LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 5

CONNECT TO THE INTERNET

Oleh:

Aiko Anatasha Wendiono NIM. 2310817320013

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
JUNI 2025

LEMBAR PENGESAHAN
LAPORAN PRAKTIKUM PEMROGRAMAN I
MODUL 1

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

Nama Praktikan	: Aiko Anatasha Wendiono
NIM	: 2310817320013

Menyetujui,	Mengetahui,
Asisten Praktikum	Dosen Penanggung Jawab Praktikum
Zulfa Auliya Akbar	Muti`a Maulida S.Kom M.T.I
NIM. 2210817210026	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. ThemePreferenceManager/datastore/data

```
package com.example.mladventure.data.datastore

import android.content.Context
import
androidx.datastore.preferences.core.booleanPreference
sKey
import androidx.datastore.preferences.core.edit
import
androidx.datastore.preferences.preferencesDataStore
import kotlinx.coroutines.flow.Flow
import kotlinx.coroutines.flow.map

private val Context.dataStore by
preferencesDataStore(name = "settings")

class ThemePreferenceManager(private val context:
Context) {
    companion object {
        private val DARK_MODE_KEY =
booleanPreferencesKey("dark_mode")
    }

    val isDarkMode: Flow<Boolean> =
context.dataStore.data.map { prefs ->
        prefs[DARK_MODE_KEY] ?: false
    }

    suspend fun setDarkMode(enabled: Boolean) {
        context.dataStore.edit { prefs ->
            prefs[DARK_MODE_KEY] = enabled
        }
    }
}
```

Tabel 1. Source Code Jawaban Soal 1

2. CharacterDao/local/data

```
package com.example.mladventure.data.local

import androidx.room.*
import kotlinx.coroutines.flow.Flow

@Dao
interface CharacterDao {
    @Query("SELECT * FROM characters")
    fun getAll(): Flow<List<CharacterEntity>>

    @Insert(onConflict = OnConflictStrategy.REPLACE)
    suspend fun insertAll(characters: 
List<CharacterEntity>)

    @Update
    suspend fun update(character: CharacterEntity)
}
```

Tabel 2. Source Code Jawaban Soal 1

3. CharacterDatabase/local/data

```
package com.example.mladventure.data.local

import android.content.Context
import androidx.room.Database
import androidx.room.Room
import androidx.room.RoomDatabase

@Database(entities = [CharacterEntity::class], version
= 1)
abstract class CharacterDatabase : RoomDatabase() {
    abstract fun characterDao(): CharacterDao

    companion object {
        @Volatile      private      var      INSTANCE:
CharacterDatabase? = null

        fun       getInstance(context:       Context):
CharacterDatabase =
            INSTANCE ?: synchronized(this) {
                Room.databaseBuilder(
                    context.applicationContext,
                    CharacterDatabase::class.java,
                    "character_database"
                ).build().also { INSTANCE = it }
            }
    }
}
```

Tabel 3. Source Code Jawaban Soal 1

4. CharacterEntity/local/data

```kotlin
package com.example.mladventure.data.local

import androidx.room.Entity
import androidx.room.PrimaryKey

@Entity(tableName = "characters")
data class CharacterEntity(
    @PrimaryKey val name: String,
    val alias: String,
    val imageUrl: String,
    val description: String,
    val wikiUrl: String,
    val isFavorite: Boolean = false
)
```

Tabel 4. Source Code Jawaban Soal 1

5. ApiResponse/remote/data

```kotlin
package com.example.mladventure.data.remote

sealed class ApiResponse<out T> {
    data class Success<T>(val data: T) :
ApiResponse<T>()
    data class Error(val message: String) :
ApiResponse<Nothing>()
    object Loading : ApiResponse<Nothing>()
}
```

Tabel 5. Source Code Jawaban Soal 1

6. CharacterApi/remote/data

```
package com.example.mladventure.data.remote

import retrofit2.http.GET

interface CharacterApi {
    @GET("characters")
    suspend fun getCharacters(): List<CharacterDto>

    companion object {
        const          val          BASE_URL          =
"https://mlacharacters.free.beeceptor.com/data"
    }
}
```

Tabel 6. Source Code Jawaban Soal 1

7. **CharacterDto/remote/data**

```
package com.example.mladventure.data.remote

import kotlinx.serialization.Serializable

@Serializable
data class CharacterDto(
    val name: String,
    val alias: String,
    val imageUrl: String,
    val description: String,
    val wikiUrl: String
)
```

