LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 5

CONNECT TO THE INTERNET

Oleh:

Allano Lintang Ertantora **NIM. 2310817210004**

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
JUNI 2025

LEMBAR PENGESAHAN
LAPORAN PRAKTIKUM PEMROGRAMAN MOBILE
MODUL 5

Laporan Praktikum Pemrograman Mobile Modul 5: Connect To The Internet ini disusun sebagai syarat lulus mata kuliah Praktikum Pemrograman Mobile. Laporan Praktikum ini dikerjakan oleh:

Nama Praktikan : Zulfa Auliya Akbar
NIM : 2210817210026

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:

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.kt

Tabel 1. Source Code AppDatabase.kt

```
1  package com.allano.alquran.data.local
2
3  import android.content.Context
4  import androidx.room.Database
5  import androidx.room.Room
6  import androidx.room.RoomDatabase
7
8  @Database(entities = [SurahEntity::class,
   AyahEntity::class], version = 4, exportSchema = false)
9  abstract class AppDatabase : RoomDatabase() {
10     abstract fun surahDao(): SurahDao
11     abstract fun ayahDao(): AyahDao
```

```
12
13      companion object {
14          @Volatile
15          private var INSTANCE: AppDatabase? = null
16
17          fun getDatabase(context: Context): AppDatabase {
18              return INSTANCE ?: synchronized(this) {
19                  val instance = Room.databaseBuilder(
20                      context.applicationContext,
21                      AppDatabase::class.java,
22                      "quran_database"
23                  ).fallbackToDestructiveMigration().build()
24                  INSTANCE = instance
25                  instance
26              }
27          }
28      }
29  }
```

2. AyahDao.kt

Tabel 2. Source Code AyahDao.kt

```
1   package com.allano.alquran.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 AyahDao {
11
12      @Query("SELECT * FROM ayahs WHERE surahNumber =
    :surahNumber ORDER BY numberInSurah ASC")
13      fun getAyahsBySurahNumber(surahNumber: Int):
    Flow<List<AyahEntity>>
14
15      @Insert(onConflict = OnConflictStrategy.REPLACE)
16      suspend fun insertAll(ayahs: List<AyahEntity>)
17
18  }
```

3. AyahEntity.kt

Tabel 3. Source Code AyahEntity.kt

```
1   package com.allano.alquran.data.local
2
```

```
3  import androidx.room.Entity
4  import androidx.room.ForeignKey
5  import androidx.room.Index
6  import androidx.room.PrimaryKey
7
8  @Entity(
9      tableName = "ayahs",
10     foreignKeys = [
11         ForeignKey(
12             entity = SurahEntity::class,
13             parentColumns = ["number"],
14             childColumns = ["surahNumber"],
15             onDelete = ForeignKey.CASCADE
16         )
17     ],
18     indices = [Index(value = ["surahNumber"])]
19 )
20 data class AyahEntity(
21     @PrimaryKey
22     val number: Int,
23     val surahNumber: Int,
24     val numberInSurah: Int,
25     val arabicText: String,
26     val englishText: String
27 )
```

4. SurahDao.kt

Tabel 4. Source Code SurahDao.kt

```
1  package com.allano.alquran.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 SurahDao {
11     @Query("SELECT * FROM surahs")
12     fun getAllSurahs(): Flow<List<SurahEntity>>
13
14     @Insert(onConflict = OnConflictStrategy.REPLACE)
15     suspend fun insertAll(surahs: List<SurahEntity>)
16
17     @Query("DELETE FROM surahs")
18
```

```
19      suspend fun deleteAll()
    }
```

5. SurahEntity.kt

Tabel 5. Source Code SurahEntity.kt

```
1   package com.allano.alquran.data.local
2
3   import androidx.room.Entity
4   import androidx.room.PrimaryKey
5
6   @Entity(tableName = "surahs")
7   data class SurahEntity(
8       @PrimaryKey val number: Int,
9       val name: String,
10      val englishName: String,
11      val englishNameTranslation: String,
12      val revelationType: String,
13      val numberOfAyahs: Int
14  )
```

6. Ayah.kt

Tabel 6. Source Code Ayah.kt

```
1   @file:OptIn(InternalSerializationApi::class)
2
3   package com.allano.alquran.data.model
4
5   import kotlinx.serialization.InternalSerializationApi
6   import kotlinx.serialization.Serializable
7
8   @Serializable
9   data class Ayah(
10      val number: Int,
11      val numberInSurah: Int,
12      val text: String,
13      val juz: Int
14  )
15
16  @Serializable
17  data class SurahDetail(
18      val number: Int,
19      val name: String,
20      val englishName: String,
21      val englishNameTranslation: String,
22      val revelationType: String,
23      val ayahs: List<Ayah>
24  )
```

```
25
26 @Serializable
27 data class MultiEditionSurahDetailResponse(
28     val code: Int,
29     val status: String,
30     val data: List<SurahDetail>
31 )
32
33 data class MergedAyah(
34     val numberInSurah: Int,
35     val arabicText: String,
36     val englishText: String
37 )
```

7. Surah.kt

Tabel 7. Source Code Surah.kt

```
1  @file:OptIn(InternalSerializationApi::class)
2
3  package com.allano.alquran.data.model
4
5  import kotlinx.serialization.InternalSerializationApi
6  import kotlinx.serialization.SerialName
7  import kotlinx.serialization.Serializable
8  import kotlin.OptIn
9
10
11 @Serializable
12 data class Surah(
13     val number: Int,
14     val name: String,
15     @SerialName("englishName")
16     val englishName: String,
17     @SerialName("englishNameTranslation")
18     val englishNameTranslation: String,
19     @SerialName("revelationType")
20     val revelationType: String,
21     val numberOfAyahs: Int
22 )
23
24 @Serializable
25 data class SurahResponse(
26     val code: Int,
27     val status: String,
28     val data: List<Surah>
29 )
30
31 sealed class ApiResponse<out T> {
```

```
32      data class Success<out T>(val data: T) :
    ApiResponse<T>()
33      data class Error(val errorMessage: String) :
    ApiResponse<Nothing>()
34  }
```

