Software Engineering II Project

Digital Cookbook

“Recipe Box”

Part task: Favourites Collection Feature

Group One

Bo Jiao

Hanzhi Zhuang

Hao Yuan

2021.07.02

Content

1. Specification

1.1. Description

1.1.1. Digital cookbook

Digital cookbook “Recipe Box” is a software where users can read, edit or delete stored recipes and add new ones. With this software, it is convenient for users to find the recipe needed for cooking at the moment.

1.1.2. Additional task

Favourites Collection Feature

- User can set a bookmark for a recipe and unset it anytime when visiting a specific recipe page with details
- In the digital cookbook application main page there is the possibility to show a bookmark page by clicking on “Favourites” button. An overview of all bookmarked recipes can be found there. By clicking on a specific recipe, the user will be forwarded to the recipe page with details.

1.2. User characteristics

Age Groups

- The age range we target at is from 16 to 60, since those age groups are the main force for cooking on their own.

Language and culture

- Users should be able to read in English because the software will be written in English.

Motivations

- Both users with great enthusiasm for cooking and ordinary people who make meals for families will enjoy the convenience this application brings.

Related skills

- Users must have basic computer operating skills to operate this application. Besides, basic cooking skills are expected.

Device compatibility

- User's electronic device should be able to run JAVA programs as this application will be written in JAVA. Otherwise, this application won't work.

1.3. Data

Data that should be stored during the use of this application are show below:

- Recipe
  - id
  - recipe name
  - preparation time of the recipe
  - cooking time of the recipe
  - amount of people can be served
  - instruction steps of the recipe
  - favourited or not
- Ingredients
  - id
  - ingredient name
  - quantity needed
  - unit of the quantity
  - pre-treatment of the ingredient
- Pictures

1.4. Functional requirements

- Digital Cookbook is a desktop software that displays stored recipes and allows its user to manage (create, read, update and delete) the recipes.

    - The application should provide user a column space for displaying brief information of recipes. This column should be next to the preview picture of the recipe and with clicking on a specific recipe, the main recipe view shall show the detailed information of the selected recipe.

    - The application 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.

- Digital Cookbook stores recipe information including: Recipe Name, Preparation Time, Cook Time, Number of People Served, Ingredients (in terms of Quantity, Unit, Name and Pre-treatment), Instructions.(See Data part above)

- If the user changes the default Number of People Served, the Digital Cookbook should be able to calculate and display the corresponding Ingredient Quantity.

- The Digital Cookbook should allow its user to search for a recipe by Name.

    - The system should provide user a search bar to type in the name of recipes. The system should also provide user the view for search results which shares the similar design with the main recipe view stated above.

- Additional Feature(See additional task above)

1.5. Non-functional requirements

- The application will be written using Java according to Java code conventions and include necessary Javadoc and comments.

- The application should be able to run on Java Runtime Environment 8 or higher versions on different operating systems (e.g. Windows, macOS, Linux, etc.)

- The GUI of the software will be written using Java Swing/FX according to MVC design pattern. The interface should be easy to read which using simple buttons and bars.

- All data of recipes should be safely stored in MySQL database. The application should also be able to maintain a normal running status of the database.

- The software should respond to user actions in less than 5 seconds.

- Time used to learn how to use this application should be less than half an hour. And after a month without using it, the user is expected to remember how to use the application again.