Tabel 7. Source Code Jawaban Soal 1

8. **CharacterRepository/repository**

```kotlin
package com.example.mladventure.repository

import
com.example.mladventure.data.local.CharacterDao
import
com.example.mladventure.data.local.CharacterEntity
import
com.example.mladventure.data.remote.CharacterApi
import kotlinx.coroutines.flow.Flow

class CharacterRepository(
    private val api: CharacterApi,
    private val dao: CharacterDao
) {
    val characters: Flow<List<CharacterEntity>> =
dao.getAll()

    suspend fun refreshCharacters() {
        try {
            val dtoList = api.getCharacters()
            val entities = dtoList.map {
                CharacterEntity(
                    name = it.name,
                    alias = it.alias,
                    imageUrl = it.imageUrl,
                    description = it.description,
                    wikiUrl = it.wikiUrl
                )
            }
            dao.insertAll(entities)
        } catch (e: Exception) {
            e.printStackTrace()
        }
    }

    suspend        fun        toggleFavorite(character:
CharacterEntity) {
        dao.update(character.copy(isFavorite         =
!character.isFavorite))
    }
}
```

Tabel 8. Source Code Jawaban Soal 1

9. SettingScreen.kt/settings

```
package com.example.mladventure.settings

import androidx.compose.foundation.layout.*
import androidx.compose.material3.*
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.unit.dp

@Composable
fun SettingsScreen(
    isDarkMode: Boolean,
    onToggleTheme: (Boolean) -> Unit
) {
    Column(
        modifier = Modifier
            .fillMaxSize()
            .padding(24.dp),
        verticalArrangement = Arrangement.Center,
        horizontalAlignment =
Alignment.CenterHorizontally
    ) {
        Text("Dark Mode", style =
MaterialTheme.typography.titleMedium)
        Spacer(modifier = Modifier.height(8.dp))
        Switch(
            checked = isDarkMode,
            onCheckedChange = onToggleTheme
        )
    }
}
```

Tabel 9. Source Code Jawaban Soal 1

10. CharacterDetailScreen.kt/ui.theme

```
package com.example.mladventure.ui.theme

import androidx.compose.foundation.Image
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.lazy.LazyColumn
import androidx.compose.material.icons.Icons
import
androidx.compose.material.icons.filled.ArrowBack
import androidx.compose.material3.*
import androidx.compose.runtime.Composable
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.layout.ContentScale
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import androidx.navigation.NavController
import coil.compose.rememberAsyncImagePainter
import
com.example.mladventure.data.local.CharacterEntity


@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun CharacterDetailScreen(character: CharacterEntity,
navController: NavController) {
    Scaffold(
        containerColor = Color(0xFF1C1C1E),
        topBar = {
            CenterAlignedTopAppBar(
                title = {},
                navigationIcon = {
                    IconButton(onClick = { navController.popBackStack() }) {
                        Icon(
                            imageVector = Icons.Default.ArrowBack,
                            contentDescription = "Back",
                            tint = Color.White
                        )
                    }
                },
                colors = TopAppBarDefaults.centerAlignedTopAppBarColors(
                    containerColor = Color.Transparent
                )
            )
        }
    ) { innerPadding ->
        LazyColumn(
            modifier = Modifier
                .padding(innerPadding)
                .fillMaxSize()
        ) {
            item {
                Image(
                    painter = rememberAsyncImagePainter(character.imageUrl),
                    contentDescription = character.name,
                    contentScale = ContentScale.Crop,
                    modifier = Modifier
                        .fillMaxWidth()
                        .height(500.dp)
                )
            }

            item {
                Column(modifier                                     =
Modifier.padding(16.dp)) {
                    Row(
                        modifier                                    =
Modifier.fillMaxWidth(),
                        horizontalArrangement                       =
Arrangement.SpaceBetween
                    ) {
                        Text(
                            text = character.name,
                            color = Color.White,
                            style                                   =
MaterialTheme.typography.titleLarge
                        )
                        Text(
                            text = "2016",
                            color = Color.LightGray,
                            style                                   =
MaterialTheme.typography.bodyMedium
                        )
                    }

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

