**CAR WASH STATION**

**BY**

**STUDENT NAME I: ANAS AHMED ID: 362049763**

**STUDENY NAME II: RASLAN RAZOQ ID: 362051303**

**STUDENT NAME III: MOHMMED JAMAL ID: 362049583**

**Supervised by: Dr-YAZED ALSAAWY**



**Graduate project 1 (midterm report)**

**Faculty of Computer and Information Systems**

**Islamic University of Madinah**

**10/2020**

**Table of content**

Contents

[**Abstract** 5](#_Toc58534773)

[**1.0-Introduction:** 6](#_Toc58534774)

[**1.1-Aim:** 7](#_Toc58534775)

[**1.2-Overview:** 7](#_Toc58534776)

[**1.3-Problem statement:** 7](#_Toc58534777)

[**1.4-Objectives:** 8](#_Toc58534778)

[**1.5-Gantt chart:** 9](#_Toc58534779)

[**1.6-Framework:** 9](#_Toc58534780)

[**2.0-Focus in the project:** 10](#_Toc58534781)

[**2.1-Overview of technology:** 10](#_Toc58534782)

[**2.2-Literature review:** 10](#_Toc58534783)

[**2.2.1-Systems and method for car wash providers:** 10](#_Toc58534784)

[**2.2.1.1-Ordinary carwash service procedure :** 10](#_Toc58534785)

[**2.2.1.2-Modern on-site car wash mechanism approach:** 11](#_Toc58534786)

[**2.2.2-Mobile application and the added value to people life:** 11](#_Toc58534787)

[**2.2.3-What is the advantage of mobile application in the car wash industry field?** 12](#_Toc58534788)

[**2.2.4-Number of Saudi family's cars per household :** 12](#_Toc58534789)

[**2.2.5-Pricing car wash services:** 13](#_Toc58534790)

[**2.2.6-Car wash appointment absent service:** 13](#_Toc58534791)

[**2.3-The technology:** 13](#_Toc58534792)

[**2.3.0- Flutter:** 13](#_Toc58534793)

[**2.3.1- Laravel PHP framework:** 14](#_Toc58534794)

[**2.3.2- Database:** 14](#_Toc58534795)

[**2.4-Past experience and similar applications:** 14](#_Toc58534796)

[**2.4.0 Ghaseel app:** 14](#_Toc58534797)

[**2.4.1 Sayar app:** 15](#_Toc58534798)

[**2.5-Summary:** 15](#_Toc58534799)

[**3.0- Requirement** 18](#_Toc58534800)

[**3.1- Designing** 19](#_Toc58534801)

[**3.2-Functional Requirement:** 19](#_Toc58534802)

[**3.3-Non-Functional Requirement:** 20](#_Toc58534803)

[**3.4-System documentation:** 21](#_Toc58534804)

[**3.5-UML diagrams:** 21](#_Toc58534805)

[**3.5.1 Use case:** 21](#_Toc58534806)

[**3.5.1.4 Use case Diagram:** 22](#_Toc58534807)

[**1- Customer:** 23](#_Toc58534808)

[**2-Car wash station:** 24](#_Toc58534809)

[**3-Mobile car wash:** 24](#_Toc58534810)

[**4-Admin:** 24](#_Toc58534811)

[**3.5.2 Class diagram:** 25](#_Toc58534812)

[**3.5.2.1 Class diagram (Diagram)** 25](#_Toc58534813)

[**3.5.3 Sequence diagram:** 26](#_Toc58534814)

[**3.5.3.1 Sequence Diagrams** 26](#_Toc58534815)

[**3.6 Entity relationships** 27](#_Toc58534816)

[**3.7 Database tables** 27](#_Toc58534817)

[**Future work :** 29](#_Toc58534818)

[**4.0-Flutter mobile application** 30](#_Toc58534819)

[**4.1** **Laravel application models** 43](#_Toc58534820)

[**4.2** **Laravel application controller** 48](#_Toc58534821)

[**References:** 61](#_Toc58534822)

**Table of figures**

|  |  |
| --- | --- |
| **Page No** | **Chapter 1** |
| 10 | Figure (1) Gantt chart |
| 11 | Figure (2) Framework |
|  | **Chapter 2** |
| 22 | Figure (3) Use case |
|  | **Chapter 3** |
| 24 | Figure (4) Class diagram. |
| 25 | Figure(5) Sequence application. |
| 26 | Figure(10) Entity of actors. |

# **Abstract**

Nowadays, mobile and web applications are the best way to reach customers. Using applications to perform daily tasks is now prevalent. So, the developers always try to present the best app to make your life more comfortable. So, we thought to provide an application that would make it easier for customers to request a car wash service or book an appointment before going to the car wash station or provider, evaluate the service provided, and many advantages. We will use a flutter framework to program our application to ensure access to the largest segment of customers, and the control panel will be built using the Laravel framework. We first start with identify our aim, subject area, and argument, doing some literary analysis is the second step to read another research, journal, and conference to understand the area we are looking for after this identify the objective and the method must be the third step in the software system getting requirement, design the system, implement some code, testing, and training is the important thing for the objective, then we find a method for each objective.

**CAR WASH STATION**

**Chapter 1**

# **1.0-Introduction:**

In today’s era, utilize and engagement of smartphones and mobile applications is not considered a new thing. There are Serval categories and different classifications of mobile applications, users can make use of, i.e., references, utilities, fitness, calendar, games, news, and others. Based on their smartphone operating systems, users may download any application that matches their needs either via the App Store or Play Store. It is no deniable that the use of mobile applications assists users. One of the user's needs is bringing life back to their cars with services like a car wash.

Car wash service providers have regulation approaches used for cleaning the exteriors and interiors of the cars. It assists the procedure of washing and cleaning all types of vehicles. The increase in the automotive industry produces an increasing requirement for car wash service providers. Different aspects are leading the expansion of the car wash service providers market are the growth in manufacturing and selling cars, reduced home local car wash, and increase revenue for car wash investors. Also, the growth of the market is leading by many attributes like higher safety, time efficiency, improved efficiency, soft administration, reliability, and consumer contentment.

Car washing systems are very common in developed countries and industrialized countries. Costumers are attracted to the awareness of the advantages of car wash services such as improved efficiency and safety. Furthermore, the growing environmental awareness between humans and citizens is leading the growth of car wash services, which guarantee the good deployment of water resources and appropriate throwing of hard tush, thus assisting the growth of the market for car wash systems.

There are many car washes in Saudi Arabia and the world as it is considered one of the successful small projects if it has strict management and excellent work. The types of car wash differ in terms of traditional, automatic, and mobile. Besides, mobile washing cars demand is growing day by day. From here came the idea of ​​the system and the mobile application where I can limit the collection of most types of the existing car washers in one place so that he can a customer choose from and choose the one appropriate for him.

# **1.1-Aim:**

An application that will allow the user (customer) to request service, to book the car washing service at a certain time of the day (along with other washing options). The car wash station will provide the real-time availability of the service and give the customer update on when he can reserve, and the nature of the service needed, and the mode of payment.

# **1.2-Overview:**

The overall outlook over the system of the car wash has two main parts, the first part is for booking car wash services online, and the second one is for the providers and managers of the car wash services to control the business, review workers, and provide full reports for all processes. In the current system, it is an old school system, without any management technics so costumer now dealing with varying issues when cleaning their cars with the normal car wash, example an issue may customer face such as time-consuming.

# **1.3-Problem statement:**

1. The crowding and the time-wasting at car wash line
2. To know the specific sold time for every car wash
3. Know types of specific services
4. Review the quality of worker
5. Finding the nearest car wash to customers and evaluating the quality

# **1.4-Objectives:**

The main objective of the car wash system is making car wash service full automated form the customer reservation to paying and reviewing the service

Sub-objectives are:

1. Knowing and reviewing the evaluation of customers in the car wash application in terms of services provided and offers "feedback".
2. Provide a home car cleaning service.
3. Prepare and provide a car wash for the type of service required and chosen by the customer
4. provide several payment methods for the user, and the ability to pay remotely.
5. Locate the location of car washes.
6. Reduce the annoying crowd at car washes by knowledge of the car wash status and Specify the period for cleaning the car process.
7. Daily inventory for car wash productivity.
8. Provide an opportunity for freelancer services for on-site car wash service.

# **1.5-Gantt chart:**

Figure (1) Gantt chart

# **1.6-Framework:**

Figure (2) Framework

**Chapter 2**

# **2.0-Focus in the project:**

We need to concentrate on part of the social issues that waste time in them, we have found that the registration and reservation mechanism for appointments is the mechanism that needs to be improved and converted to the online system, and that is what we are going to improve in the original system.

# **2.1-Overview of technology:**

In general, society has turned into being based on technology in many aspects, and mobile and web applications are the best way to reach customers since this situation become part of our way of life there is a high number of the population have a smartphone/tablet using applications to perform daily tasks is now prevalent that provide to accessibility to the internet. The car wash domain is far away from web and applications technologies at this moment, but it will be a big change and a real challenge.

# **2.2-Literature review:**

## **2.2.1-Systems and method for car wash providers:**

## **2.2.1.1-Ordinary carwash service procedure :**

The traditional approach of car wash usually obligatory the individual consumer to drive his car to a certain car wash site. Moreover, Traditional car wash services may use fully automatic or partially automated car wash unit, or manually washing utilize hand car wash with car wash staff, purgation, and airing every car, besides that the quality of an automated car wash is never as good as a hand wash[1]. These traditional approaches of car wash led to obvious disadvantages, like obligatory a consumer went to a specific car wash site and obligatory the consumer to physically wait in line until other cars in front of the consumer's car are finishing. Accordingly, busy people frequently retreat to delaying the car wash, or as an alternative wash their cars at the house in their free time.[2]

## **2.2.1.2-Modern on-site car wash mechanism approach:**

From the car wash service and the industry's perspective, these disadvantages linked with the traditional approaches of car wash are stray business chances and lost revenue[3]. As a result, in particular markets, a mobile on-site car wash that could make use of a van nor track to carry car wash tools, supplies, and car wash technicians to a consumer's distinct location, such as the consumer's place of business car-park or at the consumer's home, as a business sector, they have demonstrated their ability to profit from the car wash service industry. An additional advantage or utility of the mobile on-site car wash services is a make efficient awareness of specific details from a consumer's view and creating a higher profit limit from a service supplier's or provider's perspective. Because a consumer ordering mobile on-site car wash is interested to pay prerequisite for the good and satisfaction of having the car wash completed at a chosen place or site, this frequently common to order value-added services for that kind of on-site car wash service consumer, like internal car wash out, specific detailing [2].

## **2.2.2-Mobile application and the added value to people life:**

In past years, society has transformed profoundly, repercussion of the development of Information and Communication Technologies (ICT), which potentiated the evolution of a wide range of new services and solutions that facilitate interaction globally [4]. Smartphones and mobile applications participate in this growth and have become part of human daily life, guiding the fast expansion in the development of software implementations for this level of technologies. A considered part of purchases that occupied a critical place since 2010 and many of users of these smartphone devices do not visit conventional websites in their everyday needs[5]. The presentation of smartphones and mobile applications has significantly affected many domains including the delivery and reservation services industry, given its portability, mobility, and autonomy features [6].

The continuous development of Information Technology, Communications, and Electronics contributed decisively to the paradigm shift, which led to accommodate the customer in the decision-making circle, so the economic activities of local dimension such as, delivery and reservation services are challenging in present and tend to be preeminent shortly, by providing competitive offers and present their utilities or services to the consumer.[7]

Technology improvement has led to the growth of this mobile application to a new dimension. The mobile application is referred to as an application which is also known as a type of application software to run on a mobile device, i.e., smartphone or tablet[8,9].

Smartphones and mobile application technologies are one of the rapid expansions and most used internationally [12]. Due to the increasing popularity of mobile applications, the car washing industry did not obtain a proportion in this field which supports the success of our application.

Humans utilize new technology and devices to achieve their daily needs. One of these needs is to purchase online by requesting online product orders. Individuals positioning online orders while they are in-home. Nevertheless, about 33% of consumers are away from their houses [13]. This promotes the method of on-site car wash service in our system and application.

## **2.2.3-What is the advantage of mobile application in the car wash industry field?**

The world looking ahead to smartphones and mobile applications these days as a trend. Do not underestimate the importance and abilities of smartphones and mobile application software, it's able to provide their convenience features, ease of use, simpler interface, and much more to use. With such, mobile applications can create engagement and allegiance between their customers.[12]

What is required for perfection is a technology and a properly functioning mechanism that can specify and improve human resources of different industry sectors in agreement with the demands and changes in those fields. The unstoppable appeal of mobile application systems can be used as a human resources technology, this will consider as a significant move that will facilitate human resources to develop manpower productivity in Saudi Arabia and many countries. Having this technology work in real-time on employees' mobile devices will enhance interaction productivity and retention. Reliance on mobile applications can make engaged employees feel more connected and increase their likelihood of continuing by 37%[14]. The improvement in work and employees' output that our system and application will provide to car wash service providers will one of our success factors.

## **2.2.4-Number of Saudi family's cars per household :**

Local automobile consumption has progressive growth, motor car ownership increases year by year. In a 2016 survey by General Authority for Statistics in Saudi Arabia, 92% of Saudi households have at least one car in their daily use where they are living and this number capable to increase in the next years [15]. The increase in the car proprietorship creates tremendous opportunities for the car wash industry. The car wash industry has the specificity of constant service centers and restricted service potentiality [3]. All these factors will assist our system and application and ensure expansion among a big sample of users and customers.

## **2.2.5-Pricing car wash services:**

The service time cost is the main factor influencing the choice of service pricing strategy under the fixed service capacity situation. The subscription pricing strategy is indeed preferable to the per-use pricing strategy, although companies cannot effectively control the wait time for customer service when adopting this pricing mechanism. Service capacity cost and service time cost jointly affect the choice of an organization's service pricing strategy when service capacity can be modified. For a subscription pricing strategy, the absolute advantage of a subscription pricing strategy for every use is very small, while the absolute benefits of a subscription pricing strategy for a per-user pricing strategy are evident[7]. Depending on this study, we will provide two approaches for pricing, the car wash providers will choose the most appropriate for them.

