Prueba - Superhéroes

- Para realizar esta prueba debes haber estudiado previamente todo el material disponibilizado correspondiente al módulo.
- Una vez terminada la prueba, comprime la carpeta y sube el `.zip`

Descripción

Imagínate tener toda la información relacionada con Superhéroes en una sola app, que además es rápida y funciona a la perfección en lugares con poca cobertura de Internet.

Para eso vamos a armar la arquitectura en una versión inicial que funcionará como MVP

Para el MVP se requiere mostrar una lista de los nombres de los superhéroes con un scroll funcional que son leídos desde la base de datos (DB) y actualizados usando la REST API de superhéroes de akabab

Las herramientas para construir esta versión serán:

● ROOM como gestor de datos

● Retrofit para consumir una REST API

● LiveData para propagar los cambios

● Recyclerview para mostrar los superhéroes

Además para solucionar algunos problemas comunes se puede utilizar los siguientes patrones

● MVVM

● Repository

● Dao

● Factory

Instrucciones

1. Crear un proyecto para Kotlin y definir el nombre de la app
2. Agregar el permiso de Internet al manifiesto

Hito 1

En esta fase inicial se debe almacenar y gestionar los datos. Para eso se implementa la base de datos usando ROOM y las entidad para el personaje. La gestión de los datos se hace usando Dao para acceder a la DB y un repositorio que se encarga de consumir la información desde una API y usando Dao guardar la lista de personajes.

ROOM (4 puntos)

3. Agregar ROOM como dependencia
4. Definir el objeto de dominio SuperheroCharacter que va a servir como entidad de la DB y para recuperar los datos de la API (**1 Punto**)
5. Definir la interfaz Dao para obtener y guardar los personajes (**1 Punto**)
6. Definir la base de datos heredando de RoomDatabase (**1 Punto**)
7. Usar el patrón Singleton para mantener la instancia de la DB que será utilizada (**1 Punto**)

Repository (6 puntos)

8. Agregar Retrofit como dependencia. Para convertir JSON se puede utilizar Moshi agregando la dependencia

```
implementation "com.squareup.retrofit2:converter-moshi:2.4.0"
```

9. Crear la interfaz para consumir la REST API que tiene los superhéroes. El endpoint que se va a utilizar es /all.json para obtener la lista completa de superhéroes (**1 Puntos**)
10. Crear la implementación para utilizar Retrofit (**3 Puntos**)
11. Crear el repositorio que actualiza su lista de superhéroes consumiendo desde la API usando Retrofit y guardando en ROOM usando la interfaz Dao (**2 Puntos**)

Hito 2

Con la información disponible vamos a enlazar la vista usando MVVM para desplegar una lista usando RecyclerView. El viewModel utiliza la información expuesta por el repositorio usando LiveData para poder notificar al adaptador con la información nueva

MVVM (4 puntos)

1. Crear el ViewModel extendiendo desde AndroidViewModel (**1 Punto**)
2. El ViewModel tiene una instancia del repositorio (**1 Punto**)
3. Al inicializarse el ViewModel debe llamar al repositorio para obtener la lista de superhéroes (**1 Punto**)
4. El ViewModel guarda la lista de superhéroes que recupera desde el repositorio para ser consumida por la vista (**1 punto**)

View (6 puntos)

5. Adaptar el layout principal agregando el recyclerView (**0.25 punto**)
6. Crear el layout para el ítem de la lista (**1.75 puntos**)
7. Crear el adaptador a utilizar en el recyclerview (Adapter + ViewHolder) (**1 punto**)
8. En la actividad o fragmento que contenga el recyclerview debe asignar el adaptador al recyclerView (**1 punto**)
9. Usando el ViewModel, observar los cambios en la lista de superhéroes y actualizar el adaptador con la nueva información (**2 puntos**)

Mejoras

● Agregar la imagen a cada item. Para esto se debe modificar la clase SuperheroCharacter para que pueda leer de la API y almacenar en la DB las url de las imágenes (1 té o café)

● Agregar un fragmento con el detalle para cada personaje al hacer click sobre el item en la lista (1 té o café).