Functional Specification

CA400 - 4th Year Project

Project Title:

‘MyAllergyAssistant - Android app to scan food ingredients for allergens’

Students:

- Daragh Prizeman - 19459734
- George Eskander - 19451972

Supervisor: Paul Clarke

Date of Completion: 07/11/2022

Table of Contents

1. Introduction

1.1 Overview

The system to be developed is an Android application using React Native[1], built to assist people with food allergies. The app will help reduce the likelihood of an allergic reaction by allowing the user to scan products quickly of the many products they buy. This can be done by the user either using Optical Character Recognition (OCR) technology or barcode scanning. In the case of barcode scanning, when a barcode is scanned, a request will be made to some external database or API such as Open Food Facts[2] to retrieve the product's information. Users will be able to report the product by scanning its barcode, this will be stored in our database. This will remind the user and other users with similar allergies that it may cause an allergic reaction. The system's backend will be built and hosted on the cloud through AWS Lambda[3], this will be used for user authentication, and use or store data that will support the functionality of the app. A list of allergens will be stored in a dataset based on up to date, credible sources[4].

We believe there is a need for this app because it will help keep users more informed about the products that they purchase and consume, as users will be able to scan a product's ingredients or barcode and see if it contains any of their allergens, or if the product has been reported by other users for containing any additional allergens not listed on the packaging. We hope that by keeping users informed about the products in this manner, that this app will help prevent consumers from having allergic reactions.

The system will require some initial information of the user's allergens, access to the user's camera and a working internet connection in order for the app to function as intended.

1.2 Business Context

This development of this application is not sponsored by any business organisation.

This system may be deployed using Docker and on the cloud through AWS.

[1] React Native
[2] Open Food Facts
[3] AWS Lambda
[4] FSAI, InformAll, FARE

1.3 Glossary

Docker - platform used for packing applications into containers.

React Native - framework made for making cross-platform mobile applications without having to rewrite code for each platform and without needing to write native code.

AWS - Amazon Web Services; cloud computing platform and APIs hosted by Amazon, which we will be using for the backend of this project.

API - Application Programming Interface; used for communications between different applications.

Barcode - a visual code that represents data that can be scanned, usually found on food packaging.

Optical Character Recognition (OCR) - The process of extracting text from images. Also referred to as ‘text-recognition’.

Open Food Facts - Open Food Facts is a food products database, that we will use to identify products (https://world.openfoodfacts.org/)

2. General Description

2.1 Product / System Functions

Below is a list of functions that our app will use.

- Register
- Login
- Input selection of user allergies
- Scan ingredients text
- Scan barcode
- Report product
- Edit selection of user allergies
- Edit account information
- Delete account

2.2 User Characteristics and Objectives

The users of the application will be people with Android phones, people who may have food allergies, or may want to be informed about whether or not a given food product contains any allergens.

From the user's perspective, this app should allow them to create an account, and store their allergy information. It should allow the user to take a picture of either the ingredients or the barcode on food packaging, and should display to the user whether the product is safe to eat or may contain any of their allergens. After scanning a product's barcode, the user should also be able to see if the product has previously been reported by another user for containing traces of an allergen not listed on the food's ingredients list.

Some additional features that may be on a user's wishlist are:

- The ability to edit account information at any time.
- The ability to view the products that you have previously scanned.
- The ability to search for a product by name, instead of scanning.
- The ability to see the total number of allergen detections performed by the app.

These potential features on the above wishlist are features that will not be prioritised initially, but will be explored provided we have the core functionality of the app completed and have enough time.

2.3 Operational Scenarios

A list of operational scenarios is given below to show, from the user's perspective, how the system will be used in different situations.

1. **User starts the application**

   The app's UI will load, if the user has not given permission, the app will ask for access to their camera. If the app is able to access the camera, then it will show an authentication component to login/register if they haven't logged in before for some time.

2. **User registers**

   If a user doesn't have an account, they can register by clicking the register button. They will input their information in the register form, and then submit. If submission is successful, their account will then be created in our database. If the submission fails, an error message is displayed to the user.

3. **User login**

   A user can log into their account, provided that they already created an account through the registration process. If a user tries to login to an account using invalid details, they will receive an error message.

4. **User selects allergens**

   If the user wants to select/change their allergens, they can go into account information, and tap to edit them. A list of allergens they have selected and not selected will then pop up where they can select/unselect and submit.

