**Week 05 group assignment**

**Group 6**

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

1. **Requirement modeling**

**1.The actors and use cases of the system include:**

**Actors:**

-User

-Manage

-Administrators

**Use cases:**

Signup

Login

Search by Food Name

Search by Categories

Search by Nutrition

Search by Calories

Send Message

Add Food

Send Feedback

Update food

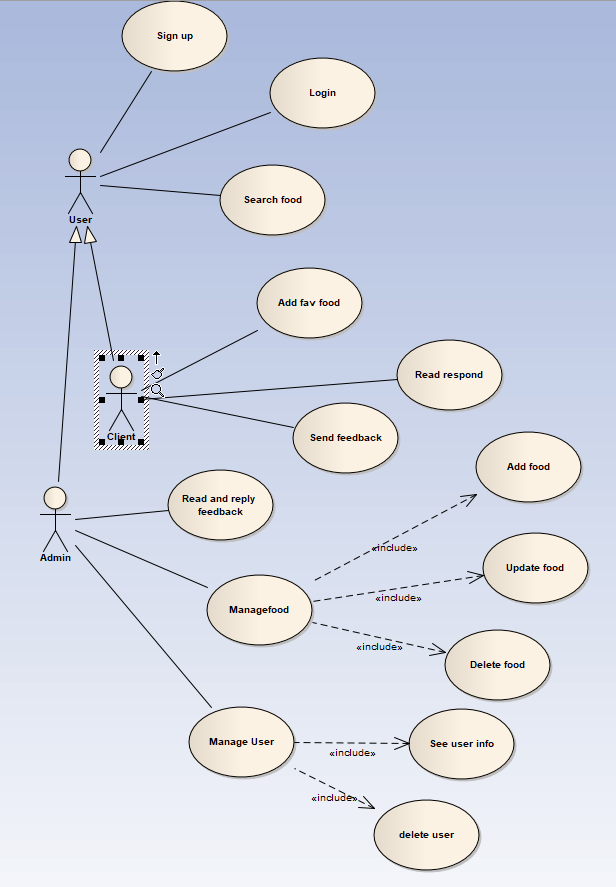
Delete Food

Delete account

See user info

Add favorite food

**2.Draw the usecase diagram(s) for the system.**



**3.Make usecase specifications for all usecases**

|  |  |
| --- | --- |
| **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** | 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) |  |

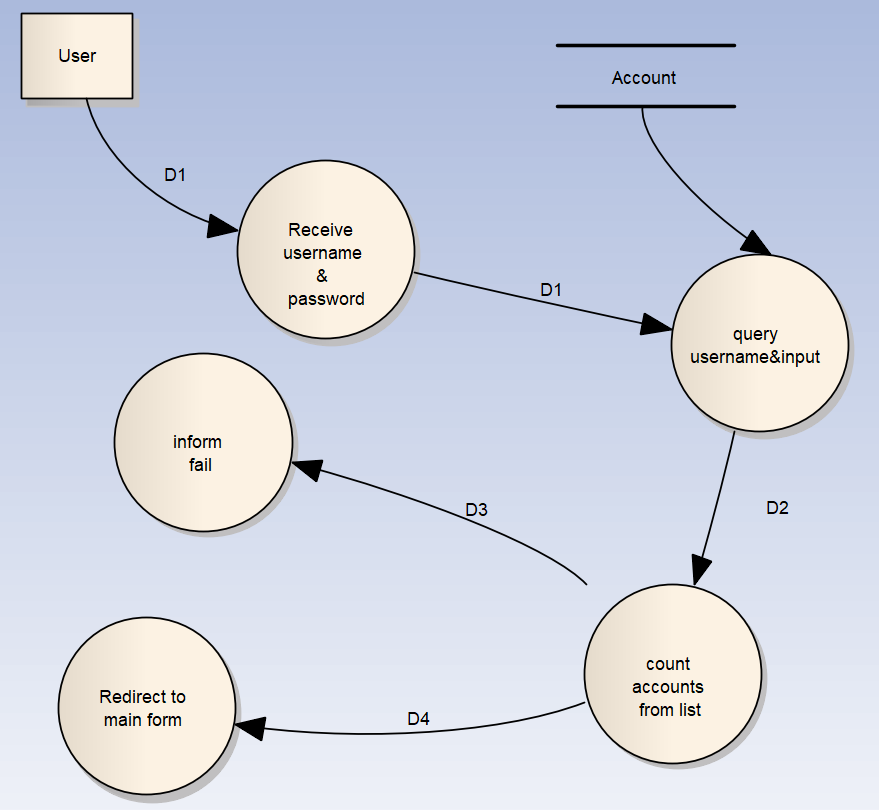
|  |  |
| --- | --- |
| **Name** | See user info |
| **Brief Description** | View the user basic information like user name, email , phone number, and log in activities. |
| **Actor(s)** | Admin |
| **Basic Flow** | The use case starts when the users see the main UI of the software  1.The User click 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) |  |

