

# SC2006 Lab 1 Deliverables

Lab index	SDAA	
Group number	31	
Group members	Liu Xiaotao	U2320836F
	Low Jo Yi, Nicole	U2321370D
	Mahi Pandey	U2321382F
	Mehta Rishika	U2323133H
	Zhang Yichi	U2320736J
	Zhao Qixian	U2321752L

<b>SC2006 Lab 1 Deliverables.....</b>	<b>1</b>
<b>Data Dictionary.....</b>	<b>5</b>
<b>Functional Requirements:.....</b>	<b>7</b>
<b>1. Registration.....</b>	<b>7</b>
1.1. Users must be able to register with an email address.....	7
1.2. Users must be able to register via different methods.....	8
<b>2. Login.....</b>	<b>10</b>
2.1. Users must be able to log in with an email address.....	10
2.2. Users must be able to log in via their Google Account.....	10
<b>3. Profile Management.....</b>	<b>11</b>
3.1. Users must be able to upload and update their personal information.....	11
3.2. Users must be able to view their personal information.....	11
3.3. Users' profile data must be updated in real-time and reflected across all relevant features.....	12
3.4. Users must be able to delete their profile.....	12
3.5. Users must be able to select their language preference.....	12
<b>4. Nutrition Calculation.....</b>	<b>13</b>
4.1. Users must be able to input the meals they consume.....	13
4.2. The system must calculate the calories and nutrient content based on user input.....	13
4.3. Users must be able to track their nutrition intake.....	13
4.4. The system must calculate the user's nutrition needs using a specified algorithm (such as the Mifflin-St Jeor equation or Harris-Benedict equation). ..	14
4.5. Users must be able to view their nutrition summary on the dashboard...	14
<b>5. Food Availability.....</b>	<b>14</b>
5.1 Users must be able to see food availability.....	14
5.2 The system must provide daily price information for the suggested food items.....	15
<b>6. Nutrition Reference Sheet.....</b>	<b>15</b>
6.1. The system must include a dynamically generated nutrition reference sheet.....	15
6.2. The system must present the nutritional data in a user-friendly format... ..	16
6.3. The system must allow integration of the nutrition reference sheet into other system features.....	16
6.4. The system must support both user searches and category-based	

browsing.....	16
<b>7. Meal Planner.....</b>	<b>17</b>
7.1. The system must generate personalised meal plans using Welloh.....	17
7.2. The system must provide three meal plan options for the user to choose from.....	17
7.3. The system must allow users to record their selected meal plan.....	17
7.4. The system must regularly update suggested meal plans based on user progress and feedback.....	18
7.5. The system must provide detailed nutritional information for each meal in the plan.....	18
<b>8. The Welloh Chatbot.....</b>	<b>18</b>
8.1. The Welloh chatbot must provide real-time responses to user queries related to diet and nutrition.....	18
8.2. The chatbot must be accessible through the EatWellthy application interface.....	19
8.3. The chatbot must support natural language, text-based interactions.....	19
8.4. The chatbot must handle multiple queries within a single session.....	19
8.5. The chatbot must be able to assist users with meal planning and tracking.	19
8.6. The chatbot must handle user feedback and continuously improve.....	20
<b>9. Location of the user.....</b>	<b>20</b>
9.1. The system must identify all branches of the four major supermarkets...	20
9.2. The system must identify the user's current location.....	20
9.3. The system must suggest the nearest supermarket branches where the user can buy the suggested food items.....	20
9.4. The system must allow users to save preferred supermarket locations..	21
<b>10. Logout.....</b>	<b>21</b>
10.1. The system must allow users to securely log out of their account.....	21
10.2. The system must securely terminate the user's session upon logout....	21
10.3. The system must redirect the user after a successful logout.....	22
<b>Non Functional Requirements:.....</b>	<b>23</b>
<b>1. Usability: Friendliness.....</b>	<b>24</b>
1.1 User Interface (UI) Design and Accessibility.....	24
1.2 Navigation and Task Completion.....	24
1.3 Help and Support.....	24
<b>2. Performance: Qualifiable Attributes.....</b>	<b>24</b>
2.1 All the calculations must be based on a reliable algorithm.....	24

2.2 System Operations and Responsiveness.....	25
2.3 Login and Authentication.....	25
2.4 User Interaction.....	25
<b>3. Scalability: Up scale Easily.....</b>	<b>25</b>
3.1 User Base Growth.....	25
3.2 Database and Storage Scalability.....	25
3.3 Feature Expansion.....	26
<b>4. Reliability: robustness, safety.....</b>	<b>26</b>
4.1 System Monitoring.....	26
4.2 Data Leakage Prevention Capability.....	26
<b>5. Supportability: maintainability.....</b>	<b>27</b>
5.1 Code Maintainability and Transparency.....	27
5.2 System Monitoring and Alerts.....	27
5.3 Documentation and Knowledge Management.....	27
5.4 Maintenance and Updates.....	27
<b>6. Security: authentication.....</b>	<b>28</b>
6.1 User Authentication.....	28
6.2 Data Encryption.....	28
6.3 Access Control.....	28
6.4 Data Privacy and Compliance.....	28
6.5 Vulnerability Management.....	29
<b>Use Case Diagram.....</b>	<b>30</b>
I.For Functional requirement #1 (Registration).....	31
U0101 New User Registration using email.....	31
U0102 New User Registration using Google Account.....	33
U0103 User Login via Email.....	35
U0104 User Login via Google Account.....	36
U0105 Reset Forgotten Password.....	37
U0106 Change Account Password.....	38
II. For Functional Requirement #3 (Profile Management).....	39
U0201 View User Profile.....	39
U0202 Update User Profile Information.....	40
U0203 Manage Dietary Preferences.....	41
U0204 Update Security Settings.....	42
U0205 Delete User Account.....	43
III. For Functional Requirement #4 (Nutrition Calculation).....	45

U0301 Log Daily Meal.....	45
U0302 Edit Logged Meal.....	47
U0303 Delete Logged Meal.....	49
U0304 View Meal History.....	51
U0305 Generate Personalized Diet Plan.....	53
U0306 Sync Diet Plan with Google Calendar.....	55
IV. For Functional Requirement #8 (Welloh Chatbot).....	57
U0401 Ask Nutritional Questions.....	57
U0402 Receive Nutritional Alerts.....	59
U0403 Query Nutritional Information via Chatbot.....	62
V. For Functional Requirement # (Nutrition Reference Sheet).....	64
U0501 Search for Food Items.....	64
U0502 Update Food Database.....	66
U0503 Request store availability data.....	68
U0504 Manage User data.....	70
U0505 Backup and Restore.....	72
VI. For Functional Requirement #7 (Meal Planner).....	74
U0601 View Dietary History.....	74
U0602 Analyze Nutritional Intake Trends.....	75
U0603 Generate User Reports.....	76
U0604 Export Nutritional Data.....	77
<b>UI Design.....</b>	<b>78</b>

# Data Dictionary

Terms	Definition
EatWellthy	The name of the web application that helps users record and track their diet, manage nutrition, and receive personalised consumption and purchase suggestions based on their input and dietary preferences.
User	The authorised end user who has a registered account and is using services provided by the EatWellthy web application.
UserID	Unique integer identifier for each user within the system. (Primary key in database)
Username	The name chosen by the user for logging into EatWellthy and display.
Email	The email address associated with the user's account, used for account recovery and notifications.
Password	An encrypted string of characters used by the user to access their account.
Profile	Collection of user-specific data including age, weight, height, budget, allergies and dietary preferences, linked to UserID.
User Registration	Creation of a UserID and Profile containing email, password, username. The information of the user will be stored in the database.
User Login	Web accessibility to users who have already registered can login through username or email and password, or through Google Account linking.
Dashboard	A centralised interface where users can view summaries of their nutritional intake, progress tracking, diet suggestions, and other personalised insights.
Custom Food Entry	A feature that allows users to manually input food items, including their nutritional information, into the user's database only.

Dietary Preferences	User-selected options like halal, vegetarian, and allergies, which the system filters when providing diet suggestions and tracking nutrition.
Nutritional Values	Includes information such as calories, proteins, fats, carbohydrates, vitamins, and minerals, relevant to the food items entered by the user or predefined in the Nutrition Database.
Nutrition Database	The centralised database that stores Nutritional Values for all food items, both predefined and user-added entries.
Diet Planner	A tool within EatWellthy that allows users to schedule and organise their meals based on diet and recipe suggestions.
Progress Tracker	A tool that visualises the user's progress over time, including weight changes, adherence to nutritional goals, and other health metrics.
Diet Suggestions	Personalised recommendations provided by the application based on the user's dietary preferences, nutritional goals, and health conditions.
Google Calendar Integration	A feature allowing users to sync their diet suggestions and meal plans with Google Calendar, including setting reminders for meals and nutritional goals.
Food Availability	Live data on the availability of suggested food items in local stores, updated daily, allowing users to make informed shopping decisions.
Feedback	Comments provided by Users regarding EatWellthy's performance, suggestions for new features, or reporting of issues.
Welloh	An AI-driven chat interface powered by the ChatGPT API, trained in healthy diet-related data, that allows users to ask dietary questions and receive personalised responses.
My Location	A feature that uses Google Maps services to determine the user's location and suggest nearby shops that offer the required food items.

# Functional Requirements:

## 1. Registration.

1.1. Users must be able to register with an email address.

1.1.1 System must provide three input fields to input information.

1.1.1.1 One of the input fields must be a username.

1.1.1.2 One of the input fields must be an email address.

1.1.1.3 One of the input fields must be a password.

1.1.2. System must ensure that the user fills in all the input fields before allowing registration.

1.1.3. System must verify all input information.

1.1.3.1. System must verify that the username is unique.

1.1.3.2. System must verify that the email meets the requirements.

1.1.3.2.1. System must verify that the email is valid with a character “@”.

1.1.3.2.2. System must verify that the email is unique.

1.1.3.2.3. System must direct the user to the LogIn Page when the input email is linked to a registered account.

1.1.3.3. System must verify that the password meets the requirements.

1.1.3.3.1. The password must be a minimum of 8 characters

1.1.3.3.2. The password shall have at least one uppercase letter

1.1.3.3.3. The password shall have at least one lowercase letter

1.1.3.3.4. The password shall have at least one number

1.1.3.3.5. The password shall have at least one special character (e.g., !, @, #, \$)

1.1.4. System must ask the user to input information again when their previous input failed the verification.

1.1.5 System must send a six digits OTP to verify the user's email address.

1.1.5.1. System must provide one input field for the user to input the OTP received in their email.

1.1.5.2. System must provide a request button for the user to request the OTP again.

1.1.5.3. System must ensure that OTP is valid for a limited period.

1.1.6. System must register an account for the user when the user email is verified.

1.1.6.1 System must assign a unique UserID to the user's account.

1.1.6.2. System must record the information to the user's account.

1.1.7 System must redirect user to Login Page when user's account is created.

## 1.2. Users must be able to register via different methods.

1.2.1. Users must be able to register via their Google Account.

1.2.1.1. System must provide one input field to input a Google email address.

1.2.1.1.1. System must verify that the Google email is valid that ends with "@gmail.com".

1.2.1.1.2. System must verify that the Google email is unique.

1.2.1.1.3. System must direct the user to the LogIn Page when the input Google email is linked to a registered account.

1.2.1.2 System must provide one input field to input a username when Google Account is linked.

1.2.1.2.1. System must verify that the username is unique.

1.2.1.3 System must register an account when Google Account is linked and Username is verified.

1.2.1.3.1 System must assign a unique UserID to the user's account.

1.2.1.3.2 System must record the information to the user's account.

1.2.1.4. System must redirect the user to the Home Page when the user's account is created.

## **2. Login**

**2.1. Users must be able to log in with an email address.**

2.1.1. System must provide two input fields to input login credentials.

2.1.1.1. One of the input fields must be for the email address.

2.1.1.2. One of the input fields must be for the password.

2.1.2. System must ensure that the user fills in both input fields before allowing login.

2.1.3. System must verify the email and password provided by the user.

2.1.3.1. System must verify that the email exists in the system.

2.1.3.2. System must verify that the password matches the one associated with the provided email.

2.1.3.3. System must allow the user three attempts to input the correct password.

2.1.3.3.1. System must lock the account for a specified period after three failed login attempts.

2.1.3.3.2. System must notify the user to reset password via email when the account is locked.

2.1.4. System must redirect the user to the Home Page upon successful login.

2.1.5. System must provide a “Forgot Password” option.

2.1.5.1. System must provide one input field for the user to input their registered email address.

2.1.5.2. System must send a password reset link to the provided email.

2.1.5.3. System must verify the password reset request by asking the user to input a new password that meets the password requirements (as defined in Section 1.1.3.3).

**2.2. Users must be able to log in via their Google Account.**

2.2.1. System must provide one input field to input a Google email address.

2.2.1.1. System must verify that the Google email is valid and ends with “@gmail.com.”

2.2.1.2. System must verify that the Google email is linked to an existing account in the system.

2.2.2. System must redirect the user to the Home Page upon successful login via Google.

### 3. Profile Management.

3.1. Users must be able to upload and update their personal information.

3.1.1. Users must be able to access this functionality through the dashboard interface.

3.1.2. Users must be able to input and update their profile details.

3.1.2.1. Users must be able to input their age.

3.1.2.2. Users must be able to input their height.

3.1.2.3. Users must be able to input their weight.

3.1.2.4. Users must be able to input their expected daily meal budget on average.

3.1.2.5. Users must be able to upload or update their profile picture.

3.1.2.5.1. System must support image file formats such as JPEG and PNG.

3.1.2.5.2. Users must be able to preview their profile picture before saving.

3.1.3. Users must be able to manage their communication and notification preferences (e.g., email notifications, reminders) within their profile.

3.2. Users must be able to view their personal information.

3.2.1. Users must be able to access this functionality through the dashboard interface.

3.2.2. Users must be able to view their profile details.

3.2.2.1. Users must be able to view their age.

- 3.2.2.2. Users must be able to view their height.
- 3.2.2.3. Users must be able to view their weight.
- 3.2.2.4. Users must be able to view their budget.

### 3.3. Users' profile data must be updated in real-time and reflected across all relevant features.

- 3.3.1. Users must see any changes made to their profile (e.g., weight, budget) immediately reflected in personalised recommendations, meal plans, or other related features.
- 3.3.2. Users must have consistent and up-to-date profile data across all instances of the platform.

3.3.3. Users must be reminded within the app weekly to update their profile data (weight and height etc.)

### 3.4. Users must be able to delete their profile.

- 3.4.1. Users must be able to delete their account through the profile management interface.
- 3.4.2. System must provide two confirmation steps to prevent accidental deletion.
- 3.4.3. System must inform users of the data retention policy upon initiating account deletion.

### 3.5. Users must be able to select their language preference.

- 3.5.1. Users must be able to choose their preferred language for the application interface.
- 3.5.2. System must support at least four languages (English by default, Simplified Chinese, Malay, Tamil) as per the application requirements.

## 4. Nutrition Calculation

### 4.1. Users must be able to input the meals they consume.

- 4.1.1. Users must be able to access this functionality through the “Nutrition Calculator” in the navbar.
- 4.1.2. Users must be able to input meal details manually.
  - 4.1.2.1. Users must be able to input the name of the food item.
  - 4.1.2.2. Users must be able to input the portion size or quantity.
  - 4.1.2.3. Users must be able to input any additional details (e.g., preparation method).

### 4.2. The system must calculate the calories and nutrient content based on user input.

- 4.2.1. The system must retrieve nutritional information from the Nutritionix API.
- 4.2.2. The system must calculate the total calories and nutrients (e.g., protein, carbohydrates, fats, vitamins) for the meal.
- 4.2.3. The system must display the calculated calories and nutrient content to the user in real-time on the dashboard.

### 4.3. Users must be able to track their nutrition intake.

- 4.3.1. The system must update the user's daily nutrition intake based on the meals inputted.
- 4.3.2. Users must be able to view a summary of their daily, weekly, and monthly nutrition intake on the dashboard.
- 4.3.3. Users must be able to view detailed information about each nutrient (e.g., total grams of protein consumed).

4.4. The system must calculate the user's nutrition needs using a specified algorithm (such as the Mifflin-St Jeor equation or Harris-Benedict equation).

