

**TUGAS UNGUIDED  
PEMROGRAMAN PERANGKAT BERGERAK**

**MODUL XI  
DATA STORAGE LANJUTAN II**



**Disusun Oleh :  
TEGAR KANG AGENG GILANG  
2311104018  
SE 0701**

**Asisten Praktikum :  
Nita Fitrotul Mar'ah  
Aflah Rizkyadhafin Nurfikri**

**Dosen Pengampu :  
Yudha Islami Sulistya, S.Kom., M.Cs.**

**PROGRAM STUDI S1 SOFTWARE ENGINEERING  
FAKULTAS INFORMATIKA  
TELKOM UNIVERSITY PURWOKERTO  
2025**

# TUGAS UNGUIDED

## A. SOAL

### Tugas Mandiri (Unguided)

#### 1. (Soal)

Buatlah sebuah database beserta tabel di dalam **Supabase** untuk menyimpan data **Biodata Buku** pada perpustakaan digital sederhana. Data buku terdiri dari:

- Judul
- Penulis
- Tahun Terbit
- Genre

Data yang ditambahkan harus disimpan ke Supabase dan ditampilkan melalui terminal.

#### Alur Pengerjaan:

##### A. Database

Buat database baru di Supabase lalu tambahkan tabel 'perpustakaan' yang memuat struktur berikut:

Kolom	Tipe data	Keterangan
id	uuid	PK
judul	text	-
penulis	text	-
tahun_terbit	integer	-
genre	text	-
created_at	timestamp	default:now()

B. Tampilkan data yang telah diambil dari database melalui terminal, lalu screenshot hasilnya.

C. Lakukan operasi Create dan Read untuk menampilkan data dalam format yang berbeda.

```
flutter: [{"id": "10c73b1c-560a-4800-9c4d-a73570e2a19c", "created_at": "2025-11-16T14:55:59.5691+00:00", "judul": "Bumi", "penulis": "Tere Liye", "tahun_terbit": 2014, "genre": "Fantasi"}]
```

## B. JAWABAN

### Source code main:

```
import 'package:flutter/material.dart';
import 'package:unguided/connection.dart';

void main() {
  WidgetsFlutterBinding.ensureInitialized();
  Connection();
  runApp(const MyApp());
}

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

  // This widget is the root of your application.
  @override
```

```

Widget build(BuildContext context) {
  return MaterialApp(
    title: 'Flutter Demo',
    theme: ThemeData(
      // This is the theme of your application.
      //
      // TRY THIS: Try running your application with "flutter run".
      You'll see
      // the application has a purple toolbar. Then, without quitting
      the app,
      // try changing the seedColor in the colorScheme below to
      Colors.green
      // and then invoke "hot reload" (save your changes or press the
      "hot
      // reload" button in a Flutter-supported IDE, or press "r" if you
      used
      // the command line to start the app).
      //
      // Notice that the counter didn't reset back to zero; the
      application
      // state is not lost during the reload. To reset the state, use
      hot
      // restart instead.
      //
      // This works for code too, not just values: Most code changes can
      be
      // tested with just a hot reload.
      colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
    ),
    home: const MyHomePage(title: 'Flutter Demo Home Page'),
  );
}

class MyHomePage extends StatefulWidget {
  const MyHomePage({super.key, required this.title});

  final String title;

  @override
  State<MyHomePage> createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: Text(widget.title),
        ),
        body: Center(
          child: Column(

```

```

        mainAxisAlignment: MainAxisAlignment.center,
        children: <Widget>[
          ElevatedButton(
            onPressed: () {
              insertData({
                "judul": "3726MDPL",
                "penulis": "Nurwina Sari",
                "tahun_terbit": "2024",
                "genre": "Romance",
              });
            },
            child: const Text("Tambah Data"),
          ),
          ElevatedButton(
            onPressed: () {
              getData();
            },
            child: const Text("Lihat Data"),
          ),
          ElevatedButton(
            onPressed: () {
              updateData({
                "id": "",
                "tahun_terbit": "2025",
                "genre": "SOL",
              });
            },
            child: const Text("Update Data"),
          ),
          ElevatedButton(
            onPressed: () {
              deleteData("f40f6591-fda2-49eb-93dd-569f5e06d095");
            },
            child: const Text("Hapus Data"),
          ),
        ],
      ),
    ),
  ),
);
}
}

