Software Requirements Specification

for

Recipe Search Application

Version 1.0 approved

Prepared by Osamah Mohammad

Ahmed Abdelrahman

Ahmed Al-Houri

Jamshid Artykov

Eastern Mediterranean University

11/12/2022

Table of Contents

Table of Contents ii

Revision History ii

1. Introduction 1

1.1. Purpose 1

1.1. Document Conventions 1

1.2. Intended Audience and Reading Suggestions 1

1.3. Product Scope 1

1.4. References 1

2. Overall Description 2

2.1. Product Perspective 2

2.2. Product Functions 2

2.3. User Classes and Characteristics 2

2.4. Operating Environment 3

2.5. Design and Implementation Constraints 3

2.6. User Documentation 3

2.7. Assumptions and Dependencies 3

3. External Interface Requirements 3

3.1. User Interfaces 3

3.2. Hardware Interfaces 5

3.3. Software Interfaces 5

3.4. Communications Interfaces 5

4. System Features 5

4.1. Account Management 5

4.2. Search 5

4.3. View Recipes 6

4.4. Favorites 6

4.5. Custom Recipes 7

5. Nonfunctional Requirements 7

5.1. Performance Requirements 7

5.2. Safety Requirements 8

5.3. Security Requirements 8

5.4. Software Quality Attributes 9

5.4.1 Availability 9

5.4.2 Usability 9

5.4.3 Reliability 9

5.4.4 Responsiveness 9

5.5 Business Rules 10

Appendix A: Glossary 10

Appendix B: Analysis Models 10

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| First Revision | 11/12/2022 | Initial Version of the requirement specification | 1.0 |
|  |  |  |  |

# Introduction

## Purpose

This document provides the specification for the development of a food recipe search website. The specification encompasses all the systems involved in the operation of the website from the user interface to the backend and database systems.

## Document Conventions

In this document, the Times font will be used for all contained text and 12pt will be used for all text except for headings, figure titles, and notes.

## Intended Audience and Reading Suggestions

This specification is intended to be read by all the developers involved in the project along with the project and product managers. Sections 2 and 3 should be enough for project and product managers while developers should read all the rest of the document.

## Product Scope

The aim of this project is to create a recipe search web application that allows users to search for the recipe of any dishes that they would want to cook. Users can visit the website, type the dish they want to prepare into a search box, and the website would show a list of results. The project encompasses the whole system from backend to user interface. However, we will use a third-party library to implement the searching functionality.

## References

<https://www.figma.com/>

<https://www.canva.com/>

<https://app.diagrams.net/>

<https://m3.material.io/>

# Overall Description

## Product Perspective

This SRS describes a standalone product that is not part of a family nor is it a subsystem within a larger product. The recipe search application is a complete website that users can visit to search for recipes.

Here is an overall architecture of the system:

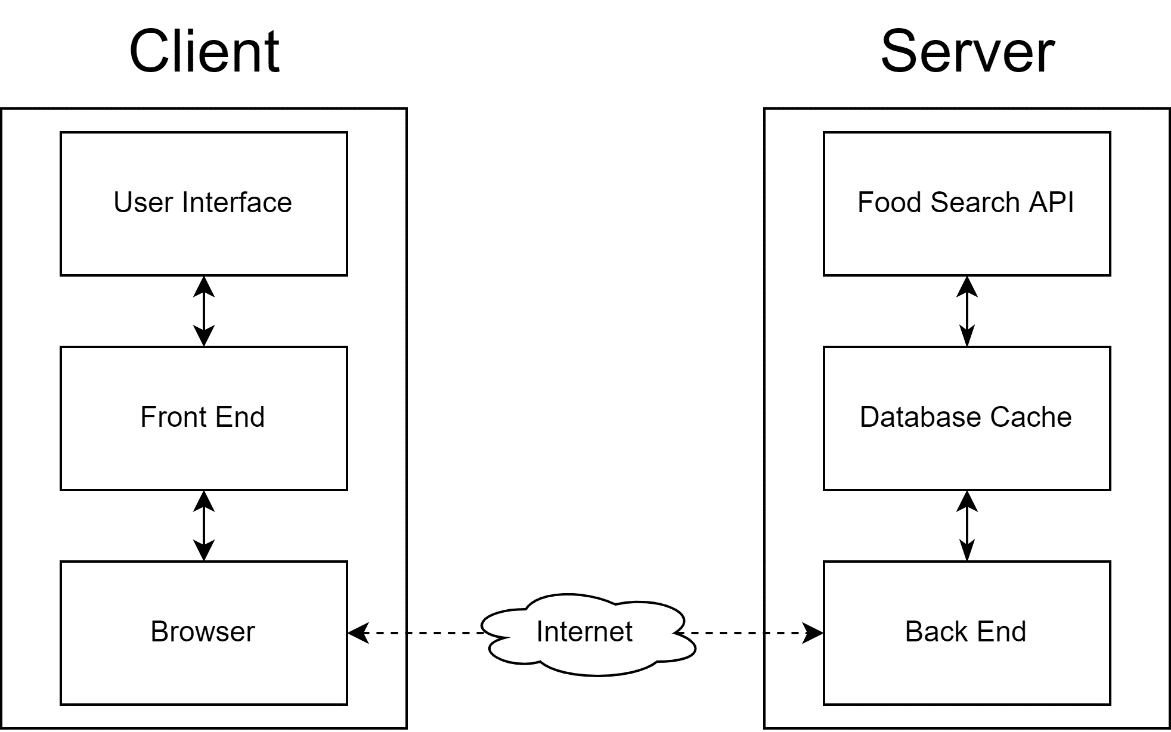


Figure 1: Architecture of the system

## Product Functions

The product has three main functions:

1. Searching for recipes
2. View Ingredients of recipes
3. Mark recipes as favorite

## User Classes and Characteristics

In essence, this website can be visited and used by anybody with basic web browsing knowledge. This website will be most attractive to beginners learning to cook and professionals seeking to learn how to prepare something new. In addition to those, we also expect the website to be used by cooking enthusiasts.

## Operating Environment

Since our software is a website, the front end and user interface will be operating within the visitor's web browser, most probably a Chromium-based browser (Google Chrome, Microsoft Edge, Opera, Brave, etc.). For the backend, we will use the Node.js environment running on a Windows based server since that is what we have the most experience with and have easy access to.

## Design and Implementation Constraints

Since we are developing a web application, then the front end will be limited by whatever functionality is supported by the web browser that the end-user is using. Due to security considerations, we will have to use the HTTPS protocol which means that we will have to purchase a domain name along with an SSL certificate attached to that domain name. For the backend, we will need to make a contract with an ISP in order increase our internet bandwidth and have a globally routable IP address.

## User Documentation

Since the concept behind the website is simple, most users will know how to navigate and use the website provided that the user interface is clear and intuitive. Still, in the front page we will add a link to a page explaining all the features of the website and the step by step on how to search for a recipe and all the options that can be used when doing so.

## Assumptions and Dependencies

One of the external dependencies of our project is on a third-party API called Forkify. We use this API to get the list of recipes from a search term. Our website availability depends on the availability of this API. We also depend on a Express.js framework in Node.js in order to program the website and rely on the interfaces it provides.