8. ApiClient.kt

Tabel 8. Source Code ApiClient.kt

```
1   package com.allano.alquran.data.network
2
3   import
    com.jakewharton.retrofit2.converter.kotlinx.serialization.asConverterFactory
4   import kotlinx.serialization.json.Json
5   import okhttp3.OkHttpClient
6   import okhttp3.logging.HttpLoggingInterceptor
7   import okhttp3.MediaType.Companion.toMediaType
8   import retrofit2.Retrofit
9   import java.util.concurrent.TimeUnit
10
11  object ApiClient {
12      private const val BASE_URL = "https://api.alquran.cloud/"
13
14      private val json = Json { ignoreUnknownKeys = true }
15
16      private val loggingInterceptor = HttpLoggingInterceptor().apply {
17          level = HttpLoggingInterceptor.Level.BODY
18      }
19
20      private val client = OkHttpClient.Builder()
21          .addInterceptor(loggingInterceptor)
22          .connectTimeout(30, TimeUnit.SECONDS)
23          .readTimeout(30, TimeUnit.SECONDS)
24          .build()
25
26      val apiService: ApiService by lazy {
27          Retrofit.Builder()
28              .baseUrl(BASE_URL)
    .addConverterFactory(json.asConverterFactory("application/json".toMediaType()))
29              .client(client)
30              .build()
31              .create(ApiService::class.java)
32      }
33  }
```

9. ApiService.kt

Tabel 9. Source Code ApiService.kt

```
1  package com.allano.alquran.data.network
2
3  import com.allano.alquran.data.model.SurahResponse
4  import
   com.allano.alquran.data.model.MultiEditionSurahDetailResponse
5  import retrofit2.Response
6  import retrofit2.http.GET
7  import retrofit2.http.Path
8
9
10 interface ApiService {
11     @GET("v1/surah")
12     suspend fun getAllSurahs(): Response<SurahResponse>
13
14     @GET("v1/surah/{number}/editions/quran-uthmani,en.asad")
15     suspend fun getSurahDetailWithMultipleEditions(
16         @Path("number") surahNumber: Int
17     ): Response<MultiEditionSurahDetailResponse>
18
19 }
```

10. SurahRepository.kt

Tabel 10. Source Code SurahRepository.kt

```
1  package com.allano.alquran.data
2
3  import android.util.Log
4  import com.allano.alquran.data.local.AyahDao
5  import com.allano.alquran.data.local.AyahEntity
6  import com.allano.alquran.data.local.SurahDao
7  import com.allano.alquran.data.local.SurahEntity
8  import com.allano.alquran.data.model.MergedAyah
9  import com.allano.alquran.data.model.Surah
10 import com.allano.alquran.data.network.ApiClient
11 import kotlinx.coroutines.flow.Flow
12
13 class SurahRepository(private val surahDao: SurahDao, private val
   ayahDao: AyahDao) {
14
15
16     fun getSurahsStream(): Flow<List<SurahEntity>> =
   surahDao.getAllSurahs()
17
18     suspend fun refreshSurahs() {
19         try {
20             Log.d("SurahRepository", "Mencoba mengambil data dari
```

```
    network...")
21              val response = ApiClient.apiService.getAllSurahs()
22
23              if (response.isSuccessful) {
24                  val surahsFromApi = response.body()?.data ?:
    emptyList()
25                  Log.d("SurahRepository", "Sukses mengambil
    ${surahsFromApi.size} surah dari API")
26                  if (surahsFromApi.isNotEmpty()) {
27                      val surahEntities = surahsFromApi.map {
    it.toEntity() }
28                      surahDao.insertAll(surahEntities)
29                      Log.d("SurahRepository", "Data berhasil disimpan
    ke Room.")
30                  }
31              } else {
32                  Log.e("SurahRepository", "API Error:
    ${response.code()} ${response.message()}")
33              }
34          } catch (e: Exception) {
35              Log.e("SurahRepository", "Network Error: ${e.message}",
    e)
36          }
37      }
38      fun getAyahStream(surahNumber: Int): Flow<List<AyahEntity>> {
39          return ayahDao.getAyahsBySurahNumber(surahNumber)
40      }
41      suspend fun refreshSurahDetail(surahNumber: Int) {
42          try {
43              val response =
    ApiClient.apiService.getSurahDetailWithMultipleEditions(surahNumber)
44              if (response.isSuccessful) {
45                  val responseData = response.body()?.data
46                  val arabicEdition = responseData?.getOrNull(0)
47                  val englishEdition = responseData?.getOrNull(1)
48
49                  if (arabicEdition != null && englishEdition != null)
    {
50                      val mergedAyahs =
    arabicEdition.ayahs.zip(englishEdition.ayahs).map { (arabicAyah,
    englishAyah) ->
51                          AyahEntity(
52                              number = arabicAyah.number,
53                              surahNumber = surahNumber,
54                              numberInSurah =
    arabicAyah.numberInSurah,
55                              arabicText = arabicAyah.text,
```

```
                                englishText = englishAyah.text
                            )
                        }
                        ayahDao.insertAll(mergedAyahs)
                        Log.d("SurahRepository", "Detail ayat untuk surah $surahNumber disimpan ke DB.")
                    }
                }
            } catch (e: Exception) {
                Log.e("SurahRepository", "Gagal me-refresh detail surah: ", e)
            }
        }
        suspend fun getSurahDetailFromApi(surahNumber: Int): List<MergedAyah>? {
            Log.d("RepoDetail", "Memulai pengambilan detail untuk surah: $surahNumber")
            return try {
                val response = ApiClient.apiService.getSurahDetailWithMultipleEditions(surahNumber)
                Log.d("RepoDetail", "Panggilan API berhasil (isSuccessful): ${response.isSuccessful}")

                if (response.isSuccessful) {
                    val responseData = response.body()?.data
                    Log.d("RepoDetail", "Jumlah edisi yang diterima dari API: ${responseData?.size}")

                    val arabicEdition = responseData?.getOrNull(0)
                    val englishEdition = responseData?.getOrNull(1)
                    Log.d("RepoDetail", "Edisi Arab ditemukan: ${arabicEdition != null}")
                    Log.d("RepoDetail", "Edisi Inggris ditemukan: ${englishEdition != null}")

                    if (arabicEdition != null && englishEdition != null) {
                        val mergedList = arabicEdition.ayahs.zip(englishEdition.ayahs).map { (arabicAyah, englishAyah) ->
                            MergedAyah(
                                numberInSurah = arabicAyah.numberInSurah,
                                arabicText = arabicAyah.text,
                                englishText = englishAyah.text
                            )
                        }
```

```
90                      Log.d("RepoDetail", "Berhasil menggabungkan
    ${mergedList.size} ayat.")
91                      return mergedList
92                  } else {
93                      Log.e("RepoDetail", "Salah satu atau kedua edisi
    (Arab/Inggris) tidak ditemukan dalam respons API.")
94                      return null
95                  }
96              } else {
97                  Log.e("RepoDetail", "Panggilan API tidak sukses.
    Kode: ${response.code()}")
98                  return null
99              }
100         } catch (e: Exception) {
101             Log.e("RepoDetail", "Terjadi EXCEPTION saat mengambil
    detail: ", e)
102             return null
103         }
104     }
105     private fun Surah.toEntity(): SurahEntity {
106         return SurahEntity(
107             number = this.number,
108             name = this.name,
109             englishName = this.englishName,
110             englishNameTranslation = this.englishNameTranslation,
111             revelationType = this.revelationType,
112             numberOfAyahs = this.numberOfAyahs
113         )
114     }
115 }
```

