LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 4

CONNECT TO THE INTERNET

Oleh:

Hari Octavian Delrossi NIM. 2210817210033

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
JUNI 2024

LEMBAR PENGESAHAN
LAPORAN PRAKTIKUM PEMROGRAMAN MOBILE
MODUL 4

Laporan Praktikum Pemrograman Mobile Modul 4: Connect to The Internet ini disusun sebagai syarat lulus mata kuliah Praktikum Pemrograman Mobile. Laporan Praktikum ini dikerjakan oleh:

Nama Praktikan : Hari Octavian Delrossi
NIM : 2210817210033

Menyetujui, Asisten Praktikum	Mengetahui, Dosen Penanggung Jawab Praktikum
Muhammad Asandy Putra NIM. 2110817310002	Muti'a Maulida, S.Kom., M.T.I. NIP. 198810272019032013

DAFTAR ISI

DAFTAR GAMBAR

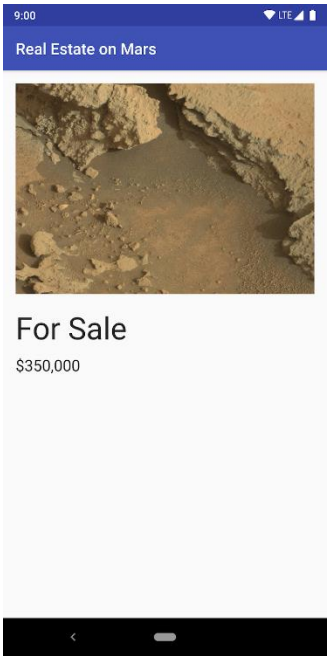
DAFTAR TABEL

SOAL PRAKTIKUM

Buatlah sebuah aplikasi Android sederhana untuk menampilkan data dari Internet melalui Public API

1. Daftar Public API yang dapat digunakan dapat dilihat pada link berikut: https://github.com/public-apis/public-apis (dapat juga mengambil diluar dari link tersebut)
2. Pada saat dijalankan, aplikasi akan terhubung dengan Internet untuk menarik data dari **Public API** tersebut
3. Gunakan library tambahan yaitu **Retrofit** untuk mempermudah proses koneksi internet
4. Gunakan library tambahan yaitu **Moshi** untuk mempermudah proses data JSON
5. Gunakan library tambahan yaitu **Glide** untuk memuat dan menyimpan gambar dalam cache berdasarkan URL (opsional)
6. Data tersebut kemudian ditampilkan dalam bentuk **RecyclerView**
7. Masing-masing data di RecyclerView tersebut dapat diklik untuk menampilkan detailnya
8. Gunakan **LiveData** dan **ViewModel** untuk mempertahankan state dari aplikasi pada saat Configuration Changes
9. Saat pengguna merotasi tampilan handphone dari Portrait menjadi Landscape maka tampilan data yang sudah ada tidak boleh hilang

Contoh aplikasi:



A. Source Code

Tabel 1. Source Code Hasil Jawaban MainActivity.kt

```
1   package com.example.cartoons
2
3   import android.content.Intent
4   import android.os.Bundle
5   import android.util.Log
6   import androidx.appcompat.app.AppCompatActivity
7   import androidx.appcompat.widget.Toolbar
8   import androidx.lifecycle.lifecycleScope
9   import androidx.recyclerview.widget.LinearLayoutManager
    import androidx.recyclerview.widget.RecyclerView
10  import kotlinx.coroutines.launch
11
12  class   MainActivity : AppCompatActivity() {
13
14      private lateinit var recyclerView: RecyclerView
15      private lateinit var cartoonAdapter: CartoonAdapter
16      private lateinit var cartoons: List<Cartoon>

17      override fun onCreate(savedInstanceState: Bundle?) {
18          super.onCreate(savedInstanceState)
19          setContentView(R.layout.activity_main)
```

```
        val toolbar: Toolbar = findViewById(R.id.toolbar)
        setSupportActionBar(toolbar)

        recyclerView = findViewById(R.id.recyclerView)
        recyclerView.layoutManager =
LinearLayoutManager(this)

        lifecycleScope.launch {
            try {
                cartoons =
RetrofitInstance.api.getCartoons()
                cartoons.forEach { cartoon ->
                    Log.d("MainActivity", "Cartoon:
$cartoon")
                }
                cartoonAdapter = CartoonAdapter(cartoons)
{ cartoon ->
                    val intent =
Intent(this@MainActivity, DetailActivity::class.java)
                    intent.putExtra("cartoon", cartoon)
                    startActivity(intent)
                }
                recyclerView.adapter = cartoonAdapter
            } catch (e: Exception) {
                e.printStackTrace()
                Log.e("MainActivity", "Error fetching
cartoons", e)
            }
        }
    }
}
```

