LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 4

VIEW MODEL AND DEBUGGING

Oleh:

Siti Ratna Dwinta Sari NIM. 2310817120002

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
MEI 2024

LEMBAR PENGESAHAN
LAPORAN PRAKTIKUM PEMROGRAMAN I
MODUL 4

Laporan Praktikum Pemrograman Mobile Modul 4: ViewModel and Debugging 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. 198810272019032013

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

SOAL 1

Lanjutkan aplikasi Android berbasis XML dan Jetpack Compose yang sudah dibuat pada Modul 3 dengan menambahkan modifikasi sesuai ketentuan berikut:

a. Buatlah sebuah ViewModel untuk menyimpan dan mengelola data dari list item. Data tidak boleh disimpan langsung di dalam Fragment atau Activity.
b. Gunakan ViewModelFactory dalam pembuatan ViewModel
c. Gunakan StateFlow untuk mengelola event onClick dan data list item dari ViewModel
   Fragment
d. gunakan logging untuk event berikut:
   a. Log saat data item masuk ke dalam list
   b. Log saat tombol Detail dan tombol Explicit Intent ditekan
   c. Log data dari list yang dipilih ketika berpindah ke halaman Detail
e. Gunakan tool Debugger di Android Studio untuk melakukan debugging pada aplikasi. Cari setidaknya satu breakpoint yang relevan dengan aplikasi. Lalu, gunakan fitur Step Into, Step Over, dan Step Out. Setelah itu, jelaskan fungsi Debugger, cara menggunakan Debugger, serta fitur Step Into, Step Over, dan Step Out

A. Source Code

1. MainActivity.kt

Tabel 1. Source Code Jawaban Soal 1 MainActivity.kt

```
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. DetailFragment.kt

Tabel 2. Source Code Jawaban Soal 1 DetailFragment.kt

```
 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  }
```

3. HomeFragment.kt

Tabel 3. Source Code Jawaban Soal 1 HomeFragment.kt

```
1   package com.example.dreamstravel
2
3   import NegaraViewModel
4   import NegaraViewModelFactory
5   import android.content.Intent
6   import android.os.Bundle
7   import android.util.Log
8   import android.view.LayoutInflater
9   import android.view.View
10  import android.view.ViewGroup
11  import androidx.fragment.app.Fragment
12  import androidx.lifecycle.ViewModelProvider
13  import androidx.lifecycle.lifecycleScope
14  import androidx.recyclerview.widget.LinearLayoutManager
15  import com.example.dreamstravel.databinding.FragmentHomeBinding
16  import kotlinx.coroutines.launch
17  import androidx.core.net.toUri
18  import androidx.navigation.fragment.findNavController
19
20  class HomeFragment : Fragment() {
21      private var _binding: FragmentHomeBinding? = null
22      private val binding get() = _binding!!
23
24      private lateinit var viewModel: NegaraViewModel
25      private lateinit var adapter: NegaraAdapter
```

```
26
27      override fun onCreateView(inflater: LayoutInflater,
    container: ViewGroup?, savedInstanceState: Bundle?): View {
28          _binding = FragmentHomeBinding.inflate(inflater,
    container, false)
29          return binding.root
30      }
31
32      override fun onViewCreated(view: View,
    savedInstanceState: Bundle?) {
33          // Gunakan ViewModelFactory
34          val factory =
    NegaraViewModelFactory(requireActivity().application)
35          viewModel = ViewModelProvider(this,
    factory)[NegaraViewModel::class.java]
36
37          // Setup Adapter tanpa navController, dengan dua
    callback
38          adapter = NegaraAdapter(
39              listNegara = emptyList(),
40              onDetailClick = { negara ->
41                  Log.d("HomeFragment", "Tombol Detail
    ditekan: ${negara.nama}")
42                  viewModel.onItemClicked(negara)
43
44                  val bundle = Bundle().apply {
45                      putInt("gambarResId",
    negara.gambarResId)
46                      putString("namaNegara", negara.nama)
47                      putString("alasan", negara.alasan)
48                  }
49
50                  Log.d("HomeFragment", "Navigasi ke detail
    dengan: ${negara.nama}")
51
    findNavController().navigate(R.id.detailFragment, bundle)
52              },
53              onWikiClick = { negara ->
54                  Log.d("HomeFragment", "Tombol Wiki ditekan:
    ${negara.nama}")
55                  val intent = Intent(Intent.ACTION_VIEW,
    negara.wikiUrl.toUri())
56                  startActivity(intent)
57              }
58          )
59
60
```

