LAPORAN PENUGASAN UJIAN TENGAH SEMESTER

PEMROGRAMAN MOBILE 1

Dosen Pengampu: Nova Agustina, ST., M.Kom.

Disusun Oleh:

Nama : **Bayu Aji Prayoga**

NIM : **23552011194**

Kelas : **TIF RP-23 CNS A**

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS INDUSTRI KREATIF

UNIVERSITAS TEKNOLOGI BANDUNG

2025

Nama Matakuliah/SKS	: Mobile Programming 1 / 3 SKS
Dosen	: Nova Agustina, M.Kom.
Waktu/Sifat Ujian	: Tugas Besar
Kelas	: TIF RP 23 CNS A

Perhatikan setiap butir soal berikut dengan teliti. Pastikan Anda sudah memenuhi semua kriteria sebelum disubmit!

Essay

1. Apa fungsi findViewById?
2. Apa syarat pemanggilan method findViewById? Buat contohnya dan screenshot source code nya!
3. Error apa yang terjadi jika file kotlin salah menginisialisasi findViewById atau objek pada xml belum diinisialisasi?
4. Buat sebuah contoh program untuk menampilkan pesan error Resources.NotFoundException! Screenshot logcat-nya!
5. Kumpulkan dalam bentuk pdf di Elearning (Soal essay digabung dengan soal studi kasus cek point 7 Studi Kasus)

Studi Kasus

1. Buatlah sebuah program sederhana yang terdiri dari 4 Activity menggunakan Android Native (Java + XML) yang terdiri dari:

a SplashScreen Activity

b Login Activity

c Register Activity

d List Chating

2. Ketentuan: Silahkan membuat splashcreen dengan baik.
3. Pada Register Activity, minimal terdapat objek: TextView, EditText, Button, ImageView!
4. Tampilkan event Log, Toast dan Toast pada saat Button Register di klik.

5. Pada List Chating terdapat data yang ditampilkan dalam listview
6. Upload project di Github.
7. **Jelaskan fungsi setiap baris source code pada file kotlin dan submit dalam bentuk pdf pada Elearning**
8. **Link Github harus tercantum pada pdf (point 7)**

PLAGIAT/POINT 6 TIDAK TERPENUHI = NILAI STUDI KASUS 0

Kepala Departemen Teknik Informatika Yasti Aisyah Primianjani, S.Kom.		Dosen Koordinator Nova Agustina, M.Kom.	
Tanggal 26 / 04 / 2025	Tanda Tangan	Tanggal 26 / 04 / 2025	Tanda Tangan

A. Essay

1. **findViewById** adalah sebuah method di Android yang digunakan untuk mencari dan mendapatkan referensi ke sebuah View (komponen UI) yang ada di layout XML berdasarkan ID-nya. Dengan method ini, kita bisa menghubungkan komponen UI yang sudah didefinisikan di file XML ke dalam kode Kotlin atau Java agar bisa dimanipulasi secara programatik.
2. Syarat pemanggilan findViewById:
    a Layout XML yang berisi View dengan ID yang ingin dicari harus sudah di-inflate (biasanya dengan `setContentView()` di Activity atau `inflate()` di Fragment).
    b ID yang dicari harus benar-benar ada di layout yang sedang aktif.
    c Tipe data variabel yang menampung hasil findViewById harus sesuai dengan tipe View yang dicari (misal Button, TextView, dll).
    d Jika menggunakan Kotlin, biasanya perlu casting ke tipe View yang sesuai (walau dengan Kotlin Android Extensions atau View Binding, ini bisa lebih mudah).

