# Proyecto de Recetas

## Nuevo proyecto

### En GitHub: https://github.com/antoniuskas/receitas.git

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

# Configuramos la aplicación

## Añadimos las dependencias necesarias

CONTROL + ALT + SHIFT + S

File – Proyect Structure

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

## Aplicamos los cambios

Texto

El contenido generado por IA puede ser incorrecto.

## Habilitamos el viewBinding,

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

No olvidar: Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

## Creación del modelo de datos

Vamos a crear una enumeración para determina las categorías de recetas

Primero creamos el paquete model

Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

## Ahora voy a crear los Strings para que sea multiidioma

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

## Creo para los principales idiomas

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Y para china 😊

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

## Ahora para crear la enumeración

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

## Asociamos cada elemento de la enumeración con un recurso definido en strings.xml

Texto

El contenido generado por IA puede ser incorrecto.

## También creamos una enumeración para la Dificultad

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Creamos el data class para Receta.kt, como es una clase para almacenar datos la haremos como data classs y Kotlin nos proporciona automáticamente los métodos equals, hasCode, toString y copy. Y podemos añadir otros métodos para trabajar con los datos de las recetas.

Texto

El contenido generado por IA puede ser incorrecto.

Creamos la clase converters, Es **imprescindible** crear primero la clase Converters, porque Room necesita saber cómo guardar tus enum (Categoria y Dificultad) y el campo photo: ByteArray? en la base de datos.

Texto

El contenido generado por IA puede ser incorrecto.

Vamos a crear la clase RecetaDAO para decir como accedemos a los datos

Contiene funciones que sirve entre la aplicación y la base de datos, define como insertamos, borramos datos de la tabla recetas en Room.

Texto

El contenido generado por IA puede ser incorrecto.

## Ahora creamos la base de datos local utilizando Room.

RecetaDatabase es como el **"motor de base de datos Room"**, y la app lo utiliza para obtener el DAO (recetaDao()) que contiene las funciones para acceder a las recetas almacenadas.

Texto

El contenido generado por IA puede ser incorrecto.

## Crear RecetaViewModel

Actúa como puente entre la interfaz de usuario (Fragment) y el repositorio de datos (RecetaRepository)

Contiene un StateFlow que mantiene actualizada la lista de recetas en tiempo real

Observa los cambios de la base de datos a través del Repositorioo

Llama a las funciones del Repositorio para insertar, borrar, actualizar o recuperar las recetas.

Pero lo hace de un modo transparente.

Los fragment miran el StaeFlow del ViewModel y actualizan la interfaz automáticamente.

Texto

El contenido generado por IA puede ser incorrecto.

Si el usuario gira la pantalla o cambia entre fragmentos el ViewModel mantiene los datos.

## Crear el RecetaRepository

El repositorio actúa como intermediario entre el ViewModel y la base de datos.

Texto

El contenido generado por IA puede ser incorrecto.

Primero creamos el item\_receta.xml que es un layout de una receta individual para mostrar en el RecyclerView.

Texto

El contenido generado por IA puede ser incorrecto.

## Creamos el layout en res/layout/item\_receta.xml

Como tenemos el binding activado Android Studio generará automáticamente la clase ItemRecetaBinding ya que tenemos habilitado ViewBinding en el build.gradle. (lo pusimos al principio a true).

Utilizamos recursos para que los nombres aparezcan en el idioma correspondiente.

Texto

El contenido generado por IA puede ser incorrecto.

## Ahora crearemos el RecetaAdapter

* Va a mostar una lista de recetas en un RecyclerView
* Va a detectar cuando se pulsa una receta

Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

Ahora debemos mostrar todas las recetas en una lista y hacer algo cuando pulsamoe en ellas.

## Crearmos ListaRecetasFragment

**¿Qué debe hacer ListaRecetasFragment?**

1. **Inflar el layout** con FragmentListaRecetasBinding.
2. **Configurar el RecyclerView** y asociar el RecetaAdapter.
3. **Observar el StateFlow del ViewModel** con la lista de recetas.
4. **Reaccionar a los clics** (por ejemplo, navegar al detalle).

Primero deberemos hacer un layout llamado fragment\_lista\_recetas.xml que tenga al menos un RecyclerView

Solo necesitamos uno de momento ya que no vamos a mostrar varias listas independientes en la misma pantalla, como por ejemplo como aparece en Netflix o amazon, como recomendadas para ti, populares, etc. Así tendríamos por ejemplo varios RecyclerView dentro de un ScrollView

También si por ejemplo quisiera una lista de categorías y luego abajo filtradas las recetas tendría

<constraintLayout>

RecyclerView Horizontal 🡪 para las categorías

RecyclerView Vertical 🡪 para las recetas filtradas de cada categoría

<ConstraintLayout>

## GIT

Antes de nada voy a subir el proyecto a git.

En la terminal miro si tengo git

Texto

El contenido generado por IA puede ser incorrecto.

Como hice un git init, lo elimino



Reviso la configuración de Android Studio

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Hacemos clic en Test para comprobar la versión y que AE lo detecta.

