COMP9900 Project Report

Meal Recommendation System

Team: Brainstorm

Lab: M18Q

z5228748 Hanyue Jiang z5228748@ad.unsw.edu.au Back-end

z5319735 Shiyu Nie z5319735@ad.unsw.edu.au Front-end

z5295651 Zening Wang z5295651@ad.unsw.edu.au Front-end

z5298115 Han Yan z5298115@ad.unsw.edu.au Front-end

z5290496 Xueqing Ren z5290496@ad.unsw.edu.au Back-end

Submission date:

18 Nov 2022

Contents

1. Overview

1.1 Background

With the rapid development of the Internet, people's access to information is more abundant and convenient than before, and less limited by time and space. For people who love cooking, they can easily obtain a lot of recipe information, but also need to spend more time to screen out the parts they are interested in. In order to help cooking enthusiasts more easily obtain useful recipe information, and complete the improvement and innovation of food by exchanging recipes between each other, we designed this website to collect and display recipes from all over the world.

1.2 Project Description

First of all, we hope that users can use most of the content related to recipes on this website as tourists (i.e. without login). The design of this part is closer to the traditional form of sharing recipes through physical materials such as books, which also provides convenience for some users who only want to quickly obtain recipes.

Considering that the website will have a large number of recipes with the increase of users, it will be very difficult for some users without clear goals to obtain a favorite recipe. For this reason, in addition to the common recipe search function, we also designed the function of filtering according to the meal type, ingredients, cooking time and method of recipes. In addition, we also designed an innovative ranking list function to display the most popular recipes in the entire website. By calculating the popularity of recipes, we hope that this ranking can more truly reflect users' preferences for recipes, so as to provide better suggestions. The recommended recipes on the website can also help people to obtain the recipes they may be interested in and thus bring better user experience.

Registered users can enjoy more functions of communication about recipes than tourists. Having a personal account enables them to share their own recipes, collect favorite recipes or follow the recipes published by specific users. In order to facilitate users' use and the beauty of the page, users have fixed templates when uploading and modifying their personal recipes, so they do not need to worry about difficulties in operation. Compared with other recipe websites, we pay more attention to supporting the communication and strengthening the connection between users. For this purpose, we provide the subscription function between registered users, so that a user can continuously obtain the recipes published by their subscribers and easily find the ones they may be interested in more.

1.3 System Architecture

The project adopts separate front-end system and back-end system, as shown in the figure below. Two systems do not depend on each other and can interact with each other through Http

Request/Response JSON. And it is a 3-layer structure shown below:

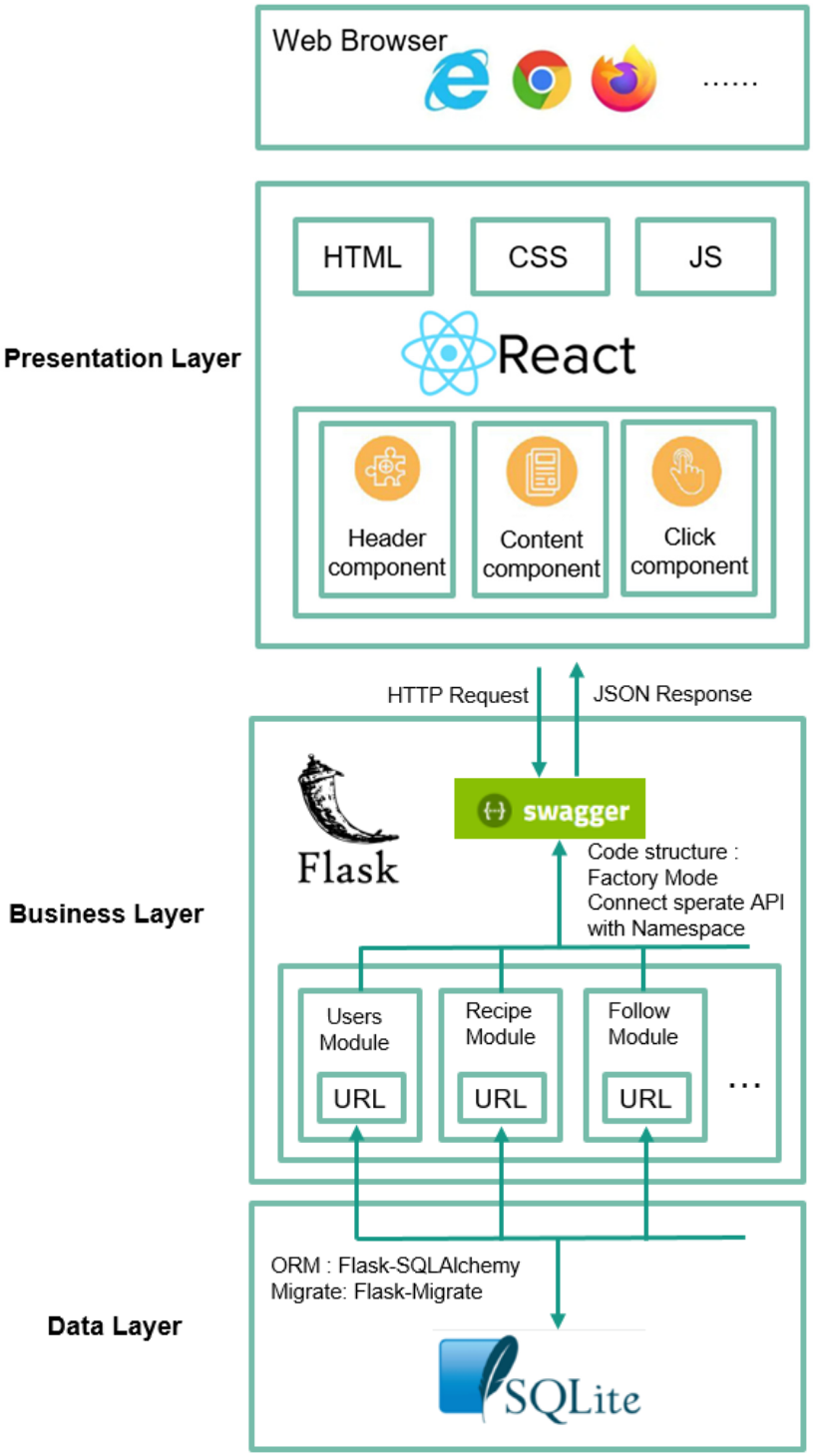

Figure 1.1 System Architecture

1.3.1 Presentation Layer

The presentation layer is what the system user sees or interacts with.

- Front-end framework: React

For our presentation layer, the React framework is used for the front-end design. React[1] improves Web performance by introducing the concept of virtual DOM and performing micro operations in it to implement local updates to the actual DOM. Virtual DOM also provides a standardized API to achieve cross browser compatibility. In addition, modular UI components can be independently developed and tested, which improves the maintainability of the code.

- Main component : Antd

The main component used is Antd which is a React UI component library based on Ant Design design system[2]. It is a high-quality React component built with TypeScript and can provide complete type definition files.

1.3.2 Business Layer

The business layer is the middle layer between presentation layer and data layer which is responsible for all business logic.

- Language: Python
- Framework: Flask

We use Flask framework for the back-end design in our business layer. Flask is a micro web framework written in Python. It is relatively easy for development and the environment is simple to deploy. Besides, Flask is also compatible with a variety of databases and templates.
Flask-SQLAlchemy is used for ORM operations when interacting with database. Without writing any SQL language to do CRUD, we can avoid SQL injection and ensure the security of the whole system.

- Extension: Flask-RESTX

Flask-RESTX is a Flask plug-in that supports RESTful. It is used for writing standardized interfaces and supports swagger documents.

- Test Framework: Unittest

To test our code, we choose Unittest. Unittest is a built-in unit test module of Python[3]. It has a complete test structure which supports the execution of automated tests, organizes test case sets, provides rich assertion methods, and finally generates test reports.

1.3.3 Data Layer

Data layer is where we store persistent and temporary data related to our system.

- Database: SQLite
  Store all text data in Relational Tables.
  Store pictures in flask static folder and store pic URL to database.

The database we use is SQLite which is a lightweight ACID compliant relational database system. As it is easy to do migrate with Flask-migrate module for SQLite, the development efficiency can be improved.

2. Data Model

2.1 Data Preparation

In order to test and better display the functions of the website, we added some data related to user accounts and recipes in the database in advance.

We created 4 user accounts and used them to add a total of 21 recipes. According to the functions, the data types we added and descriptions are as follows:

Data Type Involved	**Function**	**Description**
User	User Registration & Validation & Profile	Four user accounts with data of email, username, password, avatar, gender and introduction. *Only half of the accounts contain real name data.
	Subscribe Module	Each user has a varying number of subscribers and fans.
Recipe	Recipes Maintaining	Twenty-one recipes with data of name, meal type, main ingredient, ingredient detail, method, time, photo, description and view number(times). *Not every recipe has been viewed, and most recipes have been viewed different times.
	Searching Module	
	Recipes Ranking	
	Recommendation System	
User & Recipe	Like & Comment on Recipes	Each user has one or several favorite recipes but not every recipe is liked or commented by some users. Only three users make comments on recipes.
	Favorites Module	

You can check the detailed built-in data in **demo.db** in the code repository.

2.2 Database Architecture

We use SQLite to store data and the structure is shown in the following figure.

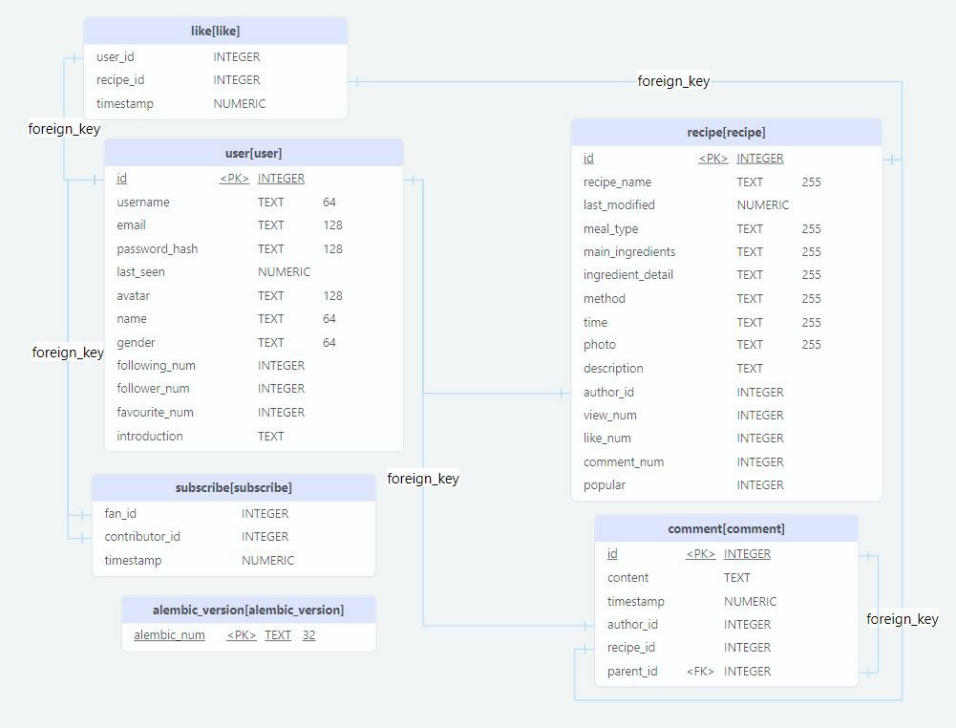

Figure 2.1 Database Schema

2.2.1 Table User

The user table stores the data of user account, with 12 fields in total.

- Field ***id*** is the primary key of this table. Each user will be given a unique user id when registering. This field is also used in the rest of tables as foreign key.
- Field ***username*** and ***email*** store the information entered when a user registers.
- Field ***password_hash*** is the data encrypted by the password entered by users.
- Field **last_seen** is the time of a user's last login.
- Field ***avatar*** stores the url of the image a user uploaded to use as his or her avatar.
- Field ***name***, ***gender*** and ***description*** store the information entered optionally when a user edits his or her personal information.
- Field ***following_num*** shows the number of subscribers of a user, and will be upgraded when a user subscribe or unsubscribe other users.
- Field ***follower_num*** shows the number of fans of a user, and will be upgraded when other users subscribe or unsubscribe this user.
- Field ***favourite_num*** shows the number of liked recipes of a user, and will be upgraded when a users liked or disliked recipes.

2.2.2 Table Recipe

The recipe table stores the data of recipes, with 15 fields in total.

- Field ***id*** is the primary key of this table. Each recipe will be given a unique recipe id when created. This field is also used in table *comment* and table *like* as foreign key.
- Field ***recipe_name***, ***meal_type***, ***main_ingredients***, ***ingredient_detail***, ***method, time*** and ***description*** stores the information entered when a recipe is created or modified separately.
- Field ***last_modified*** is the time of a recipe's last edit.
- Field ***photo*** stores the url of the image a user uploaded to use as a recipe's photo.
- Field ***author_id*** is the foreign key from user table's field ***id***. It indicates the id of the creator of a recipe. Each user can create many recipes.
- Field ***view_num*** shows the number of view times of a recipe, and will be upgraded when the recipe is viewed. Viewing of tourists and repeated viewing of a recipe by the same user will also increase this value.
- Field ***like_num*** shows the number of users liked one recipe. Users cannot like the recipes published by themselves.
- Field ***comment_num*** shows the number of comment published on one recipe.
- Field ***popular*** is the popularity calculated according to the other data of a recipe. The specific calculation method is described in detail in the following text.

2.2.3 Table Like

The like table shows the relationship between users and their favourite recipes, with 3 fields in total.

- Field ***user_id*** is the foreign key from user table's field ***id***. Field ***recipe_id*** is the foreign key from recipe table's field ***id***. If a user liked a recipe, the user's id and the corresponding favorite recipe's id will be stored in the table together.
- Field ***timestamp*** is the time when a user liked a recipe.

2.2.4 Table Comment

The comment table shows the relationship among comments left on recipes, comments' authors and recipes, with 6 fields in total.

- Field ***id*** is the primary key of this table. Each comment will be given a unique comment id when published.
- Field ***content*** stores the information entered when a user comments.
- Field ***timestamp*** is the time when a comment was made by a user.
- Field ***author_id*** is the foreign key from user table's field ***id***. It indicates the id of the author of a comment.
- Field ***recipe_id*** is the foreign key from recipe table's field ***id***. It indicates the id of the commented recipe. There can be many comments under each recipe.
- Field ***parent_id*** is the foreign key from comment table's field ***id***. It was originally used for second level comments, but since this function was not implemented, this field was left blank.

2.2.5 Table Subscribe

The subscribe table shows the relationship between users and users, with 3 fields in total.

- Both field ***fan_id*** and field ***contributor_id*** are the foreign key from user table's field ***id***. If a user subscribes another user, the user's id will be stored as fan_id and the corresponding subscriber's id will be stored as contributor_id in the table together.
- Field ***timestamp*** is the time when one user subscribe another user/followed by another user.

3. Functionalities

Figure 3.1 Project objectives from Proposal

In our proposal, we list eight different objectives for this project. So in this section, we will demonstrate our functionalities depend on our objectives.

3.1 Home Page

The home page consists of an information bar at the top and a recipe recommendation block at the bottom. The information bar contains the logo and name of the website, and functions' buttons including filtering recipes, searching recipes and viewing ranking list. The recipes on the home page are arranged in the reverse order of the last modified time, showing the recipes' detailed information (including name, picture, creator, and number of comments, likes and views).

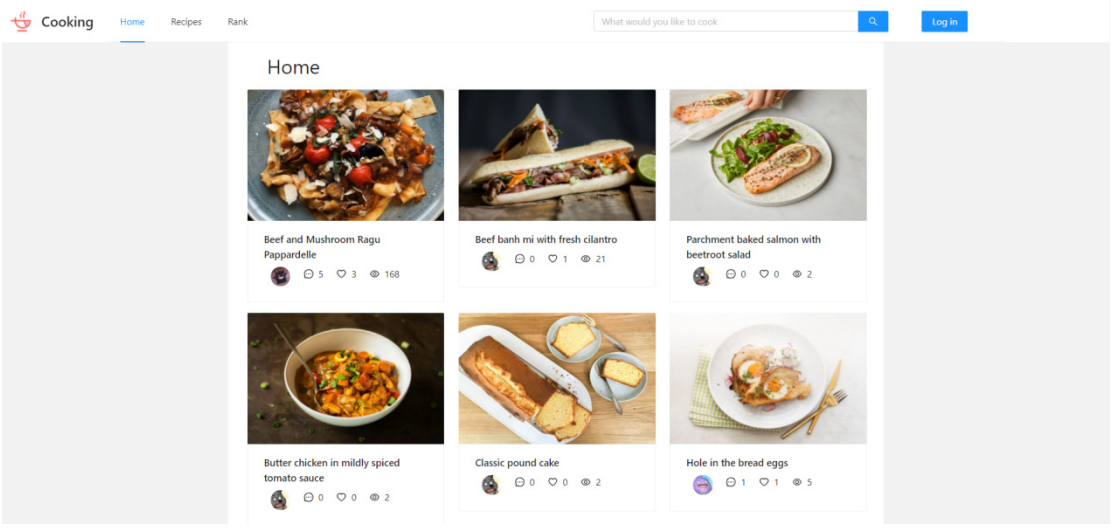

Figure 3.1.1 Home Page (Pull up to the top)

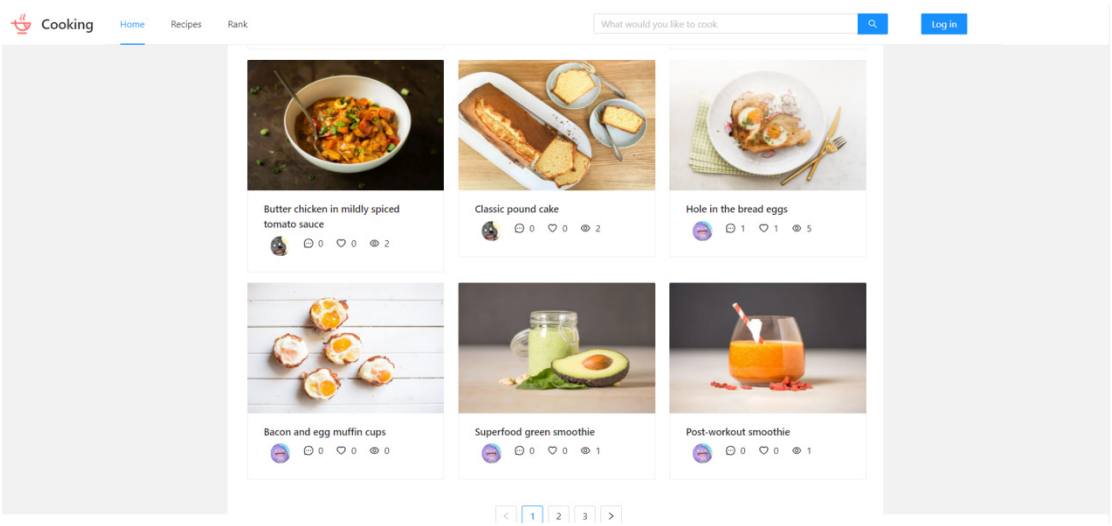

Figure 3.1.2 Home Page (Pull down to the bottom)

Users can browse all the contents of the homepage as tourists (i.e. without login), use all the functions in the information bar and read the specific contents of one recipe.

