

Chapter 5

Web and Mobile Applications

5.1 Introduction

In this chapter, we will discuss how to operate the web application that is primarily linked to the project and contains a database for all registered drivers with their respective information, including their health status. We also have a mobile application that provides services to anyone who registers in it. The mobile app also displays driver data and the location where any car issues may occur.

5.2 Web Application

The **Figure 5.1** shows us the home page of the web page. The website contains a dashboard divided into categories, each category has a specialized admin who is allowed to log in to his page only by username and password.

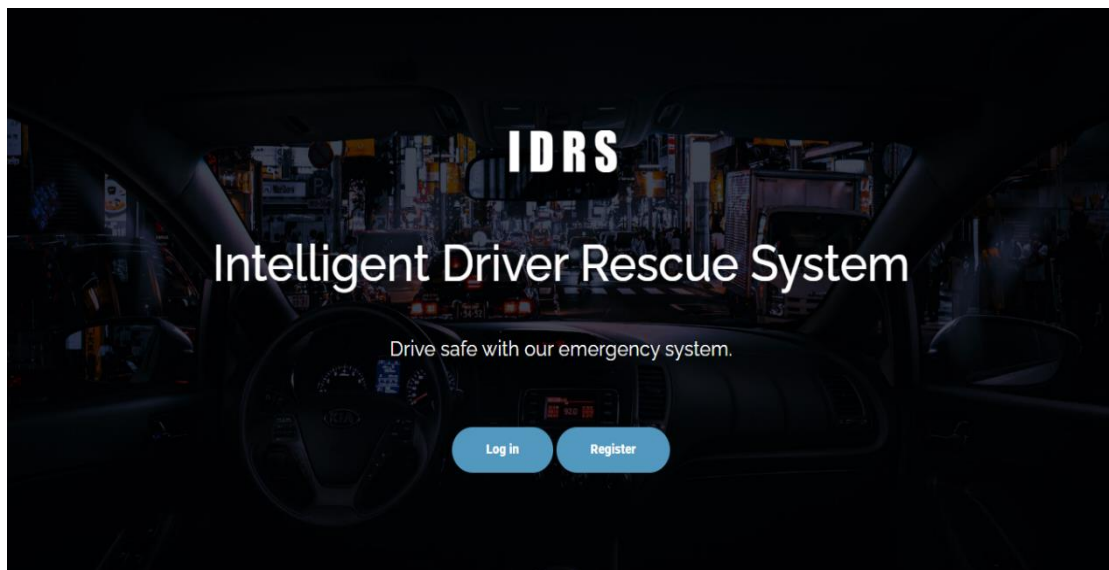


Fig.5.1: Main Page

The first admin function is to add the driver's data such as name, password, National-id, Health diseases, etc... and sends it to the database, the second admin is responsible for following up on cars that have a problem at present and deciding which driver has the most dangerous state depends on the data came from smartwatch and database and send the car id

to the third admin, The third admin's job is to control the car remotely through the main office until it reaches a safe place In the case that the driver lost consciousness. Once the user enters the website, the dashboard shown in **Figure 5.2** will appear on screen.

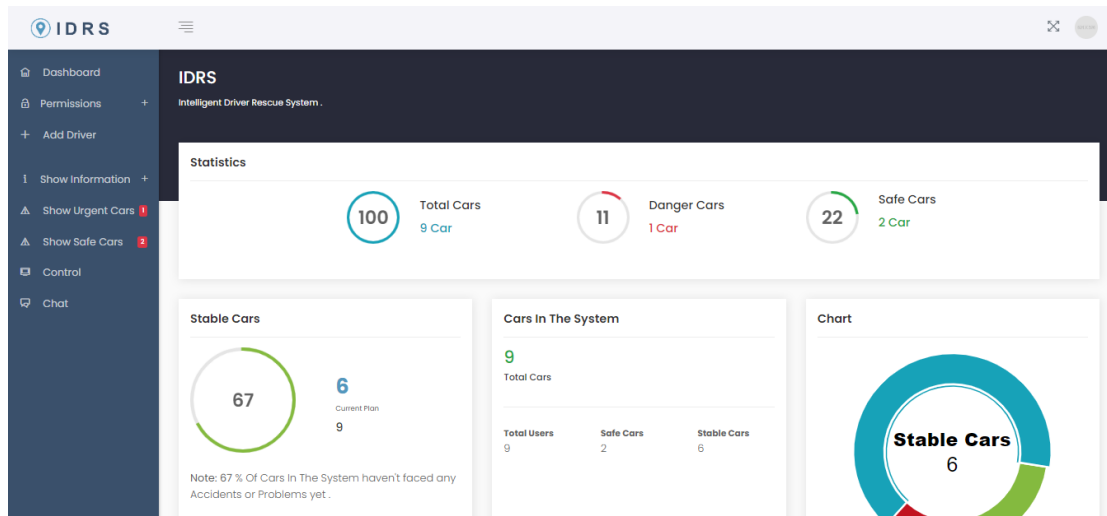


Fig.5.2: Dashboard

It includes the number of cars registered in the system and whether they were in danger or safe. The sidebar contains the tabs that allow the user to control the dashboard. The permission tab shown in **Figure 5.3** consists of two pages the user menu and user permission. The user menu page contains all the users of the website managers, users, owners, and control users.

