

**LAPORAN PRAKTIKUM  
PEMROGRAMAN MOBILE  
MODUL 2**



**ANDROID LAYOUT WITH COMPOSE**

**Oleh:**

**Avantio Fierza Patria NIM. 2310817310001**

**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 With Compose ini disusun sebagai syarat lulus mata kuliah Praktikum Pemrograman Mobile. Laporan Praktikum ini dikerjakan oleh:

Nama Praktikan : Avantio Fierza Patria  
NIM : 2310817310001

Menyetujui,  
Asisten Praktikum

Mengetahui,  
Dosen Penanggung Jawab Praktikum

Muhammad Raka Azwar  
NIM. 2210817210012

Andreyan Rizky Baskara, S.Kom.,  
M.Kom.  
NIP. 19930703 201903 01 011

## DAFTAR ISI

LEMBAR PENGESAHAN.....	i
DAFTAR ISI .....	ii
DAFTAR GAMBAR.....	iii
DAFTAR TABEL .....	iv
SOAL 1 .....	1
A. Source Code menggunakan Jetpack Compose .....	2
B. Source Code menggunakan XML .....	5
C. Output Program .....	8
D. Pembahasan .....	10
E. Tautan GIT .....	13

## DAFTAR GAMBAR

Gambar 1. 1 Gambar Tampilan awal aplikasi .....	1
Gambar 1. 2 Tampilan Pilihan Persentase Tip .....	2
Gambar 1. 3 Tampilan Aplikasi Setelah Dijalankan .....	2
Gambar 1. 4 Screenshot Tampilan Awal.....	8
Gambar 1. 5 Screenshot Tampilan Pilihan Tip.....	9
Gambar 1. 6 Screenshot Aplikasi Setelah Dijalankan.....	10

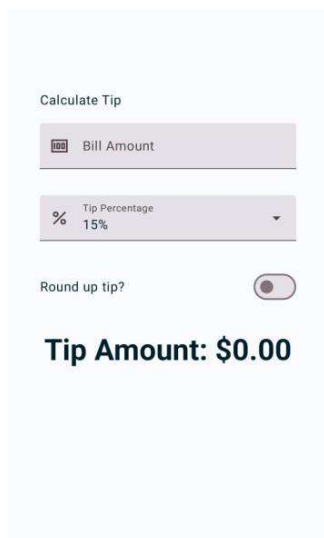
## **DAFTAR TABEL**

Tabel 1. 1 Source Code Jawaban Soal 1 MainActivity.kt Jetpack Compose .....	5
Tabel 1. 2 Source Code Jawaban Soal 1 MainActivity.kt XML .....	6
Tabel 1. 3 Source Code Jawaban Soal 1 activity_maint .XML.....	7

## SOAL 1

### Soal Praktikum:

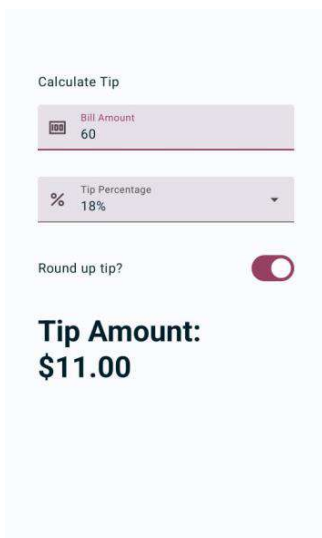
1. Buatlah sebuah aplikasi kalkulator tip menggunakan XML dan Jetpack Compose yang dirancang untuk membantu pengguna menghitung tip yang sesuai berdasarkan total biaya layanan yang mereka terima. Fitur-fitur yang diharapkan dalam aplikasi ini mencakup:
  - A. Input biaya layanan: Pengguna dapat memasukkan total biaya layanan yang diterima dalam bentuk nominal.
  - B. Pilihan persentase tip: Pengguna dapat memilih persentase tip yang diinginkan.
  - C. Pengaturan pembulatan tip: Pengguna dapat memilih untuk membulatkan tip ke angka yang lebih tinggi.
  - D. Tampilan hasil: Aplikasi akan menampilkan jumlah tip yang harus dibayar secara langsung setelah pengguna memberikan input.
2. Jelaskan perbedaan dari implementasi XML dan Jetpack Compose beserta kelebihan dan kekurangan dari masing-masing implementasi.



