LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 5

CONNECT TO THE INTERNET

Oleh:

Putra Whyra Pratama S. **NIM. 2310817210029**

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 : Putra Whyra Pratama S.
NIM : 2310817210029

Menyetujui, Asisten Praktikum	Mengetahui, Dosen Penanggung Jawab Praktikum
Zulfa Auliya Akbar NIM. 2210817210026	Muti`a Maulida S.Kom M.T.I 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. MainActivity.kt

Tabel 1. Source Code MainActivity.kt

```
1  package com.example.monsterhunterarmor
2
3  import android.os.Bundle
4  import androidx.appcompat.app.AppCompatActivity
5  import
   com.example.monsterhunterarmor.databinding.ActivityMainBindi
   ng
6
7  class MainActivity : AppCompatActivity() {
8      private lateinit var binding: ActivityMainBinding
9
10     override fun onCreate(savedInstanceState: Bundle?) {
11         super.onCreate(savedInstanceState)
12         binding =
   ActivityMainBinding.inflate(layoutInflater)
13         setContentView(binding.root)
14     }
15 }
```

2. ArmorAdapter.kt

Tabel 2. Source Code ArmorAdapter.kt

```
1  package com.example.monsterhunterarmor.adapter
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 com.bumptech.glide.Glide
9  import com.example.monsterhunterarmor.R
10 import com.example.monsterhunterarmor.data.local.ArmorEntity
11 import
   com.example.monsterhunterarmor.databinding.ItemArmorBinding
12
13 class ArmorAdapter(
14     private val listener: OnArmorClickListener
15 ) : ListAdapter<ArmorEntity,
   ArmorAdapter.ArmorViewHolder>(DIFF_CALLBACK) {
16
17     interface OnArmorClickListener {
18         fun onDetailClick(armor: ArmorEntity)
19         fun onSearchClick(armor: ArmorEntity)
20     }
21
```

```
22      override fun onCreateViewHolder(parent: ViewGroup,
    viewType: Int): ArmorViewHolder {
23          val binding =
    ItemArmorBinding.inflate(LayoutInflater.from(parent.context),
    parent, false)
24          return ArmorViewHolder(binding, listener)
25      }
26
27      override fun onBindViewHolder(holder: ArmorViewHolder,
    position: Int) {
28          val armor = getItem(position)
29          holder.bind(armor)
30      }
31
32      class ArmorViewHolder(
33          private val binding: ItemArmorBinding,
34          private val listener: OnArmorClickListener
35      ) : RecyclerView.ViewHolder(binding.root) {
36
37          fun bind(armor: ArmorEntity) {
38              binding.tvArmorName.text = armor.name
39              binding.tvArmorInfo.text = "Rank: ${armor.rank} |
    Type: ${armor.type}"
40
41              Glide.with(itemView.context)
42                  .load(armor.imageUrl)
43
    .placeholder(R.drawable.ic_launcher_background)
44                  .error(R.drawable.ic_launcher_foreground)
45                  .into(binding.ivArmorPhoto)
46
47              binding.btnDetail.setOnClickListener {
48                  listener.onDetailClick(armor)
49              }
50
51              binding.btnSearch.setOnClickListener {
52                  listener.onSearchClick(armor)
53              }
54          }
55      }
56
57      companion object {
58          val DIFF_CALLBACK = object :
    DiffUtil.ItemCallback<ArmorEntity>() {
59              override fun areItemsTheSame(oldItem:
    ArmorEntity, newItem: ArmorEntity): Boolean {
60                  return oldItem.id == newItem.id
61              }
62
63              override fun areContentsTheSame(oldItem:
    ArmorEntity, newItem: ArmorEntity): Boolean {
64                  return oldItem == newItem
```

```
65              }
66          }
67      }
68  }
```

3. ArmorDao.kt

Tabel 3. Source Code ArmorDao.kt

```
1   package com.example.monsterhunterarmor.data.local
2
3   import androidx.room.Dao
4   import androidx.room.Insert
5   import androidx.room.OnConflictStrategy
6   import androidx.room.Query
7   import kotlinx.coroutines.flow.Flow
8
9   @Dao
10  interface ArmorDao {
11      @Query("SELECT * FROM armor")
12      fun getAllArmor(): Flow<List<ArmorEntity>>
13
14      @Insert(onConflict = OnConflictStrategy.REPLACE)
15      suspend fun insertAll(armorList: List<ArmorEntity>)
16
17      @Query("DELETE FROM armor")
18      suspend fun deleteAll()
19  }
```

4. ArmorDatabase.kt

Tabel 4. Source Code ArmorDatabase.kt

```
1   package com.example.monsterhunterarmor.data.local
2
3   import android.content.Context
4   import androidx.room.Database
5   import androidx.room.Room
6   import androidx.room.RoomDatabase
7   import androidx.room.TypeConverters
8
9   @Database(entities = [ArmorEntity::class], version = 2,
    exportSchema = false)
10  @TypeConverters(Converters::class)
11  abstract class ArmorDatabase : RoomDatabase() {
12      abstract fun armorDao(): ArmorDao
13
14      companion object {
15          @Volatile
16          private var INSTANCE: ArmorDatabase? = null
17
```

```
18          fun getDatabase(context: Context): ArmorDatabase {
19              return INSTANCE ?: synchronized(this) {
20                  val instance = Room.databaseBuilder(
21                      context.applicationContext,
22                      ArmorDatabase::class.java,
23                      "armor_database"
24                  )
25                      .fallbackToDestructiveMigration()
26                      .build()
27                  INSTANCE = instance
28                  instance
29              }
30          }
31      }
32  }
```

5. ArmorEntity.kt

Tabel 5. Source Code ArmorEntity.kt

```
1   package com.example.monsterhunterarmor.data.local
2
3   import android.os.Parcelable
4   import androidx.room.Embedded
5   import androidx.room.Entity
6   import androidx.room.PrimaryKey
7   import
    com.example.monsterhunterarmor.data.model.ArmorDefense
8   import
    com.example.monsterhunterarmor.data.model.ArmorResistances
9   import com.example.monsterhunterarmor.data.model.ArmorSkill
10  import kotlinx.parcelize.Parcelize
11
12  @Parcelize
13  @Entity(tableName = "armor")
14  data class ArmorEntity(
15      @PrimaryKey
16      val id: Int,
17      val name: String,
18      val rank: String,
19      val type: String,
20      val imageUrl: String?,
21
22      @Embedded
23      val defense: ArmorDefense,
24
25      @Embedded
26      val resistances: ArmorResistances,
27
28      val skills: List<ArmorSkill>
29
```

```
30  ) : Parcelable
```

