1. **food application (see food nutrition information)**

Member Group 6

|  |  |  |
| --- | --- | --- |
| STT | Name | ID |
| 1 | Bùi Tuấn Anh | 17110001 |
| 2 | Đinh Quốc Thụy | 17110081 |
| 3 | Nguyễn Khắc Hoài Thương | 17110083 |

1. **Software description:**

This is an application that helps you find information about the food you are looking for, and also provides nutrition information in your diet. From there help you to regulate in eating and drinking to have good health. Especially, the application will classify the vegetarian food for you. Through the application, users can create a menu that is suitable and healthy for themselves.

In addition, user can request a new food to admin updating. This feature make application more and more diverse in food.

1. **Current business status report:**

By the internet, everything related to information becomes very easy for users: just need a computer or mobile connected to the internet and a data stream to search almost instantly ... the whole world about the problem that you interested will appear, full of information, images and even sometimes the sound if you need it. It also is development of the search for meal information around the world, significantly changing the face of eating culture, improving the quality of human meals.

Proper nutrition is the cornerstone of health and intelligence. A proper diet that meets the body's needs at different stages of the life cycle will create optimal growth and development for the fetus and young children, as well as effective prevention against the majority of diseases have a good physical strength, good health, high labor productivity and a good quality of life for all ages.

So that, we decided to implement the project "C-Food"

Requirements:

* Store information about:
  + - Food: name, nutrition, weight, id, image, categories, nutrition, ingredients, calories, protein, fat.
    - Categories: id, name
    - Request message: id, message, image, date, status, author.
    - Response messages: id, message, date.
    - Nutrition: Vitamins, Macronutrients, Minerals
    - User: id, name, email, gender, age, weight, favorite food
* Searching:
  + Food: based on name, categories, calories, favorite food…
  + User information: name, email…
  + Request: status, date
  + Vegetarian food: find all vegetarian food
* Calculate:
  + Calories: calculate calories of total food in meal, or base on weight of food.
  + Nutrition: calculate how many percent of each nutrition in food.
  + Statistic: which food is liked the most, total request message in week, month,…
* Report:
  + none

1. List of requirements for software:
   1. Functional business requirements

Division: User ID: user

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **OD** | **Function** | **Type** | **Constraint /**  **Formula code** | **Form code** | **Notes** |
| 1 | View list food | Searching |  |  |  |
| 2 | Search food | Searching | Searching by: name, calories, categories… |  |  |
| 3 | View food details | Searching |  | Food\_detail\_form | Users have to click on food for watch food detail |
| 4 | Like food | Storage | If the current food state is “unlike”, change to “like” and contrary. |  |  |
| 5 | Create meal | Storage | The food must exist on database |  |  |
| 6 | Create message | Storage | Message\_form |  |  |

***Food\_detail\_form:***

Watch detail food

|  |  |  |  |
| --- | --- | --- | --- |
| Image | | Name: … | Weight: … |
| INGREDIENTS | NUTRITIONS |
| LIKE | YES/NO | ….. | …. |
| CATEGORIES | …. |
| NOTE | Vegetarian? |
| Description |
| … |

**Division**: Admin **ID**: manage\_user

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Function** | **Type** | **Constraint /**  **Formula code** | **Form code** | **Notes** |
| 1 | View user info | Searching |  |  |  |
| 2 | Delete account | Storage |  |  |  |
| 3 | Modify user info | Storage |  |  |  |
| 4 | Create account | Storage |  |  |  |
| 5 | Add food | Storage | New food must no exist in database | Add\_food\_form |  |
| 6 | Delete Food | Storage |  |  |  |
| 7 | Update Food | Storage |  |  |  |
| 8 | Calculate calories, proteins, fat | Calculate | Base on weight of each food |  |  |
| 9 | Statistic food watched and liked | Calculate |  |  |  |
| 10 | Create message | Storage | Message\_form |  |  |
| 11 | View message | Searching |  |  |  |
| 12 | Delete message | Storage |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

***Add\_food\_form:***

Add food

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Image | | Name: … | | Weight: … | | |
| INGREDIENTS | | NUTRITIONS | | |
| CATEGORIES: |  | Name | Weight: | Macronutrient | Carbs | value |
| NOTE | Vegetarian? | Proteins | value |
| Description | Fats | value |
| … | … | … | Vitamins | A | Value |
| B | value |
| C | value |
| … | … | … | … |
| Mineral | Calcium | Value |
| … | … |

***message\_form:***

Create message

|  |  |  |
| --- | --- | --- |
| Image | Date: | Dd/mm/yy |
| Title | …. |
|  | Message: | … |

