Software Requirements Specification

for

Park n Go Mobile Application

Version 1.5 approved

Prepared by:

Pinnepu Jaswanth,

Kat Tsz Yin,

Tharakan Rohan Roy,

Thin Lat Han,

Zou Yulin,

Li Ziyang

Team A+PPLE

16/04/2020

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Thin Lat Han	10/04/20	First draft for SRS	1.0
Li Ziyang	17/04/20	Final draft for SRS	1.5

Name	Date	Reason For Changes	Version
Thin Lat Han	10/04/20	First draft for SRS	1.0
Pinnepu Jaswanth	12/04/20	Revision of first draft	1.1
Kat Tsz Yin	14/04/20	Second version of first draft for SRS - Remove non applicable segments from SRS	1.2
Zou Yulin	15/04/20	Revision of second draft	1.3
Tharakan Rohan Roy	16/04/20	Third version of second draft for SRS - Add in all components from SRS	1.4
Li Ziyang	17/04/20	Final Submission Version - Added in formating - Updating content page	1.5

1. Introduction

1.1 Purpose

This SRS document describes the functional and non-functional requirements for Park n Go. This document will explain the main features and design requirements of the application to relevant stakeholders and serve as an agreement before and after the development of the application.

1.2 Document Conventions

DC-1: This SRS is created with reference to the IEEE template for Software Requirement Specification Documents. Modification has been made to adapt it to the nature of a school project.

DC-2: The headings of segments in this SRS has the following specifications: Font: Arial, Size: 18, Style: Bold formatted for the main heading and decrease by 2 for every subheading level under the main heading.

DC-3: The content texts of this SRS has the following specifications: Font: Arial, Size: 12, Style: regular print

DC-4: The content with special attention in this document shall be bolded .

DC-5: References to other parts of the SRS or other documents shall be italicized .

DC-6: Spacing of 1.5 is added after paragraph for readability.

DC-7: As far as functional requirements concerns, nesting of number and indentation represents hierarchy of the requirements (overall - detailed - more detailed and so on).

DC-8: Bulletting or abbreviated heading of a segment to list common ideas together.

1.3 Intended Audience and Reading Suggestions

This document is intended for developers and testers of the application who will use it as a strict guideline in their implementation and maintenance process. It is also intended for the course supervisors of CE2006 to better understand our project.

Readers are suggested to follow the flow of the document to understand the scope of the project, functional and non-functional requirements, interfaces involved, system features and supporting diagrams.

A Data Dictionary in *Appendix A* is provided to ensure that all readers have a clear definition of the terms that are used in this document to avoid any misunderstanding. For developers and testers, details of the use cases, class diagrams, sequence diagrams and state machine flows are attached to *Appendix B* that covers analysis models.

1.4 Product Scope

Park n Go is a mobile application developed using Flutter software development kit (SDK) that can be used on both iOS and Android mobile platforms for car users in Singapore to ease their journey planning. Users can find car parks near current location by setting their current location with GPS in their phones. Alternatively, users can search for a desired location using Google Maps Application Program Interface (API) to provide suggestions for the search of location and find car parks near the desired location. Then, users can see the available vacancy of car parks which is provided by government API from data.gov.sg. Users can see through the colour of the pins that locate the locations of the car parks nearby. For example, if the pin is red, it means that there is no available vacancy for the car park; however, if it is green, it means there is sufficient available vacancy. Users can also click on each pin to know the exact number of available vacancies in each car park. Additionally, users can click the pin of a car park and then the pop-up to save the car park as a bookmark for easy access the next time the users want to find the location.

Park n Go aims to aid users in journey planning by providing them with real-time available vacancy of carparks and hence, remove the hassle and time involved in finding a place to park when driving towards their destination

1.5 References

The following documentations are used when developing the Park n Go mobile application.

