LAPORAN PRAKTIKUM

UJIAN TENGAH SEMESTER

Disusun untuk memenuhi Ujian Tengah Semester matakuliah Pemograman Mobile 1

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

Disusun Oleh:

Syifa Aulia Fitri – 23552011013

TIF RP 223 CID-A

PROGRAM STUDI TEKNIK INFORMATIKA

UNIVERSITAS TEKNOLOGI BANDUNG

2025

Essay

1. Apa fungsi findViewById?

➔ Fungsi findViewById dalam pengembangan aplikasi Android (terutama dengan Java atau Kotlin) digunakan untuk menghubungkan variabel objek di dalam kode program dengan elemen atau *view* yang telah didefinisikan dalam layout XML. Sederhananya, ketika mendesain antarmuka pengguna (UI) menggunakan file XML (misalnya activity_main.xml), setiap elemen visual seperti TextView, Button, EditText, dan lainnya memiliki ID unik yang ditetapkan. findViewById memungkinkan untuk menemukan *view* tersebut berdasarkan ID-nya dari dalam kode Kotlin atau Java, sehingga dapat memanipulasi atau berinteraksi dengannya (misalnya, mengubah teks pada TextView, mengatur aksi klik pada Button, atau mengambil input dari EditText).

2. Apa syarat pemanggilan method findViewById? Buat contohnya dan screenshot source code nya!

   Syarat utama untuk memanggil *method* `findViewById` adalah:

➔ Layout XML harus sudah di-inflate atau diatur sebagai tampilan aktif**.** Ini biasanya terjadi di dalam *method* `onCreate()` pada sebuah `Activity` atau di dalam *method* `onCreateView()` pada sebuah `Fragment`. Proses *inflate* ini akan membaca struktur XML dan membuat objek-objek *view* yang sesuai di memori.

➔ ID dari *view* yang ingin diakses harus sudah didefinisikan dengan benar di dalam file layout XML menggunakan atribut `android:id="@+id/nama_id"`.

```
<TextView
    android:id="@+id/teksSapaan"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Halo Dunia!"
    android:textSize="20sp" />
```

```
class ContohActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_contoh) // Meng-inflate layout XML

        val textViewSapaan: TextView = findViewById(R.id.teksSapaan) // Memar
        textViewSapaan.text = "Selamat Pagi!" // Memanipulasi TextView
    }
```

Pada contoh di atas:

- `setContentView(R.layout.activity_contoh)` memastikan bahwa layout `activity_contoh.xml` telah di-*inflate* dan menjadi tampilan aktif.
- `findViewById(R.id.teksSapaan)` mencari *view* dengan ID `teksSapaan` di dalam layout yang aktif dan mengembalikan referensi ke objek `TextView`.

- Hasil dari `findViewById` kemudian di-*cast* (secara implisit di Kotlin) ke tipe `TextView` dan disimpan dalam variabel `textViewSapaan`.

3. Error apa yang terjadi jika file kotlin salah menginisialisasi findViewById atau objek pada xml belum diinisialisasi?

   Jika salah menginisialisasi findViewById (misalnya, menggunakan ID yang tidak ada di layout) atau mencoba mengakses objek *view* sebelum layout di-*inflate*, maka akan mendapatkan error :

   `kotlin.UninitializedPropertyAccessException`

   atau

   `java.lang.NullPointerException`, saat runtime.

   ➔ `kotlin.UninitializedPropertyAccessException`: Ini lebih sering terjadi jika mendeklarasikan properti *view* di kelas Kotlin tanpa langsung menginisialisasinya dan mencoba menggunakannya sebelum memanggil findViewById di dalam onCreate() atau onCreateView(). Kotlin menekankan keamanan *null*, dan jika properti yang tidak diinisialisasi diakses, error ini akan dilempar.

   ➔ `java.lang.NullPointerException`: Ini bisa terjadi jika findViewById mengembalikan null (karena ID tidak ditemukan dalam layout yang di-*inflate*) dan mencoba mengakses properti atau memanggil *method* pada variabel yang bernilai null tersebut.