5. **User scans ingredients**

   A user can scan a product’s ingredients by pressing the ‘Scan Ingredients’ button. The app will then use the user’s camera to scan the ingredients on the product packaging. The user will receive a warning message if the app cannot recognise any text from the image (if the image is too blurry etc). A message will then be displayed to the user indicating that the product is either safe to eat, or may contain one of their allergens.

6. **User scans product barcode**

   A user can scan a barcode by pressing the ‘Scan Barcode’ button. The app will then use the user’s camera to scan for the barcode on the product packaging. If detected, the app will make a request for the product’s information to an external API/database such as Open Food Facts. If found, a result will be returned to the user whether it is safe to eat or not, potentially including any other product information if available. If not found, a user will receive a warning message saying the product could not be found.

7. **User reports product**

   A user can report a product if the product caused them to have an allergic reaction to an allergen that wasn’t listed on the product’s ingredients list. When another user then scans the product’s barcode, they will be warned that the product was reported by another user.

2.4 Constraints

A list is given below of the constraints put on the design team:

- This application should be compatible with all Android phones, provided they have a working camera and an internet connection.
- The app should take no longer than 10 seconds to scan the ingredients of the product. An additional 10 seconds should be accounted for if translation is required.
- The app should take no longer than 10 seconds to retrieve the product information after scanning the barcode.
- The team has no prior experience with technologies such as OCR, barcode scanning, AWS Lambda, and machine learning, time will be needed to learn these.
- The development of this app is constrained by time, as the final deliverables for this project are due on the 28th of April 2023.

3. Functional Requirements

Below is a list of the functional requirements for our application to work as intended.

3.1. Set app permissions

Description

The user must be able to accept permissions from the app to access their camera and/or photo library. If the app is missing a permission, the app must ask the user for permission in order to continue using the app.

Criticality

This requirement is a high priority, app permissions are required in order for the app to function properly. The app needs access to the photo in order to scan the product.

Technical Issues

May need to check if a user is potentially able to remove a required permission while using the app in order to prevent bugs.

Dependencies with other requirements

N/A

3.2. Register

Description

The user must be able to register for an account by pressing the 'Register' button on the home screen. They will then have to fill in the registration form with their details including their name, email address and password. Provided the details provided are valid, the user's account will then be registered and added to our database. If their details are invalid, then an error message will be displayed to the user.

Criticality

The registration functionality is critical for this application to authenticate our users, and for them to be able to login to the app. We will work on the registration and login functionality together at the beginning of the development process.

Technical Issues

AWS is a new technology to the development team, so they will have to learn how to implement user authentication using AWS as a backend,

Users' account information, such as email and passwords, will need to be stored securely using AWS Cognito.

Dependencies with other requirements

The registration and login functionality goes hand in hand for user authentication, so we will focus on these at the beginning of the development process. Users must have accepted their permissions, see requirement 3.1.

3.3. Login

Description

After having registered their account, the user must be able to login into their account through the login component, the component must be accessible to the user through the authentication screen, by the user tapping the login button or the app starting at the login component. The login component must have a form where the user should be able to enter and submit their login credentials, and then log the user in if their credentials are correct as in the database. If the credentials provided by the user are incorrect then an error message should be displayed.

Criticality

This function is required otherwise the user would not be able to log back into the app and use its features. We will work on the registration and login functionality together at the beginning of the development process.

Technical Issues

AWS is a new technology to the development team, so we will have to learn how to implement user authentication using AWS as a backend.

Users' account information, such as email and passwords, will be stored securely using AWS Cognito.

Dependencies with other requirements

The login and registration functionality goes hand in hand for user authentication, so we will focus on these at the beginning of the development process. Users must have accepted their permissions, and registered, see requirements 3.1 and 3.2 respectively.

3.4. Select user allergies

Description

When the user logs in to their account for the first time, they will be prompted to input their allergy information by selecting the food allergens that affect them. The user will have the ability to add and remove allergens linked to their account at any time through the settings screen.

Criticality

It is critical to the core functionality of our app that the user is able to select their allergens. The application will search for these allergens in the scanned product's ingredients.

Being able to update their allergens is of medium priority, as it doesn't affect the core functionality of the app.

Technical Issues

Users' allergy information will be kept securely in our database using DynamoDB[5].

