LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 5

CONNECT TO THE INTERNET

Oleh:

Damarjati Suryo Laksono **NIM. 2310817210014**

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	: Damarjati Suryo Laksono
NIM	: 2310817210014

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:

1. 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. AppDatabase

```
package com.example.myapi_test

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

@Database(entities = [CharacterInfoEntity::class], version =
1, exportSchema = false)
```

```kotlin
abstract class AppDatabase : RoomDatabase() {

    abstract fun characterDao(): CharacterDao

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

        fun getDatabase(context: Context): AppDatabase {
            return INSTANCE ?: synchronized(this) {
                val instance = Room.databaseBuilder(
                    context.applicationContext,
                    AppDatabase::class.java,
                    "character_database"
                ).build()
                INSTANCE = instance
                instance
            }
        }
    }
}
```

Tabel 1. 1 Source Code AppDatabase.kt

2. CacheMapper.kt

```kotlin
package com.example.myapi_test

fun CharacterInfo.toCharacterInfoEntity():
CharacterInfoEntity {
    return CharacterInfoEntity(
        characterId = this.characterId,
        characterName = this.characterName,
        characterUrl = this.characterUrl,
        characterImageUrl = this.characterImageUrl,
        japaneseVoiceActor = this.japaneseVoiceActor,
        englishVoiceActor = this.englishVoiceActor,
        favorites = this.favorites
    )
}

fun CharacterInfoEntity.toCharacterInfo(): CharacterInfo {
    return CharacterInfo(
        characterId = this.characterId,
        characterName = this.characterName,
        characterUrl = this.characterUrl,
        characterImageUrl = this.characterImageUrl,
        japaneseVoiceActor = this.japaneseVoiceActor,
        englishVoiceActor = this.englishVoiceActor,
        favorites = this.favorites
    )
}

```

Tabel 1. 2 Source Code CacheMapper.kt

3. **CharacterAdapter.kt**

```
package com.example.myapi_test

import android.content.Intent
import android.net.Uri
import android.view.LayoutInflater
import android.view.ViewGroup
import androidx.recyclerview.widget.RecyclerView
import com.bumptech.glide.Glide
import com.example.myapi_test.databinding.ItemListBinding
import java.text.NumberFormat
import java.util.Locale

class CharacterAdapter(
```

```kotlin
    private var characters: List<CharacterInfo>,
    private val onDetailClick: (CharacterInfo) -> Unit
) :
RecyclerView.Adapter<CharacterAdapter.CharacterViewHolder>()
{

    inner class CharacterViewHolder(val binding:
ItemListBinding) :
        RecyclerView.ViewHolder(binding.root) {

        fun bind(character: CharacterInfo) {
            binding.apply {
                textViewCharacterName.text =
character.characterName
                val formattedFavorites =
NumberFormat.getNumberInstance(Locale.US).format(character.f
avorites)
                textViewFavorites.text = formattedFavorites

                val vaJapaneseText = "JP:
${character.japaneseVoiceActor ?: "N/A"}"
                val vaEnglishText = "EN:
${character.englishVoiceActor ?: "N/A"}"
                textViewVoiceActors.text =
"$vaJapaneseText\n$vaEnglishText"

                Glide.with(imageViewCharacter.context)
                    .load(character.characterImageUrl)
                    .into(imageViewCharacter)

                buttonDetail.setOnClickListener {
                    onDetailClick(character)
                }

                buttonUrl.setOnClickListener {
                    val intent = Intent(Intent.ACTION_VIEW,
Uri.parse(character.characterUrl))

imageViewCharacter.context.startActivity(intent)
                }
            }
        }
    }

    override fun onCreateViewHolder(parent: ViewGroup,
viewType: Int): CharacterViewHolder {
        val binding =
ItemListBinding.inflate(LayoutInflater.from(parent.context),
parent, false)
        return CharacterViewHolder(binding)
    }

    override fun getItemCount() = characters.size

    override fun onBindViewHolder(holder:
CharacterViewHolder, position: Int) {
        holder.bind(characters[position])
    }

    fun setData(newCharacters: List<CharacterInfo>) {
        characters = newCharacters
        notifyDataSetChanged()
    }
}
```

Tabel 1. 3 Source Code CharacterAdapter.kt

4. CharacterDao

```kotlin
package com.example.myapi_test

import androidx.room.Dao
import androidx.room.Insert
import androidx.room.OnConflictStrategy
import androidx.room.Query
import kotlinx.coroutines.flow.Flow

@Dao
interface CharacterDao {

    @Query("SELECT * FROM characters")
    fun getCharacters(): Flow<List<CharacterInfoEntity>>

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

    @Query("DELETE FROM characters")
    suspend fun clearAll()
}
```