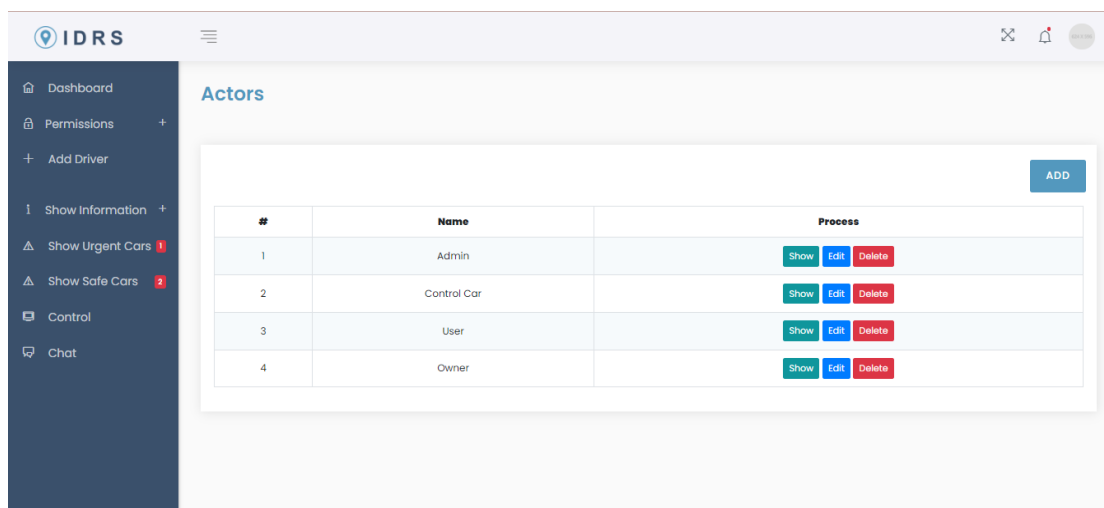


Fig.5.3: Permission Page

The user permission page contains the database of all the website users; names, email, type and status. **Figure 5.4** shows the list of users that allows us to add, edit or delete users

#	User name	Email	User Status	User Type	Process
1	Ahmed Nader	ahmednader@gmail.com	active	Owner	Edit Delete
2	ahmed Hamdy	ahmedhamdy@gmail.com	active	Admin	Edit Delete

Fig.5.4: User List

Add driver tab consists of four pages to input data. The first page contains the driver's information shown in **Figure 5.5**; name, e-mail, phone number and the emergency contact of the driver.

Driver Informations				
Name	Email	Gender	Phone Number	Emergency Number
Ahmed	ahmednnnn@gmail.com	male	1234567892	201277807195
City	Address	Birth Date	Nationality	National-ID
cairo	mahalla	2000-03-17	egyptian	1234568878955432

Fig.5.5: Driver Information

Based on the provided information, it seems that the driver-related information is displayed in an icon or section labeled as “Driver”. The second page shown in **Figure 5.6** contains the car information; car number and car model.

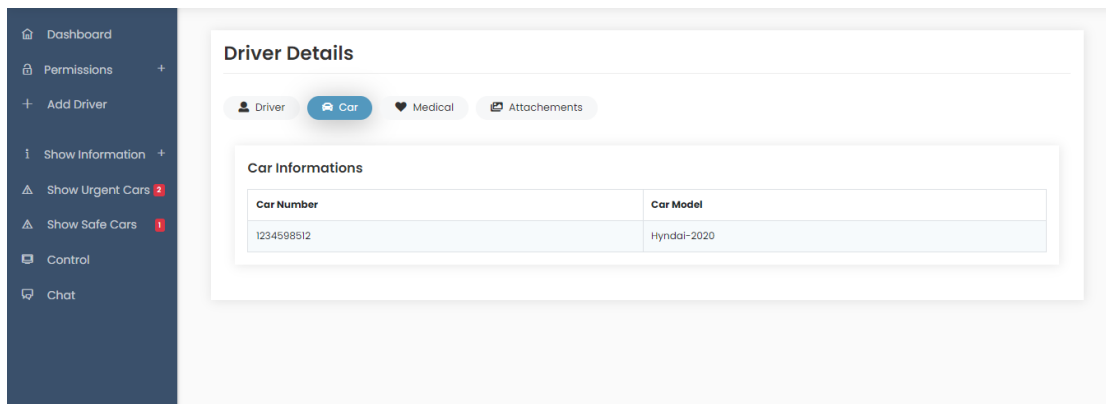


Fig.5.6: Car Information

Based on the provided information, it seems that the car-related information is displayed in an icon or section labeled as “Car”. The third page shown in **Figure 5.7** contains the medical information of the driver; blood pressure, diabetes and blood type.

