Chef-AI Documentation

Team members: Guido Asbun, Adrian Vazquez, and Will Bui
GitHub Link for project: https://github.com/guidoasbun/recipe-finder-se

Vision

What exactly is our project?

- A recipe finder using the OpenAI API to make “ChatGPT” calls for three recipes using three ingredients that the user logs in
- Prints three recipes to interface using three ingredients

Goal of this project:

- Generate fast, easy, and delicious recipes using AI
- Familiarity with full stack development
- API creation and integration into the application
- Integration of AI generated data and images

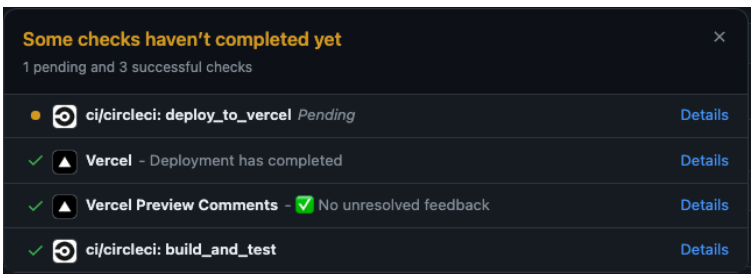
Stakeholders:

- Hungry college students and full time workers
- People who do not know what kind of recipes they can make
- You guys!

Project Management

- We are managing our project through Jira, where we implement Agile methodologies, specifically applying the use of agile epics.
- An agile epic is a large body of work that can be broken down into a number of smaller stories (called “issues” in Jira.
- These epics are delivered over a series of sprints (a total of seven sprints for this project), where sprints overlap epics, but allows for us to accomplish our sprints and epics without the issues of constantly having to add to our backlog if the epic was not completed.

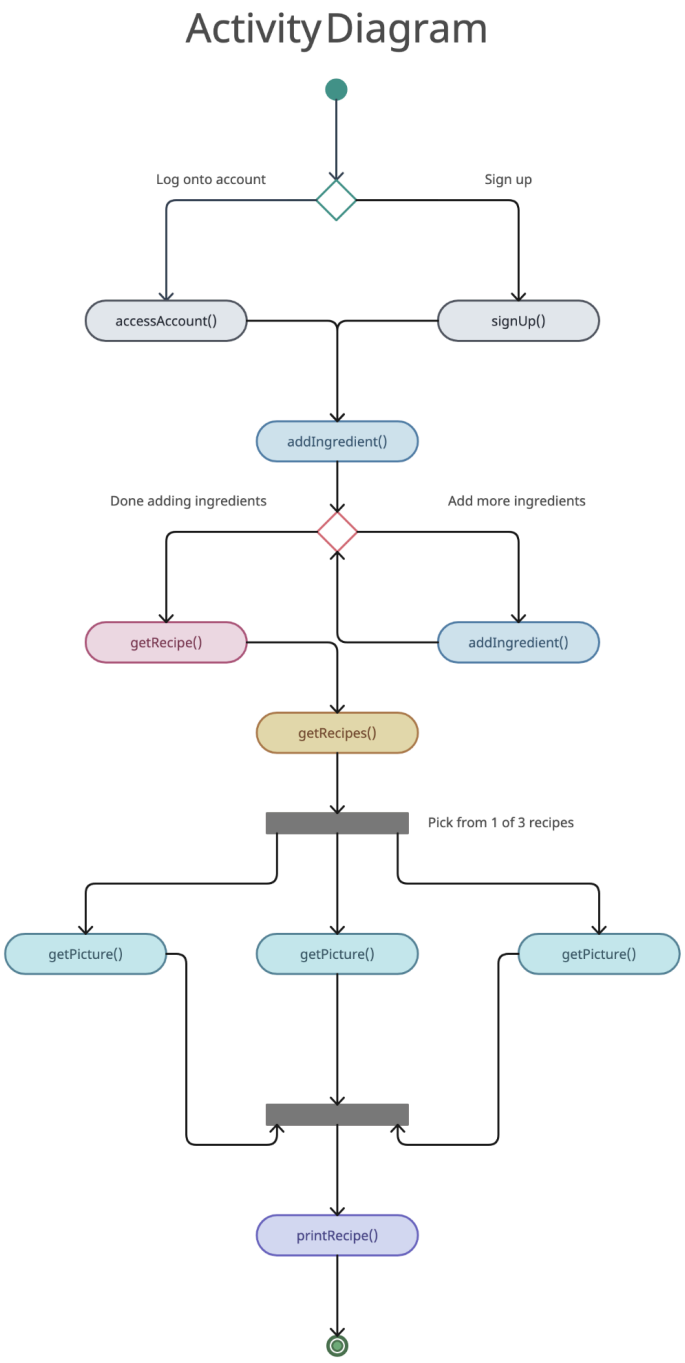
Continuous Integration and Continuous Delivery/Deployment (CI/CD)



- While using git (through GitHub) for version control, we implemented CI/CD through CircleCI to streamline each pull request, integrating both testing and deployment together.

