LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 4

ViewModel and Debugging

Oleh:

Alya Rosaan NIM. 2310817320006

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
MEI 2024

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 Praktikum ini dikerjakan oleh:

Nama Praktikan : Alya Rosaan
NIM : 2310817320006

Menyetujui, Asisten Praktikum	Mengetahui, Dosen Penanggung Jawab Praktikum
Salsabila Syifa NIM. 2010817320004	Andreyan Rizky Baskara, S.Kom., M.Kom. NIP. 19930703 201903 01 011

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

Soal Praktikum

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 untuk membuat parameter dengan tipe data String di dalam ViewModel
c. Gunakan StateFlow untuk mengelola event onClick dan data list item dari ViewModel ke Fragment
d. Install dan gunakan library Timber untuk logging 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
    d. Gunakan tool Debugger di Android Studio untuk melakukan debugging pada aplikasi.
    e. Cari setidaknya satu breakpoint yang relevan dengan aplikasi. Lalu, gunakan fitur Step
    f. Into, Step Over, dan Step Out. Setelah itu, jelaskan fungsi Debugger, cara
    g. menggunakan Debugger, serta fitur Step Into, Step Over, dan Step Out

2. Jelaskan Application class dalam arsitektur aplikasi Android dan fungsinya Aplikasi harus dapat mempertahankan fitur-fitur yang sudah dibuat pada modul sebelumnya. Berikut adalah contoh debugging dalam Android Studio.

