

## PROYEK INDIVIDU MATA KULIAH

Nama Mata Kuliah	:	Pengembangan Aplikasi Perangkat Bergerak
Kelas	:	TIF - B
Nama Mahasiswa	:	Nadhif Rif'at Rasendriya / 235150201111074
Nama Dosen Pengampu	:	Ir. Adam Hendra Brata, S.Kom., M.T., M.Sc.

### Bab 1 Android Studio

```

@Composable
fun LandingScreen(onStartClick: () -> Unit, modifier: Modifier = Modifier
    .fillMaxSize()
    .padding(all = 32.dp),
    verticalArrangement = Arrangement.Center,
    horizontalAlignment = Alignment.CenterHorizontally
) {
    Column(
        modifier = modifier
            .fillMaxSize()
            .padding(all = 32.dp),
        verticalArrangement = Arrangement.Center,
        horizontalAlignment = Alignment.CenterHorizontally
    ) {
        Text(
            text = "Selamat Datang di Travelupa!",
            style = MaterialTheme.typography.headlineMedium,
            textAlign = TextAlign.Center
        )
        Spacer(modifier = Modifier.height(16.dp))
        Text(
            text = "Solusi buat kamu yang lupa kemana-mana.",
            style = MaterialTheme.typography.bodyLarge,
            textAlign = TextAlign.Center
        )
    }
}

```



Pada tahap awal ini, saya melakukan instalasi dan konfigurasi lingkungan kerja menggunakan Android Studio. Saya membuat *project* baru bernama "**Travelupa**" dengan konfigurasi *Minimum SDK API 24 (Android 7.0)* dan menggunakan *Build Configuration Language* berbasis Kotlin DSL agar sesuai dengan standar pengembangan Android modern.

## Bab 2 Jetpack Compose 1

Implementasi dasar UI *Hello World* atau tampilan statis pada bab ini telah saya lebur dan kembangkan langsung menjadi antarmuka yang lebih kompleks pada Bab 3. Hal ini dilakukan untuk efisiensi penulisan kode dan langsung menerapkan struktur *layout* yang dinamis.

## Bab 3 Jetpack Compose 2

The screenshot shows two instances of the Android Studio interface side-by-side. Both instances have the project structure open, showing the 'app' module with its files like 'Manifests', 'kotlin/java', 'res', and various XML and Kotlin files. The top instance is focused on the 'MainActivity.kt' file, which contains code for defining a list of travel destinations (TempatWisata) using data classes. The bottom instance is focused on the 'build.gradle.kts' file, which defines the project's build configuration, including plugins, dependencies, and default configurations for the app.

**MainActivity.kt (Top Tab)**

```
val daftarTempatWisata = listOf(
    TempatWisata(
        nama = "Tumpak Sewu",
        deskripsi = "Air terjun tercantik di Jawa Timur.",
        gambar = R.drawable.tumpak_sewu
    ),
    TempatWisata(
        nama = "Gunung Bromo",
        deskripsi = "Matahari terbitnya begus banget",
        gambar = R.drawable.gunung_bromo
    ),
    TempatWisata(
        nama = "Gunung Arunika",
        deskripsi = "Ayo main roblox",
        gambar = R.drawable.mountain_arunika
    )
)

class MainActivity : ComponentActivity() {
```

