Project Title

**Global Task Bridge**

Old name: International Student Chef Handbook

Project Description – No Change

We are proposing to develop a web app tailored specifically for international students, aiming to transform the way they approach cooking and dining in a foreign country.

Our primary objective is to simplify the process of finding suitable recipes, learning about diverse cuisines, and managing their daily meals. By integrating multiple open - source culinary APIs, we plan to enable users to search for recipes based on a variety of criteria, such as available ingredients in local markets, budget constraints, cooking time, and personal taste preferences.

Functionally, the envisioned app will not only provide a powerful multi criteria search function but also offer unique features tailored to international students. For instance, it will include cultural insights for each recipe, explaining the origin and significance of the dish in its native culture. It will also have a section dedicated to substituting hard to find ingredients with local alternatives. Additionally, users will be able to create and manage their own meal plans, syncing them across different devices. The app will present recipes in an easy to follow format, with step by step instructions, high quality images, and even video tutorials for more complex dishes.

In contrast to general recipe apps, which lack the specific focus on international students' needs, our proposed app will be a one - stop solution for this unique user group. It will be publicly accessible and free of charge, removing any financial barriers for students.

The target user base of this app will be international students studying in various countries around the world. Whether they are undergraduate students just starting to learn cooking or postgraduate students juggling a busy academic schedule, this app will be an invaluable resource. This initiative holds great potential to fill an existing gap in the market by providing a comprehensive, user friendly, and cost effective solution for international students' culinary needs.

Progress Summary - New

Features Completed

**Story 1** - Multicriteria Recipe Search: Users can create filters specifying ingredients available in local markets, budget limits, cooking time, cuisine type, and taste preferences to obtain relevant recipe search results.

**Story 2** - Recipe Collection by Meal Plan: Users can create meal plans (like weekly or monthly) and save all relevant recipe search results under these plans.

**Story 3** - Recipe List Display: Search results are presented in an organized list, including essential details such as recipe names, cooking times, difficulty levels, and cuisine categories.

**Story 4** - Recipe Sharing: Users can share favorite recipes via social media platforms, email, or messaging apps with friends and fellow students.

**Story 5** - Video Guided Cooking: For complex recipes, users can access video tutorials that guide them step by step through the cooking process.

**Story 6** - User Review and Rating: Users can rate recipes and leave reviews, sharing their experiences and providing feedback for other users.

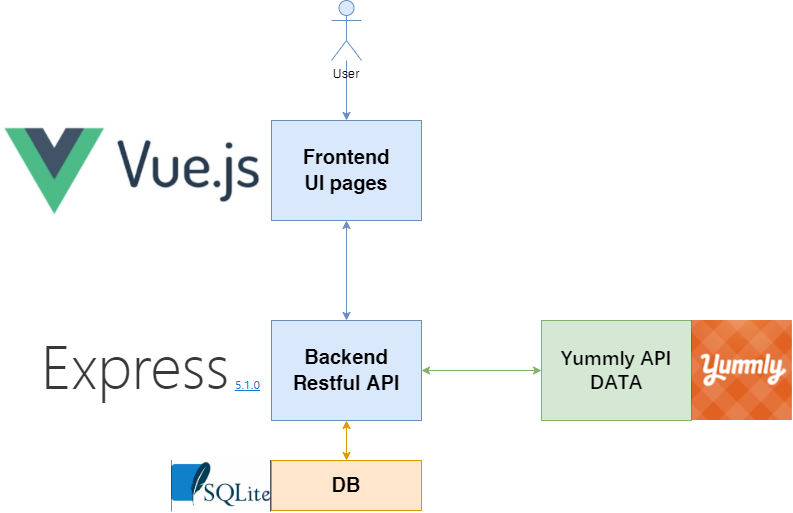
|  |  |  |
| --- | --- | --- |
| User Story | Frontend | Backend |
| Story 1 | Done | Delay |
| Story 2 | Part done, Delay | Delay |
| Story 3 | Done | - |
| Story 4 | Done | - |
| Story 5 | Delay | Delay |
| Story 6 | Delay | Delay |

Challenges Encountered

1. At the beginning, I didn't pay attention to the project requirements and used the wrong database and backend, so the backend was redone.
2. I didn't understand the management of sessions and spent a lot of time on it.
3. I encountered cross-domain problems and it took me a long time to solve them. The API must be called using Express in order to succeed.
4. There are too few official examples of the Yummly API. I need to repeatedly search for various possibilities and debug repeatedly. This took an extremely long time.
5. The CSS style for the front end took a lot of time.

