

Private Car Sharing Application (CAR2SHARE)

By

Yara Wael Taha

Sara Ahmed AlAnsary

Sara Fahad AlGhannam

Nawal Khalid Almujel

Supervised By

Dr. Khulood Almani

A Graduation Project Report Submitted to
College of Computer Sciences and Information at PNU
in Partial Fulfillment of the Requirements for the
Degree of
Bachelor of Science
in
Information Systems

CCIS, PNU

Riyadh, KSA

1441 - 1442H

Table of Contents

List of Tables	5
List of Figures	6
List of Symbols and Abbreviations	8
Acknowledgment	9
Abstract	10
Introduction	1
Problem Statement	1
Proposed Solution	2
Project Description	3
Project Methodology	3
RAD Model Diagram	4
Stage 1: Business Modeling	4
Stage 2: Data Modeling	4
Stage 3: Process Modeling	5
Stage 4: Application Generation	5
Stage 5: Testing and Turnover	5
Project Domain	5
Project Limitations	6
Background Information	7
Background information	7
2.2 Related Work Survey	8
Turo	8
Drive Drive Car	8
Nobobil.no	9
2.3 Proposed & Similar System Comparison	10
Chapter 3: System Analysis	11
3.1 System Analysis	11
3.2 Requirements Determination and Collection	11
3.2.1 Surveys	11

3.2.2 Functional Requirements	16
3.2.3 Non-Functional Requirements	17
3.3 Software/Hardware Requirements:	18
3.4 Requirements Analysis	20
3.3.1 Use Case Diagram	20
3.3.2 Class Diagram	21
3.3.3 Sequence Diagram	22
Chapter 4: System Design	23
4.1 System Architecture	23
4.1.1 Description of components	24
4.2 System Design Tools	25
Chapter 5: Implementation	32
5.1 Implementation requirements	32
5.1.1 Hardware Requirements	32
5.1.2 Software Requirements	33
5.1.2.1 Admin interface	33
5.1.2.1.1 Programming languages	33
5.1.2.1.2 Programs	34
5.1.2.1.2 Framework Environment	34
5.1.2.2 User Interface	35
5.2 Implementation details	35
5.3 Code Screenshots	36
5.3.1 I/O Screens	42
Chapter 6: Testing	53
Testing:	53
6.1 Testing Plan	53
6.2 Testing Strategy	54
6.2.1 Unit Testing	54
6.2.2 Functional Testing	54
6.2.3 Acceptance Testing	54

6.3 Test Cases	55
6.3.1 Admin Test Case	55
6.3.2 Renter Test Case	56
6.3.3 Rentee Test Case	57
6.4 Test Results	58
6.4.1 Admin Test Results	58
6.4.2 Renter Test Results	58
6.4.2 Rentee Test Results	59
Chapter 7: Conclusion	60
References	61
Appendix	63

List of Tables

Table 2.1 - Proposed & Similar System Comparison

Table 3.1 - Functional Requirements Table

Table 3.2 - Non-Functional Requirements Table

Table 3.3 – Hardware Requirements Table

Table 3.4 – Software Requirements Table

Table 6.1 – Admin Test Case

Table 6.2 – Renter Test Case

Table 6.3 – Admin Test Case

Table 6.4 – Admin Test Case Results

Table 6.5 – Renter Test Case Results

Table 6.6 – Rentee Test Case Results

List of Figures

Figure 1.1	4
Figure 2.1 Related Work (Turo)	8
Figure 3- Related Work (Drive Drive Car)	8
Figure 4 - Related Work (Nabobil.no)	9
Figure 5 - Survey Question 1 Statistics	12
Figure 6 - Survey Question 2 Statistics	12
Figure 7 - Survey Question 3 Statistics	13
Figure 8 - Survey Question 4 Statistics	14
Figure 9- Survey Question 5 Statistics	15
Figure 10 - Survey Question 6 Statistics	15
Figure 11 - Use Case Diagram	20
Figure 12 - Class Diagram	21
Figure 13 - Sequence Diagram	22
Figure 14 - System Architecture	23
Figure 15 - System Design Tools	25
Figure 16 - ERD	26
Figure 17 - Interface 1	27
Figure 18- Interface 2	27
Figure 19 - Interface 3	28
Figure 20 - Interface 4	28
Figure 21 - Interface 5	29
Figure 22 - Interface 6	29
Figure 23 - Interface 7	30
Figure 24 - Interface 8	30
Figure 25 - Interface 9	31
Figure 26 - Code 1	36
Figure 27 - Code 1 Output	36
Figure 28 - Code 2	37
Figure 29 - Code 2 Output	37
Figure 30 - Code 3	38
Figure 31 - Code 4	38
Figure 32 - Code 5	39
Figure 33 - Code 6	39
Figure 34 - Code 7	40
Figure 35 - Code 8	40
Figure 36 - Code 9	41
Figure 37 - Code 10	41
Figure 38 - I/O 1	42
Figure 39 - I/O 2	42
Figure 40 - I/O 3	43
Figure 41 - I/O 4	43
Figure 42 - I/O 5	44
Figure 43 - I/O 6	44
Figure 44 - I/O 7	45

Figure 45 - I/O 8	45
Figure 46 - I/O 9	46
Figure 47 - I/O 10	46
Figure 48 - I/O 11	47
Figure 49 - I/O 12	47
Figure 50 - I/O 13	48
Figure 51 - I/O 14	48
Figure 52 - Admin Interface 1	49
Figure 53 - Admin Interface 2	50
Figure 54 - Admin Interface 3	50
Figure 55 - Admin Interface 4	51
Figure 56 - Admin Interface 5	51
Figure 57 - Admin Interface 6	52
Figure 58 - Booking Code	63
Figure 59 - Location Code	63
Figure 60 - Car Database Code	64
Figure 61 - Thread Code	64
Figure 62 - Attributes Table	65
Figure 63 - Login Page Code	65

List of Symbols and Abbreviations

PNU	Princess Nora bint Abdul Rahman University
GCC	Gulf Cooperation Council
KSA	Kingdom of Saudi Arabia
RAD	Rapid Application Development
UK	United Kingdom
US	United States
UAT	User Acceptance Testing

Acknowledgment

First of all, we would like to thank Allah for giving us the power, patience, and knowledge we needed to complete this portion of the project.

Secondly, we would like to express our gratitude towards the faculty of Computer and Information Sciences at Princess Nourah University without whom we wouldn't have learned the valuable knowledge and skills needed to build this project. We would also like to extend our deepest heartfelt thanks to our amazing supervisor Dr. Khuloud AlMaani who has supported us every step of the way.

Lastly, we would like to thank our wonderful families for their continuous support and motivation.

Abstract

This project is about developing an application platform that provides a peer-to-peer car rental service that allows for a more efficient and seamless car renting experience. Car2share provides its users with the opportunity to make an additional income by putting their car up for rental while being fully insured. Additionally, it enables people in need of a car to be able to obtain any kind of vehicle quickly and efficiently.

Keywords	Meaning
Rentee	The person renting out their car
Renter	The person renting a car
Peer-to-peer	Allows “ peers ” (individual computer systems) to connect over the internet to share files.
Testing	Conducted through scripts with the motive of finding errors in software. It deals with tests for the entire application.

Chapter 1: Introduction

Problem Statement

Car rental is a service that enables individuals to get around even though they do not personally own a car. With that being said, car rental companies only have a select number of locations to which you must go to rent a car, making them not easily accessible. With the rapid evolution of technology and the need for quick user satisfaction, the old-fashioned approach for car rental used in Saudi Arabia is not suitable for the modern approach to technology the Kingdom is striving for.

As of 2018, Saudi Arabia implemented a 5% (five percent) tax regulation [1]. This took a financial toll nationwide, especially towards the social class which was living comfortably pre-tax. That social class usually entails owning multiple cars which were considered common. The aftermath of the tax regulation included gas prices spiking, salaries decreasing, and an entire nation becoming financially cautious.

Careem [2] is a car booking service established in the middle east which became available in Saudi Arabia as of 2015, shortly followed by Uber [3]. Both companies provide one-way rides from any location to your desired destination. Their pricing is established based on availability, distance, timing, and vehicle type. When both companies entered the Kingdom they took advantage of a gap in the market, which was a lack of availability for car booking services, so they were able to set prices in a flexible manner that suits their preference. However, both applications simply provide car booking services, which are distinctly different from car rental services.

At the beginning of 2020, everything in regards to the economy changed, not only in Saudi Arabia but globally. It started when COVID-19 reached the Kingdom [4] causing individuals' paychecks to decrease, the country's resources being used to the furthest extent, oil prices decreasing worldwide which led to a large gap in the Kingdom's income. All of these factors combined caused an increase in taxes to be implemented from 5% (five percent) to 15% (fifteen percent) leading to a more financial struggle for individuals. With this increase, multiple car owners resorted to putting their other cars aside and only using one car. Rendering those extra cars useless. This is where Car2Share comes.

This project aims to develop an application service platform for peer-to-peer private car-sharing [5] that allows car owners to convert their personal vehicles into shared cars that can be rented to other drivers on a short-term basis. It would also provide a car rental service in areas and small towns that have no access to normal commercial car rentals. Car2share application service is the personalized alternative to traditional car rental, car finance, and inflexible business car leasing, which fits with the ‘on-demand mentality of the era. The proposed application service does not have the overheads of traditional car rental companies, which in turn provides substantial savings on the renter accompanied by low costs for vehicle owners which means that market expansion is possible. The platform manages all car rentals, insurance, and the end to end payments and billing through an online system.

Proposed Solution

Car2Share proposes making car rental services more efficient, allowing them to be beneficial for both the car owner and the renter, as well as providing a seamless experience of renting a car and receiving daily income from renting out your personal car.

With Car2share being a peer-to-peer platform, the process of picking up the car from the select locations available from traditional car rental companies is eliminated. Both the renter and the rentee are able to communicate and mutually agree on a location that is compatible with both parties.

A portion of Careem and Uber audience will be swayed towards Car2Share and the service it will provide, this is mainly because their prices have skyrocketed the past year making it less attractive and not as affordable as it was before, additionally, it will no longer be the only option for individuals who own a license but no personal car.

Women were granted the ability to obtain their driving licenses in 2018 [6], a large portion of those women were able to get their license instantly, however, they didn't have the financial ability to purchase a car yet. Car2share will be a great alternative solution to enable women to drive without owning a personal car.

With the initial implementation of taxes and the later increase [7], the individuals with multiple cars have the ability to have additional income from renting out their cars that are not

being used as much, helping them deal with the financial strain that has been developed due to the increase in expense of living in the Kingdom.

Project Description

Car2Share is an application that will allow car owners who do not use their cars all the time to rent them out either by the hour or by day. This is enabled through a peer-to-peer platform that provides for individuals who require a car for a longer period than just a one-time car ride. The application will include an interface for both the renter and the rentee, which will accommodate each of their needs. One of the main aspects of the platform is that insurance will be provided to protect all parties.

Project Methodology

An Agile Methodology [8] was chosen for Car2Share, specifically, Rapid Application Development. RAD [9] is a type of software development that does not need a lot of time or resources for planning but instead uses a prototyping method to introduce a product. RAD follows through with five steps that consist of Business Modeling, Data Modeling, Process Modeling, Application Generation, and Testing and Modeling. The previously mentioned steps are the best option when developing an application such as Car2Share as it is more flexible and dynamic.

RAD Model Diagram

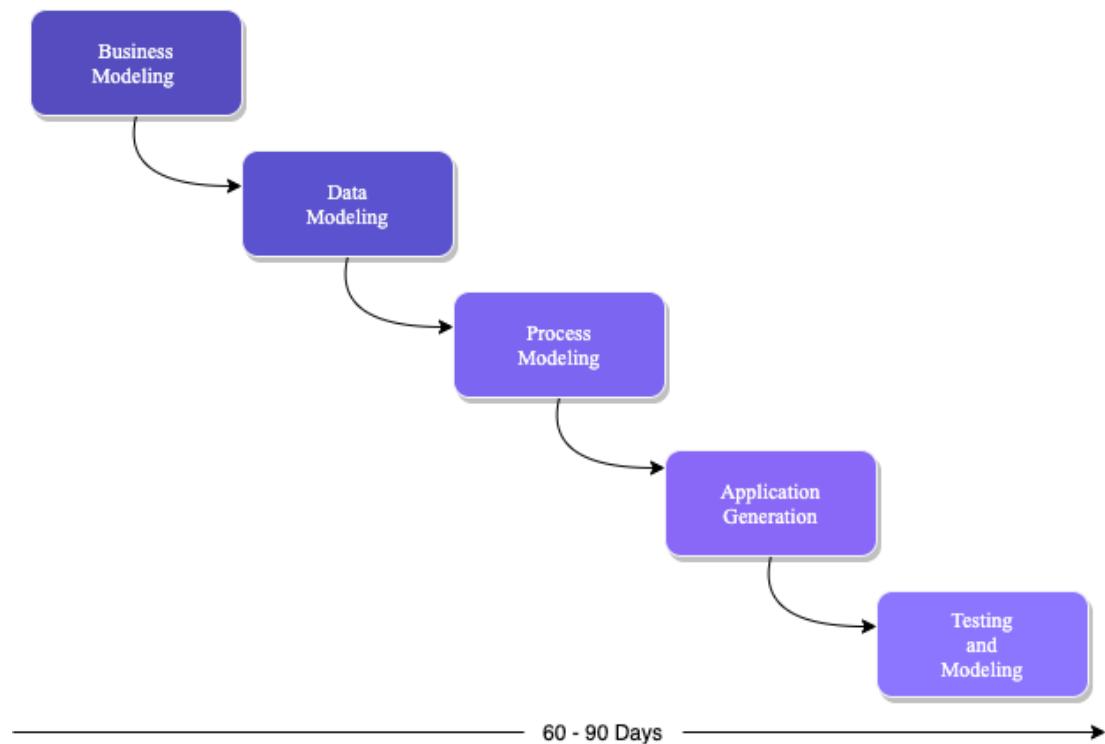


Figure 1.1

Stage 1: Business Modeling

The Business modeling step in RAD modeling entails retrieving information from multiple sources related to the platform through multiple business-related sources. The information is combined into a description that is useful how the information obtained can be used when processed, and what specific information is most valuable to help the interface succeed.

Stage 2: Data Modeling

In this Stage, the information gathered in the Business Modeling phase is reviewed and analyzed. The information is grouped into different categories that can be critical to our project. The quality of each data group is carefully observed and given an accurate description. This phase also helps to determine the RAD system development architecture

from the user feedback, allowing the creation of prototypes. Also, it is repeated many times as the project evolves.

Stage 3: Process Modeling

During the Process Modeling phase, all the information that has been gathered during the Data Modeling steps is converted into the required usable information needed to achieve specific business objectives as per the business model, furthermore, changes and optimization can be done, and the sets of data can be further defined. Any description for adding, removing, or changing the data objects is also conducted during this phase.

Stage 4: Application Generation

The Application Generation step is where the actual system is built, when all the needs, features, and information gathered for the application are put into codes, and where various tools are used to convert processes and data models into actual prototypes that can be used in the next step.

Stage 5: Testing and Turnover

The Testing and Turnover Stage involves testing the prototypes independently after each iteration so that the overall testing time can be reduced.

During this stage it is reassured, there should not be any major and preferably minor problems with our prototype since most of the elements have already been tested earlier.

Project Domain

Car2share will offer its services in Saudi Arabia and the Gulf Cooperation Council (GCC) countries. It will be a business with multiple locations operating in a specific area, in order to expand the project's overall economic activity and to lead to high-level competition among businesses.

Project Limitations

When it comes to limitations there are a few mentionable ones.

- Insurance, legal policies, and registration will differ from region to region, this may cause some limitations when expanding the service to multiple regions
- Cost requirements may be high at first to accomplish all the required tasks prototyping, testing, and implementation

System Aims and Objectives

This system aims to provide peer-to-peer car rental service, car owners will be able to list their cars making them available for renters to rent.

- Make car rental services more accessible, convenient, and flexible to use.
- Allow individuals with multiple cars to have another source of income.
- To ease consumer tasks whenever they need to rent a car.
- Reduce manual work of rental service.
- Make renting out your car more secure.
- Design a user-friendly interface for the application.

Chapter 2: Background Information

2.1 Background information

With the recent tax increase in Saudi Arabia, which led to some financial strain, leading people to seek extra forms of income, Car2Share provides the opportunity to those who are willing to rent out any cars they own. Through the platform, individuals will be able to register themselves and their vehicles into the database. Once registered and approved, they will be covered by the Car2Share insurance plan. At that point they will be able to begin renting out their car, so, their vehicle and information will be displayed on the interface for local users. As for the renters, they will be asked to register with their personal information, especially their driver's license, and then be able to view the options available based on their needs and preferences.

Finding the right car for you:

When trying to find the right car to rent, there is a list of needs and preferences users have. In order to serve those needs, the following questions may be asked:

1. What size of car do you prefer?
2. What car brands do you prefer?
3. What features would you like to be available for you?
4. What is your car renting budget?
5. Would you prefer to rent by the hour or by the day?
6. What car model would you prefer?

Tools used for the project:

Mainly, as a source of communication between renters and rentees, there is a chat feature on the application which enables the exchange of information between the two parties. This feature enables the monitoring of communication to maintain the safety and security of everyone involved.

2.2 Related Work Survey

Through research, we were able to discover multiple similar platforms that are available across the world. Of which include, Turo, Drive Drive Car, and Nobobil.no

1. Turo



Figure 2.1 Related Work (Turo)

Turo [10] is an app available on both the Apple store and Google Play store. Turo is a car-sharing marketplace where guests can book any car they want, wherever they want it, from a vibrant community of local hosts across the US, Canada, and the UK. Guests choose from a unique selection of nearby cars, while hosts earn extra money to offset the costs of car ownership. They offer a wide range of vehicles like trucks, luxury cars, vans, etc. Turo is only available in cities in the UK, the US, and Canada. It seems to be a 35% more affordable option compared to traditional car renting options.

2. Drive Drive Car



Figure 3- Related Work (Drive Drive Car)

Drive Drive Car [11] is an online platform that enables both individuals and companies through its peer-to-peer car-sharing services. The platform allows individuals and companies to list their cars for rent whenever they are not using them for their personal use. Through this platform, they intend to make the process of renting or listing a car easier for clients, increase the availability of cars, a differentiation of cars, and other additional services.

3. Nobobil.no



Figure 4 - Related Work (Nabobil.no)

Nabobil.no [12] is a Norwegian peer-to-peer marketplace for underutilized personal vehicles. It combines people that need a car with people that own a car. You are also allowed to choose whether you prefer to rent the car by the hour, day, week, or month.

One of Nabobil's advantages is it is completely free to join, as you get access to your own virtual garage with cars of all genres.

Nabobil is the perfect option of how a convenient service can make shared mobility not just in big cities, it is everywhere. With their peer-to-peer model, Nabobil makes mobility widely available and permits registered users to put existing cars to better use. With good quality and affordable car to rent for a weekend break or vacation.

