Software Requirements Specification

(SRS) Document

NOM NOM

4/25/2025

[Version 1]

By: Sara Abukhalaf, Amna Sohail,

Fady Eskandr, Bunroung Heak, Carson Bernard

Github project link :

https://github.com/nomnom-app-uncg/senior_capstone

Table of Contents

1. Introduction

1.1 Purpose Nom Nom is a one-stop platform for food enthusiasts, home cooks, and culinary explorers. It provides recipe discovery, pantry tracking, AI-assisted cooking help, and curated culinary content all in one app.

1.2 Document Conventions This Software Requirements Specification (SRS) document provides a detailed roadmap of the Nom Nom application's features, architecture, and technical requirements. It is structured to assist developers, project managers, testers, and academic reviewers in understanding every part of the project.

The key features of Nom Nom highlighted in this document include:

- Virtual Fridge: This feature allows users to manage a digital inventory of their real-life ingredients. It helps users keep track of what they have at home and even alerts them when they're running low on specific items.
- Picture to Recipe: Users can upload a photo of a dish, and using AI, the app attempts to identify the dish and return a list of possible recipes. This enhances food discovery and helps users recreate meals they've seen or eaten elsewhere.
- Recipe Finder: This tool lets users input any combination of ingredients, and the system uses AI to return compatible recipe ideas. It's designed to reduce food waste and make meal planning faster and smarter

1.3 Definitions, Acronyms, and Abbreviations

- **AI**: Artificial Intelligence
- **API**: Application Programming Interface
- **JWT**: JSON Web Token
- **UI/UX**: User Interface / User Experience
- **Expo**: A framework and platform for universal React applications

1.4 Intended Audience This Software Requirements Specification (SRS) is intended to serve multiple stakeholders involved in the development, deployment, and review of the Nom Nom application. Its primary audience includes developers who will use this document as a technical blueprint for building the frontend and backend systems, ensuring that each feature—such as the virtual fridge, AI-based recipe generation, and picture-to-recipe functionalities—is implemented according to defined requirements. Additionally, project managers and coordinators will rely on this document to assess progress, allocate resources, and evaluate compliance with deadlines and scope. Quality assurance testers will use it to create test cases and validate that both functional and non-functional requirements are being met. The academic reviewers or instructors overseeing this senior capstone project will also find this document valuable in evaluating the team's understanding of software engineering principles, project planning, and execution.

Finally, this document is also written with future developers or maintainers in mind, as it outlines system architecture, expected behavior, and expansion plans that can guide continued development or feature rollouts post-deployment. By addressing each group's needs, this document acts as a central source of truth for the project lifecycle.

1.5 Project Scope

Nom Nom serves as a smart digital kitchen companion that merges everyday cooking needs with cutting-edge AI-driven features. The scope of the project includes delivering a cross-platform application—accessible on both web and mobile—that helps users manage their kitchen inventory, discover new meals, and develop more sustainable cooking habits. One of the primary features is the **Virtual Fridge**, which allows users to track their real-life ingredients, thereby reducing food waste and encouraging meal planning around what's already available at home.

Another innovative feature is **Picture to Recipe**, where users upload an image of a dish and receive suggested recipes that resemble the meal. This leverages AI to help users recreate meals they've enjoyed elsewhere or identify food they're curious about. In addition, the **Recipe Finder** lets users input any combination of ingredients they have on hand, and the system returns compatible recipe ideas using GPT-generated responses. This not only saves time but also adds a level of personalization and creativity to the cooking process.

The project scope also includes an explore page, where users can browse food articles creating a community-driven and educational experience within the app. All of these features are built to support user engagement through clean UI/UX design, AI integration, and personalized functionality. The goal is to develop an intuitive and intelligent tool that transforms how users interact with food-related content and their own kitchens. As a senior capstone project, this scope also reflects a full software development lifecycle—from planning and architecture to testing and deployment.

