

Table of Contents

1. Mission Statement	2
3. Target Audience	3
3. Functional Requirements	3
3.1. Startup page	3
3.1.1 Description	3
3.1.2 Functional Requirements	3
3.2. Map page	5
3.3. Sidebar of the map page	6
3.3.1 Description	6
3.3.2 Functional Requirements	6
3.4. Registration and login	8
3.4.1 Description	8
3.4.2 Functional Requirements	8
3.5. User account features	9
3.4.1 Description	9
3.4.2 Functional Requirements	9
4. Non-functional Requirements	9
5. Data Dictionary	10
6. Use Case Model	15
6.1. Preliminary Use cases	15
6.2. Use case diagram	16
6.3. Use case description	17
Register_001	17
Login_001	19
Login_002	21
Logout_001	23
Startup_001	25
Startup_002	28
Startup_003	30
UserAcc_001	32
UserAcc_002	34
UserAcc_003	36
Map_001	37
Map_002	39
Button_001	41
Button_002	43
Button_003	45

Button_004	47
Button_005	49
Button_006	51
Button_007	53
Button_008	55
Button_009	57
Button_010	59
7. UI Mockups	61

1. Mission Statement

WhereToPark is a web application that enables drivers to effortlessly locate convenient car park locations in close proximity to their destinations. The web application will be furnished with other useful information such as parking fees, number of available lots and height limit of the car parks for users to make an informed decision. WhereToPark utilizes data from Google Maps and Data.gov.sg to enable our users to locate any car park within Singapore.

2. Market Research

With the vehicle population size in Singapore approaching 1 million, there is a need for more car parks to accommodate them. However, with only 12,000 car parks providing around 1.4 million parking lots means that the ratio of cars to car parks is almost saturated. This is very much evident from the increasing difficulty to find car parks in recent years.

3. Target Audience

Our web application targets motorists whose primary mode of transport is private vehicles. Using this web application makes it easier for such people to find parking lots especially during festive seasons or peak periods in areas commonly visited by many. Informed decisions can also be made by them with additional information such as parking fees, available lots and distance from their desired destination also available at their disposal.

3. Functional Requirements

3.1. Startup page

3.1.1 Description

The first page that the user sees when they enter the web application.

3.1.2 Functional Requirements

1. The web application's startup page must display three mutually exclusive options
2. The user must select one of the three option to start the process
 - 2.1. If the user selects the 'Enter WhereToPark Destination' search bar
 - 2.1.1. The user must enter their destination
 - 2.1.2. Destination must be in the form of an entire *address*, *postal code*, or a *general area* in Singapore
 - 2.1.3. The web application must send a request to Google Maps API to confirm whether location exists
 - 2.1.4. The web application must input the destination into the Google Maps API
 - 2.1.5. The web application must direct the user to the application's map page
 - 2.2. If the user selects 'Free Roam'
 - 2.2.1. The web application must direct the user to the web application's map page
 - 2.2.2. The web application must not require users to input a destination
 - 2.3. If the user selects 'Near Me'
 - 2.3.1. The web application must request for permission to access the current location of the user
 - 2.3.2. The user must grant permission to the web application
 - 2.3.3. If permission is not granted
 - 2.3.3.1. Web application must show a pop out to notify users that location services are not enabled
 - 2.3.3.2. Web application must remain at the startup page
 - 2.3.4. If permission is granted
 - 2.3.4.1. Web application must retrieve the user's current location
 - 2.3.4.2. Web application must input the location information into Google Maps API
 - 2.3.4.3. Web application must direct the user to the application's map page

3.2. Map page

3.2.1 Description

The page showcases the map of Singapore, and encompasses the sidebar.

3.2.2 Functional Requirements

1. The map page must first display the entire map of Singapore
2. The map page must then display according to the user's selection from the startup page
3. If the user entered their destination into the 'WhereToPark Destination' search bar
 - 3.1. The map page must then be zoomed into the adjacent area of the destination entered
 - 3.2. The map page must pin all carpark locations within the default radius from the destination
4. If the user selected 'Free Roam'
 - 4.1. The map page must pin all carpark locations available
 - 4.2. The map page must require users to *pinpoint* a central location to access the additional sidebar sorting and filtering features
 - 4.3. If no central location is pinpointed, the sidebar will display all the car parks available in Singapore as they are presented in the government's carpark database
5. If the user selected 'Near Me'
 - 5.1. The map page must be zoomed into the adjacent area of the user's current location
 - 5.2. The map page must pin all carpark locations within the default radius set from the destination
6. The map page must allow users to navigate around the map freely
7. The map page must allow users to zoom in and out of the displayed location
8. The map page must allow users to click on any of the pinned cap park locations to access the particular information of the car park

3.3. Sidebar of the map page

3.3.1 Description

Sidebar icon at the top right of the map page allows the user to open the sidebar to access more in-depth features and information regarding various car parks.

3.3.2 Functional Requirements

1. The map page must have a sidebar icon on the top right corner of the map page
2. The map page must click on the sidebar icon to open the sidebar
3. The sidebar must require users to select how car parks are predominantly sorted
 - 3.1. The user must only select either sorting by *Distance*, *Rate*, or *Available Slots*
 - 3.1.1. By default, *Distance* must be sorted by shortest to longest distance
 - 3.1.2. By default, *Rate* must be sorted by lowest to highest rate
 - 3.1.3. By default, *Available Slots* must be sorted by highest to lowest availability
 - 3.2. Clicking the up-down arrow must allow users to reverse the sorting direction
 - 3.3. If the user chooses to sort by *Distance*
 - 3.3.1. The web application must use the Google Maps API to retrieve all relative distances between all the car parks from the destination
 - 3.3.2. The web application must group and rank car parks according to the distance from the destination
 - 3.3.3. The web application must display the distance accordingly in the sidebar
 - 3.4. If the user chooses to sort by *Rate*
 - 3.4.1. The web application must use the government's carpark database to obtain the latest information regarding the rates
 - 3.4.2. The web application must group and rank car parks according to the rates
 - 3.4.3. The web application must display the results accordingly in the sidebar
 - 3.5. If the user chooses to sort by *Available Slots*
 - 3.5.1. The web application must use the government's carpark database to obtain the latest information regarding the available spaces
 - 3.5.2. The web application must group and rank car parks according to the availability of slots
 - 3.5.3. The web application must display the results accordingly in the sidebar

