each other and the messages that they send to each other. Collaboration diagrams are used to show how objects interact to perform the behavior of a particular use case, or a part of a use case (Lewis, 2023). Steps for producing collaboration Diagram are:

- First Use Case Description was studied and Domain Classes was selected.
- Consideration of object of Domain classes contribute in creating collaboration diagram and is affected by the use case process. If any objects are created and use case process obtain any data from objects.
- Expanded use case description was written and nouns in Typical course of events was listed and if process any message then it was made as objects.
- Each object symbol for each domain class object was created and control object and boundary object were added in hire a vehicle collaboration diagram. Boundary object helps to manage screen interaction has same name as use case having addition UI whereas control object helps interact with other objects and boundary object.
- An association line connecting any pair of Objects was drawn to make objects able to send message. After that using arrow, the message was passed from user to UI to controller to other objects.

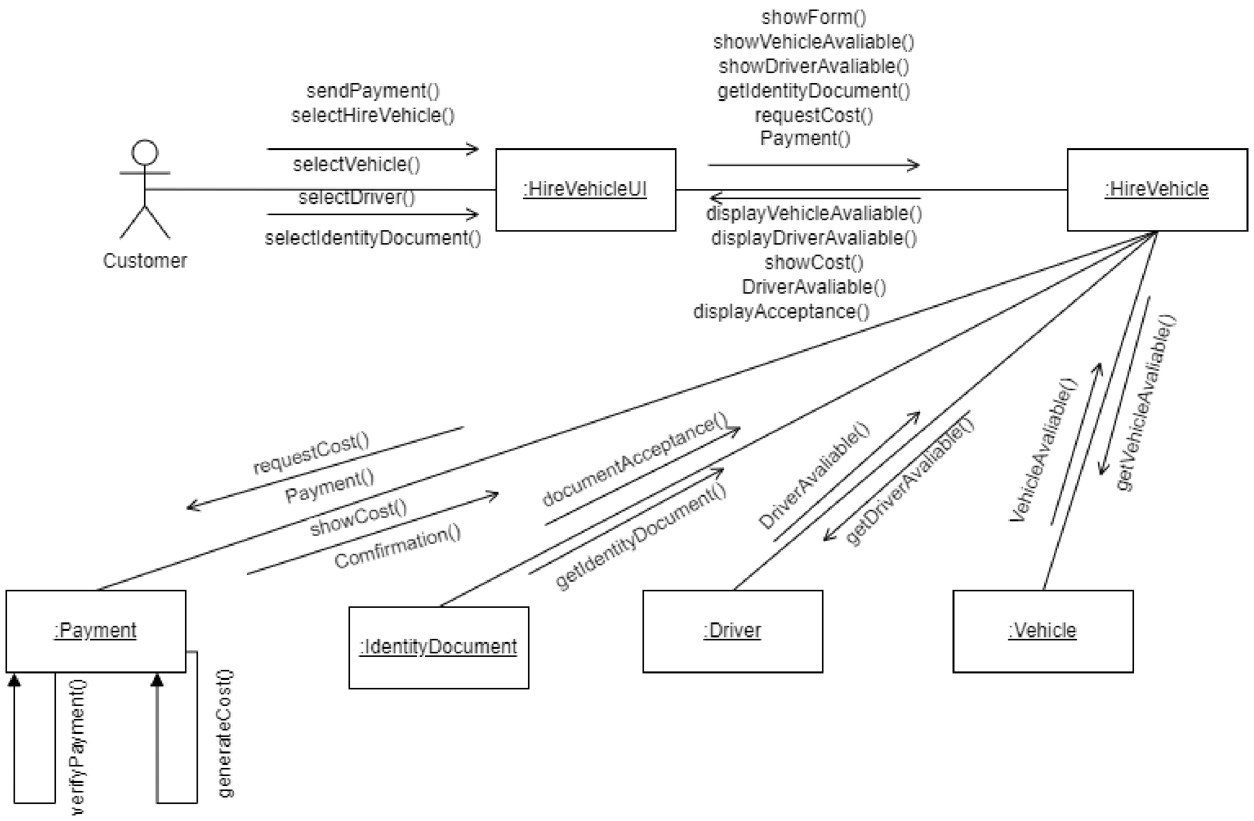


Figure 3 Collaboration diagram

In the above figure the collaboration diagram of the hire a vehicle is shown where customer acts as actor, send message to boundary object HireVehicleUI, HireVehicle sends message to controller object and controller objects sends message to other object like vehicle, driver, payment and identityDocument.

3.5. Sequence Diagram

Sequence diagrams visualize the sequence of messages that is used to perform a specific functionality. Sequence diagram and collaboration diagram show the same information, but simply present it differently. Steps for producing collaboration Diagram are:

- First Use Case Description was studied and Domain Classes was selected.
- Expanded use case description was written and nouns in Typical course of events was listed and if process any message then it was made as objects.

- Each object symbol for each domain class object was created and control object and boundary object were added in hire a vehicle collaboration diagram. Boundary object helps to manage screen interaction has same name as use case having addition UI whereas control object helps interact with other objects and boundary object. Actor was chosen which is customer.
- After creating all the object activation bar was kept in all object including actor when message travel from one object to another. Using the object lifeline all message was passed between object.

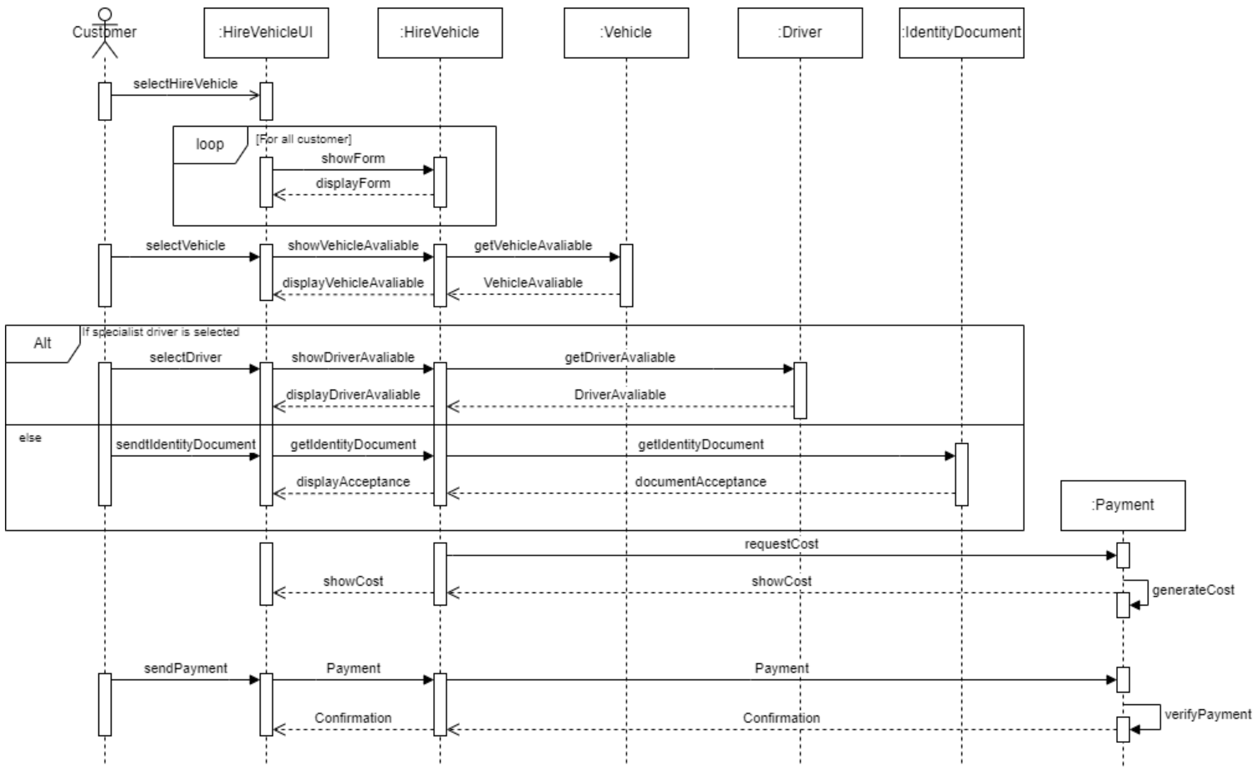


Figure 4 Sequence Diagram

In the above figure the sequence diagram of the hire a vehicle is shown where customer acts as actor, send message to boundary object HireVehicleUI which gets activated when message is sent to it, HireVehicle sends message to controller object and controller objects sends message to other object like vehicle, driver, payment and identityDocument. Whereas loop is used to show form and select driver and identity document is option. Payments gets activated after the driver and vehicle chosen and is used to self-call in the sequence diagram.

4. Structural Diagram

Structure diagrams show the things in the modeled system. In a more technical term, they show different objects in a system. It is depicting the static structure of different elements which is used extensively in documenting the software architecture of a software system (Nishadha, 2022).

4.1. Class Diagram

Class diagram is a structure diagram for designing and modeling software models' software in a high-level abstraction and without having to look at the source code. Class diagram shows names and attributes of a class, connection between the classes and sometimes also the methods of the class. Class diagram is blueprint for a building or a piece of machinery, see the parts used to make it and how they are assembled that provides a sense of orientation, detailed insight into the structure of the system (FONSECA, 2022). Steps to produce class diagram are listed below:

- Using the use case diagram first class were identified. Actors were made class where as other use case directly related to actors were also made classes.
- If a use case contains and extends or include other use cases it was made class.
- The use case that was linked with other use case using extend, include or generalization were made methods of those use cases.
- Attributes were discovered using the class functionality and the relationship between class was produce by understanding class relation with other.
- If a class had a "is a" relationship then inheritance was used, if the classes had "has a" relationship then whole part was used (if classes destroy if other class is destroyed then composition was used and if didn't then aggregation was used) and in other classes association was used as relationship.

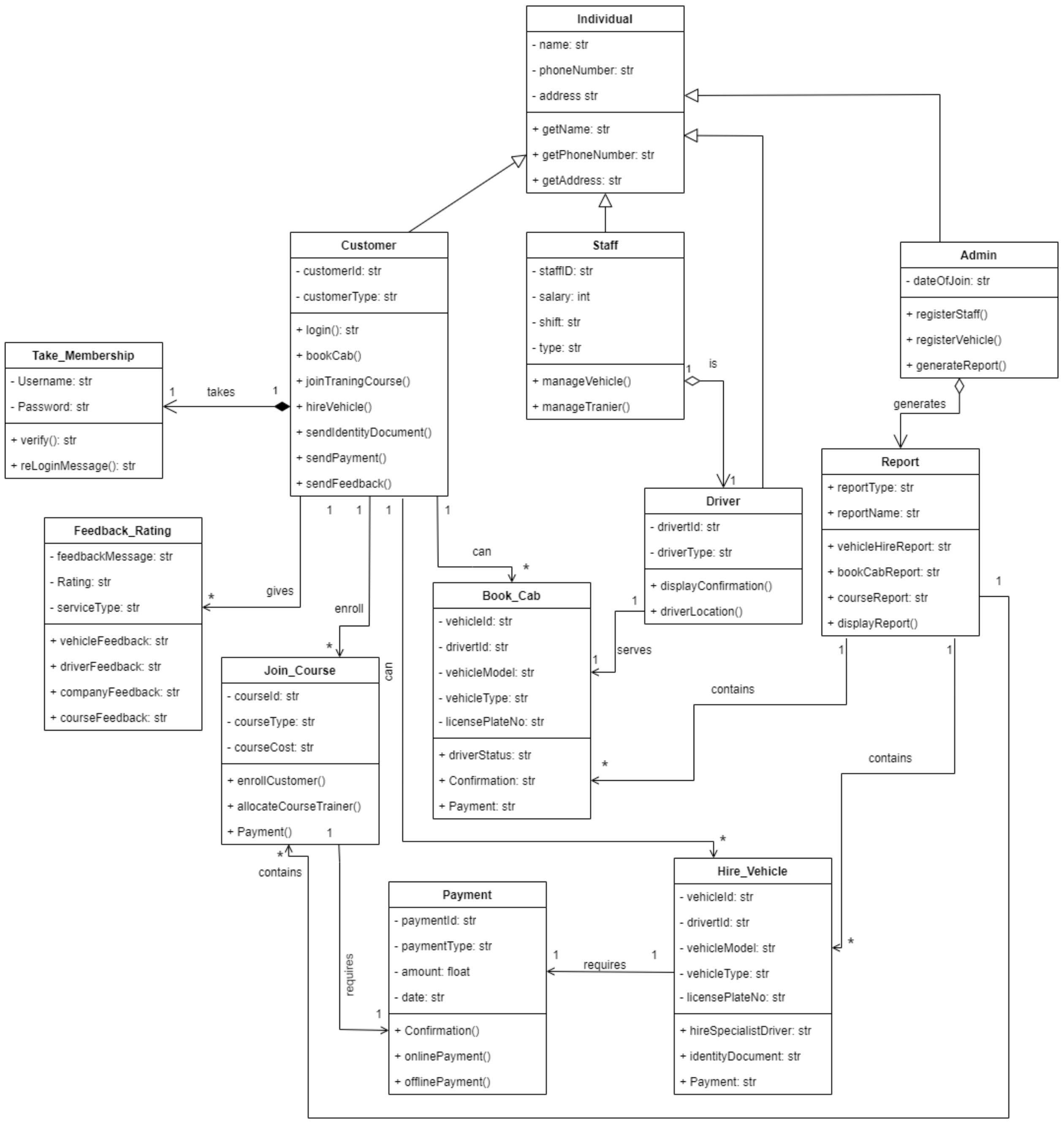


Figure 5 Class Diagram

In the above figure the class diagram of allgemein transportation is shown where individual is parent class of customer, staff, driver, and admin. Customer and take membership has composition relationship. Driver and staff, admin and report have aggregation relationship. All the other class are related by association.