6. Converters.kt

Tabel 6. Source Code Converters.kt

```
1   package com.example.monsterhunterarmor.data.local
2
3   import androidx.room.TypeConverter
4   import com.example.monsterhunterarmor.data.model.ArmorSkill
5   import kotlinx.serialization.encodeToString
6   import kotlinx.serialization.json.Json
7
8   class Converters {
9       @TypeConverter
10      fun fromSkillList(skills: List<ArmorSkill>): String {
11          return Json.encodeToString(skills)
12      }
13
14      @TypeConverter
15      fun toSkillList(skillsJson: String): List<ArmorSkill> {
16          return Json.decodeFromString(skillsJson)
17      }
18  }
```

7. ArmorModels.kt

Tabel 7. Source Code ArmorModels.kt

```
1   package com.example.monsterhunterarmor.data.model
2
3   import android.os.Parcelable
4   import kotlinx.parcelize.Parcelize
5   import kotlinx.serialization.Serializable
6
7   @Serializable
8   data class ArmorResponse(
9       val id: Int,
10      val name: String,
11      val rank: String,
12      val type: String,
13      val assets: ArmorAssets? = null,
14      val defense: ArmorDefense,
15      val resistances: ArmorResistances,
16      val skills: List<ArmorSkill>,
17      val slots: List<ArmorSlot>
18  )
19
20  @Serializable
21  data class ArmorAssets(
22      val imageMale: String? = null,
```

```
23      val imageFemale: String? = null
24  )
25
26  @Parcelize
27  @Serializable
28  data class ArmorDefense(
29      val base: Int,
30      val max: Int,
31      val augmented: Int
32  ) : Parcelable
33
34  @Parcelize
35  @Serializable
36  data class ArmorResistances(
37      val fire: Int,
38      val water: Int,
39      val ice: Int,
40      val thunder: Int,
41      val dragon: Int
42  ) : Parcelable
43
44  @Parcelize
45  @Serializable
46  data class ArmorSkill(
47      val id: Int,
48      val level: Int,
49      val skillName: String,
50      val description: String
51  ) : Parcelable
52
53  @Serializable
54  data class ArmorSlot(
55      val rank: Int
56  )
```

8. ApiResponse.kt

Tabel 8. Source Code ApiResponse.kt

```
1   package com.example.monsterhunterarmor.data.remote
2
3   sealed class ApiResponse<out R> {
4       data class Success<out T>(val data: T) :
    ApiResponse<T>()
5       data class Error(val errorMessage: String) :
    ApiResponse<Nothing>()
6       object Loading : ApiResponse<Nothing>()
7   }
```

9. ArmorApiService.kt

Tabel 9. Source Code ArmorApiService.kt

```
1  package com.example.monsterhunterarmor.data.remote
2
3  import
   com.example.monsterhunterarmor.data.model.ArmorResponse
4  import retrofit2.http.GET
5
6  interface ArmorApiService {
7      @GET("armor")
8      suspend fun getArmor(): List<ArmorResponse>
9  }
```

10. AppContainer.kt

Tabel 10. Source Code AppContainer.kt

```
1  package com.example.monsterhunterarmor.di
2
3  import android.content.Context
4  import
   com.example.monsterhunterarmor.data.local.ArmorDatabase
5  import
   com.example.monsterhunterarmor.data.remote.ArmorApiService
6  import
   com.example.monsterhunterarmor.repository.ArmorRepository
7  import
   com.jakewharton.retrofit2.converter.kotlinx.serialization.as
   ConverterFactory
8  import kotlinx.serialization.json.Json
9  import okhttp3.MediaType.Companion.toMediaType
10 import okhttp3.OkHttpClient
11 import okhttp3.logging.HttpLoggingInterceptor
12 import retrofit2.Retrofit
13
14 class AppContainer(private val context: Context) {
15
16     private val json = Json { ignoreUnknownKeys = true }
17
18     private val loggingInterceptor =
19
   HttpLoggingInterceptor().setLevel(HttpLoggingInterceptor.Lev
   el.BODY)
20
21     private val client = OkHttpClient.Builder()
22         .addInterceptor(loggingInterceptor)
23         .build()
24
```

```
25      private val retrofit = Retrofit.Builder()
26          .baseUrl("https://mhw-db.com/")
27          .addConverterFactory(json.asConverterFactory("application/json".toMediaType()))
28          .client(client)
29          .build()
30
31      private val apiService: ArmorApiService by lazy {
32          retrofit.create(ArmorApiService::class.java)
33      }
34
35      private val armorDb: ArmorDatabase by lazy {
36          ArmorDatabase.getDatabase(context)
37      }
38
39      val armorRepository: ArmorRepository by lazy {
40          ArmorRepository(apiService, armorDb.armorDao())
41      }
42  }
```

