

GUIDED
PRAKTIKUM PEMROGRAMAN PERANGKAT BERGERAK
MODUL X
DATA STORAGE BAGIAN 1



Disusun Oleh :
Maria Nathasya Desfera Pangestu / 2211104008
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

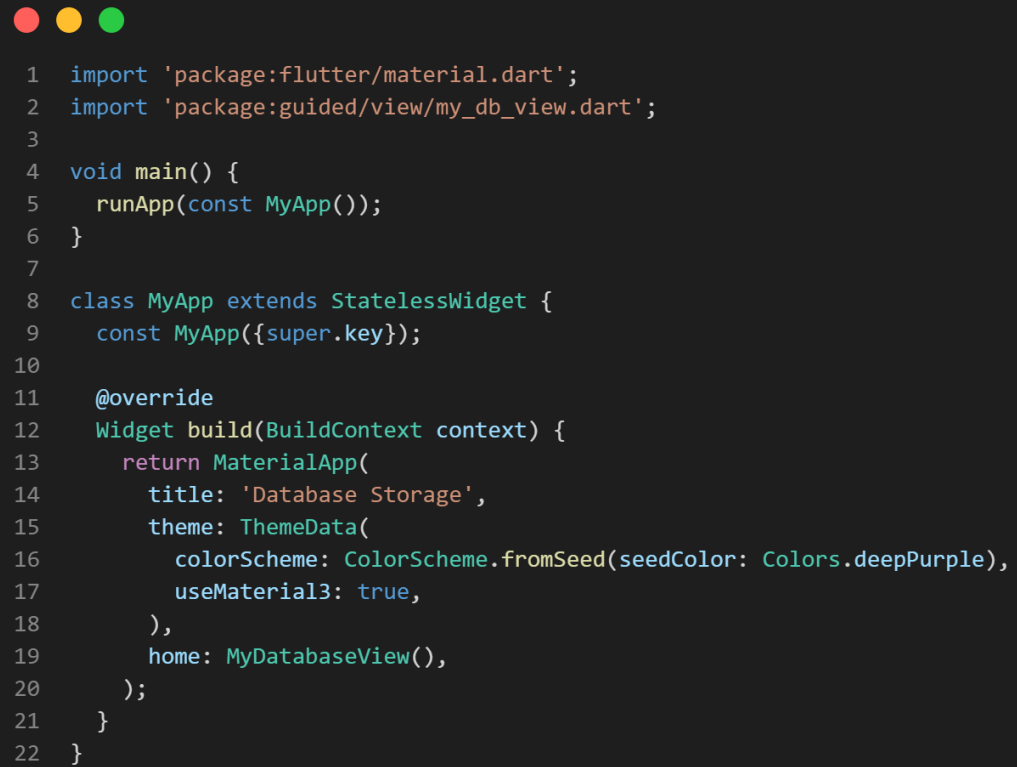
2024

GUIDED

1. Source code db_helper.dart

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

2. Source code main.dart



```
1  import 'package:flutter/material.dart';
2  import 'package:guided/view/my_db_view.dart';
3
4  void main() {
5    runApp(const MyApp());
6  }
7
8  class MyApp extends StatelessWidget {
9    const MyApp({super.key});
10
11    @override
12    Widget build(BuildContext context) {
13      return MaterialApp(
14        title: 'Database Storage',
15        theme: ThemeData(
16          colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
17          useMaterial3: true,
18        ),
19        home: MyDatabaseView(),
20      );
21    }
22  }
```