Gambar 1. 1 Gambar Tampilan awal aplikasi



Gambar 1. 2 Tampilan Pilihan Persentase Tip



Gambar 1. 3 Tampilan Aplikasi Setelah Dijalankan

## A. Source Code menggunakan Jetpack Compose

### MainActivity.kt

```

1 package com.example.kalkulator
2
3 import android.os.Bundle
4 import androidx.activity.ComponentActivity
5 import androidx.activity.compose.setContent
6 import androidx.compose.foundation.layout.*
7 import androidx.compose.foundation.text.KeyboardOptions
8 import androidx.compose.material3.*
9 import androidx.compose.runtime.*
10 import androidx.compose.ui.Alignment

```

```

11 import androidx.compose.ui.Modifier
12 import androidx.compose.ui.text.input.KeyboardType
13 import androidx.compose.ui.unit.dp
14 import com.example.kalkulator.ui.theme.KalkulatorTheme
15 import kotlin.math.ceil
16
17 class MainActivity : ComponentActivity() {
18     override fun onCreate(savedInstanceState: Bundle?) {
19         super.onCreate(savedInstanceState)
20         setContent {
21             KalkulatorTheme {
22                 Surface(
23                     modifier = Modifier.fillMaxSize(),
24                     color
25 MaterialTheme.colorScheme.background
26                 ) {
27                     TipCalculatorCompose()
28                 }
29             }
30         }
31     }
32 }
33
34 @OptIn(ExperimentalMaterial3Api::class)
35 @Composable
36 fun TipCalculatorCompose() {
37     var cost by remember { mutableStateOf("") }
38     var tipPercent by remember { mutableStateOf(15) }
39     var roundUp by remember { mutableStateOf(false) }
40     val tip = calculateTip(cost.toDoubleOrNull() ?: 0.0,
41 tipPercent, roundUp)
42
43     Column(
44         modifier = Modifier
45             .padding(16.dp)
46             .fillMaxWidth()
47     ) {
48         Text("Kalkulator Tip", style
49 MaterialTheme.typography.headlineSmall)
50
51         Spacer(modifier = Modifier.height(16.dp))
52
53         OutlinedTextField(
54             value = cost,
55             onValueChange = { cost = it },
56             label = { Text("Biaya Layanan") },
57             keyboardOptions = KeyboardOptions(keyboardType =
58 KeyboardType.Number),
59             modifier = Modifier.fillMaxWidth()
60         )
61
62         Spacer(modifier = Modifier.height(16.dp))
63
64         var expanded by remember { mutableStateOf(false) }

```



```

65         val tipOptions = listOf(10, 15, 20, 25)
66
67         ExposedDropDownMenuBox(
68             expanded = expanded,
69             onExpandedChange = { expanded = !expanded }
70         ) {
71             OutlinedTextField(
72                 value = "$tipPercent%",
73                 onValueChange = {},
74                 readOnly = true,
75                 label = { Text("Persentase Tip") },
76                 trailingIcon = {
77 ExposedDropDownMenuDefaults.TrailingIcon(expanded) },
78                 modifier =
79 Modifier.menuAnchor().fillMaxWidth()
80             )
81
82             ExposedDropDownMenu(
83                 expanded = expanded,
84                 onDismissRequest = { expanded = false }
85             ) {
86                 tipOptions.forEach { percent ->
87                     DropdownMenuItem(
88                         text = { Text("$percent%") },
89                         onClick = {
90                             tipPercent = percent
91                             expanded = false
92                         }
93                     )
94                 }
95             }
96         }
97
98         Spacer(modifier = Modifier.height(16.dp))
99
100        Row(
101            verticalAlignment = Alignment.CenterVertically,
102            modifier = Modifier.fillMaxWidth()
103        ) {
104            Text("Round up tip?", modifier =
105 Modifier.weight(1f))
106            Switch(
107                checked = roundUp,
108                onCheckedChange = { roundUp = it },
109                colors = SwitchDefaults.colors(
110                    checkedThumbColor =
111 MaterialTheme.colorScheme.primary
112                )
113            )
114        }
115
116        Spacer(modifier = Modifier.height(24.dp))
117
118        Text(

