# LAPORAN PRAKTIKUM
# PEMROGRAMAN MOBILE
# MODUL 3


**Build a Scrollable List**

**Oleh:**

**Muhammad Raka Azwar  NIM. 2210817210012**

**PROGRAM STUDI TEKNOLOGI INFORMASI**
**FAKULTAS TEKNIK**
**UNIVERSITAS LAMBUNG MANGKURAT**
**APRIL 2025**

LEMBAR PENGESAHAN

# LAPORAN PRAKTIKUM PEMROGRAMAN I

# MODUL 3

Laporan Praktikum Pemrograman Mobile Modul 3: Build a Scrollable List Android Basic with Kotlin ini disusun sebagai syarat lulus mata kuliah Praktikum Pemrograman Mobile. Laporan Prakitkum ini dikerjakan oleh:

Nama Praktikan : Raymond Hariyono
NIM : 2310817210007

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

# SOAL 1

**Soal Praktikum:**

1. Buatlah sebuah aplikasi Android menggunakan XML dan Jetpack Compose yang dapat menampilkan list dengan ketentuan berikut:
    1. List menggunakan fungsi RecyclerView (XML) dan 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 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

## A. Source Code

### Source Code XML

### item_film.XML

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.cardview.widget.CardView
android:layout_width="match_parent"
    android:id="@+id/card_view"
    android:layout_height="wrap_content"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_margin="8dp"
    app:cardCornerRadius="22dp"
    app:cardElevation="4dp"
    android:layout_gravity="center"
    xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:android="http://schemas.android.com/apk/res/android"
>

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:padding="8dp">

        <ImageView
            android:id="@+id/film_image"
            android:layout_width="150dp"
            android:layout_height="200dp"
            android:scaleType="centerCrop"

app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0.0"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
```

```
32              app:layout_constraintVertical_bias="0.0"
33              tools:src="@tools:sample/avatars" />
34
35          <TextView
36              android:id="@+id/film_title"
37              android:layout_width="120dp"
38              android:layout_height="wrap_content"
39              android:layout_marginStart="12dp"
40              android:textSize="13sp"
41              android:textStyle="bold"
42
43  app:layout_constraintBottom_toBottomOf="parent"
44              app:layout_constraintEnd_toEndOf="parent"
45              app:layout_constraintHorizontal_bias="0.0"
46
47  app:layout_constraintStart_toEndOf="@+id/film_image"
48              app:layout_constraintTop_toTopOf="parent"
49              app:layout_constraintVertical_bias="0.082"
50              tools:text="judul film" />
51
52          <TextView
53              android:id="@+id/film_year"
54              android:layout_width="wrap_content"
55              android:layout_height="wrap_content"
56              android:layout_marginEnd="20dp"
57              android:textSize="13sp"
58              android:textStyle="bold"
59
60  app:layout_constraintBottom_toBottomOf="parent"
61              app:layout_constraintEnd_toEndOf="parent"
62
```

```
63              app:layout_constraintTop_toTopOf="parent"
64              app:layout_constraintVertical_bias="0.083"
65              tools:ignore="MissingConstraints"
66              tools:text="year" />
67
68          <TextView
69              android:id="@+id/film_desc"
70              android:layout_marginStart="10dp"
71              android:layout_width="0dp"
72              android:layout_height="wrap_content"
73              android:textSize="10sp"
74
75  app:layout_constraintBottom_toTopOf="@+id/btn_imdb"
76              app:layout_constraintEnd_toEndOf="parent"
77              app:layout_constraintHorizontal_bias="0.266"
78
79  app:layout_constraintStart_toEndOf="@+id/film_image"
80
81  app:layout_constraintTop_toBottomOf="@+id/film_title"
82              app:layout_constraintVertical_bias="0.2"
83              tools:text="hab" />
84
85          <Button
86              android:id="@+id/btn_imdb"
87              android:layout_width="wrap_content"
88              android:layout_height="wrap_content"
89
90  app:layout_constraintBottom_toBottomOf="parent"
91
92  app:layout_constraintEnd_toStartOf="@+id/btn_detail"
93              app:layout_constraintHorizontal_bias="0.266"
```

```
94
95  app:layout_constraintStart_toEndOf="@+id/film_image"
96              app:layout_constraintTop_toTopOf="parent"
97              app:layout_constraintVertical_bias="0.947"
98              android:textSize="10sp"
99              android:text="@string/btn_imdb" />
10
0           <Button
10              android:id="@+id/btn_detail"
1               android:layout_width="wrap_content"
10              android:layout_height="wrap_content"
2
10  app:layout_constraintBottom_toBottomOf="parent"
3               app:layout_constraintEnd_toEndOf="parent"
10              app:layout_constraintHorizontal_bias="1.0"
4
10  app:layout_constraintStart_toEndOf="@+id/film_image"
5               app:layout_constraintTop_toTopOf="parent"
10              app:layout_constraintVertical_bias="0.947"
6               android:textSize="10sp"
10              android:text="@string/btn_detail" />
7
10      </androidx.constraintlayout.widget.ConstraintLayout>
8   </androidx.cardview.widget.CardView>
10
9
11
0
11
1
```

*Tabel 1. Source Code file item_film XML Jawaban Soal 1 XML*

**Fragment_detail.XML**

```
1  <?xml            version="1.0"            encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout
3  xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:app="http://schemas.android.com/apk/res-auto"
5      xmlns:tools="http://schemas.android.com/tools"
6      android:layout_width="match_parent"
7      android:layout_height="match_parent"
8      android:padding="8dp"
9      tools:context=".DetailFragment">
10
11     <android.widget.ScrollView
12         android:layout_width="match_parent"
13         android:layout_height="match_parent">
14
15         <LinearLayout
16             android:orientation="vertical"
17             android:layout_width="match_parent"
18             android:layout_height="wrap_content"
19             android:padding="8dp"
20             android:gravity="center">
21
22             <ImageView
23                 android:id="@+id/detail_image"
24                 android:layout_width="wrap_content"
25                 android:layout_height="400dp"
26                 android:scaleType="centerCrop"
27                 tools:src="@tools:sample/avatars"
28                 app:layout_constraintTop_toTopOf="parent"
29
30 app:layout_constraintStart_toStartOf="parent"
```

```
31                  app:layout_constraintEnd_toEndOf="parent" />
32
33              <TextView
34                  android:id="@+id/detail_title"
35                  android:layout_width="wrap_content"
36                  android:layout_height="wrap_content"
37                  android:layout_marginTop="16dp"
38                  android:textSize="20sp"
39                  android:textStyle="bold"
40                  app:layout_constraintEnd_toEndOf="parent"
41
42  app:layout_constraintStart_toStartOf="parent"
43                  tools:text="The Lord of the Rings" />
44
45              <TextView
46                  android:id="@+id/detail_year"
47                  android:layout_width="wrap_content"
48                  android:layout_height="wrap_content"
49                  android:layout_marginTop="32dp"
50                  android:textColor="#666"
51                  android:textSize="25sp"
52                  android:textStyle="italic"
53                  app:layout_constraintEnd_toEndOf="parent"
54
55  app:layout_constraintStart_toStartOf="parent"
56
57  app:layout_constraintTop_toBottomOf="@id/detail_title"
58                  tools:text="2001"                          />
59
60              <TextView
61                  android:id="@+id/detail_desc"
```

```
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginVertical="30dp"
                android:layout_marginTop="8dp"
                android:textSize="14sp"

app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintHorizontal_bias="0.0"

app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@id/detail_year"
                app:layout_constraintVertical_bias="0.037"
                tools:text="A meek Hobbit from the Shire and
eight companions set out on a journey to destroy the powerful
One                         Ring..."                         />
        </LinearLayout>


