LAPORAN AKHIR PRAKTIKUM
PEMROGRAMAN MOBILE

Oleh:

Hari Octavian Delrossi NIM. 2210817210033

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
JUNI 2024

LEMBAR PENGESAHAN
LAPORAN AKHIR PRAKTIKUM PEMROGRAMAN MOBILE

Laporan Akhir Praktikum Pemrograman Mobile ini disusun sebagai syarat lulus mata kuliah Praktikum Pemrograman Mobile. Laporan Akhir Praktikum ini dikerjakan oleh:

Nama Praktikan : Hari Octavian Delrossi
NIM : 2210817210033

Menyetujui, Asisten Praktikum	Mengetahui, Dosen Penanggung Jawab Praktikum
Muhammad Asandy Putra NIM. 2110817310002	Muti’a Maulida, S.Kom., M.T.I. NIP. 198810272019032013

DAFTAR ISI

DAFTAR GAMBAR

PRAKTIKUM MODUL 1

PRAKTIKUM MODUL 2

PRAKTIKUM MODUL 3

PRAKTIKUM MODUL 4

PRAKTIKUM MODUL 5

DAFTAR TABEL

PRAKTIKUM MODUL 1

PRAKTIKUM MODUL 2

PRAKTIKUM MODUL 3

PRAKTIKUM MODUL 4

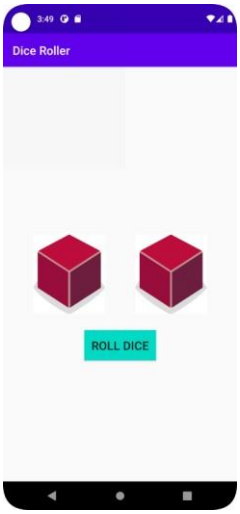
PRAKTIKUM MODUL 5

PRAKTIKUM MODUL 1

SOAL

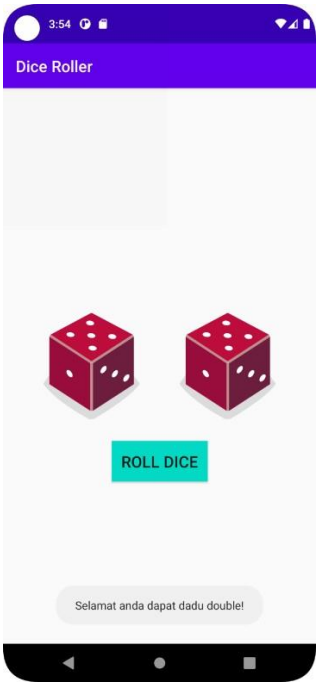
Buatlah sebuah aplikasi yang dapat menampilkan 2 (dua) buah dadu yang dapat berubah-ubah tampilannya pada saat user menekan tombol “Roll Dice”. Aturan aplikasi yang akan dibangun adalah sebagaimana berikut:

1. Tampilan awal aplikasi setelah dijalankan akan menampilkan 2 buah dadu kosong seperti dapat dilihat pada Gambar 1.



2. Setelah user menekan tombol “Roll Dice” maka masing-masing dadu akan memunculkan sisi dadu masing-masing dengan angka antara 1 s/d 6. Apabila user mendapatkan nilai dadu yang berbeda antara Dadu 1 dengan Dadu 2 maka akan menampilkan pesan “Anda belum beruntung!” seperti dapat dilihat pada Gambar 2.

3. Apabila user mendapatkan nilai dadu yang sama antara Dadu 1 dan Dadu 2 atau nilai double, maka aplikasi akan menampilkan pesan “Selamat anda dapat dadu double!” seperti dapat dilihat pada Gambar 3.
4. Upload aplikasi yang telah anda buat ke dalam repository github ke dalam folder Module 2 dalam bentuk project. Jangan lupa untuk melakukan Clean Project sebelum mengupload pekerjaan anda pada repo.
5. Untuk gambar dadu dapat didownload pada link berikut: https://drive.google.com/u/0/uc?id=147HT2lIH5qin3z5ta7H9y2N_5OMW81Ll&ex port=download



A. Source Code

Tabel 1. Source Code Hasil Jawaban MainActivity.kt Modul 1

```
1  package com.example.daduberputar
2
3  import android.os.Bundle import android.widget.Button
4  import android.widget.ImageView
5  import android.widget.Toast
6  import androidx.appcompat.app.AppCompatActivity
7
8  class MainActivity : AppCompatActivity() {
9
10
11
```

```
12      override fun onCreate(savedInstanceState: Bundle?)
13  {
14          super.onCreate(savedInstanceState)
15          setContentView(R.layout.main_activity)
16
17  val  diceImage01:   ImageView =
    findViewById(R.id.imageView)
18
19  diceImage01.setImageResource(R.drawable.dadu_0)
20
21  val  diceImage02:   ImageView =
    findViewById(R.id.imageView2)
22
23  diceImage02.setImageResource(R.drawable.dadu_0)
24
25  val  rollButton:    Button    =
    findViewById(R.id.button)
26
27  rollButton.setOnClickListener { rollDice() }
28  }
29
30  private fun rollDice() {
31  val dice1 = Dice(6)
32  val diceRoll1 = dice1.roll()
33
34  val  diceImage1:    ImageView =
35  findViewById(R.id.imageView)
36  when (diceRoll1) {
37  1    ->
    diceImage1.setImageResource(R.drawable.dice_1)
38  2    ->
    diceImage1.setImageResource(R.drawable.dice_2)
39  3    ->
    diceImage1.setImageResource(R.drawable.dice_3)
40  4    ->
    diceImage1.setImageResource(R.drawable.dice_4)
41  5    ->
    diceImage1.setImageResource(R.drawable.dice_5)
42  6    ->
    diceImage1.setImageResource(R.drawable.dice_6)
43  }
```

```
44
45 val dice2 = Dice(6)
46         val diceRoll2 = dice2.roll()
47
48          val       diceImage2:       ImageView
    =findViewById(R.id.imageView2)
49         when (diceRoll2) {
50             1                                          -
               >
    diceImage2.setImageResource(R.drawable.dice_1)
51             2                                          -
               >
    diceImage2.setImageResource(R.drawable.dice_2)
52             3                                          -
               >
    diceImage2.setImageResource(R.drawable.dice_3)
               4                                          -
53             >
    diceImage2.setImageResource(R.drawable.dice_4)
               5                                          -
54             >
    diceImage2.setImageResource(R.drawable.dice_5)
               6                                          -
55             >
    diceImage2.setImageResource(R.drawable.dice_6)
         }
56
57         if (diceRoll1 == diceRoll2){
58              val toast = Toast.makeText(this,
    "Selamatanda dapat dadu double!",
    Toast.LENGTH_SHORT)
59             toast.show()
60         } else {
61              val toast = Toast.makeText(this, "Anda
    belum beruntung!", Toast.LENGTH_SHORT)
62             toast.show() }
63
64  }
65  }
66
67  class Dice(private val numSides: Int)
68
```

```
69
70      {fun roll(): Int {
            return (1..numSides).random()
        }
}
```

Tabel 2. Source Code Hasil Jawaban GameFragment.kt Modul 1

```
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=".MainActivity">
8
9  <Button
10 android:id="@+id/button"
11 android:layout_width="wrap_content"
12 android:layout_height="wrap_content"
13 android:layout_marginTop="52dp"
14 android:text="Roll"
15 app:layout_constraintEnd_toEndOf="parent"
16 app:layout_constraintHorizontal_bias="0.498"
17 app:layout_constraintStart_toStartOf="parent"
18
19 app:layout_constraintTop_toBottomOf="@+id/imageView" />
20
21 <ImageView
22 android:id="@+id/imageView"
23 android:layout_width="150dp"
24 android:layout_height="150dp"
25 app:layout_constraintBottom_toBottomOf="parent"
26 app:layout_constraintEnd_toEndOf="parent"
27 app:layout_constraintHorizontal_bias="0.201"
28 app:layout_constraintStart_toStartOf="parent"
29 app:layout_constraintTop_toTopOf="parent"
30 app:layout_constraintVertical_bias="0.449"
31 />
32
33 <ImageView
```

```
34 android:id="@+id/imageView2"
35 android:layout_width="150dp"
36 android:layout_height="150dp"
37 app:layout_constraintBottom_toBottomOf="parent"
38 app:layout_constraintEnd_toEndOf="parent"
39 app:layout_constraintHorizontal_bias="0.837"
40 app:layout_constraintStart_toStartOf="parent"
41 app:layout_constraintTop_toTopOf="parent"
42 app:layout_constraintVertical_bias="0.449"
43 />
44
45 </androidx.constraintlayout.widget.ConstraintLayout>
```

Tabel 3. Source Code Hasil Jawaban GameViewModel.kt Modul 1

```
1  package com.example.android.unscramble.ui.game
2
3  import android.text.Spannable
4  import android.text.SpannableString
5  import android.text.style.TtsSpan
6  import android.util.Log
7  import androidx.lifecycle.LiveData
8  import androidx.lifecycle.MutableLiveData
9  import androidx.lifecycle.Transformations
10 import androidx.lifecycle.ViewModel
11
12 class GameViewModel : ViewModel() {
13     private val _score = MutableLiveData(0)
14     val score: LiveData<Int>
15         get() = _score
16
17     private val _currentWordCount = MutableLiveData(0)
18     val currentWordCount: LiveData<Int>
19         get() = _currentWordCount
20
21     private val _currentScrambledWord =
   MutableLiveData<String>()
22     val currentScrambledWord: LiveData<Spannable> =
   Transformations.map(_currentScrambledWord) {
23         if (it == null) {
24             SpannableString("")
25         } else {
```

```
            val scrambledWord = it.toString()
            val       spannable:       Spannable       =
SpannableString(scrambledWord)
            spannable.setSpan(

TtsSpan.VerbatimBuilder(scrambledWord).build(),
                    0,
                    scrambledWord.length,
                    Spannable.SPAN_INCLUSIVE_INCLUSIVE
            )
            spannable
        }
    }

    private   var   wordsList:   MutableList<String>   =
mutableListOf()
    private lateinit var currentWord: String

    init {
        getNextWord()
    }

    private fun getNextWord() {
        currentWord = allWordsList.random()
        val tempWord = currentWord.toCharArray()
        tempWord.shuffle()

        while      (String(tempWord).equals(currentWord,
false)) {
            tempWord.shuffle()
        }
        if (wordsList.contains(currentWord)) {
            getNextWord()
        } else {
            Log.d("Unscramble",            "currentWord=
$currentWord")
            _currentScrambledWord.value                =
String(tempWord)
            _currentWordCount.value                    =
_currentWordCount.value?.inc()
            wordsList.add(currentWord)
```

```
60          }
61      }
62
63      fun reinitializeData() {
64          _score.value = 0
65          _currentWordCount.value = 0
66          wordsList.clear()
67          getNextWord()
68      }
69
70      private fun increaseScore() {
71          _score.value                                    =
    _score.value?.plus(SCORE_INCREASE)
72      }
73
74      fun isUserWordCorrect(playerWord: String): Boolean {
75          if (playerWord.equals(currentWord, true)) {
76              increaseScore()
77              return true
78          }
79          return false
80      }
81
82      fun nextWord(): Boolean {
83          return    if    (_currentWordCount.value!!    <
    MAX_NO_OF_WORDS) {
84              getNextWord()
85              true
86          } else false
87      }
88  }
```

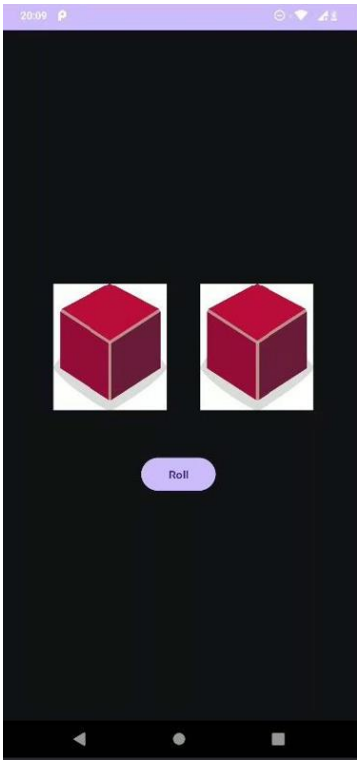
Tabel 4. Source Code Hasil Jawaban ListofWords.kt Modul 1

```
1   package com.example.android.unscramble.ui.game
2
3   const val MAX_NO_OF_WORDS = 10
4   const val SCORE_INCREASE = 20
5
6   val allWordsList: List<String> =
7       listOf("hewan",  "mobil",  "anekdot",  "alfabet",
    "semua",  "hebat",  "bangkit",  "balon",  "keranjang",
```