2. Architecture

2.1. Use case diagrams

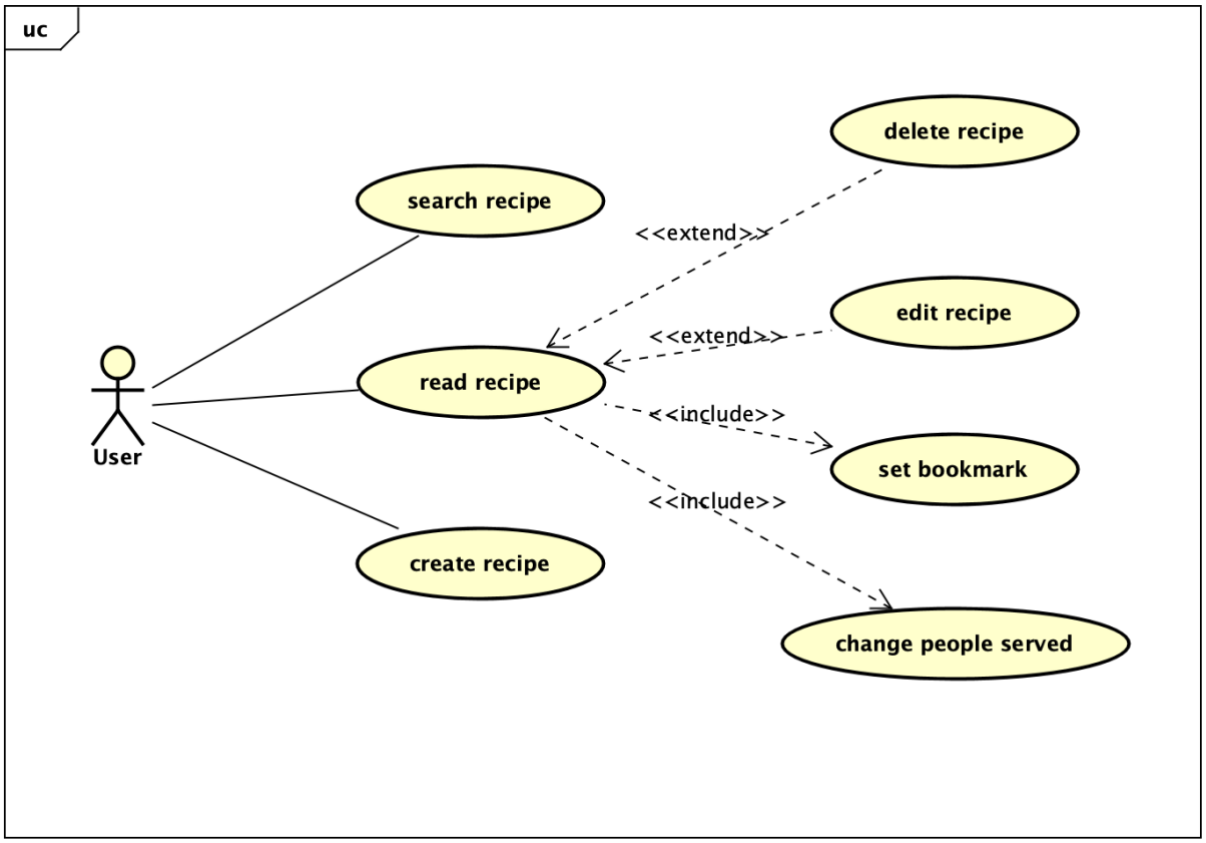


Figure 2-1 Use-case Diagram

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

- When the user opens the application, they can view all the recipes listed in the main page. Also, by clicking “Favourites”, the favourited recipes will be filtered out. In the same page, they can search for a specific recipe by entering its name in the text field and clicking “search” button. They can also create new recipes by clicking “add” button.
- If the user wants to view a specific recipe, they can double click on it to read detailed information. When viewing the details page:

- The user can set/unset current recipe as favorite recipe when he/she is viewing the recipe by clicking the "favorite" button.
- The user can change the default Number of People Served, the Digital Cookbook will calculate and display the corresponding Ingredient Quantity.
- The user can edit or delete a recipe when viewing it in detailed view.
-

2.2. Class diagrams

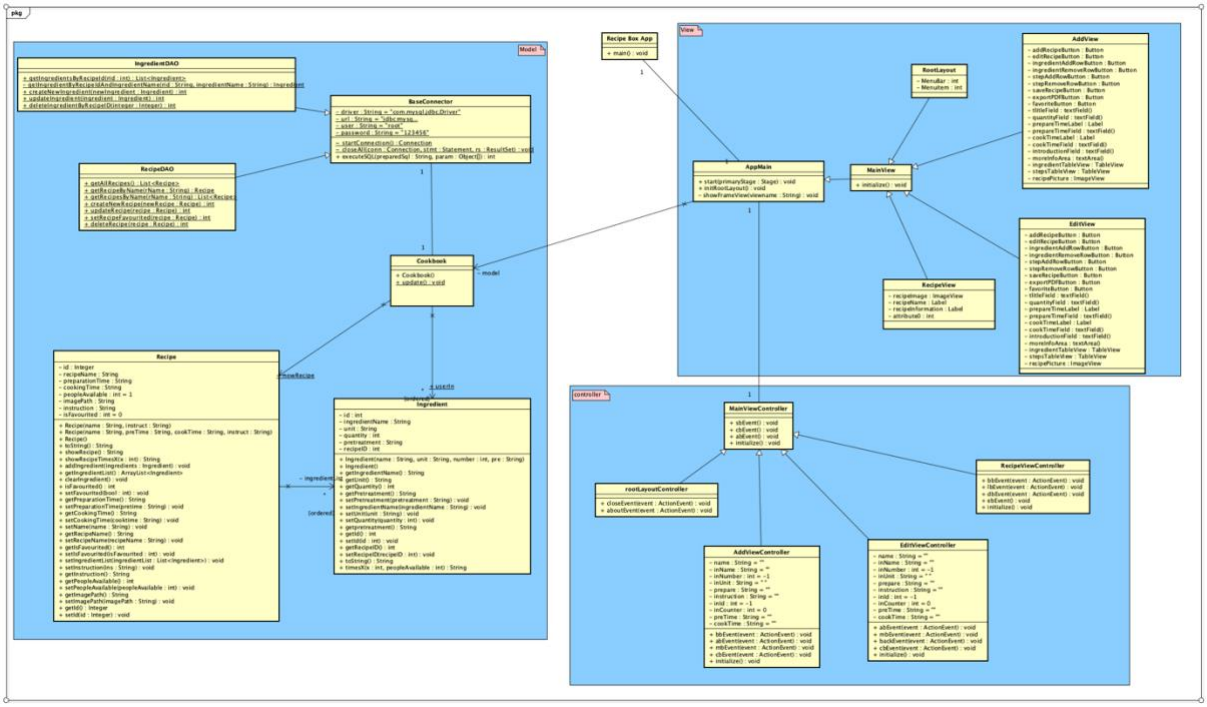


Figure 2-2 Class Diagram

The software of our group is designed basically based on this Class diagram. The software uses MVC (Model, View, and Controller) model as shown in the diagram. It is divided into 4 parts: DAO, Model, View and Controller. This model separates the whole task of the software into 3 minor tasks which means each part only need to focus on its own task without considering other parts' job. This helps a lot when programming, because it has a clear structure. Also, we can just contribute work to each member simply according to the MVC model.