## **2.2.6-Car wash appointment absent service:**

In the car wash, there is one shift for the working time and it has 12 hours so it will take 11 hours to wash the car and one hour will be the emergency time where the customer will give 5 minutes to confirm the attending if the customer did not confirm, message sents to the next customer. We will bear in mind that there will be an absence in some car wash service appointment, the solution was created for this service, which revolves around the customer being late for more than 10 minutes from the specified time, the customer next appointment will be modified and he will be informed of this via app notification or SMS. If we consider that most customers are absent from their appointment and the next customer will also have 5 minutes to affirm, so these five minutes will be observed at the time of emergency.

# **2.3-The technology:**

## **2.3.0- Flutter:**

Flutter is a free and open-source mobile user interface framework created by Google and released in May 2017. In short, it lets you create an original mobile app with just one codebase. This means that you can use one programming language and one codebase to create two different applications[16], which we use because it will make the mobile programming phase faster in the beginning and have a good community that will give us more future in the future.

## **2.3.1- Laravel PHP framework:**

Laravel is a free and open-source PHP web framework, created by Taylor Orwell and aims to develop web applications according to the Model-View-Controlled (MVC) Architectural Model (MVC) based on Symfony. Some of the features of Laravel are a modular bundling system with a dedicated dependency manager, various ways to access relational databases, utilities that help in deploying and maintaining applications[17], and we use Laravel because it is a high level of security and has many packages that will improve our work in the background with APIs available for free.

## **2.3.2- Database:**

MySQL server is a SQL compliant server, and it has a major advantage that it’s free, and we use it because it has high data security, have a high performance, On-demand scalability, and the flexibility of open source, also MySQL is based on a Client-Server model.

# **2.4-Past experience and similar applications:**

## **2.4.0 Ghaseel app:**

it is an application that allows car wash service providers to display their services in the application and allows customers to review and request laundry service for the place specified by the customer[18]. How Ghaseel works:

1. The customer specifies the location of the car to be washed.
2. Determine the type of vehicle
3. Browse the available laundries to request the service
4. paying off
5. Rating



services plans provided by Ghaseel app[18]

## **2.4.1 Sayar app:**

It is an application that connects mobile service providers with customers in one place and allows customers to choose various car care services at different prices to implement them at the time and place that suits them. .[19]

How Sayar works:

1. The customer specifies the place and time for the service.
2. Choose the service and package provider.
3. paying off
4. Rating

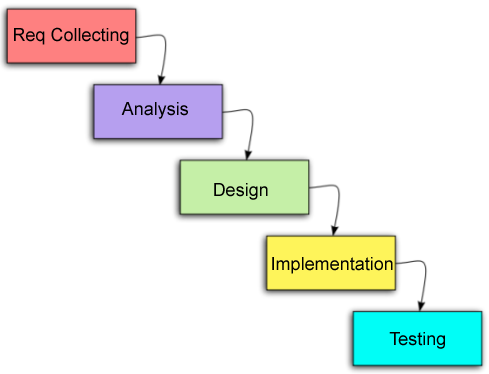
Both above applications are for mobile car wash service provider companies only, so this will give our application more advantage in locating the nearest car wash service provider or station.

# **2.5-Summary:**

The facility of getting car wash services from home is becoming more and more reality in today’s world. Besides, Saudi Arabia weather is covered with dust a lot, so washing cars is a matter of concern to all car owners. In this project, we suggested an app to solve the hardness of getting a car wash service from home or at a car wash. This app aims to assist people to make their cars clean in one minute. Based on all these factors and data provided that will support the success of the application from the community’s reliance on mobile applications to meet their daily needs and the market’s need for change and to keep pace with this technical movement, the car wash field is in dire need of change and this is what our app will do.

**Functionality analysis**

**Chapter 3**



***Requirement Collecting:***

Traditional car wash methods are time consuming and inconvenient for most costumers, furthermore, there is not any approach to review the car wash services, besides the car wash service providers are interested in making more revenue, so we determine to gather the requirement from the costumers and service providers to see their needs, the improvements they are looking forward, and their issues. we take all these requirements from a survey among customers and interviewing the car wash owners and providers.

***Analysis and Design:***

It’s a mobile application for consumers and a control panel for car wash service providers so after getting the requirement, we will start design the interfaces for the mobile application that we will implement also, perhaps we will do it in the papers, interface, algorithm, and pseudo-code the entire system parts, mobile application and control panel. This will show how system parts will interact with each other, such as data flows in the system using a data flow diagram for example.

***Implementation:***

After concluding gathering the requirements and mapping the interfaces, phase, then the implementation phase is started to be work on it. The implementation will be concentrating on design the mobile application interfaces and the control panel interfaces, then connect the database to these interfaces, then coding the API that will connect the mobile application to the control panel.

***Testing:***

The testing phase must take place; it has been standardly made to experiment to produce accurate results and make it more efficient in terms of discovering errors and making more beneficially to the users or providers contentment. Testing the agenda is different from testing techniques such as unit tests, integration tests, and user tests.

***The finish on time:***

We are planning to do our best and try to complete the project on the date that the faculty define, but of course, we will cover the main things that make the project deliver the best results and solutions to our problem statement.

***Program without bugs:***

We are sure that our mobile application and control panel will give the best performance with the least errors, at the final step we will make the entire system is tested to confirm the system will be easy to use for all stockholders.

# **3.0- Requirement**

The condition that we took about car wash service providers before personal sessions and discussion and take some suggestions from car wash owners to improve car wash service and know the best way to improve and make them easy and convenient, and we also made some visits to car wash laundries to learn how to make them more efficient.

# **3.1- Designing**

First, we did some storyboard about the interfaces, draw some series on papers and for each one writing notes include a description of what happening after we finish it discuss it together and implement it in our IDE, the last step publishes it to application stores and web.

# **3.2-Functional Requirement:**

**Login ID**:

Any user who uses the application shall have a Login ID and Password.

**Appointment Booking:**

The app will allow the customer to book the most appropriate time and date to wash the car.

**Appointment availability:**

The application should allow the user if there is any available appointment to wash the car and the name of the car wash laundries.

**The information about car wash:**

The application should allow the user to view the car wash information and services.

**Service type:**

Request a car wash service, whether in a car wash place or another place (home or workplace).

**Payment method:**

The application should allow providing several payment methods for users.

**The user review:**

The application should allow the user to write a review and report on the service provided and the available offers.

**Independent work:**

The application should allow to enabling laundry workers to participate in free or independent work.

# **3.3-Non-Functional Requirement:**

**User Identification:**

The application requires the user to identify themselves and register with the application.

**Modification:**

Any modification (insert, delete, and update) for the Database shall be synchronized and done only by the user in the ward.

**Rights of car wash employees**:

Car wash personnel must be able to view all information but will not be able to modify any information in it.

**Administrators' Rights**:

Administrators must be able to view and amend all information and add new services.

**Response Time**:

The system shall give responses in 1 second after checking the user information and the information of car wash laundries.

**User-interface**:

The user-interface screen shall respond within 5 seconds.

**Back-Up**:

The system shall provide the capability to back-up the Data.

**Errors**:

The system shall keep a log of all the errors.

**Availability:**

The system shall be available all the time.

# **3.4-System documentation:**

User ( service provider, customers, and admin ) will have short documentation about how the system functioning and it will contain another part such as a brief description, advantage, and future work about the system and application and all that will to make the system and application easy to use for the user.

# **3.5-UML diagrams:**

## **3.5.1 Use case:**

The purpose of using a use case diagram is to summarize the actions between the system’s users (actors), and the system they use. The UML Use case diagram is appropriate for:

1. Display the objectives between user and system interactions
2. Organizing and defining the functional requirements in a system
3. Determining the requirement for the system.

**There are three main components of the use case diagram:**

1. **3.5.1.1Actors:**

The users who interact with the system, he is can be a person, an organization, or another system they can make interact with your system or application. This external object produces or receives data.

1. **3.5.1.2 System:**

It is a sequence of actions and interactions between the actors and the system they use. The system can be referred to as a scenario.

1. **3.5.1.3 Goals:**

Goals representing the result of the use case. The diagram should show the activities to reach the goals.

## **3.5.1.4 Use case Diagram:**

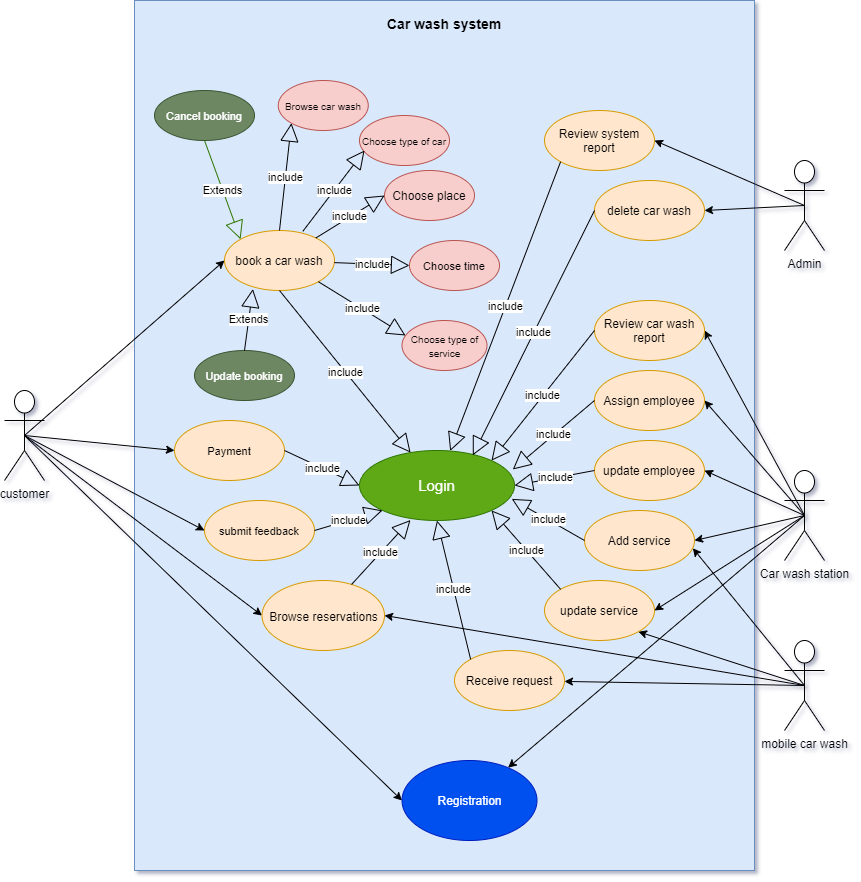


Figure (3) Use case

### **1- Customer:**

First of all, when the customer opens the order, he will create an account, which includes the information that the application will request to create the account, and the required information is: name, identifier, address, phone number, in the other case of the car wash and car wash employees who already had their information in the database that was previously registered, the schedule will open to choose an appointment, and he can also amend or cancel the appointment and show their information.

### **2-Car wash station:**

The server will transfer the client's information to the database after successful registration, and the server will display the available appointments on weekdays. The client will choose the available day and the appropriate time, and then the server will send a confirmation message via the application, then the server will transfer the chosen appointment to the database. In unusual cases, the appointment may be delayed, after which the server will send the new appointment through the application to the client.

### **3-Mobile car wash:**

The mobile car wash service provider will receive the requests and orders for on-site type of services, accept it, give the approval, and visit the customer location to start the car wash service.

### **4-Admin:**

The admin is responsible for the accept new car wash provider and browse the system reports, if there is any violate of rules admin has permissions to decide what to do and delete violators.

# **3.5.2 Class diagram:**

It is a part of UML diagram that modeling its classes, attributes, operations, and relationships between objects.

# **3.5.2.1 Class diagram (Diagram)**

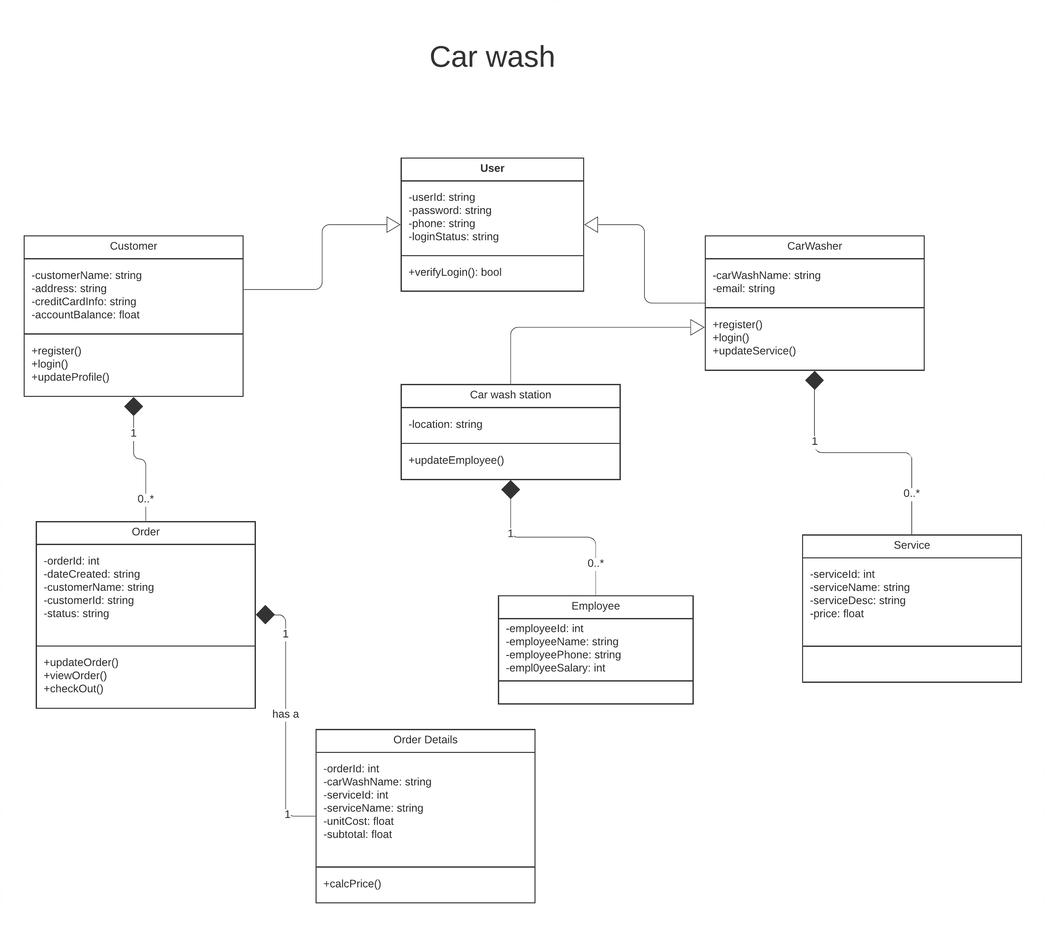


Figure (4) Class diagram.

