Live Demo Outline

Problem statement: Many people struggle with deciding what to cook with the ingredients they have in their pantry. The limited variety of ingredients options and the lack of inspiration makes it difficult to come up with creative and nutritious meal ideas.

Solution: Created a web application that allows users to create a virtual pantry and generate recipe ideas from selected ingredients in the pantry.

Target Audience: People who want to cook meals at home.

Main Functionalities of our application:

1. User sign in and registration of account.
2. Add and remove ingredients for "MyPantry".
3. Generate recipes ideas from selected ingredients.
4. View saved recipes in a 'Saved Recipes' page.
5. Locate nearby grocery marts.
6. Edit User Profile.

APIs used:

1. Firebase Authentication API
2. Google sign in API
3. Spoonacular Recipe Information by ID API
4. Spoonacular search recipe by ingredients API
5. Google Map API
6. Google Places API
7. Firestore API methods
    - onSnapshot
    - getDocs
    - getFirestore
    - deleteDoc

Application Demo Script:

To use our mypantry webapp, we first need to create an account. Once clicked on “signup”, registration form is given.
For each input field, there are different validations for fullname, email and password. The signup button will be disabled until all input fields are valid. Once users successfully created an account, a message will be displayed to the user to check their inbox and verify their account. When users try to create an account using email that already exist in our app, an error message is sent.

Next for login users are given two options, either normal log in or login by google. If users tried to login without verifying their account, an error message is sent to the user until they verified it.

When users click on “Profile”, the app navigates to the profile page. In the profile page, user’s particulars are displayed, “View saved recipes” button and method for updating of profile fields are provided.

When user clicks on view saved recipes, it navigates them to saved recipes page and recipes saved by them are shown. Users can also remove recipes from their saved recipes.

Updating of profile form allows users to update their fullname and email while request users to type in their current password to confirm changes. For both input field, the same validation criterias are placed. Update profile button will be disabled until all inputs are valid.

If users entered an email that already has an account or wrong password, updating of profile will be unsuccessful and a error message will be displayed. Once users successfully update their profile, a message will be displayed to request users to check their new email’s inbox and verify the account. Users will then be logged out and asked to relogin with new credentials.

When users click on “Log out”, users will be navigated to the login page and in the event that users forgets their password and click on “Forgot Password?”, users will be navigated to the forgot password page and input their email address to reset their password.

<u>MyPantry outline</u>

- Right-hand side has all the possible ingredients the user can add to his pantry
- Clicking on an ingredient adds it to the pantry view a create operation by firebase
- Left-hand side shows the ingredients the user has in his pantry, which represents the ingredients the user has in his kitchen.
- Clicking on delete will toggle the mode to delete ingredients from the user’s pantry
- Clicking on an ingredient in the pantry adds it to the Ingredients to be used list
- Generate recipes will utilize the ingredients in the to-use list.
- Clear will clear all the ingredients in the to-use list.
- These are the recipes displayed
- It will show the ingredients used, ingredients not used, and ingredients missing.
- Clicking on the image will direct the user to the recipe website where the full information and steps of making the recipe are given.

- Save recipe will save the recipe into the user’s saved recipe page, which is a way for the user to favorite his recipe if he finds it delicious and wants to make it again.

Nearby Stores

- The nearby stores shows all stores in a 1500m radius within the user’s location
- The user can change the radius by inputting a number in the text field above and pressing enter
- If the user doesn’t input a valid number, a pop-up will appear.
- The icons represent the stores within radius of the user
- Clicking on the view in google maps will direct the user onto the google maps page
- The user can drag the little human figure onto the map to view the his surroundings
- The user can toggle between map and satellite view
- Below displays a list of stores that is within radius of the user.

Technologies we used:

1. React JS
2. Firebase
3. Google Cloud

Software Engineering Practices:

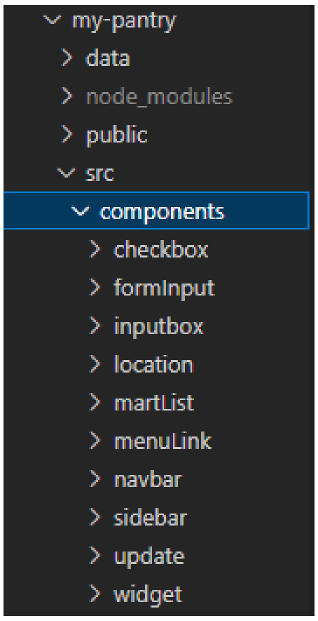
1. **Kanban (Agile Method)**
   Kanban is an agile method that allows us to visualize workflow and track project progress.

2. **Frequent Refactoring**
   We continuously modify and reorganize our code to improve its readability and make it easier to understand and maintain. The variables are also renamed to make their functionality and purpose clearer.

Software Principles Applied:

1. **The Separation of Concerns (SOC) Principle**
   Code is organized into small reusable components in ReactJs. This promotes modularity as we can isolate the functionality and make changes to the code without affecting other parts of the application.

2. **Don't Repeat Yourself (DRY) Principle**
Creating components such as the “navbar” that can be reused throughout the application. This reduces the amount of code to be written and avoids code duplication.



```
⌄ my-pantry
  › data
  › node_modules
  › public
  ⌄ src
    ⌄ components
      › checkbox
      › formInput
      › inputbox
      › location
      › martList
      › menuLink
      › navbar
      › sidebar
      › update
      › widget
```

<u>System Architecture diagram:</u>
We employ a three-tier architecture that organizes applications into three logical and physical computing tiers: the presentation tier, or user interface; the application tier, where data is processed; and the data tier, where the data associated with the application is stored and managed.

<u>Good Traceability:</u>

1. **Version Control**
   Used GitHub to track history and changes
2. **Documentation**
   We frequently update our SRS and Use Case Documents. It ensures that the requirements are well-defined and shown throughout the software development lifecycle.

<u>Design Patterns:</u> We used the MVC design pattern. We separated the application into three parts: the model, which represents the data and business logic; the view, which presents the data to the user; and the controller, which handles user input and updates the model and view

accordingly. For our code, we used React, which is a framework that supports the MVC pattern.

Testing: We performed black and white box testing on our web app. An example of our black box testing is for the “Creating an account” function. The equivalence classes for the different inputs are listed in the table on the left. The table on the right contains the test cases we have designed to test the function. In the first test case, all the inputs are from the valid input class, but in the test cases, all but one input is from the valid class.

An example of white box testing we performed is shown here for the “update profile” function. The control flow diagram can be seen in the figure on the right. From that, we have obtained 4 different execution paths as seen on the left-hand side. From this, we can obtain the cyclomatic complexity of 4.