2.3. Structure/MVC

The structure (MVC) is shown below:

- Model
  Model is basically the entity classes: Recipe, Ingredient and Cookbook. Those are related to data processing.
  - DAO (Part of Model that have been extracted)
    DAO is responsible for connecting the database.
- View
  We use FXML to implement the view part. This is mainly for rendering graphical user interface.
- Controller
  Controller defines software reactions to user input or actions.

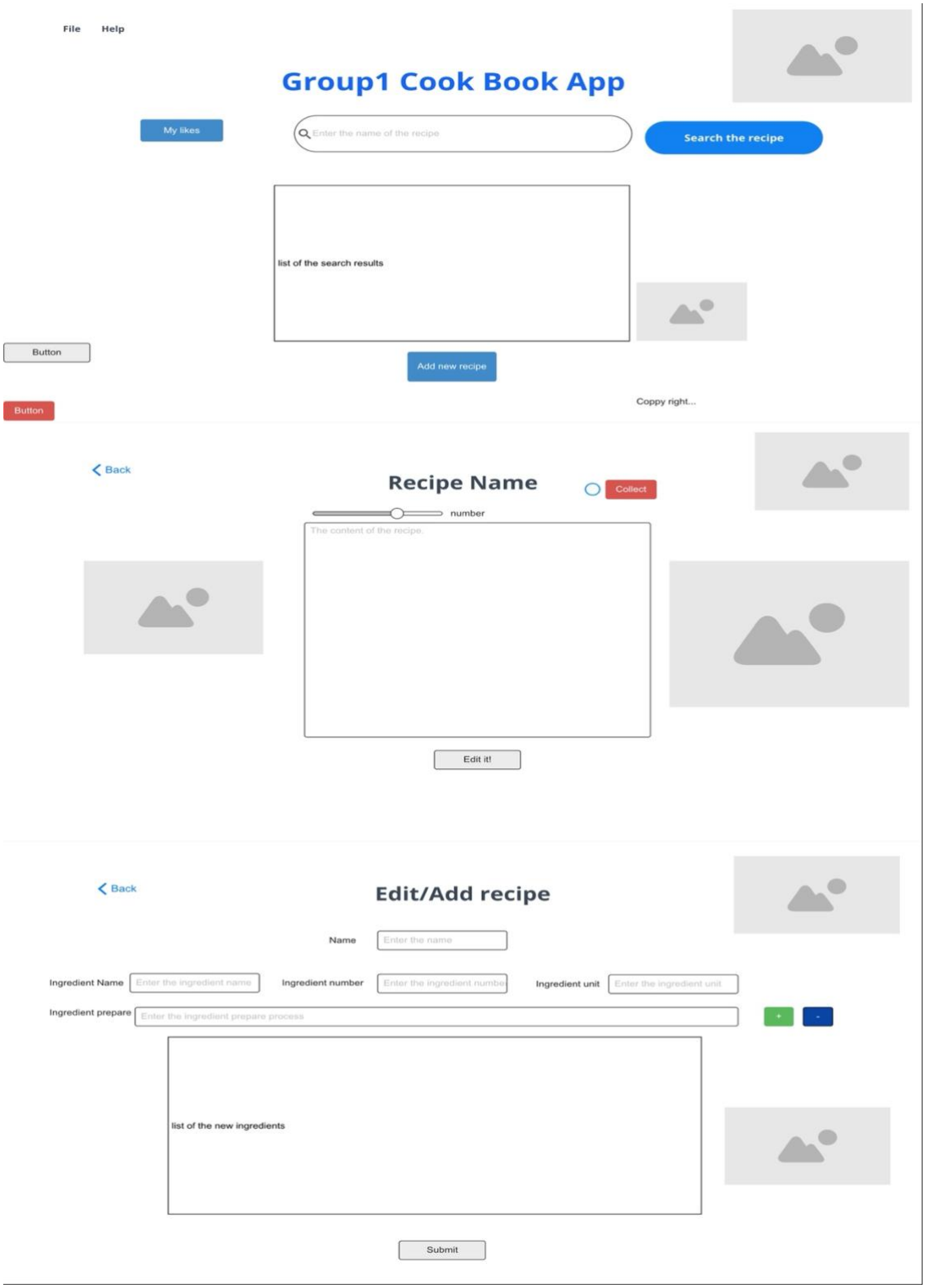
In our digital cookbook, the MVC model is mainly used for designing the UI. The View consists of four user interfaces, which are realized by the FXML codes generated in the Scene Builder and their further load by the FXML Loader in Java; The Controllers, basically Java Objects implements the interface 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 and entities objects. 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.

