

---

# Software Requirements Specification

for

## 2ManyFoods

Version 1.0 approved

**Prepared by**

Tan Jun Jie Terence  
Shahrel Chua Zong Yuan  
Tan Zhong Liang  
Kee Jun Xi, Ansel  
Taing Meyling  
Yan Hanxuan  
Zhan Yi Yun

**Nanyang Technological University, Team 2Many**

**27th August 2025**

# Table of Contents

<b>1. Introduction</b>	<b>1</b>
1.1 Purpose	1
1.2 Document Conventions	1
1.3 Intended Audience and Reading Suggestions	1
1.4 Product Scope	2
1.5 References	2
<b>2. Overall Description</b>	<b>2</b>
2.1 Product Perspective	2
2.2 Product Functions	2
2.3 User Classes and Characteristics	3
2.4 Operating Environment	3
2.5 Design and Implementation Constraints	4
2.6 User Documentation	4
2.7 Assumptions and Dependencies	4
<b>3. External Interface Requirements</b>	<b>4</b>
3.1 User Interfaces	4
3.2 Hardware Interfaces	4
3.3 Software Interfaces	5
3.4 Communications Interfaces	5
<b>4. Application Features</b>	<b>5</b>
4.1 Food Searching	5
4.2 Location and Map Integration	6
4.3 Community Features	6
4.4 Group Formation	6
4.5 Group Decision-Making	7
<b>5. Other Nonfunctional Requirements</b>	<b>7</b>
5.1 Performance Requirements	7
5.2 Safety Requirements	8
5.3 Security Requirements	8
5.4 Software Quality Attributes	9
5.5 Business Rules	9
<b>6. Other Requirements</b>	<b>9</b>

## **Revision History**

<b>Name</b>	<b>Date</b>	<b>Reason For Changes</b>	<b>Version</b>

# 1. Introduction

## 1.1 Purpose

*This document specifies the software requirements for the application 2ManyFoods Version 1.*

*With 2ManyFoods, we aim to facilitate people coming together and eating, eliminating the friction that is always involved in coming to a consensus on where to eat. We aim to do so in a personalised manner, tailoring the suggestions to either the individual, or the group they are with, taking into account any dietary restrictions involved.*

## 1.2 Document Conventions

*<Describe any standards or typographical conventions that were followed when writing this SRS, such as fonts or highlighting that have special significance. For example, state whether priorities for higher-level requirements are assumed to be inherited by detailed requirements, or whether every requirement statement is to have its own priority.>*

## 1.3 Intended Audience and Reading Suggestions

This SRS is organised as follows:

- Introduction – the section you are reading right now describes the purpose, scope and definitions
- Overall Description – highlights core concepts and functions of our application
- External Interface Requirements – describes the interaction within and between user, hardware, software and communication interfaces
- application Features – detailed functional requirements, structured feature by feature
- Other Non-functional Requirements – quality attributes of the application
- Other Requirements – regulatory and other non-application constraints
- Appendices – where you'll find extra information about this project including, but not limited to, analysis models, glossary, etc.

Reading suggestions will be underlined.

**All readers should begin with the Introduction and Overall Description**

This document is intended for use by a variety of stakeholders who are involved in the development of the application:

- *Developers:*
  - Will use this document as a reference for functional and functional requirements, to help them in their design and implementation of the application
  - Read application Features, External Interface Requirements and Non-functional Requirements
- *Project Managers:*
  - Will use this to monitor the progress of the project according to the agreed requirements

- Read *application Features* at a summary level, pay attention to *Other Requirements* to ensure development of the application is in alignment with external regulations
- *Marketing and Business Staff:*
  - Will use this document to verify that their business goals and customer needs align with this application's functionality
  - Read selected *application Features* that reflect user benefits and business value
- *End Users and Testers:*
  - Will use this application to design test cases to validate application behaviour
  - Read *application Features*, paying special attention to workflow descriptions
  - Testers follow up on *Non-functional requirements* to derive test cases
- *Documentation Writers:*
  - Will use this document to understand the application capabilities, user interactions and workflows to prepare accurate user manuals and help guides
  - Use *Overall Description*, *application Features* and *External Interface Requirements* as the foundation for user manuals

## 1.4 Product Scope

*2ManyFoods is an application that helps with recommendations of food options. It helps to ease finding food for young adults who may not be as financially well off by providing a range of options within their budget. It helps users who have difficulty in finding food, especially those with dietary restrictions. This application streamlined the view of the kind of places that provide the food options. Food options are recommended according to the user consumption history and preferences.*

## 1.5 References

*<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, application requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.>*

# 2. Overall Description

## 2.1 Product Perspective

2ManyFood consists of a new concept, and does not build off of an existing app

## 2.2 Product Functions

*<Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, so only a high level summary (such as a bullet list) is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or object class diagram, is often effective.>*

At the login page, the users will have to provide their email and password for verification.

**Core User Functions:**

- Restriction Management:
  - Add and manage user food preferences
  - Add and manage dietary requirements
- Search and filtering:
  - Search for food by keywords
  - Filter options by:
    - Healthier choices
    - Price
- Location and Map Integration:
  - Place a pin or enter an address for desired location
  - View eateries in a surrounding area via a map
- Budget Management
  - Receive food recommendations align with budget
- Community Features:
  - Provide feedback on eateries and specific food options

**Group Functions:**

- Group Formations:
  - Form permanent or temporary groups
  - Join groups via invite link
  - Support up to 20 users in a group
- Decision-making:
  - Upon finalizing a group, each user indicate their hunger level (with an option of “no preferences”)
    - This will act as a weightage to prioritise food preferences for recommendations.