5. Further Development

In order to complete this project methodology, design pattern, architecture pattern, testing and deployment will be carried out. This section will contain how following things are selected.

5.1. Software Development Models

The process of planning, designing, producing, testing, and delivering high-quality software at the lowest cost and ideally in the shortest period of time is known as the Software Development Life Cycle, or SDLC. To do this, software engineering teams must select the best software development model based on the needs of their company, the expectations of their stakeholders, and the project at hand. There are several software development methods, and each has unique benefits and drawbacks (Mleziva, 2020). Types of Software Development Models are:

- Waterfall Model

  Waterfall model is a model used only when the requirements are very well known, clear and fixed. This model is simple and easy to understand and use but it has high amounts of risk and uncertainty.

- Spiral Model

  The Spiral Model focuses on risk assessment and used when costs and risk evaluation is important and when users are unsure of their needs. It is good for large and critical projects whereas project's success is highly dependent on the risk analysis phase.

- Prototype Model

  A prototype is an initial version of a software system that is used to demonstrate concepts, try out design options, and find out more about the problem and its possible solutions. Since in this methodology a working model of the system is provided, the users get a better understanding of the system being whereas leads to implementing and then repairing way of building systems.

5.2. Software design Pattern

Software design pattern provides reusable solution to recurring design problems which provides a template to solve problem that can be re-used in many situations. Design patterns capture solution to problem that is evolved over time and helps to reuse the knowledge of experience software developer over time. Software design Pattern helps to capture solutions which have been applied to certain problems and make it accessible to non-experts in a standard form, make it easier to reuse successful designs and facilitate design modifications, improves design documentation, reduces development time and helps to build flexible and extensible applications. Some types of Software design Pattern are factory pattern, Singleton Pattern and, Prototype Pattern and MVC Design Pattern.

5.2.1. MVC Design Pattern

MVC stand for Model-View-Controller where model is object carrying data, view is visualization of data and controller is data flow from model and view and vice-versa. MCV completely separates the calculations and interface from each other. As components have a low dependency on each other, they are easy to maintain. MCV design pattern is used to made collaboration diagram and sequence diagram. In this model holds all data and methods or processing or calculations used to work with those data, view is the Interface and controller helps to coordinates interactions between view and model.

5.3. Software Architecture Pattern

Software Architecture Pattern is outline that allows you to express and define a structural schema for all kinds of software system, reusable solution and provides predefined set of subsystems, roles and responsibilities, including the rules and roadmap for defining relationship. Knowing each architecture's characteristics, strengths, and weaknesses becomes important for choosing the right one to meet your business objectives. Some type Software Architecture Pattern are:

- Layered Architecture Pattern

  Layered Architecture Pattern is separated into layers of subtasks and they are arranged one above another. Each layer has unique tasks to do and all the layers are independent of one another. Since each layer is independent, one can modify the code inside a layer without affecting others.

- Event-Driven Architecture Pattern

Events are actions. Whenever something happens inside or outside your organization, an event has occurred. These actions can be anything like book a cab, hire vehicle. It is made up of decoupled, single-purpose event processing components that asynchronously receive and process events.

- Microkernel Architecture Pattern

Microkernel Architecture Pattern has two component a core system and several plugin modules. The core system works on minimal functionality to keep the system operational and the plug-in modules are independent components with specialized processing.

- Microservices Architecture Pattern

Microservices Architecture Pattern is pattern to create a number of different tiny programs and then create a new little program every time someone wants to add a new feature. The components are deployed as separate units through an effective, streamlined delivery pipeline. The pattern's benefits are enhanced scalability and a high degree of decoupling within the application

Microservices Architecture Pattern will be used in further development of the system as it helps to develop same components deploy them separately, and tested them without interdependency on any other service component which helps to test separated module like book a cab, hire vehicle. Using this pattern scalability can be achieved and rapid development can be remotely.

5.4. Programming Paradigms

Programming Paradigms is style of programming but is not to a specific programming language, but rather refers to the way you program. It is strategy followed by programming languages when implemented. It is important to choose a programming paradigm to code. There are three types of Programming Paradigms:

- Procedural Programming Paradigms

Procedural Programming Paradigms splits instructions into procedures it accomplish a given task and cause a desired side effect. It is sequential execution steps from top to bottom. For example, C, Pasal etc

- Functional Programming Paradigms

Functional Programming Paradigms composed of programs in short functions. Functions do not modify any values outside the scope of that function and are not affected by values outside their scope. For example, JavaScript, Haskell etc.

- Object-Oriented Programming Paradigms

Object-Oriented Programming Paradigms is most popular programming paradigms which is based on class and objects. It has ability to directly associate real-world business problems in term of code. For example, Java, Ruby etc.

In the project Object-Oriented Programming Paradigms will be used as it helps to reuse code and helps to interact using object and call-in different classes and supports abstraction. It has high security, faster development, its lower cost of development and produce high quality software.

5.5. Testing

Testing refers process of testing source code written. The modules are brought together (Integrated) to form the system and tested and find any error and bug in the system to prevent system crash. In this project various testing will be carried out listed below:

- Unit Testing

Unit testing is a type of software testing that focuses on individual units or components of a software system. So, each component of the system is created and tested for example if hire vehicle component is created it first it will be tested.

- Regression Testing

Regression test is a test that is performed to make sure that previously working functionality still works, after changes elsewhere in the system. Regression testing helps to ensure that changes do not introduce unintended behavior or additional errors.

- Whitebox Testing

Whitebox testing approach that allows testers to inspect and verify the inner workings of a software system its code, infrastructure and integrations with external systems. In this this code is unknown and requirement. It is mostly done by customer.

- Blackbox Testing

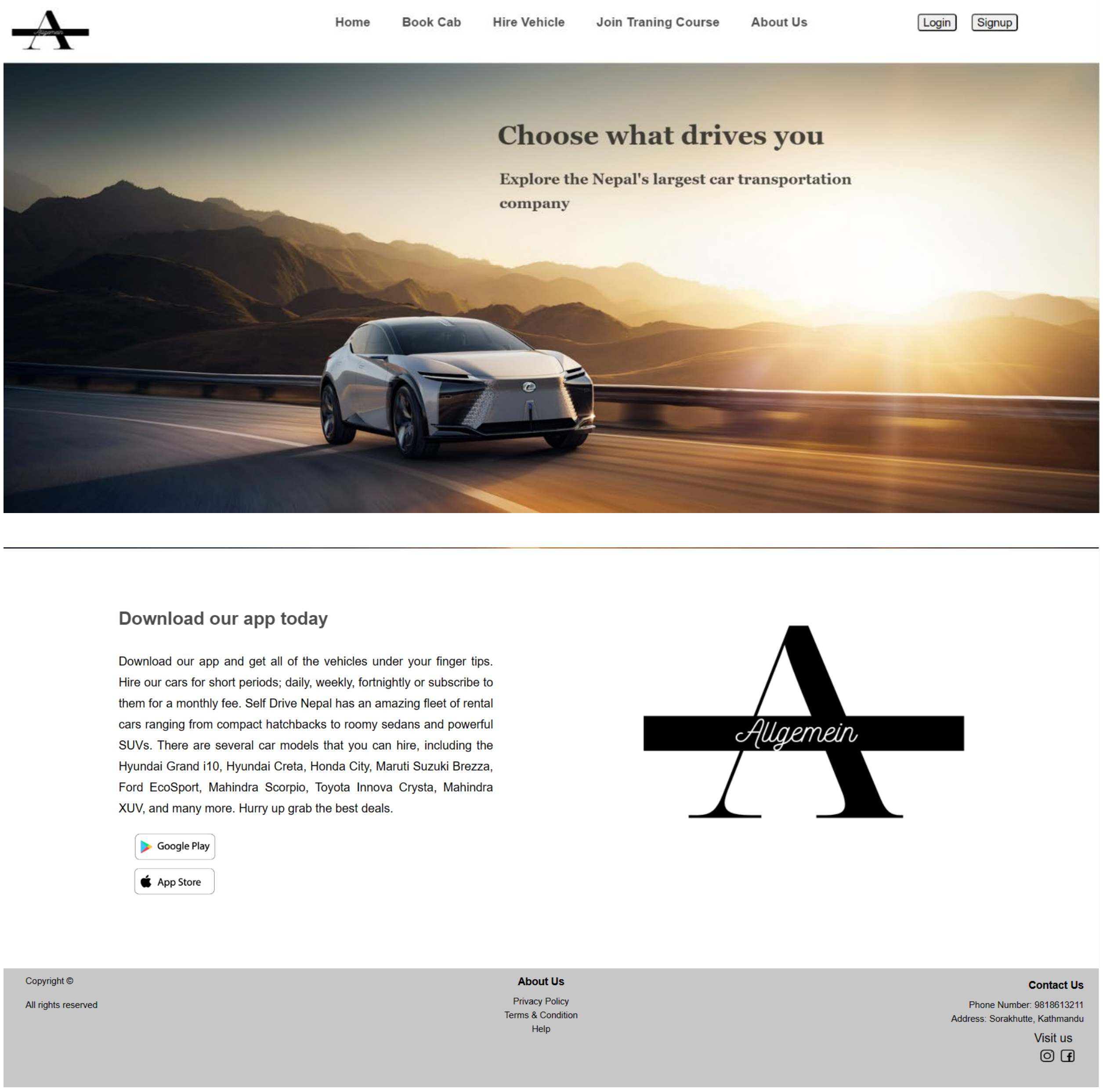
In blackbox testing the functionality of each module is tested with regards to its specification's requirements and its events. Only the correct input/output relationship is inspected. This is main done by knowing all the code and components usually done by developer.

5.6. Deployment

After the development and testing of the system it is really important to carry out deployment which helps to integrate the system in cloud or other data center for use. The customer should be able to use the system as well as the admin. After tested system is carried out for the implementation into a working environment and problems identified in the integration and testing phase are also resolved and better update is carried out. It is required to operate and provide service for the maintenance of the system. The updates and maintenance of system is most important and according to need the system should be flexible, scalable.

6. Prototype

In the prototype of the system, website of the system is created using html, css, javascript without using bootstrap. Prototype helps to know about the components needs to create in the system.



Choose what drives you

Explore the Nepal's largest car transportation company

Download our app today

Download our app and get all of the vehicles under your finger tips. Hire our cars for short periods; daily, weekly, fortnightly or subscribe to them for a monthly fee. Self Drive Nepal has an amazing fleet of rental cars ranging from compact hatchbacks to roomy sedans and powerful SUVs. There are several car models that you can hire, including the Hyundai Grand i10, Hyundai Creta, Honda City, Maruti Suzuki Brezza, Ford EcoSport, Mahindra Scorpio, Toyota Innova Crysta, Mahindra XUV, and many more. Hurry up grab the best deals.

