

LAPORAN TEORI MOBILE PROGRAMMING
MODUL 12



Nama : Firman Fadilah Noor
NIM : 240605110083
Kelas : B
Tanggal : 1 November 2025

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
GANJIL 2025/2026

I. Tujuan

1. Memahami cara pengambilan Lokasi perangkat menggunakan GPS di Flutter.
2. Menerapkan paket geolocation dan geocoding.
3. Penanganan izin Lokasi dan eror.
4. Pembuatan antarmuka interaktif yang menampilkan informasi Lokasi secara real time.

II. Langkah Kerja

1. Penambahan izin Lokasi di `AndroidManifest.xml`.
2. Menambahkan depensi pada file `pubspec.yaml`.
3. Penambahan UI menggunakan Scaffold, Card, dan ElevatedButton.
4. Implementasi fungsi `_getLocation()` beserta logika pengecekan layanan Lokasi, permintaan izin, pengambilan posisi, dan reverse geocoding.
5. Penggunaan `setState()` untuk memperbarui tampilan berdasarkan perubahan data Lokasi.

III. Screenshot Hasil

a. Kode Program

- `main.dart`

```
import 'package:flutter/material.dart';
import 'package:google_fonts/google_fonts.dart';
import 'package:geolocator/geolocator.dart';
import 'package:geocoding/geocoding.dart';

void main() {
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Lokasi Saya',
      debugShowCheckedModeBanner: false,
      theme: ThemeData(
        primarySwatch: Colors.indigo,
        textTheme: GoogleFonts.poppinsTextTheme(),
      ),
      home: const GeolocationScreen(),
    );
  }
}

class GeolocationScreen extends StatefulWidget {
  const GeolocationScreen({super.key});

  @override
  State<GeolocationScreen> createState() => _GeolocationScreenState();
}

class _GeolocationScreenState extends State<GeolocationScreen> {
  String? _kecamatan;
  String? _kota;
  bool _isLoading = false;
  String? _errorMessage;
```

codesnap.dev

```
Future<void> _getLocation() async {
  setState(() {
    _isLoading = true;
    _errorMessage = null;
    _kecamatan = null;
    _kota = null;
  });
  try {
    // Cek apakah GPS ON
    bool serviceEnabled = await Geolocator.isLocationServiceEnabled();
    if (!serviceEnabled) {
      throw Exception('GPS tidak aktif. Harap aktifkan terlebih dahulu.');
    }

    // Cek izin lokasi
    LocationPermission permission = await Geolocator.checkPermission();
    if (permission == LocationPermission.denied) {
      permission = await Geolocator.requestPermission();
      if (permission == LocationPermission.denied) {
        throw Exception('Izin lokasi ditolak.');
      }
    }

    if (permission == LocationPermission.deniedForever) {
      throw Exception(
        'Izin lokasi ditolak permanen. Buka pengaturan aplikasi.');
    }

    // Ambil posisi saat ini
    Position position = await Geolocator.getCurrentPosition(
      desiredAccuracy: LocationAccuracy.high,
    );

    // Reverse geocoding → alamat
    List<Placemark> placemarks = await placemarkFromCoordinates(
      position.latitude, position.longitude);

    if (placemarks.isNotEmpty) {
      Placemark place = placemarks[0];

      setState(() {
        _kecamatan = place.subLocality;
        _kota = place.subAdministrativeArea;
      });
    } else {
      throw Exception('Alamat tidak ditemukan.');
    }
  } catch (e) {
    setState(() {
      _errorMessage = e.toString().replaceAll("Exception:", "").trim();
    });
  } finally {
    setState(() {
      _isLoading = false;
    });
  }
}
```

