

# Project Documentation

---

## CookBook

### 1. Introduction

- Project Title: cookbook

**Team ID** : NM2025TMID42570

**Team Size** : 4

**Team Leader** : Reena .P (reena8825883572@gmail.com)

**Team member** : Preethika. L (preethika19102006@gmail.com)

**Team member** : Rohini . J (rohinipapa1234@gmail.com)

**Team member** : Sabiha. S (sabihasafi786786@gmail.com)

### 2. Project Overview:

The primary purpose of CookBookDB is to serve as a comprehensive, digital recipe management system. It aims to connect users with a vast collection of recipes and provide them with the tools to organize, discover, and interact with culinary content

#### Project Objective:

The primary goal of the MongoDB Cookbook project is to build a flexible, scalable, and efficient recipe management system where users can store, retrieve, update, and search for cooking recipes. By leveraging MongoDB's NoSQL document-based structure, the system can handle diverse and nested data formats that are typical in recipe storage (e.g., ingredients, steps, categories, user ratings).

**Flexible Schema:** Recipes may have varying numbers of ingredients, steps, or tags — MongoDB's document model handles this naturally.

**Embedded Documents:** Ingredients and instructions can be stored within the main recipe document, making it easier and faster to retrieve the entire recipe.

**Scalability:** As the number of recipes and users grows, MongoDB can scale horizontally to handle large volumes of data efficiently.

**Indexing & Search:** MongoDB supports text indexes and compound indexes, making it ideal for building fast

### 3. Architecture

- Frontend: React.js with Bootstrap and Material UI
- Backend: Node.js and Express.js managing server logic and API endpoints
- Database: MongoDB stores user data, project information, applications, and chat messages

### 4. Setup Instructions

- **Prerequisites:** – Node.js – MongoDB

– Git – React.js – Express.js – Mongoose – Visual Studio Code . Install Node.js

Node.js is required to run JavaScript outside the browser and to use npm (Node Package Manager).

#### Steps:

##### **Download Node.js:**

Go to <https://nodejs.org>

Choose the LTS (Recommended) version (Long-Term Support – stable version).

##### **Run the Installer:**

Open the downloaded file.

**Follow the installation steps:**

Accept the license agreement

Choose installation location (keep default)

Check the option "Automatically install the necessary tools" if prompted

Click Install.

**Verify Installation:**

Open Command Prompt (CMD) or Terminal.

Type:

`node -v`

You should see a version number (e.g., v20.x.x).

**Check npm (Node Package Manager):**

`npm -v`

You should see a version number too (e.g., 10.x.x).

✓ Node.js and npm are now installed.

**2. Install Visual Studio Code (VS Code)**

VS Code is a code editor used to write, debug, and run JavaScript/Node.js programs.

**Steps:**

**Download VS Code:**

Go to <https://code.visualstudio.com>

Download for your OS (Windows, macOS, Linux).

**Install VS Code:**

Run the installer.

Select options like "Add to PATH" and "Open with Code" during installation (recommended).

Finish installation.

### **Open VS Code:**

Launch the app.

You'll see a welcome screen.

### **3. Setup VS Code for Node.js**

To make coding easier, install some useful extensions:

JavaScript (ES6) Snippets – for faster JS coding.

Node.js Extension Pack – for debugging and running Node apps.

Prettier - Code Formatter – to keep code clean and well-formatted.

#### **How to Install Extensions:**

Open VS Code → Go to Extensions (left sidebar icon or Ctrl+Shift+X).

Search for the extensions above and click Install.

### **4. Test Setup**

#### **Create a Folder:**

Create a new folder, e.g., node-test.

#### **Open in VS Code:**

Open VS Code → File → Open Folder → select node-test.

#### **Create a File:**

Create a file called app.js.

#### **Add this code:**

```
console.log("Node.js is working!");
```

#### **Run the File:**

Open VS Code Terminal (Ctrl + ~).

