UAS
PRAKTIKUM PEMROGRAMAN MOBILE

JUDUL JOBSHEET

.

Untuk Memenuhi Ujian Akhir Semester
Mata Kuliah Praktikum Pemrograman Mobile
Dosen Pengampu: Safri Adam, S.Kom., M.Kom

Disusun Oleh:

Aryo Wibowo (NIM: 3202116031)

Alda Rahmanita (NIM: 3202116076)

Uchi Hardiana (NIM: 3202116077)

Walillah (NIM: 3202116032)

Ifdul Rahman (NIM:3202116033)

PROGRAM STUDI D3 TEKNIK INFORMATIKA

JURUSAN TEKNIK ELEKTRO

POLITEKNIK NEGERI PONTIANAK

2023

KATA PENGANTAR

Puji dan syukur kami panjatkan kepada Allah SWT atas rahmat dan karunia-nya sehingga Ujian Akhir Semester" dapat terselesaikan dengan baik. ini merupakan salah satu UAS yang diberikan oleh dosen pengampu mata kuliah Praktikum Pemrograman Mobile kepada mahasiswa Program Studi D3 Teknik Informatika Jurusan Teknik Elektro.

Uas ini kami membahas tentang ..? di aplikasi mobile. Demikian laporan ini kami buat semoga bermanfaat.

Pontianak, 18 Agustus 2023

Penyusun,

(Kelompok ?)

UJIAN AKHIR SEMESTER

Data

Source Code :

```dart
import 'package:hive/hive.dart';
import 'package:hive_flutter/hive_flutter.dart';
import '../models/note.dart';

class HiveDatabase {
  final _myBox = Hive.box('note_database');

  List<Note> loadNotes() {
    List<Note> savedNotesFormatted = [];

    if (_myBox.get("ALL_NOTES") != null) {
      List<dynamic> savedNotes = _myBox.get("ALL_NOTES");
      for (int i = 0; i < savedNotes.length; i++) {
        Note individualNote =
            Note(id: savedNotes[i][0], text: savedNotes[i][1]);
        savedNotesFormatted.add(individualNote);
      }
    } else {
      savedNotesFormatted.add(
        Note(id: 0, text: 'First Note'),
      );
    }
    return savedNotesFormatted;
  }

  void savedNotes(List<Note> allNotes) {
    List<List<dynamic>> allNotesFormatted = [];
    for (var note in allNotes) {
      int id = note.id;
      String text = note.text;
      allNotesFormatted.add([id, text]);
    }

    _myBox.put("ALL_NOTES", allNotesFormatted);
  }
}

```

Keterangan :

Kode tersebut menunjukkan penggunaan Hive untuk menyimpan dan memuat data dalam bentuk objek Note dengan memanfaatkan kotak (box) Hive. Ini adalah pendekatan efisien untuk mengelola data lokal dalam aplikasi Flutter.

Model => Note data.dart

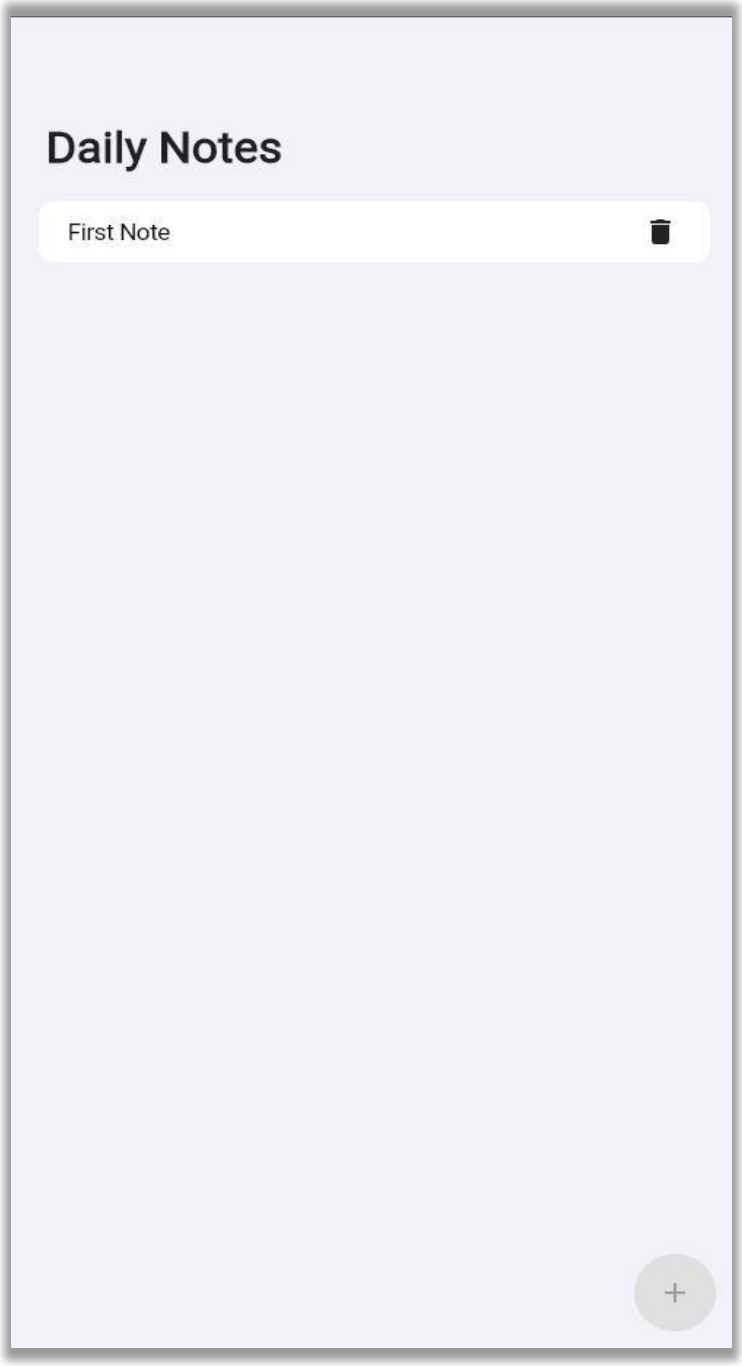
Source Code :

```dart
import 'package:flutter/material.dart';
import 'package:notestute/data/hive_database.dart';
import 'package:notestute/models/note.dart';

class NoteData extends ChangeNotifier {

  final db = HiveDatabase();

  // overall list of notes
  List<Note> allNotes = [

  ];


  void initializeNotes() {
    allNotes = db.loadNotes();
  }

  // get notes
  List<Note> getAllNotes() {
    return allNotes;
  }

  // add a new note
  void addNewNote(Note note) {
    allNotes.add(note);
    notifyListeners();
  }

  // update note
  void updateNote(Note note, String text) {
    for (int i=0; i < allNotes.length; i++) {
      if (allNotes[i].id == note.id){
        allNotes[i].text = text;
      }

    }notifyListeners();
  }

  // delete note
  void deleteNode(Note note) {
    allNotes.remove(note);
    notifyListeners();
  }
}
```

