

**Software Engineering II Project**

**Digital Cookbook**

Part task: 1. Login & Logout & Registration

2. PDF Export

3. like it – collection of favorite dishes

4. Search by ingredients

**Group Three “Are You Hungry“**

Hua Yichen

Wang Jungang

Shan Jiaxiang

Kong Yu

**Content**

[**1.** **Specification** 3](#_Toc450063039)

[**1.1.** **Description** 3](#_Toc450063040)

[**1.1.1.** **Cleaning help** **错误!未定义书签。**](#_Toc450063041)

[**1.1.2.** **Additional Task** 3](#_Toc450063042)

[**1.2.** **Product functions** 3](#_Toc450063043)

[**1.3.** **User characteristics** 5](#_Toc450063044)

[**1.4.** **Functional requirements** 5](#_Toc450063045)

[**1.5.** **Non-functional requirements** 6](#_Toc450063046)

[**2.** **UML Specification** 8](#_Toc450063047)

[**2.1.** **Use Cases** 8](#_Toc450063048)

[**2.2.** **Class Diagram** 10](#_Toc450063049)

[**2.3.** **E-R Diagram** 10](#_Toc450063049)

[**3.** **GUI Design** 11](#_Toc450063050)

[**3.1.** **Structure** 11](#_Toc450063051)

[**3.2.** **Screenshots** 11](#_Toc450063052)

[**4.** **Test** 12](#_Toc450063053)

[**4.1.** **Description** 12](#_Toc450063054)

[**4.2.** **Results** 12](#_Toc450063055)

[**5.** **Evaluation** 13](#_Toc450063056)

[**5.1.** **Group Work** 13](#_Toc450063057)

[**5.2.** **Task Responsibilities** 13](#_Toc450063058)

# **Specification**

# **Description**

# **Digital cookbook**

Digital cookbook is a software in which users can read recipes. With this software, users will be convenient to find solutions on how to make specific dishes.

A user can browse the whole list of recipes without any additional requirements. For those conservative users who prefer papery media, the software provides “Export to PDF” function. Also, if a user register for an account, he can add recipes to his favourites. When users have ingredients but only have a vapour idea on what dishes they can cook, the “search by ingredients” function can help them find which to choose.

# **Additional Task**

***➢ Export PDF file***

Users can export one recipe to a PDF file.

***➢ Search for ingredients***

Users can search recipes by the ingredients they selected.

***➢ Register and log in***

Users can register for their own account. With an account, users can add recipes to their favorites.

***➢ Add to favorites***

Users can add recipes to their favorites. The favorited recipes can be found in “My Favorites”.

# **Product functions**

Details of functions of cookbook are as follows:

When we open this system, this application will show a welcome page. After entering the correct username and password, the home page will appear with a search bar on the top and a list of brief description and details of all recipes at the button. Clicking on one recipe, the detailed information of this recipe will show in the right interface.

**➢ *PDF Exporting***

Users can extract the basic information of specific recipe - in form of PDF file - by clicking “PDF Exporting” button in the User Interface, through which the recipe can be shared to others in the Internet or be printed out so that the user can read it while cooking.

***➢ Searching Recipe***

Users can search recipes they want by offering key words regarding the basic information of the recipe (e.g. name of recipe or ingredients etc.) to the “Search Area” offered in the User Interface. The correct results will be shown in the area below the “Search Area”.

***➢ Editing/Adding Recipe***

Users can edit or add recipe to our system. For every recipe there exists a “Edit” button, through which the users can feel free to edit on the information of the recipe (e.g. procedures, ingredients or upload an image etc.). In our User Interface we also offer an “Add” button, through which users can add new recipes by providing basic information of a recipe.

***➢ Delete Recipe***

We offer in each recipe a “Delete” button, through which the user can delete the recipe whenever they find it totally wrong or out-of-date. In case of data security, all the deletion are logical.

***➢ Login & Logout & Registration***

For using our digital cookbook, the user must register. During the registration, the user must fill out the form about basic information like username and password. For every time the user want to search in the cookbook, they should login the system first. After finishing using the system, the user should logout.

