

Professional Training Report

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

By

KIRAN M(41120094)



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

APRIL - 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 **KIRAN M(41120094)** who carried out the project entitled "Car Rent and Purchase App" under my supervision from **January 2024 to May 2024**.

Internal Guide

Dr.C.GEETHA,M.Tech.,Ph.D.,

Head of the Department

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

Submitted for Viva voce Examination held on 08/05/2024

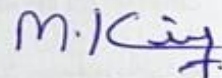
Internal Examiner
External Examiner

DECLARATION

I Kiran M hereby declare that the Project Report entitled "Car Rent and Purchase App" done by me under the guidance of Dr.C.GEETHA, M.Tech., Ph.D., 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** 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

, 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 **Dr. C. GEETHA M.TECH., Ph.D** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my phase-2 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 **Kiran M (Reg No: 41120094)** 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, **Kiran M** has closely worked as a part of the **Flutter** by carrying out the project entitled "**Car Rent and Purchase App**".

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 light blue circular stamp.

Ravikumar S
CEO & Managing Director
Hackwit Technologies Pvt Ltd

CERTIFICATE OF COMPLETION



ABSTRACT

The Car Rental and Purchase Management System is a comprehensive Flutter project aimed at facilitating the process of both renting and buying cars through a user-friendly mobile application. The system caters to the needs of both individuals and businesses, offering a seamless experience for users looking to rent a car for short-term use or purchase a vehicle for long-term ownership.

The project incorporates key features such as user authentication, car browsing, booking/reservation system, payment integration, and administrative functionalities. Users can easily browse through available cars, view detailed information including specifications and rental/purchase pricing, and make bookings or purchases directly through the app.

The application provides a secure payment gateway for transactions, ensuring smooth and hassle-free payments. Additionally, users have the flexibility to manage their bookings, view rental history, and access support services within the app.

Administrators have access to a dashboard where they can manage inventory, add new cars, update pricing, monitor bookings, and handle user inquiries. The system also includes analytics features to track performance metrics and make informed decisions to optimize operations.

Overall, the Car Rental and Purchase Management System offers a convenient platform for both customers and businesses in the automotive industry, streamlining the process of renting and purchasing cars while providing a user- friendly and efficient experience.

TABLE OF CONTENTS

Chapter No	TITLE	Page No.
	ABSTRACT	vii
	LIST OF ABBREVIATIONS	ix
	LIST OF FIGURES	x
1	INTRODUCTION	1
	1.1 FLUTTER	3
	1.2 FIREBASE	3
	1.3 IMPLEMENTING FIREBASE IN YOUR APP	4
	1.4 PROBLEM STATEMENT	5
	1.5 KEY FEATURES	7
2	METHODOLOGY	9
	2.1 TECHNOLOGIES	10
	2.2 PROPOSED APPLICATION ARCHITECHTURE	12
	2.3 DEVELOPMENT AND DEPLOYMENT SETUP	14
3	RESULTS AND DISCUSSION	18
	3.1 CONFIGURATIONS	19
4	CONCLUSION AND FUTURE WORK	22
	4.1 CONCLUSION	22
	4.2 FUTURE WORK	22
	REFERENCES	24
	APPENDICES	25
	A. SAMPLE CODE	25
	B. SCREENSHOTS	29

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
UI	User Interface
UX	User Experience
IOS	IPhone Operation System
UAT	User Acceptance Testing
API	Application Programming Interface

LIST OF FIGURES

Figure No.	Name of the figure	Page No.
2.1	Flow Diagram	10
2.2	Proposed Application Architecture	12
2.3	Firebase Architecture	13
2.4	Project Creation	14
2.5	Firebase Connection	15
2.6	Firebase Authentication	16
2.7	Flutter Run	17
2.8	Onesignal notification	17
3.1	App icon	19
3.2	Firebase Configurations	19

CHAPTER 1

INTRODUCTION

Welcome to our cutting-edge car rental and purchase app, built with Flutter and powered by Firebase! Our app offers a seamless experience for users looking to rent or purchase vehicles, all in one convenient platform.

With our app, users can easily browse through a wide selection of cars, view detailed information and images, compare prices, and make bookings or purchases with just a few taps. Whether you need a car for a weekend getaway or are looking to buy your dream car, our app has you covered.

Firebase integration ensures that user data is securely stored and synchronized across devices, providing a seamless experience no matter where you access the app from. Real-time updates and notifications keep users informed about their bookings, purchases, and any new offers or promotions.

We are committed to providing a hassle-free and enjoyable experience for our users, and we believe that our Flutter car rental and purchase app with Firebase sets a new standard in the automotive industry. Download our app now and experience the future of car rental and purchase!

The Car Buy and Rent Flutter Project aims to address this need by providing a comprehensive solution for individuals and businesses seeking to rent or purchase vehicles seamlessly through a mobile application.

With the proliferation of smartphones and the increasing reliance on digital platforms, there's a growing demand for efficient and user-friendly solutions in various domains, including the automotive industry. The Car

Buy and Rent Flutter Project endeavors to offer a one-stop solution for users looking to explore their options for renting or buying vehicles.

Whether it's a compact sedan for a weekend road trip, a spacious SUV for a family vacation, or a luxury car for a special occasion, the app aims to cater to diverse preferences and requirements.

By integrating advanced features such as user authentication, real-time inventory management, secure payment processing, and seamless booking/reservation systems, the project aims to streamline the entire car rental and purchase experience.

Furthermore, the project emphasizes the importance of user experience and accessibility. The application will feature an intuitive interface, responsive design, and smooth navigation to ensure a pleasant and hassle-free experience for users across different devices and platforms.

Whether accessing the app from a smartphone, tablet, or desktop computer, users can expect a consistent and seamless experience throughout their journey—from browsing available cars to completing transactions.

This project seeks to revolutionize the car rental and purchase process by leveraging the power of Flutter—a popular cross-platform development framework—to create a modern and intuitive mobile application.

The Car Buy and Rent Flutter Project aims to address these challenges by developing a comprehensive mobile application that serves as a centralized platform for car rental and purchase services.

Leveraging the power of Flutter—a cross-platform development framework—the project will create a user-friendly interface accessible via smartphones, tablets, and desktops.

1.1 FLUTTER:

1. What is Flutter?

- Flutter is an open-source UI software development kit created by Google.
- It is used to develop natively compiled applications for mobile, web, and desktop from a single codebase.

2. Key Features of Flutter:

- **Fast Development:** Hot reload allows you to quickly see changes.
- **Expressive UI:** Rich set of material design and Cupertino (iOS-flavor) widgets.
- **Native Performance:** Flutter apps are compiled to native code, providing near-native performance.

3. How Flutter Works:

- Flutter uses the Dart programming language.
- It includes a rich set of widgets for building interfaces.
- The Flutter framework handles gestures, animations, and more.

1.2 Firebase:

1. What is Firebase?

- Firebase is a platform developed by Google for creating mobile and web applications.
- It provides various services like Firestore (NoSQL database), Firebase Authentication, Cloud Storage, and more.

2. Key Features of Firebase:

- **Realtime Database:** Firebase offers a real-time NoSQL database.
- **Authentication:** Easy-to-use authentication methods like email/password, Google sign-in, etc.
- **Cloud Firestore:** A scalable database for mobile, web, and server development.

- **Cloud Functions:** Run backend code in response to events triggered by Firebase features.

3. How Firebase Works with Flutter:

- Firebase provides Flutter plugins to interact with its services.
- You can integrate Firebase services into your Flutter app to add authentication, database, and storage functionalities.

1.3 Implementing Firebase in Your App:

1. Setting up Firebase:

- Create a Firebase project in the Firebase Console.
- Add your app to the project and download the ``google-services.json`` file for Android or ``GoogleService-Info.plist`` file for iOS.

2. Adding Firebase to Your Flutter App:

- Add the Firebase Flutter plugins to your ``pubspec.yaml`` file.
- Initialize Firebase in your Flutter app using the downloaded configuration files.

3. Using Firebase Services:

- Use Firebase Authentication to manage user authentication.
- Use Cloud Firestore or the Database to store and retrieve data.
- Use Cloud Storage to store user-generated content like images or files.

4. Handling Firebase Security Rules:

- Configure Firebase security rules to control access to your data.
- Ensure that only authenticated users can access certain parts of your database.

5. Testing and Deployment:

- Test your app to ensure that Firebase integration works as expected.
- Deploy your Flutter app with Firebase to the Google Play Store or Apple App Store.

This is a general overview of Flutter and Firebase concepts. For your specific app, you'll need to dive deeper into each topic and implement the necessary features to create a successful car rental and purchase app.

1.4 Problem Statement:

Develop a Flutter mobile application that enables users to rent and purchase cars using Firebase as the backend. The app should provide a user-friendly interface for browsing available cars, making rental or purchase requests, and managing user accounts. The app should also include features such as real-time updates on car availability, secure user authentication, and seamless integration with Firebase services for data storage and management.

1. User Authentication and Profiles:

- **Theory:** User authentication is crucial for ensuring that only authorized users can access the app's features. By implementing Firebase Authentication, users can securely create accounts, sign in, and manage their profiles. This enhances the app's security and provides a personalized experience for users.

- **Implementation:** Use Firebase Authentication to handle user sign-up, sign-in, and profile management. Utilize Firebase Firestore to store user profiles and related information.

2. Car Listings and Details:

- **Theory:** Displaying a comprehensive list of available cars with detailed information helps users make informed decisions. By showcasing key details such as model, make, year, price, and features, users can easily compare and select cars that meet their requirements.

- **Implementation:** Use Firebase Firestore to store and retrieve car listings and details. Use Flutter's UI components to display the information in a user-friendly manner.

3. Search and Filter Options:

- **Theory:** Providing robust search and filter options enhances the user experience by allowing users to quickly find cars that match their preferences. This feature improves usability and helps users find cars more efficiently.

- **Implementation:** Implement search and filter functionalities using Flutter's built-in widgets. Use Firebase Firestore queries to fetch relevant car data based on user-selected criteria.

4. Booking and Payment:

- **Theory:** Enabling users to book cars and make payments within the app streamlines the rental and purchase process. Secure payment gateways ensure that transactions are safe and convenient for users.

