SB Foods - Detailed Project Report

# Introduction

Project Title: SB Foods: Online Food Ordering and Management System

# Team Members:

* 1. **Jameeru J** - Team Lead, Backend Development.
  2. **Abdul Kalam K** - Frontend Development.
  3. **Madhavan M** - Database Design and Integration.
  4. **Sridhar R** - Testing and Quality Assurance.

# Project Overview Purpose:

SB Foods is an online food ordering platform designed to streamline interactions between customers, restaurants, and admins. The platform facilitates product listing, cart management, and secure ordering processes. Restaurants benefit from a dashboard to manage their products and orders efficiently.

# Features:

* Comprehensive Product
* Catalog: Browse food items across diverse restaurants with detailed descriptions, reviews, pricing, and discounts.
* Secure Checkout Process: Ensure safe transactions with a seamless user interface.
* Order Details and History: Track orders, including payment methods, shipping addresses, and order summaries.
* Admin Management: Control over users, products, and restaurant approvals.
* Restaurant Dashboard: Manage listings, monitor order activity, and view order details.

# Architecture Frontend:

The frontend is built with React.js, employing reusable components, state management using React Context API or Redux, and responsive design with Material UI and CSS. Pages include:

 Home

 Product Catalog

 Cart

 Checkout

 Order History

# Backend:

The backend uses Node.js with Express.js to handle REST API calls, middleware functions, and server-side operations. Key functionalities include:

 User authentication (JWT-based).

 Order and cart management APIs.

 Admin operations for user and product management.

# Database:

The database is designed with MongoDB, utilizing the Mongoose ODM. Key collections include:

* **Users:** Stores user details and credentials.
* **Restaurants:** Holds restaurant information and product listings.
* **Products:** Details of food items, including pricing and categories.
* **Carts:** Tracks items added by users.
* **Orders:** Records completed orders and associated details.

1. Setup Instructions Prerequisites:

* Install Node.js and npm .

**(Download Link) :** <https://nodejs.org/en/download/package-manager>

* Install MongoDB

**(Download Link) :** <https://www.mongodb.com/try/download/compass>

* Install Git

**(Download Link) :** <https://git-scm.com/downloads>

* Install an IDE like Visual Studio Code

**(Download Link) :** <https://code.visualstudio.com/download>

* Installation:

Clone the Repository:

git clone <https://github.com/Jameeru/Sbfood_Website>.git

bash

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

# Install Dependencies:

bash

npm install

Set Up Environment Variables:

Create a .env file in the root directory with the following:

* 1. MONGO\_URI=your\_mongodb\_connection\_string
  2. PORT=5000

# Folder Structure Client (React):

* 1. src/components: Contains reusable components like ProductCard, CartItem, etc.
  2. src/pages: Includes primary pages like Home.js, Cart.js, and Checkout.js.
  3. src/context: Manages application state using Context API.

# Server (Node.js):

* 1. routes: Contains API route definitions (userRoutes.js, productRoutes.js).
  2. controllers: Business logic for each route (userController.js, orderController.js).
  3. models: MongoDB schemas (User.js, Order.js, Product.js).

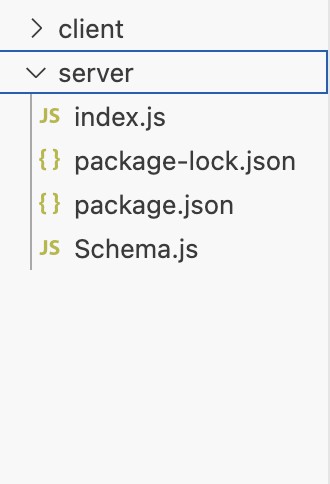
# Running the Application Frontend:

**bash**

**<<**cd client

**<<**npm start

# Backend:



**bash**

**>>**cd server

**>>**npm start

Access the application at **http://localhost:3000.**

# API Documentation

Endpoints:

# Users:

 POST /api/users/register – Registers a new user.

 POST /api/users/login – Logs in a user and returns a JWT.

# Products:

 GET /api/products – Fetch all products.

# Cart:

 POST /api/cart – Add item to cart.

 GET /api/cart/:userId – Retrieve user's cart.

# Orders:

 POST /api/orders – Place an order.

 GET /api/orders/:userId – Fetch user's order history.

# Authentication

JWT Authentication: Tokens are generated upon login and stored in localStorage. Role-based Access Control:

Users: Restricted to product browsing and ordering.

Admins: Full access to user, product, and order management.

# User Interface

Key pages include:

 Home Page: Displays available restaurants and products.

 Cart Page: Showcases items added by the user with options to edit/remove.

 Checkout Page: Captures address and payment details.

 Order History: Displays previous orders.

