

**LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 2**



ANDROID LAYOUT

Oleh:

Dina Izzati Elfadheya NIM. 2310817120001

**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 : Dina Izzati Elfadheya

NIM : 2310817120001

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	8
B. Output Program.....	15
C. Pembahasan.....	19
TAUTAN GIT	25

DAFTAR GAMBAR

Gambar 1 Contoh Tampilan Awal Aplikasi.....	6
Gambar 2 Contoh Tampilan Aplikasi Setelah Dijalankan	7
Gambar 3 Screenshot Output Tampilan Splash Screen.....	15
Gambar 4 Screenshot Output Tampilan Awal	16
Gambar 5 Screenshot Output Tampilan Aplikasi Setelah dijalankan	17
Gambar 6 Screenshot Output Tampilan Aplikasi Jika Input Salah	18

DAFTAR TABEL

Tabel 1 Source Code Jawaban MainActivity.kt	9
Tabel 2 Source Code Jawaban Activity_Main.XML	12
Tabel 3 Source Code Jawaban Button_Color.xml.....	12
Tabel 4 Source Code Jawaban ViewModel.kt.....	13
Tabel 5 Source Code Jawaban values\themes.xml.....	13
Tabel 6 Source Code Jawaban AndroidManifest.xml	14

SOAL 1

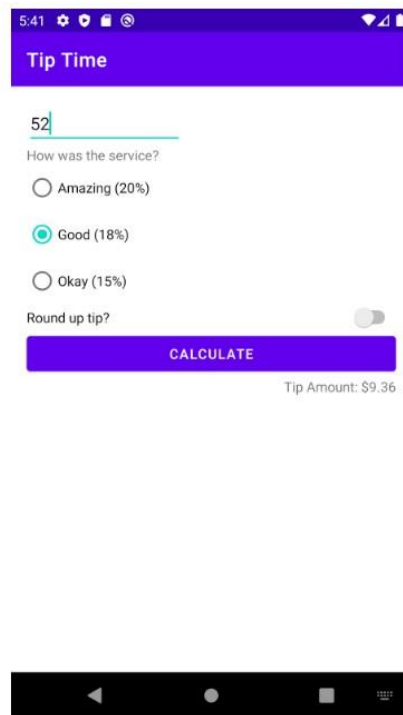
Buatlah sebuah aplikasi kalkulator tip yang dirancang untuk membantu pengguna menghitung tip yang sesuai berdasarkan total biaya layanan yang mereka terima.

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.

a

Gambar 1 Contoh Tampilan Awal Aplikasi



*Gambar 2 Contoh Tampilan Aplikasi
Setelah Dijalankan*

A. Source Code

1. MainActivity.kt

```
1 package com.example.tiptime
2
3 import android.os.Bundle
4 import android.util.Log
5 import android.widget.*
6 import androidx.appcompat.app.AppCompatActivity
7 import
8     androidx.core.splashscreen.SplashScreen.Companion.installSplashScreen
9 import java.text.NumberFormat
10 import kotlin.math.ceil
11 import androidx.activity.viewModels
12 import com.example.tiptime.TipViewModel
13
14 class MainActivity : AppCompatActivity() {
15     private val viewModel: TipViewModel by
16     viewModels()
17
18     override fun onCreate(savedInstanceState:
19     Bundle?) {
20         super.onCreate(savedInstanceState)
21         Thread.sleep(3000)
22         installSplashScreen()
23         setContentView(R.layout.activity_main)
24
25         val costOfServiceEditText =
26         findViewById<EditText>(R.id.cost_of_service_edit
27         _text)
28         val tipOptions =
29         findViewById<RadioGroup>(R.id.tip_options)
30         val roundUpSwitch =
31         findViewById<Switch>(R.id.round_up_switch)
32         val calculateButton =
33         findViewById<Button>(R.id.calculate_button)
34         val tipResult =
35         findViewById<TextView>(R.id.tip_result)
36
37         if (viewModel.tipResult.isNotEmpty()) {
38             tipResult.text = "Tip Amount:
39             ${viewModel.tipResult}"
40         }
41
42         calculateButton.setOnClickListener {
43             val costInput =
44             costOfServiceEditText.text.toString()
```



```

35         val cost =
36         costInput.toDoubleOrNull()
37
38         if (costInput.isEmpty()) {
39             tipResult.text = ""
40             Toast.makeText(this, "Masukkan
biaya layanan!", Toast.LENGTH_SHORT).show()
41             return@setOnClickListener
42         }
43
44         if (cost == null || cost <= 0.0) {
45             tipResult.text = ""
46             Toast.makeText(
47                 this,
48                 "Input tidak valid. Masukkan
angka yang benar!",
49                 Toast.LENGTH_SHORT
50             ).show()
51             return@setOnClickListener
52         }
53
54         val tipPercentage = when
55         (tipOptions.checkedRadioButtonId) {
56             R.id.amazing_option -> 0.20
57             R.id.good_option -> 0.18
58             R.id.okay_option -> 0.15
59             else -> {
60                 Toast.makeText(
61                     this,
62                     "Pilih kualitas layanan
terlebih dahulu!",
63                     Toast.LENGTH_SHORT
64                 ).show()
65                 return@setOnClickListener
66             }
67         }
68         val result =
69         viewModel.calculateTip(cost, tipPercentage,
70         roundUpSwitch.isChecked)
71         tipResult.text = "Tip Amount:
$result"
72
73         Log.d("TipTime", "Tip dihitung:
$result")
74     }
75 }

