LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 3

BUILD A SCROLLABLE LIST

Oleh:

Muhammad Raihan NIM. 2310817110008

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
MEI 2025

LEMBAR PENGESAHAN
LAPORAN PRAKTIKUM PEMROGRAMAN I
MODUL 3

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

Nama Praktikan : Muhammad Raihan
NIM : 2310817110008

Menyetujui, Asisten Praktikum	Mengetahui, Dosen Penanggung Jawab Praktikum
Muhammad Raka Azwar NIM. 2210817210012	Andreyan Rizky Baskara, S.Kom., M.Kom. NIP. 19930703 201903 01 011

DAFTAR ISI

DAFTAR GAMBAR

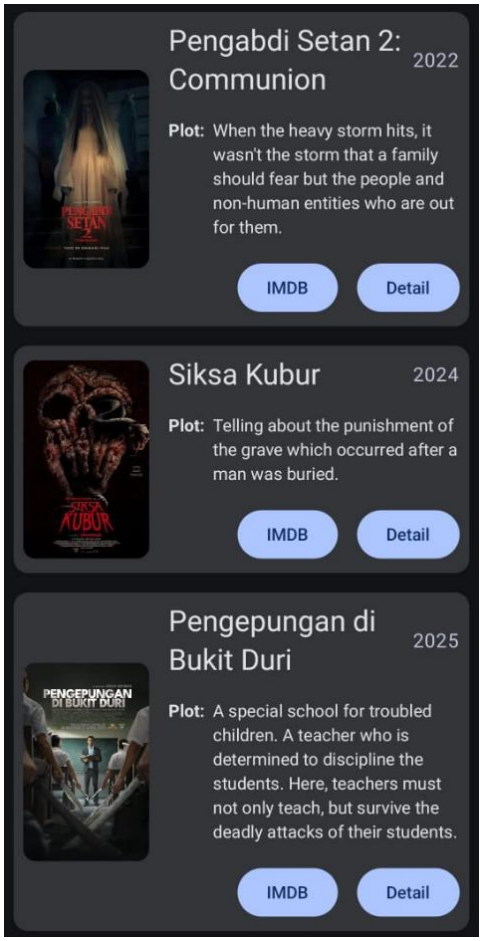
DAFTAR TABEL

SOAL 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

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 1. 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 2. Contoh UI Detail

A. Source Code

1) Versi Jetpack Compose

MainActivity.kt

Tabel 1. Source Code Jetpack Compose MainActivity.kt Soal 1

```
package com.example.scrollablecompose

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.activity.enableEdgeToEdge
import androidx.compose.runtime.Composable
import androidx.navigation.compose.NavHost
import androidx.navigation.compose.composable
import
androidx.navigation.compose.rememberNavController
import
com.example.scrollablecompose.screens.DetailScreen
import
com.example.scrollablecompose.screens.ListScreen
import
com.example.scrollablecompose.ui.theme.ScrollableCom
poseTheme
```

```
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState:
Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContent {
            ScrollableComposeTheme {
                MyFavKamenRiderApp()
            }
        }
    }
}

@Composable
fun MyFavKamenRiderApp() {
    val navController = rememberNavController()
    NavHost(navController = navController,
startDestination = Routes.listScreen, builder = {
        composable(Routes.listScreen) {
            ListScreen(navController)
        }
        composable(Routes.detailScreen+"/{id}") {
            val idString =
it.arguments?.getString("id")
            val id = idString?.toIntOrNull() ?: 0
            DetailScreen(navController, id)
        }
    })
}
```

Card.kt

Tabel 2. Source Code Jetpack Compose Card.kt Soal 1

```
package com.example.scrollablecompose.screens

import android.content.Context
import android.content.Intent
import android.content.res.Configuration
import android.net.Uri
import androidx.compose.foundation.Image
import
androidx.compose.foundation.layout.Arrangement
import androidx.compose.foundation.layout.Column
import
androidx.compose.foundation.layout.PaddingValues
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.layout.size
import androidx.compose.foundation.layout.width
import
androidx.compose.foundation.layout.wrapContentHeight
import
androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.material3.Button
import androidx.compose.material3.Card
import androidx.compose.material3.CardDefaults
import androidx.compose.material3.MaterialTheme
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.draw.clip
import androidx.compose.ui.layout.ContentScale
import androidx.compose.ui.platform.LocalContext
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.res.stringResource
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import androidx.navigation.NavController
import com.example.scrollablecompose.R
import com.example.scrollablecompose.Routes
import
com.example.scrollablecompose.model.KamenRider

@Composable
fun   RiderCard(rider:   KamenRider,   navController:
NavController, orientationMode: Int) {
    val context = LocalContext.current

    if(orientationMode                                  ==
Configuration.ORIENTATION_PORTRAIT){
        RiderCardPortrait(rider,      navController,
context)
    }
    else{
        RiderCardLandscape(rider,     navController,
context)
```

```
    }

}

@Composable
fun       RiderCardPortrait(rider:       KamenRider,
navController: NavController, context: Context){
    Card(
        modifier = Modifier
            .padding(7.dp)
            .fillMaxWidth()
            .wrapContentHeight(),
        shape = MaterialTheme.shapes.medium,
        colors = CardDefaults.cardColors(
            containerColor                              =
MaterialTheme.colorScheme.surface
        ),
        elevation = CardDefaults.cardElevation(
            defaultElevation = 5.dp
        )
    ) {
        Row(
            verticalAlignment = Alignment.Top,
            modifier = Modifier.padding(5.dp)
        ) {
            Image(
                painter   =   painterResource(id   =
rider.posterRes),
                contentDescription     =     "Poster
${rider.name}",
                modifier = Modifier
                    .size(width  =  120.dp,  height  =
150.dp)
                    .padding(8.dp)

.align(Alignment.CenterVertically)

.clip(RoundedCornerShape(15.dp)),
                contentScale                            =
ContentScale.FillBounds,
            )
            Column(Modifier.padding(8.dp)) {
                Row (
                    modifier = Modifier
                        .fillMaxWidth()
                ) {
                    Text(
```

```
                        text = rider.name,
                        fontSize = 20.sp,
                        fontWeight                    =
FontWeight.Bold,
                        color                         =
MaterialTheme.colorScheme.onSurface,
                        modifier = Modifier

.align(Alignment.CenterVertically)
                            .weight(1f)
                    )
                    Spacer(Modifier.width(20.dp))
                    Text(
                        text                          =
rider.year.toString(),
                        style                         =
MaterialTheme.typography.labelMedium,
                        fontSize = 15.sp,
                        color                         =
MaterialTheme.colorScheme.onSurface,
                        modifier                      =
Modifier.align(Alignment.CenterVertically)
                    )
                }

                Spacer(Modifier.height(8.dp))
                Text(
                    text                              =
stringResource(R.string.description),
                    style                             =
MaterialTheme.typography.labelMedium,
                    color                             =
MaterialTheme.colorScheme.onSurface
                )
                Text(
                    text = rider.description,
                    style                             =
MaterialTheme.typography.bodySmall,
                    color                             =
MaterialTheme.colorScheme.onSurface,
                    textAlign = TextAlign.Justify
                )
                Spacer(Modifier.height(8.dp))

                Row (
                    Modifier.fillMaxWidth(),
```

```
                    horizontalArrangement           =
Arrangement.End
                ) {
                    Button(
                        shape                       =
RoundedCornerShape(12.dp),
                        contentPadding              =
PaddingValues(horizontal = 15.dp),
                        onClick = {
                            val       intent        =
Intent(Intent.ACTION_VIEW)
                                intent.data         =
Uri.parse(rider.imdbUrl)

intent.setPackage("com.android.chrome")

context.startActivity(intent)
                        }
                    ) {

Text(stringResource(R.string.imdb), fontSize = 13.sp)
                    }
                    Spacer(Modifier.width(10.dp))
                    Button(
                        shape                       =
RoundedCornerShape(12.dp),
                        contentPadding              =
PaddingValues(horizontal = 15.dp),
                        onClick = {

navController.navigate(Routes.detailScreen+"/${rider
.id}")
                        }
                    ) {

Text(stringResource(R.string.detail),   fontSize   =
13.sp)
                    }
                }
            }
        }
    }
}

@Composable
fun       RiderCardLandscape(rider:       KamenRider,
navController: NavController, context: Context){
```

```
    Card(
        modifier = Modifier
            .padding(7.dp)
            .fillMaxWidth()
            .wrapContentHeight(),
        shape = MaterialTheme.shapes.medium,
        colors = CardDefaults.cardColors(
            containerColor                              =
MaterialTheme.colorScheme.surface
        ),
        elevation = CardDefaults.cardElevation(
            defaultElevation = 5.dp
        )
    ) {
        Row(
            verticalAlignment = Alignment.Top,
            modifier = Modifier.padding(5.dp)
        ) {
            Image(
                painter   =   painterResource(id   =
rider.posterRes),
                contentDescription     =     "Poster
${rider.name}",
                modifier = Modifier
                    .size(width  =  192.dp,  height  =
240.dp)
                    .padding(8.dp)

.align(Alignment.CenterVertically)

.clip(RoundedCornerShape(15.dp)),
                contentScale                            =
ContentScale.FillBounds,
            )
            Spacer(Modifier.width(5.dp))
            Column(Modifier.padding(8.dp)) {
                Row (
                    modifier = Modifier
                        .fillMaxWidth()
                ) {
                    Text(
                        text = rider.name,
                        fontSize = 30.sp,
                        fontWeight                      =
FontWeight.Bold,
                        color                           =
MaterialTheme.colorScheme.onSurface,
```

```
                        modifier = Modifier

.align(Alignment.CenterVertically)
                            .weight(1f)
                    )
                    Spacer(Modifier.width(20.dp))
                    Text(
                        text                        =
rider.year.toString(),
                        style                       =
MaterialTheme.typography.labelMedium,
                        fontSize = 20.sp,
                        color                       =
MaterialTheme.colorScheme.onSurface,
                        modifier                    =
Modifier.align(Alignment.CenterVertically)
                    )
                }

                Spacer(Modifier.height(15.dp))
                Text(
                    text                            =
stringResource(R.string.description),
                    style                           =
MaterialTheme.typography.labelMedium,
                    fontSize = 20.sp,
                    color                           =
MaterialTheme.colorScheme.onSurface
                )
                Spacer(Modifier.height(5.dp))
                Text(
                    text = rider.description,
                    style                           =
MaterialTheme.typography.bodyMedium,
                    fontSize = 18.sp,
                    color                           =
MaterialTheme.colorScheme.onSurface,
                    textAlign = TextAlign.Justify
                )
                Spacer(Modifier.height(50.dp))

                Row (
                    Modifier.fillMaxWidth(),
                    horizontalArrangement           =
Arrangement.End
                ) {
                    Button(
```

```
                            shape                           =
RoundedCornerShape(16.dp),
                            contentPadding                  =
PaddingValues(vertical = 15.dp, horizontal = 20.dp),
                            onClick = {
                                val       intent       =
Intent(Intent.ACTION_VIEW)
                                intent.data             =
Uri.parse(rider.imdbUrl)

intent.setPackage("com.android.chrome")

context.startActivity(intent)
                            }
                        ) {

Text(stringResource(R.string.imdb), fontSize = 18.sp)
                        }
                        Spacer(Modifier.width(10.dp))
                        Button(
                            shape                           =
RoundedCornerShape(16.dp),
                            contentPadding                  =
PaddingValues(vertical = 15.dp, horizontal = 20.dp),
                            onClick = {

navController.navigate(Routes.detailScreen+"/${rider
.id}")
                            }
                        ) {

Text(stringResource(R.string.detail),   fontSize   =
18.sp)
                        }
                    }
                }
            }
        }
    }
}
```

