LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 2

Android Layout

Oleh:

Muhammad Rizky NIM. 2310817310011

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 : Muhammad Rizky
NIM : 2310817310011

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

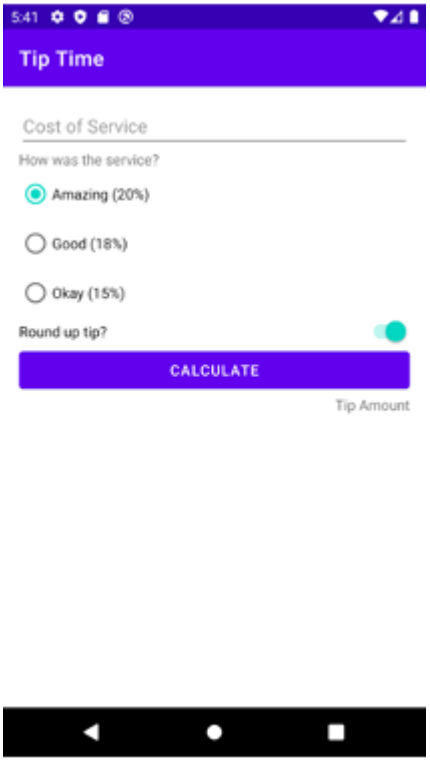
DAFTAR GAMBAR

DAFTAR TABEL

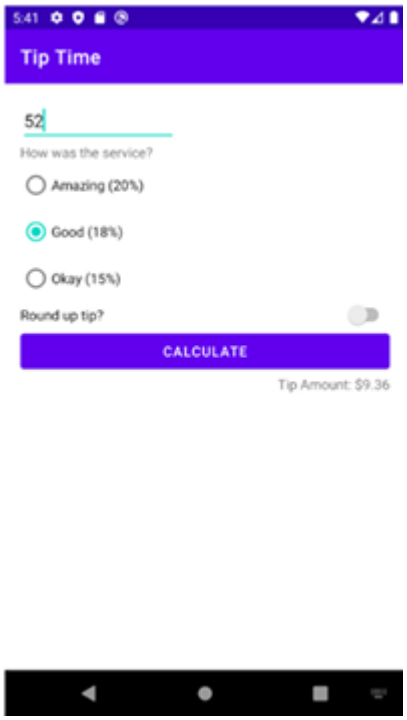
SOAL 1

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

Table 1. Source Code Jawaban Soal 1

```
1   package com.example.modul_2
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.layout.Arrangement
8   import androidx.compose.foundation.layout.Column
9   import androidx.compose.foundation.layout.Row
10  import androidx.compose.foundation.layout.Spacer
11  import androidx.compose.foundation.layout.fillMaxSize
12  import androidx.compose.foundation.layout.fillMaxWidth
13  import androidx.compose.foundation.layout.height
14  import androidx.compose.foundation.layout.padding
15  import androidx.compose.material3.Button
16  import androidx.compose.material3.ButtonDefaults
17  import androidx.compose.material3.ExperimentalMaterial3Api
18  import androidx.compose.material3.MaterialTheme
19  import androidx.compose.material3.OutlinedTextField
20  import androidx.compose.material3.RadioButton
21  import androidx.compose.material3.RadioButtonDefaults
```

```
import androidx.compose.material3.Scaffold
import androidx.compose.material3.Switch
import androidx.compose.material3.SwitchDefaults
import androidx.compose.material3.Text
import androidx.compose.material3.TopAppBar
import androidx.compose.material3.TopAppBarDefaults
import androidx.compose.runtime.Composable
import androidx.compose.runtime.collectAsState
import androidx.compose.runtime.getValue
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
import
androidx.core.splashscreen.SplashScreen.Companion.installSplashScreen
import androidx.lifecycle.viewmodel.compose.viewModel
import androidx.compose.foundation.rememberScrollState
import androidx.compose.foundation.verticalScroll
import com.example.modul_2.ui.theme.Modul_2Theme

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        installSplashScreen()
        enableEdgeToEdge()
        setContent {
            Modul_2Theme {
                TipTime()
            }
        }
    }
}

@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun TipTime(viewModel: TipTimeViewModel = viewModel()){
    val costInput by viewModel.costInput.collectAsState()
    val                    selectedPercentage                        by
viewModel.selectedPercentage.collectAsState()
    val roundTip by viewModel.roundTip.collectAsState()
    val tipAmount by viewModel.tipAmount.collectAsState()

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

    Scaffold(
        topBar = {
            TopAppBar(
                title = { Text("Tip Time", color = Color.White) },