[5] DynamoDB

Dependencies with other requirements

The user must have accepted permissions, and be logged in to their account in order to select their allergens. See requirements 3.1 and 3.3 respectively.

3.5. Take/select image for scan

Description

Users must have the ability to take a picture using their camera, and also be able to select an image from their photo gallery on their phone. This image will then be used by the app to scan product ingredients or a barcode.

Criticality

This functionality is of critical importance, as the app needs to be able to get an image from the user.

Technical Issues

We will need to ensure that the image selected by the user is clear enough for the OCR to work. If the app cannot detect a barcode or any ingredients, then an error message will be displayed to the user.

Dependencies with other requirements

The user must have accepted permissions, and be logged in to their account in order to select their allergens. See requirements 3.1 and 3.3 respectively.

3.6. Scan product barcode

Description

With the user's inputted image, the app should be able to recognise the barcode that the user provides. Using the barcode information, the app must try to retrieve product information through an external database (Open Food Facts API), find any allergens, if any, and give the user their scan results (see requirement 3.8.).

Criticality

This functionality is of critical importance, as the app needs to be able to scan products to tell the user if it is safe to eat or not.

Technical Issues

The user is responsible for the image/camera to be in a well-lit environment, in focus, and with the barcode being displayed in full flatly enough to be recognised. If the app cannot detect a barcode, then an error message must be displayed to the user.

The app may need to manipulate the image in order to improve accuracy. If product information is retrieved, it may need to be translated if it is in a foreign language.

Dependencies with other requirements

The user must have inputted a photo and the results of the scan must come after scanning the barcode. See requirements 3.5. and 3.8. respectively.

3.7. Scan product ingredients

Description

With the user's inputted image, the app should be able to recognise the text which potentially contains ingredients, find any allergens, if any, and then give the scan results (see requirement 3.8).

Criticality

This functionality is of critical importance, as the app needs to be able to scan products to tell the user if it is safe to eat or not.

Technical Issues

The user is responsible for the image/camera to be in a well-lit environment, in focus, and with the product ingredients being displayed in full flatly enough to be recognised. If the app cannot detect any ingredients, then an error message must be displayed to the user.

The app may need to manipulate the image in order to improve accuracy. If product information is retrieved, it may need to be translated if it is in a foreign language.

Dependencies with other requirements

The user must have inputted a photo and the results of the scan must come after scanning the ingredients. See requirements 3.5. and 3.8. respectively.

3.8. Allergen Identification Algorithm

Description

Once the app has scanned the product and got the text of its ingredients list, this text will be the input to our allergy identification algorithm to check if the product contains an allergen or not. The output of this algorithm will be the scan results message that will be displayed to the user. When deciding if a product is safe to eat or not, our algorithm will first scan the ingredients list for any of the user's allergen keywords. If any of the user's allergens are detected in the product ingredients, then a warning message is displayed to the user that the product is not safe to eat.

Next, if none of the user's allergens are found in the product ingredients, the algorithm will check all previous reports related to the scanned product, to check if the user's allergen was listed as a potential cause to the allergic reaction. If the user's allergen is found as a potential cause in any of the reports, a warning message will be displayed to the user, and they will be advised that the product may not be safe to eat.

If none of the user's allergens are found in the product ingredients or in any of the previous reports of the product, then the product will be deemed safe to eat, but the user will still be made aware of all reports of the product for other allergens too.

Criticality

This is of critical importance, as it is essential to the functionality of our app that the user is informed if the product is safe to eat or if it contains one of their allergens, or if the product has previously been reported by another user.

Technical Issues

The result of the scan should be clearly displayed to the user, so that if a product contains a user's allergen there cannot be any confusion.

If no ingredients can be detected, then an error message should be displayed to the user.

Dependencies with other requirements

The user must be logged in, and have inputted a photo to be scanned. See requirements 3.3 and 3.5. In order for the results of the scan to be shown to the user, the scan must first have taken place. If the user has scanned a product that has been reported by at least one other user, the result should mention this in the result. See requirements 3.6, 3.7, and 3.9.

3.9. Report product

Description

If the user experiences an allergic reaction to a product that doesn’t list any of their allergens in the product ingredients, the user should be able to report the product by pressing the ‘Report product’ button in our app. The app should then store the product information and user allergens involved in our database.

