“SB Foods – Food Ordering app”



NAAN MUDHALVAN PROJECT

NM1042 MERN STACK POWERED BY MONGO DB

**Submitted by**

# D BALAKUMAR (412721205005)

# J GOKUL RAJA (412721205012)

# R SUGUMAR (412721205048)

# D SIVAKAVINDRAN (412721205046)

# V SAISIVAA (412721205040)

DEPARTMENT OF INFORMATION TECHNOLOGY



**TAGORE ENGINEERING COLLEGE**

**RATHINAMANGALAM, CHENNAI 600 127**

**ANNA UNIVERSITY: CHENNAI 600 025**

**NOVEMBER 2024**

**BONAFIDE CERTIFICATE**

Certified that this project report **“Sb foods – Food Ordering app”** is the bonafide work of **Mr. D. Balakumar, Mr. J. Gokul raja, Mr. V. Saisivaa, Mr. R. Sugumar, Mr. D. Sivakavindran** of final year students of Information Technology, who carried out the Naan Mudhalvan project work in **NM 1042 MERN STACK POWERED BY MONGO DB** under my supervision.

**Signature of the Course coordinator Signature of the HOD**

**INTERNAL EXAMINER EXTERNAL EXAMINER**

## TABLE OF CONTENT

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO** | **TITLE** | **PAGE NO** |
| **1.** | **INTRODUCTION** | 1 |
|  |  |
| 2. | **Project Overview** | 9 |
| 3. | **Architecture** | 11 |
| 4. | **Setup Instructions** | 13 |
| 5. | **Folder Structure** | 37 |
| 6. | **Running the Application** | 69 |
| 7. | **Authentication Interface** | 72 |
| 8. | **Testing** |  |
| 9. | **Screenshots or Demo** |  |
| 10. | **Known Issues** |  |
| 11. | **Future Enhancements** |  |

\

**1.INTRODUCTION:**

 **Project Title**: SB Foods - Online Food Ordering Platform

 **Team Members**:

BalaKumar D

Sai Sivaa V

Sugumar R

Sivakavindiran D

Gokul Raja J

SB Foods is a modern online food ordering platform developed with the MERN stack—MongoDB, Express.js, React, and Node.js. This application is designed to simplify the food ordering experience by providing users with detailed information about each dish, including descriptions, pricing, reviews, and current promotions, allowing them to make informed choices effortlessly. The platform’s primary goal is to create a fast, intuitive, and reliable ordering process, from dish selection to checkout, with real-time order confirmations enhancing user satisfaction. The application also includes an Admin Dashboard to manage user profiles, orders, and product information, ensuring a robust backend system. With a responsive design, SB Foods caters to both desktop and mobile users, providing a seamless experience regardless of device, day, or time.

### **2. Project Overview**

* **Purpose**: The SB Foods project is an online food ordering application aimed at simplifying the food selection and ordering process. By providing users with detailed information about each dish, SB Foods allows for easy, informed decision-making, creating a quick, reliable, and enjoyable ordering experience.
* **Features**:
  + **User Account Management**: Sign-up, login, and profile management.
  + **Dish Exploration**: Users can view dish details, reviews, pricing, and promotions.
  + **Ordering Process**: Seamless ordering with real-time order confirmation.
  + **Admin Dashboard**: Allows admins to manage user profiles, orders, and product information.
  + **Real-Time Notifications**: Order status updates.
  + **Responsive Design**: Optimized for desktop and mobile devices.

### **3. Architecture**

* **Frontend**: Built using React, the frontend provides a responsive, single-page experience, allowing users to browse dishes, manage profiles, and complete the order process efficiently.
* **Backend**: The backend is built with Node.js and Express.js, which manage API endpoints for handling user data, order processing, and admin functionalities. The server handles business logic, processes orders, and manages notifications.
* **Database**: MongoDB stores data for users, dishes, orders, and admin details. The database schema includes:
  + **User Collection**: Stores user profiles, order history, and login credentials.
  + **Dish Collection**: Stores dish names, descriptions, prices, reviews, and promotion details.
  + **Order Collection**: Tracks orders with user information, order items, status, and delivery info.

### **4. Setup Instructions**

* **Prerequisites**:
  + **Node.js** and **npm** (for backend and frontend development).
  + **MongoDB** (for data storage).
  + **Git** (for version control).
* **Installation**:
  + Clone the repository: git clone https://github.com/your-repo/sb-foods.git
  + Navigate to the server and client directories and install dependencies:
    - Backend: cd server -> npm install
    - Frontend: cd client -> npm install
  + Set up environment variables (.env files) for sensitive information like database URLs and API keys.

### **5. Folder Structure**

* **Client**: The React frontend includes components, services, styles, and utils for organizing the UI logic and styling.
  + **/src**
    - **/components**: Holds reusable UI components (e.g., DishCard, OrderForm).
    - **/pages**: Contains main pages (e.g., Home, Menu, Checkout).
    - **/services**: API service files for interacting with backend endpoints.
* **Server**: The Node.js backend includes routes, controllers, and models for handling HTTP requests and managing data.
  + **/src**
    - **/routes**: Contains API routes (e.g., /orders, /dishes, /users).
    - **/controllers**: Business logic for handling data requests.
    - **/models**: Mongoose models for MongoDB collections (e.g., User, Dish, Order).

### **6. Running the Application**

* **Frontend**:
  + Navigate to the client directory: cd client
  + Start the frontend server: npm start
* **Backend**:
  + Navigate to the server directory: cd server
  + Start the backend server: npm start

### **7. API Documentation**

* **Endpoints**:
  + **User Management**:
    - POST /api/users/register: Register a new user.
    - POST /api/users/login: Authenticate a user.
  + **Dishes**:
    - GET /api/dishes: Retrieve all available dishes.
    - GET /api/dishes/:id: Retrieve details for a specific dish.
  + **Orders**:
    - POST /api/orders: Place a new order.
    - GET /api/orders/:userId: Retrieve all orders for a user.

### **8. Authentication**

* **User Authentication**:
  + **JWT Tokens**: Secure authentication using JSON Web Tokens (JWT) for access control.
  + **Session Management**: User sessions are managed with token-based authentication to prevent unauthorized access.
* **Admin Authorization**:
  + Admins have access to specific routes and features, ensuring data security and restricted access to sensitive operations.