```
61          binding.rvNegara.layoutManager                          =
    LinearLayoutManager(requireContext())
62          binding.rvNegara.adapter = adapter
63
64          // Observe StateFlow dari ViewModel
65          lifecycleScope.launch {
66              viewModel.listNegara.collect { negaraList ->
67                  Log.d("HomeFragment",   "Data   listNegara
    masuk (${negaraList.size} item)")
68                  adapter.updateData(negaraList)
69              }
70          }
71
72          // Panggil loadNegara agar data dimuat
73          viewModel.loadNegara()
74      }
75
76      override fun onDestroyView() {
77          super.onDestroyView()
78          _binding = null
79      }
80  }
```

4. NegaraAdapter.kt

Tabel 4. Source Code Jawaban Soal 1 NegaraAdapter.kt

```
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.recyclerview.widget.RecyclerView
8   import com.example.dreamstravel.databinding.ItemNegaraBinding
9
10  class NegaraAdapter(
11      private var listNegara: List<NegaraModel>,
12      private val onDetailClick: (NegaraModel) -> Unit,
13      private val onWikiClick: (NegaraModel) -> Unit
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
    holder.binding.imgItemPhoto.setImageResource(negara.gambarResId)
27
28          holder.binding.btnWiki.setOnClickListener {
29              onWikiClick(negara)
30          }
31
32          holder.binding.buttonDetail.setOnClickListener {
33              onDetailClick(negara)
34          }
35      }
36
37      override fun getItemCount() = listNegara.size
38
39      fun updateData(newList: List<NegaraModel>) {
40          listNegara = newList
41          notifyDataSetChanged()
42      }
43  }
```

5. NegaraModel.kt

Tabel 5. Source Code Jawaban Soal 1 NegaraModel.kt

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

6. NegaraViewModel.kt

Tabel 6. Source Code Jawaban Soal 1 NegaraViewModel.kt

```
1  import android.app.Application
2  import android.util.Log
3  import androidx.lifecycle.AndroidViewModel
4  import androidx.lifecycle.viewModelScope
5  import kotlinx.coroutines.flow.MutableStateFlow
6  import kotlinx.coroutines.flow.StateFlow
7  import kotlinx.coroutines.launch
8  import com.example.dreamstravel.NegaraModel
9  import com.example.dreamstravel.R
10
11 class NegaraViewModel(application: Application) :
   AndroidViewModel(application) {
12
13     private val context =
   getApplication<Application>().applicationContext
14
15     private val _listNegara =
   MutableStateFlow<List<NegaraModel>>(emptyList())
16     val listNegara: StateFlow<List<NegaraModel>> get() =
   _listNegara
17
18     private val _selectedNegara =
   MutableStateFlow<NegaraModel?>(null)
19     val selectedNegara: StateFlow<NegaraModel?> get() =
   _selectedNegara
20
21     fun loadNegara() {
22         viewModelScope.launch {
23             val namaNegara =
   context.resources.getStringArray(R.array.list_negara)
24             val tahun =
   context.resources.getStringArray(R.array.list_tahun)
25             val alasan =
   context.resources.getStringArray(R.array.list_alasan)
26             val link =
   context.resources.getStringArray(R.array.list_link)
27             val gambar = listOf(
28                 R.drawable.japan,
29                 R.drawable.swiss,
30                 R.drawable.korsel,
31                 R.drawable.arabsaudi,
32                 R.drawable.france
33             )
34
```

