LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 5

Mengambil Data dari Internet

Oleh:

Bachrul Uluum **NIM. 2010817210025**

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
MEI 2022

LEMBAR PENGESAHAN
LAPORAN PRAKTIKUM PEMROGRAMAN I
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 : Bachrul Uluum
NIM : 2010817210025

Menyetujui, Asisten Praktikum	Mengetahui, Dosen Penanggung Jawab Praktikum
Rezi Rahdianor NIM. 1810817210019	Andreyan Rizky Baskara, S.Kom., M.Kom.. NIP. 199307032019031011

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

SOAL

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 Moshi 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

Path : app/src/main/java/com/uluumbch/poetrymodul5/network/
Nama File : Poetry.kt

```
package com.uluumbch.poetrymodul5.network
data class Poetry(
    val title : String,
    val author : String,
    val lines : List<String>,
    val linecount : Int
)
```

Nama file: PoetryServiceApi.kt

```
package com.uluumbch.poetrymodul5.network

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


private const val BASE_URL = "https://poetrydb.org"

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

private val retrofit = Retrofit.Builder()
    .addConverterFactory(MoshiConverterFactory.create(moshi))
    .baseUrl(BASE_URL)
    .build()

interface PoetryServiceApi{
    @GET("/author/Robinson")
    suspend fun getData() : List<Poetry>
}

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

Path : app/src/main/java/com/uluumbch/poetrymodul5/ui/
Nama file: PoetryDetailFragment.kt

```
package com.uluumbch.poetrymodul5.ui

import android.os.Bundle
import android.view.LayoutInflater
import android.view.MenuItem
import android.view.View
import android.view.ViewGroup
import android.widget.TextView
import androidx.appcompat.app.AppCompatActivity
import androidx.fragment.app.Fragment
import androidx.fragment.app.activityViewModels
import androidx.lifecycle.ViewModelProvider
```

```
import androidx.navigation.fragment.findNavController
import com.uluumbch.poetrymodul5.MainActivity
import com.uluumbch.poetrymodul5.R
import com.uluumbch.poetrymodul5.databinding.FragmentPoetryDetailBinding


class PoetryDetailFragment : Fragment() {
    private val viewModel: PoetryViewModel by activityViewModels()

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

        (activity      as      AppCompatActivity).supportActionBar?.title      =
viewModel.poem.value?.title
        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_poetryDetailFragment_to_poetryListFragme
nt)
        }
        return true
    }
}
```

Nama file: PoetryListFragment.kt

```
package com.uluumbch.poetrymodul5.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.lifecycle.ViewModelProvider
import androidx.navigation.fragment.findNavController
import androidx.recyclerview.widget.LinearLayoutManager
import com.google.android.material.divider.MaterialDividerItemDecoration
import com.google.android.material.snackbar.Snackbar
import com.uluumbch.poetrymodul5.R
```

```
import com.uluumbch.poetrymodul5.databinding.FragmentPoetryListBinding

class PoetryListFragment : Fragment(){
    private val viewModel: PoetryViewModel by activityViewModels()

    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        val binding = FragmentPoetryListBinding.inflate(inflater)
        viewModel.getPoetryList()
        binding.lifecycleOwner = this
        binding.viewModel = viewModel
        binding.recyclerView.adapter = PoetryListAdapter(PoetryListener { poetry -
>
            viewModel.onPoetryClicked(poetry)
            findNavController()
                .navigate(R.id.action_poetryListFragment_to_poetryDetailFragment)
        })

