LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 2

ANDROID LAYOUT WITH COMPOSE

Oleh:

Muhammad Raihan NIM. 2310817110008

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
APRIL 2025

LEMBAR PENGESAHAN
LAPORAN PRAKTIKUM PEMROGRAMAN I
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 : Muhammad Raihan
NIM : 2310817110008

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

DAFTAR GAMBAR

DAFTAR TABEL

SOAL 1

Soal Praktikum:

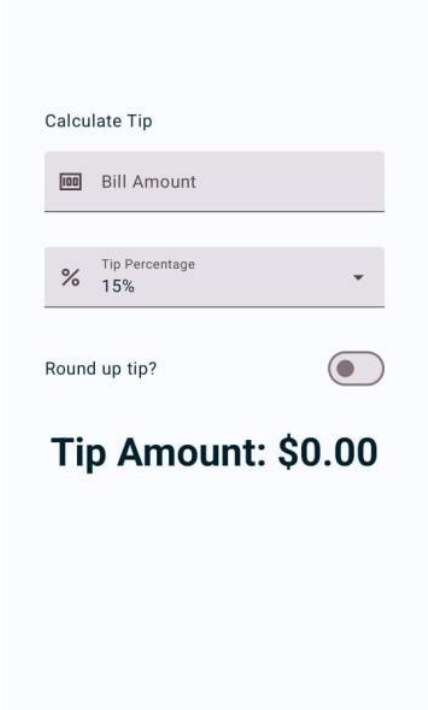
Buatlah sebuah aplikasi kalkulator tip menggunakan XML dan Jetpack Compose yang dirancang untuk membantu pengguna menghitung tip yang sesuai berdasarkan total biaya layanan yang mereka terima. Fitur-fitur yang diharapkan dalam aplikasi ini mencakup:

a. Input biaya layanan: Pengguna dapat memasukkan total biaya layanan yang diterima dalam bentuk nominal.

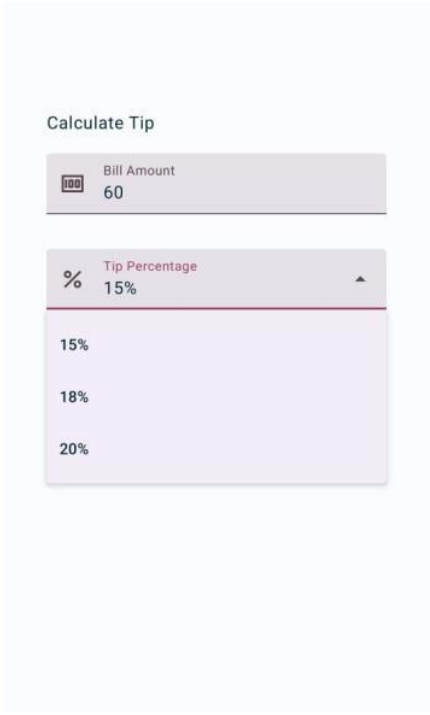
b. Pilihan persentase tip: Pengguna dapat memilih persentase tip yang diinginkan.

c. Pengaturan pembulatan tip: Pengguna dapat memilih untuk membulatkan tip ke angka yang lebih tinggi.

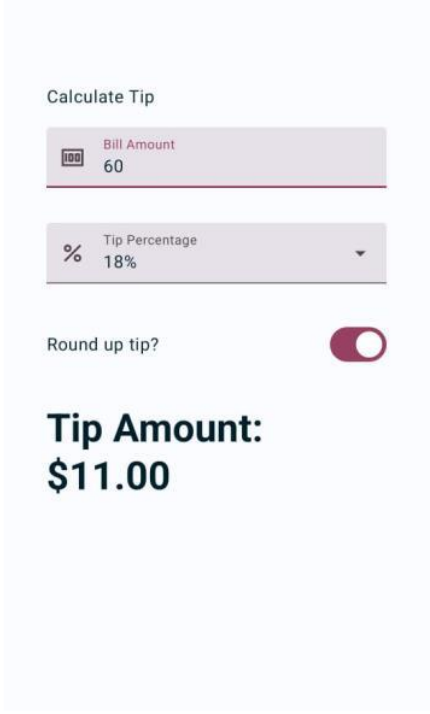
d. Tampilan hasil: Aplikasi akan menampilkan jumlah tip yang harus dibayar secara langsung setelah pengguna memberikan input.