```
35              val negaraList = namaNegara.indices.map { i ->
36                  NegaraModel(
37                      namaNegara[i],
38                      tahun[i],
39                      alasan[i],
40                      gambar[i],
41                      link[i]
42                  )
43              }
44
45              _listNegara.value = negaraList
46              Log.d("NegaraViewModel", "List negara dimuat:
    ${negaraList.size} item")
47          }
48      }
49
50      fun onItemClicked(negara: NegaraModel) {
51          _selectedNegara.value = negara
52          Log.d("NegaraViewModel",      "Negara      dipilih:
    ${negara.nama}")
53      }
54  }
```

7. ViewModelFactory.kt

Tabel 7. Source Code Jawaban Soal 1 ViewModelFactory

```
1   import android.app.Application
2   import androidx.lifecycle.ViewModel
3   import androidx.lifecycle.ViewModelProvider
4
5   class   NegaraViewModelFactory(private   val   application:
    Application) : ViewModelProvider.Factory {
6       override   fun   <T   :   ViewModel>   create(modelClass:
    Class<T>): T {
7           if
8   (modelClass.isAssignableFrom(NegaraViewModel::class.java))
    {
9               return NegaraViewModel(application) as T
10          }
11          throw  IllegalArgumentException("Unknown  ViewModel
    class")
12      }
13  }
```

8. Activity_main.xml

Tabel 8. Source Code Jawaban Soal 1 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      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      android:orientation="vertical">
8
9      <TextView
10         android:layout_width="wrap_content"
11         android:layout_height="wrap_content"
12         android:text="Selamat datang di DreamsTravel!"
13         android:textSize="18sp"
14         android:layout_gravity="center_horizontal"
15         android:layout_margin="16dp"/>
16
17     <androidx.fragment.app.FragmentContainerView
18         android:id="@+id/nav_host_fragment"
19
20 android:name="androidx.navigation.fragment.NavHostFragment"
           android:layout_width="match_parent"
21         android:layout_height="match_parent"
22         app:defaultNavHost="true"
23         app:navGraph="@navigation/nav_graph" />
24 </LinearLayout>
```

9. Fragment_detail.xml

Tabel 9. Source Code Jawaban Soal 1 Fragment_detail.xml

```
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      xmlns:tools="http://schemas.android.com/tools"
5      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="40sp"

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="match_parent"
32          android:layout_height="wrap_content"
33          android:justificationMode="inter_word"
34          android:textAlignment="viewStart"
35          android:layout_margin="16dp"
36          android:textSize="16sp"
37          android:layout_marginTop="16dp"
38
    app:layout_constraintTop_toBottomOf="@id/tvCountryName"
39          app:layout_constraintStart_toStartOf="parent"
40          app:layout_constraintEnd_toEndOf="parent"
41          android:text="Reason"/>
42  </androidx.constraintlayout.widget.ConstraintLayout>
```

10. Fragment_home.xml

Tabel 10. Source Code Jawaban Soal 1 Fragment_home.xml

```
1   <?xml version="1.0" encoding="utf-8"?>
    <androidx.constraintlayout.widget.ConstraintLayout
2   xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:app="http://schemas.android.com/apk/res-auto"
3       android:layout_width="match_parent"
4       android:layout_height="match_parent"
5       xmlns:tools="http://schemas.android.com/tools">
6
7       <androidx.recyclerview.widget.RecyclerView
```

```
        android:id="@+id/rvNegara"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:clipToPadding="false"
        android:padding="8dp"
        android:contentDescription="cute"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        tools:listitem="@layout/item_negara" />
</androidx.constraintlayout.widget.ConstraintLayout>