4.4.1. The system must calculate the user's daily nutrition needs based on profile data (e.g., age, weight, height, activity level).

4.4.2. The system must compare the user's nutrition intake with their calculated needs.

4.4.3. The system must provide feedback to the user if their intake is above or below their nutritional needs.

4.5. Users must be able to view their nutrition summary on the dashboard.

4.5.1. The system must display the user's nutrition intake vs. their calculated needs on the dashboard.

4.5.2. The system must highlight any discrepancies between intake and needs in a graphic method (e.g., colour-coded indicators).

## 5. Food Availability.

5.1 Users must be able to see food availability.

5.1.1. Users must be able to access this functionality through the dashboard interface.

5.1.2. The system must suggest food items based on the user's nutritional needs, allergies and preferences.

5.1.3. The system must gather availability and pricing information from at least three supermarkets: FairPrice, Cold Storage and Sheng Siong.

5.1.4. The system must display the availability status of each food item for all four stores.

5.1.5. The system must update availability information daily to ensure accuracy.

**5.2 The system must provide daily price information for the suggested food items.**

5.2.1. The system must collect pricing data from at least four FairPrice, Prime, Cold Storage, and Sheng Siong using a web crawler.

5.2.2. The system must ensure that the web crawler collects data daily at a specified time (e.g., 12 pm).

5.2.3. The system must display the prices for each food item alongside its availability status in the four stores.

5.2.4. The system must allow users to sort or filter suggested food items based on price or store availability.

## **6. Nutrition Reference Sheet.**

**6.1. The system must include a dynamically generated nutrition reference sheet.**

6.1.1. The system must retrieve accurate nutritional values for food items from the Nutritionix API.

6.1.2. The system must ensure that the nutritional data is up-to-date by fetching information from the Nutritionix API at least once a week.

6.1.3. If specific nutritional information is not available from the Nutritionix API, the system must use the Welloh to provide approximate nutritional data.

6.1.4. Users must be notified when information is sourced from external APIs (e.g., "Data provided by Nutritionix" or "Generated by Welloh").

**6.2. The system must present the nutritional data in a user-friendly format.**

6.2.1. The system must display nutritional values in a table format, allowing users to easily compare different food items.

6.2.2. The system must offer visual aids, such as bar charts, icons, or colour coding, to help users quickly understand the nutritional content (e.g., low-calorie items, high-protein options, high-sodium warnings).

6.2.3. The system must allow users to sort or filter the data by various criteria, such as calorie count, protein content, or dietary restrictions (e.g., vegetarian, halal).

**6.3. The system must allow integration of the nutrition reference sheet into other system features.**

6.3.1. The system must integrate the nutrition reference sheet with meal planning and food tracking features, allowing users to reference nutritional information directly when planning or logging meals.

**6.4. The system must support both user searches and category-based browsing.**

6.4.1. Users must be able to search for specific food items using a search bar within the nutrition reference sheet.

6.4.2. Users must also be able to browse predefined categories (e.g., fruits, vegetables, grains) to explore nutritional information for related food items.

## 7. Meal Planner.

7.1. The system must generate personalised meal plans using Welloh.

7.1.1. The system must generate meal plans based on user preferences, credentials, and dietary needs.

7.1.1.1. The system must consider user-provided information such as age, gender, weight, height, activity level, and any specific dietary preferences (e.g., vegetarian, halal).

7.1.1.2. The system must account for user-specific nutritional goals, health conditions, and any allergies or dietary restrictions.

7.1.1.3. The system must consider user budget constraints when suggesting meal plans, optimising for cost-effectiveness.

7.2. The system must provide three meal plan options for the user to choose from.

7.2.1. Each meal plan must meet the user's daily caloric requirements and nutritional needs.

7.2.2. The system must ensure that each meal plan offers a balanced variety of food and drink items.

7.2.3. The system must present the three meal plan options in a clear and organised format, allowing users to easily compare and select their preferred plan.

7.3. The system must allow users to record their selected meal plan.

7.3.1. Users must be able to save their selected meal plan to their dashboard.

7.3.2. The system must offer an option for users to sync their selected meal plan with their Google Calendar.

7.3.3. The system must automatically update the user's nutritional tracking and progress based on the selected meal plan.

**7.4. The system must regularly update suggested meal plans based on user progress and feedback.**

7.4.1. The system must prompt users for feedback on the meal plans and adjust future suggestions accordingly.

7.4.2. The system must update meal plans dynamically if the user's profile data changes (e.g., weight loss or changes in dietary preferences).

7.4.3. The system must allow users to request new meal plan suggestions at any time, taking into account their most recent data and preferences.

**7.5. The system must provide detailed nutritional information for each meal in the plan.**

7.5.1. The system must display the nutritional content of each meal, including calories, macronutrients (protein, carbohydrates, fats), and any other relevant nutrients (e.g., vitamins, minerals).

7.5.2. The system must present this nutritional information in both a tabular format and with visual aids, such as charts or graphs, to help users understand the nutritional value of their selected meal plan.

## **8. The Welloh Chatbot**

**8.1. The Welloh chatbot must provide real-time responses to user queries related to diet and nutrition.**

8.1.1. The chatbot must use the ChatGPT API to generate responses that are accurate, informative, and relevant to the user's needs.

8.1.2. The chatbot must provide personalised responses by considering the user's profile information, such as age, weight, height, activity level, dietary preferences, and health conditions.

**8.2. The chatbot must be accessible through the EatWellthy application interface.**

8.2.1. Users must be able to interact with the chatbot from any section of the EatWellthy application.

8.2.2. The chatbot must be available 24/7, providing continuous support without interruptions.

**8.3. The chatbot must support natural language, text-based interactions.**

8.3.1. Users must be able to type questions in a natural, conversational manner, and the chatbot must respond in a similarly natural language style.

8.3.2. The chatbot must be able to understand and respond to a wide range of dietary and nutrition-related topics, from general inquiries to specific health-related questions.

**8.4. The chatbot must handle multiple queries within a single session.**

8.4.1. Users must be able to ask multiple questions in one session without needing to restart or refresh the interaction.

8.4.2. The chatbot must maintain context within a session, allowing for follow-up questions and clarifications based on previous responses.

**8.5. The chatbot must be able to assist users with meal planning and tracking.**

8.5.1. The chatbot must suggest meals or snacks based on the user's dietary preferences and nutritional needs.

8.5.2. Users must be able to record chatbot-suggested meals directly into their meal plan or nutritional tracking within the application.

**8.6. The chatbot must handle user feedback and continuously improve.**

8.6.1. The system must allow users to rate the responses provided by the chatbot.

8.6.2. The system must analyse user feedback to improve the chatbot's response accuracy and relevance over time.

## **9. Location of the user**

**9.1. The system must identify all branches of the four major supermarkets.**

9.1.1. The system must maintain an up-to-date database of all supermarket branch locations, including their addresses and operational hours.

9.1.2. The system must update this database at least once a week to ensure accuracy, reflecting any new branches, closures, or changes in operational hours.

**9.2. The system must identify the user's current location.**

9.2.1. The system must use GPS to accurately determine the user's current location.

9.2.2. The system must prompt the user for permission to access their location data, in compliance with privacy regulations.

**9.3. The system must suggest the nearest supermarket branches where the user can buy the suggested food items.**

9.3.1. The system must calculate the distance between the user's location and the nearest supermarket branches of FairPrice, Prime, Cold Storage, and Giant.

9.4. The system must allow users to save preferred supermarket locations.

9.4.1. Users must be able to mark certain supermarket branches as "preferred" for future reference.

9.4.2. The system must prioritise suggesting these preferred locations in future searches, if they are within a feasible distance.

## 10. Logout

10.1. The system must allow users to securely log out of their account.

10.1.1. The system must provide a clearly visible "Logout" button, accessible from all pages of the application.

10.1.2. The system must confirm the user's intention to log out by displaying a confirmation prompt when the "Logout" button is clicked.

10.1.2.1. The confirmation prompt must include a message asking if the user is sure they want to log out, along with "Yes" and "No" options.

10.2. The system must securely terminate the user's session upon logout.

10.2.1. The system must immediately terminate the user's session upon confirmation, ensuring that all session tokens are invalidated.

10.2.2. The system must ensure that all session data is cleared from the client side (e.g., cookies, local storage) to prevent unauthorised access after logout.

10.3. The system must redirect the user after a successful logout.

10.3.1. The system must redirect the user to the login page or the homepage after a successful logout.

10.3.2. The system must display a message confirming that the user has been logged out.

# Non Functional Requirements:

## Non-functional Requirements

- Usability:** friendliness  
Ease with which a user can learn to operate, prepare inputs for, and interpret outputs of a system or component  
Example: conventions for interface
- Reliability:** robustness, safety  
Ability of a system or component to perform its required functions under stated conditions for a specified period of time  
Example: mean time to failure, ability to withstand attack, ability to detect faults
- Performance:** qualifiable attributes  
Example: response time, throughput, availability, accuracy
- Scalability:** up capacity easily  
Ability to accommodate larger volumes (whether of users, throughput, data) over time  
Example: highest volume under which system still performs normally
- Supportability:** maintainability  
Ease of changes to the system after deployment  
Example: ability to change to deal with additional domain concepts, new technology or fix defects
- Security:** authentication  
How well are the system and its data protected against attacks?  
Example: protection of sensitive info, adherence to standards, adoption of best practices

## How to Write Functional Requirements

Template	Example
The <system name> shall <system response>	The software package shall include an installer.
WHILE <system state>, the <system name> shall <system response>	While in Low Power Mode, the software shall keep the display brightness at the Minimum Level.
WHEN <trigger> <optional precondition>, the <system name> shall <system response>	When a DVD is inserted into the DVD player the OS shall spin up the optical drive.
IF <unwanted condition or event>, THEN the <system name> shall <system response>	If the memory checksum is invalid, then the software shall display an error message.
WHERE <feature is included>, the <system name> shall <system response>	Where a HDMI port is present, the software shall allow the user to select HD content for viewing.

**Ubiquitous Requirements**  
Define a fundamental property of the system, no preconditions or trigger

**State-driven Requirements**  
Triggered while the system is in a specific state

**Event-driven Requirements**  
Initiated when and only when a trigger occurs or is detected

**Unwanted Behaviour Requirements**  
Handle unwanted behaviors including error conditions, failures, faults, disturbances and other undesired events

**Optional Feature Requirements**  
Invoked only in systems that include the particular optional feature

Source: [https://www.aria.org/conferences2013/files/ICCGI13/ICCGI\\_2013\\_Tutorial\\_Terzakis.pdf](https://www.aria.org/conferences2013/files/ICCGI13/ICCGI_2013_Tutorial_Terzakis.pdf)

# **1. Usability: Friendliness**

## **1.1 User Interface (UI) Design and Accessibility**

1.1.1 The design must provide full information for users with mobile and desktop devices

1.1.2 The system must achieve a System Usability Scale (SUS) score of 80 or higher in user testing.

1.1.3 The colour scheme must pass a minimum contrast ratio of 4.5:1 for text and background colours to support readability.

## **1.2 Navigation and Task Completion**

1.2.1 The user interface must allow a new user to complete core tasks (e.g., logging food, viewing diet suggestions) within 3 minutes without external assistance.

1.2.2 The dashboard interface must allow users to customize the display (e.g., choosing which statistics to view) within 3 clicks or taps.

1.2.3 All forms must include inline validation, providing real-time feedback within 0.5 seconds after the user completes a field.

## **1.3 Help and Support**

1.3.1 A help centre or FAQ section must be accessible within 3 clicks from any page, providing detailed explanations and video tutorials for common tasks.

# **2. Performance: Qualifiable Attributes**

## **2.1 All the calculations must be based on a reliable algorithm**

2.1.1 The algorithm must be able to process the calculations within 5 seconds

2.1.1.1 The system should calculate daily caloric needs based on user input within 3 seconds.

2.1.1.2 The system should calculate the nutrition in the diet through the Nutrition Database

## **2.2 System Operations and Responsiveness**

2.2.1 All the user actions (eg. logging in, generating diet plans etc.) must be processed and returned within 5 seconds of making the request

2.2.2 Standard database queries like fetching the User profile must return results within 2 seconds

2.2.3 All user interface components (e.g., buttons, sliders, forms) should respond to user interactions within 1 second.

## **2.3 Login and Authentication**

2.3.1 The system should authenticate users within 3 seconds of submission of credentials

2.3.2 OAuth-based logins (e.g., Google ) should be completed within 2.5 seconds.

## **2.4 User Interaction**

2.4.1 The page load time to fully load the dashboard after login should not exceed 3 seconds for 95% of users.

2.4.2 Transitions between UI states (e.g., loading new data on the dashboard) should not exceed 2 seconds

# **3. Scalability: Up scale Easily**

## **3.1 User Base Growth**

3.1.1 The system must be capable of scaling to support a 100% increase in the user base (from 10,000 to 20,000 concurrent users) within 24 hours without requiring significant downtime or architectural changes.

3.1.2 Auto-scaling mechanisms must dynamically allocate resources to maintain an average page load time of under 3 seconds, even as user numbers increase.

## **3.2 Database and Storage Scalability**

3.2.1 The database must ensure its normal functionality with increased data entries, such as user profiles, custom food entries, and nutritional data, without manual intervention.

3.2.2 As new food items and nutritional data are introduced, the database must be capable of integrating these updates seamlessly. The update can be done once a week.

3.2.3 The system must be designed to update new food stores and food availability information without downtime. The update can be done at 4am every morning automatically.

3.2.4 Automated archiving of older data, such as food logs for users who are inactive for over one year, must be implemented to maintain efficient database operations.

### 3.3 Feature Expansion

3.3.1 The system architecture must be modular, allowing new features such as additional API integrations or functionalities to be added with no more than a 10% increase, per additional feature, in response time for existing features.

3.3.2 New features must be independently deployable, ensuring updates can be implemented without system-wide disruption.

3.3.3 New features must come with a user friendly guide so that 90% users can locate the new feature and understand its core functionality within 1 min.

## 4. Reliability: robustness, safety

### 4.1 System Monitoring

4.1.1 The system must continuously monitor key reliability metrics, including uptime, error rates, and response times.

4.1.2 The system must not return errors for 99% of total transactions.

4.1.3 The System should respond within 2 seconds for 95% of requests.

4.1.4 The system should reach a mean time between failures over 7 days.

### 4.2 Data Leakage Prevention Capability

4.2.1 The system must label users' personal privacy information as secret and record their usage in the log file.

## 5. Supportability: maintainability

### 5.1 Code Maintainability and Transparency

5.1.1 The codebase must have consistent naming conventions, modular design, and thorough documentation, to ensure that the system is easy to maintain and extend.

5.1.2 The system must use version control (e.g., Git) with well-documented commit messages, ensuring that changes can be tracked, reviewed, and rolled back if necessary.

### 5.2 System Monitoring and Alerts

5.2.1 The system must include real-time monitoring for key performance metrics (e.g., server load, response times, database query times), with alerts set up to notify the development team when metrics exceed 80% of defined thresholds.

5.2.2 Logs for all critical system events (e.g., user logins, data updates, API calls) must be maintained and stored for a minimum of 3 months, allowing for auditing and troubleshooting.

### 5.3 Documentation and Knowledge Management

5.3.1 System documentation must be maintained, including data dictionary, use case, requirements, architecture diagrams, API documentation, and setup guides.

5.3.2 A knowledge base must be created and regularly updated with common issues, troubleshooting steps, and FAQs to assist both the support team and end-users.

### 5.4 Maintenance and Updates

5.4.1 The system must support zero-downtime deployments, allowing updates to be rolled out without disrupting user access or functionality.

5.4.2 Regular maintenance windows must be scheduled, and users must be notified in advance. These windows should be used for tasks like database optimization, security patching, and performance tuning.

## **6. Security: authentication**

### **6.1 User Authentication**

6.1.1 The system must implement secure user authentication using a combination of email and password, encrypted using at least SHA-256 encryption, or through authenticated Google account linking.

6.1.2 The system must enforce strong password policies, requiring a minimum of 8 characters, including a mix of at least 3 forms out of uppercase, lowercase, numbers, and special characters.

### **6.2 Data Encryption**

6.2.1 All data transmitted between the client and server must be encrypted using TLS 1.2 or higher to prevent unauthorized access during data transfer.

