LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 2

ANDROID LAYOUT

Oleh:

Sheila Sabina **NIM. 2310817220028**

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
APRIL 2025

LEMBAR PENGESAHAN
LAPORAN PRAKTIKUM PEMROGRAMAN I
MODUL 2

Laporan Praktikum Pemrograman Mobile Modul 2: Android Layout ini disusun sebagai syarat lulus mata kuliah Praktikum Pemrograman Mobile. Laporan Prakitkum ini dikerjakan oleh:

Nama Praktikan : Sheila Sabina
NIM : 2310817220028

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

LEMBAR PENGESAHAN 2
DAFTAR ISI 3
DAFTAR GAMBAR 4
DAFTAR TABEL 5
SOAL 1 6
A. Source Code 7
B. Output Program 12
C. Pembahasan 13
D. Tautan Git 20

DAFTAR GAMBAR

Gambar 1. Screenshot Hasil Jawaban Soal 1 13

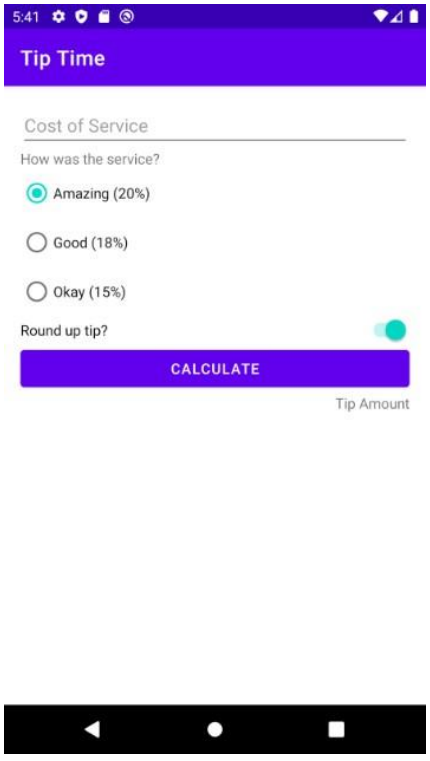
DAFTAR TABEL

Tabel 1. Source Code Jawaban Soal 1 8
Tabel 2. Source Code Jawaban Soal 1 11
Tabel 3. Source Code Jawaban Soal 1 12

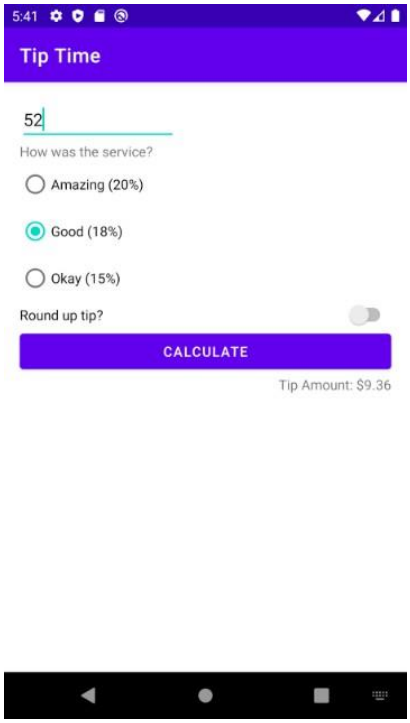
SOAL 1

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 Presentase Tip: Pengguna dapat memilih presentase 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.
4. 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