3.2 User Registration & Validation & Profile

3.2.1 Login

The login page can be accessed through the “Log in” button in the top right corner of home page. If a user already has his or her own account, he or she can log in by using the email and password.

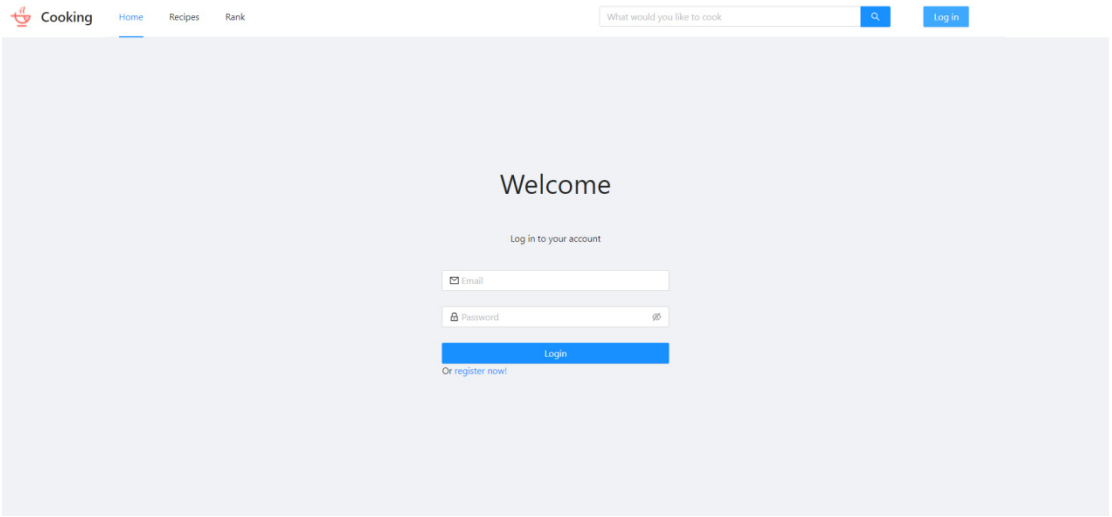

Figure 3.2.1 Login Page

All the inputs on this page should not be blank, otherwise prompt text will pop up and the parts that has not been entered will be identified by red boxes.

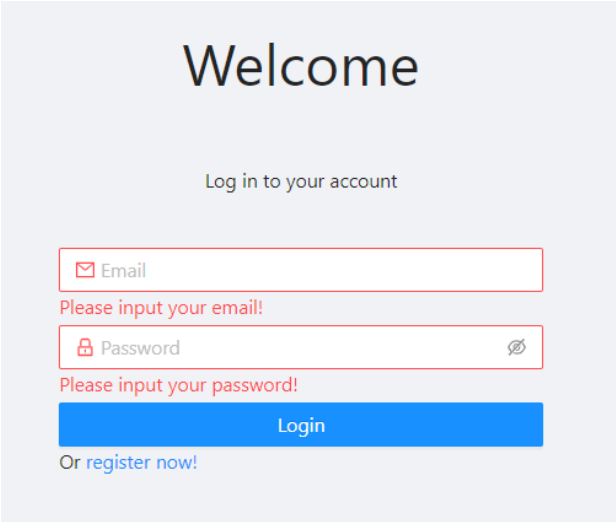

Figure 3.2.2 Blank Input Prompt of Login Page

The user needs to enter the correct registration email and password to log in. If the email and password used by a user do not match, an error message prompt box will pop up at the top of the page and the input contents will be cleared.

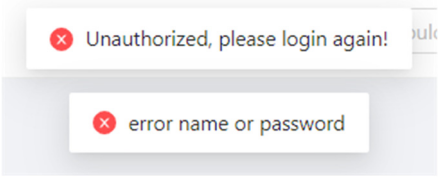

Figure 3.2.3 Error Message for Wrong Email or Password

3.2.2 Register

If a user does not have an account, he or she can find the registration link under the login button of the login page(i.e. the blue words “register now!”).

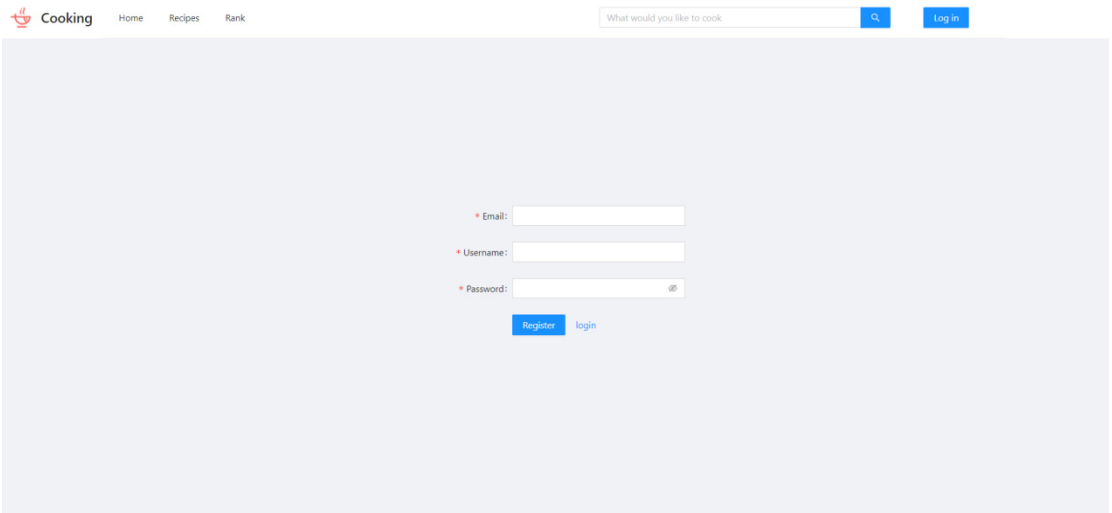

Figure 3.2.4 Register Page

At the time of registration, the user needs to enter his or her email, username and password. If one or more of the three items are not filled in, there will pop up prompt text correspondingly and the parts that has not been entered will be identified by red boxes. The validation of email, name and password will be checked.

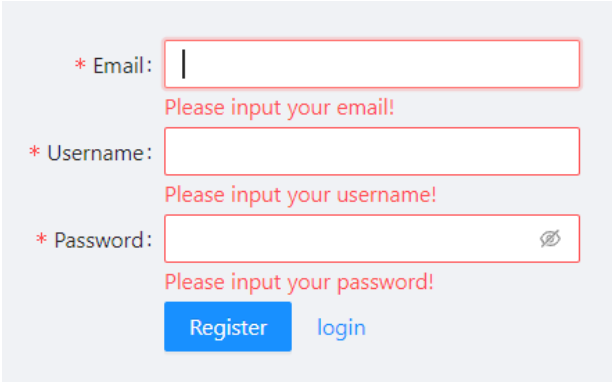

Figure 3.2.5 Blank Input Prompt of Register Page

The email must be in the form of "username@domain", otherwise it will show prompt text at the bottom of the input box.

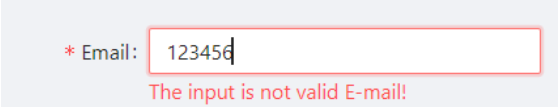

Figure 3.2.6 Error Prompt for Invalid Email

The user name can only contain uppercase and lowercase letters and underscores.

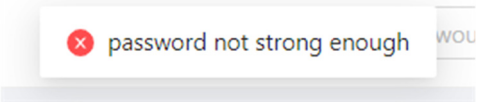

Figure 3.2.7 Error Message for Invalid Password

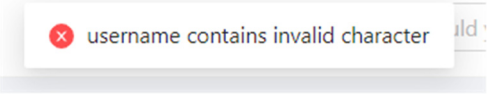

Figure 3.2.8 Error Message for Invalid Username

The password must be at least eight characters and contain uppercase and lowercase letters and numbers simultaneously.
Invalid username or password will both pop up an error message at the top of the page after the user clicks the register button below.
If a user who already has an account enters this page by mistake, he or she can return to the login page by clicking the blue word beside the register button at the bottom of the page or the login button at the upper right corner of the page.

3.2.3 Personal Information Page

3.2.3.1 Personal Information Main Page

After a user logs in, a small icon of the user's avatar will be added in the upper right corner of the home page. When the mouse slides over the avatar icon, a floating box will appear below. The user's personal information page can be entered by clicking the avatar in the floating box.

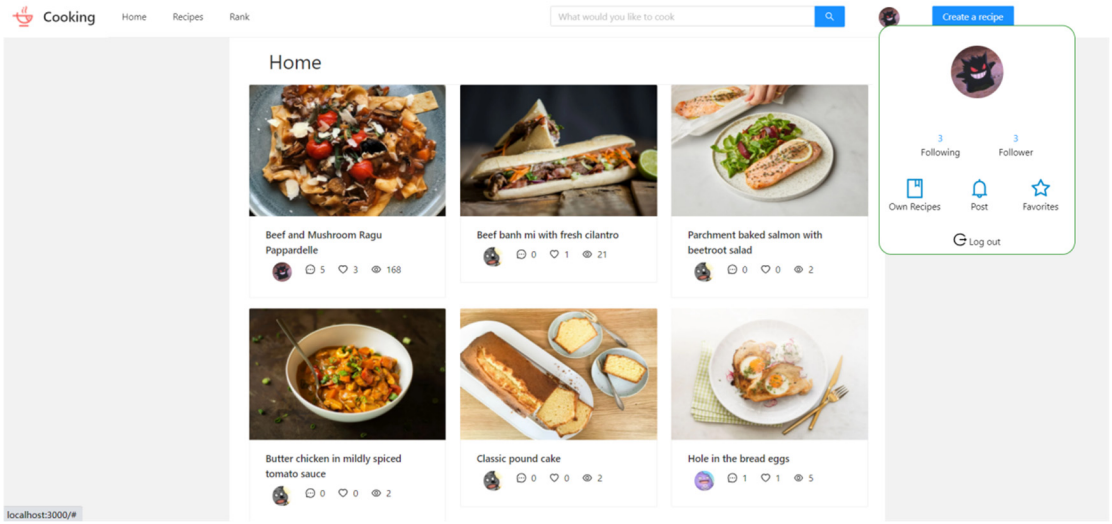

Figure 3.2.9 Home Page after Login with Floating Box (Top Right)

The user's personal information page will display a user's detailed information, including registration information (username and email) and optional information (real name, gender, introduction and avatar).

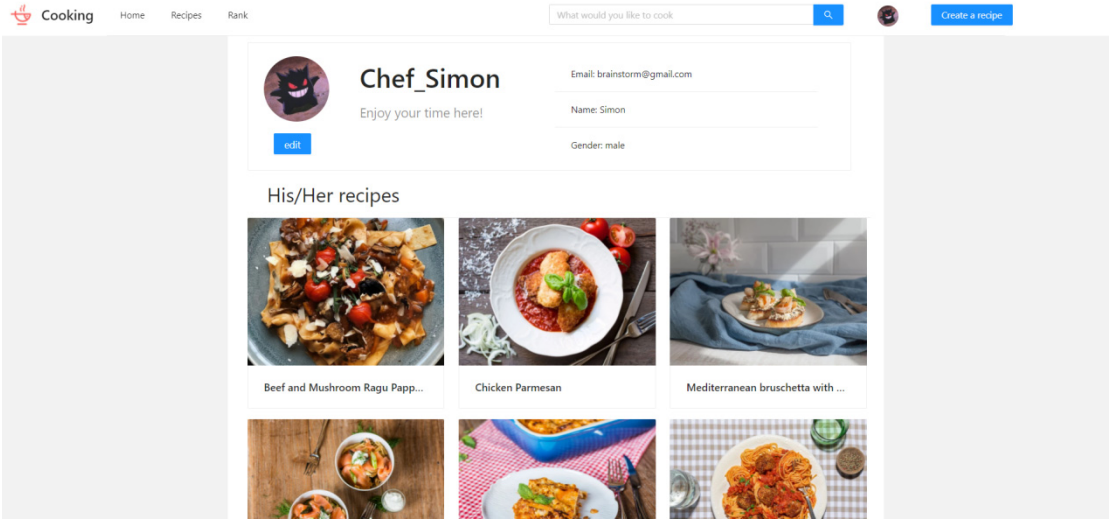

Figure 3.2.10 Personal Information Page (Pull up to the top)

In addition, the personal recipes uploaded by one user will also be displayed on this page. The pages of recipes can be turned through the buttons at the bottom.

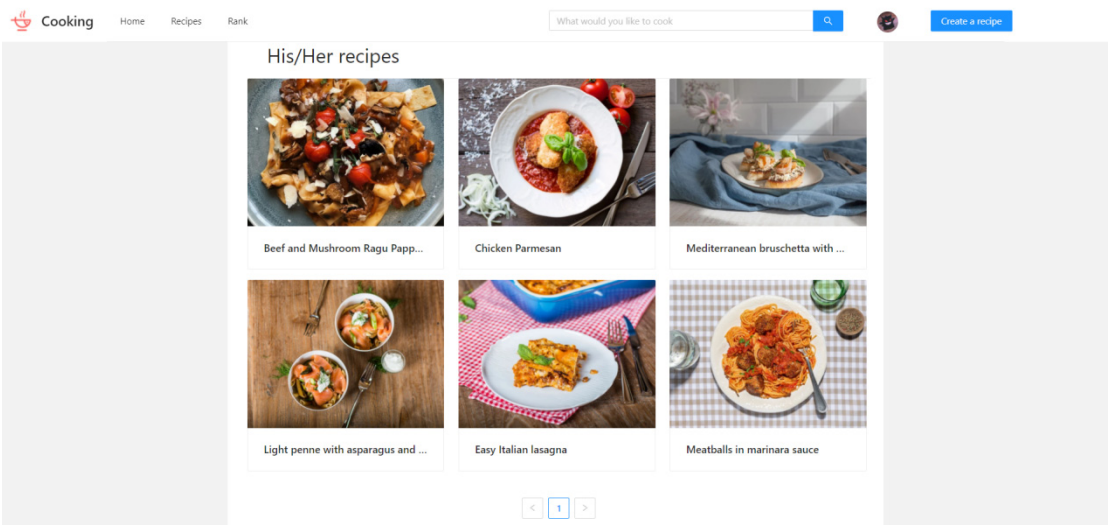

Figure 3.2.11 Personal Information Page (Pull down to the bottom)

3.2.3.2 Personal Information Edit Page

The user can modify his or her personal information through the edit button below the avatar. Except from the modification of username and password, one can choose to add gender, real name, introduction and avatar optionally. For gender, a drop-down window is used to restrict users from selecting male or female. The length of introduction is restricted to 30 characters and the type of avatar should be selected from jpg, jpeg and png.

After successful modification, it will jump to the home page of the website. Besides, if the password is changed, the user will be logged out immediately and needs to log in again with the new password.

