LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 5

CONNECT TO THE INTERNET

Oleh:

Jovan Gilbert Natamasindah NIM. 2310817310002

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
JUNI 2025

LEMBAR PENGESAHAN
LAPORAN PRAKTIKUM PEMROGRAMAN I
MODUL 5

Laporan Praktikum Pemrograman Mobile Modul 5: Connect to the Internet ini disusun sebagai syarat lulus mata kuliah Praktikum Pemrograman Mobile. Laporan Prakitkum ini dikerjakan oleh:

Nama Praktikan : Jovan Gilbert Natamasindah
NIM : 2310817310002

Menyetujui, Asisten Praktikum	Mengetahui, Dosen Penanggung Jawab Praktikum
Zulfa Auliya Akbar NIM. 2210817210026	Muti`a Maulida S.Kom M.T.I NIP. 19881027 201903 20 13

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

SOAL 1

Soal Praktikum:

Lanjutkan aplikasi Android yang sudah dibuat pada Modul 4 dengan menambahkan modifikasi

sesuai ketentuan berikut:

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

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

A. Source Code

1. MainActivity.kt

```
1   package com.android.modul5
2
3   import MovieListScreen
4   import android.content.Context
5   import android.os.Bundle
6   import androidx.activity.ComponentActivity
7   import androidx.activity.compose.setContent
8   import androidx.activity.enableEdgeToEdge
9   import androidx.compose.animation.slideInHorizontally
10  import androidx.compose.animation.slideOutHorizontally
11  import androidx.compose.material3.Text
12  import androidx.compose.runtime.Composable
13  import androidx.compose.runtime.getValue
14  import androidx.compose.runtime.mutableStateOf
15  import androidx.compose.runtime.remember
16  import androidx.compose.runtime.setValue
17  import androidx.compose.ui.platform.LocalContext
18  import androidx.core.content.edit
```

```
19 import androidx.lifecycle.viewmodel.compose.viewModel
20 import androidx.navigation.NavType
21 import androidx.navigation.compose.NavHost
22 import androidx.navigation.compose.composable
23 import androidx.navigation.compose.rememberNavController
24 import androidx.navigation.navArgument
25 import
   com.android.modul5.presentation.screens.MovieDetailScreen
26 import
   com.android.modul5.presentation.ui.theme.MODUL5Theme
27 import
   com.android.modul5.presentation.viewmodel.MovieViewModel
28
29 class MainActivity : ComponentActivity() {
30     override fun onCreate(savedInstanceState: Bundle?) {
31         super.onCreate(savedInstanceState)
32         enableEdgeToEdge()
33         setContent {
34             val context = LocalContext.current
35             val prefs = remember
   {context.getSharedPreferences("app_prefs",
   Context.MODE_PRIVATE)}
36             var isDarkMode by remember {
   mutableStateOf(prefs.getBoolean("is_dark_mode", false)) }
37
38             MODUL5Theme(darkTheme = isDarkMode) {
39                 NavMovies(
40                     isDarkMode = isDarkMode,
41                     onToggle = { newValue ->
42                         isDarkMode = newValue
43                         prefs.edit {
   putBoolean("is_dark_mode", newValue) }
44                     })
45             }
46         }
47     }
48 }
49
50 @Composable
51 fun NavMovies(isDarkMode: Boolean, onToggle: (Boolean) ->
   Unit) {
52     val navController = rememberNavController()
53     val movieViewModel: MovieViewModel = viewModel()
54
55     NavHost(
56         navController = navController,
57         startDestination = "movie_list",
```

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

Tabel 1. Source Code Jawaban Soal 1 MainActivity

2. presentation/viewmodel/MovieViewModel.kt

```
1  package com.android.modul5.presentation.viewmodel
2
3  import androidx.lifecycle.ViewModel
4  import androidx.lifecycle.viewModelScope
5  import com.android.modul5.data.api.RetrofitClient
6  import com.android.modul5.domain.model.Movie
```

```
import kotlinx.coroutines.flow.MutableStateFlow
import kotlinx.coroutines.flow.StateFlow
import kotlinx.coroutines.launch

class MovieViewModel: ViewModel() {
    private val API_KEY =
"9d2494a8a2a5c08592c8e963a74c799a"

    private val _Movies =
MutableStateFlow<List<Movie>>(emptyList())
    val Movies: StateFlow<List<Movie>> = _Movies

    private val _selectedMovieDetail =
MutableStateFlow<Movie?>(null)
    val selectedMovieDetail: StateFlow<Movie?> =
_selectedMovieDetail

    private val _errorMsg =
MutableStateFlow<String?>(null)
    val errorMsg: StateFlow<String?> = _errorMsg

    init {
        fetchMovies()
    }

    fun fetchMovies() {
        viewModelScope.launch {
            _errorMsg.value = null
            val Response =
RetrofitClient.tmdbAPI.getPopularMovies(API_KEY)
            try {
                if (Response.isSuccessful) {
                    Response.body()?.let { _Movies.value =
it.results }
                }
                else {_errorMsg.value = "Error in movie
List= ${Response.code()} - ${Response.message()}"}
            }
            catch (err: Exception) {_errorMsg.value =
"Exception: ${err.localizedMessage} ?: Unknown error"}
        }
    }

    fun fetchMoviebyDetailbyID(movieId: Int) {
        _errorMsg.value = null
        viewModelScope.launch {
```

```
44              val responseDetail =
    RetrofitClient.tmdbAPI.getMovieDetailbyID(movieId,API_KEY)
45              try {
46                  if (responseDetail.isSuccessful) {
47                      _selectedMovieDetail.value =
    responseDetail.body()
48                  }
49                  else {_errorMsg.value = "Error in movie
    detail = ${responseDetail.code()} -
    ${responseDetail.message()}"}
50              }
51              catch (err: Exception) {_errorMsg.value =
    "Exception: ${err.localizedMessage} ?: Unknown error"}
52          }
53      }
54  }
```

