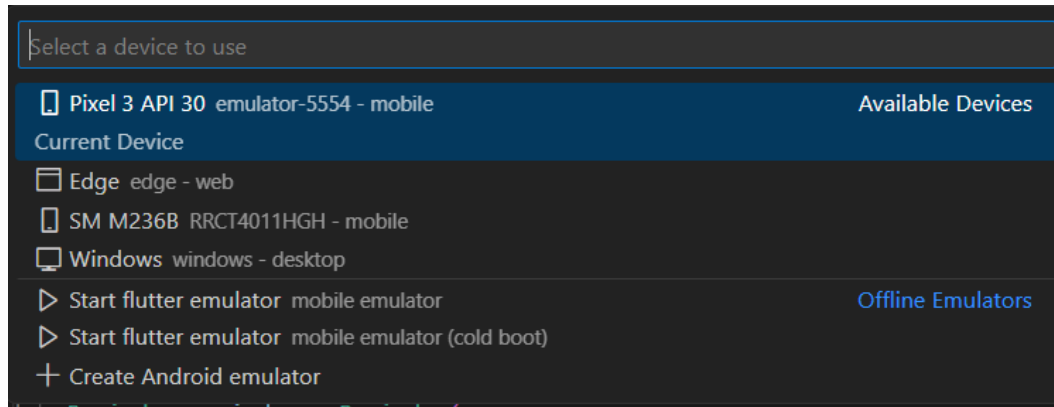Nama : Annisa Divayu Andriyani

NIM : 20200140136

MK : Software Testing & Quality-Assurance
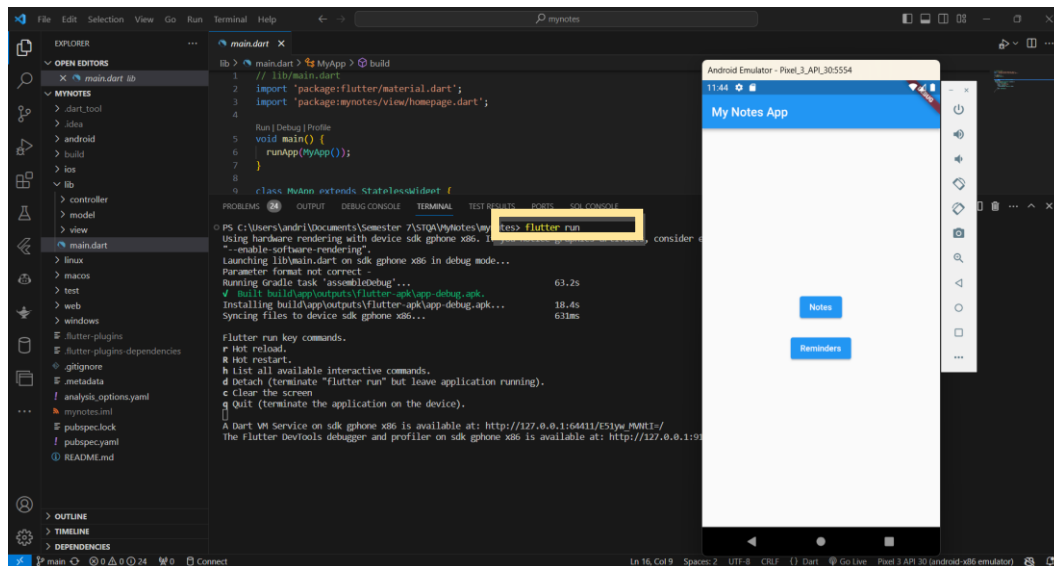
UNIT TESTING APLIKASI MY NOTES

Link GitHub : https://github.com/eunannana/mynotes.git

- Untuk melakukan debugging dan testing, run terlebih dahulu aplikasi dengan memilih emulator
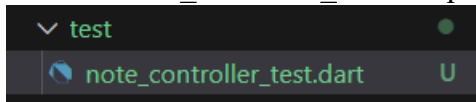

- Buka terminal lalu masukkan "flutter run" dan tunggu hingga aplikasi berjalan dengan baik

- Untuk melakukan debugging dan testing hentikan aplikasi terlebih dahulu dan mulai debugging dan testing