6.2.2 The system must employ database encryption for stored sensitive data, ensuring that even if the database is compromised, the data remains secure.

### **6.3 Access Control**

6.3.1 Administrative actions (e.g., data deletion, role assignment) must be logged when the system is open to public usage, and access to these logs should be restricted to authorized personnel only.

6.3.2 All user actions that involve sensitive operations must be logged with a timestamp and user ID, with logs retained for at least 3 months.

### **6.4 Data Privacy and Compliance**

6.4.1 The system must comply with data protection regulations, such as GDPR or PDPA, ensuring that user data is handled according to legal requirements.

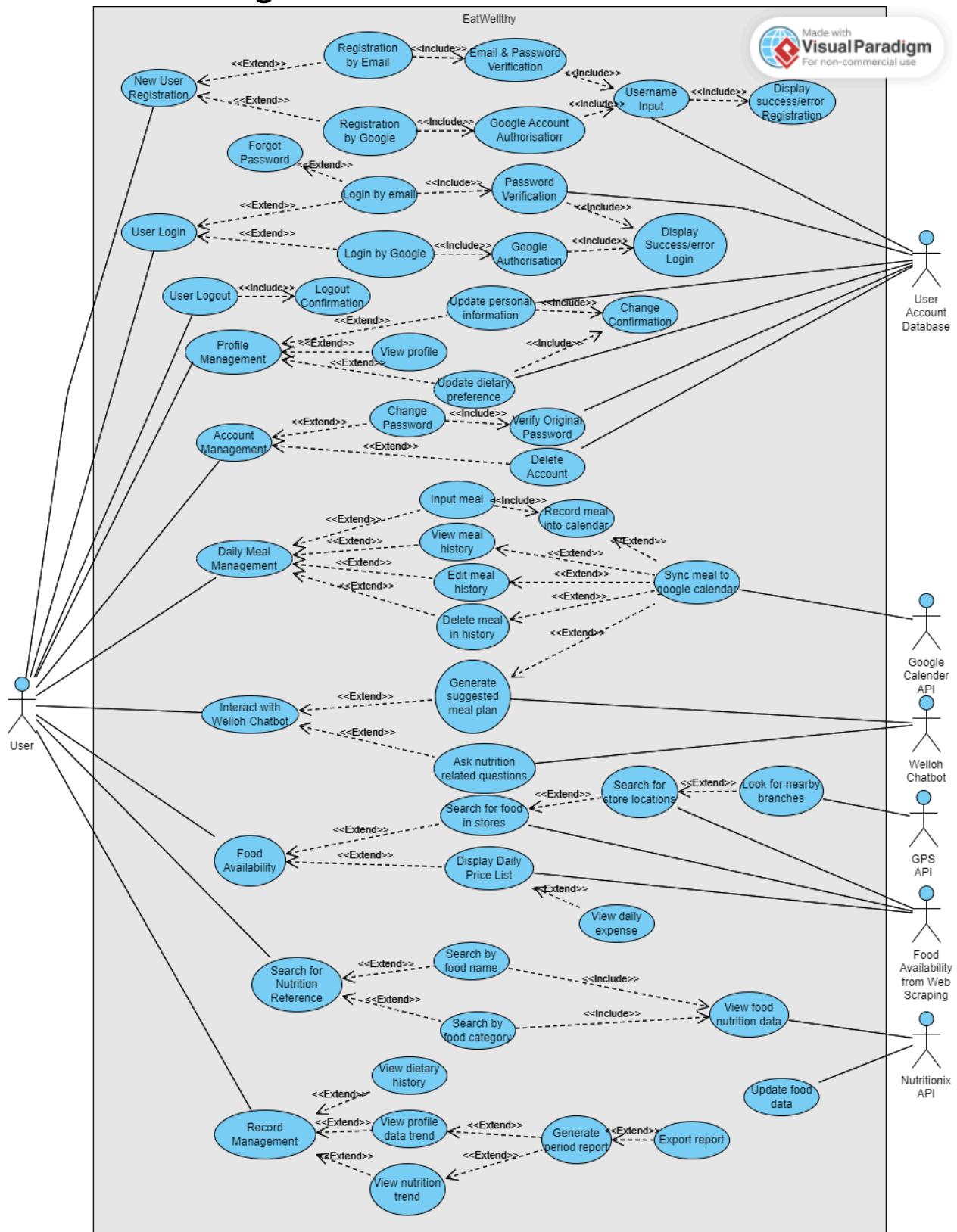
6.4.2 The system must include a privacy policy that clearly explains how user data is collected, used, stored, and protected, with explicit user consent required for data collection and processing.

## 6.5 Vulnerability Management

6.5.1 The system must undergo regular security audits and penetration testing, at least twice a year, to identify and address potential vulnerabilities.

6.5.2 An incident response plan must be in place, detailing the steps to be taken in the event of a data breach, including user notification, data recovery, and mitigation strategies.

# Use Case Diagram



## I.For Functional requirement #1 (Registration)

### U0101 New User Registration using email

Use Case ID:	U0101		
Use Case Name:	New User Registration using email		
Created By:	Zhao Qixian	Last Updated By:	Mahi Pandey
Date Created:	29/8/2024	Date Last Updated:	04/09/2024

Actor:	New User
Description:	It enables new users to create an account within EatWellthy's system database. To create an account, users are required to provide their Username, Email, and Password. New Users will be assigned a unique UserID.
Preconditions:	<ol style="list-style-type: none"> <li>1. The selected Username must be unique.</li> <li>2. Passwords must adhere to the following requirements: <ul style="list-style-type: none"> <li>• Must contain at least 3 out of these 4 conditions: small letter character, capital letter character, number, special character(e.g., @, #, !).</li> <li>• Must be a minimum of 8 characters in length.</li> </ul> </li> <li>3. The supplied Email must not be associated with an existing user account.</li> </ol>
Postconditions:	<p>Upon a successful registration, the system seamlessly executes the following actions:</p> <ul style="list-style-type: none"> <li>• Instantly dispatches a confirmation link to the user's provided Email address, ensuring account verification.</li> <li>• Redirection to the main page of EatWellthy occurs seamlessly.</li> <li>• The user is assigned a unique UserID</li> <li>• The user is now equipped to access the app's features, using their newly created Username and Password, or their Email and Password.</li> <li>• The user's UserID, email, Username and encrypted Password is stored in the database.</li> </ul>
Priority:	High
Frequency of Use:	One-off
Flow of Events:	<ol style="list-style-type: none"> <li>1. The new user opens EatWellthy and selects the "New User Register" option.</li> <li>2. The system presents the user with a registration form, soliciting the following details: <ul style="list-style-type: none"> <li>• Name</li> <li>• Username (I want to be addressed as)</li> </ul> </li> </ol>

	<ul style="list-style-type: none"> <li>• Email</li> <li>• Password</li> </ul> <p>3. The user fills in the necessary information and clicks “Submit”.</p> <p>4. The system conducts immediate validation on the user's input, verifying compliance with the preconditions mentioned earlier.</p> <ul style="list-style-type: none"> <li>• If any precondition is not met, the system displays precise error messages, guiding the user to correct their input.</li> <li>• If the input is valid, the registration process advances, and the user is directed to email verification.</li> </ul>
Alternative Flows:	<ol style="list-style-type: none"> <li>1. Validation Errors <ul style="list-style-type: none"> <li>• If user inputs don't meet preconditions (e.g., invalid password, duplicate Username, existing Email), the system displays error messages for correction.</li> <li>• The user corrects input and resubmits for validation.</li> <li>• Process continues when all input is valid.</li> </ul> </li> <li>2. Password Reset <ul style="list-style-type: none"> <li>• Users can initiate a password reset if forgotten or facing login issues.</li> <li>• The system sends a reset link to the user's Email.</li> <li>• User clicks the link to set a new password.</li> <li>• Users can log in with a new password.</li> </ul> </li> <li>3. Confirmation Link Resend: <ul style="list-style-type: none"> <li>• Users can request a resend of the confirmation link every 60 seconds.</li> <li>• The system sends a new confirmation link to their Email.</li> <li>• User clicks the link to confirm Email and activate the account</li> </ul> </li> </ol>
Exceptions:	<ol style="list-style-type: none"> <li>1. Email Not Received: <ol style="list-style-type: none"> <li>a. If the user does not receive the confirmation email, they may request a resend. If the issue persists, the system advises the user to check spam/junk folders or contact support.</li> </ol> </li> </ol>
Includes:	<ol style="list-style-type: none"> <li>1. Email &amp; Password Verification</li> <li>2. Username Input</li> <li>3. Display Success/Error Registration</li> </ol>
Special Requirements:	<ol style="list-style-type: none"> <li>1. The system must ensure that passwords are stored securely using encryption.</li> <li>2. The system should handle email delivery promptly and securely.</li> </ol>
Assumptions:	<ol style="list-style-type: none"> <li>1. It is assumed that the user has a valid and accessible email address.</li> <li>2. It is assumed that the user understands and complies with the password requirements.</li> </ol>
Notes and Issues:	

## U0102 New User Registration using Google Account

Use Case ID:	U0102		
Use Case Name:	New User Registration using Google Account		
Created By:	Zhao Qixian	Last Updated By:	Mahi Pandey
Date Created:	29/8/2024	Date Last Updated:	04/09/2024

Actor:	New User
Description:	This use case allows new users to register within the EatWellthy system using their existing Google account. By opting for Google-based registration, users can bypass the manual entry of Email, and Password. The system will automatically retrieve the user's Google Email and assign a unique UserID within the EatWellthy database.
Preconditions:	The user must possess an active Google account. The Google account must not already be linked to an existing EatWellthy user account.
Postconditions:	Upon successful registration, the system performs the following actions: <ol style="list-style-type: none"> <li>1. The user is assigned a unique UserID within the system.</li> <li>2. The user is automatically logged in and redirected to the main page of EatWellthy.</li> <li>3. The user's Google Email and associated UserID are stored in the database.</li> <li>4. The user is equipped to access the app's features using their Google account credentials.</li> </ol>
Priority:	Medium
Frequency of Use:	One-off
Flow of Events:	<p>The new user opens EatWellthy and selects the "Register with Google" option.</p> <p>The system prompts the user to sign in to their Google account. The user selects or enters their Google account credentials and authorises access.</p> <p>The system retrieves the user's Google Email and validates it against the EatWellthy database.</p> <ul style="list-style-type: none"> <li>• If the Google Email is already associated with an existing user, an error message is displayed.</li> <li>• If the Google Email is valid and unique, the registration process continues.</li> </ul> <p>The system assigns a unique UserID to the new user and stores the Google Email and UserID in the database.</p> <p>The user is redirected to the main page of EatWellthy and is automatically logged in.</p>
Alternative Flows:	<ol style="list-style-type: none"> <li>1. Google Account Authorization Failure:</li> </ol>

	<ul style="list-style-type: none"> <li>• If the user is unable to authorise their Google account (e.g., incorrect credentials, network issues), the system displays an appropriate error message.</li> <li>• The user may retry authorization or choose another registration method.</li> </ul> <p>2. Existing Google Email:</p> <ul style="list-style-type: none"> <li>• If the Google Email is already linked to an EatWellthy account, the system notifies the user and offers the option to log in instead.</li> </ul>
Exceptions:	If the Google API is unavailable, the system should inform the user and suggest trying again later or using an alternate registration method.
Includes:	<ol style="list-style-type: none"> <li>1. Google Account Authorisation</li> <li>2. Username Input</li> <li>3. <u>Display Success/Error Registration</u></li> </ol>
Special Requirements:	The system must adhere to Google's OAuth 2.0 standards for secure authentication and data retrieval.
Assumptions:	It is assumed that the user has a Google account and can access it for registration.
Notes and Issues:	

## U0103 User Login via Email

Use Case ID:	U0103		
Use Case Name:	User Login via Email		
Created By:	Mahi Pandey	Last Updated By:	Mahi Pandey
Date Created:	01/09/2024	Date Last Updated:	02/09/2024

Actor:	Registered User
Description:	This use case enables registered users to log into the EatWellthy system using their email and password. Successful login grants access to the app's features.
Preconditions:	<ol style="list-style-type: none"> <li>1. The user must have a valid registered account.</li> <li>2. The user must provide a valid email and password.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. The user is successfully logged into the system.</li> <li>2. The user's session is authenticated and maintained</li> <li>3. The user is redirected to the EatWellthy main page</li> </ol>
Priority:	High
Frequency of Use:	Every time user wishes to login, Multiple Times
Flow of Events:	<ol style="list-style-type: none"> <li>1. The user opens the EatWellthy app and selects the Login option.</li> <li>2. The system presents the login form, requesting Email and Password.</li> <li>3. The user enters their Email and Password and clicks "Login"</li> <li>4. The system validates the credentials. <ul style="list-style-type: none"> <li>• If valid, the system logs in the user and redirects them to the main page.</li> <li>• If invalid, the system displays an error message.</li> </ul> </li> </ol>
Alternative Flows:	<ol style="list-style-type: none"> <li>1. Invalid Credentials <ul style="list-style-type: none"> <li>• The system informs the user of incorrect Email or Password and prompts re-entry.</li> </ul> </li> <li>2. Password Reset <ul style="list-style-type: none"> <li>• If the user forgets their password, they can select "Forgot Password" to initiate a reset process.</li> </ul> </li> </ol>
Exceptions:	<ol style="list-style-type: none"> <li>1. If the system is down or the database is unreachable, an error message is displayed, and the user is asked to try again later.</li> </ol>
Includes:	<ol style="list-style-type: none"> <li>1. Password Verification</li> <li>2. Display Success/Error Login</li> </ol>
Special Requirements:	The system must ensure secure transmission of login credentials.
Assumptions:	It is assumed that the user has registered and verified their email address.
Notes and Issues:	Consider implementing CAPTCHA to avoid bots.

## U0104 User Login via Google Account

Use Case ID:	U0104		
Use Case Name:	User Login via Google Account		
Created By:	Mahi Pandey	Last Updated By:	Mahi Pandey
Date Created:	02/09/2024	Date Last Updated:	02/09/2024

Actor:	Registered User
Description:	This use case allows users to log into the EatWellthy system using their Google Account. Upon successful login, users are granted access to the website's features.
Preconditions:	<ol style="list-style-type: none"> <li>1. The user must have a Google account</li> <li>2. The Google account must be linked to an EatWellthy user profile.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. The user is successfully logged into the system.</li> <li>2. The user session is authenticated and maintained.</li> <li>3. The user is redirected to the EatWellthy main page.</li> </ol>
Priority:	Medium
Frequency of Use:	Every time user wishes to login, Multiple Times
Flow of Events:	<ol style="list-style-type: none"> <li>1. The user selects "Login with Google".</li> <li>2. System redirects the user to Google's login page.</li> <li>3. The user enters their Google credentials and authorises access.</li> <li>4. System retrieves the user's Google Account. <ul style="list-style-type: none"> <li>• If valid, the user is logged in and redirected to the main page.</li> <li>• If invalid, an error message is displayed.</li> </ul> </li> </ol>
Alternative Flows:	<ol style="list-style-type: none"> <li>1. Google Authorization Failure <ul style="list-style-type: none"> <li>• If the user fails to log into their Google Account, the system displays an error message.</li> <li>• The user can retry or choose another login method</li> </ul> </li> <li>2. Account Does Not Exist <ul style="list-style-type: none"> <li>• If the user does not have a pre-existing account linked to their Google Account, the system displays an error message.</li> <li>• The user can retry or choose another login method</li> </ul> </li> </ol>
Exceptions:	<ol style="list-style-type: none"> <li>1. If Google API is unavailable, the user is informed and asked to try again later or use email login method.</li> </ol>
Includes:	<ol style="list-style-type: none"> <li>1. Google Authorisation</li> <li>2. Display Success/Error Login</li> </ol>
Special Requirements:	System must comply with Google's OAuth 2.0 standards.
Assumptions:	The user has a valid and accessible Google account.
Notes and Issues:	

## U0105 Reset Forgotten Password

Use Case ID:	U0105		
Use Case Name:	Reset Forgotten Password		
Created By:	Mahi Pandey	Last Updated By:	Mahi Pandey
Date Created:	02/09/2024	Date Last Updated:	02/09/2024

Actor:	Registered User
Description:	This use case enables users who have forgotten their password to reset it using their registered email address.
Preconditions:	<ol style="list-style-type: none"><li>1. The user must have a registered account with EatWellthy.</li><li>2. The user must have access to the registered email address.</li></ol>
Postconditions:	<ol style="list-style-type: none"><li>1. The user successfully resets their password.</li><li>2. The user is prompted to log in with the new password.</li></ol>
Priority:	Medium
Frequency of Use:	Infrequent, as needed
Flow of Events:	<ol style="list-style-type: none"><li>1. The user selects "Forgot Password" on the login page.</li><li>2. The system prompts the user to enter their registered email.</li><li>3. The user enters their email and clicks "Submit".</li><li>4. The system verifies the email and sends a password reset link.</li><li>5. The user receives the email and clicks the reset link.</li><li>6. The system prompts the user to enter and confirm a new password.</li><li>7. The user submits the new password, and the system updates their account.</li><li>8. The user is prompted to log in with the new password.</li></ol>
Alternative Flows:	<ol style="list-style-type: none"><li>1. Invalid Email<ul style="list-style-type: none"><li>• If the entered email is not registered, the system informs the user and prompts for re-entry.</li></ul></li></ol>
Exceptions:	If the system cannot send the reset email, the user is informed of a delay and asked to try again later.
Includes:	
Special Requirements:	The system must ensure secure handling of password reset requests.
Assumptions:	The user can access their registered email account.
Notes and Issues:	Consider implementing a rate limit for password reset requests to prevent abuse.

