

**LAPORAN PRAKTIKUM  
PEMROGRAMAN PERANGKAT BERGERAK**

**MODUL X  
DATA STORAGE (BAGIAN 1)**



**Disusun Oleh :**

**Muhammad Abdul Aziz / 2211104026**

**SE0601**

**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**

## GUIDED

### File helper/db\_helper.dart :

```
1 import 'package:sqflite/sqflite.dart';
2 import 'package:path/path.dart';
3
4 class DatabaseHelper {
5   static final DatabaseHelper _instance = DatabaseHelper._internal();
6   static Database? _database;
7
8   // factory constructor untuk mengembalikan instance singleton
9   factory DatabaseHelper() {
10     return _instance;
11   }
12
13   // Private constructor
14   DatabaseHelper._internal();
15
16   // Getter untuk database
17   Future<Database> get database async {
18     if (_database != null) return _database!;
19     {
20       _database = await _initDatabase();
21       return _database!;
22     }
23   }
24
25   // inisiasi database
26   Future<Database> _initDatabase() async {
27     // mendapatkan path untuk database
28     String path = join(await getDatabasesPath(), 'my_prakdatabase.db');
29     // membuka database
30     return await openDatabase(
31       path,
32       version: 1,
33       onCreate: _onCreate,
34     );
35   }
36
37   //membuat tabel saat db pertama kali dibuat
38   Future<void> _onCreate(Database db, int version) async {
39     await db.execute('''
40     CREATE TABLE my_table(
41     id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
42     title TEXT,
43     description TEXT,
44     createdAt TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP)
45     ''');
46   }
47
48   // metode memasukan data ke dalam tabel
49   Future<int> insert(Map<String, dynamic> row) async {
50     Database db = await database;
51     int result = await db.insert('my_table', row);
52     print('Inserted row: $result'); // Log untuk memeriksa hasil insert
53     return result;
54   }
55
56   // metode mengambil semua data dari tabel
57   Future<List<Map<String, dynamic>>> queryAllRows() async {
58     Database db = await database;
59     List<Map<String, dynamic>> result = await db.query('my_table');
60     print('Fetched data: $result'); // Log untuk memeriksa data yang diambil
61     return result;
62   }
63
64   // metode untuk memperbarui data dalam tabel
65   Future<int> update(Map<String, dynamic> row) async {
66     Database db = await database;
67     int id = row['id'];
68     return await db.update('my_table', row, where: 'id = ?', whereArgs: [id]);
69   }
70
71   // metode menghapus data dari tabel
72   Future<int> delete(int id) async {
73     Database db = await database;
74     return await db.delete('my_table', where: 'id = ?', whereArgs: [id]);
75   }
76 }
```