11. AyahAdapter.kt

Tabel 11. Source Code AyahAdapter.kt

```
1   package com.allano.alquran.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 com.allano.alquran.data.local.AyahEntity
9   import com.allano.alquran.databinding.ItemAyahBinding
10
11  class AyahAdapter : ListAdapter<AyahEntity,
    AyahAdapter.AyahViewHolder>(AyahDiffCallback()) {
12
13      override fun onCreateViewHolder(parent: ViewGroup,
```

```
    viewType: Int): AyahViewHolder {
14          val binding =
    ItemAyahBinding.inflate(LayoutInflater.from(parent.context),
    parent, false)
15          return AyahViewHolder(binding)
16      }
17
18      override fun onBindViewHolder(holder: AyahViewHolder,
    position: Int) {
19          holder.bind(getItem(position))
20      }
21
22      class AyahViewHolder(private val binding:
    ItemAyahBinding) : RecyclerView.ViewHolder(binding.root) {
23          fun bind(ayahEntity: AyahEntity) {
24              binding.tvAyahNumber.text =
    ayahEntity.numberInSurah.toString()
25              binding.tvAyahTextArabic.text =
    ayahEntity.arabicText
26              binding.tvAyahTextEnglish.text =
    ayahEntity.englishText
27          }
28      }
29  }
30
31  class AyahDiffCallback : DiffUtil.ItemCallback<AyahEntity>()
    {
32      override fun areItemsTheSame(oldItem: AyahEntity,
    newItem: AyahEntity): Boolean {
33          return oldItem.number == newItem.number
34      }
35
36      override fun areContentsTheSame(oldItem: AyahEntity,
    newItem: AyahEntity): Boolean {
37          return oldItem == newItem
38      }
39  }
```

12. DetailFragment.kt

Tabel 12. Source Code DetailFragment.kt

```
1   package com.allano.alquran.ui
2
3   import android.os.Bundle
4   import android.util.Log
5   import android.view.LayoutInflater
6   import android.view.View
7   import android.view.ViewGroup
```

```
8  import androidx.core.view.isVisible
9  import androidx.fragment.app.Fragment
10 import androidx.fragment.app.activityViewModels
11 import androidx.fragment.app.viewModels
12 import androidx.lifecycle.lifecycleScope
13 import androidx.navigation.fragment.navArgs
14 import androidx.recyclerview.widget.LinearLayoutManager
15 import
   com.allano.alquran.databinding.FragmentDetailBinding
16 import kotlinx.coroutines.flow.collectLatest
17 import kotlinx.coroutines.launch
18
19 class DetailFragment : Fragment() {
20
21     private var _binding: FragmentDetailBinding? = null
22     private val binding get() = _binding!!
23
24     private val args: DetailFragmentArgs by navArgs()
25
26     private val detailViewModel: DetailViewModel by
   viewModels {
27
28 DetailViewModel.Factory(requireActivity().application,
   args.surahNumber)
29     }
30
31     private val surahViewModel: SurahViewModel by
   activityViewModels()
32
33     private lateinit var ayahAdapter: AyahAdapter
34
35     override fun onCreateView(
36         inflater: LayoutInflater, container: ViewGroup?,
37         savedInstanceState: Bundle?
       ): View {
38         _binding =
   FragmentDetailBinding.inflate(inflater, container, false)
39         return binding.root
40     }
41
42     override fun onViewCreated(view: View,
   savedInstanceState: Bundle?) {
43         super.onViewCreated(view, savedInstanceState)
44
45         setupInfoAwal()
46         setupRecyclerView()
47         observeAyahs()
```

```
48      }
49
50      private fun setupInfoAwal() {
51          val surah = surahViewModel.surahs.value.find { it.number == args.surahNumber }
52          surah?.let {
53              binding.tvDetailSurahName.text = it.englishName
54              binding.tvDetailSurahArabicName.text = it.name
55              binding.tvDetailTranslation.text = "\"${it.englishNameTranslation}\""
56              binding.tvDetailInfo.text = "Surah No: ${it.number} • ${it.revelationType} • ${it.numberOfAyahs} Verses"
57
58          }
59      }
60
61      private fun setupRecyclerView() {
62          ayahAdapter = AyahAdapter()
63          binding.rvAyahs.apply {
64              adapter = ayahAdapter
65              layoutManager = LinearLayoutManager(requireContext())
66          }
67      }
68
69      private fun observeAyahs() {
70          viewLifecycleOwner.lifecycleScope.launch {
71              detailViewModel.ayahs.collectLatest { ayahsList ->
72                  Log.d("DetailFragmentUI", "Menerima update dengan ${ayahsList.size} ayat.")
73
74                  binding.detailProgressBar.isVisible = ayahsList.isEmpty()
75
76                  ayahAdapter.submitList(ayahsList)
77              }
78          }
79      }
80
81      override fun onDestroyView() {
81          super.onDestroyView()
83          _binding = null
84
```

```
85      }
    }
```

