ParkGSU - Parking Application for Mobile Users

Made by ParkNet Digital

Members: Kawsar Ali, Luv Dabhi, Jaylen Hill, Alex Nyugen, Avyuktkrishna Ramasamy

This is a Flutter project developed in Dart, designed to facilitate parking reservations, loyalty point tracking, and local event discovery. It integrates with Firebase for user authentication, Firestore database, and several other Flutter packages for enhanced functionality.

Table of Contents

Getting Started

To set up and run this project, follow the steps below.

1. Prerequisites - Ensure you have the following installed:
    a. Flutter 3.24.4
    b. Dart 3.5.4
    c. Firebase project setup (for authentication and Firestore)
    d. Android/iOS device or emulator to run the app

2. Clone the Repository
    Clone the project to your local machine using the following command:
    `git clone https://github.com/KrishnaDOS/ParkNet_Digital.git`
    Alternatively, if you'd like to set up the project from a zip file, use the zip file provided with the submission. Or, if you're downloading a zip file through the git repo, do the following:
    a. Visit the GitHub repository (https://github.com/KrishnaDOS/ParkNet_Digital).

b. Click the "Code" button and select "Download ZIP".
c. Extract the contents of the zip file to your local machine.

3. Install Dependencies

   Navigate to the project directory and run the below command to install the dependencies:

   ```
   flutter pub get
   ```

OR

If you are not using the repo or the zip file directly to test the app you can open the zip and open the apk with a valid apk opener like bluestacks etc.

Dependencies

The project uses the following dependencies:

- flutter: SDK for Flutter applications
- qr_flutter: ^4.0.0 ; A Flutter plugin for generating QR codes
- firebase_core: ^3.3.0 ; Core package for initializing Firebase in Flutter
- firebase_auth: ^5.3.1 ; Firebase authentication for signing in users
- cloud_firestore: ^5.2.1 ; Firestore database for storing user and reservation data
- geolocator: ^13.0.1 ; Provides geolocation services (latitude/longitude)
- flutter_map: ^7.0.2 ; A Flutter map widget based on Leaflet
- latlong2: ^0.9.1 ; Provides latitude and longitude functionalities
- geocoding: ^2.0.4 ; Geocoding functionality (convert coordinates to addresses)
- webview_flutter: ^4.0.0 ; A Flutter plugin for displaying web content
- cupertino_icons: ^1.0.8 ; Provides iOS style icons for your app
- mockito: ^5.4.4 ; A mocking framework for Dart, used for unit testing
- build_runner: ^2.4.13 ; A tool for running code generation tasks (required for some dependencies)

Key Features

- Parking Reservations: Users can make and manage their parking reservations.
  - Nearest Parking Deck: Users can reserve spots based on their current location
  - Specific Parking Deck: Users can reserve spots at a specific location

- Loyalty Points System: Users get loyalty points each time they reserve a spot.
- Geolocation and Maps: Users can see their location & nearby parking locations on a map.
- Local Event Finder: Users can find parking near local events.
- Customer Support: Users can put in requests/tickets for customer support.

Configuration

1. Firebase Setup for Flutter
    a. Android:
    In your `android/build.gradle`, add the following inside `dependencies`:
    ```
    groovy classpath
    'com.google.gms:google-services:4.3.15'
    ```
    In your `android/app/build.gradle`, add this line at the bottom:
    ```
    groovy apply plugin: 'com.google.gms.google-services'
    ```
    b. iOS:
    Ensure you have `CocoaPods` installed (`brew install cocoapods`), and run:
    ```
    bash cd ios && pod install && cd ..
    ```
2. Geolocation Permissions
    For Android and iOS, you'll need to set up location permissions in their respective configuration files:

a. Android (`android/app/src/main/AndroidManifest.xml`):
```
<uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION"/>
```

b. iOS (`ios/Runner/Info.plist`):

```
<key>NSLocationWhenInUseUsageDescription</key>
<string>We need your location to find nearby parking.</string>
```

How to Run the Project

1. Make sure all dependencies are installed using `flutter pub get`.
2. Connect a device or start an emulator.
3. Run the project using the following command: `bash flutter run`
4. The app should now be running on your emulator or connected device.

Troubleshooting

- Firebase Connection Issues: Double-check that you have correctly set up the Firebase project and added the configuration files (`google-services.json` or `GoogleService-Info.plist`).
- Geolocation Permissions: If location-based features are not working, ensure the app has the correct permissions for accessing location data (both in the app and on the device).