Implementation Document

for

PedalPal

Version 1.0

Prepared by

Group # 4 **Group Name: Bit Brewers**

Raghav Manglik	220854	raghavkmanglik@gmail.com
Amogh Bhagwat	220288	amogh.2004b@gmail.com
Srishti Chandra	221088	chandra.srishti2403@gmail.com
Wadkar Srujan Nitin	221212	srujanwadkar@gmail.com
Anaswar K B	220138	anaswarkb013@gmail.com
Khushi Gupta	220531	khushi07g@gmail.com
Ananya Singh Baghel	220136	ananyabaghel2004@gmail.com
Pathe Nevish Ashok	220757	nevu.pathe1234@gmail.com
Debraj Karmakar	220329	debraj2003jsr@gmail.com
Kaneez Fatima	220496	kaneezfatimamehdi7@gmail.com

Course:	CS253
Mentor TA:	Mr. Bharat
Instructor:	Prof. Indranil Saha
Date:	March 18, 2024

Contents

1. Revisions

Version	Primary Author(s)	Description of Version	Date Completed
v1.0	Raghav Manglik Amogh Bhagwat Srishti Chandra Wadkar Srujan Nitin Pathe Nevish Ashok Debraj Karmakar Khushi Gupta Ananya Baghel Anaswar K B Kaneez Fatima	First version of the Software Implementation Document	18/03/2024

2. Implementation Details

Our App, ***PedalPal*** has *three major* components :

2.1 Backend

1. **Programming Language:** PedalPal's backend is built using **Python**. Python's simplicity, readability, and rich library support make it a highly flexible choice for development work.
2. **Framework:** We have used **Django framework**. We chose Django because we believe it to be robust yet simple, reliable and scalable. Some other advantages of Django which makes it a popular choice are:
   - **Simple Sytnax:** Django is written in Python.
   - **Security:** Django offers robust user authentication functionality.
   - **Built-in Admin Interface:** Django includes an admin interface to quickly create, read, update, and delete data without writing additional code.
   - **Object-Relational Mapping:** Django's ORM simplifies database interactions by abstracting SQL queries into Python code, reducing efforts and increasing productivity.
   - **Rapid Development:** Django supports rapid development, like its ”batteries-included” philosophy, saves time and effort, making development faster and more efficient.
   - **Community Support:** Django has a large and active community of developers, providing extensive resources, support, and third-party packages.
   - **Automatic URL Routing:** Django automatically handles URL routing, simplifying the mapping of URLs to views and reducing manual configuration overhead.
   - **Modularity of Code:** Django's modularity allows developers to break down their code into smaller, reusable components, making it easier to manage and maintain.
3. **Libraries and Packages:**
   - **Django REST Framework:** A toolkit we used for building web APIs with Django. It simplifies tasks like handling data, authentication, and permissions. Further DRF's browsable API makes it easy to explore and interact with API during development.
   - **Django Admin LTE:** Django Admin LTE improves the appearance and functionality of Django admin pages. It also offers customization options for tweaking the design and including appealing charts for better data presentation.
   - **PySerial:** PySerial is a Python library that helps you communicate with serial devices, like Arduino boards , by providing an easy-to-use interface for communication.

2.2 Frontend

1. **Programming Language:** PedalPal's frontend is built using **Dart**. Dart's C-style syntax with modern object oriented approach makes it a popular choice for development.
2. **Framework:** We have used **Flutter Framework**. We chose Flutter because we believe it simplifies Android app development with its fast performance, beautiful UI, and amazing reload features. Some other advantages of Flutter which makes it a popular choice are:

- **Expressive and Flexible UI:** Flutter provides a rich set of pre-designed widgets that can be easily customised, resulting in a beautiful and expressive UI.
- **Rapid Development:** Flutter provides a rich set of pre-designed widgets that can be easily customised, resulting in a beautiful and expressive UI.
- **Consistent UI Across Platforms** Flutter ensures consistent UI across different platforms, eliminating the need for platform-specific UI development.

3. **Libraries and Packages:**
   - **flutter-secure-storage :** We used Flutter Secure Storage to securely store sensitive data such as user authentication tokens. It ensures that data is encrypted and protected from unauthorized access.
   - **material.dart:** We used material.dart to use the Material Design components provided by Google. It ensures that the application follows the standard design guidelines.
   - **razorpay_flutter.dart:** We used for integrating the Razorpay payment gateway into our Flutter application to include payment functionality in our Flutter app.
   - **http.dart:** We used this to provide utilities for making HTTPS requests and handling HTTPS responses in our Flutter application.
   - **flutter_dotenv.dart:** We used this package to load environment variables from a .env file in our Flutter projects.
   - **fluttertoast.dart:** We used this to provide a simple and customizable way to display toast notifications in a Flutter application.

