Cognac Bandits - Final Report SENG2021 Deliverable 5

Part 1: Requirements Analysis

Problem Statements

- There is no search app or system to find popular cocktails
- There is no centralised platform or forum for people to discuss and suggest cocktail combinations
- There is currently no system which provides a shopping list for what ingredients you are missing to make a particular cocktail
- Current discussions and articles relating to cocktails are messy and confusing
- Novice cocktail makers, have no centralised application where you can seek instructions on how to make a cocktail

Purpose of our System

Our web application, Liqueur, is designed to address the issues raised in the above problem statements. Liqueur acts as a centralised platform where cocktail makers can search and find cocktails with filtered assistance, along with a community driven recommendation system. Through the large access of information about cocktails and ingredients, our web application is able to provide users with a means of formulating shopping lists and discussions about cocktails, assisting them in their cocktail making journey.

Updated User Stories

Story #1:

Feature: Search cocktail by ingredient

As a: cocktail maker

So that: I can find a particular cocktail containing ingredient/s

I want to: search cocktails based on their ingredients

Scenario: Searching for a cocktail by its ingredients

GIVEN I am on the homepage of the web app

WHEN I enter an ingredient on the search bar input box or choose an ingredient from navigation on the top of the page
THEN I can see the cocktails matching with ingredient/s
WHEN I can click on a specific cocktail
THEN I am redirected to cocktail page with its information

Story #2:

Feature: Search cocktail by name

As a: novice cocktail maker

So that: I can know how to make a specific cocktail

I want to: search cocktail name to view method to make it

Scenario: Searching for a cocktail to find method on how to make it

GIVEN I am on the homepage of the web app
WHEN I enter the name of a cocktail on the search bar input box
THEN I can see the cocktail/s listed
WHEN I can click on the cocktail
AND I am redirected to cocktail page with its information
THEN I am presented with the method on how to make it

Story #3:

Feature: Viewing Discussion Forum for Cocktails

As a: novice cocktail maker

So that: I can read suggestions on specific cocktails

I want to: browse a forum of comments posted by users for the cocktail I am looking at

Scenario: Viewing a specific cocktail comments in the forum

GIVEN I am on the homepage of the web app
WHEN I enter the name of a cocktail on the search bar input box
THEN I can see the cocktail/s listed
WHEN I can click on the cocktail
AND I scroll to the bottom of the cocktail information page
THEN I can read comments and suggestions made by other users on this cocktail

Story #4:

Feature: Creating an account

As a: regular cocktail maker

So that: I can post comments on cocktails

I want to: engage in conversations / discussions on cocktail pages to expand my knowledge on cocktails

Scenario: Creating a user account

GIVEN I am on the homepage of the web app
WHEN I click on the 'Register' button on the top right of the navigation bar
THEN I am redirected to the register page
WHEN I enter a new username and a password into the input boxes
AND I click on 'Register' button below the input boxes
THEN I am logged into the website with my account

Story #5:

Feature: Logging into an account

As a: frequent user of the web application

So that: I can access my account to post comments

I want to: log into my account.

Scenario: Logging into a user account

GIVEN I am on the homepage of the web app
WHEN I click on the 'Sign In' button on the top right of the navigation bar
THEN I am redirected to the sign in page
WHEN I enter my username and password into the input boxes
AND I click on the 'Sign in' button below the input boxes
THEN I am logged into the website with my account

Story #6:

Feature: Logging out of an account

As a: frequent user of the web application

So that: I ensure security that no one comments under my name

I want to: log out of my account

Scenario: Logging out of a user account

GIVEN I am on the homepage of the web app
AND I have registered and logged in
WHEN I click on the 'Logout' button on the top right of the navigation bar
THEN I am signed out of my account

Story #7:

Feature: Add all ingredients to Shopping List

As a: novice cocktail maker

So that: I can make a cocktail from scratch

I want to: form a shopping list of ingredients I need to make a cocktail

Scenario: Creating a fresh shopping list of a particular cocktail

GIVEN I am on the homepage of the web app
AND I have registered and logged in
WHEN I search on the input box or choose a particular cocktail from navigation on the top of the page and make selection from drinks
AND I click the 'Add to shopping list' button beside the name of the cocktail
THEN The ingredient/s are added to my shopping list
WHEN I click the 'Shopping List' tab on the navigation bar on the top of the page
THEN I can view the shopping list with the added ingredients

Story #8:

Feature: Add individual ingredient to Shopping List

As a: cocktail maker

So that: I can make a cocktail with ingredients I have

I want to: form a shopping list of ingredients that are left to complete ingredient list of a cocktail

Scenario: Creating a completing shopping list of a particular cocktail

GIVEN I am on the homepage of the web app
AND I have registered and logged in
WHEN I search on the input box or choose a particular cocktail from navigation on the top of the page and make selection
AND I click on 'Add to shopping list' in the dropdown menu of the ingredients that I am missing
THEN The ingredients are added to the shopping list
WHEN I click the 'Shopping List' tab on the navigation bar on the top of the page
THEN I can view the shopping list with the added ingredients

Story #9:

Feature: Commenting on Forum for Cocktails

As an: expert / experienced cocktail maker

So that: I can post suggestions and comments on specific cocktails

I want to: browse a cocktails and post comments for advice / recommendations

Scenario: Commenting on a specific cocktail

GIVEN I am on the homepage of the web app
AND I have registered and logged in
WHEN I enter the name of a cocktail on the search bar input box or choose the cocktail from navigation on the top of the page
THEN I can see the cocktail/s listed
WHEN I can click on the cocktail
AND I scroll to the commenting section below instructions on the cocktail page
THEN I can view the comment text box
WHEN I enter a message in the text box
AND I click on the 'Post' button
THEN I can see the comment posted by me on the cocktail page

Story #10:

Feature: Deleting own comments on the forum

As an: expert / experienced cocktail maker

So that: I can remove my unwanted or incorrect comments posted

I want to: remove my old comments which could be outdated

Scenario: Deleting a comment on the page of a specific cocktail

GIVEN I am on the homepage of the web app
AND I have registered and logged in
WHEN I enter the name of a cocktail on the search bar input box or choose the cocktail from navigation on the top of the page
THEN I can see the cocktail listed
WHEN I can click on the cocktail
AND I scroll to the commenting section below instructions on the cocktail page
THEN I can view the comments
WHEN I press 'Delete' next to one of my comments
AND I press 'OK' for the confirmation of deleting the comment
THEN I can see the comment has disappeared on the cocktail page

Story #11:

Feature: Viewing the information of ingredients of a cocktail

As an: novice cocktail maker

So that: I can learn more about what the specific cocktail is made up of

I want to: see the information of ingredients of a cocktail

Scenario: Viewing the information of an ingredient after searching for a cocktail

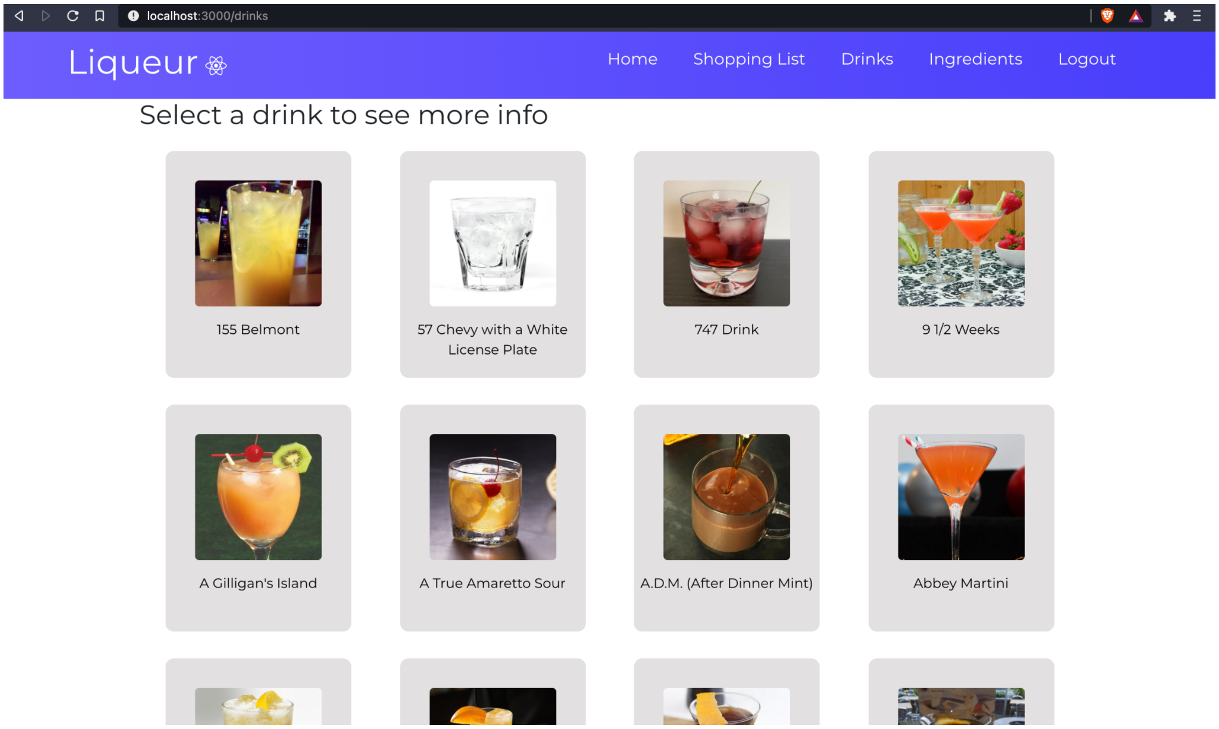
GIVEN I am on the homepage of the web app
AND I have registered and logged in
WHEN I enter the name of a cocktail on the search bar input box
AND I click on a cocktail
THEN I can see the cocktail's information
WHEN I click on one of the ingredients of the cocktail
AND I click on 'View more info' on the dropdown menu
THEN I can see the ingredients' information

Cognac Bandits - Final Report SENG2021 Deliverable 5

Achievements

Based on the requirements derived from our problem statements and user stories, our web application has successfully provided a centralised platform where a range of cocktails can be searched for, as seen in *Figure A* below.

