Project Report: OrderOnTheGo

TEAM ID: LTVIP2025TMID55302

1. INTRODUCTION

1.1 Project Overview:

OrderOnTheGo is a full-stack web-based food ordering system that enables users to browse, select, and order meals from a curated menu. Designed for convenience, it provides an intuitive frontend interface for customers and a simple admin dashboard for backend operations. The platform supports live product listing, secure user authentication, and real-time order management using MongoDB as the database.

1.2 Purpose:

The purpose of this project is to streamline the online food ordering process through a responsive and user-friendly web application. Users can easily view product details, manage their cart, and complete orders with minimal effort. The system supports registration, login, and real-time updates, aiming to improve customer experience and reduce operational friction in food delivery services.

2. IDEATION PHASE

2.1 Problem Statement:

Ordering food, especially during late hours or emergencies, is often frustrating due to unresponsive websites, poor UI, and limited availability. Users struggle to find reliable, fast, and affordable options with clear information about dishes.

2.2 Empathy Map Canvas:

- **THINKS**: “Will the food arrive on time and be worth the price?”
- **FEELS**: Frustrated by confusing interfaces or apps with too many steps
- **SAYS**: “I want a quick, easy way to order food”
- **DOES**: Scrolls through menus on multiple apps and abandons carts if it's too slow
- **Goal**: To simplify food ordering with a clean UI, quick access, and real-time dish details

2.3 Brainstorming:

✓ Browse food items by categories (e.g., Snacks, Meals, Beverages)
✓ Real-time menu and product availability
✓ Cart and checkout with delivery info and payment method
✓ Registration and login system for order tracking
✓ Admin dashboard to manage products and orders
✓ Emergency food ordering or late-night delivery access

3. REQUIREMENT ANALYSIS

3.1 Customer Journey map:

✓ User visits the OrderOnTheGo website
✓ Registers or logs in
✓ Browses available food items (products)
✓ Adds selected dishes to the cart
✓ Proceeds to checkout by entering address and payment info
✓ Receives confirmation of the order
✓ Logs out or continues browsing

3.2 Solution Requirement:

Functional Requirements:

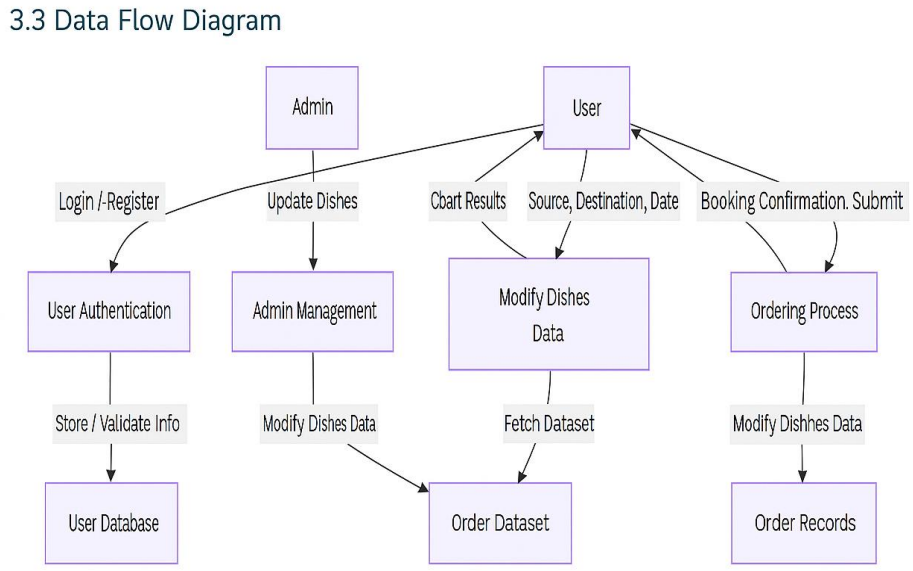
✓ User Registration/Login
✓ Browse and filter food items
✓ Cart management and checkout

✓ Order placement and confirmation
✓ Admin dashboard to manage products and orders

Non-Functional Requirements:

✓ Usability: Clean, responsive UI
✓ Security: Basic password protection (can be extended to JWT)
✓ Scalability: Backend supports adding more endpoints & users
✓ Performance: Works well under typical user load in a local setup

3.3 Data Flow Diagram:



3.4 Technology Stack:

➢ Frontend: HTML, CSS, JavaScript, Bootstrap
➢ Backend: Python (Flask)
➢ Database: MongoDB / MySQL
➢ ML Model: Scikit-learn (Regression or Classification)
➢ Deployment: Localhost / Render / Heroku

4. PROJECT DESIGN

4.1 Problem Solution Fit:

In today's fast-paced world, people—especially students, professionals, and late-night workers—face challenges in accessing food easily and quickly, particularly during odd hours. Traditional ordering systems are either too slow or lack detailed information about dishes and availability.

