UNIVERSITY MALAYSIA TERENGGANU
FACULTY OF OCEAN ENGINEERING TECHNOLOGY & INFORMATICS

CSM3114
Framework Based Mobile Application Development

Project Submission
PROJECT 2
‘SWIFTSPOT - SMART PARKING APPLICATION’

Prepared by:
Muhamad Arif Ajmal bin Afsanisham
S61658

Prepared for:
Dr. Mohamad Nor bin Hassan

BACHELOR OF COMPUTER SCIENCE (MOBILE COMPUTING)

WITH HONOURS

SEMESTER I 2023/2024

Table of Contents

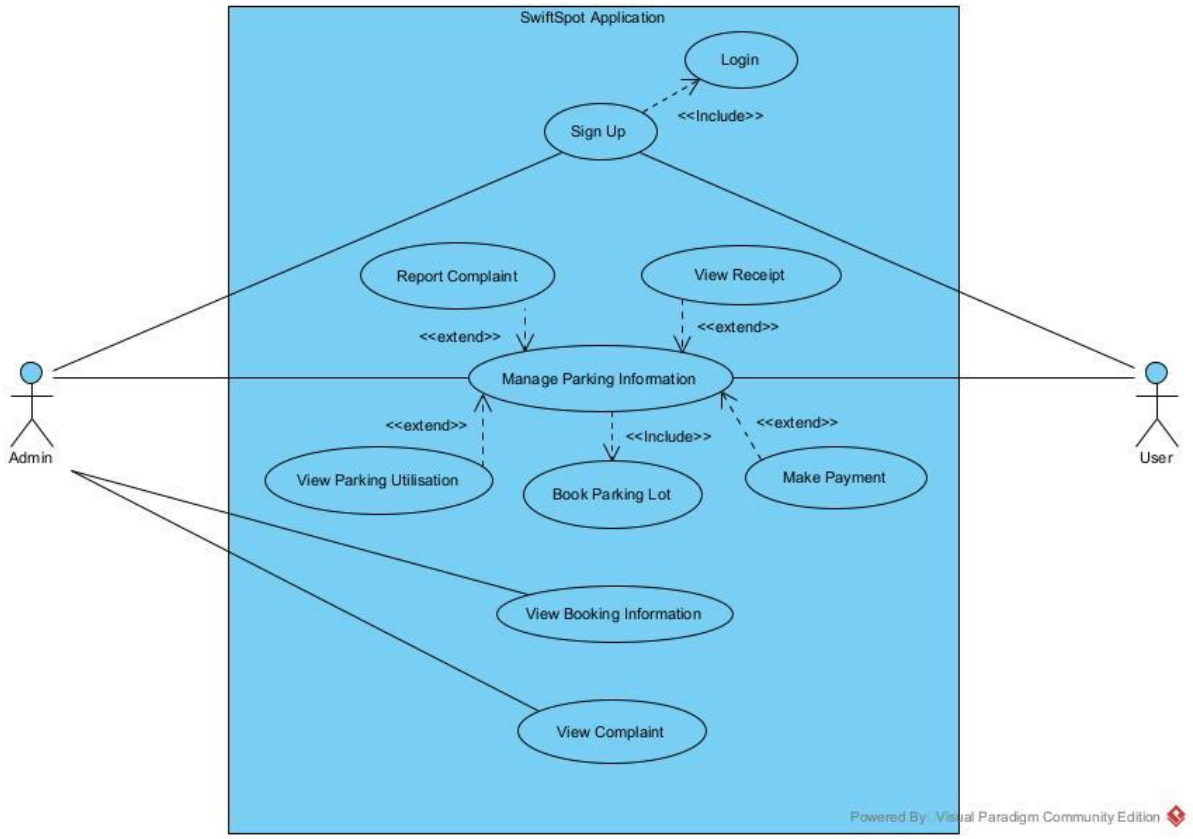
1 Summary of Prototype

SwiftSpot, a Flutter parking app, revolutionizes the parking experience for users by offering a comprehensive and user-friendly mobile application. This app serves as a one-stop solution for individuals seeking information about parking facilities, providing detailed insights into location, parking type, costs, hours of operation, and accessibility features. Users can effortlessly navigate through the intuitive interface to access essential details about various parking lots, enabling them to make well-informed decisions based on their preferences and requirements.