Keterangan :

Pada Kode Note.data.dart di atas meruupakan bagian dari sebuah aplikasi Flutter yang mengelola data catatan (notes) menggunakan Hive database. Ini merangkum kelas NoteData yang berperan sebagai penyedia data dan pengelola perubahan (ChangeNotifier) untuk catatan-catatan dalam aplikasi.

Output :



Pages => editing_note_page.dart

Source Code :

```dart
import 'dart:html';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter_quill/flutter_quill.dart';
import 'package:notestute/models/note_data.dart';
import 'package:provider/provider.dart';

import '../models/note.dart';

class EditingNotePage extends StatefulWidget {
  Note note;
  bool isNewNote;
  EditingNotePage({
    super.key,
    required this.note,
    required this.isNewNote,
  });

  @override
  State<EditingNotePage> createState() => _EditingNotePageState();
}

class _EditingNotePageState extends State<EditingNotePage> {
  QuillController _controller = QuillController.basic();

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

  // load existing note
  void loadExistingNote() {
    final doc = Document()..insert(0, widget.note.text);
    setState(() {
      _controller = QuillController(
          document: doc, selection: const TextSelection.collapsed(offset: 0));
    });
  }

  // add new note
  void addNewNote() {

    int id = Provider.of<NoteData>(context, listen: false).getAllNotes().length;
    // get text from editor
    String text = _controller.document.toPlainText();
    // add the new note
    Provider.of<NoteData>(context, listen: false).addNewNote(
      Note(id: id, text: text),
    );
  }

  // update existing note
  void updateNote() {
    String text = _controller.document.toPlainText();
    Provider.of<NoteData>(context, listen: false).updateNote(widget.note, text);
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: CupertinoColors.systemGroupedBackground,
      appBar: AppBar(
        elevation: 0,
        backgroundColor: Colors.transparent,
        leading: IconButton(
          icon: const Icon(Icons.arrow_back),
          onPressed: () {
            if (widget.isNewNote && !_controller.document.isEmpty()) {
              addNewNote();
            }

            else {
              updateNote();
            }

            Navigator.pop(context);
          },
          color: Colors.black,
          ),
      ),
```

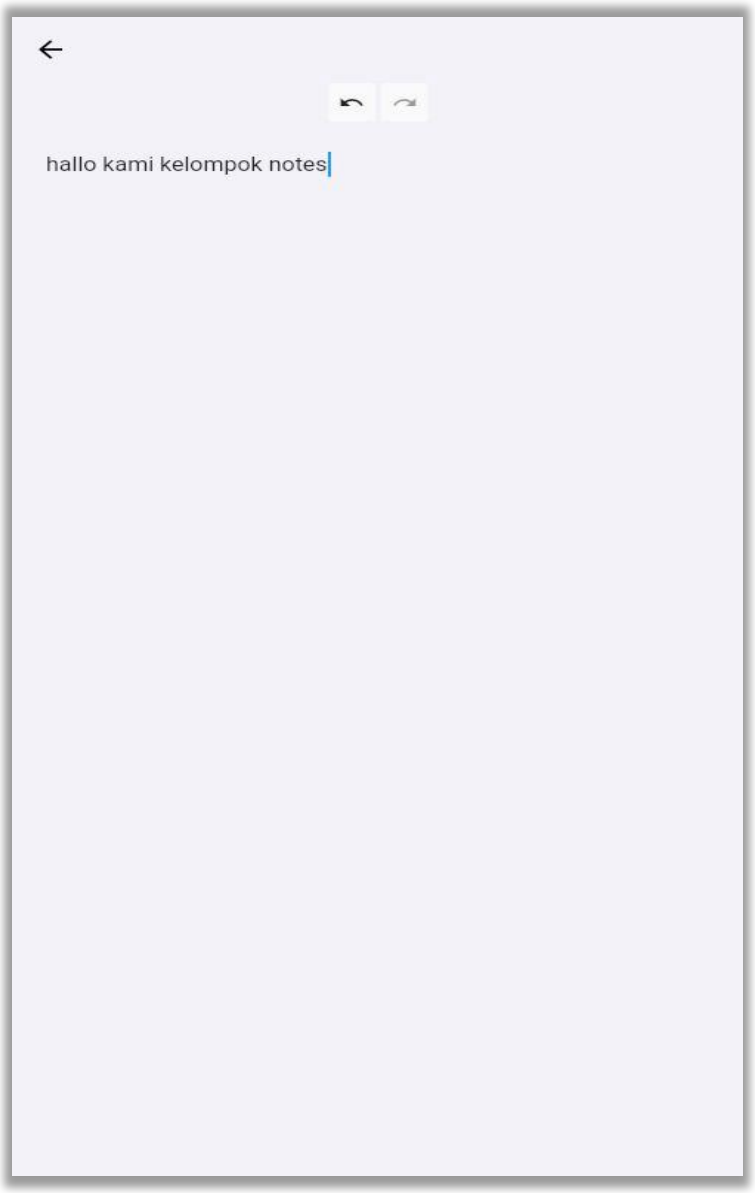
```
      body: Column(
        children: [
          // toolbar
          QuillToolbar.basic(
            controller: _controller,
            showAlignmentButtons: false,
            showBackgroundColorButton: false,
            showCenterAlignment: false,
            showCodeBlock: false,
            showColorButton: false,
            showDirection: false,
            showFontFamily: false,
            showDividers: false,
            showIndent: false,
            showHeaderStyle: false,
            showLink: false,
            showSearchButton: false,
            showInlineCode: false,
            showQuote: false,
            showListNumbers: false,
            showListBullets: false,
            showClearFormat: false,
            showFontSize: false,
            showBoldButton: false,
            showItalicButton: false,
            showUnderLineButton: false,
            showStrikeThrough: false,
            showListCheck: false,
          ),

          Expanded(
            child: Container(
              padding: const EdgeInsets.all(25),
              child: QuillEditor.basic(
                controller: _controller,
                readOnly: false,
              ),
            ),
          )
        ],
      ),
    );
  }
}