Jetpack Compose

A. Source Code

MainActivity.kt

```
1   package com.example.myanimelistapp
2
3   import android.content.Intent
4   import android.net.Uri
5   import android.os.Bundle
6   import androidx.activity.ComponentActivity
7   import androidx.activity.compose.setContent
8   import timber.log.Timber
9   import androidx.activity.enableEdgeToEdge
10  import androidx.compose.foundation.Image
11  import androidx.compose.foundation.layout.*
12  import androidx.compose.foundation.lazy.LazyColumn
13  import androidx.compose.foundation.lazy.items
14  import androidx.compose.foundation.shape.RoundedCornerShape
15  import androidx.compose.material3.*
```

```
16  import androidx.compose.runtime.Composable
17  import androidx.compose.runtime.collectAsState
18  import androidx.compose.runtime.getValue
19  import androidx.compose.ui.Modifier
20  import androidx.compose.ui.draw.clip
21  import androidx.compose.ui.graphics.Color
22  import androidx.compose.ui.layout.ContentScale
23  import androidx.compose.ui.platform.LocalContext
24  import androidx.compose.ui.res.painterResource
25  import androidx.compose.ui.text.style.TextOverflow
26  import androidx.compose.ui.tooling.preview.Preview
27  import androidx.compose.ui.unit.dp
28  import androidx.navigation.NavType
29  import androidx.navigation.NavHostController
30  import androidx.navigation.compose.*
31  import androidx.navigation.navArgument
32  import com.example.myanimelistapp.ui.theme.MyAnimeListAppTheme
33
34  class MainActivity : ComponentActivity() {
35      override fun onCreate(savedInstanceState: Bundle?) {
36          Timber.plant(Timber.DebugTree())
37          super.onCreate(savedInstanceState)
38          Timber.d("aplikasi dimulai")
39          enableEdgeToEdge()
40          setContent {
41              MyAnimeListAppTheme {
42                  val navController = rememberNavController()
43                  NavHost(navController = navController,
    startDestination = "list") {
44
45                      composable("list") {
46                          ItemList(navController = navController)
47                      }
48                      composable(
49                          route = "detail/{title}/{detail}/{imageRes}",
                            arguments = listOf(
50                              navArgument("title") { type =
    NavType.StringType },
51                              navArgument("detail") { type =
    NavType.StringType },
52                              navArgument("imageRes") { type =
    NavType.IntType }
53                          )
54                      ) { backStackEntry ->
55                          val title =
    backStackEntry.arguments?.getString("title") ?: ""
56                          val detail =
    backStackEntry.arguments?.getString("detail") ?: ""
57                          val imageRes =
    backStackEntry.arguments?.getInt("imageRes") ?: 0
58
```

```
59                          DetailScreen(title = title, detail = detail,
    imageRes = imageRes)
60                      }
61                  }
62              }
63          }
64      }
65  }
66
67
68  // === Data class ===
697 data class ListItem(
0       val title: String,
71      val subtitle: String,
72      val detail: String,
73      val imageRes: Int,
74      val url: String
75  )
76
77  // === Sample data ===
78  val sampleItems = listOf(
        ListItem("Naruto", "\"Naruto Uzumaki adalah seorang ninja remaja
    yang bercita-cita menjadi Hokage, pemimpin desa Konoha. Ia menyimpan
    kekuatan rubah berekor sembilan dalam tubuhnya, dan harus menghadapi
    penolakan, musuh kuat, serta perjalanan keras untuk membuktikan
    dirinya.\"\n", "\"Naruto Uzumaki adalah seorang ninja remaja yang
    bercita-cita menjadi Hokage, pemimpin desa Konoha. Ia menyimpan
    kekuatan rubah berekor sembilan dalam tubuhnya, dan harus menghadapi
    penolakan, musuh kuat, serta perjalanan keras untuk membuktikan
    dirinya.\"\n", R.drawable.naruto,
    "https://myanimelist.net/anime/20/Naruto"),
79      ListItem("One Piece", "\"Monkey D. Luffy berlayar bersama kru
    Bajak Laut Topi Jerami untuk mencari harta karun legendaris One
    Piece. Dengan kemampuan buah iblis dan semangat pantang menyerah,
    mereka menjelajahi lautan, melawan bajak laut, dan mengejar mimpi
    menjadi Raja Bajak Laut.\"\n", "\"Monkey D. Luffy berlayar bersama
    kru Bajak Laut Topi Jerami untuk mencari harta karun legendaris One
    Piece. Dengan kemampuan buah iblis dan semangat pantang menyerah,
    mereka menjelajahi lautan, melawan bajak laut, dan mengejar mimpi
    menjadi Raja Bajak Laut.\"\n", R.drawable.op,
    "https://myanimelist.net/anime/21/One_Piece"),
80      ListItem("Attack on Titan", "\"Dalam dunia di mana umat manusia
    terancam punah oleh para Titan pemakan manusia, Eren Yeager bersumpah
    untuk membalas dendam setelah desanya dihancurkan. Ia bergabung
    dengan militer dan mengungkap misteri kelam di balik keberadaan Titan
    dan sejarah manusia.\"\n", "\"Dalam dunia di mana umat manusia
    terancam punah oleh para Titan pemakan manusia, Eren Yeager bersumpah
    untuk membalas dendam setelah desanya dihancurkan. Ia bergabung
    dengan militer dan mengungkap misteri kelam di balik keberadaan Titan
    dan sejarah manusia.\"\n", R.drawable.aot,
    "https://myanimelist.net/anime/16498/Shingeki_no_Kyojin"),
81      ListItem("Demon Slayer", "\"Tanjiro Kamado menjadi pembasmi iblis
```

```
      setelah keluarganya dibantai dan adiknya berubah menjadi iblis.
      Bersama teman-temannya, Tanjiro menghadapi berbagai iblis berbahaya
      demi mencari cara menyelamatkan adiknya dan membalas kejahatan yang
      terjadi.\"\n", "\"Tanjiro Kamado menjadi pembasmi iblis setelah
      keluarganya dibantai dan adiknya berubah menjadi iblis. Bersama
      teman-temannya, Tanjiro menghadapi berbagai iblis berbahaya demi
      mencari cara menyelamatkan adiknya dan membalas kejahatan yang
      terjadi.\"\n", R.drawable.kny,
      "https://myanimelist.net/anime/38000/Kimetsu_no_Yaiba"),
82        ListItem("Jujutsu Kaisen", "\"Yuji Itadori, seorang siswa SMA
      dengan kekuatan fisik luar biasa, tanpa sengaja memakan jari iblis
      terkutuk Sukuna. Ia bergabung dengan sekolah Jujutsu untuk
      mengendalikan kekuatan itu dan melawan kutukan yang mengancam umat
      manusia.\"\n", "\"Yuji Itadori, seorang siswa SMA dengan kekuatan
      fisik luar biasa, tanpa sengaja memakan jari iblis terkutuk Sukuna.
      Ia bergabung dengan sekolah Jujutsu untuk mengendalikan kekuatan itu
      dan melawan kutukan yang mengancam umat manusia.\"\n",
      R.drawable.jjk, "https://myanimelist.net/anime/40748/Jujutsu_Kaisen")
      )

83    // === List Screen ===
84    @Composable
85    fun ItemList(navController: NavHostController) {
86        val context = LocalContext.current
87        val viewModel: AnimeViewModel =
      androidx.lifecycle.viewmodel.compose.viewModel(
88            factory = AnimeViewModelFactory("ItemList")
89        )
90        val itemList by viewModel.animeList.collectAsState()
91
92        LazyColumn(modifier = Modifier.padding(8.dp)) {
93            items(itemList) { item ->
94                Card(
95                    colors = CardDefaults.cardColors(
96                        containerColor = Color(0xFFF0F0F0) // warna latar
      Card
97                    ),
98                    shape = RoundedCornerShape(16.dp),
99                    modifier = Modifier
100                       .padding(8.dp)
101                       .fillMaxWidth()
102               ) {
103                   Row(modifier = Modifier.padding(16.dp)) {
104                       Image(
                              painter = painterResource(id =
105   item.imageRes),
106                           contentDescription = null,
107                           contentScale = ContentScale.Crop,
108                           modifier = Modifier
109                               .width(150.dp)
110                               .height(250.dp)
111                               .clip(RoundedCornerShape(12.dp))
```

```
112                     )
113                     Spacer(modifier = Modifier.width(16.dp))
114                     Column(
115                         modifier = Modifier
116                             .weight(1f)
117                             .fillMaxHeight(),
                        verticalArrangement =
118 Arrangement.spacedBy(8.dp)
                        ) {
119                         Text(item.title, style =
    MaterialTheme.typography.titleMedium)
120                         Text(item.subtitle, style =
    MaterialTheme.typography.bodySmall, maxLines = Int.MAX_VALUE,
121                             overflow = TextOverflow.Visible)
122
123                         Column(verticalArrangement =
    Arrangement.spacedBy(8.dp)) {
124                             Button(
125                                 onClick = {
126                                     viewModel.logItemClick(item,
    "Detail")
127                                     Timber.d("Navigasi ke detail:
    ${item.title}")
128
129 navController.navigate("detail/${item.title}/${item.detail}/${item.im
    ageRes}")
130                                 },
131                                 colors = ButtonDefaults.buttonColors(
                                        containerColor =
    Color(0xFF3A4C8B), // warna biru keunguan
132                                     contentColor = Color.White //
    teks putih
133                                 ),
134                                 modifier = Modifier.widthIn(min =
    100.dp, max = 140.dp)
135                             ) {
136                                 Text("Detail", style =
    MaterialTheme.typography.labelLarge)
137                             }
138
139                             Button(
140                                 onClick = {
141                                     viewModel.logItemClick(item,
    "Open URL")
142                                     Timber.d("Membuka URL:
    ${item.url}")
143                                     val intent =
    Intent(Intent.ACTION_VIEW, Uri.parse(item.url))
144                                     context.startActivity(intent)
145                                 },
                                    colors = ButtonDefaults.buttonColors(
                                        containerColor =
```

```
    Color(0xFF3A4C8B), // warna biru keunguan
146                                     contentColor = Color.White //
147 teks putih
148                                 ),
149                                 modifier = Modifier.widthIn(min =
    100.dp, max = 140.dp)
150                             ) {
151                                 Text("Open URL", style =
    MaterialTheme.typography.labelLarge)
152                             }
153                         }
154                     }
155                 }
156             }
157         }
158     }
160 }
161
162
163 // === Detail Screen ===
164 @Composable
165 fun DetailScreen(title: String, detail: String, imageRes: Int) {
166     Column(
167         modifier = Modifier
168             .fillMaxSize()
169             .padding(16.dp)
170     ) {
171         Image(
172             painter = painterResource(id = imageRes),
173             contentDescription = null,
174             contentScale = ContentScale.Crop,
175             modifier = Modifier
176                 .fillMaxWidth()
177                 .height(600.dp)
178                 .clip(RoundedCornerShape(12.dp))
179         )
180
181         Spacer(modifier = Modifier.height(16.dp))
182
183         Text(text = title, style =
    MaterialTheme.typography.headlineSmall)
184         Spacer(modifier = Modifier.height(8.dp))
185         Text(
186             text = detail,
187             style = MaterialTheme.typography.bodyLarge,
188             maxLines = Int.MAX_VALUE,
189             overflow = TextOverflow.Visible
190         )
191     }
192 }
193
194 @Preview(showBackground = true)
```

```
195 @Composable
195 fun DefaultPreview() {
196     MyAnimeListAppTheme {
197         ItemList(rememberNavController())
198     }
199 }
```