1.5.1 Flutter SDK documentation

Documentation on Flutter SDK:
https://flutter.dev/

1.5.2 Singapore Government dataset and API

API for available vacancy of each carpark
https://data.gov.sg/dataset/carpark-availability

API for information of each carpark
https://data.gov.sg/dataset/hdb-carpark-information

1.5.3 Google Public API

API collections for Google Developers:
https://developers.google.com/products/develop

Google API for auto-completing of suggestions during destination input:
https://developers.google.com/places/web-service/autocomplete

1.5.4 IEEE SRS Template

Template used by this SRS:
https://ieeexplore.ieee.org/document/278253

1.5.5 GPS Coordinate Standards

WGS_84 (used by Google):
https://www.unoosa.org/pdf/icg/2012/template/WGS_84.pdf

SVY21 / Singapore ™:
https://epsg.io/3414

Link for converting SVY21 to WGS84:
https://epsg.io/transform#s_srs=3414&t_srs=3857

1.5.6 Use Case Models

The use case diagram and use case descriptions for requirement analysis of this project can be obtained from the following links:

Drive:
https://drive.google.com/file/d/1wSaHSSFqMpW2Lro_HsXq4CwlvC3EmhqO/view?usp=sharing

2. Overall Description

2.1 Product Perspective

This product is a new, self-contained mobile application. It can be deployed on both Android and iOS devices with an operating system of Android Jelly Bean, v16, 4.1.x or newer, or iOS 8 or newer. It utilizes google map API to provide map and location service. API and data from data.gov.sg to obtain information of the carparks, in particular real-time data of the availability of carparks. The app also contains a local SQLite database that stores user-specific information, namely the car park and locations the user bookmarks.

2.2 Product Functions

The following function is provided in the ParknGo application:

- Display Current location:
- Search Location by Name
- Display Car Park Information
- Add Bookmark
- Select Bookmark
- Manage Bookmark

2.3 User Classes and Characteristics

Users of this app would include drivers of small to medium cars or minivans. These may include daily car users such as those commuting work or going grocery shopping, who might sometimes need help finding parking spots. However, the app might be particularly useful for delivery drivers or private car-hiring drivers, who would need to find parking places in various locations across the island depending on the situation. The app would also be helpful for tourists who rented a car or families deciding where to hangout during the weekends, by checking for available parking spots before deciding where to go.

The search for destination function is essential for those planning their trip ahead, such as family and tourists. The current location feature is useful for those might running into parking problems anytime, such as freelance drivers, while the bookmarking function would be helpful for those going to repeated locations, such as those traveling for work or daily errands.

2.4 Operating Environment

The mobile application can work on any iOS devices (iPhone 4S or newer) and ARM Android devices. The device should have stable internet access to ensure that the application can access the google map API and data.gov.sg API to function properly. GPS-based or similar location service should be available on the device to provide current location information. The device should also have more than **500MB** of space to install the application.

2.5 Design and Implementation Constraints

The app is developed using the Flutter package to enable cross-platform deployment of the same application. It uses google map service to display maps and car park location. Json data from data.gov.sg is used to obtain car park information. SQLite database is used to store user specific info locally.

Good software design patterns such as Model-View-Controller should be incorporated in the design of the application. This would make potential upgrades and expansions in future possible. For example, the model and controller can be reused for a possible website that provides the same service to desktop users. The user interface can also be upgraded without changing the models and controllers.

2.6 User Documentation

UD-1: There are **no user manuals** as the app has a simple and intuitive design.

UD-2: There is a **demo video about the application usage is available** as part of this project's submission and users may refer to it.

2.7 Assumptions and Dependencies

A1: The application assumes government data APIs can provide reliable data regarding car parks at real time of its core functions. It also assumed that google map API can reliably produce location search results and display the map of Singapore. If the APIs malfunction, the application will not be able to function as normal.

A2: The application is developed using Flutter SDK and is only compatible for versions above Android 8.0 iOS 8.0 version.

A3: The application requires a WiFi or cellular network to access the APIs and database to access and retrieve information. It is also assumed that GPS feature is enabled.

A4: The application is only usable in Singapore and attempts to search for car parks outside Singapore will not have a result. Extension to other countries may be conducted in the future.