- Debuging dan flutter test note_controller
  1. Buat file note_controller_test.dart pada folder test
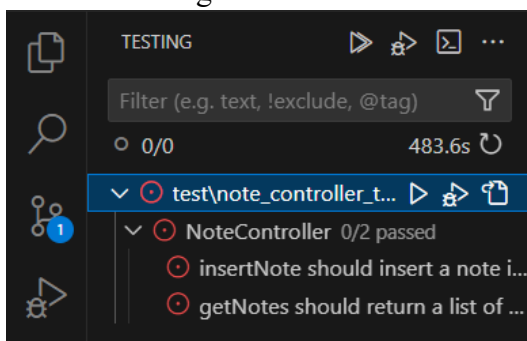

  2. Tulis code berikut

```dart
import 'package:flutter_test/flutter_test.dart';
import 'package:mynotes/controller/note_controller.dart';
import 'package:mynotes/model/note_model.dart';

void main() {
  group('NoteController', () {
    test('insertNote should insert a note into the database', () async {
      // Arrange
      final noteController = NoteController();
      final note = Note(id: 1, title: 'Test Note', content: 'Test Content');
      // Act
      final result = await noteController.insertNote(note);
      // Assert
      expect(result, 1);
    });
    test('getNotes should return a list of notes from the database', () async {
      // Arrange
      final noteController = NoteController();

      // Act
      final notes = await noteController.getNotes();

      // Assert
      expect(notes, isList);
    });
  });
}
```

  3. Untuk run code testing, pilih menu Testing pada side menu

  4. Klik icon debug untuk memulai test code

5. Pada terminal baru, ketik "flutter test" untuk memastikan flutter test berjalan dengan baik

- Debuging dan flutter reminder_controller test
  1. Buat file reminder_controller_test.dart pada folder test
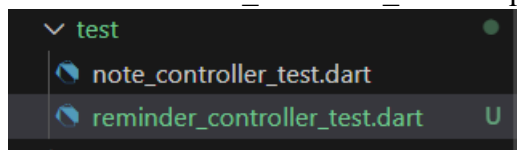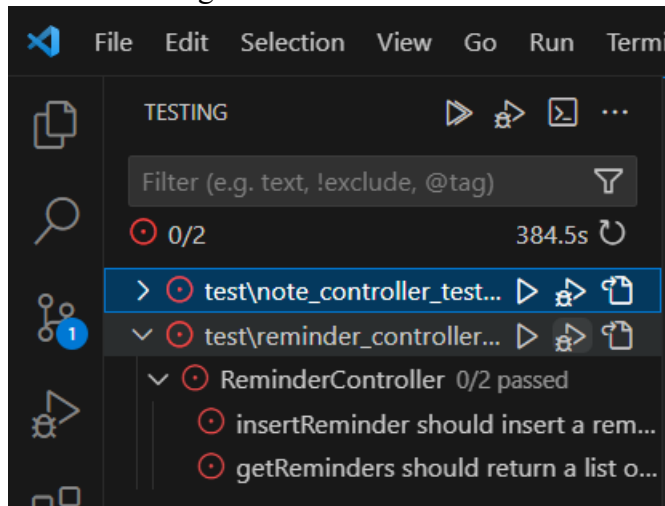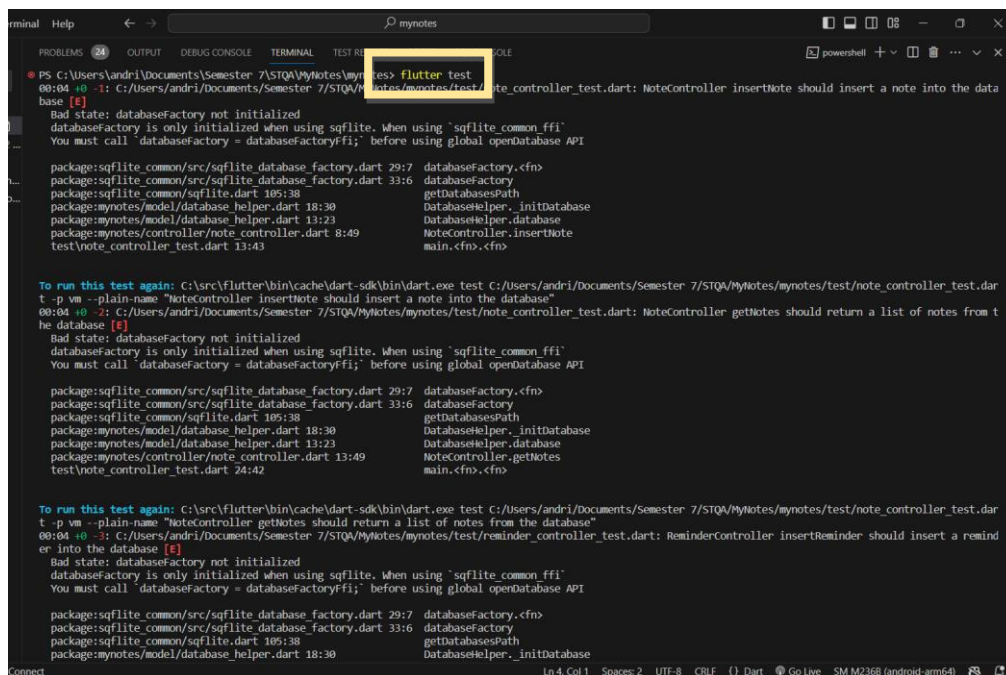