Tabel 1. 4 Source Code CharacterDao.kt

5. CharacterInfoEntity

```kotlin
package com.example.myapi_test

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

@Entity(tableName = "characters")
data class CharacterInfoEntity(
    @PrimaryKey
    val characterId: Int,
    val characterName: String,
    val characterUrl: String,
    val characterImageUrl: String,
    val japaneseVoiceActor: String?,
    val englishVoiceActor: String?,
    val favorites: Int
)
```

Tabel 1. 5 Source Code CharacterInfoEntity.kt

6. CharacterMapper.kt

```
package com.example.myapi_test

fun mapToCharacterInfoList(characterDataList:
List<CharacterListItem>?): List<CharacterInfo> {
    return characterDataList?.map { data ->
        CharacterInfo(
            characterId = data.character.malId,
            characterName = data.character.name ?: "Name not
found",
            characterUrl = data.character.url,
            characterImageUrl =
data.character.images?.jpg?.imageUrl ?: "",

            japaneseVoiceActor = data.voices?.find {
it.language.equals("Japanese", ignoreCase = true)
}?.person?.name,
            englishVoiceActor = data.voices?.find {
it.language.equals("English", ignoreCase = true)
}?.person?.name,
            favorites = data.favorites
        )
    } ?: emptyList()
}
```

Tabel 1. 6 Source Code CharacterMapper.kt

7. CharacterModels.kt

```kotlin
package com.example.myapi_test
import android.os.Parcelable
import kotlinx.parcelize.Parcelize

import com.google.gson.annotations.SerializedName

@Parcelize
data class CharacterInfo(
    val characterId: Int,
    val characterName: String,
    val characterUrl: String,
    val characterImageUrl: String,
    val japaneseVoiceActor: String?,
    val englishVoiceActor: String?,
    val favorites: Int
) : Parcelable

data class AnimeCharactersResponse(
    @SerializedName("data")
    val data: List<CharacterListItem>
)

data class Character(
    @SerializedName("mal_id")
    val malId: Int,

    @SerializedName("url")
    val url: String,

    @SerializedName("images")
    val images: Images?,

    @SerializedName("name")
    val name: String?
)

data class CharacterListItem(
    @SerializedName("character")
    val character: Character,

    @SerializedName("role")
    val role: String,

    @SerializedName("favorites")
    val favorites: Int,

    @SerializedName("voice_actors")
    val voices: List<Voice>?
)

data class CharacterDetailResponse(
    @SerializedName("data")
    val data: CharacterDetails
)

data class CharacterDetails(
    @SerializedName("mal_id")
    val malId: Int,

    @SerializedName("url")
    val url: String,

    @SerializedName("images")
    val images: Images,

    @SerializedName("name")
    val name: String,

    @SerializedName("name_kanji")
    val nameKanji: String?,

    @SerializedName("favorites")
    val favorites: Int,

    @SerializedName("about")
    val about: String?,

    @SerializedName("voices")
    val voices: List<Voice>
)

data class Voice(
    @SerializedName("person")
    val person: Person,
    @SerializedName("language")
    val language: String
)

data class Person(
    @SerializedName("mal_id")
    val malId: Int,
    @SerializedName("url")
    val url: String,
    @SerializedName("images")
    val images: PersonImages,
    @SerializedName("name")
    val name: String
)

data class Images(
    @SerializedName("jpg")
    val jpg: ImageType
)

data class ImageType(
    @SerializedName("image_url")
    val imageUrl: String
)

data class PersonImages(
    @SerializedName("jpg")
    val jpg: ImageType
)

```

Tabel 1. 7 Source Code CharacterModel.kt

8. CharacterRepository

```
package com.example.myapi_test

import kotlinx.coroutines.flow.Flow
import kotlinx.coroutines.flow.map

class CharacterRepository(
    private val apiService: JikanApiService,
    private val characterDao: CharacterDao
) {

    fun getAnimeCharacters(): Flow<List<CharacterInfo>> {
        return characterDao.getCharacters().map { entities -
>
            entities.map { it.toCharacterInfo() }
        }
    }

    suspend fun refreshCharacters(animeId: Int) {
```

```kotlin
        val response =
apiService.getAnimeCharacters(animeId)
        val characterInfoList =
mapToCharacterInfoList(response.data)

        characterDao.clearAll()
        characterDao.insertAll(characterInfoList.map {
it.toCharacterInfoEntity() })
    }

    suspend fun getCharacterDetails(characterId: Int):
CharacterDetailResponse {
        return apiService.getCharacterDetails(characterId)
    }
}

