👩‍💻 Assignment Title:

Full-Stack Payment Dashboard System using NestJS & Flutter

🎯 Objective:

Design and build a secure, real-time **Payment Management Dashboard** for admins to:

- View and filter transactions
- Add and manage users
- Simulate payment flows
- View reports (e.g., revenue trends, failed transactions)

📦 Tech Stack

Layer	Tech
Frontend	Flutter (Mobile)
Backend	NestJS (Node.js)
Database	PostgreSQL or MongoDB
Auth	JWT (OAuth optional)
Optional	WebSockets, Redis, Pub/Sub

🔧 Features to Build

📲 Frontend (Flutter)

1. Login Screen

- Login with username/password
- JWT stored securely
-

2. Dashboard

- Show:
    - Total transactions today/this week
    - Revenue generated
    - Failed transactions
    - A simple revenue line chart
- Use fake/mock payment data for UI testing

3. Transactions Page

- Paginated list of all transactions
- Filter by:
    - Date range
    - Status (success, failed, pending)
    - Payment method
- Click to see details

4. Add Payment Page (simulate)

- Form to simulate a new payment
    - Amount, method, receiver, status
- POSTs to NestJS backend

🧠 Backend (NestJS)

1. Auth Module

- `POST /auth/login`
- JWT-based
- One hardcoded admin user is okay

2. Payments Module

- `GET /payments`: List with filter + pagination
- `GET /payments/:id`: Details
- `POST /payments`: Create a payment (simulate)
- `GET /payments/stats`: Metrics for dashboard

3. Users Module

- `GET /users`: Admin list
- `POST /users`: Add new intern/admin
- Add roles (`admin`, `viewer`)

🧪 Bonus Features (if time permits)

- WebSocket support for **real-time updates**
- Google Cloud Pub/Sub or Redis event queue
- Export transactions as CSV
- Unit + E2E testing (e.g., Jest for backend)

🎓 Learning Goals

Skill	Description
API Design	Build RESTful routes with filtering & validation
Auth	Secure access via JWT, role-based access
State Management	Frontend data binding, filters
Charts	Use Flutter chart libs (e.g., `fl_chart`)
Clean Architecture	Modular NestJS & widget-based Flutter
DevOps (Bonus)	Deploy on GCP/AWS or Docker

📁 Suggested Folder Structure

📦 Backend (NestJS)

```
src/
├── auth/
├── users/
├── payments/
├── common/
```

📦 Frontend (Flutter)

```
lib/
├── views/
├── services/
├── models/
├── widgets/
├── screens/
```

📤 Submission Guidelines

- Host backend on [Render/GCP/Glitch] or local with Postman
- Flutter frontend (Web or Android)
- Provide a GitHub repo with:
    - `README.md` (setup + screenshots)
    - DB schema or SQL dump
    - Sample login credentials
    - Sample payment data (seed)

🗓️ Deadline

- **2-3 days** expected timeline.