

Nama : M Rezky Raya Kilwouw

NIM : 222313190

Kelas : 3SI1

PRAKTIKUM 9

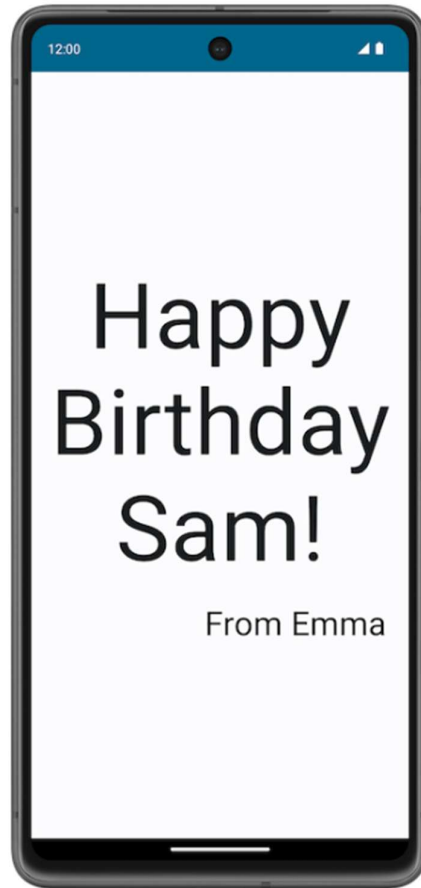
## PEMROGRAMAN PLATFORM KHUSUS

1. Ikuti dan selesaikan tutorial pada link berikut <https://developer.android.com/codelabs/basic-android-kotlin-compose-text-composables> kemudian claim learning badge

1. Before you begin

Menjelaskan bahwa dalam codelab ini Akan menggunakan Jetpack Compose untuk membuat aplikasi Android sederhana yang menampilkan pesan Happy Birthday di layar.

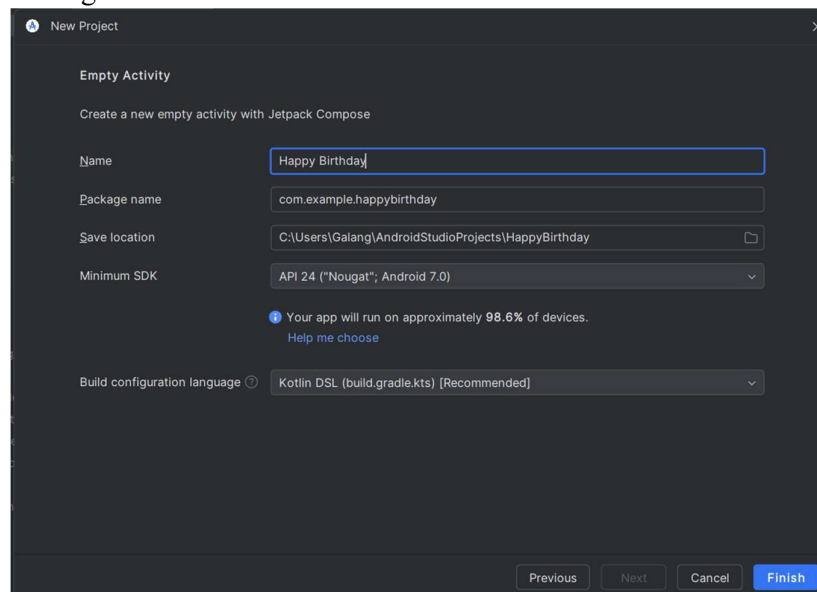
- Prasyarat
  - Cara membuat aplikasi di Android Studio.
  - Cara menjalankan aplikasi di emulator atau perangkat Android.
- Yang akan dipelajari
  - Cara menulis fungsi composable, seperti fungsi composable Text, Column, dan Row.
  - Cara menampilkan teks di aplikasi dalam tata letak.
  - Cara memformat teks, seperti mengubah ukuran teks.
- Yang akan dibangun
  - Aplikasi Android yang menampilkan ucapan ulang tahun dalam format teks yang akan terlihat seperti screenshot ini jika sudah selesai:



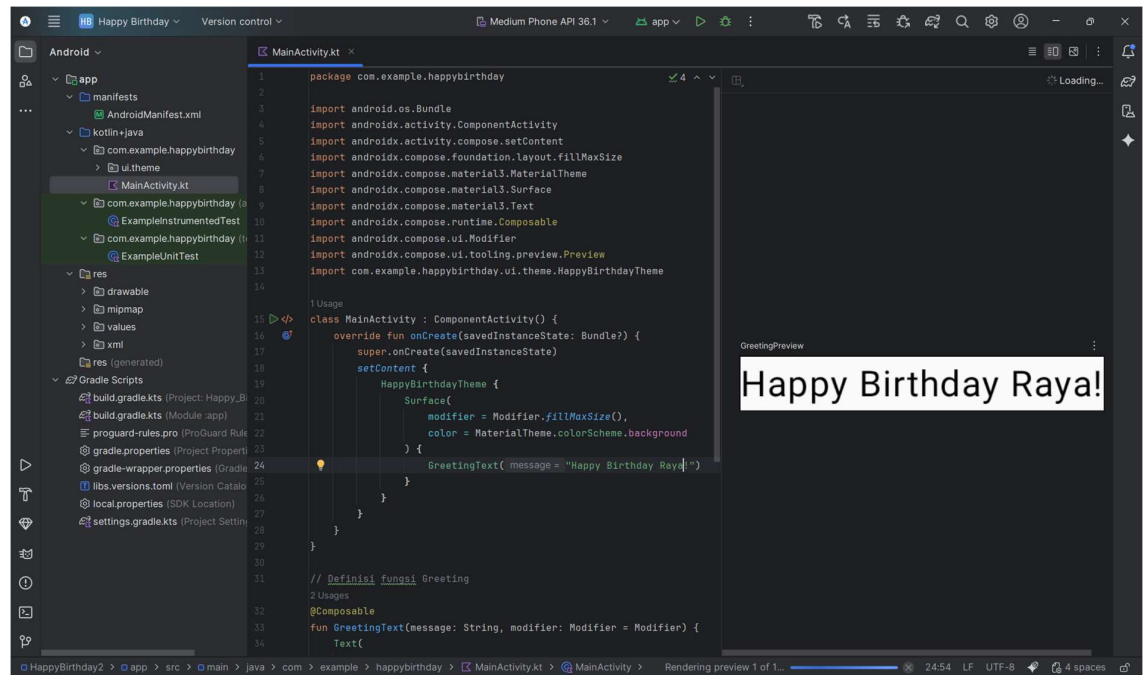
- Yang akan dibutuhkan
  - Komputer yang dilengkapi Android Studio

## 2. Set up a Happy Birthday app

- Konfigurasi awal



- Output



### 3. What is Jetpack Compose?

Jetpack Compose adalah toolkit modern untuk membuat UI Android. Compose menyederhanakan dan mempercepat pengembangan UI di Android dengan lebih sedikit kode, alat yang canggih, dan kemampuan Kotlin yang intuitif. Compose dapat membantu Anda membuat UI dengan menentukan sekumpulan fungsi, yang disebut fungsi composable, yang mengambil data dan menjelaskan elemen UI.

Fungsi composable

Fungsi composable adalah elemen dasar penyusun UI di Compose. Fungsi composable:

- Menjelaskan beberapa bagian UI Anda.
- Tidak menampilkan apa pun.
- Mengambil beberapa input dan menghasilkan elemen yang ditampilkan di layar.

Anotasi

Anotasi adalah cara untuk melampirkan informasi tambahan ke kode. Informasi ini membantu alat seperti compiler Jetpack Compose, dan developer lain memahami kode aplikasi.

Anotasi diterapkan dengan menambahkan awalan nama (anotasi) dengan karakter `@` di awal deklarasi yang Anda anotasi. Berbagai elemen kode, seperti properti, fungsi, dan class, dapat dianotasi. Kemudian dalam kursus ini, Anda akan mempelajari class.

Nama fungsi composable

Fungsi compose yang tidak menampilkan apa pun dan menyertakan anotasi `@Composable` HARUS diberi nama menggunakan Pascal case. Pascal case

mengacu pada konvensi penamaan yang menggunakan huruf kapital untuk huruf pertama setiap kata dalam kata majemuk. Perbedaan antara Pascal case dan camel case adalah semua kata dalam Pascal case harus ditulis dalam huruf kapital. Dalam camel case, kata pertama dapat menggunakan huruf kapital atau tidak.

