LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 4

VIEWMODEL AND DEBUGGING

Oleh:

Nur Hikmah **NIM. 2310817120010**

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
MEI 2025

LEMBAR PENGESAHAN
LAPORAN PRAKTIKUM PEMROGRAMAN MOBILE
MODUL 4

Laporan Praktikum Pemrograman Mobile Modul 4: ViewModel and Debugging ini disusun sebagai syarat lulus mata kuliah Praktikum Pemrograman Mobile. Laporan Prakitkum ini dikerjakan oleh:

Nama Praktikan : Nur Hikmah
NIM : 2310817120010

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

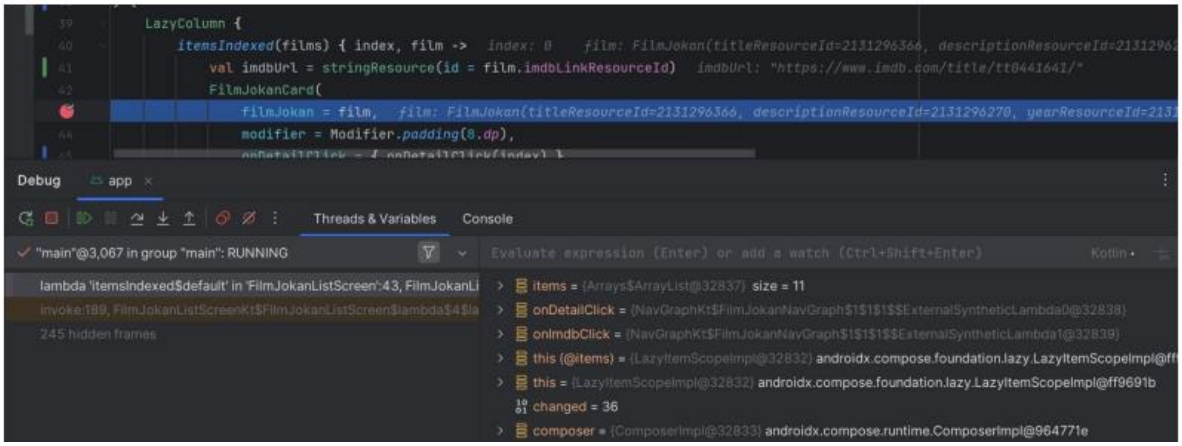
DAFTAR TABEL

SOAL 1

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 ke 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

Aplikasi harus dapat mempertahankan fitur-fitur yang sudah dibuat pada modul sebelumnya. Berikut adalah contoh debugging dalam Android Studio.



Gambar 1. Contoh Penggunaan Debugger

A. Source Code

1. HomeFragment.kt

```
1   package com.example.modul4
2
3   import android.os.Bundle
4   import android.os.Parcelable
5   import androidx.fragment.app.Fragment
6   import android.view.LayoutInflater
7   import android.view.View
8   import android.view.ViewGroup
9   import androidx.lifecycle.ViewModelProvider
10  import androidx.lifecycle.lifecycleScope
11  import androidx.navigation.fragment.findNavController
12  import androidx.recyclerview.widget.LinearLayoutManager
13  import com.example.modul4.databinding.FragmentHomeBinding
14
15  class HomeFragment : Fragment() {
16      private var recyclerViewState: Parcelable? = null
17      private lateinit var viewModel: TasViewModel
18      private var _binding: FragmentHomeBinding? = null
19      private val binding get() = _binding!!
20
21      override fun onCreateView(
22          inflater: LayoutInflater, container: ViewGroup?,
23          savedInstanceState: Bundle?
24      ): View {
25          _binding = FragmentHomeBinding.inflate(inflater,
    container, false)
26          return binding.root
27      }
28
29     override fun onViewCreated(view: View,
    savedInstanceState: Bundle?) {
30          val factory = TasViewModelFactory()
31          viewModel = ViewModelProvider(requireActivity(),
    factory)[TasViewModel::class.java]
32          viewModel.loadData(generateTasList())
33          lifecycleScope.launchWhenStarted {
34              viewModel.tasList.collect { tasList ->
35                  val adapter = AdapterTas(tasList) { item ->
36                      viewModel.selectTas(item)
37                  }
38                  binding.recyclerView.layoutManager =
    LinearLayoutManager(requireContext())
39                  binding.recyclerView.adapter = adapter
40              }
41          }
42          lifecycleScope.launchWhenStarted {
43              viewModel.navigateToDetail.collect {
44             if (findNavController().currentDestination?.id
    == R.id.homeFragment)
```

```
45  findNavController().navigate(R.id.action_homeFragment_to_de
    tailFragment)
46                  }
47              }
48          }
49      }
50
51      override fun onDestroyView() {
52          super.onDestroyView()
53          _binding = null
54      }
55
56      private fun generateTasList(): List<ItemTas> {
57          val context = requireContext()
58          val names =
    context.resources.getStringArray(R.array.data_name)
59          val harga =
    context.resources.getStringArray(R.array.data_harga)
60          val deskripsi =
    context.resources.getStringArray(R.array.data_deskripsi)
61          val links =
    context.resources.getStringArray(R.array.data_link)
62
63          val imageIds = listOf(
64              R.drawable.marsya_bag,
65              R.drawable.lyn_bag,
66              R.drawable.reigny_bag,
67              R.drawable.madeline_bag,
68              R.drawable.tote_bag,
69              R.drawable.mini_cendy_bag,
70              R.drawable.amora_bag,
71              R.drawable.gradien_bag,
72              R.drawable.lysmare_bag,
73              R.drawable.delphine_bag
74          )
75
76          val list = mutableListOf<ItemTas>()
77          for (i in names.indices) {
78              list.add(
79                  ItemTas(
80                      nama = names[i],
81                      harga = harga[i],
82                      deskripsi = deskripsi[i],
83                      gambar = imageIds[i],
84                      link = links[i]
85                  )
86              )
87          }
88          return list
89      }
90  }
```

Tabel 1. Source Code Jawaban Soal 1

2. DetailFragment.kt

```
1   package com.example.modul4
2
3   import android.os.Bundle
4   import android.util.Log
5   import android.view.LayoutInflater
6   import android.view.View
7   import android.view.ViewGroup
8   import androidx.fragment.app.Fragment
9   import androidx.lifecycle.ViewModelProvider
10  import androidx.lifecycle.lifecycleScope
11  import com.example.modul4.databinding.FragmentDetailBinding
12  import kotlinx.coroutines.launch
13
14  class DetailFragment : Fragment() {
15      private lateinit var viewModel: TasViewModel
16      private var _binding: FragmentDetailBinding? = null
17      private val binding get() = _binding!!
18
19      override fun onCreateView(
20          inflater: LayoutInflater, container: ViewGroup?,
21          savedInstanceState: Bundle?
22      ): View {
23          _binding = FragmentDetailBinding.inflate(inflater,
    container, false)
24          return binding.root
25      }
26
27      override fun onViewCreated(view: View,
    savedInstanceState: Bundle?) {
28          super.onViewCreated(view, savedInstanceState)
29          viewModel =
    ViewModelProvider(requireActivity())[TasViewModel::class.ja
    va]
30          lifecycleScope.launch {
31              viewModel.selectedTas.collect { tas ->
32                  Log.d("DetailFragment", "Tas dari
    ViewModel: $tas")
33                  if (tas != null) {
34                      binding.tvName.text = tas.nama
35                      binding.tvHarga.text =
    getString(R.string.label_harga) + " " + tas.harga
36                      binding.tvDeskripsi.text =
    tas.deskripsi
37
    binding.imgItemPhoto.setImageResource(tas.gambar)
38                  } else {
39                      Log.d("DetailFragment", "Tas masih null
    dari ViewModel")
40                  }
```

```
41              }
42          }
43      }
44
45      override fun onDestroyView() {
46          super.onDestroyView()
47          viewModel.clearSelectedTas()
48          _binding = null
49      }
50  }
```

Tabel 2. Source Code Jawaban Soal 1

3. ItemTas.kt

```
1   package com.example.modul4
2
3   import android.os.Parcelable
4   import kotlinx.parcelize.Parcelize
5
6   @Parcelize
7   data class ItemTas(
8       val nama: String,
9       val gambar: Int,
10      val harga: String,
11      val deskripsi: String,
12      val link: String
13  ) : Parcelable
```

Tabel 3. Source Code Jawaban Soal 1

4. AdapterTas.kt

```
1   package com.example.modul4
2
3   import android.content.Intent
4   import android.net.Uri
5   import android.os.Bundle
6   import android.util.Log
7   import android.view.LayoutInflater
8   import android.view.ViewGroup
9   import androidx.navigation.NavController
10  import androidx.recyclerview.widget.RecyclerView
11  import com.example.modul4.databinding.ItemTasBinding
12
13  class AdapterTas(
14      private val listTas: List<ItemTas>,
15      private val onItemClick: (ItemTas) -> Unit
16  ) : RecyclerView.Adapter<AdapterTas.ListViewHolder>() {
17
18     inner class ListViewHolder(val binding: ItemTasBinding) :
19          RecyclerView.ViewHolder(binding.root) {
```

```
20
21          fun bind(tas: ItemTas) {
22              binding.tvItemName.text = tas.nama
23              binding.tvItemPrice.text = tas.harga
24             binding.imgItemPhoto.setImageResource(tas.gambar)
25              binding.buttonDetail.setOnClickListener {
26                  Log.d("AdapterTas", "Tombol detail diklik")
27                  onItemClick(tas)
28              }
29              binding.buttonShopee.setOnClickListener {
30                  Log.d("AdapterTas", "Tombol Shopee diklik:
    ${tas.link}")
31                  val intent = Intent(Intent.ACTION_VIEW,
    Uri.parse(tas.link))
32                  binding.root.context.startActivity(intent)
33              }
34          }
35      }
36
37      override fun onCreateViewHolder(parent: ViewGroup,
    viewType: Int): ListViewHolder {
38          val binding =
    ItemTasBinding.inflate(LayoutInflater.from(parent.context),
    parent, false)
39          return ListViewHolder(binding)
40      }
41
42      override fun onBindViewHolder(holder: ListViewHolder,
    position: Int) {
43          holder.bind(listTas[position])
44      }
45
46      override fun getItemCount(): Int = listTas.size
47  }
```