**II.Choose 10 most important usecases to draw DFD for it**

**1.Login**



* 1. **Description**:

D1 : username and passwrod

D2 : list of accounts

D3 : no accounts in list message

D4 : one account in list message

* 1. **Algorithm**

S1 : receive username and password from user.

S2 : connect to the database

S3 : query the account table in the database with the username and password provided

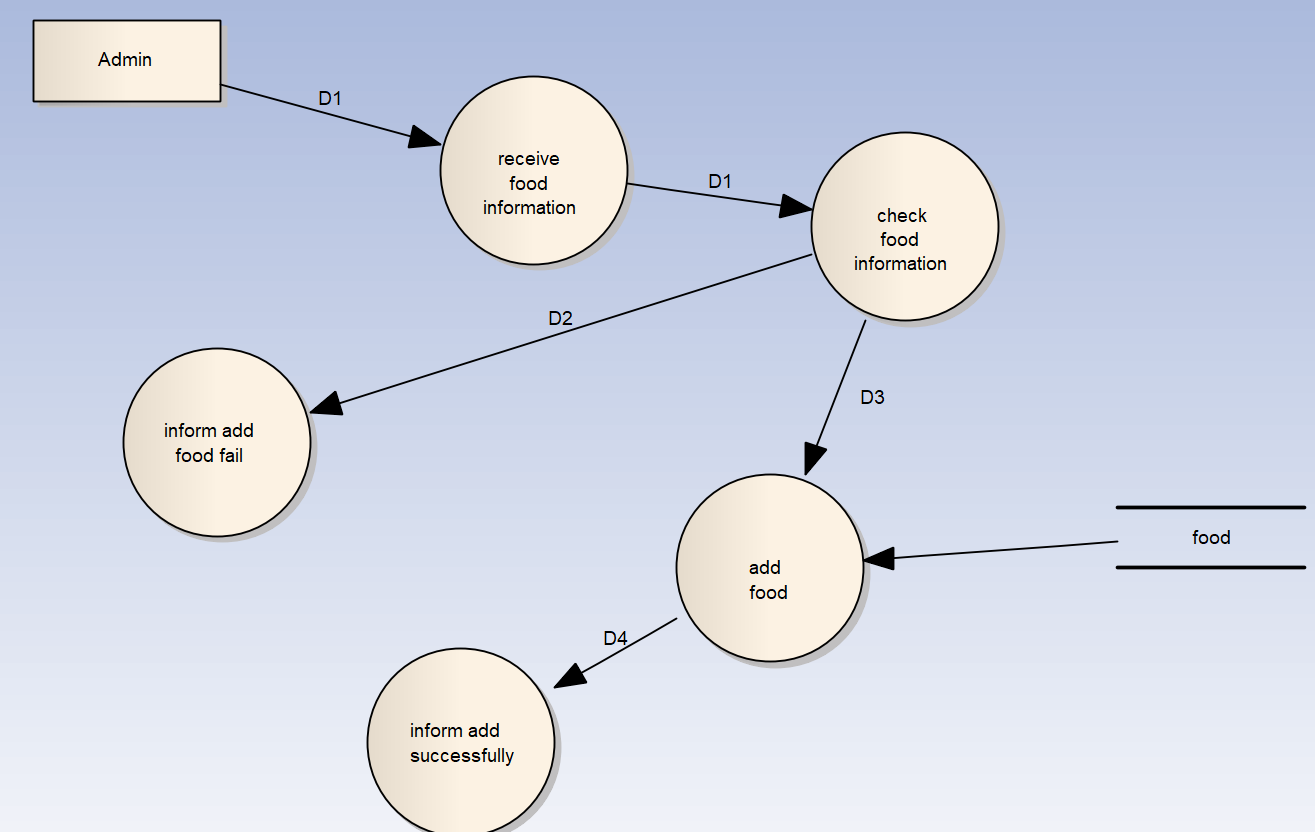
S4 : get the query result then compare .