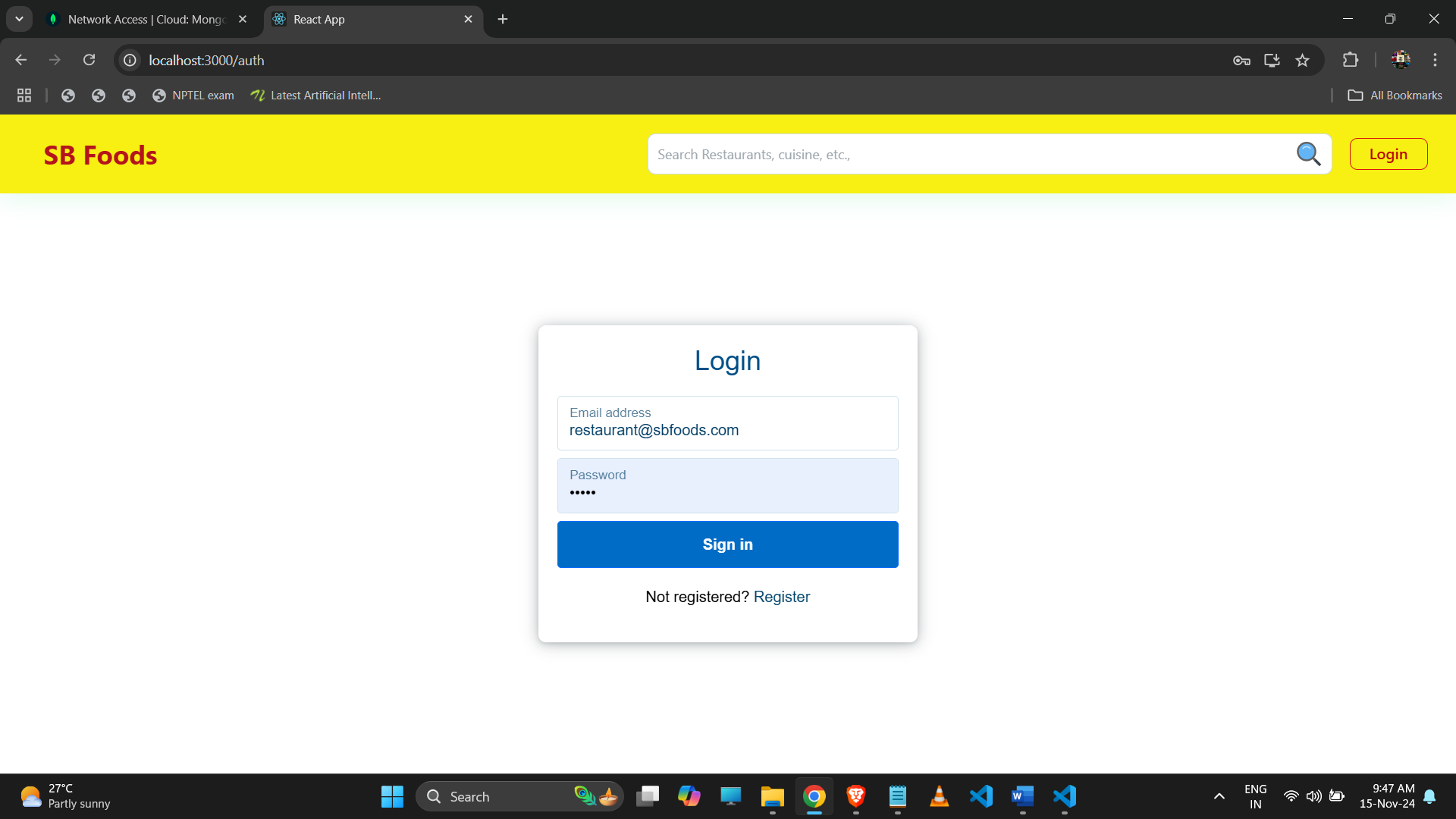
### **9. User Interface**

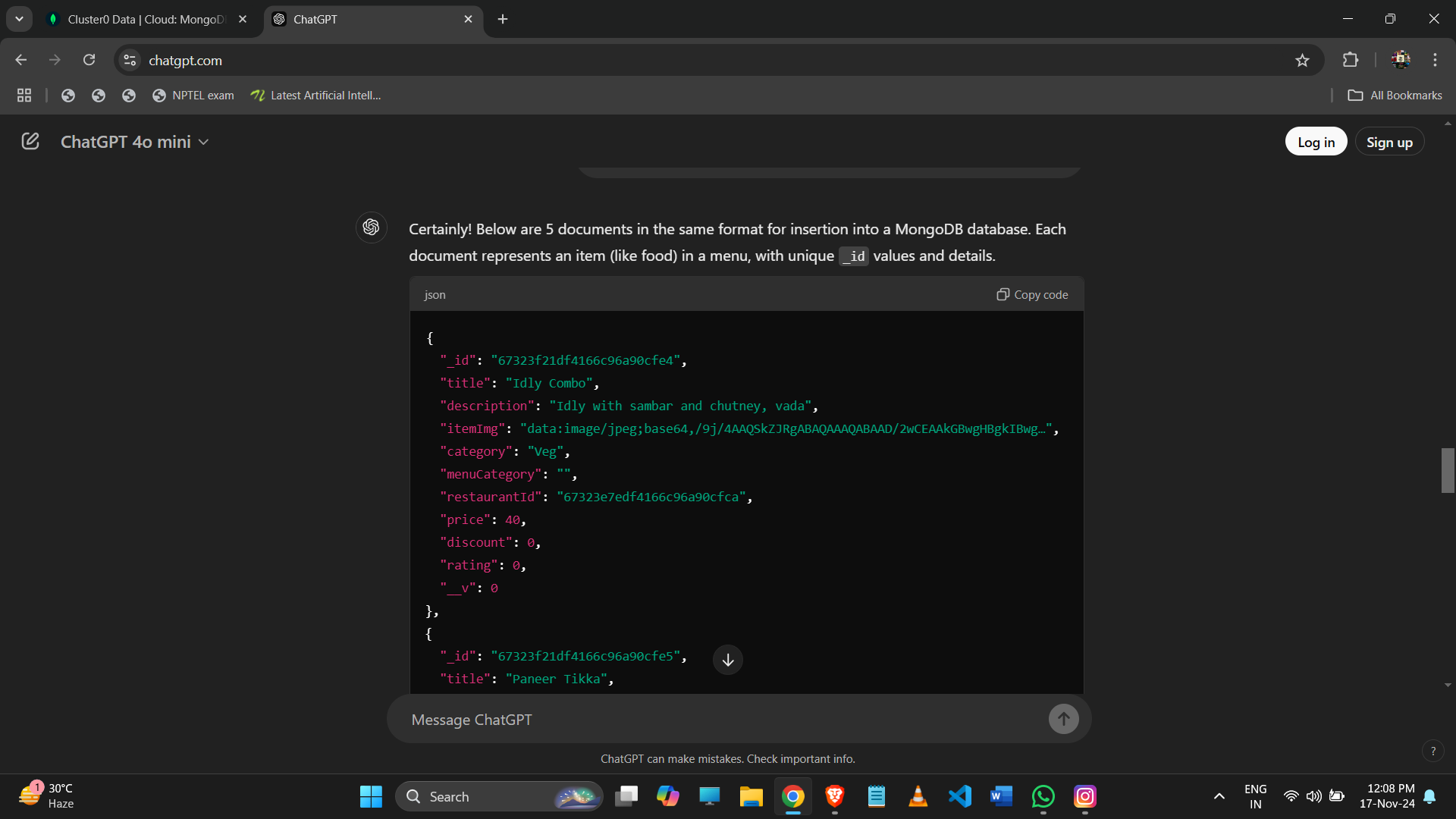
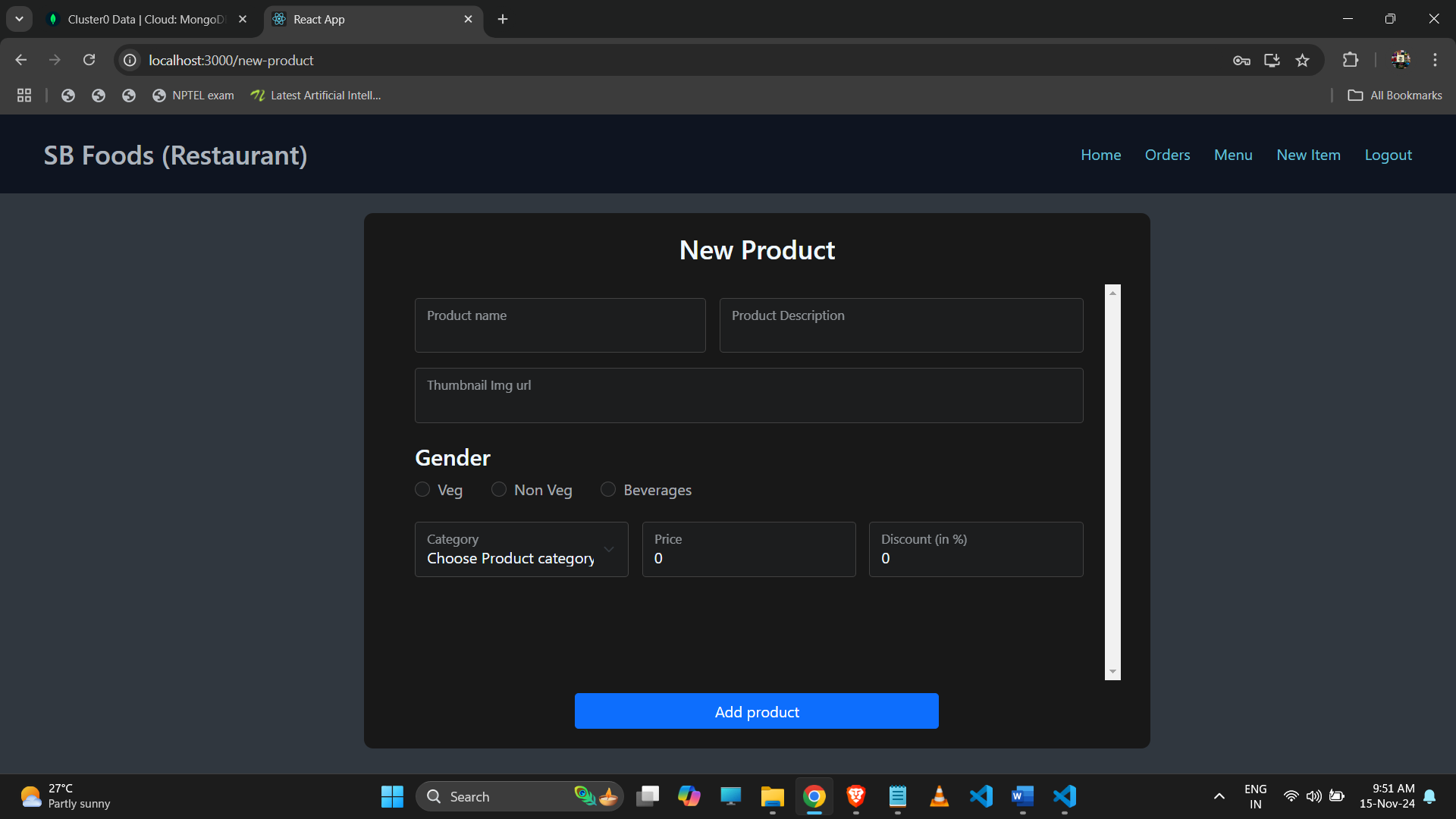
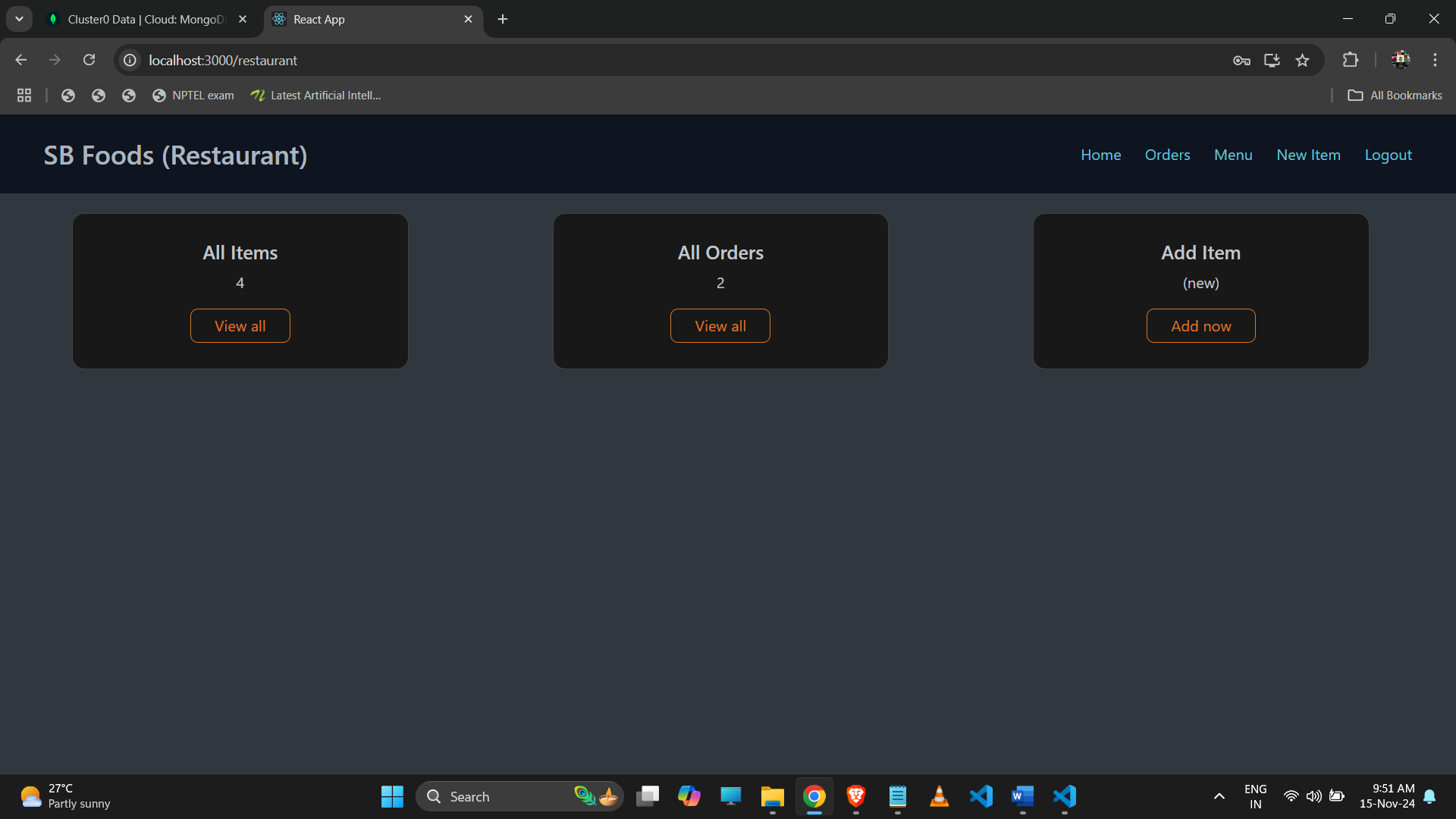
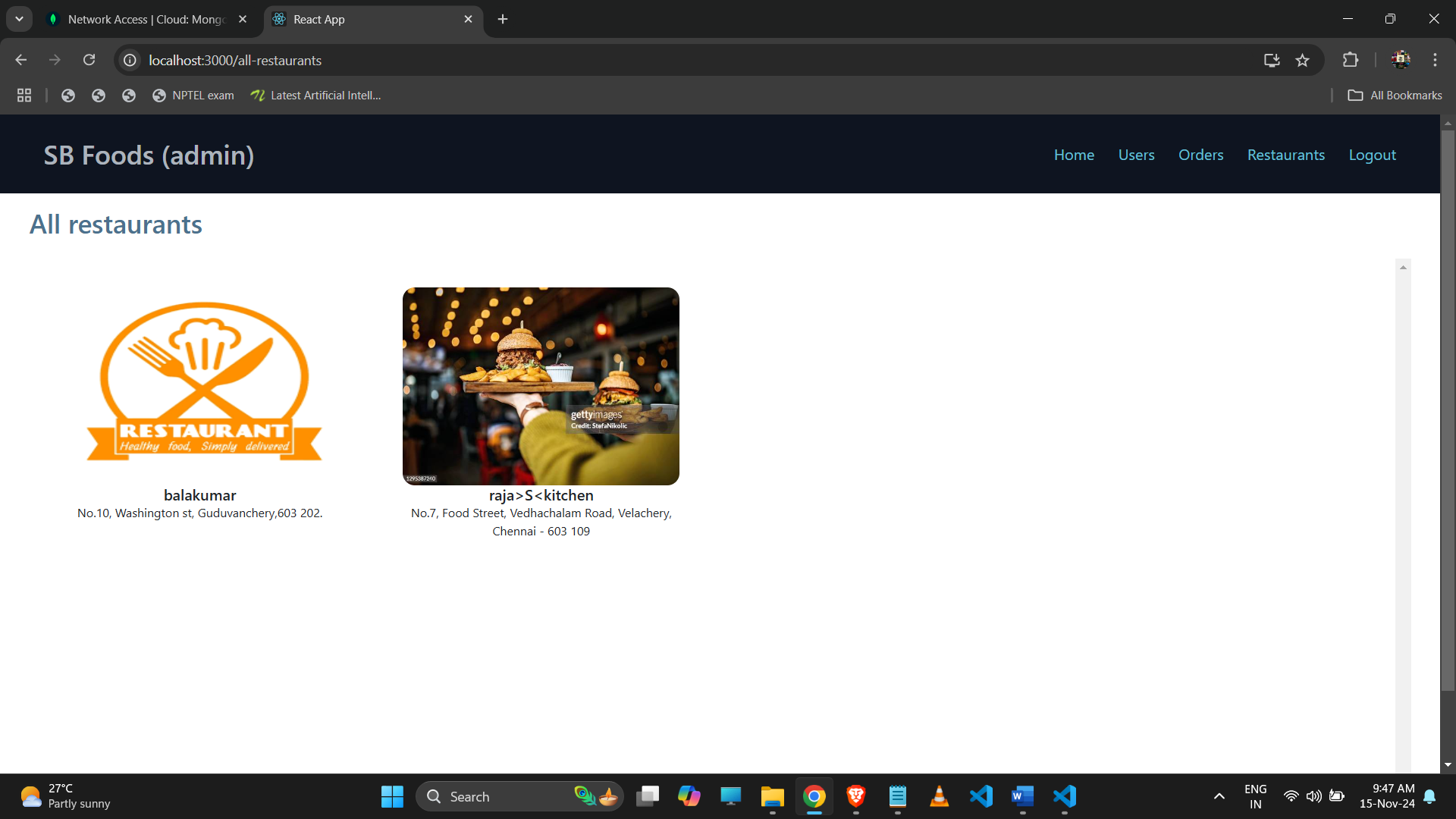
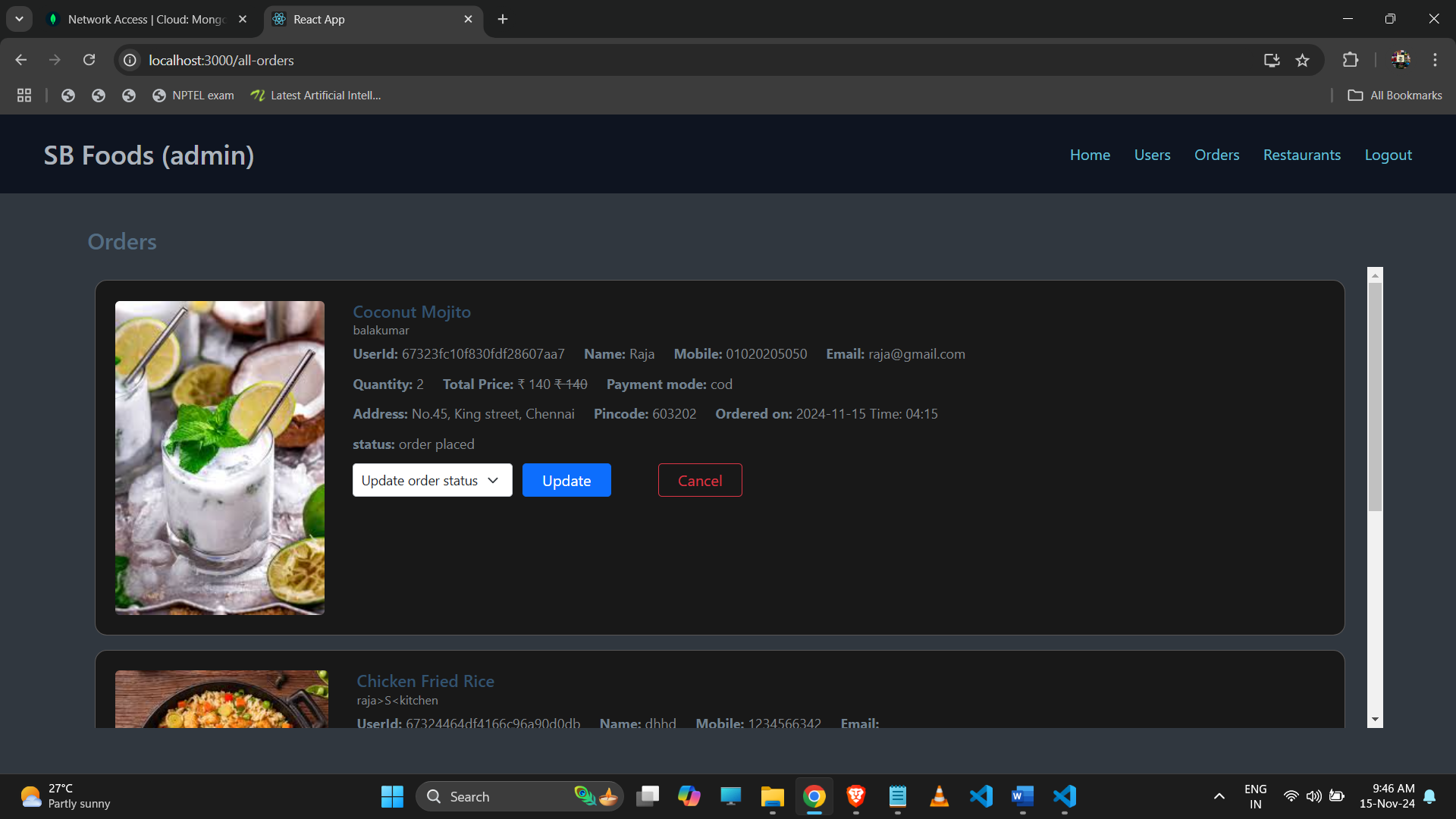