Tabel 4. Source Code Jawaban Soal 1

5. MainActivity.kt

```
1   package com.example.modul4
2
3   import android.os.Bundle
4   import androidx.appcompat.app.AppCompatActivity
5   import androidx.navigation.findNavController
6   import androidx.navigation.fragment.NavHostFragment
7   import com.example.modul4.databinding.ActivityMainBinding
8
9   class MainActivity : AppCompatActivity() {
10      private lateinit var binding: ActivityMainBinding
11      override fun onCreate(savedInstanceState: Bundle?) {
12          super.onCreate(savedInstanceState)
13          binding = ActivityMainBinding.inflate(layoutInflater)
```

```
14          setContentView(binding.root)
15      }
16      override fun onBackPressed() {
17          val navHostFragment =
    supportFragmentManager.findFragmentById(R.id.nav_host_fragment)
    as NavHostFragment
18          val navController = navHostFragment.navController
19          if (!navController.popBackStack()) {
20              super.onBackPressed()
21          }
22      }
23  }
```

Tabel 5. Source Code Jawaban Soal 1

6. TasViewModel.kt

```
1   package com.example.modul4
2
3   import android.util.Log
4   import androidx.lifecycle.ViewModel
5   import androidx.lifecycle.viewModelScope
6   import kotlinx.coroutines.flow.*
7   import kotlinx.coroutines.launch
8
9   class TasViewModel : ViewModel() {
10
11      private val _tasList =
    MutableStateFlow<List<ItemTas>>(emptyList())
12      val tasList: StateFlow<List<ItemTas>> = _tasList
13
14      private val _selectedTas =
    MutableStateFlow<ItemTas?>(null)
15      val selectedTas: StateFlow<ItemTas?> = _selectedTas
16
17      private val _navigateToDetail =
    MutableSharedFlow<Unit>()
18      val navigateToDetail: SharedFlow<Unit> =
    _navigateToDetail
19
20      fun selectTas(item: ItemTas?) {
21          _selectedTas.value = item
22          Log.d("TasViewModel", "Item dipilih: ${item?.nama}")
23          if (item != null) {
24              viewModelScope.launch {
25                  _navigateToDetail.emit(Unit)
26              }
27          }
28      }
29
30      fun loadData(data: List<ItemTas>) {
31          _tasList.value = data
```

```
32          Log.d("TasViewModel", "Data item dimuat:
    ${data.size} item")
33      }
34
35      fun clearSelectedTas() {
36          _selectedTas.value = null
37      }
38  }
```

Tabel 6. Source Code Jawaban Soal 1

7. TasViewModelFactory.kt

```
1   package com.example.modul4
2
3   import androidx.lifecycle.ViewModel
4   import androidx.lifecycle.ViewModelProvider
5
6   class TasViewModelFactory : ViewModelProvider.Factory {
7       override fun <T : ViewModel> create(modelClass:
    Class<T>): T {
8   if (modelClass.isAssignableFrom(TasViewModel::class.java)) {
9               return TasViewModel() as T
10          }
11          throw IllegalArgumentException("Unknown ViewModel
    class: ${modelClass.name}")
12      }
13  }
```

Tabel 7. Source Code Jawaban Soal 1

8. fragment_home.xml

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
3       xmlns:tools="http://schemas.android.com/tools"
4       android:id="@+id/homeFragment"
5       android:layout_width="match_parent"
6       android:layout_height="match_parent"
7       android:orientation="vertical"
8       tools:context=".HomeFragment">
9
10      <TextView
11          android:layout_width="match_parent"
12          android:layout_height="wrap_content"
13          android:layout_marginTop="0dp"
14          android:background="#D1B3FF"
15          android:padding="15dp"
16          android:text="ShopBag - Your Bag Your Style"
17          android:textStyle="bold"
18          android:textColor="@color/black"
```

```
        android:textSize="25sp" />

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:padding="8dp" />

</LinearLayout>
```

Tabel 8. Source Code Jawaban Soal 1

9. item_tas.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.cardview.widget.CardView
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:id="@+id/card_view"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:layout_marginStart="8dp"
    android:layout_marginTop="4dp"
    android:layout_marginEnd="8dp"
    android:layout_marginBottom="4dp"
    app:cardCornerRadius="12dp"
    app:cardElevation="10dp">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="#F6F6F6"
        android:padding="20dp">

        <ImageView
            android:id="@+id/img_item_photo"
            android:layout_width="135dp"
            android:layout_height="140dp"
            android:scaleType="centerCrop"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintBottom_toBottomOf="parent"
            android:contentDescription="@string/app_name"
            tools:src="@drawable/marsya_bag" />

        <TextView
            android:id="@+id/tv_item_name"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:text="Tas Kulit Elegan"
```

```
38              android:textColor="@color/black"
39              android:textStyle="bold"
40              android:textSize="20sp"
41              app:layout_constraintTop_toTopOf="parent"
42        app:layout_constraintStart_toEndOf="@id/img_item_photo"
43              app:layout_constraintEnd_toEndOf="parent"
44              app:layout_constraintHorizontal_bias="0.0"
45              android:layout_marginStart="8dp"
46              tools:ignore="MissingConstraints" />
47
48          <TextView
49              android:id="@+id/tv_item_price"
50              android:layout_width="0dp"
51              android:layout_height="wrap_content"
52              android:text="Rp 350.000"
53              android:textSize="16sp"
54              android:textColor="@color/black"
55         app:layout_constraintTop_toBottomOf="@id/tv_item_name"
56        app:layout_constraintStart_toStartOf="@id/tv_item_name"
57              app:layout_constraintEnd_toEndOf="parent"
58              android:layout_marginTop="4dp" />
59
60          <Button
61              android:id="@+id/button_shopee"
62              android:layout_width="wrap_content"
63              android:layout_height="wrap_content"
64              android:layout_marginStart="8dp"
65              android:layout_marginTop="40dp"
66              android:text="@string/btn_shopee"
67        app:layout_constraintStart_toStartOf="@id/tv_item_name"
68      app:layout_constraintTop_toBottomOf="@id/tv_item_price"/>
69
70          <Button
71              android:id="@+id/button_detail"
72              android:layout_width="wrap_content"
73              android:layout_height="wrap_content"
74              android:layout_marginStart="20dp"
75              android:text="@string/btn_detail"
76         app:layout_constraintStart_toEndOf="@id/button_shopee"
77         app:layout_constraintTop_toTopOf="@id/button_shopee"/>
78      </androidx.constraintlayout.widget.ConstraintLayout>
79  </androidx.cardview.widget.CardView>
```

Tabel 9. Source Code Jawaban Soal 1

10. fragment_detail.xml

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

```
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp"
    android:background="#F6F6F6"
    tools:context=".DetailFragment">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="#F6F6F6">

        <ImageView
            android:id="@+id/img_item_photo"
            android:layout_width="350dp"
            android:layout_height="350dp"
            android:scaleType="centerCrop"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            android:contentDescription="@string/app_name"
            tools:src="@drawable/marsya_bag" />

        <TextView
            android:id="@+id/tv_name"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textStyle="bold"
            android:textColor="@color/black"
            android:textSize="25sp"
            android:layout_marginTop="16dp"
     app:layout_constraintTop_toBottomOf="@id/img_item_photo"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            tools:text="BOSTANTEN [Marsya Bag]" />

        <TextView
            android:id="@+id/tv_harga"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textColor="@android:color/holo_red_dark"
            android:textStyle="bold"
            android:textSize="18sp"
            android:layout_marginTop="8dp"
            app:layout_constraintTop_toBottomOf="@id/tv_name"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            tools:text="Harga: Rp179.999" />

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

```
57              android:layout_marginTop="30dp"
58              android:textSize="16sp"
59              android:textColor="@color/black"
60              android:background="#F6F6F6"
61              android:textAlignment="center"
62              android:lineSpacingExtra="5dp"
63              android:padding="8dp"
64             app:layout_constraintTop_toBottomOf="@id/tv_harga"
65              app:layout_constraintStart_toStartOf="parent"
66              app:layout_constraintEnd_toEndOf="parent"
67              tools:text="Deskripsi lengkap tentang tas
   berbahan kulit PU premium, cocok untuk kuliah dan kerja." />
68
69      </androidx.constraintlayout.widget.ConstraintLayout>
70  </ScrollView>
```

Tabel 10. Source Code Jawaban Soal 1

11. nav_graph.xml

```
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       app:startDestination="@id/homeFragment">
6
7       <fragment
8           android:id="@+id/homeFragment"
9           android:name="com.example.modul3.HomeFragment"
10          android:label="fragment_home"
11          tools:layout="@layout/fragment_home">
12          <action
13        android:id="@+id/action_homeFragment_to_detailFragment"
14              app:destination="@id/detailFragment" />
15      </fragment>
16
17      <fragment
18          android:id="@+id/detailFragment"
19          android:name="com.example.modul3.DetailFragment"
20          android:label="fragment_detail"
21          tools:layout="@layout/fragment_detail" />
22  </navigation>
```

Tabel 11. Source Code Jawaban Soal 1

12. strings.xml

```
1   <resources>
2       <string name="app_name">MODUL3</string>
3       <string name="btn_detail">Detail</string>
4       <string name="btn_shopee">Shopee</string>
```

