Day 4

? Day 4 ? Refactor to Clean Architecture, Repository Pattern, and Unit of Work

Objective

Refactor the application to follow Clean Architecture principles. Introduce the Repository Pattern and implement the Unit of Work pattern to manage data persistence more effectively.

Tasks

1. Apply Clean Architecture Structure

- Restructure the solution into clearly defined layers:
  - Domain (entities, value objects, business rules)
  - Application (CQRS handlers, interfaces, validation)
  - Infrastructure (database, external services)
  - Presentation/API (minimal API, controllers if applicable)
- Ensure clear dependency flow:
  - Domain has no dependencies.
  - Application depends only on Domain.
  - Infrastructure depends on Application and Domain.
  - API depends on all layers via interfaces.

2. Introduce the Repository Pattern

- Create interfaces for data access at the application layer (e.g., IProductRepository, IOrderRepository).
- Implement the interfaces in the infrastructure layer using EF Core.
- Replace direct use of DbContext in handlers with injected repository interfaces.

Day 4

3. Implement Unit of Work Pattern

- Create a IUnitOfWork interface that includes repositories and a SaveChangesAsync method.

- Use the Unit of Work in command handlers to group related data operations.

- Ensure transactional integrity across operations within a single use case.

4. Ensure Business Rules Are Isolated

- Move business rules into the domain layer using methods or domain services.

- Avoid placing business logic inside the infrastructure or API layers.

- Validate that all business decisions are made within the domain or application layers.

5. Final Cleanup and Review

- Remove any obsolete code or unused services.

- Verify that all functionality from previous days still works.

- Ensure layers are correctly enforced and the codebase is maintainable.