Technical Handover Document (Front-end)

Overview

The primary goal of our project is to facilitate the management of NUS car parks by providing a simulation model with a user-friendly interface that predicts occupancy rates in neighbouring car parks in the case of car park closure. Closures may be due to renovations or events, and University Campus Infrastructure (UCI) staff would need to plan for the rerouting of vehicles and reassignment of red (season) and white (visitor) lots. The model would assist in more efficient planning for the user and allow optimal usage of car park space.

Design

The front-end team aimed to build a user-friendly and interactive interface which allows the user to easily customise the simulation scenario according to requirements. It would also present the simulation results to the user in a concise but comprehensive manner. To do this, we started from a bare bones wireframe and slowly built upon and refined it over multiple rounds of interviews with the user.

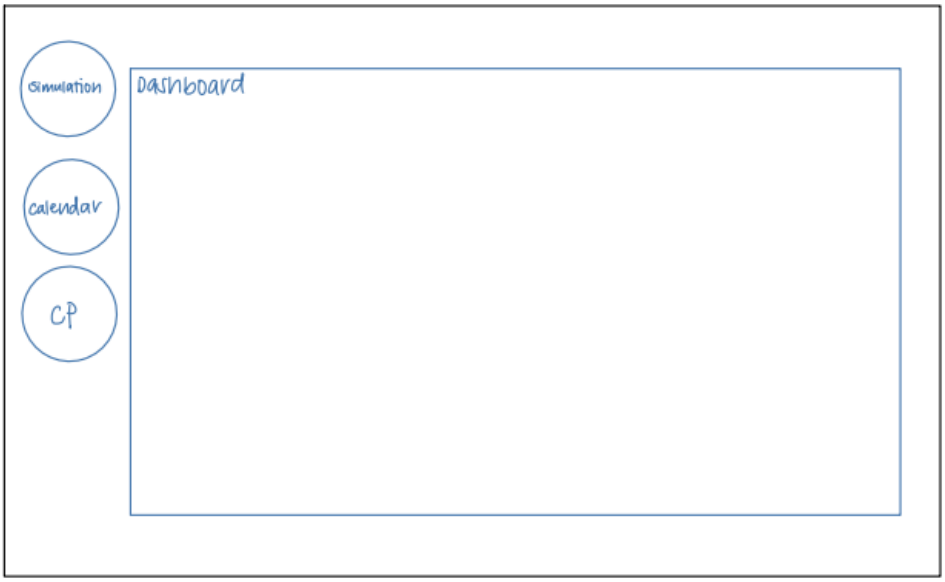

Fig 1: Wireframe 1

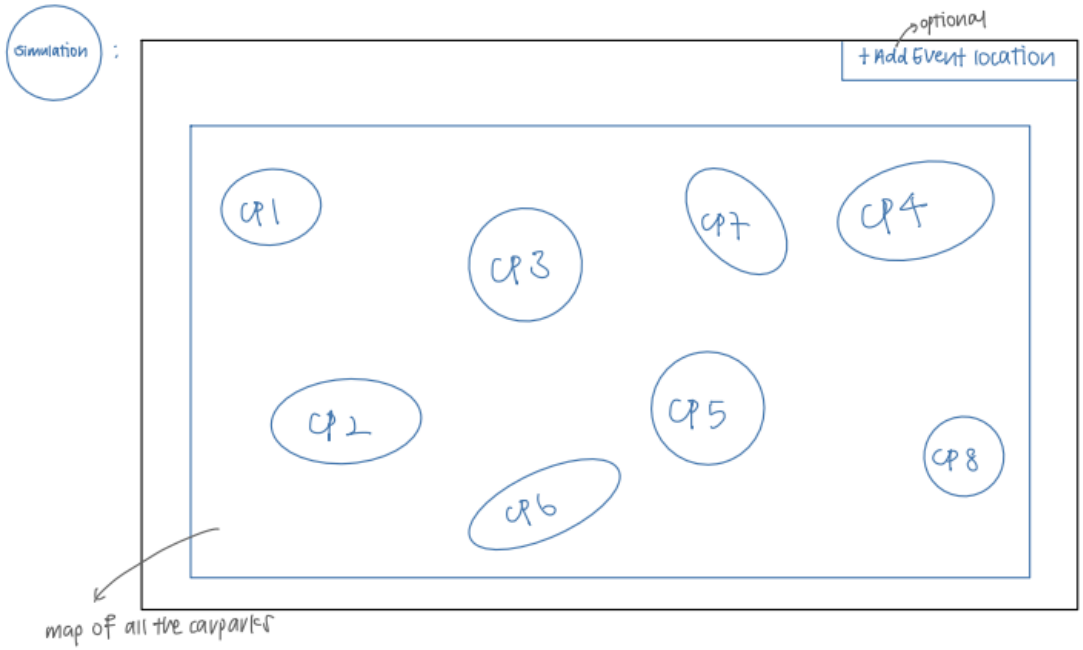

Fig 1.1: Simulation

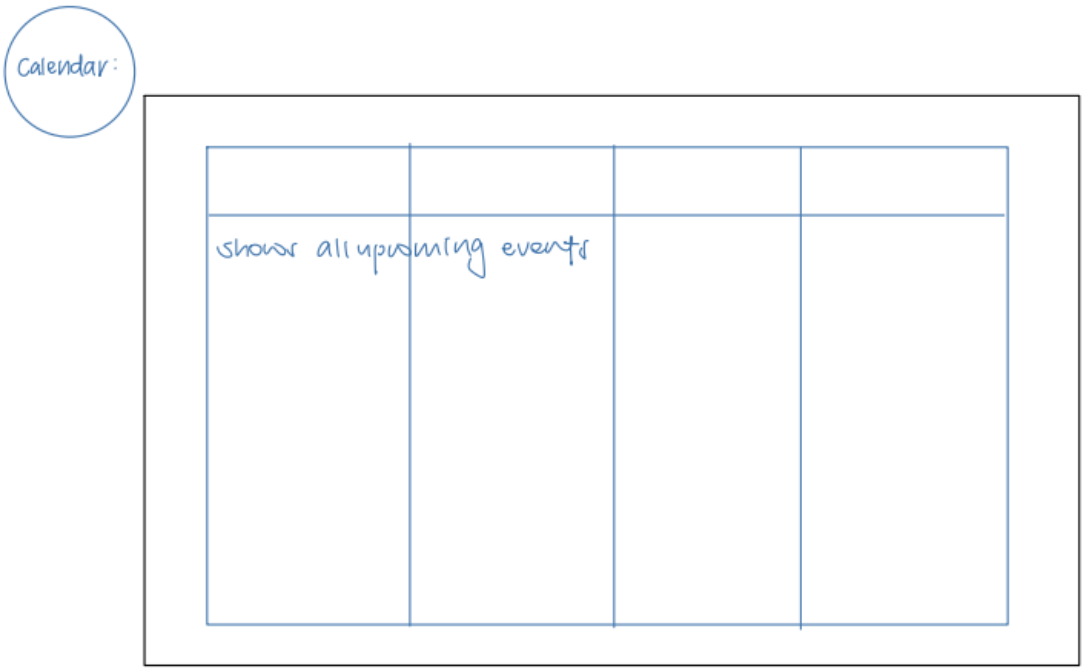

Fig 1.2: Calendar

