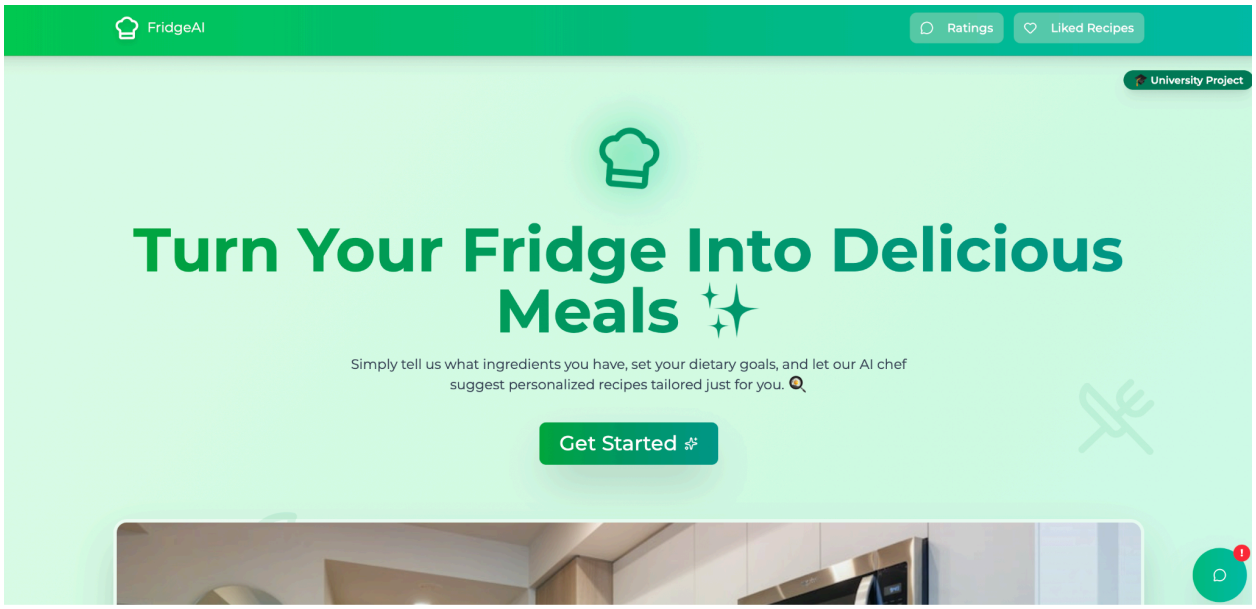
Developed by Olga Khan
Introduction to Software Engineering

Final Exam. University Project

December 23, 2025

FridgeAI: Your Smart Meal Planner



Introduction

In today's fast-paced lifestyle, many individuals struggle with meal planning despite having sufficient food ingredients at home. This often leads to repetitive meals, lack of creativity in cooking, and unnecessary food waste. At the same time, people increasingly follow specific dietary preferences and health goals such as veganism, weight loss, or balanced nutrition, which further complicates daily meal decisions.

FridgeAI is a web-based intelligent recipe recommendation system designed to address these challenges. The application allows users to input the ingredients they currently have and receive personalized recipe ideas based on selected meal types, dietary preferences, and health goals. By leveraging artificial intelligence and modern web technologies, FridgeAI aims to simplify cooking decisions, promote healthier eating habits, and reduce food waste.

The system is implemented as a website with a redesigned frontend of a Figma prototype and a backend powered by Supabase, which securely connects the application to an OpenAI service for recipe generation.

1. SDLC Plan

1.1 Selected SDLC Model: Agile (Iterative & Incremental)

Justification:

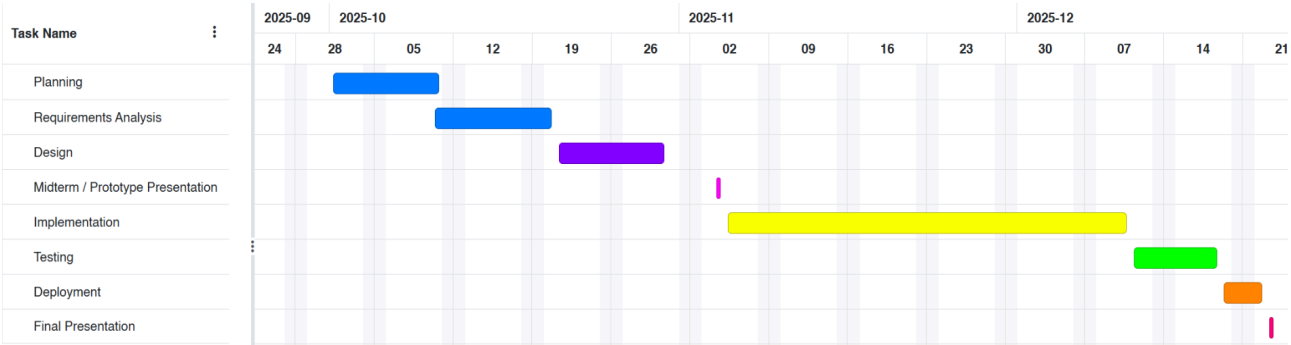
FridgeAI relies on user interaction, AI-generated outputs, and frequent UI/UX adjustments. Agile allows:

- Iterative improvements based on testing and feedback
- Flexible refinement of AI prompts and filtering logic
- Faster development of a working MVP (Minimum Viable Product)

Since the frontend structure was initially generated using Figma AI and then redesigned and implemented manually, Agile fits well with incremental refinement.

Timeline

- **Planning:** Define problem, scope, and objectives
- **Requirements:** Identify user needs and system requirements
- **Design:** Create system architecture and UI structure
- ***Midterm / Prototype Presentation***
- **Implementation:** Develop frontend, backend, and AI integration
- **Testing:** Verify functionality and usability
- **Deployment (Optional):** Host the application online
- ***Final Presentation***



*Visual representation using Gantt Chart Software

2. Requirements (SRS)

1.1 Purpose

This document defines the functional and non-functional requirements for **FridgeAI**, a web-based application that generates recipe ideas using artificial intelligence based on user-input ingredients, meal types, dietary preferences, and health goals.

1.2 Scope

FridgeAI aims to reduce food waste and simplify meal planning by helping users turn available ingredients into practical and personalized recipes. The system provides full recipe details, allows users to save liked recipes, and collect feedback to improve user experience.

---

2. Overall Description

2.1 Product Perspective

FridgeAI is a standalone web application that follows a client–server architecture. The frontend provides user interaction and recipe display, while the backend (Supabase) handles data processing and AI communication.

Problem Statement

Many people waste food because they do not know what meals can be prepared using the ingredients they already have. This issue is worsened by dietary preferences and health goals. FridgeAI solves this by generating recipe ideas based on available ingredients and user preferences.

Users (Target Audience)

- Students living independently or in shared housing
- Busy professionals with limited time for meal planning
- Families and households seeking convenient meal ideas
- Health-conscious individuals focused on balanced nutrition

- Users with dietary restrictions (vegan, vegetarian, gluten-free, lactose-free, etc.)
- People pursuing specific health goals such as weight loss, muscle gain, or healthy maintenance

---

3. Functional Requirements

- **Users can input ingredients:** Users can manually enter one or more ingredients they currently have. These ingredients form the basis for recipe generation.
- **Users can select the suggested ingredients:** The system suggests additional ingredients they *might* want to include, allowing the AI to give recipes that incorporate them when possible.
- **Users can select meal type:** Users can choose breakfast, lunch, dinner, or snacks to ensure recipes match the appropriate meal context.
- **Users can choose dietary preferences:** The system supports filtering recipes according to dietary needs such as vegan, vegetarian, gluten-free, or other restrictions.
- **Users can select health goals:** Users can specify goals such as weight loss, muscle gain, or balanced nutrition. Recipe suggestions will be tailored accordingly.
- **System generates AI-based recipes:** Recipes are dynamically generated using AI based on all user inputs, preferences, and goals.
- **Recipes are displayed with full details:** Each recipe includes ingredients, preparation instructions, estimated cooking time, difficulty level, and approximate calorie count, making it easy for users to follow.
- **Users can save recipes:** Users can mark recipes as “liked” to save them for future reference, creating a personalized recipe collection.
- **Users can leave feedback or rate recipes:** Users can provide ratings and feedback on generated recipes, allowing the system to track satisfaction and improve recommendations over time.
- **Recipes are displayed clearly and accessibly:** The UI presents recipes in a structured, readable format for easy understanding.