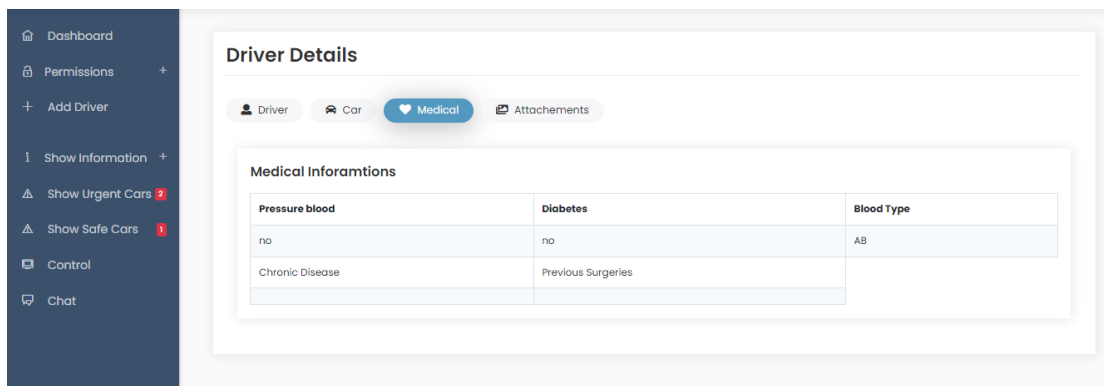


Fig.5.7: Driver’s Medical Information

Based on the provided information, it seems that the medical-related information is displayed in an icon or section labeled as “Medical”. The fourth page shown in **Figure 5.8** contains the attachments. The user uploads his personal photo, car license and driver license.

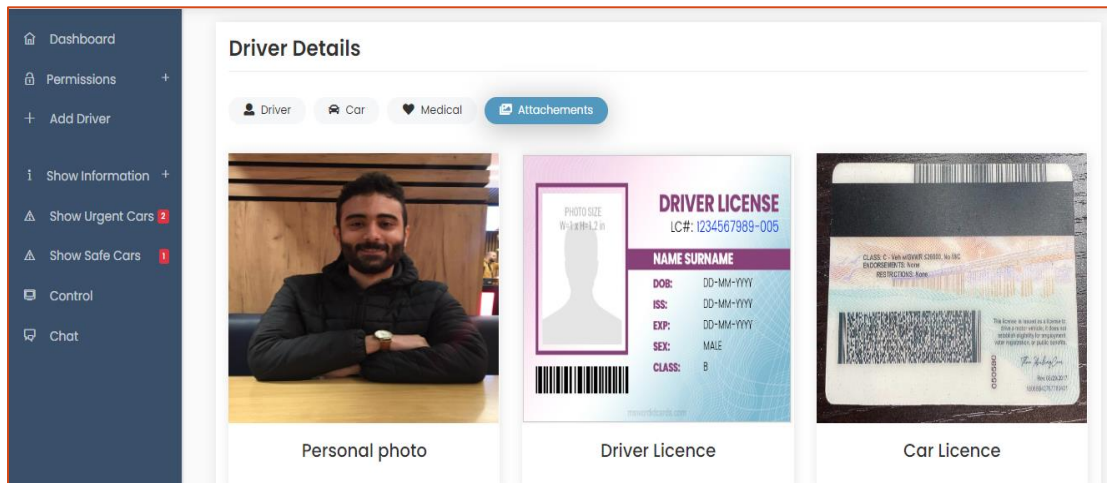


Fig.5.8: Driver Attachments

Based on the provided information, it seems that the attachments-related information is displayed in an icon or section labeled as “Attachment”. After registering, a message shown in **Figure 5.9** is sent to the user's g-mail to inform the user the he has been registered on the system.