**build.gradle.kts (Bottom Tab)**

```
plugins {
    alias(libs.plugins.android.application)
    alias(libs.plugins.kotlin.android)
    alias(libs.plugins.kotlin.compose)
}

android {
    namespace = "com.example.travelupa"
    compileSdk = 36

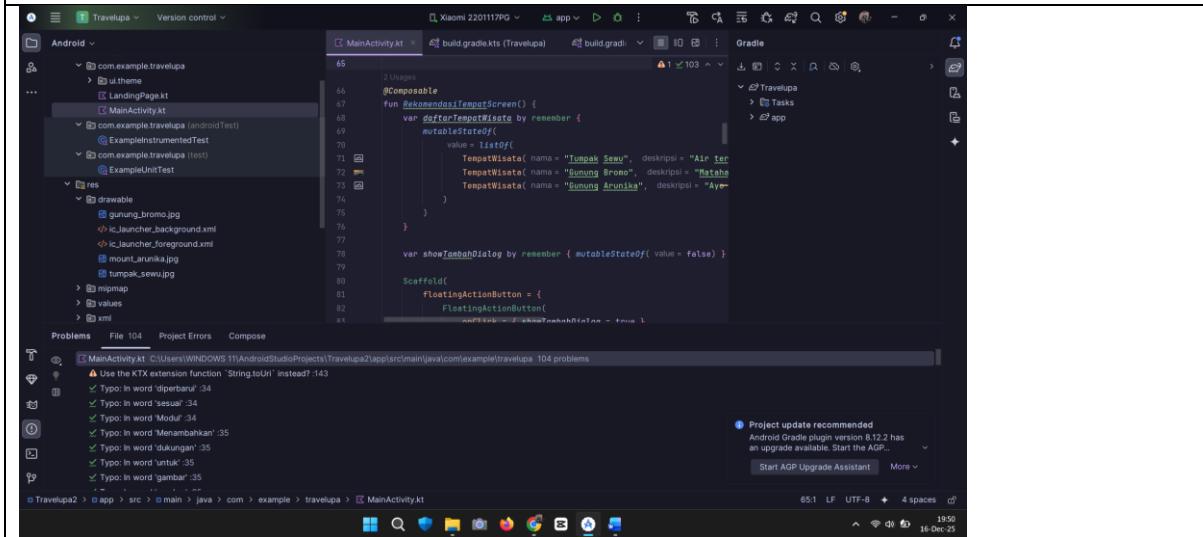
    defaultConfig {
        applicationId = "com.example.travelupa"
        minSdk = 24
        targetSdk = 36
        versionCode = 1
        versionName = "1.0"

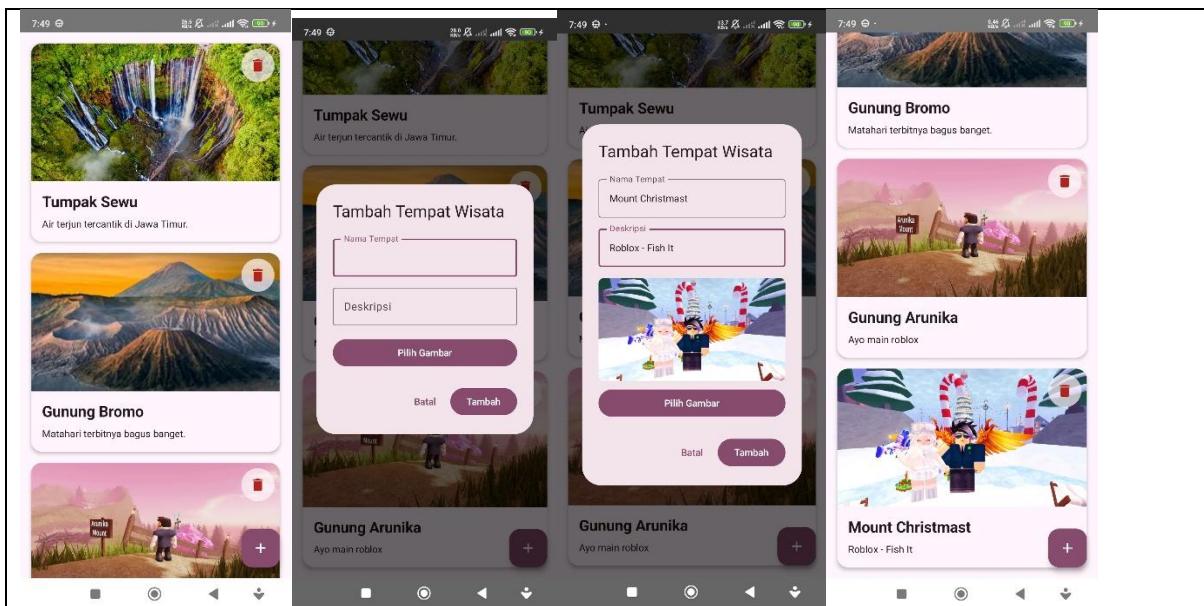
        testInstrumentationRunner = "androidx.test.runner.AndroidJUnitRunner"
    }
}
```



Pada bab ini, saya membangun tampilan utama aplikasi (*Home Screen*) menggunakan **Jetpack Compose**. Saya memanfaatkan komponen `LazyColumn` untuk membuat daftar tempat wisata yang dapat di-*scroll* secara efisien. Setiap *item* wisata ditampilkan menggunakan komponen `Card` yang membungkus `Image` dan `Text` agar tampilan terlihat rapi dan menarik.