ListScreen.kt

Tabel 3. Source Code Jetpack Compose ListScreen.kt Soal 1

```
package com.example.scrollablecompose.screens

import
androidx.compose.foundation.layout.PaddingValues
```

```
import
androidx.compose.foundation.layout.fillMaxWidth
import androidx.compose.foundation.layout.padding
import androidx.compose.foundation.lazy.LazyColumn
import androidx.compose.foundation.lazy.items
import
androidx.compose.material3.ExperimentalMaterial3Api
import androidx.compose.material3.MaterialTheme
import androidx.compose.material3.Scaffold
import androidx.compose.material3.Text
import androidx.compose.material3.TopAppBar
import androidx.compose.material3.TopAppBarDefaults
import androidx.compose.runtime.Composable
import androidx.compose.runtime.getValue
import androidx.compose.runtime.mutableIntStateOf
import androidx.compose.runtime.remember
import androidx.compose.ui.Modifier
import
androidx.compose.ui.platform.LocalConfiguration
import androidx.compose.ui.res.stringResource
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
import androidx.navigation.NavController
import
androidx.navigation.compose.rememberNavController
import com.example.scrollablecompose.R
import
com.example.scrollablecompose.data.getKamenRiderList
import
com.example.scrollablecompose.ui.theme.ScrollableCom
poseTheme

@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun ListScreen(navController: NavController) {
    val config = LocalConfiguration.current
    val     orientationMode     by     remember     {
mutableIntStateOf(config.orientation) }

    Scaffold(
        topBar = {
            TopAppBar(
                title = {

Text(stringResource(R.string.topappbarr_title))
                },
```

```
                colors                                  =
TopAppBarDefaults.topAppBarColors(
                    containerColor                      =
MaterialTheme.colorScheme.primary,
                    titleContentColor                   =
MaterialTheme.colorScheme.onPrimary
                )
            )
        }
    ) { innerPadding ->
        LazyColumn(
            modifier = Modifier
                .fillMaxWidth()
                .padding(innerPadding),
            contentPadding = PaddingValues(16.dp)
        ) {
            items(getKamenRiderList()) { rider ->
                RiderCard(rider,      navController,
orientationMode)
            }
        }
    }
}

@Preview(
    showBackground = true,
    name = "Redmi Note 13",
    widthDp = 390,
    heightDp = 800
)
@Composable
fun ListScreenPortraitPreview() {
    ScrollableComposeTheme {
        val navController = rememberNavController()
        ListScreen(navController = navController)
    }
}

@Preview(
    showBackground = true,
    name = "Redmi Note 13",
    widthDp = 800,
    heightDp = 390
)
@Composable
fun ListScreenLandscapePreview() {
    ScrollableComposeTheme {
```

```
            val navController = rememberNavController()
            ListScreen(navController = navController)
        }
}
```

DetailScreen.kt

Tabel 4. Source Code Jetpack Compose Detail.kt Soal 1

```
package com.example.scrollablecompose.screens

import android.content.res.Configuration
import androidx.compose.foundation.Image
import androidx.compose.foundation.layout.Column
import
androidx.compose.foundation.layout.PaddingValues
import androidx.compose.foundation.layout.Row
import androidx.compose.foundation.layout.Spacer
import
androidx.compose.foundation.layout.fillMaxSize
import
androidx.compose.foundation.layout.fillMaxWidth
import androidx.compose.foundation.layout.height
import androidx.compose.foundation.layout.padding
import androidx.compose.foundation.layout.size
import androidx.compose.foundation.layout.width
import
androidx.compose.foundation.rememberScrollState
import
androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.foundation.verticalScroll
import androidx.compose.material.icons.Icons
import
androidx.compose.material.icons.automirrored.filled.
ArrowBack
import
androidx.compose.material3.ExperimentalMaterial3Api
import androidx.compose.material3.Icon
import androidx.compose.material3.IconButton
import androidx.compose.material3.MaterialTheme
import androidx.compose.material3.Scaffold
import androidx.compose.material3.Text
import androidx.compose.material3.TopAppBar
import androidx.compose.material3.TopAppBarDefaults
import androidx.compose.runtime.Composable
import androidx.compose.runtime.getValue
import androidx.compose.runtime.mutableIntStateOf
import androidx.compose.runtime.remember
```

```
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.draw.clip
import androidx.compose.ui.layout.ContentScale
import
androidx.compose.ui.platform.LocalConfiguration
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.res.stringResource
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import androidx.navigation.NavController
import
androidx.navigation.compose.rememberNavController
import com.example.scrollablecompose.R
import
com.example.scrollablecompose.data.getKamenRiderList
import
com.example.scrollablecompose.model.KamenRider
import
com.example.scrollablecompose.ui.theme.ScrollableCom
poseTheme

@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun DetailScreen(navController: NavController, id:
Int) {
    val rider = getKamenRiderList().find { it.id ==
id }
    val config = LocalConfiguration.current
    val    orientationMode    by    remember    {
mutableIntStateOf(config.orientation) }

    Scaffold (
        topBar = {
            TopAppBar(
                title = {

Text(stringResource(R.string.detail))
                },
                navigationIcon = {
                    IconButton(    onClick    =    {
navController.navigateUp() } ) {
                        Icon(
```

```
                                imageVector             =
Icons.AutoMirrored.Filled.ArrowBack,
                                contentDescription      =
null
                            )
                        }
                    },
                    colors                                  =
TopAppBarDefaults.topAppBarColors(
                        containerColor                      =
MaterialTheme.colorScheme.primary,
                        titleContentColor                   =
MaterialTheme.colorScheme.onPrimary,
                        navigationIconContentColor      =
MaterialTheme.colorScheme.onPrimary
                    )
                )
            }
        ) { padding ->
            if (rider != null) {
                if(orientationMode                          ==
Configuration.ORIENTATION_PORTRAIT) {
                    DetailPortraitLayout(rider, padding)
                }
                else{
                    DetailLandscapeLayout(rider,
padding)
                }
            }
            else{
                Text(
                    text = "Data not found",
                    fontSize = 25.sp
                )
            }
        }
}

@Composable
fun    DetailPortraitLayout(rider    :    KamenRider,
paddingValues: PaddingValues) {
    val scrollState = rememberScrollState()

    Column (
        modifier = Modifier
            .padding(paddingValues)
```

```
            .padding(vertical = 15.dp, horizontal =
20.dp)
            .verticalScroll(scrollState)
            .fillMaxSize(),
        horizontalAlignment                          =
Alignment.CenterHorizontally
    ) {
        Image(
            painter                                  =
painterResource(rider.posterRes),
            contentDescription       =       "Poster
${rider.name}",
            contentScale = ContentScale.FillBounds,
            modifier = Modifier
                .size(width   =   300.dp,   height   =
400.dp)
                .clip(RoundedCornerShape(25.dp))
        )
        Spacer(Modifier.height(15.dp))
        Text(
            text = rider.name,
            fontSize = 30.sp,
            fontWeight = FontWeight.Bold,
            textAlign = TextAlign.Center
        )
        Spacer(Modifier.height(20.dp))
        Column (
            modifier = Modifier.fillMaxWidth()
        ) {
            Text(
                text = stringResource(R.string.year,
rider.year),
                fontSize = 18.sp,
                fontWeight = FontWeight.Bold
            )

            Spacer(Modifier.height(15.dp))

            Text(
                text                                 =
stringResource(R.string.description),
                fontSize = 18.sp,
                fontWeight = FontWeight.Bold
            )
            Spacer(Modifier.height(3.dp))
            Text(
                text = rider.description,
```

```
                textAlign = TextAlign.Justify
            )
        }
    }
}

@Composable
fun     DetailLandscapeLayout(rider:     KamenRider,
paddingValues: PaddingValues) {
    val scrollState = rememberScrollState()

    Row (
        modifier = Modifier
            .fillMaxWidth()
            .verticalScroll(scrollState)
            .padding(paddingValues)
            .padding(vertical = 15.dp, horizontal =
20.dp)
    ) {
        Image(
            painter    =    painterResource(id    =
rider.posterRes),
            contentDescription       =       "Poster
${rider.name}",
            modifier = Modifier
                .size(width   =   240.dp,   height   =
300.dp)
                .padding(8.dp)
                .align(Alignment.CenterVertically)
                .clip(RoundedCornerShape(15.dp)),
            contentScale = ContentScale.FillBounds,
        )
        Spacer(Modifier.width(5.dp))
        Column(Modifier.padding(8.dp)) {

            Text(
                text = rider.name,
                fontSize = 35.sp,
                fontWeight = FontWeight.Bold,
            )

            Spacer(Modifier.height(15.dp))
            Text(
                text = stringResource(R.string.year,
rider.year),
                fontSize = 21.sp,
                fontWeight = FontWeight.Bold
```

```
            )

            Spacer(Modifier.height(15.dp))
            Text(
                text                                    =
stringResource(R.string.description),
                fontSize = 21.sp,
                fontWeight = FontWeight.Bold,
                color                                   =
MaterialTheme.colorScheme.onSurface
            )
            Text(
                text = rider.description,
                fontSize = 20.sp,
                color                                   =
MaterialTheme.colorScheme.onSurface,
                textAlign = TextAlign.Justify
            )
        }
    }

}

@Preview(
    showBackground = true,
    widthDp = 390,
    heightDp = 800,
    name = "Redmi Note 13 Portrait"
)
@Composable
fun DetailScreenPortraitPreview() {
    ScrollableComposeTheme {
        val navController = rememberNavController()
        DetailScreen(navController,
getKamenRiderList()[0].id)
    }
}

@Preview(
    showBackground = true,
    widthDp = 800,
    heightDp = 390,
    name = "Redmi Note 13 Landscape"
)
@Composable
fun DetailScreenLandscapePreview() {
    ScrollableComposeTheme {
```

```
        val navController = rememberNavController()
        DetailScreen(navController,
getKamenRiderList()[0].id)
    }
}
```

