PRAKTIKUM_8

-

Daftar Isi

1 TUGAS_1

1.1 lib/repositories/repositories.dart

```dart
export 'user_repository.dart';
```

1.2 lib/repositories/user_repository.dart

```dart
import 'package:sqflite/sqflite.dart';
import 'package:tugas_1/models/models.dart';
import 'package:tugas_1/services/services.dart';

class UserRepository {
  final tableName = 'users';

  Future<User?> getUserByUsername(String username) async {
    final db = await DBService.getDatabase();
    final result = await db.query(
      tableName,
      columns: [
        'id',
        'fullname',
        'username',
        'password',
      ],
      where: 'username = ?',
      whereArgs: [username],
    );

    db.close();

    if (result.isNotEmpty) {
      return User.fromMap(result.first);
    } else {
      return null;
    }
  }

  Future<void> addUser(User newUser) async {
    final db = await DBService.getDatabase();
    await db.insert(
      tableName,
      newUser.toMap(),
      conflictAlgorithm: ConflictAlgorithm.replace,
    );
    db.close();
  }
}
```

1.3 lib/models/user.dart

```dart
class User {
  int? _id;
  String? fullname;
  String username;
  String password;

  User({
    this.fullname,
    required this.username,
    required this.password,
    int? id,
  }) : _id = id;

  int? get id => _id;

  set id(int? id) {
    if (_id == null) {
      _id = id;
    } else {
      throw Exception('ID sudah diatur dan tidak bisa diubah lagi.');
    }
  }

  Map<String, dynamic> toMap() {
    return {
      'id': _id,
      'fullname': fullname,
      'username': username,
      'password': password,
    };
  }

  factory User.fromMap(Map<String, dynamic> map) {
    final user = User(
      fullname: map['fullname'],
      username: map['username'],
      password: map['password'],
      id: map['id'],
    );
    return user;
  }
}
```

1.4 lib/models/models.dart

```dart
export 'user.dart';
```

1.5 lib/main.dart

```dart
import 'package:flutter/material.dart';
import 'package:tugas_1/views/views.dart';

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

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

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      initialRoute: '/login',
      routes: {
        '/login': (context) => const LoginPage(),
        '/register': (context) => const RegisterPage(),
        '/main': (context) => const MainPage(),
      },
    );
  }
}
```

1.6 lib/views/register_page.dart

```dart
import 'package:flutter/material.dart';
import 'package:tugas_1/services/services.dart';

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

  @override
  State<RegisterPage> createState() => _RegisterPageState();
}

class _RegisterPageState extends State<RegisterPage> {
  final fullnameController = TextEditingController();
  final usernameController = TextEditingController();
  final passwordController = TextEditingController();
  final auth = AuthService();
```

```dart

  void _registerButtonAction() async {
    final fullname = fullnameController.text.trim();
    final username = usernameController.text.trim();
    final password = passwordController.text.trim();
    final success = await auth.register(fullname, username, password);
    debugPrint(auth.currentUser.toString());

    if (success && mounted) {
      _showSnackBar(content: 'Registrasi berhasil. Mengalihkan ke
          halaman home.');
      Navigator.of(context).pushReplacementNamed('/main');
    } else {
      _showSnackBar(content: 'Register gagal.');
    }
  }

  void _loginButtonAction() {
    Navigator.pushReplacementNamed(context, '/login');
  }

  void _showSnackBar({required String content}) {
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text(content)),
    );
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('Halaman Register'),
      ),
      body: SafeArea(
        child: Padding(
          padding: const EdgeInsets.all(16.0),
          child: Column(
            children: [
              TextField(
                controller: fullnameController,
                decoration: const InputDecoration(
                  labelText: 'Fullname',
                  border: OutlineInputBorder(),
                ),
              ),
              const SizedBox(height: 16),
              TextField(
                controller: usernameController,
                decoration: const InputDecoration(
                  labelText: 'Username',
                  border: OutlineInputBorder(),
```

```
66                  ),
67                ),
68                const SizedBox(height: 16),
69                TextField(
70                  controller: passwordController,
71                  obscureText: true,
72                  decoration: const InputDecoration(
73                    labelText: 'Password',
74                    border: OutlineInputBorder(),
75                  ),
76                ),
77                const SizedBox(height: 16),
78                ElevatedButton(
79                  onPressed: _registerButtonAction,
80                  child: const Text('Daftar'),
81                ),
82                const SizedBox(height: 16),
83                TextButton(
84                  onPressed: _loginButtonAction,
85                  child: const Text('Sudah punya akun? Login!'),
86                ),
87              ],
88            ),
89          ),
90        ),
91      );
92    }
93  }
```