In general, by applying this MVC model, we really 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 programmer by allowing them to focus only on one part of model. And when bugs occur, this model help us to locate which part of model breaks, and then able to inform the responsible person to fix the bug.

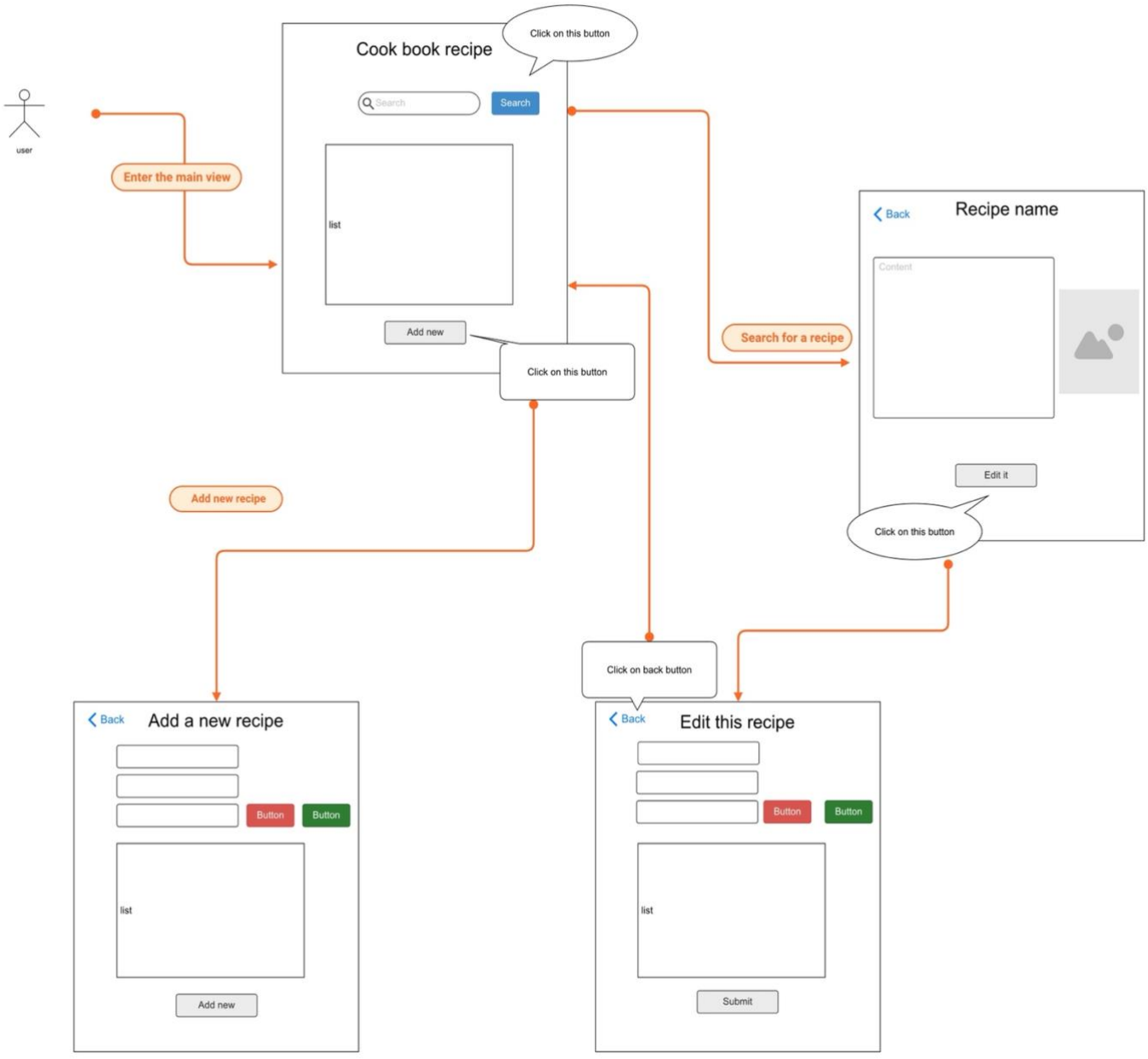
3. GUI Design

3.1. Structure/wireframes

3.1.1 Mockup



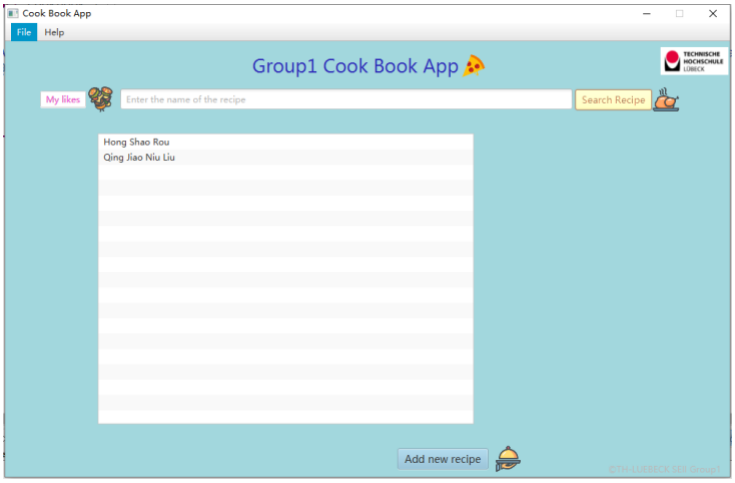
3.1.2 Wireframe



3.2. Application of usability heuristics with screenshots

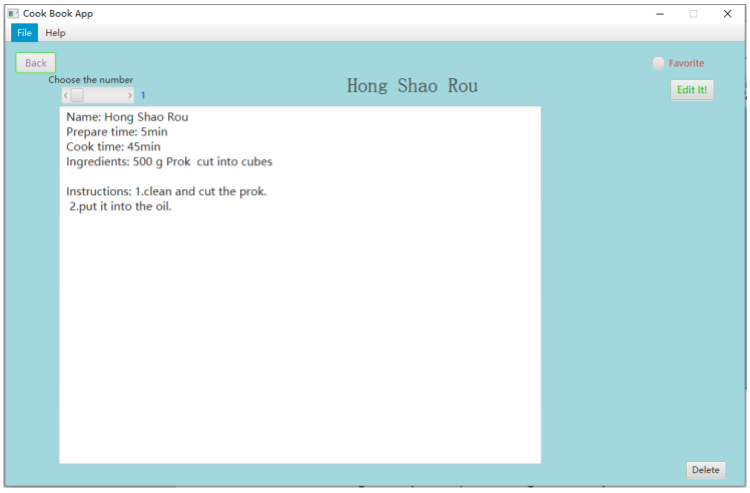
First of all, our design principle for GUI is: simple and clear!

3.2.1 Home



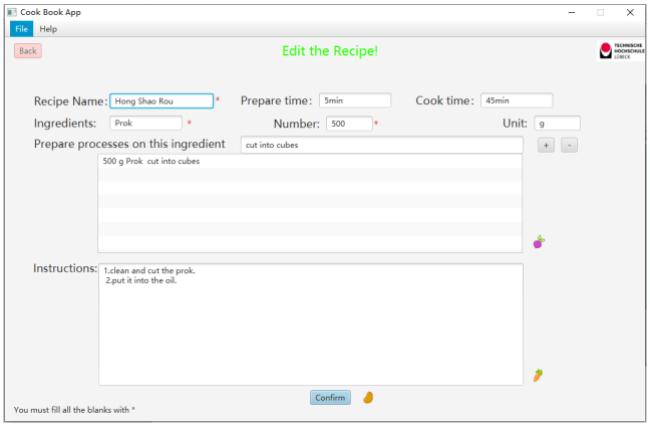
In the HOME, users can see the search box on the head and a search button beside it. Users can fill in the search text