# Cook book :
# Your virtual kitchen assistant

## 1. INTRODUCTION

PROJECT TITLE: COOK BOOK : YOUR VIRTUAL KITCHEN ASSISTANT

**TEAM ID:** NM2025TMID40404
TEAM LEADER: Pragathy.R
Email: pragathy7107@gmail.com

**TEAM MEMBERS:**
1. Anisha begum.C - hameedhachan240@gmail.com
2. Diviyasirii.D - diviyaseenu@gmail.com
3. Charulata.D - charu666latha@gmail.com
4. Ragavi S.R - srragavi12@gmail.com

## 2. PROJECT OVERVIEW

Purpose: The project Cook Book: Your Virtual Kitchen Assistant is designed to help users explore, organize, and create recipes. It provides tools for meal planning, grocery management, and personalized suggestions based on preferences.

**Extended Purpose:**
This project aims to make cooking accessible to everyone, from beginners to professional chefs. It helps busy individuals save time by organizing meal plans and shopping lists. It also assists health-conscious users by tracking nutrition and calories, while enabling families to plan weekly meals collaboratively.

**Features:**
- Recipe search and storage with filtering by cuisine, difficulty, or ingredients.
- Meal planning assistant that can suggest weekly menus.
- Grocery list generation synced with recipes.
- Step-by-step cooking guidance with timers and tips.
- Nutrition and calorie tracking for health-focused users.
- User reviews and ratings to promote community interaction.
- AI-powered personalized suggestions to reduce decision fatigue.

## 3. ARCHITECTURE

The application is designed using a three-tier architecture with frontend, backend, and database layers.

- **Frontend:** Built with React.js, styled using Bootstrap and Material UI for responsiveness.
- **Backend:** Node.js with Express.js handles authentication, user requests, and business logic.
- **Database:** MongoDB stores recipes, user profiles, preferences, and grocery data.

**Additional Considerations:**
- RESTful API endpoints allow smooth communication between frontend and backend.

• JWT authentication ensures secure access.
• The system is designed to be scalable, supporting large recipe datasets and concurrent users.
• Future updates can integrate with IoT devices like smart fridges to suggest recipes based on available ingredients.

## 4. SETUP INSTRUCTIONS

Prerequisites: Node.js, MongoDB, Git, React.js, Express.js

Installation Steps:
1. Clone the repository from GitHub.
2. Navigate to the client folder and run `npm install` to install frontend dependencies.
3. Navigate to the server folder and run `npm install` to install backend dependencies.
4. Configure environment variables (MongoDB URI, JWT secret, API keys if any).
5. Start the backend with `npm start` inside the server folder.
6. Start the frontend with `npm start` inside the client folder.
7. Access the application at http://localhost:3000

Troubleshooting:
• Ensure MongoDB service is running.
• If ports clash, update the default ports in configuration files.
• Use `npm run build` to prepare production deployment.

## 5. FOLDER STRUCTURE

CookBook/
|-- client/ (React frontend)
| |__ components/ - reusable UI components like buttons, forms, cards
| |__ pages/ - full-page layouts such as Dashboard, Login, Recipe Details
|-- server/ (Node.js backend)
| |__ routes/ - API endpoints for recipes, users, groceries
| |__ models/ - Mongoose schemas for user, recipe, grocery list
| |__ controllers/ - Business logic to process API requests

This modular structure ensures easy scalability and maintainability.

## 6. RUNNING THE APPLICATION

• Frontend: cd client → npm start
• Backend: cd server → npm start
• Access via http://localhost:3000

Additional Notes:
• For production, run `npm run build` inside the client folder to generate optimized files.
• Configure a reverse proxy (like Nginx) for deployment.

• Use environment variables to manage sensitive credentials.
• Docker can be used to containerize both client and server for easy deployment.

## 7. API DOCUMENTATION

User APIs:
- POST /api/user/register → Registers a new user.
- POST /api/user/login → Authenticates user and returns JWT token.

Recipes APIs:
- POST /api/recipes/create → Add a new recipe with title, ingredients, and instructions.
- GET /api/recipes/:id → Retrieve recipe details by ID.

Meal Plans APIs:
- POST /api/mealplan/create → Create a weekly meal plan.
- GET /api/mealplan/:id → Retrieve specific meal plan.

Grocery List APIs:
- POST /api/grocery/add → Add items to grocery list.
- GET /api/grocery/:id → Fetch grocery list details.

Example Request (JSON):
{ "title": "Pasta", "ingredients": ["pasta", "tomato sauce"], "instructions": "Boil pasta and add sauce." }

## 8. AUTHENTICATION

• JWT-based authentication ensures only authorized users can access protected routes.
• Middleware verifies tokens for every request.
• Token expiry is set for security, with refresh tokens available.
• User roles:
- Admin: Can manage recipes, users, and system settings.
- Standard User: Can search, save, and manage personal recipes.

This role-based system ensures proper access management.

## 9. USER INTERFACE

The application provides an intuitive and user-friendly interface.

• Landing Page → Welcomes users and highlights key features.
• Recipe Dashboard → Displays saved recipes, trending suggestions, and quick filters.
• Meal Planner → Interactive calendar to plan meals across the week.
• Grocery List Manager → Generates and manages shopping lists with export options.
• Admin Panel → Allows admins to manage content and users.
• Recipe Details Page → Provides full cooking instructions, nutrition breakdown, and user

reviews.

## 10. TESTING

Testing ensures the reliability of the application.

- Manual testing during development milestones.
- API testing using Postman for endpoints validation.
- Automated unit testing for recipe and grocery modules using Jest/Mocha.
- Integration testing ensures communication between frontend and backend.
- End-to-end testing (E2E) using tools like Cypress to simulate user behavior.

## 11. SCREENSHOTS OR DEMO

Screenshots of the landing page, recipe search, grocery list, and meal planner will be provided in the demo version.

Future Demo Plans:
- Include a video walkthrough of the cooking assistant in action.
- Provide GIFs showcasing interactive features like recipe search filters and grocery generation.


## 12. KNOWN ISSUES

- Limited offline functionality; requires internet for full use.
- Some advanced dietary filters (e.g., vegan keto) are not yet implemented.
- Mobile version is still under development and may have layout inconsistencies.
- Performance issues may occur with extremely large datasets.

## 13. FUTURE ENHANCEMENTS

- AI-based personalized recipe recommendations using machine learning.
- Integration with smart kitchen appliances (smart ovens, fridges, etc.).
- Voice-controlled cooking assistant for hands-free experience.
- Global recipe sharing community where users can upload and rate recipes.
- Support for AR/VR guided cooking for immersive learning experiences.
- Gamification elements such as cooking challenges and achievements to engage use