LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 2

ANDROID LAYOUT

Oleh:

Galih Aji Sabdaraya NIM. 2310817210005

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 Praktikum ini dikerjakan oleh:

Nama Praktikan : Galih Aji Sabdaraya
NIM : 2310817210005

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

DAFTAR ISI

DAFTAR GAMBAR

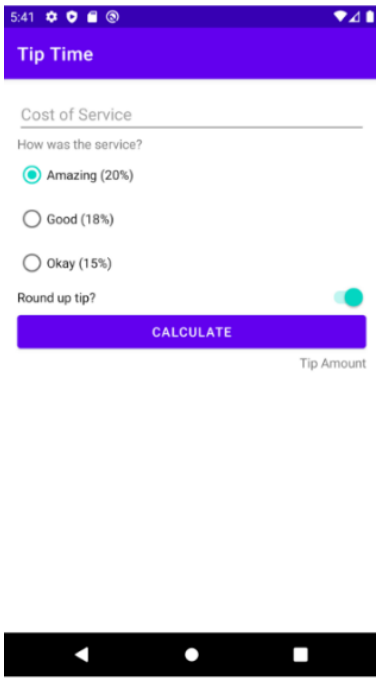
DAFTAR TABEL

SOAL 1

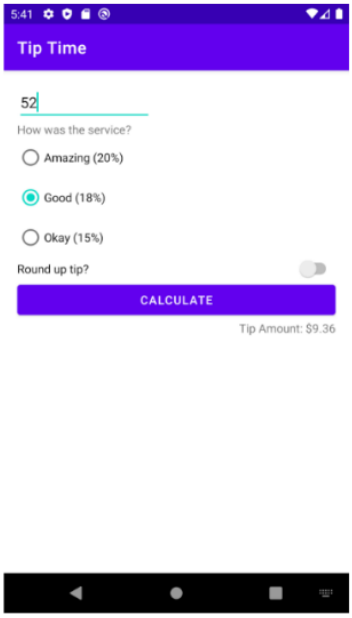
Soal Praktikum:

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. Contoh Tampilan Awal Aplikasi



Gambar 2. Contoh Tampilan Aplikasi Setelah Dijalankan

A. Source Code

1. MainActivity.kt

Tabel 1. Source Code MainActivity.kt

```
 1  package com.example.tipcalculator
 2
 3  import android.content.Context
 4  import android.os.Bundle
 5  import android.view.inputmethod.EditorInfo
 6  import android.view.inputmethod.InputMethodManager
 7  import android.widget.Toast
 8  import androidx.activity.enableEdgeToEdge
 9  import androidx.activity.viewModels
10  import androidx.appcompat.app.AppCompatActivity
11  import androidx.core.view.ViewCompat
12  import androidx.core.view.WindowInsetsCompat
13  import androidx.lifecycle.Observer
14  import
    com.example.tipcalculator.databinding.ActivityMainBinding
15  import com.example.tipcalculator.viewmodel.TipViewModel
16
17  class MainActivity : AppCompatActivity() {
18
19      private lateinit var  binding: ActivityMainBinding
20      private  val viewModel: TipViewModel by viewModels()
```

```
21
22      override fun onCreate(savedInstanceState: Bundle?) {
23          super.onCreate(savedInstanceState)
24          enableEdgeToEdge()
25
26          binding =
    ActivityMainBinding.inflate(layoutInflater)
27          setContentView(binding.root)
28
29  ViewCompat.setOnApplyWindowInsetsListener(binding.topBar) {
    v, insets ->
30              val topInset =
    insets.getInsets(WindowInsetsCompat.Type.systemBars()).top
31              v.setPadding(v.paddingLeft, topInset,
    v.paddingRight, v.paddingBottom)
32              insets
33          }
34
35          viewModel.tipAmount.observe(this, Observer { tip ->
36              binding.tipResult.text = tip
37          })
38
39          binding.calculateButton.setOnClickListener {
40              val cost =
    binding.costService.text.toString().toDoubleOrNull()
41
42              if(cost == null || cost < 0.0){
43                  Toast.makeText(this, "Silahkan masukkan
    harga yang valid!", Toast.LENGTH_SHORT).show()
44                  return@setOnClickListener
45              }
46
47              hideKeyboard()
48
49              viewModel.setCostService(cost)
50              val tipPercent = when
    (binding.tipOption.checkedRadioButtonId){
51                  R.id.optionAmazing -> 0.20
52                  R.id.optionGood -> 0.18
53                  R.id.optionOkay -> 0.15
54                  else -> 0.15
55              }
56
57              viewModel.setTipPercent(tipPercent)
58              viewModel.setRoundUp(binding.roundTip.isChecked)
59              viewModel.calculateTip()
```

```
60          }
61
62          binding.costService.setOnEditorActionListener { v,
    actionId, event ->
63              if (actionId == EditorInfo.IME_ACTION_DONE) {
64                  binding.calculateButton.performClick()
65                  hideKeyboard()
66                  true
67              } else {
68                  false
69              }
70          }
71      }
72
73      private fun hideKeyboard() {
74          val imm =
    getSystemService(Context.INPUT_METHOD_SERVICE) as
    InputMethodManager

75  imm.hideSoftInputFromWindow(binding.costService.windowToken,
    0)
76      }
77  }
```