Tabel 2. Source Code Hasil Jawaban DetailActivity.kt

```
1  package com.example.cartoons
2
3  import android.graphics.PorterDuff
4  import android.os.Bundle
5  import android.widget.ImageView
6  import android.widget.TextView
7  import androidx.appcompat.app.AppCompatActivity
8  import androidx.appcompat.widget.Toolbar
9  import com.bumptech.glide.Glide
10
11 class DetailActivity : AppCompatActivity() {
12
13     private lateinit var cartoonImage: ImageView
14     private lateinit var cartoonTitle: TextView
15     private lateinit var cartoonCreator: TextView
16     private lateinit var cartoonYear: TextView
17     private lateinit var cartoonGenre: TextView
18     private lateinit var cartoonEpisodes: TextView
19     private lateinit var toolbar: Toolbar
20
21     override fun onCreate(savedInstanceState: Bundle?)
   {
22         super.onCreate(savedInstanceState)
23         setContentView(R.layout.activity_detail)
24
25         toolbar = findViewById(R.id.toolbar)
26         setSupportActionBar(toolbar)
27         supportActionBar?.setDisplayHomeAsUpEnabled
   (true)
28         supportActionBar?.title = ""
29
30         toolbar.navigationIcon?.setColorFilter(resources.
   getColor(android.R.color.white),PorterDuff.Mode.SRC_ATOP)
31
32         cartoonImage = findViewById(R.id.cartoonImage)
33         cartoonTitle = findViewById(R.id.cartoonTitle)
34         cartoonCreator                                    =
   findViewById(R.id.cartoonCreator)
35         cartoonYear = findViewById(R.id.cartoonYear)
36         cartoonGenre = findViewById(R.id.cartoonGenre)
```

```
        cartoonEpisodes = findViewById(R.id.cartoonEpisodes)

        val cartoon = intent.getSerializableExtra("cartoon") as Cartoon
        cartoonTitle.text = cartoon.title
        cartoonCreator.text = cartoon.creator.joinToString(", ")
        cartoonYear.text = cartoon.year.toString()
        cartoonGenre.text = cartoon.genre.joinToString(", ") ?: "No genre available"
        cartoonEpisodes.text = "Total Episodes: ${cartoon.episodes}"

        toolbar.title = cartoon.title

        Glide.with(this)
            .load(cartoon.image)
            .placeholder(R.drawable.placeholder_image)
            .error(R.drawable.error_image)
            .into(cartoonImage)
    }

    override fun onSupportNavigateUp(): Boolean {
        onBackPressed()
        return true
    }
}
```

Tabel 3. Source Code Hasil Jawaban Cartoon.kt

```
package com.example.cartoons

import com.squareup.moshi.Json
import java.io.Serializable

data class Cartoon(
    val id: Int,
    @Json(name = "title") val title: String,
    @Json(name = "year") val year: Int,
    @Json(name = "creator") val creator: List<String>,
    @Json(name = "image") val image: String?,
```

```
12      @Json(name = "genre") val genre: List<String>,
13      @Json(name = "episodes") val episodes: Int
14  ) : Serializable
```