```

                    Text(
                        text = "Deskripsi:",
                        color = Color.White,
                        fontWeight                 =
androidx.compose.ui.text.font.FontWeight.Bold,
                        style                      =
MaterialTheme.typography.bodyMedium
                    )

                    Spacer(modifier                =
Modifier.height(4.dp))

                    Text(
                        text = character.description,
                        color = Color(0xFFCCCCCC),
                        style                      =
MaterialTheme.typography.bodySmall,
                        lineHeight = 20.sp
                    )
                }
            }
        }
    }
}
```

Tabel 10. Source Code Jawaban Soal 1

11. CharacterViewModel/ui.theme

```
package com.example.mladventure.ui.theme

import android.util.Log
import androidx.lifecycle.ViewModel
import androidx.lifecycle.viewModelScope
import com.example.mladventure.Character
import kotlinx.coroutines.flow.MutableStateFlow
import kotlinx.coroutines.flow.StateFlow
import kotlinx.coroutines.launch

class CharacterViewModel : ViewModel() {

    private        val        _characterList        =
MutableStateFlow<List<Character>>(emptyList())
    val  characterList:  StateFlow<List<Character>>  =
_characterList

    fun setCharacterList(characters: List<Character>)
{
        viewModelScope.launch {
            Log.d("CharacterViewModel",   "Data   item
masuk ke dalam list")
            _characterList.value = characters
        }
    }

    fun  logItemClick(character:  Character,  action:
String) {
        Log.d("CharacterViewModel",  "Tombol  $action
ditekan untuk ${character.name}")
    }

    fun logNavigateDetail(character: Character) {
        Log.d("CharacterViewModel",    "Navigasi    ke
detail karakter: ${character.name}")
    }
}
```

Tabel 11. Source Code Jawaban Soal 1

12. CharacterViewModelFactory/ui.theme

```
package com.example.mladventure.ui.theme


import androidx.lifecycle.ViewModel
import androidx.lifecycle.ViewModelProvider

class           CharacterViewModelFactory            :
ViewModelProvider.Factory {
    override  fun  <T  :  ViewModel>  create(modelClass:
Class<T>): T {
        if
(modelClass.isAssignableFrom(CharacterViewModel::clas
s.java)) {
            return CharacterViewModel() as T
        }
        throw       IllegalArgumentException("Unknown
ViewModel class")
    }
}
```

Tabel 12. Source Code Jawaban Soal 1

13. Character

```kotlin
package com.example.mladventure

data class Character(
    val name: String,
    val alias: String,
    val imageRes: Int,
    val description: String,
    val wikiUrl: String
)
```

Tabel 13. Source Code Jawaban Soal 1

14. CharacterListScreen.kt

```
package com.example.mladventure

import android.content.Intent
import android.net.Uri
import androidx.compose.foundation.Image
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.lazy.LazyColumn
import androidx.compose.foundation.lazy.items
import
androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.material3.*
import androidx.compose.runtime.Composable
import androidx.compose.runtime.collectAsState
import androidx.compose.runtime.getValue
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.layout.ContentScale
import androidx.compose.ui.platform.LocalContext
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.unit.dp
import androidx.navigation.NavController
import
com.example.mladventure.ui.theme.CharacterViewModel

@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun CharacterListScreen(
    navController: NavController,
    viewModel: CharacterViewModel
```

```
) {
    val characters by viewModel.characterList.collectAsState()
    val context = LocalContext.current


    LazyColumn(
        modifier = Modifier
            .fillMaxSize()
            .padding(8.dp),
        contentPadding = PaddingValues(top = 60.dp),
        verticalArrangement = Arrangement.spacedBy(12.dp)
    ) {
        items(characters) { character ->
            Card(
                shape = RoundedCornerShape(12.dp),
                colors = CardDefaults.cardColors(containerColor = Color(0xFF2B2B2B)),
                elevation = CardDefaults.cardElevation(4.dp),
                modifier = Modifier.fillMaxWidth()
            ) {
                Row(modifier = Modifier.padding(12.dp)) {
                    Image(
                        painter = painterResource(id = character.imageRes),
                        contentDescription = character.name,
                        contentScale                =
ContentScale.Crop,
                        modifier = Modifier
                            .width(90.dp)
                            .height(140.dp)
                            .padding(end = 12.dp)
                    )