2.3 Database

For the Database, we have used **PostgreSQL** as our data base program. We chose PostgreSQL for its robustness, reliability, and extensive feature set. Some other advantages which makes it a popular choice are:

- **Open-source and community-driven**
- **Extensible:** PostgreSQL's extensive range of extensions allows users to tailor the database to their specific needs, enhancing its adaptability.
- **Scalability and High Performance:** Utilizing advanced optimization and storage mechanisms, PostgreSQL delivers exceptional performance under heavy workloads.
- **Cross-Platform Compatibility:** PostgreSQL runs on various operating systems and integrates easily with multiple programming languages and application stacks.

2.4 Building Systems:

As for our Building Systems, we used **Gradle**, An advanced build toolkit used by **Android Studio**, to automate and provide flexible ways to compile, build, and package Android applications.

3. Codebase

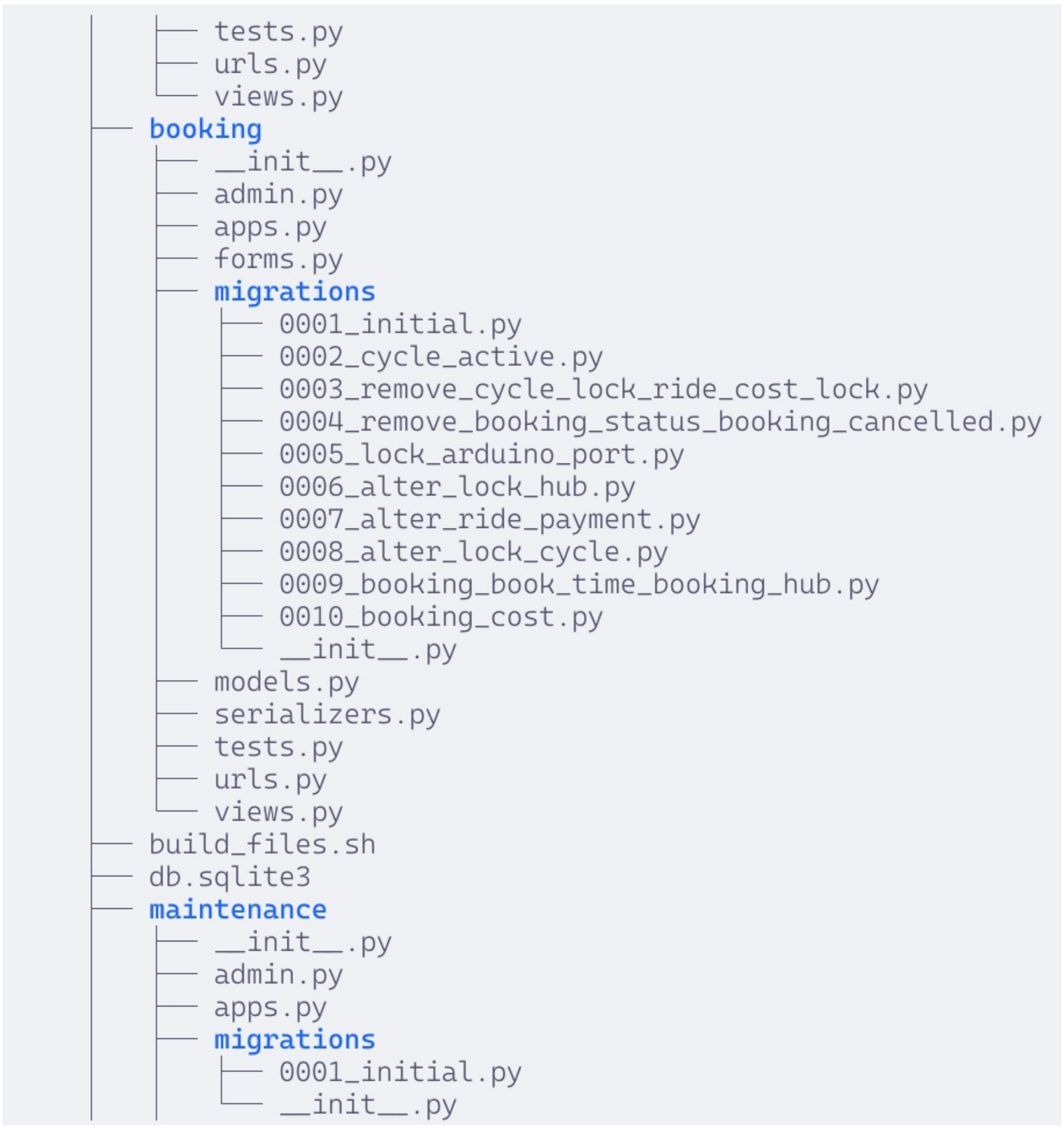
3.1 GitHub Repository

Link: https://github.com/Pedal-Pal-CS253/pedal-pal