1. MainActivity.kt

```
1   package com.example.androidlayout
2
3   import android.os.Bundle
4   import android.widget.*
5   import androidx.activity.viewModels
6   import androidx.appcompat.app.AppCompatActivity
7   import androidx.lifecycle.Observer
8
9   class MainActivity : AppCompatActivity() {
10
11      private val tipViewModel: TipViewModel by viewModels()
12
13      override fun onCreate(savedInstanceState: Bundle?) {
14          super.onCreate(savedInstanceState)
15          setContentView(R.layout.activity_main)
16
17          val etBiaya = findViewById<EditText>(R.id.etBiaya)
18          val rgTip = findViewById<RadioGroup>(R.id.rgTip)
19          val switchBulatkan =
20  findViewById<Switch>(R.id.switchBulatkan)
21          val btnHitung = findViewById<Button>(R.id.btnHitung)
22          val tvHasil = findViewById<TextView>(R.id.tvHasil)
23
```

```
24          tipViewModel.biayaInput.observe(this) {
25              if (etBiaya.text.toString() != it)
26  etBiaya.setText(it)
27          }
28
29          tipViewModel.bulatkan.observe(this) {
30              switchBulatkan.isChecked = it
31          }
32
33          tipViewModel.tipResult.observe(this) {
34              tvHasil.text = it
35          }
36
37          btnHitung.setOnClickListener {
38              val biayaInput = etBiaya.text.toString()
39
40              if (biayaInput.isEmpty()) {
41                  Toast.makeText(this, "Masukkan biaya layanan
42  terlebih dahulu", Toast.LENGTH_SHORT).show()
43                  return@setOnClickListener
44              }
45
46              val biaya = biayaInput.toDoubleOrNull()
47              if (biaya == null || biaya <= 0) {
48                  Toast.makeText(this, "Biaya layanan harus lebih
49  dari 0", Toast.LENGTH_SHORT).show()
50                  return@setOnClickListener
51              }
52
53              val persenTip = when (rgTip.checkedRadioButtonId) {
54                  R.id.rb15 -> 0.15
55                  R.id.rb18 -> 0.18
56                  R.id.rb20 -> 0.20
57                  else -> {
58                      Toast.makeText(this, "Pilih persentase
59  tip", Toast.LENGTH_SHORT).show()
60                      return@setOnClickListener
61                  }
62              }
63
64              tipViewModel.setBiaya(biayaInput)
65              tipViewModel.setBulatkan(switchBulatkan.isChecked)
66              tipViewModel.setPersenTip(persenTip)
67              tipViewModel.hitungTip()
68          }
69      }
70  }
```

Tabel 1. Source Code Jawaban Soal 1

2. activity_main.xml

```
<?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="match_parent"
    android:background="@android:color/white"
    android:orientation="vertical">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="56dp"
        android:background="#6200EE"
        android:gravity="center_vertical"
        android:paddingStart="16dp"
        android:text="Tip Time"
        android:textColor="@android:color/white"
        android:textSize="20sp" />

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:padding="16dp">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical">

            <EditText
                android:id="@+id/etBiaya"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:backgroundTint="#6200EE"
                android:hint="Cost of Service"
                android:inputType="numberDecimal" />

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginTop="16dp"
                android:text="How was the service?"
                android:textColor="#888888" />

            <RadioGroup
                android:id="@+id/rgTip"
                android:layout_width="match_parent"
                android:layout_height="wrap_content">

                <RadioButton
                    android:id="@+id/rb20"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
```

```
                    android:text="Amazing (20%)" />

                <RadioButton
                    android:id="@+id/rb18"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:text="Good (18%)" />

                <RadioButton
                    android:id="@+id/rb15"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:text="Okay (15%)" />
            </RadioGroup>

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="16dp"
                android:orientation="horizontal">

                <TextView
                    android:layout_width="0dp"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"
                    android:text="Round up tip?" />

                <Switch
                    android:id="@+id/switchBulatkan"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content" />
            </LinearLayout>


            <Button
                android:id="@+id/btnHitung"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="24dp"
                android:backgroundTint="#6200EE"
                android:text="CALCULATE"
                android:textColor="@android:color/white" />

            <TextView
                android:id="@+id/tvHasil"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="24dp"
                android:gravity="end"
                android:text="Tip Amount"
                android:textColor="#888888"
                android:textSize="16sp" />
```

```
        </LinearLayout>
    </ScrollView>
</LinearLayout>
```