Tabel 2. Source Code Jawaban Soal 1 MovieViewModel

3. presentation/screens/MovieDetailScreen.kt

```
1   package com.android.modul5.presentation.screens
2
3   import androidx.compose.foundation.layout.Box
4   import androidx.compose.foundation.layout.Column
5   import androidx.compose.foundation.layout.Spacer
6   import androidx.compose.foundation.layout.fillMaxSize
7   import androidx.compose.foundation.layout.height
8   import androidx.compose.foundation.layout.padding
9   import androidx.compose.foundation.layout.width
10  import androidx.compose.foundation.lazy.LazyColumn
11  import androidx.compose.foundation.lazy.items
12  import androidx.compose.foundation.rememberScrollState
13  import androidx.compose.foundation.verticalScroll
14  import androidx.compose.material3.Button
15  import androidx.compose.material3.Scaffold
16  import androidx.compose.material3.Text
17  import androidx.compose.runtime.Composable
18  import androidx.compose.runtime.DisposableEffect
19  import androidx.compose.runtime.LaunchedEffect
20  import androidx.compose.runtime.collectAsState
21  import androidx.compose.runtime.getValue
22  import androidx.compose.ui.Alignment
23  import androidx.compose.ui.Modifier
24  import androidx.compose.ui.focus.focusModifier
25  import androidx.compose.ui.text.font.FontWeight
26  import androidx.compose.ui.text.style.TextAlign
27  import androidx.compose.ui.unit.dp
28  import androidx.navigation.NavController
```

```
29 import com.android.modul5.presentation.components.ButtonNav
30 import com.android.modul5.presentation.components.Desc
31 import com.android.modul5.presentation.components.Glide
32 import com.android.modul5.presentation.components.MovieCard
33 import com.android.modul5.presentation.components.TopBar
34 import com.android.modul5.presentation.viewmodel.MovieViewModel
35 import
   com.bumptech.glide.integration.compose.ExperimentalGlideComposeApi
36 import com.bumptech.glide.integration.compose.GlideImage
37
38 @OptIn(ExperimentalGlideComposeApi::class)
39 @Composable
40 fun MovieDetailScreen(movieId: Int, movieViewModel:
   MovieViewModel, navController: NavController) {
41     val movieDetail by
   movieViewModel.selectedMovieDetail.collectAsState()
42     val errMsg by movieViewModel.errorMsg.collectAsState()
43
44     LaunchedEffect(movieId) {
45         movieViewModel.fetchMoviebyDetailbyID(movieId)
46     }
47
48     Scaffold(
49         topBar = { TopBar( Title = movieDetail?.title ?: "Detail
   Film") }
50     ) { innerPadding ->
51         Column(
52             modifier = Modifier
53                 .fillMaxSize()
54                 .padding(innerPadding)
55                 .padding(15.dp)
56                 .verticalScroll(rememberScrollState()),
57             horizontalAlignment = Alignment.CenterHorizontally
58         ) {
59             errMsg?.let {msg -> Text(msg) }
60             Box(
61                 modifier = Modifier
62                     .width(780.dp)
63                     .height(300.dp)
64             ) {
65                 Glide(movieDetail?.posterPath)
66             }
67             Spacer(modifier = Modifier.height(10.dp))
68             Desc("Tanggal Rilis", movieDetail?.releaseDate)
69             Desc("Popularitas", movieDetail?.popularity)
70             Desc("Rata-rata Vote", movieDetail?.voteAverage)
71             Spacer(modifier = Modifier.height(10.dp))
```

```
72              Text("${movieDetail?.overview}", textAlign =
    TextAlign.Justify)
73              Spacer(modifier = Modifier.height(10.dp))
74              ButtonNav("Kembali", navController)
75          }
76      }
77  }
```