3. Source code my_db_view.dart

```
1  import 'package:flutter/material.dart';
2  import 'package:guided/helper/db_helper.dart';
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      _refreshData();
20      super.initState();
21    }
22
23    @override
24    void dispose() {
25      _titleController.dispose();
26      _descriptionController.dispose();
27      super.dispose();
28    }
29
30    void _refreshData() async {
31      final data = await dbHelper.queryAllRows();
32      setState(() {
33        _dbData = data;
34      });
35    }
36
37    void _addData() async {
38      if (_titleController.text.isEmpty || _descriptionController.text.isEmpty) {
39        _showSnackBar('Title and Description cannot be empty!');
40        return;
41      }
42      await dbHelper.insert({
43        'title': _titleController.text,
44        'description': _descriptionController.text,
45      });
46      _titleController.clear();
47      _descriptionController.clear();
48      _refreshData();
49    }
50
51    void _updateData(int id) async {
52      if (_titleController.text.isEmpty || _descriptionController.text.isEmpty) {
53        _showSnackBar('Title and Description cannot be empty!');
54        return;
55      }
56      await dbHelper.update({
57        'id': id,
58        'title': _titleController.text,
59        'description': _descriptionController.text,
60      });
61      _titleController.clear();
62      _descriptionController.clear();
63      _refreshData();
64    }
65
66    void _deleteData(int id) async {
67      await dbHelper.delete(id);
68      _refreshData();
69    }
70
71    void _showEditDialog(Map<String, dynamic> item) {
72      _titleController.text = item['title'];
73      _descriptionController.text = item['description'];
74    }
```

```

74
75     showDialog(
76         context: context,
77         builder: (context) {
78             return AlertDialog(
79                 title: const Text('Edit Item'),
80                 content: Column(
81                     mainAxisAlignment: MainAxisAlignment.min,
82                     children: [
83                         TextField(
84                             controller: _titleController,
85                             decoration: const InputDecoration(labelText: 'Title'),
86                         ),
87                         TextField(
88                             controller: _descriptionController,
89                             decoration: const InputDecoration(labelText: 'Description'),
90                         ),
91                     ],
92                 ),
93                 actions: [
94                     TextButton(
95                         onPressed: () {
96                             Navigator.of(context).pop();
97                         },
98                         child: const Text('Cancel'),
99                     ),
100                     TextButton(
101                         onPressed: () {
102                             _updateData(item['id']);
103                             Navigator.of(context).pop();
104                         },
105                         child: const Text('Save'),
106                     ),
107                 ],
108             );
109         },
110     );
111 }
112
113 void _showSnackBar(String message) {
114     ScaffoldMessenger.of(context)
115         .showSnackBar(SnackBar(content: Text(message)));
116 }
117

```

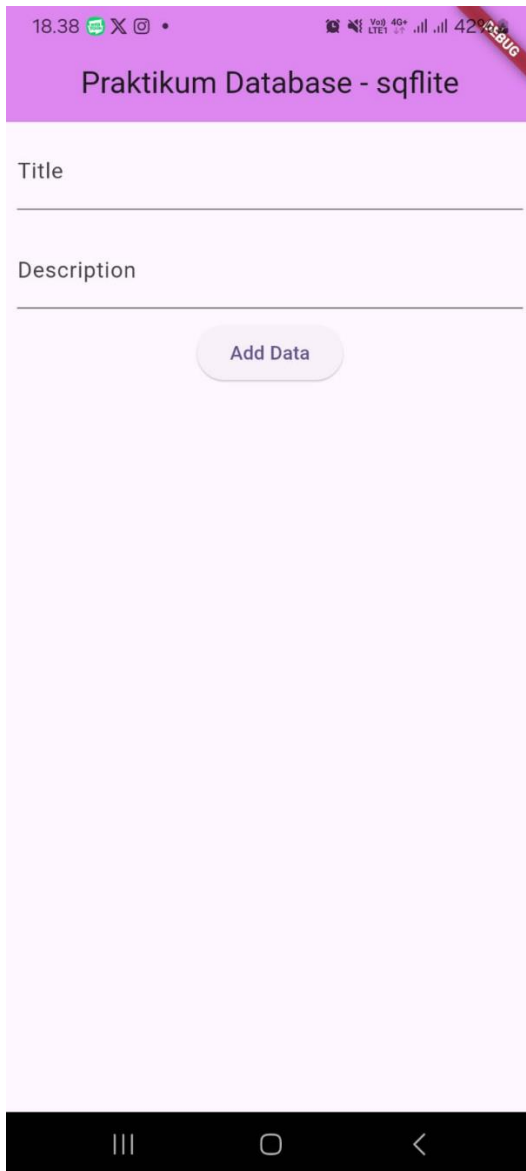
```

117
118 @override
119 Widget build(BuildContext context) {
120   return Scaffold(
121     appBar: AppBar(
122       title: const Text('Praktikum Database - sqflite'),
123       backgroundColor: const Color.fromARGB(255, 222, 135, 240),
124       centerTitle: true,
125     ),
126     body: Column(
127       children: [
128         Padding(
129           padding: const EdgeInsets.all(8.0),
130           child: TextField(
131             controller: _titleController,
132             decoration: const InputDecoration(labelText: 'Title'),
133           ),
134         ),
135         Padding(
136           padding: const EdgeInsets.all(8.0),
137           child: TextField(
138             controller: _descriptionController,
139             decoration: const InputDecoration(labelText: 'Description'),
140           ),
141         ),
142         ElevatedButton(
143           onPressed: _addData,
144           child: const Text('Add Data'),
145         ),
146         Expanded(
147           child: ListView.builder(
148             itemCount: _dbData.length,
149             itemBuilder: (context, index) {
150               final item = _dbData[index];
151               return ListTile(
152                 title: Text(item['title']),
153                 subtitle: Text(item['description']),
154                 trailing: Row(
155                   mainAxisAlignment: MainAxisAlignment.min,
156                   children: [
157                     IconButton(
158                       icon: const Icon(Icons.edit),
159                       onPressed: () {
160                         _showEditDialog(item);
161                       },
162                     ),
163                     IconButton(
164                       icon: const Icon(Icons.delete),
165                       onPressed: () => _deleteData(item['id']),
166                     ),
167                   ],
168                 ),
169               );
170             },
171           ),
172         ),
173       ],
174     ),
175   );
176 }
177 }

