LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 2

ANDROID LAYOUT

Oleh:

Regina Silva M **NIM. 2310817220011**

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
APRIL 2024

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 : Regina Silva Maharatini
NIM : 2310817220011

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

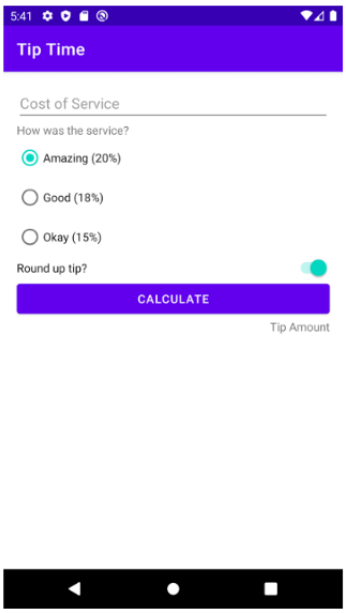
Soal Praktikum:
Buatlah sebuah aplikasi kalkulator tip yang dirancang untuk membantu pengguna menghitung tip yang sesuai berdasarkan total biaya layanan yang mereka terima. Fitur-fitur yang diharapkan dalam aplikasi ini mencakup:

1. Input Biaya Layanan: Pengguna dapat memasukkan total biaya layanan yang diterima dalam bentuk nominal.

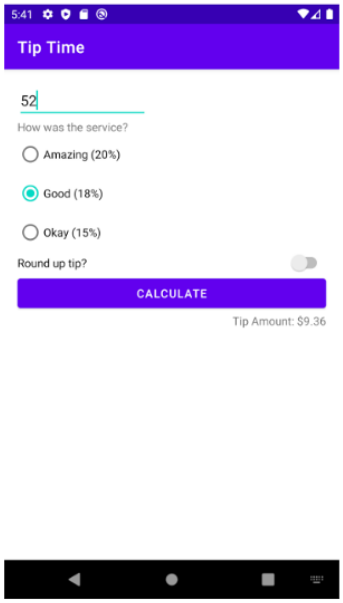
2. Pilihan Persentase Tip: Pengguna dapat memilih persentase tip yang diinginkan dari opsi yang disediakan, yaitu 15%, 18%, dan 20%.

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 Tampilan Awal Aplikasi



Gambar 2 Tampilan Aplikasi Setelah Dijalankan

A. Source Code

1. MainActivity.kt

```
1   package com.example.tipapp
2
3   import android.os.Bundle
4   import androidx.activity.ComponentActivity
5   import androidx.activity.compose.setContent
6   import androidx.activity.enableEdgeToEdge
7   import androidx.compose.foundation.background
8   import androidx.compose.foundation.layout.*
9   import androidx.compose.material3.*
10  import androidx.compose.runtime.Composable
11  import androidx.compose.ui.Alignment
12  import androidx.compose.ui.Modifier
13  import androidx.compose.ui.graphics.Color
14  import androidx.compose.ui.unit.dp
15  import androidx.compose.ui.unit.sp
16  import com.example.tipapp.ui.theme.TipAppTheme
17  import com.example.tipapp.ui.theme.TipScreen
18
19  class MainActivity : ComponentActivity() {
20      override fun onCreate(savedInstanceState: Bundle?) {
21          super.onCreate(savedInstanceState)
22          enableEdgeToEdge()
23          setContent {
24              TipAppTheme {
25                  Surface(modifier = Modifier.fillMaxSize()) {
26                      TipTimeApp()
```

```
                }
            }
        }
    }
}

@Composable
fun TipTimeApp() {
    Column(modifier = Modifier.fillMaxSize()) {
        Box(
            modifier = Modifier
                .fillMaxWidth()
                .height(56.dp)
                .background(Color(0xFF6200EE)),
            contentAlignment = Alignment.Center
        ) {
            Text(
                text = "Tip Time",
                fontSize = 20.sp,
                color = Color.White
            )
        }
        TipScreen(modifier = Modifier.padding(16.dp))
    }
}