```

Keterangan :

Pada Kode di atas digunakan untuk mengedit catatan (notes) menggunakan editor teks yang disediakan oleh pustaka flutter_quill. Ini merangkum halaman EditingNotePage yang berfungsi untuk menambah atau mengedit catatan dalam aplikasi.

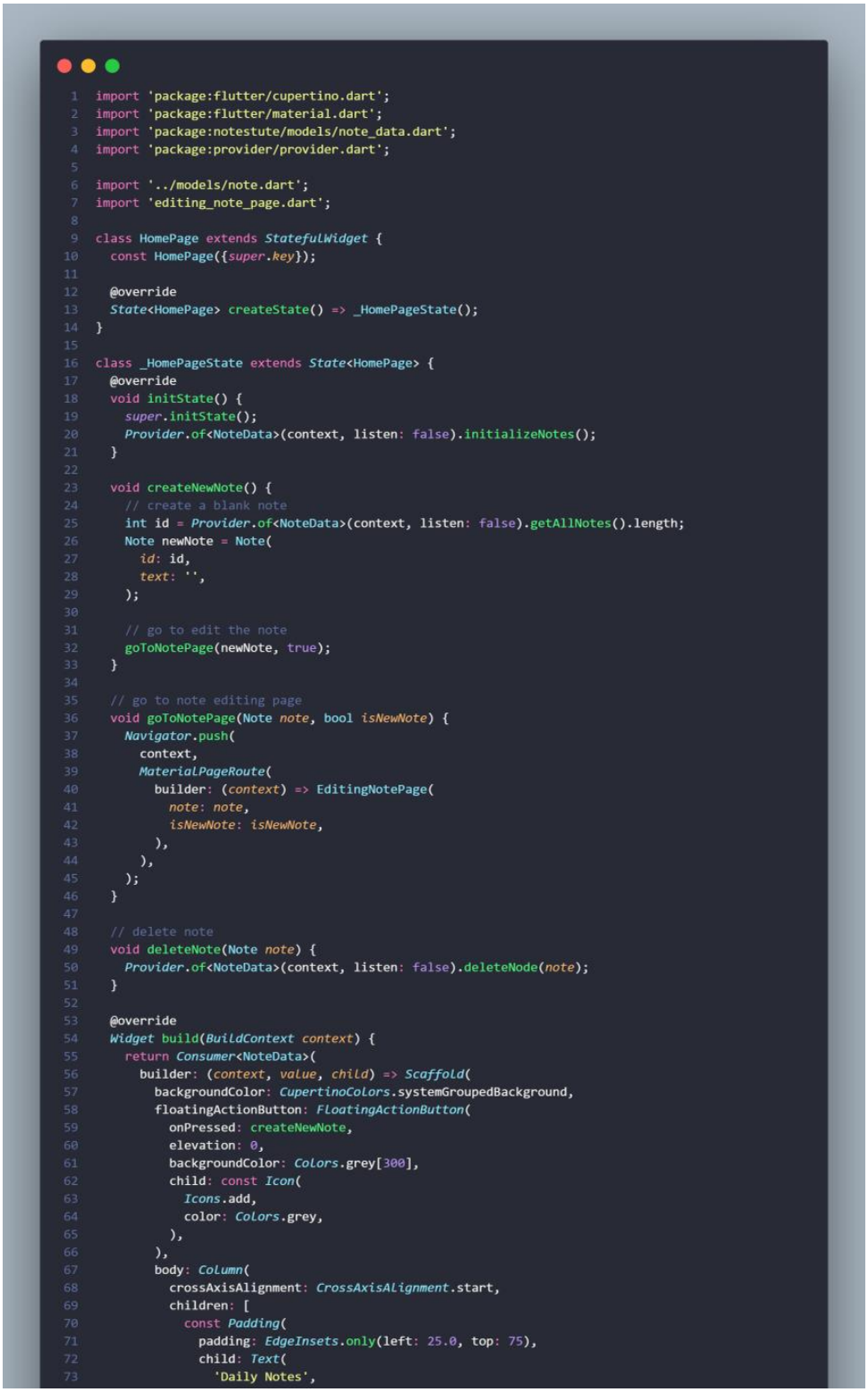
Pada dasarnya, kelas EditingNotePage ini mengatur tampilan dan logika interaksi untuk mengedit dan menyimpan catatan menggunakan editor teks Quill. Hal ini juga berinteraksi dengan NoteData melalui penyedia data (Provider) untuk menambah atau mengupdate catatan.

