ANALISIS USERSERVICE

Nombre:

Juan Carlos Berdugo Gómez

FABRICA DE SOFTWARE

Dirigido:

JEFE E INSTRUCTORES

CENTRO DE GESTION DE MERCADOS LOGISTICA Y TECNOLOGIAS DE LA INFORMACION

1. Identifica correctamente las capas de Clean Architecture:

SI

El proyecto sigue la Clean Architecture, estructurando el código en las siguientes capas dentro de userservice/app/:

domain: Define entidades, value objects, excepciones y las interfaces de los repositorios.

application: Contiene los casos de uso (use cases) y servicios de aplicación.

infrastructure: Implementa detalles técnicos como la conexión a la base de datos, configuración y repositorios concretos.

presentation: Expone la API REST mediante routers y define los esquemas de validación (Pydantic).

Esta organización permite independencia, testabilidad y escalabilidad.

2. Reconoce las entidades del dominio:

SI

Las entidades principales del dominio están en app/domain/entities/:

User: Representa al usuario del sistema, con atributos como id (UUID), email, nombre, contraseña, roles, estado, etc.

Role: Define los roles de usuario, permitiendo control de acceso y permisos.

Estas entidades encapsulan la lógica de negocio y no dependen de frameworks externos.

3. Identifica los repositorios y sus interfaces:

SI

Las interfaces de repositorio están en app/domain/repositories/, por ejemplo, UserRepositoryInterface.

Estas interfaces definen los métodos necesarios para interactuar con las entidades (buscar, guardar, actualizar, eliminar).

Las implementaciones concretas se encuentran en app/infrastructure/repositories/, desacoplando la lógica de negocio de la tecnología de persistencia.

4. Analiza los use cases principales:

SI

Los casos de uso de autenticación están en app/application/use_cases/auth_use_cases.py e incluyen: LoginUseCase, RefreshTokenUseCase, LogoutUseCase, ValidateTokenUseCase, ForgotPasswordUseCase, ResetPasswordUseCase, ForceChangePasswordUseCase.

Cada caso de uso orquesta la lógica necesaria, interactuando con entidades, value objects, servicios y repositorios, siguiendo el principio de responsabilidad única.

5. Verifica la implementación de autenticación y JWT:

SI

La generación y validación de JWT se realiza en los casos de uso de autenticación. Se implementa access token y refresh token, ambos con expiración y claims seguros. La validación de tokens se realiza mediante dependencias/middlewares en la capa de presentación, asegurando endpoints protegidos. Se siguen buenas prácticas: expiración corta para access tokens, rotación y persistencia de refresh tokens, y revocación en logout o cambio de credenciales.