**CS5591 Advanced Software Engineering**

**Calorie 2 Grocery Increment#2 Report**

**Project Group #12**



**Team Members**

Sravani Punyamurthula

Vaishnavi Aienampudi

Leela Naga Devi Gajula

Vinaya Podduturi

**Project Goal:**

The goal of our project is to develop an android application “Calorie2grocery”, which helps its users to plan for a proper meal to maintain healthy life style. Firstly, it provides the calorie count of a recipe. The app has a browser which allows the user to import a recipe and get a list of ingredients. The app then displays the calorie count for each ingredient in the recipe. Each user has his/her own specifications and preferences for recipes. Hence the app provides its users with an option to edit the ingredient list. Users can add or remove ingredients and can get the total calorie count of that recipe. The calorie count helps its users to take healthy choices about their meal. Secondly, it automatically generates a shopping list so that the user does not miss anything that he needs to buy. It provides us the flexibility to log the groceries available at our home. Our app will generate alert notifications about the expiry dates of groceries which in turn helps in reducing the wastage of food. Based on the recipe chosen it, prepares the list of ingredients to buy in order to prepare the recipe. Thus the app functions as a unified meal planner which takes care of grocery management as well as provide calorie information to plan for a healthy meal.

**Increment #2**

We have developed an application which helps the user do the following:

* Add groceries items that are available at home to a list.
* View grocery list
* Search for a recipe.
* See the ingredient list for that recipe.
* Import the ingredient list to the local system.

We developed a native android application for adding the list of groceries along with quantity. We have used web technologies for implementing the client side logic rest of the functionalities listed above. We are storing the data in SQL Server remote database. Hence we have created rest services to push/pull the data from the database.

**Application Specifications:**

|  |  |
| --- | --- |
|  | **Tools** |
| Platform | Android 4.0.3 |
| UML Diagrams | Microsoft Visio |
| Languages | Java, C# , ASP.Net , JQuery , JavaScript , AJAX |
| Database | SQL Server |
| Planning | ScrumDo |
| Version Control System | GIT |
| Existing Rest Services | Recipe Search |
| New API’s | Grocery list Import API |

**Existing Services:**

**1. Recipe Search & Diet API :**

This API provides services for the following:

* Nutritional Analysis
* Recipe Search

We are using Recipe Search and Diet API to search for a recipe. This API provides information about the ingredients, the preparation time and procedure for a recipe. This API provides detailed information about the quantity of ingredients, total calorie count of the recipe etc. Also this API provides additional search capabilities like we can include restrictions on allergic items etc. Hence we are using this API to get the list of recipes based on a keyword and import the ingredient list.

API: <https://api.edamam.com/search?q=chicken&app_id=51aba909&app_key=9fcd3aa5746d2a423a350cee3ea4d57d> Reference:<https://developer.edamam.com/>

**New Services:**

We have created 2 rest services for implementing the server side logic for this application.

1. **Grocery List**: This API is used to maintain the grocery lists. We have implemented two methods in this API.
   1. Add grocery Item: This method is used to insert the data entered by the user in the database. User name, grocery name, quantity and units are given as a input to this method.

API: [http://kc-sce-cs551-](http://kc-sce-cs551-2.kc.umkc.edu/aspnet_client/MPG12/Sravani/AddGrocery/Service1.svc/addgrocery/Sravani/butter/30/gm)

[2.kc.umkc.edu/aspnet\_client/MPG12/Sravani/AddGrocery/Service1.svc/addgrocery/Sra vani/butter/30/gm](http://kc-sce-cs551-2.kc.umkc.edu/aspnet_client/MPG12/Sravani/AddGrocery/Service1.svc/addgrocery/Sravani/butter/30/gm)

* 1. View grocery list: This method is used to retrieve the grocery list from the database. Username is passed as an input to this method.

API: [http://kc-sce-cs551-](http://kc-sce-cs551-2.kc.umkc.edu/aspnet_client/MPG12/Sravani/viewGrocery/Service1.svc/JsonGrocery/Sravani)

[2.kc.umkc.edu/aspnet\_client/MPG12/Sravani/viewGrocery/Service1.svc/JsonGrocery/S ravani](http://kc-sce-cs551-2.kc.umkc.edu/aspnet_client/MPG12/Sravani/viewGrocery/Service1.svc/JsonGrocery/Sravani)

1. **Import Recipe:** This API is created to import the recipe information from the recipe search API and store the recipe name and ingredient list in the database. We need to import this information to the database so that we can enable the users to customize the recipe.

API: [http://kc-sce-cs551-](http://kc-sce-cs551-2.kc.umkc.edu/aspnet_client/MPG12/vinaya2/RecipeIngredient/Service1.svc/data/)

[2.kc.umkc.edu/aspnet\_client/MPG12/vinaya2/RecipeIngredient/Service1.svc/data/](http://kc-sce-cs551-2.kc.umkc.edu/aspnet_client/MPG12/vinaya2/RecipeIngredient/Service1.svc/data/)

**Use Case Template for Adding grocery items:**

|  |  |
| --- | --- |
| **Use Case Name** | Add grocery items |
| **Primary Actor** | User |
| **Pre-Conditions** | 1. User has the app installed on his/her mobile. 2. User should be logged into the app. |
| **Guarantee(Post Conditions)** | Grocery items are stored in the database |
| **Main Success Scenario** | 1. User clicks on “Add Grocery” button in the home page. 2. The system transitions the UI to the “Add Grocery” layout. 3. User enters the grocery item name, quantity, selects the measuring unit from the drop down menu and clicks on “Add”. 4. System adds the grocery information to the database. 5. System sends a notification that the grocery item is successfully added to the database. 6. User can go back to the home page by clicking on return button. |
| **Extensions** | 3a. User enters incorrect value for quantity. Systems throws an error asking the user to add proper value.  4a. If the grocery item is already present in the list, systems sends an alert saying the given name already exists in the list. The user can update the quantity of the item in the list if desired.  5a. User can add as many groceries items as he needed to by following steps 3 & 4 and then go back to the home page. |