```
5
6       <string-array name="data_name">
7           <item>BOSTANTEN [Marsya Bag]</item>
8           <item>BOSTANTEN [LYN Bag]</item>
9           <item>BOSTANTEN [Reigny Bag]</item>
10          <item>BOSTANTEN [Madeline Bag]</item>
11          <item>BOSTANTEN [Totebag]</item>
12          <item>BOSTANTEN [Mini Cendy Bag]</item>
13          <item>BOSTANTEN [Amora Bag]</item>
14          <item>MOSSDOOM [Gradien Hand Bag]</item>
15          <item>MOSSDOOM [Lysmare Bag]</item>
16          <item>MOSSDOOM [Delphine Bag]</item>
17      </string-array>
18
19      <string-array name="data_photo">
20          <item>@drawable/marsya_bag</item>
21          <item>@drawable/lyn_bag</item>
22          <item>@drawable/reigny_bag</item>
23          <item>@drawable/madeline_bag</item>
24          <item>@drawable/tote_bag</item>
25          <item>@drawable/mini_cendy_bag</item>
26          <item>@drawable/amora_bag</item>
27          <item>@drawable/gradien_bag</item>
28          <item>@drawable/lysmare_bag</item>
29          <item>@drawable/delphine_bag</item>
30      </string-array>
31
32      <string-array name="data_link">
33          <item>https://id.shp.ee/ixmk3vP</item>
34          <item>https://id.shp.ee/56fWCro</item>
35          <item>https://id.shp.ee/957MHuZ</item>
36          <item>https://id.shp.ee/7ARV68N</item>
37          <item>https://id.shp.ee/quuhB8b</item>
38          <item>https://id.shp.ee/RPVhVds</item>
39          <item>https://id.shp.ee/vKZm8c9</item>
40          <item>https://id.shp.ee/6tzCzpz</item>
41          <item>https://id.shp.ee/t3osSMq</item>
42          <item>https://id.shp.ee/GaVtJQ1</item>
43      </string-array>
44
45      <string-array name="data_harga">
46          <item>Rp179.999</item>
47          <item>Rp159.999</item>
48          <item>Rp189.999</item>
49          <item>Rp179.999</item>
50          <item>Rp169.999</item>
51          <item>Rp159.999</item>
52          <item>Rp189.999</item>
53          <item>Rp249.000</item>
54          <item>Rp189.000</item>
55          <item>Rp199.000</item>
56      </string-array>
```

```
57
58      <string-array name="data_deskripsi">
59          <item>(Bostanten) 100% asli. Selamat datang di toko
    Bostanten, di mana kami berusaha menyediakan produk yang
    memenuhi kebutuhan kualitas dan efisiensi pelanggan. Beli
    produk kami dengan mudah dari pusat perbelanjaan Shopee
    karena kami menjamin keaslian 100%. Bahan terbuat dari kulit
    PU dengan pilihan warna Coklat, Hitam, Off Putih, Biru dan
    Coklat, Aprikot dan Hitam, Kuning Biru Pink, Off Putih dan
    Coklat, serta Off Putih dan Hitam. Berukuran 19.5 cm (Lebar)
    × 11.5 cm (Tinggi) × 7 cm (Tebal), dengan berat sekitar 0.35
    kg. Tas ini memiliki ruang penyimpanan besar, satu kantong,
    serta tali belakang yang dapat disesuaikan antara 98–122 cm.
    Cocok digunakan untuk sekolah, kantor, perjalanan, perjalanan
    bisnis, belanja, pertemuan, makan, kencan, dan acara sosial
    seperti pesta atau konser. Target penggunanya adalah wanita,
    gadis, pekerja kantoran, pelajar, dan remaja.</item>
60          <item>(Bostanten) 100% Asli. Selamat datang di toko
    Bostanten, di mana kami akan berusaha untuk menyediakan
    produk yang memenuhi kebutuhan kualitas dan efisiensi kepada
    pelanggan. Beli produk kami dengan mudah dari pusat
    perbelanjaan Shopee karena kami menjamin keaslian 100%. Tas
    ini terbuat dari bahan kulit PU dengan pilihan warna Coklat,
    Off Putih, Hitam, dan Aprikot Coklat. Memiliki dimensi 29 cm
    (Lebar) × 14 cm (Tinggi) × 9 cm (Tebal), dan berat sekitar
    0.34 kg. Bagian dalam terdiri dari ruang penyimpanan besar,
    satu tas masukan dalam, dan satu tas zipper internal.
    Pegangan tas memiliki jarak sedang yaitu 31 cm. Sangat cocok
    digunakan untuk sekolah, kantor, pesta, dan
    perjalanan.</item>
61          <item>(Bostanten) 100% asli. Selamat datang di toko
    Bostanten, di mana kami akan berusaha untuk menyediakan
    produk yang memenuhi kebutuhan kualitas dan efisiensi kepada
    pelanggan. Beli produk kami dengan mudah dari pusat
    perbelanjaan Shopee karena kami menjamin keaslian 100%. Tas
    ini menggunakan bahan kulit PU dan tersedia dalam warna Off
    Putih, Warna Kamel, dan Hitam. Ukurannya adalah 20 cm (Lebar)
    × 16 cm (Tinggi) × 9 cm (Tebal), dengan berat 0.27 kg.
    Dilengkapi dengan ruang penyimpanan besar, tas masukan, dan
    sabuk bahu dengan tinggi tengah sekitar 92–112 cm yang dapat
    disesuaikan dan dilepas pasang. Pegangan tas berukuran
    sedang, yaitu 9 cm. Cocok untuk sekolah, kantor, pesta,
    maupun perjalanan.</item>
62          <item>(Bostanten) 100% asli. Selamat datang di toko
    Bostanten, di mana kami akan berusaha untuk menyediakan
    produk yang memenuhi kebutuhan kualitas dan efisiensi kepada
    pelanggan. Beli produk kami dengan mudah dari pusat
    perbelanjaan Shopee karena kami menjamin keaslian 100%. Tas
    ini terbuat dari bahan kulit PU dan tersedia dalam warna Off
    Putih, Warna Kamel, Hitam, Abu-abu, dan Coklat. Berukuran
    28.5 cm (Lebar) × 21.5 cm (Tinggi) × 7 cm (Tebal), dengan
    berat 0.29 kg. Bagian dalam terdiri dari ruang penyimpanan
```

	besar, satu tas zipper belakang, dan satu saku masukan depan. Sabuk bahunya memiliki tinggi tengah sekitar 0–111 cm dan tidak dapat disesuaikan maupun dilepas. Tas ini cocok digunakan untuk sekolah, kantor, pesta, maupun perjalanan.</item>
63	<item>(Bostanten) 100% asli. Selamat datang di toko Bostanten, di mana kami akan berusaha untuk menyediakan produk yang memenuhi kebutuhan kualitas dan efisiensi kepada pelanggan. Beli produk kami dengan mudah dari pusat perbelanjaan Shopee karena kami menjamin keaslian 100%. Keserbagunaan, kepraktisan, elegan, dan gaya kedewasaan adalah inti keindahan dari tas ini. Bahan warna solid dan lekukan yang natural memberikan tampilan yang serbaguna serta memiliki plastisitas yang tepat untuk menonjolkan gaya dewasa wanita. Barang pegangan terpopuler di musim ini adalah tote bag—tas jinjing berkapasitas besar yang berkualitas baik dan terbuat dari bahan tahan air. Kapasitasnya sangat luar biasa; Anda bisa memasukkan laptop, buku teks, tablet, dan barang lainnya dengan mudah. Kepraktisannya sempurna dan sangat ramah bagi pelajar, pekerja kantoran, maupun ibu-ibu yang membutuhkan ruang ekstra. Tas ini terbuat dari bahan kulit PU dan tersedia dalam kombinasi warna Off Putih dan Coklat, Off Putih dan Pink, serta Off Putih dan Hijau. Berukuran 40 cm (Lebar) × 27 cm (Tinggi) × 12 cm (Tebal) dengan berat 0.50 kg. Dilengkapi ruang penyimpanan besar, kantong, dan kantong resleting, serta tali belakang sepanjang 64 cm. Tote bag ini bukan sekadar barang sederhana yang bisa dipakai kapan saja, tetapi juga mencerminkan sikap terhadap kehidupan. Cocok digunakan di sekolah, kantor, perjalanan, bisnis, belanja, pertemuan, makan, kencan, maupun acara sosial seperti pesta dan konser. Target penggunanya adalah wanita, gadis, pekerja kantoran, pelajar, dan remaja.</item>
64	<item>(Bostanten) 100% asli. Selamat datang di toko Bostanten, di mana kami akan berusaha untuk menyediakan produk yang memenuhi kebutuhan kualitas dan efisiensi kepada pelanggan. Beli produk kami dengan mudah dari pusat perbelanjaan Shopee karena kami menjamin keaslian 100%. Tas ini terbuat dari bahan kulit PU dan tersedia dalam berbagai warna, seperti Coklat, Off Putih, Hitam, Aprikot dan Coklat, Off Putih dan Coklat, serta Off Putih dan Hitam. Dengan dimensi 12 cm (Lebar) × 18.5 cm (Tinggi) × 6.5 cm (Tebal) dan berat 0.16 kg, tas ini memiliki ruang penyimpanan besar dan satu kantong tambahan. Dilengkapi dengan tali belakang sepanjang 132 cm, tas ini cocok digunakan di sekolah, kantor, perjalanan, perjalanan bisnis, belanja, pertemuan, makan, kencan, serta berbagai acara sosial seperti pesta dan konser. Target penggunanya ialah wanita, gadis, pekerja kantoran, pelajar, dan remaja.</item>
65	<item>(Bostanten) 100% asli. Selamat datang di toko Bostanten, di mana kami akan berusaha untuk menyediakan produk yang memenuhi kebutuhan kualitas dan efisiensi kepada