# External Interface Requirements

## User Interfaces

the user interface is designed to be easy to use and comfortable to look at, so the UI has dark theme that make looking at the UI more comfortable, also the UI built to be simple with minimal details and at the same time making sure important information are present, and color theme of the web app is compatible with the branding, so everything feels consistent, here is a couple of screenshots of the web pages and the rest will be in APPENDIX:

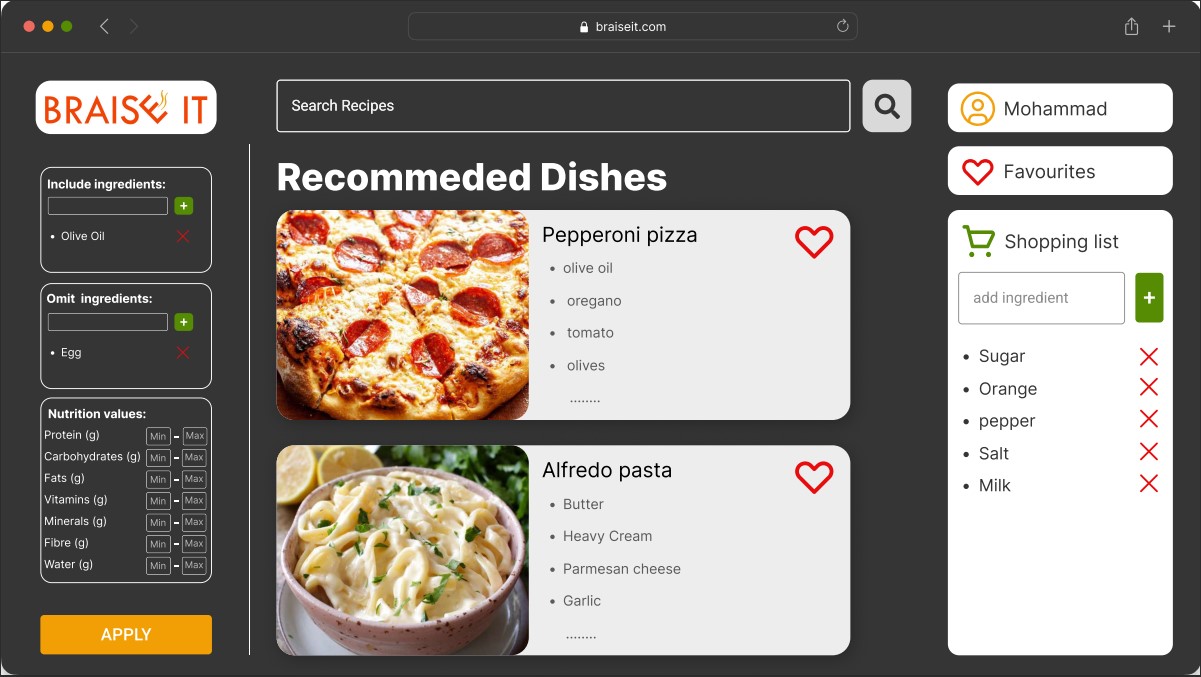


Figure 2 Main page

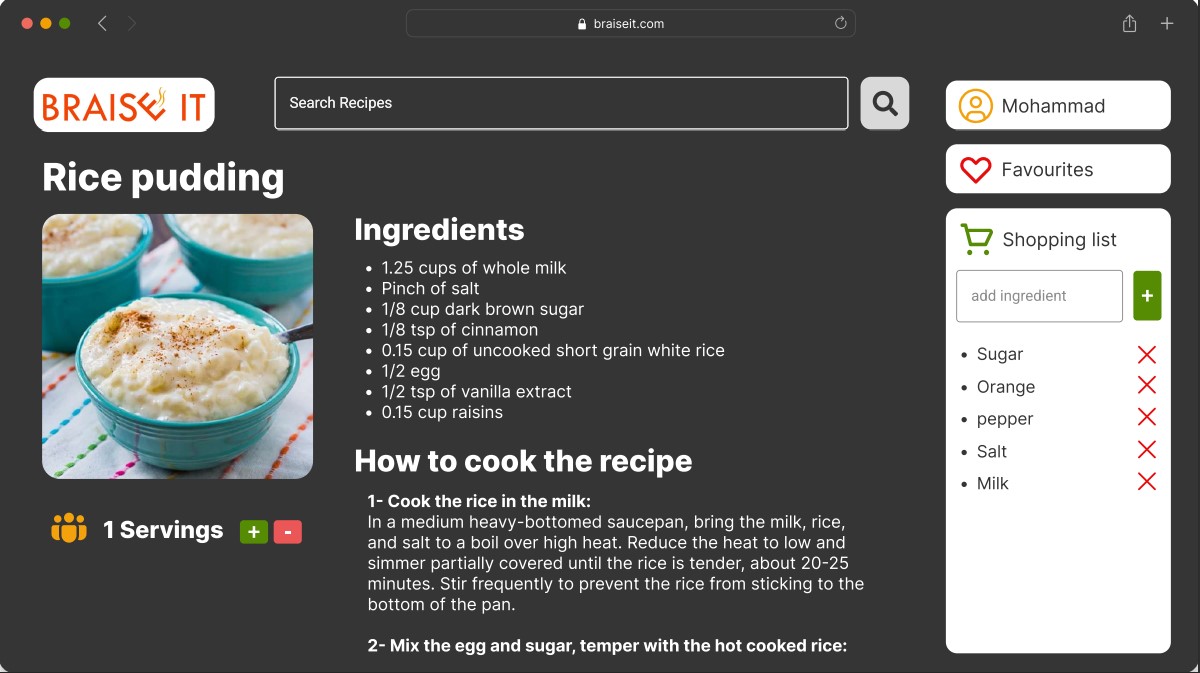


Figure 3 Recipe page

## Hardware Interfaces

The web app will be hosted on AWS servers so users can access it remotely, and users will need device that has web browser to be able to access the app, and the communication protocol between the user device and servers will be HTTPS.

## Software Interfaces

Recipe search application will communicate with Forkify database through an API key provided by owner of the database. The incoming data will be loaded into the system and later on will be used to display to user.

## Communications Interfaces

Users can access the app using browser since it’s a web app, and the communication protocol will be HTTPS which is the most common protocol and provides good security.

# System Features

## Account Management

**4.1.1 Description and Priority**

Users of the system updates their profile information after creating an account. This feature does not hold any priority and won’t affect other features of the system if it is not implemented.

**4.1.2 Stimulus/Response Sequences**

After navigating to their profile page user can update information by updating granted from.

**4.1.3 Functional Requirements**

REQ-1: User shall be able to create account and login.

REQ-2: User shall be able view and edit profile information.

## Search

**4.2.1 Description and Priority**

User searches recipes based on their dish name, ingredients and calculate number of servings. This is the main functionality of the system and holds high priority.

**4.2.2 Stimulus/Response Sequences**

After entering keywords to the search bar, user will be granted with the list of ingredients listed

**4.2.3 Functional Requirements**

REQ-1: User shall be able search recipes based on dish name and ingredients used.

REQ-2: Calculate amount of ingredients based on desired servings.

REQ-3: Filter down the search to list to omit certain ingredients.

## View Recipes

**4.3.1 Description and Priority**

After searching user navigates to desired recipe which shows the information about the recipe, ingredients, and source of the recipe. User can add ingredients to shopping list. This functionality holds high priority.

**4.3.2 Stimulus/Response Sequences**

Users navigates to recipe and performs available features.

**4.3.3 Functional Requirements**

REQ-1: User shall be able to add ingredients to shopping list

REQ-2: Users shall be able to view nutritional values of the ingredient.

Optional REQ-1: Show price of ingredients.

## Favorites

**4.4.1 Description and Priority**

Users can save the liked recipes to view them again without searching for it. User may save multiple recipes. This functionality holds medium level of priority.

**4.4.2 Stimulus/Response Sequences**

User clicks on heart sign to save the recipe.

**4.4.3 Functional Requirements**

REQ-1: User shall be able to mark favorite recipes.

REQ-2: Users shall be able to view saved recipes.

REQ-3: Users shall be able to rate the recipes.

REQ-4: Users shall be able to remove the marked recipes from the favorites list.

## Custom Recipes

**4.5.1 Description and Priority**

Users create their own recipes. The created recipe is only accessible by the creator of the recipe. Priority of this feature is small.

**4.5.2 Stimulus/Response Sequences**

Users clicks on create recipe and fills the form.

**4.5.3 Functional Requirements**

REQ-1: User shall be able to create their own recipes.

REQ-2: Users shall be able to view created recipes.

REQ-3: Users shall be able to create recipe with up to 10 ingredients.

REQ-4: Users shall be able to delete the created recipe.

# Nonfunctional Requirements

## Performance Requirements

The website's loading time shall not exceed 1 second and the website should not require more than 10 mbps bandwidth.

These can be measured in the "networking" tab in the browser's development menu. A screenshot is shown below. Here, in the bottom, the measurement highlighted in red is the amount of downloaded data and the measurement highlighted in cyan is the time it takes for the website to load. The bandwidth used is the data transferred divided by the loading time.