Output :

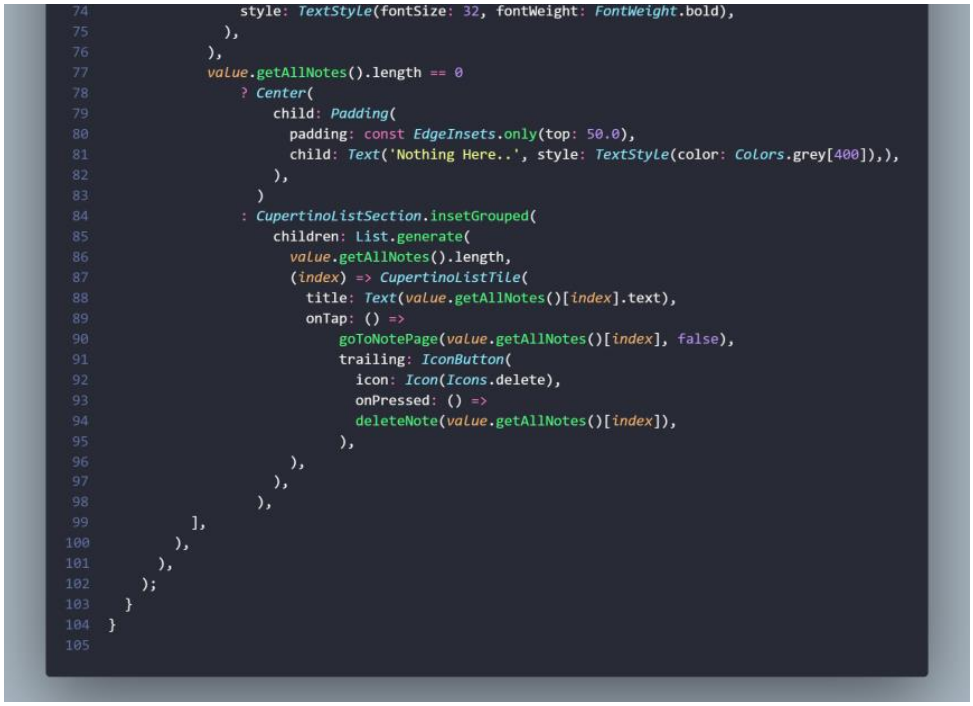


Pages => home_page.dart

Source Code :



```dart
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:notestute/models/note_data.dart';
import 'package:provider/provider.dart';

import '../models/note.dart';
import 'editing_note_page.dart';

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

  @override
  State<HomePage> createState() => _HomePageState();
}

class _HomePageState extends State<HomePage> {
  @override
  void initState() {
    super.initState();
    Provider.of<NoteData>(context, listen: false).initializeNotes();
  }

  void createNewNote() {
    // create a blank note
    int id = Provider.of<NoteData>(context, listen: false).getAllNotes().length;
    Note newNote = Note(
      id: id,
      text: '',
    );

    // go to edit the note
    goToNotePage(newNote, true);
  }

  // go to note editing page
  void goToNotePage(Note note, bool isNewNote) {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (context) => EditingNotePage(
          note: note,
          isNewNote: isNewNote,
        ),
      ),
    );
  }

  // delete note
  void deleteNote(Note note) {
    Provider.of<NoteData>(context, listen: false).deleteNode(note);
  }

  @override
  Widget build(BuildContext context) {
    return Consumer<NoteData>(
      builder: (context, value, child) => Scaffold(
        backgroundColor: CupertinoColors.systemGroupedBackground,
        floatingActionButton: FloatingActionButton(
          onPressed: createNewNote,
          elevation: 0,
          backgroundColor: Colors.grey[300],
          child: const Icon(
            Icons.add,
            color: Colors.grey,
          ),
        ),
        body: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            const Padding(
              padding: EdgeInsets.only(left: 25.0, top: 75),
              child: Text(
                'Daily Notes',
```



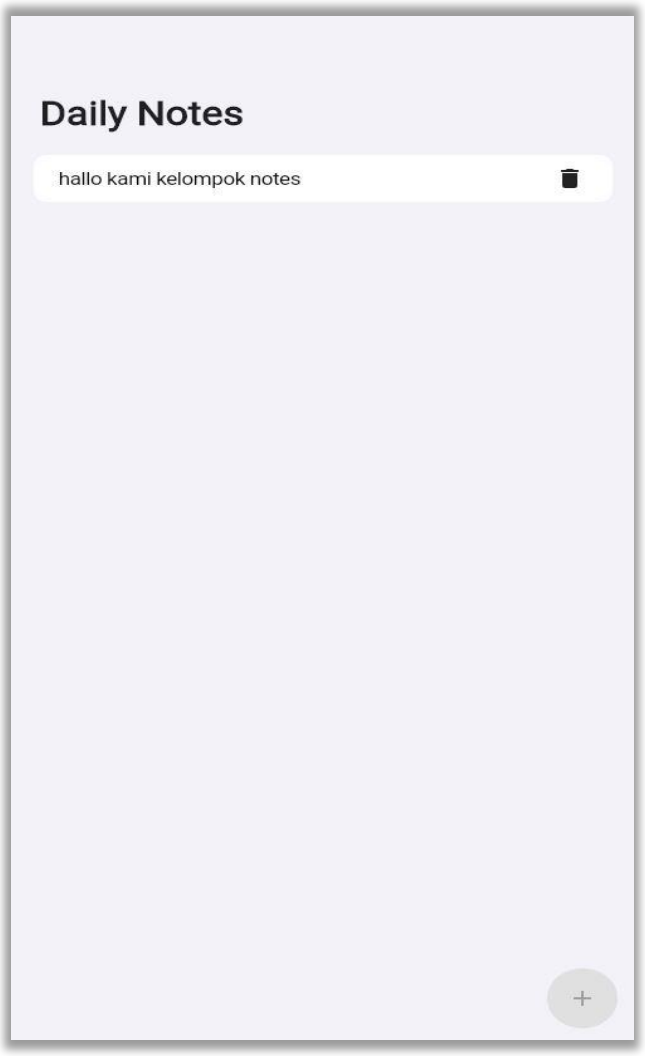
```
                style: TextStyle(fontSize: 32, fontWeight: FontWeight.bold),
              ),
            ),
            value.getAllNotes().length == 0
                ? Center(
                    child: Padding(
                      padding: const EdgeInsets.only(top: 50.0),
                      child: Text('Nothing Here..', style: TextStyle(color: Colors.grey[400]),),
                    ),
                  )
                : CupertinoListSection.insetGrouped(
                    children: List.generate(
                      value.getAllNotes().length,
                      (index) => CupertinoListTile(
                        title: Text(value.getAllNotes()[index].text),
                        onTap: () =>
                            goToNotePage(value.getAllNotes()[index], false),
                            trailing: IconButton(
                              icon: Icon(Icons.delete),
                              onPressed: () =>
                              deleteNote(value.getAllNotes()[index]),
                            ),
                      ),
                    ),
                  ),
          ],
        ),
      ),
    );
  }
}
```