- **Implementation:** Integrate payment gateways such as Stripe or PayPal for secure transactions. Use Firebase Firestore to manage booking details and payment status.

5. Real-time Updates:

- **Theory:** Providing real-time updates on car availability and booking status keeps users informed and enhances their overall experience. Users can make timely decisions based on the latest information.

- **Implementation:** Use Firebase Realtime Database or Firestore's real-time updates to notify users of changes in car availability, booking status, and other relevant information.

6. Notifications:

- **Theory:** Push notifications are essential for keeping users engaged and informed about their bookings, payments, and any updates related to their transactions. Notifications improve user engagement and retention.

- **Implementation:** Use Firebase Cloud Messaging (FCM) to send push notifications to users' devices. Implement logic to trigger notifications for relevant events such as booking confirmation or payment reminders.

7. Feedback and Support:

- **Theory:** Providing users with a feedback mechanism and support options helps enhance the app's quality and user satisfaction. Users can provide suggestions, report issues, and seek assistance when needed.

- **Implementation:** Include feedback forms within the app for users to provide comments and ratings. Provide contact options for users to reach out to support staff for assistance.

8. Admin Dashboard:

- **Theory:** An admin dashboard allows car providers to manage their listings, view booking requests, and update car availability. This feature provides car providers with a convenient way to manage their inventory and communicate with users.

- **Implementation:** Develop an admin dashboard using Flutter web or desktop support. Use Firebase Firestore to sync data between the app and the admin dashboard, enabling real-time updates.

9. Analytics and Reporting:

- **Theory:** Analytics tools help track user interactions, monitor app performance, and generate reports on key metrics. This data can be used to improve the app's functionality, user experience, and overall performance.

- **Implementation:** Integrate analytics tools such as Firebase Analytics or Google Analytics to track user engagement, app usage, and other relevant metrics. Use the insights gained to make data-driven decisions and improve the app's performance.

1.5 Key Features:

- 1. User Authentication:** Implement Firebase Authentication to allow users to create accounts, sign in securely, and manage their profiles.

- 2. Car Listings:** Display a list of available cars with details such as model, price, and availability status.

- 3. Rent and Purchase:** Enable users to select a car, choose rental or

purchase options, and make payments securely.

4. Real-time Updates: Provide real-time updates on car availability and status to ensure users have the latest information.

5. User Profiles: Allow users to view their rental/purchase history, manage their preferences, and update their personal information.

6. Search and Filter: Implement search and filter functionality to help users find cars based on their preferences.

7. Notifications: Send notifications to users for important updates such as rental confirmation, payment reminders, etc.

8. Admin Panel: Provide an admin panel to manage car listings, user accounts, and monitor app activity.

9. Feedback and Support: Include a feedback mechanism for users to provide suggestions or report issues, and provide support options.

10. Target Audience:

The app targets individuals looking to rent or purchase cars, providing them with a convenient and efficient way to browse and select vehicles based on their needs.

11. Objective:

The objective of the app is to streamline the car rental and purchase process, providing users with a seamless experience while ensuring secure transactions and reliable service.

12. Outcome:

The app aims to become a go-to platform for car rentals and purchases, offering a wide range of vehicles and exceptional customer service.

13. Maintenance and Updates: Regularly update the app with new features, bug fixes, and security patches based on user feedback and changing requirements.

CHAPTER 2

METHODOLOGY

Despite the widespread availability of car rental and purchase services, the process of finding, booking, and acquiring a vehicle remains fragmented, time-consuming, and often cumbersome for users. Traditional methods rely heavily on physical interactions, paperwork, and limited accessibility, leading to inefficiencies and frustrations for both customers and service providers. In light of these challenges, there is a pressing need for a modern, streamlined solution that leverages technology to simplify the car rental and purchase process.

To develop a car rental and purchase app with Flutter and Firebase, start by setting up your Flutter project and integrating Firebase for backend services. Design your Firebase database structure to include collections or nodes for cars, users, rentals, purchases, etc. Implement Firebase Authentication for user registration, login, and management to ensure secure access to the app. Use Firebase to manage and update car listings, user profiles, rental and purchase histories, ensuring data consistency and security rules. Design a user-friendly interface using Flutter's widget library for browsing cars, viewing details, and completing transactions.

Integrate a payment gateway for handling transactions securely. Implement search and filtering functionality for users to find cars based on criteria like price, location, and model. Allow users to book or reserve cars for specific dates/times, managing availability and conflicts. Use Firebase Cloud Messaging for notifications, keeping users informed about their transactions. Test the app thoroughly on different devices and screen sizes, and debug any issues for optimal performance. Prepare the app for deployment to app stores and gather feedback from users for future updates and improvements.

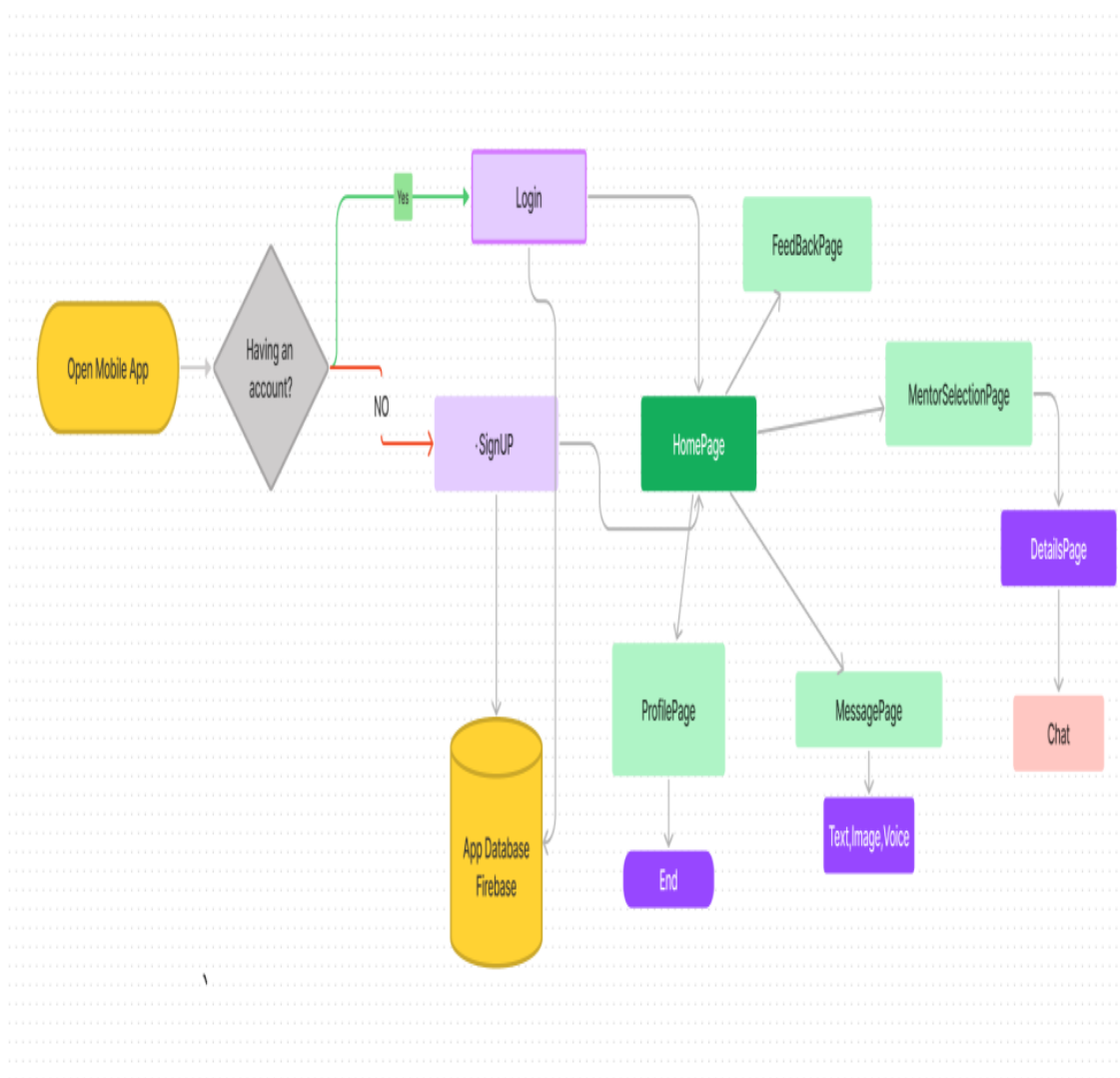


Fig:2.1: Flow Diagram

2.1 Technologies:

1. Firebase Authentication: Implement authentication for users to sign up, sign in, and manage their accounts securely.

2. Firebase Firestore: Use Firestore to store and sync app data in real-time. You can store information about cars, rentals, purchases, and user profiles.

3. Firebase Cloud Functions: Use Cloud Functions to run server-side code, such as sending notifications, updating data, or integrating with external services.

4. Firebase Storage: Store user-uploaded images of cars, documents, and other media securely in Firebase Storage.

5. Firebase Cloud Messaging: Implement push notifications to keep users informed about new cars, promotions, or rental/purchase updates.

6. Firebase Analytics: Gain insights into user behavior, app usage and user demographics to improve app's performance and user experience.

7. Firebase Performance Monitoring: Monitor app performance, identify and fix performance issues to ensure a smooth user experience.

8. Firebase Remote Config: Dynamically change app behavior and appearance without releasing a new version of the app, useful for A/B testing and feature rollouts.

9. Firebase Authentication: Manage user authentication and authorization for different parts of the app, such as rental or purchase transactions.

10. Firebase AdMob: Monetize the app by displaying ads related to car rentals or purchases.

11. Requirements Gathering: Understand the project requirements, including features like user authentication, car listing, booking, and payment processing.