## Bab 4 UI State





Di sini saya menerapkan konsep *State Management* agar aplikasi menjadi interaktif. Saya menggunakan `mutableStateOf` dan `remember` untuk memantau perubahan data. Fitur yang saya tambahkan meliputi **Floating Action Button (FAB)** yang memunculkan *Dialog* input untuk menambah data wisata baru, serta tombol hapus pada setiap item *card* yang dapat memperbarui tampilan *list* secara *real-time*.

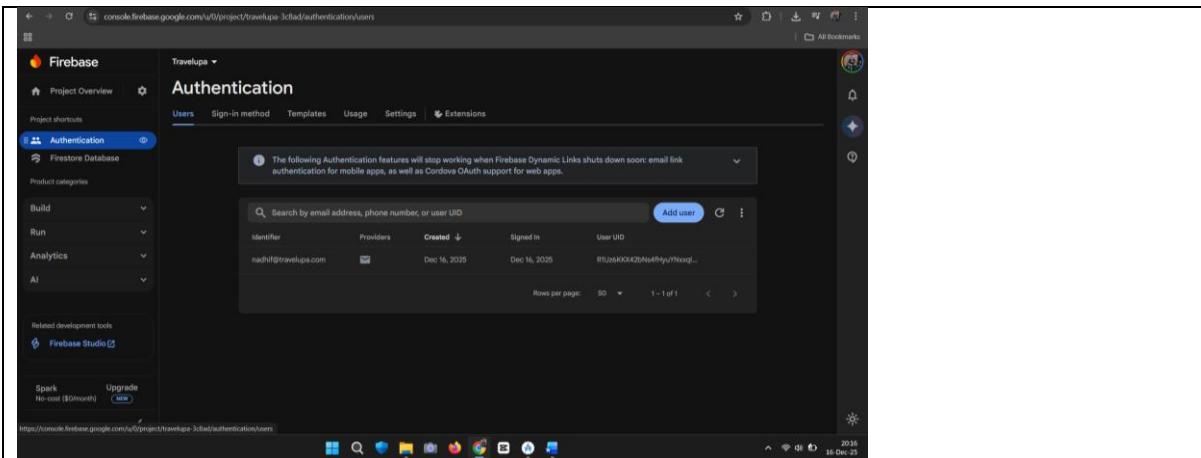
## Bab 5 Firebase dan REST API

Tunjukkan screenshot dari Firebase Console dimana project dibuat

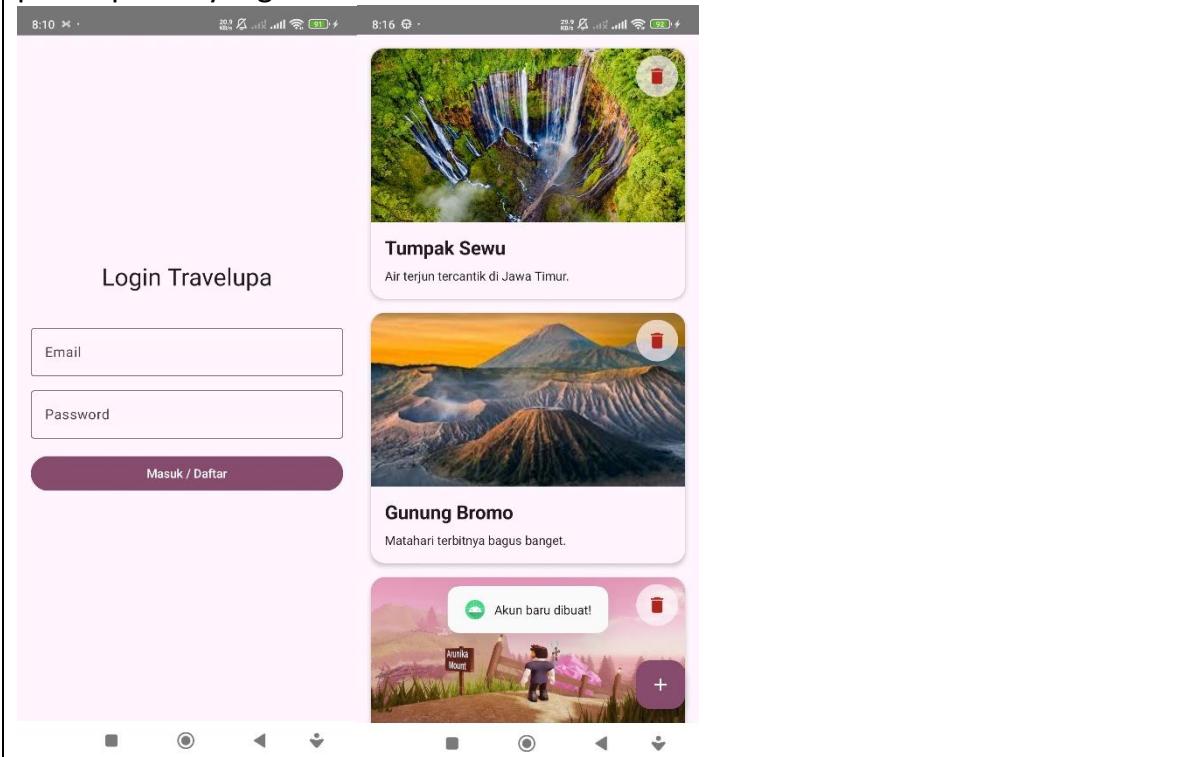
The screenshot displays two windows of the Firebase Console.

The top window shows the "Create a database" dialog. It has sections for "Start in production mode" (with a note about security rules) and "Start in test mode" (with a note about test mode rules). A button at the bottom right says "Create".

The bottom window shows the "Firestore Database" section of the "Travelupa" project. It lists "Cloud Firestore" and "Database". Under "Database", there is a collection named "(default)" with a "Start collection" button. The status bar at the bottom indicates "Your database is ready to go... just add data".



Tunjukkan screenshot dari UI tampilan yang menunjukkan penerapan Firebase pada aplikasi yang sudah dibuat



Saya menghubungkan aplikasi Travelupa dengan layanan **Google Firebase**. Langkah yang saya lakukan meliputi konfigurasi google-services.json, mengaktifkan fitur **Firebase Authentication** untuk membuat halaman Login pengguna, serta menyiapkan **Cloud Firestore** sebagai basis data *cloud* untuk menyimpan informasi nama dan deskripsi tempat wisata agar dapat diakses secara *online*.

## Bab 6 Kotlin Coroutines

Tunjukkan screenshot dari Android Studio yang menunjukkan kode yang dibuat untuk menunjukkan penerapan Kotlin Coroutines pada aplikasi yang sudah dibuat