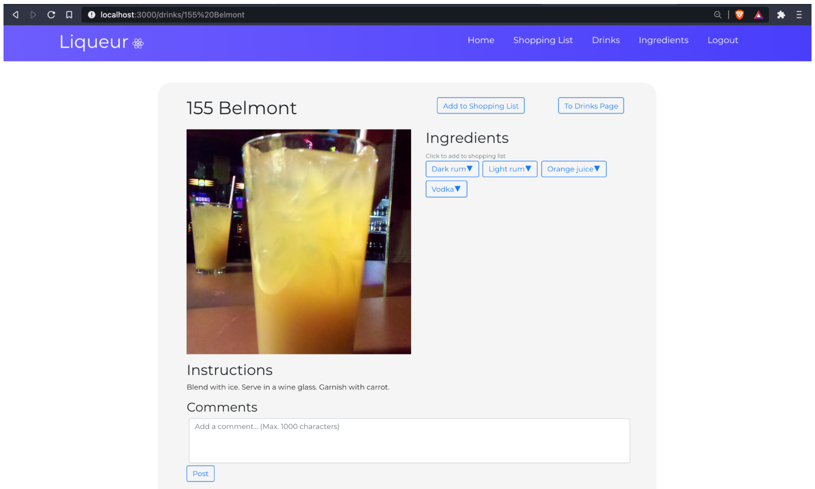
Figure A:



localhost:3000/drinks

Select a drink to see more info

Furthermore, our web application has provided a means of discussion and recommendations through the commenting system on each cocktail page, seen in ***Figure B:***



localhost:3000/drinks/155%20Belmont

155 Belmont

Ingredients

Click to add to shopping list

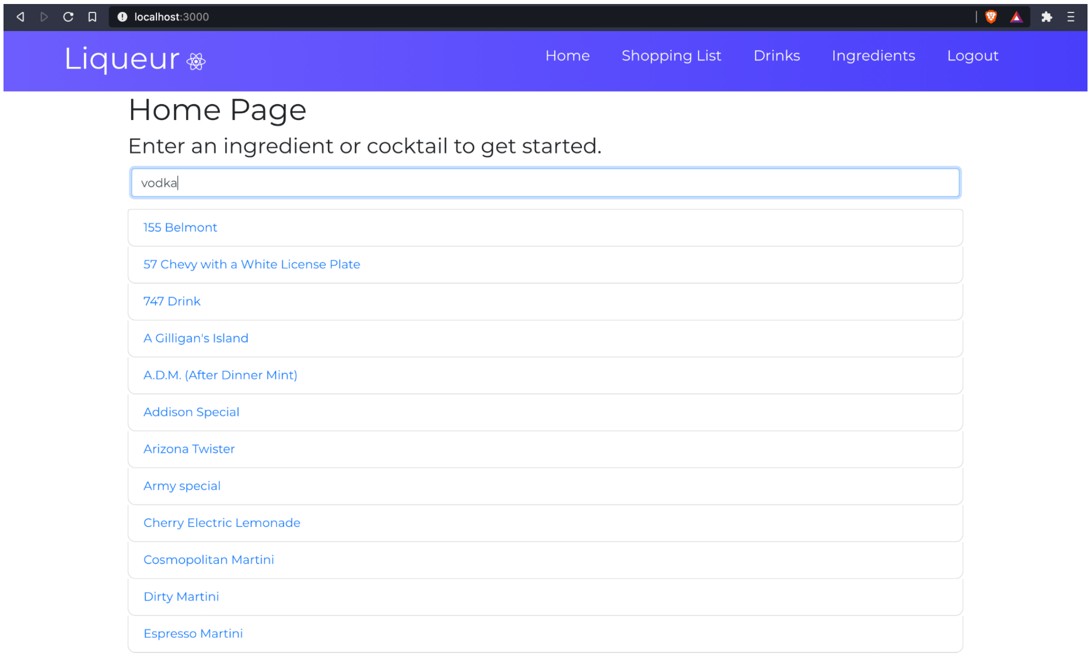
Instructions

Blend with ice. Serve in a wine glass. Garnish with carrot.

Comments

Add a comment... (Max. 1000 characters)

With these sources of information available to users, they are able to utilise tools such as the search bar and shopping list to explore and learn more about cocktail making. The search bar allows users to filter through specific ingredients or cocktails which they favour, allowing for cocktail makers to explore their preferences. This is seen in ***Figure C*:**



Home Page

Enter an ingredient or cocktail to get started.

vodka

- 155 Belmont
- 57 Chevy with a White License Plate
- 747 Drink
- A Gilligan's Island
- A.D.M. (After Dinner Mint)
- Addison Special
- Arizona Twister
- Army special
- Cherry Electric Lemonade
- Cosmopolitan Martini
- Dirty Martini
- Espresso Martini

Part 2: Software Architecture

External Data Sources

The main external data source the web application will be using is the free *TheCocktailDB* API.

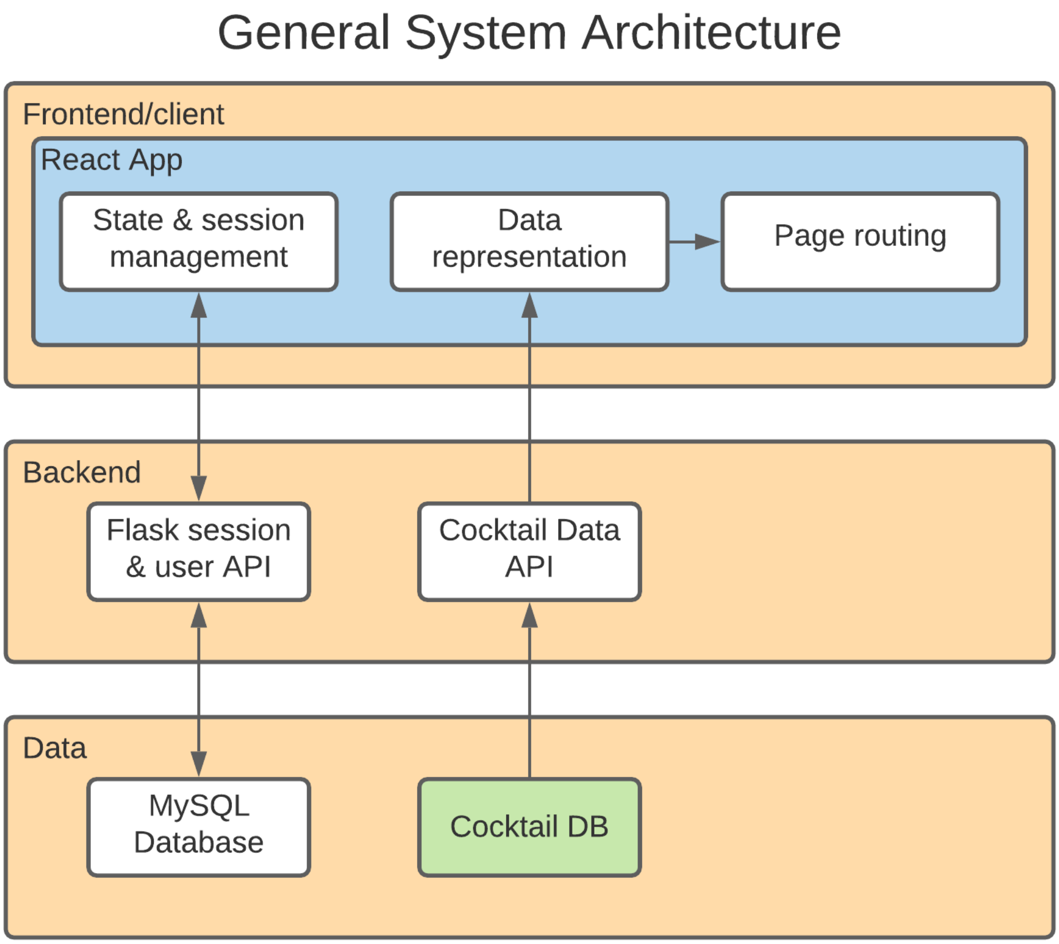
The API has features which allow the user to search through a database of cocktails and filter through it. There are methods to search the database by cocktail name and a method to search by cocktail ingredient. These methods help in satisfying the first user story referring to the user wanting to search a particular cocktail or ingredient. The API also can look up full cocktail details which satisfies the second user story referring to searching for details on how to make a specific cocktail.

Software Components (Web Stack)

The software components architecture gives an abstract representation of the system as a whole and the primary communications between each of the significant software components.

Figure 1 interlays said significant software components into the desired architecture of the system, with arrows denoting abstract data flow. This diagram gives an idea of the structure but more detail on each component will be described in the following sections.