S5 : if in the query result has one account proceeds S7 or else proceeds S6.

S6 : inform fail login then go back to S1.

S7 : inform successfully then redirects to main form.

**2.Add food**



* 1. **Description**:

D1 : food information

D2 : invalid information message

D3 : add food successfully message

* 1. **Algorithm**:

S1 : receive food information

S2 : check information validity .If the information is valid then proceed S4 or else proceed S3.

S3 : inform that adding fool is fail return to S1.

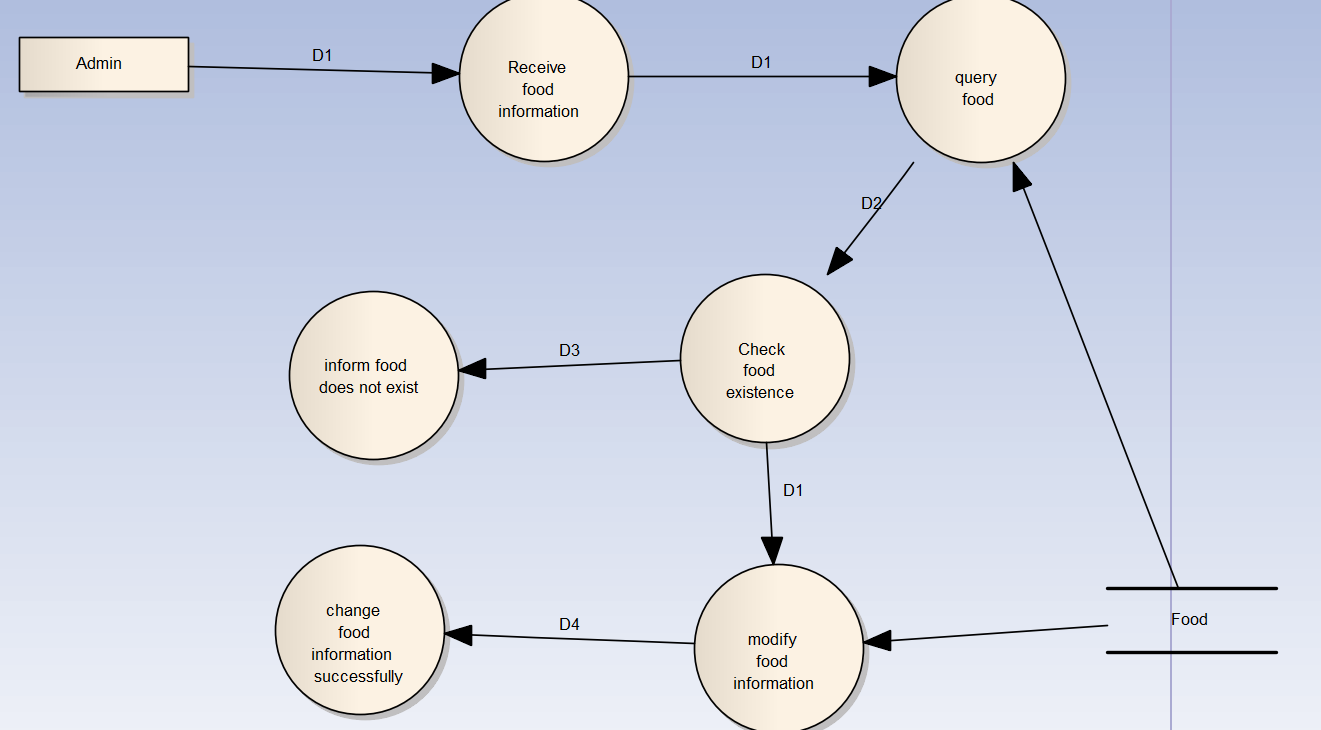
S4 : open connection to the database.

S5 : add food to the database.

S6 : close database connection.

S7 : inform add successfully.

