LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 3

BUILD A SCROLLABLE LIST

Oleh:

Siti Ratna Dwinta Sari NIM. 2310817120002

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
MEI 2025

LEMBAR PENGESAHAN
LAPORAN PRAKTIKUM PEMROGRAMAN MOBILE
MODUL 3

Laporan Praktikum Pemrograman Mobile Modul 3: Build A Scrollable List ini disusun sebagai syarat lulus mata kuliah Praktikum Pemrograman Mobile. Laporan Praktikum ini dikerjakan oleh:

Nama Praktikan : Siti Ratna Dwinta Sari
NIM : 2310817120002

Menyetujui, Asisten Praktikum	Mengetahui, Dosen Penanggung Jawab Praktikum
Zulfa Auliya Akbar NIM. 2210817210026	Muti`a Maulida S.Kom M.T.I NIP. 19881027 201903 20 13

DAFTAR ISI

DAFTAR GAMBAR

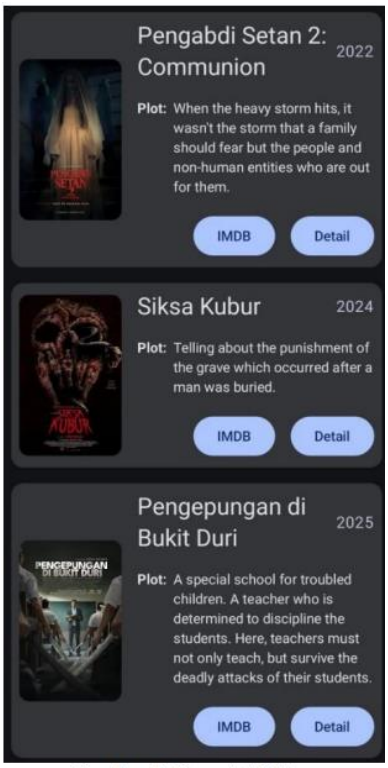
DAFTAR TABEL

SOAL 1

Buatlah sebuah aplikasi Android menggunakan XML atau Jetpack Compose yang dapat menampilkan list dengan ketentuan berikut:

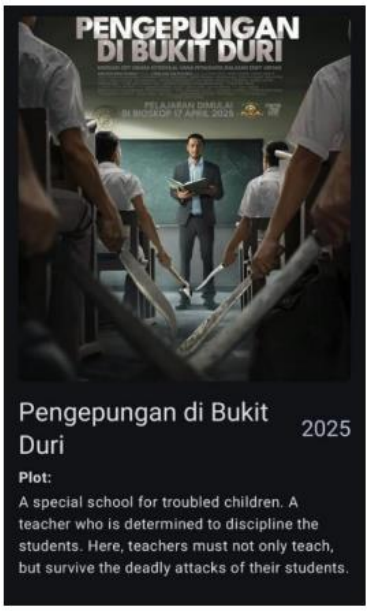
1. List menggunakan fungsi RecyclerView (XML) atau LazyColumn (Compose)
2. List paling sedikit menampilkan 5 item. Tema item yang ingin ditampilkan bebas 3. Item pada list menampilkan teks dan gambar sesuai dengan contoh di bawah 4.Terdapat 2 button dalam list, dengan fungsi berikut:
    a. Button pertama menggunakan intent eksplisit untuk membuka aplikasi atau browser lain
    b. Button kedua menggunakan Navigation component/intent untuk membuka laman detail item
5. Sudut item pada list dan gambar di dalam list melengkung atau rounded corner menggunakan Radius
6. Saat orientasi perangkat berubah/dirotasi, baik ke portrait maupun landscape, aplikasi responsif dan dapat menunjukkan list dengan baik. Data di dalam list tidak boleh hilang
7. Aplikasi menggunakan arsitektur single activity (satu activity memiliki beberapa fragment)
8. Aplikasi berbasis XML harus menggunakan ViewBinding

UI item list harus berisi 1 gambar, 2 button (intent eksplisit dan navigasi), dan 2 baris teks dan setiap baris memiliki 2 teks yang berbeda. Diusahakan agar desain UI item list menyerupai UI berikut:



Gambar 1. Contoh UI List

Desain UI laman detail bebas, tetapi diusahakan untuk mengikuti kaidah desain Material Design dan data item ditampilkan penuh di laman detail seperti contoh berikut:



Gambar 2. Contoh UI Detail

A. Source Code

1. MainActivity.kt

Tabel 1. Source Code Jawaban Soal 1

```
1  package com.example.dreamstravel
2
3  import android.os.Bundle
4  import androidx.activity.compose.setContent
5  import androidx.activity.enableEdgeToEdge
6  import androidx.compose.foundation.layout.fillMaxSize
7  import androidx.compose.foundation.layout.padding
8  import androidx.compose.material3.Scaffold
9  import androidx.compose.material3.Text
10 import androidx.compose.runtime.Composable
11 import androidx.compose.ui.Modifier
12 import androidx.compose.ui.tooling.preview.Preview
13 import androidx.appcompat.app.AppCompatActivity
14 import androidx.navigation.fragment.NavHostFragment
15 import com.example.dreamstravel.databinding.ActivityMainBinding
16 import com.example.dreamstravel.ui.theme.DreamsTravelTheme
17
18 class MainActivity : AppCompatActivity() {
19
20     private lateinit var binding: ActivityMainBinding  // ←
   Binding object
21
22     override fun onCreate(savedInstanceState: Bundle?) {
23         super.onCreate(savedInstanceState)
24
25         binding = ActivityMainBinding.inflate(layoutInflater)
26         setContentView(binding.root)
27
28         val navHostFragment =
   supportFragmentManager.findFragmentById(R.id.nav_host_fragment)
   as NavHostFragment
29         val navController = navHostFragment.navController
30     }
31 }
```

