

SC2006: Software Engineering

Lab 1 Deliverables

WanderMap

Ang Yvette (U2220779L)

Chua Sym Eyan (U2221430B)

Tan Hui Ling (U2222280A)

Dan He (N2304782D)

1. Content Page

1. Content Page.....	2
2. Introduction.....	3
3. Functional Requirements.....	4
4. Non-Functional Requirements.....	7
5. Data Dictionary.....	9
6. Use Case Model.....	10
6.1. Use Case Model.....	10
6.2. Use Case Description.....	11
7. UI Mock-ups.....	25
7.1. Login page.....	25
7.2. Landing page.....	25
7.3. Planning page.....	26
7.4. Adding locations.....	28
7.5. Saving itinerary.....	29
7.6. Edit draft.....	29
7.7. Browse: Locations.....	30
7.8. Browse: Categories.....	31
7.9. Browse: Top itineraries.....	33

2. Introduction

WanderMap is a web application that helps tourists plan their trips in Singapore. Users are able to upload attractions they wish to visit to the web application and receive optimised routes to shorten their travel times. Users can also share their routes with others to view and interact with, and browse through routes uploaded by other users for inspiration.

3. Functional Requirements

1. Users shall be able to authenticate their account.
 - 1.1. Users should be able to create an account.
 - 1.1.1. Users must input a valid email address to create an account.
 - 1.1.1.1. The system must be able to check if the email address is in a valid email format.
 - 1.1.1.1.1. If the email is not in a valid format, the system should display an “Invalid email” message.
 - 1.1.1.2. The system must be able to check that repeat uses of emails are not allowed.
 - 1.1.1.2.1. The system should check if the email is already registered in the database.
 - 1.1.1.2.2. If the email is already registered in the database, the system should display a “Email is already in use” message.
 - 1.1.2. Users must input a valid password to create an account.
 - 1.1.2.1. A valid password must meet the requirements of being at least 15 characters long, contain at least 1 uppercase character and at least 1 number.
 - 1.1.2.1.1. If the password does not meet the requirements, the system should display a “Invalid password” message.
 - 1.2. Users should be able to log into their accounts.
 - 1.2.1. Users must key in a previously registered email address and password to log in.
 - 1.2.1.1. The system must check if the email address and corresponding password exists in the database
 - 1.2.1.1.1. If the email does not exist in the database or the password does not correspond to the email, the system should display an “Invalid account” message.
2. Users that have successfully logged in shall be able to plan their travel itinerary.
 - 2.1. Travel itinerary shall include a list of attractions.
 - 2.1.1. The list of attractions must contain a minimum of 1 and a maximum of 5.
 - 2.1.2. Each attraction in the list must have the same format.
 - 2.1.2.1. Each attraction in the list must include a name and an address.

- 2.1.2.2. Users shall be able to include a brief description of no more than 100 words with the attraction.
 - 2.1.2.3. Users shall be able to provide a rating for each attraction.
 - 2.2. If the travel itinerary includes more than one attraction, it shall also include a route between them.
 - 2.2.1. The user must select an attraction to be the start of the route.
 - 2.2.2. The system must be able to generate a route between the attractions with the shortest travel time starting at the selected attraction.
 - 2.3. Users shall be able to manage their unsubmitted travel itineraries.
 - 2.3.1. Users shall be able to add and remove attractions from their travel itinerary.
 - 2.3.2. Users shall be able to save their travel itinerary to edit at a later date.
 - 2.4. Users shall be able to submit their travel itinerary for other users to view.
 - 2.4.1. Users must name their travel itinerary before submission.
 - 2.4.2. Users must rate their travel itinerary from a scale of 0-5 before submission.
 - 2.4.3. Users shall be able to select up to 3 categories to associate with their travel itinerary before submission.
 - 2.4.4. Users shall be able to include a brief description of no more than 500 words with their travel itinerary before submission.
 - 2.4.5. Users shall be able to delete their travel itinerary after submission.
- 3. Users that have successfully logged in shall be able to view travel itinerary submitted by other users.
 - 3.1. The system shall be able to sort and display itineraries according to the categories selected by the user.
 - 3.1.1. Users shall be able to select preset categories
 - 3.1.1.1. Users shall be able to select at most 3 categories
 - 3.1.1.2. The system shall display all itineraries in that category based on the most upvotes, then followed by the most recent.
 - 3.1.1.3. If none of the itineraries have the selected categories, the system shall display 'No items found'.
 - 3.2. The system shall be able to sort and display itineraries containing an attraction selected by the user.
 - 3.2.1. User shall be able to select an attraction

- 3.2.2. If an attraction is selected, the system shall display all itineraries containing the attraction based on the most upvotes, then followed by the most recent.
- 3.3. The system shall be able to sort and display top itineraries.
 - 3.3.1. The system shall display all itineraries based on the most upvotes, then followed by the most recent.
- 3.4. Users shall be able to review the itineraries submitted by other users.
 - 3.4.1. Users shall be able to upvote itineraries by other users.
 - 3.4.2. Users shall be able to bookmark itineraries to view at a later date.