2.3 Proposed & Similar System Comparison

Table 2.1 - Proposed & Similar System Comparison

Features	Car2share	Turo	Drive Drive car	Nabobil.no
Geographical Location (MENA)	✓	x	x	x
Insurance Providence	✓	✓	✓	✓
In-App Communication	✓	✓	x	x
Filtering Options	✓	✓	x	x
Rate and Review Renters/Vehicles	✓	✓	✓	✓
Nearest Car and Distance	✓	✓	✓	✓
Security Alerts for Regional Limits	✓	x	x	x
Quick Payments (Apple Pay, STC Pay)	✓	✓	✓	✓

Chapter 3: System Analysis

3.1 System Analysis

System analysis is a method where facts are gathered and interpreted, problems are defined, and a system is decomposed into its components. This is done in order to study the system and its elements to be able to identify the system components. The system analysis will enable us to ensure that the system components are all working efficiently and can accomplish their purpose. The following is what was established for Car2Share.

The Car2Share system aspires to allow peer-to-peer car rental services which somewhat eliminates the traditional car rental systems.

There will be two interfaces made available. One which will be created for the renters through which they will register their information, create an account, and then be able to use the application to rent a car. As for the rentee, the interface available to them will enable them to register their information and their vehicle information into the system, and once their information is validated and they are considered eligible they will be able to put their vehicles up for rent.

3.2 Requirements Determination and Collection

Collecting requirements can be done in several ways. In this project, information is gathered using a survey. This method is chosen as it is a more accurate, fast way to understand our potential customer's needs and expectations.

3.2.1 Surveys

A survey is a research method in which people are asked a set of questions to gather information, opinions, and insights on a particular topic[KAI]. Surveys can be carried out in many ways depending on the methods being used and the targets to be accomplished.

In this study, the survey was created on Google forms, with six total questions for potential car renters and rentees. The survey aims to explore peoples' willingness to rent their own cars to strangers if their vehicles were fully insured by Car2share while also measuring peoples'

needs and acceptance for peer-to-peer car rentals. This survey was circulated using social media outlets such as Whatsapp, Twitter, and Snapchat with a total of 566 responses.

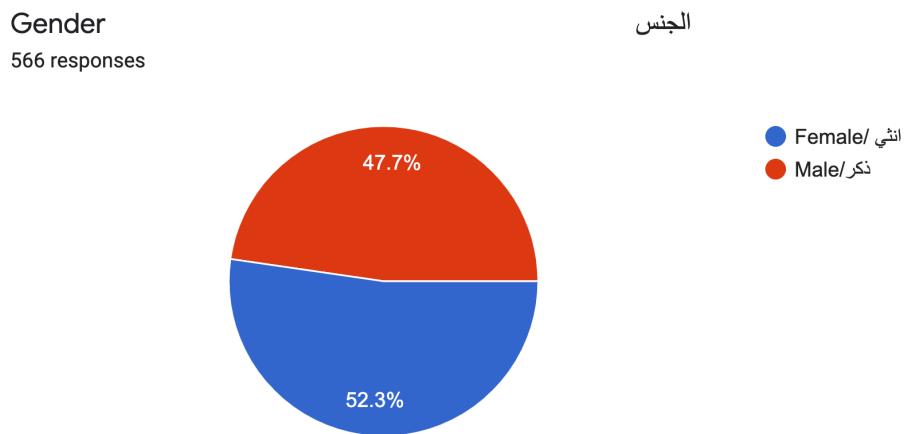


Figure 5 - Survey Question 1 Statistics

This chart (Figure 5), suggests that most of our participants and potential users are females, which further supports our initial statement that since the lift of the women driving ban, they have created a larger need for car services.

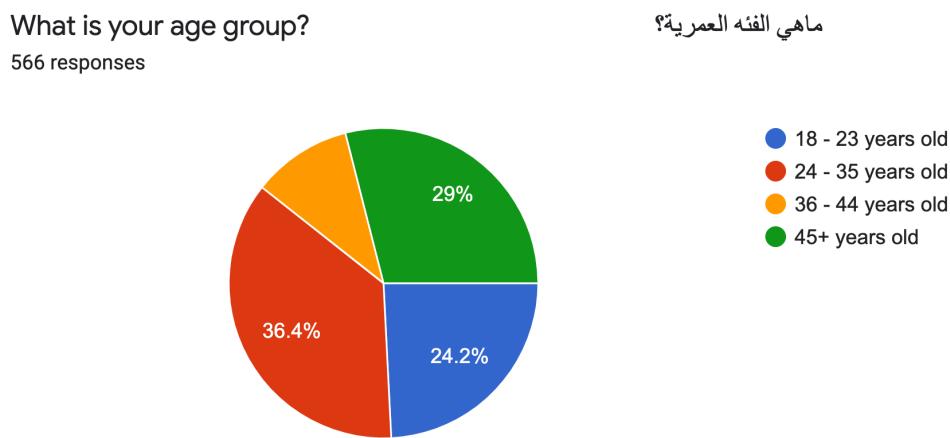


Figure 6 - Survey Question 2 Statistics

In figure 6, the question asked was the age group of the participants, which resulted in the majority being the age group between 24-35-year-olds which amounts to 36.4% of all users. This suggests that most of our potential users are within that age group which provides important insights.

For a reasonable price, would you rent a car privately owned? (not by a company)

بسعر معقول، هل ممكن ان تستاجر سيارة خاصة؟

566 responses

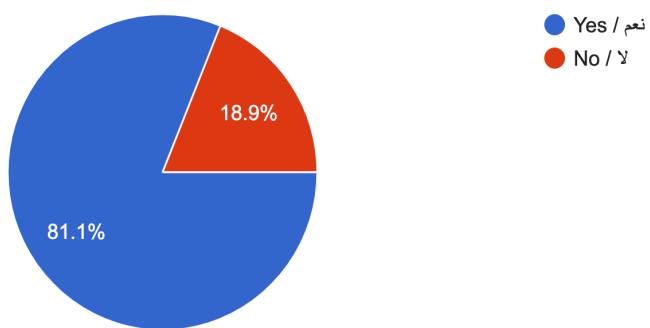


Figure 7 - Survey Question 3 Statistics

Given the response shown in this chart (Figure 7), it indicates that 81.1% of people are willing to rent out their personal cars for additional income. These statistics demonstrate that given the current circumstances, with the rise of taxes and financial strains, people are more inclined to rent out their cars.

If we give you full insurance, would you rent out your personal car or any extra cars for extra income?
إذا قدمنا لك تأم.. فجر سيارتك الشخصية أو أي سيارات إضافية للحصول على دخل إضافي؟
566 responses

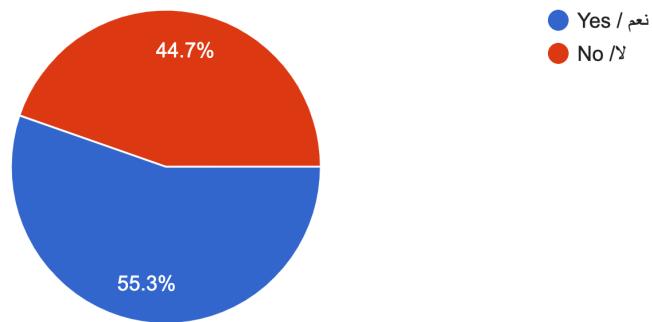


Figure 8 - Survey Question 4 Statistics

We asked the target audience in the chart above (Figure 8) if they would be willing to rent out their personal car, this almost came down to; a little over half of the population saying yes, and the other rest saying no.

Would you find car rental per hour or per day a beneficial service?

هل تجد خدمة تأجير السيارات بالساعة او باليوم، خدمة مفيدة؟

566 responses

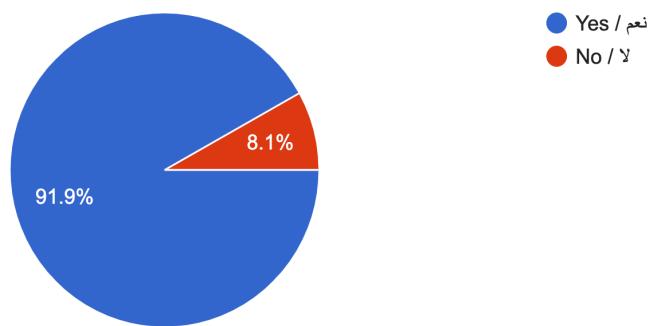


Figure 9- Survey Question 5 Statistics

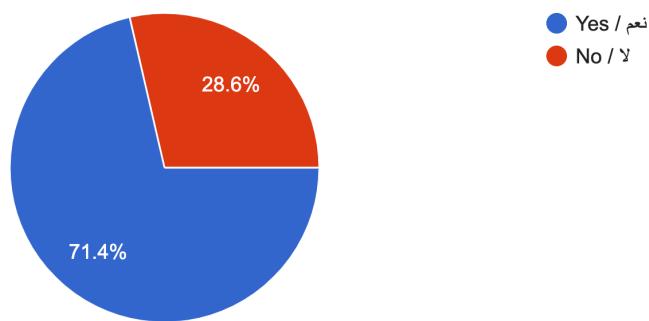
One of the questions asked, shown in figure 9, was if the survey participants found car rental by the hour/by day a beneficial service. 91.9% (520 votes) chose yes, while 8.1% (46 votes) chose no. These numbers display a positive vision for how the audience views our service.

Figure 10 - Survey Question 6 Statistics

Do you know anyone who would benefit from this type of service?

هل تعرف أي شخص يمكن أن يستفيد من هذا النوع من الخدمة؟

566 responses



According to this chart (Figure 10), We asked our participants if they think anyone would benefit from this type of service, And we got 260 Responses saying yes and 119 saying no.

3.2.2 Functional Requirements

Functional requirements [13] define what the system is supposed to accomplish, what it should present, as well as the behavior of the system.

Table 3.1 - Functional Requirements Table

Rentee Registration / Sign Up	The system will allow the rentee to register on the system, requiring all basic information including; <ul style="list-style-type: none">- full name- ID number- email address- phone number- password- car type- car model- car registration
Renter Registration / Sign Up	The system will allow car renters to register in the system before being able to rent a car, they'll need to input; <ul style="list-style-type: none">- full name- ID number- driver license- email address- phone number- password
Login	If Rentee or Renter is already registered they can directly login to their accounts
Browse and Select Car	The system will allow renters to browse

	cars they'd be interested in and will have the option to filter selection.
Automatic Update	Automatic updates to the displayed cars are updated in the database once the reservation has been made by a renter
Rentee Notified	After a customer/renter has reserved a car, the owner of the car (rentee) is notified of the rental reservation
Feedback	Renter: Renters will be given the option to rate the car they rented as well as any additional comments Rentee: Rentee will be given the option to rate the renter's use of the car as well as any additional comments

3.2.3 Non-Functional Requirements

Non-Functional Requirements [14] defines the system's attributes that should be considered for implementing our project.

Table 3.2 - Non-Functional Requirements Table

1- Usability:	<ul style="list-style-type: none"> - Users should be able to understand the flow of the App easily without any guidelines. - The Graphical user interface(GUI) will be user friendly
---------------	--

2-Security:	All of the Car2share users(both the renter and the rentee)have a privilege and can access the system using the email/username/number as well as the driver's license will be required for becoming a Car2share user.
3- Responsiveness:	The application has to be responsive to user input, if any external interrupt occurs it has to return to the same state. For example: if the user was using the app and got interrupted by a Phone call/Facetime etc, The app should return to the same page and save the state.
4-Performance:	The application's performance is measured by its responsive time. For example: when an Application is made to start up it shouldn't take more than 3 seconds to load the initial screen.

3.3 Software/Hardware Requirements:

Hardware Requirements:

The hardware requirements are the physical computing resources that we need to consider while implementing our project.

Table 3.3 - Hardware requirements Table

Item	Purpose
Personal Laptop	Used for researching purposes, programming, creating an interface
External hard disk	For more storage space to keep the data
Smart Phone	It gives us the ability to test and use the application, and communicate between team members.

Software Requirements:

Software Requirements represents the programs we will be using for Car2Share that perform different tasks on a computer system.

Table 3.4 - Software Requirements Table

Item	Example
Internet Browser	Safari and google chrome to search about any related work.
Microsoft Office Programs	Word and PowerPoint for documenting and presenting purposes.
Operating Systems	Since The operating system plays a pivotal role in all development. Our application supports Android and iOS because they are the most popular.
Diagram's Applications	Draw.io, DrawExpress, and for the interface: proto.io

