Project Mission Statement

Team Snapple is developing a mobile app that enables drivers to plan their route to their destination, by navigating to available and budget-friendly parking spaces.

The project will be considered complete when the mobile application is tested, where it will re-route the user to the most suitable parking space.

This project enables users to navigate to the most suitable parking spaces around their destination, based on price and availability. The users will be able to optimize cost, walking distance to destination and waiting time for an available parking spot, to prevent the users from being overcharged for a parking spot due to a lack of knowledge of other alternative parking spaces.

Functional Requirements

System functionality to be performed

- The user must be able to provide the start and destination location through a search on google maps.

- The user must be able to see available car parks near the chosen destination location.

- The user must be able to list available car parks by price.

- The user must be able to list available car parks by availability.

- The user must be able to know different paths to their destination, which is the user selected car park.

- The user must be given an option to be re-routed to another location if the car park availability falls very low or falls to zero.

- The user must be able to select their desired choice of car parks to park at.

Information to be processed

- The system must display time in 12-hour format for clarity.

- The system must display time taken to reach destination in hh/mm format.

- The system must display the distance from their start point to end destination in km format

- The system must display 1 decimal place.

Interface with other systems

- The system must be able to use google maps API to give users location and car park information for start and destination points.

- The system must be able to use google routes API to give users the best route from A to B. (https://developers.google.com/maps/documentation)

- The system must be able to use GPS to locate user current position.

- The system must be able to retrieve carpark pricing information using car park prices API for comparison purposes.
(https://data.gov.sg/dataset/carpark-rates)

- The system must be able to retrieve HDB carpark availability information using carpark lots API for comparison purposes.
(https://data.gov.sg/dataset/hdb-carpark-information)

- The system must be able to retrieve transport information using public transport API for route planning.

-The system must be able to give an option to adjust routes depending on the car park availability and prices nearby the user's destination.

Term	**Data Type**	**Data Format**	**Field Size**	**Definition**	**Example**
First Name	Text		30	Name of the user	Luke, Samuel
Last Name	Text		30	Surname of the user	Chow, Wong
Start Location	Text		50	Start location of the user's journey	Banyan Hall NTU, NUS Tembusu College

End Location	Text		50	End location of the user's journey	Ion Orchard, Paragon
Distance	Integer	NN	5	Distance travelled by the user (km)	10, 20
Duration	Integer	HH:MM	4	Duration user is parking the car in parking space	01:00, 02:30

Non-Functional Requirements

Usability

- The application must be able to translate into different languages based on the user's phone's setting.

- The interface must have a consistent visual layout.

    - The interface must have a consistent font.

    - The interface must have a consistent colour.

- Every popup must contain informative feedback.

-The application must be responsive.

    -The application must not hang for 5 seconds or more.

Reliability

- The application shall be constantly updated daily according to any changes in car park prices.

- The application shall accurately compute carparks that are near the user's destination.

Performance

- The application must be able to compute options of the car park within 5 seconds.

- The application must be able to update location real-time via GPS signal readings.

- The application must be able to store and display up to 30 days of history and information.

- The application shall boot up within 3 seconds after opening the application.

- The application must be able to retrieve login information and favourites within 5 seconds.

Security

- The database shall make use of reliable cloud drive memory (Amazon/Google) for storage of data.

- User shall be able to migrate the user profile from one device to another by logging into his or her account.

Data Dictionary

Term	Definition
GPS	The Global Positioning System (GPS) is a space-based navigation system that provides the user with the location and time information in all weather conditions.
Routes	Routes are suggested based on car parks availability and prices from the start point to the end destination.
Application	The application retrieves the location on the backend and provides optimized routes and distance.
User	The human user of the application, in this case, a driver.
Car Parks	Car Parks including most forms of parking space, such as HDB car parks, shopping malls, attractions and hotels.

Database	The system that stores all the user profiles, favourite locations and past searches.

DELIVERABLES

• Documentation of functional and non-functional requirements - *Document the requirements in appropriate technical format. The requirements should clearly state who performs what system functionality, taking what input and producing what output. Atomise the requirements such that they are verifiable and traceable. See page 126~130 of Fox for requirements specification heuristics. You will be writing test cases in Lab #4 to verify the requirements. You will also be asked to demonstrate traceability from requirements to the final product.*

• Data dictionary - *Document important terms of your application (e.g., user, device, input, output) in a data dictionary. Explain each term with a brief description. Identify attributes of each term and relationships between terms.*

• Initial Use Case Model, consisting of Use Case diagrams and Use Case descriptions - *From the set of functional requirements, identify the preliminary Use Cases. Depict them on a Use Case diagram using the UML modeling tool. 3.3.2 For each Use Case, start writing the use case description about how the user interacts with the system to carry out the system functionality. As a rule-of-thumb, each Use Case should have a maximum of 6~7 steps in its flow of events. A small Use Case indicates that the functionality has been sliced too finely; a large Use Case can be further broken down. 3.3.3 Iterate over your Use Case model to identify included Use Cases, extended Use Cases, and generalization relationships, if any*

• UI Mockups

Please submit the deliverables to your SVN repository (under the folder “lab1”) before Lab#2 starts. Your Lab Supervisor will want to see and discuss the deliverables with you during Lab #2. The initial Use Case model will be refined and elaborated in Lab#2