11. ArmorDetailFragment.kt

Tabel 11. Source Code ArmorDetailFragment.kt

```
1   package com.example.monsterhunterarmor.presentation.detail
2
3   import android.os.Bundle
4   import android.view.LayoutInflater
5   import android.view.View
6   import android.view.ViewGroup
7   import android.widget.TextView
8   import androidx.fragment.app.Fragment
9   import androidx.navigation.fragment.navArgs
10  import com.bumptech.glide.Glide
11  import com.example.monsterhunterarmor.data.model.ArmorSkill
12  import com.example.monsterhunterarmor.databinding.FragmentArmorDetailBinding
13
14  class ArmorDetailFragment : Fragment() {
15
16      private var _binding: FragmentArmorDetailBinding? = null
17      private val binding get() = _binding!!
18
19      private val args: ArmorDetailFragmentArgs by navArgs()
20
21      override fun onCreateView(
22          inflater: LayoutInflater, container: ViewGroup?,
23          savedInstanceState: Bundle?
24      ): View {
```

```
25          _binding                                                =
    FragmentArmorDetailBinding.inflate(inflater,       container,
    false)
26          return binding.root
27      }
28
29      override       fun       onViewCreated(view:       View,
    savedInstanceState: Bundle?) {
30          super.onViewCreated(view, savedInstanceState)
31          val armor = args.armor
32
33          Glide.with(this)
34              .load(armor.imageUrl)
35              .into(binding.ivArmorDetailPhoto)
36
37          binding.tvArmorDetailName.text = armor.name
38
39          val defense = armor.defense
40          binding.tvDefenseStats.text = "Base: ${defense.base}
    | Max: ${defense.max} | Augmented: ${defense.augmented}"
41
42          val res = armor.resistances
43          binding.tvResistancesStats.text = "Fire: ${res.fire},
    Water: ${res.water}, Thunder: ${res.thunder}, Ice: ${res.ice},
    Dragon: ${res.dragon}"
44
45          addSkillsToLayout(armor.skills)
46      }
47
48      private fun addSkillsToLayout(skills: List<ArmorSkill>) {
49          if (skills.isEmpty()) {
50              val noSkillsText = TextView(context).apply {
51                  text = "No skills available."
52
    setTextAppearance(com.google.android.material.R.style.TextAp
    pearance_MaterialComponents_Body2)
53              }
54              binding.llSkillsContainer.addView(noSkillsText)
55              return
56          }
57
58          skills.forEach { skill ->
59              val skillTitle = TextView(context).apply {
60                  text      =      "${skill.skillName}      (Lv.
    ${skill.level})"
61
    setTextAppearance(com.google.android.material.R.style.TextAp
    pearance_MaterialComponents_Subtitle1)
62                  layoutParams = ViewGroup.MarginLayoutParams(
63                      ViewGroup.LayoutParams.MATCH_PARENT,
64                      ViewGroup.LayoutParams.WRAP_CONTENT
65                  ).apply {
```

```
66                      topMargin               =               if
    (binding.llSkillsContainer.childCount > 1) 24 else 8
67                  }
68              }
69
70              val skillDescription = TextView(context).apply {
71                  text = skill.description
72
    setTextAppearance(com.google.android.material.R.style.TextAp
    pearance_MaterialComponents_Body2)
73              }
74
75              binding.llSkillsContainer.addView(skillTitle)
76
    binding.llSkillsContainer.addView(skillDescription)
77          }
78      }
79
80      override fun onDestroyView() {
81          super.onDestroyView()
82          _binding = null
83      }
84  }
```

12. ArmorListFragment.kt

Tabel 12. Source Code ArmorListFragment.kt

```
1   package com.example.monsterhunterarmor.presentation.home
2
3   import android.content.Intent
4   import android.net.Uri
5   import android.os.Bundle
6   import android.util.Log
7   import android.view.LayoutInflater
8   import android.view.View
9   import android.view.ViewGroup
10  import android.widget.Toast
11  import androidx.core.view.isVisible
12  import androidx.fragment.app.Fragment
13  import androidx.fragment.app.viewModels
14  import androidx.lifecycle.Lifecycle
15  import androidx.lifecycle.lifecycleScope
16  import androidx.lifecycle.repeatOnLifecycle
17  import androidx.navigation.fragment.findNavController
18  import androidx.recyclerview.widget.LinearLayoutManager
19  import com.example.monsterhunterarmor.ArmorApplication
20  import com.example.monsterhunterarmor.adapter.ArmorAdapter
21  import com.example.monsterhunterarmor.data.local.ArmorEntity
22  import
    com.example.monsterhunterarmor.data.remote.ApiResponse
```

```
23  import
    com.example.monsterhunterarmor.databinding.FragmentArmorLis
    tBinding
24  import com.example.monsterhunterarmor.utils.ViewModelFactory
25  import kotlinx.coroutines.launch
26
27  class ArmorListFragment : Fragment() {
28
29      private var _binding: FragmentArmorListBinding? = null
30      private val binding get() = _binding!!
31
32      private val viewModel: ArmorViewModel by viewModels {
33          ViewModelFactory((requireActivity().application   as
    ArmorApplication).appContainer.armorRepository)
34      }
35
36      private   val   armorAdapter   =   ArmorAdapter(object   :
    ArmorAdapter.OnArmorClickListener {
37          override fun onDetailClick(armor: ArmorEntity) {
38              viewModel.onDetailButtonClicked(armor)
39          }
40
41          override fun onSearchClick(armor: ArmorEntity) {
42              viewModel.onSearchButtonClicked(armor)
43          }
44      })
45
46      override fun onCreateView(
47          inflater: LayoutInflater, container: ViewGroup?,
48          savedInstanceState: Bundle?
49      ): View {
50          _binding                                              =
    FragmentArmorListBinding.inflate(inflater, container, false)
51          return binding.root
52      }
53
54      override       fun       onViewCreated(view:       View,
    savedInstanceState: Bundle?) {
55          super.onViewCreated(view, savedInstanceState)
56
57          setupRecyclerView()
58          observeArmorData()
59          observeViewEvents()
60      }
61
62      private fun setupRecyclerView() {
63          binding.rvArmor.apply {
64              adapter = armorAdapter
65              layoutManager = LinearLayoutManager(context)
66          }
67      }
68
```

```
69      private fun observeArmorData() {
70          viewLifecycleOwner.lifecycleScope.launch {
71              repeatOnLifecycle(Lifecycle.State.STARTED) {
72                  viewModel.armorState.collect { response ->
73                      when (response) {
74                          is ApiResponse.Loading -> {
75                              binding.progressBar.isVisible  =
    true
76                          }
77                          is ApiResponse.Success -> {
78                              response.data.collect          {
    armorList ->
79
    binding.progressBar.isVisible = false
80
    armorAdapter.submitList(armorList)
81                                  if  (armorList.isNotEmpty())
    {
82
    Log.d("ArmorListFragment",      "${armorList.size}     items
    submitted.")
83                                  }
84                              }
85                          }
86                          is ApiResponse.Error -> {
87                              binding.progressBar.isVisible  =
    false
88                              Toast.makeText(context,
    response.errorMessage, Toast.LENGTH_LONG).show()
89                          }
90                      }
91                  }
92              }
93          }
94      }
95
96      private fun observeViewEvents() {
97          viewLifecycleOwner.lifecycleScope.launch {
98              repeatOnLifecycle(Lifecycle.State.STARTED) {
99                  viewModel.eventFlow.collect { event ->
100                     when (event) {
101                         is
    ArmorViewModel.ViewEvent.NavigateToDetail -> {
102                             Log.d("ArmorListFragment",
    "Navigating   to   detail   for:   ${event.armor.name}   (ID:
    ${event.armor.id})")
103                             val           action           =
    ArmorListFragmentDirections.actionArmorListFragmentToArmorD
    etailFragment(event.armor)
104
    findNavController().navigate(action)
105                         }
```

```
106                         is
    ArmorViewModel.ViewEvent.OpenBrowser -> {
107                             val           query           =
    "https://monsterhunterworld.wiki.fextralife.com/${event.que
    ry.replace(" ", "+")}"
108                             val           intent          =
    Intent(Intent.ACTION_VIEW, Uri.parse(query))
109                             startActivity(intent)
110                         }
111                     }
112                 }
113             }
114         }
115     }
116
117     override fun onDestroyView() {
118         super.onDestroyView()
119         binding.rvArmor.adapter = null
120         _binding = null
121     }
122 }
```