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

```
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_main)

        val error = resources.getString( id: 66)
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) {
            val systemBars = insets.getInsets(WindowInsetsCompat.Type.system
            v.setPadding(systemBars.left, systemBars.top, systemBars.right,
            insets
        }
    }
}
```

Error:

```
ba/com.example.coba.MainActivity}: android.content.res.Resources$NotFoundException: String resource

m.java:101)
```

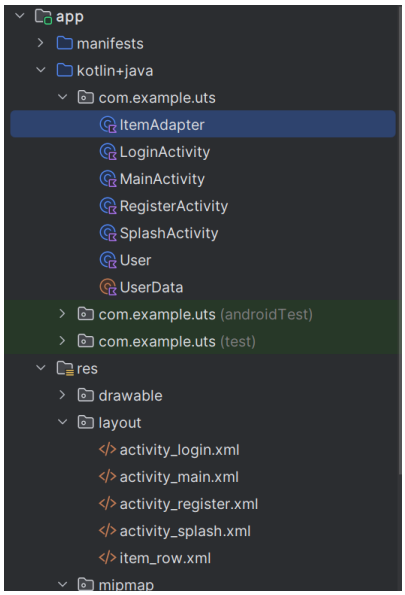
5. Kumpulkan dalam bentuk pdf di Elearning (Soal essay digabung dengan soal studi kasus cek point 7 Studi Kasus)

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 News Portal Dashboard
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 News Portal Dashboard 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

Penyelesaian Studi Kasus…

Struktur Proyek



Link Github: https://github.com/Syifaliaa15/UTS_Pemograman_Mobile1

Kode Kotlin (ItemAdapter.kt)

Kode Kotlin ini mengimplementasikan RecyclerView.Adapter yang digunakan untuk mengatur tampilan daftar item dalam aplikasi. Penjelasan setiap bagiannya adalah sebagai berikut:

a. Imports dan Deklarasi Kelas:

```
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
```

Pada bagian ini, kode mengimpor beberapa kelas dari Android SDK yang dibutuhkan untuk membuat tampilan dan interaksi dalam RecyclerView, seperti LayoutInflater untuk mengubah layout XML menjadi tampilan, dan RecyclerView.Adapter yang digunakan untuk mengelola data dalam RecyclerView.

b. Deklarasi Kelas ItemAdapter:

```
class ItemAdapter(private val items: List<String>) :
RecyclerView.Adapter<ItemAdapter.ItemViewHolder>() {
```

ItemAdapter adalah kelas yang mengatur bagaimana data (List<String> items) ditampilkan dalam RecyclerView. Kelas ini meng-extend RecyclerView.Adapter dan mengharuskan kita untuk mengimplementasikan tiga metode utama: onCreateViewHolder(), onBindViewHolder(), dan getItemCount().

c. ViewHolder (ItemViewHolder):

```
class ItemViewHolder(view: View) : RecyclerView.ViewHolder(view) {
    val textView = view.findViewById<TextView>(R.id.textViewItem)
    val imageView = view.findViewById<ImageView>(R.id.imageViewItem)
}
```

ItemViewHolder adalah kelas yang digunakan untuk menampung referensi ke tampilan (view) dari setiap item di dalam RecyclerView. Di sini, kita mengikatkan TextView dan ImageView yang ada di item_row.xml dengan findViewById().

d. Metode onCreateViewHolder():

```
override fun onCreateViewHolder(parent: ViewGroup, viewType: Int):
ItemViewHolder {
    val view = LayoutInflater.from(parent.context)
        .inflate(R.layout.item_row, parent, false)
    return ItemViewHolder(view)
}
```

Metode ini dipanggil ketika RecyclerView membutuhkan tampilan baru. Di sini, layout XML item_row.xml di-inflate menjadi view. ItemViewHolder kemudian dibuat dengan view tersebut.

e. Metode onBindViewHolder():

```
override fun onBindViewHolder(holder: ItemViewHolder, position: Int) {
    holder.textView.text = items[position]
    holder.imageView.setImageResource(R.drawable.itemspic)
}
```

Metode ini mengikat data ke tampilan yang ada di setiap item RecyclerView. Di sini, teks dari TextView diisi dengan nilai yang ada pada items[position] dan gambar di ImageView diatur dengan gambar default itemspic.

f. Metode getItemCount():

```
override fun getItemCount() = items.size
```

Metode ini mengembalikan jumlah item dalam list items. Ini penting untuk memberi tahu RecyclerView berapa banyak item yang harus ditampilkan.