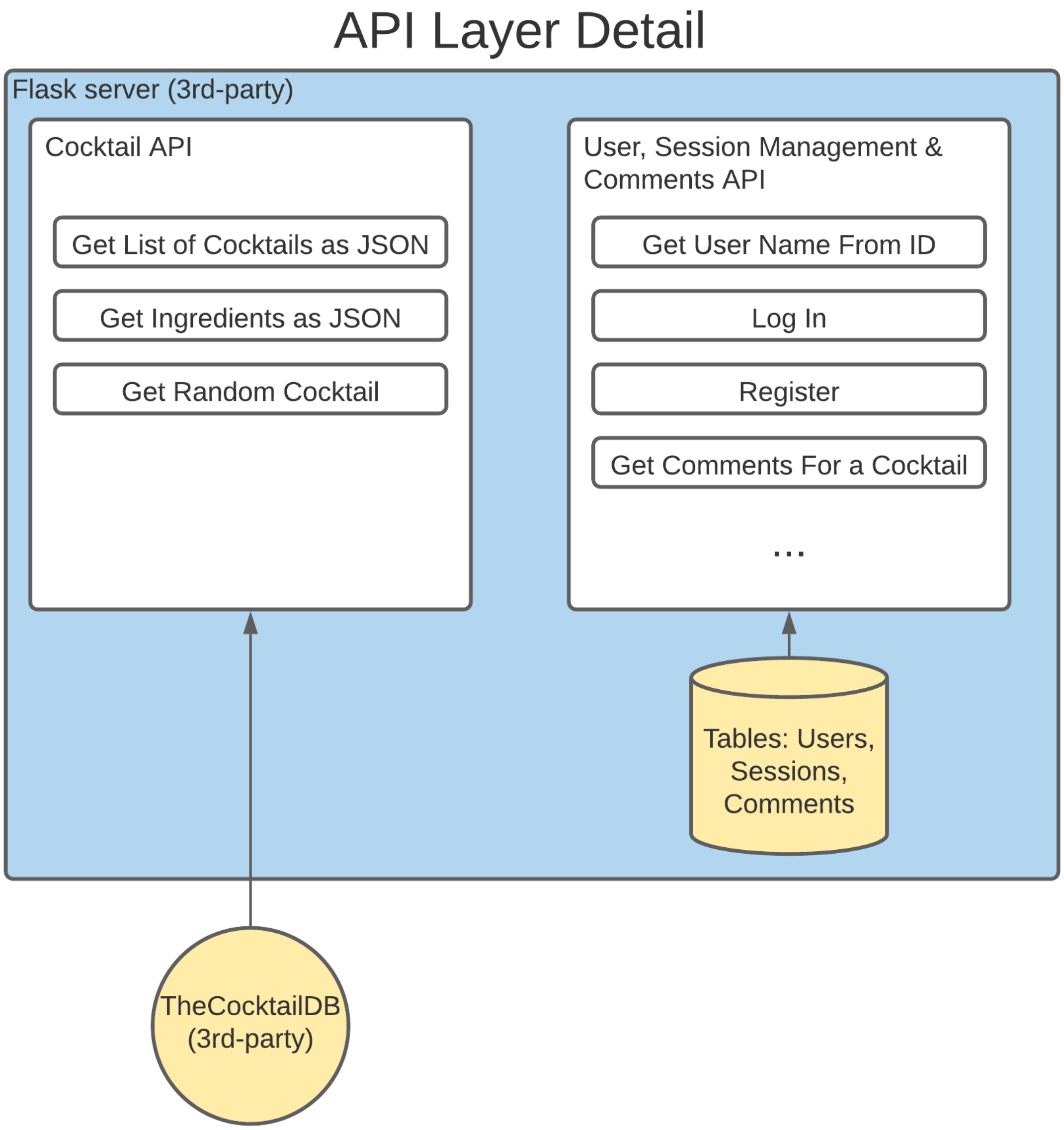
Figure 1



Backend / API Layer Detail

This section will give further detail on the API layer of the software, namely the servers that run independently of the client web application. The API layer is built with Flask in Python. The Flask API provides two primary functions, one being providing cocktail information to the client and the other performing user operations. Each of these distinct functions will be detailed below and their structure is represented in *Figure 2*.

Figure 2


API Layer Detail
Flask server (3rd-party)
Cocktail API
Get List of Cocktails as JSON
Get Ingredients as JSON
Get Random Cocktail
User, Session Management & Comments API
Get User Name From ID
Log In
Register
Get Comments For a Cocktail
...
Tables: Users, Sessions, Comments
TheCocktailDB (3rd-party)

Cocktail API

The Cocktail API will be the primary source of Cocktail data to be delivered to the user. This includes a wide range of cocktails with ingredients, a short description and the steps to follow to create the cocktail. This data is to be collected from *TheCocktailDB*, a public REST API providing cocktail information. This API gives access to 100 cocktails with the required data (ingredients, recipe, etc). The reason this API has been chosen by us is in large part because it is free. Furthermore, it follows an intuitive REST format which will reduce the time spent in development learning how to interface with it.

Our cocktail API can be seen as a mere wrapper for *TheCocktailDB*, providing only a minimal subset of the data and organising the information into a consumable JSON format. In particular, our Cocktail API will select cocktail name, ingredient, recipe data o 100 cocktails from *TheCocktailDB*, and format the data as JSON before delivering it to the client application.

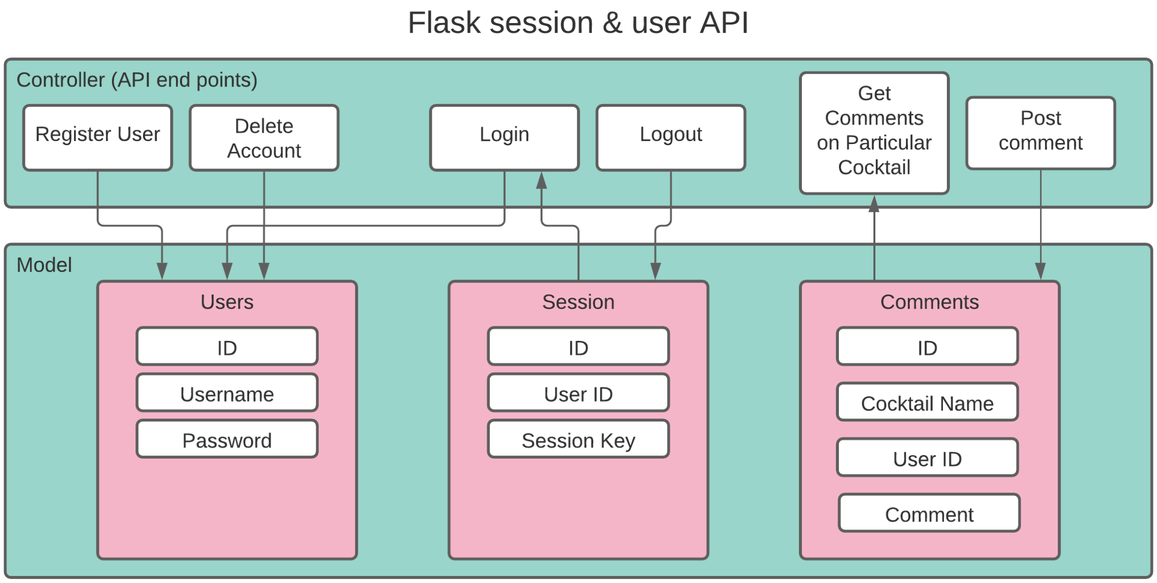
User, Session Management & Comments API

The second API server provides dynamic utility largely pertaining to users. This API will follow a REST structure. A REST structure will help separate user, session and comment data and functionality into cohesive routes.

One area of concern is the potential of high coupling to arise in placement of the commenting system. Primarily, comments are both related to cocktails (they are comments about cocktails), and users (users post comments under their username). However, comments functionality isn’t placed in the Cocktails API in which cocktail names are managed. We have grouped our commenting functionality into the latter API since cocktails names are static, so a coupling between comments and cocktail objects in the Cocktail API needn’t be developed.

Figure 3 gives a general representation of the API in terms of the Controller and Model. The Controller will handle REST routes such as “auth/login”, “comments/add/”, etc. Furthermore, the controller will delegate the procedures to follow in order to execute the appropriate task (e.g. log in: validate user credentials and create a user session).
The Model will represent database information as Python objects to allow simple interfacing between the controller and the database. Python dictionaries will be used to retrieve, create, delete and modify data entries in the database.