2. HomeFragment.kt

Tabel 2. Source Code Jawaban Soal 1

```
1  package com.example.dreamstravel
2
3  import android.os.Bundle
4  import android.view.LayoutInflater
5  import android.view.View
6  import android.view.ViewGroup
```

```
7  import androidx.fragment.app.Fragment
8  import androidx.fragment.app.viewModels
9  import androidx.navigation.fragment.findNavController
10 import androidx.recyclerview.widget.LinearLayoutManager
11 import com.example.dreamstravel.databinding.FragmentHomeBinding
12
13 class HomeFragment : Fragment() {
14     private var _binding: FragmentHomeBinding? = null
15     private val binding get() = _binding!!
16
17     private val viewModel: NegaraViewModel by viewModels()
18
19     override fun onCreateView(inflater: LayoutInflater,
   container: ViewGroup?, savedInstanceState: Bundle?): View? {
20         _binding = FragmentHomeBinding.inflate(inflater,
   container, false)
21         return binding.root
22     }
23
24     override fun onViewCreated(view: View, savedInstanceState:
   Bundle?) {
25         val adapter = NegaraAdapter(viewModel.loadNegara(),
   findNavController())
26         binding.rvNegara.layoutManager =
   LinearLayoutManager(requireContext())
27         binding.rvNegara.adapter = adapter
28     }
29
30     override fun onDestroyView() {
31         super.onDestroyView()
32         _binding = null
33     }
34
35 }
```

3. DetailFragment.kt

Tabel 3. Source Code Jawaban Soal 1

```
1  package com.example.dreamstravel
2
3  import android.os.Bundle
4  import androidx.fragment.app.Fragment
5  import android.view.LayoutInflater
6  import android.view.View
7  import android.view.ViewGroup
8
```

```
    import
9   com.example.dreamstravel.databinding.FragmentDetailBinding
10
11  class DetailFragment : Fragment() {
12
13      private var _binding: FragmentDetailBinding? = null
14      private val binding get() = _binding!!
15
16      override fun onCreateView(
17          inflater: LayoutInflater,
18          container: ViewGroup?,
19          savedInstanceState: Bundle?
20      ): View? {
            _binding = FragmentDetailBinding.inflate(inflater,
21  container, false)
22          return binding.root
23      }
24
        override fun onViewCreated(view: View, savedInstanceState:
25  Bundle?) {
            val namaNegara = arguments?.getString("namaNegara") ?:
26  "Tidak ada nama"
            val alasan = arguments?.getString("alasan") ?: "Tidak
27  ada tahun"
            val gambar = arguments?.getInt("gambarResId") ?:
28  R.drawable.japan
29
30          binding.imgCountry.setImageResource(gambar)
31          binding.tvCountryName.text = namaNegara
32          binding.tvCountryReason.text = alasan
33      }
34
35      override fun onDestroyView() {
36          super.onDestroyView()
37          _binding = null
38      }
    }
```

4. NegaraAdapter.kt

Tabel 4. Source Code Jawaban Soal 1

```
1   package com.example.dreamstravel
2
3   import android.content.Intent
4   import androidx.core.net.toUri
5   import android.view.LayoutInflater
6   import android.view.ViewGroup
```

```
7  import androidx.navigation.NavController
8  import androidx.recyclerview.widget.RecyclerView
9  import com.example.dreamstravel.databinding.ItemNegaraBinding
10
11 class NegaraAdapter(
12     private val listNegara: List<NegaraModel>,
13     private val navController: NavController
14 ) : RecyclerView.Adapter<NegaraAdapter.NegaraViewHolder>() {
15
16     inner class NegaraViewHolder(val binding: ItemNegaraBinding)
   : RecyclerView.ViewHolder(binding.root)
17
18     override fun onCreateViewHolder(parent: ViewGroup, viewType:
   Int): NegaraViewHolder {
19         val binding =
   ItemNegaraBinding.inflate(LayoutInflater.from(parent.context),
   parent, false)
20         return NegaraViewHolder(binding)
21     }
22
23     override fun onBindViewHolder(holder: NegaraViewHolder,
   position: Int) {
24         val negara = listNegara[position]
25         holder.binding.tvItemName.text = negara.nama
26
27 holder.binding.imgItemPhoto.setImageResource(negara.gambarResId)
28         holder.binding.btnWiki.setOnClickListener{
29             var url = negara.wikiUrl
30             val intent = Intent(Intent.ACTION_VIEW, url.toUri())
31             it.context.startActivity(intent)
32         }
33
34         holder.binding.buttonDetail.setOnClickListener {
35             val bundle = android.os.Bundle().apply {
36                 putInt("gambarResId", negara.gambarResId)
37                 putString("namaNegara", negara.nama)
38                 putString("alasan", negara.alasan)
39             }
40             navController.navigate(R.id.detailFragment, bundle)
41         }
42     }
43     override fun getItemCount() = listNegara.size
44 }
```