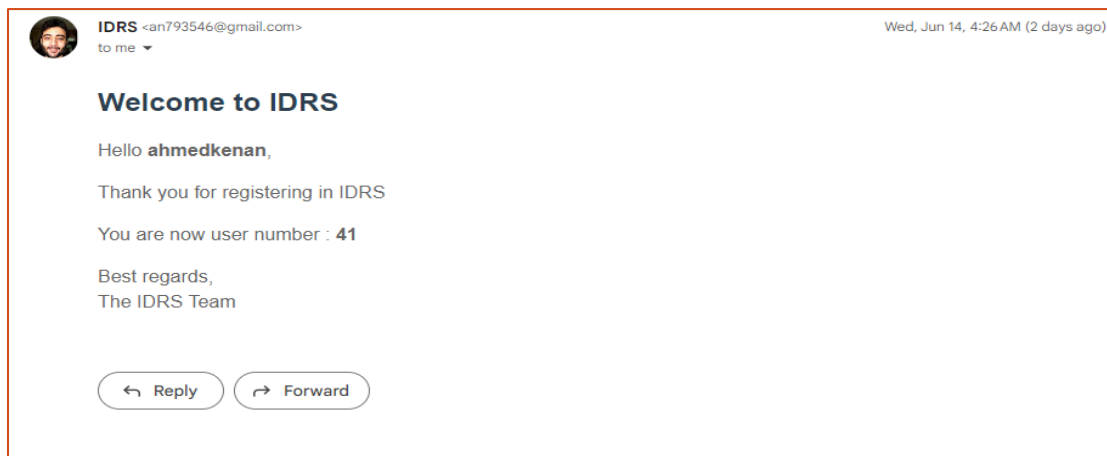


Fig.5.9: Welcome Message

The information tab shown in **Figure 5.10** contains the fields with driver's information that required to fill. By clicking on 'more info' button; four pages appear holding driver, car, medical information and attachments.

The screenshot shows the 'Driver Informations' registration page. On the left is a dark blue sidebar with the 'IDRS' logo and a menu containing: Dashboard, Permissions, Add Driver, Show Information (with a plus icon), Show Urgent Cars (with a red notification badge '1'), Show Safe Cars (with a red notification badge '2'), Control, and Chat. The main content area is titled 'Driver Informations' and contains several form fields: 'Name' and 'Email' (text inputs), 'Gender' (a dropdown menu), 'Phone Number', 'City', and 'Date Of Birth' (text inputs), 'Address' (a text input), 'Nationality' (a dropdown menu), 'National-ID' (a text input), and 'Emergency Number' (a text input). A blue 'NEXT >' button is located at the bottom right of the form.

Fig.5.10: Driver Information Registration Page

The above shows us which fields are filled with driver's information. The information tab shown in **Figure 5.11** contains the fields with car's information that required to fill.

The screenshot shows the 'Car Informations' registration page. The sidebar is identical to the previous figure. The main content area is titled 'Car Informations' and contains three form fields: 'Car license plate number' (a text input), 'Car Model' (a text input), and 'Is There someone else Drive the car ?' (a dropdown menu with the placeholder text '-- Please select --'). At the bottom of the form are two blue buttons: '< PREVIOUS' on the left and 'NEXT >' on the right.

Fig.5.11: Car Information Registration Page

So fields are filled with car information. The information tab shown in **Figure 5.12** contains the fields with Medical's information that required to fill.

The screenshot shows the 'Medical Informations' registration page. On the left is a dark blue sidebar with navigation links: Dashboard, Permissions, Add Driver, Show Information, Show Urgent Cars (with a red notification badge '1'), Show Safe Cars (with a red notification badge '2'), Control, and Chat. The main content area has a light blue header with a heart icon and the title 'Medical Informations'. Below the header are several form fields: 'Do you have a pressure problem?' with radio buttons for 'Yes' and 'No'; 'Do you have Diabetes?' with radio buttons for 'Yes' and 'No'; 'Blood Type' with a dropdown menu labeled 'Choose Your blood Type'; 'Do you have any Chronic diseases?' with radio buttons for 'Yes' and 'No'; and 'Have you had any surgeries before?' with radio buttons for 'Yes' and 'No'. At the bottom of the form are two buttons: 'PREVIOUS' on the left and 'NEXT' on the right.

Fig.5.12: Health Status Registration Page

So fields are filled with medical information. The information tab shown in **Figure 5.13** contains the fields with Medical's information that required to fill.

The screenshot shows the 'Attachements' registration page. The sidebar is identical to the previous page. The main content area has a light blue header with a document icon and the title 'Attachements'. Below the header are three file upload sections: 'Personal Photo:', 'Driver License:', and 'Car License:'. Each section has a 'Choose file' text input and a 'Browse' button. At the bottom of the form are two buttons: 'PREVIOUS' on the left and 'SUBMIT' on the right.

Fig.5.13: User Information Registration Page

So fields are filled with user information. Urgent cars tab shown in **Figure 5.14** includes the cars that face problems at the mean time. The location of the car is sent to the database and received by the website.

