LAPORAN PRAKTIKUM
PEMROGRAMAN MOBILE
MODUL 1

ANDROID BASIC WITH KOTLIN

Oleh:

Muhammad Daffa Musyafa NIM. 2310817110007

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
APRIL 2025

LEMBAR PENGESAHAN
LAPORAN PRAKTIKUM PEMROGRAMAN I
MODUL 1

Laporan Praktikum Pemrograman Mobile Modul 1: Android Basic with Kotlin ini disusun sebagai syarat lulus mata kuliah Praktikum Pemrograman Mobile. Laporan Prakitkum ini dikerjakan oleh:

Nama Praktikan : Muhammad Daffa Musyafa
NIM : 2310817210007

Menyetujui, Asisten Praktikum	Mengetahui, Dosen Penanggung Jawab Praktikum
Salsabila Syifa NIM. 2010817320004	Andreyan Rizky Baskara, S.Kom., M.Kom. NIP. 19930703 201903 01 011

DAFTAR ISI

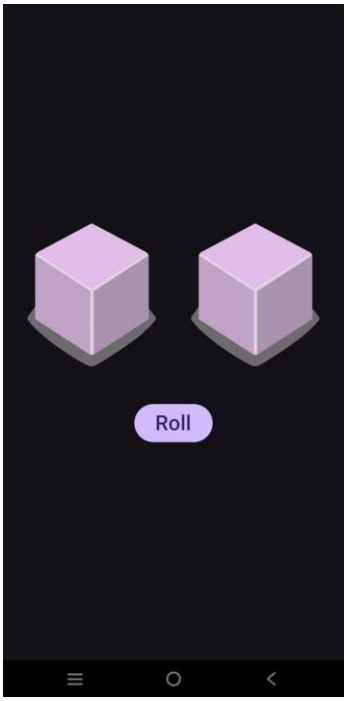
DAFTAR GAMBAR

DAFTAR TABEL

SOAL 1

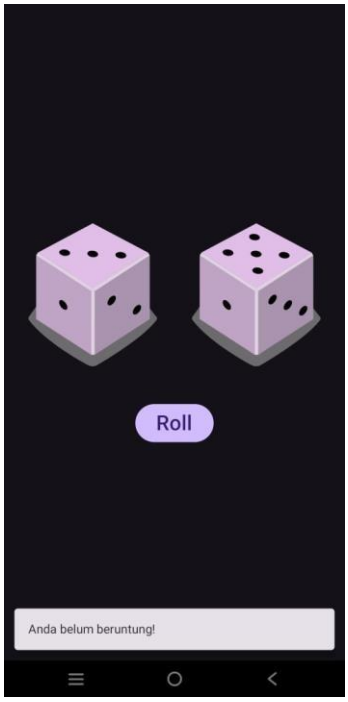
Soal Praktikum:
Buatlah sebuah aplikasi yang dapat menampilkan 2 buah dadu yang dapat berubah-ubah tampilannya pada saat user menekan tombol “Roll”. Aturan aplikasi yang akan dibangun adalah sebagaimana berikut:

1. Tampilan awal aplikasi setelah dijalankan akan menampilkan 2 buah dadu kosong seperti dapat dilihat pada Gambar 1.



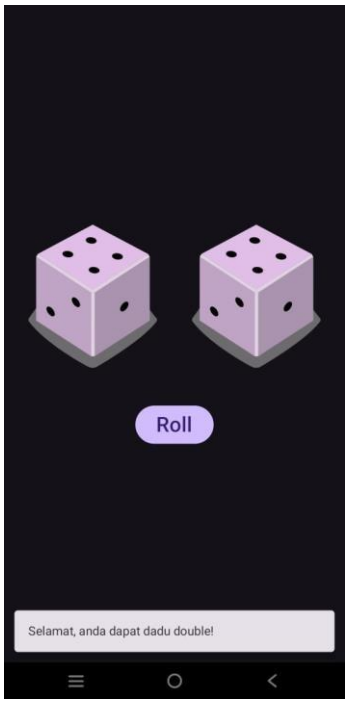
Gambar 1. Tampilan Awal Aplikasi

2. Setelah user menekan tombol “Roll” maka masing-masing dadu akan memperlihatkan sisi dadunya dengan angka antara 1 s/d 6. Apabila user mendapatkan nilai dadu yang berbeda antara Dadu 1 dengan Dadu 2, maka aplikasi akan menampilkan pesan “Anda belum beruntung!” seperti yang dapat dilihat pada Gambar 2.