# **3.5.3 Sequence diagram:**

Sequence diagram describe the relationship between classes of exchanging information between them. In addition, it has called event diagram. Sequence diagram is suitable for visualizing various scenarios.

# **3.5.3.1 Sequence Diagrams**

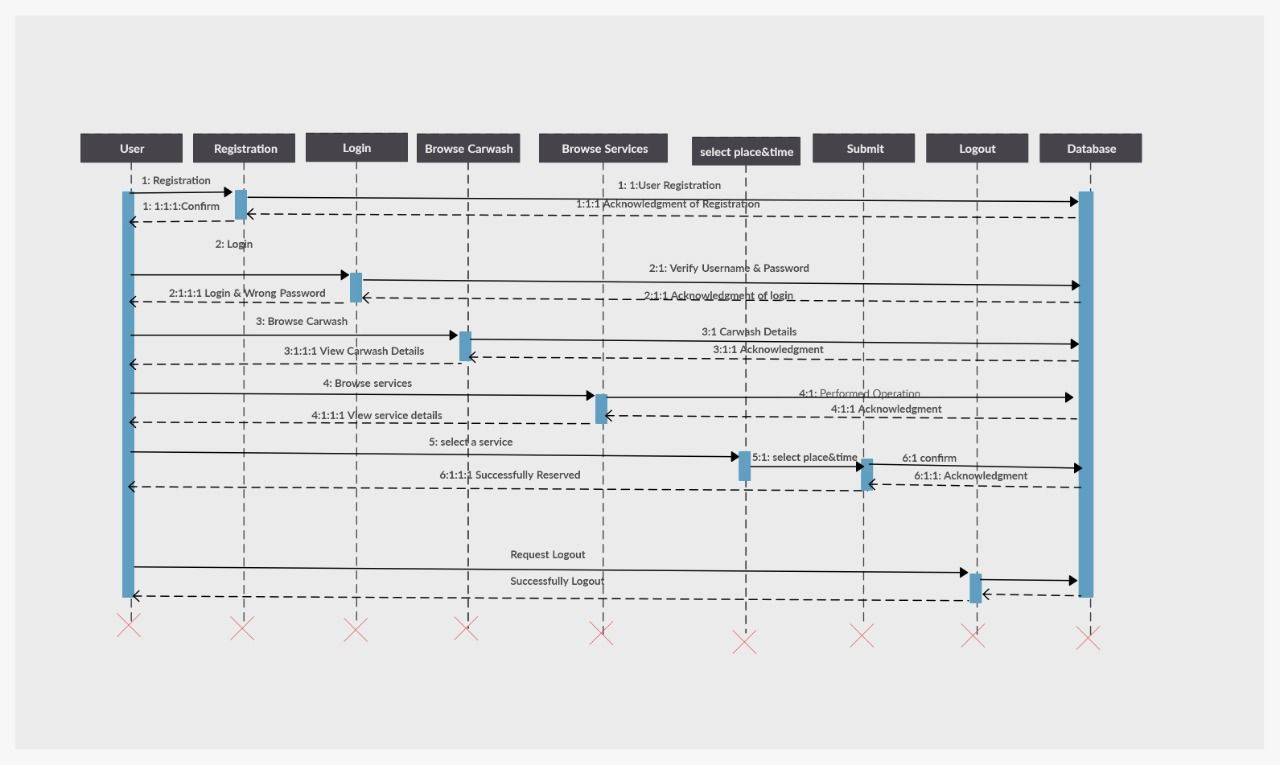


Figure (5) Sequence for application.

# **3.6 Entity relationships**

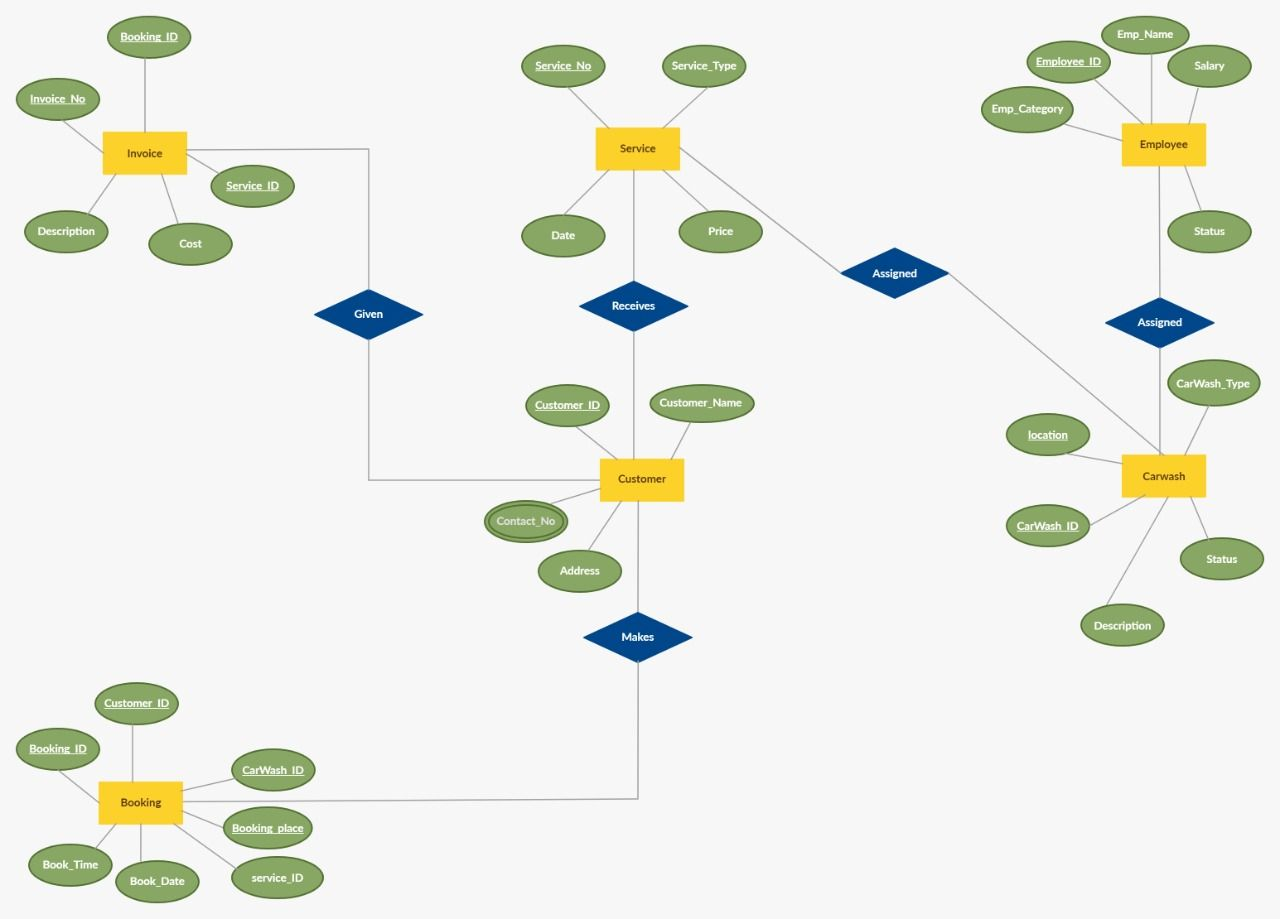


Figure (10) Entity of actors

# **3.7 Database tables**

Customer

|  |  |  |
| --- | --- | --- |
| Note | Type | Field name |
|  | String (varchar) | Customer\_Name |
|  | Integer | Customer\_ID |
|  | Integer | Contact\_No |
|  | String (varchar) | Address |

Car wash

|  |  |  |
| --- | --- | --- |
| Note | Type | Field name |
| Service\_No | String (char) | CarWash\_Type |
|  | Integer | CarWash\_ID |
|  | Integer | Status |
|  | String (varchar) | Location |
|  | String (varchar) | Description |

Employee:

|  |  |  |
| --- | --- | --- |
| Note | Type | Field name |
|  | String (varchar) | Emp\_Name |
|  | Integer | Employee\_ID |
| CarWash\_ID | Integer | Status |
|  | String (char) | Emp\_Category |
|  | Float | Salary |

Invoice:

|  |  |  |
| --- | --- | --- |
| Note | Type | Field name |
|  | Integer | Booking\_ID |
|  | Integer | Invoice\_No |
|  | Integer | service\_ID |
|  | Float | Cost |
|  | String (varchar) | Description |

Booking:

|  |  |  |
| --- | --- | --- |
| Note | Type | Field name |
|  | Integer | Booking\_ID |
|  | String (varchar) | Booking\_place |
|  | String (char) | Booking\_Time |
|  | Integer | service\_ID |
|  | String (char) | Book\_Date |
|  | Integer | CarWash\_ID |
|  | Integer | Customer\_ID |

Service:

|  |  |  |
| --- | --- | --- |
| Note | Type | Field name |
|  | Integer | Service\_No |
| Customer\_ID | String (varchar) | Service\_Type |
|  | String (char) | Date |
|  | Float | Price |

**Future work :**

1. Creates smart systems for measuring wasting water in car washing process

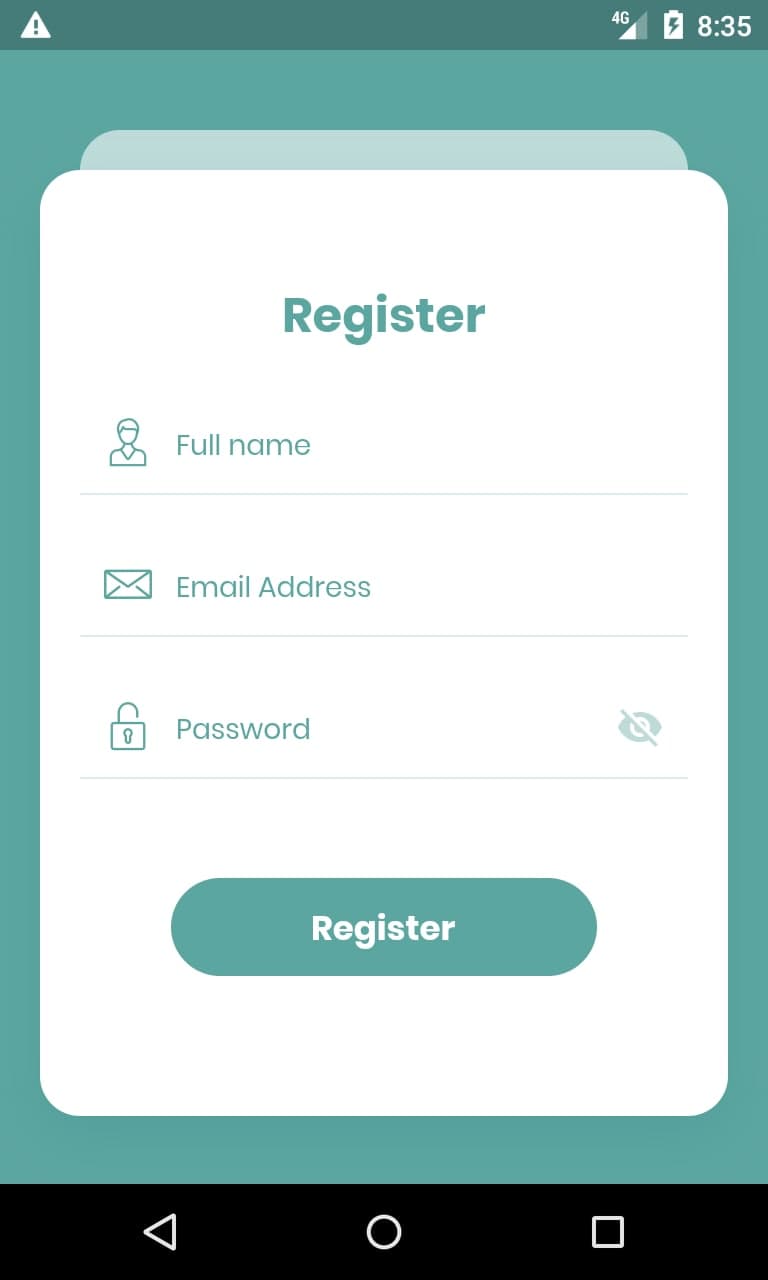
**Interface implementation**

**Chapter 4**

# **4.0-Flutter mobile application**

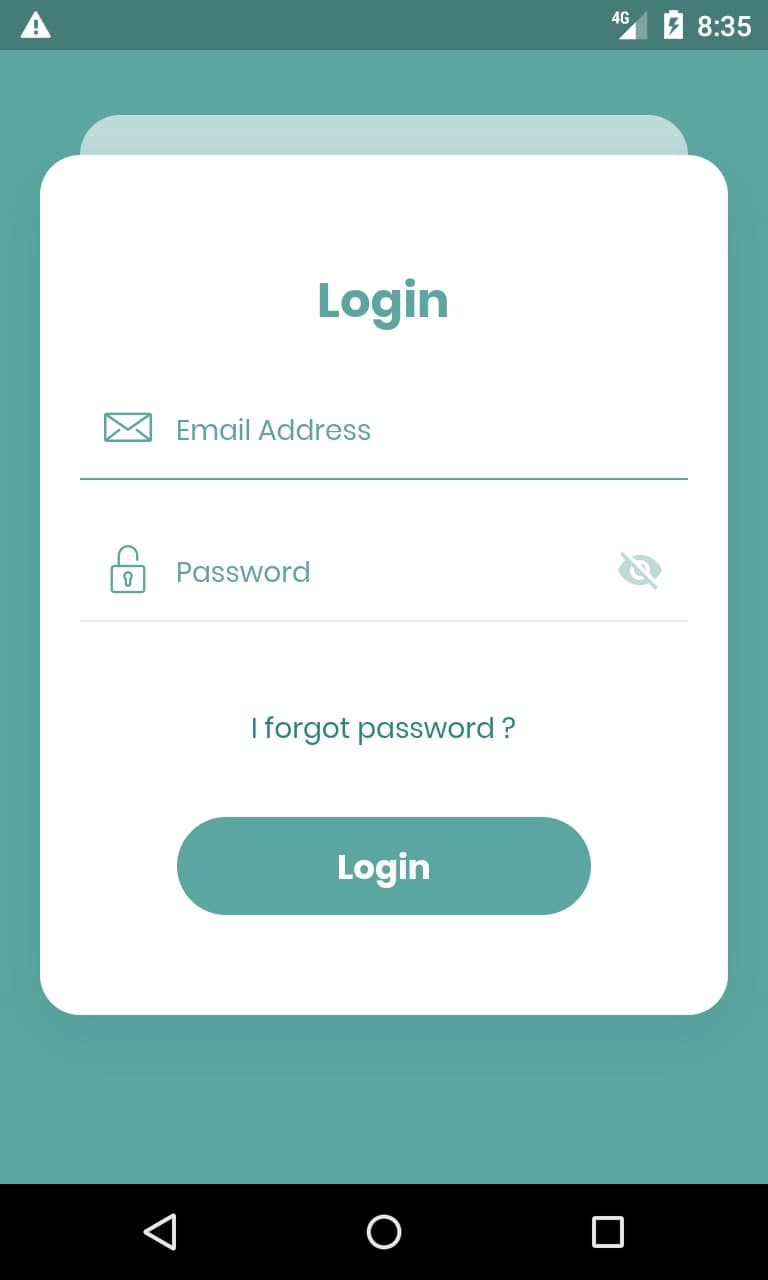
class SignUpWidget extends StatefulWidget {  
 @override  
 \_SignUpWidgetState createState() => \_SignUpWidgetState();  
}  
  
class \_SignUpWidgetState extends StateMVC<SignUpWidget> {  
 UserController \_con;  
  
