**Full Stack Development with MERN**

**Project Documentation**

**1. Introduction**

**Project Title**: SB Foods - On-Demand Food Ordering Platform

**Team ID: LTVIP2025TMID53161**

**Team Size: 4**

**Team Leader: Kunapareddy Syamala**

**Team member: Kotha Yaswanthi**

**Team member: Kotha Purandesh**

**Team member: Kothakota Likitha**

**2. Project Overview**

**Purpose:**

To provide users with a seamless digital food ordering experience while enabling restaurants and

administrators to manage their offerings and operations effectively.

**Features:**

• User registration and login

• Browse food products by restaurant

• Add to cart and checkout

• Email confirmation

• Restaurant and admin dashboards

• Order management system

**3. Architecture**

**Frontend:**

Built using React.js. Components are modular and pages are rendered dynamically based on user

authentication. Routing is handled with React Router.

**Backend:**

Node.js with Express.js is used to define RESTful APIs. Middleware handles authentication, logging,

and error-handling.

**Database:**

MongoDB Atlas stores collections for Users, Restaurants, Admins, Products, Carts, and Orders.

Mongoose ODM is used for schema definition.

**4. Setup Instructions**

**Prerequisites:**

• Node.js (v16+)

• MongoDB Atlas Account

• Git

**Installation:**

# Clone the repository

https://github.com/harsha-vardhan-reddy-07/Food-Ordering-App-MERN

# Navigate to root folder

cd Food-Ordering-App-MERN

# Install backend dependencies

npm install

# Navigate to client folder

cd client

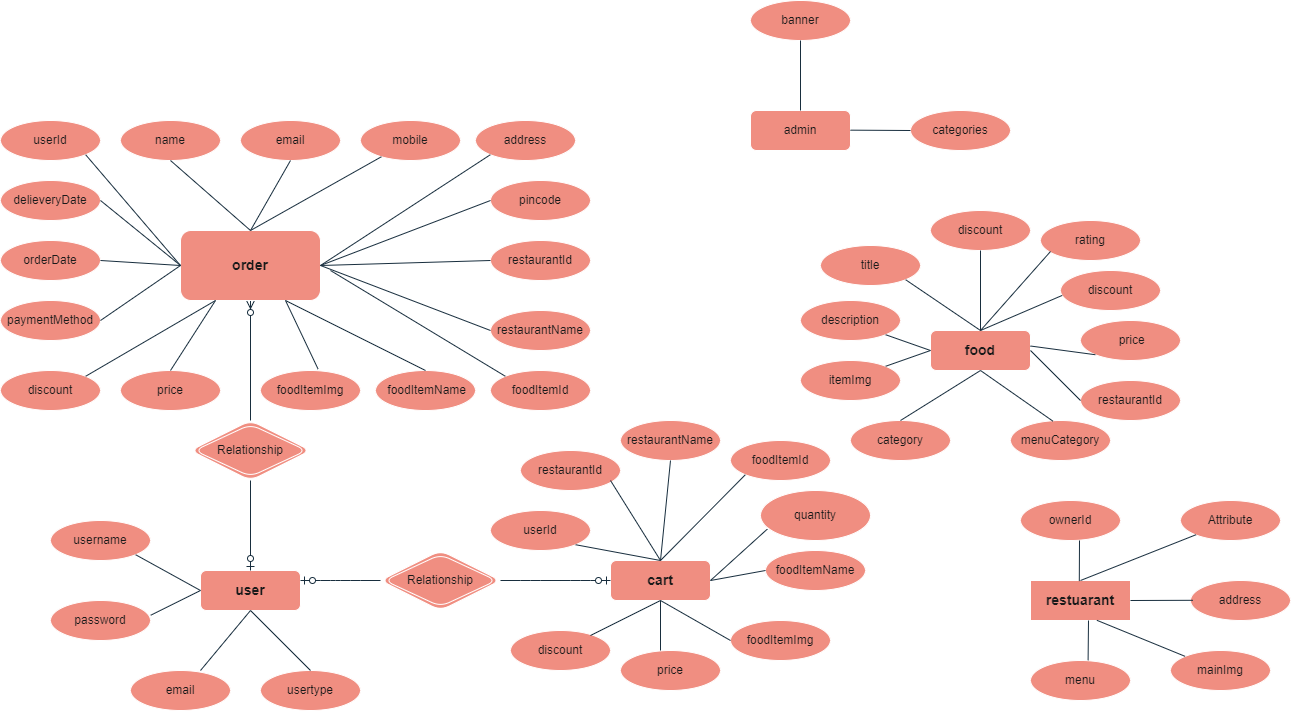
npm install

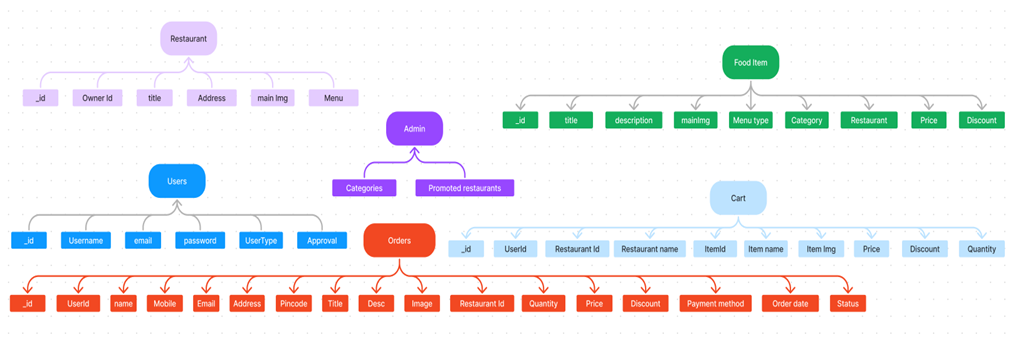
**Environment Variables:**

Create a .env file in the root directory and configure MongoDB URI, JWT secret, and PORT.

