Use Cases

for

EatWellthy

Version 2.0 approved

Prepared by LIU XIAOTAO,
LOW JO YI, NICOLE,
MAHI PANDEY
MEHTA RISHIKA,
ZHANG YICHI
ZHAO QIXIAN

TEAM 31, SDDA, NANYANG TECHNOLOGICAL UNIVERSITY

<2024-11-07>

Revision History

Name	Date	Reason For Changes	Version
Mahi	9/11/24	Template Upload & Initial Write Up	1.0
Mahi	9/11/24	First Draft	1.1
Mahi	9/11/24	Final Draft	2.0

Use Case Template

Use Case ID:			
Use Case Name:			
Created By:		Last Updated By:	
Date Created:		Date Last Updated:	

Actor:	
Description:	
Preconditions:	
Postconditions:	
Priority:	
Frequency of Use:	
Flow of Events:	
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

1. Account Management

1.1. 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:	09/11/24

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 Name, Email, and Password. New Users will be assigned a unique UserID.
Preconditions:	1. Email address must be valid and user must have access to it. 2. Passwords must be 6 characters in length. 3. The supplied Email must not be associated with an existing user account.
Postconditions:	1. Upon a successful registration, the system seamlessly executes the following actions: a. Instantly dispatches a 6 digit code to the user's provided Email address, ensuring account verification. b. Asks for the 6 digit code 2. Once Account is verified, redirection to dashboard. 3. The user is assigned a unique UserID. 4. The user is now equipped to access the app's features, using their newly created Username and Password, or their Email and Password. 5. The user's UserID, email, Username and encrypted Password are stored in the database.
Priority:	High
Frequency of Use:	One-off
Flow of Events:	1. The new user opens EatWellthy and selects the "New User Register" option. 2. The system presents the user with a registration form, soliciting the following details: • Name • Email

	• Password 3. The user fills in the necessary information and clicks “Register”. 4. Upon a successful registration, the system seamlessly executes the following actions: a. Instantly dispatches a 6-digit code to the user's provided Email address, ensuring account verification. b. Asks for the 6-digit code 5. Once Account is verified, redirection to dashboard. • If any precondition is not met, the system displays precise error messages, guiding the user to correct their input. • If the input is valid, the registration process advances, and the user is directed to email verification.
Alternative Flows:	1. Validation Errors • If user inputs don't meet preconditions (e.g., invalid password, duplicate Username, existing Email), the system displays error messages for correction. • The user corrects input and resubmits for validation. • The process continues when all input is valid. 2. Password Reset • Users can initiate a password reset if forgotten or facing login issues. • The system sends a temporary password to the user's email. • User can use the temporary password to log in and then change their password under Profile section. 3. Confirmation Link Resend: • Users can request a resend of the confirmation code every 60 seconds. • The system sends a new confirmation code to their Email.
Exceptions:	1. Email Not Received: a. If the user does not receive the confirmation code, they may request a resend. If the issue persists, the system advises the user to check spam/junk folders or contact support.
Includes:	1. Email & Password Verification

	2. Display Success/Error Registration
Special Requirements:	1. The system must ensure that passwords are stored securely using encryption. 2. The system should handle email delivery promptly and securely.
Assumptions:	1. It is assumed that the user has a valid and accessible email address. 2. It is assumed that the user understands and complies with the password requirements.
Notes and Issues:	

1.2. **U0102 User Login via Email**

Use Case ID:	U0102		
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:	1. The user must have a valid registered account. 2. The user must provide a valid email and password.
Postconditions:	1. The user is successfully logged into the system. 2. The user's session is authenticated and maintained 3. The user is redirected to the EatWellthy main page
Priority:	High
Frequency of Use:	Every time user wishes to login, Multiple Times
Flow of Events:	1. The user opens the EatWellthy app and selects the Login option. 2. The system presents the login form, requesting Email and Password. 3. The user enters their Email and Password and clicks "Login" 4. The system validates the credentials. • If valid, the system logs in the user and redirects them to the main page.

	• If invalid, the system displays an error message.
Alternative Flows:	1. Invalid Credentials • The system informs the user of incorrect Email or Password and prompts re-entry. 2. Password Reset • If the user forgets their password, they can select “Forgot Password” to initiate a reset process.
Exceptions:	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.
Includes:	1. Password Verification 2. Display Success/Error Login
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:	

1.3. **U0103 User Login via Google Account**

Use Case ID:	U0103		
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:	09/11/2024

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:	1. The user must have a Google account. 2. The Google account must be linked to an EatWellthy user profile.
Postconditions:	1. The user is successfully logged into the system. 2. The user session is authenticated and maintained. 3. The user is redirected to the EatWellthy main page.
Priority:	Medium

Frequency of Use:	Every time user wishes to login, Multiple Times
Flow of Events:	1. The user selects “Login with Google”. 2. System redirects the user to Google’s login page. 3. The user enters their Google credentials and authorises access. 4. System retrieves the user’s Google Account. • If valid, the user is logged in and redirected to the main page. • If invalid, an error message is displayed.
Alternative Flows:	1. Google Authorization Failure • If the user fails to log into their Google Account, the system displays an error message. • The user can retry or choose another login method
Exceptions:	1. If Google API is unavailable, the user is informed and asked to try again later or use email login method.
Includes:	1. Google Authorisation 2. Display Success/Error Login
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:	

1.4. **U0104 Reset Forgotten Password**

Use Case ID:	U0104		
Use Case Name:	Reset Forgotten Password		
Created By:	Mahi Pandey	Last Updated By:	Mahi Pandey
Date Created:	02/09/2024	Date Last Updated:	09/11/2024

Actor:	Registered User
Description:	This use case enables users who have forgotten their password to reset it using their registered email address.
Preconditions:	1. The user must have a registered account with EatWellthy. 2. The user must have access to the registered email address.
Postconditions:	1. The user successfully resets their password.

	2. The user is prompted to log in with the new password.
Priority:	Medium
Frequency of Use:	Infrequent, as needed
Flow of Events:	1. The user selects "Forgot Password" on the login page. 2. The system prompts the user to enter their registered email. 3. The user enters their email and clicks "Submit". 4. The system verifies the email and sends a temporary password. 5. The user receives the email and uses the temporary password to log into their account. 6. Under Profile, the user can now update their password.
Alternative Flows:	1. Invalid Email • If the entered email is not registered, the system informs the user and prompts for re-entry.
Exceptions:	If the system cannot send the reset email, the user is informed of a delay and asked to try again later.
Includes:	1. Password Reset
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.