4. Non-Functional Requirements

- **Performance:** The system should respond quickly, including generating recipes, saving liked recipes, and submitting feedback, ideally within 5 seconds.

- **Usability:** The interface must be intuitive, allowing users to input ingredients, select preferences, view recipe details, and interact with liked recipes and feedback easily.
- **Reliability:** The system should be consistently available, handling multiple users without errors or downtime.
- **Scalability:** The backend should support growing numbers of users and AI requests without affecting performance.
- **Security:** User data, including saved recipes and feedback, must be securely stored, with encrypted communication between frontend, backend, and AI service.
- **Compatibility:** The application should work on modern browsers and across desktop, tablet, and mobile devices.
- **Data Accuracy:** Recipe details such as ingredients, instructions, calories, and cooking times must be accurate and correctly displayed.
- **Maintainability:** The system should use modular code to allow future updates and feature additions.
- **Feedback and Error Handling:** Users should receive clear notifications for actions like saving recipes, submitting feedback, or entering invalid input.

---

5. Constraints and Assumptions

- An active internet connection is required for AI-based recipe generation.
- Calorie values and nutritional data are approximate and not intended for medical use.
- The quality of generated recipes depends on the accuracy and completeness of user input.

6. Conclusion

This SRS defines the core requirements for the successful development and operation of FridgeAI. By focusing on personalization, usability, and performance, the system provides practical recipe solutions while helping users reduce food waste and improve meal planning efficiency.

---

3. Design

System Architecture

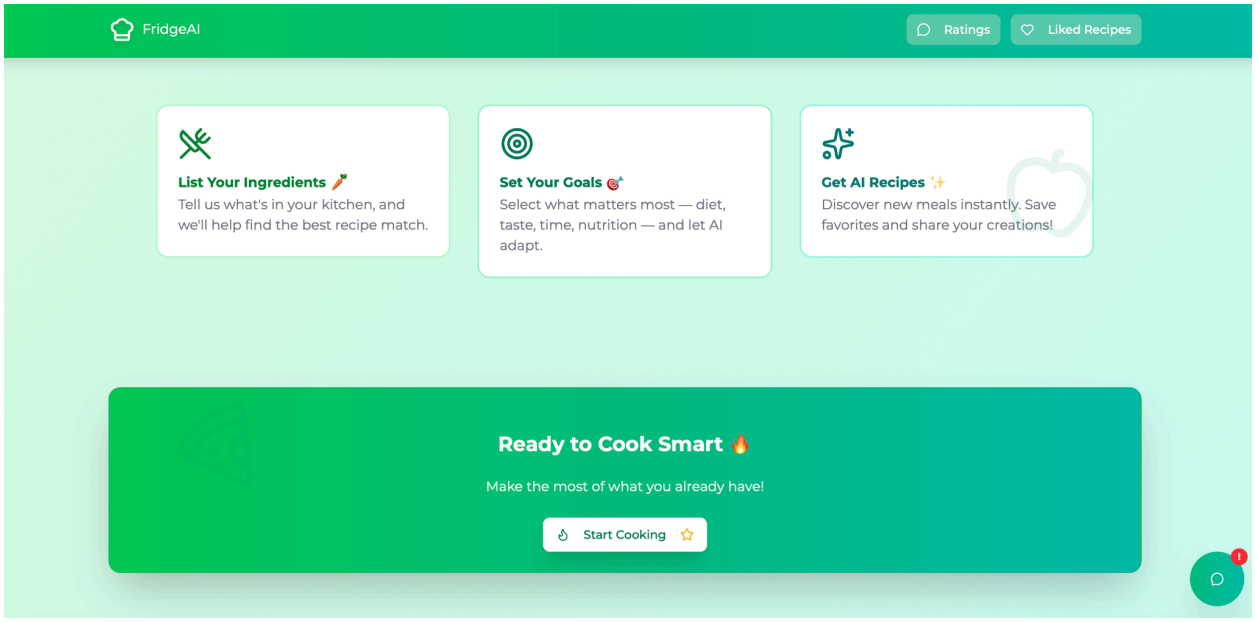
FridgeAI uses a **client–server architecture**. The frontend collects user inputs (ingredients, meal type, dietary preferences, health goals) and displays AI-generated recipes with full details, including instructions, calories, cooking time, and difficulty. The backend (Supabase) processes requests, communicates securely with the AI service, stores liked recipes, and handles user feedback.