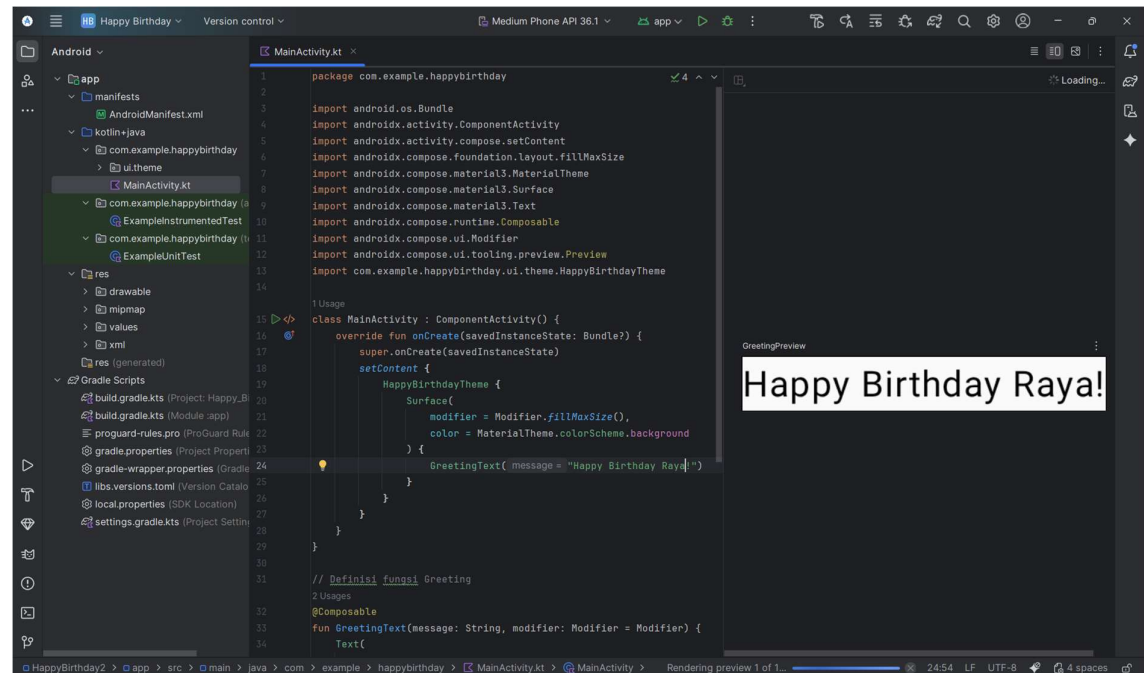
Fungsi Compose:

- *HARUS* kata benda: DoneButton()
- *BUKAN* kata kerja atau frasa kata kerja: DrawTextField()
- *BUKAN* preposisi yang dibuat menjadi kata benda: TextFieldWithLink()
- *BUKAN* kata sifat: Bright()
- *BUKAN* kata keterangan: Outside()
- Kata benda *DAPAT* diawali dengan kata sifat deskriptif: RoundIcon()

#### 4. Design pane in Android Studio

Fungsi composable harus menyediakan nilai default untuk setiap parameter agar dapat dipratinjau. Oleh karena itu, sebaiknya jangan lihat pratinjau fungsi Greeting() secara langsung. Sebagai gantinya, Anda perlu menambahkan fungsi lain, fungsi BirthdayCardPreview() dalam kasus ini, yang akan memanggil fungsi Greeting() dengan parameter yang sesuai.