The image displays two side-by-side screenshots of the Android Studio interface. Both screens show the same project structure and code editor for a file named `MainActivity.kt`. The code implements a `TempatWisata` class and a `RekomendasiTempatScreen` function using `Kotlin Coroutines`.

```
data class TempatWisata(
    val name: String,
    val deskripsi: String,
    val gambarId: Int? = null,
    val gambarUrlString: String? = null
)

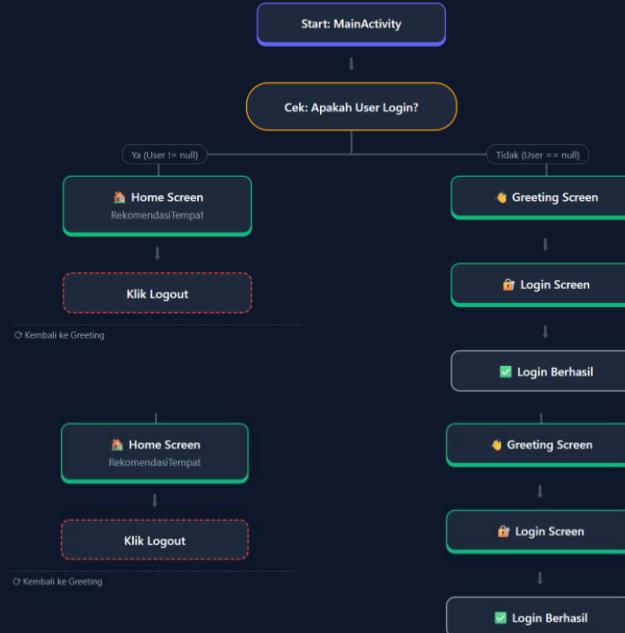
@OptIn(markerClass = ExperimentalMaterialApi::class)
@Composable
fun RekomendasiTempatScreen(
    onBackToLogin: () -> Unit
) {
    var daftarTempatWisata by remember {
        mutableStateOf(
            listOf(
                TempatWisata(name = "Tumpak Sewu", deskripsi = "Air terjun"),
                TempatWisata(name = "Gunung Bromo", deskripsi = "Batu bara"),
                TempatWisata(name = "Gunung Arjuna", deskripsi = "Awan asap")
            )
        )
    }
}
```

The bottom right corner of each screenshot shows the system status bar with the date and time.

Pada bab ini, saya mengimplementasikan **Kotlin Coroutines** untuk menangani proses **asynchronous** (latar belakang) agar tidak mengganggu performa UI utama. Penerapan utamanya ada pada proses Login dan pengecekan sesi pengguna (`currentUser`). Fitur ini memungkinkan aplikasi mendeteksi jika pengguna sudah `login` sebelumnya, sehingga aplikasi dapat langsung diarahkan ke halaman utama tanpa harus memasukkan `password` ulang.