## File view/my\_db\_view.dart :

```
1 import 'package:flutter/material.dart';
2 import 'package:guided/helper/db_helper.dart'; // Pastikan path-nya benar
3
4 class MyDatabaseView extends StatefulWidget {
5   const MyDatabaseView({super.key});
6
7   @override
8   State<MyDatabaseView> createState() => _MyDatabaseViewState();
9 }
10
11 class _MyDatabaseViewState extends State<MyDatabaseView> {
12   final DatabaseHelper dbHelper = DatabaseHelper();
13   List<Map<String, dynamic>> _dbData = [];
14   final TextEditingController _titleController = TextEditingController();
15   final TextEditingController _descriptionController = TextEditingController();
16
17   @override
18   void initState() {
19     super.initState();
20     _refreshData();
21   }
22
23   @override
24   void dispose() {
25     _titleController.dispose();
26     _descriptionController.dispose();
27     super.dispose();
28   }
29
30   // Ubah menjadi Future<void> dan beri async
31   Future<void> _refreshData() async {
32     final data = await dbHelper.queryAllRows();
33     print(data); // Tambahkan log ini
34     setState(() {
35       _dbData = data;
36     });
37   }
38
39   // Ubah menjadi Future<void> dan beri async
40   Future<void> _addData() async {
41     await dbHelper.insert({
42       'title': _titleController.text,
43       'description': _descriptionController.text,
44     });
45     _titleController.clear();
46     _descriptionController.clear();
47     await _refreshData(); // Pastikan menunggu refresh data setelah insert
48   }
49
50   // Ubah menjadi Future<void> dan beri async
51   Future<void> _updateData(int id) async {
52     await dbHelper.update({
53       'id': id,
54       'title': _titleController.text,
55       'description': _descriptionController.text,
56     });
57     _titleController.clear();
58     _descriptionController.clear();
59     await _refreshData(); // Pastikan menunggu refresh data setelah update
60   }
61
62   // Ubah menjadi Future<void> dan beri async
63   Future<void> _deleteData(int id) async {
64     await dbHelper.delete(id);
65     await _refreshData(); // Pastikan menunggu refresh data setelah delete
66   }
67
68   // Menampilkan dialog untuk mengedit data
69   void _showEditDialog(Map<String, dynamic> item) {
70     _titleController.text = item['title'];
71     _descriptionController.text = item['description'];
72
73     showDialog(
74       context: context,
75       builder: (context) {
76         return AlertDialog(
77           title: const Text('Edit Item'),
78           content: Column(
79             mainAxisAlignment: MainAxisAlignment.min,
80             children: [
81               TextField(
82                 controller: _titleController,
83                 decoration: InputDecoration(labelText: 'Title'),
84               ),
85               TextField(
86                 controller: _descriptionController,
87                 decoration: InputDecoration(labelText: 'Description'),