#### **Run:**

```
node app.js
```

## **5. Folder Structure**

SB-Works/

|-- client/

|\_\_components/

L\_\_ pages/

|\_\_ server/

|\_\_routes/

|\_\_ models/

|\_\_ controllers/

# React frontend

# Node.js backend Windows (using the MSI installer):

**Installation Directory:** C:\Program Files\MongoDB\Server\<version>\

**Data Directory:** C:\Program Files\MongoDB\Server\<version>\data\ (often a custom path is chosen during installation)

Log Directory: C:\Program Files\MongoDB\Server\<version>\log\

## 6. Running the Application

### Frontend:

cd

client

npm start

### Backend:

cd server npm

start

**Access:** Visit <http://localhost:3000>

## 7. API Documentation

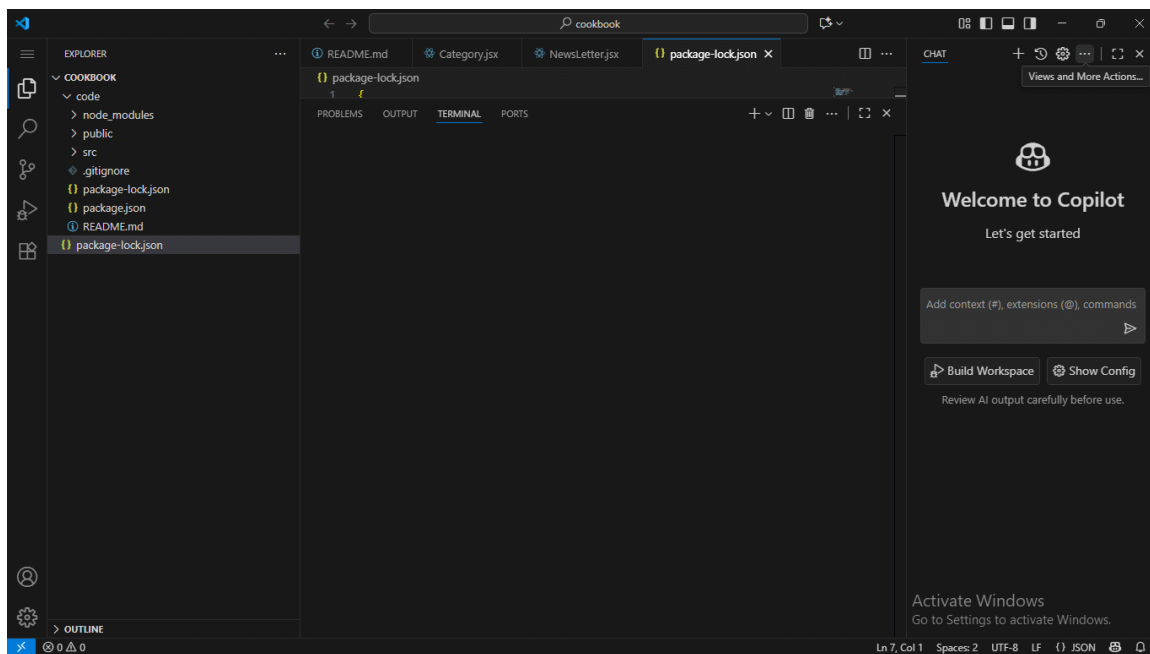
- User: – /api/user/register – /api/user/login
- Projects:
  - /api/projects/create – /api/projects/:id
- Applications: /api/apply
- Chats: – /api/chat/send – /api/chat/:userId