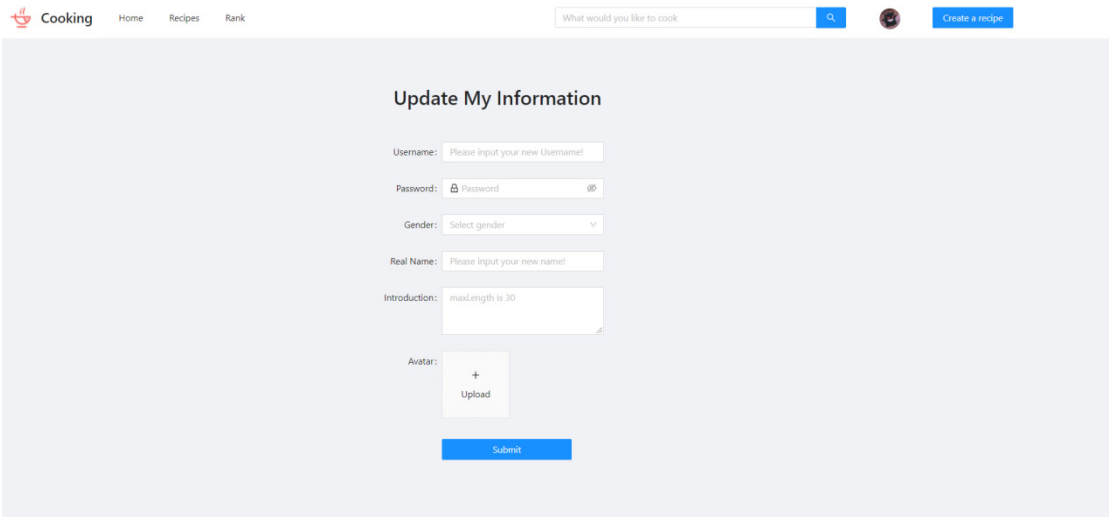

Figure 3.2.12 Personal Information Edit Page

3.2.4 User Floating Box

The floating box provides users with more intuitive data display and links to common pages. The top two parts below the avatar display the number of one user's subscribers and fans, can jump to the corresponding information page by clicking the number. The middle three icons are

linked to one's own recipe page, subscribers' recipes posting page and personal favorites page separately. At the bottom of the box, the user can exit his or her account by clicking the log out button.

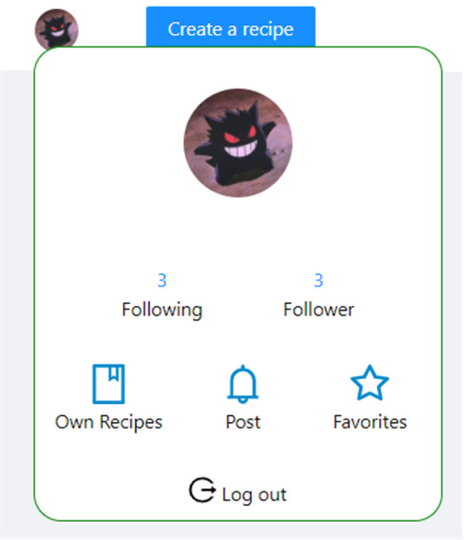

Figure 3.2.13 User Floating Box

3.3 Recipes Maintaining

3.3.1 Creation of Recipes

After the user logs in, click the Create Recipe button in the upper right corner (as shown in Figure 3.3.1) to create their own recipes. The upper right corner of the user who is not logged in is the login button (as shown in Figure 3.3.2), and the function of creating recipes cannot be used.

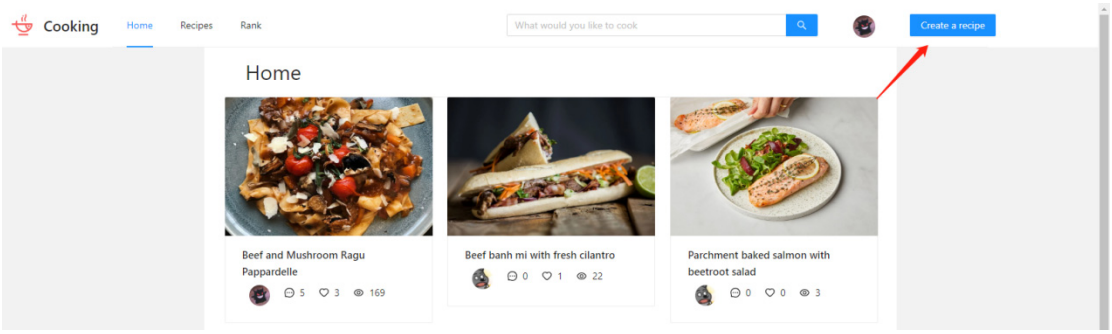

Figure 3.3.1 Recipe Creation Button

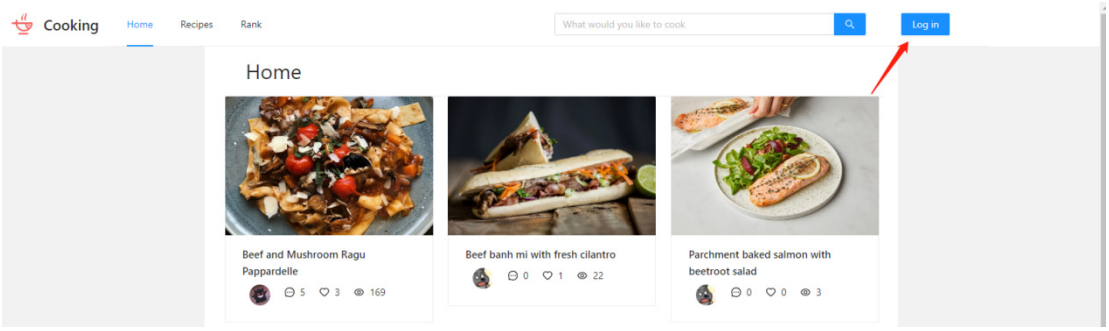

Figure 3.3.2 Login Button

Enter the recipe creation page (as shown in Figure 3.3.3), on this page you can enter the name of the recipe, meal type, main ingredients, ingredient description, time, Method, description of the recipe and pictures of the recipe. Among them, meal type, main ingredients, time and Method all have drop-down options to choose after clicking (as shown in Figure 3.3.4). The name of the recipe, ingredient description and description of the recipe need to be entered manually.

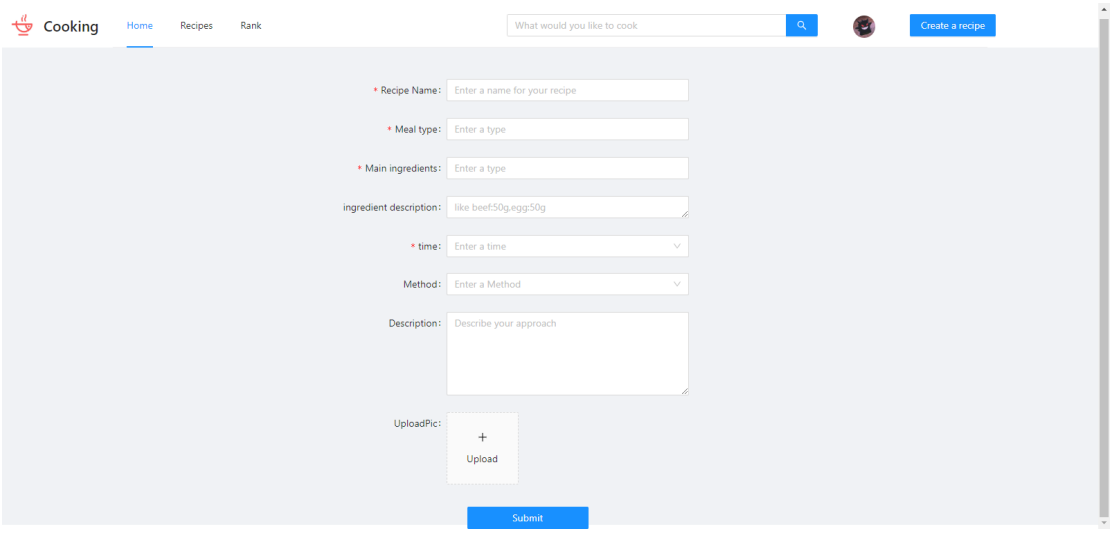

Figure 3.3.3 Recipe Creation Page

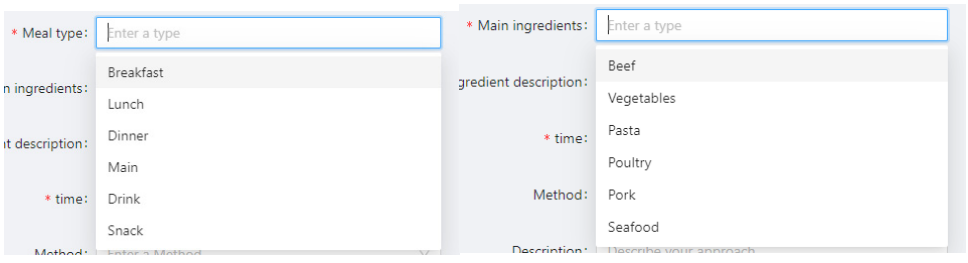

Figure 3.3.4 Drop-down Options for Meal Type, Main Ingredients, Time, and Method

After entering all the content, click the submit button at the bottom and the recipe will be successfully created (as shown in Figure 3.3.5).

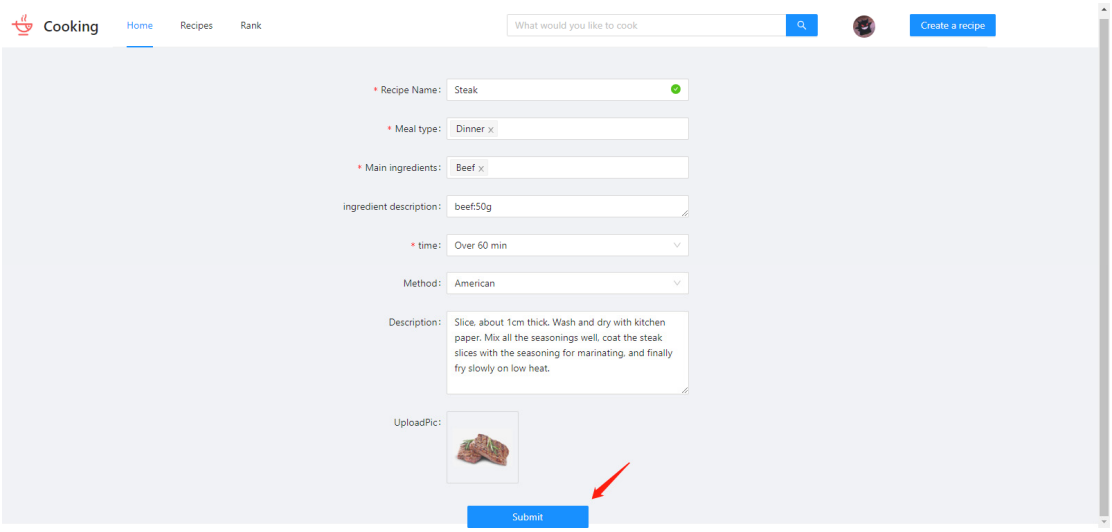

Figure 3.3.5 Recipe Creation Page (Entering All the Contents)

On the home page, you can see the newly created recipe. Clicking on the recipe takes us to the recipe homepage. On the top is the picture of the recipe and the number of views, likes and comments of the recipe. (as shown in Figure 3.3.6)

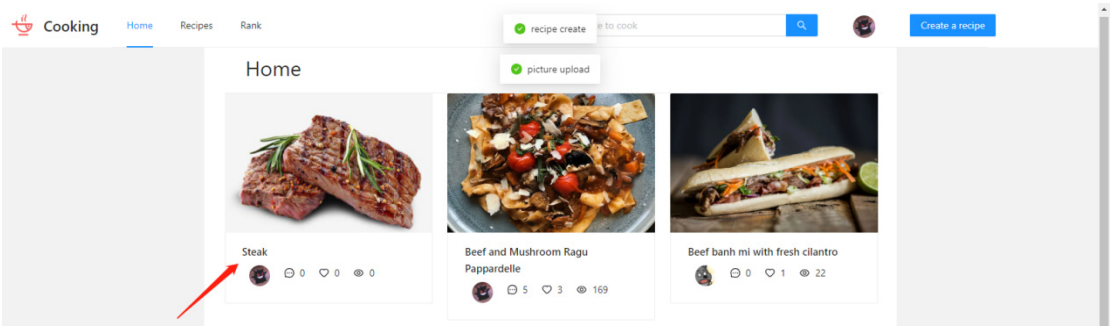

Figure 3.3.6 Home Page (Recipe Created Successfully)

In the middle is Author's information and recipe information (as shown in Figure 3.3.7 and 3.3.8).

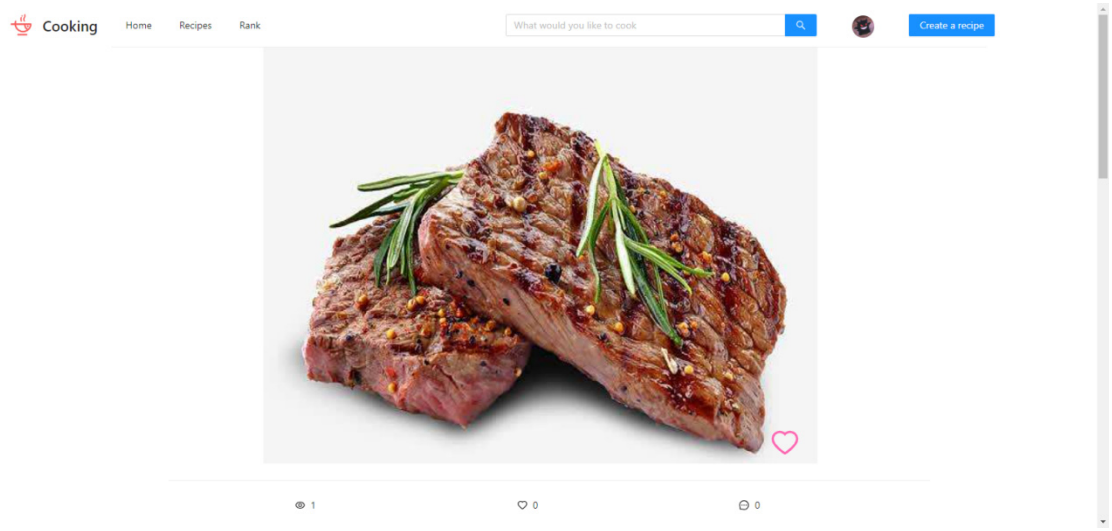

Figure 3.3.7 Recipe Homepage (Picture of the Recipe)

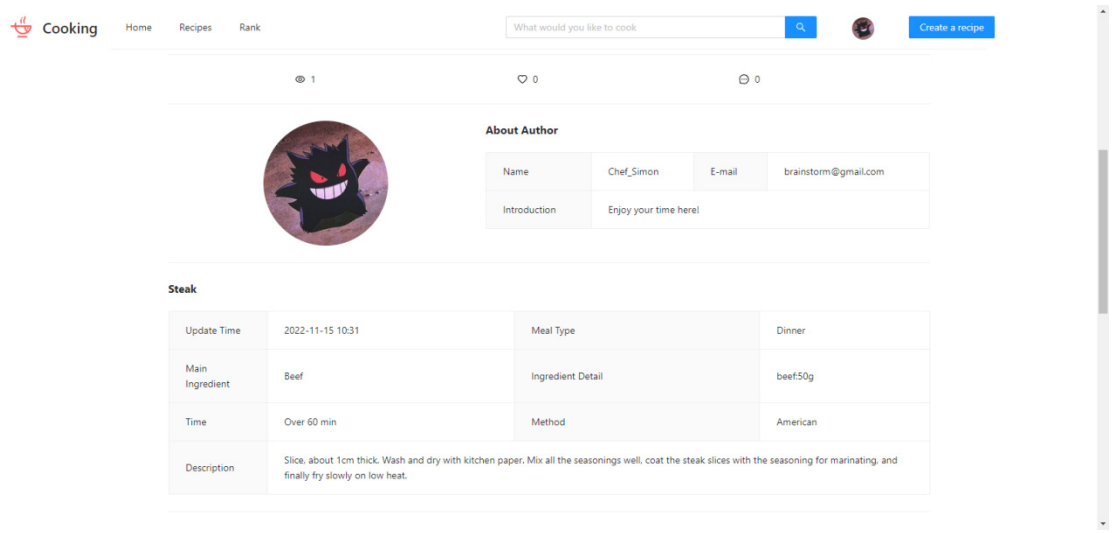

Figure 3.3.8 Recipe Homepage (Author's Information and Recipe Information)

At the bottom are related recipe recommendations and comments about recipes (as shown in Figure 3.3.9).

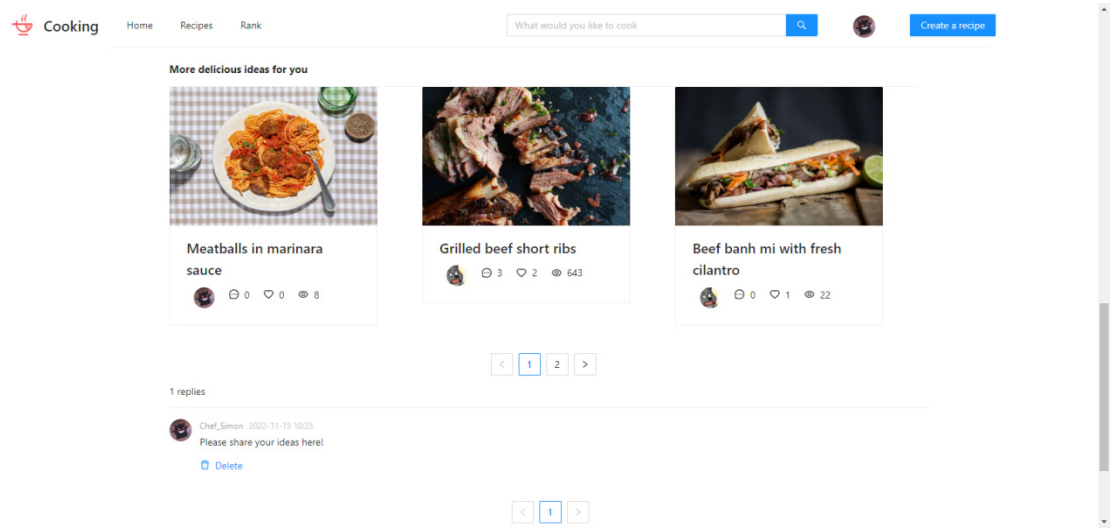

Figure 3.3.9 Recipe Homepage (Related Recipe Recommendations and Comments)

Click the Author profile picture to enter the author's homepage, where the author's detailed information and all recipes created by the author are displayed (as shown in Figure 3.3.10).