4. Non-Functional Requirements

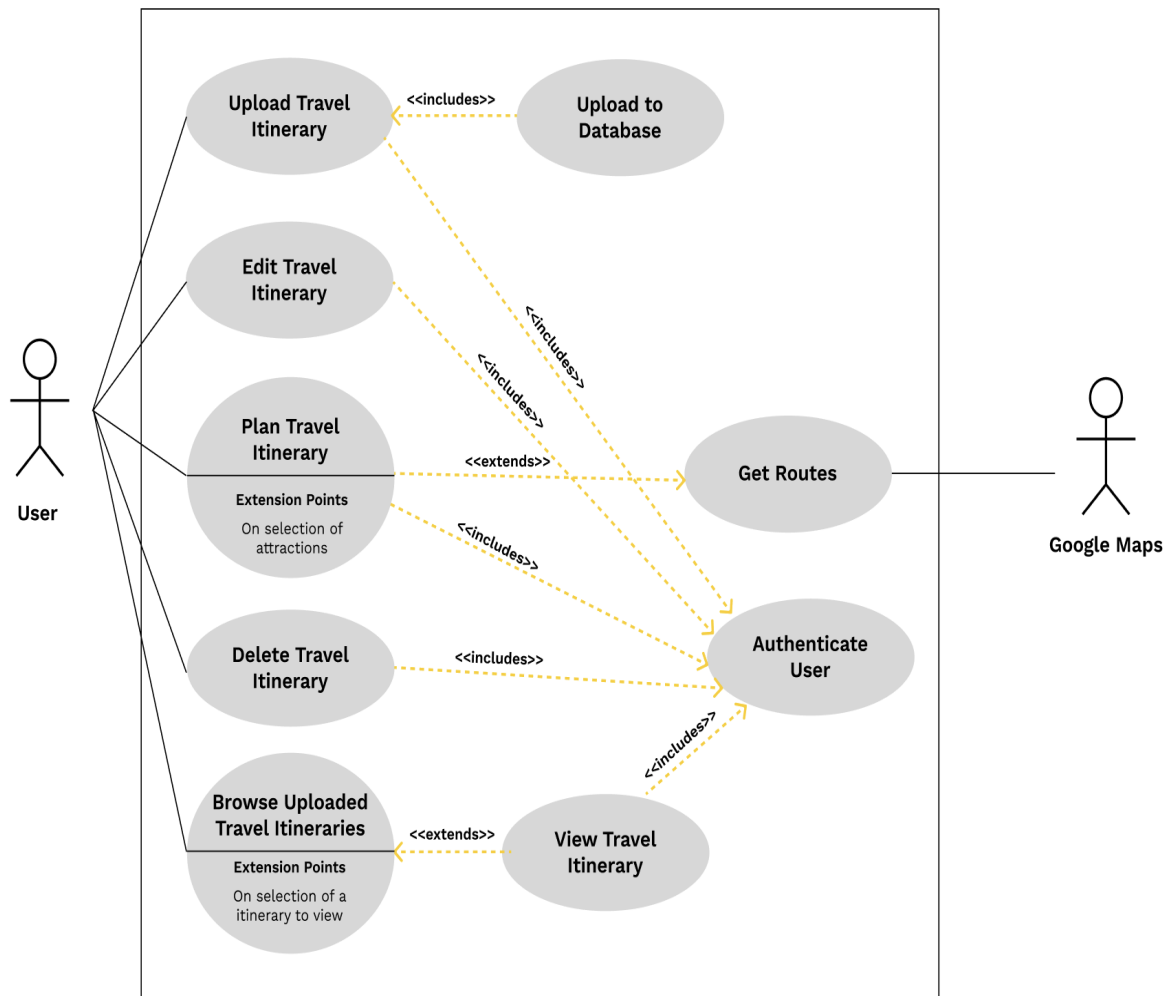
1. Usability
 - 1.1. A new user should be able to complete the account authentication and creation processes within 5 minutes of entering the web application.
 - 1.2. The system should provide error messages with < 25 words for invalid email formats and passwords.
 - 1.3. A new user should be able to create and upload their first itinerary within 10 minutes of entering the web application.
 - 1.4. A new user should be able to find and use the search functions, including the filter function, within 5 minutes of entering the web application.
2. Reliability
 - 2.1. The system shall not have planned downtime of more than 1 hour per year.
3. Performance
 - 3.1. The system must respond to user authentication requests within 5 seconds.
 - 3.2. The system must generate travel itineraries, including routes, within 10 seconds.
 - 3.3. The system should return results for the search functions within 5 seconds.
 - 3.4. The system should be able to handle a minimum of 1000 simultaneous user sessions without significant degradation in performance.
4. Supportability
 - 4.1. The code should be well organised with relevant documentation, to allow for easy readability and maintenance.
5. Scalability
 - 5.1. The system must be able to accommodate a 20% annual increase in user interactions.
6. Security
 - 6.1. User data must be securely stored using encryption techniques such as AES-256.
 - 6.2. The system should implement measures to prevent unauthorised access, with account lockouts after 5 failed login attempts.
 - 6.3. User will be automatically logged out after 30 minutes of inactivity.

5. Data Dictionary

Terms	Definition
Itinerary	A planned schedule of the attractions that the user has selected, that outlines the location and timing for each activity.
Category	Grouping of the attraction based on their shared characteristics. For example, outdoor activities, scenic activities, etc.
Plan	Based on the activities that the user has selected, the app comes up with the sequence and timing for each activity.
Attraction	A point of interest uploaded to the web application that is contains information of the name of location, address, and category of activity.
Upvote	A function of the web page that allows users to cast a vote in support of a given route by pressing a button, which in turn would affect how the route is recommended to other users.