## 8. Authentication

- JWT-based authentication for secure login
- Middleware protects private routes

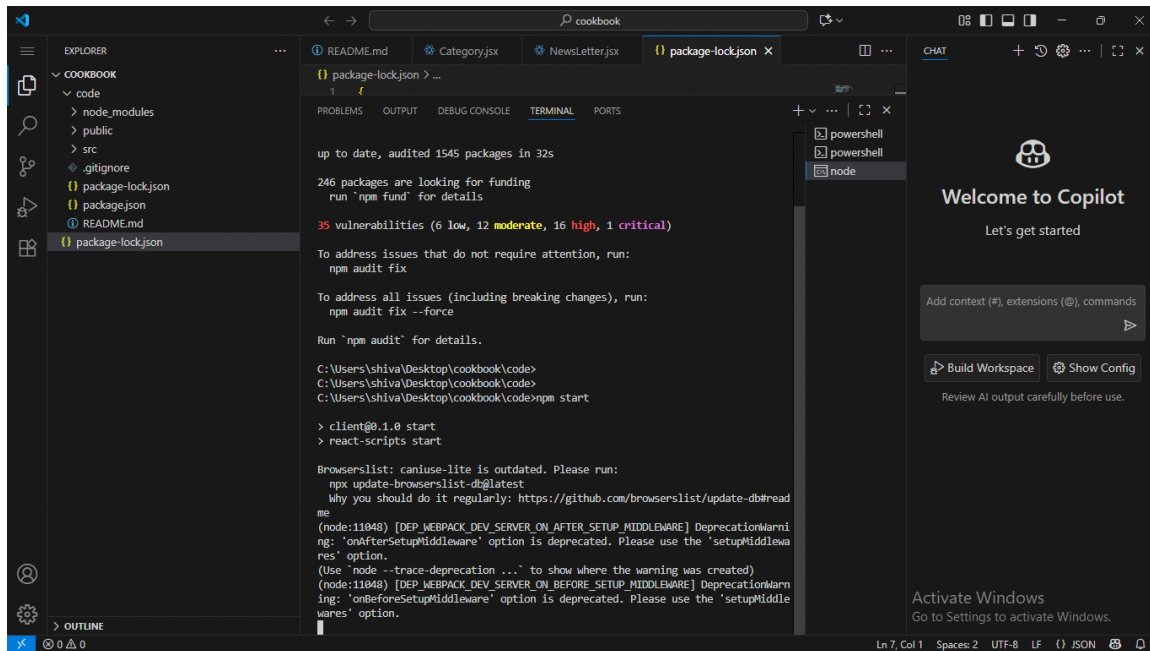
## 9. User Interface

- Landing Page :vs code



- Freelancer Dashboard
- Admin Panel

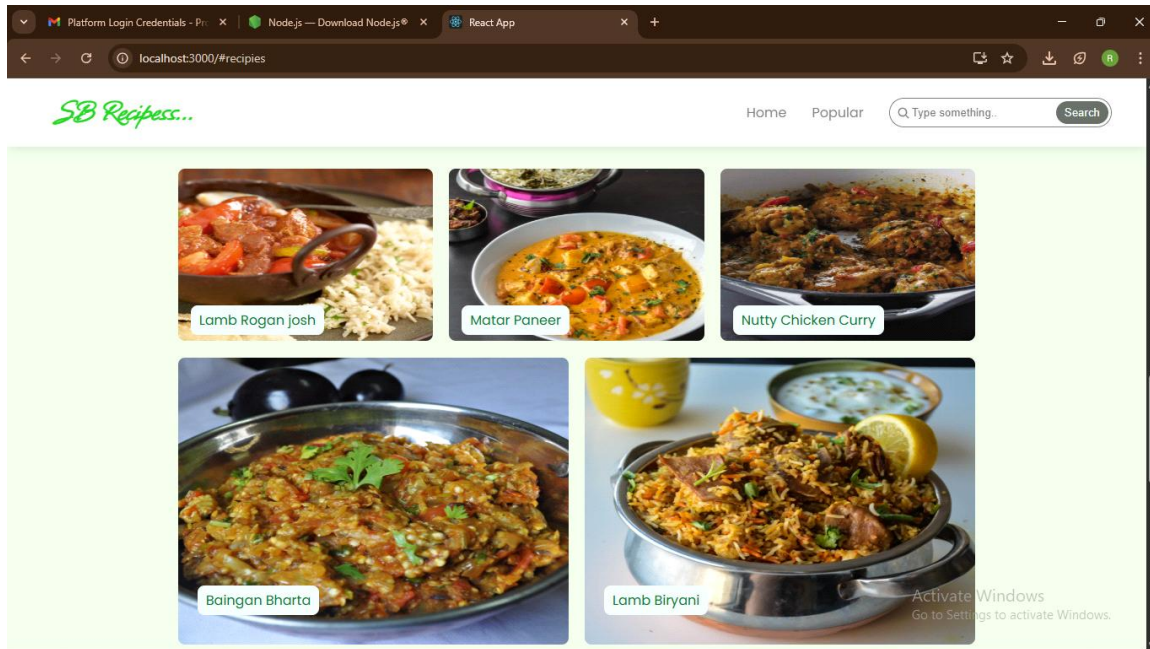
- Project Details Page



## 10. Testing

- Manual testing during milestones
- Tools: Postman, Chrome Dev Tools

## 11. Screenshots or Demo



## 12. Known Issues :

### 1. MongoDB Connection Issues

#### **Symptoms:**

MongoServerError: bad auth : Authentication failed

**MongoNetworkError:** connect ECONNREFUSED orTIMEDOUT

**Causes & Fixes:** Wrong connection string – Make sure your MONGODB\_URI is correct.

Example: mongodb+srv://<username>:<password>@cluster0.mongodb.net/<dbname>?retryWrites=true&w=majority

Unwhitelisted IP address – Go to MongoDB Atlas → Network Access → Add your current IP (0.0.0.0/0 for public access in dev).

**Wrong database name** – Use the exact database name you created (case-sensitive).

– URL encode passwords with special characters (@, #, !, etc.). ✓ Password special characters

### 2. Node.js Driver Version Mismatch



### **Symptoms:**

DeprecationWarning: current Server Discovery and Monitoring engine is deprecated  
Driver fails to connect or throws unexpected errors.

### **Fixes:**

#### **Upgrade to latest driver:**

```
npm install mongodb@latest
```

If using Mongoose:

```
npm install mongoose@latest
```

## **3. VS Code Environment Issues**

### **Symptoms:**

Environment variables not loading in process.env

Code works in terminal but not in VS Code debugger.

### **Fixes:**

Create a .env file and load it with dotenv:

```
npm install dotenv
```

```
require('dotenv').config();
```

Check VS Code launch configuration (.vscode/launch.json) to include "envFile":

```
"${workspaceFolder}/.env".
```

# **13. Future Enhancement**

## **1. Real-Time Collaboration and Notifications**

Enhance the platform by introducing real-time features.

**Live Updates:** Implement WebSocket technology (e.g., Socket.IO) to push real-time updates to the

UI. When a freelancer marks a task as complete, the client's dashboard would update instantly without a page refresh.

**App Messaging:** Add a chat feature within the Project Details page to facilitate instant communication between clients and freelancers. This would likely involve a separate `messages` collection in MongoDB, possibly using a capped collection for performance with a high volume of message Set up a system to send push notifications for key events, such as new messages, project invites, or payment confirmations.