## U0106 Change Account Password

Use Case ID:	U0106		
Use Case Name:	Change Account Password		
Created By:	Mahi Pandey	Last Updated By:	Mahi Pandey
Date Created:	02/09/2024	Date Last Updated:	02/09/2024

Actor:	Registered User
Description:	This use case allows registered users to change their account password while logged into the system.
Preconditions:	<ol style="list-style-type: none"> <li>1. The user must be logged into their account.</li> <li>2. The user must know their current password.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. The user's password is successfully changed.</li> <li>2. The system logs the user out and prompts them to log in again with the new password.</li> </ol>
Priority:	Medium
Frequency of Use:	Infrequent, as needed
Flow of Events:	<ol style="list-style-type: none"> <li>1. The user navigates to the "Account Settings" page.</li> <li>2. The system presents the option to change the password.</li> <li>3. The user enters their current password and the new password, then confirms new password.</li> <li>4. The system validates the current password and checks the new password against security requirements.</li> <li>5. If valid, the system updates the password and logs the user out.</li> <li>6. The user is prompted to log in with new password.</li> </ol>
Alternative Flows:	<ol style="list-style-type: none"> <li>1. Incorrect Current Password <ul style="list-style-type: none"> <li>• If the current password is incorrect, the system informs the user and allows re-entry.</li> </ul> </li> <li>2. Password Strength Failure <ul style="list-style-type: none"> <li>• If the new password does not meet security requirements, the system prompts the user to enter a stronger password.</li> </ul> </li> <li>3. New Password Equivalent to Current Password <ul style="list-style-type: none"> <li>• If the new password is the same as the current password, the system prompts the user to enter a different password.</li> </ul> </li> </ol>
Exceptions:	If the system is unable to update the password due to technical issues, an error message is displayed, and the user is asked to try again later.
Includes:	<ol style="list-style-type: none"> <li>1. Verify Original Password</li> </ol>
Special Requirements:	The system must ensure secure handling and storage of passwords.
Assumptions:	The user has access to their current password.
Notes and Issues:	Provide feedback on the strength of the password.

## II. For Functional Requirement #3 (Profile Management)

### U0201 View User Profile

Use Case ID:	U0201		
Use Case Name:	View User Profile		
Created By:	Mahi Pandey	Last Updated By:	Mahi Pandey
Date Created:	02/09/2024	Date Last Updated:	02/09/2024

Actor:	Logged-in & Registered User
Description:	This use case allows users to view their profile information within the EatWellthy app, including personal details, dietary preferences and activity history.
Preconditions:	<ol style="list-style-type: none"> <li>1. The user must be logged into their EatWellthy account.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. The user's profile information is displayed on the screen</li> </ol>
Priority:	Low
Frequency of Use:	Occasional
Flow of Events:	<ol style="list-style-type: none"> <li>1. The user selects "View Profile" from the account menu.</li> <li>2. The system retrieves the user's profile data from the database.</li> <li>3. The system displays the profile information to the user.</li> </ol>
Alternative Flows:	NIL
Exceptions:	If the system is unable to retrieve the profile data, an error message is displayed.
Includes:	
Special Requirements:	
Assumptions:	It is assumed that the user has profile information stored in the database.
Notes and Issues:	

## U0202 Update User Profile Information

Use Case ID:	U0202		
Use Case Name:	Update User Profile Information		
Created By:	Mahi Pandey	Last Updated By:	Mahi Pandey
Date Created:	02/09/2024	Date Last Updated:	02/09/2024

Actor:	Logged-in & Registered User
Description:	This use case allows users to update their profile information, such as name, email, dietary preferences, and other personal details within the EatWellthy website.
Preconditions:	<ol style="list-style-type: none"> <li>The user must be logged into their EatWellthy account.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>The user's profile information is successfully updated in the database.</li> </ol>
Priority:	Medium
Frequency of Use:	Occasionally, as needed
Flow of Events:	<ol style="list-style-type: none"> <li>The user navigates to "Edit Profile" from their account menu.</li> <li>The system displays the current profile information in editable fields.</li> <li>The user updates the desired fields and clicks "Save".</li> <li>The system validates the new information. <ul style="list-style-type: none"> <li>If valid, the system updates the profile in the database and confirms the update.</li> <li>If invalid, the system prompts the user to correct the information.</li> </ul> </li> </ol>
Alternative Flows:	<ol style="list-style-type: none"> <li>Invalid Input: <ul style="list-style-type: none"> <li>If any new information is invalid (e.g., incorrectly formatted email), the system informs the user and prevents saving until corrected.</li> </ul> </li> </ol>
Exceptions:	If the system fails to update the profile due to technical issues, an error message is displayed, and the user is asked to try again later.
Includes:	<ol style="list-style-type: none"> <li>Change Confirmation</li> </ol>
Special Requirements:	The system must ensure that updates are secure and reflect immediately across the platform.
Assumptions:	It is assumed that the user has valid profile information that they want updated.
Notes and Issues:	

## U0203 Manage Dietary Preferences

Use Case ID:	U0203		
Use Case Name:	Manage Dietary Preferences		
Created By:	Mahi Pandey	Last Updated By:	Mahi Pandey
Date Created:	02/09/2024	Date Last Updated:	02/09/2024

Actor:	Logged-in & registered user
Description:	This use case allows users to specify or update their dietary preferences within the EatWellthy app, which will be used to tailor meal recommendations.
Preconditions:	<ol style="list-style-type: none"> <li>The user must be logged into their EatWellthy account.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>The user's dietary preferences are successfully updated in the system.</li> </ol>
Priority:	High
Frequency of Use:	Occasional, as needed
Flow of Events:	<ol style="list-style-type: none"> <li>The user navigates to "Dietary Preferences" from their profile settings.</li> <li>The system displays the current dietary preferences.</li> <li>The user selects or modifies their dietary preferences and clicks "Save".</li> <li>The system updates the preferences in the database and confirms the update.</li> </ol>
Alternative Flows:	<ol style="list-style-type: none"> <li>Invalid Preference <ul style="list-style-type: none"> <li>If any preference selection is invalid, the system prompts the user to correct it before saving.</li> </ul> </li> </ol>
Exceptions:	If the system is unable to update dietary preferences due to technical issues, an error message is displayed, and the user is asked to try again later.
Includes:	<ol style="list-style-type: none"> <li>Change Confirmation</li> </ol>
Special Requirements:	
Assumptions:	It is assumed that the user has dietary preferences that they wish to manage.
Notes and Issues:	

## U0204 Update Security Settings

Use Case ID:	U0204		
Use Case Name:	Update Security Settings		
Created By:	Mahi Pandey	Last Updated By:	Mahi Pandey
Date Created:	02/09/2024	Date Last Updated:	02/09/2024

Actor:	Logged-in User
Description:	This use case allows users to update their security settings within the EatWellthy app, such as enabling two-factor authentication or changing security questions.
Preconditions:	<ol style="list-style-type: none"> <li>The user must be logged into their EatWellthy account.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>The user's security settings are successfully updated and take effect immediately.</li> </ol>
Priority:	Low
Frequency of Use:	Occasional, as needed
Flow of Events:	<ol style="list-style-type: none"> <li>The user navigates to "Security Settings" from their account menu.</li> <li>The system displays the current security settings.</li> <li>The user updates the desired security settings (e.g., enabling two-factor authentication).</li> <li>The system validates the changes and updates the settings in the database.</li> <li>The system confirms the update and informs the user of the changes.</li> </ol>
Alternative Flows:	<ol style="list-style-type: none"> <li>Invalid Security Settings <ul style="list-style-type: none"> <li>If any security setting is invalid or fails validation (e.g., weak security question answer), the system informs the user and prevents saving until corrected.</li> </ul> </li> </ol>
Exceptions:	If the system fails to update the security settings due to technical issues, an error message is displayed, and the user is asked to try again later.
Includes:	<ol style="list-style-type: none"> <li>Verify Password</li> </ol>
Special Requirements:	The system must ensure that security updates are handled with the highest level of protection.
Assumptions:	It is assumed that the user has security settings that they wish to manage.
Notes and Issues:	Can consider implementing an email alert system to notify users if their security information is updated.

## U0205 Delete User Account

Use Case ID:	U0205		
Use Case Name:	Delete User Account		
Created By:	Mahi Pandey	Last Updated By:	Mahi Pandey
Date Created:	02/09/2024	Date Last Updated:	02/09/2024

Actor:	Logged-in & Registered User
Description:	This use case allows users to permanently delete their EatWellthy account. Deleting an account removes all user data from the system and makes the account irretrievable.
Preconditions:	<ol style="list-style-type: none"> <li>The user must be logged into their EatWellthy account.</li> <li>The user must confirm their intention to permanently delete the account.</li> <li>The system must inform the user of the consequences of deletion, including the irretrievable loss of data.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>The user's account and all associated data are permanently removed from the system.</li> <li>The user is logged out and cannot recover the account or data.</li> </ol>
Priority:	Medium
Frequency of Use:	Infrequent, as needed
Flow of Events:	<ol style="list-style-type: none"> <li>The user navigates to "Account Settings" and selects the "Delete Account" option.</li> <li>The system prompts the user to confirm their decision to permanently delete the account.</li> <li>The user confirms the deletion by providing their password for security verification.</li> <li>The system warns the user about the permanent nature of account deletion and the loss of all data.</li> <li>The user confirms the deletion.</li> <li>The system permanently removes the user's account and all associated data from the database.</li> <li>The user is logged out and redirected to the homepage with a confirmation of account deletion.</li> </ol>
Alternative Flows:	<ol style="list-style-type: none"> <li>Cancellation of Deletion <ul style="list-style-type: none"> <li>If the user decides not to delete the account after being prompted for confirmation, they can cancel the process, and no changes are made to the account.</li> </ul> </li> <li>Data Export Option <ul style="list-style-type: none"> <li>Before confirming deletion, the system offers the user the option to export their data. If the user chooses to export, the system generates a downloadable file with the user's data.</li> </ul> </li> </ol>

Exceptions:	If the system fails to delete the account due to technical issues, an error message is displayed, and the user is advised to try again later or contact support.
Includes:	
Special Requirements:	<ol style="list-style-type: none"> <li>1. The system must ensure that the deletion process is secure and irreversible.</li> <li>2. The system should comply with relevant data protection regulations to avoid legal disputes.</li> </ol>
Assumptions:	It is assumed that the user understands the implications of account deletion, including the permanent loss of access and data.
Notes and Issues:	

### III. For Functional Requirement #4 (Nutrition Calculation)

#### U0301 Log Daily Meal

Use Case ID:	U0301		
Use Case Name:	Log Daily Meal		
Created By:	Liu Xiaotao	Last Updated By:	Liu Xiaotao
Date Created:	1/9/2024	Date Last Updated:	1/9/2024

Actor:	Current User
Description:	This use case enables users to log their daily meals into the EatWellthy system. Users can input details such as meal type (breakfast, lunch, dinner, snacks), food items, portion sizes, and time of consumption. The system will store this information.
Preconditions:	<ol style="list-style-type: none"> <li>1. The user must be logged into the EatWellthy system with a valid account.</li> <li>2. The system must have access to the user's profile to store meal logs.</li> </ol>
Postconditions:	<ul style="list-style-type: none"> <li>• The meal information is successfully stored in the user's profile in the database.</li> <li>• The system updates the user's nutritional summary for the day, reflecting the logged meal.</li> <li>• Users can view, edit, or delete the meal entry at any time.</li> <li>• The system may provide feedback or suggestions based on the nutritional content of the logged meals.</li> </ul>
Priority:	High
Frequency of Use:	Daily
Flow of Events:	<ol style="list-style-type: none"> <li>1. The user accesses the EatWellthy application and navigates to the "Log Daily Meal" section.</li> <li>2. The system presents the user with a form to input meal details: <ul style="list-style-type: none"> <li>• Meal type (e.g., breakfast, lunch, dinner, snacks)</li> <li>• Food items consumed</li> <li>• Portion sizes</li> <li>• Time of consumption</li> </ul> </li> <li>3. The user fills in the necessary information and clicks "Submit."</li> <li>4. The system validates the input: <ul style="list-style-type: none"> <li>• Ensures that all required fields are filled.</li> <li>• Verifies that the portion sizes are reasonable (e.g., not negative).</li> </ul> </li> <li>5. If the input is valid, the system stores the meal information in the user's profile.</li> </ol>

	<p>6. The system updates the user's daily nutritional summary with the new data.</p> <p>7. The system displays a confirmation message to the user, indicating that the meal has been successfully logged.</p>
Alternative Flows:	<p>1. Validation Errors:</p> <ul style="list-style-type: none"> <li>• If any required fields are missing or contain invalid data, the system displays an error message.</li> <li>• The user corrects the input and resubmits the form.</li> <li>• The process continues once the input is valid.</li> </ul> <p>2. Editing a Meal Log:</p> <ul style="list-style-type: none"> <li>• The user can navigate to their meal history and select a previously logged meal.</li> <li>• The system displays the meal details, allowing the user to make changes.</li> <li>• The user submits the changes, and the system updates the meal information in the database and adjusts the nutritional summary accordingly.</li> </ul> <p>3. Deleting a Meal Log:</p> <ul style="list-style-type: none"> <li>• The user selects a meal from their meal history that they wish to delete.</li> <li>• The system prompts the user to confirm the deletion.</li> <li>• Upon confirmation, the system removes the meal from the user's profile and adjusts the nutritional summary</li> </ul>
Exceptions:	<p>Network Failure:</p> <ul style="list-style-type: none"> <li>• If there is a network issue, the system will prompt the user to retry or save the meal log locally until the network is restored.</li> </ul> <p>System Unavailability:</p> <ul style="list-style-type: none"> <li>• If the system is down for maintenance or encounters an error, the user will receive a notification and will be unable to log meals until the issue is resolved.</li> </ul>
Includes:	Confirm message
Special Requirements:	
Assumptions:	
Notes and Issues:	

## U0302 Edit Logged Meal

Use Case ID:	U302		
Use Case Name:	Edit Logged Meal		
Created By:	Liu Xiaotao	Last Updated By:	Liu Xiaotao
Date Created:	1/9/2024	Date Last Updated:	1/9/2024