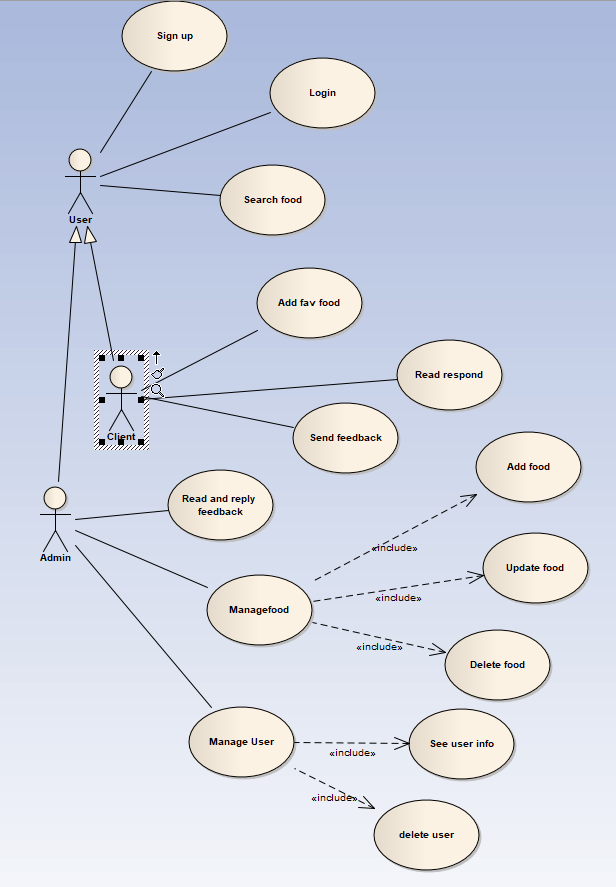
* 1. Functional system requirements

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Function | Description | Note |
| 1 | Permission | User: can use all feature of user division and create request message.  Admin: can use all division, exception: like food. |  |

* 1. Non-functional requirements.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Content | Criteria | Description | Note |
| 1 | Allow user to request new food | Develop | If users cannot find food that they want, they can request for admin to update it |  |
| 2 | Speed of sending message | Performance | User and admin have to receive message immediately. |  |
| 3 | Searching with keyword | Benefit | Allow user to search with keyword or nearly correct food name. |  |
| 4 | O-Authentication | Benefit | User can use Google account to login. | Valid Google account |

1. Modeling requirements



4.2 Scenario

|  |  |
| --- | --- |
| **Name** | Signup |
| **Brief Description** | User signup for an account to the system |
| **Actor(s)** | User has not logged in |
| **Basic Flow** | The use case starts when the user needs to perform certain functions of the system that need to register an account  1. The system displays a window for the user to enter the username and password  2. User enters username and password.  3. System verifies Username and password.  4. The system see if the username already exists.  5. The system opens the login screen. |
| **Alternative Flow** | If the user enters a username that is already existed. The following tasks were performed:  1. The system will prompt the user to enter username and password again.  3. When the user re-enters the username and password, step 3 in Basic flow is performed again. |
| **Pre-conditions** | (none) |
| **Post-condition** |  |
| Success | A new account is added to the system. |
| Failure | The system is not changed. |
| **Extension Point** |  |
| (none) |  |

|  |  |
| --- | --- |
| **Name** | Login |
| **Brief Description** | User login into the system |
| **Actor(s)** | Users have not logged in |
| **Basic Flow** | The use case starts when the user needs to perform certain functions of the system that need to authenticate access  1. The system displays a window for the user to enter the username and password  2. User enters username and password  3. System verifies Username and password.  4. The system determines the user's role  5. The system opens the home screen and opens the functions that are accessible to the user. |
| **Alternative Flow** | If the user enters the wrong username or / and password. The following tasks were performed:  1. The system describing the reason for authentication is wrong  2. The system will prompt the user to login again  3. When the user re-enters the username and password, step 3 in Basic flow is performed again. |
| **Pre-conditions** | The user already has registered a account before. |
| **Post-condition** |  |
| Success | The user is authenticated and the system shows the homepage content |
| Failure | User can not access to system |
| **Extension Point** |  |
| (none) |  |