Figure 3



Frontend / Client Detail

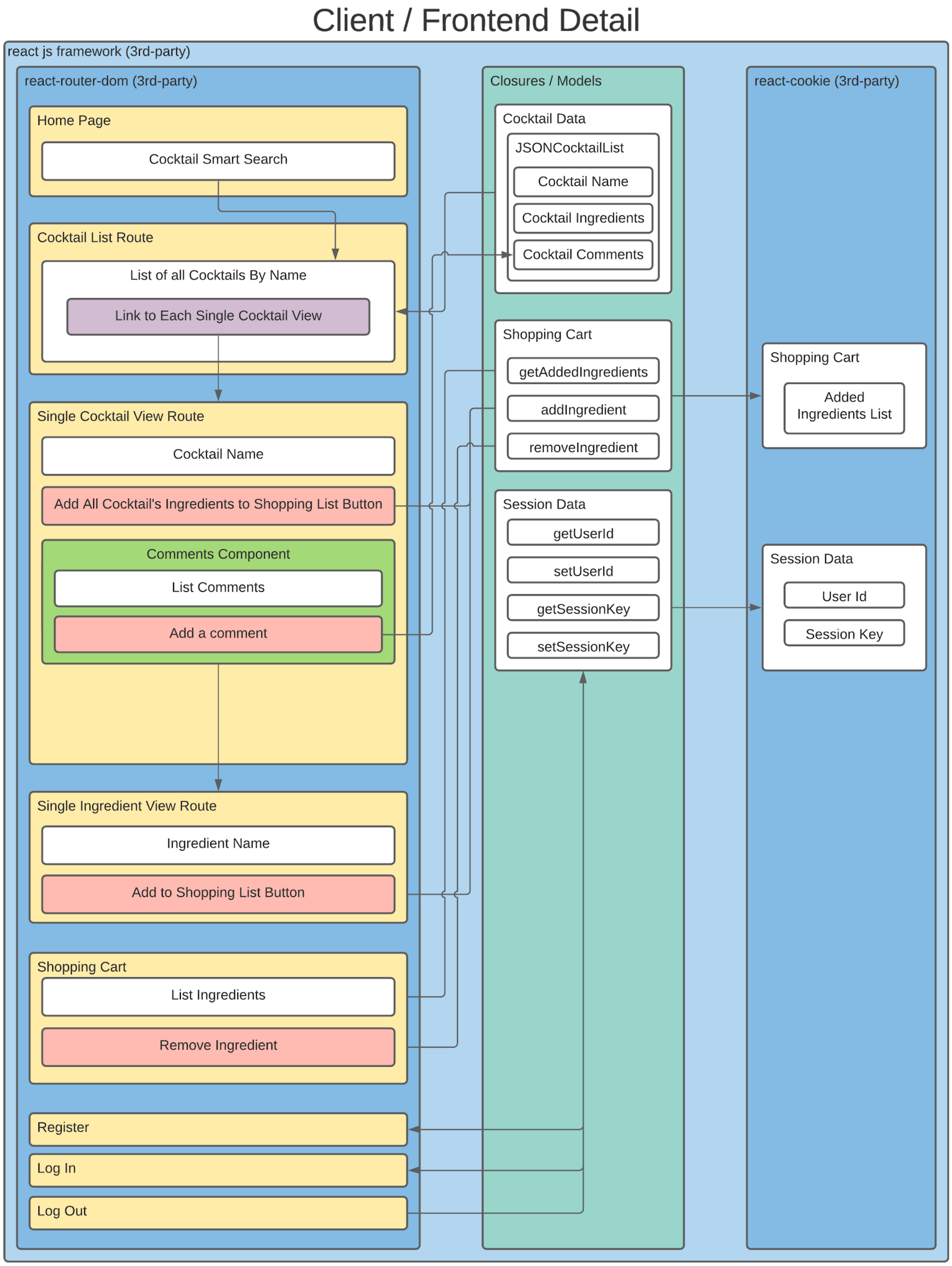
The client application functions as a React application. ReactJS is a front end JavaScript library used for building user interfaces. React can be used as a base in the development for single-page applications or mobile applications. React applications usually require the use of additional libraries for routing.

Of important note is the *Bootstrap* CSS framework. We have used *Bootstrap* on the client web application to allow us to use visually aesthetic CSS components. This was implemented into our React application using *react-bootstrap* which allows the use of Bootstrap components in the form of react components.

Figure 4 gives a visual on the components of the client. In representing the client application, we can separate the client into 3 distinct - albeit abstract - sections. These include the pages, closures and cookies

Cognac Bandits - Final Report SENG2021 Deliverable 5

Figure 4



The pages are associated with a particular URL which directs the user to the correct page via the router (*react-router-dom)*. These pages allow navigation amongst each other and logical separation of UI/UX. The pages largely are mere representations of data collected from the closures/models.

Great consideration has been placed into the methods in which data is stored and managed in the client. It is very important that we have a centralised state container for the websites in order to store vital and all-important information such as cocktail data, the shopping cart and session data. This is important since all of this information needs to be accessible by all pages and components on the website.

Significant is the comparison between using *Redux* alongside *redux-react-session,* and *react-cookie* implemented via closures*.*
Redux is a widely accepted solution to a centralised state container for *React* applications. This would work in combination with *redux-react-session* which provides further utility for storing user sessions. However, *Redux* is a large package with a complex API with more features that we need. Using *Redux* could result in large increases in development time and time learning *Redux's* API. This is why we have chosen a more straight-forward approach, that is using a small package *react-cookie,* a simple API for web cookies. We use this within closures, methods that represent state and that are accessible to the entire application. These closures will have procedures for getting a setting data to be stored website-wide in web cookies. In addition, the closures would encapsulate API integration with our backend servers, namely the Cocktails API and the User, Session management & Comments API.

Database Detail

The database consists of Users and Comments dictionaries. The Users dictionary stores the user information (username and password) as well as their shopping list. The Comments dictionary stores the comments posted on cocktail pages by specific users. Finally, sessions store the information needed to allow users to stay logged in on the client.

Cognac Bandits - Final Report SENG2021 Deliverable 5

Relating Choices to Components

Our application will be using ReactJS for the front end with Bootstrap for the styling, Python for the back end and Flask as a web framework.

We have decided to use React since many members of our team have previously used it. That being said, not everyone in our team has experience with the Javascript language so a simple javascript framework will aid in allowing quick development. React allows us to use web components which significantly reduces duplicate code and helps with maintainability via encapsulation of logic and the ability to use a component in more than one place. React also allows us to implement routing with the third-party package *react-router-dom*. This allows us to logically separate our pages despite the website being a single-page application.

Bootstrap is a free CSS framework directed towards front-end web development utilising CSS and javascript designs for topography, forms, buttons and navigation. Bootstrap features a responsive web design approach which allows web pages designed with it to render well on a variety of devices such as mobile devices as well as different window or screen sizes. All of these features will allow us to make an appealing front end that caters for a wide range of users.

Python is a well rounded language making it perfect for creating the back end of the web application. We have decided to use the Python Flask framework since the majority of our team have used it before. Flask is largely a minimal framework being described as a "micro web framework". This will allow us to produce quick prototypes during development. In particular Flask only provides the bare necessities for our web application including a server runtime and route handling, with the potential to add database and external API interfacing.

We decided to split the backend functionality into two separate API servers because our backend consists of two cohesive sets of functions, one being the management of static cocktail data and the other being web app utility including the management of user, session and comment data.

Flask is a micro web framework written in Python. Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

Choice of Platform

The web application we are building will be designed to work with UNIX like platforms such as macOS, Windows and Linux, with the main requirements of having an internet connection and a web browser.

For Windows users, for complete compatibility the web application is designed for Windows 7, Windows 8, Windows 8.1, Windows 10 or later and an Intel Pentium 4 processor or later that's SSE3 capable. With macOS support, OS X El Capitan 10.11 and later releases will be required for the web application. Finally for Linux, the web application is compatible with Ubuntu 14.04+ and Debian 8+. These are the platform requirements for current support Chromium-based browsers, hence the choice of platform for our web application.

The preferable web browser would be Google Chrome, as the application is being tested on Chrome browsers. However, other Chromium based browsers would work such as Opera, MS Edge, Vivaldi and Brave. Other non-Chromium based browsers such as Safari are tested for basic fundamental functionality of the web application, but not guaranteed to have supported formatting with alignments and fonts.

Our web application does feature support for mobile devices, meaning the pages are dynamic to the window size and shape of the browser. As both Android and iOS have support for Chromium-based browsers, the web application will be compatible with these operating systems. Specifically, at least Android Lollipop 5.0 and iOS 12 to be able to run a browser compatible with the web application.

As a result of implementing support for mobile devices, the application will be able to adapt to different screen sizes and resolutions.