- Emulator



#### 5. Add a new text element

- Menambahkan fungsi composable baru

```
package com.example.happybirthday

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.foundation.layout.fillMaxSize
```

```

import androidx.compose.material3.MaterialTheme
import androidx.compose.material3.Surface
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.ui.Modifier
import androidx.compose.ui.tooling.preview.Preview
import com.example.happybirthday.ui.theme.HappyBirthdayTheme

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            HappyBirthdayTheme {
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colorScheme.background
                ) {
                    GreetingText("Happy Birthday Raya!")
                }
            }
        }
    }
}

// Definisi fungsi Greeting
@Composable
fun GreetingText(message: String, modifier: Modifier = Modifier) {
    Text(
        text = message,
        modifier = modifier
    )
}

// Definisi fungsi Preview
@Preview(showBackground = true)
@Composable
fun GreetingPreview() {
    HappyBirthdayTheme {
        GreetingText("Happy Birthday Raya!")
    }
}

```

- Output

# Happy Birthday Raya!

6. Change font size
  - Piksel skalabel

Piksel skalabel(SP) adalah satuan ukuran untuk ukuran font. Elemen UI di aplikasi Android menggunakan dua unit pengukuran yang berbeda: Piksel Kepadatan Mandiri(DP) yang Anda gunakan nanti untuk tata letak, dan piksel skalabel (SP). Secara default, unit SP memiliki ukuran yang sama dengan unit DP, tetapi ukurannya dapat diubah berdasarkan ukuran teks pilihan pengguna di setelan ponsel.

```
package com.example.happybirthday

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.material3.MaterialTheme
import androidx.compose.material3.Surface
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.ui.Modifier
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.sp
import com.example.happybirthday.ui.theme.HappyBirthdayTheme

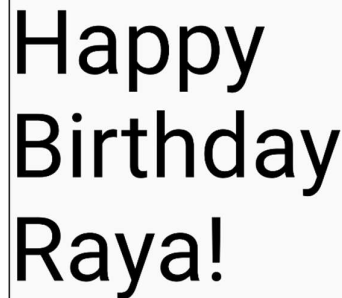
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            HappyBirthdayTheme {
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colorScheme.background
                ) {
                    GreetingText("Happy Birthday Raya!")
                }
            }
        }
    }
}

@Composable
fun GreetingText(message: String, modifier: Modifier = Modifier) {
    Text(
        text = message,
        fontSize = 100.sp,
        lineHeight = 116.sp,
        modifier = modifier
    )
}

@Preview(showBackground = true)
@Composable
fun GreetingPreview() {
```

```
HappyBirthdayTheme {
    GreetingText("Happy Birthday Raya!")
}
```

- Output



7. Add another text element

- Source Code

```
package com.example.happybirthday

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.material3.MaterialTheme
import androidx.compose.material3.Surface
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.ui.Modifier
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.sp
import com.example.happybirthday.ui.theme.HappyBirthdayTheme

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            HappyBirthdayTheme {
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colorScheme.background
                ) {
                    GreetingText(message = "Happy Birthday Raya!", from = "From
Emma")
                }
            }
        }
    }
}
```

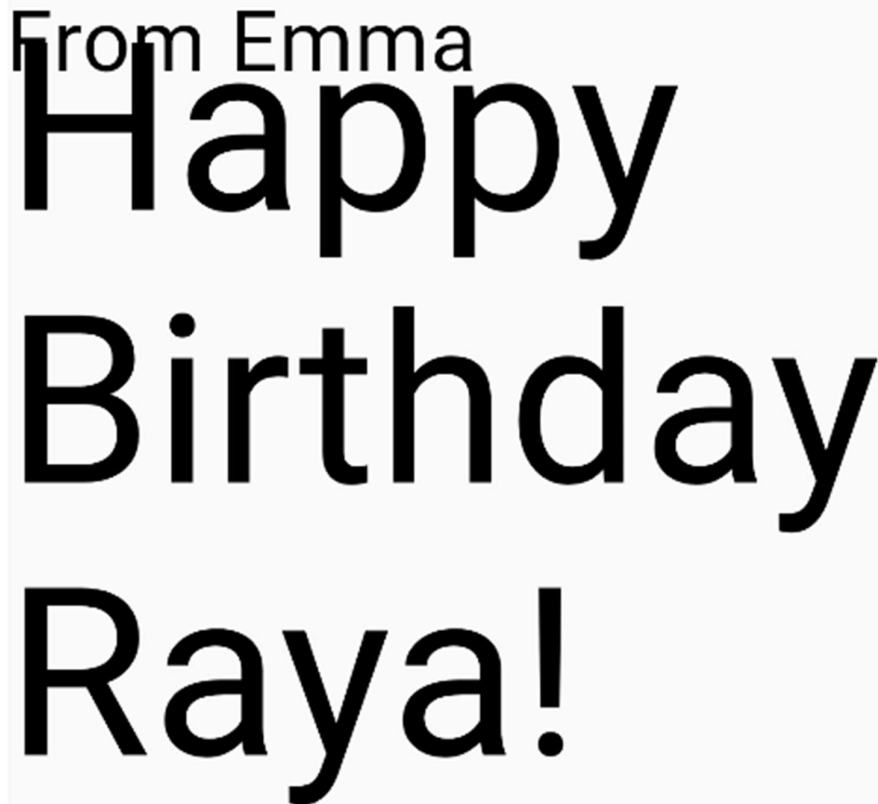
```

@Composable
fun GreetingText(message: String, from: String, modifier: Modifier = Modifier)
{
    Text(
        text = message,
        fontSize = 100.sp,
        lineHeight = 116.sp
    )
    Text(
        text = from,
        fontSize = 36.sp
    )
}

@Preview(showBackground = true)
@Composable
fun GreetingPreview() {
    HappyBirthdayTheme {
        GreetingText(message = "Happy Birthday Raya!", from = "From Emma")
    }
}