Creo el repositorio desde el Version Control que está arriba a la izquierda

Una captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Yo creo que esto es lo mismo que poner Git init en la terminal

Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.

Veo que estoy en la rama Master

Imagen de la pantalla de un celular con letras

El contenido generado por IA puede ser incorrecto.

Si hago un ls -Force en la terminal veo el .git. -Force para ver ocultos (force en minúsculas también funciona)

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Voy a crear un .gitignore para ignorar ficheros y carpetas en mi repositorio.

Los build contiene ficheros que se regeneran y ocupan mucho espacio. No tiene sentido versionarlos pues cambian cada vez que compilamos.

Los .idea , .iml son archivos de configuración de Android Studio y pueden cambiar entre equipos, no afectan al código fuente ni a la funcionalidad

Los de .gradle son temporales de configuración, y Gradle los puede generar automáticamente por lo que no es necesario almacenarlos.

Local.propierties contiene la ruta local al SDK de Android y como es distinta en cada equipo no debemos compartirla.

Los \*.jks o cualquier fichero de claves es privado y no debemos subirlo a ningún repositorio público pues contiene la clave de la firma de tu aplicación.

Puede que Android Studio lo cree al hacer el git

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

El contenido generado por IA puede ser incorrecto.

Lo abro y le pongo el resto.

## Hacemos el primer commit

**Selecciono los ficheros**

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Vamos a crear un repositorio en mi cuenta en GitHub

Texto

El contenido generado por IA puede ser incorrecto.

Podría agregar un .gitignore para Android, pero como ya lo tengo hecho, lo omito.

Hago otro commit para añadir el fichero de explicación.

## Voy a hacer el push

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Me aparece todo

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Le doy Push Anyway, hay unos problemillas, pero lo subo igual

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

Al hacer el Push se sube todo menos lo indicado

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

El contenido generado por IA puede ser incorrecto.

Creo una nueva rama y añado un readme

Imagen que contiene pájaro, ave, flor, árbol

El contenido generado por IA puede ser incorrecto.

Ahora continuo con el proyecto

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

Creo el fragment \_lista\_recetas.xml que

// El layout principal del fragmento de la lista de recetas  
// Dentro del Constraint ponemos el Recyclerview que rellena toda la pantalla

Texto

El contenido generado por IA puede ser incorrecto.

Ahora vamos a preparar para que el framento pueda mostrar la lista con un solo RecyclerView y enlazar cada campo o item con el fragmento de detalle.

Antes de nada quiero crear una nueva rama que tenga lo de master y que se llame fragment, primero un pull para asegurarse de tenerla actualizada.

Texto

El contenido generado por IA puede ser incorrecto.

Captura de pantalla de un teléfono celular

El contenido generado por IA puede ser incorrecto.

Hago el commit y el push

Creo la nueva rama

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

## Ahora creamos ListaRecetaFragment.kt

Lo que hace es mostrar la lista de re cetas almacenadas en la base de datos y gestiona las acciones del usuario.

package com.marino.receitas.ui  
  
import android.os.Bundle  
import android.view.LayoutInflater  
import android.view.View  
import android.view.ViewGroup  
import androidx.fragment.app.Fragment  
import androidx.fragment.app.viewModels  
import androidx.lifecycle.*lifecycleScope*import androidx.recyclerview.widget.LinearLayoutManager  
import com.marino.receitas.RecetaRepository  
import com.marino.receitas.RecetaViewModel  
import com.marino.receitas.adapter.RecetaAdapter  
import com.marino.receitas.data.RecetaDatabase  
import com.marino.receitas.databinding.FragmentListaRecetasBinding  
import com.marino.receitas.viewmodel.RecetaViewModelFactory  
import kotlinx.coroutines.flow.collectLatest  
import kotlinx.coroutines.launch  
  
*/\*\*  
 \* Fragmento que muestra la lista de recetas.  
 \* Requiere:  
 \* - Layout `fragment\_lista\_recetas.xml` con un RecyclerView de id `recyclerRecetas`.  
 \* - ViewBinding habilitado en build.gradle (`viewBinding = true`).  
 \*/*class ListaRecetasFragment : Fragment() {  
  
 // ViewBinding  
 private var \_binding: FragmentListaRecetasBinding? = null  
 private val binding get() = \_binding!!  
  