13. ArmorListFragment.kt

Tabel 13. Source Code ArmorViewModel.kt

```
1   package com.example.monsterhunterarmor.presentation.home
2
3   import android.util.Log
4   import androidx.lifecycle.ViewModel
5   import androidx.lifecycle.viewModelScope
6   import com.example.monsterhunterarmor.data.local.ArmorEntity
7   import com.example.monsterhunterarmor.data.remote.ApiResponse
8   import
    com.example.monsterhunterarmor.repository.ArmorRepository
9   import kotlinx.coroutines.flow.Flow
10  import kotlinx.coroutines.flow.MutableSharedFlow
11  import kotlinx.coroutines.flow.MutableStateFlow
12  import kotlinx.coroutines.flow.asSharedFlow
13  import kotlinx.coroutines.flow.StateFlow
14  import kotlinx.coroutines.flow.collectLatest
15  import kotlinx.coroutines.launch
16
17  class ArmorViewModel(private val repository: ArmorRepository)
    : ViewModel() {
18
19      private           val           _armorState           =
    MutableStateFlow<ApiResponse<Flow<List<ArmorEntity>>>>(ApiRe
    sponse.Loading)
20      val                                          armorState:
    StateFlow<ApiResponse<Flow<List<ArmorEntity>>>> = _armorState
```

```
21
22      private val _eventFlow = MutableSharedFlow<ViewEvent>()
23      val eventFlow = _eventFlow.asSharedFlow()
24
25      init {
26          fetchArmor()
27      }
28
29      fun fetchArmor() {
30          viewModelScope.launch {
31              repository.getArmorList().collectLatest {
32                  _armorState.value = it
33              }
34          }
35      }
36
37      fun onDetailButtonClicked(armor: ArmorEntity) {
38          Log.d("ArmorViewModel", "Detail button clicked for:
    ${armor.name}")
39          viewModelScope.launch {
40
    _eventFlow.emit(ViewEvent.NavigateToDetail(armor))
41          }
42      }
43
44      fun onSearchButtonClicked(armor: ArmorEntity) {
45          Log.d("ArmorViewModel", "Search button clicked for:
    ${armor.name}")
46          viewModelScope.launch {
47
    _eventFlow.emit(ViewEvent.OpenBrowser(armor.name))
48          }
49      }
50
51      sealed class ViewEvent {
52          data class NavigateToDetail(val armor: ArmorEntity) :
    ViewEvent()
53          data   class   OpenBrowser(val   query:   String)   :
    ViewEvent()
54      }
55  }
```

14. ArmorRepository.kt

Tabel 14. Source Code ArmorRepository.kt

```
1  package com.example.monsterhunterarmor.repository
2
3  import com.example.monsterhunterarmor.data.local.ArmorDao
4  import com.example.monsterhunterarmor.data.local.ArmorEntity
5  import com.example.monsterhunterarmor.data.remote.ApiResponse
6  import
   com.example.monsterhunterarmor.data.remote.ArmorApiService
7  import kotlinx.coroutines.flow.Flow
8  import kotlinx.coroutines.flow.flow
9  import java.lang.Exception
10
11 class ArmorRepository(
12     private val apiService: ArmorApiService,
13     private val armorDao: ArmorDao
14 ) {
15     fun                                          getArmorList():
   Flow<ApiResponse<Flow<List<ArmorEntity>>>> = flow {
16         emit(ApiResponse.Loading)
17         val localData = armorDao.getAllArmor()
18         emit(ApiResponse.Success(localData))
19
20         try {
21             val response = apiService.getArmor()
22             val armorEntities = response.map { armorResponse -
   >
23                 ArmorEntity(
24                     id = armorResponse.id,
25                     name = armorResponse.name,
26                     rank = armorResponse.rank,
27                     type = armorResponse.type,
28                     imageUrl = armorResponse.assets?.imageMale
   ?: armorResponse.assets?.imageFemale,
29                     defense = armorResponse.defense,
30                     resistances = armorResponse.resistances,
31                     skills = armorResponse.skills
32                 )
33             }
34             armorDao.deleteAll()
35             armorDao.insertAll(armorEntities)
36         } catch (e: Exception) {
37             emit(ApiResponse.Error("Failed   to   fetch   from
   network: ${e.message}"))
38             e.printStackTrace()
39         }
40     }
41 }
```

15. ViewModelFactory.kt

Tabel 15. Source Code ViewModelFactory.kt

```
1  package com.example.monsterhunterarmor.utils
2
3  import androidx.lifecycle.ViewModel
4  import androidx.lifecycle.ViewModelProvider
5  import
   com.example.monsterhunterarmor.presentation.home.ArmorViewMo
   del
6  import
   com.example.monsterhunterarmor.repository.ArmorRepository
7
8  class      ViewModelFactory(private      val      repository:
   ArmorRepository) : ViewModelProvider.NewInstanceFactory() {
9      @Suppress("UNCHECKED_CAST")
10     override fun <T : ViewModel> create(modelClass: Class<T>):
   T {
11         if
   (modelClass.isAssignableFrom(ArmorViewModel::class.java)) {
12             return ArmorViewModel(repository) as T
13         }
14         throw   IllegalArgumentException("Unknown   ViewModel
   class: " + modelClass.name)
15     }
16 }
```