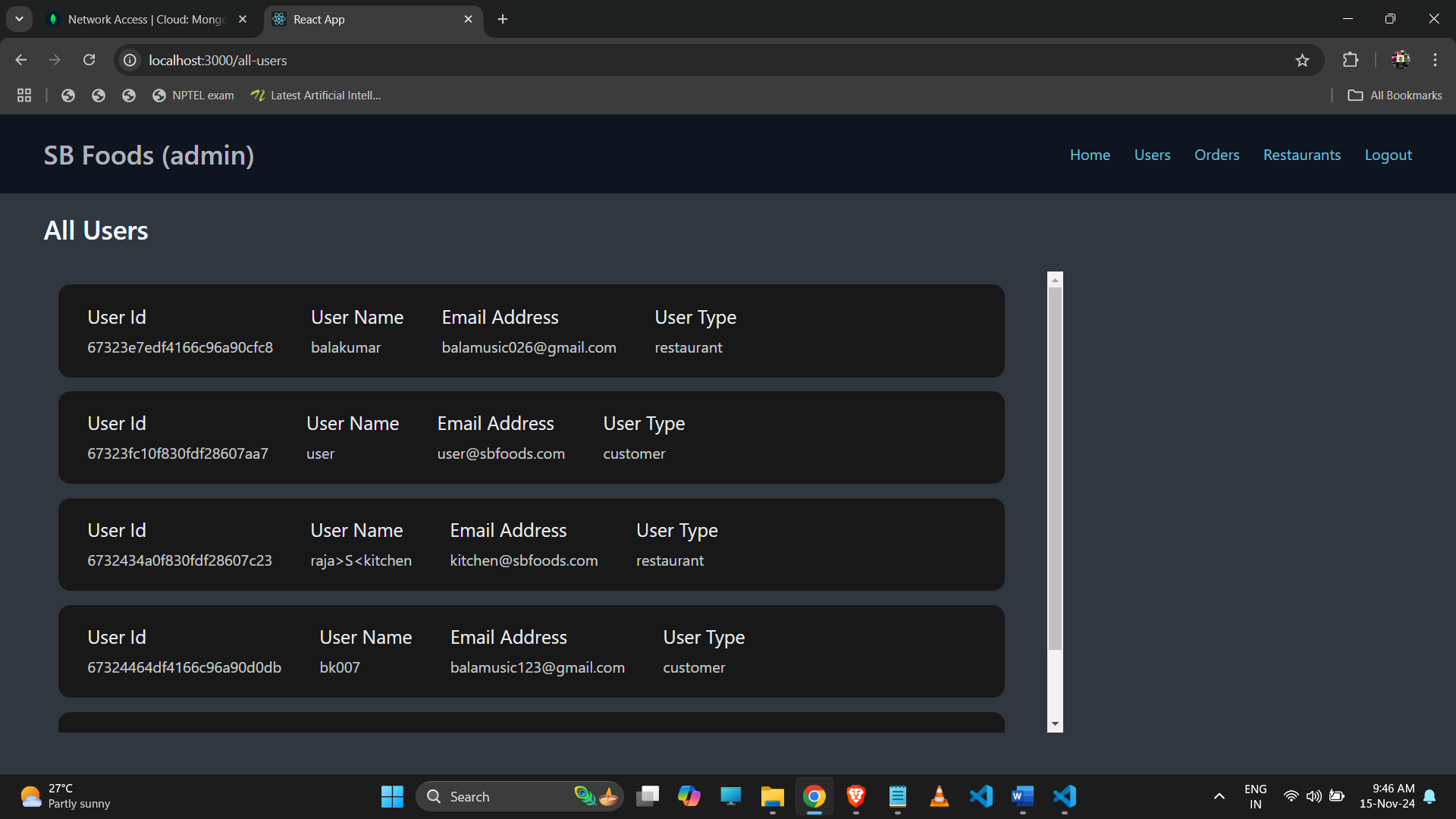
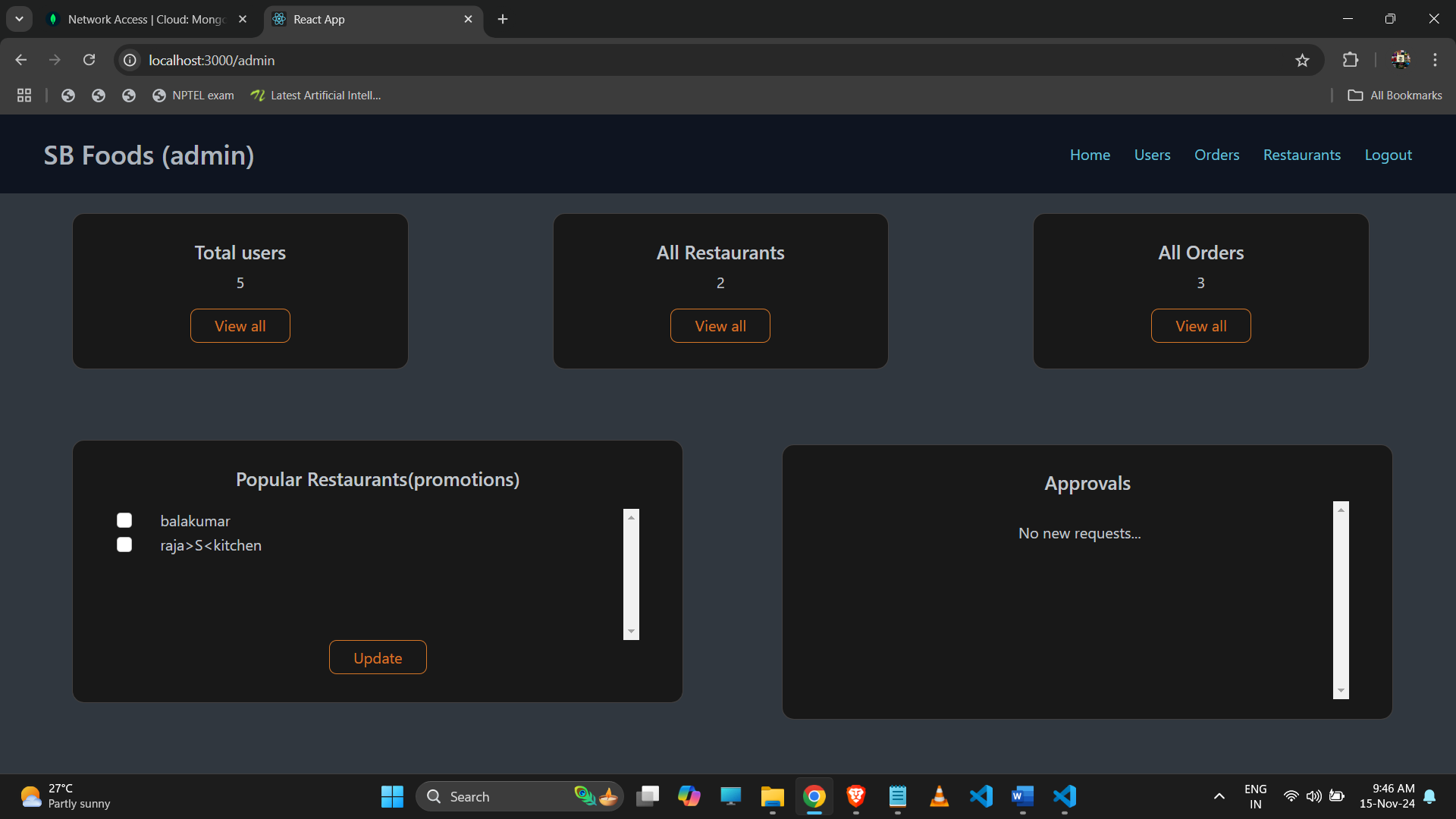
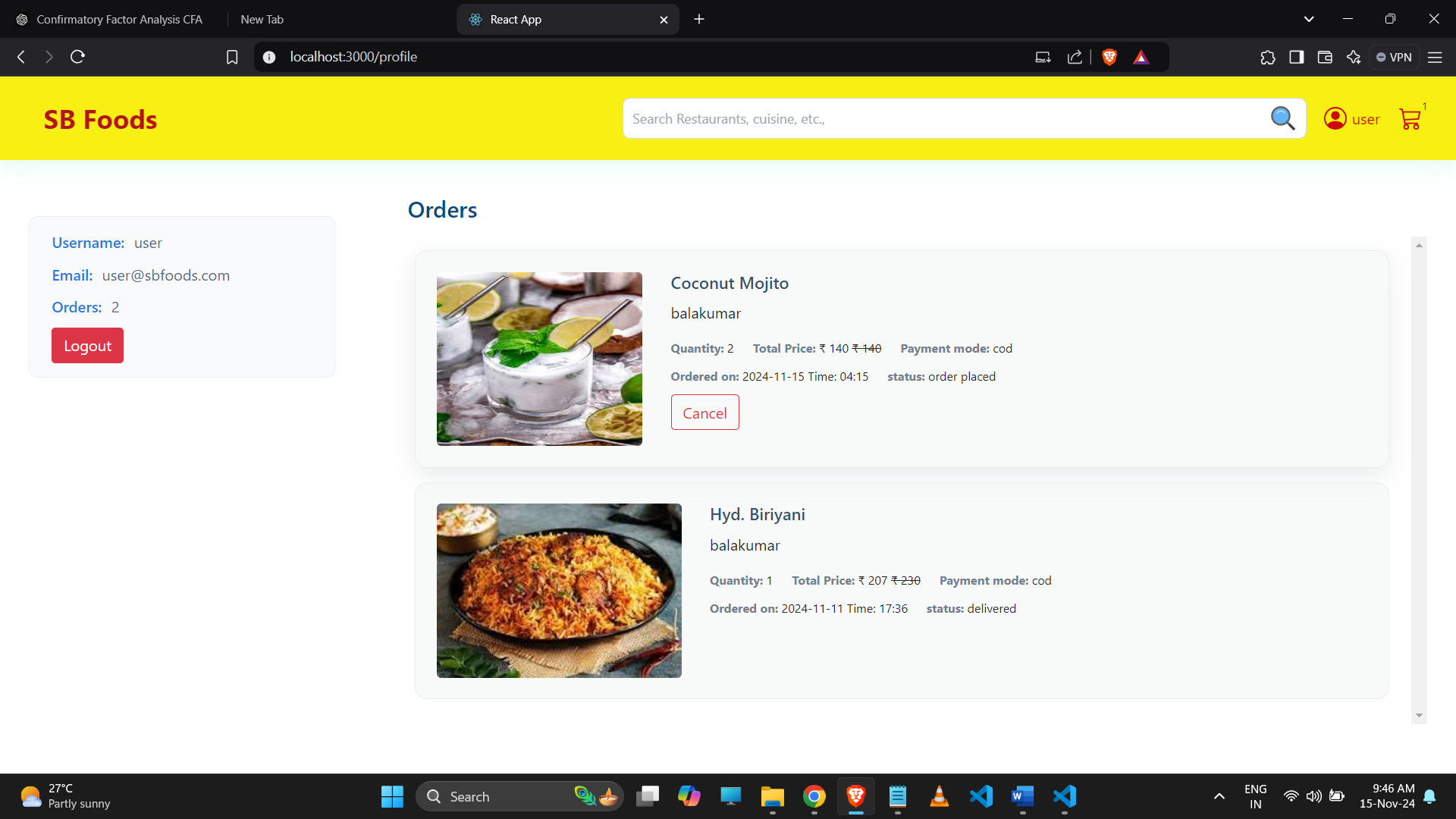
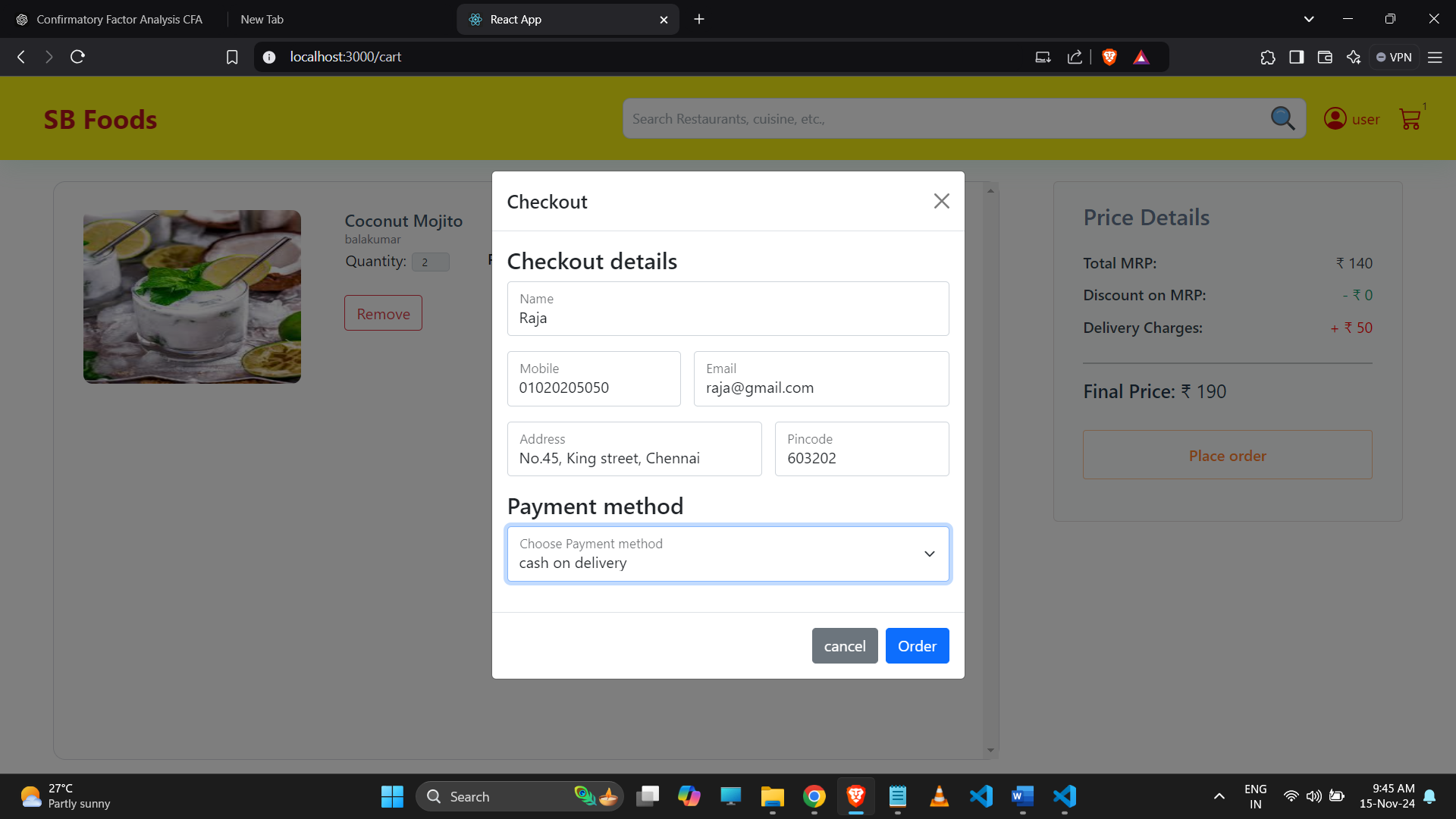
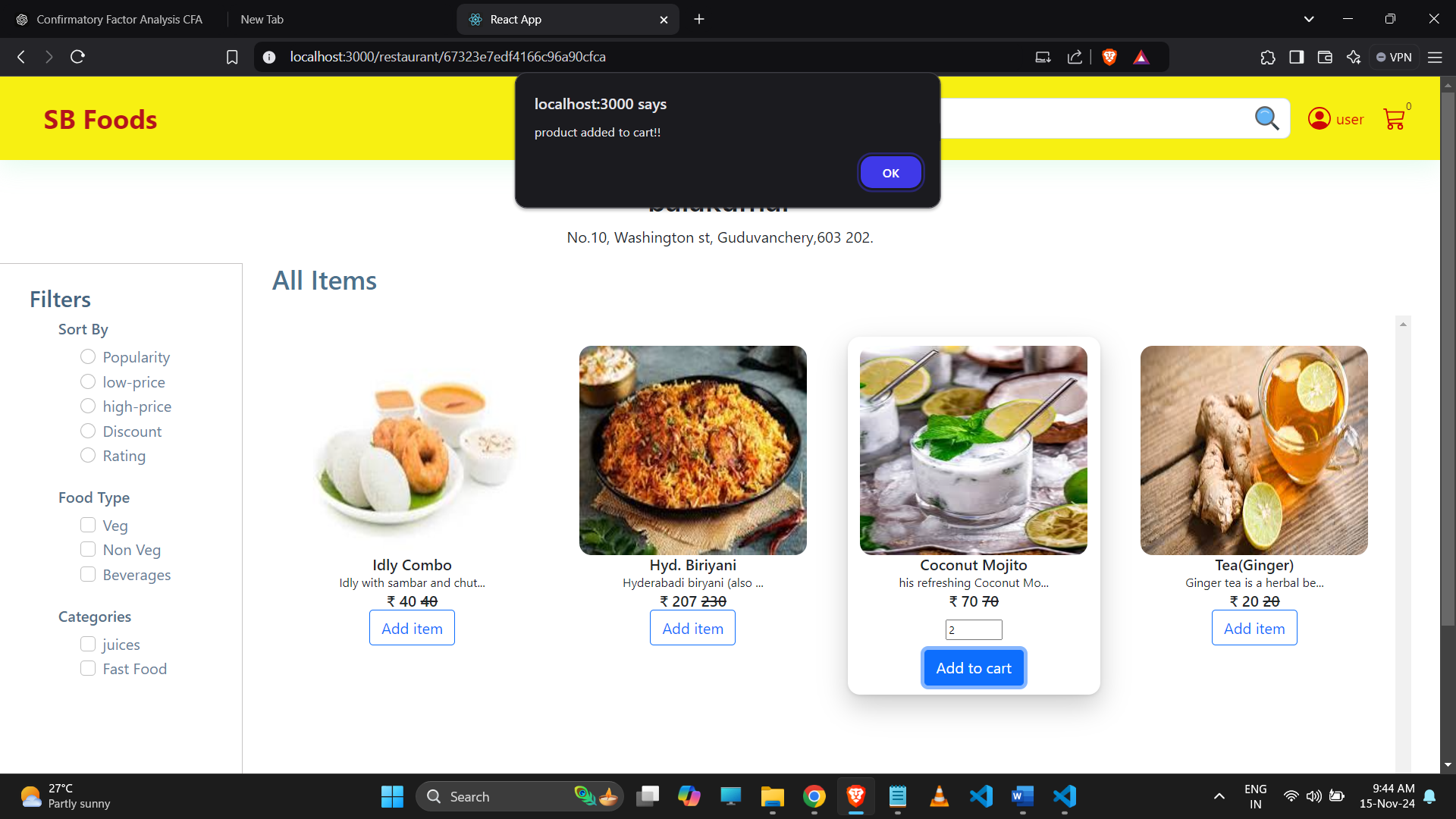
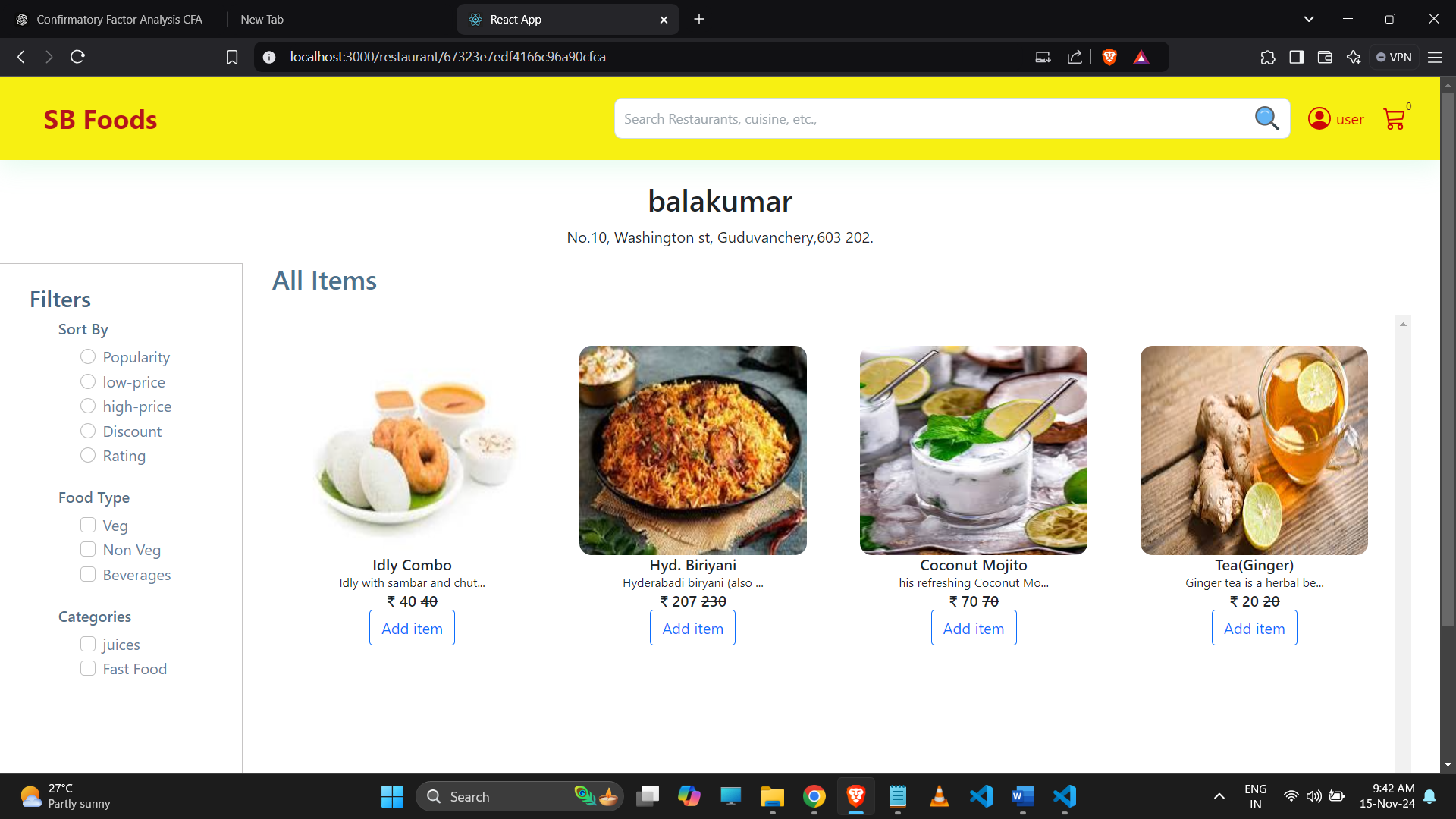
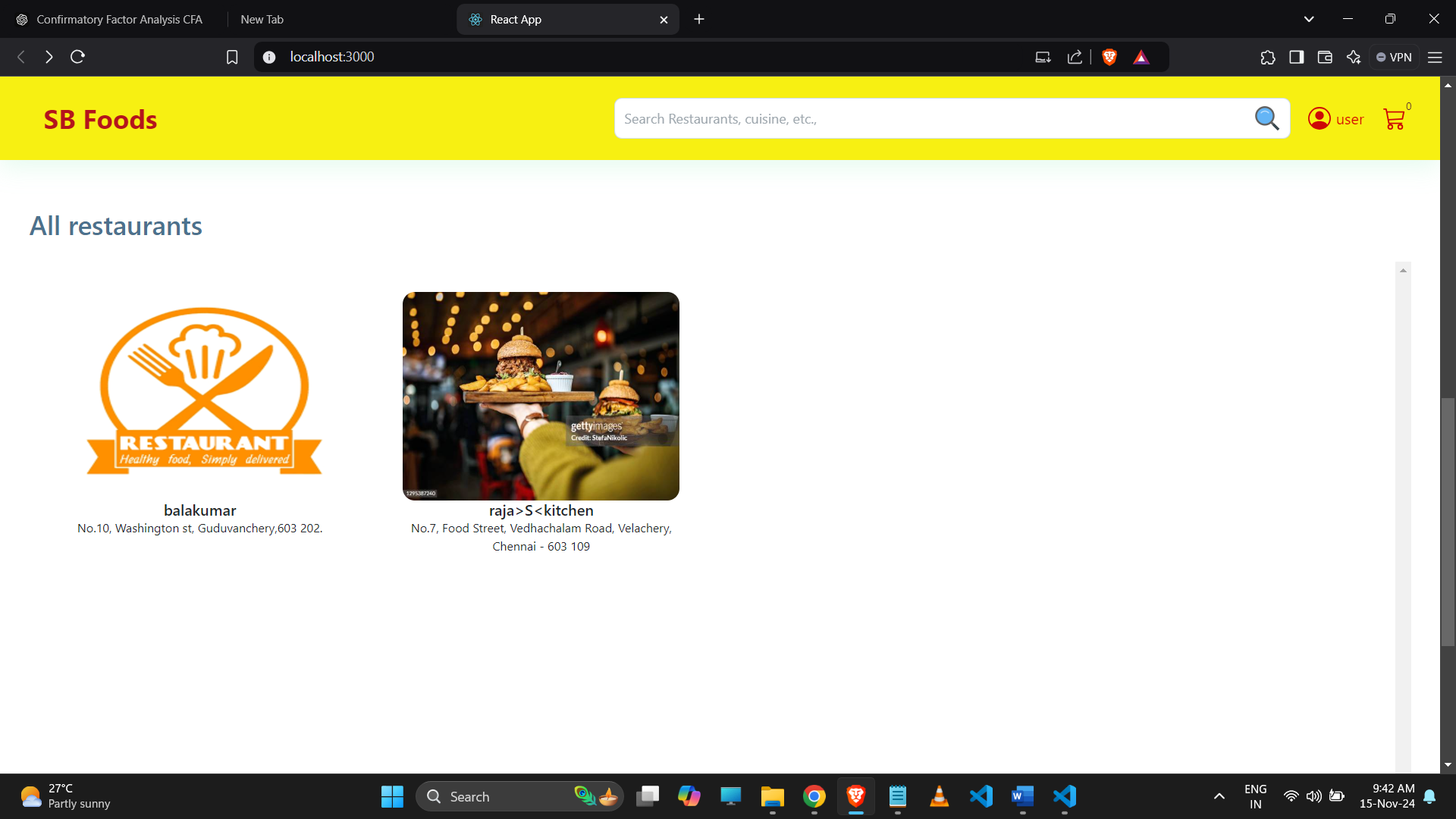
* **UI Elements**:
  + **Homepage**: Displays promotions and featured dishes.
  + **Dish Details**: Shows information like reviews, ingredients, and special offers.
  + **Order Form**: Allows users to enter delivery and payment details.
  + **Admin Dashboard**: Features for managing users, dishes, and orders.
* **Screenshots**:
  + Include screenshots or GIFs of key UI features (e.g., browsing dishes, placing an order).

### **10. Testing**

* **Testing Strategy**:
  + **Frontend Testing**: React Testing Library and Jest for unit testing components.
  + **Backend Testing**: Mocha and Chai for API endpoint testing.
  + **End-to-End Testing**: Cypress to simulate real user interactions.

### **11. Screenshots or Demo**





### **12. Known Issues**

* **Performance Delay on Initial Load**: Certain pages may load slowly due to unoptimized database queries.
* **Real-Time Notifications**: Occasionally, notifications may delay under heavy server load.
* **Responsive Design Tweaks**: Some minor design adjustments may be needed on specific screen sizes.

### **13. Future Enhancements**

* **Order Tracking**: Allow users to track the real-time location of their orders.
* **Enhanced Admin Analytics**: Add reports for better data visualization of sales and user behavior.
* **Referral Program**: Develop a referral system to attract more users.
* **AI-Based Dish Recommendations**: Implement machine learning to recommend dishes based on user preferences.