5. Folder Structure

Frontend: React.js]  
 |  
 └── Pages: Login | Menu | Cart | Checkout | Profile | Admin Dashboard  
 ↓  
[Backend: Node.js + Express]  
 |  
 ├── APIs:  
 │ ├── /api/users  
 │ ├── /api/orders  
 │ ├── /api/products  
 │ └── /api/admin  
 |  
[Database: MongoDB]  
 ├── Collections:  
 │ ├── Users  
 │ ├── Products  
 │ ├── Orders  
 │ ├── Cart  
 │ └── Admin

** **

****

**6. Running the Application**

**Frontend:**

cd client

npm start

**Backend:**

cd server

npm start

App runs at: http://localhost:3000

**7. API Documentation**

**User Routes**

• POST /api/users/register – Register user

• POST /api/users/login – Login

• GET /api/users/profile – Get user profile

**Product Routes**

• GET /api/products – Fetch all products

• POST /api/products – Add new product (admin only)

**Order Routes**

• POST /api/orders – Place an order

• GET /api/orders/:id – View specific order

**Cart Routes**

• POST /api/cart – Add to cart

• GET /api/cart/:userId – View user cart

**8. Authentication**

JWT-based authentication with tokens stored in local storage.

• Passwords are encrypted using bcrypt.

• Protected routes are guarded by middleware.

**9. User Interface**

• Registration/Login page

• Home page with food listings

• Cart and order confirmation page

• Admin and restaurant dashboards

**10. Testing**

**Testing Tools:**

• Postman for API testing

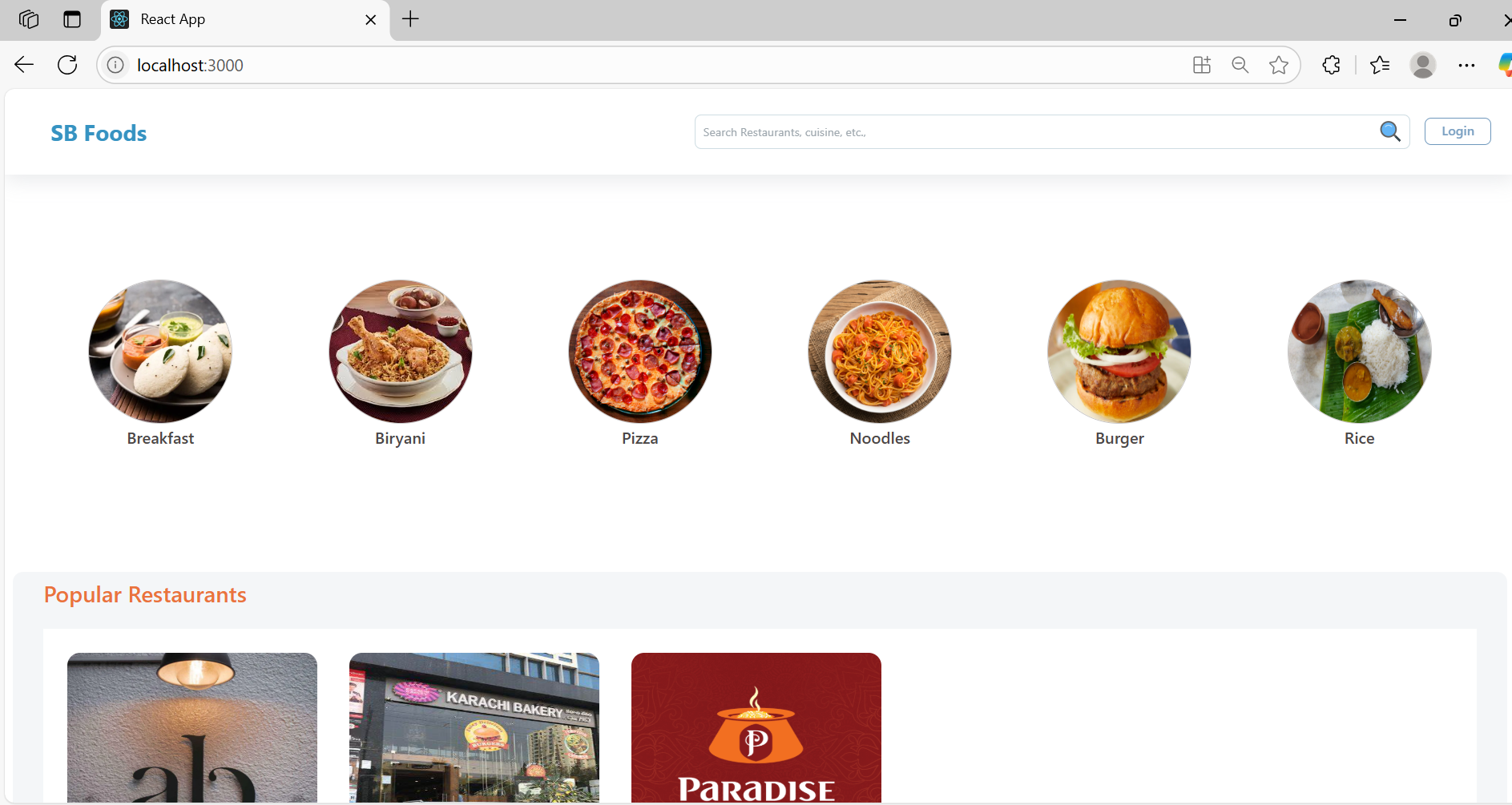
• Manual testing of UAT scenarios

• Basic unit testing with Jest (if extended)