3.4 Requirements Analysis

3.3.1 Use Case Diagram

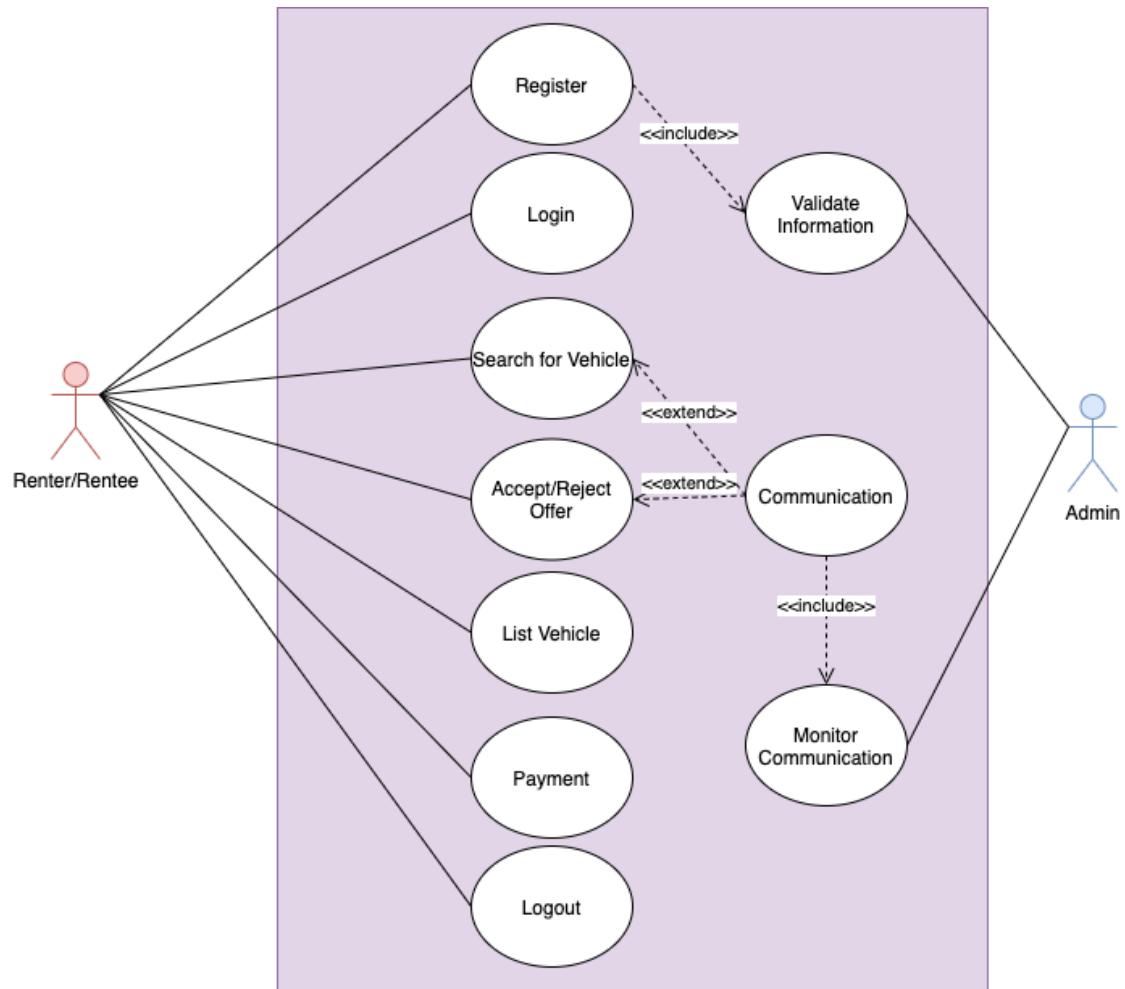


Figure 11 - Use Case Diagram

3.3.2 Class Diagram

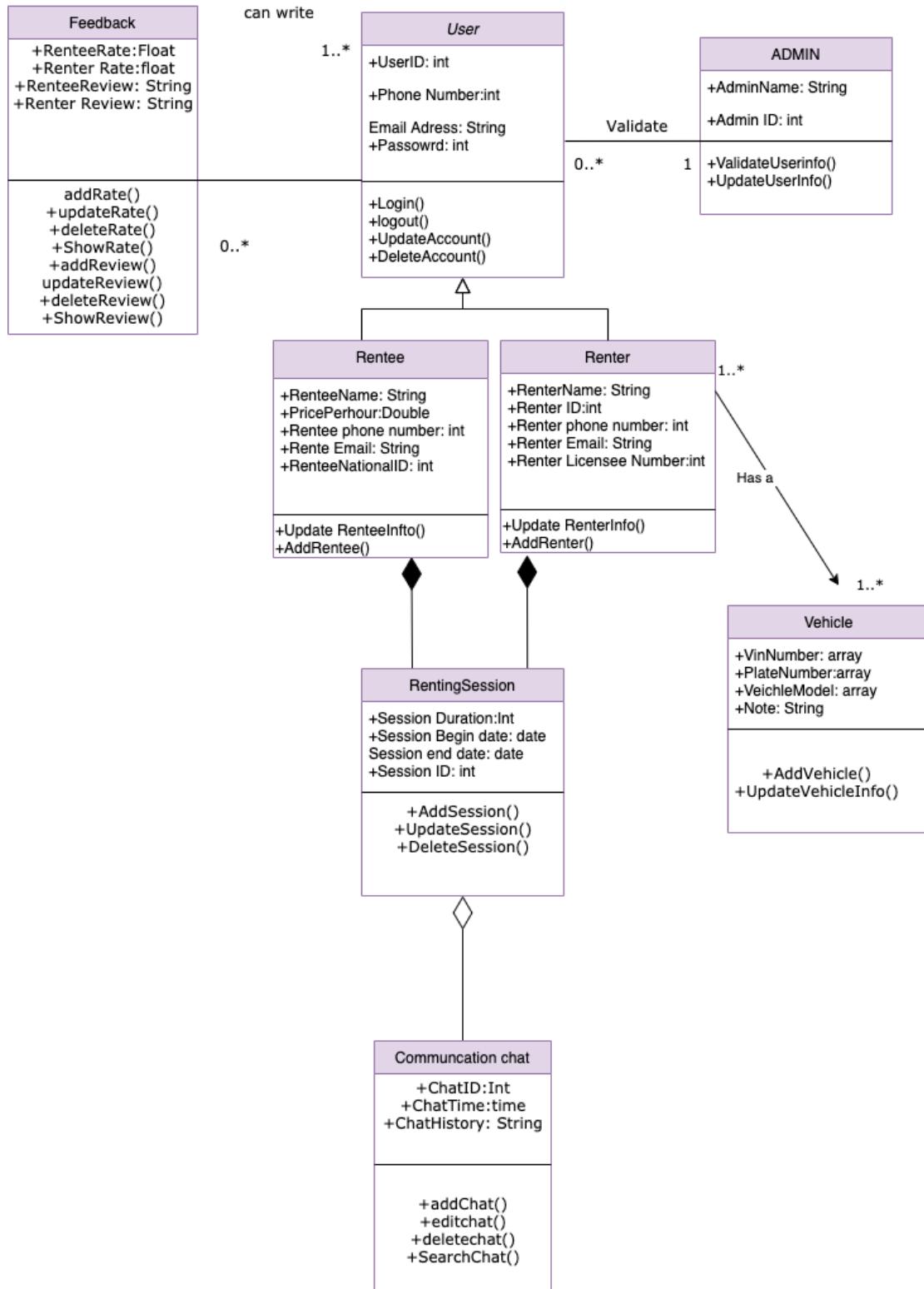


Figure 12 - Class Diagram

3.3.3 Sequence Diagram

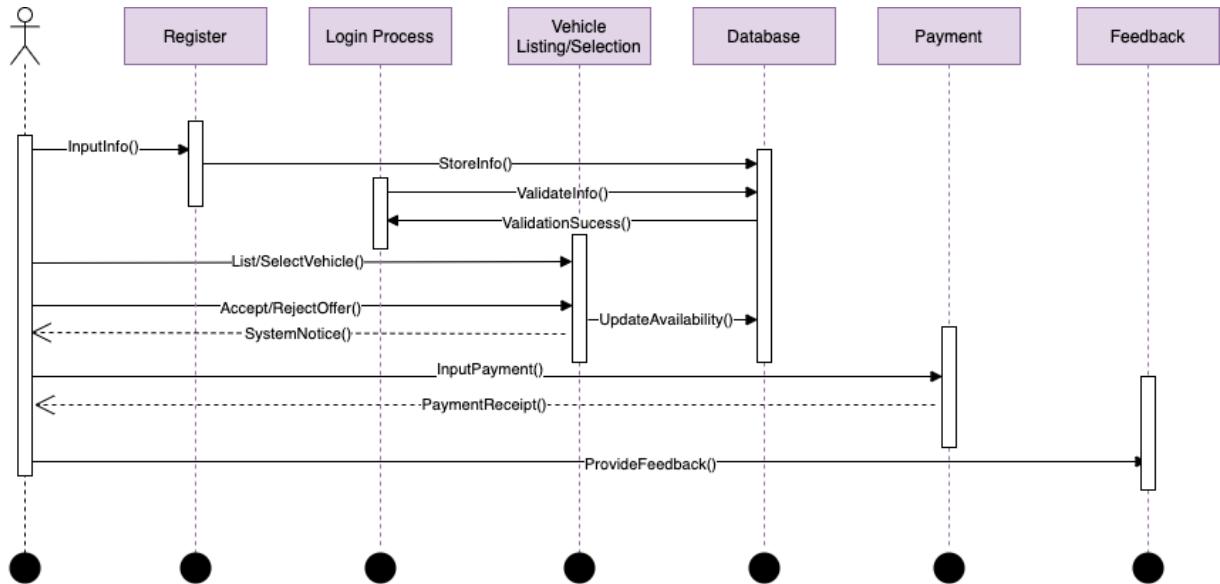


Figure 13 - Sequence Diagram

Chapter 4: System Design

4.1 System Architecture

In this section, the overall structure of the application will be demonstrated, as well as its elements, components, and how they operate to provide service for the users.

For our application, we chose the three-layered system [15] architecture which divides the system into the presentation layer, business layer, and data access layer. Each of the layers has a special role within the application. The presentation layer which is at the top of the architecture, specializes in handling the user interface, while the business layer handles performing specific request-related processes. The third and last layer in this architecture is the data access layer, which enables simplified access to data stored in permanent storage such as databases.

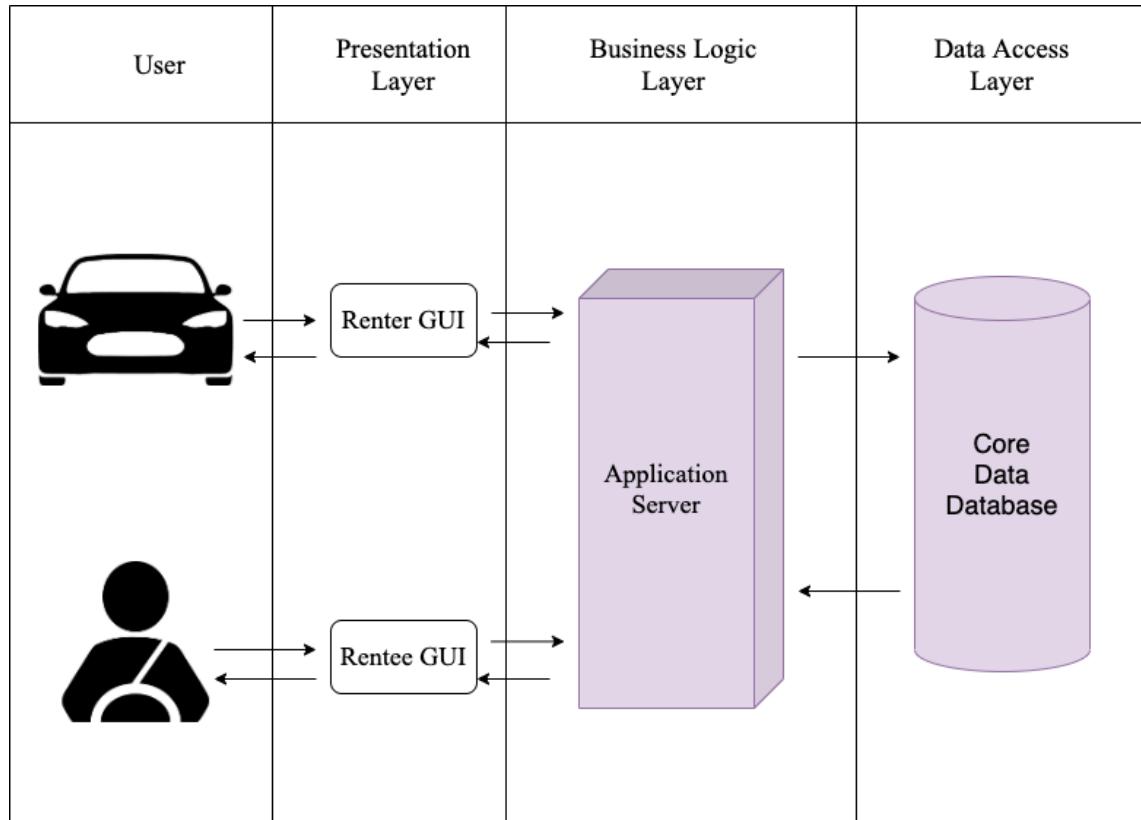


Figure 14 - System Architecture

4.1.1 Description of components

1. Users

The first and most important aspect of our system is the user, who could either be a renter or rentee. The interaction between these users and our application occurs in the presentation layer.

2. Presentation Layer

The primary function of this layer is the formatting and delivery of information to the business layer for further processing and display. The presentation layer provides services through using GUI to present content to the end-user, the GUI may be accessed through the client's mobile phones.

3. Business Logic Layer

The business logic layer is the middle stage. It contains the functional business logic that drove the core capabilities of an application. This limits the actions of the app to meet the needs of the users. Java, Swift, C#, Python, C++, etc

User operations in our application are processed and include registration, login, and logout. Rentees are able to perform operations, such as, searching for available vehicles to rent, requesting communication with a renter, and reviewing the experience afterward, which are all also processed in this layer.

As for the rentee, they're able to view requests and either accept or deny them, speak with rentees, and review the experience afterward.

4. Data Access Layer

Contact with external structures such as databases and networks are embedded in the data access layer. This breaks down to data being collected from either the Web or being cached in Core Data or User Defaults. For our application, we will be using Core Data as the basis.

4.2 System Design Tools

Several tools can be used to help facilitate system design such as flowcharts, UML as well as storyboards. To represent car2share design tools most effectively a flowchart is the optimal decision. A flowchart is a visual representation of the separate steps of a process in sequential order, it can also be used to describe various processes such as administrative or the user's process.

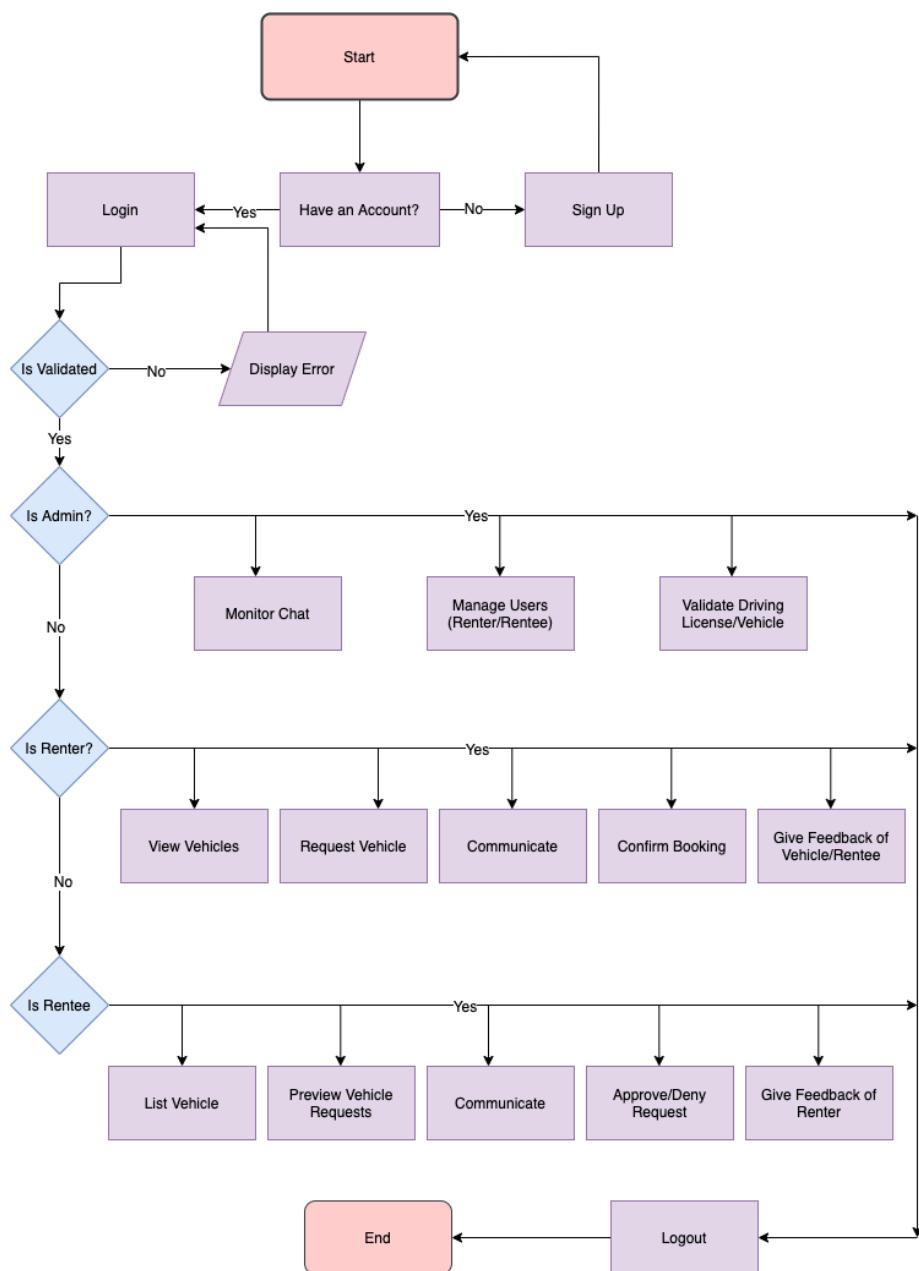


Figure 15 - System Design Tools

4.3 ERD

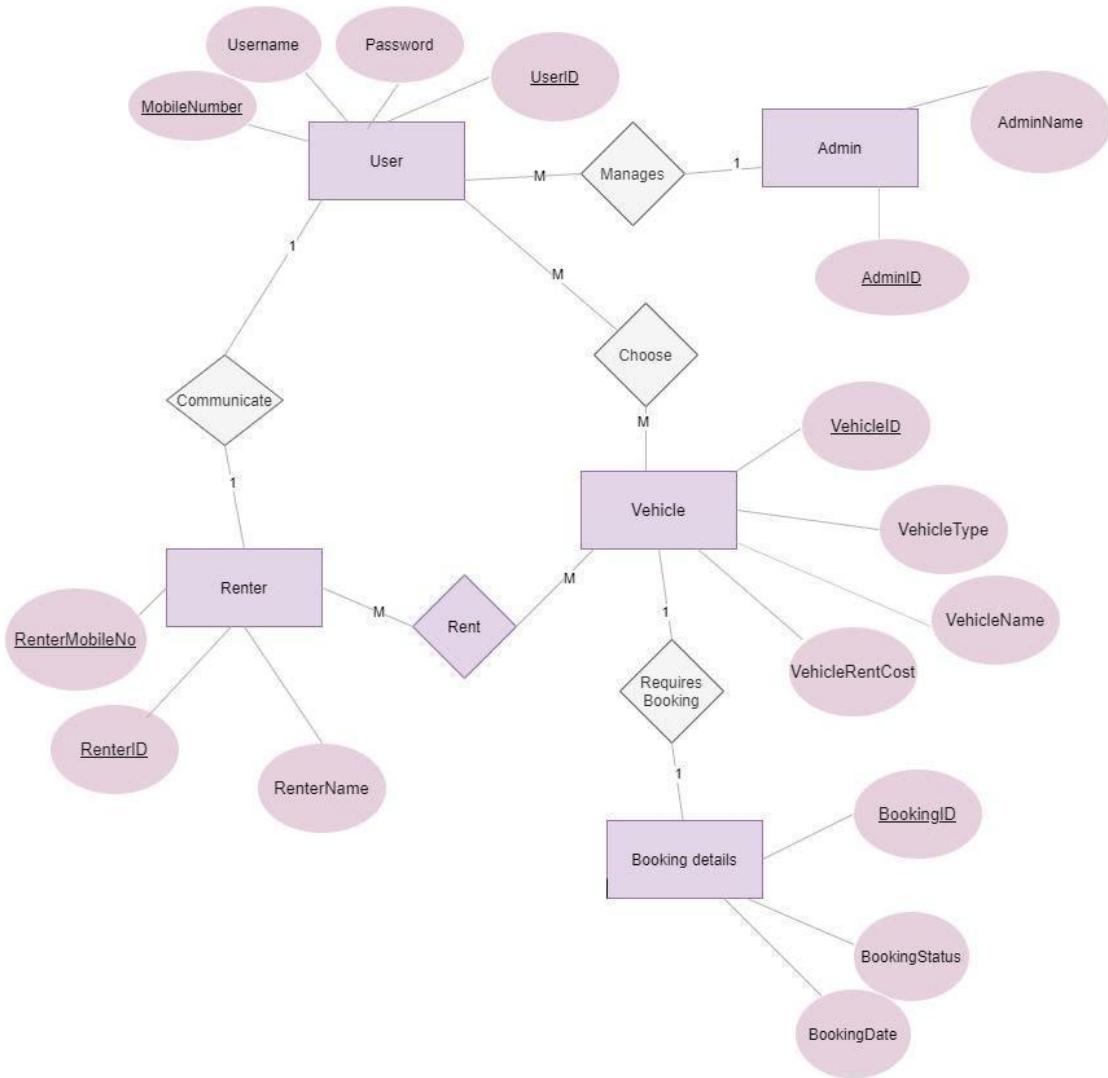
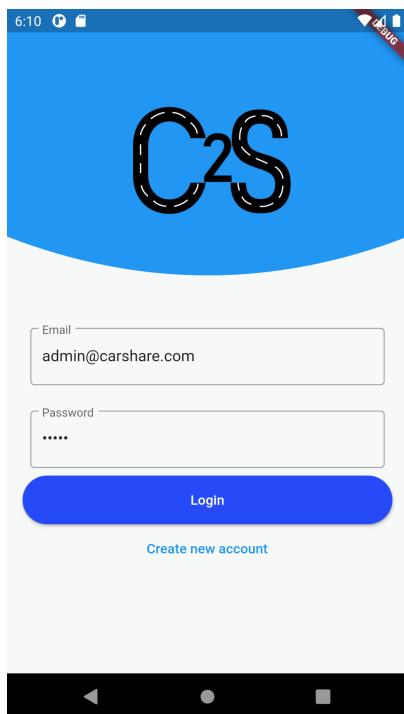


Figure 16 - ERD

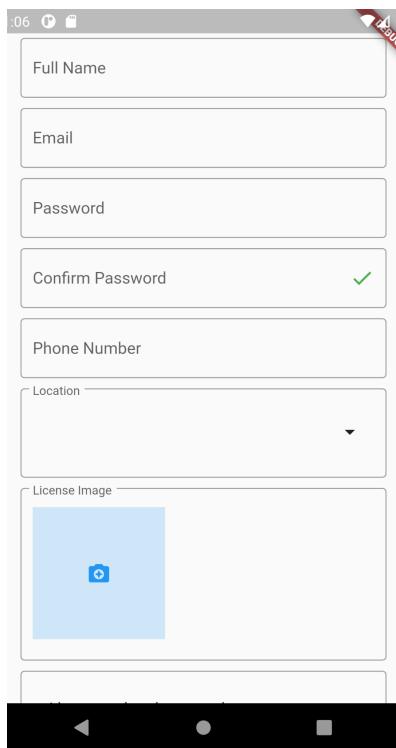
4.4 User Interface



For this interface, this is the first thing that's going to appear once the user launches the Car2Share app. The user can either choose to Log in to his/her existing account or to Sign up for a new account.

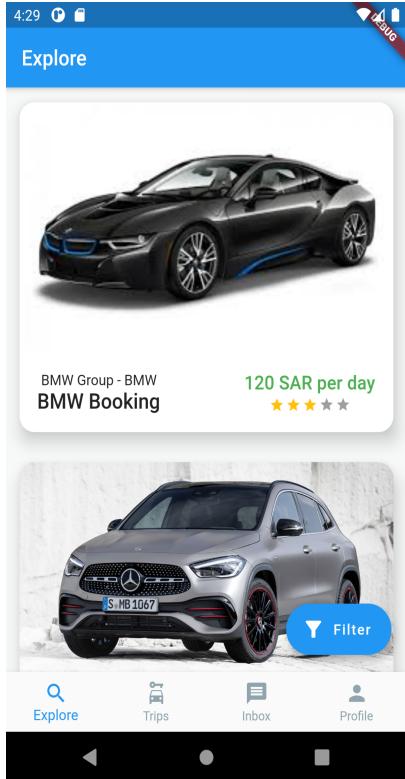
Once the user has chosen to Log in, he/she can either choose to enter the app by using their email as well as their password for verification purposes.