```

Tabel 1. Source Code Jawaban Soal 1

2. TipScreen.kt

```
package com.example.tipapp.ui.theme

import android.widget.Toast
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.selection.selectable
import androidx.compose.material3.*
import androidx.compose.runtime.*
import androidx.compose.runtime.saveable.rememberSaveable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.platform.LocalContext
import androidx.compose.ui.unit.dp
import androidx.lifecycle.viewmodel.compose.viewModel

@Composable
fun TipScreen(
    modifier: Modifier = Modifier,
    viewModel: TipViewModel = viewModel()
```

```
) {
    val context = LocalContext.current

    val amountInput by
viewModel.amountInput.collectAsState()
    val tipPercent by
viewModel.tipPercent.collectAsState()
    val roundUp by viewModel.roundUp.collectAsState()

    var showResult by rememberSaveable {
mutableStateOf(false) }
    val tip = viewModel.calculateTip()

    Column(
        modifier = modifier
            .fillMaxSize(),
        verticalArrangement = Arrangement.spacedBy(16.dp)
    ) {
        OutlinedTextField(
            value = amountInput,
            onValueChange = {
viewModel.onAmountChange(it) },
            label = { Text("Cost of Service") },
            modifier = Modifier.fillMaxWidth(),
            singleLine = true
        )

        Text("How was the service?")

        val options = listOf(
            0.20 to "Amazing (20%)",
            0.18 to "Good (18%)",
            0.15 to "Okay (15%)"
        )

        options.forEach { (value, label) ->
            Row(
                Modifier
                    .fillMaxWidth()
                    .selectable(
                        selected = tipPercent == value,
                        onClick = {
viewModel.onTipChange(value) }
                    )
                    .padding(vertical = 4.dp),
                verticalAlignment =
Alignment.CenterVertically

```

```
            ) {
                RadioButton(
                    selected = tipPercent == value,
                    onClick = {
viewModel.onTipChange(value) }
                )
                Text(label)
            }
        }

        Row(verticalAlignment =
Alignment.CenterVertically) {
            Text("Round up tip?")
            Spacer(modifier = Modifier.width(8.dp))
            Switch(
                checked = roundUp,
                onCheckedChange = {
viewModel.onRoundUpChange(it) }
            )
        }

        Button(
            onClick = {
                showResult = amountInput.isNotBlank() &&
amountInput.toDoubleOrNull() != null
                if (!showResult) {
                    Toast.makeText(context, "Please enter
a valid amount", Toast.LENGTH_SHORT).show()
                }
            },
            modifier = Modifier.fillMaxWidth(),
            colors =
ButtonDefaults.buttonColors(containerColor =
MaterialTheme.colorScheme.primary)
        ) {
            Text("CALCULATE", color =
MaterialTheme.colorScheme.onPrimary)
        }

        if (showResult) {
            Text(
                text = "Tip Amount: $%.2f".format(tip),
                style =
MaterialTheme.typography.headlineSmall
            )
        }