Gambar 2. Tampilan Dadu Setelah Di-Roll

3. Apabila user mendapatkan nilai dadu yang sama antara Dadu 1 dan Dadu 2 atau nilai double, maka aplikasi akan menampilkan pesan “Selamat, anda dapat dadu double!” seperti yang dapat dilihat pada Gambar 3.



Gambar 3. Tampilan Roll Dadu Double

4. Buatlah aplikasi tersebut menggunakan XML dan Jetpack Compose.
5. Upload aplikasi yang telah anda buat ke dalam repository GitHub ke dalam **folder Modul 1 dalam bentuk Project.** Jangan lupa untuk melakukan **Clean Project** sebelum mengupload pekerjaan anda pada repository.

6. Untuk gambar dadu dapat didownload pada link berikut: https://drive.google.com/file/d/14V3qXGdFnuoYN4AGd_9SgFh8kw8X9ySm/view?usp= sharing

A. Source Code XML

MainActivity.kt

Table 1. Source Code Jawaban soal 1 XML

```
1  package com.example.diceroller
2
3  import android.os.Bundle
4  import android.widget.Button
5  import android.widget.ImageView
6  import android.widget.Toast
7  import androidx.activity.enableEdgeToEdge
8  import androidx.appcompat.app.AppCompatActivity
9
10
11 class MainActivity : AppCompatActivity() {
12     private lateinit var rollbtn: Button
13     private lateinit var dice: ImageView
14     private lateinit var dice2: ImageView
15     override fun onCreate(savedInstanceState: Bundle?) {
16         super.onCreate(savedInstanceState)
17         enableEdgeToEdge()
18         setContentView(R.layout.activity_main)
19
20         dice = findViewById(R.id.dice_image1)
21         dice2 = findViewById(R.id.dice_image2)
22
23         rollbtn = findViewById(R.id.roll_button)
24
25         rollbtn.setOnClickListener {
26             rollDice()
27         }
28     }
29     private fun rollDice() {
30
31         val randomInt1 = (1..6).random()
32         val drawableResource1 = when (randomInt1) {
33             1 -> R.drawable.dice_1
34             2 -> R.drawable.dice_2
35             3 -> R.drawable.dice_3
36             4 -> R.drawable.dice_4
```

```
            5                ->                R.drawable.dice_5
            6                ->                R.drawable.dice_6
            else              ->               R.drawable.dice_0
        }
        val         randomInt2         =         (1..6).random()
        val   drawableResource2    =    when    (randomInt2)    {
            1                ->                R.drawable.dice_1
            2                ->                R.drawable.dice_2
            3                ->                R.drawable.dice_3
            4                ->                R.drawable.dice_4
            5                ->                R.drawable.dice_5
            6                ->                R.drawable.dice_6
            else              ->               R.drawable.dice_0
        }
        dice.setImageResource(drawableResource1)
        dice2.setImageResource(drawableResource2)

        if       (randomInt1       ==       randomInt2)       {
            Toast.makeText(this, "Selamat Kamu dapat Double!",
Toast.LENGTH_SHORT).show()
        }                          else                          {
            Toast.makeText(this,   "Anda   Kurang   Beruntung",
Toast.LENGTH_SHORT).show()
        }
    }}