Gambar 1. Tampilan Awal Aplikasi



Gambar 2. Tampilan Pilihan Persentase Tip



Gambar 3. Tampilan Aplikasi Setelah Dijalankan

A. Source Code

1) Versi Jetpack Compose

MainActivity.kt

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

```
1   package com.example.tipcalculatorcompose
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.annotation.DrawableRes
8   import androidx.annotation.StringRes
9   import
    androidx.compose.foundation.layout.Arrangement
10  import androidx.compose.foundation.layout.Column
11  import androidx.compose.foundation.layout.Row
12  import
    androidx.compose.foundation.layout.fillMaxSize
13  import
    androidx.compose.foundation.layout.fillMaxWidth
14  import androidx.compose.foundation.layout.height
15  import androidx.compose.foundation.layout.padding
16  import androidx.compose.foundation.layout.size
17  import
    androidx.compose.foundation.text.KeyboardOptions
18  import androidx.compose.material3.DropdownMenuItem
19  import
    androidx.compose.material3.ExperimentalMaterial3Api
20  import
    androidx.compose.material3.ExposedDropdownMenuBox
21  import
    androidx.compose.material3.ExposedDropdownMenuDefaul
    ts
22  import androidx.compose.material3.Icon
23  import androidx.compose.material3.Surface
24  import androidx.compose.material3.Switch
25  import androidx.compose.material3.Text
26  import androidx.compose.material3.TextField
27  import androidx.compose.runtime.Composable
28  import androidx.compose.runtime.getValue
29  import androidx.compose.runtime.mutableIntStateOf
30  import androidx.compose.runtime.mutableStateOf
31  import androidx.compose.runtime.remember
32  import androidx.compose.runtime.setValue
33  import androidx.compose.ui.Alignment
34  import androidx.compose.ui.Modifier
```

```
35  import androidx.compose.ui.res.painterResource
36  import androidx.compose.ui.res.stringResource
37  import androidx.compose.ui.text.font.FontWeight
38  import androidx.compose.ui.text.input.ImeAction
39  import androidx.compose.ui.text.input.KeyboardType
40  import androidx.compose.ui.tooling.preview.Preview
41  import androidx.compose.ui.unit.dp
42  import androidx.compose.ui.unit.sp
43  import
    com.example.tipcalculatorcompose.ui.theme.TipCalcula
    torComposeTheme
44  import java.text.NumberFormat
45  import java.util.Locale
46
47  class MainActivity : ComponentActivity() {
48      override    fun    onCreate(savedInstanceState:
49  Bundle?) {
50          super.onCreate(savedInstanceState)
51          enableEdgeToEdge()
52          setContent {
53              TipCalculatorComposeTheme {
54                  Surface (
55                      Modifier.fillMaxSize()
56                  ) {
57                      TipTimeLayout()
58                  }
59              }
60          }
61      }
62  }
63
64  @OptIn(ExperimentalMaterial3Api::class)
65  @Composable
66  fun TipTimeLayout() {
67      val options = listOf(15, 18, 20)
68      var amountInput by remember { mutableStateOf("")
    }
69      var expanded by remember { mutableStateOf(false)
    }
70      var    selectedOption    by    remember    {
    mutableIntStateOf(options[0]) }
71
72      val amount = amountInput.toDoubleOrNull() ?: 0.0
73      val tipPercent = selectedOption.toDouble()
74      var roundUp by remember { mutableStateOf(false) }
75      val tipAmount = calculateTip(amount, tipPercent,
    roundUp)
```

```
76      Column (
77          modifier = Modifier
78              .padding(horizontal = 40.dp),
79          horizontalAlignment                          =
    Alignment.CenterHorizontally,
80          verticalArrangement = Arrangement.Center
        ) {
81          Text(
82              text                                     =
    stringResource(R.string.calculate_tip),
83              modifier = Modifier
84                  .padding(bottom = 15.dp)
85                  .align(alignment = Alignment.Start)
86          )
87
88          EditNumberField(
89              label = R.string.bill_amount,
90              leadingIcon = R.drawable.money,
91              value = amountInput,
92              onValueChanged = { amountInput = it },
93              keyboardOptions = KeyboardOptions(
94                  keyboardType = KeyboardType.Number,
95                  imeAction = ImeAction.Next
96              ),
97              modifier = Modifier
98                  .padding(bottom = 15.dp)
99                  .fillMaxWidth()
100         )
101
102         ExposedDropdownMenuBox(
103             expanded = expanded,
104             onExpandedChange = {
105                 expanded = !expanded
106             },
107             modifier  =  Modifier.padding(bottom  =
    15.dp)
108         ) {
109             TextField(
110                 modifier = Modifier
111                     .menuAnchor()
112                     .fillMaxWidth(),
113                 readOnly = true,
114                 value = "$selectedOption%",
115                 onValueChange = { },
116                 label               =               {
    Text(stringResource(R.string.tip)) },
```

```
117                 leadingIcon = { Icon(painter =
    painterResource(R.drawable.percent),
118 contentDescription = null, Modifier.size(20.dp)) },
119                 trailingIcon = {

120 ExposedDropdownMenuDefaults.TrailingIcon(
121                         expanded = expanded
122                     )
123                 },
124                 colors                                  =
    ExposedDropdownMenuDefaults.textFieldColors()
125             )
126             ExposedDropdownMenu(
127                 expanded = expanded,
128                 onDismissRequest = {
129                     expanded = false
130                 }
131             ) {
132                 options.forEach { selectionOption ->
133                     DropdownMenuItem(
134                         text = { Text(text =
    "$selectionOption%") },
135                         onClick = {
136                             selectedOption          =
    selectionOption
137                             expanded = false
138                         }
139                     )
140                 }
141             }
142         }
143
144         RoundTheTipRow(
145             roundUp = roundUp,
146             onRoundUpChanged = { roundUp = it },
147             modifier = Modifier.padding(bottom =
    30.dp)
148         )
149
150         Text(
151             text                                        =
    stringResource(R.string.tip_amount, tipAmount),
152             fontSize = 27.sp,
153             fontWeight = FontWeight.Bold
154         )
155     }
156 }
```

```
157
158 @Composable
159 fun EditNumberField(
160     value: String,
161     @StringRes label: Int,
162     @DrawableRes leadingIcon: Int,
163     onValueChanged: (String) -> Unit,
164     keyboardOptions: KeyboardOptions,
165     modifier: Modifier = Modifier
166 ) {
167     TextField(
168         value = value,
169         singleLine = true,
170         modifier = modifier,
171         onValueChange = onValueChanged,
172         label = {Text(stringResource(label))},
173         leadingIcon     =     {     Icon(painter     =
    painterResource(id     =     leadingIcon),     null,
    Modifier.size(20.dp)) },
174
175         keyboardOptions = keyboardOptions
176     )
177 }
178
179 @Composable
180 fun RoundTheTipRow(
181     roundUp: Boolean,
182     onRoundUpChanged: (Boolean) -> Unit,
183     modifier: Modifier = Modifier
    ) {
184     Row(
185         modifier = modifier
186             .fillMaxWidth()
187             .height(48.dp),
188         verticalAlignment                              =
    Alignment.CenterVertically,
189         horizontalArrangement                          =
    Arrangement.SpaceBetween
190     ) {
191         Text(
192             text                                       =
    stringResource(R.string.round_up_tip),
193             modifier = Modifier.weight(1f)
194         )
195         Switch(
196             checked = roundUp,
197             onCheckedChange = onRoundUpChanged
```

```
198         )
199     }
200 }
201
202 private fun calculateTip(amount: Double, tipPercent:
    Double = 15.0, roundUp: Boolean): String {
203     var tipAmount = (tipPercent / 100) * amount
204     if(roundUp){
205         tipAmount = kotlin.math.ceil(tipAmount)
206     }
207
208     return
    NumberFormat.getCurrencyInstance(Locale.US).format(t
    ipAmount)
209 }
210
211 @Preview(
212     name = "Redmi Note 13",
213     widthDp = 360,
214     heightDp = 800,
215     showBackground = true
216 )
217 @Composable
218 fun TipCalculatorPreview() {
219     TipCalculatorComposeTheme {
220         TipTimeLayout()
221     }
222 }
223
```