88               ),
89             ],
90           ),
91           actions: [
92             TextButton(
93               onPressed: () {
94                 Navigator.of(context).pop();
95               },
96               child: Text('Cancel'),
97             ),
98             TextButton(
99               onPressed: () {
100                 _updateData(item['id']);
101                 Navigator.of(context).pop();
102               },
103               child: const Text('Save'),
104             ),
105           ],
106         );
107       },
108     );
109   }
```

```

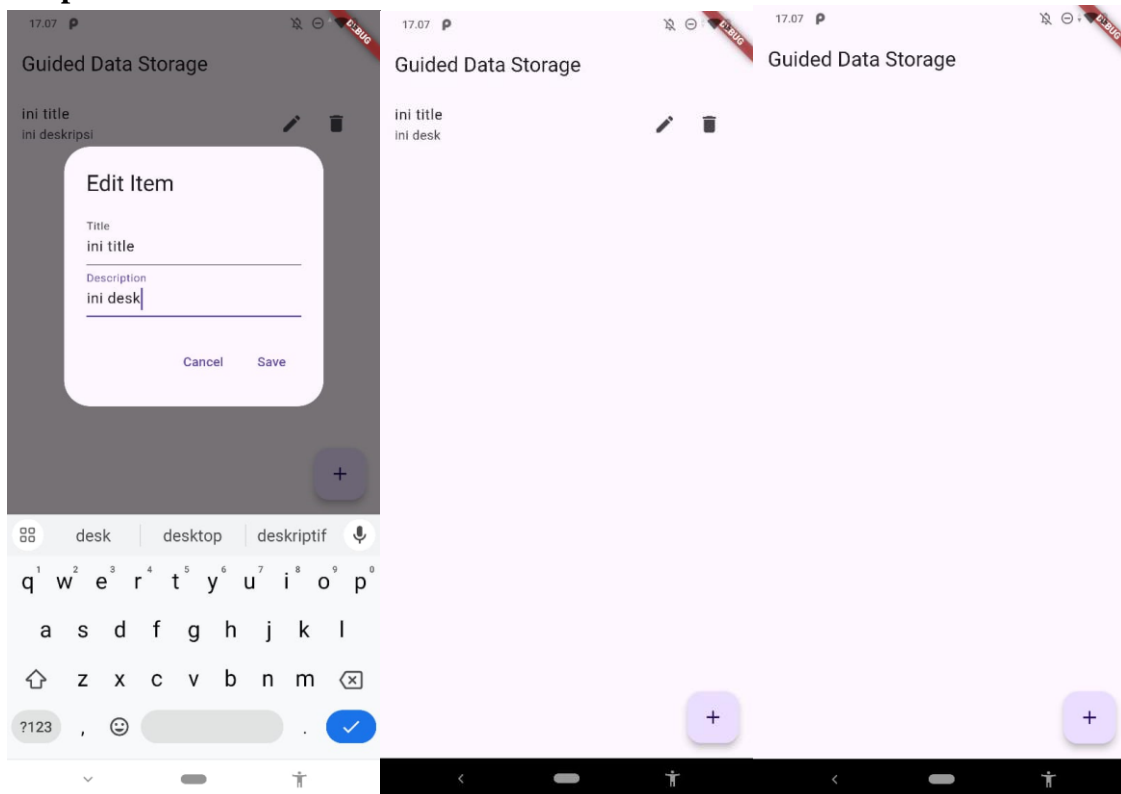
1
2 // Menampilkan dialog untuk menambahkan data
3 void _showAddDialog() {
4   _titleController.clear();
5   _descriptionController.clear();
6
7   showDialog(
8     context: context,
9     builder: (context) {
10      return AlertDialog(
11        title: Text('Add New Item'),
12        content: Column(
13          mainAxisAlignment: MainAxisAlignment.min,
14          children: [
15            TextField(
16              controller: _titleController,
17              decoration: InputDecoration(labelText: 'Title'),
18            ),
19            TextField(
20              controller: _descriptionController,
21              decoration: InputDecoration(labelText: 'Description'),
22            ),
23          ],
24        ),
25        actions: [
26          TextButton(
27            onPressed: () {
28              Navigator.of(context).pop();
29            },
30            child: Text('Cancel'),
31          ),
32          TextButton(
33            onPressed: () {
34              _addData();
35              Navigator.of(context).pop();
36            },
37            child: const Text('Add'),
38          ),
39        ],
40      );
41    },
42  );
43 }
44
45 @override
46 Widget build(BuildContext context) {
47   return Scaffold(
48     appBar: AppBar(
49       title: Text('Guided Data Storage'),
50     ),
51     body: ListView.builder(
52       itemCount: _dbData.length,
53       itemBuilder: (context, index) {
54         final item = _dbData[index];
55         print('Displaying item: ${item['title']}');
56         return ListTile(
57           title: Text(item['title'] ?? 'No Title'),
58           subtitle: Text(item['description'] ?? 'No Description'),
59           trailing: Row(
60             mainAxisAlignment: MainAxisAlignment.min,
61             children: [
62               IconButton(
63                 icon: Icon(Icons.edit),
64                 onPressed: () => _showEditDialog(item),
65               ),
66               IconButton(
67                 icon: Icon(Icons.delete),
68                 onPressed: () => _deleteData(item['id']),
69               ),
70             ],
71           ),
72         );
73       },
74     ),
75     floatingActionButton: FloatingActionButton(
76       onPressed: _showAddDialog,
77       child: Icon(Icons.add),
78     ),
79   );
80 }
81 }
82

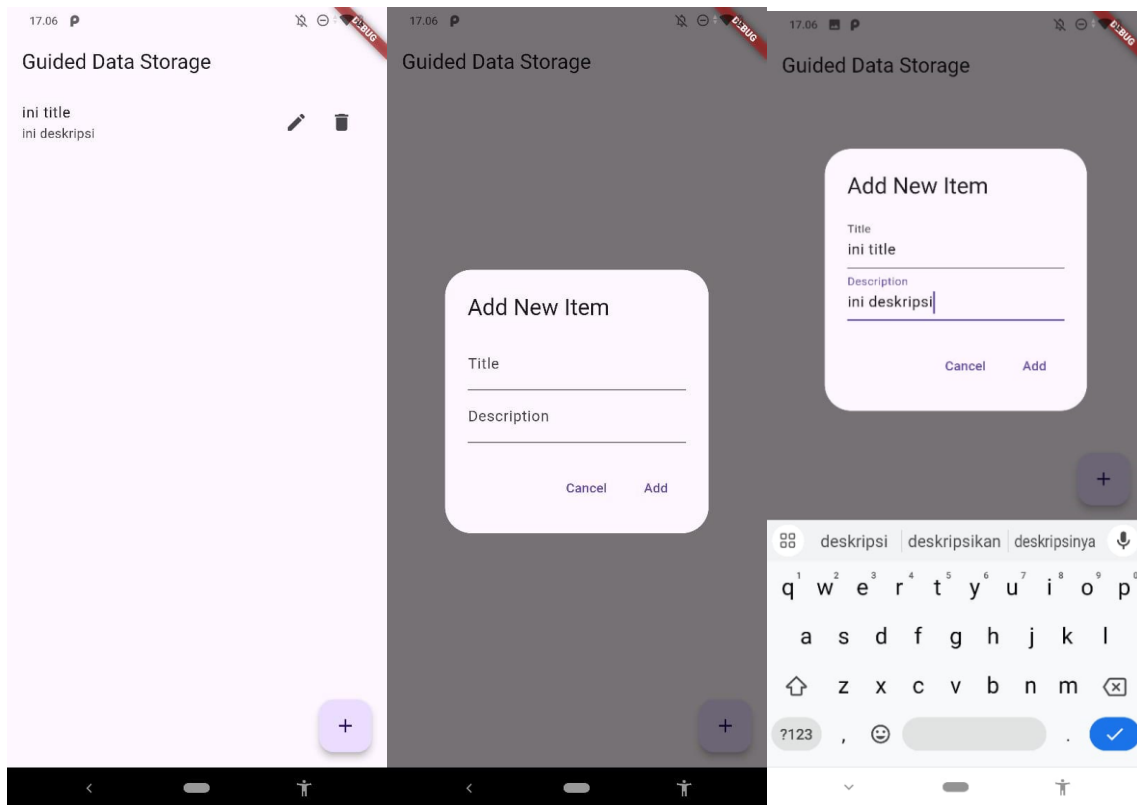
```

