LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 4

ViewModel and Debugging

Oleh:

Muhammad Rizki Saputra NIM. 2310817310014

PROGRAM STUDI TEKNOLOGI INFORMASI FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
MEI 2024

LEMBAR PENGESAHAN
LAPORAN PRAKTIKUM PEMROGRAMAN MOBILE I
MODUL 4

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

Nama Praktikan : Muhammad Rizki Saputra
NIM : 2310817310014

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

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.

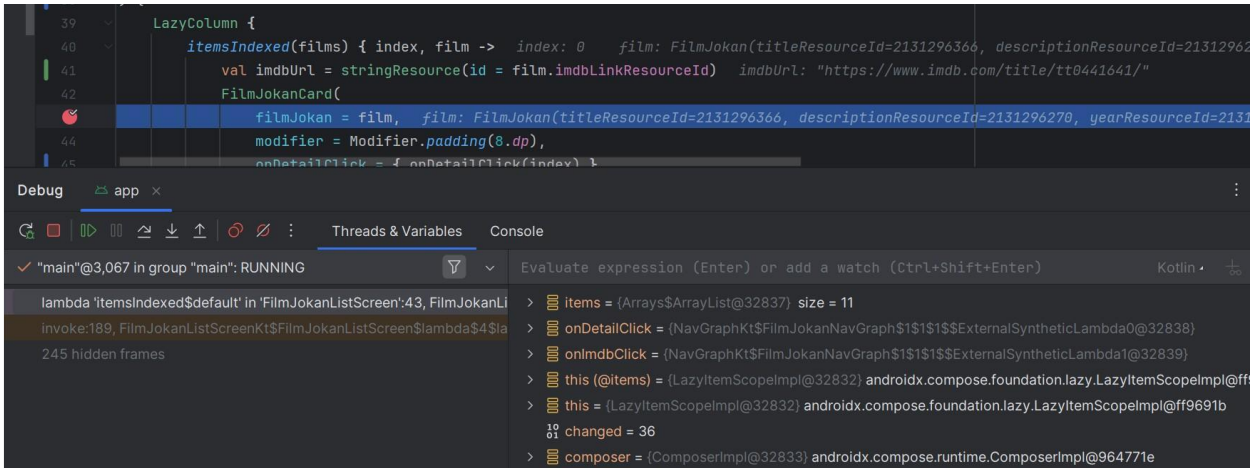
   b. Gunakan ViewModelFactory dalam pembuatan ViewModel

   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

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.