  2. Tulis code berikut

```dart
import 'package:flutter_test/flutter_test.dart';
import 'package:mynotes/controller/reminder_controller.dart';
import 'package:mynotes/model/reminder_model.dart';

void main() {
  group('ReminderController', () {
    test('insertReminder should insert a reminder into the database', () async {
      // Arrange
      final reminderController = ReminderController();
      final reminder = Reminder(id: 1, title: 'Test Reminder', dateTime: '2023-11-16 10:00:00');

      // Act
      final result = await reminderController.insertReminder(reminder);

      // Assert
      expect(result, 1); // Assuming 1 is the expected ID for the inserted reminder
    });
    test('getReminders should return a list of reminders from the database', () async {
      // Arrange
      final reminderController = ReminderController();

      // Act
      final reminders = await reminderController.getReminders();

      // Assert
      expect(reminders, isList); // Ensure it returns a list
    });
  });
}
```

3. Untuk run code testing, pilih menu Testing pada side menu

4. Klik icon debug untuk memulai test code

5. Pada terminal baru, ketik "flutter test" untuk memastikan flutter test berjalan dengan baik

- Debuging dan flutter database helper test
  1. Buat file database_helper_test.dart pada folder test
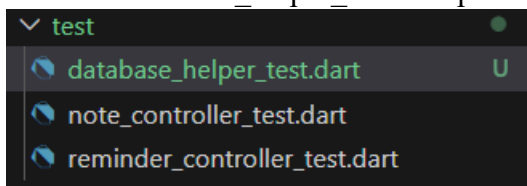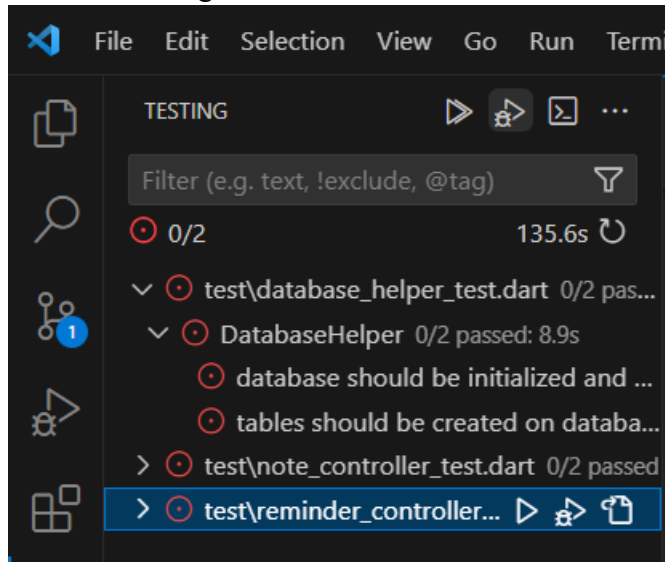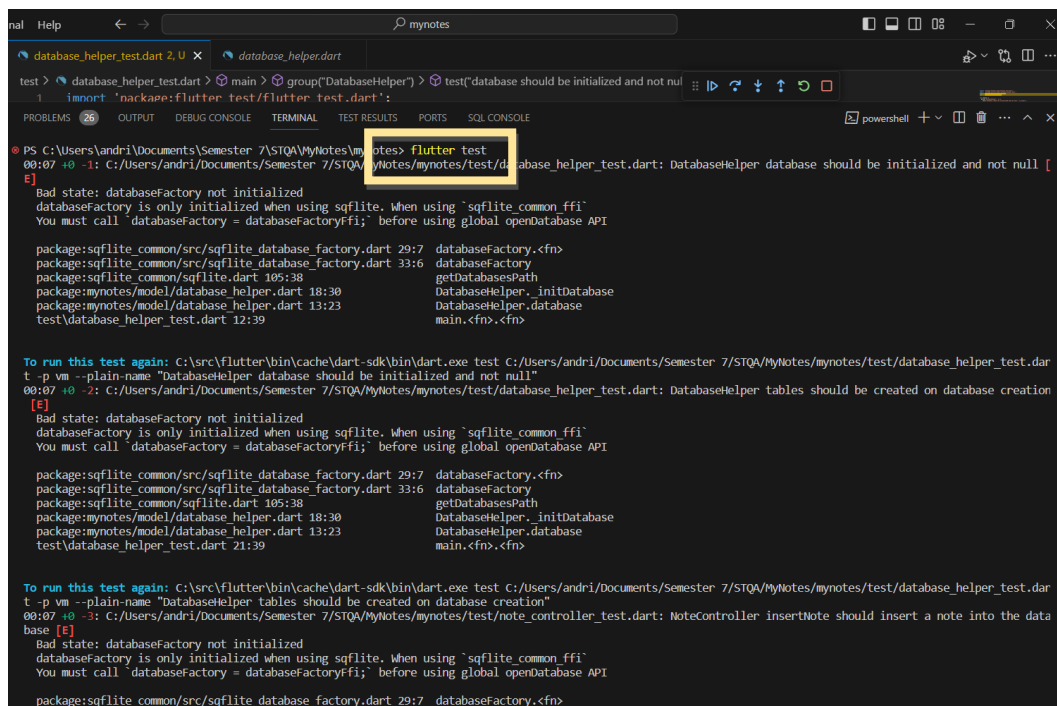

  2. Tulis code berikut


```dart
import 'package:flutter_test/flutter_test.dart';
import 'package:mynotes/model/database_helper.dart';
import 'package:sqflite/sqflite.dart';
import 'package:path/path.dart';

void main() {
  group('DatabaseHelper', () {
    test('database should be initialized and not null', () async {
      // Arrange
      final dbHelper = DatabaseHelper.instance;
      // Act
      final database = await dbHelper.database;
      // Assert
      expect(database, isNotNull);
    });

    test('tables should be created on database creation', () async {
      // Arrange
      final dbHelper = DatabaseHelper.instance;
      // Act
      final database = await dbHelper.database;
      // Get the list of tables
      final tables = await database.query('sqlite_master', where: 'type = ?', whereArgs: ['table']);
      // Assert
      expect(tables, isList);
      expect(tables.length, 2); // Assuming there are two tables: notes and reminders
    });
  });
}
```

