

OCTANE MOBILE APP

INTERDISCIPLINARY PROJECT

Submitted in partial fulfillment of the
requirements for the award of
Bachelor of Technology degree in Information Technology

by

KUMARA VADIVEL P (41120099)



DEPARTMENT OF INFORMATION TECHNOLOGY

SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Category-1 University by UGC

Accredited with Grade “A++” by NAAC

12B Status by UGC |Approved by AICTE

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

MAY - 2024



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

[DEEMED TO BE UNIVERSITY]

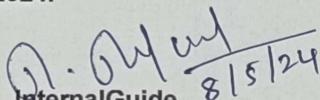
Accredited "A++" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

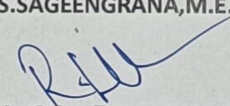
DEPARTMENT OF INFORMATION TECHNOLOGY

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **KUMARA VADIVEL P (41120099)** who carried out the project entitled "OCTANE MOBILE APP" under my supervision from January 2024 to May 2024.

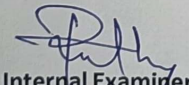

Internal Guide 8/5/24

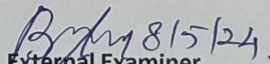
Ms.S.SAGEENGRANA,M.E.,


Head of the Department

Dr.R. SUBHASHINI, M.E., Ph.D.,

Submitted for Viva voce Examination held on 02/05/2024


Internal Examiner

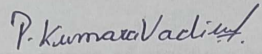

External Examiner 8/5/24

DECLARATION

I **Kumara Vadivel P** hereby declare that the Project Report entitled "OCTANE MOBILE APP" done by me under the guidance of Ms.S.SAGEENGRANA,M.E., is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in Information Technology.

DATE: 08/05/2024

PLACE: Chennai


SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D., Dean**, School of Computing **Dr.R.Subhashini M.E.,Ph.D.**,Head of the Department of Information Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Ms.S.Sageengrana, M.E., Ph.D.**, for her valuable guidance, suggestions and constant encouragement paved wayfor the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

CERTIFICATE OF COMPLETION



HACKWIT TECHNOLOGIES PVT LTD
Redefine by Technologies

149, 1C, 2nd Floor, Mount Poonamallee Rd,
Ramapuram, Chennai - 600089.
admin@hackwittechnologies.com
www.hackwittechnologies.com

10.02.2024

CERTIFICATE FOR INTERNSHIP

TO WHOMSOEVER IT MAY CONCERN

This is to Certify that **Kumara Vadivel P (Reg No: 41120099)** student of Sathyabama Institute of Science and Technology, B. Tech Information Technology, has successfully completed the Internship at Hackwit Technologies from 22nd January to 30th January 2024.

During the internship, **Kumara Vadivel P** has closely worked as a part of the **Flutter** by carrying out the project entitled "**Octane Cars**".

He demonstrated good developing and learning skill with self-motivated attitude to learn new things. His contribution to implement the Flutter was good. We wish him all the best for his future endeavors.

Best Regards,

A handwritten signature in black ink, appearing to be 'Ravikumar S', written over a faint circular stamp.

Ravikumar S
CEO & Managing Director
Hackwit Technologies Pvt Ltd

CERTIFICATE OF COMPLETION



ABSTRACT

OCTANE MOBILE APP is a dynamic and user-friendly mobile application developed using Flutter, designed facilitate the buying and selling of pre-owned cars. As the demand for reliable second-hand vehicles continues to rise, OCTANE MOBILE APP aims to bridge the gap between buyers and sellers in the automotive market by providing a seamless and intuitive platform. Key features of OCTANE MOBILE APP include comprehensive listings of pre-owned cars with detailed specifications and high-quality images, enabling users to browse through a diverse range of options conveniently. The application incorporates advanced search filters, allowing users to refine their searches based on specific criteria such as make, model, year, price range, and more, thereby streamlining the process of finding the perfect vehicle. For sellers, OCTANE MOBILE APP offers a hassle-free listing process, empowering them to create informative listings with ease and reach a wider audience of potential buyers. Through OCTANE's secure messaging system, buyers and sellers can communicate directly within the application, facilitating negotiations and ensuring transparency throughout the purchasing process. Moreover, OCTANE MOBILE APP prioritizes user trust and safety by implementing secure payment gateways and verification mechanisms, safeguarding transactions and mitigating the risk of fraudulent activities. Additionally, OCTANE MOBILE APP provides valuable resources and guidance to users, including car valuation tools, expert reviews, and tips for making informed purchasing decisions.

TABLE OF CONTENTS

Chapter No.	TITLE	Page No.
	ABSTRACT	vii
	LIST OF ABBREVIATIONS	x
	LIST OF FIGURES	xi
1	INTRODUCTION	1
1.1	Flutter	2
1.2	Dart	3
1.3	Firebase	4
1.4	Problem Statement	5
1.5	Objectives	6
2	METHODOLOGY	8
2.1	Initial Project Requirements	8
2.2	Proposed System Architecture	10
2.3	Flow Diagram	13
2.4	Modules	14
2.4.1	User Authentication and Authorization	14
2.4.2	Car Listings	15
2.4.3	Search and Filtering	16
2.4.4	Car Details	17
2.4.5	User Profile Management	18

	2.5	Development and Deployment Setup	19
3		RESULTS AND DISCUSSION	22
4		CONCLUSION AND FUTURE WORK	27
	4.1	Conclusion	27
	4.2	Future Work	28
		REFERENCES	29
		APPENDICES	30
	A.	Sample Code	30
	B.	Screen Shots	32

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
UI	- User Interface
UX	- User Experience
iOS	- iPhone Operating System
UAT	- User Acceptance Testing
IDE	- Integrated Development Environment

LIST OF FIGURES

Figure No.	Name of the Figure	Page No.
2.1	Proposed Application Architecture	10
2.2	Flow Diagram	13
2.3	Project Creation	19
2.4	Flutter Run	20
2.5	User Authentication	20
2.6	Data Model	21
3.1	Main Dart	22
3.2	Login Page	23
3.3	HomePage	23
3.4	Rolls RoycePage	24
3.5	BMWPage	24
3.6	ProfilePage	25
3.7	OCTANE App File	26

CHAPTER 1

INTRODUCTION

The automotive industry is experiencing a paradigm shift with the rising demand for second-hand cars. Factors such as affordability, depreciation rates, and environmental consciousness are driving consumers towards pre-owned vehicles. In response to this trend, OCTANE MOBILE APP emerges as a forward-thinking solution, offering a mobile application built on the Flutter framework. OCTANE is designed to streamline the process of buying and selling second-hand cars by providing a comprehensive and user-friendly platform.

OCTANE's key objective is to address the needs of both buyers and sellers in the pre-owned car market. For buyers, OCTANE MOBILE APP offers a vast selection of listings, complete with detailed specifications and high-quality images. Advanced search filters allow users to refine their searches based on specific criteria, enabling them to find the perfect vehicle quickly and efficiently. Additionally, OCTANE MOBILE APP prioritizes user trust and safety by implementing secure payment gateways and verification mechanisms, ensuring that transactions are conducted securely.

For sellers, OCTANE MOBILE APP provides a hassle-free listing process, allowing them to create informative listings and reach a wider audience of potential buyers. The platform's secure messaging system facilitates communication between buyers and sellers, fostering transparency and negotiation. OCTANE MOBILE APP also offers valuable resources such as car valuation tools and expert reviews, empowering users to make informed purchasing decisions.