 \_SignUpWidgetState() : super(UserController()) {  
 \_con = controller;  
 }  
  
 Widget build(BuildContext context) {  
 return Scaffold(  
 key: \_con.scaffoldKey,  
 backgroundColor: Theme.*of*(context).accentColor,  
 body: SingleChildScrollView(  
 child: Column(  
 children: <Widget>[  
 Stack(  
 children: <Widget>[  
 Container(  
 width: double.*infinity*,  
 padding: EdgeInsets.symmetric(vertical: 30, horizontal: 20),  
 margin: EdgeInsets.symmetric(vertical: 65, horizontal: 40),  
 decoration: BoxDecoration(  
 borderRadius: BorderRadius.circular(20),  
 color: Theme.*of*(context).primaryColor.withOpacity(0.6),  
 ),  
 ),  
 Container(  
 width: double.*infinity*,  
 padding: EdgeInsets.symmetric(vertical: 30, horizontal: 20),  
 margin: EdgeInsets.symmetric(vertical: 85, horizontal: 20),  
 decoration: BoxDecoration(  
 borderRadius: BorderRadius.circular(20),  
 color: Theme.*of*(context).primaryColor,  
 boxShadow: [  
 BoxShadow(  
 color: Theme.*of*(context).hintColor.withOpacity(0.2), offset: Offset(0, 10), blurRadius: 20)  
 ],  
 ),  
 child: Form(  
 key: \_con.loginFormKey,  
 child: Column(  
 children: <Widget>[  
 SizedBox(height: 25),  
 Text(S.*of*(context).register, style: Theme.*of*(context).textTheme.display3),  
 SizedBox(height: 20),  
 new TextFormField(  
 style: TextStyle(color: Theme.*of*(context).accentColor),  
 keyboardType: TextInputType.*text*,  
 onSaved: (input) => \_con.user.name = input,  
 validator: (input) => input.length < 3 ? S.*of*(context).should\_be\_more\_than\_3\_letters : null,  
 decoration: InputDecoration(  
 contentPadding: EdgeInsets.all(12),  
 hintText: S.*of*(context).full\_name,  
 hintStyle: Theme.*of*(context).textTheme.body1.merge(  
 TextStyle(color: Theme.*of*(context).accentColor),  
 ),  
 prefixIcon: Icon(UiIcons.*user\_1*, color: Theme.*of*(context).accentColor),  
 enabledBorder: UnderlineInputBorder(  
 borderSide: BorderSide(color: Theme.*of*(context).accentColor.withOpacity(0.2))),  
 focusedBorder:  
 UnderlineInputBorder(borderSide: BorderSide(color: Theme.*of*(context).accentColor)),  
 ),  
 ),  
 SizedBox(height: 20),  
 new TextFormField(  
 style: TextStyle(color: Theme.*of*(context).accentColor),  
 keyboardType: TextInputType.*emailAddress*,  
 onSaved: (input) => \_con.user.email = input,  
 validator: (input) => !input.contains('@') ? S.*of*(context).should\_be\_a\_valid\_email : null,  
 decoration: new InputDecoration(  
 hintText: S.*of*(context).email\_address,  
 hintStyle: Theme.*of*(context).textTheme.body1.merge(  
 TextStyle(color: Theme.*of*(context).accentColor),  
 ),  
 enabledBorder: UnderlineInputBorder(  
 borderSide: BorderSide(color: Theme.*of*(context).accentColor.withOpacity(0.2))),  
 focusedBorder:  
 UnderlineInputBorder(borderSide: BorderSide(color: Theme.*of*(context).accentColor)),  
 prefixIcon: Icon(  
 UiIcons.*envelope*,  
 color: Theme.*of*(context).accentColor,  
 ),  
 ),  
 ),  
 SizedBox(height: 20),  
 new TextFormField(  
 style: TextStyle(color: Theme.*of*(context).accentColor),  
 keyboardType: TextInputType.*text*,  
 obscureText: \_con.hidePassword,  
 onSaved: (input) => \_con.user.password = input,  
 validator: (input) => input.length < 3 ? S.*of*(context).should\_be\_more\_than\_3\_characters : null,  
 decoration: new InputDecoration(  
 hintText: S.*of*(context).password,  
 hintStyle: Theme.*of*(context).textTheme.body1.merge(  
 TextStyle(color: Theme.*of*(context).accentColor),  
 ),  
 enabledBorder: UnderlineInputBorder(  
 borderSide: BorderSide(color: Theme.*of*(context).accentColor.withOpacity(0.2))),  
 focusedBorder:  
 UnderlineInputBorder(borderSide: BorderSide(color: Theme.*of*(context).accentColor)),  
 prefixIcon: Icon(  
 UiIcons.*padlock*,  
 color: Theme.*of*(context).accentColor,  
 ),  
 suffixIcon: IconButton(  
 onPressed: () {  
 setState(() {  
 \_con.hidePassword = !\_con.hidePassword;  
 });  
 },  
 color: Theme.*of*(context).accentColor.withOpacity(0.4),  
 icon: Icon(\_con.hidePassword ? Icons.*visibility\_off* : Icons.*visibility*),  
 ),  
 ),  
 ),  
 SizedBox(height: 50),  
   
 FlatButton(  
 padding: EdgeInsets.symmetric(vertical: 12, horizontal: 70),  
 onPressed: () {  
 \_con.register();  
 },  
 child: Text(  
 S.*of*(context).register,  
 style: Theme.*of*(context).textTheme.title.merge(  
 TextStyle(color: Theme.*of*(context).primaryColor),  
 ),  
 ),  
 color: Theme.*of*(context).accentColor,  
 shape: StadiumBorder(),  
 ),  
 SizedBox(height: 40),  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 FlatButton(  
 onPressed: () {  
 Navigator.*of*(context).pushNamed('/Login');  
 },  
 child: RichText(  
 text: TextSpan(  
 style: Theme.*of*(context).textTheme.title.merge(  
 TextStyle(color: Theme.*of*(context).primaryColor,fontSize: 16,fontWeight: FontWeight.*w600*),  
 ),  
 children: [  
 TextSpan(text: S.*of*(context).i\_have\_account\_back\_to\_login),  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

signup screen



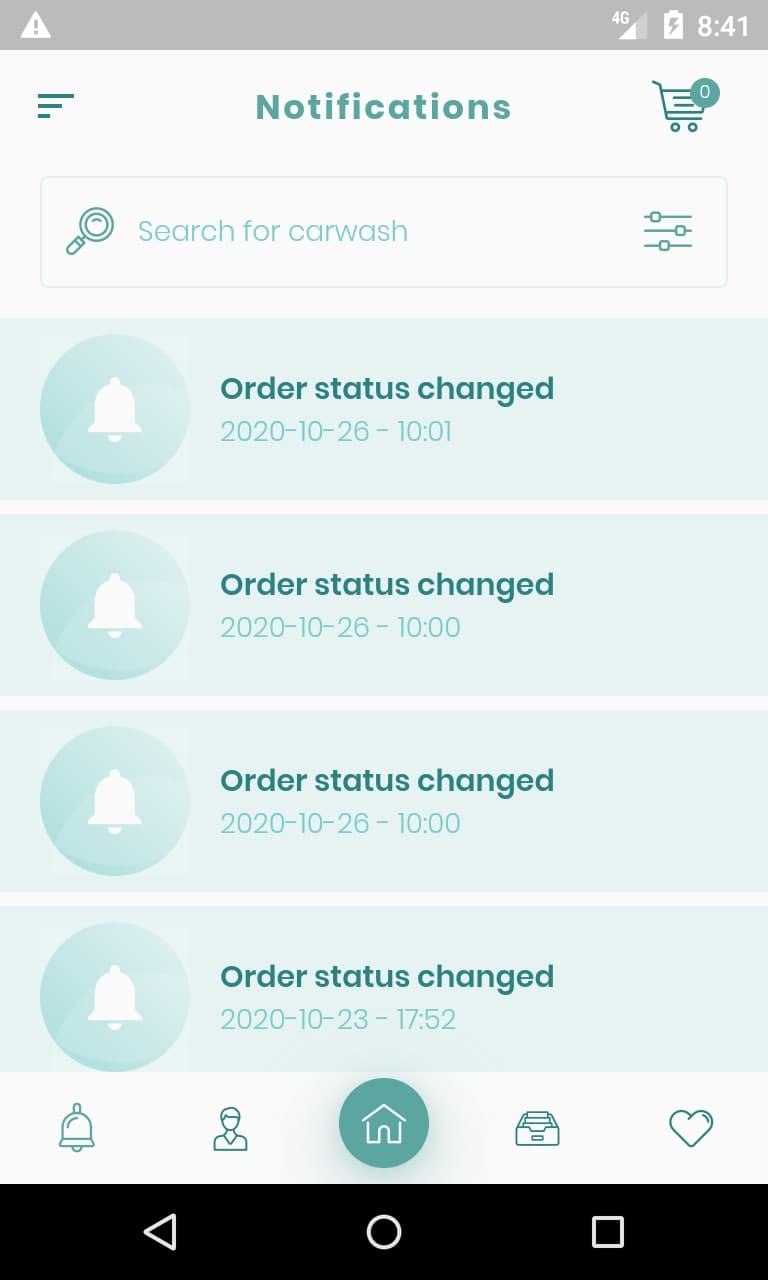
class SignInWidget extends StatefulWidget {  
 @override  
 \_SignInWidgetState createState() => \_SignInWidgetState();  
}  
  
class \_SignInWidgetState extends StateMVC<SignInWidget> {  
 UserController \_con;  
  
 \_SignInWidgetState() : super(UserController()) {  
 \_con = controller;  
 }  
 @override  
 void initState() {  
 super.initState();  
 if (userRepo.currentUser.value.apiToken != null) {  
 Navigator.*of*(context).pushReplacementNamed('/Pages', arguments: 2);  
 }  
 }  
  
 Widget build(BuildContext context) {  
 return Scaffold(  
 key: \_con.scaffoldKey,  
 backgroundColor: Theme.*of*(context).accentColor,  
 body: SingleChildScrollView(  
 child: Column(  
 children: <Widget>[  
 Stack(  
 children: <Widget>[  
 Container(  
 width: double.*infinity*,  
 padding: EdgeInsets.symmetric(vertical: 30, horizontal: 20),  
 margin: EdgeInsets.symmetric(vertical: 65, horizontal: 40),  
 decoration: BoxDecoration(  
 borderRadius: BorderRadius.circular(20),  
 color: Theme.*of*(context).primaryColor.withOpacity(0.6),  
 ),  
 ),  
 Container(  
 width: double.*infinity*,  
 padding: EdgeInsets.symmetric(vertical: 30, horizontal: 20),  
 margin: EdgeInsets.symmetric(vertical: 85, horizontal: 20),  
 decoration: BoxDecoration(  
 borderRadius: BorderRadius.circular(20),  
 color: Theme.*of*(context).primaryColor,  
 boxShadow: [  
 BoxShadow(  
 color: Theme.*of*(context).hintColor.withOpacity(0.2), offset: Offset(0, 10), blurRadius: 20)  
 ]),  
 child: Form(  
 key: \_con.loginFormKey,  
 child: Column(  
 children: <Widget>[  
 SizedBox(height: 25),  
 Text(  
 S.*of*(context).login,  
 style: Theme.*of*(context).textTheme.display3),  
 SizedBox(height: 20),  
 new TextFormField(  
 style: TextStyle(color: Theme.*of*(context).accentColor),  
 keyboardType: TextInputType.*emailAddress*,  
 onSaved: (input) => \_con.user.email = input,  
 validator: (input) => !input.contains('@') ? S.*of*(context).should\_be\_a\_valid\_email : null,  
 decoration: new InputDecoration(  
 hintText: S.*of*(context).email\_address,  
 hintStyle: Theme.*of*(context).textTheme.body1.merge(  
 TextStyle(color: Theme.*of*(context).accentColor),  
 ),  
 enabledBorder: UnderlineInputBorder(  
 borderSide: BorderSide(color: Theme.*of*(context).accentColor.withOpacity(0.2))),  
 focusedBorder:  
 UnderlineInputBorder(borderSide: BorderSide(color: Theme.*of*(context).accentColor)),  
 prefixIcon: Icon(  
 UiIcons.*envelope*,  
 color: Theme.*of*(context).accentColor,  
 ),  
 ),  
 ),  
 SizedBox(height: 20),  
 new TextFormField(  
 style: TextStyle(color: Theme.*of*(context).accentColor),  
 keyboardType: TextInputType.*text*,  
 onSaved: (input) => \_con.user.password = input,  
 validator: (input) => input.length < 3 ? S.*of*(context).should\_be\_more\_than\_3\_characters : null,  
 obscureText: \_con.hidePassword,  
 decoration: new InputDecoration(  
 hintText: S.*of*(context).password,  
 hintStyle: Theme.*of*(context).textTheme.body1.merge(  
 TextStyle(color: Theme.*of*(context).accentColor),  
 ),  
 enabledBorder: UnderlineInputBorder(  
 borderSide: BorderSide(color: Theme.*of*(context).accentColor.withOpacity(0.2))),  
 focusedBorder:  
 UnderlineInputBorder(borderSide: BorderSide(color: Theme.*of*(context).accentColor)),  
 prefixIcon: Icon(  
 UiIcons.*padlock*,  
 color: Theme.*of*(context).accentColor,  
 ),  
 suffixIcon: IconButton(  
 onPressed: () {  
 setState(() {  
 \_con.hidePassword = !\_con.hidePassword;  
 });  
 },  
 color: Theme.*of*(context).accentColor.withOpacity(0.4),  
 icon: Icon(\_con.hidePassword ? Icons.*visibility\_off* : Icons.*visibility*),  
 ),  
 ),  
 ),  
 SizedBox(height: 30),  
 FlatButton(  
 onPressed: () {  
 Navigator.*of*(context).pushReplacementNamed('/ForgetPassword');  
 },  
 child: Text(  
 S.*of*(context).i\_forgot\_password,  
 style: Theme.*of*(context).textTheme.body1,  
 ),  
 ),  
 SizedBox(height: 20),  
 FlatButton(  
 padding: EdgeInsets.symmetric(vertical: 12, horizontal: 80),  
 onPressed: () {  
 \_con.login();  
 },  
 child: Text(  
 S.*of*(context).login,  
 style: Theme.*of*(context).textTheme.title.merge(  
 TextStyle(color: Theme.*of*(context).primaryColor),  
 ),  
 ),  
 color: Theme.*of*(context).accentColor,  
 shape: StadiumBorder(),  
 ),  
 SizedBox(height: 15),  
 SizedBox(height: 5),  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 FlatButton(  
 onPressed: () {  
 Navigator.*of*(context).pushNamed('/SignUp');  
 },  
 child: RichText(  
 text: TextSpan(  
 style: Theme.*of*(context).textTheme.title.merge(  
 TextStyle(color: Theme.*of*(context).primaryColor,fontSize: 16,fontWeight: FontWeight.*w600*),  
 ),  
 children: [  
 TextSpan(text: S.*of*(context).i\_dont\_have\_an\_account),  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

