LAPORAN PRAKTIKUM
PEMROGRAMAN PERANGKAT BERGERAK

MODUL X

DATA STORAGE BAGIAN 1

Disusun Oleh :

Zivana Afra Yulianto / 2211104039

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

BAB I
PENDAHULUAN

A. DASAR TEORI

SQLite adalah sistem database relasional ringan yang digunakan untuk penyimpanan data lokal dalam aplikasi mobile. Dalam Flutter, SQLite diakses menggunakan paket sqflite, memungkinkan operasi CRUD (Create, Read, Update, Delete) untuk mengelola data. Aplikasi dirancang dengan arsitektur MVC (Model-View-Controller), di mana model mengelola data, view menyajikan antarmuka pengguna, dan controller menangani logika bisnis serta interaksi dengan database.

B. MAKSUD DAN TUJUAN

Maksud:

Melatih kemampuan mahasiswa dalam membangun aplikasi berbasis Flutter dengan SQLite untuk pengelolaan data lokal.

Tujuan:

1. Memahami integrasi SQLite dengan Flutter.
2. Mengimplementasikan operasi CRUD.
3. Mengembangkan aplikasi dengan antarmuka yang interaktif.
4. Menerapkan pola desain modular.

BAB II
IMPLEMENTASI

A. GUIDED

Code helper/db_helper.dart :

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

class DatabaseHelper {
  static final DatabaseHelper _instance = DatabaseHelper._internal();
  static Database? _database;

  factory DatabaseHelper() {
    return _instance;
  }
  DatabaseHelper._internal();

  Future<Database> get database async {
    if (_database != null) return _database!;
    {
      _database = await _initDatabase();
      return _database!;
    }
  }

  Future<Database> _initDatabase() async {
    String path = join(await getDatabasesPath(), 'my_prakdatabase.db');

    return await openDatabase(
      path,
      version: 1,
      onCreate: _onCreate,
    );
  }

  Future<void> _onCreate(Database db, int version) async {
    await db.execute('''
    Create TABLE my_table(
    id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
    title TEXT,
    description TEXT,
    createdAt TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP)''');
  }

  Future<int> insert(Map<String, dynamic> row) async {
    Database db = await database;
    return await db.insert('my_table', row);
  }

  Future<List<Map<String, dynamic>>> queryAllRows() async {
    Database db = await database;
    return await db.query('my_table');
  }

  Future<int> update(Map<String, dynamic> row) async {
    Database db = await database;
    int id = row['id'];
    return await db.update('my_table', row, where: 'id = ?', whereArgs: [id]);
  }

  Future<int> delete(int id) async {
    Database db = await database;
    return await db.delete('my_table', where: 'id = ?', whereArgs: [id]);
  }
}
```

Code view/my_db_view :

```
import 'package:flutter/material.dart';
import 'package:praktikum/guided/helper/db_helper.dart';

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

  @override
  State<MyDatabaseView> createState() => _MyDatabaseViewState();
}

class _MyDatabaseViewState extends State<MyDatabaseView> {
  final DatabaseHelper dbHelper = DatabaseHelper();
  List<Map<String, dynamic>> dbData = [];
  final TextEditingController _titleController = TextEditingController();
  final TextEditingController _descriptionController = TextEditingController();

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

  @override
  void dispose() {
    _titleController.dispose();
    _descriptionController.dispose();
    super.dispose();
  }

  void _refreshData() async {
    final data = await dbHelper.queryAllRows();
    setState(() {
      dbData = data;
    });
  }

  void _addData() async {
    await dbHelper.insert({
      'title': _titleController.text,
      'description': _descriptionController.text,
    });
    _titleController.clear();
    _descriptionController.clear();
    _refreshData();
  }

  void _updateData(int id) async {
    await dbHelper.update({
      'id': id,
      'title': _titleController.text,
      'description': _descriptionController.text,
    });
    _titleController.clear();
    _descriptionController.clear();
    _refreshData();
  }

  void _deleteData(int id) async {
    await dbHelper.delete(id);
    _refreshData();
  }