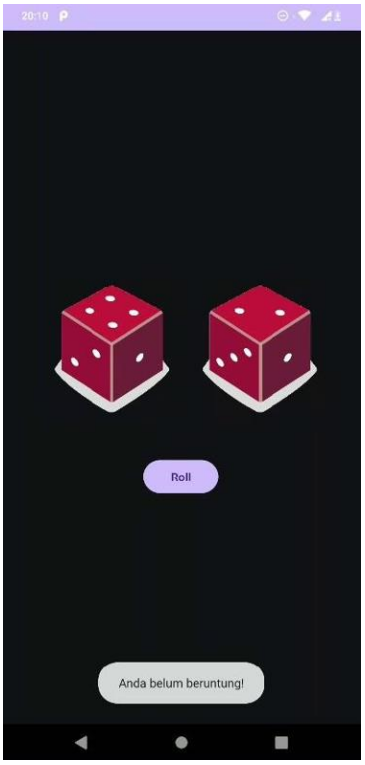
	"bangku", "terbaik", "ulang tahun", "buku", "tas kerja", "kamera", "berkemah",
8	"lilin", "kucing", "kembang kol", "obrolan", "anak-anak", "kelas", "klasik", "ruang kelas", "kopi", "berwarna-warni", "kue", "kreatif", "pelayaran", "menari", "siang hari",
9	"dinosaurus", "kenop pintu", "makan malam", "mimpi", "senja", "makan", "gajah", "zamrud", "mengerikan", "listrik", "selesai", "bunga", "mengikuti", "rubah", "bingkai",
10	"bebas", "sering", "corong", "hijau", "gitar", "belanja", "gelas", "hebat", "tertawa kecil", "potong rambut", "setengah", "buatan sendiri", "terjadi", "madu", "bergegas",
11	"seratus", "es", "iglo", "berinvestasi", "mengundang", "ikon", "memperkenalkan", "lelucon", "ceria", "jurnal", "melompat", "bergabung", "kanguru", "papan ketik", "dapur", "koala",
12	"baik hati", "kaleidoskop", "pemandangan", "terlambat", "tertawa", "belajar", "lemon", "surat", "lili", "majalah", "laut", "permen marshmallow", "labirin", "bermeditasi", "melodi",
13	"menit", "monumen", "bulan", "sepeda motor", "gunung", "musik", "utara", "hidung", "malam", "nama", "tidak pernah", "bernegosiasi", "nomor", "berlawanan", "gurita", "oak",
14	"pesanan", "buka", "kutub", "kemas", "lukisan", "orang", "piknik", "bantal", "pizza", "podcast", "presentasi", "anak anjing", "teka-teki", "resep", "melepaskan", "restoran",
15	"berputar", "mundur", "ruangan", "lari", "rahasia", "benih", "kapal", "kemeja", "seharusnya", "kecil", "pesawat luar angkasa", "melihat bintang", "keterampilan", "jalan", "gaya", "matahari terbit", "taksi",
16	"rapi", "pengatur waktu", "bersama", "gigi", "turis", "bepergian", "truk", "di bawah", "berguna", "unicorn", "unik", "meningkatkan", "seragam", "vas", "biola", "pengunjung", "visi",
17	"volume", "pemandangan", "walrus", "berkelana", "dunia", "musim dingin", "baik", "angin

```
    puyuh", "sinar-X", "xilofon", "yoga", "yogurt", "yoyo",
    "kamu", "tahun", "lezat", "zebra",
18          "zigzag", "zoologi", "zona", "semangat")
```

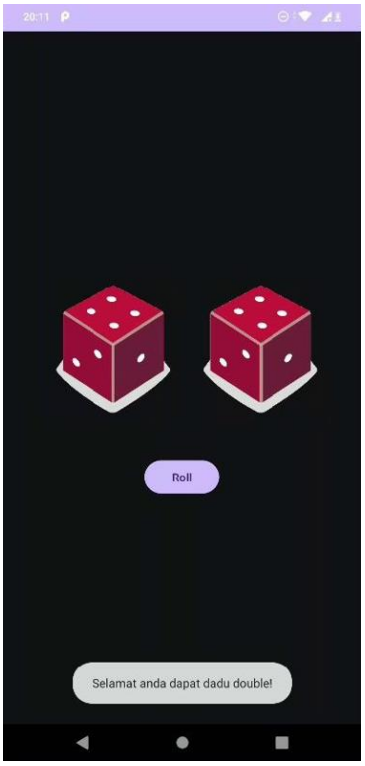
B. Output Program



Gambar 1. Praktikum 1 (Screenshot 1)



Gambar 2. Praktikum 1 (Screenshot 2)



Gambar 3. Praktikum 1 (Screenshot 3)

C. Pembahasan

- MainActivity.kt

Baris [3-7] : Digunakan untuk mengimpor beberapa kelas dari library Android

Baris [14-26] : Mempersiapkan UI dengan mengatur tata letak dari file activity_main menggunakan metode setContentView(). Selanjutnya, dua objek ImageView diinisialisasi untuk menampilkan gambar dadu default. Tombol yang diberi nama rollButton juga diinisialisasi untuk menanggapi interaksi pengguna. Ketika tombol tersebut ditekan, fungsi rollDice() akan dipanggil.

Baris [28-53] : Fungsi ini dipanggil ketika pengguna menekan tombol untuk mengocok dadu. Pertama, sebuah objek dadu dengan enam sisi diinisialisasi, kemudian fungsi roll() dari objek dadu tersebut dipanggil untuk menghasilkan angka acak antara 1 dan 6. Berdasarkan hasil lemparan dadu, gambar dadu yang sesuai ditampilkan di ImageView menggunakan ekspresi when.

Baris [55-63] : Setelah kedua dadu di-gulirkan, kondisi if digunakan untuk memeriksa apakah hasil guliran keduanya sama. Jika hasilnya sama, maka sebuah pesan toast akan ditampilkan dengan teks "Selamat anda dapat dadu double!". Jika tidak, maka pesan toast dengan teks "Anda belum beruntung!" akan ditampilkan.

Baris [65-70] : Kelas ini memiliki satu properti bernama numSides yang menentukan jumlah sisi dadu dan juga memiliki sebuah metode bernama roll(), yang mengembalikan hasil lemparan dadu, yaitu angka acak antara 1 dan jumlah sisi dadu. Fungsi random() digunakan untuk menghasilkan angka acak dalam rentang yang diberikan.

- activity_main.xml

Baris [1-8] : Antarmuka pengguna dari MainActivity akan dibangun menggunakan ConstraintLayout. Lebar dan tinggi layout diatur agar sesuai dengan parentnya. Ini berarti layout akan mengisi seluruh ruang yang tersedia di layar.

Baris [10-19] : Berfungsi untuk menampilkan tombol dilayar. Tombol ini memiliki ID unik button. Atribut layout_width dan layout_height diatur agar ukurannya menyesuaikan dengan ukuran teks di dalamnya. Atribut layout_marginTop menentukan jarak dari atas ke tombol (52dp). Atribut app:layout_constraintStart_toStartOf dan app:layout_constraintEnd_toEndOf digunakan untuk mengikat tombol ke batas start dan end dari parent-nya. app:layout_constraintTop_toBottomOf="@+id/imageView" mengatur

tombol agar terletak di bawah ImageView yang memiliki ID imageView. Atribut app:layout_constraintHorizontal_bias menentukan seberapa bias tombol diletakkan secara horizontal dalam ConstraintLayout, dengan nilai 0,498 menandakan posisi tengah horizontal. Baris [21-43] : sebuah ImageView yang berfungsi untuk menampilkan gambar kedua dadu dalam tata letak aktivitas Android. Ukuran ImageView diatur menjadi 150dp x 150dp menggunakan atribut layout_width dan layout_height. ImageView diletakkan di tengah-tengah layar menggunakan atribut app:layout_constraintStart_toStartOf, app:layout_constraintEnd_toEndOf, app:layout_constraintTop_toTopOf, dan app:layout_constraintBottom_toBottomOf yang diikat ke parent-nya (ConstraintLayout). Atribut app:layout_constraintHorizontal_bias dan app:layout_constraintVertical_bias mengatur seberapa jauh dari pinggir dan bagian atas layar ImageView diletakkan dengan nilai bias yang sesuai.

D. Tautan Git

Praktikum-Pemrograman-Mobile/Praktikum Modul 1 at main · harioct/Praktikum-Pemrograman-Mobile (github.com)

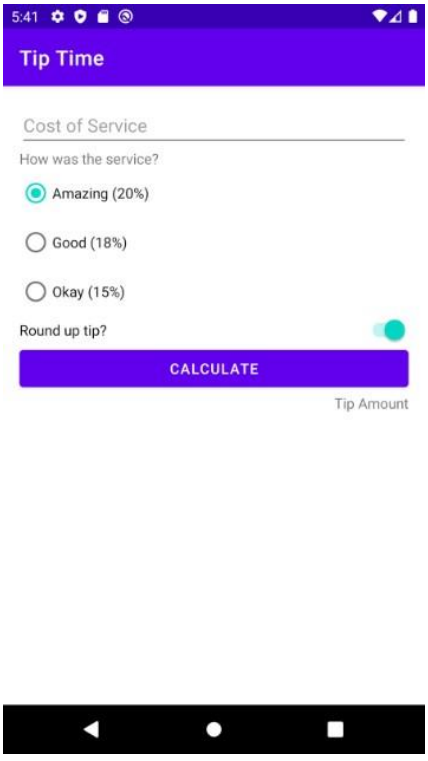
PRAKTIKUM MODUL 2

SOAL

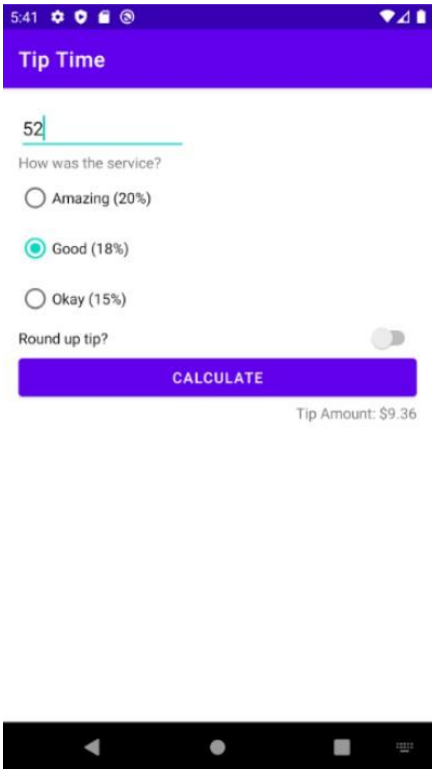
Buatlah sebuah aplikasi kalkulator tip yang dirancang untuk membantu pengguna menghitung tip yang sesuai berdasarkan total biaya layanan yang mereka terima. Fitur-fitur yang diharapkan dalam aplikasi ini mencakup:

1. Input Biaya Layanan: Pengguna dapat memasukkan total biaya layanan yang diterima dalam bentuk nominal.
2. Pilihan Persentase Tip: Pengguna dapat memilih persentase tip yang diinginkan dari opsi yang disediakan, yaitu 15%, 18%, dan 20%.
3. Pengaturan Pembulatan Tip: Pengguna dapat memilih untuk membulatkan tip ke angka yang lebih tinggi.

Tampilan Hasil: Aplikasi akan menampilkan jumlah tip yang harus dibayar secara langsung setelah pengguna memberikan input.