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

*Tabel 2. Source Code File fragment_detail Jawaban Soal 1 XML*

**Fragment_home.XML**

```
 1  <?xml version="1.0" encoding="utf-8"?>
 2  <androidx.constraintlayout.widget.ConstraintLayout
 3  xmlns:android="http://schemas.android.com/apk/res/android"
 4      xmlns:tools="http://schemas.android.com/tools"
 5      android:layout_width="match_parent"
 6      android:layout_height="match_parent"
 7      xmlns:app="http://schemas.android.com/apk/res-auto"
 8      tools:context=".HomeFragment">
 9
10      <androidx.recyclerview.widget.RecyclerView
11          android:id="@+id/rvFilm"
12          android:layout_width="0dp"
13          android:layout_height="0dp"
14          android:layout_margin="15dp"
15          app:layout_constraintBottom_toBottomOf="parent"
16          app:layout_constraintEnd_toEndOf="parent"
17          app:layout_constraintStart_toStartOf="parent"
18          app:layout_constraintTop_toTopOf="parent"
19          />
20
21  </androidx.constraintlayout.widget.ConstraintLayout>
```

*Tabel 3. Source Code File fragment_home.XML Jawaban Soal 1 XML*

**Activiy_main.XML**

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <FrameLayout
3  xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      tools:context=".MainActivity"
8      android:id="@+id/frame_container">
9  </FrameLayout>
```

*Tabel 4. Source Code File activity_main.XML  Jawaban  Soal 1 XML.*

**Film.kt**

```
1  package com.example.listxml
2  import android.os.Parcelable
3  import kotlinx.parcelize.Parcelize
4
5  @Parcelize
6  data class Film (
7      val title : String? = null,
8      val year : String? = null,
9      val desc : String? = null,
10     val image : Int = 0,
11     val imdb : String = "",
12     val detail : String = "", // URL for the detail page
13 ): Parcelable
```

*Tabel 5 Source Code File Film Jawaban Soal 1 XML*

**FilmAdapter.kt**

```
1   package com.example.listxml
2   import android.view.LayoutInflater
3   import android.view.ViewGroup
4   import androidx.recyclerview.widget.RecyclerView
5   import com.example.listxml.databinding.ItemFilmBinding
6
7   interface OnFilmClickListener {
8       fun onDetailClicked(film: Film)
9       fun onImdbClicked(imdbUrl: String)
10  }
11  class FilmAdapter (private val filmList: ArrayList<Film>,
12  private val listener: OnFilmClickListener)
13      : RecyclerView.Adapter<FilmAdapter.FilmViewHolder>()
14  {
15
16      inner class FilmViewHolder(val binding:
17  ItemFilmBinding) : RecyclerView.ViewHolder(binding.root)
18
19      override fun onCreateViewHolder(parent: ViewGroup,
20  viewType: Int): FilmViewHolder {
21          val binding =
22  ItemFilmBinding.inflate(LayoutInflater.from(parent.context)
23  , parent, false)
24          return FilmViewHolder(binding)
25      }
26
27      override fun onBindViewHolder(holder:
28  FilmViewHolder, position: Int) {
29          val film = filmList[position]
30          holder.binding.apply {
31              filmImage.setImageResource(film.image)
```

```
32                  filmTitle.text = film.title
33                  filmYear.text = film.year
34                  filmDesc.text = film.desc
35                  btnImdb.setOnClickListener {
36                      listener.onImdbClicked(film.imdb)
37                  }
38                  btnDetail.setOnClickListener {
39                      listener.onDetailClicked(film)
40                  }
41              }
42          }
43
44          override fun getItemCount(): Int {
45              return filmList.size
46          }
47      }
```

*Tabel 6. Source Code File FilmAdapter Jawaban Soal 1 XML*

**DetailFragment.kt**

```
1  package com.example.listxml
2
3  import android.os.Bundle
4  import androidx.fragment.app.Fragment
5  import android.view.LayoutInflater
6  import android.view.View
7  import android.view.ViewGroup
```

```
8  import com.example.listxml.databinding.FragmentDetailBinding
9
10 class DetailFragment : Fragment() {
11
12     private var _binding: FragmentDetailBinding? = null
13     private val binding get() = _binding!!
14
15     private var title: String? = null
16     private var desc: String? = null
17     private var year: String? = null
18     private var image: Int = -1
19
20     override fun onCreate(savedInstanceState: Bundle?) {
21         super.onCreate(savedInstanceState)
22
23         arguments?.let {
24             title = it.getString("EXTRA_TITLE")
25             desc = it.getString("EXTRA_DESC")
26             year = it.getString("EXTRA_YEAR")
27             image = it.getInt("EXTRA_IMAGE")
28         }
29     }
30
31     override fun onCreateView(
32         inflater: LayoutInflater, container: ViewGroup?,
33         savedInstanceState: Bundle?
34     ): View {
35         _binding  =  FragmentDetailBinding.inflate(inflater,
36 container, false)
37         return binding.root
38     }
```

```
39
40      override fun onViewCreated(view: View,
41  savedInstanceState: Bundle?) {
42          super.onViewCreated(view, savedInstanceState)
43
44          binding.detailTitle.text = title
45          binding.detailDesc.text = desc
46          binding.detailYear.text = year
47          binding.detailImage.setImageResource(image)
48      }
49
50      override fun onDestroyView() {
51          super.onDestroyView()
52          _binding = null
53      }
54  }
```

*Tabel 7 Source Code File DetailFragment Jawaban Soal 1 XML.*

**HomeFragment.kt**

```
1   package com.example.listxml
2
3   import android.content.Intent
4   import android.net.Uri
5   import android.os.Bundle
6   import androidx.fragment.app.Fragment
7   import android.view.LayoutInflater
8   import android.view.View
9   import android.view.ViewGroup
10  import androidx.recyclerview.widget.LinearLayoutManager
```

```
11 import com.example.listxml.databinding.FragmentHomeBinding
12
13 class HomeFragment : Fragment() {
14     private var _binding: FragmentHomeBinding? = null
15     private val binding get() = _binding!!
16     private lateinit var filmAdapter: FilmAdapter
17     private val list = ArrayList<Film>()
18
19     override fun onCreateView(
20         inflater: LayoutInflater, container: ViewGroup?,
21         savedInstanceState: Bundle?
22     ): View {
23         _binding = FragmentHomeBinding.inflate(inflater,
24 container, false)
25
26         list.clear()
27         list.addAll(getListFilm())
28         setupRecyclerView()
29         return binding.root
30     }
31
32     private fun setupRecyclerView() {
33         filmAdapter = FilmAdapter(
34             list,
35             object : OnFilmClickListener {
36                 override fun onDetailClicked(film: Film) {
37                     val detailFragment =
38 DetailFragment().apply {
39                         arguments = Bundle().apply {
40                             putString("EXTRA_TITLE",
41 film.title)
```

```
                                putString("EXTRA_DESC",
film.desc)
                                putInt("EXTRA_IMAGE",
film.image)
                            }
                        }

parentFragmentManager.beginTransaction()
                            .replace(R.id.frame_container,
detailFragment)
                            .addToBackStack(null)
                            .commit()
                    }

                    override fun onImdbClicked(imdbUrl: String)
{
                        val intent = Intent(Intent.ACTION_VIEW,
Uri.parse(imdbUrl))
                        startActivity(intent)
                    }
                },
            )
            binding.rvFilm.apply {
                layoutManager = LinearLayoutManager(context)
                adapter = filmAdapter
            }
        }

        private fun getListFilm(): ArrayList<Film> {
            val dataTitle =
resources.getStringArray(R.array.data_filmTitle)
```