```

activity_main.xml

Table 2. Source Code Jawaban soal 1 XML

```
<?xml              version="1.0"              encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <ImageView
        android:id="@+id/dice_image1"
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:layout_marginStart="25dp"
        android:src="@drawable/dice_0"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        />

    <ImageView
```

```
23          android:id="@+id/dice_image2"
24          android:layout_width="200dp"
25          android:layout_height="200dp"
26          android:layout_marginEnd="25dp"
27          android:src="@drawable/dice_0"
28          app:layout_constraintBottom_toBottomOf="parent"
29          app:layout_constraintEnd_toEndOf="parent"
30          app:layout_constraintTop_toTopOf="parent"
31          />
32
33      <Button
34          android:id="@+id/roll_button"
35          android:layout_width="wrap_content"
36          android:layout_height="wrap_content"
37          app:layout_constraintBottom_toBottomOf="parent"
38          app:layout_constraintEnd_toEndOf="parent"
39          app:layout_constraintHorizontal_bias="0.498"
40          app:layout_constraintStart_toStartOf="parent"
41          app:layout_constraintTop_toTopOf="parent"
42          android:layout_marginTop="250dp"
43          android:text="@string/roll"
44          android:textSize="20sp"/>
45  </androidx.constraintlayout.widget.ConstraintLayout>
```

Tabel 2. Source Code Jawaban Soal 1

B. Output Program

Gambar 4. Screenshot Hasil Jawaban Soal 1 XML

Gambar 5. Screenshot Hasil Jawaban Soal 1 XML

C. Pembahasan

MainActivity.kt:

Pada line 1, dideklarasikan nama package file Kotlin
Pada line 3-8, Mengimport komponen komponen UI yang dibutuhkan untuk mendefinisikan dalam XML.

Pada line 10-27, Class MainActivity yang merupakan turunan dari AppCompatActivity yang merupakan kelas bawaan android yang memberi dukungan untuk fitur-fitur yang kompotibel dengan banyak versi android. Kemudian mendeklarasikan variable `private lateinit var rollbtn: Button` untuk tombol dadu, `private lateinit var dice: ImageView` dan `private lateinit var dice2: ImageView` untuk gambarnya dengan nama variabel dice dan dice2, `lateinit` berarti variable ini akan diinisialisasikan nanti biasanya di dalam `onCreate`. Kemudian `override fun onCreate(savedInstanceState: Bundle?)` { adalah fungsi *lifecycle*, untuk dipanggil saat activity pertama kali dibuat dan sebagai tempat untuk mengatur layout dan inisialisasi komponen UI. Kemudian `enableEdgeToEdge()` untuk tampilan *fullscreen* `setContentView(R.layout.activity_main)` untuk mengatur layout dari file *activity_main.xml*.Kemudian menghubungkan varibel *dice* dan *dice2* tadi ke file XML sebagai gambarnya dengan menggunakan `findViewById(R.id.dice_image1)` dan `findViewById(R.id.dice_image2)`, *findViewById()* untuk mencari komponen dari file XML berdasarkan ID-nya, id dice adalah *dice_image1* dan *dice_image2*. Kemudian

menghubungkan variabel *rollbtn* tadi sebagai *button* dengan cara yang sama menggunakan *findViewById()* dengan mencari ID di XMLnya *roll_button*. Kemudian `rollbtn.setOnClickListener` menambahkan listener untuk tombol *rollbtn* tadi, kemudian *rollDice()* adalah fungsi untuk tombol tadi di pencet maka menjalankan fungsi *rollDice()*.
Pada line 28-58, membuat fungsi `private fun rollDice()` untuk roll dadunya, lalu membuat `val randomInt1 = (1..6).random()` untuk men-acak antara 1-6 dadu yang *dice*, kemudian membuat variabel `val drawableResource1 = when (randomInt1) {`, when untuk memilih gambar dadu dari file XML berdasarakn angka acak, seperti switch di java, kemudian membuat meninsialisasinya dengan `1 -> R.drawable.dice_1 2-> R.drawable.dice_2 3-> R.drawable.dice_3 4-> R.drawable.dice_4 5-> R.drawable.dice_5 6-> R.drawable.dice_6 else -> R.drawable.dice_0`, dari angka 1 kemudian mencari dengan resource kemudian mengambil di drawable dengan nama file dice_1 seperti ini `R.drawable.dice_1`, dan selanjutnya sampai 6, else untuk menampilakan dadu di awal sebelum acak yaitu dadu kosong. Kemudian `val drawableResource2 = when (randomInt2` sama seperti yang sebelumnya untuk dadu yang kedua.Kemudian dice.setImageResource(drawableResource1) dan dice2.setImageResource(drawableResource2) untuk menampilkan gambar 2 dadu di layar nanti. Kemudian membuat kondisi menggunakan if, `if (randomInt1 == randomInt2)` jika nilai dari *randomInt1* sama dengan *randomInt2*, maka akan menampilkan *toast* berisi *"Selamat Kamu dapat Double!"*, jika tidak sama maka akan menampilkan toast *"Anda Kurang Beruntung"*, dan masing masing pesan akan di tampilkan dengan waktu yang sebentar, *toast* adalah popup kecil dibawah layar untuk memberi informasi sementara.