## Bab 7 Jetpack Navigation

Gambarkan alur navigasi yang sudah dibuat dalam program



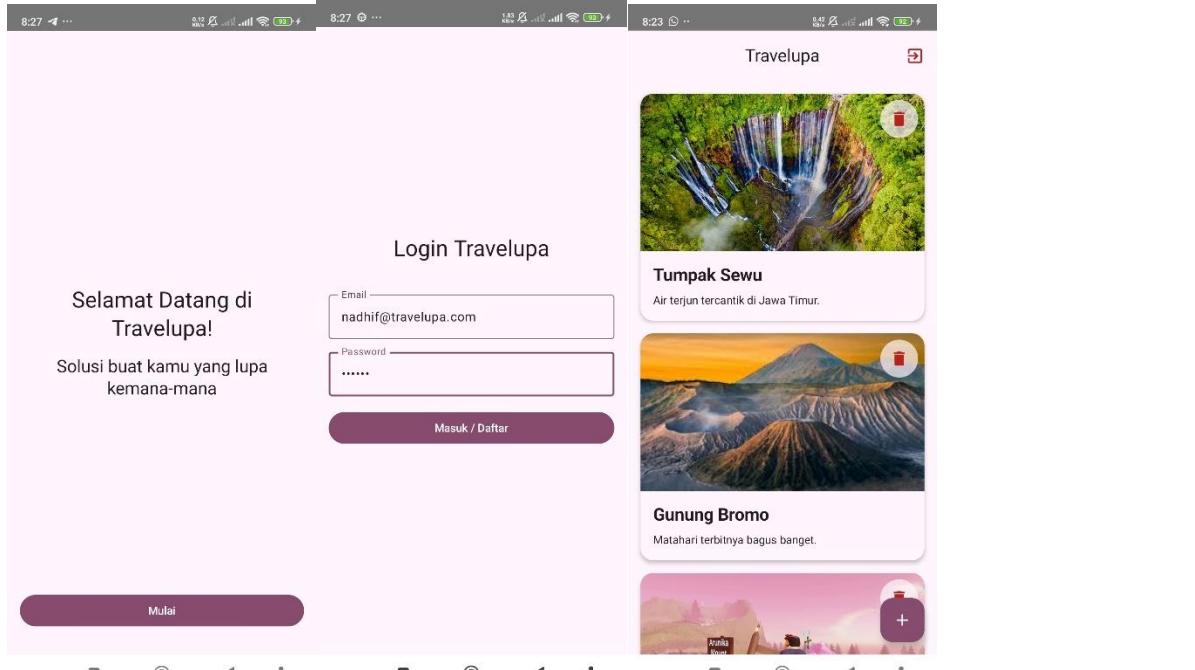
## Penjelasan Alur Program

- 1. Start (MainActivity):** Aplikasi dimulai dengan memeriks 'currentUser' dari Firebase.
  - 2. Logika Percabangan:**
    - Jalur Kiri (Sudah Login):** User langsung masuk ke Home Screen. Tombol Logout akan mengembalikan user ke Greeting.
    - Jalur Kanan (Belum Login):** User disambut di Greeting Screen, lalu menekan tombol Jalur Mulai untuk Login. Jika sukses, user akan dipindahkan ke Home.

The screenshot shows the Android Studio interface with the following details:

- Project Structure:** The left sidebar shows the project structure under the "Android" tab. It includes modules like "app", "kotlin+java", and "test".
- MainActivity.kt:** The main code editor window displays the `MainActivity.kt` file. The code implements a `AppNavigation` function using Compose navigation.