Keterangan :

Kode di atas adalah bagian dari aplikasi Flutter yang menampilkan daftar catatan (notes) pada halaman utama (HomePage). Ini menggunakan penyedia data NoteData dari Provider untuk mengelola catatan-catatan dalam aplikasi dan memungkinkan pengguna menambah, mengedit, dan menghapus catatan.

Kelas HomePage ini mengatur tampilan dan interaksi pada halaman utama, serta menggunakan Consumer untuk mendengarkan perubahan data catatan dan merefresh tampilan ketika terjadi perubahan.

Output :



Pages => main.dart

Source Code :

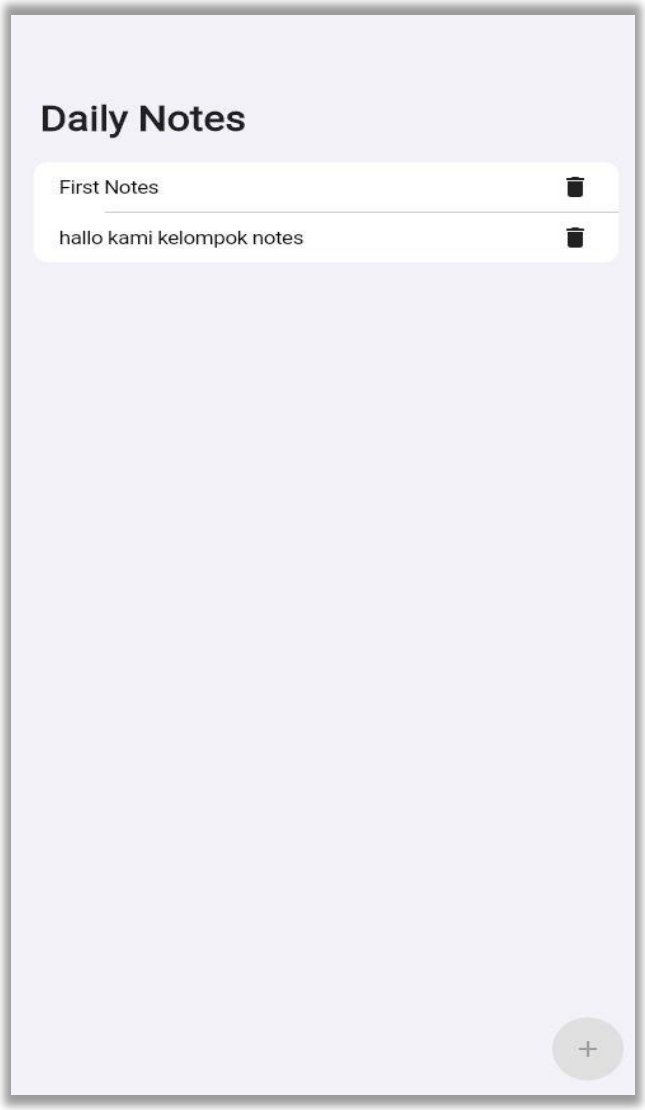
```dart
import 'package:flutter/material.dart';
import 'package:hive/hive.dart';
import 'package:hive_flutter/adapters.dart';
import 'package:notestute/models/note_data.dart';
import 'package:provider/provider.dart';
import 'pages/home_page.dart';

void main() async {
  await Hive.initFlutter();
  await Hive.openBox('note_database');
  runApp(const MyApp());
}
class MyApp extends StatelessWidget {
  const MyApp({super.key});
  @override
  Widget build(BuildContext) {
    return ChangeNotifierProvider(
      create: (context) => NoteData(),
      builder: (context, child) => const MaterialApp(
        debugShowCheckedModeBanner: false,
        home: HomePage(),
      ),
      );
  }
}
```

Keterangan :

Kode di atas menggunakan Hive database untuk mengelola catatan (notes) dan menyediakan tampilan melalui MaterialApp. Ini merangkum langkah-langkah utama dalam inisialisasi dan peluncuran aplikasi.

Dengan menggabungkan pustaka Hive, penyedia data Provider, dan tampilan menggunakan MaterialApp, kode ini menginisialisasi dan menjalankan aplikasi Flutter yang dapat mengelola catatan dan menampilkannya pada halaman utama (HomePage).