activity_main.xml:

Pada line 1, standar deklarasi file XML `<?xml version="1.0" encoding="utf-8"?>`
Pada line 2-8, `<androidx.constraintlayout.widget.ConstraintLayout` untuk membuat layout secara fleksibel untuk menempatkan elemen bedasarkan posisi elemen lainnya. Kemudian pada baris yang 3 dan 4 itu bawaan atau *default* dari XMLnya, kemudian layout lebar dan tingginya menggunakan *match_parent* yaitu berarti mengikuti ukuran layar penuh `android:layout_width="match_parent"` dan `android:layout_height="match_parent"`, kemudian `tools:content=".MainActivity"` ini menunjukkan kalau XMLnya di gunakan oleh class MainActivity.

Pada line 10-19, untuk membuat *ImageView* untuk menampilkan gambar, kemudian `android:id="@+id/dice_image1"` untuk memberikan id pada *imageview* tadi dengan nama *dice_image1*, `android:layout_width="200dp"` dan `android:layout_height="200dp"` untuk menentukan ukuran dadu yaitu *200dp*, `android:layout_marginStart="25dp"` untuk mengasih jarak gambar dari kiri

layar. Kemudian `android:src="@drawable/dice_0"` untuk gambar tampilan awal yang ditampilkan. Kemudian `app:layout_constraintBottom_toBottomOf="parent"`, `app:layout_constraintStart_toStartOf="parent"` dan `app:layout_constraintTop_toTopOf="parent"`, memberi constrain atau jangkar untuk mengikat posisi gambar supaya tetap pada tempatnya, untuk yang `app:layout_constraintBottom_toBottomOf="parent"` dari bawah, `app:layout_constraintStart_toStartOf="parent"` dari kiri dan `app:layout_constraintTop_toTopOf="parent"` dari atas.

Pada line 21-30, sama membuat *ImageView* untuk gambar kedua namu ada sedikit berbeda pada bagian constraintnya, untuk gambar ini menggunkana constraint `app:layout_constraintEnd_toEndOf="parent"`, yaitu membuat gambar dari sebelah kanan dan juga `android:layout_marginEnd="25dp"` untuk membuat jarak dari sisi kanan.

Pada line 32-43, Membuat *Button*, pertama membuat *id* untuk buttonnya agar bisa di akses di *MainActivity* `android:id="@+id/roll_button"` dengan ini id dari buttonnnya adalah *roll_button*. Kemudian `android:layout_width="wrap_content"` dan `android:layout_height="wrap_content"` untuk lebar tombol mengikuti isi teksnya. `Kemudian android:layout_marginTop="250dp"` untuk memberi jarak dari atas supaya tombol dibawah gambar. Kemudian membuat text dengan `android:text="@string/roll"` dengan isi teksnya roll mengambil dari teks file strings.xml. Kemudian `android:textSize="20sp"` untuk text sizenya ukuran 20sp. Kemudian membuat jangkar atau constrain, untuk button ini menggunakan semua sisi `app:layout_constraintBottom_toBottomOf="parent"` untuk bagian bawah, `app:layout_constraintEnd_toEndOf="parent"` untuk bagian kanan, `app:layout_constraintStart_toStartOf="parent"` untuk kiri ,`app:layout_constraintTop_toTopOf="parent"` untuk bagian atas. Kemudian `/androidx.constraintlayout.widget.ConstraintLayout>` adalah tag penutup.