Actor:	Current User
Description:	This use case allows users to edit a previously logged meal in the EatWellthy system. Users can modify details such as meal type, food items, portion sizes, and time of consumption. The system updates the user's nutritional summary to reflect the changes.
Preconditions:	<ol style="list-style-type: none"> <li>1. The user must be logged into the EatWellthy system with a valid account.</li> <li>2. The user must have previously logged a meal that they wish to edit.</li> <li>3. The system must have access to the user's meal history.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. The updated meal information is successfully stored in the user's profile in the database.</li> <li>2. The system recalculates and updates the user's daily nutritional summary based on the changes.</li> <li>3. The user can view the updated meal entry and the adjusted nutritional information.</li> </ol>
Priority:	
Frequency of Use:	As needed
Flow of Events:	<ol style="list-style-type: none"> <li>1. The user accesses the EatWellthy application and navigates to their meal history.</li> <li>2. The system presents a list of previously logged meals, organized by date and time.</li> <li>3. The user selects the meal they wish to edit.</li> <li>4. The system displays the meal details, including: <ul style="list-style-type: none"> <li>• Meal type (e.g., breakfast, lunch, dinner, snacks)</li> <li>• Food items consumed</li> <li>• Portion sizes</li> <li>• Time of consumption</li> </ul> </li> <li>5. The user makes the necessary changes to the meal details.</li> <li>6. The user clicks "Save" or "Update" to submit the changes.</li> <li>7. The system validates the new input: <ul style="list-style-type: none"> <li>• Ensures all required fields are filled.</li> </ul> </li> </ol>

	<ul style="list-style-type: none"> <li>• Verifies that the portion sizes are reasonable (e.g., not negative).</li> <li>8. If the input is valid, the system updates the meal information in the user's profile.</li> <li>9. The system recalculates the user's daily nutritional summary to reflect the updated meal data.</li> <li>10. The system displays a confirmation message, indicating that the meal has been successfully updated.</li> </ul>
Alternative Flows:	<ol style="list-style-type: none"> <li>1. Validation Errors: <ul style="list-style-type: none"> <li>• If any required fields are missing or contain invalid data, the system displays an error message.</li> <li>• The user corrects the input and resubmits the form.</li> <li>• The process continues once the input is valid.</li> </ul> </li> <li>2. Cancelling an Edit: <ul style="list-style-type: none"> <li>• The user may choose to cancel the edit at any time before saving.</li> <li>• The system discards any changes made and returns to the meal history view without altering the original meal log.</li> </ul> </li> <li>3. Undoing Changes: <ul style="list-style-type: none"> <li>• After saving an edit, the user may realize they made a mistake.</li> </ul> </li> </ol>
Exceptions:	<ol style="list-style-type: none"> <li>1. Network Failure: <ul style="list-style-type: none"> <li>• If there is a network issue, the system will prompt the user to retry or save the meal log locally until the network is restored.</li> </ul> </li> <li>2. System Unavailability: <ul style="list-style-type: none"> <li>• If the system is down for maintenance or encounters an error, the user will receive a notification and will be unable to log meals until the issue is resolved.</li> </ul> </li> </ol>
Includes:	Confirmation message
Special Requirements:	
Assumptions:	
Notes and Issues:	

## U0303 Delete Logged Meal

Use Case ID:	U303		
Use Case Name:	Delete Logged Meal		
Created By:	LiuXiaotao	Last Updated By:	Liu Xiaotao
Date Created:	1/9/2024	Date Last Updated:	1/9/2024

Actor:	Current User
Description:	This use case allows users to delete a previously logged meal from the EatWellthy system. Once deleted, the meal information is removed from the user's profile, and the system updates the user's daily nutritional summary to reflect the deletion.
Preconditions:	<ol style="list-style-type: none"> <li>1. The user must be logged into the EatWellthy system with a valid account.</li> <li>2. The user must have at least one previously logged meal in their history.</li> <li>3. The system must have access to the user's meal history.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. The selected meal is permanently removed from the user's profile in the database.</li> <li>2. The system recalculates and updates the user's daily nutritional summary based on the deletion.</li> <li>3. The user can view their updated meal history and nutritional information.</li> </ol>
Priority:	
Frequency of Use:	As needed
Flow of Events:	<ol style="list-style-type: none"> <li>1. The user accesses the EatWellthy application and navigates to their meal history.</li> <li>2. The system presents a list of previously logged meals, organized by date and time.</li> <li>3. The user selects the meal they wish to delete.</li> <li>4. The system displays the meal details, including: <ul style="list-style-type: none"> <li>• Meal type (e.g., breakfast, lunch, dinner, snacks)</li> <li>• Food items consumed</li> <li>• Portion sizes</li> <li>• Time of consumption</li> </ul> </li> <li>5. The user clicks the "Delete" button to remove the selected meal.</li> <li>6. The system prompts the user to confirm the deletion to prevent accidental removal.</li> <li>7. The user confirms the deletion by selecting "Yes" or "Confirm".</li> <li>8. The system validates the request and proceeds to delete the meal from the user's profile.</li> <li>9. The system recalculates the user's daily nutritional summary to reflect the removal of the meal data.</li> </ol>

	<p>10. The system displays a confirmation message, indicating that the meal has been successfully deleted.</p> <p>11. The user's meal history is updated to exclude the deleted meal.</p>
Alternative Flows:	<p>1. Cancelling the Deletion:</p> <ul style="list-style-type: none"> <li>• If the user decides not to delete the meal, they can select "No" or "Cancel" at the confirmation prompt.</li> <li>• The system aborts the deletion process and returns to the meal history view with no changes made.</li> </ul> <p>2. Restoring a Deleted Meal:</p> <ul style="list-style-type: none"> <li>• If the user accidentally deletes a meal, they may have the option to restore it within a limited time (e.g., 5 minutes) by selecting an "Undo" option.</li> <li>• The system restores the deleted meal details and updates the nutritional summary accordingly.</li> </ul> <p>3. Multiple Deletions:</p> <ul style="list-style-type: none"> <li>• The user can select and delete multiple meals at once.</li> <li>• The system provides a batch delete option, and the user confirms the deletion of all selected meals.</li> <li>• The system then updates the nutritional summary based on the collective deletion.</li> </ul>
Exceptions:	<p>1. Network Failure:</p> <ul style="list-style-type: none"> <li>• If there is a network issue, the system will prompt the user to retry or save the meal log locally until the network is restored.</li> </ul> <p>2. System Unavailability:</p> <ul style="list-style-type: none"> <li>• If the system is down for maintenance or encounters an error, the user will receive a notification and will be unable to log meals until the issue is resolved.</li> </ul>
Includes:	
Special Requirements:	The system should ask for confirmation before the data is deleted
Assumptions:	
Notes and Issues:	

## U0304 View Meal History

Use Case ID:	U304		
Use Case Name:	View Meal History		
Created By:	Liu Xiaotao	Last Updated By:	Liu Xiaotao
Date Created:	1/9/2024	Date Last Updated:	1/9/2024

Actor:	Current User
Description:	This use case allows users to view their logged meal history within the EatWellthy system. Users can access details of past meals, including the type of meal, food items consumed, portion sizes, and nutritional information.
Preconditions:	<ol style="list-style-type: none"> <li>1. The user must be logged into the EatWellthy system with a valid account.</li> <li>2. The user must have logged at least one meal in the system.</li> <li>3. The system must have access to the user's meal history data.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. The user must be logged into the EatWellthy system with a valid account.</li> <li>2. The user must have logged at least one meal in the system.</li> <li>3. The system must have access to the user's meal history data.</li> </ol>
Priority:	
Frequency of Use:	As needed
Flow of Events:	<ol style="list-style-type: none"> <li>1. The user accesses the EatWellthy application and navigates to the "Meal History" section.</li> <li>2. The system retrieves the user's logged meal data from the database.</li> <li>3. The system presents the meal history in a chronological list, displaying the following for each meal: <ul style="list-style-type: none"> <li>• Date and time of the meal</li> <li>• Meal type (e.g., breakfast, lunch, dinner, snacks)</li> <li>• Summary of food items consumed</li> <li>• Total calories and key nutritional information (e.g., protein, carbs, fats)</li> </ul> </li> <li>4. The user selects a specific meal entry to view more detailed information.</li> <li>5. The system displays the full details of the selected meal, including: <ul style="list-style-type: none"> <li>• Individual food items with portion sizes</li> </ul> </li> </ol>

	<ul style="list-style-type: none"> <li>• Detailed nutritional information for each food item (e.g., calories, macronutrients)</li> <li>• Any notes or additional information entered by the user</li> <li>6. The user can navigate back to the overall meal history or select another meal to view.</li> <li>7. The user can use filtering options (e.g., by date, meal type) to refine the meal history display.</li> <li>8. The system allows the user to search for specific meals by keywords or dates.</li> <li>9. The user can exit the meal history section and return to the main dashboard or another section of the application.</li> </ul>
Alternative Flows:	<ol style="list-style-type: none"> <li>1. No Logged Meals: <ul style="list-style-type: none"> <li>• If the user has not logged any meals, the system displays a message indicating that no meal history is available.</li> <li>• The user is prompted to log their first meal.</li> </ul> </li> <li>2. Filtering and Sorting: <ul style="list-style-type: none"> <li>• The user can apply filters to view specific types of meals (e.g., only breakfasts) or sort the history by criteria such as date, calories, or specific nutrients.</li> <li>• The system adjusts the display to show only the filtered or sorted meals.</li> </ul> </li> <li>3. Searching Meal History: <ul style="list-style-type: none"> <li>• The user enters a search term (e.g., "salad," "chicken") in the search bar.</li> <li>• The system displays meals that match the search criteria, highlighting the relevant entries.</li> </ul> </li> </ol>
Exceptions:	<ol style="list-style-type: none"> <li>1. Data Retrieval Failure: <ul style="list-style-type: none"> <li>• If the system fails to retrieve meal history due to a network or server issue, the user is notified with an error message.</li> <li>• The system may offer an option to retry or advise the user to check back later.</li> </ul> </li> <li>2. Incomplete Data: <ul style="list-style-type: none"> <li>• If any part of the meal history data is incomplete or corrupted, the system displays a warning and attempts to recover or display what is available.</li> <li>• The user is informed about the incomplete data and is given options to refresh or contact support.</li> </ul> </li> </ol>
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	Avoid the situation that the system loaded the data of another user

## U0305 Generate Personalized Diet Plan

Use Case ID:	U305		
Use Case Name:	Generate Personalised Diet Plan		
Created By:	Liu Xiaotao	Last Updated By:	Liu Xiaotao
Date Created:	2/9/2024	Date Last Updated:	2/9/2024

Actor:	Current User
Description:	This use case allows users to generate a personalized diet plan tailored to their individual health goals, dietary preferences, and nutritional needs. The system takes into account various user inputs, such as age, gender, weight, height, activity level, and specific dietary restrictions or preferences, to create a customized diet plan.
Preconditions:	<ol style="list-style-type: none"> <li>1. The user must be logged into the EatWellthy application with a valid account.</li> <li>2. The user must provide relevant health and dietary information within the application.</li> <li>3. The system must have access to a comprehensive nutrition database to generate accurate diet plans.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. The user receives a personalized diet plan that aligns with their health goals and dietary preferences.</li> <li>2. The generated diet plan is stored in the user's profile for easy access and future reference.</li> <li>3. The user can follow the diet plan and track their progress within the EatWellthy application.</li> </ol>
Priority:	High
Frequency of Use:	Daily
Flow of Events:	<ol style="list-style-type: none"> <li>1. The user logs into the EatWellthy application and navigates to the "Generate Diet Plan" section.</li> <li>2. The system prompts the user to provide or update the following information: <ul style="list-style-type: none"> <li>• Age, Gender, Weight, Height</li> <li>• Activity level (e.g., sedentary, lightly active, moderately active, very active)</li> <li>• Health goals (e.g., weight loss, muscle gain, maintenance)</li> <li>• Dietary preferences (e.g., vegetarian, vegan, low-carb, gluten-free)</li> <li>• Any food allergies or restrictions</li> </ul> </li> <li>3. The user enters the required information and confirms their input.</li> <li>4. The system validates the user's input for completeness and accuracy. If any required</li> </ol>

	<p>information is missing or incorrect, the system prompts the user to correct it.</p> <ol style="list-style-type: none"> <li>5. Upon successful validation, the system accesses its nutrition database and calculate the user's daily caloric needs and macronutrient distribution (e.g., protein, fats, carbohydrates).</li> <li>6. The system generates a personalized diet plan, including:           <ul style="list-style-type: none"> <li>• Suggested daily calorie intake</li> <li>• Recommended meals and snacks</li> <li>• Portion sizes for each meal</li> <li>• Nutritional breakdown (e.g., calories, macronutrients, vitamins, minerals) for each meal</li> <li>• Meal timing recommendations (e.g., breakfast, lunch, dinner, snacks)</li> </ul> </li> <li>7. The system displays the personalized diet plan to user, allowing them to review and confirm the plan.</li> <li>8. The user can either accept the diet plan as-is or make adjustments (e.g., swap meals, modify portion sizes).</li> <li>9. Once satisfied, the user saves the personalized diet plan to their profile.</li> <li>10. The system stores the diet plan in the user's profile and makes it accessible from the dashboard for daily reference and meal logging.</li> </ol>
Alternative Flows:	<ol style="list-style-type: none"> <li>1. Plan Regeneration           <ul style="list-style-type: none"> <li>• If the user is not satisfied with the initial diet plan, they can regenerate the plan with adjusted inputs (e.g., changing the health goal or activity level).</li> <li>• The system generates a new plan based on the updated inputs.</li> </ul> </li> </ol>
Exceptions:	<ol style="list-style-type: none"> <li>1. Database Access Issues:           <ul style="list-style-type: none"> <li>• If the system cannot access the nutrition database (e.g., due to a network issue), it displays an error message and suggests the user try again later.</li> <li>• The user is informed that their current data will be saved and can be used to generate the diet plan once the issue is resolved.</li> </ul> </li> <li>2. Inconsistent or Conflicting Inputs:           <ul style="list-style-type: none"> <li>• If the user's inputs are inconsistent or conflict with each other (e.g., selecting both high-carb and low-carb preferences), the system alerts the user and requests clarification.</li> <li>• The user can modify their inputs to resolve the conflict and proceed with generating the diet plan.</li> </ul> </li> </ol>
Includes:	
Special Requirements:	User should be able to copy the generated plan and paste it in their personal diet site
Assumptions:	
Notes and Issues:	

## U0306 Sync Diet Plan with Google Calendar

Use Case ID:	U306		
Use Case Name:	Sync Diet Plan with Google Calendar		
Created By:	Liu Xiaotao	Last Updated By:	Liu Xiaotao
Date Created:	1/9/2024	Date Last Updated:	1/9/2024

Actor:	Current User
Description:	This use case allows users to sync their diet plan from EatWellthy with their Google Calendar. By doing this, users can have meal reminders, diet plan events, and nutritional goals automatically appear in their Google Calendar.
Preconditions:	<ol style="list-style-type: none"> <li>1. The user must be logged into the EatWellthy system with a valid account.</li> <li>2. The user must have a diet plan created within the EatWellthy application.</li> <li>3. The user must have an active Google account and must be signed into Google services.</li> <li>4. The user must authorise the EatWellthy app to access and manage their Google Calendar.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. The user's diet plan is successfully synced with their Google Calendar.</li> <li>2. The system automatically creates calendar events for each meal or diet-related activity, based on the user's diet plan.</li> <li>3. Users receive notifications on their devices as per the scheduled times in the Google Calendar.</li> </ol>
Priority:	
Frequency of Use:	Frequently (every time the plan is generated)
Flow of Events:	<ol style="list-style-type: none"> <li>1. The user logs into the EatWellthy application and navigates to the "Diet Plan" section.</li> <li>2. The user selects the option to sync their diet plan with Google Calendar.</li> <li>3. The system prompts the user to sign in to their Google account, if they haven't already.</li> <li>4. The system requests the necessary permissions to access and manage the user's Google Calendar.</li> <li>5. The user grants permission, allowing the EatWellthy app to access their Google Calendar.</li> <li>6. The system retrieves the user's diet plan and begins syncing the data with the Google Calendar.</li> <li>7. For each meal or diet-related event in plan, system: <ul style="list-style-type: none"> <li>• Creates a corresponding event in Google Calendar.</li> <li>• Sets the event time and date based on meal schedule.</li> <li>• Includes details in the event description, such as the meal type, food items, and nutritional goals.</li> </ul> </li> </ol>