4. The sidebar must also provide optional secondary customizable car park filtering selections
 - 4.1. The customizations must allow users to further filter the suggested car parks by car park types
 - 4.1.1. The web application must use the government's car park database to obtain the information regarding the car park type
 - 4.2. The customizations must allow users to further filter the suggested car parks that allows for night parking
 - 4.2.1. The web application must use the government's car park database to obtain the information regarding the night parking
 - 4.3. The customization must allow users to further filter the suggested car parks by their vehicle types
 - 4.3.1. The web application must use the government's car park database to obtain the information regarding the vehicle type
 - 4.4. The web application must allow users to filter car parks by car park rates
 - 4.4.1. The web application must use the government's car park database to obtain the information regarding the car park rates
 - 4.5. The web application must allow users to filter car parks by distance to their destination
 - 4.5.1. The web application must use the Google Maps API to retrieve all relative distances between all the car parks from the destination
5. The web application must first sort the car parks by the chosen *Distance*, *Rate*, or *Available Slots*
6. The web application must then further filter the results by the secondary car park filtering customizations, if any
7. The web application must display a list of car parks sorted and filtered as specified by users
8. The web application must allow users to choose their desired car park from the list displayed
 - 8.1. The web application must use Google Maps API to direct the users from their current location to the carpark using the most optimal route
9. The sidebar must allow users to select the option to calculate the estimated price for parking at a specific car park
 - 9.1. Users must enter the duration and timing that they intend to park for at the location
 - 9.2. The web application must use the car park API to obtain the price per hour
 - 9.3. The web application must calculate the estimated price and display it for the user in the sidebar, below the this selection

10. The web application must allow users to select the option to compare with another car parks in the sidebar
 - 10.1. User must select another car park to compare with from the map
 - 10.2. The web application must display all relevant informations regarding the two car parks for the user in a pop-up tab
 - 10.2.1. The web application must display the distance between the user and each car park
 - 10.2.2. The web application must display the distance between each car park and the destination
 - 10.2.3. The web application must display the number of available lots in each car park
 - 10.2.4. The web application must display information regarding the car park type
 - 10.2.5. The web application must display the availability of night parking in each of the car park
 - 10.2.6. The web application must display the vehicle lots that are available in each car park
 - 10.2.7. The web application must display the hourly rate for each car park
11. The web application must allow registered users to view their favorite car parks.

3.4. Registration and login

3.4.1 Description

User icon button at the top of the startup and map page allows users to sign in to an existing account. The account is non-compulsory, and serves to allow users to input preset settings for ease of usage in the future.

3.4.2 Functional Requirements

1. The web application must allow the new users to register for an account
 - 1.1. The web application must request for the following information from the user
 - i. Username
 - ii. Password
 - iii. Confirm Password
 - iv. Email
 - 1.2. The web application must not allow users to register using an existing username
 - 1.3. The web application must not allow users to register using an existing email address
 - 1.4. The web application must allow the user to change their password after registration if they choose to do so
2. The web application must allow users to retrieve their account if they have forgotten their password
 - 2.1. The web application must ask for the user's registered email address

- 2.2. The web application must send an email to the user's registered email address to allow users to reset their password
3. The web application must allow users to retrieve their username if they have forgotten it
 - 3.1. The web application must ask for the user's registered email address
 - 3.2. The web application must query the WhereToParkDatabase for the corresponding username
 - 3.3. The web application must send the username to the user's registered email
4. The web application must allow the user to login if they have an existing account
 - 4.1. The user must login with a valid username and password
 - 4.2. The web application must prompt the user if username does not correspond with password
5. The web application must allow the users to log out if they choose to do so

3.5. User account features

3.4.1 Description

Additional features that come with user accounts.

3.4.2 Functional Requirements

1. The web application must allow the user to set and save preferences
 - 1.1. The user must be allowed to specify their default radius of suggested car parks which are displayed
 - 1.2. The user must be allowed to specify their preferred car park types
 - 1.3. The user must be allowed to specify their preference for car parks that have night parking
 - 1.4. The user must be allowed to specify their vehicle type
2. The web application must allow users to add car parks to their favorite list by selecting the star at the sidebar
 - 2.1. The web application must update the user's favorite list into the WhereToParkDatabase
 - 2.2. The web application must allow users to easily view their favorite car park lists by clicking the favorite button in the sidebar
3. The web application must prioritize the default preferences set by the user when car park searches are done

4. Non-functional Requirements

1. Web application Availability
 - 1.1. Web application shall be operational 24/7
2. Web application Responsiveness
 - 2.1. Every page on the web application shall take less than a minute to load
 - 2.2. Car park suggestions shall be available within a minute
3. Web application Performance
 - 3.1. The route to the carpark shall be calculated within a minute
 - 3.2. Authentication email must be sent within 5 minutes
4. User Friendliness
 - 4.1. Web application shall be properly formatted to the user's device
 - 4.2. Buttons unavailable for use at a particular moment shall be grayed out
 - 4.3. Web application shall predict and suggest destinations regardless of users' minor typo errors
 - 4.4. Sidebar shall cover less than half of the screen
5. Security
 - 5.1. User must authenticate that he is a human through the CAPTCHA security measure before accessing the startup page

5. Data Dictionary

Title	Definition
Car park Name	The official name of the car park or mall/hotel which it is in
Category	The area/region of the car park in Singapore
Car Park No	The number allocated to each car park by the Housing Development Board to uniquely identify them
Address	The street address which belongs to that specific car park, within Singapore. A postal code would also suffice in providing the address.
Coordinates	Formed by the X and Y coordinates, this is to pinpoint the exact location of the car park on a map
Travel Distance	The distance of the car park from the user's current location
Car Park Type	The type of car park. The car park could be a Surface car park or a multi-storey car park
Type of Parking System	There are two types of car parks, those using the coupon system, and those with the Electronic Parking System (EPS)
Short Term Parking	If the car park allows you to park without a valid season parking
General Area	This refers to street names, districts or city areas.
Adjacent Area	The regions within walking distance from the destination
Distance	The length of space between the car park and the user's destination
Price	The total amount in SGD the user will pay to park in a given parking lot for a specific duration
Available Slots	The number of empty lots within the car park
Pinpoint	Selecting a location from the map page during 'Free Roam' to access sidebar features
Pin	A popup which is placed on the map to place more attention at a specific point or entity on the map

Free Parking	When users can park their car for free for a specified time, if free parking is available
Night Parking	When non-season parking holders can park in the car park between 10:30pm to 7:00am under the Night Parking scheme
Car Park Decks	The number of levels which the car park has
Gantry Height	The height of the car park gantry, in meters
Car Park Basement	The car park's underground level
Total Lots	The maximum capacity of the car park
Lot Type	The type of lot which is available within the car park
Weekdays Rate	The parking rates during the weekday for a specified period of time for a given car park
Saturday Rate	The parking rates on Saturday for a specified period of time for a given car park
Sunday and Public Holiday Rate	The parking rates on Sunday or Public Holidays for a specified period of time for a given car park
Monthly Rate	The monthly rate for season parking at that car park
Parking Place Code	The official code for the car park
Season Parking Hours	The hours which the car park is available for season parking
Ticket Type	The type of ticket which the car park provides for season parking
During login: 1. Valid Username/Password 2. Invalid Username/Password	1. Username entered is already in the database and password entered corresponds to its username 2. Username entered is not in the database or the password entered does not corresponds to its username
During reset:	1. Username must not already be in the database