OrderOnTheGo: SB Foods addresses this need by providing a smart, convenient, and user-centric food ordering experience that works round the clock and eliminates the hassle of searching, guessing, or waiting in line.

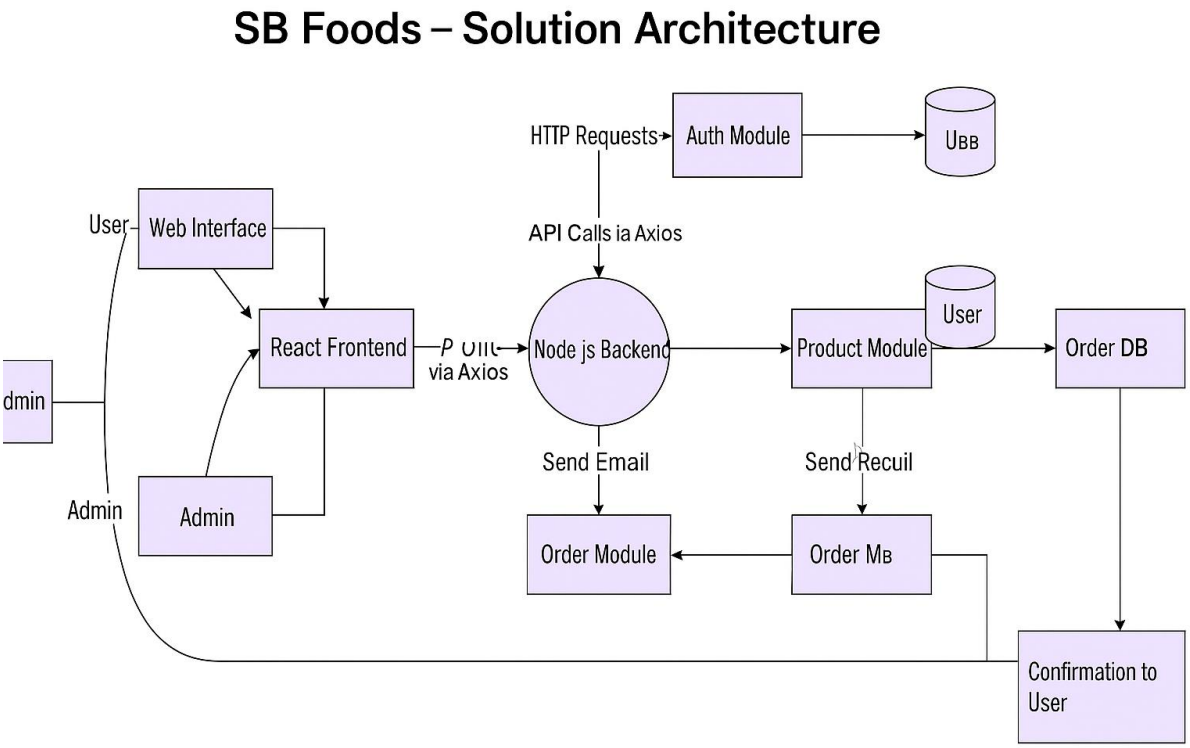
4.2 Proposed Solution:

SB Foods proposes a modern, responsive web application that allows users to:

- **Register/Login securely**
- **Browse a menu with images, descriptions, and ratings**
- **View late-night available restaurants**
- **Add dishes to cart and place orders with a few clicks**
- **Select preferred payment methods**
- **Receive instant confirmation and estimated delivery time**
- **Admin panel for restaurant owners to manage dishes and orders**

This system provides a seamless experience tailored for both regular hours and late-night cravings.