signin screen

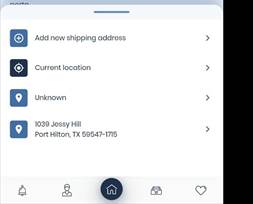


class NotificationsWidget extends StatefulWidget {  
  
 final GlobalKey<ScaffoldState> parentScaffoldKey;  
  
 const NotificationsWidget({Key key, this.parentScaffoldKey}) : super(key: key);  
 @override  
 \_NotificationsWidgetState createState() => \_NotificationsWidgetState();  
}  
  
class \_NotificationsWidgetState extends StateMVC<NotificationsWidget> {  
 NotificationController \_con;  
  
 \_NotificationsWidgetState() : super(NotificationController()){  
 \_con =controller;  
 }  
  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 key:\_con.scaffoldKey,  
 appBar: AppBar(  
 automaticallyImplyLeading: false,  
 leading: new IconButton(  
 icon: new Icon(Icons.*sort*, color: Theme.*of*(context).hintColor),  
 onPressed: () => widget.parentScaffoldKey.currentState.openDrawer(),  
 ),  
 backgroundColor: Colors.*transparent*,  
 elevation: 0,  
 centerTitle: true,  
 title: ValueListenableBuilder(  
 valueListenable: settingsRepo.setting,  
 builder: (context, value, child) {  
 return Text(  
 S.*of*(context).notifications,  
 style: Theme.*of*(context).textTheme.title.merge(TextStyle(letterSpacing: 1.3)),  
 );  
 },  
 ),  
 actions: <Widget>[  
 new ShoppingCartButtonWidget(  
 iconColor: Theme.*of*(context).hintColor, labelColor: Theme.*of*(context).accentColor  
 ),  
 ],  
 ),  
 body: currentUser.value.apiToken == null  
 ? PermissionDeniedWidget()  
 : RefreshIndicator(  
 onRefresh: \_con.refreshNotifications,  
 child:SingleChildScrollView(  
 padding: EdgeInsets.symmetric(vertical: 7),  
 child: Column(  
 children: <Widget>[  
 Padding(  
 padding: const EdgeInsets.symmetric(horizontal: 20),  
 child: SearchBarWidget(  
 onClickFilter: (event) {  
 widget.parentScaffoldKey.currentState.openEndDrawer();  
 },  
 ),  
 ),  
 Offstage(  
 offstage: \_con.notifications.isEmpty,  
 child: ListView.separated(  
 padding: EdgeInsets.symmetric(vertical: 15),  
 shrinkWrap: true,  
 primary: false,  
 itemCount: \_con.notifications.length,  
 separatorBuilder: (context, index) {  
 return SizedBox(height: 7);  
 },  
 itemBuilder: (context, index) {  
 return NotificationItemWidget(  
 notification: \_con.notifications.elementAt(index),  
 onDismissed: (notification) {  
 setState(() {  
 \_con.removeFromNotification(\_con.notifications.elementAt(index));  
 });  
 },  
 );  
 },  
 ),  
 ),  
 Offstage(  
 offstage: \_con.notifications.isNotEmpty,  
 child: EmptyNotificationsWidget(),  
 )  
 ],  
 ),  
 ),  
 )  
 );  
 }  
}

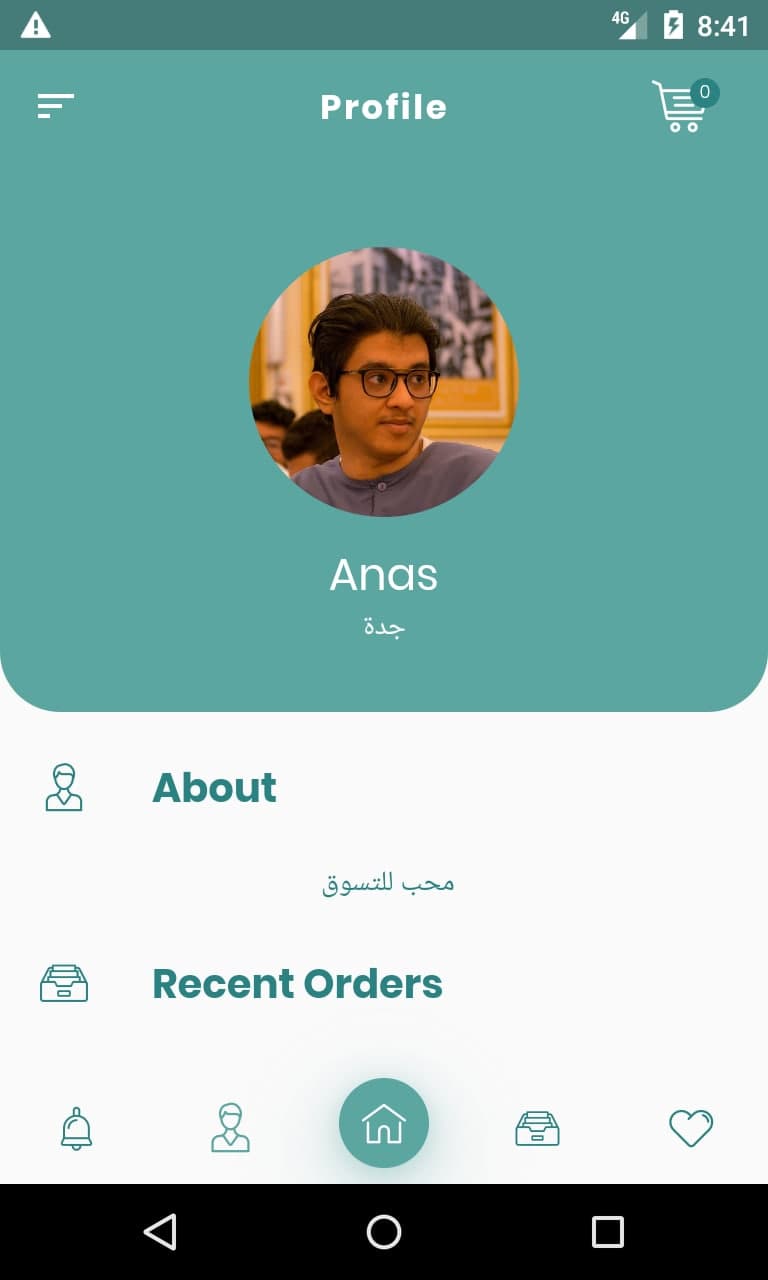
notification screen



class DeliveryAddressesWidget extends StatefulWidget {  
 final RouteArgument routeArgument;  
  
 DeliveryAddressesWidget({Key key, this.routeArgument}) : super(key: key);  
  
 @override  
 \_DeliveryAddressesWidgetState createState() => \_DeliveryAddressesWidgetState();  
}  
  
class \_DeliveryAddressesWidgetState extends StateMVC<DeliveryAddressesWidget> {  
 DeliveryAddressesController \_con;  
 PaymentMethodList list;  
  
 \_DeliveryAddressesWidgetState() : super(DeliveryAddressesController()) {  
 \_con = controller;  
 }  
  
 @override  
 void initState() {  
 list = new PaymentMethodList();  
 super.initState();  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 key: \_con.scaffoldKey,  
 appBar: AppBar(  
 backgroundColor: Colors.*transparent*,  
 elevation: 0,  
 centerTitle: true,  
 leading: new IconButton(  
 icon: new Icon(UiIcons.*return\_icon*,  
 color: Theme.*of*(context).hintColor),  
 onPressed: () => Navigator.*of*(context).pop(),  
 ),  
 title: Text(  
 S.*of*(context).delivery\_addresses,  
 style: Theme.*of*(context).textTheme.title.merge(TextStyle(letterSpacing: 1.3)),  
 ),  
 actions: <Widget>[  
 new ShoppingCartButtonWidget(iconColor: Theme.*of*(context).hintColor, labelColor: Theme.*of*(context).accentColor),  
 ],  
 ),  
 floatingActionButton:// \_con.cart != null && \_con.cart.product.store.availableForDelivery  
 FloatingActionButton(  
 onPressed: () async {  
 LocationResult result = await showLocationPicker(  
 context,  
 setting.value.googleMapsKey,  
 initialCenter: LatLng(deliveryAddress.value?.latitude ?? 0, deliveryAddress.value?.longitude ?? 0),  
 //automaticallyAnimateToCurrentLocation: true,  
 //mapStylePath: 'assets/mapStyle.json',  
 myLocationButtonEnabled: true,  
 //resultCardAlignment: Alignment.bottomCenter,  
 );  
 \_con.addAddress(new Address.fromJSON({  
 'address': result.address,  
 'latitude': result.latLng.latitude,  
 'longitude': result.latLng.longitude,  
 }));  
 print("result = $result");  
 //setState(() => \_pickedLocation = result);  
 },  
 backgroundColor: Theme.*of*(context).accentColor,  
 child: Icon(  
 Icons.*add*,  
 color: Theme.*of*(context).primaryColor,  
 )),  
 //: SizedBox(height: 0),  
 body: RefreshIndicator(  
 onRefresh: \_con.refreshAddresses,  
 child: SingleChildScrollView(  
 child: Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 mainAxisAlignment: MainAxisAlignment.start,  
 mainAxisSize: MainAxisSize.max,  
 children: <Widget>[  
 Padding(  
 padding: const EdgeInsets.only(top: 10, left: 20, right: 20),  
 child: ListTile(  
 contentPadding: EdgeInsets.symmetric(vertical: 0),  
 leading: Icon(  
 Icons.*map*,  
 color: Theme.*of*(context).hintColor,  
 ),  
 title: Text(  
 S.*of*(context).delivery\_addresses,  
 maxLines: 1,  
 overflow: TextOverflow.ellipsis,  
 style: Theme.*of*(context).textTheme.display1,  
 ),  
 subtitle: Text(  
 S.*of*(context).long\_press\_to\_edit\_item\_swipe\_item\_to\_delete\_it,  
 maxLines: 2,  
 overflow: TextOverflow.ellipsis,  
 style: Theme.*of*(context).textTheme.caption,  
 ),  
 ),  
 ),  
  
 \_con.addresses.isEmpty  
 ? CircularLoadingWidget(height: 250)  
 : ListView.separated(  
 padding: EdgeInsets.symmetric(vertical: 15),  
 scrollDirection: Axis.vertical,  
 shrinkWrap: true,  
 primary: false,  
 itemCount: \_con.addresses.length,  
 separatorBuilder: (context, index) {  
 return SizedBox(height: 15);  
 },  
 itemBuilder: (context, index) {  
 return DeliveryAddressesItemWidget(  
 address: \_con.addresses.elementAt(index),  
 onPressed: (Address \_address) {  
 DeliveryAddressDialog(  
 context: context,  
 address: \_address,  
 onChanged: (Address \_address) {  
 \_con.updateAddress(\_address);  
 },  
 );  
 },  
 onLongPress: (Address \_address) {  
 DeliveryAddressDialog(  
 context: context,  
 address: \_address,  
 onChanged: (Address \_address) {  
 \_con.updateAddress(\_address);  
 },  
 );  
 },  
 onDismissed: (Address \_address) {  
 \_con.removeDeliveryAddress(\_address);  
 },  
 );  
 },  
 ),  
 ],  
 ),  
 ),  
 ),  
 );

addresses screen 

class AccountWidget extends StatefulWidget {  
 final GlobalKey<ScaffoldState> parentScaffoldKey;  
  
 AccountWidget({Key key, this.parentScaffoldKey}) : super(key: key);  
 @override  
 \_AccountWidgetState createState() => \_AccountWidgetState();  
}  
  
class \_AccountWidgetState extends StateMVC<AccountWidget> {  
 ProfileController \_con;  
  