Tabel 3. Source Code Jawaban Soal 1 MovieDetailScreen

4. presentation/screens/MovieListScreen.kt

```
1   import androidx.compose.foundation.layout.Arrangement
2   import androidx.compose.foundation.layout.Column
3   import
    androidx.compose.foundation.layout.ExperimentalLayoutApi
4   import androidx.compose.foundation.layout.FlowRow
5   import androidx.compose.foundation.layout.fillMaxSize
6   import androidx.compose.foundation.layout.fillMaxWidth
7   import androidx.compose.foundation.layout.padding
8   import androidx.compose.foundation.rememberScrollState
9   import androidx.compose.foundation.verticalScroll
10  import androidx.compose.material3.Scaffold
11  import androidx.compose.material3.Text
12  import androidx.compose.runtime.Composable
13  import androidx.compose.runtime.collectAsState
14  import androidx.compose.runtime.getValue
15  import androidx.compose.ui.Modifier
16  import androidx.navigation.NavController
17  import
    com.android.modul5.presentation.components.MovieCard
18  import com.android.modul5.presentation.components.TopBar
19  import
    com.android.modul5.presentation.viewmodel.MovieViewModel
20  import
    com.android.modul5.presentation.components.DarkModeSwitch
21
22  @OptIn(ExperimentalLayoutApi::class)
23  @Composable
24  fun MovieListScreen(movieViewModel: MovieViewModel,
    navController: NavController, isDarkMode: Boolean,
    onToggle: (Boolean) -> Unit) {
25      val Movies by movieViewModel.Movies.collectAsState()
26      val errMsg by
    movieViewModel.errorMsg.collectAsState()
27
28      Scaffold(
29          topBar = { TopBar("SIMOVIE") }
30      ) { innerPadding ->
```

```
31          Column(
32              modifier = Modifier
33                  .fillMaxSize()
34                  .padding(innerPadding)
35                  .verticalScroll(rememberScrollState()),
36          ) {
37              DarkModeSwitch(isDarkMode = isDarkMode,
    onToggle = onToggle)
38
39              errMsg?.let {msg -> Text(msg) }
40              FlowRow(
41                  modifier = Modifier
42                      .fillMaxWidth(),
43                  horizontalArrangement =
    Arrangement.Center
44              ) {
45                  Movies.forEach { movie ->
46                      MovieCard(movieItem = movie,
    navController)
47                  }
48              }
49          }
50      }
51  }
```

Tabel 4. Source Code Jawaban Soal 1 MovieListScreen

5. presentation/components/TopBar.kt

```
1   package com.android.modul5.presentation.components
2
3   import
    androidx.compose.material3.ExperimentalMaterial3Api
4   import androidx.compose.material3.MaterialTheme
5   import androidx.compose.material3.Text
6   import androidx.compose.material3.TopAppBar
7   import androidx.compose.material3.TopAppBarDefaults
8   import androidx.compose.runtime.Composable
9   import androidx.compose.ui.text.font.FontWeight
10
11  @OptIn(ExperimentalMaterial3Api::class)
12  @Composable
13  fun TopBar(Title: String) {
14      TopAppBar(
15          title = {
16              Text(
17                  Title,
18                  fontWeight = FontWeight.Bold,
19              ) },
```

```
20          colors = TopAppBarDefaults.topAppBarColors(
21              containerColor =
    MaterialTheme.colorScheme.primary,
22              titleContentColor =
    MaterialTheme.colorScheme.tertiary
23          )
24      )
25  }
```

Tabel 5. Source Code Jawaban Soal 1 TopBar

6. presentation/components/Text.kt

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

Tabel 6. Source Code Jawaban Soal 1 Text

7. presentation/components/movieCard.kt

```
1   package com.android.modul5.presentation.components
2
3   import androidx.compose.foundation.layout.Column
4   import androidx.compose.foundation.layout.Spacer
5   import androidx.compose.foundation.layout.fillMaxWidth
6   import androidx.compose.foundation.layout.height
7   import androidx.compose.foundation.layout.padding
8   import androidx.compose.foundation.layout.width
9   import androidx.compose.material3.Card
10  import androidx.compose.material3.CardDefaults
11  import androidx.compose.material3.MaterialTheme
12  import androidx.compose.material3.Text
```

```
13 import androidx.compose.runtime.Composable
14 import androidx.compose.ui.Modifier
15 import androidx.compose.ui.text.font.FontWeight
16 import androidx.compose.ui.text.style.TextAlign
17 import androidx.compose.ui.unit.dp
18 import androidx.compose.ui.unit.sp
19 import androidx.navigation.NavController
20 import com.android.modul5.domain.model.Movie
21
22 @Composable
23 fun MovieCard(movieItem: Movie, navController:
   NavController) {
24     Card(
25        modifier = Modifier
26            .width(180.dp)
27            .padding(8.dp),
28         shape = MaterialTheme.shapes.small,
29         elevation =
   CardDefaults.cardElevation(defaultElevation = 10.dp)
30         ) {
31         Glide(movieItem.posterPath)
32         Column(modifier =Modifier.padding(horizontal =
   8.dp)) {
33             Text(text = movieItem.title, fontWeight =
   FontWeight.W600, lineHeight = 17.sp, modifier =
   Modifier.fillMaxWidth(), textAlign = TextAlign.Center)
34             Spacer(modifier = Modifier.height(15.dp))
35             Text(text = "Asal Negara:
   ${movieItem.originalLanguage}", fontSize = 14.sp)
36             Text(text = "Tanggal Rilis:
   ${movieItem.releaseDate}", fontSize = 14.sp)
37         }
38         Spacer(modifier = Modifier.height(15.dp))
39         ButtonNav("Detail", navController, movieItem.id)
40         Spacer(modifier = Modifier.height(10.dp))
41
42     }
43 }
```

Tabel 7. Source Code Jawaban Soal 1 movieCard

8. presentation/components/Glide.kt

```
1 package com.android.modul5.presentation.components
2
3 import androidx.compose.foundation.layout.Column
4 import androidx.compose.foundation.layout.fillMaxWidth
5 import androidx.compose.runtime.Composable
6 import androidx.compose.ui.Alignment
```