Copyright ©

All rights reserved

About Us

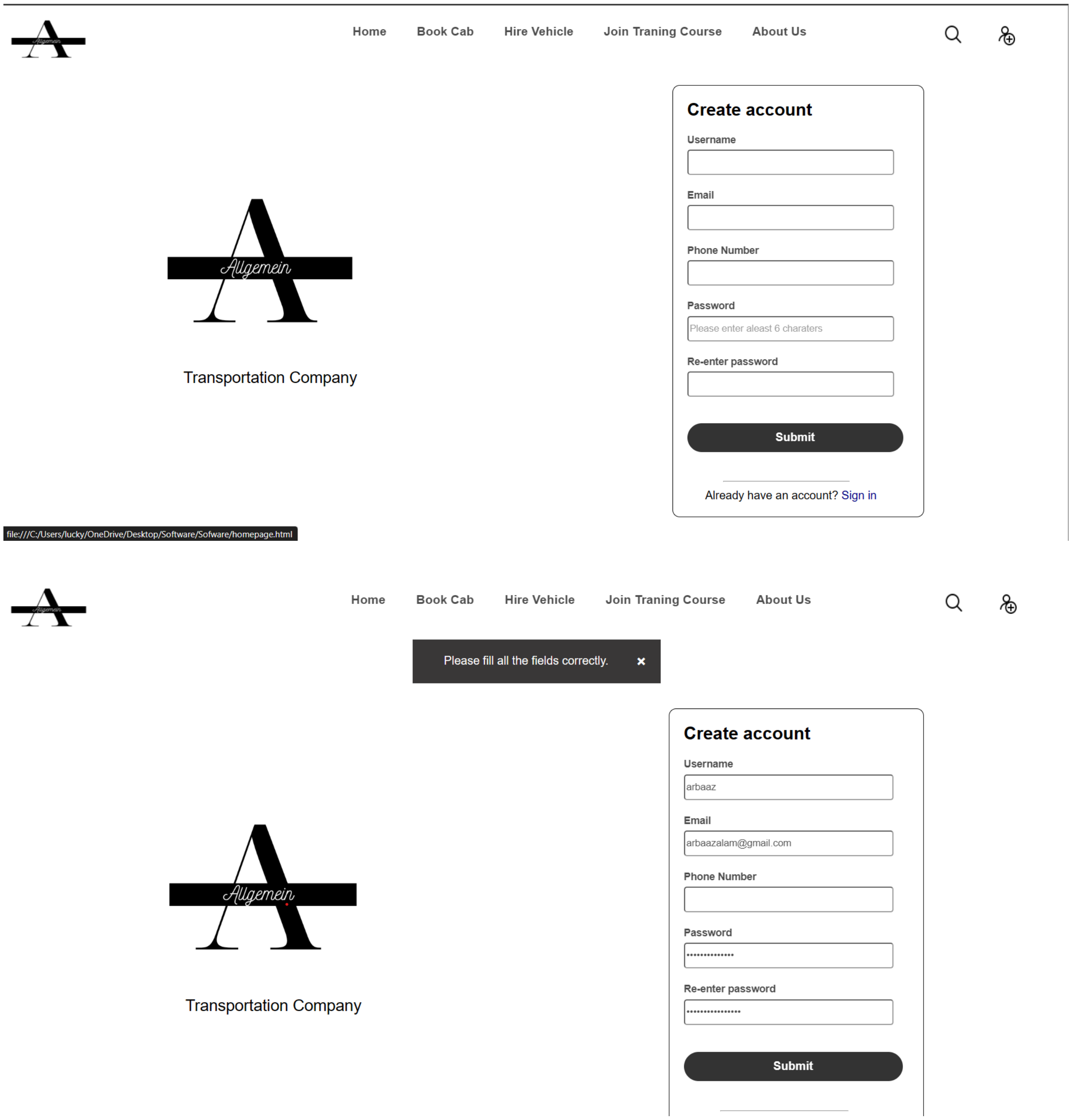
Privacy Policy
Terms & Condition
Help

Contact Us

Phone Number: 9818613211
Address: Sorakhutte, Kathmandu

Figure 6 Prototype login homepage

The above images are the home page which opens first for the user and admin which gives options to user to login and signup.



Create account

Username

Email

Phone Number

Password

Please enter aleast 6 charaters

Re-enter password

Submit

Already have an account? Sign in

file:///C:/Users/lucky/OneDrive/Desktop/Software/Sofware/homepage.html

Please fill all the fields correctly. ×

Create account

Username

arbaaz

Email

arbaazalam@gmail.com

Phone Number

Password

••••••••••••••

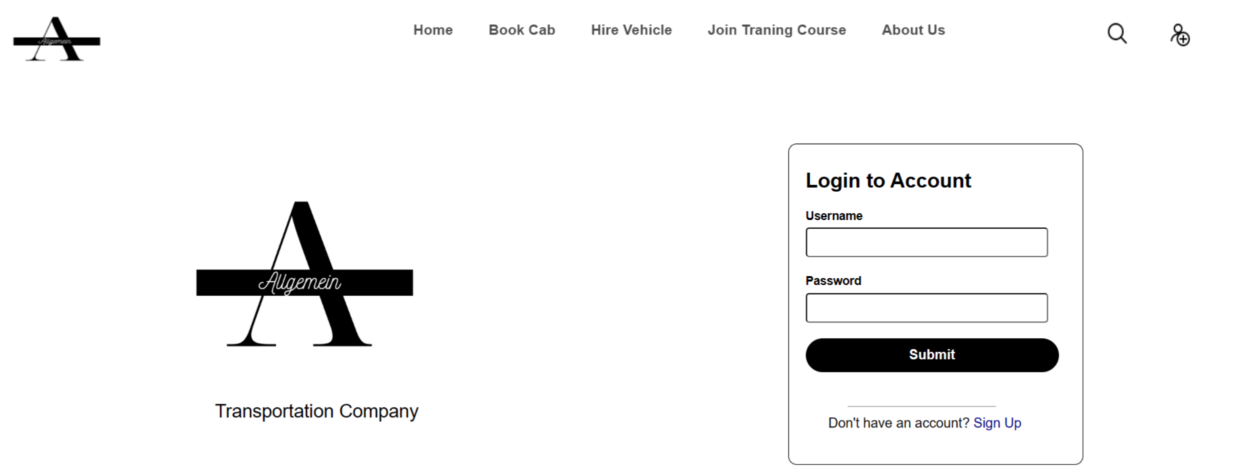
Re-enter password

••••••••••••••••

Submit

Figure 7 Prototype signup page

The above image is the sign-up page where user can sign and if any field are empty while filling the form it shows to fill all the details. When user successfully register, the page is redirected to login page.



The above image is the login page where user can login as well as admin.

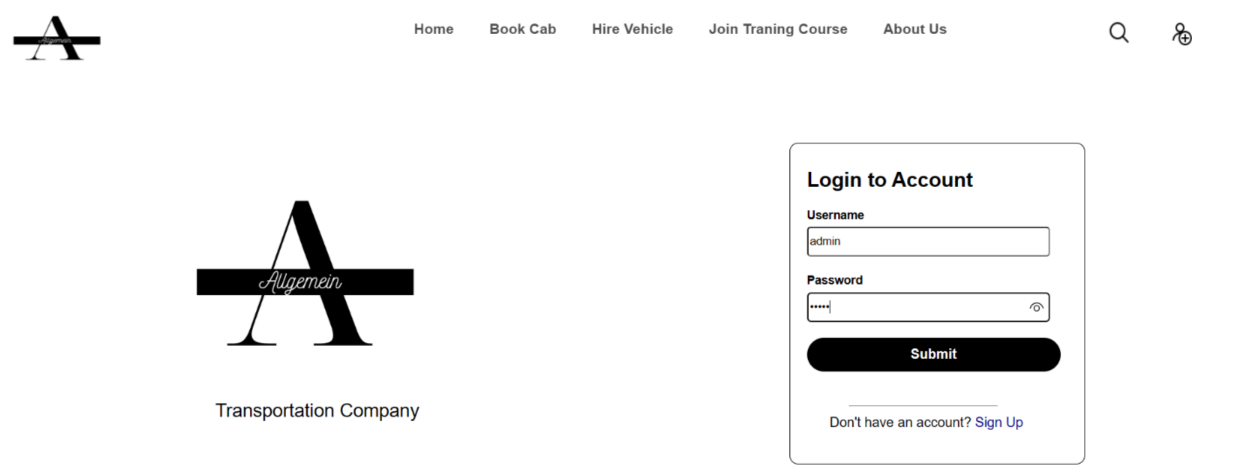


Figure 8 Prototype sign in page

Admin is given fixed credentials when admin login that using that credentials the home page of admin opens when submit button is clicked.

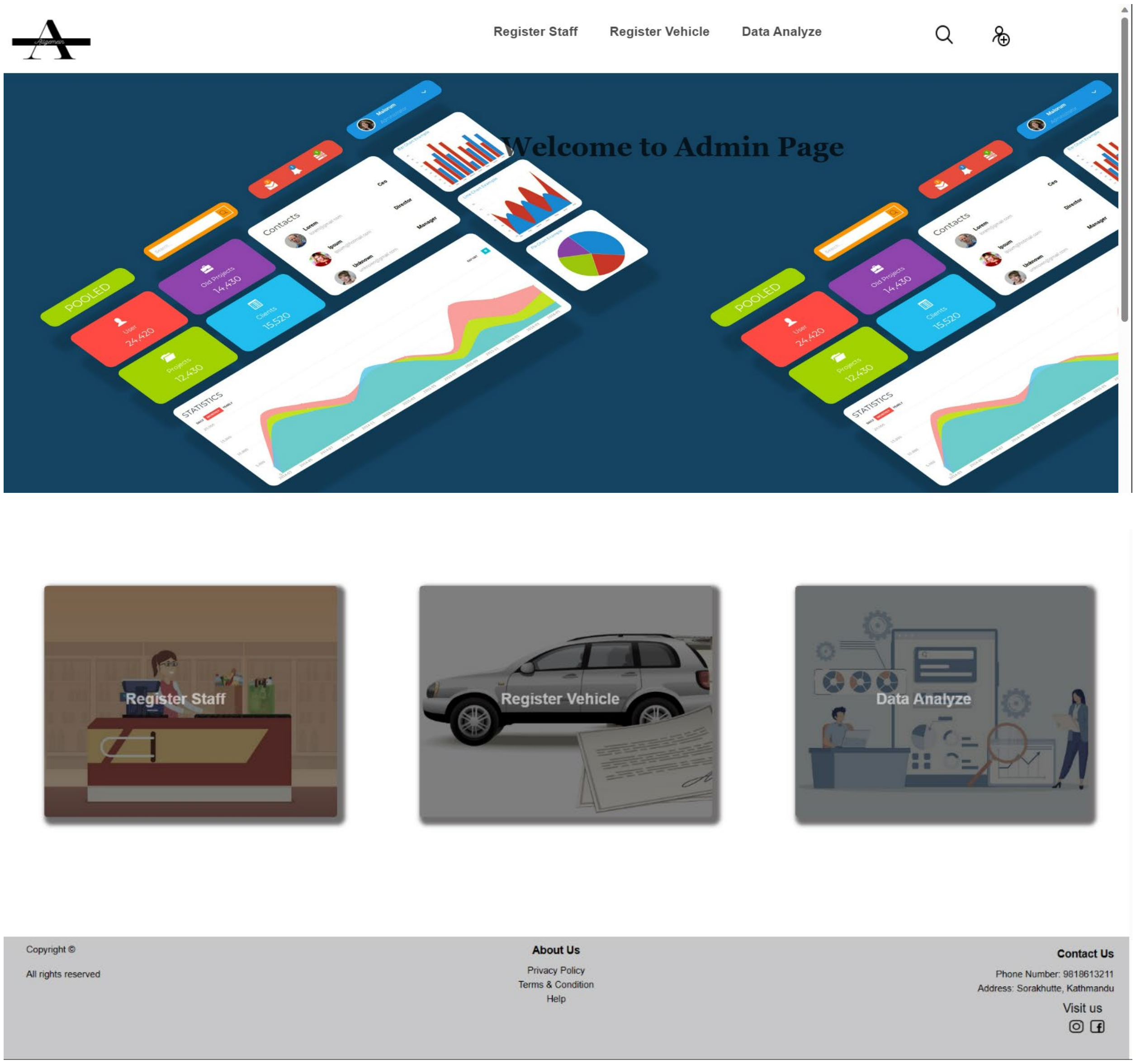


Figure 9 Prototype admin homepage

The above is the admin page where admin can login and register staff, vehicle and generate report.