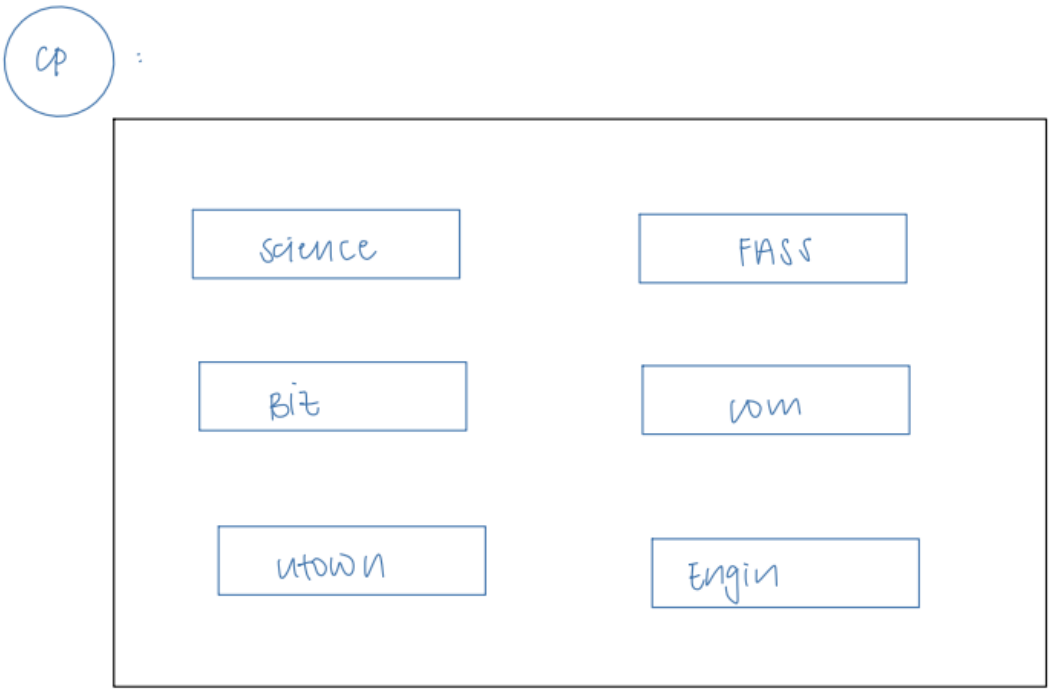

Fig 1.3: CP

As shown in Fig 1, our initial wireframe consisted of a dashboard with links to three pages; Simulation, Calendar and Car Park (CP). As we decided to proceed using this as a foundational structure, we began to design the interface more extensively in our next iteration.

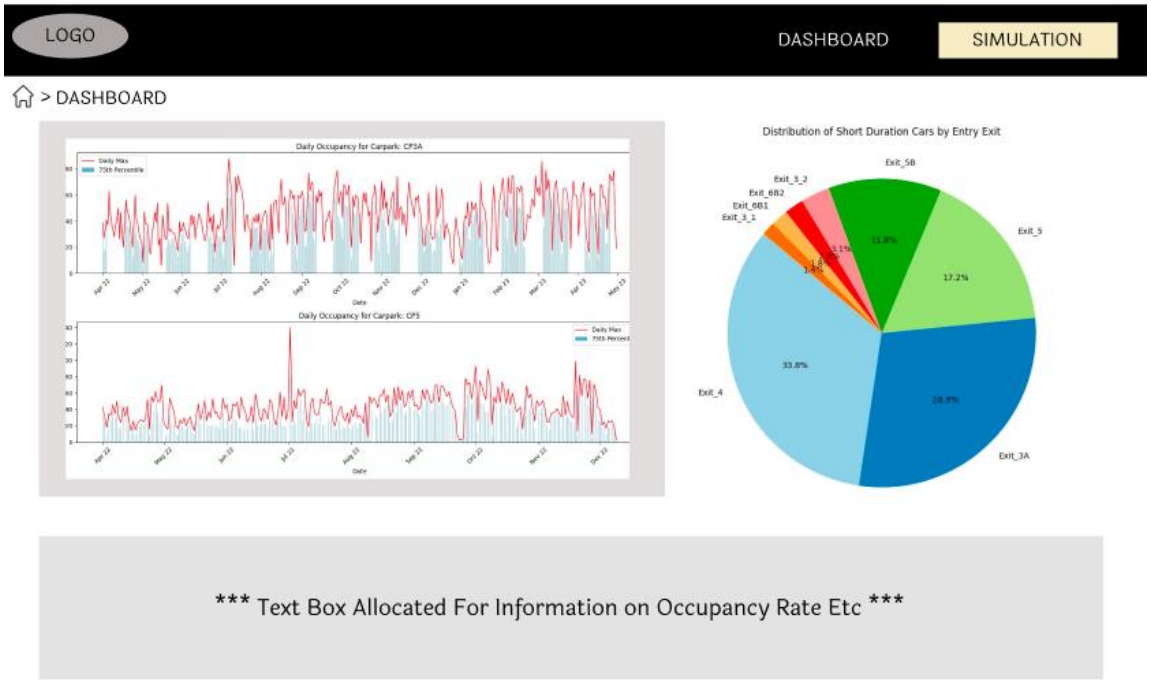

Fig 2: Wireframe 2

RETURN TO DASHBOARD

Fig 2.1: Simulation Page

RETURN TO DASHBOARD

← OCTOBER 2023 →

SUN	MON	TUE	WED	THU	FRI	SAT
1	2	3	4	5	6	7
8	9	10 CP15 Maintenance	11	12	13	14
15	16	17	18	19 NUS EVENT	20	21
22	23	24	25	26	27	28
29	30	31				

