# Scenario:

You’ve been hired to build a lightweight User Authentication and Profile Management microservice for a larger distributed system. This service will manage user registration, login, token-based authentication, and profile information.

# Task: Build a Sample Authentication API

## Features to Implement:

1. **User Registration**
   * Accept email, password, and name.
   * Return an appropriate response if the email is already registered.

## Login

* + Accept email and password.
  + Return a JWT token on success.
  + Return proper error codes on failure (e.g., invalid credentials).

## Get Current User Profile

* + Secured endpoint requiring a valid JWT.
  + Returns user's name and email.

## Update Profile

* + Allows changing the user's name.
  + Secured with JWT.

# Architectural Requirements:

* Apply Clean Architecture principles (Separation between API layer, Application layer, Domain, and Infrastructure).
* Use **Domain-Driven Design (DDD)** concepts where applicable (e.g., Value Objects, Aggregates, Domain Services).
* Handle errors with custom exceptions like AuthException.

# Technical Requirements:

* .NET 6 or later
* Entity Framework Core (In-Memory or SQLite)
* JWT-based authentication
* Minimal API or Controller-based (candidate’s choice)
* Clean, maintainable, testable code
* Use Dependency Injection appropriately

# Bonus (Optional):

* Email verification flow (using a dummy token or a mock email sender)
* Refresh token support
* Unit tests for domain logic
* Swagger/OpenAPI documentation

# Evaluation Criteria:

|  |  |
| --- | --- |
| **Criteria** | **Description** |
| **Code Structure** | Clean Architecture, project structure, and organization |
| **Domain Modeling** | Use of domain models, aggregates, value objects |
| **Authentication Handling** | Secure and proper JWT handling, middleware usage |
| **Exception Management** | Custom exceptions with proper HTTP response mapping |
| **Code Quality** | Readability, testability, naming, DRY principles |
| **API Design** | RESTful principles, request/response consistency |

**Criteria Description**

|  |  |
| --- | --- |
| **Extensibility** | Can this be extended easily for more features? |
| **Bonus Features** | Extra effort on unit tests, refresh tokens, etc. |

# Deliverables:

* GitHub repo or zip file containing:
  + Source code
  + README with setup instructions
  + Sample requests (Postman collection or cURL)