|  |  |
| --- | --- |
| **Name** | Search by Food Name |
| **Brief Description** | User looks up food information in the system |
| **Actor(s)** | Admin, User |
| **Basic Flow** | Use case triggers when the user chooses a food searching function.  1. The system has a input text bar for the user to input the food name  2. User inputs food name.  3. System will verify and look up food name.  4. The system will list all information about the food that matching with text that user inputs. |
| **Alternative Flow** | If the user input the wrong food name. The following tasks were performed:  1. The system notifies that food does not exist.  2. The system requests the user to input again.  3. When the user re-input the food name, step 3 in Basic flow is performed. |
| **Pre-conditions** | Food Name are already in system. |
| **Post-condition** |  |
| Success | The system will return user the information about the food nutrition by food name. |
| Failure | The system will notify the user that the food does not exist. |
| **Extension Point** |  |
| (none) |  |

|  |  |
| --- | --- |
| **Name** | Search by Categories |
| **Brief Description** | User looks up food are satisfied with categories inputed in the system |
| **Actor(s)** | Admin, User |
| **Basic Flow** | Use case triggers when the user chooses a food searching function.  1. The system has a input text bar for the user to input the categories  2. User inputs categories.  3. System will verify and look up categories.  4. The system will list all information about the food that matching with categories that user inputs. |
| **Alternative Flow** | If the user input the invalid categories. The following tasks were performed:  1. The system notifies that categories does not exist.  2. The system requests the user to input again.  3. When the user re-input the food name, step 3 in Basic flow is performed. |
| **Pre-conditions** | Categories are already in system. |
| **Post-condition** |  |
| Success | The system will return user the information about the food nutrition by categories. |
| Failure | The system will notify the user that the food does not exist. |
| **Extension Point** |  |
| (none) |  |

|  |  |
| --- | --- |
| **Name** | Search by Nutrition |
| **Brief Description** | User looks up food information in the system |
| **Actor(s)** | Admin, User |
| **Basic Flow** | Use case triggers when the user chooses a food searching function.  1. The system has a input text bar for the user to input the nutrition  2. User inputs nutrition.  3. System will verify and look up all food matching nutrition.  4. The system will list all information about the food that matching with nutrition that user inputs. |
| **Alternative Flow** | If the user input the wrong categories. The following tasks were performed:  1. The system notifies that food does not exist.  2. The system requests the user to input again.  3. When the user re-input the food name, step 3 in Basic flow is performed. |
| **Pre-conditions** | Nutrition is already in system. |
| **Post-condition** |  |
| Success | The system will return user the information about the food nutrition by food name. |
| Failure | The system will notify the user that the food name does not exist. |
| **Extension Point** |  |
| (none) |  |

|  |  |
| --- | --- |
| **Name** | Search by Calories |
| **Brief Description** | User looks up food information in the system |
| **Actor(s)** | Admin, User |
| **Basic Flow** | Use case triggers when the user chooses a food searching function.  1. The system has a input text bar for the user to input the calories  2. User inputs amount of calories.  3. System will verify and look up all food having nearly amount of calories that user input.  4. The system will list all information about the valid food. |
| **Alternative Flow** | If the user input the wrong categories. The following tasks were performed:  1. The system notifies that food does not exist.  2. The system requests the user to input again.  3. When the user re-input the calories, step 3 in Basic flow is performed. |
| **Pre-conditions** | Amount of calories is already in system. |
| **Post-condition** |  |
| Success | The system will return user the information about the food nutrition based on calories. |
| Failure | The system will notify the user that the food name does not exist. |
| **Extension Point** |  |
| (none) |  |

|  |  |
| --- | --- |
| **Name** | Send Message |
| **Brief Description** | User send message to admin for some purpose. |
| **Actor(s)** | User |
| **Basic Flow** | Use case triggers when the user chooses sending message function.  1. The system show up a form used to send message and attachment.  2. User fill the text area then click the “Send” button to send.  3. System will display a modal to notify the status of message.  4.Finally, system will transport this message to mail box of admin. |
| **Alternative Flow** |  |
| **Pre-conditions** | User have to login system successfully. |
| **Post-condition** |  |
| Success | The system will return a modal to notify the user sent successfully. |
| Failure | The system will return a modal to notify the user sent fail. |
| **Extension Point** |  |
| (none) |  |

|  |  |
| --- | --- |
| **Name** | Send Feedback |
| **Brief Description** | Admin send feedback to user. |
| **Actor(s)** | Admin |
| **Basic Flow** | Use case triggers when the admin chooses feedback function in mail box.  1. The system show up a list of messages from user and choose one of them to feedback.  2. The system show up a form used to send feedback.  3. Admin fill the text area then click the “Send” button to send.  4. System will display a modal to notify the status of message.  5.Finally, system will transport this message to message box of user. |
| **Alternative Flow** |  |
| **Pre-conditions** | Admin have to login system successfully. |
| **Post-condition** |  |
| Success | The system will return a modal to notify the admin sent successfully. |
| Failure | The system will return a modal to notify the admin sent fail. |
| **Extension Point** |  |
| (none) |  |