Gambar 1 Contoh penggunaan debugger

A. Source Code

- DinoRepositoryImpl.kt

```
1   package com.example.modul4.data.repository
2
3   import com.example.modul4.R
4   import com.example.modul4.domain.model.Dino
5   import com.example.modul4.domain.repository.DinoRepository
6   import kotlinx.coroutines.flow.Flow
7   import kotlinx.coroutines.flow.flowOf
8
9   class DinoRepositoryImpl : DinoRepository {
10
11      private val dinos = listOf(
12          Dino(
13              name = "Tyrannosaurus Rex",
14              description = """
15                  Tyrannosaurus Rex adalah predator terbesar di
16  zamannya, hidup sekitar 68 hingga 66 juta tahun yang lalu di
17  akhir periode Kapur. Dengan panjang mencapai 12 meter dan tinggi
18  hampir 4 meter, T-Rex memiliki rahang sangat kuat dan gigi
19  setajam pisau untuk merobek daging mangsanya.
20
21                  Meskipun tangan depannya kecil, tubuhnya yang
22  besar dan kekuatan gigitan luar biasa menjadikannya salah satu
23  dinosaurus paling menakutkan. Fosilnya ditemukan di Amerika
24  Utara dan menjadi ikon dalam dunia paleontologi.
25              """.trimIndent(),
26              shortDesc = "Predator terbesar di zamannya dengan
27  rahang yang sangat kuat.",
28              period = "Akhir Periode Kapur",
29              imageRes = R.drawable.tyrannosaurusrex,
30              type = "Karnivora",
31              wikiUrl =
32  "https://en.wikipedia.org/wiki/Tyrannosaurus"
33          ),
34          Dino(
35              name = "Triceratops",
36              description = """
37                  Triceratops adalah dinosaurus herbivora besar
38  dengan tiga tanduk dan pelindung tengkorak besar, hidup
39  berdampingan dengan T-Rex di akhir Kapur sekitar 68 juta tahun
40  yang lalu. Ia mencapai panjang sekitar 9 meter dan berat hingga
41  12 ton.
42
43                  Tanduk panjangnya diyakini digunakan untuk
44  pertahanan diri dan pertarungan sesama jantan saat musim kawin.
45  Struktur tengkoraknya juga mungkin berfungsi untuk pengaturan
46  suhu tubuh atau sebagai alat komunikasi visual.
47              """.trimIndent(),
48              shortDesc = "Herbivora besar dengan tiga tanduk
49
50
51
```

```
ikonik di wajahnya.",
            period = "Akhir Periode Kapur",
            imageRes = R.drawable.triceratops,
            type = "Herbivora",
            wikiUrl =
"https://en.wikipedia.org/wiki/Triceratops"
        ),
        Dino(
            name = "Velociraptor",
            description = """
                Velociraptor adalah dinosaurus kecil dan
karnivora yang terkenal karena kecepatannya dan diduga memiliki
bulu. Ia hidup sekitar 75 hingga 71 juta tahun yang lalu di
wilayah yang kini merupakan Mongolia.

                Meski hanya seukuran kalkun, Velociraptor sangat
gesit dan cerdas, serta berburu dalam kelompok. Cakar melengkung
pada kaki belakangnya digunakan untuk mencengkram dan
melumpuhkan mangsa.
            """.trimIndent(),
            shortDesc = "Karnivora kecil yang terkenal karena
kecepatan dan kecerdasannya.",
            period = "75-71 Juta Tahun Lalu",
            imageRes = R.drawable.velociraptor,
            type = "Karnivora",
            wikiUrl =
"https://en.wikipedia.org/wiki/Velociraptor"
        ),
        Dino(
            name = "Stegosaurus",
            description = """
                Stegosaurus adalah dinosaurus herbivora dari
periode Jurassic akhir, terkenal dengan piring tulang besar di
punggung dan ekor berduri yang disebut thagomizer. Ia hidup
sekitar 155 juta tahun lalu dan memiliki otak relatif kecil.

                Piring punggungnya kemungkinan digunakan untuk
menakuti pemangsa, menarik pasangan, atau mengatur suhu tubuh.
Ekor berdurinya menjadi senjata efektif melawan predator seperti
Allosaurus.
            """.trimIndent(),
            shortDesc = "Herbivora dengan piring tulang besar di
punggung dan ekor berduri.",
            period = "Periode Jurassic Akhir",
            imageRes = R.drawable.stegosaurus,
            type = "Herbivora",
            wikiUrl =
```

```
"https://en.wikipedia.org/wiki/Stegosaurus"
        ),
        Dino(
            name = "Brachiosaurus",
            description = """
                Brachiosaurus adalah salah satu sauropoda
terbesar yang hidup sekitar 154 hingga 150 juta tahun lalu pada
periode Jurassic. Ciri khasnya adalah leher panjang yang
memungkinkan ia meraih daun dari pohon tinggi.

                Berbeda dari sauropoda lain, kaki depan
Brachiosaurus lebih panjang daripada kaki belakang, membuat
posturnya menjulang tinggi. Ia diperkirakan memiliki berat
hingga 40 ton dan panjang lebih dari 25 meter.
            """.trimIndent(),
            shortDesc = "Sauropoda raksasa dengan leher sangat
panjang dan kaki depan yang tinggi.",
            period = "Periode Jurassic",
            imageRes = R.drawable.brachiosaurus,
            type = "Herbivora",
            wikiUrl =
"https://en.wikipedia.org/wiki/Brachiosaurus"
        ),
        Dino(
            name = "Spinosaurus",
            description = """
                Spinosaurus adalah dinosaurus karnivora terbesar
yang hidup sekitar 112–93 juta tahun lalu di Afrika Utara. Ciri
utamanya adalah layar punggung tinggi seperti layar kapal,
kemungkinan digunakan untuk menampilkan diri atau mengatur suhu.

                Ia diyakini sebagai semi-akuatik, berburu ikan
dan hewan air lainnya di sungai purba. Dengan panjang mencapai
15 meter, Spinosaurus lebih besar dari T-Rex dan sangat adaptif
terhadap lingkungan air.
            """.trimIndent(),
            shortDesc = "Karnivora semi-akuatik dengan layar
punggung yang khas.",
            period = "112-93 Juta Tahun Lalu",
            imageRes = R.drawable.spinosaurus,
            type = "Karnivora",
            wikiUrl =
"https://en.wikipedia.org/wiki/Spinosaurus"
        ),
        Dino(
            name = "Ankylosaurus",
            description = """
                Ankylosaurus adalah dinosaurus herbivora dari
akhir periode Kapur, terkenal karena tubuhnya yang dilapisi zira
dan ekor besar berbentuk gada. Panjangnya mencapai 6 hingga 8


```

```
meter dan berat hingga 8 ton.

                Bentuk tubuhnya seperti tank membuatnya hampir
kebal terhadap serangan predator. Ekor berotot dan keras dapat
digunakan untuk menyerang balik dan melukai lawan secara fatal.
            """.trimIndent(),
            shortDesc = "Herbivora berlapis zirah dengan ekor
besar berbentuk gada.",
            period = "Akhir Periode Kapur",
            imageRes = R.drawable.ankylosaurus,
            type = "Herbivora",
            wikiUrl =
"https://en.wikipedia.org/wiki/Ankylosaurus"
        ),
        Dino(
            name = "Allosaurus",
            description = """
                Allosaurus adalah predator utama dari periode
Jurassic, hidup sekitar 155 hingga 145 juta tahun lalu. Ia
memiliki tengkorak besar, gigi tajam, dan tubuh ramping yang
memungkinkannya berburu dengan kecepatan tinggi.

                Allosaurus sering disebut 'singa Jurassic'
karena perannya sebagai pemburu puncak. Ia mungkin berburu dalam
kelompok dan mangsanya termasuk sauropoda besar seperti
Diplodocus dan Camarasaurus.
            """.trimIndent(),
            shortDesc = "Predator puncak dari periode Jurassic,
sering disebut 'singa Jurassic'.",
            period = "Periode Jurassic Akhir",
            imageRes = R.drawable.allosaurus,
            type = "Karnivora",
            wikiUrl = "https://en.wikipedia.org/wiki/Allosaurus"
        ),
        Dino(
            name = "Diplodocus",
            description = """
                Diplodocus adalah sauropoda raksasa dari periode
Jurassic, dikenal karena leher dan ekor super panjang. Panjang
tubuhnya bisa mencapai 27 meter, menjadikannya salah satu
dinosaurus terpanjang yang pernah hidup.

                Ia hidup di Amerika Utara sekitar 154 juta tahun
lalu dan memakan tumbuhan rendah di hutan purba. Ekor panjangnya
kemungkinan digunakan untuk pertahanan atau komunikasi sonik
seperti cambuk.
            """.trimIndent(),
            shortDesc = "Salah satu dinosaurus terpanjang dengan
```

```
leher dan ekor super panjang.",
            period = "Periode Jurassic",
            imageRes = R.drawable.diplodocus,
            type = "Herbivora",
            wikiUrl = "https://en.wikipedia.org/wiki/Diplodocus"
        ),
        Dino(
            name = "Parasaurolophus",
            description = """
                Parasaurolophus adalah dinosaurus herbivora dari
akhir Kapur yang dikenal karena jambul panjang berongga di
kepalanya. Jambul tersebut kemungkinan digunakan untuk
menghasilkan suara, menarik pasangan, atau membantu mengatur
suhu tubuh.

                Ia berjalan dengan dua atau empat kaki dan hidup
dalam kawanan. Panjang tubuhnya sekitar 10 meter dan dikenal
sebagai bagian dari kelompok hadrosaur atau dinosaurus bebek.
            """.trimIndent(),
            shortDesc = "Herbivora dengan jambul panjang
berongga yang khas di kepalanya.",
            period = "Akhir Periode Kapur",
            imageRes = R.drawable.parasaurolophus,
            type = "Herbivora",
            wikiUrl =
"https://en.wikipedia.org/wiki/Parasaurolophus"
        )
    )

    fun getDinoList(): List<Dino> = dinos

    fun getDinoByName(name: String): Dino? =
        dinos.find { it.name.equals(name, ignoreCase = true) }

    fun getDinoFlow(): Flow<List<Dino>> = flowOf(dinos)

    override fun getAllDinos(): Flow<List<Dino>> = flowOf(dinos)

    override fun searchDinos(query: String): Flow<List<Dino>> =
        flowOf(dinos.filter { it.name.contains(query, ignoreCase
= true) })
}
```