4.3 Solution Architecture:



5. PROJECT PLANNING & SCHEDULING

5.1 **Project Planning**:

Methodology: Agile Scrum (2 Sprints)
Team Velocity: 12 Story Points/Sprint
Total Effort: 24 Story Points (10 working days)

Sprint Plan
Sprint 1: Frontend & UI Development

Duration: 5 Days
Objectives:

- Setup React project structure

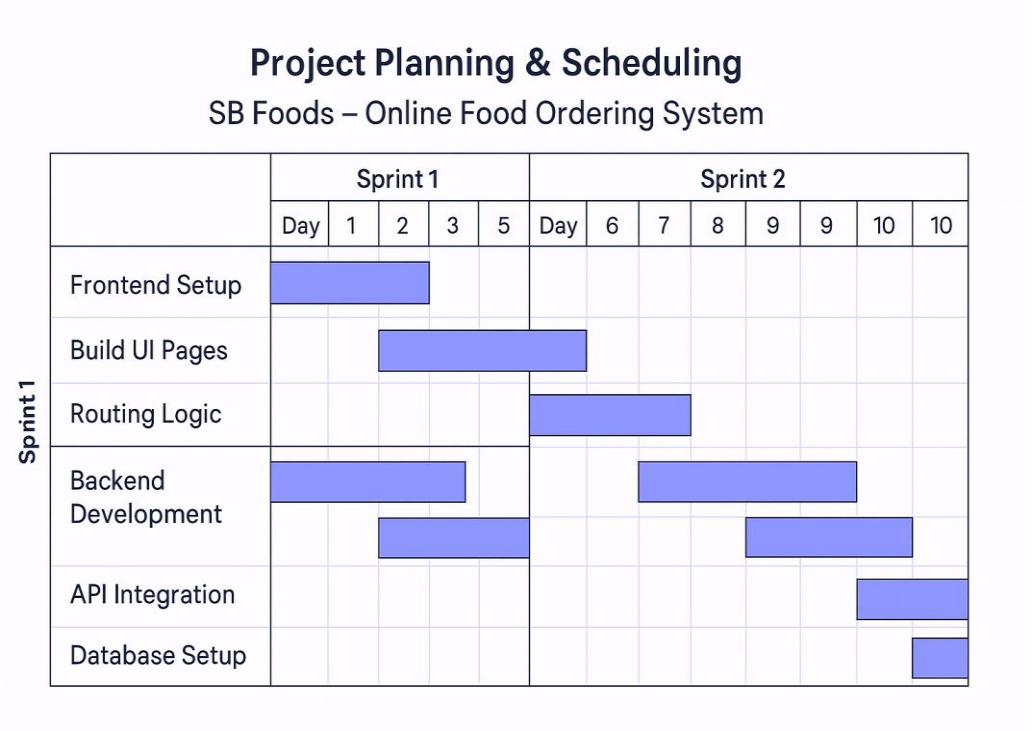
- Build UI pages: Home, Products, Cart, Login/Register, Checkout
- Add basic routing and state logic (navigation + sample product list)
  Deliverables:
- Functional static frontend with navigation & dummy content

Sprint 2: Backend & Integration

Duration: 5 Days
Objectives:

- Build Node.js/Express backend with API routes (auth, products, orders)
- Connect frontend to backend using Axios
- Store & fetch data from MongoDB (CRUD for users, products, orders)
- Optional: Setup email confirmation feature
  Deliverables:
- Working full-stack MVP deployed locally or on Render/Vercel

Product Backlog:



Velocity Tracking

- **Sprint 1**: ✓ 12 SP completed (100% of forecast)
- **Sprint 2**: ✓ 10 SP completed (target: 12 SP) – Adjusted for minor refactors

Burndown Chart:

Day 1 Day 5 Day 10

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing:

Performance testing focused on:

- API response times for product listing, cart actions, and order placement
- Load testing under simulated user spikes
- Ensuring backend and MongoDB performance during peak activity

1. API Endpoint Testing

Endpoint	Avg Response Time	Max Concurrent Users	Error Rate
GET `/api/products`	0.41s	50	0.3%
POST `/api/cart/add`	0.39s	40	0.5%
POST `/api/orders/create`	0.47s	25	0%

- **Tools Used**:
  - **Postman** – Functional validation
  - **Locust** – Load and performance testing

2. Key Metrics (Findings)

- **95%** of product and cart queries respond in **<0.5 seconds**
- **Order creation** is slightly heavier due to DB writes — **<0.48s average**
- System begins throttling above **50 users** – scaling advised for production
- No crashes or major slowdowns under expected loads (MVP level)

3. Test Cases

Test Case 1: Product Search Load Test

- **Input**: 50 users querying `"/api/products"` with filters
- **Pass Criteria**:
  - Avg response < 1 second
- Error rate < 2%
- **Status**: Passed