2) Versi XML

MainActivity.kt

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

```
1   package com.example.tipcalculatorxml
2
3   import android.os.Bundle
4   import android.widget.ArrayAdapter
5   import androidx.appcompat.app.AppCompatActivity
6   import androidx.core.widget.addTextChangedListener
7   import
    com.example.tipcalculatorxml.databinding.ActivityMain
    Binding
8   import kotlin.math.ceil
9
10  class MainActivity : AppCompatActivity() {
```

```
11      private lateinit var binding: ActivityMainBinding
12
13      override fun onCreate(savedInstanceState: Bundle?)
    {
14          super.onCreate(savedInstanceState)
15          binding                                    =
    ActivityMainBinding.inflate(layoutInflater)
16          setContentView(binding.root)

17          val items = listOf("15%", "18%", "20%")
18          val     adapter     =     ArrayAdapter(this,
    R.layout.list_items, items)
19          binding.dropdownField.setAdapter(adapter)
20          binding.dropdownField.setText(items[0],
    false)
21          binding.tipAmount.text                     =
    getString(R.string.tip_amount, 0.0)
22
23          fun updateTip(isRoundUp: Boolean) {
24              val              billAmount            =
    binding.billAmountInput.text.toString().toDoubleOrNul
    l() ?: 0.0
25              val              tipPercent            =
    binding.dropdownField.text.toString().replace("%",
    "").toDoubleOrNull() ?: 0.0
26              var tipAmount = billAmount * tipPercent /
    100
27              if(isRoundUp)      {      tipAmount      =
    ceil(tipAmount) }
28              binding.tipAmount.text                 =
    getString(R.string.tip_amount, tipAmount)
29          }
30
31
32  binding.billAmountInput.addTextChangedListener {

33  updateTip(binding.switchRoundUp.isChecked)
            }
34          binding.dropdownField.addTextChangedListener
    {

35  updateTip(binding.switchRoundUp.isChecked)
36          }
37
38  binding.switchRoundUp.setOnCheckedChangeListener { _,
    isChecked ->
39              updateTip(isChecked)
```

```
40          }
41
42          updateTip(false)
43      }
44  }
```