field and click the search button to get the recipe he/she wants. In the list view, there are some recipes' name, when clicking one of them, the picture will show in the image view next to the list. When double clicking the recipe name, it will change to the recipe scene.

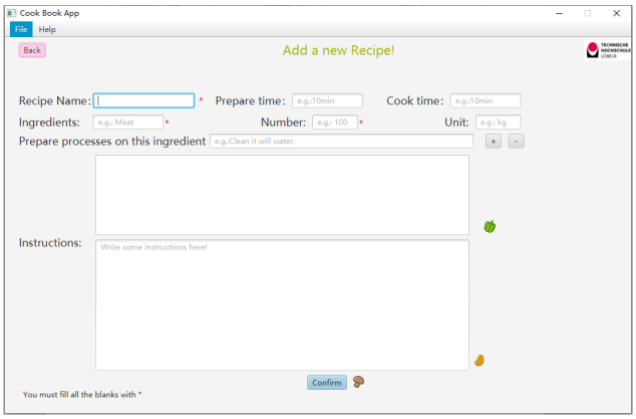
3.2.2 Recipe view



In the recipe scene, users can see the name of this recipe at the top and the content of the recipe all in the text area. A scroll bar can be dragged and the number will change according to it. Then users can know how much ingredients should he/she buy if he/she want to cook for certain number of guests. Also, the picture of the recipe can be shown in the right side next to the text.

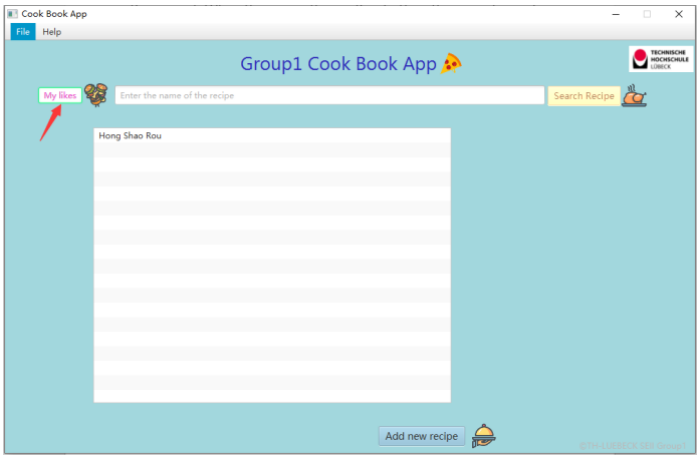
3.2.3 Edit recipe view and add new recipe view