Routes.kt

Tabel 5. Source Code Jetpack Compose Routes.kt Soal 1

```
package com.example.scrollablecompose

object Routes {
    val listScreen = "list_screen"
    val detailScreen = "detail_screen"
}
```

KamenRider.kt

Tabel 6. Source Code Jetpack Compose KamenRider.kt Soal 1

```
package com.example.scrollablecompose.model

data class KamenRider (
    val id: Int,
    val name: String,
    val description: String,
    val year: Int,
    val imdbUrl: String,
    val posterRes : Int
)
```

KamenRiderList.kt

Tabel 7. Source Code Jetpack Compose KamenRiderList.kt Soal 1

```
package com.example.scrollablecompose.data

import com.example.scrollablecompose.R
import
com.example.scrollablecompose.model.KamenRider

fun getKamenRiderList(): List<KamenRider> {
    return listOf(
        KamenRider(
            id = 1,
            name = "Kamen Rider Geats",
            description = "Para peserta bertarung
dalam Desire Grand Prix untuk menjadi pahlawan ideal
di dunia seperti permainan.",
```

```
            year = 2022,
            imdbUrl                                         =
"https://www.imdb.com/title/tt20758104/?ref_=ext_shr
_lnk",
            posterRes = R.drawable.kr_geats
        ),
        KamenRider(
            id = 2,
            name = "Kamen Rider Build",
            description = "Seorang fisikawan jenius
berubah menjadi Build untuk mengungkap kebenaran di
balik Kotak Pandora.",
            year = 2017,
            imdbUrl                                         =
"https://www.imdb.com/title/tt6982472/?ref_=ext_shr_
lnk",
            posterRes = R.drawable.kr_build
        ),
        KamenRider(
            id = 3,
            name = "Kamen Rider Zero One",
            description = "Di dunia yang dipenuhi AI
dan robot, Aruto bertarung sebagai Zero-One demi
melindungi manusia dan Humagear.",
            year = 2019,
            imdbUrl                                         =
"https://www.imdb.com/title/tt10333650/?ref_=ext_shr
_lnk",
            posterRes = R.drawable.kr_zero_one
        ),
        KamenRider(
            id = 4,
            name = "Kamen Rider W (Double)",
            description = "Dua jiwa dalam satu tubuh,
Shotaro dan Philip bekerja sama sebagai detektif untuk
melindungi Futo dari kejahatan Gaia Memory.",
            year = 2009,
            imdbUrl                                         =
"https://www.imdb.com/title/tt1483620/?ref_=ext_shr_
lnk",
            posterRes = R.drawable.kr_double
        ),
        KamenRider(
            id = 5,
            name = "Kamen Rider Gaim",
```

```
            description = "Para Rider bertarung dalam permainan yang tampak seperti hiburan, namun berubah menjadi konflik serius demi menyelamatkan dunia.",
            year = 2013,
            imdbUrl = "https://www.imdb.com/title/tt3079058/?ref_=ext_shr_lnk",
            posterRes = R.drawable.kr_gaim
        ),
        KamenRider(
            id = 6,
            name = "Kamen Rider Ex-Aid",
            description = "Seorang dokter sekaligus gamer bertarung melawan virus digital Bugster demi menyelamatkan pasien dari penyakit misterius.",
            year = 2016,
            imdbUrl = "https://www.imdb.com/title/tt5813014/?ref_=ext_shr_lnk",
            posterRes = R.drawable.kr_ex_aid
        ),
        KamenRider(
            id = 7,
            name = "Kamen Rider Gotchard",
            year = 2023,
            description = "Rinne dan Houtarou bertarung bersama menggunakan kartu Chemies untuk menjaga keseimbangan dunia.",
            imdbUrl = "https://www.imdb.com/title/tt27830205/?ref_=ext_shr_lnk",
            posterRes = R.drawable.kr_gotchard
        )
    )
}
```

2) Versi XML

MainActivity.kt

Tabel 8. Source Code XML MainActivity.kt

```
1  package com.example.scrollablexml
2
3  import android.os.Bundle
4  import androidx.activity.enableEdgeToEdge
5  import androidx.appcompat.app.AppCompatActivity
6
```

```
7   class MainActivity : AppCompatActivity() {
8       override fun onCreate(savedInstanceState:
    Bundle?) {
9           super.onCreate(savedInstanceState)
10          enableEdgeToEdge()
11          setContentView(R.layout.activity_main)
12      }
13  }
```

MainAdapter.kt

Tabel 9. Source Code XML MainAdapter.kt

```
1   package com.example.scrollablexml
2
3   import android.view.LayoutInflater
4   import android.view.View
5   import android.view.ViewGroup
6   import android.widget.Button
7   import android.widget.ImageView
8   import android.widget.TextView
9   import androidx.recyclerview.widget.RecyclerView
10  import com.example.scrollablexml.model.KamenRider
11
12  class MainAdapter (
13      private val listRider: List<KamenRider>,
14      private val onImdbClick: (String) -> Unit,
15      private val onDetailClick: (Int) -> Unit
16
17  ) : RecyclerView.Adapter<MainAdapter.ViewHolder>() {
18      class ViewHolder(view: View):
19  RecyclerView.ViewHolder(view){
20          val riderImage =
21  view.findViewById<ImageView>(R.id.riderImage)
22          val riderName =
23  view.findViewById<TextView>(R.id.riderName)
24          val riderYear =
25  view.findViewById<TextView>(R.id.riderYear)
26          val riderDesc =
27  view.findViewById<TextView>(R.id.descBody)
28          val imdbBtn =
29  view.findViewById<Button>(R.id.imdbBtn)
30          val detailBtn =
31  view.findViewById<Button>(R.id.detailBtn)
32      }
33
34      override fun onCreateViewHolder(parent:
35  ViewGroup, viewType: Int): ViewHolder {
```

```
36          return
37  ViewHolder(LayoutInflater.from(parent.context).infla
38  te(R.layout.adapter_main, parent, false))
        }

        override fun getItemCount() = listRider.size

        override fun onBindViewHolder(holder: ViewHolder,
    position: Int) {
            val rider = listRider[position]

    holder.riderImage.setImageResource(rider.posterRes)
            holder.riderName.text = rider.name
            holder.riderYear.text = "${rider.year}"
            holder.riderDesc.text = rider.description
            holder.imdbBtn.setOnClickListener          {
    onImdbClick(rider.imdbUrl) }
            holder.detailBtn.setOnClickListener        {
    onDetailClick(rider.id) }
        }
    }
```

ListFragment.kt

Tabel 10. Source Code XML ListFragment.kt

```
1   package com.example.scrollablexml
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
11  androidx.navigation.fragment.findNavController
12  import androidx.recyclerview.widget.RecyclerView
13  import
14  com.example.scrollablexml.data.getKamenRiderList
15
16  // TODO: Rename parameter arguments, choose names that
17  match
18  //  the  fragment  initialization  parameters,  e.g.
19  ARG_ITEM_NUMBER
20  private const val ARG_PARAM1 = "param1"
21  private const val ARG_PARAM2 = "param2"
22
```

```
/**
 * A simple [Fragment] subclass.
 * Use the [ListFragment.newInstance] factory method to
 * create an instance of this fragment.
 */
class ListFragment : Fragment() {
    // TODO: Rename and change types of parameters
    private var param1: String? = null
    private var param2: String? = null

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        arguments?.let {
            param1 = it.getString(ARG_PARAM1)
            param2 = it.getString(ARG_PARAM2)
        }
    }

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // Inflate the layout for this fragment
        val view = inflater.inflate(R.layout.fragment_list, container, false)
        val recyclerView = view.findViewById<RecyclerView>(R.id.recyclerView)

        val mainAdapter = MainAdapter(
            listRider = getKamenRiderList(),
            onDetailClick = { riderId ->
                val action = ListFragmentDirections.actionListFragmentToDetailFragment(riderId)
                findNavController().navigate(action)
            },
            onImdbClick = { imdbUrl ->
                val intent = Intent(Intent.ACTION_VIEW, Uri.parse(imdbUrl)).apply {
                    setPackage("com.android.chrome")
                }
                startActivity(intent)
```

```
70              }
71          )
72
73          recyclerView.adapter = mainAdapter
74          return view
75      }
76
77
78      companion object {
79          /**
80           * Use this factory method to create a new
81  instance of
82           *   this   fragment   using   the   provided
    parameters.
           *
           * @param param1 Parameter 1.
           * @param param2 Parameter 2.
           *  @return  A  new  instance  of  fragment
    listFragment.
           */
          // TODO: Rename and change types and number
    of parameters
          @JvmStatic
          fun   newInstance(param1:   String,   param2:
    String) =
              ListFragment().apply {
                  arguments = Bundle().apply {
                      putString(ARG_PARAM1, param1)
                      putString(ARG_PARAM2, param2)
                  }
              }
      }
  }
```

DetailFragment.kt

Tabel 11. Source Code XML DetailFragment.kt

```
1   package com.example.scrollablexml
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 android.widget.TextView
9   import androidx.navigation.fragment.navArgs
10
```

```
import
com.example.scrollablexml.data.getKamenRiderList
import
com.google.android.material.appbar.MaterialToolbar
import
com.google.android.material.imageview.ShapeableImage
View

// TODO: Rename parameter arguments, choose names that
match
//  the  fragment  initialization  parameters,  e.g.
ARG_ITEM_NUMBER
private const val ARG_PARAM1 = "param1"
private const val ARG_PARAM2 = "param2"

/**
 * A simple [Fragment] subclass.
 *  Use  the  [DetailFragment.newInstance]  factory
method to
 * create an instance of this fragment.
 */
class DetailFragment : Fragment() {
    // TODO: Rename and change types of parameters
    private var param1: String? = null
    private var param2: String? = null
    private val args: DetailFragmentArgs by navArgs()

    override    fun    onCreate(savedInstanceState:
Bundle?) {
        super.onCreate(savedInstanceState)
        arguments?.let {
            param1 = it.getString(ARG_PARAM1)
            param2 = it.getString(ARG_PARAM2)
        }
    }

    override fun onCreateView(
        inflater:    LayoutInflater,    container:
ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // Inflate the layout for this fragment
        val                  view                  =
inflater.inflate(R.layout.fragment_detail,
container, false)
```

```
        val                 toolbar                 =
view.findViewById<MaterialToolbar>(R.id.topAppBar)
        toolbar.setNavigationOnClickListener {

requireActivity().onBackPressedDispatcher.onBackPres
sed()
        }

        val rider = getKamenRiderList().find { it.id
== args.riderId }


view.findViewById<ShapeableImageView>(R.id.riderImag
e).setImageResource(rider?.posterRes ?: 0)

view.findViewById<TextView>(R.id.riderName).text   =
rider?.name

view.findViewById<TextView>(R.id.riderYear).text   =
getString(R.string.detail_year, rider?.year)

view.findViewById<TextView>(R.id.descBody).text    =
rider?.description

        return view
    }

    companion object {
        /**
         * Use this factory method to create a new
instance of
         *   this   fragment   using   the   provided
parameters.
         *
         * @param param1 Parameter 1.
         * @param param2 Parameter 2.
         *  @return  A  new  instance  of  fragment
detailFragment.
         */
        // TODO: Rename and change types and number
of parameters
        @JvmStatic
        fun   newInstance(param1:   String,   param2:
String) =
            DetailFragment().apply {
                arguments = Bundle().apply {
                    putString(ARG_PARAM1, param1)
```

```
                        putString(ARG_PARAM2, param2)
                    }
                }
            }
}
```