A standout feature of the app is its reservation functionality, allowing users to easily select a preferred parking lot, specify the desired duration, and seamlessly make reservations. The integration with Firebase Realtime Database ensures efficient and secure data storage, providing a robust foundation for the app's functionality. Users can also utilize the platform to file complaints or offer feedback, establishing a dynamic communication channel between parking facility users and management. This feature contributes to creating a responsive and user-centric parking ecosystem.

The user interface is designed with a focus on simplicity and visual appeal, ensuring an optimal user experience. With functionalities such as reservation management, streamlined payment processing, and an integrated complaints system, SwiftSpot aims to simplify and enhance the overall parking process. By providing users with a feature-rich and convenient tool, the app strives to contribute positively to the parking industry, making parking more accessible, efficient, and user-oriented.

2 Use Case Diagram



3 Tree Widgets Used for Designing and Developing Application

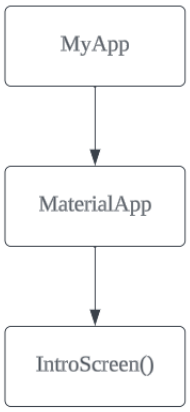


Figure 3.1 main.dart

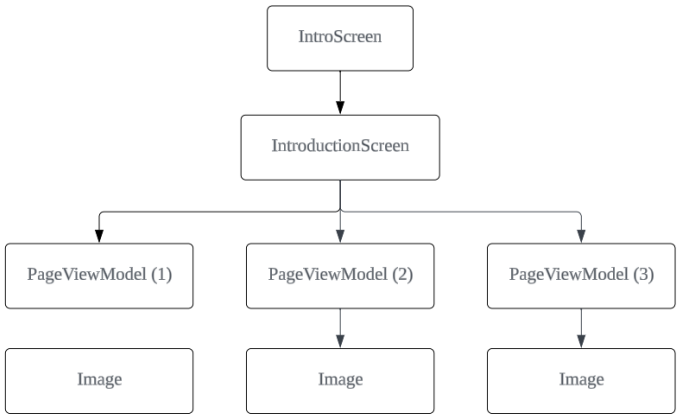


Figure 3.2 intro_screen.dart