Cognac Bandits - Final Report SENG2021 Deliverable 5

Summary Table

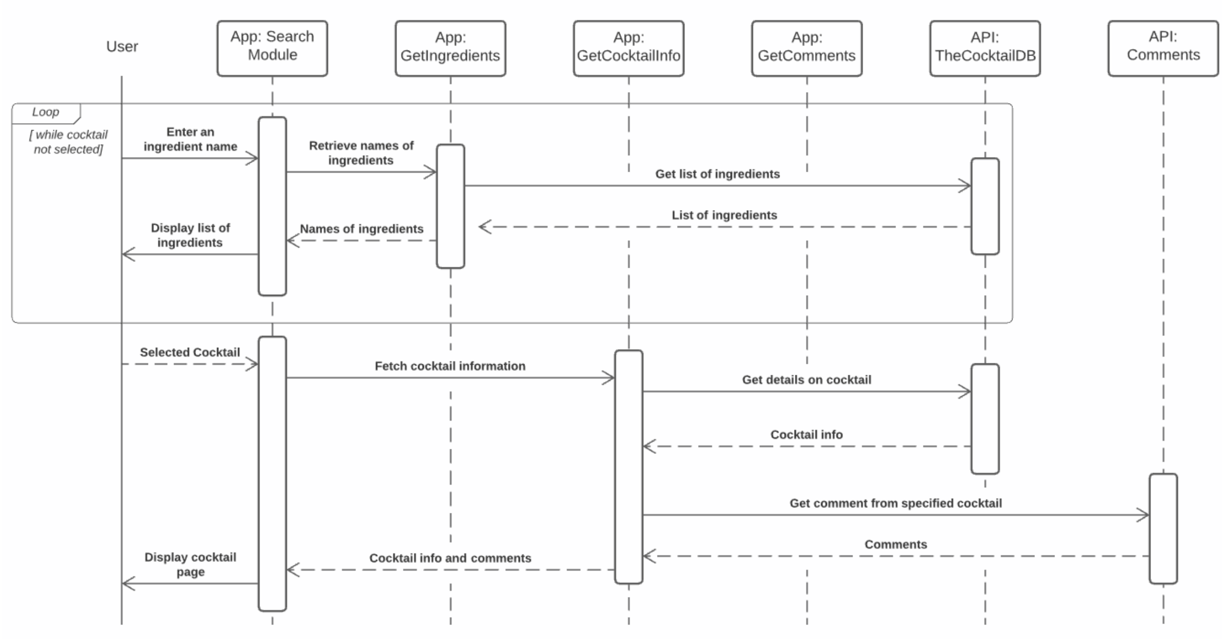
Architecture element	**Choice**	**Benefit/Achievement**
External data sources	TheCocktailDB(API)	Provides sufficient details on the information including ingredients and instructions on most cocktails.
CSS framework	Bootstrap	Abundant resources are accessible, easy for beginners, UI style meets our requirements.
JavaScript library	ReactJs	Simple to use, team has pre-existing experience with React
Web framework	Flask	Team has experience in developing application with Flask, simple and flexible for beginners,
Platform	Windows, macOS, Linux (Chromium-based browser supported) Mobile operating system (iOS, Android)	Browsers e.g. Google Chrome is available on all systems, widely used, cross-platform.
Backend programming language	Python	The programming language for Flask, open-source, cross-platform, many useful libraries for development.

Part 3: Software Design

Sequence Diagrams

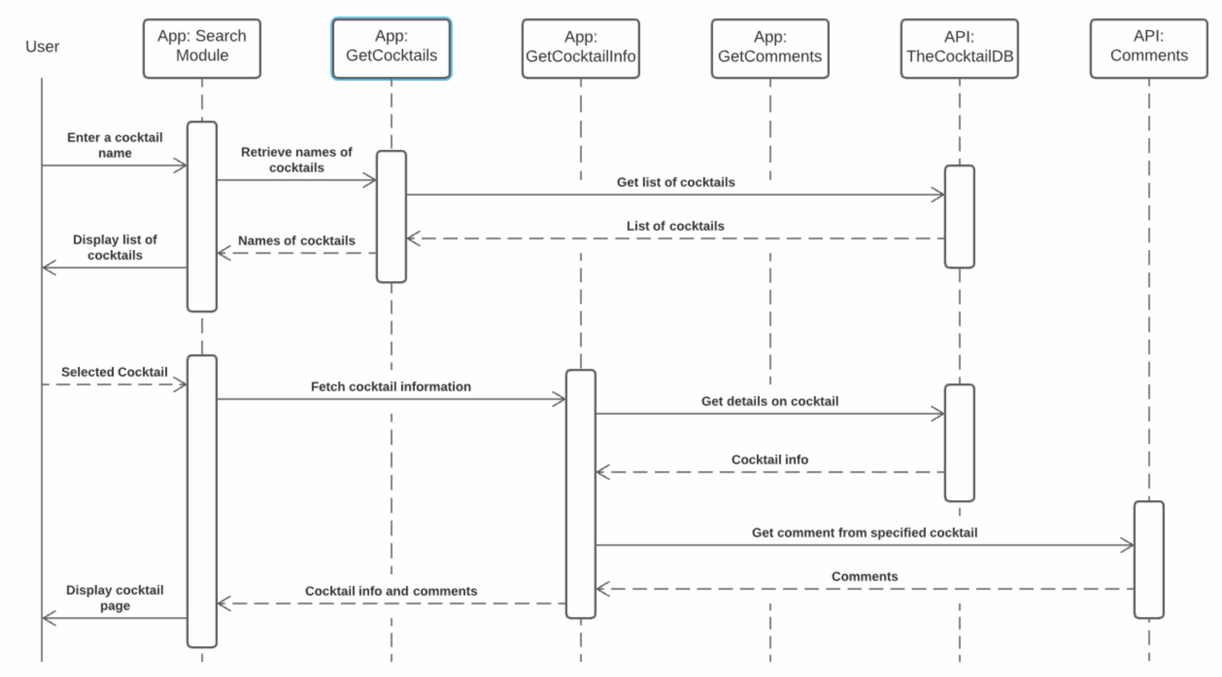
User Story 1: Search cocktail by ingredient