1.1.FLUTTER

Flutter, a robust cross-platform UI toolkit developed by Google, is at the heart of , OCTANE MOBILE APP, shaping the application's foundation and user interface. The utilization of Flutter brings a paradigm shift to the traditional app development approach, offering a single codebase that transcends platform-specific limitations. This innovation allows , OCTANE MOBILE APP to deliver a visually captivating and responsive user interface, adeptly adjusting to both iOS and Android environments.

The efficiency of Flutter lies in its ability to streamline the development process, significantly reducing the complexities associated with managing separate codebases for different platforms. Developers benefit from a cohesive workflow, resulting in accelerated development cycles and enhanced productivity.

Moreover, Flutter's hot reload feature enables real-time code changes and instant visual feedback, facilitating quick iterations and adjustments.

One of the key advantages of Flutter is its capability to ensure a consistent and high-quality user experience across diverse devices. By leveraging Flutter's richset of customizable widgets, , OCTANE MOBILE APP achieves a cohesive design language and smooth interactions, irrespective of the underlying operating system. Users can seamlessly transition between devices without compromising on the application's performance or aesthetics.

1.2.DART

i)Unified Codebase:

Dart, a versatile programming language known for its client-optimized capabilities, to drive both frontend (Flutter) and backend development. By harnessing Dart's flexibility, OCTANE MOBILE APP maintains a unified codebase, ensuring consistency and facilitating seamless collaboration between frontend and backend teams. This unified approach streamlines development efforts.

ii)Asynchronous Programming:

Efficient handling of concurrent operations is paramount for the seamless functioning of OCTANE MOBILE APP. Dart's robust support for asynchronous programming is instrumental in achieving this goal. By leveraging asynchronous programming, OCTANE MOBILE APP can effortlessly manage a multitude of tasks concurrently, including user requests, database operations, and external API calls. This capability ensures that the application remains responsive and performs optimally, even when faced with complex and resource-intensive tasks. With Dart's asynchronous features like Futures and async/await syntax, OCTANE MOBILE APP can execute these operations non-blockingly, mitigating delays or disruptions that could compromise user experience.

iii)Scalability:

Scalability is paramount for OCTANE MOBILE APP to accommodate potential growth and evolving user demands. Leveraging Dart's scalability features, OCTANE MOBILE APP ensures readiness to handle increased traffic and growing user engagement seamlessly. As the platform's user base expands and the demand for second-hand cars rises, Dart's inherent scalability enable OCTANE MOBILE APP to efficiently scale its infrastructure and resources. This ensures that the application remains responsive and performs optimally, even during periods of high demand. By utilizing Dart's scalability features, OCTANE MOBILE APP .

1.3.FIREBASE

i)Real -time Data Synchronization:

Firebase Cloud Firestore's real-time data synchronization ensures seamless updates across multiple devices in OCTANE. This feature allows users to instantly see changes in car listings, pricing updates,. With real-time synchronization, OCTANE users can make informed decisions in real-time.

ii)Firebase Authentication for User Security:

Firebase Authentication provides a secure and straightforward user authentication process for OCTANE MOBILE APP. By leveraging Firebase Authentication, OCTANE MOBILE APP ensures that only authenticated users can access the platform. Multi-factor authentication and identity verification features enhance security, safeguarding user data and transactions within the application.

iii) Serverless Cloud Functions:

Firebase offers serverless cloud functions that enable the execution of backend logic without managing servers. OCTANE utilizes these functions for various tasks, such as sending notifications, updating user data, and handling asynchronous operations. This serverless architecture enhances scalability and efficiency.

iv)Firebase Cloud Storage for Media Assets:

Firebase's serverless cloud functions are utilized in OCTANE MOBILE APP to execute backend logic for managing servers. These functions handle tasks such as sending notifications, processing, and updating user preferences. By leveraging serverless architecture, OCTANE MOBILE APP enhances scalability and efficiency, ensuring smooth performance even during periods of high demand.

1.4.PROBLEM STATEMENT

In the contemporary landscape of second-hand car sales, individuals often face challenges stemming from fragmented platforms and limited resources. Existing solutions lack a comprehensive approach that caters to the diverse needs of both buyers and sellers, resulting in a disjointed experience and missed opportunities for seamless transactions.

The current ecosystem for buying and selling pre-owned cars is characterized by disjointed platforms that fail to provide a unified solution for users. Separate platforms cater to various aspects of the car-buying process, such as browsing listings, communicating with sellers, and completing transactions. This fragmentation complicates the user experience and often leads to inefficiencies in the buying and selling process.

Furthermore, existing platforms may lack inclusivity, restricting access to certain demographics or geographical locations. This limited reach hampers the diversity of perspectives and experiences available to users, potentially excluding valuable contributors from the marketplace.

One of the primary challenges faced by individuals in the second-hand car market is the lack of a cohesive platform that integrates all aspects of the buying and selling process. This fragmented approach can lead to inefficiencies, missed opportunities, and a lack of transparency for both buyers and sellers.

Moreover, the absence of inclusivity in existing platforms can further exacerbate these challenges, limiting access to certain demographics or geographic regions. This lack of diversity can result in a narrow range of perspectives and experiences,

The OCTANE MOBILE APP platform aims to address these challenges by providing a comprehensive solution for buying and selling second-hand cars. By integrating all aspects of the transaction process into a single platform, OCTANE streamlines the user experience and enhances transparency for both buyers and sellers. Additionally, OCTAN MOBILE APP prioritizes inclusivity, ensuring that all users have equal access to the platform and its resources, regardless of their background or location.

Ultimately, OCTANE MOBILE APP seeks to revolutionize the second-hand car market by offering a unified, inclusive, and user-friendly platform that meets the diverse needs of buyers and sellers alike. Through its innovative approach, OCTANE MOBILE APP aims to redefine the car-buying experience and empower individuals to make informed decisions with confidence.

1.5.OBJECTIVES

The primary objective of OCTANE is to establish a versatile and inclusive platform that seamlessly integrates all aspects of the second-hand car buying and selling process. The platform aims to bridge the gap between fragmented resources and the dynamic needs of users, providing a comprehensive solution for both buyers and sellers in the pre-owned car market.

Key Challenges:

i) Technical Complexity:

The use of advanced technologies like Flutter, Dart, and Firebase might have presented technical challenges such as compatibility issues, learning curves, or debugging complexities.

ii) Cross-Platform Development:

Developing a mobile application that functions seamlessly on both iOS and Android platforms could have posed challenges in terms of platform-specific requirements, UI/UX design considerations, and optimization.

iii) Security Concerns:

Integrating Firebase for user authentication and data storage may have required careful consideration of security measures to protect user data from potential vulnerabilities or breaches.

iv) User Engagement and Retention:

Creating a platform that effectively engages users and encourages ongoing participation can be challenging. Ensuring features are intuitive, valuable, and consistently updated to retain user interest may have been a challenge.