1.6 Technology Challenges

- Integration of OpenAI's GPT model with user-submitted ingredients
- Synchronization between local storage and remote backend
- Ensuring a seamless user experience across different screen sizes and operating systems

1.7 References

- React Native & Expo documentation
- OpenAI API documentation
- Axios documentation
- TypeScript documentation

2. General Description

2.1 Product Features

User Authentication: Users can securely register and log in using their email and password. This feature ensures personalized access to saved recipes, ingredient tracking, and AI interactions. It also supports session management to keep users logged in across visits.

Virtual Fridge: This feature acts as a digital pantry where users can add or remove ingredients they have at home. It provides a simple interface for tracking available items and plays a central role in generating personalized recipe suggestions. The virtual fridge can help reduce food waste by encouraging users to cook with what they already have.

AI-Powered Recipe Generation: Using OpenAI's GPT technology, this feature enables users to input a list of ingredients and receive creative and relevant recipe suggestions. It parses user inputs and delivers recipe titles, steps, and even images. This is one of the most intelligent features in the app, turning basic ingredient lists into cooking inspiration.

Save Recipes: After recipes are generated by AI, users can choose to save their favorites for future use. Saved recipes are tied to the user's account and can be accessed anytime. This feature allows users to build their own digital cookbook of meals they've tried or want to revisit.

Browse Culinary Articles: This content hub presents users with curated articles, trending dishes, nutrition tips, and more. The goal is to offer not just utility, but also an educational and entertaining experience that keeps users engaged with the app beyond just cooking.

Picture to Recipe: This feature is designed to allow users to upload an image of a dish and receive AI-generated guesses of what it is, alongside suggested recipes. This makes food discovery more visual and spontaneous, enhancing the user experience even further.

2.2 User Class and Characteristics

- Anonymous Users: Can browse the app but cannot save recipes.
- Registered Users: Full access to ingredient management, saving recipes, and personalized content.

2.3 Operating Environment

The Nom Nom application is built to operate across multiple platforms, ensuring broad accessibility and user reach. It is developed using **React Native** with **Expo**, allowing for smooth deployment on both Android and iOS devices. Users can install the app via app stores or run it through Expo Go for testing and development. On desktop and laptop environments, the application is accessible through major modern browsers such as Chrome, Firefox, Safari, and

Edge. The backend relies on **Node.js** and **Express**, interacting with frontend components via RESTful APIs. These APIs handle authentication, recipe saving, and communication with external services like OpenAI for AI-powered features.

2.4 Constraints

The development and deployment of Nom Nom are subject to several constraints. A major limitation is the dependency on external APIs, particularly the OpenAI API for generating AI-based recipe suggestions. This introduces constraints related to usage limits, service costs, and potential downtimes. The app also needs to handle sensitive user data securely, necessitating secure storage and transmission using JWT tokens and HTTPS protocols. Platform-specific constraints may also impact performance—users on older devices or with limited bandwidth may experience slower load times or reduced functionality. Cross-platform consistency must be maintained, requiring extensive testing and UI adjustments to ensure the app behaves uniformly across Android, iOS, and web interfaces. Finally, as a student-developed capstone project, team bandwidth and timeline constraints also influence the scope and complexity of what can be built and maintained.

2.5 Assumptions and Dependencies

- APIs for recipes and chat functions are accessible
- Backend is deployed and functional

