**NestJs Training - Assignment**

Task Management System

Let's create a simple task management system using the knowledge we have learnt so far. The system should have the following components:

**User:**

1. User Module: This module will handle user registration.
2. User service: Service to perform following operations on a user.
   1. Create a new User
   2. Get User Details
   3. Update User
3. User Controller: A controller that handles incoming HTTP requests related to user operations. Use an appropriate HTTP method for each request. User controller must implement following methods:
   1. Register a User
   2. Update User
   3. Get User Details
4. User Entity: A data model representing a user with the following properties:
   1. User Id (Should be UUID)
   2. First Name
   3. Last Name
   4. Email
   5. Username
   6. Password
   7. Created At
   8. Updated At

**Auth:**

1. Auth Module: This module will handle authentication.
2. Auth Service: A service to perform login and authentication. Use JWT for authentication.
3. Auth Controller: A controller that handles login requests.

Task:

1. Task Module: This module will handle all CRUD operations for tasks.
2. Task Entity: A data model representing a task with properties like I
   1. Task Id (should be UUID)
   2. Title
   3. Description
   4. Status
   5. Due Date
   6. Created At
   7. Updated At
3. Task Service: A service to perform CRUD operations on tasks. Service should allow a user to perform following operations
4. Task Controller: A controller that handles incoming HTTP requests related to task operations. Use an appropriate method for each request. Task Controller must implement following methods:
   1. Get tasks list
   2. Get Task by ID
   3. Update Task
   4. Create New Task
   5. Delete Task
   6. Get tasks by status or due date or due date within a date range.

Notes:

1. Implement Auth Guard to authenticate a request.
2. User should have visibility to his data only.
3. Add validation on each API and use Pipes for validation.
4. Use Exceptions and Custom Exception Filter to return any error such as un-authorized or resource not found etc.
5. Use any database you are comfortable with for data persistence.
6. Add Tests cases for each controller.