```
73          val dataDesc =
74  resources.getStringArray(R.array.data_filmDesc)
75          val dataImage =
76  resources.obtainTypedArray(R.array.data_filmImage)
77          val dataYear =
78  resources.getStringArray(R.array.data_filmYear)
79          val dataImdb =
80  resources.getStringArray(R.array.data_filmImdb)
81          val listFilm = ArrayList<Film>()
82          for (i in dataTitle.indices) {
83              val film = Film(
84                  dataTitle[i],
85                  dataYear[i],
86                  dataDesc[i],
87                  dataImage.getResourceId(i, -1),
88                  dataImdb[i]
89              )
90              listFilm.add(film)
91          }
92          dataImage.recycle()
93          return listFilm
94      }
95  }
```

*Tabel 8. Source Code File HomeFragment Jawaban Soal 1 XML*

**MainActivity.kt**

```
1  package com.example.listxml
2
3  import android.os.Bundle
4  import androidx.appcompat.app.AppCompatActivity
5
6  class MainActivity : AppCompatActivity() {
7      override fun onCreate(savedInstanceState: Bundle?) {
8          super.onCreate(savedInstanceState)
9          setContentView(R.layout.activity_main)
10
11 val fragmentManager = supportFragmentManager
12         val homeFragment = HomeFragment()
13         val fragment =
14 fragmentManager.findFragmentByTag(HomeFragment::class.java.
15 s
16 ampleName)
17         if (fragment !is HomeFragment) {
18             fragmentManager
19                 .beginTransaction()
20                 .add(R.id.frame_container, homeFragment,
21 HomeFragment::class.java.simpleName)
22                 .commit()
23         }
24     }
25 }
```

*Tabel 9. Source Code File MainActivity Jawaban Soal 1 XML*

**Source Code Compose**

**Films**

```
1  package com.example.listxml.data
2
3  data class Films(
4      val title: String,
5      val year: String,
6      val description: String,
7      val image: Int = 0,
8      val imdbUrl: String
9  )
```

*Tabel 10. Source Code File Films.kt Jawaban Soal 1 XML.*

**FilmListItems.kt**

```
1  package com.example.listxml
2  import android.content.Intent
3  import android.net.Uri
4  import androidx.compose.foundation.Image
5  import androidx.compose.foundation.layout.Arrangement
6  import androidx.compose.foundation.layout.Column
7  import androidx.compose.foundation.layout.Row
8  import androidx.compose.foundation.layout.Spacer
9  import androidx.compose.foundation.layout.fillMaxWidth
10 import androidx.compose.foundation.layout.padding
11 import androidx.compose.foundation.layout.size
12 import androidx.compose.foundation.layout.width
```

```
13  import androidx.compose.foundation.layout.wrapContentWidth
14  import androidx.compose.foundation.shape.RoundedCornerShape
15  import androidx.compose.material3.Button
16  import androidx.compose.material3.Card
17  import androidx.compose.material3.CardDefaults
18  import androidx.compose.material3.Text
19  import androidx.compose.runtime.Composable
20  import androidx.compose.ui.Alignment
21  import androidx.compose.ui.Modifier
22  import androidx.compose.ui.draw.clip
23  import androidx.compose.ui.graphics.Color
24  import androidx.compose.ui.layout.ContentScale
25  import androidx.compose.ui.platform.LocalContext
26  import androidx.compose.ui.res.painterResource
27  import androidx.compose.ui.text.TextStyle
28  import androidx.compose.ui.text.font.FontWeight
29  import androidx.compose.ui.text.style.TextOverflow
30  import androidx.compose.ui.tooling.preview.Preview
31  import androidx.compose.ui.unit.dp
32  import androidx.compose.ui.unit.sp
33  import androidx.navigation.NavController
34  import androidx.navigation.compose.rememberNavController
```

```
35 import com.example.listxml.data.Films
36
37
38 @Composable
39 fun FilmListItems(films: Films, navController:
NavController) {
40     val context = LocalContext.current
41     Card(
42         modifier = Modifier
43             .padding(horizontal = 8.dp, vertical = 8.dp)
44             .fillMaxWidth(),
45         elevation =
CardDefaults.cardElevation(defaultElevation = 2.dp),
46         colors = CardDefaults.cardColors(containerColor =
Color(0xC6C4C2C2)),
47         shape = RoundedCornerShape(16.dp),
48     ) {
49         Row {
50             FilmImage(films = films,
51
52             )
53             Spacer(modifier = Modifier.width(8.dp))
54
55
56
```

```
57              Column(
58                  modifier = Modifier
59                      .padding(16.dp)
60                      .fillMaxWidth(),
61                  verticalArrangement = Arrangement.Bottom,
62                  horizontalAlignment = Alignment.Start
63              ) {
64                  Row(
65                      modifier = Modifier.fillMaxWidth(),
66                      horizontalArrangement                    =
Arrangement.SpaceBetween
67
68                  )
69                  {
70                      Text(
71                          text = films.title,
72                          style   =   TextStyle(fontWeight   =
FontWeight.Bold, fontSize = 12.sp),
73                          modifier = Modifier.weight(1f),
74                          maxLines = 2,
75                          overflow = TextOverflow.Ellipsis,
76                          color = Color.White
77                      )
78                      Text(
```

```
79                          text = films.year,
80                          style = TextStyle(fontWeight =
FontWeight.Bold, fontSize = 12.sp),
81
82                          color = Color.White
83
84                      )
85                  }
86                  Text(
87                      text = films.description,
88                      style = TextStyle(fontSize = 10.sp),
89                      color = Color.White
90                  )
91                  Row(
92                      modifier = Modifier.fillMaxWidth()
93                      .padding(top = 30.dp),
94                      horizontalArrangement =
Arrangement.Start
95                  ) {
96
97                      Button(
98                          onClick = {
99                              val intent =
Intent(Intent.ACTION_VIEW, Uri.parse(films.imdbUrl))
100                             context.startActivity(intent)
```

```
101                         },
102                         modifier = Modifier.padding(top =
8.dp).weight(1f).wrapContentWidth(),
103
104                         shape = RoundedCornerShape(16.dp),
105                     ) {
106                         Text(text = "IMDB",
107                             style = TextStyle(fontWeight =
FontWeight.Bold, fontSize = 8.sp)
108                         )
109                     }
110                     Spacer(modifier = Modifier.width(5.dp))
111
112                     Button(
113                         onClick = {
114
115 navController.navigate("detail/${films.title}")
116                         },
117                         modifier = Modifier.padding(top =
118 8.dp).weight(1f).wrapContentWidth(),
119                         shape = RoundedCornerShape(16.dp),
120                     ) {
121                         Text(text = "Detail",
122                             style = TextStyle(fontWeight =
FontWeight.Bold, fontSize = 8.sp)
```

```
123                             )
124                         }
125                     }
126
127                 }
128             }
129         }
130 }
131
132 @Composable
133 fun FilmImage(films: Films){
134     Image(
135         painter = painterResource(id = films.image),
136         contentDescription = null,
137         modifier = Modifier
138             .size(150.dp)
139             .padding(16.dp)
140             .clip(RoundedCornerShape(16.dp)),
141         contentScale = ContentScale.Crop,
142     )
143 }
```

*Tabel 11. Source Code File FilmListItems.kt Jawaban  Soal 1 XML.*

**DetailContent.kt**

```
1  package com.example.listxml
2
3  import androidx.compose.foundation.Image
4  import androidx.compose.foundation.layout.Arrangement
5  import androidx.compose.foundation.layout.Column
6  import androidx.compose.foundation.layout.Row
7  import androidx.compose.foundation.layout.Spacer
8  import androidx.compose.foundation.layout.fillMaxSize
9  import androidx.compose.foundation.layout.fillMaxWidth
10 import androidx.compose.foundation.layout.height
11 import androidx.compose.foundation.layout.padding
12 import androidx.compose.foundation.layout.width
13 import androidx.compose.foundation.shape.RoundedCornerShape
14 import androidx.compose.material3.Text
15 import androidx.compose.runtime.Composable
16 import androidx.compose.ui.Alignment
17 import androidx.compose.ui.Modifier
18 import androidx.compose.ui.draw.clip
19 import androidx.compose.ui.layout.ContentScale
20 import androidx.compose.ui.res.painterResource
21 import androidx.compose.ui.text.TextStyle
```