Tabel 1. Source Code Compose

AnimeViewModel.kt

```
1   package com.example.myanimelistapp
2
3   import androidx.lifecycle.ViewModel
4   import kotlinx.coroutines.flow.MutableStateFlow
5   import kotlinx.coroutines.flow.StateFlow
6   import kotlinx.coroutines.flow.asStateFlow
7   import timber.log.Timber
8
9   class AnimeViewModel(private val source: String) : ViewModel() {
10      private val _animeList =
11  MutableStateFlow<List<ListItem>>(emptyList())
12      val animeList: StateFlow<List<ListItem>> =
13  _animeList.asStateFlow()
14
15      private val _selectedItem = MutableStateFlow<ListItem?>(null)
16      val selectedItem: StateFlow<ListItem?> =
17  _selectedItem.asStateFlow()
18
19      init {
20          _animeList.value = sampleItems
21          Timber.d("Data item berhasil dimuat: ${sampleItems.size}
22  item")
23      }
24
25      fun onDetailClicked(item: ListItem) {
26          _selectedItem.value = item
27          Timber.d("[$source] Tombol Detail ditekan: ${item.title}")
28      }
29
30      fun onUrlClicked(item: ListItem) {
31          Timber.d("[$source] Tombol URL ditekan: ${item.title}")
32      }
33
34      fun logItemClick(item: ListItem, s: String) {
35
36      }
37  }
```

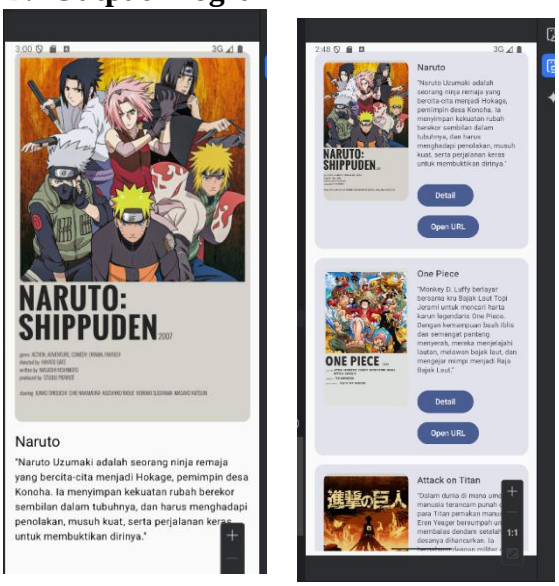
ViewModelFactory.kt

```
1   package com.example.myanimelistapp
2
3   import androidx.lifecycle.ViewModel
4   import androidx.lifecycle.ViewModelProvider
5
6   class AnimeViewModelFactory(private val source: String) :
7   ViewModelProvider.Factory {
8       override fun <T : ViewModel> create(modelClass: Class<T>): T {
9           if (modelClass.isAssignableFrom(AnimeViewModel::class.java))
10  {
11              return AnimeViewModel(source) as T
12          }
13          throw IllegalArgumentException("Unknown ViewModel class")
14      }
15  }
```

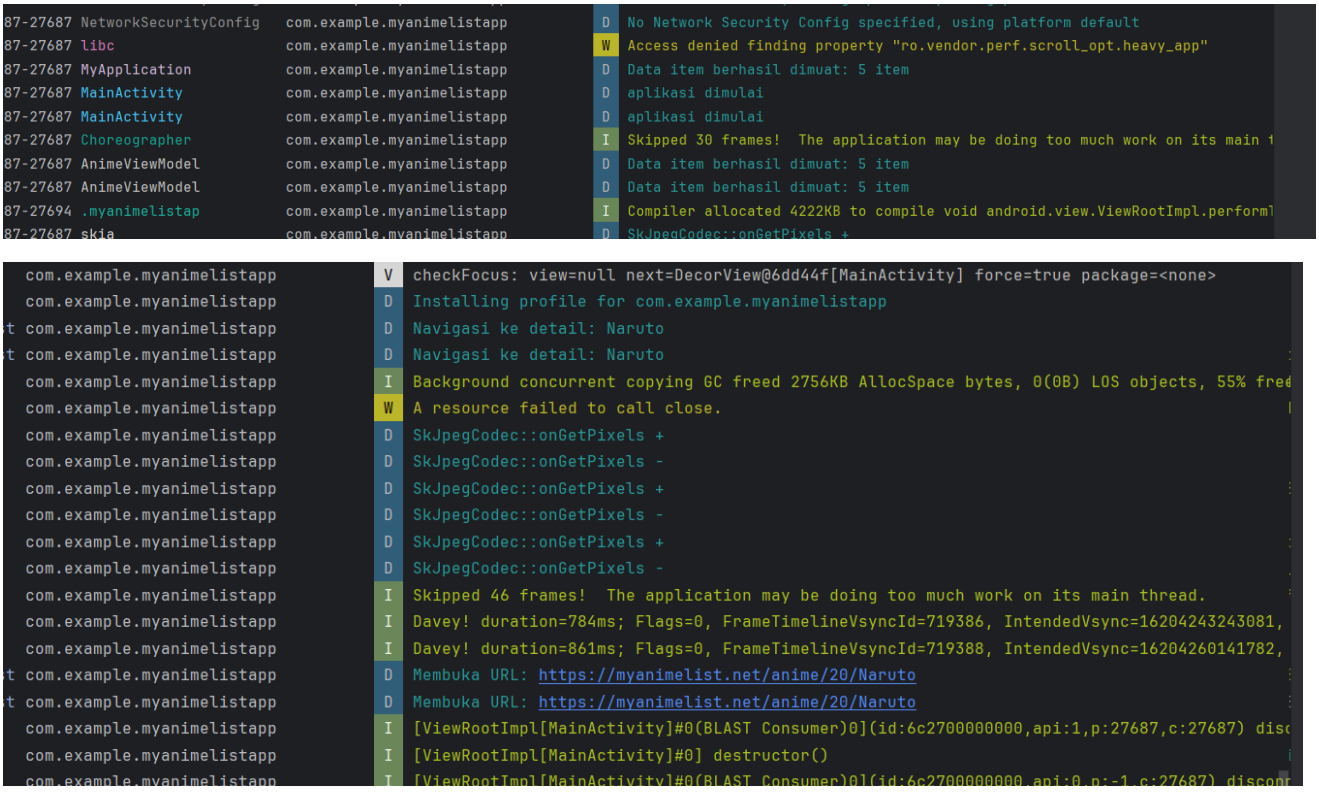
MyApplication.kt

```
1   package com.example.myanimelistapp
2
3   import android.app.Application
4   import timber.log.Timber
5
6   class MyApplication : Application() {
7       override fun onCreate() {
8           super.onCreate()
9           Timber.plant(Timber.DebugTree())
10          Timber.d("Data item berhasil dimuat: ${sampleItems.size}
11  item")
12      }
13  }
14
15
```