If the user is unsure which one of their allergens was in the product, the app will first display a message to the user recommending them to consult with a doctor to check if they have an allergy that they weren’t aware of to something that was listed in the ingredients list. If the user is sure that they are not allergic to anything that was listed, but still had an allergic reaction to the product, then the app will mark all of the user’s allergens as potential causes of the allergic reaction in the report. Then when another user scans the product, they will see all of the potential allergens that have been previously reported, and will be advised that the product may not be safe to eat.

Criticality

This functionality is of medium priority, as it is a key feature of the application, however the other key features would still work without it.

Technical Issues

We will need to research if reporting a product can be supported, by scanning the product barcode or inputting the product name or both.

Dependencies with other requirements

The user must be logged into their account, in order to report a product. See requirement 3.3.

3.10. Update account

Description

A user should be able to update their account details, such as resetting their password at any time through the settings screen. The user should receive a verification email when updating their account information.

Criticality

This is of low priority, as it doesn’t affect the core functionality of the application.

Technical Issues

The user's details will be stored securely using DynamoDB when updating their account information.

Dependencies with other requirements

The user must be logged in to their account, in order to change their account information, such as email and password. See requirement 3.3.

3.11. Delete account

Description

A user should be able to delete their account at any time, which would remove the user’s account from our database.

Criticality

This is of low priority, as it doesn’t affect the core functionality of the application.

Technical Issues

We need to be careful when deleting a user’s data from our database, that any unrelated data doesn’t accidentally get deleted.

Dependencies with other requirements

The user must first be logged in to their account, in order for them to be able to delete it. See requirement 3.3.

4. System Architecture

Below is a graphical representation of the architecture behind our application.

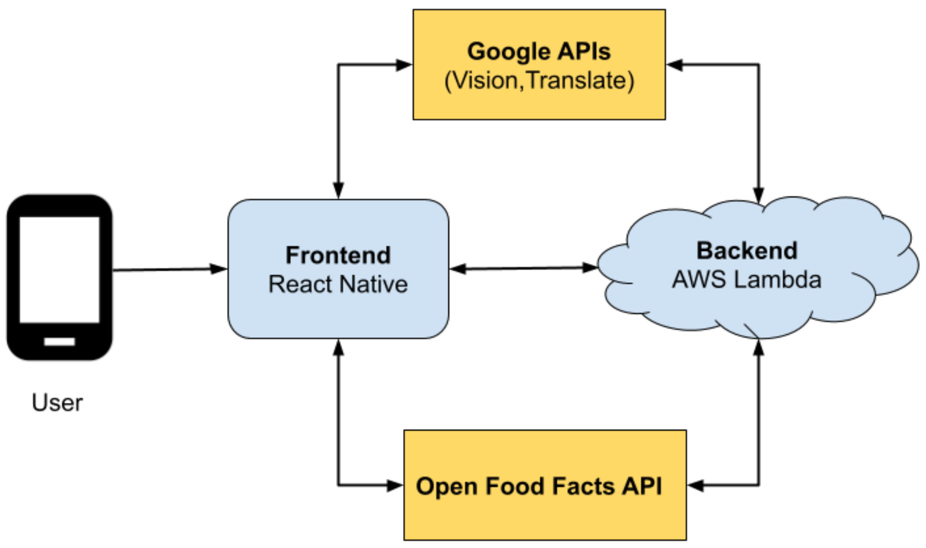

Figure 1. System Architecture Diagram

Frontend
We will be using React Native for the frontend development of our system. It will be used to create the UI for the Android application and will communicate with the backend and third party components using API calls.

Backend

AWS Lambda will be used for our backend development of the system. We plan to use DynamoDB for our database, Cognito for user authentication, and any other useful AWS technologies.
DynamoDB and Cognito will be used to secure data such as credentials.

3rd Party Components
We plan to make use of the following 3rd party components:

- Open Food Facts API - to identify a product from the barcode.
- Google Cloud Vision API[6] - potentially for OCR.
- Google Translate API[7] - to translate ingredients text from a foreign language to English.

[6] Google Vision API
[7] Google Translate API

5. High-Level Design

Below is a data flow diagram which displays how the system will work.