1.5. **U0105 Update Password**

Use Case ID:	U0105		
Use Case Name:	Change Account Password		
Created By:	Mahi Pandey	Last Updated By:	Mahi Pandey
Date Created:	02/09/2024	Date Last Updated:	09/11/2024

Actor:	Registered User
Description:	This use case allows registered users to change their account password while logged into the system.
Preconditions:	1. The user must be logged into their account. 2. The user must know their current password.
Postconditions:	1. The user’s password is successfully changed.

	2. The system logs the user out and prompts them to log in again with the new password.
Priority:	Medium
Frequency of Use:	Infrequent, as needed
Flow of Events:	1. The user navigates to the "Profile" page. 2. The system presents the option to change password. 3. The user enters the new password, then confirms the new password. 4. The system validates the current password and checks the new password against security requirements. 5. If valid, the system updates the password and logs the user out. 6. The user is prompted to log in with new password.
Alternative Flows:	1. Password Strength Failure • If the new password does not meet security requirements, the system prompts the user to enter a stronger password. 2. Passwords Do Not Match • If the two passwords do not match each other, an error message is displayed.
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:	1. Verify Original Password
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.

1.6. **U0106 Delete User Account**

Use Case ID:	U0106		
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:	1. The user must be logged into EatWellthy account. 2. The user must confirm their intention to permanently delete the account. 3. The system must inform user of the consequences of deletion, including the irretrievable loss of data.
Postconditions:	1. The user's account and all associated data are permanently removed from the system. 2. The user is logged out and cannot recover the account or data.
Priority:	Medium
Frequency of Use:	Infrequent, as needed
Flow of Events:	1. The user navigates to "Profile" and selects the "Delete Account" option. 2. The system prompts the user to confirm their decision to permanently delete the account. 3. The user confirms the deletion by providing their password for security verification. 4. The system warns the user about the permanent nature of account deletion and the loss of all data. 5. The user confirms the deletion. 6. The system permanently removes the user's account and all associated data from the database. 7. The user is logged out and redirected to the homepage with a confirmation of account deletion.
Alternative Flows:	1. Cancellation of Deletion • 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.
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:	1. The system must ensure that the deletion process is secure and irreversible. 2. The system should comply with relevant data protection regulations to avoid legal disputes.
Assumptions:	It is assumed that the user understands the implications of account deletion, including the permanent loss of access and data.
Notes and Issues:	

2. Profile

2.1. **U0201 Update Basic Profile Information**

Use Case ID:	U0201		
Use Case Name:	Update Basic 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 view and update their profile information within the EatWellthy app, including personal details, dietary preferences and activity history.
Preconditions:	1. The user must be logged into their EatWellthy account.
Postconditions:	1. The user's profile information is displayed on the screen. 2. Dashboard is personalised based on the given information
Priority:	High
Frequency of Use:	Occasional
Flow of Events:	1. The user selects "Profile" from the dashboard menu. 2. The user manually enters the following information: a. Name b. Gender c. Age d. Height (cm) e. Current Weight (kg) f. Target Weight (kg) g. Daily Budget (SGD) h. Activity Level i. Dietary Preferences j. Allergies 3. The user presses the "Update Profile" button and system displays a success message. 4. Data is stored in the database 5. Dashboard is updated accordingly, and data is also uploaded to WellOh.

Alternative Flows:	1. Invalid Input a. If any inputs are invalid, an error message is displayed, and the user is prompted to re-enter the field.
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:	
Special Requirements:	
Assumptions:	1. It is assumed that the user has valid profile information that they want updated.
Notes and Issues:	

2.2. **U0202 Manage Dietary Preferences**

Use Case ID:	U0202		
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:	1. The user must be logged into their EatWellthy account.
Postconditions:	1. The user's dietary preferences are successfully updated in the system.
Priority:	High
Frequency of Use:	Occasional, as needed
Flow of Events:	1. The user navigates to "Dietary Preferences" from their profile. 2. The system displays the current dietary preferences. 3. The user modifies their dietary preferences and allergies and clicks "Update". 4. The system updates the preferences in the database and confirms the update.
Alternative Flows:	1. Invalid Preference

	• If any preference selection is invalid, the system prompts the user to correct it before saving.
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:	1. Change Confirmation
Special Requirements:	Allergies must be separated by commas.
Assumptions:	1. It is assumed that the user has dietary preferences that they wish to manage.
Notes and Issues:	

3. Dashboard

3.1. **U0301 Using the Nutritional Information Finder**

Use Case ID:	U0301		
Use Case Name:	Using the Nutritional Information Finder		
Created By:	Mehta Rishika	Last Updated By:	Mahi Pandey
Date Created:	29/8/2024	Date Last Updated:	9/11/2024

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 name of the item through the Nutritionix API da. The system will retrieve the desired information from the food database.
Preconditions:	1. The user must be logged in. 2. The user must provide valid input.
Postconditions:	1. Upon successful request, the system performs the following actions: • The system retrieves the data from the food database • The system displays the caloric intake of the user based on the data it retrieves. • The user is equipped to access the app's features using their Google account credentials.
Priority:	Low
Frequency of Use:	As needed
Flow of Events:	1. The user will log in to EatWellthy using their credentials. 2. In the dashboard, the user navigates to "Nutrition Info Finder" 3. On the "Nutrition Info Finder" page, the user enters the food that they wish to look up. • If the user enters an invalid input, the system displays the message: "Input is invalid, please enter a correct value." • 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. 4. The system displays the nutritional and caloric value. 5. The user can check the nutritional content of other food items.