Data flow: User ➜ Frontend ➜ Backend ➜ AI Service ➜ Backend ➜ Frontend ➜ User

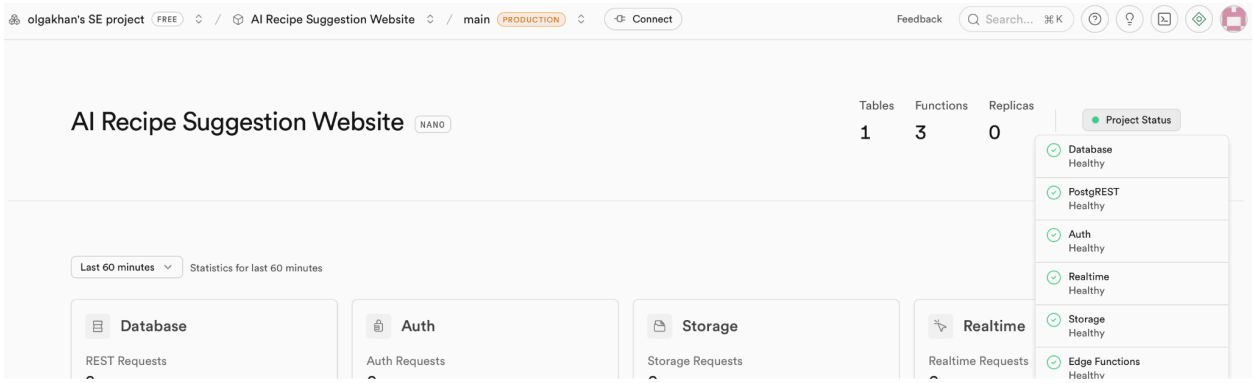
---

Design Decisions

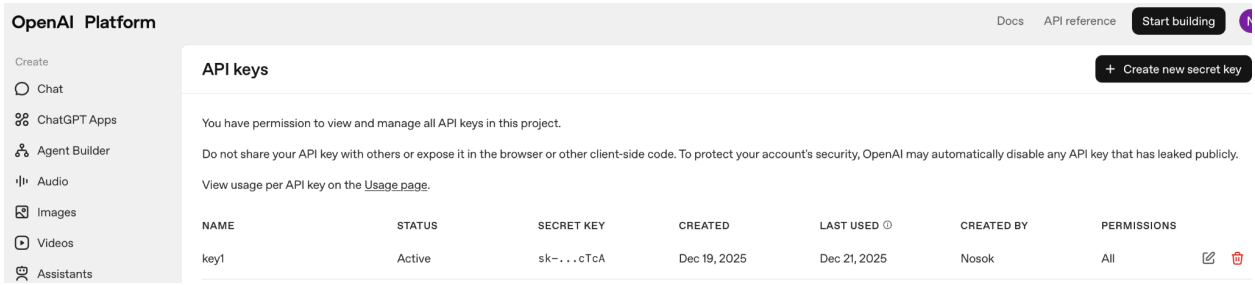
- **Web-Based Platform:** Accessible on any device with a browser, no installation required.