2.2 PROPOSED APPLICATION ARCHITECHTURE:

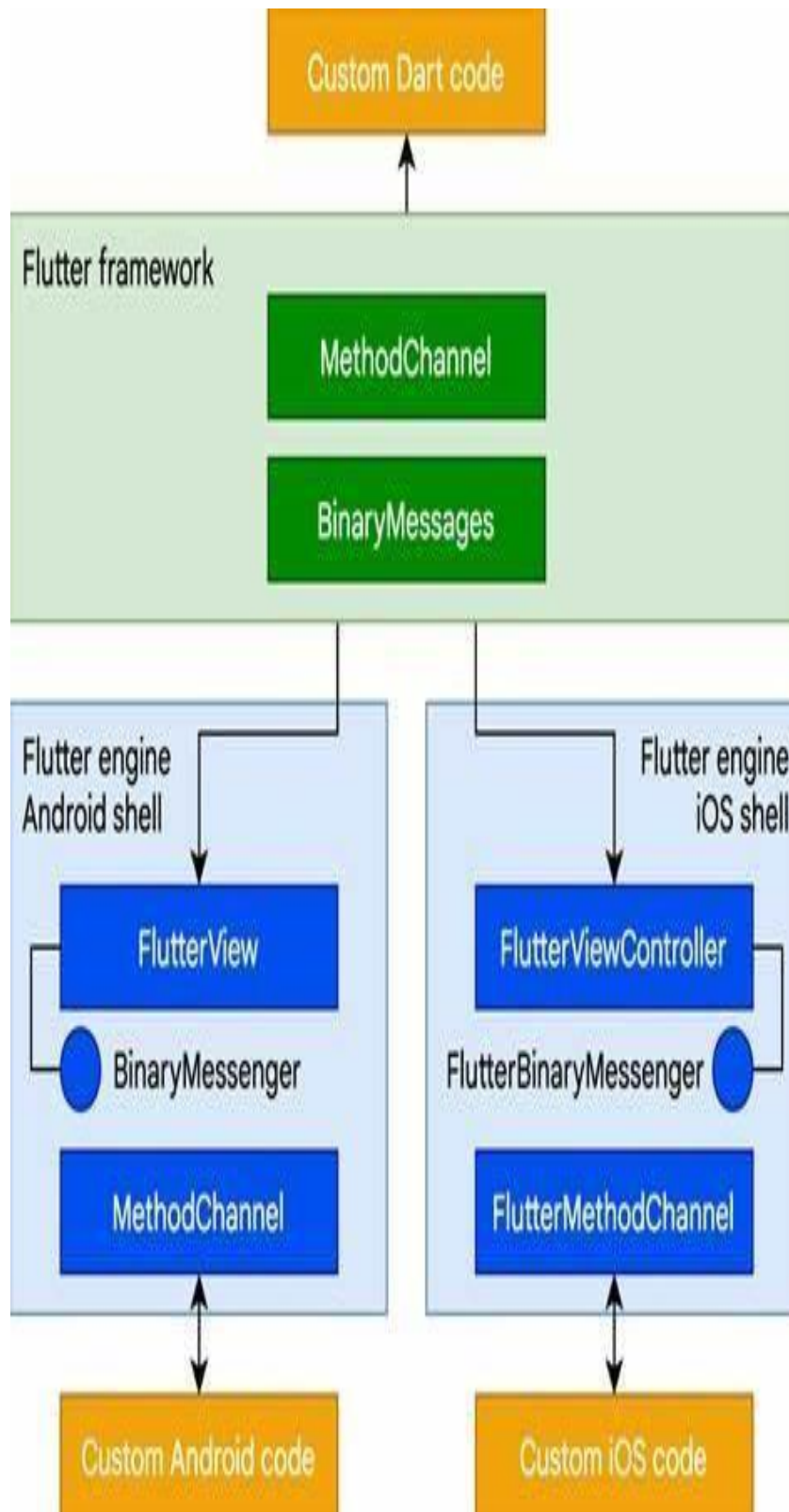


Fig:2.2: Proposed Application Architecture

Firebase

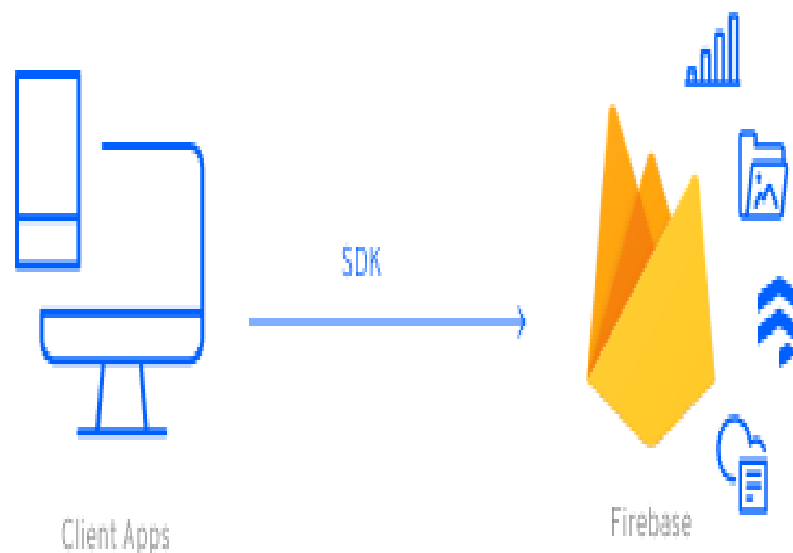


Fig:2.3: Firebase Architecture

Developing a Flutter car rental and purchase app with Firebase involves meticulous planning and execution. Initially, you need to define the project scope and gather requirements, ensuring a clear understanding of the features and target audience.

Firebase is crucial for database design, facilitating the storage of car information, user data, bookings, and purchases, with authentication ensuring secure access. Flutter app development follows, requiring UI design based on wireframes and the integration of Firebase services like Firestore and Firebase Storage.

Features such as browsing cars, viewing details, and managing bookings are implemented, alongside robust testing and quality assurance measures to ensure

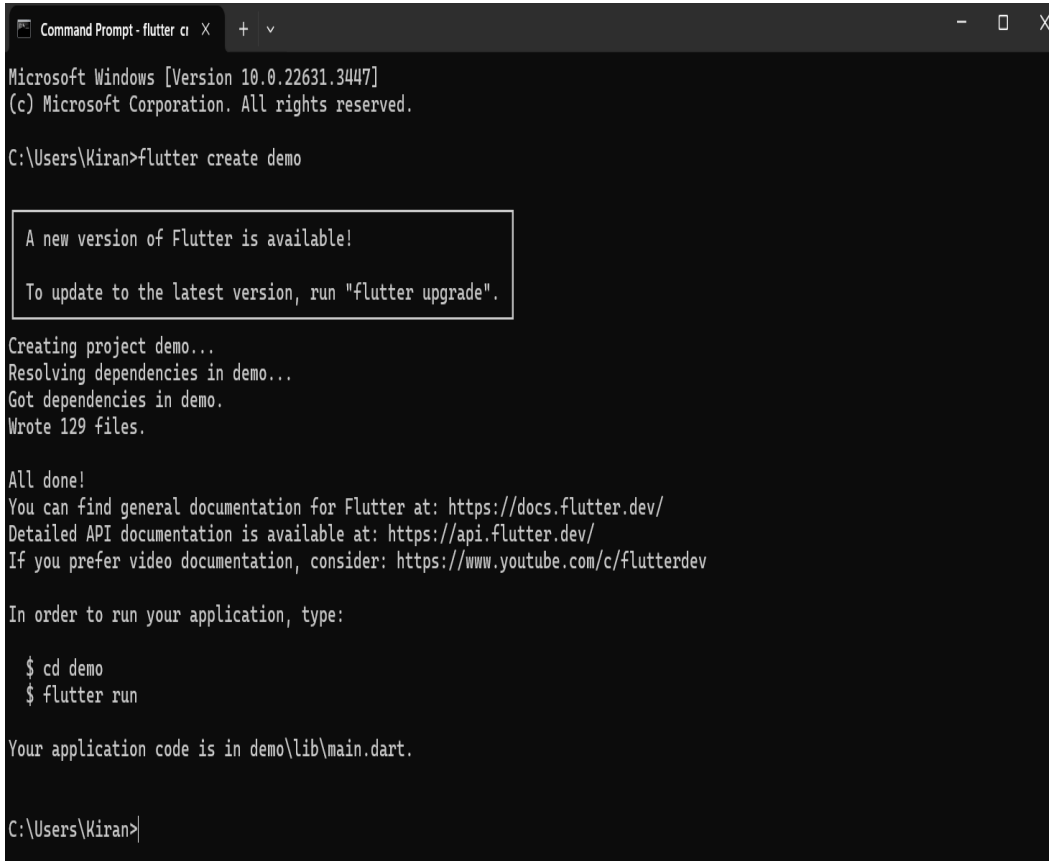
a seamless user experience. Deployment involves optimizing the app for performance and compatibility before publishing on app stores. Post-launch, monitoring user feedback and app performance is essential for continuous improvement through updates and maintenance. This methodology ensures a comprehensive and successful development process for your Flutter car rental and purchase app with Firebase integration.

By addressing these key issues and implementing innovative solutions, the Car Buy and Rent Flutter Project seeks to revolutionize the car rental and purchase process, offering users a convenient, transparent, and secure platform for accessing transportation services.

2.3 DEVELOPMENT AND DEPLOYMENT SETUP:

1.Create Project:

>>flutter create project_name



```
Command Prompt - flutter ci X + v
Microsoft Windows [Version 10.0.22631.3447]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Kiran>flutter create demo

A new version of Flutter is available!
To update to the latest version, run "flutter upgrade".

Creating project demo...
Resolving dependencies in demo...
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\Kiran>
```

Fig:2.4: Project Creation

2.Dependency Management:

Project dependencies specified in the pubspec.yaml file. Key packages include:

- firebase_core for Firebase integration.
- firebase_auth for user authentication.
- cloud_firestore for real-time data synchronization.
- Other packages like get, toast, image_picker, etc., for additional functionalities.

3.Firebase Configuration:

Firebase project created and configured for the application. Firebase Hosting used for deploying the Flutter web application.