Gambar 1 Tampilan Awal Aplikasi



Gambar 2 Tampilan Aplikasi Setelah Dijalankan

A. Source Code

Tabel 5. Source Code Hasil Jawaban MainActivity.kt Modul 2

```
1  package com.example.tiptime
2
3  import android.os.Bundle import android.widget.*
4  import   androidx.appcompat.app.AppCompatActivity   import
5  java.text.NumberFormat
6  import kotlin.math.ceil
7
8  class MainActivity : AppCompatActivity() {
9
10 override   fun   onCreate(savedInstanceState:   Bundle?)   {
11 super.onCreate(savedInstanceState)
12 setContentView(R.layout.activity_main)
13
14 val            calcButton:            Button            =
15 findViewById(R.id.calculate_button)
16 calcButton.setOnClickListener { calcTip() }
17 }
```

```
18
19 private fun calcTip() {
20 val costEditText: EditText =
   findViewById(R.id.cost_of_service_edit_text)
21 val costString = costEditText.text.toString()
22
23
24 if (costString.isEmpty()) {
25 displayTip(0.0)
26 return
27 }
28
29 val cost = costString.toDoubleOrNull()
30
31
32 if (cost == null || cost == 0.0) {
33 displayTip(0.0)
34 return
35 }
36
37 val tipPercent = when
   (findViewById<RadioGroup>(R.id.tip_options).checkedRadioB
   uttonId)
38 {
39 R.id.amazing_option -> 0.20
40 R.id.good_option -> 0.18
41 else -> 0.15
42 }
43
44 var tip = tipPercent * cost
45 val roundUp =
   findViewById<Switch>(R.id.round_up_switch).isChecked
46 if (roundUp) {
47 tip = ceil(tip)
48 }

49 displayTip(tip)
50 }
51
52 private fun displayTip(tip: Double) {
53 val formattedTip =
```

```
    NumberFormat.getCurrencyInstance().format(tip)
54  val tipResult: TextView = findViewById(R.id.tip_result)
55  tipResult.text = getString(R.string.tip_amount,
    formattedTip)
56  }
57  }
```

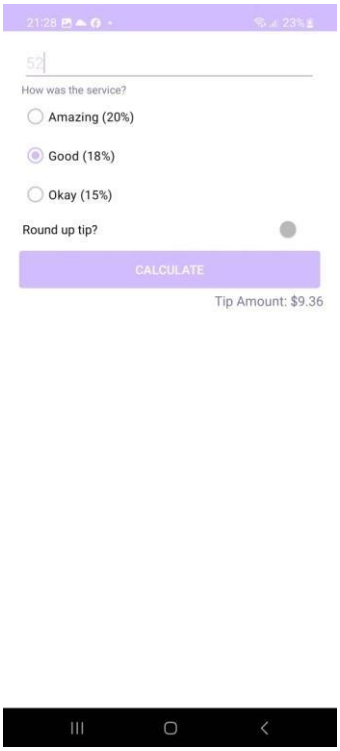
Tabel 6. Source Code Hasil Jawaban GameFragment.kt Modul 2

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <androidx.constraintlayout.widget.ConstraintLayout
3   xmlns:android="http://schemas.android.com/apk/res/android"
4   xmlns:tools="http://schemas.android.com/tools"
5   android:layout_width="match_parent"
6   android:background="@color/white"
7   android:layout_height="match_parent"
8   xmlns:app="http://schemas.android.com/apk/res-auto"
9   tools:context=".MainActivity">
10
11  <EditText
12  android:id="@+id/cost_of_service_edit_text"
13  android:layout_width="361dp"
14  android:layout_height="47dp"
15  android:layout_margin="16dp"
16  android:layout_marginTop="16dp"
17  android:textColorHint="@color/material_dynamic_tertiary50"
18  android:hint="Cost                of                Service"
19  android:inputType="numberDecimal"
20  app:layout_constraintEnd_toEndOf="parent"
21  app:layout_constraintHorizontal_bias="0.46"
22  app:layout_constraintStart_toStartOf="parent"
23  app:layout_constraintTop_toTopOf="parent" />
24
25  <TextView
26  android:id="@+id/tip"
27  android:layout_width="wrap_content"
28  android:layout_height="wrap_content"
29  android:text="How was the service?"
30  android:textColor="@color/material_dynamic_tertiary50"
31  android:textSize="12sp"
32  app:layout_constraintEnd_toEndOf="parent"
33  app:layout_constraintHorizontal_bias="0.081"
```

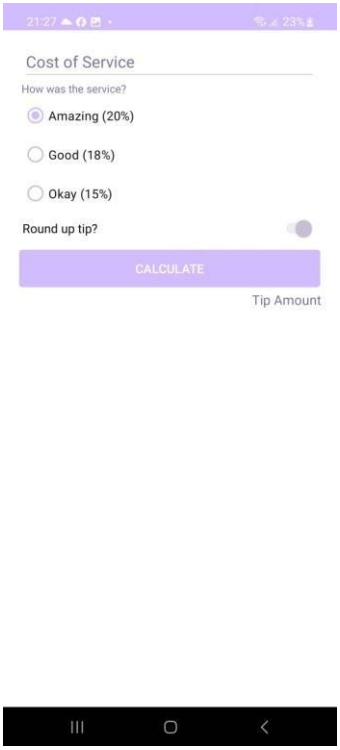
```
34  app:layout_constraintStart_toStartOf="parent"
35  app:layout_constraintTop_toBottomOf="@+id/cost_of_service_
    edit_text" />
36
37  <RadioGroup
38  android:id="@+id/tip_options"
39  android:layout_width="wrap_content"
40  android:layout_height="wrap_content"
41  android:layout_marginStart="24dp"
42  app:layout_constraintStart_toStartOf="parent"
43  app:layout_constraintTop_toBottomOf="@+id/tip">
44
45  <RadioButton
46  android:id="@+id/amazing_option"
47  android:layout_width="wrap_content"
48  android:layout_height="wrap_content"
49  android:textColor="@android:color/black"
50  android:checked="true" android:text="Amazing (20%)" />
51
52  <RadioButton
53  android:id="@+id/good_option"
54  android:layout_width="wrap_content"
55  android:layout_height="wrap_content"
56  android:textColor="@android:color/black"
57  android:text="Good (18%)" />
58
59  <RadioButton
60  android:id="@+id/okay_option"
61  android:layout_width="wrap_content"
62  android:layout_height="wrap_content"
63  android:textColor="@android:color/black"
64  android:text="Okay (15%)" />
65  </RadioGroup>
66
67  <Switch
68  android:id="@+id/round_up_switch"
69  android:layout_width="360dp"
70  android:layout_height="29dp"
71  android:layout_marginStart="24dp"
72  android:layout_marginTop="4dp"
73  android:textColor="@android:color/black"
```

```
74  android:text="Round                    up                    tip?"
75  app:layout_constraintStart_toStartOf="parent"
76  app:layout_constraintTop_toBottomOf="@id/tip_options" />
77
78  <Button
79  android:id="@+id/calculate_button"
80  android:layout_width="371dp"
81  android:layout_height="46dp"
82  android:layout_marginTop="12dp"
83  android:background="@drawable/button"
84  android:textColor="@color/white"
85  android:text="CALCULATE"
86  app:layout_constraintEnd_toEndOf="parent"
87  app:layout_constraintHorizontal_bias="0.49"
88  app:layout_constraintStart_toStartOf="parent"
89  app:layout_constraintTop_toBottomOf="@id/round_up_switch"
    />
90
91  <TextView
92  android:id="@+id/tip_result"
93  android:layout_width="wrap_content"
94  android:layout_height="wrap_content"
95  android:layout_marginTop="4dp"
96  android:text="Tip Amount"
97  android:textColor="@color/material_dynamic_tertiary50"
98  android:textSize="15sp"
99  app:layout_constraintEnd_toEndOf="parent"
100 app:layout_constraintHorizontal_bias="0.94"
101 app:layout_constraintStart_toStartOf="parent"
102 app:layout_constraintTop_toBottomOf="@id/calculate_button"
103 />
104
105 </androidx.constraintlayout.widget.ConstraintLayout>
```

B. Output Program



Gambar 4. Praktikum 2 (Screenshot 1)



Gambar 5. Praktikum 2 (Screenshot 2)

C. Pembahasan

- MainActivity.kt

Baris [3] – [7], mengimpor paket-paket yang diperlukan seperti android.os.Bundle untuk menangani data antar aktivitas, android.widget.* untuk mengakses komponen antarmuka pengguna seperti tombol dan teks, dan java.text.NumberFormat untuk format angka. Kode juga mengimpor kotlin.math.ceil yang digunakan untuk membulatkan angka ke atas. Baris [11] – [17], metode onCreate yang menginisialisasi aktivitas ketika aplikasi Android pertama kali dibuat. Di dalam metode ini, super.onCreate(savedInstanceState) memanggil implementasi dasar dari onCreate untuk menyelesaikan inisialisasi aktivitas. Metode setContentView(R.layout.activity_main) digunakan untuk menetapkan tata letak tampilan aktivitas dari berkas XML activity_main. Kode ini juga mendeklarasikan sebuah tombol dengan mencari tampilan yang memiliki ID calculate_button dari tataletak yang ditetapkan. Kemudian, setOnClickListener ditetapkan untuk calcButton, sehingga ketika tombol tersebut diklik, metode calcTip() akan dipanggil untuk melakukan operasi tertentu, seperti perhitungan tip. Baris [19] – [50], menghitung dan menampilkan tip berdasarkan biaya layanan yang diinputkan. Pertama, biaya layanan diambil dari EditText dengan ID cost_of_service_edit_text sebagai string (costString). Jika string ini kosong atau tidak valid, tip diatur ke 0.0. Persentase tip ditentukan berdasarkan opsi RadioGroup yang dipilih (20% untuk "amazing", 18% untuk "good", 15% untuk lainnya). Tip dihitung dengan mengalikan biaya dengan persentase tip, dan jika saklar Switch dengan ID round_up_switch aktif, tip dibulatkan ke atas menggunakan ceil. Terakhir, fungsi memanggil displayTip untuk menampilkan nilai tip. Baris [52] – [56], menampilkan nilai tip yang telah dihitung.Tip diberikan sebagai parameter tip bertipe Double. Nilai tip diformat menjadi mata uang menggunakan NumberFormat.getCurrencyInstance().format(tip), dan hasilnya disimpan dalam formattedTip. TextView dengan ID tip_result kemudian diambil, dan teksnya diatur menggunakan getString(R.string.tip_amount, formattedTip), di mana R.string.tip_amount adalah string resource yang memungkinkan penyisipan formattedTip ke dalam teks.

- activity_main.xml

Baris [11] – [23], digunakan untuk menerima input biaya layanan dalam aplikasi Android. Elemen ini memiliki ID cost_of_service_edit_text dan lebar serta tinggi masing-masing 361dp dan 47dp. Margin di sekitar elemen diatur sebesar 16dp, dengan margin atas juga sebesar 16dp. Warna hint teks diatur menggunakan

@color/material_dynamic_tertiary50, dan hint-nya adalah "Cost of Service". Jenis input ditetapkan sebagai numberDecimal, memungkinkan pengguna memasukkan angka desimal. EditText ini diatur dengan constraint untuk menempatkannya di tengah horizontal (dengan bias 0.46) dan di bagian atas layout, mengikatkan ujung kiri dan kanannya ke ujung layout utama. Baris [25] – [35], digunakan untuk menampilkan teks yang menanyakan "How was the service?" di dalam aplikasi Android. Elemen ini memiliki ID tip dan lebar serta tingginya diatur sebagai wrap_content, sehingga ukurannya akan menyesuaikan dengan konten teksnya. Warna teks diatur menggunakan @color/material_dynamic_tertiary50 dan ukuran teksnya adalah 12sp. Elemen ini ditempatkan di dalam layout dengan constraint, mengikat ujung kanan dan kirinya ke ujung layout utama, dengan bias horizontal sebesar 0.081. Selain itu, elemen ini diatur untuk ditempatkan di bawah EditText dengan ID cost_of_service_edit_text, memastikan bahwa elemen ini berada di bawah input biaya layanan dalam tata letak vertikal. Baris [37] – [66], mengelompokkan tiga RadioButton untuk memilih persentase tip dalam aplikasi Android. RadioGroup ini diatur dengan lebar dan tinggi wrap_content, margin kiri 24dp, dan diletakkan di bawah elemen TextView dengan ID tip. Di dalamnya, terdapat tiga RadioButton: amazing_option dengan teks "Amazing (20%)" yang diatur sebagai pilihan default dan berwarna hitam, good_option dengan teks "Good (18%)", serta okay_option dengan teks "Okay (15%)", keduanya juga berwarna hitam. Setiap RadioButton memiliki lebar dan tinggi wrap_content, menyesuaikan dengan konten teks masing- masing.

D. Tautan Git

Praktikum-Pemrograman-Mobile/Praktikum Modul 2 at main · harioct/Praktikum-Pemrograman-Mobile (github.com)

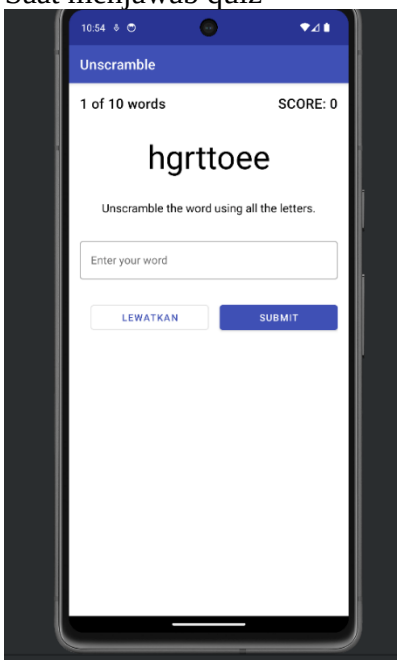
PRAKTIKUM MODUL 3

SOAL

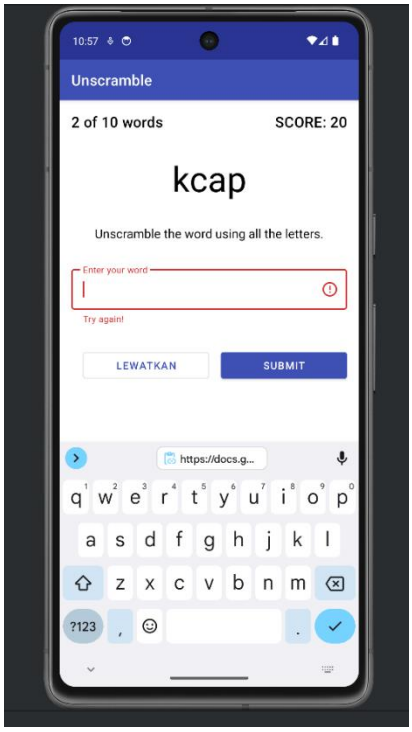
Buat sebuah aplikasi Android sederhana yang disebut "Unscramble" yang tampilannya menggunakan bahasa Indonesia. Aplikasi ini akan menampilkan kata-kata yang diacak sebanyak 10 soal, dan pengguna harus menebak kata yang benar. Soal yang tidak bisa dijawab bisa dilewati dan setiap satu soal yang benar bernilai 20.

Contoh Hasil Aplikasi :

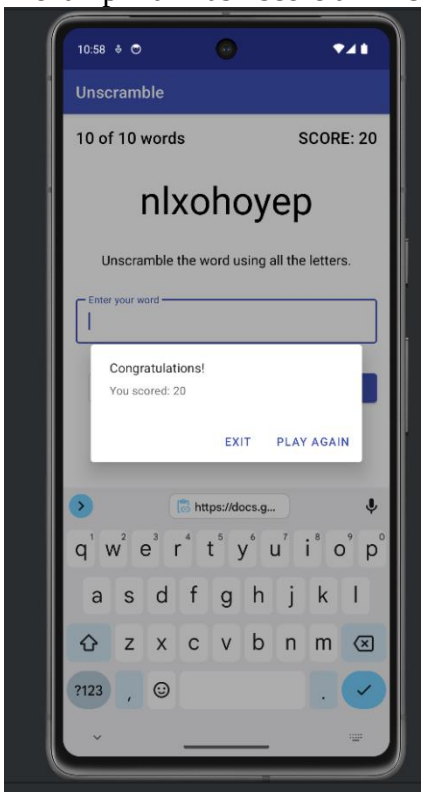
1. Saat menjawab quiz



2. Saat field input dibiarkan kosong dan disubmit



3. Menampilkan hasil score akhir setelah 10 soal terjawab



A. Source Code

Tabel 7. Source Code Hasil Jawaban MainActivity.kt Modul 3