Figure 17 - Interface 1



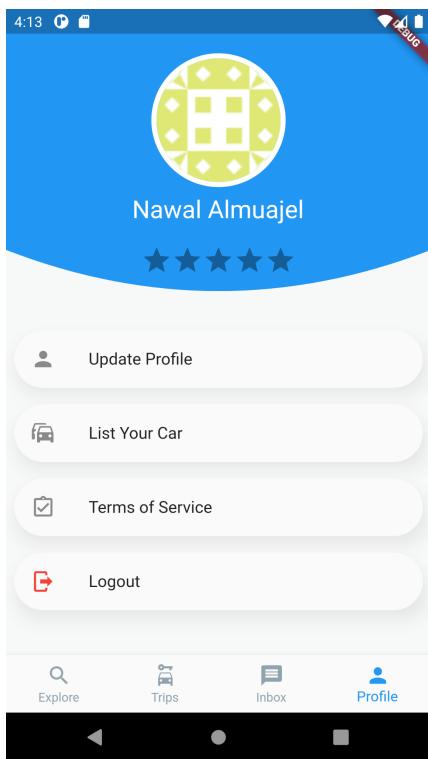
This interface has the sign-up page in which it prompts both the users, renter, and rentee to enter critical information such as the first and last name, email, password, phone number, and driving license ID to continue the sign-up process.

Figure 18- Interface 2



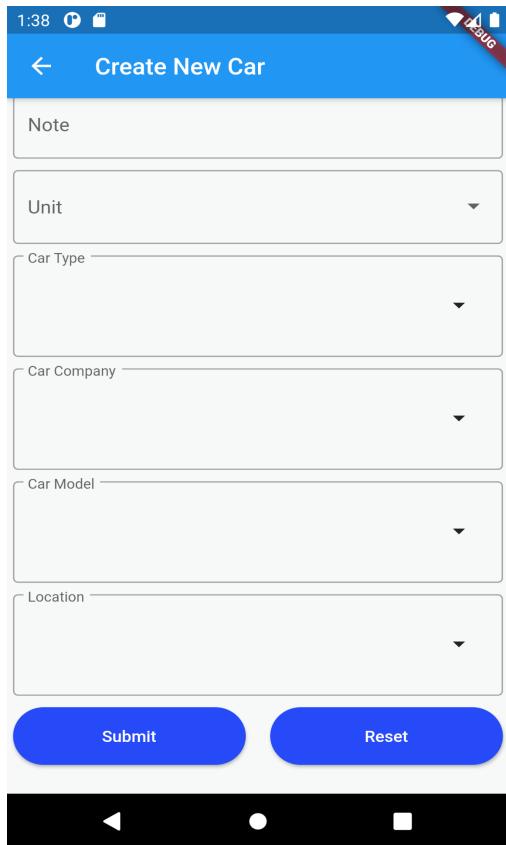
In this interface, it shows the explore page in which all the possible choices for the renter are displayed. Additionally, it also includes the filter feature which summarizes the user's options based on the user's preference and needs.

Figure 19 - Interface 3



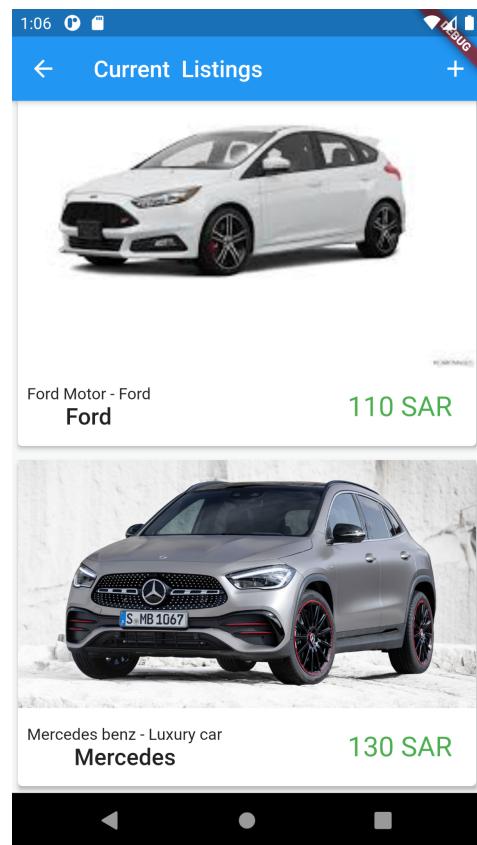
For the profile interface, the profile figure gives you the option to redirect to your personal information, Update profile, list your car, Current Listing, Term of Service. It also gives the user the option to log out.

Figure 20 - Interface 4



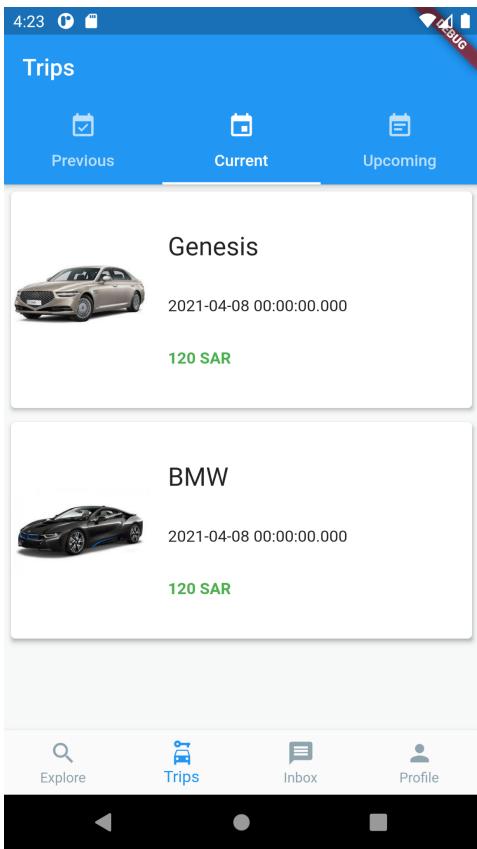
For the List your car interface, each user has the option to set up their car for listing and assign all the information needed.

Figure 21 - Interface 5



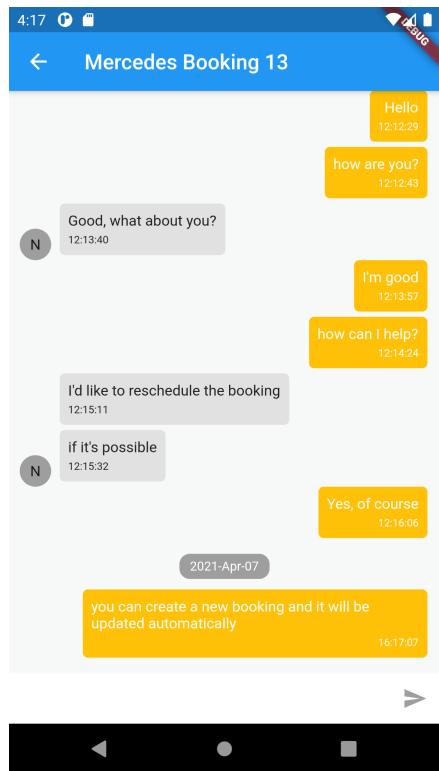
Once a vehicle has been registered, the user will be able to view their listings on the “Current Listings” page

Figure 22 - Interface 6



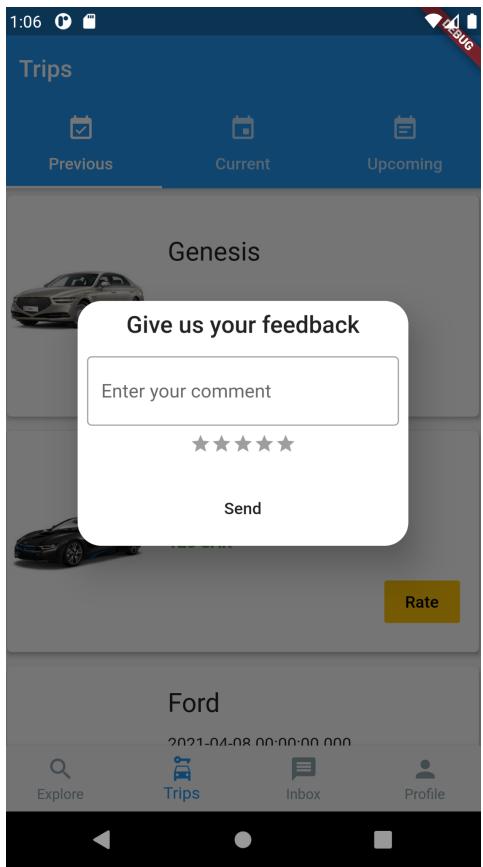
This interface allows the user to view previous trips. The page allows you to view the date, price, and more details about the trips the user has taken before and also the upcoming, current trips.

Figure 23 - Interface 7



This interface shows the exchange between a potential renter with the rentee where they specify the possible time and the availability of the requested car.

Figure 24 - Interface 8



The feedback interface enables the user to give feedback that car2share may benefit from. the feedback is provided through both filling out stars that resemble the level of satisfaction, and through enabling the user to express their opinion in a text box.

Figure 25 - Interface 9

Chapter 5: Implementation

Implementation

In this chapter, we will introduce the implementation techniques, hardware, and software that were used in this portion of the project. We will showcase snapshots of various interfaces of our system both in its last and revised version which may show some minor differences from our first perspective of the application in the previous chapters.

The system implementation is the fourth phase which is the Application Generation phase of the system life cycle of the RAD model. In this phase the project comes to life and begins to take shape, it also involves putting the final requirements and features together. The project becomes visible to the users by the end of this phase.

5.1 Implementation requirements

5.1.1 Hardware Requirements

In this project, the hardware requirements are:

1 - Personal Computer

- Handheld pointing device like a mouse
- Text entry device like a keyboard
- OS: Windows or Mac
- Possessor: 1.5 GHz o
- RAM: 2 GB o
- Wireless connectivity

2- Smartphone

- It gives us the ability to test and use the application.
- communicate between team members.

5.1.2 Software Requirements

In software requirements, the requirements are:

- Online Diagram Software to draw the UML diagrams
- Chrome to search about the needed information
- Microsoft Office programs such as Word processor, PowerPoint to document and present the research.
- MySQL database and web browser
- Web framework which is Laravel
- IntelliJ idea to analyze the code and to provides in-depth coding assistance, quick navigation, clever error analysis
- AndroidStudio to create an Android Virtual Device (AVD) that the emulator can use to install the Car2Share application

5.1.2.1 Admin interface

We will use two programming languages in admin interface implementation, Dart for the frontend and PHP for the backend. We will also use two programs to work and write the code, the first program, MAMP for database and host server, and the second program is PhpStorm it is a commercial, cross-platform IDE (integrated development environment) for PHP and works with our environment, which is Laravel valet.

5.1.2.1.1 Programming languages

1- Dart

Dart is a client-optimized programming language for apps on multiple platforms. It is used to build mobile, desktop, server, and web applications

2- PHP

It stands for "Hypertext Preprocessor." A prevalent scripting language that is used to create dynamic Web pages. PHP combining syntax from the C, Java, and Perl languages.

5.1.2.1.2 Programs

1- MAMP

It stands for "Macintosh, Apache, Mysql, and PHP." It is a local server environment used to create hosts, is a stack composed of open-source, and software used to develop dynamic websites. It also provides a Phpmyadmin, an administration tool for MySQL; it is a portable web application written by PHP.

2- PhpStorm

PHP is a server-side scripting language. that is used to develop Static websites or Dynamic websites or Web applications.

3- IntelliJ IDEA

IntelliJ IDEA is an Integrated Development Environment for application development. IntelliJ IDEA is one of the top powerful code editors and tools. Also, it supports many programming languages such as C++, Java, and Dart.

4- Emulators

The emulators are programs created to Create and manage virtual devices to test the application with real devices' capabilities; we will use Android Emulator to test our application on Android OS and use XCode to test our application on iOS OS.

5.1.2.1.2 Framework Environment

Laravel Valet

After in-depth research on framework environments, we found the Laravel valet is the best framework for our project's requirements. Valet is a framework environment for Php; it is a fully integrated environment that provides fast and easy development tools and packages with minimal resources. We chose Laravel valet in our project; because we do not need a fully virtualized development environment for this project, and Laravel valet only requires PHP and MySQL.

5.1.2.2 User Interface

We will use Flutter framework and dart programming language in user interface implementation for both frontend and backend. We will also, for coding, use IntelliJ IDEA to write the code. For the code testing, we will use two emulators, Android Emulator, to test the application on the Android operating system and XCode to test the application on the iOS operating system

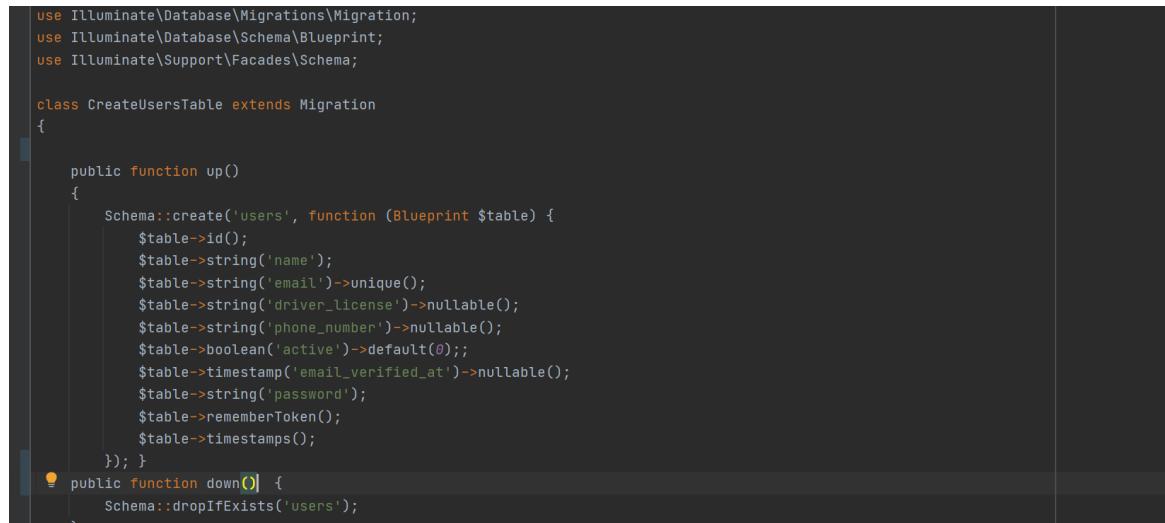
5.2 Implementation details

We are working in car2share to improve car rental by making it an easy and enjoyable experience. After research and drawing the use case, Entity-Relationship Diagram (ERD) and class diagram, sequence diagram. we will start programming by using PHP and Dart for admin and users. We will also use DB and coding programs, as explained in detail in the software requirements.

5.3 Code Screenshots

1-Database

For this PHP code, we have created a database table that contains rows and columns for ID, name, email, password, driver's license.

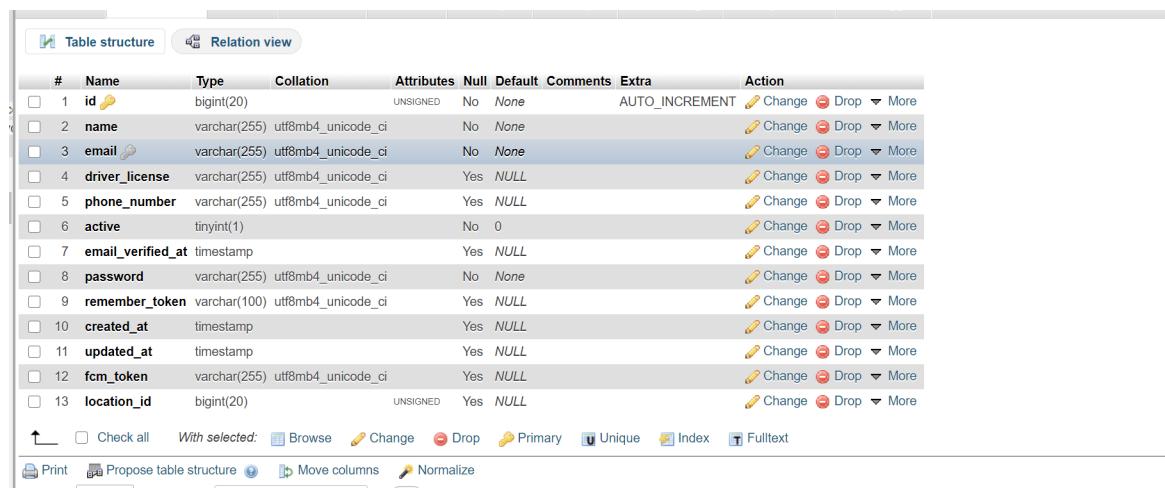


```
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateUsersTable extends Migration
{
    public function up()
    {
        Schema::create('users', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->string('email')->unique();
            $table->string('driver_license')->nullable();
            $table->string('phone_number')->nullable();
            $table->boolean('active')->default(0);
            $table->timestamp('email_verified_at')->nullable();
            $table->string('password');
            $table->rememberToken();
            $table->timestamps();
        });
    }

    public function down()
    {
        Schema::dropIfExists('users');
    }
}
```

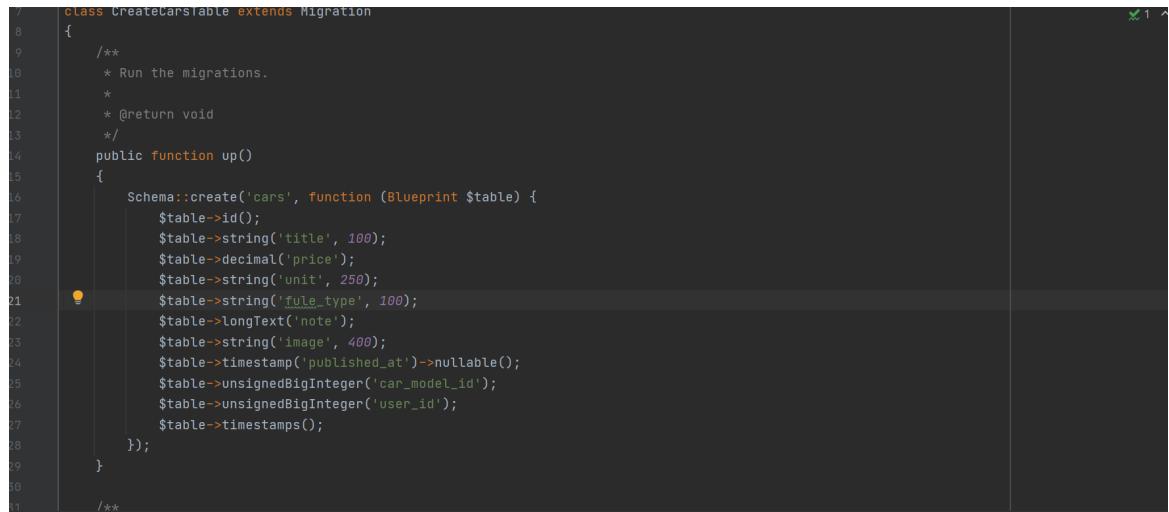
Figure 26 - Code 1



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	bigint(20)		UNSIGNED	No	None		AUTO_INCREMENT	Change Drop More
2	name	varchar(255)	utf8mb4_unicode_ci		No	None			Change Drop More
3	email	varchar(255)	utf8mb4_unicode_ci		No	None			Change Drop More
4	driver_license	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop More
5	phone_number	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop More
6	active	tinyint(1)			No	0			Change Drop More
7	email_verified_at	timestamp			Yes	NULL			Change Drop More
8	password	varchar(255)	utf8mb4_unicode_ci		No	None			Change Drop More
9	remember_token	varchar(100)	utf8mb4_unicode_ci		Yes	NULL			Change Drop More
10	created_at	timestamp			Yes	NULL			Change Drop More
11	updated_at	timestamp			Yes	NULL			Change Drop More
12	fcm_token	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop More
13	location_id	bigint(20)		UNSIGNED	Yes	NULL			Change Drop More