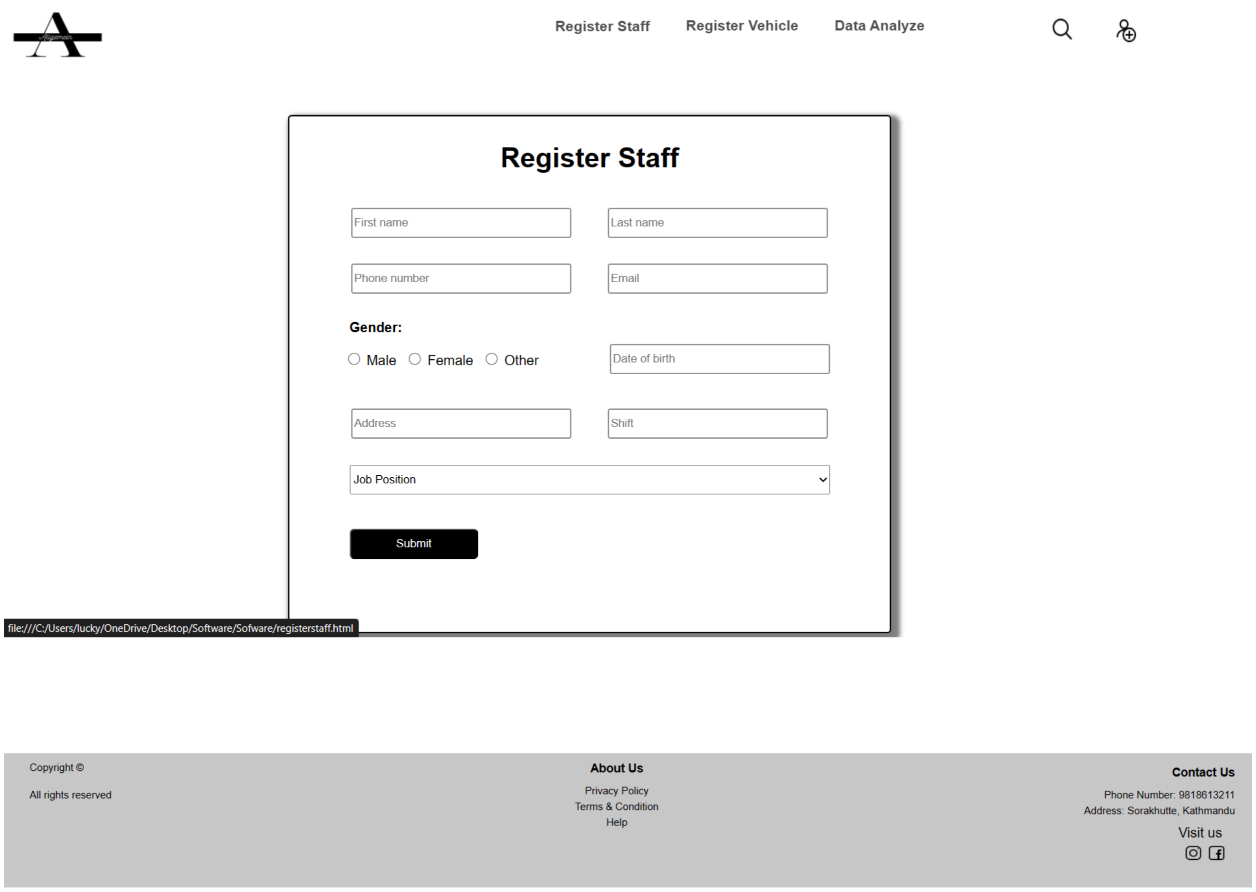
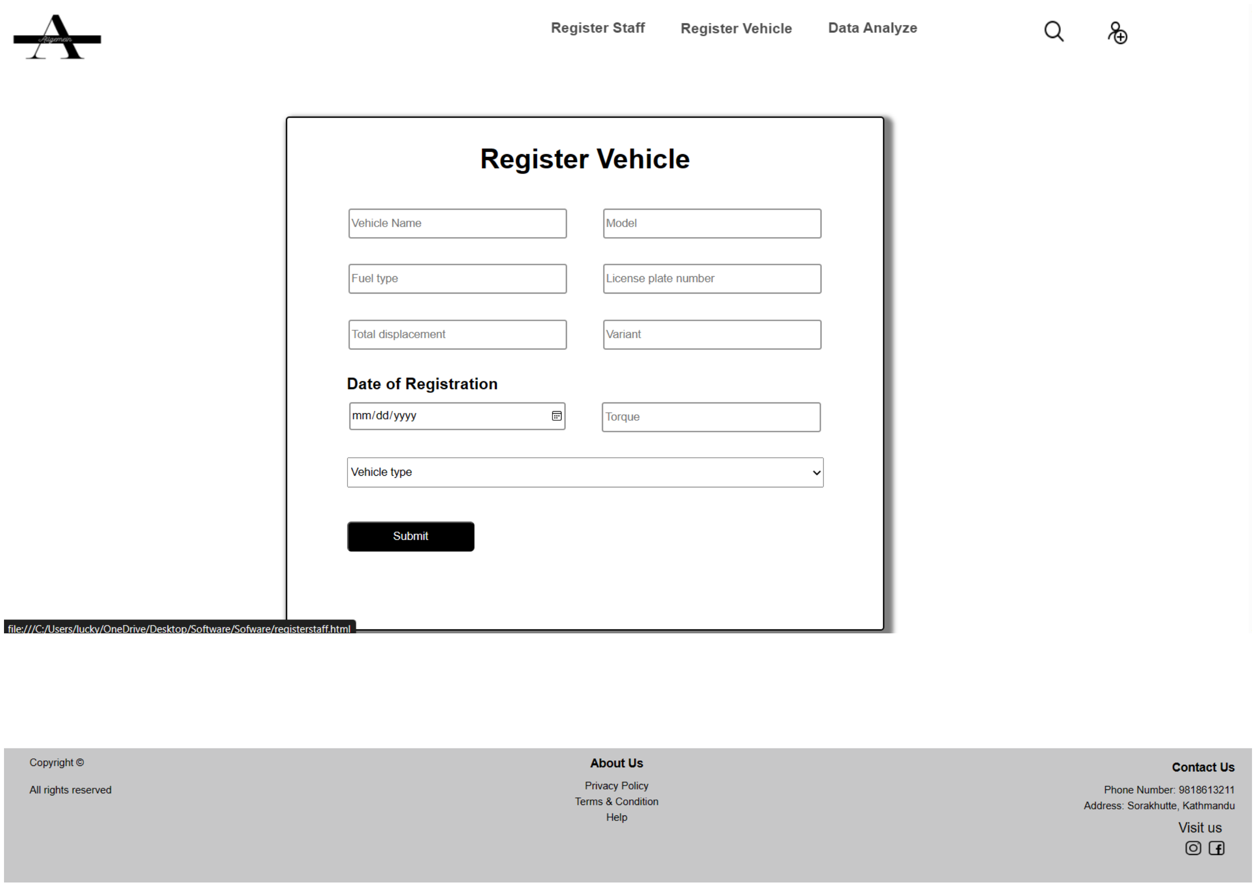


Figure 10 Prototype register staff page

When admin click on register staff this page is open which helps to submit data of staff.



Register Vehicle

Vehicle Name	Model
Fuel type	License plate number
Total displacement	Variant

Date of Registration

mm/dd/yyyy	Torque

Vehicle type

Submit

file:///C:/Users/lucky/OneDrive/Desktop/Software/Sofware/registerstaff.html

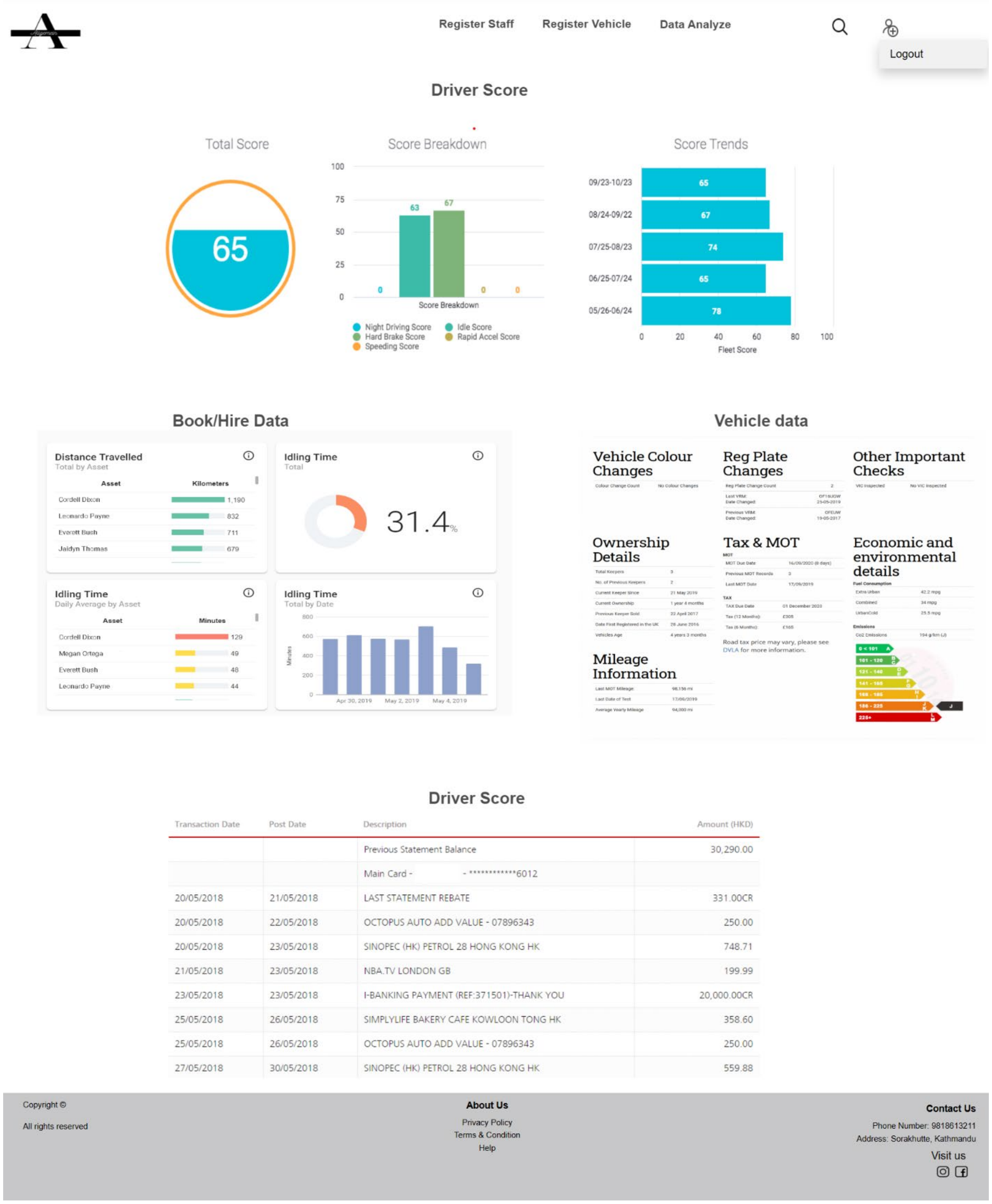
Copyright ©
All rights reserved

About Us
Privacy Policy
Terms & Condition
Help

Contact Us
Phone Number: 9818613211
Address: Sorakhutte, Kathmandu
Visit us

Figure 11 Prototype register Vehicle page

When admin click on register vehicle this page is open which helps to submit data of vehicle.