2. TipViewModel.kt

Tabel 2. Source Code TipViewModel.kt

```
 1  package com.example.tipcalculator.viewmodel
 2
 3  import androidx.lifecycle.LiveData
 4  import androidx.lifecycle.MutableLiveData
 5  import androidx.lifecycle.ViewModel
 6  import kotlin.math.ceil
 7
 8  class TipViewModel : ViewModel() {
 9      private val _tipAmount = MutableLiveData<String>()
10      val tipAmount: LiveData<String> get() = _tipAmount
11
12      private var costService = 0.0
13      private var tipPercent = 0.20
14      private var roundUp = true
15
16      fun setCostService (cost: Double){
17          costService = if(cost < 0) 0.0 else cost
18      }
19
```

```
20      fun setTipPercent(percent: Double){
21          tipPercent = percent
22      }
23
24      fun setRoundUp(round: Boolean){
25          roundUp = round
26      }
27
28      fun calculateTip() {
29          var tip = costService * tipPercent
30          if(roundUp) {
31              tip = ceil(tip)
32          }
33
34          _tipAmount.value = "Tip Amount: $%.2f" .format(tip)
35      }
36  }
```

3. activity_main.xml

Tabel 3. Source Code activity_main.xml

```
 1  <?xml version="1.0" encoding="utf-8"?>
 2  <ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
 3      xmlns:app="http://schemas.android.com/apk/res-auto"
 4      xmlns:tools="http://schemas.android.com/tools"
 5      android:id="@+id/scrollView"
 6      android:layout_width="match_parent"
 7      android:layout_height="match_parent"
 8      android:fitsSystemWindows="true"
 9      tools:context=".MainActivity">
10
11      <androidx.constraintlayout.widget.ConstraintLayout
12          android:id="@+id/main"
13          android:layout_width="match_parent"
14          android:layout_height="wrap_content"
15          android:paddingBottom="16dp">
16
17          <TextView
18              android:id="@+id/topBar"
19              android:layout_width="match_parent"
20              android:layout_height="50dp"
21              android:background="#7223BD"
22              android:gravity="center_vertical"
23              android:paddingStart="16dp"
24              android:paddingEnd="16dp"
```

```
25              android:text="Tip Time"
26              android:textAlignment="gravity"
27              android:textColor="#FFFFFF"
28              android:textColorLink="#5CB9C5"
29              android:textSize="18sp"
30              app:layout_constraintStart_toStartOf="parent"
31              app:layout_constraintTop_toTopOf="parent" />
32
33          <EditText
34              android:id="@+id/costService"
35              android:layout_width="match_parent"
36              android:layout_height="50dp"
37              android:layout_margin="16dp"
38              android:layout_marginTop="16dp"
39              android:ems="10"
40              android:hint="Cost of Service"
41              android:inputType="number|numberDecimal"
42              app:layout_constraintStart_toStartOf="parent"

43  app:layout_constraintTop_toBottomOf="@+id/topBar" />
44
45          <TextView
46              android:id="@+id/serviceQuestion"
47              android:layout_width="wrap_content"
48              android:layout_height="wrap_content"
49              android:layout_marginStart="16dp"
50              android:text="How was the service?"
51              android:textColor="#808080"
52              app:layout_constraintStart_toStartOf="parent"

53  app:layout_constraintTop_toBottomOf="@+id/costService" />
54
55          <RadioGroup
56              android:id="@+id/tipOption"
57              android:layout_width="wrap_content"
58              android:layout_height="wrap_content"
59              android:layout_marginStart="16dp"
60              android:checkedButton="@id/optionAmazing"
61              app:layout_constraintStart_toStartOf="parent"

62  app:layout_constraintTop_toBottomOf="@+id/serviceQuestion">
63
64              <RadioButton
65                  android:id="@+id/optionAmazing"
66                  android:layout_width="match_parent"
67                  android:layout_height="wrap_content"
68                  android:text="Amazing (20%)" />
```

```
 69
 70              <RadioButton
 71                  android:id="@+id/optionGood"
 72                  android:layout_width="match_parent"
 73                  android:layout_height="wrap_content"
 74                  android:text="Good (18%)" />
 75
 76              <RadioButton
 77                  android:id="@+id/optionOkay"
 78                  android:layout_width="match_parent"
 79                  android:layout_height="wrap_content"
 80                  android:text="Okay (15%)" />
 81          </RadioGroup>
 82
 83          <Switch
 84              android:id="@+id/roundTip"
 85              android:layout_width="match_parent"
 86              android:layout_height="48dp"
 87              android:layout_marginStart="16dp"
 88              android:layout_marginEnd="16dp"
 89              android:checked="true"
 90              android:text="Round up tip?"
 91              android:textAlignment="textStart"
 92              app:layout_constraintEnd_toEndOf="parent"

 93  app:layout_constraintStart_toStartOf="@id/tipOption"

 94  app:layout_constraintTop_toBottomOf="@id/tipOption" />
 95
 96          <Button
 97              android:id="@+id/calculateButton"
 98              android:layout_width="0dp"
 99              android:layout_height="wrap_content"
100              android:layout_marginStart="16dp"
101              android:layout_marginEnd="16dp"
102              android:backgroundTint="#7223BD"
103              android:text="CALCULATE"
104              app:layout_constraintEnd_toEndOf="parent"
105              app:layout_constraintStart_toStartOf="parent"

106  app:layout_constraintTop_toBottomOf="@id/roundTip" />
107
108          <TextView
109              android:id="@+id/tipResult"
110              android:layout_width="wrap_content"
111              android:layout_height="wrap_content"
112              android:layout_marginEnd="16dp"
```

