UNIVERSIDAD DON BOSCO

FACULTAD DE INGENIERÍA

Desarrollo de Software Móviles

Trabajo de Investigación – Patrón MVVM

Laboratorio

O4L

Rodríguez Salazar, Rocío Esmeralda RS200052

Vides Navas Rodrigo Josué VN191709

Ing. Alexander Alberto Sigüenza Campos

Soyapango, San Salvador 28/04/2023

Indice

INTRODUCCION

En la creación de software móviles se pueden presentar distintos inconvenientes, para ello se pueden encontrar los distintos modelos o patrones de arquitectura de software, estas ayudan a estructurar el software y estandarizarlo para volverlo escalable.

Uno de estos modelos es MVVM o Model View ViewModel, este se basa en separar la interfaz con la cual se relaciona el usuario en su caso View, y el modelo.

Dentro de la investigación se podrá analizar este modelo y como se utiliza dentro de un Software móvil, además de las ventajas de este.

[1]

[1] (KeepCoding, 2023)

PATRON MVVM

El patrón MVVM por sus siglas Model View ViewModel, es un patrón de diseño el cual se encarga de dividir la interfaz del usuario el cual sería View del modelo del software móvil “model”, esto facilita para el desarrollador los procesos de pruebas, evaluación y mantenimiento del software móvil. Parte de los beneficios de usar este patrón de arquitectura es que, al separar el proyecto, permite a los desarrolladores crear pruebas unitarias para el Model View y el modelo, sin hacer uso de la vista.

Además con el patrón mmvm, diseño y desarrollo pueden trabajar de manera simultánea e independiente, cada uno en sus componentes ya que mientras los diseñadores tienen la posibilidad de enfocarse en la vista, los desarrolladores pueden hacerse cargo de los componentes de la vista y del modelo de vista.

COMPONENTES DEL PATRON MVVM

El patron MVVM se divide en Model, View y ViewModel, se puede comprender a View como la parte visual o interfaz del usuario, luego Model sería el modelo o toda la parte lógica del negocio o software móvil en este caso, y por último el viewmodel sería el puente entre las dos anteriores o logica de presentacion.

Resumiendo la lógica del patrón MVVM se puede ver de la siguiente forma, el usuario interactua con view esta envia un evento a viewmodel para que esta envie la data a model, model da acceso a datos y viewmodel lo recive para enviarlo a View y que se muestre al usuario. [2]

[2] (Rodriguez, 2020)

IMPLEMENTACION DEL PATRON MVVM

Para implementar el patrón MVVM dentro de Android studio utilizando Kotlin se deben seguir una serie de pasos, los cuales son:

1. Crear un modelo de datos (Model): El modelo de datos representa la fuente de datos y la lógica de negocio de la aplicación. Este puede ser una clase simple de Kotlinque contenga las propiedades necesarias para almacenar y manipular los datos.

2. Crear un ViewModel: El ViewModel actúa como intermediario entre la vista y el modelo. Es responsable de proporcionar datos a la vista y controlar la lógica de la interfaz de usuario. Para crear el ViewModel, se debecrear una clase que extienda la clase ViewModel de Android y que contenga las propiedades y métodos necesarios para proporcionar datos a la vista.

3. Crear la Vista (View): La vista representa la interfaz de usuario de la aplicación. En Android, esto puede ser una actividad, un fragmento, una vista personalizada, entre otros. Los desarrolladores deben asegurarse de que la vista esté diseñada de tal manera que pueda enlazarse fácilmente con el ViewModel.

4. Conectar la Vista y el ViewModel: La vista y el ViewModel deben estar conectados de tal manera que la vista pueda solicitar datos del ViewModel y recibir actualizaciones cuando los datos cambien. En Android, esto se puede hacer utilizando la biblioteca de enlace de datos (Data Binding).

5. Implementar la Lógica de Negocios: Una vez que se haya establecido la conexión entre la vista y el ViewModel, se debe implementar la lógica de negocio de la aplicación en el ViewModel. Esto puede incluir la recuperación de datos de una API REST, el procesamiento de datos, la validación de entradas del usuario, etc.

6. Actualizar la vista: Finalmente, se tiene asegurar que la vista se actualice correctamente cuando cambien los datos en el ViewModel. Esto se hace mediante la vinculación de datos y el uso de métodos de actualización en el ViewModel.

VENTAJAS Y DESVENTAJAS DEL PATRON MVVM

Del mismo modo, este método puede generar ventajas y desventajas, las cuales pueden ser:

Ventajas

Separación de responsabilidades: MVVM permite separar la lógica de negocio de la aplicación de la interfaz de usuario (UI). Esto ayuda a mantener el código organizado y modular, lo que facilita la escalabilidad y el mantenimiento de la aplicación.

Pruebas unitarias: El patrón MVVM facilita la realización de pruebas unitarias porque separa la lógica de negocio de la interfaz de usuario. Esto significa que se pueden realizar pruebas en la lógica de negocio sin necesidad de una interfaz de usuario.

Reutilización de código: La separación de responsabilidades facilita la reutilización de código. Por ejemplo, se pueden utilizar los mismos modelos y ViewModel en diferentes vistas, lo que reduce la cantidad de código duplicado.

Enlace de datos: MVVM facilita el enlace de datos mediante el uso de enlaces de datos bidireccionales, lo que permite que la vista se actualice automáticamente cuando cambian los datos en el ViewModel.

Desventajas

Curva de aprendizaje: MVVM puede tener una curva de aprendizaje pronunciada para los desarrolladores que no están familiarizados con el patrón. Además, la implementación de MVVM puede ser más complicada que otros patrones de arquitectura.

Sobrecarga de código: A veces, la implementación de MVVM puede requerir una gran cantidad de código adicional, lo que puede hacer que el código sea más difícil de leer y mantener.

Dependencias: MVVM puede requerir el uso de bibliotecas adicionales para facilitar el enlace de datos y otras funcionalidades, lo que puede aumentar las dependencias de la aplicación.

Bibliografia

KeepCoding. (21 de abril de 2023). *KeepCoding-TechSchool*. Obtenido de ¿Qué es el patrón de arquitectura MVVM?: https://keepcoding.io/blog/que-es-el-patron-de-arquitectura-mvvm/

Rodriguez, E. (21 de Dec de 2020). *Inmediatum*. Obtenido de MVVM – Qué es y como funciona.: https://inmediatum.com/blog/ingenieria/mvvm-que-es-y-como-funciona/