```

- Output



From Emma  
Happy  
Birthday  
Raya!

8. Arrange the text elements in a row and column



- Source Code

```
package com.example.happybirthday

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.material3.MaterialTheme
import androidx.compose.material3.Surface
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.ui.Modifier
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.sp
import com.example.happybirthday.ui.theme.HappyBirthdayTheme

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            HappyBirthdayTheme {
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colorScheme.background
                ) {
                    GreetingText(message = "Happy Birthday Raya!", from = "From
Emma")
                }
            }
        }
    }
}

@Composable
fun GreetingText(message: String, from: String, modifier: Modifier = Modifier)
{
    Column(
        modifier = modifier
    ) {
        Text(
            text = message,
            fontSize = 100.sp,
            lineHeight = 116.sp
        )
        Text(
            text = from,
            fontSize = 36.sp
        )
    }
}
```

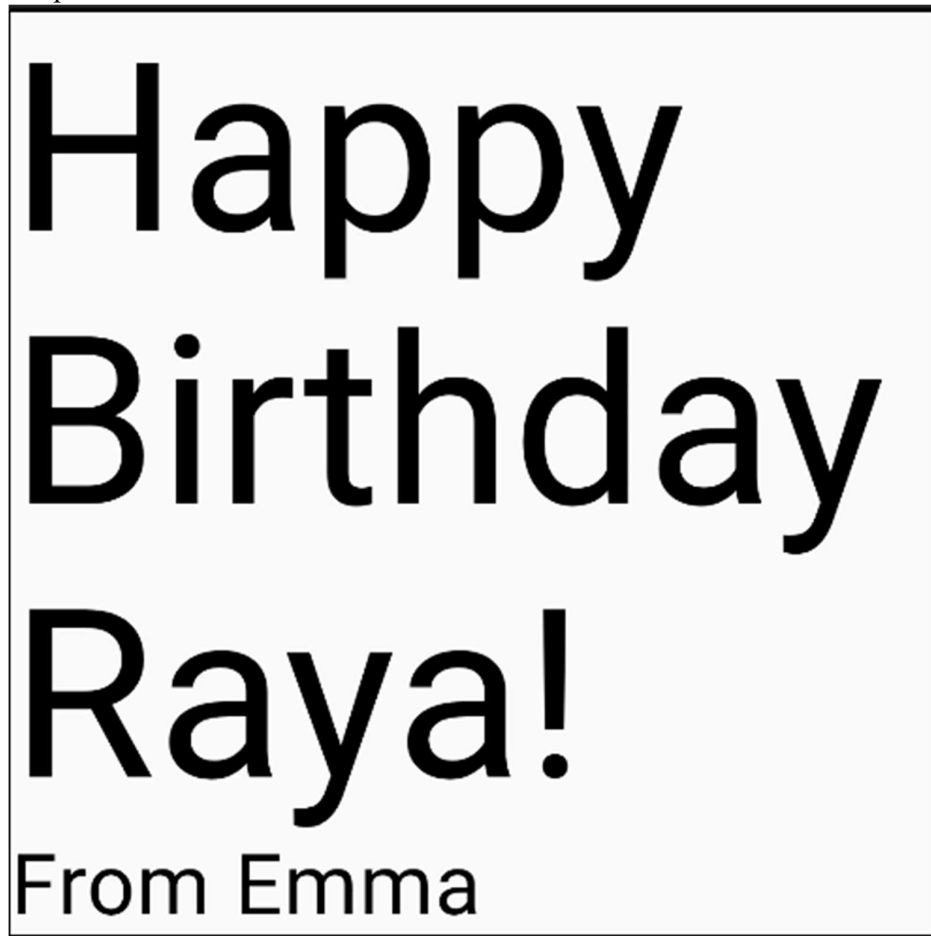
```

    }
}

@Preview(showBackground = true)
@Composable
fun GreetingPreview() {
    HappyBirthdayTheme {
        GreetingText(message = "Happy Birthday Raya!", from = "From Emma")
    }
}