## File main.dart :

```
1 import 'package:flutter/material.dart';
2 import 'package:guided/view/my_db_view.dart';
3
4 void main() {
5   runApp(MyApp());
6 }
7
8 class MyApp extends StatelessWidget {
9   @override
10  Widget build(BuildContext context) {
11    return MaterialApp(
12      title: 'Guided Data Storage',
13      theme: ThemeData(
14        primarySwatch: Colors.blue,
15      ),
16      home: MyDatabaseView(),
17    );
18  }
19 }
20
```

## Output :





### Penjelasan :

Kode ini adalah aplikasi Flutter dengan integrasi SQLite untuk pengelolaan data menggunakan paket ``sqlite``. ``DatabaseHelper`` adalah kelas singleton untuk mengelola database SQLite, menyediakan fungsi CRUD (Create, Read, Update, Delete) pada tabel ``my_table``. ``MyDatabaseView`` adalah widget utama dengan antarmuka pengguna untuk menampilkan data dari database dalam bentuk daftar, serta menyediakan fitur untuk menambah, mengedit, dan menghapus data melalui dialog input. Komponen utama aplikasi meliputi pengaturan database, logika pengelolaan data, serta antarmuka berbasis Flutter. Aplikasi ini diinisialisasi melalui ``main()`` yang memuat ``MyDatabaseView`` sebagai halaman utama.

## UNGUIDED

### File helper/db\_helper.dart :

```
1 import 'package:path/path.dart';
2 import 'package:sqflite/sqflite.dart';
3
4 class DatabaseHelper {
5   static final DatabaseHelper _instance = DatabaseHelper._internal();
6   static Database? _database;
7
8   factory DatabaseHelper() => _instance;
9
10  DatabaseHelper._internal();
11
12  Future<Database> get database async {
13    if (_database != null) return _database!;
14    _database = await _initDatabase();
15    return _database!;
16  }
17
18  Future<Database> _initDatabase() async {
19    String path = join(await getDatabasesPath(), 'my_database.db');
20    return await openDatabase(
21      path,
22      version: 1,
23      onCreate: _onCreate,
24    );
25  }
26
27  Future<void> _onCreate(Database db, int version) async {
28    await db.execute('''
29      CREATE TABLE mahasiswa(
30        id INTEGER PRIMARY KEY AUTOINCREMENT,
31        nama TEXT,
32        nim TEXT,
33        alamat TEXT,
34        hobi TEXT
35      )
36    ''');
37  }
38
39  Future<int> insert(Map<String, dynamic> row) async {
40    Database db = await database;
41    return await db.insert('mahasiswa', row);
42  }
43
44  Future<List<Map<String, dynamic>>> queryAllRows() async {
45    Database db = await database;
46    return await db.query('mahasiswa');
47  }
48
49  Future<int> update(Map<String, dynamic> row) async {
50    Database db = await database;
51    return await db.update(
52      'mahasiswa',
53      row,
54      where: 'id = ?',
55      whereArgs: [row['id']],
56    );
57  }
58
59  Future<int> delete(int id) async {
60    Database db = await database;
61    return await db.delete(
62      'mahasiswa',
63      where: 'id = ?',
64      whereArgs: [id],
65    );
66  }
67 }
68
```

## File view/add\_mahasiswa.dart

```
1 import 'package:flutter/material.dart';
2 import '../helper/db_helper.dart';
3
4 class AddMahasiswaView extends StatefulWidget {
5   final Function refreshData;
6
7   const AddMahasiswaView({Key? key, required this.refreshData})
8     : super(key: key);
9
10  @override
11  _AddMahasiswaViewState createState() => _AddMahasiswaViewState();
12 }
13
14 class _AddMahasiswaViewState extends State<AddMahasiswaView> {
15   final _namaController = TextEditingController();
16   final _nimController = TextEditingController();
17   final _alamatController = TextEditingController();
18   final _hobiController = TextEditingController();
19   final dbHelper = DatabaseHelper();
20
21   void addMahasiswa() async {
22     await dbHelper.insert({
23       'nama': _namaController.text,
24       'nim': _nimController.text,
25       'alamat': _alamatController.text,
26       'hobi': _hobiController.text,
27     });
28     widget.refreshData();
29     Navigator.pop(context);
30   }
31
32   // Method to build a text field with icon
33   Widget buildTextField({
34     required TextEditingController controller,
35     required String label,
36     required IconData icon,
37   }) {
38     return TextField(
39       controller: controller,
40       decoration: InputDecoration(
41         labelText: label,
42         prefixIcon: Icon(icon),
43         border: OutlineInputBorder(
44           borderRadius: BorderRadius.circular(8),
45         ),
46       ),
47     );
48   }
49
50   @override
51   Widget build(BuildContext context) {
52     return Scaffold(
53       appBar: AppBar(
54         title: const Text(
55           'Tambah Mahasiswa',
56           style: TextStyle(fontWeight: FontWeight.bold, color: Colors.white),
57         ),
58         centerTitle: true,
59         backgroundColor: Colors.brown,
60       ),
61       body: Padding(
62         padding: const EdgeInsets.all(16.0),
63         child: Column(
64           children: [
65             buildTextField(
66               controller: _namaController,
67               label: 'Nama',
68               icon: Icons.person,
69             ),
70             const SizedBox(height: 16),
71             buildTextField(
72               controller: _nimController,
73               label: 'NIM',
74               icon: Icons.book_outlined,
75             ),
76             const SizedBox(height: 16),
77             buildTextField(
78               controller: _alamatController,
79               label: 'Alamat',
80               icon: Icons.location_on,
81             ),
82             const SizedBox(height: 16),
83             buildTextField(
84               controller: _hobiController,
85               label: 'Hobi',
86               icon: Icons.try_sms_star,
87             ),
88             const SizedBox(height: 20),
89             ElevatedButton(
90               onPressed: addMahasiswa,
91               style: ElevatedButton.styleFrom(
92                 backgroundColor: Colors.brown,
93                 padding:
94                   const EdgeInsets.symmetric(vertical: 12, horizontal: 20),
95                 shape: RoundedRectangleBorder(
96                   borderRadius: BorderRadius.circular(8),
97                 ),
98               ),
99               child: const Text(
100                 'Simpan',
101                 style: TextStyle(
102                   fontSize: 16, color: Color.fromARGB(255, 255, 255, 255)),
103               ),
104             ),
105           ],
106         ),
107       ),
108     );
109   }
110 }
111
```