```

Tabel 1. 8 Source Code CharacterRepository.kt

9. CharacterViewModelFactory

```kotlin
package com.example.myapi_test

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

class CharacterViewModelFactory(private val repository:
CharacterRepository) : ViewModelProvider.Factory {

    override fun <T : ViewModel> create(modelClass:
Class<T>): T {
        return when {

modelClass.isAssignableFrom(HomeViewModel::class.java) -> {
                HomeViewModel(repository) as T
            }

modelClass.isAssignableFrom(DetailViewModel::class.java) ->
{
                DetailViewModel(repository) as T
            }
            else -> throw IllegalArgumentException("Unknown
ViewModel class: ${modelClass.name}")
        }
    }
}

```
Tabel 1. 9 Source Code CharacterViewModelFactory.kt

10. DetailFragment

```
package com.example.myapi_test

import android.content.Intent
import android.net.Uri
import android.os.Bundle
import android.util.Log
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import androidx.fragment.app.Fragment
import androidx.lifecycle.ViewModelProvider
import androidx.navigation.fragment.navArgs
import com.bumptech.glide.Glide
import
com.example.myapi_test.databinding.FragmentDetailBinding
import java.text.NumberFormat
import java.util.Locale

class DetailFragment : Fragment() {

    private var _binding: FragmentDetailBinding? = null
    private val binding get() = _binding!!
    private val args: DetailFragmentArgs by navArgs()

    private lateinit var viewModel: DetailViewModel

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View {
        _binding = FragmentDetailBinding.inflate(inflater,
container, false)
        return binding.root
    }

    override fun onViewCreated(view: View,
savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)

        val characterDao =
AppDatabase.getDatabase(requireContext()).characterDao()
        val repository =
CharacterRepository(RetrofitInstance.api, characterDao)
        val viewModelFactory =
CharacterViewModelFactory(repository)

        viewModel = ViewModelProvider(this,
viewModelFactory)[DetailViewModel::class.java]

        bindInitialData(args.character)

        setupObservers()

viewModel.fetchCharacterDetails(args.character.characterId)
    }

    private fun bindInitialData(character: CharacterInfo) {
        binding.apply {
            textViewNameDetail.text =
character.characterName
            val formattedFavorites =
NumberFormat.getNumberInstance(Locale.US)
                .format(character.favorites)
            textViewFavoritesDetail.text =
"$formattedFavorites Favorites"
            textViewVoiceActorsList.text = "Loading voice
actors..."

            Glide.with(this@DetailFragment)
                .load(character.characterImageUrl)
                .into(imageViewCharacterDetail)

            buttonViewProfileDetail.setOnClickListener {
                val intent = Intent(Intent.ACTION_VIEW,
Uri.parse(character.characterUrl))
                context?.startActivity(intent)
            }
        }
    }

    private fun setupObservers() {

viewModel.characterDetails.observe(viewLifecycleOwner) {
details ->
            details?.let { bindFullData(it) }
        }

        viewModel.error.observe(viewLifecycleOwner) {
errorMessage ->
            errorMessage?.let {
                binding.textViewVoiceActorsList.text = it
                Log.e("DetailFragment", "Error: $it")
            }
        }
    }

    private fun bindFullData(details: CharacterDetails) {
        binding.apply {
            textViewNameDetail.text = details.name
            val formattedFavorites = 
NumberFormat.getNumberInstance(Locale.US)
                .format(details.favorites)
            textViewFavoritesDetail.text = 
"$formattedFavorites Favorites"

            val voiceActorsText = 
details.voices.joinToString("\n") { voice ->
                "${voice.language}: ${voice.person.name}"
            }

            textViewVoiceActorsList.text = 
voiceActorsText.ifEmpty { "No voice actor information 
available." }
        }
    }

    override fun onDestroyView() {
        super.onDestroyView()
        _binding = null
    }
}
```

