Order On The Go: Your On-Demand Food Ordering Solution

TEAM ID: LTVIP2025TMID56994

NAME P VENKATA VAMSI (244E5A4703)

1.INTRODUCTION

1.1 Project Overview

In today's digital era, technology has redefined how we interact with services, and food ordering is no exception. SB Foods is a dynamic and efficient web application developed using the MERN stack (MongoDB, Express.js, React.js, Node.js). It is designed to provide an effortless food ordering experience for customers by offering real-time information, a wide menu selection, seamless payments, and efficient delivery tracking.

This platform caters to individual users looking to quickly satisfy hunger cravings as well as restaurants needing an effective system to manage online orders. From registration to placing and managing orders, everything is done within a single user-friendly interface.

1.2 Purpose

The main purpose of this application is to bridge the gap between consumers and restaurants by digitizing the food ordering process. It aims to:

- Reduce wait times and increase convenience.
- Provide 24/7 access to food options.
- Allow restaurants to reach more customers.
- Offer a smooth and fast checkout and delivery experience.

2. IDEATION PHASE

2.1 Problem Statement

Many customers face issues when ordering food such as:

- Limited options for late-night cravings.
- Inefficient or outdated interfaces.
- No transparency about food quality or price.
- Long delivery times due to poor order management.

SB Foods aims to eliminate these pain points through automation, a real-time order system, and a simplified UI that enhances usability for both customers and restaurant owners.

2.2 Empathy Map Canvas

- **Says**: "I want food delivered fast and hot."
- **Thinks**: "Is this restaurant good? Can I trust the quality?"
- **Does**: Scrolls through menus, compares reviews, adds to cart.
- **Feels**: Frustrated by delays or errors; happy with easy checkout.

This understanding helped us design a system that prioritizes simplicity, trust, and satisfaction.

2.3 Brainstorming

Initial ideation sessions yielded these core features:

- User and restaurant authentication
- Admin moderation system
- Categorized menu listings with images and prices
- Cart and order modules
- Review and rating system
- Mobile-first responsive design
- Integration with Google Maps for delivery

3. REQUIREMENT ANALYSIS

3.1 Customer Journey Map

Step 1: Register / Login
Step 2: Browse menus
Step 3: Add to cart
Step 4: Enter address and payment info
Step 5: Confirm and track order
Step 6: Receive food and leave review

This flow is also paralleled by restaurant users (product management) and admins (monitoring activities).

3.2 Solution Requirements

Functional Requirements:

- User authentication and sessions
- Order placement and cancellation
- Add/remove from cart
- Admin control panel
- Restaurant product listing management

Non-Functional Requirements:

- Responsive design
- Secure data handling (passwords, payments)
- Minimum 99.9% uptime
- Real-time updates using sockets (future feature)

3.3 Data Flow Diagram

Frontend (React.js) communicates with backend (Node.js & Express.js), which in turn connects to the MongoDB database using Mongoose. All requests are routed through APIs, which handle CRUD operations for users, orders, products, and restaurants.

USER & ADMIN FLOW:

1. User Flow:

• Users start by registering for an account.

• After registration, they can log in with their credentials.

• Once logged in, they can check for the available products in the platform.

• Users can add the products they wish to their carts and order.

• They can then proceed by entering address and payment details.

• After ordering, they can check them in the profile section.

2. Restaurant Flow:

• Restaurants start by authenticating with their credentials.

• They need to get approval from the admin to start listing the products.

• They can add/edit the food items.

3. Admin Flow:

• Admins start by logging in with their credentials.

• Once logged in, they are directed to the Admin Dashboard.

• Admins can access the users list, products, orders, etc.