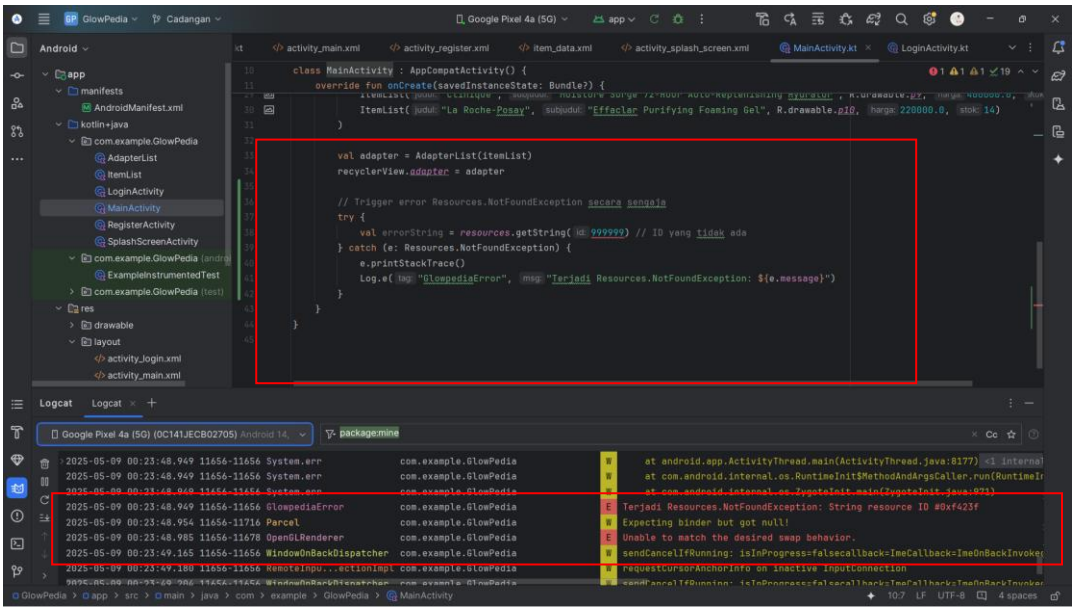
    Contohnya:

```kotlin
package com.example.myapp

import android.os.Bundle
import android.widget.TextView
import androidx.appcompat.app.AppCompatActivity

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        val myTextView = findViewById<TextView>(R.id.myTextView)
        myTextView.text = "Halo Dunia!"
    }
}
```

3. Jika Kamu memanggil findViewById dengan ID yang tidak ada di layout yang sedang aktif, maka method ini akan mengembalikan null. Jika Anda mencoba menggunakan objek tersebut tanpa pengecekan null, maka akan terjadi:
    a **NullPointerException** saat kamu mencoba mengakses objek tersebut
    b Contoh error yang sering muncul:

```
java.lang.NullPointerException: Attempt to invoke virtual method 'void android.widget.Button.setOnClickListener
```

4. Buat sebuah contoh program untuk menampilkan pesan error Resources.NotFoundException! Screenshot logcat-nya!



5. Kumpulkan dalam bentuk pdf di Elearning (Soal essay digabung dengan soal studi kasus cek point 7 Studi Kasus)

   PDF di push ke github: https://github.com/BayuAjiPrayoga/UTS-PM-1-Bayu-Aji-Prayoga-23552011194

B. Studi Kasus

AdapterList.kt	package com.example.GlowPedia
	// Mendefinisikan package tempat kelas ini berada, sebagai namespace agar tidak bentrok dengan kelas lain.
	
	import android.view.LayoutInflater
	import android.view.View
	import android.view.ViewGroup
	import android.widget.ImageView
	import android.widget.TextView
	import androidx.recyclerview.widget.RecyclerView
	// Import library dan kelas yang dibutuhkan untuk membuat adapter RecyclerView dan mengelola tampilan item.
	
	class AdapterList(private val itemList: List<ItemList>) : RecyclerView.Adapter<AdapterList.ViewHolder>() {

	// Mendefinisikan kelas AdapterList yang merupakan adapter untuk RecyclerView.
	// Adapter ini menerima list data bertipe ItemList sebagai sumber data yang akan ditampilkan.
	
	class ViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {
	// Kelas ViewHolder bertugas menyimpan referensi ke view-item yang akan digunakan untuk menampilkan data.
	// ViewHolder ini mengoptimalkan performa RecyclerView dengan menghindari pencarian view berulang.
	
	val imageView: ImageView = itemView.findViewById(R.id.item_image)
	// Mendapatkan referensi ImageView dari layout item dengan id item_image, untuk menampilkan gambar.
	
	val judul: TextView = itemView.findViewById(R.id.title)
	// Mendapatkan referensi TextView dengan id title, untuk menampilkan judul item.
	
	val subJudul: TextView = itemView.findViewById(R.id.sub_title)
	// Mendapatkan referensi TextView dengan id sub_title, untuk menampilkan subjudul item.
	
	val harga: TextView = itemView.findViewById(R.id.harga) // TextView untuk harga
	// Mendapatkan referensi TextView dengan id harga, untuk menampilkan harga item.
	