activity_main.xml

Tabel 3. Source Code activity_main.xml Jawaban Soal 1 Versi XML

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/an
    droid"
3       xmlns:app="http://schemas.android.com/apk/res-
    auto"
4       xmlns:tools="http://schemas.android.com/tools"
5       android:id="@+id/main"
6       android:layout_width="match_parent"
7       android:layout_height="match_parent"
8       android:padding="40dp"
9       tools:context=".MainActivity">
10
11  <androidx.constraintlayout.widget.ConstraintLayout
12          android:layout_width="0dp"
13          android:layout_height="wrap_content"
14          app:layout_constraintTop_toTopOf="parent"
15  app:layout_constraintBottom_toBottomOf="parent"
16  app:layout_constraintStart_toStartOf="parent"
17          app:layout_constraintEnd_toEndOf="parent">
18
19          <TextView
20              android:id="@+id/calc_tip"
21              android:layout_width="0dp"
22              android:layout_height="wrap_content"
23
    app:layout_constraintTop_toTopOf="parent"
24
    app:layout_constraintStart_toStartOf="parent"
25
    app:layout_constraintEnd_toEndOf="parent"
26              android:text="@string/calc_tip"
27              android:textSize="18sp"/>
28
```

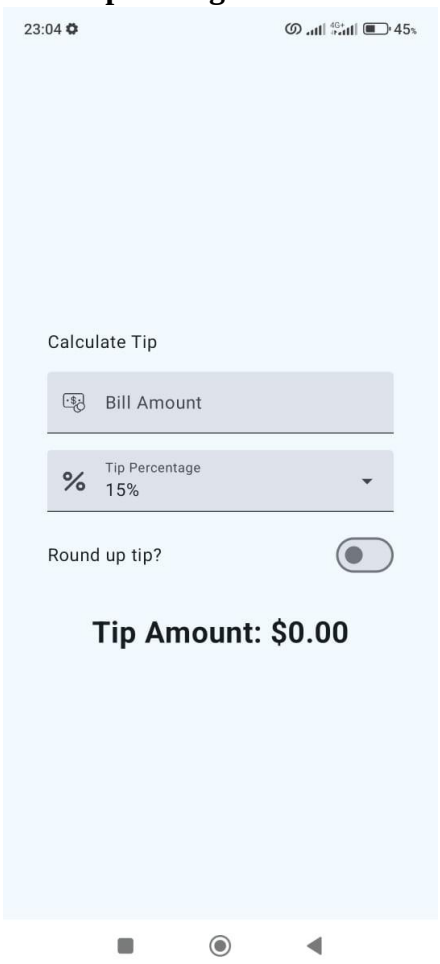
```
29  <com.google.android.material.textfield.TextInputLayo
    ut
30              android:id="@+id/bill_amount"
31              android:layout_width="0dp"
32              android:layout_height="wrap_content"
33              android:layout_marginTop="15dp"
34              android:hint="@string/bill_amount"
35              android:minHeight="48dp"

36  app:layout_constraintEnd_toEndOf="parent"

37  app:layout_constraintStart_toStartOf="parent"

38  app:layout_constraintTop_toBottomOf="@id/calc_tip"

39  app:startIconDrawable="@drawable/money_resized">
40

41  <com.google.android.material.textfield.TextInputEdit
    Text
42                  android:id="@+id/bill_amount_input"
43                  android:layout_width="match_parent"
44                  android:layout_height="wrap_content"
45                  android:inputType="numberDecimal" />
46

47  </com.google.android.material.textfield.TextInputLay
    out>
48

49  <com.google.android.material.textfield.TextInputLayo
    ut
50              android:id="@+id/dropdown_layout"
51              android:layout_width="0dp"

52  app:layout_constraintTop_toBottomOf="@id/bill_amount
    "

53  app:layout_constraintStart_toStartOf="parent"

54  app:layout_constraintEnd_toEndOf="parent"
55              android:layout_height="wrap_content"
56              android:layout_marginTop="15dp"

57  app:startIconDrawable="@drawable/percent_resized"
58              android:hint="@string/tip_percentage"
```