Tabel 1. 10 Source CodeDetailFragment.kt

11. DetailViewModel

```
package com.example.myapi_test

import androidx.lifecycle.LiveData
import androidx.lifecycle.MutableLiveData
import androidx.lifecycle.ViewModel
import androidx.lifecycle.viewModelScope
import kotlinx.coroutines.launch
import java.io.IOException

class DetailViewModel(private val repository:
CharacterRepository) : ViewModel() {

    private val _characterDetails =
MutableLiveData<CharacterDetails>()
    val characterDetails: LiveData<CharacterDetails> get() =
_characterDetails

    private val _error = MutableLiveData<String?>()
    val error: LiveData<String?> get() = _error

    fun fetchCharacterDetails(characterId: Int) {
        viewModelScope.launch {
            try {
                val response =
repository.getCharacterDetails(characterId)
                _characterDetails.value = response.data
                _error.value = null //
            } catch (e: IOException) {
                _error.value = "Failed to load details due
to a network error."
            } catch (e: Exception) {
                _error.value = "An unexpected error occurred
while fetching details."
```

```
            }
        }
    }
}
```

Tabel 1. 11 Source Code DetailViewModel

12. HomeFragment

```
package com.example.myapi_test

import android.os.Bundle
import android.util.Log
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.Toast
import androidx.fragment.app.Fragment
import androidx.lifecycle.ViewModelProvider
import androidx.navigation.findNavController
import androidx.recyclerview.widget.LinearLayoutManager
import
com.example.myapi_test.databinding.FragmentHomeBinding

class HomeFragment : Fragment() {

    private var _binding: FragmentHomeBinding? = null
    private val binding get() = _binding!!

    private lateinit var characterAdapter: CharacterAdapter
    private lateinit var viewModel: HomeViewModel

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View {
```

```kotlin
        _binding = FragmentHomeBinding.inflate(inflater,
container, false)
        return binding.root
    }

    override fun onViewCreated(view: View,
savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)

        val characterDao =
AppDatabase.getDatabase(requireContext()).characterDao()

        val repository =
CharacterRepository(RetrofitInstance.api, characterDao)
        val viewModelFactory =
CharacterViewModelFactory(repository)

        viewModel = ViewModelProvider(this,
viewModelFactory)[HomeViewModel::class.java]

        setupRecyclerView()
        setupObservers()
    }

    private fun setupRecyclerView() {
        characterAdapter = CharacterAdapter(emptyList()) {
character ->
            val action =
HomeFragmentDirections.actionHomeFragmentToDetailFragment(ch
aracter)

requireActivity().findNavController(R.id.nav_host_fragment).
navigate(action)
        }
        binding.recyclerView.apply {
            layoutManager = LinearLayoutManager(context)
            adapter = characterAdapter
        }
    }

    private fun setupObservers() {
        viewModel.characters.observe(viewLifecycleOwner) {
characters ->
            if (characters.isNullOrEmpty()) {
                binding.textViewEmptyCache.visibility =
View.VISIBLE
                binding.recyclerView.visibility = View.GONE
            } else {
                binding.textViewEmptyCache.visibility =
View.GONE
                binding.recyclerView.visibility =
View.VISIBLE
                characterAdapter.setData(characters)
            }
        }

        viewModel.isLoading.observe(viewLifecycleOwner) {
isLoading ->
            binding.progressBar.visibility = if (isLoading)
View.VISIBLE else View.GONE
        }

        viewModel.error.observe(viewLifecycleOwner) {
errorMessage ->
            errorMessage?.let {
                Toast.makeText(context, it,
Toast.LENGTH_LONG).show()
                Log.e("HomeFragment", "Error: $it")
            }
        }
```

```kotlin
    }

    override fun onDestroyView() {
        super.onDestroyView()
        _binding = null
    }
}
```

Tabel 1. 12 Source Code HomeFragment.kt

13. HomeViewModel

```kotlin
package com.example.myapi_test

import androidx.lifecycle.LiveData
import androidx.lifecycle.MutableLiveData
import androidx.lifecycle.ViewModel
import androidx.lifecycle.asLiveData
import androidx.lifecycle.viewModelScope
import kotlinx.coroutines.launch
import java.io.IOException

class HomeViewModel(private val repository:
CharacterRepository) : ViewModel() {

    val characters: LiveData<List<CharacterInfo>> =
repository.getAnimeCharacters().asLiveData()

    private val _isLoading = MutableLiveData<Boolean>()
    val isLoading: LiveData<Boolean> get() = _isLoading

    private val _error = MutableLiveData<String?>()
    val error: LiveData<String?> get() = _error

    init {
        refreshCharacterData()
    }

    fun refreshCharacterData() {
        viewModelScope.launch {
            _isLoading.value = true
            try {
                repository.refreshCharacters(52991)
                _error.value = null
            } catch (e: IOException) {
                _error.value = "Network error. Displaying cached data."
            } catch (e: Exception) {
                _error.value = "An unexpected error occurred during refresh."
            } finally {
                _isLoading.value = false
            }
        }
    }
}
```