```
    pelanggan. Beli produk kami dengan mudah dari pusat
    perbelanjaan Shopee karena kami menjamin keaslian 100%. Tas
    ini terbuat dari bahan kulit PU dan tersedia dalam pilihan
    warna Off Putih, Hitam, Coklat, dan Aprikot. Dengan dimensi
    22.5 cm (Lebar) × 26.5 cm (Tinggi) × 3.5 cm (Tebal) dan berat
    0.28 kg, tas ini dilengkapi dengan ruang penyimpanan besar,
    satu kantong, dan satu kantong resleting. Tali belakangnya
    memiliki panjang antara 103–124 cm. Tas ini cocok digunakan
    untuk sekolah, kantor, perjalanan, perjalanan bisnis,
    belanja, pertemuan, makan, kencan, serta acara sosial seperti
    pesta dan konser. Target penggunanya ialah wanita, gadis,
    pekerja kantoran, pelajar, dan remaja.</item>
66          <item>Selamat datang di produk MOSSDOOM, dijamin
    keasliannya 100%! Tas ini terbuat dari bahan kulit PU. Dengan
    ukuran panjang 19.5 cm, lebar 8 cm, tinggi 16.5 cm, dan
    dengan berat 0.36 kg. Tersedia hanya warna Pink. Tas ini
    dilengkapi dengan satu saku penyimpanan besar, satu saku
    sisipan, satu saku beritsleting, dan tinggi pegangan tas
    tangan sebesar 7cm. Untuk tali bahu memiliki panjang antara
    47.5-59cm (dapat disesuaikan). Semua gambar produk ini
    diambil dalam kehidupan nyata, jadi dijamin keaslinya.</item>
67          <item>Selamat datang di official MOSSDOOM store, 100%
    menjamin keaslian brand produk kami. Tas ini terbuat dari
    bahan kulit PU. Berukuran 20 cm (Panjang) x 8 cm (Lebar) x 14
    cm (Tinggi) dan berat 0.36 kg. Tas ini tersedia dalam pilihan
    warna Blue, Light Pink, Pink, Brown, Beige White, Black. Tas
    ini dilengkapi dengan ruang penyimpanan besar, satu saku
    sisipan, satu saku beresleting, dan dengan tali bahu setinggi
    kira-kira 106-130cm (yang dapat disesuaikan, bisa dilepas
    pasang). Semua foto produk kami adalah foto asli, diambil
    dari store official. Jadi dijamin keasliannya.</item>
68          <item>Selamat datang di official MOSSDOOM store, 100%
    menjamin keaslian brand produk kami. Tas ini terbuat dari
    bahan kulit PU. Dengan ukuran 17 cm (Panjang) x 9 cm (Lebar)
    x 10 cm (Tinggi) dan berat 0.29 kg. Tersedia dengan beberapa
    pilhan warna Pink, Blue, Purple, Apricot, Beige White, Black.
    Tas ini memiliki struktur internal dengan saku penyimpanan
    besar, tinggi tali bahunya sekitar 110-121 cm (yang dapat
    disesuaikan dan bisa dilepas pasang). Semua foto produk kami
    adalah foto asli, diambil dari store official. Jadi dijamin
    keasliannya.</item>
69      </string-array>
70
71      <string name="label_nama">Nama</string>
72      <string name="label_harga">Harga:</string>
73
74      <string name="placeholder_nama">Nama Tas</string>
75      <string name="placeholder_harga">Harga Tas</string>
76  </resources>
```

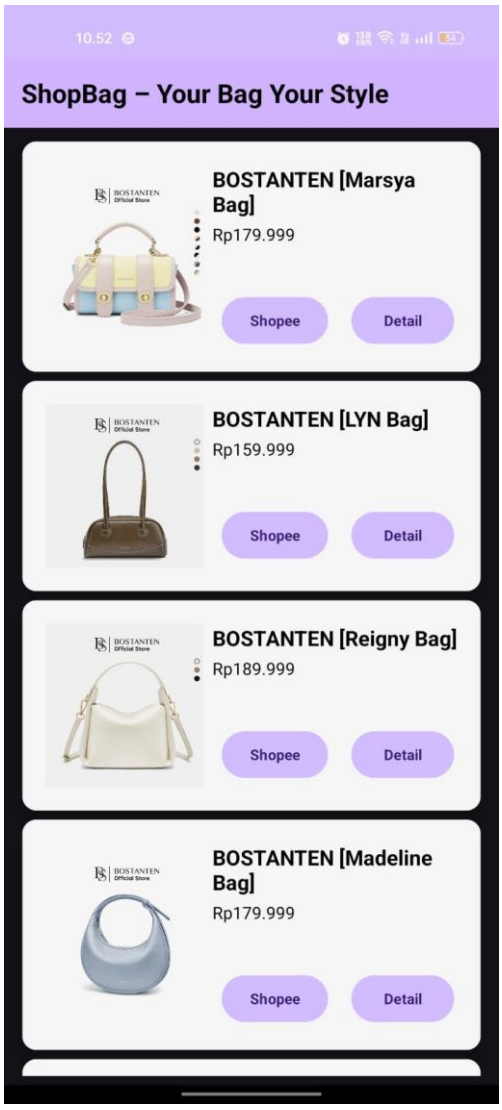
Tabel 12. Source Code Jawaban Soal 1

13. activity_main.xml

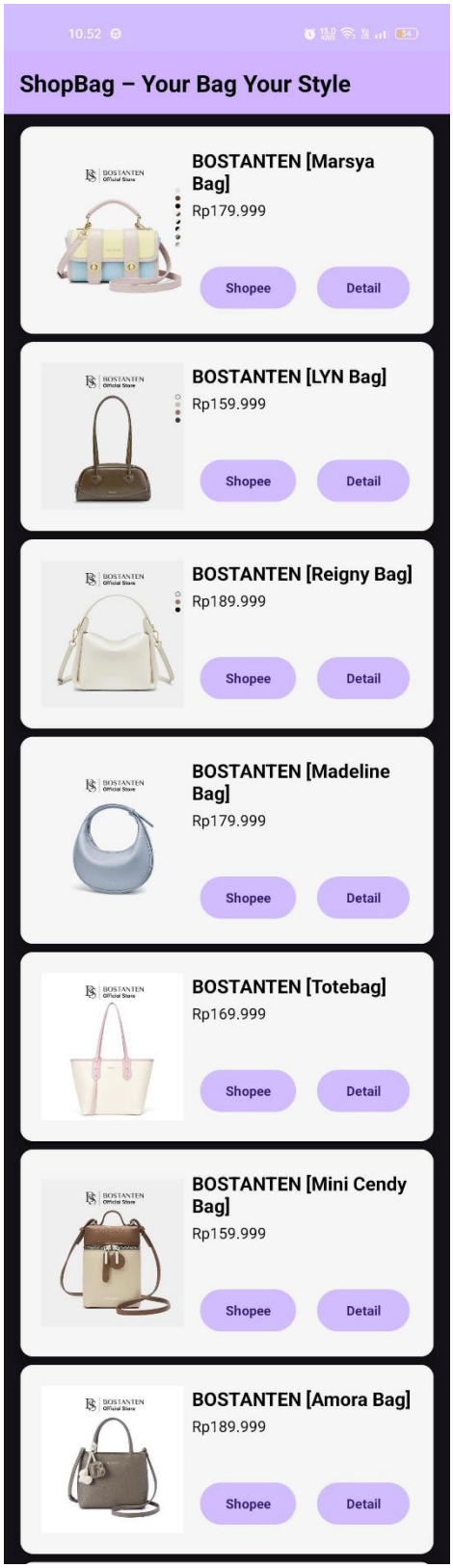
```
1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout
3  xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:tools="http://schemas.android.com/tools"
5      xmlns:app="http://schemas.android.com/apk/res-auto"
6      android:layout_width="match_parent"
7      android:layout_height="match_parent"
8      tools:context=".MainActivity"
9      tools:ignore="ExtraText">
10
11     <androidx.fragment.app.FragmentContainerView
12         android:id="@+id/nav_host_fragme
13   android:name="androidx.navigation.fragment.NavHostFragment"
14         android:layout_width="match_parent"
15         android:layout_height="match_parent"
16         app:navGraph="@navigation/nav_graph"
17         app:defaultNavHost="true" />
18 </androidx.constraintlayout.widget.ConstraintLayout>
```

Tabel 13. Source Code Jawaban Soal 1

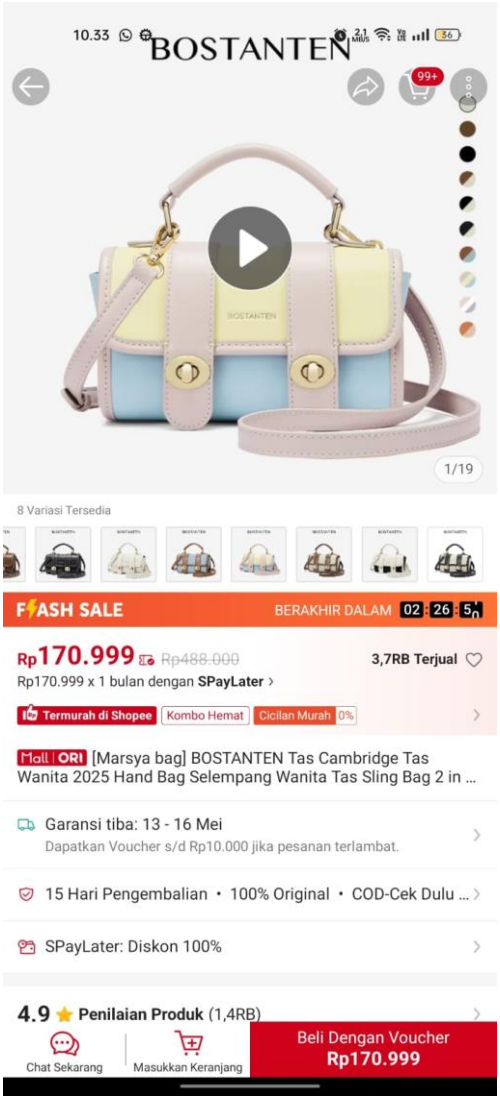
B. Output Program



Gambar 2. Screenshot Hasil Jawaban Soal 1 UI List

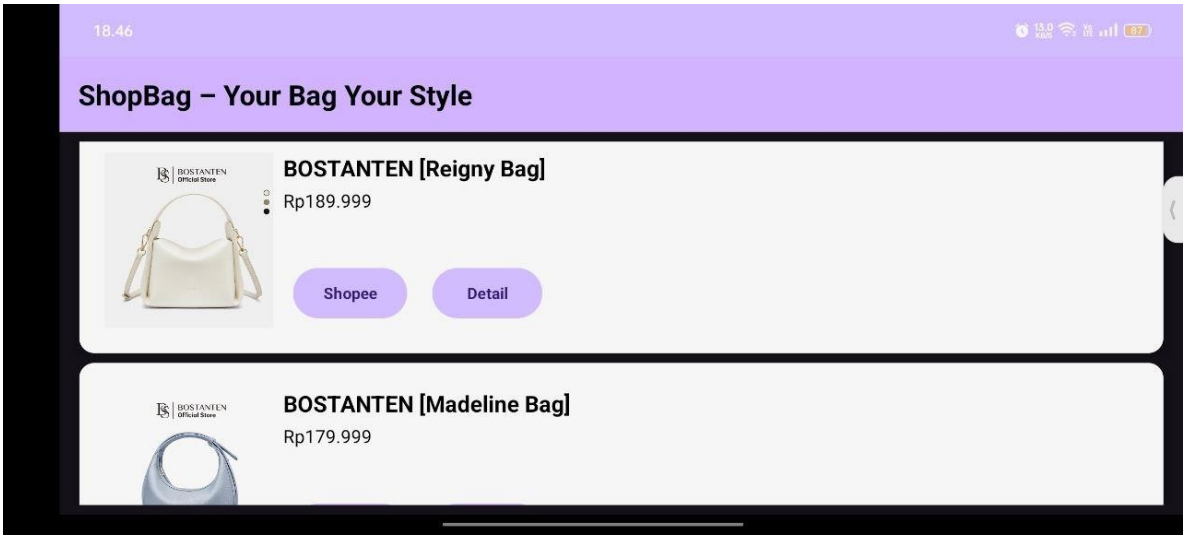


Gambar 3. Screenshot Hasil Jawaban Soal 1 UI List 10 Item

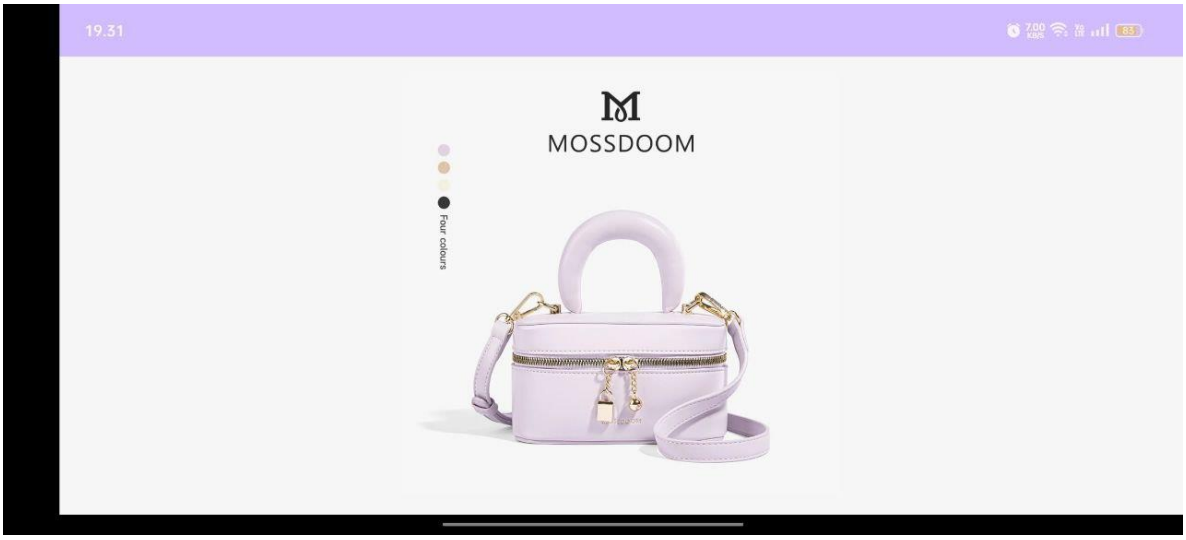


Gambar 4. Screenshot Hasil Jawaban Soal 1 Berpindah ke Aplikasi Lain Saat Menekan Tombol Shopee

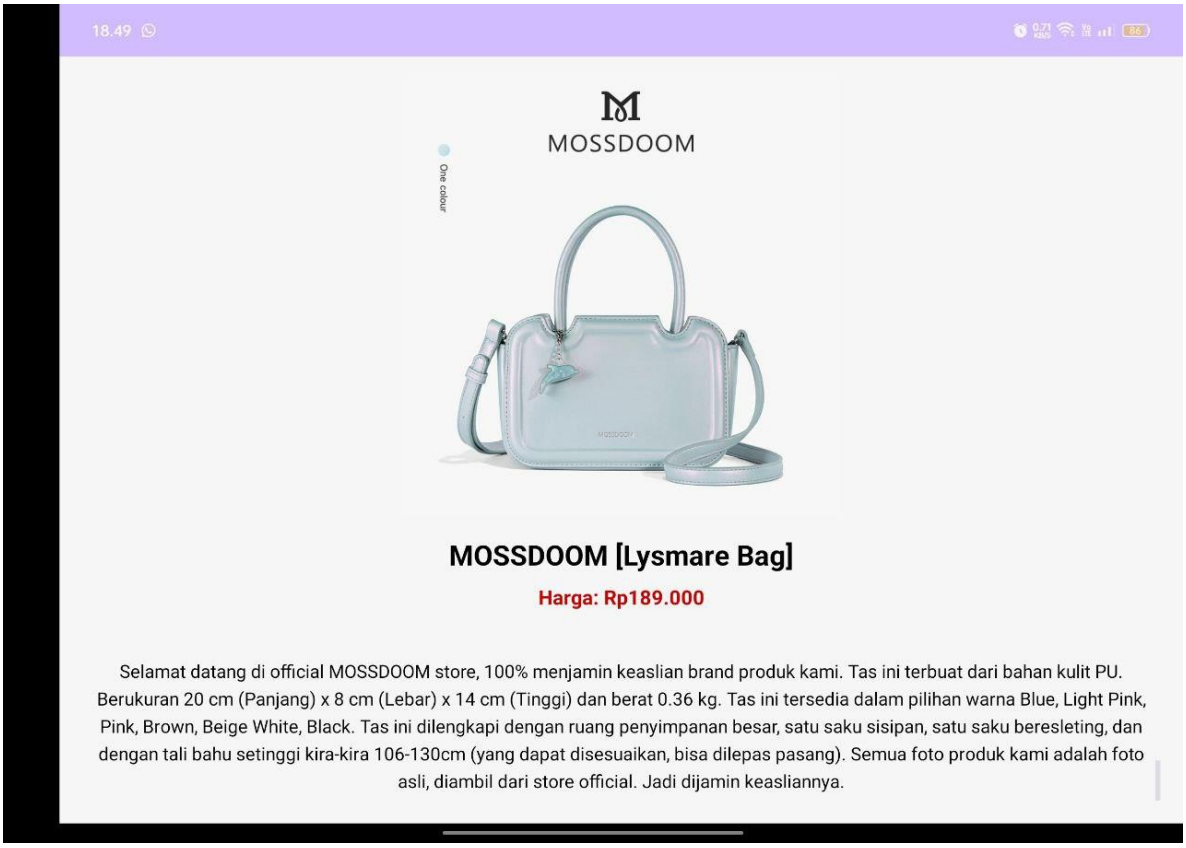
Gambar 5. Screenshot Hasil Jawaban Soal 1 UI Detail Saat Menekan Tombol Detail



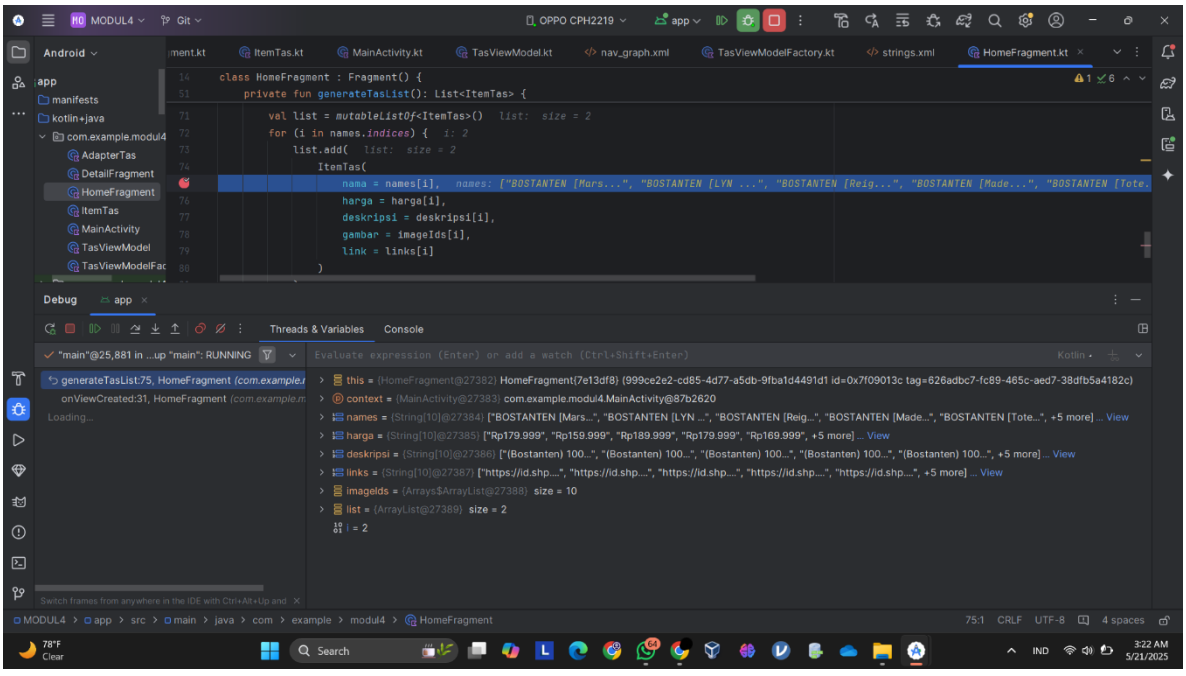
Gambar 6. Screenshot Hasil Jawaban Soal 1 UI List Item Rotate



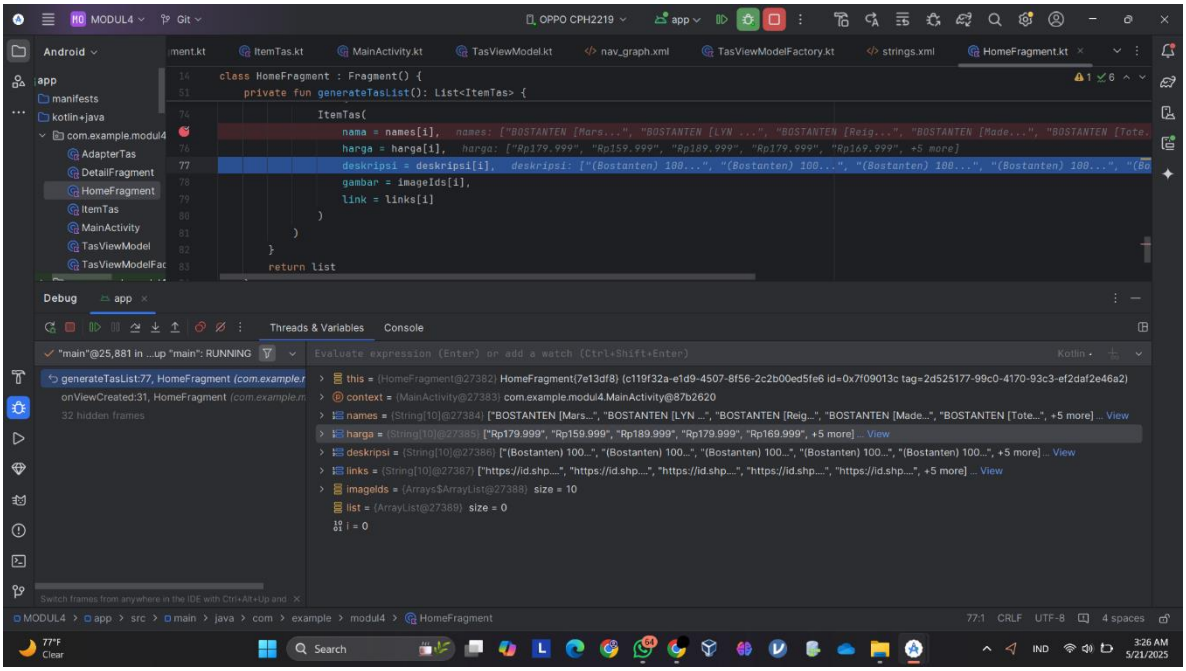
Gambar 7. Screenshot Hasil Jawaban Soal 1 UI Detail Item Rotate



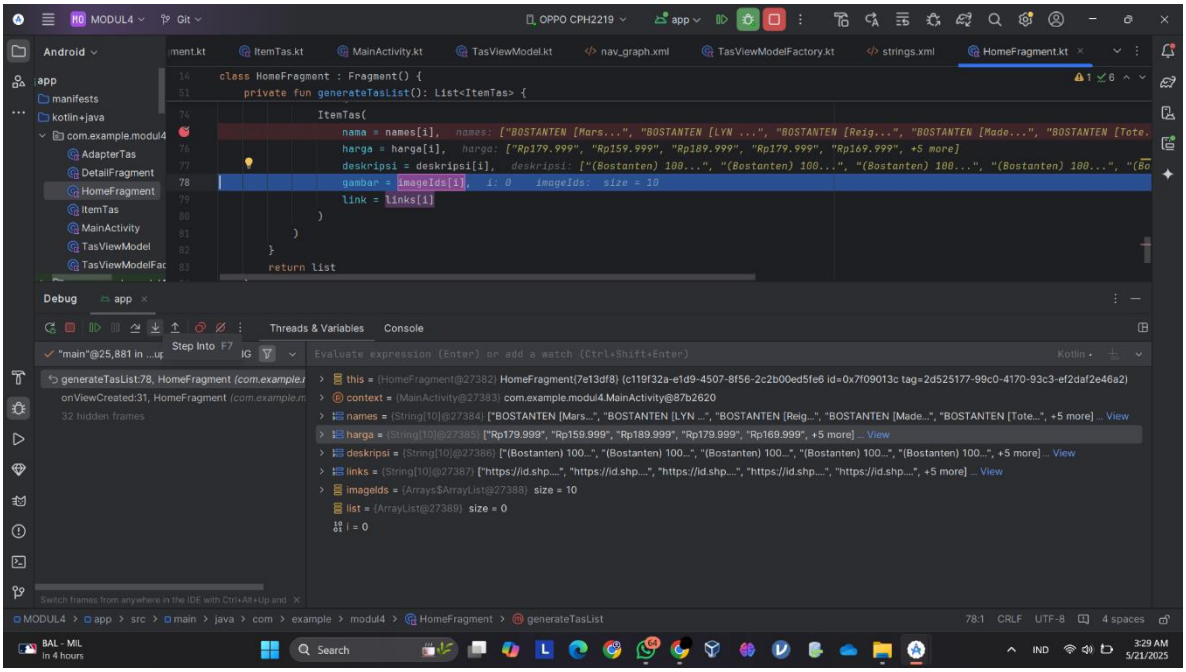
Gambar 8. Screenshot Hasil Jawaban Soal 1 UI Detail Item Rotate dan Scrollable



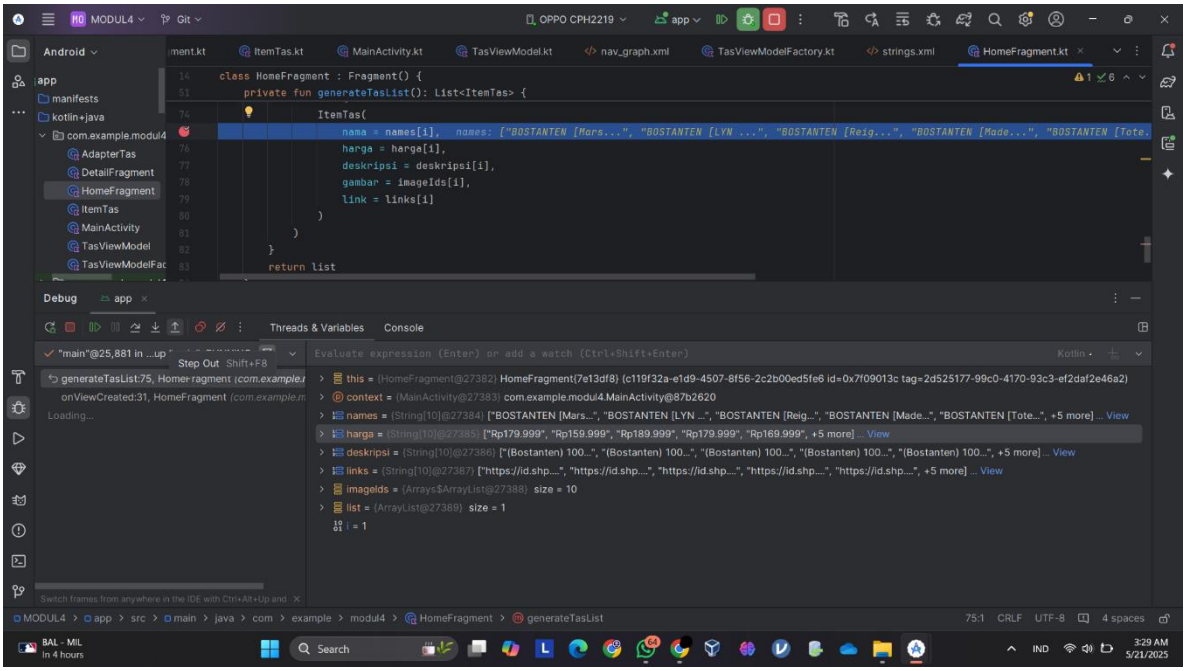
Gambar 9. Screenshot Hasil Jawaban Soal 1 Debugging Aplikasi



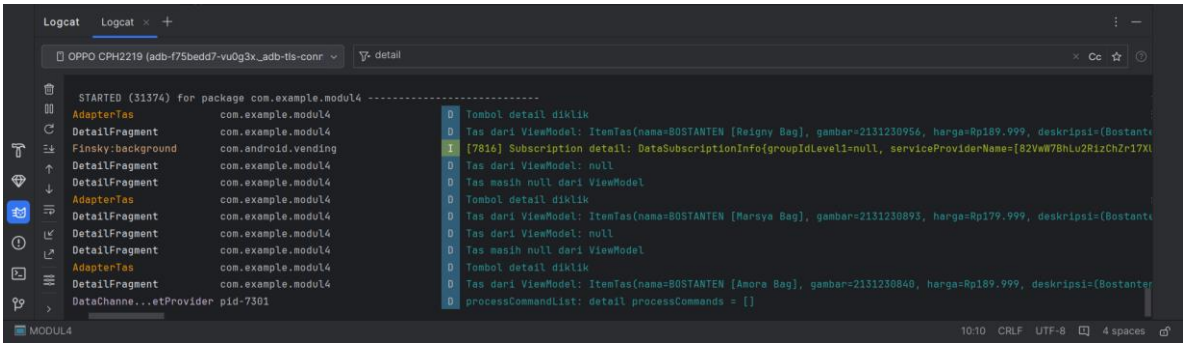
Gambar 10. Screenshot Hasil Jawaban Soal 1 Debugging Aplikasi Step Over



Gambar 11. Screenshot Hasil Jawaban Soal 1 Debugging Aplikasi Step Into



Gambar 12. Screenshot Hasil Jawaban Soal 1 Debugging Aplikasi Step Out



Gambar 13. Screenshot Hasil Jawaban Soal 1 Logging