Fig 2.2: Calendar

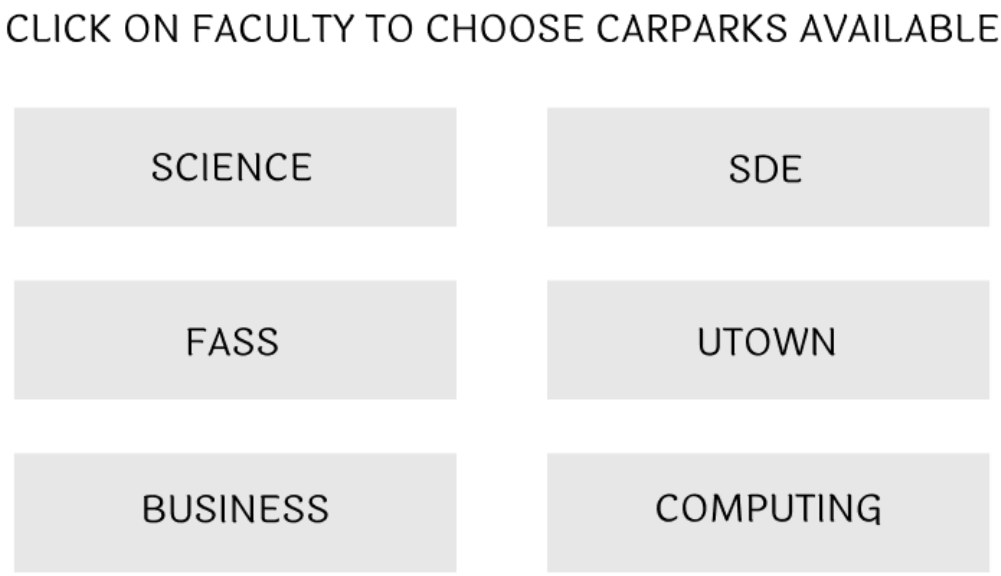

Fig 2.3: CP

The result is as shown in Fig 2. After the first interview with the user, we decided to remove the CP page entirely. We also noted that the Calendar page was not a high priority and should be integrated with the simulation if included. Additionally, the user wanted to see the output when we ran the simulation and, in particular, a summary of the results.