Kode Kotlin - LoginActivity.kt

Pada kode Kotlin ini, LoginActivity mengatur interaksi dengan elemen-elemen UI (seperti EditText dan Button) yang ada di layout XML, serta mengimplementasikan logika untuk validasi login.

```
package com.example.uts

import android.content.Intent
import android.os.Bundle
import android.widget.*
import androidx.appcompat.app.AppCompatActivity

class LoginActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_login)

        val emailInput = findViewById<EditText>(R.id.editTextEmail)
        val passwordInput = findViewById<EditText>(R.id.editTextPassword)
        val loginButton = findViewById<Button>(R.id.buttonLogin)
        val registerButton = findViewById<Button>(R.id.buttonRegister)

        // Pindah ke RegisterActivity
        registerButton.setOnClickListener {
            startActivity(Intent(this, RegisterActivity::class.java))
        }

        // Validasi login
        loginButton.setOnClickListener {
            val email = emailInput.text.toString()
            val password = passwordInput.text.toString()

            // Mencocokkan data email dan password dengan data user yang ada
            val user = UserData.userList.find { it.email == email &&
it.password == password }

            if (user != null) {
                startActivity(Intent(this, MainActivity::class.java)) //
Berpindah ke MainActivity
                finish() // Menutup LoginActivity agar tidak bisa kembali
lagi ke halaman login
            } else {
                // Menampilkan pesan kesalahan jika login gagal
                Toast.makeText(this, "Email atau password salah",
Toast.LENGTH_SHORT).show()
            }
        }
```

```
    }
}
```

Penjelasan:

- Di dalam onCreate(), kita menginisialisasi elemen UI seperti EditText untuk email dan password, serta tombol login dan register.
- registerButton akan mengarahkan pengguna ke halaman pendaftaran (RegisterActivity) ketika diklik.
- loginButton memvalidasi email dan password yang dimasukkan dengan mencocokkannya dengan data pengguna yang ada dalam UserData.userList. Jika data valid, aplikasi berpindah ke MainActivity, dan jika tidak, pesan kesalahan akan muncul menggunakan Toast.

Kode Kotlin - RegisterActivity.kt

RegisterActivity adalah aktivitas yang menangani proses pendaftaran pengguna baru. Aktivitas ini memungkinkan pengguna untuk memasukkan email dan password untuk membuat akun baru. Setelah pendaftaran berhasil, data pengguna disimpan dalam UserData.userList dan aplikasi mengarahkan pengguna kembali ke halaman login.

```
class RegisterActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_register)

        // Mendapatkan referensi ke elemen-elemen UI
        val emailInput = findViewById<EditText>(R.id.editTextEmail)
        val passwordInput = findViewById<EditText>(R.id.editTextPassword)
        val                          registerButton                          =
findViewById<Button>(R.id.buttonRegisterAccount)

        // Menambahkan pengguna baru ke daftar user ketika tombol register
diklik
        registerButton.setOnClickListener {
            val email = emailInput.text.toString()   // Mengambil email yang
dimasukkan
            val  password  =  passwordInput.text.toString()    //  Mengambil
password yang dimasukkan

            // Menambahkan user baru ke UserData.userList
            UserData.userList.add(User(email, password))

            // Menampilkan pesan sukses
            Toast.makeText(this,        "Akun        berhasil        dibuat",
Toast.LENGTH_SHORT).show()

            // Mengarahkan kembali ke LoginActivity
            startActivity(Intent(this, LoginActivity::class.java))
```

```
            finish() // Menutup RegisterActivity agar tidak dapat kembali ke
halaman registrasi
        }
    }
}
```

Penjelasan:

- Di dalam onCreate(), elemen-elemen UI seperti EditText untuk email dan password, serta tombol Button untuk melakukan registrasi diinisialisasi.
- Ketika tombol registerButton diklik, aplikasi mengambil nilai dari emailInput dan passwordInput lalu membuat objek User baru.
- Objek User ini kemudian ditambahkan ke UserData.userList, yang merupakan sebuah ArrayList<User> tempat menyimpan data pengguna yang terdaftar.
- Setelah itu, aplikasi menampilkan pesan Toast untuk memberi tahu pengguna bahwa akun telah berhasil dibuat.
- Aplikasi kemudian berpindah kembali ke LoginActivity, dan finish() digunakan untuk menutup RegisterActivity agar pengguna tidak dapat kembali lagi ke halaman registrasi dengan tombol back.