5. NegaraModel.kt

Tabel 5. Source Code Jawaban Soal 1

```
1  package com.example.dreamstravel
2
3
4  data class NegaraModel(
5      val nama: String,
6      val tahun: String,
7      val alasan: String,
8      val gambarResId: Int,
9      val wikiUrl: String
10 )
```

6. NegaraViewModel.kt

Tabel 6. Source Code Jawaban Soal 1

```
1  package com.example.dreamstravel
2  import android.app.Application
3  import androidx.lifecycle.AndroidViewModel
4  class NegaraViewModel(application: Application) :
   AndroidViewModel(application) {
5      private val context =
   getApplication<Application>().applicationContext
6
7
8      fun loadNegara(): List<NegaraModel> {
9          val namaNegara =
   context.resources.getStringArray(R.array.list_negara)
10         val tahun =
   context.resources.getStringArray(R.array.list_tahun)
11         val alasan =
   context.resources.getStringArray(R.array.list_alasan)
12         val link =
   context.resources.getStringArray(R.array.list_link)
13         val gambar = listOf(
14             R.drawable.japan,
15             R.drawable.swiss,
16             R.drawable.korsel,
17             R.drawable.arabsaudi,
18             R.drawable.france
19         )
20
21         return namaNegara.indices.map { i ->
22             NegaraModel(
23                 namaNegara[i],
24                 tahun[i],
```

```
                    alasan[i],
                    gambar[i],
                    link[i]
                )
            }
        }
}
```

7. Activity_main.xml

Tabel 7. Source Code Jawaban Soal 1

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Selamat datang di DreamsTravel!"
        android:textSize="18sp"
        android:layout_gravity="center_horizontal"
        android:layout_margin="16dp"/>

    <androidx.fragment.app.FragmentContainerView
        android:id="@+id/nav_host_fragment"

android:name="androidx.navigation.fragment.NavHostFragment"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        app:defaultNavHost="true"
        app:navGraph="@navigation/nav_graph" />
</LinearLayout>
```

8. Fragment_detail.xml

Tabel 8. Source Code Jawaban Soal 1

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
```

```
6       android:layout_height="match_parent"
7       tools:context=".DetailFragment">
8
9       <ImageView
10          android:id="@+id/imgCountry"
11          android:layout_width="200dp"
12          android:layout_height="200dp"
13          android:layout_marginTop="32dp"
14          android:layout_centerHorizontal="true"
15          app:layout_constraintTop_toTopOf="parent"
16          app:layout_constraintEnd_toEndOf="parent"
17          app:layout_constraintStart_toStartOf="parent" />
18
19      <TextView
20          android:id="@+id/tvCountryName"
21          android:layout_width="wrap_content"
22          android:layout_height="wrap_content"
23          android:textSize="18sp"
24
    app:layout_constraintTop_toBottomOf="@id/imgCountry"
25          app:layout_constraintStart_toStartOf="parent"
26          app:layout_constraintEnd_toEndOf="parent"
27          android:text="Country Name"/>
28
29      <TextView
30          android:id="@+id/tvCountryReason"
31          android:layout_width="wrap_content"
32          android:layout_height="wrap_content"

33  app:layout_constraintTop_toBottomOf="@id/tvCountryName"
34          app:layout_constraintStart_toStartOf="parent"
35          app:layout_constraintEnd_toEndOf="parent"
36          android:text="Reason"/>
37  </androidx.constraintlayout.widget.ConstraintLayout>
```

9. Fragment_home.xml

Tabel 9. Source Code Jawaban Soal 1

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <androidx.constraintlayout.widget.ConstraintLayout
    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       xmlns:tools="http://schemas.android.com/tools">
7
```

```
8       <androidx.recyclerview.widget.RecyclerView
9           android:id="@+id/rvNegara"
10          android:layout_width="0dp"
11          android:layout_height="0dp"
12          android:clipToPadding="false"
13          android:padding="8dp"
14          android:contentDescription="cute"
15          app:layout_constraintTop_toTopOf="parent"
16          app:layout_constraintBottom_toBottomOf="parent"
17          app:layout_constraintStart_toStartOf="parent"
18          app:layout_constraintEnd_toEndOf="parent"
19          tools:listitem="@layout/item_negara" />
20  </androidx.constraintlayout.widget.ConstraintLayout>
```

10. Item_negara.xml

Tabel 10. Source Code Jawaban Soal 1

```
1   <?xml version=”1.0” encoding=”utf-8”?>
2   <15ndroid.cardview.widget.CardView
    xmlns:android=”http://schemas.android.com/apk/res/android”
3       xmlns:card_view=”http://schemas.android.com/apk/res-
    auto”
4       xmlns:tools=”http://schemas.android.com/tools”
5       xmlns:app=”http://schemas.android.com/apk/res-auto”
6       android:layout_width=”match_parent”
7       android:layout_height=”wrap_content”
8       android:layout_margin=”@dimen/card_margin”
9       app:cardCornerRadius=”@dimen/card_radius”
10      app:cardElevation=”4dp”>
11
12      <15ndroid.constraintlayout.widget.ConstraintLayout
13          android:layout_width=”match_parent”
14          android:layout_height=”wrap_content”
15          android:padding=”@dimen/card_padding”>
16
17
    <com.google.android.material.imageview.ShapeableImageView
18              android:id=”@+id/img_item_photo”
19              android:layout_width=”80dp”
20              android:layout_height=”100dp”
21              android:scaleType=”centerCrop”
22              app:layout_constraintStart_toStartOf=”parent”
23              app:layout_constraintTop_toTopOf=”parent”
24
    app:layout_constraintBottom_toBottomOf=”parent”
25
```