```

Tabel 1 Source Code Jawaban MainActivity.kt

2. Activity_main.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:id="@+id/mainLayout"
6      android:layout_width="match_parent"
7      android:layout_height="match_parent"
8      android:orientation="vertical"
9      android:padding="16dp"
10     tools:context=".MainActivity">
11     <TextView
12         android:id="@+id/toolBarText"
13         android:layout_width="match_parent"
14         android:layout_height="wrap_content"
15         android:paddingBottom="8dp"
16         android:text="Tip Time"
17         android:textSize="18sp"
18
19     android:background="@android:color/white"
20     android:backgroundTint="#ebccff" />
21     <EditText
22         android:id="@+id/cost_of_service_edit_text"
23         android:layout_width="match_parent"
24         android:layout_height="wrap_content"
25         android:inputType="numberDecimal"
26         android:hint="Cost of Service"
27         android:padding="8dp"
28
29     android:background="@android:drawable/edit_text"
30     />
31     <TextView
32         android:id="@+id/question"
33         android:layout_width="wrap_content"
34         android:layout_height="wrap_content"
35         android:text="How was the service?"
36         android:layout_marginTop="16dp" />
37     <RadioGroup
38         android:id="@+id/tip_options"
39         android:layout_width="wrap_content"
40         android:layout_height="wrap_content">
41
```