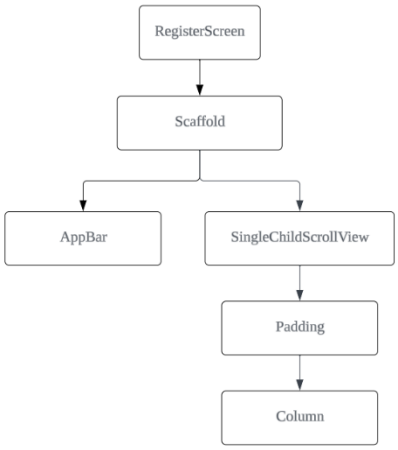


Figure 3.3 register_screen.dart

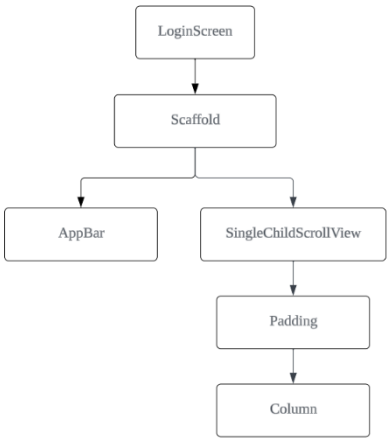


Figure 3.4 login_screen.dart

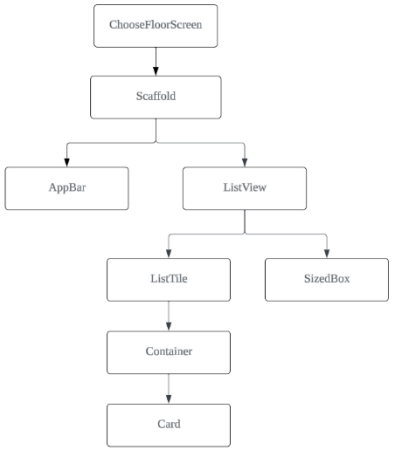


Figure 3.5 parking_floor.dart

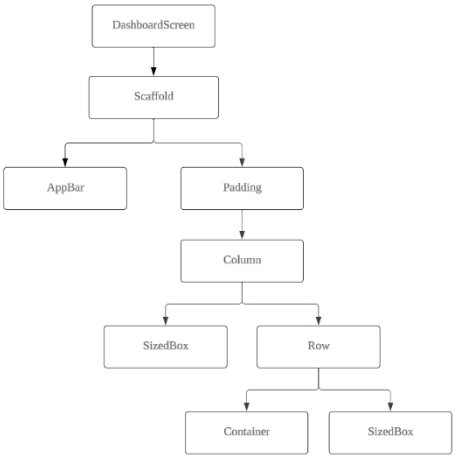


Figure 3.6 dashboard_screen.dart

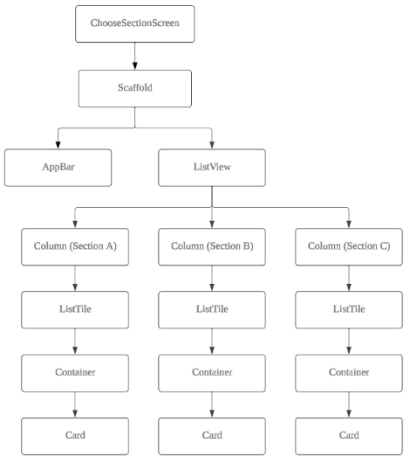


Figure 3.7 parking_section.dart

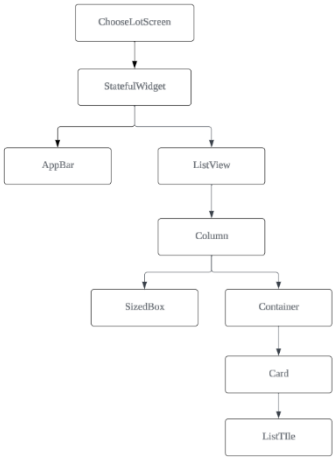


Figure 1.8 parking_lot.dart

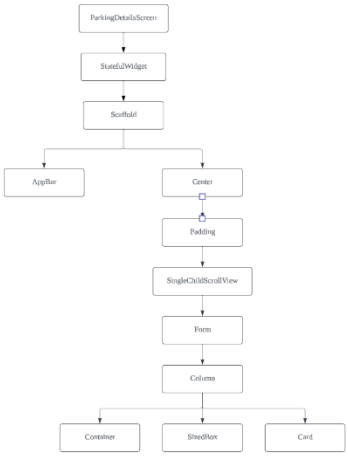


Figure 3.9 parking_details.dart

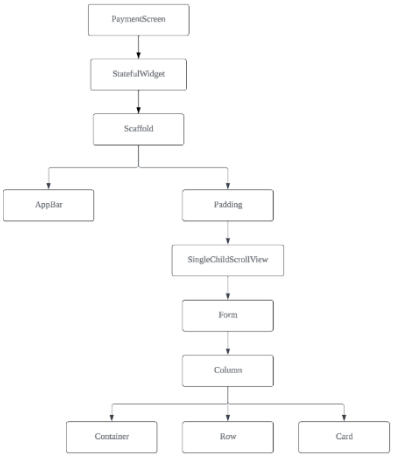


Figure 3.10 payment_screen.dart

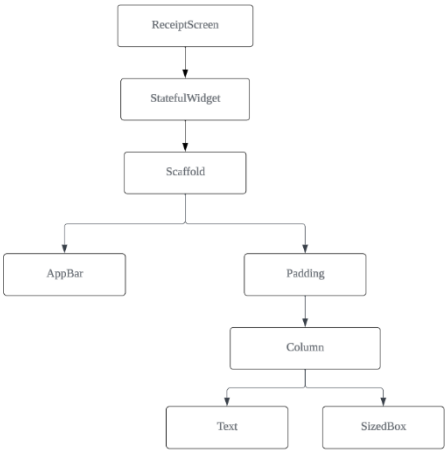


Figure 2.11 receipt_screen.dart

Figure 3.12 complaint_screen.dart

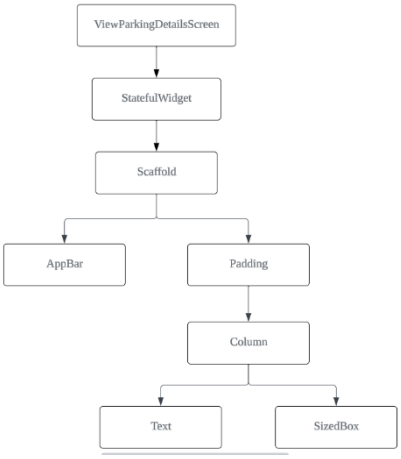


Figure 3.13 details_screen.dart

4 Flutter Widget and Features

SwiftSpot harnesses the versatility of Flutter's widget-based architecture, employing a variety of widgets to construct a dynamic and user-friendly interface. The use of `StatefulWidget` and `State` classes is pivotal for managing the application's mutable state, ensuring seamless interaction and responsiveness. Each screen, such as the `ParkingDetailsScreen`, leverages these Flutter classes to handle state changes dynamically, allowing users to interact with the app effortlessly.