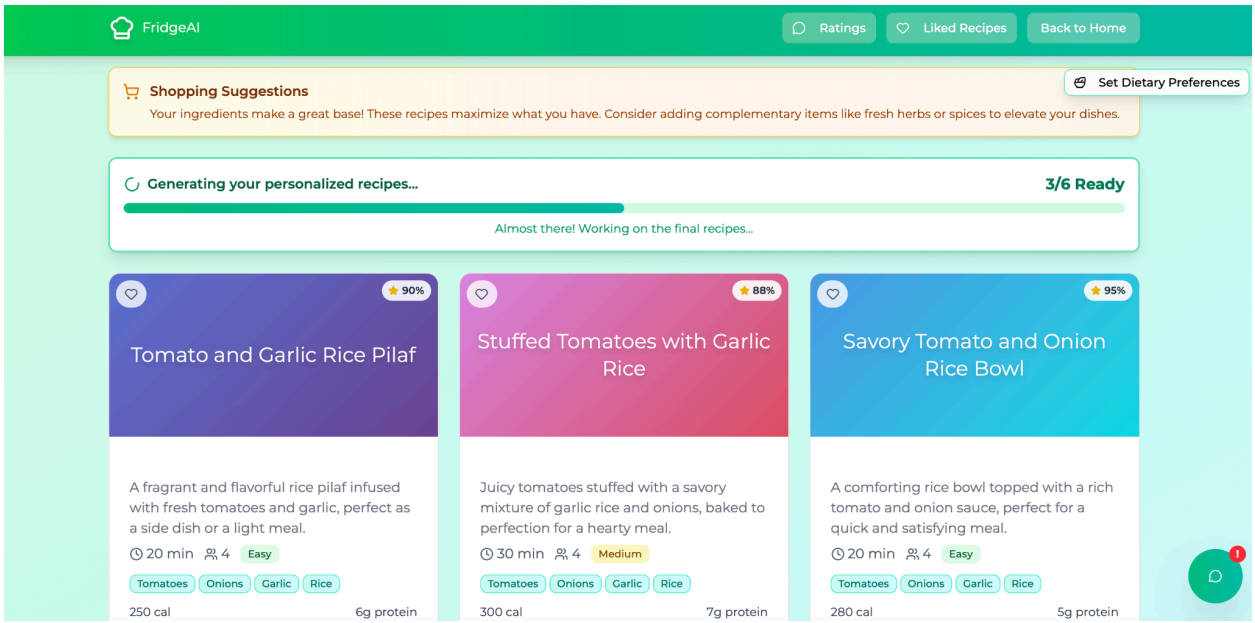
- **Supabase Backend:** Provides secure API handling, authentication, and data storage.



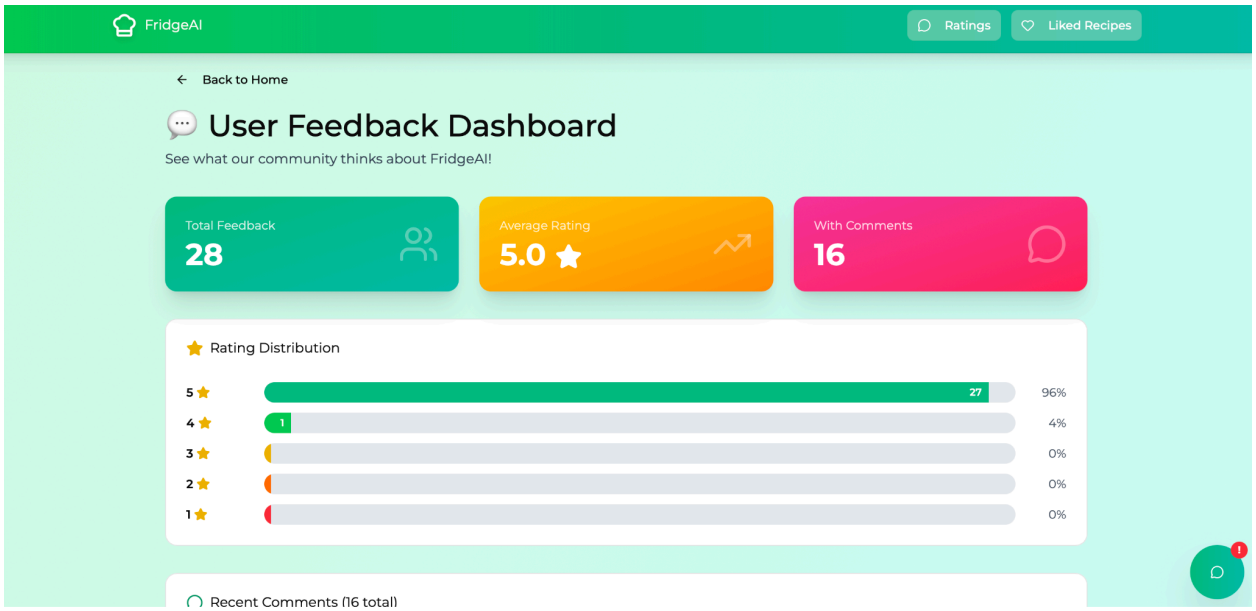
- **Server-Side AI Integration:** Keeps API keys safe and allows consistent prompt formatting.