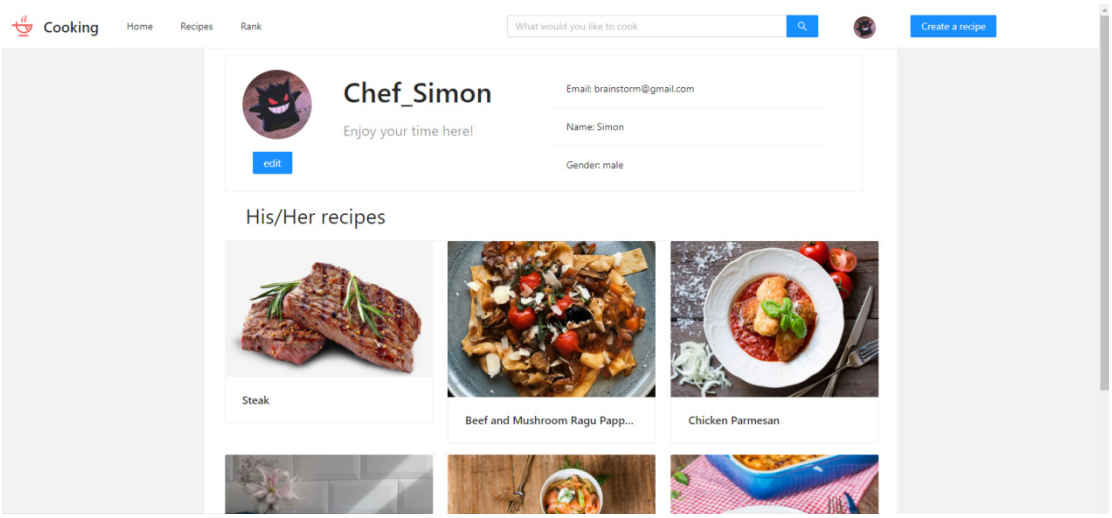

Figure 3.3.10 Author's Homepage

3.3.2 Editing of Recipes

Click the profile picture, click own recipes (as shown in Figure 3.3.11) to enter the own recipes page. On this page, you can see all the recipes created by the user, and you can also modify or delete these recipes (as shown in Figure 3.3.12).

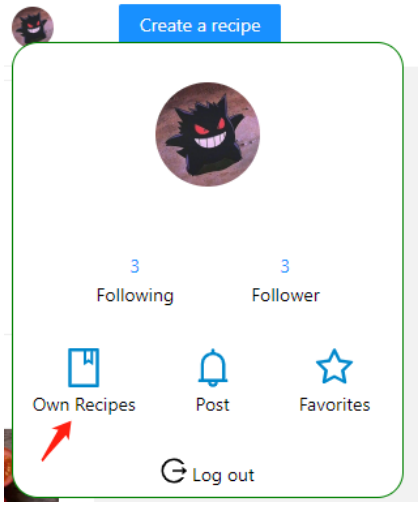

Figure 3.3.11 Own Recipes Button

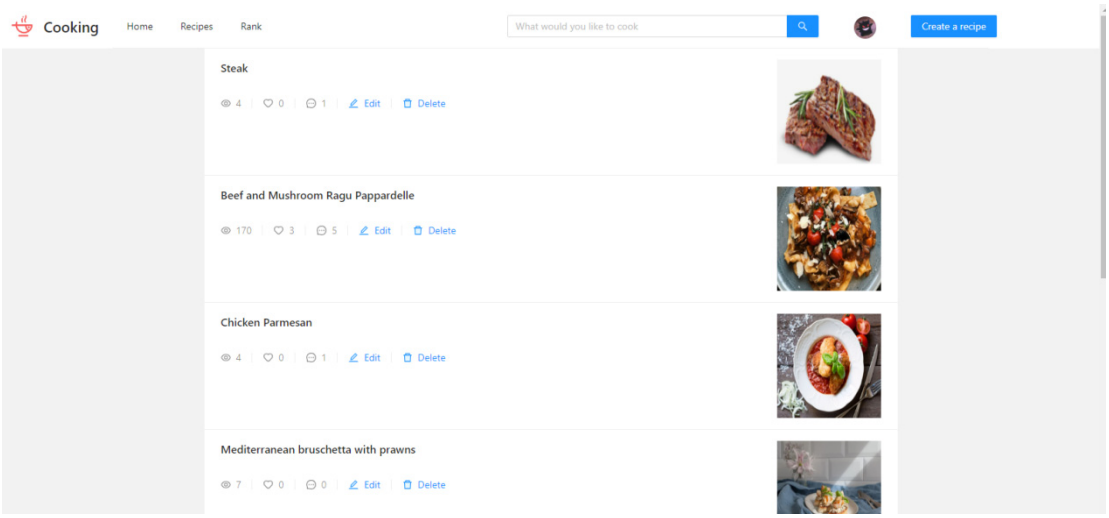

Figure 3.3.12 Own Recipes Page

3.3.2.1 Modification of Recipes

Select the recipe you want to modify on the own recipes page, click the Edit button and then click the Yes button to enter the recipe modification page (as shown in Figure 3.3.13).

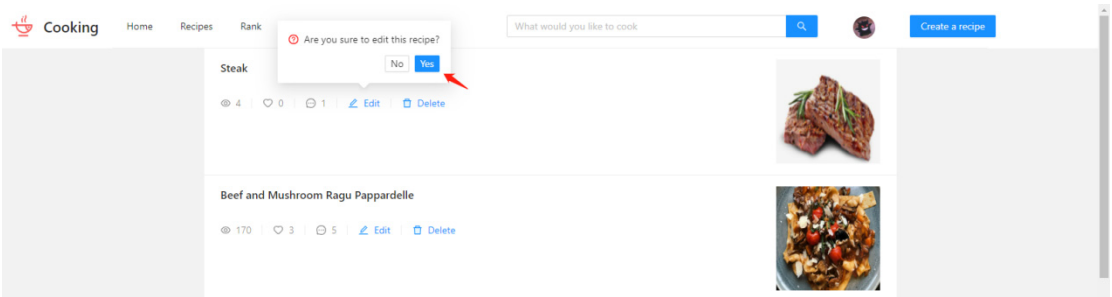

Figure 3.3.13 Own Recipes Page (Modification of Recipes)

On this page, you can modify the existing recipes. After the modification is completed, click

the submit button below to save the modified content (as shown in Figure 3.3.14).

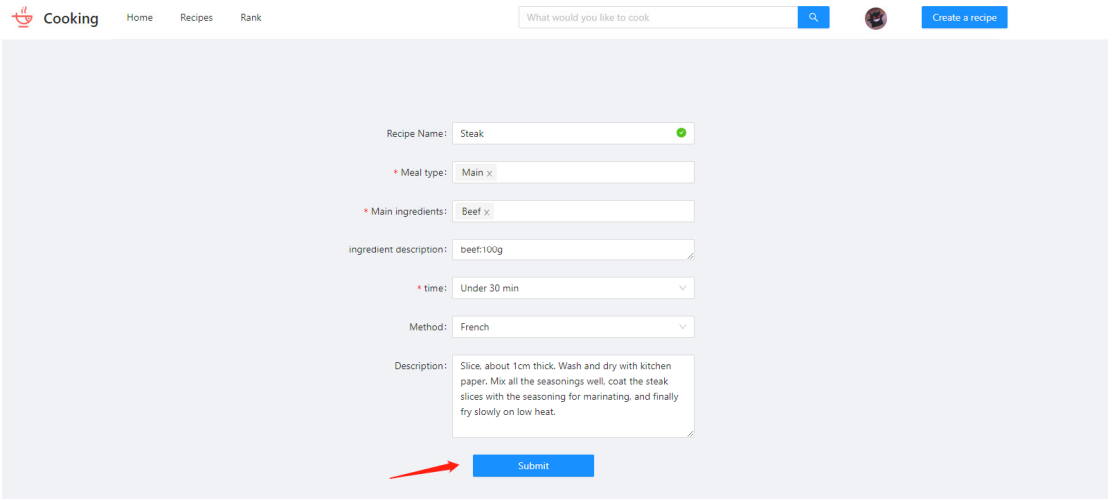

Figure 3.3.14 Recipe Information Modification Page

The user can enter the recipe homepage and see that the information has been modified successfully (as shown in Figure 3.3.15).

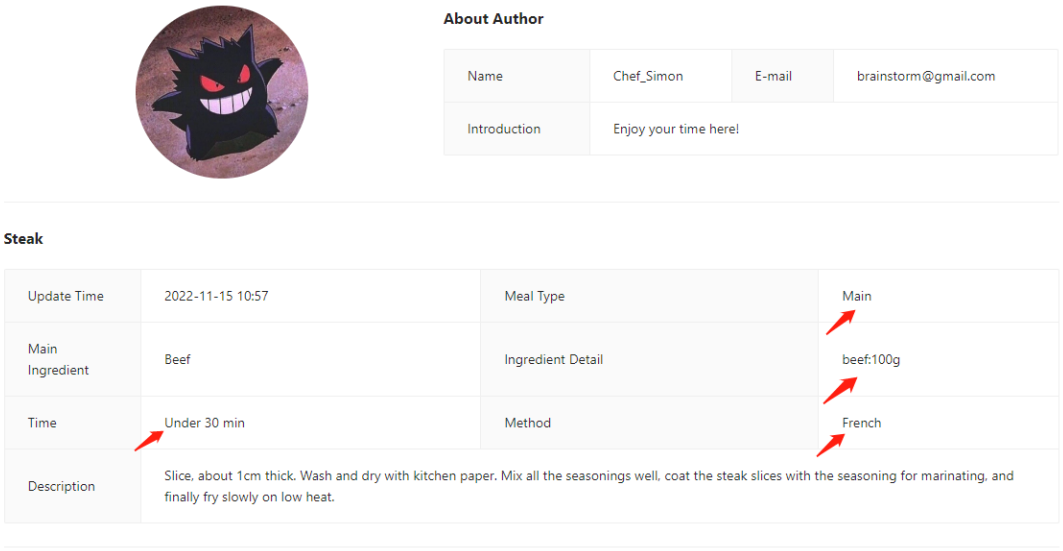

Figure 3.3.15 Recipe Homepage (Information Modified Successfully)

3.3.2.2 Deletion of Recipes

Select the recipe you want to delete on the own recipes page, click the Delete button and then click the Yes button to delete the recipe (as shown in Figure 3.3.16).

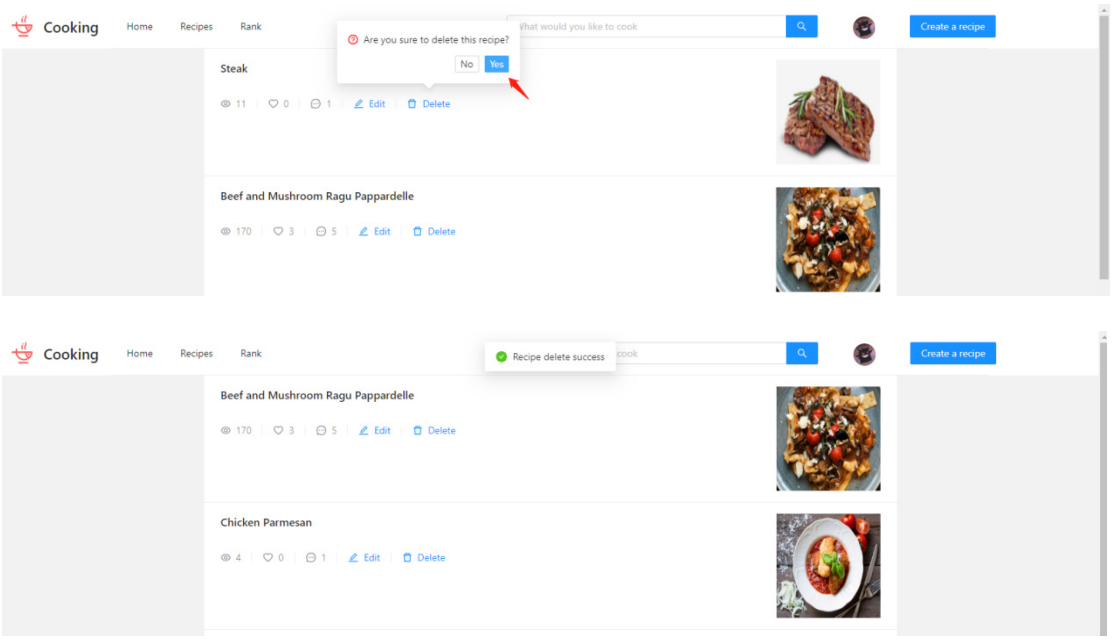

Figure 3.3.16 Deletion of Recipes

3.4 Like & Comment on Recipes

3.4.1 Like on Recipe Page

Users can not only browse the recipe page, but also interact with recipes, such as comments and likes.These two functions are only available to logged in users.

First is the like function. Users can see a heart-shaped icon in the lower right corner of the recipe image on the recipe page.

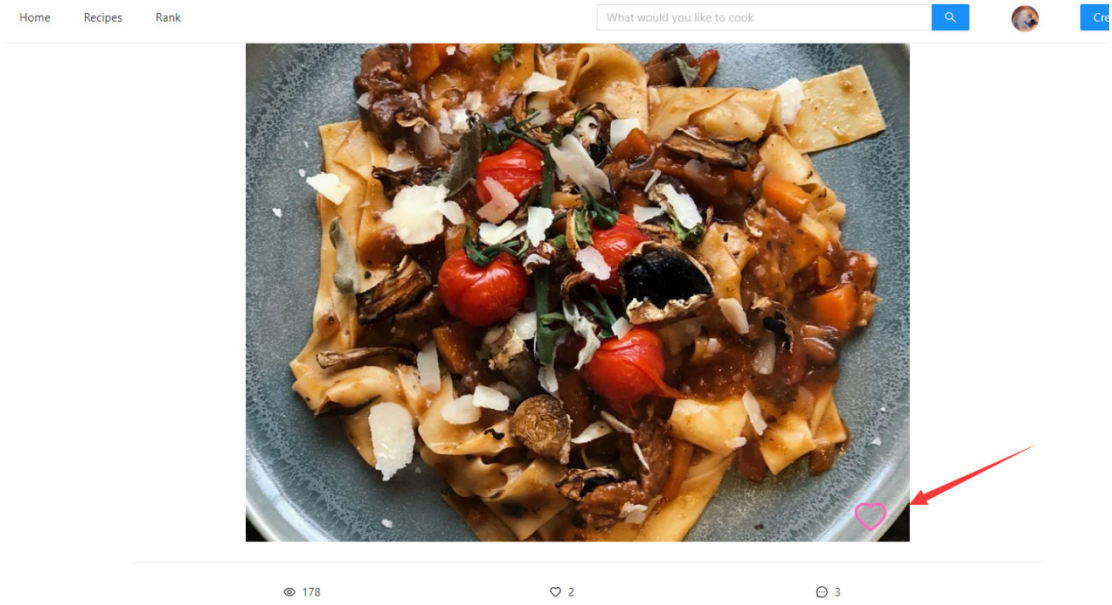

Figure 3.4.1 Like Icon on Recipe Page

Users can click this icon to like this recipe. After clicking this icon, the heart shape will change its style, and the likes statistics under the recipe image will also change.

Figure 3.4.2 Click Like Icon

If you click this icon again, your lik will be cancelled, and the statistics at the bottom of the picture will also be reduced.

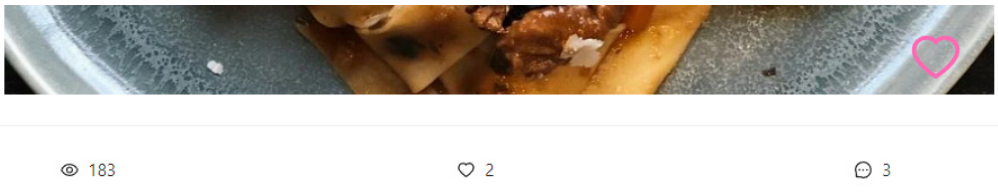

Figure 3.4.3 Click Like Icon Again

Users cannot like the recipes they create.. When the recipe owner tries to like your own recipe, the system will feed back error information.

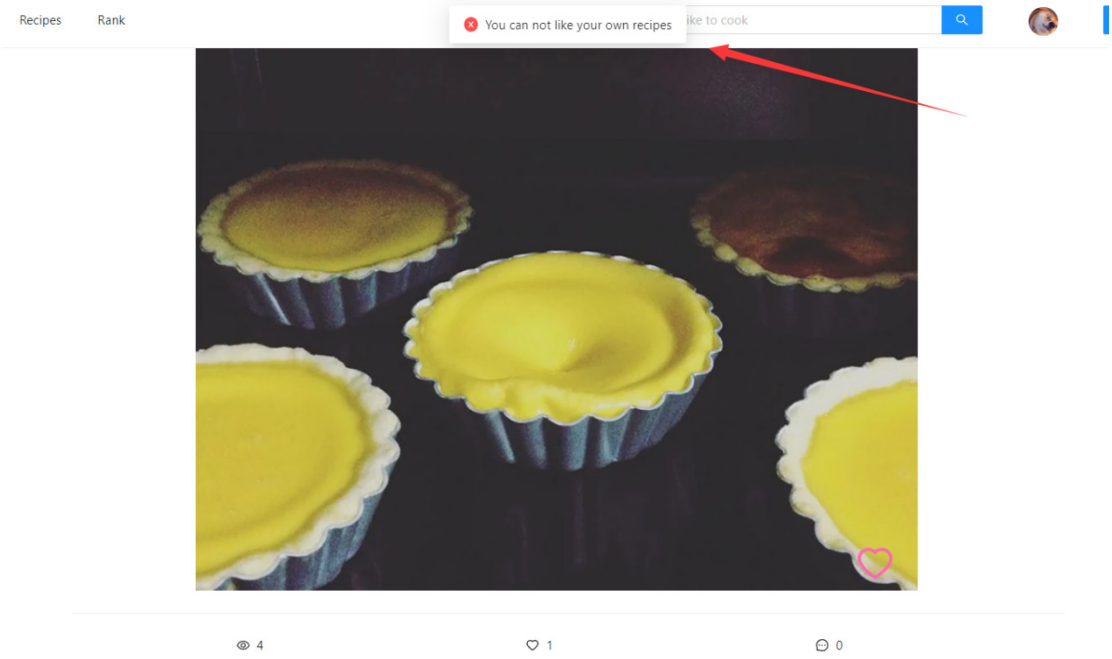

Figure 3.4.4 Click own recipe's Like icon

3.4.2 Comment on recipe page

The comment function is located at the bottom of the recipe page.