CHAPTER 2

METHODOLOGY

2.1.INITIAL PROJECT REQUIREMENTS:

The initial project requirements for OCTANE MOBILE APP encompass various aspects such as functionalities, technologies, and packages.

i)Platform Compatibility:

- 1) Develop a mobile application named " OCTANE MOBILE APP".
- 2) The mobile app should be compatible with both iOS and Android platforms to reach a wider audience. Address both academic and job-related queries within theplatform.

ii)User Registration and Authentication:

Implement a user registration system where users can create accounts securely. Authentication mechanisms such as email verification or social media login should be included to ensure user privacy.

iii) Car Listings Database:

- 1) Develop a comprehensive database of second-hand cars available for sale, including details such as make, model, year, mileage, price, and additional features.
- 2) The database should be easily searchable and regularly updated with new listings.

iv) User Interface Design:

Design an intuitive and user-friendly interface for the app, allowing users to easily browse through car listings, view details, and interact with the app's features. Consider implementing modern design principles and user experience best practices to enhance usability.

v) Packages Used:

i) material.dart:

This package provides widgets and utilities for implementing Material Design components in Flutter apps. It includes widgets for creating UI elements like buttons, app bars, and bottom navigation bars.

ii) homeOne.dart:

This appears to be a custom file containing a widget named HomeOne. Assuming it's part of your project, it likely contains the implementation for the home screen or a related component.

iii) profilePage.dart:

Similarly, this is another custom file containing a widget named ProfilePage. It's likely responsible for displaying the user's profile information or related functionalities.

iv) get.dart:

This package is used for state management and navigation in Flutter apps. It provides a simple and intuitive API for managing app state and navigating between screens. In this code snippet, the Get package is used for navigation, specifically for navigating to the LoginPage when the user logs out or deletes their account.

v) luttertoast.dart:

This package is used for displaying toast messages in Flutter apps. It provides a simple API for showing toast notifications to users. In this code snippet, the Fluttertoast package is used to display

messages such as "Enter an email", "Enter a password", "Login success", and "Invalid email/password" based on user actions during the login process.

vi) get.dart:

This package is used for state management and navigation in Flutter apps. It provides a simple and intuitive API for managing app state and navigating between screens. In this code snippet, the Get package is used for navigation, specifically for navigating to the HomePage after successful login.

2.2.PROPOSED APPLICATION ARCHITECTURE

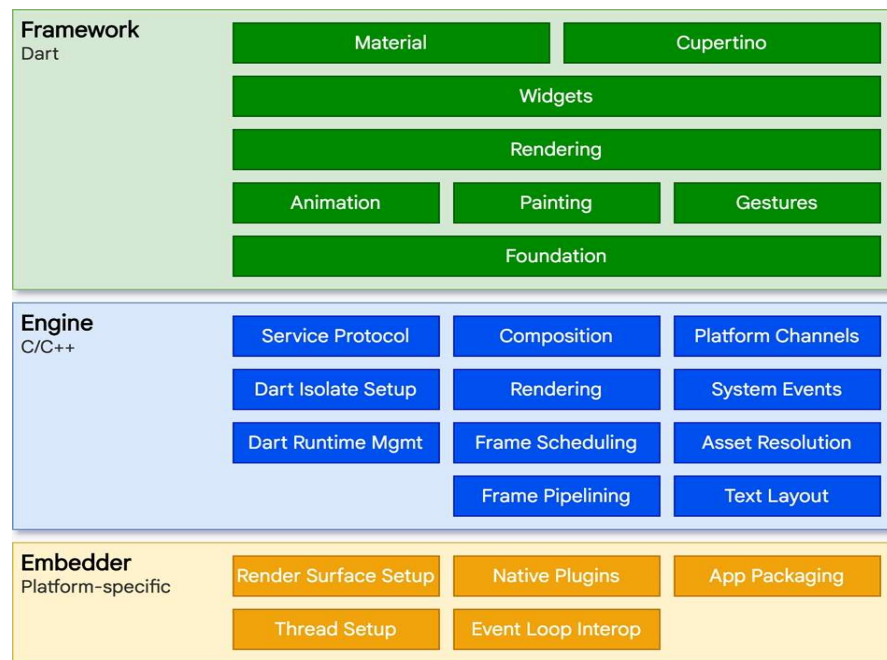


Fig:2.1: Proposed Application Architecture

The architecture of Flutter development source code primarily revolves around the Flutter framework itself, which follows a reactive, declarative programming paradigm. Flutter applications are built using a combination of Dart code, widgets, and various APIs provided by the Flutter SDK. Here's a breakdown of the key architectural components:

1.Flutter SDK:

The Flutter software development kit (SDK) provides everything developers need to build cross-platform mobile applications. It includes libraries, APIs, tools, and the Dart runtime.

2.Dart Programming Language:

Dart is the programming language used for building Flutter apps. It's a modern, object-oriented language with features like strong typing, asynchronous programming, and a rich standard library. Dart code is compiled ahead-of-time (AOT) into native machine code for performance.

3.Widgets:

Widgets are the building blocks of Flutter UIs. Everything in a Flutter app is a widget, from basic elements like buttons and text inputs to complex layouts and animations. Widgets can be stateless or stateful, and they are composed hierarchically to create the user interface.

4.Widget Tree:

The UI of a Flutter app is represented as a widget tree, where each widget represents a UI element. Widgets can have child widgets, forming a tree structure that defines the layout and composition of the UI.

5.State Management:

Flutter provides various techniques for managing the state of an application, including `setState` for managing internal widget state, `InheritedWidget` for sharing state across the widget tree, and external state management solutions like Provider, Bloc, Redux, or Riverpod.

6.Flutter Engine:

The Flutter engine is the runtime and rendering engine that powers Flutter apps. It's written in C++ and provides low-level rendering capabilities using Skia, a 2D graphics library. The engine manages tasks like layout, painting, and gesture recognition.

7.Platform Channels:

Flutter apps can communicate with platform-specific code using platform channels. This enables integration with native platform features and APIs that are not available in the Flutter framework directly. Developers can use platform channels to call native code written in languages like Java (for Android) or Objective-C/Swift (for iOS).

8.Plugins:

Flutter plugins are packages that provide access to platform-specific APIs and services. They encapsulate platform channel communication and provide a Flutter-friendly interface for interacting with native features like camera, geolocation, sensors, and more.

9.Material Design and Cupertino Widgets:

Flutter provides two sets of widgets for building UIs that adhere to Material Design guidelines (for Android) and Cupertino design guidelines (for iOS). These widgets offer platform-specific styling and behaviors to create native-looking experiences on both platforms.

2.3.FLOW DIAGRAM

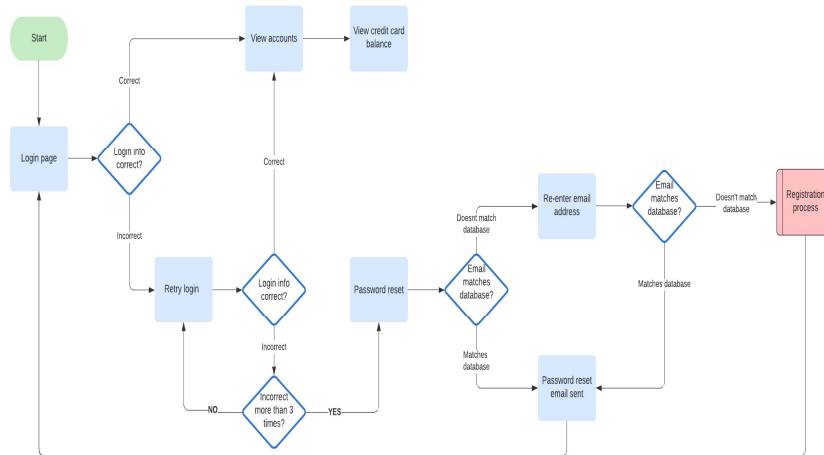


Fig:2.2: Flow Diagram

The following steps are used to describe the above flow diagram:

- i) The user opens the mobile app.
- ii) The app prompts the user to log in or sign up if they don't have an account yet.
- iii) If the user chooses to sign up, they will be taken to the sign-up page.
- iv) If the user is already registered, they can log in directly.
- v) Once the user is logged in, they will be taken to the home page.
- vi) From the home page, the user can access various features of the app such as their profile page, feedback page, message page, mentor selection page, and details page.
- vii) The mentor selection page allows the user to choose a mentor and view their details.
- viii) The details page provides information about the mentor.
- ix) The user can also leave feedback on the app.