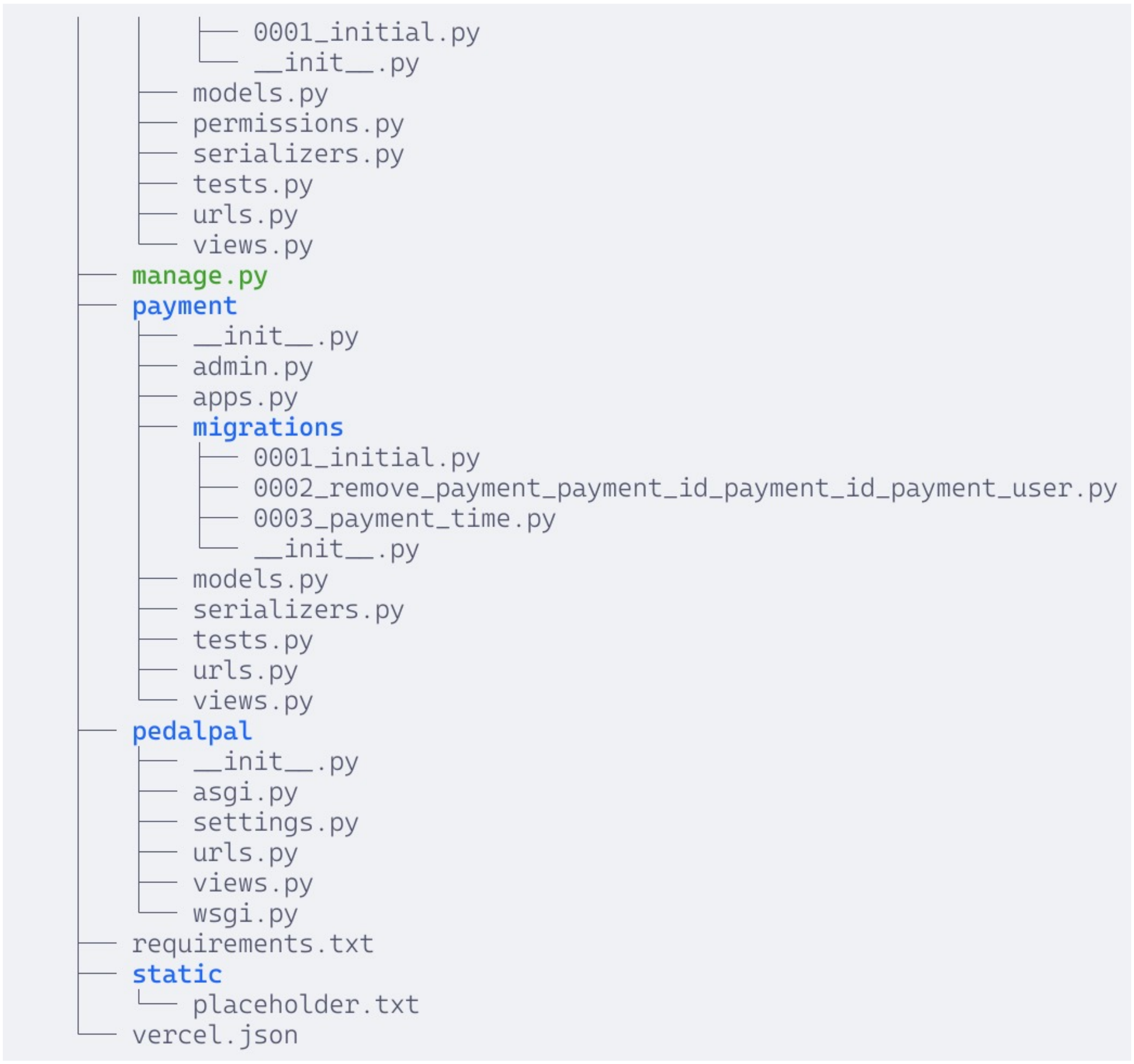
3.2 Codebase Navigation

3.2.1 Backend Directory: pedal-pal-backend

```
├── README.md
└── pedalpal
    ├── analytics
    │   ├── __init__.py
    │   ├── admin.py
    │   ├── apps.py
    │   ├── migrations
    │   │   ├── 0001_initial.py
    │   │   ├── 0002_initial.py
    │   │   ├── 0003_delete_wallet.py
    │   │   └── __init__.py
    │   ├── models.py
    │   ├── tests.py
    │   ├── urls.py
    │   └── views.py
    ├── authentication
    │   ├── __init__.py
    │   ├── admin.py
    │   ├── apps.py
    │   ├── email.py
    │   ├── migrations
    │   │   ├── 0001_initial.py
    │   │   ├── 0002_profile_balance.py
    │   │   ├── 0003_profile_is_verified_profile_otp.py
    │   │   ├── 0004_remove_profile_is_verified_alter_profile_is_active.py
    │   │   └── __init__.py
    │   ├── models.py
    │   ├── serializers.py
    │   ├── signals.py
    │   ├── templates
    │   │   └── email
    │   │       ├── otp_email.txt
    │   │       └── password_reset_email.txt
    │   ├── tests.py
```