	val stok: TextView = itemView.findViewById(R.id.stok) // TextView untuk stok
	// Mendapatkan referensi TextView dengan id stok, untuk menampilkan jumlah stok item.
	}
	
	override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {
	// Fungsi ini dipanggil saat RecyclerView membutuhkan ViewHolder baru.

```
    // Bertugas membuat dan meng-inflate layout item dari XML menjadi objek View.

        val view = LayoutInflater.from(parent.context).inflate(R.layout.item_data, parent, false)
        // Meng-inflate layout item_data.xml menjadi View yang akan digunakan untuk satu item di RecyclerView.

        return ViewHolder(view)
        // Mengembalikan ViewHolder baru yang berisi view hasil inflate tadi.
    }

    override fun onBindViewHolder(holder: ViewHolder, position: Int) {
    // Fungsi ini dipanggil untuk mengikat data ke ViewHolder pada posisi tertentu.
    // Di sini data dari itemList diambil dan ditampilkan ke dalam view yang sesuai.

        val item = itemList[position]
        // Mengambil data item pada posisi tertentu dari list.

        holder.judul.text = item.judul
        // Mengatur teks judul pada TextView judul di ViewHolder.

        holder.subJudul.text = item.subjudul
        // Mengatur teks subjudul pada TextView subJudul di ViewHolder.


        holder.imageView.setImageResource(item.imageResId)
        // Mengatur gambar pada ImageView menggunakan resource ID yang ada di item.

        // Menampilkan harga dan stok
        holder.harga.text = "Rp${item.harga.toInt()}"
        // Mengatur teks harga dengan format "Rp" diikuti nilai harga yang dikonversi ke integer.

        holder.stok.text = "Stok: ${item.stok}"
        // Mengatur teks stok dengan format "Stok:"
```

	diikuti jumlah stok item.
	}
	
	override fun getItemCount(): Int = itemList.size
	// Mengembalikan jumlah total item yang ada di list, digunakan RecyclerView untuk menentukan berapa banyak item yang akan ditampilkan.
	}

ItemList.kt	//ItemList adalah model data yang menyimpan semua informasi penting tentang satu item.
	//Data class memudahkan pengelolaan data dan integrasi dengan UI.
	//Adapter menggunakan objek ItemList untuk menampilkan data secara dinamis di RecyclerView.
	
	package com.example.GlowPedia
	
	data class ItemList(
	val judul: String,
	val subjudul: String,
	val imageResId: Int,
	val harga: Double,
	val stok: Int
	)

LoginActivity.kt	package com.example.GlowPedia
	// Mendefinisikan package aplikasi, sebagai namespace agar kelas ini tidak bentrok dengan kelas lain.
	
	import android.content.Context
	import android.content.Intent
	import android.os.Bundle
	import android.util.Patterns
	import android.widget.Toast
	import androidx.appcompat.app.AppCompatActivity
	import com.google.android.material.button.MaterialButton
	import com.google.android.material.textfield.TextInputEditText
	import android.widget.TextView
	// Import berbagai kelas dan komponen yang dibutuhkan:
	// - Context dan Intent untuk navigasi antar activity

```
dan akses konteks aplikasi.
// - Bundle untuk menyimpan state activity.
// - Patterns untuk validasi pola, khususnya email.
// - Toast untuk menampilkan pesan singkat ke
pengguna.
// - Komponen UI dari Material Design seperti
MaterialButton dan TextInputEditText.
// - TextView untuk elemen teks yang bisa diklik
(misal link daftar).

class LoginActivity : AppCompatActivity() {
  // Kelas LoginActivity yang merupakan activity
untuk halaman login pengguna.
  // Turunan dari AppCompatActivity agar
kompatibel dengan berbagai versi Android dan fitur
modern.

  // Deklarasi variabel UI yang akan dihubungkan
dengan elemen layout
  private lateinit var etEmail: TextInputEditText    //
Input field untuk email
  private lateinit var etPassword: TextInputEditText
// Input field untuk password
  private lateinit var btnLogin: MaterialButton      //
Tombol login
  private lateinit var tvRegister: TextView          //
TextView yang berfungsi sebagai link ke halaman
registrasi

  override fun onCreate(savedInstanceState:
Bundle?) {
    super.onCreate(savedInstanceState)
    // Memanggil superclass dan menginisialisasi
activity

    setContentView(R.layout.activity_login)
    // Mengatur layout activity dengan file XML
activity_login.xml

    // Menghubungkan variabel dengan elemen UI
berdasarkan ID yang ada di layout
    etEmail = findViewById(R.id.etEmail)
    etPassword = findViewById(R.id.etPassword)
    btnLogin = findViewById(R.id.btnLogin)
    tvRegister = findViewById(R.id.tvRegister)

    // Menetapkan aksi ketika tombol login ditekan
```

```
        btnLogin.setOnClickListener {
            // Mengambil input dari user dan
menghilangkan spasi di awal/akhir
            val emailInput = etEmail.text.toString().trim()
            val passwordInput =
etPassword.text.toString().trim()

            // Validasi input kosong
            if (emailInput.isEmpty() ||
passwordInput.isEmpty()) {
                Toast.makeText(this, "Email dan password
harus diisi", Toast.LENGTH_SHORT).show()
                // Menampilkan pesan singkat bahwa email
dan password wajib diisi
                return@setOnClickListener
                // Menghentikan eksekusi lebih lanjut jika
input kosong
            }

            // Validasi format email menggunakan pola
regex bawaan Android
            if
(!Patterns.EMAIL_ADDRESS.matcher(emailInput).m
atches()) {
                Toast.makeText(this, "Format email tidak
valid", Toast.LENGTH_SHORT).show()
                // Menampilkan pesan jika format email tidak
sesuai standar
                return@setOnClickListener
            }

            // Mengakses SharedPreferences untuk
mengambil data user yang sudah tersimpan
            val sharedPref =
getSharedPreferences("UserPrefs",
Context.MODE_PRIVATE)
            val savedEmail =
sharedPref.getString("username", null)
            val savedPassword =
sharedPref.getString("password", null)

            // Cek apakah data user sudah pernah
disimpan (apakah sudah registrasi)
            if (savedEmail == null || savedPassword ==
null) {
                Toast.makeText(this, "Belum ada akun
terdaftar, silakan daftar dulu",
```