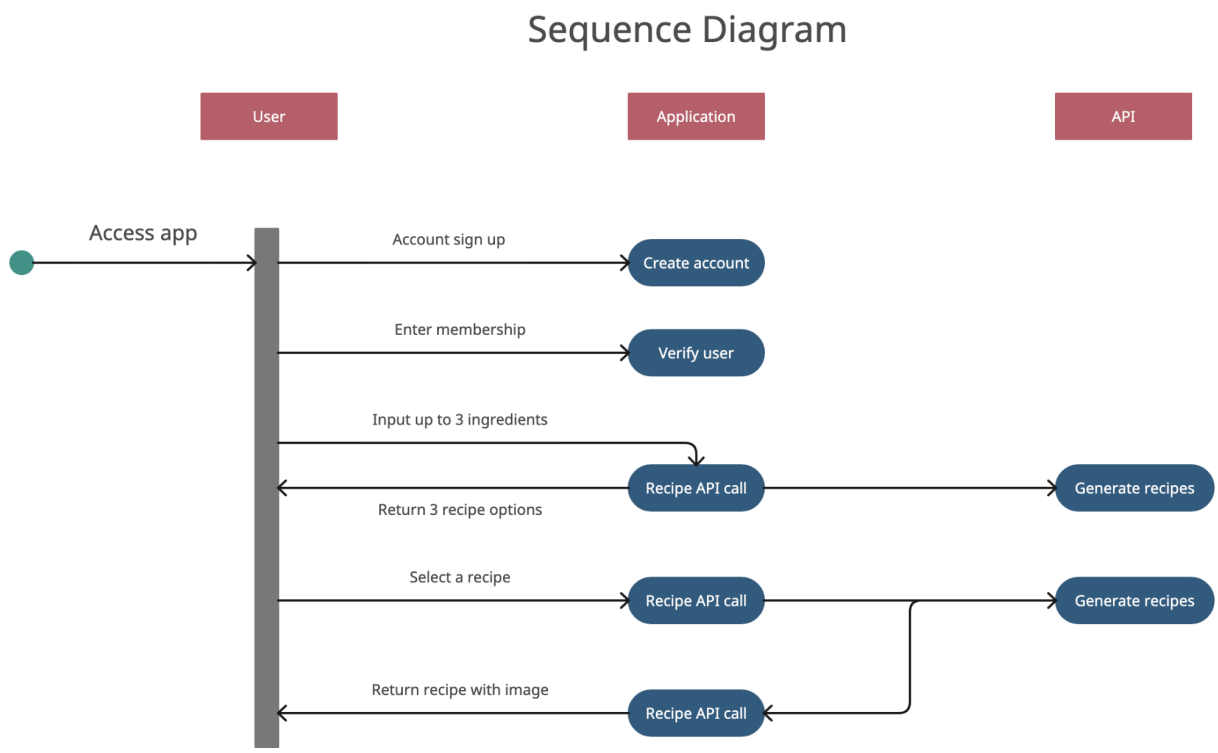
Activity Diagram

- This is a simplification of how the app should work through the use of a UML activity diagram. It is from the perspective of the user and how their decisions affect the application.



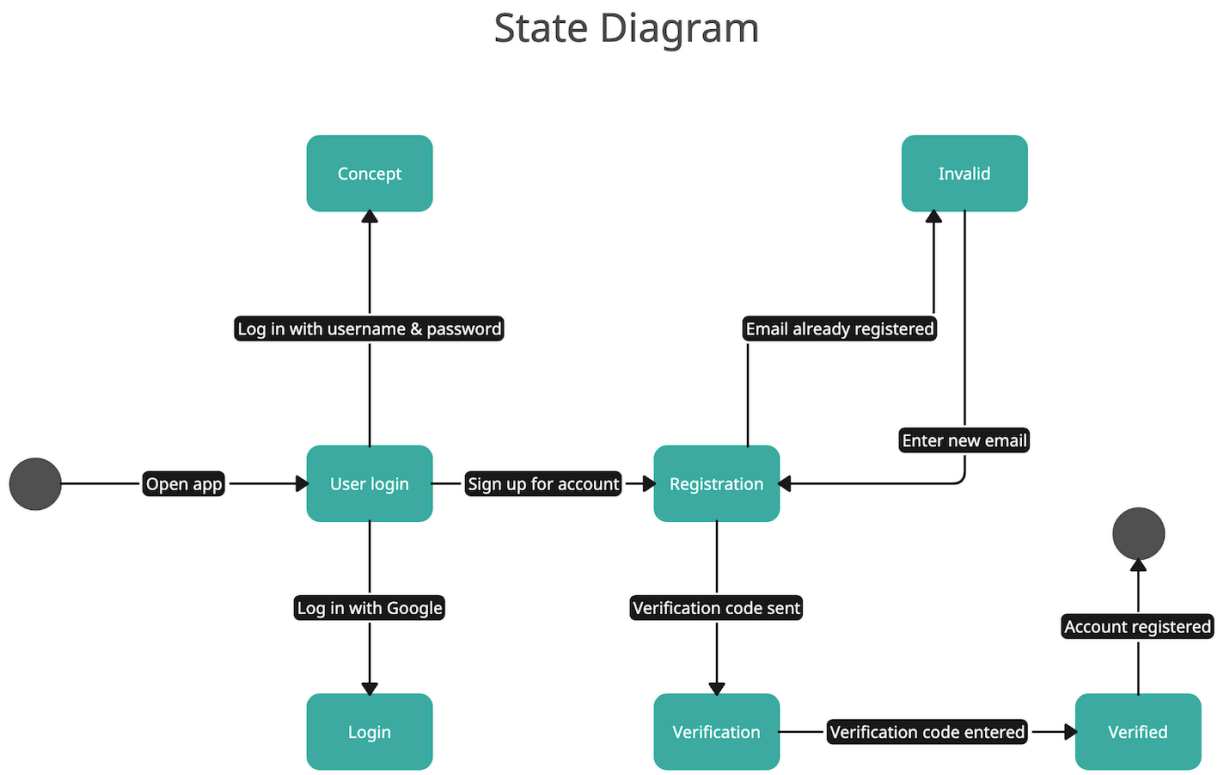
Sequence Diagram

- The sequence diagram covers how the user, application, and API interacts with each other.



