LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 2

ANDROID LAYOUT

Oleh:

Nur Hikmah **NIM. 2310817120010**

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
APRIL 2025

LEMBAR PENGESAHAN
LAPORAN PRAKTIKUM PEMROGRAMAN MOBILE
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 : Nur Hikmah
NIM : 2310817120010

Menyetujui, Asisten Praktikum	Mengetahui, Dosen Penanggung Jawab Praktikum
Zulfa Auliya Akbar NIM. 2210817210026	Muti`a Maulida S.Kom M.T.I NIP. 19881027 201903 20 13

DAFTAR ISI

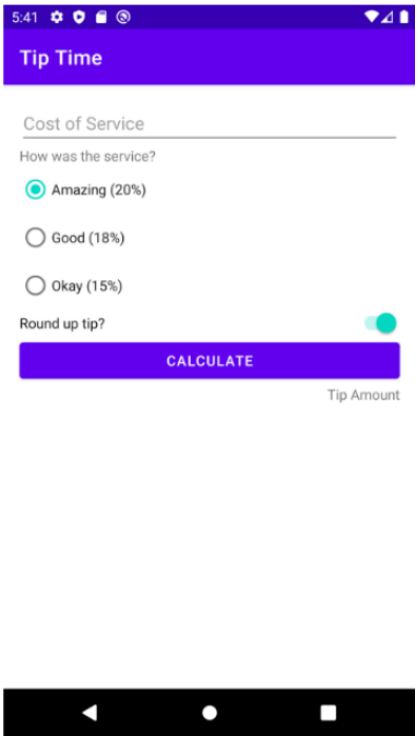
DAFTAR GAMBAR

DAFTAR TABEL

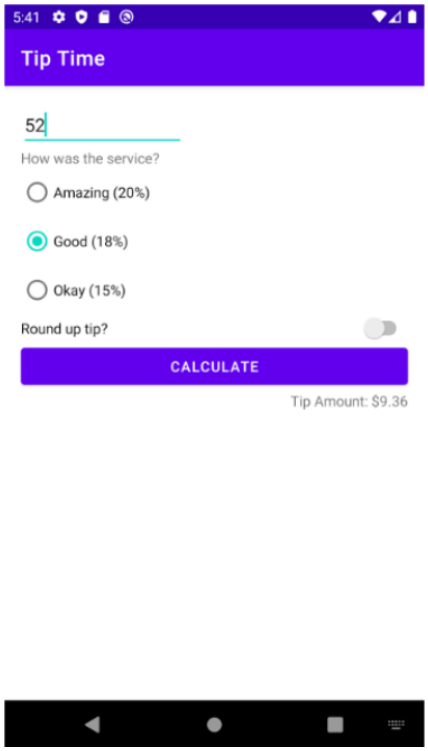
SOAL 1

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.
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.modul2
2
3  import android.os.Bundle
4  import android.widget.*
5  import androidx.activity.viewModels
6  import androidx.appcompat.app.AppCompatActivity
7  import java.text.NumberFormat
8  import kotlin.math.ceil
9
10 class MainActivity : AppCompatActivity() {
11
12     private val viewModel: TipViewModel by viewModels()
13
14     override fun onCreate(savedInstanceState: Bundle?) {
15         super.onCreate(savedInstanceState)
16         setContentView(R.layout.activity_main)
17
18         val costOfServiceEditText =
   findViewById<EditText>(R.id.edit_cost_of_service)
19         val tipOptions =
   findViewById<RadioGroup>(R.id.tip_options)
20         val roundUpSwitch =
   findViewById<Switch>(R.id.switch_round_up)
21         val calculateButton =
```