**Use Case Template to View the grocery list:**

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | View grocery list | |
| **Primary Actor** | User | |
| **Pre-Conditions** | 1. User has the app installed on his/her mobile. 2. User should be logged into the app. | |
| **Guarantee(Post Conditions)** | User can view the list of groceries.  User can update the quantity of grocery items. | |
| **Main Success Scenario** | 1. User clicks on “View Grocery List” button in the home page. 2. The system transitions the UI to the “View Grocery” layout. 3. System fetches the grocery list of the user from the database. | |
|  | 4. | System displays the grocery list to the user. |
|  | 5. | User can update the quantity of any of the grocery items. |
|  | 6. | System updates the database to reflect the changes made by the user. |
|  | 7. | System sends a notification saying the update is successful. |
| **Extensions** | None |  |

**Use Case Templates for searching and importing a recipe:**

|  |  |
| --- | --- |
| **Use Case Name** | Search & import a recipe |
| **Primary Actor** | API |
| **Pre-Conditions** | The user must be a valid user to use the app |
| **Post-Conditions** | Retrieves the results for the search |
| **Main Success Scenario** | 1. If the user wants to search for a recipe, then, the user enters the keyword to search. 2. User clicks on the search recipes. 3. The system displays ten recipe names. 4. If the user wants to see the ingredient list, then the user can click on the recipe name. 5. System displays ingredient list along with quantity. 6. If the user wants to import the ingredient list then click on import. 7. The system stores the ingredient list in the database. |
| **Extensions** | None |
| **Special Requirements** | None |

**UML CLASS DIAGRAM:**

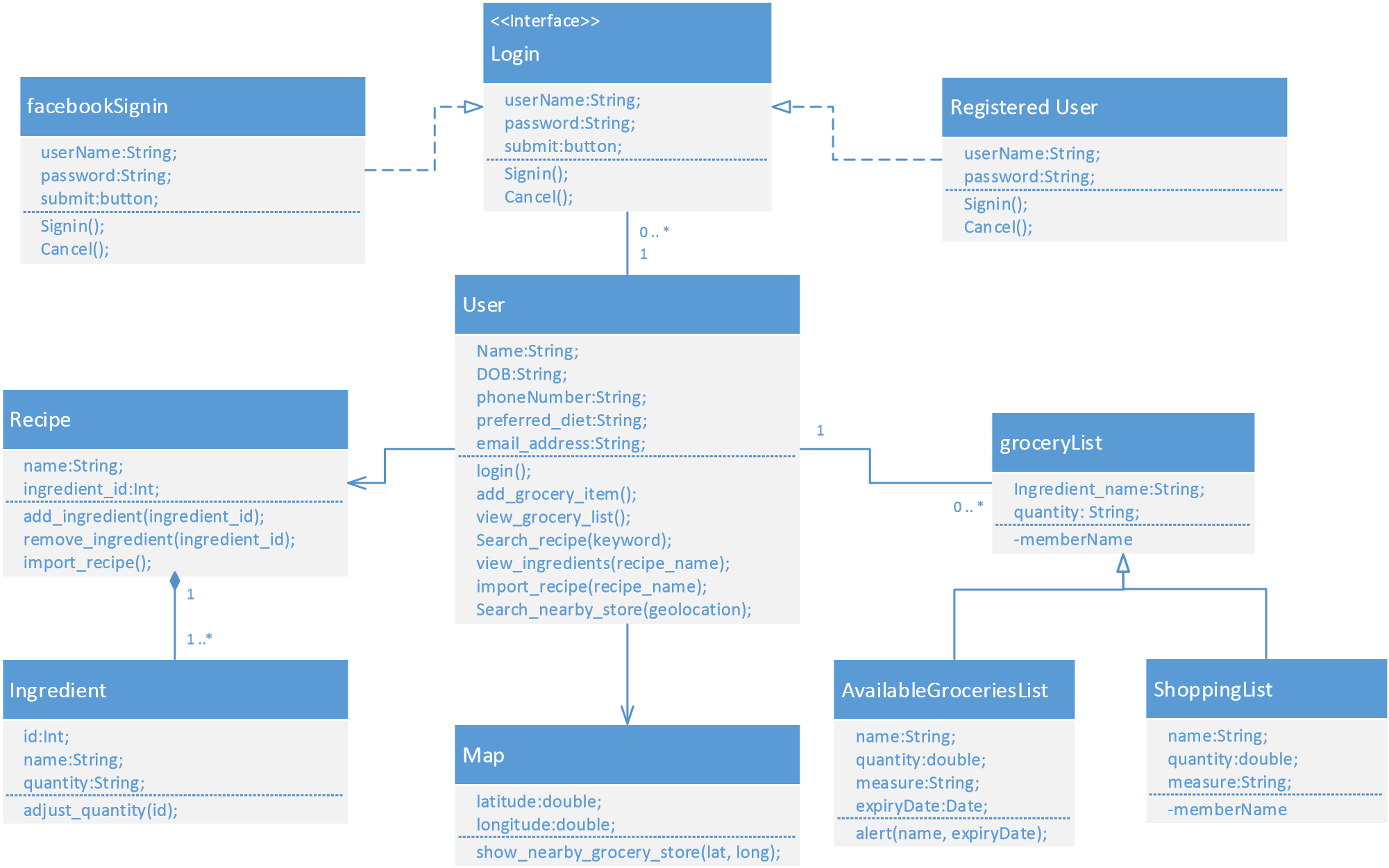
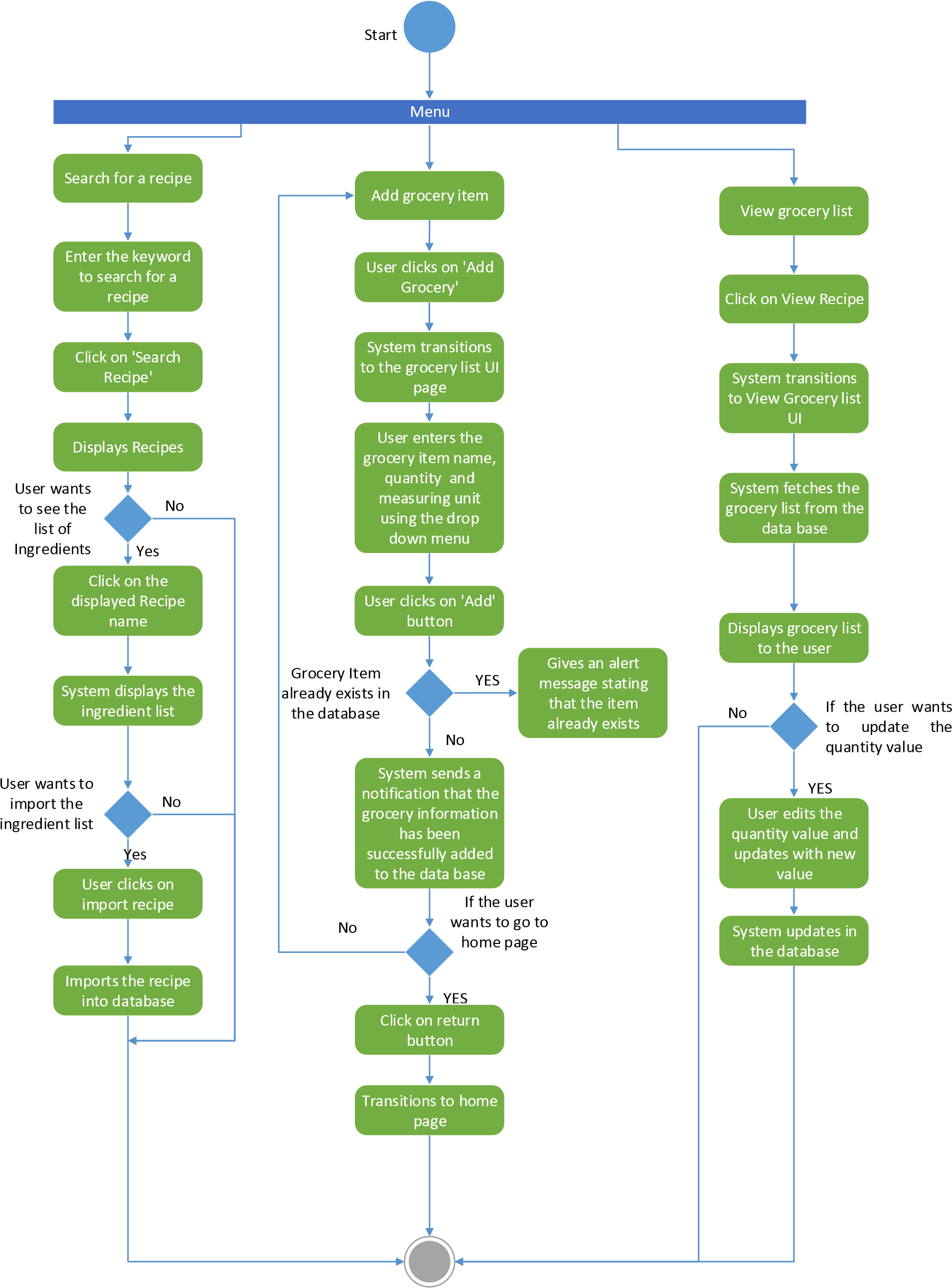