3. Functional Requirements

3.1 Primary

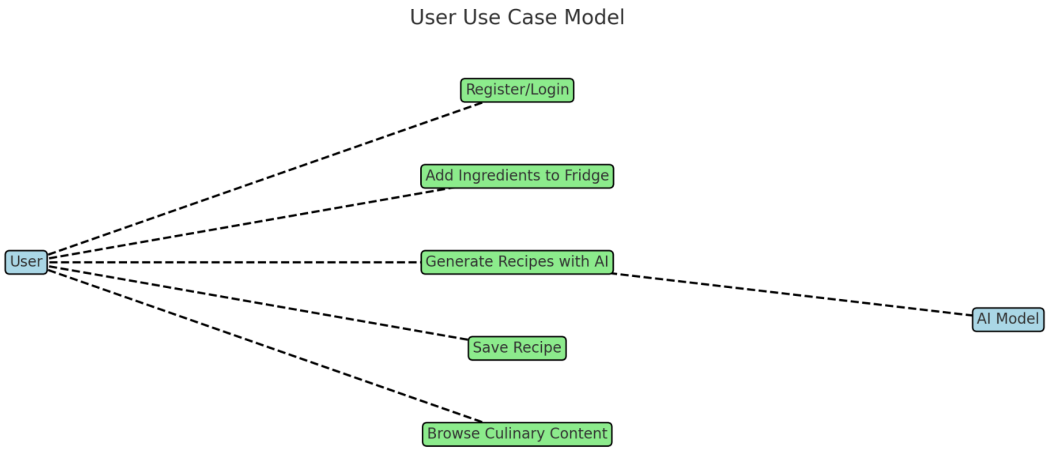
- Users can register/login/logout
- Users can search and view recipes
- Users can manage pantry ingredients
- Users can chat with the cooking assistant

3.2 Secondary

- Users can browse trending culinary articles
- Users can view saved ingredient lists

3.3 Use-Case Model

3.3.1 Use-Case Model Diagram



3.3.2 Use-Case Model Descriptions

3.3.2.1 Actor: User

Description:
The general User is the primary actor of the system. This includes anyone who visits the website or opens the app to interact with recipes, the fridge feature, or the AI assistant.

Associated Use Cases:

Register/Login: The user creates an account or logs into an existing one to unlock features like saving recipes.

Add Ingredients to Fridge: The user enters ingredients they have on hand to build their virtual pantry.

Generate Recipes with AI: The user submits their fridge ingredients to get smart recipe suggestions.

Save Recipe: The user can tap a heart icon to save generated recipes for later use.

Browse Culinary Content: The user can scroll through articles, food tips, or trending dishes in the hub section.

3.3.2.3 Actor: AI Model

Description:
The AI Model is an external actor that handles logic and content generation based on user input.

Associated Use Cases:

Generate Recipes with AI: When a user submits a list of ingredients, the AI model returns formatted recipes with titles, descriptions, and images.

3.3.3 Use-Case Model Scenarios

3.3.3.1 Actor: User

Use-Case Name: Register/Login
Initial Assumption: The user has opened the Nom Nom app or website and is presented with the login or registration option.
Normal Scenario: The user enters their email and password to log in, or fills out a registration form to create a new account.
What Can Go Wrong: The user enters incorrect credentials or the server is unreachable.
System State on Completion: If successful, the user is authenticated and granted access to personalized features.

Use-Case Name: Add Ingredients to Fridge
Initial Assumption: The user is logged in and navigating the Fridge screen.
Normal Scenario: The user types in ingredients one by one, adding them to their virtual inventory.
What Can Go Wrong: A duplicate entry is made or input is left blank. The system alerts the user appropriately.
System State on Completion: The ingredient list is updated and stored locally (and optionally synced to the user's profile).

Use-Case Name: Generate Recipes with AI
Initial Assumption: The user has input a list of ingredients and is ready to receive suggestions.
Normal Scenario: Upon clicking "Find Recipes," the system sends the data to the AI model, which returns formatted suggestions.
What Can Go Wrong: The AI returns no valid recipes or the service times out.
System State on Completion: Recipes are displayed to the user, complete with names, ingredients, and preview images.

Use-Case Name: Save Recipe
Initial Assumption: The user is logged in and viewing a recipe returned by the AI.
Normal Scenario: The user taps a heart icon or similar element to save the recipe to their

account.
What Can Go Wrong: User is not logged in, or there is a network error.
System State on Completion: The recipe is added to the user's saved list, and the UI reflects the saved state.