***➢ “Like it” Button***

For every recipe the users can mark it as “Favorite” by clicking the heart icon in the left side, which mean the recipe is marked as favorite. The favorited recipes can be found in the favorite folder.

# **User characteristics**

***➢ Age***

The age range we target at is from 16 to 60. It is important that users are able to operate some cooking machines or tools. People who are too young or too old may face danger when operating.

***➢ Language and culture***

Users should be able to read in English because the contents are written in English. Besides, this application is friendly to users from different food cultures, because it will include many different kinds of recipes.

***➢ Motivations***

Users with great enthusiasm for cooking will find usability in this application.

***➢ Related skills***

Users must have basic computer operating skills to operate this application. Besides, basic cooking skills are expected because these skills won't be included in the digital cook book.

***➢Device compatibility***

User’s electronic device should be able to run JAVA programs otherwise this application won't work.

# **Functional requirements**

***➢ Welcoming & Login Panel and Tutorials***

The system should provide user with a welcome page, showing the name of our system, welcome information, a brief introduction to our application & login panel (username and password) and a start button to enter our application. Meanwhile, the system should also provide user with a registration button, when user click the button, a new page consists of information form will exist for users to filling in.

***➢ Page Shifting Buttons***

The system should provide user a column containing page shifting buttons which allow users to shift among different views.

***➢ Brief Recipe View***

The system should provide user a column space for displaying brief information of recipes like boxes piling up. This column should be next to the main recipe view and with clicking on the boxes, the main recipe view shall show the detailed information of the selected recipe.

***➢ Search Area and Results***

The system should provide user a search bar to type in the keywords (on Ingredients or Name of Recipes) and can use buttons to shift the search function between search by recipe name and search by ingredient name. The system should also provide user the view for search results which shares the similar design with the main recipe view and the brief recipe column.

***➢ Favorite Recipe – “Like It” Button***

The system should provide user functions to add a recipe into or remove the recipe from the favorite folder. The system should provide user a “Like it” button to click on. When user click the button, the recipe is sent to The Favorite Folder of the user as a collection of user’s favorite recipes.

***➢ Adding & Editing and Deleting Recipes***

The system should provide user functions to add a new recipe and edit existing ones. The User Interface should provide user a page for adding a new recipe or editing an existing one. The new page shows a form consists of basic information of the recipe. There should be functions to save the new recipes and save the changes on existing recipes. Moreover, the system should provide users with deletion button to delete a recipe logically.

**➢ *PDF Exporting***

The system should provide users with the ability of extracting the basic information of specific recipe - in form of PDF file - by clicking “PDF Exporting” button in the User Interface, through which the recipe can be shared to others in the Internet or be printed out so that the user can read it while cooking.

# **Non-functional requirements**

***➢ Accessibility***

The application should be free of charge or partly free so that everyone can download it from the Internet. The interface should be easy to read which using simple buttons and bars. Time used to learn how to use this application should be less than half an hour.

***➢ Adaptability***

The application should be able to run on Java Runtime Environment 8 (JRE 8) or higher on different operating systems (e.g. Windows, Mac OS, Android, etc.).

***➢ Data management and security***

The application should avoid illegal modification of internal files at any time. It should also be able to maintain a normal running status of the database. The application should also have the visibility of program progress while downloading or uploading data.

***➢ Failure management***

The application should be able to work continuously for at least 12 hours without any errors. In the case that unexpected failure occurs, the application itself should be able to inform the user what exact kind of error occurred and then try to recover to the previous normal status.