3. Untuk run code testing, pilih menu Testing pada side menu

4. Klik icon debug untuk memulai test code

5. Pada terminal baru, ketik "flutter test" untuk memastikan flutter test berjalan dengan baik

- Debuging dan flutter homepage test
  1. Buat file homepage_test.dart pada folder test
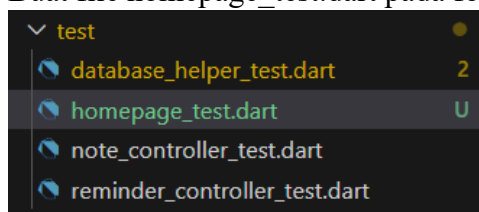
  2. Tulis code berikut


```dart
import 'package:flutter/material.dart';
import 'package:flutter_test/flutter_test.dart';
import 'package:mynotes/view/homepage.dart';
import 'package:mynotes/view/note_page.dart';
import 'package:mynotes/view/reminder_page.dart';

void main() {
  testWidgets('HomePage should contain two ElevatedButtons', (WidgetTester tester) async {
    // Build app and trigger a frame.
    await tester.pumpWidget(MaterialApp(
      home: HomePage(),
    )); // MaterialApp

    // Verify that the page contains two ElevatedButtons
    expect(find.byType(ElevatedButton), findsNWidgets(2));
  });

  testWidgets('Tapping Notes button should navigate to NotePage', (WidgetTester tester) async {
    // Build app and trigger a frame.
    await tester.pumpWidget(MaterialApp(
      home: HomePage(),
    )); // MaterialApp

    // Tap the 'Notes' button
    await tester.tap(find.widgetWithText(ElevatedButton, 'Notes'));

    // Wait for the Navigator to finish pushing a new route
    await tester.pumpAndSettle();

    // Verify that the NotePage is on the screen
    expect(find.byType(NotePage), findsOneWidget);
  });
```
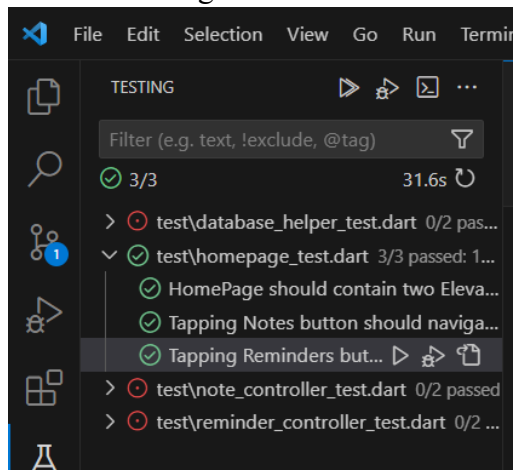
```
homepage_test.dart U ✕
st > ◆ homepage_test.dart > ...
      Run | Debug
18    testWidgets('Tapping Notes button should navigate to NotePage', (WidgetTester tester) async {
19      // Build app and trigger a frame.
20      await tester.pumpWidget(MaterialApp(
21        └─home: HomePage(),
22      )); // MaterialApp
23
24      // Tap the 'Notes' button
25      await tester.tap(find.widgetWithText(ElevatedButton, 'Notes'));
26
27      // Wait for the Navigator to finish pushing a new route
28      await tester.pumpAndSettle();
29
30      // Verify that the NotePage is on the screen
31      expect(find.byType(NotePage), findsOneWidget);
32    });
33
      Run | Debug
34    testWidgets('Tapping Reminders button should navigate to ReminderPage', (WidgetTester tester) async {
35      // Build app and trigger a frame.
36      await tester.pumpWidget(MaterialApp(
37        └─home: HomePage(),
38      )); // MaterialApp
39
40      // Tap the 'Reminders' button
41      await tester.tap(find.widgetWithText(ElevatedButton, 'Reminders'));
42
43      // Wait for the Navigator to finish pushing a new route
44      await tester.pumpAndSettle();
45
46      // Verify that the ReminderPage is on the screen
47      expect(find.byType(ReminderPage), findsOneWidget);
48    });
49  }
50
```

3. Untuk run code testing, pilih menu Testing pada side menu

4. Klik icon debug untuk memulai test code

5. Pada terminal baru, ketik "flutter test" untuk memastikan flutter test berjalan dengan baik

- Debuging dan flutter database helper test
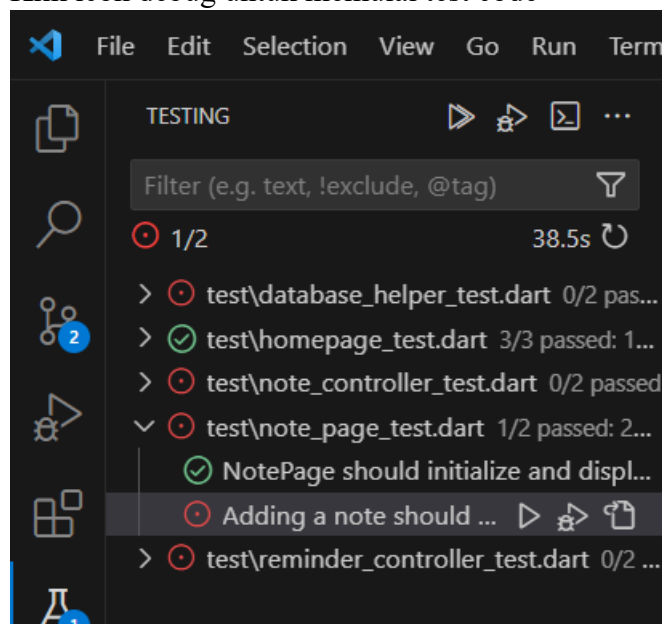 6. Buat file database_helper_test.dart pada folder test