```

42         <RadioButton
43             android:id="@+id/amazing_option"
44             android:layout_width="wrap_content"
45             android:layout_height="wrap_content"
46             android:text="Amazing (20%) " />
47
48         <RadioButton
49             android:id="@+id/good_option"
50             android:layout_width="wrap_content"
51             android:layout_height="wrap_content"
52             android:text="Good (18%) " />
53
54         <RadioButton
55             android:id="@+id/okay_option"
56             android:layout_width="wrap_content"
57             android:layout_height="wrap_content"
58             android:text="Okay (15%) " />
59     </RadioGroup>
60
61     <LinearLayout
62         android:layout_width="wrap_content"
63         android:layout_height="wrap_content"
64         android:orientation="horizontal"
65         android:layout_marginTop="16dp"
66         android:gravity="center_vertical">
67
68         <TextView
69             android:layout_width="wrap_content"
70             android:layout_height="wrap_content"
71             android:text="Round up tip?" />
72
73         <Switch
74             android:id="@+id/round_up_switch"
75             android:layout_width="wrap_content"
76             android:layout_height="wrap_content"
77             android:layout_marginStart="240dp"
78     />
79 </LinearLayout>
80
81     <android.widget.Button
82         android:id="@+id/calculate_button"
83         android:layout_width="match_parent"
84         android:layout_height="wrap_content"
85         android:text="CALCULATE"
86         android:layout_marginTop="24dp"
87         android:background="@drawable/button_color"
88         android:textColor="@android:color/white"
89     />

```

88	
89	<TextView
90	android:id="@+id/tip_result"
91	android:layout_width="wrap_content"
92	android:layout_height="wrap_content"
93	android:layout_marginTop="16dp"
94	android:text="Tip Amount:"
95	android:textSize="16sp"
96	android:layout_marginStart="240dp" />
97	</LinearLayout>

Tabel 2 Source Code Jawaban Activity_Main.XML

3. Button_Color.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="#7d499d" />
6	<corners android:radius="12dp" />
7	</shape>
8	</item>
9	
10	<item>
11	<shape android:shape="rectangle">
12	<solid android:color="#d395f9" />
13	<corners android:radius="12dp" />
14	</shape>
15	</item>
16	</selector>

Tabel 3 Source Code Jawaban Button_Color.xml

4. ViewModel.kt

1	package com.example.tiptime
2	
3	import androidx.lifecycle.ViewModel
4	import java.text.NumberFormat
5	import kotlin.math.ceil
6	
7	class TipViewModel : ViewModel() {
8	var tipResult: String = ""
9	
10	fun calculateTip(cost: Double, tipPercent:
	Double, roundUp: Boolean): String {
11	var tip = cost * tipPercent

12	if (roundUp) tip = ceil(tip)
13	
14	tipResult =
15	NumberFormat.getCurrencyInstance().format(tip)
16	return tipResult
17	}

Tabel 4 Source Code Jawaban ViewModel.kt

5. Values/themes.xml

1	<resources
	xmlns:tools="http://schemas.android.com/tools">
2	<!-- Base application theme. -->
3	<style name="Base.Theme.TipTime"
	parent="Theme.Material3.DayNight.NoActionBar">
4	<!-- Customize your light theme here. --
	>
5	<item name=
	"colorPrimary">@color/white</item>
6	<item name=
	"colorPrimaryVariant">@color/white</item>
7	<item name=
	"colorOnPrimary">@color/white</item>
8	<item name=
	"android:statusBarColor">?attr/colorPrimaryVaria
	nt</item>
9	<!-- <item
	name="colorPrimary">@color/my_light_primary</ite
	m> -->
10	</style>
11	<style name="Theme.TipTime"
	parent="Base.Theme.TipTime" />
12	
13	<style name="Theme.App.SplashScreen"
	parent="Theme.SplashScreen">
14	<item
	name="windowSplashScreenBackground">@color/white
	</item>
15	<item
	name="windowSplashScreenAnimatedIcon">@drawable/
	android</item>
16	<item
	name="postSplashScreenTheme">@style/Base.Theme.T
	ipTime</item>