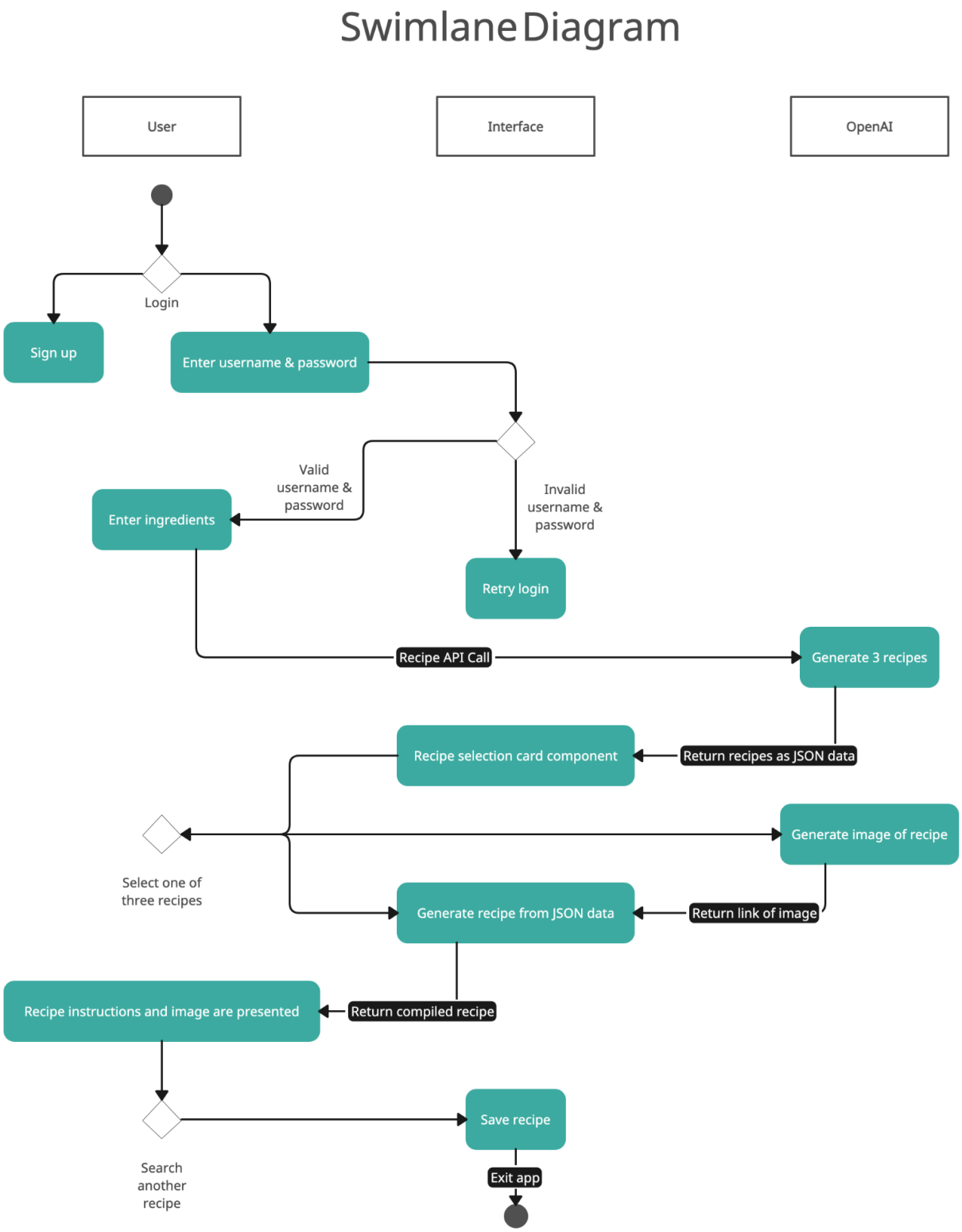
State Diagram

- This state diagram covers the login, sign up and authentication procedures



Swimlane Diagram

- This swimlane diagram breaks down the integration between the user, the interface, and OpenAI–from the start of the application to the end of the application.