```
22 import androidx.compose.ui.text.font.FontWeight
23 import androidx.compose.ui.unit.dp
24 import androidx.compose.ui.unit.sp
25 import com.example.listxml.data.Films
26
27 @Composable
28 fun DetailContent(films: Films) {
29     Column(
30         modifier = Modifier
31             .fillMaxSize()
32             .padding(16.dp)
33     ) {
34         DetailImageContent(films = films)
35         Spacer(modifier = Modifier.height(16.dp))
36
37         // ROW: title kiri, year kanan
38         Row(
39             modifier = Modifier.fillMaxWidth(),
40             horizontalArrangement = Arrangement.SpaceBetween
41         ) {
42             Text(
43                 text = films.title,
```

```
                style = TextStyle(
                    fontWeight = FontWeight.Bold,
                    fontSize = 18.sp
                ),
                modifier = Modifier.weight(1f)
            )
            Text(
                text = films.year,
                style = TextStyle(
                    fontWeight = FontWeight.Bold,
                    fontSize = 18.sp

                )
            )
        }

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

        // DESKRIPSI
        Text(
            text = films.description,
            style = TextStyle(
```

```
                fontSize = 14.sp
            ),
            modifier = Modifier.padding(top = 8.dp)
        )
    }
}

@Composable
fun DetailImageContent(films: Films) {
    Column(
        modifier = Modifier
            .fillMaxWidth()
            .padding(16.dp),
        horizontalAlignment = Alignment.CenterHorizontally,
        verticalArrangement = Arrangement.Center
    ) {
        Image(
            painter = painterResource(id = films.image),
            contentDescription = null,
            modifier = Modifier
                .width(250.dp)
                .height(250.dp)
```

```
88                  .clip(RoundedCornerShape(16.dp)),
89              contentScale = ContentScale.Crop
90          )
91      }
92  }
```

*Tabel 12. Source Code File detailContent Jawaban  Soal 1 XML*

**HomeContent.kt**

```
1   package com.example.listxml
2
3   import androidx.compose.foundation.background
4   import androidx.compose.foundation.layout.PaddingValues
5   import androidx.compose.foundation.layout.fillMaxSize
6   import androidx.compose.foundation.lazy.LazyColumn
7   import androidx.compose.foundation.lazy.items
8   import androidx.compose.runtime.Composable
9   import androidx.compose.runtime.saveable.rememberSaveable
10  import androidx.compose.ui.Modifier
11  import androidx.compose.ui.graphics.Color
12  import androidx.compose.ui.unit.dp
13  import androidx.navigation.NavController
14  import com.example.listxml.data.DataProvider
```

```
15
16
17  @Composable
18  fun HomeContent(navController: NavController) {
19      androidx.compose.foundation.layout.Box(
20          modifier = Modifier
21              .fillMaxSize()
22              .background(Color.Black)
23      ) {
24          val films = rememberSaveable { DataProvider.filmList
    }
25
26          LazyColumn(
27              contentPadding = PaddingValues(horizontal =
    16.dp, vertical = 20.dp)
28          ) {
29              items(
30                  items = films,
31                  itemContent = {
32                      FilmListItems(films = it, navController =
33  navController)
34                  }
35              )
36          }
```

```
37      }
38  }
```

*Tabel 13. Source Code File HomeContent Jawaban  Soal 1 XML*

**MainActivity.kt**

```
1   package com.example.listxml
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.runtime.Composable
8   import androidx.navigation.NavType
9   import androidx.navigation.compose.NavHost
10  import androidx.navigation.compose.composable
11  import androidx.navigation.compose.rememberNavController
12  import androidx.navigation.navArgument
13  import com.example.listxml.data.DataProvider
14  import com.example.listxml.ui.theme.ListXMLTheme
15
16  class MainActivity : ComponentActivity() {
17      override fun onCreate(savedInstanceState: Bundle?) {
18          super.onCreate(savedInstanceState)
```

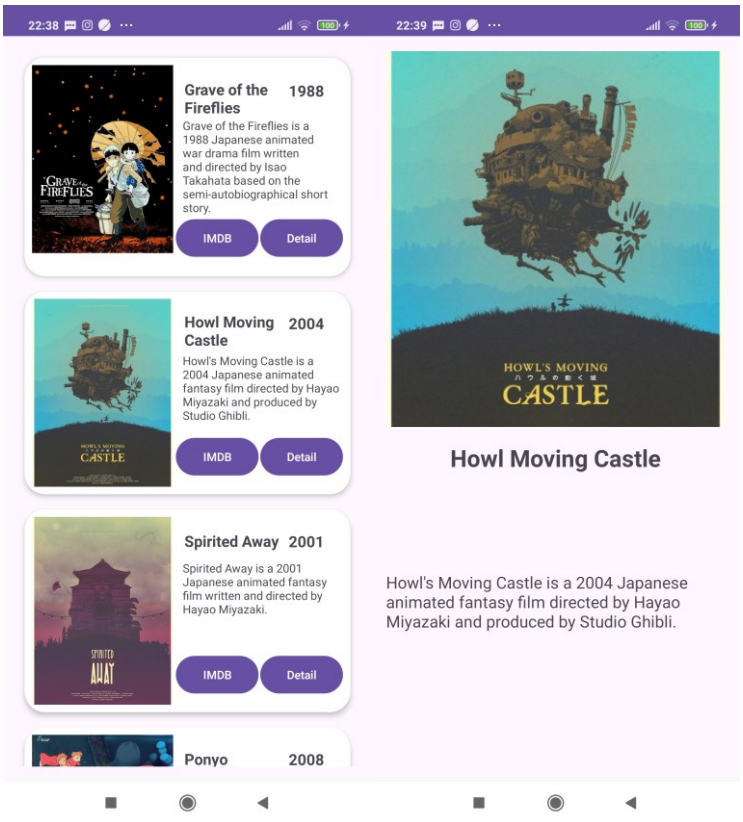
```
        enableEdgeToEdge()
        setContent {
            ListXMLTheme {
                GhibliFilms()
            }
        }
    }
}

@Composable
fun GhibliFilms () {
    val navController = rememberNavController()
    NavHost(
        navController = navController,
        startDestination = "home"
    ) {
        composable("home") {
            HomeContent(navController)
        }
        composable(
            route = "detail/{title}",
            arguments = listOf(navArgument("title") { type =
NavType.StringType }) // Ganti ke StringType
```