```

- Output



9. Add greeting to the app

- Source Code

```

package com.example.happybirthday

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.foundation.layout.Arrangement
import androidx.compose.foundation.layout.Column

```

```

import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.padding
import androidx.compose.material3.MaterialTheme
import androidx.compose.material3.Surface
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import com.example.happybirthday.ui.theme.HappyBirthdayTheme

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            HappyBirthdayTheme {
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colorScheme.background
                ) {
                    GreetingText(
                        message = "Happy Birthday Raya!",
                        from = "From Emma",
                        modifier = Modifier.padding(8.dp)
                    )
                }
            }
        }
    }
}

@Composable
fun GreetingText(message: String, from: String, modifier: Modifier = Modifier)
{
    Column(
        verticalArrangement = Arrangement.Center,
        modifier = modifier
    ) {
        Text(
            text = message,
            fontSize = 100.sp,
            lineHeight = 116.sp,
            textAlign = TextAlign.Center
        )
        Text(
            text = from,
            fontSize = 36.sp,

```

```
        modifier = Modifier
            .padding(16.dp)
            .align(alignment = Alignment.End)
    )
}

@Preview(showBackground = true)
@Composable
fun GreetingPreview() {
    HappyBirthdayTheme {
        GreetingText(
            message = "Happy Birthday Raya!",
            from = "From Emma",
            modifier = Modifier.padding(8.dp)
        )
    }
}
```

- Output Emulator

9:53



Happy  
Birthday  
Raya!

From Emma

- Output Preview

# Happy Birthday Raya!

From Emma

10. Get the solution code

- [Source Code](#)

```
package com.example.happybirthday

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.foundation.layout.Arrangement
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.padding
import androidx.compose.material3.MaterialTheme
import androidx.compose.material3.Surface
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
```

```

import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import com.example.happybirthday.ui.theme.HappyBirthdayTheme

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            HappyBirthdayTheme {
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colorScheme.background
                ) {
                    GreetingText(
                        message = "Happy Birthday Raya!",
                        from = "From Emma",
                        modifier = Modifier.padding(8.dp)
                    )
                }
            }
        }
    }
}

@Composable
fun GreetingText(message: String, from: String, modifier: Modifier = Modifier)
{
    Column(
        verticalArrangement = Arrangement.Center,
        modifier = modifier
    ) {
        Text(
            text = message,
            fontSize = 100.sp,
            lineHeight = 116.sp,
            textAlign = TextAlign.Center
        )
        Text(
            text = from,
            fontSize = 36.sp,
            modifier = Modifier
                .padding(16.dp)
                .align(alignment = Alignment.End)
        )
    }
}

@Preview(showBackground = true)