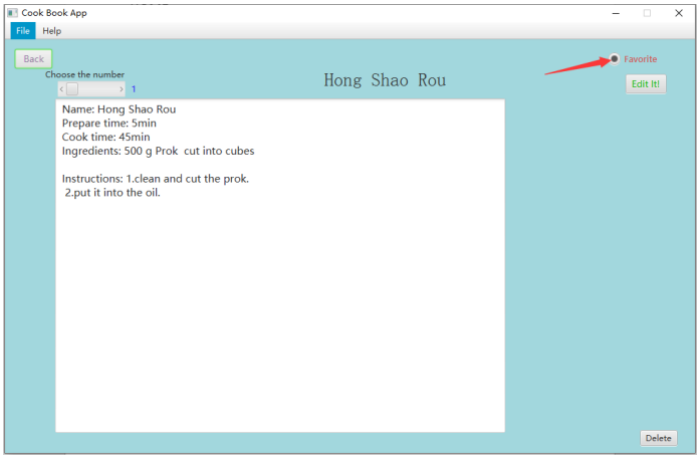


The add new recipe and edit recipe is about the same. Users can change an existing recipe or add a new recipe all by himself in these two views. They must write a name of the recipe and must add at least one ingredient by clicking the add button on the left side. They can also choose whether add the instructions, prepare time, cook time or not. When they press the confirm button, they can choose to look at the new recipe added by themselves just now or back to the HOME.

3.2.3 Favourite

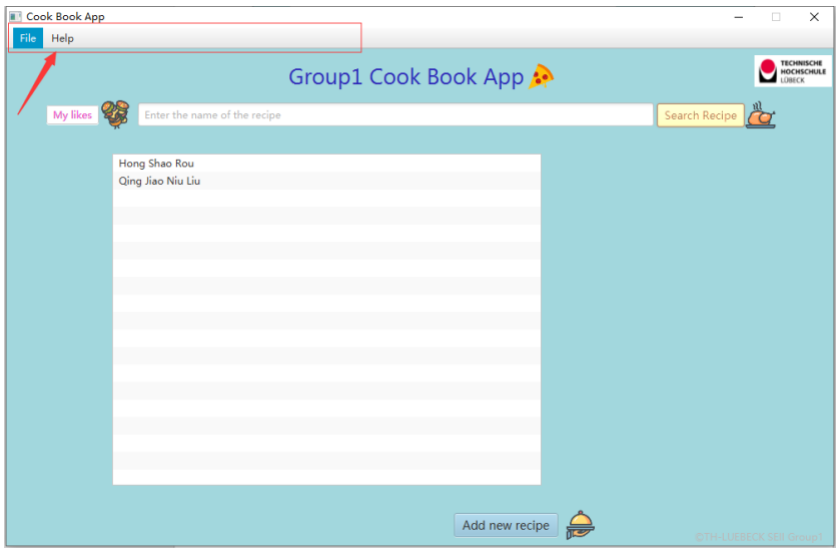


There is a button in the HOME, clicking on it and users can visit their collections, their favourite recipes will show in the list.



Click on the “Favourite” button can add this recipe to the collection. Click that again can cancel it.

3.2.4 Menu Bar

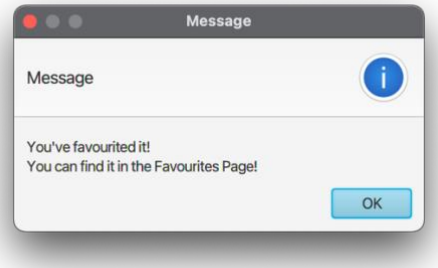


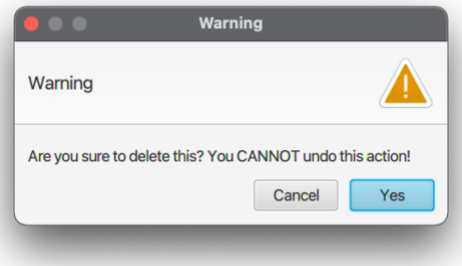
Users can find the menu bar in every scene. They can click on the menu item whenever they want to. There are several functions in the menu bar.

3.2.5 Hint Text

From the pictures above, we add hint text to help our users as they may have no idea what should they do when using our software.

3.2.6 Notifications and Error Prevention





Last but not least, we implemented message/confirm notifications to notify users what has been/will be done and the potential risks to make sure that the user are well-informed.

Warning alerts will be displayed when users want to make decisions such as confirm deletion on recipes; Error alters will be displayed when users made a mistake on specific area such as entering characters instead of numbers to quantity text field; Notifying alters will be displayed after the users' confirmation and no invalid input is detected, such as save recipes successfully.

4. Test

4.1. EC/BVA

4.1.1. Description

We performed the black box test (Robust Weak)EC/BVA wherever applicable as it is an efficient method for deriving test cases. This helps us test whether the application works in the way we planned and in a reasonable manner.

The details are in the next section below.