```
41          ) { backStackEntry ->
42              val                     title                     =
backStackEntry.arguments?.getString("title")
43
44              val film = DataProvider.filmList.find { it.title
== title }
45              if (film != null) {
46                  DetailContent(films = film)
47              }
48          }
49      }
50  }
51
52
```

*Tabel 14. Source Code File MainActivity Jawaban  Soal 1 XML*

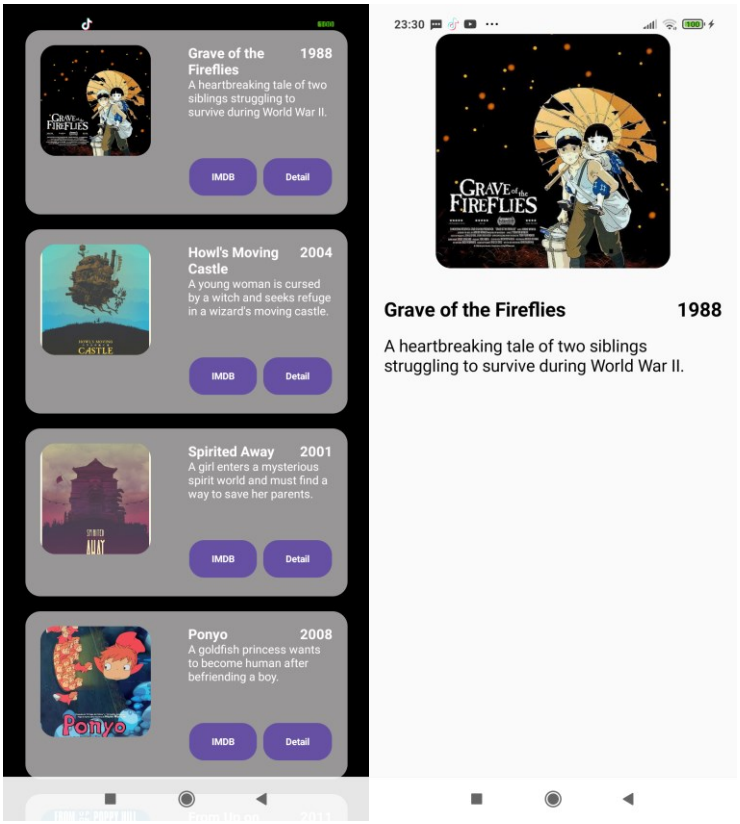
## B. Output Program

## Output Program XML




*Gambar 1. Screenshot Hasil Output Program soal 1 XML*

**Output Program Compose**




*Gambar 2. Screenshot Hasil Output Program soal 1 Compose*

## C. Pembahasan

### Pembahasan XML

### ItemList.XML

Memakai CardView untuk membuat tampilan item seperti kartu yang memiliki rounded corner dan shadow. Dengan lebar match_parent, tinggi sesuai dengan isinya, jarak antar item 8dp, sudut melengkungnya 8dp, banyangnnya setebal 4dp.

Isi CardView menggunakan Constraint layout untuk engatur posisi elemen dengan constraint, dengan lebar sesuai dengan parentnya dan tingginya menyesuaikan isinya, dan padding 8dp untuk jarak item ke tepi kartu.

Menggunakan ImageView untuk poster film yang diberi id film_image, dengan ukuran tetap (150x200 dp), menggunakan centerCrop agar gambar diperbesar dan dipotong agar mengisi penuh skalanya tanpa merubah rasio gambar, posisi gambar berada pada kiri parent.

Menggunakan TextView untuk Judul film, yang diberi id film_title, dengan ukuran 120dp, dan teks bold, yang memiliki jarak dari gambar 12dp dengan MarginStart, posisi constraintnnya dikanan gambar agak keatas,

Menggunakan TextView utnuk Tahun film, yang diberi id  film_year yang constrainntnya berada dikanan teksView Judul

Menggunakan TeksView untuk deskripsi film yang layout widthnya 0dp agar mengisi area di antara constraint start dan end, dengan ukuran teks 10sp, yang posisinya dibawah judul dan diatas tombol Imdb.

Button tombol dengan id btn_imdb yang memiliki teks dari resource string, yang posisinya sebelah kanan gammbar dan sebelah kiri tombol  detail.

**DetailFragment.XML**

ScrollView yang memungkinkan agar layar dapat discroll secara vertikal, yang widthnya memenuhi layar.

Menggunakan LinearLayout yang merupakan child dari ScrollView, yang orientasinya vertikal, jadi semua isis ditumpuk dari atas je bawah, yang gravity center agar semua elemen berada di Tengah horizontal

ImageView untuk poster film, lebar warp content dengan tinggi 400dp, menggunakan scaleType = centerCrop untuk gambar diperbesar agar menuhi area dengan memotong bagian yang tidak muat

TextView untuk judul film dengan teks sebsar 20sp dan jarak 16dp dari gambar

TextView untuk tahun rilis yang menampilkan teks sebesar 25sp, italic dan warna abu

TextView untuk deskripsi film  dengan teks sebesar 14sp, dan memiliki jarak 8dp dari tahun, MarginVertical 30dp → Jarak atas & bawah luas, agar teks tidak menumpuk.

**Fragment_Home.XML**

Kode XML ini adalah layout untuk HomeFragment yang menggunakan ConstraintLayout sebagai root layout. Pada bagian atas, terdapat deklarasi namespace seperti xmlns:android, xmlns:app, dan xmlns:tools. Namespace app digunakan untuk atribut khusus ConstraintLayout, sedangkan tools

dipakai untuk preview di Android Studio dan menunjukkan bahwa layout ini milik HomeFragment (tools:context=".HomeFragment"). Layout ini berukuran match_parent baik untuk lebar maupun tinggi, artinya akan memenuhi seluruh ukuran layar.

Di dalam ConstraintLayout, terdapat sebuah RecyclerView dengan ID rvFilm, yang akan digunakan untuk menampilkan daftar film dalam bentuk list. RecyclerView ini memiliki layout_width dan layout_height sebesar 0dp, yang berarti ukurannya ditentukan oleh constraint yang diterapkan (disebut match constraints dalam ConstraintLayout). RecyclerView ini diberi margin 15dp di semua sisi agar ada jarak dari tepi layar.

Constraint diterapkan dengan app:layout_constraintTop_toTopOf="parent", app:layout_constraintBottom_toBottomOf="parent",app:layout_constraintStart_toStartOf="parent",dan app:layout_constraintEnd_toEndOf="parent", yang artinya RecyclerView menempel ke semua sisi parent (ConstraintLayout). Karena ukuran 0dp dengan constraint penuh seperti ini, RecyclerView akan mengisi seluruh ruang yang tersedia di dalam ConstraintLayout, namun tetap memberi jarak 15dp dari tepi berkat margin yang ditetapkan.

**Activity_Main.XML**

FrameLayout ini memiliki layout_width dan layout_height sebesar match_parent, yang berarti layout ini akan memenuhi seluruh ukuran layar perangkat. FrameLayout sendiri adalah layout sederhana yang biasanya digunakan untuk menampung satu elemen (atau beberapa elemen yang ditumpuk), dan d FrameLayout diberi id @+id/frame_container.

Pemberian id frame_container menunjukkan bahwa layout ini kemungkinan besar digunakan sebagai wadah (container) untuk fragment. Biasanya dalam arsitektur aplikasi berbasis fragment, FrameLayout berfungsi sebagai tempat fragment dimasukkan dan digantikan (replace) selama navigasi aplikasi. Karena FrameLayout ini tidak memiliki child (kosong), perannya memang hanya sebagai container dinamis yang akan diisi fragment lewat FragmentTransaction dari kode Kotlin.

**Film.kt**

@Parcelize merupakan anotasi Kotlin yang otomatis menghasilkan implementasi Parcelable (tanpa harus manual pakai writeToParcel, describeContents, dll).

Parcelize memerintahkan compiler Kotlin untuk membuat parcelable untuk class film, data class otomatis punya equals(), hashCode(), toString() copy() → Untuk salin objek dan ubah properti

tertentu. componentN() → Untuk destructuring (misal val (title, year) = film). Data class berisi title bertip string yang nullable, year tipe string yang defaultnya null, desc untuk deskripsi film yang defaultnya null, val image yang bertipe int karena resource id di anroid bertipe int, default 0, yang artinya tidak ada gambar, imdb untuk link imbd dengan tipe string non-nullabel, dengan deafult empty string, detail, untuk ke halaman detail, bertipe strinf, mengimplemnatiksan parcelable dibawah class film.

**FilmAdapter.kt**

Import library LayoutInflater = objek untuk 'menyulap' XML layout jadi View nyata di Kotlin. ViewGroup = parent view container untuk item RecyclerView. Import class RecyclerView → komponen daftar modern Android. Import View Binding class yang otomatis dibuat dari file item_film.xml. Membuuat Interface "kontrak" fungsi yang wajib di-implement siapa pun yang pakai adapter, fungsi onDetailClicked merupakan fungsi yang akan dipanggil saat tombol detail di-lik, yang mengirim object film, agar halaman detail bisa dibuka, fungsi fun onImdbClicked mendeklrasikan adapter, class adapter yang berfungsi untuk menjembatani data dengan tamplian RecyclerView, val filmlist parameter list data film, vaal listener parameter listener untuk implementasi interface untuk klik tombol. Membuat ViewHolder yang memegang satu item tamplian pada RecycleView, menggunakan inner class agar bisa mengakses variabel/method di FilmAdapter. Override onCreateViewHolder() Dipanggil sekali saat RecyclerView butuh ViewHolder baru. Inflasi layout item_film.xml → jadi objek view binding. Return ViewHolder yang pegang binding. Override onBindViewHolder Dipanggil setiap kali RecyclerView mau menampilkan data baru di item (posisi position). Ambil data film sesuai posisi binding.apply { ... }. Ovveride GetItemCoubt Menentukan berapa banyak item di RecyclerView. Return ukuran filmList.