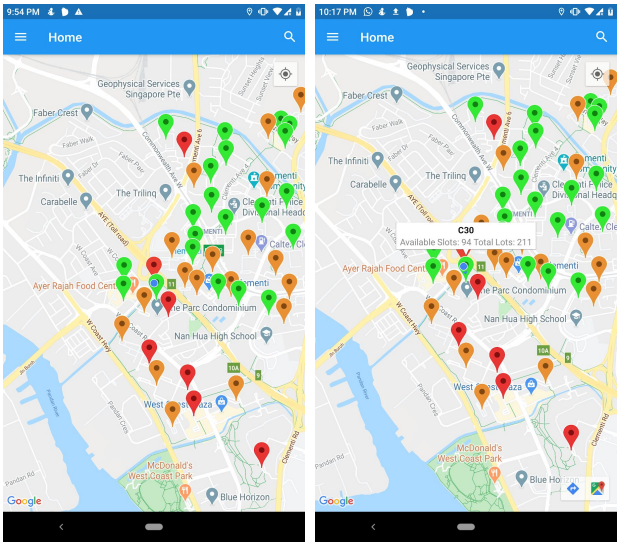
3. External Interface Requirements

3.1 User Interfaces

A key difference between other parking applications and Pack n Go is its simple to use interface and ability to bookmark frequently used car parks for convenience.

There are five main screen layouts designed for this application and they are detailed below:

Screen Layout 1: User see the available car parks near current location

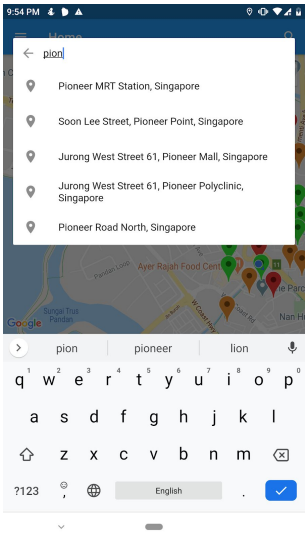


Upon entering the application, the user will be able to see their current position denoted by a blue dot and the car parks near their location.

Each car park is denoted by a coloured pin to indicate car park occupancy. Red means no available vacancy, orange means average number of available vacancies and green means many available vacancies.

Users then can press on a pin to see the exact number of available vacancies in that car park which is shown in the white dialog box.

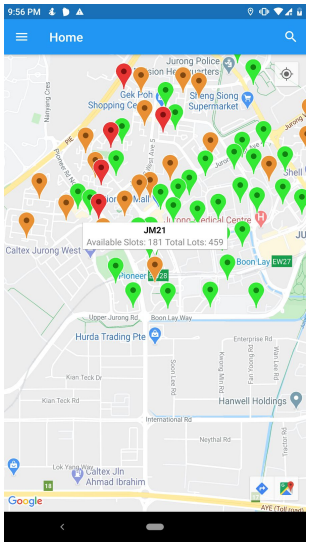
Screen Layout 2: User enters their desired location

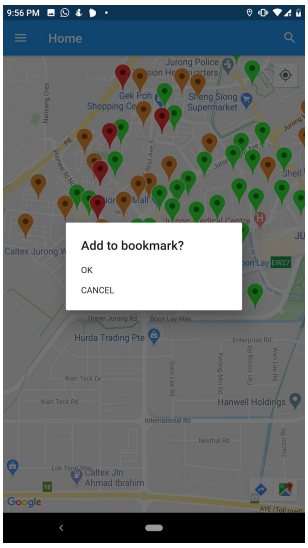


The application uses a simple and intuitive search icon on the top right corner which will open up a search bar where users can enter their desired location. Then, the user will be able to select an option from the dropdown list that offers suggestions related to the input.

After the user selects an option, *screen layout 1* with the location centered at the desired location of the user will be displayed.

Screen Layout 3: User add bookmark and see the bookmarks

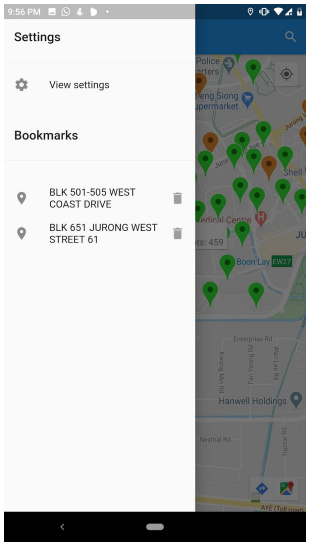




When the user presses the white dialog that shows the available vacancy in the car park, a white box pop-up. The pop-up will ask if the user wants to add the car park to the bookmark.

When the user presses “ok”, the car park will be added to the bookmark bar in *Screen Layout 4*.

