**Backend Task - Project Management System (NestJS)**

Objective:

Build a Project Management System backend using NestJS with:

* SQL database (PostgreSQL or MySQL)
* TypeORM
* REST API endpoints
* Security layer: JWT authentication and role-based authorization
* Advanced relational database schemaRequired Features:

1. Authentication

* JWT-based login & registration
* Password hashing using bcrypt
* Public routes: login, register
* Protected routes: all others

2. User Roles

* Roles: Admin, Manager, Developer
* Use role-based guards to restrict access to certain routes

3. Entities and Relationships

User:

* Can have many tasks
* Can be assigned to multiple projects (many-to-many)

Project:

* Can have many users (team members)
* Can have many tasks

Task:

* Belongs to a project
* Assigned to a user- Can have many comments

Comment:

* Belongs to a task- Includes author and content

Make sure to implement:

* One-to-Many
* Many-to-One
* Many-to-Many- Join tables where appropriate
* API Requirements (Sample):

User

* POST /auth/register
* POST /auth/login
* GET /users/me
* GET /users/:id/tasks (all tasks assigned to a user)

Project

* POST /projects (Admin/Manager only)
* GET /projects/:id
* PATCH /projects/:id
* DELETE /projects/:id
* POST /projects/:id/add-user/:userId

Task

* POST /projects/:projectId/tasks
* GET /tasks/:id
* PATCH /tasks/:id
* DELETE /tasks/:id

Comment

* POST /tasks/:taskId/comments
* GET /tasks/:taskId/comments

Security

* Use JWT for authentication
* Secure endpoints using guards
* Implement custom decorators and guards for role-based access
* Validate tokens and check role claims

Extra Credit (Optional)

* Pagination and filtering for tasks/comments
* Assign priority/status to tasks
* Soft delete for tasks/comments
* Audit fields (createdAt, updatedAt)