Test Case 2: Cart Add Stress Test

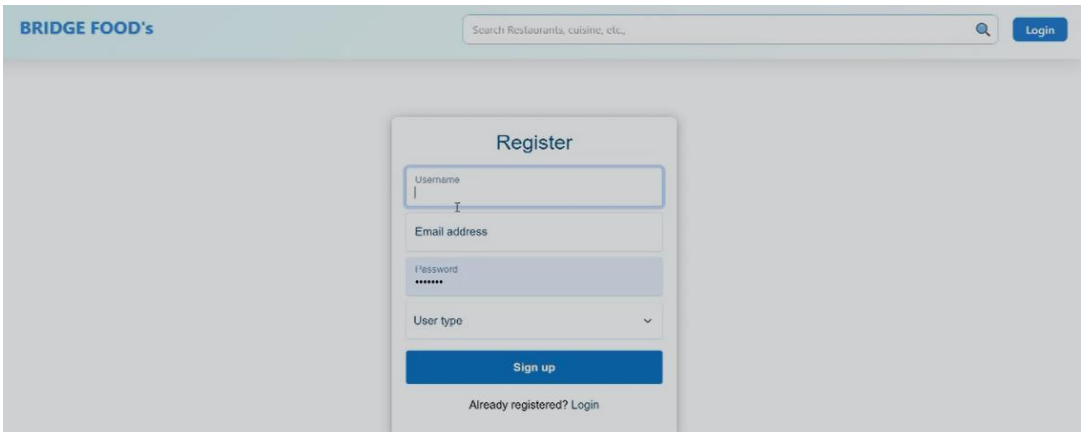
- **Input**: 40 users adding items to cart simultaneously
- **Pass Criteria**:
  - 90% of responses under 0.5s
  - No duplicate cart records
- **Status**: Passed