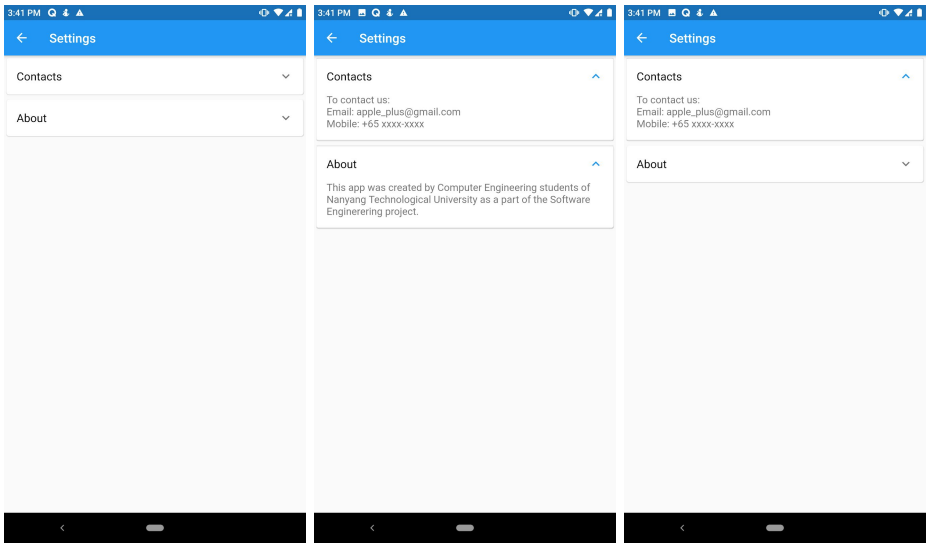
Screen Layout 4: User choose a bookmark from the bookmark bar



The application uses a simple and intuitive sidebar icon on top left where the user can press to see “Settings” and “Bookmarks”.

When the user presses one of the bookmarks, *screen layout 1* with the location centered at the bookmarked location that the user had chosen will be displayed.

Screen Layout 5: User choose View setting to view more information about the application



When the user presses the “View settings” in *Screen Layout 4,* setting page will be displayed.

Upon, pressing expand for “Contacts” or “About”, the tab will be expanded, displaying more informations

3.2 Hardware Interfaces

To run the application, the user must use a mobile phone that operates on a minimum Android 8.0 (Oreo) or iOS 8.0 version operating system.

The mobile phone must be connected to a WiFi network or cellular data. Additionally, the mobile phone must allow the application to access the phone’s GPS (Global Positioning System)

The mobile phone should have enough storage space (**at least 500MB**) to download and store data the application is processing.

3.3 Software Interfaces

For Development:

OS: Windows 10 and Mac OS X are the main computer platforms

Tools: Android Studio 3.5.2 , Flutter SDK 1.8.3 , Android Emulator or Android phone with Android 8.0++ with developer mode enabled.

Data format: CSV and JSON (from APIs)

Data sharing in between classes via public variables and public classes.

For Application Functionalities and Deployment:

Database:

- Information of car parks under Housing Development Board (HDB) in Singapore

  Request Data: Carparks' location, ID, Total lots and available slots.

  Return Data: CSV file containing information of the carparks.

- Store information of bookmarks

  Request Data: Carparks' location and name.

  Return Data: Convert a list to an object for the database which stores the carpark's location and name.

API1: Google Places API

- Place Autocomplete

  Request Data: String information of user's desired destination was sent from Park n Go application to the Google Place API via HTTP GET request protocol.

  Return Date: List of predicted matching destinations (*Screen Layout 2*) in Singapore is returned to the application.

- Place Details

  Request Data: A destination selected from the drop-down list of suggested places will be sent to Google Place API via HTTP GET request protocol.

  Return Data: JSON file containing latitude and longitude of the selected destination. This information will be used as the anchor point to search for nearby car parks.

API2: Car park Availability API from Data.gov.sg

  Request Data: A list of car parks from *Database* (storing a dataset of car parks in Singapore) will be sent to Google Place API via HTTP GET request protocol.

  Return Data: JSON file containing car park availability information that is updated real-time.