6. Use Case Model

6.1. Use Case Model



6.2. Use Case Description

Use Case ID:	001		
Use Case Name:	Authenticate user		
Created By:	Eyan Chua	Last Updated By:	Eyan Chua
Date Created:	01/02/2024	Date Last Updated:	01/02/2024

Actor:	Users of the website
Description:	All users must be able to log into their account before they are able to plan or post itineraries. This is to provide legitimacy for the itineraries uploaded and protect the quality of posted itineraries.
Precondition:	User has not been authenticated.
Postcondition:	User has successfully signed in or signed up.
Priority:	High
Frequency of Use:	Frequent
Flow of Events:	<ol style="list-style-type: none">1. Users can choose to sign in if they have already created an account, or sign up if they are a first time user.2. If the user chooses to sign up, the app requests for a valid email and password.3. If the user chooses to sign in, the app requests for the email and password that was previously used to create the account.4. If the sign up credentials are valid, the system creates a new account for the user.5. If the sign in credentials are correct, the user is able to access their account.
Alternative Flows:	AF - S4 If the sign up credentials are not valid, the app will display an

	<p>error message and the app will return to step 2.</p> <p>AF - S5</p> <p>If the sign in credentials are not valid, the app will display an error message and the app will return to step 3.</p>
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and issues:	-

Use Case ID:	002		
Use Case Name:	Create and plan travel itinerary		
Created By:	Ang Yvette	Last Updated By:	Ang Yvette
Date Created:	01/02/2024	Date Last Updated:	01/02/2024

Actor:	User
Description:	User can input addresses of different attractions or activities he/she would like to visit. The website would then suggest an itinerary using the shortest path using Google Maps route API.
Precondition:	User has the website opened and an account created.
Postcondition:	User receives a suggested itinerary with estimated time required.
Priority:	Medium
Frequency of Use:	Frequent
Flow of Events:	<ol style="list-style-type: none"> 1. User selects 'Plan travel' option 2. User inputs the locations they would like to travel to. 3. If more then one attraction chosen, user chooses one attraction to be their starting point. 4. The app uses the Google Maps API to calculate and display an end-to-end route from the user's location to the date idea's location and other included locations. 5. The app estimates the time required for the planned date. 6. User selects as many categories they consider the route to be related to. 7. User can choose to discard, save or share the planned route.
Alternative Flows:	<p>AF - S3</p> <p>If the Google Maps API is unable to provide location for the User input location, an error message will be displayed. App returns to</p>

	Step 3.
Exceptions:	-
Includes:	<ol style="list-style-type: none"> 1. Google Maps Routes API to plan routes. 2. Google Maps Places API to get valid businesses.
Special Requirements:	-
Assumptions:	-
Notes and issues:	Time estimation will be based on expected duration at the intended time of travel. The time provided would be an estimation as it does not take into account how much time the user spends at each location.

Use Case ID:	003		
Use Case Name:	Get routes		
Created By:	Tan Hui Ling	Last Updated By:	Tan Hui Ling
Date Created:	01/02/2024	Date Last Updated:	01/02/2024

Actor:	Google Maps
Description:	Retrieve the optimal route between listed attractions in the travel itinerary on an integrated map feature. The optimal route shall start at the selected starting attraction and provide the shortest possible travel time between each attraction. This will help users save time on travelling and hence more effectively plan their day.
Precondition:	User has inputted a minimum of 2 attractions into their travel itinerary.
Postcondition:	System successfully displays the optimal route on the integrated map feature to the user.
Priority:	High
Frequency of Use:	Frequent
Flow of Events:	<ol style="list-style-type: none"> 1. User inputs the addresses of the attractions they wish to visit through text input. 2. For every combination of 2 attractions, the system will input the addresses to Google Maps Routes API. 3. If Google Maps Routes API can find the imputed addresses and a valid route, it returns the details of route to the system. 4. The system will extract and store the travel time information for each combination of attractions. 5. The system will provide the shortest possible travel combination between all attractions, given the selected starting attraction.

	<p>6. The system will input the addresses of the attractions in order of the shortest travel combination to Google Maps Routes API.</p> <p>7. If Google Maps Routes API can find the imputed addresses and a valid route, it returns the details of route to the system.</p> <p>8. The system displays the route returned by Google Maps Routes API on the Integrated Map.</p>
Alternative Flows:	<p>AF - S3</p> <p>If Google Maps Routes API cannot find a valid address for an attraction, the system will return an error message and prompt the user to re-enter the address. The system will return to step 2.</p> <p>AF - S7</p> <p>If Google Maps Routes API cannot find a valid route between attractions, the system will return an error message and prompt the user to re-enter all attractions. The system will return to step 2.</p>
Exceptions:	-
Includes:	-
Special Requirements:	Google Maps Location API
Assumptions:	-
Notes and issues:	-

Use Case ID:	004		
Use Case Name:	Upload travel itinerary		
Created By:	Tan Hui Ling	Last Updated By:	Tan Hui Ling
Date Created:	01/02/2024	Date Last Updated:	01/02/2024