13. DetailViewModel.kt

Tabel 13. Source Code DetailViewModel.kt

```
1   package com.allano.alquran.ui
2
3   import android.app.Application
4   import androidx.lifecycle.*
5   import com.allano.alquran.data.SurahRepository
6   import com.allano.alquran.data.local.AppDatabase
7   import com.allano.alquran.data.local.AyahEntity
8   import kotlinx.coroutines.flow.SharingStarted
9   import kotlinx.coroutines.flow.StateFlow
10  import kotlinx.coroutines.flow.stateIn
11  import kotlinx.coroutines.launch
12
13  class DetailViewModel(application: Application, private
    val surahNumber: Int) : AndroidViewModel(application) {
14
15      private val repository: SurahRepository
16
17      val ayahs: StateFlow<List<AyahEntity>>
18
19      init {
20          val surahDao =
    AppDatabase.getDatabase(application).surahDao()
21          val ayahDao =
    AppDatabase.getDatabase(application).ayahDao()
22          repository = SurahRepository(surahDao, ayahDao)
23
24          ayahs =
    repository.getAyahStream(surahNumber).stateIn(
25              scope = viewModelScope,
26              started =
    SharingStarted.WhileSubscribed(5000),
27              initialValue = emptyList()
28          )
29
30          viewModelScope.launch {
31              repository.refreshSurahDetail(surahNumber)
32          }
33      }
34
35      class Factory(private val app: Application, private
    val surahNumber: Int) : ViewModelProvider.Factory {
36          override fun <T : ViewModel> create(modelClass:
```

```
    Class<T>): T {
37              if
    (modelClass.isAssignableFrom(DetailViewModel::class.java))
    {
38                  @Suppress("UNCHECKED_CAST")
39                  return DetailViewModel(app, surahNumber)
    as T
40              }
41              throw IllegalArgumentException("Unknown
    ViewModel class")
42          }
43      }
44  }
```

14. HomeFragment.kt

Tabel 14. Source Code HomeFragment.kt

```
1   package com.allano.alquran.ui
2
3   import android.os.Bundle
4   import android.util.Log
5   import android.view.LayoutInflater
6   import android.view.View
7   import android.view.ViewGroup
8   import androidx.core.view.isVisible
9   import androidx.fragment.app.Fragment
10  import androidx.lifecycle.lifecycleScope
11  import androidx.navigation.fragment.findNavController
12  import androidx.recyclerview.widget.LinearLayoutManager
13  import kotlinx.coroutines.flow.collectLatest
14  import kotlinx.coroutines.launch
15  import com.allano.alquran.databinding.FragmentHomeBinding
16  import androidx.fragment.app.activityViewModels
17
18  class HomeFragment : Fragment() {
19
20      private var _binding: FragmentHomeBinding? = null
21      private val binding get() = _binding!!
22
23      private val viewModel: SurahViewModel by activityViewModels {
24          SurahViewModel.Factory(requireActivity().application)
25      }
26
27      override fun onCreateView(
28          inflater: LayoutInflater, container: ViewGroup?,
29          savedInstanceState: Bundle?
30      ): View {
```

```
31          _binding = FragmentHomeBinding.inflate(inflater, container,
    false)
32          return binding.root
33      }
34
35      override fun onViewCreated(view: View, savedInstanceState:
    Bundle?) {
36          super.onViewCreated(view, savedInstanceState)
37
38          val surahAdapter = SurahAdapter(viewModel)
39          binding.recyclerView.apply {
40              layoutManager = LinearLayoutManager(context)
41              adapter = surahAdapter
42          }
43
44          viewLifecycleOwner.lifecycleScope.launch {
45              binding.progressBar.isVisible = true
46
47              viewModel.surahs.collectLatest { surahs ->
48                  Log.d("HomeFragment", "Updating UI with
    ${surahs.size} surahs.")
49
50                  binding.progressBar.isVisible = false
51
52                  if (surahs.isNotEmpty()) {
53                      binding.tvError.isVisible = false
54                      surahAdapter.submitList(surahs)
55                  } else {
56                      binding.tvError.isVisible = true
57                  }
58              }
59          }
60
61          viewLifecycleOwner.lifecycleScope.launch {
62              viewModel.navigateToDetail.collect { surah ->
63                  surah?.let {
64                      val action =
    HomeFragmentDirections.actionHomeFragmentToDetailFragment(it.number)
65                      findNavController().navigate(action)
66                      viewModel.onDetailNavigated()
67                  }
68              }
69          }
70      }
71
72      override fun onDestroyView() {
73          super.onDestroyView()
```

```
74          _binding = null
75      }
76  }
```

15. SurahAdapter.kt

Tabel 15. Source Code SurahAdapter.kt

```
1   package com.allano.alquran.ui
2
3   import android.util.Log
4   import android.view.LayoutInflater
5   import android.view.ViewGroup
6   import androidx.recyclerview.widget.DiffUtil
7   import androidx.recyclerview.widget.ListAdapter
8   import androidx.recyclerview.widget.RecyclerView
9   import com.allano.alquran.data.local.SurahEntity
10  import com.allano.alquran.databinding.ItemSurahBinding
11
12  class SurahAdapter(private val viewModel: SurahViewModel) :
13      ListAdapter<SurahEntity,
    SurahAdapter.SurahViewHolder>(SurahDiffCallback()) {
14
15      override fun onCreateViewHolder(parent: ViewGroup,
    viewType: Int): SurahViewHolder {
16          val binding =
    ItemSurahBinding.inflate(LayoutInflater.from(parent.context),
    parent, false)
17          return SurahViewHolder(binding)
18      }
19
20      override fun onBindViewHolder(holder: SurahViewHolder,
    position: Int) {
21          val surah = getItem(position)
22          holder.bind(surah)
24          holder.itemView.setOnClickListener {
25              Log.d("SurahAdapter", "Item clicked:
    ${surah.englishName}")
26              viewModel.onSurahClicked(surah)
27          }
28      }
29
30      class SurahViewHolder(private val binding:
    ItemSurahBinding) :
31          RecyclerView.ViewHolder(binding.root) {
32          fun bind(surah: SurahEntity) {
33              binding.tvSurahNumber.text =
    surah.number.toString()
```

```
34              binding.tvSurahName.text = surah.englishName
35              binding.tvSurahTranslation.text =
    surah.englishNameTranslation
36              binding.tvSurahInfo.text =
    "${surah.revelationType} - ${surah.numberOfAyahs} verses"
37          }
38      }
39  }
40
41  class SurahDiffCallback :
    DiffUtil.ItemCallback<SurahEntity>() {
42      override fun areItemsTheSame(oldItem: SurahEntity,
    newItem: SurahEntity): Boolean {
43          return oldItem.number == newItem.number
44      }
45
46      override fun areContentsTheSame(oldItem: SurahEntity,
    newItem: SurahEntity): Boolean {
47          return oldItem == newItem
48      }
49  }
```