**DetailFragment.kt**

Mendeklarasikan class DetailFragment, sebagai turunan class Fragment, : fragment() artinya DetailFragment merupakan subclass dari androidx.fragment.app.Fragment. Mendeklarasikan variabel private var _binding untuk menyimpan objek ViewBinding yang menghubungkan layout XML (fragment_detail.xml) dengan kode Kotlin. Tipe data yang digunakan adalah nullable (FragmentDetailBinding?) untuk menyesuaikan siklus hidup fragment.

private val binding get() = _binding!!: Properti ini menyediakan akses non-null terhadap objek binding. Penggunaan !! menunjukkan bahwa _binding tidak boleh null saat properti ini diakses. Pola ini bertujuan untuk meminimalkan NullPointerException saat View sudah tersedia.

Mendeklarasikan variabel data yang digunakan utnk menyimpan nilai yang berasal dari arguments Menggunakan Nullaabel bisa saja argument tidak mengirimkan semua data.

Mengambil arguments onCreate methods, mengecek argyment apakah tidak null, jika tidak maka let akan mengeksuki blok dan mengambil dengan get.

Inflasi Layout onCreateView, method ini digunakan untuk membuat dan mengembalikan View fragment, inflasi dilakukan dengan mengubah layout XML menjadi objek ViewBinding. inflater: Objek untuk mengkonversi file XML ke dalam View. container: Parent ViewGroup tempat fragment akan ditempelkan. false: Menandakan bahwa View tidak langsung ditambahkan ke parent. Menyimpan hasil binding ke variabel _binding, dan mengembalikan root view dari layour sebagai hasil dari onCreateView

Menampilkan Data ke View Method onViewCreated, metod yang akan dipanggil setelah view selsai dibuat oleh onCreateView, binding data dengaan Id yang sesuai

Membershikan binding dengan methods onDestroyView, metode yang dipanggil saat view fragment dihancurkan, dan Menghapus referensi objek ViewBinding agar tidak terjadi memory leak akibat referensi View yang masih tertahan setelah View dihapus dari memori.

**HomeFragment.kt**

FragmentHomeBinding Menghubungkan layout XML (fragment_home.xml) ke kode Kotlin menggunakan View Binding.

import ...: Mengimpor kelas-kelas yang digunakan, seperti Fragment, Intent, Uri, LinearLayoutManager, dan View Binding (FragmentHomeBinding), Baris Import Fungsi Utama

Intent, Uri Membuka halaman IMDb menggunakan browser (implicit intent). Bundle Mengirim data antar fragment (DetailFragment). Fragment Membuat fragment(HomeFragment). LayoutInflater, View, ViewGroup Membuat dan mengelola tampilan fragment. LinearLayoutManagerMengatur tata letak list RecyclerView. MEndeklarasikan Xlaa

HomeFragment yang merupakan turunan dari Fragment, Variabel binding _bindingl yang merupakan variabel nullable untuk menyimpan instance binding (diatur menjadi null untuk mencegah memory leak) val binding: property non-nullable dengan delegasi getter . Variabel Adapter dan Data, lateinit var filmAdapter untuk recycleView dan val list ArrayList yang menayimpan data film.

Metode onCreateView, mengoverride untuk fragment untuk membuat tampilan, dengan Parameter, inflater: Objek untuk mengubah layout XML menjadi View. container: ViewGroup induk. savedInstanceState: Data instance sebelumnya (jika ada).

Menginisialisasi binding menggunakan layout XML fragment_home.xml.

Persiapan data dan RecycleVIew List.clear() memberishkan list agar tidak terjadi duplikasi, list.addAll(getListFilm()): Menambahkan data film dari metode getListFilm().setupRecyclerView(): Memanggil metode untuk menyiapkan RecyclerView. Mengembalikan binding.root.

Methods setupRecyclerView, merupakan metode private untuk konfogurasi RecyclerView dan adapter, menginialisasi adapter Membuat instance FilmAdapter dengan parameter list dan listener klik anonim. Mengimplementasi klik listener membuat instannce DetailFragment dan mengisi arguments dengan Bundle berisi data film (title, desc, image). Mengganti fragment yang tampil dengan DetailFragment menggunakan FragmentTransaction dan addToBackStack(null): Memungkinkan pengguna kembali ke fragment sebelumnya dengan tombol back. onImdbClicked Membuka link Imdb menggunakan implicit intent. Mengkonfigurasi RecyclerView layoutManager untuk RecyclerVIew menggunakan layout list vertikal, dan adapter menetapkan filmAdapter ke RecyclerView.

Metod getListFilm menggunakan metod private untuk mengambil data film fari resource., mengambil Data Resource, dan disimpan kedalam dataTitle, dataDesc, dataImage, dataYear, dataImdb. Memasukkan data ke list, Melakukan looping berdasarkan indeks dataTitle, dan membuat objek film dan menabahkan ke listFilm, dataImage.recycle() Melepas TypedArray dari memori setelah digunakan dan Mengembalikan list data film.

**MainActivity.kt**

class MainActivity: Mendefinisikan kelas bernama MainActivity sebagai kelas utama (entry point) aplikasi.: AppCompatActivity(): Menyatakan bahwa MainActivity merupakan subclass dari AppCompatActivity. AppCompatActivity merupakan kelas dasar untuk activity yang mendukung fitur ActionBar dan kompatibilitas lintas versi Android.

Override Metode onCreate, mendefiniskan metode onCreate yang dipanggil activity pertama kali dibuat. savedInstanceState: Bundle?: Parameter ini menyimpan data state activity sebelumnya (jika ada) saat terjadi rekreasi activity (seperti rotasi layar). super.onCreate(savedInstanceState): Memanggil implementasi onCreate dari superclass (AppCompatActivity) untuk memastikan proses inisialisasi standar tetap berjalan.

Menentukan Layout Activity, setContentView mmemasang layout XML activity_main.xml sebagai antarmuka pengguna untuk activity ini, R.layout.activity_main: Merujuk ke file layout res/layout/activity_main.xml yang berisi komponen antarmuka (seperti FrameLayout yang nanti menjadi container fragment).

Sehabi layout ditentukn aplikasi menginisialisasi FragmentManager melalui perintah val fragmentManager = supportFragmentManager. FragmentManager bertanggung jawab untuk menangani semua transaksi terkait fragment dalam activity, seperti menambahkan, mengganti, atau menghapus fragment. Kemudian dibuat objek HomeFragment baru dengan perintah val homeFragment = HomeFragment(). Objek ini mewakili fragment yang akan ditampilkan dalam container activity.

Aplikasi melakukan pengecekan apakah fragment dengan yang sama sudaha dalam FragmentManager, findFragmentByTag untuk memastikan apakah fragment dengan tag nama kelas HomeFragment sudah ada. Jika belum ada, maka fragment ditambahkan ke container menggunakan beginTransaction().add().commit().

**Pembahasan Compose**

**Films.kt**

Data class Bernama Films, dengan berisi title bertipe string, year bertipe string, description bertipe string, imge bertipe int dan immutable artinya jika nilai ini tidak diberikan saat objek dibuat, maka secara otomatis bernilai 0, imdbUrl bertipe string