```

11. Item_negara.xml

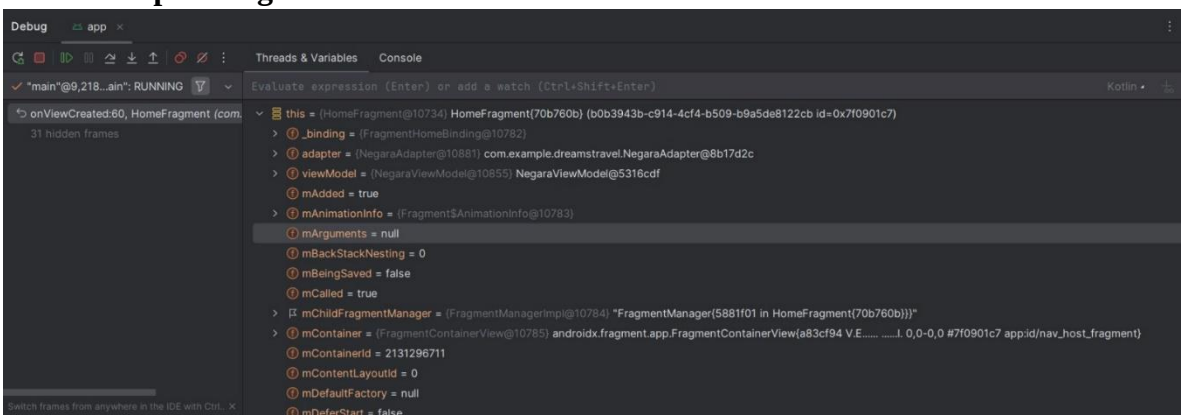
Tabel 11. Source Code Jawaban Soal 1 Item_negara.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.cardview.widget.CardView
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:card_view="http://schemas.android.com/apk/res-
auto"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="@dimen/card_margin"
    app:cardCornerRadius="@dimen/card_radius"
    app:cardElevation="4dp">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:padding="@dimen/card_padding">


<com.google.android.material.imageview.ShapeableImageView
            android:id="@+id/img_item_photo"
            android:layout_width="80dp"
            android:layout_height="100dp"
            android:scaleType="centerCrop"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"

app:layout_constraintBottom_toBottomOf="parent"

```

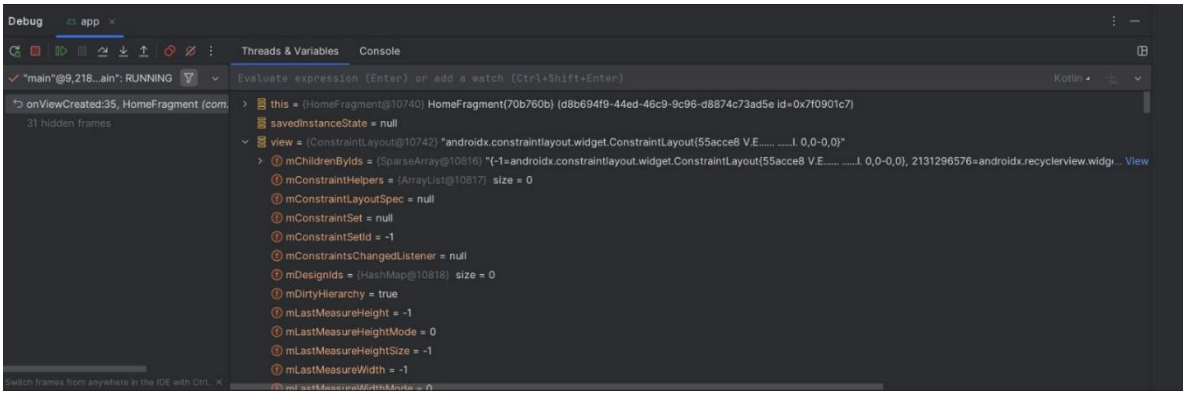
```
            android:contentDescription="@string/img_desc"
/>