Table 1 Source Code DinoRepositoryImpl

- Dino.kt

```
1   package com.example.modul4.domain.model
2
3   data class Dino(
4       val name: String,
5       val description: String,
6       val shortDesc: String,
7       val period: String,
8       val imageRes: Int,
9       val type: String,
10      val wikiUrl: String
11  )
```

Table 2 Source Code Dino.kt

- DinoRepository.kt

```
1   package com.example.modul4.domain.repository
2
3   import com.example.modul4.domain.model.Dino
4   import kotlinx.coroutines.flow.Flow
5
6   interface DinoRepository {
7       fun getAllDinos(): Flow<List<Dino>>
8       fun searchDinos(query: String): Flow<List<Dino>>
9   }
10
11
```

Table 3 Source Code DInoRepository.kt

- GetDinoListUseCase.kt

```
1   package com.example.modul4.domain.usecase
2
3   import com.example.modul4.domain.model.Dino
4   import com.example.modul4.domain.repository.DinoRepository
5   import kotlinx.coroutines.flow.Flow
6
7   class SearchDinosUseCase(private val repository: DinoRepository)
8   {
9       operator fun invoke(query: String): Flow<List<Dino>> =
10          repository.searchDinos(query)
11
12
```

```
}
}
```

Table 4 Source Code GetDinoListUseCase.kt

- DinoApp.kt

```
1  package com.example.modul4.ui.theme.presentation
2
3  import androidx.compose.runtime.Composable
4  import androidx.lifecycle.viewmodel.compose.viewModel
5  import androidx.navigation.NavHostController
6  import androidx.navigation.compose.NavHost
7  import androidx.navigation.compose.composable
8  import androidx.navigation.compose.rememberNavController
9  import com.example.modul4.presentation.ui.screen.DinoDetailScreen
10 import com.example.modul4.presentation.ui.screen.DinoListScreen
11 import com.example.modul4.presentation.viewmodel.DinoViewModel
12 import
13 com.example.modul4.presentation.viewmodel.DinoViewModelFactory
14
15 @Composable
16 fun DinoApp() {
17
18     val navController: NavHostController =
19 rememberNavController()
20
21     val viewModel: DinoViewModel = viewModel(factory =
22 DinoViewModelFactory())
23
24
25     NavHost(navController = navController, startDestination =
26 "dino_list") {
27         composable("dino_list") {
28             DinoListScreen(navController = navController,
29 viewModel = viewModel)
30         }
31
32         composable("detail") {
33             DinoDetailScreen(viewModel = viewModel, navController
34 = navController)
35         }
36     }
37 }
38
```