**FilmListItems.kt**

Bagian import Mengimpor kelas Intent dan Uri yang digunakan untuk membuat intent eksplisit (membuka URL) di Android. Mengimpor komponen dasar UI Jetpack Compose: Image untuk gambar, serta layout Row, Column, Spacer, Modifier seperti padding, size, dan fillMaxWidth. RoundedCornerShape digunakan untuk membuat sudut elemen UI menjadi membulat. Mengimpor komponen Material Design 3 seperti Button, Card, Text, serta properti pendukung kartu seperti CardDefaults. Menandakan fungsi composable, yaitu fungsi yang membangun UI dalam Jetpack Compose. Mengimpor elemen dari package UI seperti Modifier, Color, Alignment, ContentScale (pengaturan skala gambar), painterResource (mengambil gambar dari resource), TextStyle, dan properti teks. Mengimpor elemen navigasi Compose: NavController untuk mengelola perpindahan antar layar dan rememberNavController untuk membuat instance NavController. Mengimpor model data Films dari package data.

Fungsi FilmListItems Mendeklarasikan fungsi composable bernama FilmListItems. Fungsi ini menerima parameter films (objek Films) dan navController (untuk navigasi layar), val context mengambil context saat ini dari compose, yang diperlukan saar membuat intent Android, Membuat card dengan padding 8 dp dan lebar penuh, dan menetapkan bayangan kartu sebesar 2dp, Menetapkan warna latar kartu dengan kode warna abu abu, dan roundedCornerShape untuk membuat sudut melengkunnng 16dp. Membuat row untuk menyusun kartu secara horizontal, memanggil fungsi composable FilmImage utuk menampilkan gambar film, memberi jarak 8dp dengan spacer untuk jarak antara gambar dan teks. Memmbuat column didalam row unruk menyusun judul,deskripsi, dan tombol dengan padding 16dp dan lebar penuh. Membuat row didalam untuk memuat judul dan tahun film, dan diberi SpaceBetween diantar keduanya, Menampilkan judul film dengan teks tebal ukuran 12sp, maksimal 2 baris (dengan elipsis jika kepanjangan). Menampilkan tahun film dengan teks tebal 12sp dan warna putih. Menampilkan deskripsi film dengan ukuran teks 10sp warna putih. Membuat baris tombol dengan padding atas 30dp. Tombol pertama (IMDB). Saat ditekan, membuka URL IMDb menggunakan intent browser. Tombol diberi padding atas 8dp, bobot 1 (membagi ruang), dan sudut membulat. Teks dalam tombol IMDB (tebal, 8sp). Memberi jarak horizontal 5dp antar tombol. Tombol kedua (Detail). Saat ditekan, berpindah ke halaman detail dengan argumen title. Teks dalam tombol Detail (tebal, 8sp).

Fungsi FilmImage mendefinisikan fungsi composable FilmImage untuk menampilkan gambar film. Image untuk menampilkan gambar dari resource ID (poster film), contentDescription = null, Tidak ada deskripsi gambar (aksesibilitas). Modifier mengatur gambar berukuran 150dp, padding 16dp, dan sudut melengkung 16dp, gambar dicrop agar memenuhi kotak tampilan

**DetailContent.kt**

Penjelasan Import Image → Komponen untuk menampilkan gambar di Jetpack Compose (seperti <ImageView> di XML). Arrangement → Mengatur posisi child dalam layout, Column → Layout vertikal untuk menyusun komponen dari atas ke bawah, Row → Layout horizontal untuk menyusun komponen dari kiri ke kanan, Spacer → Komponen kosong untuk memberikan jarak/ruang antar elemen UI, fillMaxSize → Modifier untuk membuat komponen memenuhi seluruh ukuran parent (100% lebar dan tinggi), fillMaxWidth → Modifier untuk membuat komponen memenuhi seluruh lebar parent, height → Modifier untuk menentukan tinggi spesifik suatu komponen, padding → Modifier untuk memberi jarak di dalam/bagian dalam komponen, width → Modifier untuk menentukan lebar spesifik suatu komponen, RoundedCornerShape → Untuk membuat sudut komponen membulat (digunakan untuk gambar atau card), Text → Widget Compose untuk menampilkan teks di layar, @Composable → Anotasi yang menunjukkan bahwa fungsi ini membuat bagian dari UI. Alignment → Untuk menyusun posisi komponen dalam layout, seperti Alignment.CenterHorizontally, Modifier → Untuk memodifikasi tampilan atau perilaku suatu komponen (contoh: padding, size, alignment), clip → Modifier untuk memotong (meng-clip) komponen mengikuti bentuk tertentu (contohnya sudut membulat), ContentScale → Mengatur bagaimana gambar mengisi ruangnya (contoh: ContentScale.Crop akan crop gambar agar memenuhi ruang), painterResource → Digunakan untuk mengambil gambar dari file drawable berdasarkan resource ID, TextStyle → Untuk mengatur style dari teks seperti warna, ukuran font, berat font (bold/normal), FontWeight → Untuk mengatur ketebalan huruf (contoh: FontWeight.Bold, FontWeight.Light), dp → Satuan ukuran density-independent pixel di Compose (seperti dp di XML), sp → Satuan ukuran untuk teks, yang mempertimbangkan preferensi ukuran teks user, Films → Data class milik kamu sendiri (data class Films) yang berisi informasi tentang film (judul, tahun, deskripsi, gambar, dan URL IMDB).

Membuat fungsi Composable bernama Detailcontent yang fungsinya menerima parameter films: films, artinya data detail 1 film (title, year, image, dll). Membuat Colum untuk layout vertikal,

modifier.fillMaxSize agar kolom memenuhi layar penuh dan memberi padding untuk jarak tepi 16dp, Memanggil fungsi DetailImageContent untuk menampilkan gambar film. Mengasih jarak vertikal 16dp antara gambar dan teks. Row untuk layout horizontal untuk teks title dan year, fillMaxWidth(), Row selebar layar. Dan spaceBetween agar komponen kiri dan kanan berada di ujung. Menampilan judul film dengan bold dan ukuran font 18sp, dan weigth(1f) Biar teks judul mengisis ruang sisa disebelah kiri. Menampilkan tahun film Text() dengan style yang sama dengan title dan tidak menggunakan weigth biar tetap kecil di kanan. Memberikan jarak vertikal 8dp antara row dan deskripsi, Menampilkan deskripsi film dengan Text() dengan ukuran 14sp dan diberi padding atas 8dp biar memberi jarak

Membuat fungsi DetailImageContent dengan paramter films: Films, Membuat Column untuk layout vertikal dengan Modifier: lebar memenuhi layar dengan padding 16dp, dengag centeHorizontally agar isi kolom ditang horizontal, dan verticalArrangement.Center, isi kolom di tengah vertikal, menampilkan gambar image() dari resource pakai id dari films.image dan contentDescription = null karena ini gambar dekorasi, gak butuh deskripsi aksesibilitas, gambar dengan ukuran 250x250dp dan menjadi rounded corder dengan ukuran 16dp, Crop gambar supaya penuh kotak 250x250, meski bagian gambar terpotong

**HomeContent.kt**

import androidx.compose.foundation.background → Untuk memberi warna latar belakang pada composable (seperti Box atau Column). import androidx.compose.foundation.layout.PaddingValues → Untuk memberi padding (jarak) di dalam komponen seperti LazyColumn. import androidx.compose.foundation.layout.fillMaxSize → Modifier agar komponen memenuhi lebar dan tinggi layar. import androidx.compose.foundation.lazy.LazyColumn → List scrollable yang efisien (seperti RecyclerView versi Compose). import androidx.compose.foundation.lazy.items → Untuk mengulangi daftar item di LazyColumn (seperti adapter di RecyclerView). import androidx.compose.runtime.Composable → Anotasi agar fungsi bisa dipanggil di UI Compose. import androidx.compose.runtime.saveable.rememberSaveable → Untuk menyimpan nilai agar tetap ada saat rotasi layar (state safe). import androidx.compose.ui.Modifier → Modifier untuk mengatur tampilan, ukuran, padding, dll.