1. Valid Username/Password 2. Invalid Username/Password	2. Username exists in the database
Favorites List	A list of car parks which the user frequently visits or would like to bookmark for ease of future use
User	Anyone who uses the web application
Registered User	User who has successfully logged into an existing account within the app
Guest	User who is not logged into any account
Destination	The location which the user has input into the address bar (the location which we want to search around for car parks nearby)
Radius	Used when we use the “Near me” button, the radius refers to the range from the Destination which we want to see all available car parks within
Default settings	<p>For guest users, default radius, car park type, night parking and vehicle type are pre-defined in the WhereToPark Database.</p> <ul style="list-style-type: none"> - Default radius : 400m - Default car park type: not pre-defined - Default night parking: not pre-defined - Default vehicle type: Car <p>Registered users are able to specify their own preferences for future</p>
WhereToPark Database	Contains users’ account information, default settings and other related data
Carpark Database	Contains Singapore’s carpark information from the government API
WhereToPark Database Helper	Controls access to the WhereToParkDatabase
Carpark Database helper	Controls access to the CarparkDatabase
Location	Controls and interacts with Google Maps API for all location relevant

Helper	services. It also validates location access.
Sidebar	Tab at the right side which encapsulates all the different additional features, and displays relevant car park information
StartUp Page	First page which the user sees once they enter the web application
LoginPage	Page for users to login to their account, gaining access to features unique to registered users
RegisterAccountPage	Page which allows users to create an account
UserAccountPage	Page which allows users to change and specify their own preferences such as search radius or vehicle type
MapPage	Page which contains the map of Singapore, and the toggle for the sidebar

6. Use Case Model

6.1. Preliminary Use cases

1. Register_001 Create an account
2. Login_001 Log into the account
3. Login_002 Change Password
4. Logout_001 Log out of the account
5. User_001 Specific searching
6. User_002 Browse through map
7. User_003 Near Me
8. UserAcc_001 Set and Save Preferred Features
9. UserAcc_002 Save car parks to favorite list
10. UserAcc_003 View Favorite Car Parks
11. Map_001 Search result or current location result
12. Map_002 Free roaming through map
13. Button_001 Change radius
14. Button_002 Filter by available spaces
15. Button_003 Filter by car park types
16. Button_004 Filter by night parking
17. Button_005 Filter by vehicle types
18. Button_006 Routing to selected car park
19. Button_007 Calculating estimated price for parking session
20. Button_008 Compare between car parks
21. Button_009 Filter by car park rates
22. Button_010 Filter by car park distance

6.2. Use case diagram

(Attached as Appendix A)



6.3. Use case description

Register_001

Use Case ID:	Register_001		
Use Case Name:	Create an account		
Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	14/03/2022

Actor:	Guest (Initiating actor)
Description:	Users create an account to log into the web application.
Preconditions:	-
Postconditions:	Username and password are verified and account is created.
Priority:	High
Frequency of Use:	Used once per user if they would like to create an account
Flow of Events:	To create an account:

	<ol style="list-style-type: none"> 1. User enters preferred username, password, confirm password and email 2. User clicks on the signup button 3. System sends the input fields entered by the user to be registered by the WhereToPark server <p>Account created successfully:</p> <ol style="list-style-type: none"> 4. System transfers users to login page
Alternative Flows:	<p>E1. Password and confirm password fields do not match</p> <ol style="list-style-type: none"> 1. User enters preferred username, password, confirm password and email 2. User clicks on the signup button 3. The system sends an error message to inform the user <p>E2. Username or email is already taken</p> <ol style="list-style-type: none"> 1. User enters preferred username, password, confirm password and email 2. User clicks on the signup button 3. The system sends an error message to inform the user 4. User retypes a username or email <p>E3. A required field is empty</p> <ol style="list-style-type: none"> 1. User enters preferred username, password, confirm password and email 2. User clicks on the signup button 3. System detects an empty field 4. The system sends an error message to inform the user
Exceptions:	-
Includes:	-
Special Requirements:	<ol style="list-style-type: none"> 1. Guest must have a laptop/mobile phone

	2. Guest must have stable internet connection
Assumptions:	-
Notes and Issues:	-

Login_001

Use Case ID:	Login_001		
Use Case Name:	Login to account		
Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	15/02/2022

Actor:	Guest (Initiating actor)
Description:	Registered users sign into the application
Preconditions:	-
Postconditions:	User logs into an existing account

Priority:	High
Frequency of Use:	Used once for each distinct user who uses the app and wishes to log in to their account
Flow of Events:	<p>To Login:</p> <ol style="list-style-type: none"> 1. User inputs their username and password and clicks "Login" 2. System authenticates user by matching the username with the password
Alternative Flows:	<p>Forget Password - If the user forgets his password and wishes to recover it:</p> <ol style="list-style-type: none"> 1. User will click on the forget password button 2. System will send an email to the respective user email to provide them with a one time PIN to reset their password <p>Forget Username - If the user forgets his username and wishes to recover it:</p> <ol style="list-style-type: none"> 3. User will click on the forget username button 4. System will ask the user for the email that is attached to the account for authentication 5. System will send an email to the respective user email to inform them of their username
Exceptions:	<p>E1. Invalid User/Password - In the case where the provided username or email does not exist or does not match the password:</p> <ol style="list-style-type: none"> 1. System displays an error message 2. User has to retype username/email or password and try again

Includes:	-
Special Requirements:	<ol style="list-style-type: none"> 1. Guest must have a laptop/mobile phone 2. Guest must have stable internet connection
Assumptions:	-
Notes and Issues:	-

Login_002

Use Case ID:	Login_002		
Use Case Name:	Change Password		
Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	15/02/2022

Actor:	Guest (Initiating actor)
Description:	Registered users can change their current password
Preconditions:	User must be currently logged into an account

Postconditions:	User account has a new password
Priority:	Medium
Frequency of Use:	Used once every time the user wishes to change their account password
Flow of Events:	<ol style="list-style-type: none"> 1. Registered user inputs their current password 2. System sends the input field entered by the user to be verified by the WhereToPark server 3. Register user keys in their new password and confirm password 4. System sends the input field entered by the user to be updated in the WhereToPark server
Alternative Flows:	<p>AF.1 Invalid Password - In the case where the provided password does not match the current password associated with the logged in account</p> <ol style="list-style-type: none"> 3. System displays an error message 4. Registered user will have to enter password again <p>AF.2 New password typed is the same as the previous one</p> <ol style="list-style-type: none"> 3. System displays an error message 4. Registered user re-enters a new password and confirm password
Exceptions:	-
Includes:	-
Special Requirements:	-

Assumptions:	-
Notes and Issues:	-

Logout_001

Use Case ID:	Logout_001		
Use Case Name:	Log out of Account		
Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	15/02/2022

Actor:	Guest (Initiating actor)
Description:	Registered Users can log out of their account
Preconditions:	Users have to be logged into an existing account

Postconditions:	User is logged out of their account
Priority:	High
Frequency of Use:	Used once per session by registered users if they would like log out of their account
Flow of Events:	<p>To log out:</p> <ol style="list-style-type: none"> 1. User presses the log out button 2. System logs user out of current account
Alternative Flows:	
Exceptions:	
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

User_001

Use Case ID:	User_001		
Use Case Name:	Specific searching		
Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	15/02/2022

Actor:	Guest (Initiating actor) Gov.com.sg API Google Maps API
Description:	User arrives at the startup page and chooses to enter a postal code / address / general area into the WhereToPark search box
Preconditions:	Guest needs to know their desired address (postal code, address or landmark)
Postconditions:	Web application sends a request to Google Maps API to confirm the location in Singapore
Priority:	High
Frequency of Use:	At least once every startup session should the user choose this option