  void _showEditDialog(Map<String, dynamic> item) {
    _titleController.text = item['title'];
    _descriptionController.text = item['description'];
```

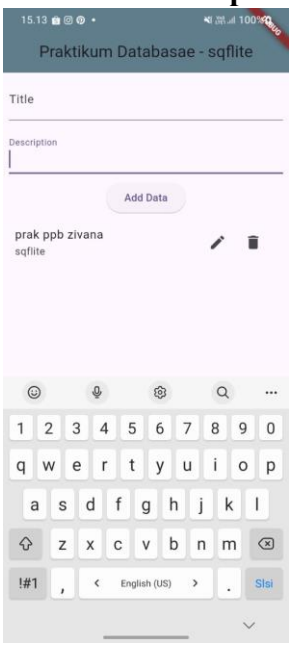
```
    showDialog(
        context: context,
        builder: (context) {
          return AlertDialog(
            title: Text('Edit Item'),
            content: Column(
              mainAxisSize: MainAxisSize.min,
              children: [
                TextField(
                  controller: _titleController,
                  decoration: InputDecoration(labelText: 'Title'),
                ),
                TextField(
                  controller: _descriptionController,
                  decoration: InputDecoration(labelText: 'Decoration'),
                )
              ],
            ),
            actions: [
              TextButton(
                onPressed: () {
                  Navigator.of(context).pop();
                },
                child: Text('Cancel'),
              ),
              TextButton(
                onPressed: () {
                  _updateData(item['id']);
                  Navigator.of(context).pop();
                },
                child: Text('Save'),
              ),
            ],
          );
        });
  }

  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(
          'Praktikum Databasae - sqflite',
        ),
        backgroundColor: Colors.blueGrey,
        centerTitle: true,
      ),
      body: Column(
        children: [
          Padding(
            padding: const EdgeInsets.all(8.0),
            child: TextField(
              controller: _titleController,
              decoration: InputDecoration(
                labelText: 'Title',
              ),
            ),
          ),
          Padding(
            padding: const EdgeInsets.all(8.0),
            child: TextField(
              controller: _descriptionController,
              decoration: InputDecoration(
                labelText: 'Description',
              ),
            ),
          ),
          ElevatedButton(
            onPressed: _addData,
            child: Text('Add Data'),
          ),
```

```
      Expanded(
              child: ListView.builder(
            itemCount: dbData.length,
            itemBuilder: (context, index) {
              final item = dbData[index];
              return ListTile(
                title: Text(item['title']),
                subtitle: Text(item['description']),
                trailing: Row(
                  mainAxisSize: MainAxisSize.min,
                  children: [
                    IconButton(
                      onPressed: () {
                        _showEditDialog(item);
                      },
                      icon: Icon(Icons.edit),
                    ),
                    IconButton(
                      onPressed: () {
                        _deleteData(item['id']);
                      },
                      icon: Icon(Icons.delete),
                    ),
                  ],
                ),
              );
            },
          ))
        ],
      ),
    );
  }
}
```

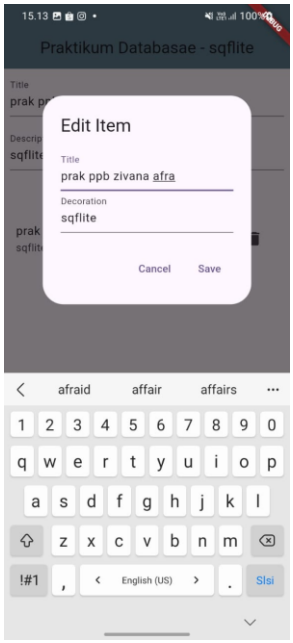
Code main.dart :

```
import 'package:flutter/material.dart';
import 'package:praktikum/guided/view/my_db_view.dart';

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

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

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
        useMaterial3: true,
      ),
      home: MyDatabaseView(),

    );
  }
}
```

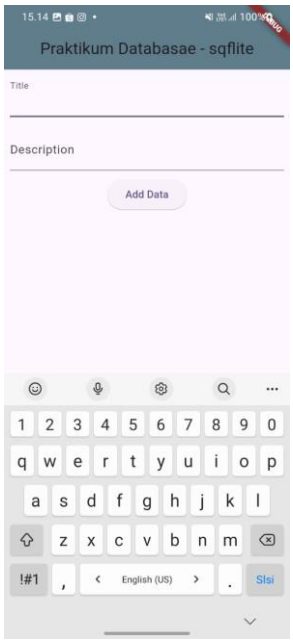
Screenshoot Output :



(output add)



(edit)



(delete)

Deskipsi :

Aplikasi ini adalah program Flutter yang menggunakan SQLite untuk menyimpan dan mengelola data lokal. Pengguna dapat menambahkan, melihat, mengedit, dan menghapus data dalam tabel yang berisi judul dan deskripsi. Operasi database dilakukan melalui kelas `DatabaseHelper`, sementara antarmuka pengguna dirancang menggunakan `ListView` untuk menampilkan data secara dinamis. Aplikasi ini cocok untuk memahami dasar CRUD dengan SQLite di Flutter.

B. UNGUIDED

Code helpers/db_helper.dart :