4.1.2. Results

Equivalence Test for searching recipes

EC classes are defined in below:

Test fields	Valid Equivalence Classes	Invalid Equivalence classes
Search bars for recipes in the main panel and the collection panel	EC1 = {text | text that includes letters, numbers and other characters or empty}	EC2 = {text | empty text }

The EC test results:

	Input	Output
EC1	‘Hong’	Displaying all the recipes that include the phrase ‘Hong’
EC2	empty	No filtered results and show all the recipe in the system.

Equivalence Test for adding and editing recipes

The adding and editing recipe panels are identical, so the EC tests are done together and results are presented together.

EC classes are defined in below:

Test fields	Valid Equivalence Classes	Invalid Equivalence classes
Recipe Name	RN1 = {text | text that includes letters, numbers and other characters; length <= 50}	RN2 = {text | empty text} RN3 = {text | length > 50}
Prepare time	PT1 = {text | text that includes letters, numbers and other characters; length <= 10}	PT2 = {text | length > 10}
Cook time	CT1 = {text | text that includes letters, numbers and other characters; length <= 10}	CT2 = {text | length > 10}
Ingredient Name	IN1 = {text | text that includes letters, numbers and other characters; length <= 30}	IN2 = {text | empty text} IN3 = {text | length > 30}
Quantity (ingredient quantity)	Q1 = {integer}	Q2 = {text | empty text} Q3 = {non-integer value or other characters}
Unit (unit of ingredient)	IU1 = {text | text that includes letters, numbers and other characters; length <= 10}	IU2 = {text | length > 10}
Pretreatment of the ingredient	PP1 = {text | text that includes letters, numbers and other characters; length <= 50}	PP2 = {text | length > 50}
Instructions (on the recipe)	I1 = {text | text that includes letters, numbers and other characters; length <= 1000}	I2 = {text | length > 1000}

The EC test results:

	Input	Output
RN1	‘Gong Bao Ji Ding’	Succeeded in modifying, no alerts prompted, user can continue to change the next item.
RN2	empty	Prompt alert saying “Name can’t be nothing!” and the recipe cannot be saved.

RN3	‘Gong Bao Ji Ding Hong Shao Rou Suan La Fen Shui Jiaooo’	Error message will be prompted in the console.
PT1	’15 min’	Succeeded in modifying, no alerts prompted, user can continue to change the next item.
PT2	’15 hour min’	Error message will be prompted in the console.
CT1	’1 hour’	Succeeded in modifying, no alerts prompted, user can continue to change the next item.
CT2	‘for a long long time’	Error message will be prompted in the console.
IN1	‘sugar’	Succeeded in modifying, no alerts prompted, user can continue to change the next item.
IN2	empty	Prompt alert saying “Ingredient Name can’t be nothing!” and the recipe cannot be saved.
IN3	‘sugar sugar sugar sugar sugar sugar’	Error message will be prompted in the console.
Q1	‘10’	Succeeded in modifying, no alerts prompted, user can continue to change the next item.
Q2	Empty	Prompt alert saying “Must be a number!” and the recipe be saved with the ingredient quantity of 1.
Q3	‘idontknow#’ or ‘3.0’	Prompt alert saying “Must be a number!” and the recipe be saved with the ingredient quantity of 1.
IU1	‘kilogram’	Succeeded in modifying, no alerts prompted, user can continue to change the next item.
IU2	‘kilogramkilogram’	Error message will be prompted in the console.
PP1	‘boiled, steamed and fried’	Succeeded in modifying, no alerts prompted, user can continue to change the next item.
PP2	‘stemmed, halved crosswise, and seeded, peeled, thinly sliced’	Error message will be prompted in the console.
I1	‘1.Mix together cornstarch and 1 tbsp. of the soy sauce in a medium bowl.’	Succeeded in modifying, no alerts prompted, user can continue to change the next item.
I2	‘1.Mix together cornstarch and 1 tbsp. of the soy sauce in a medium bowl. 2.Add chicken, toss well, and set aside to marinate for 30 minutes. 3.Meanwhile, mix together the remaining 3 tbsp. soy sauce, rice wine, sugar, stock, vinegar, sesame oil, and dark soy sauce. 4.Set aside.	Error message will be prompted in the console, unable to save the recipe.

	5.Heat peanut oil in a wok or large nonstick skillet over high heat until just beginning to smoke. 1.Mix together cornstarch and 1 tbsp. of the soy sauce in a medium bowl. 2.Add chicken, toss well, and set aside to marinate for 30 minutes. 3.Meanwhile, mix together the remaining 3 tbsp. soy sauce, rice wine, sugar, stock, vinegar, sesame oil, and dark soy sauce. 4.Set aside. 5.Heat peanut oil in a wok or large nonstick skillet over high heat until just beginning to smoke.'	

Boundary value analysis:

As there are no set of values or entries in all the equivalence classes defined above, there is no need to conduct boundary value analysis.

	Input	Output

4.2. Usability test

4.2.1. Description

Usability is a measure of how easy it is to use a product to perform prescribed tasks and it is a user-centered attribute of a good software. Therefore, the users must be included in the usability test process.