Driver Score

Book/Hire Data

Distance Travelled
Total by Asset

Asset	Kilometers
Cordell Dixon	1,190
Leonardo Payne	832
Everett Bush	711
Jaidyn Thomas	679

Idling Time
Total

Idling Time
Daily Average by Asset

Asset	Minutes
Cordell Dixon	129
Megan Ortega	49
Everett Bush	48
Leonardo Payne	44

Idling Time
Total by Date

Vehicle data

Vehicle Colour Changes

Colour Change Count	No Colour Changes

Reg Plate Changes

Reg Plate Change Count	2
Last VRM: Date Changed:	OF16UGW 25-05-2019
Previous VRM: Date Changed:	OFEUW 19-05-2017

Other Important Checks

VIC Inspected	No VIC Inspected

Ownership Details

Total Keepers	3
No. of Previous Keepers	2
Current Keeper Since	21 May 2019
Current Ownership	1 year 4 months
Previous Keeper Sold	22 April 2017
Date First Registered in the UK	28 June 2016
Vehicles Age	4 years 3 months

Tax & MOT

MOT

MOT Due Date	16/09/2020 (8 days)
Previous MOT Records	3
Last MOT Date	17/09/2019

TAX

TAX Due Date	01 December 2020
Tax (12 Months):	£305
Tax (6 Months):	£165

Road tax price may vary, please see DVLA for more information.

Economic and environmental details

Fuel Consumption

Extra Urban	42.2 mpg
Combined	34 mpg
UrbanCold	25.5 mpg

Emissions

Co2 Emissions	194 g/km (J)

Mileage Information

Last MOT Mileage:	98,156 mi
Last Date of Test	17/09/2019
Average Yearly Mileage	94,000 mi

Driver Score

Transaction Date	Post Date	Description	Amount (HKD)
		Previous Statement Balance	30,290.00
		Main Card - - ************6012	
20/05/2018	21/05/2018	LAST STATEMENT REBATE	331.00CR
20/05/2018	22/05/2018	OCTOPUS AUTO ADD VALUE - 07896343	250.00
20/05/2018	23/05/2018	SINOPEC (HK) PETROL 28 HONG KONG HK	748.71
21/05/2018	23/05/2018	NBA.TV LONDON GB	199.99
23/05/2018	23/05/2018	I-BANKING PAYMENT (REF:371501)-THANK YOU	20,000.00CR
25/05/2018	26/05/2018	SIMPLYLIFE BAKERY CAFE KOWLOON TONG HK	358.60
25/05/2018	26/05/2018	OCTOPUS AUTO ADD VALUE - 07896343	250.00
27/05/2018	30/05/2018	SINOPEC (HK) PETROL 28 HONG KONG HK	559.88

Copyright ©
All rights reserved

About Us
Privacy Policy
Terms & Condition
Help

Contact Us
Phone Number: 9818613211
Address: Sorakhutte, Kathmandu
Visit us

Figure 12 Prototype report page

Admin can see the report by clicking data analyze on the navigation bar and admin can logout using the logout button which redirect to login page.

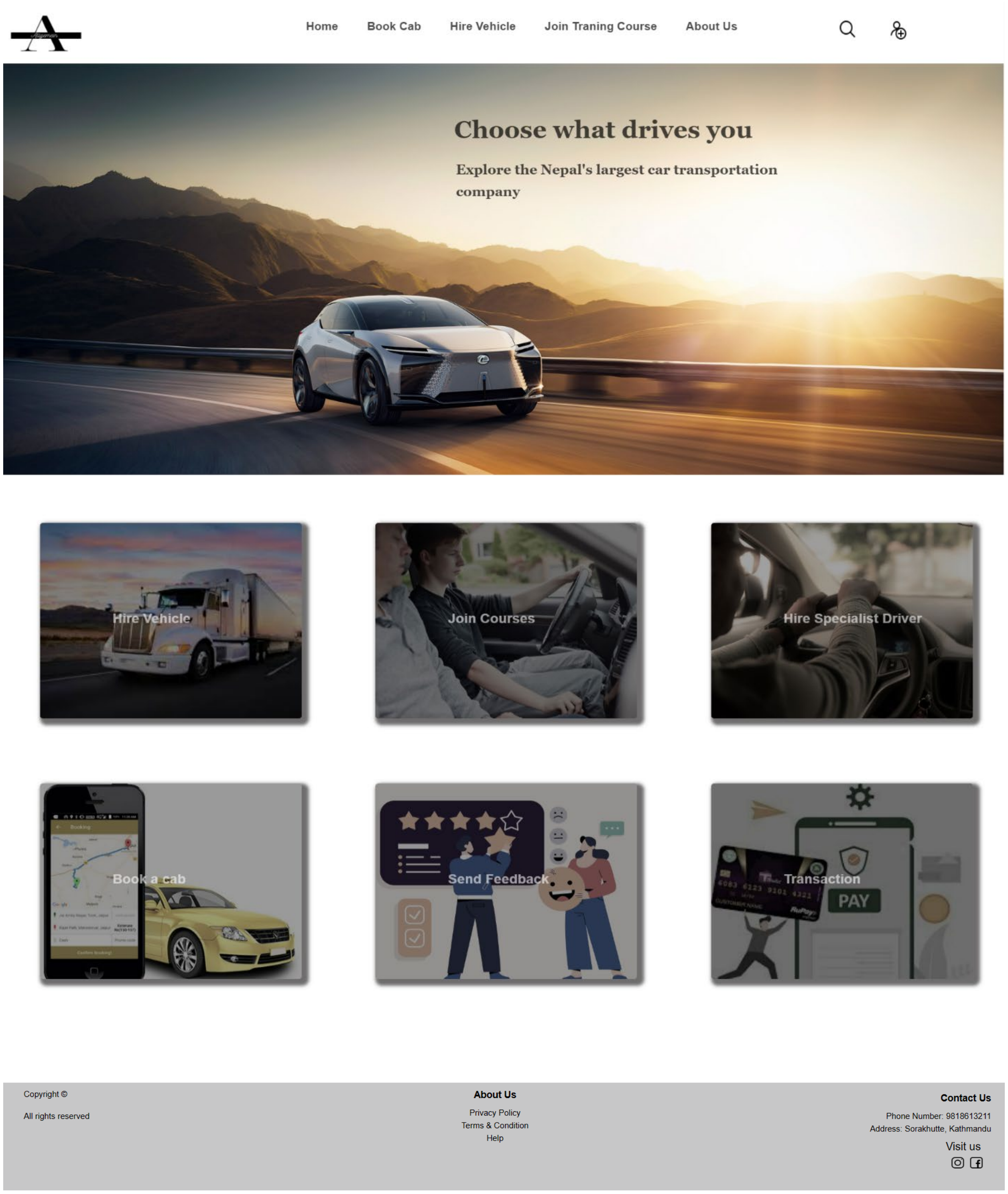
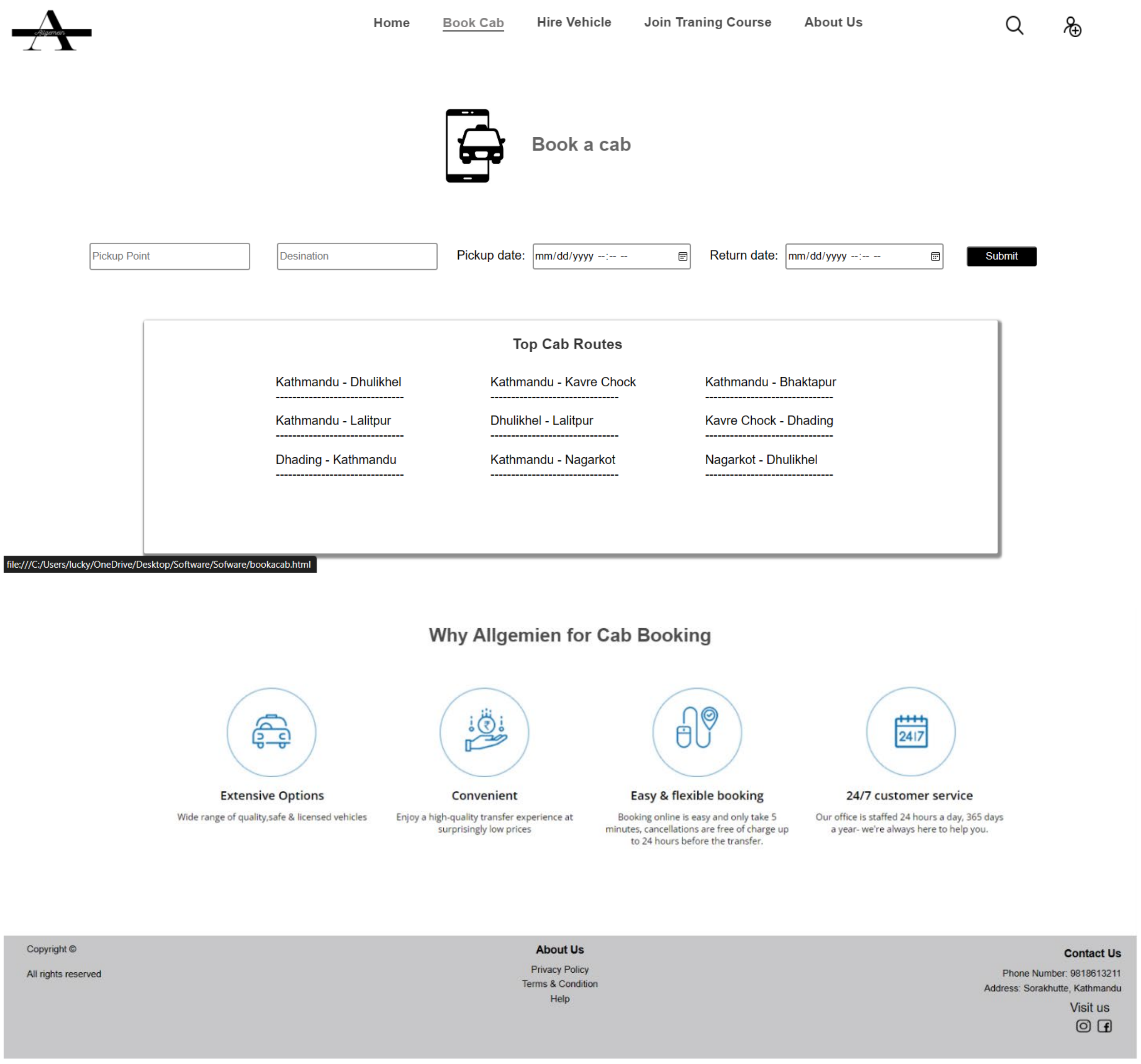


Figure 13 Prototype homepage for user

This the home page for the user, when user login using credentials, this page is shown to user where they book cab, hire vehicle, hire specialist driver, book a cab and send feedback.



Top Cab Routes

Kathmandu - Dhulikhel	Kathmandu - Kavre Chock	Kathmandu - Bhaktapur
Kathmandu - Lalitpur	Dhulikhel - Lalitpur	Kavre Chock - Dhading
Dhading - Kathmandu	Kathmandu - Nagarkot	Nagarkot - Dhulikhel

file:///C:/Users/lucky/OneDrive/Desktop/Software/Sofware/bookacab.html

Why Allgemien for Cab Booking

Extensive Options
Wide range of quality,safe & licensed vehicles

Convenient
Enjoy a high-quality transfer experience at surprisingly low prices

Easy & flexible booking
Booking online is easy and only take 5 minutes, cancellations are free of charge up to 24 hours before the transfer.

24/7 customer service
Our office is staffed 24 hours a day, 365 days a year- we're always here to help you.

Copyright ©
All rights reserved

About Us
Privacy Policy
Terms & Condition
Help

Contact Us
Phone Number: 9818613211
Address: Sorakhutte, Kathmandu
Visit us

Figure 14 Prototype book a cab page

When user clicks on book cab on the navigation. This page is shown to user which takes the details of the booking.