Tabel 4. Source Code Hasil Jawaban CartoonAdapter.kt

```
1   package com.example.cartoons
2
3   import android.view.LayoutInflater
4   import android.view.View
5   import android.view.ViewGroup
6   import android.widget.ImageView
7   import android.widget.TextView
8   import androidx.recyclerview.widget.RecyclerView
9   import com.bumptech.glide.Glide
10  import com.bumptech.glide.request.RequestOptions
11
12  class CartoonAdapter(
13      private val cartoons: List<Cartoon>,
14      private val onItemClick: (Cartoon) -> Unit
15  ) : RecyclerView.Adapter<CartoonAdapter.
    CartoonViewHolder>() {
16
17      override fun onCreateViewHolder(parent:
    ViewGroup, viewType: Int): CartoonViewHolder {
18          val view = LayoutInflater.from
    (parent.context).inflate
    (R.layout.item_cartoon, parent, false)
19          return CartoonViewHolder(view)
20      }
21
22      override fun onBindViewHolder(holder:
    CartoonViewHolder, position: Int) {
23          holder.bind(cartoons[position],
    onItemClick)
24      }
25
26      override fun getItemCount(): Int =
    cartoons.size
27
28      class CartoonViewHolder(itemView:
    View) : RecyclerView.ViewHolder(itemView) {
```

```
29          private val cartoonImage:
    ImageView = itemView.findViewById(R.id.cartoonImage)
30          private val cartoonTitle:
    TextView = itemView.findViewById(R.id.cartoonTitle)
31          private val cartoonCreator:
    TextView = itemView.findViewById(R.id.cartoonCreator)
32
33          fun bind(cartoon: Cartoon, onItemClick:
    (Cartoon) -> Unit) {
34              cartoonTitle.text = cartoon.title
35              cartoonCreator.text = cartoon.creator.
    joinToString(", ")
36
37              val requestOptions = RequestOptions()
38                  .placeholder(R.drawable.
    placeholder_image)
39                  .error(R.drawable.error_image)
40
41              Glide.with(itemView.context)
42                  .load(cartoon.image)
43                  .apply(requestOptions)
44                  .into(cartoonImage)
45
46              itemView.setOnClickListener {
    onItemClick(cartoon) }
47          }
48      }
49  }
```

Tabel 5. Source Code Hasil Jawaban CartoonApi.kt

```
1   package com.example.cartoons
2
3   import retrofit2.http.GET
4
5   interface CartoonApi {
6       @GET("cartoons2D")
7       suspend fun getCartoons(): List<Cartoon>
8   }
```

Tabel 6. Source Code Hasil Jawaban CartoonViewModel.kt

```
1   package com.example.cartoons
```

```
2
3  import androidx.lifecycle.LiveData
4  import androidx.lifecycle.MutableLiveData
5  import androidx.lifecycle.ViewModel
6  import androidx.lifecycle.viewModelScope
7  import kotlinx.coroutines.Dispatchers
8  import kotlinx.coroutines.launch
9
10 class CartoonViewModel : ViewModel() {
11
12     private           val           _cartoons           =
   MutableLiveData<List<Cartoon>>()
13     val   cartoons:   LiveData<List<Cartoon>>   get()   =
   _cartoons
14
15     init {
16         fetchCartoons()
17     }
18
19     private fun fetchCartoons() {
20         viewModelScope.launch(Dispatchers.IO) {
21             try {
22                 val             cartoonList             =
   RetrofitInstance.api.getCartoons()
23                 _cartoons.postValue(cartoonList)
24             } catch (e: Exception) {
25                 e.printStackTrace()
26             }
27         }
28     }
29
30     fun filterCartoons(query: String?) {
31         if (query.isNullOrEmpty()) {
32             fetchCartoons()
33         } else {
34             val filteredList = _cartoons.value?.filter {
35                 it.title?.contains(query,   ignoreCase   =
   true) == true ||
36                         it.genre.any    {    genre    ->
   genre.contains(query, ignoreCase = true) }
37             }
```

```
38              _cartoons.postValue(filteredList)
39          }
40      }
41  }
```