        (activity as AppCompatActivity).supportActionBar?.title = "List puisi"
        // menambahkan dekorasi garis bawah

binding.recyclerView.addItemDecoration(MaterialDividerItemDecoration(requireConte
xt(), LinearLayoutManager.VERTICAL))


        return binding.root
    }
}
```

Nama file: PoetryListAdapter.kt

```
package com.uluumbch.poetrymodul5.ui

import android.view.LayoutInflater
import android.view.ViewGroup
import androidx.recyclerview.widget.DiffUtil
import androidx.recyclerview.widget.RecyclerView
import androidx.recyclerview.widget.ListAdapter
import com.uluumbch.poetrymodul5.databinding.ListViewItemBinding
import com.uluumbch.poetrymodul5.network.Poetry

class PoetryListAdapter(private val clickListener: PoetryListener) :
    ListAdapter<Poetry, PoetryListAdapter.PoetryViewHolder>(DiffCallback)
{
    class PoetryViewHolder(
        var binding: ListViewItemBinding
    ) : RecyclerView.ViewHolder(binding.root){
        fun bind(clickListener: PoetryListener, poetry: Poetry){
            binding.poetry = poetry
            binding.clickListener = clickListener
            binding.executePendingBindings()
        }
    }

    companion object DiffCallback : DiffUtil.ItemCallback<Poetry>(){
```

```
        override fun areItemsTheSame(oldItem: Poetry, newItem: Poetry): Boolean {
            return oldItem.title == newItem.title
        }

        override fun areContentsTheSame(oldItem: Poetry, newItem: Poetry): Boolean
{
            return  oldItem.author  ==  newItem.author  &&  oldItem.linecount  ==
newItem.linecount
        }

    }

    override  fun  onCreateViewHolder(parent:  ViewGroup,  viewType:  Int)  :
PoetryViewHolder {
        val layoutInflater = LayoutInflater.from(parent.context)
        return PoetryViewHolder(
            ListViewItemBinding.inflate(layoutInflater, parent, false)
        )
    }

    override fun onBindViewHolder(holder: PoetryViewHolder, position: Int){
        val poetry = getItem(position)
        holder.bind(clickListener, poetry)
    }



}

class PoetryListener(val clickListener: (poetry: Poetry ) -> Unit){
    fun onClick(poetry: Poetry) = clickListener(poetry)
}
```

Nama file: PoetryeViewModel.kt

```
package com.uluumbch.poetrymodul5.ui

import android.util.Log
import androidx.lifecycle.LiveData
import androidx.lifecycle.MutableLiveData
import androidx.lifecycle.ViewModel
import androidx.lifecycle.viewModelScope
import com.uluumbch.poetrymodul5.network.Poetry
import com.uluumbch.poetrymodul5.network.PoetryApi
import kotlinx.coroutines.launch
import java.lang.Exception

enum class PoetryApiStatus { LOADING, ERROR, DONE }

class PoetryViewModel : ViewModel() {
    private val _status = MutableLiveData<PoetryApiStatus>()
    val status: LiveData<PoetryApiStatus> = _status

    private val _poetry = MutableLiveData<List<Poetry>>()
    val poetry: LiveData<List<Poetry>> = _poetry

    private val _poem = MutableLiveData<Poetry>()
    val poem: LiveData<Poetry> = _poem
```

```
    fun listToString(list: List<String>): String {
        return list.joinToString("\n")
    }

    fun getPoetryList() {
        viewModelScope.launch {
            _status.value = PoetryApiStatus.LOADING
            try {
                _poetry.value = PoetryApi.retrofitServiceApi.getData()
                _status.value = PoetryApiStatus.DONE
            } catch (e: Exception) {
                _poetry.value = listOf()
                _status.value = PoetryApiStatus.ERROR
            }
        }
    }