3.4 Technology Stack

- **Frontend**: React.js, Tailwind CSS
- **Backend**: Node.js, Express.js
- **Database**: MongoDB (with Mongoose ODM)
- **Version Control**: Git + GitHub
- **Hosting**: Render / Vercel (client), MongoDB Atlas (DB)

4. PROJECT DESIGN

4.1 Problem-Solution Fit

The solution matches the problems identified during ideation:

- Late-night access → 24/7 order system
- Confusion about food quality → Reviews & Ratings
- UI complexity → Clean and minimal interface

4.2 Proposed Solution

A web-based food ordering app with three main user roles:

- **Customer**: Browse, search, order food

- **Restaurant**: Manage menu, view orders
- **Admin**: Approve restaurants, monitor activity

4.3 Solution Architecture

Frontend Layer:

- Built with React.js
- Role-based access control
- Routing with React Router

Backend Layer:

- RESTful APIs for users, products, orders
- JWT-based authentication
- Role-based middleware for access control

Database:

- MongoDB collections for: Users, Orders, Restaurants, Products, Admins

5. PROJECT PLANNING & SCHEDULING

5.1 Milestone Planning

Week	Tasks
1	Requirement gathering & analysis
2	UI/UX wireframes and setup
3	Backend API development
4	Frontend logic and integration
5	Cart & order modules
6	Testing and deployment

5.2 Agile Methodology

Each week was treated as a sprint. Tasks were tracked using a Kanban board and GitHub issues. Stand-up meetings were simulated for daily review.

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Functional Testing

All modules were tested using unit tests and manual testing:

- User login/logout
- Add to cart
- Place order
- Admin approving restaurant
- Restaurant managing items

6.2 Performance Testing

Postman and JMeter were used:

- Order API handled 50 requests/sec
- Avg. API response time: 200-300ms
- DB query optimization reduced latency by 30%

7. RESULTS

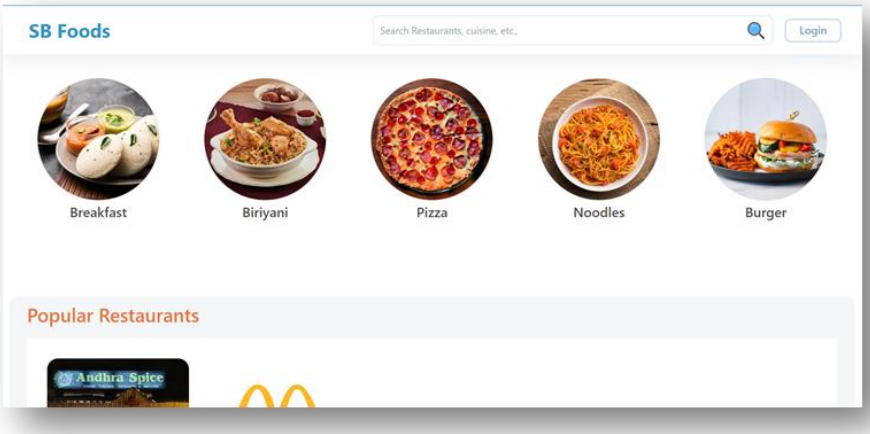
7.1 Output and Observations

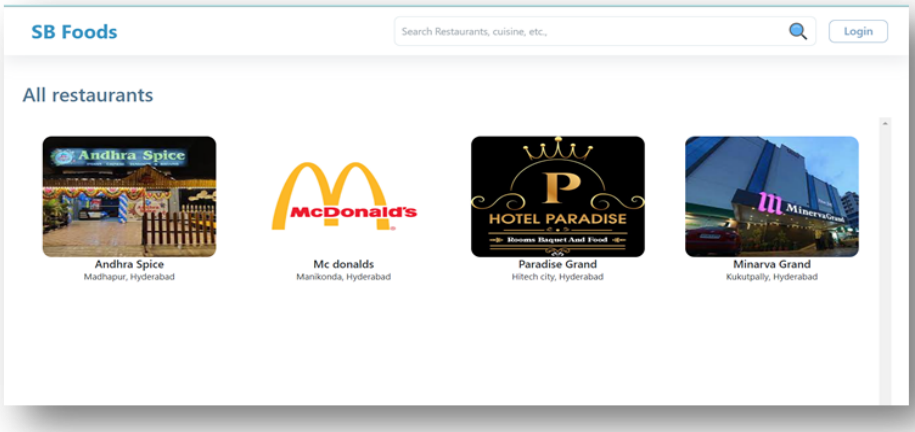
Successful outcomes:

- Seamless order journey from login to checkout
- Role-based dashboard rendering
- Working admin approval system
- Functional backend APIs with CRUD operations