16. SurahViewModel.kt

Tabel 16. Source Code SurahViewModel.kt

```
1   package com.allano.alquran.ui
2
3   import android.app.Application
4   import android.util.Log
5   import androidx.lifecycle.*
6   import com.allano.alquran.data.SurahRepository
7   import com.allano.alquran.data.local.AppDatabase
8   import com.allano.alquran.data.local.SurahEntity
9   import com.allano.alquran.data.model.MergedAyah
10  import kotlinx.coroutines.flow.MutableStateFlow
11  import kotlinx.coroutines.flow.StateFlow
12  import kotlinx.coroutines.flow.catch
14  import kotlinx.coroutines.launch
15
16  class SurahViewModel(application: Application) :
    AndroidViewModel(application) {
18
19      private val repository: SurahRepository
20
21      private val _surahs =
    MutableStateFlow<List<SurahEntity>>(emptyList())
22      val surahs: StateFlow<List<SurahEntity>> = _surahs
```

```
23
24      private val _navigateToDetail =
    MutableStateFlow<SurahEntity?>(null)
25      val navigateToDetail: StateFlow<SurahEntity?> =
    _navigateToDetail
26
27      init {
28          val database =
    AppDatabase.getDatabase(application)
29          val surahDao = database.surahDao()
30          val ayahDao = database.ayahDao()
31          repository = SurahRepository(surahDao, ayahDao)
32          listenToDatabase()
33          refreshDataFromServer()
34      }
35
36      private fun listenToDatabase() {
37          viewModelScope.launch {
38              repository.getSurahsStream()
39                  .catch { e ->
40                      Log.e("SurahViewModel", "Error
    collecting from DB", e)
41                  }
42                  .collect { surahList ->
43                      _surahs.value = surahList
44                      Log.d("SurahViewModel", "Menerima
    ${surahList.size} surah dari database.")
45                  }
46          }
47      }
48
49      private fun refreshDataFromServer() {
50          viewModelScope.launch {
51              repository.refreshSurahs()
52          }
53      }
54
55      suspend fun getSurahDetailFromApi(surahNumber: Int):
    List<MergedAyah>? {
56          return
    repository.getSurahDetailFromApi(surahNumber)
57      }
58
59      fun onSurahClicked(surah: SurahEntity) {
60          _navigateToDetail.value = surah
61      }
62
```

```
63      fun onDetailNavigated() {
64          _navigateToDetail.value = null
65      }
66
67      class Factory(private val app: Application) :
    ViewModelProvider.Factory {
68          override fun <T : ViewModel> create(modelClass:
    Class<T>): T {
69              if
    (modelClass.isAssignableFrom(SurahViewModel::class.java))
70  {
71                  @Suppress("UNCHECKED_CAST")
72                  return SurahViewModel(app) as T
73              }
74              throw IllegalArgumentException("Unable to
    construct viewmodel")
75          }
76      }
77  }
```

17. MainActivity.kt

Tabel 17. Source Code MainActivity.kt

```
1   package com.allano.alquran
2
3   import androidx.appcompat.app.AppCompatActivity
4   import android.os.Bundle
5
6   class MainActivity : AppCompatActivity() {
7       override fun onCreate(savedInstanceState: Bundle?) {
8           super.onCreate(savedInstanceState)
9           setContentView(R.layout.activity_main)
10      }
11  }
```

18. activity_main.xml

Tabel 18. Source Code activity_main.xml

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <androidx.fragment.app.FragmentContainerView
    xmlns:android="http://schemas.android.com/apk/res/android"
3       xmlns:app="http://schemas.android.com/apk/res-auto"
4       android:id="@+id/nav_host_fragment"
5       android:layout_width="match_parent"
6       android:layout_height="match_parent"
7
    android:name="androidx.navigation.fragment.NavHostFragment"
8       app:defaultNavHost="true"
9
```

```
10      app:navGraph="@navigation/nav_graph"
        />
```

19. fragment_detail.xml

Tabel 19. Source Code fragment_detail.xml

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <androidx.core.widget.NestedScrollView
    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=".ui.DetailFragment">
8
9       <LinearLayout
10          android:layout_width="match_parent"
11          android:layout_height="wrap_content"
12          android:orientation="vertical">
13
14          <androidx.constraintlayout.widget.ConstraintLayout
15              android:layout_width="match_parent"
16              android:layout_height="wrap_content"
17              android:padding="24dp">
18
19              <TextView
20                  android:id="@+id/tv_detail_surah_name"
21                  android:layout_width="0dp"
22                  android:layout_height="wrap_content"
23
    android:textAppearance="?attr/textAppearanceHeadline4"
24                  android:textStyle="bold"
25                  app:layout_constraintEnd_toEndOf="parent"
26
    app:layout_constraintStart_toStartOf="parent"
27                  app:layout_constraintTop_toTopOf="parent"
28                  tools:text="Al-Fatiha" />
29
30              <TextView
31
    android:id="@+id/tv_detail_surah_arabic_name"
32                  android:layout_width="0dp"
33                  android:layout_height="wrap_content"
34                  android:layout_marginTop="8dp"
35
    android:textAppearance="?attr/textAppearanceHeadline5"
36                  app:layout_constraintEnd_toEndOf="parent"
37
```

```
    app:layout_constraintStart_toStartOf="parent"
38
    app:layout_constraintTop_toBottomOf="@+id/tv_detail_surah_
    name"
39                  tools:text="سُورَةُ ٱلْفَـاتِـحَةِ" />
40
41              <TextView
42                  android:id="@+id/tv_detail_translation"
43                  android:layout_width="0dp"
44                  android:layout_height="wrap_content"
45                  android:layout_marginTop="4dp"
46
    android:textAppearance="?attr/textAppearanceBody1"
47                  android:textStyle="italic"
48                  app:layout_constraintEnd_toEndOf="parent"
49
    app:layout_constraintStart_toStartOf="parent"
50
    app:layout_constraintTop_toBottomOf="@+id/tv_detail_surah_
    arabic_name"
51                  tools:text="&amp;quot;The Opening&amp;quot;" />
52
53              <TextView
54                  android:id="@+id/tv_detail_info"
55                  android:layout_width="0dp"
56                  android:layout_height="wrap_content"
57                  android:layout_marginTop="16dp"
58
    android:textAppearance="?attr/textAppearanceBody2"
59
    android:textColor="?android:attr/textColorSecondary"
60                  app:layout_constraintEnd_toEndOf="parent"
61
    app:layout_constraintStart_toStartOf="parent"
62
    app:layout_constraintTop_toBottomOf="@+id/tv_detail_transl
    ation"
63                  tools:text="Surah No: 1 • Meccan • 7
    Verses" />
64
65
    </androidx.constraintlayout.widget.ConstraintLayout>
66
67          <ProgressBar
68              android:id="@+id/detail_progress_bar"
69              style="?android:attr/progressBarStyle"
70              android:layout_width="wrap_content"
```

```
71              android:layout_height="wrap_content"
72              android:layout_gravity="center_horizontal"
73              android:layout_marginTop="16dp"
74              android:visibility="visible"
75              tools:visibility="gone" />
76
77          <androidx.recyclerview.widget.RecyclerView
78              android:id="@+id/rv_ayahs"
79              android:layout_width="match_parent"
80              android:layout_height="wrap_content"
81              android:nestedScrollingEnabled="false"
82              tools:itemCount="5"
83              tools:listitem="@layout/item_ayah" />
84
85      </LinearLayout>
86  </androidx.core.widget.NestedScrollView>
```