Alternative Flows:	1. Valid Input But No Data Found 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."
Exceptions:	1. Database Unavailable • If the food database is down or unreachable, the system displays: "Service is currently unavailable. Please try again later."
Includes:	
Special Requirements:	
Assumptions:	It is assumed that User has already registered and possesses an account to access the features of the website. The food database is up-to-date and contains accurate nutritional information.
Notes and Issues:	

3.2. **U0302 Generate AI-Powered Dietary Suggestions**

Use Case ID:	U0302		
Use Case Name:	Generate AI-Powered Dietary Suggestions		
Created By:	Liu Xiaotao	Last Updated By:	Mahi Pandey
Date Created:	2/9/2024	Date Last Updated:	09/11/2024

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:	1. The user must be logged into the EatWellthy application with a valid account. 2. The user must provide relevant health and dietary information within the application. 3. The system must have access to a comprehensive nutrition database to generate accurate diet plans.
Postconditions:	1. The user receives a personalized diet plan that aligns with their health goals and dietary preferences.

	2. The generated diet plan is stored in the user's profile for easy access and future reference. 3. The user can follow the diet plan and track their progress within the EatWellthy application.
Priority:	High
Frequency of Use:	Daily
Flow of Events:	1. The user logs into the EatWellthy application and navigates to the "Generate Diet Plan" section in the Dashboard. 2. In Profile, the system prompts the user to provide or update the following information: • Age,Gender, Weight, Height • Activity level (e.g., sedentary, lightly active, moderately active, very active) • Health goals (e.g., weight loss, muscle gain, maintenance) • Dietary preferences (e.g., vegetarian, vegan, low-carb, gluten-free) • Any food allergies or restrictions 3. The user enters the required information and confirms their input. 4. The system validates the user's input for completeness and accuracy. If any required information is missing or incorrect, the system prompts the user to correct it. 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). 6. The system generates a personalized diet plan, including: • Suggested daily calorie intake • Recommended meals and snacks • Portion sizes for each meal • Nutritional breakdown (e.g., calories, macronutrients, vitamins, minerals) for each meal • Meal timing recommendations (e.g., breakfast, lunch, dinner, snacks) 7. The system displays the personalized diet plan to user, allowing them to review and confirm the plan. 8. The system stores the diet plan in the user's profile and makes it accessible from the dashboard for daily reference and meal logging.
Alternative Flows:	1. Plan Regeneration

	• 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). • The system generates a new plan based on the updated inputs.
Exceptions:	1. Database Access Issues: • 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. • 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. 2. Inconsistent or Conflicting Inputs: • 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. • The user can modify their inputs to resolve the conflict and proceed with generating the diet plan.
Includes:	
Special Requirements:	User should be able to copy the generated plan and paste it in their personal diet site
Assumptions:	1. The user has updated their “Profile” with the necessary details.
Notes and Issues:	

4. Tracker

4.1. U0401 Log Your Daily Meals

Use Case ID:	U0401		
Use Case Name:	Log Daily Meal		
Created By:	Liu Xiaotao	Last Updated By:	Mahi Pandey
Date Created:	1/9/2024	Date Last Updated:	09/11/2024

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:	1. The user must be logged into the EatWellthy system with a valid account. 2. The system must have access to the user's profile to store meal logs.
Postconditions:	• The meal information is successfully stored in the user's profile in the database. • The system updates the user's nutritional summary for the day, reflecting the logged meal. • Users can view, edit, or delete the meal entry at any time. • The system may provide feedback or suggestions based on the nutritional content of the logged meals.
Priority:	High
Frequency of Use:	Daily
Flow of Events:	1. The user accesses the EatWellthy application and navigates to the "Tracker" section. 2. The system presents the user with a form to input meal details: • Meal type (e.g., breakfast, lunch, dinner, snacks) • Food items consumed • Portion sizes 3. The user fills in the necessary information and clicks "Submit." 4. The system validates the input: • Ensures that all required fields are filled. • Verifies that the portion sizes are reasonable (e.g., not negative).

	5. If the input is valid, the system stores the meal information in the user's profile. 6. The system updates the user's daily nutritional summary with the new data. 7. The system displays a confirmation message to the user, indicating that the meal has been successfully logged.
Alternative Flows:	1. Validation Errors: • If any required fields are missing or contain invalid data, the system displays an error message. • The user corrects the input and resubmits the form. • The process continues once the input is valid. 2. Editing a Meal Log: • The user can navigate to their meal history and select a previously logged meal. • The system displays the meal details, allowing the user to make changes. • The user submits the changes, and the system updates the meal information in the database and adjusts the nutritional summary accordingly. 3. Deleting a Meal Log: • The user selects a meal from their meal history that they wish to delete. • The system prompts the user to confirm the deletion. • Upon confirmation, the system removes the meal from the user's profile and adjusts the nutritional summary
Exceptions:	Network Failure: • 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. System Unavailability: • 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.
Includes:	Confirm message

Special Requirements:	
Assumptions:	
Notes and Issues:	

4.2. **U0402 Edit Logged Meal**

Use Case ID:	U0402		
Use Case Name:	Edit Logged Meal		
Created By:	Liu Xiaotao	Last Updated By:	Mahi Pandey
Date Created:	1/9/2024	Date Last Updated:	9/11/2024

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:	1. The user must be logged into the EatWellthy system with a valid account. 2. The user must have previously logged a meal that they wish to edit. 3. The system must have access to the user's meal history.
Postconditions:	1. The updated meal information is successfully stored in the user's profile in the database. 2. The system recalculates and updates the user's daily nutritional summary based on the changes. 3. The user can view the updated meal entry and the adjusted nutritional information.
Priority:	Low
Frequency of Use:	As needed
Flow of Events:	1. The user accesses the EatWellthy application and navigates to their meal history. 2. The system presents a list of previously logged meals, organised by date and time. 3. The user selects the meal they wish to edit. 4. The system displays the meal details, including: • Meal type (e.g., breakfast, lunch, dinner, snacks) • Food items consumed • Portion sizes • Time of consumption

	5. The user makes the necessary changes to the meal details. 6. The user clicks “Save” to submit the changes. 7. The system validates the new input: • Ensures all required fields are filled. • Verifies that the portion sizes are reasonable (e.g., not negative). 8. If the input is valid, the system updates the meal information in the user’s profile. 9. The system recalculates the user’s daily nutritional summary to reflect the updated meal data. 10. The system displays a confirmation message, indicating that the meal has been successfully updated.
Alternative Flows:	1. Validation Errors: • If any required fields are missing or contain invalid data, the system displays an error message. • The user corrects the input and resubmits the form. • The process continues once the input is valid. 2. Cancelling an Edit: • The user may choose to cancel the edit at any time before saving. • The system discards any changes made and returns to the meal history view without altering the original meal log.
Exceptions:	1. Network Failure: • 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. 2. System Unavailability: • 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.
Includes:	
Special Requirements:	
Assumptions:	1. Users have already previously logged a meal that they now wish to edit.
Notes and Issues:	