Tabel 2. Source Code Jawaban Soal 1

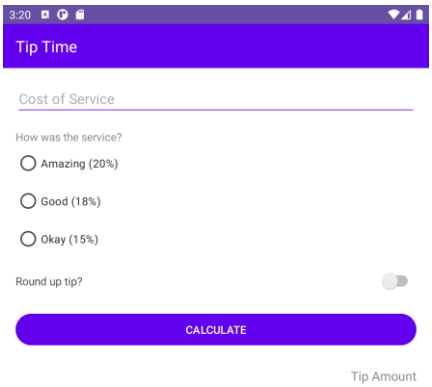
3. TipViewModel.kt

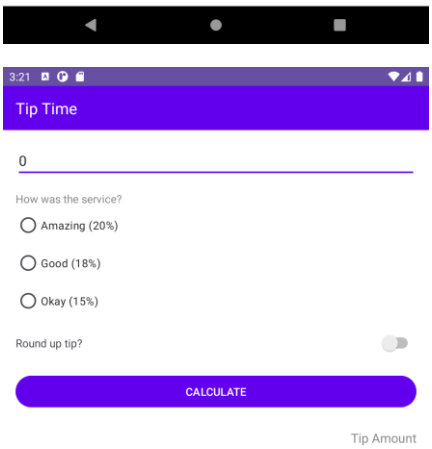
```
package com.example.androidlayout

import androidx.lifecycle.LiveData
import androidx.lifecycle.MutableLiveData
import androidx.lifecycle.ViewModel
import kotlin.math.ceil

class TipViewModel : ViewModel() {

    private val _biayaInput = MutableLiveData<String>()
    val biayaInput: LiveData<String> = _biayaInput

    private val _tipResult = MutableLiveData<String>()
    val tipResult: LiveData<String> = _tipResult

    private val _bulatkan = MutableLiveData<Boolean>()
    val bulatkan: LiveData<Boolean> = _bulatkan

    private var persenTip: Double = 0.0

    fun setBiaya(biaya: String) {
        _biayaInput.value = biaya
    }

    fun setBulatkan(value: Boolean) {
        _bulatkan.value = value
    }

    fun setPersenTip(value: Double) {
        persenTip = value
    }

    fun hitungTip() {
        val biaya = _biayaInput.value?.toDoubleOrNull()
        if (biaya == null || biaya <= 0 || persenTip == 0.0) {
            _tipResult.value = ""
            return
        }

        var tip = biaya * persenTip
        if (_bulatkan.value == true) {
            tip = ceil(tip)
        }

        _tipResult.value = "Jumlah tip: Rp
```

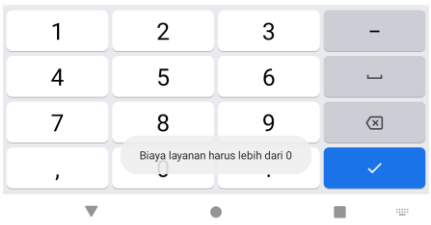
```
46  ${"%.2f".format(tip)}"
47      }
48  }
```