    fun onPoetryClicked(poetry: Poetry) {
        _poem.value = poetry
    }


}
```

Path : app/src/main/java/com/uluumbch/poetrymodul5/
Nama file: BindingAdapters.kt

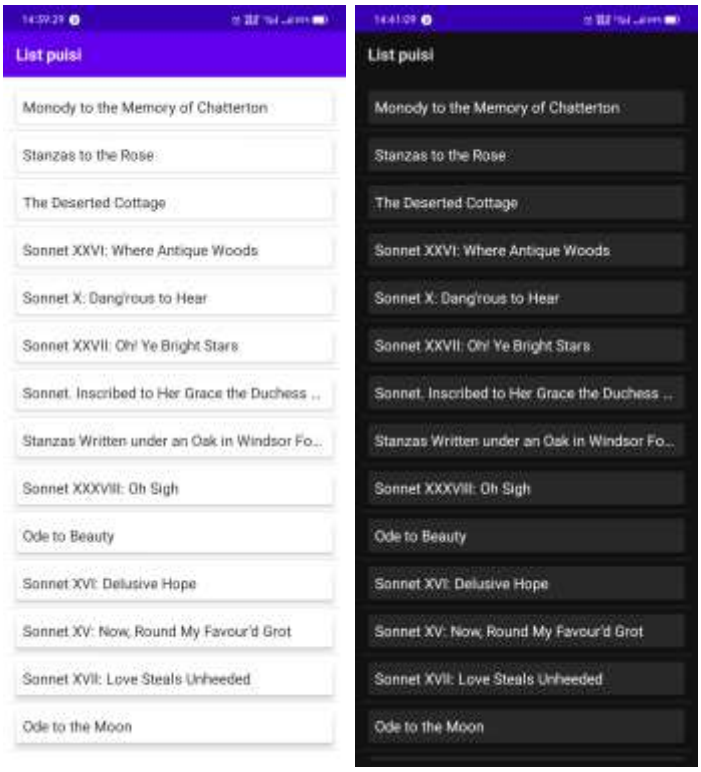
```
package com.uluumbch.poetrymodul5

import android.view.View
import android.widget.ImageView
import androidx.databinding.BindingAdapter
import androidx.recyclerview.widget.RecyclerView
import com.uluumbch.poetrymodul5.network.Poetry
import com.uluumbch.poetrymodul5.network.PoetryApi
import com.uluumbch.poetrymodul5.ui.PoetryApiStatus
import com.uluumbch.poetrymodul5.ui.PoetryListAdapter

@BindingAdapter("listData")
fun bindRecyclerView(recyclerView: RecyclerView, data: List<Poetry>?){
    val adapter = recyclerView.adapter as PoetryListAdapter
    adapter.submitList(data)
}
@BindingAdapter("apiStatus")
fun bindStatus(statusImageView: ImageView, status: PoetryApiStatus?) {
    when(status) {
        PoetryApiStatus.LOADING -> {
            statusImageView.visibility = View.VISIBLE
            statusImageView.setImageResource(R.drawable.loading_animation)
        }
        PoetryApiStatus.DONE -> {
            statusImageView.visibility = View.GONE
        }
        PoetryApiStatus.ERROR -> {
            statusImageView.visibility = View.VISIBLE
            statusImageView.setImageResource(R.drawable.ic_connection_error)
        }
```

```
    }
}
```

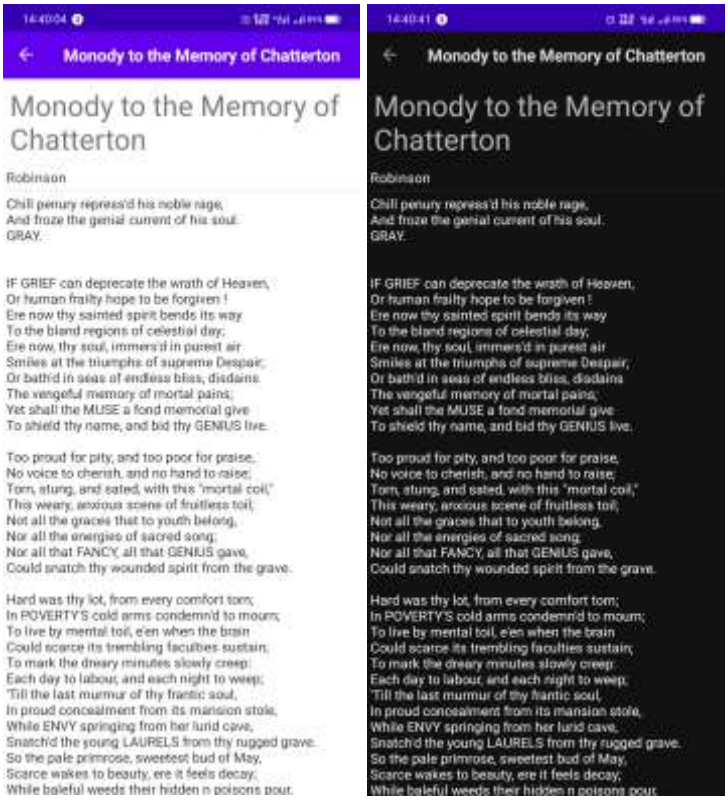
Nama file: MainActivity.kt