7. Tulis code berikut

```dart
// test/note_page_test.dart
import 'package:flutter/material.dart';
import 'package:flutter_test/flutter_test.dart';
import 'package:mynotes/controller/note_controller.dart';
import 'package:mynotes/model/note_model.dart';
import 'package:mynotes/view/note_page.dart';

void main() {
  testWidgets('NotePage should initialize and display widgets', (WidgetTester tester) async {
    // Build app and trigger a frame.
    await tester.pumpWidget(MaterialApp(
      home: NotePage(),
    )); // MaterialApp
    // Verify that the title and content TextFields are displayed.
    expect(find.byType(TextField), findsNWidgets(2));
    // Verify that the Add Note button is displayed.
    expect(find.byType(ElevatedButton), findsOneWidget);
  });

  testWidgets('Adding a note should update the note list', (WidgetTester tester) async {
    // Build app and trigger a frame.
    await tester.pumpWidget(MaterialApp(
      home: NotePage(),
    )); // MaterialApp
    // Enter title and content in TextFields
    await tester.enterText(find.widgetWithText(TextField, 'Title'), 'Test Title');
    await tester.enterText(find.widgetWithText(TextField, 'Content'), 'Test Content');
    // Tap the Add Note button
    await tester.tap(find.widgetWithText(ElevatedButton, 'Add Note'));
    // Wait for the async operation to complete
    await tester.pump();
    // Verify that the note list is updated
    expect(find.text('Test Title'), findsOneWidget);
    expect(find.text('Test Content'), findsOneWidget);
  });
}
```

8. Untuk run code testing, pilih menu Testing pada side menu

9. Klik icon debug untuk memulai test code

10. Pada terminal baru, ketik "flutter test" untuk memastikan flutter test berjalan dengan baik

- Debuging dan flutter reminder page test
  1. Buat file reminder_page_test.dart pada folder test

2. Tulis code berikut

```dart
import 'package:flutter/material.dart';
import 'package:flutter_test/flutter_test.dart';
import 'package:mynotes/controller/reminder_controller.dart';
import 'package:mynotes/model/reminder_model.dart';
import 'package:mynotes/view/reminder_page.dart';
void main() {
  testWidgets('ReminderPage should initialize and display widgets', (WidgetTester tester) async {
    // Build app and trigger a frame.
    await tester.pumpWidget(MaterialApp(
      home: ReminderPage(),
    )); // MaterialApp
    // Verify that the title and dateTime TextFields are displayed.
    expect(find.byType(TextField), findsNWidgets(2));
    // Verify that the Add Reminder button is displayed.
    expect(find.byType(ElevatedButton), findsOneWidget);
  });
  testWidgets('Adding a reminder should update the reminder list', (WidgetTester tester) async {
    // Build app and trigger a frame.
    await tester.pumpWidget(MaterialApp(
      home: ReminderPage(),
    )); // MaterialApp
    // Enter title and dateTime in TextFields
    await tester.enterText(find.widgetWithText(TextField, 'Title'), 'Test Title');
    await tester.enterText(find.widgetWithText(TextField, 'Date and Time'), '2023-11-16 10:00:00');
    // Tap the Add Reminder button
    await tester.tap(find.widgetWithText(ElevatedButton, 'Add Reminder'));
    // Wait for the async operation to complete
    await tester.pump();
    // Verify that the reminder list is updated
    expect(find.text('Test Title'), findsOneWidget);
    expect(find.text('2023-11-16 10:00:00'), findsOneWidget);
  });
}
```

3. Untuk run code testing, pilih menu Testing pada side menu

4. Klik icon debug untuk memulai test code

5. Pada terminal baru, ketik "flutter test" untuk memastikan flutter test berjalan dengan baik