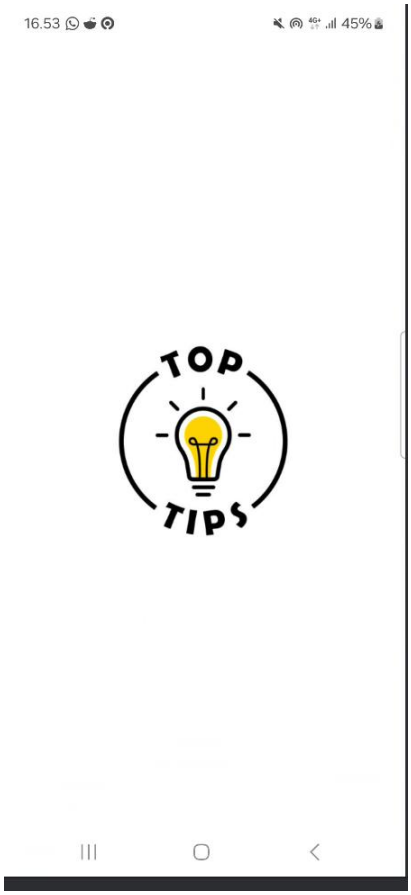
```
113             android:textSize="18sp"
114             android:text="Tip Amount"
115             android:textColor="#808080"
116             app:layout_constraintEnd_toEndOf="parent"

117 app:layout_constraintTop_toBottomOf="@+id/calculateButton"
    />
118
119     </androidx.constraintlayout.widget.ConstraintLayout>
120 </ScrollView>
```

