**Ice Cream Parlor Delivery Website Project Report**

**1. Introduction**

**1.1 Project Overview**

The Ice Cream Parlor Delivery Website is a full-stack web application designed to facilitate the ordering and delivery of ice cream directly to customers' homes. This project leverages the MERN (MongoDB, Express.js, React.js, Node.js) stack for development and includes a simple payment gateway implementation.

**1.2 Objectives**

* To create a user-friendly interface for ordering ice cream.
* To provide a secure and efficient checkout process with both Stripe and Cash on Delivery (COD) payment options.
* To manage orders effectively through an admin panel.
* To ensure scalability and maintainability using modern web development practices.

**2. System Architecture**

**2.1 Frontend**

The frontend of the application is built using React.js. The core features include:

* **Product Listing**: Displays all available ice cream products.
* **Cart Management**: Allows users to add or remove items from the cart.
* **Order Placement**: Provides a checkout process with multiple payment options (Stripe and COD).
* **Order Tracking**: Enables users to track their orders.

**2.2 Backend**

The backend is developed using Node.js and Express.js, with MongoDB as the database. Key components include:

* **User Authentication**: Secure login and registration using JWT.
* **Order Management**: Handles order creation, payment processing, and status updates.
* **Admin Panel**: Allows administrators to view, update, and manage orders.
* **API Endpoints**: Provides RESTful API services for frontend interaction.

**3. Database Design**

**3.1 MongoDB Schema**

The database consists of multiple collections, including:

* **Users**: Stores user credentials, personal information, and cart data.
* **Products**: Contains details of each ice cream product, including name, price, and description.
* **Orders**: Records all orders placed by users, including payment details, order status, and delivery address.

**4. Implementation**

**4.1 Frontend**

* **React Components**:
  + **HomePage**: Displays the list of available ice creams.
  + **Cart**: Manages items added by the user and calculates the total amount.
  + **Checkout**: Allows users to choose their payment method and place an order.
  + **MyOrders**: Allows users to see their current and previous orders.
* **State Management**:
  + **Context API** is used to manage global state, including the cart, user authentication, and product list.

**4.2 Backend**

* **API Routes**:
  + **User Routes**: Handles user registration, login, and profile management.
  + **Product Routes**: Manages CRUD operations for ice cream products.
  + **Order Routes**: Handles order placement, payment processing, and order status updates.
  + **Cart Routes**: Handles adding and removing item from cart with getting the whole cart as well.
* **Payment Integration**:
  + **Stripe**: Implemented for secure credit/debit card transactions.
  + **Cash on Delivery (COD)**: Added as an alternative payment method for customers who prefer offline payment.

**4.3 Key Features**

* **Secure User Authentication**: JWT-based authentication ensures user data is secure.
* **Multiple Payment Options**: Users can choose between Stripe and COD during checkout.
* **Admin Panel**: Provides administrators with the ability to manage orders and view user details.

**5. Testing**

**5.1 Unit Testing**

* **Jest** was used to write unit tests for React components and backend API endpoints.
* **Testing Focus**:
  + User authentication flow.
  + Order placement and cart management.
  + Payment processing with Stripe and COD.

**5.2 Integration Testing**

* **Tested** the complete flow from product selection to order placement and payment.
* **Tools**: Postman was used to test API endpoints, ensuring data consistency and error handling.

**5.3 User Acceptance Testing (UAT)**

* Conducted by simulating user interactions with the web app to ensure it meets the project’s objectives.

**6. Challenges and Solutions**

**6.1 Stripe Integration**

* **Challenge**: Ensuring seamless payment processing and secure transactions.
* **Solution**: Leveraged Stripe’s documentation and SDKs to integrate payment seamlessly into the app.

**6.2 State Management**

* **Challenge**: Managing state across multiple components, especially in handling the cart and user sessions.
* **Solution**: Used React Context API to manage global state efficiently.

**6.3 Admin Panel**

* **Challenge**: Providing a comprehensive overview of orders for administrators.
* **Solution**: Implemented a simple yet effective admin panel with filters for order management.

**7. Conclusion**

The Ice Cream Parlor Delivery Website successfully meets the project’s objectives by providing a smooth, user-friendly interface for ordering ice cream, secure payment options, and efficient order management. The application is scalable and can easily be expanded with new features in the future.

**8. Future Enhancements**

* **Adding Order Tracking**: Implement real-time order tracking for customers.
* **Mobile App**: Develop a mobile application for iOS and Android to enhance accessibility.
* **Promotional Offers**: Integrate discounts and promotional offers to attract more customers.
* **User Reviews**: Allow users to leave reviews and ratings for products.

**9. Appendix**

**9.1 Code Repositories**

* **Frontend**: [https://github.com/maryaminam/IceCreamParlor/tree/main/frontend]
* **Backend**: [https://github.com/maryaminam/IceCreamParlor/tree/main/backend]
* **Admin**: [https://github.com/maryaminam/IceCreamParlor/tree/main/admin]

**9.2 Tools and Technologies**

* **Frontend**: React.js, Context API, Axios
* **Backend**: Node.js, Express.js, MongoDB
* **Database**: MongoDB
* **Payment Gateway**: Stripe
* **Version Control**: Git