Universidad Politécnica de Chiapas

Ingeniería en Tecnologías de la Información e Innovación Digital

[Programación para Móviles]

[C1–A3-Login compose]

[Nomenclatura del nombre de archivo: C1–A3-Login compose-223216-DanielPeregrinoPerez.pdf]

[Alumno – Peregrino Pérez Daniel] - [223216]

Docente: [José Alonso Macias Montoya]

Fecha de entrega: [07/05/2025]

1. DESCRIPCIÓN DE LA ACTIVIDAD

1.1. Enunciado del problema

Desarrollar una pantalla de inicio de sesión funcional utilizando Jetpack Compose en Android Studio, implementando validación básica de campos.

1.2. Objetivos de aprendizaje

Definición clara

• Crear una interfaz de inicio de sesión con validación de campos

Objetivos específicos

• Implementar una interfaz de usuario utilizando Jetpack Compose
• Manejar el estado de los campos de entrada

2. FUNDAMENTOS TEÓRICOS

Jetpack Compose representa un enfoque moderno para el desarrollo de interfaces de usuario en Android, utilizando un paradigma declarativo que simplifica la creación de interfaces interactivas. La gestión del estado en Compose se realiza mediante el uso de mutableStateOf y remember, permitiendo actualizaciones reactivas de la interfaz de usuario.

La arquitectura de componibles en Kotlin permite crear interfaces modulares y reutilizables, con un control preciso sobre el diseño y el comportamiento de los elementos de la interfaz.

3. DESARROLLO DE LA ACTIVIDAD

3.1. Desarrollo

- Configuración de un nuevo proyecto en Android Studio
  • Creación de una pantalla de inicio de sesión
  • Implementación de campos de texto para email y contraseña
  • Desarrollo de validación básica de campos
  • Manejo de estado de la interfaz

3.3. Implementación (opcional)

```kotlin

@Composable
fun LoginScreen(
    onLoginClick: (String, String) -> Unit
) {
    //Campos a usarse
    var email by remember { mutableStateOf( value: "") }
    var password by remember { mutableStateOf( value: "") }
    var errorMessage by remember { mutableStateOf( value: "") }

    Column(
        modifier = Modifier
            .fillMaxSize()
            .padding(16.dp),
        verticalArrangement = Arrangement.Center,
        horizontalAlignment = Alignment.CenterHorizontally
    ) {
        Text(
            text = "Inicio de sesión",
            fontFamily = FontFamily.Serif, // Cambia Serif por tu fuente personalizada
            fontSize = 30.sp,              // Tamaño de fuente
            fontWeight = FontWeight.Bold,  // Peso
            style = MaterialTheme.typography.titleLarge,
            modifier = Modifier.padding(8.dp)
        )

        Spacer(modifier = Modifier.height(16.dp))

        // Campo de texto para correo electrónico
        OutlinedTextField(
            value = email,
```

1. Define una función **@Composable** llamada **LoginScreen** que recibe una función **onLoginClick** como parámetro.
2. Utiliza **remember** y **mutableStateOf** para guardar y actualizar el **email**, **password** y un **errorMessage**.
3. Organiza los elementos (Título, campos de texto, botón) verticalmente usando un **Column**.
4. Usa **OutlinedTextField** para los campos de entrada, configurando el tipo de teclado y la transformación visual para la contraseña.
5. El **Button** tiene una lógica **onClick** que:
    - Valida si los campos están vacíos.
    - Actualiza **errorMessage** si hay un error.
    - Llama a **onLoginClick** (pasándole el email y password) si los campos son válidos.

6. Muestra condicionalmente el **errorMessage**.
7. Incluye una función **@Preview** para facilitar el diseño y visualización en Android Studio.

```
package com.example.myapplication                                    Analyzing...


import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import com.example.myapplication.ui.theme.MyApplicationTheme // Este nombre debe coincidir con el nombre

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            MyApplicationTheme {  // Este nombre debe coincidir con el nombre de tu proyecto
                LoginScreen(
                    onLoginClick = { email, password ->
                        // Aquí puedes manejar la autenticación
                        println("Email: $email, Password: $password")
                    }
                )
            }
        }
    }
}
```

1. Es la primera pantalla que se carga.
2. Configura el entorno de Jetpack Compose.
3. Aplica el tema visual de tu aplicación.
4. Muestra el **LoginScreen**.
5. Define qué hacer (**println**) cuando el **LoginScreen** informa que el usuario hizo clic en "Login".