1.7 lib/views/main_page.dart

```
1  import 'package:flutter/material.dart';
2  import 'package:tugas_1/services/services.dart';
3
4  class MainPage extends StatefulWidget {
5    const MainPage({super.key});
6
7    @override
8    State<MainPage> createState() => _MainPageState();
9  }
10
11 class _MainPageState extends State<MainPage> {
12   final auth = AuthService();
13
14   void _logoutButtonAction() {
15     final success = auth.logout();
16     if (success) {
17       Navigator.pushReplacementNamed(context, '/login');
18     }
```

```
19    }
20
21    @override
22    Widget build(BuildContext context) {
23      return Scaffold(
24        body: Center(
25          child: Column(
26            mainAxisAlignment: MainAxisAlignment.center,
27            children: [
28              Text(
29                  'Selamat datang ${auth.currentUser!.fullname}, Ini
                        halaman home'),
30              ElevatedButton(
31                onPressed: _logoutButtonAction,
32                child: const Text('Logout'),
33              ),
34            ],
35          ),
36        ),
37      );
38    }
39  }
```

1.8 lib/views/login_page.dart

```
1   import 'package:flutter/material.dart';
2   import 'package:tugas_1/services/services.dart';
3
4   class LoginPage extends StatefulWidget {
5     const LoginPage({super.key});
6
7     @override
8     State<LoginPage> createState() => _LoginPageState();
9   }
10
11  class _LoginPageState extends State<LoginPage> {
12    final usernameController = TextEditingController();
13    final passwordController = TextEditingController();
14    final auth = AuthService();
15
16    void _loginButtonAction() async {
17      final username = usernameController.text.trim();
18      final password = passwordController.text.trim();
19      final success = await auth.login(username, password);
20
21      if (success && mounted) {
22        Navigator.of(context).pushReplacementNamed('/main');
23      } else {
```

```
24        _showSnackBar(content: 'Login gagal. Periksa kembali kredensial
              Anda.');
25      }
26    }
27
28    void _registerButtonAction() {
29      Navigator.pushReplacementNamed(context, '/register');
30    }
31
32    void _showSnackBar({required String content}) {
33      ScaffoldMessenger.of(context).showSnackBar(
34        SnackBar(content: Text(content)),
35      );
36    }
37
38    @override
39    Widget build(BuildContext context) {
40      return Scaffold(
41        appBar: AppBar(
42          title: const Text('Halaman Login'),
43        ),
44        body: SafeArea(
45          child: Padding(
46            padding: const EdgeInsets.all(16.0),
47            child: Column(
48              children: [
49                TextField(
50                  controller: usernameController,
51                  decoration: const InputDecoration(
52                    labelText: 'Username',
53                    border: OutlineInputBorder(),
54                  ),
55                ),
56                const SizedBox(height: 16),
57                TextField(
58                  controller: passwordController,
59                  obscureText: true,
60                  decoration: const InputDecoration(
61                    labelText: 'Password',
62                    border: OutlineInputBorder(),
63                  ),
64                ),
65                const SizedBox(height: 16),
66                ElevatedButton(
67                  onPressed: _loginButtonAction,
68                  child: const Text('Login'),
69                ),
70                const SizedBox(height: 16),
71                TextButton(
72                  onPressed: _registerButtonAction,
73                  child: const Text('Belum Punya Akun? Daftar!'),
```

```
                ),
              ],
            ),
          ),
        ),
      );
    }
}
```

1.9 lib/views/views.dart

```dart
export 'login_page.dart';
export 'main_page.dart';
export 'register_page.dart';
```

1.10 lib/services/db_service.dart

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

class DBService {
  static Database? _database;

  static Future<Database> getDatabase() async {
    if (_database != null) return _database!;

    _database = await _initDB();
    return _database!;
  }

  static Future<Database> _initDB() async {
    final dbPath = await getDatabasesPath();
    final path = join(dbPath, 'app_database.db');

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

  static Future<void> _onCreate(Database db, int version) async {
    await db.execute('''
      CREATE TABLE users(
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        fullname TEXT,
        username TEXT,
```