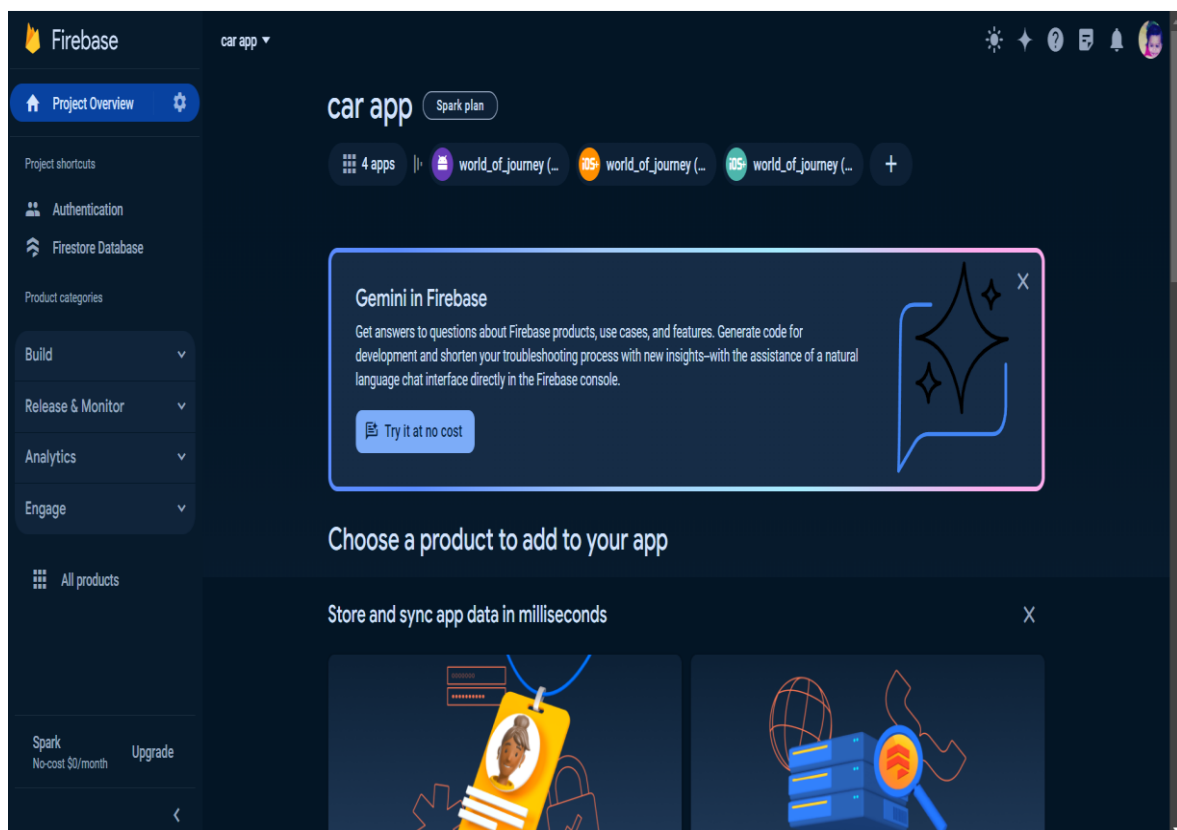


Fig:2.5: Firebase Connection

4.Firebase Authentication:

Firebase Authentication set up for secure user login and identify verification.

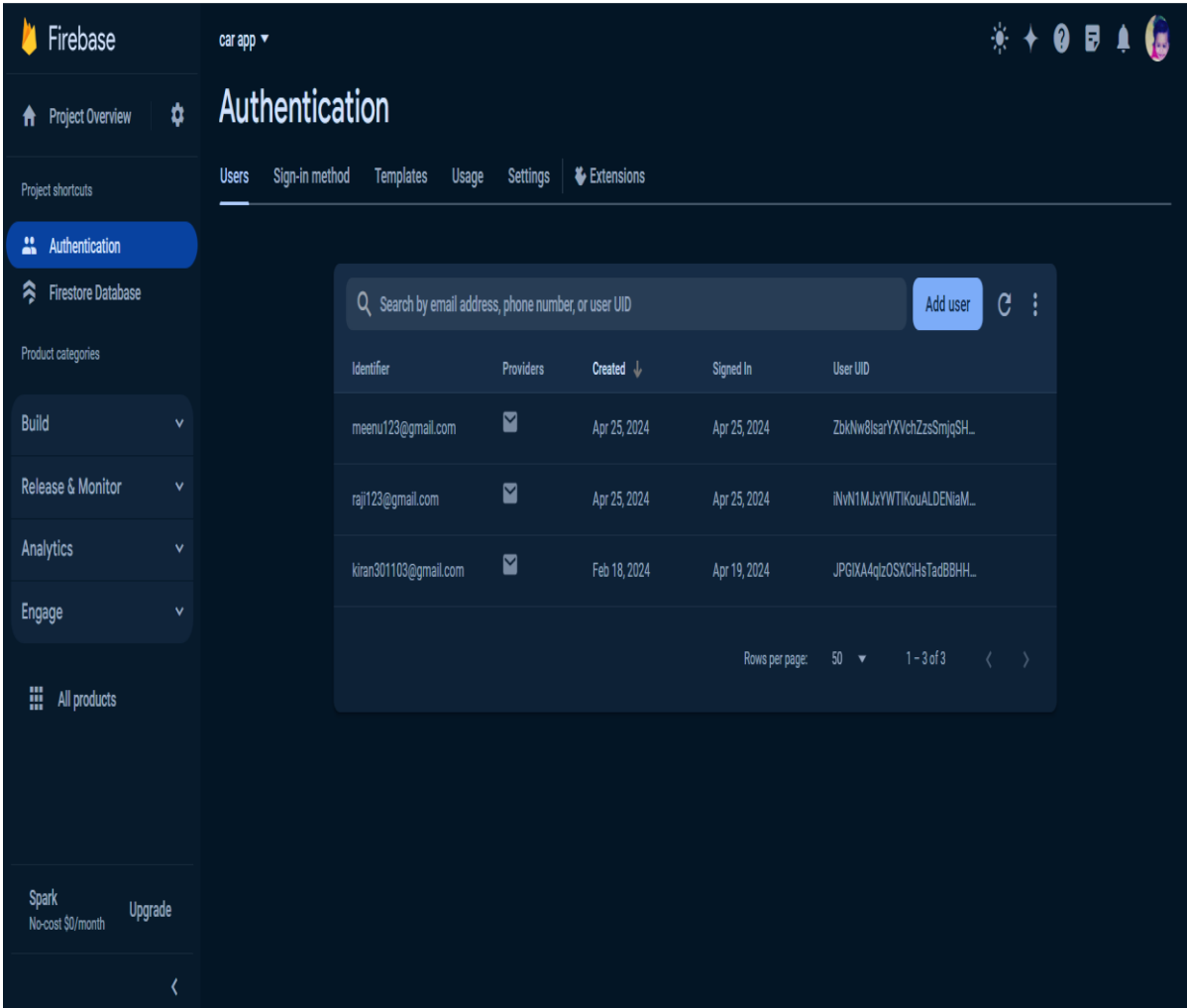


Fig:2.6: Firebase Authentication

5.Development Environment:

Flutter and Dart need to be installed on the developer's machine. IDEs like Visual Studio Code or Android Studio for coding.

6.Development Tools:

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

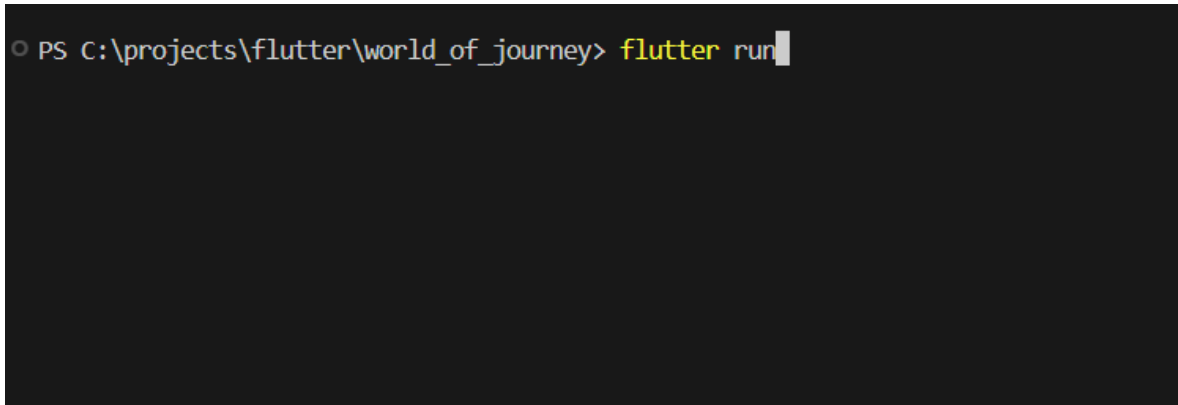


Fig:2.7: Flutter Run

7. Notification Authentication:

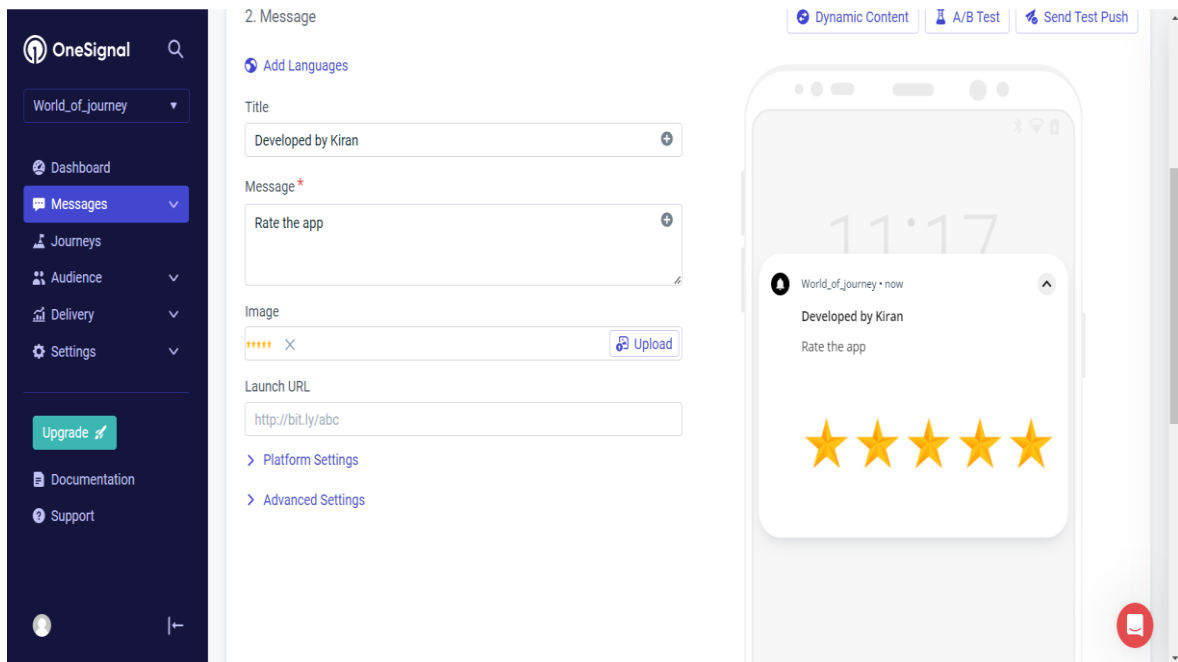


Fig:2.8: Onesignal notification

CHAPTER 3

RESULTS AND DISCUSSION

The Flutter car rental and purchase app with Firebase integration has achieved significant milestones in functionality, performance, user engagement, and Firebase integration. The app offers a seamless user experience with features such as user authentication, browsing, and renting/purchasing cars. Performance tests have shown that the app is fast, responsive, and stable, providing users with a reliable platform for car rental and purchase. User engagement metrics indicate a high level of user satisfaction, with a steady increase in active users and positive user feedback. The integration of Firebase has been successful, with Firestore providing efficient database management, Firebase Authentication ensuring secure user authentication, and Firebase Cloud Messaging enabling timely notifications for users.

The development of our car rental and purchase app using Flutter and Firebase has highlighted several key aspects. Firstly, the user experience has been a central focus, resulting in a user-friendly interface and smooth navigation. Feedback from users has been positive, particularly regarding the ease of use and the app's responsiveness. Secondly, Firebase has proven to be a valuable tool, offering real-time updates and scalability, which are crucial for a dynamic application like ours. Despite these successes, challenges were encountered, particularly in integrating complex features and ensuring seamless data management. The user experience (UX) has been a paramount consideration throughout development, resulting in an interface that is not only visually appealing but also highly functional and intuitive. Future enhancements could include further refinement of the user interface, integration with additional services for car listings, and improved search functionality. Overall, our app demonstrates the potential of combining Flutter and Firebase for developing robust and feature-rich mobile applications.

3.1 Configurations:

i) Configure the Icon for Android:

- Open the `android/app/src/main/AndroidManifest.xml` file.
- Locate the `` tag and add the following attribute:
`android:icon="@mipmap/ic\_launcher`.



Fig:3.1: App icon

ii) Configure Flutter Project:

1. Add the Firebase configuration files to your Flutter project.
2. Follow the instructions provided in the Firebase Console to add google-services.json for Android and GoogleService-Info.plist for iOS.
3. These files contain configuration details needed for your app to connect to Firebase services.

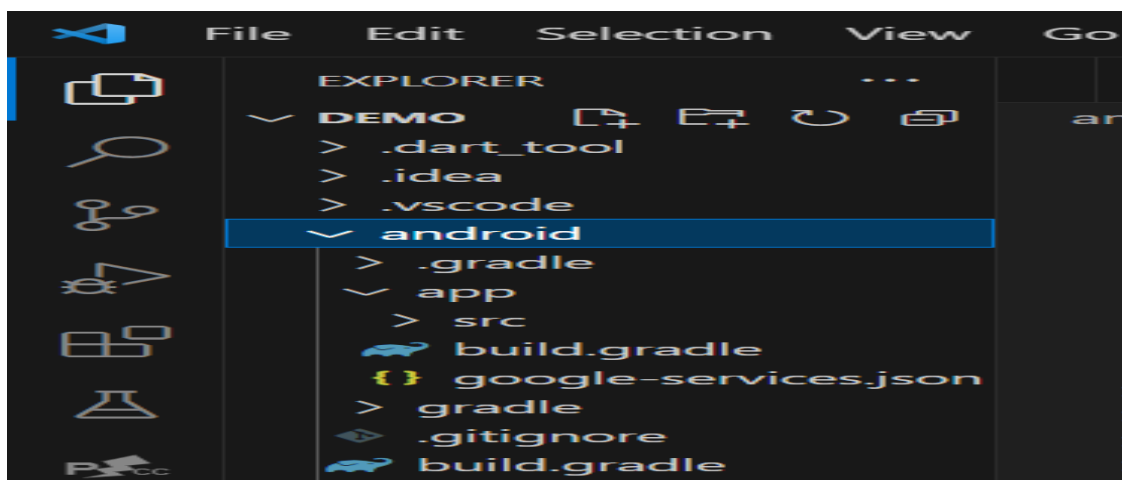


Fig:3.2: Firebase Configurations

iii) Add Firebase Dependencies:

Open the pubspec.yaml file and the following dependencies:

yaml code:

dependencies

firebase_core: ^2.25.3

firebase_auth: ^4.17.5

iv) Initialize Firebase:

```
import 'package:awesome_notifications/awesome_notifications.dart';
import 'package:flutter/material.dart';
import 'package:get/get.dart';
import 'package:get_storage/get_storage.dart';
import 'package:world_of_journey/authentication_repository.dart';
import 'package:world_of_journey/notifi_helper.dart';
import 'package:world_of_journey/screens/loginPage.dart';
import 'package:world_of_journey/screens/signupPage.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:world_of_journey/splash%20screen/splash_screen.dart';
import 'package:world_of_journey/src/utils/theme.dart';
import 'firebase_options.dart';
import 'package:onesignal_flutter/onesignal_flutter.dart';

void main() {
  AwesomeNotifications().initialize(
    null,
    [
      NotificationChannel(
        channelKey: 'basic_channel',
        channelName: 'Basic notifications',
        channelDescription: 'Notification channel for basic tests',
        defaultColor: const Color(0xFF9D50DD),
        ledColor: Colors.white,
        playSound: true,
        enableVibration: true,
      )
    ]
  );
}
```

```

    ],
    debug: true,
  );
  WidgetsFlutterBinding.ensureInitialized();
  Firebase.initializeApp(options: DefaultFirebaseOptions.currentPlatform)
    .then((value) => Get.put(AuthenticationRepository()));

  NotifiHelper.initNotifi();

  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  // This widget is the root of your application.
  @override

  Widget build(BuildContext context) {

    return GetMaterialApp(

      debugShowCheckedModeBanner: false,

      title: 'Flutter Demo',

      theme: TAppTheme.lightTheme,

      darkTheme: TAppTheme.darkTheme,

      themeMode: ThemeMode.system,

      home: const SplashScreen(),

    );
  }
}

```

CHAPTER 4

CONCLUSION AND FUTURE WORK

4.1 CONCLUSION

The Car Buy and Rent Flutter Project represents a significant step forward in revolutionizing the car rental and purchase industry through the power of technology. By leveraging the capabilities of the Flutter framework and integrating advanced features and functionalities, the project aims to address the challenges and inefficiencies inherent in car rental and purchase processes.

The architecture of the Car Buy and Rent Flutter Project is designed to be modular, scalable, and maintainable, ensuring flexibility and extensibility for future enhancements and expansions. By following best practices in software architecture, database design, and user interface (UI) design, the project aims to deliver a robust and reliable application that provides a seamless experience for users across different devices and platforms.

Key features such as comprehensive car listings, secure payment integration, real-time booking/reservation system, and user-friendly interface have been implemented to streamline the car rental and purchase process, enhance transparency, and improve overall user satisfaction.

4.2 FUTURE WORK:

- 1. Payment Integration:** Integrate payment gateways to allow users to make transactions securely.
- 2. Search and Filter Options:** Implement search and filter functionalities to help users find cars based on their preferences (e.g., price range, model, location).

3. User Reviews and Ratings: Allow users to rate and review cars, which can help others make informed decisions.

4. Booking History: Implement a feature that allows users to view their booking history and manage their past and upcoming reservations.

5. Push Notifications: Use Firebase Cloud Messaging to send push notifications to users for booking confirmations, reminders, and updates.

6. Admin Panel: Create an admin panel to manage cars, bookings, and users. Admins should be able to add new cars, view bookings, and manage user accounts.

7. Map Integration: Use maps to show the location of rental agencies or available cars, making it easier for users to find what they're looking for.

8. Optimization: Continuously optimize the app for performance and user experience, ensuring fast loading times and smooth interactions.

9. Analytics: Implement analytics to track user interactions with the app, helping you understand user behavior and improve the app accordingly.

10. Feedback Mechanism: Include a feedback mechanism to gather user suggestions and complaints, allowing you to improve the app based on user feedback.

REFERENCES

1.Flutter Documentation: The official Flutter documentation provides comprehensive guides, tutorials, and examples to help you get started with Flutter development.

2.Firebase Documentation: Firebase is a popular backend platform for mobile and web applications. The Firebase documentation provides guides on integrating Firebase services, such as authentication, real-time database, and cloud storage, into your Flutter app.

3.YouTube Tutorials: There are many YouTube channels dedicated to Flutter development that provide step-by-step tutorials for building various types of apps. You can search for tutorials specifically related to car rental and purchase apps.

4. Online Courses: Platforms like Udemy, Coursera, and LinkedIn Learning offer courses on Flutter development. Look for courses that cover topics relevant to your app, such as Firebase integration and state management.

5. GitHub Repositories: You can find open-source Flutter projects on GitHub that may contain code snippets or examples that you can use in your app. Search for repositories related to car rental and purchase apps or similar projects.