```
            android:contentDescription="@string/img_desc"
/>

        <TextView
            android:id="@+id/tv_item_name"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:textSize="18sp"
            android:textStyle="bold"
            android:textColor="@android:color/black"

app:layout_constraintStart_toEndOf="@id/img_item_photo"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0"
            android:layout_marginStart="8dp"
            android:layout_marginEnd="8dp"
            tools:text="Jepang" />

        <Button
            android:id="@+id/button_detail"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/btn_detail"
            android:minHeight="48dp"
            android:minWidth="48dp"

android:paddingHorizontal="@dimen/button_padding"

app:layout_constraintTop_toBottomOf="@id/tv_item_name"

app:layout_constraintStart_toStartOf="@id/tv_item_name"

app:layout_constraintBottom_toBottomOf="parent" />

        <Button
            android:id="@+id/btn_wiki"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/btn_wiki"
            android:minHeight="48dp"
            android:minWidth="48dp"

android:paddingHorizontal="@dimen/button_padding"

app:layout_constraintTop_toBottomOf="@id/tv_item_name"
```

```
64
   app:layout_constraintStart_toEndOf=”@id/button_detail”
65
66 app:layout_constraintBottom_toBottomOf=”parent”
67             android:layout_marginStart=”8dp” />
68
       </android.constraintlayout.widget.ConstraintLayout>
   </androidx.cardview.widget.CardView>
```

11. nav_graph.xml

Tabel 11. Source Code Jawaban Soal 1

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <navigation
   xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:id="@+id/nav_graph"
6      app:startDestination="@id/homeFragment">
7
8      <fragment
9          android:id="@+id/homeFragment"
10
   android:name="com.example.dreamstravel.HomeFragment"
11         android:label="Home"
12         tools:layout="@layout/fragment_home">
13         <action
14
   android:id="@+id/action_homeFragment_to_detailFragment"
15             app:destination="@id/detailFragment" />
16     </fragment>
17
18     <fragment
19         android:id="@+id/detailFragment"
20
   android:name="com.example.dreamstravel.DetailFragment"
21         android:label="Detail"
22         tools:layout="@layout/fragment_detail" />
23 </navigation>
```

12. strings.xml

Tabel 12. Source Code Jawaban Soal 1

```
1  <resources>
2      <string name="app_name">DreamsTravel</string>
```

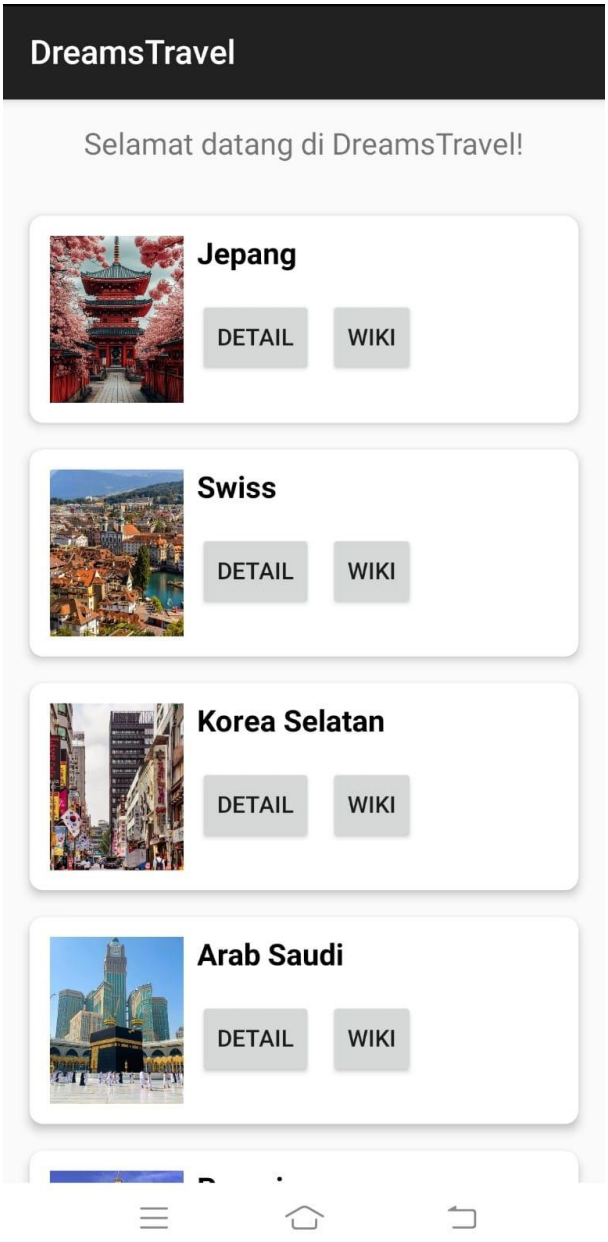
```
    <string name="btn_detail">Detail</string>
    <string name="btn_wiki">Wiki</string>
    <string name="img_desc">Gambar negara impian</string>
    <string-array name="list_negara">
        <item>Jepang</item>
        <item>Swiss</item>
        <item>Korea Selatan</item>
        <item>Arab Saudi</item>
        <item>Prancis</item>
    </string-array>
    <string-array name="list_tahun">
        <item>2025</item>
        <item>2026</item>
        <item>2027</item>
        <item>2028</item>
        <item>2029</item>
    </string-array>
    <string-array name="data_gambar">
        <item>@drawable/japan</item>
        <item>@drawable/swiss</item>
        <item>@drawable/korsel</item>
        <item>@drawable/arabsaudi</item>
        <item>@drawable/france</item>
    </string-array>
    <string-array name="list_alasan">
        <item>Saya suka negara jepang karena budaya dan ingin ketemu Kento Yamazaki</item>
        <item>Negara Swiss itu pemandangan alamnya bagus banget, kayak alam mimpi!</item>
        <item>Saya suka K-Pop dan K-Dramanya juga semua sangat seru, alasan saya ingin kesini karena ingin bertemu Enhypen</item>
        <item>Karena saya beragama islam, saya ingin melaksanakan ibadah haji</item>
        <item>Saya juga ingin ke Prancis, karena katanya paris ini kota yang romantis</item>
    </string-array>

    <string-array name="list_link">
        <item>https://en.wikipedia.org/wiki/Japan</item>
<item>https://en.wikipedia.org/wiki/Switzerland</item>
<item>https://id.wikipedia.org/wiki/Korea_Selatan</item>
<item>https://id.wikipedia.org/wiki/Arab_Saudi</item>
        <item>https://id.wikipedia.org/wiki/Prancis</item>
```

```
    </string-array>

</resources>
```

