LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 5

MENGAMBIL DATA DARI INTERNET

Oleh:

Salsabila Syifa **NIM. 2010817320004**

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
JUNI 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 Mobile. Laporan Praktikum ini dikerjakan oleh:

Nama Praktikan : Salsabila Syifa
NIM : 20101817320004

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

DAFTAR ISI

DAFTAR GAMBAR

SOAL 1

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

MainActivity.kt

```
1   package com.example.modul5
2
3   import androidx.appcompat.app.AppCompatActivity
4   import android.os.Bundle
5   import androidx.navigation.NavController
6   import androidx.navigation.fragment.NavHostFragment
7   import androidx.navigation.ui.NavigationUI
8
9   class MainActivity : AppCompatActivity() {
10      private lateinit var navController: NavController
11
12      override fun onCreate(savedInstanceState: Bundle?) {
13          super.onCreate(savedInstanceState)
14  //                                  val    binding    =
    ActivityMainBinding.inflate(layoutInflater)
15          setContentView(R.layout.activity_main)
16          val               navHostFragment                =
    supportFragmentManager.findFragmentById(R.id.nav_host_fr
    agment) as NavHostFragment
17          navController = navHostFragment.navController
18
19  NavigationUI.setupActionBarWithNavController(this,
    navController)
20      }
21  }
```

BindingAdapters.kt

```
1   package com.example.modul5
2
3   import android.view.View
4   import android.widget.ImageView
5   import androidx.databinding.BindingAdapter
6   import androidx.recyclerview.widget.RecyclerView
7   import com.example.modul5.network.Poetry
8   import com.example.modul5.ui.PoetryApiStatus
9   import com.example.modul5.ui.PoetryListAdapter
10
11  @BindingAdapter("listData")
12  fun  bindRecyclerView(recyclerView:  RecyclerView,  data:
    List<Poetry>?){
13      val     adapter     =     recyclerView.adapter     as
    PoetryListAdapter
14      adapter.submitList(data)
15  }
16
17  @BindingAdapter("apiStatus")
```

```
fun bindStatus(statusImageView: ImageView, status:
PoetryApiStatus?) {
    when(status) {
        PoetryApiStatus.LOADING -> {
            statusImageView.visibility = View.VISIBLE

statusImageView.setImageResource(R.drawable.loading_anim
ation)
        }
        PoetryApiStatus.DONE -> {
            statusImageView.visibility = View.GONE
        }
        PoetryApiStatus.ERROR -> {
            statusImageView.visibility = View.VISIBLE

statusImageView.setImageResource(R.drawable.ic_connectio
n_error)
        }
    }
}
```

Poetry.kt

```
package com.example.modul5.network

data class Poetry(
    val title : String,
    val author : String,
    val lines : List<String>,
    val linecount : Int
)
```

PoetryServiceApi.kt

```
package com.example.modul5.network

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


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

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

```

```
17  private val retrofit = Retrofit.Builder()
18      .addConverterFactory(MoshiConverterFactory.create(mo
    shi))
19      .baseUrl(BASE_URL)
20      .build()
21
22  interface PoetryServiceApi{
23      @GET("/author/William Shakespeare")
24      suspend fun getData() : List<Poetry>
25  }
26
27  object PoetryApi{
28      val retrofitServiceApi : PoetryServiceApi by lazy {
29          retrofit.create(PoetryServiceApi::class.java)
30      }
31  }
```

PoetryDetailFragment.kt

```
1   package com.example.modul5.ui
2
3   import android.os.Bundle
4   import android.view.LayoutInflater
5   import android.view.MenuItem
6   import android.view.View
7   import android.view.ViewGroup
8   import androidx.appcompat.app.AppCompatActivity
9   import androidx.fragment.app.Fragment
10  import androidx.fragment.app.activityViewModels
11  import androidx.navigation.fragment.findNavController
12  import com.example.modul5.R
13  import
14  com.example.modul5.databinding.FragmentPoetryDetailBindi
    ng
15
16  class PoetryDetailFragment : Fragment() {
17      private    val    viewModel:    PoetryViewModel    by
    activityViewModels()
18
19      override fun onCreateView(
20          inflater: LayoutInflater,
21          container: ViewGroup?,
22          savedInstanceState: Bundle?
23      ): View? {
24          val                    binding                    =
    FragmentPoetryDetailBinding.inflate(inflater)
25          binding.lifecycleOwner = this
26          binding.viewModel = viewModel
27
```

```
28          (activity as AppCompatActivity).supportActionBar?.title = viewModel.poem.value?.title
29          return binding.root
30      }
31
32      override fun onCreate(savedInstanceState: Bundle?) {
33          super.onCreate(savedInstanceState)
34          setHasOptionsMenu(true)
35
36      }
37
38      override fun onOptionsItemSelected(item: MenuItem): Boolean {
39          when(item.itemId){
40              android.R.id.home -> findNavController().navigate(R.id.action_poetryDetailFragment_to_poetryListFragment)
41          }
42          return true
43      }
44  }
```