## 2.3 User Classes and Characteristics

*<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>*

## 2.4 Operating Environment

*<Describe the environment in which the software will operate, including the hardware platform, operating application and versions, and any other software components or applications with which it must peacefully coexist.>*

**Hardware Platform:**

Users will be accessing the application through a website regardless if its a computer or mobile device.

## 2.5 Design and Implementation Constraints

*<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer's organization will be responsible for maintaining the delivered software).>*

## 2.6 User Documentation

*<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>*

## 2.7 Assumptions and Dependencies

*<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>*

# 3. External Interface Requirements

## 3.1 User Interfaces

*<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>*

**TBC**

## 3.2 Hardware Interfaces

*<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the application. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>*

**TBC**

### 3.3 Software Interfaces

*<Describe the connections between this product and other specific software components (name and version), including databases, operating applications, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the application and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating application), specify this as an implementation constraint.>*

**TBC**

### 3.4 Communications Interfaces

*<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>*

**TBC**

## 4. Application Features

*<This template illustrates organizing the functional requirements for the product by application features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>*

### 4.1 Food Searching

#### 4.1.1 Description and Priority

4.1.1.1 Users can search for food options based on a series of chosen filters  
Priority: High

#### 4.1.2 Stimulus/Response Sequence

4.1.2.1 Stimulus: User chooses their filter  
Response: application provides the results

#### 4.1.3 Functional Requirements

REQ-1            The system shall allow users to search for eateries



- REQ-1.1 The system shall allow users to filter by keyword searches, a given set of cuisines, price
- REQ-2 Results are filtered by the user's preferences and dietary requirements
  - REQ-2.1 Results are filtered by whether the restaurant is open at the current time
  - REQ-2.2 Users are able to customise their preferences
  - REQ-2.3 Users are able to customise their dietary requirements
  - REQ-2.4 Results display information about an eatery
    - REQ-2.4.1 Results include the estimated price range of the eatery
    - REQ-2.4.2 Results include the cuisine of the eatery
    - REQ-2.4.3 Results include the distance of the eatery from the user
- REQ-3 The application deprioritises eateries that sell food that has been chosen by the user recently
- REQ-4 Users are able to select one of the results

## 4.2 Location and Map Integration

### 4.2.1 Description and Priority

4.2.1.1 Users can specify a location and view nearby eateries through a map interface.

- Priority: High

### 4.2.2 Stimulus/Response Sequence

- Stimulus: User pins a location or enters an address, and specifies the radius of the search.
- Response: application displays nearby eateries on the map.

### 4.2.3 Functional Requirements

- REQ-1.1 The system shall allow users to input a location by entering an address.
- REQ-1.2 The system shall allow users to input a location by placing a pin on the map.
- REQ-2 The system shall display eateries within a defined radius of the chosen location on the map.
- REQ-3 The system shall return an error if no eateries are found near the chosen location.

## 4.3 Community Features

### 4.3.1 Description and Priority

4.3.1.1 Users can provide feedback on eateries and food options to enhance community knowledge.

- Priority: Medium

#### 4.3.2 Stimulus/Response Sequence

- Stimulus: User submits feedback or rating.
- Response: The application stores and displays feedback for other users.

#### 4.3.3 Functional Requirements

- REQ-1.1      The system shall allow users to rate eateries
- REQ-1.2      The system shall allow users to rate food items.
- REQ-2        The system shall allow users to provide textual feedback on eateries and food items.
- REQ-3        The system shall make submitted feedback visible to other users.
- REQ-4        The system shall display a message if no reviews are available for an eatery.

### 4.4 Group Formation

#### 4.4.1 Description and Priority

- 4.4.1.1 Users can form permanent or temporary groups.
- 4.4.1.2 Users can join groups via adding username or invite links.
- 4.4.1.3 Users can collaborate on food choices.
- Priority: High

#### 4.4.2 Stimulus/Response Sequence

- Stimulus: User creates or joins a group.
- Response: The application adds the user to the group and enables group features.

#### 4.4.3 Functional Requirements

- REQ-1.1      The system shall allow users to create permanent groups.
- REQ-1.2      The system shall allow users to create temporary groups.
- REQ-2.1      The system shall allow users to join groups via a username
- REQ-2.2      The system shall allow users to join groups via invite links.
- REQ-3        The system must support up to 20 users in a group simultaneously.
- REQ-4        The system must display an error message if a group exceeds the maximum size.

### 4.5 Group Decision-Making

#### 4.5.1 Description and Priority

- 4.5.1.1 Users in a group can indicate hunger levels to weight recommendations and vote on final food options.
- Priority: High

### 4.5.2 Stimulus/Response Sequence

- Stimulus: Users indicate hunger levels or select “no preference” and vote on options.
- Response: application prioritizes recommendations based on group hunger levels and aggregates votes.

### 4.5.3 Functional Requirements

- REQ-1.1 The system shall allow users to indicate hunger level.
- REQ-1.2 The system shall allow users to choose a “no preference” option.
- REQ-2 The system shall weigh user food preferences according to hunger levels.
- REQ-3 The system shall allow group members to vote on recommended food options.
- REQ-4.1 The system must present the most voted option as the final group decision.
- REQ-4.2 The system must display an appropriate message if no votes are cast.

## 5. Other Nonfunctional Requirements

### 5.1 Performance Requirements

*<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time applications. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>*

- The application should also be able to support multiple users using it simultaneously, with no noticeable change in speed when searching or displaying results.
- The application should be able to display results quickly (within 2 seconds) after a user inputs their desired search.
- When a group invite is sent or someone joins a group via link, the application should update to add them into the group within 3 seconds.
- The displayed search results should update within 3 seconds of any changes in filters.