```dart
import 'package:sqflite/sqflite.dart';
import 'package:path/path.dart';
import '../models/student.dart';

class DBHelper {
  static Database? _database;

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

  Future<Database> _initDB() async {
    final dbPath = await getDatabasesPath();
    return openDatabase(
      join(dbPath, 'students.db'),
      onCreate: (db, version) {
        return db.execute('''
          CREATE TABLE students(
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            name TEXT,
            nim TEXT,
            address TEXT,
            hobby TEXT
          )
        ''');
      },
      version: 1,
    );
  }

  Future<void> insertStudent(Student student) async {
    final db = await database;
    await db.insert('students', student.toMap());
  }

  Future<List<Student>> fetchStudents() async {
    final db = await database;
    final maps = await db.query('students');
    return List.generate(maps.length, (i) => Student.fromMap(maps[i]));
  }

  // Tambahkan metode delete di sini
  Future<void> deleteStudent(int id) async {
    final db = await database;
    await db.delete(
      'students',
      where: 'id = ?',
      whereArgs: [id],
    );
  }
}
```

Code models/student.dart :

```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,
    };
  }

  factory Student.fromMap(Map<String, dynamic> map) {
    return Student(
      id: map['id'],
      name: map['name'],
      nim: map['nim'],
      address: map['address'],
      hobby: map['hobby'],
    );
  }
}
```

Code screen/add_student_screen.dart :

```dart
import 'package:flutter/material.dart';
import '../helpers/db_helper.dart';
import '../models/student.dart';

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

class _AddStudentScreenState extends State<AddStudentScreen> {
  final _formKey = GlobalKey<FormState>();
  final _nameController = TextEditingController();
  final _nimController = TextEditingController();
  final _addressController = TextEditingController();
  final _hobbyController = TextEditingController();

  Future<void> _submitData() async {
    if (!_formKey.currentState!.validate()) return;

    final student = Student(
      name: _nameController.text,
      nim: _nimController.text,
      address: _addressController.text,
      hobby: _hobbyController.text,
    );

    final dbHelper = DBHelper();
    await dbHelper.insertStudent(student);

    // Show SnackBar for feedback
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(
        content: Text('Data Mahasiswa Berhasil Disimpan!'),
        backgroundColor: Colors.green,
        duration: Duration(seconds: 2),
      ),
    );
```

```
    Future.delayed(Duration(seconds: 2), () {
      Navigator.pop(context, true);
    });
  }
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(
          'Tambah Mahasiswa',
          style: TextStyle(color: Colors.white, fontWeight: FontWeight.bold),
        ),
        backgroundColor: Color.fromARGB(255, 8, 84, 29),
        elevation: 0,
      ),
      body: Container(
        decoration: BoxDecoration(
          gradient: LinearGradient(
            colors: [
              const Color.fromARGB(255, 230, 250, 240),
              const Color.fromARGB(255, 200, 235, 210)
            ],
            begin: Alignment.topCenter,
            end: Alignment.bottomCenter,
          ),
        ),
        child: Center(
          child: SingleChildScrollView(
            child: Padding(
              padding: const EdgeInsets.symmetric(horizontal: 20.0),
              child: Card(
                shape: RoundedRectangleBorder(
                  borderRadius: BorderRadius.circular(15),
                ),
                elevation: 8,
                child: Padding(
                  padding: const EdgeInsets.all(20.0),
                  child: Form(
                    key: _formKey,
                    child: Column(
                      crossAxisAlignment: CrossAxisAlignment.stretch,
                      children: [
                        _buildTextField(_nameController, 'Nama', Icons.person),
                        _buildTextField(
                            _nimController, 'NIM', Icons.confirmation_number),
                        _buildTextField(
                            _addressController, 'Alamat', Icons.location_on),
                        _buildTextField(
                            _hobbyController, 'Hobi', Icons.art_track),
                        SizedBox(height: 30),
                        ElevatedButton(
                          onPressed: _submitData,
                          style: ElevatedButton.styleFrom(
                            backgroundColor: Color.fromARGB(255, 8, 84, 29),
                            shape: RoundedRectangleBorder(
                              borderRadius: BorderRadius.circular(20),
                            ),
                            padding: EdgeInsets.symmetric(
                                horizontal: 50, vertical: 15),
                            shadowColor: Colors.greenAccent,
                            elevation: 5,
                          ),
                          child: Text(
                            'Simpan',
                            style: TextStyle(
                              fontSize: 18,
                              fontWeight: FontWeight.bold,
                              color: Colors.white,
                            ),
                          ),
                        ),
                      ],
                    ),
                  ),
                ),
              ),
            ),
          ),
        ),
      ),
    );
  }
```

