

**LAPORAN PRAKTIKUM GUIDED & UNGUIDED
PEMROGRAMAN PERANGKAT BERGERAK**

**MODUL X
DATA STORAGE (BAGIAN I)**



Disusun Oleh :

Fikri Khairul Fajri/2211104052

SE-06-02

Asisten Praktikum :

Muhammad Faza Zulian Gesit Al Barru

Aisyah Hasna Aulia

Dosen Pengampu :

Yudha Islami Sulistya, S.Kom., M.Cs.

**PROGRAM STUDI S1 SOFTWARE ENGINEERING
FAKULTAS INFORMATIKA**

TELKOM UNIVERSITY PURWOKERTO

2024

TUGAS PENDAHULUAN

SOAL

1. Jelaskan secara singkat fungsi SQLite dalam pengembangan aplikasi mobile!
2. Apa saja yang dimaksud dengan operasi CRUD? Berikan penjelasan singkat untuk masing-masing operasi!
3. Tuliskan kode SQL untuk membuat tabel bernama *users* dengan kolom berikut :
 - id (integer, primary key, auto increment)
 - name (text)
 - email (text)
 - createdAt (timestamp, default value adalah waktu sekarang)
4. Sebutkan langkah-langkah utama untuk menggunakan plugin sqflite di dalam Flutter!
5. Lengkapi kode berikut untuk membaca semua data dari tabel *users* menggunakan sqflite.

```
static Future<List<Map<String, dynamic>>> getUsers() async {  
  final db = await SQLHelper.db();    return  
  db.query(______);  
}
```

A. Jawaban :

1. SQLite adalah database ringan yang digunakan untuk menyimpan data secara lokal di aplikasi mobile.
2. CRUD adalah akronim yang digunakan dalam pengembangan perangkat lunak untuk mendefinisikan empat operasi dasar yang dapat dilakukan terhadap data dalam suatu sistem. CRUD mencakup :
 - **Create**
Operasi untuk menambahkan data baru ke dalam sistem. Contohnya, menambahkan pengguna baru ke dalam database atau membuat postingan baru.
 - Contoh:
 1. Di SQL: `INSERT INTO users (name, email) VALUES ('Fikri', 'Fikri@example.com');`
 2. Di API: Menggunakan metode HTTP POST.
 - **Read**
Operasi untuk membaca atau mengambil data dari sistem. Biasanya digunakan untuk menampilkan data kepada pengguna.
 - Contoh:
 - a. Di SQL: `SELECT * FROM users;`
 - b. Di API: Menggunakan metode HTTP GET.
 - **Update**
Operasi untuk mengubah data yang sudah ada. Ini dilakukan ketika kita perlu memperbarui informasi tertentu.
 - Contoh:
 - a. Di SQL: `UPDATE users SET email='Fikri@example.com' WHERE id=1;`
 - b. Di API: Menggunakan metode HTTP PUT atau PATCH.
 - **Delete**
Operasi untuk menghapus data dari sistem. Hal ini dilakukan untuk menghilangkan data yang tidak lagi diperlukan.
 - Contoh:
 - a. Di SQL: `DELETE FROM users WHERE id=1;`

b. Di API: Menggunakan metode HTTP DELETE.

3. Code :

```
CREATE TABLE users (  
  id INTEGER PRIMARY KEY AUTOINCREMENT,  
  name TEXT,  
  email TEXT,  
  createdAt TIMESTAMP DEFAULT  
  CURRENT_TIMESTAMP
```

4. Langkah-langkah utama menggunakan **plugin sqflite** di Flutter:

- a. Tambahkan dependensi sqflite dan path di file pubspec.yaml.
- b. Buat file helper untuk mengatur koneksi database.
- c. Inisialisasi database menggunakan openDatabase dan buat tabel yang diperlukan.
- d. Implementasikan metode CRUD (Create, Read, Update, Delete) dalam database helper.
- e. Panggil metode tersebut dalam aplikasi untuk mengelola data.

