# Full Stack Development with MERN

## Project Documentation

## 1. Introduction

Project Title: Food Ordering App Using MERN  
Team Members:  
- Sriraghavan K (Team Leader): Backend  
- Srimanjunath R: Frontend  
- Vignesh M: Database  
- Raja: Testing

## 2. Project Overview

Purpose:  
The primary goal of this project is to develop a seamless food delivery platform connecting customers and restaurants. The platform aims to provide a simple and intuitive way for users to order food, ensuring efficient delivery management. Advanced features enhance the user experience for customers, restaurant owners, and administrators.

Features:  
1. User Authentication and Authorization: Secure login, registration, and role-based access.  
2. Restaurant Listing and Menu Management: Dynamic restaurant profiles and easy-to-use menu tools.  
3. Order Placement and Real-Time Tracking: Track order preparation and delivery status.  
4. Separate Restaurant Login: Independent dashboards for restaurant owners.  
5. Admin Panel for Management: Tools for managing users, restaurants, and orders.  
6. Late-Night Ordering Feature: Night mode and options for late operations.  
7. Analytics Dashboard: Insights into trends, sales, and customer preferences.

## 3. Architecture

Frontend:  
- Built with React.js for reusable, modular components.  
- Styled using Tailwind CSS and Bootstrap for responsive and consistent design.  
- Animations incorporated via Framer Motion for interactive user experiences.

Backend:  
- Node.js with Express.js for RESTful APIs and efficient request handling.  
- Middleware for authentication, error handling, and route protection.

Database:  
- MongoDB as the primary database, offering flexibility with NoSQL.  
- Mongoose used for schema modeling, ensuring data consistency and validation.

## 4. Setup Instructions

Prerequisites:  
- Node.js  
- MongoDB

Installation:  
1. Clone the repository.  
2. Navigate to the frontend (`/src`) and backend (`/server`) directories.  
3. Install dependencies using `npm install`.  
4. Set up a `.env` file for environment variables, including the MongoDB URI and any API keys.

## 5. Folder Structure

Frontend (Client):  
  
client/  
├── src/  
│ ├── components/  
│ ├── context/  
│ ├── pages/  
│ ├── styles/  
│ ├── App.js  
│ └── index.js  
└── package.json

Backend (Server):  
  
server/  
├── api/  
├── models/  
├── routes/  
├── middleware/  
├── config/  
└── package.json

## 6. Running the Application

Frontend:  
1. Navigate to the `src` directory.  
2. Install dependencies with `npm install`.  
3. Start the server using `npm start`.  
4. Default URL: [http://localhost:3000].

Backend:  
1. Navigate to the `server` directory.  
2. Install dependencies with `npm install`.  
3. Start the server using `npm start`.  
4. Default URL: [http://localhost:5000].

## 7. API Documentation

Authentication:  
- Register User: `POST /api/auth/register`  
 - Request: `{ "name": "John", "email": "john@example.com", "password": "1234" }`  
 - Response: `{ "message": "User registered successfully", "userId": "1" }`  
  
- Login User: `POST /api/auth/login`  
 - Request: `{ "email": "john@example.com", "password": "1234" }`  
 - Response: `{ "token": "xyz", "userId": "1" }`  
  
Restaurant Management:  
- Get All Restaurants: `GET /api/restaurants`  
 - Response: `[ { "id": "1", "name": "Pizza Place", "cuisine": "Italian" } ]`

## 8. Authentication

- JWT used for secure sessions.  
- Tokens stored in client-side local storage.  
- Role-based access control ensures secure user permissions.

## 9. User Interface

Features:  
- Intuitive and responsive designs.  
- Screenshots include the home page, restaurant dashboard, and admin panel.  
- Scan the provided QR code for a video demo showcasing functionality.

## 10. Testing

1. Unit Testing: Individual React components tested with Jest.  
2. Integration Testing: Verified seamless frontend-backend interactions.  
3. API Testing: RESTful endpoints tested with Postman.  
4. End-to-End Testing: Real-world scenarios simulated using Cypress.

## 11. Future Enhancements

1. Add a loyalty program for frequent customers.  
2. Integrate with multiple payment gateways like Stripe and Razorpay.  
3. Develop a mobile app version using React Native for enhanced accessibility.