Figure 5 - Sequence Diagram 1



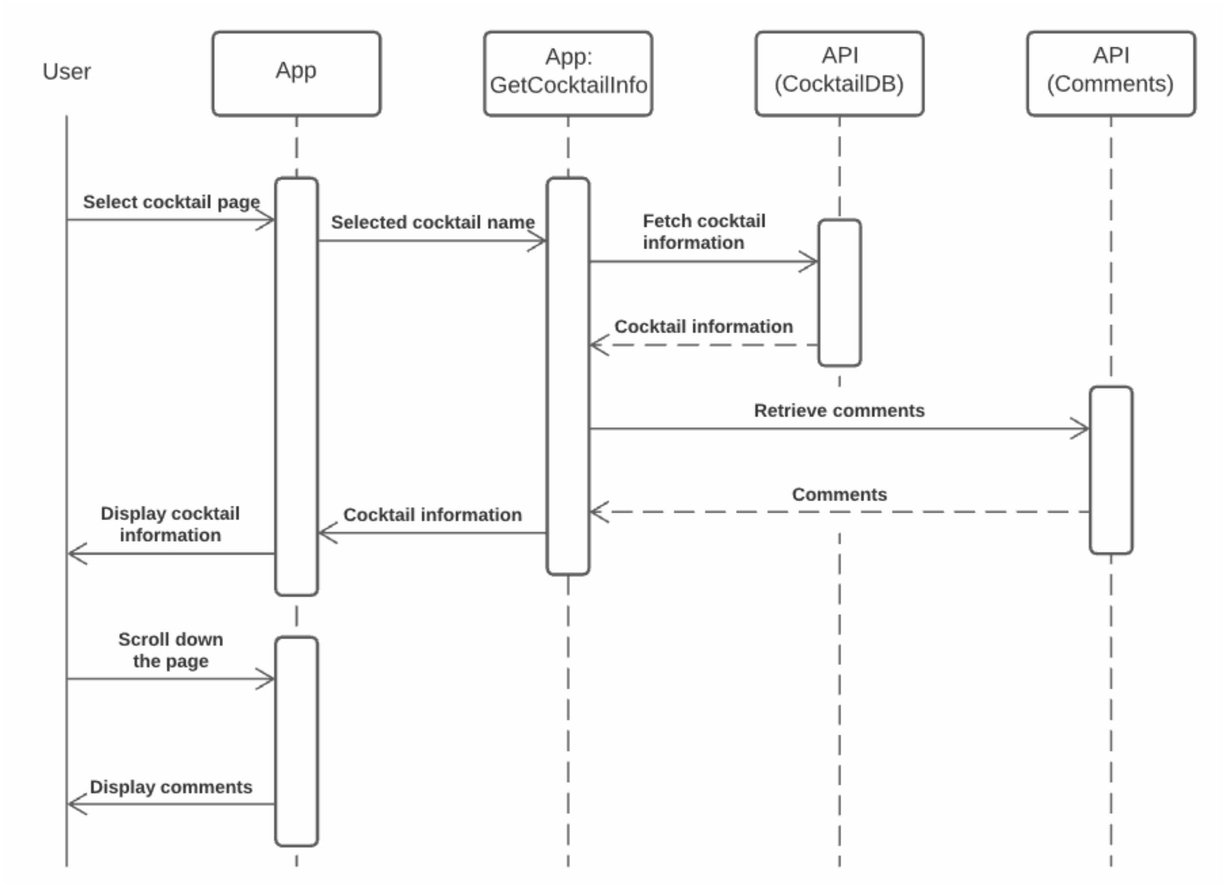
User Story 2:Search cocktail by name

Figure 6 - Sequence Diagram 2



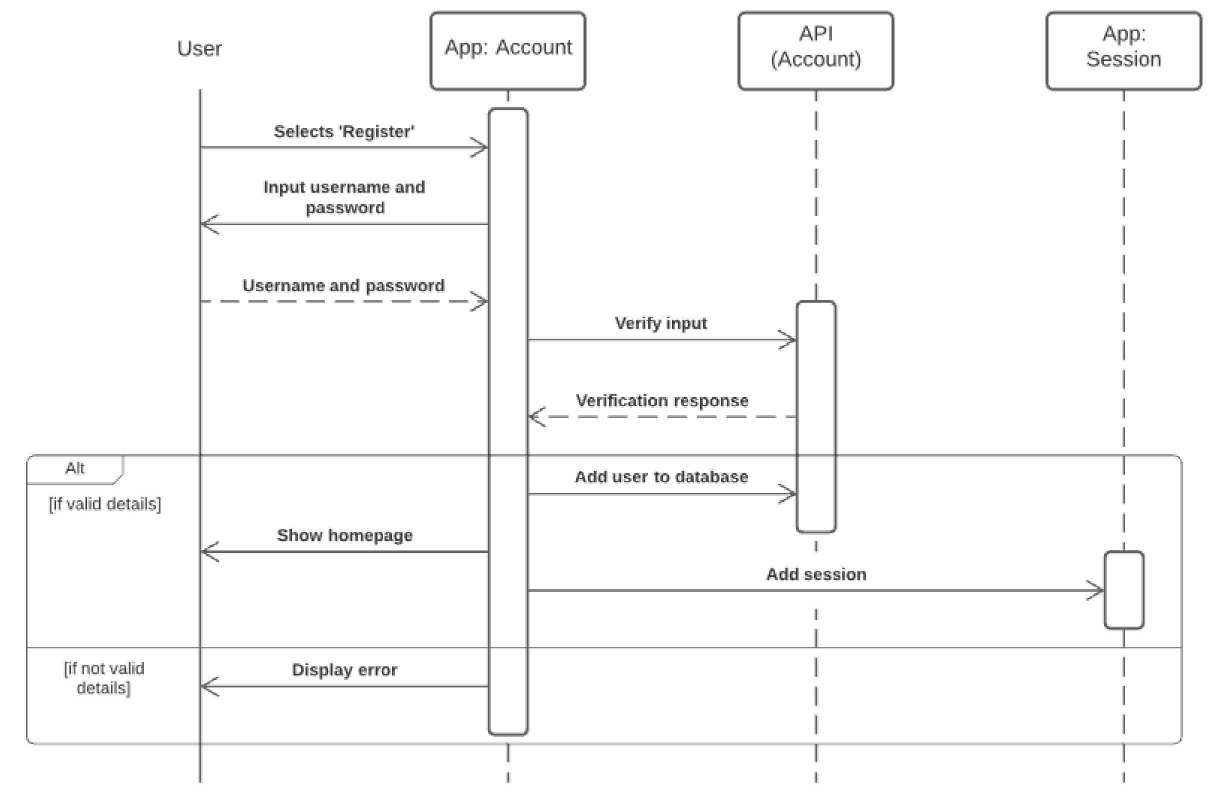
User Story 3: Viewing Discussion Forum for Cocktails