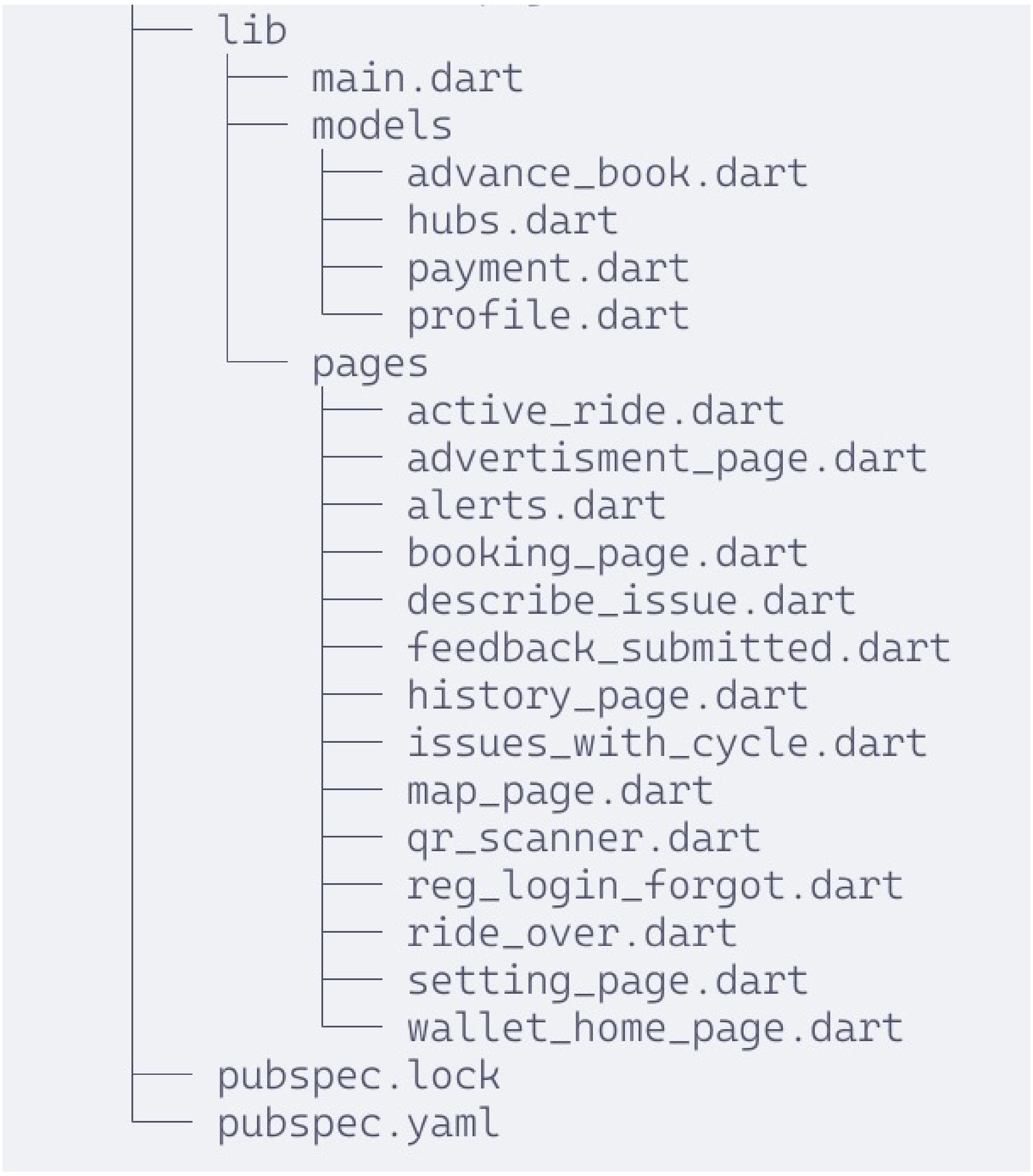
```
│   ├── tests.py
│   ├── urls.py
│   └── views.py
├── booking
│   ├── __init__.py
│   ├── admin.py
│   ├── apps.py
│   ├── forms.py
│   ├── migrations
│   │   ├── 0001_initial.py
│   │   ├── 0002_cycle_active.py
│   │   ├── 0003_remove_cycle_lock_ride_cost_lock.py
│   │   ├── 0004_remove_booking_status_booking_cancelled.py
│   │   ├── 0005_lock_arduino_port.py
│   │   ├── 0006_alter_lock_hub.py
│   │   ├── 0007_alter_ride_payment.py
│   │   ├── 0008_alter_lock_cycle.py
│   │   ├── 0009_booking_book_time_booking_hub.py
│   │   ├── 0010_booking_cost.py
│   │   └── __init__.py
│   ├── models.py
│   ├── serializers.py
│   ├── tests.py
│   ├── urls.py
│   └── views.py
├── build_files.sh
├── db.sqlite3
├── maintenance
│   ├── __init__.py
│   ├── admin.py
│   ├── apps.py
│   ├── migrations
│   │   ├── 0001_initial.py
│   │   └── __init__.py
│   │
```



```
│   │   ├── 0001_initial.py
│   │   └── __init__.py
│   ├── models.py
│   ├── permissions.py
│   ├── serializers.py
│   ├── tests.py
│   ├── urls.py
│   └── views.py
├── manage.py
├── payment
│   ├── __init__.py
│   ├── admin.py
│   ├── apps.py
│   ├── migrations
│   │   ├── 0001_initial.py
│   │   ├── 0002_remove_payment_payment_id_payment_id_payment_user.py
│   │   ├── 0003_payment_time.py
│   │   └── __init__.py
│   ├── models.py
│   ├── serializers.py
│   ├── tests.py
│   ├── urls.py
│   └── views.py
├── pedalpal
│   ├── __init__.py
│   ├── asgi.py
│   ├── settings.py
│   ├── urls.py
│   ├── views.py
│   └── wsgi.py
├── requirements.txt
├── static
│   └── placeholder.txt
└── vercel.json
```