4.3. **U0403 Delete Logged Meal**

Use Case ID:	U0403
Use Case Name:	Delete Logged Meal

Created By:	LiuXiaotao	Last Updated By:	Mahi Pandey
Date Created:	1/9/2024	Date Last Updated:	9/11/2024

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:	1. The user must be logged into the EatWellthy system with a valid account. 2. The user must have at least one previously logged meal in their history. 3. The system must have access to the user's meal history.
Postconditions:	1. The selected meal is permanently removed from the user's profile in the database. 2. The system recalculates and updates the user's daily nutritional summary based on the deletion. 3. The user can view their updated meal history and nutritional information.
Priority:	
Frequency of Use:	As needed
Flow of Events:	1. The user accesses the EatWellthy application and navigates to their meal history. 2. The system presents a list of previously logged meals, organized by date and time. 3. The user selects the meal they wish to delete. 4. The system displays the meal details, including: • Meal type (e.g., breakfast, lunch, dinner, snacks) • Food items consumed • Portion sizes • Time of consumption 5. The user clicks the "Delete" button to remove the selected meal. 6. The system validates the request and proceeds to delete the meal from the user's profile. 7. The system recalculates the user's daily nutritional summary to reflect the removal of the meal data. 8. The user's meal history is updated to exclude the deleted meal.

Alternative Flows:	NIL
Exceptions:	1. Network Failure: • 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. 2. System Unavailability: • 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.
Includes:	
Special Requirements:	
Assumptions:	1. User has a previously logged meal that they want to know delete 2. The user is sure about deleting the current meal.
Notes and Issues:	

4.4. **U0404 View and Search Meal History**

Use Case ID:	U0404		
Use Case Name:	View and Search Meal History		
Created By:	Liu Xiaotao	Last Updated By:	Mahi Pandey
Date Created:	1/9/2024	Date Last Updated:	09/11/2024

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:	1. The user must be logged into the EatWellthy system with a valid account. 2. The user must have logged at least one meal in the system. 3. The system must have access to the user's meal history data.
Postconditions:	1. The user must be logged into the EatWellthy system with a valid account. 2. The user must have logged at least one meal in the system.

	3. The system must have access to the user's meal history data.
Priority:	High
Frequency of Use:	As needed
Flow of Events:	1. The user accesses the EatWellthy application and navigates to the "History Search" section. 2. The user inputs the date for which they want to see the meal history for. 3. The system retrieves the user's logged meal data from the database. 4. The system presents the meal history in a chronological list, displaying the following for each meal: • Date and time of the meal • Meal type (e.g., breakfast, lunch, dinner, snacks) • Summary of food items consumed • Total calories and key nutritional information (e.g., protein, carbs, fats) 5. The user can exit the meal history section and return to the main dashboard or another section of the application.
Alternative Flows:	1. No Logged Meals: • If the user has not logged any meals, the system displays a message indicating that no meal history is available. • The user is prompted to log their first meal.
Exceptions:	1. Data Retrieval Failure: • If the system fails to retrieve meal history due to a network or server issue, the user is notified with an error message. • The system may offer an option to retry or advise the user to check back later. 2. Incomplete Data: • 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. • The user is informed about incomplete data and is given options to refresh or contact support.
Includes:	
Special Requirements:	

Assumptions:	1. The user has logged meals for the day they wish to search up.
Notes and Issues:	Avoid the situation that the system loaded the data of another user

4.5. **U0405 Add Customised Food Option**

Use Case ID:	U0405		
Use Case Name:	Add Customised Food Option		
Created By:	Liu Xiaotao	Last Updated By:	Mahi Pandey
Date Created:	1/9/2024	Date Last Updated:	09/11/2024

Actor:	Current User
Description:	This use case allows the user to add a custom food item to their profile. The customized food option can include specific nutritional information, portion size, and any additional dietary notes, which will then be considered in the user's personalized recommendations.
Preconditions:	1. The user must be logged into their account. 2. The user must have navigated to the section where they can manage or add food options.
Postconditions:	1. The custom food option is saved to the user's profile and can be used for future meal planning and recommendations. 2. The system updates the user's nutritional profile to include the new customized option.
Priority:	Medium
Frequency of Use:	Expected to be used infrequently, mainly when users want to add foods that are not in the existing database.
Flow of Events:	1. The user selects the option to add a custom food item. 2. The system displays a form where the user can input details about the food item (e.g., name, portion size, calories, macronutrients, etc.). 3. The user completes the form and submits it. 4. The system validates the information and saves the custom food item to the user's profile. 5. The system confirms to the user that the custom food option has been successfully added.
Alternative Flows:	**Alternative Flow 1**: User Cancels Action • At any point during the flow, the user can cancel the action. • The system will discard any entered information and return the user to the previous screen.

	Alternative Flow 2: Duplicate Food Option • If the user tries to add a food item that already exists in the system or their profile, the system prompts the user to update the existing entry instead of adding a duplicate.
Exceptions:	1. Validation Error – If the user enters invalid data (e.g., negative values for calories), the system prompts the user to correct the information. 2. Connection Error – If there is a network issue while submitting, the system notifies the user and prompts them to try again later.
Includes:	1. User Authentication
Special Requirements:	Flexible data entry, secure handling and storage of user data
Assumptions:	1. The user is aware of the nutritional information of the custom food option
Notes and Issues:	

5. Analysis

5.1. U0501 Generate User Report

Use Case ID:	U0501		
Use Case Name:	Generate User Reports		
Created By:	Zhang Yichi	Last Updated By:	Mahi Pandey
Date Created:	08/30/2024	Date Last Updated:	9/11/2024

Actor:	User
Description:	System generates an analysis report for the user based on their profile information.
Preconditions:	1. User must be logged in. 2. The system should have access to all relevant user data. 3. Report generation functionalities must be implemented.
Postconditions:	1. The system generates a comprehensive report based on user data. 2. The report accurately reflects the user's dietary and nutritional intake.
Priority:	Medium