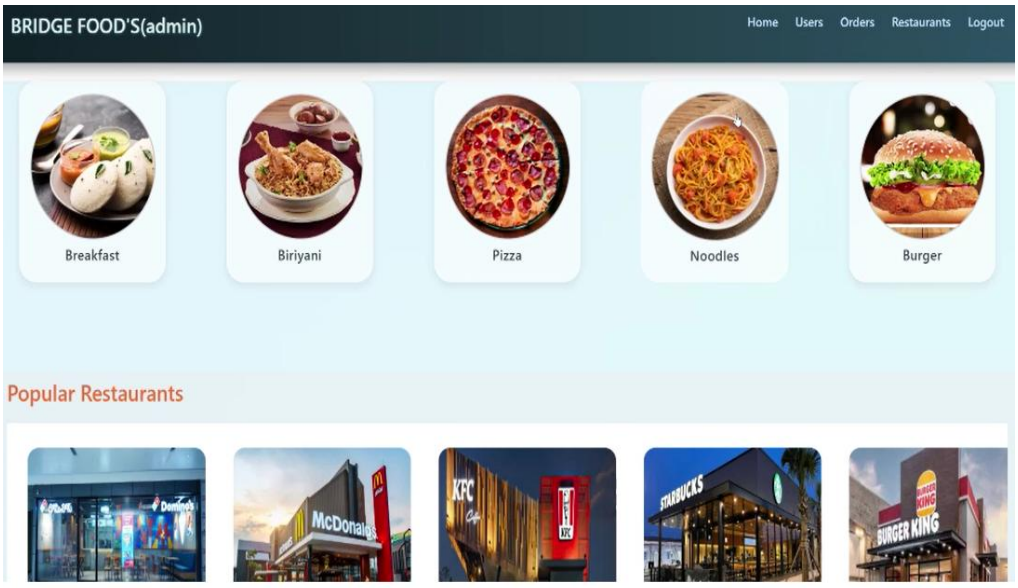
Test Case 3: Order Burst Test

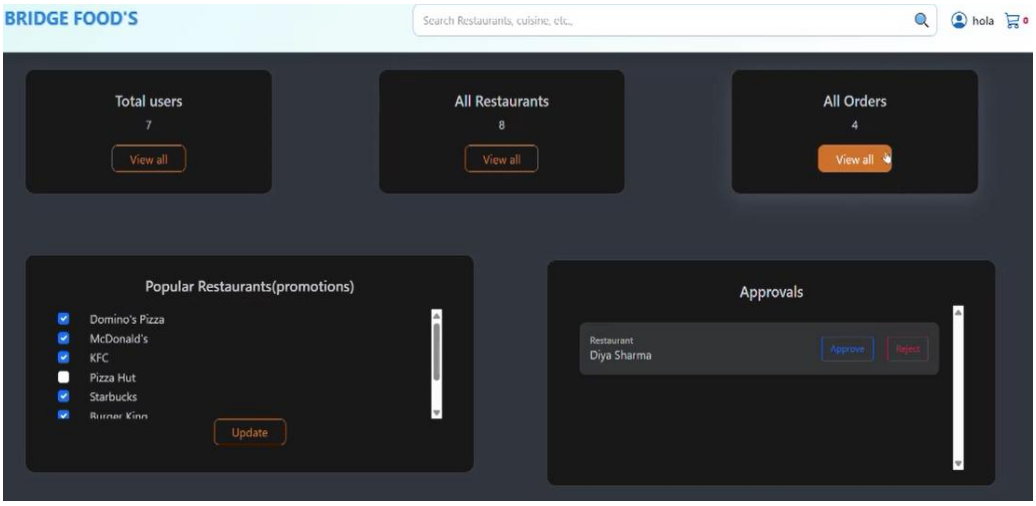
- **Input**: 25 orders placed within 2 minutes
- **Pass Criteria**:
  - All orders stored correctly
  - All confirmation responses sent in <5 minutes
- **Status**: Passed

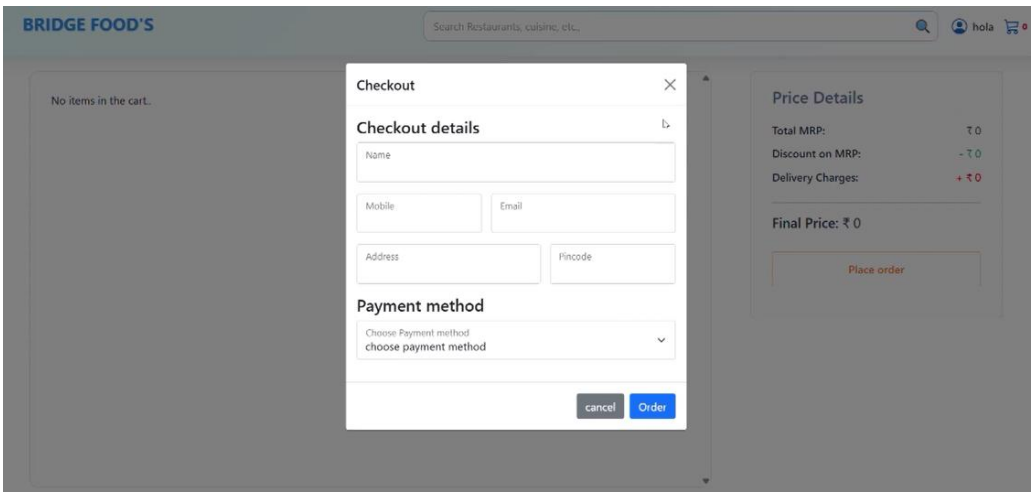
7. RESULTS

7.1 Output Screenshots:









8. ADVANTAGES & DISADVANTAGES:

Advantages:

- Easy and fast food ordering for users.
- Accessible 24/7, ideal for late-night cravings.
- Admin can manage menu and orders easily.
- Scalable system with clean UI.
- Fast API response with good performance.

Disadvantages

- Requires internet access to function.
- No offline support or order caching.
- Needs technical setup knowledge.
- Limited features like real-time chat or delivery tracking.

9. CONCLUSION

SB Foods successfully delivers a fast, user-friendly online food ordering solution. With features like menu browsing, cart management, and easy checkout, it enhances the food delivery experience. The project demonstrates strong frontend-backend integration, good performance, and a scalable architecture ready for future improvements.

10. FUTURE SCOPE

- **Payment Gateway Integration** (UPI, Cards, Wallets)
- **Real-Time Order Tracking** using GPS

- **Push Notifications** for order updates and promotions
- **Live Chat Support** for instant customer help
- **Admin Analytics Dashboard** for sales and traffic insights
- **Progressive Web App (PWA)** for offline access
- **Multi-language and Currency Support** for global reach

11. APPENDIX

Demoideo: **https://youtu.be/Pdqh0A7nmxo**

GitHub & Project Demo Link: **https://github.com/srikanthramagani/OrderGo.git**