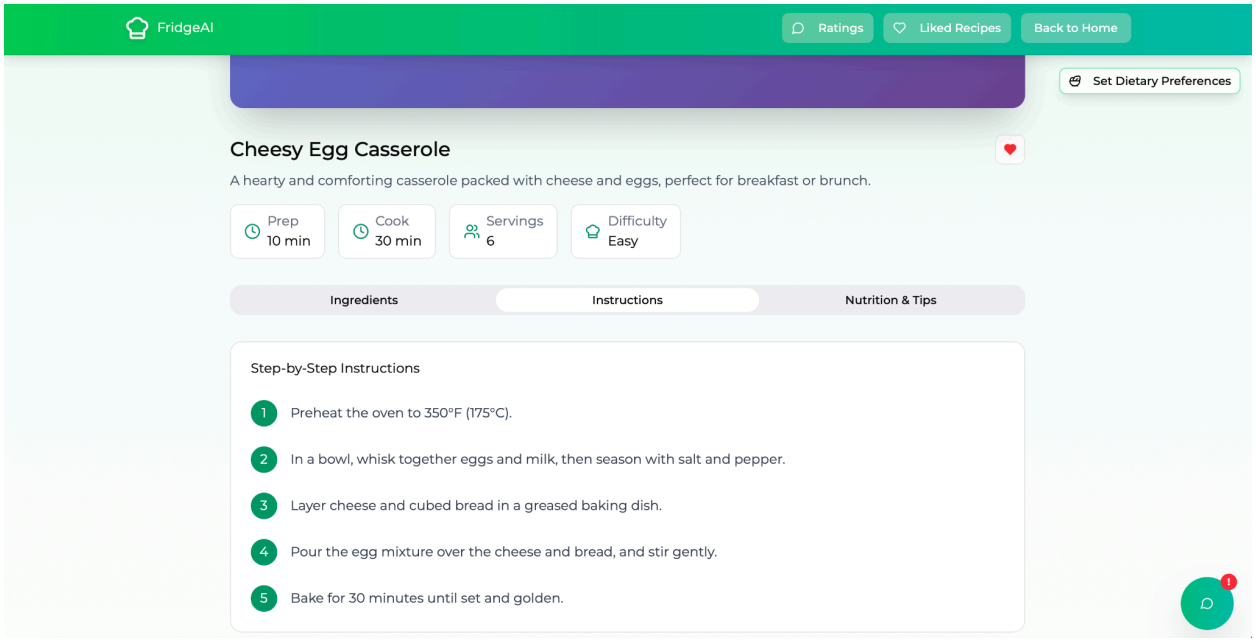


- **Redesigned Frontend:** Improved usability and responsiveness compared to the original Figma AI skeleton.

