LAPORAN PRAKTIKUM
PEMROGRAMAN PERANGKAT BERGERAK

MODUL X
DATA STORAGE

Disusun Oleh :

Arzario Irsyad Al Fatih/2211104032

SE 06 2

Asisten Praktikum :

Muhammad Faza Zulian Gesit Al Barru

Aisyah Hasna Aulia

Dosen Pengampu :

Yudha Islami Sulistya

PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAKFAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO2024

1. **GUIDED**

   a. **Data Storage**

   Source Code

   - main.dart

```dart
import 'package:flutter/material.dart';
import 'package:praktikum_10/view/my_db_view.dart';

void main() {
  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: 'PPB',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor:
Colors.deepPurple),
        useMaterial3: true,
      ),
      // ignore: prefer_const_constructors
      home: MyDatabaseView (title: 'Praktikum 10'),
    );
  }
}
```

   - my_db_view.dart

```dart
import 'package:flutter/material.dart';
import 'package:praktikum_10/helper/db_helper.dart';

class MyDatabaseView extends StatefulWidget {
  const MyDatabaseView({super.key, required String title});

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

class _MyDatabaseViewState extends State<MyDatabaseView> {
```

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

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

```
  _refreshData();
  _titleController.clear();
  _descriptionController.clear();
}

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: 'Description',
              ),
            ),
          ],
        ),
        actions: [
          TextButton(
            child: Text('Cancel'),
            onPressed: () {
              Navigator.of(context).pop();
            },
          ),
```

```
            TextButton(
              child: Text('Save'),
              onPressed: () {
                _updateData(item['id']);
                Navigator.of(context).pop();
              },
            ),
          ],
        );
      },
    );
  }

Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Praktikum Database - sqflite'),
      ),
      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: _insertData,
            child: Text('Add Data'),
          ),
          // Use Expanded to take available space for ListView
```

```
          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(
                        icon: Icon(Icons.edit),
                        onPressed: () {
                          _showEditDialog(item);
                        },
                      ),
                      IconButton(
                        icon: Icon(Icons.delete),
                        onPressed: () {
                          _deleteData(item['id']);
                        },
                      ),
                    ],
                  ),
                );
              },
            ),
          ),
        ],
      ),
    );
  }
```

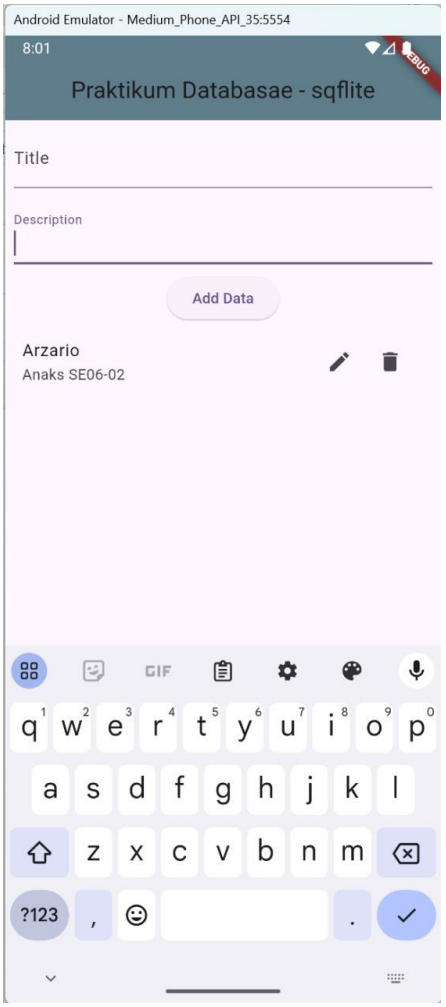
- db_helper.dart

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

class DatabaseHelper {
  // Singleton instance
  static final DatabaseHelper _instance =
DatabaseHelper._internal();
  static Database? _database;
```

```dart
  // Constructor factory untuk mengembalikan instance
  factory DatabaseHelper() {
    return _instance;
  }

  // Constructor privat untuk singleton
  DatabaseHelper._internal();

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

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

    // Membuka atau membuat database
    return await openDatabase(
      path,
      version: 1,
      onCreate: _onCreate,
    );
  }

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

  // Menambahkan data baru
  Future<int> insert(Map<String, dynamic> row) async {
```

```
    Database db = await database;
    return await db.insert('my_table', row);
  }

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

  // Membaca data berdasarkan ID
  Future<Map<String, dynamic>?> getItem(int id) async {
    Database db = await database;
    final List<Map<String, dynamic>> result = await db.query(
      'my_table',
      where: 'id = ?',
      whereArgs: [id],
      limit: 1,
    );
    return result.isNotEmpty ? result.first : null;
  }