### 5.2 Safety Requirements

*<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product's design or use. Define any safety certifications that must be satisfied.>*

- The application should specify when the food within a certain eatery contains certain allergens such as shellfish or peanuts.
- Photos of food of certain dietary restrictions must be verified by several other users before being considered valid.
- In order to attach a label to an eatery, a threshold of votes must be reached.

### 5.3 Security Requirements

*<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>*

- Users are required to register with a suitably strong password:
  - Not similar to username
  - Does not contain:
    - Name
    - Date of birth
    - Common words such as “password”
  - Contains:
    - A special character
    - Both lower and uppercase characters
    - Both alphabets and numbers
  - Minimum length of 14 characters
- Email at registration required to be verified in order to prevent spam accounts being made.
- Database should be suitably secured, such as hashed and salted passwords, and the database should be safe from SQL injection.

### 5.4 Software Quality Attributes

*<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>*

- The information on the application should be kept up to date, with any increases in price of menu items, closing or opening of new eateries reflected as soon as possible.
- The application should support seamless addition of new eateries.
- The application should be mobile compatible to allow users to use it while on the move.
- The application should be available at all times with minimal downtime.
- Information on availability of certain food items (such as the ones shown by other users) should be updated and validated roughly every week.

## 5.5 Business Rules

*<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>*

- Users should only be able to search and filter for their preference, and are only able to leave certain comments and photos of food sold at eateries that they have selected.

## 6. Other Requirements

*<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>*

- The database should be able to store numerous information about users, such as their usernames, password, email address and groups they are a part of.
- The database should also store information about various eateries, their opening hours, any dietary restrictions they fulfill, type of cuisine, and average price of their menu, and availability of any discounts and food items.
- Queries made to the database should be flexible and simple in order to facilitate the various filters.

## Appendix A: Data Dictionary

*<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>*

<b>Term</b>	<b>Definition</b>	<b>Relationships</b>
<b>User</b>	An actor that indicates their preferences and restrictions.	Uses the application to get recommendations for food options.
<b>Administrator</b>	User with elevated privileges allowing them to manage the application.	Approves the attachment of labels to the eateries.
<b>Group</b>	A group of users inviting each other through links or through user ID.	Users are able to add other users to form a group. Users can hold a vote to decide on the final eatery selection. The collective preferences are taken into account when the recommendations for the group are generated.
<b>Community</b>	Refers to the collective of users.	Votes from the collective of people are required to set non-standard labels on the eateries.
<b>Eatery</b>	Shop that sells food and/or drinks. It can be either halal or non-halal, selling a variety of	Users select an eatery from the list of recommendations A history of eateries is associated with

	cuisines. An eatery can have more than one cuisine.	each user.
<b>Cuisine</b>	Food of a user based on dietary restrictions, preferences and budget.	Eateries may offer certain cuisines and users may choose their preferred cuisine.
<b>Dietary Restrictions</b>	Can be restrictions such as vegan, vegetarian, halal or other medical dietary restrictions.	Users may require specific restrictions regardless of cuisine, which affect recommendations.

## Appendix B: Analysis Models

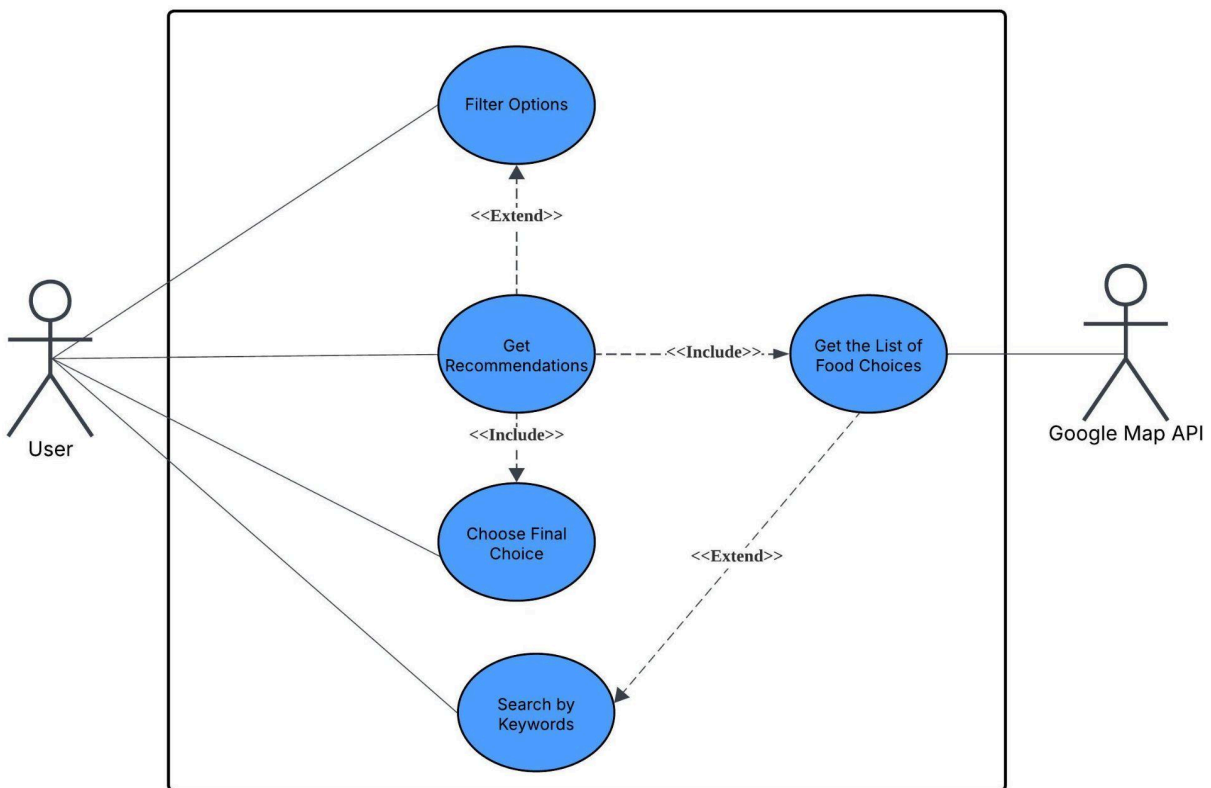
*<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>*