Figure 7 - Sequence Diagram 3



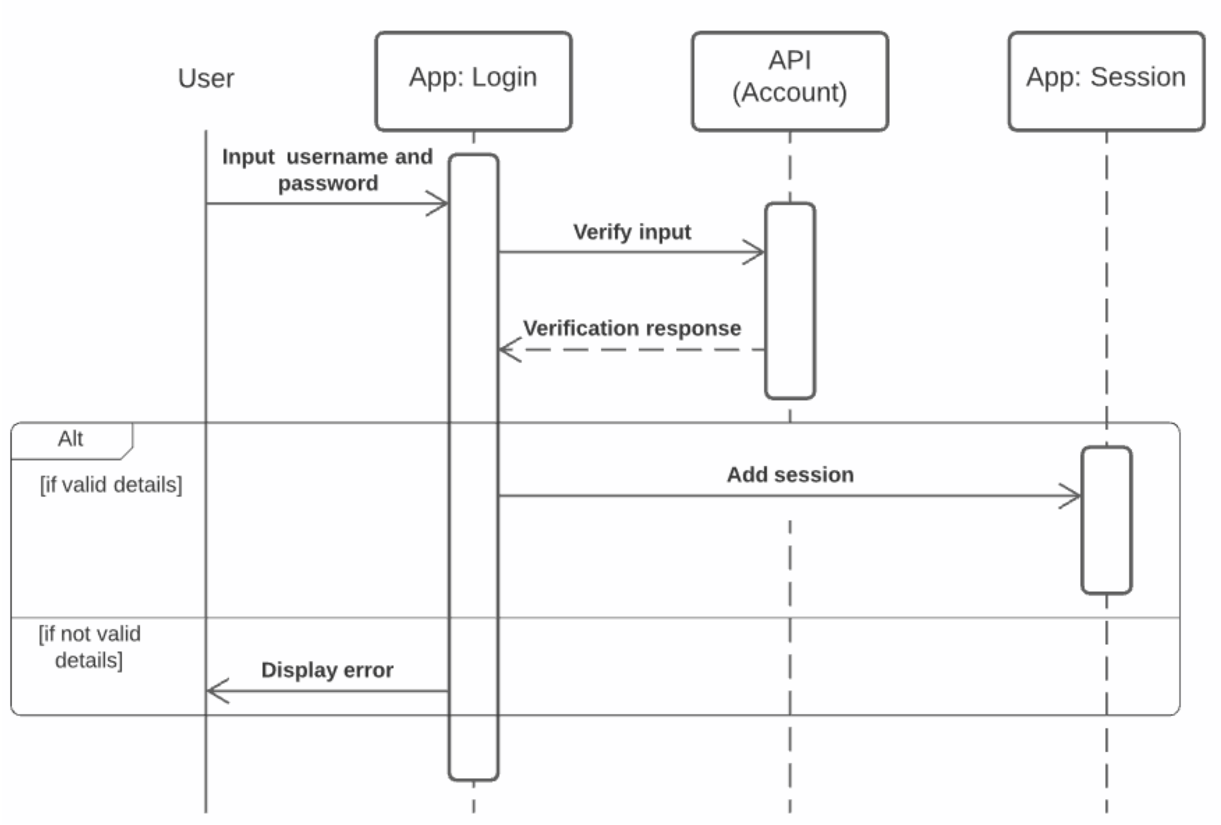
User Story 4: Creating an account

Figure 8 - Sequence Diagram 4



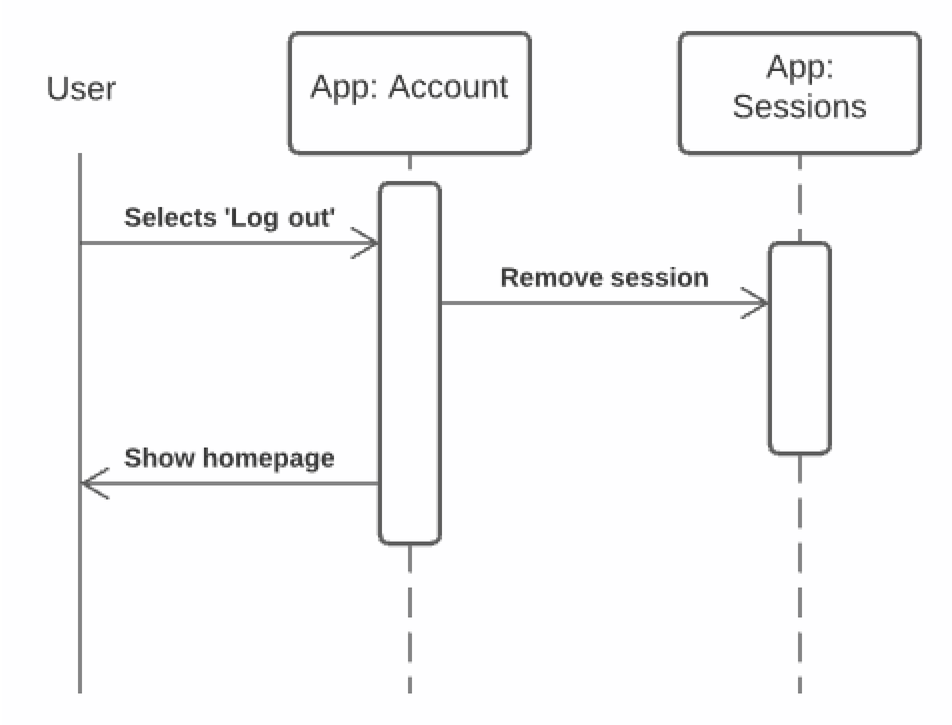
User Story 5: Logging into an account

Figure 9 - Sequence Diagram 5



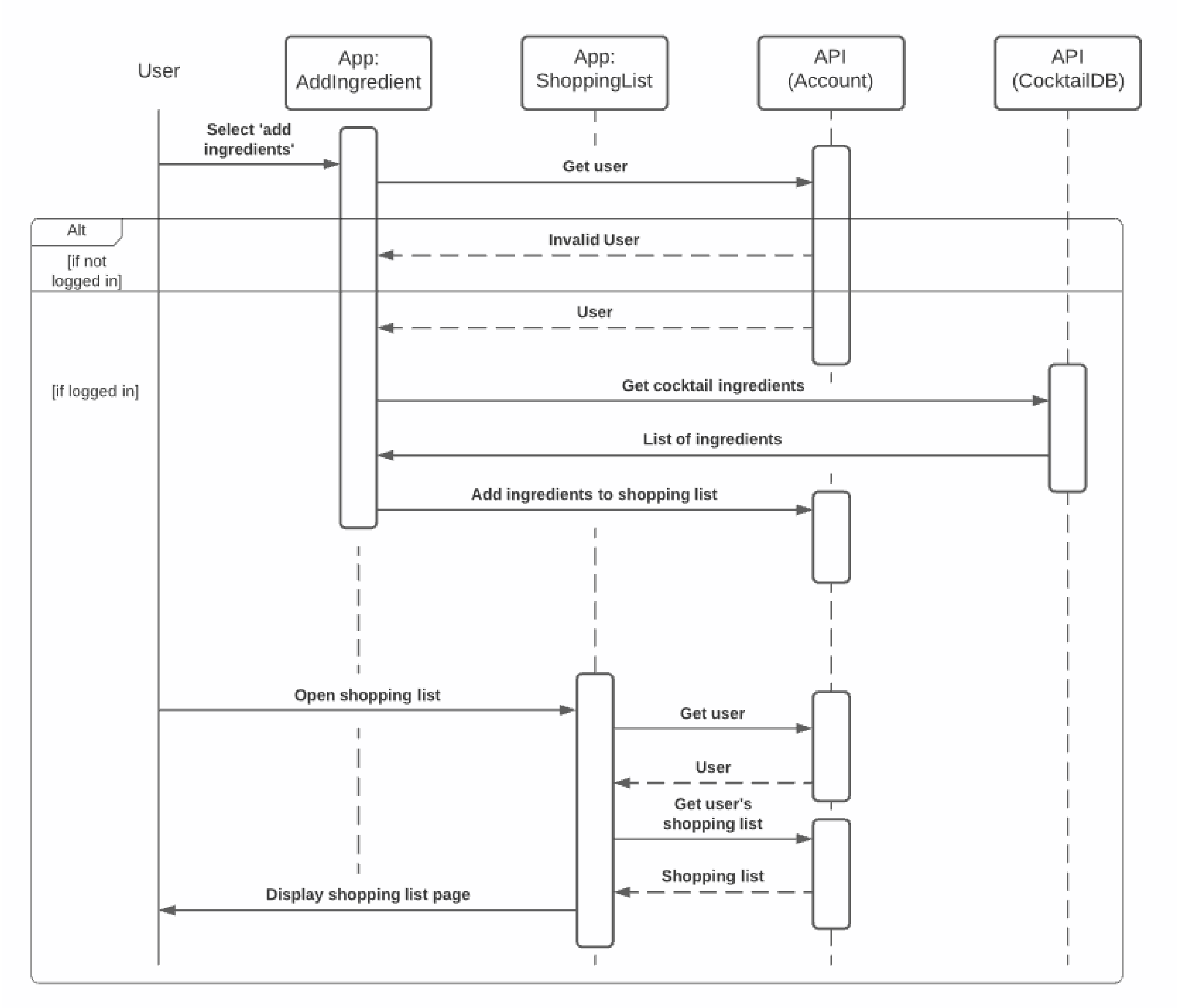
User Story 6: Logging out of an account

Figure 10 - Sequence Diagram 6



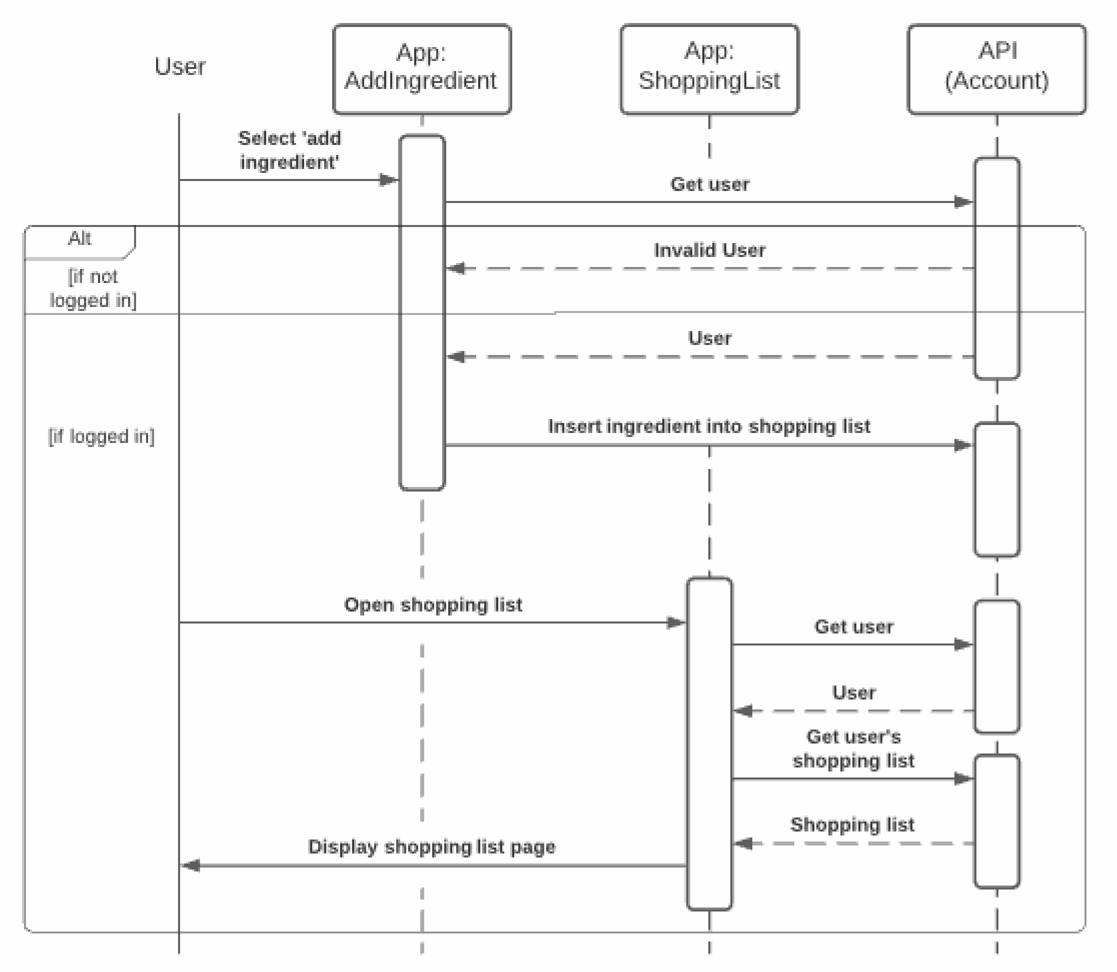
User Story 7: Add all ingredients to Shopping List

Figure 11: Sequence Diagram 7



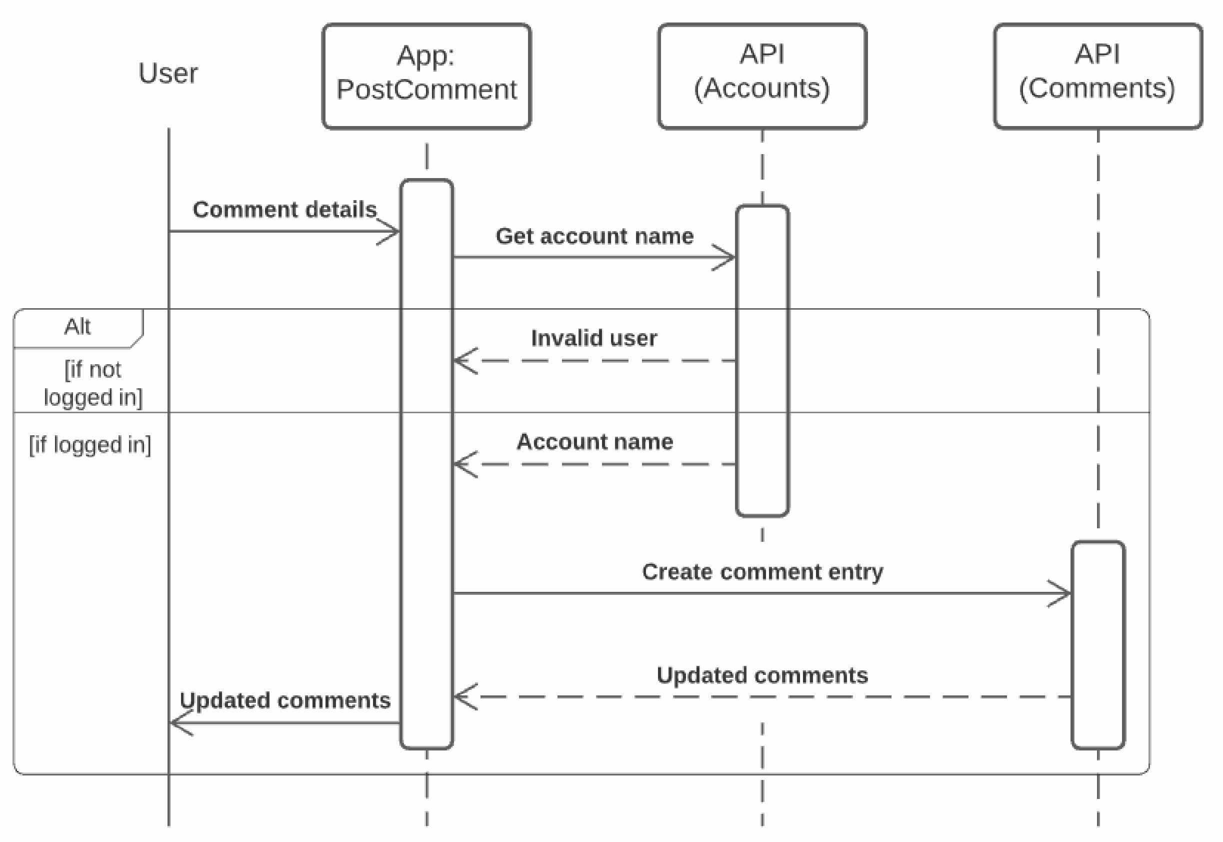
User Story 8: Add individual ingredient to Shopping List