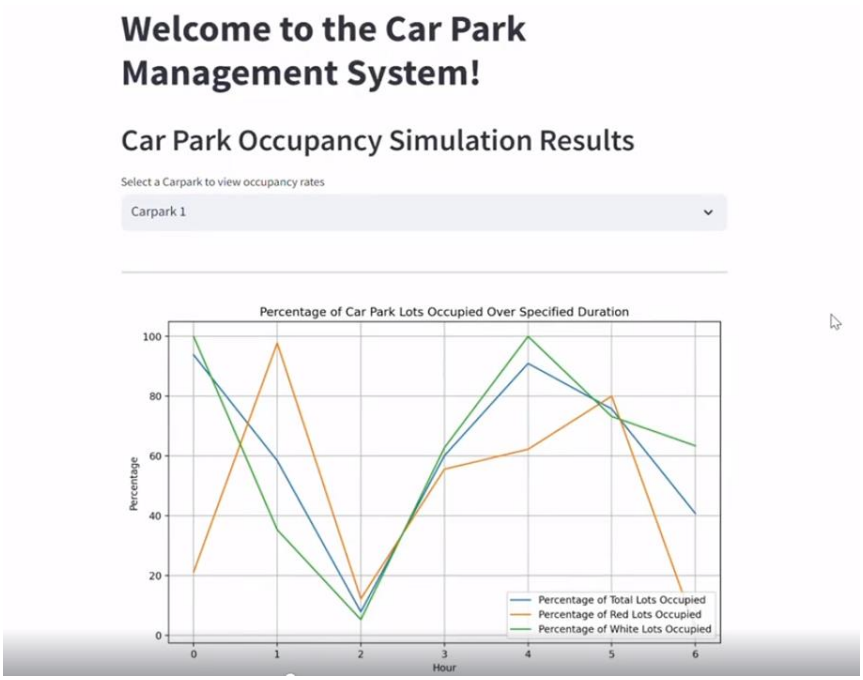

Fig 3: Simulation Summary

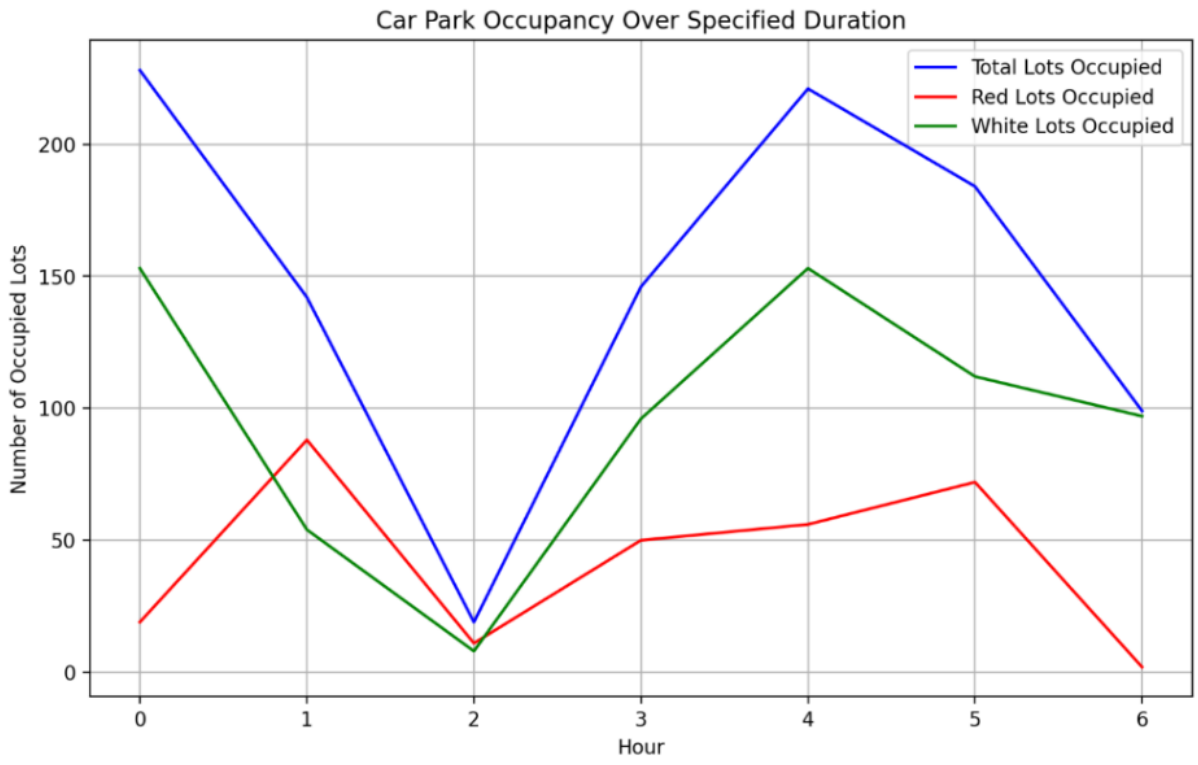

Fig 3.1: Actual Figure of Vehicles

With further rounds of interviews with the user, we smoothed out some additional details such as representing the actual figure of vehicles parked in a selected car park in the simulation summary (Fig 2.1). We initially only displayed the figure in terms of percentages of red, white and total lots (as shown in Fig 2) to assess whether the utilisation of parking spaces was balanced, but feedback from the user told us that an actual figure was helpful for a clearer understanding of the operational impact.