```
59  style="@style/Widget.Material3.TextInputLayout.OutlinedBox.ExposedDropdownMenu">
60
61              <AutoCompleteTextView
62                  android:id="@+id/dropdown_field"
63                  android:layout_width="match_parent"
64                  android:layout_height="wrap_content"
65                  android:inputType="none"
66  tools:ignore="LabelFor,SpeakableTextPresentCheck" />
67
68  </com.google.android.material.textfield.TextInputLayout>
69
70  <androidx.constraintlayout.widget.ConstraintLayout
71              android:id="@+id/roundUpLayout"
72              android:layout_width="0dp"
73              android:layout_height="wrap_content"
74  app:layout_constraintTop_toBottomOf="@id/dropdown_layout"
75  app:layout_constraintStart_toStartOf="parent"
76  app:layout_constraintEnd_toEndOf="parent"
77              android:layout_marginTop="10dp">
78
79              <TextView
80                  android:id="@+id/roundUpTip"
81                  android:layout_width="wrap_content"
82                  android:layout_height="wrap_content"
83  app:layout_constraintTop_toTopOf="parent"
84  app:layout_constraintBottom_toBottomOf="parent"
85  app:layout_constraintStart_toStartOf="parent"
86                  android:text="@string/round_up_tip"
87                  android:textSize="16sp"/>
88
89              <View
90                  android:id="@+id/spacer"
91                  android:layout_width="0dp"
92                  android:layout_height="0dp"
```

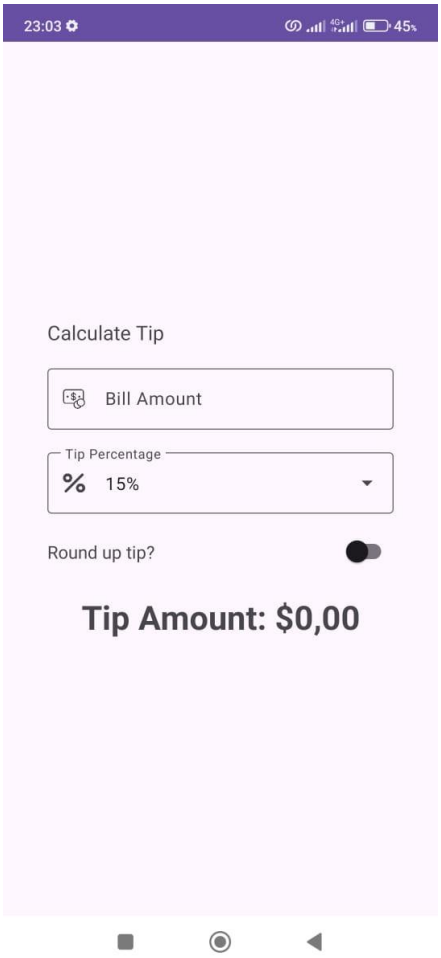
```
93  app:layout_constraintStart_toEndOf="@id/roundUpTip"
94  app:layout_constraintTop_toTopOf="parent"
95  app:layout_constraintBottom_toBottomOf="parent"
96  app:layout_constraintEnd_toStartOf="@id/switch_round
    _up"
97  app:layout_constraintHorizontal_weight="1"/>
98
99  <com.google.android.material.switchmaterial.SwitchMa
    terial
100                 android:id="@+id/switch_round_up"
101                 android:layout_width="wrap_content"
102                 android:layout_height="wrap_content"
103 app:layout_constraintTop_toTopOf="parent"
104 app:layout_constraintEnd_toEndOf="parent"
105                 android:padding="0dp"
106                 />
107
108
109
110
111
112 </androidx.constraintlayout.widget.ConstraintLayout>
113
114         <TextView
115             android:id="@+id/tip_amount"
116             android:layout_width="0dp"
117             android:layout_height="wrap_content"
118 app:layout_constraintTop_toBottomOf="@id/roundUpLayo
    ut"
119 app:layout_constraintStart_toStartOf="parent"
120 app:layout_constraintEnd_toEndOf="parent"
121             android:layout_marginTop="15dp"
122             android:textAlignment="center"
123             android:textSize="30sp"
124             android:textStyle="bold"/>
```