```

#### Source code connection:

```

import 'package:supabase_flutter/supabase_flutter.dart';

void Connection() async {
  await Supabase.initialize(
    url: "https://sozgnocklidlqxeuiorw.supabase.co",
    anonKey:
"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSIsInJlZiI6InNv
emdub2NrbGlkbHF4ZXVpb3J3Iiwicm9sZSI6ImFub24iLCJpYXQiOiJlZ3NjI5MDgxNTIsImV4cC
I6ImJA3ODQ4NDE1Mn0.1ebtojPjQQcgHSTCA-J2FnVtAkPmyMGfp-UQSn2E8aI",

```

```

    );
}

final db = Supabase.instance.client;
String table = "perpustakaan";

Future<void> insertData(Map<String, dynamic> row) async {
  try {
    await db.from(table).insert(row);

    print("data berhasil terkirim");
  } catch (e) {
    print(e.toString());
  }
}

Future<List> getData() async {
  final fetchData = await db.from(table).select();
  print(fetchData.toList().toString());

  return fetchData;
}

Future<void> updateData(Map<String, dynamic> row) async {
  try {
    await db.from(table).update(row).eq("id", row["id"]);

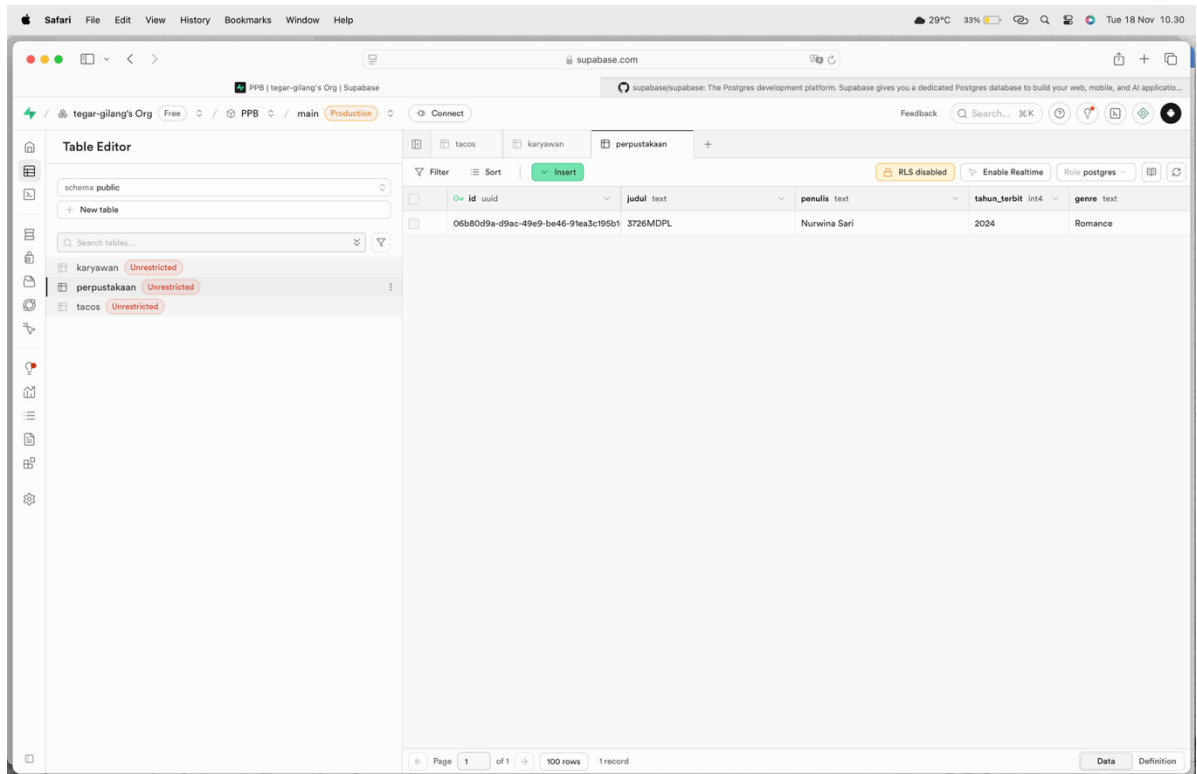
    print("Update data ${row["id"]} berhasil");
  } catch (e) {
    print(e.toString());
  }
}

Future<void> deleteData(String id) async {
  try {
    await db.from(table).delete().eq("id", id);

    print("Hapus data $id berhasil");
  } catch (e) {
    print(e.toString());
  }
}