2.4.MODULES

2.4.1.User Authentication and Authorization::

Allow users to register, log in, and manage their accounts securely.Implement authentication mechanisms such as email/password, social login (e.g., Google, Facebook), or phone number verification.Handle user roles and permissions for accessing app features

Algorithm:

- i) User Registration: When a user registers for the app, they provide necessary details such as email, password, and optionally, additional information like name and phone number.Validate the input data to ensure it meets requirements (e.g., strong password policy).Store user credentials securely in the database, hashing the password using a strong cryptographic hash function (e.g., bcrypt).
- ii) User Login: When a user attempts to log in, verify their credentials by comparing the provided email and password with the stored values in the database.If the credentials match, generate a unique authentication token (JWT or JSON Web Token) containing the user's ID and any relevant metadata.Return the token to the client, which will be used to authenticate subsequent requests.
- iii) Token-Based Authentication:Require clients to include the authentication token in the header of each request to protected endpoints. Validate the token server-side to ensure it's not expired, hasn't been tampered with, and belongs to a valid user.If the token is valid, allow access to the requested resource; otherwise, deny access and return an appropriate error message.
- iv) Secure Communication:Ensure all communication between the client and server is encrypted using HTTPS to prevent eavesdropping and man-in-the-middle attacks.Implement other security best practices such as rate limiting, CSRF protection, and input validation to mitigate common security risks.

2.4.2.CAR LISTINGS:

Display a list of available second-hand cars with basic information (e.g., make, model, year, price). Enable users to view detailed information about each car, including specifications, mileage, and additional features.

Algorithm:

- i) **Fetch Car Listings:** Retrieve a list of available second-hand cars from the backend server/database. Ensure that the data fetched includes basic information such as make, model, year, price, and a unique identifier for each car.
- ii) **Display Basic Car Information:** Iterate through the list of car listings. For each car, display basic information such as make, model, year, and price in a visually appealing and organized manner on the app's UI. Use a suitable UI component, such as a list view or grid view, to present the car listings to the user.
- iii) **Handle User Interaction:** Implement functionality to allow users to select a car listing to view detailed information. Listen for user actions, such as taps or clicks, on the car listings displayed in the app.
- iv) **Navigate Back:** Provide a way for users to navigate back to the list of car listings after viewing detailed information about a specific car. Implement navigation controls, such as a back button or gesture swipe, to allow users to return to the previous screen seamlessly.
- v) **Reset Filters:** Offer an option for users to reset or clear the applied filters revert to the original unfiltered list of car listings.

2.4.3.SEARCH AND FILTERING:

Implement search functionality to allow users to find specific cars based on criteria such as make, model, price range, etc. Provide filtering options to refine search results and facilitate quick navigation.

Algorithm:

- i) Search Functionality: Provide a search bar or input field where users can enter keywords or criteria to search for specific cars. Listen for user input events (e.g., keystrokes) to trigger the search functionality. As the user types, dynamically filter the list of car listings based on matching criteria such as make, model, year, price, or any other relevant attributes.
- ii) Filtering Options: Offer filtering options to allow users to refine search results based on specific criteria. Display filtering controls, such as dropdown menus, sliders, or checkboxes, for attributes like make, model, year, price range, mileage, and more. Allow users to select multiple filtering options to narrow down the search results further. Apply the selected filters to the list of car listings and update the displayed results accordingly.
- iii) Apply Filters: Listen for user interactions with the filtering controls to apply the selected filters to the list of car listings. When a user selects or modifies a filtering option, dynamically update the displayed car listings to reflect the filtered results. Ensure that the filtering process is efficient and responsive, even for large datasets, to provide a seamless user experience. Images and associates them with their respective user profiles.
- iv) Reset Filters: Offer an option for users to reset or clear the applied filters revert to the original unfiltered list of car listings.

2.4.4.CAR DETAILS AND IMAGEE GALLERY:

Display detailed information about each car, including images, specifications, and seller contact information. Implement a gallery view for users to browse through multiple images of the car.

Algorithm:

i)Fetch Car Details: When a user selects a car listing from the list, retrieve detailed information about the selected car from the backend server/database. Ensure that the data fetched includes specifications (e.g., engine type, transmission, fuel type), mileage, seller contact information, and a list of images for the car.

ii)Display Car Details: Present the detailed information about the selected car in a separate screen or modal window. Display relevant details such as make, model, year, price, mileage, and seller contact information (e.g., name, email, phone number). Use appropriate UI components, such as text fields, labels, and buttons, to organize and present the information in a user-friendly manner

iii)Implement Image Gallery: Display a gallery view below or alongside the car details section to showcase multiple images of the car. Render thumbnails or small previews of each image in the gallery for quick navigation. Allow users to swipe or scroll through the images horizontally to browse through the entire image collection.

vi)Error Handling: Implement error handling mechanisms to gracefully handle scenarios where car details or image retrieval fails due to network issues or server errors. Display informative error messages to the user and provide options for retrying the operation or seeking further assistance.

2.4.5.USER PROFILE MANAGEMENT:

Allow users to create and manage their profiles within the app.Include features such as profile editing, viewing purchase history, and managing saved favorites.

Algorithm:

i)User Registration: Allow users to register for an account within the app by providing necessary details such as email, password, and optionally, additional information like name and phone number.Validate the input data to ensure it meets requirements (e.g., valid email format, strong password policy).Store user credentials securely in the database, hashing the password using a strong cryptographic hash function (e.g., bcrypt).

ii) User Login:Implement user authentication mechanisms to allow registered users to log in to their accounts securely.Verify user credentials by comparing the provided email and password with the stored values in the database.Upon successful login, generate a unique authentication token (e.g., JWT) to authenticate subsequent requests.

iii)Profile Editing:Implement features for users to edit and update their profile information as needed.Provide UI controls for users to modify their profile details, such as text fields, dropdown menus, and image upload functionality for changing profile pictures.Validate and sanitize user input to ensure data integrity and prevent errors or security vulnerabilities.

iv)Security and Access Control:Implement security measures to protect user profile data and ensure privacy and confidentiality.Enforce access control mechanisms to restrict access to sensitive profile features and information based on user authentication and authorization.