**3.Update food**



* 1. **Description**

D1 : food information

D2 : list of food

D3 : Cannot find food message

D4 : modify food successfully message

* 1. **Algorithm**

S1 : Receive food information about the food to update

S2 : open database connection

S3 : query to get food list in the database.

S4 : Check the food list . If the food list contains one food then proceeds S6 or else proceeds S5.

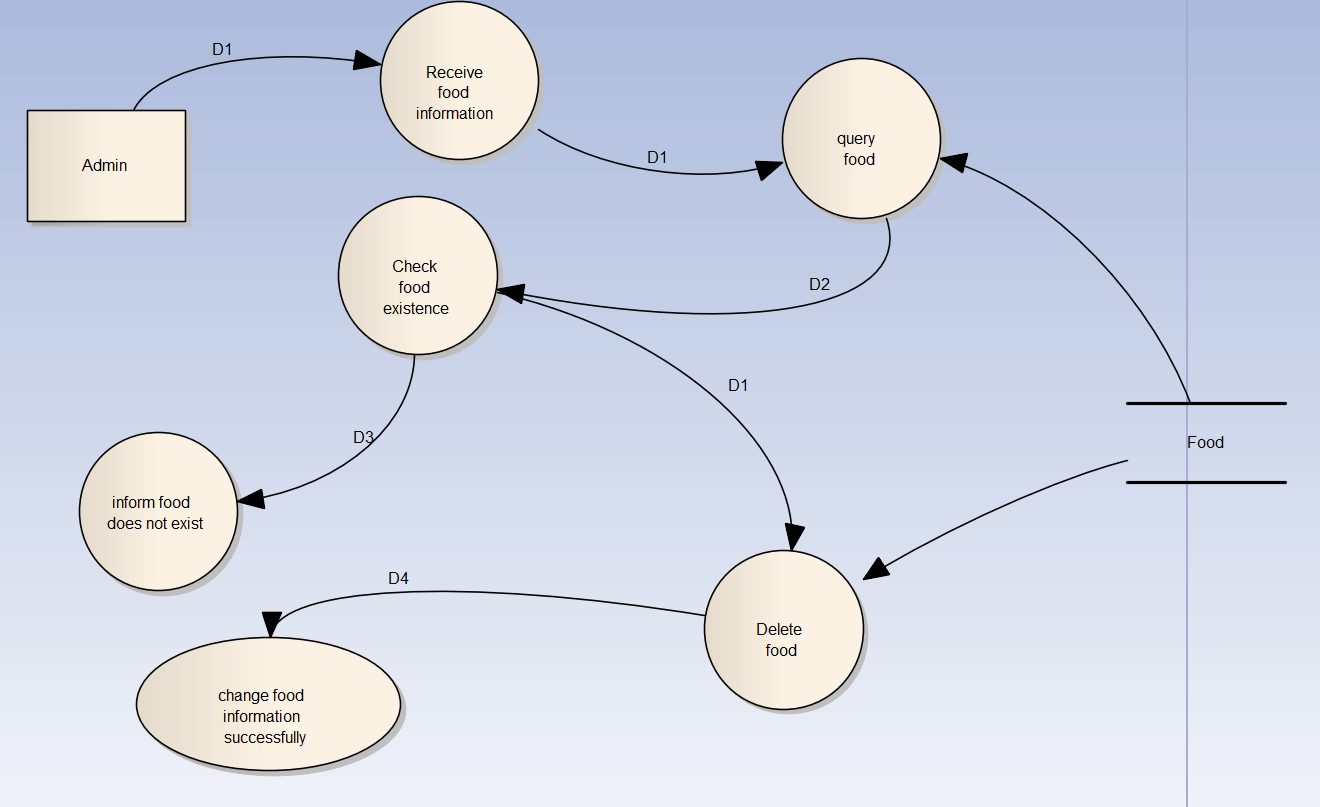
S5 : inform that food does not exist.

S6 : modify the food information then saves to the database.

S7 : close database connection.

S8 : inform that update food successfully.

**4.Delete food**



* 1. **Description**

D1 : food information

D2 : list of food

D3 : Cannot find food message

D4 : delete food successfully message

* 1. **Algorithm**

S1 : Receive food information about the food to update

S2 : open database connection

S3 : query to get food list in the database.