Tabel 1. 13 Source Code HomeViewModel.kt

14. JikanApiService

```kotlin
package com.example.myapi_test

import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import androidx.navigation.NavController
import androidx.navigation.fragment.NavHostFragment
import androidx.navigation.ui.setupActionBarWithNavController
import androidx.core.splashscreen.SplashScreen.Companion.installSp
```

```
lashScreen
import
com.example.myapi_test.databinding.ActivityMainBinding

class MainActivity : AppCompatActivity() {
    private lateinit var navController: NavController
    private lateinit var binding: ActivityMainBinding

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        installSplashScreen()
        binding =
ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)

        setSupportActionBar(binding.toolbar)

        val navHostFragment = supportFragmentManager
            .findFragmentById(R.id.nav_host_fragment) as
NavHostFragment
        navController = navHostFragment.navController

        setupActionBarWithNavController(navController)
    }

    override fun onSupportNavigateUp(): Boolean {
        return navController.navigateUp() ||
super.onSupportNavigateUp()
    }
}
```

Tabel 1. 14 Source Code JikanApiService.kt

15. MainActivity

```
package com.example.myapi_test

import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import androidx.navigation.NavController
import androidx.navigation.fragment.NavHostFragment
import
androidx.navigation.ui.setupActionBarWithNavController
import
androidx.core.splashscreen.SplashScreen.Companion.installSp
lashScreen
import
com.example.myapi_test.databinding.ActivityMainBinding

class MainActivity : AppCompatActivity() {
    private lateinit var navController: NavController
    private lateinit var binding: ActivityMainBinding

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        installSplashScreen()
        binding =
ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)

        setSupportActionBar(binding.toolbar)

        val navHostFragment = supportFragmentManager
            .findFragmentById(R.id.nav_host_fragment) as
NavHostFragment
        navController = navHostFragment.navController

        setupActionBarWithNavController(navController)
    }

    override fun onSupportNavigateUp(): Boolean {
        return navController.navigateUp() ||
super.onSupportNavigateUp()
    }
}
```

Tabel 1. 15 Source Code MainActivity.kt

16. Retrofit Instance

```
package com.example.myapi_test

import retrofit2.Retrofit
import retrofit2.converter.gson.GsonConverterFactory

object RetrofitInstance {
    private const val BASE_URL =
"https://api.jikan.moe/v4/"

    private val retrofit by lazy {
        Retrofit.Builder()
            .baseUrl(BASE_URL)

.addConverterFactory(GsonConverterFactory.create())
            .build()
    }

    val api: JikanApiService by lazy {
        retrofit.create(JikanApiService::class.java)
    }
}
```

Tabel 1. 16 Source Code RetrofitInstance.kt

17. activity_main.xml

```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=".MainActivity"

    android:background="@color/backgroundColor">

    <com.google.android.material.appbar.MaterialToolbar
        android:id="@+id/toolbar"
        android:layout_width="match_parent"
        android:layout_height="?attr/actionBarSize"
        android:background="?attr/colorPrimary"
        app:titleTextColor="?attr/colorOnPrimary"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent" />

    <androidx.fragment.app.FragmentContainerView
        android:id="@+id/nav_host_fragment"

android:name="androidx.navigation.fragment.NavHostFragment"
        android:layout_width="0dp"
        android:layout_height="0dp"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toBottomOf="@id/toolbar"
        app:layout_constraintBottom_toBottomOf="parent"
        app:defaultNavHost="true"
        app:navGraph="@navigation/nav_graph" />

```

```

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

Tabel 1. 17 Source Code activity_main.xml

18. fragment_detail.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fillViewport="true"
    android:background="@color/backgroundColor"
    tools:context=".DetailFragment">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center_horizontal"
        android:orientation="vertical"
        android:padding="16dp">

        <ImageView
            android:id="@+id/imageViewCharacterDetail"
            android:layout_width="200dp"
            android:layout_height="300dp"

android:contentDescription="@string/character_image_descrip
tion"
            android:scaleType="centerCrop"
            tools:src="@tools:sample/avatars" />

        <TextView
            android:id="@+id/textViewNameDetail"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginTop="16dp"
            android:textSize="24sp"
            android:textStyle="bold"
            android:textColor="@color/textColorPrimary"
            tools:text="Character Name" />

        <TextView
            android:id="@+id/textViewFavoritesDetail"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginTop="8dp"
            android:textSize="16sp"
            android:textColor="@color/textColorSecondary"
            tools:text="123,456 Favorites" />

        <TextView
            android:id="@+id/textViewVoiceActorsList"
            android:layout_width="wrap_content"
android:layout_height="wrap_content"
            android:layout_marginTop="16dp"
            android:textAlignment="center"
            android:textSize="14sp"
            android:textColor="@color/textColorSecondary"
            tools:text="Japanese: VA Name\nEnglish: VA
Name" />

        <Button
            android:id="@+id/buttonViewProfileDetail"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginTop="24dp"
            android:text="@string/view_profile_button" />