Flow of Events:	<p>While location is being typed in:</p> <ol style="list-style-type: none"> 1. User enters a location in Singapore 2. Web application sends a request to Google Maps API to confirm whether location exists 3. Web application comes up with predictions and suggestions for the location from which the user can choose from <p>If user enters valid address/ clicks on the suggested location:</p> <ol style="list-style-type: none"> 4. Location suggestion will be narrowed down to the address that was inputted 5. User confirms the input 6. Web application routes user to next page (Use Case Map_001) <p>If user enters invalid address / does not click on the suggested location:</p> <ol style="list-style-type: none"> 4. Web application will continue to suggest similar addresses to the user's input 5. User chooses and confirms one of the suggested locations 6. Web application routes user to next page (Use Case Map_001)
Alternative Flows:	-

Exceptions:	<p>E1. If a required field is empty:</p> <ol style="list-style-type: none"> 1. The system detects a field is empty 2. The application sends an empty field error 3. Users enters a location in Singapore <p>E2. The actor does not proceed to enter a complete address or click on the location suggestion:</p> <ol style="list-style-type: none"> 1. The system continues to suggest similar addresses to the user's input 2. User either enters a complete address or click on one of the suggest locations
Includes:	Map_001
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

User_002

Use Case ID:	User_002		
Use Case Name:	Browse Through Map		
Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	15/02/2022

Actor:	Guest (Initiating actor)
Description:	User arrives at the startup page and choses to navigate throughout the map
Preconditions:	-
Postconditions:	-
Priority:	High
Frequency of Use:	At least once every startup session should the user choose this option
Flow of Events:	<ol style="list-style-type: none"> 1. User clicks on the 'Navigate Through Map' button 2. Web application routes user to next page

Alternative Flows:	-
Exceptions:	-
Includes:	Map_002
Special Requirements:	<ol style="list-style-type: none"> 1. Guest must have a laptop/mobile phone 2. Guest must have stable internet connection
Assumptions:	-
Notes and Issues:	-

User_003

Use Case ID:	User_003		
Use Case Name:	Near Me		
Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	15/02/2022

Actor:	Guest (Initiating actor)
--------	--------------------------

	Google Maps API
Description:	User arrives at the startup page and chooses to look for car park near their current location
Preconditions:	Guest must enable location services on their device
Postconditions:	Web application sends a request to Google Maps API to confirm user's location in Singapore
Priority:	High
Frequency of Use:	At least once every startup session should the user choose this option
Flow of Events:	<ol style="list-style-type: none"> 1. User clicks on the button 2. Web application tries to obtain permission from user for location data access 3. User consents to providing location data access 4. Web application obtains location data and uses Google Maps API to confirm user's location 5. Web application routes user to next page (Use Case Map_001)
Alternative Flows:	-
Exceptions:	<p>E1. User denies web application's access to their location:</p> <ol style="list-style-type: none"> 1. User clicks on the button 2. Web application tries to obtain permission from user for location data access 3. User does not consent to providing location data access 4. The application sends an error to user

Includes:	Map_001
Special Requirements:	<ol style="list-style-type: none"> 1. Guest must have a laptop/mobile phone 2. Guest must have stable internet connection
Assumptions:	-
Notes and Issues:	-

UserAcc_001

Use Case ID:	UserAcc_001		
Use Case Name:	Set and Save Preferred Features		
Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	15/02/2022

Actor:	<p>Guest (Initiating actor)</p> <p>WhereToPark server</p>
Description:	Registered users can set and make their feature preferences default. The default features preferences

	chosen will be saved to be used during subsequent web application use.
Preconditions:	User must be logged into an account
Postconditions:	Feature preferences are set as default for the specific account
Priority:	High
Frequency of Use:	Registered users may choose this feature more than once
Flow of Events:	<ol style="list-style-type: none"> 1. Registered users choose their preferred choice for the features 2. System sends the selected inputs to the WhereToPark server to be saved
Alternative Flows:	-
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

UserAcc_002

Use Case ID:	UserAcc_002		
Use Case Name:	Save car parks to favorite list		
Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	15/02/2022

Actor:	Guest (Initiating actor) WhereToPark server
Description:	Registered users can save car parks to their favorite list.
Preconditions:	User must be logged into an account
Postconditions:	Selected car parks is added to favorite list of that specific account
Priority:	High
Frequency of Use:	Registered users may use this feature more than once
Flow of Events:	<ol style="list-style-type: none"> 1. Registered users choose their preferred car parks to be added into favorite list 2. System sends selected inputs to the WhereToPark server

	3. WhereToPark server stores the selected inputs to the favorite list specific to that registered account
Alternative Flows:	-
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

UserAcc_003

Use Case ID:	UserAcc_003		
Use Case Name:	View favorites carparks		
Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	15/02/2022

Actor:	Registered user (Initiating actor) WhereToPark server
Description:	Registered users can view car parks in their favorite list under a tab in the sidebar.
Preconditions:	User must be logged into an account
Postconditions:	List of favorite car parks is displayed
Priority:	Medium
Frequency of Use:	Registered users may use this feature more than once
Flow of Events:	<ol style="list-style-type: none"> 1. User selects "Favorites" button 2. Web application retrieves favorite car parks from WhereToPark database 3. Web application displays a list of registered user's favorite car parks
Alternative Flows:	-
Exceptions:	-
Includes:	-
Special Requirements:	-

Assumptions:	-
Notes and Issues:	-

Map_001

Use Case ID:	Map_001		
Use Case Name:	Search result or current location result		
Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	15/02/2022

Actor:	Guest (Initiating actor) Gov.com.sg API Google Maps API
Description:	Web page that will be displayed if the user chose to either input a location into the search box, or to click the button that finds nearest car parks.
Preconditions:	User chose to input a location into search box in previous page OR

	User chose to find nearest car parks to their current location
Postconditions:	Web application will display a map that pinpoints the given location, and highlights all the car parks within a certain radius
Priority:	High
Frequency of Use:	Whenever user searches for a location in the previous page OR Whenever user clicks the button to find car parks near them
Flow of Events:	1. Web application displays a map that pinpoints the user's location, and highlights all the car parks within a certain radius
Alternative Flows:	-
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	
Notes and Issues:	-

Map_002

Use Case ID:	Map_002		
Use Case Name:	Free roaming through map		
Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	15/02/2022

Actor:	Guest (Initiating actor) Gov.com.sg API Google Maps API
Description:	The map that will allow the user to browse through freely and view any car parks across Singapore
Preconditions:	User chose the option for free roaming with the map
Postconditions:	Map will be shown for user to browse through to find a car park if he chooses the option to free roam through the map

Priority:	High
Frequency of Use:	Whenever the user chooses the option for free roaming with the map
Flow of Events:	<ol style="list-style-type: none"> 1. User clicks on the button 2. Web application displays a map that the user can use to free roam with
Alternative Flows:	-
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Button_001

Use Case ID:	Button_001
--------------	------------

Use Case Name:	Change radius		
Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	15/02/2022