D. Source Code Compose

MainActivity.kt

Table 3 Source Code Jawaban soal 1 Compose

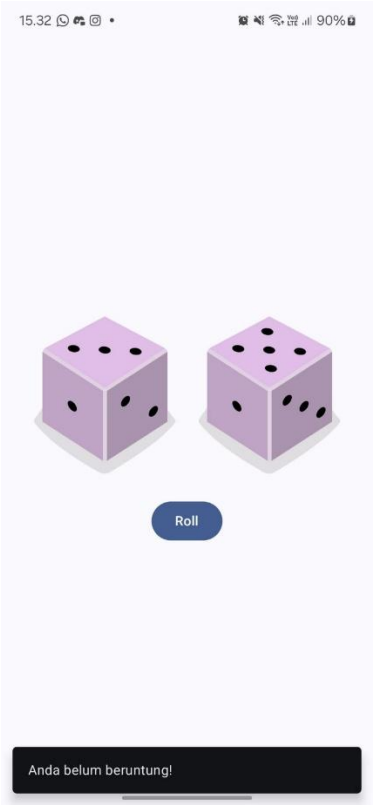
```
1  package com.example.diceroller
2
3  import android.os.Bundle
4  import androidx.activity.ComponentActivity
5  import androidx.activity.compose.setContent
6  import androidx.activity.enableEdgeToEdge
7  import androidx.compose.foundation.Image
8  import androidx.compose.foundation.layout.Arrangement
```

```
9  import androidx.compose.foundation.layout.Column
10 import androidx.compose.foundation.layout.Row
11 import androidx.compose.foundation.layout.Spacer
12 import androidx.compose.foundation.layout.fillMaxSize
13 import androidx.compose.foundation.layout.padding
14 import androidx.compose.foundation.layout.width
15 import androidx.compose.foundation.layout.wrapContentSize
16 import androidx.compose.material3.Button
17 import androidx.compose.material3.Scaffold
18 import androidx.compose.material3.SnackbarDuration
19 import androidx.compose.material3.SnackbarHost
20 import androidx.compose.material3.SnackbarHostState
21 import androidx.compose.material3.Text
22 import androidx.compose.runtime.Composable
23 import androidx.compose.runtime.getValue
24 import androidx.compose.runtime.mutableStateOf
25 import androidx.compose.runtime.remember
26 import androidx.compose.runtime.rememberCoroutineScope
27 import androidx.compose.runtime.setValue
28 import androidx.compose.ui.Alignment
29 import androidx.compose.ui.Modifier
30 import androidx.compose.ui.res.painterResource
31 import androidx.compose.ui.res.stringResource
32 import androidx.compose.ui.tooling.preview.Preview
33 import androidx.compose.ui.unit.dp
34 import com.example.diceroller.ui.theme.DiceRollerTheme
35 import kotlinx.coroutines.launch
36
37 class MainActivity : ComponentActivity() {
38     override fun onCreate(savedInstanceState: Bundle?) {
39         super.onCreate(savedInstanceState)
40         enableEdgeToEdge()
41         setContent {
42             DiceRollerTheme {
43                 DiceRollerApp()
44             }
45         }
46     }
47
48     @Composable
49     fun DiceWithButtonAndImage(
50         modifier: Modifier,
51         showSnackbar: (String) -> Unit
52     ) {
53         var result1 by remember { mutableStateOf(0) }
54         val imageResource1 = when (result1) {
55             1 -> R.drawable.dice_1
56             2 -> R.drawable.dice_2
57             3 -> R.drawable.dice_3
58             4 -> R.drawable.dice_4
59             5 -> R.drawable.dice_5
60             6 -> R.drawable.dice_6
```

```
61              else              ->              R.drawable.dice_0
62          }
63          var  result2  by  remember  {  mutableStateOf(0)  }
64          val    imageResource2    =    when    (result2)    {
65              1                 ->              R.drawable.dice_1
66              2                 ->              R.drawable.dice_2
67              3                 ->              R.drawable.dice_3
68              4                 ->              R.drawable.dice_4
