Adolfo Lopez Villanueva

Topic: Recipe Organizer Website

GitHub Repository: https://github.com/LatestStream/CS440_Project1

Short Demo Video: https://youtu.be/LVF__Hmznh0

I. Application Description

The Recipe Organizer Website is designed to help users have a safe place to save all their favorite recipes. The user can create an account, or login if an account is already made, and start adding the recipes that they desire to save digitally and into the account. Of course, now, the user can only just add recipes. Hopefully later in development, the user can remove recipes, edit recipes, and share these recipes with other users.

Some more features that are possible to be implemented in the future are adding images/videos to each recipe to help illustrate certain steps, import recipes from online sources, and share these recipes to other platforms. With these features, the usefulness of this website will hopefully increase.

II. Technologies Used

For this project we chose a lightweight full-stack JavaScript approach that allows us to build a responsive web application while keeping development simple and organized. The backend of the system is built using Node.js with the Express library, which provides an efficient way to create routes, handle HTTP requests, and manage communication between the client and server. Express was used since it was minimal, easy to configure, and well-suited for small to medium web applications that do not require heavy frameworks.

HTML, CSS, and JavaScript were used to built the frontend of the application, creating a clean and interactive user interface where users can input recipe information and view their saved recipes. Using simple JavaScript allows the architecture flexible, for later modifications

As for our database system for this application, SQLite was originally planned due to it being lightweight, requires no separate server installation, and stores data locally in a single file. However, due to time constraints and my current knowledge of implementation of SQLite and

configuration, it was unable to be successfully implement SQLite into the system at this time. Recipe data is currently handled temporarily without a fully connected database.

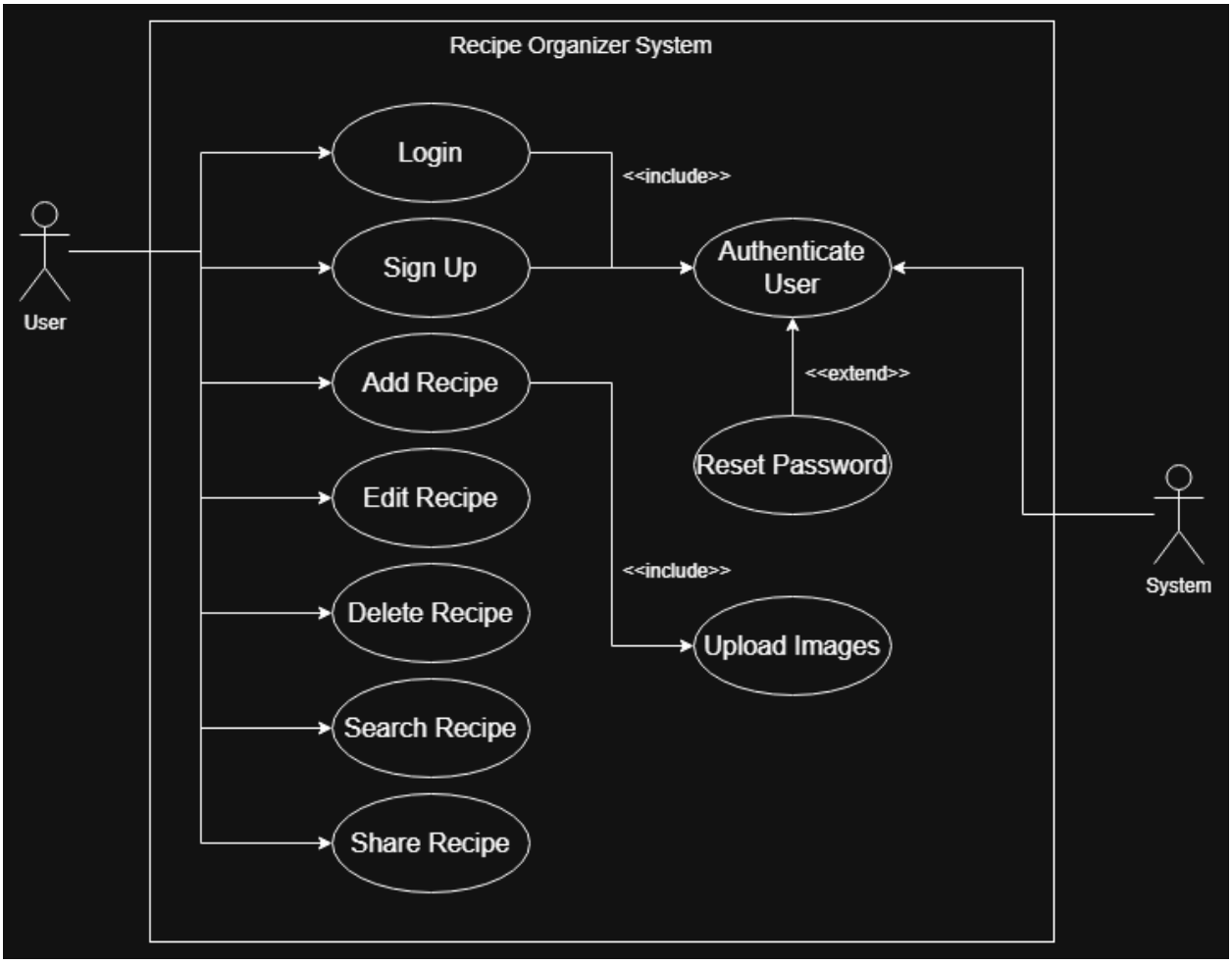
III. Architecture

There was no consideration for architecture or plan of structure. There is some structure however, using the client folder to store all of the code that dealt with the UI of the website (i.e. pages, scripts in Java and style of the website in CSS), using the server folder for any configuration of the database and node.js, and using the doc folder to hold any documentation of the project.

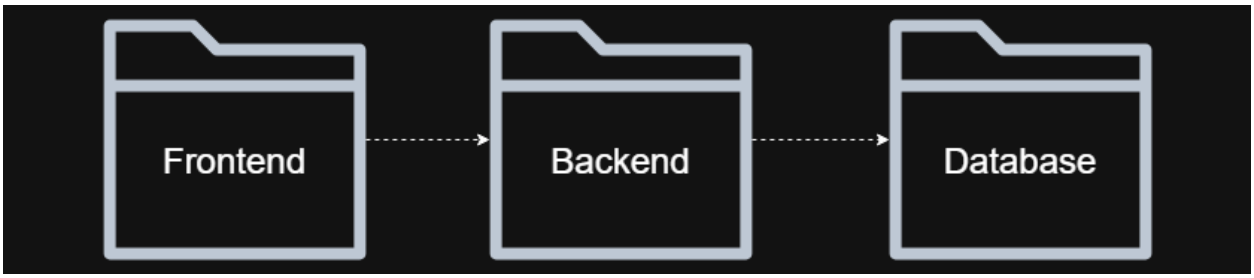
To put it into an architecture, this system follows a client-server architecture, which separates the frontend interface from the backend logic and database making into a three tier system. The frontend layer focuses on the user interface, displaying each page such as the login, signup and dashboard. The backend layer uses Express and is responsible for handling routes such as login authentication, account creation, and recipe submission. The database layer will consist of an SQLite database file soon.

IV. Diagrams

Use Case Diagram



Package Diagram



Activity Diagram