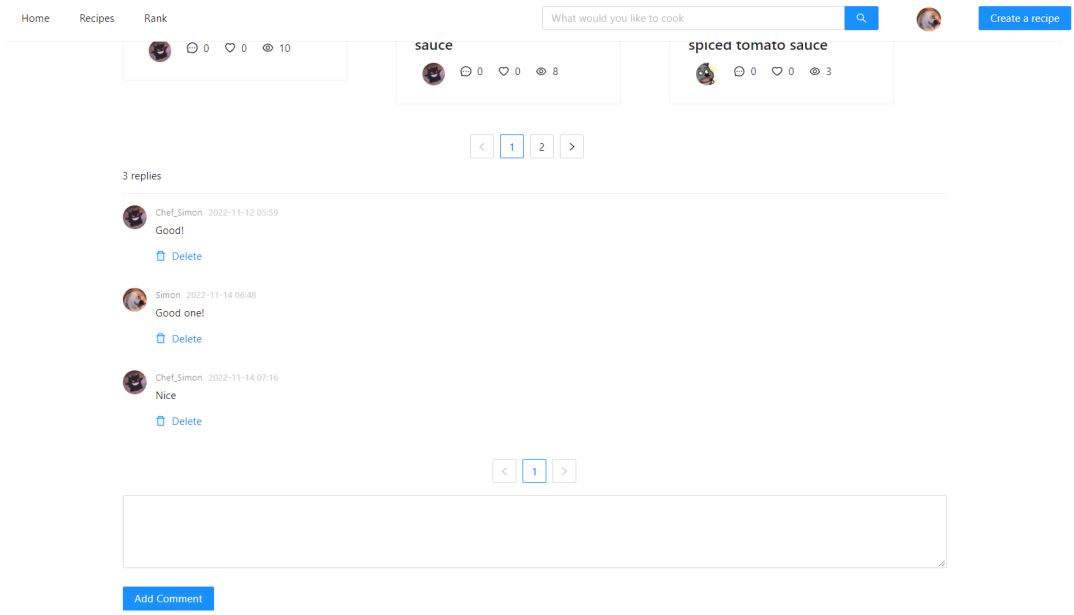

Figure 3.4.5 Comment on Recipe Page

After entering the content you want to comment on in the lower input box, click the Add Comment button. After a successful comment, the system will pop up a success message at the top of the page, and the user's comments will also be displayed.

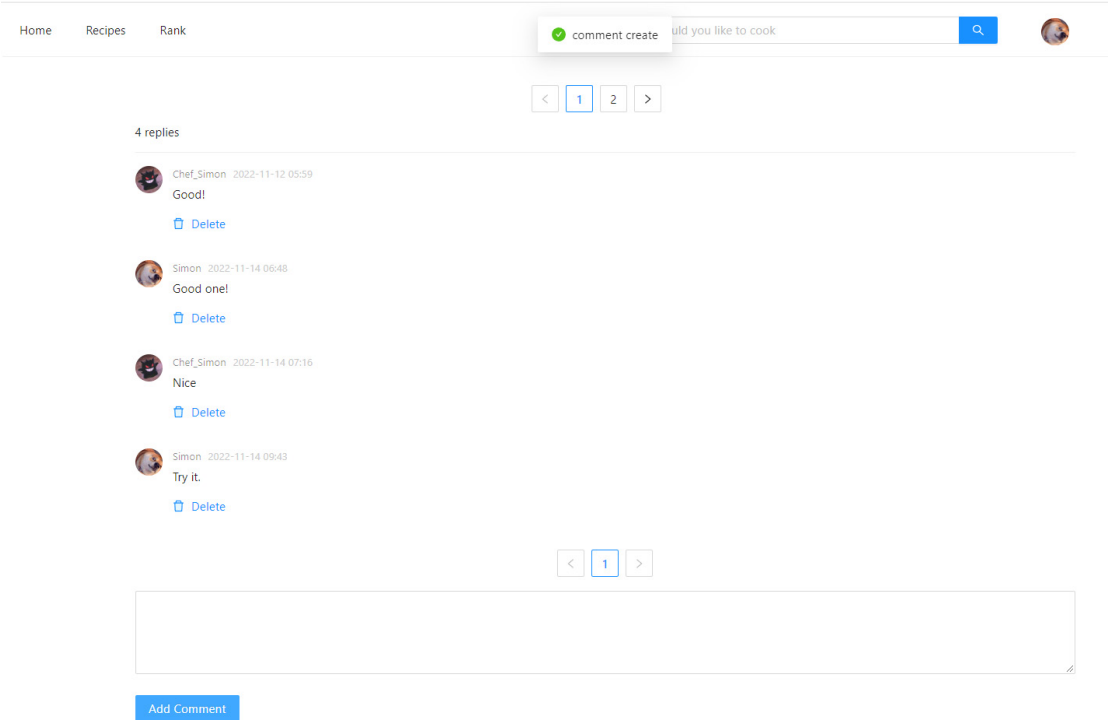

Figure 3.4.6 Add a Comment

Users can click the delete button to delete their own comments.

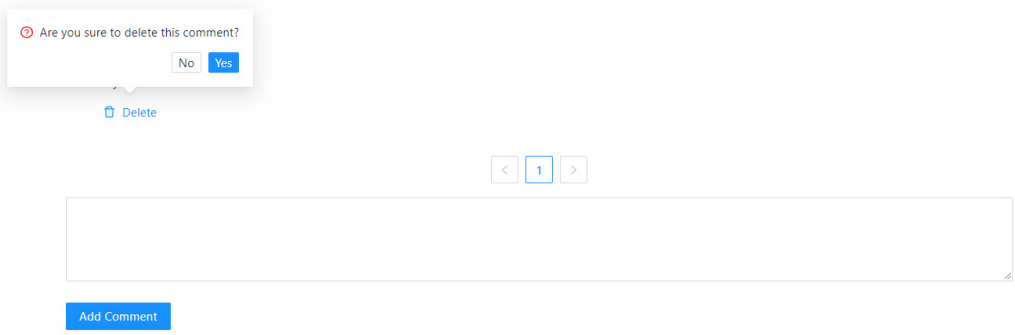

Figure 3.4.7 Delete the Comment

The creator of a recipe can delete all the comments under his recipe, but the non creator cannot delete other people's comments. If the user tries to delete, an error message will pop up.

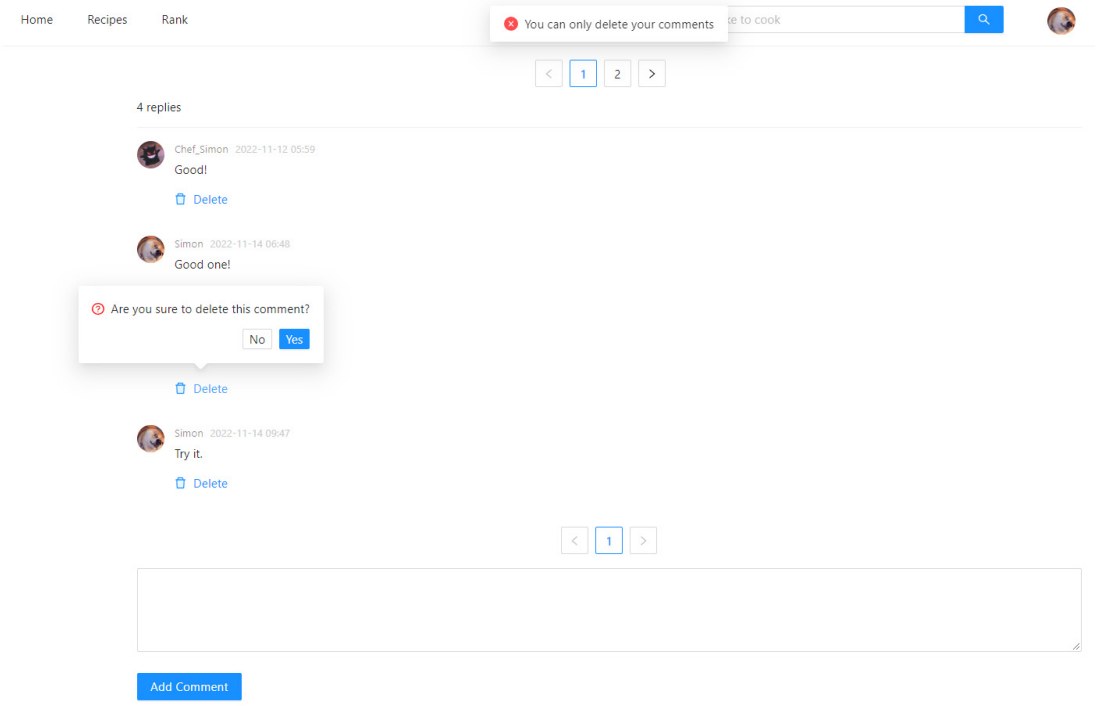

Figure 3.4.8 Try to Delete Own Comment

As a tourist, the user can also view the specific contents of one recipe. It should be noted that the tourists will not be able to use the comment function of the recipe page, nor can they view the comments on one recipe published by others.

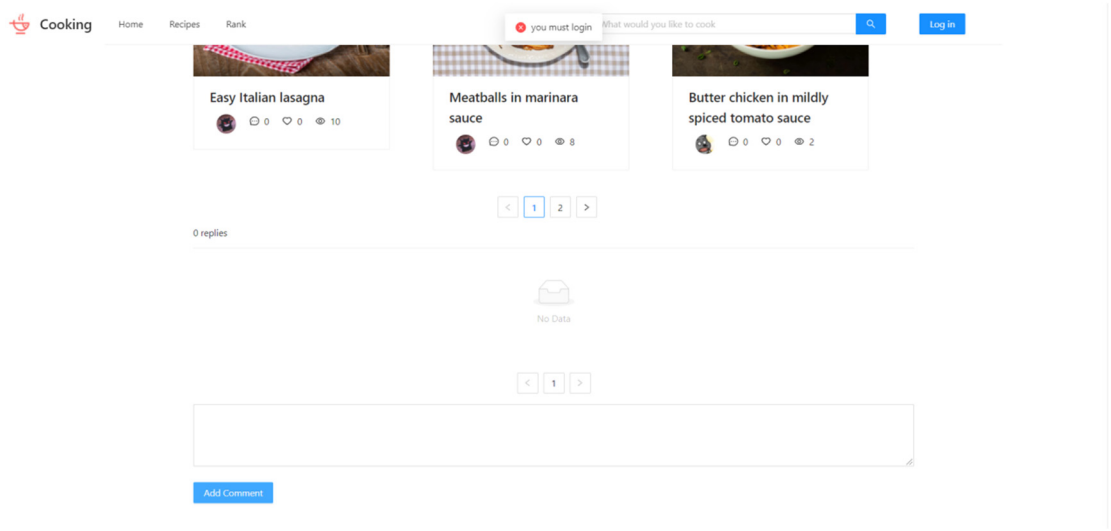

Figure 3.4.9 Error Message without Login

3.5 Favorites Module

The Favorites function is located at the lower right corner of the user information panel. Click the Favorites button to access the Favorites page.

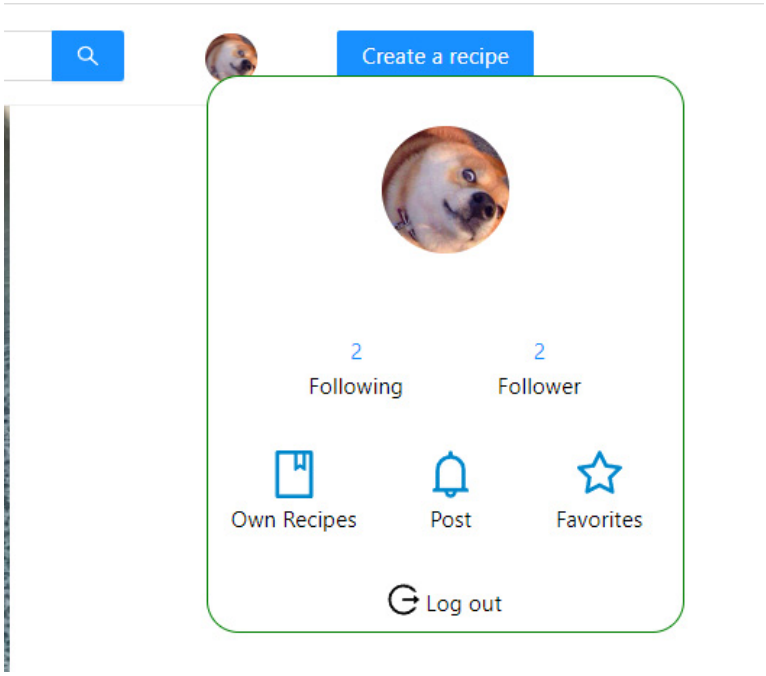

Figure 3.5.1 Favorites Function on User Panel

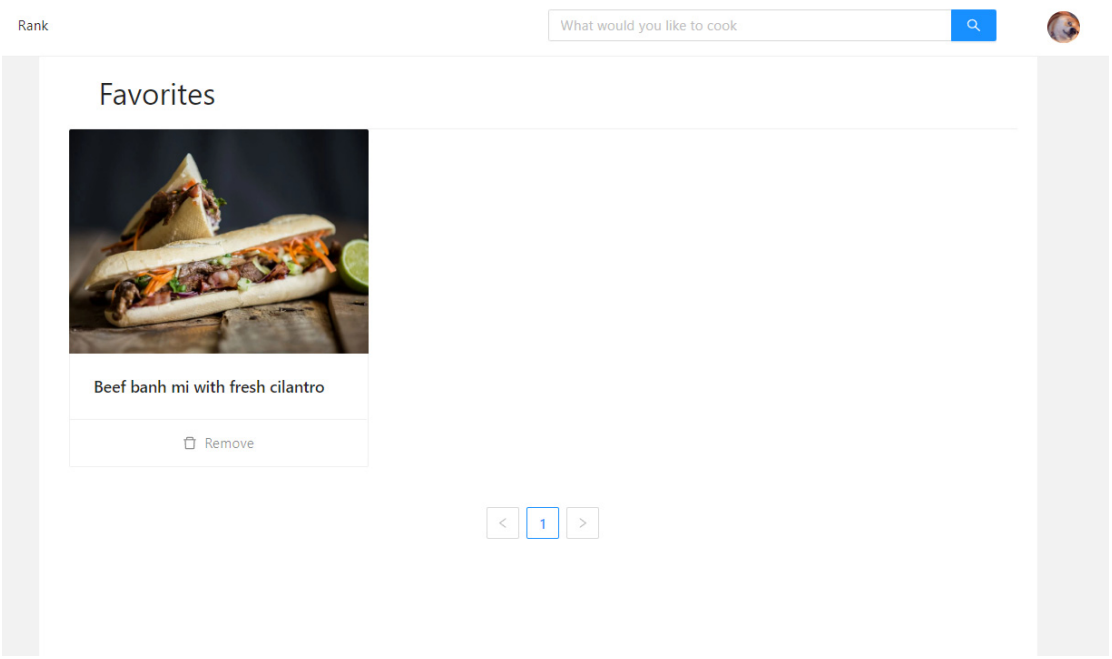

Figure 3.5.2 Favorites Page

The favorites function is linked with the like function. When you like a recipe, it will be added to your favorites.