2.5.DEVELOPMENT AND DEPLOYMENT SETUP

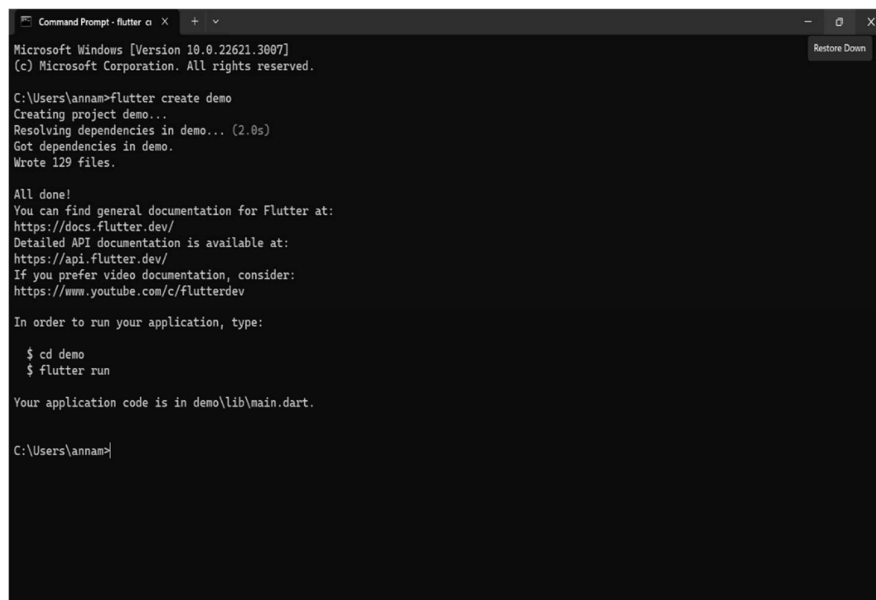
1. Development Environment:

Flutter and Dart need to be installed on the developer's machine.

IDEs like Visual Studio Code or Android Studio for coding.

2. Create Project:

```
>>flutter create project_name
```



```
Command Prompt - flutter a x + v
Microsoft Windows [Version 10.0.22621.3007]
(c) Microsoft Corporation. All rights reserved.

C:\Users\annam>flutter create demo
Creating project demo...
Resolving dependencies in demo... (2.0s)
Got dependencies in demo.
Wrote 129 files.

All done!
You can find general documentation for Flutter at:
https://docs.flutter.dev/
Detailed API documentation is available at:
https://api.flutter.dev/
If you prefer video documentation, consider:
https://www.youtube.com/c/flutterdev

In order to run your application, type:

$ cd demo
$ flutter run

Your application code is in demo\lib\main.dart.

C:\Users\annam>
```

Fig:2.3: Project Creation

From Fig:2.3 Open your terminal or command prompt. Navigate to the directory where you want to create the Flutter project.

3. Development Tools

Flutter commands used for testing and building the application. Real-time testing on Android and iOS devices.



Fig:2.4: Flutter Run

From Fig:2.4 To use the flutter run command, navigate to your Flutter project directory in the terminal or command prompt, and simply type flutter run followed by any additional flags or options you want to specify.

4. Algorithmic Approach:

To develop a OCTANE application, you can follow a high-level algorithmic approach.

4.1. User Authentication:

Example using LoginPage Authentication dart code:

```
class _LoginPageState extends State<LoginPage> {  
  TextEditingController emailController =  
    TextEditingController();  
  TextEditingController passwordController =  
    TextEditingController();  
  
  void loginAuth() {  
    String loginMail = "kumar@gmail.com";  
    String loginPass = "1204";  
  }  
}
```

Fig:2.5: User Authentication

From Fig:2.5 say a Flutter code snippet defines a stateful widget named `_LoginPageState` which represents the state for a login page.

4.2.Data Model:

Example using Main.dart code

```
import 'package:flutter/material.dart';
import 'package:get/get.dart';
import 'package:jawa_project/loginPage.dart'; // Importing from
the correct project

void main() {
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({Key? key});

  @override
  Widget build(BuildContext context) {
    return GetMaterialApp(
      debugShowCheckedModeBanner: false,
      home: const LoginPage(),
    );
  }
}
```

Fig:2.6:Data Model

From Figure:2.6 The code initializes a Flutter application with the Get package for state management and navigation, setting the LoginPage as the initial screen.

4.3.Testing:

Conduct thorough testing of the application, including unit tests, integrationtests, and user testing to ensure functionality and usability.

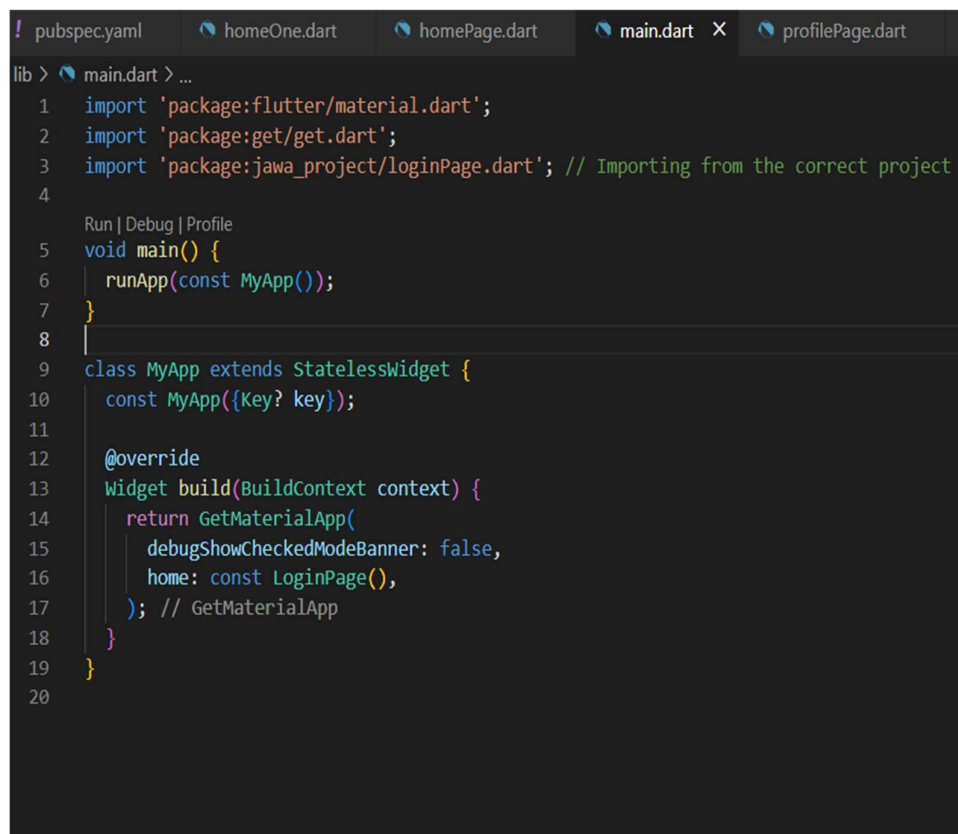
CHAPTER 3

RESULTS AND DISCUSSION

FRONT END

i)main.dart:

This Fig:3.1 shows the file is typically the entry point of a Flutter application. It initializes the app, setup necessary configurations, and runs the main application widget.



```
! pubspec.yaml  homeOne.dart  homePage.dart  main.dart X  profilePage.dart
lib > main.dart > ...
1  import 'package:flutter/material.dart';
2  import 'package:get/get.dart';
3  import 'package:jawa_project/loginPage.dart'; // Importing from the correct project
4
Run | Debug | Profile
5  void main() {
6    runApp(const MyApp());
7  }
8
9  class MyApp extends StatelessWidget {
10   const MyApp({Key? key});
11
12   @override
13   Widget build(BuildContext context) {
14     return GetMaterialApp(
15       debugShowCheckedModeBanner: false,
16       home: const LoginPage(),
17     ); // GetMaterialApp
18   }
19 }
20
```

Fig:3.1: Main Dart

ii) loginPage.dart

This Fig:3.2 Shows The file likely contains the UI and logic for user login. It includes form fields for entering credentials and a mechanism to authenticate users.