	Toast.LENGTH_SHORT).show()
	// Jika belum ada data, beri tahu user untuk melakukan registrasi terlebih dahulu
	}
	// Jika data ada, cek apakah input user cocok dengan data yang tersimpan
	else if (emailInput == savedEmail && passwordInput == savedPassword) {
	Toast.makeText(this, "Login berhasil!", Toast.LENGTH_SHORT).show()
	// Jika cocok, tampilkan pesan sukses login
	
	// Pindah ke MainActivity setelah login berhasil
	startActivity(Intent(this, MainActivity::class.java))
	finish()
	// Mengakhiri LoginActivity agar tidak bisa kembali ke halaman login dengan tombol back
	} else {
	Toast.makeText(this, "Email atau password salah", Toast.LENGTH_SHORT).show()
	// Jika email atau password tidak cocok, tampilkan pesan error
	}
	}
	
	// Menetapkan aksi ketika teks "Daftar" diklik
	tvRegister.setOnClickListener {
	// Membuka RegisterActivity untuk pendaftaran akun baru
	startActivity(Intent(this, RegisterActivity::class.java))
	}
	}
	}

MainActivity.kt	package com.example.GlowPedia // Mendefinisikan package aplikasi sebagai namespace agar kelas ini tidak bentrok dengan kelas lain. import android.content.res.Resources import android.os.Bundle import android.util.Log import androidx.appcompat.app.AppCompatActivity import

	androidx.recyclerview.widget.LinearLayoutManager
	import androidx.recyclerview.widget.RecyclerView
	// Import berbagai kelas yang dibutuhkan:
	// - AppCompatActivity sebagai base class untuk activity dengan fitur kompatibilitas.
	// - RecyclerView dan LinearLayoutManager untuk menampilkan daftar item secara efisien.
	// - Resources dan Log (meskipun di kode ini Resources dan Log tidak digunakan, bisa dihapus untuk bersih).
	
	class MainActivity : AppCompatActivity() {
	// Kelas MainActivity yang merupakan entry point utama aplikasi setelah login.
	// Turunan dari AppCompatActivity agar kompatibel dengan berbagai versi Android dan fitur modern.
	
	override fun onCreate(savedInstanceState: Bundle?) {
	super.onCreate(savedInstanceState)
	// Memanggil superclass dan menginisialisasi activity
	
	setContentView(R.layout.activity_main)
	// Mengatur layout activity dengan file XML activity_main.xml
	// Layout ini harus berisi RecyclerView dengan id "recycler_view"
	
	val recyclerView = findViewById<RecyclerView>(R.id.recycler_view)
	// Menghubungkan variabel recyclerView dengan elemen RecyclerView di layout berdasarkan ID
	
	recyclerView.layoutManager = LinearLayoutManager(this)
	// Mengatur layout manager RecyclerView menjadi LinearLayoutManager
	// LinearLayoutManager menampilkan item secara vertikal dalam bentuk list
	// Ini penting agar RecyclerView tahu bagaimana menata itemnya
	
	recyclerView.setHasFixedSize(true)
	// Memberi tahu RecyclerView bahwa ukuran layout tidak akan berubah berdasarkan isi adapter