S4 : Check the food list . If the food list contains one food then proceeds S6 or else proceeds S5.

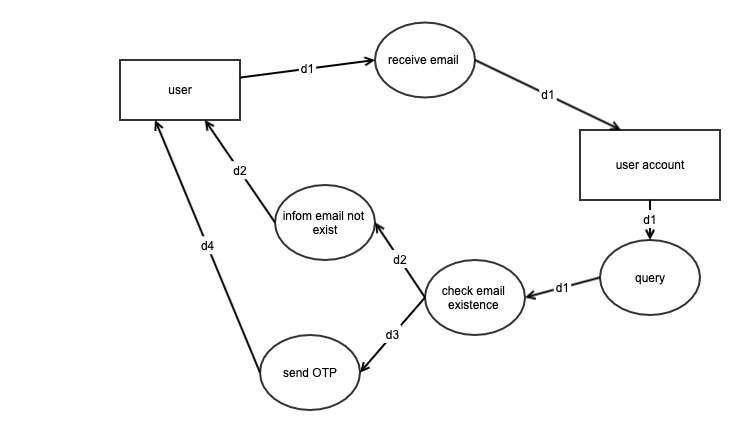
S5 : inform that food does not exist.

S6 : delete the food information in the database.

S7 : close database connection.

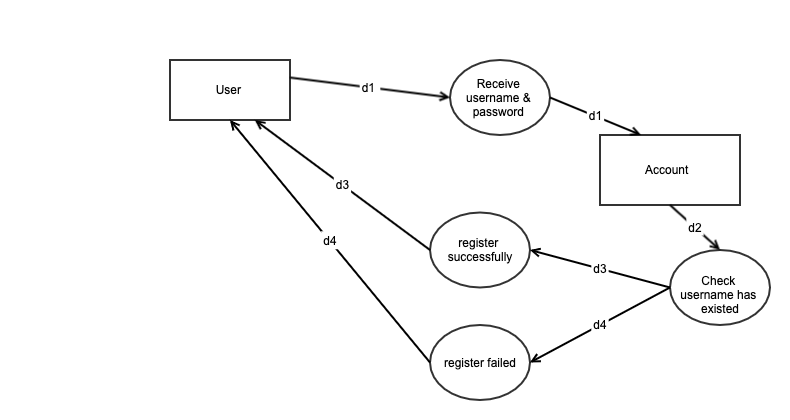
S8 : inform that delete food successfully.

1. **Forget password:**



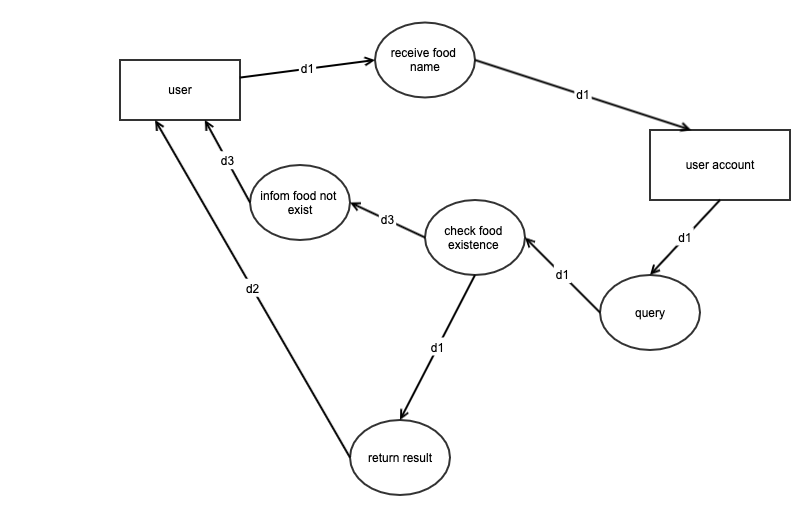
* 1. D1: email
  2. D2: can not find email message
  3. D3: found email
  4. D4: OPT code