```

```

@Composable
fun BirthdayCardPreview() {
    HappyBirthdayTheme {
        GreetingText(
            message = "Happy Birthday Raya!",
            from = "From Emma",
            modifier = Modifier.padding(8.dp)
        )
    }
}

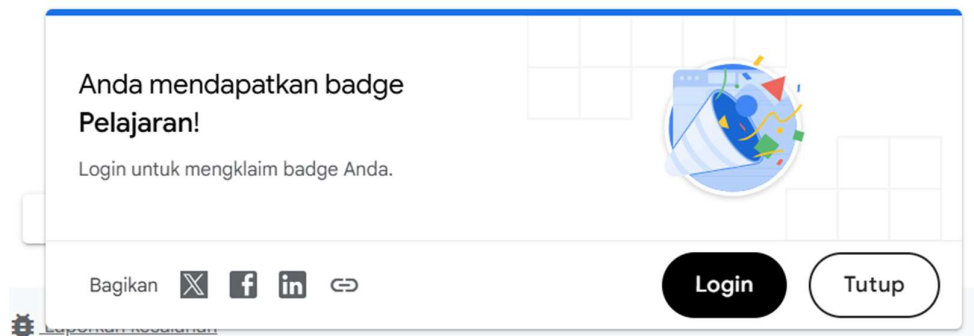
```

- Output masih sama dengan tahap 9

## 11. Conclusion

### Ringkasan

- Jetpack Compose adalah toolkit modern untuk membangun UI Android. Jetpack Compose menyederhanakan dan mempercepat pengembangan UI di Android dengan lebih sedikit kode, alat yang canggih, dan API Kotlin yang intuitif.
- Antarmuka pengguna (UI) aplikasi adalah tampilan visual yang Anda lihat di layar: teks, gambar, tombol, dan berbagai jenis elemen lainnya.
- Fungsi composable adalah elemen penyusun dasar Compose. Fungsi composable adalah fungsi yang mendeskripsikan beberapa bagian UI Anda.
- Fungsi composable dianotasi menggunakan anotasi `@Composable`; anotasi ini memberi tahu compiler Compose bahwa fungsi ini dimaksudkan untuk mengonversi data menjadi UI.
- Tiga elemen tata letak standar dan dasar di Compose adalah Column, Row,, dan Box. Tiga elemen tersebut adalah fungsi Composable yang menggunakan konten Composable sehingga Anda dapat menempatkan item di dalamnya. Misalnya, setiap turunan dalam Row akan ditempatkan berdampingan secara horizontal.



2. untuk memahami siklus aktivitas android pelajari link berikut <https://developer.android.com/guide/components/activities/activity-lifecycle?hl=id> dan jawab pertanyaan berikut ini:

pada saat kapan, kondisi bagaimana, dan user melakukan apa method-method berikut dieksekusi:

1. onCreate()

Jawab :

- Kapan Dieksekusi:



- Dipanggil saat Activity dibuat untuk pertama kalinya dalam memory
- Ini adalah method paling pertama yang dipanggil dalam lifecycle
- Hanya dipanggil sekali kecuali Activity di-destroy dan dibuat ulang
- Kondisi:
  - Activity belum ada di memory sama sekali (instance baru)
  - Sistem Android mengalokasikan memory untuk Activity
  - Activity dalam status "Created" tapi belum visible
- User Melakukan Apa:
  - Tap icon aplikasi di launcher → onCreate() dipanggil untuk Activity pertama
  - Klik button yang membuka Activity baru → onCreate() dipanggil untuk Activity baru tersebut
  - Rotasi layar (landscape ↔ portrait) → Activity dihancurkan dan onCreate() dipanggil lagi untuk membuat ulang
  - Membuka deep link atau notification → onCreate() dipanggil untuk Activity tujuan

## 2. onStart()

- Kapan Dieksekusi:
  - Dipanggil setelah onCreate() selesai
  - Atau dipanggil setelah onRestart() jika Activity kembali dari stopped state
  - Dipanggil setiap kali Activity akan menjadi visible
- Kondisi:
  - Activity mulai terlihat di layar (visible)
  - Tapi user belum bisa berinteraksi (not in foreground yet)
  - Activity dalam transisi menuju foreground
  - UI sudah mulai di-render
- User Melakukan Apa:
  - Pertama kali membuka aplikasi → setelah onCreate(), onStart() dipanggil
  - Kembali dari Home screen → onRestart() → onStart()
  - Menutup dialog fullscreen yang menutupi Activity → onStart()
  - Kembali dari Activity lain dengan tombol Back → onStart()