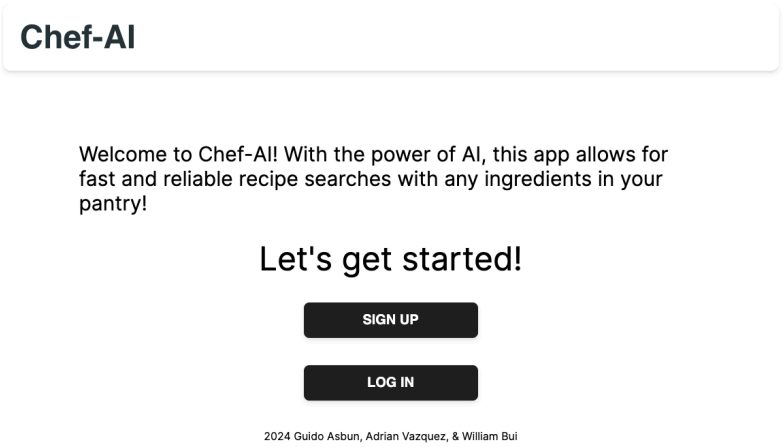


Requirements

Functional requirements

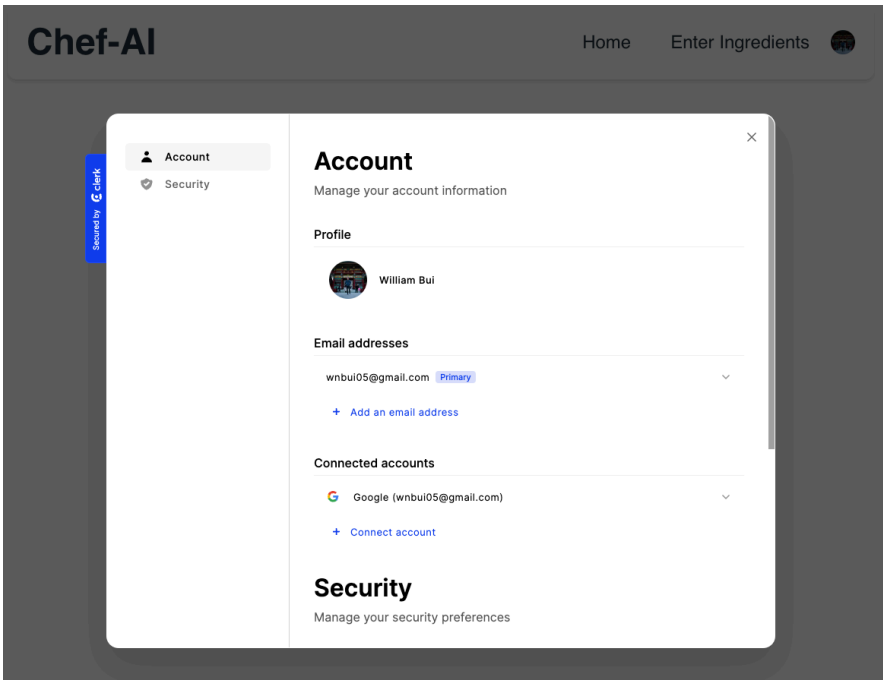
1. User account creation and authentication

   The app should allow users to create accounts and log in using credentials (email and password). There is an option to log in using Google if users do not want to create their own username and password.



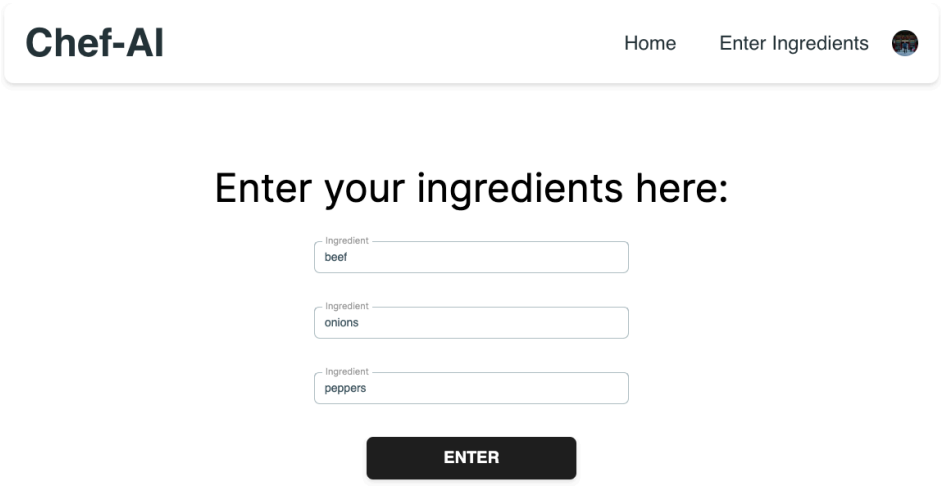
2. User account settings

   The app should allow users to edit their accounts–adding additional authorized emails and devices, change passwords, and delete their accounts if they no longer would like to use the service.



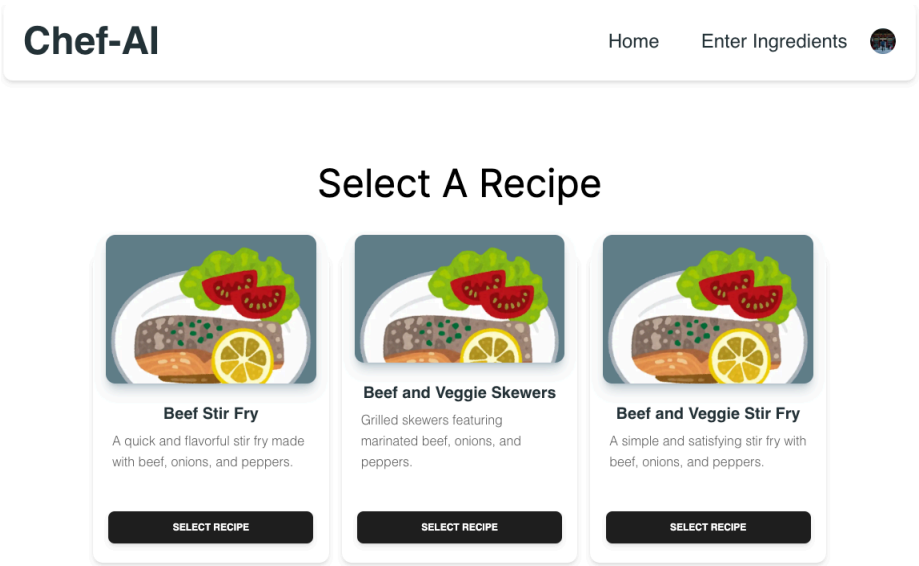
3. User inputs

The app will allow users to take in up to three ingredients to generate a recipe using OpenAI's API. The limit on the ingredients will force users to focus on the ingredients that they have on hand and the ingredients that they really want to use for the recipe. On a cost perspective, limiting to 3 inputs will control the cost of usage of OpenAI's API, where the cost will scale as more tokens (words) are prompted by the API call.