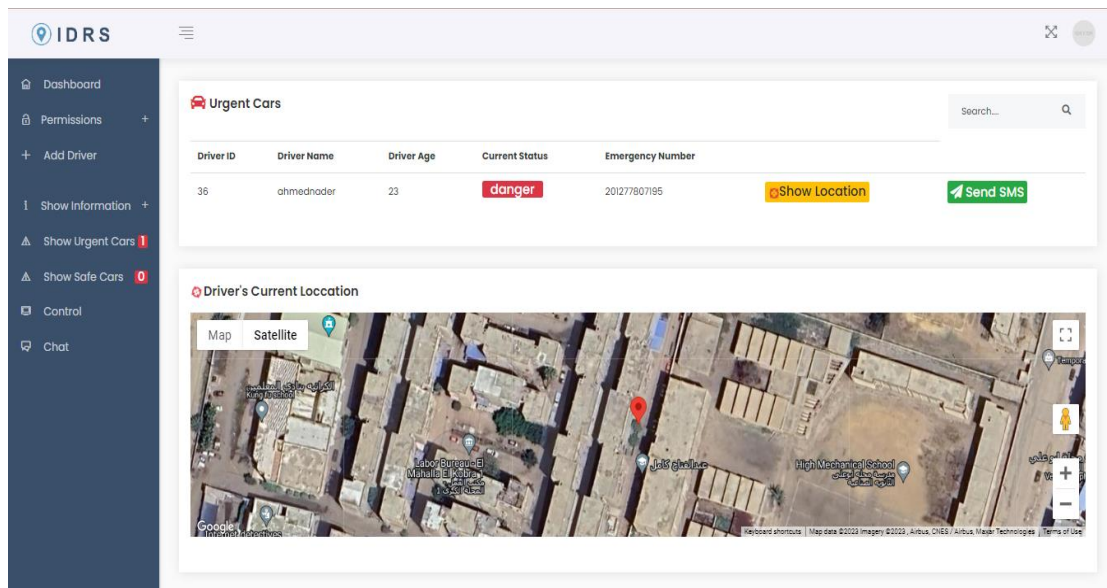


Fig.5.14: Car Location

By clicking on 'show location' button; a google map pops up on screen to show the current location of the car. shows the condition of the car in case of a problem as well as the location of the accident. When clicking on 'send SMS' button; a message shown in **Figure 5.15** is sent to the emergency contact of the driver to inform them of his medical condition and location.

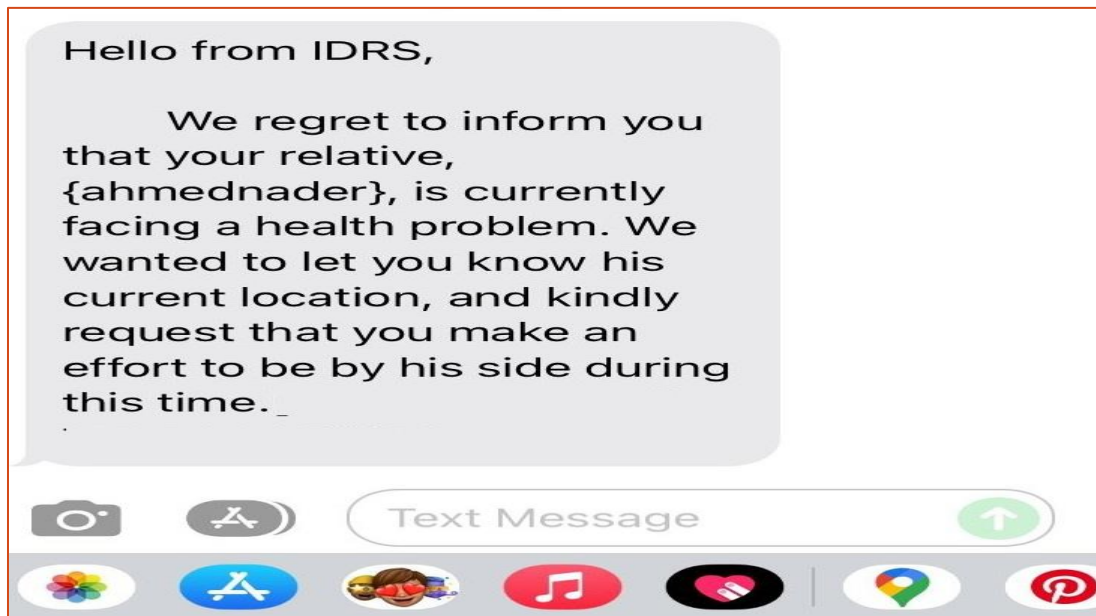


Fig.5.15: Distress Message

Safe cars shown in **Figure 5.16** shows the cars that has been dealt with and is safe now to function properly.

Driver ID	Driver Name	Driver Age	Current Status
36	ahmednader	23	done

Fig.5.16: Safe Car

The status of the car is transformed to done. shows the condition of the car in safe position. Chat page shown in **Figure 5.17** is accustomed to respond to the system user's inquiries and questions by qualified assistants to help user better understand and use the system with ease.

