**Food Ordering Application Project**

1. **Project Overview**

The project aims to build a comprehensive food ordering application, enabling users to browse food items, add them to a cart, and place orders seamlessly. It combines a user-facing website, an admin panel for order management, and a backend server for data processing and API management.

1. **Purpose of the Project**

- Enhancing User Experience: Provide a smooth and user-friendly experience for customers who want to order food online. Users can create accounts, save their favorite items, and track their orders.

- Streamlining Order Management: Facilitate restaurants and food businesses to manage orders in real-time through a dedicated admin panel, allowing quick updates on order status and inventory management.

- Secure and Efficient Payment Processing: Integrate a robust payment gateway (Stripe) to ensure secure transactions, offering a reliable way for users to make payments and for restaurants to receive funds quickly.

- Scalability and Flexibility: Create a scalable solution using modern web technologies that can easily accommodate an increasing number of users and orders without compromising performance.

- Seamless Integration Between Frontend and Backend: Enable effective communication between frontend and backend services to ensure a cohesive flow of data, minimizing downtime and improving reliability.

1. **Technologies Used**

- ***React JS*** : For building a dynamic, responsive user interface that allows for seamless navigation between pages and components like product lists, shopping carts, and user profiles. React’s component-based architecture ensures code reusability and a structured development process.

- ***Redux*** : Employed for state management to handle user authentication, cart items, and order status across the application, providing a predictable state container that simplifies data flow.

- ***Node JS & Express*** : Used to build a robust backend server capable of handling API requests, user authentication (via JWT), and communication with the database. Express simplifies routing and middleware integration.

- ***MongoDB*** : Chosen as the database for its flexibility in handling a variety of data types and scalability. Mongoose is used to define schemas and models, enabling efficient queries and data validation.

- ***Stripe*** : Integrated for processing payments, managing customer billing, and ensuring secure transactions. Stripe's API simplifies the payment process, making it easy to handle both one-time payments and subscription-based services.

- ***TailwindCSS*** : For designing a modern, responsive user interface that works seamlessly across devices, ensuring users have a consistent experience regardless of their screen size.

1. **Detailed Project Creation Process**

- ***Frontend Development*** :  
 - Create reusable components for different sections like product listings, user profile, and checkout.  
 - Implement client-side routing for smooth navigation using `react-router-dom`.  
 - Use Axios for making API calls to the backend, handling data fetching, user login/logout, and managing cart items.

- ***Admin Panel Development*** :  
 - Develop a dashboard using React that allows admins to monitor and update order statuses, manage food items, and view sales reports.  
 - Implement secure login for admins with role-based access control to restrict certain functionalities.

- ***Backend Development*** :  
 - Set up Express routes for user registration, login, and order management.  
 - Use JWT for secure user authentication and sessions.  
 - Implement endpoints for CRUD operations on food items and managing user orders.

- ***Payment Integration*** :  
 - Configure Stripe’s API keys in a secure environment.  
 - Create payment flows using Stripe’s `Checkout` and `Payment Intents` to handle various payment scenarios.  
 - Listen for Stripe webhooks to update order statuses upon successful payment completion.

1. **Team Members and Their Roles**

- Frontend Developer ( Mahmoud Fathi ) :

Responsible for building all React components, integrating API calls, and designing the user interface.

- Backend Developer ( Amir El Sayed ):

Focuses on creating RESTful APIs with Node.js and Express, handling database queries, and integrating payment processing.

- Database Engineer ( Ahmed Yousry ):

Designs MongoDB schemas for user data, orders, and product listings, optimizing queries for faster performance.

- Project Manager ( Mahmoud Fathi ):

Oversees project milestones, ensures smooth coordination between team members, and reviews code for quality and consistency.

- QA Engineer ( Amir El Sayed ):

Tests all functionalities, including payment flows, user authentication, and order management, ensuring that the application is free of bugs before deployment.

**6. How to Run the Project**

- Prerequisites:  
 - Ensure `Node.js` and `npm` are installed.  
 - Set up a MongoDB database and configure the connection URI.  
 - Create a Stripe account and obtain the API keys.

- Setup Instructions:  
 - Clone the repository and navigate into the project directory.  
 - Run `npm install` in both the frontend and backend directories to install all dependencies.

- Create a `.env` file in the backend with the required environment variables such as `MONGODB\_URI`, `STRIPE\_SECRET\_KEY`, and `JWT\_SECRET`.

- Running the Application:  
 - Start the backend server with `npm run server` to enable API services.  
 - Launch the frontend using `npm start` to serve the React application.

URL:https://drive.google.com/file/d/1Xm6Sw48tvhEpviXuSKFqAU4U9\_FHYce1/view?usp=drive\_link