Tabel 7. Source Code Hasil Jawaban RetrofitInstance.kt

```
1   package com.example.cartoons
2
3   import retrofit2.Retrofit
4   import retrofit2.converter.moshi.MoshiConverterFactory
5   import com.squareup.moshi.Moshi
6   import
    com.squareup.moshi.kotlin.reflect.KotlinJsonAdapterFactory
7
8   object RetrofitInstance {
9
10      private val moshi = Moshi.Builder()
11          .add(KotlinJsonAdapterFactory())
12          .build()
13
14      private val retrofit by lazy {
15          Retrofit.Builder()
16
    .baseUrl("https://api.sampleapis.com/cartoons/")
17
    .addConverterFactory(MoshiConverterFactory.create(moshi))
18              .build()
19      }
20
21      val api: CartoonApi by lazy {
22          retrofit.create(CartoonApi::class.java)
23      }
24  }
```

Tabel 8. Source Code Hasil Jawaban activity_main.xml

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
3       xmlns:app="http://schemas.android.com/apk/res-auto"
4       android:layout_width="match_parent"
5       android:layout_height="match_parent"
```

```
6       android:orientation="vertical">
7
8       <androidx.appcompat.widget.Toolbar
9           android:id="@+id/toolbar"
10          android:layout_width="match_parent"
11          android:layout_height="?attr/actionBarSize"
12          android:background="?attr/colorPrimary"
13          android:elevation="4dp"
14          app:title="List Cartoons"
15          app:titleTextColor="@android:color/white" />
16
17      <androidx.recyclerview.widget.RecyclerView
18          android:id="@+id/recyclerView"
19          android:layout_width="match_parent"
20          android:layout_height="match_parent"
21          android:padding="16dp" />
22  </LinearLayout>
```

Tabel 9. Source Code Hasil Jawaban activity_detail.xml

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
3       xmlns:app="http://schemas.android.com/apk/res-auto"
4       android:layout_width="match_parent"
5       android:layout_height="match_parent"
6       android:orientation="vertical">
7
8       <androidx.appcompat.widget.Toolbar
9           android:id="@+id/toolbar"
10          android:layout_width="match_parent"
11          android:layout_height="?attr/actionBarSize"
12          android:background="?attr/colorPrimary"
13          android:elevation="4dp"
14          app:titleTextColor="@android:color/white" />
15
16      <ScrollView
17          android:layout_width="match_parent"
18          android:layout_height="match_parent"
19          android:padding="16dp">
20
21          <LinearLayout
```

```
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical">

            <ImageView
                android:id="@+id/cartoonImage"
                android:layout_width="match_parent"
                android:layout_height="600dp"
                android:scaleType="centerCrop" />

            <TextView
                android:id="@+id/cartoonTitle"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:textStyle="bold"
                android:textSize="24sp"
                android:paddingTop="8dp" />

            <TextView
                android:id="@+id/cartoonCreator"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:paddingTop="4dp" />

            <TextView
                android:id="@+id/cartoonYear"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:paddingTop="4dp" />

            <TextView
                android:id="@+id/cartoonGenre"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:paddingTop="8dp"
                android:scrollbars="vertical" />

            <TextView
                android:id="@+id/cartoonEpisodes"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
```