|  |  |
| --- | --- |
| **Name** | Add Food |
| **Brief Description** | Add Food and information about the food to the database |
| **Actor(s)** | Admin |
| **Basic Flow** | The use case starts when the users see the main UI of the software  1.The User click on the button with text “Manage Food” on it. 2.User clicks button “add food” and the new window promted.  3.User enters the food name, the nutritions, uploads the image and select the catalog.  4. User clicks save food.  5.The use case ends. |
| **Alternative Flow** |  |
| (none) |  |
| **Pre-conditions** |  |
| Logined as admin | The user has be logged in as admin |
| **Post-condition** |  |
| Success | The new food is successfully added to the system |
| Failure | The exception is prompted to the user. |
| **Extension Point** |  |
| (none) |  |

|  |  |
| --- | --- |
| **Name** | Update food |
| **Brief Description** | Edit food information like nutrition and catalog or image |
| **Actor(s)** | Admin |
| **Basic Flow** | The use case starts when the users see the main UI of the software  1.The User click on the button with text “Manage Food” on it.  2.User will see a list of food. User double click in any row of the list.  3.The user will modify the information like nutritions , image or catalog.  4.The user clicks “Save”. |
| **Alternative Flow** |  |
| (none) |  |
| **Pre-conditions** |  |
| Logined as admin | The user has be logged in as admin |
| **Post-condition** |  |
| Success | The food is successfully modified to the system |
| Failure | The exception is prompted to the user. |
| **Extension Point** |  |
| (none) |  |

|  |  |
| --- | --- |
| **Name** | Delete Food |
| **Brief Description** | Delete Food and information about the food from the database |
| **Actor(s)** | Admin |
| **Basic Flow** | The use case starts when the users see the main UI of the software  1.The User click on the button with text “Manage Food” on it.  2. User will see a list of food. User double click in any row of the list.  3. User clicks delete food.  4.The use case ends. |
| **Alternative Flow** |  |
| (none) |  |
| **Pre-conditions** |  |
| Logined as admin | The user has be logged in as admin |
| **Post-condition** |  |
| Success | The new food is successfully removed from the system |
| Failure | The exception is prompted to the user. |
| **Extension Point** |  |
| (none) |  |

|  |  |
| --- | --- |
| **Name** | See user info |
| **Brief Description** | View the user basic information like user name, email , phone number, and log in activities. |
| **Actor(s)** | Admin |
| **Basic Flow** | The use case starts when the users see the main UI of the software  1.The User click on the button with text “Manage accounts” on it.  2. User will see a list users and with the basic information.  4.The use case ends. |
| **Alternative Flow** |  |
| (none) |  |
| **Pre-conditions** |  |
| Logined as admin | The user has be logged in as admin |
| **Post-condition** |  |
| Accounts info found | User will see a list of user info |
| No Account | User will see an empty list |
| **Extension Point** |  |
| (none) |  |

|  |  |
| --- | --- |
| **Name** | Delete account |
| **Brief Description** | Delete the account from the database |
| **Actor(s)** | Admin |
| **Basic Flow** | The use case starts when the users see the main UI of the software  1.The User click on the button with text “Manage accounts” on it.  2. User will see a list users and with the basic information.  3.User click any row in the list.  4.User click delete.  5.Use case ends. |
| **Alternative Flow** |  |
| (none) |  |
| **Pre-conditions** |  |
| Logined as admin | The user has be logged in as admin |
| **Post-condition** |  |
| Successfully | The account is removed from the system |
| Failure | A message will prompt to the user due to some reasons. |
| **Extension Point** |  |
| (none) |  |

|  |  |
| --- | --- |
| **Name** | Add favorite food |
| **Brief Description** | User add his/her faverite food |
| **Actor(s)** | User |
| **Basic Flow** | The use case starts when the users see the main form  1.The User will see a list of food.  2.In each row there will be a like button.  3.User clicks like button.  4.Use case ends. |
| **Alternative Flow** |  |
| (none) |  |
| **Pre-conditions** |  |
| Logined as user | The user has be logged in first |
| **Post-condition** |  |
| Successfully | The food is added to favorite list |
| Failure | A message will prompt to the user due to some reasons. |
| **Extension Point** |  |
| (none) |  |