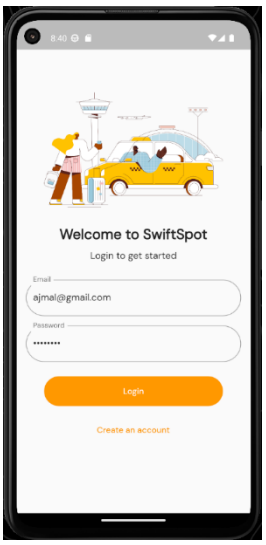
Navigation within the app is facilitated by the `AppBar` and `Scaffold` widgets, providing a consistent design language and framework across screens. These widgets ensure a cohesive visual structure, enhancing user navigation and maintaining a polished aesthetic. Additionally, the app utilizes the `Form` and `TextFormField` widgets to capture and validate user inputs effectively, enhancing the overall user experience.

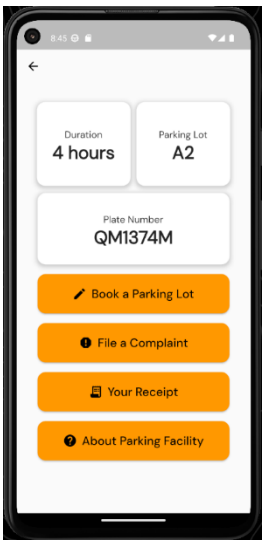
Interactive elements like the `Slider` widget enable users to intuitively select parking durations, offering a visually pleasing and user-friendly way to interact with the app. The use of `AlertDialog` further contributes to user feedback by displaying confirmation dialogs, providing clear responses to user actions and guiding them seamlessly through the app's flow.

Furthermore, the integration of the `http` package facilitates HTTP requests, enabling communication with the Firebase Realtime Database. This integration empowers the app to store and retrieve essential data such as parking details, payments, and complaints. The use of `DropdownButton` widgets streamlines the selection of predefined options, ensuring a compact and accessible means for users to choose complaint types, floors, and parking lots.