```

```
                colors                                                  =
TopAppBarDefaults.topAppBarColors(containerColor = Color.Red)
            )
        }
    ) { innerPadding ->
        Column(
            modifier = Modifier
                .fillMaxSize()
                .padding(innerPadding)
                .padding(16.dp)
                .verticalScroll(rememberScrollState())
        ) {
            Spacer(modifier = Modifier.height(8.dp))

            OutlinedTextField(
                value = costInput,
                onValueChange = viewModel::onCostChanged,
                label = { Text("Cost of service") },
                singleLine = true,
                modifier = Modifier.fillMaxWidth()
            )

            Text("How     was     the     service?",     style     =
MaterialTheme.typography.titleMedium)
            Spacer(modifier = Modifier.height(8.dp))

            radioOptions.forEach { option ->
                Row(verticalAlignment = Alignment.CenterVertically) {
                    RadioButton(
                        selected     =     selectedPercentage     ==
option.second,
                        onClick                  =                  {
viewModel.onPercentageChanged(option.second) },
                        colors = RadioButtonDefaults.colors(
                            selectedColor = Color.Yellow,
                            unselectedColor = Color.Gray,
                        )
                    )
                    Text(text    =    option.first,    modifier    =
Modifier.padding(start = 8.dp))
                }
            }

            Row(
                verticalAlignment = Alignment.CenterVertically,
                horizontalArrangement = Arrangement.SpaceBetween,
                modifier = Modifier.fillMaxWidth()
            ) {
                Text("Round tip?")
                Switch(
                    checked = roundTip,
                    onCheckedChange = viewModel::onRoundTipChanged,
```

```
                    colors = SwitchDefaults.colors(
                        checkedThumbColor = Color.Red,
                        checkedTrackColor = Color.White,
                        checkedBorderColor = Color.Red
                    )
                )
            }

            Button(
                onClick = viewModel::calculateTip,
                colors = ButtonDefaults.buttonColors(containerColor =
Color.Red),
                modifier = Modifier.fillMaxWidth()
            ) {
                Text("Calculate")
            }

            Text(
                text = tipAmount,
                modifier = Modifier.fillMaxWidth(),
                textAlign = TextAlign.End
            )
        }
    }
}

@Preview(showBackground = true)
@Composable
fun TipTimePreview() {
    Modul_2Theme {
        TipTime()
    }
}
```

2. TipTimeViewModel.kt

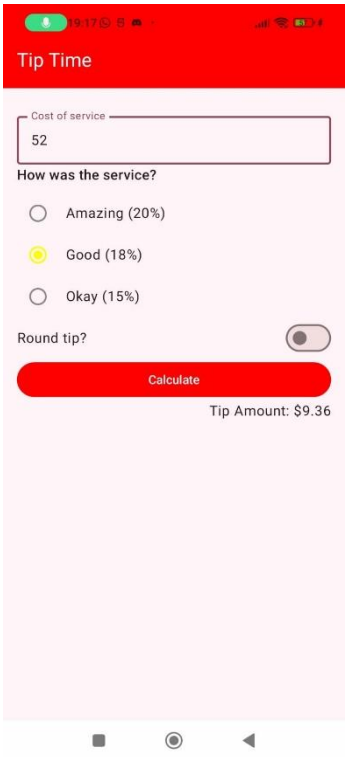
Table 2. Source Code Jawaban Soal 1

```
package com.example.modul_2

import androidx.lifecycle.ViewModel
import kotlinx.coroutines.flow.MutableStateFlow
import kotlinx.coroutines.flow.StateFlow
import kotlin.math.ceil

class TipTimeViewModel : ViewModel() {

    private val _costInput = MutableStateFlow("")
    val costInput: StateFlow<String> = _costInput

    private val _selectedPercentage = MutableStateFlow(0.20)
```