                    Column(modifier                  =
Modifier.weight(1f)) {
                        Row(
                            modifier                =
Modifier.fillMaxWidth(),
                            horizontalArrangement   =
Arrangement.SpaceBetween
                        ) {
                            Text(
                                text                =
character.name,
                                style               =
MaterialTheme.typography.titleMedium,
                                color = Color.White,
                                fontWeight          =
FontWeight.Bold
                            )
                            Text(
                                text   =   "2016",   //
Masih dummy, bisa pakai data jika tersedia
                                style               =
MaterialTheme.typography.labelMedium,

                                color                =
Color.LightGray
                            )
                        }

                        Spacer(modifier              =
Modifier.height(6.dp))

                        Row(
                            modifier                 =
Modifier.fillMaxWidth(),
                            verticalAlignment        =
Alignment.Top
                        ) {
                            Text(
                                text = "Deskripsi: ",
                                color = Color.White,
                                fontWeight           =
FontWeight.Bold,
                                style                =
MaterialTheme.typography.bodyMedium
                            )

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

                            Text(
                                text                 =
character.description,
                                color = Color.White,

                                style                =
MaterialTheme.typography.bodySmall,
                                modifier             =
Modifier.weight(1f)
                            )
                        }

                        Spacer(modifier              =
Modifier.height(10.dp))

                        Row(horizontalArrangement    =
Arrangement.spacedBy(8.dp)) {
                            Button(
                                onClick = {

viewModel.logItemClick(character, "Detail Hero")
                                    val    intent    =
Intent(Intent.ACTION_VIEW,
Uri.parse(character.wikiUrl))

context.startActivity(intent)
                                },
                                colors               =
ButtonDefaults.buttonColors(containerColor           =
Color(0xFF9BB1EB)),
                                modifier             =
Modifier.weight(1f)
                            ) {
                                Text("Detail    Hero",
color = Color.White)
                            }

                            Button(
                                onClick = {
                                    viewModel.logItemClick(character, "Deskripsi")
                                    navController.navigate("detail/${character.name}")
                                },
                                colors = ButtonDefaults.buttonColors(containerColor = Color(0xFF9BB1EB)),
                                modifier = Modifier.weight(1f)
                            ) {
                                Text("Deskripsi", color = Color.White)
                            }
                        }
                    }
                }
            }
        }
    }
}
```

Tabel 14. Source Code Jawaban Soal 1

15. MainActivity

```
package com.example.mladventure

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.material3.MaterialTheme
import androidx.compose.material3.Surface
import androidx.lifecycle.viewmodel.compose.viewModel
import
androidx.navigation.compose.rememberNavController
import
com.example.mladventure.ui.theme.CharacterViewModel
import
com.example.mladventure.ui.theme.CharacterViewModelFa
ctory
import
com.example.mladventure.ui.theme.MLACharactersTheme
import
com.example.mladventure.data.datastore.ThemePreferenc
eManager
import kotlinx.coroutines.flow.first
import androidx.compose.runtime.*
import kotlinx.coroutines.runBlocking

class MainActivity : ComponentActivity() {
    private      lateinit      var      themeManager:
ThemePreferenceManager

    override fun onCreate(savedInstanceState: Bundle?)
{
        super.onCreate(savedInstanceState)
        themeManager =
ThemePreferenceManager(applicationContext)

        setContent {
            val isDarkMode by
themeManager.isDarkMode.collectAsState(initial =
false)

            MLACharactersTheme(darkTheme =
isDarkMode) {
                Surface(color =
MaterialTheme.colorScheme.background) {
                    val navController =
rememberNavController()
                    val viewModel: CharacterViewModel
= viewModel(
                        factory =
CharacterViewModelFactory()
                    )

                    NavGraph(
                        navController =
navController,
                        viewModel = viewModel,
                        isDarkMode = isDarkMode,
                        onToggleTheme = { enabled ->
                            runBlocking {

themeManager.setDarkMode(enabled)
                            }
                        }
                    )
                }
            }
        }
    }
}
```

Tabel 15. Source Code Jawaban Soal 1

16. MLACharacterApplication.kt