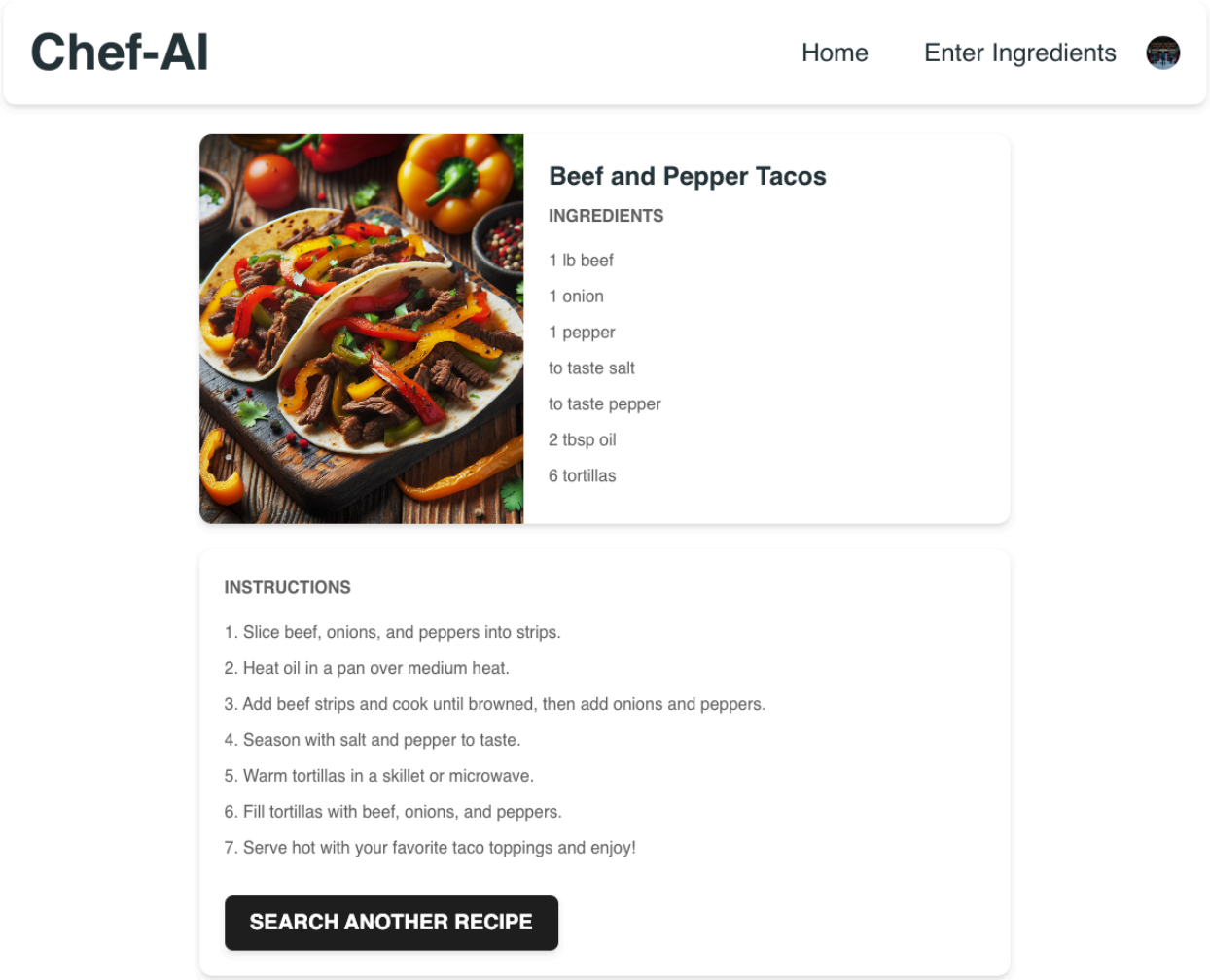
4. Recipe selection

Users can choose up to three recipes, allowing users to have options and make it feel as if the app/AI is not deciding everything for the user. Users want to be involved and have options.



5. Save recipes

   Since users have to sign up to use the service, having unique accounts allows for us to integrate in a feature that will allow users to save their favorite recipes. There are three benefits to this. First, having a database of saved recipes allows users to quickly access their previous recipe history, without having to generate a new recipe. Second, accessing old recipes cuts down on users from prompting the API call to OpenAI, which cuts down on the cost of using the API. Third, all generated recipes can be stored into a database, where future recipe searches can be cross referenced on this database first before, and if the ingredients match a previous generated recipe, that recipe will be called first, saving costs. If the ingredients do not match any recipes in the database, then the app can prompt the API call to OpenAI to generate recipes.



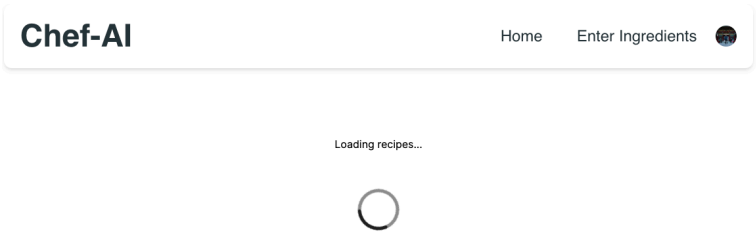
Non-functional requirements

1. Media friendly interface

   The application will be web and mobile friendly. More users are using their mobile devices or tablets now, especially in the kitchen, where there is limited space. For a majority of the time, we should assume that users are using mobile devices and tablets.

