UNIVERSIDAD LAS PALMAS DE GRAN CANARIA

PROGRAMACIÓN DE APLICACIONES MÓVILES NATIVAS

INFORME SEMANA 6:

DISEÑO DE LA BASE DE DATOS

PRESENTADO POR: IVONNE YULIETH OJEDA CUERVO

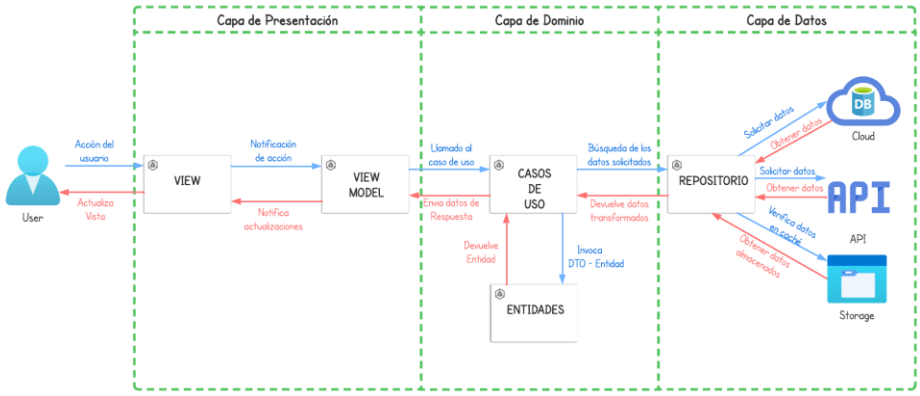
AZ930245

LAS PALMAS DE GRAN CANARIA – 2023

DISEÑO DE LA BASE DE DATOS

1. Identifica y resalta los modelos y componentes relacionados con la capa de persistencia.

Revisando nuevamente el modelo de arquitectura propuesto para el proyecto:



Identificamos claramente que los componentes de Repositorio y Entidades son los directamente relacionados con la capa de persistencia debido a que:

Los Repositorios son responsables de interactuar con el almacenamiento de datos y pueden realizar la transformación de datos para asignar datos entre el formato sin procesar almacenado en la fuente de datos y las entidades utilizadas en los casos de uso.

Y Las Entidades son compartidas entre los casos de uso y los repositorios y sirven como puente para mantener la representación de datos consistente.

2. Selecciona una tecnología para la base de datos de tu aplicación (Room, Firebase, SQLite, Realm, DataStore ...etc). Justifica brevemente tu elección, considerando las necesidades de tu aplicación.

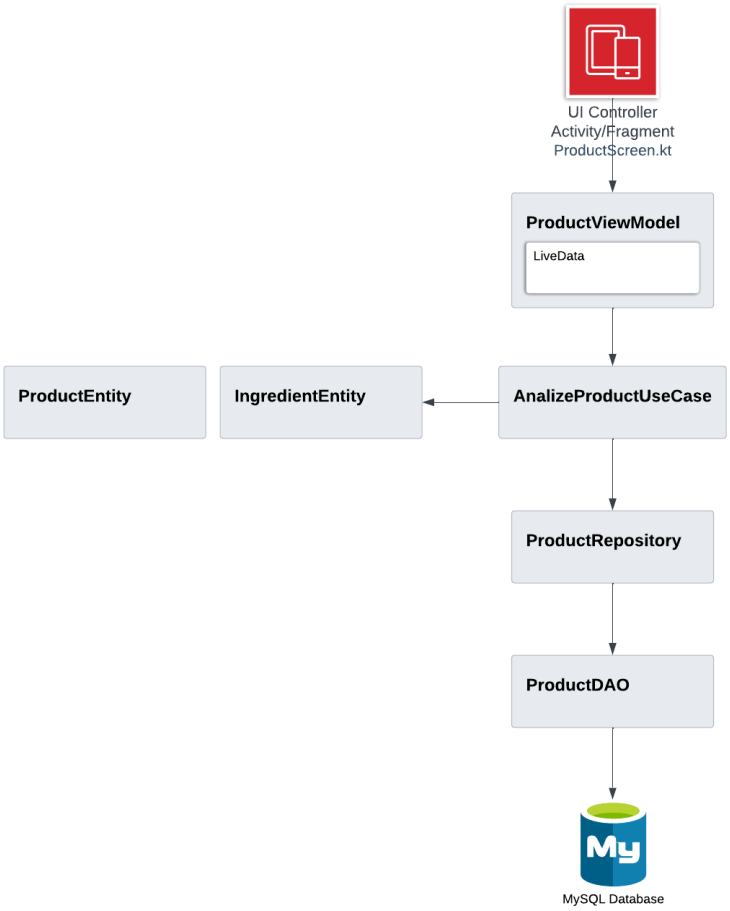
Dado el propósito de almacenar información de ingredientes consultados, reduciendo la carga en la API y promoviendo la participación de la comunidad en reseñas, resulta fundamental una base de datos centralizada. La estructura de datos claramente definida y las relaciones directas subrayan la importancia de una base de datos relacional. Por lo tanto, se ha optado por MySQL, un sistema de gestión de bases de datos de código abierto que brinda ventajas clave:

- Escalabilidad: MySQL se adapta eficazmente a las necesidades cambiantes, lo que es esencial para proyectos en evolución, permitiendo un manejo óptimo del crecimiento de datos y tráfico.
- Rendimiento Eficiente: MySQL se destaca por su capacidad para ejecutar consultas rápidamente, lo que garantiza una respuesta ágil y una experiencia de usuario mejorada.
- Código Abierto: MySQL es de código abierto, respaldado por una comunidad global de desarrolladores y usuarios, lo que promueve la colaboración y asegura la calidad y seguridad del sistema con actualizaciones continuas.

3. Define cómo se integrará la lógica que gestionará la capa de persistencia en tu proyecto. Puedes complementar el diagrama empleado en la actividad sobre la arquitectura de tu aplicación. Esto puede incluir entidades, atributos, relaciones, DAOs, repositorios.

   Importante: bastaría con un sólo módulo de la aplicación

El siguiente modelo refleja la persistencia de la funcionalidad que analiza los ingredientes de un producto cosmético:



En este ejemplo:

AnalizeProductUseCase define la lógica empresarial de analizar un producto. Tiene el método analizar producto que recibe una lista de ingredientes y retorna un producto con una lista de ingredientes y sus caracteristicas.

ProductEntity contiene el nombre del producto y una lista de ingredientes.

Ingrediententity contiene un nombre, función cosmética, indicador EWR, indicador CIR y efectos notables.

ProductRepository implementa AnalizeProductUseCase y llama a ProductDAO que realiza la consulta a la BD

La interfaz de usuario se implementa en ProductScreen.kt.

ViewModel, ProductViewModel, conecta la interfaz de usuario con la lógica empresarial.

4. Reflexiona sobre cambios futuros:

- ¿Qué pasa si en un futuro se quisiera cambiar el motor de base de datos?

En resumen, con una arquitectura adecuada como MVVM/Clean Architecture y buenas prácticas de diseño, cambiar el motor de base de datos debería ser manejable. Aunque requerirá trabajo en la capa de persistencia y pruebas exhaustivas, la capa de dominio y la lógica de la aplicación en general no deberían verse afectadas en gran medida por este tipo de cambio.

- ¿Qué partes de tu aplicación tendrías que modificar?

Tendría que modificar el repositorio y el DAO.

- ¿Qué dificultades anticipas?

Si la implementación de la arquitectura se realizó correctamente no debería existir mayor dificultad más allá de lo complicado que sea conectarse a la nueva base de datos o cambiar la sintaxis de las consultas si el motor tiene un lenguaje de consulta específico.

- ¿Cómo podrías diseñar tu aplicación para minimizar el impacto de tal cambio?

La razón principal de optar por la arquitectura MVVM/Clean es precisamente reducir al máximo el impacto de cualquier cambio, incluyendo aquellos relacionados con el motor de la base de datos.

BIBLIOGRAFÍA

[1]“Clean Architecture: The Key to Modular and Testable Android Apps - Tech Tips with Pinar,” Aug. 29, 2023. https://pinartechtips.com/clean-architecture-the-key-to-modular-and-testable-android-apps/ (accessed Nov. 02, 2023).

[2]Rivaldy, “Room Database with MVVM Architecture | Android Jetpack | CRUD,” Medium, Oct. 29, 2021. https://rivaldy.medium.com/room-database-with-mvvm-architecture-android-jetpack-crud-25dd0f476514 (accessed Nov. 02, 2023).