```kotlin
package com.example.mladventure

import kotlinx.serialization.json.Json
import okhttp3.MediaType
import okhttp3.MediaType.Companion.toMediaType
import okhttp3.RequestBody
import okhttp3.ResponseBody
import retrofit2.Converter
import retrofit2.Retrofit
import java.lang.reflect.Type
import kotlinx.serialization.serializer
import kotlinx.serialization.encodeToString
import kotlinx.serialization.decodeFromString
import okhttp3.RequestBody.Companion.toRequestBody

class JsonConverterFactory(
    private val json: Json,
    private val contentType: MediaType
) : Converter.Factory() {

    companion object {
        fun create(
            json: Json = Json { ignoreUnknownKeys =
true },
            contentType: MediaType =
"application/json".toMediaType()
        ): JsonConverterFactory =
JsonConverterFactory(json, contentType)
    }

    override fun responseBodyConverter(
        type: Type,
        annotations: Array<Annotation>,
        retrofit: Retrofit
    ): Converter<ResponseBody, *> {
        val                serializer                =
json.serializersModule.serializer(type)
        return Converter { body ->
            json.decodeFromString(serializer,
body.string())
        }
    }

    override fun requestBodyConverter(
        type: Type,
        parameterAnnotations: Array<Annotation>,
        methodAnnotations: Array<Annotation>,
        retrofit: Retrofit
    ): Converter<*, RequestBody> {
        val                serializer                =
json.serializersModule.serializer(type)
        return Converter<Any, RequestBody> { value ->
            val                content                =
json.encodeToString(serializer, value)
            content.toRequestBody(contentType)
        }
    }
}
```

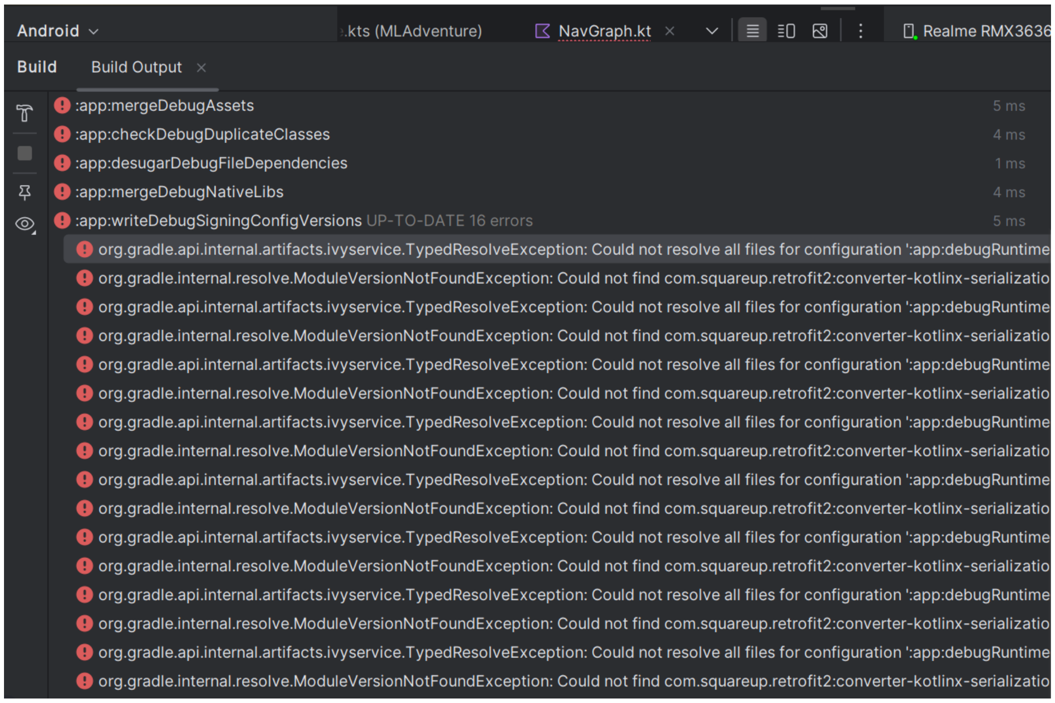
Tabel 16. Source Code Jawaban Soal 1

17. NavGraph.kt

```kotlin
package com.example.mladventure

import androidx.compose.runtime.Composable
import androidx.compose.runtime.collectAsState
import androidx.navigation.NavHostController
import androidx.navigation.compose.NavHost
import androidx.navigation.compose.composable
import
com.example.mladventure.ui.theme.CharacterViewModel
import com.example.mladventure.CharacterListScreen
import
com.example.mladventure.ui.theme.CharacterDetailScree
n
import
com.example.mladventure.settings.SettingsScreen