2. Security

   Security is always a concern. What should be implemented to prevent users from abusing the API calls (one to generate recipes and another to generate the associated image to the recipe). Account creation will allow us to implement several security features. First, it verifies that the user is a human user and not a bot. Second, we can implement a daily maximum on generated recipes. Eventually, there will be a paid tier that will allow for unlimited recipe generations, which would future enhance security. There are simpler implementations that could prevent abuse, such as the loading screens that we created, with built in timers, which prevents users from spam calling the API, preventing any abuse.



3. Simple and intuitive

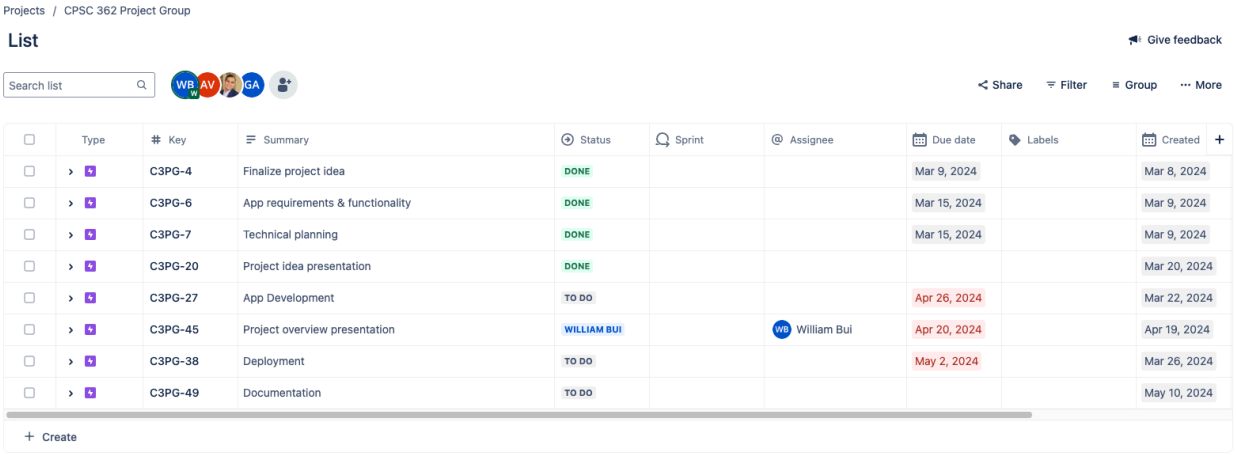
   Simplifying the application by making it easy and intuitive for the user will encourage them to use it more often. Cooking should be the main focus. The application should take in the least amount of inputs, have simple directions, obvious buttons and their placements to make it attractive and easy for users.

4. Making the application scalable

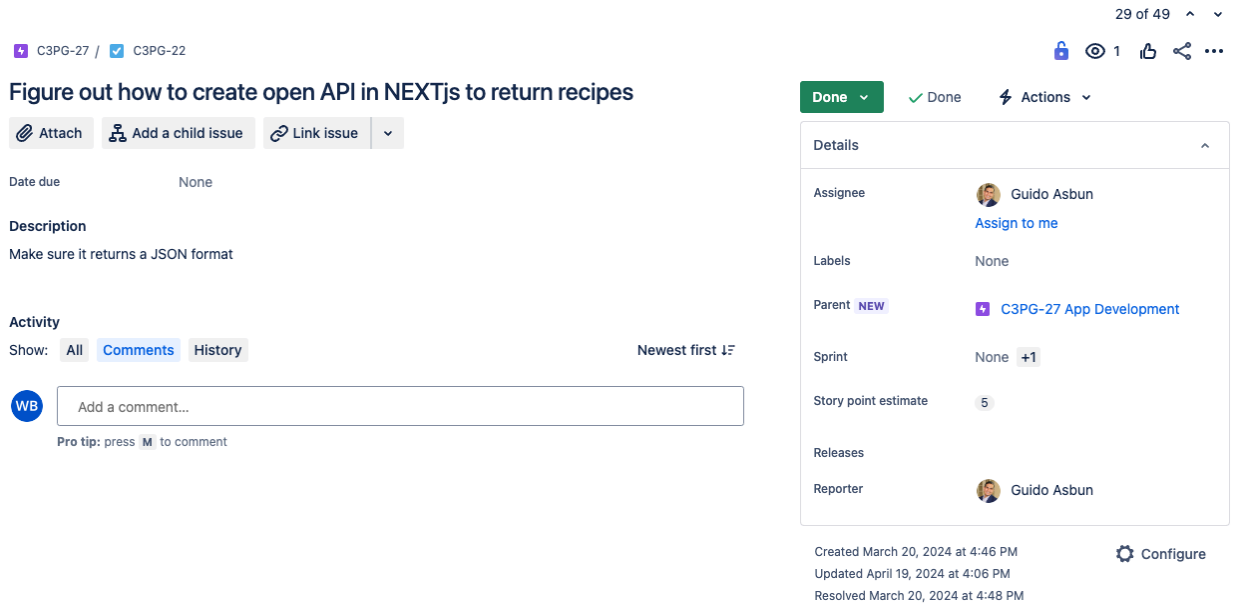
   There are several things to think about with scalability. Security is one focus, and that has been covered in another non-functional requirement. Next is the integration of user authentication, accounts, and a database to allow for the application to scale.