## Appendix C: To Be Determined List

*<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>*

Source: [http://www.frontiernet.net/~kwiegers/process\\_assets/srs\\_template.doc](http://www.frontiernet.net/~kwiegers/process_assets/srs_template.doc)

## Use Case Diagram

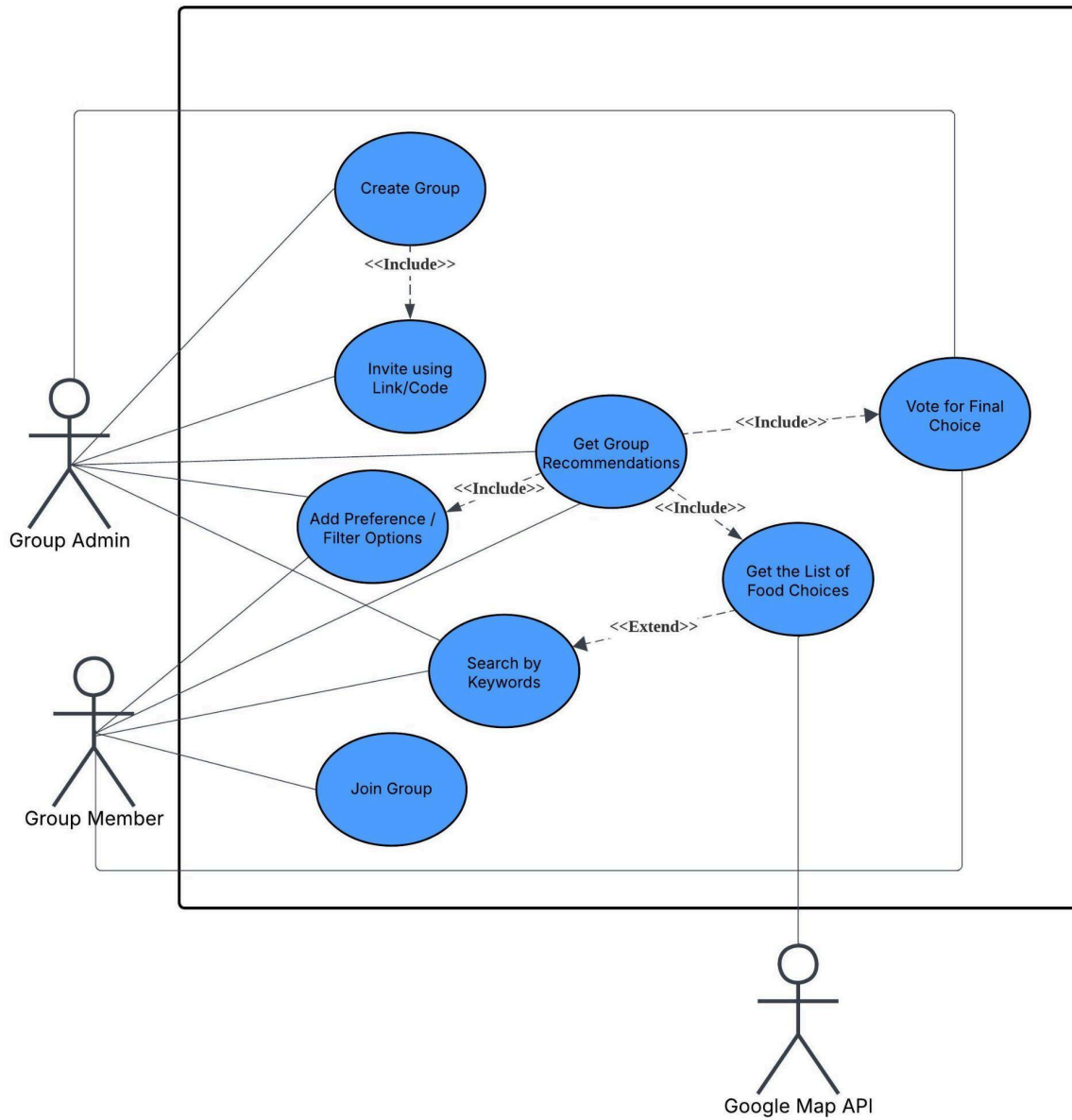


Use Case ID:	IR01		
Use Case Name:	Individual Recommendation		
Created By:	Tan Jun Jie Terence	Last Updated By/Date:	Taing Meyling
Date Created:	28/8/2025		10/09/2025

Actor:	User
Description:	The application recommends a list of food options to the user based on their preference and filter options, allowing the user to choose a final choice.