```

119	text = "Jumlah Tip: Rp%.2f".format(tip),
120	style = MaterialTheme.typography.titleMedium
121	)
122	}
123	}
124	fun calculateTip(amount: Double, tipPercent: Int, roundUp:
125	Boolean): Double {
126	var tip = amount * tipPercent / 100
127	if (roundUp) tip = ceil(tip)
128	return tip
	}

*Tabel 1. 1 Source Code Jawaban Soal 1 MainActivity.kt Jetpack Compose*

## **B. Source Code menggunakan XML**

### **MainActivity.kt**

1	package com.example.diceroller2
2	
3	import android.os.Bundle
4	import android.widget.Button
5	import android.widget.ImageView
6	import android.widget.TextView
7	import androidx.appcompat.app.AppCompatActivity
8	import kotlin.random.Random
9	
10	class MainActivity : AppCompatActivity() {
11	override fun onCreate(savedInstanceState: Bundle?) {
12	super.onCreate(savedInstanceState)
13	setContentView(R.layout.activity_main)
14	
15	// Ambil referensi dari UI
16	val dice1Image: ImageView =
17	findViewById(R.id.dice1Image)
18	val dice2Image: ImageView =
19	findViewById(R.id.dice2Image)
20	val btnRoll: Button = findViewById(R.id.btnRoll)
21	val tvMessage: TextView = findViewById(R.id.tvMessage)
22	
23	btnRoll.setOnClickListener {
24	val dice1Value = Random.nextInt(1, 7)
25	val dice2Value = Random.nextInt(1, 7)
26	
27	// Update gambar dadu
28	
29	dice1Image.setImageResource(getDiceImage(dice1Value))
30	

```

31
32 dice2Image.setImageResource(getDiceImage(dice2Value))
33
34         // Update pesan berdasarkan hasil
35         tvMessage.text = if (dice1Value == dice2Value) {
36             "🎉 Selamat, anda dapat dadu double! 🎉"
37         } else {
38             "😞 Anda belum beruntung!"
39         }
40     }
41 }
42
43 private fun getDiceImage(value: Int): Int {
44     return when (value) {
45         1 -> R.drawable.dice_1
46         2 -> R.drawable.dice_2
47         3 -> R.drawable.dice_3
48         4 -> R.drawable.dice_4
49         5 -> R.drawable.dice_5
50         6 -> R.drawable.dice_6
51         else -> R.drawable.dice_1
52     }
53 }
54 }