69              5                 ->              R.drawable.dice_5
70              6                 ->              R.drawable.dice_6
71              else              ->              R.drawable.dice_0
72          }
73          Column(
74              modifier                =                modifier,
75              horizontalAlignment = Alignment.CenterHorizontally
76          )                                                       {
77              Row(
78                  horizontalArrangement   =   Arrangement.Center,
79                  modifier              =              Modifier
80              )                                                   {
81                  Image(
82                      modifier     =     Modifier.width(170.dp),
83                      painter = painterResource(imageResource1),
84                      contentDescription   =   result1.toString()
85                  )
86                  Image(
87                      modifier     =     Modifier.width(170.dp),
88                      painter = painterResource(imageResource2),
89                      contentDescription   =   result2.toString()
90                  )
91              }
92
93              Button(onClick                  =                {
94                  result1           =           (1..6).random()
95                  result2           =           (1..6).random()
96
97                  val  message  =  if  (result1  ==  result2)  {
98                      "Selamat   anda   dapat   dadu   double!"
99                  }                    else                    {
100                     "Anda          belum          beruntung!"
101                 }
102
103                 showSnackbar(message)
104             })                                                  {
105                 Text("Roll")
106             }
107         }
108
109     }
110
111     @Preview(showBackground             =              true)
112     @Composable
```

```
112     fun                    DiceRollerApp()                    {
113         val snackbarHostState = remember { SnackbarHostState()
114 }
115         val    corutineScope    =    rememberCoroutineScope()
116         Scaffold(
117             snackbarHost = { SnackbarHost(snackbarHostState) }
118         )            {            paddingValues            ->
119             DiceWithButtonAndImage(
121                 modifier              =              Modifier
122                     .fillMaxSize()
123                     .wrapContentSize(Alignment.Center)
124                     .padding(paddingValues),
125                 showSnackbar      =      {      message      ->
126                     corutineScope.launch                    {
127                         snackbarHostState.showSnackbar(
128                             message         =         message,
129                             duration  =  SnackbarDuration.Short
130                         )
131                     }
132                 }
133             )
134         }
135     }
136 }
```

