Project Documentation

1. Introduction

- **Project Title:** Food Delivery App
- **Team Members:**
    - **Thomas Tharun W**
    - **Syed Uzayr Ahmed**
    - **Perarasan S**
    - **Andrew Nathan A**

2. Project Overview

- **Purpose:**
  The Food Delivery App allows users to browse a variety of food options, place orders, and track their deliveries conveniently. It aims to streamline the food ordering process for both customers and restaurants.
- **Features:**
  Key features of the application include:
    - User authentication and authorization.
    - A searchable menu with categories and filters.
    - Real-time order tracking.
    - Admin dashboard for restaurant management.

3. Architecture

Frontend

The frontend is developed using **React**, offering:

- A dynamic user interface with reusable components.
- Responsive design for desktop and mobile devices.

Backend

The backend is powered by **Node.js** and **Express.js**, featuring:

- RESTful APIs to handle user and order data.
- Middleware for validation and authentication.

Database

MongoDB is used for data storage, with:

- Schemas for users, orders, and menu items.
- Efficient querying and indexing for performance.

4. Setup Instructions

Prerequisites

Ensure you have the following installed:

- Node.js
- MongoDB
- Git

Installation

Clone the repository:
bash
Copy code
```
git clone https://github.com/tommyboiii004/Food-Delivery-App.git
cd Food-Delivery-App
```

Install dependencies:
bash
Copy code
```
cd client
npm install
cd ../server
npm install
```

1.
2. Set up environment variables:

Create a `.env` file in the `server` directory with:
plaintext
Copy code
```
PORT=XXXX
MONGO_URI=your_mongodb_connection_string
JWT_SECRET=your_secret_key
```

◦

---

5. Folder Structure

Client

- **`src/components`**: UI components like navbar, cards, and modals.
- **`src/pages`**: Views for each route (e.g., Home, Cart, Orders).
- **`src/services`**: API interaction logic.

Server

- **`routes/`**: Defines API endpoints for users, orders, and admin features.
- **`models/`**: MongoDB schemas for storing data.
- **`middleware/`**: Custom middleware for handling authentication and error logging.

6. Running the Application

Frontend

To start the React frontend:

bash
Copy code
```
cd client
npm start
```

Backend

To start the Node.js backend:

bash
Copy code
```
cd server
npm start
```

7. API Documentation

Example API Endpoint:

- **Endpoint:** `/api/orders`

- **Method:** POST
- **Parameters:**
    - `userId` (string)
    - `items` (array)

Example Response:
json
Copy code
```
{
  "message": "Order placed successfully",
  "orderId": "12345"
}
```

8. Authentication

- **JWT-based authentication:** Secure tokens are used for login and authorization.
- Protected routes ensure only authenticated users can access specific features.

9. User Interface

[Include descriptions and screenshots of: login page, menu page, order tracking, and admin dashboard.]

10. Testing

- **Testing Tools:** Jest for unit tests and Postman for API tests.
- **Approach:**
    - Unit tests for critical functions.
    - Integration tests for API endpoints.

11. Screenshots or Demo

[Add relevant screenshots or provide a link to a hosted demo, if available.]

12. Known Issues

- [Issue 1: Description and workaround]

- [Issue 2: Description and workaround]

13. Future Enhancements

- Integration with third-party delivery services.
- Enhanced order tracking with live location updates.
- Additional payment gateway support.