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

Tabel 8. Source Code Jawaban Soal 1 Glide

9. presentation/components/DarkModeSwitchkt

```
1  package com.android.modul5.presentation.components
2
3
4  import androidx.compose.foundation.layout.Arrangement
5  import androidx.compose.foundation.layout.Row
6  import androidx.compose.foundation.layout.Spacer
7  import androidx.compose.foundation.layout.fillMaxSize
8  import androidx.compose.foundation.layout.size
9  import androidx.compose.foundation.layout.width
10 import androidx.compose.material3.Icon
11 import androidx.compose.material3.Switch
12 import androidx.compose.runtime.Composable
13 import androidx.compose.ui.Alignment
14 import androidx.compose.ui.Modifier
15 import androidx.compose.ui.res.painterResource
16 import androidx.compose.ui.unit.dp
17 import com.android.modul5.R
18
19 @Composable
20 fun DarkModeSwitch(isDarkMode: Boolean, onToggle:
   (Boolean)->Unit) {
21     Row(
22         modifier = Modifier.fillMaxSize(),
23         verticalAlignment = Alignment.CenterVertically,
24         horizontalArrangement = Arrangement.Center,
25
26     ) {
```

```
27          Icon(
28              painter = painterResource(id =
    R.drawable.sun_solid),
29              contentDescription = "Icon Sun" ,
30              modifier = Modifier.size(20.dp)
31
32          )
33          Spacer(modifier = Modifier.width(5.dp))
34          Switch(
35              checked = isDarkMode,
36              onCheckedChange = onToggle
37          )
38          Spacer(modifier = Modifier.width(5.dp))
39          Icon(
40              painter = painterResource(id =
    R.drawable.moon_solid),
41              contentDescription = "Icon moon",
42              modifier = Modifier.size(20.dp)
43          )
44      }
45
46  }
```

Tabel 9. Source Code Jawaban Soal 1 DarkModeSwitch

10. presentation/components/ButtonNav.kt

```
1   package com.android.modul5.presentation.components
2
3   import androidx.compose.foundation.layout.Column
4   import androidx.compose.foundation.layout.fillMaxSize
5   import androidx.compose.foundation.layout.height
6   import androidx.compose.foundation.layout.width
7   import androidx.compose.material3.Button
8   import androidx.compose.material3.MaterialTheme
9   import androidx.compose.material3.Text
10  import androidx.compose.runtime.Composable
11  import androidx.compose.ui.Alignment
12  import androidx.compose.ui.Modifier
13  import androidx.compose.ui.unit.dp
14  import androidx.navigation.NavController
15
16  @Composable
17  fun ButtonNav(action: String, navController:
    NavController, movieId: Int = 0) {
18      Column(modifier = Modifier.fillMaxSize(),
    horizontalAlignment = Alignment.CenterHorizontally) {
19          Button(
20              onClick = {
```

```
                    if (action == "Detail")
{navController.navigate("movie_detail/${movieId}")}
                    else {navController.popBackStack()}
                },
                shape = MaterialTheme.shapes.small,
                modifier =
Modifier.width(100.dp).height(35.dp)
            ) {
                Text(action, color =
MaterialTheme.colorScheme.tertiary)
            }
        }

}
```

Tabel 10. Source Code Jawaban Soal 1 ButtonNav

11. domain/model/Movie.kt

```
package com.android.modul5.domain.model

import kotlinx.serialization.SerialName
import kotlinx.serialization.Serializable

@Serializable
data class MovieResponse(
    val page: Int,
    val results: List<Movie>,
    val total_pages: Int,
    val total_results: Int
)

@Serializable
data class Movie(
    val id: Int,
    val popularity: Double,
    @SerialName("original_language")
    val originalLanguage: String,
    val overview: String,
    @SerialName("poster_path")
    val posterPath: String?,
    @SerialName("release_date")
    val releaseDate: String,
    val title: String,
    @SerialName("vote_average")
    val voteAverage: Double,
)
```