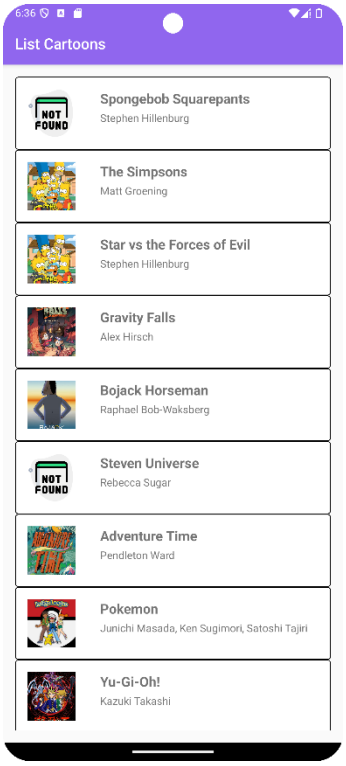
```
                android:paddingTop="4dp" />
        </LinearLayout>
    </ScrollView>
</LinearLayout>
```

Tabel 10. Source Code Hasil Jawaban item_cartoon.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:padding="16dp"
    android:background="@drawable/border">

    <ImageView
        android:id="@+id/cartoonImage"
        android:layout_width="64dp"
        android:layout_height="64dp"
        android:layout_marginEnd="16dp"
        android:scaleType="centerCrop" />

    <LinearLayout
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:orientation="vertical"
        android:paddingStart="16dp">

        <TextView
            android:id="@+id/cartoonTitle"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textStyle="bold"
            android:textSize="18sp" />

        <TextView
            android:id="@+id/cartoonCreator"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:paddingTop="4dp" />
```