Frequency of Use:	Monthly (Users typically generate reports on a monthly basis.)
Flow of Events:	1. User logs into the system. 2. User navigates to the “Analysis” section. 3. The system retrieves necessary data from the database. 4. The system generates the report and displays it on the screen or allows download.
Alternative Flows:	1. If the system fails to generate the report, an error message is displayed to the user. 2. If no data is available for the selected parameters, the system informs the user.
Exceptions:	1. Data inconsistencies or missing data may affect the accuracy of the analysis.
Includes:	
Special Requirements:	
Assumptions:	1. The user has provided accurate profile information
Notes and Issues:	

6. Grocery

6.1. U0601 Navigate to Grocery Store Pages

Use Case ID:	U0601		
Use Case Name:	Navigate to Grocery Store Pages		
Created By:	Mahi Pandey	Last Updated By:	Mahi Pandey
Date Created:	09/11/2024	Date Last Updated:	09/11/2024

Actor:	Current User
Description:	This use case allows the user to navigate to various grocery store pages within the *EatWellthy* application. Users can view available stores and select one to see items, offers, or other relevant information provided by the grocery store.
Preconditions:	1. The user must be logged into their account. 2. The user must have a stable internet connection to load store information. 3. The application must have preconfigured grocery store data, including store names, locations, and any available promotional details.
Postconditions:	1. The user is successfully navigated to the selected grocery store page.
Priority:	Medium
Frequency of Use:	Expected to be used moderately, particularly when users want to explore store-specific items or promotions.
Flow of Events:	1. The user selects the “Grocery Stores” option from the main menu or homepage. 2. The system displays a list of available grocery stores, with basic information for each (e.g., name, location, and logo). 3. The user selects a specific store to view. 4. The system loads the page for the selected grocery store, displaying available items, promotions, and any relevant details. 5. The user can browse the items, add items to their list, or navigate back to the main store selection page.
Alternative Flows:	NIL

Exceptions:	1. Connection Error – If there is a network issue while loading the store page, the system notifies the user and prompts them to retry.
Includes:	
Special Requirements:	
Assumptions:	1. The application has up-to-date information on grocery stores, including their products and promotions. 2. Users may access this feature mainly for browsing and adding items to their shopping lists.
Notes and Issues:	

7. Calendar

7.1. U0701 Add Event in Calendar

Use Case ID:	U0701		
Use Case Name:	Add Event in Calendar		
Created By:	Mahi Pandey	Last Updated By:	Mahi Pandey
Date Created:	09/11/2024	Date Last Updated:	09/11/2024

Actor:	Current User
Description:	This use case allows the user to add an event to the in-app calendar. The event could include reminders for meal planning, grocery shopping, or fitness activities that align with the user's dietary and wellness goals.
Preconditions:	1. The user must be logged into their account. 2. The application must have access to the in-app calendar feature or integrated calendar functionality.
Postconditions:	1. The new event is saved in the user's calendar. 2. The system may send notifications or reminders based on the event's scheduled time.
Priority:	Medium
Frequency of Use:	Expected to be used occasionally, particularly when users plan upcoming activities related to meal prep, grocery shopping, or health goals.
Flow of Events:	1. The user selects the "Calendar" option from the side bar. 2. The system displays the calendar interface, showing the current month and any existing events. 3. The user selects a specific date and time to add a new event. 4. The system displays a form where the user can input details such as event title, description, time, and frequency (one-time or recurring). 5. The user completes the form and submits it. 6. The system saves the event in the calendar and confirms the addition. 7. The user can view the new event in the calendar interface.
Alternative Flows:	**Alternative Flow 1**: User Cancels Event Addition • At any point, the user can cancel the event addition process, and the system will discard any entered information. **Alternative Flow 2**: Edit or Delete Event

	• If the user wishes to change or remove an event, they can select the existing event in the calendar, and the system will provide options to edit or delete it.
Exceptions:	1. Invalid Date or Time – If the user selects an invalid date or time (e.g., a past date or overlapping event), the system prompts them to choose a valid date and time. 2. Connection Error – If there is a network issue when saving the event, the system notifies the user and prompts them to try again.
Includes:	
Special Requirements:	
Assumptions:	1. The user will primarily use this feature to add events related to food, fitness, or wellness. 2. The system's calendar functionality is up-to-date and accurately displays event information.
Notes and Issues:	

7.2. **U0702 Update Event in Calendar**

Use Case ID:	U0702		
Use Case Name:	Update Event in Calendar		
Created By:	Mahi Pandey	Last Updated By:	Mahi Pandey
Date Created:	09/11/2024	Date Last Updated:	09/11/2024

Actor:	Current User
Description:	This use case allows the user to update an existing event in the in-app calendar. Users may modify event details such as the title, date, time, description, or frequency to ensure their calendar reflects their latest plans and commitments.
Preconditions:	1. The user must be logged into their account. 2. There must be an existing event in the calendar that the user can update.
Postconditions:	1. The event is successfully updated in the calendar with the new information.
Priority:	Medium
Frequency of Use:	Expected to be used occasionally, particularly when users need to adjust event details due to changes in their schedule.
Flow of Events:	1. The user navigates to the calendar section and selects an existing event they wish to update. 2. The system displays the event's current details.

	3. The user selects the “Edit” option and updates the desired details (e.g., title, date, time, description). 4. The user submits the updated information. 5. The system saves the updated event details and confirms the change. 6. The updated event now appears in the calendar with the new details.
Alternative Flows:	**Alternative Flow 1**: User Cancels Update • The user can cancel the update process at any time, and the system discards any changes.
Exceptions:	1. Invalid Update Data – If the user enters invalid information (e.g., end time before start time), the system prompts the user to correct the input. 2. Connection Error – If a network issue occurs while updating the event, the system notifies the user and prompts them to try again later.
Includes:	
Special Requirements:	
Assumptions:	1. The user has a basic understanding of event management and calendar functions. 2. The system accurately tracks and displays updated event information.
Notes and Issues:	

7.3. **U0703 Add Event to Google Calendar**

Use Case ID:	U0703		
Use Case Name:	Add Event to Google Calendar		
Created By:	Liu Xiaotao	Last Updated By:	Mahi Pandey
Date Created:	1/9/2024	Date Last Updated:	9/11/2024

Actor:	Current User
Description:	This use case allows the user to add an event created within the *EatWellthy* app to their Google Calendar. This feature provides convenience for users who want to manage their events and reminders in a unified calendar system.
Preconditions:	1. The user must be logged into their *EatWellthy* account. 2. The user must have a Google account and grant permission for *EatWellthy* to access and modify their Google Calendar.
Postconditions:	1. The selected event is successfully added to the user’s Google Calendar with the specified details (e.g., title, date, time, and any additional notes). 2. The system confirms that the event has been synced with Google Calendar.