Tabel 11. Source Code Jawaban Soal 1 Movie

12. data/api/TMDBAPI.kt

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

Tabel 12. Source Code Jawaban Soal 1 TMDBAPI

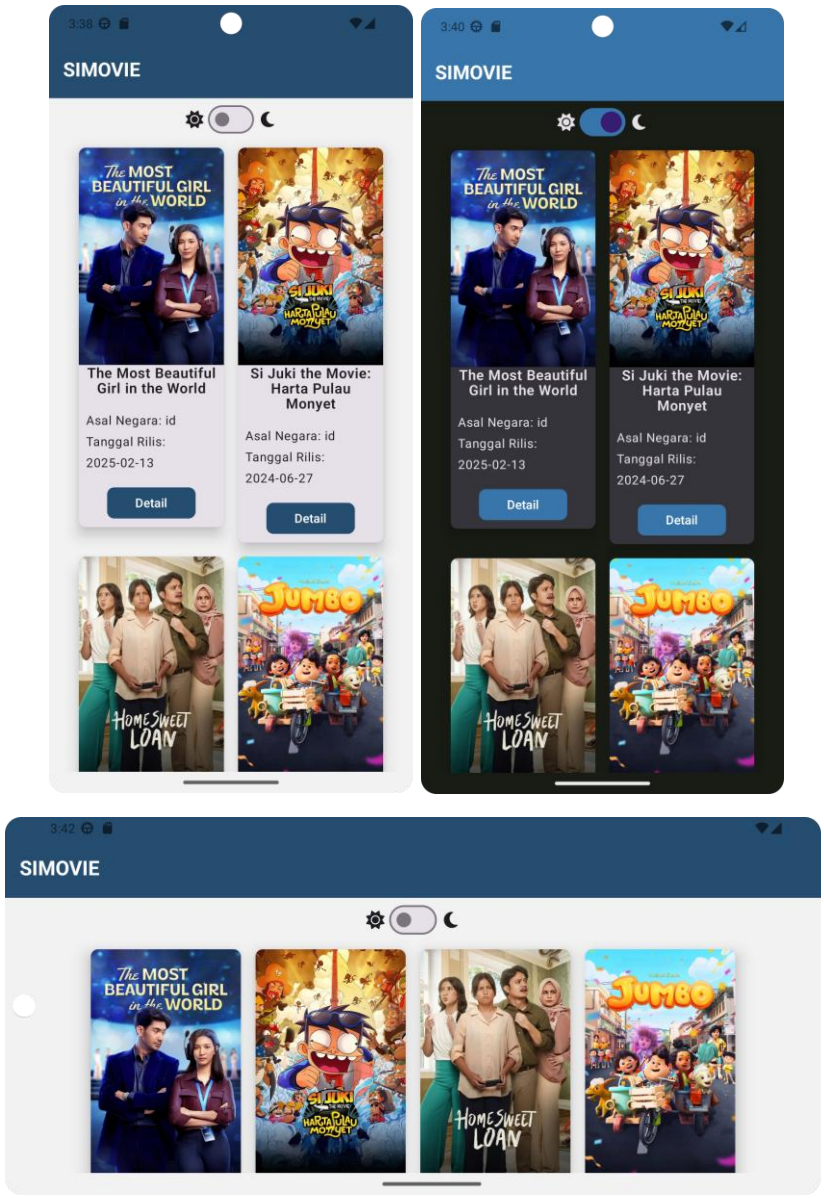
13. data/api/RetrofitClient.kt

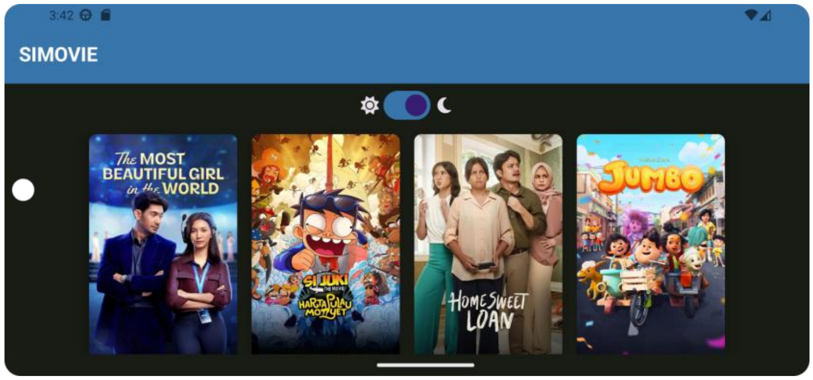
```
1  package com.android.modul5.data.api
2
3  import kotlinx.serialization.json.Json
4  import okhttp3.MediaType.Companion.toMediaType
5  import retrofit2.Retrofit
6  import
   retrofit2.converter.kotlinx.serialization.asConverterFactory
7
8  object RetrofitClient {
9      private val json = Json {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