Preconditions:	<ol style="list-style-type: none"> <li>1. The User must be logged in</li> <li>2. The Google Maps API must be connected</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. The User is able to choose a final food choice</li> <li>2. The final choice is logged</li> </ol>
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> <li>1. User can choose to filter the options or search a keyword <ol style="list-style-type: none"> <li>a. If a keyword is provided, application decided preferences are ignored and only the dietary restrictions are taken into account</li> </ol> </li> <li>2. The application will then give some food recommendations</li> <li>3. From the list of food options, the user chooses the final food choice</li> <li>4. This choice is saved by the application for future reference.</li> </ol>
Alternative Flows:	<ol style="list-style-type: none"> <li>1. After the application outputs the list of food, if the user changes the filter or preference <ol style="list-style-type: none"> <li>a. a new list of food options will have to be generated and displayed</li> </ol> </li> <li>2. If User provides conflicting filters and preferences (e.g. Halal but search for pork), <ol style="list-style-type: none"> <li>a. system detects conflicts and informs the User</li> </ol> </li> <li>3. If no food options are found, <ol style="list-style-type: none"> <li>a. Inform User and ask User to modify search criteria</li> </ol> </li> </ol>
Exceptions:	<ol style="list-style-type: none"> <li>1. The User does not select a final food choice <ol style="list-style-type: none"> <li>a. The application will not save anything to the user history</li> </ol> </li> <li>2. If the Google API drops due to connection timeout or any other reason, <ol style="list-style-type: none"> <li>a. Display an error message asking the user to try again in 5 minutes</li> <li>b. Log the error and notify the System Admin about the issue</li> </ol> </li> <li>3. If the User is not logged in <ol style="list-style-type: none"> <li>a. Prompt them to log in at <a href="#">L01</a></li> </ol> </li> </ol>
Includes:	-
Special Requirements:	-
Assumptions:	<ol style="list-style-type: none"> <li>1. The Google Maps API is connected</li> </ol>
Notes and Issues:	-

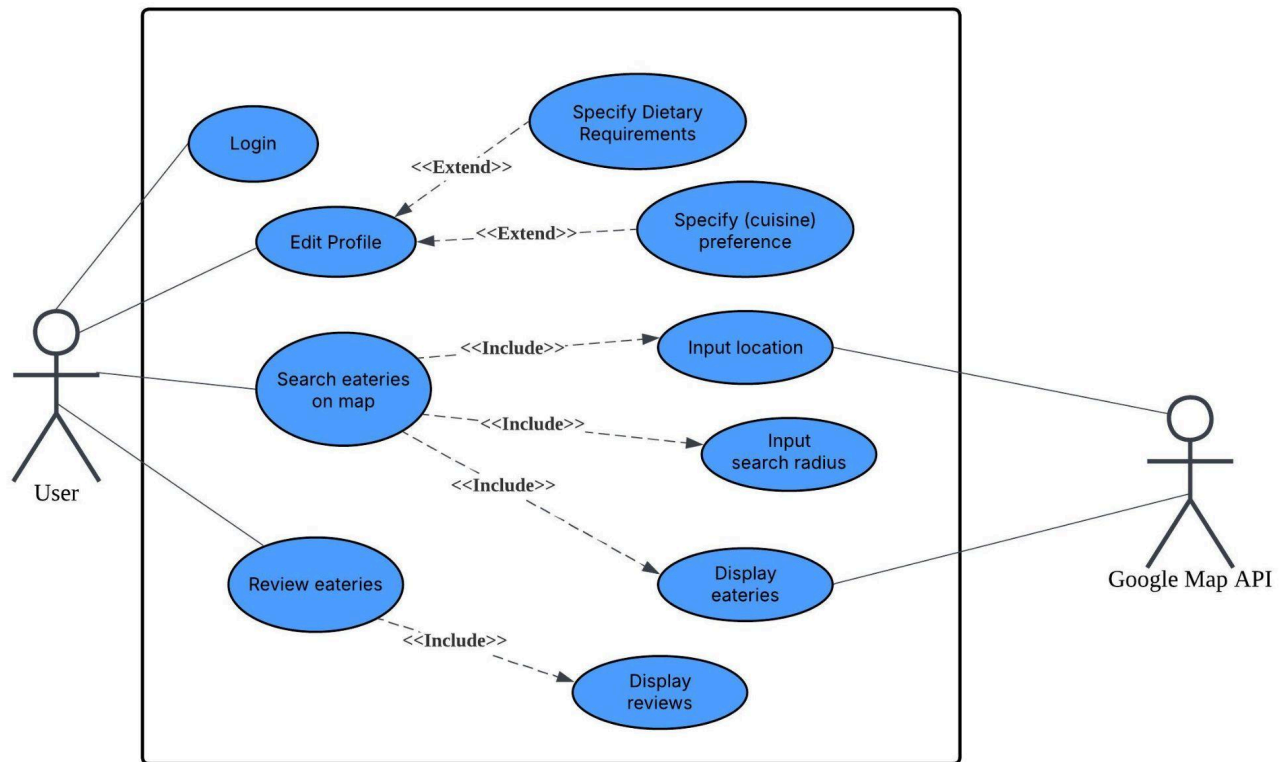


Use Case ID:	GR01		
Use Case Name:	Group Recommendation		
Created By:	Tan Jun Jie Terence	Last Updated By:	Tan Jun Jie Terence

Date Created:	28/8/2025	Date Last Updated:	10/9/2025
---------------	-----------	--------------------	-----------