	<p>8. The system confirms the successful sync and notifies the user that their diet plan is now available in Google Calendar.</p> <p>9. The user can now view, edit, or receive reminders for their diet plan directly from Google Calendar.</p>
Alternative Flows:	<p>1. Authorization Failure:</p> <ul style="list-style-type: none"> <li>• If the user denies the authorization request, the system displays a message explaining that syncing cannot proceed without the necessary permissions.</li> <li>• The user is given the option to retry the authorization process.</li> </ul> <p>2. Sync Failure:</p> <ul style="list-style-type: none"> <li>• If the system encounters an issue while syncing (e.g., network issues, Google API errors), the system displays an error message.</li> <li>• The user is given options to retry the sync or to troubleshoot the issue.</li> </ul> <p>3. Partial Sync:</p> <ul style="list-style-type: none"> <li>• If only part of the diet plan syncs successfully (e.g., some events were not created), the system notifies the user of the partial success.</li> <li>• The system provides details on which events were not synced and offers options to retry syncing only those events.</li> </ul>
Exceptions:	<p>1. Network Issues:</p> <ul style="list-style-type: none"> <li>• If there is a network connectivity problem during the sync, the system alerts the user and pauses the syncing process until the connection is restored.</li> <li>• The user can manually retry the sync after resolving any network issues.</li> </ul> <p>2. Google Calendar Quota Limits:</p> <ul style="list-style-type: none"> <li>• If the user's Google Calendar has reached its event creation limit, the system notifies the user and cancels the sync process.</li> <li>• The user is advised to clear up space in their Google Calendar or contact Google support for further assistance.</li> </ul> <p>3. Calendar Event Conflicts:</p> <ul style="list-style-type: none"> <li>• If there are conflicting events in the Google Calendar during the sync (e.g., overlapping times with existing events), the system warns the user.</li> <li>• The user can choose to skip the conflicting events or manually resolve the conflicts in Google Calendar.</li> </ul>
Includes:	
Special Requirements:	The system should access and interact with Google calendar successfully.
Assumptions:	We assume that the user have a Google account and the system is able to access it.
Notes and Issues:	

## IV. For Functional Requirement #8 (Welloh Chatbot)

### U0401 Ask Nutritional Questions

Use Case ID:	U401		
Use Case Name:	Ask Nutritional Questions		
Created By:	Liu Xiaotao	Last Updated By:	Liu Xiaotao
Date Created:	2/9/2024	Date Last Updated:	2/9/2024

Actor:	Current User
Description:	This use case allows users to interact with the Welloh ChatBot to ask various nutritional questions. The Welloh ChatBot is powered by GPT-based technology, fine-tuned to provide accurate, relevant, and personalized nutritional information. Users can inquire about dietary advice, nutritional content of foods, meal suggestions, and other related topics. The chatbot is designed to deliver responses quickly and efficiently.
Preconditions:	<ol style="list-style-type: none"> <li>1. The user must be logged into the EatWellthy application.</li> <li>2. The Welloh ChatBot must be connected to the internet and have access to the latest nutritional databases and GPT-based model.</li> <li>3. The ChatBot's underlying GPT model must be optimised for performance, ensuring quick and relevant responses.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. The user receives an accurate and relevant response to their nutritional question</li> <li>2. The system logs the interaction for future reference and potential improvement of the ChatBot's responses.</li> <li>3. The user can save the response or add suggested meals directly to their meal plan within the app.</li> </ol>
Priority:	High
Frequency of Use:	As needed
Flow of Events:	<ol style="list-style-type: none"> <li>1. The user opens the EatWellthy application and navigates to the "Welloh ChatBot" feature.</li> <li>2. The user types a nutritional question into the ChatBot interface. Examples of questions include: <ul style="list-style-type: none"> <li>• "What are the health benefits of eating avocados?"</li> <li>• "How many calories are in a medium-sized apple?"</li> <li>• "Can you suggest a high-protein breakfast?"</li> </ul> </li> <li>3. The system receives the user's input and processes the question through the Welloh ChatBot's GPT-based model.</li> </ol>

	<ol style="list-style-type: none"> <li>4. The ChatBot analyzes the question, retrieving relevant information from its nutritional database and applying natural language processing to generate a clear and concise response.</li> <li>5. The system displays the response to the user within 2 seconds of receiving the question. The response may include: <ul style="list-style-type: none"> <li>• Nutritional information (e.g., calorie content, macronutrient breakdown)</li> <li>• Health benefits and potential risks of certain foods</li> <li>• Personalized meal suggestions based on the user's profile and dietary preferences</li> </ul> </li> <li>6. The user reviews the response and can either ask follow-up questions, save the response, or incorporate suggestions into their meal plan.</li> <li>7. The system logs the interaction for future reference, potentially improving future responses through machine learning.</li> </ol>
Alternative Flows:	<ol style="list-style-type: none"> <li>1. Unavailable Information: <ul style="list-style-type: none"> <li>• If the ChatBot cannot find relevant data for the user's query, it informs the user and offers to search for related information or suggest alternative questions.</li> <li>• The user can either modify their question or accept the ChatBot's suggestions.</li> </ul> </li> <li>2. Technical Issues: <ul style="list-style-type: none"> <li>• If the ChatBot experiences technical issues (e.g., connectivity problems, server downtime), it informs the user and suggests trying again later.</li> <li>• The system may offer offline alternatives, such as browsing the FAQ section or accessing preloaded nutritional guides.</li> </ul> </li> </ol>
Exceptions:	<ol style="list-style-type: none"> <li>1. Excessive Response Time: <ul style="list-style-type: none"> <li>• If the ChatBot takes longer than 2 seconds to generate a response, the system displays a message indicating that the response is delayed.</li> <li>• The user is given the option to wait or rephrase the question for a potentially faster response.</li> </ul> </li> </ol>
Includes:	
Special Requirements:	
Assumptions:	It assume the system can have a stable GPT service
Notes and Issues:	May need some prompt techniques for an accurate response.

## U0402 Receive Nutritional Alerts

Use Case ID:	U402		
Use Case Name:	Receive Nutritional Alerts		
Created By:	Liu Xiaotao	Last Updated By:	Liu Xiaotao
Date Created:	2/9/2024	Date Last Updated:	2/9/2024

Actor:	Current User
Description:	<p>This use case allows users to receive alerts and notifications related to their nutritional intake. These alerts may include reminders to log meals, notifications of nutritional goals being met or missed, and suggestions for adjustments based on their diet plan.</p>
Preconditions:	<ol style="list-style-type: none"> <li>1. The user logs into the EatWellthy application and navigates to the settings menu.</li> <li>2. The user ensures that notifications are enabled and that the necessary permissions are granted for the app to send alerts to their device.</li> <li>3. The user sets their preferences for receiving alerts, such as frequency (e.g., daily, weekly), type of alerts (e.g., meal reminders, goal tracking), and preferred times for receiving notifications.</li> <li>4. As the user logs meals or updates their diet plan, the system continuously monitors their nutritional intake and progress towards their goals.</li> <li>5. When the system detects a condition that triggers an alert (e.g., a meal is missed, a nutritional goal is met or missed), it generates a notification.</li> <li>6. The notification includes relevant information, such as: <ul style="list-style-type: none"> <li>• A reminder to log a meal if it hasn't been recorded within a certain time frame.</li> <li>• An alert that a nutritional goal has been met or is at risk of being missed, with suggestions on how to stay on track.</li> <li>• Tips or suggestions for adjusting the diet to improve nutritional balance.</li> </ul> </li> <li>7. The system sends the notification to the user's device, ensuring it arrives at the preferred time set by the user.</li> <li>8. The user receives the alert on their device, reviews the information, and takes appropriate action, such as logging a meal or adjusting their diet plan.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. Notification Settings Disabled: <ul style="list-style-type: none"> <li>• If the user has not enabled notifications, the system displays a message within the app, encouraging the user to enable notifications to receive important alerts.</li> </ul> </li> </ol>

	<ul style="list-style-type: none"> <li>• The user can navigate to the settings to enable notifications if desired.</li> </ul> <p>2. Missed Alerts:</p> <ul style="list-style-type: none"> <li>• If the user misses an alert (e.g., due to being offline), the system logs the alert and resends it when the user is back online or next opens the app.</li> </ul> <p>3. Custom Alert Frequency:</p> <ul style="list-style-type: none"> <li>• If the user sets a custom frequency for receiving alerts (e.g., only once a day), the system respects this setting and consolidates multiple alerts into a single notification.</li> </ul> <p>• The user receives a summary alert at their preferred time, covering all relevant information.</p>
Priority:	Middle
Frequency of Use:	As needed
Flow of Events:	<ol style="list-style-type: none"> <li>1. The user logs into the EatWellthy application and navigates to the settings menu.</li> <li>2. The user ensures that notifications are enabled and that the necessary permissions are granted for the app to send alerts to their device.</li> <li>3. The user sets their preferences for receiving alerts, such as frequency (e.g., daily, weekly), type of alerts (e.g., meal reminders, goal tracking), and preferred times for receiving notifications.</li> <li>4. As the user logs meals or updates their diet plan, the system continuously monitors their nutritional intake and progress towards their goals.</li> <li>5. When the system detects a condition that triggers an alert (e.g., a meal is missed, a nutritional goal is met or missed), it generates a notification.</li> <li>6. The notification includes relevant information, such as: <ul style="list-style-type: none"> <li>• A reminder to log a meal if it hasn't been recorded within a certain time frame.</li> <li>• An alert that a nutritional goal has been met or is at risk of being missed, with suggestions on how to stay on track.</li> <li>• Tips or suggestions for adjusting the diet to improve nutritional balance.</li> </ul> </li> <li>7. The system sends the notification to the user's device, ensuring it arrives at the preferred time set by the user.</li> <li>8. The user receives the alert on their device, reviews the information, and takes appropriate action, such as logging a meal or adjusting their diet plan.</li> </ol>
Alternative Flows:	<ol style="list-style-type: none"> <li>1. Notification Settings Disabled: <ul style="list-style-type: none"> <li>• If the user has not enabled notifications, the system displays a message within the app, encouraging the</li> </ul> </li> </ol>

	<p>user to enable notifications to receive important alerts.</p> <ul style="list-style-type: none"> <li>The user can navigate to the settings to enable notifications if desired.</li> </ul> <p>2. Missed Alerts:</p> <ul style="list-style-type: none"> <li>If the user misses an alert (e.g., due to being offline), the system logs the alert and resends it when the user is back online or next opens the app.</li> <li>The user can view missed alerts in a dedicated "Notification History" section within the app.</li> </ul> <p>3. Custom Alert Frequency:</p> <ul style="list-style-type: none"> <li>If the user sets a custom frequency for receiving alerts (e.g., only once a day), the system respects this setting and consolidates multiple alerts into a single notification.</li> <li>The user receives a summary alert at their preferred time, covering all relevant information</li> </ul>
Exceptions:	<ol style="list-style-type: none"> <li>Device Notification Issues: <ul style="list-style-type: none"> <li>If the device is unable to receive notifications due to technical issues (e.g., Do Not Disturb mode, network problems), the system logs the alert and attempts to resend it once the issue is resolved.</li> <li>The user can view a history of missed alerts within the app.</li> </ul> </li> <li>Nutritional Data Sync Delays: <ul style="list-style-type: none"> <li>If there is a delay in syncing the user's nutritional data (e.g., due to a slow network), the system may send alerts later than expected.</li> <li>The system prioritizes timely delivery and informs the user if there is a delay in processing their nutritional data.</li> </ul> </li> <li>Inaccurate Alerts: <ul style="list-style-type: none"> <li>If an alert is generated based on incorrect or incomplete data (e.g., user input errors), the system allows the user to dismiss or correct the alert.</li> <li>The system may prompt the user to review their logged meals or update their diet plan to ensure accuracy.</li> </ul> </li> </ol>
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	The alert should consider the users' feeling i.e. a gentle expression is needed

## U0403 Query Nutritional Information via Chatbot

Use Case ID:	U403		
Use Case Name:	Query Nutritional Information via Chatbot		
Created By:	Liu Xiaotao	Last Updated By:	Liu Xiaotao
Date Created:	2/9/2024	Date Last Updated:	2/9/2024

Actor:	Current User
Description:	This use case allows users to query specific nutritional information via the Welloh ChatBot. The ChatBot, powered by a GPT-based model optimized for enhanced performance, provides detailed nutritional data on various foods, ingredients, and meals. Users can inquire about calorie counts, macronutrient breakdowns, vitamin content, and more, receiving tailored responses based on their dietary preferences and needs.
Preconditions:	<ol style="list-style-type: none"> <li>1. The user must be logged into the EatWellthy application.</li> <li>2. The Welloh ChatBot must be operational, with access to the latest nutritional databases and the GPT-based model.</li> <li>3. The ChatBot's system must be up-to-date, ensuring that the nutritional information provided is accurate and reliable.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. The user receives precise nutritional information based on their query.</li> <li>2. The system logs the interaction for future reference, helping to improve the accuracy and relevance of future responses.</li> <li>3. The user has the option to save the queried information or add related items to their meal plan.</li> </ol>
Priority:	Medium
Frequency of Use:	As needed
Flow of Events:	<ol style="list-style-type: none"> <li>1. The user opens the EatWellthy application and navigates to the "Welloh ChatBot" feature.</li> <li>2. The user types a specific query regarding nutritional information into the ChatBot interface. Examples of queries include: <ul style="list-style-type: none"> <li>• "How many grams of protein are in a chicken breast?"</li> <li>• "What is the fat content of 100 grams of almonds?"</li> <li>• "How much vitamin C is in an orange?"</li> </ul> </li> <li>3. The system receives the user's query and processes it through the Welloh ChatBot's GPT-based model.</li> <li>4. The ChatBot analyzes the query, retrieves relevant nutritional data from its database, and generates a response.</li> </ol>

	<p>5. The system displays the nutritional information to the user within 2 seconds of receiving the query. The response may include:</p> <ul style="list-style-type: none"> <li>● Calorie content</li> <li>● Macronutrient breakdown (proteins, fats, carbohydrates)</li> <li>● Micronutrient details (vitamins, minerals)</li> <li>● Additional health-related information or advice</li> </ul> <p>6. The user reviews the information and can choose to:</p> <ul style="list-style-type: none"> <li>● Save the data for future reference</li> <li>● Add the item to their daily meal log</li> <li>● Ask follow-up questions for more details</li> </ul> <p>7. The system logs the interaction for future analysis and improvement of the ChatBot's responses.</p>
Alternative Flows:	<p>1. Unavailable Information:</p> <ul style="list-style-type: none"> <li>● If the ChatBot cannot find specific nutritional data for the user's query, it informs the user and offers alternatives (e.g., similar foods or items).</li> <li>● The user can choose to refine their query or select from the suggested alternatives.</li> </ul>
Exceptions:	<p>1. Excessive Response Time:</p> <ul style="list-style-type: none"> <li>● If the ChatBot takes longer than 2 seconds to generate a response, the system displays a message indicating that the response is delayed.</li> <li>● The user is given the option to wait or rephrase the question for a potentially faster response.</li> </ul>
Includes:	
Special Requirements:	
Assumptions:	It assume the system can have a stable GPT service
Notes and Issues:	May need some prompt techniques for an accurate response.