Priority:	Medium
Frequency of Use:	Expected to be used occasionally, especially when users prefer to consolidate their events in one calendar.
Flow of Events:	1. The user selects an event in the *EatWellthy* app they wish to add to Google Calendar. 2. The system prompts the user to connect their Google account if they have not already done so. 3. The user provides authorization for *EatWellthy* to access their Google Calendar. 4. The system displays a confirmation prompt showing the event details to be synced. 5. The user confirms the event addition. 6. The system adds the event to Google Calendar and displays a success message.
Alternative Flows:	**Alternative Flow 1**: Authorization Denied • If the user denies authorization, the system displays a message explaining that Google Calendar integration requires permission and returns the user to the event page. **Alternative Flow 2**: User Cancels Event Addition • The user can cancel the process at any time before the final confirmation, and the system discards any unsynced changes.
Exceptions:	1. Authorization Error – If the system encounters an issue during authorization (e.g., invalid credentials), it prompts the user to retry or check their Google account settings. 2. Connection Error – If there is a network issue while syncing, the system notifies the user and prompts them to try again later.
Includes:	1. User Authentication 2. Google API Integration
Special Requirements:	The feature should comply with Google's API usage policies and limits.
Assumptions:	1. The user has a functional Google account and understands the authorization process. 2. The Google Calendar API is available and operational during the event addition process.
Notes and Issues:	

7.4. **U0704 Sync Diet Plan with Calendar**

Use Case ID:	U0704
Use Case Name:	Sync Diet Plan with Calendar

Created By:	Liu Xiaotao	Last Updated By:	Mahi Pandey
Date Created:	1/9/2024	Date Last Updated:	9/11/2024

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:	1. The user must be logged into the EatWellthy system with a valid account. 2. The user must have a diet plan created within the EatWellthy application. 3. The user must have an active Google account and must be signed into Google services. 4. The user must authorise the EatWellthy app to access and manage their Google Calendar.
Postconditions:	1. The user's diet plan is successfully synced with their Calendar. 2. The system automatically creates calendar events for each meal or diet-related activity, based on the user's diet plan. 3. Users are prompted to add the diet plan to their Google Calendar as well. 4. Users receive notifications on their devices as per the scheduled times in the Google Calendar.
Priority:	Medium
Frequency of Use:	Frequently (every time the plan is generated)
Flow of Events:	1. The user logs into the EatWellthy application and navigates to the "Diet Plan" section in the Dashboard. 2. The user selects the option to sync their diet plan with Calendar. 3. The diet plan is added to the EatWellthy calendar. 4. The system prompts the user to add their diet plan to their Google Calendar. 5. The system prompts the user to sign in to their Google account, if they haven't already. 6. The system requests the necessary permissions to access and manage the user's Google Calendar. 7. The user grants permission, allowing the EatWellthy app to access their Google Calendar. 8. The system retrieves the user's diet plan and begins syncing the data with the Google Calendar. 9. For each meal or diet-related event in plan, system: • Creates a corresponding event in Google Calendar.

	• Sets the event time and date based on the meal schedule. • Includes details in the event description, such as the meal type, food items, and nutritional goals. 8. The system confirms the successful sync and notifies the user that their diet plan is now available in Google Calendar. 9. The user can now view, edit, or receive reminders for their diet plan directly from Google Calendar.
Alternative Flows:	1. User does not wish to Sync to Google Calendar • If the user denies the need to add to Google Calendar, the diet plan is stored in the Eatwellthy Calendar. 2. Authorization Failure: • If the user denies the authorization request, the system displays a message explaining that syncing cannot proceed without the necessary permissions. • The user is given the option to retry the authorization process. 3. Sync Failure: • If the system encounters an issue while syncing (e.g., network issues, Google API errors), the system displays an error message. • The user is given options to retry the sync or to troubleshoot the issue. 4. Partial Sync: • 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. • The system provides details on which events were not synced and offers options to retry syncing only those events.
Exceptions:	1. Network Issues: • 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. • The user can manually retry the sync after resolving any network issues. 2. Google Calendar Quota Limits: • If the user's Google Calendar has reached its event creation limit, the system notifies the user and cancels the sync process. • The user is advised to clear up space in their Google Calendar or contact Google support for further assistance. 3. Calendar Event Conflicts:

	• If there are conflicting events in the Google Calendar during the sync (e.g., overlapping times with existing events), the system warns the user. • The user can choose to skip conflicting events or manually resolve the conflicts in Google Calendar.
Includes:	1. User Authentication 2. Google API Integration
Special Requirements:	The system should access and interact with Google calendar successfully.
Assumptions:	1. We assume that the user has a Google account, and the system can access it.
Notes and Issues:	

8. FAQs

8.1. U0801 Display FAQ Page

Use Case ID:	U0801		
Use Case Name:	Display FAQ Page		
Created By:	Mahi Pandey	Last Updated By:	Mahi Pandey
Date Created:	9/11/2024	Date Last Updated:	9/11/2024

Actor:	Current User
Description:	This use case allows the user to view the Frequently Asked Questions (FAQ) page. The FAQ page provides users with answers to common questions about *EatWellthy*, helping them navigate the application, understand its features, and troubleshoot basic issues.
Preconditions:	1. The user must have access to the *EatWellthy* application.
Postconditions:	2. The FAQ page is displayed, and the user can view and scroll through a list of commonly asked questions and answers.
Priority:	Low
Frequency of Use:	Expected to be used occasionally, primarily when users need guidance or have questions about using *EatWellthy*.