PoetryListFragment.kt

```
1   package com.example.modul5.ui
2
3   import android.os.Bundle
4   import android.view.LayoutInflater
5   import android.view.View
6   import android.view.ViewGroup
7   import androidx.appcompat.app.AppCompatActivity
8   import androidx.fragment.app.Fragment
9   import androidx.fragment.app.activityViewModels
10  import androidx.navigation.fragment.findNavController
11  import androidx.recyclerview.widget.LinearLayoutManager
12  import com.example.modul5.R
13  import com.example.modul5.databinding.FragmentPoetryListBinding
14  import com.google.android.material.divider.MaterialDividerItemDecoration
15
16
17  class PoetryListFragment : Fragment(){
18      private val viewModel: PoetryViewModel by activityViewModels()
19
20      override fun onCreateView(
            inflater: LayoutInflater,
```

```
21          container: ViewGroup?,
22          savedInstanceState: Bundle?
23      ): View? {
            val                    binding                    =
24  FragmentPoetryListBinding.inflate(inflater)
25          viewModel.getPoetryList()
26          binding.lifecycleOwner = this
27          binding.viewModel = viewModel
28          binding.recyclerView.adapter                       =
    PoetryListAdapter(PoetryListener { poetry ->
29              viewModel.onPoetryClicked(poetry)
30              findNavController()
31                  .navigate(R.id.action_poetryListFragment
    _to_poetryDetailFragment)
32          })
33
34          (activity                                          as
    AppCompatActivity).supportActionBar?.title    =    "Karya
    Puisi William Shakespeare"
35
36
37  binding.recyclerView.addItemDecoration(MaterialDividerIt
    emDecoration(requireContext(),
38  LinearLayoutManager.VERTICAL))
39
40
41          return binding.root
42      }
43  }
```

PoetryViewModel.kt

```
1   package com.example.modul5.ui
2
3   import androidx.lifecycle.LiveData
4   import androidx.lifecycle.MutableLiveData
5   import androidx.lifecycle.ViewModel
6   import androidx.lifecycle.viewModelScope
7   import com.example.modul5.network.Poetry
8   import com.example.modul5.network.PoetryApi
9   import kotlinx.coroutines.launch
10  import java.lang.Exception
11
12  enum class PoetryApiStatus { LOADING, ERROR, DONE }
13
14  class PoetryViewModel : ViewModel() {
15      private           val           _status           =
    MutableLiveData<PoetryApiStatus>()
16      val status: LiveData<PoetryApiStatus> = _status
```

```
17
18      private            val            _poetry            =
    MutableLiveData<List<Poetry>>()
19      val poetry: LiveData<List<Poetry>> = _poetry
20
21      private val _poem = MutableLiveData<Poetry>()
22      val poem: LiveData<Poetry> = _poem
23
24
25      fun listToString(list: List<String>): String {
26          return list.joinToString("\n")
27      }
28
29      fun getPoetryList() {
30          viewModelScope.launch {
31              _status.value = PoetryApiStatus.LOADING
32              try {
33                  _poetry.value                              =
    PoetryApi.retrofitServiceApi.getData()
34                  _status.value = PoetryApiStatus.DONE
35              } catch (e: Exception) {
36                  _poetry.value = listOf()
37                  _status.value = PoetryApiStatus.ERROR
38              }
39          }
40      }
41
42      fun onPoetryClicked(poetry: Poetry) {
43          _poem.value = poetry
44      }
45
46
47  }
```

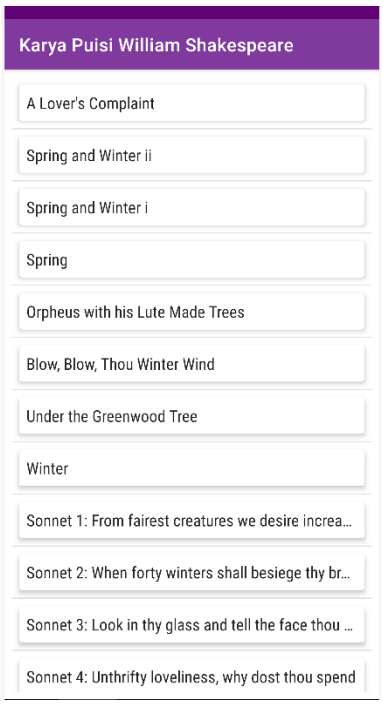
PoetryListAdapter.kt

```
1   package com.example.modul5.ui
2
3   import android.view.LayoutInflater
4   import android.view.ViewGroup
5   import androidx.recyclerview.widget.DiffUtil
6   import androidx.recyclerview.widget.ListAdapter
7   import androidx.recyclerview.widget.RecyclerView
8   import
9   com.example.modul5.databinding.ListViewItemBinding
10  import com.example.modul5.network.Poetry
11
12  class   PoetryListAdapter(private   val   clickListener:
    PoetryListener) :
```