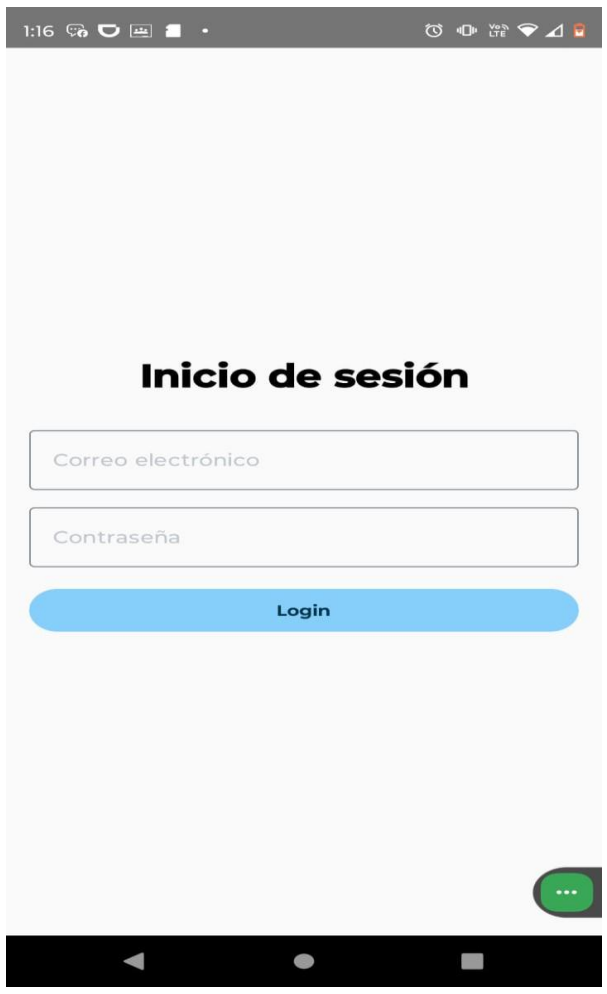
Cómo funcionan juntos:

1. **MainActivity** se inicia y llama a **setContent**.
2. Dentro de **setContent**, **MyApplicationTheme** envuelve a **LoginScreen**.
3. **MainActivity** le pasa una función a **LoginScreen** a través del parámetro **onLoginClick**. Esta función, en este caso, es **{ email, password -> println("Email: $email, Password: $password") }**.
4. El usuario interactúa con **LoginScreen**, ingresando su email y contraseña. Estas entradas actualizan las variables de estado **email** y **password** dentro de **LoginScreen**.

5. Cuando el usuario presiona el botón "Login" en **LoginScreen**:
    - Se ejecuta la validación.
    - Si la validación es exitosa, se llama a **onLoginClick(email, password)**. Esto efectivamente ejecuta la función que **MainActivity** le pasó: **println("Email: $email, Password: $password")**.
    - Los datos (email y password) "viajan" desde **LoginScreen** de vuelta a **MainActivity** a través de esta llamada de función.

4. RESULTADOS

4.1. Resultados obtenidos



5. CONCLUSIONES

Aprendizaje:

• Profundización en el uso de Jetpack Compose

• Comprensión de la gestión de estado en interfaces móviles

Objetivos:

• Objetivos de aprendizaje completados satisfactoriamente

• Implementación exitosa de interfaz de inicio de sesión

Aplicación:

• Base para futuros proyectos de autenticación

• Comprensión de principios de diseño de interfaces móviles

6. DIFICULTADES Y SOLUCIONES

Problema: Manejo de estado y validación de formularios	Causas: Complejidad de la gestión de estado en Compose	Solución: Uso de mutableStateOf y remember para control de estado	Verificación: Validación funcional de campos de entrada

7. REFERENCIAS

Jose Alonso Matias. (2025). *Login compose*. Google Classroom. https://classroom.google.com/u/2/c/Nzc1ODA1NDc3MDEx/m/Nzc5NTM0MDYwMDU5/details

RÚBRICA DE AUTOEVALUACIÓN

Criterio | Nivel alcanzado | Justificación

Comprensión del problema | Intermedio | Implementación completa de interfaz de login

Diseño de la solución | Intermedio | Diseño modular con Jetpack Compose

Implementación | Intermedio | Código funcional con validaciones

Pruebas realizadas | Básico | Pruebas básicas de funcionalidad de interfaz

Documentación | Intermedio | Documentación detallada del proceso