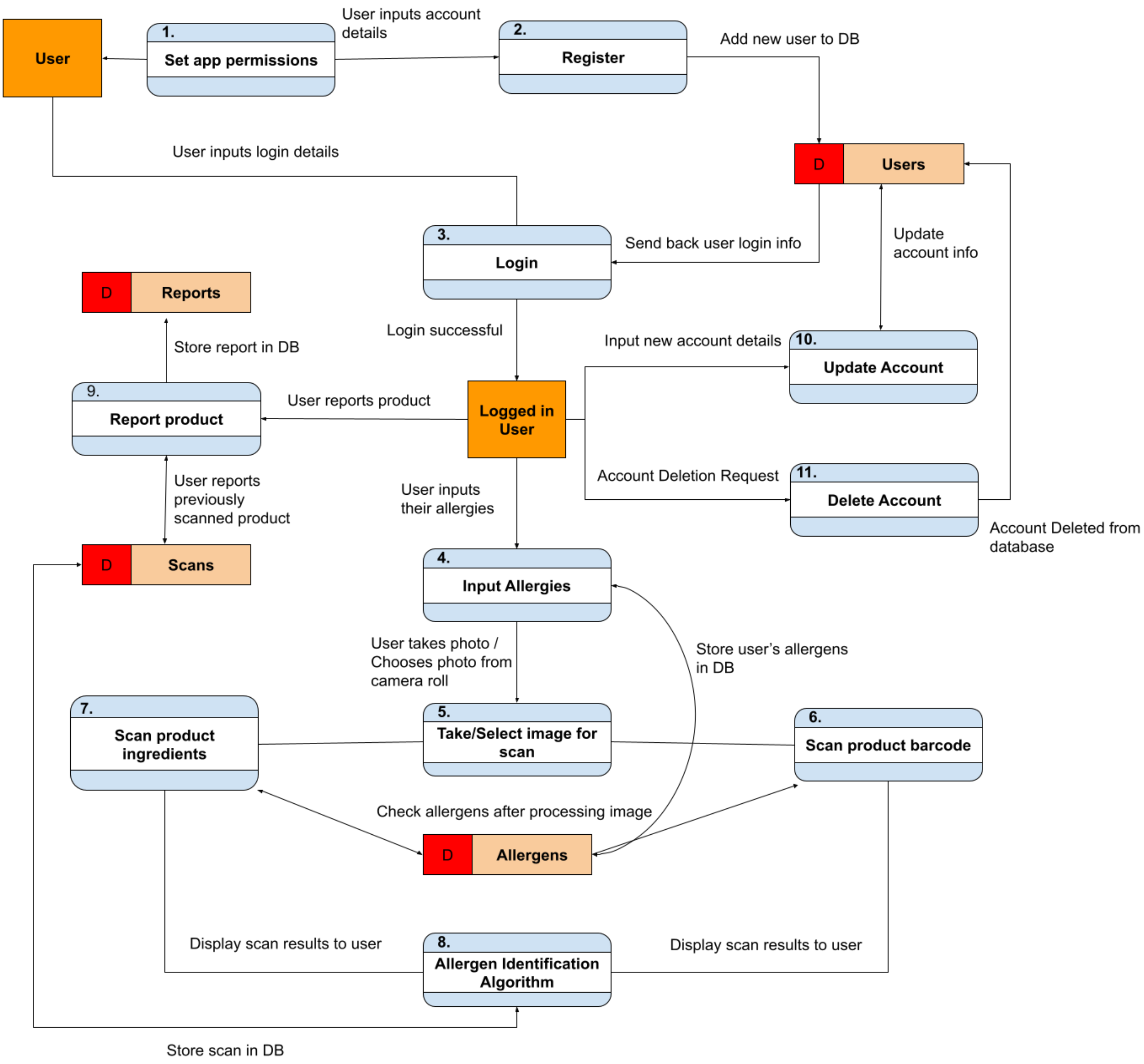
User
1.
Set app permissions
User inputs account details
2.
Register
Add new user to DB
D
Users
User inputs login details
3.
Login
Send back user login info
Update account info
D
Reports
Store report in DB
Login successful
Input new account details
10.
Update Account
9.
Report product
User reports product
Logged in User
Account Deletion Request
11.
Delete Account
User reports previously scanned product
User inputs their allergies
Account Deleted from database
D
Scans
4.
Input Allergies
User takes photo / Chooses photo from camera roll
Store user's allergens in DB
7.
Scan product ingredients
5.
Take/Select image for scan
6.
Scan product barcode
Check allergens after processing image
D
Allergens
Display scan results to user
8.
Allergen Identification Algorithm
Display scan results to user
Store scan in DB

