C# API Development Project — Grading Rubric

Instructor: Anthony Ackah-Nyanzu

1. Project Architecture – 4 pts

Follow a clean 4-project structure: API, Service, Data Layer (Dapper), and Database project. Logic should be properly separated.

2. Authentication & Authorization – 4 pts

Implement JWT or similar authentication. Protect certain routes with role-based or policy-based authorization.

3. Dependency Injection & Interfaces – 3 pts

Use interfaces for all services and repositories. Register them in .NET Core’s built-in dependency injection container.

4. Data Layer (Dapper & SQL) – 3 pts

Use Dapper for database access. Write parameterized SQL queries or stored procedures. No SQL in controllers or services.

5. Functionality & API Endpoints – 4 pts

Implement CRUD and any other necessary endpoints. Return JSON responses with proper HTTP status codes.

6. Code Quality & Design – 2 pts

Follow C# naming conventions, use async/await where appropriate, and ensure clean, well-commented, and readable code.

7. Documentation & Testing – 2 pts

Include a README with setup instructions, Swagger/OpenAPI documentation, and basic testing if possible.

Category	Points
Project Architecture	4
Authentication & Authorization	4
Dependency Injection & Interfaces	3
Data Layer (Dapper & SQL)	3
Functionality & API Endpoints	4
Code Quality & Design	2
Documentation & Testing	2
Total	20

Key Reminders for Students:
• Do not write SQL in controllers or service classes — use repositories with Dapper.
• Always parameterize queries (no string concatenation).

• Use proper HTTP status codes (200, 201, 400, 401, 404, etc.).
• Keep logic isolated in its correct layer — API ↔ Service ↔ Repository ↔ Database.
• Document your setup and include working examples (via Swagger or Postman).