import androidx.compose.ui.graphics.Color → Untuk mengatur warna. import androidx.compose.ui.unit.dp → Satuan ukuran density-independent pixel untuk padding, margin, size. import androidx.navigation.NavController → Digunakan untuk navigasi antar halaman (Composable). import com.example.listxml.data.DataProvider → Mengambil data list film dari DataProvider (file data buatan sendiri).

Fungsinya: Menampilkan daftar film (list) di halaman Home.

Membuat fungsi Composable bernama HomeContent yang menerima parameter navController: NavController, artinya fungsi ini bisa melakukan navigasi saat item film diklik. Membungkus isi konten dengan Box layout, tujuannya supaya isi bisa ditumpuk dan memenuhi layar penuh. Modifier yang digunakan adalah fillMaxSize() agar Box selebar dan setinggi layar, lalu diberi background(Color.Black) supaya latar belakang layar berwarna hitam.

Membuat variabel films yang berisi list film dari DataProvider.filmList. Menggunakan rememberSaveable agar data list film tetap aman saat orientasi layar berubah (misalnya: rotate screen). Setelah itu, membuat LazyColumn yaitu komponen list yang bisa di-scroll secara efisien. LazyColumn diberi contentPadding yaitu jarak horizontal 16dp dan vertikal 20dp agar isi list tidak menempel pinggir layar.

Di dalam LazyColumn, menggunakan items untuk mengulangi data films. Setiap item film akan dipanggil ke dalam fungsi FilmListItems, dengan parameter films = it (artinya tiap film di-iterasi), dan navController diteruskan agar item bisa melakukan navigasi saat diklik.

**MainActivity.kt**

import android.os.Bundle → Untuk menangani bundle lifecycle saat activity dibuat. import androidx.activity.ComponentActivity → Base class untuk Activity yang mendukung Jetpack Compose. import androidx.activity.compose.setContent → Untuk mengganti konten activity dengan composable (UI Compose). import androidx.activity.enableEdgeToEdge → Agar UI bisa menggambar hingga tepi layar (immersive, edge-to-edge layout). import androidx.compose.runtime.Composable → Anotasi agar fungsi bisa dipanggil di UI Compose. import androidx.navigation.NavType → Digunakan untuk menentukan tipe data argument dalam navigation route. import androidx.navigation.compose.NavHost → Container untuk navigasi antar Composable (seperti fragment container). import androidx.navigation.compose.composable →

Untuk mendefinisikan halaman (screen) composable dalam NavHost. import androidx.navigation.compose.rememberNavController → Untuk membuat dan menyimpan NavController di Compose. import androidx.navigation.navArgument → Untuk mendeklarasikan argument di route navigasi. import com.example.listxml.data.DataProvider → Mengambil data list film dari DataProvider (file data buatan sendiri). import com.example.listxml.ui.theme.ListXMLTheme → Menggunakan tema custom (warna, typography) yang dibuat sendiri.

Kode ini adalah implementasi navigasi antar halaman pada aplikasi menggunakan Jetpack Compose Navigation. Pada bagian import, beberapa komponen penting digunakan: ComponentActivity sebagai base class untuk activity berbasis Compose, setContent untuk menampilkan UI dari composable, serta enableEdgeToEdge agar tampilan UI aplikasi bisa memenuhi layar hingga tepi (immersive). Import lainnya adalah untuk navigasi, seperti NavHost, composable, rememberNavController, dan navArgument yang digunakan dalam pengelolaan navigasi antar composable. DataProvider digunakan sebagai sumber data list film, sedangkan ListXMLTheme dipakai untuk menerapkan tema aplikasi.

Di dalam MainActivity, fungsi onCreate memanggil enableEdgeToEdge dan setContent. Di dalam setContent, digunakan ListXMLTheme agar aplikasi memakai tema yang sudah ditentukan, lalu memanggil fungsi composable GhibliFilms() sebagai tampilan utama aplikasi. Fungsi GhibliFilms bertugas untuk mengatur navigasi. Pertama, membuat NavController dengan rememberNavController untuk menyimpan status navigasi. Lalu menggunakan NavHost dengan parameter startDestination = "home" agar halaman pertama yang ditampilkan adalah halaman home.

Di dalam NavHost, ada dua route yang didefinisikan. Route pertama adalah "home", yang menampilkan HomeContent(navController) agar halaman home bisa muncul dan menerima NavController untuk navigasi lebih lanjut. Route kedua adalah "detail/{title}", yang artinya halaman detail menerima argument title (judul film). Argument ini didefinisikan sebagai navArgument("title") { type = NavType.StringType }. Ketika halaman detail dibuka, argument title diambil dari backStackEntry.arguments, lalu dicari data film yang sesuai dengan judul tersebut dari DataProvider.filmList menggunakan fungsi .find. Jika data film ditemukan (film ≠

null), maka halaman detail akan menampilkan DetailContent(films = film) untuk menunjukkan detail film yang dipilih oleh user.

Secara keseluruhan, kode ini berfungsi untuk mengatur alur navigasi sederhana dari halaman daftar film (home) menuju halaman detail film (detail), menggunakan Jetpack Compose dan Navigation Component tanpa XML. Semua pengelolaan UI dilakukan dalam kode Kotlin berbasis composable

## SOAL 2

**Pembahasan**

Fleksibilitas: RecyclerView memberi lebih banyak kontrol, melalui implementasi kustomisasi seperti pengelolaan jenis item, animasi, dan interaksi seperti swipe dan menarik-dan-menjatuhkan, yang sulit ditentukan dengan LazyColumn.

Performa untuk Daftar Besar: Di sisi lain, RecyclerView jauh lebih efisien dengan daftar panjang, berkat teknik view recycling dan manajemen memori yang lebih baik. Oleh karena itu, itu lebih sesuai untuk situasi di mana aplikasi Anda mungkin memerlukan daftar panjang.
Kontrol Layout:

RecyclerView memiliki banyak pilihan layout manager yang lebih kompleks dibandingkan dengan yang sudah tersedia pada LazyColumnKontrol

Layout: RecyclerView menawarkan berbagai pilihan layout manager (seperti grid atau staggered grid) yang lebih kompleks daripada yang disediakan LazyColumn.

Kompatibilitas: Karena sudah lama ada, RecyclerView didukung oleh banyak library dan aplikasi lama, sementara LazyColumn masih baru dan terbatas pada Jetpack Compose.

Multiple View Types: RecyclerView lebih mudah menangani berbagai jenis tampilan dalam satu daftar, memberikan kontrol lebih besar saat menampilkan data yang beragam.

Meskipun LazyColumn lebih sederhana dan cocok untuk aplikasi yang lebih ringan, RecyclerView tetap menjadi pilihan utama untuk aplikasi yang membutuhkan kontrol lebih besar dan pengelolaan daftar yang lebih kompleks.

**LINK GITHUB**

https://github.com/raymondhariyono/mobile-praktikum/tree/main/modul3