 \_AccountWidgetState() : super(ProfileController()) {  
 \_con = controller;  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 final theme = Theme.*of*(context).copyWith(dividerColor: Colors.*transparent*);  
 return Scaffold(  
 key: \_con.scaffoldKey,  
 drawer: DrawerWidget(),  
 appBar: AppBar(  
 automaticallyImplyLeading: false,  
 leading: new IconButton(  
 icon: new Icon(Icons.*sort*, color: currentUser.value.apiToken == null?Theme.*of*(context).hintColor:Theme.*of*(context).primaryColor),  
 onPressed: () => widget.parentScaffoldKey.currentState.openDrawer(),  
 ),  
  
 backgroundColor: currentUser.value.apiToken == null?Colors.*transparent*:Theme.*of*(context).accentColor,  
 elevation: 0,  
 centerTitle: true,  
 title: Text(  
 S.*of*(context).profile,  
 style: currentUser.value.apiToken == null  
 ?Theme.*of*(context)  
 .textTheme  
 .title  
 .merge(TextStyle(letterSpacing: 1.3))  
 :Theme.*of*(context)  
 .textTheme  
 .title  
 .merge(TextStyle(letterSpacing: 1.3, color:Theme.*of*(context).primaryColor)),  
 ),  
 actions: <Widget>[  
 new ShoppingCartButtonWidget(  
 iconColor: currentUser.value.apiToken == null?Theme.*of*(context).hintColor:Theme.*of*(context).primaryColor,  
 labelColor: currentUser.value.apiToken == null?Theme.*of*(context).accentColor:Theme.*of*(context).hintColor),  
 ],  
 ),  
 body: currentUser.value.apiToken == null  
 ? PermissionDeniedWidget()  
 : SingleChildScrollView(  
// padding: const EdgeInsets.symmetric(horizontal: 0, vertical: 10),  
 child: Column(  
 children: <Widget>[  
 ProfileAvatarWidget(user: currentUser.value),  
 ListTile(  
 contentPadding: EdgeInsets.symmetric(horizontal: 20, vertical: 10),  
 leading: Icon(  
 UiIcons.*user\_1*,  
 color: Theme.*of*(context).hintColor,  
 ),  
 title: Text(  
 S.*of*(context).about,  
 style: Theme.*of*(context).textTheme.display1,  
 ),  
 ),  
 Padding(  
 padding: const EdgeInsets.symmetric(horizontal: 20),  
 child: Text(  
 currentUser.value?.bio ?? "",  
 style: Theme.*of*(context).textTheme.body1,  
 ),  
 ),  
 ListTile(  
 contentPadding: EdgeInsets.symmetric(horizontal: 20, vertical: 10),  
 leading: Icon(  
 UiIcons.*inbox*,  
 color: Theme.*of*(context).hintColor,  
 ),  
 title: Text(  
 S.*of*(context).recent\_orders,  
 style: Theme.*of*(context).textTheme.display1,  
 ),  
 ),  
 \_con.recentOrders.isEmpty  
 ? CircularLoadingWidget(height: 200)  
 : ListView.builder(  
 scrollDirection: Axis.vertical,  
 shrinkWrap: true,  
 primary: false,  
 itemCount: \_con.recentOrders.length,  
 itemBuilder: (context, index) {  
 return Theme(  
 data: theme,  
 child: ExpansionTile(  
 initiallyExpanded: true,  
 title: Row(  
 children: <Widget>[  
 Expanded(  
 child: Text(  
 '${S.*of*(context).order\_id}: #${\_con.recentOrders.elementAt(index).id}')),  
 Text(  
 '${\_con.recentOrders.elementAt(index).orderStatus.status}',  
 style: Theme.*of*(context).textTheme.caption,  
 ),  
 ],  
 ),  
 children: List.generate(\_con.recentOrders.elementAt(index).productOrders.length,  
 (indexProduct) {  
 return OrderItemWidget(  
 heroTag: 'recent\_orders',  
 order: \_con.recentOrders.elementAt(index),  
 productOrder:  
 \_con.recentOrders.elementAt(index).productOrders.elementAt(indexProduct));  
 }),  
 ),  
 );  
 },  
 ),  
 ],  
 ),  
 ),  
 );  
 }  
}

account profile

* 1. **Laravel application models**

*/\*\**

*\* Class User*  
 *\** ***@package*** *App\Models*  
 *\** ***@version*** *July 10, 2018, 11:44 am UTC*  
 *\**  
 *\** ***@property*** *string name*  
 *\** ***@property*** *string email*  
 *\** ***@property*** *string password*  
 *\** ***@property*** *string api\_token*  
 *\** ***@property*** *string device\_token*  
 *\*/*  
class User extends *Authenticatable* implements HasMedia  
{  
 use Notifiable;  
 use Billable;  
 use HasMediaTrait {  
 getFirstMediaUrl as protected getFirstMediaUrlTrait;  
 }  
 use HasRoles;  
  
 */\*\**  
 *\* Validation rules*  
 *\**  
 *\** ***@var*** *array*  
 *\*/*  
public static *$rules* = [  
 'name' => 'required|string|max:255',  
 'email' => 'required|string|email|max:255|unique:users',  
 // 'password' => 'required',  
 ];  
 public $table = 'users';  
 */\*\**  
 *\* The attributes that are mass assignable.*  
 *\**  
 *\** ***@var*** *array*  
 *\*/*  
public $fillable = [  
 'name',  
 'email',  
 'password',  
 'api\_token',  
 'device\_token',  
 ];  
 */\*\**  
 *\* The attributes that should be casted to native types.*  
 *\**  
 *\** ***@var*** *array*  
 *\*/*  
protected $casts = [  
 'name' => 'string',  
 'email' => 'string',  
 'password' => 'string',  
 'api\_token' => 'string',  
 'device\_token' => 'string',  
 'remember\_token' => 'string'  
 ];  
 */\*\**  
 *\* New Attributes*  
 *\**  
 *\** ***@var*** *array*  
 *\*/*  
protected $appends = [  
 'custom\_fields',  
 'has\_media'  
 ];  
  
 */\*\**  
 *\* The attributes that should be hidden for arrays.*  
 *\**  
 *\** ***@var*** *array*  
 *\*/*  
protected $hidden = [  
 'password', 'remember\_token',  
 ];  
  
 */\*\**  
 *\* Specifies the user's FCM token*  
 *\**  
 *\** ***@return*** *string*  
 *\*/*  
public function routeNotificationForFcm($notification)  
 {  
 return $this->device\_token;  
 }  
  
 */\*\**  
 *\** ***@param*** *Media|null $media*  
 *\** ***@throws*** *\Spatie\Image\Exceptions\InvalidManipulation*  
 *\*/*  
public function registerMediaConversions(\Spatie\MediaLibrary\Models\Media $media = null)  
 {  
 $this->addMediaConversion('thumb')  
 ->fit(Manipulations::*FIT\_CROP*, 200, 200)  
 ->sharpen(10);  
  
 $this->addMediaConversion('icon')  
 ->fit(Manipulations::*FIT\_CROP*, 100, 100)  
 ->sharpen(10);  
 }  
  
 */\*\**  
 *\* to generate media url in case of fallback will*  
 *\* return the file type icon*  
 *\** ***@param*** *string $conversion*  
 *\** ***@return*** *string url*  
 *\*/*  
public function getFirstMediaUrl($collectionName = 'default', $conversion = '')  
 {  
 $url = $this->getFirstMediaUrlTrait($collectionName);  
 if ($url) {  
 $array = explode('.', $url);  
 $extension = strtolower(end($array));  
 if (in\_array($extension, config('medialibrary.extensions\_has\_thumb'))) {  
 return asset($this->getFirstMediaUrlTrait($collectionName, $conversion));  
 } else {  
 return asset(config('medialibrary.icons\_folder') . '/' . $extension . '.png');  
 }  
 }else{  
 return asset('images/avatar\_default.png');  
 }  
 }  
  
 public function getCustomFieldsAttribute()  
 {  
 $hasCustomField = in\_array(static::class, setting('custom\_field\_models', []));  
 if (!$hasCustomField) {  
 return [];  
 }  
 $array = $this->customFieldsValues()  
 ->join('custom\_fields', 'custom\_fields.id', '=', 'custom\_field\_values.custom\_field\_id')  
// ->where('custom\_fields.in\_table', '=', true)  
 ->select(['value', 'view', 'name'])  
 ->get()->toArray();  
  
 return convertToAssoc($array, 'name');  
 }  
  
 public function customFieldsValues()  
 {  
 return $this->morphMany('App\Models\CustomFieldValue', 'customizable');  
 }  
  
 */\*\**  
 *\* Add Media to api results*  
 *\** ***@return*** *bool*  
 *\*/*  
public function getHasMediaAttribute()  
 {  
 return $this->hasMedia('avatar') ? true : false;  
 }  
  
 */\*\**  
 *\** ***@return*** *\Illuminate\Database\Eloquent\Relations\BelongsToMany*  
 *\*\*/*  
public function stores()  
 {  
 return $this->belongsToMany(\App\Models\Store::class, 'user\_stores');  
 }

user model

*\*/*class DeliveryAddress extends *Model*{ public $table = 'delivery\_addresses';  public $fillable = [ 'description', 'address', 'latitude', 'longitude', 'is\_default', 'user\_id' ]; */\*\** *\* The attributes that should be casted to native types.* *\** *\** ***@var*** *array* *\*/*protected $casts = [ 'description' => 'string', 'address' => 'string', 'latitude' => 'double', 'longitude' => 'double', 'is\_default' => 'boolean', 'user\_id' => 'integer' ]; */\*\** *\* Validation rules* *\** *\** ***@var*** *array* *\*/*public static *$rules* = [ 'description' => 'required', 'address' => 'required', 'user\_id' => 'required|exists:users,id' ]; */\*\** *\* New Attributes* *\** *\** ***@var*** *array* *\*/*protected $appends = [ 'custom\_fields',  ]; public function customFieldsValues() { return $this->morphMany('App\Models\CustomFieldValue', 'customizable'); } public function getCustomFieldsAttribute() { $hasCustomField = in\_array(static::class,setting('custom\_field\_models',[])); if (!$hasCustomField){ return []; } $array = $this->customFieldsValues() ->join('custom\_fields','custom\_fields.id','=','custom\_field\_values.custom\_field\_id') ->where('custom\_fields.in\_table','=',true) ->get()->toArray(); return convertToAssoc($array,'name'); } */\*\** *\** ***@return*** *\Illuminate\Database\Eloquent\Relations\BelongsTo* *\*\*/*public function user() { return $this->belongsTo(\App\Models\User::class, 'user\_id', 'id'); }

address model

class Notification extends *Model*{ public $table = 'notifications'; protected $primaryKey = 'id'; // or null public $incrementing = false;  public $fillable = [ 'type', 'read\_at' ]; */\*\** *\* The attributes that should be casted to native types.* *\** *\** ***@var*** *array* *\*/*protected $casts = [ 'type' => 'string', 'read\_at' => 'date' ]; */\*\** *\* Validation rules* *\** *\** ***@var*** *array* *\*/*public static *$rules* = [ 'type' => 'required', ]; */\*\** *\* New Attributes* *\** *\** ***@var*** *array* *\*/*protected $appends = [ 'custom\_fields',  ]; public function customFieldsValues() { return $this->morphMany('App\Models\CustomFieldValue', 'customizable'); } public function getCustomFieldsAttribute() { $hasCustomField = in\_array(static::class,setting('custom\_field\_models',[])); if (!$hasCustomField){ return []; } $array = $this->customFieldsValues() ->join('custom\_fields','custom\_fields.id','=','custom\_field\_values.custom\_field\_id') ->where('custom\_fields.in\_table','=',true) ->get()->toArray(); return convertToAssoc($array,'name'); } */\*\** *\** ***@return*** *\Illuminate\Database\Eloquent\Relations\BelongsTo* *\*\*/*public function user() { return $this->belongsTo(\App\Models\User::class, 'notifiable\_id', 'id'); } }

notification model

* 1. **Laravel application controller**

class LoginController extends Controller  
{  
 /\*  
 |--------------------------------------------------------------------------  
 | Login Controller  
 |--------------------------------------------------------------------------  
 |  
 | This controller handles authenticating users for the application and  
 | redirecting them to your home screen. The controller uses a trait  
 | to conveniently provide its functionality to your applications.  
 |  
 \*/  
  
 use AuthenticatesUsers;  
  
 */\*\**  
 *\* Where to redirect users after login.*  
 *\**  
 *\** ***@var*** *string*  
 *\*/*  
protected $redirectTo = '/';  
 private $userRepository;  
 private $uploadRepository;  
 private $roleRepository;  
  
 */\*\**  
 *\* Create a new controller instance.*  
 *\**  
 *\** ***@return*** *void*  
 *\*/*  
public function \_\_construct(UserRepository $userRepository, UploadRepository $uploadRepository, RoleRepository $roleRepository)  
 {  
 $this->middleware('guest')->except('logout');  
 $this->userRepository = $userRepository;  
 $this->uploadRepository = $uploadRepository;  
 $this->roleRepository = $roleRepository;  
 }  
  
  
}

login controller

class RegisterController extends Controller  
{  
 /\*  
 |--------------------------------------------------------------------------  
 | Register Controller  
 |--------------------------------------------------------------------------  
 |  
 | This controller handles the registration of new users as well as their  
 | validation and creation. By default this controller uses a trait to  
 | provide this functionality without requiring any additional code.  
 |  
 \*/  
  
 use RegistersUsers;  
  
 */\*\**  
 *\* Where to redirect users after registration.*  
 *\**  
 *\** ***@var*** *string*  
 *\*/*  
protected $redirectTo = '/';  
  
 private $userRepository;  
 private $uploadRepository;  
 private $roleRepository;  
  
 */\*\**  
 *\* Create a new controller instance.*  
 *\**  
 *\** ***@return*** *void*  
 *\*/*  
public function \_\_construct(UserRepository $userRepository, UploadRepository $uploadRepository, RoleRepository $roleRepository)  
 {  
 $this->middleware('guest');  
 $this->userRepository = $userRepository;  
 $this->uploadRepository = $uploadRepository;  
 $this->roleRepository = $roleRepository;  
 }  
  
 */\*\**  
 *\* Get a validator for an incoming registration request.*  
 *\**  
 *\** ***@param*** *array $data*  
 *\** ***@return*** *\Illuminate\Contracts\Validation\Validator*  
 *\*/*  
protected function validator(array $data)  
 {  
 return Validator::make($data, [  
 'name' => 'required|string|max:255',  
 'email' => 'required|string|email|max:255|unique:users',  
 'password' => 'required|string|min:6|confirmed',  
 ]);  
 }  
  