Actor:	User
Description:	The application allows users to create a group and invite people to join. Recommendations are then chosen based on the collective preferences and restrictions, with weightage provided by the group members. The users in the group can then vote to decide which food option to choose from the list.
Preconditions:	<ol style="list-style-type: none"> <li>1. The User must be logged in</li> <li>2. User must be part of a group</li> <li>3. The Google Maps API must be connected</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. A final food choice will be selected after the group voting</li> </ol>
Priority:	High
Frequency of Use:	Mid
Flow of Events:	<ol style="list-style-type: none"> <li>1. The user is able to create a group and send invites to other users using a link/code.</li> <li>2. Other users will then join the group using the link/code</li> <li>3. The users in the group can choose to filter the options/add their preference or search a keyword <ol style="list-style-type: none"> <li>a. If a keyword is provided, application decided preferences are ignored and only the dietary restrictions are taken into account</li> </ol> </li> <li>4. There will then be a group voting to decide the final food choice.</li> </ol>
Alternative Flows:	<ol style="list-style-type: none"> <li>1. After the application generates the list of food options for voting. However, if new members are added the recommendations, <ol style="list-style-type: none"> <li>a. filtering and preference will be reset and</li> <li>b. a new set of recommendations will be generated based on the new preferences and filters.</li> </ol> </li> <li>2. The group voting ended up in a draw <ol style="list-style-type: none"> <li>a. The application will randomise and choose one of the food options.</li> </ol> </li> </ol>
Exceptions:	<ol style="list-style-type: none"> <li>1. If the page crashes for a member, their vote is considered void for that voting and the app will display a message to inform the member.</li> </ol>
Includes:	-

Special Requirements:	-
Assumptions:	1. The Google Maps API is connected
Notes and Issues:	-



Use Case ID:	L01		
Use Case Name:	Login		
Created By:	Zhan Yi Yun	Last Updated By:	Zhan Yi Yun
Date Created:	28/08/25	Date Last Updated:	10/9/2025

Actor:	User
--------	------

Description:	User shall provide valid email account and password to access the application
Preconditions:	1. User is registered in the application
Postconditions:	User gets verified and redirected to the application's main dashboard
Priority:	High
Frequency of Use:	Low
Flow of Events:	<ol style="list-style-type: none"> <li>1. User enters email and password</li> <li>2. User clicks "Login"</li> <li>3. application verifies user's credentials</li> <li>4. application redirects users to dashboard if user's credentials are valid</li> </ol>
Alternative Flows:	<p>AF-S1: User forgets password</p> <ol style="list-style-type: none"> <li>1. User clicks "Forget Password"</li> <li>2. User enters email and clicks "Submit"</li> <li>3. Application checks if email exists in system database</li> <li>4. Application generates a unique password reset token and sends an email to the user with a link containing the token</li> <li>5. User clicks on the link in their email to reset their password</li> <li>6. User is redirected to reset password page and enters new password</li> <li>7. Application validates new password and updates the user's password in the database</li> <li>8. User is redirected to login page</li> </ol>
Exceptions:	<ol style="list-style-type: none"> <li>1. If User email is not found in the system, Application displays error message suggesting that email is not associated with any account</li> <li>2. If a user enters wrong credentials 3 times, the account is locked for 5 minutes.</li> </ol>
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	EP01		
Use Case Name:	Edit Profile		
Created By:	Zhan Yi Yun	Last Updated By:	Zhan Yi Yun
Date Created:	28/08/25	Date Last Updated:	10/9/2025

Actor:	User
Description:	User update personal profile data (such as username, profile photo, dietary requirements and dietary preferences).
Preconditions:	<ol style="list-style-type: none"> <li>1. User logged in</li> <li>2. User has existing profile</li> </ol>
Postconditions:	Changes successfully reflected in user's profile.
Priority:	Medium
Frequency of Use:	Low
Flow of Events:	<ol style="list-style-type: none"> <li>1. User navigates to their profile</li> <li>2. User clicks "Edit Profile"</li> <li>3. User updates fields (e.g username, profile photo, dietary requirement, dietary preferences etc)</li> <li>4. User clicks "Save changes"</li> <li>5. application validates and saves changes</li> <li>6. Application updates the database</li> <li>7. application displays successful update.</li> </ol>
Alternative Flows:	User cancels edit before saving: <ol style="list-style-type: none"> <li>1. User clicks "Cancel"</li> <li>2. application redirects user back to previous page</li> <li>3. User attributes remain unchanged</li> </ol>
Exceptions:	The new inputs are not valid, the app will not save and display an error message: <ol style="list-style-type: none"> <li>1. New username might not be available</li> <li>2. Uploaded photo is not supported</li> </ol>
Includes:	<a href="#">SR01</a> , <a href="#">SP01</a>

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

Use Case ID:	SR01		
Use Case Name:	Specify Dietary Requirements		
Created By:	Zhan Yi Yun	Last Updated By:	Zhan Yi Yun
Date Created:	28/08/25	Date Last Updated:	10/9/2025

Actor:	User
Description:	User select one or more dietary requirement(s) to personalise their application experience. (application will prioritise food recommendations based on user's selected dietary requirement(s))
Preconditions:	1. User logged in 2. User has access to their profile settings
Postconditions:	1. Dietary requirements saved and correctly displayed on user's profile 2. Food recommendations by application prioritized based on user's dietary requirements
Priority:	High
Frequency of Use:	Low
Flow of Events:	1. User navigates to their profile 2. User clicks "Edit Profile" 3. User selects the field "Dietary Requirements" 4. Application displays list of dietary options (eg halal, vegan, vegetarian, gluten-free etc) 5. User selects one or more options. 6. User clicks "Save changes" 7. Application validates and saves changes 8. Application displays a successful update.