3.4 Communications Interfaces

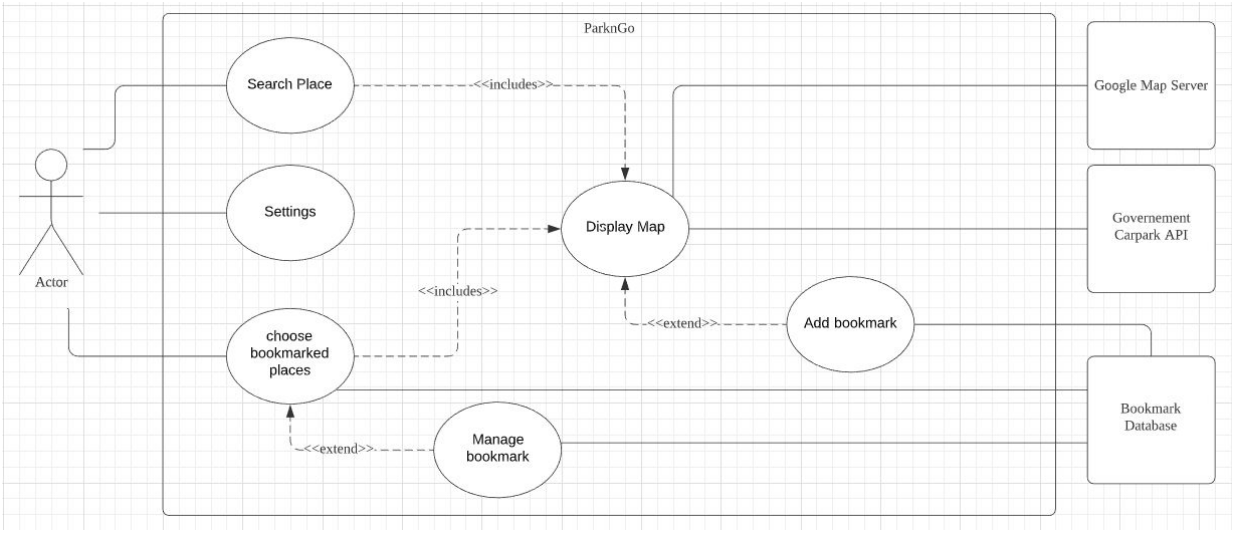
Park n Go requires a steady internet connection and GPS to load data from government data's API and Google Maps API to get the required information of carparks and their occupancies, and current or desired location.

HTTP GET protocol is the main mode of communication between our application and the relevant, aforementioned APIs.

4. System Features

The system features are based on use cases that are grouped by their functionalities.

For detailed use case descriptions of each sub-feature, please refer to the Use Case Descriptions and Use Case Diagram documents and how to obtain them as stated in *1.5.6 Use Case Models*.



4.1 Display Map (default display)

4.1.1 Description and Priority

This feature shows the map of Singapore with all car parks near a selected location, each indicated by a colored marker that shows the relative availability of the car park. When the user pressed on a marker, exact details of the car park will be displayed, including the exact number of parking lots available.

This feature is of high priority since most of the functions in the application use this feature.

4.1.2 Preconditions and Postconditions

Precondition: The location which act as the anchor point of the map is selected by one of the following:

- Upon entering the application, the current location is retrieved from device location service and set as the anchor location.
- User chooses one of the bookmarks and the bookmark is set as the anchor location.
- User selects one of the suggested locations from *4.2 Search Location by Name* and it is set as the anchor location.

Postcondition: A map is displayed, showing car parks near the anchor point. Each car park is indicated with a different coloured marker which shows the relative availability.

4.1.3 Flow of event

1. The system connects to the google map API, retrieves google map and displays it on the screen.
2. The system obtained the anchor location from one of the precondition in *4.1.2 Preconditions and Postconditions*
3. The system shows the car parks near the anchor location using the data from the government car park availability API.
4. The car parks are indicated by markers with different colours to show relative availability of parking lots. (E.g. Red indicate no available parking lot)
5. If the user clicks on any of the car park markers, the system retrieves information about the availability of the car park from API and displays it.

4.2 Search Location by Name

4.2.1 Description and Priority

This feature collects location keywords from the user. Next, it will show a list of potential matching locations for the user to choose. This enables the user to pinpoint the exact location.

This feature is of moderate priority since it is used when the user wants to find the location by searching.