16. ArmorApplication.kt

Tabel 16. Source Code ArmorApplication.kt

```
1  package com.example.monsterhunterarmor
2
3  import android.app.Application
4  import com.example.monsterhunterarmor.di.AppContainer
5
6  class ArmorApplication : Application() {
7      lateinit var appContainer: AppContainer
8      override fun onCreate() {
9          super.onCreate()
10         appContainer = AppContainer(this)
11     }
12 }
```

17. activity_main.xml

Tabel 17. Source Code activity_main.xml

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <androidx.fragment.app.FragmentContainerView
3   xmlns:android="http://schemas.android.com/apk/res/android"
4       xmlns:app="http://schemas.android.com/apk/res-auto"
5       xmlns:tools="http://schemas.android.com/tools"
        android:id="@+id/nav_host_fragment"

6   android:name="androidx.navigation.fragment.NavHostFragment"
7       android:layout_width="match_parent"
8       android:layout_height="match_parent"
9       app:defaultNavHost="true"
10      app:navGraph="@navigation/main_nav_graph"
11      tools:context=".MainActivity" />
```

18. fragment_armor_detail.xml

Tabel 18. Source Code fragment_armor_detail.xml

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
3       xmlns:app="http://schemas.android.com/apk/res-auto"
4       xmlns:tools="http://schemas.android.com/tools"
5       android:layout_width="match_parent"
6       android:layout_height="match_parent"
7
    tools:context=".presentation.detail.ArmorDetailFragment">
8
9       <androidx.constraintlayout.widget.ConstraintLayout
10          android:layout_width="match_parent"
11          android:layout_height="wrap_content"
12          android:paddingBottom="16dp">
13
14          <ImageView
15              android:id="@+id/iv_armor_detail_photo"
16              android:layout_width="0dp"
17              android:layout_height="300dp"
18              android:scaleType="centerCrop"
19
    android:contentDescription="@string/armor_image"
20              app:layout_constraintEnd_toEndOf="parent"
21              app:layout_constraintStart_toStartOf="parent"
22              app:layout_constraintTop_toTopOf="parent"
23              tools:src="@tools:sample/backgrounds/scenic" />
24
25          <TextView
26              android:id="@+id/tv_armor_detail_name"
27              android:layout_width="0dp"
```

```
28              android:layout_height="wrap_content"
29              android:layout_marginHorizontal="16dp"
30              android:layout_marginTop="16dp"
31
    android:textAppearance="?attr/textAppearanceHeadlineSmall"
32              android:textStyle="bold"
33              app:layout_constraintEnd_toEndOf="parent"
34              app:layout_constraintStart_toStartOf="parent"
35
    app:layout_constraintTop_toBottomOf="@id/iv_armor_detail_ph
    oto"
36              tools:text="Direwolf Armor" />
37
38          <com.google.android.material.card.MaterialCardView
39              android:id="@+id/card_defense"
40              android:layout_width="0dp"
41              android:layout_height="wrap_content"
42              android:layout_marginTop="16dp"
43              app:cardCornerRadius="12dp"
44
    app:layout_constraintEnd_toEndOf="@+id/tv_armor_detail_name
    "
45
    app:layout_constraintStart_toStartOf="@+id/tv_armor_detail_
    name"
46
    app:layout_constraintTop_toBottomOf="@+id/tv_armor_detail_n
    ame">
47
48              <LinearLayout
49                  android:layout_width="match_parent"
50                  android:layout_height="wrap_content"
51                  android:orientation="vertical"
52                  android:padding="16dp">
53
54                  <TextView
55                      android:layout_width="wrap_content"
56                      android:layout_height="wrap_content"
57                      android:text="Defense"
58
    android:textAppearance="?attr/textAppearanceTitleMedium" />
59
60                  <TextView
61                      android:id="@+id/tv_defense_stats"
62                      android:layout_width="wrap_content"
63                      android:layout_height="wrap_content"
64                      android:layout_marginTop="8dp"
65
    android:textAppearance="?attr/textAppearanceBodyMedium"
66                      tools:text="Base:  70  |  Max:  110  |
    Augmented: 140" />
67              </LinearLayout>
```

```
68
    </com.google.android.material.card.MaterialCardView>
69
70          <com.google.android.material.card.MaterialCardView
71              android:id="@+id/card_resistances"
72              android:layout_width="0dp"
73              android:layout_height="wrap_content"
74              android:layout_marginTop="16dp"
75              app:cardCornerRadius="12dp"
76
    app:layout_constraintEnd_toEndOf="@+id/card_defense"
77
    app:layout_constraintStart_toStartOf="@+id/card_defense"
78
    app:layout_constraintTop_toBottomOf="@+id/card_defense">
79
80              <LinearLayout
81                  android:layout_width="match_parent"
82                  android:layout_height="wrap_content"
83                  android:orientation="vertical"
84                  android:padding="16dp">
85
86                  <TextView
87                      android:layout_width="wrap_content"
88                      android:layout_height="wrap_content"
89                      android:text="Resistances"
90
    android:textAppearance="?attr/textAppearanceTitleMedium" />
91
92                  <TextView
93                      android:id="@+id/tv_resistances_stats"
94                      android:layout_width="wrap_content"
95                      android:layout_height="wrap_content"
96                      android:layout_marginTop="8dp"
97
    android:textAppearance="?attr/textAppearanceBodyMedium"
98                      tools:text="Fire: 2, Water: -1, Thunder:
    0, Ice: 0, Dragon: 3" />
99              </LinearLayout>
100
    </com.google.android.material.card.MaterialCardView>
101
102         <com.google.android.material.card.MaterialCardView
103             android:id="@+id/card_skills"
104             android:layout_width="0dp"
105             android:layout_height="wrap_content"
106             android:layout_marginTop="16dp"
107             app:cardCornerRadius="12dp"
108
    app:layout_constraintEnd_toEndOf="@+id/card_resistances"
```

```
109
    app:layout_constraintStart_toStartOf="@+id/card_resistances
    "
110
    app:layout_constraintTop_toBottomOf="@+id/card_resistances"
    >
111
112             <LinearLayout
113                 android:id="@+id/ll_skills_container"
114                 android:layout_width="match_parent"
115                 android:layout_height="wrap_content"
116                 android:orientation="vertical"
117                 android:padding="16dp">
118
119                 <TextView
120                     android:layout_width="wrap_content"
121                     android:layout_height="wrap_content"
122                     android:layout_marginBottom="8dp"
123                     android:text="Skills"
124
    android:textAppearance="?attr/textAppearanceTitleMedium" />
125             </LinearLayout>
126
    </com.google.android.material.card.MaterialCardView>
127
128     </androidx.constraintlayout.widget.ConstraintLayout>
129 </ScrollView>
```