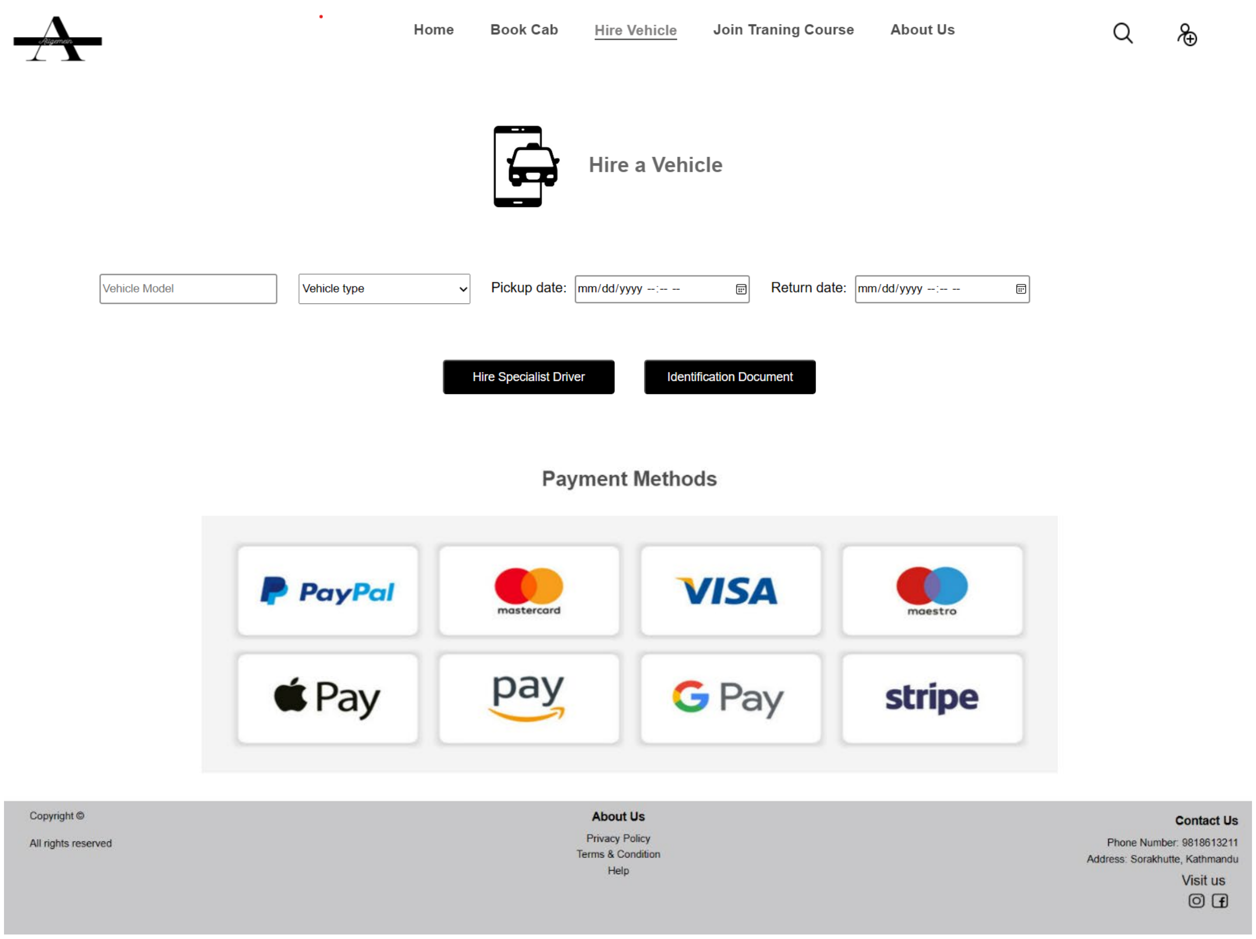
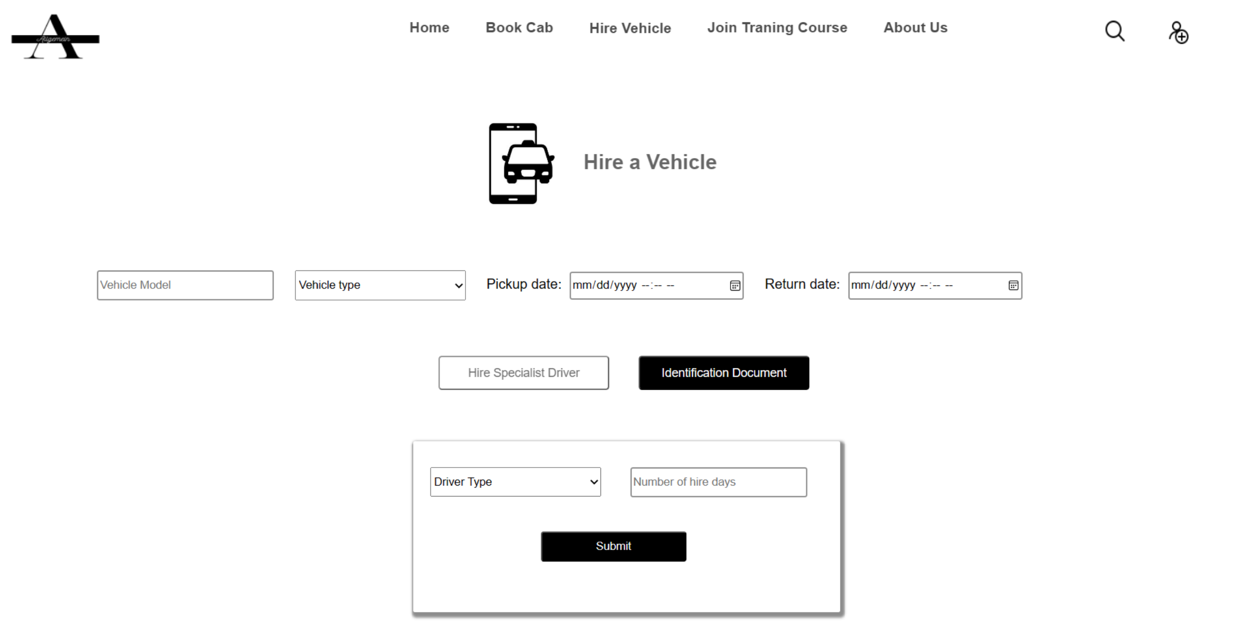


Figure 15 Prototype hire vehicle page

When user clicks on the hire vehicle this form is shown with payment options.



When user clicks on hire specialist driver the driver type and submit button is shown.

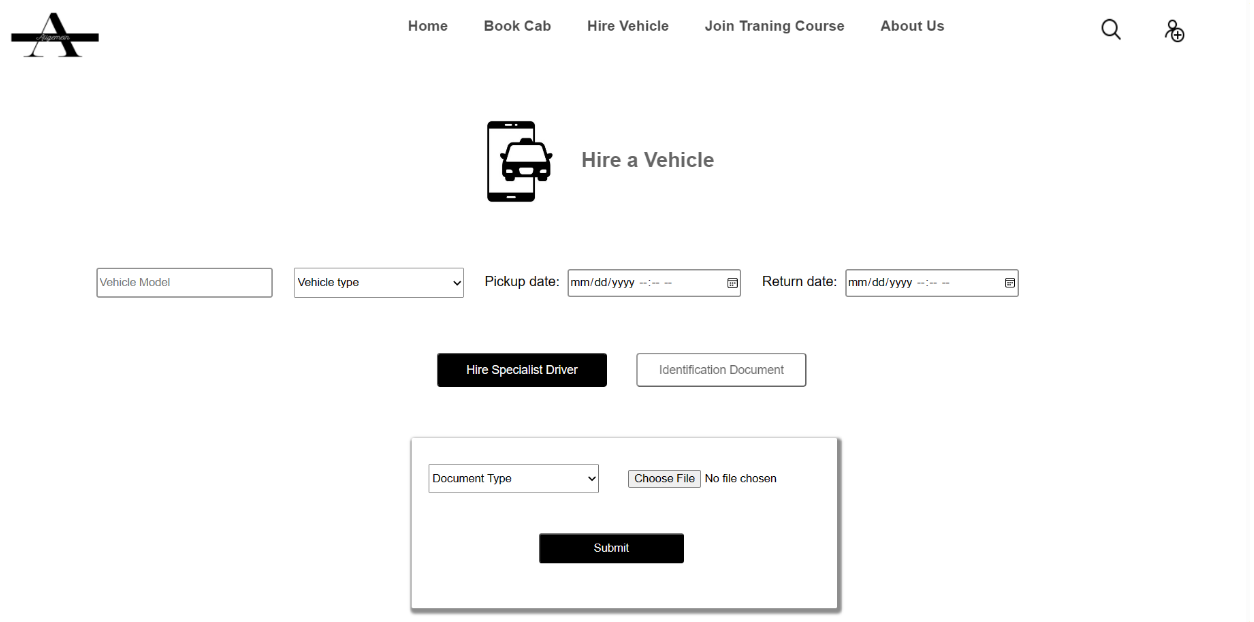
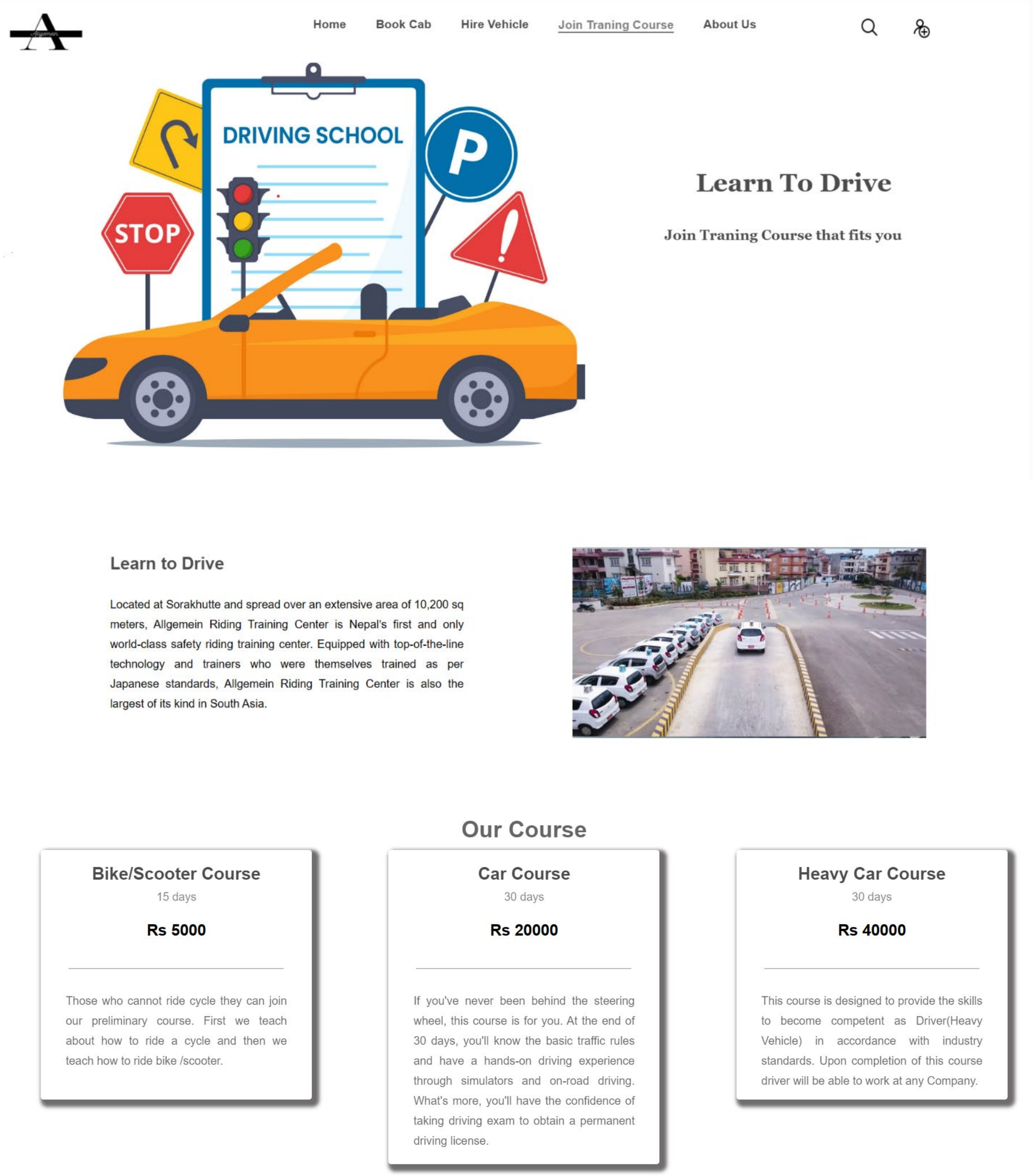


Figure 16 Prototype hire driver/document page

When user clicks on identity document the document type and submit button is shown with option to choose file from their computer.