```
125
126 </androidx.constraintlayout.widget.ConstraintLayout>
127
128 </androidx.constraintlayout.widget.ConstraintLayout>
```

B. Output Program



Gambar 4. Screenshot Hasil Jawaban Soal 1 Versi Jetpack Compose



Gambar 5. Screenshot Hasil Jawaban Soal 1 Versi XML

C. Pembahasan

1) Versi Jetpack Compose

MainActivity.kt

Pada baris pertama, terdapat deklarasi package com.example.tipcalculatorcompose yang menandai bahwa file ini berada dalam namespace com.example.tipcalculatorcompose. Ini penting dalam pengelompokan file dan penghindaran konflik nama class antar modul atau library.

Selanjutnya, baris-baris awal kode diisi dengan berbagai import statement yang digunakan untuk memanggil fungsi-fungsi dan komponen UI dari Android dan Jetpack Compose. Beberapa di antaranya adalah androidx.compose.foundation.* untuk elemen layout dasar, androidx.compose.material3.* untuk komponen Material Design versi 3, serta androidx.compose.runtime.* untuk mengelola state dalam Compose. Selain itu,

terdapat juga import dari java.text.NumberFormat dan java.util.Locale yang digunakan untuk memformat hasil tip menjadi format mata uang.

Kelas utama aplikasi didefinisikan dengan class MainActivity : ComponentActivity() yang menandakan bahwa MainActivity adalah sebuah activity berbasis ComponentActivity. Di dalam metode onCreate, fungsi enableEdgeToEdge() dipanggil agar konten dapat ditampilkan penuh hingga ke tepi layar. Setelah itu, setContent digunakan untuk menampilkan UI dengan memanggil TipTimeLayout() di dalam tema TipCalculatorComposeTheme, dan seluruh UI dibungkus dalam Surface dengan ukuran memenuhi layar.

Fungsi TipTimeLayout() adalah fungsi @Composable yang berisi keseluruhan tampilan kalkulator tip. Fungsi ini ditandai dengan anotasi @OptIn(ExperimentalMaterial3Api::class) karena menggunakan komponen eksperimental seperti ExposedDropdownMenuBox. Di dalam fungsi ini, dibuat beberapa state: amountInput (menyimpan input jumlah tagihan dalam bentuk string), expanded (menyimpan status apakah dropdown menu terbuka atau tidak), dan selectedOption (menyimpan pilihan persen tip yang aktif). Nilai input dari pengguna dikonversi ke Double menggunakan toDoubleOrNull(), dan apabila gagal maka nilai default 0.0 akan digunakan. Nilai persen tip diambil dari selectedOption, dan roundUp adalah boolean yang menentukan apakah hasil tip perlu dibulatkan ke atas. Hasil akhir tip dihitung dengan memanggil calculateTip() dan disimpan dalam tipAmount.

Tampilan UI utama menggunakan Column, yaitu layout vertikal, dengan properti horizontalAlignment diatur ke tengah dan verticalArrangement ke tengah juga. Komponen pertama di dalam kolom ini adalah Text yang menampilkan judul "Calculate Tip". Komponen ini diberi padding bawah dan disejajarkan ke kiri.

Setelah judul, terdapat input field yang dibangun dengan memanggil fungsi EditNumberField. Field ini menerima input angka dari pengguna sebagai jumlah tagihan, dilengkapi dengan label dan ikon uang di bagian kiri. Field ini hanya menerima satu baris teks, dengan jenis keyboard angka (KeyboardType.Number) dan aksi keyboard berikutnya (ImeAction.Next).

Komponen berikutnya adalah ExposedDropdownMenuBox, sebuah komponen untuk menampilkan dropdown menu Material3. Di dalamnya terdapat TextField yang hanya bisa dibaca (readOnly), menampilkan persen tip yang sedang dipilih. Bagian ikon kiri menampilkan ikon persen, dan bagian kanan menampilkan ikon panah dropdown. Saat dropdown terbuka (expanded == true), ditampilkan daftar opsi persen tip yang telah ditentukan (15%, 18%, dan 20%). Saat pengguna memilih salah satu, selectedOption akan diperbarui, dan dropdown akan tertutup.

Setelah dropdown, ditampilkan baris pengaturan untuk round up tip menggunakan fungsi RoundTheTipRow. Baris ini menampilkan teks "Round up tip?" di kiri dan Switch di

kanan. Status Switch mengikuti nilai roundUp dan memperbarui nilai tersebut saat pengguna mengubahnya.

Terakhir, hasil perhitungan tip ditampilkan dengan Text besar dan tebal menggunakan fontSize 27.sp dan fontWeight = FontWeight.Bold. Nilai ini merupakan hasil format dari calculateTip().

Fungsi EditNumberField adalah @Composable yang menerima string input, label, ikon, dan fungsi callback untuk perubahan nilai. Fungsi ini digunakan untuk membangun TextField yang dapat digunakan kembali di tempat lain jika dibutuhkan. Leading icon disisipkan menggunakan painterResource dengan ukuran 20.dp.

Fungsi RoundTheTipRow juga merupakan @Composable yang menampilkan baris dengan teks dan Switch. Komponen ini disusun horizontal dengan Row, di mana teks di sisi kiri dan switch di sisi kanan. Tekan switch akan memicu onRoundUpChanged untuk memperbarui state roundUp.

Fungsi calculateTip adalah fungsi non-komposabel biasa yang menerima jumlah tagihan, persen tip, dan opsi pembulatan. Perhitungan dilakukan dengan mengalikan jumlah tagihan dan persen, lalu membulatkannya ke atas jika roundUp bernilai true menggunakan kotlin.math.ceil(). Hasil akhirnya diformat ke mata uang menggunakan NumberFormat.getCurrencyInstance(Locale.US).