Actor:	User
Description:	Users can upload their travel itineraries onto the website for other users to view and interact with. The travel itinerary includes information such as the rating of the route, addresses of attractions included in the route, and descriptions of the route and the attractions. This allows users to plan out their day with more confidence and efficiency.
Precondition:	User is logged into the system
Postcondition:	Travel itinerary is successfully uploaded onto the website
Priority:	High
Frequency of Use:	Frequent
Flow of Events:	<ol style="list-style-type: none"> 1. User selects "Create new travel itinerary" on the homepage of the website. 2. User selects and adds the attractions they wish to visit or have visited into the itinerary. 3. The user selects to upload the travel itinerary. 4. The system processes the information. 5. The system uploads the travel itinerary and adds it to its database 6. The website displays a confirmation message to the user.
Alternative Flows:	<p>AF - S4</p> <p>If the system is unsuccessful in uploading the travel itinerary, the website will return an error message and prompt the user to try</p>

	again. The system returns to step 2.
Exceptions:	-
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and issues:	-

Use Case ID:	005		
Use Case Name:	Edit travel itinerary		
Created By:	Eyan Chua	Last Updated By:	Eyan Chua
Date Created:	01/02/2024	Date Last Updated:	01/02/2024

Actor:	Users
Description:	User can edit unuploaded itineraries.
Precondition:	User is successfully logged in and has previously uploaded itineraries.
Postcondition:	Itinerary is successfully edited.
Priority:	Medium
Frequency of Use:	Seldom
Flow of Events:	<ol style="list-style-type: none"> 1. User selects an itinerary that they have created and saved but not yet uploaded. 2. User now has the option to edit the itinerary. 3. The user can choose to make amendments to the itinerary through adding or deleting attractions. 4. The app processes the edit request. 5. The app displays a success message if the request has been processed.
Alternative Flows:	AF - S4 If the app is unable to process the request, an error message will be displayed and the app returns back to step 3.
Exceptions:	-
Includes:	<ol style="list-style-type: none"> 1. Authentication of user as a user can only edit their own itineraries.

Special Requirements:	-
Assumptions:	-
Notes and issues:	-

Use Case ID:	006		
Use Case Name:	Delete travel itinerary		
Created By:	Eyan Chua	Last Updated By:	Eyan Chua
Date Created:	01/02/2024	Date Last Updated:	01/02/2024

Actor:	Users
Description:	User can delete their uploaded itineraries.
Precondition:	User is successfully logged in and has previously uploaded itineraries.
Postcondition:	Itinerary is successfully edited or deleted.
Priority:	Medium
Frequency of Use:	Seldom
Flow of Events:	<ol style="list-style-type: none"> 1. User selects an itinerary that they have previously uploaded. 2. User now has the option to delete the itinerary. 3. The app sends a confirmation message to delete the itinerary. 4. The app then processes the delete or edit request. 5. The app displays a success message if the request has been processed.
Alternative Flows:	<p>AF - S2</p> <p>If the app is unable to process the request, an error message will be displayed and the app returns back to step 2.</p>
Exceptions:	-
Includes:	<ol style="list-style-type: none"> 2. Authentication of user as a user can only edit/delete their own itineraries.

Special Requirements:	-
Assumptions:	-
Notes and issues:	-

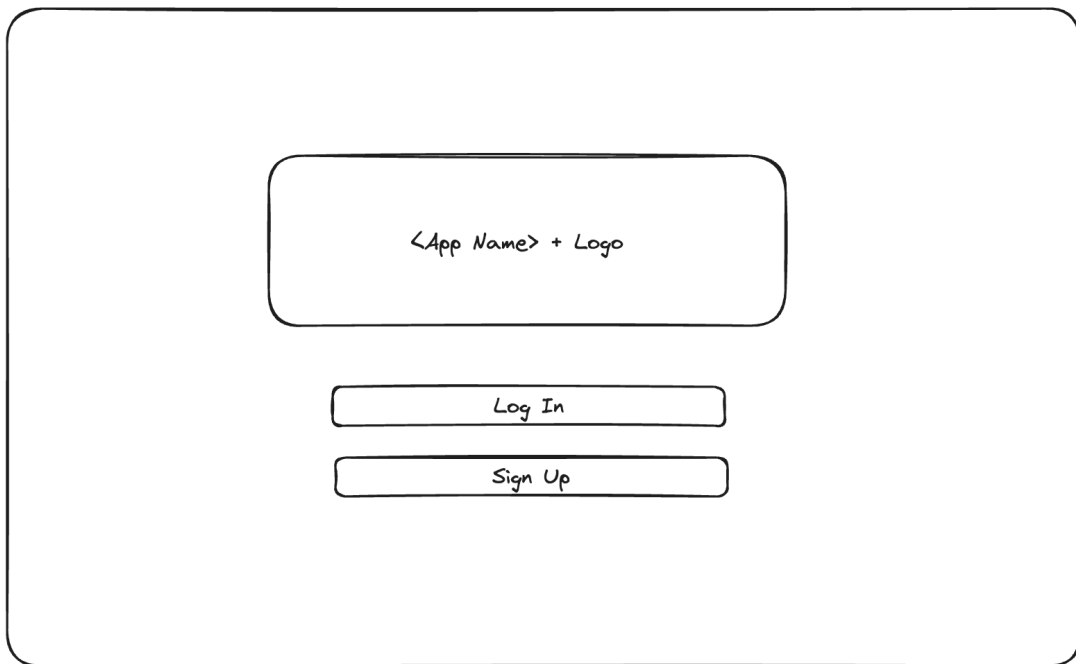
Use Case ID:	007		
Use Case Name:	Browse uploaded travel itineraries		
Created By:	Ang Yvette	Last Updated By:	Ang Yvette
Date Created:	01/02/2024	Date Last Updated:	01/02/2024