Learn To Drive

Join Traning Course that fits you

Learn to Drive

Located at Sorakhutte and spread over an extensive area of 10,200 sq meters, Allgemein Riding Training Center is Nepal's first and only world-class safety riding training center. Equipped with top-of-the-line technology and trainers who were themselves trained as per Japanese standards, Allgemein Riding Training Center is also the largest of its kind in South Asia.

Our Course

Bike/Scooter Course
15 days
Rs 5000
Those who cannot ride cycle they can join our preliminary course. First we teach about how to ride a cycle and then we teach how to ride bike /scooter.

Car Course
30 days
Rs 20000
If you've never been behind the steering wheel, this course is for you. At the end of 30 days, you'll know the basic traffic rules and have a hands-on driving experience through simulators and on-road driving. What's more, you'll have the confidence of taking driving exam to obtain a permanent driving license.

Heavy Car Course
30 days
Rs 40000
This course is designed to provide the skills to become competent as Driver(Heavy Vehicle) in accordance with industry standards. Upon completion of this course driver will be able to work at any Company.

Figure 17 Prototype join course page

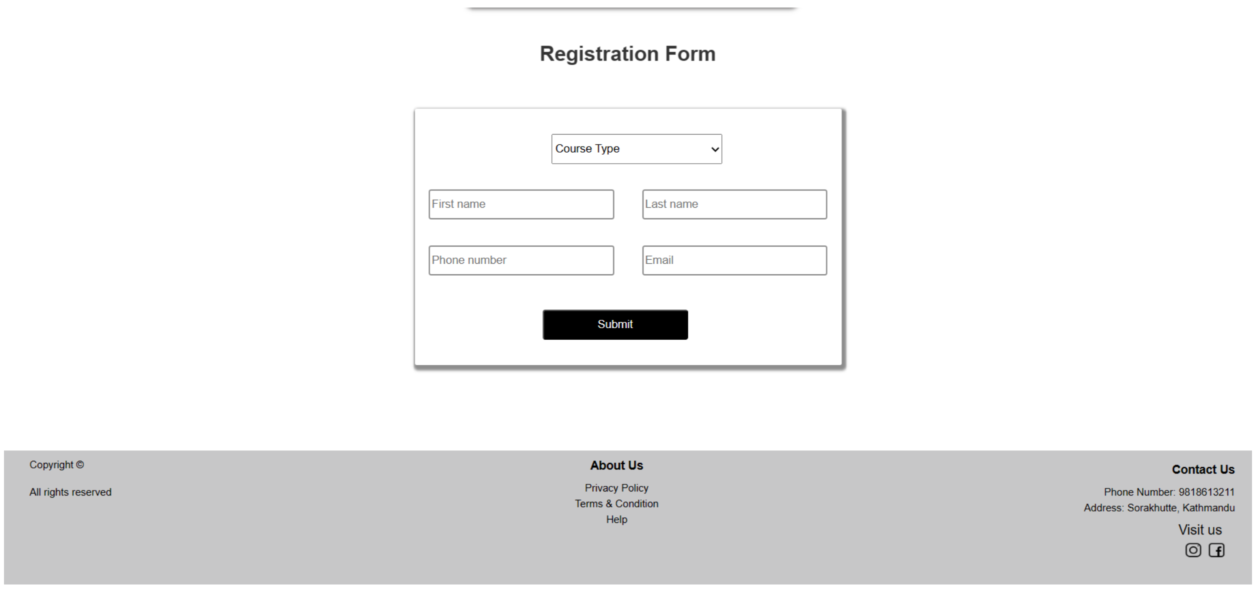
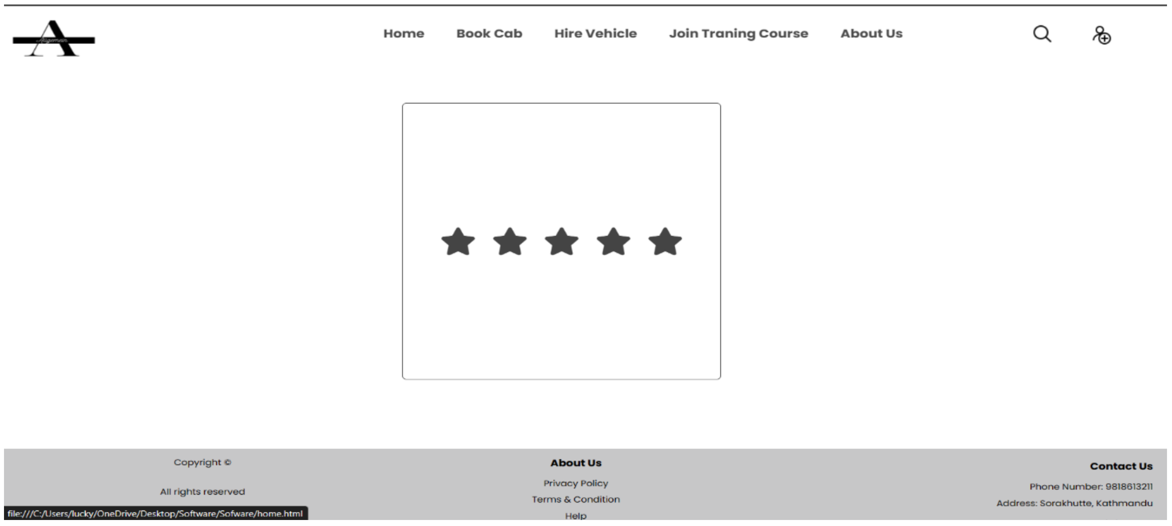
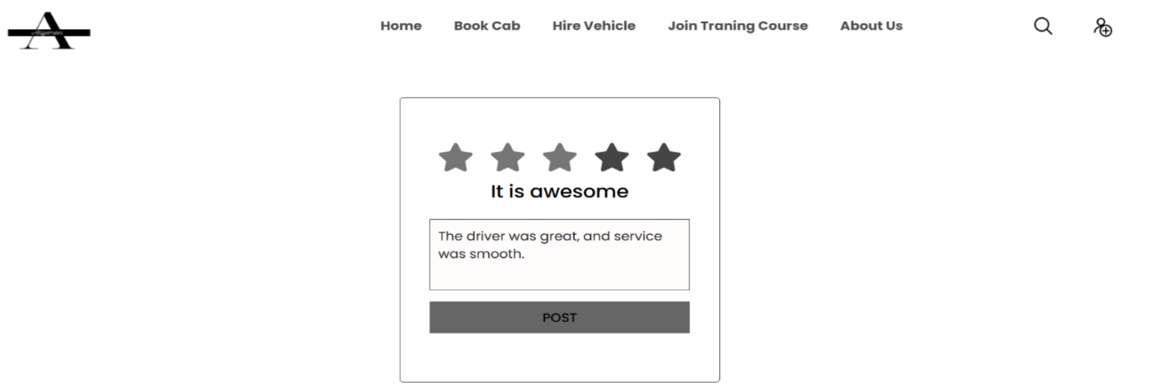


Figure 18 Prototype join course form page

When user clicks on join training course the above page is shown which contains details of course with the form to join the course.



When user clicks on send feedback from the home page then it shows this page where user can rate and leave comment.



The driver was great, and service was smooth.

Using the start user can rate where as send description using the text area.

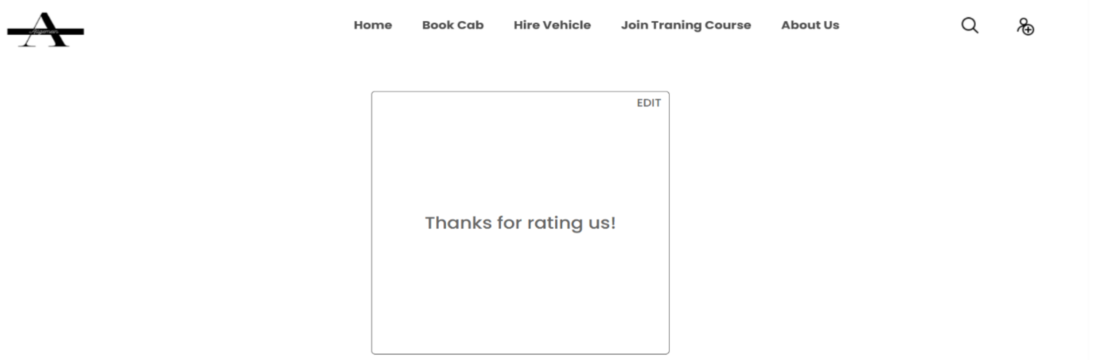


Figure 19 Prototype feedback page

After the post submit button, the user can again edit by clicking on edit button.

7. Conclusion

‘Allegmein’ a company wants to established the transportation system. In order to establish system many components need to create before coding. Even choosing programming paradigms fall under software development phase. In this coursework many new software engineering concepts were used like Gantt chart, use case diagram, sequence diagram, collaboration diagram and class diagram. All those diagram falls under Unified Modeling Language (UML) which is the collection of best engineering practices that successfully model large and complex systems. It helps project teams communicate, explore potential design, and validate the architecture design of the software. In this coursework various software design pattern, architecture pattern, development model, testing and deployment was done. Using prototype model Gantt chart was created which helps to allocated time to task. Using MCV design pattern collaboration and sequence diagram was created.

Prototype of the system was created using html, CSS, JavaScript to know about system further. Using it 12 pages were created which haves almost all the functionality that the Allgemeins system needs. While creating various diagram a deep thinking was need which helped to think as a software engineer and helped to enhanced our thinking how software is created. Various problem arose while creating diagram and even while choosing pattern but those problem were overcome by comprehensive thinking. Allgemeins system software design, creating prototype, Gantt chart and many other diagrams was created in this coursework.

References

FONSECA, L., 2022. *How to Make a Class Diagram [+Examples].* [Online]
Available at: https://venngage.com/blog/class-diagram/
[Accessed 25 04 2023].

greg, 2021. *UML diagram types: everything you need to know.* [Online]
Available at: https://www.gleek.io/blog/uml-diagram-types
[Accessed 22 04 2023].

Lewis, S., 2023. *collaboration diagram.* [Online]
Available at: https://www.techtarget.com/searchsoftwarequality/definition/collaboration-diagram#:~:text=A%20collaboration%20diagram%2C%20also%20known,the%20role%20of%20each%20object.
[Accessed 25 04 2023].