Actor:	<p>Guest (Initiating actor)</p> <p>Gov.com.sg API</p> <p>Google Maps API</p>
Description:	Changes the radius of car parks being displayed from the specified location. To use this feature, users must not be in the free roam mode.
Preconditions:	-
Postconditions:	Radius from current specified location increases or decreases, and more/less car parks are being shown
Priority:	Medium
Frequency of Use:	Whenever user toggles the radius filter
Flow of Events:	<ol style="list-style-type: none"> 1. User changes the current radius 2. Web application will adjust and display the car parks within the newly selected radius

Alternative Flows:	For registered users: 1. Web application will filter by the specified radius preference of the user 2. Web application will adjust and display the car parks within the default radius selected
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Button_002

Use Case ID:	Button_002		
Use Case Name:	Filter by Available Spaces		
Created By:		Last Updated By:	

Date Created:	14/2/2022	Date Last Updated:	15/02/2022
---------------	-----------	--------------------	------------

Actor:	<p>Guest (Initiating actor)</p> <p>Gov.com.sg API</p> <p>Google Maps API</p>
Description:	<p>Users may choose to filter the car parks based on the number of available parking lots in each of them. It would be sorted in descending order by default. Users can opt to reverse the sorting order as needed by clicking the up-down button. Users must have already selected the Sorted by Available Spaces' button to use this feature.</p>
Preconditions:	-
Postconditions:	<p>Web application sorts the car parks within the given radius based on the number of available lots in descending/ascending order.</p>
Priority:	Medium
Frequency of Use:	Whenever user toggles the 'Filter by Available Spaces' button
Flow of Events:	<ol style="list-style-type: none"> 1. User inputs whether the car parks should be sorted by the available spaces within a carpark 2. Web application will sort and display the car parks within the given radius of the destination in the sidebar.

Alternative Flows:	-
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Button_003

Use Case ID:	Button_003		
Use Case Name:	Filter by car park types		
Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	15/02/2022

Actor:	Guest (Initiating actor)
--------	--------------------------

	Gov.com.sg API Google Maps API
Description:	Users may choose to filter the suggested car parks based on the type of car park, such as whether they are multi-storey car parks or surface level car parks. To use this feature, the user must have selected the 'Filter by car park types' button.
Preconditions:	-
Postconditions:	Web application sorts the car parks within the given radius based on the type of car park
Priority:	Medium
Frequency of Use:	Whenever the user toggles the 'Filter by Car Park Type' button
Flow of Events:	<ol style="list-style-type: none"> 1. User toggles the 'Filter by Car Park Type' button 2. Web application will sort and display the car parks within the given radius of the destination based on their car park types
Alternative Flows:	<p>For registered users:</p> <ol style="list-style-type: none"> 1. Web application will filter by the preferred car park types of the user 2. Web application will sort and display the car parks within the given radius of the destination, while prioritizing the preferred car park type, set as default
Exceptions:	-

Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Button_004

Use Case ID:	Button_004		
Use Case Name:	Filter by night parking		
Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	15/02/2022

Actor:	Guest (Initiating actor) Gov.com.sg API Google Maps API
--------	---

Description:	Users may choose to filter the car parks by whether the car park allows night parking. The user must have selected the 'Filter by night parking' button to use this feature.
Preconditions:	-
Postconditions:	Web application sorts the car parks within the given radius based on whether the car park allows night parking
Priority:	Medium
Frequency of Use:	Whenever the user toggles the 'Filter by night parking' button
Flow of Events:	<ol style="list-style-type: none"> 1. User inputs whether the car parks should be sorted by availability of night parking 2. Web application will sort and display the car parks within the given radius based on whether the car parks allow night parking
Alternative Flows:	<p>For registered users:</p> <ol style="list-style-type: none"> 1. Web application will determine whether user prefers car parks that have night parking 2. Web application will sort and display the car parks within the given radius, based on the default night parking settings selected by the user
Exceptions:	-
Includes:	-

Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Button_005

Use Case ID:	Button_005		
Use Case Name:	Filter by vehicle types		
Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	15/02/2022

Actor:	<p>Guest (Initiating actor)</p> <p>Gov.com.sg API</p> <p>Google Maps API</p>
Description:	Users may choose to filter the car parks by suitability for their vehicle types. The user must have selected the 'Filter by vehicle type' button.

Preconditions:	-
Postconditions:	Web application sorts the car parks within the given radius based on the suitability for vehicle type(s) selected
Priority:	Medium
Frequency of Use:	Whenever the user toggles the 'Filter by vehicle type' button
Flow of Events:	<ol style="list-style-type: none"> 1. User inputs whether the car parks should be sorted by suitability for the vehicle types selected 2. Web application will sort and display the car parks within the given radius based on the suitability for the vehicle types selected
Alternative Flows:	<p>For registered users,</p> <ol style="list-style-type: none"> 1. Web application will filter car parks that are suitable to the specified vehicle type 2. Web application will sort and display the car parks within the given radius, based on the suitability of the vehicle types selected and saved by the registered user
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-

Notes and Issues:	-
-------------------	---

Button_006

Use Case ID:	Button_006		
Use Case Name:	Routing to selected car park		
Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	15/02/2022

Actor:	Guest (Initiating actor) Gov.com.sg API Google Maps API
Description:	The web application will leverage Google Maps to route the user to a specific car park that they have chosen.
Preconditions:	-
Postconditions:	Web application will open Google Maps and route the user towards their destination

Priority:	Medium
Frequency of Use:	Whenever the user chooses a car park destination
Flow of Events:	<ol style="list-style-type: none"> 1. User chooses a car park that they wish to park at 2. Web application will open Google Maps for the user, with the car park as the destination
Alternative Flows:	-
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Button_007

Use Case ID:	Button_007
Use Case Name:	Calculating estimated price for parking session

Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	15/02/2022

Actor:	Guest (Initiating actor) Gov.com.sg API Google Maps API
Description:	Upon selecting a carpark by clicking on it, the approximate intended parking duration is inputted by the user. The web application will calculate the total estimated parking price by leveraging the Gov.com.sg API to identify parking rates.
Preconditions:	-
Postconditions:	Web application will show the calculated total parking price for the selected car park during the parking session.
Priority:	Medium
Frequency of Use:	Whenever the user toggles the 'Calculate Price' button
Flow of Events:	<ol style="list-style-type: none"> 1. User selects a carpark 2. User inputs the approximate time they intend to park 3. Web application calculates and displays the total parking price

Alternative Flows:	-
Exceptions:	<p>E1. User enters an invalid input (ex alphabet)</p> <ol style="list-style-type: none"> 1. User enters an invalid input 2. User clicks 'calculate price' button 3. System displays an error message
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Button_008

Use Case ID:	Button_008		
Use Case Name:	Compare between car parks		
Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	15/02/2022