17	</style>
18	</resources>

Tabel 5 Source Code Jawaban values/themes.xml

6. AndroidManifest.xml

1	<?xml version="1.0" encoding="utf-8"?>
2	<manifest
	xmlns:android="http://schemas.android.com/apk/re
	s/android"
3	xmlns:tools="http://schemas.android.com/tools">
4	
5	<application
6	android:allowBackup="true"
7	android:dataExtractionRules="@xml/data_extractio
	n_rules"
8	android:fullBackupContent="@xml/backup_rules"
9	android:icon="@mipmap/ic_launcher"
10	android:label="@string/app_name"
11	android:roundIcon="@mipmap/ic_launcher_round"
12	android:supportsRtl="true"
13	android:theme="@style/Theme.TipTime"
14	tools:targetApi="31">
15	<activity
16	android:name=".ViewModel"
17	android:exported="false" />
18	<activity
19	android:name=".MainActivity"
20	android:exported="true"
21	android:theme="@style/Theme.App.SplashScreen">
22	<intent-filter>
23	<action
	android:name="android.intent.action.MAIN" />
24	<category
	android:name="android.intent.category.LAUNCHER"
	/>
25	</intent-filter>
26	</activity>
27	</application>
28	
29	</manifest>

Tabel 6 Source Code Jawaban AndroidManifest.xml

B. Output Program

12:47  



*Gambar 3 Screenshot Output Tampilan
Splash Screen*

Tip Time

Cost of Service

How was the service?

☒ Amazing (20%)

☐ Good (18%)

☐ Okay (15%)

Round up tip?

CALCULATE

Tip Amount:

Gambar 4 Screenshot Output Tampilan Awal

Tip Time

52

How was the service?

☐ Amazing (20%)

☒ Good (18%)

☐ Okay (15%)

Round up tip? ☐

CALCULATE

Tip Amount: \$9.36

Gambar 5 Screenshot Output Tampilan Aplikasi Setelah dijalankan

Tip Time

0

How was the service?


☐ Amazing (20%)

☐ Good (18%)

☐ Okay (15%)

Round up tip? ☐

CALCULATE

 Input tidak valid. Masukkan angka yang benar!

Gambar 6 Screenshot Output Tampilan Aplikasi Jika Input Salah

C. Pembahasan

1. MainActivity.kt

Pada baris 1 `Package` berfungsi sebagai nama unik untuk mengelompokkan kelas-kelas dalam proyek Android. Baris 3–11 merupakan bagian impor dari berbagai kelas penting yang dibutuhkan supaya aplikasi bisa menjalankan fitur UI modern, menyimpan data lewat `ViewModel`, dan menangani input/output dengan baik di Android. Baris 14 kelas `MainActivity` adalah tempat utama aplikasi dijalankan saat pengguna membukanya. Baris 15 membuat properti `viewModel` dengan bantuan `byviewModels()` yang otomatis membuat objek `TipViewModel`. Ini berguna supaya data tetap tersimpan meskipun layar diputar atau konfigurasi berubah. Baris 17 `onCreate()` adalah fungsi yang dijalankan saat pertama kali aktivitas dibuat. Baris 18 `Thread.sleep(3000)` digunakan untuk memberi jeda selama 3 detik agar splash screen terlihat lebih lama. Baris 19 memanggil `installSplashScreen()` untuk menampilkan splash screen saat aplikasi dibuka. Baris 20 `setContentView(R.layout.activity_main)` berfungsi untuk menampilkan tampilan layar dari file XML bernama `activity_main.xml`. Baris 22–27 menghubungkan komponen UI di XML ke variabel Kotlin dengan `findViewById`, seperti `EditText`, `RadioGroup`, `Switch`, dan `Button`. Baris 29–31 mengecek apakah sebelumnya sudah ada hasil `tipResult` yang disimpan di `ViewModel`. Kalau ada, hasilnya langsung ditampilkan di `TextView`. Baris 33 menambahkan aksi ketika tombol `calculateButton` ditekan dengan `setOnClickListener`. Baris 34 mengambil input dari `EditText` dan menyimpannya sebagai teks. Baris 35 mencoba mengubah teks tadi menjadi angka desimal dengan `toDoubleOrNull()`, yang akan jadi null kalau input tidak valid. Baris 37–40 kalau input kosong, akan muncul pesan `Toast` untuk memberi tahu pengguna bahwa input belum diisi, lalu proses berhenti. Baris 42–47 mengecek apakah input valid dan lebih dari nol. Kalau tidak, maka akan muncul pesan error dan hasil tidak akan ditampilkan. Baris 49–