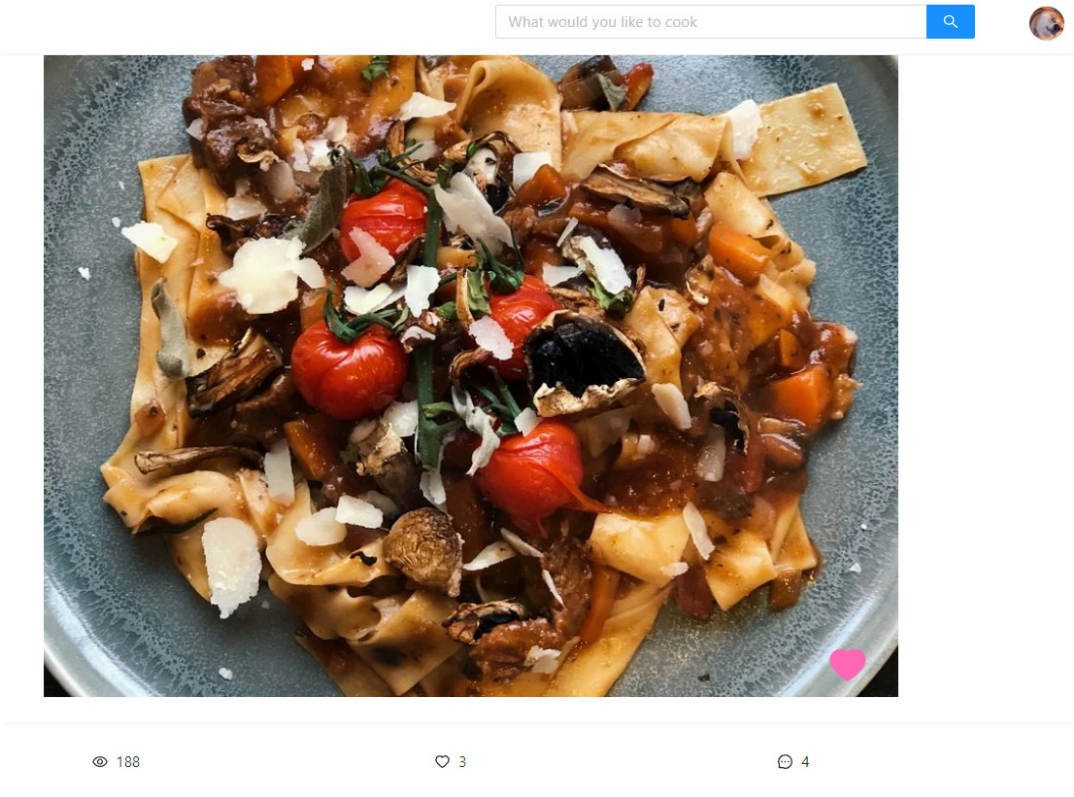

Figure 3.5.3 Favorites with Like

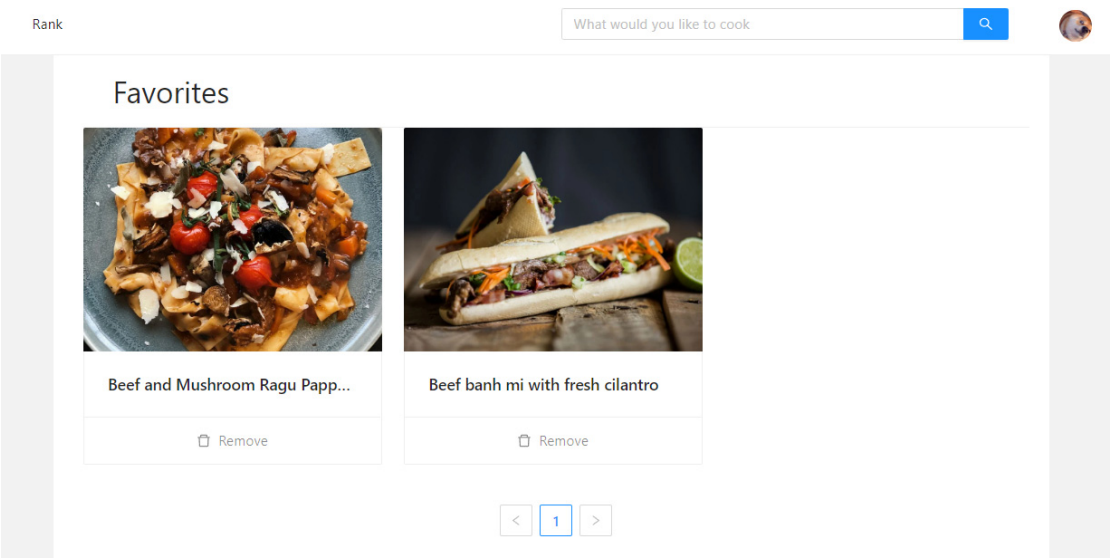

Figure 3.5.4 After Like a Recipe

In Favorites, you can click the picture to directly access the recipes you want to view. Alternatively, you can click the remove button to remove the recipe from your favorites.

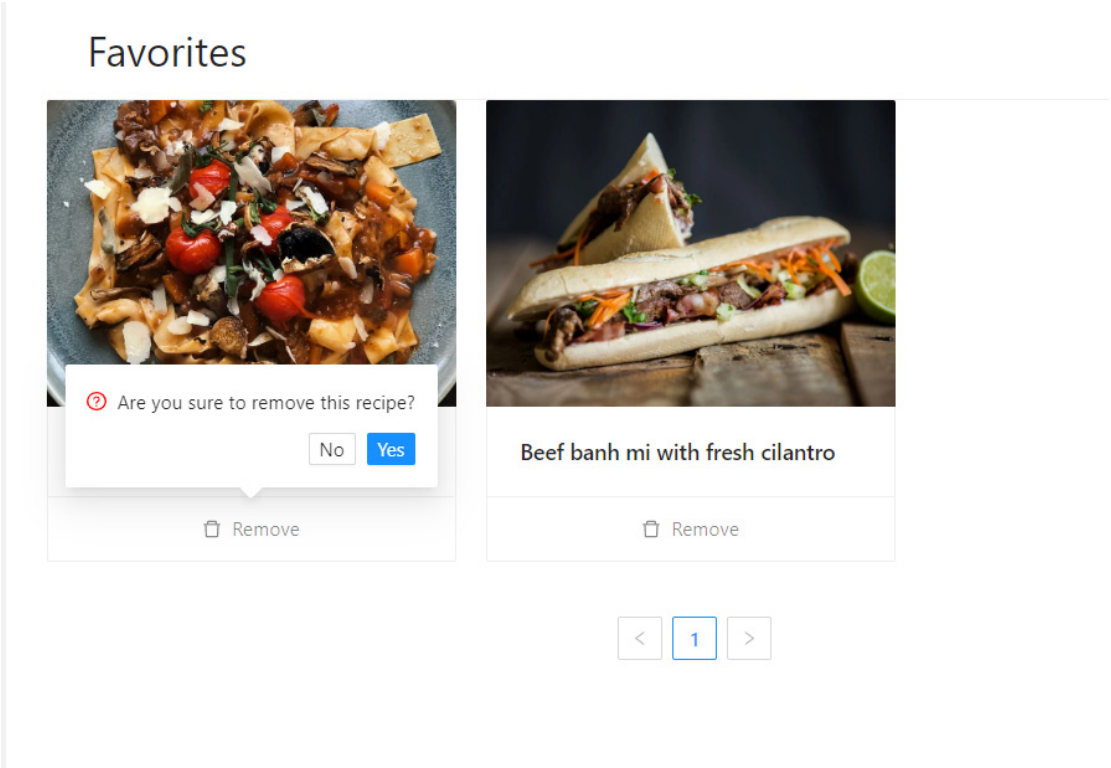

Figure 3.5.5 Remove a Recipe

3.6 Subscribe Module

Subscribe Module is one of the novel functionalities in our system. User can see the number of other users who he/she follow and the number of fans in the lower right corner. We can see that

this user has 2 following and 3 followers.

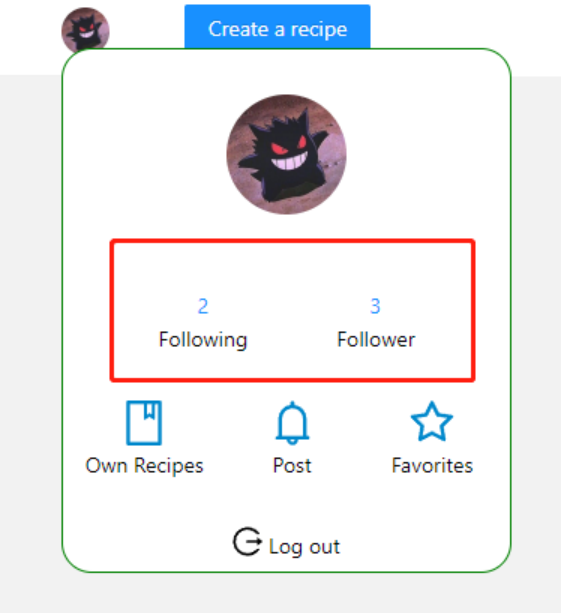

Figure 3.6.1 Show the Number of Following/Follower

If you want to subscribe some other users, you can go into his/her personal information page and click the follow button. If you have followed him/her, the button will change to unfollow and you can click it to unsubscribe.

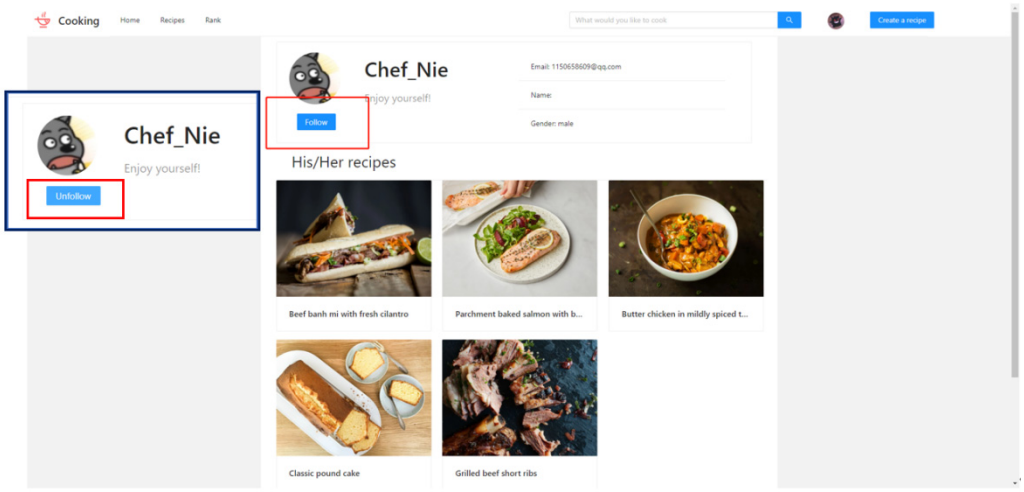

Figure 3.6.2 Follow/Unfollow Button

User can check the personal following and follower list by click the number in Figure 3.6.1. This list will show the basic information of other users and you can click his/her name to enter his/her personal information page.

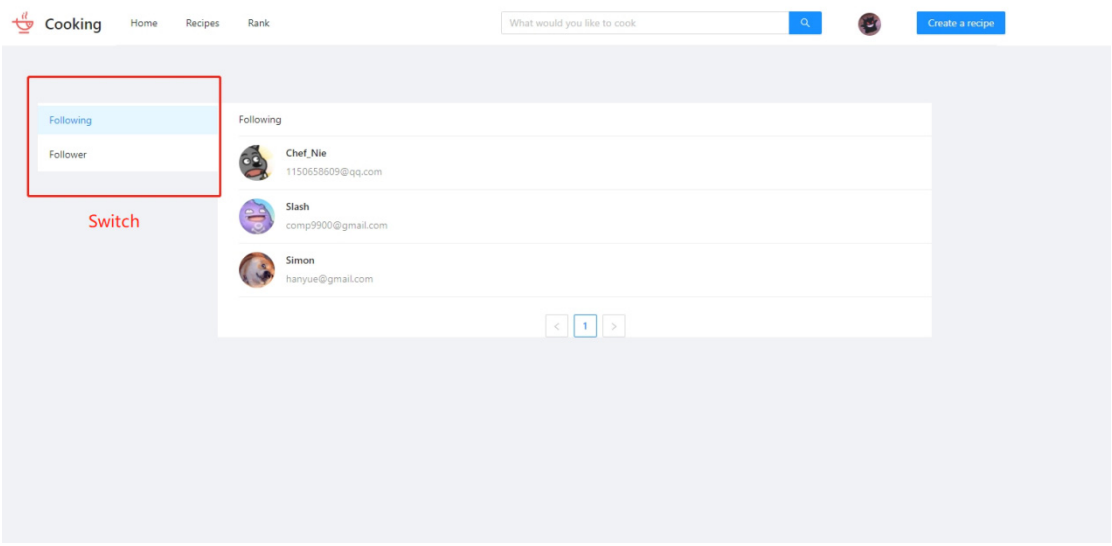

Figure 3.6.3 Following/Follower List

Then, user can also see the post, which cover all the recent recipes created by users I subscribe to. Click the post button show in Figure 3.6.1, you can enter into. User can see the name, picture and some basic information about recipes. In addition, the post list is ranked by modified time. If you want to see more, you can click a recipe entry to go into its information page.

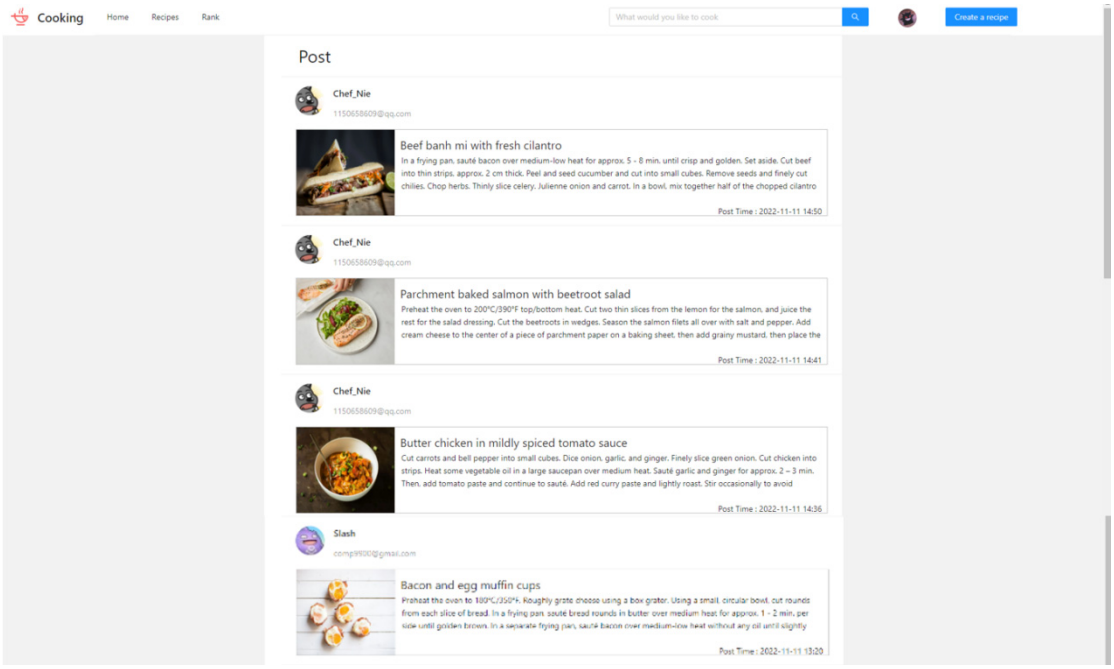

Figure 3.6.4 Post List

3.7 Search Module

As for the search module, there is a text input frame in the navigation. So, user can see it in each page, and this function is also applied to anytime. User can input what he/she want to search in this frame. After enter the text and click button, the system will jump to the search page and get

the result. For example, input ‘beef’.

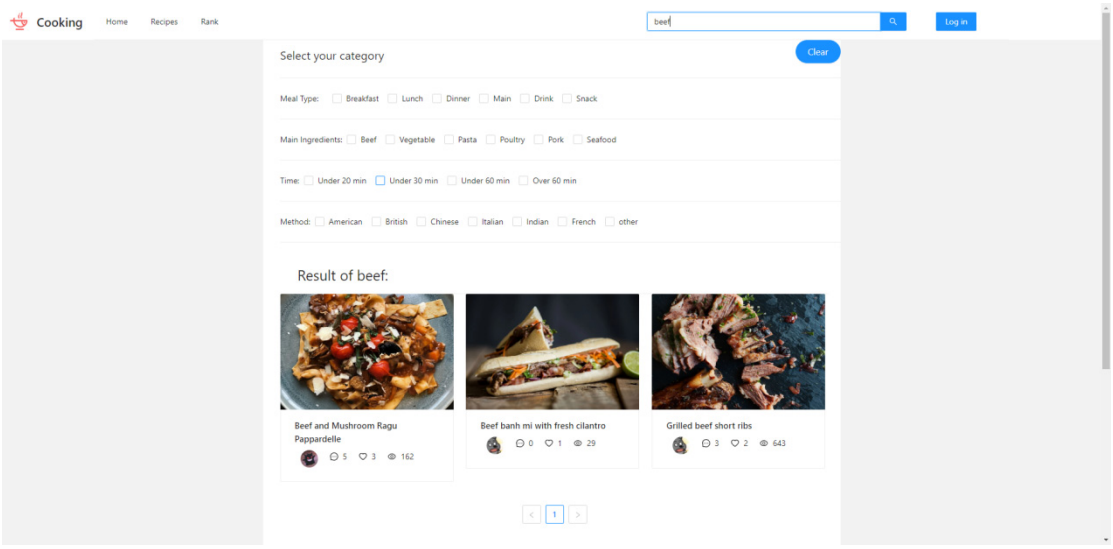

Figure 3.7.1 Search Page(‘beef’)

This system also supports the fuzzy search, which means in addition to the search string itself, the system will also display recipes containing this string in the text. For example, if user search ‘be’, it will also return the result includes recipes, which contains ‘beef’, with another recipe, which contains ‘be’.