Figure 1: UML Class Diagram of the entire system

**Activity Diagram for the functionality implemented in Increment 2:**



**Sequence Diagram:**

**USER**

**REST API for**

**Remote**

**Database**

**REMOTE**

**DATABASE**

Add items to grocery list

**Recipe**

**Import API**

**Recipe**

**Search API**

items added

Views list of items

Sends the request to the database

Displays the grocery items from the database

Update Quantity of the grocery

Updates the quantity value into the data base

Search a recipe

Returns the recipe list

Select the desired recipe and get the ingredient list

Displays the ingredient list

Import the recipe and ingredient list

Imports the recipe and ingredient ist to the remote database

Figure 3: Sequence Diagram

**Increment#2 Implementation:**

1. **Implementation of User Interface:**

We have developed four layouts as part of UI implementation.

* 1. Home Page
  2. Add Grocery
  3. View Grocery List
  4. Search Recipe

1. **Screenshots:**

# Home Page

User is provided with three buttons:

* Add Grocery Button: User can click this button If he wants to add grocery items to his list.
* View Grocery List: If user wants to view the grocery list, he/she could click on this button.
* Search Recipe button: If user wants to search for a recipe, then he/she could click on this button.

Screenshot 1: Home Page

# Add Grocery

* When the user clicks on Add grocery button, the UI is transitioned to add grocery item page.
* User enters the grocery item name, quantitiy.
* User can select the measuring unit from the drop down menu

**Technical Details:**

* Client side logic is implemented in android studio as a native app.

Screenshot 2: Add Grocery page

# Add Grocery

* After entering the details about the grocery item, user clicks on add.
* System adds the information to the Sql server database.

**Technical Details:**

* We have created a web service for inserting the data entered by the user in the grocery\_list table in the remote database.

Screenshot 3: Add Grocery page

# Add Grocery

* System sends a notification to the user saying the grocery item is added successfully.
* After user clicks ok the add grocery UI is refreshed, the user can add more grocery items or use the home button to go to the home page.

**Technical Details:**

* Web service to add the grocery items to database sends the result of the insert operation in the form of Json.
* Created a class for parsing the Json string.
* Used alert dialog to display the output message from the web service as a notification.

Screen shot 4: Add Grocery notificaition

# Add Grocery

* If the user tries to add the same name again, then the system throws an alert saying “a grocery item with the same name already exists in the list.

