Universidad Politécnica de Chiapas

Ingeniería en Tecnologías de la Información e Innovación Digital

[Programación para Móviles]

[C1–A5-Login vinculado a API]

[Nomenclatura del nombre de archivo: C1–A5-Login vinculado a API-223216-DanielPeregrinoPerez.pdf]

[Alumno – Peregrino Pérez Daniel] - [223216]

Docente: [José Alonso Macias Montoya]

Fecha de entrega: [19/05/2025]

1. DESCRIPCIÓN DE LA ACTIVIDAD

1.1. Enunciado del problema

La actividad consistió en desarrollar una aplicación cliente para Android que interactúa con una API RESTful ya desplegada. Esta API gestiona la autenticación y los datos de los usuarios. Las funcionalidades principales requeridas fueron:

- Registro de nuevos usuarios
- Inicio de sesión de usuarios existentes
- Persistencia del estado de sesión
- Navegación a una pantalla principal tras autenticación exitosa

1.2. Objetivos de aprendizaje

- Comprender e implementar el patrón de arquitectura MVVM (Model-View-ViewModel).
- Usar Kotlin como lenguaje principal de desarrollo.
- Integrar y consumir una API REST utilizando la librería Retrofit.
- Manejar la asincronía en operaciones de red mediante Coroutines.
- Persistir datos como tokens con SharedPreferences.
- Diseñar interfaces responsivas con XML, LiveData y ViewBinding.
- Depurar problemas comunes con Logcat y herramientas de Android Studio.

2. FUNDAMENTOS TEÓRICOS

Durante el desarrollo de esta actividad se utilizaron los siguientes conceptos y tecnologías:

- **Kotlin:** Lenguaje moderno, conciso y seguro, oficial para Android.
- **MVVM:** Patrón que separa UI (View), lógica (ViewModel) y datos (Model), facilitando mantenimiento y pruebas.
- **Android SDK:** Conjunto de herramientas para desarrollar aplicaciones Android.
- **Retrofit:** Cliente HTTP que permite consumir APIs REST de manera sencilla.
- **Gson:** Librería para serializar y deserializar objetos Kotlin/Java a JSON.
- **Coroutines:** Manejo de asincronía sin bloquear el hilo principal.
- **LiveData y ViewModel:** Componentes Jetpack para UI reactiva.
- **ViewBinding:** Facilita la interacción segura con vistas XML.
- **SharedPreferences:** Almacena datos clave-valor de forma persistente.
- **JWT:** Tokens de autenticación que permiten validar la identidad del usuario.
- **Material Components:** Biblioteca de UI moderna con base en Material Design.

3. DESARROLLO DE LA ACTIVIDAD

3.1. Desarrollo

- Creación de un nuevo proyecto en Android Studio.
- Adición de dependencias: Retrofit, Gson, ViewModel, LiveData, Coroutines, Material Components.
- Activación de ViewBinding.
- Configuración del permiso de Internet en AndroidManifest.xml.atributos de tema.

3.3. Implementación (opcional)

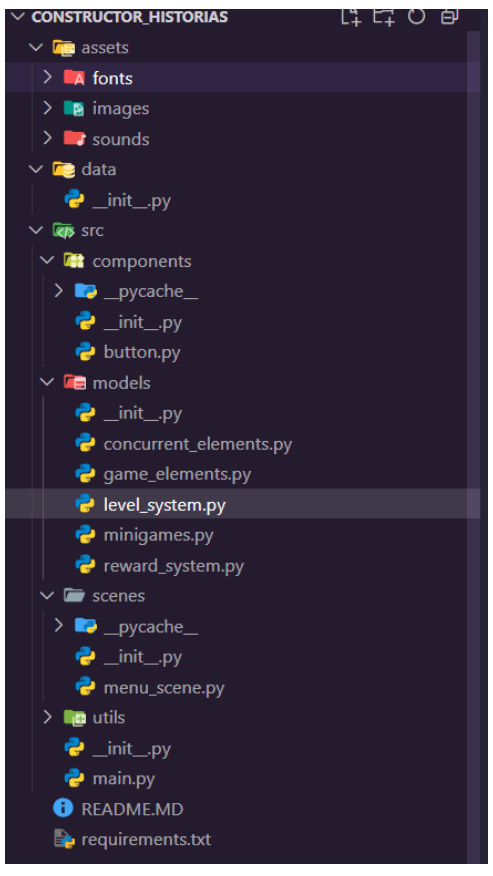
```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <!-- Permiso para acceso a internet -->
    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Login_Validado"
        tools:targetApi="31">

        <!-- Actividad de Login -->
        <activity
            android:name=".ui.login.LoginActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <!-- Actividad principal post-login -->
        <activity
            android:name=".ui.main.MainActivity"
            android:exported="false" />
    </application>
```



Capa de Datos (Data Layer):

- Modelado de clases para peticiones/respuestas (LoginRequest, RegisterRequest, User, etc.).
- Definición de ApiService con los endpoints (/auth/login, /auth/register).
- Configuración del cliente Retrofit.
- Implementación de AuthRepository como interfaz entre ViewModel y la API.

