SB FOODS

Food Ordering Application

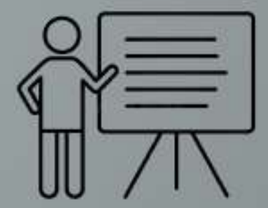
MONDODB, EXPRESS, REACT AND NODE.JS

PRESENTATION OVERVIEW

TEAM MEMBERS

- Alice A(310821104009)
- Juliya A(310821104043)
- Harini M(310821104034)
- Rajesh Kanna A(310821104306)

1. Abstract
2. Introduction
3. System Requirements
4. System Design
5. Implementation
6. Key Functionalities
7. Testing
8. Results
9. Challenges and solutions
10. Future Enhancements
11. Conclusion
12. Project Setup for Github



ABSTRACT

This Project is a Web-based food delivery application designed to streamline online food ordering. The app enables users to browse items, add them to a cart, and place orders, while an admin panel manages food items and orders. Built with React for the Frontend, Express for the Backend and MongoDB for the Database, this application offers an integrated approach to online food ordering with user authentication, cart management and order processing

INTRODUCTION

The Objective of this project is to create an efficient, user-friendly food delivery application. It focuses on intuitive navigation, responsive design and easy order processing to enhance user experience and simplify food delivery management for admins.

SYSTEM REQUIREMENTS

SOFTWARE:

- ***Frontend***: *React*
- ***Backend***: *Node.js, Express*
- ***Database***: *MongoDB*
- ***Additional Tools***: *NPM, Postman (for API testing), GitHub (for version control*

HARDWARE:

- ***Processor***: *Intel i3 or above*
- ***RAM***: *4GB minimum*
- ***Storage***: *500 MB for app data*

MODULES:

1. ***User Module***: *Handles user registration, login, and authentication.*
2. ***Admin Module***: *Manages food items and orders.*
3. ***Cart Management***: *Allows users to add, update, or remove items.*
4. ***Order Processing***: *Processes orders placed by users.*

SYSTEM DESIGN

ARCHITECTURE:

The application uses a client-server architecture, with the frontend handling the user interface and the backend managing API endpoints, user authentication, and database interactions. MongoDB is used as the primary data source.

DATABASE SCHEMA:

*- **Users**: _id, username, email, password*
*- **Food Items**: _id, name, description, price, category, image*
*- **Orders**: _id, userId, items, totalPrice, status*

TOOLS AND TECHNOLOGIES USED:

1. ***User Module***: *Handles user registration, login, and authentication.*
2. ***Admin Module***: *Manages food items and orders.*
3. ***Cart Management***: *Allows users to add, update, or remove items.*
4. ***Order Processing***: *Processes orders placed by users.*

IMPLEMENTATION

FRONTEND:

The React frontend includes components for navigation, displaying food items, cart management, and order submission. React Router manages routes, while Axios is used for API calls to the backend.

BACKEND:

Node.js and Express are used to create REST APIs for user authentication, food item retrieval, and order management. Mongoose connects to MongoDB and performs data validation.

ADMIN PANEL:

Admin users can manage menu items and orders using a separate panel, providing CRUD operations on the food items and viewing orders.

KEY FUNCTIONALITIES

USER FEATURES:

- ***Registration & Login***: *Users can register or log in with their credentials.*
- ***Menu Browsing***: *Users view available food items and search by category.*
- ***Cart Management***: *Add, remove, or update items in the cart.*
- ***Order Placement***: *Place orders and view order status.*

ADMIN FEATURES:

- ***Food Item Management***: *Add, update, or delete menu items.*
- ***Order Management***: *View and update the status of orders.*

TESTING

TESTING TYPES:

- ***Unit Testing***: *Individual components and API endpoints.*
- ***Integration Testing***: *Testing interactions between frontend, backend, and database.*
- ***User Acceptance Testing (UAT)***: *Ensures the final product meets user expectations.*

TESTING TOOLS:

Jest for frontend tests, Postman for API testing, and manual testing for UI validation.

RESULTS

PERFORMANCE METRICS:

The application performs efficiently under moderate load, with an average response time of less than 200ms for most API requests.

USER FEEDBACK:

Initial test users found the application easy to navigate, with a smooth and intuitive order process.

CHALLENGES AND SOLUTIONS

CHALLENGES:

1. *Implementing secure payment processing.*
2. *Ensuring database reliability and scalability.*

SOLUTIONS:

1. *Used Stripe for secure, test-mode payment integration.*
2. *Used MongoDB with data validation through Mongoose.*

FUTURE ENHANCEMENT

- Introduces AI chatbots to assist with FAQs and real-time support.
- Add a recommendation engine based on user preferences.
- Introduce delivery partner functionality.
-Add live order tracking with delivery time estimate.
-.Add an Eatlist feature to save favorites and provide personalized dish recommendations.

CONCLUSION

This project successfully demonstrates a streamlined approach to online food ordering with user and admin functionalities. It is built using modern web technologies, including React, Node.js, and MongoDB, ensuring scalability and a good user experience.

PROJECT SETUP FOR GITHUB

FRONTEND:

1. *Navigate to the frontend directory:*
   cd frontend

2. *Install dependencies:*
   npm install

3. *Start the frontend server:*
   npm run server

Front-end

BACKEND:

1. Navigate to the backend directory:
cd backend

2. Install dependencies:
npm install

3. Start the backend server:
npm run server
npm install express mongoose cors dotenv body-parser bcryptjs jsonwebtoken stripe morgan helmet
npm install react react-dom react-router-dom axios
npm install --save-dev nodemon
npm install axios

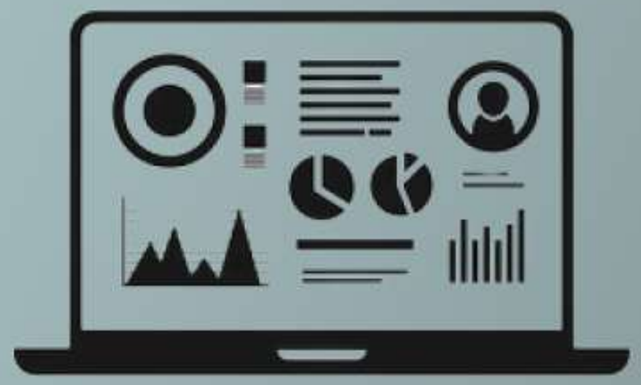
BACKEND

ADMIN PANEL:

1. Navigate to the admin directory:
cd admin

2. Install dependencies:
npm install

3. Start the admin panel:
npm run dev



THANK YOU