KamenRider.kt

Tabel 12. Source Code XML KamenRider.kt

```
1   package com.example.scrollablexml.model
2
3   data class KamenRider(
4       val id: Int,
5       val name: String,
6       val description: String,
7       val year: Int,
8       val imdbUrl: String,
9       val posterRes : Int
10  )
```

KamenRiderList.kt

Tabel 13. Source Code XML KamenRiderList.kt

```
1   package com.example.scrollablexml.data
2
3   import com.example.scrollablexml.R
4   import com.example.scrollablexml.model.KamenRider
5
6   fun getKamenRiderList(): List<KamenRider> {
7       return listOf(
8           KamenRider(
9               id = 1,
10              name = "Kamen Rider Geats",
11              description = "Para peserta bertarung
12  dalam Desire Grand Prix untuk menjadi pahlawan ideal
13  di dunia seperti permainan.",
14              year = 2022,
15              imdbUrl                                 =
16  "https://www.imdb.com/title/tt20758104/?ref_=ext_shr
17  _lnk",
18              posterRes = R.drawable.kr_geats
19          ),
20          KamenRider(
21              id = 2,
22              name = "Kamen Rider Build",
23
24
```

```
            description = "Seorang fisikawan jenius berubah menjadi Build untuk mengungkap kebenaran di balik Kotak Pandora.",
            year = 2017,
            imdbUrl = "https://www.imdb.com/title/tt6982472/?ref_=ext_shr_lnk",
            posterRes = R.drawable.kr_build
        ),
        KamenRider(
            id = 3,
            name = "Kamen Rider Zero One",
            description = "Di dunia yang dipenuhi AI dan robot, Aruto bertarung sebagai Zero-One demi melindungi manusia dan Humagear.",
            year = 2019,
            imdbUrl = "https://www.imdb.com/title/tt10333650/?ref_=ext_shr_lnk",
            posterRes = R.drawable.kr_zero_one
        ),
        KamenRider(
            id = 4,
            name = "Kamen Rider W (Double)",
            description = "Dua jiwa dalam satu tubuh, Shotaro dan Philip bekerja sama sebagai detektif untuk melindungi Futo dari kejahatan Gaia Memory.",
            year = 2009,
            imdbUrl = "https://www.imdb.com/title/tt1483620/?ref_=ext_shr_lnk",
            posterRes = R.drawable.kr_double
        ),
        KamenRider(
            id = 5,
            name = "Kamen Rider Gaim",
            description = "Para Rider bertarung dalam permainan yang tampak seperti hiburan, namun berubah menjadi konflik serius demi menyelamatkan dunia.",
            year = 2013,
            imdbUrl = "https://www.imdb.com/title/tt3079058/?ref_=ext_shr_lnk",
            posterRes = R.drawable.kr_gaim
        ),
        KamenRider(
            id = 6,
```

```
            name = "Kamen Rider Ex-Aid",
            description = "Seorang dokter sekaligus gamer bertarung melawan virus digital Bugster demi menyelamatkan pasien dari penyakit misterius.",
            year = 2016,
            imdbUrl = "https://www.imdb.com/title/tt5813014/?ref_=ext_shr_lnk",
            posterRes = R.drawable.kr_ex_aid
        ),
        KamenRider(
            id = 7,
            name = "Kamen Rider Gotchard",
            year = 2023,
            description = "Rinne dan Houtarou bertarung bersama menggunakan kartu Chemies untuk menjaga keseimbangan dunia.",
            imdbUrl = "https://www.imdb.com/title/tt27830205/?ref_=ext_shr_lnk",
            posterRes = R.drawable.kr_gotchard
        )
    )
}
```

activity_main.xml

Tabel 14. Source Code XML activity_main.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout
3
4  xmlns:android="http://schemas.android.com/apk/res/an
5  droid"
6      xmlns:app="http://schemas.android.com/apk/res-
7  auto"
8      xmlns:tools="http://schemas.android.com/tools"
9      android:id="@+id/main"
10     android:fitsSystemWindows="true"
11     android:layout_width="match_parent"
12     android:layout_height="match_parent"
13     tools:context=".MainActivity">
14
15     <androidx.fragment.app.FragmentContainerView
16         android:id="@+id/fragmentContainerView6"
17
18 android:name="androidx.navigation.fragment.NavHostFr
19 agment"
```

```
        android:layout_width="0dp"
        android:layout_height="0dp"
        app:defaultNavHost="true"
        app:navGraph="@navigation/main_graph"
        app:layout_constraintTop_toTopOf="parent"

app:layout_constraintBottom_toBottomOf="parent"

app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"/>
</androidx.constraintlayout.widget.ConstraintLayout>
```

adapter_main.xml

Tabel 15. Source Code XML adapter_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/an
droid"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    xmlns:app="http://schemas.android.com/apk/res-
auto"
    xmlns:tools="http://schemas.android.com/tools">

    <androidx.cardview.widget.CardView
        android:id="@+id/cardView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:layout_constraintTop_toTopOf="parent"

app:layout_constraintBottom_toBottomOf="parent"

app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        android:layout_margin="16dp"

app:cardBackgroundColor="?android:attr/colorBackgrou
nd"
        app:cardElevation="4dp"
        app:cardCornerRadius="15dp">


<androidx.constraintlayout.widget.ConstraintLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:padding="15dp">
```

```
33
34
35  <com.google.android.material.imageview.ShapeableImag
36  eView
37                  android:id="@+id/riderImage"
38                  android:layout_width="120dp"
39                  android:layout_height="150dp"
40                  android:scaleType="centerCrop"
41                  android:src="@drawable/kr_geats"
42
43  app:layout_constraintBottom_toBottomOf="parent"
44
45  app:layout_constraintStart_toStartOf="parent"
46
47  app:layout_constraintTop_toTopOf="parent"
48
49  app:shapeAppearanceOverlay="@style/RoundedImageView"
50  />
51
52
53
54  <androidx.constraintlayout.widget.ConstraintLayout
55                  android:layout_width="0dp"
56                  android:layout_height="match_parent"
57
58  app:layout_constraintTop_toTopOf="parent"
59
60  app:layout_constraintEnd_toEndOf="parent"
61
62  app:layout_constraintStart_toEndOf="@id/riderImage"
63                  android:layout_marginStart="15dp">
64
65
66  <androidx.constraintlayout.widget.ConstraintLayout
67
68  android:id="@+id/titleYearLayout"
69                      android:layout_width="0dp"
70
71  android:layout_height="wrap_content"
72
73  app:layout_constraintStart_toStartOf="parent"
74
75  app:layout_constraintEnd_toEndOf="parent"
76
77  app:layout_constraintTop_toTopOf="parent">
78
79                      <TextView
```

```
                        android:id="@+id/riderName"
                        android:layout_width="0dp"

android:layout_height="wrap_content"

app:layout_constraintTop_toTopOf="parent"

app:layout_constraintBottom_toBottomOf="parent"

app:layout_constraintStart_toStartOf="parent"

app:layout_constraintEnd_toStartOf="@id/riderYear"

app:layout_constraintHorizontal_weight="6"
                        android:textSize="20sp"
                        tools:text="Kamen      Rider
Geats"
                        android:textStyle="bold" />

                    <TextView
                        android:id="@+id/riderYear"
                        android:layout_width="0dp"

android:layout_height="wrap_content"

app:layout_constraintTop_toTopOf="parent"

app:layout_constraintBottom_toBottomOf="parent"

app:layout_constraintStart_toEndOf="@id/riderName"

app:layout_constraintEnd_toEndOf="parent"

app:layout_constraintHorizontal_weight="2"
                        android:gravity="end"
                        android:textSize="16sp"
                        android:textStyle="bold"
                        tools:text="2022" />


</androidx.constraintlayout.widget.ConstraintLayout>

                <TextView
                    android:id="@+id/descTitle"

android:layout_width="wrap_content"

```

```
127
128 android:layout_height="wrap_content"
129
130 app:layout_constraintTop_toBottomOf="@id/titleYearLa
131 yout"
132
133 app:layout_constraintStart_toStartOf="parent"
134                     android:layout_marginTop="8dp"
135                     android:text="@string/desc"
136                     android:textStyle="bold" />
137
138                 <TextView
139                     android:id="@+id/descBody"
140                     android:layout_width="0dp"
141
142 android:layout_height="wrap_content"
143
144 app:layout_constraintTop_toBottomOf="@id/descTitle"
145
    app:layout_constraintStart_toStartOf="parent"

    app:layout_constraintEnd_toEndOf="parent"
                     android:layout_marginTop="3dp"

    android:justificationMode="inter_word"
                     tools:text="Deskripsi    panjang
    lorem ipsum lorem ipsum lorem ipsum lorem ipsum lorem
    ipsum" />


    <androidx.constraintlayout.widget.ConstraintLayout
                     android:layout_width="0dp"

    android:layout_height="wrap_content"

    app:layout_constraintTop_toBottomOf="@id/descBody"

    app:layout_constraintStart_toStartOf="parent"

    app:layout_constraintEnd_toEndOf="parent"
                     android:layout_marginTop="15dp">

                     <Button
                         android:id="@+id/imdbBtn"
                         android:layout_width="80dp"

    android:layout_height="wrap_content"
```

```
android:layout_marginEnd="5dp"
                        android:textSize="12sp"
                        app:cornerRadius="15dp"

app:layout_constraintBottom_toBottomOf="parent"

app:layout_constraintEnd_toStartOf="@id/detailBtn"

app:layout_constraintTop_toTopOf="parent"
                        android:text="@string/imdb"
/>

                    <Button
                        android:id="@+id/detailBtn"
                        android:layout_width="80dp"

android:layout_height="wrap_content"
                        android:textSize="12sp"
                        app:cornerRadius="15dp"

app:layout_constraintBottom_toBottomOf="parent"

app:layout_constraintEnd_toEndOf="parent"

app:layout_constraintTop_toTopOf="parent"

android:text="@string/detail" />


</androidx.constraintlayout.widget.ConstraintLayout>


</androidx.constraintlayout.widget.ConstraintLayout>



</androidx.constraintlayout.widget.ConstraintLayout>


    </androidx.cardview.widget.CardView>

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

fragment_list.xml

Tabel 16. Source Code XML fragment_list.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout

xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ListFragment">

    <com.google.android.material.appbar.AppBarLayout
        android:id="@+id/topAppBar"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:background="?attr/colorPrimary"
        app:layout_constraintTop_toTopOf="parent"

app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent">


<com.google.android.material.appbar.MaterialToolbar
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            app:title="My Favorite Kamen Rider"

app:titleTextColor="?attr/colorOnPrimary"/>

</com.google.android.material.appbar.AppBarLayout>

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerView"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:padding="10dp"
        tools:listitem="@layout/adapter_main"
        tools:itemCount="7"

app:layoutManager="androidx.recyclerview.widget.LinearLayoutManager"

app:layout_constraintTop_toBottomOf="@id/topAppBar"

app:layout_constraintBottom_toBottomOf="parent"
```

```
app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"/>
</androidx.constraintlayout.widget.ConstraintLayout>
```