```
        password TEXT
      )
    ''');
  }

  static Future<void> close() async {
    final db = await getDatabase();
    db.close();
  }
}
```

1.11 lib/services/auth_service.dart

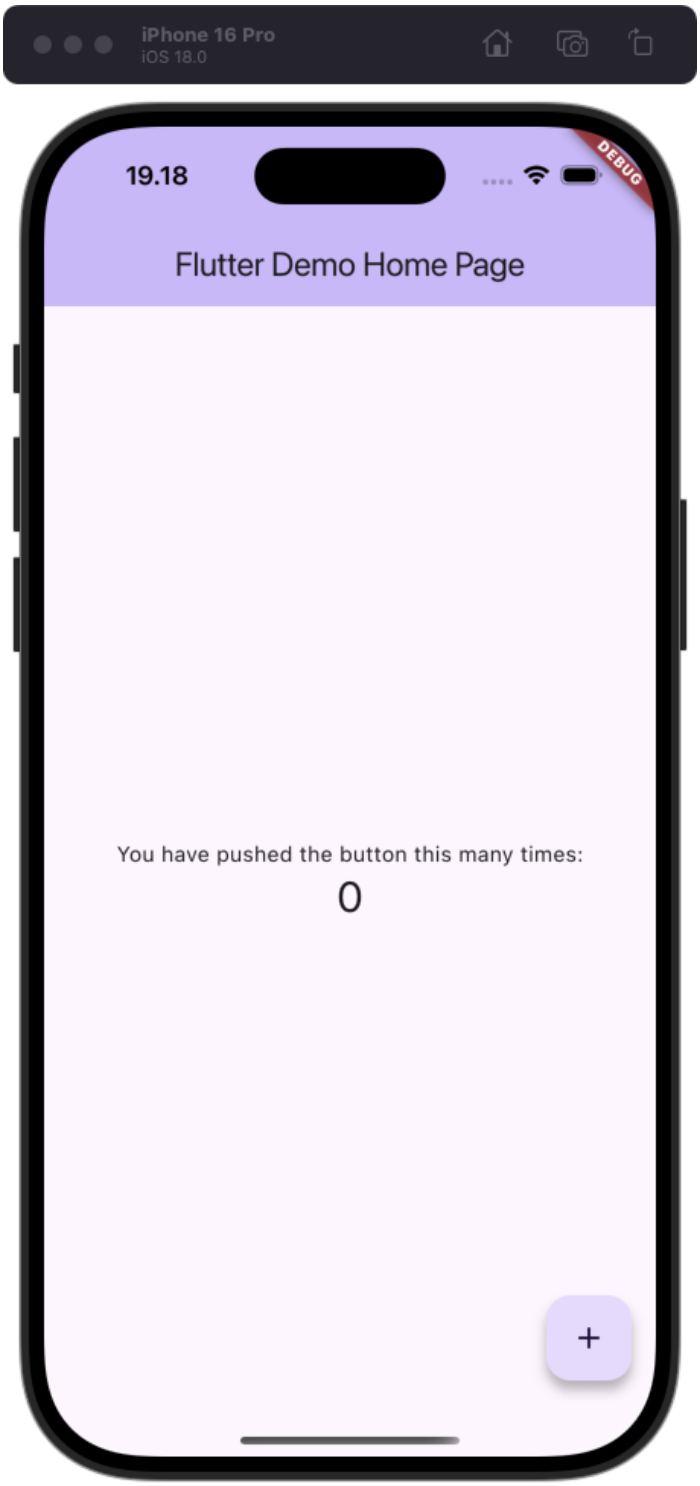
```dart
import 'dart:convert';
import 'package:crypto/crypto.dart';
import 'package:tugas_1/repositories/repositories.dart';
import 'package:tugas_1/models/models.dart';

class AuthService {
  static final AuthService _instance = AuthService._internal();
  factory AuthService() => _instance;
  AuthService._internal();

  final UserRepository _userRepository = UserRepository();

  User? _currentUser;

  User? get currentUser => _currentUser;

  String _hashPassword(String password) {
    final bytes = utf8.encode(password);
    final digest = sha256.convert(bytes);
    return digest.toString();
  }