```

```
125         }
126 }
127
```

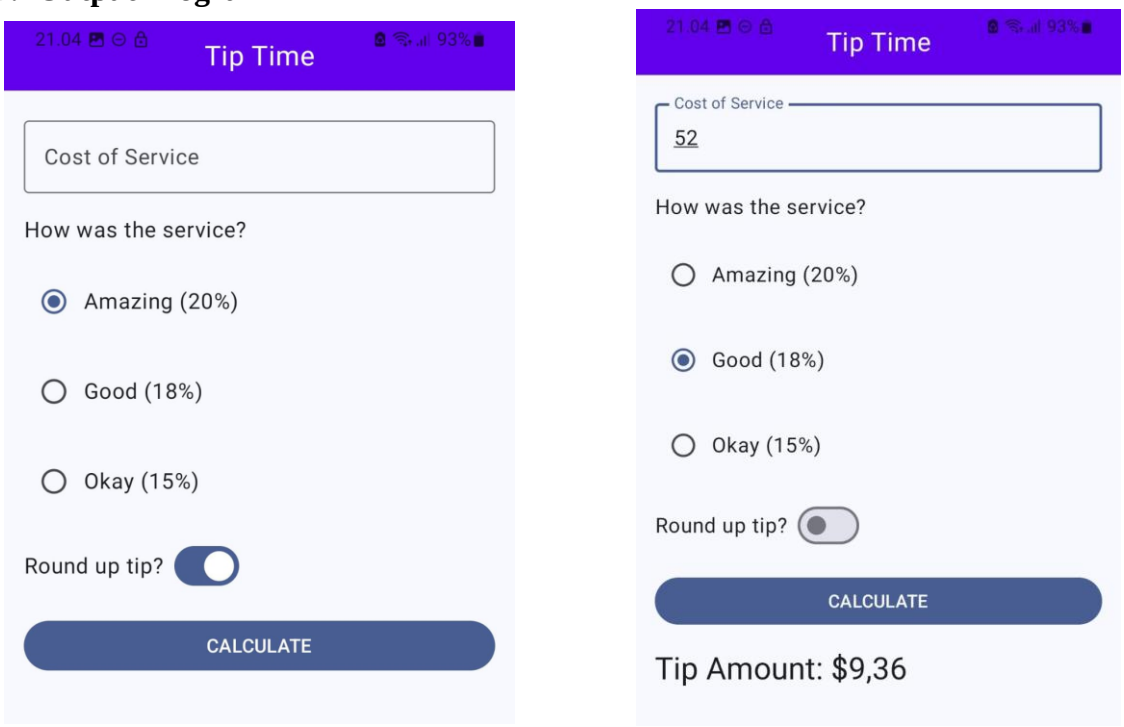
Tabel 2. Source Code Jawaban Soal 1

3. TipViewModel.kt

```
1  package com.example.tipapp.ui.theme
2
3  import androidx.lifecycle.ViewModel
4  import kotlinx.coroutines.flow.MutableStateFlow
5  import kotlinx.coroutines.flow.StateFlow
6  import kotlin.math.ceil
7
8  class TipViewModel : ViewModel() {
9      private val _amountInput = MutableStateFlow("")
10     val amountInput: StateFlow<String> = _amountInput
11
12     private val _tipPercent = MutableStateFlow(0.20)
13     val tipPercent: StateFlow<Double> = _tipPercent
14
15     private val _roundUp = MutableStateFlow(false)
16     val roundUp: StateFlow<Boolean> = _roundUp
17
18     fun onAmountChange(newAmount: String) {
19         _amountInput.value = newAmount
20     }
21
22     fun onTipChange(percent: Double) {
23         _tipPercent.value = percent
24     }
25
26     fun onRoundUpChange(value: Boolean) {
27         _roundUp.value = value
28     }
29
30     fun calculateTip(): Double {
31         val amount = _amountInput.value.toDoubleOrNull()
32 ?: return 0.0
33         var result = amount * _tipPercent.value
34         if (_roundUp.value) result = ceil(result)
35         return result
36     }
37 }
38
```

Tabel 3. Source Code Jawaban Soal 1

B. Output Program



Gambar 1.Screenshot Hasil Jawaban Soal 1

C. Pembahasan

1. MainActivity.kt:

Kelas `MainActivity` adalah tempat pertama aplikasi dijalankan. Di dalam metode `onCreate()`, fungsi `enableEdgeToEdge()` digunakan agar tampilan aplikasi bisa tampil penuh hingga batas layar. Selanjutnya, seluruh antarmuka dibuat dengan Jetpack Compose melalui fungsi `setContent`, lalu dibungkus dengan `TipAppTheme` agar tampilan aplikasi mengikuti tema Material Design. Di dalamnya, fungsi `TipTimeApp()` digunakan untuk menampilkan dua bagian utama, yaitu AppBar berwarna ungu dengan judul "Tip Time", dan konten utama yang dipanggil melalui `TipScreen`, yaitu layar kalkulator tip. Di sini saya tidak menggunakan layout XML, jadi seluruh UI dirancang langsung menggunakan Kotlin Compose, sehingga lebih modern dan efisien.

2. TipScreen.kt

Fungsi `TipScreen()` merupakan bagian utama UI yang menampilkan form kalkulator tip. Fungsi ini menampilkan input biaya layanan menggunakan `OutlinedTextField`, tiga pilihan rating layanan (Amazing, Good, Okay) menggunakan `RadioButton`, switch untuk membulatkan tip, dan tombol "CALCULATE" untuk menghitung hasil. Semua data (input

angka, pilihan persentase, dan switch) dikelola oleh `ViewModel` agar tetap bertahan meskipun terjadi rotasi layar. Jika tombol ditekan tanpa input yang valid, akan muncul pesan toast. Saat input valid, hasil perhitungan tip ditampilkan dalam format dollar secara otomatis. Pendekatan ini menjadikan UI responsif, reaktif, dan tidak mudah ter-reset karena menggunakan `StateFlow` dan `rememberSaveable`.

3. TipViewModel.kt

`TipViewModel` adalah kelas yang mengatur dan menyimpan data aplikasi seperti input biaya, persentase tip, dan status pembulatan. Semua data disimpan dalam bentuk `StateFlow`, sehingga bisa diamati secara real-time di UI. Fungsi `onAmountChange()`, `onTipChange()`, dan `onRoundUpChange()` digunakan untuk memperbarui nilai-nilai tersebut saat pengguna berinteraksi. Fungsi `calculateTip()` bertugas menghitung nilai tip berdasarkan input pengguna dan akan membulatkan hasilnya jika opsi pembulatan diaktifkan. Dengan penggunaan ViewModel ini, data tetap aman meskipun pengguna merotasi layar atau sistem mereset ulang komponen UI.

D. Tautan Git

Berikut adalah tautan untuk source code yang telah dibuat.
https://github.com/rgnsilvas/Pemrograman-Mobile/tree/master/MODUL2