fragment_detail.xml

Tabel 17. Source Code XML fragment_detail.xml

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <androidx.constraintlayout.widget.ConstraintLayout
3
4   xmlns:android="http://schemas.android.com/apk/res/an
5   droid"
6       xmlns:app="http://schemas.android.com/apk/res-
7   auto"
8       xmlns:tools="http://schemas.android.com/tools"
9       android:layout_width="match_parent"
10      android:layout_height="match_parent"
11      tools:context=".DetailFragment">
12
13      <com.google.android.material.appbar.AppBarLayout
14          android:id="@+id/topAppBarLayout"
15          android:layout_width="0dp"
16          android:layout_height="wrap_content"
17          android:background="?attr/colorPrimary"
18          app:layout_constraintTop_toTopOf="parent"
19
20  app:layout_constraintStart_toStartOf="parent"
21          app:layout_constraintEnd_toEndOf="parent">
22
23
24  <com.google.android.material.appbar.MaterialToolbar
25              android:id="@+id/topAppBar"
26              android:layout_width="match_parent"
27              android:layout_height="match_parent"
28              app:title="Detail"
29
30  app:titleTextColor="?attr/colorOnPrimary"
31
32  app:navigationIcon="@drawable/baseline_arrow_back_24
33  "
34
35  app:navigationIconTint="?attr/colorOnPrimary"/>
36
37  </com.google.android.material.appbar.AppBarLayout>
38
39      <ScrollView
```

```
        android:layout_width="0dp"
        android:layout_height="0dp"

app:layout_constraintTop_toBottomOf="@id/topAppBarLa
yout"

app:layout_constraintBottom_toBottomOf="parent"

app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent">


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

app:layout_constraintTop_toTopOf="parent"

app:layout_constraintStart_toStartOf="parent"

app:layout_constraintEnd_toEndOf="parent">


<com.google.android.material.imageview.ShapeableImag
eView
                android:id="@+id/riderImage"
                android:layout_width="300dp"
                android:layout_height="400dp"
                android:scaleType="centerCrop"

app:shapeAppearanceOverlay="@style/RoundedImageView"

app:layout_constraintTop_toTopOf="parent"

app:layout_constraintStart_toStartOf="parent"

app:layout_constraintEnd_toEndOf="parent"
                android:src="@drawable/kr_geats"/>

            <TextView
                android:id="@+id/riderName"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginTop="15dp"

app:layout_constraintTop_toBottomOf="@id/riderImage"
```

```
87
88  app:layout_constraintStart_toStartOf="parent"
89
90  app:layout_constraintEnd_toEndOf="parent"
91                  tools:text="Kamen Rider Gotchard"
92                  android:gravity="center_horizontal"
93                  android:textSize="35sp"
94                  android:textStyle="bold"/>
95
96              <TextView
97                  android:id="@+id/riderYear"
98                  android:layout_width="0dp"
99                  android:layout_height="wrap_content"
100                 android:layout_marginTop="20dp"
101
102 app:layout_constraintTop_toBottomOf="@id/riderName"
103
104 app:layout_constraintStart_toStartOf="parent"
105
106 app:layout_constraintEnd_toEndOf="parent"
107                 tools:text="Tahun: 2022"
108                 android:textSize="22sp"
109                 android:textStyle="bold"/>
110
              <TextView
                  android:id="@+id/descTitle"
                  android:layout_width="match_parent"
                  android:layout_height="wrap_content"
                  android:layout_marginTop="15dp"

    app:layout_constraintTop_toBottomOf="@id/riderYear"

    app:layout_constraintStart_toStartOf="parent"

    app:layout_constraintEnd_toEndOf="parent"
                  android:text="@string/desc"
                  android:textSize="22sp"
                  android:textStyle="bold"/>

              <TextView
                  android:id="@+id/descBody"
                  android:layout_width="match_parent"
                  android:layout_height="wrap_content"
                  android:layout_marginTop="7dp"

    app:layout_constraintTop_toBottomOf="@id/descTitle"
```

```
app:layout_constraintStart_toStartOf="parent"

app:layout_constraintEnd_toEndOf="parent"
                tools:text="Deskripsi  panjang  lorem
ipsum lorem ipsum lorem ipsum lorem ipsum lorem ipsum"
                android:textSize="20sp"

android:justificationMode="inter_word"/>


</androidx.constraintlayout.widget.ConstraintLayout>

    </ScrollView>


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

fragment_detail.xml (landscape)

Tabel 18. Source Code XML fragment_detail.xml (landscape)

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <androidx.constraintlayout.widget.ConstraintLayout
3
4   xmlns:android="http://schemas.android.com/apk/res/an
5   droid"
6       xmlns:app="http://schemas.android.com/apk/res-
7   auto"
8       xmlns:tools="http://schemas.android.com/tools"
9       android:layout_width="match_parent"
10      android:layout_height="match_parent"
11      tools:context=".DetailFragment">
12
13      <com.google.android.material.appbar.AppBarLayout
14          android:id="@+id/topAppBarLayout"
15          android:layout_width="0dp"
16          android:layout_height="wrap_content"
17          android:background="?attr/colorPrimary"
18          app:layout_constraintTop_toTopOf="parent"
19
20  app:layout_constraintStart_toStartOf="parent"
21          app:layout_constraintEnd_toEndOf="parent">
22
23
24  <com.google.android.material.appbar.MaterialToolbar
25              android:id="@+id/topAppBar"
26              android:layout_width="match_parent"
```

```
            android:layout_height="match_parent"
            app:title="Detail"

app:titleTextColor="?attr/colorOnPrimary"

app:navigationIcon="@drawable/baseline_arrow_back_24
"

app:navigationIconTint="?attr/colorOnPrimary"/>

</com.google.android.material.appbar.AppBarLayout>

    <ScrollView
        android:layout_width="0dp"
        android:layout_height="0dp"

app:layout_constraintTop_toBottomOf="@id/topAppBarLa
yout"

app:layout_constraintBottom_toBottomOf="parent"

app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent">


<androidx.constraintlayout.widget.ConstraintLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:padding="25dp"

app:layout_constraintTop_toTopOf="parent"

app:layout_constraintStart_toStartOf="parent"

app:layout_constraintEnd_toEndOf="parent">


<com.google.android.material.imageview.ShapeableImag
eView
                android:id="@+id/riderImage"
                android:layout_width="240dp"
                android:layout_height="300dp"
                android:scaleType="centerCrop"

app:shapeAppearanceOverlay="@style/RoundedImageView"

app:layout_constraintTop_toTopOf="parent"
```

```
74
75  app:layout_constraintBottom_toBottomOf="parent"
76
77  app:layout_constraintStart_toStartOf="parent"
78
79  app:layout_constraintEnd_toStartOf="@id/infoText"
80                  android:src="@drawable/kr_geats"/>
81
82
83  <androidx.constraintlayout.widget.ConstraintLayout
84                  android:id="@+id/infoText"
85                  android:layout_width="0dp"
86                  android:layout_height="wrap_content"
87                  android:layout_marginStart="20dp"
88
89  app:layout_constraintTop_toTopOf="parent"
90
91  app:layout_constraintStart_toEndOf="@id/riderImage"
92
93  app:layout_constraintEnd_toEndOf="parent">
94
95                  <TextView
96                      android:id="@+id/riderName"
97
98  android:layout_width="wrap_content"
99
100 android:layout_height="wrap_content"
101
102 app:layout_constraintTop_toTopOf="parent"
103
104 app:layout_constraintStart_toStartOf="parent"
105                     tools:text="Kamen Rider Geats"
106                     android:textSize="40sp"
107                     android:textStyle="bold"/>
108
109                 <TextView
110                     android:id="@+id/riderYear"
111                     android:layout_width="0dp"
112
113 android:layout_height="wrap_content"
114                     android:layout_marginTop="15dp"
115
116 app:layout_constraintTop_toBottomOf="@id/riderName"
117
118 app:layout_constraintStart_toStartOf="parent"
119
120 app:layout_constraintEnd_toEndOf="parent"
```

```
                    tools:text="Tahun: 2022"
                    android:textSize="25sp"
                    android:textStyle="bold"/>

                <TextView
                    android:id="@+id/descTitle"

android:layout_width="match_parent"

android:layout_height="wrap_content"
                    android:layout_marginTop="10dp"

app:layout_constraintTop_toBottomOf="@id/riderYear"

app:layout_constraintStart_toStartOf="parent"
                    android:text="@string/desc"
                    android:textSize="25sp"
                    android:textStyle="bold"/>