Table 5 Source Code DinoApp.kt

- DinoDetailScreen.kt

```
1  package com.example.modul4.presentation.ui.screen
2
3  import androidx.compose.foundation.Image
4  import androidx.compose.foundation.background
5  import androidx.compose.foundation.layout.*
6  import androidx.compose.foundation.rememberScrollState
7  import androidx.compose.foundation.shape.RoundedCornerShape
8  import androidx.compose.foundation.verticalScroll
9  import androidx.compose.material3.Button
10 import androidx.compose.material3.ButtonDefaults
11 import androidx.compose.material3.Text
12 import androidx.compose.runtime.Composable
13 import androidx.compose.runtime.collectAsState
14 import androidx.compose.runtime.getValue
15 import androidx.compose.ui.Alignment
16 import androidx.compose.ui.Modifier
17 import androidx.compose.ui.draw.clip
18 import androidx.compose.ui.graphics.Color
19 import androidx.compose.ui.res.painterResource
20 import androidx.compose.ui.text.font.FontWeight
21 import androidx.compose.ui.text.style.TextAlign
22 import androidx.compose.ui.unit.dp
23 import androidx.compose.ui.unit.sp
24 import androidx.navigation.NavController
25 import com.example.modul4.presentation.viewmodel.DinoViewModel
26
27 @Composable
28 fun DinoDetailScreen(
29     viewModel: DinoViewModel,
30     navController: NavController
31 ) {
32     val selectedDino by viewModel.selectedDino.collectAsState()
33
34     if (selectedDino == null) {
35         Box(modifier = Modifier.fillMaxSize(), contentAlignment =
36 Alignment.Center) {
37             Text("Data tidak tersedia")
38         }
39         return
40     }
41
42     val dino = selectedDino!!
43
44     Column(
45         modifier = Modifier
46             .fillMaxSize()
47             .background(Color(0xFFF4F1DE))
48             .verticalScroll(rememberScrollState())
49             .padding(16.dp)
50     ) {
51         Image(
             painter = painterResource(id = dino.imageRes),
```

```
            contentDescription = dino.name,
            modifier = Modifier
                .fillMaxWidth()
                .height(250.dp)
                .clip(RoundedCornerShape(20.dp))
        )
        Spacer(modifier = Modifier.height(16.dp))
        Text(
            text = dino.name,
            fontSize = 24.sp,
            fontWeight = FontWeight.Bold,
            textAlign = TextAlign.Center,
            color = Color(0xFF3D405B),
            modifier = Modifier.fillMaxWidth()
        )
        Spacer(modifier = Modifier.height(12.dp))
        Text(
            text = dino.description,
            fontSize = 14.sp,
            textAlign = TextAlign.Justify,
            lineHeight = 20.sp,
            color = Color(0xFF3D405B)
        )
        Spacer(modifier = Modifier.height(24.dp))
        Button(
            onClick = { navController.popBackStack() },
            modifier =
Modifier.align(Alignment.CenterHorizontally),
            colors = ButtonDefaults.buttonColors(containerColor =
Color(0xFFE07A5F))
        ) {
            Text("Kembali", color = Color.White)
        }
        Spacer(modifier = Modifier.height(16.dp))
    }
}
```

Table 6 Source Code DinoDetailScreen.kt

- DinoListScreen.kt

```
1  package com.example.modul4.presentation.ui.screen
2
3  import android.content.Intent
4  import android.net.Uri
5  import android.util.Log
6  import androidx.compose.foundation.Image
7  import androidx.compose.foundation.background
8  import androidx.compose.foundation.layout.*
9  import androidx.compose.foundation.lazy.grid.GridCells
10 import androidx.compose.foundation.lazy.grid.LazyVerticalGrid
11 import androidx.compose.foundation.lazy.grid.items
12 import androidx.compose.foundation.shape.RoundedCornerShape
13 import androidx.compose.material3.*
14 import androidx.compose.runtime.*
15 import androidx.compose.ui.Alignment
16 import androidx.compose.ui.Modifier
17 import androidx.compose.ui.draw.clip
18 import androidx.compose.ui.graphics.Color
19 import androidx.compose.ui.layout.ContentScale
20 import androidx.compose.ui.platform.LocalContext
21 import androidx.compose.ui.res.painterResource
22 import androidx.compose.ui.text.font.FontWeight
23 import androidx.compose.ui.unit.dp
24 import androidx.compose.ui.unit.sp
25 import androidx.navigation.NavController
26 import com.example.modul4.presentation.viewmodel.DinoViewModel
27
28 @Composable
29 fun DinoListScreen(
30     navController: NavController,
31     viewModel: DinoViewModel
32 ) {
33     val context = LocalContext.current
34     var searchQuery by remember { mutableStateOf("") }
35     var selectedFilter by remember { mutableStateOf("Semua") }
36     val types = listOf("Semua", "Herbivora", "Karnivora")
37     val dinoList by viewModel.dinoList.collectAsState()
38
39     val filteredList = dinoList.filter {
40         it.name.contains(searchQuery, ignoreCase = true) &&
41                 (selectedFilter == "Semua" || it.type ==
42 selectedFilter)
43     }
44
45     Column(
46         modifier = Modifier
47             .background(Color(0xFFF4F1DE))
48             .fillMaxSize()
49     ) {
```

```
50          Box(
51              modifier = Modifier
52                  .fillMaxWidth()
53                  .background(Color(0xFF81B29A))
54                  .padding(16.dp)
55          ) {
56              Text(
57                  text = "Jenis-Jenis Dinosaurus",
58                  fontSize = 24.sp,
59                  fontWeight = FontWeight.Bold,
60                  color = Color.White,
61                  modifier = Modifier.align(Alignment.CenterStart)
62              )
63          }
64
65
66          OutlinedTextField(
67              value = searchQuery,
68              onValueChange = { searchQuery = it },
69              label = { Text("Cari dinosaurus...") },
70              shape = RoundedCornerShape(24.dp),
71              modifier = Modifier
72                  .fillMaxWidth()
73                  .padding(16.dp)
74          )
75
76
77          Row(
78              modifier = Modifier
79                  .fillMaxWidth()
80                  .padding(horizontal = 16.dp),
81              horizontalArrangement = Arrangement.SpaceBetween,
82              verticalAlignment = Alignment.CenterVertically
83          ) {
84              Text("Filter berdasarkan tipe:", fontSize = 14.sp,
85  color = Color(0xFF3D405B))
86              DropdownMenuBox(types, selectedFilter) {
87  selectedFilter = it }
88          }
89
90          LazyVerticalGrid(
91              columns = GridCells.Fixed(2),
92              contentPadding = PaddingValues(16.dp),
93              verticalArrangement = Arrangement.spacedBy(12.dp),
94              horizontalArrangement = Arrangement.spacedBy(12.dp),
95              modifier = Modifier.fillMaxSize()
96          ) {
97              items(filteredList) { dino ->
98                  Card(
99                      modifier = Modifier.fillMaxWidth(),
100                     colors =
CardDefaults.cardColors(containerColor = Color(0xFFFFF7F0)),
```

```
                    elevation =
CardDefaults.cardElevation(defaultElevation = 4.dp),
                    shape = RoundedCornerShape(16.dp)
                ) {
                    Column {
                        Image(
                            painter = painterResource(id =
dino.imageRes),
                            contentDescription = dino.name,
                            contentScale = ContentScale.Crop,
                            modifier = Modifier
                                .fillMaxWidth()
                                .height(140.dp)
                                .clip(RoundedCornerShape(topStart
= 16.dp, topEnd = 16.dp))
                        )
                        Column(modifier = Modifier.padding(8.dp))
{
                            Spacer(modifier =
Modifier.height(6.dp))
                            Text(
                                dino.name,
                                fontSize = 16.sp,
                                fontWeight = FontWeight.Bold,
                                color = Color(0xFF3D405B)
                            )
                            Spacer(modifier =
Modifier.height(6.dp))

                            Row(
                                modifier =
Modifier.fillMaxWidth(),
                                horizontalArrangement =
Arrangement.SpaceBetween
                            ) {
                                Button(
                                    onClick = {
                                        Log.d("DinoListScreen",
"Tombol Detail ditekan untuk: ${dino.name}")

viewModel.selectDino(dino)

navController.navigate("detail")
                                    },
                                    colors =
ButtonDefaults.buttonColors(containerColor = Color(0xFF81B29A)),
                                    shape =
RoundedCornerShape(12.dp),
                                    modifier =
Modifier.weight(1f)
                                ) {
                                    Text("Info Detail", color =
```

```
Color.White, fontSize = 12.sp)
                                }

                                Spacer(modifier =
Modifier.width(8.dp))

                                Button(
                                    onClick = {
                                        Log.d("DinoListScreen",
"Tombol Wikipedia ditekan untuk: ${dino.name}")
                                        val intent =
Intent(Intent.ACTION_VIEW, Uri.parse(dino.wikiUrl))
16
165 context.startActivity(intent)
166                                     },
167                                     colors =
168 ButtonDefaults.buttonColors(containerColor = Color(0xFFE07A5F)),
169                                     shape =
170 RoundedCornerShape(12.dp),
171                                     modifier =
172 Modifier.weight(1f)
173                                 ) {
174                                     Text("Wikipedia", color =
175 Color.White, fontSize = 12.sp)
176                                 }
177                             }
178                         }
179                     }
180                 }
181             }
182         }
183     }
184 }
185
186
187 @Composable
188 fun DropdownMenuBox(
189     options: List<String>,
190     selectedOption: String,
191     onOptionSelected: (String) -> Unit
192 ) {
193     var expanded by remember { mutableStateOf(false) }
194
195     Box {
196         Button(
197             onClick = { expanded = true },
198             colors = ButtonDefaults.buttonColors(containerColor =
199 Color(0xFF81B29A))
200         ) {
201             Text(selectedOption, color = Color.White)
202         }
```

```
203         DropdownMenu(
204             expanded = expanded,
205             onDismissRequest = { expanded = false }
206         ) {
207             options.forEach { label ->
208                 DropdownMenuItem(
209                     text = { Text(label) },
210                     onClick = {
211                         onOptionSelected(label)
212                         expanded = false
213                     }
214                 )
215             }
216         }
217     }
218 }
219
```