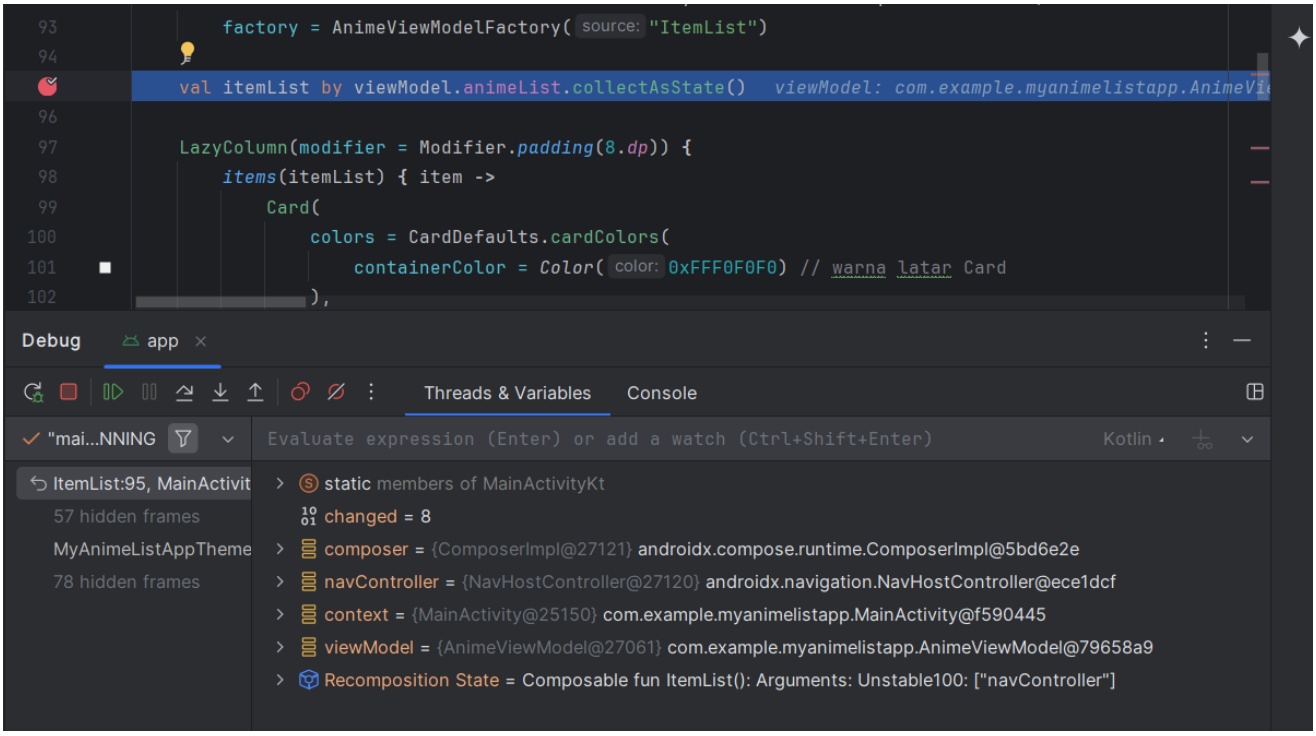
B. Output Program



Gambar 1 Screenshot Output Compose



Gambar 2 Screenshot Logging



Gambar 3 Screenshot Debugging

C. Pembahasan

MainActivity.kt:

1. Inisialisasi Timber di MainActivity.onCreate() Pada baris 11 :
   Timber.plant(Timber.DebugTree()) :
   Menanam (plant) DebugTree untuk mengaktifkan logging debug selama pengembangan.
   Pada baris 13 : Timber.d("aplikasi dimulai") :
   Menulis log debug "aplikasi dimulai" ke Logcat saat aplikasi mulai berjalan.
2. Logging di ItemList Composable (bagian tombol Detail), Pada baris 104:
   viewModel.logItemClick(item, "Detail")
   Pada baris 105: Timber.d("Navigasi ke detail: ${item.title}"
   logItemClick() di ViewModel (fungsi log khusus) dipanggil untuk mencatat klik.
   Dan Timber mencatat log debug navigasi ke detail anime dengan judul tertentu.
3. Logging di ItemList Composable (bagian tombol Open URL) Pada baris 116:
   viewModel.logItemClick(item, "Open URL")
   Pada baris 117: Timber.d("Membuka URL: ${item.url}") Log klik dibukanya URL dicatat oleh ViewModel. Dan Timber mencatat URL yang akan dibuka di browser.

AnimeViewModel.kt:

1. Pada Baris 15: Timber.d("Data item berhasil dimuat: ${sampleItems.size} item")
   Log ini menandakan data anime sudah berhasil dimuat ke StateFlow _animeList. Ini membantu memastikan data sudah siap dipakai di UI.

2. Pada Baris 20: Timber.d("[$source] Tombol Detail ditekan: ${item.title}")
   Ketika tombol Detail ditekan di UI, ViewModel mencatat log ini. [$source] adalah label yang berasal dari konstruktor ViewModel (misal "ItemList"), memudahkan kamu melacak asal event.
3. Pada Baris 24:
   Timber.d("[$source] Tombol URL ditekan: ${item.title}")
   Saat tombol Open URL ditekan, ViewModel juga mencatat klik ini. Memudahkan pelacakan interaksi user pada URL.
4. Pada Baris 27-29:
   Fungsi logItemClick(item: ListItem, s: String) belum diisi.
   Kamu bisa gunakan ini untuk menggabungkan logging detail dan url dalam satu fungsi, misalnya:
   ```
   fun logItemClick(item: ListItem, action: String) {
      Timber.d("[$source] Tombol $action ditekan: ${item.title}")
   }
   ```

AnimeViewModelFactory.kt:

1. Pada Baris 5: class AnimeViewModelFactory(private val source: String) : ViewModelProvider.Factory {
   Mendefinisikan kelas AnimeViewModelFactory yang mengimplementasikan interface ViewModelProvider.Factory.
   Tujuannya adalah membuat factory khusus yang dapat membuat instance AnimeViewModel dengan parameter source.
   source adalah properti yang diterima melalui konstruktor dan akan diteruskan ke AnimeViewModel.
2. Pada Baris 6: override fun <T : ViewModel> create(modelClass: Class<T>): T {
   Meng-override metode create dari interface ViewModelProvider.Factory.
   Metode ini bertugas membuat instance ViewModel yang diminta oleh framework Android.
   Parameter modelClass adalah tipe kelas ViewModel yang ingin dibuat.
3. Pada Baris 7: if (modelClass.isAssignableFrom(AnimeViewModel::class.java)) {
   Mengecek apakah kelas yang diminta (modelClass) adalah AnimeViewModel atau subclass-nya. Ini penting untuk memastikan factory ini hanya membuat AnimeViewModel.
4. Pada Baris 8: return AnimeViewModel(source) as T
   Jika benar kelas yang diminta adalah AnimeViewModel, buat instance baru AnimeViewModel dengan mengirimkan parameter source yang sudah disimpan di factory.
   Kemudian cast hasilnya menjadi tipe generic T yang diminta.
5. Pada Baris 10: throw IllegalArgumentException("Unknown ViewModel class")
   Jika kelas yang diminta bukan AnimeViewModel, lempar exception IllegalArgumentException untuk memberi tahu bahwa factory ini tidak dapat membuat kelas ViewModel lain.