Output :



TEST=> model =>android.device_info_fake.dart

Source code :

```dart
const fakeAndroidBuildVersion = <String, dynamic>{
  'sdkInt': 16,
  'baseOS': 'baseOS',
  'previewSdkInt': 30,
  'release': 'release',
  'codename': 'codename',
  'incremental': 'incremental',
  'securityPatch': 'securityPatch',
};

const fakeDisplayMetrics = <String, dynamic>{
  'widthPx': 1080.0,
  'heightPx': 2220.0,
  'xDpi': 530.0859,
  'yDpi': 529.4639,
};

const fakeSupportedAbis = <String>['arm64-v8a', 'x86', 'x86_64'];
const fakeSupported32BitAbis = <String?>['x86 (IA-32)', 'MMX'];
const fakeSupported64BitAbis = <String?>['x86-64', 'MMX', 'SSSE3'];
const fakeSystemFeatures = ['FEATURE_AUDIO_PRO', 'FEATURE_AUDIO_OUTPUT'];

const fakeAndroidDeviceInfo = <String, dynamic>{
  'id': 'id',
  'host': 'host',
  'tags': 'tags',
  'type': 'type',
  'model': 'model',
  'board': 'board',
  'brand': 'Google',
  'device': 'device',
  'product': 'product',
  'display': 'display',
  'hardware': 'hardware',
  'isPhysicalDevice': true,
  'bootloader': 'bootloader',
  'fingerprint': 'fingerprint',
  'manufacturer': 'manufacturer',
  'supportedAbis': fakeSupportedAbis,
  'systemFeatures': fakeSystemFeatures,
  'version': fakeAndroidBuildVersion,
  'supported64BitAbis': fakeSupported64BitAbis,
  'supported32BitAbis': fakeSupported32BitAbis,
  'displayMetrics': fakeDisplayMetrics,
  'serialNumber': 'SERIAL',
};
```

Keterangan :

Kode di atas mendefinisikan sejumlah variabel konstan yang menyimpan informasi palsu (dummy) tentang perangkat Android dan lingkungan sistem. Ini biasanya digunakan untuk pengujian atau simulasi dalam pengembangan aplikasi.

TEST=> model =>android.device_info_test.dart

Source code :

```dart
// ignore_for_file: deprecated_member_use_from_same_package

import 'package:device_info_plus/src/model/android_device_info.dart';
import 'package:flutter_test/flutter_test.dart';

import 'android_device_info_fake.dart';

void main() {
  group('$AndroidDeviceInfo', () {
    group('fromMap | toMap', () {
      test('fromMap should return $AndroidDeviceInfo with correct values', () {
        final androidDeviceInfo =
            AndroidDeviceInfo.fromMap(fakeAndroidDeviceInfo);

        expect(androidDeviceInfo.id, 'id');
        expect(androidDeviceInfo.host, 'host');
        expect(androidDeviceInfo.tags, 'tags');
        expect(androidDeviceInfo.type, 'type');
        expect(androidDeviceInfo.model, 'model');
        expect(androidDeviceInfo.board, 'board');
        expect(androidDeviceInfo.brand, 'Google');
        expect(androidDeviceInfo.device, 'device');
        expect(androidDeviceInfo.product, 'product');
        expect(androidDeviceInfo.display, 'display');
        expect(androidDeviceInfo.hardware, 'hardware');
        expect(androidDeviceInfo.bootloader, 'bootloader');
        expect(androidDeviceInfo.isPhysicalDevice, isTrue);
        expect(androidDeviceInfo.fingerprint, 'fingerprint');
        expect(androidDeviceInfo.manufacturer, 'manufacturer');
        expect(androidDeviceInfo.supportedAbis, fakeSupportedAbis);
        expect(androidDeviceInfo.systemFeatures, fakeSystemFeatures);
        expect(androidDeviceInfo.supported32BitAbis, fakeSupported32BitAbis);
        expect(androidDeviceInfo.supported64BitAbis, fakeSupported64BitAbis);
        expect(androidDeviceInfo.version.sdkInt, 16);
        expect(androidDeviceInfo.version.baseOS, 'baseOS');
        expect(androidDeviceInfo.version.previewSdkInt, 30);
        expect(androidDeviceInfo.version.release, 'release');
        expect(androidDeviceInfo.version.codename, 'codename');
        expect(androidDeviceInfo.version.incremental, 'incremental');
        expect(androidDeviceInfo.version.securityPatch, 'securityPatch');
        expect(androidDeviceInfo.displayMetrics.widthPx, 1080);
        expect(androidDeviceInfo.displayMetrics.heightPx, 2220);
        expect(androidDeviceInfo.displayMetrics.xDpi, 530.0859);
        expect(androidDeviceInfo.displayMetrics.yDpi, 529.4639);
        expect(androidDeviceInfo.serialNumber, 'SERIAL');
      });

      test('toMap should return map with correct key and map', () {
        final androidDeviceInfo =
            AndroidDeviceInfo.fromMap(fakeAndroidDeviceInfo);

        expect(androidDeviceInfo.data, fakeAndroidDeviceInfo);
      });
    });
  });
}
```

Keterangan :

Kode di atas adalah bagian dari pengujian unit untuk kelas AndroidDeviceInfo dalam pustaka device_info_plus. Ini merangkum serangkaian pengujian yang memeriksa konversi dari dan ke objek AndroidDeviceInfo menggunakan data palsu (dummy).

TEST=>Model => ios_device_info_test.dart

Source Code :

```dart
// ignore_for_file: deprecated_member_use_from_same_package

import 'package:device_info_plus/device_info_plus.dart';
import 'package:flutter_test/flutter_test.dart';

void main() {
  group('$IosDeviceInfo', () {
    group('fromMap | toMap', () {
      const iosUtsnameMap = <String, dynamic>{
        'release': 'release',
        'version': 'version',
        'machine': 'machine',
        'sysname': 'sysname',
        'nodename': 'nodename',
      };
      const iosDeviceInfoMap = <String, dynamic>{
        'name': 'name',
        'model': 'model',
        'utsname': iosUtsnameMap,
        'systemName': 'systemName',
        'isPhysicalDevice': 'true',
        'systemVersion': 'systemVersion',
        'localizedModel': 'localizedModel',
        'identifierForVendor': 'identifierForVendor',
      };

      test('fromMap should return $IosDeviceInfo with correct values', () {
        final iosDeviceInfo = IosDeviceInfo.fromMap(iosDeviceInfoMap);

        expect(iosDeviceInfo.name, 'name');
        expect(iosDeviceInfo.model, 'model');
        expect(iosDeviceInfo.isPhysicalDevice, isTrue);
        expect(iosDeviceInfo.systemName, 'systemName');
        expect(iosDeviceInfo.systemVersion, 'systemVersion');
        expect(iosDeviceInfo.localizedModel, 'localizedModel');
        expect(iosDeviceInfo.utsname.release, 'release');
        expect(iosDeviceInfo.utsname.version, 'version');
        expect(iosDeviceInfo.utsname.machine, 'machine');
        expect(iosDeviceInfo.utsname.sysname, 'sysname');
        expect(iosDeviceInfo.utsname.nodename, 'nodename');
      });

      test('toMap should return map with correct key and map', () {
        final iosDeviceInfo = IosDeviceInfo.fromMap(iosDeviceInfoMap);
        expect(iosDeviceInfo.data, iosDeviceInfoMap);
      });
    });
  });
}
```