```

## Screenshot Output



```
A Dart VM Service on Chrome is available at: http://127.0.0.1:51540/GlK0yQBywdE=
Starting application from main method in: org-dartlang-app:/web_entrypoint.dart.
supabase.supabase_flutter: INFO: ***** Supabase init completed *****
The Flutter DevTools debugger and profiler on Chrome is available at:
http://127.0.0.1:9101?uri=http://127.0.0.1:51540/GlK0yQBywdE=
data berhasil terkirim
[{id: 06b80d9a-d9ac-49e9-be46-91ea3c195b16, judul: 3726MDPL, penulis: Nurwina Sari,
tahun_terbit: 2024, genre: Romance, created_at: 2025-11-18T03:30:39.129284+00:00}]
```

perpustakaan		
id	uuid	
judul	text	
penulis	text	
tahun_terbit	int4	
genre	text	
created_at	timestampz	

## Deskripsi Program

Kode di atas merupakan implementasi aplikasi Flutter sederhana yang terhubung dengan Supabase sebagai backend untuk melakukan operasi CRUD (Create, Read, Update, Delete) pada sebuah tabel bernama *perpustakaan*. Pada bagian `main()`, aplikasi terlebih dahulu melakukan inisialisasi Flutter

melalui `WidgetsFlutterBinding.ensureInitialized()` dan menjalankan fungsi `Connection()` untuk menghubungkan aplikasi ke Supabase menggunakan URL dan anon key yang telah ditentukan. Setelah koneksi berhasil dibangun, aplikasi dijalankan melalui `runApp(const MyApp())`.

Kelas `MyApp` berfungsi sebagai root widget aplikasi dan menggunakan `MaterialApp` untuk mengatur tema serta halaman awal. Halaman utama aplikasi didefinisikan oleh widget `MyHomePage`, yang merupakan *StatefulWidget* karena berisi elemen UI yang berpotensi berubah. Pada method `build()` di dalam `_MyHomePageState`, dibuat sebuah tampilan sederhana menggunakan `Scaffold` yang terdiri dari `AppBar` dan beberapa tombol di dalam layout `Column`, masing-masing terhubung dengan fungsi CRUD Supabase.

Tombol “Tambah Data” memanggil fungsi `insertData()` yang mengirim data buku baru seperti judul, penulis, tahun terbit, dan genre ke tabel Supabase. Tombol “Lihat Data” menjalankan fungsi `getData()` untuk mengambil seluruh data dari tabel dan menampilkannya pada console. Tombol “Update Data” memanggil fungsi `updateData()` untuk mengubah data berdasarkan ID tertentu. Sedangkan tombol “Hapus Data” menjalankan fungsi `deleteData()` untuk menghapus satu baris data berdasarkan ID yang diberikan.

Bagian akhir kode berisi implementasi fungsi-fungsi CRUD menggunakan Supabase Client. Fungsi `insertData()` menambahkan baris baru ke tabel, `getData()` mengambil semua data, `updateData()` memperbarui baris berdasarkan kolom ID, dan `deleteData()` menghapus baris tertentu. Seluruh operasi dibungkus dengan blok `try-catch` untuk menampilkan pesan error jika terjadi kegagalan. Dengan demikian, kode ini membentuk sebuah aplikasi Flutter yang terintegrasi dengan backend Supabase untuk mengelola data perpustakaan secara real-time.