LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 5

MENGAMBIL DATA DARI INTERNET

Oleh:

Putri Ridha Amalia NIM. 2010817120007

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
MARET 2022

LEMBAR PENGESAHAN
LAPORAN PRAKTIKUM PEMROGRAMAN MOBILE
MODUL 5

Laporan Praktikum Pemrograman Mobile Modul 5: Mengambil Data dari Internet ini disusun sebagai syarat lulus mata kuliah Praktikum Pemrograman Mobile. Laporan Prakitkum ini dikerjakan oleh:

Nama Praktikan	: Putri Ridha Amalia
NIM	: 2010817120007

Menyetujui,	Mengetahui,
Asisten Praktikum	Dosen Penanggung Jawab Praktikum
Rezi Rahadianor	Andreyan Rizky Baskara, S.Kom., M.Kom.
NIM. 1810817210019	NIP. 19930703 201903 1 011

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

SOAL

Buatlah sebuah aplikasi Android sederhana dengan spesifikasi sebagai berikut:

Buatlah sebuah aplikasi Android sederhana untuk menampilkan data dari Internet melalui Public API

1. Daftar Public API yang dapat digunakan dapat dilihat pada link berikut: https://github.com/public-apis/public-apis (dapat juga mengambil diluar dari link tersebut)
2. Pada saat dijalankan, aplikasi akan terhubung dengan Internet untuk menarik data dari Public API tersebut
3. Gunakan library tambahan yaitu Retrofit untuk mempermudah proses koneksi internet
4. Gunakan library tambahan yaitu Mochi untuk mempermudah proses data JSON
5. Data tersebut kemudian ditampilkan dalam bentuk RecyclerView
6. Masing-masing data di RecyclerView tersebut dapat diklik untuk menampilkan detailnya
7. Gunakan LiveData dan ViewModel untuk mempertahankan state dari aplikasi pada saat Configuration Changes
8. Saat pengguna merotasi tampilan handphone dari Portrait menjadi Landscape maka tampilan data yang sudah ada tidak boleh hilang

A. Source Code

Table 1 Source Code MortyProperty.kt

```
   package com.example.modul5.network

data class MortyProperty (
    val results : List<MortyItem>? = null,
)
data class MortyItem (
    val name: String? = null,
    val image : String? = null,
    val status : String? = null,
    val spesies: String? = null,
    val gender: String? = null
    )
```

Table 2 Source Code MortyServiceApi.kt

```kotlin
package com.example.modul5.network

import com.squareup.moshi.Moshi
import
com.squareup.moshi.kotlin.reflect.KotlinJsonAdapterFactory
import retrofit2.Retrofit
import retrofit2.converter.moshi.MoshiConverterFactory
import retrofit2.http.GET

private const val BASE_URL =
"https://rickandmortyapi.com/api/"

private val moshi = Moshi.Builder()
    .add(KotlinJsonAdapterFactory())
    .build()

private val retrofit = Retrofit.Builder()

.addConverterFactory(MoshiConverterFactory.create(moshi))
    .baseUrl(BASE_URL)
    .build()

interface MortyServiceApi {
    @GET("character")
    suspend fun getMorty() : MortyProperty
}

object MortyApi{
    val retrofitServiceApi : MortyServiceApi by lazy {
        retrofit.create(MortyServiceApi::class.java)
    }
}
```

Table 3 Source Code MortyDetailFragment.kt

```kotlin
package com.example.modul5.ui


import android.os.Bundle

import android.view.LayoutInflater

import android.view.MenuItem
import android.view.View
import android.view.ViewGroup
import androidx.fragment.app.Fragment
import androidx.fragment.app.activityViewModels
import androidx.navigation.fragment.findNavController
import com.example.modul5.R
import
com.example.modul5.databinding.FragmentMortyDetailBinding


class MortyDetailFragment: Fragment() {
    private val viewModel: MortyViewModel by
activityViewModels()


    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        val binding =
FragmentMortyDetailBinding.inflate(inflater)


        binding.lifecycleOwner = this
        binding.viewModel = viewModel


        return binding.root
    }
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setHasOptionsMenu(true)
    }
    override fun onOptionsItemSelected(item: MenuItem): Boolean {
        when(item.itemId){
            android.R.id.home -> findNavController().navigate(R.id.action_mortyDetailFragment_to_mortyListFragment)
        }
        return true
    }
}
```