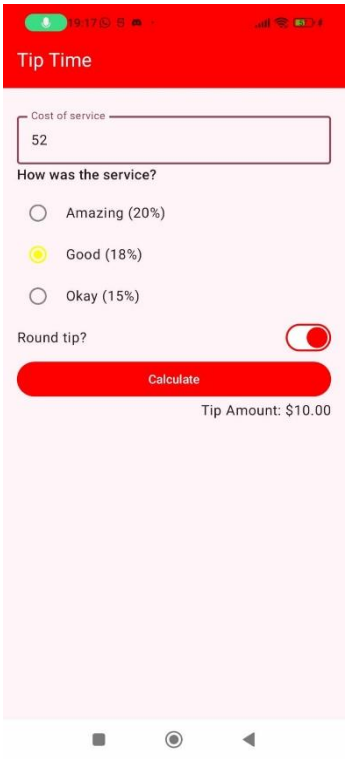
```
    val selectedPercentage: StateFlow<Double> = _selectedPercentage

    private val _roundTip = MutableStateFlow(true)
    val roundTip: StateFlow<Boolean> = _roundTip

    private val _tipAmount = MutableStateFlow("Tip Amount")
    val tipAmount: StateFlow<String> = _tipAmount

    fun onCostChanged(newCost: String) {
        _costInput.value = newCost
    }

    fun onPercentageChanged(newPercentage: Double) {
        _selectedPercentage.value = newPercentage
    }

    fun onRoundTipChanged(newRound: Boolean) {
        _roundTip.value = newRound
    }

    fun calculateTip() {
        val cost = costInput.value.toDoubleOrNull()
        if (cost != null) {
            val tip = cost * selectedPercentage.value
            val finalTip = if (roundTip.value) ceil(tip) else tip
            _tipAmount.value = "Tip Amount: $%.2f".format(finalTip)
        } else {
            _tipAmount.value = "Tip Amount"
        }
    }
}
```

B. Output Program

Gambar 1. Screenshot Splashscreen Aplikasi



Gambar 2. Screenshot Tampilan Aplikasi



Gambar 3. Screenshot Tampilan Aplikasi

Gambar 4. Screenshot Tampilan Aplikasi

C. Pembahasan

1. MainActivity.kt:

Line 1, Pada line ini, dideklarasikan nama package file Kotlin com.example.modul_2

Line 3, Diimport Bundle, yang digunakan untuk menyimpan data dan mengirimkan data antar aktivitas, sering digunakan dalam metode onCreate() untuk menerima data yang dikirimkan ke aktivitas baru

Line 4, Diimport ComponentActivity dari AndroidX yang digunakan sebagai superclass untuk aktivitas yang menggunakan Jetpack Compose.

Line 5, Diimport setContent yang digunakan untuk mengonfigurasi konten UI di dalam activity

Line 6, Diimport enableEdgeToEdge, yang memungkinkan konten UI aplikasi agar dapat dilihat hingga ke tepi layar perangkat

Line 7-22, Diimport beberapa komponen layout seperti Column, Row, Spacer, fillMaxSize, padding, yang digunakan untuk pengaturan tata letak antarmuka pengguna dalam Compose

Line 23, Diimport Button dari Material3, digunakan untuk membuat tombol interaktif di UI

Line 24, Diimport ButtonDefaults, yang memungkinkan kita mengatur warna default atau gaya tombol, seperti mengubah warna latar belakang atau tek

Line 25, Diimport ExperimentalMaterial3Api sebagai anotasi yang menandakan bahwa kita menggunakan API Material3 yang masih dalam status eksperimental

Line 26, Diimport MaterialTheme, yang memungkinkan kita untuk menggunakan tema Material Design pada UI, seperti pengaturan warna dan tipografi

Line 27, Diimport OutlinedTextField, yang digunakan untuk membuat field input dengan garis tepi di sekitarnya

Line 28, Diimport RadioButton, yang digunakan untuk menampilkan pilihan tunggal dalam bentuk tombol radio, yang hanya memungkinkan pengguna memilih satu dari beberapa opsi yang tersedia

Line 29, Diimport RadioButtonDefaults, yang digunakan untuk mengatur warna dan warna radio button, seperti warna ketika dipilih atau tidak dipilih

Line 30, Diimport Scaffold, komponen tata letak utama di Jetpack Compose yang mendukung struktur UI seperti top app bar, floating action button (FAB), dan body konten

Line 31, Diimport Switch, yang digunakan untuk membuat komponen toggle, memungkinkan pengguna untuk memilih antara dua status (on/off)

Line 32, Diimport SwitchDefaults, untuk mengatur warna dan tampilan dari komponen switch, termasuk warna latar belakang dan thumb ketika berada dalam kondisi on/of