                <TextView
                    android:id="@+id/descBody"

android:layout_width="match_parent"

android:layout_height="wrap_content"
                    android:layout_marginTop="7dp"

app:layout_constraintTop_toBottomOf="@id/descTitle"

app:layout_constraintStart_toStartOf="parent"

app:layout_constraintEnd_toEndOf="parent"
                    tools:text="Deskripsi    panjang
lorem ipsum lorem ipsum lorem ipsum lorem ipsum lorem
ipsum"
                    android:textSize="22sp"

android:justificationMode="inter_word"/>


</androidx.constraintlayout.widget.ConstraintLayout>



</androidx.constraintlayout.widget.ConstraintLayout>

    </ScrollView>
```

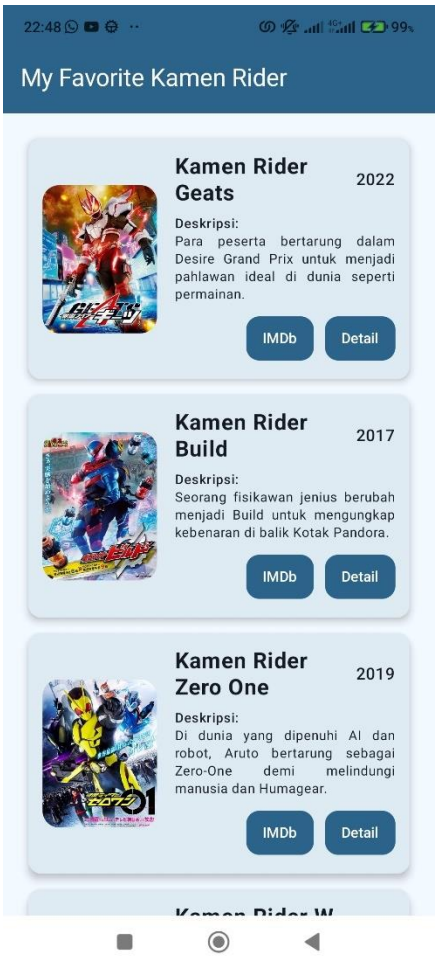
```
</androidx.constraintlayout.widget.ConstraintLayout>
```

main_graph.xml

Tabel 19. Source Code XML main_graph.xml

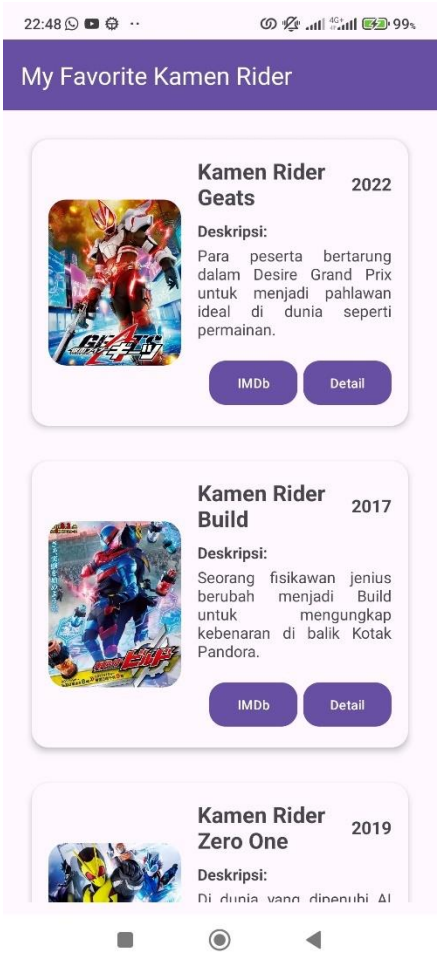
```
1  <?xml version="1.0" encoding="utf-8"?>
2  <navigation
3  xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:app="http://schemas.android.com/apk/res-auto"
5      xmlns:tools="http://schemas.android.com/tools"
       android:id="@+id/main_graph"
6      app:startDestination="@id/listFragment">
7
8      <fragment
9          android:id="@+id/detailFragment"

10 android:name="com.example.scrollablexml.DetailFragment"
11         android:label="fragment_detail"
12         tools:layout="@layout/fragment_detail" >
13         <argument
14             android:name="riderId"
15             app:argType="integer" />
16     </fragment>
17     <fragment
18         android:id="@+id/listFragment"

19 android:name="com.example.scrollablexml.ListFragment"
20         android:label="fragment_list"
21         tools:layout="@layout/fragment_list" >
22         <action