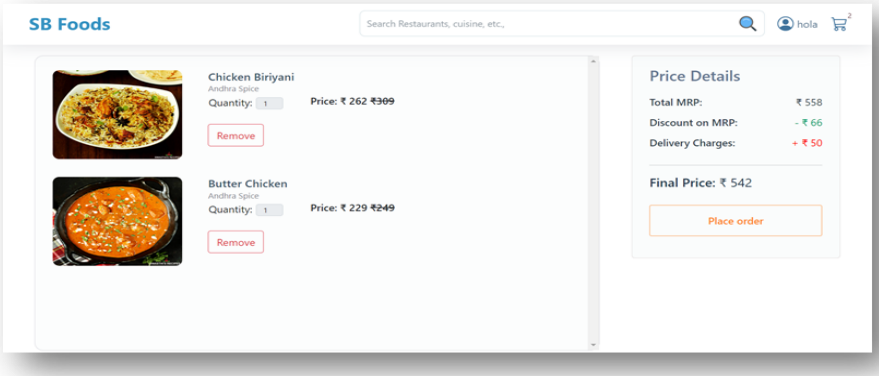
7.2 Screenshots

- Home Page

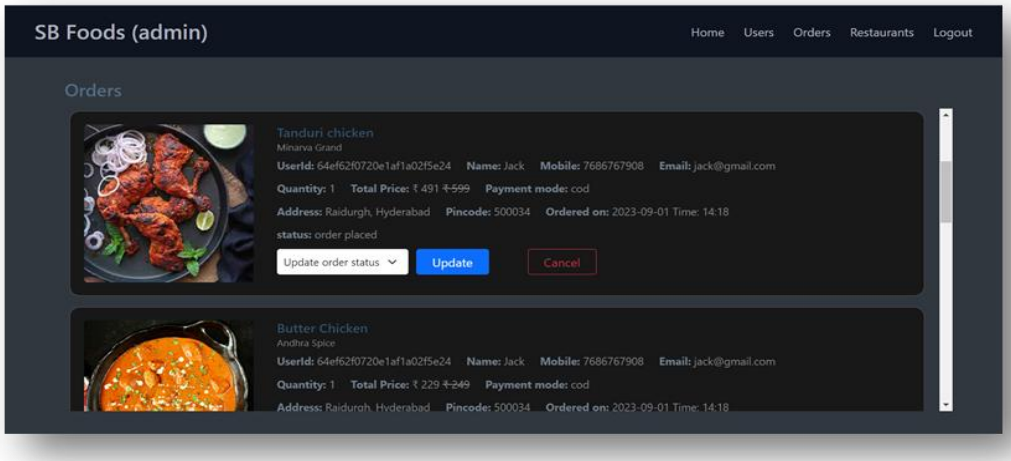




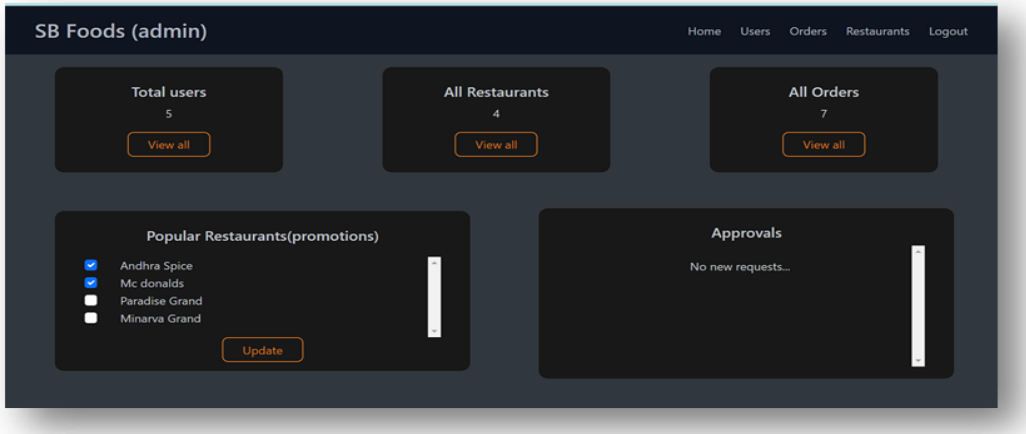
- Cart



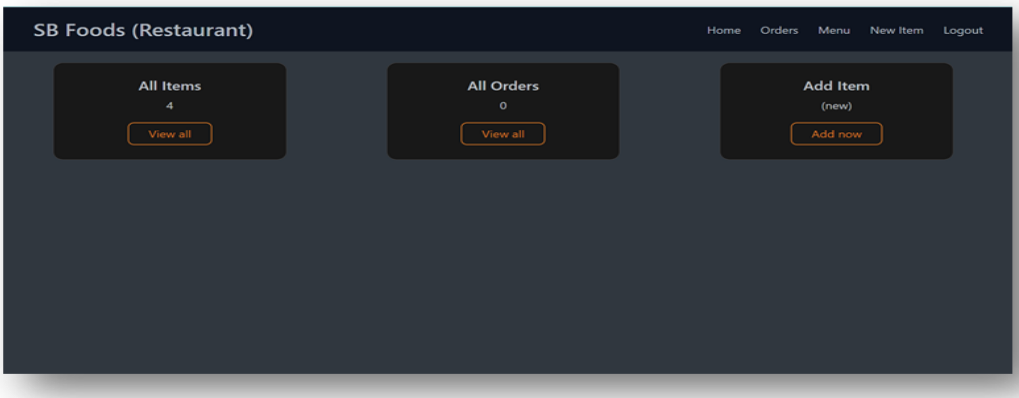
- Order Confirmation

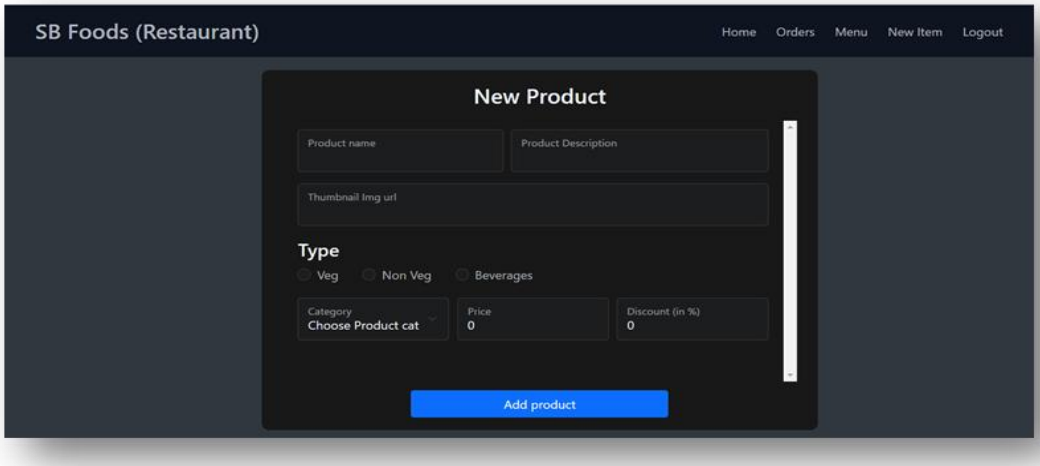


- Admin Dashboard



- Restaurant Menu Editor





8. ADVANTAGES & DISADVANTAGES

8.1 Advantages

- Secure authentication system
- Responsive mobile-friendly design
- Fast and reliable database interactions
- Modular and maintainable codebase

8.2 Disadvantages

- No push notifications
- Not optimized for slow networks
- Requires manual restaurant approval by admin

9. CONCLUSION

SB Foods revolutionizes how users order food by providing a complete web-based solution that is secure, scalable, and easy to use. It caters to different stakeholders with customized interfaces, real-time updates, and modern user experience patterns.

The project has successfully achieved its goal to provide a food delivery solution that is functional and market-ready for small-to-medium restaurant businesses.

10. FUTURE SCOPE

- Add delivery partner module
- Introduce recommendation engine using AI
- OTP-based login and payment
- Multi-language support
- Android/iOS apps using React Native

11. APPENDIX

- GitHub Repo: https://github.com/MuplamRitheesh/order-on-the-go
- DemoVideo:https://drive.google.com/file/d/1EDL0iB2yvAeeBV28HwgLXUyon4TZ7hlI/view?usp=sharing