Line 33, Diimport Text, digunakan untuk menampilkan teks di UI Ini adalah komponen dasar untuk menampilkan string di layar

Line 34, Diimport TopAppBar, digunakan untuk menampilkan bar bagian atas aplikasi yang biasanya berisi judul aplikasi atau tindakan penting lainnya

Line 35, Diimport TopAppBarDefaults, untuk mengatur warna default dan gaya dari top app bar, seperti latar belakang dan warna teks

Line 36, Diimport Composable, adalah anotasi yang digunakan untuk menandai fungsi sebagai composable, yaitu fungsi yang mengembalikan UI dalam Jetpack Compose

Line 37, Diimport collectAsState, digunakan untuk mengubah aliran data (Flow) dari ViewModel menjadi state yang dapat dipantau di Compose

Line 38, Diimport getValue, digunakan untuk mendeklarasikan delegasi properti dengan menggunakan by, memungkinkan kita untuk mengakses nilai property secara otomatis

Line 39, Diimport Alignment, digunakan untuk mengatur posisi elemen dalam layout, baik secara vertikal maupun horizontal

Line 40, Diimport Modifier, digunakan untuk menambahkan atau memodifikasi properti dari elemen UI seperti ukuran, padding, margin, dan lain-lain

Line 41, Diimport Color, digunakan untuk mengatur warna elemen UI seperti latar belakang, teks, tombol, dan lain-lain

Line 42, Diimport TextAlign, digunakan untuk mengatur perataan teks secara horizontal di dalam komponen Tex

Line 43, Diimport tooling.preview.Preview, digunakan untuk menampilkan preview dari UI Compose dalam Android Studio sebelum dijalankan pada perangkat

Line 44, Diimport unit.dp, digunakan untuk menentukan ukuran dalam satuan density-independent pixels (dp), yang berguna untuk membuat UI responsif di berbagai ukuran layar

Line 45, Diimport installSplashScreen, digunakan untuk mengonfigurasi splash screen pada aplikasi Android yang muncul saat aplikasi dijalankan

Line 46, Diimport viewModel, digunakan untuk mengakses instance dari ViewModel untuk memisahkan logika bisnis dan UI dalam aplikas

Line 47, Diimport rememberScrollState, digunakan untuk membuat state yang mengingat status scroll saat pengguna menggulir layout

Line 48, Diimport verticalScroll, digunakan untuk memberikan kemampuan scroll vertikal pada layout, memungkinkan pengguna untuk menggulir konten

Line 49, Diimport Modul_2Theme, adalah tema kustom aplikasi yang diterapkan ke seluruh tampilan antarmuka

Line 51, Di dalam MainActivity, kita mendefinisikan fungsi onCreate() Fungsi ini dijalankan saat aktivitas pertama kali dibuat dan digunakan untuk mengonfigurasi konten UI dan pengaturan lainny

Line 52, Dalam fungsi onCreate(), kita memanggil installSplashScreen() untuk menampilkan layar pembuka saat aplikasi mulai Fungsi ini menyediakan efek loading sementara aplikasi berjalan

Line 53, Selanjutnya, enableEdgeToEdge() digunakan untuk membuat layout aplikasi bisa merentang hingga ke tepi layar perangkat, memberikan tampilan modern dengan edge-to-edge design

Line 54, setContent {} adalah fungsi Compose yang menggantikan setContentView() Di sini, kita menggunakan tema kustom Modul_2Theme dan menyetel UI yang akan ditampilkan di aktivitas, yaitu fungsi TipTime()

Line 56, Fungsi TipTime() adalah composable yang digunakan untuk membuat UI dari aplikasi Fungsi ini mengelola semua elemen interaktif, seperti inputan pengguna dan tampilan hasil kalkulasi

Line 57, viewModel() digunakan untuk mendapatkan instance dari TipTimeViewModel, yang menyimpan data dan logika bisnis dari aplikasi

Line 58-60, Mendeklarasikan variabel dengan collectAsState(), yang digunakan untuk mengambil nilai terbaru dari aliran data yang dikendalikan oleh ViewModel Variabel-variabel ini berisi nilai yang berkaitan dengan input pengguna seperti biaya layanan, pilihan persentase tip, apakah tip dibulatkan, dan jumlah tip yang dihitung

Line 62, radioOptions adalah daftar pasangan label teks dan nilai persentase yang digunakan dalam pilihan radio button untuk memilih tingkat pelayanan (misalnya, Amazing 20%)