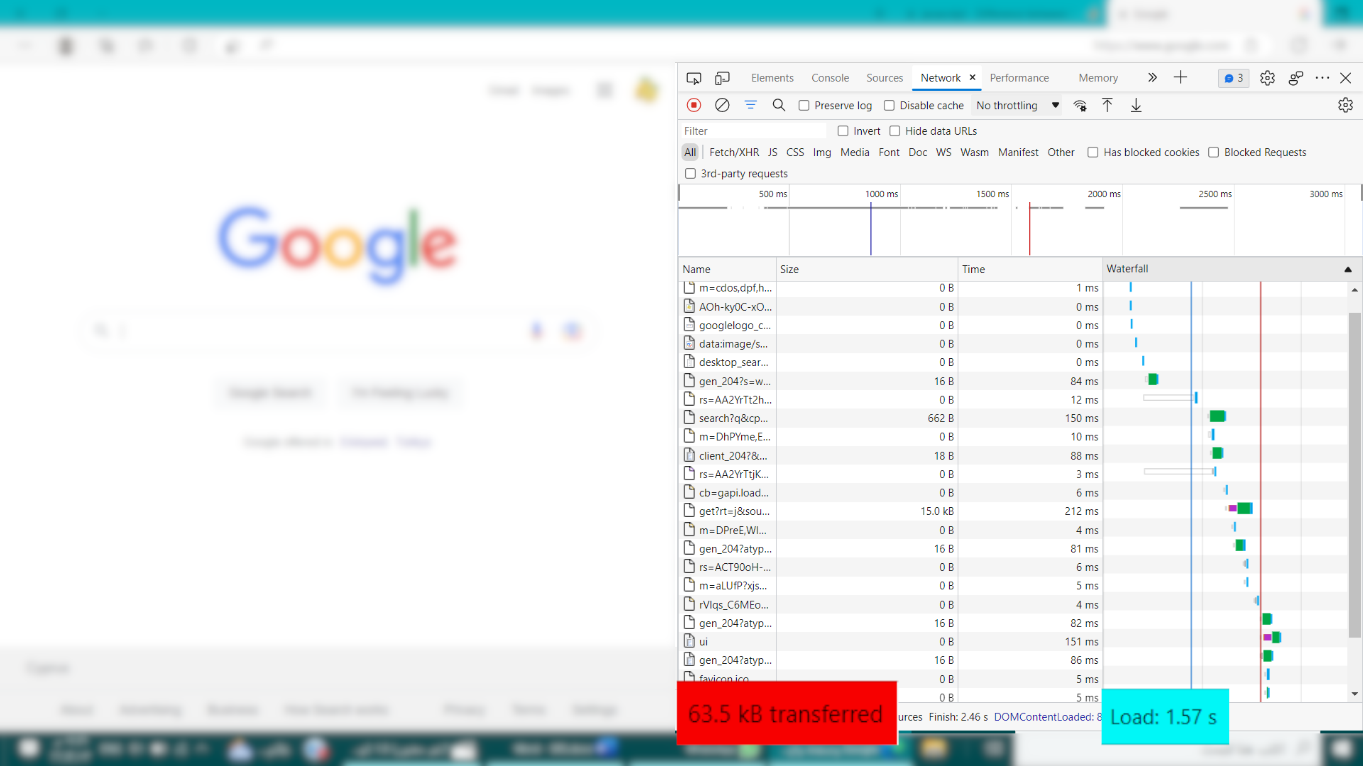


Figure 4: Measuring bandwidth and loading time in the browser.

## Safety Requirements

Because the project is just about the development of a recipe searching website, there are no significant safety concerns with respect to the website itself. However, an inexperienced user could harm themselves when trying to prepare a recipe from the website (pressure cooker, boiling oil, burning oven, spilling hot water, etc.), but there is nothing the website could do to prevent these problems other than displaying a disclaimer/warning to the user when they visit.

## Security Requirements

The system should be able to protect sensitive data, user should be able to access to his/her own personal data only and cannot access any other user’s personal information. Furthermore, all external communications between the system’s data server and clients must be encrypted. Some examples of our security requirements:

* **Password creation:** the system should ask the user to enter a strong password according to the system's policy. For instance, the password should have a minimum number of characters as well as a number and two capital letters.
* **Security question:** the system should ask the user to answer a specific question that the user only knows. Thus, we can maintain the security of the system by checking the identity of the user logged in.
* **Account locking:** to protect and secure the user’s personal information from any possible hackers or theft, the system will shut down the account after a set number of trials of failed login. The user will be able to unlock his account by calling the company to verify the identification of the user and to create a new password for his account.
* **Encryption:** The website will obtain an SSL certificate that will prevent hackers from intercepting communications and raise its level of security.

## Software Quality Attributes

This part will detail the aspects that must be examined and certified before we can declare that our system is of high quality.

* A suitable programming language should be selected for the system to function without difficulties in multiple operating systems (OS).
* To quickly alter undesirable places or incorporate more features, the software should have a modular as well as an understandable coding design.
* The website will be displayed and set up in a way that is both aesthetically pleasing and user-friendly. In addition, feedback will be given and there will be visible indicators like pop-ups and notifications to let users know when there are any modifications.

## 5.4.1 Availability

The system availability will depend on Forkify. But, the system should be available at least 99.999% of the time for anyone who wishes to use the system.

## 5.4.2 Usability

The system should be quite self-explanatory, simple, user-friendly, and easy to use. Users should be able to rapidly pick up on the system.

## 5.4.3 Reliability

Users can access their accounts 99% of the time without failure. The system will have a backup server in case of any failure, the backup server should launch in less than 5 minutes.

## 5.4.4 Responsiveness

System will be visible on a variety size of devices including tablet, mobile, and desktop. System characteristics will be the same on all platforms. However, the system’s features will work in different ways to accommodate the various viewport and navigation options.

## 5.4.5 Compatibility

* The system must work well on all browsers.
* The system must support a variety of hardware, software, and network configurations.
* The system should support all types of hardware versions and operating systems.

## 5.5 Business Rules

* Users should be able to search for one recipe at a time.
* Users must create the password according to the system’s policy.
* Users must enter a valid username and password.
* Users should be able to mark favorite dishes only when they are logged in.
* There should be no limitation on the number of dishes users can mark as their favorite.
* Users must enter all mandatory fields.
* The system must maintain standard formats for email and date of birth.

Appendix A: Glossary

**Recipe/Dish:**Name of the food being prepared

**Ingredients:**Individual components of a recipe/dish

**UI:** User Interface

**API:** Application Programming Interface

**OS:** operating system

**HTTPS:** Hypertext Transfer Protocol Secure

**AWS:** Amazon Web Services

Appendix B: Analysis Models

Diagram

Description automatically generated

Figure 5: Dataflow design of the system

This is the process map diagram that illustrates the steps that we take throughout the development of the website. It mostly follows traditional SDLC but with more stakeholder involvement in the requirements and design phase.