***➢ Legal and licensing issues***

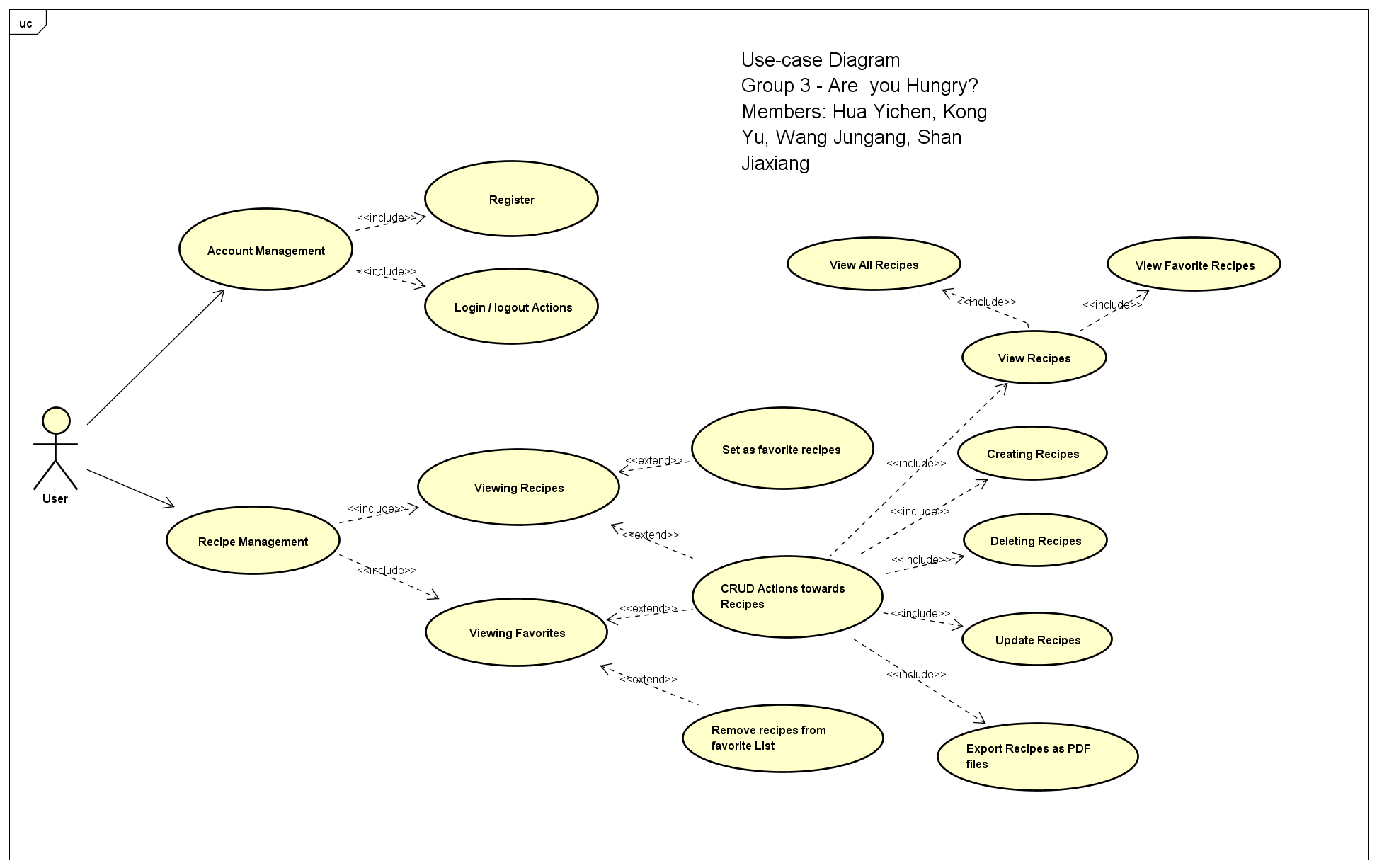
The application should not contravene local laws and regulations, especially the copyright problems should be avoided.

***➢ Maintainability***

The application should be easy to update or fix critical coding problems through legal access (usually by the programmer). The internal code should not be visible to everyone, but it should be visible to all programmers in the team. The code should be written according to the Java code conventions.