Flow of Events:	1. The user selects the “FAQ” option from the side bar. 2. The system loads the FAQ page, displaying a list of categorized questions and answers. 3. The user scrolls through the list to find answers. 4. If needed, the user can navigate back to the previous page or other parts of the application.
Alternative Flows:	**Alternative Flow 1**: FAQ Page Not Available • If the FAQ page fails to load (e.g., due to network issues), the system displays an error message and provides options to retry or contact support.
Exceptions:	1. Network Error – If there is a network issue, the FAQ page may not load. The system notifies the user and offers a retry option. 2. Page Not Found – If the FAQ page URL is incorrect or the page is unavailable, the system shows an error message and suggests navigating to the help center.
Includes:	
Special Requirements:	
Assumptions:	1. The FAQ content is kept up to date with the latest information about *EatWellthy*.
Notes and Issues:	

9. Location

9.1. U0901 Search for Near-by Grocery Stores

Use Case ID:	U0901		
Use Case Name:	Search for Near-by Grocery Stores		
Created By:	Mahi Pandey	Last Updated By:	Mahi Pandey
Date Created:	9/11/2024	Date Last Updated:	9/11/2024

Actor:	Current User
Description:	This use case enables users to search for nearby grocery stores based on their current location or a location they key into the search bar. The feature aims to help users find accessible stores for convenient shopping, displaying relevant store details such as location, hours, and potential offers.
Preconditions:	1. The user must be logged into their *EatWellthy* account. 2. The application must have permission to access the user's location. 3. The device must have internet connectivity for retrieving store data.
Postconditions:	1. The user is presented with a list of grocery stores within a specified radius of their current location.
Priority:	High
Frequency of Use:	Expected to be used frequently, especially by users looking for nearby grocery options or planning their shopping trips.
Flow of Events:	• The user navigates to the search feature and selects the option to find nearby grocery stores. • The system requests permission to access the user's current location (if not already granted). • The system retrieves the user's location and searches for grocery stores within a specified radius. • The system displays a list of nearby grocery stores, including basic information such as store name, address, and distance. • The user can select a store from the list to view more details, such as store hours, directions, or available promotions.
Alternative Flows:	**Alternative Flow 1**: User Denies Location Access

	• If the user denies access to their location, the system displays a message explaining that location access is required to use this feature and returns the user to the main search page. **Alternative Flow 2**: Search by Zip Code or Address • If the user does not want to share their current location, they can manually enter a zip code or address to search for nearby stores. • The system retrieves stores based on the provided location and displays results as in the primary flow.
Exceptions:	1. Location Error – If the system cannot retrieve the user's location, it notifies the user and prompts them to check device settings or enter a location manually. 2. No Stores Found – If there are no stores within the search radius, the system displays a message suggesting the user expand the radius or try another location.
Includes:	1. Google Maps API Integration
Special Requirements:	
Assumptions:	1. Users may rely on this feature to quickly locate grocery stores near them.
Notes and Issues:	

10. **WellOh**

10.1. **U1001 Ask Nutritional Questions**

Use Case ID:	U1001		
Use Case Name:	Ask Nutritional Questions		
Created By:	Liu Xiaotao	Last Updated By:	Mahi Pandey
Date Created:	02/09/2024	Date Last Updated:	09/11/2024

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 personalised 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:	1. The user must be logged into the EatWellthy application. 2. The Welloh ChatBot must be connected to the internet and have access to the latest nutritional databases and GPT-based model. 3. The ChatBot's underlying GPT model must be optimised for performance, ensuring quick and relevant responses.
Postconditions:	1. The user receives an accurate and relevant response to their nutritional question 2. The system logs the interaction for future reference and potential improvement of the ChatBot's responses.
Priority:	High
Frequency of Use:	As needed
Flow of Events:	1. The user opens the EatWellthy application and navigates to the "Welloh ChatBot" feature. 2. The user types a nutritional question into the ChatBot interface. Examples of questions include: • "What are the health benefits of eating avocados?" • "How many calories are in a medium-sized apple?" • "Can you suggest a high-protein breakfast?"

	3. The system receives the user's input and processes the question through the Welloh ChatBot's GPT-based model. 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. 5. The system displays the response to the user within 2 seconds of receiving the question. The response may include: • Nutritional information (e.g., calorie content, macronutrient breakdown) • Health benefits and potential risks of certain foods • Personalized meal suggestions based on the user's profile and dietary preferences 6. The user reviews the response and can either ask follow-up questions, save the response, or incorporate suggestions into their meal plan. 7. The system logs the interaction for future reference, potentially improving future responses through machine learning.
Alternative Flows:	1. Unavailable Information: • 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. • The user can either modify their question or accept the ChatBot's suggestions. 2. Technical Issues: • If the ChatBot experiences technical issues (e.g., connectivity problems, server downtime), it informs the user and suggests trying again later. • The system may offer offline alternatives, such as browsing the FAQ section or accessing preloaded nutritional guides.
Exceptions:	1. Excessive Response Time: • If the ChatBot takes longer than 2 seconds to generate a response, the system displays a message indicating that the response is delayed. • The user is given the option to wait or rephrase the question for a potentially faster response.
Includes:	
Special Requirements:	
Assumptions:	1. It assumed the system can have a stable GPT service. 2. The user has a valid nutritional question.

Notes and Issues:	May need some prompt techniques for an accurate response.