```
1  package com.example.android.unscramble
2
3  import android.os.Bundle
4  import androidx.appcompat.app.AppCompatActivity
5
6  class MainActivity : AppCompatActivity() {
7
8      override fun onCreate(savedInstanceState: Bundle?)
   {
9          super.onCreate(savedInstanceState)
10         setContentView(R.layout.main_activity)
11     }
12 }
```

Tabel 8. Source Code Hasil Jawaban GameFragment.kt Modul 3

```
1  package com.example.android.unscramble.ui.game
2
3  import android.os.Bundle
4  import android.view.LayoutInflater
5  import android.view.View
6  import android.view.ViewGroup
7  import androidx.databinding.DataBindingUtil
8  import androidx.fragment.app.Fragment
9  import androidx.fragment.app.viewModels
10 import com.example.android.unscramble.R
11 import com.example.android.unscramble.databinding.
   GameFragmentBinding
12 import com.google.android.material.dialog.
   MaterialAlertDialogBuilder
13
14 class GameFragment : Fragment() {
15
16     private lateinit var binding: GameFragmentBinding
17
18     private val viewModel: GameViewModel by viewModels()
19
20     override fun onCreateView(
21         inflater: LayoutInflater, container: ViewGroup?,
22         savedInstanceState: Bundle?
23     ): View {
```

```
24
25          binding    =    DataBindingUtil.inflate(inflater,
    R.layout.game_fragment, container, false)
26          return binding.root
27      }
28
29      override      fun      onViewCreated(view:      View,
    savedInstanceState: Bundle?) {
30          super.onViewCreated(view, savedInstanceState)
31
32          binding.gameViewModel = viewModel
33          binding.maxNoOfWords = MAX_NO_OF_WORDS
34
35          binding.lifecycleOwner = viewLifecycleOwner
36
37          binding.submit.setOnClickListener                {
    onSubmitWord() }
38          binding.skip.setOnClickListener { onSkipWord() }
39      }
40
41      private fun onSubmitWord() {
42          val                  playerWord                  =
    binding.textInputEditText.text.toString()
43
44          if (viewModel.isUserWordCorrect(playerWord)) {
45              setErrorTextField(false)
46              if (!viewModel.nextWord()) {
47                  showFinalScoreDialog()
48              }
49          } else {
50              setErrorTextField(true)
51          }
52      }
53
54      private fun onSkipWord() {
55          if (viewModel.nextWord()) {
56              setErrorTextField(false)
57          } else {
58              showFinalScoreDialog()
59          }
60      }
```

```
61
62      private fun showFinalScoreDialog() {
63          MaterialAlertDialogBuilder(requireContext())
64
65  .setTitle(getString(R.string.congratulations))
66
67  .setMessage(getString(R.string.you_scored,
    viewModel.score.value))
68                  .setCancelable(false)
69
70  .setNegativeButton(getString(R.string.exit)) { _, _ ->
71                      exitGame()
72                  }
73
74  .setPositiveButton(getString(R.string.play_again)) { _, _
    ->
75                      restartGame()
76                  }
77                  .show()
78      }
79
80      private fun restartGame() {
81          viewModel.reinitializeData()
82          setErrorTextField(false)
83      }
84
85      private fun exitGame() {
86          activity?.finish()
87      }
88
89      private fun setErrorTextField(error: Boolean) {
90          if (error) {
91              binding.textField.isErrorEnabled = true
92              binding.textField.error                    =
    getString(R.string.try_again)
93          } else {
94              binding.textField.isErrorEnabled = false
                binding.textInputEditText.text = null
            }
        }
    }
```

Tabel 9. Source Code Hasil Jawaban GameViewModel.kt Modul 3

```
1   package com.example.android.unscramble.ui.game
2
3   import android.text.Spannable
4   import android.text.SpannableString
5   import android.text.style.TtsSpan
6   import android.util.Log
7   import androidx.lifecycle.LiveData
8   import androidx.lifecycle.MutableLiveData
9   import androidx.lifecycle.Transformations
10  import androidx.lifecycle.ViewModel
11
12  class GameViewModel : ViewModel() {
13      private val _score = MutableLiveData(0)
14      val score: LiveData<Int>
15          get() = _score
16
17      private val _currentWordCount = MutableLiveData(0)
18      val currentWordCount: LiveData<Int>
19          get() = _currentWordCount
20
21      private       val       _currentScrambledWord       =
    MutableLiveData<String>()
22      val   currentScrambledWord:   LiveData<Spannable>   =
    Transformations.map(_currentScrambledWord) {
23          if (it == null) {
24              SpannableString("")
25          } else {
26              val scrambledWord = it.toString()
27              val       spannable:       Spannable       =
    SpannableString(scrambledWord)
28              spannable.setSpan(
29
    TtsSpan.VerbatimBuilder(scrambledWord).build(),
30                      0,
31                      scrambledWord.length,
32                      Spannable.SPAN_INCLUSIVE_INCLUSIVE
33              )
34              spannable
35          }
```

```
36      }
37
38      private var wordsList: MutableList<String> =
    mutableListOf()
39      private lateinit var currentWord: String
40
41      init {
42          getNextWord()
43      }
44
45      private fun getNextWord() {
46          currentWord = allWordsList.random()
47          val tempWord = currentWord.toCharArray()
48          tempWord.shuffle()
49
50          while (String(tempWord).equals(currentWord,
    false)) {
51              tempWord.shuffle()
52          }
53          if (wordsList.contains(currentWord)) {
54              getNextWord()
55          } else {
56              Log.d("Unscramble", "currentWord=
    $currentWord")
57              _currentScrambledWord.value =
    String(tempWord)
58              _currentWordCount.value =
    _currentWordCount.value?.inc()
59              wordsList.add(currentWord)
60          }
61      }
62
63      fun reinitializeData() {
64          _score.value = 0
65          _currentWordCount.value = 0
66          wordsList.clear()
67          getNextWord()
68      }
69
70      private fun increaseScore() {
71
```

```
        _score.value                                          =
72  _score.value?.plus(SCORE_INCREASE)
73      }
74
75      fun isUserWordCorrect(playerWord: String): Boolean {
76          if (playerWord.equals(currentWord, true)) {
77              increaseScore()
78              return true
79          }
80          return false
81      }
82
83      fun nextWord(): Boolean {
            return    if    (_currentWordCount.value!!    <
84  MAX_NO_OF_WORDS) {
85              getNextWord()
86              true
87          } else false
88      }
    }
```

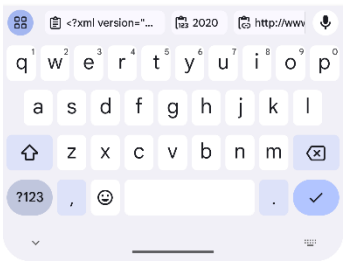
Tabel 10. Source Code Hasil Jawaban ListofWords.kt Modul 3