  Future<bool> login(String username, String password) async {
    final hashedPassword = _hashPassword(password);
    final user = await _userRepository.getUserByUsername(username);

    if (user != null && user.password == hashedPassword) {
      _currentUser = user;
      return true;
    }

    return false;
  }

  Future<bool> register(
    String fullname,
```

```
37      String username,
38      String password,
39    ) async {
40      final existingUser = await _userRepository.getUserByUsername(
          username);
41      existingUser ?? false;
42
43      final newUser = User(
44        fullname: fullname,
45        username: username,
46        password: _hashPassword(password),
47      );
48
49      await _userRepository.addUser(newUser);
50      _currentUser = newUser;
51
52      return true;
53    }
54
55    bool logout() {
56      _currentUser = null;
57      return true;
58    }
59
60    bool isLoggedIn() {
61      return _currentUser != null;
62    }
63  }
```

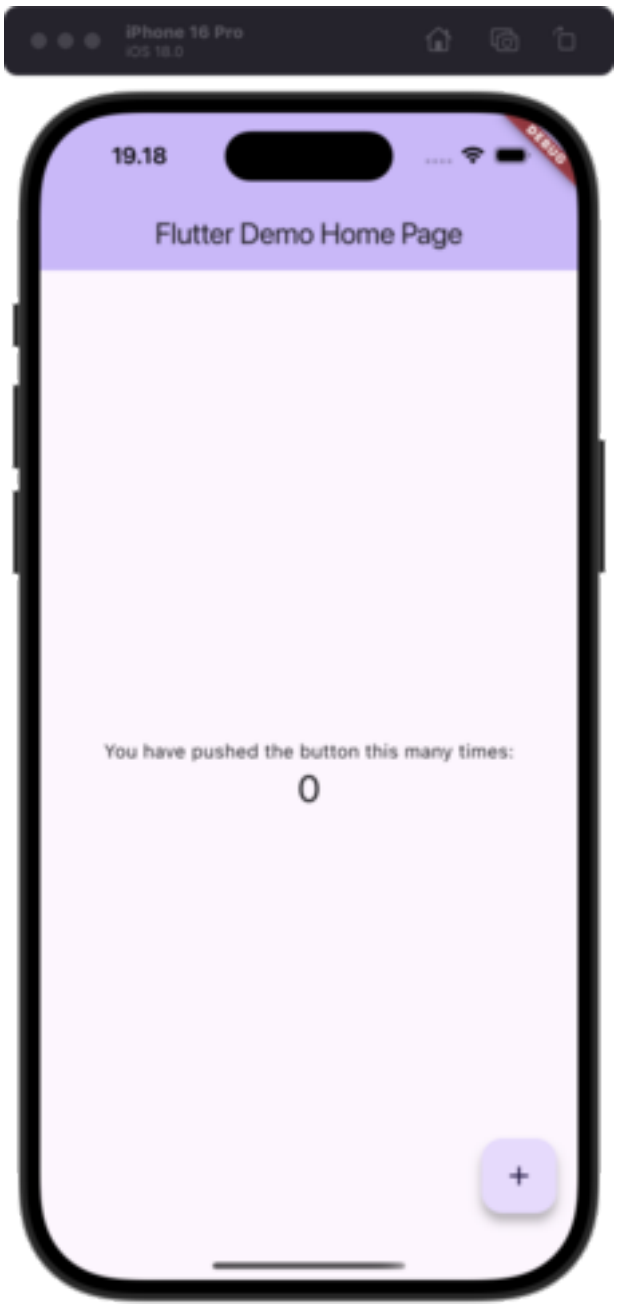
1.12 lib/services/services.dart

```
1  export 'auth_service.dart';
2  export 'db_service.dart';
```

1.13 screenshot



Screenshot 1: Screenshot 2024-11-24 at 19.18.34



Screenshot 2: Nov 24 Screenshot from iLoveIMG

2 LATIHAN_SQLITE

2.1 lib/database/db_helper.dart

```dart
import 'package:flutter_aplication_sqlite/models/contact.dart';
import 'package:sqflite/sqflite.dart';
import 'package:path/path.dart' as p;

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

  final String tableName = 'tablecontact';
  final String columnId = 'id';
  final String columnName = 'name';
  final String columnPhone = 'phone';
  final String columnEmail = 'email';
  final String columnCompany = 'company';
  DbHelper._internal();
  factory DbHelper() => _instance;

  Future<Database?> get _db async {
    if (_database != null) {
      return _database;
    }
    _database = await _initDb();
    return _database;
  }

  Future<Database?> _initDb() async {
    String databasePath = await getDatabasesPath();
    String path = p.join(databasePath, 'contact.db');
    return await openDatabase(path, version: 1, onCreate: _onCreate);
  }

  Future<void> _onCreate(Database db, int version) async {
    var sql = "CREATE TABLE $tableName($columnId INTEGER PRIMARY KEY, "
        "$columnName TEXT,"
        "$columnPhone TEXT,"
        "$columnEmail TEXT,"
        "$columnCompany TEXT)";
    await db.execute(sql);
  }

  Future<int?> savecontact(Contact contact) async {
    var dbClient = await _db;
    return await dbClient!.insert(tableName, contact.toMap());
  }