Use-Case Name: Browse Culinary Content
Initial Assumption: The user has access to the Culinary Hub tab.
Normal Scenario: The user scrolls through food-related articles, visual content, or featured tips curated by the team.
What Can Go Wrong: The API fails to load content or a media element does not display.
System State on Completion: Content loads and is available for interaction, increasing user engagement and app stickiness.

4. Technical Requirements

4.1.1 User Interfaces

Nom Nom provides a clean, modern, and user-friendly interface across both mobile and web platforms. The front-end is developed using React Native for the mobile application and React for the web interface, ensuring consistency in design and responsiveness across devices.

- **Mobile Application (React Native):** The mobile UI is designed using React Native and styled with reusable components. It includes navigation tabs, modal popups for recipe previews, and an interactive ingredient input section. The layout is optimized for both Android and iOS devices, leveraging native performance and animations through the Expo framework.

- **Web Application (React):** The web version mirrors the mobile layout but is optimized for larger screens. It offers the same functionality with responsive layouts and component-based rendering. It allows users to register, log in, add ingredients, and receive recipe suggestions just like on mobile.

Both interfaces prioritize intuitive navigation, minimal input requirements, and visually appealing recipe card displays with image previews and save options. Colors and gradients are chosen to evoke a fresh, culinary-inspired aesthetic.

4.1.2 Hardware Interfaces

Nom Nom is designed to run on a wide range of devices with internet access and modern browsers. The application does not require any specific hardware modules but must support basic display, touch, and keyboard input capabilities.

- **Mobile Devices:** The app supports smartphones and tablets running Android or iOS with React Native and Expo compatibility.

- **Desktop/Laptop Browsers:** The web app runs in major modern browsers such as Chrome, Safari, Firefox, and Edge, on both Windows and macOS platforms.

No additional peripheral hardware is required for the system to function.

4.1.4 Software Interfaces

Nom Nom is built using a modular and scalable tech stack. Each software component integrates with others to form a responsive, AI-powered application.

- **React Native:** Used to build the mobile version of the application with a native look and feel for both iOS and Android platforms.

- **React (Web):** Enables the creation of a dynamic and responsive front-end for browser users.

- **TypeScript:** A typed superset of JavaScript, used across both mobile and web codebases to improve reliability, maintainability, and developer productivity.

- **Axios:** Handles all HTTP requests from the client side, enabling communication with the backend API for user login, recipe generation, and saving data.

- **Expo:** A development framework and platform for React Native that accelerates deployment and testing across mobile devices.

- **OpenAI (GPT):** Used via a service module to interpret user-provided ingredients and return AI-generated recipes. It enhances user engagement by providing dynamic and personalized suggestions.

- **Backend (Node.js):** Manages user authentication, stores saved recipes, handles fridge data, and integrates with external services like GPT. The backend exposes RESTful API endpoints that are consumed by the client applications.

5. Non-Functional Requirements

5.1 Performance Requirements

Nom Nom is designed to perform smoothly across both mobile and web platforms. Page transitions, user actions (such as saving recipes or adding ingredients), and recipe generation should occur with minimal lag to ensure a seamless user experience. All AI-generated content should return within 5 seconds of the request being made. The application should consume no more than 200MB of memory on client devices and under 100MB for local storage tasks such as managing saved ingredients and cached user data. The user should be able to create a profile, add ingredients, and receive recipe suggestions in under 5 minutes total for first-time use.

5.2 Safety Requirements

User data—including saved recipes, login credentials, and ingredient lists—will be securely handled using token-based authentication and encrypted storage. The system will implement proper error handling to avoid exposing sensitive information in the event of crashes or API failures.

5.3 Security Requirements

All user interactions will be conducted over secure HTTPS connections. Users must authenticate through a login system protected by encrypted credentials and secure tokens. Only authenticated users will be able to save recipes or access personalized data.

5.4 Software Quality Attributes

- Availability: The application should be accessible 99.9% of the time during normal operational hours.