Table 4 Source Code MortyListAdapter.kt

```
 package com.example.modul5.ui

import android.view.LayoutInflater
import android.view.ViewGroup
import androidx.recyclerview.widget.ListAdapter
import androidx.recyclerview.widget.DiffUtil
import androidx.recyclerview.widget.RecyclerView
import com.example.modul5.databinding.ListViewItemBinding
import com.example.modul5.network.MortyItem

class MortyListAdapter ( val clickListener: MortyListener) :
    ListAdapter<MortyItem, MortyListAdapter.MortyViewHolder>(DiffCallback) {

    class MortyViewHolder(
        var binding: ListViewItemBinding
    ) : RecyclerView.ViewHolder(binding.root) {
        fun bind(clickListener: MortyListener, morty: MortyItem) {
            binding.morty = morty
            binding.clickListener = clickListener
            binding.executePendingBindings()
        }
    }

    companion object DiffCallback:
DiffUtil.ItemCallback<MortyItem>() {
        override fun areItemsTheSame(oldItem: MortyItem,
newItem: MortyItem): Boolean {
            return oldItem.name == newItem.name
        }

        override fun areContentsTheSame(oldItem:
MortyItem, newItem: MortyItem): Boolean {
            return oldItem.image == newItem.image
                    && oldItem.status == newItem.status
                    && oldItem.spesies == newItem.spesies
                    && oldItem.gender == newItem.gender
        }
    }
    override fun onCreateViewHolder(parent: ViewGroup,
viewType: Int): MortyViewHolder {
        val layoutInflater =
LayoutInflater.from(parent.context)
        return MortyViewHolder(
            ListViewItemBinding.inflate(layoutInflater,
parent, false)
        )
    }
    override fun onBindViewHolder(holder: MortyViewHolder,
position: Int) {
        val morty = getItem(position)
        holder.bind(clickListener, morty)
    }
}

class MortyListener(val clickListener: (morty: MortyItem)
-> Unit) {
    fun onClick(morty: MortyItem) = clickListener(morty)
}
```

Table 5 Source Code MortyListFragment.kt

```
 package com.example.modul5.ui

```

```
import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import androidx.appcompat.app.AppCompatActivity
import androidx.fragment.app.Fragment
import androidx.fragment.app.activityViewModels
import androidx.navigation.fragment.findNavController
import com.example.modul5.R
import com.example.modul5.databinding.FragmentMortyListBinding


class MortyListFragment: Fragment() {
    private val viewModel: MortyViewModel by activityViewModels()


    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        val binding = FragmentMortyListBinding.inflate(inflater)
        viewModel.getMortyList()
        binding.lifecycleOwner = this
        binding.viewModel = viewModel
        binding.recyclerView.adapter = MortyListAdapter(MortyListener { morty ->
            viewModel.onMortyClicked(morty)
```

```
                findNavController()

.navigate(R.id.action_mortyListFragment_to_mortyDetailFrag
ment)
        })
        (activity as
AppCompatActivity).supportActionBar?.title = "The Rick and
Morty"
        return binding.root
    }
}
```

Table 6 Source Code MortyViewModel.kt

```
 package com.example.modul5.ui

import androidx.lifecycle.LiveData
import androidx.lifecycle.MutableLiveData
import androidx.lifecycle.ViewModel
import androidx.lifecycle.viewModelScope
import com.example.modul5.network.MortyApi
import com.example.modul5.network.MortyItem
import kotlinx.coroutines.launch

enum class MortyApiStatus { LOADING, ERROR, DONE}

class MortyViewModel: ViewModel() {
    private val _status =
MutableLiveData<MortyApiStatus>()
    val status: LiveData<MortyApiStatus> = _status

    private val _mortys =
MutableLiveData<List<MortyItem>?> ()
    val mortys: MutableLiveData<List<MortyItem>?> =
_mortys

    private val _morty = MutableLiveData<MortyItem>()
    val morty: LiveData<MortyItem> = _morty

    fun getMortyList() {
```

```kotlin
        viewModelScope.launch {
            _status.value = MortyApiStatus.LOADING
            try {
                _mortys.value =
MortyApi.retrofitServiceApi.getMorty().results
                _status.value = MortyApiStatus.DONE
            } catch (e: Exception) {
                _mortys.value = listOf()
                _status.value = MortyApiStatus.ERROR
            }
        }
    }

    fun onMortyClicked(morty: MortyItem) {
        _morty.value = morty
    }
}
```