C. Pembahasan

1. HomeFragment.kt

Pada line 1, `package com.example.modul4` digunakan untuk menentukan nama package dari aplikasi Android. Package ini adalah namespace unik untuk memisahkan file dalam aplikasi. Kemudian, pada line 3 sampai 13, class-class penting yang digunakan seperti `Fragment`, `LayoutInflater`, `RecyclerView`, binding untuk layout `FragmentHome`, dan lain sebagainya. Lalu, line 15 sampai 19 membuat fragment bernama `HomeFragment` yang merupakan turunan dari `Fragment`. `recyclerViewState` disiapkan untuk menyimpan posisi scroll RecyclerView. `viewModel` ialah untuk referensi ke `TasViewModel` yang menyimpan data tas dan navigasi. `_binding` digunakan untuk mengakses elemen layout dari `FragmentHomeBinding`. `binding` getter yang memastikan `_binding` tidak `null` (pakai `!!`). Line 21 sampai 27, membuat tampilan `Fragment` dari layout XML (`fragment_home.xml`) menggunakan View Binding. Line 29 sampai 49 terdapat fungsi onViewCreated untuk memulai penggunaan UI. Juga terdapat inisialisasi ViewModel, `ViewModelProvider` digunakan untuk mendapatkan instance `TasViewModel` agar datanya tetap bertahan saat fragment berpindah-pindah dan `requireActivity()` artinya ViewModel milik activity (bisa diakses banyak fragment). Pada saat memuat data akan melakukan pemanggilan fungsi `loadData()` untuk mengisi list tas ke dalam `StateFlow` di ViewModel, `generateTasList()` adalah fungsi yang membuat data dummy dari resource XML. Di line tersebut juga terdapat fungsi untuk menampilkan RecyclerView, menunggu data dari `viewModel.tasList` (yang berupa `StateFlow`). Saat data muncul, maka akan membuat `AdapterTas` baru dan memasukkan daftar item, enyambungkan RecyclerView ke adapter, dan mengatur layout-nya sebagai daftar vertikal (LinearLayoutManager). Callback `item -> viewModel.selectTas(item)` digunakan untuk memilih item yang akan ditampilkan di detail (disimpan di ViewModel). Pada navigasi ke DetailFragment, melakukan pemantauan event `navigateToDetail` dari ViewModel. Jika event

terjadi (misalnya item diklik), dan saat ini masih di HomeFragment, maka akan berpindah ke DetailFragment lewat Navigation Component.

Selanjutnya, pada line 51 sampai 54 terdapat method `onDestroyView`, yang di mana akan membersihkan View Binding agar tidak terjadi memory leak ketika fragment dihancurkan. Pada line 56 sampai 61 terdapat fungsi generateTasList(), fungsi untuk membuat list dummy tas dari `res/values/strings.xml` dan res/drawable/. Fungsi ini melakukan penggabungan nama tas, harga, deskripsi, gambar, dan link produk. Line 63 sampai 74, menyediakan daftar ID gambar tas dalam bentuk list. Line 76 sampai 90, menggabungkan data teks dan gambar menjadi objek `ItemTas`, lalu dikembalikan sebagai list yang akan ditampilkan di `RecyclerView`.

2. DetailFragment.kt

Pada line 1, `package com.example.modul4` digunakan untuk menentukan nama package dari aplikasi Android. Package ini adalah namespace unik untuk memisahkan file dalam aplikasi. Kemudian, pada line 3 sampai 12, class-class penting yang digunakan seperti `lifecycle fragment, log debugging, View Binding`, dan lain sebagainya. Lalu, line 14 sampai 17 mendefinisikan fragment detail bernama `DetailFragment` sebagai turunan dari `Fragment`, yang di dalamnya terdapat `viewModel` instance dari `TasViewModel`, digunakan untuk mengambil data tas yang dipilih. `_binding` untuk View Binding dari layout `fragment_detail.xml`, dan `binding` akses yang aman (non-null) ke `_binding`. Line 19 sampai 25 membuat dan mengembalikan tampilan (UI) fragment. Layout di-inflate menggunakan View Binding, sehingga bisa langsung akses elemen UI dari `fragment_detail.xml` lewat `binding`. Line 27 sampai 43 terdapat penginisialan ViewModel dengan mengambil `TasViewModel milik Activity` agar bisa berbagi data dengan `HomeFragment`. Mengamati data `selectedTas` (kemungkinan `StateFlow<ItemTas?>`). Saat tas dipilih, akan otomatis masuk ke sini. Jika `tas` tidak kosong, maka akan menampilkan nama tas ke `tvName`, harga ke `tvHarga`, deskripsi ke `tvDeskripsi`, dan gambar ke

`imgItemPhoto`. Selain itu, terdapat logging ntuk debugging di Logcat, mengecek apakah data `tas` benar-benar diterima. Line 45 sampai 50, `viewModel.clearSelectedTas()` digunakan untuk membersihkan data `selectedTas`, agar ketika kembali ke Home, data sebelumnya tidak nyangkut. `_binding = null` digunakan untuk mencegah memory leak.

3. ItemTas.kt