- Code Inspection:** A tooltip is visible at the bottom left, indicating multiple "Typo" errors in the word "Definiskan".
- Bottom Navigation:** The bottom navigation bar shows the current file path: `Travelupa2 > app > src > main > java > com > example > travelupa > MainActivity.kt`.

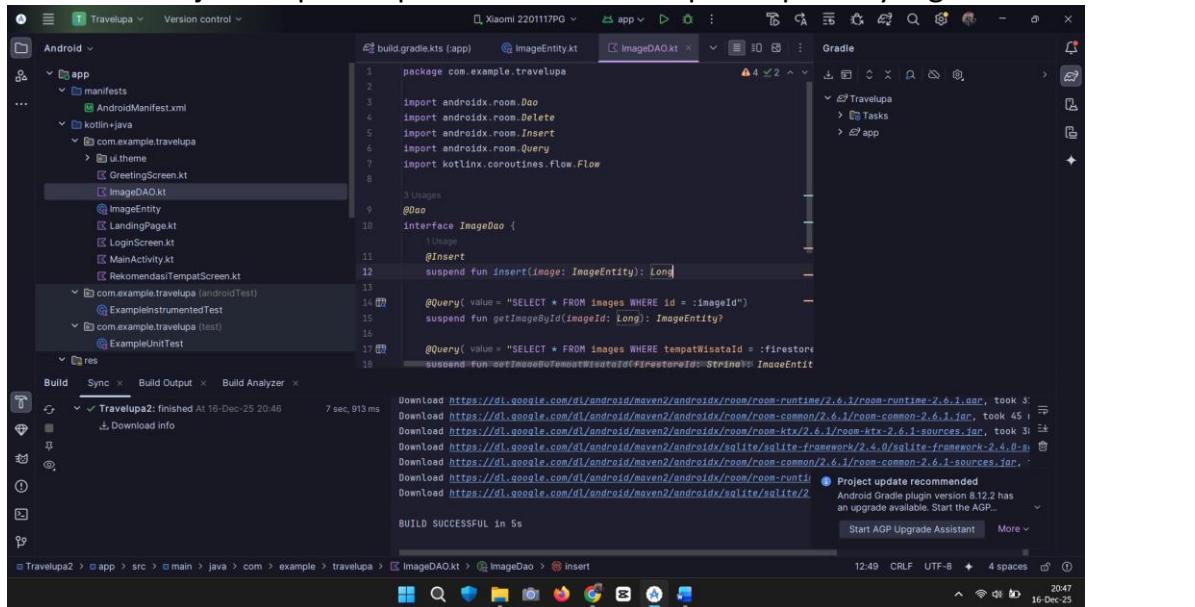
Tunjukkan screenshot dari UI tampilan yang menunjukkan penerapan Navigation



Saya menyusun alur navigasi antar layar menggunakan **Jetpack Navigation Component**. Saya membuat NavHost yang mengatur rute perpindahan dari **Greeting Screen** (Halaman Sapaan) -> **Login Screen** -> **Home Screen**. Saya juga menambahkan logika *PopUpTo* untuk membersihkan riwayat navigasi agar pengguna tidak kembali ke halaman login saat menekan tombol *back* di halaman utama.