XML

D. Source Code

MainActivity.kt

```
1  package com.example.myanimelistappxml
2
3  import android.content.Intent
4  import android.net.Uri
5  import android.os.Bundle
6  import androidx.activity.viewModels
7  import androidx.appcompat.app.AppCompatActivity
8  import androidx.lifecycle.lifecycleScope
9  import androidx.recyclerview.widget.LinearLayoutManager
10 import androidx.recyclerview.widget.RecyclerView
11 import com.example.myanimelistapp.AnimeAdapter
12 import com.example.myanimelistapp.MainViewModel
13 import kotlinx.coroutines.launch
14 import timber.log.Timber
15
16 class MainActivity : AppCompatActivity() {
17
18     private val viewModel: MainViewModel by viewModels()
19
20     override fun onCreate(savedInstanceState: Bundle?) {
21         Timber.d("MainActivity onCreate() dimulai")
22         super.onCreate(savedInstanceState)
23         setContentView(R.layout.activity_main)
24
25         val recyclerView: RecyclerView =
26 findViewById(R.id.recyclerView)
27         recyclerView.layoutManager =
28 LinearLayoutManager(this)
29
30         lifecycleScope.launch {
31             viewModel.animeList.collect { list ->
32                 Timber.d("List dimuat dengan ${list.size}
33 item:")
34                 list.forEach {
35                     Timber.d("- ${it.title}")
36                 }
37                 recyclerView.adapter = AnimeAdapter(
38                     list,
39                     onDetailClick = { item ->
40                         Timber.d("Tombol Detail diklik untuk:
41 ${item.title}")
42
43
```

```
                            val intent =
Intent(this@MainActivity, DetailActivity::class.java).apply {
                                putExtra("title", item.title)
                                putExtra("desc", item.detailDesc)
                                putExtra("imageResId",
item.imageResId)
                            }
                            startActivity(intent)
                        },
                        onOpenUrlClick = { item ->
                            Timber.d("Tombol (Open URL) diklik
untuk: ${item.title}")
                            val intent =
Intent(Intent.ACTION_VIEW, Uri.parse(item.url))
                            startActivity(intent)
                        }
                    )
                }
            }
        }
}
```

ListItem.kt

```
package com.example.myanimelistapp

data class ListItem(
    val title: String,
    val subtitle: String,
    val imageResId: Int,
    val detailDesc: String,
    val url: String
)
```

AnimeAdapter.kt

```
package com.example.myanimelistapp

import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.Button
import android.widget.ImageView
import android.widget.TextView
import androidx.recyclerview.widget.RecyclerView
import com.example.myanimelistappxml.R

```

```
class AnimeAdapter(
    private val animeList: List<ListItem>,
    private val onDetailClick: (ListItem) -> Unit,
    private val onOpenUrlClick: (ListItem) -> Unit
) : RecyclerView.Adapter<AnimeAdapter.AnimeViewHolder>() {

    class AnimeViewHolder(view: View) :
RecyclerView.ViewHolder(view) {
        val imgAnime: ImageView = view.findViewById(R.id.imgAnime)
        val tvTitle: TextView = view.findViewById(R.id.tvTitle)
        val tvSubtitle: TextView =
view.findViewById(R.id.tvSubtitle)
        val btnDetail: Button = view.findViewById(R.id.btnDetail)
        val btnOpenUrl: Button =
view.findViewById(R.id.btnOpenUrl)
    }

    override fun onCreateViewHolder(parent: ViewGroup, viewType:
Int): AnimeViewHolder {
        val view =
LayoutInflater.from(parent.context).inflate(R.layout.item_anime,
parent, false)
        return AnimeViewHolder(view)
    }

    override fun onBindViewHolder(holder: AnimeViewHolder,
position: Int) {
        val item = animeList[position]
        holder.imgAnime.setImageResource(item.imageResId)
        holder.tvTitle.text = item.title
        holder.tvSubtitle.text = item.subtitle
        holder.btnDetail.setOnClickListener { onDetailClick(item)
}
        holder.btnOpenUrl.setOnClickListener {
onOpenUrlClick(item) }
    }

    override fun getItemCount() = animeList.size
}
```

DetailActivity.kt

```
package com.example.myanimelistappxml

import android.os.Bundle
import android.widget.ImageView
import android.widget.TextView
import androidx.appcompat.app.AppCompatActivity
import timber.log.Timber

```

```
9   class DetailActivity : AppCompatActivity() {
10      override fun onCreate(savedInstanceState: Bundle?) {
11          super.onCreate(savedInstanceState)
12          setContentView(R.layout.activity_detail)
13
14          val img = findViewById<ImageView>(R.id.imgDetail)
15          val tvTitle = findViewById<TextView>(R.id.tvDetailTitle)
16          val tvDesc = findViewById<TextView>(R.id.tvDetailDesc)
17
18          val title = intent.getStringExtra("title")
19          val desc = intent.getStringExtra("desc")
20          val imageResId = intent.getIntExtra("imageResId", 0)
21
22          Timber.d("Berpindah ke Detail: $title")
23
24          tvTitle.text = title
25          tvDesc.text = desc
26          img.setImageResource(imageResId)
        }
    }
```