Pada line 1, `package com.example.modul4` digunakan untuk menentukan nama package dari aplikasi Android. Package ini adalah namespace unik untuk memisahkan file dalam aplikasi. Line 3 dan 4 mengimpor `Parcelable`, yang di mana itu interface untuk mengirim data antar komponen Android (seperti antar fragment atau activity) dan `@Parcelize` adalah annotation dari Kotlin untuk membuat proses parceling otomatis tanpa harus menulis `writeToParcel()` dan `CREATOR`. Lalu, line 6 sampai 13 membuat class bisa dikirim lewat Bundle dengan mudah dengan cara `@Parcelize`. `data class` ialah class khusus di Kotlin untuk menyimpan data dengan otomatis menyediakan fungsi seperti `equals()`, `hashCode()`, dan `toString()`. `ItemTas` menyimpan informasi untuk 1 item tas, `nama` untuk nama produknya, `gambar` untuk ID resource drawable gambar produk (`Int`), `harga` untuk harga produk (sebagai string, bisa mengandung Rp, titik, dsb), `deskripsi` untuk penjelasan produk di halaman detail, dan `link` adalah URL untuk ke Shopee atau marketplace lainnya. Fungsi utama dari class ini digunakan untuk menyimpan data tas dalam daftar dan mengirim data ke `DetailFragment` melalui `Bundle` secara aman dan efisien.

4. AdapterTas.kt

Pada line 1, `package com.example.modul4` digunakan untuk menentukan nama package dari aplikasi Android. Package ini adalah namespace unik untuk memisahkan file dalam aplikasi. Kemudian, pada line 3 sampai 11, class-class penting yang digunakan. Lalu, line 13 sampai 16 mendefinisikan class `AdapterTas`. `AdapterTas` menerima dua parameter, `listTas` ialah daftar

objek `ItemTas` untuk ditampilkan dan `onItemClick` lambda function untuk menangani klik tombol detail. Fungsi ini akan dipanggil di `HomeFragment` untuk navigasi ke `DetailFragment`. Adapter ini menghubungkan data dengan tampilan di `RecyclerView`. Line 18 dan 19, `inner class ListViewHolder` merepresentasikan tiap item di dalam daftar `RecyclerView` dengan menggunakan View Binding (`ItemTasBinding`) untuk mengakses elemen UI. Line 21 sampai 24, mengisi teks dan gambar dari objek `ItemTas` ke komponen layout (`TextView`, `ImageView`) dan menampilkan nama, harga, dan gambar tas ke tampilan masing-masing.

Line 25 sampai 28 terdapat proses ketika tombol Detail diklik, saat tombol "Detail" diklik, fungsi `onItemClick(tas)` dipanggil. Fungsi ini didefinisikan di `HomeFragment`, yang kemudian akan memicu navigasi ke `DetailFragment`. Line 29 sampai 35 melakukan proses pada tombol Shopee, ketika tombol Shopee diklik, maka intent dibuka untuk mengunjungi link eksternal (ke halaman Shopee). Line 37 sampai 40, membuat `ViewHolder` baru dari layout XML `item_tas.xml`. Line 42 sampai 44, mengisi data ke `ViewHolder` berdasarkan posisi dalam list. Line 46 dan 47, mengembalikan jumlah total item dalam list, dengan menentukan jumlah item yang akan ditampilkan di RecyclerView.

5. MainActivity.kt

Pada line 1, `package com.example.modul4` digunakan untuk menentukan nama package dari aplikasi Android. Package ini adalah namespace unik untuk memisahkan file dalam aplikasi. Kemudian, pada line 3 sampai 7, mengimpor kelas yang dibutuhkan, diantaranya yaitu `Bundle` untuk menyimpan state saat `Activity` dibuat, `AppCompatActivity` ialah superclass dari semua activity modern, dan `ActivityMainBinding` ialah class auto-generated dari layout `activity_main.xml` (fitur View Binding). Line 9 sampai 15, `ActivityMainBinding` ialah View Binding untuk layout `activity_main.xml`. `setContentView(binding.root)` untuk menampilkan layout utama yang diatur lewat View Binding. Dalam layout

`activity_main.xml`, ada `FragmentContainerView` atau `NavHostFragment` dengan `id = nav_host_fragment`. Line 16 sampai 23 melakukan penanganan tombol kembali (`onBacPressed`). Ketika tombol back ditekan, maka akan mengecek apakah masih ada fragment di backstack navigasi. Jika masih ada, akan kembali ke fragment sebelumnya menggunakan `popBackStack()`. Jika tidak ada lagi, maka akan memanggil `super.onBackPressed()` untuk keluar dari aplikasi atau kembali ke activity sebelumnya.

6. TasViewModel.kt

Pada line 1, `package com.example.modul4` digunakan untuk menentukan nama package dari aplikasi Android. Package ini adalah namespace unik untuk memisahkan file dalam aplikasi. Kemudian, pada line 3 sampai 7 terdapat pengimporan kelas-kelas yang dibutuhkan. Line 9 sampai 11, `_tasList` ialah flow internal (mutable) berisi list tas dan `tasList` ialah flow eksternal (immutable) yang diamati `HomeFragment`. Line 14 dan 15 melakukan penyimpanan 1 tas yang sedang dipilih user dan digunakan oleh `DetailFragment` untuk menampilkan detail. Line 17 dan 18 ialah event navigasi ke halaman detail, digunakan untuk memberitahu UI agar pindah ke `DetailFragment`. `SharedFlow` cocok untuk event satu-kali seperti navigasi (berbeda dengan `StateFlow` yang menyimpan nilai terakhir).

Pada line 20 sampai 28, menyimpan item yang dipilih ke `_selectedTas`, lalu mengirimkan sinyal event navigasi melalui `_navigateToDetail`. Line 30 sampai 33 melakukan pemuatan data tas, dengan memasukkan daftar tas ke dalam `StateFlow` dan dipanggil sekali saat `HomeFragment` muncul. Line 35 sampai 38 melakukan pembersihan item tas yang terpilih, dipanggil saat `DetailFragment` dihancurkan agar ViewModel tidak menyimpan referensi ke data lama.

7. TasViewModelFactory.kt