## Bab 8 Room Database

Tunjukkan screenshot dari Android Studio yang menunjukkan kode yang dibuat untuk menunjukkan penerapan Room Database pada aplikasi yang sudah dibuat



```

1 package com.example.travelupa
2
3 import androidx.room.Entity
4 import androidx.room.PrimaryKey
5
6 @Entity(tableName = "Images")
7 data class ImageEntity(
8     @PrimaryKey(autoGenerate = true)
9     val id: Long = 0,
10    val localPath: String,
11    val tempatWisataId: String? = null
12 )

```

```

35 }
36
37 class MainActivity : ComponentActivity() {
38     private lateinit var db: AppDatabase
39
40     private lateinit var imageDao: ImageDao
41
42     override fun onCreate(savedInstanceState: Bundle?) {
43         super.onCreate(savedInstanceState)
44         FirebaseApp.initializeApp(context = this)
45
46         db = Room.databaseBuilder(
47             applicationContext,
48             AppDatabase::class.java, name = "travelupa-database"
49         ).build()
50         imageDao = db.imageDao()
51
52     }
53
54     val currentUser: FirebaseAuthUser? = FirebaseAuth.getInstance().currentUser

```

Untuk meningkatkan kemampuan aplikasi dalam menyimpan data gambar secara persisten (tetap ada meski aplikasi ditutup), saya mengimplementasikan **Room Database (SQLite)**. Saya membuat struktur tabel (Entity), akses data (DAO), dan inisialisasi database (AppDatabase). Saya juga membuat logika sinkronisasi dimana gambar disimpan secara fisik di penyimpanan lokal perangkat, dan *path*-nya dicatat di Room serta disinkronkan ke Firestore.