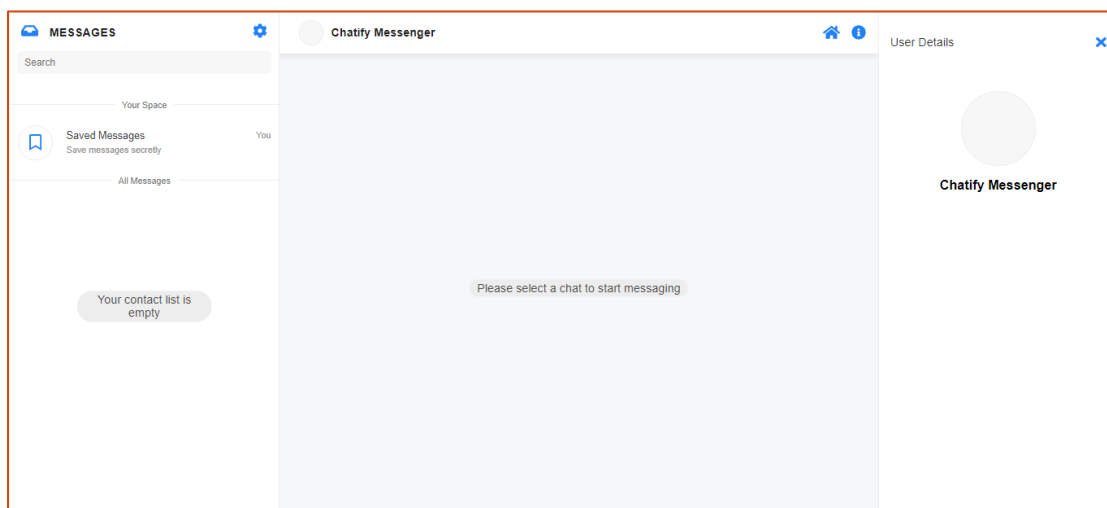


Fig.5.17: Chat Page

5.3 Mobile Application

We chose Flutter to build our mobile app, we harnessed the power of a single codebase, rapid development with hot reload, visually stunning UI widgets, high-performance rendering, seamless integration with native features, a modern and maintainable programming language, a thriving community for support, efficient testing and debugging tools, cost savings in development, and the reliability of a framework backed by Google.

5.3.1 Reasons of Using Flutter

There are several reasons why developers choose to use Flutter for building mobile applications. Here are some key reasons:

- Flutter offers a single codebase approach, allowing us to develop a mobile app that runs seamlessly on both iOS and Android platforms, saving time and resources.
- With Flutter's hot reload feature, we can instantly see changes in the app's UI and functionality, enabling rapid development and iteration.
- The extensive collection of customizable UI widgets provided by Flutter allows us to create visually stunning and consistent user interfaces across different platforms.
- Flutter's high-performance rendering engine and direct communication with the platform enable smooth animations and a responsive user experience.
- By utilizing Flutter's rich set of plugins, we can easily access and integrate native device features and APIs, enhancing our app's capabilities.
- The Dart programming language, used in Flutter, offers a familiar syntax and modern features like reactive programming and asynchronous programming, enabling us to write clean and maintainable code.
- The active Flutter community provides a wealth of resources, tutorials, and open-source packages, making it easier to overcome challenges and accelerate development.
- Flutter's testing and debugging tools simplify the process of identifying and fixing issues, ensuring a robust and stable app.
- Building a mobile app with Flutter offers cost savings by reducing the need for separate development teams or resources for iOS and Android platforms.

- Flutter, backed by Google, ensures long-term support, frequent updates, and integration with other Google services, providing a reliable and future-proof framework for mobile app development.

5.3.2 Flutter lifecycle

Flutter is an open-source UI (User Interface) toolkit developed by Google. It allows developers to create high-performance, cross-platform applications for mobile, web, and desktop platforms. The lifecycle of a Flutter application as shown in **Figure 5.18** refers to the sequence of events and states that an application goes through from its initialization to its termination. Here's the lifecycle of a Flutter application:

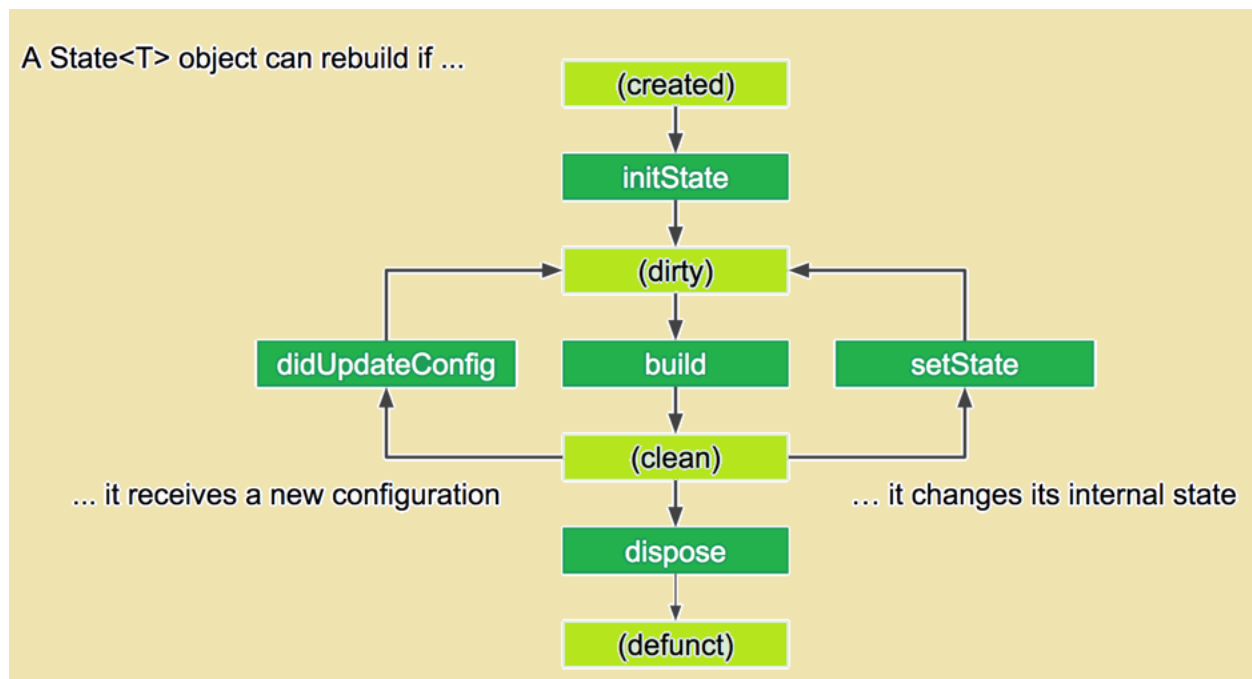


Fig.5.18: Flutter Lifecycle Diagram

Here's a detailed explanation of the lifecycle of a Flutter application:

- 1. Initialization:** When a Flutter application starts, the main function is called, which typically initializes the app and runs the app's main event loop. The framework initializes various essential components, such as the rendering engine, the Dart Virtual Machine (VM), and the platform-specific plugins.

2. **runApp():** The runApp() function is called, which typically creates the root widget of the application and starts the Flutter framework.
3. The root widget represents the entire application and is usually an instance of a subclass of StatelessWidget or StatefulWidget.
4. **Widget Initialization:** When the root widget is created, its createState() method is called for StatefulWidget and the corresponding State object is created. The State object is responsible for maintaining the mutable state of the widget.
5. **initState():** After the State object is created, the initState() method is called. This method is used to initialize the state of the widget, set up subscriptions, and perform any necessary one-time initialization tasks.
6. **Widget Building:** The framework calls the build() method of the widget whenever it needs to rebuild the widget's UI. The build() method returns a widget hierarchy representing the current state of the widget. The framework then performs a diffing algorithm to determine the differences between the new widget hierarchy and the previous one and efficiently updates the UI accordingly.
7. **User Interaction:** As the user interacts with the application, events such as button presses, gestures, or text input are captured by the framework and propagated to the appropriate widgets. Widget-specific event handling methods like onTap(), onChanged(), or onSubmit() can be used to respond to user actions.
8. **State Updates:** When a user action or any other event triggers a state change in the widget, the setState() method is called. The setState() method updates the widget's internal state and marks it as dirty, signaling the framework that the widget needs to be rebuilt.
9. **Rebuilding Widgets:** When a widget is marked as dirty, the framework schedules a rebuild of the widget and its descendant widgets. The build() method of the widget is called again to create a new widget hierarchy reflecting the updated state. The framework then updates the UI by applying the necessary changes based on the widget diffing algorithm.

10. App Lifecycle Events: During the application's lifecycle, various system events can occur, such as app initialization, app suspension, app resumption, and app termination. Flutter provides lifecycle callbacks that allow developers to handle these events and perform appropriate actions, such as `didChangeAppLifecycleState()`.

11. Termination: When the application is about to terminate, `dispose()` method of the `State` object is called. The `dispose()` method allows the widget to release any resources, subscriptions, or connections it acquired during its lifetime.

5.2.3 Working Mechanisms

First of all, when you open the mobile application shown in **Figure 5.19**, the home page appears if this is your first time accessing the application or if you don't have an email, you should register.



Fig.5.19: User Informaion

Otherwise, if you already have an email, you can log in. After logging in as shown in **Figure 5.20**, your entered data and a profile picture will be displayed on the screen.



Fig.5.20: User Informaion

The page shown in **Figure 5.21** contains the fields with driver's information that required to fill.

The image shows a mobile application screen titled 'Personal Details'. At the top, there is a blue header with the text 'Personal Details'. Below this, there is a list of input fields for personal information: NAME, GENDER, PHONE NUMBER, CITY, DATE OF BIRTH, ADDRESS, and NATIONALITY. Each field is represented by a white rounded rectangle with the label text inside. The background of the screen is a blue abstract design with a car icon.