Terakhir, terdapat fungsi TipCalculatorPreview yang menggunakan anotasi @Preview. Fungsi ini membuat pratinjau tampilan aplikasi dengan ukuran layar menyerupai Redmi Note 13 (360x800 dp). Ini sangat membantu dalam proses desain UI agar dapat terlihat seperti tampil di perangkat nyata.

2) Versi XML

MainActivity.kt:

Kode dimulai dengan deklarasi package com.example.tipcalculatorxml, yang menunjukkan bahwa file ini termasuk dalam package bernama com.example.tipcalculatorxml. Ini merupakan cara umum dalam Android untuk mengorganisasi file sumber dan menghindari konflik nama antar class.

Setelah itu, berbagai import statement digunakan untuk memanggil class dan fungsi penting dari Android dan Kotlin. android.os.Bundle digunakan dalam siklus hidup activity. android.widget.ArrayAdapter diperlukan untuk menyediakan daftar opsi pada dropdown. androidx.appcompat.app.AppCompatActivity adalah superclass untuk activity yang menggunakan kompatibilitas fitur lama pada Android. androidx.core.widget.addTextChangedListener menyediakan ekstensi Kotlin untuk mendengarkan perubahan teks secara lebih sederhana.

com.example.tipcalculatorxml.databinding.ActivityMainBinding digunakan untuk mengakses elemen-elemen UI yang didefinisikan di file XML melalui View Binding. Terakhir, kotlin.math.ceil digunakan untuk melakukan pembulatan ke atas pada hasil tip.

Di dalam kelas MainActivity, terdapat deklarasi private lateinit var binding: ActivityMainBinding. Ini adalah deklarasi late-initialized property yang memungkinkan kita untuk menginisialisasi binding nanti di onCreate, namun tetap bisa mengaksesnya sebagai properti.

Fungsi onCreate adalah titik masuk utama ketika activity dijalankan. Di dalamnya, binding diinisialisasi dengan memanggil ActivityMainBinding.inflate(layoutInflater), yang akan mengikat elemen UI dari layout XML ke dalam objek Kotlin. Kemudian setContentView(binding.root) akan menampilkan layout tersebut sebagai tampilan activity.

Selanjutnya, dibuat sebuah daftar string berisi pilihan persen tip, yaitu "15%", "18%", dan "20%". Daftar ini kemudian digunakan untuk membuat ArrayAdapter, yang berfungsi sebagai adapter untuk dropdown AutoCompleteTextView. Adapter ini menggunakan layout R.layout.list_items sebagai tampilan dari masing-masing item dalam daftar. Dropdown (binding.dropdownField) kemudian diatur agar menggunakan adapter ini, dan secara default menampilkan pilihan pertama ("15%"). Label tip (binding.tipAmount) diatur agar menampilkan "Tip amount: $0.0" sebagai nilai awal.

Sebuah fungsi lokal bernama updateTip(isRoundUp: Boolean) didefinisikan di dalam onCreate. Fungsi ini bertanggung jawab menghitung dan menampilkan tip berdasarkan input pengguna. Di dalam fungsi ini, teks dari billAmountInput dikonversi ke Double, jika gagal maka diasumsikan 0.0. Demikian juga dengan nilai persen tip yang diambil dari teks dropdown, dengan karakter % dibuang sebelum dikonversi menjadi angka desimal. Kemudian dilakukan perhitungan tip: billAmount * tipPercent / 100. Jika switch round up dalam keadaan aktif (isRoundUp == true), maka nilai tip dibulatkan ke atas dengan fungsi ceil. Terakhir, hasil tip diformat dan ditampilkan di binding.tipAmount menggunakan getString(R.string.tip_amount, tipAmount) yang akan menghasilkan string seperti "Tip amount: $2.0".

Untuk merespons perubahan pengguna, listener ditambahkan pada tiga komponen UI. Pertama, addTextChangedListener dipasang ke billAmountInput sehingga setiap perubahan angka tagihan akan menghitung ulang tip. Kedua, dropdownField juga memiliki listener yang akan memicu perhitungan ulang saat persen tip diubah. Ketiga, setOnCheckedChangeListener digunakan pada switch switchRoundUp, yang akan memanggil updateTip dengan nilai boolean baru setiap kali switch diubah.

Akhirnya, sebelum activity selesai diinisialisasi, updateTip(false) dipanggil untuk pertama kalinya agar hasil tip langsung dihitung berdasarkan kondisi awal aplikasi, tanpa harus menunggu interaksi pengguna.

activity_main.xml:

File XML ini menggunakan ConstraintLayout sebagai root layout, yaitu sebuah jenis layout yang fleksibel dan efisien dalam menyusun elemen UI dengan membatasi posisi relatif terhadap elemen lain. Root ConstraintLayout ini memiliki atribut lebar dan tinggi match_parent, yang berarti akan mengisi seluruh ruang yang tersedia di layar. Padding sebesar 40dp ditambahkan di sekeliling layout agar isi tidak menempel ke tepi layar. Atribut tools:context=".MainActivity" memberitahu Android Studio bahwa layout ini terkait dengan MainActivity.

Di dalam root layout, terdapat lagi sebuah ConstraintLayout yang menjadi pembungkus semua elemen UI utama. Layout ini diatur agar berada tepat di tengah parent-nya, dengan mengaitkan semua sisi (Top, Bottom, Start, dan End) ke sisi parent, dan lebar diatur 0dp agar menyesuaikan dengan constraint (match constraints).