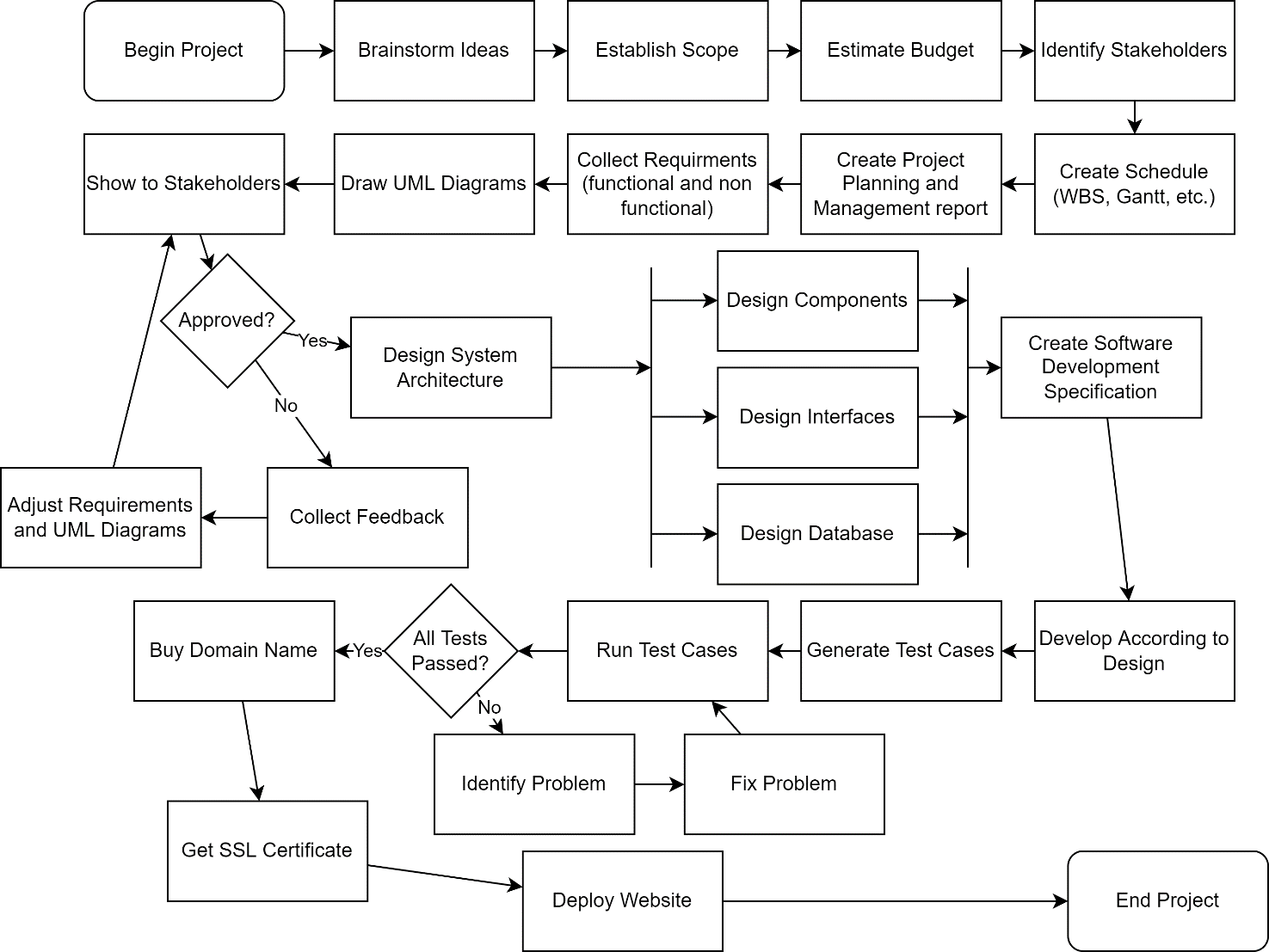


Figure 6: Development Process Map

The following is a sequence diagram that illustrates the flow of information between the User interface, the backend and the third-party Forkify API.

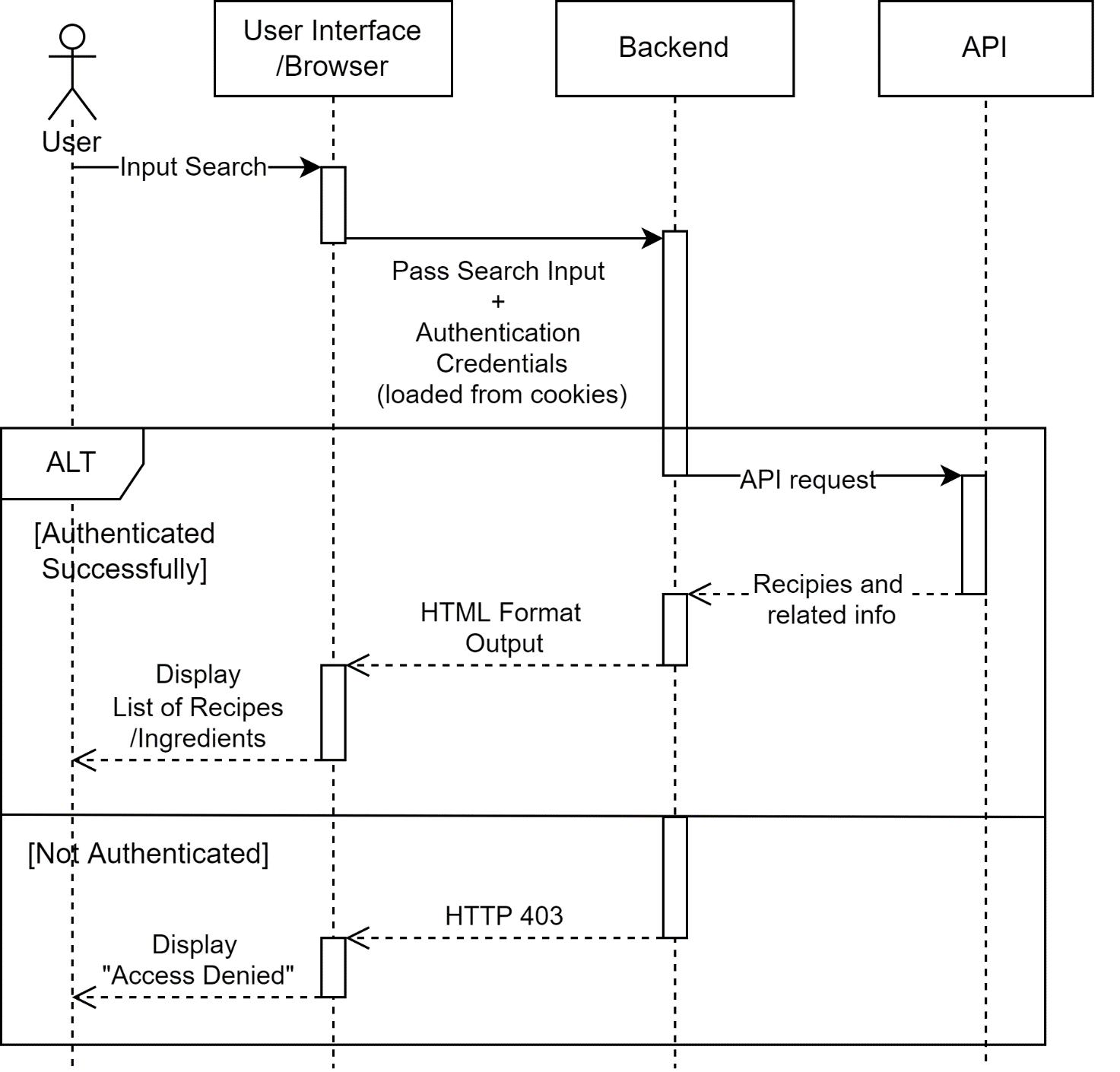


Figure 7: Sequence Diagram

The following is an activity diagrams that shows the procedures followed by the system when the use logs in and performs a search operation