	// Ini meningkatkan performa karena RecyclerView tidak perlu menghitung ulang ukuran saat data berubah
	
	// Membuat daftar item yang akan ditampilkan di RecyclerView
	// Setiap item adalah objek ItemList yang berisi judul, subjudul, gambar, harga, dan stok
	val itemList = listOf(
	ItemList("Cetaphil", "Gentle Skin Cleanser", R.drawable.p1, 120000.0, 15),
	ItemList("Neutrogena", "Hydro Boost Water Gel", R.drawable.p2, 250000.0, 20),
	ItemList("The Body Shop", "Tea Tree Oil", R.drawable.p3, 150000.0, 10),
	ItemList("L'Oréal Paris", "Revitalift Bright Reveal Brightening Peel Pads", R.drawable.p4, 200000.0, 8),
	ItemList("Nivea", "MicellAIR Skin Breathe", R.drawable.p5, 100000.0, 25),
	ItemList("Kiehl's", "Calendula Deep Cleansing Foaming Face Wash", R.drawable.p6, 350000.0, 5),
	ItemList("Avene", "Cicalfate Restorative Skin Cream", R.drawable.p7, 280000.0, 12),
	ItemList("Olay", "Regenerist Micro-Sculpting Cream", R.drawable.p8, 300000.0, 18),
	ItemList("Clinique", "Moisture Surge 72-Hour Auto-Replenishing Hydrator", R.drawable.p9, 400000.0, 6),
	ItemList("La Roche-Posay", "Effaclar Purifying Foaming Gel", R.drawable.p10, 220000.0, 14)
	)
	// Data ini bersifat statis dan hardcoded di sini, biasanya bisa diambil dari database atau API
	
	val adapter = AdapterList(itemList)
	// Membuat instance AdapterList dengan data itemList
	// Adapter ini bertugas menghubungkan data dengan tampilan item di RecyclerView
	
	recyclerView.adapter = adapter
	// Mengatur adapter RecyclerView agar menampilkan data yang sudah disiapkan
	// Setelah ini, RecyclerView akan memanggil adapter untuk membuat dan mengikat view item sesuai data

	} }

RegisterActivity.kt	package com.example.GlowPedia
	// Mendefinisikan package aplikasi sebagai namespace agar kelas ini tidak bentrok dengan kelas lain.
	
	import android.content.res.Resources
	import android.os.Bundle
	import android.util.Log
	import androidx.appcompat.app.AppCompatActivity
	import androidx.recyclerview.widget.LinearLayoutManager
	import androidx.recyclerview.widget.RecyclerView
	// Import berbagai kelas yang dibutuhkan:
	// - AppCompatActivity sebagai base class untuk activity dengan fitur kompatibilitas.
	// - RecyclerView dan LinearLayoutManager untuk menampilkan daftar item secara efisien.
	// - Resources dan Log (meskipun di kode ini Resources dan Log tidak digunakan, bisa dihapus untuk bersih).
	
	class MainActivity : AppCompatActivity() {
	// Kelas MainActivity yang merupakan entry point utama aplikasi setelah login.
	// Turunan dari AppCompatActivity agar kompatibel dengan berbagai versi Android dan fitur modern.
	
	override fun onCreate(savedInstanceState: Bundle?) {
	super.onCreate(savedInstanceState)
	// Memanggil superclass dan menginisialisasi activity
	
	setContentView(R.layout.activity_main)
	// Mengatur layout activity dengan file XML activity_main.xml
	// Layout ini harus berisi RecyclerView dengan id "recycler_view"
	