B. Output Program



Gambar 1. Screenshot Halaman Utama

Gambar 2. Screenshot Halaman Detail

Gambar 3. Screenshot Halaman Link Wikipedia

C. Pembahasan

1. MainActivity.kt:

Kode di atas adalah isi dari file MainActivity.kt dalam proyek Android yang menggunakan Jetpack Navigation Component dan View Binding. Kelas MainActivity ini merupakan entry point atau layar utama saat aplikasi pertama kali dijalankan. Kode ini menggunakan pendekatan berbasis Activity tradisional (bukan Compose) meskipun ada sedikit bagian Compose yang belum digunakan secara aktif.

Pertama-tama, ada properti binding yang dibuat dengan tipe ActivityMainBinding. Ini adalah fitur View Binding yang memudahkan kita mengakses elemen-elemen XML tanpa perlu pakai `findViewById()`. Di dalam `fungsi onCreate()`, yaitu fungsi yang pertama kali dijalankan saat activity dibuat, kita menginisialisasi objek binding dengan cara meng-inflate layout dari activity_main.xml, lalu menampilkannya dengan `setContentView(binding.root)`.

Selanjutnya, kita mengambil fragment navigasi (komponen yang mengatur perpindahan antar layar/halaman di aplikasi) yang ada di dalam activity_main.xml melalui ID `nav_host_fragment`. Fragment ini kemudian dikonversi ke tipe `NavHostFragment` agar bisa kita akses `navController-nya`, yaitu komponen yang mengatur logika navigasi seperti berpindah halaman. Meskipun tidak ada aksi lanjut di kode ini, biasanya navController ini digunakan untuk mengatur arah navigasi berdasarkan interaksi pengguna.

2. HomeFragment.kt

Kode di atas adalah isi dari `HomeFragment.kt`, yaitu salah satu tampilan (fragment) di dalam aplikasi Android yang menampilkan daftar negara menggunakan `RecyclerView`. Fragment ini memakai View Binding dan juga `ViewModel` untuk mengambil data.

Pertama-tama, ada properti _binding yang digunakan untuk mengakses elemen-elemen layout dari file `fragment_home.xml` tanpa harus pakai `findViewById()`. Karena lifecycle fragment bisa berubah-ubah, binding ini di-null-kan kembali di `onDestroyView()` untuk mencegah kebocoran memori.

Kemudian, ada `viewModel` bertipe `NegaraViewModel` yang digunakan untuk mengambil data negara. `viewModels()` adalah cara instan untuk mendapatkan instance ViewModel yang sesuai dengan fragment ini.

Fungsi `onCreateView()` akan dipanggil saat fragment akan membuat tampilannya. Di sini, layout dari fragment di-inflate dan dikembalikan. Setelah tampilan dibuat, fungsi `onViewCreated()` akan dipanggil, dan di sinilah logika utama ditaruh. Dalam hal ini, dibuatlah sebuah adapter NegaraAdapter, yang isinya diisi dari data yang diambil lewat `viewModel.loadNegara()`. Adapter ini juga diberi `findNavController()` agar bisa mengatur navigasi saat item di-klik. RecyclerView kemudian disetel dengan LinearLayoutManager (agar tampilannya ke bawah seperti daftar biasa) dan dihubungkan dengan adapter.

3. DetailFragment.kt

Kode di atas adalah isi dari `DetailFragment.kt`, yaitu salah satu tampilan dalam aplikasi yang bertugas menampilkan detail dari sebuah negara yang dipilih oleh pengguna. Fragment ini menggunakan View Binding agar bisa mengakses elemen-elemen layout fragment_detail.xml dengan lebih aman dan efisien tanpa `findViewById()`.

Pertama-tama, dibuat variabel _binding yang akan menyimpan objek binding sementara fragment aktif. Nilai ini kemudian diakses melalui binding yang aman karena sudah dipastikan tidak null. Binding ini di-inflate pada fungsi `onCreateView()`, lalu digunakan untuk menampilkan isi layout fragment tersebut.