# **UML Specification**

# **Use Cases**

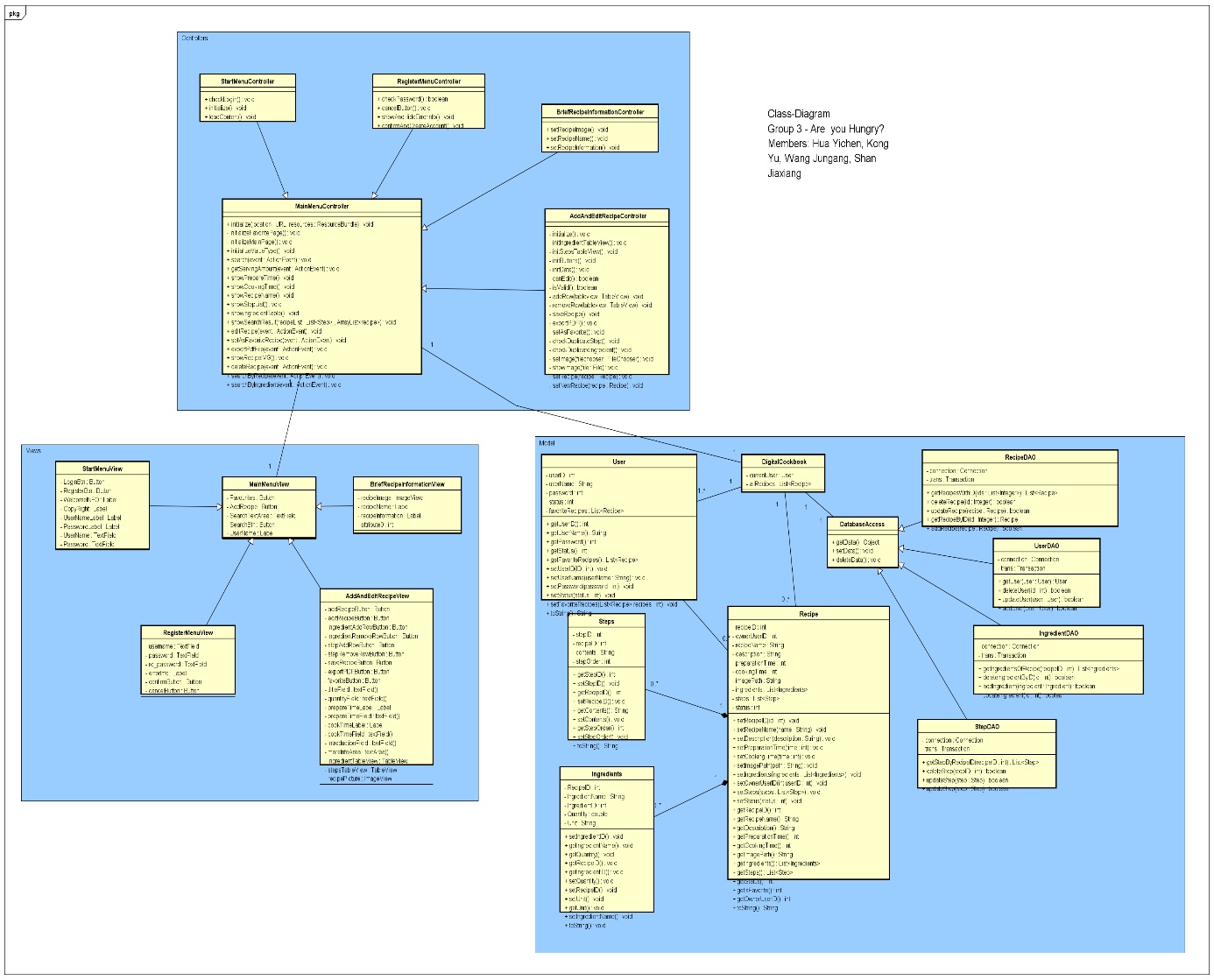


As shown in the Use-Case diagram above, there are eleven use cases for the digital cookbook, which are explained in detail below:

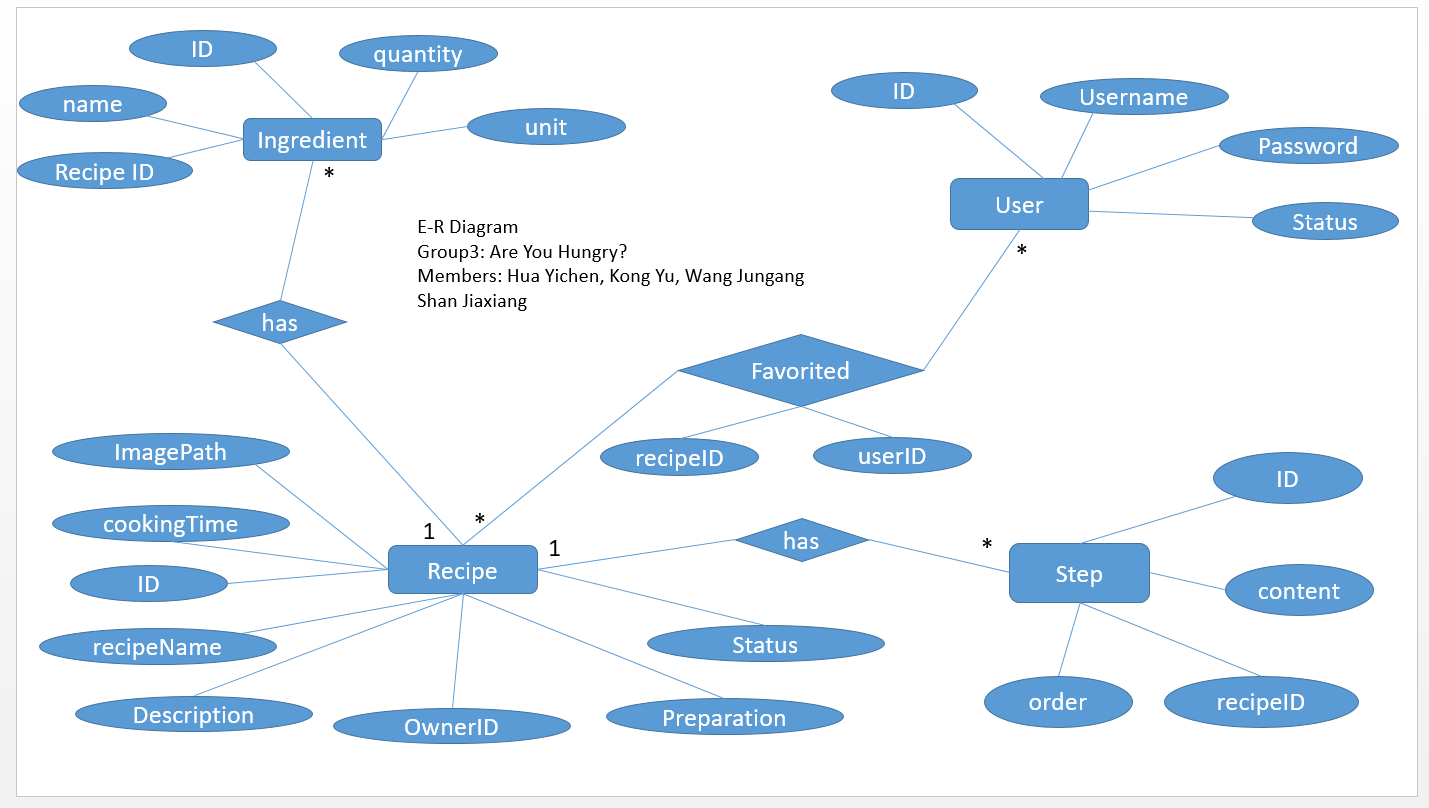
We classify function of our system into two main categories: Account Management and Recipe Management.

