https://redesigned-goldfish-5gx45g6575xqh4g5x.github.dev/

```
// main.dart
import 'package:flutter/material.dart';
import 'package:provider/provider.dart';
import 'package:sqflite/sqflite.dart';
import 'package:path/path.dart';

void main() {
  runApp(ChangeNotifierProvider(
    create: (context) => PlanetProvider(),
    child: MyApp(),
  ));
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Planetas CRUD',
      theme: ThemeData(primarySwatch: Colors.blue),
      home: PlanetListScreen(),
    );
  }
}
```

```
class Planet {
  int? id;
  String name;
  String description;

  Planet({this.id, required this.name, required this.description});

  Map<String, dynamic> toMap() {
    return {'id': id, 'name': name, 'description': description};
  }

  factory Planet.fromMap(Map<String, dynamic> map) {
    return Planet(id: map['id'], name: map['name'], description: map['description']);
  }
}

class PlanetDatabase {
  static Future<Database> getDatabase() async {
    return openDatabase(
      join(await getDatabasesPath(), 'planets.db'),
      onCreate: (db, version) {
        return db.execute(
          "CREATE TABLE planets(id INTEGER PRIMARY KEY, name TEXT, description TEXT)"
        );
      },
      version: 1,
    );
  }
```

```
  static Future<void> insertPlanet(Planet planet) async {
    final db = await getDatabase();
    await db.insert('planets', planet.toMap(), conflictAlgorithm:
ConflictAlgorithm.replace);
  }

  static Future<List<Planet>> getPlanets() async {
    final db = await getDatabase();
    final List<Map<String, dynamic>> maps = await db.query('planets');
    return List.generate(maps.length, (i) => Planet.fromMap(maps[i]));
  }

  static Future<void> updatePlanet(Planet planet) async {
    final db = await getDatabase();
    await db.update('planets', planet.toMap(), where: 'id = ?', whereArgs: [planet.id]);
  }

  static Future<void> deletePlanet(int id) async {
    final db = await getDatabase();
    await db.delete('planets', where: 'id = ?', whereArgs: [id]);
  }
}

class PlanetProvider extends ChangeNotifier {
  List<Planet> _planets = [];

  List<Planet> get planets => _planets;
```

```
  Future<void> loadPlanets() async {
    _planets = await PlanetDatabase.getPlanets();
    notifyListeners();
  }

  Future<void> addPlanet(String name, String description) async {
    await PlanetDatabase.insertPlanet(Planet(name: name, description: description));
    await loadPlanets();
  }

  Future<void> updatePlanet(Planet planet) async {
    await PlanetDatabase.updatePlanet(planet);
    await loadPlanets();
  }

  Future<void> deletePlanet(int id) async {
    await PlanetDatabase.deletePlanet(id);
    await loadPlanets();
  }
}

class PlanetListScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text("Planetas")),
      body: Consumer<PlanetProvider>(
```

```
      builder: (context, provider, child) {
        return ListView.builder(
          itemCount: provider.planets.length,
          itemBuilder: (context, index) {
            final planet = provider.planets[index];
            return ListTile(
              title: Text(planet.name),
              subtitle: Text(planet.description),
              trailing: IconButton(
                icon: Icon(Icons.delete, color: Colors.red),
                onPressed: () => provider.deletePlanet(planet.id!),
              ),
              onTap: () {
                showDialog(
                  context: context,
                  builder: (context) {
                    TextEditingController nameController = TextEditingController(text:
planet.name);
                    TextEditingController descriptionController = TextEditingController(text:
planet.description);
                    return AlertDialog(
                      title: Text("Editar Planeta"),
                      content: Column(
                        mainAxisSize: MainAxisSize.min,
                        children: [
                          TextField(controller: nameController, decoration:
InputDecoration(labelText: "Nome")),
                          TextField(controller: descriptionController, decoration:
InputDecoration(labelText: "Descrição")),
```

```
                ],
              ),
              actions: [
                TextButton(
                  onPressed: () {
                    provider.updatePlanet(Planet(id: planet.id, name: nameController.text,
description: descriptionController.text));
                    Navigator.pop(context);
                  },
                  child: Text("Salvar"),
                )
              ],
            );
          },
        );
      },
    );
  },
),
floatingActionButton: FloatingActionButton(
  child: Icon(Icons.add),
  onPressed: () {
    showDialog(
      context: context,
      builder: (context) {
        TextEditingController nameController = TextEditingController();
```

```
        TextEditingController descriptionController = TextEditingController();
        return AlertDialog(
         title: Text("Adicionar Planeta"),
         content: Column(
          mainAxisSize: MainAxisSize.min,
          children: [
           TextField(controller: nameController, decoration: InputDecoration(labelText:
"Nome")),
           TextField(controller: descriptionController, decoration:
InputDecoration(labelText: "Descrição")),
          ],
         ),
         actions: [
          TextButton(
           onPressed: () {
            Provider.of<PlanetProvider>(context, listen:
false).addPlanet(nameController.text, descriptionController.text);
            Navigator.pop(context);
           },
           child: Text("Salvar"),
          )
         ],
        );
       },
      );
     },
    ),
   );
  }
```

```
}
```