```
1   package com.example.android.unscramble.ui.game
2
3   const val MAX_NO_OF_WORDS = 10
4   const val SCORE_INCREASE = 20
5
6   val allWordsList: List<String> =
7       listOf("hewan",  "mobil",  "anekdot",  "alfabet",
    "semua",  "hebat",  "bangkit",  "balon",  "keranjang",
    "bangku",  "terbaik",  "ulang  tahun",  "buku",  "tas
    kerja", "kamera", "berkemah",
8           "lilin", "kucing", "kembang kol", "obrolan",
    "anak-anak", "kelas", "klasik", "ruang kelas", "kopi",
    "berwarna-warni",   "kue",   "kreatif",   "pelayaran",
    "menari", "siang hari",
9           "dinosaurus", "kenop pintu", "makan malam",
    "mimpi",   "senja",   "makan",   "gajah",   "zamrud",
    "mengerikan",    "listrik",    "selesai",    "bunga",
    "mengikuti", "rubah", "bingkai",
10
```

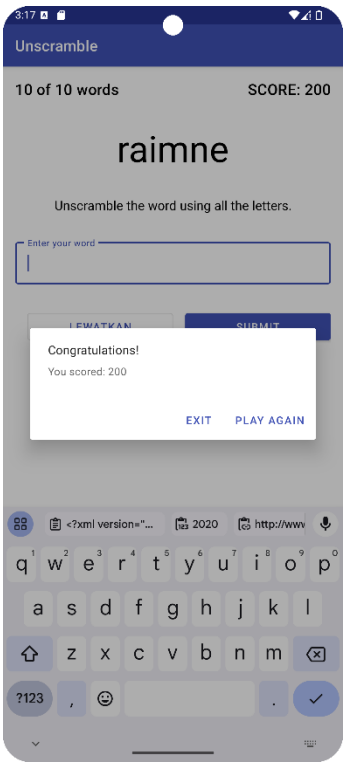
```
        "bebas", "sering", "corong", "hijau", "gitar",
    "belanja", "gelas", "hebat", "tertawa kecil", "potong
    rambut", "setengah", "buatan sendiri", "terjadi",
11  "madu", "bergegas",
        "seratus", "es", "iglo", "berinvestasi",
    "mengundang", "ikon", "memperkenalkan", "lelucon",
    "ceria", "jurnal", "melompat", "bergabung", "kanguru",
12  "papan ketik", "dapur", "koala",
        "baik hati", "kaleidoskop", "pemandangan",
    "terlambat", "tertawa", "belajar", "lemon", "surat",
    "lili", "majalah", "laut", "permen marshmallow",
13  "labirin", "bermeditasi", "melodi",
        "menit", "monumen", "bulan", "sepeda motor",
    "gunung", "musik", "utara", "hidung", "malam", "nama",
    "tidak pernah", "bernegosiasi", "nomor", "berlawanan",
14  "gurita", "oak",
        "pesanan", "buka", "kutub", "kemas",
    "lukisan", "orang", "piknik", "bantal", "pizza",
    "podcast", "presentasi", "anak anjing", "teka-teki",
15  "resep", "melepaskan", "restoran",
        "berputar", "mundur", "ruangan", "lari",
    "rahasia", "benih", "kapal", "kemeja", "seharusnya",
    "kecil", "pesawat luar angkasa", "melihat bintang",
    "keterampilan", "jalan", "gaya", "matahari terbit",
16  "taksi",
        "rapi", "pengatur waktu", "bersama", "gigi",
    "turis", "bepergian", "truk", "di bawah", "berguna",
    "unicorn", "unik", "meningkatkan", "seragam", "vas",
17  "biola", "pengunjung", "visi",
        "volume", "pemandangan", "walrus",
    "berkelana", "dunia", "musim dingin", "baik", "angin
    puyuh", "sinar-X", "xilofon", "yoga", "yogurt", "yoyo",
18  "kamu", "tahun", "lezat", "zebra",
        "zigzag", "zoologi", "zona", "semangat")
```

B. Output Program

Gambar 6. Praktikum 3 (Screenshot 1)



Gambar 7. Praktikum 3 (Screenshot 2)



Gambar 8. Praktikum 3 (Screenshot 3)

C. Pembahasan

- MainActivity.kt

MainActivity merupakan subclass dari AppCompatActivity, yang adalah kelas dasar untuk aktivitas dengan dukungan penuh terhadap fitur-fitur modern pada perangkat Android. Dalam metode onCreate, yang dipanggil saat aktivitas pertama kali dibuat, aktivitas ini menginisialisasi dirinya sendiri dan menetapkan tampilan konten menggunakan file layout main_activity dari resource R.layout. super.onCreate(savedInstanceState) memastikan bahwa aktivitas mengurus inisialisasi bawaan yang ditangani oleh AppCompatActivity.

- GameFragment.kt

Sebuah fragmen dalam aplikasi Android yang menampilkan antarmuka permainan. GameFragment mewarisi kelas Fragment dan menggunakan Data Binding untuk menghubungkan data dengan UI. Di dalam onCreateView, fragmen menginisialisasi binding menggunakan DataBindingUtil.inflate untuk mengembalikan root view yang ditetapkan dalam R.layout.game_fragment. Di dalam onViewCreated, view model (GameViewModel) dihubungkan dengan binding dan beberapa properti seperti maxNoOfWords diatur. Metode onSubmitWord memeriksa apakah kata yang dimasukkan pengguna benar, dan jika benar,

lanjut ke kata berikutnya atau tampilkan dialog skor akhir jika permainan selesai. Metode onSkipWord memungkinkan pengguna melewati kata dan juga dapat memicu dialog skor akhir. showFinalScoreDialog menampilkan dialog akhir permainan dengan opsi untuk keluar atau bermain lagi. restartGame mengatur ulang data permainan dan menghapus pesan kesalahan, sementara exitGame mengakhiri aktivitas. setErrorTextField mengatur status pesan kesalahan pada input pengguna.

- GameViewModel.kt

GameViewModel adalah kelas yang bertindak sebagai ViewModel untuk permainan "Unscramble" dalam aplikasi Android. Kelas ini menggunakan LiveData untuk menyimpan dan memantau perubahan data permainan seperti skor, jumlah kata saat ini, dan kata yang diacak. MutableLiveData digunakan untuk memodifikasi nilai-nilai ini, sementara LiveData dipakai untuk memberikan akses baca saja dari data tersebut kepada UI. Transformations.map mengubah kata yang diacak menjadi Spannable, yang berguna untuk penanganan teks khusus seperti Text-to-Speech.

Dalam inisialisasinya, getNextWord dipanggil untuk mengambil kata acak dari allWordsList, mengacak hurufnya, dan memastikan kata acak tersebut tidak sama dengan kata aslinya. Kata yang baru kemudian disimpan dan daftar kata yang telah digunakan diperbarui.

Metode reinitializeData mengatur ulang skor, jumlah kata, dan daftar kata yang telah digunakan, kemudian memanggil getNextWord untuk memulai permainan baru. increaseScore meningkatkan skor ketika pengguna menebak kata dengan benar. isUserWordCorrect memeriksa apakah kata yang dimasukkan pengguna benar, dan jika benar, skor ditingkatkan. nextWord menentukan apakah permainan harus dilanjutkan dengan kata berikutnya atau tidak, berdasarkan batas jumlah kata yang telah ditetapkan.

- ListofWords.kt

Mendefinisikan dua konstanta dan sebuah daftar kata untuk permainan "Unscramble". Konstanta MAX_NO_OF_WORDS menetapkan jumlah maksimum kata yang akan digunakan dalam satu permainan sebagai 10, sedangkan SCORE_INCREASE menentukan peningkatan skor sebanyak 20 poin untuk setiap kata yang berhasil ditebak dengan benar. Daftar allWordsList berisi berbagai kata dalam bahasa Indonesia yang akan diacak dan digunakan dalam permainan, seperti "hewan", "mobil", "kamera", dan "musik". Konstanta

dan daftar kata ini digunakan oleh GameViewModel untuk mengelola logika permainan, termasuk penilaian dan penyediaan kata-kata acak kepada pengguna.

D. Tautan Git

Praktikum-Pemrograman-Mobile/Praktikum Modul 3 at main · harioct/Praktikum-Pemrograman-Mobile (github.com)

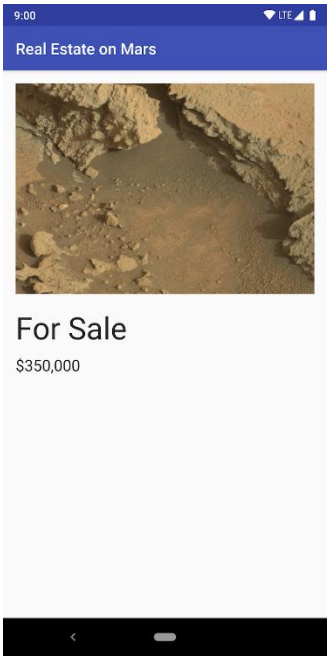
PRAKTIKUM MODUL 4

SOAL

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

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

Contoh aplikasi:



A. Source Code

Tabel 11. Source Code Hasil Jawaban MainActivity.kt Modul 4

```
1  package com.example.cartoons
2
3  import android.content.Intent
4  import android.os.Bundle
5  import android.util.Log
6  import androidx.appcompat.app.AppCompatActivity
7  import androidx.appcompat.widget.Toolbar
8  import androidx.lifecycle.lifecycleScope
9  import androidx.recyclerview.widget.LinearLayoutManager
10 import androidx.recyclerview.widget.RecyclerView
11 import kotlinx.coroutines.launch
12
13 class   MainActivity : AppCompatActivity() {
14
15     private lateinit var recyclerView: RecyclerView
16     private lateinit var cartoonAdapter: CartoonAdapter
17     private lateinit var cartoons: List<Cartoon>
18
19     override fun onCreate(savedInstanceState: Bundle?) {
20         super.onCreate(savedInstanceState)
21         setContentView(R.layout.activity_main)
22
```

```
23          val toolbar: Toolbar = findViewById(R.id.toolbar)
24          setSupportActionBar(toolbar)
25
26          recyclerView = findViewById(R.id.recyclerView)
27          recyclerView.layoutManager                          =
    LinearLayoutManager(this)
28
29          lifecycleScope.launch {
30              try {
31                  cartoons                                    =
    RetrofitInstance.api.getCartoons()
32                  cartoons.forEach { cartoon ->
33                      Log.d("MainActivity",       "Cartoon:
    $cartoon")
34                  }
35                  cartoonAdapter = CartoonAdapter(cartoons)
    { cartoon ->
36                      val             intent              =
    Intent(this@MainActivity, DetailActivity::class.java)
37                      intent.putExtra("cartoon", cartoon)
38                      startActivity(intent)
39                  }
40                  recyclerView.adapter = cartoonAdapter
41              } catch (e: Exception) {
42                  e.printStackTrace()
43                  Log.e("MainActivity",   "Error   fetching
    cartoons", e)
44              }
45          }
46      }
47  }
```

Tabel 12. Source Code Hasil Jawaban DetailActivity.kt Modul 4

```
1  package com.example.cartoons
2
3  import android.graphics.PorterDuff
4  import android.os.Bundle
5  import android.widget.ImageView
6  import android.widget.TextView
7  import androidx.appcompat.app.AppCompatActivity
8  import androidx.appcompat.widget.Toolbar
9  import com.bumptech.glide.Glide
10
11 class DetailActivity : AppCompatActivity() {
12
13     private lateinit var cartoonImage: ImageView
14     private lateinit var cartoonTitle: TextView
15     private lateinit var cartoonCreator: TextView
16     private lateinit var cartoonYear: TextView
17     private lateinit var cartoonGenre: TextView
18     private lateinit var cartoonEpisodes: TextView
19     private lateinit var toolbar: Toolbar
20
21     override fun onCreate(savedInstanceState: Bundle?)
   {
22         super.onCreate(savedInstanceState)
23         setContentView(R.layout.activity_detail)
24
25         toolbar = findViewById(R.id.toolbar)
26         setSupportActionBar(toolbar)
27         supportActionBar?.setDisplayHomeAsUpEnabled
   (true)
28         supportActionBar?.title = ""
29
30         toolbar.navigationIcon?.setColorFilter(resources.
   getColor(android.R.color.white),PorterDuff.Mode.SRC_ATOP)
31
32         cartoonImage = findViewById(R.id.cartoonImage)
33         cartoonTitle = findViewById(R.id.cartoonTitle)
34         cartoonCreator                                    =
   findViewById(R.id.cartoonCreator)
35         cartoonYear = findViewById(R.id.cartoonYear)
36         cartoonGenre = findViewById(R.id.cartoonGenre)
```

```
        cartoonEpisodes = findViewById(R.id.cartoonEpisodes)

        val cartoon = intent.getSerializableExtra("cartoon") as Cartoon
        cartoonTitle.text = cartoon.title
        cartoonCreator.text = cartoon.creator.joinToString(", ")
        cartoonYear.text = cartoon.year.toString()
        cartoonGenre.text = cartoon.genre.joinToString(", ") ?: "No genre available"
        cartoonEpisodes.text = "Total Episodes: ${cartoon.episodes}"

        toolbar.title = cartoon.title

        Glide.with(this)
            .load(cartoon.image)
            .placeholder(R.drawable.placeholder_image)
            .error(R.drawable.error_image)
            .into(cartoonImage)
    }

    override fun onSupportNavigateUp(): Boolean {
        onBackPressed()
        return true
    }
}
```

Tabel 13. Source Code Hasil Jawaban Cartoon.kt Modul 4

```
package com.example.cartoons

import com.squareup.moshi.Json
import java.io.Serializable

data class Cartoon(
    val id: Int,
    @Json(name = "title") val title: String,
    @Json(name = "year") val year: Int,
    @Json(name = "creator") val creator: List<String>,
    @Json(name = "image") val image: String?,
```

```
    @Json(name = "genre") val genre: List<String>,
    @Json(name = "episodes") val episodes: Int
) : Serializable
```

Tabel 14. Source Code Hasil Jawaban CartoonAdapter.kt Modul 4

```
package com.example.cartoons

import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.ImageView
import android.widget.TextView
import androidx.recyclerview.widget.RecyclerView
import com.bumptech.glide.Glide
import com.bumptech.glide.request.RequestOptions

class CartoonAdapter(
    private val cartoons: List<Cartoon>,
    private val onItemClick: (Cartoon) -> Unit
) : RecyclerView.Adapter<CartoonAdapter.
CartoonViewHolder>() {

    override fun onCreateViewHolder(parent:
ViewGroup, viewType: Int): CartoonViewHolder {
        val view = LayoutInflater.from
(parent.context).inflate
(R.layout.item_cartoon, parent, false)
        return CartoonViewHolder(view)
    }

    override fun onBindViewHolder(holder:
CartoonViewHolder, position: Int) {
        holder.bind(cartoons[position],
onItemClick)
    }

    override fun getItemCount(): Int =
cartoons.size

    class CartoonViewHolder(itemView:
View) : RecyclerView.ViewHolder(itemView) {
```

```
29          private val cartoonImage:
    ImageView = itemView.findViewById(R.id.cartoonImage)
30          private val cartoonTitle:
    TextView = itemView.findViewById(R.id.cartoonTitle)
31          private val cartoonCreator:
    TextView = itemView.findViewById(R.id.cartoonCreator)
32
33          fun bind(cartoon: Cartoon, onItemClick:
    (Cartoon) -> Unit) {
34              cartoonTitle.text = cartoon.title
35              cartoonCreator.text = cartoon.creator.
    joinToString(", ")
36
37              val requestOptions = RequestOptions()
38                  .placeholder(R.drawable.
    placeholder_image)
39                  .error(R.drawable.error_image)
40
41              Glide.with(itemView.context)
42                  .load(cartoon.image)
43                  .apply(requestOptions)
44                  .into(cartoonImage)
45
46              itemView.setOnClickListener {
    onItemClick(cartoon) }
47          }
48      }
49  }
```

Tabel 15. Source Code Hasil Jawaban CartoonApi.kt Modul 4

```
1   package com.example.cartoons
2
3   import retrofit2.http.GET
4
5   interface CartoonApi {
6       @GET("cartoons2D")
7       suspend fun getCartoons(): List<Cartoon>
8   }
```

Tabel 16. Source Code Hasil Jawaban CartoonViewModel.kt Modul 4