**Technical Details:**

* Before insert, we are implemented a logic to check if a record is already present in the grocery\_list table with the same name.

Screen shot 5: Add Grocery update alert

# View Grocery List

* If the user clicks on “View grocery list” button in home page then the UI transitions to this page.

**Technical Details:**

* Implemented the functionality for displaying the grocery list using web technologies.
* Used Web view to display the page.

Screen shot 6: View Grocery list

# View Grocery List

* Once the user clicks on “Grocery List” button, the web page is loaded into the web view.

**Technical Details:**

* Used JQuery to implement the client side logic.

Screen shot 7: View Grocery list

# View Grocery List

* Once the user clicks on View Grocery List button within the page, the list of groceries is displayed in a table.

**Technical Details:**

* We have created a web service to retrieve data from the remote database
* We have used Ajax function to call the API and display the data
* Used twitter Bootstrap styling for table.

Screen shot 8: View Grocery list

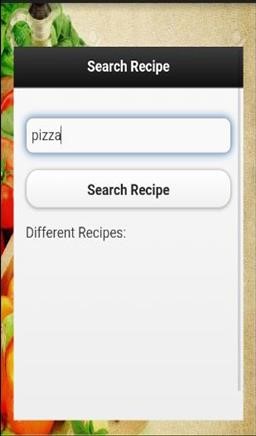
# Search Recipe

* The user gets the search recipe page when he clicks the search recipe button in home page.
* The user can search any recipe by giving the keywords in the search bar.

**Technical Details:**

* The client side logic is implemented using web technologies JavaScript, Jquery & Ajax
* Used bootstrap style sheets.

Screenshot 9: Search Recipe



# Search Recipe

* User enters pizza as a search key to get various recipe on pizza
* User clicks the search recipe button

**Technical Details:**

* We are using the recipe & diet api to fetch the recipe information based on

keyword search

Screenshot 10: Search Recipe

# Search Recipe

 System displays the list of recipes for pizza.

Screenshot 11: Recipe List

# Search Recipe

* If user wants to view the ingredients required for preparing a recipe, user clicks on recipe name
* System displays the ingredient list for the recipe.
* If the user wants to import the recipe, user clicks on “Import Recipe” button
* System loads the recipe & ingredients details to the database.

Screenshot 12: Ingredient List

**3) Database Implementation**

For this increment we created a database “calorie2grocery” in SQL server database in remote server. In this database we created three tables. They are:

* **User\_profile**

This table is used to store the user information like user\_id, username, password, email, phone number and food preference.

* **Grocery\_list**

This table is used to store the list of grocery items that are available at users home. In this table user can save the grocery item along with the quantity.

* **Recipe**

This table stores the information regarding the recipe and the ingredient list when the user searches for a recipe and wants to import the ingredients and recipe.

**User\_profile**

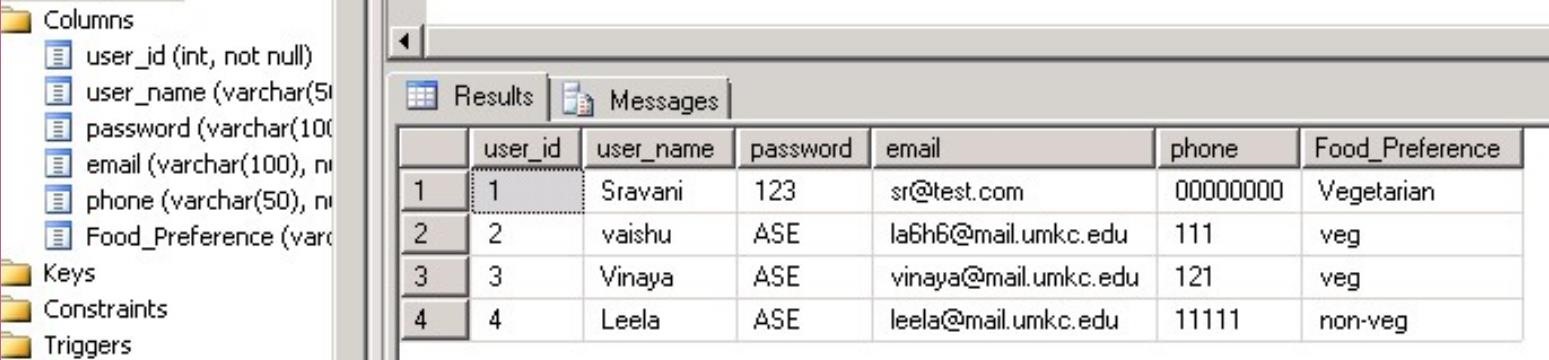
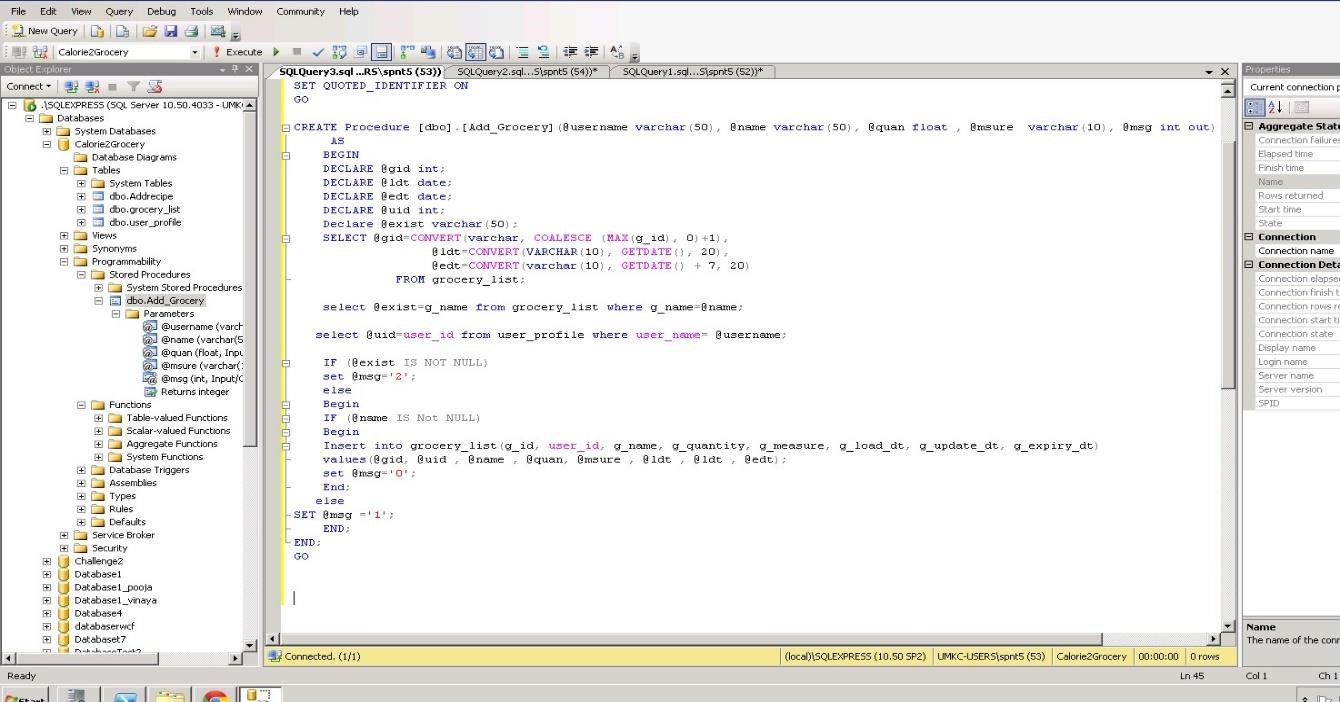