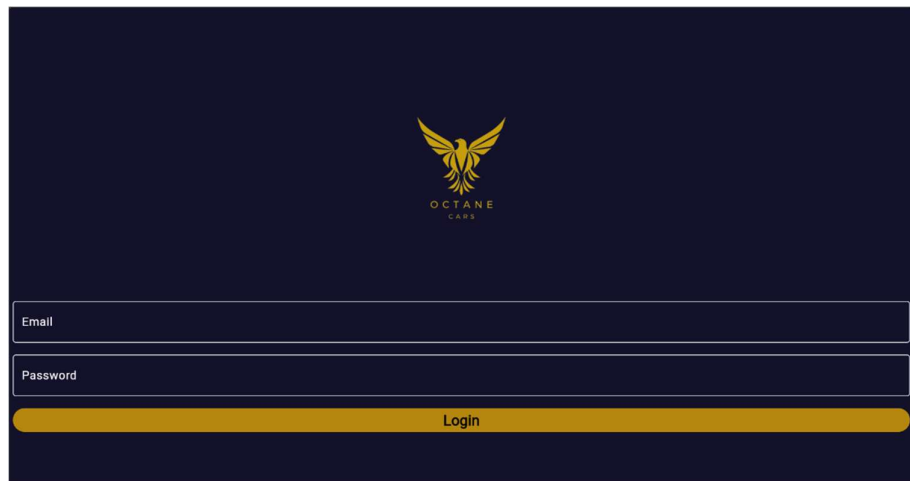


Fig:3.2. Login Page

iii)HomePage.dart:

This Fig :3.3 is likely shows a cars Brand logos.

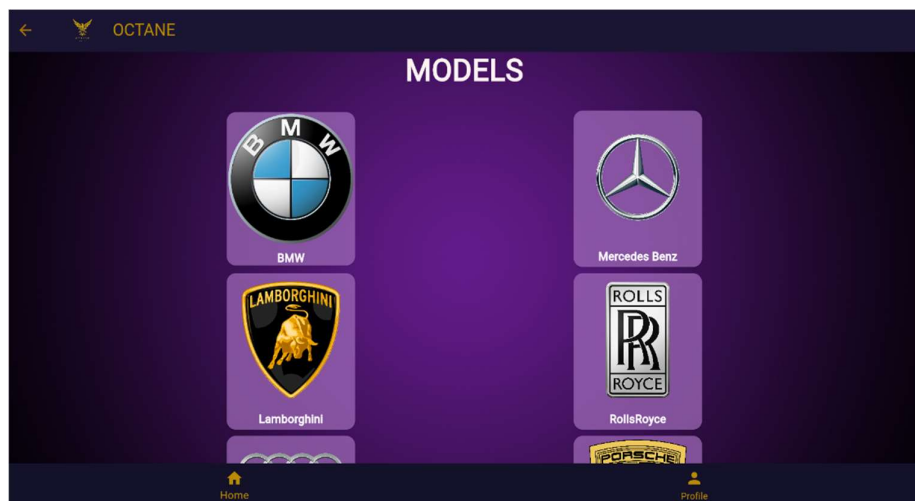


Fig:3.3. HomePage

iv)Rolls RoycePage.dart:

This Fig:3.4 shows the file represents the main content of your app after a successful OnTap to Rolls Royce card . . It contains Rolls RoycePage with various features.

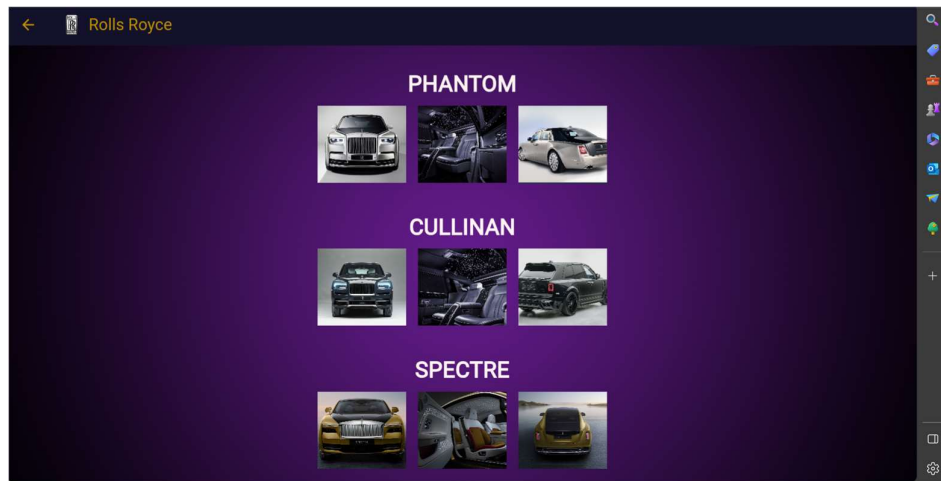


Fig.3.4 Rolls RoycePage

v) BMWPage.dart:

This Fig:3.5 shows file represents the main content of your app after a successful OnTap to BMW Logo card . . It contains BMWPage with various features.

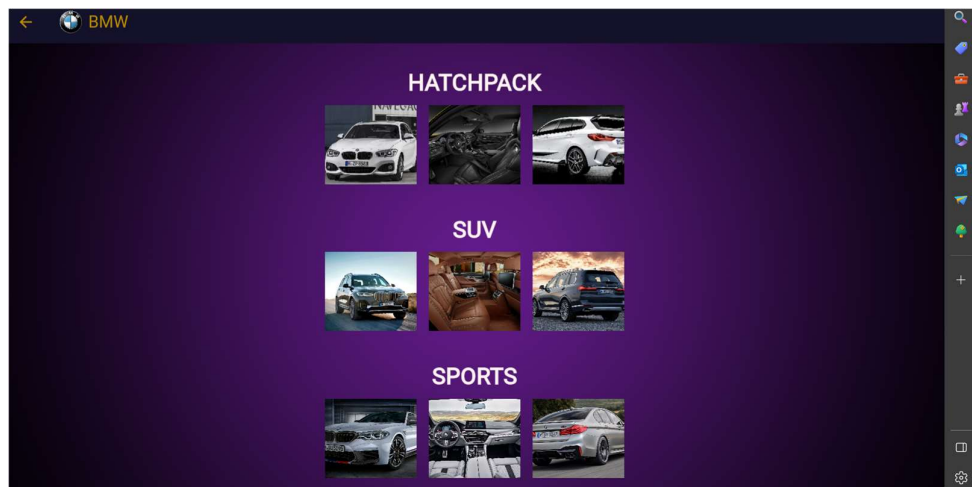


Fig.3.5. BMWPage

vi)profilePage.dart:

This Fig:3.6 is likely responsible for displaying and editing user profiles. It includes information about the user, such as their name, profile picture, and additional details.