Figure 2. Data Flow Diagram

6. Preliminary Schedule

Below is a GANTT chart that displays our preliminary schedule for this project. As discussed in our project approval, we will be using a pair-programming approach working together on each section using Code With Me[8] and/or LiveShare.[9]

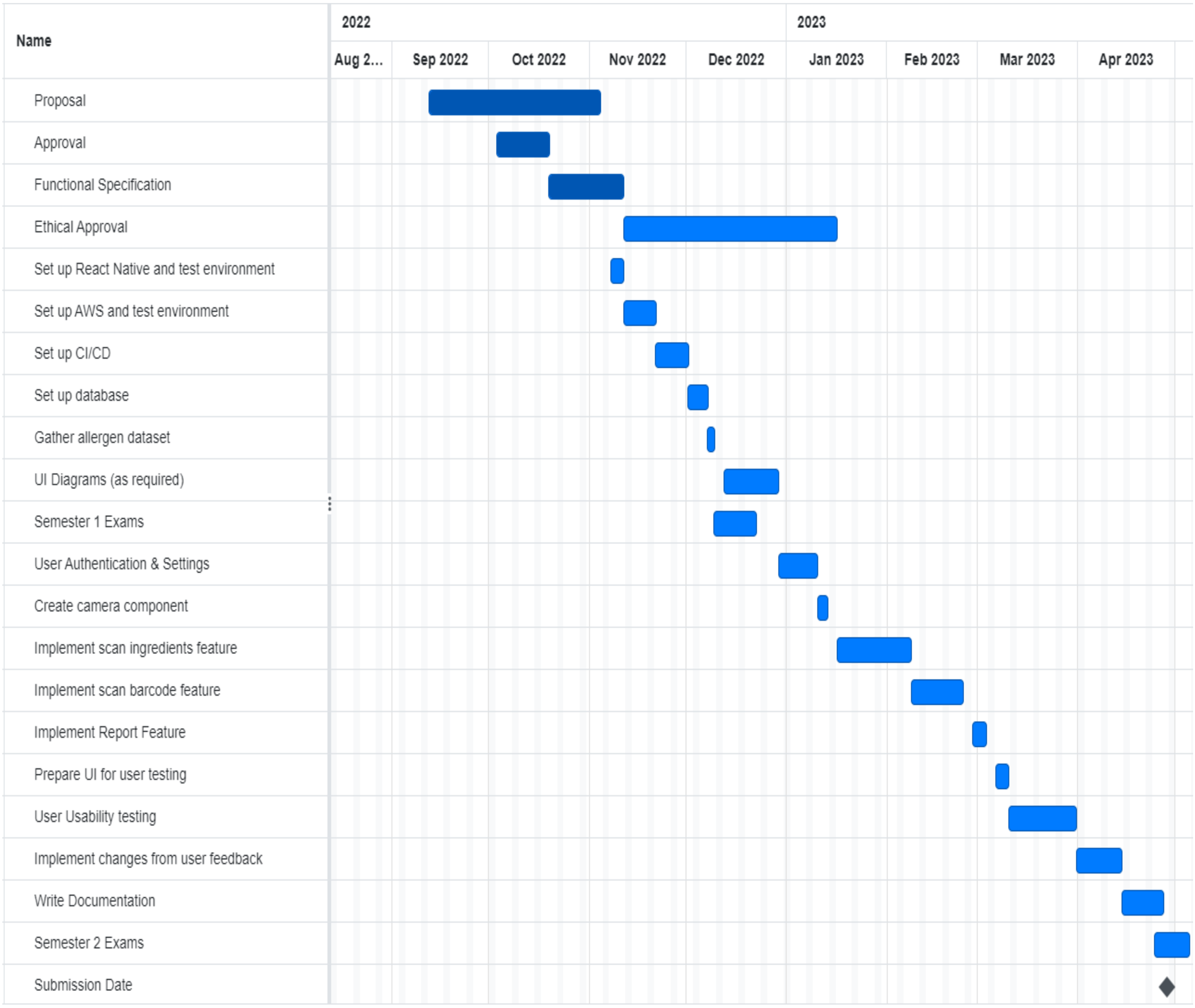

Figure 3. Preliminary Schedule Gantt Chart

[8] JetBrains Code With Me
[9] Visual Studio Code's LiveShare