Fig.5.21: Personal Details

The above shows fields are required to be filled with driver's information. The page shown in **Figure 5.22** contains the fields with health information that required to fill.

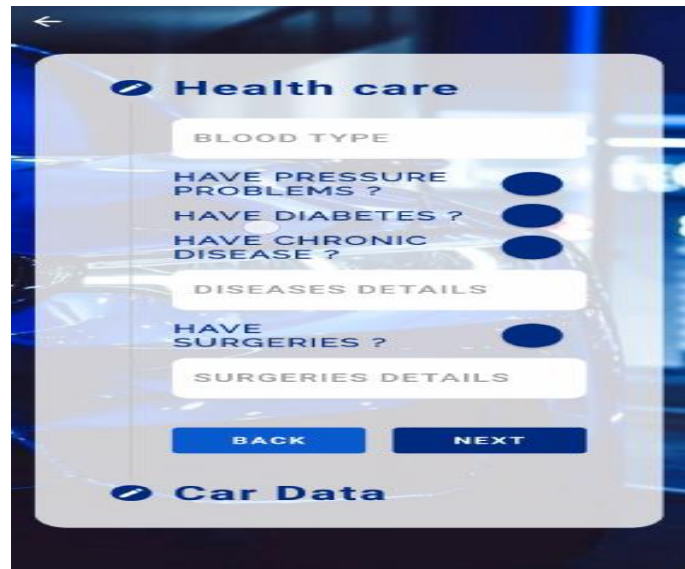
A mobile application screen titled "Health care" with a back arrow at the top left. The form contains several input fields: "BLOOD TYPE", "DISEASES DETAILS", and "SURGERIES DETAILS". Below these are three questions with radio button options: "HAVE PRESSURE PROBLEMS?", "HAVE DIABETES?", and "HAVE CHRONIC DISEASE?". At the bottom, there is a "HAVE SURGERIES?" question with a radio button, and another "SURGERIES DETAILS" field. Two blue buttons, "BACK" and "NEXT", are positioned at the bottom of the form. Below the form, there is a section header "Car Data" with a checkmark icon.

Fig.5.22: Health Care

The above shows fields are required to be filled with health information. The page shown in **Figure 5.23** contains the fields with car information that required to fill.

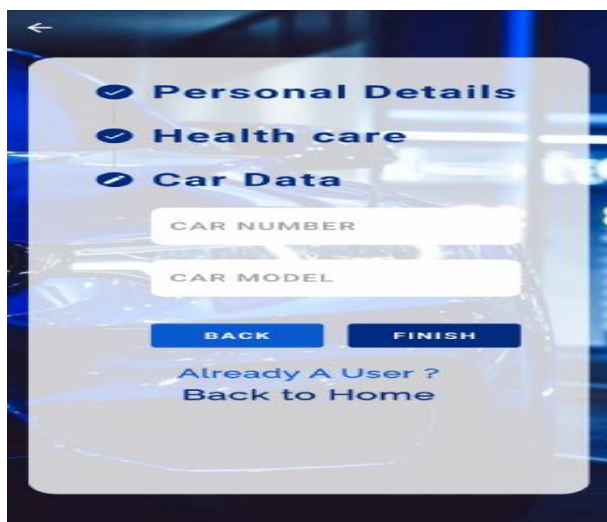
A mobile application screen titled "Car Data" with a back arrow at the top left. The form contains two input fields: "CAR NUMBER" and "CAR MODEL". Below these are two blue buttons, "BACK" and "FINISH". At the bottom, there is a link that says "Already A User ? Back to Home". Above the form, there are three section headers with checkmark icons: "Personal Details", "Health care", and "Car Data".

Fig.5.23: Login Options

A mobile application screen titled "User Info" with a back arrow at the top left. It displays user information: a profile picture, the name "ahmed nader", and the email "ahmed.nader@idrs.com". Below this, there are links for "Car Status" and "View Profile". A section titled "I Why IDRS?" is partially visible. A "Danger Alert" dialog box is overlaid on the screen, asking "Allow IDRS to access this device's location?". It shows two map icons labeled "Precise" and "Approximate". At the bottom of the dialog, there are three options: "While using the app", "Only this time", and "Don't allow".

Fig.5.24: Car location

The above shows fields are required to be filled with car information. The page shown in **Figure 5.24** shows the location of car.

Summary

In this chapter, we covered the different parts of the software, whether it's a web application or a mobile app. We discussed registration methods and the pages that appear, along with the information displayed on each page. We also discussed the general components of the application and how the roles of the registered users can vary, such as regular users, owners, administrators, and more. We explained how to display driver information, including their health status, car data, and associated documents like personal photos, driver's licenses, or vehicle licenses. Additionally, we mentioned a page that shows both critical vehicles and those that have been repaired.