```

```
    </LinearLayout>
</ScrollView>
```

Tabel 1. 18 Source Code fragment_detail.xml

19. fragment_home.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=".HomeFragment"

    android:background="@color/backgroundColor">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerView"
        android:layout_width="0dp"
        android:layout_height="0dp"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent" />

    <ProgressBar
        android:id="@+id/progressBar"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:visibility="gone"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        tools:visibility="visible" />

    <TextView
        android:id="@+id/text_view_empty_cache"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="No cached data available. Please
check your network."
        android:visibility="gone"

        android:textColor="@color/textColorSecondary"

        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent" />

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

Tabel 1. 19 Source Code fragment_home.xml

20. item_list.xml

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


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="wrap_content"
    android:layout_margin="8dp"
    app:cardCornerRadius="12dp"
    app:cardElevation="4dp"
    app:cardBackgroundColor="@color/cardBackgroundColor">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:padding="16dp">

        <ImageView
            android:id="@+id/imageViewCharacter"
            android:layout_width="100dp"
            android:layout_height="150dp"
            android:scaleType="centerCrop"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            tools:src="@tools:sample/avatars"

android:contentDescription="@string/character_image_descrip
tion" />

        <TextView
            android:id="@+id/textViewCharacterName"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginStart="16dp"
            android:textSize="20sp"
            android:textStyle="bold"
            android:textColor="@color/textColorPrimary"
            app:layout_constraintEnd_toEndOf="parent"

app:layout_constraintStart_toEndOf="@id/imageViewCharacter"

app:layout_constraintTop_toTopOf="@id/imageViewCharacter"
            tools:text="Character Name" />

        <ImageView
            android:id="@+id/iconFavorites"
            android:layout_width="16dp"
            android:layout_height="16dp"
            android:layout_marginTop="12dp"
            android:src="@drawable/ic_star_gold"
            app:tint="@color/starColor"

app:layout_constraintStart_toStartOf="@id/textViewCharacter
Name"

app:layout_constraintTop_toBottomOf="@id/textViewCharacterN
ame"

app:layout_constraintBottom_toBottomOf="@id/textViewFavorit
es"

android:contentDescription="@string/favorites_icon_descript
ion" />


        <TextView
            android:id="@+id/textViewFavorites"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginStart="6dp"
            android:textSize="14sp"
            android:textStyle="bold"
            android:textColor="@color/textColorSecondary"

app:layout_constraintTop_toTopOf="@id/iconFavorites"

app:layout_constraintStart_toEndOf="@id/iconFavorites"
            tools:text="98,425" />


        <TextView
```

```
                android:id="@+id/textViewVoiceActors"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_marginTop="12dp"
                android:textSize="14sp"
                android:textColor="@color/textColorSecondary"

app:layout_constraintTop_toBottomOf="@id/textViewFavorites"

app:layout_constraintStart_toStartOf="@id/textViewCharacter
Name"
                app:layout_constraintEnd_toEndOf="parent"
                tools:text="JP: Voice Actor\nEN: Voice Actor"
/>

            <Button
                android:id="@+id/buttonDetail"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginTop="16dp"
                android:text="@string/details_button"
                app:layout_constraintEnd_toEndOf="parent"

app:layout_constraintTop_toBottomOf="@id/imageViewCharacter
"

app:layout_constraintBottom_toBottomOf="parent"/>

            <Button
                android:id="@+id/buttonUrl"

style="@style/Widget.MaterialComponents.Button.TextButton"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginEnd="8dp"
```