Fig: User \_profiles table containing user data

**Grocery\_list Insert Procedure:**



The above screenshot shows the logic which is used to validate the grocery items before inserting into the database. If the grocery item is already present in the database, it will give an output message stating that particular grocery item already exists in the data base.

**Grocery List table:**

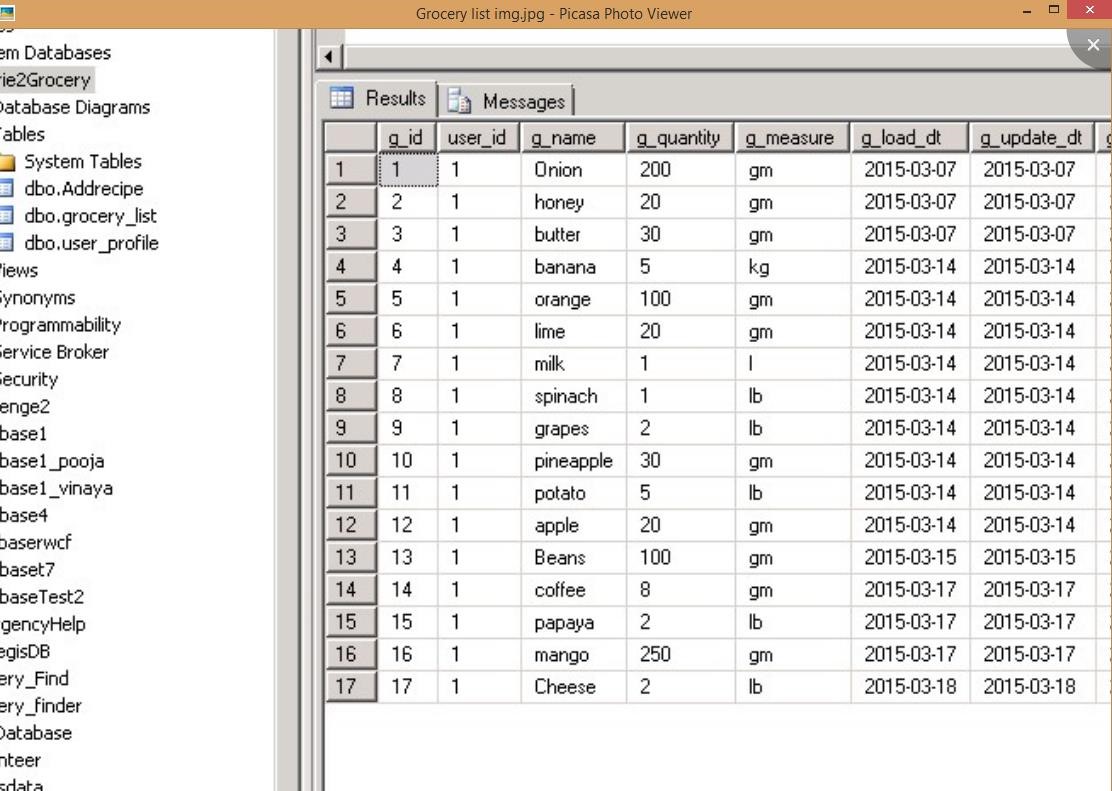


Fig : Table containing the grocery item information

**Recipe**

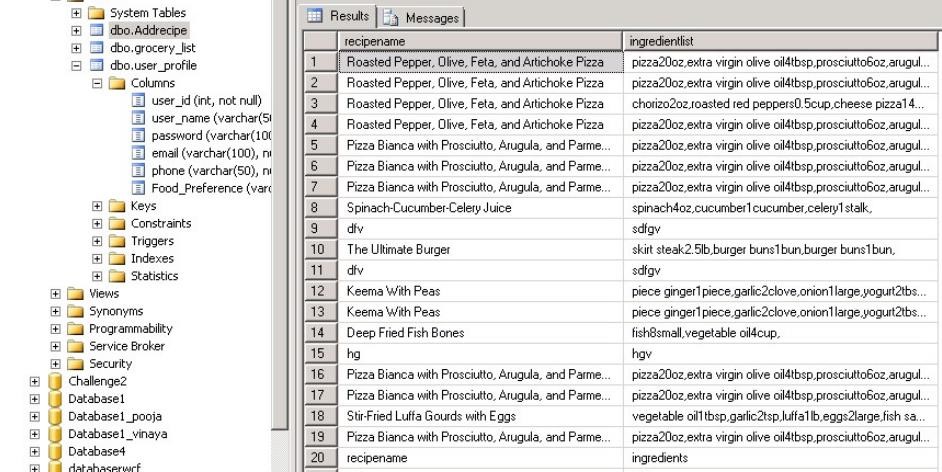


Fig: Table containing the imported recipe information

1. **Design Of Unit Test Cases**

**UI Testing:**

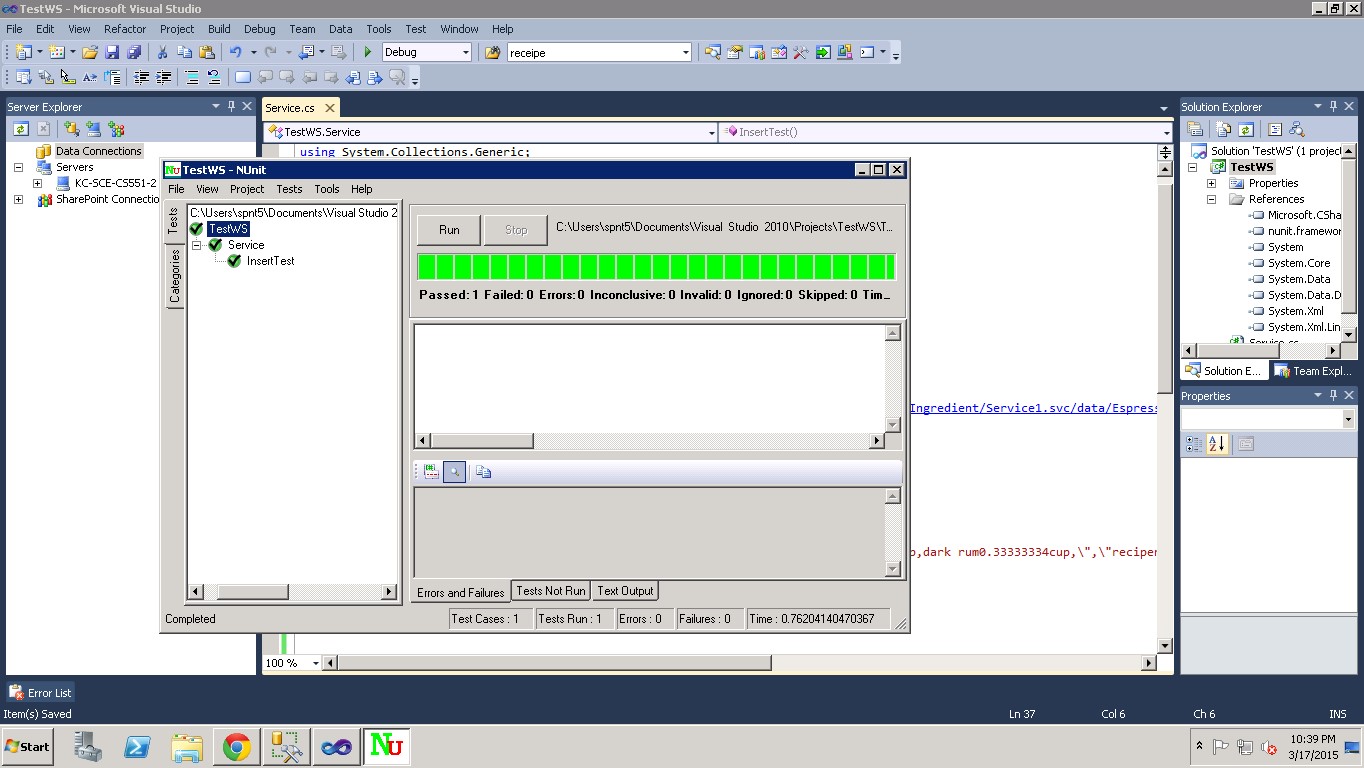
The following test cases has been written for UI testing and the actual output is validated with expected output. All of the below test cases which have been written obtained positive results.

|  |  |  |  |
| --- | --- | --- | --- |
| **Page** | **Button** | **Expected** | **Actual** |
| Home | Search Recipes | Open browser for search recipe | Open browser for search recipe |
|  | View Groceries list | Display the My groceries page | Displays the My groceries page |
|  | Add Groceries | Display the Add groceries page | Display the Add groceries page |
| Search recipe | Browse for a recipe | Should display list of the different recipes | Displays the list of  different recipes |
|  | Select a particular recipe | Should display the ingredients list for the  selected recipe | Displays the ingredients list for the selected recipe |
|  | Import recipe | The ingredients list of the selected recipe should be added to the remote database | The ingredients list of the selected recipe is added to the remote database |
| My Groceries | Add groceries | The list of items should be added to the database | The list of items are added to the database |
|  | View my grocery list | Should display the list of grocery items present | Displays the list of grocery items present |
|  | Update the quantities of an item | Should allow us to update the quantity of an item | Allowing us to update the quantity of an item |