Technical Details

Code and Architecture - New



* The entire project is separated from the front and back ends.
* The front end adopts the Vue framework to complete all page interactions, including Story 1 to 6.
* Users only interact with the front end.
* The back end adopts the express framework and interacts with the third-party API Yummly to obtain recipe data.
* The back-end API provides API services to the front-end, and the API interface is designed in the Restful API style.
* The database adopts SQLite. Since the amount of data involved in this project is small, using this database is sufficient.

API Integration – No Change

1.External API Selection

We choose Yummly API.

2. Purpose of the API

It was chosen firstly because it is free and secondly because the amount of data it provides is very large. Yummly has more data and higher quality than other free recipes.

3. Type of Data to Retrieve

The Yummly API is used to get data about recipes, but only the main fields are listed here.

Recipe: *Recipe name, Recipe Description, Classification of cuisines, Cooking time, Difficulty level*

Ingredients: *The ingredients list, Pictures of ingredients, Food substitution information*

Tutorial: *Detailed steps, Step pictures or videos*

Healthy Data: *Calorie, protein, fat, carbohydrate content, Nutrition Label, Score data*

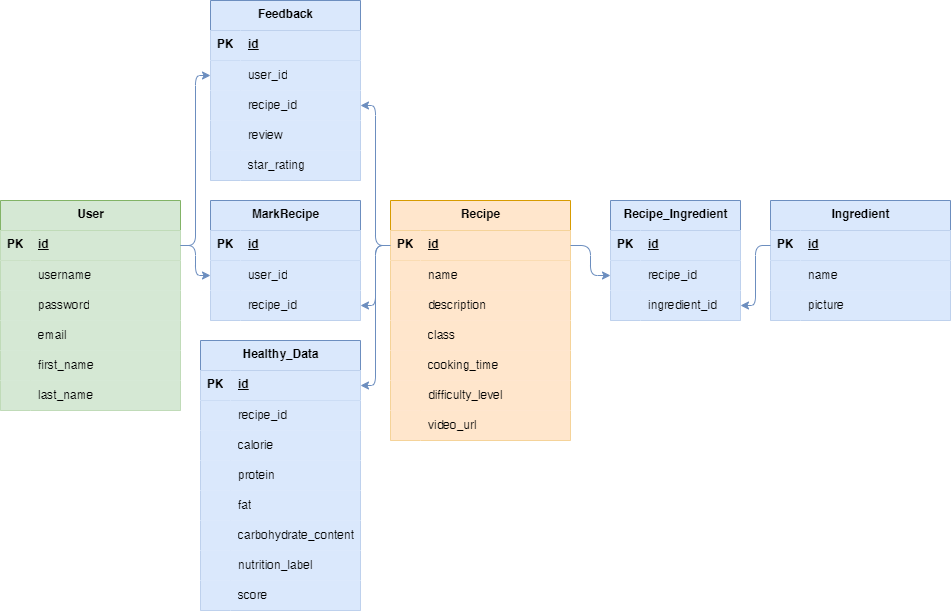
Feedback: *User Reviews, Menu picture, Cooking video link*

4. Combining API Data with My Database

When we use our app to search for a recipe, we get the results from the Yummly API in real time. Due to the large amount of data on this site, the results of live queries are not saved to our database.

If a user follows a recipe or adds it to their meal plan, we save the recipe data to our database.

Database Design – No Change



User Service Description – No Change

**Story 1** - Multicriteria Recipe Search: Users can create filters specifying ingredients available in local markets, budget limits, cooking time, cuisine type, and taste preferences to obtain relevant recipe search results.