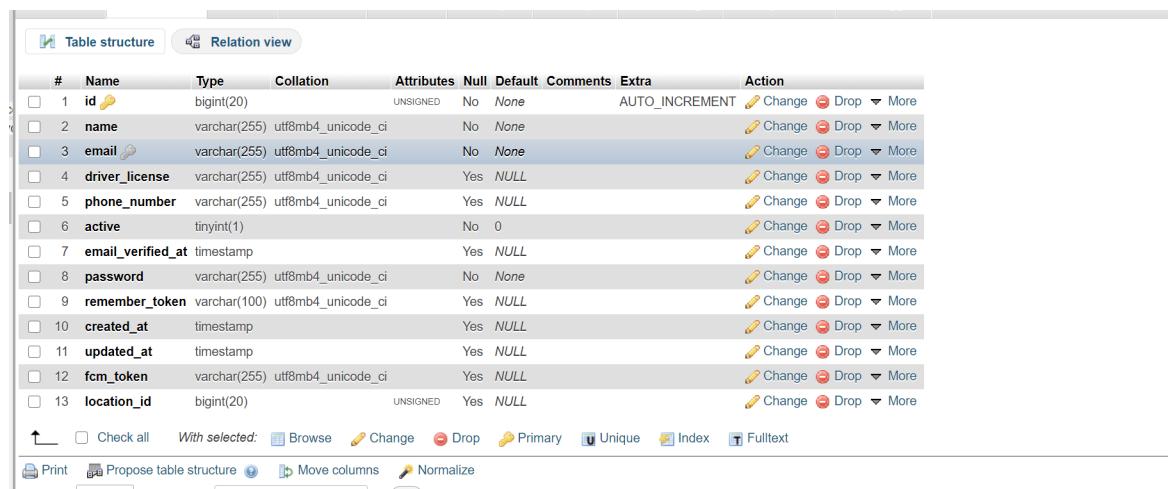
Figure 27 - Code 1 Output

And for this PHP code, we have created another database table for creating a car



```
1 // 
2 class CreateCarsTable extends Migration
3 {
4     /**
5      * Run the migrations.
6      *
7      * @return void
8      */
9     public function up()
10    {
11         Schema::create('cars', function (Blueprint $table) {
12             $table->id();
13             $table->string('title', 100);
14             $table->decimal('price');
15             $table->string('unit', 250);
16             $table->string('file_type', 100);
17             $table->longText('note');
18             $table->string('image', 400);
19             $table->timestamp('published_at')->nullable();
20             $table->unsignedBigInteger('car_model_id');
21             $table->unsignedBigInteger('user_id');
22             $table->timestamps();
23         });
24     }
25 
26     /**
27     */
28 }
```

Figure 28 - Code 2



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	<code>id</code>	bigint(20)		UNSIGNED	No	None		AUTO_INCREMENT	Change Drop More
2	<code>name</code>	varchar(255)	utf8mb4_unicode_ci		No	None			Change Drop More
3	<code>email</code>	varchar(255)	utf8mb4_unicode_ci		No	None			Change Drop More
4	<code>driver_license</code>	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop More
5	<code>phone_number</code>	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop More
6	<code>active</code>	tinyint(1)			No	0			Change Drop More
7	<code>email_verified_at</code>	timestamp			Yes	NULL			Change Drop More
8	<code>password</code>	varchar(255)	utf8mb4_unicode_ci		No	None			Change Drop More
9	<code>remember_token</code>	varchar(100)	utf8mb4_unicode_ci		Yes	NULL			Change Drop More
10	<code>created_at</code>	timestamp			Yes	NULL			Change Drop More
11	<code>updated_at</code>	timestamp			Yes	NULL			Change Drop More
12	<code>fcm_token</code>	varchar(255)	utf8mb4_unicode_ci		Yes	NULL			Change Drop More
13	<code>location_id</code>	bigint(20)		UNSIGNED	Yes	NULL			Change Drop More

Figure 29 - Code 2 Output

2- Users registration

In the user registration code, we have imported some dart packages to access built-in and user-defined packages into our dart source file so that each class can refer to a class that is in another package by directly using its name. Also, we have provided a registration form with email and password fields. When a user provides a username and password, the inputs need to be validated, as we have added some validation functions to the field

```
import 'package:car_2_share/home/bottom_navigation_page.dart';
import 'package:car_2_share/models/location.dart';
import 'package:car_2_share/models/token.dart';
import 'package:car_2_share/services/network.dart';
import 'package:dio/dio.dart';
import 'package:flutter/material.dart';
import 'package:flutter_custom_clippers/flutter_custom_clippers.dart';
import 'package:flutter_form_builder/flutter_form_builder.dart';
import 'package:form_builder_image_picker/form_builder_image_picker.dart';
import 'package:get/get.dart';

class RegisterPage extends StatefulWidget {
  static Route route() {
    return MaterialPageRoute<void>(builder: (_) => RegisterPage());
  }

  @override
  _RegisterPageState createState() => _RegisterPageState();
}

class _RegisterPageState extends State<RegisterPage> {
  final GlobalKey<FormBuilderState> _formKey = GlobalKey<FormBuilderState>();

  String _emailError, _phoneNumber;

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Container(
        padding: EdgeInsets.all(14.0),
        child: FormBuilder(
          key: _formKey,
          autovalidateMode: AutovalidateMode.onUserInteraction,
          child: Padding(
            padding: const EdgeInsets.all(14.0),
            child: Column(
              children: [
                const SizedBox(height: 10),
                FormBuilderTextField(
                  name: 'name',
                  decoration: InputDecoration(
                    labelText: 'Full Name',
                  ),
                  validator: FormBuilderValidators.compose([
                    FormBuilderValidators.required(context),
                  ]),
                ),
                const SizedBox(height: 10),
                FormBuilderTextField(
                  name: 'email',
                  decoration: InputDecoration(...),
                  validator: FormBuilderValidators.compose(...),
                ),
                const SizedBox(height: 10),
                FormBuilderTextField(
                  name: 'password',
                  decoration: InputDecoration(...),
                ),
              ],
            ),
          ),
        ),
      ),
    );
  }
}
```

Figure 30 - Code 3

```
  child: FormBuilder(
    key: _formKey,
    autovalidateMode: AutovalidateMode.onUserInteraction,
    child: Padding(
      padding: const EdgeInsets.all(14.0),
      child: Column(
        children: [
          const SizedBox(height: 10),
          FormBuilderTextField(
            name: 'name',
            decoration: InputDecoration(
              labelText: 'Full Name',
            ),
            validator: FormBuilderValidators.compose([
              FormBuilderValidators.required(context),
            ]),
          ),
          const SizedBox(height: 10),
          FormBuilderTextField(
            name: 'email',
            decoration: InputDecoration(...),
            validator: FormBuilderValidators.compose(...),
          ),
          const SizedBox(height: 10),
          FormBuilderTextField(
            name: 'password',
            decoration: InputDecoration(...),
          ),
        ],
      ),
    ),
  ),
}
```

Figure 31 - Code 4

```
    child: FormBuilder(
      key: _formKey,
      autovalidateMode: AutovalidateMode.onUserInteraction,
      child: Padding(
        padding: const EdgeInsets.all(14.0),
        child: Column(
          children: [
            const SizedBox(height: 10),
            FormBuilderTextField(
              name: 'name',
              decoration: InputDecoration(
                labelText: 'Full Name',
              ), // InputDecoration
              validator: FormBuilderValidators.compose([
                FormBuilderValidators.required(context),
              ]),
            ), // FormBuilderTextField
            const SizedBox(height: 10),
            FormBuilderTextField(
              name: 'email',
              decoration: InputDecoration(...), // InputDecoration
              validator: FormBuilderValidators.compose(...),
            ), // FormBuilderTextField
            const SizedBox(height: 10),
            FormBuilderTextField(
              name: 'password',
              decoration: InputDecoration(...),
            ),
          ],
        ),
      ),
    ),
  ),

```

Figure 32 - Code 5

2-Login page:

For the login code page, we have two text fields, email, a, and password, to get login/sign-in credentials from the user. Also, we have used a function that will validate the passed value for the email and password.

```
children: [
  const SizedBox(height: 30),
  Padding(
    padding: const EdgeInsets.all(3.0),
    child: FormBuilderTextField(
      name: 'email',
      initialValue: 'admin@carshare.com',
      decoration: InputDecoration(
        labelText: 'Email',
        errorText: _emailError,
      ), // InputDecoration
    ), // FormBuilderTextField, Padding, ListView
  ), // FormBuilder
], // Padding( // Container
padding: const EdgeInsets.all(3.0),

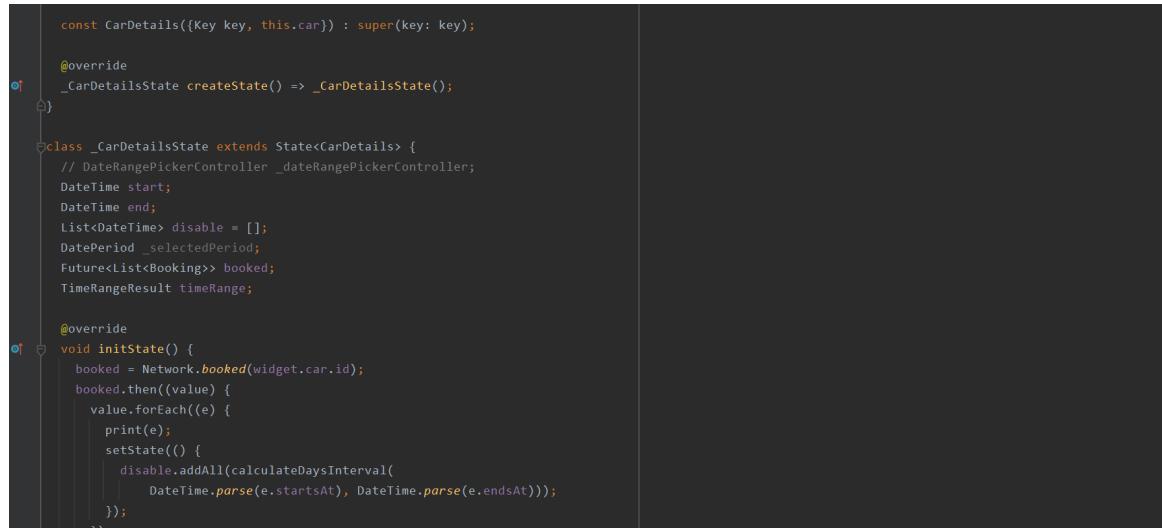
child: FormBuilderTextField(
  name: 'password',
  initialValue: '12345678',
  decoration: InputDecoration(labelText: 'Password'),
  obscureText: true,
  validator: FormBuilderValidators.compose([
    FormBuilderValidators.required(context),
    FormBuilderValidators.minLength(context, 8),
  ]),
),

```

Figure 33 - Code 6

3- Car details page

The renter set up their car for listing and assigned all the information needed, such as the booking date, car company, car type, car model, price.



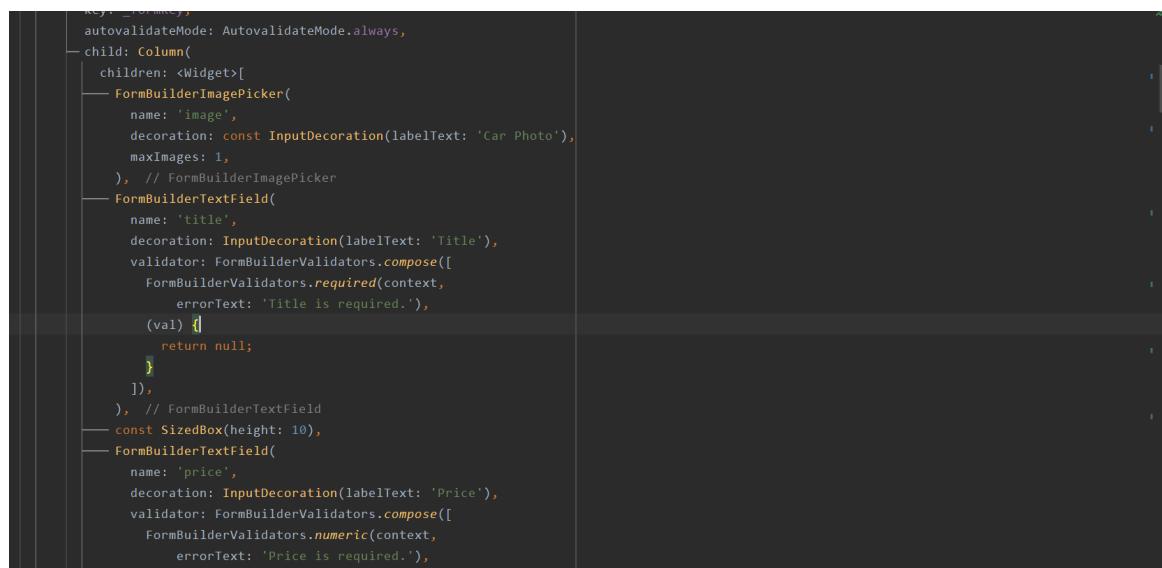
```
const CarDetails({Key key, this.car}) : super(key: key);

@Override
_CarDetailsState createState() => _CarDetailsState();
}

class _CarDetailsState extends State<CarDetails> {
// DateRangePickerController _dateRangePickerController;
DateTime start;
DateTime end;
List<DateTime> disable = [];
DatePeriod _selectedPeriod;
Future<List<Booking>> booked;
TimeRangeResult timeRange;

@Override
void initState() {
booked = Network.booked(widget.car.id);
booked.then((value) {
value.forEach((e) {
print(e);
setState(() {
disable.addAll(calculateDaysInterval(
DateTime.parse(e.startsAt), DateTime.parse(e.endsAt))));
});
});
}
}
```

Figure 34 - Code 7

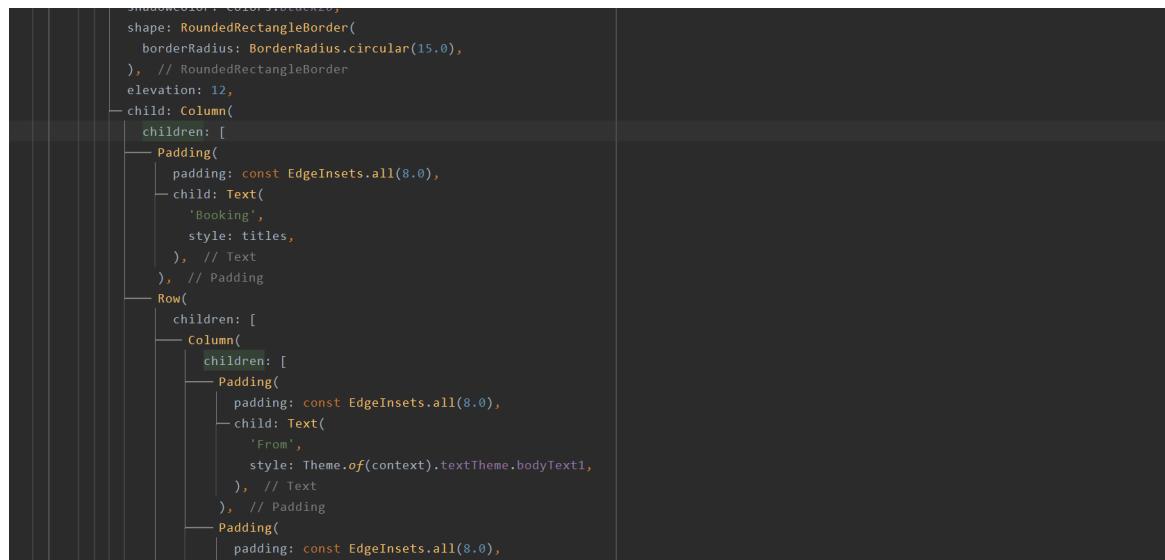


```
key: _carNameKey,
autovalidateMode: AutovalidateMode.always,
child: Column(
children: <Widget>[
FormBuilderImagePicker(
name: 'image',
decoration: const InputDecoration(labelText: 'Car Photo'),
maxImages: 1,
),
FormBuilderTextField(
name: 'title',
decoration: InputDecoration(labelText: 'Title'),
validator: FormBuilderValidators.compose([
FormBuilderValidators.required(context,
errorText: 'Title is required.'),
(val) =>
return null;
]),
),
const SizedBox(height: 10),
FormBuilderTextField(
name: 'price',
decoration: InputDecoration(labelText: 'Price'),
validator: FormBuilderValidators.compose([
FormBuilderValidators.numeric(context,
errorText: 'Price is required.'),
```

Figure 35 - Code 8

3- Trip details page

This code page allows users to view the date, price, and more details about the trips the user has taken before and also the upcoming, current trips.

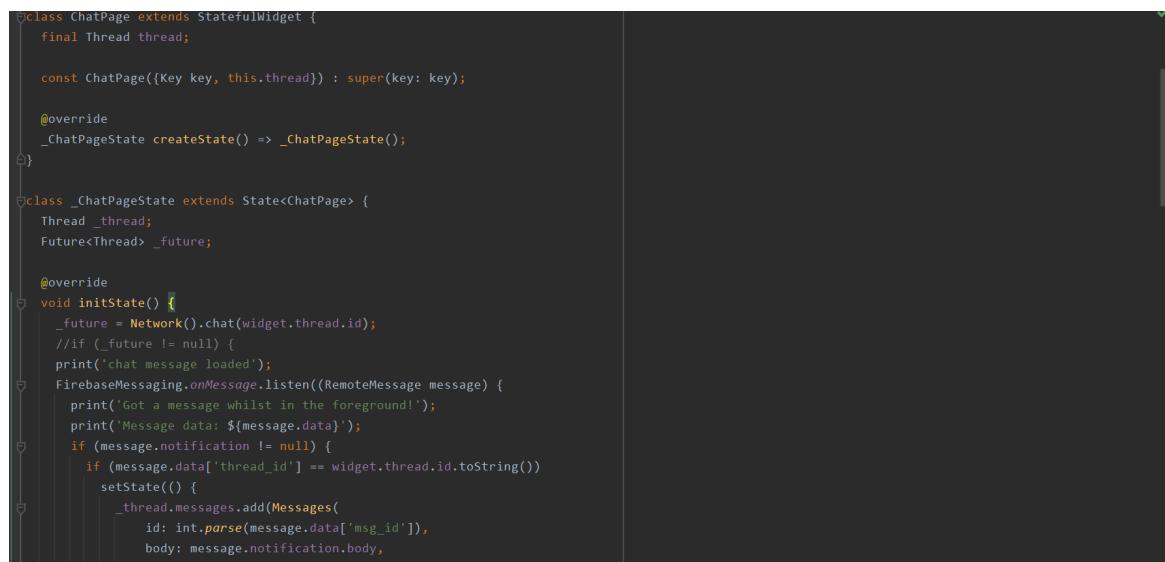


The screenshot shows a detailed view of a Flutter widget tree. It starts with a `Container` with a `shape: RoundedRectangleBorder`, `borderRadius: BorderRadius.circular(15.0)`, and `elevation: 12`. Inside is a `Column` containing a `Padding` with `padding: const EdgeInsets.all(8.0)`, a `Text` with the content 'Booking' and style `titles`, another `Padding`, and a `Row`. The `Row` contains a `Column` with a `Padding` and a `Text` with the content 'From' and style `Theme.of(context).textTheme.bodyText1`. Below this is another `Padding`.