```
    findViewById<Button>(R.id.button_calculate)
22          val tipResultTextView =
    findViewById<TextView>(R.id.text_tip_result)
23
24          tipResultTextView.text = viewModel.tipResult
25
26          calculateButton.setOnClickListener {
27              val costInput =
    costOfServiceEditText.text.toString()
28              val cost = costInput.toDoubleOrNull()
29
30              if (costInput.isEmpty()) {
31                  tipResultTextView.text = ""
32                  Toast.makeText(this, "Input tidak boleh
    kosong. Harap masukkan angka!", Toast.LENGTH_SHORT).show()
33                  return@setOnClickListener
34              }
35
36              if (cost == null || cost <= 0.0) {
37                  tipResultTextView.text = ""
38                  Toast.makeText(this, "Harap masukkan angka
    yang valid dan lebih dari 0!", Toast.LENGTH_SHORT).show()
39                  return@setOnClickListener
40              }
41
42              val tipPercentage = when
    (tipOptions.checkedRadioButtonId) {
43                  R.id.radio_amazing -> 0.20
44                  R.id.radio_good -> 0.18
45                  else -> 0.15
46              }
47              var tip = cost * tipPercentage
48
49              if (roundUpSwitch.isChecked) {
50                  tip = ceil(tip)
51              }
52
53              val formattedTip =
    NumberFormat.getCurrencyInstance().format(tip)
54
55              viewModel.tipResult = "Tip Amount: $formattedTip"
56              tipResultTextView.text = viewModel.tipResult
57          }
58      }
59  }
```

Tabel 1. Source Code Jawaban Soal 1

2. TipViewModel.kt

```
1  package com.example.modul2
2
3  import androidx.lifecycle.LiveData
4  import androidx.lifecycle.MutableLiveData
5  import androidx.lifecycle.ViewModel
6
7  class TipViewModel : ViewModel() {
8      var tipResult: String = ""
9
10     private val _tipAmount = MutableLiveData<Double>()
11     val tipAmount: LiveData<Double> get() = _tipAmount
12
13     fun calculateTip(totalAmount: Double, tipPercentage:
   Double) {
14         val tip = totalAmount * (tipPercentage / 100)
15         _tipAmount.value = tip
16     }
17
18     fun resetTip() {
19         _tipAmount.value = 0.0
20     }
21 }
```

Tabel 2. Source Code Jawaban Soal 1

3. activity_main.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:tools="http://schemas.android.com/tools"
4      xmlns:app="http://schemas.android.com/apk/res-auto"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      tools:context=".MainActivity">
8
9      <TextView
10         android:id="@+id/text_header"
11         android:layout_width="0dp"
12         android:layout_height="65dp"
13         android:background="#6200EE"
14         android:gravity="center_vertical"
15         android:paddingStart="16dp"
16         android:paddingEnd="16dp"
17         android:text="Tip Time"
18         android:textColor="@android:color/white"
19         android:textSize="23sp"
20         android:textStyle="bold"
21         app:layout_constraintTop_toTopOf="parent"
22         app:layout_constraintStart_toStartOf="parent"
23         app:layout_constraintEnd_toEndOf="parent" />
```

```
24
25      <ScrollView
26          android:id="@+id/scroll_area"
27          android:layout_width="0dp"
28          android:layout_height="0dp"
29          android:padding="16dp"
30          android:fillViewport="true"
31         app:layout_constraintTop_toBottomOf="@id/text_header"
32          app:layout_constraintBottom_toBottomOf="parent"
33          app:layout_constraintStart_toStartOf="parent"
34          app:layout_constraintEnd_toEndOf="parent">
35
36          <LinearLayout
37              android:layout_width="match_parent"
38              android:layout_height="wrap_content"
39              android:orientation="vertical">
40
41              <EditText
42                  android:id="@+id/edit_cost_of_service"
43                  android:layout_width="match_parent"
44                  android:layout_height="wrap_content"
45                  android:hint="Cost of Service"
46                  android:inputType="numberDecimal" />
47
48              <TextView
49                  android:layout_width="match_parent"
50                  android:layout_height="wrap_content"
51                  android:hint="How was the service?"
52                  android:textSize="15sp" />
53
54              <RadioGroup
55                  android:id="@+id/tip_options"
56                  android:layout_width="match_parent"
57                  android:layout_height="wrap_content"
58                  android:orientation="vertical"
59                  android:paddingTop="8dp">
60
61                  <RadioButton
62                      android:id="@+id/radio_amazing"
63                      android:layout_width="wrap_content"
64                      android:layout_height="wrap_content"
65                      android:checked="true"
66                      android:text="Amazing (20%)" />
67
68                  <RadioButton
69                      android:id="@+id/radio_good"
70                      android:layout_width="wrap_content"
71                      android:layout_height="wrap_content"
72                      android:text="Good (18%)" />
73
74                  <RadioButton
75                      android:id="@+id/radio_okay"
```

```
                    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:orientation="horizontal"
                android:layout_marginTop="16dp"
                android:gravity="center_vertical">

                <TextView
                    android:layout_width="0dp"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"
                    android:text="Round up tip?"
                    android:textSize="15sp"
                    android:textStyle="bold" />

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

            <android.widget.Button
                android:id="@+id/button_calculate"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="20dp"
                android:background="@drawable/button_shape"
                android:text="CALCULATE"
                android:textSize="17sp"
                android:textColor="@android:color/white" />

            <TextView
                android:id="@+id/text_tip_result"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="16dp"
                android:text="Tip Amount:"
                android:textSize="15sp"
                android:textStyle="bold"
                android:gravity="end"/>
        </LinearLayout>
    </ScrollView>
</androidx.constraintlayout.widget.ConstraintLayout>
```

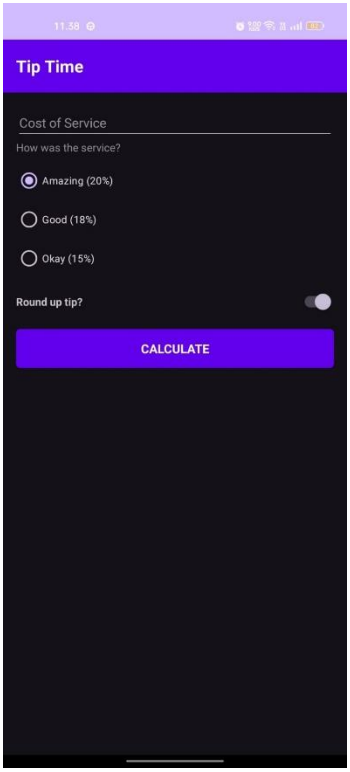
Tabel 3. Source Code Jawaban Soal 1

4. button_shape.xml

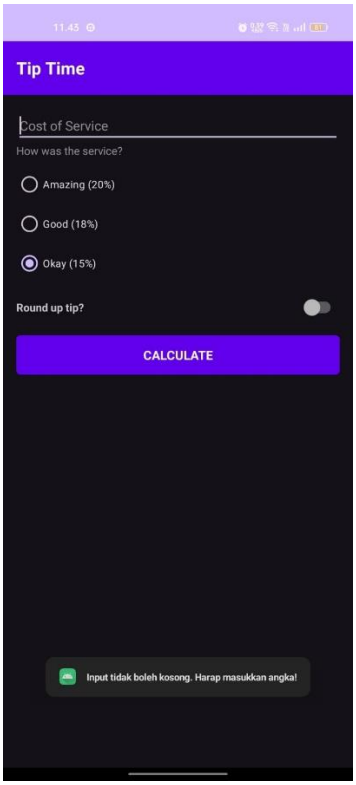
```
1   <?xml version="1.0" encoding="utf-8"?>
2   <selector
    xmlns:android="http://schemas.android.com/apk/res/android">
3       <item android:state_pressed="true">
4           <shape android:shape="rectangle">
5               <solid android:color="#6200EE" />
6               <corners android:radius="5dp" />
7           </shape>
8       </item>
9
10      <item>
11          <shape android:shape="rectangle">
12              <solid android:color="#6200EE" />
13              <corners android:radius="5dp" />
14          </shape>
15      </item>
16  </selector>
```

Tabel 4. Source Code Jawaban Soal 1

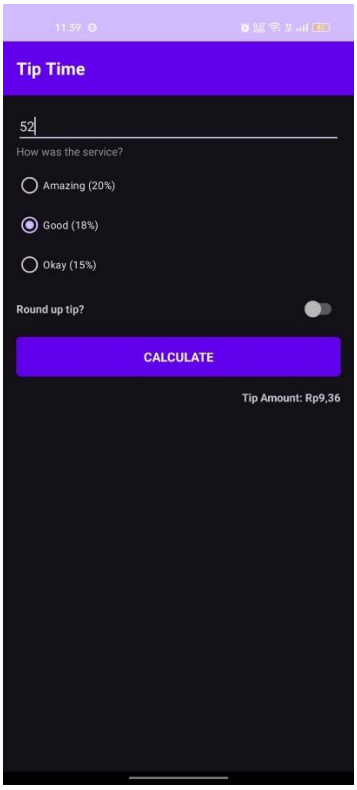
B. Output Program



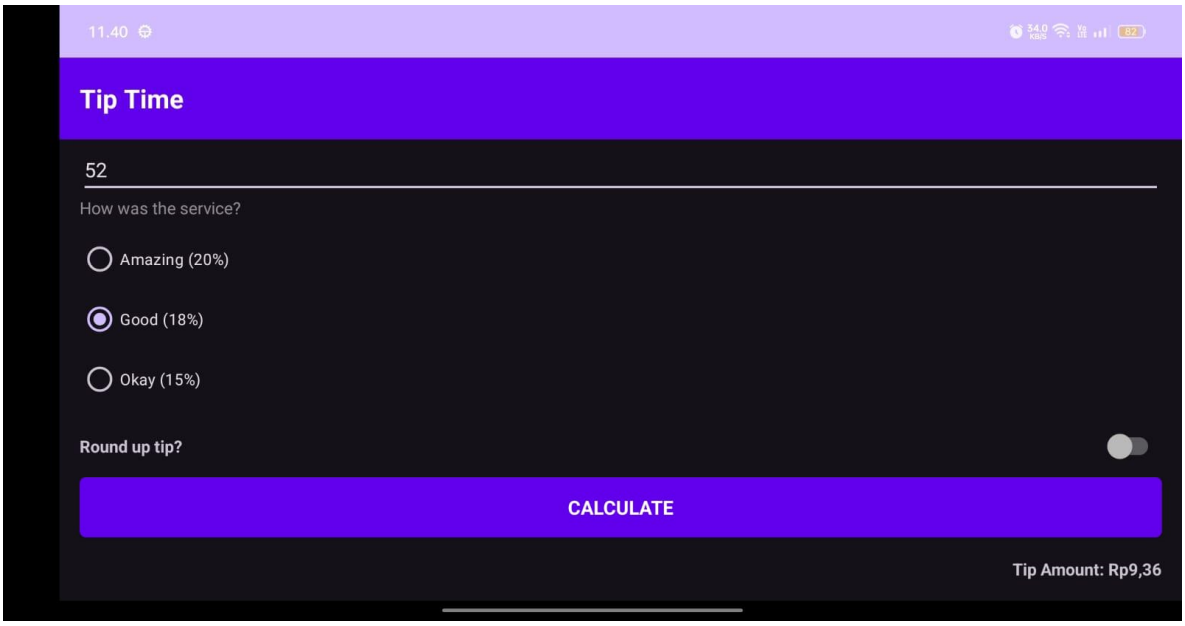
Gambar 3. Screenshot Hasil Jawaban Soal 1 Tampilan Awal Aplikasi



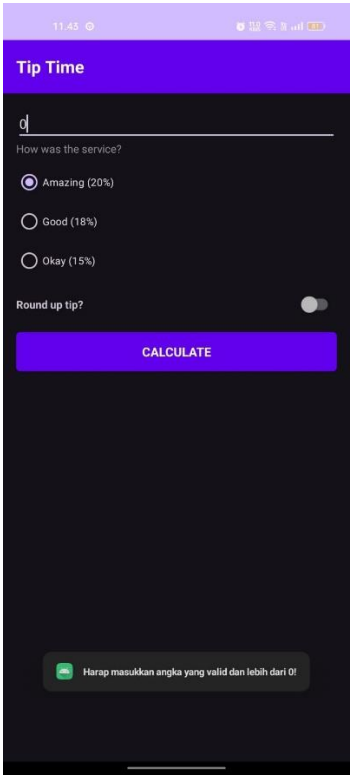
Gambar 4. Screenshot Hasil Jawaban Soal 1 Tampilan Error Ketika Tidak Ada Penginputan (Kosong)



Gambar 5. Screenshot Hasil Jawaban Soal 1 Tampilan Aplikasi Setelah Dijalankan



Gambar 6. Screenshot Hasil Jawaban Soal 1 Tampilan Aplikasi Setalah Dijalan dengan Merotasi Device



Gambar 7. Screenshot Hasil Jawaban Soal 1 Tampilan Error Ketika Menginputkan 0

C. Pembahasan

1. MainActivity.kt:

Pada line 1, `package com.example.modul2` digunakan untuk menentukan nama package dari aplikasi Android. Package ini adalah namespace unik untuk memisahkan file dalam aplikasi. Kemudian, pada line 3 sampai 8, terdapat pengimporan kelas dan beberapa fungsi yang diperlukan, diantaranya yaitu `Bundle` digunakan untuk menyimpan data saat Activity dibuat, `android.widget.*` untuk semua widget Android seperti `EditText`, `Button`, `TextView`, dsb, `viewModels` digunakan untuk mengakses ViewModel dalam activity, `AppCompatActivity` ialah Activity dasar yang kompatibel ke belakang, `NumberFormat` digunakan untuk memformat angka ke bentuk mata uang, dan `ceil` ialah fungsi untuk membulatkan angka ke atas. Lalu, pada line 10 mendeklarasikan `MainActivity`, `MainActivity` adalah kelas utama yang mewarisi dari `AppCompatActivity`, yaitu komponen dasar dari sebuah tampilan (activity) di Android. Line 12 menggunakan ViewModel dari `androidx.lifecycle` untuk menyimpan dan mengelola data yang bertahan saat rotasi layar atau perubahan konfigurasi. Selain itu, `by viewModels()` adalah cara delegasi untuk mendapatkan instance `TipViewModel` yang sesuai dengan lifecycle `MainActivity`. Pada line 14 terdapat override fungsi `onCreate`, fungsi ini dipanggil saat Activity dibuat pertama kali. Line 15, memanggil implementasi `onCreate` dari `AppCompatActivity` untuk melakukan inisialisasi standar. Line 16, menghubungkan Activity dengan layout XML (`activity_main.xml`) agar komponen UI bisa digunakan.

Pada line 18, mencari komponen `EditText` berdasarkan ID `edit_cost_of_service` di layout dan menyimpannya ke variabel. Line 19, mencari komponen `RadioGroup` berdasarkan ID `tip_options`. Line 20, mencari komponen `Switch` berdasarkan ID `switch_round_up`. Line 21, mencari komponen `Button` berdasarkan ID `button_calculate`. Line 22, mencari komponen `TextView` berdasarkan ID `text_tip_result`. Line 24, menampilkan nilai `tipResult` yang sudah disimpan di ViewModel ke dalam

`TextView`. Line 26, menambahkan event listener ke `calculateButton`. Artinya, saat tombol diklik, kode di dalam blok ini akan dijalankan. Line 27, mengambil teks dari `EditText` dan mengubahnya menjadi `String`. Line 28, mencoba mengubah input `costInput` menjadi angka `Double`. Jika gagal (misal huruf atau kosong), hasilnya `null`. Line 30 sampai 34, jika input kosong maka membersihkan teks hasil tip, menampilkan pesan toast "Input tidak boleh kosong. Harap masukkan angka!", dan menghentikan proses klik tombol (`return@setOnClickListener`). Line 36 sampai 40, jika input bukan angka valid atau angka ≤ 0, maka akan membersihkan teks hasil tip, menampilkan pesan kesalahan (toast) "Harap masukkan angka yang valid dan lebih dari 0!", dan menghentikan proses. Line 42 sampai 46, menentukan persentase tip berdasarkan radio button yang dipilih, yaitu Amazing (20%), Good (18%), dan lainnya (15%). Line 47, menghitung jumlah tip awal = biaya * persentase tip.

Pada line 49 sampai 51 disitu, jika pengguna mencentang `roundUpSwitch`, bulatkan tip ke atas menggunakan `ceil()`. Line 53, mengubah angka tip menjadi format mata uang lokal (Rp, $, atau yang sesuai). Line 55, menyimpan hasil tip yang sudah diformat ke dalam ViewModel (`tipResult`). Line 56 sampai 59, menampilkan hasil tip di `TextView`, serta mengakhiri dari `calculateButton.setOnClickListener`, `onCreate`, dan `class MainActivity`.

2. TipViewModel.kt

Pada line 1, menyatakan bahwa file ini berada dalam package `com.example.modul2`. Package digunakan untuk mengelompokkan file agar lebih terorganisir. Line 3 sampai 5 ada beberapa pengimporan class, diantaranya yaitu `LiveData` ialah objek data yang bisa di-observe (diperhatikan), berguna untuk UI, `MutableLiveData` ialah versi bisa diubah dari `LiveData`, dan `ViewModel` digunakan untuk komponen yang menyimpan dan mengelola data UI secara aman dari perubahan konfigurasi (seperti rotasi layar). Line 7 mendefinisikan class `TipViewModel`, yang mewarisi (`extends`) `ViewModel`. ViewModel ini

digunakan untuk menyimpan data yang dibutuhkan `MainActivity`, agar tidak hilang saat rotasi atau perubahan konfigurasi. Line 8 terdapat variabel publik bertipe `String`, yang akan menyimpan teks hasil perhitungan tip (“`Tip Amount: Rp2,00`”), ini bisa langsung diakses dari activity seperti `viewModel.tipResult`. Line 10, mendefinisikan variabel private `_tipAmount` bertipe `MutableLiveData<Double>`. `MutableLiveData` berarti nilainya bisa diubah (mutable). Angka desimal (`Double`) menyimpan nilai tip mentah sebelum diformat. Tanda _ (underscore) adalah konvensi penamaan untuk variabel private backing dari `LiveData`. Line 11, membuat versi publik dari `_tipAmount`, tapi dalam bentuk `LiveData` yang read-only dari luar. Tujuannya agar hanya ViewModel yang bisa mengubah nilainya, sedangkan komponen UI hanya bisa membaca.

Pada line 13 terdapat fungsi publik yang menerima dua parameter, yaitu `totalAmount` digunakan untuk jumlah biaya dan `tipPercentage` untuk persentase tip (misal 15 untuk 15%). Line 14, melakukan penghitungan nilai tip berdasarkan rumus, `tip = totalAmount × (persentase ÷ 100)`. Misal: 100.000 × 0.15 = 15.000. Line 15, menyimpan hasil tip ke dalam `_tipAmount` agar `LiveData` bisa memicu observer di UI. Line 18 terdapat fungsi untuk mereset nilai tip ke 0, yang bisa digunakan ketika pengguna mau menghitung ulang. Line 19 sampai 21, mengatur ulang `LiveData` menjadi 0.0, yang berarti tidak ada tip saat ini, serta melakukan penutupan fungsi `resetTip` dan class `TipViewModel`.

3. activity_main.xml

Pada line 1, terdapat deklarasi XML versi 1.0, menandakan file ini adalah file XML dengan encoding UTF-8. Pada line 2, Root layout dari tampilan ini adalah `ConstraintLayout`. Artinya semua elemen di dalamnya akan diatur posisinya berdasarkan constraint (seperti "terhubung ke atas/bawah/kiri/kanan"). Kemudian, line 3 dan 4 mendeklarasikan namespace supaya kita bisa pakai atribut `android:`, `tools:`, dan `app:` dalam XML. Line 5 dan 6, `ConstraintLayout` disitu akan berfungsi untuk mengisi seluruh layar. Line 7, menandakan bahwa layout tersebut

dipakai oleh `MainActivity`. `tools:` hanya untuk preview di Android Studio, tidak berpengaruh ke app yang jalan. Line 9 sampai 23 merupakan bagian header TextView (judul). Bagian ini berisikan teks judul “Tip Time” di bagian atas, dengan background ungu `#6200EE`, teks putih, ukuran besar (`23sp`), huruf tebal (`bold`), dan diatur supaya posisinya nempel di atas layar (pakai `layout_constraintTop_toTopOf="parent"`). Pada line 25 sampai 34 adalah ScrollView, bagian bawah dari header. ScrollView memungkinkan isi form yang bisa di-scroll kalau kontennya panjang. ScrollView ini dihubungkan ke bawah header dan ke bawah parent. Di dalam `ScrollView` ada `LinearLayout` vertical yaitu di line 36 sampai 39. Komponen-komponen di dalamnya akan ditumpuk vertikal dari atas ke bawah.

Line 41 sampai 46 merupakan bagian `EditText` (input biaya servis). Input untuk biaya layanan (service) dari user, yang hanya menerima angka desimal (karena `inputType="numberDecimal"`). Line 48 sampai 52 masuk ke bagian `TextView` (label untuk pertanyaan). Disini memberi label teks “How was the service?”, ini cuma teks biasa, tidak bisa diklik. Line 54 sampai 79 ialah bagian `RadioGroup` (pilihan kualitas servis). Kumpulan pilihan jawaban dengan 3 `RadioButton`, Amazing (20%), Good (18%), Okay (15%). Ini hanya bisa memilih satu opsi dalam satu waktu. Line 81 sampai 100 merupakan bagian `LinearLayout` (round up pilihan). Bagian ini membuat baris horizontal yang berisi, `TextView` “Round up tip?” dan `Switch` untuk memilih apakah tip dibulatkan ke atas. Line 102 sampai 110 bagian `Button` (hitung tip). Bagian ini ketika tombol diklik akan menghitung tip-nya. Ada background khusus (`@drawable/button_shape`) untuk mengatur shape dari tombol tersebut yang ingin membentuk seperti kotak. Line 112 sampai 123 ialah bagian `TextView` (hasil perhitungan). Bagian ini akan menampilkan hasil perhitungan tip dan diatur supaya rata kanan (`gravity="end"`) di dalam LinearLayout.

4. button_shape.xml

Line 1 ialah baris standar di semua file XML Android. Line 2, selector adalah daftar kondisi tampilan (misalnya tombol ditekan, aktif, tidak aktif, dll). Ini digunakan untuk mengubah gaya tombol sesuai statusnya (pressed, focused, default, dll). Line 3 sampai 8 ialah kondisi saat tombol ditekan (`state_pressed="true"`), saat tombol ditekan, maka tampilannya akan berbentuk kotak (`rectangle`), warna solid ungu `#6200EE`, dan sudut-sudut melengkung dengan radius `5dp`. Line 10 sampai 16 ialah kondisi default (tidak ditekan), ini tampilan default tombol ketika tidak ditekan tetap ungu dan sudutnya melengkung. Karena warna pressed dan default sama, maka tampilannya mungkin terlihat tidak berubah saat ditekan.

D. Tautan Git

Berikut adalah tautan untuk source code yang telah dibuat.

https://github.com/nrhikmahhh99/Pemrograman-Mobile