  Future<List?> getAllcontact() async {
```

```
47      var dbClient = await _db;
48      var result = await dbClient!.query(tableName, columns: [
49        columnId,
50        columnName,
51        columnCompany,
52        columnPhone,
53        columnEmail
54      ]);
55      return result.toList();
56    }
57
58    Future<int?> updatecontact(Contact contact) async {
59      var dbClient = await _db;
60      return await dbClient!.update(tableName, contact.toMap(),
61          where: '$columnId = ?', whereArgs: [contact.id]);
62    }
63
64    Future<int?> deletecontact(int id) async {
65      var dbClient = await _db;
66      return await dbClient!
67          .delete(tableName, where: '$columnId = ?', whereArgs: [id]);
68    }
69  }
```

2.2 lib/models/contact.dart

```
1   class Contact {
2     int? id;
3     String? name;
4     String? phone;
5     String? email;
6     String? company;
7
8     Contact({
9       this.id,
10      this.name,
11      this.phone,
12      this.email,
13      this.company,
14    });
15
16    Map<String, dynamic> toMap() {
17      var map = <String, dynamic>{};
18      if (id != null) {
19        map['id'] = id;
20      }
21      map['name'] = name;
22      map['phone'] = phone;
23      map['email'] = email;
```

```
    map['company'] = company;
    return map;
  }

  Contact.fromMap(Map<String, dynamic> map) {
    id = map['id'];
    name = map['name'];
    phone = map['phone'];
    email = map['email'];
    company = map['company'];
  }
}
```

2.3 lib/main.dart

```
import 'package:flutter/material.dart';
import 'package:flutter_aplication_sqlite/contact_page.dart';

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

class MainApp extends StatelessWidget {
  const MainApp({super.key});
  @override
  Widget build(BuildContext context) {
    return const MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Flutter Demo',
      home: ContactPage(),
    );
  }
}
```

2.4 lib/add_contact_page.dart

```
// ignore_for_file: prefer_const_constructors_in_immutables
import 'package:flutter/material.dart';
import 'package:flutter_aplication_sqlite/database/db_helper.dart';
import 'package:flutter_aplication_sqlite/models/contact.dart';

class AddContactPage extends StatefulWidget {
  final Contact? contact;

  AddContactPage({super.key, this.contact});

  @override
```

```
12    State<AddContactPage> createState() => _AddContactPageState();
13  }
14
15  class _AddContactPageState extends State<AddContactPage> {
16    final DbHelper db = DbHelper();
17
18    late TextEditingController nameController;
19    late TextEditingController phoneController;
20    late TextEditingController emailController;
21    late TextEditingController companyController;
22
23    @override
24    void initState() {
25      super.initState();
26      nameController = TextEditingController(
27          text: widget.contact == null ? '' : widget.contact!.name);
28      phoneController = TextEditingController(
29          text: widget.contact == null ? '' : widget.contact!.phone);
30      emailController = TextEditingController(
31          text: widget.contact == null ? '' : widget.contact!.email);
32      companyController = TextEditingController(
33          text: widget.contact == null ? '' : widget.contact!.company);
34    }
35
36    @override
37    void dispose() {
38      nameController.dispose();
39      phoneController.dispose();
40      emailController.dispose();
41      companyController.dispose();
42      super.dispose();
43    }
44
45    @override
46    Widget build(BuildContext context) {
47      return Scaffold(
48        appBar: AppBar(
49          title: const Text('Form Contact'),
50        ),
51        body: ListView(
52          padding: const EdgeInsets.all(16.0),
53          children: [
54            Padding(
55              padding: const EdgeInsets.only(top: 20),
56              child: TextField(
57                controller: nameController,
58                decoration: InputDecoration(
59                  labelText: 'Name',
60                  border: OutlineInputBorder(
61                    borderRadius: BorderRadius.circular(8),
62                  ),
```

