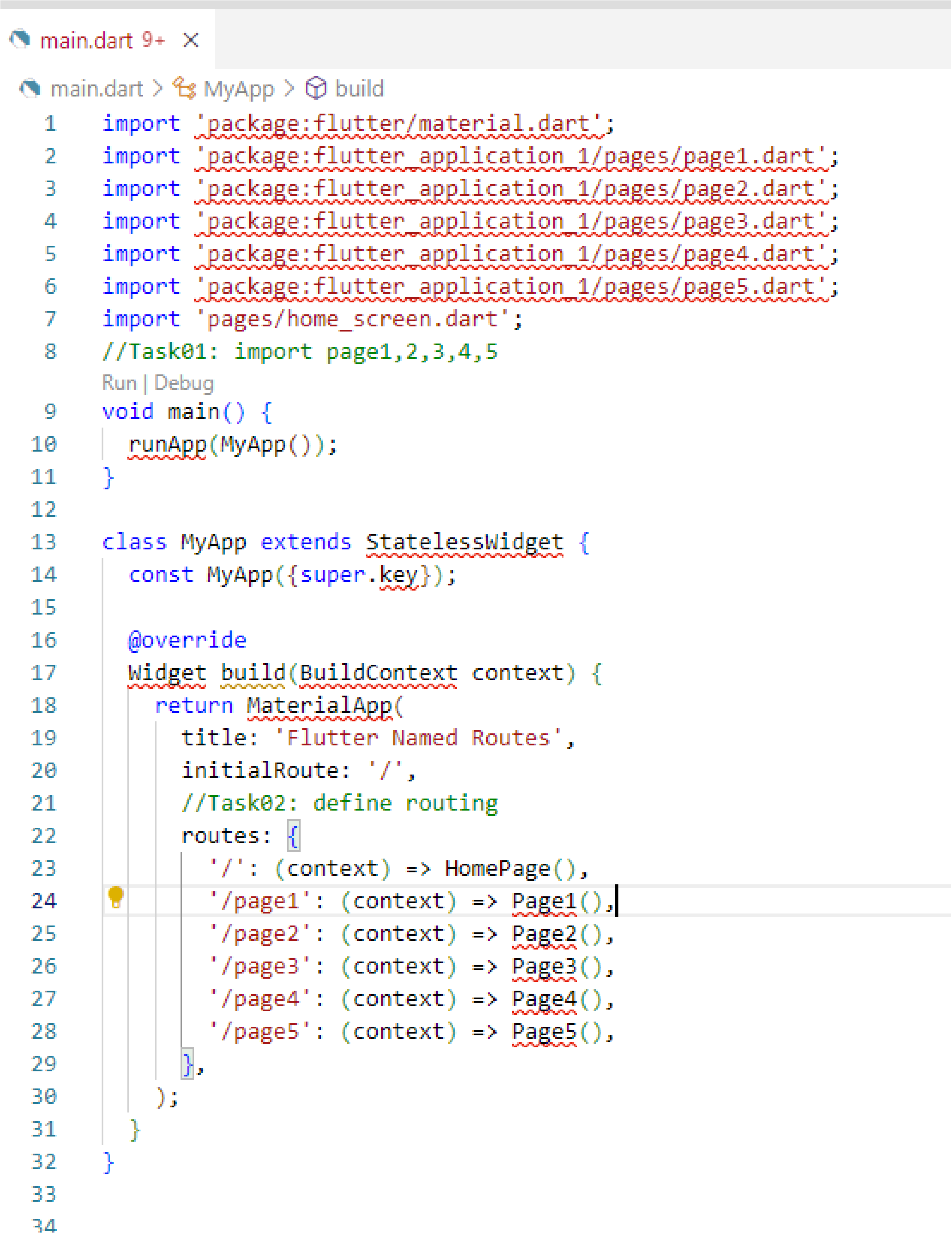
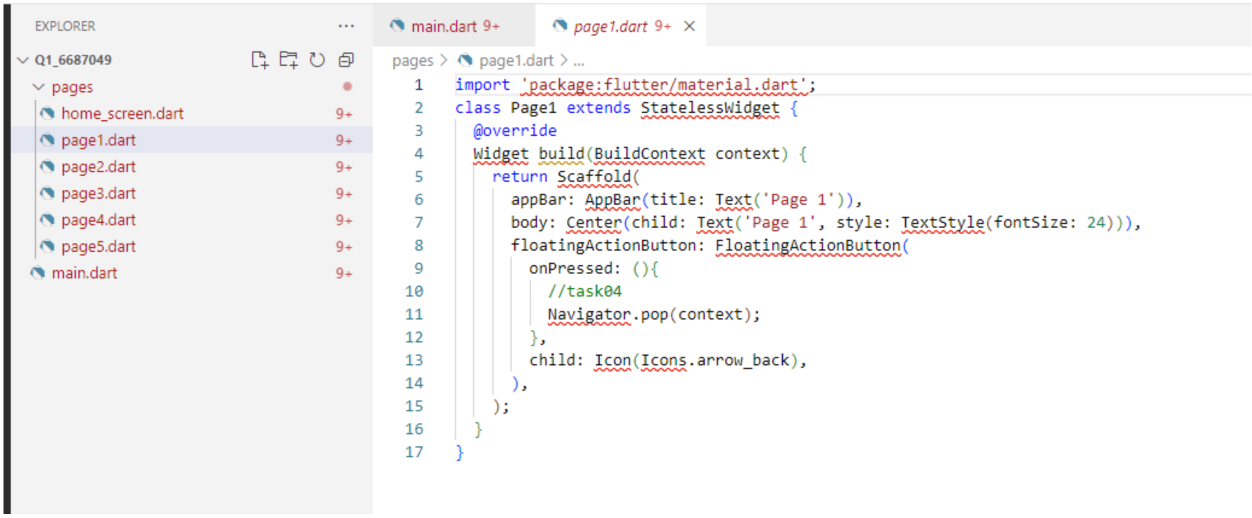
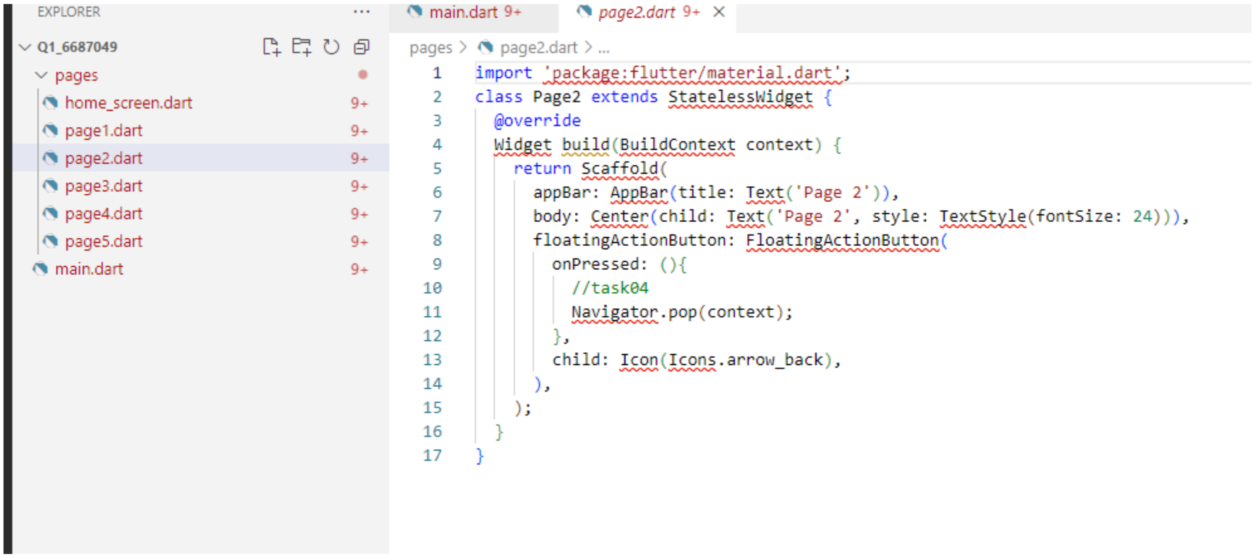
L9 -Q1



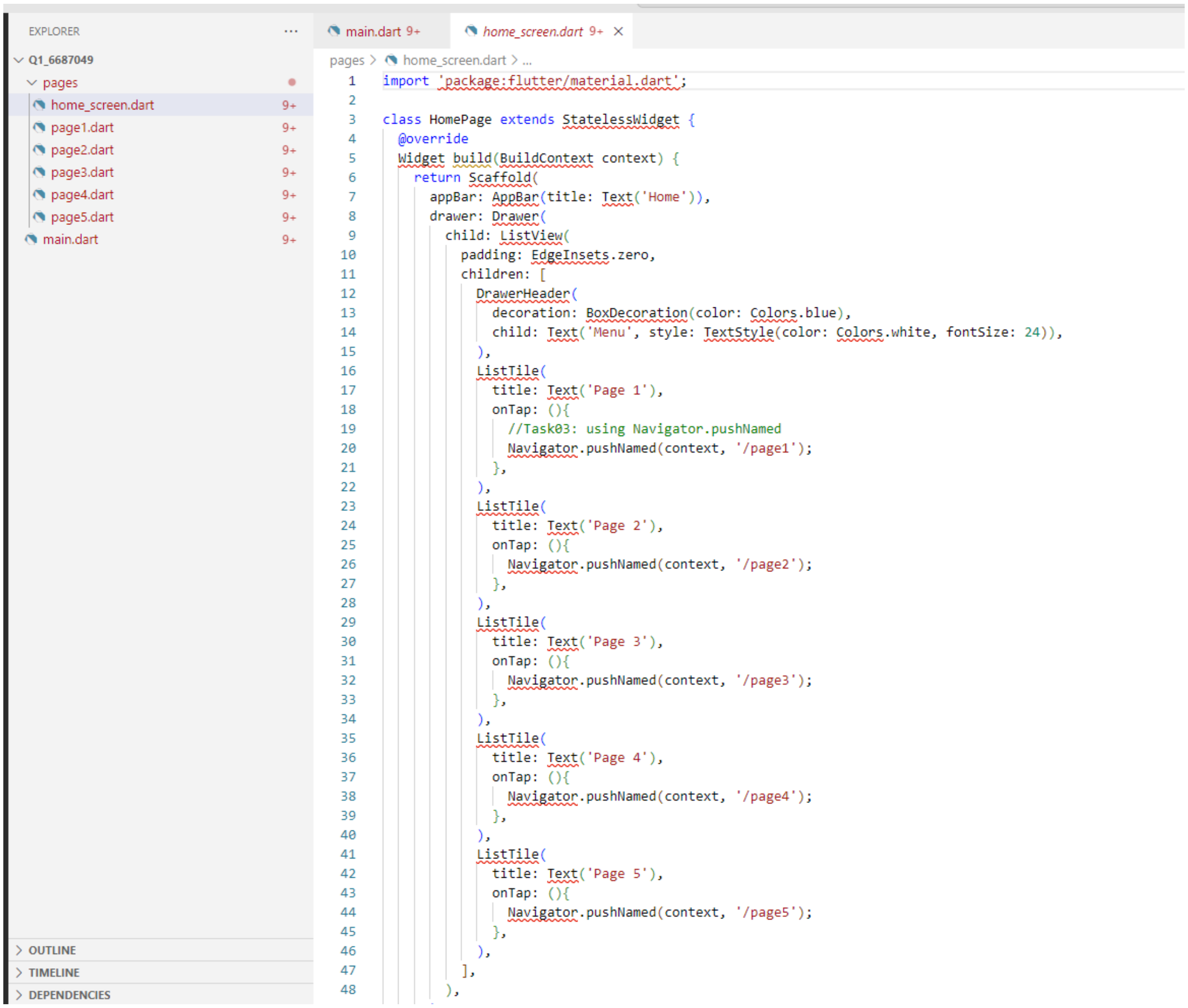
```dart
import 'package:flutter/material.dart';
import 'package:flutter_application_1/pages/page1.dart';
import 'package:flutter_application_1/pages/page2.dart';
import 'package:flutter_application_1/pages/page3.dart';
import 'package:flutter_application_1/pages/page4.dart';
import 'package:flutter_application_1/pages/page5.dart';