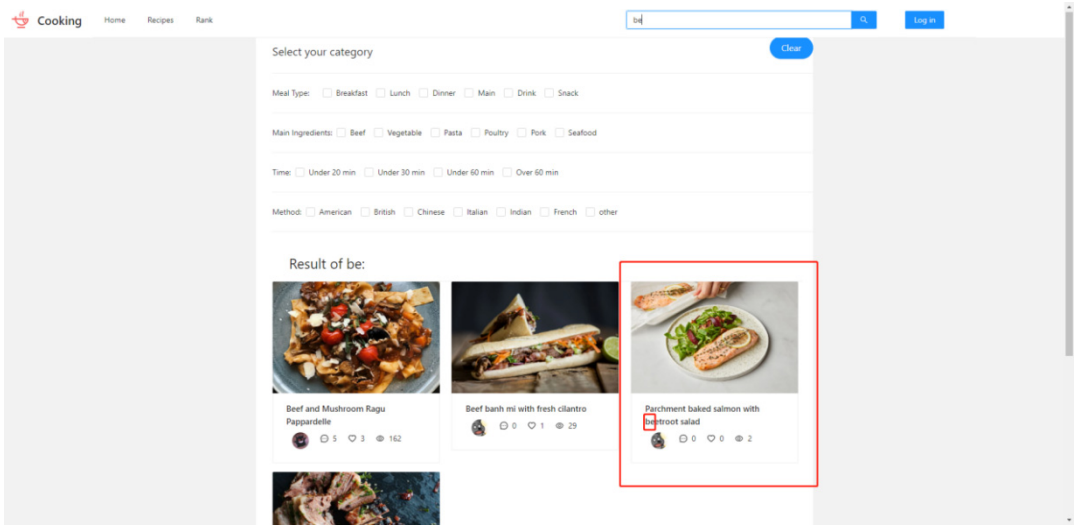

Figure 3.7.2 Search Page(‘be’)

Another way to enter the search page is to click the recipe button on the left of navigation.

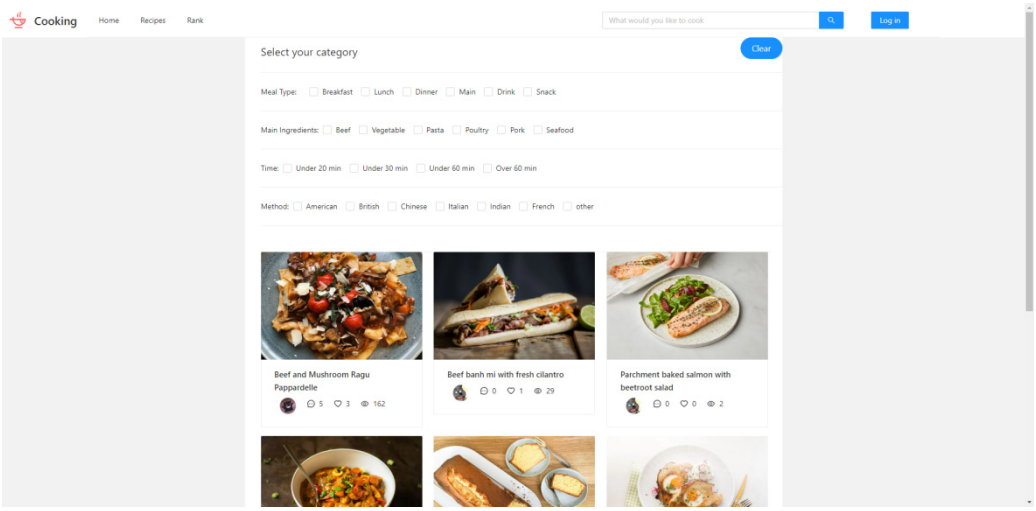

Figure 3.7.3 Search Page (without Input)

The system provides users with various classification options. User can search any combination of categories. If user select more than one option in same label, it will show the union result. For example, if you select Chinese and Italian in method, it will show Chinese recipes and Italian recipes. (Figure 3.7.4).

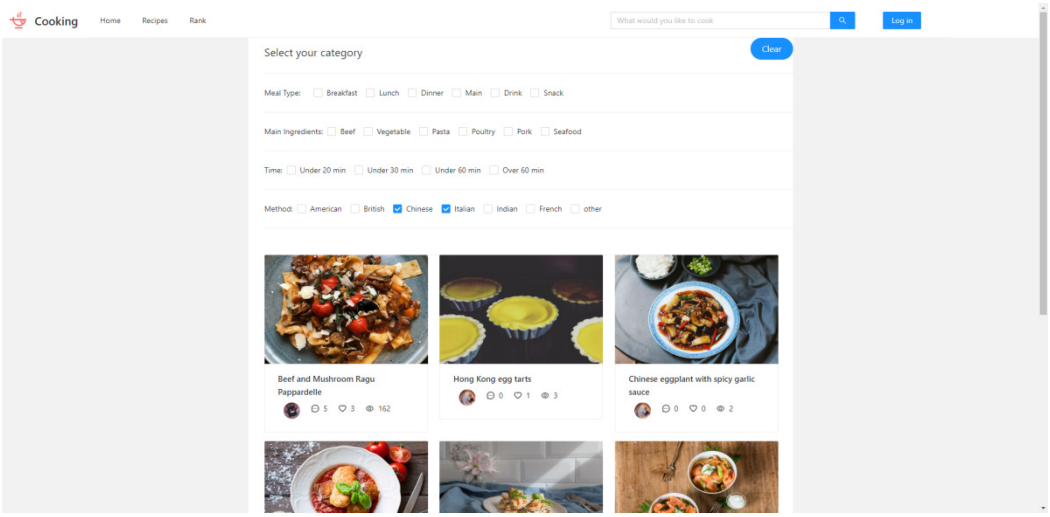

Figure 3.7.4 Search Page (Chinese and Italian)

If user select more than one option in different label, it will show the intersection result. For example, if you select Italian in method, and Under 20 min in Time, it will show Italian recipes which time is under 20 min. (Figure 3.7.5).

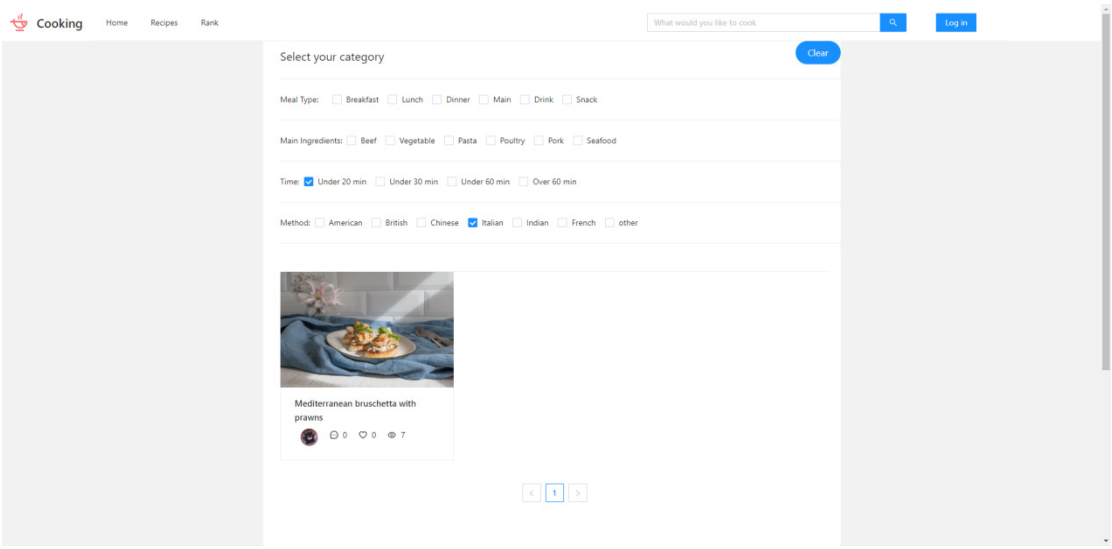

Figure 3.7.5 Search Page (Under 20 min and Italian)

In addition, the system also supports the mix search by text input and selection box. The result is intersection of them. For example, if you input the text 'beef' and select the Main in meal type, it will show 'beef' recipes which is the main. (Figure 3.7.6).

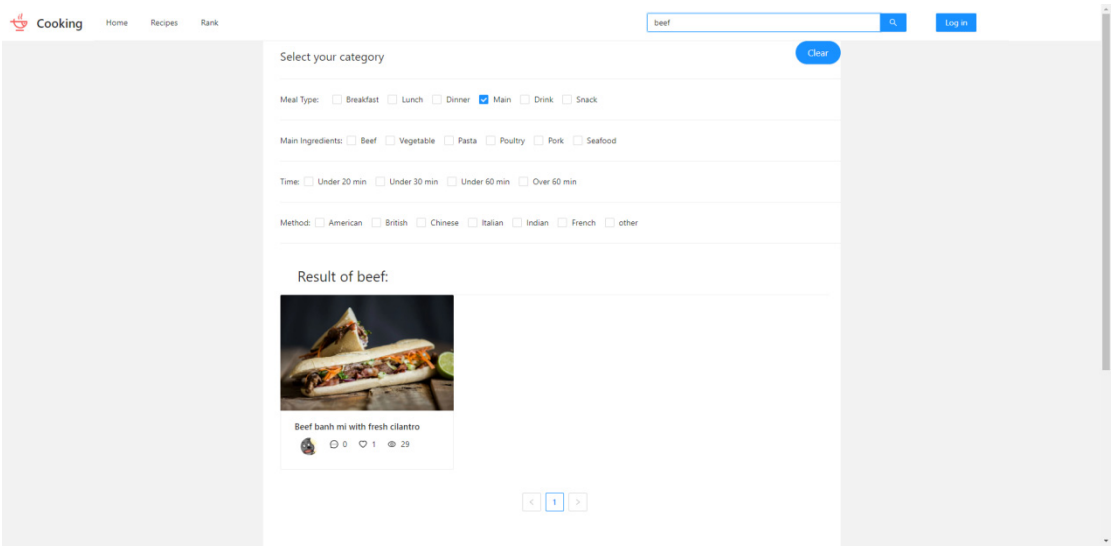

Figure 3.7.6 Search Page ('beef' and main)

3.8 Recipe Ranking

Recipe ranking is a novel functionality of our system. User can get to Ranking Page from the Rank button in the information bar. When user reached Ranking Page, user will see a top-10 recipe recommendation list sorted by popular rate. Ranking list demonstrates recipes with recipe name, author, thumbnail, and its popular rate. User can click recipe name then user will be directed to the recipe's main page.

Above the ranking list provides a time filter with three selections: within today, within 1 week

and within 1 month. Select any filter then the ranking list will only include recipes in the target time period.

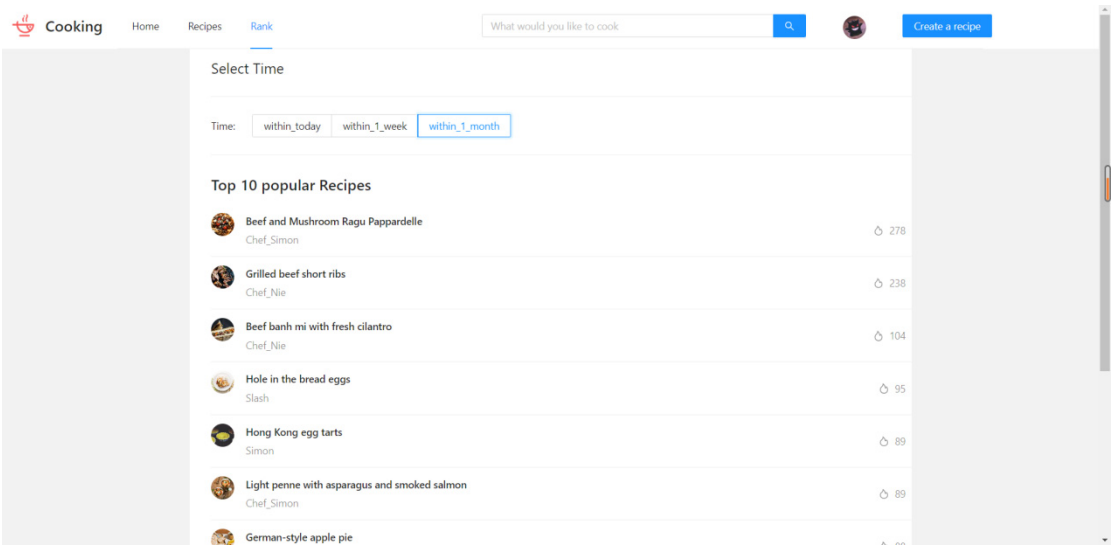

Figure 3.8.1 Time Filter

The popular rate is related to last modified time, interactions (views, comments, likes) and its author. We designed a formula to calculate popular rate:

$$P = \frac{(100 + 1*V + 5*L + 10*C)*1.1^{log(F+1)}}{Exp(k*(T-T_{last}))}$$

P: popular
V: view number
L: like number
C: comment number
F: follower number from author
K: attenuation coefficient
T: current time
T_{last}: last modified time

More explanations will be shown in Section 5 **Challenge Implementation**.

3.9 Recommendation System

Our recommendation system provides recommendation for recipes that are similar to a target recipe. The recommendation is shown in Recipe Main Page. Below the recipe information there's a sub title --- More delicious ideas for you. And you can see the recommend recipes with a thumbnail, author avatar and recipe stats. Recipes are ordered by similarity rates and there will be at most 5 recipes in the recommend list. If a recipe has no similar recipes, the recommend list will have no data.

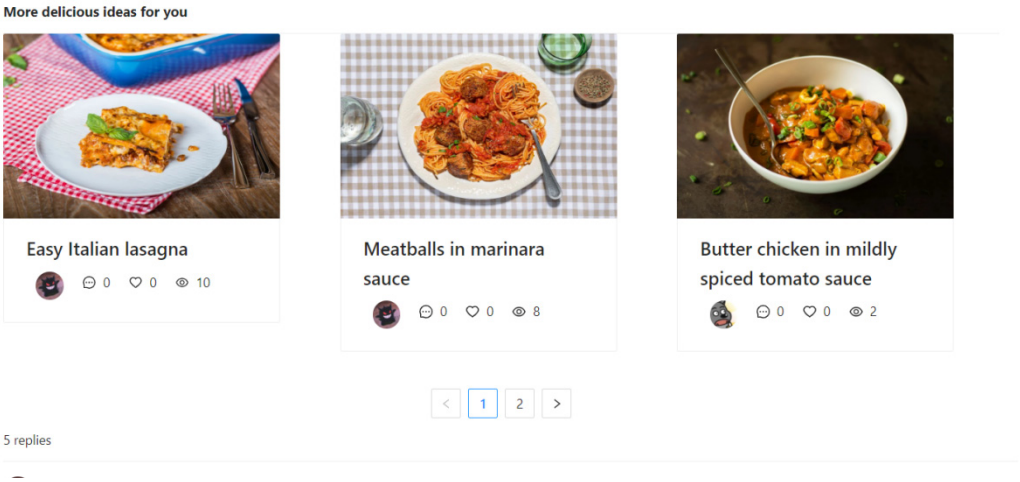

Figure 3.8.2 Recommendation for Recipes

The recommendation is based on the target recipe's ingredients. In order to calculate the similarity between two ingredients sets, we use Jaccard Similarity as our metric.

$$Jaccard(X,Y) = \frac{X \cap Y}{X \cup Y}$$

Jaccard Similarity is used to compare the similarity and difference between finite sample sets. The larger the Jaccard coefficient is, the higher the sample similarity is. And it is defined as the ratio of the size of the intersection of set A and B to the size of the union of set A and B. So, for a target recipe, we will get its ingredients from database and make a set A, then we go through any other recipe from database and get set B, then get a Jaccard Similarity list. Finally, we get top-5 recipes with highest similarity. And recipe with 0 similarity will not be included.

4. Third-Party Function

4.1 React

React is a JAVASCRIPT library for building user interfaces.React is mainly used to build UI. Many people think React is the V (view) in MVC.React originated from the internal project of Facebook, which was used to set up Instagram website and was open source in May 2013.React has high performance and simple code logic. More and more people have begun to pay attention to and use it.

The project itself has grown from the earliest UI engine to a complete set of front-end and back-end Web App solutions. The derived React Native project has a more ambitious goal. I hope to write Native App in the way of writing Web App. If it can be implemented, the entire Internet industry will be subverted, because the same group of people only need to write the UI once to run on the server, browser and mobile phone at the same time.

React is licensed under MIT License. Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files.

4.2 Antd

Antd is a React UI component library based on Ant Design design system, which is mainly used to develop enterprise level middle and back office products.

It has the following characteristics. The interactive language and visual style extracted from enterprise level middle and back office products. High quality React components out of the box.Developed using TypeScript to provide a complete type definition file.Full link development and design tool system.Dozens of international languages are supported.The ability to customize topics in depth for each detail.

Antd is licensed under MIT License. Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files. Its copy right is from 2015 - present.

4.3 Flask-RESTX

Flask-RESTX is an extension for Flask that adds support for quickly building REST APIs. It provides a coherent collection of decorators and tools to describe APIs and expose its documentation properly (using Swagger).[4]

Flask-RESTX is licensed under BSD 3-Clause License. It is granted to any person obtaining a copy of this software for free and associated documentation by author @noirbizarre. Its copyright is from 2020 – present.

4.4 Flask-SQLAlchemy

Flask-SQLAlchemy is an extension for Flask. It is a ORM framework that add support to Flask applications. It provides grammar to do CRUD so you don't need to write SQL query and this prevents SQL injection attacks. Flask-SQLAlchemy also support data migration which will make convenience to update the database.

Flask-SQLAlchemy is licensed BSD 3-Clause "New" or "Revised" License. Commercial use, Modification and Private use is permissioned. Its copyright is from 2010 – present.

4.5 User authentication

Our system uses JWT to do API authentication. When a client wants to start interacting with the API, it does Basic Auth authentication with an email as account name and password, and then gets a temporary token. As long as the token is valid, the client can send an API request with the token to pass authentication. Once a token expires, you need to apply for a new token. And all the logic to implement this is in backend/apis/auth.py file. And code in this file in referred from an opensource project from GitHub and its link is give below:
https://github.com/wangy8961/flask-vuejs-madblog/blob/master/back-end/app/api/auth.py

This opensource project is licensed under MIT. Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files. Its copyright is from 2017 – present.