        <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
        </androidx.constraintlayout.widget.ConstraintLayout>
    </androidx.cardview.widget.CardView>
```

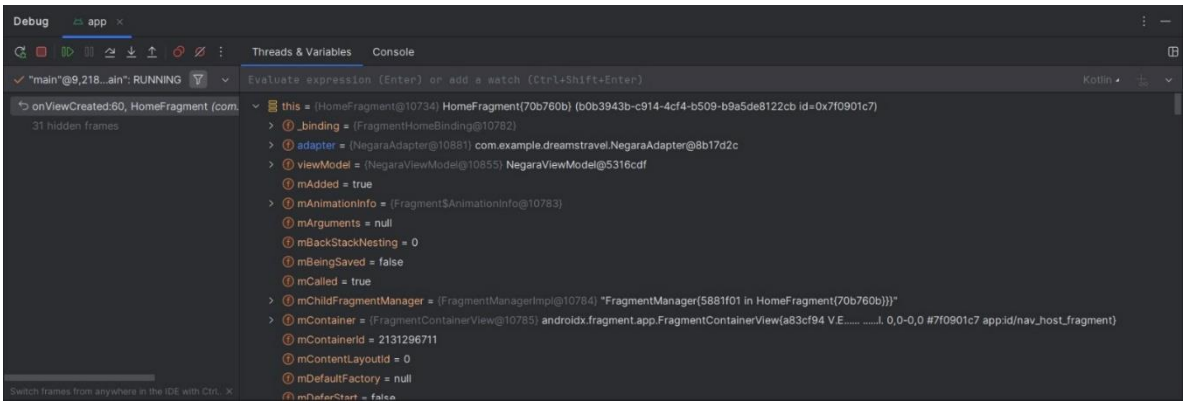
B. Output Program



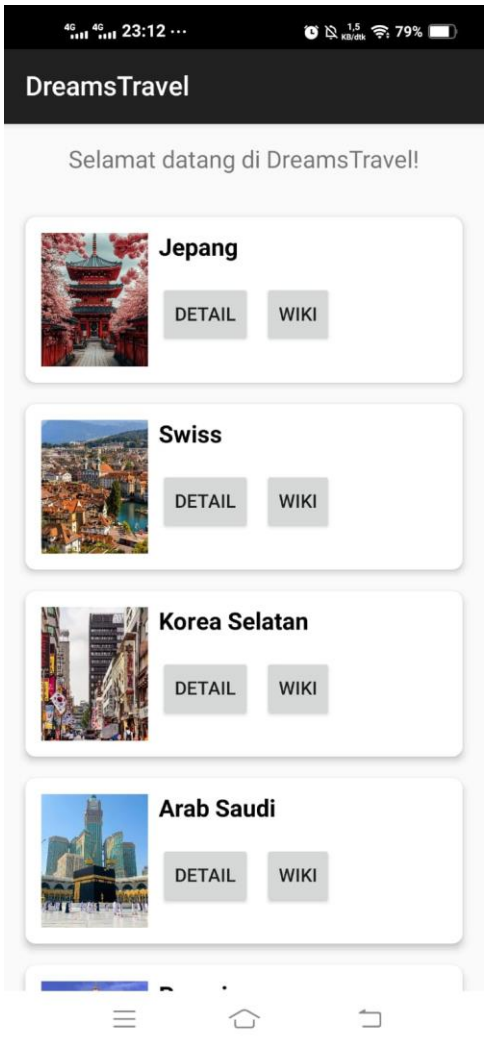
Gambar 1. Debugger Step Into



Gambar 2. Debugger Step Over



Gambar 3. Debugger Step Out



Gambar 4. Tampilan Home

Gambar 5. Tampilan Detail

C. Pembahasan

1. MainActivity.kt:

Kode di atas adalah bagian dari aplikasi Android yang ditulis menggunakan Kotlin dan Jetpack Compose. Kelas `MainActivity` merupakan aktivitas utama yang dijalankan saat aplikasi dibuka. Di dalam `onCreate()`, `ActivityMainBinding` digunakan untuk menghubungkan file XML `activity_main.xml` ke kode Kotlin, sehingga elemen UI bisa diakses langsung. Kemudian, sistem navigasi menggunakan NavHostFragment diinisialisasi agar aplikasi bisa berpindah-pindah antar fragment.

2. DetailFragment.kt

Kode di atas adalah kelas `DetailFragment`, yaitu salah satu halaman (fragment) dalam aplikasi Android yang menampilkan detail suatu negara. Fragment ini

menggunakan view binding (`FragmentDetailBinding`) untuk menghubungkan elemen-elemen di layout XML dengan kode Kotlin. Saat fragment dibuat (`onCreateView`), layout-nya di-inflate dan disimpan ke dalam variabel binding. Di `onViewCreated`, fragment mengambil data seperti nama negara, alasan, dan gambar dari argument (data yang dikirim dari fragment sebelumnya). Data ini kemudian ditampilkan ke tampilan UI. Terakhir, di `onDestroyView`, binding di-set ke null untuk menghindari memory leak.

3. **HomeFragment.kt**

   Kode di atas merupakan implementasi dari `HomeFragment,` yaitu halaman utama yang menampilkan daftar negara yang ingin dikunjungi. Fragment ini menggunakan `FragmentHomeBinding` untuk menghubungkan layout XML ke kode Kotlin, serta menggunakan `ViewModel` untuk mengelola data negara. Di dalam `onViewCreated`, `NegaraViewModel` diinisialisasi menggunakan `ViewModelFactory`, lalu `NegaraAdapter` digunakan untuk menampilkan data dalam bentuk daftar (RecyclerView). Adapter ini punya dua aksi: ketika tombol Detail ditekan, aplikasi akan menavigasi ke `DetailFragment` sambil mengirim data negara melalui `Bundle`; sedangkan jika tombol Wiki ditekan, akan membuka halaman Wikipedia negara tersebut di browser. Data negara dimuat dan diamati menggunakan `StateFlow`, dan jika ada perubahan, RecyclerView akan diperbarui. Terakhir, binding dibersihkan di `onDestroyView` untuk menghindari memory leak.

4. **NegaraAdapter.kt**