```
63                ),
64              ),
65            ),
66            Padding(
67              padding: const EdgeInsets.only(top: 20),
68              child: TextField(
69                controller: phoneController,
70                decoration: InputDecoration(
71                  labelText: 'Mobile No',
72                  border: OutlineInputBorder(
73                    borderRadius: BorderRadius.circular(8),
74                  ),
75                ),
76              ),
77            ),
78            Padding(
79              padding: const EdgeInsets.only(top: 20),
80              child: TextField(
81                controller: emailController,
82                decoration: InputDecoration(
83                  labelText: 'Email',
84                  border: OutlineInputBorder(
85                    borderRadius: BorderRadius.circular(8),
86                  ),
87                ),
88              ),
89            ),
90            Padding(
91              padding: const EdgeInsets.only(top: 20),
92              child: TextField(
93                controller: companyController,
94                decoration: InputDecoration(
95                  labelText: 'Company',
96                  border: OutlineInputBorder(
97                    borderRadius: BorderRadius.circular(8),
98                  ),
99                ),
100             ),
101           ),
102           Padding(
103             padding: const EdgeInsets.only(top: 20),
104             child: ElevatedButton(
105               onPressed: () => _upsertContact(),
106               child: Text(widget.contact == null ? 'Add' : 'Update'),
107             ),
108           ),
109         ],
110       ),
111     );
112   }
113
```

```
114   Future<void> _upsertContact() async {
115   if (widget.contact != null) {
116     // Update contact
117     await db.updatecontact(Contact(
118       id: widget.contact!.id,
119       name: nameController.text,
120       phone: phoneController.text,
121       email: emailController.text,
122       company: companyController.text,
123     ));
124
125     if (mounted) {
126       // Gunakan context jika widget masih aktif
127       Navigator.pop(context, 'update');
128     }
129   } else {
130     // Add new contact
131     await db.savecontact(Contact(
132       name: nameController.text,
133       phone: phoneController.text,
134       email: emailController.text,
135       company: companyController.text,
136     ));
137
138     if (mounted) {
139       // Gunakan context jika widget masih aktif
140       Navigator.pop(context, 'save');
141     }
142   }
143 }
144
145 }
```

2.5 lib/contact_page.dart

```
1  import 'package:flutter/material.dart';
2  import 'package:flutter_aplication_sqlite/add_contact_page.dart';
3  import 'package:flutter_aplication_sqlite/database/db_helper.dart';
4  import 'package:flutter_aplication_sqlite/models/contact.dart';
5
6  class ContactPage extends StatefulWidget {
7    const ContactPage({super.key});
8
9    @override
10   State<ContactPage> createState() => _ContactPageState();
11 }
12
13 class _ContactPageState extends State<ContactPage> {
14   List<Contact> listContact = [];
```

```dart
  final DbHelper db = DbHelper();

  @override
  void initState() {
    super.initState();
    // Menjalankan fungsi getAllContact saat pertama kali dimuat
    _getAllContact();
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Center(
          child: Text("Contact App"),
        ),
      ),
      body: ListView.builder(
        itemCount: listContact.length,
        itemBuilder: (context, index) {
          Contact contact = listContact[index];
          return Padding(
            padding: const EdgeInsets.only(top: 20),
            child: ListTile(
              leading: const Icon(
                Icons.person,
                size: 50,
              ),
              title: Text(contact.name ?? ''),
              subtitle: Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                  const SizedBox(height: 8),
                  Text("Email: ${contact.email ?? '-'}"),
                  const SizedBox(height: 8),
                  Text("Phone: ${contact.phone ?? '-'}"),
                  const SizedBox(height: 8),
                  Text("Company: ${contact.company ?? '-'}"),
                ],
              ),
              trailing: FittedBox(
                fit: BoxFit.fill,
                child: Row(
                  children: [
                    // Button edit
                    IconButton(
                      onPressed: () {
                        _openFormEdit(contact);
                      },
                      icon: const Icon(Icons.edit),
                    ),
```

```
                    // Button hapus
                    IconButton(
                      icon: const Icon(Icons.delete),
                      onPressed: () {
                        _showDeleteConfirmation(contact, index);
                      },
                    ),
                  ],
                ),
              ),
            ),
          );
        },
      ),
      // Membuat button mengapung di bagian bawah kanan layar
      floatingActionButton: FloatingActionButton(
        onPressed: _openFormCreate,
        child: const Icon(Icons.add),
      ),
    );
  }

  Future<void> _getAllContact() async {
    var list = await db.getAllcontact();
    setState(() {
      listContact.clear();
      for (var contact in list ?? []) {
        listContact.add(Contact.fromMap(contact));
      }
    });
  }

  Future<void> _deleteContact(Contact contact, int position) async {
    await db.deletecontact(contact.id!);
    setState(() {
      listContact.removeAt(position);
    });
  }

  Future<void> _openFormCreate() async {
    var result = await Navigator.push(
      context,
      MaterialPageRoute(builder: (context) => AddContactPage()),
    );
    if (result == 'save') {
      await _getAllContact();
    }
  }

  Future<void> _openFormEdit(Contact contact) async {
    var result = await Navigator.push(
```