20. fragment_home.xml

Tabel 20. Source Code fragment_home.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=".ui.HomeFragment">
8
9       <androidx.recyclerview.widget.RecyclerView
10          android:id="@+id/recycler_view"
11          android:layout_width="0dp"
12          android:layout_height="0dp"
13          android:clipToPadding="false"
14          android:paddingVertical="4dp"
15          android:contentDescription="FUCK THE BLINDS"
16          app:layout_constraintBottom_toBottomOf="parent"
17          app:layout_constraintEnd_toEndOf="parent"
18          app:layout_constraintStart_toStartOf="parent"
19          app:layout_constraintTop_toTopOf="parent" />
20
21      <ProgressBar
22          android:id="@+id/progress_bar"
23          style="?android:attr/progressBarStyle"
24          android:layout_width="wrap_content"
25          android:layout_height="wrap_content"
26          android:visibility="gone"
27          app:layout_constraintBottom_toBottomOf="parent"
```

```
28          app:layout_constraintEnd_toEndOf="parent"
29          app:layout_constraintStart_toStartOf="parent"
30          app:layout_constraintTop_toTopOf="parent"
31          android:contentDescription="Fuck the blinds"
32          tools:visibility="visible"/>
33
34      <TextView
35          android:id="@+id/tv_error"
36          android:layout_width="wrap_content"
37          android:layout_height="wrap_content"
38          android:text="Loading."
39          android:visibility="gone"
40          app:layout_constraintBottom_toBottomOf="parent"
41          app:layout_constraintEnd_toEndOf="parent"
42          app:layout_constraintStart_toStartOf="parent"
43          app:layout_constraintTop_toTopOf="parent"
44          tools:visibility="visible"/>
45
46  </androidx.constraintlayout.widget.ConstraintLayout>
```

21. item_ayah.xml

Tabel 21. Source Code item_ayah.xml

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
3       xmlns:tools="http://schemas.android.com/tools"
4       android:layout_width="match_parent"
5       android:layout_height="wrap_content"
6       android:orientation="vertical"
7       android:paddingHorizontal="16dp"
8       android:paddingVertical="12dp">
9
10      <TextView
11          android:id="@+id/tv_ayah_number"
12          android:layout_width="wrap_content"
13          android:layout_height="wrap_content"
14          android:layout_gravity="start"
15          android:paddingHorizontal="8dp"
16          android:paddingVertical="2dp"
17          android:textColor="?attr/colorPrimary"
18          android:textStyle="bold"
19          tools:text="1" />
20
21      <TextView
22          android:id="@+id/tv_ayah_text_arabic"
23          android:layout_width="match_parent"
24          android:layout_height="wrap_content"
```

```
25          android:layout_marginTop="8dp"
26          android:gravity="end"
27          android:lineSpacingMultiplier="1.5"
28      android:textAppearance="?attr/textAppearanceHeadline6"
29          android:textColor="?android:attr/textColorPrimary"
30          tools:text="بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ"
31          />
32
33      <TextView
34          android:id="@+id/tv_ayah_text_english"
35          android:layout_width="match_parent"
36          android:layout_height="wrap_content"
37          android:layout_marginTop="4dp"
38          android:lineSpacingMultiplier="1.2"
39          android:textAppearance="?attr/textAppearanceBody2"
40
    android:textColor="?android:attr/textColorSecondary"
41          tools:text="In the name of Allah, the Entirely
    Merciful, the Especially Merciful." />
42
43  </LinearLayout>
```