 */\*\**  
 *\* Create a new user instance after a valid registration.*  
 *\**  
 *\** ***@param*** *array $data*  
 *\** ***@return***  
*\*/*  
protected function create(array $data)  
 {  
 $user = new User;  
 $user->name = $data['name'];  
 $user->email = $data['email'];  
 $user->password = Hash::make($data['password']);  
 $user->api\_token = str\_random(60);  
 $user->save();  
  
 $defaultRoles = $this->roleRepository->findByField('default', '1');  
 $defaultRoles = $defaultRoles->pluck('name')->toArray();  
 $user->assignRole($defaultRoles);  
  
 return $user;  
 }  
}

register controller

class NotificationController extends Controller  
{  
 */\*\** ***@var*** *NotificationRepository \*/*  
private $notificationRepository;  
  
 */\*\**  
 *\** ***@var*** *CustomFieldRepository*  
 *\*/*  
private $customFieldRepository;  
  
 private $userRepository;  
  
 public function \_\_construct(NotificationRepository $notificationRepo, CustomFieldRepository $customFieldRepo,  
 UserRepository $userRepo)  
 {  
 parent::*\_\_construct*();  
 $this->notificationRepository = $notificationRepo;  
 $this->customFieldRepository = $customFieldRepo;  
 $this->userRepository = $userRepo;  
 }  
  
 */\*\**  
 *\* Display a listing of the Notification.*  
 *\**  
 *\** ***@param*** *NotificationDataTable $notificationDataTable*  
 *\** ***@return*** *Response*  
 *\*/*  
public function index(NotificationDataTable $notificationDataTable)  
 {  
 return $notificationDataTable->render('notifications.index');  
 }  
  
 */\*\**  
 *\* Show the form for creating a new Notification.*  
 *\**  
 *\** ***@return*** *Response*  
 *\*/*  
public function create()  
 {  
 $user = $this->userRepository->pluck('name', 'id');  
  
 $hasCustomField = in\_array($this->notificationRepository->model(), setting('custom\_field\_models', []));  
 if ($hasCustomField) {  
 $customFields = $this->customFieldRepository->findByField('custom\_field\_model', $this->notificationRepository->model());  
 $html = generateCustomField($customFields);  
 }  
 return view('notifications.create')->with("customFields", isset($html) ? $html : false)->with("user", $user);  
 }  
  
 */\*\**  
 *\* Store a newly created Notification in storage.*  
 *\**  
 *\** ***@param*** *CreateNotificationRequest $request*  
 *\**  
 *\** ***@return*** *Response*  
 *\*/*  
public function store(CreateNotificationRequest $request)  
 {  
 $input = $request->all();  
 $customFields = $this->customFieldRepository->findByField('custom\_field\_model', $this->notificationRepository->model());  
 try {  
 $notification = $this->notificationRepository->create($input);  
 $notification->customFieldsValues()->createMany(getCustomFieldsValues($customFields, $request));  
  
 } catch (ValidatorException $e) {  
 Flash::*error*($e->getMessage());  
 }  
  
 Flash::*success*(\_\_('lang.saved\_successfully', ['operator' => \_\_('lang.notification')]));  
  
 return redirect(route('notifications.index'));  
 }  
  
 */\*\**  
 *\* Display the specified Notification.*  
 *\**  
 *\** ***@param*** *int $id*  
 *\**  
 *\** ***@return*** *Response*  
 *\*/*  
public function show($id)  
 {  
 $notification = $this->notificationRepository->findWithoutFail($id);  
  
 if (empty($notification)) {  
 Flash::*error*('Notification not found');  
  
 return redirect(route('notifications.index'));  
 }  
 try {  
 //dd((new Carbon('now'))->format('Y-m-d H:i:s'));  
 $notification = $this->notificationRepository->update(['read\_at' => (new Carbon())], $id);  
  
 } catch (ValidatorException $e) {  
 Flash::*error*($e->getMessage());  
 } catch (\Exception $e) {  
 Flash::*error*($e->getMessage());  
 }  
  
 return redirect(route('notifications.index'));  
 }  
  
 */\*\**  
 *\* Show the form for editing the specified Notification.*  
 *\**  
 *\** ***@param*** *int $id*  
 *\**  
 *\** ***@return*** *Response*  
 *\*/*  
public function edit($id)  
 {  
 $notification = $this->notificationRepository->findWithoutFail($id);  
 $user = $this->userRepository->pluck('name', 'id');  
  
  
 if (empty($notification)) {  
 Flash::*error*(\_\_('lang.not\_found', ['operator' => \_\_('lang.notification')]));  
  
 return redirect(route('notifications.index'));  
 }  
 $customFieldsValues = $notification->customFieldsValues()->with('customField')->get();  
 $customFields = $this->customFieldRepository->findByField('custom\_field\_model', $this->notificationRepository->model());  
 $hasCustomField = in\_array($this->notificationRepository->model(), setting('custom\_field\_models', []));  
 if ($hasCustomField) {  
 $html = generateCustomField($customFields, $customFieldsValues);  
 }  
  
 return view('notifications.edit')->with('notification', $notification)->with("customFields", isset($html) ? $html : false)->with("user", $user);  
 }  
  
 */\*\**  
 *\* Update the specified Notification in storage.*  
 *\**  
 *\** ***@param*** *int $id*  
 *\** ***@param*** *UpdateNotificationRequest $request*  
 *\**  
 *\** ***@return*** *Response*  
 *\*/*  
public function update($id, UpdateNotificationRequest $request)  
 {  
 $notification = $this->notificationRepository->findWithoutFail($id);  
  
 if (empty($notification)) {  
 Flash::*error*('Notification not found');  
 return redirect(route('notifications.index'));  
 }  
 $input = $request->all();  
 $customFields = $this->customFieldRepository->findByField('custom\_field\_model', $this->notificationRepository->model());  
 try {  
 $notification = $this->notificationRepository->update($input, $id);  
 foreach (getCustomFieldsValues($customFields, $request) as $value) {  
 $notification->customFieldsValues()  
 ->updateOrCreate(['custom\_field\_id' => $value['custom\_field\_id']], $value);  
 }  
 } catch (ValidatorException $e) {  
 Flash::*error*($e->getMessage());  
 }  
  
 Flash::*success*(\_\_('lang.updated\_successfully', ['operator' => \_\_('lang.notification')]));  
  
 return redirect(route('notifications.index'));  
 }  
  
 */\*\**  
 *\* Remove the specified Notification from storage.*  
 *\**  
 *\** ***@param*** *int $id*  
 *\**  
 *\** ***@return*** *Response*  
 *\*/*  
public function destroy($id)  
 {  
 $notification = $this->notificationRepository->findWithoutFail($id);  
  
 if (empty($notification)) {  
 Flash::*error*('Notification not found');  
  
 return redirect(route('notifications.index'));  
 }  
  
 $this->notificationRepository->delete($id);  
  
 Flash::*success*(\_\_('lang.deleted\_successfully', ['operator' => \_\_('lang.notification')]));  
  
 return redirect(route('notifications.index'));  
 }  
  
 */\*\**  
 *\* Remove Media of Notification*  
 *\** ***@param*** *Request $request*  
 *\*/*  
public function removeMedia(Request $request)  
 {  
 $input = $request->all();  
 $notification = $this->notificationRepository->findWithoutFail($input['id']);  
 try {  
 if ($notification->hasMedia($input['collection'])) {  
 $notification->getFirstMedia($input['collection'])->delete();  
 }  
 } catch (\Exception $e) {  
 Log::error($e->getMessage());  
 }  
 }  
}

notification controller

class DeliveryAddressController extends Controller{ */\*\** ***@var*** *DeliveryAddressRepository \*/*private $deliveryAddressRepository; */\*\** *\** ***@var*** *CustomFieldRepository* *\*/*private $customFieldRepository; */\*\** *\** ***@var*** *UserRepository* *\*/*private $userRepository; public function \_\_construct(DeliveryAddressRepository $deliveryAddressRepo, CustomFieldRepository $customFieldRepo , UserRepository $userRepo) { parent::*\_\_construct*(); $this->deliveryAddressRepository = $deliveryAddressRepo; $this->customFieldRepository = $customFieldRepo; $this->userRepository = $userRepo; } */\*\** *\* Display a listing of the DeliveryAddress.* *\** *\** ***@param*** *DeliveryAddressDataTable $deliveryAddressDataTable* *\** ***@return*** *Response* *\*/*public function index(DeliveryAddressDataTable $deliveryAddressDataTable) { return $deliveryAddressDataTable->render('delivery\_addresses.index'); } */\*\** *\* Show the form for creating a new DeliveryAddress.* *\** *\** ***@return*** *Response* *\*/*public function create() { $user = $this->userRepository->pluck('name','id');  $hasCustomField = in\_array($this->deliveryAddressRepository->model(),setting('custom\_field\_models',[])); if($hasCustomField){ $customFields = $this->customFieldRepository->findByField('custom\_field\_model', $this->deliveryAddressRepository->model()); $html = generateCustomField($customFields); } return view('delivery\_addresses.create')->with("customFields", isset($html) ? $html : false)->with("user",$user); } */\*\** *\* Store a newly created DeliveryAddress in storage.* *\** *\** ***@param*** *CreateDeliveryAddressRequest $request* *\** *\** ***@return*** *Response* *\*/*public function store(CreateDeliveryAddressRequest $request) { $input = $request->all(); $customFields = $this->customFieldRepository->findByField('custom\_field\_model', $this->deliveryAddressRepository->model()); try { $deliveryAddress = $this->deliveryAddressRepository->create($input); $deliveryAddress->customFieldsValues()->createMany(getCustomFieldsValues($customFields,$request));  } catch (ValidatorException $e) { Flash::*error*($e->getMessage()); } Flash::*success*(\_\_('lang.saved\_successfully',['operator' => \_\_('lang.delivery\_address')])); return redirect(route('deliveryAddresses.index')); } */\*\** *\* Display the specified DeliveryAddress.* *\** *\** ***@param*** *int $id* *\** *\** ***@return*** *Response* *\*/*public function show($id) { $deliveryAddress = $this->deliveryAddressRepository->findWithoutFail($id); if (empty($deliveryAddress)) { Flash::*error*('Delivery Address not found'); return redirect(route('deliveryAddresses.index')); } return view('delivery\_addresses.show')->with('deliveryAddress', $deliveryAddress); } */\*\** *\* Show the form for editing the specified DeliveryAddress.* *\** *\** ***@param*** *int $id* *\** *\** ***@return*** *Response* *\*/*public function edit($id) { $deliveryAddress = $this->deliveryAddressRepository->findWithoutFail($id); $user = $this->userRepository->pluck('name','id');  if (empty($deliveryAddress)) { Flash::*error*(\_\_('lang.not\_found',['operator' => \_\_('lang.delivery\_address')])); return redirect(route('deliveryAddresses.index')); } $customFieldsValues = $deliveryAddress->customFieldsValues()->with('customField')->get(); $customFields = $this->customFieldRepository->findByField('custom\_field\_model', $this->deliveryAddressRepository->model()); $hasCustomField = in\_array($this->deliveryAddressRepository->model(),setting('custom\_field\_models',[])); if($hasCustomField) { $html = generateCustomField($customFields, $customFieldsValues); } return view('delivery\_addresses.edit')->with('deliveryAddress', $deliveryAddress)->with("customFields", isset($html) ? $html : false)->with("user",$user); } */\*\** *\* Update the specified DeliveryAddress in storage.* *\** *\** ***@param*** *int $id* *\** ***@param*** *UpdateDeliveryAddressRequest $request* *\** *\** ***@return*** *Response* *\*/*public function update($id, UpdateDeliveryAddressRequest $request) { $deliveryAddress = $this->deliveryAddressRepository->findWithoutFail($id); if (empty($deliveryAddress)) { Flash::*error*('Delivery Address not found'); return redirect(route('deliveryAddresses.index')); } $input = $request->all(); $customFields = $this->customFieldRepository->findByField('custom\_field\_model', $this->deliveryAddressRepository->model()); try { $deliveryAddress = $this->deliveryAddressRepository->update($input, $id);   foreach (getCustomFieldsValues($customFields, $request) as $value){ $deliveryAddress->customFieldsValues() ->updateOrCreate(['custom\_field\_id'=>$value['custom\_field\_id']],$value); } } catch (ValidatorException $e) { Flash::*error*($e->getMessage()); } Flash::*success*(\_\_('lang.updated\_successfully',['operator' => \_\_('lang.delivery\_address')])); return redirect(route('deliveryAddresses.index')); } */\*\** *\* Remove the specified DeliveryAddress from storage.* *\** *\** ***@param*** *int $id* *\** *\** ***@return*** *Response* *\*/*public function destroy($id) { $deliveryAddress = $this->deliveryAddressRepository->findWithoutFail($id); if (empty($deliveryAddress)) { Flash::*error*('Delivery Address not found'); return redirect(route('deliveryAddresses.index')); } $this->deliveryAddressRepository->delete($id); Flash::*success*(\_\_('lang.deleted\_successfully',['operator' => \_\_('lang.delivery\_address')])); return redirect(route('deliveryAddresses.index')); } */\*\** *\* Remove Media of DeliveryAddress* *\** ***@param*** *Request $request* *\*/*public function removeMedia(Request $request) { $input = $request->all(); $deliveryAddress = $this->deliveryAddressRepository->findWithoutFail($input['id']); try { if($deliveryAddress->hasMedia($input['collection'])){ $deliveryAddress->getFirstMedia($input['collection'])->delete(); } } catch (\Exception $e) { Log::error($e->getMessage()); } }}