Actor:	<p>Guest (Initiating actor)</p> <p>Gov.com.sg API</p> <p>Google Maps API</p>
Description:	<p>If a user has already selected a car park, this button would allow them to select another car park and compare between the two in terms of</p> <ol style="list-style-type: none"> 1. Distance between user and car park 2. Distance between car park and destination 3. Number of available lots in car park 4. Car Park type 5. The availability of night parking in that car park 6. The vehicle types which the car park has lots for 7. The estimated rate for parking between the two car parks
Preconditions:	-
Postconditions:	Users have a side by side information of the two car parks which they want to compare
Priority:	Medium
Frequency of Use:	Whenever they click on the 'Compare between car parks' button
Flow of Events:	<ol style="list-style-type: none"> 1. User already has a selected car park 2. User presses the 'Compare between car park button' 3. Web application requests for the user to pick another car park 4. User selects and confirms another car park 5. Web application displays a side by side view of both car park statistics

Alternative Flows:	-
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Button_009

Use Case ID:	Button_009		
Use Case Name:	Filter by car park rates		
Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	15/02/2022

Actor:	Guest (Initiating actor)
--------	--------------------------

	Gov.com.sg API Google Maps API
Description:	Allows the user to filter car parks by rates
Preconditions:	-
Postconditions:	Car parks will be ranked according to their rates per hour
Priority:	Medium
Frequency of Use:	Whenever user toggles the button
Flow of Events:	<ol style="list-style-type: none"> 1. User toggles the filter button 2. Web application filters the car park by prices
Alternative Flows:	-
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Button_010

Use Case ID:	Button_010		
Use Case Name:	Filter by car park distance		
Created By:		Last Updated By:	
Date Created:	14/2/2022	Date Last Updated:	15/02/2022

Actor:	Guest (Initiating actor) Gov.com.sg API Google Maps API
Description:	Allows the user to filter car parks by distance to their destinations
Preconditions:	-
Postconditions:	Car parks will be ranked according to the distance from the destination (lowest to highest)
Priority:	Medium
Frequency of Use:	Whenever user toggles the button

Flow of Events:	<ol style="list-style-type: none"> 1. User toggles the filter button 2. Web application filters the car park by distance to distance
Alternative Flows:	-
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