3.2.2 Frontend Directory: pedal-pal-frontend

```
├── README.md
└── frontend
    ├── README.md
    ├── analysis_options.yaml
    ├── android
    │   ├── app
    │   │   └── src
    │   │       ├── debug
    │   │       │   └── AndroidManifest.xml
    │   │       ├── main
    │   │       │   ├── AndroidManifest.xml
    │   │       │   ├── java
    │   │       │   │   ├── com
    │   │       │   │   │   └── example
    │   │       │   │   │       └── frontend
    │   │       │   │   │           └── MainActivity.java
    │   │       │   │   └── io
    │   │       │   │       └── flutter
    │   │       │   │           └── plugins
    │   │       │   │               └── GeneratedPluginRegistrant.java
    │   │       │   └── res
    │   │       │       ├── drawable
    │   │       │       │   └── launch_background.xml
    │   │       │       ├── drawable-v21
    │   │       │       │   └── launch_background.xml
    │   │       │       ├── mipmap-hdpi
    │   │       │       │   └── ic_launcher.png
    │   │       │       ├── mipmap-mdpi
    │   │       │       │   └── ic_launcher.png
    │   │       │       ├── mipmap-xhdpi
    │   │       │       │   └── ic_launcher.png
    │   │       │       ├── mipmap-xxhdpi
    │   │       │       │   └── ic_launcher.png
    │   │       │       ├── mipmap-xxxhdpi
│   │       │       ├── mipmap-xxxhdpi
│   │       │       │   └── ic_launcher.png
│   │       │       ├── values
│   │       │       │   └── styles.xml
│   │       │       └── values-night
│   │       │           └── styles.xml
│   │       └── profile
│   │           └── AndroidManifest.xml
│   ├── gradle
│   │   └── wrapper
│   │       ├── gradle-wrapper.jar
│   │       └── gradle-wrapper.properties
│   ├── gradle.properties
│   ├── gradlew
│   ├── gradlew.bat
│   ├── local.defaults.properties
│   ├── local.properties
│   ├── secrets.properties
│   └── settings.gradle
├── assets
│   ├── confirmation.png
│   ├── google_fonts
│   │   ├── Lato-Bold.ttf
│   │   ├── Lato-Regular.ttf
│   │   └── Lato-Thin.ttf
│   ├── pedal_pal_logo.png
│   ├── pedal_pal_logo_small.png
│   ├── pedal_pal_logo_text.png
│   └── profile_photo.png
├── flutter_01.png
├── lib
│   ├── main.dart
│   ├── models
│   │   ├── advance_book.dart
```



```
├── lib
│   ├── main.dart
│   ├── models
│   │   ├── advance_book.dart
│   │   ├── hubs.dart
│   │   ├── payment.dart
│   │   └── profile.dart
│   └── pages
│       ├── active_ride.dart
│       ├── advertisment_page.dart
│       ├── alerts.dart
│       ├── booking_page.dart
│       ├── describe_issue.dart
│       ├── feedback_submitted.dart
│       ├── history_page.dart
│       ├── issues_with_cycle.dart
│       ├── map_page.dart
│       ├── qr_scanner.dart
│       ├── reg_login_forgot.dart
│       ├── ride_over.dart
│       ├── setting_page.dart
│       └── wallet_home_page.dart
├── pubspec.lock
└── pubspec.yaml
```

3.2.3 Instructions to Run the App:

- To download and use our application, please install the pedal-pal-v1.0.apk file given here.
- The administrator view can be accessed here.
- **NOTE:** Currently the application is available on the Android platform only. The backend of our application is hosted on the Vercel platform.

3.2.4 Instructions to Setup Development Environment:

- **Frontend**
  1. Create a fork to the repo in your account. Any changes should be done here and send a pull request to merge your changes.
  2. Open the project (cloned repo) in Android Studio. Run flutter pub get to get the updated dependencies.
  3. Create a .env file in the same directory as pubspec.yaml. The contents of this file are in our #frontend channel of discord. (This file contains sensitive information, hence it is not on GitHub)
  4. Create a secrets.properties file in the same directory as local.properties. Put the Google Maps API key in this file.
- **Backend**
  1. Create a fork to the repo in your account. Any changes should be done here and send a pull request to merge your changes.
  2. Create a new virtual environment using python3.9 -m venv <name>, and activate it using source <name>/bin/activate (Note that Python 3.9 is required for this to work)
  3. Install requirements.txt using pip install -r requirements.txt
  4. Create a .env file in the same directory as requirements.txt. The contents of this file are in our #backend channel of discord. (This file contains sensitive information, hence it is not on GitHub)
  5. Run command pre-commit install in the directory where .git directory is stored. This configures the git hook. (This is for ensuring that code is well formatted before performing git commit).
  6. After commiting changes, if the code is not well formatted, it gets automatically formatted and you need to commit it once again.

4. Completeness

4.1 Implemented Features

4.1.1 Login + Registration

- **Registration Page**
  - Allows users to register with Pedal Pal by providing their full name, phone number, email address, and password.
  - After entering details, the user receives a link on their email to activate their account. On clicking the link, the email is verified and user can login.
- **Login Page**
  - Allows registered users to log in using their email and password.
- **Forgot Password Page**
  - Enables users to initiate the password reset process by providing their registered email address.
- **Password Reset Page**
  - Users can reset their password by providing a new password and confirming it.
  - After successfully resetting the password, users are shown a confirmation page.