Lalu, di dalam `onViewCreated()`, fragment ini mengambil data yang dikirim dari fragment sebelumnya menggunakan arguments. Data yang diambil berupa `namaNegara` (nama negara), `alasan` (alasan memilih negara tersebut), dan `gambarResId` (ID gambar dari drawable). Kalau data tidak ditemukan, maka akan ditampilkan nilai default seperti "Tidak ada nama" atau `gambar default R.drawable.japan`.

Setelah data berhasil diambil, semuanya ditampilkan ke tampilan: gambar dimunculkan dengan setImageResource, nama negara ditaruh di `tvCountryName`, dan alasannya di `tvCountryReason`. Terakhir, pada `onDestroyView()`, binding di-null-kan agar tidak menyebabkan memory leak.

4. **NegaraAdapter.kt**

Kode di atas adalah isi dari kelas `NegaraAdapter`, yaitu adapter untuk `RecyclerView` yang menampilkan daftar negara. Adapter ini bertugas mengatur bagaimana data ditampilkan di setiap item daftar. NegaraAdapter menerima dua parameter: `listNegara` (daftar data negara dari model NegaraModel) dan `navController` (yang digunakan untuk navigasi ke tampilan detail). Di dalamnya ada kelas NegaraViewHolder sebagai inner class, yang menyimpan binding dari layout item_negara.xml. Binding ini digunakan untuk mengakses elemen UI dalam tiap item list tanpa harus menggunakan `findViewById()`.

Fungsi `onCreateViewHolder()` dipanggil saat `RecyclerView` ingin membuat tampilan baru. Di sini, layout item negara di-inflate dan dibungkus ke dalam NegaraViewHolder. Kemudian, fungsi `onBindViewHolder()` akan mengisi data ke dalam item berdasarkan posisi item dalam list. Misalnya, nama negara ditampilkan ke tvItemName, gambar negara ke `imgItemPhoto`.

Ketika tombol Wikipedia (btnWiki) diklik, aplikasi akan membuka link Wikipedia dari negara tersebut di browser, menggunakan Intent.`ACTION_VIEW` dan mengonversi URL-nya menjadi Uri. Lalu ada tombol buttonDetail, yang ketika diklik akan membawa pengguna ke `DetailFragment` dengan membawa data negara tersebut menggunakan Bundle. Data yang dikirim meliputi gambar, nama negara, dan alasan memilih negara tersebut. Navigasi dilakukan lewat `navController.navigate`() menuju detailFragment.

5. **NegaraModel.kt**

Kode di atas adalah deklarasi dari NegaraModel, yaitu sebuah data class di Kotlin yang digunakan untuk merepresentasikan data dari satu negara dalam aplikasi. Karena ini data class, maka otomatis Kotlin akan menyediakan fungsi-fungsi standar seperti `toString()`, `equals()`, dan `hashCode()` secara otomatis, yang sangat membantu untuk keperluan data.

Di dalam NegaraModel, ada lima properti yang mewakili informasi tentang negara tersebut:

- `nama`: berisi nama negara (misalnya "Jepang"),
- `tahun`: berisi informasi tahun kunjungan atau rencana (meskipun di kode sebelumnya belum digunakan),

- `alasan`: alasan kenapa negara itu dipilih,
- `gambarResId`: menyimpan ID resource dari gambar negara (biasanya dari folder res/drawable)
- `wikiUrl`: berisi link ke Wikipedia negara tersebut yang nantinya bisa dibuka lewat browser.

Model ini nantinya digunakan oleh NegaraAdapter untuk ditampilkan ke tampilan list dan juga dikirim ke `DetailFragment`. Jadi, bisa dibilang `NegaraModel` ini adalah blueprint atau cetakan dari setiap item negara yang ditampilkan di aplikasi.

6. NegaraViewModel.kt

Kode di atas adalah isi dari `NegaraViewModel`, yaitu kelas yang berfungsi sebagai jembatan antara data dan tampilan (UI) dalam arsitektur MVVM (`Model-View-ViewModel`). Kelas ini menuruni `AndroidViewModel`, yang artinya ia bisa mengakses Application langsung, berguna ketika kita butuh akses ke resource seperti string array atau drawable yang ada di res. Di dalamnya, ada fungsi `loadNegara()` yang akan memuat data dari file `strings.xml` dan mengembalikannya sebagai daftar objek NegaraModel. Data yang dimuat meliputi:

- `namaNegara`: array string yang berisi nama-nama negara,
- `tahun`: array string tahun kunjungan atau keinginan mengunjungi,
- `alasan`: array string yang menjelaskan alasan memilih negara tersebut,
- `link`: array string berisi URL Wikipedia tiap negara,
- `gambar`: daftar resource ID gambar negara dari folder drawable.

Fungsi ini kemudian menggunakan indices.map untuk menggabungkan data-data dari array tadi berdasarkan indeks yang sama, sehingga setiap elemen array (misalnya nama ke-0, tahun ke-0, dan seterusnya) dikombinasikan menjadi satu objek `NegaraModel`. Hasil akhirnya adalah sebuah list berisi data lengkap tiap negara yang siap ditampilkan di `RecyclerView` melalui `NegaraAdapter`.