4.2.2 Preconditions and Postconditions

Precondition: User selects the “search” icon on the top right corner

Postcondition: The system receives the exact location to be used as the anchored location and passes to *4.1 Display Map (default display).*

4.2.3 Flow of event

1. App users select the “search” icon.
2. App user entered keywords for the desired location into the search box.
3. System retrieves a list of possible exact locations from google API based on the keywords and displays the list as a drop down list.
    3.1. Suggestion will only appear if the keyword matches locations in Singapore.
4. App users select one of the options in the list and the option is set as the anchored location.
5. The anchored location is passed to *4.1 Display Map (default display)*

4.3 Settings

4.3.1 Description and Priority

This feature allows users to know more information about the application developers.

This feature is of low priority since it is not used in other features of the application.

4.3.2 Preconditions and Postconditions

Precondition: User selects the “menu” icon on the top left corner and selects the “View settings” bar.

Postcondition: Setting page will be displayed where “Contacts” and “About” is displayed upon pressing the “expand” icon

4.3.3 Flow of event

1. User selects “View setting”.
2. User selects the “expand” icon for contacts or about
3. Information about the expanded tab is displayed

4.4 Add Bookmark

4.4.1 Description and Priority

This feature allows the user to save a location into the bookmark bar for future use.This feature.When viewing the information of a car park, the user can click the car park and choose to bookmark that car park.

This feature is of medium priority, as not all users may feel the need to bookmark places, and the main function of checking car park availability is unaffected.

4.4.2 Preconditions and Postconditions

Precondition: A map is displayed using “Display Map” and markers are in place.

Postcondition: Display a message “Bookmark added” and new bookmark is added to bookmark database

4.4.3 Flow of events

1. App user select one of the marker on the map (Either the current/selected location or carpark location)
2. A pop-up appear, showing details of that location marked by the markers and a “Add as bookmark” button
3. App user clicks the button.
4. The bookmark is then added to the bookmark database
5. System display “Bookmark added”

4.5 Select Bookmark

4.5.1 Description and Priority

This feature allows the app user to choose from a list of bookmarked places displayed, and then display that car park on a map.

The priority of this feature is medium, as this feature is only useful for those people that need to park in a place regularly.

4.3.2 Preconditions and Postconditions

Preconditions: App user has bookmarked places stored in bookmark database

Postconditions: One of the following :

- App users select one of the bookmarked places, opening up the map of the place.
- App user exits the tab by tapping on another button

4.3.3 Flow of Event

1. App user select the tab “Choose Bookmarked Places”
2. The system request information from the Bookmark Database
3. If there are bookmarked places stored in Bookmark Database, System will display all the bookmarked places
4. If the App user selects one of the bookmarked places, System will display the selected bookmark location on a map using the DISPLAY MAP use case.

4.6 Manage Bookmark

4.3.1 Description and Priority

This feature allows users to remove bookmarks from “Bookmarked Place”. They can remove bookmarks that they no longer want or need by clicking a button in the view of the bookmark list.

4.3.2 Preconditions and Postconditions

Preconditions: App user pressed “Bookmarked Places” and there exists some bookmarked places stored in bookmark database

Postconditions: A bookmark is removed from “Bookmarked Place” and bookmark database

4.3.3 Flow of Events

1. App user is at “Bookmarked Place”
2. App user presses the “remove bookmark” button on the right of a bookmark.
3. Bookmark Database remove that specific bookmark
4. “Bookmarked Place” is refreshed and shows all bookmarked places except the one that is removed.

5. Nonfunctional Requirements

5.1 Usability

1. The user must be able to get help messages from FAQs.
2. The user must be able to give feedback.

5.2 Reliability

1. The system must be able to refresh data every time the app open.
2. The user must be able to restart the app within 10 seconds if the system crashed.
3. Full system functionality must be restored after system restarts.

5.3 Performance

1. The system must be able to show available car parks with empty parking lots nearby within 2 seconds when the destination location is entered.
2. The system must be able to track search history for 30 days.

5.4 Security

1. The location of the user must not be used by other parties for any other purposes.

5.5 Supportability

1. The database must be replaceable with any commercial product supporting JSON queries.

6. Data Dictionary

Car Park	Place to park cars. Contain parking lots
Parking Lots	Parking spaces which can accommodate only one car per space. There is 2 status for each parking lot - empty or occupied
Bookmarked Place	A tab where previously saved locations are saved by pressing “bookmarked” on the map.
Location service	App user’s location obtained from GPS
API (Application Programming Interface)	A software intermediary that allows two applications to talk to each other, which here is the Carpark availability government API and Google Maps API
Markers	A symbol to indicate a specific location on the map.