```
  Widget _buildTextField(
      TextEditingController controller, String label, IconData icon) {
    return Padding(
      padding: const EdgeInsets.symmetric(vertical: 12.0),
      child: TextFormField(
        controller: controller,
        decoration: InputDecoration(
          prefixIcon: Icon(icon, color: Color.fromARGB(255, 8, 84, 29)),
          labelText: label,
          labelStyle: TextStyle(color: Colors.black54),
          border: OutlineInputBorder(
            borderRadius: BorderRadius.circular(15),
          ),
          filled: true,
          fillColor: Colors.white,
          contentPadding: EdgeInsets.symmetric(vertical: 18, horizontal: 20),
          enabledBorder: OutlineInputBorder(
            borderSide: BorderSide(color: Colors.greenAccent, width: 1.5),
            borderRadius: BorderRadius.circular(15),
          ),
          focusedBorder: OutlineInputBorder(
            borderSide:
                BorderSide(color: Color.fromARGB(255, 8, 84, 29), width: 2),
            borderRadius: BorderRadius.circular(15),
          ),
        ),
        validator: (value) =>
            value!.isEmpty ? '$label tidak boleh kosong' : null,
        style: TextStyle(fontSize: 16),
      ),
    );
  }
}
```

Code screens/home_screen.dart :

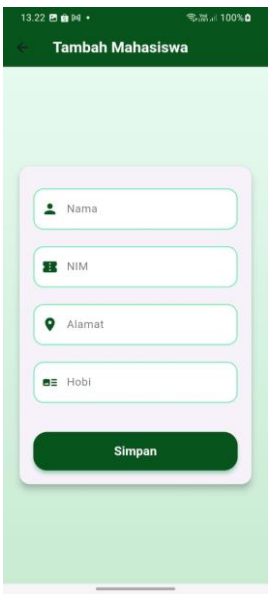
```
import 'package:flutter/material.dart';
import '../helpers/db_helper.dart';
import '../models/student.dart';
import 'add_student_screen.dart';

class HomeScreen extends StatefulWidget {
  @override
  _HomeScreenState createState() => _HomeScreenState();
}

class _HomeScreenState extends State<HomeScreen> {
  late DBHelper dbHelper;
  List<Student> students = [];

  @override
  void initState() {
    super.initState();
    dbHelper = DBHelper();
    _loadStudents();
  }

  Future<void> _loadStudents() async {
    final data = await dbHelper.fetchStudents();
    setState(() {
      students = data;
    });
  }

  Future<void> _deleteStudent(int id) async {
    // Menampilkan dialog konfirmasi sebelum menghapus
    final shouldDelete = await showDialog(
      context: context,
      builder: (ctx) => AlertDialog(
        title: Text('Hapus Mahasiswa'),
        content: Text('Apakah Anda yakin ingin menghapus mahasiswa ini?'),
        actions: [
```

```
          TextButton(
            onPressed: () => Navigator.of(ctx).pop(false),
            child: Text('Batal'),
          ),
          TextButton(
            onPressed: () => Navigator.of(ctx).pop(true),
            child: Text(
              'Hapus',
              style: TextStyle(color: Colors.red),
            ),
          ),
        ],
      ),
    );

    if (shouldDelete == true) {
      await dbHelper.deleteStudent(id);
      _loadStudents();