```
1   package com.example.cartoons
```

```
2
3  import androidx.lifecycle.LiveData
4  import androidx.lifecycle.MutableLiveData
5  import androidx.lifecycle.ViewModel
6  import androidx.lifecycle.viewModelScope
7  import kotlinx.coroutines.Dispatchers
8  import kotlinx.coroutines.launch
9
10 class CartoonViewModel : ViewModel() {
11
12     private           val           _cartoons           =
   MutableLiveData<List<Cartoon>>()
13     val   cartoons:   LiveData<List<Cartoon>>   get()   =
   _cartoons
14
15     init {
16         fetchCartoons()
17     }
18
19     private fun fetchCartoons() {
20         viewModelScope.launch(Dispatchers.IO) {
21             try {
22                 val             cartoonList             =
   RetrofitInstance.api.getCartoons()
23                 _cartoons.postValue(cartoonList)
24             } catch (e: Exception) {
25                 e.printStackTrace()
26             }
27         }
28     }
29
30     fun filterCartoons(query: String?) {
31         if (query.isNullOrEmpty()) {
32             fetchCartoons()
33         } else {
34             val filteredList = _cartoons.value?.filter {
35                 it.title?.contains(query,   ignoreCase   =
   true) == true ||
36                         it.genre.any    {    genre    ->
   genre.contains(query, ignoreCase = true) }
37             }
```

```
38              _cartoons.postValue(filteredList)
39          }
40      }
41  }
```

Tabel 17. Source Code Hasil Jawaban RetrofitInstance.kt Modul 4

```
1   package com.example.cartoons
2
3   import retrofit2.Retrofit
4   import retrofit2.converter.moshi.MoshiConverterFactory
5   import com.squareup.moshi.Moshi
6   import
    com.squareup.moshi.kotlin.reflect.KotlinJsonAdapterFactory
7
8   object RetrofitInstance {
9
10      private val moshi = Moshi.Builder()
11          .add(KotlinJsonAdapterFactory())
12          .build()
13
14      private val retrofit by lazy {
15          Retrofit.Builder()
16
    .baseUrl("https://api.sampleapis.com/cartoons/")
17
    .addConverterFactory(MoshiConverterFactory.create(moshi))
18              .build()
19      }
20
21      val api: CartoonApi by lazy {
22          retrofit.create(CartoonApi::class.java)
23      }
24  }
```

Tabel 18. Source Code Hasil Jawaban activity_main.xml Modul 4

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
3       xmlns:app="http://schemas.android.com/apk/res-auto"
4       android:layout_width="match_parent"
5       android:layout_height="match_parent"
```

```
6       android:orientation="vertical">
7
8       <androidx.appcompat.widget.Toolbar
9           android:id="@+id/toolbar"
10          android:layout_width="match_parent"
11          android:layout_height="?attr/actionBarSize"
12          android:background="?attr/colorPrimary"
13          android:elevation="4dp"
14          app:title="List Cartoons"
15          app:titleTextColor="@android:color/white" />
16
17      <androidx.recyclerview.widget.RecyclerView
18          android:id="@+id/recyclerView"
19          android:layout_width="match_parent"
20          android:layout_height="match_parent"
21          android:padding="16dp" />
22  </LinearLayout>
```

Tabel 19. Source Code Hasil Jawaban activity_detail.xml Modul 4

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
3       xmlns:app="http://schemas.android.com/apk/res-auto"
4       android:layout_width="match_parent"
5       android:layout_height="match_parent"
6       android:orientation="vertical">
7
8       <androidx.appcompat.widget.Toolbar
9           android:id="@+id/toolbar"
10          android:layout_width="match_parent"
11          android:layout_height="?attr/actionBarSize"
12          android:background="?attr/colorPrimary"
13          android:elevation="4dp"
14          app:titleTextColor="@android:color/white" />
15
16      <ScrollView
17          android:layout_width="match_parent"
18          android:layout_height="match_parent"
19          android:padding="16dp">
20
21          <LinearLayout
```

```
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical">

            <ImageView
                android:id="@+id/cartoonImage"
                android:layout_width="match_parent"
                android:layout_height="600dp"
                android:scaleType="centerCrop" />

            <TextView
                android:id="@+id/cartoonTitle"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:textStyle="bold"
                android:textSize="24sp"
                android:paddingTop="8dp" />

            <TextView
                android:id="@+id/cartoonCreator"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:paddingTop="4dp" />

            <TextView
                android:id="@+id/cartoonYear"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:paddingTop="4dp" />

            <TextView
                android:id="@+id/cartoonGenre"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:paddingTop="8dp"
                android:scrollbars="vertical" />

            <TextView
                android:id="@+id/cartoonEpisodes"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
```

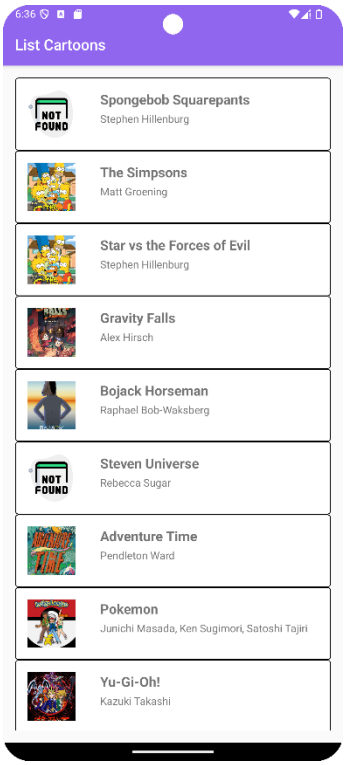
```
                android:paddingTop="4dp" />
        </LinearLayout>
    </ScrollView>
</LinearLayout>
```

Tabel 20. Source Code Hasil Jawaban item_cartoon.xml Modul 4

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:padding="16dp"
    android:background="@drawable/border">

    <ImageView
        android:id="@+id/cartoonImage"
        android:layout_width="64dp"
        android:layout_height="64dp"
        android:layout_marginEnd="16dp"
        android:scaleType="centerCrop" />

    <LinearLayout
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:orientation="vertical"
        android:paddingStart="16dp">

        <TextView
            android:id="@+id/cartoonTitle"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textStyle="bold"
            android:textSize="18sp" />

        <TextView
            android:id="@+id/cartoonCreator"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:paddingTop="4dp" />
```

```
35      </LinearLayout>
36  </LinearLayout>
```

B. Output Program



Gambar 9. Praktikum 4 (Screenshot 1)

Gambar 10. Praktikum 4 (Screenshot 2)

C. Pembahasan

- MainActivity.kt

Kelas MainActivity yang menampilkan daftar kartun menggunakan RecyclerView dan Retrofit untuk mengambil data dari API. Dalam metode onCreate, layout utama diatur menggunakan setContentView, dan Toolbar diinisialisasi dan diatur sebagai ActionBar. RecyclerView diinisialisasi dan diset untuk menggunakan LinearLayoutManager. Dengan menggunakan lifecycleScope.launch, aplikasi memulai coroutine untuk mengambil data kartun dari API. Jika data berhasil diambil, log masing-masing kartun dicatat dan CartoonAdapter diinisialisasi dengan daftar kartun tersebut, dan RecyclerView diatur untuk menggunakan adapter tersebut. Jika terjadi kesalahan saat mengambil data, kesalahan akan dicatat di log.

- DetailActivity.kt

DetailActivity menampilkan detail dari sebuah kartun yang dipilih dari daftar sebelumnya. Dalam metode onCreate, layout utama diatur menggunakan setContentView, dan Toolbar diinisialisasi dan diatur sebagai ActionBar dengan menampilkan tombol back dan mengatur warnanya. Berbagai elemen UI seperti ImageView dan TextView diinisialisasi untuk

menampilkan gambar, judul, pembuat, tahun, genre, dan jumlah episode dari kartun. Data kartun diambil dari Intent menggunakan getSerializableExtra, dan setiap elemen UI diatur dengan data yang sesuai. Gambar kartun dimuat menggunakan library Glide, dengan placeholder dan error image untuk mengatasi masalah pemuatan gambar. Metode onSupportNavigateUp digunakan untuk menangani navigasi kembali ke aktivitas sebelumnya ketika tombol back di Toolbar ditekan.

- Cartoon.kt

Cartoon digunakan untuk merepresentasikan objek kartun. Data class ini menggunakan Moshi untuk deserialisasi JSON, yang ditunjukkan oleh anotasi @Json pada beberapa properti. Cartoon memiliki beberapa properti: id (integer), title (string), year (integer), creator (list of strings), image (nullable string), genre (list of strings), dan episodes (integer). Implementasi interface Serializable memungkinkan objek Cartoon untuk dilewatkan antar aktivitas melalui Intent. Properti image diizinkan null untuk menangani kasus di mana gambar tidak tersedia.

- CartoonAdapter.kt

CartoonAdapter berfungsi sebagai adapter untuk RecyclerView dalam aplikasi Android, yang digunakan untuk menampilkan daftar kartun. Adapter ini menerima daftar objek Cartoon dan lambda onItemClick yang dipanggil ketika sebuah item dalam daftar diklik. Metode onCreateViewHolder mengembalikan CartoonViewHolder dengan menginflate layout item kartun (item_cartoon). Metode onBindViewHolder mengikat data kartun ke ViewHolder di posisi tertentu. getItemCount mengembalikan jumlah kartun dalam daftar.

Kelas CartoonViewHolder adalah ViewHolder untuk adapter ini dan berisi referensi ke elemen UI dalam layout item kartun seperti ImageView dan TextView. Metode bind mengatur teks untuk judul dan pembuat kartun, dan menggunakan library Glide untuk memuat gambar kartun dengan placeholder dan error image. Lambda onItemClick dipanggil ketika item diklik, memungkinkan interaksi pengguna untuk memicu tindakan lebih lanjut, seperti membuka detail kartun.

- CartoonApi.kt

CartoonApi menggunakan Retrofit, sebuah untuk membuat permintaan HTTP di aplikasi Android. Antarmuka ini mendeklarasikan sebuah fungsi getCartoons yang ditandai dengan anotasi @GET untuk menunjukkan bahwa ini adalah permintaan GET ke endpoint

"cartoons2D". Fungsi ini ditandai dengan suspend, yang berarti ia dapat dijalankan dalam coroutine dan memungkinkan operasi jaringan dilakukan secara asinkron. Fungsi ini mengembalikan daftar objek Cartoon, yang akan di-deserialize oleh Retrofit dari respons JSON yang diterima dari server.

- CartoonViewModel.kt

CartoonViewModel berfungsi untuk menyimpan dan mengelola data yang berhubungan dengan antarmuka pengguna dalam siklus hidup aktivitas atau fragmen. CartoonViewModel menggunakan LiveData untuk mengamati perubahan data kartun. MutableLiveData _cartoons adalah variabel privat yang diubah dalam ViewModel ini, sementara LiveData cartoons adalah versi publik yang dapat diamati dari luar.

Pada inisialisasi, fungsi fetchCartoons dipanggil untuk mengambil daftar kartun dari API menggunakan coroutine dalam viewModelScope dengan dispatcher IO untuk operasi jaringan. Jika permintaan berhasil, hasilnya diposting ke _cartoons. Jika terjadi kesalahan, exception ditangani dengan mencetak stack trace.

Fungsi filterCartoons digunakan untuk menyaring daftar kartun berdasarkan kueri pencarian. Jika kueri kosong atau null, fetchCartoons dipanggil ulang untuk memuat ulang daftar lengkap. Jika kueri tidak kosong, daftar kartun yang ada difilter berdasarkan judul atau genre yang mengandung kueri, dan hasilnya diposting kembali ke _cartoons untuk diperbarui dalam tampilan.

- RetrofitInstance.kt

RetrofitInstance, sebuah singleton yang menyediakan instance Retrofit untuk aplikasi Android. Objek ini mengonfigurasi Moshi dengan KotlinJsonAdapterFactory untuk mendukung parsing JSON ke objek Kotlin. Instance Retrofit dibuat dengan Retrofit.Builder, menetapkan baseUrl ke "https://api.sampleapis.com/cartoons/" dan menambahkan konverter Moshi. Singleton ini juga menyediakan properti api yang menginisialisasi CartoonApi menggunakan metode create dari Retrofit, memungkinkan aplikasi untuk melakukan panggilan jaringan ke API kartun dengan mudah dan efisien tanpa perlu mengkonfigurasi Retrofit berulang kali.

- activity_main.xml

Layout XML untuk menampilkan daftar kartun menggunakan RecyclerView. Layout ini menggunakan LinearLayout dengan orientasi vertikal dan berisi dua komponen utama: Toolbar dan RecyclerView. Toolbar berada di bagian atas dengan lebar penuh, tinggi sesuai ukuran standar ActionBar, latar belakang warna utama aplikasi, elevasi untuk efek bayangan, dan judul "List Cartoons" dengan teks berwarna putih. Di bawahnya, RecyclerView yang juga memiliki lebar dan tinggi penuh, dengan padding 16dp, digunakan untuk menampilkan daftar kartun yang diambil dari API. Layout ini memastikan tampilan yang terstruktur dan mudah diatur untuk daftar kartun.