Figure 36 - Code 9

4- Chat page

For this page, we have used the firebase cloud messaging package to enable a renter to send and receive messages in real-time through the communication chat



```
class ChatPage extends StatefulWidget {
    final Thread thread;

    const ChatPage({Key key, this.thread}) : super(key: key);

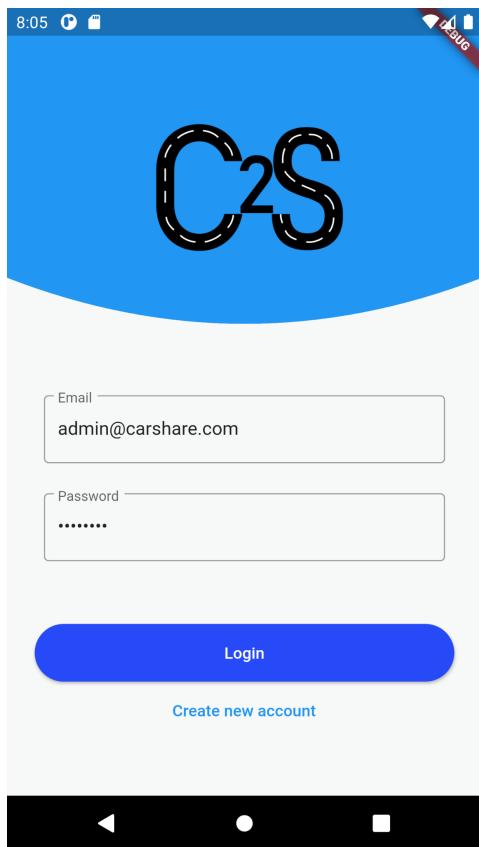
    @override
    _ChatPageState createState() => _ChatPageState();
}

class _ChatPageState extends State<ChatPage> {
    Thread _thread;
    Future<Thread> _future;

    @override
    void initState() {
        _future = Network().chat(widget.thread.id);
        //if (_future != null) {
        //print('chat message loaded');
        FirebaseMessaging.onMessage.listen((RemoteMessage message) {
            print('Got a message whilst in the foreground!');
            print('Message data: ${message.data}');
            if (message.notification != null) {
                if (message.data['thread_id'] == widget.thread.id.toString()) {
                    setState(() {
                        _thread.messages.add(Messages(
                            id: int.parse(message.data['msg_id']),
                            body: message.notification.body,
                        ));
                    });
                }
            }
        });
    }
}
```

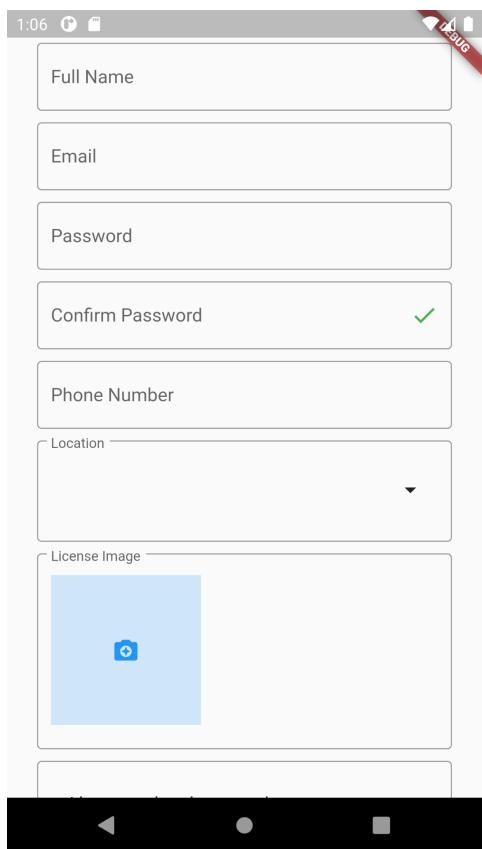
Figure 37 - Code 10

5.3.1 I/O Screens



When the user opens the Car2Share application, this is the first thing that appears. The user has the option of signing in to an existing account or creating a new one.

Figure 38 - I/O 1



The sign-up page of this interface asks users to enter essential details such as first and last names, email, password, phone number, and driving license image, and location in order to begin the sign-up process.

Figure 39 - I/O 2

The screenshot shows a mobile application interface for signing up a vehicle. At the top, there is a red banner with the text "BUG". Below it, there are four input fields: "Password", "Confirm Password" (with a green checkmark), "Phone Number", and "Location" (with a dropdown arrow). Under "Location", there is a placeholder "License Image" with a blue camera icon. Below these fields is a checkbox labeled "I have read and accept the terms of service." followed by a toggle switch. At the bottom is a large blue "Signup" button.

This interface allows the users to list their vehicle by inputting the required information.

Figure 40 - I/O 3

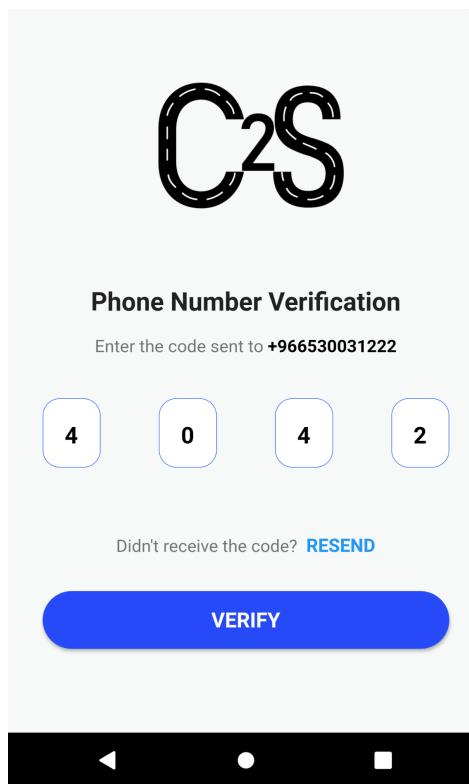
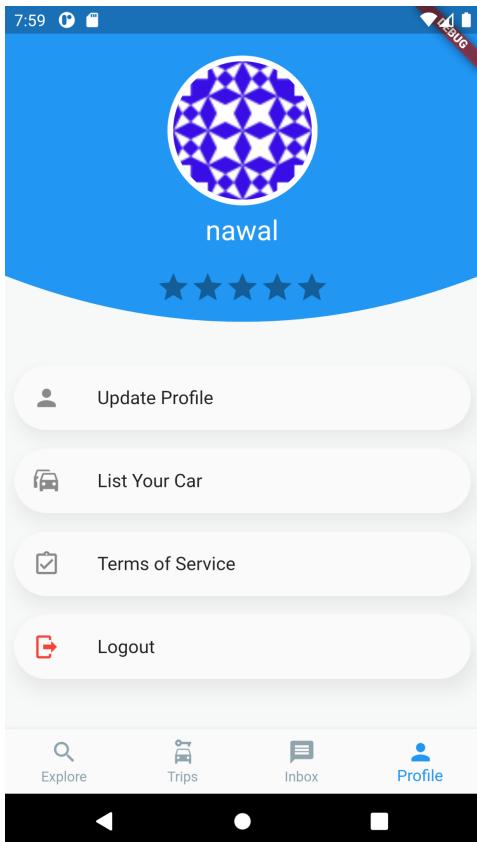


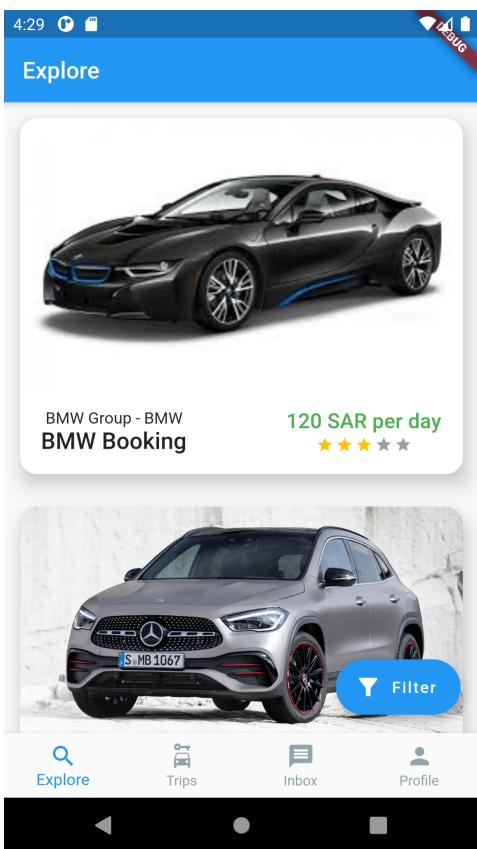
Figure 41 - I/O 4

After the user registers to car2share, a verification code will be sent to their mobile phone number to log in to their account successfully.



The profile page allows you to go to your personal details, update your profile, list your vehicle, view the Terms of Service. The user can also log out of the system.

Figure 42 - I/O 5



This interface displays the explore tab, which lists all of the options available to the renter. It also has a filter function that summarizes the user's choices based on their preferences and requirements.

Figure 43 - I/O 6



For this interface, it shows that this car can be booked by the date

Figure 44 - I/O 7

S	M	T	W	T	F	S
28	29	30	31	1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	1
2	3	4	5	6	7	8

From

12:00 AM 12:10 AM 12:20 AM 12:30 AM 12:40

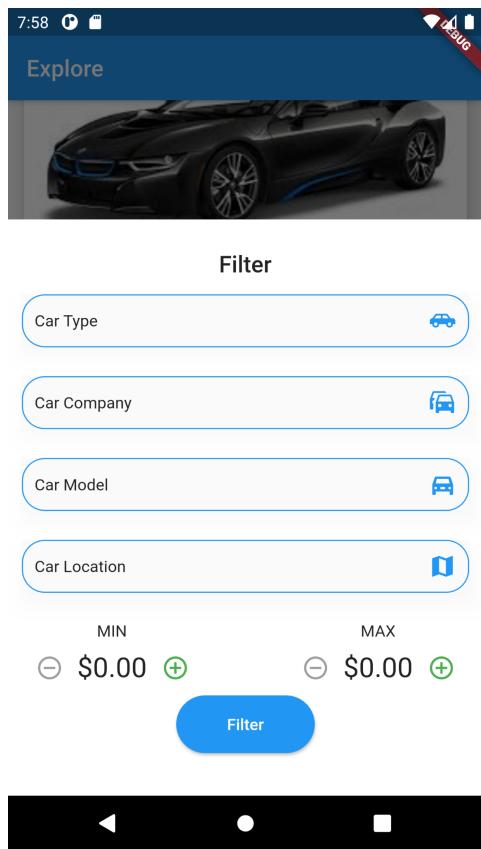
To

1:00 AM 2:00 AM 3:00 AM 4:00 AM 5:00

Confirm

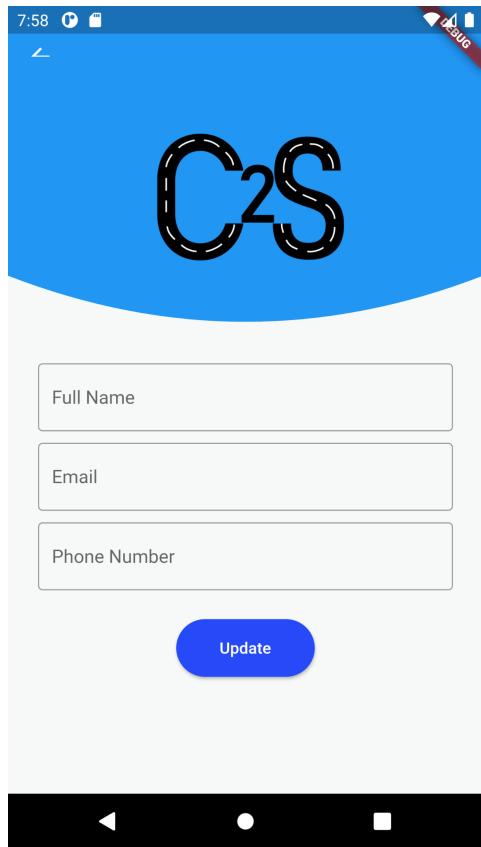
As for this interface, the user can book the car by the hour

Figure 45 - I/O 8



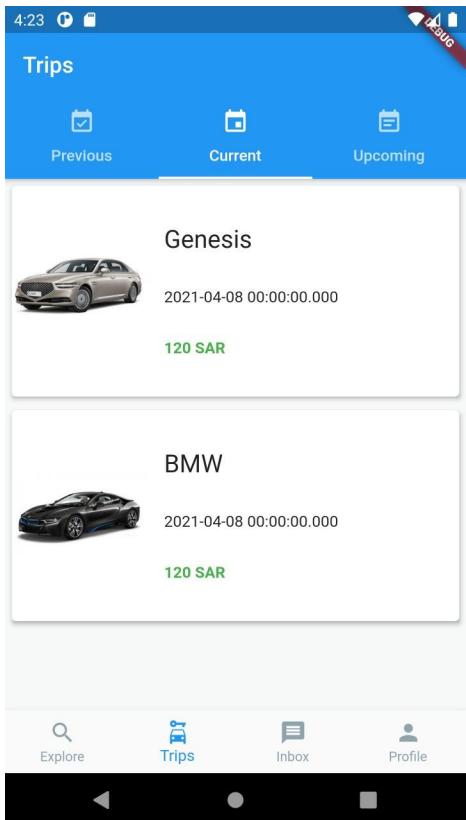
For this interface, the user can filter the cars on the explore page according to their preference of the car type, car company, car model, car location as well as car price

Figure 46 - I/O 9



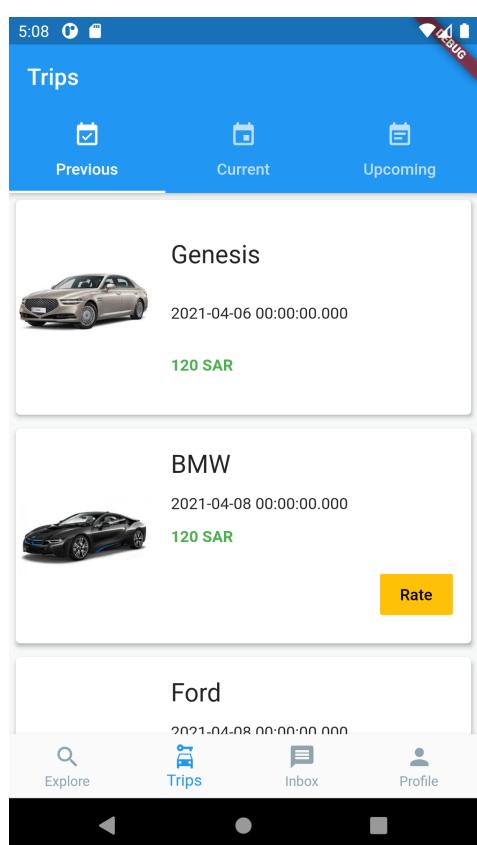
On the profile page, once the user clicks on the update profile, the users will be allowed to update and modify their name, email, phone number at any time if needed.

Figure 47 - I/O 10



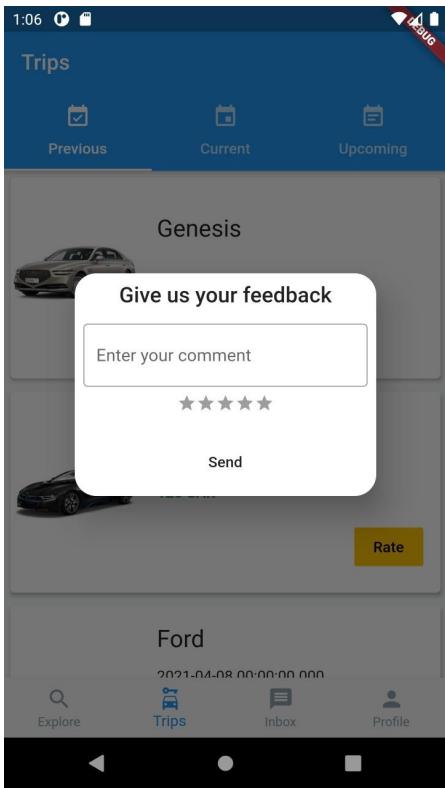
On the trips page, once the user clicks on the update profile, the users will be allowed to view previous, current and upcoming trips as well as the required information.

Figure 48 - I/O 11



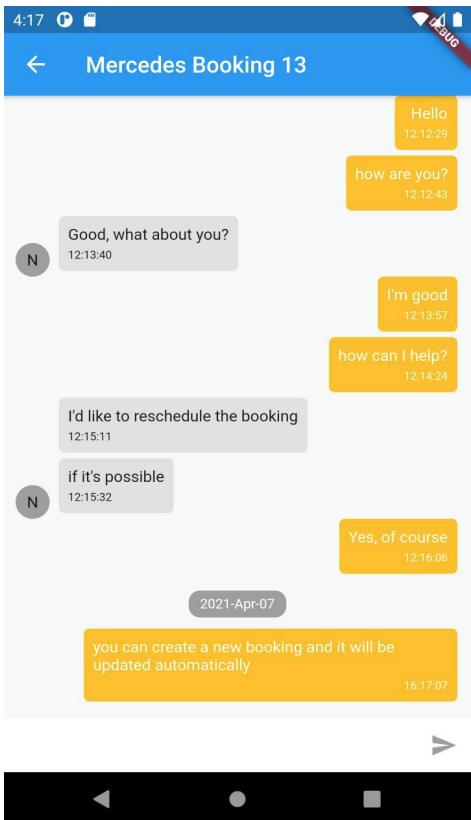
On the trips page, here we have the previous trips tab, the user will be permitted to rate their experience from the yellow Rate button.

Figure 49 - I/O 12



After clicking the Rate button, the interface will prompt the user to rate from 1-5 stars as well inputting any additional comment.

Figure 50 - I/O 13



On the inbox page, the rentee and renter will be able to communicate back and forth for any required communication, inquiry or extra information about the vehicle.