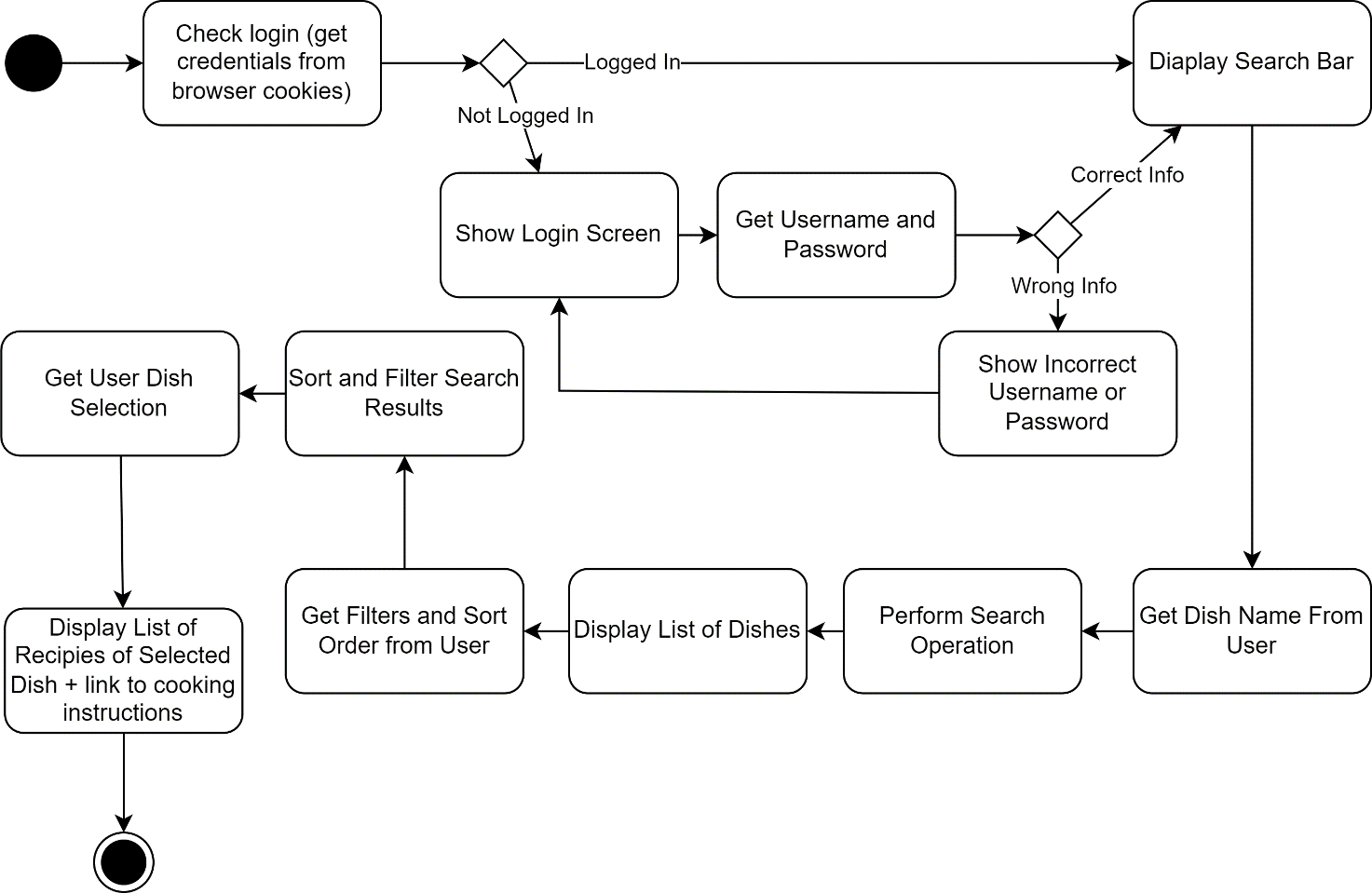


Figure 8: Activity Diagram

The following is a QFD, also known as the house of quality. This diagram shows the customer requirements, the technical requirements and the relationship between them, and based on that the technical requirements are prioritized and ranked. In addition, above each technical requirement, there is a triangle pointing up or down showing if that requirement should be minimized or maximized. On the top, the correlation between the technical requirements is shown. Additionally, on the right, there is competitor research showing how we are performing in comparison to our competitors.

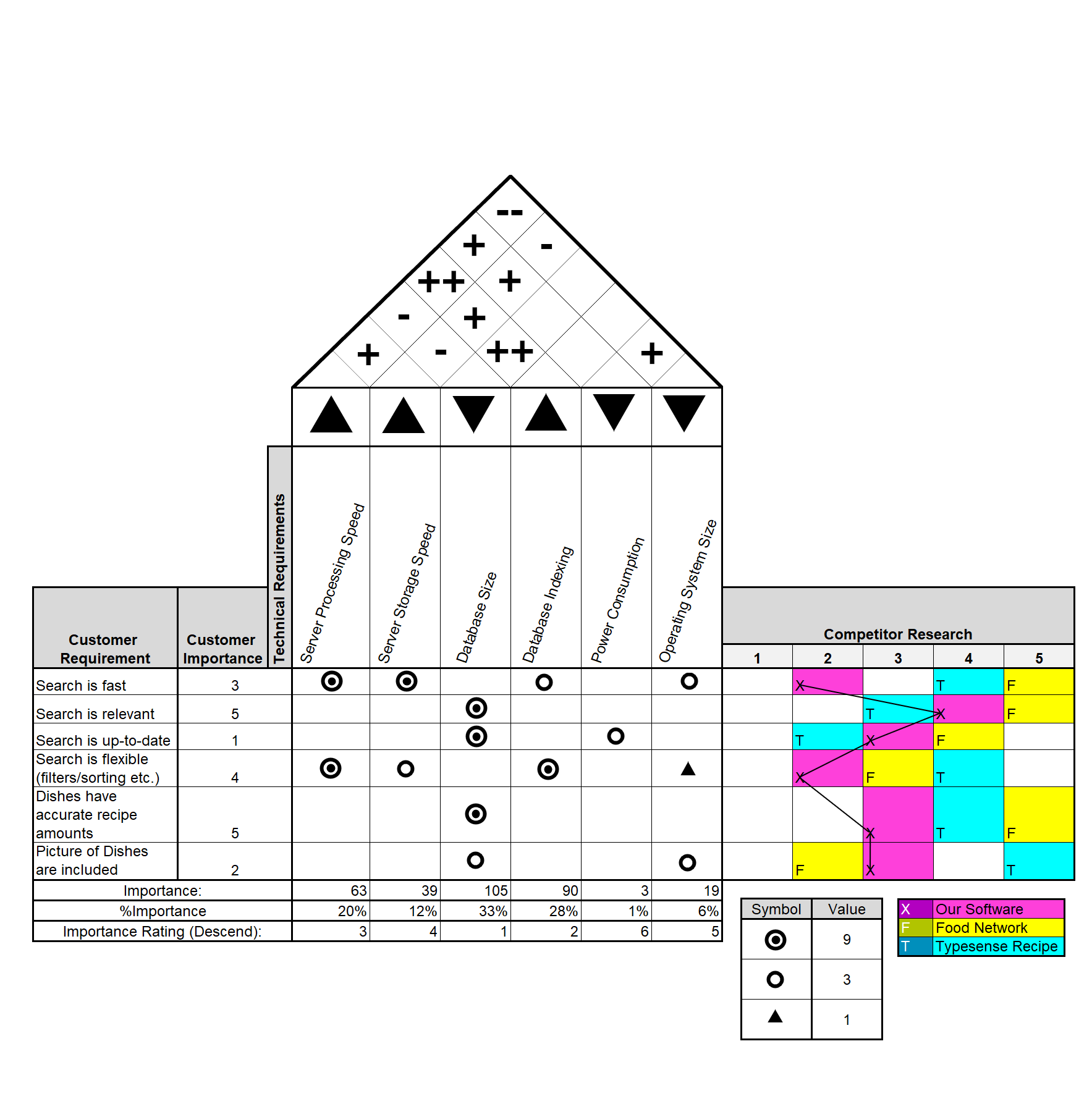


Figure 9: Quality Function Deployment Diagram (House of Quality)

The following is a context diagram of the system. It illustrates how the system connects user to Forkify API.

Diagram, schematic

Description automatically generated

Figure 10: Context Diagram

The following is the **Use case** diagram, it demonstrates all possible users’ interaction with the system

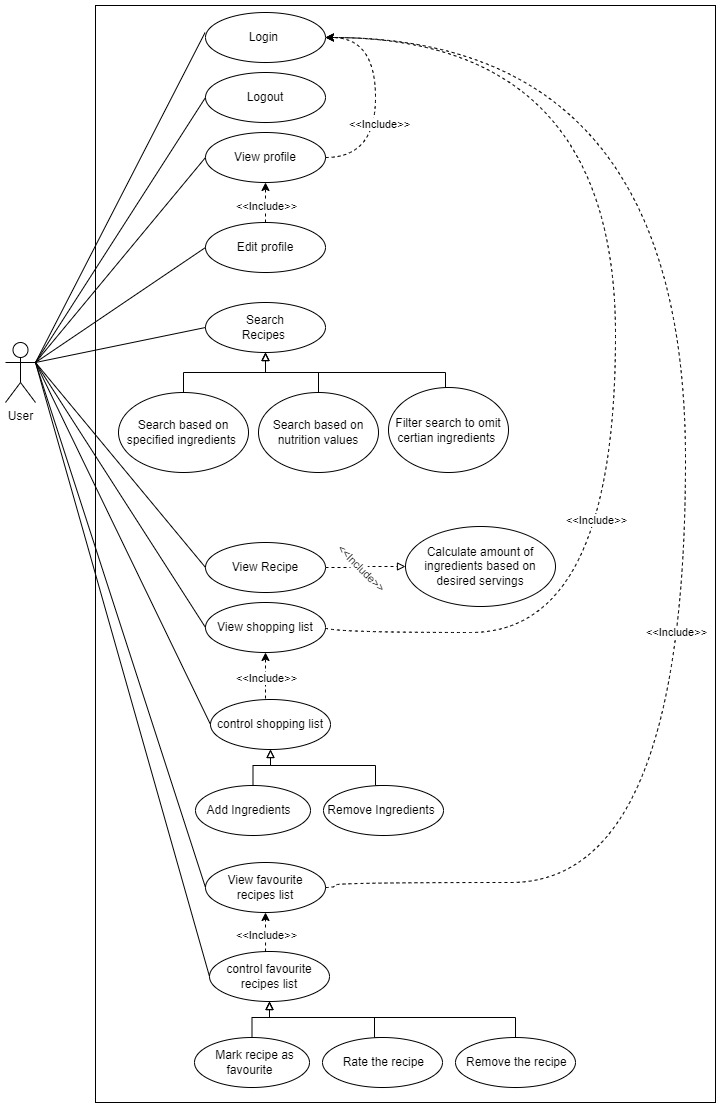


Figure 11 Use case diagram

**SWOT analysis** stands for Strength, Weakness, Opportunity, and Threats, and it’s a technique used to analyze the situation of the organization