```

4. Hasil Output

Tampilan awal dan saat menambahkan data



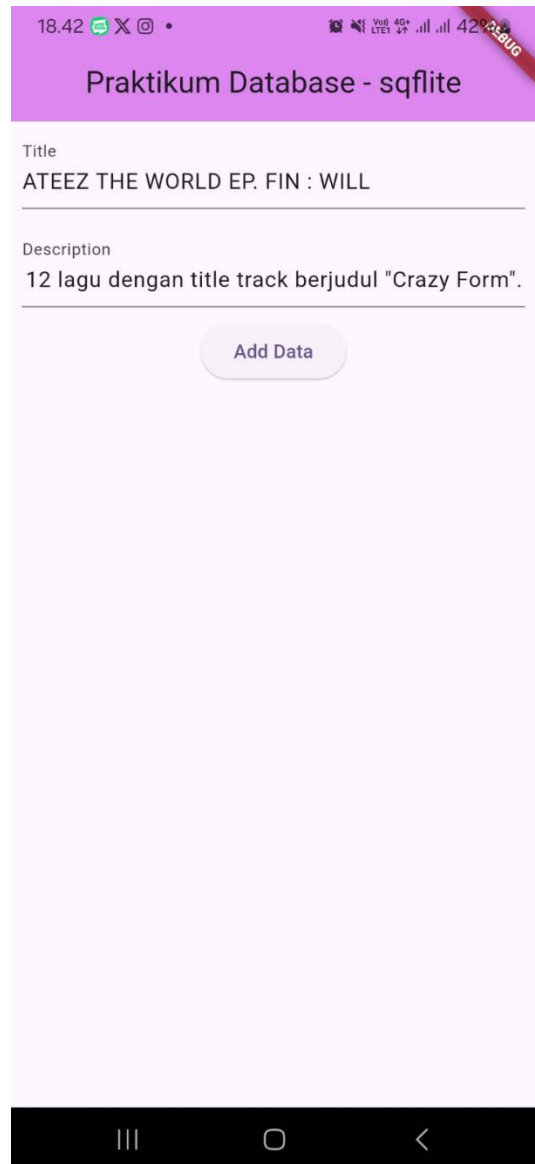
18.38

Praktikum Database - sqflite

Title

Description

Add Data



18.42

Praktikum Database - sqflite

Title






ATEEZ THE WORLD EP. FIN : WILL

Description

12 lagu dengan title track berjudul "Crazy Form".