 // ViewModel usando repository y factory  
 private val viewModel: RecetaViewModel by *viewModels* **{** RecetaViewModelFactory(  
 RecetaRepository(  
 RecetaDatabase.getDatabase(requireContext()).recetaDao()  
 )  
 )  
 **}** // Adapter para mostrar las recetas  
 private lateinit var recetaAdapter: RecetaAdapter  
  
 override fun onCreateView(  
 inflater: LayoutInflater,  
 container: ViewGroup?,  
 savedInstanceState: Bundle?  
 ): View {  
 \_binding = FragmentListaRecetasBinding.inflate(inflater, container, false)  
 return binding.*root* }  
  
 override fun onViewCreated(view: View, savedInstanceState: Bundle?) {  
 super.onViewCreated(view, savedInstanceState)  
  
 // Inicializa el adapter con lista vacía  
 recetaAdapter = RecetaAdapter(*emptyList*())  
  
 // Configura el RecyclerView  
 binding.recyclerRecetas.*apply* **{** *layoutManager* = LinearLayoutManager(requireContext())  
 *adapter* = recetaAdapter  
 **}** // Observa el flow de recetas y actualiza el adapter  
 *viewLifecycleOwner*.*lifecycleScope*.*launch* **{** viewModel.todasLasRecetas.collectLatest **{** lista **->** recetaAdapter.actualizarLista(lista)  
 **}  
 }** }  
  
 override fun onDestroyView() {  
 super.onDestroyView()  
 \_binding = null  
 }  
}

en activity\_main.xml vamos a poner un contenedor para fragmentos

Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

Al final las dependencias han quedado así, y eso que he optado por lo más simple y no poner el Navigation que me daba muchos errores.

plugins **{** alias(*libs*.*plugins*.*android*.*application*)  
 alias(*libs*.*plugins*.*kotlin*.*android*)  
 *kotlin*("kapt") // Plugin KAPT para Room  
 //id("androidx.navigation.safeargs.kotlin") // Safe Args plugin  
**}***android* **{** namespace = "com.marino.receitas"  
 compileSdk = 35  
  
 defaultConfig **{** applicationId = "com.marino.receitas"  
 minSdk = 24  
 targetSdk = 35  
 versionCode = 1  
 versionName = "1.0"  
  
 testInstrumentationRunner = "androidx.test.runner.AndroidJUnitRunner"  
 **}** buildTypes **{** *release* **{** isMinifyEnabled = false  
 proguardFiles(  
 getDefaultProguardFile("proguard-android-optimize.txt"),  
 "proguard-rules.pro"  
 )  
 **}  
 }** compileOptions **{** sourceCompatibility = JavaVersion.*VERSION\_11* targetCompatibility = JavaVersion.*VERSION\_11* **}** *kotlinOptions* **{** jvmTarget = "11"  
 **}** buildFeatures **{** viewBinding = true  
 **}  
}***dependencies* **{** val roomVersion = "2.7.1"  
 // Core Android  
 *implementation*(*libs*.*androidx*.*core*.*ktx*)  
 *implementation*(*libs*.*androidx*.*appcompat*)  
 *implementation*(*libs*.*material*)  
 *implementation*(*libs*.*androidx*.*activity*)  
 *implementation*(*libs*.*androidx*.*constraintlayout*)  
 *implementation*(*libs*.*androidx*.*fragment*.*ktx*)  
  
 // Lifecycle  
 *implementation*(*libs*.*androidx*.*lifecycle*.*livedata*.*ktx*)  
 *implementation*(*libs*.*androidx*.*lifecycle*.*viewmodel*)  
  
 // Coroutines  
 *implementation*(*libs*.*kotlinx*.*coroutines*.*android*)  
  
 // Navigation Component  
 *implementation*(*libs*.*androidx*.*navigation*.*fragment*.*ktx*)  
 //implementation(libs.androidx.navigation.ui.ktx)  
  
 // Gson for JSON  
 *implementation*(*libs*.*gson*)  
 *implementation*(*libs*.*support*.*annotations*)  
 *implementation*(*libs*.*androidx*.*annotation*)  
  
 // Testing  
 *testImplementation*(*libs*.*junit*)  
 *androidTestImplementation*(*libs*.*androidx*.*junit*)  
 *androidTestImplementation*(*libs*.*androidx*.*espresso*.*core*)  
  
 // Room dependencies  
 //val room\_version = "2.7.1"  
  
 *implementation*("androidx.room:room-runtime:$roomVersion")  
  
 // If this project uses any Kotlin source, use Kotlin Symbol Processing (KSP)  
 // See Add the KSP plugin to your project  
 //ksp("androidx.room:room-compiler:$room\_version")  
  
 // If this project only uses Java source, use the Java annotationProcessor  
 // No additional plugins are necessary  
 *kapt*("androidx.room:room-compiler:$roomVersion") // Usar KAPT en lugar de annotationProcessor  
 //*annotationProcessor*("androidx.room:room-compiler:$roomVersion")  
  
 // Opcionales  
 *implementation*("androidx.room:room-rxjava2:$roomVersion")  
 *implementation*("androidx.room:room-rxjava3:$roomVersion")  
 *implementation*("androidx.room:room-guava:$roomVersion")  
 *testImplementation*("androidx.room:room-testing:$roomVersion")  
 *implementation*("androidx.room:room-paging:$roomVersion")  
**}**

Al ejecutar vemos la lista de recetas.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Es lo que me ha dado tiempo a hacer. Las dependencias me han llevado muchas horas para poder funcionar, he puesto las de los apuntes y a mi no me han funcionado correctamente. Al final se ha ejecutado así. Tenía pensado crear otra tabla para los pasos de la receta, poner la imagen, pero no.

Luego la IA se equivoca bastante y la monta.

Un saludo