Kode Kotlin - User.kt

File User.kt mendefinisikan data class User yang menyimpan informasi mengenai pengguna, yaitu email dan password.

```
object UserData {
    val userList = ArrayList<User>() // Menyimpan daftar user yang telah
mendaftar
}
```

Penjelasan:

- UserData adalah object di Kotlin, yang berarti ini adalah sebuah singleton (hanya ada satu instance dari objek ini di seluruh aplikasi).
- userList adalah ArrayList<User> yang digunakan untuk menyimpan semua pengguna yang telah mendaftar. Pengguna baru ditambahkan ke dalam daftar ini saat mereka berhasil mendaftar melalui RegisterActivity.

Kode Kotlin - UserData.kt

File UserData.kt mendefinisikan objek UserData yang berfungsi untuk menyimpan semua pengguna yang sudah mendaftar.

```
object UserData {
```

```
    val userList = ArrayList<User>() // Menyimpan daftar user yang telah
mendaftar
}
```

Penjelasan:

- UserData adalah object di Kotlin, yang berarti ini adalah sebuah singleton (hanya ada satu instance dari objek ini di seluruh aplikasi).
- userList adalah ArrayList<User> yang digunakan untuk menyimpan semua pengguna yang telah mendaftar. Pengguna baru ditambahkan ke dalam daftar ini saat mereka berhasil mendaftar melalui RegisterActivity.

SplashActivity (Kotlin)

SplashActivity adalah aktivitas pertama yang muncul saat aplikasi dijalankan. Biasanya digunakan untuk menampilkan logo atau animasi pembuka, dan setelah beberapa waktu (biasanya beberapa detik), aplikasi akan melanjutkan ke layar utama, misalnya halaman login atau beranda aplikasi.

```
package com.example.uts

import android.content.Intent
import android.os.Bundle
import android.os.Handler
import android.os.Looper
import androidx.appcompat.app.AppCompatActivity

class SplashActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_splash)

        // Tunggu 2 detik, lalu ke LoginActivity
        Handler(Looper.getMainLooper()).postDelayed({
            startActivity(Intent(this, LoginActivity::class.java))
            finish() // Menutup SplashActivity agar tidak dapat kembali ke
halaman splash
        }, 2000) // Delay 2 detik
    }
}
```

Penjelasan:

- onCreate(): Pada saat aplikasi dijalankan, SplashActivity akan diinisialisasi dan layout activity_splash akan di-set sebagai tampilan.
- Handler(Looper.getMainLooper()).postDelayed({...}, 2000): Fungsi ini mengatur penundaan selama 2 detik. Setelah 2 detik, aplikasi akan memulai LoginActivity menggunakan Intent. Setelah berpindah ke aktivitas login, finish() dipanggil untuk menutup SplashActivity, sehingga pengguna tidak bisa kembali ke halaman splash jika menekan tombol back.
- 2000 adalah waktu tunggu dalam milidetik, yang berarti aplikasi akan beralih ke halaman login setelah 2 detik**.**

MainActivity (Kotlin)

MainActivity adalah halaman utama aplikasi yang muncul setelah pengguna berhasil login. Pada kode ini, RecyclerView digunakan untuk menampilkan daftar item yang dapat diklik atau dipilih, seperti kategori produk.

```
package com.example.uts

import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import androidx.recyclerview.widget.LinearLayoutManager
import androidx.recyclerview.widget.RecyclerView

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        val recyclerView = findViewById<RecyclerView>(R.id.recyclerView)
        recyclerView.layoutManager = LinearLayoutManager(this) // Menggunakan
LinearLayoutManager untuk list vertikal

        // Contoh data barang
        val items = listOf("Kategori Pakaian", "Kategori Elektronik",
"Kategori Perlengkapan Rumah")
        recyclerView.adapter = ItemAdapter(items) // Menyambungkan
RecyclerView dengan Adapter
    }
}
```

Penjelasan:

- onCreate(): Saat MainActivity dibuka, metode ini akan dipanggil. Layout activity_main di-set dan RecyclerView diinisialisasi.
- LinearLayoutManager(this): Layout manager ini digunakan untuk menampilkan item dalam bentuk daftar vertikal. RecyclerView akan menggulirkan item secara vertikal, satu per satu.
- ItemAdapter(items): Adapter ini menghubungkan data items yang berisi kategori produk dengan RecyclerView. Di sini, data kategori yang ditampilkan adalah "Kategori Pakaian", "Kategori Elektronik", dan "Kategori Perlengkapan Rumah".