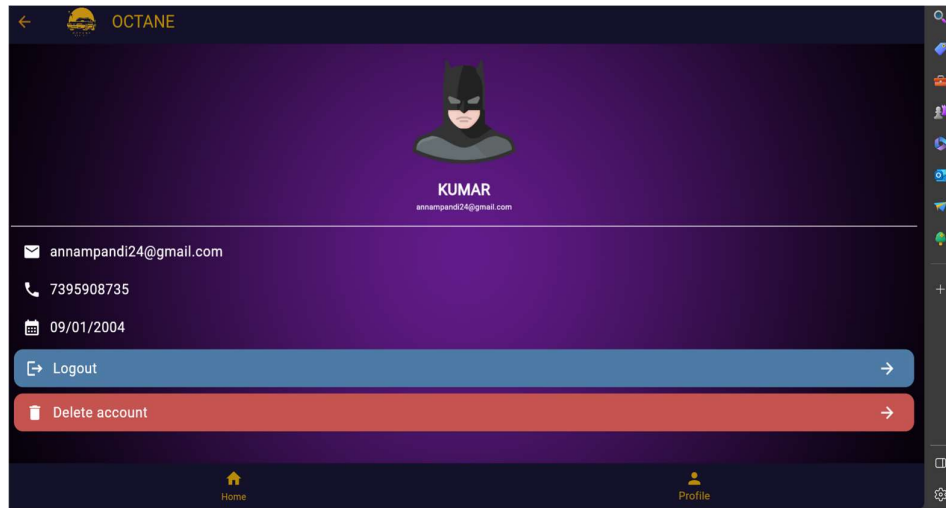


Fig:3.6:Profile Page

vii) BenzPage.dart:

This file represents the main content of your app after a successful OnTap to Benz Logo card.It contains BenzPage with various features.

viii) AudiPage.dart:

This file represents the main content of your app after a successful OnTap to Audi Logo card. It contains AudiPage with various features.

ix) PorschePage.dart:

This file represents the main content of your app after a successful OnTap to Porsche Logo card .

x) LamborghiniPage.dart

This file represents the main content of your app after a successful OnTap to Lamborghini Logo card . It contains LamborghiniPage with various features.

OCTANE APPLICATION FILE:

The OCTANE application stands as a successful and transformative platform that effectively addresses the needs of both students and mentors in the realm of collaborative learning and mentorship. The utilization of Flutter and Dart technologies ensures a visually appealing and responsive user interface, providing a seamless experience across different platforms, be it iOS or Android.

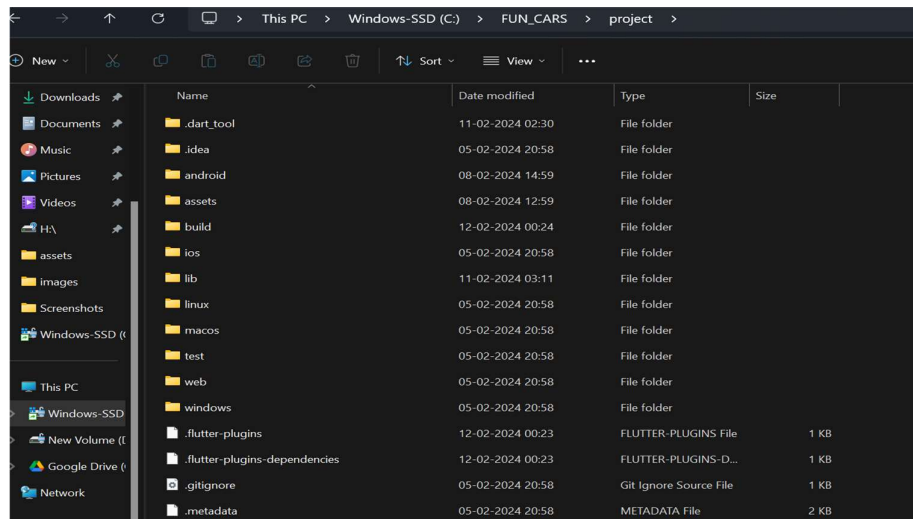


Fig.3.7 OCTANE App

This Fig:3.7 is likely contains code for a Flutter application designed to facilitate second-hand car trading, providing features such as car listings, user authentication, and profile management.

CHAPTER 4

CONCLUSION AND FUTURE WORK

4.1. CONCLUSION

In summary, our project embodies a paradigm shift in the realm of mobile applications by introducing OCTANE, a cutting-edge platform that seamlessly integrates academic learning with professional mentorship. Leveraging the advanced capabilities of Flutter and Dart, we have crafted an intuitive and cross-platform interface, ensuring a consistent and immersive experience for users across iOS and Android devices. The incorporation of Firebase adds a layer of security and enhances the app's functionality, particularly in user authentication.

OCTANE MOBILE APP emerges as a vibrant hub for collaborative learning, offering a space not only for academic discussions but also for tackling real-world challenges, refining interview skills, and exploring industry-specific topics. Its inclusive environment fosters diversity and encourages active participation, enabling learners and mentors from diverse backgrounds to share insights and collectively bridge the gap between theory and practice.

Through its user-centric design, real-time communication features, and emphasis on fostering meaningful connections, OCTANE redefines the landscape of digital mentorship. Our project underscores the transformative potential of technology in creating enriched learning environments that connect individuals, promote professional growth, and facilitate the synthesis of knowledge.

4.2. FUTURE WORK

1.Enhanced User Profiles:

Implement additional fields and customization options in user profiles, allowing users to showcase more details about their skills, interests, and experiences.

2.Advanced Mentor Matching Algorithm:

Develop a more sophisticated algorithm for mentor-student matching, considering factors like specific expertise, project interests, and availability to enhance the quality of mentorship connections.

3.Real-time Collaboration Tools:

Integrate real-time collaboration tools, such as collaborative document editing, code sharing, or video conferencing, to facilitate more interactive mentor-student sessions.

4.Community Forums and Discussions:

Expand the platform by introducing community forums or discussion boards where users can engage in broader discussions, ask open-ended questions, and share knowledge with a wider audience.

5.Integration with Learning Resources:

Collaborate with educational platforms or content providers to integrate learning resources directly into the app, providing users with access to tutorials, courses, and industry-specific materials

REFERENCES

1. Journal.s:

[1] B. Thompson, "International Journal of Educational Technology in Higher Education" pp.1-15, Feb 2021.

[2] D. Garcia, E. Lee, F. Chen, "Technology-Enhanced Mentorship", "International Journal of Educational Technology in Higher Education" pp.1-15 Jun 2021.

[3] K. Patel, L. Nguyen, "Exploring the Role of Technology in Mentoring Programs for Underrepresented Students", "Computers & Education" pp.1-25, May 2021.

[4] M. Jones, N. Williams, "Digital Tools for Mentorship", "Journal of Interactive Learning Research", pp.321-335 Dec 2020.

[5] S. Gupta, "Technology-Mediated Mentorship", "British Journal of Educational Technology", volume:50, pp.1234-1248, Jan 2019.

2. Books:

[1] K. Patel, "Flutter for Mobile App Development in Education", Publisher: "Packt", Publishing Year: 2019.