7. **Activity_main.xml**

XML di atas merupakan bagian dari layout aplikasi Android yang ditulis dalam bahasa XML, dan biasanya digunakan untuk menentukan tampilan antarmuka pengguna (UI). Layout ini menggunakan `LinearLayout` sebagai wadah utamanya dengan orientasi vertikal, yang berarti semua elemen di dalamnya akan ditampilkan dari atas ke bawah. Pertama, ada sebuah elemen TextView yang digunakan untuk menampilkan teks `"Selamat datang di DreamsTravel!".` Teks ini ditampilkan di tengah secara horizontal (`layout_gravity="center_horizontal")` dan diberi margin sebesar `16dp` agar tidak terlalu rapat dengan elemen lain. Ukuran teksnya diatur menjadi `18sp` agar cukup besar dan mudah dibaca. Kemudian, di bawah `TextView`, terdapat `FragmentContainerView` yang berfungsi sebagai tempat untuk menampilkan fragment navigasi. `FragmentContainerView` ini menggunakan NavHostFragment dari library Jetpack Navigation, yang memungkinkan kita untuk mengatur navigasi antar fragment di dalam aplikasi. `Properti app:defaultNavHost="true"` memastikan bahwa fragment ini akan menangani aksi navigasi secara default, dan `app:navGraph="@navigation/nav_graph"` menunjukkan file grafik navigasi yang digunakan untuk menentukan alur berpindah antar fragment dalam aplikasi.

8. **Fragment_detail.xml**

XML di atas merupakan layout tampilan detail pada aplikasi Android yang menggunakan `ConstraintLayout` sebagai root layout-nya. `ConstraintLayout` ini memungkinkan setiap elemen UI untuk diatur posisinya secara fleksibel dan presisi dengan menghubungkannya ke elemen lain atau ke parent layout. Layout ini biasanya digunakan agar tampilan lebih responsif di berbagai ukuran layar. Pertama, terdapat sebuah `ImageView` dengan ID `imgCountry` yang berfungsi untuk menampilkan gambar, misalnya gambar bendera atau pemandangan dari suatu negara. Ukuran gambar ini diatur menjadi `200dp x 200dp` dan ditempatkan di tengah atas layar dengan margin atas `32dp`. Penempatan ini diatur menggunakan properti constraint seperti `app:layout_constraintTop_toTopOf="parent"` dan `app:layout_constraintStart_toStartOf="parent",` sehingga posisinya

selalu berada di bagian atas dan tengah secara horizontal. Di bawah gambar, ada `TextView` dengan ID `tvCountryName` yang digunakan untuk menampilkan nama negara. Teks ini akan muncul tepat di bawah gambar karena dihubungkan menggunakan `layout_constraintTop_toBottomOf="@id/imgCountry".` Teks ini juga dibuat agar berada di tengah secara horizontal. Terakhir, ada lagi `TextView` dengan `ID tvCountryReason` yang menampilkan alasan kenapa negara tersebut menarik atau layak dikunjungi. Sama seperti elemen sebelumnya, teks ini diletakkan di bawah `tvCountryName` dan juga diatur agar berada di tengah.

9. Fragment_home.xml

XML di atas adalah layout untuk sebuah tampilan dalam aplikasi Android yang menggunakan `ConstraintLayout` sebagai layout utamanya. Di dalam layout ini hanya terdapat satu komponen, yaitu `RecyclerView`, yang merupakan elemen penting ketika kita ingin menampilkan daftar data dalam bentuk scroll, seperti daftar negara, daftar tempat wisata, atau item lainnya.

`RecyclerView` di sini memiliki `ID rvNegara`, dan ukurannya diatur agar memenuhi seluruh layar dengan menggunakan `layout_width="0dp"` dan `layout_height="0dp"` lalu dihubungkan ke semua sisi parent menggunakan constraint (Top, Bottom, Start, End). Ini artinya `RecyclerView` akan membentang dari atas sampai bawah dan dari kiri ke kanan layar.

`Properti clipToPadding="false"` berarti elemen di dalam daftar bisa terlihat meskipun berada di area padding, dan `padding="8dp"` memberikan jarak di sekeliling isi daftar agar tidak terlalu mepet ke tepi layar. Selain itu, `tools:listitem="@layout/item_negara"` adalah petunjuk khusus untuk Android Studio agar saat kita melihat preview-nya, akan ditampilkan contoh isi daftar yang berasal dari layout item bernama item_negara.

10. Item_negara.xml

XML di atas adalah layout untuk sebuah item dalam daftar (biasanya digunakan dalam `RecyclerView`) dan dirancang menggunakan komponen `CardView` untuk memberikan tampilan yang elegan dan modern, seperti kartu dengan sudut melengkung dan bayangan

(elevation). CardView ini berisi ConstraintLayout yang digunakan untuk mengatur posisi elemen-elemen di dalamnya agar tetap responsif di berbagai ukuran layar.

Di dalam CardView, terdapat tiga elemen utama. Pertama, ada ShapeableImageView yang digunakan untuk menampilkan gambar dari item, misalnya gambar negara. Gambar ini diposisikan di sebelah kiri, dan menggunakan centerCrop agar gambar terlihat rapi dan memenuhi ruang yang disediakan tanpa merusak proporsi.