**delivery address controller**

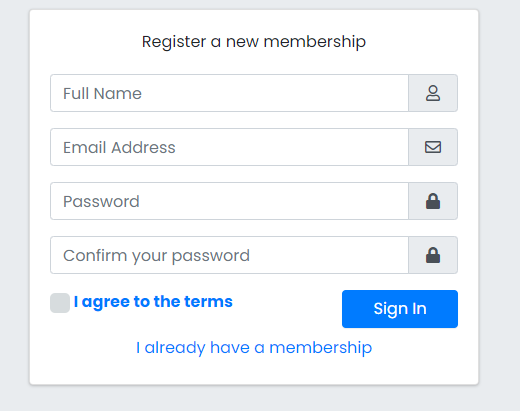
**4.3- Laravel application web page**

<!DOCTYPE html><html lang="{{setting('language','app()->getLocale()')}}"><head> <meta charset="utf-8"> <meta http-equiv="X-UA-Compatible" content="IE=edge"> <title>{{setting('app\_name')}} | {{setting('app\_short\_description')}}</title> <!-- Tell the browser to be responsive to screen width --> <meta name="viewport" content="width=device-width, initial-scale=1"> <link rel="icon" type="image/png" href="{{$app\_logo}}" /> <!-- Font Awesome --> <link rel="stylesheet" href="{{asset('plugins/font-awesome/css/font-awesome.min.css')}}"> <!-- Ionicons --> <link href="https://unpkg.com/ionicons@4.1.2/dist/css/ionicons.min.css" rel="stylesheet"> <script src="https://kit.fontawesome.com/f129a6342d.js" crossorigin="anonymous"></script> <!-- Theme style --> <link rel="stylesheet" href="{{asset('dist/css/adminlte.css')}}"> @if(app()->getLocale()=='ar') <link rel="stylesheet" href="{{asset('dist/css/rtl/adminlte.css')}}"> @endif <!-- iCheck --> <link rel="stylesheet" href="{{asset('plugins/iCheck/flat/blue.css')}}">{{--<!-- Bootstrap -->--}} <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css" integrity="sha384-JcKb8q3iqJ61gNV9KGb8thSsNjpSL0n8PARn9HuZOnIxN0hoP+VmmDGMN5t9UJ0Z" crossorigin="anonymous">{{--<link rel="stylesheet" href="{{asset('plugins/bootstrap/css/bootstrap.min.css')}}">--}}<!-- Google Font: Poppins --> <link href="https://fonts.googleapis.com/css?family=Poppins:300,400,400i,700" rel="stylesheet"> <link rel="stylesheet" href="{{asset('css/custom.css')}}"> @if(app()->getLocale()=='ar') <link rel="stylesheet" href="{{asset('dist/css/rtl/custom.css')}}"> @endif</head><body class="hold-transition login-page"><div class="login-box"> <div class="login-logo"> <a href="{{ url('/') }}"><img src="{{$app\_logo}}" alt="{{setting('app\_name')}}"></a> </div> <div class="card"> @yield('register\_bar') </div> <!-- /.login-logo --> <div class="card"> @yield('content') </div></div><!-- /.login-box --><!-- script --><script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj" crossorigin="anonymous"></script><script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js" integrity="sha384-9/reFTGAW83EW2RDu2S0VKaIzap3H66lZH81PoYlFhbGU+6BZp6G7niu735Sk7lN" crossorigin="anonymous"></script><script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js" integrity="sha384-B4gt1jrGC7Jh4AgTPSdUtOBvfO8shuf57BaghqFfPlYxofvL8/KUEfYiJOMMV+rV" crossorigin="anonymous"></script><!-- jQuery --><script src="{{asset('plugins/jquery/jquery.min.js')}}"></script><!-- Bootstrap 4 --><script src="{{asset('plugins/bootstrap/js/bootstrap.bundle.min.js')}}"></script><!-- iCheck --><script src="{{asset('plugins/iCheck/icheck.min.js')}}"></script><script> $(function () { $('.icheck input').iCheck({ checkboxClass: 'icheckbox\_flat-blue', radioClass: 'iradio\_flat-blue', increaseArea: '20%' // optional }) })</script>@yield('script')</body></html>

**default blade page**

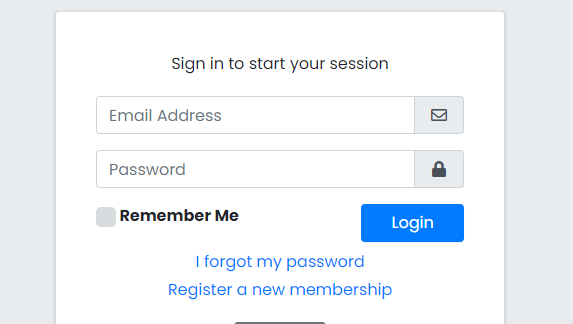
@extends('layouts.auth.default')@section('content') <div class="card-body login-card-body"> <p class="login-box-msg">{{\_\_('auth.register\_new\_member')}}</p> <form action="{{ url('/register') }}" method="post"> {!! csrf\_field() !!} <div class="input-group mb-3"> <input value="{{ old('name') }}" type="name" class="form-control {{ $errors->has('name') ? ' is-invalid' : '' }}" name="name" placeholder="{{\_\_('auth.name')}}" aria-label="{{\_\_('auth.name')}}"> <div class="input-group-append"> <span class="input-group-text"><i class="fa fa-user"></i></span> </div> @if ($errors->has('name')) <div class="invalid-feedback"> {{ $errors->first('name') }} </div> @endif </div> <div class="input-group mb-3"> <input value="{{ old('email') }}" type="email" class="form-control {{ $errors->has('email') ? ' is-invalid' : '' }}" name="email" placeholder="{{\_\_('auth.email')}}" aria-label="{{\_\_('auth.email')}}"> <div class="input-group-append"> <span class="input-group-text"><i class="fa fa-envelope"></i></span> </div> @if ($errors->has('email')) <div class="invalid-feedback"> {{ $errors->first('email') }} </div> @endif </div> <div class="input-group mb-3"> <input value="{{ old('password') }}" type="password" class="form-control {{ $errors->has('password') ? ' is-invalid' : '' }}" name="password" placeholder="{{\_\_('auth.password')}}" aria-label="{{\_\_('auth.password')}}"> <div class="input-group-append"> <span class="input-group-text"><i class="fa fa-lock"></i></span> </div> @if ($errors->has('password')) <div class="invalid-feedback"> {{ $errors->first('password') }} </div> @endif </div> <div class="input-group mb-3"> <input value="{{ old('password\_confirmation') }}" type="password" class="form-control {{ $errors->has('password\_confirmation') ? ' is-invalid' : '' }}" name="password\_confirmation" placeholder="{{\_\_('auth.password\_confirmation')}}" aria-label="{{\_\_('auth.password\_confirmation')}}"> <div class="input-group-append"> <span class="input-group-text"><i class="fa fa-lock"></i></span> </div> @if ($errors->has('password\_confirmation')) <div class="invalid-feedback"> {{ $errors->first('password\_confirmation') }} </div> @endif </div> <div class="row mb-2"> <div class="col-8"> <div class="checkbox icheck"> <label > <input type="checkbox" name="remember"> <a href="{{ url('/terms') }}" >{{\_\_('auth.agree')}} </a></label> </div> </div> <!-- /.col --> <div class="col-4"> <button type="submit" class="btn btn-primary btn-block">{{\_\_('auth.register')}}</button> </div> <!-- /.col --> </div> </form> @if(setting('enable\_facebook',false) || setting('enable\_google',false) || setting('enable\_twitter',false)) <div class="social-auth-links text-center mb-3"> <p style="text-transform: uppercase">- {{\_\_('lang.or')}} -</p> @if(setting('enable\_facebook',false)) <a href="{{url('login/facebook')}}" class="btn btn-block btn-facebook"> <i class="fa fa-facebook mr-2"></i> {{\_\_('auth.login\_facebook')}} </a> @endif @if(setting('enable\_google',false)) <a href="{{url('login/google')}}" class="btn btn-block btn-google"> <i class="fa fa-google-plus mr-2"></i> {{\_\_('auth.login\_google')}} </a> @endif @if(setting('enable\_twitter',false)) <a href="{{url('login/twitter')}}" class="btn btn-block btn-twitter"> <i class="fa fa-twitter mr-2"></i> {{\_\_('auth.login\_twitter')}} </a> @endif </div> <!-- /.social-auth-links --> @endif <p class="mb-1 text-center"> <a href="{{ url('/login') }}">{{\_\_('auth.already\_member')}}</a> </p> </div> <!-- /.login-card-body -->@endsection

**register page**

****

@extends('layouts.auth.default')  
@section('content')  
 <div class="card-body login-card-body">  
 <div class="card-body login-card-body">  
 <p class="login-box-msg">{{\_\_('auth.login\_title')}}</p>  
  
 <form action="{{ url('/login') }}" method="post">  
 {!! csrf\_field() !!}  
  
 <div class="input-group mb-3">  
 <input value="{{ old('email') }}" type="email" class="form-control {{ $errors->has('email') ? ' is-invalid' : '' }}" name="email" placeholder="{{ \_\_('auth.email') }}" aria-label="{{ \_\_('auth.email') }}">  
 <div class="input-group-append">  
 <span class="input-group-text"><i class="fa fa-envelope"></i></span>  
 </div>  
 @if ($errors->has('email'))  
 <div class="invalid-feedback">  
 {{ $errors->first('email') }}  
 </div>  
 @endif  
 </div>  
  
 <div class="input-group mb-3">  
 <input value="{{ old('password') }}" type="password" class="form-control {{ $errors->has('password') ? ' is-invalid' : '' }}" name="password" placeholder="{{\_\_('auth.password')}}" aria-label="{{\_\_('auth.password')}}">  
 <div class="input-group-append">  
 <span class="input-group-text"><i class="fa fa-lock"></i></span>  
 </div>  
 @if ($errors->has('password'))  
 <div class="invalid-feedback">  
 {{ $errors->first('password') }}  
 </div>  
 @endif  
 </div>  
  
 <div class="row mb-2">  
 <div class="col-8">  
 <div class="checkbox icheck">  
 <label> <input type="checkbox" name="remember"> {{\_\_('auth.remember\_me')}}  
 </label>  
 </div>  
 </div>  
 <!-- /.col -->  
 <div class="col-4">  
 <button type="submit" class="btn btn-primary btn-block">{{\_\_('auth.login')}}</button>  
 </div>  
 <!-- /.col -->  
 </div>  
 </form>  
  
 <p class="mb-1 text-center">  
 <a href="{{ url('/password/reset') }}">{{\_\_('auth.forgot\_password')}}</a>  
 </p>  
 <p class="mb-0 text-center">  
 <a href="{{ url('/register') }}" class="text-center">{{\_\_('auth.register\_new\_member')}}</a>  
 </p>  
 <div class="col-4" style="margin-left: auto; margin-right: auto; padding-top: 20px">  
 <button id="switchLanguage" type="submit" class="btn btn-secondary btn-block">{{\_\_('auth.login.lang')}}</button>  
 </div>  
 </div>  
 </div>  
@endsection

login page



# **References:**

1. Pollack, B., 2009. Mobile automated-hand car wash. U.S. Patent Application 12/150,339.
2. Kim, S. and Park, E., 2014. Integrated carwash client service management system with real-time work scheduling process and carwash order and reservation for a car parking facility-based hand carwash. U.S. Patent Application 14/306,232.
3. Z. Jiuru, "Comparative Study on the Service Pricing Strategy of Car Wash Industry," 2019 Chinese Control And Decision Conference (CCDC), Nanchang, China, 2019, pp. 6098-6103, doi: 10.1109/CCDC.2019.8833180.
4. Neuhofer, B., Buhalis, D., Ladkin, A., “Smart technologies for personalized experiences: a case study in the hospitality domain”, Journal Electronic Markets, vol. 25, Issue 3, pp. 243-254, September 2015. Springer.
5. D. Fisher, “Websites are being replaced,” ABA Banking Journal, pp. 21-25, January 2012 Pollack, B., 2009. Mobile.
6. Jantunen E., · Giordamlis C., · Adgar · A. and Emmanouilidis C., “Mobile devices and services maintenance”, in E-maintenance, Holmberg, K., Adgar, A., Arnaiz, A., Jantunen, E., Mascolo and J.,Mekid, S. Eds. Springer, 2010, pp. 227-246.
7. M. Esteves and A. Pereira, "Y.S.Y.D. - You Stay You Demand: User-centered design approach for mobile hospitality application," 2015 International Conference on Interactive Mobile Communication Technologies and Learning (IMCL), Thessaloniki, 2015, pp. 318-322, doi: 10.1109/IMCTL.2015.7359611.
8. V. N. Inukollu, D. D. Keshamoni, T. Kang and M. Inukollu. Factors Influencing Quality Of Mobile Apps: Role Of Mobile App Development Life Cycle. International Journal of Software Engineering & Applications (IJSEA). Available: https://arxiv.org/pdf/1410.4537.pdf, 2014. 5(5): 15-34.
9. Phongtraychack and D. Dolgaya. Evolution of Mobile Applications. MATEC Web of Conferences 155, 01027. Available: https://doi.org/10.1051/matecconf/2018155010272018), 2018.
10. Fulenwider, G., “Web site creation for mobile devices”, Journal of Computing Sciences in Colleges, Vol. 28, Issue 5, pp. 132-133, May 2013.
11. Atawneh, S., Al-Kasasbeh, B. and Ben Rshed, M. (2019) ‘Android-Based Mobile Application for Door-to-Door Product Delivery’, International Journal of Interactive Mobile Technologies, 13(3), pp. 125–142. doi: 10.3991/ijim.v12i7.9598.
12. N. Anwar, M. A. M. Rizal, H. A. Mustamum, K. M. Taib, A. A. Razak and Z. Nordin, "Mobile Application Development: A Preliminary Study," 2020 International Conference on Information Management and Technology (ICIMTech), Bandung, Indonesia, 2020, pp. 951-956, doi: 10.1109/ICIMTech50083.2020.9211289.
13. V. N. Inukollu, D. D. Keshamoni, T. Kang and M. Inukollu. Factors Influencing Quality Of Mobile Apps: Role Of Mobile App Development Life Cycle. International Journal of Software Engineering & Applications (IJSEA). Available: https://arxiv.org/pdf/1410.4537.pdf, 2014. 5(5): 15-34.
14. OBISI, C. and Gbadamosi, A. A. O. (2017) ‘Impact of Mobile Applications on Workforce Productivity in Road Transport Industry in Nigeria’, BVIMSR Journal of Management Research, 9(2), pp. 116–129. Available at: http://search.ebscohost.com.sdl.idm.oclc.org/login.aspx?direct=true&db=bsu&AN=126549258&site=eds-live (Accessed: 30 October 2020).
15. (GaStat), G. A. f. S., 2016. demographic research, Saudi Arabia: General Authority for Statistics (GaStat).
16. Flutter documentation. Retrieved October 20, 2020, from <https://flutter.dev/docs>
17. Maks Surguy (July 27, 2013). "History of Laravel PHP framework, Eloquence emerging". maxoffsky.com. Retrieved October 20, 2020.
18. غسل سيارتك مع غسيل. (n.d.). Retrieved October 15, 2020, from http://www.ghaseel.com/
19. Connecting Mobile Car Wash. (n.d.). Retrieved October 15, 2020, from https://sayarapp.com/