## V. For Functional Requirement # (Nutrition Reference Sheet)

### U0501 Search for Food Items

Use Case ID:	U0501		
Use Case Name:	Search for Food Items		
Created By:	Mehta Rishika	Last Updated By:	Mehta Rishika
Date Created:	29/8/2024	Date Last Updated:	29/8/2024

Actor:	Logged in User
Description:	This use case allows users who are logged into EatWellthy to be able to search for food items and their nutritional and caloric value. This can be done by passing the data of the meal plan into the EatWellthy system. The system will retrieve the desired information from the food database.
Preconditions:	<ol style="list-style-type: none"> <li>1. The user must be logged in.</li> <li>2. The user must provide valid input.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. Upon successful request, the system performs the following actions: <ul style="list-style-type: none"> <li>• The system retrieves the data from the food database</li> <li>• The system displays the caloric intake of the user based on the data it retrieves.</li> <li>• The user is equipped to access the app's features using their Google account credentials.</li> </ul> </li> </ol>
Priority:	
Frequency of Use:	
Flow of Events:	<ol style="list-style-type: none"> <li>1. The user will log in to EatWellthy using their credentials</li> <li>2. The user clicks on "Nutrition Calculator"</li> <li>3. The system displays a page where the user can enter the meal they want to check the caloric value for.</li> <li>4. If the user enters an invalid input, the system displays the message: "Input is invalid, please enter a correct value." and the user is redirected to "Nutrition Calculator" again.</li> <li>5. If the user inputs a valid input, the system fetches the necessary data from the database and gives the nutritional and caloric value of the given meal.</li> <li>6. The system displays the nutritional and caloric value.</li> <li>7. The user can check the nutritional content of other food items.</li> </ol>

Alternative Flows:	<p>1. Valid Input But No Data Found</p> <p>If the meal data entered is valid but not found in the database, the system displays: "No data available for the entered meal. Please try another meal."</p>
Exceptions:	<p>1. Database Unavailable</p> <ul style="list-style-type: none"> <li>• If the food database is down or unreachable, the system displays: "Service is currently unavailable. Please try again later."</li> </ul>
Includes:	
Special Requirements:	
Assumptions:	<p>It is assumed that User has already registered and possesses an account to access the features of the website.</p> <p>The food database is up-to-date and contains accurate nutritional information.</p>
Notes and Issues:	

## U0502 Update Food Database

Use Case ID:	U0502		
Use Case Name:	Update Food database		
Created By:	Mehta Rishika	Last Updated By:	Mehta Rishika
Date Created:	29/8/2024	Date Last Updated:	29/8/2024

Actor:	System Integration Service
Description:	The system updates the food database dynamically in response to updates from the API to ensure that the application has the most current and accurate food and nutritional information available.
Preconditions:	<p>The system must have an active and functional API connection for retrieving updates.</p> <p>The food database must be capable of being updated dynamically without manual intervention.</p> <p>The API provides data in a format compatible with the database schema.</p>
Postconditions:	<ol style="list-style-type: none"> <li>1. The food database is updated with the latest information from the API.</li> <li>2. Users can access and view the most current nutritional information through the application.</li> <li>3. Any changes are logged for tracking and auditing purposes.</li> </ol>
Priority:	
Frequency of Use:	As updates from the API are received.
Flow of Events:	<ol style="list-style-type: none"> <li>1. Trigger Update - The API sends a notification or data update to the system.</li> <li>2. Retrieve Data - The system connects to the API and retrieves the updated food data.</li> <li>3. Process Data - The system processes the data, performing validation and transformation as needed.</li> <li>4. Update Database - The system updates the food database with the new or modified information.</li> </ol>
Alternative Flows:	<ol style="list-style-type: none"> <li>1. Data Retrieval Issue <ul style="list-style-type: none"> <li>• If the API cannot be reached or data retrieval fails, the system logs the error and retries the operation or alerts the administrator.</li> </ul> </li> <li>2. Data Processing Error <ul style="list-style-type: none"> <li>• If the data is invalid or processing fails, the system logs the issue and may either retry or notify the administrator for manual review.</li> </ul> </li> </ol>
Exceptions:	<ol style="list-style-type: none"> <li>1. If the API is down or returns an error, the system handles the exception by implementing a retry mechanism or notifying support personnel.</li> <li>2. If an issue occurs during the database update (e.g., connectivity issues or schema mismatches), the</li> </ol>

	system logs the error and attempts to roll back changes if possible.
Includes:	
Special Requirements:	
Assumptions:	<ol style="list-style-type: none"><li>1. The API is reliable and delivers data in a timely and accurate manner.</li><li>2. The database schema is designed to accommodate dynamic updates and changes.</li></ol>
Notes and Issues:	

## U0503 Request store availability data

Use Case ID:	U0503		
Use Case Name:	Request Store Availability Data		
Created By:	Mehta Rishika	Last Updated By:	Mehta Rishika
Date Created:	29/8/2024	Date Last Updated:	29/8/2024

Actor:	System Integration Service / Web Crawler
Description:	The system requests and retrieves store availability data from various supermarkets in Singapore to provide users with up-to-date information on product availability and pricing. This data helps users make cost-effective food choices by comparing prices and availability across different stores.
Preconditions:	<ol style="list-style-type: none"> <li>1. The system must have access to a reliable web crawler or API that provides store availability data.</li> <li>2. The store availability data source must be operational and accessible.</li> <li>3. The system must be able to handle and parse data from multiple sources.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. Store availability data is retrieved and updated in the system's database.</li> <li>2. Users can view current availability and prices of food items from different stores.</li> <li>3. Any discrepancies or issues in data retrieval are logged and addressed.</li> </ol>
Priority:	
Frequency of Use:	Daily at 12 pm
Flow of Events:	<ol style="list-style-type: none"> <li>1. The system initiates a request for store availability data on a scheduled basis at 12 pm.</li> <li>2. The system uses a web crawler or API to fetch data from participating supermarkets.</li> <li>3. The system processes and parses the retrieved data, extracting relevant information such as product availability and pricing.</li> <li>4. The system updates its internal database with the new store availability data.</li> <li>5. Users can access the updated information through the application's interface.</li> </ol>
Alternative Flows:	<ol style="list-style-type: none"> <li>1. If the data source is unavailable or returns an error, the system logs the failure and may retry the request or notify the administrator.</li> <li>2. If the data format is incorrect or parsing fails, the system logs the error and may attempt to correct the issue or request data again.</li> </ol>

Exceptions:	1. If the store availability data source is down, the system handles the exception by notifying support and possibly using cached data if available.
Includes:	
Special Requirements:	
Assumptions:	<ol style="list-style-type: none"> <li>1. The web crawler or API providing store data is reliable and regularly updated.</li> <li>2. The system's database schema supports the integration of store availability and pricing data.</li> </ol>
Notes and Issues:	

## U0504 Manage User data

Use Case ID:	U0504		
Use Case Name:	Manage User data		
Created By:	Mehta Rishika	Last Updated By:	Mehta Rishika
Date Created:	29/8/2024	Date Last Updated:	29/8/2024

Actor:	System Service
Description:	The system autonomously manages user data within the EatWellthy application. This includes creating, updating, retrieving, and deleting user profiles based on user actions and system requirements. The system ensures that user data is accurately maintained and secure.
Preconditions:	<ol style="list-style-type: none"> <li>1. The system must have access to the user database.</li> <li>2. User profiles must exist in the database for updates or deletions.</li> <li>3. The system must ensure that data management actions comply with security and privacy regulations.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. User profiles are accurately created, updated, or deleted as required.</li> <li>2. The database reflects the most current user information.</li> <li>3. The integrity and confidentiality of user data are preserved.</li> </ol>
Priority:	
Frequency of Use:	Depends on user actions and needs
Flow of Events:	<ol style="list-style-type: none"> <li>1. The system detects a request to create, update, retrieve, or delete a user profile.</li> <li>2. The system verifies the user's identity and permissions automatically if applicable.</li> <li>3. Data Management Actions - <ul style="list-style-type: none"> <li>• Create Profile: The system adds a new user profile to the database based on provided information upon registering. The system displays "User successfully registered"</li> <li>• Update Profile: The system updates an existing user profile with new data, ensuring validation and correctness. The system displays "Profile updated successfully"</li> <li>• Retrieve Profile: The system retrieves and displays user profile data as requested.</li> <li>• Delete Profile: The system removes a user profile from the database based on the request.</li> </ul> </li> <li>4. The system ensures that the database is updated in real time to reflect the changes made.</li> </ol>

Alternative Flows:	<ol style="list-style-type: none"> <li>If there is an issue during the profile update process, the system handles the error by logging it and notifying the user or administrator a message, "Unable to update the profile, please try again".</li> <li>If the deletion fails, the system logs the error and informs the user.</li> </ol>
Exceptions:	<ol style="list-style-type: none"> <li>The system denies access to data management functions if the request lacks proper authorization and logs the attempt.</li> </ol>
Includes:	<ol style="list-style-type: none"> <li>Authentication and authorization mechanisms.</li> <li>Data validation processes.</li> <li>Error handling and logging functions.</li> </ol>
Special Requirements:	
Assumptions:	<ol style="list-style-type: none"> <li>The system has the necessary access and permissions to manage user data.</li> <li>Data management functions are integrated with the system's security and privacy protocols.</li> </ol>
Notes and Issues:	

## U0505 Backup and Restore

Use Case ID:	U0505		
Use Case Name:	Backup and Restore		
Created By:	Mehta Rishika	Last Updated By:	Mehta Rishika
Date Created:	29/8/2024	Date Last Updated:	29/8/2024

Actor:	System server
Description:	The system manages the backup and restoration of data to ensure data integrity and availability. This includes creating periodic backups of user data and other critical information, as well as restoring data from backups when necessary. This process helps in data recovery in case of system failures, data corruption, or accidental loss.
Preconditions:	<ol style="list-style-type: none"> <li>1. The system must have access to the database and backup storage.</li> <li>2. Backup policies and schedules must be defined and configured.</li> <li>3. Data to be backed up must be in a consistent state.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. Backup files are created and stored securely according to the backup schedule.</li> <li>2. Data is restored accurately from backup files as needed.</li> <li>3. The system ensures the integrity and availability of user data.</li> </ol>
Priority:	High
Frequency of Use:	Regularly scheduled (weekly) for data recovery
Flow of Events:	<ol style="list-style-type: none"> <li>1. The system triggers a backup process according to the predefined schedule or upon request.</li> <li>2. The system collects and compiles user data and other critical information from the database.</li> <li>3. The data is compressed and encrypted if necessary.</li> <li>4. The system stores the backup in a secure, designated storage location.</li> <li>5. Confirmation by updating the logs.</li> </ol>
Alternative Flows:	<ol style="list-style-type: none"> <li>1. If the backup process fails, the system logs the error and notifies the system administrator. The system may attempt to retry the backup or alert the administrator to take corrective actions.</li> </ol>
Exceptions:	<ol style="list-style-type: none"> <li>1. If there is not enough storage space for backups, the system notifies the administrator and may halt the backup process until space is freed up.</li> <li>2. If the data in the backup file is corrupted, the system logs the issue and may attempt to use an alternative backup or pop an error.</li> </ol>
Includes:	
Special Requirements:	

Assumptions:	<ol style="list-style-type: none"><li>1. The backup and restore mechanisms are properly configured and tested.</li><li>2. Backup storage is secure and accessible to the system.</li></ol>
Notes and Issues:	

## VI. For Functional Requirement #7 (Meal Planner)

### U0601 View Dietary History

Use Case ID:	U0601		
Use Case Name:	View Dietary History		
Created By:	Zhang Yichi	Last Updated By:	2024/08/30
Date Created:	2024/08/30	Date Last Updated:	2024/08/30

Actor:	User (login)
Description:	User requests to view their dietary history in a structured format that shows the meals consumed over a specific period.
Preconditions:	<ol style="list-style-type: none"> <li>1. User must be logged in</li> <li>2. The dietary history must be recorded and stored in the system.</li> <li>3. The system should have access to the user's dietary records.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. The system displays the dietary history in a readable format.</li> <li>2. The dietary data reflects the user's consumption accurately.</li> <li>3. The dietary history is presented without any errors.</li> </ol>
Priority:	Medium
Frequency of Use:	Daily (Users typically check their dietary history once a day or week.)
Flow of Events:	<ol style="list-style-type: none"> <li>1. User logs into the system.</li> <li>2. User navigates to the dietary history section.</li> <li>3. User selects the desired time frame for viewing dietary history.</li> <li>4. The system retrieves dietary records from the database.</li> <li>5. The system displays the dietary history on the screen.</li> </ol>
Alternative Flows:	<ol style="list-style-type: none"> <li>1. If the system fails to retrieve dietary records, an error message is displayed to the user.</li> <li>2. If no records are found for the selected time frame, the system informs the user with a message.</li> </ol>
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	The records are stored fully without glitch
Notes and Issues:	

## U0602 Analyze Nutritional Intake Trends

Use Case ID:	U0602		
Use Case Name:	Analyse Nutritional Intake Trends		
Created By:	Zhang Yichi	Last Updated By:	08/30/2024
Date Created:	08/30/2024	Date Last Updated:	08/30/2024

Actor:	User
Description:	User analyzes nutritional intake trends based on their historical dietary data.
Preconditions:	<ol style="list-style-type: none"> <li>1. User must be logged in.</li> <li>2. The system must have access to historical dietary data.</li> <li>3. Nutritional analysis algorithms must be implemented in the system.</li> </ol>
Postconditions:	<p>Related data sum up correctly</p> <ol style="list-style-type: none"> <li>1. The nutrition data of the chosen items would be draw from nutrition sheets correctly</li> <li>2. The system provides a visual and statistical analysis of the user's nutritional intake trends.</li> <li>3. The trends reflect the user's consumption habits accurately.</li> </ol>
Priority:	High
Frequency of Use:	Weekly (Users often review their nutritional trends weekly.)
Flow of Events:	<ol style="list-style-type: none"> <li>1. User logs into the system.</li> <li>2. User navigates to the nutritional analysis section.</li> <li>3. User selects the desired parameters for analysis (e.g., time frame, specific nutrients).</li> <li>4. The system retrieves relevant dietary data from the database.</li> <li>5. The system processes the data and generates an analysis report.</li> <li>6. The system displays the nutritional intake trends on the screen.</li> </ol>
Alternative Flows:	<ol style="list-style-type: none"> <li>1. If the system fails to generate an analysis report, an error message is displayed to the user.</li> <li>2. If no data is available for the selected parameters, the system informs the user accordingly.</li> </ol>
Exceptions:	<ol style="list-style-type: none"> <li>1. Data inconsistencies or missing data may affect the accuracy of the analysis.</li> </ol>
Includes:	
Special Requirements:	
Assumptions:	The record are stored fully without glitch
Notes and Issues:	

## U0603 Generate User Reports

Use Case ID:	U0603		
Use Case Name:	Generate User Reports		
Created By:	Zhang Yichi	Last Updated By:	08/30/2024
Date Created:	08/30/2024	Date Last Updated:	08/30/2024

Actor:	User
Description:	Users generate detailed reports on their dietary and nutritional intake.
Preconditions:	<ol style="list-style-type: none"><li>1. User must be logged in.</li><li>2. The system should have access to all relevant user data.</li><li>3. Report generation functionalities must be implemented.</li></ol>
Postconditions:	<ol style="list-style-type: none"><li>1. The system generates a comprehensive report based on user data.</li><li>2. The report accurately reflects the user's dietary and nutritional intake.</li></ol>
Priority:	Medium
Frequency of Use:	Monthly (Users typically generate reports on a monthly basis.)
Flow of Events:	<ol style="list-style-type: none"><li>1. User logs into the system.</li><li>2. User navigates to the report generation section.</li><li>3. User selects the desired report type and parameters.</li><li>4. The system retrieves necessary data from database.</li><li>5. The system generates the report and displays it on the screen or allows download.</li></ol>
Alternative Flows:	<ol style="list-style-type: none"><li>1. If the system fails to generate the report, an error message is displayed to the user.</li><li>2. If no data is available for the selected parameters, the system informs the user.</li></ol>
Exceptions:	<ol style="list-style-type: none"><li>1. Data inconsistencies or missing data may affect the accuracy of the analysis.</li></ol>
Includes:	<ol style="list-style-type: none"><li>1. System errors during data retrieval or report generation.</li></ol>
Special Requirements:	
Assumptions:	The records are stored fully without glitch
Notes and Issues:	