```
117         context,
118         MaterialPageRoute(
119           builder: (context) => AddContactPage(contact: contact),
120         ),
121       );
122       if (result == 'update') {
123         await _getAllContact();
124       }
125     }
126
127     void _showDeleteConfirmation(Contact contact, int index) {
128       showDialog(
129         context: context,
130         builder: (context) {
131           return AlertDialog(
132             title: const Text("Information"),
133             content: Text("Yakin ingin menghapus data\n\n${contact.name}?
                  "),
134             actions: [
135               TextButton(
136                 onPressed: () {
137                   _deleteContact(contact, index);
138                   Navigator.pop(context);
139                 },
140                 child: const Text("Ya"),
141               ),
142               TextButton(
143                 onPressed: () {
144                   Navigator.pop(context);
145                 },
146                 child: const Text("Tidak"),
147               ),
148             ],
149           );
150         },
151       );
152     }
153   }
```

2.6 screenshot

Screenshot 3: screenshot-4

Screenshot 4: screenshot-5

Screenshot 5: screenshot-2

Screenshot 6: screenshot-3

Screenshot 7: screenshot-1

3 TUGAS_2

3.1 lib/repositories/repositories.dart

```dart
export 'product_repository.dart';
```

3.2 lib/repositories/product_repository.dart

```dart
import 'package:inventory_app/services/services.dart';
import 'package:inventory_app/models/models.dart';
import 'package:sqflite/sqflite.dart';

class ProductRepository {
  final tableName = 'products';

  Future<void> addProduct(Product newProduct) async {
    final db = await DBService.getDatabase();
    final id = await db.insert(
      tableName,
      newProduct.toMap(),
      conflictAlgorithm: ConflictAlgorithm.replace,
    );
    db.close();
    newProduct.id = id;
  }

  Future<List<Product>?> getAllProduct() async {
    final db = await DBService.getDatabase();
    final result = await db.query(tableName, columns: [
      'id',
      'name',
      'purchase_price',
      'selling_price',
      'stock',
    ]);
    db.close();

    if (result.isNotEmpty) {
      return result.map((productMap) => Product.fromMap(productMap)).
        toList();
    } else {
      return null;
    }
  }

  Future<int?> updateProduct(Product product) async {
    final db = await DBService.getDatabase();
    final result = await db.update(
```

```
      tableName,
      product.toMap(),
      where: 'id = ?',
      whereArgs: [product.id.toString()],
    );
    db.close();

    return result;
  }

  Future<int?> deleteProduct(Product product) async {
    final db = await DBService.getDatabase();
    final result = await db.delete(
      tableName,
      where: 'id = ?',
      whereArgs: [product.id.toString()],
    );
    db.close();

    return result;
  }
}
```

3.3 lib/models/product.dart

```
class Product {
  int? _id;
  String name;
  int purchasePrice;
  int sellingPrice;
  int stock;

  Product({
    int? id,
    required this.name,
    required this.purchasePrice,
    required this.sellingPrice,
    required this.stock,
  }) : _id = id;

  int? get id => _id;

  set id(int? id) {
    if (_id == null) {
      _id = id;
    } else {
      throw Exception('ID sudah diatur dan tidak bisa dirubah lagi');
    }
  }
```

```
25
26    Map<String, dynamic> toMap() {
27      return {
28        'id': _id,
29        'name': name,
30        'purchase_price': purchasePrice,
31        'selling_price': sellingPrice,
32        'stock': stock,
33      };
34    }
35
36    factory Product.fromMap(Map<String, dynamic> map) {
37      return Product(
38        name: map['name'],
39        purchasePrice: map['purchase_price'],
40        sellingPrice: map['selling_price'],
41        stock: map['stock'],
42        id: map['id'],
43      );
44    }
45  }
```

3.4 lib/models/models.dart

```
1  export 'product.dart';
```

3.5 lib/main.dart

```
1  import 'package:flutter/material.dart';
2  import 'package:inventory_app/views/views.dart';
3
4  void main() {
5    runApp(const App());
6  }
7
8  class App extends StatefulWidget {
9    const App({super.key});
10
11   @override
12   State<App> createState() => _AppState();
13 }
14
15 class _AppState extends State<App> {
16   @override
17   Widget build(BuildContext context) {
18     return MaterialApp(
19       home: HomePage(),
```

```
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple)
            ,
        useMaterial3: true,
      ),
    );
  }
}
```

3.6 lib/views/home_page.dart