```
             android:text="@string/view_profile_button"

app:layout_constraintBaseline_toBaselineOf="@id/buttonDetai
l"

app:layout_constraintEnd_toStartOf="@id/buttonDetail" />

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

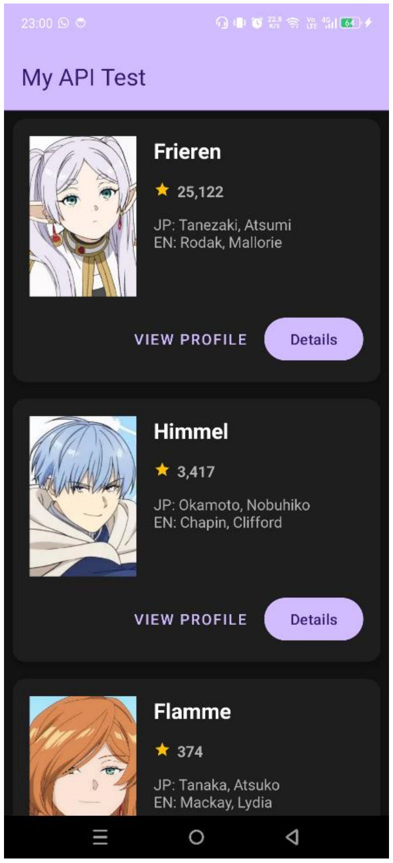
Tabel 1. 20 Source Code item_list.xml

21. nav_graph.xml

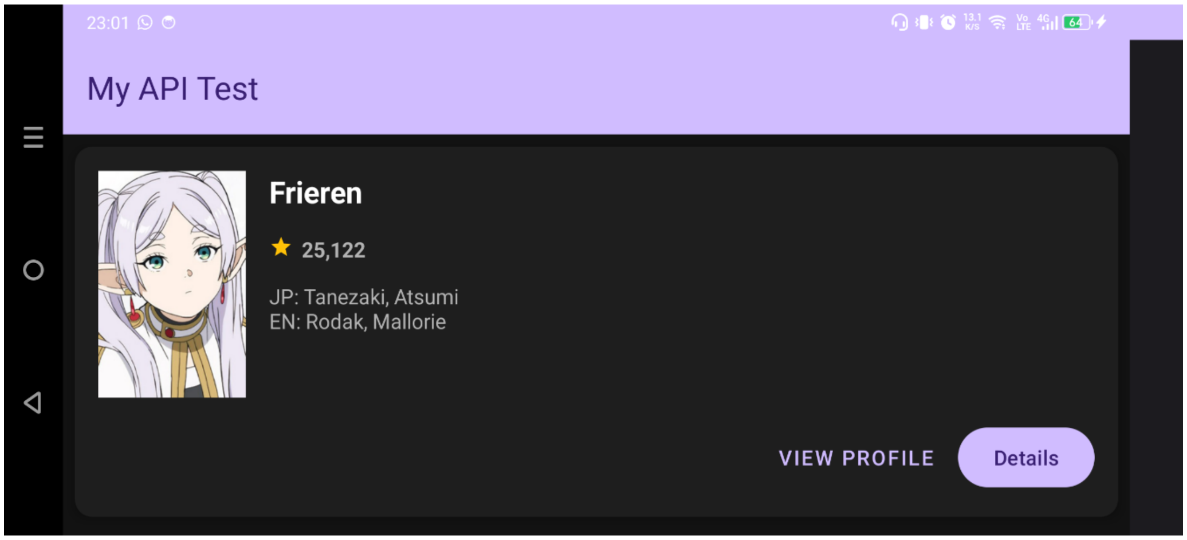
```
<?xml version="1.0" encoding="utf-8"?>
<navigation
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/nav_graph"
    app:startDestination="@id/homeFragment">

    <fragment
        android:id="@+id/homeFragment"
        android:name="com.example.myapi_test.HomeFragment"
        tools:layout="@layout/fragment_home">

        <action

android:id="@+id/action_homeFragment_to_detailFragment"
            app:destination="@id/detailFragment" />

    </fragment>

    <fragment
        android:id="@+id/detailFragment"
```

```
                
android:name="com.example.myapi_test.DetailFragment"
        tools:layout="@layout/fragment_detail">

        <argument
            android:name="character"

app:argType="com.example.myapi_test.CharacterInfo" />
    </fragment>

</navigation>
```
Tabel 1. 21 Source Code nav_graph.xml