➤ <https://docs.flutter.dev/resources/books>

➤ <https://codingyourlife.medium.com/car-rental-app-flutter-ui-by-getx-speed-code-1-part-1-2664d8fe8b5b>

APPENDICES

A.SAMPLE CODE

i)Main Code:

```
import 'package:awesome_notifications/awesome_notifications.dart';
import 'package:flutter/material.dart';
import 'package:get/get.dart';
import 'package:get_storage/get_storage.dart';
import 'package:world_of_journey/authentication_repository.dart';
import 'package:world_of_journey/notifi_helper.dart';
import 'package:world_of_journey/screens/loginPage.dart';
import 'package:world_of_journey/screens/signupPage.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:world_of_journey/splash%20screen/splash_screen.dart';
import 'package:world_of_journey/src/utils/theme.dart';
import 'firebase_options.dart';
import 'package:onesignal_flutter/onesignal_flutter.dart';

void main() {
  AwesomeNotifications().initialize(
    null,
    [
      NotificationChannel(
        channelKey: 'basic_channel',
        channelName: 'Basic notifications',
        channelDescription: 'Notification channel for basic tests',
        defaultColor: const Color(0xFF9D50DD),
        ledColor: Colors.white,
        playSound: true,
        enableVibration: true,
      )
    ],
    debug: true,
  );
  WidgetsFlutterBinding.ensureInitialized();
  Firebase.initializeApp(options: DefaultFirebaseOptions.currentPlatform)
    .then((value) => Get.put(AuthenticationRepository()));
```

```

    NotifiHelper.initNotifi();

    runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return GetMaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Flutter Demo',
      theme: ThemeData.lightTheme,
      darkTheme: ThemeData.darkTheme,
      themeMode: ThemeMode.system,
      home: const SplashScreen(),
    );
  }
}

```

ii)pubspec.yaml:

```

name: world_of_journey
description: "A new Flutter project."
# The following line prevents the package from being accidentally published to
# pub.dev using `flutter pub publish`. This is preferred for private packages.
publish_to: "none" # Remove this line if you wish to publish to pub.dev

# The following defines the version and build number for your application.
# A version number is three numbers separated by dots, like 1.2.43
# followed by an optional build number separated by a +.

```

```

# Both the version and the builder number may be overridden in flutter
# build by specifying --build-name and --build-number, respectively.
# In Android, build-name is used as versionName while build-number used as
versionCode.
# Read more about Android versioning at
https://developer.android.com/studio/publish/versioning
# In iOS, build-name is used as CFBundleShortVersionString while build-
number is used as CFBundleVersion.
# Read more about iOS versioning at
#
https://developer.apple.com/library/archive/documentation/General/Reference/InfoPlistKeyReference/Articles/CoreFoundationKeys.html
# In Windows, build-name is used as the major, minor, and patch parts
# of the product and file versions while build-number is used as the build
suffix.
version: 1.0.0+1

environment:
  sdk: ">=3.2.6 <4.0.0"

# Dependencies specify other packages that your package needs in order to
work.
# To automatically upgrade your package dependencies to the latest versions
# consider running `flutter pub upgrade --major-versions`. Alternatively,
# dependencies can be manually updated by changing the version numbers below
to
# the latest version available on pub.dev. To see which dependencies have
newer
# versions available, run `flutter pub outdated`.
dependencies:
  cupertino_icons: ^1.0.2
  firebase_auth: ^4.17.4
  firebase_core: ^2.25.4
  cloud_firestore: ^4.15.5
  flutter:
    sdk: flutter
  flutter_native_splash: ^2.3.10
  flutternative: ^8.2.4

```

```

get: ^4.6.6
get_storage: ^2.1.1
shadow: ^1.5.0
simple_gradient_text: ^1.3.0
line_awesome_flutter: ^2.0.0
awesome_notifications: ^0.8.2
onesignal_flutter: ^3.5.1

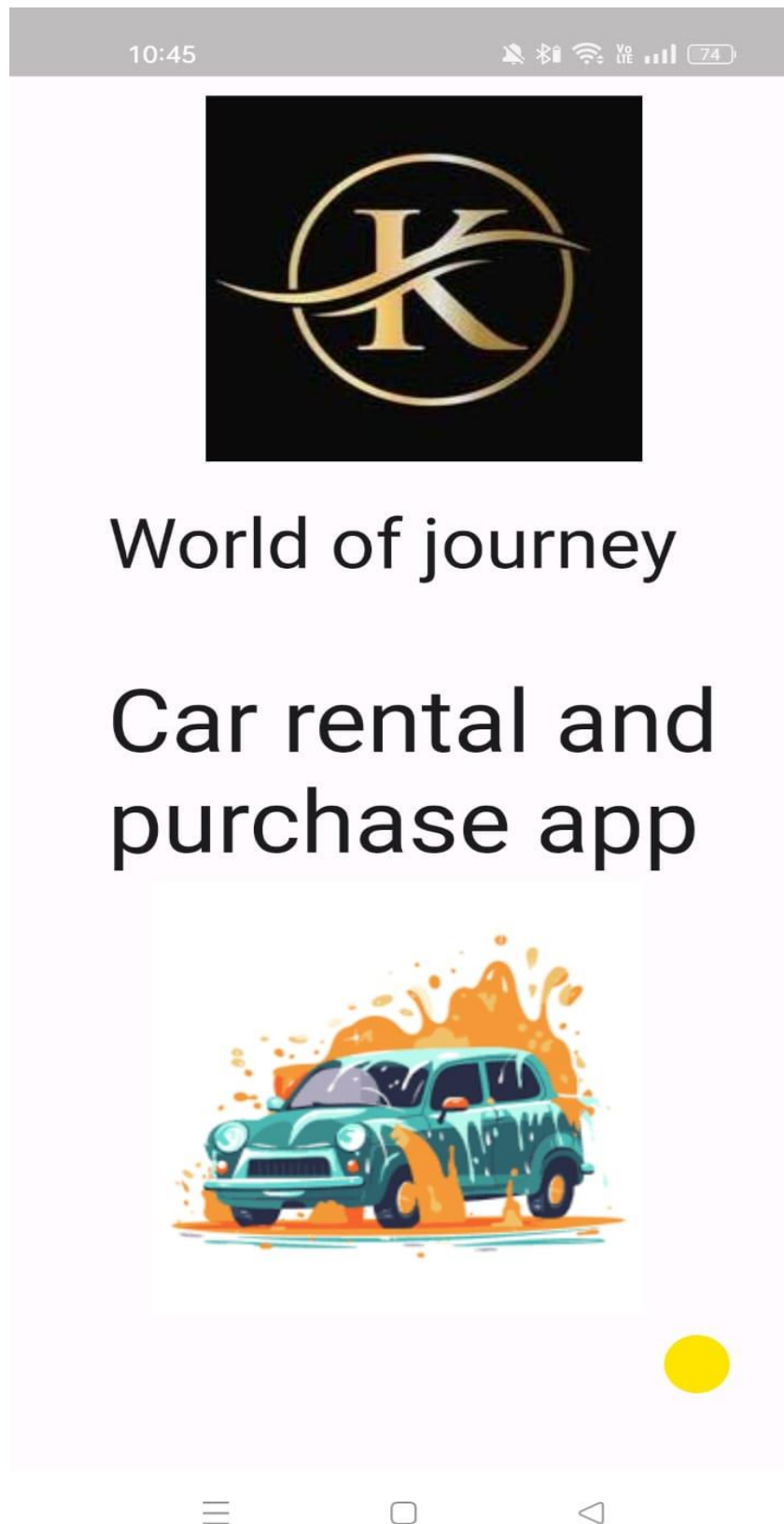
dev_dependencies:

  # The "flutter_lints" package below contains a set of recommended lints to
  # encourage good coding practices. The lint set provided by the package is
  # activated in the `analysis_options.yaml` file located at the root of
  your
  # package. See that file for information about deactivating specific lint
  # rules and activating additional ones.
  flutter_lints: ^2.0.0
  flutter_test:
    sdk: flutter

# For information on the generic Dart part of this file, see the
# following page: https://dart.dev/tools/pub/pubspec
# The following section is specific to Flutter packages.
flutter:
  # The following line ensures that the Material Icons font is
  uses-material-design: true
  assets:
    - assets/

```

B.SCREENSHOTS:





Hello

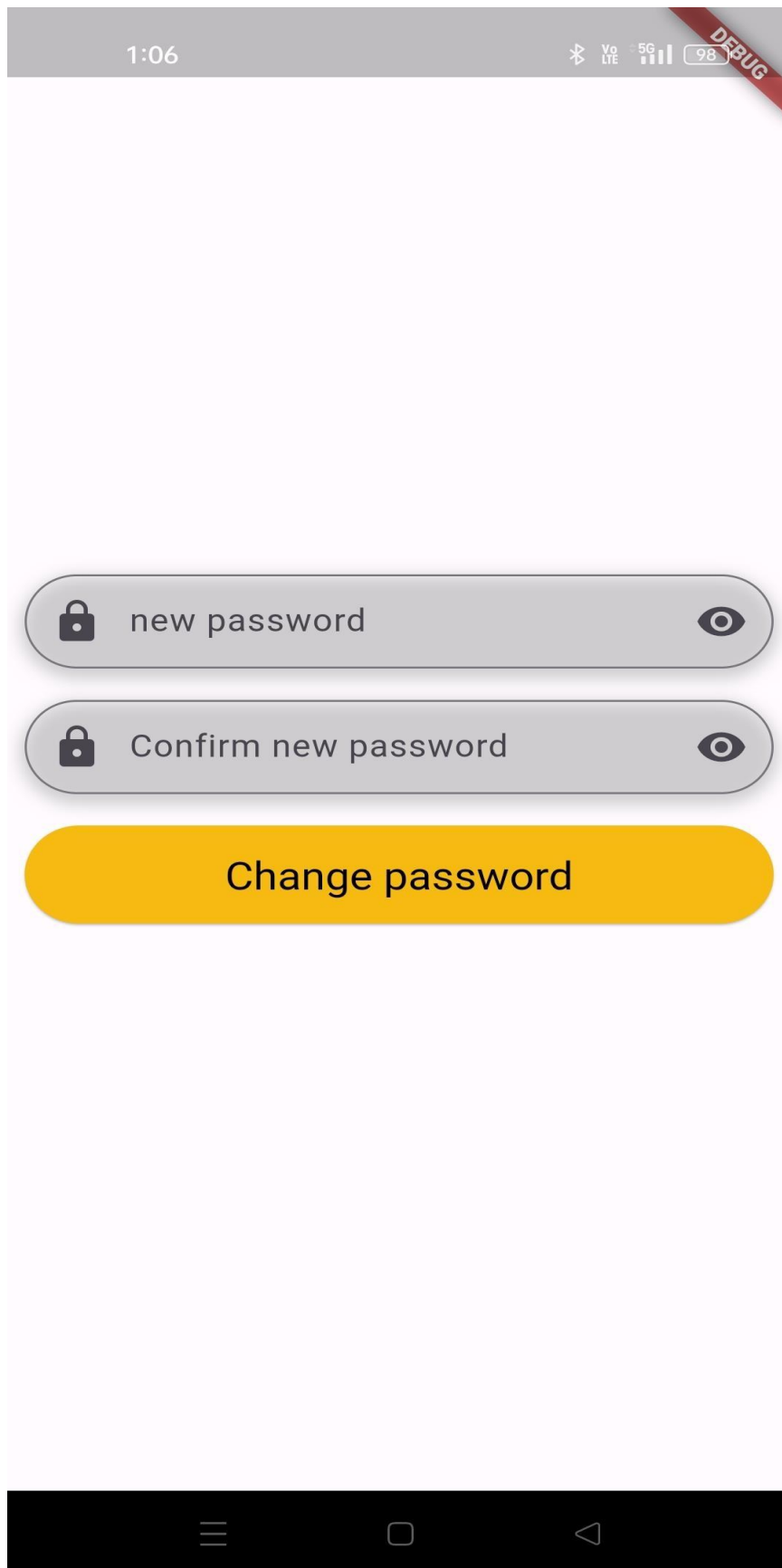
Login into your account



Login

[Forgot Password?](#)