## File view/main\_view.dart

```
1 import 'package:flutter/material.dart';
2 import 'package:unguided/helper/db_helper.dart';
3 import 'package:unguided/view/add_mahasiswa_view.dart';
4
5 class MainView extends StatefulWidget {
6   @override
7   _MainViewState createState() => _MainViewState();
8 }
9
10 class _MainViewState extends State<MainView> {
11   final dbHelper = DatabaseHelper();
12   List<Map<String, dynamic>> mahasiswaList = [];
13
14   void refreshData() async {
15     final data = await dbHelper.queryAllRows();
16     setState(() {
17       mahasiswaList = data;
18     });
19   }
20
21   @override
22   void initState() {
23     super.initState();
24     refreshData();
25   }
26
27   @override
28   Widget build(BuildContext context) {
29     return Scaffold(
30       appBar: AppBar(
31         title: const Text(
32           'Biodata Mahasiswa',
33           style: TextStyle(
34             fontWeight: FontWeight.bold,
35             color: Color.fromARGB(255, 255, 255, 255)),
36       ),
37       centerTitle: true,
38       backgroundColor: Colors.brown,
39     ),
40     body: mahasiswaList.isEmpty
41       ? const Center(
42         child: Text(
43           'Belum ada data mahasiswa.',
44           style: TextStyle(fontSize: 16, color: Colors.grey),
45         ),
46       )
47       : ListView.builder(
48         itemCount: mahasiswaList.length,
49         padding: const EdgeInsets.symmetric(vertical: 10),
50         itemBuilder: (context, index) {
51           final item = mahasiswaList[index];
52           return Card(
53             margin:
54               const EdgeInsets.symmetric(vertical: 8, horizontal: 16),
55             shape: RoundedRectangleBorder(
56               borderRadius: BorderRadius.circular(12),
57             ),
58             elevation: 3,
59             child: ListTile(
60               contentPadding: const EdgeInsets.all(16),
61               title: Text(
62                 item['nama'],
63                 style: const TextStyle(
64                   fontWeight: FontWeight.bold, fontSize: 18),
65               ),
66               subtitle: Column(
67                 crossAxisAlignment: CrossAxisAlignment.start,
68                 children: [
69                   const SizedBox(height: 8),
70                   Text('NIM: ${item['nim']}'),
71                   Text('Alamat: ${item['alamat']}'),
72                   Text('Hobi: ${item['hobi']}'),
73                 ],
74               ),
75             ),
76           );
77         },
78       ),
79       floatingActionButton: FloatingActionButton(
80         backgroundColor: Colors.brown,
81         child: const Icon(Icons.add),
82         onPressed: () {
83           Navigator.push(
84             context,
85             MaterialPageRoute(
86               builder: (context) => AddMahasiswaView(refreshData: refreshData),
87             ),
88           );
89         },
90       ),
91     );
92   }
93 }
94
```

### File model/mahasiswa\_model.dart :

```
1  class Mahasiswa {
2    int? id;
3    String nama;
4    String nim;
5    String alamat;
6    String hobi;
7
8    Mahasiswa(
9      {this.id,
10     required this.nama,
11     required this.nim,
12     required this.alamat,
13     required this.hobi});
14
15     Map<String, dynamic> toMap() {
16       return {
17         'id': id,
18         'nama': nama,
19         'nim': nim,
20         'alamat': alamat,
21         'hobi': hobi,
22       };
23     }
24   }
25 }
```

### File main.dart

```
1  import 'package:flutter/material.dart';
2  import 'view/main_view.dart';
3
4  void main() {
5    runApp(const MainApp());
6  }
7
8  class MainApp extends StatelessWidget {
9    const MainApp({Key? key}) : super(key: key);
10
11    @override
12    Widget build(BuildContext context) {
13      return MaterialApp(
14        debugShowCheckedModeBanner: false,
15        home: MainView(),
16      );
17    }
18  }
19 }
```

## Output :

The image displays two screenshots of a mobile application interface for managing student data.

**Left Screenshot: Tambah Mahasiswa (Add Student)**

- Nama:** alpian
- NIM:** 2211104024
- Alamat:** wonosobo
- Hobi:** musikan
- Simpan** button

**Right Screenshot: Biodata Mahasiswa (Student Biodata)**

- Muhammad Abdul Aziz**  
NIM: 2211104026  
Alamat: Merden  
Hobi: Mbolang
- Devrin Anggun Saputri**  
NIM: 2211104001  
Alamat: Karangtengah  
Hobi: mancing
- alpian**  
NIM: 2211104024  
Alamat: wonosobo  
Hobi: musikan

A floating action button (+) is visible at the bottom right of the Biodata screen.

## Penjelasan :

Aplikasi ini adalah implementasi Flutter untuk pengelolaan data mahasiswa menggunakan SQLite dengan fokus pada operasi Create dan Read. Kelas `DatabaseHelper` bertanggung jawab mengelola database, termasuk membuat tabel `mahasiswa` dengan kolom `id`, `nama`, `nim`, `alamat`, dan `hobi`. Data mahasiswa dapat ditambahkan menggunakan halaman `AddMahasiswaView`, di mana pengguna mengisi form dengan informasi mahasiswa, yang kemudian disimpan ke database melalui metode `insert`. Data yang telah tersimpan ditampilkan dalam halaman `MainView` menggunakan ListView dengan widget Card. Antarmuka ini memberikan pengalaman sederhana untuk menambahkan data mahasiswa baru dan melihat daftar mahasiswa yang tersimpan di database.