Figure 51 - I/O 14

Dashboard

When the admin accesses Car2Share, the first thing displayed will be the Dashboard in which the admin can manage, view, and monitor the information. These Dashboards organize, store and display the vital information that the admin must see and be aware of into one easily accessible place. In our dashboard, the admin can see the number of new users, newly registered car types, active users, booking count, and new cars in statistical charts. The admin has the choice of displaying the information for one day, one week, 30 days, or one year. Additionally, the admin has a panel displayed on the left of their screen that shows the options which showcase their various privileges in the application such as Car Companies, Car Models, Car Types, Cars (that are registered).

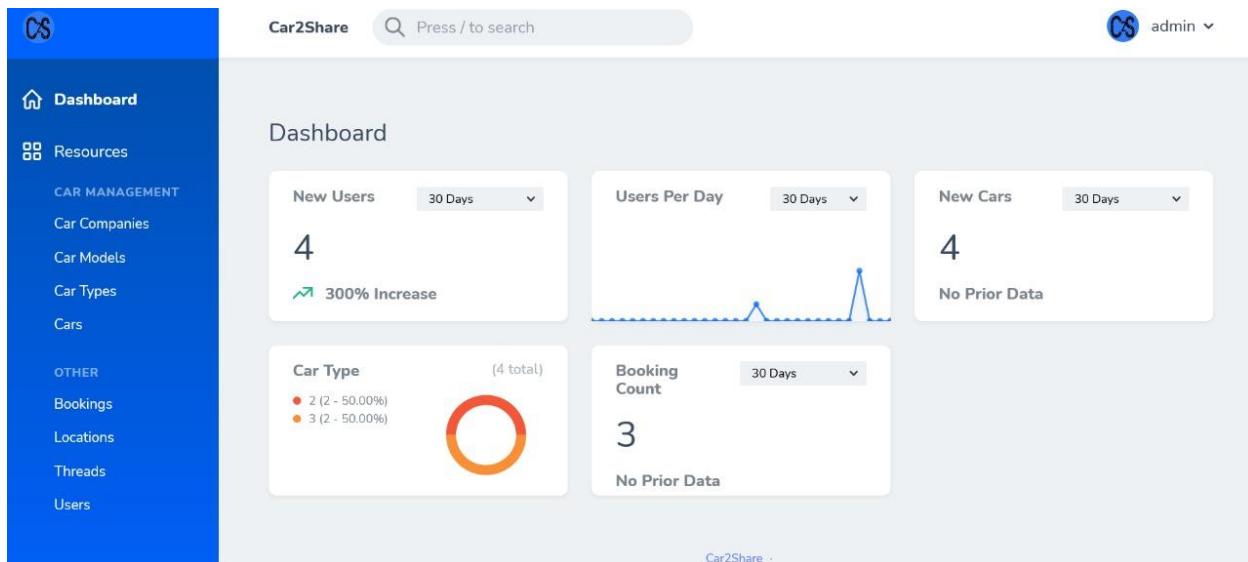
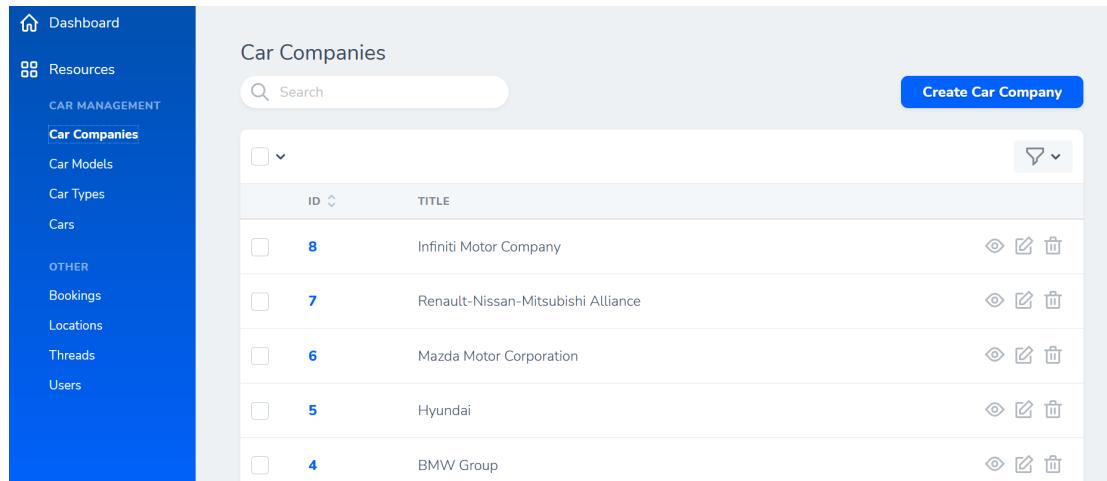


Figure 52 - Admin Interface 1

Car Companies page

When the admin opens the car companies page, it will display the list of the car companies that users can book. Also, The admin has the privilege to view, edit, delete any of the listed car companies. as well as creating new car companies.



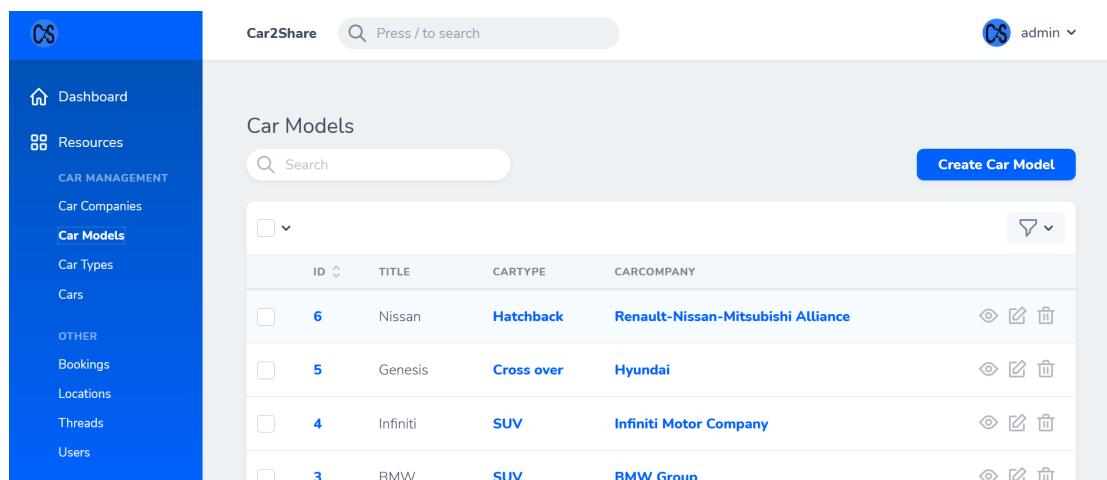
The screenshot shows the 'Car Companies' section of the admin interface. On the left, a sidebar menu includes 'Dashboard', 'Resources', 'CAR MANAGEMENT' (selected), 'Car Companies' (selected), 'Car Models', 'Car Types', 'Cars', 'OTHER' (disabled), 'Bookings', 'Locations', 'Threads', and 'Users'. The main area is titled 'Car Companies' with a search bar and a 'Create Car Company' button. A table lists six car companies: Infiniti Motor Company (ID 8), Renault-Nissan-Mitsubishi Alliance (ID 7), Mazda Motor Corporation (ID 6), Hyundai (ID 5), and BMW Group (ID 4). Each row has edit and delete icons.

ID	TITLE	Actions
8	Infiniti Motor Company	
7	Renault-Nissan-Mitsubishi Alliance	
6	Mazda Motor Corporation	
5	Hyundai	
4	BMW Group	

Figure 53 - Admin Interface 2

Car models page

On the car models page, it will display table details about each car, including; the title, car type, and the car company.



The screenshot shows the 'Car Models' section of the admin interface. The sidebar is identical to Figure 53. The main area is titled 'Car Models' with a search bar and a 'Create Car Model' button. A table lists five car models: Nissan (Hatchback, Renault-Nissan-Mitsubishi Alliance), Genesis (Cross over, Hyundai), Infiniti (SUV, Infiniti Motor Company), and BMW (SUV, BMW Group). Each row has edit and delete icons.

ID	TITLE	CARTYPE	CARCOMPANY	Actions
6	Nissan	Hatchback	Renault-Nissan-Mitsubishi Alliance	
5	Genesis	Cross over	Hyundai	
4	Infiniti	SUV	Infiniti Motor Company	
3	BMW	SUV	BMW Group	

Figure 54 - Admin Interface 3

Bookings page

From the admin's control panel, the admin has access to the bookings page from which they can see all bookings that took place, the dates of those bookings, the prices, users, and types of cars booked.

The screenshot shows the 'Bookings' page of the Admin Interface. On the left is a sidebar with navigation links: Dashboard, Resources, CAR MANAGEMENT (Car Companies, Car Models, Car Types, Cars), OTHER (Locations, Threads, Users), and Bookings (which is currently selected). The main area is titled 'Bookings' with a search bar and a 'Create Booking' button. A table lists three bookings:

ID	STARTS_AT	ENDS_AT	PRICE	USER	CAR	Actions
3	2021-03-20 12:00:00 AM	2021-03-21 12:00:00 AM	120	SaraFahad	BMW Booking	
2	2021-03-10 12:00:00 AM	2021-03-17 12:00:00 AM	840	Nawal Almuajel	Genesis Booking Request	
1	2021-03-03 12:00:00 AM	2021-03-04 12:00:00 AM	120	Nawal Almuajel	BMW Booking	

Below the table are 'Previous' and 'Next' navigation buttons, and at the bottom is a 'Car2Share .'

Figure 55 - Admin Interface 4

Threads page

On this page, the admin has full access to all communication that takes place between the renter and the rentee.

The screenshot shows the 'Threads' page of the Admin Interface. The sidebar is identical to the 'Bookings' page, with the 'Threads' link also being selected. The main area is titled 'Threads' with a search bar and a 'Create Thread' button. A table lists four threads:

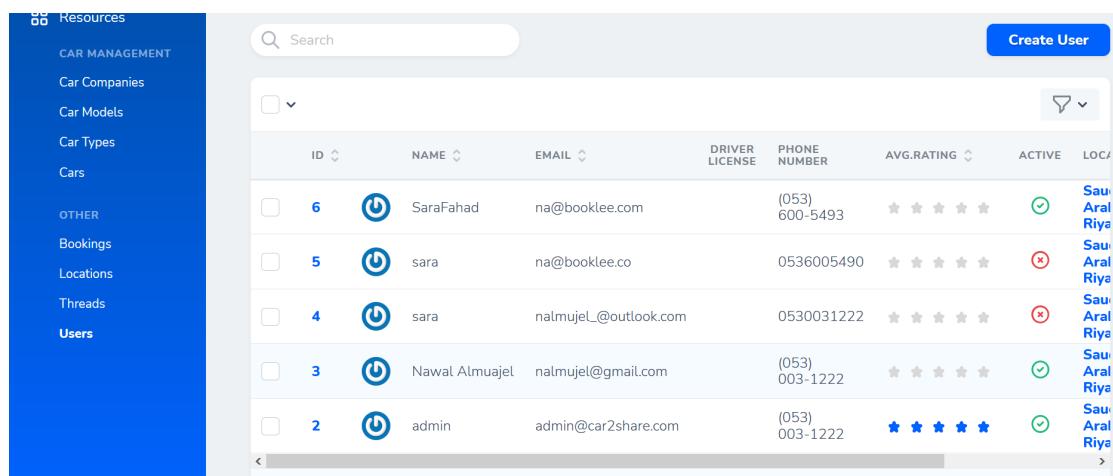
ID	SUBJECT	Actions
12	Car Booking	
11	Mercedes Booking	
10	Booking request	
9	Booking Request	

Figure 56 - Admin Interface 5

Users page

Users are managed from the Admin page through the Users tab. In the Users table, the admin can edit and delete users in our system. the table provides the following information about Car2share users:

- **ID:** The user's ID in car2share.
- **Name** The name of the user.
- **Email:** The user's email address.
- **Driver's license:** This shows an image of the Driver's license
- **Phone Number:** This shows the phone number of the user
- **Rating:** Users give rating points and the system takes an average rating and displays it in the overall review summary.
- **Active:** It shows if the users are active/inactive
- **Location:** It shows the registered location of each user.



The screenshot shows the Admin Interface with the 'Users' tab selected. On the left, there is a sidebar with 'CAR MANAGEMENT' and 'OTHER' sections. Under 'CAR MANAGEMENT', there are links for 'Car Companies', 'Car Models', 'Car Types', and 'Cars'. Under 'OTHER', there are links for 'Bookings', 'Locations', 'Threads', and 'Users'. The main area features a search bar and a 'Create User' button. Below is a table with columns: ID, NAME, EMAIL, DRIVER LICENSE, PHONE NUMBER, AVG.RATING, ACTIVE, and LOCATION. The table contains five rows of data:

ID	NAME	EMAIL	DRIVER LICENSE	PHONE NUMBER	AVG.RATING	ACTIVE	LOCATION
6	SaraFahad	na@booklee.com	(053) 600-5493	★ ★ ★ ★ ★	✓	Sau	
5	sara	na@booklee.co	0536005490	★ ★ ★ ★ ★	✗	Aral	
4	sara	nalmujel_@outlook.com	0530031222	★ ★ ★ ★ ★	✗	Riya	
3	Nawal Almuajel	nalmujel@gmail.com	(053) 003-1222	★ ★ ★ ★ ★	✓	Sau	
2	admin	admin@car2share.com	(053) 003-1222	★ ★ ★ ★ ★	✓	Aral	

Figure 57 - Admin Interface 6

Chapter 6: Testing

Testing:

Chapter 6 will cover the testing phase of the implementation process. The testing phase is very important as it allows us to try out the application before officially “going live”, working out any possible issues, and ensuring the application is in its best form. In this chapter, the methods used to test will be presented. Specifically, the Android Emulator.

6.1 Testing Plan

The testing plan is the portion of this report in which the scope of the testing and activities are detailed. The Android Emulator, provided by Android studio, was used to test all aspects and functions of the system. The details will be covered in this report. The system was tested repeatedly during the implementation phase. It included testing all the functionality, performance, features, and database. The testing strategy begins with the unit components, moving forward to the system’s entire functionality, and finally with the user acceptance testing (UAT).

The project was separated into two parts, the admin interface, and the user interface. The admin interface was made as a website while the user interface is the application. Once those were established, the work was segmented into units and tested individually after implementation to ensure that any errors were identified and fixed.

6.2 Testing Strategy

6.2.1 Unit Testing

The primary goal of unit testing is to divide the system into the part and test them each individually. That is the process followed for Car2Share to help ensure that each unit behaves ideally before it is integrated into the system as a whole.

6.2.2 Functional Testing

The functional testing helps ensure the application is meeting the set specifications appropriately. Thus, the performance of the system is tested to examine the functionality, performance, features, database, communication chat, and security, ensuring that it simulates the system's real use accurately.

6.2.3 Acceptance Testing

Also known as UAT, acceptance testing is the most critical phase of testing. It is the factor that decides if the client accepts the interface or not. As well, the test aims to assess compliance with business requirements which involves usability, functionality, user interface, and performance.

6.3 Test Cases

6.3.1 Admin Test Case

Table 6.1 – Admin Test Case

Unit	Test Case	Expected Result	Actual Result	Success/Fail
	Login/Logout	Login: Admin should be able to login into the application. Logout: Admin should be able to log out of the application.	As expected	Success
	Add	Enable the admin to add new information whether it be about the users or the vehicles.	As expected	Success
	Delete	Enables the admin to delete information about either the users or the vehicles.	As expected	Success
	Edit	Enables the admin to edit information about either the users or the vehicles.	As expected	Success
	Search	Available on all pages, allowing the admin to search by ID, vehicle, or username.	As expected	Success
	Check Password	Checks password quality when registering, later checks if the password is correct when signing in.	As expected	Success
	View Information	Enables admins to access all user information, excluding passwords, as well as all service information.	As expected	Success
	Filter	Enables renter to filter the through table as well as the vehicles table.	As expected	Success

6.3.2 Renter Test Case

Table 6.2 – Renter Test Case

Unit	Test Case	Expected Result	Actual Result	Success/Fail
	Login/Logout	Login: renter should be able to login into the application. Logout: renter should be able to log out of the application.	As expected	Success
	Create Account	Enables renter to create a new account.	As expected	Success
	Check Password	Checks password quality when registering, later checks if the password is correct when signing in.	As expected	Success
	Authentication	Renter will receive an SMS when registering to verify the inputted phone number	As expected	Success
	List Vehicle	Enables the renter to list their vehicle on the application.	As expected	Success
	Delete	Enables the renter to delete information about either the vehicle/s listed or their account.	As expected	Success
	Edit	Enables the renter to edit information about either the vehicle/s listed or their account.	As expected	Success
	Change Password	Enables renter to change their account password if needed	As expected	Success
	Search	Enables the renter to search any page in the application.	As expected	Success
	View	Enables renter to view all pages that are in the application.	As expected	Success
	Communicate Through Chat	Enables renter to send and receive messages in real-time through the communication chat	As expected	Success
	Feedback	Enables renter to review the experience after it has ended	As expected	Success

6.3.3 Rentee Test Case

Table 6.3 – Admin Test Case

Unit	Test Case	Expected Result	Actual Result	Success/Fail
	Login/Logout	Login: rentee should be able to login into the application. Logout: rentee should be able to log out of the application.	As expected	Success
	Create Account	Enables rentee to create a new account.	As expected	Success
	Check Password	Checks password quality when registering, later checks if the password is correct when signing in.	As expected	Success
	Authentication	Rentee will receive an SMS when registering to verify the inputted phone number	As expected	Success
	Delete	Enables the rentee to delete information on their account.	As expected	Success
	Edit	Enables the rentee to edit information on their account.	As expected	Success
	Change Password	Enables rentee to change their account password if needed	As expected	Success
	Search	Enables the rentee to search any page in the application.	As expected	Success
	View	Enables renter to view all pages that are in the application.	As expected	Success
	Filter	Enables rentee to filter through the given options on the explore page	As expected	Success
	Communicate Through Chat	Enables rentee to send and receive messages in real-time through the communication chat	As expected	Success
	Feedback	Enables rentee to review the experience after it has ended	As expected	Success

6.4 Test Results

6.4.1 Admin Test Results

Table 6.4 – Admin Test Case Results

Unit	Test Case	Percentage of Completeness
	Login/Logout	100%
	Add	100%
	Delete	100%
	Edit	100%
	Search	100%
	Check Password	100%
	View Information	100%
	Filter	100%

6.4.2 Renter Test Results

Table 6.5 – Renter Test Case Results

Unit	Test Case	Percentage of Completeness
	Login/Logout	100%
	Create Account	100%
	Check Password	100%
	Authentication	100%
	List Vehicle	100%
	Delete	100%
	Edit	100%
	Change Password	100%

	Search	100%
	View	100%
	Communicate Through Chat	100%
	Feedback	100%

6.4.2 Rentee Test Results

Table 6.6 – Rentee Test Case Results

Unit	Test Case	Percentage of Completeness
	Login/Logout	100%
	Create Account	100%
	Check Password	100%
	Authentication	100%
	Delete	100%
	Edit	100%
	Change Password	100%
	Search	100%
	View	100%
	Filter	100%
	Communicate Through Chat	100%
	Feedback	100%

Chapter 7: Conclusion

7.1 Evaluation

By spending a lot of time studying and exploring the best ways to tackle our project, we were able to fulfill our targets at the end of the project. We aimed to modernize the conventional car rental process, making it more beneficial to both customers (the renter and the rentee). The test results show that Car2Share has been developed successfully and without errors. Our project aims to make the process of renting a car as easy, convenient, and effective as possible. A high degree of protection has also been achieved, as part of the registration process on Car2Share, the user is required to provide a verification number via phone number.