@Composable
fun NavGraph(
    navController: NavHostController,
    viewModel: CharacterViewModel,
    isDarkMode: Boolean,
    onToggleTheme: (Boolean) -> Unit
) {
    val charactersState =
viewModel.characterList.collectAsState()
    val characters = charactersState.value

    NavHost(navController = navController,
startDestination = "list") {
        composable("list") {
            CharacterListScreen(
                viewModel = viewModel,
                navController = navController
            )
        }

        composable("detail/{name}") { backStackEntry ->
            val name = backStackEntry.arguments?.getString("name")
            val character = characters.find { it.name == name }
            character?.let {
                CharacterDetailScreen(
                    character = it,
                    navController = navController
                )
            }
        }

        composable("settings") {
            SettingsScreen(
                isDarkMode = isDarkMode,
                onToggleTheme = onToggleTheme
            )
        }
    }
}
```

Tabel 17. Source Code Jawaban Soal 1

B. Output Program



```
Android ∨                    .kts (MLAdventure)    NavGraph.kt ×                    Realme RMX3636
Build    Build Output ×
:app:mergeDebugAssets                                                                    5 ms
:app:checkDebugDuplicateClasses                                                          4 ms
:app:desugarDebugFileDependencies                                                        1 ms
:app:mergeDebugNativeLibs                                                                4 ms
:app:writeDebugSigningConfigVersions UP-TO-DATE 16 errors                                5 ms
  org.gradle.api.internal.artifacts.ivyservice.TypedResolveException: Could not resolve all files for configuration ':app:debugRuntime
  org.gradle.internal.resolve.ModuleVersionNotFoundException: Could not find com.squareup.retrofit2:converter-kotlinx-serializatio
  org.gradle.api.internal.artifacts.ivyservice.TypedResolveException: Could not resolve all files for configuration ':app:debugRuntime
  org.gradle.internal.resolve.ModuleVersionNotFoundException: Could not find com.squareup.retrofit2:converter-kotlinx-serializatio
  org.gradle.api.internal.artifacts.ivyservice.TypedResolveException: Could not resolve all files for configuration ':app:debugRuntime
  org.gradle.internal.resolve.ModuleVersionNotFoundException: Could not find com.squareup.retrofit2:converter-kotlinx-serializatio
  org.gradle.api.internal.artifacts.ivyservice.TypedResolveException: Could not resolve all files for configuration ':app:debugRuntime
  org.gradle.internal.resolve.ModuleVersionNotFoundException: Could not find com.squareup.retrofit2:converter-kotlinx-serializatio
  org.gradle.api.internal.artifacts.ivyservice.TypedResolveException: Could not resolve all files for configuration ':app:debugRuntime
  org.gradle.internal.resolve.ModuleVersionNotFoundException: Could not find com.squareup.retrofit2:converter-kotlinx-serializatio
  org.gradle.api.internal.artifacts.ivyservice.TypedResolveException: Could not resolve all files for configuration ':app:debugRuntime
  org.gradle.internal.resolve.ModuleVersionNotFoundException: Could not find com.squareup.retrofit2:converter-kotlinx-serializatio
  org.gradle.api.internal.artifacts.ivyservice.TypedResolveException: Could not resolve all files for configuration ':app:debugRuntime
  org.gradle.internal.resolve.ModuleVersionNotFoundException: Could not find com.squareup.retrofit2:converter-kotlinx-serializatio
  org.gradle.api.internal.artifacts.ivyservice.TypedResolveException: Could not resolve all files for configuration ':app:debugRuntime
  org.gradle.internal.resolve.ModuleVersionNotFoundException: Could not find com.squareup.retrofit2:converter-kotlinx-serializatio
```

Gambar 1. Screenshot Hasil Jawaban Soal 1

C. Pembahasan

1. **ThemePreferenceManager/datastore/data**

