ENSE 375 - Software Testing & Validation

Recipe Management System

Ramsha Naeem (200507574)
Kristina Langgard (200323529)
Pratik Gadhiya (200518183)

Table of Contents

1 Introduction

This project focuses on designing and developing a Recipe Management System, a Java-based desktop application that allows users to create, manage, and search their personal recipes.

We aim to simplify the process of storing and accessing culinary information with mindfulness of a user-friendly experience. In traditional alternatives, there is a lack of organization and limited accessible tools for individuals who enjoy cooking and want to manage their recipes. Existing solutions are often either overly complicated or too limited in functionality. We wish to modernize the experience to be convenient and approachable.

2 Design Problem

2.1 Problem Definition

2.2 Design Requirements

From our vision and outlined design problem, we are putting an emphasis on the database interactivity and readability of our program, focusing on its core functions and how users will interact with them.

2.2.1 Functions

These core functions are as outlined:

1. Create/store a recipe:
    a. Ability to create or upload a recipe in-app.
    b. Ability to save the recipe for later use.
2. Viewing a recipe:
    a. Ability to retrieve a saved recipe when needed.
    b. Ability to retrieve said recipe on different devices, such as a desktop computer, phone, or tablet.
    c. Ability to view recipes while offline.
3. Organizing recipes:
    a. Ability to apply custom tags/filters to recipes.
    b. Ability to sort by custom tags.
    c. Ability to search from saved recipes.
    d. Ability to favourite recipes.
    e. Ability to remove recipes that are no longer needed.
4. Sharing recipes:
    a. Ability to back-up a collection of recipes.
    b. Ability to share a collection of recipes with another user.

2.2.2 Objectives

To accomplish our goals with this application, we have the following objectives in place:

1. User-friendly:
    - the system should offer a clean and intuitive interface.
2. Efficient:
    - the system should respond quickly and minimize unnecessary steps.
3. Reliable:
    - the system should handle data accurately and predictably.
4. Maintainable:
    - the system should follow clean design principles with modular components to support future development.
5. Secure:
    - the system should validate user input to prevent invalid or malicious entries and ensure data integrity.

2.2.3 Constraints

Offline capability:

- An online database is not necessary to store and retrieve data, but rather a local storage system that mimics the concepts for retrieval and management in large collections.
- Sharing recipes and backing up a collection of recipes must be done in a way that can be done through file sharing, such as use of a JSON file to store and read information between independent instances of the application. This must include an import/export feature to handle these files for users so they may accomplish this seamlessly.

Security:

- No information needs to be passed online, and removes the need for account verifications.
- File and input verification are important to the integrity and function of the application in all aspects, so we must ensure all inputs to the application are appropriate and malicious behaviour is avoided. This should be achieved by preventing the use of restricted characters and file types.

File types and inputs:

- Files may not necessarily be text-based, and may also include images.
- Tags and filters can and should be able to be used based on text criteria, but must also be applied should the media a user intends to use includes images.

Tags and filters:

- They may not be based on text-skimming technology due to the variance in accepted file types.
- User may edit or change them independently.
- Favoriting recipes as a unique tag and sort option.

Removing recipes and data that are no longer wanted:

- Properly disposing of related data to remove unnecessary bloat and preserve privacy.

Economically:

- We do not wish to use any external services or paid APIs to accomplish the functions of our application.
- It is made to be free and open-source, to the benefit of our users.

Ethically:

- No user account or personal data should be collected, as part of the mindfulness of privacy and security, and all user inputs or outputs shall be handled equally without bias.

3.0 Solutions

3.1 Solution 1 (Java)

3.2 Solution 2 (PHP/MySQL)

Our next consideration, since we already anticipated something web-based, was to use PHP and mySQL. Since we intend to store and retrieve recipes, this seemed like a natural choice, encouraging dynamic interaction. However, it lacked two key interests in our design; the ability to be standalone, and the ability to be easy to test.

We want our Recipe Management System to be able to be used on multiple devices, and be able to upload and download recipes of the user's own collection, so it may be used offline, and that is not feasible with something that exists solely on the web. Additionally, it would require the use of a domain, which, even if we hosted one ourselves, would have a cost associated with it. Lastly, there were also security risks to consider with it being online all the time. While the data we may be storing is unlikely to be sensitive, we should still be considering our users' privacy as a priority.