So far, only **five real user and our three developers** are involved in the usability test, but the designers cannot represent the users, because they have taken part in the process of developing and they know exactly how things are going, namely, they have more intimate knowledge. Therefore, the flaws can only be detected by testing the real users.

User profile analysis and Sample Choice

The target users should be people who understand English aging from 16 to 60. Besides, they should have basic computer skills to operate the software. The users' devices should be able to run java programs. Hence, we managed to successfully invite 5 students from other class without professional software developing knowledge to help us to complete the test.

Methods and Goals

There are five quality components of usability: learnability, efficiency, memorability, errors, and satisfaction. To test the five components, we decide to set tasks to our users and see how they accomplish these tasks without help just making records. Besides, we will design a questionnaire to the users. The questionnaire will focus on questions how the software looks like from a user's perspective.

- Overall purpose: How usable is our application?
- Objectives: Is the features easy to use for the user?
- Type of test: Performance
- Task:
  - Can user Favourite a recipe and view it in the Favourites Collection page?
  - Can user add/edit one recipe?
- Performance Objective: If the task was completed successfully.

Records

1. The time the users need to accomplish every single task.

2. Record the number of errors the users make during every task.

Preparations Beforehand:

- Task list
- Consent forms
- Questionnaire

Team Roles:

- Briefer: Bo Jiao
- Note taker: Hao Yuan

Questionnaire

1. Do you think it is easy to accomplish the task?

A. Very easy B. Easy C. Normal D. Difficult

2. How do you like the layout of the interface?

A. Very clean B. Clean C. Normal D. Messy

3. How do you like the interface from an aesthetic perspective?

A. Very beautiful B. Beautiful C. Normal D. Ugly

4. Do you think the software is helpful in real life when you need a cooking navigator?

A. Very helpful B. Helpful C. Just so- so D. Not helpful

5. How many stars will you give to the software?

A. ★★★★★ B. ★★★★ C. ★★★ D. ★★ E. ★ F. None

6. In which way do you like the software?

A. Easy to operate B. Beautiful interface C. Multiple functions D. Satisfy my demand

7. Do you have any suggestions for this software?

4.2.2. Evaluation/results

Before real user test, we three developers already did some usability test by ourselves and made modifications to the software.

The real user test result is pretty gratifying. All five test takers can perform given task in less than 5 minutes without any trouble. They all feel the task is easy to perform and think the application is helpful in real life. Two of them think our UI design is very clean and very beautiful. And our application is rated 4.5 stars out of 5 generally.

There is one suggestion from one of the test takers:

“Hope u guys will make a mobile version of it. Definitely gonna download it!”

5. Evaluation

5.1. Group work

We three as a group, hold regular internal meetings every week on Wednesday afternoon after external meetings with the supervisor – Mr. Zhan. The meetings cover the problems regarding the project, the feedbacks from the supervisor as well as the further plans for project development. Also, we three team members meet as we need during the development.

Moreover, after the meetings, Bo Jiao - as the team leader - will always make arrangement about the upcoming tasks for each teammate based on the previous work and evaluations. And also keep an eye on the overall progress of the whole group.

We three always keep an eye on each other's progress, help each other when needed and promote each other to get to the next step.

5.2. Task responsibilities

For task distribution:

- Bo Jiao: mainly responsible for
  - Entity classes
  - DAO and controller implementation (partly involved)
  - made modifications in GUI design
  - in charge of the whole developing process
- Hanzhi Zhuang: mainly responsible for
  - DAO implementation
  - database design
- Hao Yuan: mainly responsible for
  - GUI (view) design
  - controller implementation

For other parts, we three made equal contribution.

Details can be found below:

	Bo Jiao	Hanzhi Zhuang	Hao Yuan
Regarding to Codes			
entrance.RecipeBoxApp.java	●		
Controller.AddViewController.java	○		●
Controller.EditViewController.java	●		●
Controller.MainViewController.java	○		●
Controller.RecipeViewController.java	●		●
Controller.rootLayoutController.java	●		●
dao.BaseConnector.java		●	
dao.RecipeDAO.java	●	●	
dao.IngredientDAO.java	○	●	
model.Cookbook.java	●		○
model.Ingredient.java	●	○	○
model.Recipe.java	●	○	○
view.AppMain.java			●
view.AddView.fxml	●	○	●
view.EditView.fxml	●	○	●
view.RecipeView.fxml	●	○	●
view.MainView.fxml	●	○	●
view.rootLayout.fxml	○		●
Other stuffs			
data.sql	●	●	
uml diagrams	●	●	○
EC/BVA test	○	●	○
Usability test	●	●	●

Table 5-1 Group Contribution

In this table, "●" represents the member was responsible for the file and did the main job and "○" represents the member helped or did an assistant contribution for the file. If there exist more than one "●" or "○" in a row, that means they made equal contribution in that participation level.

6. Acknowledgement

[1] Some icons used to decorate the application are from https://sc.chinaz.com/tubiao/index.html .

[2] External link to www.allrecipes.com is added.

[3] Some recipe pictures are from https://rasamalaysia.com .

[4] External jar from apache.org is used to translate new line '\n' character between database and java application.