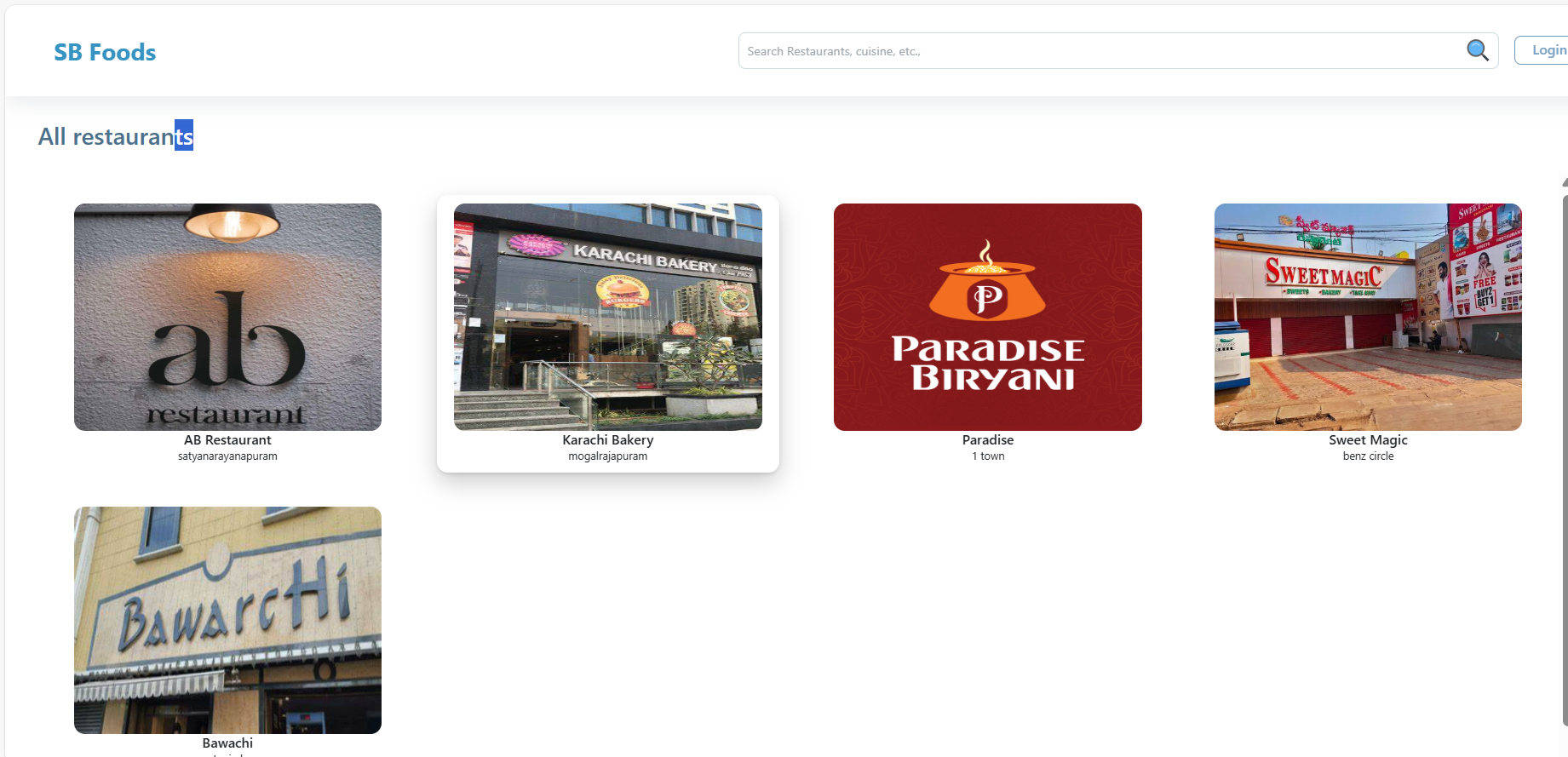
**11. Screenshots or Demo**

GitHub Repository: <https://github.com/syamalakunapareddy1/OrderOnTheGo-Your-On-Demand-Food-Ordering-Solution>