	val recyclerView = findViewById<RecyclerView>(R.id.recycler_view)
	// Menghubungkan variabel recyclerView dengan elemen RecyclerView di layout berdasarkan ID

```
        recyclerView.layoutManager =
LinearLayoutManager(this)
        // Mengatur layout manager RecyclerView
menjadi LinearLayoutManager
        // LinearLayoutManager menampilkan item
secara vertikal dalam bentuk list
        // Ini penting agar RecyclerView tahu bagaimana
menata itemnya

        recyclerView.setHasFixedSize(true)
        // Memberi tahu RecyclerView bahwa ukuran
layout tidak akan berubah berdasarkan isi adapter
        // Ini meningkatkan performa karena
RecyclerView tidak perlu menghitung ulang ukuran
saat data berubah

        // Membuat daftar item yang akan ditampilkan di
RecyclerView
        // Setiap item adalah objek ItemList yang berisi
judul, subjudul, gambar, harga, dan stok
        val itemList = listOf(
            ItemList("Cetaphil", "Gentle Skin Cleanser",
R.drawable.p1, 120000.0, 15),
            ItemList("Neutrogena", "Hydro Boost Water
Gel", R.drawable.p2, 250000.0, 20),
            ItemList("The Body Shop", "Tea Tree Oil",
R.drawable.p3, 150000.0, 10),
            ItemList("L'Oréal Paris", "Revitalift Bright
Reveal Brightening Peel Pads", R.drawable.p4,
200000.0, 8),
            ItemList("Nivea", "MicellAIR Skin Breathe",
R.drawable.p5, 100000.0, 25),
            ItemList("Kiehl's", "Calendula Deep Cleansing
Foaming Face Wash", R.drawable.p6, 350000.0, 5),
            ItemList("Avene", "Cicalfate Restorative Skin
Cream", R.drawable.p7, 280000.0, 12),
            ItemList("Olay", "Regenerist Micro-Sculpting
Cream", R.drawable.p8, 300000.0, 18),
            ItemList("Clinique", "Moisture Surge 72-Hour
Auto-Replenishing Hydrator", R.drawable.p9,
400000.0, 6),
            ItemList("La Roche-Posay", "Effaclar Purifying
Foaming Gel", R.drawable.p10, 220000.0, 14)
        )
        // Data ini bersifat statis dan hardcoded di sini,
biasanya bisa diambil dari database atau API
```

	val adapter = AdapterList(itemList)
	// Membuat instance AdapterList dengan data itemList
	// Adapter ini bertugas menghubungkan data dengan tampilan item di RecyclerView
	
	recyclerView.adapter = adapter
	// Mengatur adapter RecyclerView agar menampilkan data yang sudah disiapkan
	// Setelah ini, RecyclerView akan memanggil adapter untuk membuat dan mengikat view item sesuai data
	}
	}

SplashScreenActivity.kt	package com.example.GlowPedia
	// Mendefinisikan package aplikasi sebagai namespace agar kelas ini tidak bentrok dengan kelas lain.
	
	import android.content.Intent
	import android.os.Bundle
	import android.os.Handler
	import android.os.Looper
	import androidx.appcompat.app.AppCompatActivity
	// Import kelas-kelas penting:
	// - Intent untuk berpindah antar activity
	// - Bundle untuk menyimpan state activity
	// - Handler dan Looper untuk menjalankan kode dengan delay di thread UI
	// - AppCompatActivity sebagai superclass activity modern
	
	class SplashScreenActivity : AppCompatActivity() {
	// Kelas SplashScreenActivity yang menampilkan layar pembuka (splash screen)
	// Turunan dari AppCompatActivity agar kompatibel dengan berbagai versi Android dan fitur modern.
	
	private val splashTimeOut: Long = 2500 // Durasi splash screen dalam milidetik (2.5 detik)
	
	override fun onCreate(savedInstanceState: Bundle?) {
	super.onCreate(savedInstanceState)

	// Memanggil superclass dan menginisialisasi activity
	
	setContentView(R.layout.activity_splash_screen)
	// Mengatur layout activity dengan file XML activity_splash_screen.xml
	// Layout ini biasanya berisi logo atau animasi pembuka aplikasi
	
	// Menggunakan Handler dengan Looper utama (UI thread) untuk menunda eksekusi kode
	Handler(Looper.getMainLooper()).postDelayed({
	// Kode di dalam blok ini akan dijalankan setelah delay splashTimeOut
	
	startActivity(Intent(this, LoginActivity::class.java))
	// Memulai LoginActivity setelah splash screen selesai
	
	finish()
	// Mengakhiri SplashScreenActivity agar user tidak bisa kembali ke splash screen dengan tombol back
	}, splashTimeOut)
	}
	}

Link Github:	https://github.com/BayuAjiPrayoga/UTS-PM-1-Bayu-Aji-Prayoga-23552011194
Link Demo:	https://drive.google.com/file/d/1HQLb640wnfTbfWQwbo6X7PYrEQFROJzT/