# Testing

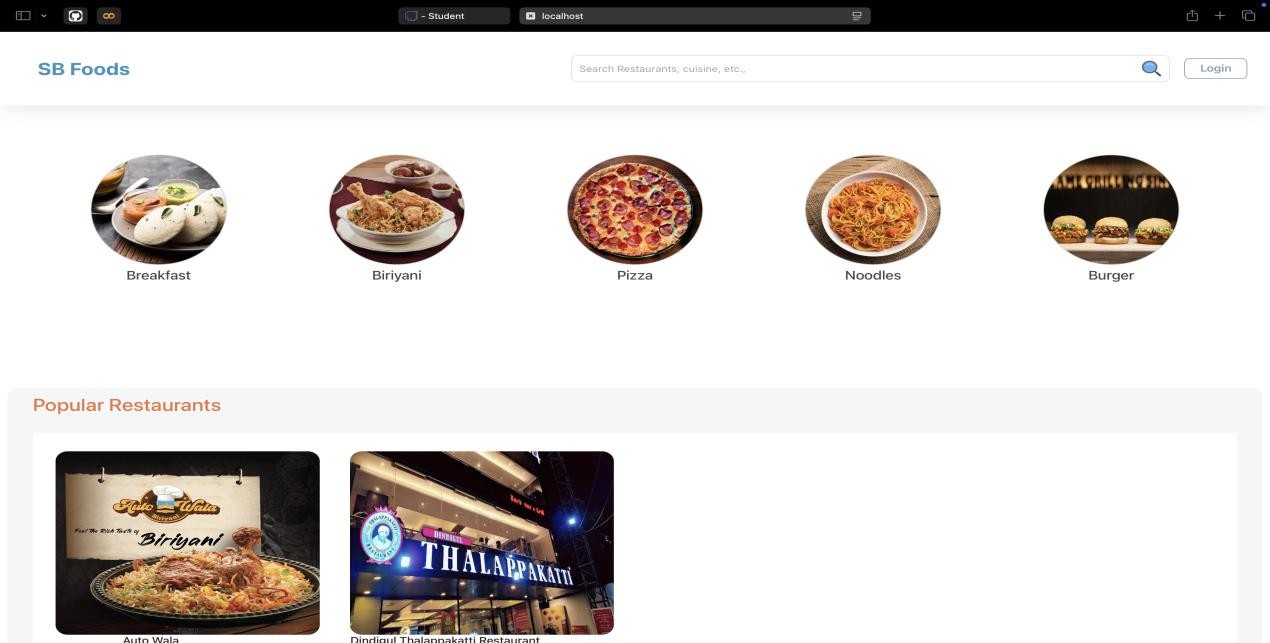
**Frontend:** Jest and React Testing Library for component testing.

**Backend:** Mocha and Chai for API endpoint testing.

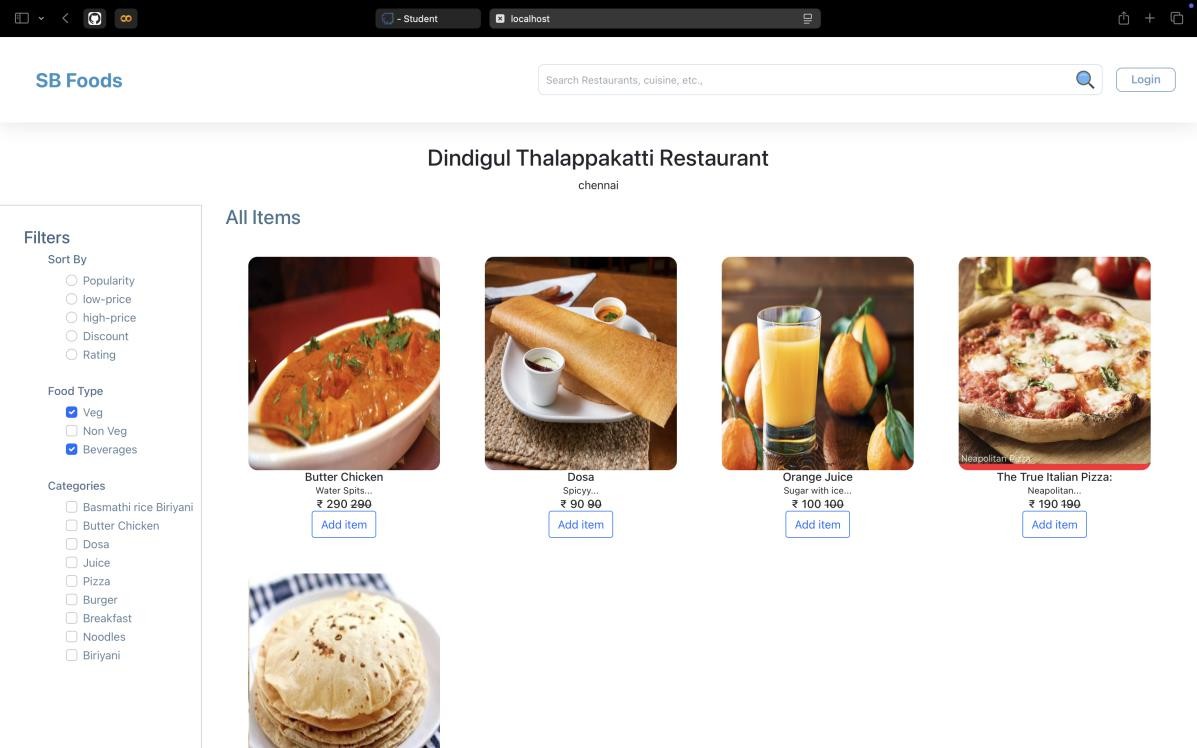
# Screenshots or Demo

Screenshots:

**Home Page**



**Cart Page**



**Checkout Flow**

Demo:

[ <https://drive.google.com/file/d/1g6XH1CLsRswCbywnfFp5ENNUPCF2Ks9l/view?usp=sharing> ]

# Known Issues

Occasionally slow response times when querying large datasets. Lack of email notifications for order confirmation.

# Future Enhancements

**Mobile Application:** Develop a companion app for iOS and Android. **Recommendation System:** Suggest popular products based on user history. **Payment Integration:** Add PayPal and Apple Pay.

**Real-Time Order Tracking:** Enable GPS-based delivery tracking.