23 android:id="@+id/action_listFragment_to_detailFragment"
24             app:destination="@id/detailFragment" />
25     </fragment>
26 </navigation>
```

B. Output Program



Gambar 3. Screenshot List Screen Soal 1 Versi Jetpack Compose

Gambar 4. Screenshot Detail Screen Soal 1 Versi Jetpack Compose



Gambar 5. Screenshot List Screen Soal 1 Versi XML

Gambar 6. Screenshot Detail Screen Soal 1 Versi XML

C. Pembahasan

1) Versi Jetpack Compose

MainActivity.kt

Pada MainActivity.kt, kode ini membangun aplikasi Android berbasis Jetpack Compose yang menggunakan sistem navigasi antar layar. Di dalam MainActivity, fungsi enableEdgeToEdge() digunakan untuk mengaktifkan tampilan layar penuh, kemudian UI diatur menggunakan setContent yang membungkus aplikasi dengan tema ScrollableComposeTheme. Fungsi MyFavKamenRiderApp() dipanggil untuk mengatur navigasi antar komponen menggunakan NavHost dan rememberNavController. Navigasi dimulai dari ListScreen, dan saat item dipilih, akan diarahkan ke DetailScreen dengan parameter id yang diambil dari rute. Jika id tidak valid, maka nilai default-nya adalah 0.

Card.kt

Pada Card.kt, terdapat komponen UI yang mendefinisikan dua tampilan berbeda berdasarkan orientasi perangkat: portrait dan landscape. Fungsi utama di sini adalah RiderCard, yang menerima objek KamenRider, NavController, dan mode orientasi sebagai parameter. Berdasarkan mode orientasi perangkat, RiderCard akan memanggil salah satu dari dua fungsi komposabel: RiderCardPortrait untuk tampilan portrait dan RiderCardLandscape untuk tampilan landscape.

Fungsi RiderCardPortrait dan RiderCardLandscape keduanya mengimplementasikan sebuah Card yang berisi informasi mengenai Kamen Rider. Pada tampilan portrait, informasi ditampilkan dalam format yang lebih vertikal, dengan gambar poster di sebelah kiri dan teks deskripsi di sebelah kanan. Pada tampilan landscape, pengaturan layout disesuaikan agar lebih horizontal dengan gambar yang lebih besar, serta teks yang lebih luas dan lebih banyak ruang untuk penjelasan.

Kedua tampilan ini memiliki komponen yang sama, seperti nama rider, tahun tayang, deskripsi, dan dua tombol. Tombol pertama mengarahkan pengguna ke halaman IMDb Kamen Rider yang bersangkutan, sementara tombol kedua mengarahkan ke halaman detail menggunakan NavController. Desain ini memastikan pengalaman pengguna yang optimal dengan menyesuaikan tata letak dan ukuran elemen UI berdasarkan orientasi layar perangkat yang digunakan.

ListScreen.kt

File ListScreen.kt berfungsi sebagai tampilan utama aplikasi yang menampilkan daftar karakter Kamen Rider menggunakan LazyColumn. Fungsi ListScreen menerima NavController sebagai parameter untuk mendukung navigasi ke tampilan detail ketika salah satu item dipilih. Dalam komponen ini, LocalConfiguration digunakan untuk mendapatkan konfigurasi layar saat ini, terutama orientasi perangkat (potret atau lanskap) yang disimpan ke dalam variabel orientationMode untuk kemudian diteruskan ke setiap kartu rider agar tampilannya bisa disesuaikan jika dibutuhkan.

Struktur visual utama dibangun dengan Scaffold, yang memiliki TopAppBar sebagai header dan daftar rider sebagai isi. AppBar ini menampilkan judul yang diambil dari stringResource, dan menggunakan skema warna dari MaterialTheme untuk tampilan yang konsisten. Daftar Kamen Rider ditampilkan dalam LazyColumn yang memungkinkan scroll vertikal dan menggunakan items untuk menghasilkan item secara efisien. Masing-masing item menggunakan komponen RiderCard (yang tidak ditampilkan dalam file ini) dan diberikan navController serta informasi orientasi sebagai parameter, memungkinkan setiap kartu untuk merespons interaksi pengguna, seperti klik untuk melihat detail.

Selain itu, dua fungsi @Preview disediakan untuk melihat pratinjau tampilan daftar pada dua mode orientasi layar: potret dan lanskap. Ini membantu pengembang memastikan bahwa tampilan tetap responsif dan nyaman dilihat di berbagai ukuran dan arah layar. Dengan demikian, file ini menjadi pusat navigasi awal dalam aplikasi, menyajikan daftar Kamen Rider secara bersih, adaptif, dan siap untuk berinteraksi.

DetailScreen.kt

Pada DetailScreen.kt, file ini membentuk tampilan detail dari setiap karakter Kamen Rider yang dipilih dari daftar utama. Fungsi utama DetailScreen menerima NavController dan id untuk mencari data KamenRider berdasarkan ID dari daftar yang tersedia. Berdasarkan orientasi perangkat yang dideteksi dengan LocalConfiguration, tampilan akan disesuaikan menjadi potret atau lanskap menggunakan dua layout terpisah: DetailPortraitLayout dan DetailLandscapeLayout.

Jika orientasi perangkat dalam mode potret, layout menampilkan gambar rider dalam ukuran besar di atas, lalu diikuti nama, tahun, dan deskripsi dalam susunan kolom vertikal yang dapat discroll. Sedangkan pada mode lanskap, layout menggunakan Row untuk menyusun gambar di sisi kiri dan informasi di sisi kanan secara horizontal, agar lebih optimal memanfaatkan lebar layar. Gambar menggunakan Image dengan efek rounded corner, dan teks ditata rapi menggunakan Text, Spacer, dan pengaturan font yang menonjolkan judul serta informasi penting.

Navigasi balik ke halaman sebelumnya disediakan oleh TopAppBar dengan ikon panah kembali. Jika data rider tidak ditemukan (misalnya karena ID tidak valid), akan ditampilkan teks "Data not found". Terakhir, disediakan juga dua fungsi @Preview untuk menampilkan pratinjau tampilan secara langsung di Android Studio, baik untuk mode potret maupun lanskap. Dengan struktur ini, aplikasi memberikan pengalaman visual yang fleksibel dan responsif untuk berbagai orientasi layar.

Routes.kt

Pada Routes.kt, didefinisikan sebuah objek Routes yang berfungsi untuk menyimpan konstanta yang digunakan dalam navigasi antar layar pada aplikasi. Objek ini menyimpan dua properti: listScreen dan detailScreen, yang masing-masing mewakili rute untuk layar daftar dan layar detail dalam aplikasi. Rute listScreen bertujuan untuk menavigasi ke layar yang menampilkan daftar item, sementara detailScreen digunakan untuk menavigasi ke layar yang menampilkan detail dari item yang dipilih. Dengan mendefinisikan rute-rute ini dalam satu tempat, aplikasi menjadi lebih terorganisir dan memudahkan pengelolaan navigasi, karena jika ada perubahan dalam nama rute, hanya perlu mengubahnya di file Routes.kt saja tanpa harus mencarinya di seluruh proyek. Hal

ini juga membuat kode lebih mudah dipelihara dan mengurangi kemungkinan kesalahan saat menggunakan rute dalam aplikasi.

KamenRider.kt

Pada KamenRider.kt, didefinisikan sebuah kelas data KamenRider yang digunakan untuk merepresentasikan data mengenai karakter Kamen Rider dalam aplikasi. Kelas ini memiliki beberapa properti yang menggambarkan informasi penting tentang Kamen Rider, yaitu: id (tipe data Int) yang berfungsi sebagai identifier unik untuk setiap Kamen Rider, name (tipe data String) yang menyimpan nama Kamen Rider, description (tipe data String) yang menyimpan deskripsi tentang karakter tersebut, year (tipe data Int) yang menyimpan tahun debut Kamen Rider, imdbUrl (tipe data String) yang menyimpan URL ke halaman IMDb Kamen Rider, dan posterRes (tipe data Int) yang menyimpan referensi ke sumber daya gambar poster Kamen Rider. Kelas data ini memungkinkan pemrograman yang lebih efisien karena secara otomatis menyediakan metode untuk membandingkan, menyalin, dan mendeskripsikan objek KamenRider. Dengan menggunakan kelas ini, aplikasi dapat menyimpan dan menampilkan data terkait Kamen Rider dengan struktur yang jelas dan mudah diakses.

KamenRiderList.kt

Pada KamenRiderList.kt, terdapat sebuah fungsi bernama getKamenRiderList() yang mengembalikan daftar (list) objek KamenRider. Fungsi ini berfungsi untuk menyediakan data statis mengenai berbagai karakter Kamen Rider yang ada dalam aplikasi. Setiap objek KamenRider yang dimasukkan dalam daftar berisi informasi tentang id, name, description, year, imdbUrl, dan posterRes. id adalah identifier unik untuk setiap karakter, name berisi nama Kamen Rider, description menjelaskan latar belakang cerita karakter tersebut, year menunjukkan tahun debut Kamen Rider, imdbUrl adalah link menuju halaman IMDb karakter tersebut, dan posterRes merujuk pada sumber daya gambar poster Kamen Rider yang disertakan dalam aplikasi. Fungsi ini menyusun daftar Kamen Rider yang berbeda, seperti Kamen Rider Geats, Kamen Rider Build, dan lainnya, yang masing-masing memiliki deskripsi dan tahun tayang yang unik. Dengan menggunakan fungsi ini, aplikasi dapat menampilkan daftar Kamen Rider yang terstruktur dengan baik, lengkap dengan gambar dan informasi yang relevan.

2) Versi XML

MainActivity.kt:

Pada MainActivity.kt, terdapat sebuah MainActivity yang merupakan kelas utama untuk aplikasi Android. Kelas ini mewarisi AppCompatActivity, yang merupakan kelas dasar untuk aktivitas yang mendukung fitur-fitur kompatibilitas dengan versi Android lebih lama. Pada metode onCreate(), yang dipanggil saat aktivitas pertama kali dibuat, dipanggil metode super.onCreate(savedInstanceState) untuk memastikan siklus hidup aktivitas berjalan dengan benar. Selanjutnya, metode enableEdgeToEdge() dipanggil untuk memungkinkan tampilan aplikasi menutupi area layar secara penuh, termasuk area tepi (edge-to-edge), memberikan pengalaman visual yang lebih imersif. Setelah itu, setContentView(R.layout.activity_main) digunakan untuk menetapkan tampilan antarmuka pengguna dengan menghubungkan layout XML yang berada dalam file activity_main.xml ke aktivitas ini. Dengan begitu, aplikasi akan menampilkan UI sesuai dengan desain yang didefinisikan dalam XML tersebut saat aktivitas ini dimulai.

MainAdapter.kt

Pada MainAdapter.kt, terdapat sebuah kelas MainAdapter yang merupakan adapter untuk RecyclerView, digunakan untuk menampilkan daftar karakter Kamen Rider dalam tampilan daftar (list) di aplikasi. MainAdapter menerima tiga parameter: listRider yang berisi daftar objek KamenRider, serta dua lambda function onImdbClick dan onDetailClick yang digunakan untuk menangani aksi ketika tombol IMDb dan Detail diklik. Adapter ini memiliki kelas internal ViewHolder yang bertanggung jawab untuk mengikat data ke tampilan individual item. Dalam metode bind(), data dari objek KamenRider seperti gambar, nama, tahun, dan deskripsi ditampilkan di tampilan item menggunakan AdapterMainBinding, yang memetakan elemen UI di XML ke objek di kode. Tombol IMDb dan Detail masing-masing diberi listener untuk menanggapi klik, yang akan menjalankan lambda function yang diberikan. Metode onCreateViewHolder() digunakan untuk membuat tampilan item baru dengan AdapterMainBinding, sementara onBindViewHolder() mengikat data ke tampilan untuk setiap posisi dalam daftar. Metode getItemCount() mengembalikan jumlah item dalam listRider. Adapter ini memudahkan pengelolaan dan penampilan data dalam format yang terstruktur di RecyclerView.

ListFragment.kt

Pada ListFragment.kt, terdapat sebuah fragment yang bertugas untuk menampilkan daftar Kamen Rider dalam bentuk RecyclerView. Di dalam fragment ini, terdapat binding dengan layout FragmentListBinding yang memungkinkan pengikatan elemen UI ke dalam kode. Pada onCreateView(), fragment ini menginisialisasi MainAdapter yang digunakan untuk menampilkan daftar KamenRider dengan memanfaatkan data yang diperoleh dari fungsi getKamenRiderList(). Adapter ini menerima dua fungsi lambda, yaitu onDetailClick yang akan menavigasi ke fragment detail menggunakan

NavController, dan onImdbClick yang membuka URL IMDb di aplikasi browser menggunakan Intent. Ketika item di daftar dipilih, aksi untuk menavigasi ke fragment lain dilakukan dengan ListFragmentDirections.actionListFragmentToDetailFragment(riderId). Adapter kemudian dihubungkan dengan RecyclerView melalui binding, dan data ditampilkan sesuai dengan desain. onDestroyView() digunakan untuk membersihkan referensi binding setelah tampilan fragment dihancurkan. Selain itu, terdapat metode newInstance() yang memungkinkan pembuatan instance fragment dengan argumen tertentu. Fragment ini bertanggung jawab untuk menampilkan daftar Kamen Rider dan menangani interaksi pengguna seperti navigasi dan membuka IMDb.

DetailFragment.kt

Pada DetailFragment.kt, fragment ini bertugas untuk menampilkan detail informasi mengenai Kamen Rider yang dipilih pengguna dari daftar. Fragment ini menggunakan ViewBinding dengan mengikat elemen UI dari layout FragmentDetailBinding. Pada onCreateView(), fragment mendapatkan parameter riderId yang diteruskan melalui navArgs(), yang digunakan untuk mencari data Kamen Rider sesuai dengan ID yang diberikan. Fungsi getKamenRiderList() digunakan untuk memperoleh daftar Kamen Rider, dan kemudian mencari item yang cocok dengan ID tersebut. Jika data ditemukan, informasi seperti poster, nama, tahun, dan deskripsi Kamen Rider ditampilkan pada tampilan menggunakan binding. Selain itu, setNavigationOnClickListener pada topAppBar digunakan untuk menangani aksi ketika pengguna menekan tombol kembali di bagian atas aplikasi, yang akan memicu aksi onBackPressed() untuk kembali ke halaman sebelumnya. Pada onDestroyView(), referensi _binding dibersihkan untuk mencegah memory leak setelah tampilan fragment dihancurkan. Fragment ini memastikan pengguna dapat melihat informasi detail mengenai Kamen Rider yang dipilih.