- activity_detail.xml

Layout XML untuk menampilkan informasi rinci tentang sebuah kartun. Layout ini menggunakan LinearLayout sebagai root dengan orientasi vertikal, berisi Toolbar dan ScrollView untuk mengatur elemen-elemen UI secara terstruktur. Toolbar di bagian atas memiliki lebar penuh dan tinggi sesuai ukuran standar ActionBar, dengan latar belakang warna utama aplikasi, dan elevasi untuk memberikan efek bayangan. Judul Toolbar tidak ditentukan di sini karena kemungkinan akan diatur secara dinamis di dalam kode.

Di bawah Toolbar, terdapat ScrollView yang memungkinkan konten di dalamnya untuk digulir, memastikan semua informasi dapat dilihat meskipun melebihi tinggi layar. Di dalam ScrollView, terdapat LinearLayout vertikal yang berisi komponen-komponen UI untuk menampilkan detail kartun. ImageView dengan ID cartoonImage digunakan untuk menampilkan gambar kartun, dengan lebar penuh dan tinggi tetap 600dp serta scaleType centerCrop untuk memastikan gambar memenuhi seluruh area yang tersedia. Beberapa TextView digunakan untuk menampilkan berbagai informasi kartun: cartoonTitle untuk judul dengan gaya teks tebal dan ukuran 24sp, cartoonCreator untuk pembuat kartun, cartoonYear untuk tahun rilis, cartoonGenre untuk genre dengan scrollbar vertikal jika teksnya panjang, dan cartoonEpisodes untuk jumlah episode. Setiap TextView memiliki padding untuk memastikan tampilan yang rapi dan mudah dibaca. Layout ini dirancang untuk memberikan tampilan yang komprehensif dan informatif bagi pengguna yang ingin melihat detail dari kartun yang dipilih.

- item_cartoon.xml

Layout XML untuk menampilkan informasi ringkas tentang sebuah kartun. Layout ini menggunakan LinearLayout dengan orientasi horizontal sebagai root, dan mengatur elemen-elemen UI secara terstruktur untuk menampilkan gambar dan informasi kartun.

Di dalam LinearLayout utama, terdapat ImageView dengan ID cartoonImage yang digunakan untuk menampilkan gambar kartun. ImageView ini memiliki lebar dan tinggi tetap 64dp, margin akhir 16dp, dan scaleType centerCrop untuk memastikan gambar memenuhi area yang tersedia dengan proporsi yang tepat. Selanjutnya, terdapat LinearLayout vertikal di sebelah kanan ImageView, yang memiliki lebar 0dp dengan layout_weight 1 untuk mengambil sisa ruang yang tersedia di baris horizontal tersebut. LinearLayout vertikal ini memiliki padding awal 16dp dan berisi dua TextView.

TextView pertama dengan ID cartoonTitle digunakan untuk menampilkan judul kartun, dengan gaya teks tebal dan ukuran teks 18sp. TextView kedua dengan ID cartoonCreator digunakan untuk menampilkan nama pembuat kartun, dengan padding atas 4dp untuk memberikan sedikit ruang antara teks judul dan teks pembuat. Layout ini juga memiliki padding keseluruhan 16dp dan latar belakang yang menggunakan drawable border untuk memberikan efek batas pada setiap item dalam RecyclerView, membuatnya tampak lebih terorganisir dan menarik.

D. Tautan Git

Praktikum-Pemrograman-Mobile/Praktikum Modul 4 at main · harioct/Praktikum-Pemrograman-Mobile (github.com)

PRAKTIKUM MODUL 5

SOAL

Buat sebuah aplikasi e-commerce menggunakan Material Design Components (MDC). Aplikasi ini akan menampilkan daftar produk, rincian produk, dan keranjang belanja. User interface untuk halaman utama yang menampilkan daftar produk menggunakan RecyclerView sesuai dengan prinsip Material Design yang akan dibuat. Navigasi antar layar akan ditambahkan menggunakan Navigation Component untuk mengatur navigasi antara halaman utama dan halaman rincian produk. Model data untuk produk, adapter untuk RecyclerView, serta layout untuk item produk juga akan dibuat. Terakhir, ViewModel untuk menyimpan data keranjang belanja akan dibuat dan fungsionalitas untuk menambahkan produk ke keranjang dari halaman rincian produk akan diimplementasikan.

A. Source Code

Tabel 21. Source Code Hasil Jawaban MainActivity.kt Modul 5

```
1  package com.google.codelabs.mdc.kotlin.shrine
2
3  import android.os.Bundle
4  import androidx.appcompat.app.AppCompatActivity
5  import androidx.fragment.app.Fragment
6
7  class MainActivity : AppCompatActivity(), NavigationHost
   {
8
9      override fun onCreate(savedInstanceState: Bundle?) {
10         super.onCreate(savedInstanceState)
11         setContentView(R.layout.shr_main_activity)
12
13         if (savedInstanceState == null) {
14             supportFragmentManager
15                     .beginTransaction()
16                     .add(R.id.container, LoginFragment())
17                     .commit()
18         }
19     }
20
21     /**
22      * Navigate to the given fragment.
```

```
23      *
24      * @param fragment       Fragment to navigate to.
25      * @param addToBackstack Whether or not the current
    fragment should be added to the backstack.
26      */
27     override    fun    navigateTo(fragment:    Fragment,
    addToBackstack: Boolean) {
28         val transaction = supportFragmentManager
29                 .beginTransaction()
30                 .replace(R.id.container, fragment)
31
32         if (addToBackstack) {
33             transaction.addToBackStack(null)
34         }
35
36         transaction.commit()
37     }
38 }
```

Tabel 22. Source Code Hasil Jawaban LoginFragment.kt Modul 5

```
1   package com.google.codelabs.mdc.kotlin.shrine
2
3   import android.os.Bundle
4   import android.text.Editable
5   import android.view.LayoutInflater
6   import android.view.View
7   import android.view.ViewGroup
8   import androidx.fragment.app.Fragment
9   import
    kotlinx.android.synthetic.main.shr_login_fragment.
    password_edit_text
10  import
    kotlinx.android.synthetic.main.shr_login_fragment.
    password_text_input
11  import kotlinx.android.synthetic.main.shr_login_
    fragment.view.next_button
12  import
    kotlinx.android.synthetic.main.shr_login_fragment.
    view.password_edit_text
13
14  /**
```

```
15  * Fragment representing the login screen for Shrine.
16  */
17  class LoginFragment : Fragment() {
18
19      override fun onCreateView(
20              inflater:     LayoutInflater,     container:
    ViewGroup?, savedInstanceState: Bundle?): View? {
21          // Inflate the layout for this fragment.
22          val                     view                    =
    inflater.inflate(R.layout.shr_login_fragment, container,
    false)
23
24          // Set an error if the password is less than 8
    characters.
25          view.next_button.setOnClickListener({
26              if
    (!isPasswordValid(password_edit_text.text!!)) {
27                  password_text_input.error               =
    getString(R.string.shr_error_password)
28              } else {
29                  // Clear the error.
30                  password_text_input.error = null
31                  // Navigate to the next Fragment.
32                  (activity                               as
    NavigationHost).navigateTo(ProductGridFragment(), false)
33              }
34          })
35
36          // Clear the error once more than 8 characters
    are typed.
37          view.password_edit_text.setOnKeyListener({ _, _,
    _ ->
38              if
    (isPasswordValid(password_edit_text.text!!)) {
39                  // Clear the error.
40                  password_text_input.error = null
41              }
42              false
43          })
44
45          return view
```

```
46      }
47
48
49      // "isPasswordValid" from "Navigate to the next
    Fragment" section method goes here
50      private fun isPasswordValid(text: Editable?): Boolean
    {
51          return text != null && text.length >= 8
52      }
53  }
```

Tabel 23. Source Code Hasil Jawaban NavigationIconClickListener.kt Modul 5

```
1   package com.google.codelabs.mdc.kotlin.shrine
2
3   import android.animation.AnimatorSet
4   import android.animation.ObjectAnimator
5   import android.app.Activity
6   import android.content.Context
7   import android.graphics.drawable.Drawable
8   import android.util.DisplayMetrics
9   import android.view.View
10  import android.view.animation.Interpolator
11  import android.widget.ImageView
12
13  /**
14   * [android.view.View.OnClickListener] used to
    translate the product grid sheet downward on
15   * the Y-axis when the navigation icon in the
    toolbar is pressed.
16   */
17  class NavigationIconClickListener @JvmOverloads
    internal constructor(
18          private val context: Context, private val
    sheet: View, private val interpolator: Interpolator? =
    null,
19          private val openIcon: Drawable? = null,
    private val closeIcon: Drawable? = null) :
    View.OnClickListener {
20
21      private val animatorSet = AnimatorSet()
22      private val height: Int
```

```
23      private var backdropShown = false
24
25      init {
26          val displayMetrics = DisplayMetrics()
27          (context                                          as
    Activity).windowManager.defaultDisplay.
    getMetrics(displayMetrics)
28          height = displayMetrics.heightPixels
29      }
30
31      override fun onClick(view: View) {
32          backdropShown = !backdropShown
33
34          // Cancel the existing animations
35          animatorSet.removeAllListeners()
36          animatorSet.end()
37          animatorSet.cancel()
38
39          updateIcon(view)
40
41          val translateY =
     height                                                -
    context.resources.getDimensionPixelSize(R.dimen.
    shr_product_grid_reveal_height)
42
43          val  animator  =  ObjectAnimator.ofFloat(sheet,
    "translationY",  (if  (backdropShown)  translateY  else
    0).toFloat())
44          animator.duration = 500
45          if (interpolator != null) {
46              animator.interpolator = interpolator
47          }
48          animatorSet.play(animator)
49          animator.start()
50      }
51
52      private fun updateIcon(view: View) {
53          if (openIcon != null && closeIcon != null) {
54              if (view !is ImageView) {
```

```
55                  throw
    IllegalArgumentException("updateIcon() must be called on
    an ImageView")
56              }
57              if (backdropShown) {
58                  view.setImageDrawable(closeIcon)
59              } else {
60                  view.setImageDrawable(openIcon)
61              }
62          }
63      }
64  }
```

Tabel 24. Source Code Hasil Jawaban NavigationHost.kt Modul 5

```
1   package com.google.codelabs.mdc.kotlin.shrine
2
3   import androidx.fragment.app.Fragment
4
5
6   /**
7    * A host (typically an `Activity`} that can display
    fragments and knows how to respond to
8    * navigation events.
9    */
10  interface NavigationHost {
11      /**
12       * Trigger a navigation to the specified fragment,
    optionally adding a transaction to the back
13       * stack to make this navigation reversible.
14       */
15      fun navigateTo(fragment: Fragment, addToBackstack:
    Boolean)
16  }
```

Tabel 25. Source Code Hasil Jawaban ProductCardRecyclerViewAdapter.kt Modul 5

```
1   package com.google.codelabs.mdc.kotlin.shrine
2
3   import android.view.LayoutInflater
4   import android.view.ViewGroup
5   import androidx.recyclerview.widget.RecyclerView
6   import com.google.codelabs.mdc.kotlin.shrine.network.
```

```
   ImageRequester
7  import com.google.codelabs.mdc.kotlin.shrine.network.
   ProductEntry
8
9  /**
10  * Adapter used to show a simple grid of products.
11  */
12 class ProductCardRecyclerViewAdapter(private val
   productList:            List<ProductEntry>)            :
   RecyclerView.Adapter<ProductCardViewHolder>() {
13
14     override fun onCreateViewHolder(parent: ViewGroup,
   viewType: Int): ProductCardViewHolder {
15         val                  layoutView                 =
   LayoutInflater.from(parent.context).inflate(R.layout.
   shr_product_card, parent, false)
16         return ProductCardViewHolder(layoutView)
17     }
18
19     override         fun         onBindViewHolder(holder:
   ProductCardViewHolder, position: Int) {
20         if (position < productList.size) {
21             val product = productList[position]
22             holder.productTitle.text = product.title
23             holder.productPrice.text = product.price
24             ImageRequester.setImageFromUrl(holder.
   productImage, product.url)
25         }
26     }
27
28     override fun getItemCount(): Int {
29         return productList.size
30     }
31 }
```

Tabel 26. Source Code Hasil Jawaban ProductCardViewHolder.kt Modul 5

```
1  package com.google.codelabs.mdc.kotlin.shrine
2
3  import android.view.View
4  import android.widget.TextView
5  import androidx.recyclerview.widget.RecyclerView
```

```
6  import com.android.volley.toolbox.NetworkImageView
7
8  class ProductCardViewHolder(itemView: View) :
   RecyclerView.ViewHolder(itemView) {
9
10     var productImage: NetworkImageView =
   itemView.findViewById(R.id.product_image)
11     var productTitle: TextView =
   itemView.findViewById(R.id.product_title)
12     var productPrice: TextView =
   itemView.findViewById(R.id.product_price)
13 }
```