```
17
    .addConverterFactory(json.asConverterFactory("application/json"
    .toMediaType()))
18              .build()
19              .create(TMDBAPI::class.java)
20      }
21  }
```

Tabel 13. Source Code Jawaban Soal 1 RetrofitClient

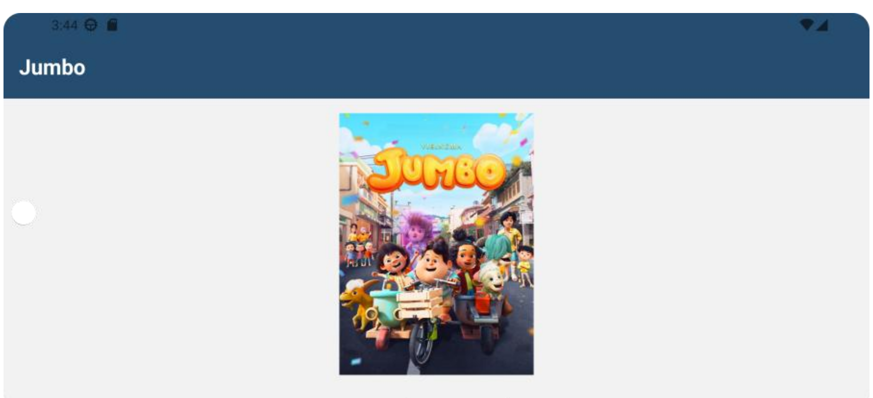
B. Output Program

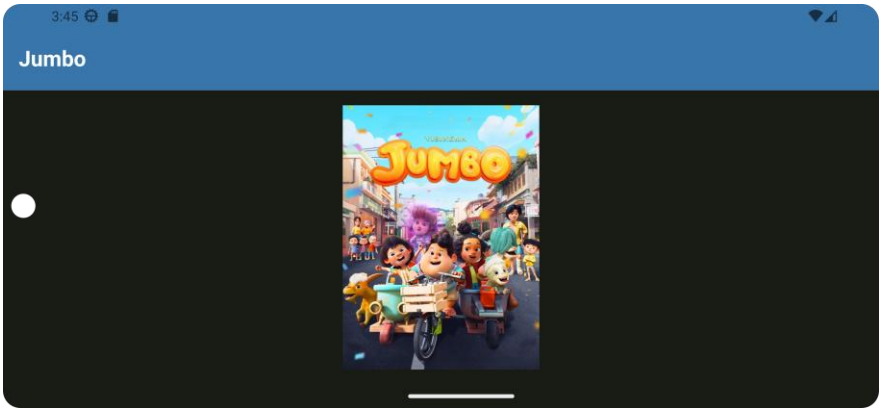



3:42
SIMOVIE
The MOST BEAUTIFUL GIRL in the WORLD
SI JUKI
HARTA PULAU MONYET
Home Sweet LOAN
JUMBO


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


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


3:44
Jumbo
JUMBO



Gambar 1 Screenshot Hasil Jawaban Soal 1

C. Pembahasan

1. MainActivity.kt:

- Pada baris 1, **package com.android.modul5** pendeklarasian nama package file Kotlin.
- Pada baris 3-27, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
- Pada baris 29, **class MainActivity : ComponentActivity(),** merupakan titik mula yang menjadi kelas utama dan akan dijalankan pertama kali saat aplikasi dibuka
- Pada baris 30, **override fun onCreate(savedInstanceState: Bundle?),** berfungsi untuk menimpa (override) fungsi oncreate dari ComponentActivity.
- Pada baris 31, **super.onCreate(savedInstanceState),** berfungsi untuk memanggil superclass dari fungsi oncreate untuk memastikan bahwa proses inisialisasi standar dari Android dijalankan sebelum logika saya dijalankan.
- Pada baris 32, **enableEdgeToEdge(),** berfungsi agar tampilan aplikasi dapat menggunakan seluruh layar dari status bar sampai navigation bar atau fullscreen layout.
- Pada baris 26, **setContent(),** digunakan untuk menampilkan UI berbasis jetpack compose ke dalam activity
- Pada baris 34, **val context = LocalContext.current** berfungsi untuk mendapatkan context dari komposisi saat ini
- Pada baris 35, **val prefs = remember {context.getSharedPreferences("app_prefs", Context.MODE_PRIVATE)}** berfungsi untuk mendapatkan instance SharedPreference dengan nama app_prefs dan diingat agar tidak di dibuat ulang setiap kali recompose.

- Pada baris 36, **var isDarkMode by remember { mutableStateOf(prefs.getBoolean("is_dark_mode", false)) }** berfungsi untuk mendeklarasikan state dengan nama isDarkMode yang nilai awalnya diambil dari SharedPrefrence.
- Pada baris 38, **Modul5Theme(darkTheme = isDarkMode),** merupakan fungsi yang berisi tema custom yang membungkus seluruh UI untuk memberikan style yang konsisten.
- Pada baris 39, **NavMovies(isDarkMode = isDarkMode, onToggle = {newValue -> isDarkMode = newValue prefs.edit {putBoolean("is_dark_mode", newValue)}})** merupakan fungsi composable untuk mengelola navigasi antar halaman, yang menerima state isDarkMode, lambda onToggle
- Pada baris 50, **@composable,** merupakan annotation untuk menandai bahwa fungsi yang dibuat dibawahnya merupakan fungsi UI di jetpack compose.
- Pada baris 51, **fun NavMovies(isDarkMode: Boolean, onToggle: (Boolean) -> Unit)** merupakan fungsi composable yang dibuat untuk menampilkan semua composable yang ingin ditampilkan ke dalam activity, seperti MovieListScreen() dan MovieDetailScreen() lalu menerima parameter isDarkMode dan onToggle.
- Pada baris 52, **val navController = rememberNavController()** berfungsi untuk membuat dan menyimpan instance dari NavController
- Pada baris 53, **val movieViewModel: MovieViewModel = viewModel()** berfungsi untuk mendapatkan MovieViewModel yang akan digunakan untuk halaman-halaman di dalam aplikasi.
- Pada baris 55, **NavHost(),** berfungsi untuk menyediakan wadah untuk navigasi antar composable screen berdasarkan rute yang ditentukan
    - Pada baris 56, **navController** merupakan controller yang bertanggung jawab atas navigasi.
    - Pada baris 57, **startDestination** merupakan rute awal yang ditampilkan ketika aplikasi dijalankan.
    - Pada baris 58, **enterTransition** berfungsi untuk mendefinisikan animasi pada saat memasuki layar
    - Pada baris 61, **exitTransition** berfungsi untuk mendefinisikan animasi pada saat keluar dari layar

    - Pada baris 64, **popEnterTransition** berfungsi untuk mendefinisikan animasi pada saat memasuki Kembali layar setelah operasi pop
    - Pada baris 67, **popExitTransition** berfungsi untuk mendefinisikan animasi pada saat keluar dari layar setelah operasi pop
- Pada baris 71 dan 74, **composable(),** berfungsi untuk mendefinisikan rute, seperti di kasus ini ada "movie_list" dan "movie_detail"
    - Pada baris 75, **route** bergunsi untuk rute dari halaman yang dituju dan akan menerima data movieId melalui url
    - Pada baris 76, **arguments** berfungsi untuk mendefinisikan daftar argument untuk diteruskan
- Pada baris 72, **MovieList(navController, ViewModel),** merupakan fungsi untuk menampilkan list yang dapat scroll sekaligus menjadi halaman utama aplikasi dan menerima parameter navController, movieViewModel, isDarkMode, dan onToggle.
- pada baris 51, **MovieDetailScreen(movieId, movieViewModel, navController)** merupakan fungsi untuk menampilkan halaman detail dari aplikasi dan menerima beberapa parameter.