**Story 2** - Recipe Collection by Meal Plan: Users can create meal plans (like weekly or monthly) and save all relevant recipe search results under these plans.

**Story 3** - Recipe List Display: Search results are presented in an organized list, including essential details such as recipe names, cooking times, difficulty levels, and cuisine categories.

**Story 4** - Recipe Sharing: Users can share favorite recipes via social media platforms, email, or messaging apps with friends and fellow students.

**Story 5** - Video Guided Cooking: For complex recipes, users can access video tutorials that guide them step by step through the cooking process.

**Story 6** - User Review and Rating: Users can rate recipes and leave reviews, sharing their experiences and providing feedback for other users.

Team Contributions – No Change

Team roles and responsibilities: Currently there is only me on the team, so my role is project management and development.

Responsible for functional design, back-end development, functional testing, documentation and final presentation.

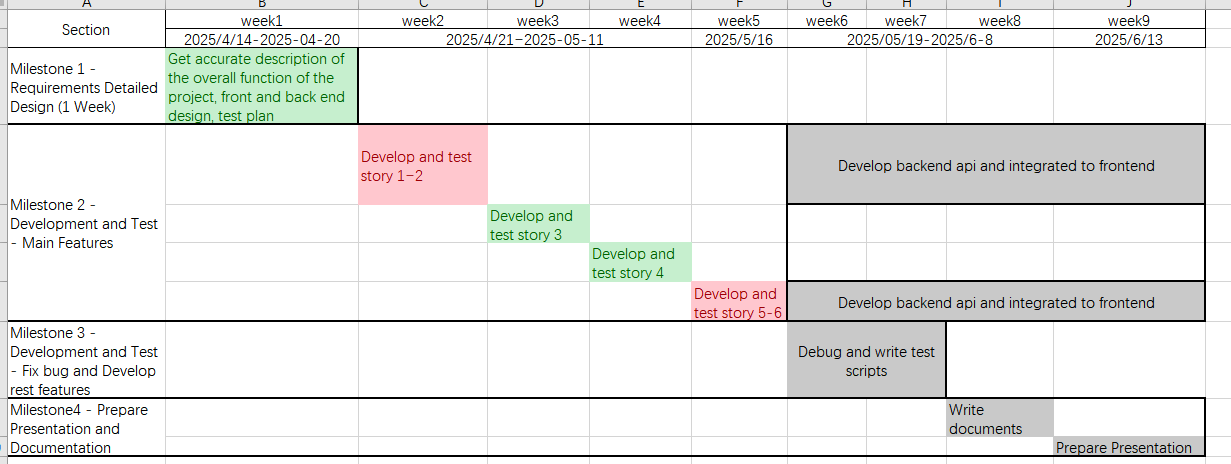
Communication strategies: Divide the project into multiple milestones by function and regularly communicate with my tutor using zoom video.

Tools or platforms you will use for collaboration

Code Management: github

Timeline management: trello, excel

Development Timeline Update - Updated



Story 1,2,3,4: Frontend work is done, **backend task is delay**.

Story 5,6: **Delay** because of previous work is delay.

Risks and Mitigation – New

Risks

1. The call to the Yummly API was not completed as planned and is still ongoing: This function has affected the most core Story1 and 3, thereby causing the delay of the organized project.
2. Because of the unfamiliarity with css, too much time was spent on the front-end work.
3. Perhaps there is not much time for writing automated tests.
4. Perhaps there is not much time for writing development documents.

Mitigations

1. Devote all the time first to completing the debugging of the Yummly API
2. All the data on the front end uses fake data. First, roughly create all the front-end functions. When the back-end API is ready, a simple integration will do.
3. Try to write documents, write tests and do development at the same time as much as possible. Because writing documents and tests are very certain things, unlike API debugging which is full of uncertainties.
4. So finish the things with strong certainty in advance, and then move on to the ones with strong uncertainty.

Feedback and Questions - New

At present, my biggest difficulty lies in the feeling that there is not enough time.

And the solution to my problem has already been stated in the previous content, so instructor is not needed for the time being.