5. Challenge Implementation

5.1 Front-end Part

5.1.1 Numerical Interaction Problems

5.1.1.1 Problem Statement

Numerical interaction problems caused by code running life cycle.

In the process of code running, you need to consider which code to execute first, which code to execute later, and the impact of repeated code calls. For example, when a user accesses a recipe, the browsing volume in the recipe needs to be changed in real time, as well as when the user comments or likes it. How to complete these immediate responses in react development and determine the execution order and update mode of each functional component when the page is loaded is a very important problem.

5.1.1.2 Solution

The solution is to use the life cycle and component monitoring functions in react. Usereffect() is a method that will be automatically executed when a new page (component) is accessed in react. With this method, we can enable the page to be loaded at the beginning, and execute those methods involving value transfer and assignment. In this way, when the page is loaded, there will be no problem that the value of a certain place is not loaded.

On the other hand, functions with preconditions can be executed after the key functions are completed by means of logical judgment such as if and while. In this way, we can simulate a life cycle environment. On the other hand, the components in react can set the monitored objects, which can be strings, arrays, etc. When the monitored objects change, the components will be re executed. With this feature, we can achieve real-time update when the value in the page changes due to the user's behavior.

5.2 Back-end Part

5.2.1 Design of Popular Formula

5.2.1.1 Problem Statement

Recipe ranking by popular is a novelty of our system and we want to design a formula to calculate the popular rate.

5.2.1.2 Solution

After discussion, we find that there are three dimensions which have impact on popular rate.
Frist one is time. Popular should decrease with time because we want our ranking list have

more recipes update recently. And older recipes are more likely to have great stats with views, comments, likes. If popular doesn't decrease, these recipes will rank high on the list and will make users feel that our web update too slow. And the popular rate should not be linear decrease because it will make a recipe maintain a high rank for a long time. So, we decide to make popular rate has an Exponential fall with time.

Second one is interaction, popular should increase with interactions because more views, likes, comments show that more users are interested in this recipe. And different interactions should have different weights, because different interactions reflect different levels of user interest. Comment should have the highest weight, like is the second and view is the last.

Third one is User. We want to reward users with more followers with higher popular rate because their recipes will be shown in more users' personal news feed. And through this reward rule. And we don't want the popular rate increase too much, so we use the logarithm of author's follower in the final formula.

Considering all these above, we finally designed the formula:

$$P = \frac{(100 + 1*V + 5*L + 10*C)*1.1^{log(F+1)}}{Exp(k*(T-T_{last}))}$$

P: popular
V: view number
L: like number
C: comment number
F: follower number from author
K: attenuation coefficient
T: current time
T_{last}: last modified time

In the formula, k is just an attenuation coefficient to adjust the gradient of descent. And rest symbol is stats from database.

5.2.2 Database Interaction

5.2.2.1 Problem Statement

Our database has some Many-to-Many relationships, this relationship requires a helper table to store foreign key pairs. So, it will be complicated to use join query when we try to search for some data. We want to find an easiest way to solve this problem.

5.2.2.2 Solution

As we use **Flask-SQLAlchemy** framework to do object relational mapping and when we are using **SQLAlchemy Model** class to create/modify our table in database, we tried to research more about **SQLAlchemy**. Finally, we find we can use **relationship** from **SQLAlchemy**[5]**.**

Relationship can create an object in your table **Model** class and can reference records in another table.

```python
class User(db.Model):
    __tablename__ = 'user'
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(64), index=True, unique=True)
    email = db.Column(db.String(128), index=True, unique=True)
    # password will not be stored as texture info
    password_hash = db.Column(db.String(128))
    last_seen = db.Column(db.DateTime(), default=datetime.utcnow)
    avatar = db.Column(db.String(128), index=True, default="/static/ava/default.jpg")
    name = db.Column(db.String(64))
    gender = db.Column(db.String(64))
    introduction = db.Column(db.Text, default="Nice to see you!")
    following_num = db.Column(db.Integer, default=0)
    follower_num = db.Column(db.Integer, default=0)
    favourite_num = db.Column(db.Integer, default=0)
    recipes = db.relationship('Recipe', backref='author', lazy='dynamic', cascade='all, delete-orphan')
    # self-reference relationship
    # object 'subscribed' will list the users you subscribed
    # object 'fans' will list the users who subscribe you
    subscribed = db.relationship('User', secondary=subscribe,
                                 primaryjoin=(subscribe.c.fan_id == id),
                                 secondaryjoin=(subscribe.c.contributor_id == id),
                                 backref=db.backref('fans', lazy='dynamic'), lazy='dynamic')

    comments = db.relationship('Comment', backref='author', lazy='dynamic',
                               cascade='all, delete-orphan')
```

Figure 5.1 Model for Table User

```python
subscribe = db.Table("subscribe",
                     db.Column("fan_id", db.Integer, db.ForeignKey("user.id")),
                     db.Column("contributor_id", db.Integer, db.ForeignKey("user.id")),
                     db.Column("timestamp", db.DateTime(), default=datetime.utcnow))
```

Figure 5.2 Helper Table

Here's the screenshot of our database User model. For **table User**, it is a more complex situation. It is a self-referential Many-to-Many relationship because a user's following users and followers are all come from **table User**. So the **subscribed** class attribute defines this self-referential Many-to-Many relationship, parameter secondary referenced **help table subscribe**, parameter primaryjoin and secondaryjoin declared how to join three **table User** together and parameter backref adds a back reference that behaves like a column to the User model. So now when we get a User, we can use **User.subscribed** to get all users that User is following and use **User.fans** to get all the followers directly. This method will simplify the ORM operations in related APIs.

6. User Manual

This section will show you how to setup our web application on CSE VLAB environment. For Software Quality Assessment, we also choose CSE VLAB environment. For functionalities, we have well demonstrated them on section 3 **Functionalities,** so this section won't repeat again.

6.1 Window Size & Web Browser Requirement

Our system preferred to be run with window size 1920 * 1080 or bigger. If the window size is not big enough, some button on information bar will be invalid. So the port number we choose is 5920.

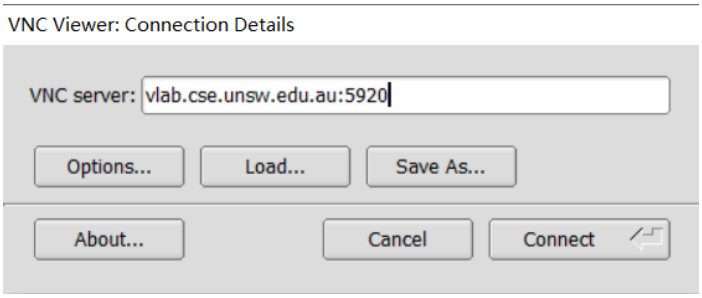

Figure 6.1 VLAB Connection

We test our system in Chrome and Firefox, so these two web browsers should both be fine.

6.2 System Setup

Unzip the Brainstorm-FinalSoftwareQuality.zip file then you will find directory back-end and directory front-end. A README is provided with User Manual. Or you can directly clone our codebase on GitHub.

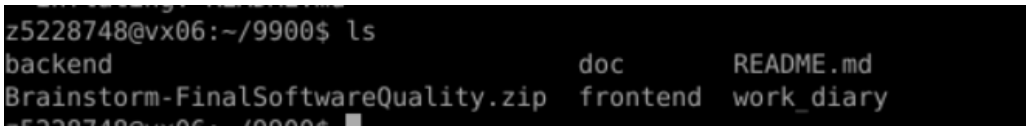

Figure 6.2 Files

6.2.1 Backend Setup

Python version on VLAB should be fine with our system and we don't find any problem when we testing. But if some problem occurs, we recommend to use Python 3.8.5.
Steps to set up backend:

1. cd "$your_path"/backend

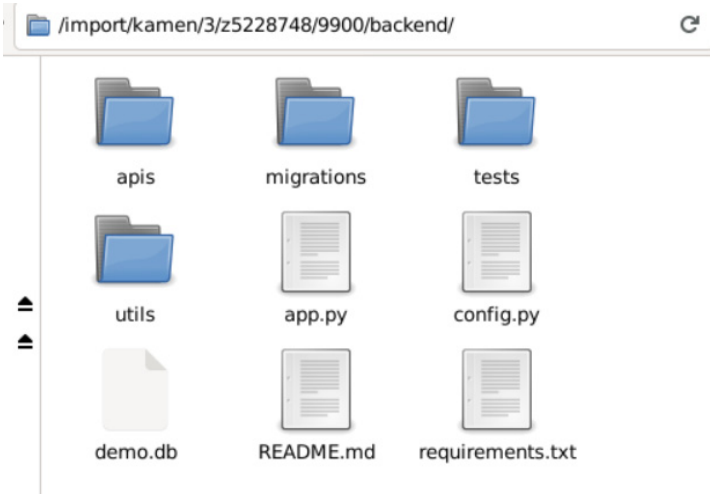

Figure 6.3 VLAB Back-end Folder

2. create a virtual environment by command:
   $ python3 -m venv venv
3. enter virtual environment by command:
   $ source venv/bin/activate

```
z5228748@vx06:~/9900$ cd backend
z5228748@vx06:~/9900/backend$ ls
apis    config.py  migrations  requirements.txt  utils
app.py  demo.db    README.md   tests
z5228748@vx06:~/9900/backend$ python3 -m venv venv
z5228748@vx06:~/9900/backend$ source venv/bin/activate
(venv) z5228748@vx06:~/9900/backend$ python3 --version
Python 3.9.2
(venv) z5228748@vx06:~/9900/backend$
```

Figure 6.4 Back-end Setup Step 1-3

Your terminal should look like the same with the Figure we provide if succeed.

4. install packages in requirements by command:
   (venv)$ pip install -r requirements.txt
5. migrate database by command:
   (venv)$ flask db upgrade
6. run backend:
   (venv)$ flask run

```
(venv) z5228748@vx06:~/9900/backend$ flask run
 * Serving Flask app "app.py" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployme
nt.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 329-809-740
```

Figure 6.5 Running on Port 5000

If all the steps succeed then you can find the system is run on Port 5000. You can use a web browser to http://127.0.0.1:5000/ then you will see our backend swagger.

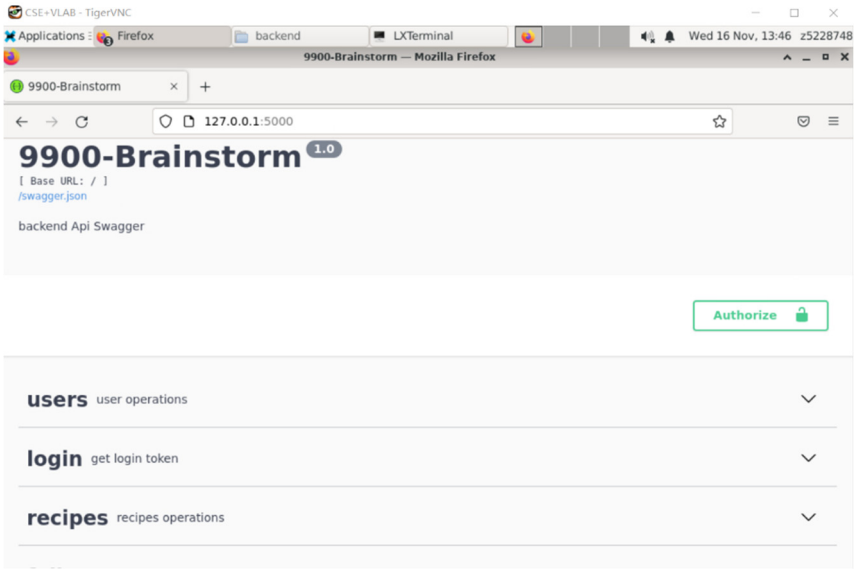

Figure 6.6 Back-end Swagger

6.2.2 Frontend Setup

make sure your environment has npm.
Steps to set up frontend:

1. cd "$your_path"/frontend
2. download node modules by command:
   $ npm i
3. run frontend by command:
   $ npm start

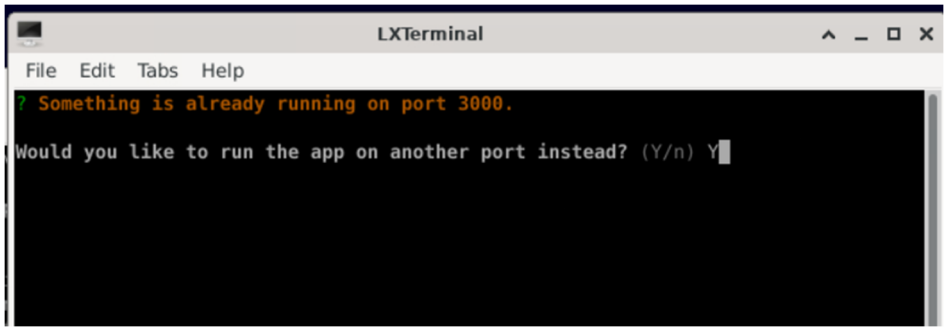

Figure 6.7 Already Running on Port 3000

On VLAB machine you may found something is already running on port 3000. To solve this, just press Y.
And if everything succeeds. You now will be directed to our web page. You can see that we have already put some data in the database. For some accounts we have created, please take a look at README.md.

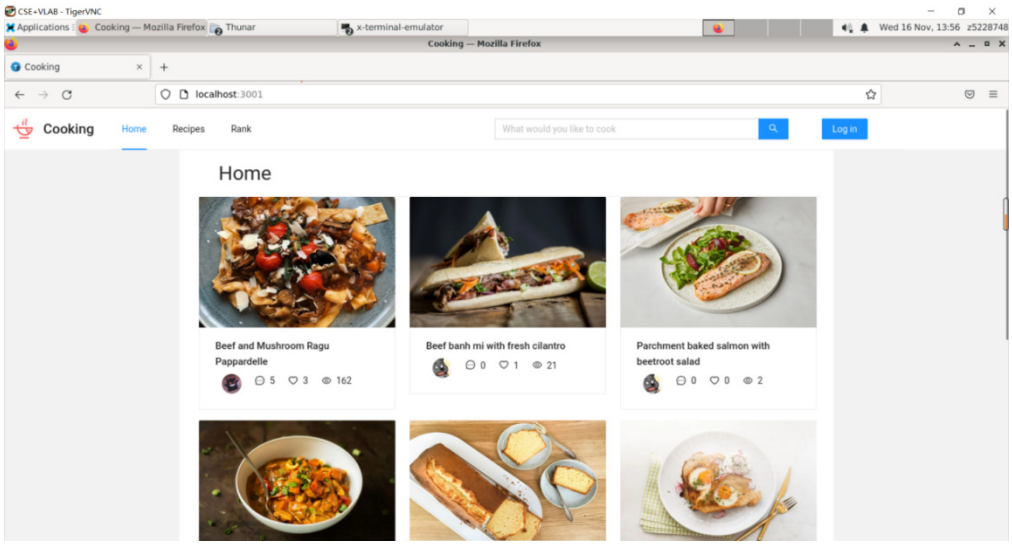

Figure 6.8 System Main Page

6.2.3 Unit Test

To validate the system, our backend includes unit test and we have already written some tests. Make sure when you run unit test, the backend and frontend system are not running.

Steps to check unit test:

1. cd "$your_path"/backend
2. enter virtual environment by command:
   $ source venv/bin/activate
3. run unit test by command:
   (venv)$ flask test

```
z5228748@vx06:~/9900$ cd backend
z5228748@vx06:~/9900/backend$ source venv/bin/activate
(venv) z5228748@vx06:~/9900/backend$ flask test
test_api_need_auth_with_jwt (test_backend.BackEndTest) ... ok
test_api_need_auth_with_no_jwt (test_backend.BackEndTest) ... ok
test_api_no_need_auth (test_backend.BackEndTest) ... ok
test_enter_no_exist_api (test_backend.BackEndTest) ... ok
test_followed_user_profile_status (test_backend.BackEndTest)
go to user main page you follow should respond True ... ok
test_get_jwt_token (test_backend.BackEndTest) ... ok
test_if_in_testing_config (test_backend.BackEndTest) ... ok
test_password_hash (test_backend.BackEndTest) ... ok
test_password_strength (test_backend.BackEndTest) ... ok
test_popular_formula (test_backend.BackEndTest)
old recipe with good stats should have lower popular ... ok
test_recipe_ingredient_structure (test_backend.BackEndTest) ... ok
test_self_user_profile_status (test_backend.BackEndTest)
go to self user profile should respond cur_user ... ok
test_unfollowed_user_profile_status (test_backend.BackEndTest)
go to user main page you do not follow should respond False ... ok

----------------------------------------------------------------------
Ran 13 tests in 1.844s

OK
```

Figure 6.9 Back-end Unit Test

If the setup is not successful or there are some problems, please contact Hanyue Jiang via z5228748@ad.unsw.edu.au.

Reference

[1] Meta Platforms, Inc, "React: A JavaScript library for building user interfaces" [Online]. Available: https://reactjs.org/
[2] XTech,"Ant Design of React" [Online].
Available: https://ant.design/docs/react/introduce-cn
[3] Unittest tutorial, “How To Use unittest”[Online].Available:
https://www.digitalocean.com/community/tutorials/how-to-use-unittest-to-write-a-test-case-for-a-function-in-python
[4] python-restx Authors, “Welcome to Flask-RESTX’s documentation!” [Online]. Available: https://flask-restx.readthedocs.io/en/latest/
[5] SQLAlchemy tutorial, “How to Use Database Relationships with Flask-SQLAlchemy” [Online] Available:
https://www.digitalocean.com/community/tutorials/how-to-use-one-to-many-database-relationships-with-flask-sqlalchemy