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:

Framework Selection

Grupo: 10-B

Profesor: Ray Brunett

Parra Galaviz ***Fecha de***

Realización:

January 6th, 2025

Overview of Jetpack Compose as a Framework

Jetpack Compose is Android's modern toolkit for building native UI. It's part of Android's Jetpack suite, designed to simplify and accelerate UI development while offering powerful tools and flexibility for developers. Instead of relying on the traditional XML-based UI design, Jetpack Compose uses a declarative approach, where you describe the UI's state, and the framework takes care of rendering the UI efficiently.

Features of Jetpack Compose

Declarative UI Design

- Compose allows you to define your UI in a declarative manner. This means you describe how the UI should look for a given state, and Compose automatically updates the UI when the state changes.
- Eliminates the complexity of manually updating views or managing layout hierarchies.

Simplified Development

- No need for XML layouts or findViewById. UI and logic are written in Kotlin, reducing boilerplate code.
- Integrates seamlessly with other Jetpack libraries and existing Android codebases.

State Management

- Built-in support for state management using State and MutableState.
- Works well with modern state containers like ViewModel and tools like Kotlin Coroutines and Flow.

Composable Functions

- The core building blocks in Compose are functions annotated with @Composable. These functions define UI components and can be nested to build complex interfaces.

Dynamic Layouts and Reusability

- Compose provides tools for creating dynamic and adaptive layouts that can adjust to screen sizes and orientations.
- Components are reusable, making it easier to maintain and scale the app.

Tooling and Debugging

- Strong integration with Android Studio provides features like Compose Preview, Live Edit, and interactive design tools.
- Makes it easier to visualize, tweak, and test UIs in real-time.

Theming and Material Design

- Supports Material Design principles out of the box with MaterialTheme.
- Allows customization of colors, typography, and shapes globally or for specific components.

Why did I choose Jetpack Compose?

I choose Jetpack Compose because it offers a modern, declarative approach to UI development that simplifies building complex user interfaces while reducing boilerplate code. Its seamless integration with the Android ecosystem and Kotlin ensures that I can leverage the full power of native Android development, including compatibility with well-established tools and libraries like ViewModels, LiveData, and Room. Additionally, the advanced tooling in Android Studio, like real-time previews and Live Edit, dramatically boosts productivity and streamlines the development process. Unlike cross-platform frameworks, Compose allows me to create apps that feel truly native, with full access to platform-specific features. With its intuitive state management and scalable architecture, Jetpack Compose enables me to build maintainable, high-performance applications that meet modern development standards.

References:

GeeksforGeeks. (2024, October 4). *Basics of Jetpack compose in Android*. GeeksforGeeks. https://www.geeksforgeeks.org/basics-of-jetpack-compose-in-android/

JetPack Compose UI App Development Toolkit - Android Developers. (n.d.). Android Developers. https://developer.android.com/compose