Keterangan :

Kode di atas adalah bagian dari pengujian unit untuk kelas IosDeviceInfo dalam pustaka device_info_plus. Ini merangkum serangkaian pengujian yang memeriksa konversi dari dan ke objek IosDeviceInfo menggunakan data palsu (dummy) untuk lingkungan sistem iOS.

TEST=>Model => Linux_device_info_test.dart

```dart
// ignore_for_file: deprecated_member_use_from_same_package

import 'package:device_info_plus/device_info_plus.dart';
import 'package:flutter_test/flutter_test.dart';

void main() {
  group('$LinuxDeviceInfo', () {
    test('toMap should return map with correct key and map', () {
      final linuxDeviceInfo = LinuxDeviceInfo(
        name: 'name',
        version: 'version',
        id: 'id',
        idLike: ['idLike'],
        versionCodename: 'versionCodename',
        versionId: 'versionId',
        prettyName: 'prettyName',
        buildId: 'buildId',
        variant: 'variant',
        variantId: 'variantId',
        machineId: 'machineId',
      );

      expect(linuxDeviceInfo.data, {
        'name': 'name',
        'version': 'version',
        'id': 'id',
        'idLike': ['idLike'],
        'versionCodename': 'versionCodename',
        'versionId': 'versionId',
        'prettyName': 'prettyName',
        'buildId': 'buildId',
        'variant': 'variant',
        'variantId': 'variantId',
        'machineId': 'machineId',
      });
    });
  });
}

```

Keterangan :

Kode di atas adalah bagian dari pengujian unit untuk kelas LinuxDeviceInfo dalam pustaka device_info_plus. Ini merangkum satu pengujian yang memeriksa konversi dari objek LinuxDeviceInfo ke peta (map) dengan menggunakan data yang ditentukan.

TEST=>Model => macos_device_info_test.dart

Source Code :

```dart
// ignore_for_file: deprecated_member_use_from_same_package

import 'package:device_info_plus/device_info_plus.dart';
import 'package:flutter_test/flutter_test.dart';

void main() {
  group('$MacOsDeviceInfo', () {
    group('fromMap | data', () {
      const macosDeviceInfoMap = <String, dynamic>{
        'arch': 'arch',
        'model': 'model',
        'activeCPUs': 4,
        'memorySize': 16,
        'cpuFrequency': 2,
        'hostName': 'hostName',
        'osRelease': 'osRelease',
        'majorVersion': 10,
        'minorVersion': 9,
        'patchVersion': 3,
        'computerName': 'computerName',
        'kernelVersion': 'kernelVersion',
        'systemGUID': null,
      };

      test('fromMap should return $MacOsDeviceInfo with correct values', () {
        final macosDeviceInfo = MacOsDeviceInfo.fromMap(macosDeviceInfoMap);

        expect(macosDeviceInfo.arch, 'arch');
        expect(macosDeviceInfo.model, 'model');
        expect(macosDeviceInfo.activeCPUs, 4);
        expect(macosDeviceInfo.memorySize, 16);
        expect(macosDeviceInfo.cpuFrequency, 2);
        expect(macosDeviceInfo.hostName, 'hostName');
        expect(macosDeviceInfo.osRelease, 'osRelease');
        expect(macosDeviceInfo.majorVersion, 10);
        expect(macosDeviceInfo.minorVersion, 9);
        expect(macosDeviceInfo.patchVersion, 3);
        expect(macosDeviceInfo.systemGUID, isNull);
      });

      test('toMap should return map with correct key and map', () {
        final macosDeviceInfo = MacOsDeviceInfo.fromMap(macosDeviceInfoMap);
        expect(macosDeviceInfo.data, macosDeviceInfoMap);
      });
    });
  });
}

```

Keterangan :

Kode di atas adalah bagian dari pengujian unit untuk kelas MacOsDeviceInfo dalam pustaka device_info_plus. Ini merangkum serangkaian pengujian yang memeriksa konversi dari dan ke objek MacOsDeviceInfo menggunakan data yang ditentukan.