```
35      </LinearLayout>
36  </LinearLayout>
```

B. Output Program



Gambar 1. Screenshot Halaman Utama

Gambar 2. Screenshot Halaman Detail

C. Pembahasan

- MainActivity.kt

Kelas MainActivity yang menampilkan daftar kartun menggunakan RecyclerView dan Retrofit untuk mengambil data dari API. Dalam metode onCreate, layout utama diatur menggunakan setContentView, dan Toolbar diinisialisasi dan diatur sebagai ActionBar. RecyclerView diinisialisasi dan diset untuk menggunakan LinearLayoutManager. Dengan menggunakan lifecycleScope.launch, aplikasi memulai coroutine untuk mengambil data kartun dari API. Jika data berhasil diambil, log masing-masing kartun dicatat dan CartoonAdapter diinisialisasi dengan daftar kartun tersebut, dan RecyclerView diatur untuk menggunakan adapter tersebut. Jika terjadi kesalahan saat mengambil data, kesalahan akan dicatat di log.

- DetailActivity.kt

DetailActivity menampilkan detail dari sebuah kartun yang dipilih dari daftar sebelumnya. Dalam metode onCreate, layout utama diatur menggunakan setContentView, dan Toolbar diinisialisasi dan diatur sebagai ActionBar dengan menampilkan tombol back dan mengatur

warnanya. Berbagai elemen UI seperti ImageView dan TextView diinisialisasi untuk menampilkan gambar, judul, pembuat, tahun, genre, dan jumlah episode dari kartun. Data kartun diambil dari Intent menggunakan getSerializableExtra, dan setiap elemen UI diatur dengan data yang sesuai. Gambar kartun dimuat menggunakan library Glide, dengan placeholder dan error image untuk mengatasi masalah pemuatan gambar. Metode onSupportNavigateUp digunakan untuk menangani navigasi kembali ke aktivitas sebelumnya ketika tombol back di Toolbar ditekan.

- Cartoon.kt

Cartoon digunakan untuk merepresentasikan objek kartun. Data class ini menggunakan Moshi untuk deserialisasi JSON, yang ditunjukkan oleh anotasi @Json pada beberapa properti. Cartoon memiliki beberapa properti: id (integer), title (string), year (integer), creator (list of strings), image (nullable string), genre (list of strings), dan episodes (integer). Implementasi interface Serializable memungkinkan objek Cartoon untuk dilewatkan antar aktivitas melalui Intent. Properti image diizinkan null untuk menangani kasus di mana gambar tidak tersedia.

- CartoonAdapter.kt

CartoonAdapter berfungsi sebagai adapter untuk RecyclerView dalam aplikasi Android, yang digunakan untuk menampilkan daftar kartun. Adapter ini menerima daftar objek Cartoon dan lambda onItemClick yang dipanggil ketika sebuah item dalam daftar diklik. Metode onCreateViewHolder mengembalikan CartoonViewHolder dengan menginflate layout item kartun (item_cartoon). Metode onBindViewHolder mengikat data kartun ke ViewHolder di posisi tertentu. getItemCount mengembalikan jumlah kartun dalam daftar.

Kelas CartoonViewHolder adalah ViewHolder untuk adapter ini dan berisi referensi ke elemen UI dalam layout item kartun seperti ImageView dan TextView. Metode bind mengatur teks untuk judul dan pembuat kartun, dan menggunakan library Glide untuk memuat gambar kartun dengan placeholder dan error image. Lambda onItemClick dipanggil ketika item diklik, memungkinkan interaksi pengguna untuk memicu tindakan lebih lanjut, seperti membuka detail kartun.

- CartoonApi.kt

CartoonApi menggunakan Retrofit, sebuah untuk membuat permintaan HTTP di aplikasi Android. Antarmuka ini mendeklarasikan sebuah fungsi getCartoons yang ditandai dengan anotasi @GET untuk menunjukkan bahwa ini adalah permintaan GET ke endpoint "cartoons2D". Fungsi ini ditandai dengan suspend, yang berarti ia dapat dijalankan dalam coroutine dan memungkinkan operasi jaringan dilakukan secara asinkron. Fungsi ini mengembalikan daftar objek Cartoon, yang akan di-deserialize oleh Retrofit dari respons JSON yang diterima dari server.

- CartoonViewModel.kt

CartoonViewModel berfungsi untuk menyimpan dan mengelola data yang berhubungan dengan antarmuka pengguna dalam siklus hidup aktivitas atau fragmen. CartoonViewModel menggunakan LiveData untuk mengamati perubahan data kartun. MutableLiveData _cartoons adalah variabel privat yang diubah dalam ViewModel ini, sementara LiveData cartoons adalah versi publik yang dapat diamati dari luar.

Pada inisialisasi, fungsi fetchCartoons dipanggil untuk mengambil daftar kartun dari API menggunakan coroutine dalam viewModelScope dengan dispatcher IO untuk operasi jaringan. Jika permintaan berhasil, hasilnya diposting ke _cartoons. Jika terjadi kesalahan, exception ditangani dengan mencetak stack trace.