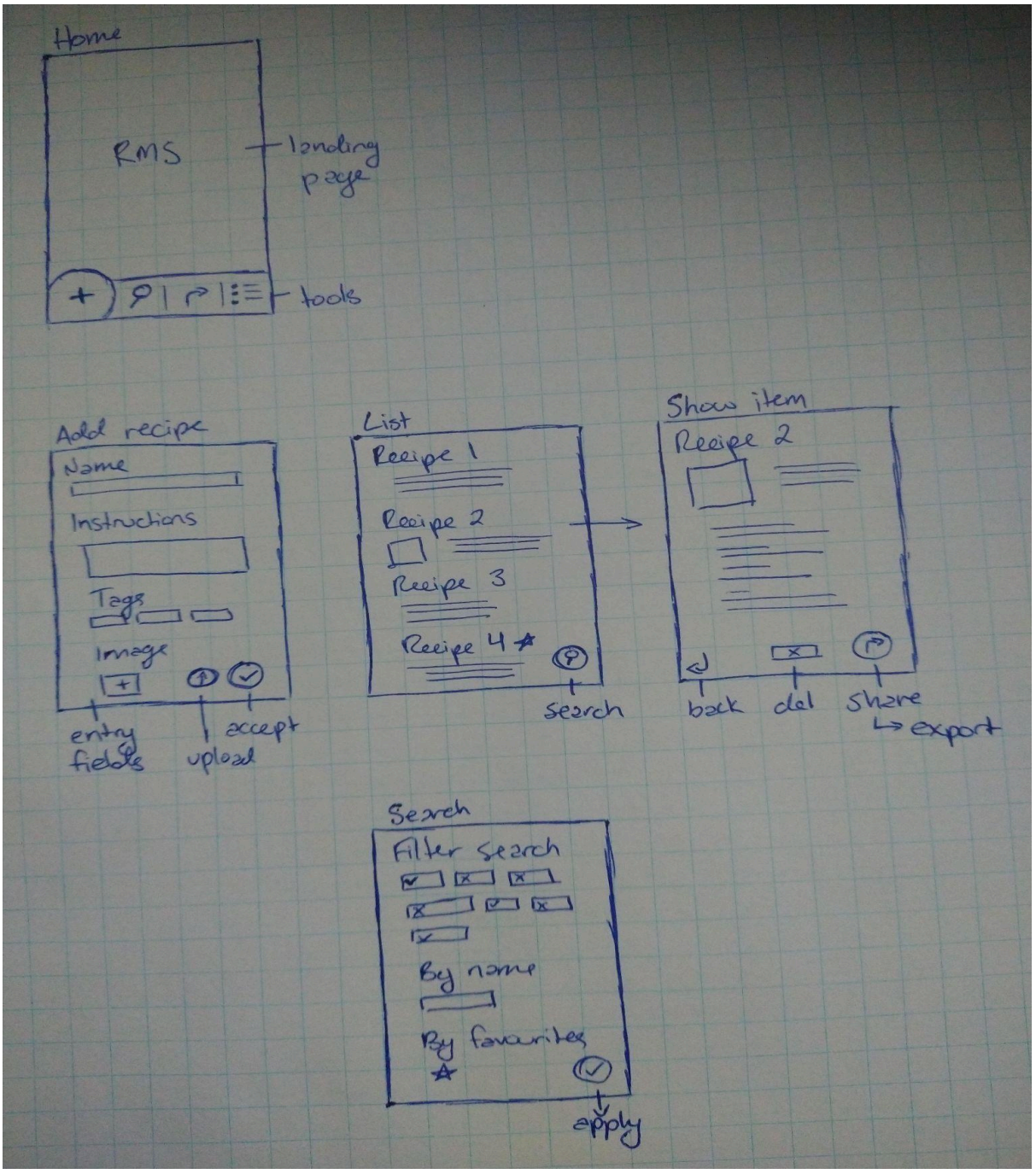
Finally, the ability to test with PHP is quite vague. Error messaging is lackluster, and the operation of commands between a database with mySQL would not prevent us from making mistakes. We consider this to be an important factor as well with this solution, as we do want to emphasize correctness and reliability in our project.

3.3 Final Solution

3.3.1 Components

3.3.2 Features

As defined previously, the key features of our app will revolve around the ability to create/save a recipe, view a saved recipe, organize recipes by custom tags, and share a database with another user.



The user should find a landing page, from which they may select any of these core features to interact with the application. Viewing the list should show all stored recipes. There should be options to filter or search through these recipes, or to set these filters from the beginning, should that be more convenient.

Upon viewing a recipe, there should be an option to export the single recipe, or the entire database, so a user may share their recipes with other users. Similarly, when adding a recipe, a user should be able to upload them manually, or select an export file they were given to upload the information and begin using the new recipe(s) right away. A user should also be able to delete a recipe if they see fit, or be able to edit, add pictures, or change instructions. This information should all be saved and kept in the local database structure.

Some of these features should feel redundant, so they maintain consistency, and that is by design. Keeping the format simple is considered a feature here, as overcomplicated visions of these types of applications exist in plenty already.

3.3.3 Environmental, Societal, Safety, and Economic Considerations

3.3.4 Limitations