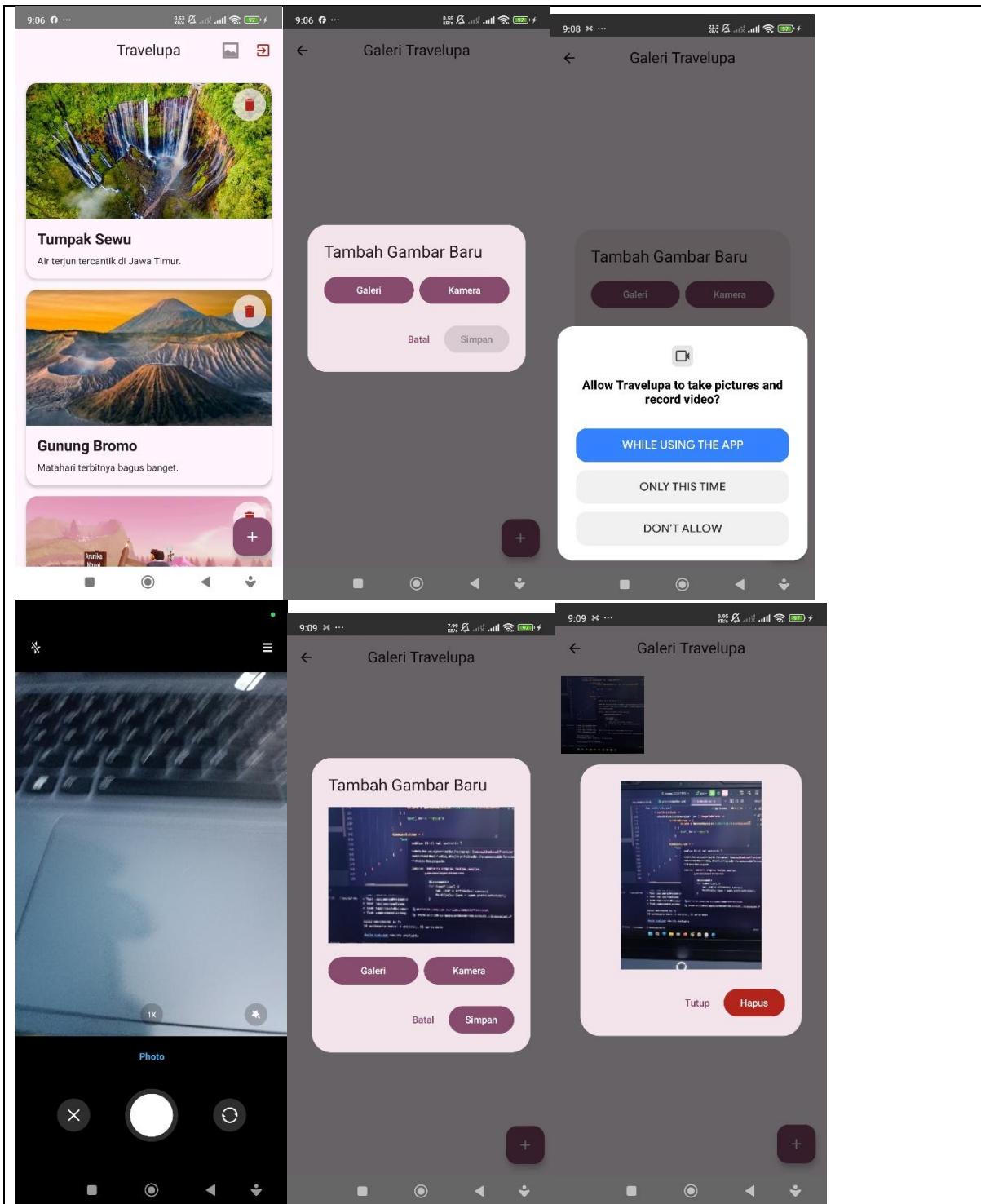
## Bab 9 CameraX

Tunjukkan screenshot dari Android Studio yang menunjukkan kode yang dibuat untuk menunjukkan penerapan CameraX pada aplikasi yang sudah dibuat

```
1 Usage
71    @Composable
72        fun AppNavigation(
73            currentUser: FirebaseAuthUser?,
74            firestore: FirebaseFirestore,
75            imageDao: ImageDAO,
76            context: Context
77        ) {
78            val navController = rememberNavController()
79            val startScreen = if (currentUser != null) Screen.Home.route else
80                Screen.Login.route
81
82            NavHost(navController, startDestination = startScreen) {
83                composable(route = Screen.Greeting.route) {
84                    GreetingScreen(
85                        onStart = {
86                            navController.navigate(route = Screen.Login.route)
87                        }
88                    )
89                }
90            }
91        }
92    
```

```
39    fun RekomendasiTempatScreen(
93        var showTambahDialog by remember { mutableStateOf(false) }
94    ) {
95        Scaffold(
96            topBar = {
97                CenterAlignedTopAppBar(
98                    title = { Text(text = "Travelupa") },
99                    actions = {
100                        IconButton(onClick = onGoToGallery) {
101                            Icon(
102                                painter = painterResource(id = android.R.drawable.ic_menu_gallery),
103                                contentDescription = "Galeri"
104                            )
105                        }
106                        IconButton(onClick = onBackToLogin) {
107                            Icon(
108                                imageVector = Icons.AutoMirrored.Filled.Exit,
109                                contentDescription = "Logout",
110                                tint = MaterialTheme.colorScheme.error
111                            )
112                        }
113                    }
114                )
115            }
116        )
117    }
118
```

Tunjukkan screenshot dari UI tampilan yang menunjukkan penerapan CameraX



Pada bab terakhir ini, saya menambahkan fitur multimedia dengan mengintegrasikan akses kamera dan galeri. Saya membuat halaman baru yaitu **GalleryScreen** yang menampilkan *grid* foto-foto wisata. Saya menggunakan *Activity Result Contracts* untuk meminta izin kamera (*Runtime Permission*) dan memungkinkan pengguna mengambil foto langsung dari aplikasi atau memilihnya dari galeri HP, yang kemudian tersimpan otomatis ke dalam database aplikasi.

### **Link Project File**

Tuliskan alamat link dari Repository project yang sudah dibuat (GitHub, Gitlab, dll)

Pastikan dapat dibuka secara publik untuk penilaian

<https://github.com/nadhif-royal/Travelupa/>