5. Code yang sudah dilengkapi :

```
static Future<List<Map<String, dynamic>>> getUsers() async  
{  
  final db = await SQLHelper.db();  
  return db.query('users');  
}
```

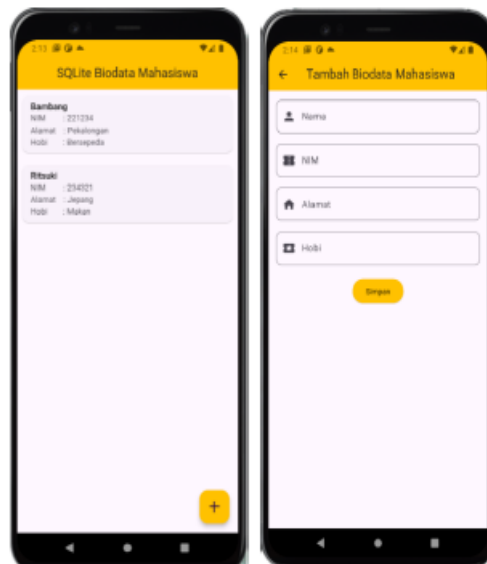
B. UNGUIDED

- Soal :

1. (Soal) Buatlah sebuah project aplikasi Flutter dengan SQLite untuk menyimpan data biodata mahasiswa yang terdiri dari nama, NIM, domisili, dan hobi. Data yang dimasukkan melalui form akan ditampilkan dalam daftar di halaman utama.

Alur Aplikasi:

- a) Form Input: Buat form input untuk menambahkan biodata mahasiswa, dengan kolom:
 - Nama
 - Nim
 - Alamat
 - Hobi
- b) Tampilkan Daftar Mahasiswa: Setelah data berhasil ditambahkan, tampilkan daftar semua data mahasiswa yang sudah disimpan di halaman utama.
- c) Implementasikan fitur Create (untuk menyimpan data mahasiswa) dan Read (untuk menampilkan daftar mahasiswa yang sudah disimpan).
- d) Contoh output:



- Jawaban :

1. Source Code :
 - a. Student_model.dart

```

class Student {
  final int? id;
  final String name;
  final String nim;
  final String address;
  final String hobby;

  Student({this.id, required this.name, required this.nim, required this.address, required
this.hobby});

  Map<String, dynamic> toMap() {
    return {
      'id': id,
      'name': name,
      'nim': nim,
      'address': address,
      'hobby': hobby,
    };
  }
}

```

b. Main.dart

```

import 'package:flutter/material.dart';
import 'add_student_screen.dart';
import 'database_helper.dart';

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'SQLite Biodata Mahasiswa',
      theme: ThemeData(
        primarySwatch: Colors.yellow,
      ),
      home: StudentListScreen(),
    );
  }
}

class StudentListScreen extends StatefulWidget {
  @override
  _StudentListScreenState createState() => _StudentListScreenState();
}

class _StudentListScreenState extends State<StudentListScreen> {
  List<Map<String, dynamic>> _students = [];

  @override
  void initState() {
    super.initState();
    _refreshStudentList();
  }

  void _refreshStudentList() async {
    final data = await DatabaseHelper.instance.getStudents();
    setState(() {
      _students = data;
    });
  }

  void _navigateToAddStudent() async {
    await Navigator.push(
      context,
      MaterialPageRoute(builder: (context) => AddStudentScreen()),
    );
    _refreshStudentList();
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('SQLite Biodata Mahasiswa'),
      ),
      body: _students.isEmpty
        ? Center(child: Text('Belum ada data mahasiswa.'))
        : ListView.builder(
            itemCount: _students.length,
            itemBuilder: (context, index) {
              final student = _students[index];
              return ListTile(
                title: Text(student['name']),
                subtitle: Text(
                  'NIM: ${student['nim']}\nAlamat: ${student['address']}\nHobi:
${student['hobby']}',
                ),
              );
            },
          ),
      floatingActionButton: FloatingActionButton(
        onPressed: _navigateToAddStudent,
        child: Icon(Icons.add),
      ),
    );
  }
}