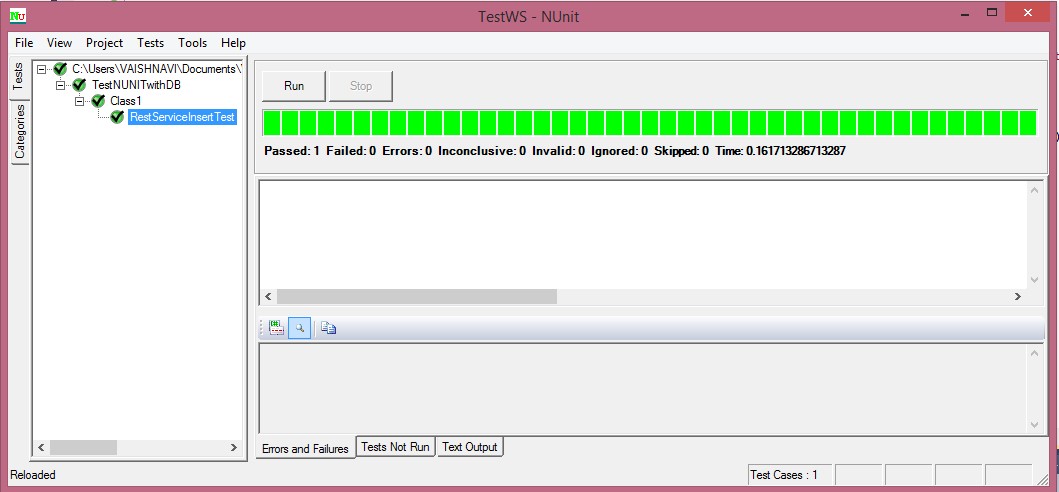
**Data Validation:**

We have implemented N-Unit test cases to validate the data for all the newly created web services. We have used Assert methods to compare the actual output with the expected output.

**Screenshot of N-Unit validation: This is used to validate the data for Import recipe API:**



**Screenshot for N-Unit validation of Add groceries API:**



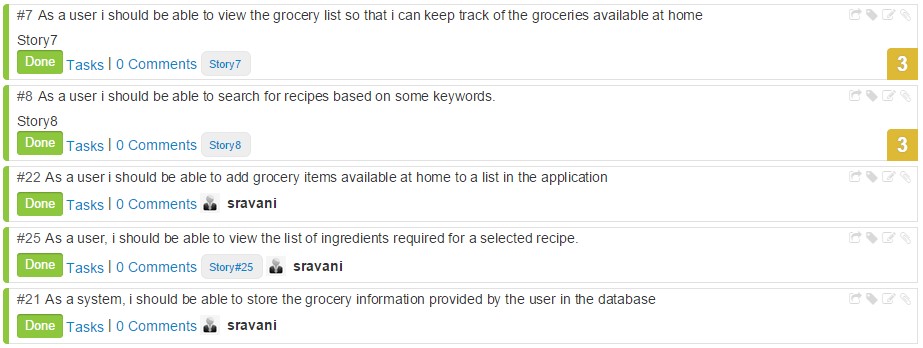
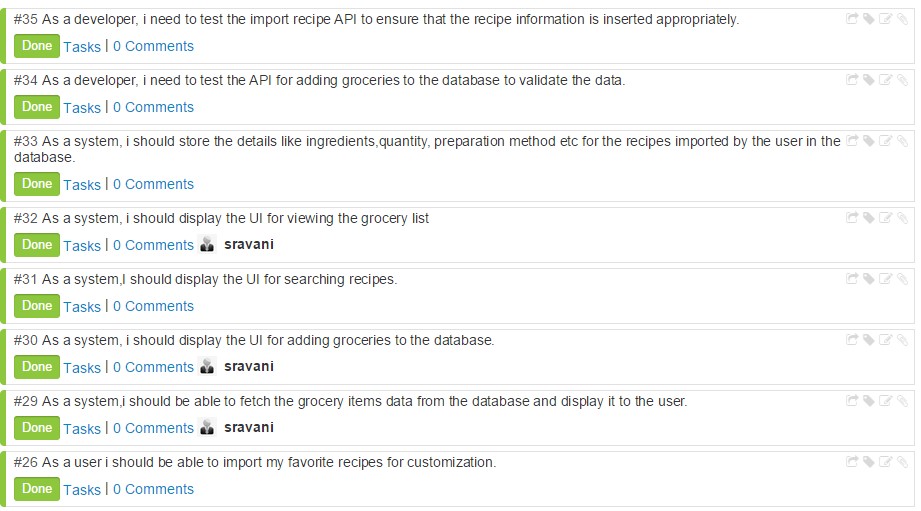
1. **ScrumDo**

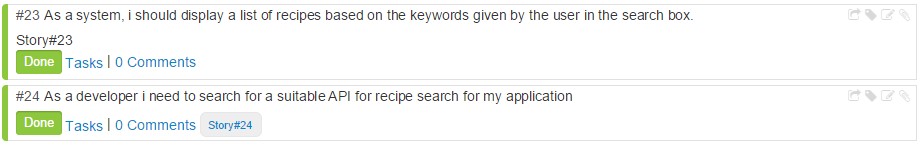
We have used Scrumdo to track the progress of the project. The project development is divided into four increments. All the functions and development tasks are written as user stories.

Link: [https://www.scrumdo.com/projects/project/calorie2grocery3/iteration/122931#](https://www.scrumdo.com/projects/project/calorie2grocery3/iteration/122931)

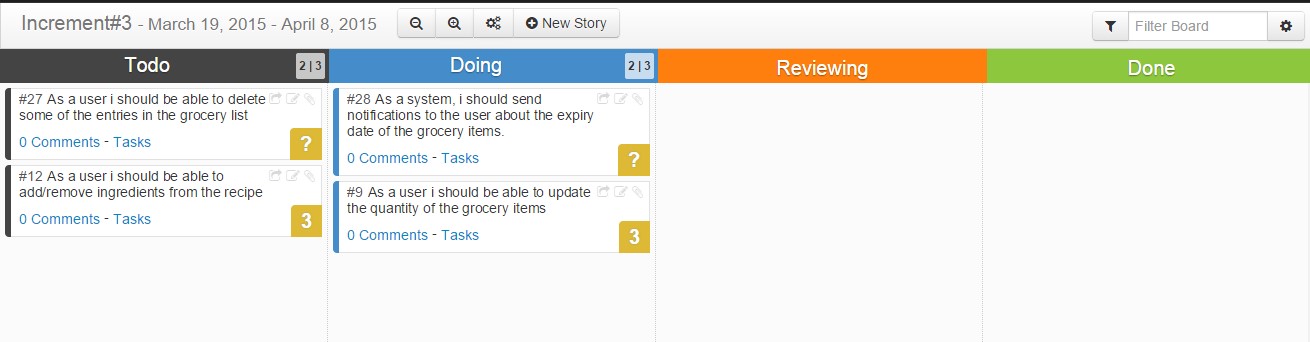
**Screenshots for Increment#2 user stories:**

All the user stories planned for increment #2 have been successfully implemented.