```
package com.uluumbch.poetrymodul5

import android.os.Bundle
import androidx.activity.viewModels
import androidx.appcompat.app.AppCompatActivity
import androidx.lifecycle.ViewModelProvider
import androidx.lifecycle.ViewModelProviders
import androidx.navigation.NavController
import androidx.navigation.fragment.NavHostFragment
import androidx.navigation.ui.NavigationUI
import com.uluumbch.poetrymodul5.ui.PoetryViewModel

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_fragment) as
NavHostFragment
        navController = navHostFragment.navController
        NavigationUI.setupActionBarWithNavController(this, navController)
    }
}
```

B. Output Program



Gambar 1 Tampilan awal aplikasi pada mode terang dan gelap



Gambar 2 Tampilan saat salah satu item di klik pada mode terang dan gelap

C. Pembahasan

Kode utama aplikasi terdapat pada file MainActivity.kt, dari file tersebut kita meng-inflate file xml yang akaan digunakan yaitu main_activity.xml. terdapat beberapa file selain MainActivity.kt yang digunakan pada aplikasi ini, diantaranya:

- **app/src/main/java/com/uluumbch/poetrymodul5/BindingAdapters.kt**

  File ini berisi fungsi-fungsi yang berguna untuk memberikan data terkait kepada RecyclerView dan juga memberikan gambar yang tepat untuk status aplikasi.

- **app/src/main/java/com/uluumbch/poetrymodul5/network/Poetry.kt**

  File ini berisi data model yang digunakan untuk meng-konversi data darti JSON ke objek milik kotlin, sehingga dapat dibaca dan diolah di bahasa pemrograman kotlin.

- **app/src/main/java/com/uluumbch/poetrymodul5/network/PoetryServiceApi.kt**

  File ini berisi fungsi dan objek yang berguna untuk menyiapkan pemanggilan ke Rest API pada server. File ini mendefinisikan url dan metode yang digunakan untuk request data. Di file ini juga converter Moshi digunakan.

- **app/src/main/java/com/uluumbch/poetrymodul5/ui/PoetryDetailFragment.kt**

  File ini berguna untuk menampilkan dan menginflate layout untuk detail data Poetry. Sehingga saat item dari API di klik, akan memulai fragment ini berisi data yang sesuai.

- **app/src/main/java/com/uluumbch/poetrymodul5/ui/PoetryListFragment.kt**

  File ini berguna untuk menampilkan list dari semua data yang didapat dari REST API. File ini akan menginflate layout yang sesuai dan menggunakannya untuk menampilkan data.

- **app/src/main/java/com/uluumbch/poetrymodul5/ui/PoetryViewmodel.kt**

  File ini berisi variabel yang digunakan untuk menampung data yang didapat setelah request ke API. File ini berisi LiveData yang diteruskan ke layout yang sesuai.

- **app/src/main/java/com/uluumbch/poetrymodul5/ui/PoetryListAdapter.kt**

  File ini berisi Adapter yang digunakan untuk menampilkan list dari data yang diambil. File ini berhubungan dengan file `PoetryListFragment.kt` . RecyclerView diterapkan pada file ini sehingga List dapat muncul ke layer aplikasi.

.

D. Tautan Git

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