19. fragment_armor_list.xml

Tabel 19. Source Code fragment_armor_list.xml

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
3       xmlns:app="http://schemas.android.com/apk/res-auto"
4       xmlns:tools="http://schemas.android.com/tools"
5       android:layout_width="match_parent"
6       android:layout_height="match_parent"
7       tools:context=".presentation.home.ArmorListFragment">
8
9       <androidx.recyclerview.widget.RecyclerView
10          android:id="@+id/rv_armor"
11          android:layout_width="0dp"
12          android:layout_height="0dp"
13          app:layout_constraintBottom_toBottomOf="parent"
14          app:layout_constraintEnd_toEndOf="parent"
15          app:layout_constraintStart_toStartOf="parent"
16          app:layout_constraintTop_toTopOf="parent"
17          tools:listitem="@layout/item_armor" />
18
```

```
    <ProgressBar
        android:id="@+id/progress_bar"
        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" />

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

20. item_armor.xml

Tabel 20. Source Code item_armor.xml

```
<?xml version="1.0" encoding="utf-8"?>
<com.google.android.material.card.MaterialCardView
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_marginHorizontal="16dp"
    android:layout_marginVertical="8dp"
    app:cardCornerRadius="16dp"
    app:cardElevation="4dp">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:padding="12dp">

        <com.google.android.material.card.MaterialCardView
            android:id="@+id/card_image"
            android:layout_width="110dp"
            android:layout_height="110dp"
            app:cardCornerRadius="12dp"
            app:cardElevation="0dp"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent">

            <ImageView
                android:id="@+id/iv_armor_photo"
                android:layout_width="match_parent"
                android:layout_height="match_parent"

android:contentDescription="@string/armor_image"
                android:scaleType="centerCrop"
                tools:src="@tools:sample/avatars" />
```

```
33          </com.google.android.material.card.MaterialCardView>
34
35          <TextView
36              android:id="@+id/tv_armor_name"
37              android:layout_width="0dp"
38              android:layout_height="wrap_content"
39              android:layout_marginStart="16dp"
40              android:ellipsize="end"
41              android:maxLines="2"
42
    android:textAppearance="?attr/textAppearanceTitleMedium"
43              android:textStyle="bold"
44              app:layout_constraintEnd_toEndOf="parent"
45
    app:layout_constraintStart_toEndOf="@+id/card_image"
46              app:layout_constraintTop_toTopOf="parent"
47              tools:text="Direwolf Mail" />
48
49          <TextView
50              android:id="@+id/tv_armor_info"
51              android:layout_width="0dp"
52              android:layout_height="wrap_content"
53              android:layout_marginTop="4dp"
54
    android:textAppearance="?attr/textAppearanceBodySmall"
55
    app:layout_constraintEnd_toEndOf="@+id/tv_armor_name"
56
    app:layout_constraintStart_toStartOf="@+id/tv_armor_name"
57
    app:layout_constraintTop_toBottomOf="@+id/tv_armor_name"
58              tools:text="Rank: G | Type: Chest" />
59
60          <androidx.constraintlayout.widget.Barrier
61              android:id="@+id/content_barrier"
62              android:layout_width="wrap_content"
63              android:layout_height="wrap_content"
64              app:barrierDirection="bottom"
65
    app:constraint_referenced_ids="card_image,tv_armor_info" />
66
67
68          <LinearLayout
69              android:layout_width="0dp"
70              android:layout_height="wrap_content"
71              android:layout_marginTop="16dp"
72              android:orientation="horizontal"
73
    app:layout_constraintEnd_toEndOf="@+id/tv_armor_name"
74
    app:layout_constraintStart_toStartOf="@+id/tv_armor_name"
```

```
75
    app:layout_constraintTop_toBottomOf="@id/content_barrier">
76
77
    <com.google.android.material.button.MaterialButton
78                  android:id="@+id/btn_search"
79                  style="?attr/materialButtonOutlinedStyle"
80                  android:layout_width="0dp"
81                  android:layout_height="wrap_content"
82                  android:layout_marginEnd="4dp"
83                  android:layout_weight="1"
84                  android:text="@string/search_on_web" />
85
86
    <com.google.android.material.button.MaterialButton
87                  android:id="@+id/btn_detail"
88                  android:layout_width="0dp"
89                  android:layout_height="wrap_content"
90                  android:layout_marginStart="4dp"
91                  android:layout_weight="1"
92                  android:text="@string/detail" />
93          </LinearLayout>
94
95      </androidx.constraintlayout.widget.ConstraintLayout>
96  </com.google.android.material.card.MaterialCardView>
```