   Kode di atas adalah kelas `NegaraAdapter` yang digunakan untuk menampilkan daftar negara dalam bentuk RecyclerView. Adapter ini menerima data berupa list dari `NegaraModel`, serta dua aksi saat tombol ditekan: `onDetailClick` dan `onWikiClick`. Di dalamnya terdapat `ViewHolder` yang menghubungkan layout `item_negara.xml` dengan data negara. Di `onBindViewHolder`, setiap item akan diisi dengan nama dan gambar negara, serta dua tombol: satu untuk membuka link Wikipedia (`btnWiki`), dan satu lagi untuk melihat detail negara

(buttonDetail). Fungsi updateData digunakan untuk memperbarui daftar negara saat datanya berubah.

5. **NegaraModel.kt**

   Kode di atas adalah data class bernama NegaraModel yang berfungsi untuk menyimpan data dari sebuah negara. Setiap objek NegaraModel berisi informasi seperti nama negara (nama), tahun impian untuk dikunjungi (tahun), alasan ingin mengunjungi negara tersebut (alasan), ID gambar dari sumber daya aplikasi (gambarResId), dan link Wikipedia negara tersebut (wikiUrl). Kelas ini biasanya digunakan sebagai model data yang akan ditampilkan di tampilan aplikasi, seperti dalam RecyclerView.

6. **NegaraViewModel.kt**

   Kode di atas adalah kelas NegaraViewModel, yang berfungsi sebagai penghubung antara data dan tampilan (UI) dalam arsitektur MVVM (Model-View-ViewModel). Kelas ini mewarisi dari AndroidViewModel karena membutuhkan akses ke konteks aplikasi. Di dalamnya ada dua StateFlow: satu untuk menyimpan daftar negara (listNegara), dan satu lagi untuk menyimpan data negara yang dipilih (selectedNegara). Fungsi loadNegara() dipanggil untuk memuat data dari resource (seperti array nama, tahun, alasan, gambar, dan link) lalu menggabungkannya menjadi list NegaraModel. List ini kemudian diberikan ke UI melalui StateFlow. Ada juga fungsi onItemClicked() yang menyimpan negara yang diklik pengguna.

7. **ViewModelFactory.kt**

   Kode di atas adalah kelas NegaraViewModel yang digunakan untuk mengelola data negara dalam aplikasi Android dengan arsitektur MVVM. Kelas ini mewarisi AndroidViewModel agar bisa mengakses resource aplikasi seperti array dari strings.xml. Di dalamnya terdapat dua StateFlow: yang pertama _listNegara menyimpan daftar semua negara, dan yang kedua _selectedNegara menyimpan negara yang dipilih. Fungsi loadNegara() digunakan untuk mengambil data nama, tahun, alasan, link Wikipedia, dan gambar dari resource, lalu digabungkan menjadi objek NegaraModel dan dimasukkan ke

StateFlow. Fungsi `onItemClicked()` dipakai untuk menyimpan data negara yang dipilih pengguna.

8. **Activity_main.xml**

   Kode XML di atas merupakan layout utama dari aplikasi yang menggunakan `LinearLayout` sebagai wadah vertikal. Di dalamnya terdapat `TextView` yang menampilkan sambutan "Selamat datang di DreamsTravel!" di tengah atas layar dengan ukuran teks 18sp dan margin 16dp. Di bawahnya, ada `FragmentContainerView` yang berfungsi sebagai tempat untuk menampilkan fragment-fragment lain berdasarkan navigasi. Komponen ini menggunakan `NavHostFragment` dari Jetpack Navigation dan diarahkan ke file `nav_graph.xml` yang berisi alur navigasi antar fragment

