# Final Project: YourOwnRecipe Application

**Student: Thi Hue Dao**

**Instructor: Pardeep Rathore Oct 15, 2022**

# Introduction

Do you enjoy cooking? Can you cook without recipe note paper? Nowadays, you can search for the right cooking recipe on the Internet, but it may take a lot of time. So the best solution is considering to build your simple recipe application.

# Scope

In the YourOwnRecipe system, the main app screen offers users a list of recipes and a search feature to help them find the recipes they need. Also, the system allows users to add, edit and delete recipes by themselves.

Below are features and technical specifications:

Features:

* + Registration
  + Login
  + List of recipes.
  + Create a recipe.
  + Edit the recipe.
  + Delete the recipe.

Technical Specifications:

* + Express for front end
  + Node for back end
  + MongoDB database
  + GitHub
  + Jira tracker tool

Domain requirements:

* + - * The web app must run on modern browsers and mobile devices

# Functional Requirements

* 1. Registration

Use Case Description:

|  |
| --- |
| Introduction: This use case allows the user to register the system |
| Actor: User |
| Precondition: The system must support sign up through auth0 third party |
| Postcondition: If use case in successful, the user is granted access to the system by using login use case |
| Event Flow:  The user clicks on Login button  The auth0 login screen display then clicks on Sign Up |
| Alternative Flow: |

# Login

Use Case Description:

|  |
| --- |
| Introduction: This use case allows the user to access the system. |
| Actor: User |
| Precondition: the system must support login from auth0 |
| Postcondition: If the use case is successful, the user is granted access to the system. |
| Event Flow:  The user enters his/her email and password on the auth0 login form  Or choose login by using a Google account |
| Alternative Flow: If the system does not validate the user's login id or password due to entering wrong, an error message will be displayed and the use case return to the beginning of the basic flow. |

# Validity Checks:

* + Email and password must valid
  + A Google account must exist

# Add recipe

Use Case Description:

|  |
| --- |
| Introduction: This use case allows the user to add a new recipe to the system. |

|  |
| --- |
| Actor: User |
| Precondition: The user must have their login id and password. |
| Postcondition: If the use case is successful, the new recipe will be added to the system. |
| The user opens the add recipe screen  The user fills out the recipe form  The user clicks on the add recipe button |
| Alternative Flow: If the user does not fill out all information in the recipe form, then an error message is flagged saying " you cannot add the new recipe in the system" and the use case returns to the basic flow. |

# Validity Checks:

Users must log in to the system.

# Delete recipe

Use Case Description:

|  |
| --- |
| Introduction: This use case allows users to delete recipes from the system. |
| Actor: User |
| Precondition: The user must have their login id and password. |
| Postcondition: If the use case is successful, the recipe will be deleted from the system. |
| Event Flow:  The user logs in to the Recipe system  The user deletes the recipe from the recipe list Recipe is deleted from the system. |
| Alternative Flow: If the user does not choose recipe that they want to delete, then an error is flagged saying " You must select recipe" and the use case returns to the basis follow. |

# Validity Checks:

Users must log in to the system.

# Edit the recipe

Use Case Description:

|  |
| --- |
| Introduction: This use case allows the user edits recipes from the system. |
| Actor: User |
| Precondition: The user must have their login id and password. |
| Postcondition: If the use case is successful, the recipe will be updated to the system. |
| Event Flow:  The user login to the system  The user opens the recipe that they want to edit  Update recipe information in the recipe form  The new recipe is added to the system. |
| Alternative Flow: |

# Validity Checks:

Users must log in to the system via auth0 third party.

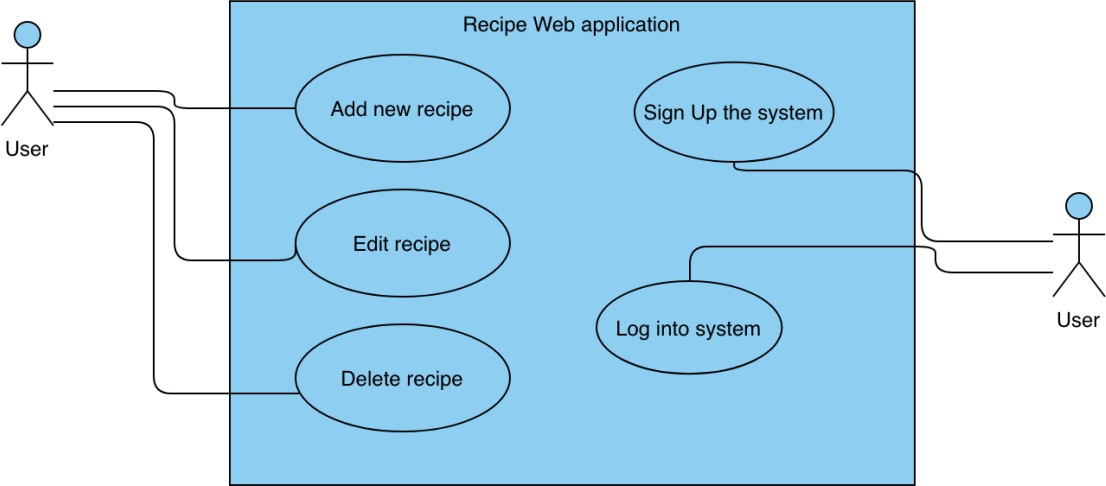
# Non-Functional requirements

* + Performance must be smooth without UI delays and freezes.

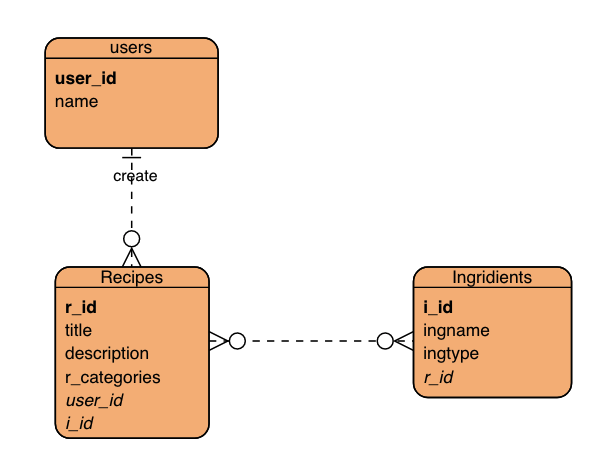
The application should load within 2-3 seconds and efficiently with minimum delays or frame freezes.

* + The UI should be user-friendly, have a bigger font, and be easy to navigate and press buttons.

# Use case Diagram



# ER Diagram



1. **Flowchart**
2. **Low fidelity prototype**
3. **High fidelity prototype**