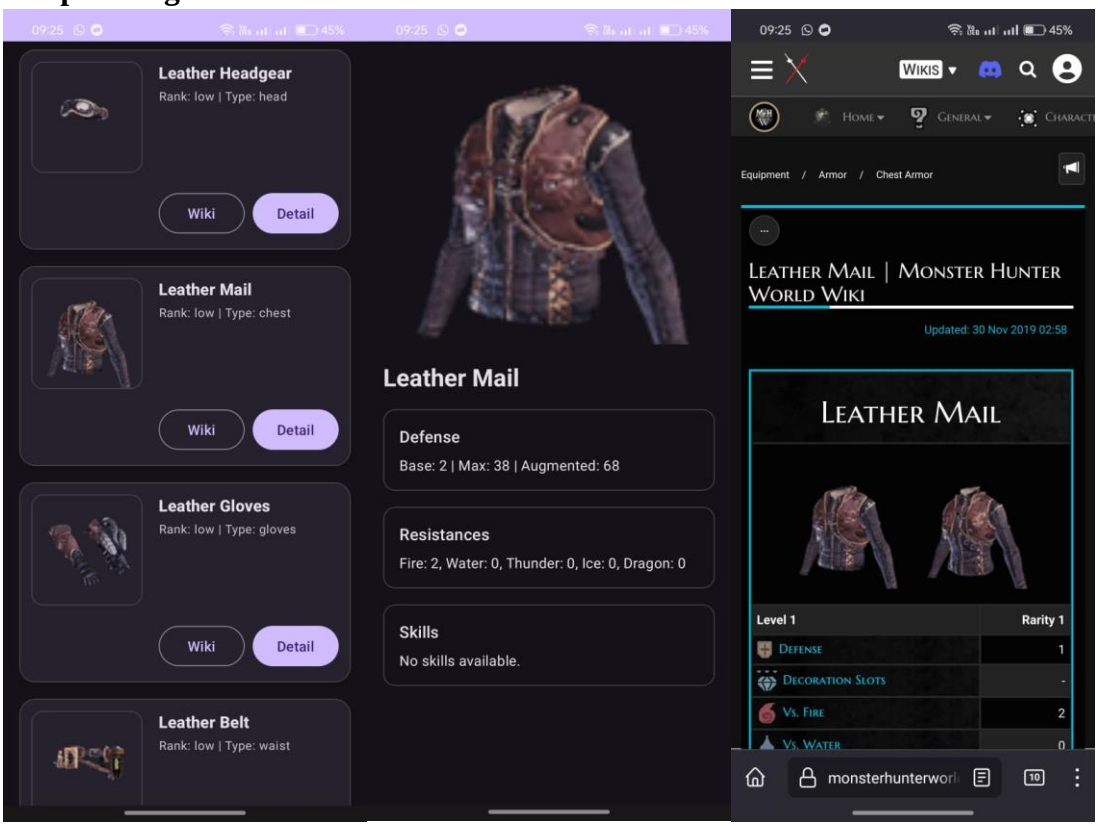
21. main_nav_graph.xml

Tabel 21. Source Code main_nav_graph.xml

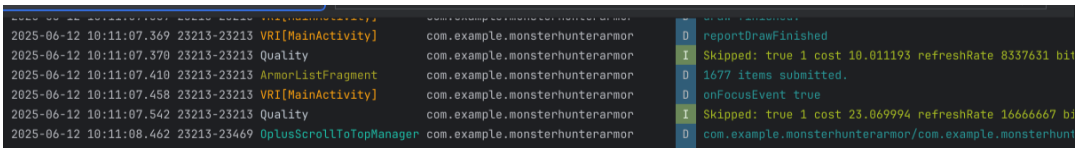
```
1   <?xml version="1.0" encoding="utf-8"?>
2   <navigation
    xmlns:android="http://schemas.android.com/apk/res/android"
3       xmlns:app="http://schemas.android.com/apk/res-auto"
4       xmlns:tools="http://schemas.android.com/tools"
5       android:id="@+id/main_nav_graph"
6       app:startDestination="@id/armorListFragment">
7
8       <fragment
9           android:id="@+id/armorListFragment"
10
    android:name="com.example.monsterhunterarmor.presentation.ho
    me.ArmorListFragment"
11          android:label="Armor List"
12          tools:layout="@layout/fragment_armor_list" >
13          <action
14
    android:id="@+id/action_armorListFragment_to_armorDetailFrag
    ment"
15              app:destination="@id/armorDetailFragment" />
16      </fragment>
```

```
17
18      <fragment
19          android:id="@+id/armorDetailFragment"
20
    android:name="com.example.monsterhunterarmor.presentation.de
    tail.ArmorDetailFragment"
21          android:label="Armor Detail"
22          tools:layout="@layout/fragment_armor_detail" >
23          <argument
24              android:name="armor"
25
    app:argType="com.example.monsterhunterarmor.data.local.Armor
    Entity" />
26      </fragment>
27
28  </navigation>
```

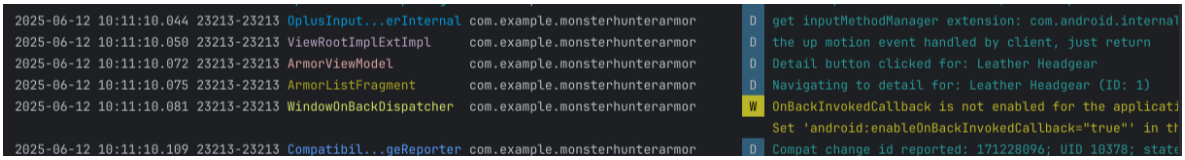
B. Output Program



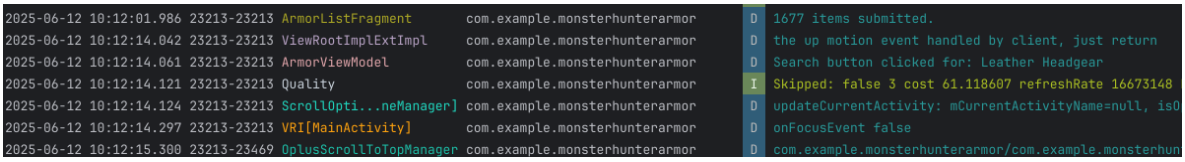
Gambar 1. Screenshot Hasil Jawaban Soal 1



Gambar 2. Screenshot Log Saat Data Item Masuk Ke Dalam List



Gambar 3. Screenshot Log Saat Tombol Detail Dan Data Dari List Yang Dipilih Ketika Berpindah Ke Halaman Detail



Gambar 4. Screenshot Log Tombol Explicit Intent Ditekan

C. Pembahasan

1. MainActivity.kt:

Merupakan entry point atau titik masuk utama aplikasi. Kelas MainActivity berfungsi sebagai host untuk NavHostFragment, yang bertanggung jawab memuat dan menampilkan berbagai fragment sesuai dengan alur navigasi aplikasi yang didefinisikan dalam main_nav_graph.xml.

2. ArmorApplication.kt:

Kelas Application kustom yang diinisialisasi saat aplikasi pertama kali dijalankan. Peran utamanya adalah untuk membuat dan mengelola dependency injection container (AppContainer) sebagai singleton, memastikan dependensi seperti repository dan database tersedia untuk seluruh siklus hidup aplikasi.