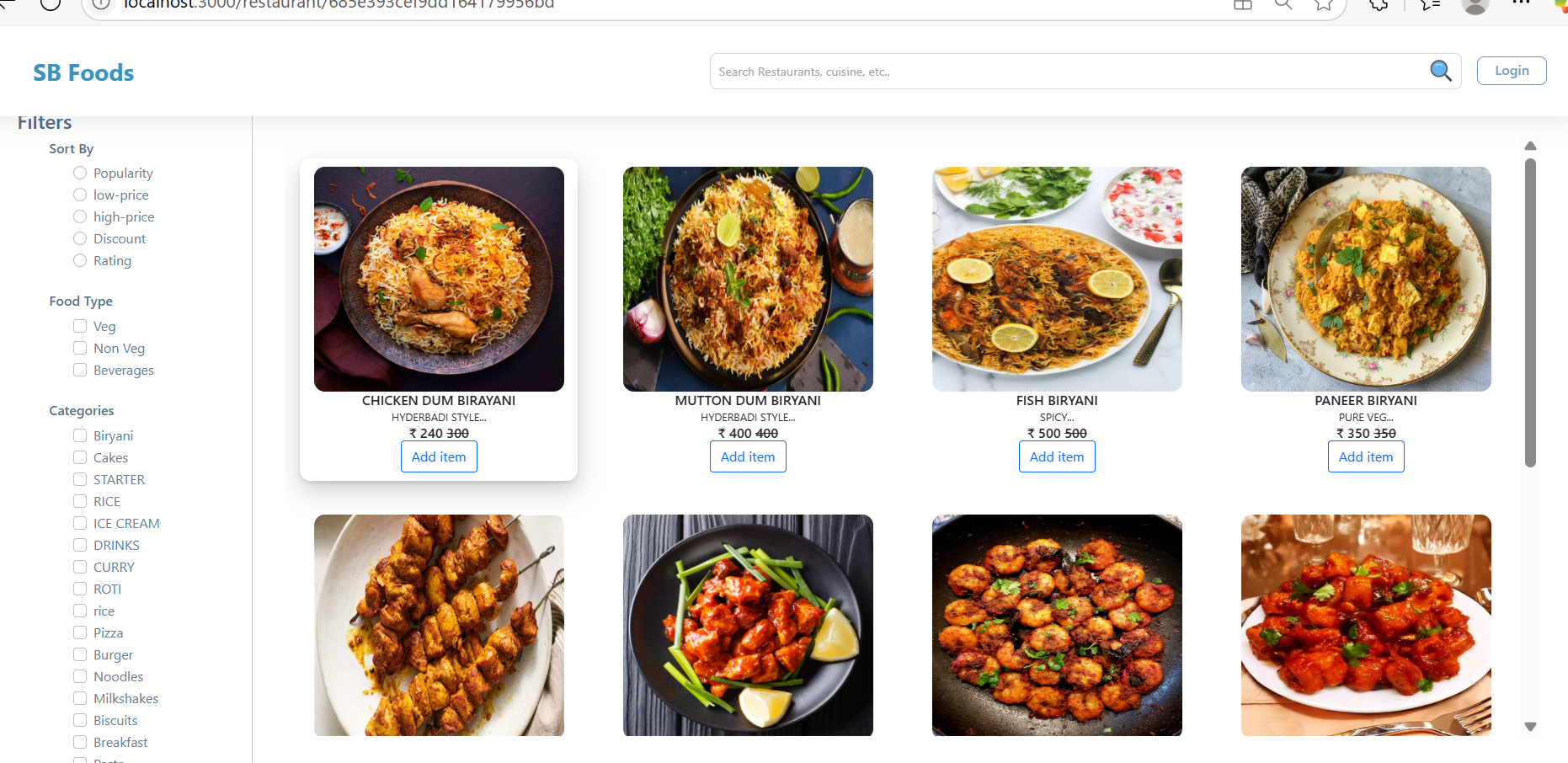
## Landing page



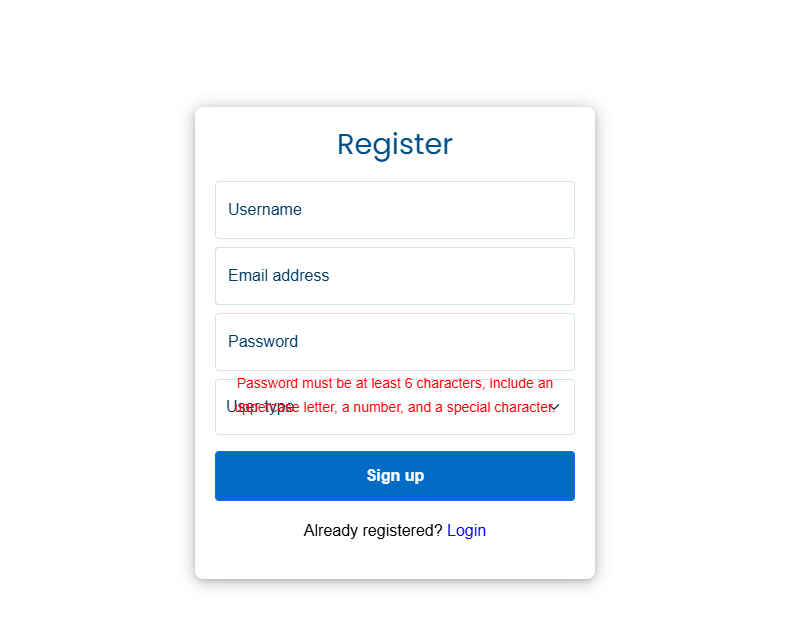
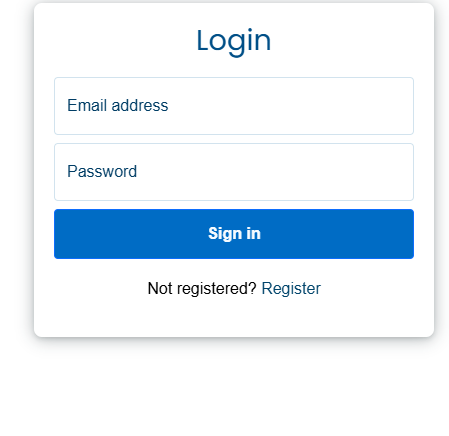
## Restaurants