4.1.2 Ride Now or Book in Advance

- **Google Map Integrated**
  - Displays a Google Map with markers representing bike hubs.
  - Users can interact with these markers to view information about each hub, such as its name and the number of available cycles.
- **Hub Information**
  - Upon tapping a hub marker on the map, a popup container appears displaying detailed information about the selected hub, including its name, the number of available cycles, and options for advanced booking.
- **Advanced Booking**
  - Users can select a date and time for advanced bike booking directly from the popup container.
  - Enhances user convenience by allowing them to plan their rides in advance.

4.1.3 Active Ride

- **User Profile Display**
  - Shows the user's profile picture and full name at the top of the screen, providing personalization and user recognition.
- **Ride Status**

  - Allows users to check their ride status.
  - Upon tapping this button, users can view information such as ongoing ride details, duration, and distance traveled.

- **Payment Mode**
  - Indicates the current payment mode used for the ride.
  - If the user is subscribed, the payment mode is the built-in wallet, otherwise the user has to make payment after his ride ends.
- **End Ride Button**
  - Initiates the process of ending the ride.
  - Upon tapping this button, users are directed to the QR code scanner screen, which allows the user to scan the lock which they are storing the cycle in. After the ride ends, payment is collected from either the wallet or via the RazorPay interface.

4.1.4 QR Scanner

- **Overall**
  - The QRScan page provides an intuitive user experience for scanning QR codes, interacting with ride-related functionalities, and handling backend communication for booking and ending rides.
- **Mode-based Functionality**
  - Executes different actions based on the mode specified when navigating to the QRScan page.
  - The mode parameter determines whether the page is used for booking a ride (book mode) or ending a ride (end mode).

4.1.5 Booking Statistics Page

- **Current Bookings**
  - Shows a list of current bookings with information such as start location, start date, and start time.
  - Each booking is displayed as a separate card-like container with a circular icon on the left representing the booking.
- **Previous Bookings**
  - Displays a list of previous bookings similar to current bookings.
  - It includes details like start location, start date, and start time for each past booking.

4.1.6 Ride History

- **Completed Rides**
  - Lists all past rides that have been successfully completed.
  - Includes information such as start location, end location, and duration for each completed ride.

4.1.7 Feedback

- **Issue Selection**
  - Users can select various issues related to the cycle by tapping on the corresponding items displayed on the screen.
  - Each issue item has a title (e.g., ”Tires low on air,” ”Sounds while riding”) and a checkbox indicating whether the issue is selected or not.
  - Tapping on an issue item toggles its selection status.
- **Issue Description**
  - If the issues stated in the issue selection page are not present, then the user can navigate to the **Describe Issue** page and give an explanation about the problem faced.
- **Confirmation**
  - When the user’s issue has been submitted to the admin, the confirmation page is displayed.

4.1.8 Subscription

- **Wallet for Subscribers**
  - All the subscribed users will have a wallet where they can add an amount for future rides and bookings.
  - When a ride has been completed, and the wallet doesn’t have enough balance, the amount will go to negative, but this privilege is present just for once.
- **Booking Option**
  - Only when a user is also a subscriber, he/she gets the benefit of booking a ride in the near future, so that when they reach the hub, they will always find a cycle to rent.

4.2 Future Development Plan:

4.2.1 7-Day Trial

- Offer a 7-day trial allowing guest users to experience subscriber benefits
- Users can decide whether to subscribe or remain as guest users after the trial period.

4.2.2 Booking Cancellation

- Implement the ability for subscribers to cancel their current bookings.
- Enhance user control and flexibility over ride reservations.

4.2.3 Profile Picture Update

- Enable users to update their profile pictures.
- Enhance user customization and database management.

A. Group Log

NOTE: Every team member worked regularly on their assigned tasks, collabrating whenever needed. This log only covers online/in-person meetings and significant discussions.

S.No	Date	Timings	Venue	Description
1	11/02/2024	21:30 to 23:00	Discord	Discussed about Implementation Work, decided the timeline and divided work among teammates
2	14/02/2024	11:00 to 14:00	RM Building	Gathered Resources for learning about Django and Flutter, and setup our machines for developement process
3	26/02/2024	21:30 to 22:00	Discord	Meet to compile our individual work and analyse our progress and plan future development process
4	02/02/2024	14:30 to 17:00	RM Building	Completed Backend and Frontend, Started working on integration process and Implementation Document
5	09/03/2024	17:30 to 18:00	Google Meet	Discussed Few Doubts Regarding Testing and Integration with TA
5	18/03/2024	21:00 to 23:30	RM Building	Completed the Implementation Work and Finalised Implementation Document