Fungsi filterCartoons digunakan untuk menyaring daftar kartun berdasarkan kueri pencarian. Jika kueri kosong atau null, fetchCartoons dipanggil ulang untuk memuat ulang daftar lengkap. Jika kueri tidak kosong, daftar kartun yang ada difilter berdasarkan judul atau genre yang mengandung kueri, dan hasilnya diposting kembali ke _cartoons untuk diperbarui dalam tampilan.

- RetrofitInstance.kt

RetrofitInstance, sebuah singleton yang menyediakan instance Retrofit untuk aplikasi Android. Objek ini mengonfigurasi Moshi dengan KotlinJsonAdapterFactory untuk mendukung parsing JSON ke objek Kotlin. Instance Retrofit dibuat dengan Retrofit.Builder, menetapkan baseUrl ke "https://api.sampleapis.com/cartoons/" dan menambahkan konverter Moshi. Singleton ini juga menyediakan properti api yang menginisialisasi CartoonApi menggunakan metode create dari Retrofit, memungkinkan aplikasi untuk melakukan

panggilan jaringan ke API kartun dengan mudah dan efisien tanpa perlu mengkonfigurasi Retrofit berulang kali.

- activity_main.xml

Layout XML untuk menampilkan daftar kartun menggunakan RecyclerView. Layout ini menggunakan LinearLayout dengan orientasi vertikal dan berisi dua komponen utama: Toolbar dan RecyclerView. Toolbar berada di bagian atas dengan lebar penuh, tinggi sesuai ukuran standar ActionBar, latar belakang warna utama aplikasi, elevasi untuk efek bayangan, dan judul "List Cartoons" dengan teks berwarna putih. Di bawahnya, RecyclerView yang juga memiliki lebar dan tinggi penuh, dengan padding 16dp, digunakan untuk menampilkan daftar kartun yang diambil dari API. Layout ini memastikan tampilan yang terstruktur dan mudah diatur untuk daftar kartun.

- activity_detail.xml

Layout XML untuk menampilkan informasi rinci tentang sebuah kartun. Layout ini menggunakan LinearLayout sebagai root dengan orientasi vertikal, berisi Toolbar dan ScrollView untuk mengatur elemen-elemen UI secara terstruktur. Toolbar di bagian atas memiliki lebar penuh dan tinggi sesuai ukuran standar ActionBar, dengan latar belakang warna utama aplikasi, dan elevasi untuk memberikan efek bayangan. Judul Toolbar tidak ditentukan di sini karena kemungkinan akan diatur secara dinamis di dalam kode.

Di bawah Toolbar, terdapat ScrollView yang memungkinkan konten di dalamnya untuk digulir, memastikan semua informasi dapat dilihat meskipun melebihi tinggi layar. Di dalam ScrollView, terdapat LinearLayout vertikal yang berisi komponen-komponen UI untuk menampilkan detail kartun. ImageView dengan ID cartoonImage digunakan untuk menampilkan gambar kartun, dengan lebar penuh dan tinggi tetap 600dp serta scaleType centerCrop untuk memastikan gambar memenuhi seluruh area yang tersedia. Beberapa TextView digunakan untuk menampilkan berbagai informasi kartun: cartoonTitle untuk judul dengan gaya teks tebal dan ukuran 24sp, cartoonCreator untuk pembuat kartun, cartoonYear untuk tahun rilis, cartoonGenre untuk genre dengan scrollbar vertikal jika teksnya panjang, dan cartoonEpisodes untuk jumlah episode. Setiap TextView memiliki padding untuk memastikan tampilan yang rapi dan mudah dibaca. Layout ini dirancang untuk memberikan tampilan yang komprehensif dan informatif bagi pengguna yang ingin melihat detail dari kartun yang dipilih.

- item_cartoon.xml

Layout XML untuk menampilkan informasi ringkas tentang sebuah kartun. Layout ini menggunakan LinearLayout dengan orientasi horizontal sebagai root, dan mengatur elemen-elemen UI secara terstruktur untuk menampilkan gambar dan informasi kartun.

Di dalam LinearLayout utama, terdapat ImageView dengan ID cartoonImage yang digunakan untuk menampilkan gambar kartun. ImageView ini memiliki lebar dan tinggi tetap 64dp, margin akhir 16dp, dan scaleType centerCrop untuk memastikan gambar memenuhi area yang tersedia dengan proporsi yang tepat. Selanjutnya, terdapat LinearLayout vertikal di sebelah kanan ImageView, yang memiliki lebar 0dp dengan layout_weight 1 untuk mengambil sisa ruang yang tersedia di baris horizontal tersebut. LinearLayout vertikal ini memiliki padding awal 16dp dan berisi dua TextView.

TextView pertama dengan ID cartoonTitle digunakan untuk menampilkan judul kartun, dengan gaya teks tebal dan ukuran teks 18sp. TextView kedua dengan ID cartoonCreator digunakan untuk menampilkan nama pembuat kartun, dengan padding atas 4dp untuk memberikan sedikit ruang antara teks judul dan teks pembuat. Layout ini juga memiliki padding keseluruhan 16dp dan latar belakang yang menggunakan drawable border untuk memberikan efek batas pada setiap item dalam RecyclerView, membuatnya tampak lebih terorganisir dan menarik.

Tautan Git

Praktikum-Pemrograman-Mobile/Praktikum Modul 4 at main · harioct/Praktikum-Pemrograman-Mobile (github.com)