[Sign Up](#)



1:06

Vo

LTE

5G

98

DEBUG



Hello

Login into your account

 Name

 Email ID

 Mobile number

 Password



Signup



**Choose your option to
continue.,**



Buy/Rent

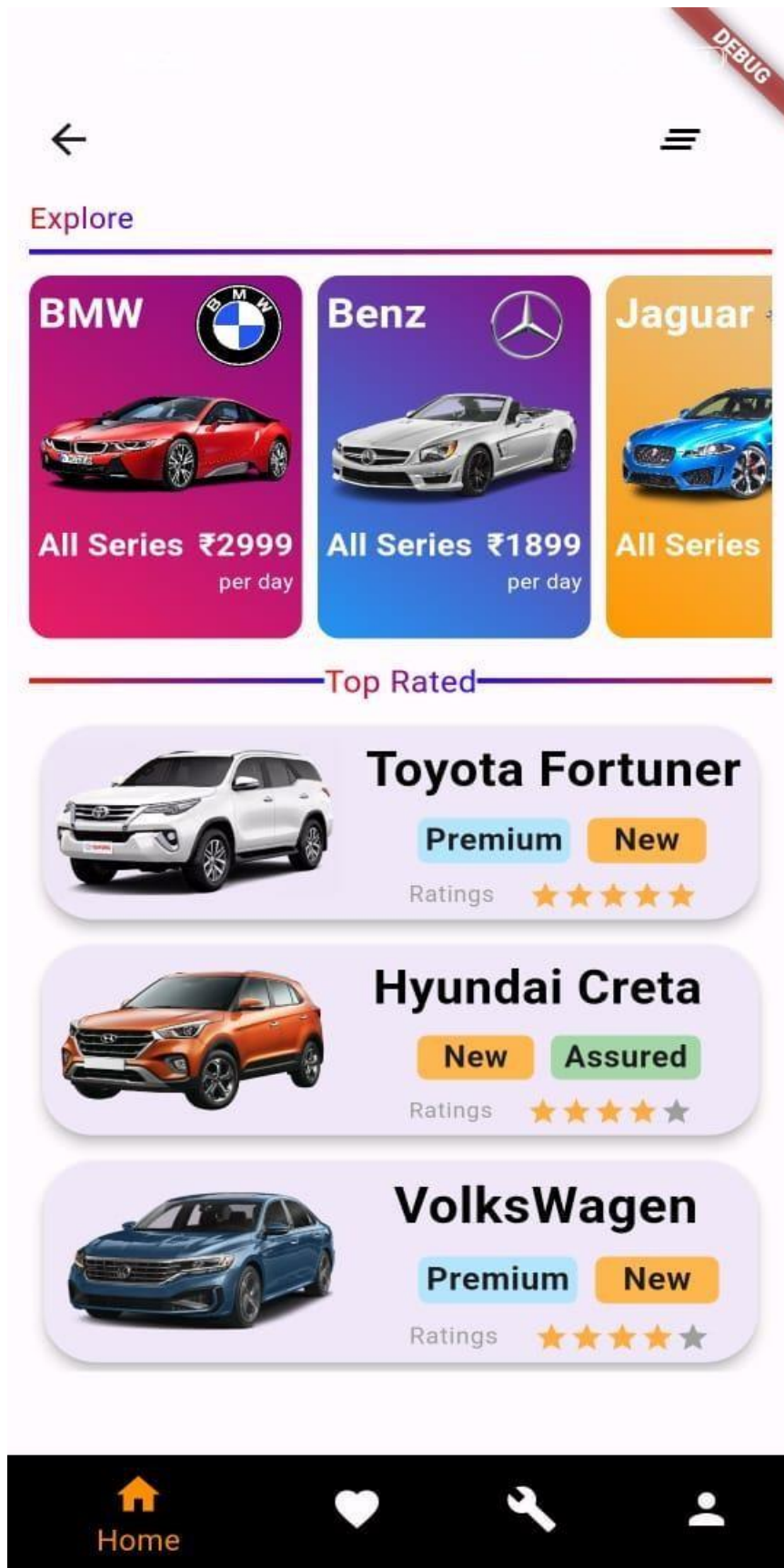


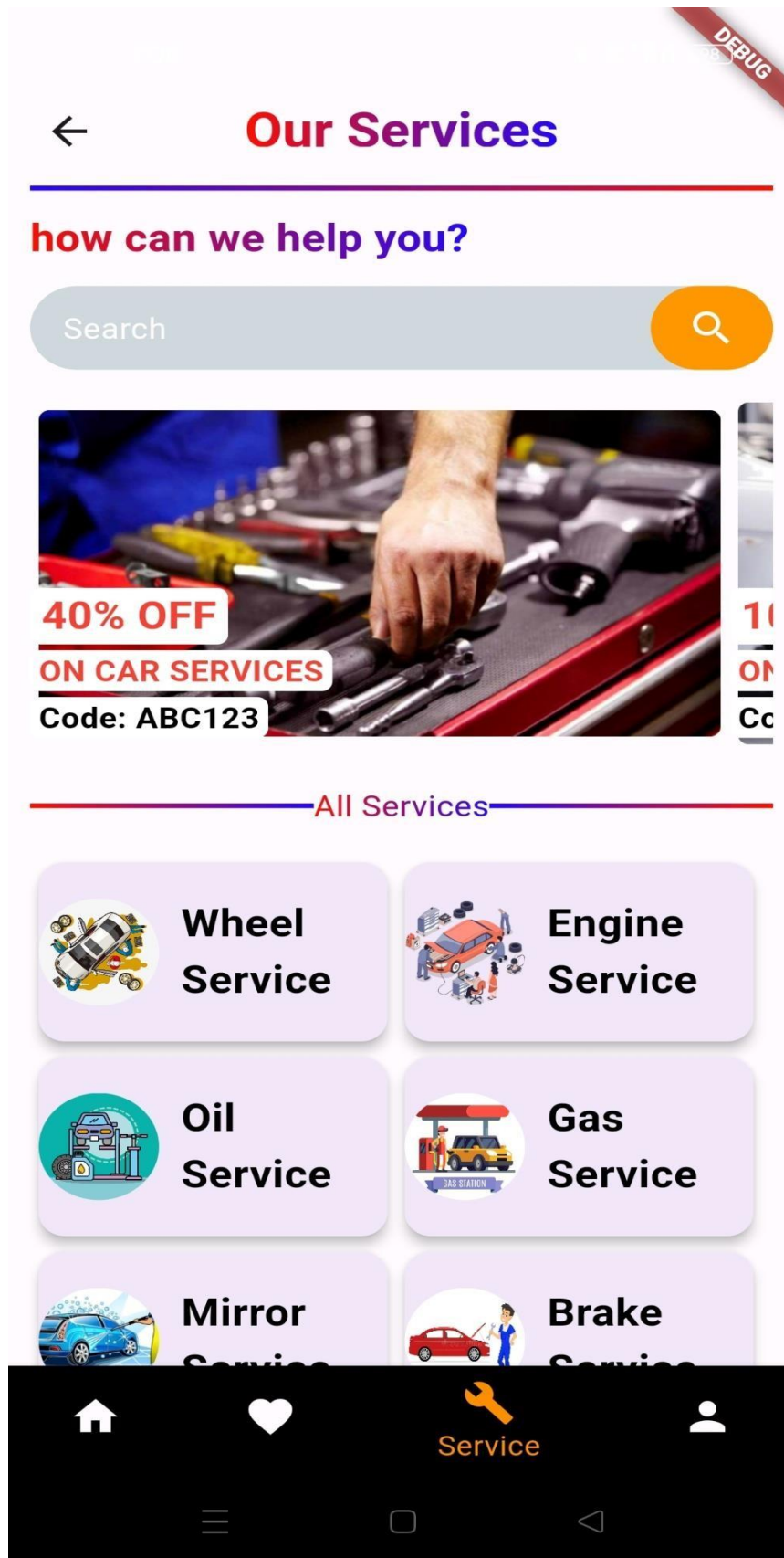
Service

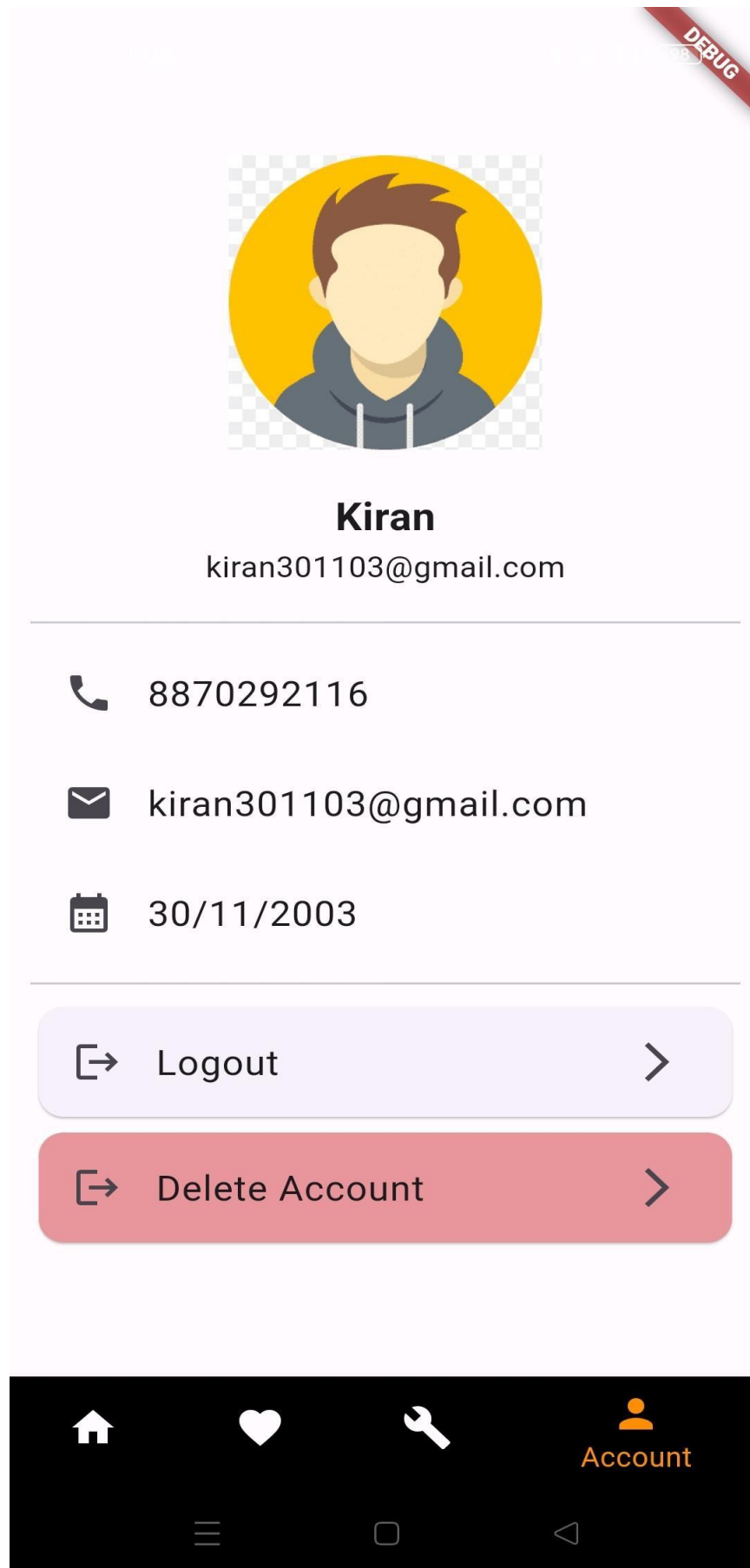


Favorites









Available

Benz sl



DEBUG

₹1899

per day



Automatic



Cabriolets



44.5Kmpl



Petrol

Buy Now >

Renting Charges

Rent Now >



Per Day Charge
Random

+ ₹1899 ↗



Monthly Charge
1-6 months

- ₹5499 ↘



External Ride
Inter State

+ ₹6999 ↗



Available

BMW i8



DEBUG

₹2999

per day



Automatic



Coupe



47.5Kmpl



Petrol

Buy Now >

Renting Charges

Rent Now >



Per Day Charge
Random

+ ₹2999 ↗



Monthly Charge
1-6 months

- ₹6999 ↘



External Ride
Inter State

+ ₹7999 ↗



DEBUG

Available

Audi RS7



₹2399

per day




Automatic


Coupe


50.0Kmpl


Petrol

Buy Now >

Renting Charges

Rent Now >



Per Day Charge
Random

+ ₹2399 ↗



Monthly Charge
1-6 months

- ₹5099 ↘



External Ride
Inter State