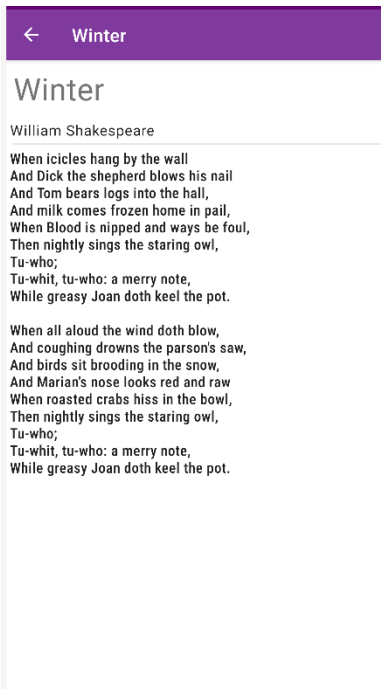
```
13      ListAdapter<Poetry,
14  PoetryListAdapter.PoetryViewHolder>(DiffCallback)
15  {
16      class PoetryViewHolder(
17          var binding: ListViewItemBinding
        ) : RecyclerView.ViewHolder(binding.root){
18          fun bind(clickListener: PoetryListener, poetry:
    Poetry){
19              binding.poetry = poetry
20              binding.clickListener = clickListener
21              binding.executePendingBindings()
22          }
23      }
24
25      companion object DiffCallback :
    DiffUtil.ItemCallback<Poetry>(){
26          override fun areItemsTheSame(oldItem: Poetry,
    newItem: Poetry): Boolean {
27              return oldItem.title == newItem.title
28          }
29
30          override fun areContentsTheSame(oldItem: Poetry,
    newItem: Poetry): Boolean {
31              return oldItem.author == newItem.author &&
    oldItem.linecount == newItem.linecount
32          }
33
34      }
35
36      override fun onCreateViewHolder(parent: ViewGroup,
    viewType: Int) : PoetryViewHolder {
37          val layoutInflater =
    LayoutInflater.from(parent.context)
38          return PoetryViewHolder(
39              ListViewItemBinding.inflate(layoutInflater,
40  parent, false)
41          )
42      }
43
44      override fun onBindViewHolder(holder:
    PoetryViewHolder, position: Int){
45          val poetry = getItem(position)
46          holder.bind(clickListener, poetry)
47      }
48
49
50
```

```
51  }
52
53  class    PoetryListener(val    clickListener:    (poetry:
    Poetry ) -> Unit){
54      fun onClick(poetry: Poetry) = clickListener(poetry)
55  }
```

B. Output Program



Gambar 1. Fragment Daftar



Gambar 2. Fragment Isi

C. Pembahasan

- MainActivity.kt adalah file kode utama aplikasi yang digunakan untuk *inflate* file xml yang digunakan yakni main_activity.xml.
- BindingAdapters.kt: File ini berisi fungsi yang berguna tidak hanya untuk meneruskan data yang relevan ke RecyclerView, tetapi juga untuk memberikan gambaran yang baik tentang status aplikasi.
- Poetry.kt: File yang berisi model data, yang digunakan untuk mengonversi data dari objek JSON ke kotlin, sehingga Anda dapat membaca dan memproses data dalam bahasa pemrograman kotlin.
- PoetryServiceApi.kt: File ini berisi fungsi dan objek yang membantu mengatur panggilan ke Rest API di server. File ini mendefinisikan URL dan metode yang digunakan untuk meminta data. File ini juga menggunakan konverter Moshi.

- PoetryDetailFragment.kt: File untuk menampilkan dan memperkuat grafik dari data detail Poetry. Oleh karena itu, ketika mengklik elemen di API akan memulai bagian itu dengan data yang sesuai.
- PoetryListFragment.kt: File ini berguna untuk menampilkan daftar semua data yang diambil dari REST API. File ini mengembangkan tata letak yang sesuai dan menggunakannya untuk menampilkan data.
- PoetryViewmodel.kt: File yang berisi variabel yang digunakan untuk menyimpan data yang diambil setelah diminta oleh API. File ini berisi LiveData yang akan diteruskan ke tata letak yang sesuai.
- PoetryListAdapter.kt: File yang berhubungan dengan file PoetryListFragment.kt ini berisi Adapter yang digunakan untuk menampilkan daftar data yang dipulihkan. RecyclerView diterapkan ke file ini sehingga daftar ditampilkan di lapisan aplikasi.

D. Tautan Git

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