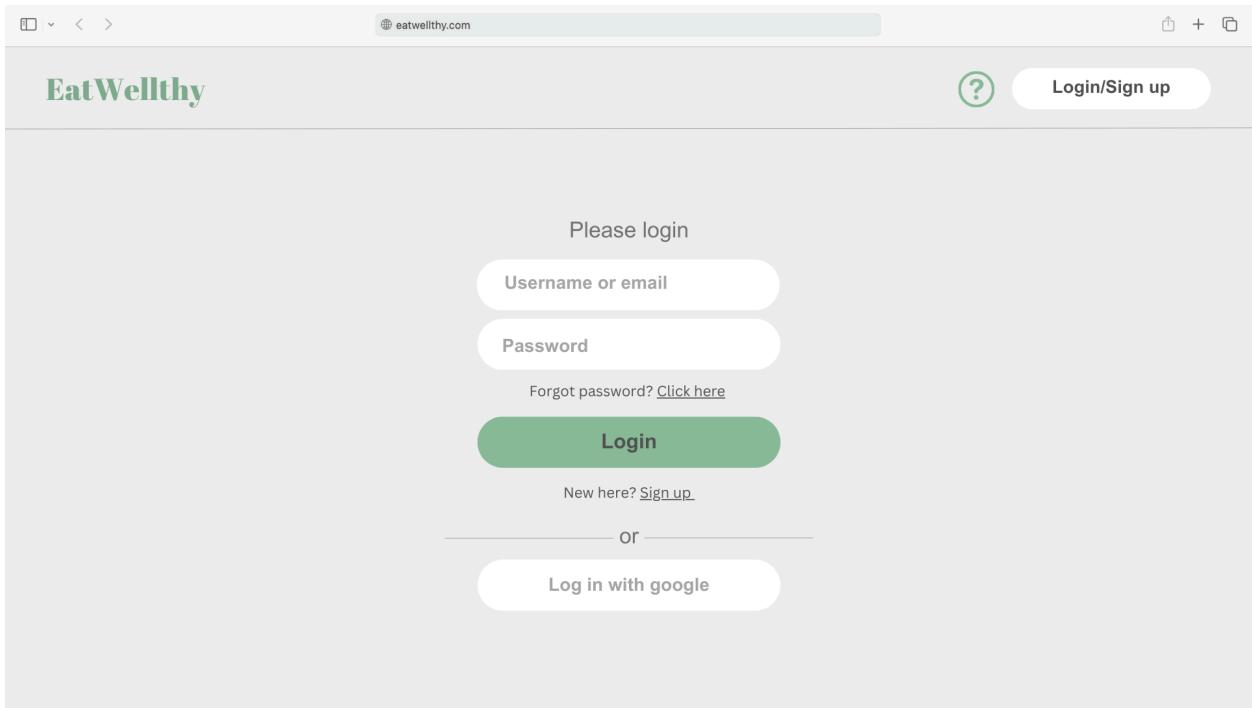
## U0604 Export Nutritional Data

Use Case ID:	U0604		
Use Case Name:	Export Nutritional Data		
Created By:	Zhang Yichi	Last Updated By:	08/30/2024
Date Created:	08/30/2024	Date Last Updated:	08/30/2024

Actor:	User
Description:	Users export their nutritional data for external analysis or record-keeping.
Preconditions:	<ol style="list-style-type: none"> <li>1. User must be logged in.</li> <li>2. The system should have access to the user's nutritional data.</li> <li>3. Export functionality must be enabled in the system.</li> </ol>
Postconditions:	<ol style="list-style-type: none"> <li>1. The system exports the nutritional data in the desired format.</li> <li>2. The exported data is accurate and complete.</li> </ol>
Priority:	Low
Frequency of Use:	Occasionally (Users export data when needed for external purposes.)
Flow of Events:	<ol style="list-style-type: none"> <li>1. User logs into the system.</li> <li>2. User navigates to the data export section.</li> <li>3. User selects the desired data range and format for export.</li> <li>4. The system retrieves the relevant data from the database.</li> <li>5. The system exports the data in the selected format and provides a download link.</li> </ol>
Alternative Flows:	<ol style="list-style-type: none"> <li>1. If the system fails to export the data, an error message is displayed to the user.</li> <li>2. If no data is available for the selected range, the system informs the user.</li> </ol>
Exceptions:	N
Includes:	
Special Requirements:	
Assumptions:	The records are stored fully without glitch
Notes and Issues:	

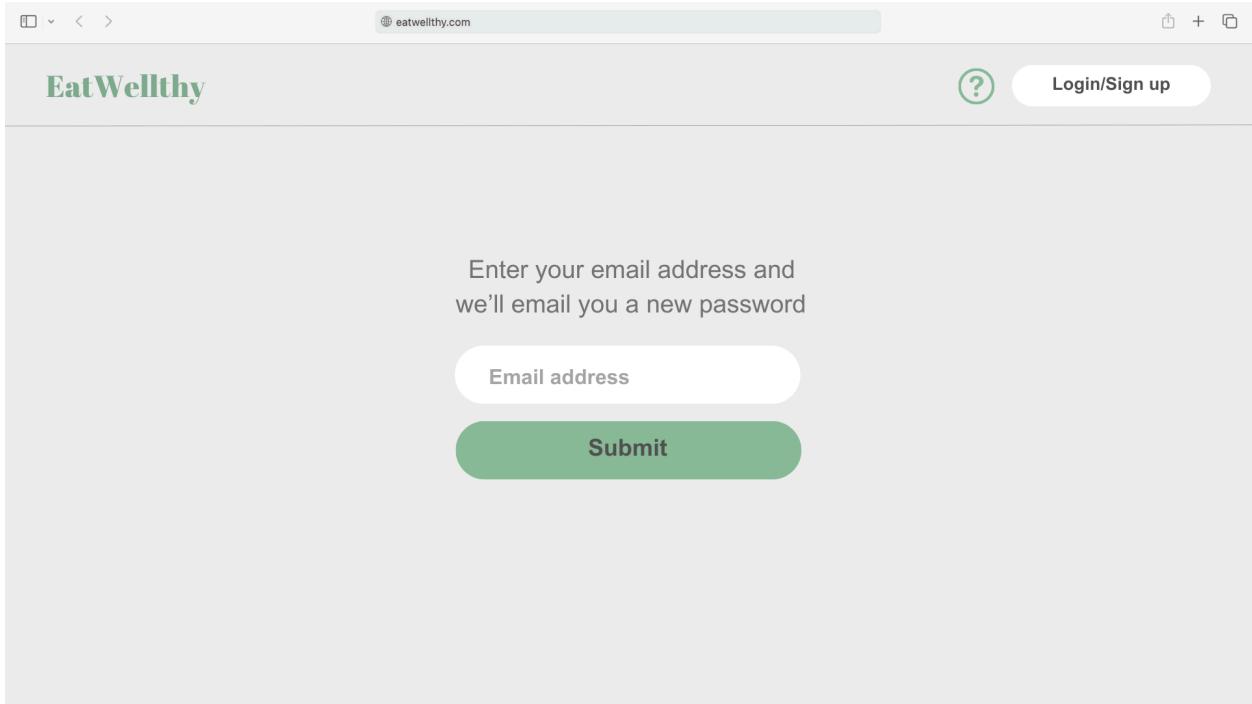
# UI Design

## Login Page



The screenshot shows a web browser window for the EatWellthy website. The URL bar displays "eatwellthy.com". The page header includes the "EatWellthy" logo, a help icon, and a "Login/Sign up" button. The main content area has a light gray background. At the top center, the text "Please login" is displayed. Below it are two input fields: "Username or email" and "Password". Underneath the password field is a link "Forgot password? [Click here](#)". A large green rounded rectangular button labeled "Login" is centered below the fields. To the right of the "Login" button is a link "New here? [Sign up](#)". A horizontal line with the word "or" in the center separates this from a "Log in with google" button.

## Forgot Password Page



The screenshot shows a web browser window for the EatWellthy website. The URL bar displays "eatwellthy.com". The page header includes the "EatWellthy" logo, a help icon, and a "Login/Sign up" button. The main content area has a light gray background. Centered text reads "Enter your email address and we'll email you a new password". Below this is an input field labeled "Email address". A large green rounded rectangular button labeled "Submit" is positioned below the input field.

## Sign Up Page

The screenshot shows the sign-up page for EatWellthy. At the top right, there is a question mark icon and a "Login/Sign up" button. The main heading "Sign up" is centered above five input fields arranged in two rows: "First Name" and "Last Name" in the top row, and "Email Address" and "Phone Number" in the bottom row. Below these is a "Password" field and a "Re-enter Password" field. A large green "Register" button is centered at the bottom. At the very bottom, a link says "Already have an account? [Login here](#)".

## Verification Page

The screenshot shows the verification page for EatWellthy. At the top right, there is a question mark icon and a "Login/Sign up" button. The main instruction "Enter the OTP we have sent to your email" is centered above an "OTP" input field and a large green "Verify" button. At the bottom, a link says "Didn't receive? [Resend OTP](#)".

## Dashboard Page

The dashboard page features a sidebar on the left with a user icon and a list of navigation links: Dashboard (selected), Profile, Nutrition Calculator, Diet Planner, Daily Price List, Welloh, Calendar, FAQs, and Location.

Main content area:

- Daily nutritional intake:** Three donut charts showing Carbohydrates (blue), Fat (red), and Protein (yellow).
- Progress Tracking:** Shows current weight (60kg), goal (55kg), and weight to go (2.5kg). It also indicates a weight loss of 0.2kg in the previous week.
- Diet suggestions:** A card for 30.8.2024 listing breakfast, lunch, and dinner meals.
- Caloric intake chart:** A pie chart showing 1350kcal / 1800kcal.

## Profile Page

The profile page features a sidebar on the left with a user icon and a list of navigation links: Dashboard (selected), Profile (selected), Nutrition Calculator, Diet Planner, Daily Price List, Welloh, Calendar, FAQs, and Location.

Main content area:

- Profile:** Fields for Age, Height (cm), Weight (kg), and Daily Budget (\$). An "Edit Profile" button is located in the top right corner.
- Dietary Preferences:** A large input field for dietary preferences.
- Change password:** Fields for current password and re-entered password.
- Allergies (if any):** A large input field for allergies.

## Nutrition Calculator Page

The screenshot shows the EatWellthy website interface. On the left is a sidebar with a user icon and a list of navigation items: Dashboard, Profile, Nutrition Calculator (which is highlighted), Diet Planner, Daily Price List, Welloh, Calendar, FAQs, and Location. The main content area has a title "Nutrition calculator". It contains three input fields: "Name of food item(s):", "Portion size/quantity:", and "Additional details (if any:)". A green button at the bottom right says "Calculate calories and nutritional content". The top right corner features a help icon, a user profile icon, and a sign-out icon.

## Diet Planner Page

The screenshot shows the EatWellthy website interface. The sidebar on the left is identical to the Nutrition Calculator page, with the "Diet Planner" item also being highlighted. The main content area has a title "Diet Planner" and displays a meal plan for two days. A table is used to show the meals for Breakfast, Lunch, and Dinner for Monday and Tuesday.

	Breakfast (8.30am to 9.30am)	Lunch (12.30pm to 1.30pm)	Dinner (6pm to 7pm)
Monday	Oatmeal with fruit and nuts	Tofu and mushroom noodle soup	Stir-fry beef and broccoli with cauliflower rice
Tuesday	Chinese style savoury oat pancakes	Steamed fish with ginger and soy sauce	Stuffed bell peppers with shrimp and rice

## Daily Price List Page

The screenshot shows the EatWellthy platform's daily price list feature. On the left, a sidebar menu includes options like Dashboard, Profile, Nutrition Calculator, Diet Planner, Daily Price List (which is selected and highlighted in blue), Welloh, Calendar, FAQs, and Location. The main content area displays three separate price lists:

- Fairprice:** Shows various grocery items with their prices, such as \$1.42 for Fresh Tomatoes and \$1.81 for Chilled Chilli Peppers.
- Cold Storage:** Shows items like \$2.40 for Fresh Baby Spinach and \$1.80 for Fresh Broccoli Florets.
- Sheng Siong:** Shows items like \$2.40 for Fresh Baby Spinach and \$1.80 for Fresh Broccoli Florets.

## Welloh Chatbot Page

The screenshot shows the EatWellthy platform's Welloh chatbot page. The sidebar menu includes Dashboard, Profile, Nutrition Calculator, Diet Planner, Daily Price List, Welloh (selected and highlighted in blue), Calendar, FAQs, and Location. The main content area features the Welloh chatbot and a Q&A section:

**Q: What are the essential nutrients I should include in my diet?**

**A:** Essential nutrients are crucial for maintaining overall health and supporting bodily functions. Here's a breakdown of the key nutrients you should include in your diet:

- Proteins:**
  - Function: Build and repair tissues, make enzymes and hormones, and support immune function.
  - Sources: Lean meats, poultry, fish, eggs, dairy products, legumes, nuts, seeds, and soy products.
- Carbohydrates:**
  - Function: Provide energy, especially for brain function and physical activity.
  - Sources: Whole grains (brown rice, oats, whole wheat), fruits, vegetables, legumes, and tubers (sweet potatoes, potatoes).
- Fats:**
  - Function: Provide long-term energy, support cell structure, and aid in the absorption of fat-soluble vitamins.
  - Sources: Healthy fats such as avocados, nuts, seeds, olive oil, and fatty fish (salmon, mackerel). Limit saturated fats and avoid trans fats.
- Vitamins:**
  - Vitamin A: Supports vision, immune function, and skin health. Found in carrots, sweet potatoes, and leafy greens.
  - Vitamin C: Boosts the immune system and helps with the absorption of iron. Found in citrus fruits, bell peppers, and strawberries.
  - Vitamin D: Adds to calcium absorption and bone health. Found in fatty fish, fortified dairy products, and exposure to sunlight.
  - Vitamin E: Acts as an antioxidant and protects cells from damage. Found in nuts, seeds, and green leafy vegetables.
  - Vitamin K: Important for blood clotting and bone health. Found in leafy greens like kale and sprouts.
- Minerals:**
  - Calcium: Essential for bone health and muscle function. Found in dairy products, leafy greens, and fortified plant-based milks.
  - Iron: Crucial for oxygen transport in the blood. Found in red meat, poultry, legumes, and fortified cereals.
  - Magnesium: Supports muscle and nerve function, and helps regulate blood sugar levels. Found in nuts, seeds, and whole grains.
  - Potassium: Helps with fluid balance, nerve function, and muscle contractions. Found in bananas, oranges, and potatoes.
- Water:**
  - Function: Vital for nearly every bodily function, including digestion, temperature regulation, and joint lubrication.
  - Sources: Water, herbal teas, fruits, and vegetables.
- Fiber:**
  - Function: Aids in digestion and helps maintain a healthy weight by promoting satiety.
  - Sources: Whole grains, fruits, vegetables, legumes, and nuts.

Each nutrient plays a specific role in maintaining health, so a balanced diet that includes a variety of foods from different food groups is essential for overall well-being. If you have specific health conditions or dietary needs, you might want to consult a healthcare provider or nutritionist for personalized advice.

## Google Calendar Page

The screenshot shows the EatWellthy app interface with the "Google Calendar" section highlighted. On the left, there's a sidebar with a user profile icon and links to Dashboard, Profile, Nutrition Calculator, Diet Planner, Daily Price List, Welloh, Calendar (which is selected and highlighted in green), FAQs, and Location. The main area is titled "Google Calendar" and displays a weekly calendar view for August 2024. A red arrow points from the "FAQs" link in the sidebar to the "Breakfast" event on Saturday, August 31, which is shown in a detailed view with a meal icon.

## FAQ Page

The screenshot shows the EatWellthy app interface with the "FAQs" section highlighted in the sidebar. The sidebar includes links to Dashboard, Profile, Nutrition Calculator, Diet Planner, Daily Price List, Welloh, Calendar, FAQs (selected and highlighted in green), and Location. The main content area is titled "General FAQs" and lists two questions with answers:

- How do I reset my password?**  
To reset your password, simply navigate to “Profile” and you will be able to change your password there.
- What if I have food allergies? How does the app handle this?**  
Under “Profile”, you can specify any allergies you have to certain foods. Our app uses this information to plan and personalise your meal, ensuring that all suggested meals do not contain your listed allergens.

## Location Page

The screenshot shows the EatWellthy website interface. At the top, there is a header bar with a search icon, a back arrow, a forward arrow, and a URL field containing "eatwellthy.com". On the right side of the header are icons for help, user profile, and sign-out.

The main content area has a title "Grocery stores near me" above a map. The map displays several red markers indicating the locations of grocery stores, such as "FairPrice Jurong West Blk 502", "FairPrice Jurong Taman Jurong", "FairPrice Jurong Gateway Road", "FairPrice Xpress", "FairPrice Finest Le Quest", "FairPrice Jurong East St 24", "FairPrice Teban Garden Rd", and "Boon Lay Complex". The map also shows major roads like "TENGAH", "NPE", and "Boon Lay Way". A "Google" logo is visible at the bottom left of the map.

To the left of the map is a sidebar titled "USER" which contains a user profile icon. Below the title are several menu items with icons: Dashboard (bar chart), Profile (person), Nutrition Calculator (apple), Diet Planner (list), Daily Price List (tag), Welloh (chat bubbles), Calendar (calendar), FAQs (question mark), and Location (location pin). The "Location" item is highlighted with a green rounded rectangle.