- Correctness: All returned recipes should match the ingredients submitted or the query entered by the user. AI content will be checked for formatting accuracy and clarity.

- Maintainability: The project follows modular code design principles with well-documented functions and clearly separated logic, making future updates and debugging efficient.

- Reusability: Components such as the recipe card, modal interface, and chat functionality are written as reusable modules to be used across multiple screens.

- Portability: The app will run in web browsers and mobile devices using the React Native framework via Expo.

5.5 Process Requirements

- Time Constraints: Development began in early February and is expected to be completed by late April, with a project demo scheduled for the end of April.

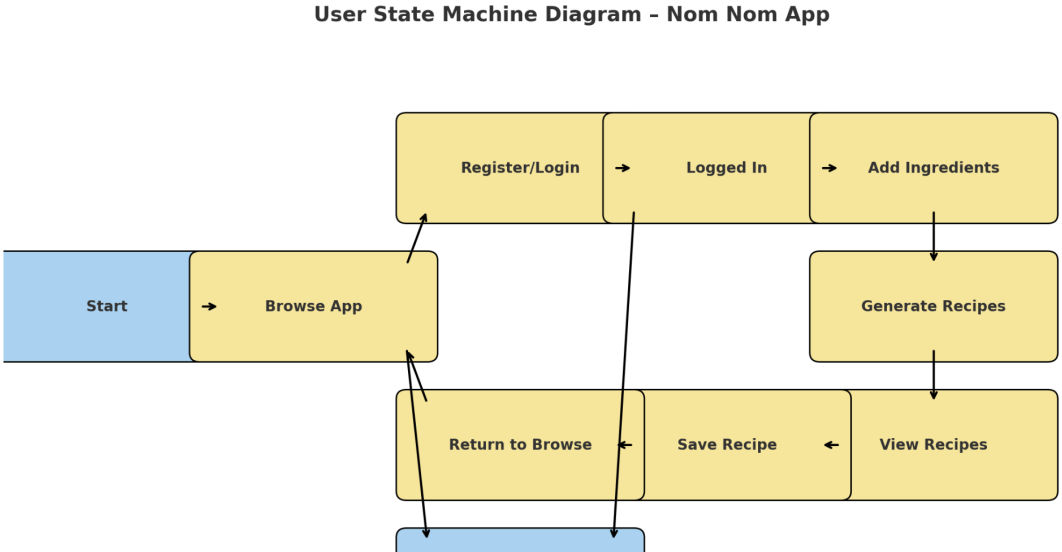
- Cost and Delivery Date: Since this is a student capstone project, there is no formal budget. All development is conducted using open-source tools. The expected delivery date is April 25, 2025.