Table 7 Source Code BindingAdapters.kt

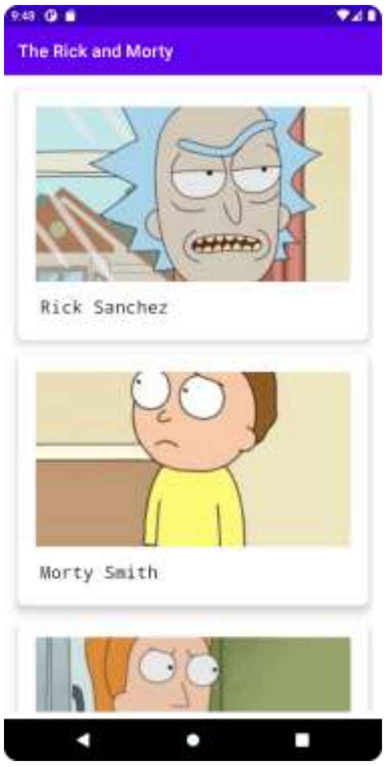
```kotlin
 package com.example.modul5

import android.view.View
import android.widget.ImageView
import androidx.core.net.toUri
import androidx.databinding.BindingAdapter
import androidx.recyclerview.widget.RecyclerView
import coil.load
import com.example.modul5.network.MortyItem
import com.example.modul5.network.MortyProperty
import com.example.modul5.ui.MortyListAdapter
import com.example.modul5.ui.MortyApiStatus


@BindingAdapter("listData")
fun bindRecyclerView(recyclerView: RecyclerView, data:
List<MortyItem>?){
    val adapter = recyclerView.adapter as MortyListAdapter
    adapter.submitList(data)
}

@BindingAdapter("imageUrl")
fun bindImage(imgView: ImageView, imgUrl: String?) {
    imgUrl?.let {
        val imgUri =
imgUrl.toUri().buildUpon().scheme("https").build()
```

```
        imgView.load(imgUri) {
            placeholder(R.drawable.loading_animation)
            error(R.drawable.ic_broken_image)
        }
    }
}

@BindingAdapter("apiStatus")
fun bindStatus(statusImageView: ImageView, status:
MortyApiStatus?) {
    when(status) {
        MortyApiStatus.LOADING -> {
            statusImageView.visibility = View.VISIBLE

statusImageView.setImageResource(R.drawable.loading_animat
ion)
        }
        MortyApiStatus.DONE -> {
            statusImageView.visibility = View.GONE
        }
        MortyApiStatus.ERROR -> {
            statusImageView.visibility = View.VISIBLE

statusImageView.setImageResource(R.drawable.ic_connection_
error)
        }
    }
}
```

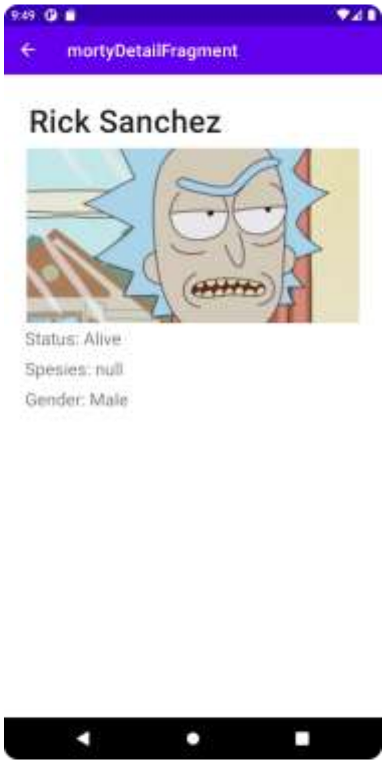
Table 8 Source Code MainActivity.kt

```
 package com.example.modul5

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import androidx.navigation.NavController
import androidx.navigation.fragment.NavHostFragment
import androidx.navigation.ui.NavigationUI
import java.security.AccessController

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

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        val navHostFragment =
supportFragmentManager.findFragmentById(R.id.nav_host_frag
ment) as NavHostFragment
        navController = navHostFragment.navController
        NavigationUI.setupActionBarWithNavController(this,
navController)
    }
}
```

B. Output Program



Gambar 1 Screenshot Tampilan Awal Aplikasi



Rick Sanchez

Status: Alive

Spesies: null

Gender: Male

Gambar 2 Screenshot Tampilan Detail Fragment

TAUTAN GIT

Berikut adalah tautan untuk source code yang telah dibuat.
https://github.com/putriridha13/praktikummobile2/tree/master/modul5