```

c. Database_helper.dart

```
import 'dart:async';
import 'package:sqflite/sqflite.dart';
import 'package:path/path.dart';

class DatabaseHelper {
  static final DatabaseHelper instance = DatabaseHelper._init();
  static Database? _database;

  DatabaseHelper._init();

  Future<Database> get database async {
    if (_database != null) return _database!;
    _database = await _initDB('students.db');
    return _database!;
  }

  Future<Database> _initDB(String filePath) async {
    final dbPath = await getDatabasesPath();
    final path = join(dbPath, filePath);
    return await openDatabase(path, version: 1, onCreate:
    _createDB);
  }

  Future _createDB(Database db, int version) async {
    await db.execute('''
      CREATE TABLE students (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        name TEXT NOT NULL,
        nim TEXT NOT NULL,
        address TEXT NOT NULL,
        hobby TEXT NOT NULL
      )
    ''');
  }

  Future<int> insertStudent(Map<String, dynamic> student) async {
    final db = await instance.database;
    return await db.insert('students', student);
  }

  Future<List<Map<String, dynamic>>> getStudents() async {
    final db = await instance.database;
    return await db.query('students', orderBy: 'id');
  }
}
```

d. Add_student_screen.dart

```
import 'package:flutter/material.dart';
import 'database_helper.dart';

class AddStudentScreen extends StatefulWidget {
  @override
  _AddStudentScreenState createState() => _AddStudentScreenState();
}

class _AddStudentScreenState extends State<AddStudentScreen> {
  final _formKey = GlobalKey<FormState>();
  String _name = '';
  String _nim = '';
  String _address = '';
  String _hobby = '';

  void _saveStudent() async {
    if (_formKey.currentState!.validate()) {
      _formKey.currentState!.save();
      await DatabaseHelper.instance.insertStudent({
        'name': _name,
        'nim': _nim,
        'address': _address,
        'hobby': _hobby,
      });
      Navigator.pop(context);
    }
  }

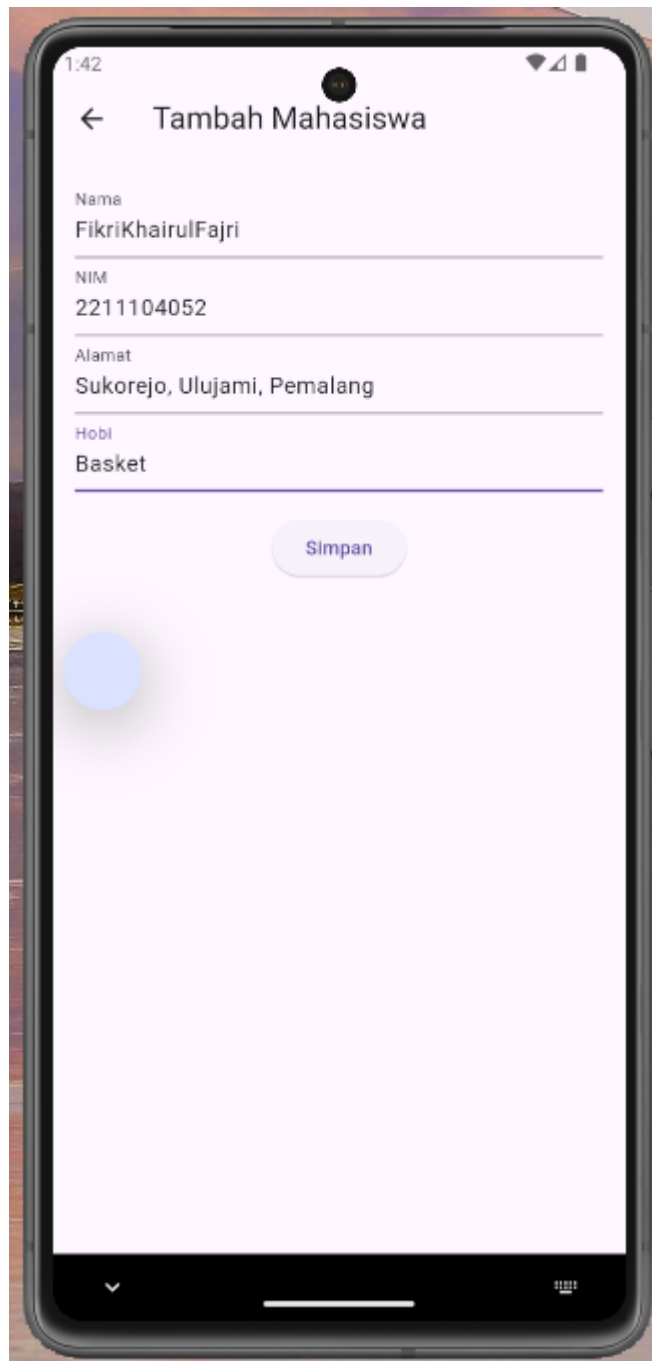
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Tambah Mahasiswa'),
      ),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Form(
          key: _formKey,
          child: Column(
            children: [
              TextFormField(
                decoration: InputDecoration(labelText: 'Nama'),
                onSave: (value) => _name = value!,
                validator: (value) =>
                  value!.isEmpty ? 'Nama tidak boleh kosong' : null,
              ),
              TextFormField(
                decoration: InputDecoration(labelText: 'NIM'),
                onSave: (value) => _nim = value!,
                validator: (value) =>
                  value!.isEmpty ? 'NIM tidak boleh kosong' : null,
              ),
              TextFormField(
                decoration: InputDecoration(labelText: 'Alamat'),
                onSave: (value) => _address = value!,
                validator: (value) =>
                  value!.isEmpty ? 'Alamat tidak boleh kosong' :
null,
              ),
              TextFormField(
                decoration: InputDecoration(labelText: 'Hobi'),
                onSave: (value) => _hobby = value!,
                validator: (value) =>
                  value!.isEmpty ? 'Hobi tidak boleh kosong' : null,
              ),
              SizedBox(height: 16),
              ElevatedButton(
                onPressed: _saveStudent,
                child: Text('Simpan'),
              ),
            ],
          ),
        ),
      ),
    );
  }
}
```