9. **Fragment_detail.xml**

   Kode XML di atas adalah layout untuk tampilan detail negara pada `DetailFragment`. Layout ini menggunakan `ConstraintLayout`, yang memungkinkan pengaturan posisi elemen UI secara fleksibel. Di dalamnya ada tiga komponen utama: pertama, `ImageView` dengan ID `imgCountry` yang akan menampilkan gambar negara; kedua, `TextView` dengan ID `tvCountryName` yang menampilkan nama negara dengan ukuran teks besar (40sp); dan ketiga, `TextView` dengan ID `tvCountryReason` yang menampilkan alasan pengguna ingin mengunjungi negara tersebut, dengan teks rata kiri dan margin agar terlihat rapi. Semua elemen diposisikan menggunakan constraint, agar tampilan tetap responsif di berbagai ukuran layar.

10. **Fragment_home.xml**

    Kode XML di atas adalah layout untuk `HomeFragment`, yang menggunakan `ConstraintLayout` sebagai wadah utama. Di dalamnya terdapat sebuah `RecyclerView` dengan ID `rvNegara`, yang digunakan untuk menampilkan daftar negara secara scrollable. Ukurannya dibuat memenuhi layar (`0dp` untuk width dan height) dan diatur posisinya agar menempel ke semua sisi parent menggunakan constraint. Properti `clipToPadding` diatur `false` agar isi tidak terpotong oleh padding, dan `tools:listitem` digunakan saat preview di Android Studio untuk menunjukkan bahwa tiap item di dalam RecyclerView akan menggunakan layout

`item_negara.xml`. Singkatnya, layout ini menampilkan daftar negara dengan desain responsif dan teratur

11. Item_negara.xml

Kode XML ini adalah layout untuk satu item dalam daftar negara (yang ditampilkan di `RecyclerView`). Layout ini menggunakan `CardView` supaya tampilannya seperti kartu dengan sudut melengkung dan bayangan (elevation). Di dalam `CardView` terdapat `ConstraintLayout` yang mengatur posisi elemen-elemen seperti gambar, nama negara, dan dua tombol.

Gambar negara ditampilkan lewat `ShapeableImageView`, lalu ada `TextView` untuk menampilkan nama negara. Di bawah nama negara ada dua `Button`: satu untuk membuka detail negara (`button_detail`), dan satu lagi untuk membuka link Wikipedia (`btn_wiki`). Semua elemen diposisikan dengan constraint agar tetap rapi di berbagai ukuran layar.

Penjelasan Debugger, fitur Step Into, Steo Over, dan Step Out

1. Definisi Debugger

   Debugger adalah alat bantu dalam pemrograman yang digunakan untuk mencari dan memperbaiki bug (kesalahan) di dalam kode. Dengan debugger, kita bisa menjalankan program langkah demi langkah, melihat isi variabel, dan mengetahui bagian mana dari kode yang menyebabkan error.
2. Cara Menggunakan Debugger
   - Set Breakpoint