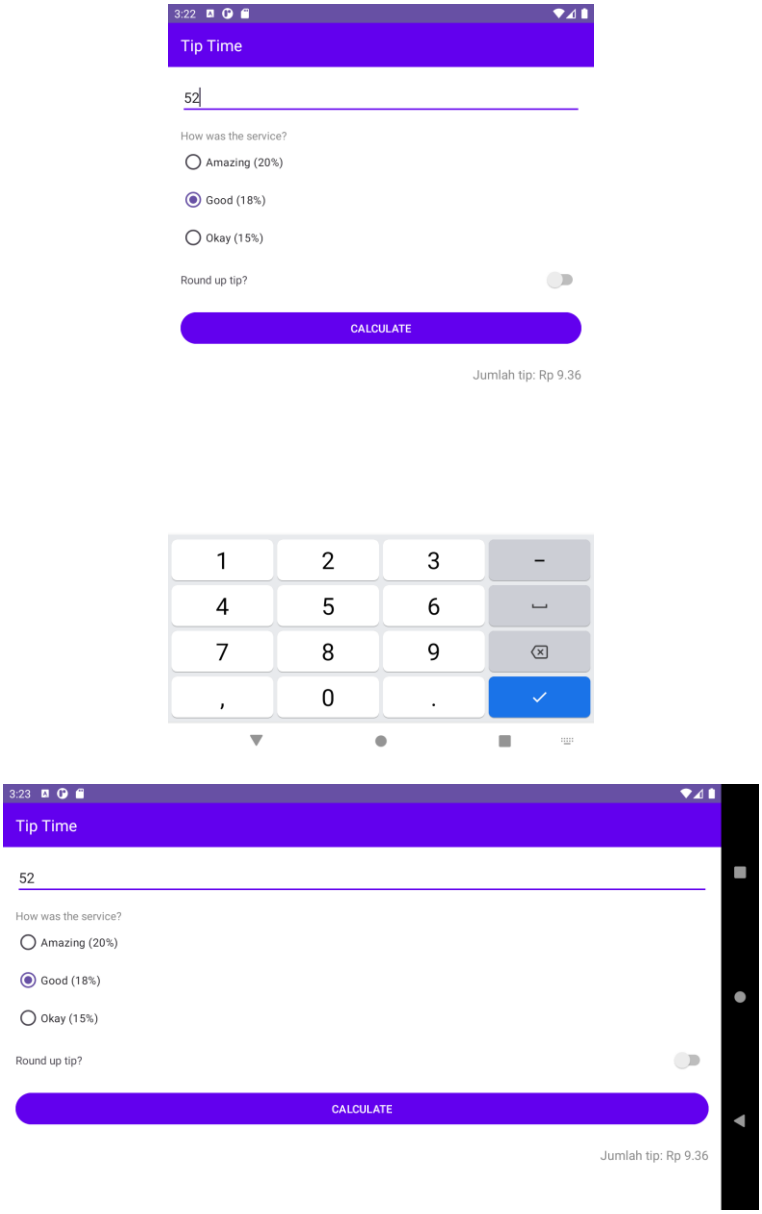
Tabel 3. Source Code Jawaban Soal 1

B. Output Program









Gambar 1. Screenshot Hasil Jawaban Soal 1

C. Pembahasan

1. MainActivity.kt:

Pada baris [1] terdapat fungsi `package com.example.androidlayout` yang digunakan untuk mendeklarasikan bahwa file tersebut termasuk dalam paket bernama com.example.androidlayout. Paket (package) berfungsi untuk mengelompokkan kelas-kelas yang saling berhubungan

agar kode lebih terstruktur dan mudah dikelola. Pada baris [3] hingga [7] terdapat fungsi `import` yang digunakan untuk mengimpor berbagai komponen yang dibutuhkan dalam pengembangan aplikasi Android. Baris `import android.os.Bundle,` yang digunakan untuk mengakses kelas bundle yang biasanya dipakai dalam `onCreate` untuk menyimpan atau mengakses data yang dikirim saat `Activity` dibuat. `android.widget.*,` digunakan untuk mengimpor semua kelas yang ada di dalam sehingga bisa langsung menggunakan kelas itu tanpa harus di tulis lengkap. `android.activiy.viewModels,` digunakan untuk mengimpor fungsi ekstensi Kotlin `ViewModels()` dari jetpack lifecycle library yang berfungsi untuk mengambil instance `ViewModel dengan cara yang aman terhadap` lifecycle `Activity` atau `Fragment`. `import androidx.appcompat.app.AppCompatActivity,` digunakan untuk mengimpor kelas `AppCompatActivity` dari `AndroidX`, basis kelas untuk Activity yang mendukung fitur – fitur modern dan kompatibel dengan Android versi lama. dan `import androidx.lifecycle.Observer` digunakan untuk mengamati perubahan LiveData. Digunakan saat ingin update UI secara otomatis saat data di `ViewModel` berubah. Pada baris [9] terdapat fungsi kelas MainActivity yang mewarisi AppCompactActivity yang merupakan class dasar untuk Activity dengan dukungan kompatibilitas ke Android versi lama.

Pada baris [11] terdapat fungsi `private val tipViewModel: TipViewModel by viewModels()` yang akan digunakan untuk menyimpan dan mengelola data aplikasi, serta delegasi dari Jetpack Lifecycle secara otomatis untuk menjaga data tetap bertahan saat rotasi layar dan sifat nya private, artinya hanya variabel hanya bisa diakses di dalam MainActivity. Pada baris [13] hingga [15] terdapat fungsi `override fun onCreate(savedInstanceState: Bundle?)` digunakan sebagai tempat mulai hidupnya Activity dan semua setup awal dilakukan dan untuk menyimpan dan memulihkan data saat terjadi perubahan konfigurasi (misalnya rotasi layar), sehingga data sebelumnya tidak hilang. Fungsi `super.onCreate(savedInstanceState)` digunakan untuk memastikan lifecycle bawaan Activity berjalan dengan benar. Fungsi