2. Output :

1. Tampilan Utama



2. Masukan Nama, Nim, Alamat, Hobi



The image shows a smartphone screen with a form titled "Tambah Mahasiswa" (Add Student). The form has four input fields: "Name" with the value "FikriKhairulFajri", "NIM" with the value "2211104052", "Alamat" (Address) with the value "Sukorejo, Ulujami, Pemalang", and "Hobi" (Hobby) with the value "Basket". A "Simpan" (Save) button is located below the form fields. The phone's status bar at the top shows the time as 1:42 and various icons. A blue circular overlay is visible on the left side of the screen.

1:42

← Tambah Mahasiswa

Name
FikriKhairulFajri

NIM
2211104052

Alamat
Sukorejo, Ulujami, Pemalang

Hobi
Basket

Simpan

3. Simpan Data



Deskripsi Program :

Program ini adalah aplikasi Flutter sederhana yang menggunakan SQLite untuk menyimpan dan menampilkan biodata mahasiswa. Aplikasi ini memiliki dua halaman utama: halaman daftar mahasiswa dan halaman untuk menambahkan data mahasiswa baru. Data yang disimpan meliputi nama, NIM, alamat, dan hobi mahasiswa. Halaman utama menampilkan daftar mahasiswa dalam format kartu (card) dengan tampilan yang rapi sesuai contoh soal. Pada halaman tambah mahasiswa, pengguna dapat mengisi form dengan validasi untuk memastikan semua kolom diisi. SQLite digunakan sebagai basis data lokal untuk menyimpan data secara permanen, sehingga data tetap ada meskipun aplikasi ditutup. Program ini dirancang agar mudah digunakan, dengan navigasi yang sederhana dan fitur "Create" serta "Read" untuk memanipulasi data.