2. presentation/screens/MovieListScreen.kt

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

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

3. presentation/screens/MovieDetailScreen.kt

- Pada baris 1, **package com.android.modul5.presentation.screens** pendeklarasian nama package file Kotlin.
- Pada baris 3, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
- Pada baris 39, **@Composable** merupakan annotation untuk menandai bahwa fungsi yang dibuat dibawahnya merupakan fungsi UI di jetpack compose.
- Pada baris 40, **fun MovieDetailScreen(movieId: Int, movieViewModel: MovieViewModel, navController: NavController)**

- Pada baris 41, **val Movies by movieViewModel.Movies.CollectAsState()** berfungsi untuk mengumpulkan aliran atau flow dari MovieViewModel.Movies menjadi state untuk memperbarui UI jika data film mengalami perubahan
- Pada baris 42, **val errMsg by movieViewModel.errorMsg.CollectAsState()** berfungsi untuk mengumpulkan aliran atau flow dari MovieViewModel.errorMsg menjadi state untuk memperbarui UI jika ada error yang terjadi
- Pada baris 44, **LaunchedEffect(movieId) {movieViewModel.fetchMoviebyDetailbyID(movieId)}** merupakan sebuah composable effect untuk memicukode didalamnya saat ada perubahan pada movieId saat composable pertama kali masuk komposisi
- pada baris 48, **scaffold** merupakan komponen layout yang menjadi kerangka dasar dari tampilan yang mempermudah dalam mengatur strktur UI karena sudah disediakan slot-slot bawannya seperti topBar.
- pada baris 49, **topBar** merupakan tempat Dimana topBar akan diletakkan di scaffold
- pada baris 51, **Column** merupakan komponen layout mirip seperti scaffold, hanya saja Column digunakan untuk menyusun elemen UI secara vertical.
    - pada baris 32, **modifier** berfungsi untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.padding* untuk memberikan jarak dalam component, *.fillmaxsize* membuat column memenuhi layar *.verticalscroll* membuat konten dapat di scroll
    - pada baris 57, **horizontalAlignment** berfungsi untuk menentukan posisi semua elemen di dalam Column secara horizontal.
- pada baris 59, **errMsg?.let {msg -> Text(msg) }** akan menampilkan pesan error jika errMsg tidak bernilai null
- pada baris 60, **Box** merupakan sebuah composable untuk menumpuk elemen
    - pada baris 61, **modifier** berfungsi untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.height* untuk mengatur tinggi dan *.width* untuk mengatur lebar.
- pada baris 60, **Glide()** merupakan fungsi yang saya buat sendiri untuk memudahkan dalam memuat gambar.
- Pada baris 67-73, **Spacer()** berfungsi untuk memberikan jarak kosong biasanya secara vertical dalam layout.
- Pada baris 68-70, **Desc()** merupakan fungsi yang saya buat sendiri untuk memudahkan dalam membuat deskripsi.

- Pada baris 72, **Text()** merupakan komponen UI yang berfunsi menampilkan teks.
    - Pada baris 72, **textAlign** untuk membuat apakah teks berada di kiri, Tengah , atau kanan
- Pada baris 74, **ButtonNav()** merupakan fungsi yang saya buat sendiri untuk menampilkan tombol ke halaman detail

4. Presentation/components/TopBar.kt

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

5. Presentation/components/Text.kt

- Pada baris 1, **package com.android.modul5.components** pendeklarasian nama package file Kotlin.

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

6. Presentation/components/movieCard.kt

- Pada baris 1, **package com.android.modul5.presentation.components** pendeklarasian nama package file Kotlin.
- Pada baris 3-20, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
- Pada baris 22, **@Composable** merupakan annotation untuk menandai bahwa fungsi yang dibuat dibawahnya merupakan fungsi UI di jetpack compose.
- Pada baris 23, **fun MovieCard(movieItem: Movie, navController: NavController)** berfungsi untuk menampilkan Movie yang didapatkan dalam bentuk Card, lalu menerima parameter movieItem dan navController
- Pada baris 24, **Card** berfungsi untuk membuat sebuah kartu.
    - Pada baris 25, **modifier m**erupakan parameter dalam Column() untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.padding* untuk memberikan jarak dalam component, *.width* untuk lebar component
    - Pada baris 26, **shape** merupakan parameter dalam Button() untuk mengatur sudut agar dapat membulat.

    - Pada baris 27, **elevation** meruapakan parameter untuk mengatur Tingkat kedalamn bayang dari elemen
- Pada baris 31, **Glide** merupakan fungsi yang saya buat sendiri untuk memudahkan dalam memuat gambar.
- Pada baris 32, **Column,** merupakan komponen layout mirip seperti scaffold, hanya saja Column digunakan untuk menyusun elemen UI secara vertical.
    - Pada baris 33, **modifier** berfungsi untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.padding* untuk memberikan jarak dalam