`setContentView(R.layout.activity_main)` digunakan untuk menampilkan layout tampilan (UI) dari file XML ke dalam Activity. Pada baris [17] hingga [22] terdapat fungsi yang digunakan untuk menghubungkan komponen UI di layout XML ke variabel di dalam kode Kotlin. Setiap `findViewById` mencari elemen berdasarkan ID-nya, sehingga bisa digunakan untuk membaca input, menampilkan hasil, atau menangani interaksi. Fungsi ini menginisialisasi semua view (EditText, RadioGroup, Switch, Button, dan TextView) agar bisa digunakan dalam logika aplikasi. Pada baris [24] hingga [35] terdapat fungsi kode yang menggunakan LiveData Observer untuk memantau perubahan data di ViewModel. Saat data berubah, UI akan otomatis diperbarui. biayaInput berguna untuk mengatur nilai EditText agar sesuai dengan data terbaru. Bulatkan digunakan untuk menyesuaikan status Switch (on/off). tipResult digunakan untuk menampilkan hasil tip ke TextView. Sehingga fungsi kode ini membuat UI selalu sinkron dengan data di ViewModel secara otomatis. Pada baris [37] hingga [60] terdapat fungsi kode logika, dimana tombol "Hitung" ditekan, akan berfungsi untuk mengambil input biaya dari pengguna, validasi input (tidak kosong, angka, dan lebih dari 0) serta menentukan persentase tip yang dipilih dari RadioButton. Jika ada yang tidak sesuai (input kosong, salah, atau belum pilih tip), akan muncul pesan kesalahan (Toast). Jadi, kode ini memproses input dan validasi sebelum menghitung tip. Pada baris [64] hingga [67] terdapat fungsi kode yang akan mengatur nilai-nilai yang diperlukan untuk menghitung tip, seperti biaya total, opsi pembulatan, dan persentase tip, kemudian memanggil fungsi untuk menghitung tip berdasarkan parameter tersebut. Semua pengaturan dilakukan dalam objek `tipViewModel`.

2. activity_main.xml

Pada baris [1] terdapat fungsi `<?xml version="1.0" encoding="utf-8"?>` yang digunakan untuk deklarasi XML yang memberi tahu bahwa file ini adalah file XML dan menggunakan versi XML 1.0 serta encoding karakter UTF-8. Ini merupakan bagian penting dari setiap file XML untuk memastikan interpretasi yang benar terhadap isi file, terutama dalam hal karakter yang digunakan. Pada baris [2] hingga [7] terdapat fungsi yang mendefinisikan sebuah LinearLayout di XML untuk

tampilan Android. `xmlns:android="http://schemas.android.com/apk/res/android"` digunakan untuk mendeklarasikan namespace Android yang diperlukan untuk atribut XML Android. `android:layout_width="match_parent"` digunakan untuk menetapkan lebar `LinearLayout` agar sesuai dengan lebar layar induk (parent). `android:layout_height="match_parent"` untuk menetapkan tinggi `LinearLayout` agar sesuai dengan tinggi layar induk (parent). `android:background="@android:color/white"` untuk memberikan latar belakang putih pada LinearLayout. `android:orientation="vertical"` untuk mengatur orientasi LinearLayout secara vertikal, yang berarti elemen-elemen di dalamnya akan ditata secara vertikal (atas ke bawah). Sehingga, kode ini membuat sebuah LinearLayout dengan latar belakang putih dan elemen-elemen di dalamnya disusun secara vertikal. Pada baris [9] hingga [17] terdapat fungsi kode TextView, dengan lebar yang menyesuaikan induknya dan tinggi 56dp. Latar belakang `<TextView` berwarna ungu (#6200EE), dengan teks "Tip Time" yang berwarna putih dan ukuran 20sp. Teks ditempatkan secara vertikal di tengah, dengan padding 16dp di sisi kiri (start), memberikan tampilan yang rapi dan teratur. Pada baris [19] hingga [22] terdapat fungsi `<ScrollView` yang mendefinisikan dengan lebar dan tinggi sesuai dengan ukuran induknya (match_parent). ScrollView ini juga diberi padding sebesar 16dp di semua sisi, sehingga konten di dalamnya tidak menempel langsung ke tepi layar. Dengan penggunaan ScrollView, elemen-elemen di dalamnya dapat digulir secara vertikal jika melebihi ukuran tampilan layar. Pada baris [24] hingga [27] terdapat fungsi `<LinearLayout` yang mendefinisikan dengan lebar yang sesuai dengan lebar induknya (match_parent) dan tinggi yang disesuaikan dengan kontennya (wrap_content). Orientasi LinearLayout ini diatur secara vertikal, yang berarti elemen-elemen di dalamnya akan disusun secara berurutan dari atas ke bawah.