TEST=>Model => windows_device_info_test.dart

Source Code :

```dart
// ignore_for_file: deprecated_member_use_from_same_package

import 'dart:typed_data';
import 'package:device_info_plus/device_info_plus.dart';
import 'package:flutter_test/flutter_test.dart';

void main() {
  group('$WindowsDeviceInfo', () {
    test('toMap should return map with correct key and map', () {
      final windowsDeviceInfo = WindowsDeviceInfo(
        computerName: 'computerName',
        numberOfCores: 4,
        systemMemoryInMegabytes: 16,
        userName: 'userName',
        majorVersion: 10,
        minorVersion: 0,
        buildNumber: 10240,
        platformId: 1,
        csdVersion: 'csdVersion',
        servicePackMajor: 1,
        servicePackMinor: 0,
        suitMask: 1,
        productType: 1,
        reserved: 1,
        buildLab: '22000.co_release.210604-1628',
        buildLabEx: '22000.1.amd64fre.co_release.210604-1628',
        digitalProductId: Uint8List.fromList([]),
        displayVersion: '21H2',
        editionId: 'Pro',
        installDate: DateTime(2022, 04, 02),
        productId: '00000-00000-0000-AAAAA',
        productName: 'Windows 10 Pro',
        registeredOwner: 'registeredOwner',
        releaseId: 'releaseId',
        deviceId: 'deviceId',
      );

      expect(windowsDeviceInfo.data, {
        'computerName': 'computerName',
        'numberOfCores': 4,
        'systemMemoryInMegabytes': 16,
        'userName': 'userName',
        'majorVersion': 10,
        'minorVersion': 0,
        'buildNumber': 10240,
        'platformId': 1,
        'csdVersion': 'csdVersion',
        'servicePackMajor': 1,
        'servicePackMinor': 0,
        'suitMask': 1,
        'productType': 1,
        'reserved': 1,
        'buildLab': '22000.co_release.210604-1628',
        'buildLabEx': '22000.1.amd64fre.co_release.210604-1628',
        'digitalProductId': Uint8List.fromList([]),
        'displayVersion': '21H2',
        'editionId': 'Pro',
        'installDate': DateTime(2022, 04, 02),
        'productId': '00000-00000-0000-AAAAA',
        'productName': 'Windows 10 Pro',
        'registeredOwner': 'registeredOwner',
        'releaseId': 'releaseId',
        'deviceId': 'deviceId',
      });
    });
  });
}
```

Keterangan :

Kode di atas adalah bagian dari pengujian unit untuk kelas WindowsDeviceInfo dalam pustaka device_info_plus. Ini merangkum satu pengujian yang memeriksa konversi dari objek WindowsDeviceInfo ke peta (map) dengan menggunakan data yang ditentukan.

MACOS=> Classes => Src => device_info_plus.pudspec

Source Code :

```ruby
# To learn more about a Podspec see http://guides.cocoapods.org/syntax/podspec.html
#
Pod::Spec.new do |s|
  s.name             = 'device_info_plus'
  s.version          = '0.0.1'
  s.summary          = 'No-op implementation of the macos device_info_plus to avoid build issues on macos'
  s.description      = <<-DESC
 No-op implementation of the device_info_plus plugin to avoid build issues on macos.
https://github.com/flutter/flutter/issues/46618
                       DESC
  s.homepage         = 'https://github.com/fluttercommunity/plus_plugins/tree/master/packages/device_info_plus'
  s.license          = { :file => '../LICENSE' }
  s.author           = { 'Flutter Community' => 'authors@fluttercommunity.dev' }
  s.source           = { :path => '.' }
  s.source_files = 'Classes/**/*'
  s.public_header_files = 'Classes/**/*.h'
  s.dependency 'FlutterMacOS'

  s.platform = :osx
  s.osx.deployment_target = '10.11'
end

```

```dart
import 'package:device_info_plus/src/model/web_browser_info.dart';
import 'package:flutter_test/flutter_test.dart';

void main() {
  test('WebBrowserInfo from Map with values', () {
    final info = WebBrowserInfo.fromMap(
      {
        'appCodeName': 'CODENAME',
        'appName': 'NAME',
        'appVersion': 'VERSION',
        'deviceMemory': 64,
        'language': 'en',
        'languages': ['en', 'es'],
        'platform': 'PLATFORM',
        'product': 'PRODUCT',
        'productSub': 'PRODUCTSUB',
        'userAgent':
            'Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:61.0) Gecko/20100101 Firefox/61.0',
        'vendor': 'VENDOR',
        'vendorSub': 'VENDORSUB',
        'hardwareConcurrency': 1,
        'maxTouchPoints': 2,
      },
    );

    expect(info.appName, 'NAME');
    expect(info.browserName, BrowserName.firefox);
  });

  test('WebBrowserInfo from empty map', () {
    final info = WebBrowserInfo.fromMap({});

    expect(info.appName, isNull);
    expect(info.browserName, BrowserName.unknown);
  });
}

```

Keterangan :

Kode tersebut adalah bagian dari pengujian unit untuk kelas WebBrowserInfo dalam pustaka device_info_plus. Ini merangkum dua pengujian yang memeriksa fungsionalitas kelas WebBrowserInfo dalam mengambil informasi tentang peramban web.

Pubspec.yaml

Source Code :

```yaml
name: device_info_plus
description: Flutter plugin providing detailed information about the device
  (make, model, etc.), and Android or iOS version the app is running on.
version: 8.2.2
homepage: https://plus.fluttercommunity.dev/
repository: https://github.com/fluttercommunity/plus_plugins/tree/main/packages/device_info_plus
issue_tracker: https://github.com/fluttercommunity/plus_plugins/labels/device_info_plus

flutter:
  plugin:
    platforms:
      android:
        package: dev.fluttercommunity.plus.device_info
        pluginClass: DeviceInfoPlusPlugin
      ios:
        pluginClass: FLTDeviceInfoPlusPlugin
      linux:
        dartPluginClass: DeviceInfoPlusLinuxPlugin
      web:
        pluginClass: DeviceInfoPlusWebPlugin
        fileName: src/device_info_plus_web.dart
      macos:
        pluginClass: DeviceInfoPlusMacosPlugin
      windows:
        dartPluginClass: DeviceInfoPlusWindowsPlugin

dependencies:
  device_info_plus_platform_interface: ^7.0.0
  ffi: ^2.0.1
  file: ^6.0.0
  flutter:
    sdk: flutter
  flutter_web_plugins:
    sdk: flutter
  meta: ^1.3.0
  win32: ">=2.7.0 <5.0.0"

dev_dependencies:
  flutter_lints: ^2.0.1
  flutter_test:
    sdk: flutter
  mockito: ^5.0.0
  test: ^1.16.4

environment:
  sdk: ">=2.12.0 <3.0.0"
  flutter: ">=2.11.0"

```

SEKIAN DAN TERIMAKASIH