KamenRider.kt

Pada KamenRider.kt, didefinisikan sebuah kelas data KamenRider yang digunakan untuk merepresentasikan data mengenai karakter Kamen Rider dalam aplikasi. Kelas ini memiliki beberapa properti yang menggambarkan informasi penting tentang Kamen Rider, yaitu: id (tipe data Int) yang berfungsi sebagai identifier unik untuk setiap Kamen Rider, name (tipe data String) yang menyimpan nama Kamen Rider, description (tipe data String) yang menyimpan deskripsi tentang karakter tersebut, year (tipe data Int) yang menyimpan tahun debut Kamen Rider, imdbUrl (tipe data String) yang menyimpan URL ke halaman IMDb Kamen Rider, dan posterRes (tipe data Int) yang menyimpan referensi ke sumber daya gambar poster Kamen Rider. Kelas data ini memungkinkan

pemrograman yang lebih efisien karena secara otomatis menyediakan metode untuk membandingkan, menyalin, dan mendeskripsikan objek KamenRider. Dengan menggunakan kelas ini, aplikasi dapat menyimpan dan menampilkan data terkait Kamen Rider dengan struktur yang jelas dan mudah diakses.

KamenRiderList.kt

Pada KamenRiderList.kt, terdapat sebuah fungsi bernama getKamenRiderList() yang mengembalikan daftar (list) objek KamenRider. Fungsi ini berfungsi untuk menyediakan data statis mengenai berbagai karakter Kamen Rider yang ada dalam aplikasi. Setiap objek KamenRider yang dimasukkan dalam daftar berisi informasi tentang id, name, description, year, imdbUrl, dan posterRes. id adalah identifier unik untuk setiap karakter, name berisi nama Kamen Rider, description menjelaskan latar belakang cerita karakter tersebut, year menunjukkan tahun debut Kamen Rider, imdbUrl adalah link menuju halaman IMDb karakter tersebut, dan posterRes merujuk pada sumber daya gambar poster Kamen Rider yang disertakan dalam aplikasi. Fungsi ini menyusun daftar Kamen Rider yang berbeda, seperti Kamen Rider Geats, Kamen Rider Build, dan lainnya, yang masing-masing memiliki deskripsi dan tahun tayang yang unik. Dengan menggunakan fungsi ini, aplikasi dapat menampilkan daftar Kamen Rider yang terstruktur dengan baik, lengkap dengan gambar dan informasi yang relevan.

activity_main.xml:

Pada activity_main.xml, layout ini menggunakan ConstraintLayout sebagai root view untuk mendefinisikan tampilan aplikasi. ConstraintLayout memberikan fleksibilitas dalam menentukan aturan tata letak dengan menggunakan constraint pada posisi elemen-elemen di dalamnya. Di dalam layout ini, terdapat FragmentContainerView yang digunakan untuk menampilkan fragment dengan ID fragmentContainerView6. Elemen ini berfungsi sebagai wadah untuk NavHostFragment, yang berperan sebagai pengontrol navigasi antar fragment dalam aplikasi. Dengan menetapkan atribut app:navGraph ke @navigation/main_graph, NavHostFragment dihubungkan dengan file graf navigasi main_graph, yang mendefinisikan jalur navigasi antara fragment. Atribut app:defaultNavHost="true" menunjukkan bahwa NavHostFragment akan menangani navigasi kembali secara default, menjadikannya sebagai fragment utama yang dapat mengelola transisi antar fragment. Layout ini memastikan bahwa FragmentContainerView mengisi seluruh ruang layar, dengan mengikatnya ke sisi atas, bawah, kiri, dan kanan dari parent container menggunakan constraints. Selain itu, atribut android:fitsSystemWindows="true" memastikan bahwa layout dapat beradaptasi dengan elemen sistem, seperti status bar atau navigasi bar, dengan memberikan ruang yang cukup pada bagian atas dan bawah.

adapter_main.xml

Pada adapter_main.xml, layout ini dirancang untuk menampilkan elemen-elemen informasi mengenai Kamen Rider dalam bentuk yang terstruktur dan menarik, menggunakan ConstraintLayout sebagai root view. Di dalamnya terdapat CardView, yang berfungsi untuk memberikan efek elevasi dan sudut melengkung pada konten, memberikan tampilan yang lebih modern dan terpisah dengan jelas dari latar belakang. CardView ini memiliki margin 16dp dan menggunakan warna latar belakang dari atribut sistem (?android:attr/colorBackground). Di dalam CardView, terdapat ConstraintLayout lagi yang berfungsi untuk menata elemen-elemen UI, dengan padding 15dp untuk memberikan ruang antara konten dan tepi layout.

Gambar Kamen Rider ditampilkan menggunakan ShapeableImageView dengan lebar 120dp dan tinggi 150dp, serta efek crop di tengah gambar agar tampil dengan proporsi yang tepat. Gambar ini terletak di sisi kiri dan dibatasi oleh constraint yang mengikatnya ke atas, bawah, kiri, dan kanan dari parent layout. Di sebelah kanan gambar, terdapat ConstraintLayout lain yang menampung dua TextView untuk menampilkan nama dan tahun Kamen Rider. Nama Rider memiliki bobot horizontal yang lebih besar (6), sementara tahun ditampilkan dengan bobot lebih kecil (2) dan diatur ke posisi kanan.

Selanjutnya, terdapat dua TextView untuk menampilkan judul dan deskripsi dari Kamen Rider. Deskripsi ditampilkan dalam teks panjang dengan justificationMode="inter_word" untuk memastikan teks diratakan secara merata. Di bagian bawah deskripsi, terdapat ConstraintLayout lainnya yang berisi dua tombol, imdbBtn dan detailBtn, yang masing-masing memiliki ukuran dan radius sudut 15dp, dengan teks yang disesuaikan untuk menampilkan "IMDb" dan "Detail". Tombol-tombol ini diatur untuk berada di posisi bawah dan di sebelah satu sama lain. Dengan desain ini, tampilan kartu Kamen Rider terlihat rapi, terstruktur, dan responsif pada berbagai ukuran layar.

fragment_list.xml

Pada fragment_list.xml, layout ini dirancang untuk menampilkan daftar Kamen Rider menggunakan RecyclerView yang diletakkan di bawah AppBarLayout. Root view menggunakan ConstraintLayout untuk memastikan elemen-elemen dalam layout dapat diatur dengan constraint yang fleksibel dan responsif pada berbagai ukuran layar. Di bagian atas, terdapat AppBarLayout yang berfungsi untuk menampung MaterialToolbar. Toolbar ini memiliki judul "My Favorite Kamen Rider" dengan warna teks yang disesuaikan menggunakan atribut tema ?attr/colorOnPrimary, serta latar belakang

menggunakan ?attr/colorPrimary, yang memberikan tampilan yang konsisten dengan tema aplikasi.

Di bawah toolbar, terdapat RecyclerView yang diatur untuk mengisi seluruh sisa ruang di bawah toolbar dengan padding 10dp di sekelilingnya. RecyclerView ini menggunakan LinearLayoutManager untuk menampilkan daftar item secara vertikal dan disesuaikan dengan layout item yang ada di adapter_main.xml. Jumlah item dalam daftar ditentukan oleh atribut tools:itemCount="7", memberikan gambaran tentang jumlah item yang akan ditampilkan selama desain. Dengan menggunakan ConstraintLayout, RecyclerView ini terhubung dengan toolbar di bagian atas dan memastikan elemen-elemen tersebut terorganisir dengan baik. Desain ini memberikan tampilan yang bersih dan terstruktur dengan ruang yang cukup untuk menampilkan daftar Kamen Rider secara efisien.

fragment_detail.xml

Pada fragment_detail.xml, layout ini dirancang untuk menampilkan detail dari Kamen Rider yang dipilih dalam aplikasi. Root view menggunakan ConstraintLayout untuk pengaturan elemen-elemen yang fleksibel dan responsif. Di bagian atas, terdapat AppBarLayout yang menampung MaterialToolbar dengan judul "Detail" dan ikon navigasi kembali (@drawable/baseline_arrow_back_24) yang memungkinkan pengguna untuk kembali ke layar sebelumnya. Warna teks pada toolbar disesuaikan menggunakan tema dengan atribut ?attr/colorOnPrimary.

Di bawah toolbar, terdapat ScrollView yang memungkinkan konten untuk digulir jika melebihi ukuran layar. Di dalam ScrollView, terdapat ConstraintLayout yang menampung elemen-elemen tampilan seperti gambar, teks, dan deskripsi Kamen Rider. ShapeableImageView digunakan untuk menampilkan gambar poster Kamen Rider dengan bentuk sudut membulat menggunakan @style/RoundedImageView. Di bawah gambar, terdapat TextView yang menampilkan nama Kamen Rider, diikuti dengan TextView lainnya yang menunjukkan tahun rilis, keduanya dengan pengaturan teks yang besar dan tebal untuk memperjelas informasi.

Selanjutnya, terdapat TextView untuk judul deskripsi yang diikuti dengan TextView lain yang menampilkan isi deskripsi Kamen Rider. Semua elemen teks ini diatur dengan margin dan jarak yang cukup agar tampak rapi dan mudah dibaca. Desain ini memberikan tampilan yang terstruktur dan informatif, memungkinkan pengguna untuk melihat detail Kamen Rider dengan jelas, dengan fungsionalitas scroll yang mendukung tampilan konten yang lebih panjang.

fragment_detail.xml (landscape)

Pada fragment_detail.xml versi landscape ini, layout dirancang untuk menampilkan detail Kamen Rider dalam orientasi landscape dengan penataan elemen yang lebih horizontal. Layout utama menggunakan ConstraintLayout sebagai root view, dengan AppBarLayout yang berisi MaterialToolbar di bagian atas, yang mencakup judul "Detail" dan ikon navigasi kembali. Toolbar ini memiliki warna teks disesuaikan dengan tema menggunakan ?attr/colorOnPrimary.

Di bawah toolbar, terdapat ScrollView yang memungkinkan tampilan untuk digulir apabila konten melebihi layar. Dalam ScrollView, terdapat ConstraintLayout dengan elemen-elemen yang disusun untuk memanfaatkan ruang lebih luas dalam orientasi landscape. ShapeableImageView digunakan untuk menampilkan gambar Kamen Rider dengan proporsi yang lebih kecil (240dp x 300dp) dan dengan bentuk sudut membulat. Gambar ini diletakkan di sisi kiri, sementara informasi teks, seperti nama Kamen Rider, tahun rilis, judul deskripsi, dan isi deskripsi, ditempatkan di sebelah kanan gambar dalam ConstraintLayout terpisah.

Pada bagian kanan, elemen-elemen teks disusun vertikal, dengan TextView untuk nama Kamen Rider yang memiliki ukuran teks besar (40sp) dan tebal, diikuti oleh teks tahun rilis yang sedikit lebih kecil, dan judul serta deskripsi yang memberikan informasi lebih mendalam. Setiap TextView memiliki margin dan jarak antar elemen agar tampilan terlihat rapi dan mudah dibaca. Pengaturan layout ini dirancang untuk menyesuaikan tampilan yang lebih luas dan horizontal, memanfaatkan ruang secara optimal dalam orientasi landscape.

main_graph.xml

Pada main_graph.xml, file ini merupakan definisi Navigation Graph yang digunakan untuk mengelola navigasi antar fragment dalam aplikasi Android. Di dalamnya, terdapat dua fragment yang didefinisikan, yaitu listFragment dan detailFragment. listFragment merupakan fragment pertama yang muncul sebagai tampilan awal (start destination) dari aplikasi, yang ditandai dengan atribut app:startDestination="@id/listFragment". Fragment ini akan menampilkan daftar Kamen Rider, yang disesuaikan dengan layout fragment_list.xml.

detailFragment adalah fragment kedua yang digunakan untuk menampilkan detail informasi Kamen Rider tertentu. Fragment ini memiliki satu argument, yaitu riderId, yang bertipe integer, yang akan diteruskan saat navigasi dari listFragment ke detailFragment. Argument ini dapat digunakan untuk menampilkan data spesifik berdasarkan Kamen Rider yang dipilih. Di dalam fragment ini, layout yang digunakan adalah fragment_detail.xml.

Untuk menghubungkan kedua fragment, terdapat sebuah action yang memungkinkan navigasi dari listFragment ke detailFragment. Action ini memiliki ID action_listFragment_to_detailFragment dan mendefinisikan bahwa tujuan navigasi adalah detailFragment. Dengan demikian, saat pengguna memilih suatu item dari listFragment, aplikasi akan menavigasi ke detailFragment dan menampilkan informasi yang lebih detail berdasarkan ID yang diteruskan.

D. Tautan Git

Berikut adalah tautan untuk source code yang telah dibuat.
Versi Jetpack Compose:
https://github.com/raihan2030/Praktikum-Pemrograman-Mobile/tree/main/Praktikum-3/Compose

Versi XML:
https://github.com/raihan2030/Praktikum-Pemrograman-Mobile/tree/main/Praktikum-3/XML

SOAL 2

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

Jawab:

Karena RecyclerView lebih fleksibel dan lebih leluasa dalam melakukan kustomisasi terhadap perilaku, tampilan, dan animasi elemen di dalam daftar yang ditampilkan Selain itu, Recycler view juga lebih unggul dalam segi performa karena RecyclerView mengoptimalkan memori dengan hanya membuat tampilan yang terlihat oleh pengguna (view recycling), ini membuatnya sangat efisien untuk aplikasi dengan data yang besar.

Jadi, meskipun LazyColumn menawarkan kemudahan dalam penggunaan dan sintaksis yang lebih bersih, RecyclerView masih tetap relevan karena lebih fleksibel, kompatibel dengan berbagai versi Android, dan lebih kuat dalam menangani skenario yang lebih kompleks.