Elemen pertama di dalam layout utama adalah sebuah TextView dengan ID calc_tip. Teks ini digunakan sebagai judul atau label utama dan menampilkan string dari resource @string/calc_tip. Lebarnya 0dp (menyesuaikan constraint), dengan tinggi wrap_content, dan teksnya berukuran 18sp.

Selanjutnya, ada sebuah TextInputLayout dengan ID bill_amount yang digunakan untuk membungkus kolom input nominal tagihan. Elemen ini menampilkan ikon uang (@drawable/money_resized) sebagai ikon awal dan memiliki hint yang berasal dari @string/bill_amount. Di dalamnya terdapat TextInputEditText dengan ID bill_amount_input, yang menerima input angka desimal (numberDecimal). Elemen ini penting karena menjadi dasar perhitungan tip nantinya.

Di bawah input nominal, terdapat TextInputLayout lain dengan ID dropdown_layout, yang berfungsi menampilkan pilihan persen tip. Layout ini menggunakan style OutlinedBox.ExposedDropdownMenu, yang secara otomatis membuat input terlihat seperti dropdown. Ikon awalnya adalah ikon persen (@drawable/percent_resized) dan hint-nya berasal dari @string/tip_percentage. Di dalamnya terdapat AutoCompleteTextView dengan ID dropdown_field, yang akan menampilkan daftar opsi persen tip yang bisa dipilih pengguna.

Setelah itu, terdapat ConstraintLayout bernama roundUpLayout, yang digunakan untuk meletakkan pengaturan pembulatan tip. Di dalamnya terdapat TextView dengan ID roundUpTip yang menampilkan label teks "Round up tip" dari string resource. Kemudian ada elemen View dengan ID spacer yang digunakan sebagai pemisah fleksibel untuk mendorong SwitchMaterial ke sisi kanan layout. Elemen SwitchMaterial dengan ID switch_round_up memungkinkan pengguna memilih apakah hasil tip akan dibulatkan ke atas atau tidak.

Terakhir, elemen TextView dengan ID tip_amount digunakan untuk menampilkan hasil perhitungan tip. Lebarnya 0dp agar mengisi ruang yang tersedia, tingginya wrap_content,

dan posisinya berada di bawah pengaturan pembulatan. Teks ini ditampilkan dengan ukuran besar 30sp, bold, dan rata tengah untuk menonjolkan hasil yang dihitung.

list_items.xml

File XML ini mendefinisikan sebuah elemen TextView yang memiliki ID textViewItem. Elemen TextView ini digunakan untuk menampilkan teks dalam suatu list item, yang kemudian bisa digunakan dalam elemen UI seperti AutoCompleteTextView atau ListView.

Atribut android:layout_width="match_parent" berarti lebar TextView akan mengisi seluruh lebar ruang yang tersedia dalam layout parent-nya. Atribut android:layout_height="wrap_content" mengindikasikan bahwa tinggi elemen ini akan menyesuaikan dengan konten teks yang ditampilkannya, sehingga hanya sebesar teks yang ditampilkan.

Atribut android:padding="16dp" memberikan jarak antara konten teks dan batas elemen TextView, memberikan ruang agar teks tidak menempel pada sisi elemen. Padding ini digunakan untuk membuat tampilan lebih rapi dan nyaman dibaca.

Atribut android:textSize="18sp" mengatur ukuran teks yang ditampilkan pada TextView. Ukuran font ini menggunakan satuan sp (scale-independent pixels), yang lebih fleksibel karena dapat menyesuaikan dengan preferensi ukuran teks yang diatur oleh pengguna pada perangkat mereka.

D. Tautan Git

Berikut adalah tautan untuk source code yang telah dibuat.
Versi Jetpack Compose:
https://github.com/raihan2030/Praktikum-Pemrograman-Mobile/tree/main/Praktikum-2/Compose
Versi XML:
https://github.com/raihan2030/Praktikum-Pemrograman-Mobile/tree/main/Praktikum-2/XML

SOAL 2

Jelaskan perbedaan dari implementasi XML dan Jetpack Compose beserta kelebihan dan kekurangan dari masing-masing implementasi.

Jawab:

1. Perbedaan Implementasi

Pada implementasi XML, struktur UI didefinisikan terlebih dahulu di file .xml, kemudian dikaitkan dengan logika dalam file Kotlin melalui View Binding. Misalnya, elemen-elemen seperti TextInputEditText, AutoCompleteTextView, SwitchMaterial, dan TextView ditulis dalam activity_main.xml, lalu diakses di MainActivity.kt menggunakan binding.namaView.

Sedangkan dalam Jetpack Compose, UI dan logika digabungkan menjadi satu alur deklaratif. Seluruh komponen UI—seperti inputan jumlah tagihan, dropdown persentase tip, switch pembulatan, dan hasil tip—langsung dibuat menggunakan fungsi-fungsi Kotlin seperti TextField, ExposedDropdownMenuBox, Switch, dan Text. Tidak ada file layout terpisah karena semua didefinisikan dalam kode Kotlin secara deklaratif.

2. Kelebihan dan Kekurangan

XML (View-based UI)

Kelebihan:

- Lebih familiar bagi developer Android lama karena sudah digunakan sejak awal Android.
- Visual Editor tersedia di Android Studio, memudahkan dalam menyusun UI secara drag-and-drop.
- Struktur lebih terpisah antara UI dan logic, yang bisa dianggap lebih rapi oleh beberapa developer.

Kekurangan:

- Membutuhkan boilerplate lebih banyak, seperti binding dan pengaturan listener manual.
- UI tidak reaktif secara otomatis terhadap perubahan state; semua pembaruan seperti perhitungan ulang tip harus dipanggil eksplisit pada setiap listener.
- Dropdown dan Switch butuh penanganan manual pada state-nya, tidak sekonsisten Compose dalam hal pengelolaan state.