Capa de Vista (View Layer):

- Diseño de LoginActivity y MainActivity usando ConstraintLayout y Material Components.
- Implementación de navegación con Intents.
- Uso de ViewBinding para acceder a elementos visuales.

Capa ViewModel:

- Creación de LoginViewModel para manejar login y registro.
- Uso de LiveData<Resource<T>> para comunicar resultados a la UI.

- Manejo de errores HTTP mediante parseo del errorBody.

Gestión de Sesión:

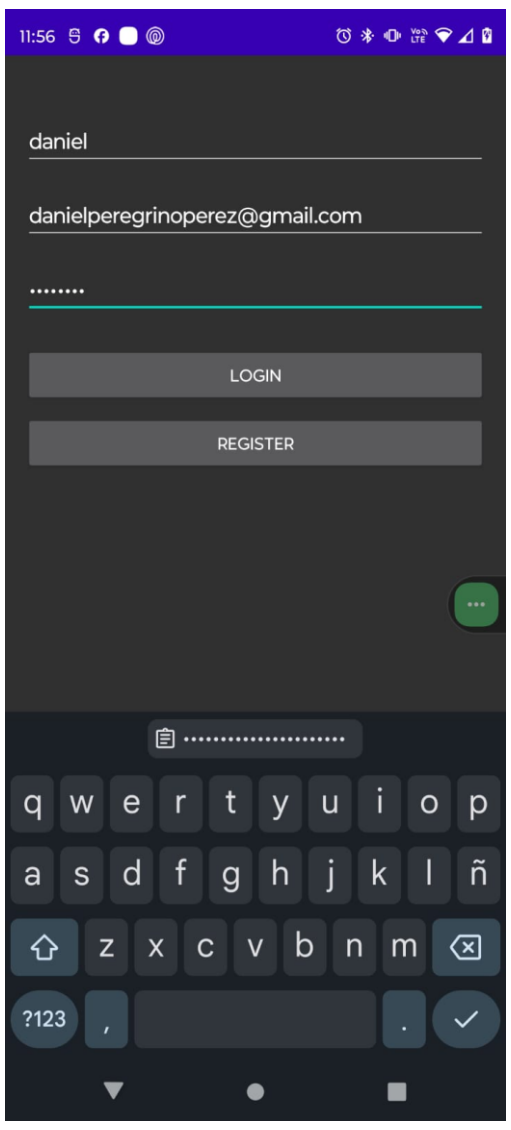
- Clase SessionManager que guarda el token JWT y datos del usuario.
- Verificación del token al iniciar la app para mantener la sesión activa

4. RESULTADOS

4.1. Resultados obtenidos

- Registro de nuevos usuarios.
- Inicio de sesión con validación de credenciales.
- Persistencia de sesión con SharedPreferences.
- Navegación hacia la pantalla principal tras login exitoso.
- Logout funcional que elimina datos de sesión.
- Manejo adecuado de errores (credenciales incorrectas, usuario existente, etc.).
- url del login:

  BASE_URL = "https://login-back-android.onrender.com/api/"



4.2. Análisis de resultados

La app funciona como se esperaba:

- La API responde correctamente a login y registro.
- El token JWT se guarda y reutiliza para mantener la sesión.
- Se observaron mensajes de error adecuados ante fallos.
- La estructura MVVM permitió una separación clara del código.

5. CONCLUSIONES

- **Se comprendió e implementó de forma efectiva el patrón MVVM.**
- **Se aplicaron tecnologías modernas y prácticas de desarrollo profesional.**
- **El uso de Retrofit y Gson simplificó la comunicación con la API.**
- **La arquitectura limpia permitió una buena organización del proyecto.**
- **Se alcanzaron los objetivos planteados en la actividad.**

6. DIFICULTADES Y SOLUCIONES

Problema	causa	solucion
Error de tema al iniciar (IllegalStateException)	El tema no heredaba de Theme.AppCompat	Se modificó a Theme.MaterialComponents.DayNight.NoActionBar
Error con ConstraintLayout	Faltaba la dependencia	Se agregó implementation "androidx.constraintlayout:constraintlayout:X.X.X"

7. REFERENCIAS

- Android Developers: https://developer.android.com
- Guía de Arquitectura Jetpack: https://developer.android.com/jetpack/guide
- Kotlin Docs: https://kotlinlang.org/docs/home.html
- Retrofit Docs: https://square.github.io/retrofit/

- Coroutines Docs: https://kotlinlang.org/docs/coroutines-overview.html
- Material Components: https://material.io/components
- Render API Hosting: https://render.com