🚌 Bus Ticket Booking System (Backend Project Requirement)

🎯 Project Overview

এই প্রজেক্টের উদ্দেশ্য হলো একটি অনলাইন বাস টিকিট বুকিং সিস্টেম তৈরি করা যেখানে ইউজার রুট ও বাস অনুযায়ী টিকিট বুক করতে পারবে এবং অ্যাডমিন পুরো সিস্টেম ম্যানেজ করবে।
Backend তৈরি হবে Node.js + Express + TypeScript + MongoDB (Mongoose) দিয়ে, সাথে থাকবে Zod validation, JWT authentication, এবং bcrypt password security।

⚙️ Technology Stack

Language: TypeScript
Framework: Express.js
Database: MongoDB (Mongoose)
Validation: Zod
Security: bcrypt, JWT
Payment Simulation: Dummy/Offline Payment Method

🧩 Main Modules

1. User Module
2. Admin Module
3. Route Module
4. Bus Module
5. Booking Module
6. Payment Module

👩‍💻 1. User Module

Features:

- Register new user (ID, name, email, phone, password)
- Secure password with bcrypt
- Login with JWT
- View profile and booking history

API Endpoints:

- POST /api/users/register → Register new user
- POST /api/users/login → Login and get token
- GET /api/users/profile → Get user info (Protected)
- GET /api/users/bookings → Get user's booking history

👨‍💼 2. Admin Module

Features:

- Admin login
- Manage buses, routes, users, and bookings
- Dashboard summary (total users, bookings, income)

API Endpoints:

- POST /api/admin/login → Admin login
- GET /api/admin/summary → Dashboard summary (Protected)

🗺️ 3. Route Module

Description:
প্রতিটি রুটে থাকবে origin, destination, এবং distance.
Bus model এ এই route reference হিসেবে থাকবে।

Fields:
{
"_id": "ObjectId",
"origin": "Dhaka",
"destination": "Chittagong",
"distance": 250
}

API Endpoints:

- POST /api/routes → Add route (admin only)
- GET /api/routes → Get all routes
- GET /api/routes/:id → Get single route
- PUT /api/routes/:id → Update route
- DELETE /api/routes/:id → Delete route

🚌 4. Bus Module

Description:
প্রতিটি বাস একটি নির্দিষ্ট route এর reference নেবে।
একটি বাসের নির্দিষ্ট fare, total seats, এবং schedule থাকবে।

Fields:
{
"_id": "ObjectId",
"busName": "Green Line Express",
"busNumber": "GLX-102",
"route": "ObjectId (ref: Route)",
"totalSeats": 40,
"availableSeats": 40,
"fare": 1200,

"departureTime": "10:00 AM",
"arrivalTime": "4:00 PM"
}

API Endpoints:

- POST /api/buses → Add new bus (admin only)
- GET /api/buses → Get all buses
- GET /api/buses/:id → Get bus details
- PUT /api/buses/:id → Update bus info
- DELETE /api/buses/:id → Delete bus

🎟️ 5. Booking Module

Description:
User একটি নির্দিষ্ট তারিখে নির্দিষ্ট বাসের সিট বুক করবে।
সিট বুক হওয়ার পর availability কমে যাবে।

Fields:
{
"_id": "ObjectId",
"userId": "ObjectId (ref: User)",
"busId": "ObjectId (ref: Bus)",
"seatsBooked": [5, 6, 7],
"journeyDate": "2025-10-15",
"totalFare": 3600,
"paymentId": "ObjectId (ref: Payment)",
"status": "confirmed"
}

API Endpoints:

- POST /api/bookings → Create booking (user only)
- GET /api/bookings → Get all bookings (admin only)
- GET /api/bookings/:id → Get booking details

- PUT /api/bookings/:id/status → Update booking status (admin)

💳 6. Payment Module

Description:
User টিকিট বুক করার সময় একটি payment method বেছে নেবে (যেমন: Cash, Card, Mobile Banking)।
এটা একটি dummy/local payment simulation হবে।

Fields:
{
"_id": "ObjectId",
"userId": "ObjectId (ref: User)",
"bookingId": "ObjectId (ref: Booking)",
"amount": 3600,
"method": "Mobile Banking",
"transactionId": "MBL12345",
"paymentStatus": "success",
"paymentDate": "2025-10-15T10:30:00Z"
}

API Endpoints:

- POST /api/payments → Make payment for booking
- GET /api/payments/:id → Get payment details
- GET /api/payments/user/:userId → Get user's all payments