1. **REGISTER**



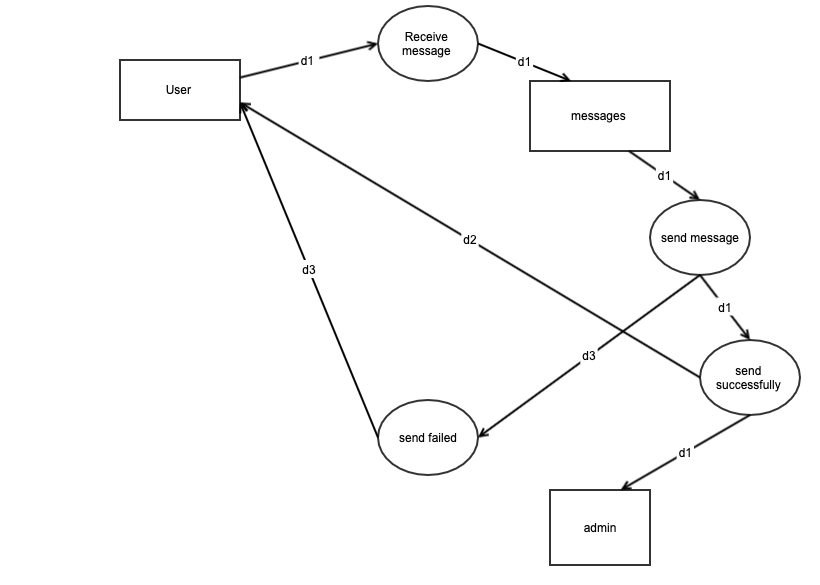
* 1. D1: Food name
  2. D2: check existence
  3. D3: register successfully message
  4. D4: register failed message

1. Search food:



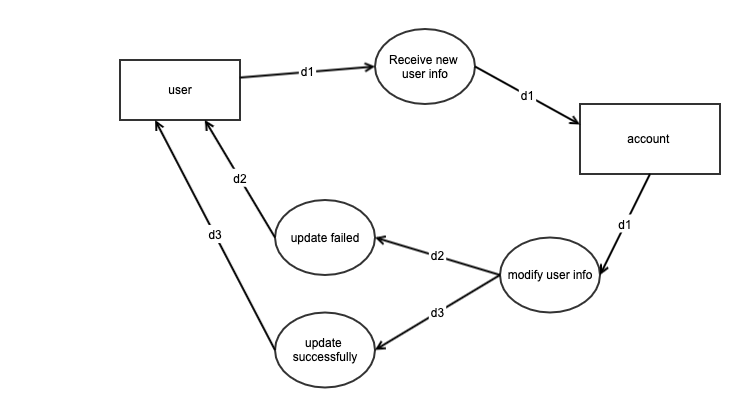
* 1. D1: food name
  2. D2: food list result
  3. D3: can not find food message

1. **Send message**



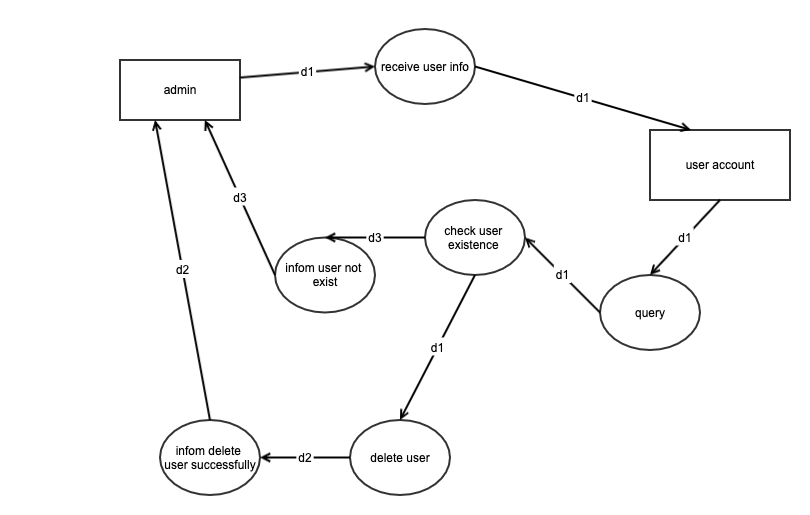
* 1. D1: request message
  2. D2: send successfully status
  3. D3: send failed status

1. **Modify user info**



* 1. D1: new user info
  2. D2: update failed status
  3. D3: updated successfully status

1. **Delete user:**



* 1. D1: user info
  2. D2: delete successfully status
  3. D3: delete failed status