10.2. **U1002 Query Personalised Nutritional Information via Chatbot**

Use Case ID:	U1002		
Use Case Name:	Query Nutritional Information via Chatbot		
Created By:	Liu Xiaotao	Last Updated By:	Mahi Pandey
Date Created:	2/9/2024	Date Last Updated:	9/11/2024

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 optimised 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:	1. The user must be logged into the EatWellthy application. 2. The Welloh ChatBot must be operational, with access to the latest nutritional databases and the GPT-based model. 3. The ChatBot's system must be up-to-date, ensuring that the nutritional information provided is accurate and reliable.
Postconditions:	1. The user receives precise nutritional information based on their query. 2. The system logs the interaction for future reference, helping to improve the accuracy and relevance of future responses. 3. The user has the option to save the queried information or add related items to their meal plan.
Priority:	Medium
Frequency of Use:	As needed
Flow of Events:	1. The user opens the EatWellthy application and navigates to the "Welloh ChatBot" feature. 2. The user types a specific query regarding nutritional information into the ChatBot interface. Examples of queries include:

	• "How can I incorporate more carbohydrates in my current plan?" • "Is it good for me to eat beef?" 3. The system receives the user's query and processes it through the Welloh ChatBot's GPT-based model. 4. The ChatBot analyzes the query, retrieves relevant nutritional data from its database, and generates a response. 5. The system displays the nutritional information to the user within 2 seconds of receiving the query. The response may include: • Calorie content • Macronutrient breakdown (proteins, fats, carbohydrates) • Micronutrient details (vitamins, minerals) • Additional health-related information or advice 6. The user reviews the information and can choose to ask follow-up questions for more details. 7. The system logs the interaction for future analysis and improvement of the ChatBot's responses.
Alternative Flows:	1. Unavailable Information: • 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). • The user can choose to refine their query or select from the suggested alternatives.
Exceptions:	1. Excessive Response Time: • If the ChatBot takes longer than 2 seconds to generate a response, the system displays a message indicating that the response is delayed. • The user is given the option to wait or rephrase the question for a potentially faster response.
Includes:	
Special Requirements:	
Assumptions:	1. It assumed the system can have a stable GPT service. 2. It is assumed that the user has updated their profile information accurately. 3. It is assumed user has a valid question
Notes and Issues:	May need some prompt techniques for an accurate response.

11. Database Management

11.1. U1101 Update Food Database

Use Case ID:	U1101		
Use Case Name:	Update Food database		
Created By:	Mehta Rishika	Last Updated By:	Mehta Rishika
Date Created:	29/8/2024	Date Last Updated:	09/11/2024

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:	The system must have an active and functional API connection for retrieving updates. The food database must be capable of being updated dynamically without manual intervention. The API provides data in a format compatible with the database schema.
Postconditions:	1. The food database is updated with the latest information from the API. 2. Users can access and view the most current nutritional information through the application. 3. Any changes are logged for tracking and auditing purposes. 4. Users can store the information in the database in their meal history.
Priority:	
Frequency of Use:	As updates from the API are received.
Flow of Events:	1. Trigger Update - The API sends a notification or data update to the system. 2. Retrieve Data - The system connects to the API and retrieves the updated food data. 3. Process Data - The system processes the data, performing validation and transformation as needed. 4. Update Database - The system updates the food database with the new or modified information.
Alternative Flows:	1. Data Retrieval Issue • If the API cannot be reached or data retrieval fails, the system logs the error and retries the operation or alerts the administrator.

	• If the user enters an invalid input, the Nutrition Finder page displays “Invalid input. Please enter a valid Food/ meal” 2. Data Processing Error • If the data is invalid or processing fails, the system logs the issue and may either retry or notify the administrator for manual review.
Exceptions:	1. If the API is down or returns an error, the system handles the exception by implementing a retry mechanism or notifying support personnel. 2. If an issue occurs during the database update (e.g., connectivity issues or schema mismatches), the system logs the error and attempts to roll back changes if possible.
Includes:	
Special Requirements:	
Assumptions:	1. The API is reliable and delivers data in a timely and accurate manner. 2. The database schema is designed to accommodate dynamic updates and changes.
Notes and Issues:	

11.2. **U1102 Manage User Data**

Use Case ID:	U1102		
Use Case Name:	Manage User data		
Created By:	Mehta Rishika	Last Updated By:	Mehta Rishika
Date Created:	29/8/2024	Date Last Updated:	09/11/2024

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:	1. The system must have access to the hashed user database. 2. User profiles must exist in the database for updates or deletions.

	3. The system must ensure that data management actions comply with security and privacy regulations.
Postconditions:	1. User profiles are accurately created, updated, or deleted as required. 2. The database reflects the most current user information. 3. The integrity and confidentiality of user data are preserved.
Priority:	
Frequency of Use:	Depends on user actions and needs
Flow of Events:	1. The system detects a request to create, update, retrieve, or delete a user profile. 2. The system verifies the user's identity and permissions automatically if applicable. 3. Data Management Actions - • Create Profile: The system adds a new user profile to the database based on provided information upon registering. The system displays "User successfully registered" • Update Profile: The system updates an existing user profile with new data, ensuring validation and correctness. The system displays "Profile updated successfully" • Retrieve Profile: The system retrieves and displays user profile data as requested. • Delete Profile: The system removes a user profile from the database based on the request. 4. The system ensures that the database is updated in real time to reflect the changes made.
Alternative Flows:	1. 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". 2. If the deletion fails, the system logs the error and informs the user.
Exceptions:	1. The system denies access to data management functions if the request lacks proper authorization and logs the attempt.
Includes:	1. Authentication and authorization mechanisms. 2. Data validation processes.

	3. Error handling and logging functions.
Special Requirements:	
Assumptions:	1. The system has the necessary access and permissions to manage user data. 2. Data management functions are integrated with the system's security and privacy protocols.
Notes and Issues:	

11.3. **U1103 Backup and Restore**

Use Case ID:	U1103		
Use Case Name:	Backup and Restore		
Created By:	Mehta Rishika	Last Updated By:	Mehta Rishika
Date Created:	29/8/2024	Date Last Updated:	09/11/2024

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:	1. The system must have access to the database and backup storage. 2. Backup policies and schedules must be defined and configured. 3. Data to be backed up must be in a consistent state.
Postconditions:	1. Backup files are created and stored securely according to the backup schedule. 2. Data is restored accurately from backup files as needed. 3. The system ensures the integrity and availability of user data.
Priority:	High
Frequency of Use:	Regularly scheduled (weekly) for data recovery
Flow of Events:	1. The system triggers a backup process automatically according to the updates received from the Nutritionix API connection. 2. The system collects and compiles user data and other critical information from the database. 3. The data is compressed and encrypted if necessary.

	4. The system stores the backup in a secure, designated storage location. 5. Confirmation by updating the logs.
Alternative Flows:	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.
Exceptions:	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. 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.
Includes:	
Special Requirements:	
Assumptions:	1. The backup and restore mechanisms are properly configured and tested. 2. Backup storage is secure and accessible to the system.
Notes and Issues:	