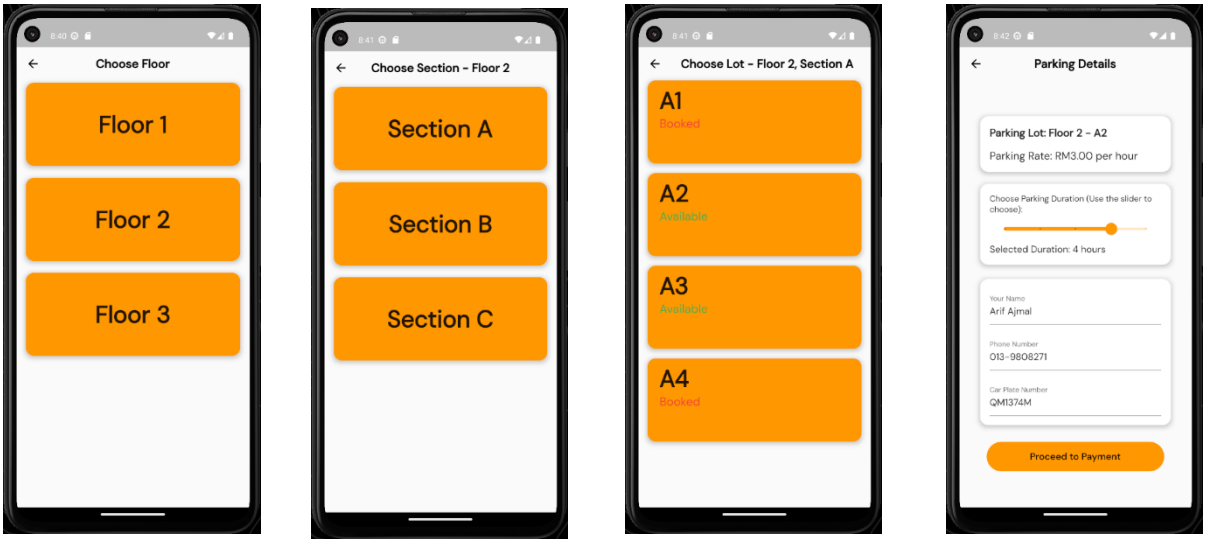
SwiftSpot's integration with Firebase Realtime Database and the incorporation of third-party packages underscore the extensibility and adaptability of Flutter. This project serves as a testament to Flutter's capability to deliver a robust and feature-rich application with a focus on user experience and seamless functionality.

5 Application User Interface

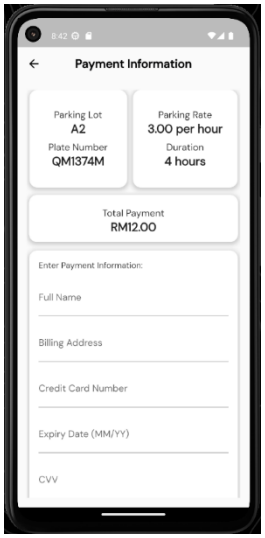


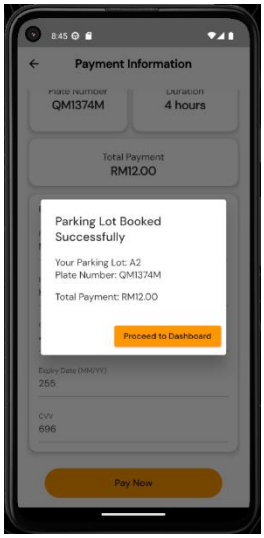


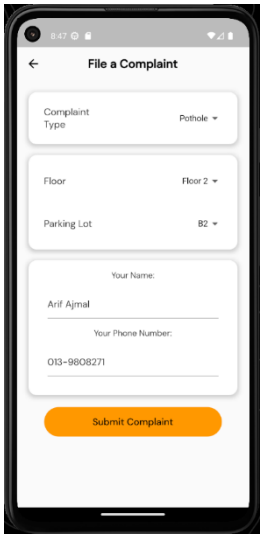
The first figure shows the introduction screen which navigate to the login screen. The second and third figure are the login screen and register screen respectively. The fourth figure is the main dashboard screen which shows the user selected parking lot, parking duration and their plate number.