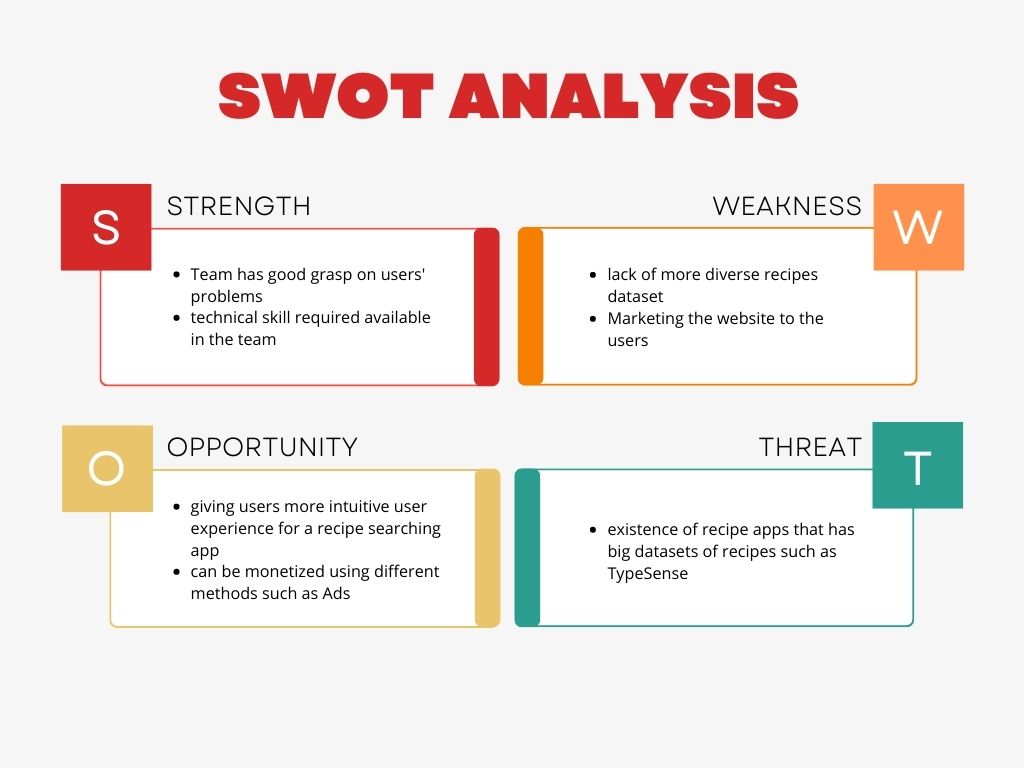


Figure 12 SWOT Analysis

**User Interface and Branding:**

* This page demonstrates the branding and can be used as loading page:



Figure 13 Branding and loading page

* This is the authentication of the website:

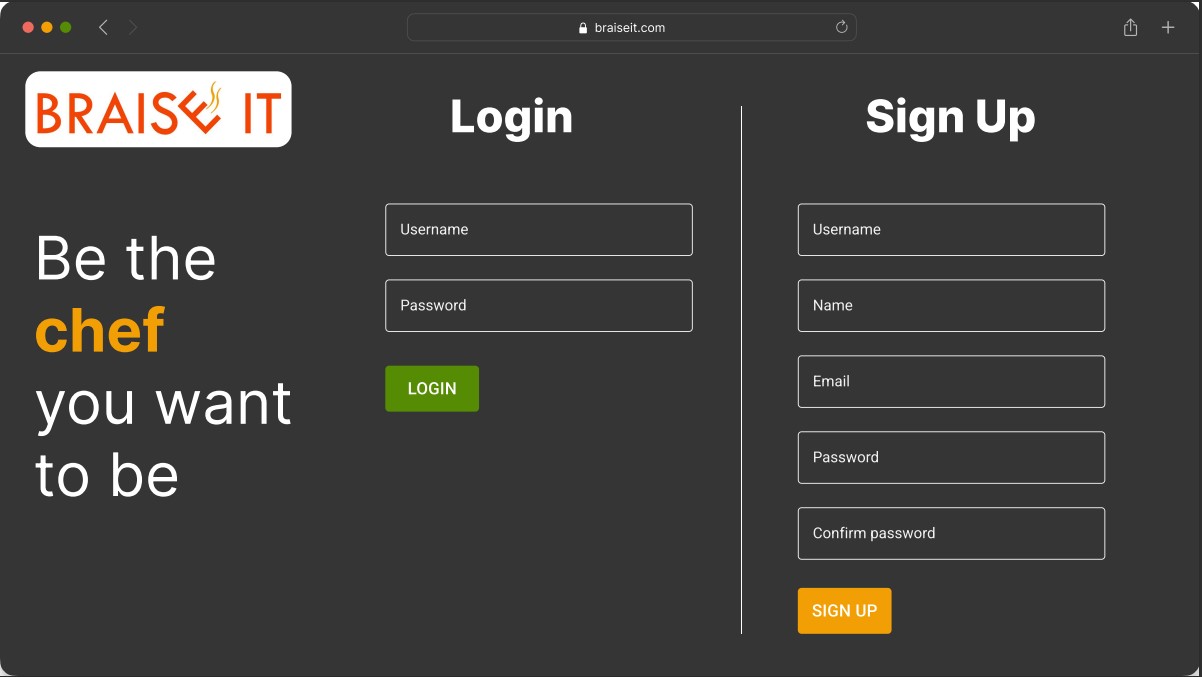


Figure 14 Authentication page

* The following is the main page, where it recommends random recipes, we note on the side there’s shopping list where the user can add ingredients that is not available in the kitchen

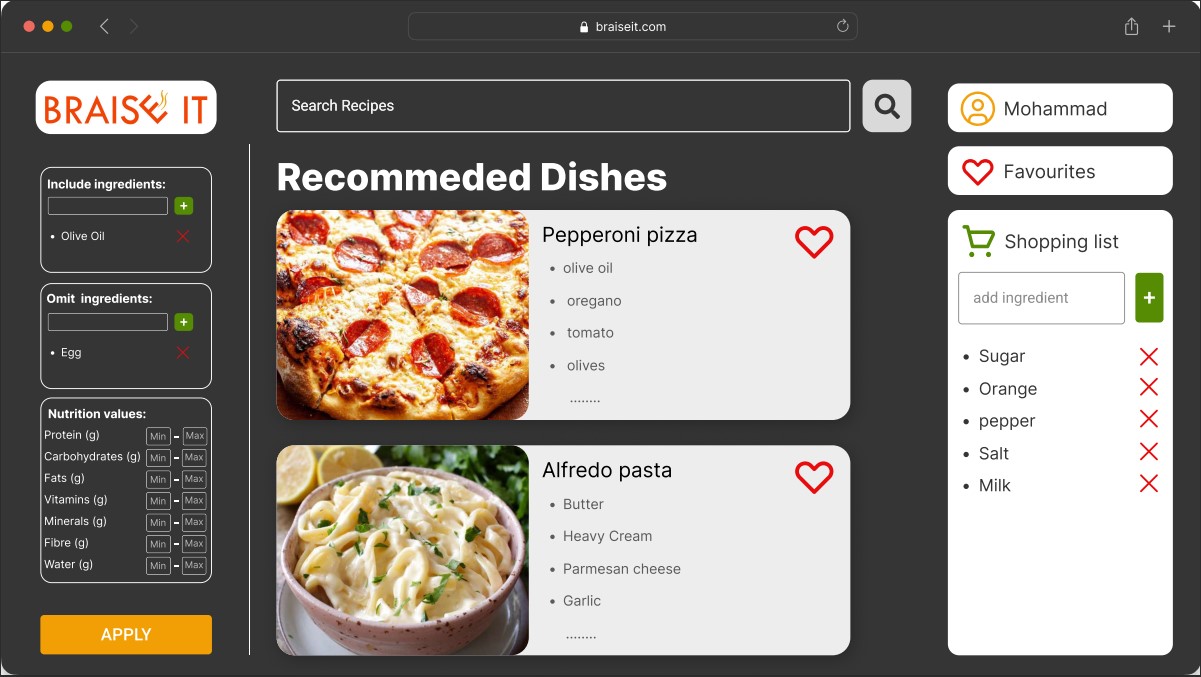


Figure 15 Main page

* This page demonstrates search results of term ‘Rice’