Pada baris [29] hinga [35] terdapa fungsi `>EditText` yang digunakan untuk memasukkan teks, dengan beberapa pengaturan sebagai berikut: lebar EditText disesuaikan dengan lebar induknya (match_parent) dan tinggi disesuaikan dengan kontennya (wrap_content). EditText ini memiliki tint latar belakang berwarna ungu

(#6200EE), menampilkan hint "Cost of Service" saat kosong, dan hanya memungkinkan input berupa angka desimal (inputType="numberDecimal"). ID yang diberikan adalah @+id/etBiaya, yang memungkinkan referensi elemen ini di kode Java atau Kotlin. Pada baris [37] hingga [42] terdapat kode fungsi <TextView yang mendefinisikan dengan lebar dan tinggi yang disesuaikan dengan kontennya (wrap_content). TextView ini memiliki margin atas sebesar 16dp, memberikan jarak antara elemen di atasnya. Teks yang ditampilkan adalah "How was the service?", dengan warna teks abu-abu (#888888). TextView ini biasanya digunakan untuk memberikan pertanyaan atau informasi kepada pengguna. Pada baris [44] hingga [47] terdapat fungsi kode <RadioGroup yang mendefinisikan dengan ID @+id/rgTip yang memiliki lebar sesuai dengan lebar induknya (match_parent) dan tinggi yang disesuaikan dengan kontennya (wrap_content). RadioGroup digunakan untuk mengelompokkan beberapa elemen RadioButton sehingga hanya satu pilihan yang dapat dipilih pada suatu waktu. Kode ini mengatur kontainer untuk elemen-elemen pilihan tip yang akan disediakan dalam aplikasi. Pada baris [49] hingga [65] terdapat fungsi kode <RadioButton yang digunakan untuk mendefinisikan tiga RadioButton dalam sebuah RadioGroup, yang memungkinkan pengguna untuk memilih satu opsi dari beberapa pilihan yang ada. Setiap RadioButton memiliki ID unik (seperti @+id/rb20, @+id/rb18, dan @+id/rb15) serta lebar dan tinggi yang disesuaikan dengan kontennya (wrap_content). Teks yang ditampilkan pada masing-masing RadioButton adalah "Amazing (20%)", "Good (18%)", dan "Okay (15%)", yang menunjukkan persentase tip yang dapat dipilih pengguna. Ketika salah satu dipilih, pilihan lainnya secara otomatis akan terhapus. Pada baris [68] hingga [84] terdapat fungsi kode <LinearLayout yang didalamnya berisi <TextView dan <Switch, dengan tujuan memberikan opsi kepada pengguna untuk membulatkan tip. LinearLayout ini memiliki lebar penuh (match_parent), tinggi menyesuaikan kontennya (wrap_content), margin atas 16dp, dan orientasi horizontal agar kedua elemen ditampilkan sejajar secara mendatar. TextView menampilkan teks "Round up tip?" dan menggunakan layout_weight="1" serta layout_width="0dp" agar mengisi ruang kosong yang tersedia. Switch dengan ID @+id/switchBulatkan berada di sebelah kanan, memungkinkan pengguna untuk mengaktifkan atau menonaktifkan

opsi pembulatan tip. Pada baris [87] hingga [94] terdapat fungsi `<Button` yang digunakan untuk sebuah LinearLayout horizontal yang berisi TextView dan Switch, dengan tujuan memberikan opsi kepada pengguna untuk membulatkan tip. LinearLayout ini memiliki lebar penuh (match_parent), tinggi menyesuaikan kontennya (wrap_content), margin atas 16dp, dan orientasi horizontal agar kedua elemen ditampilkan sejajar secara mendatar. TextView menampilkan teks "Round up tip?" dan menggunakan layout_weight="1" serta layout_width="0dp" agar mengisi ruang kosong yang tersedia. Switch dengan ID @+id/switchBulatkan berada di sebelah kanan, memungkinkan pengguna untuk mengaktifkan atau menonaktifkan opsi pembulatan tip. Pada baris [96] hingga [104] terdapat fungsi `<TextView` yang digunakan untuk mendefinisikan dengan ID @+id/tvHasil yang memiliki lebar penuh (match_parent) dan tinggi menyesuaikan kontennya (wrap_content). TextView ini diberi margin atas sebesar 24dp, teksnya "Tip Amount", warna teks abu-abu (#888888), dan ukuran teks 16sp. Properti android:gravity="end" membuat teks diratakan ke kanan. TextView ini berfungsi untuk menampilkan hasil perhitungan tip kepada pengguna. Sehingga keseluruhan dari kode ini berguna untuk mendesain antarmuka aplikasi kalkulator tip di Android, yang terdiri dari input biaya layanan, pilihan persentase tip (melalui RadioButton), opsi pembulatan tip (dengan Switch), tombol untuk menghitung tip, dan TextView untuk menampilkan hasilnya. Seluruh elemen disusun secara vertikal dalam LinearLayout, dengan ScrollView agar tampilan tetap dapat digulir jika kontennya melebihi layar.