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

  update(Map<String, Object> map) {}
}
```

Output



Deskripsi

Program Flutter ini adalah sebuah aplikasi yang menggunakan SQLite untuk menyimpan dan mengelola data secara lokal pada perangkat. `main.dart` adalah titik masuk aplikasi yang menampilkan halaman utama melalui widget `MyDatabaseView`, yang didefinisikan dalam `my_db_view.dart`. Halaman ini memungkinkan pengguna untuk memasukkan, mengedit, dan menghapus data berupa title dan description. Fungsi-fungsi CRUD (Create, Read, Update, Delete) diimplementasikan menggunakan helper yang ada di file `db_helper.dart`, di mana SQLite digunakan untuk membuat tabel database (`my_table`) dengan kolom `id`, `title`, `description`, dan `createdAt`. Aplikasi ini memanfaatkan TextField untuk memasukkan data dan ListView untuk menampilkan daftar data yang sudah tersimpan, dengan tombol untuk mengedit atau menghapus data. Desainnya menggunakan pendekatan Material Design dengan state management berbasis `setState`.

2. UNGUIDED

a. Soal 1

Source Code

- main.dart

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

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

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Biodata Mahasiswa',
      theme: ThemeData(primarySwatch: Colors.blue),
      home: const HomePage(),
    );
  }
}

class HomePage extends StatefulWidget {
  const HomePage({Key? key}) : super(key: key);

  @override
  _HomePageState createState() => _HomePageState();
}

class _HomePageState extends State<HomePage> {
  final dbHelper = DatabaseHelper.instance;
  List<Map<String, dynamic>> biodataList = [];

  final _namaController = TextEditingController();
  final _nimController = TextEditingController();
  final _domisiliController = TextEditingController();
  final _hobiController = TextEditingController();

  void _refreshData() async {
    final data = await dbHelper.queryAll();
```

```
    setState(() {
      biodataList = data;
    });
  }

  void _insertData() async {
    if (_namaController.text.isNotEmpty &&
        _nimController.text.isNotEmpty &&
        _domisiliController.text.isNotEmpty &&
        _hobiController.text.isNotEmpty) {
      await dbHelper.insert({
        'nama': _namaController.text,
        'nim': _nimController.text,
        'domisili': _domisiliController.text,
        'hobi': _hobiController.text,
      });
      _namaController.clear();
      _nimController.clear();
      _domisiliController.clear();
      _hobiController.clear();
      _refreshData();
    }
  }

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: const Text('Biodata Mahasiswa')),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          children: [
            TextField(controller: _namaController, decoration:
const InputDecoration(labelText: 'Nama')),
            TextField(controller: _nimController, decoration:
const InputDecoration(labelText: 'NIM')),
```

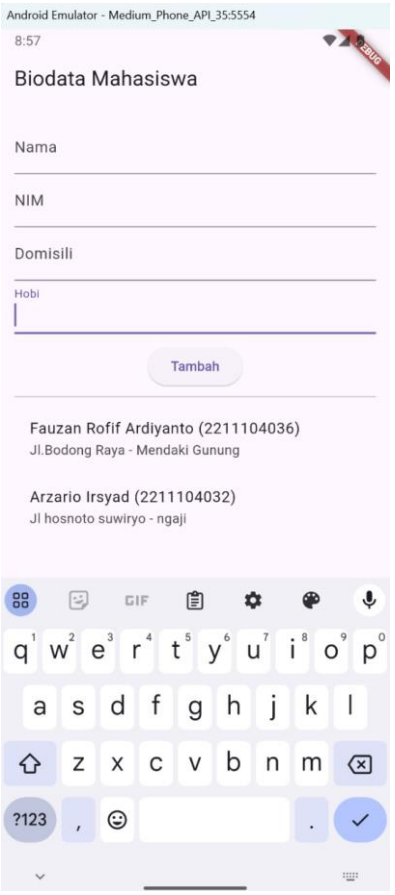
```
            TextField(controller: _domisiliController,
decoration: const InputDecoration(labelText: 'Domisili')),
            TextField(controller: _hobiController, decoration:
const InputDecoration(labelText: 'Hobi')),
            const SizedBox(height: 10),
            ElevatedButton(onPressed: _insertData, child: const
Text('Tambah')),
            const Divider(),
            Expanded(
              child: ListView.builder(
                itemCount: biodataList.length,
                itemBuilder: (context, index) {
                  final biodata = biodataList[index];
                  return ListTile(
                    title: Text('${biodata['nama']}
(${biodata['nim']})'),
                    subtitle: Text('${biodata['domisili']} -
${biodata['hobi']}'),
                  );
                },
              ),
            ),
          ],
        ),
      ),
    );
  }
}
```

- database_helper.dart

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

class DatabaseHelper {
  static final DatabaseHelper instance =
DatabaseHelper._internal();

  static Database? _database;

  DatabaseHelper._internal();

  factory DatabaseHelper() => instance;

  Future<Database> get database async {
```

```
    if (_database != null) return _database!;
    _database = await _initDatabase();
    return _database!;
  }

  Future<Database> _initDatabase() async {
    String path = join(await getDatabasesPath(), 'biodata.db');
    return await openDatabase(
      path,
      version: 1,
      onCreate: _onCreate,
    );
  }

  Future<void> _onCreate(Database db, int version) async {
    await db.execute('''
      CREATE TABLE biodata(
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        nama TEXT,
        nim TEXT,
        domisili TEXT,
        hobi TEXT
      )
    ''');
  }

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

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

Output



Deskripsi

Kode ini adalah aplikasi Flutter sederhana untuk menyimpan dan menampilkan data biodata mahasiswa menggunakan SQLite sebagai basis datanya. File `main.dart` mendefinisikan struktur utama aplikasi, termasuk halaman utama `HomePage` yang memungkinkan pengguna untuk memasukkan data berupa nama, NIM, domisili, dan hobi melalui form input berbasis TextField. Setelah data dimasukkan, pengguna dapat menyimpannya ke database SQLite menggunakan fungsi `insertData`, dan data tersebut akan ditampilkan dalam bentuk daftar di halaman utama menggunakan widget ListView. File `database_helper.dart` bertugas mengelola operasi database, seperti membuat tabel `biodata` dengan kolom `id`, `nama`, `nim`, `domisili`, dan `hobi`, serta menyediakan fungsi CRUD (Create dan Read). Dengan pendekatan ini, aplikasi mampu menyimpan data secara lokal, memperbarui daftar secara real-time, dan memberikan pengalaman pengguna yang dinamis.