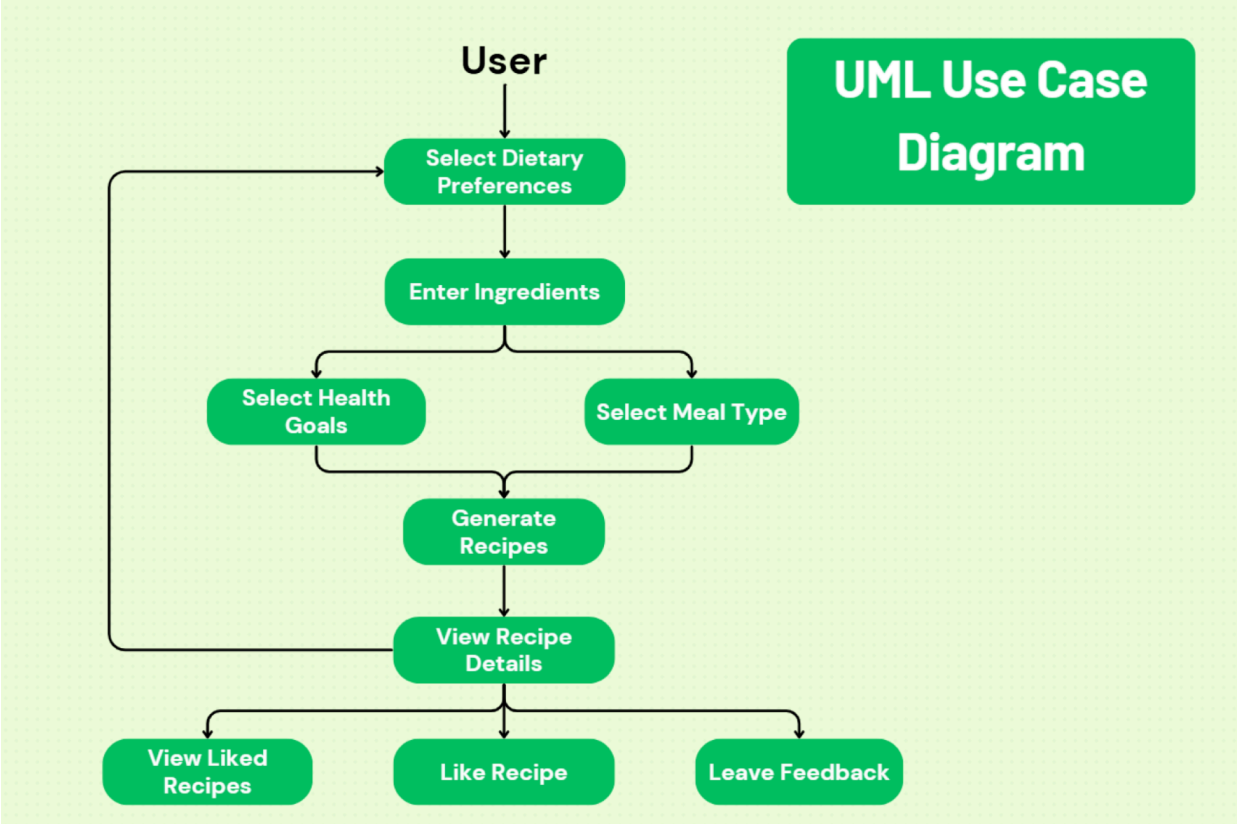


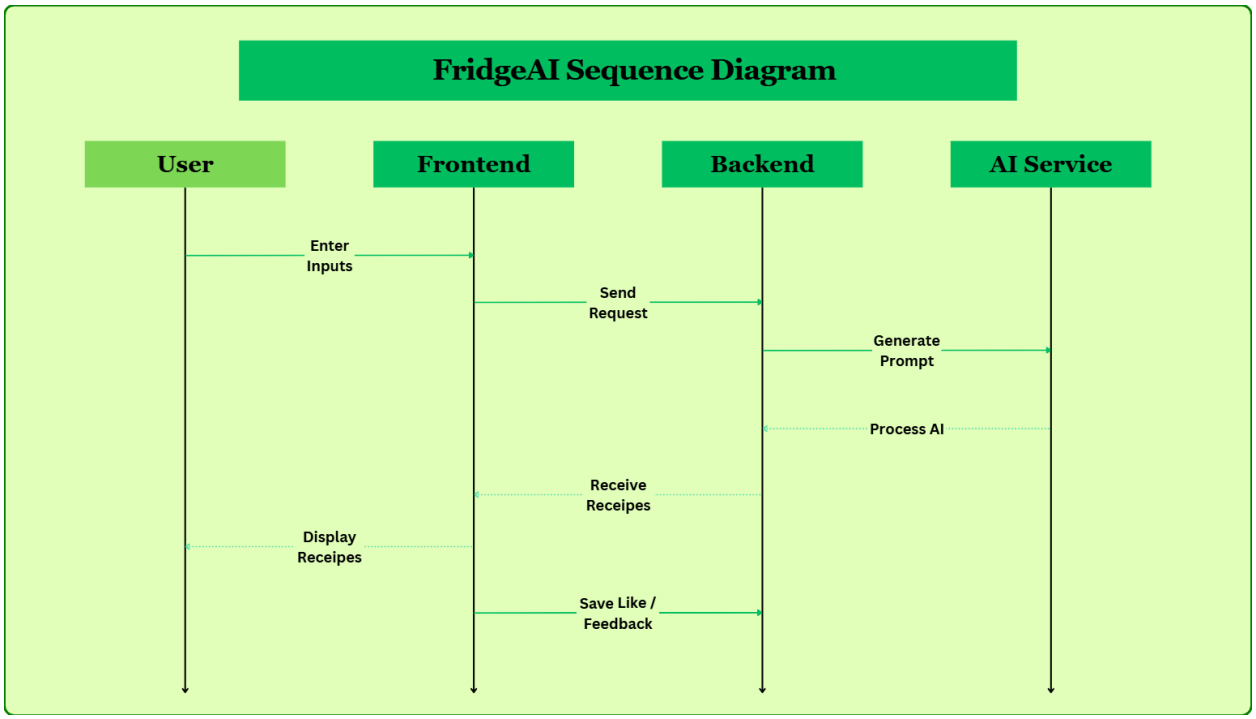
- **Modular Architecture:** Clear separation of frontend, backend, and AI logic, making future maintenance and updates easier.
- **User-Centered Design:** Includes features like optional ingredients, full recipe details, liked recipes, and feedback to enhance engagement and usability.