55 menentukan berapa persen tip berdasarkan pilihan yang dipilih di `RadioGroup`. Kalau belum memilih apa pun, akan muncul pesan supaya memilih kualitas layanan dulu. Baris 56 memanggil fungsi `calculateTip` dari `TipViewModel` dengan memasukkan nilai biaya, persen tip, dan apakah dibulatkan. Hasilnya lalu ditampilkan ke `tipResult`. Baris 58 mencatat log debug dengan `Log.d()` agar pengembang bisa melihat nilai tip yang dihitung saat tombol ditekan.

2. Activity_Main.xml

Baris 1–9 `LinearLayout` bagian ini adalah tempat utama untuk menyusun tampilan dengan arah dari atas ke bawah. `LinearLayout` ini diberi jarak ke dalam (`padding`) supaya isi layar tidak terlalu mepet ke pinggir, dan sudah terhubung dengan `MainActivity` sebagai bagian utamanya. Baris 11–19 `TextView` berfungsi sebagai judul aplikasi dengan tulisan “Tip Time”. Ukuran teks dibuat sedikit besar supaya terlihat seperti judul di bagian atas. Meski ada dua pengaturan latar belakang, hanya satu yang benar-benar bekerja untuk menampilkan warna. Baris 21–28 `EditText` digunakan agar pengguna bisa mengisi jumlah biaya layanan. Masukan hanya dibatasi untuk angka desimal dan sudah ada tulisan petunjuk (`hint`) supaya lebih jelas. Tampilan kotak input memakai latar belakang bawaan dari Android. Baris 30–35 `TextView` berfungsi sebagai label atau pertanyaan untuk menanyakan kualitas layanan, yang nantinya dijawab dengan memilih salah satu dari pilihan yang ada di radio button. Baris 37–59 `RadioGroup` bagian ini berisi tiga pilihan `RadioButton`. Setiap pilihan menunjukkan seberapa baik layanan yang diterima, seperti “Amazing”, “Good”, atau “Okay”, dan masing-masing punya persentase tip yang berbeda. Baris 61–78 `LinearLayout` horizontal ini menampilkan pertanyaan dan switch (saklar) untuk memilih apakah hasil tip mau dibulatkan ke atas. Tapi tata letaknya memakai jarak tetap (`margin`) yang bisa kurang cocok di ukuran layar yang berbeda-beda. Baris 80–87 `Button` digunakan untuk memulai proses menghitung tip. Desain tombol dibuat lebih mudah dilihat dan menarik perhatian pengguna. Baris 89–97

TextView dipakai untuk menampilkan hasil perhitungan tip setelah tombol ditekan. Teksnya dibuat agak ke kanan dengan margin, tapi cara ini kurang fleksibel kalau ditampilkan di berbagai ukuran layar.

3. Button_Color.xml

Baris 1 `<?xml version="1.0" encoding="utf-8"?>` menunjukkan bahwa file ini menggunakan format XML versi 1.0 dan karakter UTF-8. Baris 2 `<selector>` adalah elemen utama yang digunakan untuk menentukan tampilan berbeda berdasarkan keadaan (state) dari sebuah elemen UI, seperti tombol. Baris 3 `<item android:state_pressed="true">` berarti gaya ini akan diterapkan saat tombol ditekan. Baris 4 `<shape android:shape="rectangle">` membentuk tombol dengan sudut membulat. Baris 5 `<solid android:color="#7d499d" />` memberi warna ungu gelap saat tombol ditekan. Baris 6 `<corners android:radius="12dp" />` membulatkan sudut tombol sebesar 12dp. Baris 10 `<item>` tanpa atribut state berarti gaya ini digunakan saat tombol dalam keadaan normal (tidak ditekan). Baris 11 `<shape android:shape="rectangle">` membentuk tombol biasa berbentuk kotak. Baris 12 `<solid android:color="#d395f9" />` memberi warna ungu muda saat tombol tidak ditekan. Baris 13 `<corners android:radius="12dp" />` membulatkan sudut tombol seperti sebelumnya.

4. ViewModel.kt

Baris 1 `package com.example.tiptime` menunjukkan bahwa file ini berada di dalam paket bernama `com.example.tiptime`, yang berfungsi sebagai identitas tempat menyimpan kelas. Baris 3 `import androidx.lifecycle.ViewModel` digunakan untuk mengimpor `ViewModel`, yaitu komponen Android yang menyimpan dan mengelola data agar tetap ada saat orientasi layar berubah.