Tabel 27. Source Code Hasil Jawaban ProductGridFragment.kt Modul 5

```
1  package com.google.codelabs.mdc.kotlin.shrine
2
3  import android.os.Build
4  import android.os.Bundle
5  import android.view.LayoutInflater
6  import android.view.Menu
7  import android.view.MenuInflater
8  import android.view.View
9  import android.view.ViewGroup
10 import android.view.animation.
   AccelerateDecelerateInterpolator
11 import androidx.appcompat.app.AppCompatActivity
12 import androidx.core.content.ContextCompat
13 import androidx.fragment.app.Fragment
14 import androidx.recyclerview.widget.
   GridLayoutManager
15 import androidx.recyclerview.widget.RecyclerView
16 import com.google.codelabs.mdc.kotlin.
   shrine.network.ProductEntry
17 import com.google.codelabs.mdc.kotlin.shrine.
   staggeredgridlayout.
   StaggeredProductCardRecyclerViewAdapter
18 import kotlinx.android.synthetic.main.shr_
   product_grid_fragment.view.*
19
20 class ProductGridFragment : Fragment() {
21
```

```
22      override fun onCreate(savedInstanceState: Bundle?) {
23          super.onCreate(savedInstanceState)
24          setHasOptionsMenu(true)
25      }
26
27      override fun onCreateView(
28              inflater: LayoutInflater, container:
    ViewGroup?, savedInstanceState: Bundle?): View? {
29          // Inflate the layout for this fragment with
    the ProductGrid theme
30          val view = inflater.inflate(R.layout.shr_product_
    grid_fragment, container, false)
31
32          // Set up the toolbar.
33          (activity as AppCompatActivity).
    setSupportActionBar(view.app_bar)
34          view.app_bar.setNavigationOnClickListener(
    NavigationIconClickListener(
35              activity!!,
36              view.product_grid,
37              AccelerateDecelerateInterpolator(),
38              ContextCompat.getDrawable(context!!,
    R.drawable.shr_branded_menu), // Menu open icon
39              ContextCompat.getDrawable(context!!,
    R.drawable.shr_close_menu))) // Menu close icon
40
41          // Set up the RecyclerView
42          view.recycler_view.setHasFixedSize(true)
43          val gridLayoutManager =
    GridLayoutManager(context, 2, RecyclerView.
    HORIZONTAL, false)
44          gridLayoutManager.spanSizeLookup =
    object : GridLayoutManager.SpanSizeLookup() {
45              override fun getSpanSize(position:
    Int): Int {
46                  return if (position % 3 == 2)
    2 else 1
47              }
48          }
49          view.recycler_view.layoutManager
    = gridLayoutManager
```

```
50          val adapter =
    StaggeredProductCardRecyclerViewAdapter(
51              ProductEntry.initProductEntryList
    (resources))
52          view.recycler_view.adapter = adapter
53          val largePadding = resources.
    getDimensionPixelSize(R.dimen.shr_staggered_product_
    grid_spacing_large)
54          val                 smallPadding                =
    resources.getDimensionPixelSize(R.dimen.shr_staggered_
    product_grid_spacing_small)
55          view.recycler_view.addItemDecoration
    (ProductGridItemDecoration(largePadding,
    smallPadding))
56
57          // Set cut corner background for API 23+
58          if (Build.VERSION.SDK_INT >= Build.
    VERSION_CODES.M) {
59              view.product_grid.background               =
    context?.getDrawable(R.drawable.shr_product_
    grid_background_shape)
60          }
61
62          return view;
63      }
64
65      override    fun    onCreateOptionsMenu(menu:    Menu,
    menuInflater: MenuInflater) {
66          menuInflater.inflate(R.menu.shr_toolbar_menu,
    menu)
67          super.onCreateOptionsMenu(menu, menuInflater)
68      }
69  }
```

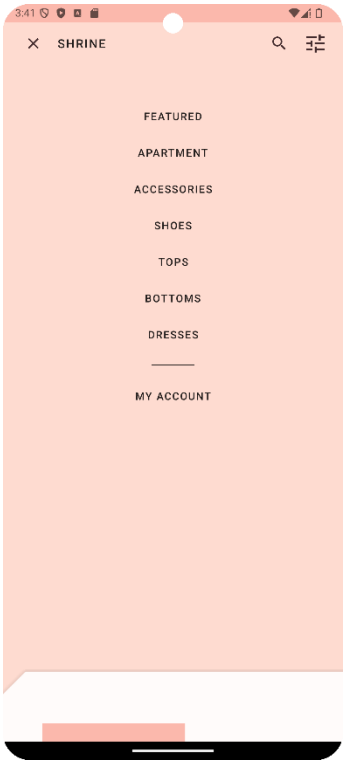
Tabel 28. Source Code Hasil Jawaban activity_main.xml Modul 5

```
1   package com.google.codelabs.mdc.kotlin.shrine
2
    import android.graphics.Rect
3   import android.view.View
4   import androidx.recyclerview.widget.RecyclerView
5
```

```
6  /**
7   *    Custom    item    decoration    for    a    vertical
   [ProductGridFragment] [RecyclerView]. Adds a
8   * small amount of padding to the left of grid items, and
   a large amount of padding to the right.
9   */
10 class ProductGridItemDecoration(private val largePadding:
11 Int,     private     val     smallPadding:     Int)     :
   RecyclerView.ItemDecoration() {
12
13     override fun getItemOffsets(outRect: Rect, view: View,
14                                 parent:     RecyclerView,
   state: RecyclerView.State) {
15         outRect.left = smallPadding
16         outRect.right = smallPadding
17     }
18 }
```

B. Output Program

Gambar 11. Praktikum 5 (Screenshot 1)



Gambar 12. Praktikum 5 (Screenshot 2)

C. Pembahasan

- MainActivity.kt

Mengimplementasikan NavigationHost untuk mengelola navigasi antar fragmen. Dalam metode onCreate, layout shr_main_activity diatur sebagai tampilan konten, dan jika savedInstanceState adalah null, LoginFragment ditambahkan ke R.id.container menggunakan supportFragmentManager. Metode navigateTo diimplementasikan untuk memungkinkan navigasi ke fragmen yang ditentukan, dengan pilihan untuk menambahkan transaksi ke backstack. Jika addToBackstack bernilai true, fragmen saat ini ditambahkan ke backstack sebelum mengganti fragmen di R.id.container dengan fragmen baru, dan kemudian transaksi tersebut dikomit.

- LoginFragment.kt

Merupakan fragmen yang mewakili layar login untuk aplikasi Shrine. Dalam metode onCreateView, layout shr_login_fragment di-inflate sebagai tampilan fragmen. Ketika tombol next_button diklik, sistem memeriksa apakah password yang dimasukkan valid (minimal 8 karakter) menggunakan metode isPasswordValid. Jika password tidak valid, sebuah error ditampilkan pada password_text_input; jika valid, error dihapus dan aplikasi

menavigasi ke ProductGridFragment menggunakan metode navigateTo dari NavigationHost. Selain itu, jika pengguna mengetik lebih dari 8 karakter dalam password_edit_text, error akan dihapus secara otomatis.

- NavigationIconClickListener.kt

Mengimplementasikan View.OnClickListener untuk mengelola interaksi saat ikon navigasi di toolbar ditekan, yang akan menggerakkan "product grid sheet" ke bawah pada sumbu Y. Kelas ini mengambil konteks, tampilan lembaran (sheet), interpolator opsional, dan ikon untuk tampilan terbuka dan tertutup sebagai parameter. Pada inisialisasi, tinggi layar diambil untuk menghitung posisi terjemahan Y.

Ketika metode onClick dipanggil, status backdropShown dibalik, animasi yang ada dibatalkan, dan ikon diperbarui. Objek ObjectAnimator digunakan untuk menggerakkan lembaran (sheet) ke posisi yang sesuai, berdasarkan apakah lembaran tersebut ditampilkan atau disembunyikan. Animasi berlangsung selama 500 milidetik, dan interpolator diterapkan jika disediakan. Metode updateIcon digunakan untuk mengganti ikon pada ImageView berdasarkan status backdropShown.

Secara keseluruhan, kelas ini mengelola animasi transisi untuk lembar produk dan ikon navigasi, memberikan pengalaman pengguna yang dinamis dan responsif.

- NavigationHost.kt

NavigationHost diimplementasikan oleh sebuah Activity untuk menampilkan fragmen dan menangani event navigasi. Antarmuka ini memiliki satu metode, navigateTo, yang menerima dua parameter: fragment dari tipe Fragment yang menentukan fragmen tujuan navigasi, dan addToBackstack dari tipe Boolean yang menentukan apakah transaksi navigasi tersebut harus ditambahkan ke backstack agar dapat dibalik. Dengan kata lain, antarmuka ini digunakan untuk mengabstraksi logika navigasi antar fragmen di dalam aplikasi, memungkinkan Activity yang mengimplementasikannya untuk menavigasi ke fragmen tertentu dan mengelola backstack secara opsional.

- ProductCardRecyclerViewAdapter.kt

Sebuah adapter untuk RecyclerView yang menampilkan grid produk. Adapter ini menerima daftar produk (productList) dan menggunakan ProductCardViewHolder untuk mengelola tampilan setiap item dalam daftar. Metode onCreateViewHolder digunakan untuk meng-

inflate layout shr_product_card dan mengembalikan instance ProductCardViewHolder, sementara metode onBindViewHolder mengikat data dari ProductEntry ke view holder berdasarkan posisi, mengatur judul, harga, dan gambar produk menggunakan ImageRequester. Metode getItemCount mengembalikan jumlah total item dalam daftar produk, memungkinkan RecyclerView mengetahui berapa banyak item yang harus ditampilkan. Dengan demikian, ProductCardRecyclerViewAdapter mengatur data produk dan menghubungkannya dengan tampilan yang sesuai dalam grid.

- ProductCardViewHolder.kt

Sebuah kelas yang memperluas RecyclerView.ViewHolder dan digunakan untuk mengelola tampilan item dalam RecyclerView yang menampilkan produk-produk. Kelas ini memiliki tiga properti: productImage, productTitle, dan productPrice. Properti productImage adalah NetworkImageView yang digunakan untuk menampilkan gambar produk yang diambil dari jaringan, sedangkan productTitle dan productPrice adalah TextView yang digunakan untuk menampilkan judul dan harga produk. Pada inisialisasi, findViewById digunakan untuk menghubungkan tampilan dari itemView dengan properti kelas ini, memastikan bahwa setiap item dalam RecyclerView memiliki referensi ke elemen UI yang sesuai untuk menampilkan data produk.

- ProductGridFragment.kt

Sebuah fragmen yang menampilkan grid produk menggunakan RecyclerView. Pada metode onCreate, fragmen disiapkan untuk memiliki menu opsional. Pada metode onCreateView, layout shr_product_grid_fragment di-inflate dan toolbar diatur menggunakan setSupportActionBar. NavigationIconClickListener dihubungkan dengan app_bar untuk mengatur ikon navigasi. RecyclerView diatur dengan GridLayoutManager untuk menampilkan item dalam dua kolom secara horizontal, dengan penyesuaian untuk ukuran span berdasarkan posisi item. Adapter StaggeredProductCardRecyclerViewAdapter dihubungkan ke RecyclerView, yang diisi dengan daftar produk yang diinisialisasi dari sumber daya. Padding besar dan kecil ditambahkan sebagai dekorasi item dalam RecyclerView. Untuk perangkat dengan API 23 atau lebih tinggi, latar belakang sudut dipotong diterapkan pada grid produk. Metode onCreateOptionsMenu meng-inflate menu toolbar dari shr_toolbar_menu. Secara keseluruhan, fragmen ini mengelola tampilan grid produk yang kaya dengan penyesuaian tata letak dan interaksi pengguna.

- ProductGridItemDecoration.kt

merupakan custom item decoration untuk RecyclerView dalam ProductGridFragment. Kelas ini memperluas RecyclerView.ItemDecoration dan digunakan untuk menambahkan padding khusus ke item-item dalam grid.

Konstruktor kelas ini menerima dua parameter: largePadding dan smallPadding, yang merupakan jumlah padding besar dan kecil yang akan diterapkan pada item-item dalam grid.

Metode getItemOffsets di-override untuk mengatur padding pada item-item dalam RecyclerView. Parameter outRect adalah objek Rect yang digunakan untuk mengatur offset padding untuk item. Padding kecil (smallPadding) diterapkan ke kiri dan kanan setiap item di grid.

Dengan demikian, ProductGridItemDecoration menyediakan cara untuk menambahkan ruang (padding) di sekitar item-item dalam grid produk, membuat tata letak lebih rapi dan teratur.

D. Tautan Git

Praktikum-Pemrograman-Mobile/Praktikum Modul 5 at main · harioct/Praktikum-Pemrograman-Mobile (github.com)