Pada line 1, `package com.example.modul4` digunakan untuk menentukan nama package dari aplikasi Android. Package ini adalah namespace unik untuk memisahkan file dalam aplikasi. Kemudian, pada line 3 dan 4 terdapat pengimporan kelas-kelas yang dibutuhkan. Line 6 sampai 13, `ViewModelProvider.Factory` adalah interface untuk membuat instance ViewModel. `create()` akan dipanggil oleh `ViewModelProvider` saat memanggil `isAssignableFrom()` digunakan untuk mengecek apakah `modelClass` adalah turunan dari `TasViewModel`. `return TasViewModel() as T` artinya akan membuat instance `TasViewModel`, lalu meng-cast-nya ke `T`.

8. fragment_home.xml

Pada line 1 sampai 8 terdapat `LinearLayout` yang di manakomponen dasar dengan orientasi vertikal untuk menyusun anak-anaknya dari atas ke bawah, `match_parent` digunakan untuk memenuhi seluruh lebar dan tinggi layar, dan `tools:context` hanya untuk keperluan preview di Android Studio (menunjukkan layout ini dipakai oleh `HomeFragment`). Line 10 sampai 19, menampilkan judul aplikasi di bagian atas, dengan background ungu (`#D1B3FF`), teks hitam, ukuran besar `25sp`, tebal, dan padding `15dp` agar teks tidak terlalu mepet. Kemudian, line 21 sampai 27 ialah tempat untuk menampilkan daftar produk tas menggunakan `RecyclerView`, juga menggunakan seluruh sisa ruang layar di bawah `TextView`. `id` tersebut yang diakses di `HomeFragment.kt` untuk menampilkan daftar tas melalui adapter.

9. item_tas.xml

Pada line 1 sampai 14 terdapat CardView (root) untuk membungkus seluruh isi item agar terlihat seperti kartu dengan bayangan. `cardCornerRadius="12dp"` untuk mengatur sudut membulat dan `cardElevation="10dp"` untuk memberikan efek bayangan agar item terlihat menonjol. Line 16 sampai 20 terdapat

ConstraintLayout (Container dalam Card), layout fleksibel untuk mengatur posisi komponen UI berdasarkan kait (constraint) terhadap komponen lain atau parent-nya. Kemudian, line 22 sampai 31 menampilkan foto produk, dengan ukuran tetap `135dp x 140dp`, dan `scaleType="centerCrop"` agar gambar memenuhi tampilan dengan cropping bagian luar. Line 33 sampai 46, menampilkan nama tas. `textStyle="bold"`, `textSize="20sp"` untuk mengatur teks nya agar nama terlihat jelas, dan ditempatkan di sebelah kanan gambar. Lalu, line 48 sampai 58 disitu berfungsi untuk menampilkan harga tas dan ditempatkan di bawah nama produk. Line 60 sampai 68 membuat tombol untuk berpindah ke marketplace Shopee. Dengan posisi di bawah harga dan di sebelah kiri. Constraint ke `@id/tv_item_price` (top) dan Constraint ke `@id/tv_item_name` (start). Line 70 sampai 79 membuat tombol untuk navigasi ke halaman detail produk. Dengan posisi di sebelah kanan tombol Shopee, menyatu sejajar atasnya.

10. fragment_detail.xml

Line 1 sampai 9 terdapat `<ScrollView>` untuk membungkus seluruh isi layout agar bisa di-scroll jika kontennya lebih panjang dari layar. ini sangat berguna untuk tampilan detail yang bisa panjang, seperti pada deskripsi produk. Line 11 sampai 14, ConstraintLayout (di dalam ScrollView) digunakan untuk menyusun posisi elemen-elemen UI dengan fleksibel. Pada line 16 sampai 25 menampilkan gambar produk ukuran besar (350dp x 350dp). `centerCrop` agar gambar tetap proporsional memenuhi area dan diposisikan di bagian atas tengah layout. Kemudian, line 27 sampai 38 menampilkan nama produk dengan ukuran besar (`25sp`) dan bold. Diposisikan di bawah gambar dan disejajarkan ke tengah. Line 40 sampai 51, menampilkan harga dengan warna merah gelap (`holo_red_dark`) dan bold. Diposisikan di bawah nama produk dan tetap di tengah. Lalu, line 53 sampai 70 menampilkan deskripsi lengkap produk. `lineSpacingExtra="5dp"` mengatur agar teks lebih nyaman dibaca. `textAlignment="center"` digunakan untuk isi teks diratakan ke tengah. `padding` dan `marginTop` cukup besar untuk memberi jarak visual. Serta lebar penuh (`match_parent`) agar deskripsi tidak terpotong.

11. nav_graph.xml

Pada line 1 sampai 5, menentukan bahwa fragment pertama kali yang ditampilkan saat app berjalan adalah `HomeFragment`. Line 7 sampai 11, mewakili untuk tampilan utama (daftar tas). Fragment ini berada di package `com.example.modul3`. Layout yang digunakan ialah `fragment_home.xml`. Line 12 sampai 15, mendefinisikan rute navigasi dari `HomeFragment` ke `DetailFragment`, hal ini akan dipicu saat user klik tombol "Detail" pada item tas. Line 17 sampai 22 adalah fragment untuk menampilkan detail dari item tas yang dipilih, dengan menggunakan layout `fragment_detail.xml`.

12. strings.xml