      // Menampilkan SnackBar untuk memberikan umpan balik
      ScaffoldMessenger.of(context).showSnackBar(
        SnackBar(
          content: Text('Mahasiswa berhasil dihapus!'),
          backgroundColor: Colors.redAccent,
          duration: Duration(seconds: 2),
        ),
      );
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(
          'Daftar Mahasiswa',
          style: TextStyle(
            fontWeight: FontWeight.bold,
            fontSize: 22,
            color: Colors.white,
          ),
        ),
        backgroundColor: const Color.fromARGB(255, 8, 84, 29),
        elevation: 8,
        centerTitle: true,
      ),
      body: Container(
        decoration: BoxDecoration(
          gradient: LinearGradient(
            colors: [
              const Color.fromARGB(255, 194, 233, 210),
              Colors.blue.shade100
            ],
            begin: Alignment.topCenter,
            end: Alignment.bottomCenter,
          ),
        ),
        child: students.isEmpty
            ? Center(
                child: Text(
                  'Belum ada data mahasiswa.',
                  style: TextStyle(
                    fontSize: 18,
                    fontWeight: FontWeight.bold,
                    color: Colors.blueGrey,
                  ),
                ),
              )
            : ListView.builder(
                itemCount: students.length,
                itemBuilder: (ctx, index) {
                  final student = students[index];
                  return AnimatedContainer(
                    duration: Duration(milliseconds: 300),
                    curve: Curves.easeInOut,
                    child: Card(
                      margin: const EdgeInsets.symmetric(
                          horizontal: 16, vertical: 10),
                      shape: RoundedRectangleBorder(
                        borderRadius: BorderRadius.circular(20),
                      ),
```

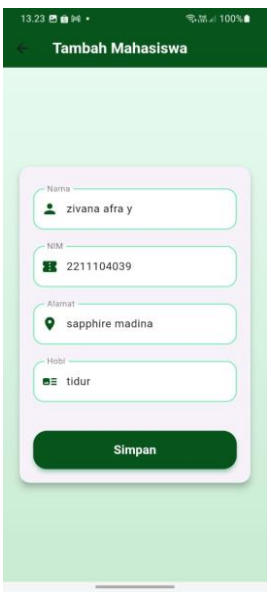
```
                      elevation: 5,
                      shadowColor: Colors.grey.withOpacity(0.5),
                      child: ListTile(
                        leading: CircleAvatar(
                          backgroundColor:
                              const Color.fromARGB(255, 100, 181, 246),
                          child: Icon(Icons.person, color: Colors.white),
                        ),
                        title: Text(
                          student.name,
                          style: TextStyle(
                            fontWeight: FontWeight.bold,
                            fontSize: 18,
                          ),
                        ),
                        subtitle: Padding(
                          padding: const EdgeInsets.only(top: 5.0),
                          child: Text(
                            'NIM: ${student.nim}\nAlamat: ${student.address}\nHobi:
${student.hobby}',
                            style: TextStyle(
                              color: Colors.blueGrey.shade700,
                              fontSize: 14,
                            ),
                          ),
                        ),
                        trailing: IconButton(
                          icon: Icon(Icons.delete_outline, color: Colors.red),
                          onPressed: () => _deleteStudent(student.id!),
                        ),
                      ),
                    ),
                  );
                },
              ),
      ),
      floatingActionButton: FloatingActionButton(
        backgroundColor: const Color.fromARGB(255, 6, 81, 13),
        onPressed: () async {
          final result = await Navigator.push(
            context,
            MaterialPageRoute(builder: (ctx) => AddStudentScreen()),
          );
          if (result == true) _loadStudents();