Table 7 Source Code DinoListScreen.kt

- DinoViewModel,kt

```
1  package com.example.modul4.presentation.viewmodel
2
3  import android.util.Log
4  import androidx.lifecycle.ViewModel
5  import androidx.lifecycle.viewModelScope
6  import com.example.modul4.data.repository.DinoRepositoryImpl
7  import com.example.modul4.domain.model.Dino
8  import kotlinx.coroutines.flow.MutableStateFlow
9  import kotlinx.coroutines.flow.StateFlow
10 import kotlinx.coroutines.launch
11
12
13 class DinoViewModel : ViewModel() {
14
15     private val repository = DinoRepositoryImpl()
16
17     private val _dinoList =
18 MutableStateFlow<List<Dino>>(emptyList())
19     val dinoList: StateFlow<List<Dino>> = _dinoList
20
21     init {
22         viewModelScope.launch {
23             repository.getAllDinos().collect {
24                 _dinoList.value = it
25                 Log.d("DinoViewModel", "Data dinosaurus berhasil
26 dimuat: ${it.size} item.")
27             }
28         }
29     }
```

```
30
31      private val _selectedDino = MutableStateFlow<Dino?>(null)
32      val selectedDino: StateFlow<Dino?> = _selectedDino
33      fun selectDino(dino: Dino) {
34          _selectedDino.value = dino
35          Log.d("DinoViewModel", "Dino dipilih: ${dino.name}")
36      }
37  }
38
```

Table 8 SOurce Code DinoViewModel.k

- DinoViewModelFactory.kt

```
1   package com.example.modul4.presentation.viewmodel
2
3   import androidx.lifecycle.ViewModel
4   import androidx.lifecycle.ViewModelProvider
5
6   class DinoViewModelFactory : ViewModelProvider.Factory {
7       override fun <T : ViewModel> create(modelClass: Class<T>): T
8   {
9           if
10  (modelClass.isAssignableFrom(DinoViewModel::class.java)) {
11              @Suppress("UNCHECKED_CAST")
12              return DinoViewModel() as T
13          }
14          throw IllegalArgumentException("Unknown ViewModel class")
15      }
16  }
17
```

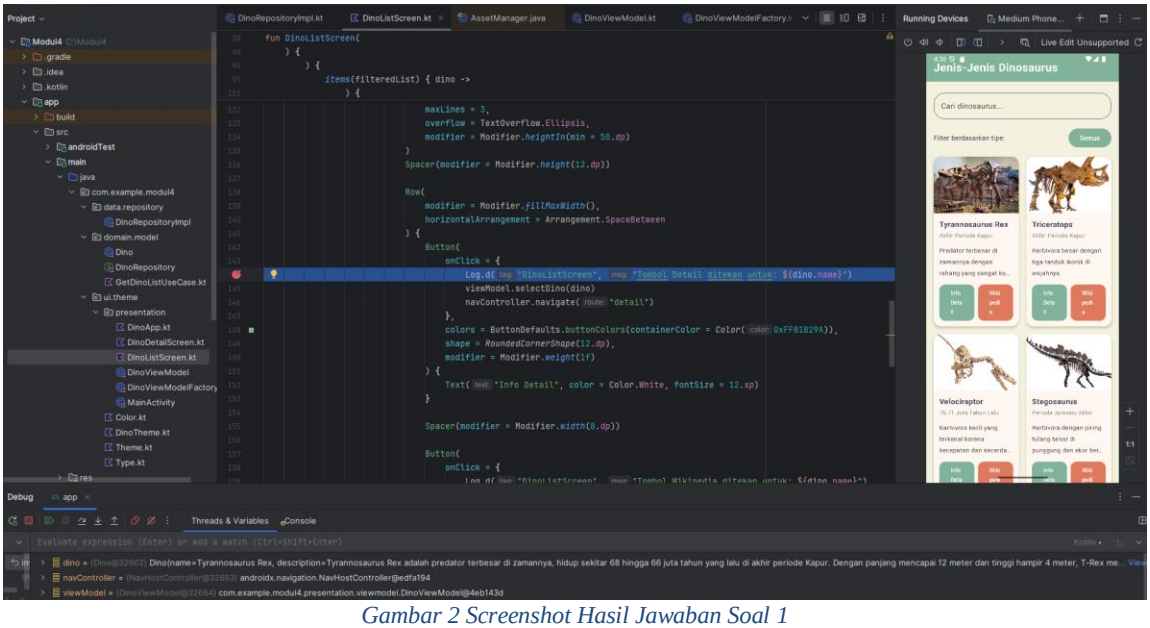
Table 9 Source Code DinoVIewModelFactoy.kt

- MainActivity.kt

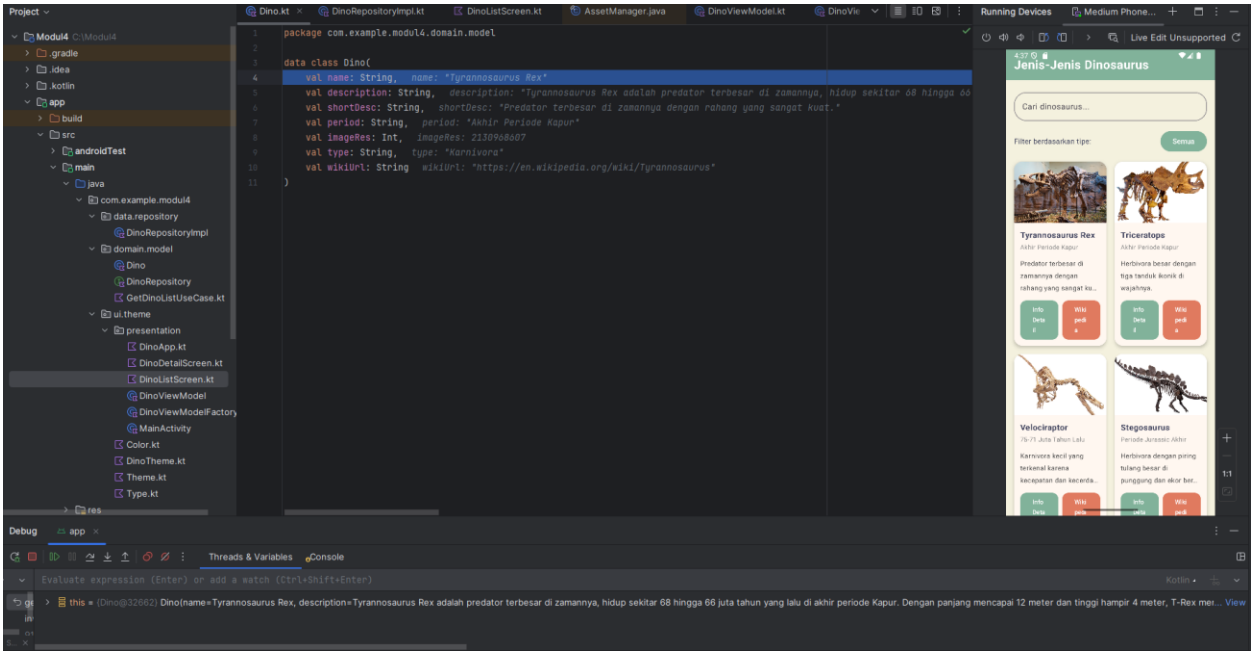
```
1   package com.example.modul4.presentation
2
3   import android.os.Bundle
4   import com.example.modul4.ui.theme.presentation.DinoApp
5   import androidx.activity.ComponentActivity
6   import androidx.activity.compose.setContent
7   import com.example.modul4.ui.theme.DinoTheme
8
9   class MainActivity : ComponentActivity() {
10      override fun onCreate(savedInstanceState: Bundle?) {
11          super.onCreate(savedInstanceState)
12          setContent {
13              DinoTheme {
14                  DinoApp()
15              }
16          }
17      }
18  }
```

Table 10 Source Code MainActivity.kt

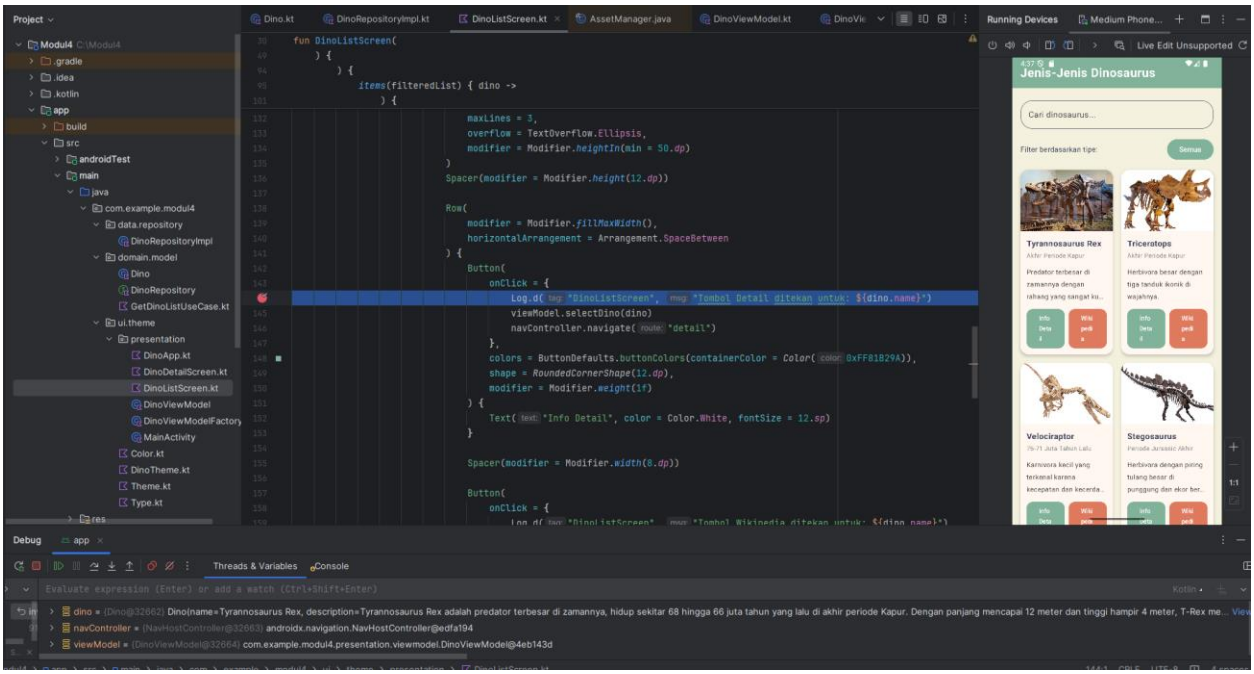
B. Output Program



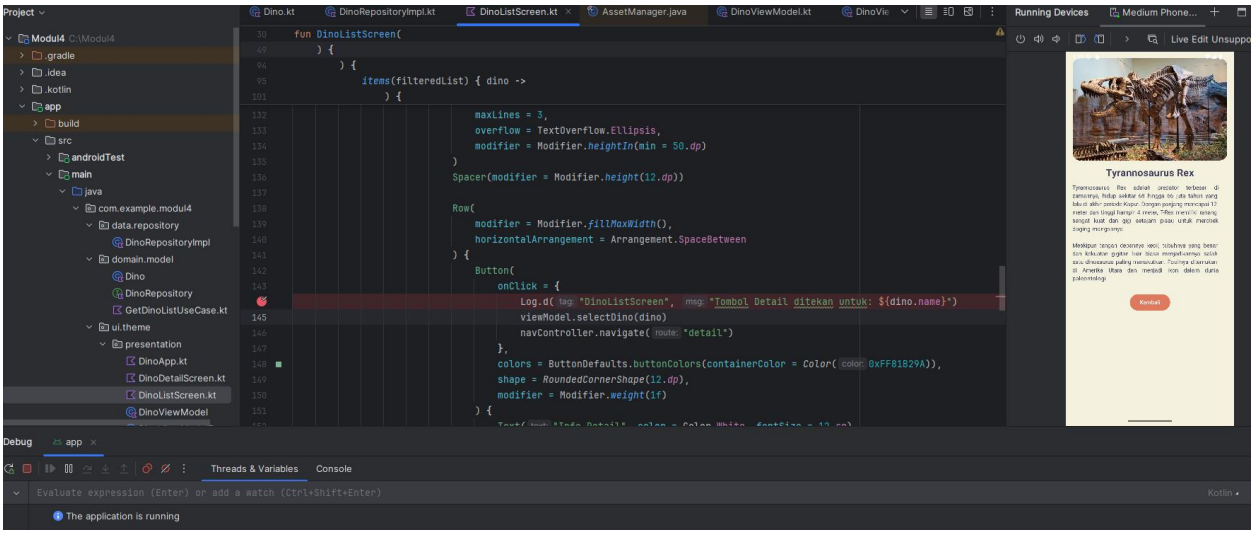
Gambar 2 Screenshot Hasil Jawaban Soal 1



Gambar 3 Screenshot Hasil Jawaban Soal 1



Gambar 4 Screenshot Hasil Jawaban Soal 1



Gambar 5 Screenshot Hasil Jawaban Soal 1

C. Pembahasan

- DinoRepositoryImpl.kt:

  Kelas DinoRepositoryImpl berisi logika untuk menyediakan data dinosaurus. Di dalamnya, terdapat sebuah properti privat bernama dinos yang merupakan sebuah List statis (hardcoded) berisi objek-objek Dino. Kelas ini juga mengimplementasikan dua fungsi *interface* yaitu getAllDinos() yang membungkus seluruh daftar dino ke dalam sebuah Flow menggunakan flowOf(), dan searchDinos(query: String) yang memfilter daftar dino berdasarkan nama yang mengandung query (tanpa mempedulikan huruf besar/kecil) sebelum membungkus hasilnya ke dalam Flow.

- Dino.kt

  Kelas ini berisikan 5 kelas yang dimana isinya ada nama(string), deskripsi(string), imageRes(integer) dan tipe(string) dan wikiUrl(string) bertujuan untuk menyimpan data

- DinoRepository.kt

  DinoRepository menetapkan dua fungsi yaitu *getAllDinos()* yang bertugas mengembalikan *Flow<List<Dino>>*, dan *searchDinos(query: String)* yang juga mengembalikan *Flow<List<Dino>>*. Penggunaan Flow dari Kotlin Coroutines memungkinkan data untuk diterima secara *asynchronous* sebagai aliran data (stream)

- SearchDonosUseCase.kt

  Bertujuan untuk mencari dinosaurus berdasarkan nama, File ini menerima query pencarian, lalu memanggil fungsi searchDinos dari repository untuk menjalankan tugasnya dan mengembalikan hasilnya.

- DinoApp.kt

  mengatur struktur utama aplikasi dan sistem navigasi. Di dalamnya, *rememberNavController()* digunakan untuk membuat NavController, yang bertugas mengelola perpindahan antar layar. Dan viewmodel disana menggunakan *viewModel(factory = DinoViewModelFactory())* untuk navigasi. Dan NavHost berguna sebagai navigasi pada semua layar

- DinoDetailScreen.kt

  Disini bertujuan untuk menampilkan informasi secara lengkap dari dinosaurus yang dipilih, dan data nya diambil melalui SelectDino yang tersimpan dalam ViewModel, lalu ada Column agar semua informasi tersusun rapi dan ada verticalScroll agar bisa scroll. Lalu juga saya meanmbahkan tombol navigasi kembali dengan cara navController.popBackStack()

- DinoListScreen.kt

  Data dinoList diambil dari ViewModel dan diubah menjadi *state* yang dapat diamati oleh Compose menggunakan collectAsState(). Lalu pada halaman ini ada OutlinedTextField yang memungkinkan pengguna mengetik searchQuery untuk mencari dinosaurus. Terdapat juga DropdownMenuBox untuk memfilter daftar berdasarkan selectedFilter ("Semua", "Herbivora", atau "Karnivora"). Lalu pada tiap card terdapat navigasi “Info Detail” dan “Wikipedia”

- DinoViewModel.kt

DinoViewModel berfungsi sebagai jembatan antara lapisan data dan lapisan UI. Di dalamnya terdapat instance dari DinoRepositoryImpl untuk mengambil data. Lalu ada dua StateFlow yaitu _dinoList yang menyimpan daftar dinosaurus dan _selectedDino yang menyimpan dinosaurus yang sedang dipilih oleh pengguna.

- DinoViewModelFactory.kt

  File ini bertujuan untuk mengontrol bagaimana DinoViewModel. Metode create di dalamnya akan memeriksa apakah kelas yang diminta adalah DinoViewModel. Jika ya, ia akan membuat dan mengembalikan instance baru dari DinoViewModel; jika tidak, ia akan melemparkan IllegalArgumentException untuk mencegah kesalahan.

- MainActivity.kt

  File ini adalah titik masuk dari aplikasi android. Jadi pada file ini terdapat metode setContent yanf berguna untuk mendefinisikan tata letak UI aplikasi menggunakan fungsi-fungsi Composable. Di sini, setContent memanggil Composable DinoApp() yang

dibungkus di dalam DinoTheme, yang bertanggung jawab untuk menerapkan tema visual (seperti warna dan font) ke seluruh aplikasi.

1. E. Debugger (Step Into, Step Over, Step Out)

Debugger adalah alat di Android Studio yang memungkinkan Anda menjalankan aplikasi baris per baris untuk memeriksa nilai variabel dan alur eksekusi kode. Ini sangat penting untuk menemukan dan memperbaiki bug.

Cara Menggunakanya:

- Menggunakan Breakpoint: Klik pada area di sebelah kiri nomor baris kode, Ini adalah *breakpoint*, titik di mana eksekusi aplikasi akan berhenti sejenak saat mode debug dijalankan.

- Mulai Debug: Jalankan aplikasi dengan menekan ikon kumbang atau melalui menu Run > Debug 'app'.

- Saat eksekusi mencapai *breakpoint*, jendela Debug akan muncul di bagian bawah, menampilkan variabel saat ini dan *call stack*.

- **Step Over (F8)**: Mengeksekusi baris kode saat ini dan pindah ke baris berikutnya dalam fungsi yang sama. Dan Step Over akan mengeksekusi seluruh fungsi tersebut tanpa masuk ke dalamnya.

- **Step Into (F7):** Step Into akan masuk ke dalam fungsi tersebut dan berhenti di baris pertama di dalamnya. Ini bertujuan untuk melihat apa yang terjadi di dalam fungsi yang Anda panggil.

- **Step Out (Shift + F8):** Mengeksekusi sisa baris kode di dalam fungsi saat ini dan keluar.

2. **Application Class** adalah sebuah kelas dasar dalam aplikasi Android yang diinisialisasi sebelum komponen lain (seperti `Activity` atau `Service`) dibuat saat proses aplikasi dimulai. Setiap aplikasi hanya memiliki satu instance `Application`.

   Fungsi Utama:

- **Inisialisasi Global**: Tempat terbaik untuk menginisialisasi library atau state yang bersifat global dan perlu ada selama siklus hidup aplikasi. Contoh: inisialisasi library logging (seperti Timber), dependency injection (seperti Hilt atau Koin), atau analytics.
- **Mengelola State Global**: Meskipun tidak disarankan untuk menyimpan data yang berhubungan dengan UI, kelas ini bisa digunakan untuk mengelola state global yang tidak terikat pada `Activity` tertentu.
- **Lifecycle Callbacks**: Anda dapat me-listen *lifecycle events* dari komponen aplikasi lain.

D. Tautan Git

https://github.com/Yoruuu00/PemprogramanMobile/tree/main/Modul4