7.2 Future Work

In the future, we hope to have Car2Share available on the App Store and Playstore, we hope to accumulate numerous investors and sponsors that help elevate and improve the application to expand beyond the Middle East. In the future, we aim to include several more features like “best offer”, discounts and coupons. Additionally, we hope to maintain the app’s usability constantly through continuous updates as we strive for a user-friendly experience that enhances the efficiency of the system and encourages our customers to share and use the application regularly.

7.3 Conclusion

In conclusion, Car2Share is a peer-to-peer car rental service primarily based in the MENA area. Our application is unique in that it provides new sources of income for its users who want to put their car up for rental, it provides people that have a license and require a car with the fastest way to access the nearest car of their preference.

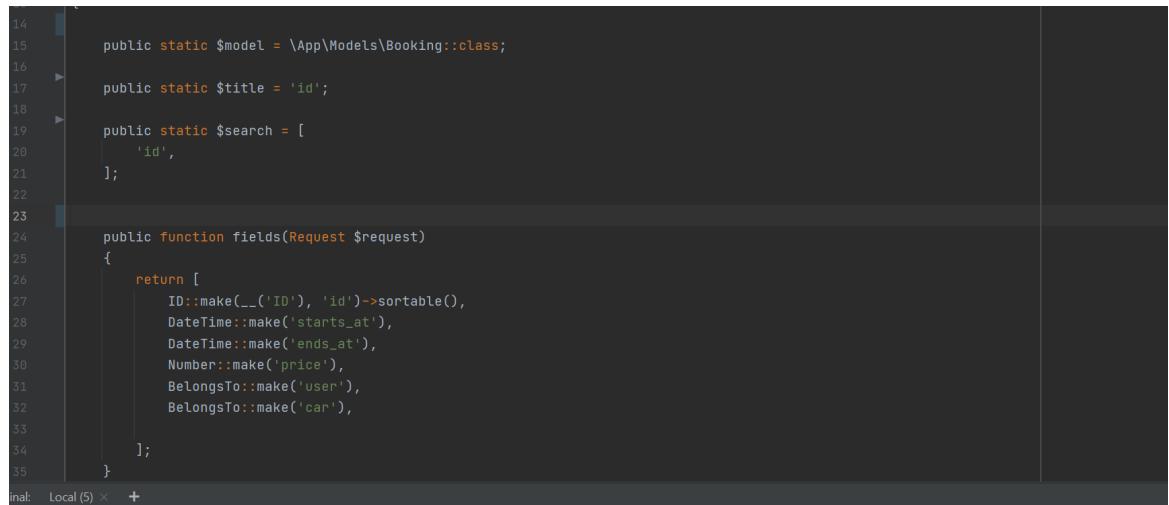
Car2Share is both an Android and IOS application that has been successfully developed and tested while also achieving its initial objectives. It also sufficiently provides its unique features to its customers in an effortless and timely manner. This project was done through the work of an integrated application such as the Dart language that adequately provides all the needs of Car2Share. We strongly hope that the targeted users of this project will benefit from this application.

References

- [1] "Saudi Arabia to implement VAT in 56 days", *Gazt.gov.sa*, 2020. [Online]. Available: https://gazt.gov.sa/en/MediaCenter/News/Pages/News_099.aspx. [Accessed: 15- Nov- 2020].
- [2] "Get a ride, drive, order food and pay with Careem", *Careem.com*, 2020. [Online]. Available: <https://www.careem.com/>. [Accessed: 15- Nov- 2020].
- [3] "About Us | Uber", *Uber*, 2020. [Online]. Available: <https://www.uber.com/sa/en/about/>. [Accessed: 15- Nov- 2020].
- [4] "Saudi economy and COVID-19: ‘Good can come of evil’", *Arab News*, 2020. [Online]. Available: <https://www.arabnews.com/node/1725751/saudi-arabia>. [Accessed: 15- Nov- 2020].
- [5] C. Valor, "Anticipated emotions and resistance to innovations: the case of p2p car sharing," *Environ. Innov. Soc. Transit.*, vol. 37, pp. 50–65, 2020.
- [6] A. Quito, "In Saudi Arabia, women can finally drive", *Quartz*, 2020. [Online]. Available: <https://qz.com/1313101/saudi-arabias-women-are-finally-allowed-to-drive-a-car-on-their-own/>. [Accessed: 15- Nov- 2020].
- [7] "KSA VAT rate to increase to 15% from 1 July 2020", *Deloitte*, 2020. [Online]. Available: <https://www2.deloitte.com/sa/en/pages/tax/articles/ksa-vat-rate-increase-15percent-1-july-2020.html>. [Accessed: 15- Nov- 2020].
- [8] S. Shaikh and S. Abro, "COMPARISON OF TRADITIONAL AND AGILE SOFTWARE DEVELOPMENT METHODOLOGY: A SHORT SURVEY", *International Journal of Software Engineering and Computer Systems*, vol. 5, no. 2, pp. 1-14, 2019. Available: [10.15282/ijsecs.5.2.2019.1.0057](https://doi.org/10.15282/ijsecs.5.2.2019.1.0057).
- [9] B. Systems, "Agile Methodologies", *Blueprintsys.com*, 2020. [Online]. Available: <https://www.blueprintsys.com/agile-development-101/agile-methodologies>. [Accessed: 15- Nov- 2020].

- [10] "Turo | The world's largest car sharing marketplace", *Turo.com*, 2020. [Online]. Available: <https://turo.com/>. [Accessed: 15- Nov- 2020].
- [11] "Rent a car or earn money sharing your car - Drive Drive Car", *Drivenetdrivecar.com*, 2020. [Online]. Available: <https://www.drivedrivecar.com/>. [Accessed: 15- Nov- 2020].
- [12] "Nabobil.no - Private car rental in your neighbourhood", *Nabobil.no*, 2020. [Online]. Available: <https://nabobil.no/en>. [Accessed: 15- Nov- 2020].
- [13] S. Alsaleh and H. Haron, "The Most Important Functional and Non-Functional Requirements of Knowledge Sharing System at Public Academic Institutions: A Case Study", *Lecture Notes on Software Engineering*, vol. 4, no. 2, pp. 157-161, 2016. Available: 10.7763/lnse.2016.v4.242.
- [14] H. Kaur and D. Sharma, "Non-Functional Requirements Research: Survey", *International Journal of Science and Engineering Applications*, vol. 3, no. 6, pp. 172-182, 2014. Available: 10.7753/ijsea0306.1003.
- [15] L. Hylving and U. Schultze, "Accomplishing the layered modular architecture in digital innovation: The case of the car's driver information module", *The Journal of Strategic Information Systems*, vol. 29, no. 3, p. 101621, 2020. Available: 10.1016/j.jsis.2020.101621.
- [16] P. real and S. 6, "Proto.io - Prototypes that feel real", *Proto.io*, 2020. [Online]. Available: <https://proto.io/>. [Accessed: 15- Nov- 2020].
- [17]"Corporate", *Tawuniya.com.sa*, 2020. [Online]. Available: <https://www.tawuniya.com.sa/en/corporate>. [Accessed: 15- Nov- 2020].
- [18] "Home Page", *Gulfunion*, 2020. [Online]. Available: <https://gulfunion.com.sa/>. [Accessed: 15- Nov- 2020].

Appendix



```
14     public static $model = \App\Models\Booking::class;
15
16     public static $title = 'id';
17
18     public static $search = [
19         'id',
20     ];
21
22
23
24     public function fields(Request $request)
25     {
26         return [
27             ID::make(__('ID'), 'id')->sortable(),
28             DateTime::make('starts_at'),
29             DateTime::make('ends_at'),
30             Number::make('price'),
31             BelongsTo::make('user'),
32             BelongsTo::make('car'),
33         ];
34     }
35 }
```

Figure 58 - Booking code



```
7     use Laravel\Nova\Fields\ID;
8     use Laravel\Nova\Fields\Text;
9     use Laravel\Nova\Http\Requests\NovaRequest;
10
11    class Location extends Resource
12    {
13
14        public static $model = \App\Models\Location::class;
15
16
17        public static $title = 'name';
18
19
20        public static $search = [
21            'name',
22        ];
23
24        public function fields(Request $request)
25        {
26            return [
27                ID::make(__('ID'), 'id')->sortable(),
28                Text::make('Name'),
29                HasMany::make('cars')
30            ];
31        }
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
288
289
289
290
291
292
293
294
295
296
297
298
299
299
300
301
302
303
304
305
306
307
308
309
309
310
311
312
313
314
315
316
317
318
319
319
320
321
322
323
324
325
326
327
328
329
329
330
331
332
333
334
335
336
337
338
339
339
340
341
342
343
344
345
346
347
348
349
349
350
351
352
353
354
355
356
357
358
359
359
360
361
362
363
364
365
366
367
368
369
369
370
371
372
373
374
375
376
377
378
379
379
380
381
382
383
384
385
386
387
388
389
389
390
391
392
393
394
395
396
397
398
399
399
400
401
402
403
404
405
406
407
408
409
409
410
411
412
413
414
415
416
417
418
419
419
420
421
422
423
424
425
426
427
428
429
429
430
431
432
433
434
435
436
437
438
439
439
440
441
442
443
444
445
446
447
448
449
449
450
451
452
453
454
455
456
457
458
459
459
460
461
462
463
464
465
466
467
468
469
469
470
471
472
473
474
475
476
477
478
479
479
480
481
482
483
484
485
486
487
487
488
489
489
490
491
492
493
494
495
496
497
497
498
499
499
500
501
502
503
504
505
506
507
508
509
509
510
511
512
513
514
515
516
517
518
519
519
520
521
522
523
524
525
526
527
528
529
529
530
531
532
533
534
535
536
537
538
539
539
540
541
542
543
544
545
546
547
548
549
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
688
689
689
690
691
692
693
694
695
696
697
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
778
779
779
780
781
782
783
784
785
786
787
788
789
789
790
791
792
793
794
795
796
797
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
878
879
879
880
881
882
883
884
885
886
887
888
888
889
889
890
891
892
893
894
895
896
897
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
918
918
919
920
921
922
923
924
925
926
927
928
928
929
929
930
931
932
933
934
935
936
937
938
938
939
940
941
942
943
944
945
946
947
948
948
949
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
978
979
979
980
981
982
983
984
985
986
987
987
988
989
989
990
991
992
993
994
995
996
997
998
999
999
1000
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1016
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1047
1047
1048
1049
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1067
1068
1068
1069
1069
1070
1071
1072
1073
1074
1075
1076
1077
1077
1078
1078
1079
1079
1080
1081
1082
1083
1084
1085
1086
1087
1087
1088
1088
1089
1089
1090
1091
1092
1093
1094
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1111
1112
1113
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1121
1122
1123
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1131
1132
1133
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1141
1142
1143
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1151
1152
1153
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1161
1162
1163
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1171
1172
1173
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1181
1182
1183
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1191
1192
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1201
1202
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1211
1212
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1221
1222
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1231
1232
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1241
1242
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1251
1252
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1261
1262
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1271
1272
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1281
1282
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1291
1292
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1301
1302
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1311
1312
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1321
1322
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1331
1332
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1341
1342
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1351
1352
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1361
1362
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1371
1372
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1381
1382
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1391
1392
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1401
1402
1403
1404
1404
1405
1405
1406
1406
1407
1407
1408
1408
1409
1409
1410
1411
1412
1413
1414
1414
1415
1415
1416
1416
1417
1417
1418
1418
1419
1419
1420
1421
1422
1423
1424
1424
1425
1425
1426
1426
1427
1427
1428
1428
1429
1429
1430
1431
1432
1433
1434
1434
1435
1435
1436
1436
1437
1437
1438
1438
1439
1439
1440
1441
1442
1443
1444
1444
1445
1445
1446
1446
1447
1447
1448
1448
1449
1449
1450
1451
1452
1453
1454
1454
1455
1455
1456
1456
1457
1457
1458
1458
1459
1459
1460
1461
1462
1463
1464
1464
1465
1465
1466
1466
1467
1467
1468
1468
1469
1469
1470
1471
1472
1473
1474
1474
1475
1475
1476
1476
1477
1477
1478
1478
1479
1479
1480
1481
1482
1483
1484
1484
1485
1485
1486
1486
1487
1487
1488
1488
1489
1489
1490
1491
1492
1493
1494
1494
1495
1495
1496
1496
1497
1497
1498
1498
1499
1499
1500
1501
1502
1503
1504
1504
1505
1505
1506
1506
1507
1507
1508
1508
1509
1509
1510
1511
1512
1513
1514
1514
1515
1515
1516
1516
1517
1517
1518
1518
1519
1519
1520
1521
1522
1523
1524
1524
1525
1525
1526
1526
1527
1527
1528
1528
1529
1529
1530
1531
1532
1533
1534
1534
1535
1535
1536
1536
1537
1537
1538
1538
1539
1539
1540
1541
1542
1543
1544
1544
1545
1545
1546
1546
1547
1547
1548
1548
1549
1549
1550
1551
1552
1553
1554
1554
1555
1555
1556
1556
1557
1557
1558
1558
1559
1559
1560
1561
1562
1563
1564
1564
1565
1565
1566
1566
1567
1567
1568
1568
1569
1569
1570
1571
1572
1573
1574
1574
1575
1575
1576
1576
1577
1577
1578
1578
1579
1579
1580
1581
1582
1583
1584
1584
1585
1585
1586
1586
1587
1587
1588
1588
1589
1589
1590
1591
1592
1593
1594
1594
1595
1595
1596
1596
1597
1597
1598
1598
1599
1599
1600
1601
1602
1603
1604
1604
1605
1605
1606
1606
1607
1607
1608
1608
1609
1609
1610
1611
1612
1613
1614
1614
1615
1615
1616
1616
1617
1617
1618
1618
1619
1619
1620
1621
1622
1623
1624
1624
1625
1625
1626
1626
1627
1627
1628
1628
1629
1629
1630
1631
1632
1633
1634
1634
1635
1635
1636
1636
1637
1637
1638
1638
1639
1639
1640
1641
1642
1643
1644
1644
1645
1645
1646
1646
1647
1647
1648
1648
1649
1649
1650
1651
1652
1653
1654
1654
1655
1655
1656
1656
1657
1657
1658
1658
1659
1659
1660
1661
1662
1663
1664
1664
1665
1665
1666
1666
1667
1667
1668
1668
1669
1669
1670
1671
1672
1673
1674
1674
1675
1675
1676
1676
1677
1677
1678
1678
1679
1679
1680
1681
1682
1683
1684
1684
1685
1685
1686
1686
1687
1687
1688
1688
1689
1689
1690
1691
1692
1693
1694
1694
1695
1695
1696
1696
1697
1697
1698
1698
1699
1699
1700
1701
1702
1703
1704
1704
1705
1705
1706
1706
1707
1707
1708
1708
1709
1709
1710
1711
1712
1713
1714
1714
1715
1715
1716
1716
1717
1717
1718
1718
1719
1719
1720
1721
1722
1723
1724
1724
1725
1725
1726
1726
1727
1727
1728
1728
1729
1729
1730
1731
1732
1733
1734
1734
1735
1735
1736
1736
1737
1737
1738
1738
1739
1739
1740
1741
1742
1743
1744
1744
1745
1745
1746
1746
1747
1747
1748
1748
1749
1749
1750
1751
1752
1753
1754
1754
1755
1755
1756
1756
1757
1757
1758
1758
1759
1759
1760
1761
1762
1763
1764
1764
1765
1765
1766
1766
1767
1767
1768
1768
1769
1769
1770
1771
1772
1773
1774
1774
1775
1775
1776
1776
1777
1777
1778
1778
1779
1779
1780
1781
1782
1783
1784
1784
1785
1785
1786
1786
1787
1787
1788
1788
1789
1789
1790
1791
1792
1793
1794
1794
1795
1795
1796
1796
1797
1797
1798
1798
1799
1799
1800
1801
1802
1803
1804
1804
1805
1805
1806
1806
1807
1807
1808
1808
1809
1809
1810
1811
1812
1813
1814
1814
1815
1815
1816
1816
1817
1817
1818
1818
1819
1819
1820
1821
1822
1823
1824
1824
1825
1825
1826
1826
1827
1827
1828
1828
1829
1829
1830
1831
1832
1833
1834
1834
1835
1835
1836
1836
1837
1837
1838
1838
1839

```

```
19  
20 class CarModel extends Resource  
21 {  
22     public static $model = \App\Models\CarModel::class;  
23  
24     public static $title = 'title';  
25     public static $search = [  
26         'id',  
27     ];  
28     public static $group = 'Car Management';  
29  
30     public function fields(Request $request)  
31     {  
32         return [  
33             ID::make()->sortable(),  
34  
35             Text::make('Title')  
36                 ->rules('required', 'string', 'max:100'),  
37  
38             BelongsTo::make('CarType', 'carType', CarType::class),  
39             BelongsTo::make('CarCompany', 'carCompany', CarCompany::class) ...  
40         ];  
41     }  
42 }  
43
```

Figure 60 - Car database code

```
15 class Thread extends Resource  
16 {  
17  
18     public static $model = \Cmgmyr\Messenger\Models\Thread::class;  
19  
20     public static $title = 'subject';  
21  
22     public static $search = [  
23         'subject',  
24     ];  
25     public function fields(Request $request)  
26     {  
27         return [  
28             ID::make(__('ID'), 'id')->sortable(),  
29             Text::make('Subject'),  
30             BelongsToMany::make('Users'),  
31             HasMany::make('Messages')  
32         ];  
33     }  
34 }  
35
```

Figure 61 - Thread code

Figure 62 - Attributes database

```

children: [
  const SizedBox(height: 30),
  Padding(
    padding: const EdgeInsets.all(8.0),
    child: FormBuilderTextField(
      name: 'email',
      initialValue: 'admin@carshare.com',
      decoration: InputDecoration(
        labelText: 'Email',
        errorText: _emailError,
      ),
      // InputDecoration
    ), // FormBuilderTextField, Padding, ListView
  ), // FormBuilder
], // Container
padding: const EdgeInsets.all(8.0),

child: FormBuilderTextField(
  name: 'password',
  initialValue: '12345678',
  decoration: InputDecoration(labelText: 'Password'),
  obscureText: true,
  validator: FormBuilderValidators.compose([
    FormBuilderValidators.required(context),
    FormBuilderValidators.minLength(context, 8),
  ]),
);

```

Figure 63 - Login Page Code