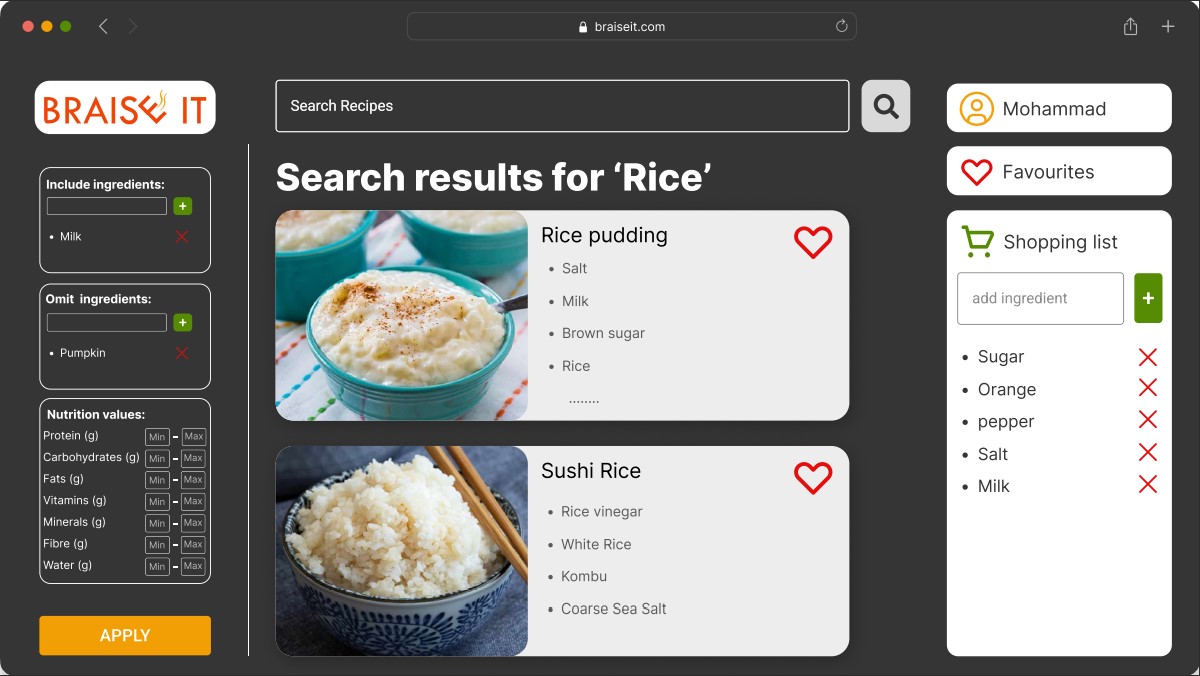


Figure 16 Search page

* This is the page that shows the recipe with its details, where the user can change number of servings so the app will calculate amount of ingredients needed accordingly:

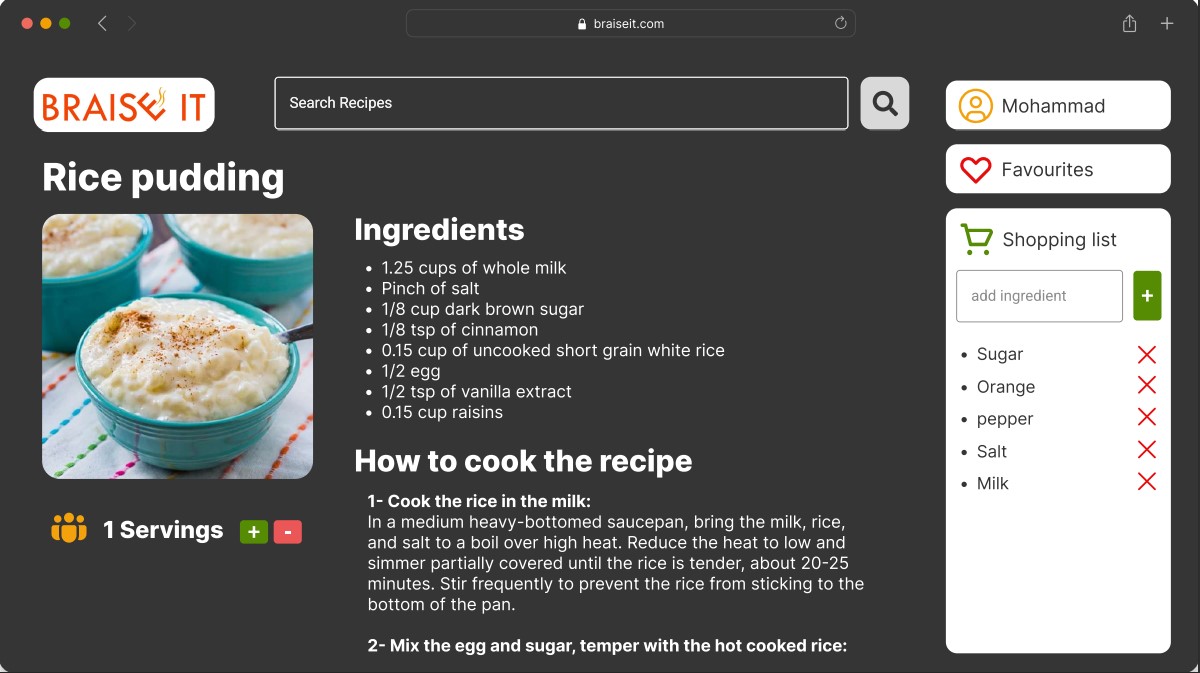


Figure 17 Recipe page

* This page shows the favorite recipes that is marked by the user:

