Diseño arquitectónico de una aplicación

Antonio Medina Santana y Carlos Rodríguez del Toro

1.- Definición del Dominio:

La aplicación que se va a desarrollar es una de escaneo de códigos de barras de libros.

La aplicación da la posibilidad de escanear los códigos, mostrando toda la información referente a cada libro escaneado.

También se puede optar por enviar la información de la obra a amigos (que se pueden añadir desde la propia aplicación), ver su disponibilidad en tiendas, agrupar por categorías cada libro escaneado según si está pendiente de lectura o si está en favoritos.

Por la variedad de los elementos y funcionalidades de la aplicación, creemos que la arquitectura MVVM es la más cómoda para trabajar en este proyecto, bajo nuestro punto de vista.

Teniendo en cuenta que necesitamos trabajar con la lógica de negocio de forma independiente, ya sea el uso de la cámara para escanear el código, analizar la información, generar los registros de los libros añadidos a la biblioteca, creemos que tenerlo todo separado con responsabilidades únicas es lo más correcto y cómodo.

También trabajar con todos estos datos para mostrarlos creemos que es costoso, pero MVVM nos permite que la vista se actualice automáticamente solo cambiando el modelo, y esperando a que la vista de modelo haga los cambios en consecuencia, obteniendo la vista que queremos.

Por todo esto, pensamos que nuestra aplicación, con las características que tiene, puede aprovechar mejor una arquitectura MVVM sobre las demás.

2.- Identificación de los principales elementos de la arquitectura elegida:

• Ejemplo: en el caso de la arquitectura hexagonal identificar los diferentes puertos que la aplicación necesitará (por ejemplo, acceso a datos, servicios externos, interfaces de usuario) y los adaptadores correspondientes.

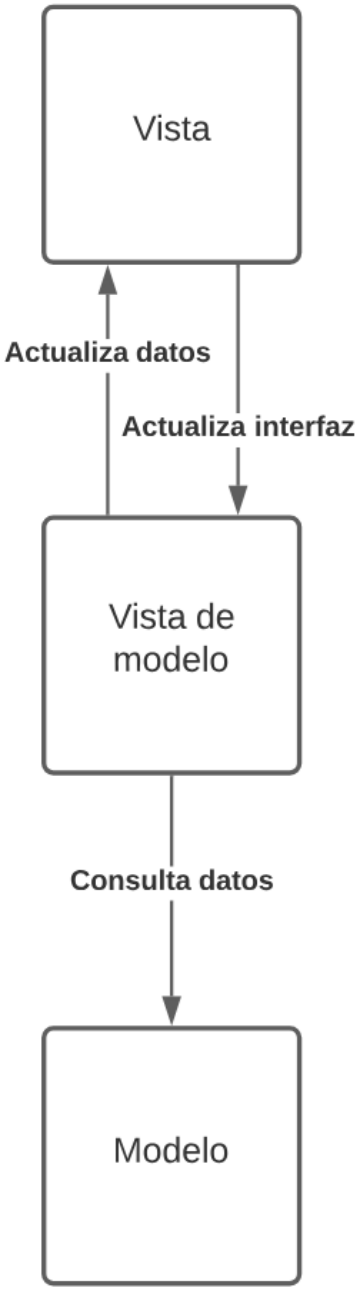
Modelo: en este caso, todo lo relacionado con el uso de la cámara y detección de códigos, almacenamiento de los códigos que se han escaneado, creación y mantenimiento de cuentas, entre otras cosas.

Vista: se encargará de mostrar toda la interfaz de usuario, comunicando a la vista de modelo cuando se comparta un libro, o se quiera añadir un amigo, o revisar la información de un título de la biblioteca.

Vista de modelo: Afectará a la vista, utilizando la lógica del negocio, usando todos los datos generados para mostrar en pantalla el código escaneado, junto a la información que se obtiene de este, o utilizará los datos modificados por la vista para trabajar a partir de la lógica de negocio.

3.- Diseño de la Arquitectura:

• Dibujar un diagrama que muestre los principales elementos de la arquitectura elegida.



4.- Caso de Uso:

• Elegir uno o dos casos de uso principales de la aplicación.

• Describir cómo se procesarían estos casos de uso a través de la arquitectura seleccionada

CU1: Escanear un código de barras de libro

1. El usuario abre la aplicación y selecciona la opción de escanear un código de barras de un libro.
2. La vista captura la solicitud del usuario y la envía al ViewModel.
3. El ViewModel inicia la cámara y procesa la captura del código de barras.
4. Una vez que se escanea con éxito el código de barras, el ViewModel se comunica con el modelo para obtener información sobre el libro relacionado con el código escaneado.
5. El Modelo consulta la base de datos o un servicio externo para obtener información detallada sobre el libro.
6. El Modelo devuelve la información al ViewModel.
7. El ViewModel actualiza la Vista con los detalles del libro, como el título, el autor, la portada, etc.
8. El usuario ve la información del libro en la interfaz de usuario.

CU2: Compartir un libro con un amigo

1. El usuario navega a la sección de su biblioteca en la aplicación y selecciona un libro.
2. La vista captura la solicitud del usuario y la envía al ViewModel.
3. El ViewModel procesa la solicitud y recupera la información del libro seleccionado del modelo.
4. El usuario selecciona la opción de compartir el libro con un amigo.
5. El ViewModel maneja la solicitud de compartir y, si es necesario, obtiene la lista de amigos del modelo.
6. El Modelo maneja la lógica de compartir y envía la información al amigo seleccionado o guarda la acción para ser realizada más tarde.
7. El ViewModel actualiza la Vista para reflejar el éxito o el resultado de la acción de compartir.

5.- Conclusión:

• Reflexionar sobre las ventajas de utilizar la arquitectura seleccionada en comparación con otros patrones arquitectónicos.

Para nuestra aplicación, creemos que utilizar la arquitectura MVVM nos ofrece varias ventajas:

- Separación de responsabilidades: MVVM permite separar claramente las responsabilidades de todas sus partes. La vista se encarga de la interfaz de usuario, la vista de modelo maneja la lógica de presentación y el modelo la lógica de negocio. Así facilitamos el desarrollo y el mantenimiento del proyecto.
- Reactividad: MVVM da mayor facilidad a la hora de actualizar la interfaz de usuario cuando cambian los datos de forma automática. La vista de modelo observa los cambios en el modelo y actualiza la vista, lo que proporciona una experiencia de usuario fluida.
- Flexibilidad y facilidad para las pruebas: MVVM da la posibilidad de probar por separado cada componente. Puedes realizar pruebas unitarias en el ViewModel sin depender de la interfaz de usuario, lo que facilita la detección de errores y la corrección.