User Stories

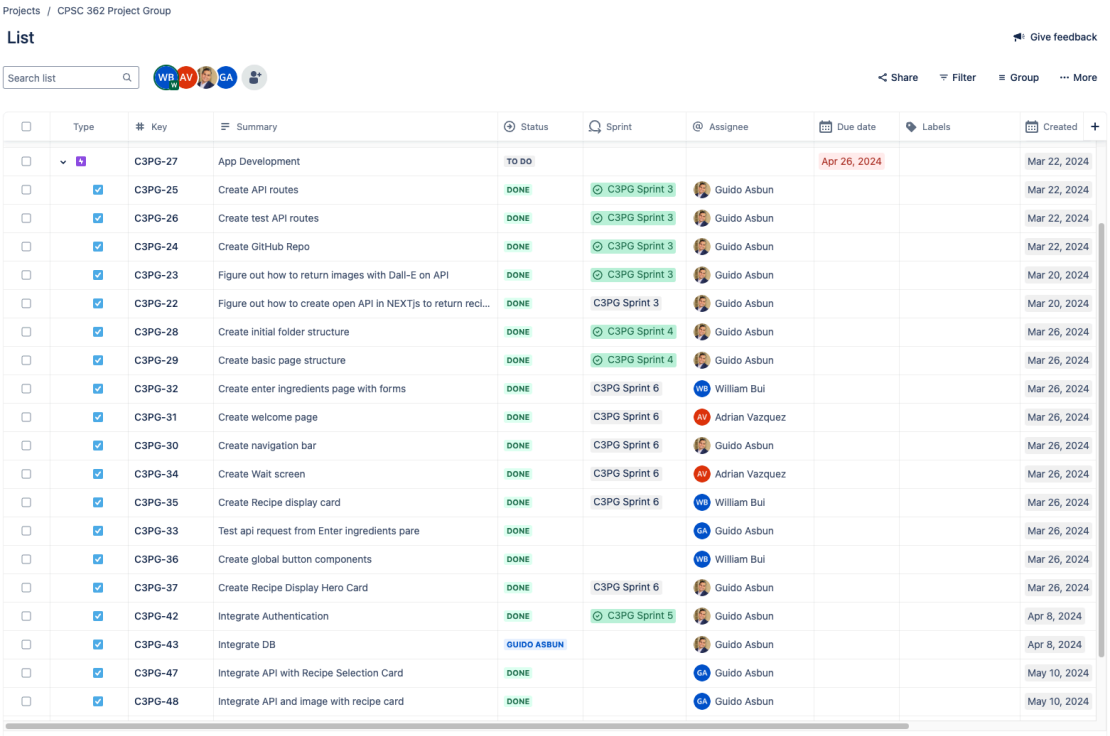
- The project is broken down into epics, which are major milestones that can encompass one to multiple sprints. During each sprint planning, user stories are created and assigned.



- Each user story is created and stored as “issues” on Jira. The issues are assigned to their appropriate sprints and developer. A story point estimate is also assigned to the issue, scored from 1-5, where 1 takes the least amount of effort and resources and 5 takes the most. These points are arbitrary and was established by our team based on the capability and needs of our team.



- List of our biggest sprint, where the bulk of our user stories (application features) were implemented. All of the requirements have been converted to user stories and listed in Jira under their respective sprints and epics.