```

Tabel 1. 2 Source Code Jawaban Soal 1 MainActivity.kt XML

### activity\_main.xml

```

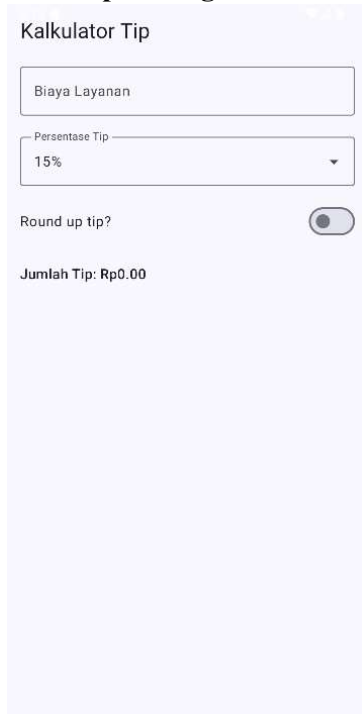
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout
3  xmlns:android="http://schemas.android.com/apk/res/android"
4      android:orientation="vertical"
5      android:padding="16dp"
6      android:layout_width="match_parent"
7      android:layout_height="match_parent">
8
9      <TextView
10         android:text="Kalkulator Tip"
11         android:textSize="24sp"
12         android:textStyle="bold"
13         android:layout_width="wrap_content"
14         android:layout_height="wrap_content" />
15
16         <EditText
17             android:id="@+id/costInput"
18             android:hint="Biaya Layanan"
19             android:inputType="numberDecimal"
20             android:layout_width="match_parent"
21             android:layout_height="wrap_content"
22             android:layout_marginTop="16dp"/>
23
24         <Spinner
25             android:id="@+id/tipSpinner"
26             android:layout_width="match_parent"

```

27	android:layout_height="wrap_content"
28	android:layout_marginTop="16dp" />
29	
30	<LinearLayout
31	android:orientation="horizontal"
32	android:layout_width="match_parent"
33	android:layout_height="wrap_content"
34	android:layout_marginTop="16dp"
35	android:gravity="center_vertical">
36	
37	<TextView
38	android:text="Round up tip?"
39	android:layout_width="0dp"
40	android:layout_weight="1"
41	android:layout_height="wrap_content"/>
42	
43	<Switch
44	android:id="@+id/roundUpSwitch"
45	android:layout_width="wrap_content"
46	android:layout_height="wrap_content"
47	android:thumbTint="@android:color/white"/>
48	</LinearLayout>
49	
50	<TextView
51	android:id="@+id/tipResult"
52	android:text="Jumlah Tip: Rp0.00"
53	android:textSize="18sp"
54	android:textStyle="bold"
55	android:layout_marginTop="24dp"
56	android:layout_width="wrap_content"
57	android:layout_height="wrap_content"/>
	</LinearLayout>

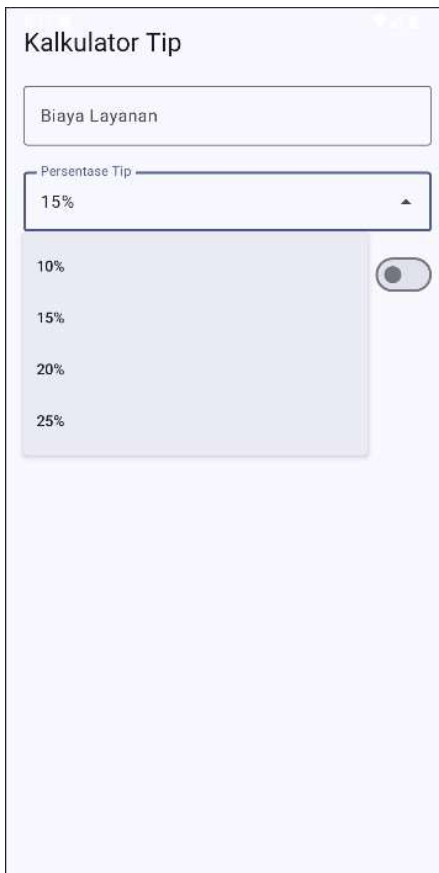
Tabel 1. 3 Source Code Jawaban Soal 1 activity\_maint .XML

