AI-Driven Parcel Management System for India Post

Project Description

This project is an AI-driven parcel management system for India Post, designed to enhance parcel delivery efficiency while providing convenience for senders, receivers, postmen, and administrators. It features three interfaces: 1. A mobile app for users (senders and receivers) and postmen. 2. A web-based admin dashboard for delivery operations. 3. Backend APIs for seamless data management.

Key Features:

- A shared user interface (mobile app) for senders and receivers.
- A dedicated postman interface for delivery management.
- A web dashboard for administrators to monitor and optimize delivery operations.
- Real-time tracking powered by Google Maps API.
- Secure OTP-based verification and SMS alerts using Twilio.

Tech Stack Summary:

- **Frontend:** React Native (Expo) for mobile apps, React.js for the web dashboard.
- **Backend:** Node.js for APIs.
- **Database:** MongoDB.
- **APIs:** Google Maps API, Twilio for SMS integration.

Project Architecture

Frontend Technologies

- **Mobile App:**
  - Built using React Native (Expo) for cross-platform compatibility.
- **Admin Dashboard:**
  - Built using React.js for responsive and dynamic web interfaces.
  - Primarily used by administrators for monitoring and managing operations.

Backend Technologies

- **Server-Side Framework:**
  - Developed using Node.js with Express for API creation and request handling.

Database

- MongoDB

APIs and Integrations

- **Google Maps API:** For route mapping, distance calculation, and traffic insights.
- **Twilio API:** For SMS notifications (parcel updates, OTPs).

Setup Instructions

Prerequisites

1. Node.js (v16.x or higher) and npm (v8.x or higher).
2. MongoDB (Installed locally or a connection to a remote MongoDB instance).
3. Expo CLI for running the React Native app (SDK 51).
4. React Developer Tools (optional, for debugging).
5. Python

File Structure

```
parcel-management-system/
  backend/               # Node.js backend APIs
  mobile-app/            # React Native app
  admin-dashboard/       # React.js admin dashboard
```

Setup Steps

1. Setup Backend

```
cd backend
npm install
```

- Create a `.env` file in the backend directory with the following variables:

```
# Server Configuration
PORT=5000
NODE_ENV=development

# MongoDB Connection
MONGO_URI=mongodb://localhost:27017/your_database_name

# JWT Authentication
JWT_SECRET=your_jwt_secret

# Logging Level
```

```
LOG_LEVEL=info

# API Keys and Secrets
SECRET_KEY=your_secret_key
API_SECRET=your_api_secret

# Email Configuration
HOST=smtp.gmail.com
SERVICE=gmail
EMAIL_PORT=587
SECURE=false
USER=your_email@gmail.com
PASS=your_email_password

#Twilio keys and tokens
TWILIO_AUTH_TOKEN=your_auth_token
TWILIO_ACCOUNT_SID=your_account_sid
TWILIO_SERVICE_SID=your_service_sid
MOBILE_NUMBER_VERIFIED=your_mobile_number_verified_by_twilio_for_getting_otp

# Application URL
BASE_URL=http://localhost:3000/
```

- Run the backend server:

```
npm start
```

2. Setup Python

```
cd backend/python
```

- Install the requirements:

```
pip install -r requirements.txt
```

- Run the python server:

```
python app.py
```

3. Setup Mobile App

```
cd ../mobile-app
npm install --legacy-peer-deps
```

- Create a `.env` file in the `mobile-app` directory with the following variables:

  ```
  EXPO_PUBLIC_API_URL=http://<your-backend-ip-address>:5000
  ```

- Run the app in development mode:

```
npm start
```

- Install **Expo Go (SDK 51)** on your mobile device to scan the QR code for testing.

4. Setup Admin Dashboard

```
cd ../admin-dashboard
npm install
```

- Create a `.env` file in the `admin-dashboard` directory with the following variables:

  ```
  VITE_API_ENDPOINT=http://localhost:5000
  ```

- Run the React.js admin dashboard:

```
npm run dev
```

Contact Information

- **Email:** badgujarmanish999@gmail.com
- **WhatsApp:** 7066619942