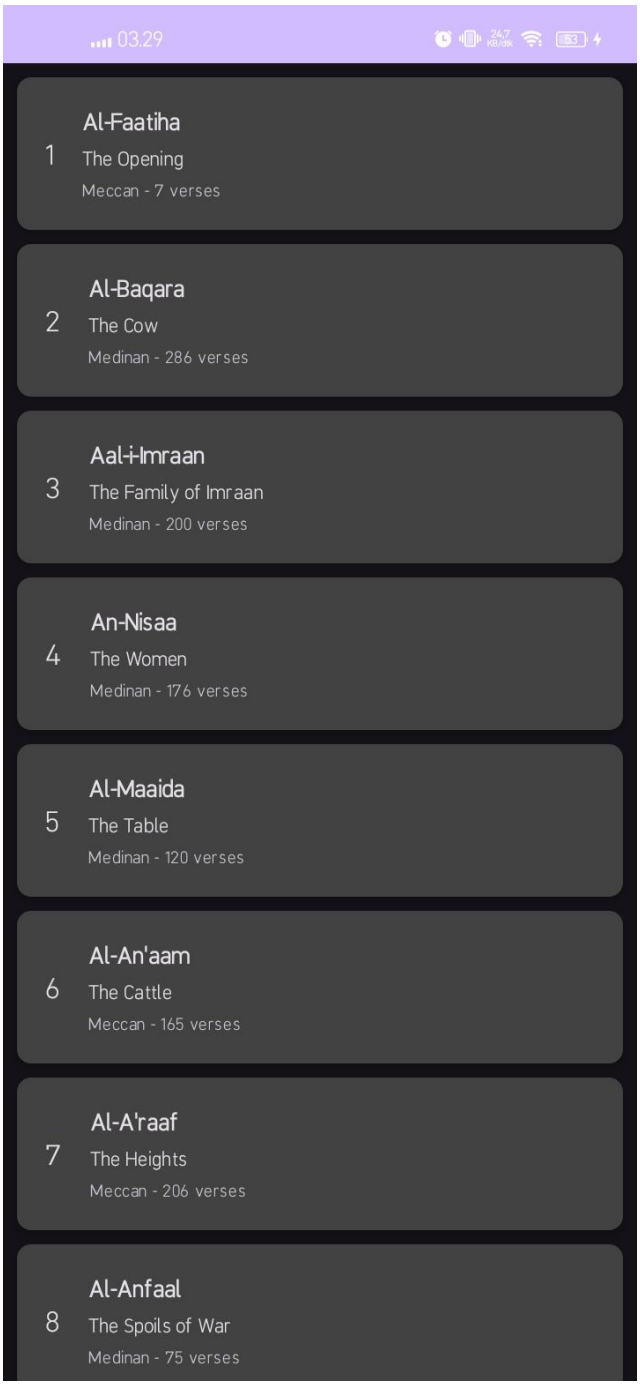
22. item_surah.xml

Tabel 22. Source Code item_surah.xml

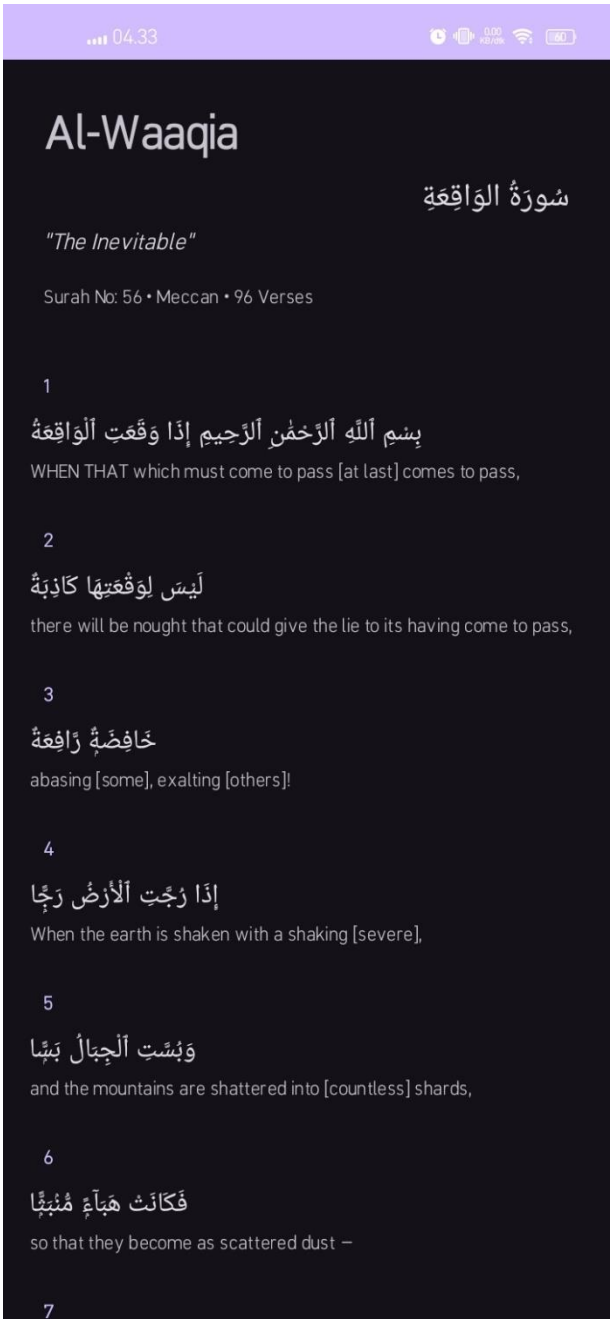
```
1   <?xml version="1.0" encoding="utf-8"?>
2   <androidx.cardview.widget.CardView
    xmlns:android="http://schemas.android.com/apk/res/android"
3       xmlns:app="http://schemas.android.com/apk/res-auto"
4       xmlns:tools="http://schemas.android.com/tools"
5       android:layout_width="match_parent"
6       android:layout_height="wrap_content"
7       android:layout_marginHorizontal="8dp"
8       android:layout_marginVertical="4dp"
9       app:cardCornerRadius="8dp"
10      app:cardElevation="4dp">
11
12      <androidx.constraintlayout.widget.ConstraintLayout
13          android:layout_width="match_parent"
14          android:layout_height="wrap_content"
15          android:padding="16dp">
16
17          <TextView
18              android:id="@+id/tvSurahNumber"
19              android:layout_width="wrap_content"
20              android:layout_height="wrap_content"
21  android:textAppearance="?attr/textAppearanceHeadline6"
22              app:layout_constraintStart_toStartOf="parent"
```

```
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintBottom_toBottomOf="parent"
            tools:text="1" />

        <TextView
            android:id="@+id/tvSurahName"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginStart="16dp"
android:textAppearance="?attr/textAppearanceListItem"
            android:textStyle="bold"
            app:layout_constraintEnd_toEndOf="parent"
      app:layout_constraintStart_toEndOf="@id/tvSurahNumber"
            app:layout_constraintTop_toTopOf="parent"
            tools:text="Al-Fatiha" />

        <TextView
            android:id="@+id/tvSurahTranslation"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginTop="4dp"
android:textAppearance="?attr/textAppearanceBody2"
         app:layout_constraintEnd_toEndOf="@id/tvSurahName"
app:layout_constraintStart_toStartOf="@id/tvSurahName"
app:layout_constraintTop_toBottomOf="@id/tvSurahName"
            tools:text="The Opening" />

        <TextView
            android:id="@+id/tvSurahInfo"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginTop="4dp"
android:textAppearance="?attr/textAppearanceCaption"
app:layout_constraintStart_toStartOf="@id/tvSurahName"
app:layout_constraintTop_toBottomOf="@id/tvSurahTranslation"
            tools:text="Meccan - 7 verses" />

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

B. Output Program



Gambar 1. Screenshot Halaman Utama



Gambar 2. Screenshot Halaman Detail

C. Pembahasan

1. SurahEntity.kt dan AyahEntity.kt:

File ini berfungsi sebagai skema untuk tabel dalam database SQLite. Anotasi @Entity memberitahu Room bahwa data class ini harus diubah menjadi tabel, setiap property di dalam class, seperti number, arabicText, dan atribut lainnya akan menjadi kolom dalam tabel

tersebut. Anotasi @PrimaryKey menandakan kolom yang nilainya unik untuk setiap baris (Unique Key) yang digunakan sebagai pengenal utama. Di AyahEntity.kt, didefinisikan @ForeignKey sebagai penghubung relasional ke tabel surah untuk menjaga integritas data

2. **SurahDao.kt dan AyahDao.kt:**

Data Access Object (DAO) adalah interface yang berisi daftar fungsi untuk berinteraksi dengan database. Dengan anotasi @Dao, room akan secara otomatis membuatkan implementasi dari setiap fungsi yang didefinisikan. Anotasi @Query berfungsi untuk menulis perintah SQL yang akan divalidasi oleh room. Anotasi @Insert dengan OnConflictStrategy.REPLACE akan menimpa data lama jika ada data baru dengan Primary Key yang sama. Semua fungsi Input/Output ditandai sebagai suspend agar bisa dipanggil coroutine tanpa memblokir thread utama, sementara fungsi yang mengembalikan Flow menyediakan stream data yang otomatis mengirimkan data terbaru ke subscribernya setiap ada perubahan di tabel.