Figure 12: Sequence Diagram 8



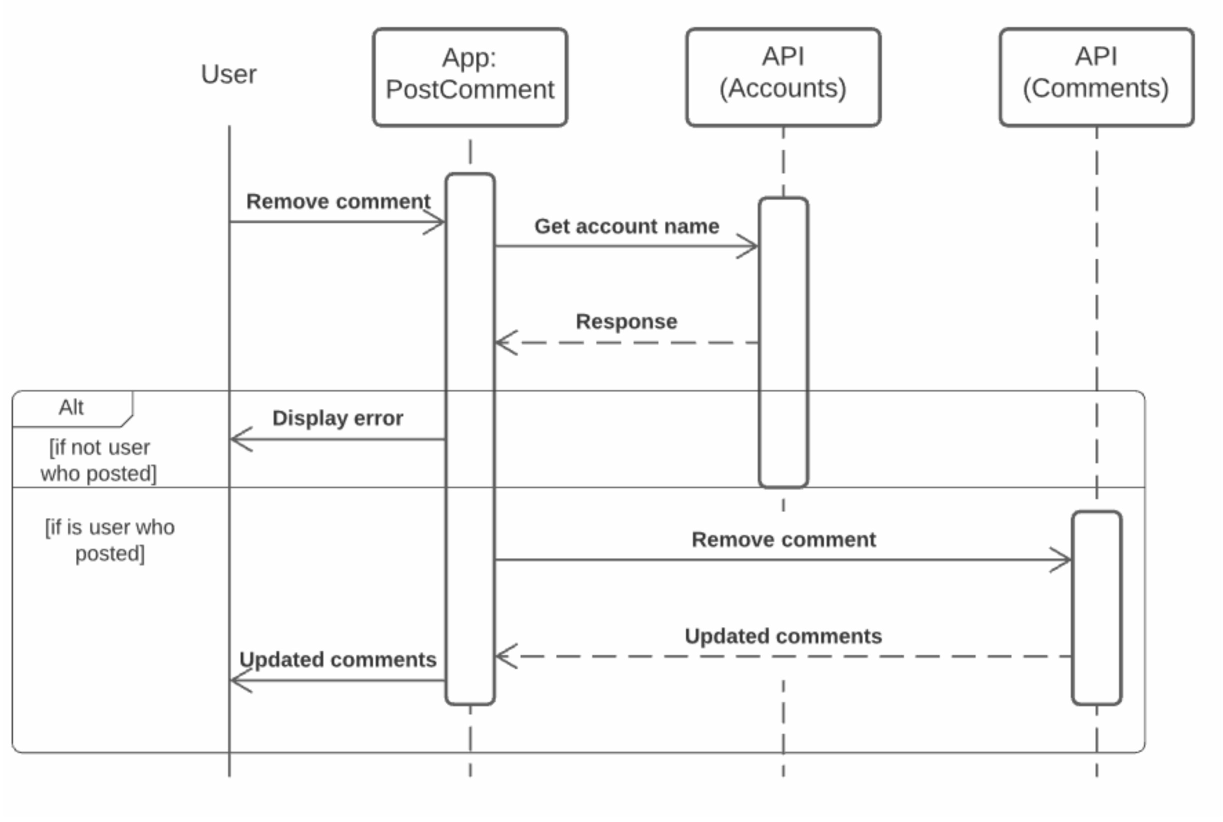
User Story 9: Commenting on Forum for Cocktails

Figure 13: Sequence Diagram 9



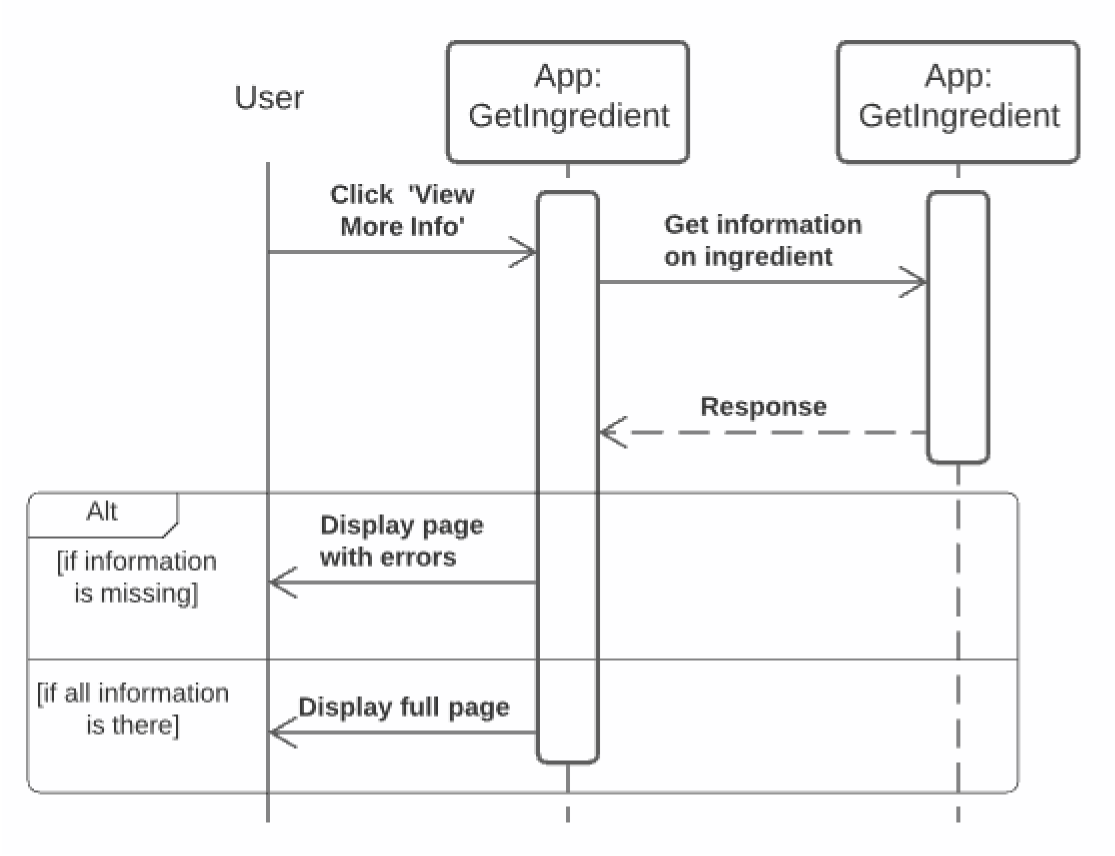
User story 10: Deleting own comments on the forum

Figure 14: Sequence Diagram 10



User Story 11: Viewing the information of ingredients of a cocktail

Figure 15: Sequence Diagram 11



Part 4: Team Organisation and Conclusion

Responsibilities and organisation of the team

In our team Nicholas was the *project owner* where they had the most knowledge and experience about the product, including the related elements such as the target audience and bartending information. With this, they had presented the majority of the requirements that were needed for our project based on their experience as a bartender. This allowed our product to be specific to the needs of the target audience.

The *scrum master* for our team was Henry, where they ensured all development and programming wasn't blocked with challenges and adversity. They had ensured all code maintained a high standard of quality, while fulfilling its purpose within the project sufficiently. Henry also took up the responsibilities and initiative to adaptively allocate roles based on what was needed and not foreseen in the planning process.

The *project manager* of the team was Aimen where the majority of communication amongst the team was facilitated and organised through them. Aimen took the responsibility to appropriately and fairly allocate roles and work per deliverable to ensure that the entire team was comfortable and excited with their duties. As project manager, this team member also ensured that everything was kept close to schedule and that all the deadlines were met. With this, they had also prioritised work in an adaptive manner where more important features and work were met with a higher urgency, ensuring the quality of the project.

The *developers* of the project were Mihail and Ben, where they had designed solutions with collaboration to ensure the project's technical challenges were overcome. All team members had contributed code and development to the project, the developers ensured all code was tested and analysed to remove unnecessary dependencies (reducing coupling), as well as improvement of features.

Reflection of project

As a team we have concluded that the project was an overall success, where each member had learnt valuable skills both technically and in a project sense. Each member collectively had shared the same vision for the product, and are satisfied with the outcome. We are proud of the web application we were able to build and how far our skills as software engineers have progressed throughout the term.

As a team we have learnt the value in planning out the development of the web application, and the ease of the steps taken to move from user stories to fully implemented features. In terms of technology, the majority of the team were using JavaScript, Node and React for the first time, and have developed significantly in building a functional frontend with this technology. Reflecting on this learning experience, it has become extremely valuable to work with these technologies and languages as they are essential with most web development work in industry. A lot of the team had persevered through the challenges of building a project from scratch completely with only online communication, and have adapted to the environment showing confidence and initiative in moving the project forward.

In general, the project was a mostly positive experience with a few minor challenges which placed obstacles in our path to completion. One of the challenges which we faced was the deadlines in relation to the time allocated for each deliverable. We felt as a team that some deliverables needed more time than others, and had wished that specifically there would have been more time for deliverable 3 than 2. However, we had adapted successfully and were able to meet the deadlines despite the scarcity for time, and are happy with the quality that we were able to produce within this time.

To conclude, our team believes the project was a success and are proud of the application that we were able to produce. Looking back if we were to have any adjustments or improvements, it would be to better time manage and plan technical development such as coding, specifically for backend data management, perhaps to have more time to implement an SQL system. However, from the new technologies that we have learnt, the project was a very enjoyable and fun experience.