### C. Output Program

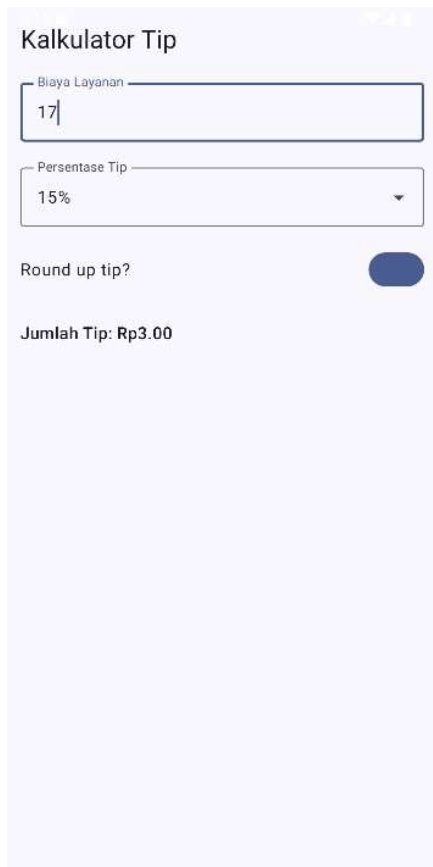


The screenshot shows a mobile application titled "Kalkulator Tip". It features a light purple background. At the top, there is a status bar with the time "12:41" and battery level "93%". Below the title, there is a text input field labeled "Biaya Layanan". Underneath that is a dropdown menu labeled "Persentase Tip" with "15%" selected. To the right of the dropdown is a toggle switch labeled "Round up tip?". At the bottom, the text "Jumlah Tip: Rp0.00" is displayed.

*Gambar 1. 4 Screenshot Tampilan Awal*



*Gambar 1. 5 Screenshot Tampilan Pilihan Tip*



Gambar 1. 6 Screenshot Aplikasi Setelah Dijalankan

## D. Pembahasan

### MainActivity.kt Jetpack Compose:

#### 1. MainActivity

Di *MainActivity*, kita menggunakan komponen *ComponentActivity* karena Compose tidak lagi menggunakan layout XML. Di dalam fungsi *onCreate*, kita langsung menetapkan konten UI menggunakan fungsi *setContent*.

Seluruh tampilan aplikasi dibungkus dalam *Surface*, dan di dalamnya kita memanggil fungsi *TipCalculatorCompose()*, yaitu UI utama kita.

## 2. Fungsi *TipCalculatorCompose()*

Fungsi ini berisi seluruh antarmuka pengguna. Kita mendefinisikan beberapa state:

- *cost* menyimpan input biaya layanan dari pengguna.
- *tipPercent* menyimpan persentase tip yang dipilih.
- *roundUp* menyimpan status switch, apakah pengguna ingin membulatkan tip ke atas.

State ini memungkinkan UI untuk secara otomatis merespons perubahan nilai yang dilakukan pengguna.

## 3. Input Biaya Layanan

Bagian ini menggunakan *OutlinedTextField*, yaitu versi *Compose* dari input teks yang familiar di *Material Design*. Ketika pengguna mengetik, nilai *cost* akan berubah, dan tip akan dihitung ulang.

## 4. Dropdown Pilihan Tip (*Spinner*)

Kita menggunakan *ExposedDropdownMenuBox*, yang fungsinya mirip seperti *Spinner* di XML. Pengguna bisa mengetuknya untuk memilih persentase tip (10%, 15%, 20%, atau 25%). Saat dipilih, nilai *tipPercent* diperbarui.

## 5. Switch "Round Up Tip"

Untuk opsi membulatkan ke atas, kita menggunakan *Switch()* yang diletakkan dalam *Row*, berdampingan dengan teks deskriptif. Jika pengguna mengaktifkan switch ini, nilai tip akan dibulatkan ke atas menggunakan *ceil()*.

## 6. Tampilan Hasil Tip

Di bagian bawah, kita menampilkan hasil perhitungan tip dalam format "Rp...", dan tampilannya akan berubah secara otomatis ketika input, persentase, atau switch diubah.

## 7. Fungsi *calculateTip()*

Fungsi ini menerima tiga parameter: jumlah biaya, persentase tip, dan status pembulatan. Fungsi ini melakukan perhitungan tip, dan jika *roundUp* bernilai *true*, maka nilai tip akan dibulatkan ke atas.