7. UI Mockups

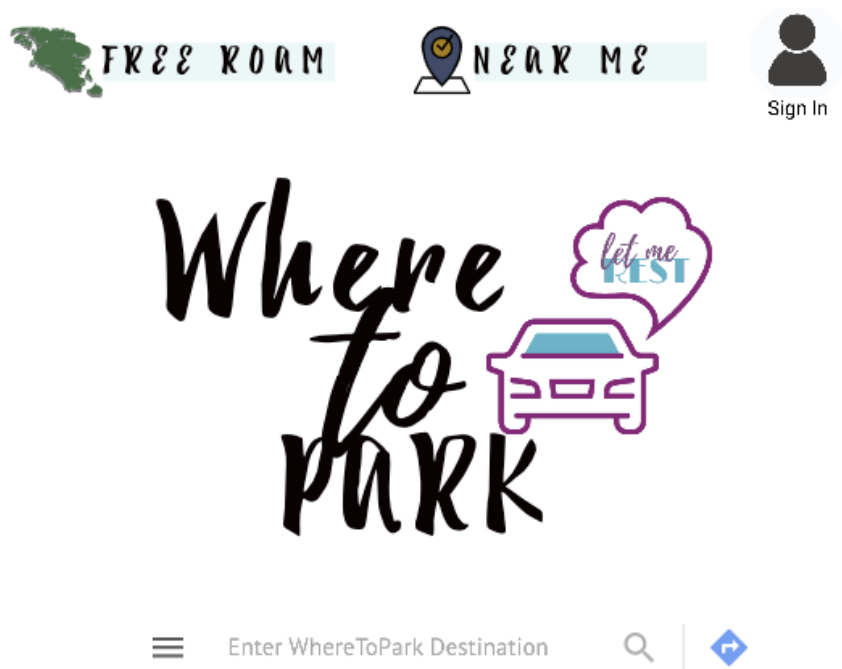


Fig 7.1 Startup page of WhereToPark web application

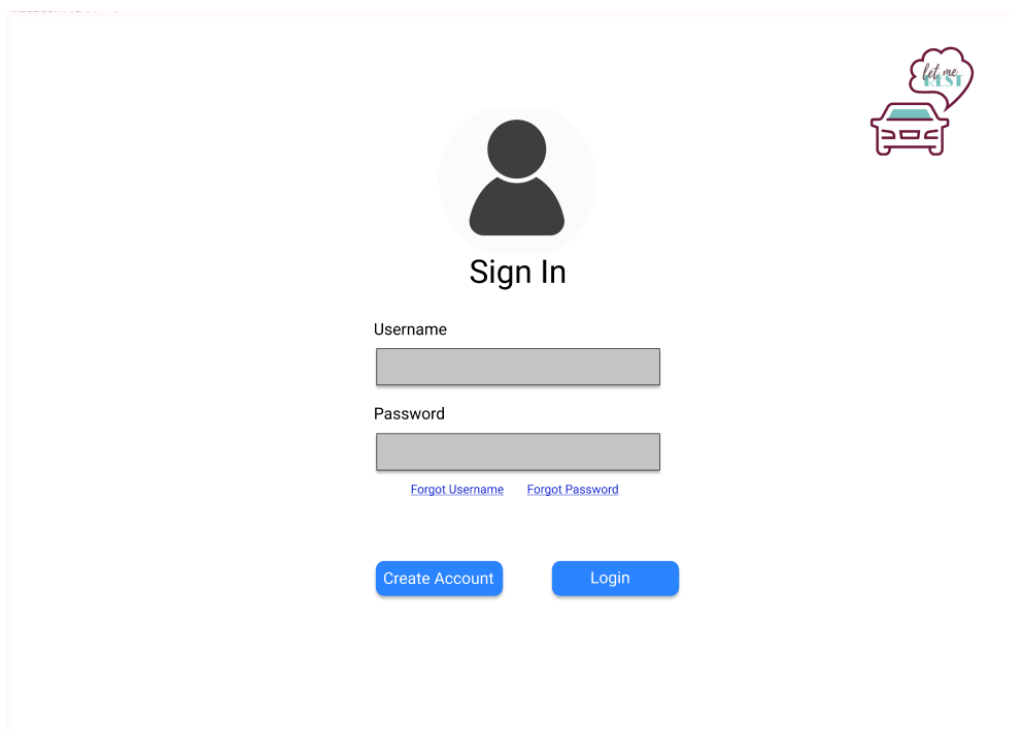




Fig 7.2 Login page





Register Account

Username

Invalid username


Email


Password

Confirm password

Submit

Fig 7.3 Register account page





Username

Radius

Car Park type

Night parking

Yes ☒ No ☐

Vehicle Type

Car

Truck

Bus

Save

Fig 7.4 User account preference page

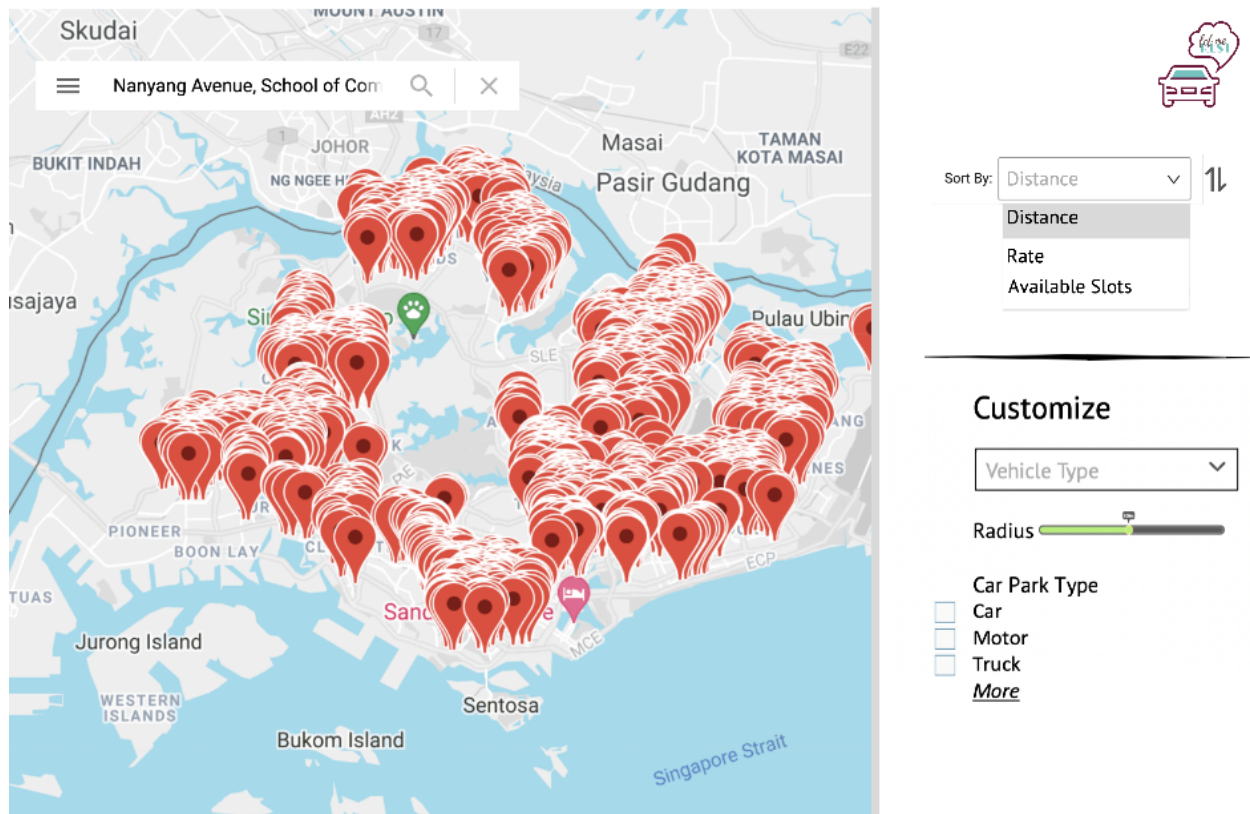


Fig 7.5 Car Park Info Page of WhereToPark web application once a car park is selected