1. Account Management – mainly involves with actions towards accounts.
2. When users open the digital cook book for the first time, they will be requested to register for a new account. Otherwise they cannot use the digital cook book.
3. After registration the page will switch to a log-in button, then users can use the registered account to log in. And they will get access to all the recipes. If the users want to log out, they can click on Account button, and choose to log out.
4. Recipe Management – mainly involves with actions towards recipes.
5. When the users have logged in, they can view all the recipes in the page. If they want to view a specific recipe or ingredient they can search what they want.
6. The digital cook book provide the users with the functionality of setting favorite recipes. The users can click on Favorites button to view their favorite recipes.
7. The user can set current recipe as favorite recipe when he/she is viewing the recipe by clicking the "favorite" button. The button will only appear in the recipe screen.
8. The user can set current recipe off favorite recipe when he/she is viewing a favorited recipe by clicking the "favorite" button again. The button will only appear in the recipe screen.
9. The user can do CRUD actions towards the recipes as follows:
10. When the user has logged in, he will be able to create a new recipe by click the “Create a recipe yourself!” button on the main page. A text field will be provided for writing.
11. The user can delete recipes created by himself by clicking the button “Delete” on the recipe page. The button will only be valid for the self-created recipes.
12. The user can update their own recipes by clicking the button “Update”. The button will only be valid for the self-created recipes.
13. The user can update their own recipes by clicking the button “Update”. The button will only be valid for the self-created recipes.
14. The user can export the recipes into PDF file by clicking the “Export to PDF” button.

# **Class Diagrams**

****

# **E-R Diagrams**



# **GUI Design**

# **Structure**

In our digital cookbook, the MVC model is mainly used for designing the UI. The View consists of four user interfaces, which are realized by the FXML codes generated in the Scene Builder and their further load by the FXML Loader in Java; The Controllers, basically Java Objects implements the interface Initializable, consist of Java attributes and methods that regulate the actions performed by the JavaFX elements (e.g. Button, TextArea and etc.); The Models consist of Data Access Object, entities objects and basic Utilities. With applying the MVC model, we do not have to change codes of the other two components while we are altering one of the components. Next, I will show each part of MVC in detail.

As for View, we have four FXML files, auto-generated by Scene Builder, which are “*Template.fxml”, “MainRecipeFrame.fxml”, “AddOrEditRecipe.fxml” and “BriefRecipeInMainPageView.fxml”.* *MainRecipeFrame* is for searching, presenting recipe information and “Like it” function, and *AddOrEditRecipe* is for editing or adding a new recipe. Moreover, *Template* is mainly responsible for User login and registration – we have designed two Input area for users to input the basic information of their accounts and two buttons to confirm login and registration.

As for controllers, for every view we bind a controller with it. So there are four controllers in our project and we name them as *“TemplateController”, “MainFrameController”, “AddOrEditRecipeController” and “BriefRecipeInformationController”*. Each controller is responsible for getting users’ inputs with GUI, and pass the parameter generated by users’ interaction to the methods, exists in Model, which can query the database to return results. Then controllers send those result data back to the GUI and the whole procedures realize what we called the Interaction.

As for Model, it consists of Entities and Data Access Objects. There are five classes are constructed to represent the entities of our software. They are “*User.java”, “Cookbook.java”, “Recipe.java”, “Ingredient.java” and “Step.java”.* The Entities classes are responsible for data storage in the memory, you can also see the term ORM for Object Relation Mapping, holding attributes that are consistent with the tables in the database. For instance, when showing all the recipes we have in our system, the system firstly executes query and fetch information of recipes, corresponding with their steps and ingredients, from the database, then load data from the Result Set into Entities for further use. Hence, entities classes will help us store the data in a more structured way. Moreover, for DAO as Data Access Objects, they are mainly responsible for data fetch and storage in the database, we divide DAOs into five categories, according their serving targets, *“IngredientsDAO.java”, “RecipeDAO.java”, “StepDAO.java”, “UserDAO.java” and “BaseDAO.java”. “IngredientsDAO.java”, “RecipeDAO.java”, “StepDAO.java”, “UserDAO”* are responsible for CRUD operations with their corresponding entities classes and “*BaseDAO”* defines basic connection properties toward database.

In summary, since it is not a large application, using MVC model takes us extra efforts to follow such a structure. But after applying this model, we actually feel the flexibility of using it. Since during coding period, we can construct (working on) model, view and constructor simultaneously. This model separates not only the tasks between different layers but also separate workload of each contributors by allowing them to focus only on one part of model. And when bugs happen, this model help us to locate which part of model breaks, and then able to inform the responsible person to fix the bug.

# **Screenshots**

# **Test**

# **Description**

# **Results**

# **Evaluation**

# **Group Work**

# **Task Responsibilities**