3. **AppDatabase.kt**

Anotasi @Database mendaftarkan semua kelas @Entity yang menjadi bagian dari skema databse dan juga mendefinisikan nomor version dari database tersebut. Nomor versi wajib dinaikkan setiap kali ada perubahan di struktur tabel. Metode .fallbackToDestructioveMigration() memerintahkan Room untuk menghapus dan membuat ulang seluruh database jika terdeteksi kenaikan versi dan tidak ada skrip migrasi spesifik yang diberikan.

4. **Ayah.kt dan SurahDetail.kt**

File-file ini berisi data class yang strukturnya berbentuk JSON yang dikirimkan API. Anotasi @Serializabel dari library KotlinX Serialization memungkinkan proses deserialization yang mengkonversi teks JSON menjadi objek Kotlin yang bisa digunakan dalam kode ini. Anotasi @SerialName digunakan sebagai pemetaan jika nama field di JSON berbeda dengan nama property di kelas Kotlin

5. **ApiService.kt**

Fungsi didalamnya merepresentasikan satu endpoint API. Anotasi @GET mendefinisikan tipe request HTTP dan relative pathnya. Anotasi @Path digunakan untuk menyisipkan nilai dinamis ke URL seperti nomor surah. Seluruh fungsi didefinisikan sebagai suspend karena panggilan jaringan adalah operasi yang harus dijalankan diluar thread utama.

6. **ApiClient.kt**

File ini bertanggung jawab untuk membuat dan mengkonfigurasi satu instance Retrofit untuk seluruh aplikasi. Didalamnya didefinisikan baseUrl dari API, konfigurasi OkHttpClient, dan ConverterFactory yang memberitahu Retrofit bagaimana memparsing JSON menjadi objek Kotlin menggunakan Serialization.

7. **SurahRepository.kt:**

Repository bertindak sebagai mediator antar sumber data (jaringan dan lokal) dan bagian dari viewModel. SurahRepository mengimplementasikan pola offline-first dengan menyediakan fungsi get…Stream yang mengembalikan Flow langsung dari DAO sebagai sumber Tunggal untuk UI, dan fungsi refresh yang merupakan fungsi suspend untuk mengambil data terbaru dari jaringan dan menyimpannya ke database lokal, yang kemudian akan diupdate secara otomatis.

8. **SurahViewModel.kt dan DetailViewModel.kt:**

ViewModel menyimpan dan mengelola data yang berhubungan dengan UI. Data didalamnya akan selamat dari perubahan konfigurasi, seperti perubahan rotasi layar. StateFlow digunakan untuk menampung state seperti daftar surah dan mengeksposnya ke UI. Operasi yang akan berjalan lama seperti meminta data ke repository akan dijalankan dalam viewModelScope, yang secara otomatis akan membatalkan semua tugasnya jika viewmodel dihancurkan.

9. **HomeFragment.kt dan DetailFragment.kt:**

Dalam onViewCreated, logika untuk menginisialisasi UI ditempatkan. ViewBinding digunakan untuk mengakses elemen dari XML secara aman. Fungsi utamanya adalam mengamati StateFlow dari Viewmodel yang menggunakan viewLifecycleOwner.lifecycleScope. setiap ada data baru dari ViewModel, blok collect akan dieksekusi untuk memperbarui tampilan. File-file ini juga mendeteksi interaksi pengguna seperti onClick dan meneruskannya ke viewModel.

10. **SurahAdapter.kt dan ListAdapter.kt:**

Adapter berguna sebagai jembatan data dan RecyclerView. ListAdapter di extend untuk menampilkan daftar data yang bisa berubah. ListAdapter menggunakan DiffUtil.ItemCallback untuk menghitung perbedaan antara data lama dan data baru lalu secara otomatis memperbaruinya. Didalamnya ada sebuah template yang memegang referensi ke item di file XML.

11. **Layout.xml**

File-file layout XML mendefinisikan tampilan visual aplikasi secara deklaratif seperti CSS. Fragment_home.xml dan fragment_detail.xml menampilkan struktur komponen untuk masing masing layar, sementara item_surah.xml dan item_ayah.xml mendefinisikan tampilan perbaris.

Tautan Git

Berikut adalah tautan untuk source code yang telah dibuat.
https://github.com/AllanoLintang/pemro-mobile/tree/main/modul4