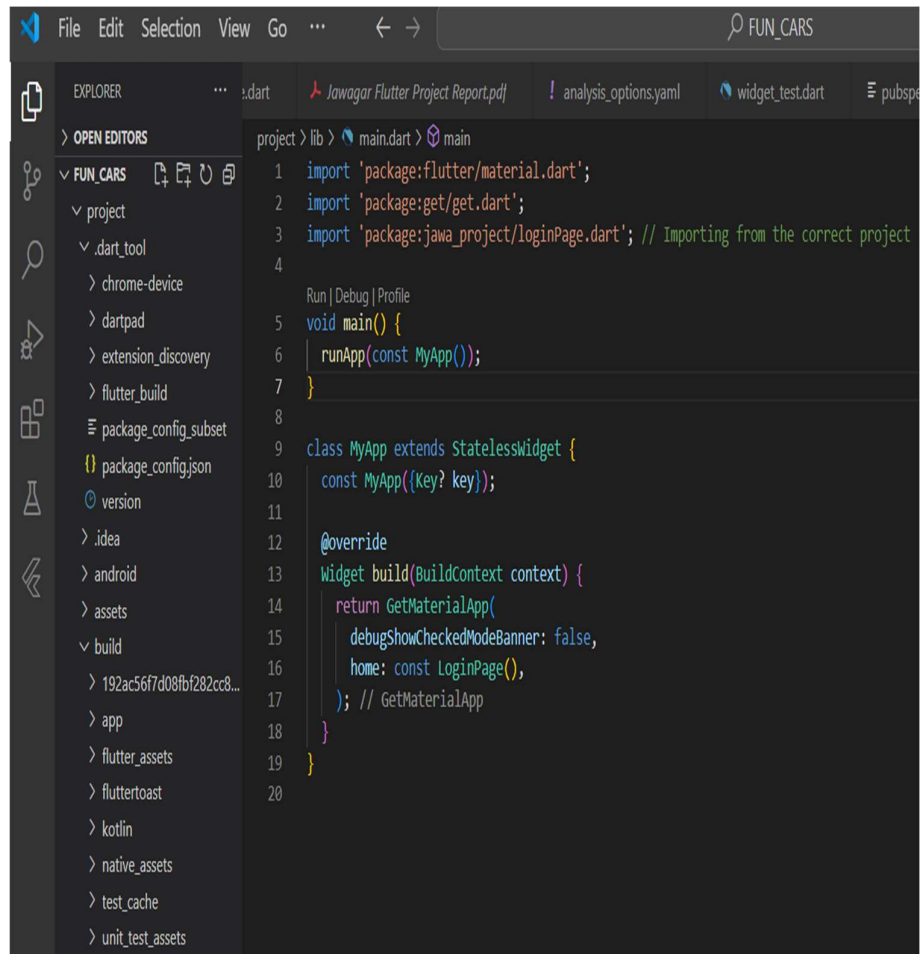
[2] K. M. Smith, "Building Virtual Communities: Learning and Change in Cyberspace", Publisher: Cambridge University Press, Year: 2019.

[3] M. Davis, "Building Virtual Communities for Knowledge Exchange", Publisher: "Wiley", year: 2021.

APPENDICES

A.SAMPLE CODE

i)Main Code:



```
File Edit Selection View Go ... < > FUN_CARS

EXPLORER
> OPEN EDITORS
FUN_CARS
  project
    .dart_tool
      chrome-device
      dartpad
      extension_discovery
      flutter_build
      package_config_subset
      package_config.json
      version
    .idea
    android
    assets
    build
      192ac56f7d08fb282cc8...
      app
      flutter_assets
      fluttertoast
      kotlin
      native_assets
      test_cache
      unit_test_assets

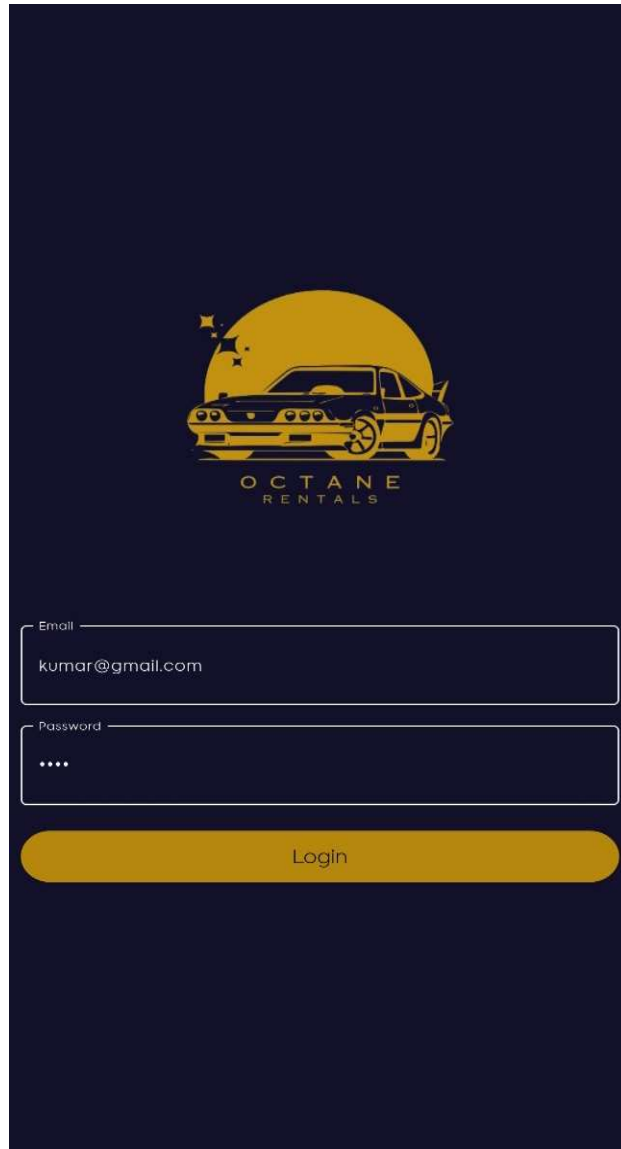
project > lib > main.dart > main
1 import 'package:flutter/material.dart';
2 import 'package:get/get.dart';
3 import 'package:java_project/loginPage.dart'; // Importing from the correct project
4
Run | Debug | Profile
5 void main() {
6   runApp(const MyApp());
7 }
8
9 class MyApp extends StatelessWidget {
10   const MyApp({Key? key});
11
12   @override
13   Widget build(BuildContext context) {
14     return GetMaterialApp(
15       debugShowCheckedModeBanner: false,
16       home: const LoginPage(),
17     ); // GetMaterialApp
18   }
19 }
20
```

ii)pubspec.yaml

```
project > ! pubspec.yaml
1  name: jawa_project
2  description: "A new Flutter project."
3
4  publish_to: 'none'
5
6  version: 1.0.0+1
7
8  environment:
9    sdk: '>=3.2.5 <4.0.0'
10 dependencies:
11   flutter:
12     sdk: flutter
13   cupertino_icons: ^1.0.2
14   get: ^4.6.6
15   fluttertoast: ^8.2.4
16
17 dev_dependencies:
18   flutter_test:
19     sdk: flutter
20   flutter_lints: ^3.0.1
21 flutter:
22   uses-material-design: true
23
24   assets:
25     - assets/images/Logo3.png
26     - assets/images/UserAvatar.png
27     - assets/images/BG.jpg
28     - assets/images/BMW logo.png
29     - assets/images/BenzLogo.png
30     - assets/images/lamborghiniLogo.png
31     - assets/images/RollsRoyceLogo.png
32     - assets/images/Audifront.jpg
33     - assets/images/Audirear.jpg
34     - assets/images/Audiinterior.jpg
35     - assets/images/BMWSUVfront.jpg
36     - assets/images/BMWSUVinterior.jpg
37     - assets/images/BMWSUVrear.jpg
38     - assets/images/BMWsportsfront.jpg
39     - assets/images/BMWsportsinterior.jpg
```

B.SCREEN SHOTS

i)Login page And Signup Page



The image shows a mobile app login screen for 'OCTANE RENTALS'. The background is a dark blue gradient. At the top, there is a logo featuring a yellow car silhouette in front of a large yellow sun, with the text 'OCTANE RENTALS' below it. Below the logo, there are two input fields: 'Email' with the text 'kumar@gmail.com' and 'Password' with four dots. A yellow 'Login' button is positioned below the password field. At the bottom, there is a link that says 'Forgot your password?'.

OCTANE
RENTALS

Email

Password

Login

[Forgot your password?](#)

ii) Homepage to select car Model:



iii) Car Models:



iv) Profile Page:

