# LAPORAN AKHIR PRAKTIKUM
# PEMROGRAMAN MOBILE


**Oleh:**

**Jovan Gilbert Natamasindah** **NIM. 2310817310002**

**PROGRAM STUDI TEKNOLOGI INFORMASI**

**FAKULTAS TEKNIK**

**UNIVERSITAS LAMBUNG MANGKURAT**

**JUNI 2025**

# LEMBAR PENGESAHAN

# LAPORAN AKHIR PRAKTIKUM PEMROGRAMAN MOBILE

Laporan Akhir Praktikum Pemrograman Mobile ini disusun sebagai syarat lulus mata kuliah Praktikum Pemrograman Mobile. Laporan Akhir Praktikum ini dikerjakan oleh:

Nama Praktikan : Jovan Gilbert Natamasindah
NIM : 2310817310002

| | |
|---|---|
| Menyetujui, | Mengetahui, |
| Asisten Praktikum | Dosen Penanggung Jawab Praktikum |
| Zulfa Auliya Akbar | Muti`a Maulida S.Kom M.T.I |
| NIM. 2210817210026 | NIP. 198810272019032013 |

# DAFTAR ISI


LEMBAR PENGESAHAN ..... 2
DAFTAR ISI ..... 3
DAFTAR GAMBAR ..... 4
DAFTAR TABEL ..... 6
MODUL 1 : ANDROID BASIC WITH KOTLIN ..... 7
SOAL 1 ..... 7
A. Source Code ..... 9
B. Output Program ..... 13
C. Pembahasan ..... 13
MODUL 2 : ANDROID LAYOUT ..... 19
SOAL 1 ..... 19
A. Source Code ..... 20
B. Output Program ..... 24
C. Pembahasan ..... 26
MODUL 3 : BUILD A SCROLLABLE LIST ..... 33
SOAL 1 ..... 33
A. Source Code ..... 35
B. Output Program ..... 44
C. Pembahasan ..... 45
SOAL 2 ..... 52
MODUL 4 : VIEWMODEL AND DEBUGGING ..... 53
SOAL 1 ..... 53
A. Source Code ..... 54
B. Output Program ..... 66
C. Pembahasan ..... 69
SOAL 2 ..... 82
MODUL 5 : CONNECT TO THE INTERNET ..... 83

SOAL 1 ..... 83
A. Source Code ..... 83
B. Output Program ..... 98
C. Pembahasan ..... 100
Tautan Git ..... 116

# DAFTAR GAMBAR

**MODUL 1 : ANDROID BASIC WITH KOTLIN**

Gambar 1. Tampilan Awal Aplikasi ..... 7
Gambar 2. Tampilan Dadu Setelah DI Roll ..... 8
Gambar 3. Tampilan Roll Dadu Double ..... 9
Gambar 4. Screenshot Hasil Jawaban Soal 1 Modul 1 ..... 13

**MODUL 2 : ANDROID LAYOUT**

Gambar 5. Tampilan Awal Aplikasi ..... 19
Gambar 6. Tampilan Aplikasi Setelah Dijalankan..... 20
Gambar 7. Screenshot Hasil Jawaban Soal 1 Modul 2 ..... 26

**MODUL 3 : BUILD A SCROLLABLE LIST**

Gambar 8. Contoh UI List..... 34
Gambar 9. Contoh UI Detail ..... 34
Gambar 10. Screenshot Hasil Jawaban Soal 1 Modul 3 ..... 44

**MODUL 4 : VIEWMODEL AND DEBUGGING**

Gambar 11. Contoh Penggunaan Debugger..... 53
Gambar 12. Screenshot Hasil Jawaban Soal 1 Modul 4 ..... 66
Gambar 13. screenshot saat data item masuk list Soal 1 Modul 4 ..... 67
Gambar 14. screenshot saat tombol explicit intent ditekan Soal 1 Modul 4..... 67
Gambar 15. screenshot data dari list yang dipilih ketika ke halaman detail Soal 1 Modul 4..... 68

**MODUL 5 : CONNECT TO THE INTERNET**

Gambar 16. Screenshot Hasil Jawaban Soal 1 Modul 5 ..... 100

# DAFTAR TABEL

**MODUL 1 : Variabel, Operator, dan Array**

Tabel 1. Source Code MainActivity Soal 1 Modul 1 ..... 9
Tabel 2. Source Code RandomDice Soal 1 Modul 1 ..... 12

**MODUL 2 : ANDROID LAYOUT**

Tabel 3. Source Code MainActivity Soal 1 Modul 2 ..... 20
Tabel 4. Source Code tipCalculator Soal 1 Modul 2 ..... 24

**MODUL 3 : BUILD A SCROLLABLE LIST**

Tabel 5. Source Code MainActivity Soal 1 Modul 3 ..... 35
Tabel 6. Source Code AirlineData Soal 1 Modul 3 ..... 39
Tabel 7. Source Code UIComponent Soal 1 Modul 3 ..... 42

**MODUL 4 : VIEWMODEL AND DEBUGGING**

Tabel 8. Source Code MainActivity Soal 1 Modul 4 ..... 54
Tabel 9. Source Code sourceData Soal 1 Modul 4 ..... 55
Tabel 10. Source Code CardProp Soal 1 Modul 4 ..... 58
Tabel 11. Source Code cardDetailScreen Soal 1 Modul 4 ..... 58
Tabel 12. Source Code cardListScreen Soal 1 Modul 4 ..... 60
Tabel 13. Source Code CardViewModel Soal 1 Modul 4 ..... 62
Tabel 14. Source Code CardViewModelFactory Soal 1 Modul 4 ..... 63
Tabel 15. Source Code UIComponent Soal 1 Modul 4 ..... 64

**MODUL 5 : CONNECT TO THE INTERNET**

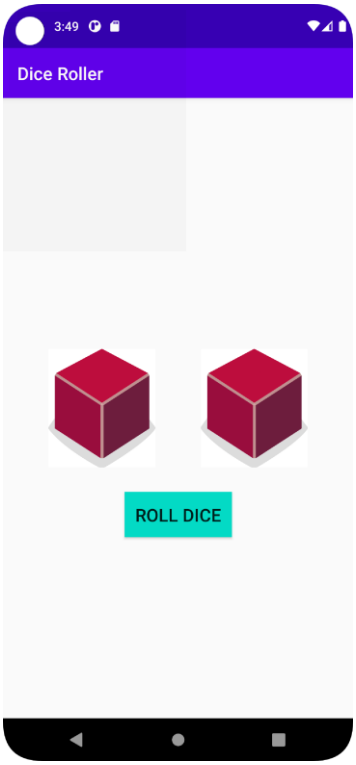
Tabel 16. Source Code MainActivity Soal 1 Modul 5 ..... 83
Tabel 17. Source Code MovieViewModel Soal 1 Modul 5 ..... 86
Tabel 18. Source Code MovieDetailScreen Soal 1 Modul 5 ..... 87
Tabel 19. Source Code MovieListScreen Soal 1 Modul 5 ..... 89
Tabel 20. Source Code TopBar Soal 1 Modul 5 ..... 90
Tabel 21. Source Code Text Soal 1 Modul 5 ..... 91
Tabel 22. Source Code movieCard Soal 1 Modul 5 ..... 92
Tabel 23. Source Code Glide Soal 1 Modul 5 ..... 93
Tabel 24. Source Code DarkModeSwitch Soal 1 Modul 5 ..... 93
Tabel 25. Source Code ButtonNav Soal 1 Modul 5 ..... 94
Tabel 26. Source Code Movie Soal 1 Modul 5 ..... 95
Tabel 27. Source Code TMDBAPI Soal 1 Modul 5 ..... 96
Tabel 28. Source Code RetrofitClient Soal 1 Modul 5 ..... 97

# MODUL 1 : ANDROID BASIC WITH KOTLIN

## SOAL 1

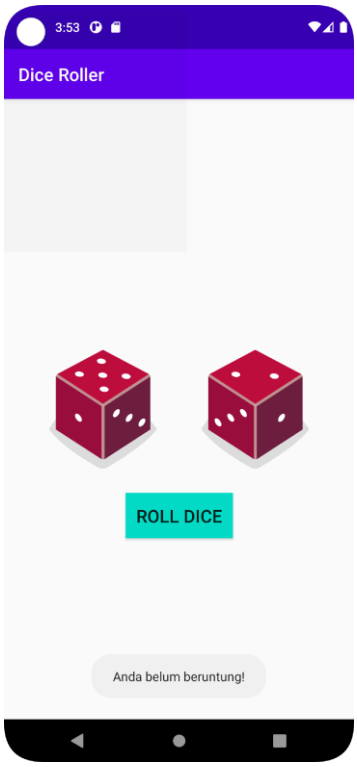
**Soal Praktikum:**
Buatlah sebuah aplikasi yang dapat menampilkan 2 (dua) buah dadu yang dapat berubah-ubah tampilannya pada saat user menekan tombol "Roll Dice". Aturan aplikasi yang akan dibangun adalah sebagaimana berikut:

1. Tampilan awal aplikasi setelah dijalankan akan menampilkan 2 buah dadu kosong seperti dapat dilihat pada Gambar 1.




*Gambar 1. Tampilan Awal Aplikasi*

2. Setelah user menekan tombol "Roll Dice" maka masing-masing dadu akan memunculkan sisi dadu masing-masing dengan angka antara 1 s/d 6. Apabila user mendapatkan nilai dadu yang berbeda antara Dadu 1 dengan Dadu 2 maka akan menampilkan pesan "Anda belum beruntung!" seperti dapat dilihat pada Gambar 2.

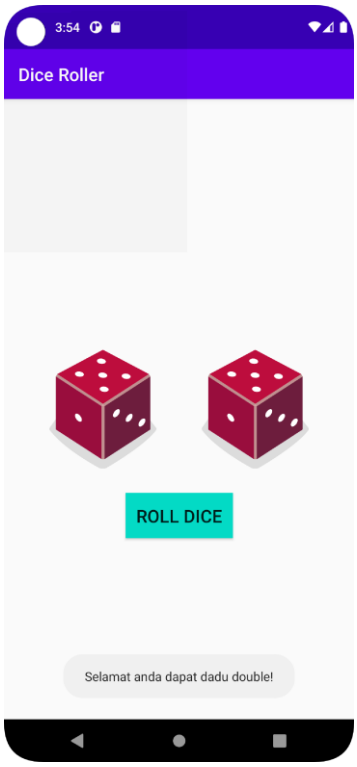



*Gambar 2. Tampilan Dadu Setelah DI Roll*

3. Apabila user mendapatkan nilai dadu yang sama antara Dadu 1 dan Dadu 2 atau nilai double, maka aplikasi akan menampilkan pesan "Selamat anda dapat dadu double!" seperti dapat dilihat pada Gambar 3.

4. Upload aplikasi yang telah anda buat kedalam repository github ke dalam **folder Module 2 dalam bentuk project.** Jangan lupa untuk melakukan **Clean Project** sebelum mengupload pekerjaan anda pada repo.

5. Untuk gambar dadu dapat didownload pada link berikut: https://drive.google.com/u/0/uc?id=147HT2lIH5qin3z5ta7H9y2N_5OMW81Ll&export= download




*Gambar 3. Tampilan Roll Dadu Double*

## A. Source Code

- **MainActivity.kt**

*Tabel 1. Source Code MainActivity Soal 1 Modul 1*

```
1   package com.android.modul1
2
3   import android.media.Image
4   import android.os.Bundle
5   import android.widget.Toast
6   import androidx.activity.ComponentActivity
7   import androidx.activity.compose.setContent
8   import androidx.activity.enableEdgeToEdge
9   import androidx.compose.foundation.Image
10  import androidx.compose.foundation.background
11  import androidx.compose.foundation.layout.Arrangement
12  import androidx.compose.foundation.layout.Column
13  import androidx.compose.foundation.layout.Row
14  import androidx.compose.foundation.layout.fillMaxSize
15  import androidx.compose.foundation.layout.padding
16  import androidx.compose.foundation.layout.statusBarsPadding
17  import androidx.compose.foundation.layout.width
18  import androidx.compose.foundation.shape.RoundedCornerShape
```

```
19  import androidx.compose.material3.Button
20  import androidx.compose.material3.ButtonDefaults
21  import androidx.compose.material3.ExperimentalMaterial3Api
22  import androidx.compose.material3.Scaffold
23  import androidx.compose.material3.Text
24  import androidx.compose.material3.TopAppBar
25  import androidx.compose.material3.TopAppBarDefaults
26  import androidx.compose.runtime.Composable
27  import androidx.compose.runtime.mutableStateOf
28  import androidx.compose.runtime.getValue
29  import androidx.compose.runtime.saveable.rememberSaveable
30  import androidx.compose.runtime.setValue
31  import androidx.compose.ui.Alignment
32  import androidx.compose.ui.Modifier
33  import androidx.compose.ui.platform.LocalContext
34  import androidx.compose.ui.res.colorResource
35  import androidx.compose.ui.res.painterResource
36  import androidx.compose.ui.unit.dp
37  import com.android.modul1.ui.theme.Modul1Theme
38
39  class MainActivity : ComponentActivity() {
40      override fun onCreate(savedInstanceState: Bundle?) {
41          super.onCreate(savedInstanceState)
42          enableEdgeToEdge()
43          setContent {
44              Modul1Theme {
45                  Display()
46              }
47          }
48      }
49  }
50
51  @Composable
52  fun Display() {
53      Scaffold(topBar = { TopBar() }) { innerPadding ->
54          RollDice(modifier = Modifier.padding(innerPadding))
55      }
56  }
57
58  @OptIn(ExperimentalMaterial3Api::class)
59  @Composable
60  fun TopBar( ) {
61      TopAppBar(
62          title = { Text("Dice Roller") },
63          colors = TopAppBarDefaults.topAppBarColors(
64              containerColor = colorResource(id =
    R.color.blue_dark),
65              titleContentColor = colorResource(id =
    R.color.white)
```

```
        ),
        modifier = Modifier
            .statusBarsPadding()
        )
}

@Composable
fun RollDice(modifier: Modifier = Modifier ) {

    var Dice1 by rememberSaveable { mutableStateOf(0) }
    var Dice2 by rememberSaveable { mutableStateOf(0) }
    val context = LocalContext.current

    val (img1, img2) = diceImg(Dice1, Dice2)
    val Img1 = img1
    val Img2 = img2

    Column(
        modifier = modifier
            .background(colorResource(id = R.color.cream))
            .fillMaxSize(),
        verticalArrangement = Arrangement.Center,
        horizontalAlignment = Alignment.CenterHorizontally
    ) {
        Row {
            Image(
                painter = painterResource(Img1),
                contentDescription = "Dice 1 image",
                modifier = Modifier
                    .width(150.dp)
            )
            Image(
                painter = painterResource(Img2),
                contentDescription = "Dice 2 image",
                modifier = Modifier
                    .width(150.dp)
            )
        }
        Button(onClick = {
            Dice1 = randomDice()
            Dice2 = randomDice()

            if(Dice1 == Dice2) {Toast.makeText(context,
"Selamat anda dapat dadu double!", Toast.LENGTH_SHORT).show()}
            else{Toast.makeText(context, "Anda belum
beruntung!", Toast.LENGTH_SHORT).show()}
        },
            colors = ButtonDefaults.buttonColors(
                contentColor = colorResource(id =
```

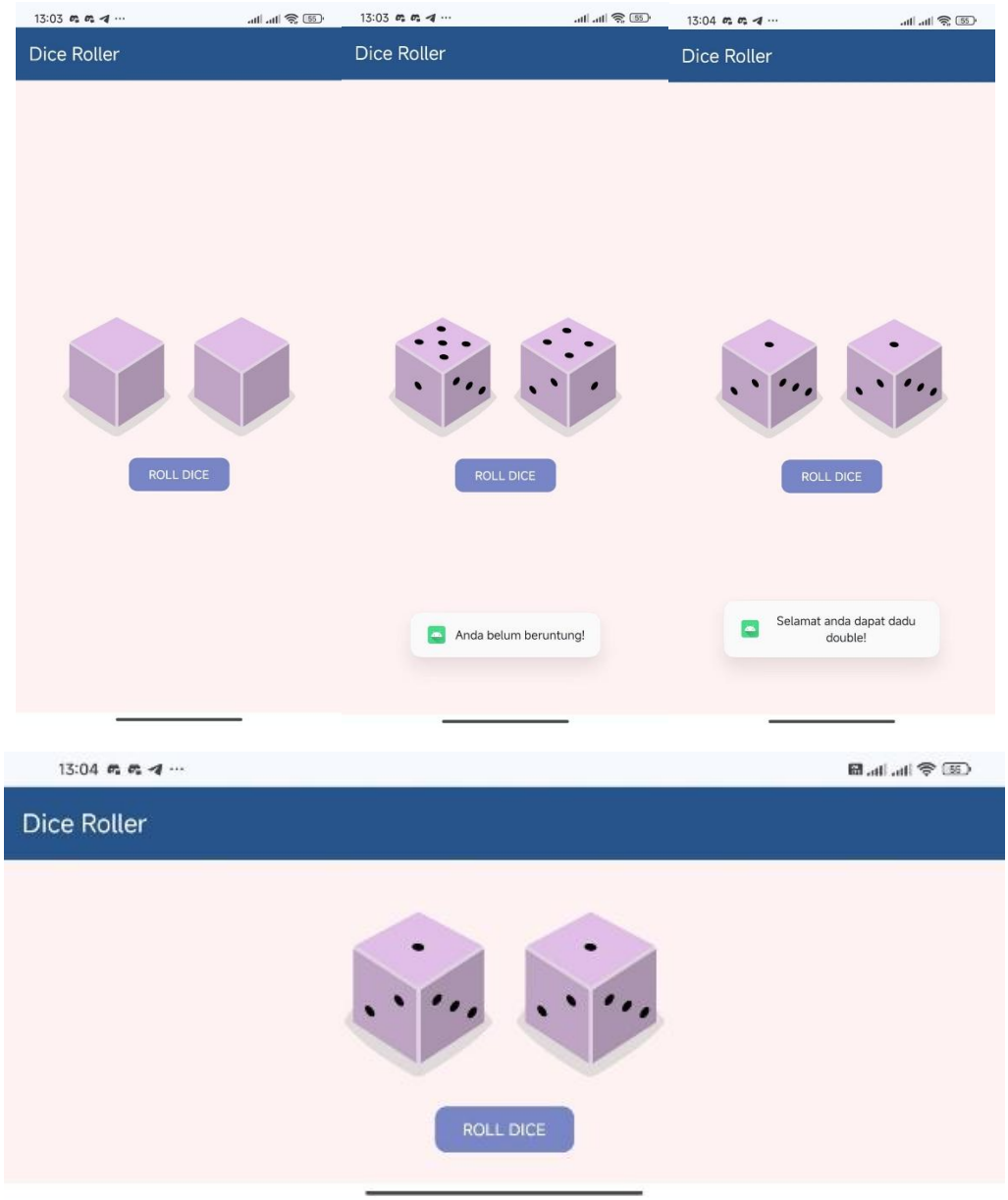
```
    R.color.white),
113                 containerColor = colorResource(id =
    R.color.blue_darker)
114             ),
115             shape = RoundedCornerShape(10.dp)
116         ) {
117             Text("ROLL DICE")
118         }
119     }
120 }
```

- **RandomDice.kt**

*Tabel 2. Source Code RandomDice Soal 1 Modul 1*

```
1   package com.android.modul1
2
3   import kotlin.random.Random
4
5   fun randomDice(): Int {return Random.nextInt(1,7)}
6
7   fun diceImg(Dice1: Int, Dice2: Int): Pair<Int, Int> {
8       var img1 = R.drawable.dice_0
9       var img2 = R.drawable.dice_0
10
11      when(Dice1) {
12          1 -> img1 = R.drawable.dice_1
13          2 -> img1 = R.drawable.dice_2
14          3 -> img1 = R.drawable.dice_3
15          4 -> img1 = R.drawable.dice_4
16          5 -> img1 = R.drawable.dice_5
17          6 -> img1 = R.drawable.dice_6
18      }
19      when(Dice2) {
20          1 -> img2 = R.drawable.dice_1
21          2 -> img2 = R.drawable.dice_2
22          3 -> img2 = R.drawable.dice_3
23          4 -> img2 = R.drawable.dice_4
24          5 -> img2 = R.drawable.dice_5
25          6 -> img2 = R.drawable.dice_6
26      }
27      return Pair(img1, img2)
28  }
```

## B. Output Program




*Gambar 4. Screenshot Hasil Jawaban Soal 1 Modul 1*

## C. Pembahasan

- **MainActivity.kt:**
  - Pada baris 1, **package com.android.modul1** pendeklarasian nama package file Kotlin.

- Pada baris 3-37, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
- Pada baris 39, **class MainActivity : ComponentActivity(),** merupakan titik mula yang menjadi kelas utama dan akan dijalankan pertama kali saat aplikasi dibuka
- Pada baris 40, **override fun onCreate(savedInstanceState: Bundle?),** berfungsi untuk menimpa (override) fungsi oncreate dari ComponentActivity.
- Pada baris 41, **super.onCreate(savedInstanceState),** berfungsi untuk memanggil superclass dari fungsi oncreate untuk memastikan bahwa proses inisialisasi standar dari Android dijalankan sebelum logika saya dijalankan.
- Pada baris 42, **enableEdgeToEdge(),** berfungsi agar tampilan aplikasi dapat menggunakan seluruh layar dari status bar sampai navigation bar atau fullscreen layout.
- Pada baris 43, **setContent(),** digunakan untuk menampilkan UI berbasis jetpack compose ke dalam activity
- Pada baris 44, **Modul1Theme,** merupakan fungsi yang berisi tema custom yang membungkus seluruh UI untuk memberikan style yang konsisten.
- Pada baris 45, **Display(),** merupakan fungsi yang dibuat untuk menampilkan aplikasi pengocok dadu ke dalam activity.
- Pada baris 51-72, **@composable,** merupakan annotation untuk menandai bahwa fungsi yang dibuat dibawahnya merupakan fungsi UI di jetpack compose.
- Pada baris 52, **fun Display(),** merupakan fungsi yang dibuat untuk menampung semua component UI yang ingin ditampilkan ke dalam activity, seperti TopBar() dan RollDice().

- Pada baris 53, **scaffold()**, merupakan komponen layout yang menjadi kerangka dasar dari tampilan yang mempermudah dalam mengatur strktur UI karena sudah disediakan slot-slot bawannya seperti topBar yang sudah saya isi dengan TopBar() dan content saya isi dengan RollDice().
- Pada baris 53, **TopBar()**, berfungsi untuk memanggil fungsi TopBar() di dalam scaffold dan dimasukkan ke dalam slot topBar untuk ditampilkan dalam activity.
- Pada baris 54, **RollDice(),** berfungsi untuk memanggil fungsi RollDice() dengan parameter Modifier.padding(innerPadding) untuk memmberikan padding bawaan dari scaffold.
- Pada baris 58, **@OptIn(ExperimentalMaterial3Api::class),** berfungsi untuk memberitahukan compiler bahwa fungsi composable dibawahnya masih menggunakan API yang bersifat eksperimental untuk menghindari error saat kompilasi.
- Pada baris 60, **fun TopBar(),** merupakan fungsi composable yang dibuat untuk menampung komponen UI TopAppBar yang akan ditampilkan ke activity tepatnya dibagian atas layar.
- Pada baris 61, **TopAppBar()**, komponen yang berfungsi untuk menampilkan elemen UI di bagian atas layar.
    - Pada baris 62, **title**, merupakan parameter dalam TopAppBar() untuk menampilkan judul .
    - Pada baris 63, **colors** merupakan parameter dalam TopAppBar() untuk mengatur warna background (containerColor) dan warna teks judul (titleContentColor).
    - Pada baris 67, **modifier** merupakan parameter dalam TopAppBar() untuk menerapkan modifikasi terhadap tampilan komponen, salah satunya memberikan .statusBarPadding() untuk menambahkan padding secara otomatis di atas layar guna menghindari tabrakan elemen dengan status bar.

- Pada baris 73, **fun RollDice(modifier: Modifier = Modifier),** merupakan fungsi composable yang dibuat untuk menampung beberapa logika sederhana mengenai aplikasi pengocok dadu, menerima parameter modifier, dan komponen UI Column, Row, Image, Button, Text yang akan ditampilkan ke activity.
- Pada baris 75 dan 76, **var** merupakan keyword untuk membuat suatu variable bersifat mutable yang menampung nilai dari berbagai tipe data.
- Pada baris 75 dan 76, **rememberSaveable** berfungsi untuk menyimpan dan mengingat nilai dari variable, dalam kasus ini Dice1 dan Dice2 agar selama komponen composable masih aktif dan jika terjadi perubahan konfigurasi layar misalnya saat dirotasi.
- Pada baris 77, **LocalContext.current** berfungsi untuk mendapatkan context dari acrivity saat ini dalam fungsi composable dan juga digunakan untuk menampilkan Toast sebagai notifikasi dari hasil dadu.
- Pada baris 77-81, **val** merupakan keyword untuk membuat suatu variable bersifat immutable yang menampung nilai dari berbagai tipe data
- Pada baris 83, **Column()** merupakan komponen layout mirip seperti scaffold, hanya saja Column digunakan untuk menyusun elemen UI secara vertical.
    - Pada baris, 84 **modifier** merupakan parameter dalam Column() untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.background* untuk memberikan warna background dan *.fillMaxSize* agar column mengisi seluruh ukuran layar.
    - Pada baris 87, **verticalArrangment** berfungsi untuk mengatur posisi vertical dari isi Column.
    - Pada baris 88, **horizontalAlignment** berfungsi untuk mengatur posisi horizontal dari isi Column.
- Pada baris 90, **Row** merupakan komponen layout yang berlawanan dari Column, Dimana jika Column menyusun elemen UI secara Vertical, maka Row menyusun elemen UI secara horizontal.

- Pada baris 91 dan 97, **Image()** merupakan komponen UI yang berfungsi untuk menampilkan gambar.
    - Pada baris 92 dan 98, **painter** merupakan parameter dalam Image() untuk memasukkan gambar dari resource agar tampil di dalam Image()
    - Pada baris 93 dan 99, **contentDescription** merupakan parameter dalam Image() untuk memberikan deskripsi gambar.
    - Pada baris 94 dan 100, **modifier** merupakan parameter dalam Image() untuk memodifikasi tampilan gambar, dalam kasus ini memodifikasi widthnya menjadi 150.dp.
- Pada baris 104, **Button()** merupakan komponen UI yang berfungsi untuk membuat tombol pada aplikasi.
    - Pada baris 104, **onClick** merpakan parameter dalam Button() untuk memberikan aksi yang akan dijalankan ketika tombol ditekan.
    - Pada baris 108 dan 109, **if** dan **else** merupakan perkondisian Dimana jika kondisi if terpenuhi maka statementnya akan dijalankan, tetapi jika tidak maka else akan dijalankan.
    - Pada baris 111, **colors** merupakan parameter dalam Button() untuk mengatur warna background (containerColor) dan warna teks (titleContentColor).
    - Pada baris 115, **shape** merupakan parameter dalam Button() untuk mengatur sudut agar dapat membulat.
    - Pada baris 117, **Text()** merupakan komponen UI yang berfunsi menampilkan teks. Dalam kasus ini menampilkan teks di dalam tombol.

- **RandomDice.kt**
    - Pada baris 1, **package com.android.modul1** pendeklarasian nama package file Kotlin.

- Pada baris 3, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose
- Pada baris 5, **fun randomDice()** merupakan fungsi untuk mengacak nilai dadu dengan cara menggunakan fungsi random untuk mengacak angak dari 1 sampai 6 yang akan ditampilkan dalam bentuk gambar dadu.
- Pada baris 5 dan 27, **return** merupakan pengembalian terhadapt suatu nilai di sebuah fungsi . jika di randomDice() akan mengembalikan angak random dan di diceImg() akan mengembalikan pasangan id gambar dadu
- Pada baris 7, **diceImg(Dice1: Int, Dice2: Int): Pair<Int, Int>** merupakan fungsi untuk mendapatkan gambar dadu berdasarkan angka random dari fungsi randomDice()
- Pada baris 8 dan 9, **var** merupakan keyword untuk membuat suatu variable bersifat mutable yang menampung nilai dari berbagai tipe data.
- Pada baris 11 dan 19, **when** merupakan perkondisian mirip seperti if else hanya saja syntaxnya lebih pendek dan mudah dipahami.

# MODUL 2 : ANDROID LAYOUT

## SOAL 1

**Soal Praktikum:**
Buatlah sebuah aplikasi kalkulator tip yang dirancang untuk membantu pengguna menghitung tip yang sesuai berdasarkan total biaya layanan yang mereka terima. Fitur-fitur yang diharapkan dalam aplikasi ini mencakup:

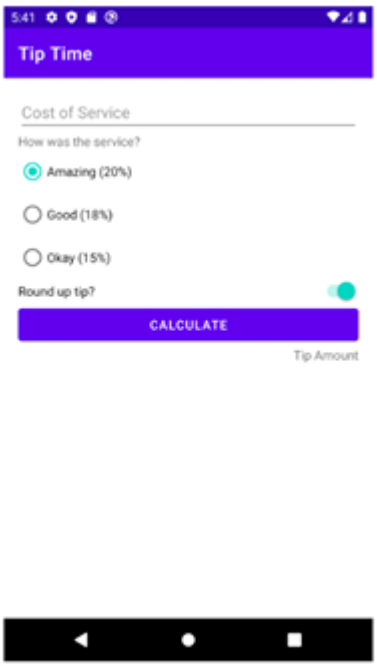
1. Input Biaya Layanan: Pengguna dapat memasukkan total biaya layanan yang diterima dalam bentuk nominal.

2. Pilihan Persentase Tip: Pengguna dapat memilih persentase tip yang diinginkan dari opsi yang disediakan, yaitu 15%, 18%, dan 20%.

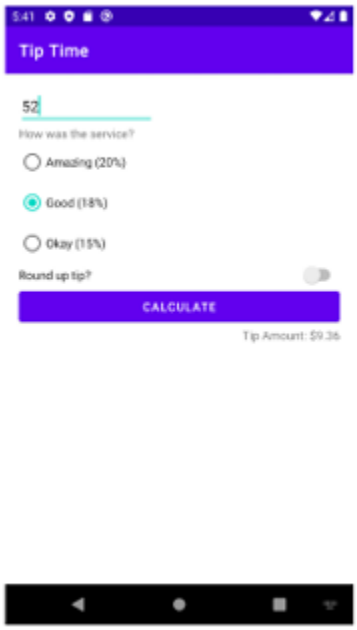
3. Pengaturan Pembulatan Tip: Pengguna dapat memilih untuk membulatkan tip ke angka yang lebih tinggi.

4. Tampilan Hasil: Aplikasi akan menampilkan jumlah tip yang harus dibayar secara langsung setelah pengguna memberikan input.

Tampilan awal aplikasi setelah dijalankan akan menampilkan 2 buah dadu kosong seperti dapat dilihat pada Gambar 1.




*Gambar 5. Tampilan Awal Aplikasi*




*Gambar 6. Tampilan Aplikasi Setelah Dijalankan*

## A. Source Code

- **MainActivity.kt**

*Tabel 3. Source Code MainActivity Soal 1 Modul 2*

```
1   package com.android.modul2
2
3   import android.os.Bundle
4   import android.widget.Toast
5   import androidx.activity.ComponentActivity
6   import androidx.activity.compose.setContent
7   import androidx.activity.enableEdgeToEdge
8   import androidx.compose.foundation.background
9   import androidx.compose.foundation.layout.Arrangement
10  import androidx.compose.foundation.layout.Column
11  import androidx.compose.foundation.layout.Row
12  import androidx.compose.foundation.layout.fillMaxSize
13  import androidx.compose.foundation.layout.fillMaxWidth
14  import androidx.compose.foundation.layout.padding
15  import androidx.compose.foundation.layout.statusBarsPadding
16  import androidx.compose.foundation.rememberScrollState
```

```
17  import androidx.compose.foundation.shape.RoundedCornerShape
18  import androidx.compose.foundation.text.KeyboardOptions
19  import androidx.compose.foundation.verticalScroll
20  import androidx.compose.material3.*
21  import androidx.compose.runtime.*
22  import androidx.compose.runtime.mutableStateOf
23  import androidx.compose.runtime.saveable.rememberSaveable
24  import androidx.compose.ui.Alignment
25  import androidx.compose.ui.Modifier
26  import androidx.compose.ui.graphics.Color
27  import androidx.compose.ui.platform.LocalContext
28  import androidx.compose.ui.res.colorResource
29  import androidx.compose.ui.text.input.KeyboardType
30  import androidx.compose.ui.unit.dp
31  import com.android.modul2.ui.theme.Modul2Theme
32
33  class MainActivity : ComponentActivity() {
34      override fun onCreate(savedInstanceState: Bundle?) {
35          super.onCreate(savedInstanceState)
36          enableEdgeToEdge()
37          setContent {
38              Modul2Theme {
39                      Display()
40              }
41          }
42      }
43  }
44
45  @Composable
46  fun Display() {
47      Scaffold(topBar = { TopBar() }) { innerPadding ->
48          TipCalc(modifier = Modifier.padding(innerPadding))
49      }
50  }
51
52  @OptIn(ExperimentalMaterial3Api::class) // agar bisa
    menggunakan topappbar yang bersifat exprerimental
53  @Composable
54  fun TopBar( ) {
55      TopAppBar(
56          title = { Text("Tip Time") },
57          colors = TopAppBarDefaults.topAppBarColors(
58              containerColor = colorResource(id =
    R.color.hijau_dark),
59              titleContentColor = colorResource(id =
    R.color.white)
60          ),
61          modifier = Modifier
62              .statusBarsPadding() // menghilangkan padding
```

```
    otomatis untuk menghindari tabrakan dengan status bar
63      )
64  }
65
66  @OptIn(ExperimentalMaterial3Api::class) // agar bisa
    menggunakan topappbar yang bersifat exprerimental
67  @Composable
68  fun TipCalc(modifier: Modifier = Modifier) {
69      var inputNilai by rememberSaveable { mutableStateOf("") }
70      var optionSelect by rememberSaveable {
    mutableStateOf("Amazing") }
71      var switchPosition by rememberSaveable {
    mutableStateOf(true) }
72      var hasil by rememberSaveable { mutableStateOf(0.0) }
73      val context = LocalContext.current
74
75      Column(
76          modifier = modifier
77              .background(colorResource(id = R.color.abu))
78              .fillMaxSize()
79              .verticalScroll(rememberScrollState())
80              .padding(horizontal = 10.dp)
81      ) {
82          TextField(
83              value = inputNilai,
84              onValueChange = {inputNilai = it},
85              label = { Text("Cost of Service") },
86              modifier = Modifier
87                  .fillMaxWidth(),
88              colors =
    TextFieldDefaults.textFieldColors(containerColor =
    Color.Transparent),
89              keyboardOptions = KeyboardOptions(keyboardType =
    KeyboardType.Number)
90          )
91          Text("How was the service?")
92
93          Row(
94              verticalAlignment = Alignment.CenterVertically
95          ){
96              RadioButton(
97                  selected = optionSelect == "Amazing",
98                  onClick = { optionSelect = "Amazing"},
99              )
100             Text("Amazing (20%)")
101         }
102         Row(
103             verticalAlignment = Alignment.CenterVertically
104         ){
```

```
105             RadioButton(
106                 selected = optionSelect == "Good",
107                 onClick = { optionSelect = "Good"}
108             )
109             Text("Good (18%)")
110         }
111         Row(
112             verticalAlignment = Alignment.CenterVertically
113         ){
114             RadioButton(
115                 selected = optionSelect == "Okay",
116                 onClick = { optionSelect = "Okay"}
117             )
118             Text("Okay (16%)")
119         }
120 
121         Row(
122             modifier = Modifier
123                 .fillMaxWidth(),
124             verticalAlignment = Alignment.CenterVertically,
125             horizontalArrangement = Arrangement.SpaceBetween
126         ) {
127             Text("Round up tip?")
128             Switch(
129                 checked = switchPosition,
130                 onCheckedChange = {isChecked -> switchPosition
    = isChecked}
131             )
132         }
133 
134         Button(
135             onClick = {
136                 if(inputNilai.isEmpty() || inputNilai == "0")
    {
137                     Toast.makeText(context, "Silahkan masukkan
    angka dan bukan nol!", Toast.LENGTH_SHORT).show()
138                     return@Button
139                 }
140                 if(inputNilai < "0") {
141                     Toast.makeText(context, "jangan masukkan
    angka negatif!", Toast.LENGTH_SHORT).show()
142                     return@Button
143                 }
144                 hasil = tipCalculator(inputNilai.toInt(),
    optionSelect, switchPosition)
145                       },
146             modifier = Modifier
147                 .fillMaxWidth(),
148             colors = ButtonDefaults.buttonColors(
```

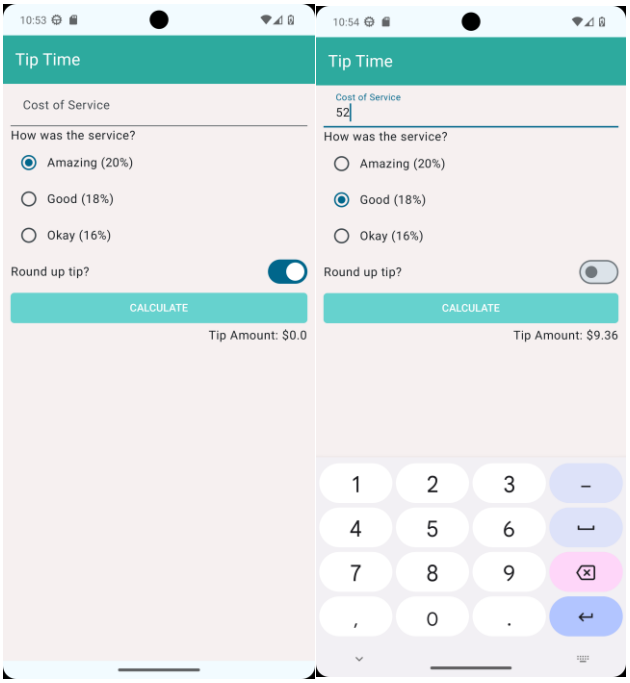
```
149                 contentColor = colorResource(R.color.white),
150                 containerColor =
    colorResource(R.color.hijau_muda)
151             ),
152             shape = RoundedCornerShape(5.dp)
153             ) { Text("CALCULATE") }
154
155         Row(
156             modifier = Modifier.fillMaxWidth(),
157             horizontalArrangement = Arrangement.End
158         ) {
159             Text("Tip Amount: $")
160             hasil.let { Text("$it") }
161         }
162     }
163 }
```

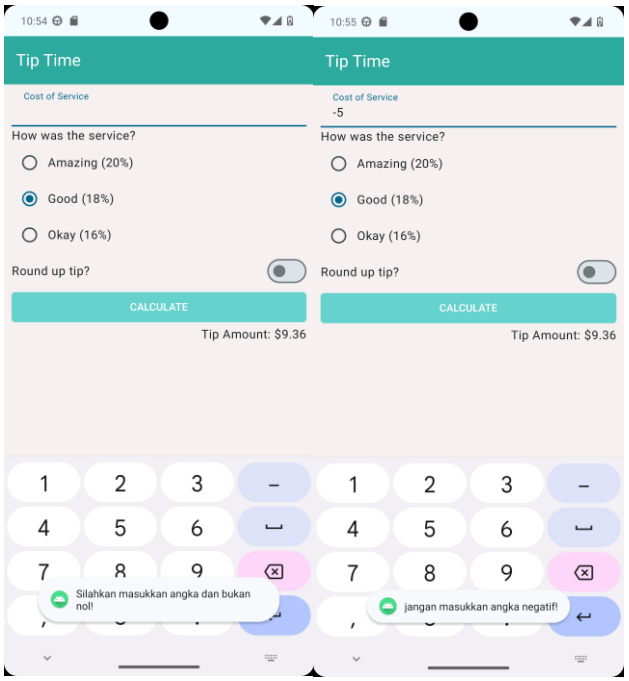
- **tipCalculator.kt**

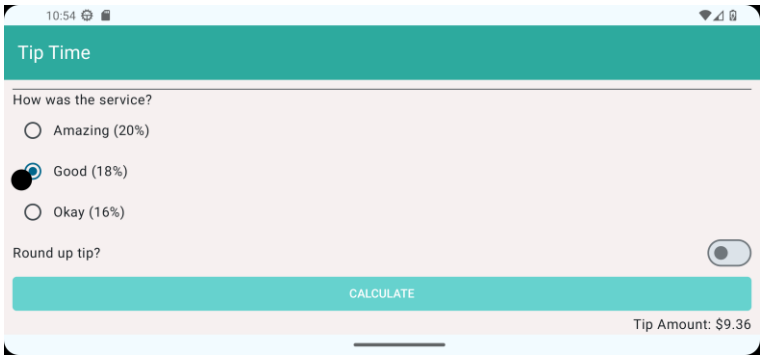
*Tabel 4. Source Code tipCalculator Soal 1 Modul 2*

```
1  package com.android.modul2
2
3  import kotlin.math.ceil
4
5  fun tipCalculator( nilai: Int,  opsi: String, switch: Boolean):
   Double{
6      var hasil = 0.0;
7      when(opsi){
8          "Amazing" -> hasil =  nilai * 20.0/100.0
9          "Good" -> hasil = nilai * 18.0/100.0
10         "Okay" -> hasil = nilai * 16.0/100.0
11     }
12     if (switch) {hasil = ceil(hasil) }
13     return hasil
14 }
```

## B. Output Program


10:53
Tip Time
Cost of Service
How was the service?
Amazing (20%)
Good (18%)
Okay (16%)
Round up tip?
CALCULATE
Tip Amount: $0.0
10:54
Tip Time
Cost of Service
52
How was the service?
Amazing (20%)
Good (18%)
Okay (16%)
Round up tip?
CALCULATE
Tip Amount: $9.36


10:54
Tip Time
Cost of Service
How was the service?
Amazing (20%)
Good (18%)
Okay (16%)
Round up tip?
CALCULATE
Tip Amount: $9.36
Silahkan masukkan angka dan bukan nol!
10:55
Tip Time
Cost of Service
-5
How was the service?
Amazing (20%)
Good (18%)
Okay (16%)
Round up tip?
CALCULATE
Tip Amount: $9.36
jangan masukkan angka negatif!




*Gambar 7. Screenshot Hasil Jawaban Soal 1 Modul 2*

## C. Pembahasan

- **MainActivity.kt:**
  - Pada baris 1, **package com.android.modul2** pendeklarasian nama package file Kotlin.
  - Pada baris 3-31, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
  - Pada baris 33, **class MainActivity : ComponentActivity(),** merupakan titik mula yang menjadi kelas utama dan akan dijalankan pertama kali saat aplikasi dibuka
  - Pada baris 34, **override fun onCreate(savedInstanceState: Bundle?),** berfungsi untuk menimpa (override) fungsi oncreate dari ComponentActivity.
  - Pada baris 35, **super.onCreate(savedInstanceState),** berfungsi untuk memanggil superclass dari fungsi oncreate untuk memastikan bahwa proses inisialisasi standar dari Android dijalankan sebelum logika saya dijalankan.
  - Pada baris 36, **enableEdgeToEdge(),** berfungsi agar tampilan aplikasi dapat menggunakan seluruh layar dari status bar sampai navigation bar atau fullscreen layout.

- Pada baris 37, **setContent(),** digunakan untuk menampilkan UI berbasis jetpack compose ke dalam activity
- Pada baris 38, **Modul2Theme,** merupakan fungsi yang berisi tema custom yang membungkus seluruh UI untuk memberikan style yang konsisten.
- Pada baris 39, **Display(),** merupakan fungsi yang dibuat untuk menampilkan aplikasi pengocok dadu ke dalam activity.
- Pada baris 45-67, **@composable,** merupakan annotation untuk menandai bahwa fungsi yang dibuat dibawahnya merupakan fungsi UI di jetpack compose.
- Pada baris 46, **fun Display(),** merupakan fungsi yang dibuat untuk menampung semua component UI yang ingin ditampilkan ke dalam activity, seperti TopBar() dan TipCalc().
- Pada baris 47, **scaffold()**, merupakan komponen layout yang menjadi kerangka dasar dari tampilan yang mempermudah dalam mengatur strktur UI karena sudah disediakan slot-slot bawannya seperti topBar yang sudah saya isi dengan TopBar() dan content saya isi dengan RollDice().
- Pada baris 47, **TopBar()**, berfungsi untuk memanggil fungsi TopBar() di dalam scaffold dan dimasukkan ke dalam slot topBar untuk ditampilkan dalam activity.
- Pada baris 48, **TipCalc(),** berfungsi untuk memanggil fungsi TipCalc() dengan parameter Modifier.padding(innerPadding) untuk memmberikan padding bawaan dari scaffold.
- Pada baris 52 dan 66, **@OptIn(ExperimentalMaterial3Api::class),** berfungsi untuk memberitahukan compiler bahwa fungsi composable dibawahnya masih menggunakan API yang bersifat eksperimental untuk menghindari error saat kompilasi.
- Pada baris 54, **fun TopBar(),** merupakan fungsi composable yang dibuat untuk menampung komponen UI TopAppBar yang akan ditampilkan ke activity tepatnya dibagian atas layar.

- Pada baris 55, **TopAppBar()**, komponen yang berfungsi untuk menampilkan elemen UI di bagian atas layar.
    - Pada baris 56, **title**, merupakan parameter dalam TopAppBar() untuk menampilkan judul .
    - Pada baris 57, **colors** merupakan parameter dalam TopAppBar() untuk mengatur  warna background (containerColor) dan warna teks judul (titleContentColor).
    - Pada baris 61, **modifier** merupakan parameter dalam TopAppBar() untuk menerapkan modifikasi terhadap tampilan komponen, salah satunya memberikan .statusBarPadding() untuk menambahkan padding secara otomatis di atas layar guna menghindari tabrakan elemen dengan status bar.
- Pada baris 68, **fun TipCalc(modifier: Modifier = Modifier),** merupakan fungsi composable yang dibuat untuk menampung beberapa logika sederhana mengenai aplikasi perhitungan tip, menerima parameter modifier, dan komponen UI Column, Row, Image, Button, Text yang akan ditampilkan ke activity.
- Pada baris 69-73, **var** merupakan keyword untuk membuat suatu variable bersifat mutable yang menampung nilai dari berbagai tipe data.
- Pada baris 69-72, **rememberSaveable** berfungsi untuk menyimpan dan mengingat nilai dari variable, dalam kasus ini inputNilai, optionSelect, switchPosition, hasil, dan context agar selama komponen composable masih aktif dan jika terjadi perubahan konfigurasi layar misalnya saat dirotasi.
- Pada baris 73, **LocalContext.current** berfungsi untuk mendapatkan context dari acrivity saat ini dalam fungsi composable dan juga digunakan untuk menampilkan Toast sebagai notifikasi dari peringatan jika user salah input.
- Pada baris 73, **val** merupakan keyword untuk membuat suatu variable bersifat immutable yang menampung nilai dari berbagai tipe data

- Pada baris 75, **Column()** merupakan komponen layout mirip seperti scaffold, hanya saja Column digunakan untuk menyusun elemen UI secara vertical.
    - Pada baris 76, **modifier** merupakan parameter dalam Column() untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.background* untuk memberikan warna background dan *.fillMaxSize* agar column mengisi seluruh ukuran layar.
- Pada baris 82, **TextField** merupakan komponen untuk menerima input dari pengguna
    - Pada baris 83, **value** merupakan nilai yang akan ditampilkan di dalam TextField
    - Pada baris 84, **onValueChange** berfungsi untuk memperbarui nilai saat pengguna mengetik sesuatu
    - Pada baris 85, **label** berfungsi untuk memberikanplaceholder saat input blm diklik pengguna, dan akan berpindah keatas saat diklik pengguna
    - Pada baris 86, **modifier** merupakan parameter dalam TextField untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.fillMaxWidth* agar column lebar TextField mengisi ukuran layar.
    - Pada baris 88, **colors** merupakan parameter dalam TextField untuk mengatur warna background (containerColor).
    - Pada baris 89, **keyboardOptions** berfungsi untuk membuat keyboard yang muncul saat TextField ditekan berupa numeric saja.
- Pada baris 91-159, **Text()** merupakan komponen UI yang berfunsi menampilkan teks.
- Pada baris 93-155, **Row** merupakan komponen layout yang berlawanan dari Column, Dimana jika Column menyusun elemen UI secara Vertical, maka Row menyusun elemen UI secara horizontal.

    - Pada baris 94-124, **verticalAlignment** berfungsi untuk menyusun elemen secara vertical.
    - Pada baris 94-124, **modifier** merupakan parameter dalam TextField untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.fillMaxWidth* agar column lebar Row mengisi ukuran layar.
    - Pada baris 94-124, **horizontalArrangement** berfungsi untuk memberi jarak maksimal antar elemen.
- Pada baris 96-114, **RadioButton** merupakan komponen UI yang memungkinkan pengguna memilih satu dari beberapa opsi yang tersedia
    - Pada baris 97-115, **selected** berfungsi untuk menentukan apakah radiobutton dipilih atau tidak
    - Pada baris 98-116, **onClick** merupakan fungsi yang dijalankan saat radio button di tekan
- Pada baris 128, **Switch** merupakan komponen berbentuk switch yang digunakan untuk menyalakan atau mematikan fitur
    - Pada baris 129, **checked** berfungsi untuk menentukan status switch saat ini apakah true atau false.
    - Pada baris 130, **onCheckedChange** merupakan fungsi yang dipanggil saat user mengubah posisi switch.
- Pada baris 134, **Button()** merupakan komponen UI yang berfungsi untuk membuat tombol pada aplikasi.
    - Pada baris 135, **onClick** merpakan parameter dalam Button() untuk memberikan aksi yang akan dijalankan ketika tombol ditekan.
    - Pada baris 136 dan 140, **if** merupakan perkondisian Dimana jika kondisi if terpenuhi maka statementnya akan dijalankan.
    - Pada baris 146, **modifier** merupakan parameter dalam TextField untuk menerapkan modifikasi terhadap tampilan komponen,

seperti *.fillMaxWidth* agar column lebar Button mengisi ukuran layar.

- Pada baris 148, **colors** merupakan parameter dalam Button() untuk mengatur warna background (containerColor) dan warna teks (titleContentColor).
- Pada baris 152, **shape** merupakan parameter dalam Button() untuk mengatur sudut agar dapat membulat.

- Pada baris 160, **.let** berfungsi untuk mengeksekusi blok kode dengan konteks objek yang dipanggil

- **RandomDice.kt**
  - Pada baris 1, **package com.android.modul2** pendeklarasian nama package file Kotlin.
  - Pada baris 3, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose
  - Pada baris 5, **fun tipCalc( nilai: Int, opsi: String, switch: Boolean): Double**
    merupakan fungsi untuk menghitung tip yang akan dibayarkan berdasarkan opsi radiobutton yang dipilih dan dapat membulatkan hasilnya keatas
  - Pada baris 6, **var** merupakan keyword untuk membuat suatu variable bersifat mutable yang menampung nilai dari berbagai tipe data.
  - Pada baris 7, **when** merupakan perkondisian mirip seperti if else hanya saja syntaxnya lebih pendek dan mudah dipahami.
  - Pada baris 12, if merupakan perkondisian Dimana jika kondisi if terpenuhi maka statementnya akan dijalankan.

- Pada baris 13, **return** merupakan pengembalian terhadapt suatu nilai di sebuah fungsi . jika di tipCalculator() akan mengembalikan hasil kalkulasi tip.

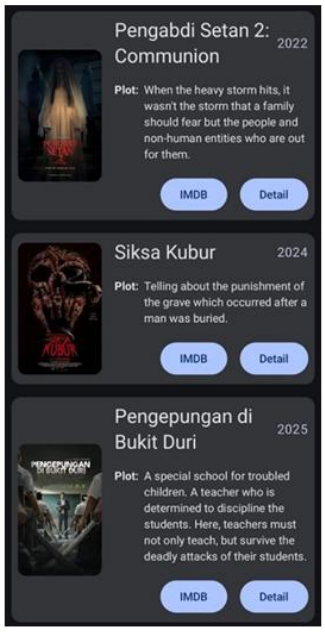
# MODUL 3 : BUILD A SCROLLABLE LIST

## SOAL 1

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

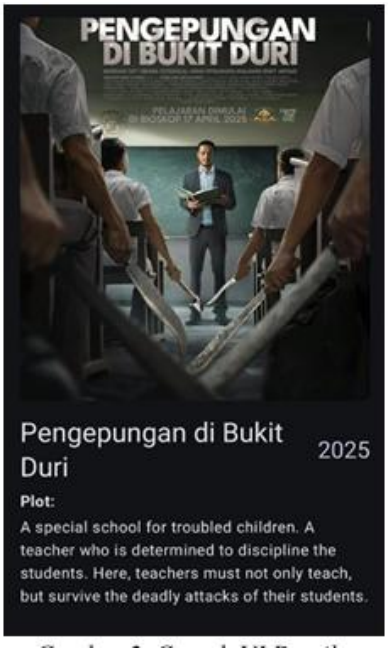
1. List menggunakan fungsi RecyclerView (XML) atau LazyColumn (Compose)
2. List paling sedikit menampilkan 5 item. Tema item yang ingin ditampilkan bebas
3. Item pada list menampilkan teks dan gambar sesuai dengan contoh di bawah
4. Terdapat 2 button dalam list, dengan fungsi berikut:
    a. Button pertama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

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




*Gambar 8. Contoh UI List*

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




*Gambar 9. Contoh UI Detail*

## A. Source Code

- **MainActivity.kt**

*Tabel 5. Source Code MainActivity Soal 1 Modul 3*

```
1   package com.android.modul3
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 androidx.activity.enableEdgeToEdge
9   import androidx.compose.foundation.Image
10  import androidx.compose.foundation.background
11  import androidx.compose.foundation.layout.Arrangement
12  import androidx.compose.foundation.layout.Column
13  import androidx.compose.foundation.layout.Row
14  import androidx.compose.foundation.layout.Spacer
15  import androidx.compose.foundation.layout.fillMaxSize
16  import androidx.compose.foundation.layout.fillMaxWidth
17  import androidx.compose.foundation.layout.height
18  import androidx.compose.foundation.layout.padding
19  import androidx.compose.foundation.layout.size
20  import
    androidx.compose.foundation.layout.statusBarsPadding
21  import androidx.compose.foundation.layout.width
22  import androidx.compose.foundation.lazy.LazyColumn
23  import androidx.compose.foundation.rememberScrollState
24  import
    androidx.compose.foundation.shape.RoundedCornerShape
25  import androidx.compose.foundation.verticalScroll
26  import androidx.compose.material3.Button
27  import androidx.compose.material3.Card
28  import androidx.compose.material3.CardDefaults
29  import
    androidx.compose.material3.ExperimentalMaterial3Api
30  import androidx.compose.material3.Scaffold
31  import androidx.compose.material3.Text
32  import androidx.compose.material3.TopAppBar
33  import androidx.compose.material3.TopAppBarDefaults
34  import androidx.compose.runtime.Composable
35  import androidx.compose.ui.Alignment
36  import androidx.compose.ui.Modifier
37  import androidx.compose.ui.draw.clip
38  import androidx.compose.ui.graphics.Color
39  import androidx.compose.ui.graphics.Shape
40  import androidx.compose.ui.platform.LocalContext
```

```
41  import androidx.compose.ui.res.colorResource
42  import androidx.compose.ui.res.painterResource
43  import androidx.compose.ui.text.font.FontWeight
44  import androidx.compose.ui.text.style.TextAlign
45  import androidx.compose.ui.unit.dp
46  import androidx.compose.ui.unit.sp
47  import
    androidx.core.splashscreen.SplashScreen.Companion.instal
    lSplashScreen
48  import androidx.navigation.NavController
49  import androidx.navigation.NavHostController
50  import androidx.navigation.compose.NavHost
51  import androidx.navigation.compose.composable
52  import androidx.navigation.compose.rememberNavController
53  import com.android.modul3.ui.theme.Modul3Theme
54  import
    com.bumptech.glide.integration.compose.ExperimentalGlide
    ComposeApi
55  import com.bumptech.glide.integration.compose.GlideImage
56
57  class MainActivity : ComponentActivity() {
58      override fun onCreate(savedInstanceState: Bundle?) {
59          super.onCreate(savedInstanceState)
60          enableEdgeToEdge()
61          setContent {
62              Modul3Theme {
63                  val navController =
    rememberNavController()
64                  Display(navController)
65              }
66          }
67      }
68  }
69
70  @Composable
71  fun Display(navController: NavHostController) {
72      Scaffold() { innerPadding ->
73          NavHost(
74              navController = navController,
75              startDestination = "card_list",
76              modifier = Modifier.padding(innerPadding)
77          ) {
78              composable("card_list") {
79                  CardList(navController)
80              }

81  composable("detail/{itemTitle}/{itemDesc}/{itemImageURL}
    ") { backStackEntry ->
82                  val itemTitle =
```

```
    backStackEntry.arguments?.getString("itemTitle")!!
83                  val itemDesc =
    backStackEntry.arguments?.getString("itemDesc")!!
84                  val itemImageURL =
    backStackEntry.arguments?.getString("itemImageURL")!!
85                  DetailPage(itemTitle, itemDesc,
    itemImageURL, navController)
86              }
87          }
88      }
89  }
90
91  @OptIn(ExperimentalGlideComposeApi::class)
92  @Composable
93  fun CardList( navController: NavController) {
94      val context = LocalContext.current
95      val bgcard = Color(0xFF7AE2CF)
96      val bgcolor = Color(0xFFFFFDF6)
97      Title("Maskapai-maskapai Penerbangan di Indonesia")
98      LazyColumn(
99          modifier = Modifier
100             .padding(top = 50.dp)
101             .background(bgcolor)
102         ){
103         items(cardProperties.size) {index ->
104             val property = cardProperties[index]
105             Card(
106                 modifier = Modifier
107                     .fillMaxWidth()
108                     .padding(6.dp)
109                     .height(200.dp),
110                 colors =
    CardDefaults.cardColors(containerColor = bgcard),
111             ) {
112                 Row(
113                     modifier = Modifier
114                         .padding(6.dp),
115                 ) {
116                     Img(property.ImageURL, 180)
117                     Column {
118                         Title(property.title)
119                         Desc(if(property.desc.length >
    80) property.desc.take(80) + "..." else property.desc)
120                         Spacer(modifier =
    Modifier.weight(1f))
121                         Row(
122                             horizontalArrangement =
    Arrangement.End,
123                             modifier = Modifier
```

```
124                                     .fillMaxWidth()
125                             ) {
126                                 Button(onClick = {
127                                     val intent =
    Intent(Intent.ACTION_VIEW, Uri.parse(property.Wiki))
128
    context.startActivity(intent)
129                                 }, modifier =
    Modifier.padding(horizontal = 6.dp),
130                                     shape =
    RoundedCornerShape(8.dp)
131                                 ) {Text("Wiki") }
132                                 Button(onClick = {
133
    navController.navigate("detail/${property.title}/${prope
    rty.desc}/
    ${Uri.encode(property.ImageURL)}")
134                                 }, shape =
    RoundedCornerShape(8.dp)) {Text("Detail") }
135                             }
136                         }
137                     }
138                 }
139             }
140         }
141 }
142
143 @OptIn(ExperimentalGlideComposeApi::class)
144 @Composable
145 fun DetailPage(itemTitle: String, itemDesc:String,
    itemImageURL:String, navController: NavController) {
146     val context = LocalContext.current
147     val namaMaskapai = detailProperties.find{ it.title
    == itemTitle}
148     Column(modifier = Modifier
149             .padding(16.dp)
150             .verticalScroll(rememberScrollState()),
151         horizontalAlignment =
    Alignment.CenterHorizontally
152         ) {
153             Img(itemImageURL, 400)
154             Title(itemTitle)
155             Spacer(modifier = Modifier.height(16.dp))
156             namaMaskapai?.let {
157                 DetailRow("Tanggal Berdiri", ":
    ${it.tglBerdiri}")
158                 DetailRow("Armada", ": ${it.armada}")
159                 DetailRow("Rute Tujuan", ": ${it.rute}
    ")
```

```
160             }
161             Spacer(modifier = Modifier.height(16.dp))
162             Desc(itemDesc)
163             Spacer(modifier = Modifier.height(16.dp))
164
165             Row(modifier =
    Modifier.fillMaxWidth(),horizontalArrangement =
    Arrangement.SpaceBetween) {
166                 Button(onClick = {
167                     val intent =
    Intent(Intent.ACTION_VIEW, Uri.parse(namaMaskapai?.let
    {it.website}))
168                     context.startActivity(intent)
169                 }, modifier =
    Modifier.padding(horizontal = 6.dp),
170                     shape = RoundedCornerShape(8.dp)
171                 ) {Text("Web $itemTitle") }
172                 Button(onClick =
    {navController.navigate("card_list")}, shape =
    RoundedCornerShape(8.dp)) {Text("Kembali") }
173             }
174         }
175     }
```

- **AirlineData.kt**

*Tabel 6. Source Code AirlineData Soal 1 Modul 3*

```
1   package com.android.modul3
2
3   data class cardProperty(
4       val title: String,
5       val desc: String,
6       val ImageURL: String,
7       val Wiki: String
8   )
9
10  val cardProperties = listOf(
11      cardProperty(
12          title = "Garuda Indonesia",
13          desc = "Garuda Indonesia adalah maskapai
    penerbangan nasional Indonesia yang didirikan pada tahun
    1949. Berbasis di Jakarta, Garuda dikenal dengan layanan
    penerbangannya yang berkualitas tinggi dan keramahannya,
    mencerminkan budaya Indonesia. Maskapai ini
    mengoperasikan penerbangan domestik dan internasional ke
    berbagai tujuan di Asia, Australia, Eropa, dan Timur
    Tengah. Garuda Indonesia juga merupakan anggota dari
    aliansi penerbangan global SkyTeam dan telah beberapa
```

```
        kali meraih penghargaan dunia atas pelayanan kabinnya.",
14          ImageURL = "https://cdn.plnspttrs.net/11964/pk-
        gib-garuda-indonesia-mcdonnell-douglas-dc-10-
        30_PlanespottersNet_714188_a594861cb0_o.jpg",
15          Wiki =
        "https://id.wikipedia.org/wiki/Garuda_Indonesia"
16      ),
17      cardProperty(
18          title = "Lion Air",
19          desc = "Lion Air adalah maskapai penerbangan
        swasta terbesar di Indonesia yang berdiri pada tahun
        1999. Fokus utamanya adalah layanan penerbangan berbiaya
        rendah (low-cost carrier) dengan rute domestik dan
        internasional. Lion Air terkenal dengan jaringan
        penerbangan yang luas dan harga tiket yang kompetitif.",
20          ImageURL = "https://cdn.plnspttrs.net/20446/pk-
        lhg-lion-air-boeing-747-
        412_PlanespottersNet_612707_2702071c86_o.jpg",
21          Wiki = "https://id.wikipedia.org/wiki/Lion_Air"
22      ),
23      cardProperty(
24          title = "Citilink",
25          desc = "Citilink adalah anak perusahaan dari
        Garuda Indonesia yang beroperasi sebagai maskapai
        berbiaya rendah. Didirikan pada tahun 2001, Citilink
        menawarkan penerbangan domestik dan regional dengan
        konsep layanan yang modern, dinamis, dan lebih santai
        untuk menarik kalangan muda dan pelancong bisnis.",
26          ImageURL = "https://cdn.plnspttrs.net/42179/pk-
        gaf-citilink-atr-72-600-72-
        212a_PlanespottersNet_1657140_83de625cfa_o.jpg",
27          Wiki = "https://id.wikipedia.org/wiki/Citilink"
28      ),
29      cardProperty(
30          title = "Super Air Jet",
31          desc = "Super Air Jet adalah maskapai baru di
        Indonesia yang mulai beroperasi pada tahun 2021.
        Mengusung konsep \"new lifestyle airline\", Super Air
        Jet fokus melayani segmen anak muda dengan harga
        terjangkau, desain modern, dan rute-rute domestik
        populer.",
32          ImageURL = "https://cdn.plnspttrs.net/14570/pk-
        std-super-air-jet-airbus-a320-232-
        wl_PlanespottersNet_1755960_137445c980_o.jpg",
33          Wiki =
        "https://www.superairjet.com/en/about.php"
34      ),
35      cardProperty(
36          title = "Batik Air",
```

```
37          desc = "Batik Air adalah maskapai layanan penuh (full-service) dari Lion Air Group yang didirikan pada tahun 2013. Batik Air menawarkan fasilitas premium seperti makanan dalam penerbangan dan hiburan di kursi, serta menghubungkan berbagai kota besar di Indonesia dan Asia.",
38          ImageURL = "https://cdn.plnspttrs.net/35032/pk-lug-batik-air-airbus-a320-214-wl_PlanespottersNet_1693958_5bbffba586_o.jpg",
39          Wiki = "https://id.wikipedia.org/wiki/Batik_Air"
40      ),
41      cardProperty(
42          title = "Pelita Air",
43          desc = "Pelita Air adalah maskapai yang awalnya fokus pada penerbangan carter dan layanan energi (minyak dan gas), namun sejak 2022 mulai mengembangkan penerbangan reguler domestik. Sebagai anak usaha Pertamina, Pelita Air membawa konsep layanan penerbangan yang nyaman dan profesional.",
44          ImageURL = "https://cdn.plnspttrs.net/22490/pk-pwd-pelita-air-service-airbus-a320-214-wl_PlanespottersNet_1698319_901e5c6a31_o.jpg",
45          Wiki = "https://id.wikipedia.org/wiki/Pelita_Air"
46      )
47  )
48
49  data class detailProperty(
50      val title: String,
51      val tglBerdiri: String,
52      val armada: String,
53      val rute: String,
54      val website:String
55  )
56
57  val detailProperties = listOf(
58      detailProperty(
59          title = "Garuda Indonesia",
60          tglBerdiri = "1 Agustus 1947",
61          armada = "77",
62          rute = "Belanda, Thailand, China, Hong Kong, Arab Saudi, Malaysia, Arab Saudi, Australia, Korea Selatan, Singapura, Jepang, United Arab Emirates",
63          website = "https://www.garuda-indonesia.com/id/id//",
64      ),
65      detailProperty(
66          title = "Lion Air",
67          tglBerdiri = "15 November 1999",
```

```
68          armada = "100",
69          rute = "Arab Saudi, China, Malaysia",
70          website = "https://www.lionair.co.id/",
71      ),
72      detailProperty(
73          title = "Citilink",
74          tglBerdiri = "16 Juli 2001",
75          armada = "57",
76          rute = "Australia, Malaysia, Papua Nugini,
    Singapura, Timor Leste, China",
77          website = "https://www.citilink.co.id/",
78      ),
79      detailProperty(
80          title = "Super Air Jet",
81          tglBerdiri = "1 March 2021",
82          armada = "61",
83          rute = "Malaysia",
84          website =
    "https://checkin.superairjet.com/dx/IUCI/",
85      ),
86      detailProperty(
87          title = "Batik Air",
88          tglBerdiri = "10 Juni 2012",
89          armada = "65",
90          rute = "Australia, Arab Saudi, China, India,
    Malaysia, Singapura, Thailand",
91          website = "https://www.batikair.com.my/",
92      ),
93      detailProperty(
94          title = "Pelita Air",
95          tglBerdiri = "24 Januari 1970",
96          armada = "33",
97          rute = "Indonesia",
98          website = "https://www.pelita-air.com/",
99      )
100 )
```

- **UiComponent.kt**

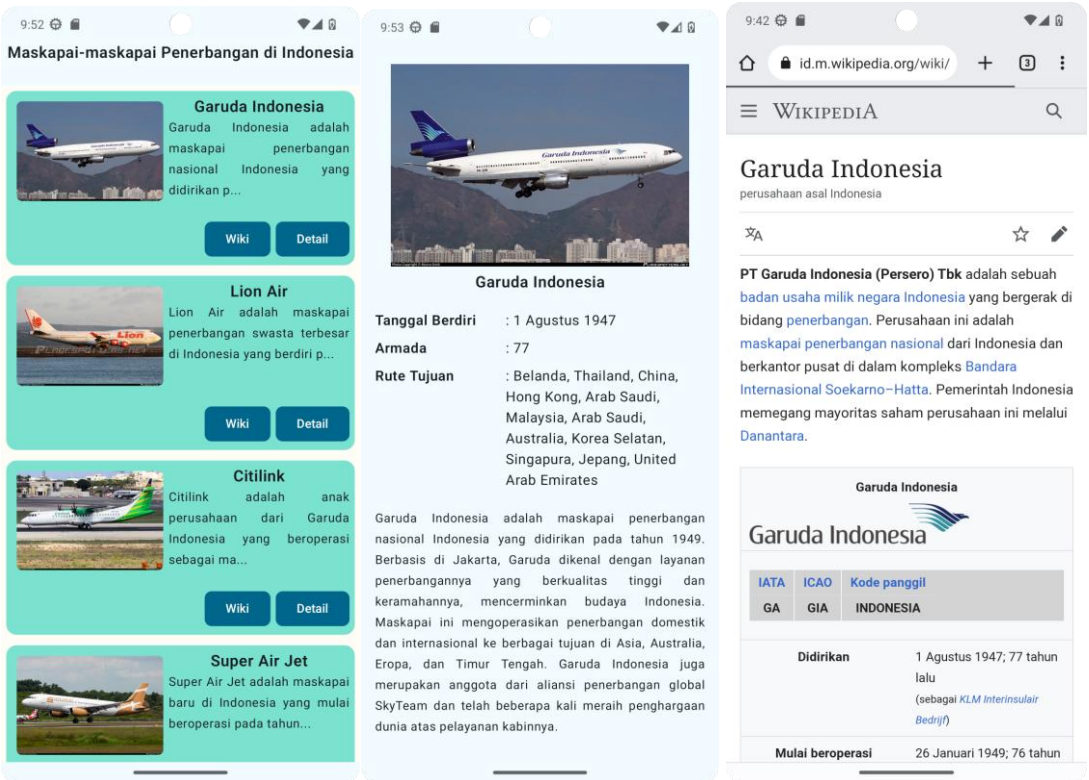
*Tabel 7. Source Code UIComponent Soal 1 Modul 3*

```
1   package com.android.modul3
2
3   import android.content.Intent
4   import android.net.Uri
5   import androidx.compose.foundation.layout.Row
6   import androidx.compose.foundation.layout.fillMaxWidth
7   import androidx.compose.foundation.layout.padding
8   import androidx.compose.foundation.layout.width
```

```
9  import
   androidx.compose.foundation.shape.RoundedCornerShape
10 import androidx.compose.material3.Button
11 import androidx.compose.material3.Text
12 import androidx.compose.runtime.Composable
13 import androidx.compose.ui.Modifier
14 import androidx.compose.ui.draw.clip
15 import androidx.compose.ui.text.font.FontWeight
16 import androidx.compose.ui.text.style.TextAlign
17 import androidx.compose.ui.unit.dp
18 import androidx.compose.ui.unit.sp
19 import
   com.bumptech.glide.integration.compose.ExperimentalGlideC
   omposeApi
20 import com.bumptech.glide.integration.compose.GlideImage
21
22 @Composable
23 fun Desc(desc: String) {
24     Text(
25         text = desc,
26         fontSize = 14.sp,
27         textAlign = TextAlign.Justify
28     )
29 }
30 @OptIn(ExperimentalGlideComposeApi::class)
31 @Composable
32 fun Img(url: String, width: Int) {
33     GlideImage(
34         model = url,
35         contentDescription = "My Image",
36         modifier = Modifier
37             .width(width.dp)
38             .padding(6.dp)
39             .clip(RoundedCornerShape(6.dp)),
40     )
41 }
42 @Composable
43 fun Title(title: String) {
44     Text(
45         text = title,
46         fontSize = 18.sp,
47         textAlign = TextAlign.Center,
48         fontWeight = FontWeight.SemiBold,
49         modifier = Modifier.fillMaxWidth()
50     )
51 }
52
53 @Composable
54 fun DetailRow(label:String, value:String) {
```

```
55      Row (
56          modifier = Modifier
57              .fillMaxWidth()
58              .padding(vertical = 4.dp)
59      ) {
60          Text(
61              text = label,
62              modifier = Modifier.width(150.dp),
63              fontWeight = FontWeight.SemiBold
64          )
65          Text(value)
66      }
67  }
```

## B. Output Program




*Gambar 10. Screenshot Hasil Jawaban Soal 1 Modul 3*

## C. Pembahasan

- **MainActivity.kt:**
  - Pada baris 1, **package com.android.modul3** pendeklarasian nama package file Kotlin.
  - Pada baris 3-55, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
  - Pada baris 57, **class MainActivity : ComponentActivity(),** merupakan titik mula yang menjadi kelas utama dan akan dijalankan pertama kali saat aplikasi dibuka
  - Pada baris 58, **override fun onCreate(savedInstanceState: Bundle?),** berfungsi untuk menimpa (override) fungsi oncreate dari ComponentActivity.
  - Pada baris 59, **super.onCreate(savedInstanceState),** berfungsi untuk memanggil superclass dari fungsi oncreate untuk memastikan bahwa proses inisialisasi standar dari Android dijalankan sebelum logika saya dijalankan.
  - Pada baris 60, **enableEdgeToEdge(),** berfungsi agar tampilan aplikasi dapat menggunakan seluruh layar dari status bar sampai navigation bar atau fullscreen layout.
  - Pada baris 61, **setContent(),** digunakan untuk menampilkan UI berbasis jetpack compose ke dalam activity
  - Pada baris 62, **Modul3Theme,** merupakan fungsi yang berisi tema custom yang membungkus seluruh UI untuk memberikan style yang konsisten.
  - Pada baris 63-147, **val** merupakan keyword untuk membuat suatu variable bersifat immutable yang menampung nilai dari berbagai tipe data.
  - Pada baris 63, **val navController = rememberNavController()** berfungsi untuk membuat dan menyimpan instance dari NavController

- Pada baris 64, **Display(navController),** merupakan fungsi yang dibuat untuk menampilkan aplikasi scrollable list ke dalam activity dan menerima parameter navController.
- Pada baris 70-144, **@composable,** merupakan annotation untuk menandai bahwa fungsi yang dibuat dibawahnya merupakan fungsi UI di jetpack compose.
- Pada baris 71, **fun Display(navController: NavHostController),** merupakan fungsi yang dibuat untuk menampilkan semua composable yang ingin ditampilkan ke dalam activity, seperti CardList() dan DetailPage() lalu menerima parameter navController.
- Pada baris 72, **scaffold()**, merupakan komponen layout yang menjadi kerangka dasar dari tampilan yang mempermudah dalam mengatur strktur UI karena sudah disediakan slot-slot bawannya seperti topBar yang sudah saya isi dengan TopBar() dan content saya isi dengan RollDice().
- Pada baris 73, **NavHost(),** berfungsi untuk menyediakan wadah untuk navigasi antar composable screen berdasarkan rute yang ditentukan
    - Pada baris 74, **navController** merupakan controller yang bertanggung jawab atas navigasi.
    - Pada baris 75, **startDestination** merupakan rute awal yang ditampilkan ketika aplikasi dijalankan.
    - Pada baris 76, modifier = **Modifier** berfungsi untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.padding* untuk memberikan jarak dalam component.
- Pada baris 73 dan 81, **composable(),** berfungsi untuk mendefinisikan rute, seperti di kasus ini ada "card_list" dan "detail"
- Pada baris 73, **CardList(navController),** merupakan fungsi untuk menampilkan list yang dapat scroll sekaligus menjadi halaman utama aplikasi.
- pada baris 82-84, **itemTitle, itemDesc, itemImageURL**

- pada baris 85, **DetailPage(itemTitle, itemDesc, itemImageURL, navController)** merupakan fungsi untuk menampilkan halaman detail dari aplikasi dan menerima beberapa parameter.
- Pada baris 91 dan 153, **@OptIn(ExperimentalGlideComposeApi::class),** berfungsi untuk memberitahukan compiler bahwa fungsi composable dibawahnya masih menggunakan API yang bersifat eksperimental untuk menghindari error saat kompilasi.
- Pada baris 68, **fun CardList( navController: NavController),** merupakan fungsi untuk menampilkan card-card dalam jumlah banyak yang da[at di scroll.
- Pada baris 94, **LocalContext.current** berfungsi untuk mendapatkan context dari acrivity saat ini dalam fungsi composable dan juga digunakan untuk menampilkan Toast sebagai notifikasi dari peringatan jika user salah input.
- Pada baris 97-154, **Title()** merupakan fungsi yang saya buat sendiri untuk memudahkan dalam membuat judul.
- Pada baris 98, **LazyColumn()** berfungsi untuk menampilkan daftar item secara vertical yang dapat di scroll tapi hanya yang terlihat di layar.
    - Pada baris 99, **modifier** merupakan parameter dalam Column() untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.padding* untuk memberikan jarak dalam componen dan *.background* untuk memberikan warna latar.
- Pada baris 103, **item()** berfungsi untuk membuat item di dalam LazyColumn sebanyak jumlah data.
- pada baris 105, **Card()** berfungsi untuk membuat sebuah kartu.
    - pada baris 106, **modifier** merupakan parameter dalam Column() untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.padding* untuk memberikan jarak dalam component,

*.height* untuk tinggi component, dan *.fillMaxWidth()* untuk membuat lebar component lebarnya selayar.

- pada baris 110, **colors** merupakan parameter dalam Card() untuk mengatur warna background (containerColor).

- Pada baris 112-165, **Row** merupakan komponen layout yang berlawanan dari Column, Dimana jika Column menyusun elemen UI secara Vertical, maka Row menyusun elemen UI secara horizontal.
  - Pada baris 112-165, **modifier** berfungsi untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.padding* untuk memberikan jarak dalam component, *.fillMaxWidth()* untuk membuat lebar component lebarnya selayar.
  - Pada baris 122 dan 165, **horizontalArrangement** berfungsi untuk memberi jarak maksimal antar elemen.
- Pada baris 116 dan 153, **Img()** merupakan fungsi yang saya buat sendiri untuk memudahkan dalam memuat gambar.
- Pada baris 117 dan 148, **Column()** merupakan komponen layout mirip seperti scaffold, hanya saja Column digunakan untuk menyusun elemen UI secara vertical.
  - Pada baris 148, **modifier** berfungsi untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.padding* untuk memberikan jarak dalam dan *verticalScroll* untuk membuat konten dapat di scroll secara vertical.
  - Pada baris 151, **horizontalAlignment** berfungsi untuk mengatur posisi elemen secara horizontal
- Pada baris 119 dan 162, **Desc()** merupakan fungsi yang saya buat sendiri untuk memudahkan dalam membuat deskripsi.
- Pada baris 120-163, **Spacer()** berfungsi untuk memberikan jarak kosong biasanya secara vertical dalam layout.
- Pada baris 126-172, **Button()** merupakan komponen UI yang berfungsi untuk membuat tombol pada aplikasi.

    - Pada baris 135, **onClick** merpakan parameter dalam Button() untuk memberikan aksi yang akan dijalankan ketika tombol ditekan.
    - Pada baris 146, **modifier** merupakan parameter dalam TextField untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.fillMaxWidth* agar column lebar Button mengisi ukuran layar.
    - Pada baris 148, **colors** merupakan parameter dalam Button() untuk mengatur warna background (containerColor) dan warna teks (titleContentColor).
    - Pada baris 152, **shape** merupakan parameter dalam Button() untuk mengatur sudut agar dapat membulat.

- **AirlineData.kt**
    - Pada baris 1, **package com.android.modul3** pendeklarasian nama package file Kotlin.
    - Pada baris 3 dan 49, **data class** meerupakan jenis kelas untuk menyimpan struktur data.
    - Pada baris 4-57, **val** merupakan keyword untuk membuat suatu variable bersifat immutable yang menampung nilai dari berbagai tipe data.
    - Pada baris 10 dan 57 **listOf** digunakan untuk membuat list dari element dengan sifat immutable.
- **UIComponent.kt**
    - Pada baris 1, **package com.android.modul3** pendeklarasian nama package file Kotlin.
    - Pada baris 3-20, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.

- Pada baris 22-53, **@Composable** merupakan annotation untuk menandai bahwa fungsi yang dibuat dibawahnya merupakan fungsi UI di jetpack compose.
- Pada baris 23, **fun Desc(desc: String)** merupakan fungsi yang berisikan Text dengan style tertentu untuk deskripsi.
- pada baris 24-65, **Text()** merupakan komponen UI yang berfunsi menampilkan teks.
    a. pada baris 2-61, **text** merupakan isi dari teks yang akan ditampilkan
    b. pada baris 26 dan 46, **fontSize** merupakan ukuran dari font yang akan ditampilkan
    c. pada baris 27 dan 47, **textAlign** untuk membuat apakah teks berada di kiri, Tengah , atau kanan
    d. pada baris 48 dan 63, **fontWeight** merupakan ketebalan dari teks yang akan ditampilkan
    e. pada baris 49 dan 62, **modifier** berfungsi untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.fillMaxWidth* agar column lebar Button mengisi ukuran layar.
- Pada baris 30, **@OptIn(ExperimentalGlideComposeApi::*class*)** berfungsi untuk memberitahukan compiler bahwa fungsi composable dibawahnya masih menggunakan API yang bersifat eksperimental untuk menghindari error saat kompilasi.
- Pada baris 32, **fun Img(url: String, width: Int)** merupakan fungsi yang berisikan gambar dengan style yang sudah ditentukan.
- Pada baris 33, **GlideImage** merupakan library glide yang digunakan untuk menampilakn gambar dar internet melalui URL
    a. Pada baris 34, **model** merupakan tempat Dimana URL diletakkan
    b. Pada baris 35, **contentDescription** merupakan deskripsi dari gambar yang akan ditampilkan

c. Pada baris 36, **modifier** merupakan parameter dalam TextField untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.width* untuk lebar, *.height* untuk tinggi, dan *.clip* untuk mengatur lengkungan gambar.

- Pada baris 43, **fun Title(title: String)** merupakan fungsi uang bersikan Text dengan style yang telah ditentukan untuk judul.
- Pada baris 33, **fun DetailRow(label:String, value:String)** merupakan fungsi untuk berisikan sepasang Text untuk halaman detail.
- Pada baris 55, **Row** merupakan komponen layout yang berlawanan dari Column, Dimana jika Column menyusun elemen UI secara Vertical, maka Row menyusun elemen UI secara horizontal.
    a. Pada baris 94-124, **modifier** merupakan parameter dalam TextField untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.fillMaxWidth* agar column lebar Row mengisi ukuran layar.

## SOAL 2

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

- Karena saat ini masih banyak project-project android yang masih belum melakukan migrasi ke jetpack compose, sehingga masih berbasis XML nah oleh sebab itu RecycleVIew masih banyak digunakan. Selain itu juga, di RecycleView lebih memberikan control kepada pengguna untuk melakukan layoutManager Kustom, Animation transition, Item Decoration, Dimana hal ini belum bisa dilakukan oleh LazyColumn, dan dalam aplikasi yang sudah berskala besar, RecycleView sering dioptimasikan hingga Tingkat yang sangat rendah yang Dimana hal ini belum bisa dilakukan di LazyColumn.

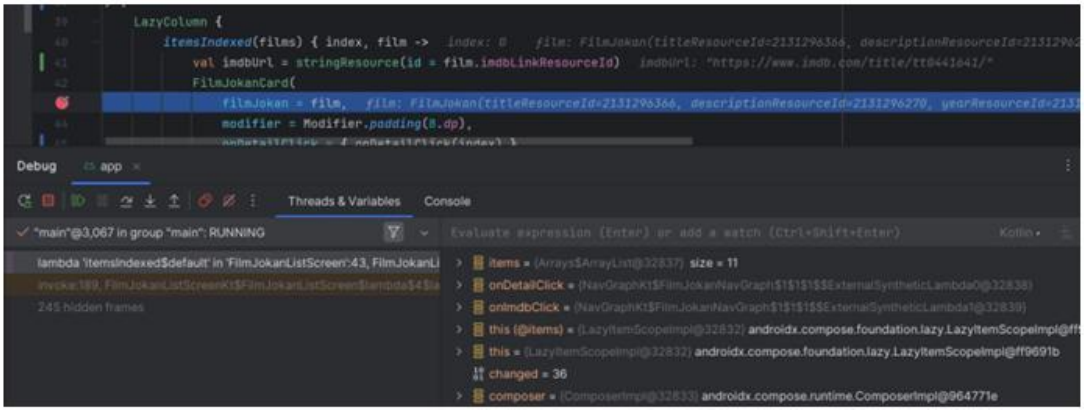
# MODUL 4 : VIEWMODEL AND DEBUGGING

## SOAL 1

**Soal Praktikum:**
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 11. Contoh Penggunaan Debugger*

## A. Source Code

- **MainActivity.kt**

*Tabel 8. Source Code MainActivity Soal 1 Modul 4*

```
1   package com.android.modul4
2
3   import android.os.Bundle
4   import androidx.activity.ComponentActivity
5   import androidx.activity.compose.setContent
6   import androidx.activity.enableEdgeToEdge
7   import androidx.compose.foundation.layout.padding
8   import androidx.compose.material3.Scaffold
9   import androidx.compose.runtime.Composable
10  import androidx.compose.ui.Modifier
11  import androidx.lifecycle.viewmodel.compose.viewModel
12  import androidx.navigation.compose.NavHost
13  import androidx.navigation.compose.composable
14  import androidx.navigation.compose.rememberNavController
15  import com.android.modul4.data.sourceData
16  import com.android.modul4.screens.CardList
17  import com.android.modul4.screens.DetailPage
18  import com.android.modul4.ui.theme.Modul4Theme
19  import com.android.modul4.viewmodel.CardViewModel
20  import com.android.modul4.viewmodel.CardViewModelFactory
21
22  class MainActivity : ComponentActivity() {
23      override fun onCreate(savedInstanceState: Bundle?) {
24          super.onCreate(savedInstanceState)
25          enableEdgeToEdge()
26          setContent {
27              Modul4Theme {
28                  Display()
29              }
30          }
31      }
32  }
33
34  @Composable
35  fun Display() {
36      val navController = rememberNavController()
37      val viewModelFactory =
    CardViewModelFactory(sourceData())
38      val viewModel: CardViewModel = viewModel(factory =
    viewModelFactory)
39
40      Scaffold() { innerPadding ->
41          NavHost(
```

```
42              navController = navController,
43              startDestination = "card_list",
44              modifier = Modifier.padding(innerPadding)
45          ) {
46              composable("card_list") {
47                  CardList(navController, viewModel)
48              }
49              composable("detail/{itemTitle}") {
    backStackEntry ->
50                  val itemTitle =
    backStackEntry.arguments?.getString("itemTitle")!!
51                  DetailPage(itemTitle, navController,
    viewModel)
52              }
53          }
54      }
55  }
```

- **data/sourceData.kt**

*Tabel 9. Source Code sourceData Soal 1 Modul 4*

```
1   package com.android.modul4.data
2
3   import com.android.modul4.models.CardProp
4
5   class sourceData {
6       fun loadCardProps(): List<CardProp> {
7           return listOf<CardProp>(
8               CardProp(
9                   title = "Garuda Indonesia",
10                  desc = "Garuda Indonesia adalah maskapai
    penerbangan nasional Indonesia yang didirikan pada tahun
    1949. Berbasis di Jakarta, Garuda dikenal dengan layanan
    penerbangannya yang berkualitas tinggi dan keramahannya,
    mencerminkan budaya Indonesia. Maskapai ini
    mengoperasikan penerbangan domestik dan internasional ke
    berbagai tujuan di Asia, Australia, Eropa, dan Timur
    Tengah. Garuda Indonesia juga merupakan anggota dari
    aliansi penerbangan global SkyTeam dan telah beberapa
    kali meraih penghargaan dunia atas pelayanan kabinnya.",
11                  ImageURL =
    "https://cdn.plnspttrs.net/11964/pk-gib-garuda-indonesia-
    mcdonnell-douglas-dc-10-
    30_PlanespottersNet_714188_a594861cb0_o.jpg",
12                  Wiki =
    "https://id.wikipedia.org/wiki/Garuda_Indonesia",
13                  tglBerdiri = "1 Agustus 1947",
14                  armada = "77",
```

```
15                  rute = "Belanda, Thailand, China, Hong
    Kong, Arab Saudi, Malaysia, Arab Saudi, Australia, Korea
    Selatan, Singapura, Jepang, United Arab Emirates",
16                  website = "https://www.garuda-
    indonesia.com/id/id//",
17              ),
18              CardProp(
19                  title = "Lion Air",
20                  desc = "Lion Air adalah maskapai
    penerbangan swasta terbesar di Indonesia yang berdiri
    pada tahun 1999. Fokus utamanya adalah layanan
    penerbangan berbiaya rendah (low-cost carrier) dengan
    rute domestik dan internasional. Lion Air terkenal dengan
    jaringan penerbangan yang luas dan harga tiket yang
    kompetitif.",
21                  ImageURL =
    "https://cdn.plnspttrs.net/20446/pk-lhg-lion-air-boeing-
    747-412_PlanespottersNet_612707_2702071c86_o.jpg",
22                  Wiki =
    "https://id.wikipedia.org/wiki/Lion_Air",
23                  tglBerdiri = "15 November 1999",
24                  armada = "100",
25                  rute = "Arab Saudi, China, Malaysia",
26                  website = "https://www.lionair.co.id/",
27              ),
28              CardProp(
29                  title = "Citilink",
30                  desc = "Citilink adalah anak perusahaan
    dari Garuda Indonesia yang beroperasi sebagai maskapai
    berbiaya rendah. Didirikan pada tahun 2001, Citilink
    menawarkan penerbangan domestik dan regional dengan
    konsep layanan yang modern, dinamis, dan lebih santai
    untuk menarik kalangan muda dan pelancong bisnis.",
31                  ImageURL =
    "https://cdn.plnspttrs.net/42179/pk-gaf-citilink-atr-72-
    600-72-212a_PlanespottersNet_1657140_83de625cfa_o.jpg",
32                  Wiki =
    "https://id.wikipedia.org/wiki/Citilink",
33                  tglBerdiri = "16 Juli 2001",
34                  armada = "57",
35                  rute = "Australia, Malaysia, Papua
    Nugini, Singapura, Timor Leste, China",
36                  website = "https://www.citilink.co.id/",
37              ),
38              CardProp(
39                  title = "Super Air Jet",
40                  desc = "Super Air Jet adalah maskapai
    baru di Indonesia yang mulai beroperasi pada tahun 2021.
    Mengusung konsep \"new lifestyle airline\", Super Air Jet
```

```
    fokus melayani segmen anak muda dengan harga terjangkau,
    desain modern, dan rute-rute domestik populer.",
41                  ImageURL =
    "https://cdn.plnspttrs.net/14570/pk-std-super-air-jet-
    airbus-a320-232-
    wl_PlanespottersNet_1755960_137445c980_o.jpg",
42                  Wiki =
    "https://www.superairjet.com/en/about.php",
43                  tglBerdiri = "1 March 2021",
44                  armada = "61",
45                  rute = "Malaysia",
46                  website =
    "https://checkin.superairjet.com/dx/IUCI/",
47              ),
48              CardProp(
49                  title = "Batik Air",
50                  desc = "Batik Air adalah maskapai layanan
    penuh (full-service) dari Lion Air Group yang didirikan
    pada tahun 2013. Batik Air menawarkan fasilitas premium
    seperti makanan dalam penerbangan dan hiburan di kursi,
    serta menghubungkan berbagai kota besar di Indonesia dan
    Asia.",
51                  ImageURL =
    "https://cdn.plnspttrs.net/35032/pk-lug-batik-air-airbus-
    a320-214-wl_PlanespottersNet_1693958_5bbffba586_o.jpg",
52                  Wiki =
    "https://id.wikipedia.org/wiki/Batik_Air",
53                  tglBerdiri = "10 Juni 2012",
54                  armada = "65",
55                  rute = "Australia, Arab Saudi, China,
    India, Malaysia, Singapura, Thailand",
56                  website = "https://www.batikair.com.my/",
57              ),
58              CardProp(
59                  title = "Pelita Air",
60                  desc = "Pelita Air adalah maskapai yang
    awalnya fokus pada penerbangan carter dan layanan energi
    (minyak dan gas), namun sejak 2022 mulai mengembangkan
    penerbangan reguler domestik. Sebagai anak usaha
    Pertamina, Pelita Air membawa konsep layanan penerbangan
    yang nyaman dan profesional.",
61                  ImageURL =
    "https://cdn.plnspttrs.net/22490/pk-pwd-pelita-air-
    service-airbus-a320-214-
    wl_PlanespottersNet_1698319_901e5c6a31_o.jpg",
62                  Wiki =
    "https://id.wikipedia.org/wiki/Pelita_Air",
63                  tglBerdiri = "24 Januari 1970",
64                  armada = "33",
```

```
                    rute = "Indonesia",
                    website = "https://www.pelita-air.com/",
                )
            )
        }
}
```

- **models/CardProp.kt**

*Tabel 10. Source Code CardProp Soal 1 Modul 4*

```
package com.android.modul4.models

data class CardProp(
    val title: String,
    val desc: String,
    val ImageURL: String,
    val Wiki: String,
    val tglBerdiri: String,
    val armada: String,
    val rute: String,
    val website:String
)
```

- **screens/cardDetailScreen.kt**

*Tabel 11. Source Code cardDetailScreen Soal 1 Modul 4*

```
package com.android.modul4.screens

import android.content.Intent
import android.net.Uri
import android.util.Log
import androidx.compose.foundation.layout.Arrangement
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.Row
import androidx.compose.foundation.layout.Spacer
import androidx.compose.foundation.layout.fillMaxWidth
import androidx.compose.foundation.layout.height
import androidx.compose.foundation.layout.padding
import androidx.compose.foundation.rememberScrollState
import
androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.foundation.verticalScroll
import androidx.compose.material3.Button
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
```

```
20 import androidx.compose.ui.Modifier
21 import androidx.compose.ui.platform.LocalContext
22 import androidx.compose.ui.unit.dp
23 import androidx.navigation.NavController
24 import com.android.modul4.Desc
25 import com.android.modul4.DetailRow
26 import com.android.modul4.Img
27 import com.android.modul4.Title
28 import com.android.modul4.viewmodel.CardViewModel
29
30 @Composable
31 fun DetailPage(itemTitle: String, navController:
   NavController, viewModel: CardViewModel) {
32     val context = LocalContext.current
33     val detailMaskapai =
   viewModel.getDetailByTitle(itemTitle)
34
35     if (detailMaskapai != null) {
36         Column(modifier = Modifier
37             .padding(16.dp)
38             .verticalScroll(rememberScrollState()),
39             horizontalAlignment =
   Alignment.CenterHorizontally
40         ) {
41             Img(detailMaskapai.ImageURL, 400)
42             Title(itemTitle)
43             Spacer(modifier = Modifier.height(16.dp))
44             detailMaskapai?.let {
45                 DetailRow("Tanggal Berdiri", ":
   ${it.tglBerdiri}")
46                 DetailRow("Armada", ": ${it.armada}")
47                 DetailRow("Rute Tujuan", ": ${it.rute} ")
48             }
49             Spacer(modifier = Modifier.height(16.dp))
50             Desc(detailMaskapai.desc)
51             Spacer(modifier = Modifier.height(16.dp))
52
53             Row(modifier =
   Modifier.fillMaxWidth(),horizontalArrangement =
   Arrangement.SpaceBetween) {
54                 Button(onClick = {
55                     Log.d("Detail", "tombol website
   ${itemTitle} ditekan")
56                     val intent =
   Intent(Intent.ACTION_VIEW, Uri.parse(detailMaskapai?.let
   {it.website}))
57                     context.startActivity(intent)
58                 }, modifier = Modifier.padding(horizontal
   = 6.dp),
```

```
59                      shape = RoundedCornerShape(8.dp)
60                  ) { Text("Web $itemTitle") }
61                  Button(onClick = {
62                      navController.navigate("card_list")
    }, shape = RoundedCornerShape(8.dp)) { Text("Kembali") }
63              }
64          }
65      }
66      else {
67          Column(modifier = Modifier.padding(16.dp)) {
68              Text("Data tidak ditemukan untuk
    \"$itemTitle\"")
69              Button(onClick = {
    navController.navigate("card_list") }) {
70                  Text("Kembali")
71              }
72          }
73      }
74  }
```

- **screens/cardListScreen.kt**

*Tabel 12. Source Code cardListScreen Soal 1 Modul 4*

```
1   package com.android.modul4.screens
2
3   import android.content.Intent
4   import android.net.Uri
5   import android.util.Log
6   import androidx.compose.foundation.background
7   import androidx.compose.foundation.layout.Arrangement
8   import androidx.compose.foundation.layout.Column
9   import androidx.compose.foundation.layout.Row
10  import androidx.compose.foundation.layout.Spacer
11  import androidx.compose.foundation.layout.fillMaxWidth
12  import androidx.compose.foundation.layout.height
13  import androidx.compose.foundation.layout.padding
14  import androidx.compose.foundation.lazy.LazyColumn
15  import
    androidx.compose.foundation.shape.RoundedCornerShape
16  import androidx.compose.material3.Button
17  import androidx.compose.material3.Card
18  import androidx.compose.material3.CardDefaults
19  import androidx.compose.material3.Text
20  import androidx.compose.runtime.Composable
21  import androidx.compose.runtime.collectAsState
22  import androidx.compose.runtime.getValue
23  import androidx.compose.ui.Modifier
24  import androidx.compose.ui.graphics.Color
```

```
25 import androidx.compose.ui.platform.LocalContext
26 import androidx.compose.ui.unit.dp
27 import androidx.navigation.NavController
28 import com.android.modul4.Desc
29 import com.android.modul4.Img
30 import com.android.modul4.Title
31 import com.android.modul4.viewmodel.CardViewModel
32
33 @Composable
34 fun CardList( navController: NavController, viewModel:
   CardViewModel) {
35     val CardList by viewModel.cardList.collectAsState()
36     val context = LocalContext.current
37     val bgcard = Color(0xFF7AE2CF)
38     val bgcolor = Color(0xFFFFFDF6)
39
40     Title("Maskapai-maskapai Penerbangan di Indonesia")
41     LazyColumn(
42         modifier = Modifier
43             .padding(top = 50.dp)
44             .background(bgcolor)
45     ){
46         items(CardList.size) { index ->
47             val property = CardList[index]
48             Card(
49                 modifier = Modifier
50                     .fillMaxWidth()
51                     .padding(6.dp)
52                     .height(200.dp),
53                 colors =
   CardDefaults.cardColors(containerColor = bgcard),
54             ) {
55                 Row(
56                     modifier = Modifier
57                         .padding(6.dp),
58                 ) {
59                     Img(property.ImageURL, 180)
60                     Column {
61                         Title(property.title)
62                         Desc(if(property.desc.length >
   80) property.desc.take(80) + "..." else property.desc)
63                         Spacer(modifier =
   Modifier.weight(1f))
64                         Row(
65                             horizontalArrangement =
   Arrangement.End,
66                             modifier = Modifier
67                                 .fillMaxWidth()
68                         ) {
```

```
69                                  Button(onClick = {
70
    viewModel.selectCard(property, "wiki")
71                                      val intent =
    Intent(Intent.ACTION_VIEW, Uri.parse(property.Wiki))
72
    context.startActivity(intent)
73                                  }, modifier =
    Modifier.padding(horizontal = 6.dp),
74                                      shape =
    RoundedCornerShape(8.dp)
75                                  ) { Text("Wiki") }
76                                  Button(onClick = {
77
    viewModel.selectCard(property, "Detail")
78
    Log.d("cardListScreen","pindah ke halaman detail dari
    item ${property.title} ")
79
    navController.navigate("detail/${property.title}")
80                                  }, shape =
    RoundedCornerShape(8.dp)) { Text("Detail") }
81                              }
82                          }
83                      }
84                  }
85              }
86          }
87  }
```

- **screens/CardViewModel.kt**

*Tabel 13. Source Code CardViewModel Soal 1 Modul 4*

```
1   package com.android.modul4.viewmodel
2
3   import android.util.Log
4   import androidx.lifecycle.ViewModel
5   import com.android.modul4.data.sourceData
6   import com.android.modul4.models.CardProp
7   import kotlinx.coroutines.flow.MutableStateFlow
8   import kotlinx.coroutines.flow.StateFlow
9
10  class CardViewModel(private val dataSource: sourceData) :
    ViewModel() {
11      private val _cardList =
    MutableStateFlow<List<CardProp>>(emptyList())
12      val cardList: StateFlow<List<CardProp>> = _cardList
13
```

```
14      private val _selectCard =
    MutableStateFlow<CardProp?>(null)
15      val selectedCard: StateFlow<CardProp?> = _selectCard
16
17      init { loadData() }
18
19      private fun loadData() {
20          val cards = dataSource.loadCardProps()
21          _cardList.value = cards
22          Log.d("CardViewMode","Card yang tampil ada
    ${cards.size} ")
23      }
24      fun getDetailByTitle(Title: String): CardProp? {
25          return _cardList.value.find { it.title == Title }
26      }
27      fun selectCard(card: CardProp, name: String) {
28          _selectCard.value = card
29          Log.d("CardViewModel", "tombol ${name}
    ${card.title} ditekan")
30      }
31  }
```

- **screens/CardViewModelFactory.kt**

*Tabel 14. Source Code CardViewModelFactory Soal 1 Modul 4*

```
1   package com.android.modul4.viewmodel
2
3   import androidx.lifecycle.ViewModel
4   import androidx.lifecycle.ViewModelProvider
5   import com.android.modul4.data.sourceData
6
7   class CardViewModelFactory(private val dataSource:
    sourceData) : ViewModelProvider.Factory {
8       override fun <T : ViewModel> create(modelClass:
    Class<T>): T {
9           if (modelClass.isAssignableFrom(CardViewModel::
    class.java)) {
10              return CardViewModel(dataSource) as T
11          }
12          throw IllegalArgumentException("Unknown
    ViewModel class")
13      }
14  }
```

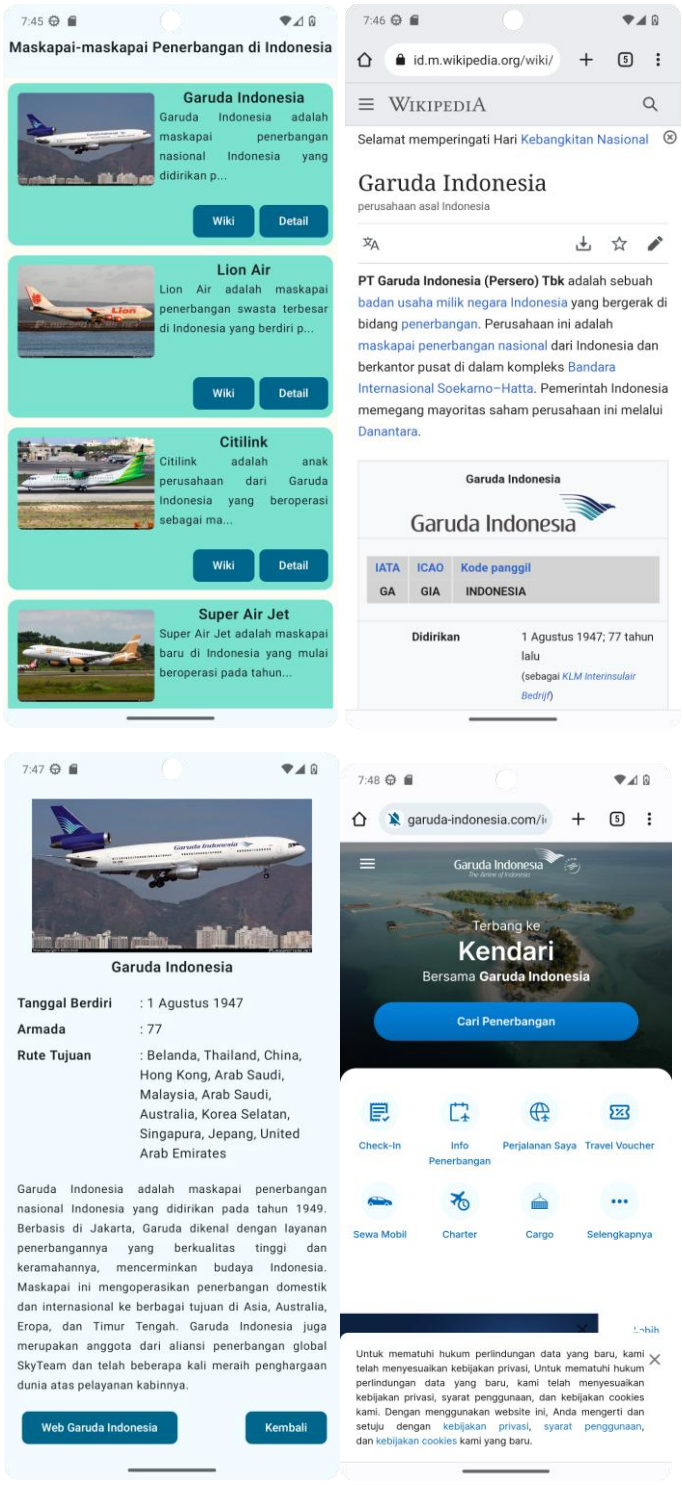
- **UIComponent.kt**

*Tabel 15. Source Code UIComponent Soal 1 Modul 4*

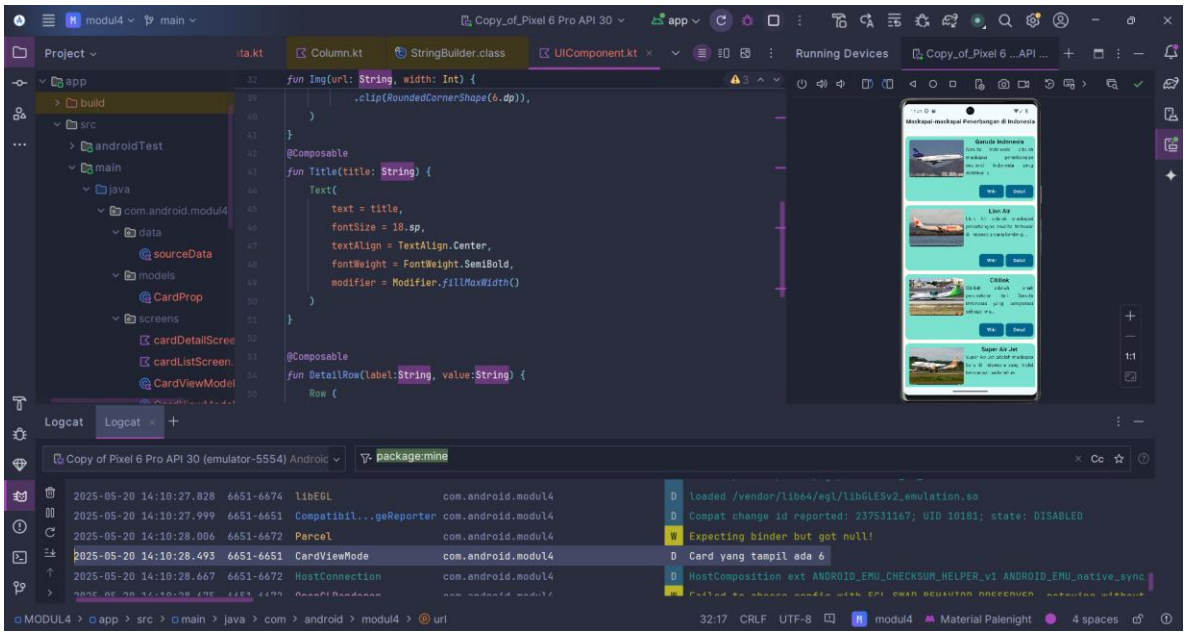
```
1   package com.android.modul3
2
3   import android.content.Intent
4   import android.net.Uri
5   import androidx.compose.foundation.layout.Row
6   import androidx.compose.foundation.layout.fillMaxWidth
7   import androidx.compose.foundation.layout.padding
8   import androidx.compose.foundation.layout.width
9   import
    androidx.compose.foundation.shape.RoundedCornerShape
10  import androidx.compose.material3.Button
11  import androidx.compose.material3.Text
12  import androidx.compose.runtime.Composable
13  import androidx.compose.ui.Modifier
14  import androidx.compose.ui.draw.clip
15  import androidx.compose.ui.text.font.FontWeight
16  import androidx.compose.ui.text.style.TextAlign
17  import androidx.compose.ui.unit.dp
18  import androidx.compose.ui.unit.sp
19  import
    com.bumptech.glide.integration.compose.ExperimentalGlideC
    omposeApi
20  import com.bumptech.glide.integration.compose.GlideImage
21
22  @Composable
23  fun Desc(desc: String) {
24      Text(
25          text = desc,
26          fontSize = 14.sp,
27          textAlign = TextAlign.Justify
28      )
29  }
30  @OptIn(ExperimentalGlideComposeApi::class)
31  @Composable
32  fun Img(url: String, width: Int) {
33      GlideImage(
34          model = url,
35          contentDescription = "My Image",
36          modifier = Modifier
37              .width(width.dp)
38              .padding(6.dp)
39              .clip(RoundedCornerShape(6.dp)),
40      )
41  }
42  @Composable
43  fun Title(title: String) {
```

```
44      Text(
45          text = title,
46          fontSize = 18.sp,
47          textAlign = TextAlign.Center,
48          fontWeight = FontWeight.SemiBold,
49          modifier = Modifier.fillMaxWidth()
50      )
51  }
52
53  @Composable
54  fun DetailRow(label:String, value:String) {
55      Row (
56          modifier = Modifier
57              .fillMaxWidth()
58              .padding(vertical = 4.dp)
59      ) {
60          Text(
61              text = label,
62              modifier = Modifier.width(150.dp),
63              fontWeight = FontWeight.SemiBold
64          )
65          Text(value)
66      }
67  }
```

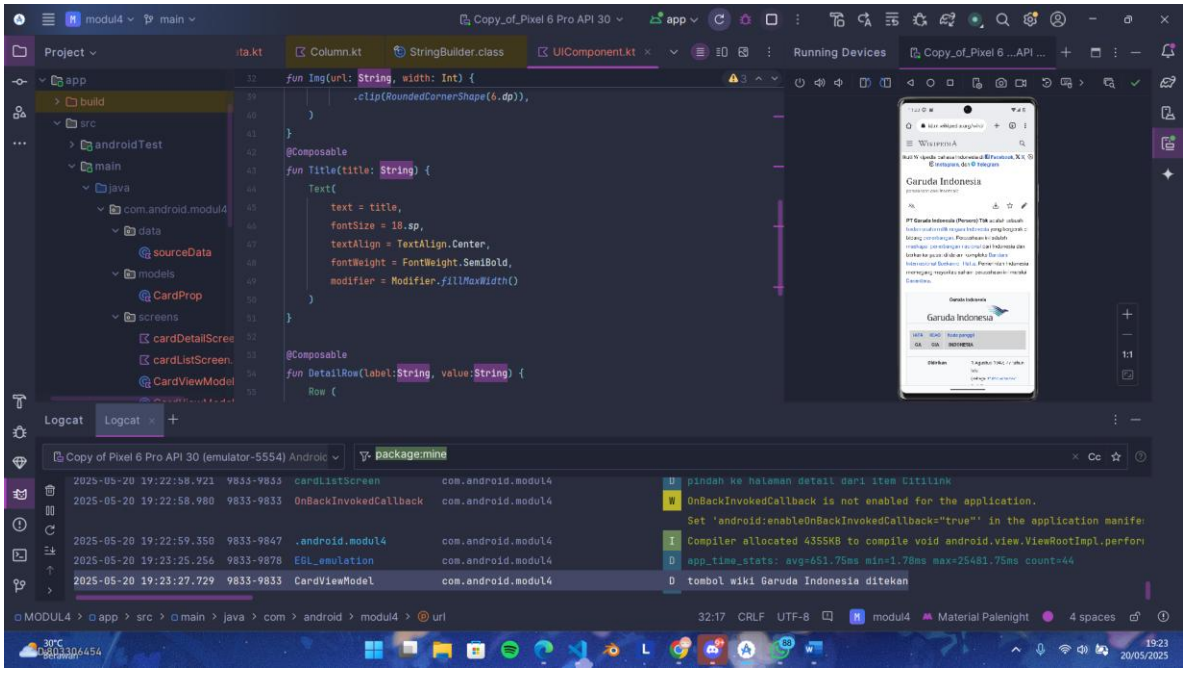
## B. Output Program



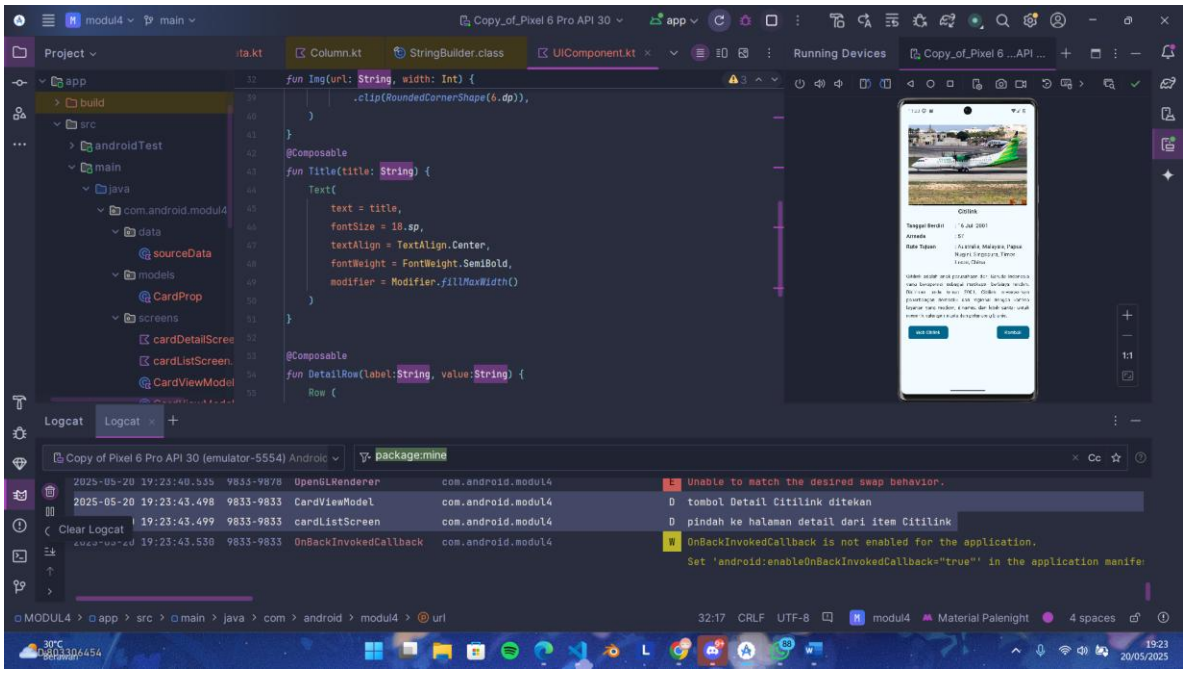

*Gambar 12. Screenshot Hasil Jawaban Soal 1 Modul 4*



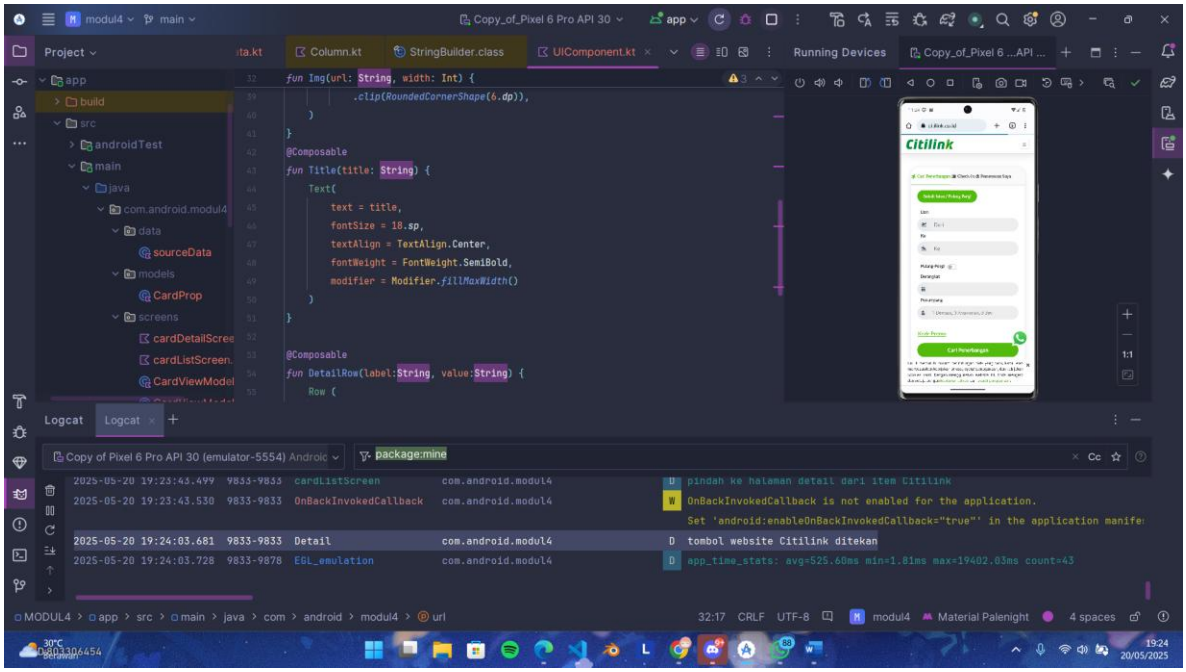
*Gambar 13. screenshot saat data item masuk list Soal 1 Modul 4*



*Gambar 14. screenshot saat tombol explicit intent ditekan Soal 1 Modul 4*




*Gambar 15. screenshot data dari list yang dipilih ketika ke halaman detail Soal 1 Modul 4*



## C. Pembahasan

- **MainActivity.kt:**
  - Pada baris 1, **package com.android.modul4** pendeklarasian nama package file Kotlin.
  - Pada baris 3-20, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
  - Pada baris 22, **class MainActivity : ComponentActivity(),** merupakan titik mula yang menjadi kelas utama dan akan dijalankan pertama kali saat aplikasi dibuka
  - Pada baris 23, **override fun onCreate(savedInstanceState: Bundle?),** berfungsi untuk menimpa (override) fungsi oncreate dari ComponentActivity.
  - Pada baris 24, **super.onCreate(savedInstanceState),** berfungsi untuk memanggil superclass dari fungsi oncreate untuk memastikan bahwa proses inisialisasi standar dari Android dijalankan sebelum logika saya dijalankan.
  - Pada baris 25, **enableEdgeToEdge(),** berfungsi agar tampilan aplikasi dapat menggunakan seluruh layar dari status bar sampai navigation bar atau fullscreen layout.
  - Pada baris 26, **setContent(),** digunakan untuk menampilkan UI berbasis jetpack compose ke dalam activity
  - Pada baris 27, **Modul4Theme,** merupakan fungsi yang berisi tema custom yang membungkus seluruh UI untuk memberikan style yang konsisten.
  - Pada baris 28, **Display(),** merupakan fungsi yang dibuat untuk menampilkan aplikasi scrollable list ke dalam activity.
  - Pada baris 34, **@composable,** merupakan annotation untuk menandai bahwa fungsi yang dibuat dibawahnya merupakan fungsi UI di jetpack compose.

- Pada baris 35, **fun Display(),** merupakan fungsi yang dibuat untuk menampilkan semua composable yang ingin ditampilkan ke dalam activity, seperti CardList() dan DetailPage() lalu menerima parameter navController.
- Pada baris 36-50, **val** merupakan keyword untuk membuat suatu variable bersifat immutable yang menampung nilai dari berbagai tipe data.
- Pada baris 36, **val navController = rememberNavController()** berfungsi untuk membuat dan menyimpan instance dari NavController
- Pada baris 37, **val viewModelFactory = CardViewModelFactory(sourceData())** berfungsi untuk membuat objek dari factory untuk membuat instance dari CardViewModel
- Pada baris 38, **val viewModel: CardViewModel = viewModel(factory = viewModelFactory)** berfungsi untuk memanggil fungsi viewModel(), sehingga viewModel dapat digunakan untuk mengakses data dan fungsi.
- Pada baris 40, **scaffold()**, merupakan komponen layout yang menjadi kerangka dasar dari tampilan yang mempermudah dalam mengatur strktur UI karena sudah disediakan slot-slot bawannya.
- Pada baris 41, **NavHost(),** berfungsi untuk menyediakan wadah untuk navigasi antar composable screen berdasarkan rute yang ditentukan
    - Pada baris 42, **navController** merupakan controller yang bertanggung jawab atas navigasi.
    - Pada baris 43, **startDestination** merupakan rute awal yang ditampilkan ketika aplikasi dijalankan.
    - Pada baris 44, modifier = **Modifier** berfungsi untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.padding* untuk memberikan jarak dalam component.
- Pada baris 46 dan 49, **composable(),** berfungsi untuk mendefinisikan rute, seperti di kasus ini ada “card_list” dan “detail”

- Pada baris 47, **CardList(navController, ViewModel),** merupakan fungsi untuk menampilkan list yang dapat scroll sekaligus menjadi halaman utama aplikasi dan menerima parameter navController dan ViewModel.
- pada baris 50, **itemTitle** berfungsi untuk mengambil parameter di navigasi melalui backstackentry.
- pada baris 51, **DetailPage(itemTitl, navController, ViewModel)** merupakan fungsi untuk menampilkan halaman detail dari aplikasi dan menerima beberapa parameter.

- **data/sourceData.kt**
  - Pada baris 1, **package com.android.modul4.data** pendeklarasian nama package file Kotlin.
  - Pada baris 3, **import com.android.modul4.models.CardProp** berfungsi untuk mengimport data class dari file CardProp.
  - Pada baris 5, **class sourceData** berfungsi untuk membuat sebuah class dengan nama sourceData.
  - Pada baris 6, **fun loadCardProps(): List<CardProp>** merupakan fungsi dengan tujuan untuk mengembalikan list dari objek CardProp yang berisikan data-data
  - Pada baris 7, **return listOf<CardProp>** merupakan list immutable yang akan dikembalikan

1. **models/CardProp.kt**
   - Pada baris 1, **package com.android.modul4.models** pendeklarasian nama package file Kotlin.
   - Pada baris 3, **data class** meerupakan jenis kelas untuk menyimpan struktur data.
   - Pada baris 4-11, **val** merupakan keyword untuk membuat suatu variable bersifat immutable yang menampung nilai dari berbagai tipe data.

- **screens/cardDetailScreen.kt**

- Pada baris 1, **package com.android.modul4.screens** pendeklarasian nama package file Kotlin.
- Pada baris 3-28, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
- Pada baris 31, **fun DetailPage( navController: NavController, viewModel: CardViewModel),** merupakan fungsi untuk menampilkan detail dari card yang dipilih dan menerima parameter berupa navController dan viewModel.
- Pada baris 32, **LocalContext.current** berfungsi untuk mendapatkan context dari acrivity saat ini dalam fungsi composable dan juga digunakan untuk menampilkan Toast sebagai notifikasi dari peringatan jika user salah input.
- Pada baris 33, **val detailMaskapai = viewModel.getDetailByTitle(itemTitle)** berfungsi untuk memanggil fungsi viewModel lalu dapatkan detail data berdasarkan judul.
- Pada baris 34, **if** merupakan percabangan yang jika kondisinya terpenuhi maka isinya akan dieksekusi.
- Pada baris 36 dan 67, **Column()** merupakan komponen layout mirip seperti scaffold, hanya saja Column digunakan untuk menyusun elemen UI secara vertical.
    a. Pada baris 36 dan 67, **modifier** merupakan parameter dalam Column() untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.padding* untuk memberikan jarak dalam componen dan *.verticalScroll* agar bisa di scroll
    b. Pada baris 65, **horizontalAlignment** berfungsi untuk menentukan layout secara horizontal.
- Pada baris 41, **Img()** merupakan fungsi yang saya buat sendiri untuk memudahkan dalam memuat gambar.

- Pada baris 42, **Title()** merupakan fungsi yang saya buat sendiri untuk memudahkan dalam membuat judul.
- Pada baris 43-51, **Spacer()** berfungsi untuk memberikan jarak kosong biasanya secara vertical dalam layout.
- Pada baris 50, **Desc()** merupakan fungsi yang saya buat sendiri untuk memudahkan dalam membuat deskripsi.
- Pada baris 53, **Row** merupakan komponen layout yang berlawanan dari Column, Dimana jika Column menyusun elemen UI secara Vertical, maka Row menyusun elemen UI secara horizontal.
    a. Pada baris 53, **modifier** berfungsi untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.padding* untuk memberikan jarak dalam component, *.fillMaxWidth()* untuk membuat lebar component lebarnya selayar.
    b. Pada baris 53, **horizontalArrangement** berfungsi untuk memberi jarak maksimal antar elemen.
- Pada baris 54 dan 61, **Button()** merupakan komponen UI yang berfungsi untuk membuat tombol pada aplikasi.
    a. Pada baris 54 dan 61, **onClick** merpakan parameter dalam Button() untuk memberikan aksi yang akan dijalankan ketika tombol ditekan.
    b. Pada baris 55, **Log.d("Detail", "tombol website ${itemTitle} ditekan")** berfungsi untuk mencetak pesan ke logcat untuk debugging.
    c. Pada baris 56, **val intent = Intent(Intent.ACTION_VIEW, Uri.parse(detailMaskapai?.let {it.website}))** berfungsi untuk membuat intent secara eksplisit untuk membuka URL di browser.
    d. Pada baris 58, **modifier** merupakan parameter dalam TextField untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.fillMaxWidth* agar column lebar Button mengisi ukuran layar.

e. Pada baris 59 dan 62, **shape** merupakan parameter dalam Button() untuk mengatur sudut agar dapat membulat.

- **screens/cardListScreen.kt**
  - Pada baris 1, **package com.android.modul4.screens** pendeklarasian nama package file Kotlin.
  - Pada baris 3-31, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
  - Pada baris 34, **fun CardList( navController: NavController, viewModel: CardViewModel),** merupakan fungsi untuk menampilkan card-card dalam jumlah banyak yang dapat di scroll dan menerima parameter berupa navController dan viewModel.
  - Pada baris 35, **val CardList by viewModel.cardList.collectAsState()** berfungsi untuk mengamati stateflow dan dikonversi menjadi state compose, lalu akan mengambil nilai .value dari collectAsState lalu akan gunakan CardList untuk UI.
  - Pada baris 36, **LocalContext.current** berfungsi untuk mendapatkan context dari acrivity saat ini dalam fungsi composable dan juga digunakan untuk menampilkan Toast sebagai notifikasi dari peringatan jika user salah input.
  - Pada baris 40 dan 61, **Title()** merupakan fungsi yang saya buat sendiri untuk memudahkan dalam membuat judul.
  - Pada baris 41, **LazyColumn()** berfungsi untuk menampilkan daftar item secara vertical yang dapat di scroll tapi hanya yang terlihat di layar.
    a. Pada baris 42, **modifier** merupakan parameter dalam Column() untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.padding* untuk memberikan jarak dalam componen dan *.background* untuk memberikan warna latar.

- Pada baris 46, **item()** berfungsi untuk membuat item di dalam LazyColumn sebanyak jumlah data.
- pada baris 48, **Card()** berfungsi untuk membuat sebuah kartu.
    a. pada baris 49, **modifier** merupakan parameter dalam Column() untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.padding* untuk memberikan jarak dalam component, *.height* untuk tinggi component, dan *.fillMaxWidth()* untuk membuat lebar component lebarnya selayar.
    b. pada baris 53, **colors** merupakan parameter dalam Card() untuk mengatur warna background (containerColor).
- Pada baris 55 dan 64, **Row** merupakan komponen layout yang berlawanan dari Column, Dimana jika Column menyusun elemen UI secara Vertical, maka Row menyusun elemen UI secara horizontal.
    a. Pada baris 56 dan 66, **modifier** berfungsi untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.padding* untuk memberikan jarak dalam component, *.fillMaxWidth()* untuk membuat lebar component lebarnya selayar.
    b. Pada baris 65, **horizontalArrangement** berfungsi untuk memberi jarak maksimal antar elemen.
- Pada baris 59, **Img()** merupakan fungsi yang saya buat sendiri untuk memudahkan dalam memuat gambar.
- Pada baris 60, **Column()** merupakan komponen layout mirip seperti scaffold, hanya saja Column digunakan untuk menyusun elemen UI secara vertical.
- Pada baris 62, **Desc()** merupakan fungsi yang saya buat sendiri untuk memudahkan dalam membuat deskripsi.
- Pada baris 63, **Spacer()** berfungsi untuk memberikan jarak kosong biasanya secara vertical dalam layout.
- Pada baris 69 dan 76, **Button()** merupakan komponen UI yang berfungsi untuk membuat tombol pada aplikasi.

a. Pada baris 69 dan 76, **onClick** merpakan parameter dalam Button() untuk memberikan aksi yang akan dijalankan ketika tombol ditekan.
b. Pada baris 73, **modifier** merupakan parameter dalam TextField untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.fillMaxWidth* agar column lebar Button mengisi ukuran layar.
c. Pada baris 74 dan 80, **shape** merupakan parameter dalam Button() untuk mengatur sudut agar dapat membulat.

- **screens/CardViewModel.kt**
  - Pada baris 1, **package com.android.modul4.viewModel** pendeklarasian nama package file Kotlin.
  - Pada baris 3-8, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
  - Pada baris 10, **class CardViewModel(private val dataSource: sourceData) : ViewModel()** berfungsi untuk membuat kelas CardViewModel yang menerima parameter dataSource
  - Pada baris 11, **private val _cardList = MutableStateFlow<List<CardProp>>(emptyList())** berfungsi untuk mendeklarasikan stateflow secara mutable yang berisi list objek CardProp, lalu emptyList() untuk menandakan bahwa data awalnya kosong
  - Pada baris 12, **val cardList: StateFlow<List<CardProp>> = _cardList** berfungsi untuk memberikan akses agar UI dapat membaca dari _cardList
  - Pada baris 14, **private val _selectCard = MutableStateFlow<CardProp?>(null)** berfungsi untuk menyimpan satu objek CardProp yang dipilih oleh user

- Pada baris 15, **val selectedCard: StateFlow<CardProp?> = _selectCard** berfungsi untuk memberikan akses agar UI dapat membaca isi dari _selectCard
- Pada baris 17, **init { loadData() }** berfungsi untuk menjalankan fungsi loadData() saat CardViewModel dibuat
- Pada baris 19, **private fun loadData()** fungsi yang dibuat untuk mengambil data dari dataSource lalu akan disimpan ke _CardList agar UI mendapatkan data terbaru
- Pada baris 20, ***val* cards = dataSource.loadCardProps()** berfungsi untuk mengambil data dari dataSource
- Pada baris 21, **_cardList.value = cards** berfungsi untuk menyimpan data yang diambil tadi ke dalam _cardList
- Pada baris 22 dan 29, **Log.d()** berfungsi untuk mencetak pesan ke logcat untuk debugging.
- Pada baris 19, **fun getDetailByTitle(Title: String): CardProp?** Fungsi yang dibuat untuk mencari objek cardProp berdasarkan judul
- Pada baris 19, **return _cardList.value.find { it.title == Title }** berfungsi untuk mengembalikan nilai _cardList yang titlenya sama
- Pada baris 27, **fun selectCard(card: CardProp, name: String)** fungsi yang dibuat untuk menyimpan data card ke dalam _selectCard, sebagai tanda bahwa user memilih card tersebut

- **screens/CardViewModelFactory.kt**
  - Pada baris 1, **package com.android.modul4.viewmodel** pendeklarasian nama package file Kotlin.
  - Pada baris 3-5, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.

- Pada baris 7, **class CardViewModelFactory(private val dataSource: sourceData) : ViewModelProvider.Factory** berfungsi untuk membuat kelas CardViewModelFactory untuk mengimplementasikan interface ViewModelProvider.Factory.
- Pada baris 8, **override *fun* <T : ViewModel> create(modelClass: Class<T>): T** berfungsi untuk melakukan override terhadap method create() lalu parameter modelClass akan menyatakan kelas ViewModel apa yang ingin dibuat.
- Pada baris 9, **if (modelClass.isAssignableFrom(CardViewModel::class.java))** berfungsi untuk mengecek apakah class yang diminta oleh modelclass apakah adalah CardViewModel atau subclassnya
- Pada baris 10, **return CardViewModel(dataSource) as T** berfungsi untuk membuat instance CardViewModel menggunakan dataSource dan dikembalikan sebagai dengan tipe T
- Pada baris 10, **throw IllegalArgumentException("Unknown ViewModel class")** berfungsi untuk melemparkan error jika ternyata modeclassnya bukan CardViewModel

- **UIComponent.kt**
  - Pada baris 1, **package com.android.modul3** pendeklarasian nama package file Kotlin.
  - Pada baris 3-20, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
  - Pada baris 22-53, **@Composable** merupakan annotation untuk menandai bahwa fungsi yang dibuat dibawahnya merupakan fungsi UI di jetpack compose.

- Pada baris 23, **fun Desc(desc: String)** merupakan fungsi yang berisikan Text dengan style tertentu untuk deskripsi.
- pada baris 24-65, **Text()** merupakan komponen UI yang berfunsi menampilkan teks.
    a. pada baris 2-61, **text** merupakan isi dari teks yang akan ditampilkan
    b. pada baris 26 dan 46, **fontSize** merupakan ukuran dari font yang akan ditampilkan
    c. pada baris 27 dan 47, **textAlign** untuk membuat apakah teks berada di kiri, Tengah , atau kanan
    d. pada baris 48 dan 63, **fontWeight** merupakan ketebalan dari teks yang akan ditampilkan
    e. pada baris 49 dan 62, **modifier** berfungsi untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.fillMaxWidth* agar column lebar Button mengisi ukuran layar.
- Pada baris 30, **@OptIn(ExperimentalGlideComposeApi::*class*)** berfungsi untuk memberitahukan compiler bahwa fungsi composable dibawahnya masih menggunakan API yang bersifat eksperimental untuk menghindari error saat kompilasi.
- Pada baris 32, **fun Img(url: String, width: Int)** merupakan fungsi yang berisikan gambar dengan style yang sudah ditentukan.
- Pada baris 33, **GlideImage** merupakan library glide yang digunakan untuk menampilakn gambar dar internet melalui URL
    a. Pada baris 34, **model** merupakan tempat Dimana URL diletakkan
    b. Pada baris 35, **contentDescription** merupakan deskripsi dari gambar yang akan ditampilkan
    c. Pada baris 36, **modifier** merupakan parameter dalam TextField untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.width* untuk lebar, *.height* untuk tinggi, dan *.clip* untuk mengatur lengkungan gambar.

- Pada baris 43, **fun Title(title: String)** merupakan fungsi uang bersikan Text dengan style yang telah ditentukan untuk judul.
- Pada baris 33, **fun DetailRow(label:String, value:String)** merupakan fungsi untuk berisikan sepasang Text untuk halaman detail.
- Pada baris 55, **Row** merupakan komponen layout yang berlawanan dari Column, Dimana jika Column menyusun elemen UI secara Vertical, maka Row menyusun elemen UI secara horizontal.
    a. Pada baris 94-124, **modifier** merupakan parameter dalam TextField untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.fillMaxWidth* agar column lebar Row mengisi ukuran layar.

Jawaban e.

1. Penjelasan Debugger serta cara pakainya
    - Debugger adalah sebuah tool atau alat di android studio yang digunakan untuk menemukan atau memperbaiki bug dalam kode.
    - Cara pakai debugger:
        - Pertama, menentukan breakpoint dari kode kita. Biasanya dengan menekan nomor baris code hingga ada bulatan merah
        - Kedua, menjalankan aplikasi dalam mode debug dengan menekan tombol seperti serangga di bagian atas dekat tombol run
        - Ketiga, setelahnya program akan terhenti tepat di breakpoint yang sudah di atur tadi, Dimana disini kita dapat melihat informasi seperti nilai variable, informasi mengenai fungsi, serta status thread.
2. Penjelasan step into, step over, step out
    - Step Into,  berfungsi untuk masuk kedalam fungsi yang dipanggil di baris yang kita beri breakpoint tadi.
    - Step over, berfungsi saat eksekusi terhenti di breakpoint kita dapat melanjutkan ke baris selanjutnya dengan menekan step over

- Step Out, berfungsi untuk keluar dari dalam fungsi yang kita masuki dengan step into tadi

## SOAL 2

Jelaskan Application class dalam arsitektur aplikasi Android dan fungsinya

- Application Class merupakan komponen global yang aplikasi android untuk menyimpan dan mengatur state atau juga konfigurasi dari aplikasi yang dibutuhkan di seluruh bagian aplikasi.
- Application class juga berfungsi untuk:
    - Inisiasi secara global, Dimana application class berfunsi sebagai tempat yang sesuai atau ideal untuk menjalankan library seperti firebase, retrofit dan lain lain yang hanya akan dieksekusi sekali dalam satu lifecycle aplikasi.
    - Menyimpan state secara global, Dimana application class berfungsi sebagai tempat penyimpanan state aplikasi untuk kebutuhan seperti autentikasi yang membutuhkan tokennya dan data preferensi pengguna yang diperlukan komponen aplikasi
    - Mengelola lifecycle aplikasi, Dimana application class dapat digunakan untuk melakukan override pada onCreate() untuk melakukan eksekusi sesuatu pada saat aplikasi dijalankan pertama kali

# MODUL 5 : CONNECT TO THE INTERNET

## SOAL 1

**Soal Praktikum:**
Lanjutkan aplikasi Android yang sudah dibuat pada Modul 4 dengan menambahkan modifikasi

sesuai ketentuan berikut:

a. Gunakan networking library seperti Retrofit atau Ktor agar aplikasi dapat mengambil data dari remote API. Dalam penggunaan networking library, sertakan generic response untuk status dan error handling pada API dan Flow untuk data stream.
b. Gunakan KotlinX Serialization sebagai library JSON.
c. Gunakan library seperti Coil atau Glide untuk image loading.
d. API yang digunakan pada modul ini bebas, contoh API gratis The Movie Database (TMDB) API yang menampilkan data film. Berikut link dokumentasi API: https://developer.themoviedb.org/docs/getting-started
e. Implementasikan konsep data persistence (misalnya offline-first app, pengaturan dark/light mode, fitur favorite, dll)
f. Gunakan caching strategy pada Room..
g. Untuk Modul 5, bebas memilih UI yang ingin digunakan, antara berbasis XML atau Jetpack Compose.

Aplikasi harus mempertahankan fitur-fitur yang dibuat pada modul sebelumnya.

### A. Source Code

- **MainActivity.kt**

*Tabel 16. Source Code MainActivity Soal 1 Modul 5*

```
1   package com.android.modul5
2
3   import MovieListScreen
4   import android.content.Context
5   import android.os.Bundle
6   import androidx.activity.ComponentActivity
7   import androidx.activity.compose.setContent
8   import androidx.activity.enableEdgeToEdge
9   import androidx.compose.animation.slideInHorizontally
10  import androidx.compose.animation.slideOutHorizontally
```

```
11 import androidx.compose.material3.Text
12 import androidx.compose.runtime.Composable
13 import androidx.compose.runtime.getValue
14 import androidx.compose.runtime.mutableStateOf
15 import androidx.compose.runtime.remember
16 import androidx.compose.runtime.setValue
17 import androidx.compose.ui.platform.LocalContext
18 import androidx.core.content.edit
19 import androidx.lifecycle.viewmodel.compose.viewModel
20 import androidx.navigation.NavType
21 import androidx.navigation.compose.NavHost
22 import androidx.navigation.compose.composable
23 import androidx.navigation.compose.rememberNavController
24 import androidx.navigation.navArgument
25 import
   com.android.modul5.presentation.screens.MovieDetailScreen
26 import
   com.android.modul5.presentation.ui.theme.MODUL5Theme
27 import
   com.android.modul5.presentation.viewmodel.MovieViewModel
28
29 class MainActivity : ComponentActivity() {
30     override fun onCreate(savedInstanceState: Bundle?) {
31         super.onCreate(savedInstanceState)
32         enableEdgeToEdge()
33         setContent {
34             val context = LocalContext.current
35             val prefs = remember
   {context.getSharedPreferences("app_prefs",
   Context.MODE_PRIVATE)}
36             var isDarkMode by remember {
   mutableStateOf(prefs.getBoolean("is_dark_mode", false)) }
37
38             MODUL5Theme(darkTheme = isDarkMode) {
39                 NavMovies(
40                     isDarkMode = isDarkMode,
41                     onToggle = { newValue ->
42                         isDarkMode = newValue
43                         prefs.edit {
   putBoolean("is_dark_mode", newValue) }
44                     })
45             }
46         }
47     }
48 }
49
50 @Composable
51 fun NavMovies(isDarkMode: Boolean, onToggle: (Boolean) ->
   Unit) {
```

```
52      val navController = rememberNavController()
53      val movieViewModel: MovieViewModel = viewModel()
54
55      NavHost(
56          navController = navController,
57          startDestination = "movie_list",
58          enterTransition = {
59              slideInHorizontally(initialOffsetX = { it })
60          },
61          exitTransition = {
62              slideOutHorizontally(targetOffsetX = { -it })
63          },
64          popEnterTransition = {
65              slideInHorizontally(initialOffsetX = { -it })
66          },
67          popExitTransition = {
68              slideOutHorizontally(targetOffsetX = { it })
69          }
70      ) {
71          composable("movie_list") {
72              MovieListScreen(movieViewModel,
    navController, isDarkMode = isDarkMode, onToggle =
    onToggle)
73          }
74          composable(
75              route = "movie_detail/{movieId}",
76              arguments = listOf(
77                  navArgument("movieId") {
78                      type = NavType.IntType
79                  }
80              )
81          ) { backStackEntry ->
82              val movieId =
    backStackEntry.arguments?.getInt("movieId")
83              if (movieId != null) {
84                  MovieDetailScreen(movieId,
    movieViewModel, navController)
85              }
86              else {
87                  Text("Film tidak ditemukan!")
88              }
89          }
90      }
91  }
```

- **presentation/viewmodel/MovieViewModel.kt**

*Tabel 17. Source Code MovieViewModel Soal 1 Modul 5*

```
1  package com.android.modul5.presentation.viewmodel
2
3  import androidx.lifecycle.ViewModel
4  import androidx.lifecycle.viewModelScope
5  import com.android.modul5.data.api.RetrofitClient
6  import com.android.modul5.domain.model.Movie
7  import kotlinx.coroutines.flow.MutableStateFlow
8  import kotlinx.coroutines.flow.StateFlow
9  import kotlinx.coroutines.launch
10
11 class MovieViewModel: ViewModel() {
12     private val API_KEY =
   "9d2494a8a2a5c08592c8e963a74c799a"
13
14     private val _Movies =
   MutableStateFlow<List<Movie>>(emptyList())
15     val Movies: StateFlow<List<Movie>> = _Movies
16
17     private val _selectedMovieDetail =
   MutableStateFlow<Movie?>(null)
18     val selectedMovieDetail: StateFlow<Movie?> =
   _selectedMovieDetail
19
20     private val _errorMsg =
   MutableStateFlow<String?>(null)
21     val errorMsg: StateFlow<String?> = _errorMsg
22
23     init {
24         fetchMovies()
25     }
26
27     fun fetchMovies() {
28         viewModelScope.launch {
29             _errorMsg.value = null
30             val Response =
   RetrofitClient.tmdbAPI.getPopularMovies(API_KEY)
31             try {
32                 if (Response.isSuccessful) {
33                     Response.body()?.let { _Movies.value
   = it.results }
34                 }
35                 else {_errorMsg.value = "Error in movie
   List= ${Response.code()} - ${Response.message()}"}
36             }
37             catch (err: Exception) {_errorMsg.value =
   "Exception: ${err.localizedMessage} ?: Unknown error"}
```

```
        }
    }

    fun fetchMoviebyDetailbyID(movieId: Int) {
        _errorMsg.value = null
        viewModelScope.launch {
            val responseDetail =
RetrofitClient.tmdbAPI.getMovieDetailbyID(movieId,API_KEY
)
            try {
                if (responseDetail.isSuccessful) {
                    _selectedMovieDetail.value =
responseDetail.body()
                }
                else {_errorMsg.value = "Error in movie
detail = ${responseDetail.code()} -
${responseDetail.message()}"}
            }
            catch (err: Exception) {_errorMsg.value =
"Exception: ${err.localizedMessage} ?: Unknown error"}
        }
    }
}
```

- **presentation/screens/MovieDetailScreen.kt**

*Tabel 18. Source Code MovieDetailScreen Soal 1 Modul 5*

```
package com.android.modul5.presentation.screens

import androidx.compose.foundation.layout.Box
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.Spacer
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.height
import androidx.compose.foundation.layout.padding
import androidx.compose.foundation.layout.width
import androidx.compose.foundation.lazy.LazyColumn
import androidx.compose.foundation.lazy.items
import androidx.compose.foundation.rememberScrollState
import androidx.compose.foundation.verticalScroll
import androidx.compose.material3.Button
import androidx.compose.material3.Scaffold
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.runtime.DisposableEffect
import androidx.compose.runtime.LaunchedEffect
import androidx.compose.runtime.collectAsState
import androidx.compose.runtime.getValue
```

```
22 import androidx.compose.ui.Alignment
23 import androidx.compose.ui.Modifier
24 import androidx.compose.ui.focus.focusModifier
25 import androidx.compose.ui.text.font.FontWeight
26 import androidx.compose.ui.text.style.TextAlign
27 import androidx.compose.ui.unit.dp
28 import androidx.navigation.NavController
29 import
   com.android.modul5.presentation.components.ButtonNav
30 import com.android.modul5.presentation.components.Desc
31 import com.android.modul5.presentation.components.Glide
32 import
   com.android.modul5.presentation.components.MovieCard
33 import com.android.modul5.presentation.components.TopBar
34 import
   com.android.modul5.presentation.viewmodel.MovieViewModel
35 import
   com.bumptech.glide.integration.compose.ExperimentalGlideCo
   mposeApi
36 import com.bumptech.glide.integration.compose.GlideImage
37
38 @OptIn(ExperimentalGlideComposeApi::class)
39 @Composable
40 fun MovieDetailScreen(movieId: Int, movieViewModel:
   MovieViewModel, navController: NavController) {
41     val movieDetail by
   movieViewModel.selectedMovieDetail.collectAsState()
42     val errMsg by movieViewModel.errorMsg.collectAsState()
43
44     LaunchedEffect(movieId) {
45         movieViewModel.fetchMoviebyDetailbyID(movieId)
46     }
47
48     Scaffold(
49         topBar = { TopBar( Title = movieDetail?.title ?:
   "Detail Film") }
50     ) { innerPadding ->
51         Column(
52             modifier = Modifier
53                 .fillMaxSize()
54                 .padding(innerPadding)
55                 .padding(15.dp)
56                 .verticalScroll(rememberScrollState()),
57             horizontalAlignment =
   Alignment.CenterHorizontally
58         ) {
59             errMsg?.let {msg -> Text(msg) }
60             Box(
61                 modifier = Modifier
```

```
62                      .width(780.dp)
63                      .height(300.dp)
64              ) {
65                  Glide(movieDetail?.posterPath)
66              }
67              Spacer(modifier = Modifier.height(10.dp))
68              Desc("Tanggal Rilis",
    movieDetail?.releaseDate)
69              Desc("Popularitas", movieDetail?.popularity)
70              Desc("Rata-rata Vote",
    movieDetail?.voteAverage)
71              Spacer(modifier = Modifier.height(10.dp))
72              Text("${movieDetail?.overview}", textAlign =
    TextAlign.Justify)
73              Spacer(modifier = Modifier.height(10.dp))
74              ButtonNav("Kembali", navController)
75          }
76      }
77  }
```

- **presentation/screens/MovieListScreen.kt**

*Tabel 19. Source Code MovieListScreen Soal 1 Modul 5*

```
1   import androidx.compose.foundation.layout.Arrangement
2   import androidx.compose.foundation.layout.Column
3   import androidx.compose.foundation.layout.
    ExperimentalLayoutApi
4   import androidx.compose.foundation.layout.FlowRow
5   import androidx.compose.foundation.layout.fillMaxSize
6   import androidx.compose.foundation.layout.fillMaxWidth
7   import androidx.compose.foundation.layout.padding
8   import androidx.compose.foundation.rememberScrollState
9   import androidx.compose.foundation.verticalScroll
10  import androidx.compose.material3.Scaffold
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.Modifier
16  import androidx.navigation.NavController
17  import
    com.android.modul5.presentation.components.MovieCard
18  import com.android.modul5.presentation.components.
    TopBar
19  import com.android.modul5.presentation.viewmodel.
    MovieViewModel
20  import com.android.modul5.presentation.components.
    DarkModeSwitch
```

```
21
22  @OptIn(ExperimentalLayoutApi::class)
23  @Composable
24  fun MovieListScreen(movieViewModel: MovieViewModel,
    navController: NavController, isDarkMode: Boolean,
    onToggle: (Boolean) -> Unit) {
25      val Movies by movieViewModel.Movies.collectAsState()
26      val errMsg by
    movieViewModel.errorMsg.collectAsState()
27
28      Scaffold(
29          topBar = { TopBar("SIMOVIE") }
30      ) { innerPadding ->
31          Column(
32              modifier = Modifier
33                  .fillMaxSize()
34                  .padding(innerPadding)
35                  .verticalScroll(rememberScrollState()),
36          ) {
37              DarkModeSwitch(isDarkMode = isDarkMode,
    onToggle = onToggle)
38
39              errMsg?.let {msg -> Text(msg) }
40              FlowRow(
41                  modifier = Modifier
42                      .fillMaxWidth(),
43                  horizontalArrangement =
    Arrangement.Center
44              ) {
45                  Movies.forEach { movie ->
46                      MovieCard(movieItem = movie,
    navController)
47                  }
48              }
49          }
50      }
51  }
```

- **presentation/components/TopBar.kt**

*Tabel 20. Source Code TopBar Soal 1 Modul 5*

```
1   package com.android.modul5.presentation.components
2
3   import
    androidx.compose.material3.ExperimentalMaterial3Api
4   import androidx.compose.material3.MaterialTheme
5   import androidx.compose.material3.Text
6   import androidx.compose.material3.TopAppBar
```

```
7  import androidx.compose.material3.TopAppBarDefaults
8  import androidx.compose.runtime.Composable
9  import androidx.compose.ui.text.font.FontWeight
10
11 @OptIn(ExperimentalMaterial3Api::class)
12 @Composable
13 fun TopBar(Title: String) {
14     TopAppBar(
15         title = {
16             Text(
17                 Title,
18                 fontWeight = FontWeight.Bold,
19             ) },
20         colors = TopAppBarDefaults.topAppBarColors(
21             containerColor =
   MaterialTheme.colorScheme.primary,
22             titleContentColor =
   MaterialTheme.colorScheme.tertiary
23         )
24     )
25 }
```

- **presentation/components/Text.kt**

*Tabel 21. Source Code Text Soal 1 Modul 5*

```
1  package com.android.modul5.presentation.components
2
3  import androidx.compose.foundation.layout.Row
4  import androidx.compose.foundation.layout.fillMaxWidth
5  import androidx.compose.material3.Text
6  import androidx.compose.runtime.Composable
7  import androidx.compose.ui.Modifier
8  import androidx.compose.ui.text.font.FontWeight
9  import androidx.compose.ui.text.style.TextAlign
10
11 @Composable
12 fun Desc(title: String, content: Any?) {
13     Row(modifier = Modifier.fillMaxWidth()) {
14         Text(
15             text = "$title = ",
16             fontWeight = FontWeight.W600,
17             textAlign = TextAlign.Start
18             )
19         Text(content.toString())
20     }
21 }
```

- **presentation/components/movieCard.kt**

*Tabel 22. Source Code movieCard Soal 1 Modul 5*

```
1  package com.android.modul5.presentation.components
2
3  import androidx.compose.foundation.layout.Column
4  import androidx.compose.foundation.layout.Spacer
5  import androidx.compose.foundation.layout.fillMaxWidth
6  import androidx.compose.foundation.layout.height
7  import androidx.compose.foundation.layout.padding
8  import androidx.compose.foundation.layout.width
9  import androidx.compose.material3.Card
10 import androidx.compose.material3.CardDefaults
11 import androidx.compose.material3.MaterialTheme
12 import androidx.compose.material3.Text
13 import androidx.compose.runtime.Composable
14 import androidx.compose.ui.Modifier
15 import androidx.compose.ui.text.font.FontWeight
16 import androidx.compose.ui.text.style.TextAlign
17 import androidx.compose.ui.unit.dp
18 import androidx.compose.ui.unit.sp
19 import androidx.navigation.NavController
20 import com.android.modul5.domain.model.Movie
21
22 @Composable
23 fun MovieCard(movieItem: Movie, navController:
   NavController) {
24     Card(
25        modifier = Modifier
26            .width(180.dp)
27            .padding(8.dp),
28         shape = MaterialTheme.shapes.small,
29         elevation =
   CardDefaults.cardElevation(defaultElevation = 10.dp)
30         ) {
31         Glide(movieItem.posterPath)
32         Column(modifier =Modifier.padding(horizontal =
   8.dp)) {
33             Text(text = movieItem.title, fontWeight =
   FontWeight.W600, lineHeight = 17.sp, modifier =
   Modifier.fillMaxWidth(), textAlign = TextAlign.Center)
34             Spacer(modifier = Modifier.height(15.dp))
35             Text(text = "Asal Negara:
   ${movieItem.originalLanguage}", fontSize = 14.sp)
36             Text(text = "Tanggal Rilis:
   ${movieItem.releaseDate}", fontSize = 14.sp)
37         }
38         Spacer(modifier = Modifier.height(15.dp))
39         ButtonNav("Detail", navController, movieItem.id)
```

```
40          Spacer(modifier = Modifier.height(10.dp))
41
42      }
43  }
```

- **presentation/components/Glide.kt**

*Tabel 23. Source Code Glide Soal 1 Modul 5*

```
1   package com.android.modul5.presentation.components
2
3   import androidx.compose.foundation.layout.Column
4   import androidx.compose.foundation.layout.fillMaxWidth
5   import androidx.compose.runtime.Composable
6   import androidx.compose.ui.Alignment
7   import androidx.compose.ui.Modifier
8   import
    com.bumptech.glide.integration.compose.ExperimentalGlideC
    omposeApi
9   import com.bumptech.glide.integration.compose.GlideImage
10
11  @OptIn(ExperimentalGlideComposeApi::class)
12  @Composable
13  fun Glide(url: String?) {
14      Column(modifier = Modifier.fillMaxWidth(),
    horizontalAlignment = Alignment.CenterHorizontally) {
15          GlideImage(
16              model =
    "https://image.tmdb.org/t/p/w780${url}",
17              contentDescription = "movie img",
18          )
19      }
20
21  }
```

- **presentation/components/DarkModeSwitchkt**

*Tabel 24. Source Code DarkModeSwitch Soal 1 Modul 5*

```
1   package com.android.modul5.presentation.components
2
3
4   import androidx.compose.foundation.layout.Arrangement
5   import androidx.compose.foundation.layout.Row
6   import androidx.compose.foundation.layout.Spacer
7   import androidx.compose.foundation.layout.fillMaxSize
8   import androidx.compose.foundation.layout.size
9   import androidx.compose.foundation.layout.width
```

```
10 import androidx.compose.material3.Icon
11 import androidx.compose.material3.Switch
12 import androidx.compose.runtime.Composable
13 import androidx.compose.ui.Alignment
14 import androidx.compose.ui.Modifier
15 import androidx.compose.ui.res.painterResource
16 import androidx.compose.ui.unit.dp
17 import com.android.modul5.R
18
19 @Composable
20 fun DarkModeSwitch(isDarkMode: Boolean, onToggle:
   (Boolean)->Unit) {
21     Row(
22         modifier = Modifier.fillMaxSize(),
23         verticalAlignment = Alignment.CenterVertically,
24         horizontalArrangement = Arrangement.Center,
25
26     ) {
27         Icon(
28             painter = painterResource(id =
   R.drawable.sun_solid),
29             contentDescription = "Icon Sun" ,
30             modifier = Modifier.size(20.dp)
31
32         )
33         Spacer(modifier = Modifier.width(5.dp))
34         Switch(
35             checked = isDarkMode,
36             onCheckedChange = onToggle
37         )
38         Spacer(modifier = Modifier.width(5.dp))
39         Icon(
40             painter = painterResource(id =
   R.drawable.moon_solid),
41             contentDescription = "Icon moon",
42             modifier = Modifier.size(20.dp)
43         )
44     }
45
46 }
```

- **presentation/components/ButtonNav.kt**

*Tabel 25. Source Code ButtonNav Soal 1 Modul 5*

```
1 package com.android.modul5.presentation.components
2
3 import androidx.compose.foundation.layout.Column
4 import androidx.compose.foundation.layout.fillMaxSize
```

```
5  import androidx.compose.foundation.layout.height
6  import androidx.compose.foundation.layout.width
7  import androidx.compose.material3.Button
8  import androidx.compose.material3.MaterialTheme
9  import androidx.compose.material3.Text
10 import androidx.compose.runtime.Composable
11 import androidx.compose.ui.Alignment
12 import androidx.compose.ui.Modifier
13 import androidx.compose.ui.unit.dp
14 import androidx.navigation.NavController
15
16 @Composable
17 fun ButtonNav(action: String, navController:
   NavController, movieId: Int = 0) {
18     Column(modifier = Modifier.fillMaxSize(),
   horizontalAlignment = Alignment.CenterHorizontally) {
19         Button(
20             onClick = {
21                 if (action == "Detail")
   {navController.navigate("movie_detail/${movieId}")}
22                 else {navController.popBackStack()}
23             },
24             shape = MaterialTheme.shapes.small,
25             modifier =
   Modifier.width(100.dp).height(35.dp)
26         ) {
27             Text(action, color =
   MaterialTheme.colorScheme.tertiary)
28         }
29     }
30
31 }
```

- **domain/model/Movie.kt**

*Tabel 26. Source Code Movie Soal 1 Modul 5*

```
1  package com.android.modul5.domain.model
2
3  import kotlinx.serialization.SerialName
4  import kotlinx.serialization.Serializable
5
6  @Serializable
7  data class MovieResponse(
8      val page: Int,
9      val results: List<Movie>,
10     val total_pages: Int,
11     val total_results: Int
12 )
```

```
13
14 @Serializable
15 data class Movie(
16     val id: Int,
17     val popularity: Double,
18     @SerialName("original_language")
19     val originalLanguage: String,
20     val overview: String,
21     @SerialName("poster_path")
22     val posterPath: String?,
23     @SerialName("release_date")
24     val releaseDate: String,
25     val title: String,
26     @SerialName("vote_average")
27     val voteAverage: Double,
28 )
```

- **data/api/TMDBAPI.kt**

*Tabel 27. Source Code TMDBAPI Soal 1 Modul 5*

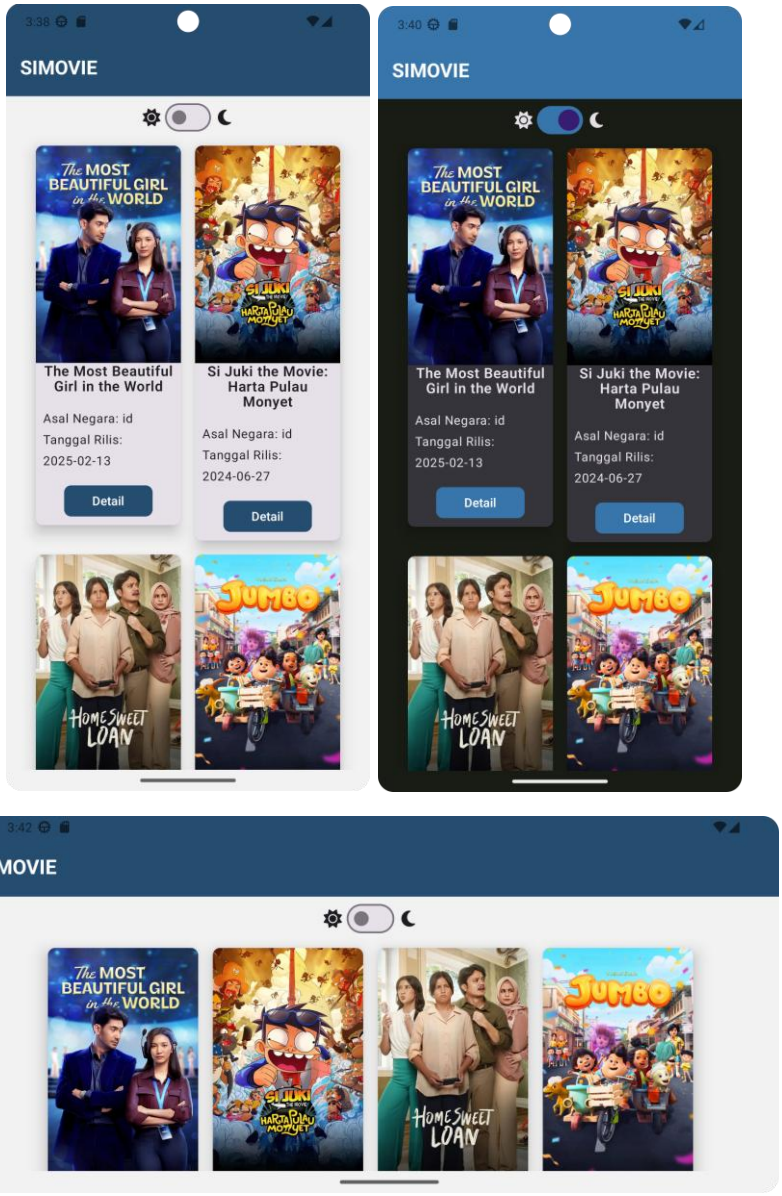
```
1  package com.android.modul5.data.api
2
3  import com.android.modul5.domain.model.Movie
4  import com.android.modul5.domain.model.MovieResponse
5  import retrofit2.Response
6  import retrofit2.http.GET
7  import retrofit2.http.Path
8  import retrofit2.http.Query
9
10 interface TMDBAPI {
11     @GET("discover/movie")
12     suspend fun getPopularMovies(
13         @Query("api_key") apikey: String,
14         @Query("with_original_language") lang: String =
   "id",
15         @Query("with_genres") genre: Int = 35,
16         @Query("page") page: Int = 1,
17     ) : Response<MovieResponse>
18
19     @GET("movie/{movie_id}")
20     suspend fun getMovieDetailbyID(
21         @Path("movie_id") movieId: Int,
22         @Query("api_key") apikey: String,
23     ) : Response<Movie>
24 }
```

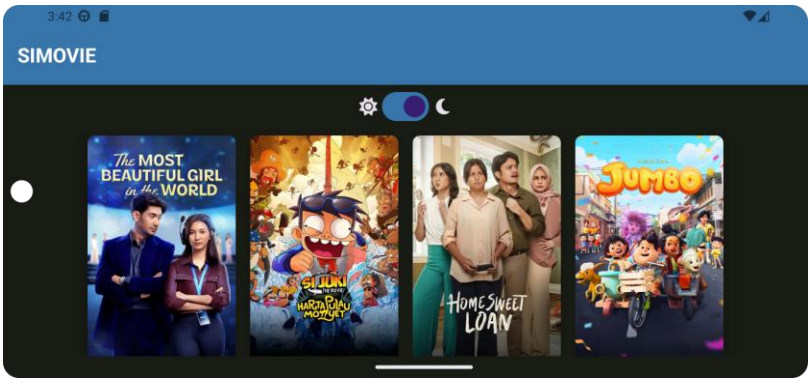
- **data/api/RetrofitClient.kt**

*Tabel 28. Source Code RetrofitClient Soal 1 Modul 5*

```
1  package com.android.modul5.data.api
2
3  import kotlinx.serialization.json.Json
4  import okhttp3.MediaType.Companion.toMediaType
5  import retrofit2.Retrofit
6  import
   retrofit2.converter.kotlinx.serialization.asConverterFact
   ory
7
8  object RetrofitClient {
9      private val json = Json {
10         ignoreUnknownKeys = true
11     }
12     private const val BASE_URL =
   "https://api.themoviedb.org/3/"
13
14     val tmdbAPI: TMDBAPI by lazy {
15         Retrofit.Builder()
16             .baseUrl(BASE_URL)
17
   .addConverterFactory(json.asConverterFactory("application
   /json"
   .toMediaType()))
18             .build()
19             .create(TMDBAPI::class.java)
20     }
21 }
```

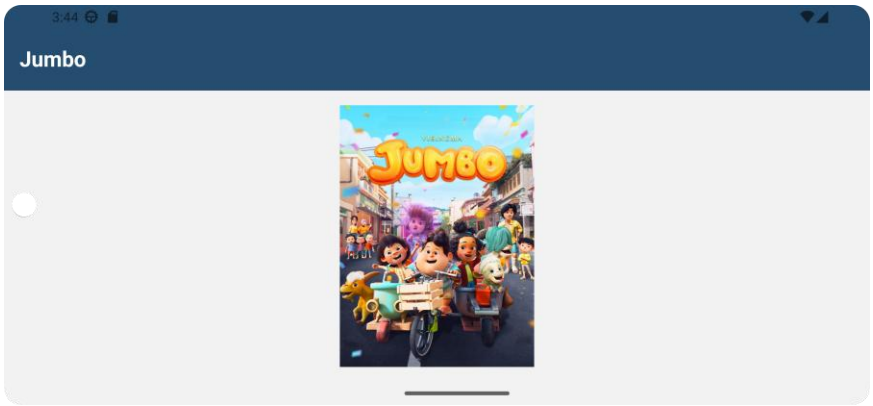
## B. Output Program

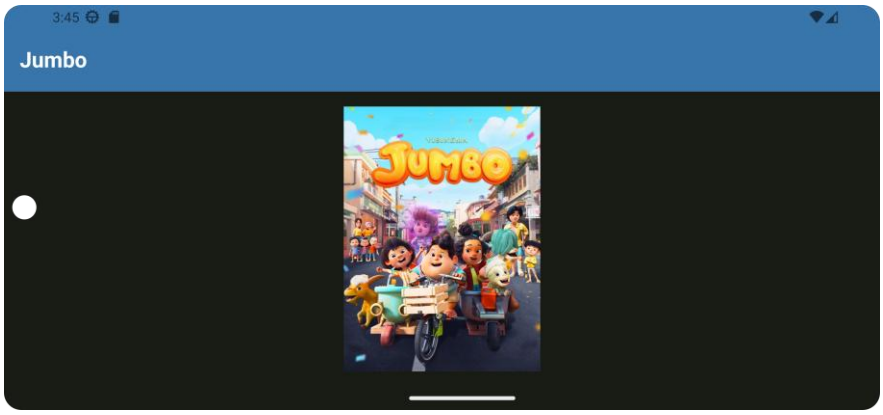



3:42
SIMOVIE
The MOST BEAUTIFUL GIRL in the WORLD
HOME SWEET LOAN
JUMBO


3:43
Jumbo
JUMBO
Tanggal Rilis = 2025-03-31
Popularitas = 2.8584
Rata-rata Vote = 6.7
Desperate to prove to his peers that he's more than a chubby kid that never wins at anything, little Don wishes to win his local talent show by performing a stageplay inspired from a storybook written by his late parents. When unfortunately a bully stole that book, at the same time a little ghost appeared from the spirit world asking for Don's help to reunite her with her parents' spirits. The two new friends then embark on an exciting adventure to help each other and learn the true meaning of friendship.
Kembali


3:44
Jumbo
JUMBO
Tanggal Rilis = 2025-03-31
Popularitas = 2.8584
Rata-rata Vote = 6.7
Desperate to prove to his peers that he's more than a chubby kid that never wins at anything, little Don wishes to win his local talent show by performing a stageplay inspired from a storybook written by his late parents. When unfortunately a bully stole that book, at the same time a little ghost appeared from the spirit world asking for Don's help to reunite her with her parents' spirits. The two new friends then embark on an exciting adventure to help each other and learn the true meaning of friendship.
Kembali


3:44
Jumbo
JUMBO




*Gambar 16. Screenshot Hasil Jawaban Soal 1 Modul 5*

## C. Pembahasan

- **MainActivity.kt:**
  - Pada baris 1, **package com.android.modul5** pendeklarasian nama package file Kotlin.
  - Pada baris 3-27, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
  - Pada baris 29, **class MainActivity : ComponentActivity(),** merupakan titik mula yang menjadi kelas utama dan akan dijalankan pertama kali saat aplikasi dibuka
  - Pada baris 30, **override fun onCreate(savedInstanceState: Bundle?),** berfungsi untuk menimpa (override) fungsi oncreate dari ComponentActivity.
  - Pada baris 31, **super.onCreate(savedInstanceState),** berfungsi untuk memanggil superclass dari fungsi oncreate untuk memastikan bahwa proses inisialisasi standar dari Android dijalankan sebelum logika saya dijalankan.
  - Pada baris 32, **enableEdgeToEdge(),** berfungsi agar tampilan aplikasi dapat menggunakan seluruh layar dari status bar sampai navigation bar atau fullscreen layout.

- Pada baris 26, **setContent(),** digunakan untuk menampilkan UI berbasis jetpack compose ke dalam activity
- Pada baris 34, **val context = LocalContext.current** berfungsi untuk mendapatkan context dari komposisi saat ini
- Pada baris 35, **val prefs = remember {context.getSharedPreferences("app_prefs", Context.MODE_PRIVATE)}** berfungsi untuk mendapatkan instance SharedPreference dengan nama app_prefs dan diingat agar tidak di dibuat ulang setiap kali recompose.
- Pada baris 36, **var isDarkMode by remember { mutableStateOf(prefs.getBoolean("is_dark_mode", false)) }** berfungsi untuk mendeklarasikan state dengan nama isDarkMode yang nilai awalnya diambil dari SharedPrefrence.
- Pada baris 38, **Modul5Theme(darkTheme = isDarkMode),** merupakan fungsi yang berisi tema custom yang membungkus seluruh UI untuk memberikan style yang konsisten.
- Pada baris 39, **NavMovies(isDarkMode = isDarkMode, onToggle = {newValue -> isDarkMode = newValue prefs.edit {putBoolean("is_dark_mode", newValue)}})** merupakan fungsi composable untuk mengelola navigasi antar halaman, yang menerima state isDarkMode, lambda onToggle
- Pada baris 50, **@composable,** merupakan annotation untuk menandai bahwa fungsi yang dibuat dibawahnya merupakan fungsi UI di jetpack compose.
- Pada baris 51, **fun NavMovies(isDarkMode: Boolean, onToggle: (Boolean) -> Unit)** merupakan fungsi composable yang dibuat untuk menampilkan semua composable yang ingin ditampilkan ke dalam activity, seperti MovieListScreen() dan MovieDetailScreen() lalu menerima parameter isDarkMode dan onToggle.

- Pada baris 52, **val navController = rememberNavController()** berfungsi untuk membuat dan menyimpan instance dari NavController
- Pada baris 53, **val movieViewModel: MovieViewModel = viewModel()** berfungsi untuk mendapatkan MovieViewModel yang akan digunakan untuk halaman-halaman di dalam aplikasi.
- Pada baris 55, **NavHost(),** berfungsi untuk menyediakan wadah untuk navigasi antar composable screen berdasarkan rute yang ditentukan
    - Pada baris 56, **navController** merupakan controller yang bertanggung jawab atas navigasi.
    - Pada baris 57, **startDestination** merupakan rute awal yang ditampilkan ketika aplikasi dijalankan.
    - Pada baris 58, **enterTransition** berfungsi untuk mendefinisikan animasi pada saat memasuki layar
    - Pada baris 61, **exitTransition** berfungsi untuk mendefinisikan animasi pada saat keluar dari layar
    - Pada baris 64, **popEnterTransition** berfungsi untuk mendefinisikan animasi pada saat memasuki Kembali layar setelah operasi pop
    - Pada baris 67, **popExitTransition** berfungsi untuk mendefinisikan animasi pada saat keluar dari layar setelah operasi pop
- Pada baris 71 dan 74, **composable(),** berfungsi untuk mendefinisikan rute, seperti di kasus ini ada "movie_list" dan "movie_detail"
    - Pada baris 75, **route** bergunsi untuk rute dari halaman yang dituju dan akan menerima data movieId melalui url
    - Pada baris 76, **arguments** berfungsi untuk mendefinisikan daftar argument untuk diteruskan
- Pada baris 72, **MovieList(navController, ViewModel),** merupakan fungsi untuk menampilkan list yang dapat scroll sekaligus menjadi halaman utama aplikasi dan menerima parameter navController, movieViewModel, isDarkMode, dan onToggle.

- pada baris 51, **MovieDetailScreen(movieId, movieViewModel, navController)** merupakan fungsi untuk menampilkan halaman detail dari aplikasi dan menerima beberapa parameter.

- **presentation/screens/MovieListScreen.kt**
  - Pada baris 1-20, **import** berfungsi untuk mengimport data class dari file CardProp.
  - Pada baris 22, **@OptIn(ExperimentalLayoutApi::class)** berfungsi untuk memberitahukan compiler bahwa fungsi composable dibawahnya masih menggunakan API yang bersifat eksperimental untuk menghindari error saat kompilasi.
  - Pada baris 23, **@Composable** merupakan annotation untuk menandai bahwa fungsi yang dibuat dibawahnya merupakan fungsi UI di jetpack compose.
  - Pada baris 24, **fun MovieListScreen(movieViewModel: MovieViewModel, navController: NavController, isDarkMode: Boolean, onToggle: (Boolean) -> Unit)** merupakan sebuah fungsi untuk menampilkan daftar film-film dari API TMDB dalam bentuk card yang menerima parameter movieViewModel, navController, isDarkMode, dan onToggle.
  - Pada baris 25, **val Movies by movieViewModel.Movies.CollectAsState()** berfungsi untuk mengumpulkan aliran atau flow dari MovieViewModel.Movies menjadi state untuk memperbarui UI jika data film mengalami perubahan
  - Pada baris 26, **val errMsg by movieViewModel.errorMsg.CollectAsState()** berfungsi untuk mengumpulkan aliran atau flow dari MovieViewModel.errorMsg menjadi state untuk memperbarui UI jika ada error yang terjadi

- pada baris 28, **scaffold** merupakan komponen layout yang menjadi kerangka dasar dari tampilan yang mempermudah dalam mengatur strktur UI karena sudah disediakan slot-slot bawannya seperti topBar.
- pada baris 29, **topBar** merupakan tempat Dimana topBar akan diletakkan di scaffold
- pada baris 31, **Column** merupakan komponen layout mirip seperti scaffold, hanya saja Column digunakan untuk menyusun elemen UI secara vertical.
    - pada baris 32, **modifier** berfungsi untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.padding* untuk memberikan jarak dalam component, *.fillmaxsize* membuat column memenuhi layar, dan *.verticalscroll* membuat konten dapat di scroll
- pada baris 37, **DarkModeSwitch(isDarkMode = isDarkMode, onToggle = onToggle)**
- pada baris 31, **errMsg?.let {msg -> Text(msg) }** akan menampilkan pesan error jika errMsg tidak bernilai null
- pada baris 39, **FlowRow** merupakan komponen layout yang digunakan untuk menata elemen-elemen didalamnya secara horizontal seperti Row, tapi bedanya jika tidak muat akan membuat baris baru dibawahnya
    a. pada baris 40, **modifier** berfungsi untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.padding* untuk memberikan jarak dalam dan *,fillmaxwidth* untuk memenuhi lebar layar
    b. pada baris 41, **horizontalArrangement** berfungsi untuk mengattur jarak elemen agar tpat berada ditengah
- pada baris 45, **Movies.forEach { movie -> MovieCard(movieItem = movie, navController)}** berfungsi untuk melakukan perulangan sebanyak objek Movies yang tersedia dan akan memanggil composable MoviesCard lalu meneruskan objek film dan NavControl

- **presentation/screens/MovieDetailScreen.kt**
  - Pada baris 1, **package com.android.modul5.presentation.screens** pendeklarasian nama package file Kotlin.
  - Pada baris 3, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
  - Pada baris 39, **@Composable** merupakan annotation untuk menandai bahwa fungsi yang dibuat dibawahnya merupakan fungsi UI di jetpack compose.
  - Pada baris 40, **fun MovieDetailScreen(movieId: Int, movieViewModel: MovieViewModel, navController: NavController)**
  - Pada baris 41, **val Movies by movieViewModel.Movies.CollectAsState()** berfungsi untuk mengumpulkan aliran atau flow dari MovieViewModel.Movies menjadi state untuk memperbarui UI jika data film mengalami perubahan
  - Pada baris 42, **val errMsg by movieViewModel.errorMsg.CollectAsState()** berfungsi untuk mengumpulkan aliran atau flow dari MovieViewModel.errorMsg menjadi state untuk memperbarui UI jika ada error yang terjadi
  - Pada baris 44, **LaunchedEffect(movieId) {movieViewModel.fetchMoviebyDetailbyID(movieId)}** merupakan sebuah composable effect untuk memicukode didalamnya saat ada perubahan pada movieId saat composable pertama kali masuk komposisi
  - pada baris 48, **scaffold** merupakan komponen layout yang menjadi kerangka dasar dari tampilan yang mempermudah dalam mengatur strktur UI karena sudah disediakan slot-slot bawannya seperti topBar.
  - pada baris 49, **topBar** merupakan tempat Dimana topBar akan diletakkan di scaffold

- pada baris 51, **Column** merupakan komponen layout mirip seperti scaffold, hanya saja Column digunakan untuk menyusun elemen UI secara vertical.
    - pada baris 32, **modifier** berfungsi untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.padding* untuk memberikan jarak dalam component, *.fillmaxsize* membuat column memenuhi layar *.verticalscroll* membuat konten dapat di scroll
    a. pada baris 57, **horizontalAlignment** berfungsi untuk menentukan posisi semua elemen di dalam Column secara horizontal.
- pada baris 59, **errMsg?.let {msg -> Text(msg) }** akan menampilkan pesan error jika errMsg tidak bernilai null
- pada baris 60, **Box** merupakan sebuah composable untuk menumpuk elemen
    - pada baris 61, **modifier** berfungsi untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.height* untuk mengatur tinggi dan *.width* untuk mengatur lebar.
- pada baris 60, **Glide()** merupakan fungsi yang saya buat sendiri untuk memudahkan dalam memuat gambar.
- Pada baris 67-73, **Spacer()** berfungsi untuk memberikan jarak kosong biasanya secara vertical dalam layout.
- Pada baris 68-70, **Desc()** merupakan fungsi yang saya buat sendiri untuk memudahkan dalam membuat deskripsi.
- Pada baris 72, **Text()** merupakan komponen UI yang berfunsi menampilkan teks.
    a. Pada baris 72, **textAlign** untuk membuat apakah teks berada di kiri, Tengah , atau kanan
- Pada baris 74, **ButtonNav()** merupakan fungsi yang saya buat sendiri untuk menampilkan tombol ke halaman detail

- **Presentation/components/TopBar.kt**
  - Pada baris 1, **package com.android.modul5.presentation.components** pendeklarasian nama package file Kotlin.
  - Pada baris 3-9, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
  - Pada baris 11, **@OptIn(ExperimentalLayoutApi::class)** berfungsi untuk memberitahukan compiler bahwa fungsi composable dibawahnya masih menggunakan API yang bersifat eksperimental untuk menghindari error saat kompilasi.
  - Pada baris 12, **@Composable** merupakan annotation untuk menandai bahwa fungsi yang dibuat dibawahnya merupakan fungsi UI di jetpack compose.
  - Pada baris 13, **fun TopBar(Title: String)** merupakan fungsi untuk membuat component yang nantinya akan dipanggil untuk menampilkan topbar di atas layar yang menerima parameter String.
  - Pada baris 14, **TopAppBar**
    - **a.** Pada baris 15, **title** merupakan parameter dalam TopAppBar() untuk menampilkan judul .
    - **b.** Pada baris 16, **Text()** merupakan komponen UI yang berfunsi menampilkan teks.
      - Pada baris 17, **Title** merupakan data dari parameter yang akan ditampilkan sebagai judul topbar.
      - Pada baris 18, **fontWeight** merupakan ketebalan dari teks yang akan ditampilkan
    - **c.** Pada baris 20, **colors** merupakan parameter dalam TopAppBar() untuk mengatur warna background (containerColor) dan warna teks judul (titleContentColor).

- **Presentation/components/Text.kt**
  - Pada baris 1, **package com.android.modul5.components** pendeklarasian nama package file Kotlin.
  - Pada baris 3-9, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
  - Pada baris 11, **@Composable** merupakan annotation untuk menandai bahwa fungsi yang dibuat dibawahnya merupakan fungsi UI di jetpack compose.
  - Pada baris 35, ***fun*** **Desc(title: String, content: Any?)**
  - Pada baris 13, **Row** merupakan komponen layout yang berlawanan dari Column, Dimana jika Column menyusun elemen UI secara Vertical, maka Row menyusun elemen UI secara horizontal.
    a. Pada baris 13, **modifier** berfungsi untuk menerapkan modifikasi terhadap tampilan komponen, *.fillMaxWidth()* untuk membuat lebar component lebarnya selayar.
  - Pada baris 14 dan 19, **Text()** merupakan komponen UI yang berfunsi menampilkan teks.
    a. Pada baris 15, **text** merupakan tempat string yang akan ditampilkan diletakkan
    b. Pada baris 16, **fontWeight** merupakan ketebalan dari teks yang akan ditampilkan
    c. Pada baris 17, **textAlign** untuk membuat apakah teks berada di kiri, Tengah , atau kanan

- **Presentation/components/movieCard.kt**
  - Pada baris 1, **package com.android.modul5.presentation.components** pendeklarasian nama package file Kotlin.

- Pada baris 3-20, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
- Pada baris 22, **@Composable** merupakan annotation untuk menandai bahwa fungsi yang dibuat dibawahnya merupakan fungsi UI di jetpack compose.
- Pada baris 23, **fun MovieCard(movieItem: Movie, navController: NavController)** berfungsi untuk menampilkan Movie yang didapatkan dalam bentuk Card, lalu menerima parameter movieItem dan navController
- Pada baris 24, **Card** berfungsi untuk membuat sebuah kartu.
    a. Pada baris 25, **modifier m**erupakan parameter dalam Column() untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.padding* untuk memberikan jarak dalam component, *.width* untuk lebar component
    b. Pada baris 26, **shape** merupakan parameter dalam Button() untuk mengatur sudut agar dapat membulat.
    c. Pada baris 27, **elevation** meruapakan parameter untuk mengatur Tingkat kedalamn bayang dari elemen
- Pada baris 31, **Glide** merupakan fungsi yang saya buat sendiri untuk memudahkan dalam memuat gambar.
- Pada baris 32, **Column,** merupakan komponen layout mirip seperti scaffold, hanya saja Column digunakan untuk menyusun elemen UI secara vertical.
    - Pada baris 33, **modifier** berfungsi untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.padding* untuk memberikan jarak dalam
- Pada baris 33-36, **Text** merupakan komponen UI yang berfunsi menampilkan teks.

a. Pada baris 33-36, **text** merupakan tempat string yang akan ditampilkan diletakkan
b. Pada baris 33, **fontWeight** merupakan ketebalan dari teks yang akan ditampilkan
c. Pada baris 33, **lineHeight** merupakan tinggi dari baris tempat teks berada
d. Pada baris 35 dan 36, **fontSize** merupakan ukuran dari font yang akan ditampilkan

- Pada baris 34-40, **Spacer** berfungsi untuk memberikan jarak kosong biasanya secara vertical dalam layout.
- Pada baris 39, **ButtonNav** merupakan fungsi yang saya buat sendiri untuk menampilkan tombol Kembali ke halaman utama

- **Presentation/components/Glide.kt**
  - Pada baris 1, **package com.android.modul5.presentation.components** pendeklarasian nama package file Kotlin.
  - Pada baris 3-9, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
  - Pada baris 11, **@OptIn(ExperimentalGlideComposeApi::*class*)** berfungsi untuk memberitahukan compiler bahwa fungsi composable dibawahnya masih menggunakan API yang bersifat eksperimental untuk menghindari error saat kompilasi.
  - Pada baris 12, **@Composable** merupakan annotation untuk menandai bahwa fungsi yang dibuat dibawahnya merupakan fungsi UI di jetpack compose.
  - Pada baris 7, **fun Glide(url: String?)** berfungsi untuk menampilkan gambar menggunakan Glide

- pada baris 14, **Column** merupakan komponen layout mirip seperti scaffold, hanya saja Column digunakan untuk menyusun elemen UI secara vertical.
    - pada baris 14, **modifier** berfungsi untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.fillmaxwidth* unruk memenuhi sesuai lebar layar
    a. pada baris 14, **horizontalAlignment** berfungsi untuk menentukan posisi semua elemen di dalam Column secara horizontal.
- Pada baris 15, **GlideImage** merupakan library glide yang digunakan untuk menampilakn gambar dar internet melalui URL
    a. Pada baris 16, **model** merupakan tempat Dimana URL diletakkan
    b. Pada baris 17, **contentDescription** merupakan deskripsi dari gambar yang akan ditampilkan

- **Presentation/components/DarkModeSwitch.kt**
    - Pada baris 1, **package com.android.modul5.presentation.components** pendeklarasian nama package file Kotlin.
    - Pada baris 4-17, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
    - Pada baris 19, **@Composable** merupakan annotation untuk menandai bahwa fungsi yang dibuat dibawahnya merupakan fungsi UI di jetpack compose.
    - Pada baris 20, **fun DarkModeSwitch(isDarkMode: Boolean, onToggle: (Boolean)->Unit)** merupakan fungsi untuk membuat switch untuk keperluan darkmode yang menrima parameter isDarkMode dan onToggle
    - Pada baris 21, **Row** merupakan komponen layout yang berlawanan dari Column, Dimana jika Column menyusun elemen UI secara Vertical, maka Row menyusun elemen UI secara horizontal.

a. Pada baris 22, **modifier** merupakan parameter untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.fillMaxWidth* agar column lebar Row mengisi ukuran layar.
b. Pada baris 23, **verticalAlignment** berfungsi untuk posisi semua elemen di dalam row secara vertical
c. Pada baris 24, **horizontalArrangement** berfungsi untuk mengattur jarak elemen agar tpat berada ditengah

- Pada baris 27 dan 39, **Icon** merupakan sebuah fungsi untuk menampilkan icon
    a. Pada baris 28 dan 40, **painter** berfungsi sebagai tempat memuat icon dari drawable
    b. Pada baris 29 dan 41, **contentDescription** berfungsi untuk mendeksripsikan icon apa yang ditampilkan
    c. Pada baris 30 dan 42, **modifier** merupakan parameter untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.size* untuk menentukan ukuran
- Pada baris 33 dan 38, **Spacer** berfungsi untuk memberikan jarak kosong biasanya secara vertical dalam layout.
- Pada baris 34, **Switch** merupakan komponen berbentuk switch yang digunakan untuk menyalakan atau mematikan fitur
    - Pada baris 35, **checked** berfungsi untuk menentukan status switch saat ini apakah true atau false.
    - Pada baris 36, **onCheckedChange** merupakan fungsi yang dipanggil saat user mengubah posisi switch.

- **Presentation/components/ButtonNav.kt**
    - Pada baris 1, **package com.android.modul5.presentation.components** pendeklarasian nama package file Kotlin.

- Pada baris 3-14, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
- Pada baris 16, **@Composable** merupakan annotation untuk menandai bahwa fungsi yang dibuat dibawahnya merupakan fungsi UI di jetpack compose.
- Pada baris 17, **fun ButtonNav(action: String, navController: NavController, movieId: Int = 0)** merupakan fungsi untuk membuat tombol untuk navigasi antar halaman yang menerima parameter action dan navController
- pada baris 18, **Column** merupakan komponen layout mirip seperti scaffold, hanya saja Column digunakan untuk menyusun elemen UI secara vertical.
    - pada baris 18, **modifier** berfungsi untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.fillmaxsize* unruk memenuhi seluruh layar
    a. pada baris 18, **horizontalAlignment** berfungsi untuk menentukan posisi semua elemen di dalam Column secara horizontal.
- Pada baris 19, **Button()** merupakan komponen UI yang berfungsi untuk membuat tombol pada aplikasi.
    a. Pada baris 20, **onClick** merpakan parameter dalam Button() untuk memberikan aksi yang akan dijalankan ketika tombol ditekan.
    b. Pada baris 24, **shape** merupakan parameter dalam Button() untuk mengatur sudut agar dapat membulat
    c. Pada baris 25, **modifier** merupakan parameter dalam TextField untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.width* dan *.height* agar column lebar dan tinggi Button mengisi ukuran layar.
- Pada baris 27, **Text** merupakan komponen UI yang berfunsi menampilkan teks.

- **domain/model/Movie.kt**
    - Pada baris 1, **package com.android.modul5.domain.model** pendeklarasian nama package file Kotlin.
    - Pada baris 3 dan 4, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
    - Pada baris 6 dan 15, **data class** merupakan jenis kelas untuk menyimpan struktur data.
    - Pada baris 5 dan 14, **@Serializable** merupakan sebuah anotasi untuk mendakan bahwa class dibawahnya akan di serialisasi atau deserialisasi.
    - Pada baris 8-27, **val** merupakan keyword untuk membuat suatu variable bersifat immutable yang menampung nilai dari berbagai tipe data.
    - Pada baris 18-26, **@SerialName** merupakan anotasi yang berfungsi untuk menghubungkan nama property di Kotlin dengan nama key di JSON

- **data/api/TMDBAPI.kt**
    - Pada baris 1, **package com.android.modul5.data.api** pendeklarasian nama package file Kotlin.
    - Pada baris 3-8, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
    - Pada baris 10, **interface** merupakan antarmukan yang akan digunakan oleh retrofit untuk membuat implementasi API TMDB
    - Pada baris 11 dan 19, **@GET** berfungsi untuk menentukan endpoint dan method HTTP untuk mencari film
    - Pada baris 12 dan 20, **suspend fun** merupakan fungsi suspend untuk melakukan panggilan API
    - Pada baris 13-22, **@Query** merupakan parameter untuk pencarian film sesuai yang dinginkan

- Pada baris 21, **@Path** merupakan parameter untuk menggantikan placeholder di URL

- **data/api/RetrofitClient.kt**
  - Pada baris 1, **package com.android.modul5.data.api** pendeklarasian nama package file Kotlin.
  - Pada baris 3-6, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
  - Pada baris 6, **object RetrofitClient** merupakan sebuah objek singleton yang hanya akan aka nada satu instansi dari Retrofitclient di seluruh aplikasi.
  - Pada baris 9 dan 12, **private const val** merupakan keyword untuk membuat suatu variable bersifat immutable yang menampung nilai dari berbagai tipe data dan bersifat privat.
  - Pada baris 9, **lazy** berfungsi agar object tidak akan dibuat segera saat inisiasi, tapi saat property diakses untuk pertama kali
  - Pada baris 10, **Retrofit.Builder()** digunakan untuk memulai prosesbuilding dari instance retrofit
  - Pada baris 10, **.baseUrl** merupakan url dasar yang digunakan untuk semua permintaan API
  - Pada baris 11, **.addConverterFactory** merupakan sebuah converter yang bertugas untuk serialisasi dan deserialisasi data
  - Pada baris 12, **.build** berfungsi untuk menyelesaikan konfigurasi dan membangun instance retrofit
  - Pada baris 13, **.create** merupakan Langkah terakhir Dimana kita akan meminta retrofit untuk mengimplementasikan antarmuka TMDBAPI

# Tautan Git

Berikut adalah tautan untuk source code yang telah dibuat.

https://github.com/AppleCraft2005/kuliah-santuy/tree/main/semesterIV/Pemrograman-Mobile