Actor:	User
Description:	User can browse travel itineraries uploaded by others. User can filter itineraries according to preset categories (attractions, nightlife, food, sightseeing, chill). User can click into each itinerary to view locations, date, duration and more information provided.
Precondition:	User has a registered account.
Postcondition:	User views a list of itineraries based on applied categories and views further information on the interested travel itinerary.
Priority:	High
Frequency of Use:	Frequent
Flow of Events:	<ol style="list-style-type: none"> 1. User selects the "Browse Travel Ideas" option. 2. User applies categories (e.g., attractions, nightlife, food, sightseeing) to narrow down the list of itinerary ideas. 3. The app retrieves and displays itinerary ideas based on the applied categories. 4. User can click on an itinerary idea to view more information. 5. User views a map feature with locations for that itinerary and a list of information such as price and activities to be done.
Alternative Flows:	<p>AF - S2</p> <p>If no date ideas match the applied filters, the app informs the user accordingly. App returns to Step 2.</p>

Exceptions:	-
Includes:	1. Get shared itinerary from other users
Special Requirements:	-
Assumptions:	-
Notes and issues:	-

7. UI Mock-ups

7.1. Login page



7.2. Landing page



7.3. Planning page

Name of Itinerary:

List Of Attractions:

<Attraction 1> Choose on Map

Description

☆☆☆☆☆

Categories:

Name of Itinerary:

List Of Attractions:


1. Pasir Ris Mangrove Walk

Description

☆☆☆☆☆

Categories:

Pasir Ris Mangrove Walk



Name of Itinerary:

Fun Outing in the East

List Of Attractions:

Starting

1. Pasir Ris Mangrove Walk

2. Downtown East

Description

☆☆☆☆☆

Add

Categories:

Sightseeing

Nightlife

Nature

Discard

Save

Upload

Q

Downtown East

Pasir Ris t

D'Resort @
Downtown East
Modern hotel with a
water park & dining

Pasir Ris Bird
Watching Tower

Pasir Ris Park
Mangrove Boardwalk

Pasir Ris
Park Area 2

Aranda Country Club

Wild Wild Wet

Dian Xiao Er
(Downtown East)

E!Hub

7.4. Adding locations

Name of Itinerary:

Fun Outing in the East

List Of Attractions:

1. Pasir Ris Mangrove Walk Starting ★

↓ 19 min walk

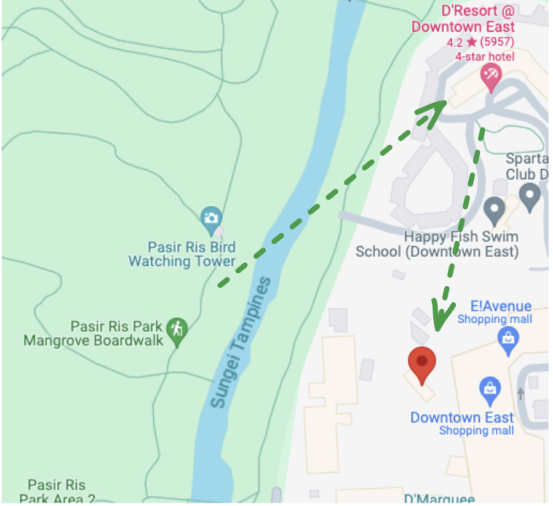
2. Downtown East ☆

3. Wild Wild Wet

Description

Discard Save Upload

Q wild wild wet



The map displays the itinerary route. It starts at Pasir Ris Mangrove Walk, goes to Downtown East (marked with a red pin and a 4.2 star rating), and then to Wild Wild Wet (marked with a red pin). The route is indicated by a dashed green line with arrows. Other locations shown on the map include Pasir Ris Bird Watching Tower, Pasir Ris Park Mangrove Boardwalk, Happy Fish Swim School (Downtown East), ElAvenue Shopping mall, Downtown East Shopping mall, and D'Resort @ Downtown East (4-star hotel). The map also shows the Surgei Tampines river and the Pasir Ris Park Area 2.

7.5. Saving itinerary

Name of Itinerary:

Fun Outing in the East

List Of Attractions:

1. Pasir Ris Mangrove Walk Starting ★

19 min walk

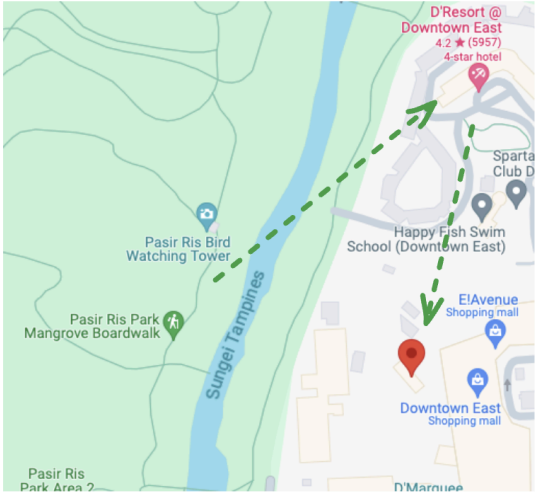
2. Downtown East ★

2 min walk

3. Wild Wild Wet ★

Discard Save Upload

Q wild wild wet



The map displays a route starting at Pasir Ris Mangrove Boardwalk, passing through the Sungei Tampines, and ending at Wild Wild Wet. Other landmarks shown include Pasir Ris Bird Watching Tower, Pasir Ris Park Area 2, D'Resort @ Downtown East, Happy Fish Swim School, Downtown East Shopping mall, and ElAvenue Shopping mall.