Baris 4 `import java.text.NumberFormat` digunakan untuk mengimpor kelas yang bisa memformat angka menjadi format uang. Baris 5 `import kotlin.math.ceil` digunakan untuk mengimpor fungsi `ceil`, yang membulatkan angka ke atas. Baris 7 `class TipViewModel: ViewModel()` mendeklarasikan kelas `TipViewModel` yang mewarisi `ViewModel`. Untuk Menyimpan data perhitungan tip agar tidak hilang saat terjadi rotasi layar. Baris 8 `var tipResult: String = ""` adalah variabel untuk menyimpan hasil perhitungan tip dalam bentuk teks. Nilainya kosong saat awal. Baris 10–14 adalah fungsi `calculateTip` yang bertugas menghitung jumlah tip, Baris 10 `fun calculateTip(...)` mendeklarasikan fungsi dengan tiga parameter: `cost` (biaya), `tipPercent` (persentase tip), dan `roundUp` (apakah dibulatkan atau tidak). Baris 11 `var tip = cost * tipPercent` menghitung jumlah tip dengan mengalikan biaya dan persentase tip. Baris 12 `if (roundUp) tip = ceil(tip)` akan membulatkan nilai tip ke atas jika opsi `roundUp` dipilih. Baris 13 `tipResult = NumberFormat.getCurrencyInstance().format(tip)` mengubah nilai tip menjadi format mata uang dan menyimpannya di `tipResult`. Baris 14 `return tipResult` mengembalikan hasil dalam bentuk teks yang sudah diformat.

5. Values/themes.xml

Baris 1 `<resources`
`xmlns:tools="http://schemas.android.com/tools">`
 adalah pembuka file resource yang berisi kumpulan style (gaya tampilan) untuk aplikasi. Baris 3–10 adalah style bernama **Base.Theme.TipTime** yang mewarisi tampilan dari `Theme.Material3.DayNight.NoActionBar`, artinya aplikasi akan mendukung mode terang dan gelap, serta tidak menampilkan Action Bar di atas. Mengatur warna utama, dan status bar menjadi warna putih

Baris 11 adalah komentar (tidak dijalankan), menunjukkan alternatif warna yang bisa digunakan jika ingin. Baris 13 `<style name="Theme.TipTime" parent="Base.Theme.TipTime" />` membuat style lain bernama `Theme.TipTime` yang menggunakan gaya dasar dari `Base.Theme.TipTime`. Baris 15–19 adalah style khusus untuk splash screen `Theme.App.SplashScreen` mewarisi `Theme.SplashScreen`. `windowSplashScreenBackground` mengatur warna latar belakang splash screen jadi putih. `windowSplashScreenAnimatedIcon` menentukan gambar ikon yang muncul saat splash screen, diambil dari `@drawable/android`. `postSplashScreenTheme` menetapkan tema yang dipakai setelah splash screen selesai, yaitu `Base.Theme.TipTime`.

6. AndroidManifest.xml

Baris 1 `<manifest>` adalah bagian utama dari file ini yang berfungsi sebagai identitas dan pengatur izin serta komponen utama aplikasi Android. Baris 2–3 menambahkan namespace yang diperlukan untuk menggunakan atribut Android dan alat bantu dari Android Studio. Baris 5–27 `<application>` adalah bagian yang mendefinisikan pengaturan utama aplikasi, seperti ikon, nama, tema, dan pengaturan backup. `android:allowBackup="true"` artinya data aplikasi bisa dibackup oleh sistem. `android:dataExtractionRules` dan `android:fullBackupContent` merujuk ke file XML yang berisi aturan backup data. `android:icon` dan `android:roundIcon` mengatur ikon aplikasi. `android:label="@string/app_name"` menampilkan nama aplikasi dari file string. `android:supportsRtl="true"` mendukung tata letak dari kanan ke `android:theme="@style/Theme.TipTime"` menetapkan tema utama aplikasi. `tools:targetApi="31"` menunjukkan target versi API Android saat pengujian di Android Studio. Baris 15-17 `<activity android:name=".ViewModel">` mendeklarasikan aktivitas

ViewModel sebagai bagian dari aplikasi, tetapi tidak bisa diakses langsung dari luar aplikasi (`exported="false"`). Baris 18–25 mendeklarasikan aktivitas `MainActivity` yang merupakan halaman pertama yang terbuka saat aplikasi dijalankan (`intent-filter` dengan aksi `MAIN` dan kategori `LAUNCHER`). `android:exported="true"` artinya aktivitas ini bisa diakses dari luar (dibutuhkan untuk aktivitas yang jadi titik masuk). `android:theme="@style/Theme.App.SplashScreen"` menetapkan bahwa `MainActivity` akan memakai tema splash screen saat pertama kali dibuka.

D. Tautan Git

https://github.com/dinzzti/Pemrograman_Mobile