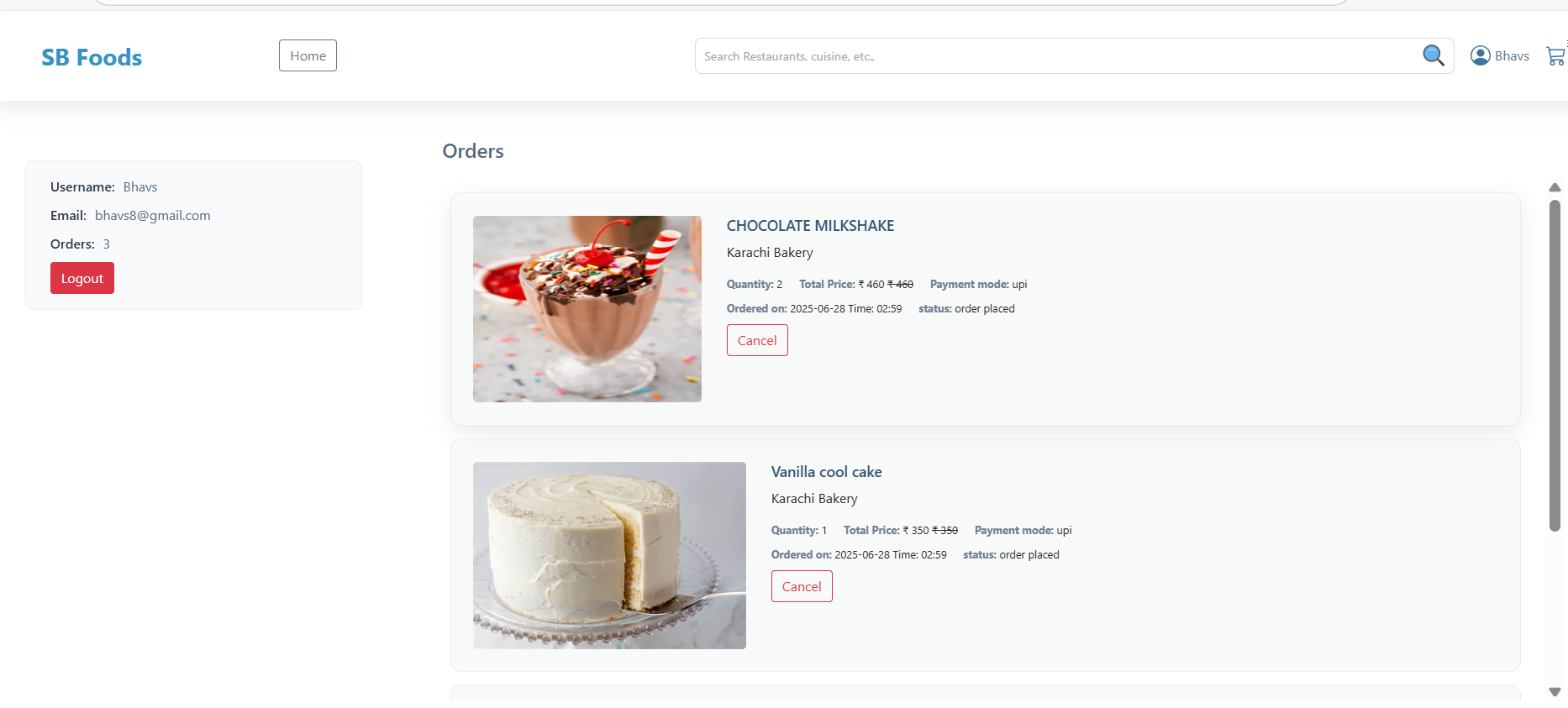
## Restaurant Menu



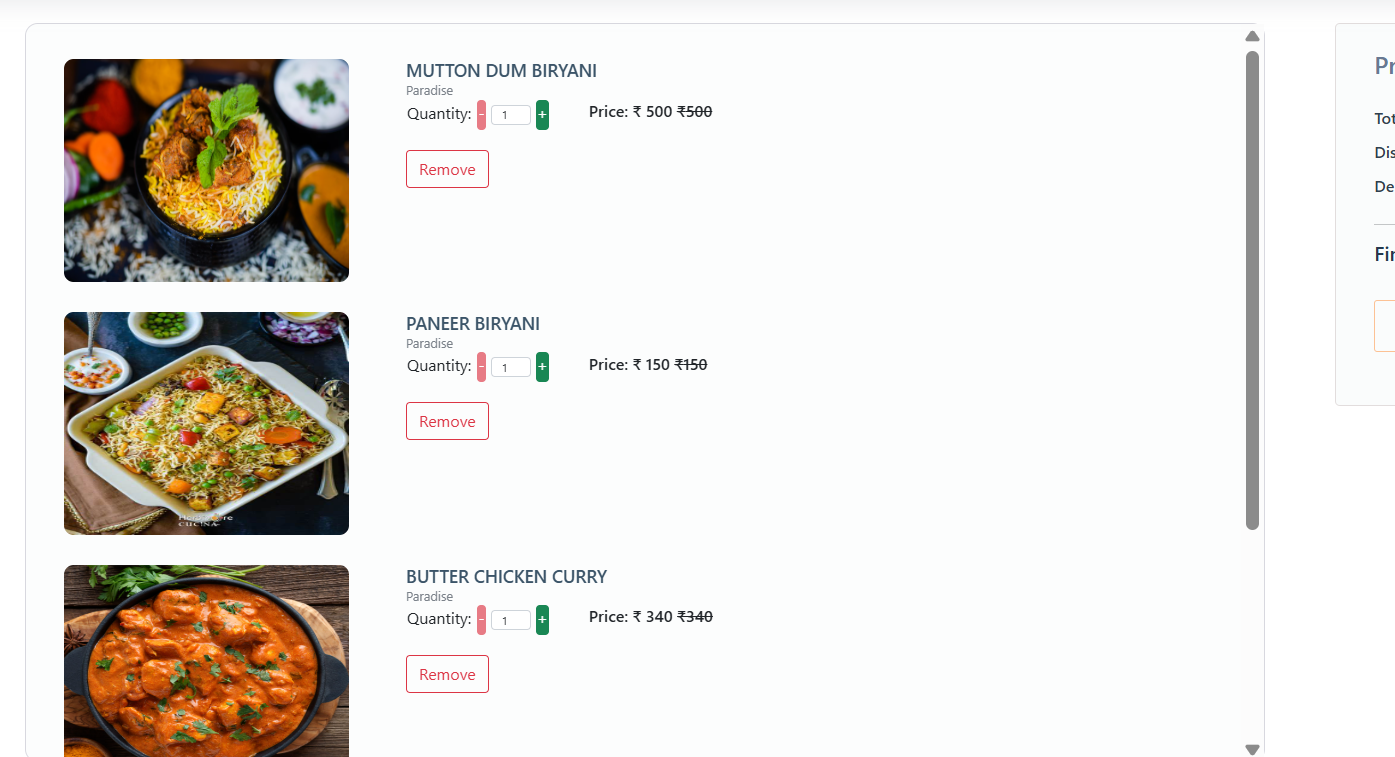
## Authentication

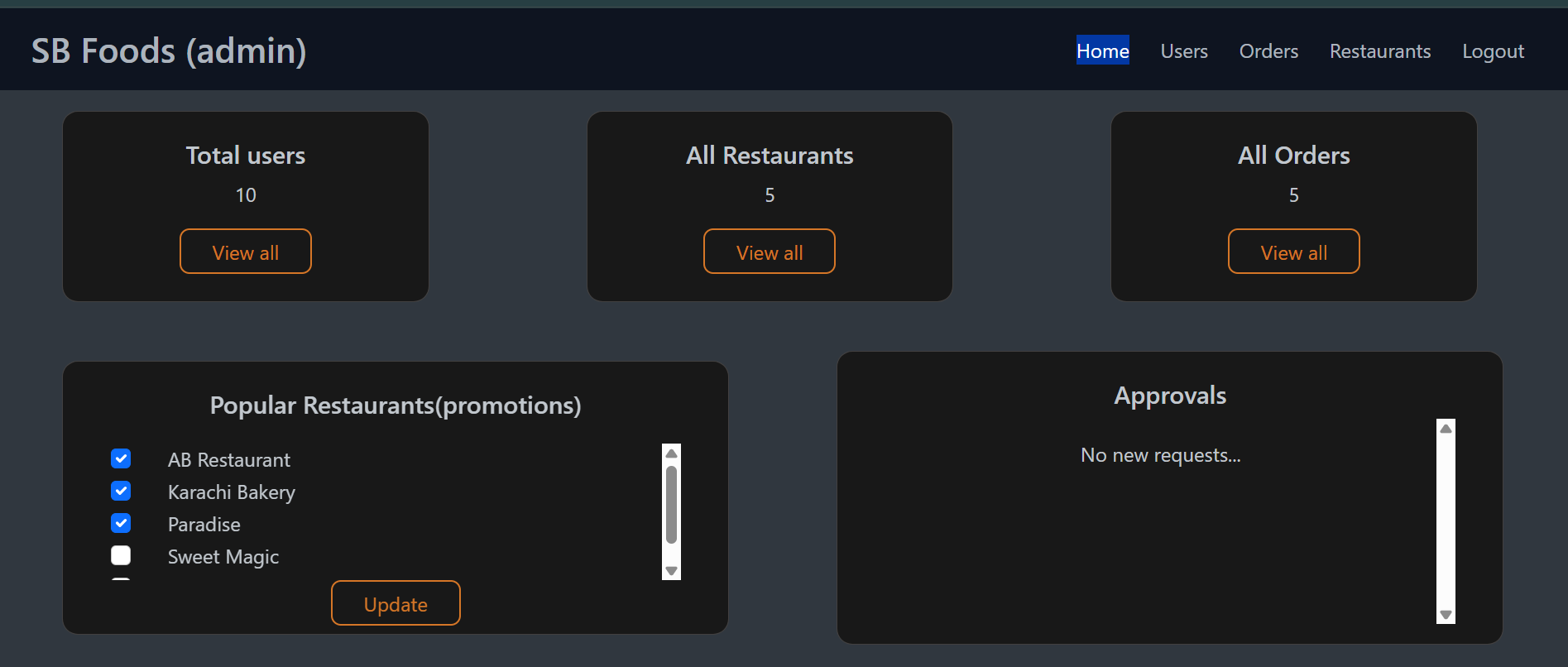
## User Profile



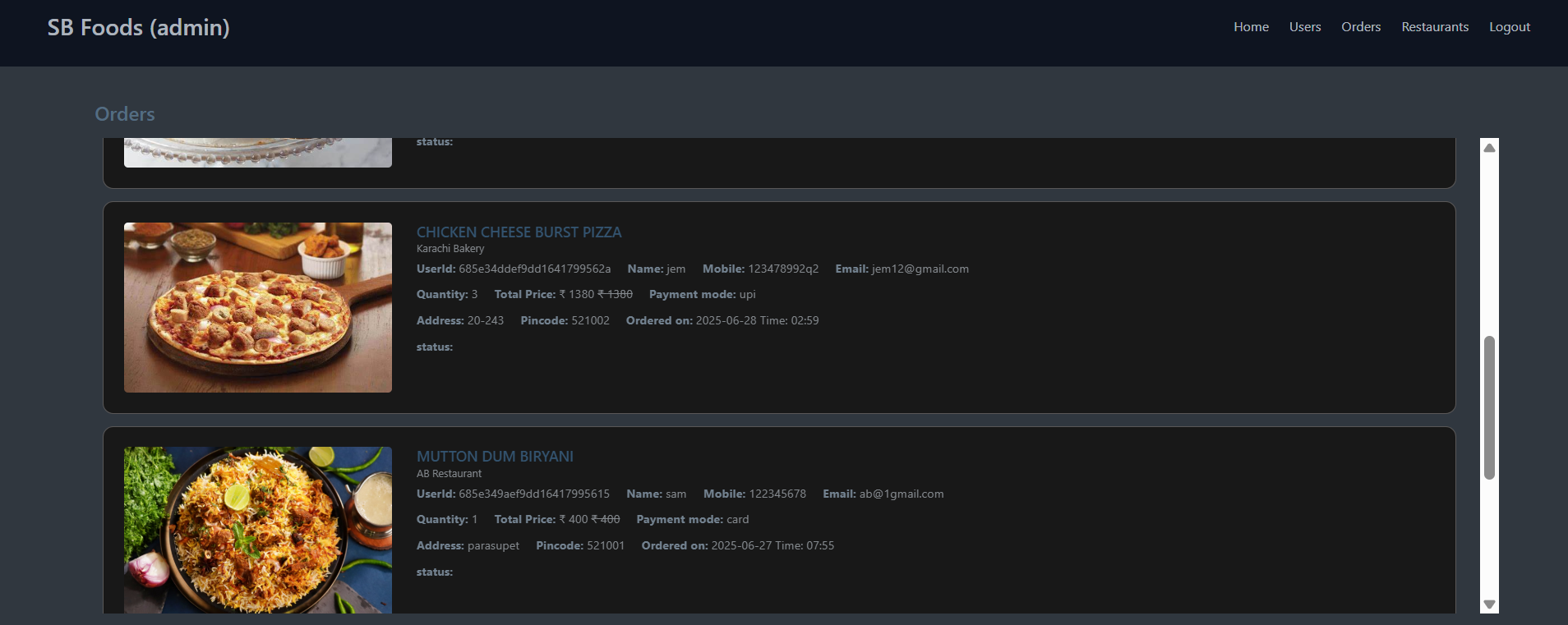
## Cart



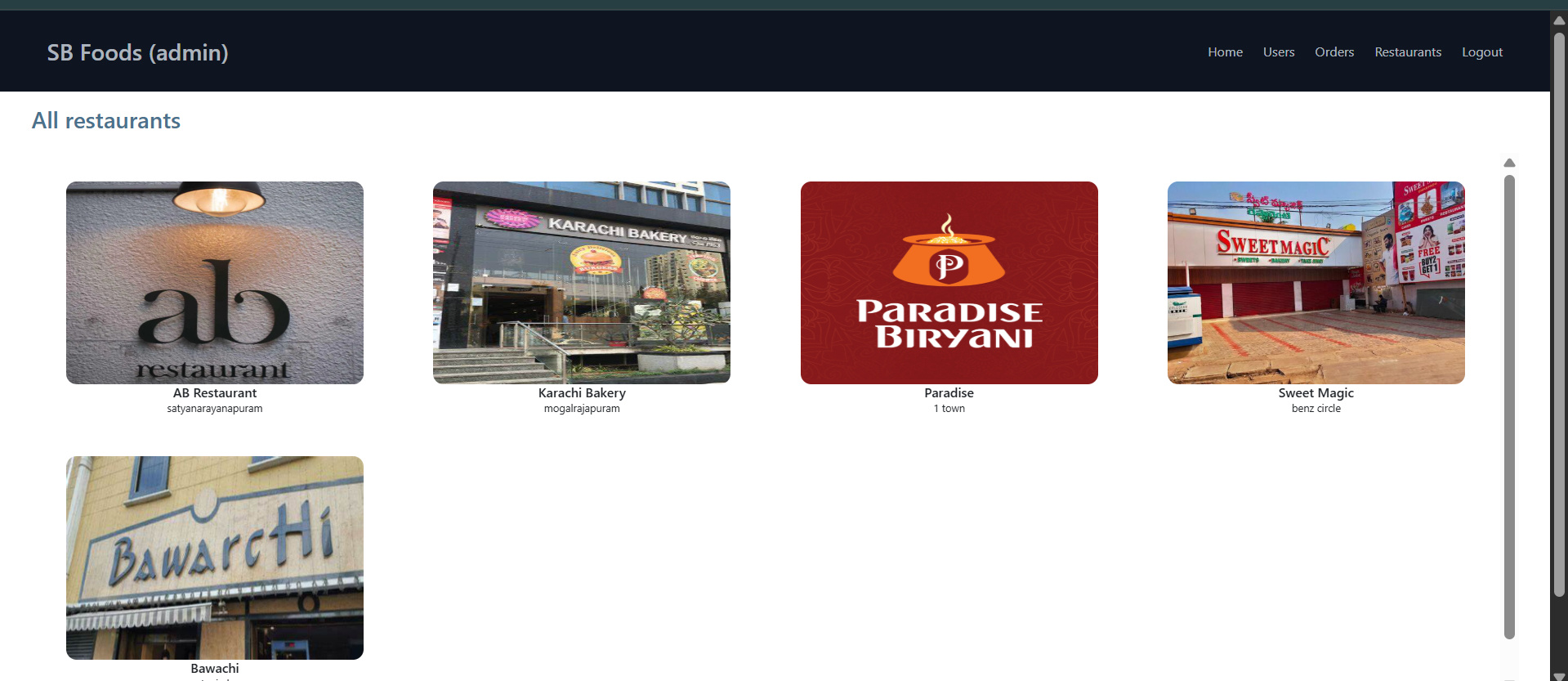
## Admin Dashboard



## All Orders



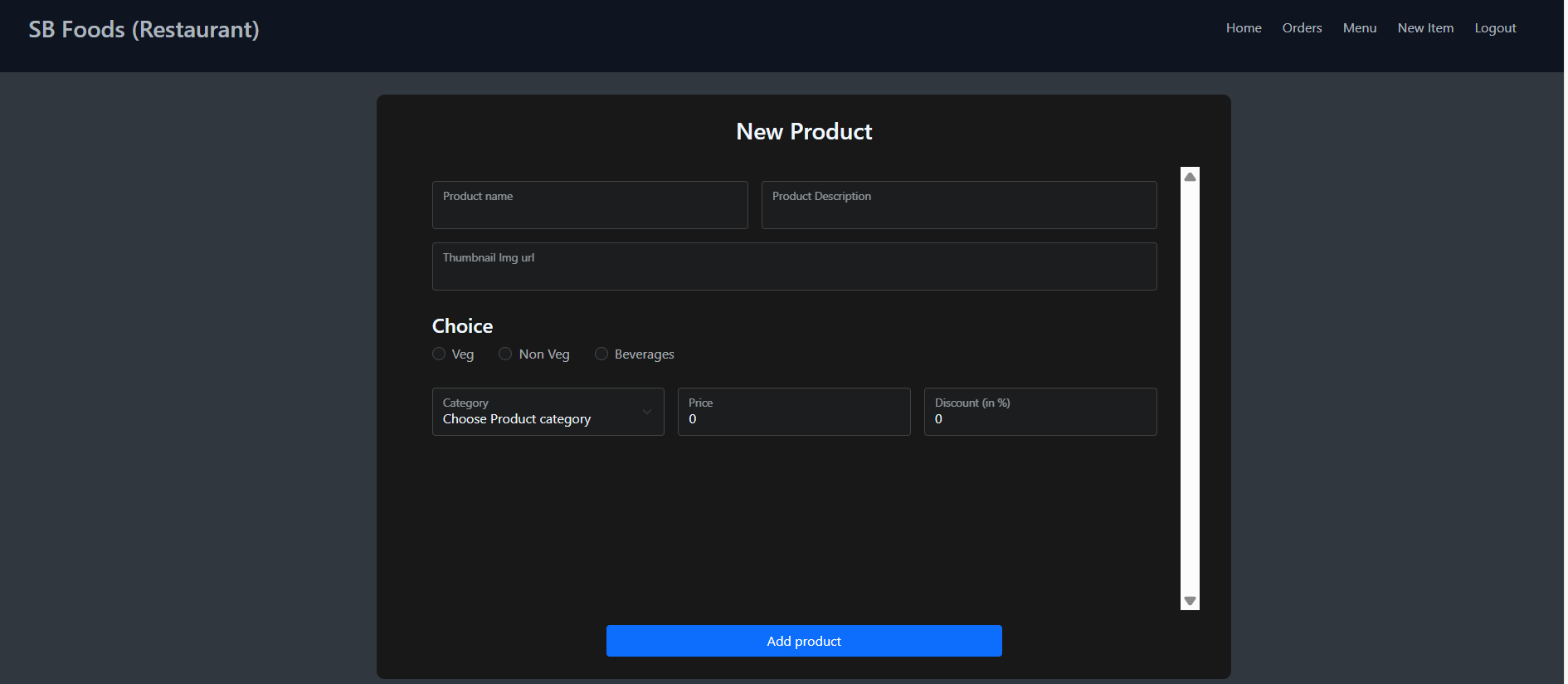
## All Restaurants



## Restaurant Dashboard

## 

## New Item



**12. Known Issues**

• No payment gateway integration

• Lack of mobile responsiveness in certain views

**13. Future Enhancements**

• Mobile app version with React Native

• AI-based food recommendations

• Wallet integration and reward system

• Multi-language support

**Appendix A: Ideation Phase – Brainstorm & Idea Prioritization**

**Team Gathering & Collaboration:**

Our team initiated the project by organizing a virtual brainstorming session to bring together diverse ideas and explore real-world problems that can be solved through technology. Each member was encouraged to share pain points they personally experienced or observed in their daily lives. The discussion focused on areas such as health, education, entertainment, and lifestyle.

**Prioritization Criteria:**

• Relevance and urgency of the problem

• Feasibility of solution within project timeline

• Technical scope and learning opportunities

• User base and social impact potential

**Selected Idea – SB Foods (Digital Food Ordering App):**

After careful analysis and ranking based on feasibility, scope, and user demand, we finalized our problem statement around late-night food cravings and inefficient food ordering systems. We selected SB Foods, a full-stack food ordering app, which allows users to explore restaurants, add items to their cart, and place orders with ease.

This idea resonated with all team members and allowed us to incorporate a variety of technical concepts including React, Node, Express, MongoDB, and JWT authentication, while ssolving a relatable real-world challenge.