Alternative Flows:	User deselects all dietary requirements 1. Application acknowledges change with same steps as above
Exceptions:	The new inputs are not valid, the app will not save and display an error message.
Includes:	-
Special Requirements:	-
Assumptions:	Dietary requirement options are pre-defined and updatable by application admin
Notes and Issues:	-

Use Case ID:	SP01		
Use Case Name:	Specify Dietary Preferences		
Created By:	Zhan Yi Yun	Last Updated By:	Zhan Yi Yun
Date Created:	28/08/25	Date Last Updated:	10/9/2025

Actor:	User
Description:	User select one or more dietary preference(s) to personalise their application experience. (application will prioritise food recommendations based on user's selected dietary preference(s))
Preconditions:	1. User logged in 2. User has access to their profile settings
Postconditions:	1. Dietary preferences saved and correctly displayed on user's profile 2. Food recommendations by application prioritized based on user's dietary preferences
Priority:	High
Frequency of Use:	Medium
Flow of Events:	1. User navigates to their profile



	2. User clicks "Edit Profile" 3. User selects the field "Dietary Preferences" 4. Application displays a list of diet types (eg plant-based, low-carb, keto, mediterranean) and a list of cuisine (e.g chinese, malay, indian, thai, japanese, korean, italian, mexican etc) 5. User selects one or more options. 6. User clicks "Save changes" 7. application validates and saves changes 8. application displays successful update.
Alternative Flows:	User deselects all dietary requirements 1. application acknowledges change with same steps as above
Exceptions:	The new inputs are not valid, the app will not save and display an error message.
Includes:	-
Special Requirements:	-
Assumptions:	Dietary preference options are pre-defined and updatable by application admin
Notes and Issues:	-

Use Case ID:	SE01		
Use Case Name:	Search Eateries on Map		
Created By:	Zhan Yi Yun	Last Updated By:	Zhan Yi Yun
Date Created:	28/08/25	Date Last Updated:	10/9/2025

Actor:	User, Google Map API
Description:	User searches for eateries using a map interface.
Preconditions:	1. Location services enabled 2. User's device has internet connection
Postconditions:	1. Map shows position of user 2. Map shows position of eateries nearby (within 1km)

	3. User able to interact with individual eateries including: <ul style="list-style-type: none"> <li>- Viewing details about the eatery</li> <li>- Getting directions to the eatery</li> </ul>
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> <li>1. User navigates to map interface</li> <li>2. Eateries near user's location are fetched and displayed as pins on user's map</li> <li>3. User may use filter bars to filter food options</li> <li>4. Filtered results shown in real time on the map</li> <li>5. User clicks on a pin to view details about the eatery</li> </ol>
Alternative Flows:	<ol style="list-style-type: none"> <li>1. User manually inputs location into search bar instead of using GPS location to search for eateries <ol style="list-style-type: none"> <li>a. Steps in <a href="#">IL01</a></li> </ol> </li> <li>2. If user denies browser or app permission to access location services when using "current location" feature <ol style="list-style-type: none"> <li>a. System prompts user to manually enter a location or</li> <li>b. Request location access again</li> </ol> </li> </ol>
Exceptions:	<ol style="list-style-type: none"> <li>1. If the Google API cannot be accessed due to connection timeout or any other reason, <ol style="list-style-type: none"> <li>a. Display an error message asking the user to try again in 5 minutes</li> <li>b. Log the error and notify the System Admin about the issue</li> </ol> </li> </ol>
Includes:	<a href="#">IL01</a> , <a href="#">IR01</a> , <a href="#">DE01</a>
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	IL01		
Use Case Name:	Input Location		
Created By:	Zhan Yi Yun	Last Updated By:	Zhan Yi Yun

Date Created:	28/08/25	Date Last Updated:	10/9/2025
---------------	----------	--------------------	-----------

Actor:	User, Google Maps API
Description:	User manually inputs location to view eateries around the area
Preconditions:	<ol style="list-style-type: none"> <li>1. Location services enabled</li> <li>2. application has access to geocoding services</li> <li>3. User's device has internet connection</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. application fetches location</li> </ol>
Priority:	High
Frequency of Use:	Medium
Flow of Events:	<ol style="list-style-type: none"> <li>1. User navigates to map interface</li> <li>2. User enters a location manually into search bar</li> <li>3. User may select a search radius (eg within 1km)</li> <li>4. application geocodes location</li> <li>5. application displays map with eateries within the default or specified search radius of the specified location</li> </ol>
Alternative Flows:	<ol style="list-style-type: none"> <li>1. User uses GPS location instead of manually inputting location to search for eateries <ol style="list-style-type: none"> <li>a. Steps in SE01</li> </ol> </li> <li>2. If user denies browser or app permission to access location services when using "current location" feature <ol style="list-style-type: none"> <li>a. System prompts user to manually enter a location or</li> <li>b. Request location access again</li> </ol> </li> </ol>
Exceptions:	<ol style="list-style-type: none"> <li>1. System does not find any eatery within the location specified by user in the database <ol style="list-style-type: none"> <li>a. System displays an error message and prompt user to check input location</li> <li>b. System redirects back to map interface</li> </ol> </li> <li>2. Google Maps API faces outage and unable to fetch eatery information <ol style="list-style-type: none"> <li>a. Application displays error message suggesting system is encountering disruptions and asks user to retry in 5 minutes.</li> <li>b. Application logs the error and informs System Admin about the issue</li> </ol> </li> <li>3. Google Maps API under system maintenance and unable to fetch eatery information <ol style="list-style-type: none"> <li>a. Application displays error message suggesting that eatery information is not available currently</li> </ol> </li> </ol>

	b. Application informs user of expected system downtime
Includes:	-
Special Requirements:	-
Assumptions:	Location input is optional, current location is used as default for recommending eateries to users
Notes and Issues:	-