4. SplashActivity.kt

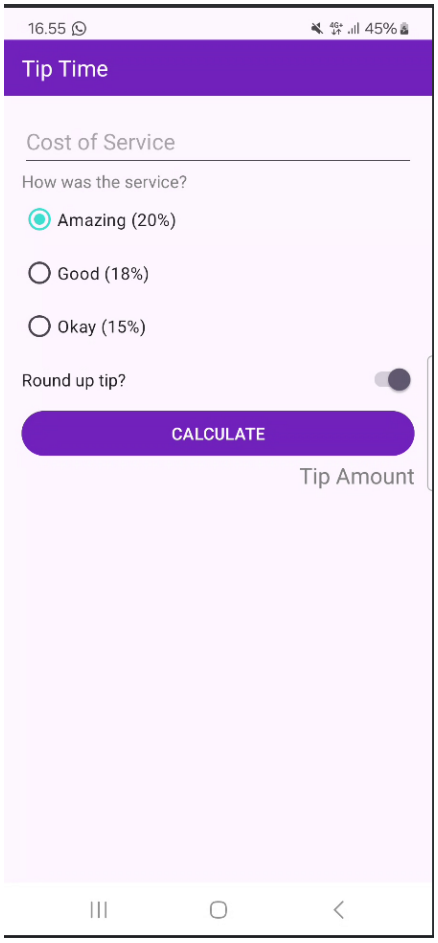
Tabel 4. Source Code SplashActivity.kt

```
 1 package com.example.tipcalculator.splash
 2
 3 import android.content.Intent
 4 import android.os.Bundle
 5 import androidx.appcompat.app.AppCompatActivity
 6 import androidx.core.splashscreen.SplashScreen.Companion.
   installSplashScreen
 7 import com.example.tipcalculator.MainActivity
 8
 9 class SplashActivity : AppCompatActivity() {
10     override fun onCreate(savedInstanceState: Bundle?) {
11         installSplashScreen()
12
13         super.onCreate(savedInstanceState)
14
15         startActivity(Intent(this,
   MainActivity::class.java))
16         finish()
17     }
18 }
```

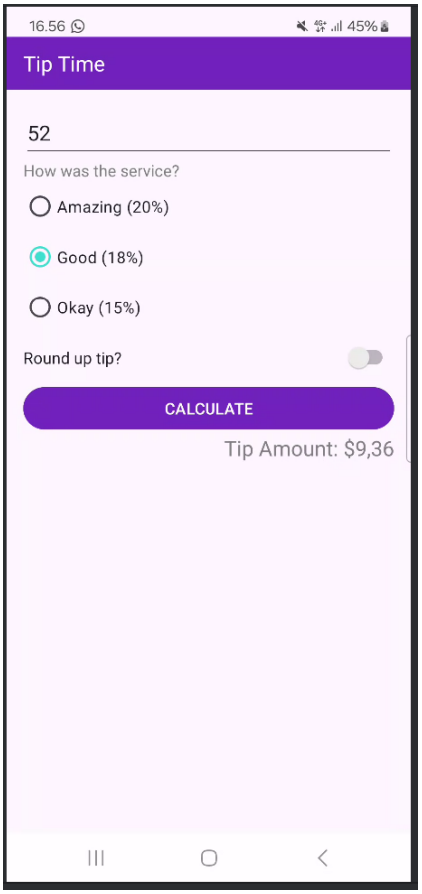
B. Input dan Output Program



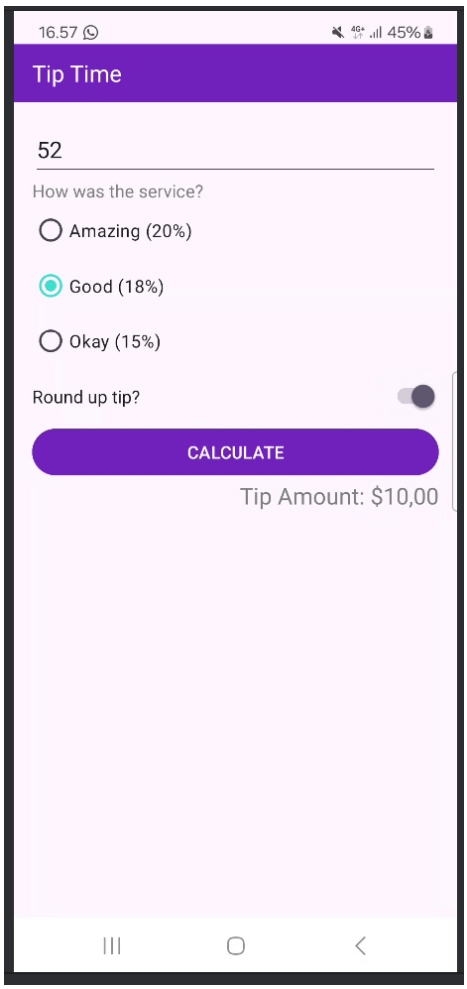
Gambar 3. Screenshot Tampilan Splash Screen



Gambar 4. Screenshot Tampilan Awal Aplikasi



Gambar 5. Screenshot Tampilan Aplikasi Setelah Dijalankan



Gambar 6. Screenshot Tampilan Aplikasi Setelah Dibulatkan

C. Pembahasan

1. MainActivity.kt:

- File ***MainActivity.kt*** adalah kelas utama yang menangani ampilan dan logika UI utama aplikasi Tip Calculator yang mengikuti arsitektur MVVM (Model-View-ViewModel).
- Baris [1] adalah deklarasi package menunjukkan bahwa file ini berada dalam package ***com.example.tipcalculator***.
- Baris [3] sampai [15] adalah mengimpor paket – paket yang dibutuhkan dalam kode ini, termasuk fitur UI, ***ViewModel***, ***data binding***, ***Context***, ***Bundle***, ***InputMethodManager***, ***TipViewModel***, dan ***enableEdgeToEdge()***.

- Baris [17], ***class MainActivity : AppCompatActivity() {*** adalah kelas utama yang menangani tampilan dan logika UI aplikasi dan mewarisi dari ***AppCompatActivity***.
- Baris [19] sampai baris [20] adalah deklarasi ***ViewBinding*** dengan binding untuk mengakses elemen UI di ***activity_main.xml*** secara langsung tanpa ***findViewById()***, dan juga deklarasi ***ViewModel*** untuk objek dari ***TipViewModel*** yang bertanggung jawab menyimpan state jumlah tip agar tidak hilang saat berubah orientasi (berubah portrait atau landscape).
- Baris [22] sampai baris [23] adalah ***onCreate()*** adalah fungsi yang dipanggil saat activity pertama kali atau titik masuk activity.
- Baris [24], ***enableEdgeToEdge()*** digunakan untuk mengaktifkan edge – to – edge layout yang berarti UI bisa memanjang sampai menyentuh bar navigasi.
- Baris [26] sampai baris [27] adalah menghubungkan layout XML ke kode Kotlin dengan menggunakan ***ViewBinding. inflate(layoutInflater)*** adalah metode statis yang digunakan untuk membuat instance dari kelas binding dan meng-inflate layout XML menjadi objek tampilan (View). ***setContentView(Binding.root)***, adalah merujuk pada root view dari layout XML (***activity_main.xml***).
- Baris [26] sampai baris [30], ***ViewCompat.setOnApplyWindowInsetsListener(binding.topBar) { v, insets -> ... }*** untuk menyesuaikan padding di bagian atas layout guna tidak tertutup oleh bar status.
- Baris [35] sampai baris [37], ***viewModel.tipAmount.observe(this, Observer { tip -> binding.tipResult.text = tip })*** adalah mengamati ***LiveData*** dari ***ViewModel*** dengan menggunakan ***observe***. Setiap kali ***tipAmount*** (hasil hitung tip) berubah di ***ViewModel***, nilai tersebut akan otomatis ditampilkan di ***TextView tipResult***.
- Baris [39] sampai baris [60], ***binding.calculateButton.setOnClickListener {…}*** adalah mengatur tindakan setelah tombol "***CALCULATE"*** ditekan. Variabel ***cost*** diinisialisasi dengan mengambil nilai input (***EditText***) ***costService***, dan mengubahnya menjadi Double. Jika input kosong atau

negatif, tampilkan pesan error menggunakan ***Toast*** dan kemudian keluar dari lambda dengan ***return@setOnClickListener***. Setelah itu, sembunyikan keyboard dengan ***hideKeyboard()*** dan kirim nilai input ke ViewModel dengan fungsi ***viewModel.setCostService(cost)***. Variabel ***tipPercent*** diinisialisasi dengan mengambil persentase tip berdasarkan radio button yang dipilih (***when (binding.tipOption.checkedRadioButtonId){..}***). Kemudian, kirim nilai persentase dan opsi pembulatan ke ViewModel menggunakan fungsi ***viewModel.setTipPercent(tipPercent)*** dan ***viewModel.setRoundUp(binding.roundTip.isChecked)***. Hitung jumlah tip dengan fungsi ***viewModel.calculateTip()***.

- Baris [62] sampai baris [70] adalah ketika tombol "Done" di keyboard ditekan, maka otomatis tombol "***CALCULATE***" diklik & keyboard disembunyikan menggunakan ***binding.costService.setOnEditorActionListener{…}***.
- Baris [73] sampai baris [76] adalah fungsi ***hideKeyboard()*** untuk menyembunyikan keyboard secara manual setelah input selesai.

2. TipViewModel.kt

- TipViewModel.kt adalah kelas yang bertanggung jawab untuk menyimpan dan mengelola data logika aplikasi (jumlah tip, cost, dll) agar UI tetap responsif saat orientasi berubah atau aktivitas diputar ulang.
- Baris [1], ***package com.example.tipcalculator.viewmodel*** adalah deklarasi package yang menunjukkan bahwa file ini berada di package ***viewmodel***.
- Baris [3] sampai baris [6] adalah mengimpor paket – paket yang digunakan dalam kode ini, yaitu ***ViewModel*** & ***LiveData*** dari Android Jetpack, dan ***ceil()*** dari ***Kotlin Math*** untuk pembulatan ke atas.
- Baris [8], ***class TipViewModel : ViewModel()*** adalah kelas ***viewModel*** yang bertanggung jawab untuk menyimpan dan mengelola data logika aplikasi (jumlah tip, cost, dll) agar UI tetap responsif.

- Baris [9] dan baris [10] adalah deklarasi variabel ***private _tipAmount*** dengan menggunakan ***MutableLiveData<String>()*** yang hanya bisa diubah dalam ***viewModel*** dan inisialisasi variabel ***tipAmount*** dengan mengambil nilai dari ***_tipAmount*** dan menggunakan ***LiveData<String> get()*** yang hanya bisa diobservasi oleh UI (***MainActivity***) tapi tidak bisa diubah dari luar.
- Baris [12] sampai baris [14] adalah inisialisasi variabel ***costService*** yang merupakan tempat biaya jasa yang diinput dengan nilai default 0.0, variabel ***tipPercent*** merupakan tempat persentase tip (default: 20%), dan variabel ***roundUp*** tempat menyimpan nilai tip dibulatkan ke atas atau tidak (default true).
- Baris [16] sampai baris [26] adalah fungsi setter yang digunakan untuk mengubah nilai variabel ***costService***, ***tipPercent***, dan ***roundUp***. Fungsi ***setCostService(cost: Double)*** adalah untuk mengatur nilai costService, dan mencegah nilai negatif (otomatis menjadi 0.0 jika < 0). Fungsi ***setTipPercent(percent: Double)*** adalah untuk mengatur persentase tip. Fungsi ***setRoundUp(round: Boolean)*** adalah untuk mengatur apakah hasil tip akan dibulatkan ke atas atau tidak.
- Baris [28] sampai baris [35] adalah fungsi ***calculateTip()*** digunakan untuk menghitung jumlah tip. Variabel ***tip*** diinisialisasi dengan nilai ***costService * tipPercent***. Jika dibulatkan (***if(roundUp)***), maka bulatkan tip ke atas dengan menggunakan ***ceil()***. Nilai perjehitungan akan di format dengan dua angka dibelakang koma dan disimpan pada ***_tipAmount***. Nilai akan otomatis dikirim ke ***observer*** di UI (***MainActivity***) menggunakan ***LiveData***.

3. Main_activity.xml

- File ***activity_main.xml*** adalah tampilan utama dari aplikasi Tip Calculator.
- Baris [1], terdapat versi XML dan encoding yang digunakan pada file ini.
- Baris [2] sampai baris [9] adalah root layout yang menggunakan ***scrollView*** agar konten dapat digulir (scroll) jika lebih panjang dari layar. ***fitsSystemWindows="true"*** digunakan agar konten tidak tertutup status bar atau navigation bar. Memiliki tinggi dan lebar menyesuaikan dengan parent (***match_parent***) serta id adalah ***scrollView***.
- Baris [11] sampai baris [15] adalah layout utama ***ConstraintLayout*** untuk semua elemen bisa diletakkan dengan fleksibel menggunakan **constraint** (misalnya di atas, di bawah, di kiri-kanan elemen lainnya). ***paddingBottom*** memberi jarak dari bagian bawah layout agar tidak terlalu mepet.
- Baris [17] sampai baris [31] adalah header ***TextView*** judul aplikasi yang menampilkan teks "***Tip Time***". Header memiliki warna background ungu: #7223BD dan memiliki id "***topBar***". Header ini memiliki posisi paling atas memenuhi lebar layar.
- Baris [33] sampai baris [43] adalah input biaya layanan dengan menggunakan ***EditText*** untuk memasukkan nominal biaya layanan. Menggunakan ***inputType = number|numberDecimal*** sehingga input hanya menerima angka dan desimal. Terdapat ***hint*** sebagai petunjuk input muncul saat belum ada teks, yaitu "***Cost of Service***". Input ini memiliki id "***costService***" dan berposisi di bawah header judul aplikasi.
- Baris [45] sampai baris [53] adalah pertanyaan "***How was the service?***" menggunakan ***TextView*** yang berposisi dibawah input biaya layanan. Teks ini sebagai judul sebelum pilihan radio button dan juga memiliki warna abu-abu #808080 untuk memberi kesan sekunder. ***TextView*** ini memiliki id "***serviceQuestion***".
- Baris [55] sampai baris [81] adalah kumpulan – kumpulan pilihan radio button menggunakan ***RadioGroup*** dan ***RadioButton***. ***RadioGroup*** dan ***RadioButton*** digunakan agar pengguna dapat memilih kualitas pelayanan yang dirasakannya. Terdapat 3 radio button dengan opsi "***Amazing (20%)***", opsi

"***Good (18%)***", dan opsi "***Okay (15%)***". ***RadioGroup*** berposisi dibawah pertanyaan tentang layanan dan memiliki id "***tipOption***". Ketiga radio button memiliki id, yaitu "***optionAmazing***", "***optionGood***", dan "***optionOkay***". Untuk nilai default dipilih pada opsi Amazing.

- Baris [83] sampai baris [94] adalah pilihan pembulatan ke atas menggunakan ***Switch***. ***Switch*** aktif secara default (***checked="true"***) dan berposisi di bawah radio button. ***Switch*** memiliki teks "***Round up tip***?" dan memiliki id "***roundTip***".
- Baris [96] sampai baris [106] adalah tombol hitung jumlah tip dengan menggunakan ***Button***. Berposisi di bawah ***Switch*** pembulatan dan memiliki warna yang sama dengan warna header (***topBar***). Tombol ini memiliki teks "***CALCULATE***" dan id "***calculateButton***".
- Baris [108] sampai baris [117] adalah menampilkan hasil perhitungan tip dengan meggunakan ***TextView***. Berposisi di bawah tombol "***CALCULATE***" dan memiliki warna teks abu – abu. Awalnya hanya teks biasa "***Tip Amount***", nanti diubah lewat ***ViewModel*** sehingga menampilkan hasil tipnya juga. ***TextView*** ini memiliki id "***tipResult***".

4. SplashActivity.kt

- File SplashActivity.kt adalah kelas yang digunakan untuk menampilkan splash screen aplikasi Tip Calculator.
- Baris [1], package ***com.example.tipcalculator.splash*** adalah deklarasi package yang menunjukkan bahwa file ini berada di folder ***splash*** dari package ***com.example.tipcalculator***.
- Baris [3] sampai baris [7] adalah mengimpor paket – paket yang digunakan dalam kelas ini, yaitu ***Intent*** untuk berpindah dari satu aktivitas ke aktivitas lain, ***installSplashScreen()*** adalah method dari Jetpack SplashScreen API, ***AppCompatActivity***, dan ***MainActivity*** adalah activity utama yang akan dibuka setelah splash selesai.
- Baris [9], ***class SplashActivity : AppCompatActivity() {*** adalah activity pertama yang dijalankan ketika aplikasi dibuka untuk menampilkan ***splash screen***.
- Baris [10] sampai baris [17] adalah fungsi ***onCreate()*** untuk titik masuk activity splash screen. Fungsi ***installSplashScreen()*** adalah untuk memasang *Jetpack SplashScreen* yang muncul secara otomatis saat app pertama dibuka. Kemudian, ***super.onCreate(savedInstanceState)*** adalah memanggil method bawaan dari ***AppCompatActivity*** untuk inisialisasi. Setelab itu, ***startActivity(Intent(this, MainActivity::class.java))*** adalah membuka ***MainActivity***, yaitu halaman utama aplikasi. Fungsi ***finish()*** adalah untuk mengakhiri ***SplashActivity*** agar tidak bisa dikembali lewat tombol back.

TAUTAN GIT

Berikut adalah tautan untuk source code yang telah dibuat.

https://github.com/Kylorts/Pemrograman-mobile/tree/main/Modul%202