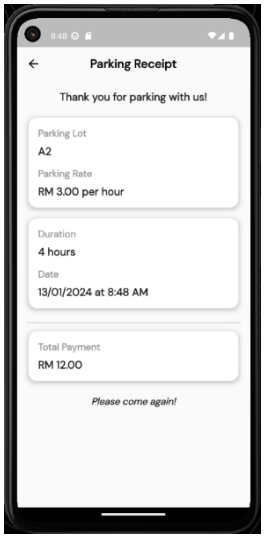


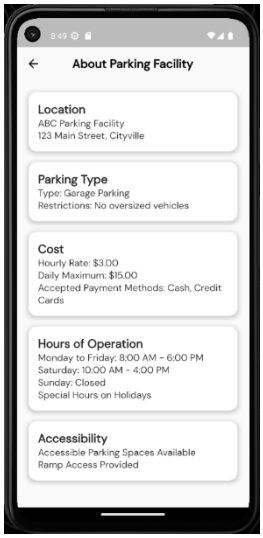
The first, second and third figure shows the navigation of user to book for their parking lot. User can choose which floor, section and parking lot that they desire before heading to parking details screen. The fourth figure is the parking details screen which will confirm what the user have chosen.











The first figure is the payment information which shows the selected parking lot and more information to the user. The second figure shows the parking confirmation before they will navigate to the main page. The third screen is the report complaint screen where user can report what to complain. The fourth screen is the parking receipt and the fifth figure is the parking facility screen.

6 Conclusion

In conclusion, SwiftSpot demonstrates the powerful capabilities of Flutter in developing a comprehensive and user-centric parking management application. The utilization of Flutter's widget-based architecture allowed for the creation of a visually appealing and responsive user interface across multiple screens. The seamless integration of state management through `StatefulWidget` and `State` classes ensures a smooth and dynamic user experience, particularly evident in features like the interactive parking duration slider.

The app's use of Firebase Realtime Database showcases Flutter's versatility in connecting with external databases, enabling efficient data storage and retrieval for parking details, payments, and complaints. The incorporation of HTTP requests via the `http` package further extends the app's functionality, establishing communication with the backend and facilitating critical operations.

User interactions are enhanced by thoughtful implementations of widgets such as `Slider`, `AlertDialog`, and `DropdownButton`, contributing to a user-friendly and intuitive design. The project not only highlights Flutter's capabilities in crafting a sophisticated application but also emphasizes its adaptability in handling diverse functionalities, including payment processing, complaint filing, and receipt generation.

Overall, SwiftSpot serves as a robust example of Flutter's efficiency, versatility, and developer-friendly features. As a powerful cross-platform framework, Flutter proves its prowess in delivering a polished and functional application while maintaining a streamlined development process. The successful execution of this project reinforces Flutter's standing as an excellent choice for building engaging and feature-rich mobile applications.

7 Reference

1. *Dropdown button not showing selected value in Flutter*. (n.d.). Stack Overflow. https://stackoverflow.com/questions/57721827/dropdown-button-not-showing-selected-value-in-flutter
2. *firebase_core | Flutter Package*. (n.d.). Dart Packages. https://pub.dev/packages/firebase_core/install
3. *firebase_database | Flutter Package*. (n.d.). Dart Packages. https://pub.dev/packages/firebase_database
4. Flutter Agency. (2020, December 9). Steps to Solve DropDown Selection is Not Displaying In Flutter? *Flutter Agency* -. https://flutteragency.com/how-to-solve-dropdown-selection-is-not-displaying-in-flutter/
5. *google_maps_flutter | Flutter Package*. (n.d.). Dart Packages. https://pub.dev/packages/google_maps_flutter
6. *How to add Icon on Elevated Button in Flutter App*. (n.d.). Flutter Campus. https://www.fluttercampus.com/guide/116/how-to-add-icon-on-elevated-button-in-flutter-app/
7. *introduction_screen | Flutter Package*. (n.d.). Dart Packages. https://pub.dev/packages/introduction_screen/install
8. Kazlauskas, M. (2023, February 6). Flutter Design patterns: 0 — Introduction - Flutter Community - medium. *Medium*. https://medium.com/flutter-community/flutter-design-patterns-0-introduction-5e88cfff6792

9. Song, N. (2022, July 22). Flutter: Creating custom color swatch for MaterialColor | by Nicholas Song | Medium. *Medium*. https://medium.com/@nickysong/creating-a-custom-color-swatch-in-flutter-554bcdcb27f3

10. *Widget catalog*. (n.d.). Flutter. https://docs.flutter.dev/ui/widgets

8 Code Submission

https://github.com/arifajmal/FrameworkMobileDevelopment.git