3. adapter/ArmorAdapter.kt:

Sebuah RecyclerView.Adapter yang bertanggung jawab untuk mengikat data armor (List<ArmorEntity>) ke tampilan item dalam daftar. ArmorAdapter menangani pembuatan ViewHolder, pengisian data ke setiap item, dan pengelolaan interaksi pengguna seperti klik, yang kemudian meneruskan event tersebut untuk diproses lebih lanjut (misalnya, navigasi ke halaman detail).

4. **data/local/ArmorEntity.kt:**

Sebuah data class yang berfungsi sebagai model tabel untuk database Room. Kelas ini menggunakan anotasi @Entity untuk mendefinisikan tabel armor. Setiap properti dalam kelas ini merepresentasikan kolom di dalam tabel tersebut.

5. **data/local/ArmorDao.kt:**

DAO (Data Access Object). Interface ini berisi deklarasi fungsi-fungsi untuk melakukan operasi CRUD (Create, Read, Update, Delete) pada tabel armor. Implementasi konkret dari DAO ini disediakan secara otomatis oleh Room.

6. **data/local/ArmorDatabase.kt:**

Kelas abstrak yang mewarisi RoomDatabase dan berfungsi sebagai konfigurasi utama untuk database aplikasi. Kelas ini mendefinisikan daftar entities (tabel) dan menyediakan akses ke DAO.

7. **data/local/Converters.kt:**

Menyediakan fungsi konversi tipe data (Type Converters) untuk Room. Fungsinya adalah mengubah tipe data kompleks yang tidak didukung secara native oleh Room (seperti List<ArmorSkill>) menjadi tipe data primitif (misalnya String JSON) agar dapat disimpan di database, dan sebaliknya.

8. **data/model/ArmorModels.kt:**

Berisi kumpulan data class yang merepresentasikan struktur data dari respons JSON API. Kelas-kelas ini digunakan oleh library kotlinx.serialization untuk melakukan parsing (deserialisasi) dari JSON menjadi objek Kotlin.

9. **data/remote/ArmorApiService.kt:**

Interface yang digunakan oleh Retrofit untuk mendefinisikan endpoints dari API. Setiap fungsi di dalam interface ini merepresentasikan satu panggilan API, seperti getArmorList() untuk mengambil daftar armor dari server.

10. data/remote/ApiResponse.kt:

Sebuah sealed class yang digunakan untuk membungkus respons dari panggilan jaringan. Ini memungkinkan penanganan state UI secara eksplisit untuk kondisi Loading, Success (dengan data), dan Error (dengan pesan kesalahan).

11. di/AppContainer.kt:

Berfungsi sebagai dependency injection container manual. Kelas ini bertanggung jawab untuk membuat dan menyediakan instance dari dependensi penting di seluruh aplikasi, seperti ArmorRepository dan ArmorApiService, untuk mempromosikan loose coupling dan kemudahan pengujian.

12. presentation/detail/ArmorDetailFragment.kt:

Fragment yang bertanggung jawab untuk menampilkan layar detail dari satu item armor. Fragment ini menerima data armor yang dipilih melalui Navigation Safe Args dan menampilkannya pada komponen UI yang sesuai.

13. presentation/home/ArmorListFragment.kt:

Fragment yang menjadi layar utama aplikasi. Bertugas menampilkan daftar armor dalam sebuah RecyclerView. Fragment ini mengobservasi data dari ArmorViewModel, menampilkan loading state dan pesan error, serta menangani input pengguna seperti pencarian dan navigasi.

14. presentation/home/ArmorViewModel.kt:

Komponen ViewModel yang menyediakan data dan state untuk ArmorListFragment. ArmorViewModel berinteraksi dengan ArmorRepository untuk mengambil data, mengelola state UI menggunakan StateFlow, dan menangani event dari UI (seperti klik) menggunakan Channel untuk memastikan logika terpisah dari tampilan.

15. repository/ArmorRepository.kt:

Bertindak sebagai Single Source of Truth (SSOT). Repository ini mengabstraksi sumber data (API atau database lokal) dari ViewModel. Ia berisi logika untuk mengambil data dari jaringan, menyimpannya ke dalam cache (database), dan menyediakan data yang konsisten untuk aplikasi.

16. utils/ViewModelFactory.kt:

Sebuah factory class yang bertujuan untuk membuat instance dari ArmorViewModel. Diperlukan karena ArmorViewModel memiliki dependensi (ArmorRepository) yang harus diinjeksikan saat pembuatan, sehingga tidak bisa menggunakan konstruktor default.

17. layout/activity_main.xml:

File layout untuk MainActivity. Berisi sebuah FragmentContainerView yang berfungsi sebagai wadah utama untuk NavHostFragment. Semua layar (fragment) dalam aplikasi akan dimuat di dalam container ini.

18. layout/fragment_armor_list.xml:

File layout XML yang mendefinisikan antarmuka pengguna untuk layar daftar armor. Layout ini berisi RecyclerView untuk menampilkan daftar data, ProgressBar sebagai indikator pemuatan, SearchView untuk fungsionalitas pencarian, dan TextView untuk menampilkan pesan status.

19. layout/fragment_armor_detail.xml:

Layout untuk halaman detail armor. Terdiri dari berbagai komponen seperti ImageView untuk gambar armor dan beberapa TextView untuk menampilkan atribut-atribut rinci seperti nama, rank, tipe, statistik pertahanan, dan resistensi.

20. layout/item_armor.xml:

Mendefinisikan layout untuk satu item dalam RecyclerView di ArmorListFragment. Layout ini berfungsi sebagai templat untuk setiap baris, yang menampilkan informasi ringkas seperti gambar, nama, dan tipe armor.

21. navigation/main_nav_graph.xml:

Pusat kendali navigasi aplikasi yang menggunakan Navigation Component. File ini mendefinisikan semua destinasi (fragment) dan actions (perpindahan antar fragment). Di sini, ArmorListFragment ditetapkan sebagai start destination dan didefinisikan pula aksi navigasi ke ArmorDetailFragment beserta argumen yang dikirimkan.

D. Tautan Git

Berikut adalah tautan untuk source code yang telah dibuat.

https://github.com/PutraWhyra789/praktikum_pemrograman_mobile/tree/a1e0d783a4a65170e613f76622c3469bbcff01b5/Module%205