MainViewModel.kt

```
1   package com.example.myanimelistapp
2
3   import androidx.lifecycle.ViewModel
4   import com.example.myanimelistappxml.R
5   import kotlinx.coroutines.flow.MutableStateFlow
6   import kotlinx.coroutines.flow.StateFlow
7
8   class MainViewModel : ViewModel() {
9       private val _animeList = MutableStateFlow(listOf(
10          ListItem("Naruto", "Seorang ninja muda yang bermimpi
    menjadi Hokage", R.drawable.naruto,
11              "Naruto adalah anime populer tentang ninja muda dengan
12  semangat juang tinggi.", "https://naruto.com"),
13          ListItem("One Piece", "Monkey D. Luffy berlayar bersama
    kru Bajak Laut Topi Jerami untuk mencari harta karun legendaris
    One Piece. Dengan kemampuan buah iblis dan semangat pantang
    menyerah, mereka menjelajahi lautan, melawan bajak laut, dan
    mengejar mimpi menjadi Raja Bajak Laut.", R.drawable.op,
                "Monkey D. Luffy berlayar bersama kru Bajak Laut Topi
    Jerami untuk mencari harta karun legendaris One Piece. Dengan
    kemampuan buah iblis dan semangat pantang menyerah, mereka
    menjelajahi lautan, melawan bajak laut, dan mengejar mimpi menjadi
    Raja Bajak Laut.", "https://naruto.com"),
14          ListItem("Attack On Titan", "Dalam dunia di mana umat
    manusia terancam punah oleh para Titan pemakan manusia, Eren
    Yeager bersumpah untuk membalas dendam setelah desanya
    dihancurkan. Ia bergabung dengan militer dan mengungkap misteri
```

```
    kelam di balik keberadaan Titan dan sejarah manusia.",
    R.drawable.aot,
                "Dalam dunia di mana umat manusia terancam punah oleh
    para Titan pemakan manusia, Eren Yeager bersumpah untuk membalas
    dendam setelah desanya dihancurkan. Ia bergabung dengan militer
    dan mengungkap misteri kelam di balik keberadaan Titan dan sejarah
    manusia.",
    "https://myanimelist.net/anime/16498/Shingeki_no_Kyojin"),
15          ListItem("Demon Slayer", "Tanjiro Kamado menjadi pembasmi
    iblis setelah keluarganya dibantai dan adiknya berubah menjadi
    iblis. Bersama teman-temannya, Tanjiro menghadapi berbagai iblis
    berbahaya demi mencari cara menyelamatkan adiknya dan membalas
    kejahatan yang terjadi.", R.drawable.kny,
                "Tanjiro Kamado menjadi pembasmi iblis setelah
    keluarganya dibantai dan adiknya berubah menjadi iblis. Bersama
    teman-temannya, Tanjiro menghadapi berbagai iblis berbahaya demi
    mencari cara menyelamatkan adiknya dan membalas kejahatan yang
    terjadi.",
    "https://myanimelist.net/anime/38000/Kimetsu_no_Yaiba"),
16          ListItem("Jujutsu Kaisen", "Yuji Itadori, seorang siswa
    SMA dengan kekuatan fisik luar biasa, tanpa sengaja memakan jari
    iblis terkutuk Sukuna. Ia bergabung dengan sekolah Jujutsu untuk
    mengendalikan kekuatan itu dan melawan kutukan yang mengancam umat
    manusia.", R.drawable.jjk,
                "Yuji Itadori, seorang siswa SMA dengan kekuatan fisik
    luar biasa, tanpa sengaja memakan jari iblis terkutuk Sukuna. Ia
    bergabung dengan sekolah Jujutsu untuk mengendalikan kekuatan itu
    dan melawan kutukan yang mengancam umat manusia.",
    "https://myanimelist.net/anime/40748/Jujutsu_Kaisen"),
        ))
17      val animeList: StateFlow<List<ListItem>> = _animeList
18  }
```

MyApplication.kt

```
1   package com.example.myanimelistappxml
2
3   import android.app.Application
4   import timber.log.Timber
5
6   class MyApplication : Application() {
7       override fun onCreate() {
8           super.onCreate()
9           Timber.plant(Timber.DebugTree())
10
11
12      }
13  }
```

Activity_main.xml

```
<?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"
    android:background="#FFFFFF"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerView"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:padding="8dp"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        android:background="@android:color/transparent"/>

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

item_anime.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"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="8dp"
    card_view:cardCornerRadius="12dp"
    card_view:cardElevation="4dp"
    card_view:cardBackgroundColor="@color/card_light">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="#FFFFFF"
        android:orientation="horizontal"
        android:padding="12dp">

        <ImageView
            android:id="@+id/imgAnime"
            android:layout_width="150dp"
            android:layout_height="220dp"
            android:layout_marginEnd="12dp"
            android:scaleType="centerCrop"
            android:src="@drawable/naruto" />
```

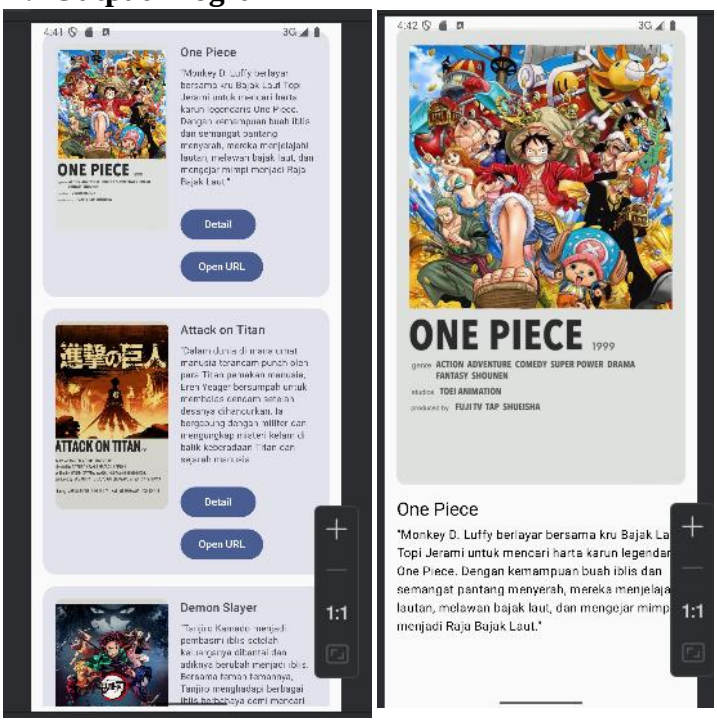
```
26
27          <LinearLayout
28              android:layout_width="0dp"
29              android:layout_height="wrap_content"
30              android:layout_weight="1"
31              android:orientation="vertical">
32
33              <TextView
34                  android:id="@+id/tvTitle"
35                  android:layout_width="wrap_content"
36                  android:layout_height="wrap_content"
37                  android:text="Judul Anime"
38                  android:textColor="#000000"
39                  android:textSize="18sp"
40                  android:textStyle="bold" />
41
42              <TextView
43                  android:id="@+id/tvSubtitle"
44                  android:layout_width="wrap_content"
45                  android:layout_height="wrap_content"
46                  android:layout_marginTop="4dp"
47                  android:text="Deskripsi singkat anime akan tampil
48  di sini..."
49                  android:textColor="#000000"
50                  android:textSize="14sp" />
51
52              <Button
53                  android:id="@+id/btnDetail"
54                  android:layout_width="wrap_content"
55                  android:layout_height="wrap_content"
56                  android:layout_marginEnd="8dp"
57                  android:backgroundTint="@color/blue_violet"
58                  android:text="Detail"
59                  android:textColor="@android:color/white" />
60
61              <Button
62                  android:id="@+id/btnOpenUrl"
63                  android:layout_width="wrap_content"
64                  android:layout_height="wrap_content"
65                  android:backgroundTint="@color/blue_violet"
66                  android:text="Open URL"
67                  android:textColor="@android:color/white" />
68
69          </LinearLayout>
70      </LinearLayout>
71  </androidx.cardview.widget.CardView>
```