   `ThemePreferenceManager` merupakan kelas yang mengelola penyimpanan preferensi tampilan tema (dark mode) pada aplikasi menggunakan Jetpack DataStore. Dengan memanfaatkan `Flow`, aplikasi dapat merespons perubahan tema secara real-time berdasarkan nilai boolean yang disimpan pada DataStore.

2. **CharacterDao/local/data**

   `CharacterDao` adalah antarmuka yang berfungsi sebagai jembatan antara aplikasi dan database lokal Room. DAO ini menyediakan fungsi untuk mengambil seluruh data karakter, menyimpan daftar karakter, dan memperbarui data karakter, termasuk status favorit, dengan operasi berbasis *suspend* dan *Flow*.

3. **CharacterDatabase/local/data**

CharacterDatabase merupakan kelas abstrak yang memperluas RoomDatabase dan bertanggung jawab menginisialisasi instance database Room serta menyediakan akses ke CharacterDao.

4. **CharacterEntity/local/data**

   CharacterEntity adalah data class yang merepresentasikan struktur tabel dalam Room database dengan anotasi @Entity. Kelas ini berisi atribut seperti name, alias, imageUrl, description, wikiUrl, dan isFavorite yang mewakili data karakter yang disimpan secara lokal.

5. **ApiResponse/remote/data**

   ApiResponse adalah kelas sealed yang mendefinisikan status respons dari API. Ada tiga kemungkinan status yaitu Success, Error, dan Loading. Struktur ini digunakan untuk memudahkan pengelolaan data dan penanganan error saat melakukan permintaan ke API.

6. **CharacterApi/remote/data**

   CharacterApi adalah antarmuka Retrofit yang mendeklarasikan endpoint API untuk mengambil data karakter dari internet. Dengan anotasi @GET, aplikasi dapat mengambil data berupa daftar karakter (CharacterDto) dari endpoint yang telah ditentukan.

7. **CharacterDto/remote/data**

   CharacterDto adalah data transfer object (DTO) yang digunakan untuk menyesuaikan struktur data yang diterima dari API. Dengan anotasi @Serializable, objek ini bisa dikonversi langsung dari dan ke format JSON menggunakan Kotlinx Serialization.

8. **CharacterRepository/repository**

   CharacterRepository bertugas sebagai perantara antara data sumber (API dan database lokal) dan lapisan presentasi (UI).

9. **SettingScreen.kt/settings**

   SettingScreen.kt merupakan komponen UI berbasis Jetpack Compose yang menampilkan pengaturan tema.

10. **CharacterDetailScreen.kt/ui.theme**

`CharacterDetailScreen.kt` adalah layar yang menampilkan detail lengkap dari karakter yang dipilih.

11. **CharacterViewModel/ui.theme**

    `CharacterViewModel` merupakan kelas ViewModel yang bertanggung jawab menyimpan state daftar karakter dan menangani log interaksi pengguna. Data disimpan dalam bentuk `StateFlow`, memungkinkan UI untuk memperbarui tampilan secara otomatis saat data berubah.

12. **CharacterViewModelFactory/ui.theme**

    `CharacterViewModelFactory` adalah pabrik ViewModel yang digunakan untuk membuat instance `CharacterViewModel.`

13. **Character**

    `Character` adalah model data biasa (plain Kotlin object) yang digunakan untuk menampilkan data karakter dalam antarmuka pengguna. Objek ini berbeda dari `CharacterEntity` atau `CharacterDto` dan biasanya digunakan di lapisan presentasi agar tidak terikat langsung dengan data dari sumber lokal atau remote.

14. **CharacterListScreen.kt**

    `CharacterListScreen.kt` menampilkan daftar karakter dalam bentuk list.

15. **MainActivity**

    `MainActivity` adalah titik masuk utama aplikasi. Di sini dilakukan pengaturan tema berdasarkan `ThemePreferenceManager`, inisialisasi `ViewModel`, serta pemanggilan fungsi `NavGraph` untuk menavigasi antar layar dalam aplikasi.

16. **MLACharacterApplication.kt**

    `MLACharacterApplication.kt` berisi konfigurasi kustom `JsonConverterFactory` yang digunakan Retrofit untuk mengonversi data JSON menggunakan Kotlinx Serialization.

17. **NavGraph.kt**

    `NavGraph.kt` mengatur semua rute navigasi dalam aplikasi.

D. Tautan Git

Berikut adalah tautan untuk source code yang telah dibuat.

Laporan-Praktikum-Pemrograman-Mobile/ at main · aikoanatashawendiono/Laporan-Praktikum-Pemrograman-Mobile