- Pada baris 33-36, **Text** merupakan komponen UI yang berfunsi menampilkan teks.
    - Pada baris 33-36, **text** merupakan tempat string yang akan ditampilkan diletakkan
    - Pada baris 33, **fontWeight** merupakan ketebalan dari teks yang akan ditampilkan
    - Pada baris 33, **lineHeight** merupakan tinggi dari baris tempat teks berada
    - Pada baris 35 dan 36, **fontSize** merupakan ukuran dari font yang akan ditampilkan
- Pada baris 34-40, **Spacer** berfungsi untuk memberikan jarak kosong biasanya secara vertical dalam layout.
- Pada baris 39, **ButtonNav** merupakan fungsi yang saya buat sendiri untuk menampilkan tombol Kembali ke halaman utama

7. Presentation/components/Glide.kt

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

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

8. Presentation/components/DarkModeSwitch.kt

- Pada baris 1, **package com.android.modul5.presentation.components** pendeklarasian nama package file Kotlin.
- Pada baris 4-17, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
- Pada baris 19, **@Composable** merupakan annotation untuk menandai bahwa fungsi yang dibuat dibawahnya merupakan fungsi UI di jetpack compose.
- Pada baris 20, **fun DarkModeSwitch(isDarkMode: Boolean, onToggle: (Boolean)->Unit)** merupakan fungsi untuk membuat switch untuk keperluan darkmode yang menrima parameter isDarkMode dan onToggle
- Pada baris 21, **Row** merupakan komponen layout yang berlawanan dari Column, Dimana jika Column menyusun elemen UI secara Vertical, maka Row menyusun elemen UI secara horizontal.
    - Pada baris 22, **modifier** merupakan parameter untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.fillMaxWidth* agar column lebar Row mengisi ukuran layar.
    - Pada baris 23, **verticalAlignment** berfungsi untuk posisi semua elemen di dalam row secara vertical
    - Pada baris 24, **horizontalArrangement** berfungsi untuk mengattur jarak elemen agar tpat berada ditengah
- Pada baris 27 dan 39, **Icon** merupakan sebuah fungsi untuk menampilkan icon
    - Pada baris 28 dan 40, **painter** berfungsi sebagai tempat memuat icon dari drawable

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

9. Presentation/components/ButtonNav.kt

- Pada baris 1, **package com.android.modul5.presentation.components** pendeklarasian nama package file Kotlin.
- Pada baris 3-14, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
- Pada baris 16, **@Composable** merupakan annotation untuk menandai bahwa fungsi yang dibuat dibawahnya merupakan fungsi UI di jetpack compose.
- Pada baris 17, **fun ButtonNav(action: String, navController: NavController, movieId: Int = 0)** merupakan fungsi untuk membuat tombol untuk navigasi antar halaman yang menerima parameter action dan navController
- pada baris 18, **Column** merupakan komponen layout mirip seperti scaffold, hanya saja Column digunakan untuk menyusun elemen UI secara vertical.
    - pada baris 18, **modifier** berfungsi untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.fillmaxsize* unruk memenuhi seluruh layar
    - pada baris 18, **horizontalAlignment** berfungsi untuk menentukan posisi semua elemen di dalam Column secara horizontal.
- Pada baris 19, **Button()** merupakan komponen UI yang berfungsi untuk membuat tombol pada aplikasi.
    - Pada baris 20, **onClick** merpakan parameter dalam Button() untuk memberikan aksi yang akan dijalankan ketika tombol ditekan.

    - Pada baris 24, **shape** merupakan parameter dalam Button() untuk mengatur sudut agar dapat membulat
    - Pada baris 25, **modifier** merupakan parameter dalam TextField untuk menerapkan modifikasi terhadap tampilan komponen, seperti *.width* dan *.height* agar column lebar dan tinggi Button mengisi ukuran layar.
- Pada baris 27, **Text** merupakan komponen UI yang berfunsi menampilkan teks.

10. domain/model/Movie.kt

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

11. data/api/TMDBAPI.kt

- Pada baris 1, **package com.android.modul5.data.api** pendeklarasian nama package file Kotlin.
- Pada baris 3-8, **import** berfungsi untuk mengimpor package atau kelas untuk mengakses komponen dan fungsi dari android SDK dan jetpack compose.
- Pada baris 10, **interface** merupakan antarmukan yang akan digunakan oleh retrofit untuk membuat implementasi API TMDB
- Pada baris 11 dan 19, **@GET** berfungsi untuk menentukan endpoint dan method HTTP untuk mencari film
- Pada baris 12 dan 20, **suspend fun** merupakan fungsi suspend untuk melakukan panggilan API
- Pada baris 13-22, **@Query** merupakan parameter untuk pencarian film sesuai yang dinginkan
- Pada baris 21, **@Path** merupakan parameter untuk menggantikan placeholder di URL

12. data/api/RetrofitClient.kt

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

D. Tautan Git

Berikut adalah tautan untuk source code yang telah dibuat.
https://github.com/AppleCraft2005/kuliah-santuy/tree/main/semesterIV/Pemrograman-Mobile