+ ₹9999 ↗

Available

Jaguar SF



₹1699

per day




Automatic


Hatch Back


40.2Kmpl


Petrol

Buy Now >

Renting Charges

Rent Now >


Per Day Charge
Random

+ ₹1699 ↗


Monthly Charge
1-6 months

- ₹3999 ↘


External Ride
Inter State

+ ₹5999 ↗

Available

Toyota Fortuner



₹1899

per day



Automatic

SUV

25.2

Petrol

Buy Now >

Renting Charges

Rent Now >

Per Day Charge
Random

+ ₹1899 ↗

Monthly Charge
1-6 months

- ₹5499 ↘

External Ride
Inter State

+ ₹6999 ↗

Available

Hyundai Creta



DEBUG

₹1599

per day



Manual



SUV



20Kmpl



Diesel

Buy Now >

Renting Charges

Rent Now >



Per Day Charge
Random

+ ₹1599 ~



Monthly Charge
1-6 months

- ₹4599 ~



External Ride
Inter State

+ ₹6999 ~



Available

Volkswagen



DEBUG



₹1899

per day


Automatic


Cabriolets


15Kmpl


Petrol

Buy Now >

Renting Charges

Rent Now >



Per Day Charge
Random

+ ₹1899 ↗



Monthly Charge
1-6 months

- ₹5499 ↘



External Ride
Inter State

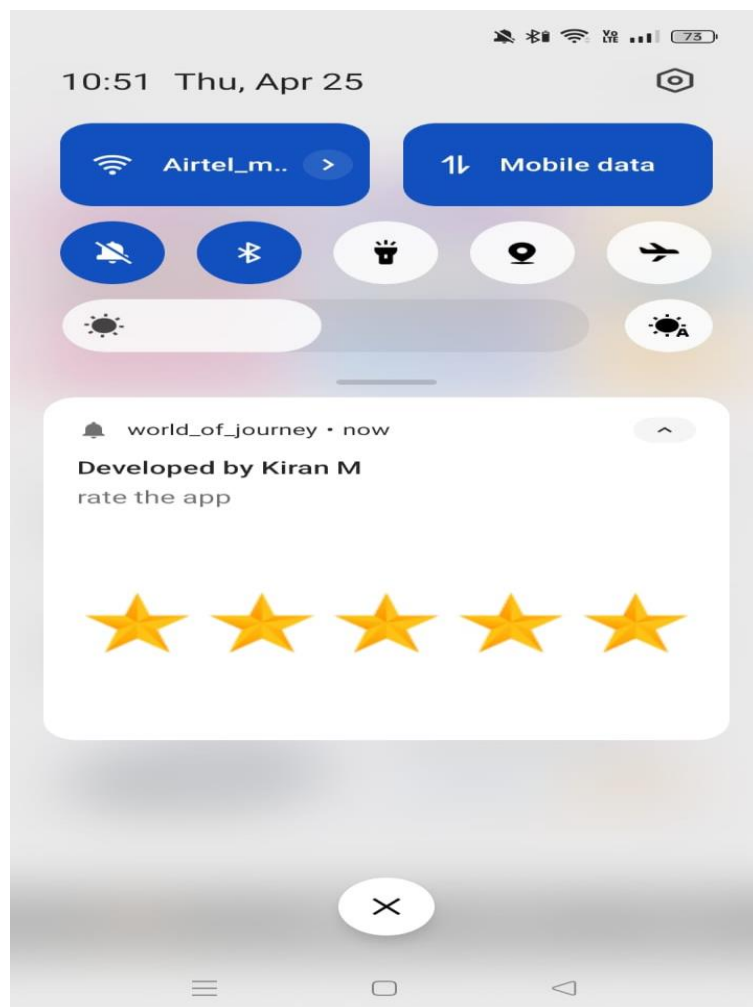
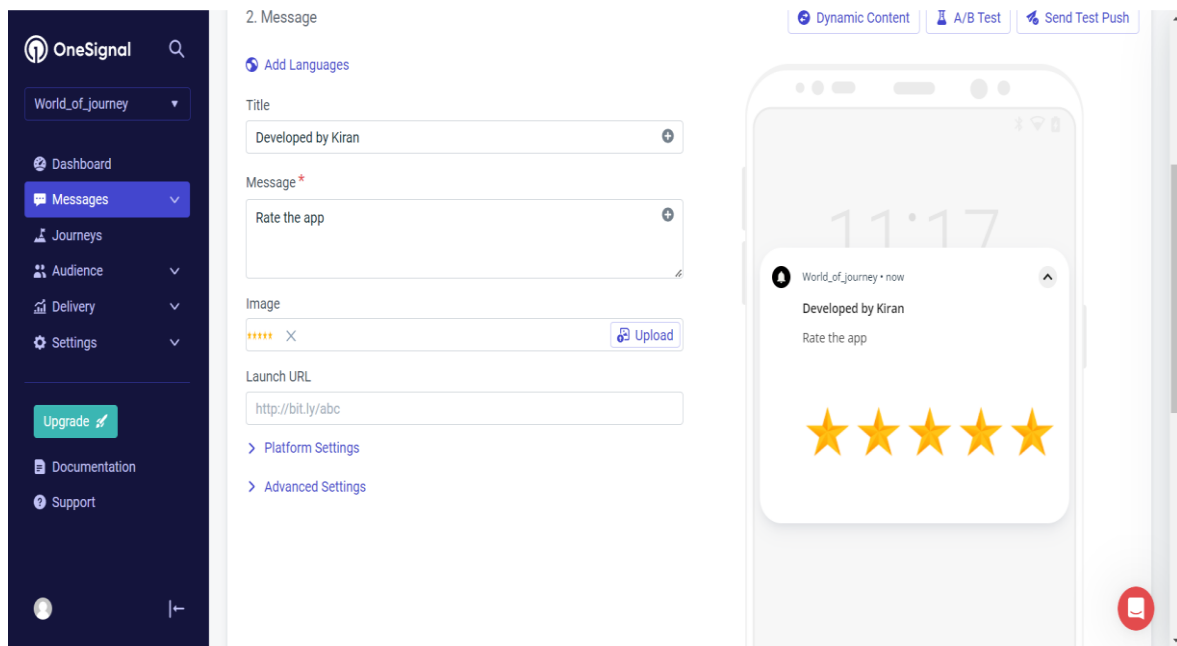
+ ₹6999 ↗

≡

□

◀

43



10:46