E. Output Program



Gambar 1. Screenshot Hasil Jawaban Soal 1

F. Pembahasan

MainActivity.kt:

Pada line 1, dideklarasikan nama package file Kotlin
Pada line 3-35, Mengimport komponen komponen UI yang dibutuhkan untuk mendefinisikan dalam compose.

Pada line 37-46, ini main activity, `class MainActivity : ComponentActivity()` , untuk ComponentActivity adalah kelas dari jetpack compose untuk tampilan UI dari kotlin langsung. Kemudian `override fun onCreate(savedInstanceState: Bundle?)`, fungsi *oncreate* adalah fungsi awal yang dijalankan saat activity, *override* berarti kamu menggantikan fungsi bawaan dari *ComponenActivity* untuk bisa dicostum sendiri. *enableEdgeToEdge()* untuk ini membuat UI *fullscreen, setContent {* untuk fungsi UI kamu akan di dalam blok ini, kemudian *DiceRollerTheme {* untuk membuat fungsi tema sendiri berguna untuk memberikan warna, font, bentuk dan lain lain, dengan isi *DiceRollerApp()* untuk memuat *Composable function utama.*
Pada line 48-109, `@Composable` ini untuk fungsi UI tampilan di jetpack compose. Kemudian `fun DiceWithButtonAndImage( modifier: Modifier, showSnackbar: (String) -> Unit )` membuat fungsi *DiceWithButtonAndImage* dengan alat bantu buat atur ukuran, posisi, padding, dan lain-lain menggunkan *modifier*, kemudian `showSnackbar` adalah lambda function untuk menampilkan Snackbar, di panggil setelah tombol ditekan. Kemudian membuat variabel var `result1 by remember { mutableStateOf(0) }` dan `var result2 by remember { mutableStateOf(0) }` untuk menyimpan angka acak untuk dadu 1 dan dadu 2, untuk remember + mutableStatof() nilai ynag bisa berubah, dan akan rekomposisi UI kalau nilainya berubah. Kemudian membuat variabel `val imageResource1 = when (result1) { ... }` dan `val imageResource2 = when (result2) { ... }` memilih gambar dadu berdasarkan angka yang muncul di result1 sama 1 sampai 6 dengan gambar masing masing `1 -> R.drawable.dice_1`, jika keluar 1 maka maenampilkan dadu 1. Kemudian pada bagian layout membaut `column` menyusun elemen secara vertikal: gambar + tombol, kemudian membuat columnya rata tengah horizontal dengan `horizontalAlignment = Alignment.CenterHorizontally`. Kemudian membuat row utnuk gambar kedua dengan Row(...) { ... }, Di dalam *Column*, kamu buat *Row* untuk meletakkan 2 dadu secara horizontal dengan posisi dadu dibuat berada di tengah baris.

Kemudian membuat `Image(…)` dibuat 2 untuk menampilkan dadu 1 dan dadu 2, dengan mengambil gambar menggunakan `painterResource(...)`, mengambil dari drawable. Kemudian `modifier = Modifier.width(170.dp)` untuk mengatur lebar gambar agar tidak terlalu over, kemudian `contentDescription = result1.toString()` untuk buat aksesibilitas. Kemudian membuat button dengan `Button(onClick = { ... })`, jadi ketika tombol di klik akan mengacak angka dadu dengan `result1 =`

`(1..6).random()` dan `result2 = (1..6).random()`, kemudian membuat kondisi dengan if `val message = if (result1 == result2)`, jika result1 sama dengan result2 maka akan mengeluarkan "Selamat anda dapat dadu double!" jika beda "Anda belum beruntung!". Kemudian menampikan *snackbar* dengan `showSnackbar(message)`. Kemudian membuat text pada *button* dengan `Text("Roll")` dengan isi Roll.

Pada line 111-135, `@Preview(showBackground = true)` untuk membuat compose bisa ditampilkan di android studio, kemudian `@composable` untuk UI compose. Kemudian membuat variabel `val snackbarHostState = remember { SnackbarHostState() }` dan `val corutineScope = rememberCoroutineScope()` untuk mengontrol snackbar, kapan ditampilkan dan isinya apa, kemudian remember {…} menyimpan state agar tidak reset saat *recomposition*. Kemudian `rememberCoroutineScope()` untuk bikin scope corutine supaya bisa menampilkan *snackbar* secara *asynchronous*. Kemudian membuat `Scaffold( snackbarHost = { SnackbarHost(snackbarHostState) }`, ini adalah layout Compose lengkap, kemudain `snackbarHost = { SnackbarHost(snackbarHostState)` untuk mengubungkan *snackbarhoststate* supaya compose tahu di mana menampilkan *snackbar*. Kemudian membuat isi dari Scaffold dengan `paddingValues` Otomatis diberikan oleh *Scaffold* untuk menghindari ketabrak status bar navigation. Kemudian isi dari `DiceWithButtonAndImage( Modifier.fillMaxSize()` untuk ukuran yang seluar layar, kemudain `wrapContentSize(Alignment.Center)` posisi konten di tengah layar, kemudian . `padding(paddingValues)` untuk kasih padding bawaan scaffold. Kemudian `showSnackbar` untuk menampilkan isi dari snackbar dan durasinya pendek dengan `SnackbarDuration.Short`.

Tautan Git
Berikut adalah tautan untuk source code yang telah dibuat.
https://github.com/Easydaf/Praktikum_Mobile