3. TipViewModel

Pada baris [1] terdapat fungsi `package com.example.androidlayout` yang digunakan untuk mendeklarasikan bahwa file tersebut termasuk dalam paket bernama com.example.androidlayout. Paket (package) berfungsi untuk mengelompokkan kelas-kelas yang saling berhubungan agar kode lebih terstruktur dan mudah dikelola. Pada baris [3] hingga [6] terdapat fungsi `import` yang digunakan untuk mengimpor berbagai komponen yang dibutuhkan dalam pengembangan aplikasi Android. Baris `import androidx.lifecycle.LiveData` digunakan untuk menyimpan data yang bisa diamati (observable) dan akan otomatis diperbarui saat nilainya berubah.

`import androidx.lifecycle.MutableLiveData` digunakan untuk mengatur atau memperbarui data agar keamanan nya terjaga. `import androidx.lifecycle.ViewModel` digunakan untuk memisahkan logic dan tampilan, dan menjaga data tetap hidup saat rotasi layar. `import kotlin.math.ceil` digunakan untuk membulatkan angka ke atas ke bilangan bulat terdekat, digunakaan saat ingin pembulatan hasil perhitungan tip ke atas. Pada baris [8] terdapat fungsi class TipViewModel: ViewModel() yang digunakan mendefinisikan kelas untuk menyimpan dan mengelola data untuk UI, secara terpisah dari lifcycle UI. Pada baris [10] hingga [19] terdapat beberapa fungsi private yang berfungsi untuk menyimpan dan mengelola data internal yang dibutuhkan dalam progress perhitungan tip, yaitu `_biayaInput` digunakan untuk menyimpan input biaya dari pengguna. `_tipResult` digunakan untuk menyimpan hasil perhitungan tip yang akan ditampilkan ke UI. `_bulatkan` digunakan untuk menyimpan status apakah hasil tip perlu dibulatkan ke atas atau tidak. `persenTip` digunakan untuk menyimpan nilai persentase tip yang akan digunakan dalam perhitungan. Semua data ini bersifat privat agar tidak bisa diubah langsung dari luar ViewModel, dan hanya bisa diakses secara aman melalui versi LiveData yang publik. Pada baris [21] hingga [46] terdapat fungsi setter yang digunakan untuk mengatur nilai input dari pengguna ke dalam variabel privat `MutableLiveData` atau variabel biasa. `fun setBiaya(biaya: String)` digunakan untuk mengatur nilai biaya yang dimasukkan pengguna. `fun setBulatkan(value: Boolean)` digunakan untuk mengatur apakah hasil tip perlu dibulatkan ke atas. `fun setPersenTip(value: Double)` digunakan untuk mengatur nilai persentase tip yang digunakan saat menghitung. `fungsi hitungTip()`, digunakan untuk menghitung jumlah tip berdasarkan input pengguna. Dengan langkah langkah mengambil nilai `biayaInput` dan mengubahnya ke `Double`, mengecek apakah input valid (tidak kosong, tidak nol, dan persen tip tidak nol), menghitung `tip = biaya * persenTip`, jika opsi `bulatkan` aktif, hasil tip dibulatkan ke atas (ceil) serta menyimpan hasil ke `_tipResult` dalam format "`Jumlah tip: Rp xx.xx`"

D. Tautan Git

https://github.com/SheilaSabina/Praktikum-Mobile/tree/master/ANDROIDLAYOUT