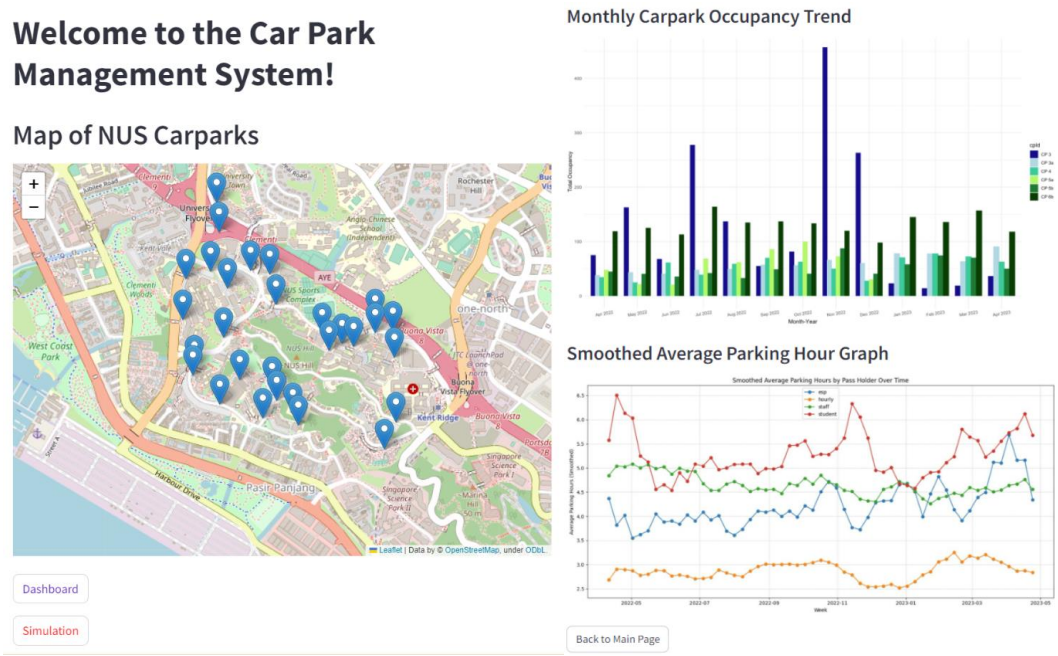

Fig 4: Final Design

User feedback ultimately led us to settle on the final design shown in Fig 4, with an interactive map on the homepage that allows users to click on individual car parks for detailed information. The two graphs shown on the dashboard, the monthly car park occupancy trend and smoothed average parking hour graph helps to identifies seasonal patterns and understand peak usage times respectively. This would best help our user in their tasks of resource allocation and maximising use of parking spaces.

In addition, to ensure an intuitive and efficient interface, we incorporated heuristic design principles such as Miller's law, a cognitive psychology concept that informs us that an average person's working memory capacity is about seven items. This principle guided our efforts to keep the interface clean and straightforward, ensuring users can easily navigate and access the information they need.

Code Base

The project is developed using Streamlit, a Python library for creating web applications with interactive user interfaces. The application provides functionality for viewing a map of NUS carparks, a dashboard displaying monthly car park occupancy trends, and a simulation feature for predicting car park occupancy.

Packages:

- Streamlit: Used for creating the web application and managing the user interface.
- Pandas: Utilized for data manipulation and handling car park data from a CSV file.
- Folium: Integrated for creating interactive and dynamic maps.
- Matplotlib and Plotly Express: Employed for generating visualizations and graphs.
- Datetime: Utilized for handling date and time information.

Pages and Functionality:

1. Main Page:
    - Displays a map of NUS carparks using Folium.
    - Allows navigation to the Dashboard and Simulation pages.

2. Dashboard Page:
    - Presents graphs showing monthly car park occupancy trends.
    - Provides a button to return to the Main Page.

3. Simulation Page:
    - Displays a map of NUS carparks.
    - Allows the user to select a car park and perform actions like closing the car park or reallocating lots.
    - Provides options for creating a simulation.
    - Offers a button to return to the Main Page.

4. Create Simulation Page:
    - Allows the user to set simulation parameters such as start and end times, and, optionally, plan for an event.
    - Validates input and calculates the simulation duration.
    - Provides a button to run the simulation.

5. Plot Simulation Page:
    - Simulates car park occupancy over the specified duration.
    - Displays graphs showing the percentage of total, red, and white lots occupied, as well as the total number of occupied lots.
    - Allows selection of a specific car park for detailed analysis.
    - Provides a button to return to the Main Page.

Session State Variables:

- selected_carpark: Stores the selected car park for simulation.
- total_lots, red_lots, white_lots: Track the total and color-specific lots for the selected car park during simulation.
- start_datetime, end_datetime: Store the start and end times for simulation.
- simulation_duration: Tracks the duration of the simulation.
- event_start_datetime, event_end_datetime: Store the start and end times for an optional event during simulation.
- expected_cars: Stores the expected number of cars for the event.
- event_carpark: Stores the car park selected for the optional event.
- carpark_to_view: Stores the car park selected for detailed analysis during simulation.

Data:

- Car park data is loaded from a CSV file named 'car_parks.csv'.
- Monthly occupancy and parking hour graphs are loaded as images for display in the dashboard.

How to Run:

- Ensure that all required Python libraries are installed (the require packages are listed in the file ‘requirements.txt’).
- Navigate to the folder containing the Python file in your terminal and run the command ‘streamlit run web.py’.

Recommendations:

- Ensure that all necessary data files (e.g., 'car_parks.csv', images) are available and accessible.
- Consider implementing additional error handling and input validation for enhanced robustness.
- Keep the environment updated with the latest versions of libraries to benefit from improvements and security patches.