Use Case ID:	IR01		
Use Case Name:	Input Search Radius		
Created By:	Zhan Yi Yun	Last Updated By:	Zhan Yi Yun
Date Created:	28/08/25	Date Last Updated:	10/9/2025

Actor:	User
Description:	User selects a search radius when manually searching for eateries.
Preconditions:	1. Location has been set
Postconditions:	1. application updates search results to only include locations within the specified radius
Priority:	Low
Frequency of Use:	Medium
Flow of Events:	1. User navigates to map interface 2. User enters a location manually into search bar 3. User selects a search radius (eg within 1km) 4. application geocodes location 5. application displays map with eateries within the search radius of the specified location
Alternative Flows:	1. User does not select search radius a. application uses a default search radius of within 2km

Exceptions:	<ol style="list-style-type: none"> <li>1. If the Google API drops due to connection timeout or any other reason, <ol style="list-style-type: none"> <li>a. Display an error message asking the user to try again in 5 minutes</li> <li>b. Log the error and notify the System Admin about the issue</li> </ol> </li> </ol>
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	DE01		
Use Case Name:	Display Eateries		
Created By:	Zhan Yi Yun	Last Updated By:	Zhan Yi Yun
Date Created:	28/08/25	Date Last Updated:	10/9/2025

Actor:	User, Google Map API
Description:	Application displays eateries on a map based on the user's specified location and search radius
Preconditions:	1. User performs valid search
Postconditions:	1. Application displays a map with eateries that correctly matches user's search criterias
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> <li>1. User performs search (IL01 &amp; IR01)</li> <li>2. Application queries backend for matching eateries</li> <li>3. Application receives list of matching eateries</li> <li>4. UI displays eateries as pins on map</li> </ol>

	5. User may select a pin to view details of the eatery
Alternative Flows:	<ol style="list-style-type: none"> <li>1. No matching eateries found <ol style="list-style-type: none"> <li>a. application informs user (eg display “No matching eateries found, Please try a different search”)</li> </ol> </li> </ol>
Exceptions:	<ol style="list-style-type: none"> <li>2. If the Google API cannot be accessed due to connection timeout or any other reason, <ol style="list-style-type: none"> <li>a. Display an error message asking the user to try again in 5 minutes</li> <li>b. Log the error and notify the System Admin about the issue</li> </ol> </li> </ol>
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	-

Use Case ID:	RE01		
Use Case Name:	Review Eateries		
Created By:	Zhan Yi Yun	Last Updated By/Date:	Taing Meyling
Date Created:	28/08/25		10/09/2025

Actor:	User
Description:	<p>User provide a review for an eatery through any of the following methods:</p> <ul style="list-style-type: none"> <li>- star rating (out of 5)</li> <li>- textual feedback</li> <li>- photos</li> </ul> <p>which helps other users make more informed decisions</p>
Preconditions:	<ol style="list-style-type: none"> <li>1. User logged into their verified account</li> <li>2. Eatery exists in database</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. Review saved and associated with eatery</li> </ol>

	2. Average rating updated accordingly (for star ratings) 3. Review visible to other users
Priority:	Medium
Frequency of Use:	Low
Flow of Events:	1. User opens eatery details 2. User clicks "Write a review" 3. Application displays review interface 4. User fills in star ratings and/or photos and/or textual feedback 5. User clicks "Submit" 6. Application saves review and updates eatery's review details
Alternative Flows:	1. User (edits or) deletes a review a. Review is updated or removed from User's review history accordingly 2. User attempts to submit multiple reviews for the same eatery in a short period of time a. Allow only one review per eatery, flag to admin for moderation 3. If photos fail to upload due to size limits and unsupported formats a. Display error and allow reupload
Exceptions:	1. Server error or timeout a. Display an error message
Includes:	-
Special Requirements:	1. Need admin/moderation tools to moderate reviews
Assumptions:	-
Notes and Issues:	1. Can consider upvote feature to prioritise helpful reviews 2. Can consider preventing fake/spam reviews

Use Case ID:	DR01		
Use Case Name:	Display Reviews		
Created By:	Zhan Yi Yun	Last Updated By:	Zhan Yi Yun
Date Created:	28/08/25	Date Last Updated:	10/9/2025

Actor:	User
Description:	Application displays user-generated reviews for a selected eatery, including ratings, textual feedback, photos, tags
Preconditions:	<ol style="list-style-type: none"> <li>1. Eatery has at least one review</li> <li>2. User visits the specific eatery and scroll to reviews section</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. All reviews displayed</li> <li>(2. Reviews can be sorted (eg by date, ratings etc))</li> </ol>
Priority:	High
Frequency of Use:	Medium
Flow of Events:	<ol style="list-style-type: none"> <li>1. User opens eatery details and scrolls to “Reviews” sections</li> <li>2. UI displays reviews</li> </ol>
Alternative Flows:	Eatery has no reviews <ol style="list-style-type: none"> <li>1. application informs user (eg display “No reviews yet”)</li> </ol>
Exceptions:	<ol style="list-style-type: none"> <li>1. Review contains inappropriate content that has yet been moderated by System Admin               <ol style="list-style-type: none"> <li>a. Application displays error message suggesting review has been hidden</li> </ol> </li> <li>2. Backend faces outage, unable to display reviews               <ol style="list-style-type: none"> <li>a. Display error message to user</li> <li>b. System flags incident to admin</li> </ol> </li> </ol>
Includes:	-
Special Requirements:	-
Assumptions:	-
Notes and Issues:	Can consider preventing fake/spam reviews by allowing users to report displayed reviews which will be flagged for moderation