Pada line 1 sampai 4 terdapat `<resources>`, yang di mana demua string dan array harus berada di dalam tag `<resources>` tersebut. Fungsi-fungsinya terdiri dari `app_name` itu nama aplikasi. `btn_detail` dan `btn_shopee` adalah label pada tombol navigasi atau detail dan tombol ke Shopee. `label_nama` dan `label_harga` untuk label yang muncul pada halaman detail produk. `placeholder_nama` dan `placeholder_harga` ialah placeholder atau teks default saat input kosong atau belum ada data. Line 6 sampai 17 terdapat `<string-array name="data_name">`, yang di mana pada array string tersebut berisi nama-nama tas dari dua merek, yaitu BOSTANTEN dan MOSSDOOM. Kemudian, line 19 sampai 30 array string untuk `data_photo`, yang berisi referensi ke gambar tas di folder `res/drawable`, dengan menggunakan notasi `@drawable/nama_gambar`. Line 32 sampai 43 terdapat array string untuk `data_link`, link akan menuju produk yang bersangkutan di marketplace Shopee. Lalu, line 45 sampai 56 terdapat array string `data_harga`, yang di mana akan menampilkan harga produk dalam Rupiah. Line 58 sampai 76 terdapat array string `data_deskripsi`, setiap `<item>` disitu menjelaskan fitur, ukuran, warna, dan target pengguna dari tas. Informasi tersebut sangat lengkap dan berorientasi pada e-commerce (penjelasan produk seperti di Shopee).

13. activity_main.xml

Pada line 1 sampai 9, `ConstraintLayout` adalah layout utama yang digunakan di sini untuk mengatur posisi elemen UI di dalamnya. `ConstraintLayout` memungkinkan penataan elemen UI dengan fleksibilitas tinggi melalui constraint (penghubung). `xmlns:android` adalah deklarasi namespace untuk atribut Android standar. `xmlns:tools` digunakan oleh Android Studio untuk memberikan data atau tips visual saat pengembangan tanpa mempengaruhi aplikasi yang sebenarnya. `tools:context=".MainActivity"` memberikan konteks di mana layout ini digunakan (dalam hal ini, `MainActivity`). `tools:ignore="ExtraText"` memberikan petunjuk kepada Android Studio untuk mengabaikan peringatan tertentu yang tidak penting (misalnya teks ekstra).

Kemudian, line 11 sampai 18 terdapat `FragmentContainerView`, yang di mana komponen ini digunakan untuk menampilkan dan mengelola fragment di dalam activity. `FragmentContainerView` menggantikan penggunaan `FrameLayout` untuk penempatan fragment dan lebih terintegrasi dengan Navigation Component Android. `android:id="@+id/nav_host_fragme"` memberikan ID untuk `FragmentContainerView`, yang nantinya bisa digunakan untuk referensi dalam kode.
`android:name="androidx.navigation.fragment.NavHostFragment"` untuk menyatakan bahwa fragment yang akan ditampilkan di dalam `FragmentContainerView` adalah `NavHostFragment`, yang merupakan bagian dari Navigation Component. Fragment ini bertugas mengelola navigasi antar fragment. `android:layout_width="match_parent"` dan `android:layout_height="match_parent"` digunakan untuk membuat `FragmentContainerView` mengambil ruang penuh dari layar, dengan lebar dan tinggi `match_parent`, sehingga mengisi seluruh layout. `app:navGraph="@navigation/nav_graph"` untuk menyediakan `NavGraph` yang berisi alur navigasi antar fragment. `@navigation/nav_graph` adalah referensi ke file `nav_graph.xml`, yang mendefinisikan destinasi dan aksi

antar fragment dalam aplikasi. `app:defaultNavHost="true"` digunakan untuk menandakan bahwa `NavHostFragment` ini adalah host utama untuk navigasi di activity ini. Artinya, fragment ini akan menangani back stack navigasi, dan tombol back perangkat akan berfungsi untuk kembali ke fragment sebelumnya (jika ada).

D. Tautan Git

Berikut adalah tautan untuk source code yang telah dibuat.

https://github.com/nrhikmahhh99/Pemrograman-Mobile

SOAL 2

Jelaskan Application class dalam arsitektur aplikasi Android dan fungsinya!

A. Pembahasan

Application adalah kelas dasar dari `android.app.Application` yang merepresentasikan global application state. Kelas ini hanya dibuat satu kali selama siklus hidup aplikasi, sebelum aktivitas, service, atau komponen lainnya dibuat. Android secara otomatis akan membuat instance dari kelas ini sebelum komponen pertama aplikasi dimulai (seperti `Activity`, `Service`, `BroadcastReceiver`, atau `ContentProvider`).

Berikut beberapa fungsi dan peran Application class dalam arsitektur:

1. Inisialisasi Global

   Tempat utama untuk menginisialisasi dependency global seperti Retrofit, Room, Firebase, atau Dagger/Hilt.
2. Dependency Injection (DI)

   Dipakai untuk menyediakan dependensi ke berbagai komponen tanpa harus inisialisasi berulang.
3. Penyimpanan State Global

   Menyimpan data yang harus hidup selama lifecycle aplikasi (jangan disalahgunakan untuk menyimpan UI state).
4. Konfigurasi Logging atau Monitoring

   Ideal untuk menginisialisasi alat log & monitoring seperti Timber, Crashlytics, Sentry, dan sebagainya.
5. Akses Context Aplikasi

   Karena Application adalah turunan dari Context, maka bisa menggunakannya di mana saja sebagai global context.

Application ini bekerja dengan cara Android Framework memanggil Application saat proses aplikasi dimulai. Hanya ada satu instance Application dalam satu proses aplikasi. Lalu, onCreate() dipanggil sekali, cocok untuk konfigurasi awal. Sehingga, kelas ini tetap hidup selama proses aplikasi masih berjalan (kecuali aplikasi dihentikan oleh sistem atau user).

Referensi:

https://developer.android.com/reference/android/app/Application

https://developer.android.com/guide/components/activities/activity-lifecycle?hl=id#app-startup

https://developer.android.com/topic/architecture?hl=id

https://stackoverflow.com/questions/3821784/whats-the-difference-between-n-and-r-n/3821817#3821817