6. Design Documents

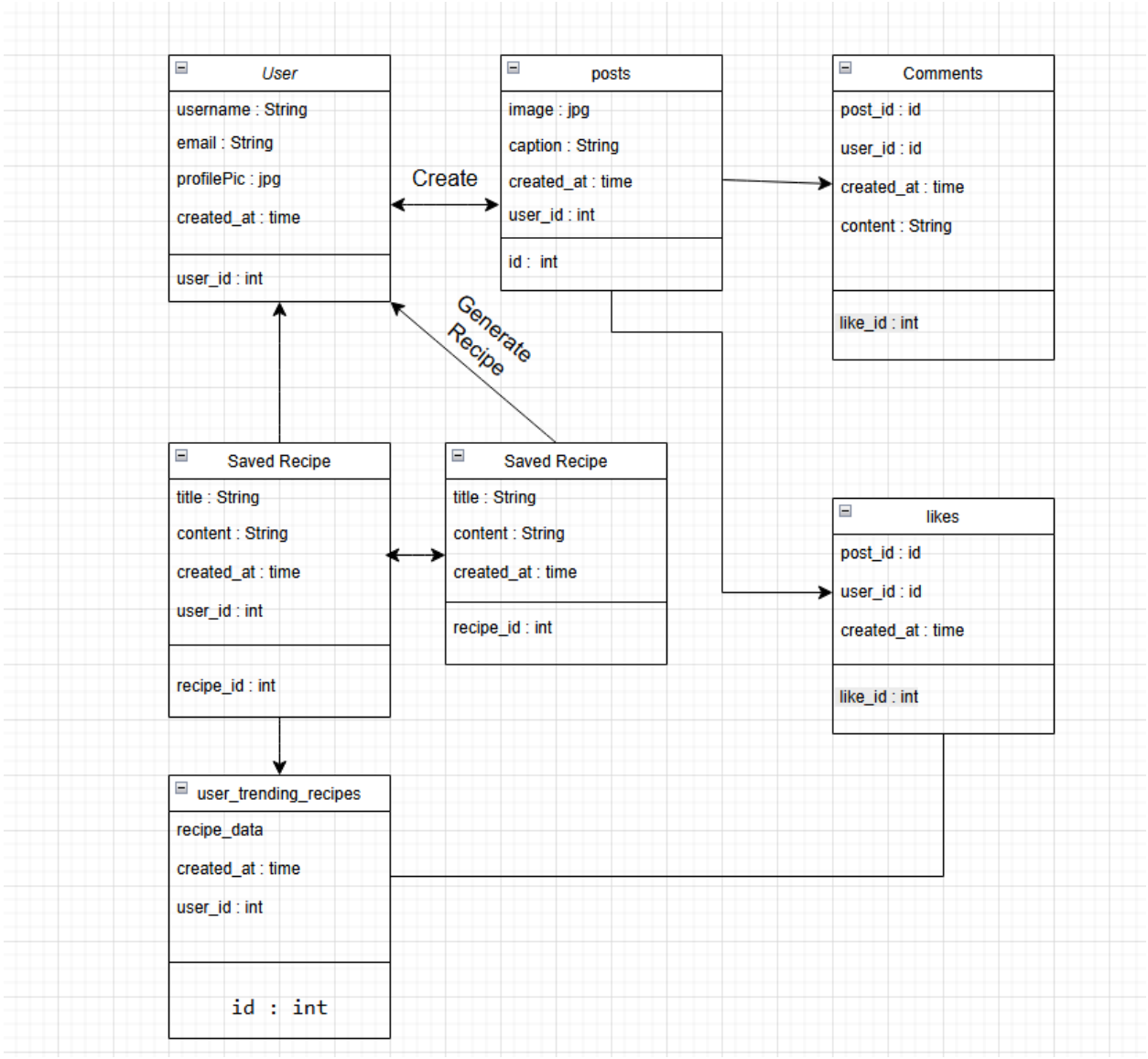
6.1 Software Architecture

- Frontend communicates with backend APIs
- Separate routes for login, chat, fridge, etc.