Line 64-72, Scaffold digunakan untuk membuat layout dasar dengan top app bar dan konten utama aplikasi Di dalam Scaffold, kita mendeklarasikan semua elemen UI yang ditampilkan di layar, seperti field input, pilihan radio button, switch, tombol, dan teks hasil perhitungan

Line 74-82, Bagian ini berisi elemen UI seperti OutlinedTextField untuk input biaya layanan, teks penjelasan, RadioButton untuk memilih tingkat pelayanan, dan Switch untuk memilih apakah tip dibulatkan

Line 84-86, Button digunakan untuk menghitung tip saat pengguna menekan tombol, di mana viewModel::calculateTip dipanggil untuk menghitung nilai tip berdasarkan input pengguna

Line 88-90, Text digunakan untuk menampilkan jumlah tip yang dihitung, dengan perataan teks ke kanan (end) menggunakan TextAlign.End

Line 92-94, TipTimePreview adalah fungsi anotasi @Preview yang memungkinkan kita melihat tampilan TipTime() secara langsung di Android Studio tanpa menjalankan aplikasi pada perangkat Ini memberikan gambaran awal dari UI tanpa perlu memuat aplikasi sepenuhnya

2. TipTimeViewModel.kt

Line 1, Mendeklarasikan nama package com.example.modul_2 sebagai lokasi file dalam struktur proyek aplikasi

Line 3, Mengimpor ViewModel dari AndroidX untuk mengelola data UI

Line 4, Mengimpor MutableStateFlow dan StateFlow dari Kotlin Coroutines untuk pengelolaan dan pemantauan data secara reaktif

Line 5, Mengimpor fungsi ceil dari kotlin.math untuk membulatkan nilai ke atas

Line 7, Mendeklarasikan kelas TipTimeViewModel yang merupakan turunan dari ViewModel, berfungsi untuk mengelola data kalkulasi tip

Line 9, Mendeklarasikan variabel _costInput bertipe MutableStateFlow<String> untuk menyimpan input biaya layanan dari pengguna

Line 10, Mendeklarasikan properti costInput bertipe StateFlow<String> untuk memberikan akses hanya-baca terhadap nilai _costInput

Line 12, Mendeklarasikan variabel _selectedPercentage bertipe MutableStateFlow<Double> untuk menyimpan persentase tip yang dipilih

Line 13, Mendeklarasikan properti selectedPercentage bertipe StateFlow<Double> untuk memberikan akses hanya-baca terhadap nilai persentase tip

Line 15, Mendeklarasikan variabel _roundTip bertipe MutableStateFlow<Boolean> untuk menyimpan status pembulatan tip

Line 16, Mendeklarasikan properti roundTip bertipe StateFlow<Boolean> untuk memberikan akses hanya-baca terhadap status pembulatan

Line 18, Mendeklarasikan variabel _tipAmount bertipe MutableStateFlow<String> untuk menyimpan hasil kalkulasi jumlah tip

Line 19, Mendeklarasikan properti tipAmount bertipe StateFlow<String> untuk memberikan akses hanya-baca terhadap nilai hasil kalkulasi tip

Line 21, Mendeklarasikan fungsi onCostChanged(newCost: String) untuk memperbarui nilai _costInput dengan input baru dari pengguna

Line 23, Mendeklarasikan fungsi onPercentageChanged(newPercentage: Double) untuk memperbarui nilai _selectedPercentage

Line 25, Mendeklarasikan fungsi onRoundTipChanged(newRound: Boolean) untuk memperbarui nilai _roundTip

Line 27, Mendeklarasikan fungsi calculateTip() untuk menghitung jumlah tip berdasarkan input biaya dan persentase tip

Line 28, Mengambil nilai cost dari costInput dan mengonversinya menjadi Double, jika tidak valid maka hasil tip tidak diperbarui

Line 30, Mengalikan biaya layanan dengan persentase tip untuk mendapatkan nilai tip awal

Line 32, Membulatkan nilai tip menggunakan ceil jika status pembulatan diaktifkan

Line 33, Memformat hasil kalkulasi tip dan menyimpannya dalam _tipAmount dalam format mata uang.

D. Tautan Git

Berikut adalah tautan untuk source code yang telah dibuat.
https://github.com/kkeshiian/Pemrograman-Mobile/tree/main/MODUL%202