Projects / CPSC 362 Project Group

List

Give feedback

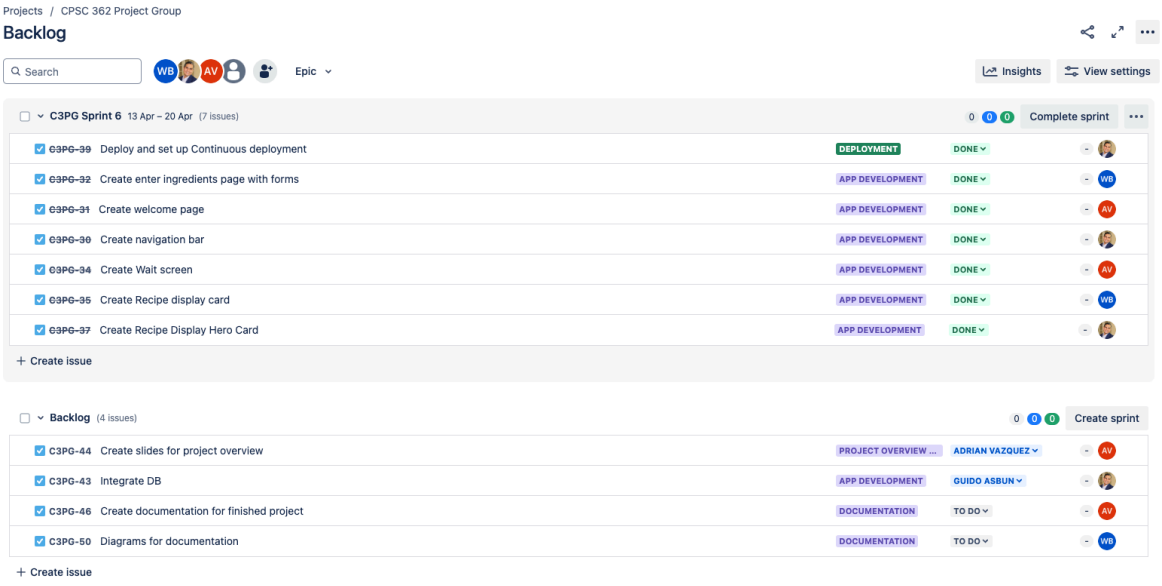
Search list

Share Filter Group More

Type	Key	Summary	Status	Sprint	Assignee	Due date	Labels	Created
	C3PG-27	App Development	TO DO			Apr 26, 2024		Mar 22, 2024
	C3PG-25	Create API routes	DONE	C3PG Sprint 3	Guido Asbun			Mar 22, 2024
	C3PG-26	Create test API routes	DONE	C3PG Sprint 3	Guido Asbun			Mar 22, 2024
	C3PG-24	Create GitHub Repo	DONE	C3PG Sprint 3	Guido Asbun			Mar 22, 2024
	C3PG-23	Figure out how to return images with Dall-E on API	DONE	C3PG Sprint 3	Guido Asbun			Mar 20, 2024
	C3PG-22	Figure out how to create open API in NEXTjs to return reci...	DONE	C3PG Sprint 3	Guido Asbun			Mar 20, 2024
	C3PG-28	Create initial folder structure	DONE	C3PG Sprint 4	Guido Asbun			Mar 26, 2024
	C3PG-29	Create basic page structure	DONE	C3PG Sprint 4	Guido Asbun			Mar 26, 2024
	C3PG-32	Create enter ingredients page with forms	DONE	C3PG Sprint 6	William Bui			Mar 26, 2024
	C3PG-31	Create welcome page	DONE	C3PG Sprint 6	Adrian Vazquez			Mar 26, 2024
	C3PG-30	Create navigation bar	DONE	C3PG Sprint 6	Guido Asbun			Mar 26, 2024
	C3PG-34	Create Wait screen	DONE	C3PG Sprint 6	Adrian Vazquez			Mar 26, 2024
	C3PG-35	Create Recipe display card	DONE	C3PG Sprint 6	William Bui			Mar 26, 2024
	C3PG-33	Test api request from Enter ingredients pare	DONE		Guido Asbun			Mar 26, 2024
	C3PG-36	Create global button components	DONE		William Bui			Mar 26, 2024
	C3PG-37	Create Recipe Display Hero Card	DONE	C3PG Sprint 6	Guido Asbun			Mar 26, 2024
	C3PG-42	Integrate Authentication	DONE	C3PG Sprint 5	Guido Asbun			Apr 8, 2024
	C3PG-43	Integrate DB	GUIDO ASBUN		Guido Asbun			Apr 8, 2024
	C3PG-47	Integrate API with Recipe Selection Card	DONE		Guido Asbun			May 10, 2024
	C3PG-48	Integrate API and image with recipe card	DONE		Guido Asbun			May 10, 2024

Product Backlog

- We are tracking the backlog through Jira
- Since the estimates have already been established in the user stories, it was easy for us to decide on how to manage our backlog–what to move to the backlog and what issues needed to be accomplished over other issues.



Projects / CPSC 362 Project Group

Backlog

Search Epic Insights View settings

C3PG Sprint 6 13 Apr – 20 Apr (7 issues) 0 0 0 Complete sprint

~~C3PG-39~~	Deploy and set up Continuous deployment	DEPLOYMENT	DONE
~~C3PG-32~~	Create enter ingredients page with forms	APP DEVELOPMENT	DONE
~~C3PG-31~~	Create welcome page	APP DEVELOPMENT	DONE
~~C3PG-30~~	Create navigation bar	APP DEVELOPMENT	DONE
~~C3PG-34~~	Create Wait screen	APP DEVELOPMENT	DONE
~~C3PG-35~~	Create Recipe display card	APP DEVELOPMENT	DONE
~~C3PG-37~~	Create Recipe Display Hero Card	APP DEVELOPMENT	DONE

+ Create issue

Backlog (4 issues) 0 0 0 Create sprint

C3PG-44	Create slides for project overview	PROJECT OVERVIEW ...	ADRIAN VAZQUEZ
C3PG-43	Integrate DB	APP DEVELOPMENT	GUIDO ASBUN
C3PG-46	Create documentation for finished project	DOCUMENTATION	TO DO
C3PG-50	Diagrams for documentation	DOCUMENTATION	TO DO

+ Create issue

Lessons Learned

Guido:

- When developing a software application or project, one of the most important components is the design and the planning.
- With proper design and planning, the coding part becomes the easiest part of the application or project.
- It is important to keep detailed documentation for communication and reference purposes. And as a way to see the progress of the project and have a more accurate perspective of the deliverables.
- Apart from the software engineering focus of this project, I learned several technologies that I had an interest in learning. For example, I learned about gateway timeouts in Vercel, I learned Authentication for routes, how to easily implement the useContext hook in react, and many more technical processes.

Will:

- Planning is the bulk of the work when it comes to software engineering. It is more important to understand the problem and the solution than it is to code out the features.
- When it comes to development work on a team, understanding the codebase and changes made by your peers is extremely important to how it affects your work and the changes that you plan to implement. It also helped me understand how everything worked underneath the hood.
- Using premade components and customizing them to your needs allowed for me to save time and allowed for me to focus on the implementation of the user stories and features.
- Learning React through Next.js.
- When you are blocked, reach out to your team for help.

Adrian:

- How to properly communicate and conduct work under stress
- You always want to present and propose ideas immediately when they come to mind
- DO NOT be afraid to ask questions or for help along the way
- Communication > Development
- Don’t be afraid to learn something new everyday
- I’d like to dig deeper on linking frontend to an API/backend and get more comfortable with full-stack development and front-end design