          // Menampilkan SnackBar setelah berhasil menambahkan data
          ScaffoldMessenger.of(context).showSnackBar(
            SnackBar(
              content: Text('Mahasiswa berhasil ditambahkan!'),
              backgroundColor: Colors.green,
              duration: Duration(seconds: 2),
            ),
          );
        },
        child: Icon(Icons.person_add, color: Colors.white, size: 30),
        elevation: 8,
        shape: RoundedRectangleBorder(
          borderRadius: BorderRadius.circular(15),
        ),
      ),
    );
  }
}
```

Screenshoot Output

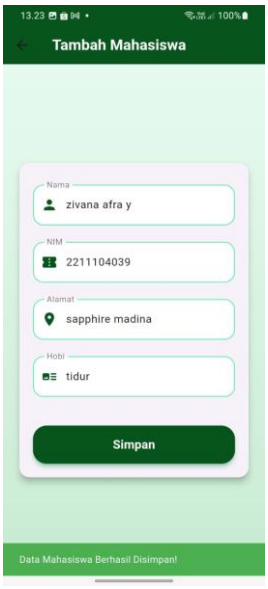
(awal)



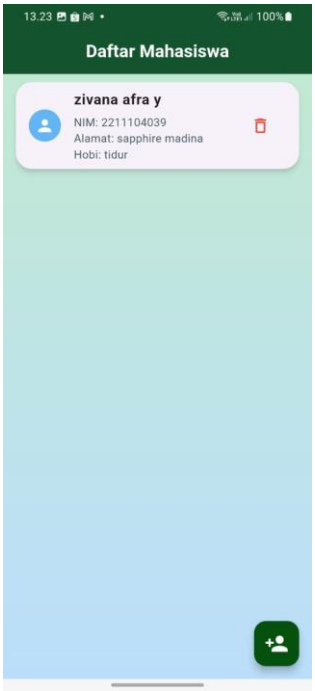
(klik logo add)



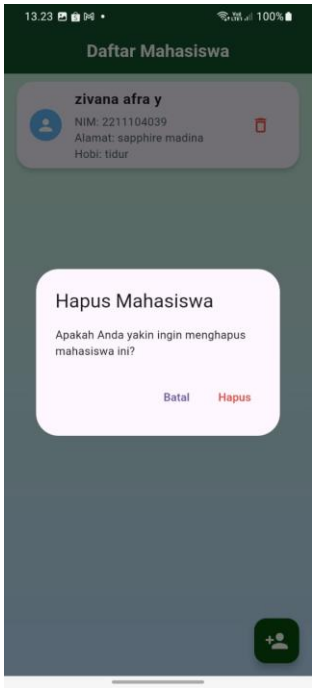
(isi data)



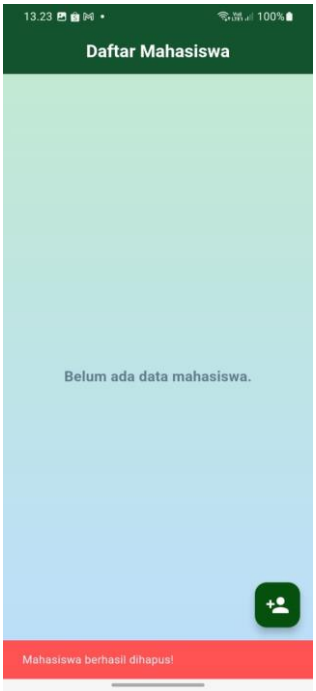
(klik simpan)



(tampil data)



(kik ikon delete)



(data terhapus)

Deskripsi Program

Program ini adalah aplikasi Flutter berbasis SQLite untuk mengelola data mahasiswa. Pengguna dapat menambahkan, melihat, dan menghapus data mahasiswa yang berisi nama, NIM, alamat, dan hobi. Aplikasi terdiri dari tiga komponen utama:

1. Database Helper (DBHelper): Mengatur interaksi dengan database SQLite, termasuk pembuatan tabel, penambahan data, pengambilan data, dan penghapusan data.
2. Model (Student): Merepresentasikan data mahasiswa dengan atribut dan fungsi konversi antara objek dan format database.
3. Antarmuka Pengguna:
    - HomeScreen: Menampilkan daftar mahasiswa dalam bentuk kartu interaktif dengan opsi untuk menghapus data.
    - AddStudentScreen: Memungkinkan pengguna menambahkan data mahasiswa melalui formulir yang terintegrasi.

Aplikasi ini menggunakan SnackBar untuk memberikan umpan balik kepada pengguna setelah melakukan operasi.

BAB III
KESIMPULAN

KESIMPULAN

SQLite adalah solusi efisien untuk penyimpanan data lokal dalam aplikasi mobile. Integrasi dengan Flutter melalui sqflite mempermudah implementasi CRUD. Praktikum ini membantu mahasiswa memahami pengelolaan data lokal dan membangun aplikasi modular serta user-friendly.