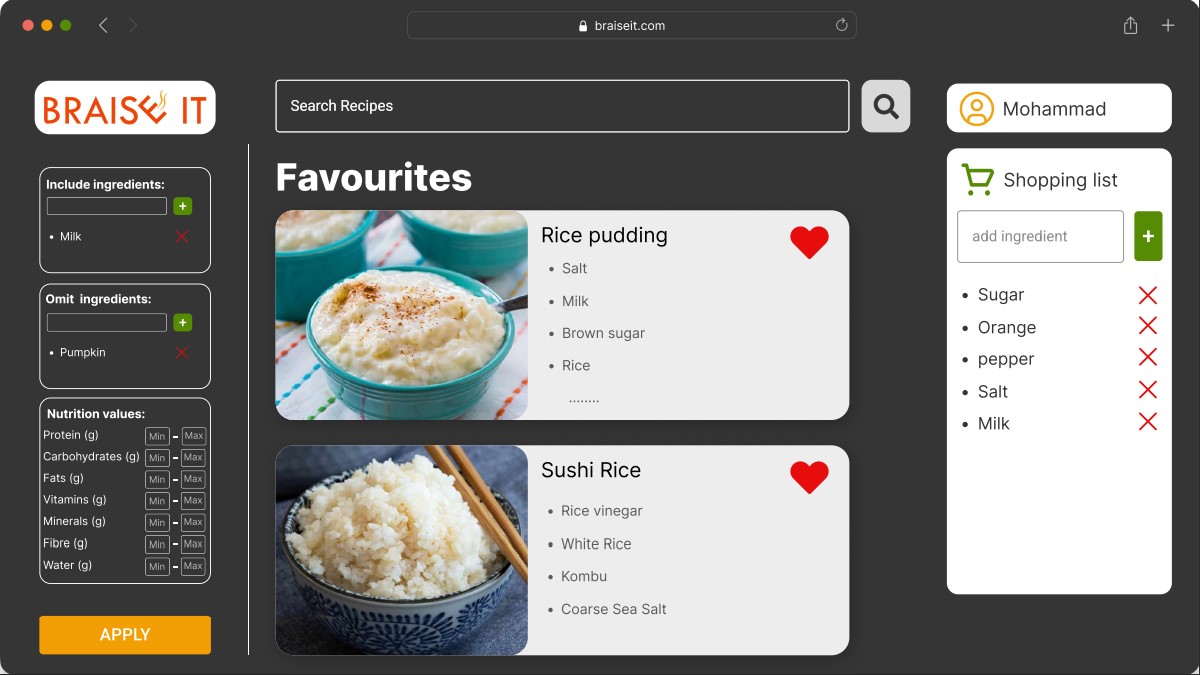
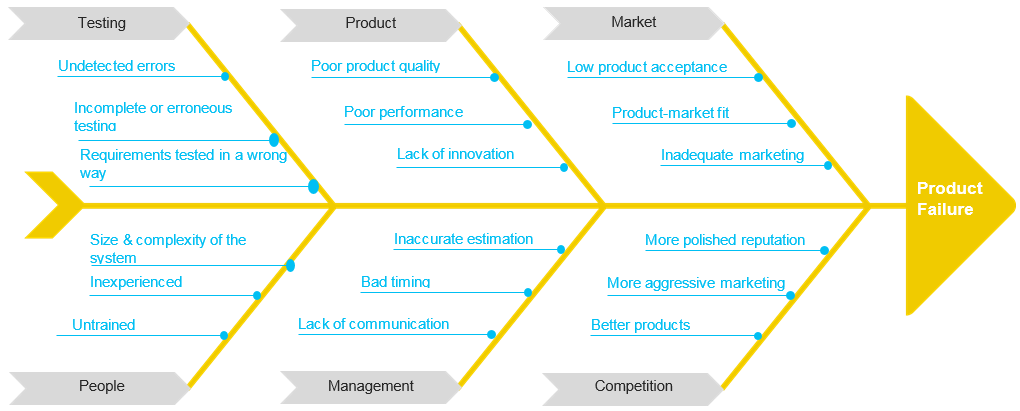
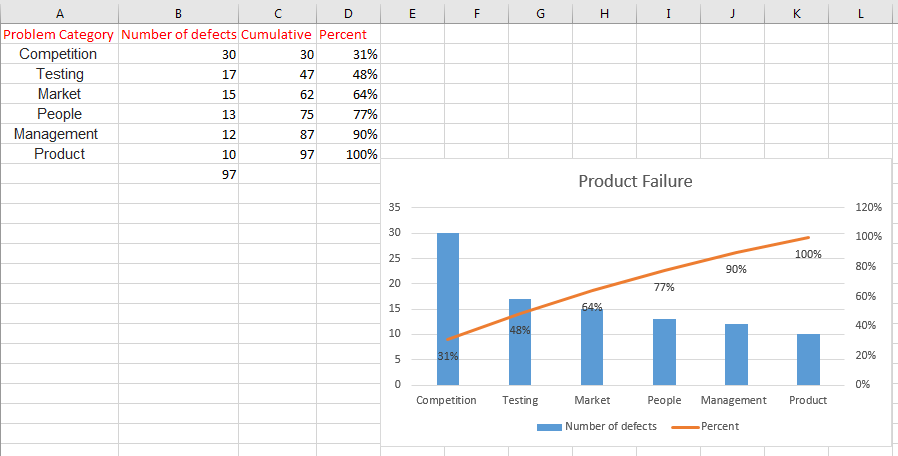


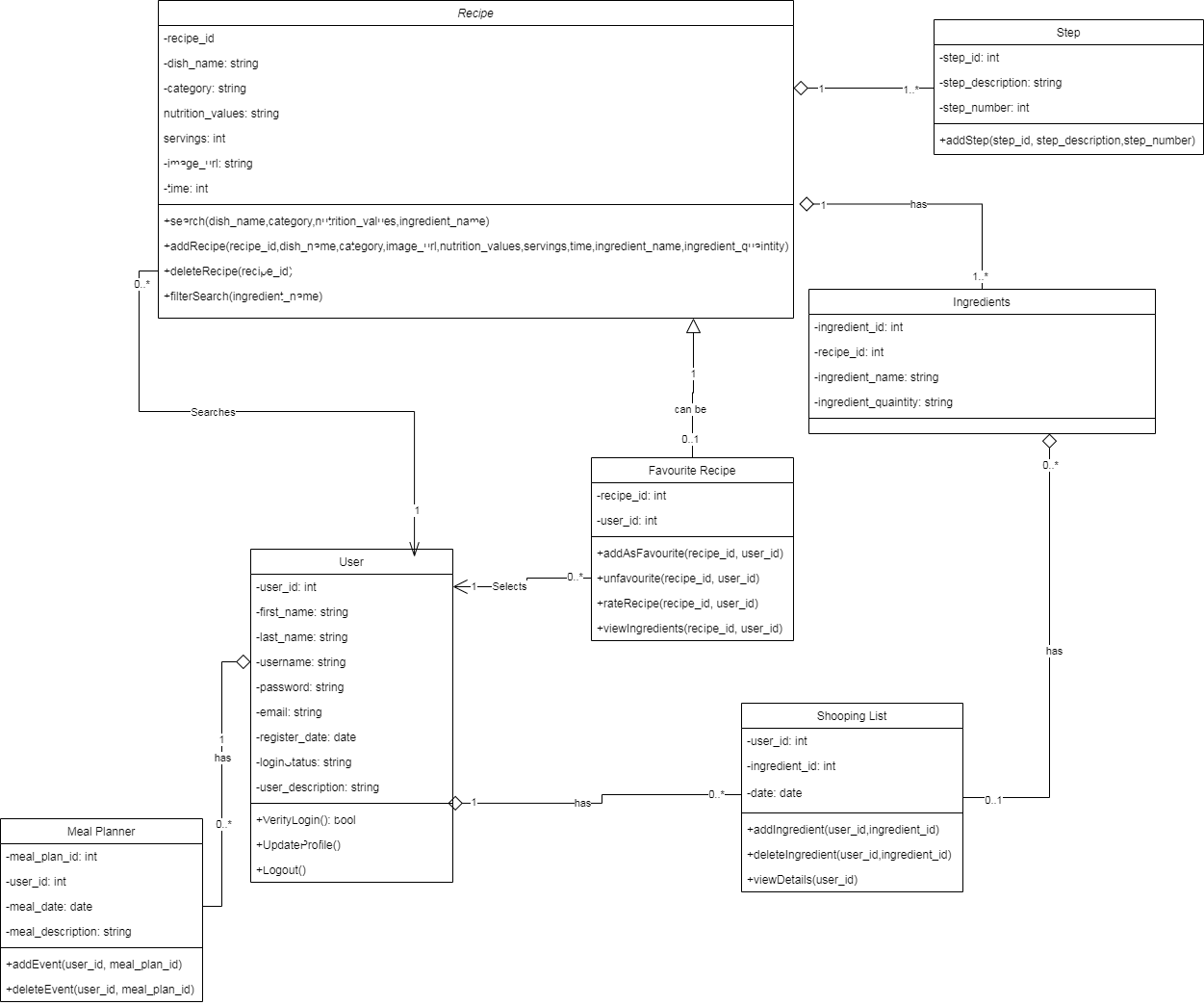
Figure 18 Favorite Recipes page

* The following figure shows the fishbone diagram which illustrates the factors that might affect the system:



* The following figure shows the pareto chart for product failure. As shown, the main occurrences of defects are due to competition, testing, market, and people which forms 77% of defects in total.

- The following figure shows the class diagram of the system.

- The following table shows the quality checklist

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No.** | **Quality metric** | **Unit** | **Requirement** | **Tools/Methods** | **Actual** | **Status** |
| 1 | Loading time | Seconds | < 1.5 | Brower developer window | 2 seconds | Failed |
| 2 | Server capacity | Maximum visitors/second | >= 1000 | Stress testing | 1050 | Passed |
| 3 | SEO | Percent | > 70% | Seositecheckup + Upcity | 80% | Passed |
| 4 | speed | ms | < 400 ms | [Pingdom](https://tools.pingdom.com/) | 300 ms | Passed |
| 5 | Availability | Percent | >= 99.999% | Site24x7 | 97% | Failed |