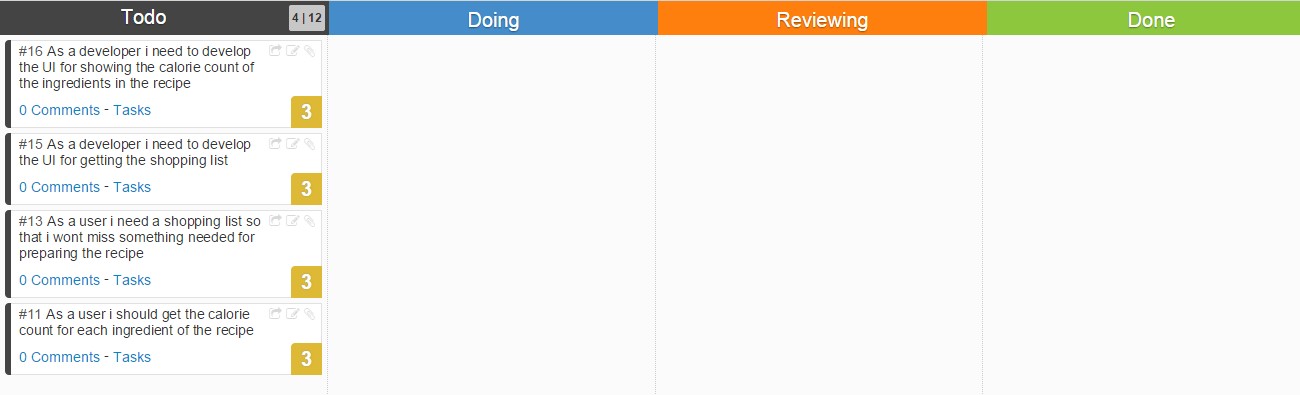




**Increment #3 user stories**



**Increment #4 user stories**



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **User Story#** | **User Story** | **Hours Worked** | **Team Member** | **Status** |
| 30 | As a system, I should create the UI for adding groceries to the database. | 3 | Sravani | Done |
| 22 | As a user I should be able to add grocery items available at home to a list in the application | 8 | Sravani | Done |
| 21 | As a system, I should be able to store the grocery information provided by the user in the database | 8 | Sravani | Done |
| 24 | As a developer I need to search for a suitable API for recipe search functionality in my application | 10 | Vaishnavi, Vinaya | Done |
| 31 | As a system, I should display the UI for searching recipes | 8 | Vinaya, Vaishnavi | Done |
| 8 | As a user I should be able to search for recipes based on some keywords. | 12 | Vinaya, Vaishnavi | Done |
| 23 | As a system, I should display a list of recipes based on the keywords given by the user in the search box. | 10 | Vinaya, Vaishnavi | Done |
| 25 | As a user, I should be able to view the list of ingredients required for a selected recipe. | 4 | Sravani | Done |
| 26 | As a user I should be able to import my favorite recipes for customization. | 22 | Leela | Done |
| 33 | As a system, I should store the details like ingredients, quantity, and preparation method etc. for the recipes imported by the user in the database. | 20 | Leela | Done |
| 32 | As a system, I should display the UI for viewing the grocery list | 2 | Sravani | Done |
| 7 | As a user I should be able to view the grocery list so that I can keep track of the groceries available at home | 5 | Sravani | Done |
| 29 | As a system, I should be able to fetch the grocery items data from the database and display it to the user. | 12 | Sravani, Vaishnavi | Done |
| 34 | As a developer, I need to test the add groceries API to validate the data inserted into the database | 2 | Vaishnavi | Done |
| 35 | As a developer , I need to test the import recipe api to ensure that the recipe information is inserted appropriately | 2 | Vinaya | Done |

**GIT Hub URL:**

The Project code and documentation are uploaded to the following git hub URL:

<https://github.com/vaishnavi5054/AdvSoftEng>

**Bibliography:**

1. Reference: <http://api.bigoven.com/>

API: http://api.bigoven.com/recipes/47725api\_key=dvx30p6vcIMHZWh3G1mghS88YvV6140D This API provides two functions:

* + Get Recipe
  + Search Recipe

This is a REST based API. This API helps to search the recipes, display the list of ingredients. It redirects to another webpage where the user can see the recipe reviews etc… It also allows to update the grocery lists in the cloud. This supports either XML or JSON formats.

1. Reference: [https://www.recipal.com](https://www.recipal.com/) API: <https://recipal.com/api/v1/recipes/522>

This API helps to search and retrieve the recipes according to the scale. We can update an ingredient into the ingredient list. It also allows the user to create the customer ingredient object. The user can create a new recipe. Update an existing recipe etc…

1. Reference:<https://developer.edamam.com/>

API:https://api.edamam.com/search?q=chicken&app\_id=51aba909&app\_key=9fcd3aa5746d2a 423a350cee3ea4d57d

This API provides services for the following:

* + Nutritional Analysis API
  + Recipe Search and Diet API

Here we are using Recipe Search and Diet API to search for a recipe. This API gives retrieves more information about the ingredients, the preparation time and procedure for that recipe. So, this API is more useful for our project as it gives the detailed information which helps us to import the ingredients. With the help of this API we can filter more while searching for a recipe like we can give allergic restrictions etc.