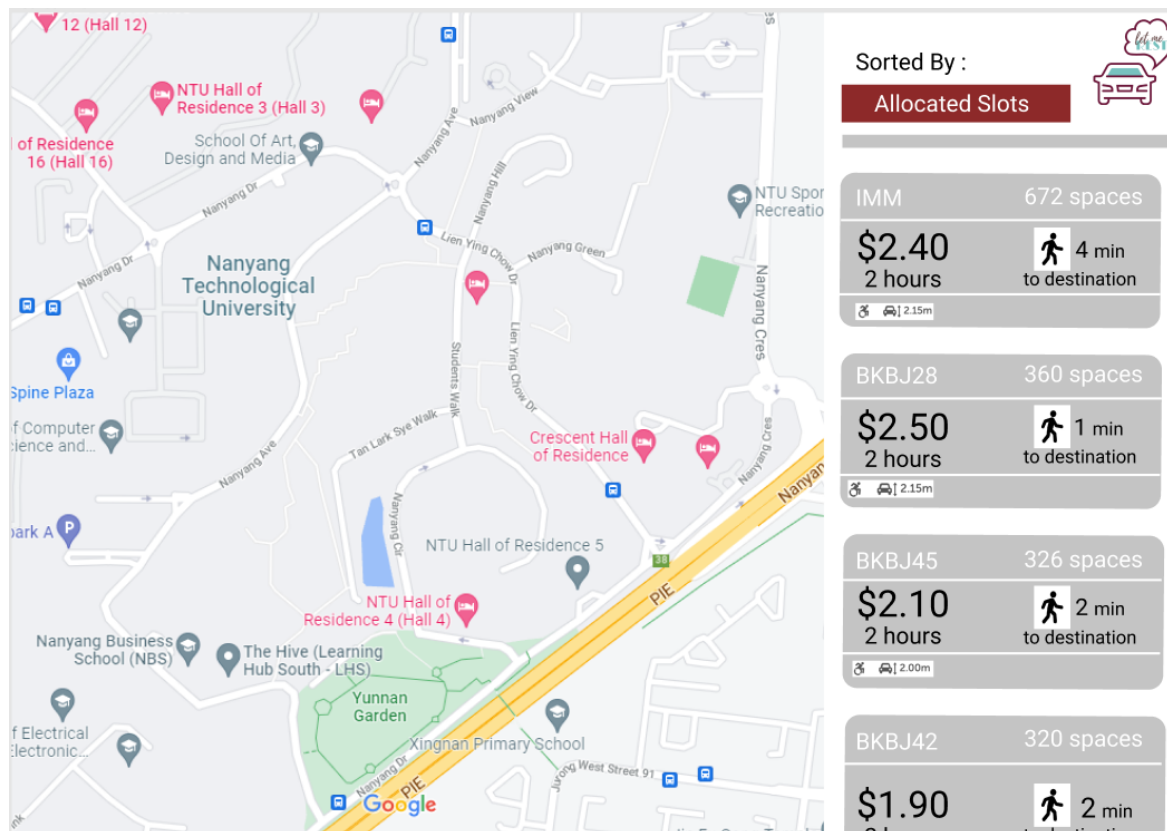


Fig 7.6 Carpark filter function

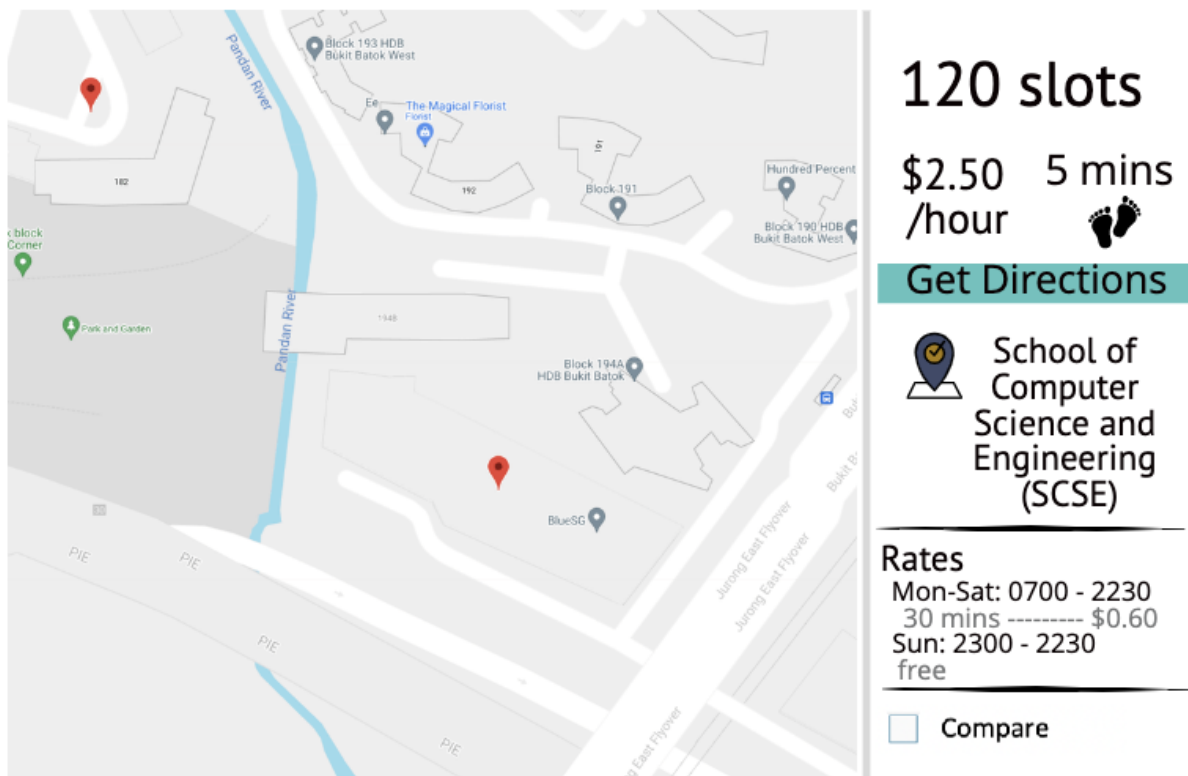


Fig 7.7 Maps Page of WhereToPark web application

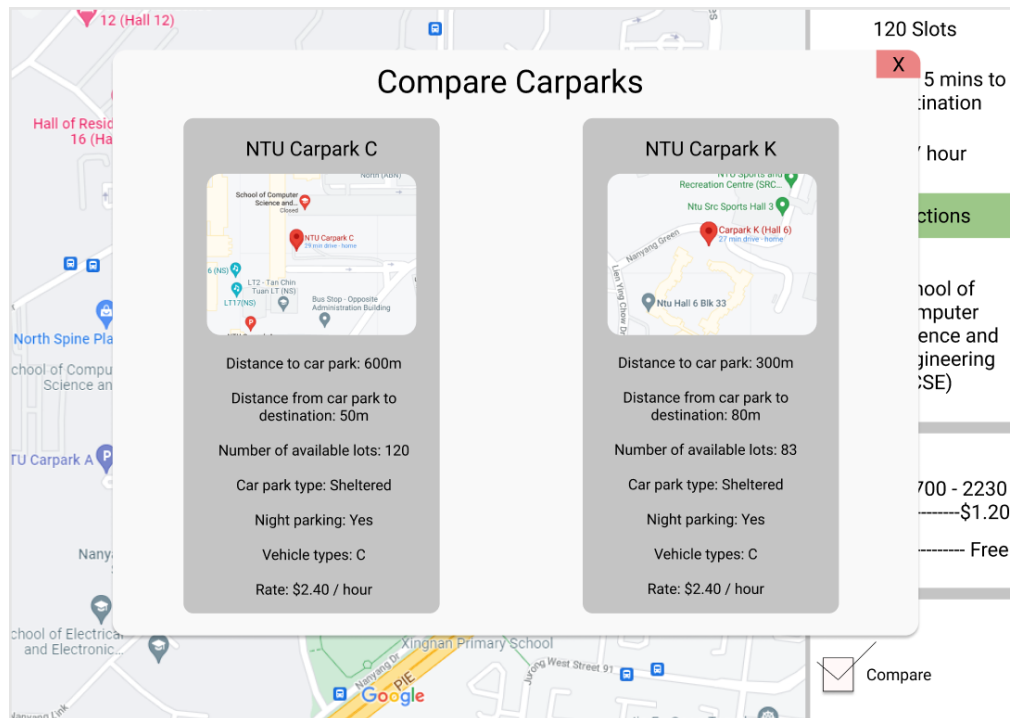


Fig 7.8 Carpark compare function

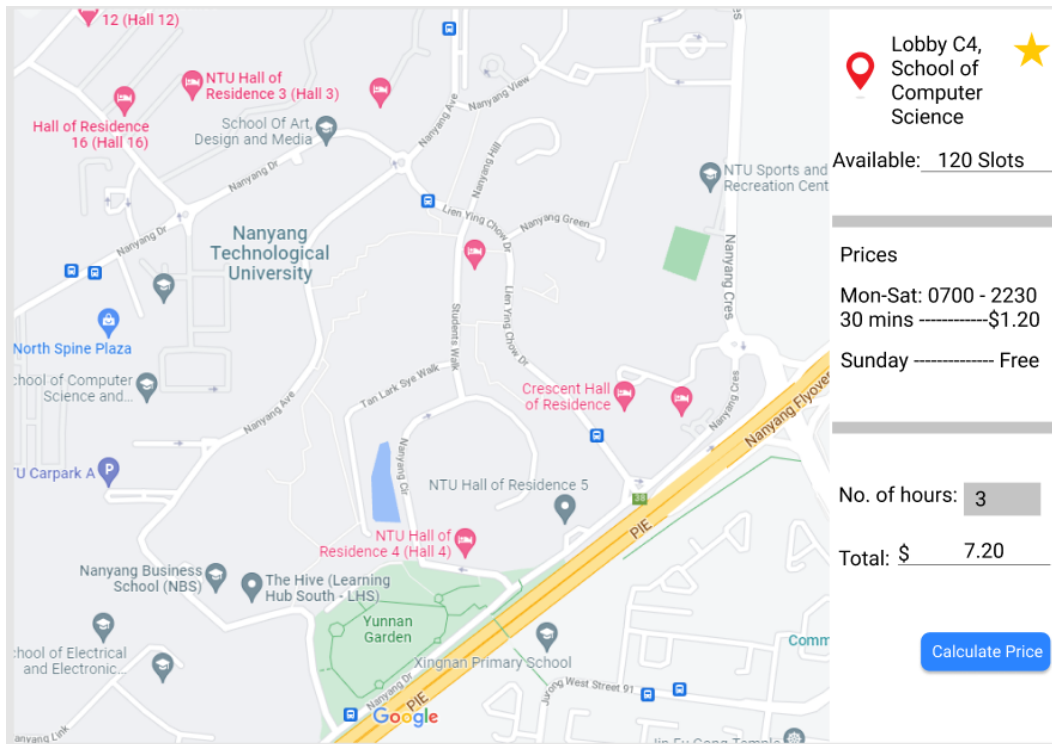


Fig 7.9 Carpark rates calculate price function