activity_detail.xml

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <ScrollView
3   xmlns:android="http://schemas.android.com/apk/res/android"
4       android:layout_width="match_parent"
5       android:layout_height="match_parent"
6       android:padding="16dp"
7       android:background="@color/white">
8
9       <LinearLayout
10          android:orientation="vertical"
11          android:layout_width="match_parent"
12          android:layout_height="wrap_content"
13          android:background="@color/white">
14
15          <ImageView
16              android:id="@+id/imgDetail"
17              android:layout_width="match_parent"
18              android:layout_height="400dp"
19              android:scaleType="centerCrop"/>
20
21          <TextView
22              android:id="@+id/tvDetailTitle"
23              android:textSize="20sp"
24              android:textStyle="bold"
25              android:layout_marginTop="16dp"
26              android:layout_width="wrap_content"
27              android:layout_height="wrap_content"
28              android:textColor="@color/black"/>
29
30          <TextView
31              android:id="@+id/tvDetailDesc"
32              android:layout_width="wrap_content"
33              android:layout_height="wrap_content"
34              android:textColor="@color/black"/>
        </LinearLayout>
    </ScrollView>
```

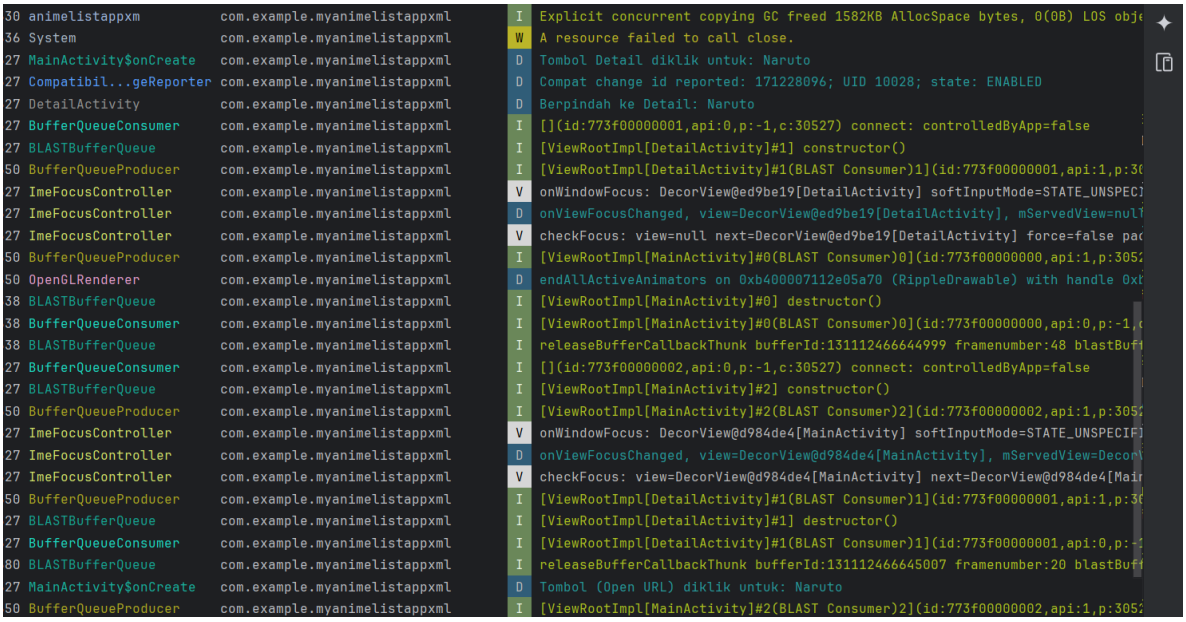
Tabel 2 Source Code XML

E. Output Program

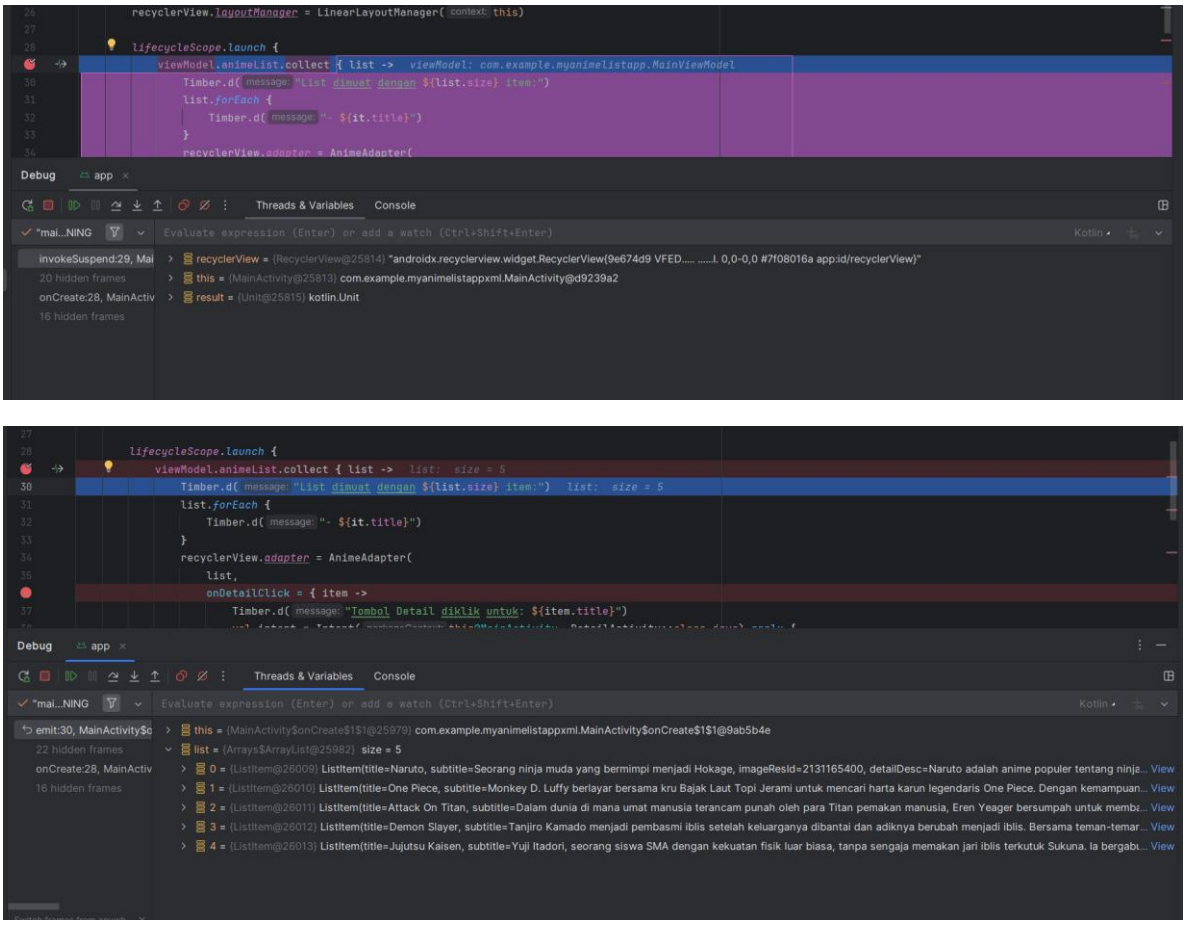


Gambar 4 Screenshot Output XML

