Task 1 – Instagram Mini Clone

Task: Build a Mini Instagram-Style Backend

Create a basic backend system and frontend similar to Instagram with the following features:

Backend Requirement:

1. User Authentication

- User signup
- User login
- Password hashing
- Return a token/session for authenticated routes

---

2. Follow System

- A user should be able to follow another user
- A user should be able to unfollow
- You should maintain proper relationships

---

3. Post Creation

- Authenticated users can create posts
- A post should have:
    - Image URL (string)
    - Caption (string)

---

4. Likes

- Users should be able to like a post
- Users should be able to unlike a post

---

5. Comments

- Users should be able to comment on posts
- Comments should show who commented and what they wrote

---

6. Feed

- Create an API to fetch a “feed” of posts
- Feed must show posts created by the users whom the logged-in user follows

---

Frontend Requirements:

You may use **React, Next.js, Angular, or Vue** (React recommended and Nextjs).

Screens to build:

Screen	Features
Login & Signup	Store token securely, redirect on login
Home Feed	List of posts with image, caption, likes, and comments
Create Post	Form to add new post (image URL + caption)
Profile Page	User posts, follower/following count, Follow/Unfollow button
Post Detail	Full view, interactive like/comment UI

Frontend Features:

- Use Fetch API or Axios for API calls
- Display data dynamically
- Update UI without page refresh (state management)
- Basic clean responsive design

Task 2 - Fit Plan Hub

A unique backend and frontend project focused on trainers, fitness plans, and paid subscriptions.

Task: FitPlanHub – Trainers & Users Platform

Create a backend for **FitPlanHub**, where certified trainers create fitness plans and users purchase & follow these plans.

1. User & Trainer Authentication

- Signup & login for both trainers and regular users
- Password hashing & token authentication

---

2. Trainer Dashboard – Create Fitness Plans

Trainers should be able to:

- Create a fitness plan
- Plan includes:
    - Title (e.g., “Fat Loss Beginner Plan”)
    - Description
    - Price (numeric)
    - Duration (e.g., 30 days)
- Edit or delete their own plans

---

3. User Subscriptions

Users should be able to:

- View all available fitness plans
- Purchase/subscribe to a plan (simulate payment, no real gateway required)
- After subscribing, they gain access to the plan

---

4. Access Control

- Only **subscribed users** can view plan details
- Non-subscribers should only see preview fields:
    - Title
    - Trainer name
    - Price

---

5. Follow Trainers

Users can:

- Follow/unfollow trainers
- View list of trainers they follow

---

6. Personalized Feed

After login:

- Show all plans created by trainers the user follows
- Show which plans the user has purchased
- Include basic trainer info in each feed item

<u>Frontend Requirements:</u>

Develop a UI that interacts with the backend.

Required Screens:

Screen	Functionality
Landing Page	Show all plans with previews

- **Login / Signup** Store token and redirect
- **Trainer Dashboard** CRUD operations on plans
- **Plan Details Page** Preview or full view based on subscription
- **User Feed** Personalized list of plans from followed trainers
- **Trainer Profile** Follow/unfollow + list of plans

Functional Expectations:

- Role-based UI (User vs Trainer)
- Subscribe button (simulate payment)
- Conditional rendering based on access (preview vs full access)

Additional Details

What Students Must Deliver

- Database design (their own schema)
- API design
- Node.js project with working endpoints
- Postman collection in Git (recommended)
- Proper README explaining how to run the project

What This Tests

- Backend logic
- DB relationships (one-to-many, many-to-many)
- Authentication
- CRUD operations
- Code structure & clarity

Recommendations

1. README File

Please create a clear README explaining the project setup and how to run it. Include a brief overview of the features you implemented.

---

2. Avoid AI-Generated Code

We will use AI-detection tools like GPTZero/Copilot Analyzer to check your code. Write the code yourself to avoid detection and to showcase your real skills.

---

3. Do Your Own Research

Use documentation and online resources to solve problems on your own. Your code should reflect your understanding, not AI output.