Kedua, terdapat TextView yang akan menampilkan nama negara (atau item lainnya). Teks ini diletakkan di samping kanan gambar dan diatur agar tampil tegas (dengan ukuran 18sp dan huruf tebal). Penempatan teks ini sangat fleksibel karena menggunakan ConstraintLayout untuk mengatur jarak dengan gambar di sebelah kiri dan batas layout di kanan. Ketiga, terdapat dua buah Button. Yang pertama adalah tombol "Detail" (button_detail) yang biasanya digunakan untuk melihat informasi lebih lengkap tentang item tersebut. Di sebelah kanannya, ada tombol "Wiki" (btn_wiki) yang kemungkinan besar mengarahkan pengguna ke halaman Wikipedia terkait item tersebut. Kedua tombol ini ditempatkan sejajar di bawah TextView dan memiliki padding agar nyaman ditekan.

11. Nav_graph.xml

XML di atas adalah bagian dari navigation graph dalam aplikasi Android yang menggunakan Jetpack Navigation Component. Navigation graph ini digunakan untuk mengatur dan mengelola alur perpindahan antar halaman atau fragment di dalam aplikasi. Jadi, kita bisa menentukan dari fragment mana ke fragment mana pengguna bisa berpindah, dan bagaimana arah alurnya. Dalam file ini, ditentukan bahwa homeFragment adalah halaman awal yang akan ditampilkan ketika aplikasi dibuka, ditandai dengan app:startDestination="@id/homeFragment". homeFragment merujuk pada com.example.dreamstravel.HomeFragment, yang tampil menggunakan layout fragment_home. Di dalamnya juga ada sebuah action, yaitu action_homeFragment_to_detailFragment, yang berarti dari homeFragment pengguna bisa melakukan navigasi ke detailFragment. Lalu, ada detailFragment yang merupakan tujuan navigasi, dengan nama com.example.dreamstravel.DetailFragment, dan menggunakan layout

fragment_detail. Fragment ini akan ditampilkan saat aksi navigasi dilakukan dari halaman utama.

12. Strings.xml

XML di atas adalah file strings.xml yang digunakan dalam proyek Android untuk menyimpan berbagai teks atau data dalam bentuk string, agar tidak ditulis langsung (hardcoded) di layout atau kode program. Hal ini sangat membantu dalam pengelolaan aplikasi, terutama untuk mendukung banyak bahasa (lokalisasi) atau saat ingin mengubah isi teks tanpa menyentuh file layout atau Java/Kotlin-nya. Pertama, terdapat beberapa string biasa, seperti app_name untuk nama aplikasi yaitu "DreamsTravel", serta btn_detail, btn_wiki, dan img_desc yang masing-masing berisi teks untuk tombol dan deskripsi gambar. Kemudian, ada juga beberapa string-array, yaitu array yang berisi kumpulan data teks. Misalnya, list_negara berisi nama-nama negara impian seperti Jepang, Swiss, hingga Prancis. Ini biasanya ditampilkan dalam bentuk daftar atau pilihan di aplikasi.

Ada juga array list_tahun yang memuat beberapa tahun dari 2025 sampai 2029, kemungkinan digunakan untuk memilih tahun impian untuk berangkat. Selanjutnya, data_gambar adalah array yang berisi referensi gambar dari folder drawable, digunakan untuk menampilkan gambar negara sesuai nama-nama di atas. Selain itu, ada list_alasan yang menjelaskan alasan kenapa pengguna ingin mengunjungi masing-masing negara—isinya seperti motivasi pribadi, misalnya karena budaya, musik, agama, atau ingin bertemu selebriti. Terakhir, ada list_link yang berisi tautan Wikipedia dari masing-masing negara, digunakan saat pengguna ingin tahu lebih detail dengan membuka informasi lewat web.

SOAL 2

Mengapa RecyclerView masih digunakan, padahal RecyclerView memiliki kode yang panjang dan bersifat boiler-plate, dibandingkan LazyColumn dengan kode yang lebih singkat?

A. Pembahasan

Alasan RecyclerView masih banyak digunakan walaupun kodenya lebih panjang dan terkesan boiler-plate adalah karena:

1. RecyclerView adalah bagian dari View system (lama) yang sudah stabil dan banyak digunakan di dunia industri. Banyak aplikasi besar yang dibangun sebelum Jetpack Compose muncul, dan mereka masih bergantung pada sistem View, termasuk RecyclerView.
2. Kompatibilitas dan dokumentasi luas. RecyclerView sudah ada sejak lama, jadi dokumentasinya lengkap, komunitasnya besar, dan banyak library serta tools yang mendukungnya. Kalau kamu butuh referensi, StackOverflow dan GitHub penuh dengan contoh.
3. RecyclerView sangat fleksibel dan bisa dikustomisasi. Meski kelihatannya ribet, RecyclerView bisa digunakan untuk berbagai jenis tampilan: daftar sederhana, grid, carousel, dan bahkan chat layout, hanya dengan mengubah layout manager atau adapter-nya.
4. Tidak semua proyek menggunakan Jetpack Compose. LazyColumn adalah bagian dari Compose, yang merupakan pendekatan UI yang relatif baru. Banyak perusahaan masih memakai sistem View karena belum semua tim siap untuk migrasi besar-besaran ke Compose.
5. Compose belum sepenuhnya matang untuk semua kasus. Meskipun Compose (dan LazyColumn) lebih singkat dan modern, dalam beberapa kasus performa atau dukungan komponennya masih kalah dengan View system, tergantung kompleksitas aplikasi.

TAUTAN GIT

Berikut adalah tautan untuk source code yang telah dibuat.

annacoded/Pemrograman-Mobile