```
import 'package:flutter/material.dart';
import 'package:inventory_app/models/models.dart';
import 'package:inventory_app/repositories/repositories.dart';
import 'package:inventory_app/views/views.dart';

class HomePage extends StatelessWidget {
  final productRepository = ProductRepository();

  HomePage({super.key});

  void _updateProduct(BuildContext context, Product product) {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (context) => ProductFormPage(product: product),
      ),
    );
  }

  @override
  Widget build(BuildContext context) {
    final product = Product(
      name: 'Barang 1',
      purchasePrice: 10,
      sellingPrice: 10,
      stock: 10,
    );

    return Scaffold(
      appBar: AppBar(title: const Text('Inventory App')),
      floatingActionButton: FloatingActionButton(
        onPressed: () => _updateProduct(context, product),
        tooltip: 'Tambah Barang',
        child: const Icon(Icons.add),
      ),
      body: FutureBuilder<List<Product>?>(
        future: productRepository.getAllProduct(),
        builder: (context, snapshot) {
```

```dart
          final connection = snapshot.connectionState;
          if (connection == ConnectionState.waiting) {
            return const Center(
              child: CircularProgressIndicator(),
            );
          }
          if (snapshot.hasError) {
            return Center(
              child: Text('Terjadi kesalahan: ${snapshot.error}'),
            );
          }
          if (snapshot.hasData && snapshot.data!.isNotEmpty) {
            final products = snapshot.data!;
            return ListView.builder(
              itemCount: products.length,
              itemBuilder: (context, index) {
                final currentProduct = products.elementAt(index);
                return Card(
                  elevation: 5,
                  margin:
                      const EdgeInsets.symmetric(horizontal: 16,
                          vertical: 8),
                  child: Padding(
                    padding: const EdgeInsets.all(12),
                    child: Row(
                      children: [
                        CircleAvatar(
                          backgroundColor: Colors.blueAccent,
                          child: Text(
                            currentProduct.name[0].toUpperCase(),
                            style: const TextStyle(
                              color: Colors.white,
                              fontWeight: FontWeight.bold,
                            ),
                          ),
                        ),
                        const SizedBox(width: 16),
                        Expanded(
                          child: Column(
                            crossAxisAlignment: CrossAxisAlignment.
                                start,
                            children: [
                              Text(
                                currentProduct.name,
                                style: const TextStyle(
                                  fontSize: 18,
                                  fontWeight: FontWeight.bold,
                                ),
                              ),
                              const SizedBox(height: 4),
                              Text('ID: ${currentProduct.id}'),
```

```
                                Text(
                                    'Purchase Price: ${currentProduct.
                                       purchasePrice}'),
                                Text(
                                    'Selling Price: ${currentProduct.
                                       sellingPrice}'),
                                Text('Stock: ${currentProduct.stock}'),
                              ],
                            ),
                          ),
                          IconButton(
                            icon: const Icon(Icons.edit, color: Colors.
                               blue),
                            onPressed: () =>
                                _updateProduct(context, currentProduct),
                          ),
                        ],
                      ),
                    ),
                  );
                },
              );
            }
            return const Center(
              child: Text('Tidak ada produk tersedia'),
            );
          },
        ),
      );
    }
}
```

3.7 lib/views/product_form_page.dart

```
import 'package:flutter/material.dart';
import 'package:inventory_app/models/models.dart';

class ProductFormPage extends StatelessWidget {
  final Product product;

  const ProductFormPage({super.key, required this.product});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(),
      body: Center(
        child: Text('Data berhasil dikirim, ${product.name}'),
      ),
```

```
    );
  }
}
```

3.8 lib/views/views.dart

```
export 'home_page.dart';
export 'product_form_page.dart';
```

3.9 lib/services/db_service.dart

```
import 'dart:developer';

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

class DBService {
  static Database? _database;

  static Future<Database> getDatabase() async {
    if (_database != null) return _database!;

    _database = await _initDB();
    return _database!;
  }

  static Future<Database> _initDB() async {
    final dbPath = await getDatabasesPath();
    final path = join(dbPath, 'app_database.db');

    log('SQLite Location: $path');

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

  static Future<void> _onCreate(Database db, int version) async {
    await db.execute('''
      CREATE TABLE products(
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        name TEXT,
        purchase_price INTEGER,
        selling_price INTEGER,
        stock INTEGER
```

```
      )
    ''');
  }

  static Future<void> close() async {
    final db = await getDatabase();
    db.close();
  }
}
```

3.10 lib/services/services.dart

```
export 'db_service.dart';
```

3.11 screenshot