7.6. Edit draft

<App Name> + Logo

Your posts Saved Drafts Bookmarked

Fun Outing in the East

Short description of itinerary xxxxx

Delete Edit

Name of Itinerary:

Fun Outing in the East

List Of Attractions:

1. Gallop StablesStarting★

↓ 19 min walk

2. Downtown East★

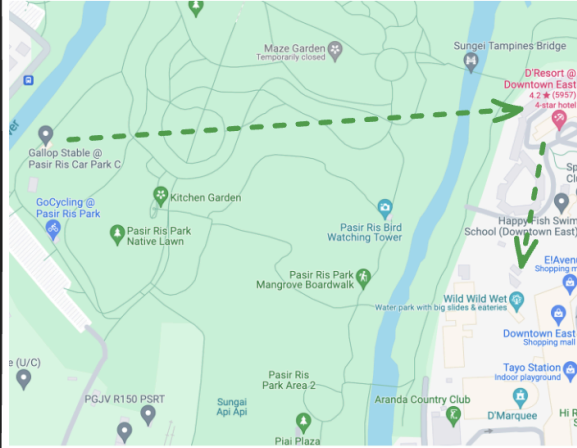
↓ 2 min walk

3. Wild Wild Wet★

Discard

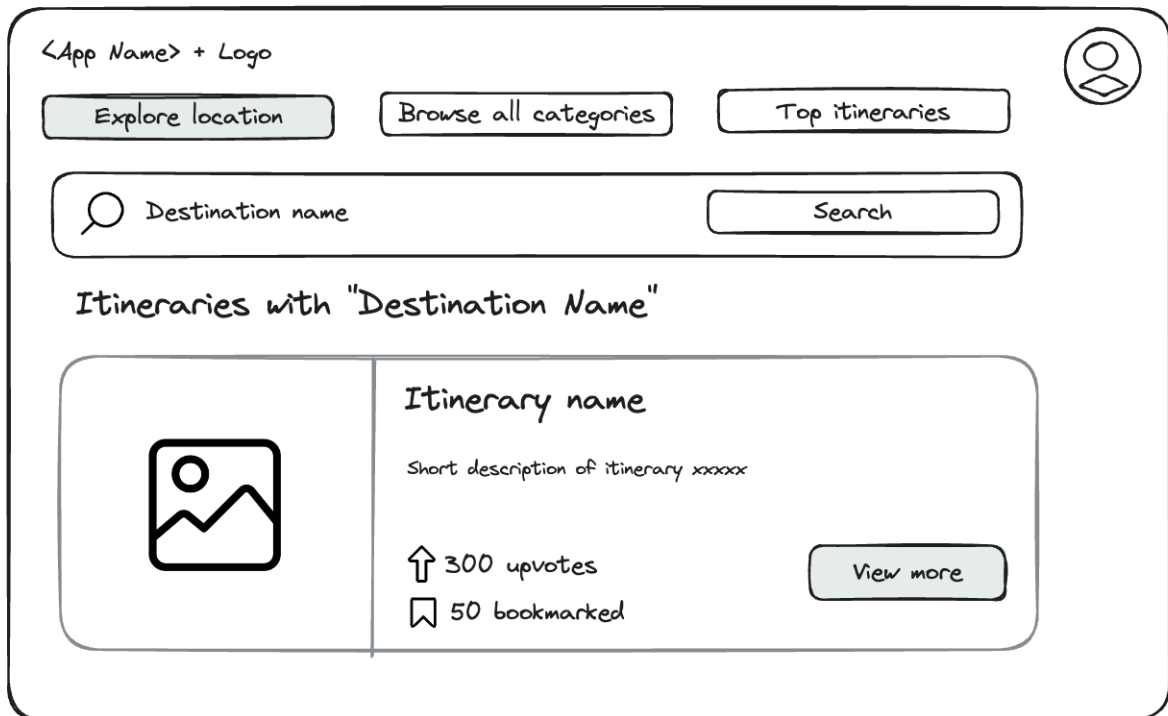
Save

Upload

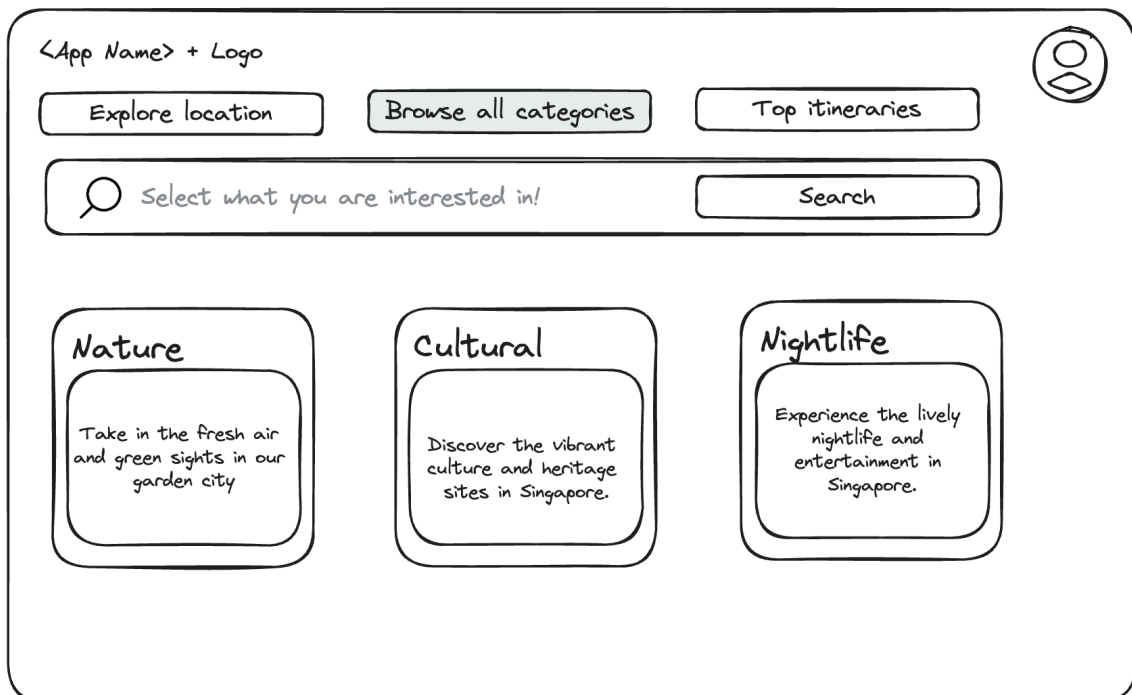


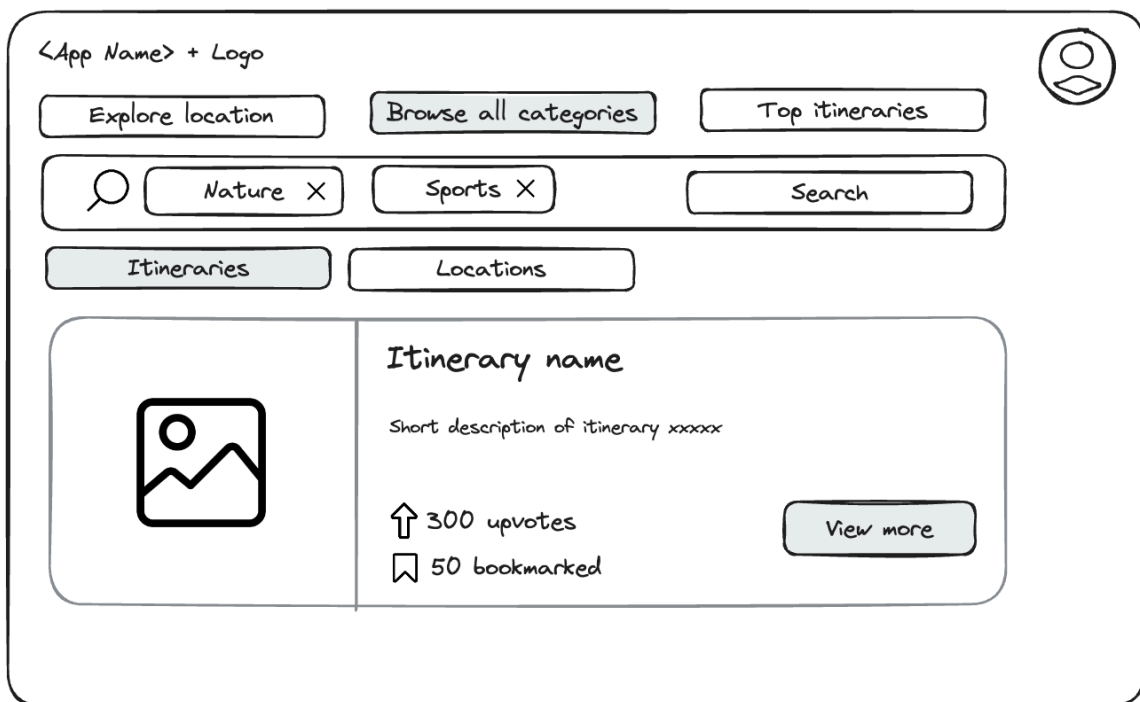
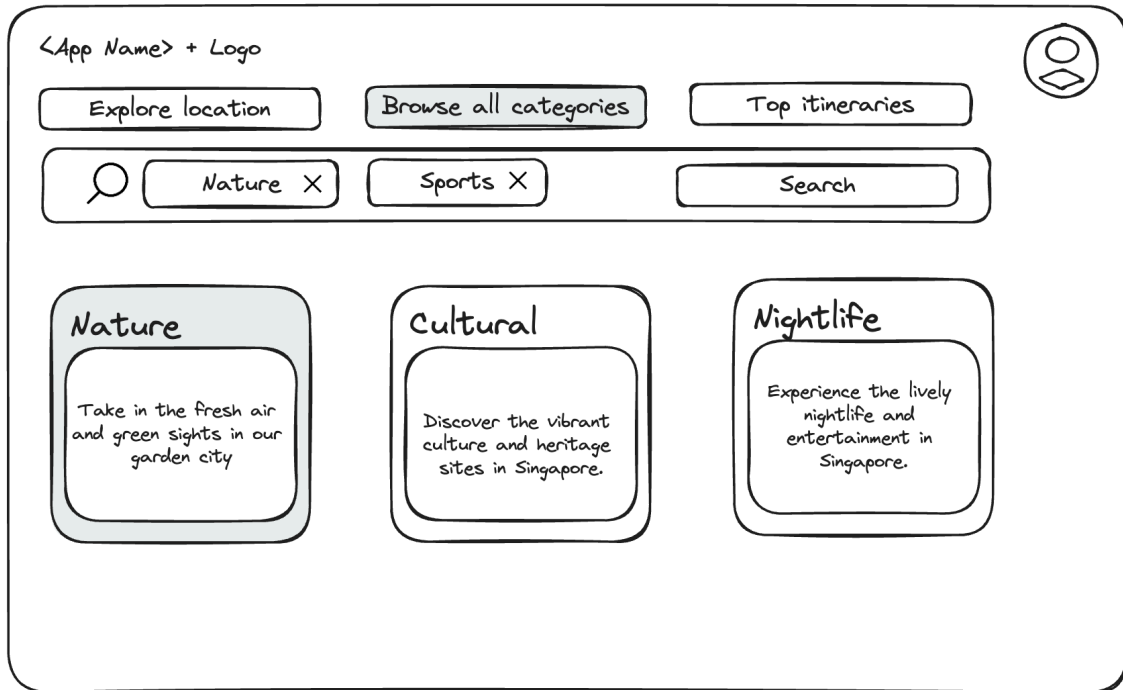
7.7. Browse: Locations

