Universidad Tecnológica de Tijuana

TSU en Tecnologías de la Información Área Desarrollo de Software Multiplataforma

Nombre:

García Arreola Howard Isaí

Subject:

Desarrollo movil Integral

Actividad:

Architecture specification

Grupo: 10-B

Profesor:

Ray Brunett Parra

Galaviz

Fecha de Realización:

January 7th, 2025

MVVM

Model-View-ViewModel (MVVM) is a software architectural pattern that facilitates the separation of the development of the graphical user interface from the business logic or back-end logic (the data model). The View Model in MVVM represents an abstraction of the View, which contains a View's state and behavior. I choose this architecture because I would like to use Jetpack Compose as the framework for mobile development; MVVM is widely use with compose and it makes the development of the app smoother.

MVVM components

Model

It represents the data and business logic of the application. It is responsible for retrieving data, processing it, and defining how the data can be changed and manipulated.

View

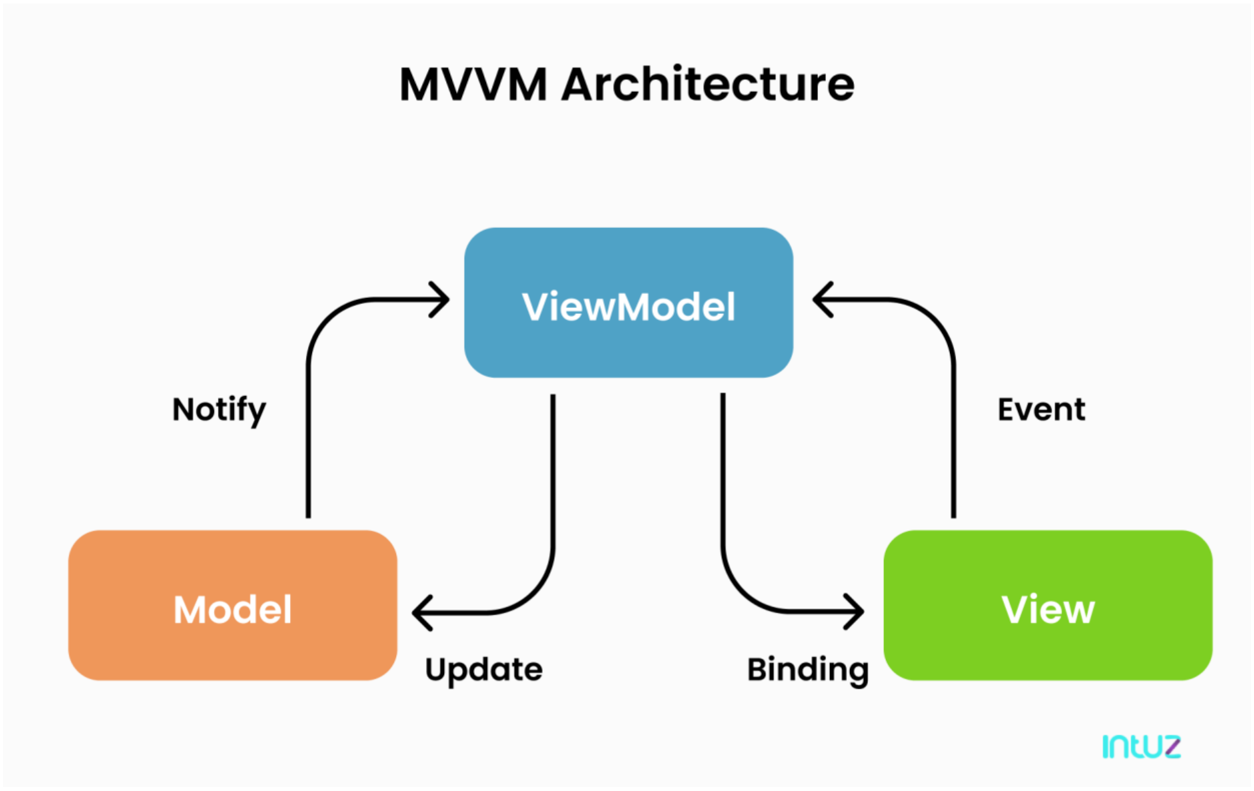
The View is responsible for defining the structure, layout, and appearance of what users see on the screen. Ideally, the View is purely declarative, meaning it defines "What" but not "How".

ViewModel

The ViewModel in MVVM architecture serves as a bridge between the Model and the View. It's responsible for handling the logic for the UI and acts as an abstraction of the View, which contains a View's state and behavior.

The ViewModel communicates with the Model and converts its data into a format that can be easily managed and presented by the View. This conversion includes implementing properties and commands to which the View can bind to, enabling the View to remain isolated from the complexities of the business logic or back-end logic.

In terms of managing the UI's state and behavior, the ViewModel exposes data-binding friendly properties and commands for the View to use. These properties and commands provide a way for the View to update in response to changes in the Model's state and react to user interactions, respectively. This means that the ViewModel handles all UI actions by using command patterns and can also manage navigation, dialog, and other UI services.



Advantages:

- Maintainability: It can remain agile and keep releasing successive versions quickly.
- Extensibility: It has the ability to replace or add new pieces of code.
- Testability: It's easier to write unit tests against a core logic.
- Transparent Communication: The view model provides a transparent interface to the view controller, which it uses to populate the view layer and interact with the model layer, which results in a transparent communication between the layers of your application.

Disadvantages:

- Some people think that for simple UIs, MVVM can be overkill.
- In bigger cases, it can be hard to design the ViewModel.
- Debugging would be a bit difficult when we have complex data bindings.

References:

GeeksforGeeks. (2023, November 1). *Introduction to Model View View Model (MVVM)*. GeeksforGeeks. https://www.geeksforgeeks.org/introduction-to-model-view-view-model-mvvm/

Goyal, A., & Goyal, A. (2024, June 18). *A comprehensive guide to mobile app architecture*. Offshore IT Services | IT Software Solutions Company. https://www.octalsoftware.com/blog/mobile-app-architecture-guide

Goyal, A., & Goyal, A. (2024, June 18). *A comprehensive guide to mobile app architecture*. Offshore IT Services | IT Software Solutions Company. https://www.octalsoftware.com/blog/mobile-app-architecture-guide