## MainActivity.kt XML:

### 1. activity\_main.xml



File ini berisi antarmuka pengguna menggunakan pendekatan tradisional. Komponen-komponen utama antara lain:

- EditText: Tempat pengguna memasukkan biaya layanan.
- Spinner: Digunakan untuk memilih persentase tip.
- Switch: Digunakan untuk memilih apakah ingin membulatkan ke atas.
- TextView: Menampilkan hasil tip.

View: Sebagai pemisah visual (divider) di bawah spinner agar tampilan lebih rapi.

## **2. MainActivity.kt**

Di sini kita menggunakan pendekatan View-based (lama). Di dalam onCreate, kita menghubungkan semua komponen dari XML ke Kotlin menggunakan findViewById.

## **3. Inisialisasi Spinner**

Kita menyiapkan daftar persentase tip dalam bentuk list (tipOptions), lalu memasangnya ke spinner menggunakan ArrayAdapter. Dengan begitu, spinner akan menampilkan pilihan seperti 10%, 15%, dst.

## **4. Fungsi updateTip()**

Fungsi ini adalah inti dari logika kalkulasi. Ia membaca nilai biaya, persentase tip yang dipilih dari spinner, dan status dari switch. Kemudian ia menghitung tip dan memperbarui teks di TextView.

## **5. Event Listener**

Agar aplikasi merespons secara real-time, kita menambahkan listener:

- Pada EditText, kita menggunakan TextWatcher agar setiap kali pengguna mengetik, tip langsung dihitung ulang.
- Pada Switch, kita mendengarkan perubahan status.
- Pada Spinner, kita merespons ketika pengguna memilih persentase baru.

Semua listener tersebut akan memanggil fungsi updateTip() agar tampilan hasil selalu sesuai input.

Perbedaan dari implementasi XML dan Jetpack Compose beserta kelebihan dan kekurangan dari masing-masing implementasi

### **XML + View System**

XML menggunakan pendekatan deklaratif statis di mana elemen UI ditulis dalam file XML terpisah dan dikaitkan dengan logika di file Kotlin/Java melalui kode imperative. Ini adalah pendekatan konvensional dan telah digunakan sejak awal Android.

#### Kelebihan:

- Sudah matang dan stabil, digunakan sejak Android awal.
- Lebih familiar bagi developer lama.
- Banyak dokumentasi, tutorial, dan komunitas luas.
- Terpisah antara desain dan logika (bisa didesain pakai layout editor).

#### Kekurangan:

- Perlu findViewById() atau ViewBinding → verbose.
- Sulit untuk membuat UI yang dinamis atau reaktif.
- Penambahan fitur atau animasi lebih kompleks.
- Struktur hierarki UI bisa rumit dan lambat saat render.

#### Jetpack Compose

Jetpack Compose adalah framework UI declarative modern berbasis Kotlin, yang memungkinkan kita membangun antarmuka UI secara langsung di dalam file Kotlin, menyatukan logika dan tampilan dalam satu tempat. Komponen UI di Compose bisa langsung merefleksikan perubahan data secara otomatis (reactive).

#### Kelebihan:

- Lebih modern, efisien, dan ringkas.
- UI dan logika digabung → mudah di-maintain.
- Reaktif → otomatis merender ulang saat state berubah.
- Mendukung tema dinamis, animasi, dan custom UI dengan lebih mudah.
- Cocok untuk arsitektur modern (MVVM, Clean Architecture).

#### Kekurangan:

- Butuh penyesuaian untuk developer lama.
- Beberapa komponen UI belum sekomplit XML (tapi terus berkembang).
- Dokumentasi dan tooling masih dalam tahap berkembang.
- Kurang cocok untuk proyek Android lama (pre-Jetpack).

#### **E. Tautan GIT**

**<https://github.com/gr1ff0m/Pemrograman-Mobile-Praktikum>**