     → Klik di samping baris kode (di IDE seperti Android Studio atau IntelliJ) untuk menandai titik berhenti (breakpoint).

     → Program akan berhenti sementara di baris itu saat dijalankan dalam mode debug.
   - Jalankan Program dalam Mode Debug

     → Klik tombol debug (ikon bug atau debug play button).

   - Gunakan fitur kontrol untuk melacak eksekusi program.

3. Fitur

1. Step Into (F7)

Masuk ke dalam fungsi yang sedang dipanggil.

Contoh: Jika ada myFunction(), debugger akan masuk ke dalam isi dari myFunction().

2. Step Over (F8)

Melewati fungsi tanpa masuk ke dalamnya.

Contoh: myFunction() tetap dijalankan, tapi debugger tidak masuk ke dalamnya — langsung ke baris berikutnya.

3. Step Out (Shift+F8)

Keluar dari fungsi saat ini dan kembali ke pemanggilnya.

Misal kamu sudah di dalam myFunction(), maka ini akan melompat keluar ke baris setelah myFunction() dipanggil.

SOAL 2

Jelaskan Application class dalam arsitektur aplikasi Android dan fungsinya

A. Pembahasan

1. Definisi Application class

Dalam arsitektur aplikasi Android, kelas Application adalah salah satu komponen utama yang digunakan untuk menyimpan dan mengelola status global aplikasi. Ini adalah titik awal kehidupan aplikasi Android, dan biasanya digunakan untuk inisialisasi yang harus dilakukan sekali seumur hidup aplikasi.

Application adalah kelas dasar dari sistem Android yang berisi semua komponen aplikasi (seperti Activity, Service, BroadcastReceiver, dan ContentProvider). Android membuat instance dari kelas ini sebelum membuat komponen lainnya saat aplikasi dijalankan.

2. Fungsi utama dari Application class

1. Inisialisasi Global

- Menjalankan kode yang harus diinisialisasi hanya sekali, seperti pustaka pihak ketiga (misalnya Retrofit, Dagger, Firebase).
- Contoh: konfigurasi logging, analytics, dependency injection.

2. Penyimpanan Data Global

- Menyimpan informasi atau state yang dibutuhkan di seluruh bagian aplikasi.
- Misalnya: token autentikasi, data pengguna sementara.

3. Mengakses Konteks Aplikasi

- Memberikan akses ke konteks aplikasi (ApplicationContext), yang bisa digunakan oleh berbagai komponen.

4. Lifecycle-aware Application

- Bisa digunakan bersama ProcessLifecycleOwner untuk mengetahui lifecycle aplikasi secara global.

3. Cara menggunakan Application class

1. Buat kelas yang extends Application

```
class MyApp : Application() {
    override fun onCreate() {
        super.onCreate()
```

```
            // Inisialisasi global
            Log.d("MyApp", "Application started")
        }
}
```

2. Deklarasikan di AndroidManifest.xml:

```
    <application
        android:name=".MyApp"
        ... >
        ...
</application>
```

B. Tautan Git

Berikut adalah tautan untuk source code yang telah dibuat.

https://github.com/annacoded/Pemrograman-Mobile