codesnap.dev

```
override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: const Text(
        'Lokasi Saya',
        style: TextStyle(color: Colors.white),
      ),
      backgroundColor: const Color(0xFF1A237E),
    ),
    body: Padding(
      padding: const EdgeInsets.all(10.0),
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        crossAxisAlignment: CrossAxisAlignment.stretch,
        children: [
          Card(
            elevation: 4.0,
            shape: RoundedRectangleBorder(
              borderRadius: BorderRadius.circular(12.0)),
            child: Container(
              padding: const EdgeInsets.symmetric(
                vertical: 32.0, horizontal: 10.0),
              child: _isLoading
                  ? const Center(child: CircularProgressIndicator())
                  : (_errorMessage != null)
                      ? Text(
                          _errorMessage,
                          textAlign: TextAlign.center,
                          style: const TextStyle(color: Colors.red),
                        )
                      : (_kecamatan == null && _kota == null)
                          ? const Text(
                              'Tekan tombol untuk menampilkan lokasi.',
                              textAlign: TextAlign.center,
                            )
                          : Column(
                              crossAxisAlignment: CrossAxisAlignment.start,
                              children: [
                                Text(
                                  'Kelurahan/Kecamatan: ${_kecamatan}',
                                  style: const TextStyle(
                                    fontSize: 16,
                                    fontWeight: FontWeight.bold,
                                  ),
                                ),
                                const SizedBox(height: 10),
                                Text(
                                  'Kota: ${_kota}',
                                  style: const TextStyle(
                                    fontSize: 16,
                                    fontWeight: FontWeight.bold,
                                  ),
                                ),
                              ],
                            ),
                  ],
                ),
            ),
            const SizedBox(height: 40),
            ElevatedButton(
              onPressed: _isLoading ? null : _getLocation,
              style: ElevatedButton.styleFrom(
                backgroundColor: const Color(0xFF1A237E),
                padding: const EdgeInsets.symmetric(vertical: 16.0),
                shape: RoundedRectangleBorder(
                  borderRadius: BorderRadius.circular(12.0),
                ),
                elevation: 5.0,
              ),
              child: const Text(
                'TAMPIKAN LOKASI SAATINI',
                style: TextStyle(
                  fontSize: 16,
                  color: Colors.white,
                ),
              ),
            ),
            const SizedBox(
              height: 10,
            ),
          ),
        ],
      ),
    ),
  );
}
```

b. Penjelasan Kode Program.

1. Menambahkan Izin Lokasi di AndroidManifest.xml

Langkah pertama adalah memberikan izin akses lokasi pada perangkat Android. Tanpa izin ini, aplikasi tidak akan bisa mengambil koordinat GPS.

Izin yang ditambahkan:

- Akses lokasi akurat
- Akses lokasi tidak akurat
- Izin saat aplikasi berjalan

Tujuannya agar aplikasi dapat:

- Mengecek status GPS
- Mengambil lokasi secara real-time
- Melakukan reverse geocoding

2. Menambahkan Dependensi di pubspec.yaml

Langkah berikutnya yaitu menambahkan **package pendukung** agar aplikasi bisa bekerja dengan fitur geolokasi.

Dependency yang ditambahkan:

- `geolocator` → mengambil koordinat & cek izin lokasi
- `geocoding` → mengubah koordinat menjadi alamat
- `google_fonts` → mempercantik tampilan teks

Setelah menambahkannya, jalankan flutter pub get untuk meng-install package.

3. Penambahan UI Menggunakan Scaffold, Card, dan ElevatedButton

Selanjutnya dibuat tampilan utama aplikasi menggunakan beberapa komponen UI:

- **Scaffold** → kerangka dasar halaman
- **AppBar** → menampilkan judul “Lokasi Saya”
- **Card** → menampilkan hasil lokasi, error, atau instruksi
- **ElevatedButton** → tombol untuk memulai proses pengambilan lokasi

Desain dibuat sederhana tapi rapi sehingga pengguna mudah memahami fungsi aplikasi.

4. Implementasi Fungsi _getLocation()

Ini merupakan bagian inti dari aplikasi.

Di dalam fungsi ini terdapat beberapa proses penting, yaitu:

◆ a. **Mengecek apakah layanan Lokasi (GPS) aktif**

Jika GPS mati → aplikasi memberi pesan error agar pengguna mengaktifkannya.

◆ b. **Mengecek dan meminta izin lokasi**

- Jika izin *denied*, aplikasi meminta izin lagi.
- Jika *denied forever*, aplikasi memberi instruksi untuk membuka pengaturan.

◆ c. **Mengambil posisi pengguna**

Jika semua izin terpenuhi, aplikasi mengambil:

- latitude
- longitude

Dengan akurasi tinggi.

◆ **d. Reverse geocoding**

Koordinat yang diperoleh kemudian diubah menjadi alamat lengkap.

Bagian yang diambil:

- Kecamatan / kelurahan → subLocality
- Kota → subAdministrativeArea

◆ **e. Menangani error**

Setiap kemungkinan error ditangani oleh blok try-catch supaya aplikasi tidak crash.