6.2 State Machine Diagram: User



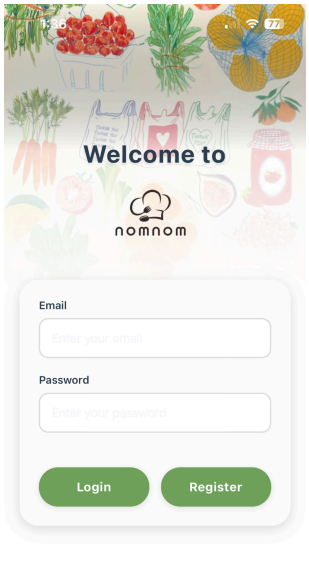
6.3 UML Class Diagram

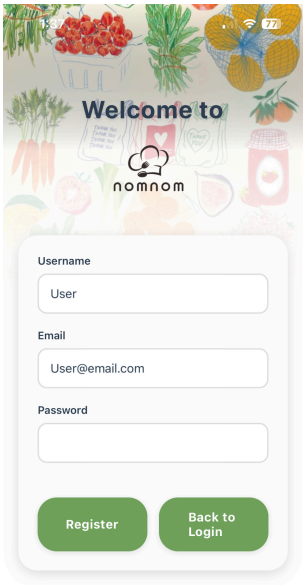


7. Scenario

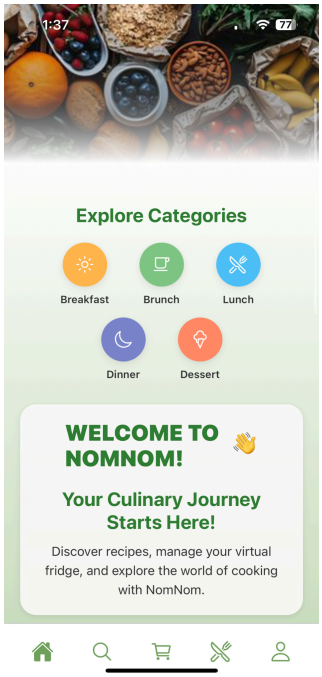
7.1 Brief Written Scenario with Screenshots

Registering and Signing In:

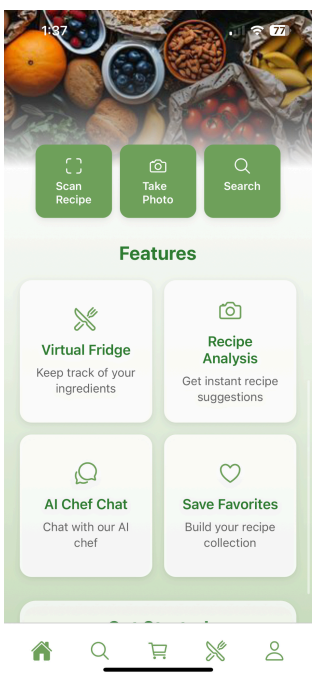


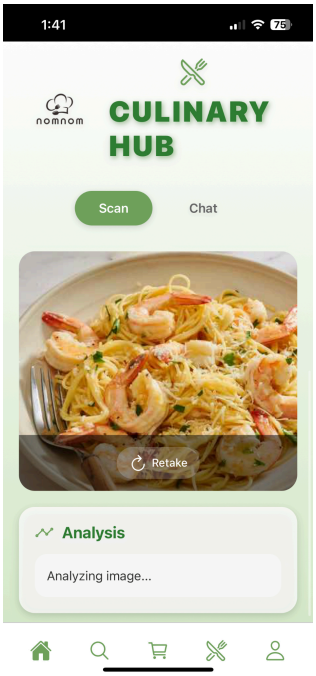


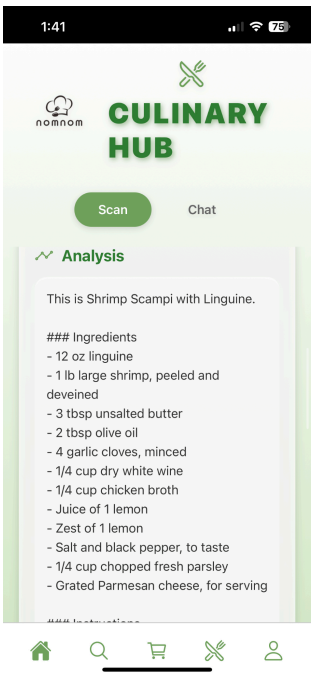
Entering Home Screen:



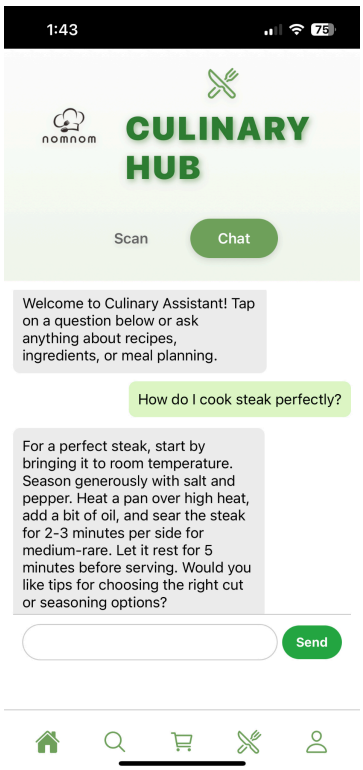
From Home Screen Going to Scan Picture to Recipe:



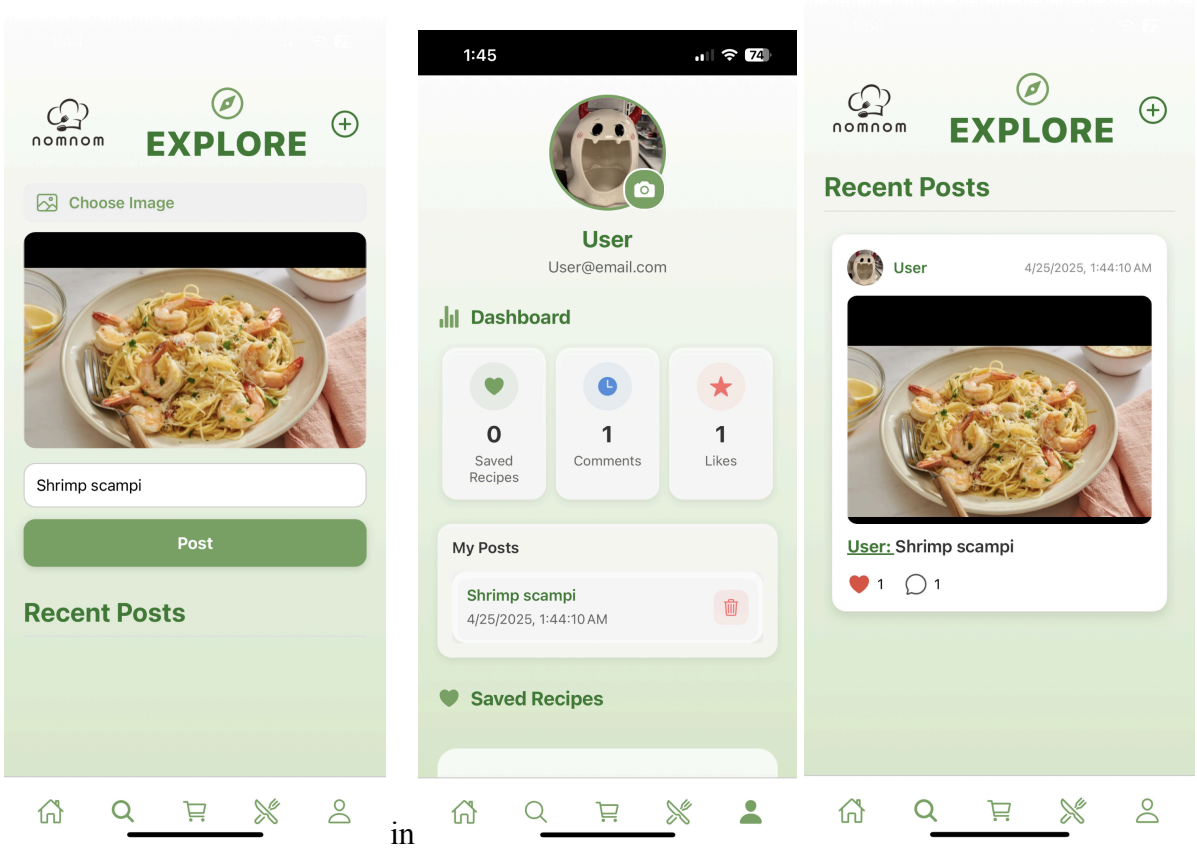




Chat box assistance:



Making a post and going to my profile to change image:



in

Generating Recipes with Ingredients Input: