



Trabajo de investigación

Desarrollo de Software para Móviles DSM941 G01T

Interfaz de Usuario en Android Studio con

Jetpack Compose

Docente:

Ing. Alexander Alberto Sigüenza Campos

Integrantes:

William Alberto García Gómez - GG212522

Christian Yahir López Hernández - LH212531

Yahir Stewart Sibrian Arriola - SA212551

Alberto Joseph Mendoza Moreno - MM200462

Juan Jose Lue Valdez -LV231966

Fecha:

21 de abril de 2025

ÍNDICE

Introducción	3
¿Qué es Jetpack Compose?	4
Ventajas	5
- Menos código	5
- Enfoque declarativo	5
- Vista previa en tiempo real	5
- Mayor flexibilidad	5
- Mejor manejo de temas y estilos	5
- Integración con herramientas modernas	5
- Mayor velocidad de desarrollo	5
- Respaldo oficial y comunidad activa	6
Descripción del Flujo Implementado y diseño de la aplicación	6
Inicio de sesión (pantalla principal)	6
Validación de errores	7
Pantalla de bienvenida	7
Conclusiones	8
GitHub	9

Introducción

En el contexto del desarrollo de aplicaciones móviles, la construcción de interfaces de usuario claras, eficientes y adaptables se ha vuelto un aspecto esencial para garantizar una buena experiencia en el uso de software. Tradicionalmente, la creación de estas interfaces en Android se realizaba mediante el uso de archivos XML combinados con código en Java o Kotlin, lo cual generaba una separación estricta entre la presentación y la lógica, y en ocasiones, procesos poco flexibles.

Para responder a estas limitaciones, Google introdujo Jetpack Compose, un moderno toolkit declarativo que permite construir interfaces gráficas directamente en Kotlin. Compose facilita el diseño y la implementación de componentes de interfaz mediante una sintaxis más limpia, legible y adaptable, promoviendo un desarrollo más ágil y eficiente.

Esta tecnología forma parte del ecosistema de Android Jetpack, y ha sido diseñada para integrarse de forma natural con las herramientas modernas de desarrollo en Android Studio. Entre sus principales ventajas se destacan la reducción significativa del código requerido, la capacidad de visualizar cambios en tiempo real, el soporte nativo para temas y estilos, y la simplificación de la navegación entre pantallas.

En el marco de esta actividad académica, se ha utilizado Jetpack Compose para implementar una aplicación básica compuesta por una pantalla de inicio de sesión, con validación de datos, y una segunda pantalla de bienvenida que responde a la entrada del usuario. La tarea permite comprender los fundamentos de este enfoque declarativo, así como los principios de navegación y manejo de estados en entornos móviles.

¿Qué es Jetpack Compose?

Jetpack Compose es un framework desarrollado por Google que permite crear interfaces gráficas de usuario (UI) directamente desde el lenguaje de programación Kotlin. A diferencia del enfoque tradicional en Android, donde se usaban archivos XML para definir la estructura visual de las pantallas y luego se vinculaban con lógica en Java o Kotlin, Compose propone una forma diferente de trabajar: todo se hace desde el código, en un único lugar, utilizando funciones que describen cómo debe mostrarse la interfaz y cómo debe comportarse.

La idea principal de Jetpack Compose es que nosotros como desarrolladores podamos declarar el aspecto de la interfaz basándose en el estado de la aplicación. En lugar de escribir instrucciones paso a paso para construir o modificar una vista, lo que se hace es describir qué debe mostrarse dependiendo de ciertas condiciones, y Compose se encarga de actualizar la pantalla automáticamente cuando esos valores cambian. Esto reduce la cantidad de código que se necesita para mantener sincronizados los datos con lo que se ve en pantalla.

Compose está basado completamente en Kotlin, lo que lo hace más coherente con las prácticas modernas del desarrollo en Android, ya que ya no es necesario mezclar XML con código Java o Kotlin. Permite aprovechar características propias del lenguaje, como lambdas, funciones de orden superior y otras herramientas funcionales que ayudan a escribir código más claro y directo.

Compose forma parte del conjunto de herramientas conocido como Android Jetpack, y fue diseñado para ser compatible con otros elementos del ecosistema, como ViewModel, LiveData, Navigation, Room, entre otros. Esto hace posible usar Compose en aplicaciones nuevas, pero también ir migrando partes de una app existente poco a poco, sin necesidad de rehacer todo el proyecto desde cero.

Ventajas

- Menos código

Compose permite definir toda la interfaz directamente en Kotlin, sin necesidad de archivos XML. Esto significa que ya no se usan métodos como `findViewById()`, y el resultado es un código más corto, más limpio y más fácil de mantener

- Enfoque declarativo

En lugar de escribir instrucciones para modificar vistas, en Compose se describe cómo debería verse la pantalla según el estado actual. Cuando los datos cambian, la interfaz se actualiza automáticamente. Esto evita tener que escribir lógica para actualizar manualmente cada componente.

- Vista previa en tiempo real

Android Studio permite ver cómo se va viendo la interfaz directamente desde el editor, sin necesidad de compilar ni ejecutar la aplicación. Esto acelera mucho el diseño y permite detectar errores visuales al instante.

- Mayor flexibilidad

Al no depender de clases de vista fijas, Compose facilita la creación de componentes personalizados. Se pueden construir elementos reutilizables sin tener que extender clases o seguir estructuras rígidas, lo cual da más libertad para diseñar.

- Mejor manejo de temas y estilos

Con Compose es más sencillo aplicar temas, colores y tipografías de forma coherente en toda la aplicación. Si se quiere hacer un cambio de estilo, basta con modificar el tema central, y eso se refleja en todos los componentes.

- Integración con herramientas modernas

Compose se lleva bien con otras bibliotecas como `ViewModel`, `LiveData`, `Navigation` y `Room`. Esto permite seguir buenas prácticas de arquitectura sin complicarse ni tener que buscar soluciones externas.

- Mayor velocidad de desarrollo

Gracias a que todo está en un solo archivo, la vista previa funciona en tiempo real y el código es más fácil de entender, se puede desarrollar más rápido que con el modelo antiguo.

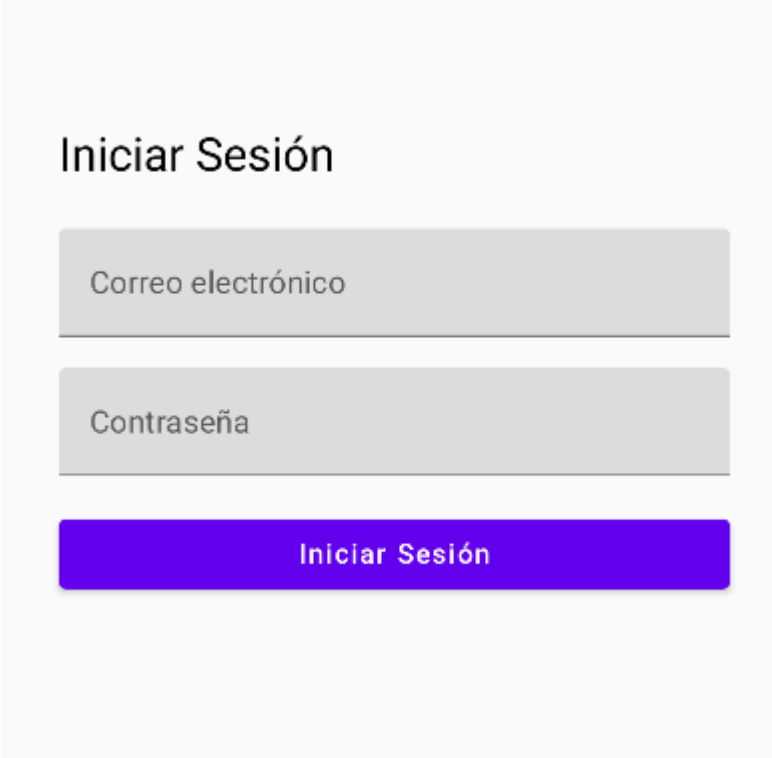
- Respaldo oficial y comunidad activa

Compose es una herramienta oficial de Google, lo que garantiza soporte a largo plazo. Además, la comunidad de desarrolladores que lo usan sigue creciendo, así que es más fácil encontrar documentación, ejemplos y soluciones en línea.

Descripción del Flujo Implementado y diseño de la aplicación

Inicio de sesión (pantalla principal)

La pantalla principal presenta un formulario de inicio de sesión con campos para correo electrónico y contraseña. El formulario valida los datos ingresados y muestra mensajes de error si los campos están vacíos o si el formato del correo no es válido.



The image shows a mockup of a login screen. It features a light gray background. At the top, the text "Iniciar Sesión" is displayed in a bold, black font. Below this, there are two input fields: the first is labeled "Correo electrónico" and the second is labeled "Contraseña". Both fields are light gray with a thin border. At the bottom, there is a prominent blue button with the text "Iniciar Sesión" in white.

Validación de errores

Cuando el usuario presiona el botón “Iniciar sesión” sin completar los campos o con un correo inválido, se muestra un mensaje de error. Esta validación se realiza localmente con Compose.

<div>Correo electrónico chris@gmail</div> <div>Contraseña</div> <div>Iniciar Sesión</div> <div>El correo electrónico no es válido.</div>	<div>Correo electrónico</div> <div>Contraseña</div> <div>Iniciar Sesión</div> <div>Por favor completa todos los campos.</div>
--	---

Pantalla de bienvenida

Una vez que los datos ingresados son correctos, la navegación se realiza a una nueva pantalla que le da la bienvenida al usuario usando su nombre de usuario (extraído del correo electrónico). Esta pantalla confirma que el inicio de sesión fue exitoso

¡Bienvenido, chris!

Has iniciado sesión correctamente.

Conclusiones

El uso de Jetpack Compose en este proyecto de login y navegación ha demostrado ser una excelente elección para desarrollar interfaces modernas y reactivas en Android. Durante la implementación, se pudo comprobar que la herramienta facilita la creación de componentes de interfaz de usuario de manera mucho más eficiente y menos propensa a errores en comparación con los métodos tradicionales basados en XML. Una de las decisiones clave fue optar por un enfoque declarativo en lugar de la gestión imperativa de vistas. Esto permitió que la estructura de la interfaz fuera más intuitiva, facilitando tanto la lectura del código como su mantenimiento. La capacidad de trabajar con Composables y los componentes reutilizables contribuyó a que el desarrollo fuera más modular y organizado.

La implementación de un flujo de navegación entre pantallas fue fundamental. La elección de usar Navigation Compose para gestionar la navegación entre la pantalla de login y la de bienvenida, y pasar parámetros como el nombre de usuario entre ellas, resultó ser sencilla y directa, destacando la integración fluida que ofrece Jetpack Compose con la arquitectura de navegación. Si bien Jetpack Compose presentó muchas ventajas, también hubo algunos desafíos. La principal dificultad fue la curva de aprendizaje para dominar el patrón de programación declarativa y comprender cómo estructurar la UI de manera eficiente. Además, la integración con bibliotecas de terceros o componentes más antiguos no siempre fue tan sencilla, lo que puede generar ciertos obstáculos al usar Jetpack Compose en proyectos más grandes o complejos. A medida que el uso de Jetpack Compose continúa creciendo, es probable que la comunidad y la documentación mejoren aún más.

A pesar de algunos desafíos iniciales, especialmente relacionados con la curva de aprendizaje, los beneficios que ofrece en términos de productividad, mantenibilidad y flexibilidad son innegables. Este proyecto refuerza la recomendación de adoptar Jetpack Compose en futuros desarrollos Android, ya que permite crear aplicaciones más rápidas de desarrollar y más fáciles de mantener, sin comprometer la experiencia del usuario.

GitHub

Enlace a repositorio:

<https://github.com/ChiriH/JetpackLogin-Foro1>

Login - Jetpack Compose

Desarrollo de Software para Móviles DSM941 G01T

Integrantes

- William Alberto García Gómez - GG212522
- Christian Yahir López Hernández - LH212531
- Yahir Stewart Sibrian Arriola - SA212551
- Alberto Joseph Mendoza Moreno - MM200462
- Juan Jose Lue Valdez -LV231966