## 3. onResume()

- Kapan Dieksekusi:
  - Dipanggil setelah onStart() selesai
  - Atau saat Activity kembali ke foreground dari paused state
  - Method terakhir sebelum Activity fully interactive
- Kondisi:
  - Activity sepenuhnya visible dan di foreground
  - User bisa berinteraksi penuh (tap, scroll, type, dll)
  - Activity dalam status "Running" atau "Active"
  - Ini adalah kondisi normal saat user menggunakan aplikasi
- User Melakukan Apa:
  - Activity siap digunakan → user bisa tap, scroll, input text
  - Kembali dari Activity lain → onResume() dipanggil

- Menutup dialog → onResume() dipanggil
- Selesai menerima telepon → onResume() dipanggil
- Keluar dari multi-window mode → onResume() dipanggil

#### 4. onPause()

- Kapan Dieksekusi:
  - Saat Activity kehilangan focus tapi masih partially visible
  - Method pertama yang dipanggil saat user meninggalkan Activity
  - Dipanggil sangat cepat (harus selesai dalam waktu singkat)
- Kondisi:
  - Activity masih terlihat sebagian (partially visible)
  - Tapi user tidak bisa berinteraksi
  - Ada Activity/window lain yang overlap dan mendapat focus
  - Activity dalam transisi meninggalkan foreground
- User Melakukan Apa:
  - Membuka dialog → Activity utama masih terlihat tapi onPause() dipanggil
  - Menerima telepon → Activity pause, phone app dapat focus
  - Multi-window mode → Activity lain mendapat focus, Activity ini onPause()
  - Membuka Activity transparan → Activity di belakang masih terlihat tapi pause
  - Notification muncul (jika fullscreen) → onPause()
  - Share sheet muncul → Activity di belakang onPause()

#### 5. onStop()

- Kapan Dieksekusi:
  - Dipanggil setelah onPause() selesai
  - Saat Activity tidak lagi visible sama sekali
  - Bisa jadi Activity dihancurkan atau hanya di-background
- Kondisi:
  - Activity sepenuhnya tertutup (completely hidden)
  - Activity tidak terlihat sama sekali di layar
  - Activity dalam status "Stopped"
  - Memory masih ada, tapi bisa di-kill sistem jika memory rendah
- User Melakukan Apa:
  - Tekan tombol Home → App masuk background, onStop() dipanggil
  - Buka aplikasi lain → Activity sebelumnya onStop()
  - Tekan tombol Recent Apps → Current activity onStop()
  - Navigasi ke Activity lain (fullscreen) → Activity sebelumnya onStop()
  - Lock screen HP → Activity yang active onStop()

#### 6. onDestroy()

- Kapan Dieksekusi:
  - Saat Activity akan dihancurkan sepenuhnya dari memory
  - Method terakhir yang dipanggil dalam lifecycle
  - Setelah ini, Activity tidak ada lagi di memory
- Kondisi:

- Activity akan dimusnahkan (destroyed)
- Memory akan dibebaskan
- Activity akan hilang dari memory
- Bisa karena user keluar atau sistem butuh memory
- User Melakukan Apa:
  - Tekan tombol Back → Activity selesai, onDestroy() dipanggil
  - Call finish() programmatically → onDestroy() dipanggil
  - Rotasi layar → onDestroy() lalu onCreate() (recreate)
  - Force close app → onDestroy() dipanggil
  - Sistem kill process karena memory rendah → onDestroy() mungkin TIDAK dipanggil!
  - Change configuration (language, theme) → onDestroy() lalu onCreate()

## 7. onRestart()

- Kapan Dieksekusi:
  - Saat Activity yang sudah stopped akan dimulai lagi
  - Hanya dipanggil jika Activity sudah pernah dibuat sebelumnya
  - Tidak dipanggil saat pertama kali Activity dibuat
- Kondisi:
  - Activity dalam status "Stopped" (dari onStop())
  - Activity masih ada di memory
  - Activity akan visible Kembali
  - Activity tidak di-destroy, hanya stopped
- User Melakukan Apa:
  - Kembali dari Home screen → onRestart() → onStart() → onResume()
  - Kembali dari Recent Apps → onRestart() → onStart() → onResume()
  - Kembali dari Activity lain (dengan Back button) → onRestart() → onStart() → onResume()
  - Unlock screen (jika Activity masih di memory) → onRestart()