B. Output Program



Gambar 1 Screenshot Hasil Jawaban Soal 1

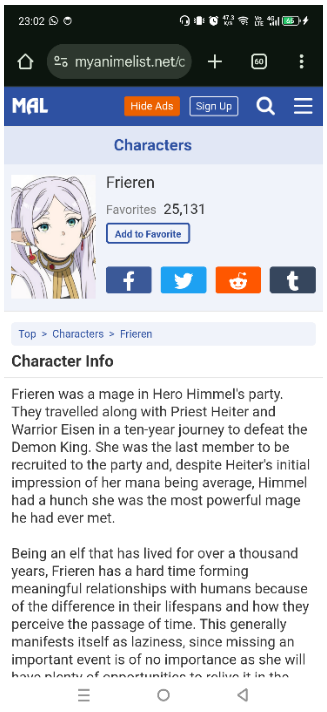


Frieren

★ 25,122

JP: Tanezaki, Atsumi
EN: Rodak, Mallorie

Gambar 2 Screenshot Hasil Jawaban Soal 1



Frieren

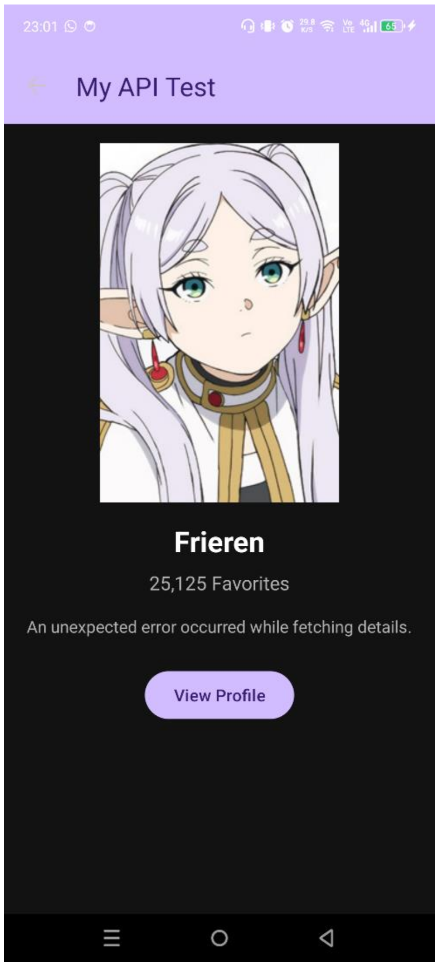
Favorites 25,131

Character Info

Frieren was a mage in Hero Himmel's party. They travelled along with Priest Heiter and Warrior Eisen in a ten-year journey to defeat the Demon King. She was the last member to be recruited to the party and, despite Heiter's initial impression of her mana being average, Himmel had a hunch she was the most powerful mage he had ever met.

Being an elf that has lived for over a thousand years, Frieren has a hard time forming meaningful relationships with humans because of the difference in their lifespans and how they perceive the passage of time. This generally manifests itself as laziness, since missing an important event is of no importance as she will

Gambar 3 Screenshot tombol Detail



An unexpected error occurred while fetching details.

Gambar 4 Screenshot tombol Info

C. Pembahasan

1. AppDatabase.kt

Digunakan sebagai penghubung utama data base room (local database) dengan cara menyediakan akses ke DAO

2. CacheMapper.kt

Digunakan untuk konversi data misal dari objek CharacterInfo akan di ubah menjadi CharacterInfoEntity dengan tujuan agar bisa di simpan di room.

3. CharacterAdapter.kt

Adapter, dan juga sebagai listener button

4. **CharacterDao.kt**

   DAO buat crud.

5. **CharacterInfoEntity.kt**

   Struktur tabel character untuk database Room

6. **CharacterMapper.kt**

   Konversi data agar bisa dipakai.

7. **CharacterModels.kt**

   Data class utama

8. **CharacterRepository.kt**

   Penguhubung antara remote database, dan local database

9. **CharacterViewModelFactory.kt**

   digunakan untuk membuat instance dari viewmodel yang lain serta untuk memastikan viewmodel memliki depedensi yang di perlukan

10. **DetailFragment.kt**

    Fragment detail yang mengambil data dari navigation, dan view model yang bersangkutan.

11. **DetailViewModel.kt**

    Viewmodel dari detail fragment.

12. **HomeFragment.kt**

HomeFragment, sebagai fragment pertama yang terbuka saat membuka aplikasi

13. HomeViewModel.kt

Viewmodel dari HomeFragment

14. JikanApiService.kt

Wadah untuk membuat request terhadap api

15. MainActivity.kt

Berisi navigasi dan instalasi splash screen

16. RetrofitInstance.kt

Untuk melakukan koneksi terhadap endpoint yang sudah di set.

17. activity_main.xml

Berisi toolbar, dan navhost

18. fragment_detail.xml

Berisi image view, dan text view untuk ui detail fragment

19. fragment_home.xml

Berisi template wadah recylerview yang akan di populate oleh item_list

20. item_list.xml

Card atau ui dari recylerview

21. nav_graph.xml

Navigasi dan data passing

D. Tautan Git

https://github.com/au290/College-Work/tree/main/Semester-4/Pemrogaman-Mobile/Modul-5