```
4 30527-30527 libc                com.example.myanimelistappxml  W  Access denied finding property "ro.vendor.perf.scroll_opt.heavy_app"
8 30527-30527 AppCompatDelegate   com.example.myanimelistappxml  D  Checking for metadata for AppLocalesMetadataHolderService : Service not four
6 30527-30527 animelistappxm      com.example.myanimelistappxml  W  Accessing hidden method Landroid/view/View;->computeFitSystemWindows(Landro
6 30527-30527 animelistappxm      com.example.myanimelistappxml  W  Accessing hidden method Landroid/view/ViewGroup;->makeOptionalFitsSystemWin
2 30527-30527 MainActivity$onCreate com.example.myanimelistappxml  D  List dimuat dengan 5 item:
4 30527-30527 MainActivity$onCreate com.example.myanimelistappxml  D  - Naruto
6 30527-30527 MainActivity$onCreate com.example.myanimelistappxml  D  - One Piece
7 30527-30527 MainActivity$onCreate com.example.myanimelistappxml  D  - Attack On Titan
9 30527-30527 MainActivity$onCreate com.example.myanimelistappxml  D  - Demon Slayer
0 30527-30527 MainActivity$onCreate com.example.myanimelistappxml  D  - Jujutsu Kaisen
4 30527-30527 BufferQueueConsumer com.example.myanimelistappxml  I  [](id:773f00000000,api:0,p:-1,c:30527) connect: controlledByApp=false
7 30527-30527 BLASTBufferQueue    com.example.myanimelistappxml  I  [ViewRootImpl[MainActivity]#0] constructor()
2 30527-30550 hw-ProcessState     com.example.myanimelistappxml  D  Binder ioctl to enable oneway spam detection failed: Invalid argument
9 30527-30527 skia                com.example.myanimelistappxml  D  SkJpegCodec::onGetPixels +
```



Gambar 5 Screenshot logging XML



Gambar 6 Screenshot Debug XML

F. Pembahasan

MainActivity.kt

Inisialisasi dan Logging Awal

1. Baris 13: Timber.d("MainActivity onCreate() dimulai")
   Logging saat MainActivity dijalankan pertama kali.
2. Baris 22–23:
3. viewModel.animeList.collect { list ->
   Timber.d("List dimuat dengan ${list.size} item:")}
   Menandai bahwa data berhasil dikoleksi dan mencetak jumlah item.
4. Baris 24–25:
   list.forEach { Timber.d("- ${it.title}") }
   Mencetak setiap judul anime satu per satu ke Logcat.
5. Baris 31: Timber.d("Tombol Detail diklik untuk: ${item.title}")
   Logging saat user menekan tombol “Detail” pada suatu item.
6. Baris 39: Timber.d("Tombol (Open URL) diklik untuk: ${item.title}")
   Logging saat user menekan tombol “Open URL”.

MainViewModel.kt

1. Baris 1–2:
   package com.example.myanimelistapp
   import androidx.lifecycle.ViewModel
   Mendefinisikan lokasi dan mengimpor ViewModel dari AndroidX.
2. Baris 4–5:
   import kotlinx.coroutines.flow.MutableStateFlow
   import kotlinx.coroutines.flow.StateFlow
   Untuk aliran data reaktif dari ViewModel ke UI.
3. Baris 7: class MainViewModel : ViewModel()
   Membuat ViewModel untuk menyimpan dan mengatur list anime.
4. Baris 8–18:
   Inisialisasi _animeList berisi daftar objek ListItem, setiap objek berisi:
   - title
   - subtitle
   - imageResId
   - detailDesc
   - url
5. Baris 19: val animeList: StateFlow<List<ListItem>> = _animeList
   Mengungkap _animeList sebagai immutable StateFlow agar dapat diobservasi dari UI.

MyApplication.kt

Inisialisasi Logging (Global)

1. Baris 1–2:
   package com.example.myanimelistappxml
   import android.app.Application
   Menyatakan package dan mengimpor kelas dasar Application.
2. Baris 3: import timber.log.Timber
   Mengimpor Timber untuk keperluan logging.
3. Baris 5: class MyApplication : Application()
   Membuat kelas turunan dari Application, tempat inisialisasi global dilakukan.
4. Baris 6–8:
   override fun onCreate() {
   super.onCreate()
   Timber.plant(Timber.DebugTree())}
   - Baris 7: super.onCreate() memanggil inisialisasi bawaan Android.
   - Baris 8: Timber.plant(Timber.DebugTree())
     ➤ Mengaktifkan log debug Timber saat aplikasi dijalankan. Logging ini akan muncul di Logcat untuk debugging selama development.

AnimeViewModelFactory.kt

11. Baris 5: class AnimeViewModelFactory(private val source: String) : ViewModelProvider.Factory
    Factory untuk membuat AnimeViewModel dengan parameter sumber (source log).
12. Baris 7: if (modelClass.isAssignableFrom(AnimeViewModel::class.java))
    Mengecek apakah class yang diminta adalah AnimeViewModel.
13. Baris 8: return AnimeViewModel(source) as T
    Membuat instance AnimeViewModel dengan parameter sumber.
14. Baris 10: throw IllegalArgumentException("Unknown ViewModel class")
    Menangani jika ViewModel yang diminta bukan AnimeViewModel.

2. Jelaskan Application class dalam arsitektur aplikasi Android dan fungsinya

Application class adalah bagian khusus dalam aplikasi Android yang digunakan untuk:

- Menjalankan kode sekali saja saat aplikasi pertama kali dibuka (sebelum activity mana pun muncul).
- Menyimpan data atau pengaturan global yang ingin dipakai di seluruh aplikasi (misalnya: logging, library pihak ketiga, atau dependency injection).

- Cocok untuk inisialisasi seperti: Timber untuk debug, Retrofit, database, atau library lainnya.

Saat kita ingin melihat log debug dari awal aplikasi berjalan, kita bisa menanam Timber di Application class. Contohnya:

```
class MyApplication : Application() {
    override fun onCreate() {
        super.onCreate()
        Timber.plant(Timber.DebugTree()) // mengaktifkan log debug
    }
}
```

Dengan ini, kita bisa mencetak log ke Logcat dari bagian mana saja dalam aplikasi, misalnya saat tombol diklik, data diambil, atau halaman dibuka. Ini sangat membantu saat mencari kesalahan (bug).

G. Tautan Git

Berikut adalah tautan untuk source code yang telah dibuat.
https://github.com/ALYAROSAAN/Pemrograman-Mobile-Modul4.git