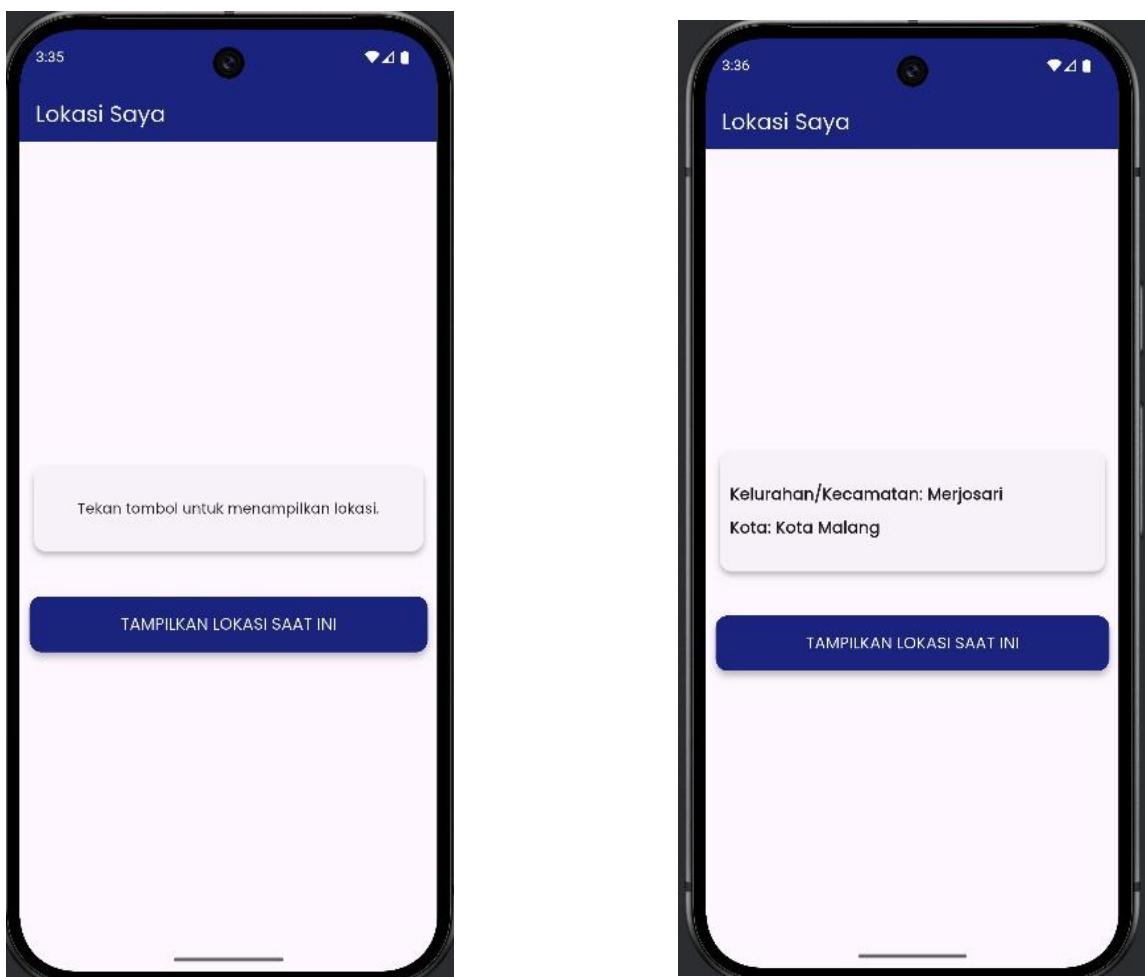
5. Penggunaan setState() untuk Memperbarui Tampilan

Setiap kali data lokasi berubah, method setState() dipanggil untuk:

- Mengupdate tampilan Card
- Menghilangkan loading indicator
- Menampilkan error jika ada
- Menampilkan hasil Kecamatan dan Kota jika berhasil

setState() memastikan UI selalu menampilkan data terbaru berdasarkan proses pengambilan lokasi.

Output:



IV. Kesimpulan.

Dari praktikum ini, dapat disimpulkan bahwa proses untuk mengambil lokasi pengguna di Flutter ternyata membutuhkan beberapa langkah penting yang saling mendukung. Mulai dari memberikan izin lokasi di *AndroidManifest.xml*, menambahkan dependency yang dibutuhkan, hingga membangun tampilan aplikasi yang sederhana namun informatif. Semua hal ini menjadi dasar agar aplikasi bisa bekerja dengan baik.

Melalui fungsi `_getLocation()`, aplikasi dapat mengecek apakah GPS aktif, meminta izin lokasi kepada pengguna, mengambil koordinat latitude dan longitude, lalu mengubahnya menjadi alamat seperti kecamatan dan kota. Setiap perubahan data kemudian ditampilkan secara langsung menggunakan `setState()`, sehingga pengguna bisa langsung melihat hasilnya tanpa perlu melakukan apa pun lagi.

Secara keseluruhan, praktikum ini memberikan pemahaman bahwa Flutter sangat mendukung fitur geolokasi dan mampu menampilkan informasi lokasi secara real time. Selain itu, penanganan error yang baik juga membantu aplikasi menjadi lebih stabil dan nyaman digunakan. Praktikum ini menjadi pengalaman yang bermanfaat untuk memahami bagaimana mengombinasikan izin, package, logika aplikasi, dan tampilan agar fitur lokasi dapat berjalan dengan lancar.