DV200

Progress Milestone Check

221076 Iné Smith

FreigthGo

Checklist

Feature	Status	Description
User Authentication	Busy Finalising	Users can register and log in as Standard User or a Transporter.
CRUD Operations	Working on it	Users can create, read, update, and delete freight bookings and user profiles.
API Integration	Almost Done	Integrated with external API for truck availability and route optimization.
Frontend Functionality	Busy Finalising	The React frontend communicates with the backend for seamless booking management and login sessions.
Backend Functionality		Node.js and Express handle all backend logic, including data processing and routing.
Responsive Design	Done	Frontend is responsive and adjusts to desktop, tablet, and mobile devices. Uses CSS frameworks like Bootstrap.
Booking Management System	Working on it, Frontend is done	Customers can book trucks based on load type, weight, and location. Bookings can be tracked by transporters and managed by admins.
Standard User Dashboard	Almost Done	Standard User can book trucks.

Transporter Dashboard	Working on it	Transporters can update their profile, truck availability, and manage assigned bookings.
Customer Booking Tracking	Under Construction	Customers can view their current bookings, status updates, and ETA of deliveries in real-time.
Profile Page	Under Construction	Users can view their details, update details or see past bookings.

System Documentation

Technical Architecture

- **Frontend:** React.js for user interfaces and state management.
- **Backend:** Node.js with Express for API routes and business logic.
- **Database:** MongoDB to store user data, bookings, and truck profiles.
- **Authentication:** .
- **Interactions:** Frontend communicates with backend through API endpoints, and backend communicates with MongoDB to store and retrieve data.

1.

Database Schema

Key Collections:

1. **Users** (Stores customer, admin, and transporter data)
    - Fields: `userID`, `role`, `username`, `email`, `password`, `createdAt`

2. **Bookings** (Manages bookings for freight services)
    - Fields: `bookingID`, `customerID`, `transporterID`, `truckType`, `weight`, `status`, `pickupLocation`, `dropOffLocation`, `createdAt`
3. **Trucks** (Stores truck information)
    - Fields: `truckID`, `transporterID`, `truckType`, `availabilityStatus`

API Endpoints

- **Authentication:**
    - `POST /api/auth/register` – Register a new user.
    - `POST /api/auth/login` – User login and receive JWT.
- **User Management:**
    - `GET /api/users` – Retrieve all users (admin access only).
    - `PUT /api/users/:id` – Update user information.
- **Booking Management:**
    - `POST /api/bookings` – Create a new booking.
    - `GET /api/bookings/:id` – Retrieve booking details.
    - `PUT /api/bookings/:id` – Update booking status.
    - `DELETE /api/bookings/:id` – Cancel a booking.

UI Design

The application consists of the following key pages:

1. **Home Page:** Displays a summary of services and links to login/register.
2. **Dashboard:** Separate dashboards for standard users and transporters, tailored to their roles.

3. **Log In and Sign Up Pages:** Users can decide how to sign up (standard user / transporter) and then log into their account.
4. **Booking Form:** Allows customers to submit freight booking requests.
5. **Responsive Design:** Tested on mobile, tablet, and desktop using Chrome DevTools to ensure usability.

Instructions for Running the Application

1. **Clone the Repository:**
   ```
   git clone <repository-url>
   cd FreightGo
   ```

2. **Install Dependencies:**
   ```
   npm install
   cd client && npm install
   ```

3. **Run Backend Server:**
   ```
   npm start
   ```

4. **Run Frontend Client:**
   ```
   cd client
   npm start
   ```

5. **Environment Setup:**
   Create a `.env` file for backend configuration with:
   ```
   MONGO_URI=<your-mongo-db-uri>
   JWT_SECRET=<your-secret-key>
   ```

Problem Statement

FreightGo: Addressing Logistics Challenges in Agriculture and Small Businesses

Efficient freight management is crucial for farmers and small businesses to transport goods reliably. However, many face difficulties in finding affordable and available trucks for deliveries, especially in rural areas. This challenge can lead to delays, increased costs, and logistical disruptions.

FreightGo offers a digital solution to bridge this gap by providing a platform where users can easily book trucks based on their needs. Customers can specify the type of truck, load weight, and delivery routes, ensuring streamlined logistics. With real-time booking management and transparent tracking, FreightGo reduces the complexity of securing transportation.

This platform empowers small businesses, farmers, and transporters by improving access to freight services. It addresses the economic impact of delayed deliveries, enhancing productivity and supporting growth in rural and urban areas alike. FreightGo thus plays a vital role in making logistics more accessible, efficient, and equitable for all users.