4. UML Diagrams





5. Implementation (Code)

FridgeAI was implemented as a **web-based application** using modern frontend and backend technologies. The frontend was developed using **TypeScript, HTML, and CSS**, building upon an initial Figma AI–generated skeleton and redesigning it to improve usability, layout, and responsiveness.

The backend is powered by **Supabase**, which manages secure data storage, user interactions (liked recipes and feedback), and communication with the AI service. AI integration is handled server-side using **OpenAI**, ensuring both security and scalability.

FridgeAI supports core features including:

- Ingredient-based recipe generation.
- Meal type selection: breakfast, lunch, dinner, snacks.
- Filtering by dietary preferences (vegetarian, vegan, keto, etc.) and health goals (weight loss, muscle gain, etc.).
- Detailed recipe instructions: calories, difficulty, cooking time.
- Liked recipes page for saved favorites.
- User feedback support.
- Preference logic:
    - Dietary preferences are prioritized; e.g., if vegetarian is selected, inputting meat will be ignored.
    - Preferences appear only once and are saved; they can be changed but otherwise won't pop up again.

The application implements a complete user flow:

```
user → dietary preference → ingredients → goals & meal type (parallel)
   → generate → view recipe details
       → [option 1] go back → redo inputs
       → [option 2] view liked recipes or leave feedback
```

The system functions as a complete, working prototype and meets the functional requirements defined in the SRS. It is deployed on **Vercel**.

6. Testing

Objective:

Verify that FridgeAI correctly generates recipes, enforces dietary preferences, supports user interactions, and handles input validation.

Testing Approach:

Manual testing for user flows and validation. Focus areas: recipe generation, dietary preference enforcement, liked recipes, feedback submission, and error handling.

Test Cases:

ID	Description	Input	Expected Result	Status
TC01	Generate vegetarian breakfast	Dietary preference: Vegetarian; Ingredients: eggs, milk; Meal type: Breakfast	Recipe generated excludes meat; breakfast recipes shown	Pass
TC02	Generate vegan lunch	Dietary preference: Vegan; Ingredients: tofu, rice; Meal type: Lunch	Recipe includes only vegan ingredients; lunch recipes shown	Pass
TC03	Generate keto dinner	Dietary preference: Keto; Ingredients: chicken, broccoli; Meal type: Dinner	Recipe follows keto rules (low carb, high fat); dinner recipes shown	Pass
TC04	Dietary preference enforcement	Dietary preference: Vegetarian; Ingredients: chicken	AI ignores meat; recipe only vegetarian	Pass
TC05	Change dietary preference	Initial: Vegan; Updated: Keto	New recipes reflect updated preference	Pass
TC06	Save liked recipe	Click “Save” on recipe	Recipe appears in liked recipes page	Pass

TC07	View liked recipes	Navigate to liked recipes	All saved recipes displayed correctly	Pass
TC08	Leave feedback	Enter text in feedback box	Feedback stored in Supabase database	Pass
TC09	Invalid input handling	Ingredients: 123, @@; Empty fields	Validation error shown; recipe not generated	Pass
TC10	AI service failure	Simulate OpenAI API error	Error message displayed to user; app does not crash	Pass

7. Deployment

https://fridgeaifinal-jug0qn9pe-olgakhans-projects.vercel.app/

Reference:

Canva https://www.canva.com/online-whiteboard/flowcharts/
Online Gantt. *Online Gantt Chart Tool*. https://www.onlinegantt.com/#/gantt
OpenAI platform. https://platform.openai.com
Supabase. https://supabase.com
Figma. https://www.figma.com