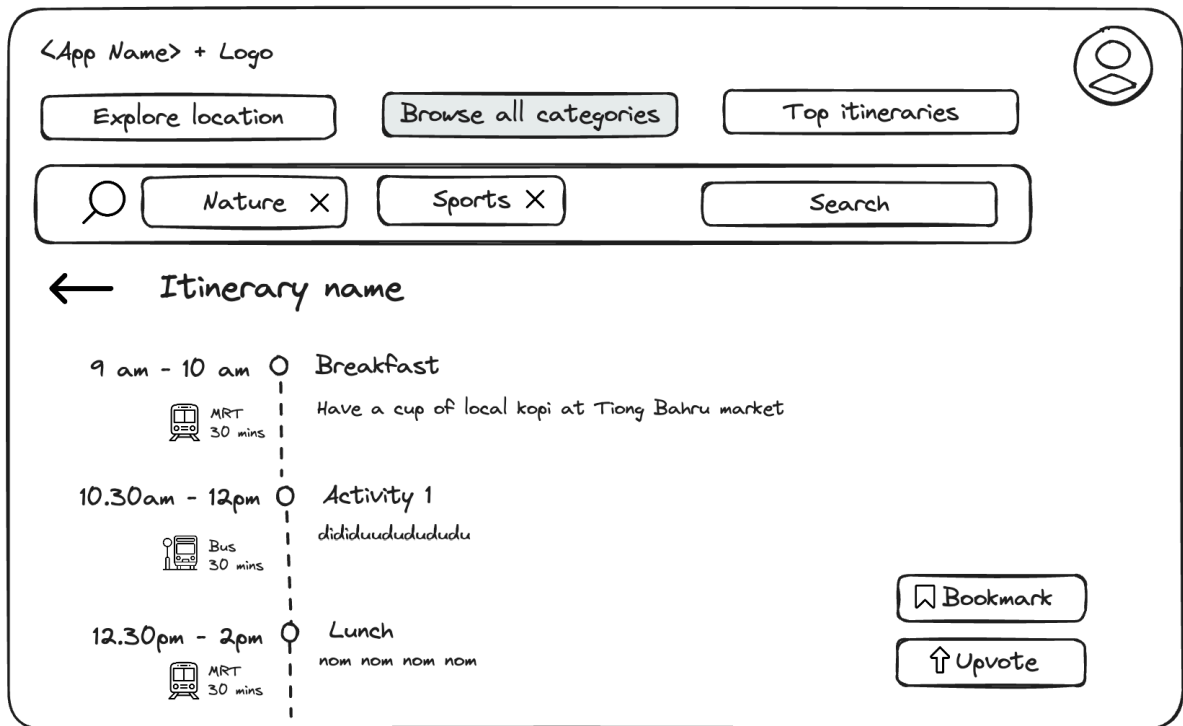
A hand-drawn wireframe of a travel application interface. At the top left, it says "<App Name> + Logo". To the right is a user profile icon. Below these are three buttons: "Explore location", "Browse all categories", and "Top itineraries". A search bar follows, containing a magnifying glass icon, the placeholder text "Enter your next destination!", and a "Search" button. Below the search bar is the heading "Top locations". This is followed by three identical destination cards. Each card features a placeholder image icon, the text "Destination Name", and a "View more" button.



7.8. Browse: Categories







7.9. Browse: Top itineraries