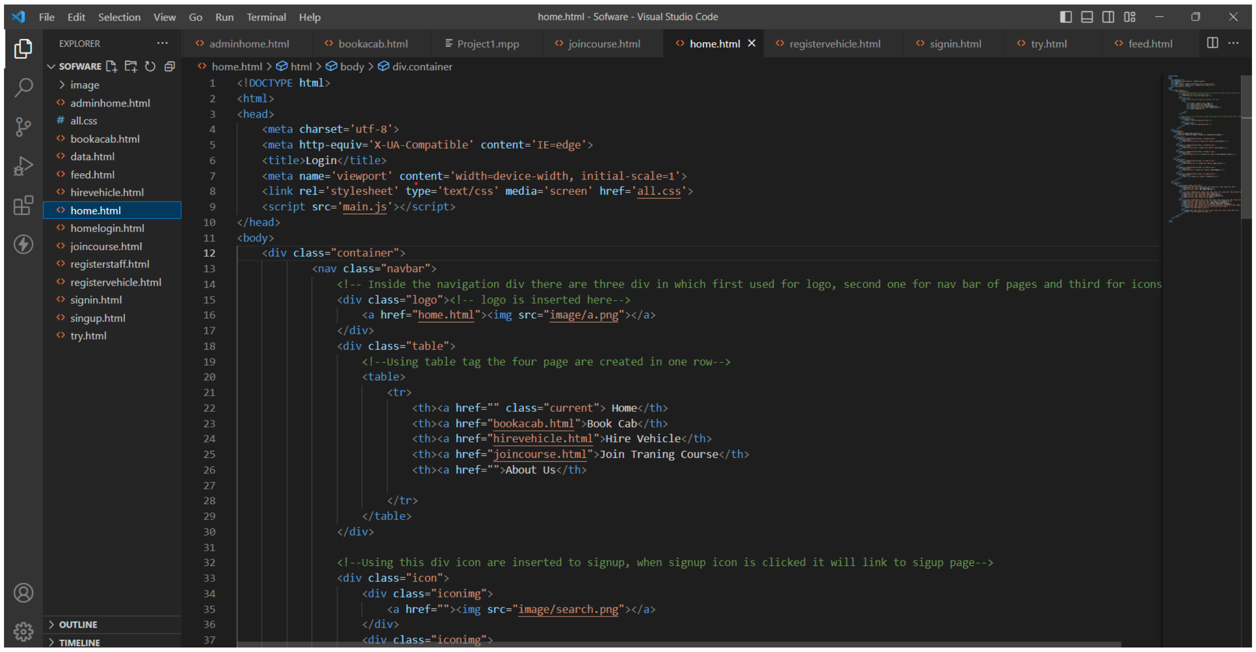
Lutkevich, B., 2021. *Gantt chart.* [Online]
Available at: https://www.techtarget.com/searchsoftwarequality/definition/Gantt-chart
[Accessed 20 04 2023].

Mleziva, M., 2020. *7 Software Development Models In Engineering You Should Know.* [Online]
Available at: https://flexagon.com/blog/7-software-development-models-you-should-know/
[Accessed 23 04 2023].

Nishadha, 2022. *UML Diagram Types Guide: Learn About All Types of UML Diagrams with Examples.* [Online]
Available at: https://creately.com/blog/diagrams/uml-diagram-types-examples/
[Accessed 23 04 2023].

Appendices

Prototype file system image



```html
<!DOCTYPE html>
<html>
<head>
    <meta charset='utf-8'>
    <meta http-equiv='X-UA-Compatible' content='IE=edge'>
    <title>Login</title>
    <meta name='viewport' content='width=device-width, initial-scale=1'>
    <link rel='stylesheet' type='text/css' media='screen' href='all.css'>
    <script src='main.js'></script>
</head>
<body>
    <div class="container">
            <nav class="navbar">
                <!-- Inside the navigation div there are three div in which first used for logo, second one for nav bar of pages and third for icons
                <div class="logo"><!-- logo is inserted here-->
                    <a href="home.html"><img src="image/a.png"></a>
                </div>
                <div class="table">
                    <!--Using table tag the four page are created in one row-->
                    <table>
                        <tr>
                            <th><a href="" class="current"> Home</th>
                            <th><a href="bookacab.html">Book Cab</th>
                            <th><a href="hirevehicle.html">Hire Vehicle</th>
                            <th><a href="joincourse.html">Join Traning Course</th>
                            <th><a href="">About Us</th>

                        </tr>
                    </table>
                </div>

                <!--Using this div icon are inserted to signup, when signup icon is clicked it will link to sigup page-->
                <div class="icon">
                    <div class="iconimg">
                        <a href=""><img src="image/search.png"></a>
                    </div>
                    <div class="iconimg">
```

All Code was done without using bootstrap and was done solely. As proof I have shared the code for only home page not all page to avoid to large documentation.

Code of homepage

```html
<!DOCTYPE html>

<html>

<head>

    <meta charset='utf-8'>

    <meta http-equiv='X-UA-Compatible' content='IE=edge'>

    <title>Login</title>

    <meta name='viewport' content='width=device-width, initial-scale=1'>

    <link rel='stylesheet' type='text/css' media='screen' href='all.css'>

    <script src='main.js'></script>

         <style>

                  .al{
                    height: 500px;
                    width: 100%;
                    display: flex;
                    justify-content: space-evenly;
                }
                .alls{
                    height: 450px;
                    width: 500px;
                }
        </style>
</head>
<body>
   <div class="container">
                    <nav class="navbar">
                        <!-- Inside the navigation div there are three div in which first
used for logo, second one for nav bar of pages and third for icons-->
                        <div class="logo"><!-- logo is inserted here-->
                            <a href="homepage.html"><img
src="image/a.png"></a>
                        </div>
                        <div class="table">
                            <!--Using table tag the four page are created in one
row-->
                            <table>
                                <tr>
                                    <th><a href="" class="current">
Home</th>
                                    <th><a href="bookacab.html">Book
Cab</th>
                                    <th><a href="hirevehicle.html">Hire
Vehicle</th>
```

```
                                                                        <th><a href="joincourse.html">Join
Traning Course</th>
                                                                        <th><a href="">About Us</th>


                                                              </tr>
                                                    </table>
                                          </div>


                                          <!--Using this div icon are inserted to signup, when signup
icon is clicked it will link to sigup page-->
                                          <div class="icon">
                                                    <div class="iconimg">
                                                              <button class="btns"><a
href="signin.html">Login</a></button>
                                                    </div>
                                                    <div class="iconimg">
                                                              <button class="btns"><a
href="singup.html">Signup</a></button>
                                                    </div>
                                          </div>
                              </nav>
    </div>
          <div class="main">
                    <div class="car">
                              <p class="aa">Choose what drives you</p>
                              <p class="aab">Explore the Nepal's largest car transportation
company</p>
                    </div>
                    <div class="al">
                              <div class="alls" style="margin-top: 80px;">
				<h2 style="opacity: 0.7;">Download our app today</h2>

				<p style="line-height: 28px; text-align: justify; margin-top: 
30px;">Download our app and get all of the vehicles under your finger tips. Hire our 
cars for short periods; daily, weekly, fortnightly or subscribe to them for a monthly fee. 
Self Drive Nepal has an amazing fleet of rental cars ranging from compact hatchbacks 
to roomy sedans and powerful SUVs. There are several car models that you can hire, 
including the Hyundai Grand i10, Hyundai Creta, Honda City, Maruti Suzuki Brezza, 
Ford EcoSport, Mahindra Scorpio, Toyota Innova Crysta, Mahindra XUV, and many 
more. Hurry up grab the best deals.</p>

				<img style="height: 120px; width: 150px;" 
src="image/ee.webp">

			</div>

			<div class="alls">

				<img src="image/a.png">

			</div>

		</div>

	</div>

	<div class="dic">

		<footer>

			<div style="height: 100px; width: 100%; margin-top: -320px;">

				<div style="height: 40px; width: 30%; float: left; margin-left: 
30px; margin-top: 10px;">

					<p style="font-size: 12px;">Copyright &copy</p><br>

					<p style="font-size: 12px";> All rights reserved </p>

				</div>

				<div align="center" style="height: 100px; width: 30%; float: 
left; margin-left: 30px; margin-top: 10px;">

					<p style="font-size: 14px; font-weight: bold;">About 
Us</p>

					<p style="font-size: 12px; margin-top: 10px">Privacy 
Policy</p>

					<p style="font-size: 12px; margin-top: 5px">Terms & 
Condition</p>
```

```
                                        <p style="font-size: 12px; margin-top: 5px">Help</p>
                                </div>
                                <div align="right" style="height: 40px; width: 30%; float: right;
margin-left: 30px; margin-top: 10px; margin-right: 20px;">
                                        <p style="font-size: 14px; margin-top: 5px; font-
weight: bold;">Contact Us</p>
                                        <p style="font-size: 12px; margin-top: 10px;">Phone
Number: 9818613211</p>
                                        <p style="font-size: 12px; margin-top: 5px;">Address:
Sorakhutte, Kathmandu</p>
                                        <p style="font-size: 16px; margin-right: 15px; margin-
bottom: 5px; margin-top: 10px;">Visit us</p>
                                        <div style="height: 20px; width: 5%; float: right;
margin-right: 10px;">
                                                <a href=""><img src="image/fac.png"></a>
                                        </div>
                                        <div style="height: 20px; width: 5%; float: right;
margin-left: 10px; margin-right: 5px;">
                                                <a href=""><img src="image/insta.png"></a>
                                        </div>
                                </div>
                        </div>
                </footer>
        </div>
</body>
</html>
```

CSS of home and other pages

```
*{
        font-family: 'Montserrat', sans-serif;
        margin: 0px;
}
.container{
   height: 90px;
        width: 95%;
   margin: 0px 0px;


}
img{
   width: 100%;
   height: 100%;
}
.navbar{/* This class is used to create navigation bar by giving following heights and
width*/
        height: 90px;
        width: 95%;
   display: flex;
   justify-content: space-between;
   align-items: center;



}


.logo{/* This class is used to create div for the logo*/
        width: 120px;
        height: 90px;
        position: relative;
}
```

```
.table{/* This class is used to create table having four column in one row which is used
to create the navbar pages*/
        width: 700px;
        height: 40px;
   margin-left: 200px;
        flex-shrink: 1;
}
table th{
   margin: 20px;
   display: inline;
        font-size: 17px;
        opacity: 0.7;
        align-items: center;
        overflow: hidden;
}
.icon{/* This class is used to create icon div*/
        width: 100px;
        height: 40px;
   position: relative;
   display: flex;
   justify-content: space-between;


}
a{/* the text decoration of link hyper link tag is removied and color is given*/
        text-decoration: none;
        color: black;
}
tr th a:hover{/* The a tag is given hover effect*/
        border-bottom: 2px solid black;
        transition: 0.2s;
        color: rgb(79, 77, 77);
}
.iconimg{/*The class is made for icon in the navigation bar for all the image*/
        height: 30px;
        width: 28px;
   display: inline-block;
        box-sizing: border-box;

}
.btns{
        padding: auto;
        border-radius: 4px;
        font-size: 15px;

}
button:hover{
        background-color: rgb(230, 230, 230);
        transition: 0.1s;
        color: aliceblue;

}
.para{/*the class is created for signup at the bottom of page */
        margin-top: 10px;
        height: 22px;
        width: 150px;
        background-color: black;
        color: white;
        border-radius: 2px solid black;
        border-radius: 5px;
        transition: 0.4s;
}
.para:hover{/* Hover effect is given to the class for signup at the bottom of page */
        background-color: #C7C7C8;
        color: black;
}

.main{


        height: 1000px;
        width: 100%;


}
.car{
        height: 600px;
        width: 100%;
        background-image: url("image/aa.jpg");
        background-size: cover;
}
.aa{
        position: absolute;
        top: 25%;
        left: 60%;
        transform: translate(-50%, -50%);
        font-size: 36px;
        font-weight: bolder;
        font-family: Georgia;
        opacity: 0.7;
        animation-name: example;
        animation-duration: 4s;
        display: flex;
}
.aab{
        position: absolute;
        top: 35%;
        left: 63.5%;
        transform: translate(-50%, -50%);
        font-size: 20px;
        font-weight: bold;
        font-family: "Georgia";
        opacity: 0.7;
        line-height: 1.6;
        animation-name: example;
        animation-duration: 4s;
}
@keyframes example {
        from {opacity: 0.1; left: ;}
        to {opacity: 1;}
        0%   {left: 20%;}
   50%  {left: 50%;}
  }

.parts{
        height: 450px;
        width: 100%;
        display: flex;
        justify-content: space-around;
        align-items: center;
        margin-bottom: -50px;

}
.is{
        background-color: black;
        height: 300px;
        width: 380px;
        background-size: cover;
        border-radius: 4px;
        box-shadow: 5px 5px 5px 5px rgb(111, 109, 109);

}
.text1{
        width: 250px;
        margin-top: 135px;
        margin-left: 105px;
        font-size: 20px;
        font-weight: bold;
        color: white;
        opacity: 0.6;
        font-family: "Georgia";
}
.is:hover{
        opacity: 0.6;
```

```
}
.dic{
        margin-top: 480px;
        height: 160px;
        background-color: #C7C7C8;

}
.h1s{
        height: 400px;
        width: 400px;
        margin-top: -50px;
}
.h2s{
        height: 500px;
        width: 500px;
}
```