Add Data

Tampilan sesudah mengklik “Add Data” dan menambahkan data baru

18.43    •  Vo 4G+ LTE1 42% 

Praktikum Database - sqflite



Title




Description

[Add Data](#)

ATEEZ THE WORLD EP. FIN :
WILL

Album ini terdiri dari 12 lagu
dengan title track berjudul "Crazy
Form".

18.44   Vo 4G LTE1 42% 

Praktikum Database - sqflite

Title

ATEEZ GOLDEN HOUR : Part.2



Description

agu dengan title track berjudul "Ice On My Teeth".




[Add Data](#)

ATEEZ THE WORLD EP. FIN :
WILL

Album ini terdiri dari 12 lagu
dengan title track berjudul "Crazy
Form".

Tampilan sesudah menambahkan data kedua dan ketiga



18.46  VoLTE 4G  42% 



Praktikum Database - sqflite



Title
ATEEZ TREASURE EP. 3 : One To All




Description
gan 2 title track berjudul "WAVE" dan "ILLUSION".

[Add Data](#)

ATEEZ THE WORLD EP. FIN : WILL
Album ini terdiri dari 12 lagu dengan title track berjudul "Crazy Form".  

ATEEZ GOLDEN HOUR : Part.2
Album ini terdiri dari 6 lagu dengan title track berjudul "Ice On My Teeth".  

III  



18.46  VoLTE 4G  42% 



Praktikum Database - sqflite



Title



Description

[Add Data](#)

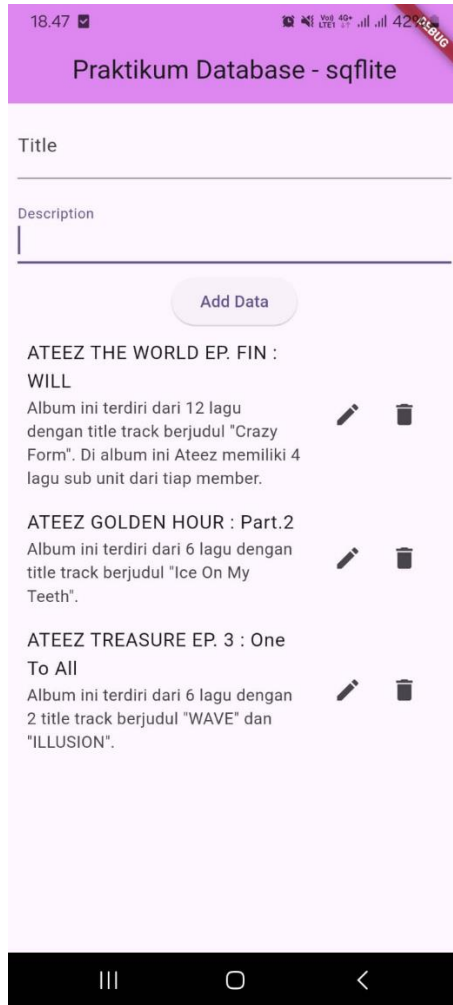
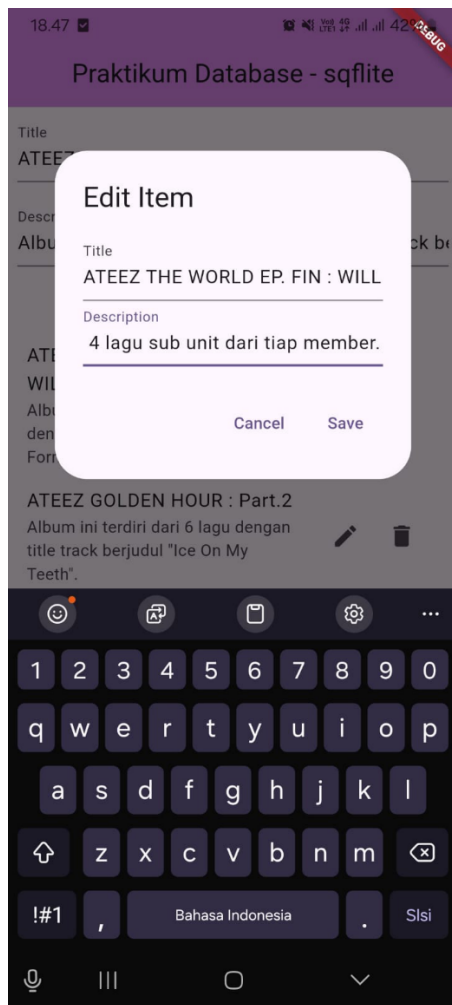
ATEEZ THE WORLD EP. FIN : WILL
Album ini terdiri dari 12 lagu dengan title track berjudul "Crazy Form".  

ATEEZ GOLDEN HOUR : Part.2
Album ini terdiri dari 6 lagu dengan title track berjudul "Ice On My Teeth".  

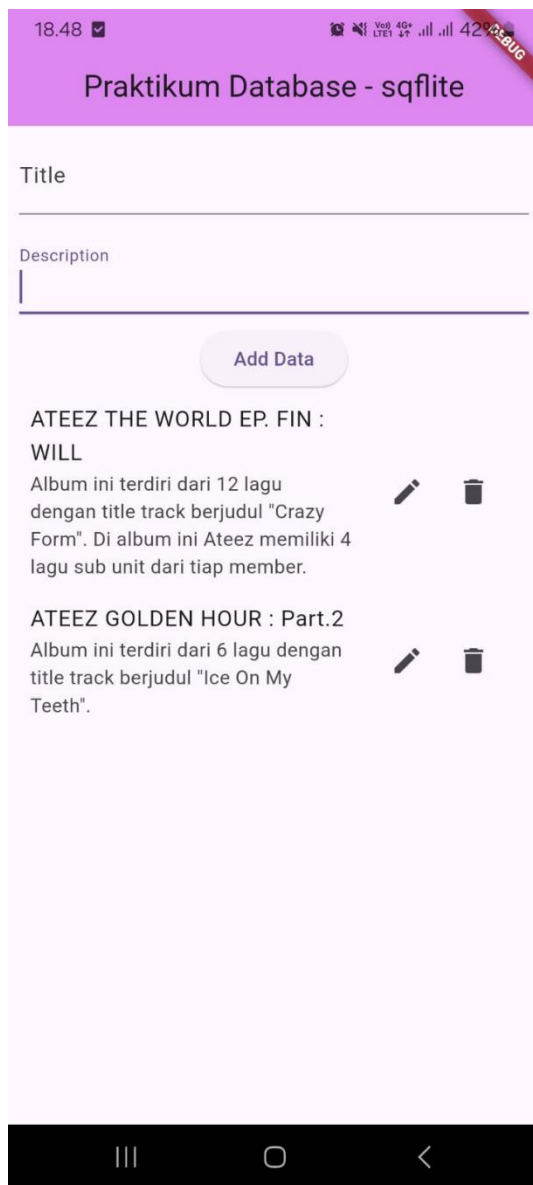
ATEEZ TREASURE EP. 3 : One To All
Album ini terdiri dari 6 lagu dengan 2 title track berjudul "WAVE" dan "ILLUSION".  

III  

Tampilan saat mengedit data dan setelah data diedit



Tampilan akhir setelah salah satu data dihapus



5. Penjelasan

Kode Flutter ini berfungsi untuk menampilkan data dari sebuah database. Dua library penting yang diimpor pada awal kode adalah `flutter/material.dart` menyediakan berbagai komponen untuk membuat antarmuka pengguna, lalu `guided/view/my_db_view.dart` sebagai tampilan utama aplikasi. Proses eksekusi dimulai dari fungsi `main()`, di mana kita menjalankan widget `MyApp` dan kelas `MyApp` mewakili seluruh aplikasi. Dalam metode `build` `MaterialApp` digunakan untuk membangun tampilan utama yang mengatur tema dasar aplikasi, seperti warna, font, dan juga menetapkan `MyDatabaseView` sebagai halaman utama. Kode programnya memungkinkan pengguna untuk bisa mengambil data (membaca data dari database dan menyimpannya untuk dilihat), menampilkan data (menampilkan data dalam format yang mudah dibaca oleh pengguna, seperti daftar, tabel, atau grafik), memungkinkan interaksi (seperti menambahkan, mengedit, atau menghapus data dari database).