import 'pages/home_screen.dart';
//Task01: import page1,2,3,4,5
void main() {
  runApp(MyApp());
}

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

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Named Routes',
      initialRoute: '/',
      //Task02: define routing
      routes: {
        '/': (context) => HomePage(),
        '/page1': (context) => Page1(),
        '/page2': (context) => Page2(),
        '/page3': (context) => Page3(),
        '/page4': (context) => Page4(),
        '/page5': (context) => Page5(),
      },
    );
  }
}
```



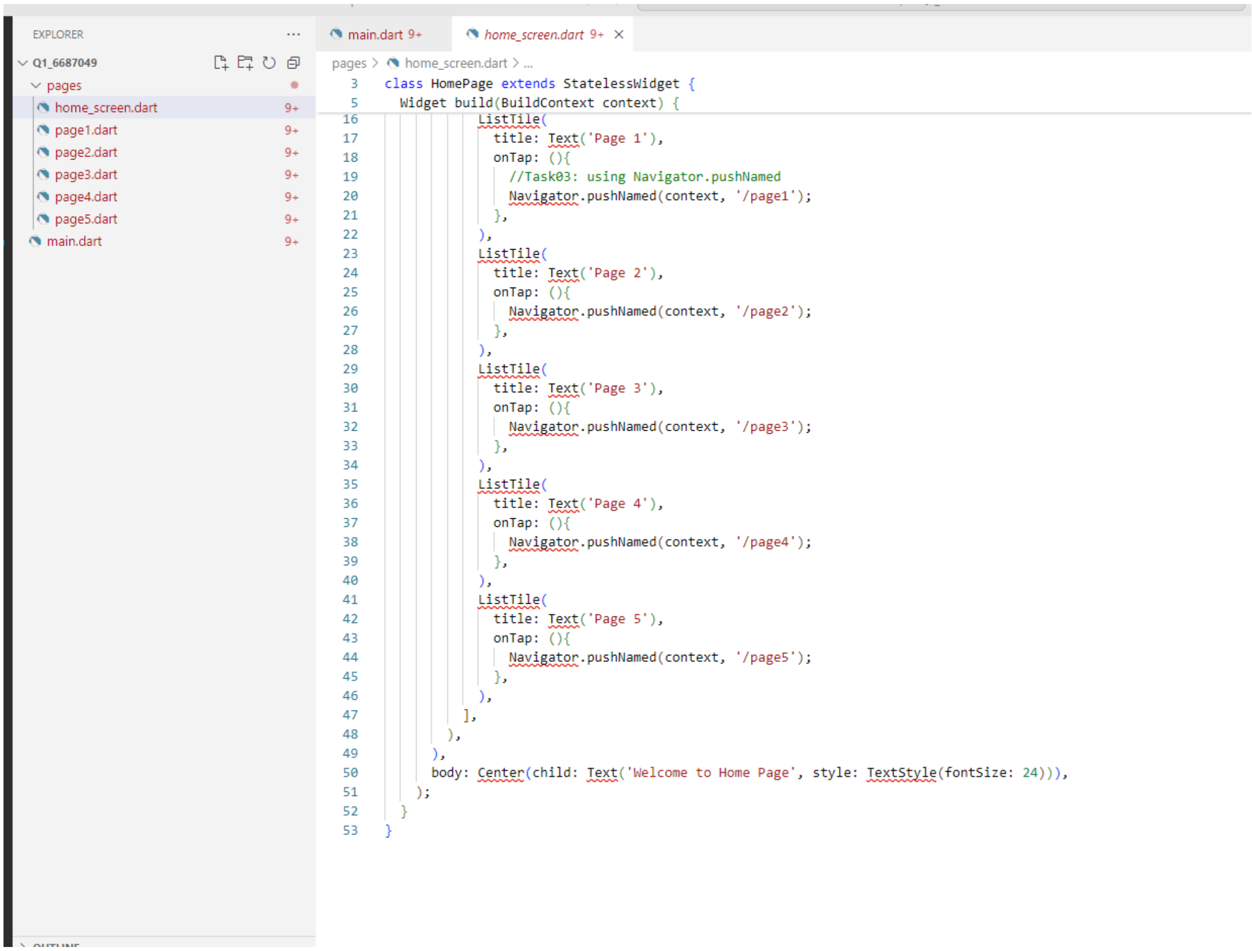
```dart
import 'package:flutter/material.dart';
class Page1 extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text('Page 1')),
      body: Center(child: Text('Page 1', style: TextStyle(fontSize: 24))),
      floatingActionButton: FloatingActionButton(
        onPressed: (){
          //task04
          Navigator.pop(context);
        },
        child: Icon(Icons.arrow_back),
      ),
    );
  }
}
```



```dart
import 'package:flutter/material.dart';
class Page2 extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text('Page 2')),
      body: Center(child: Text('Page 2', style: TextStyle(fontSize: 24))),
      floatingActionButton: FloatingActionButton(
        onPressed: (){
          //task04
          Navigator.pop(context);
        },
        child: Icon(Icons.arrow_back),
      ),
    );
  }
}
```

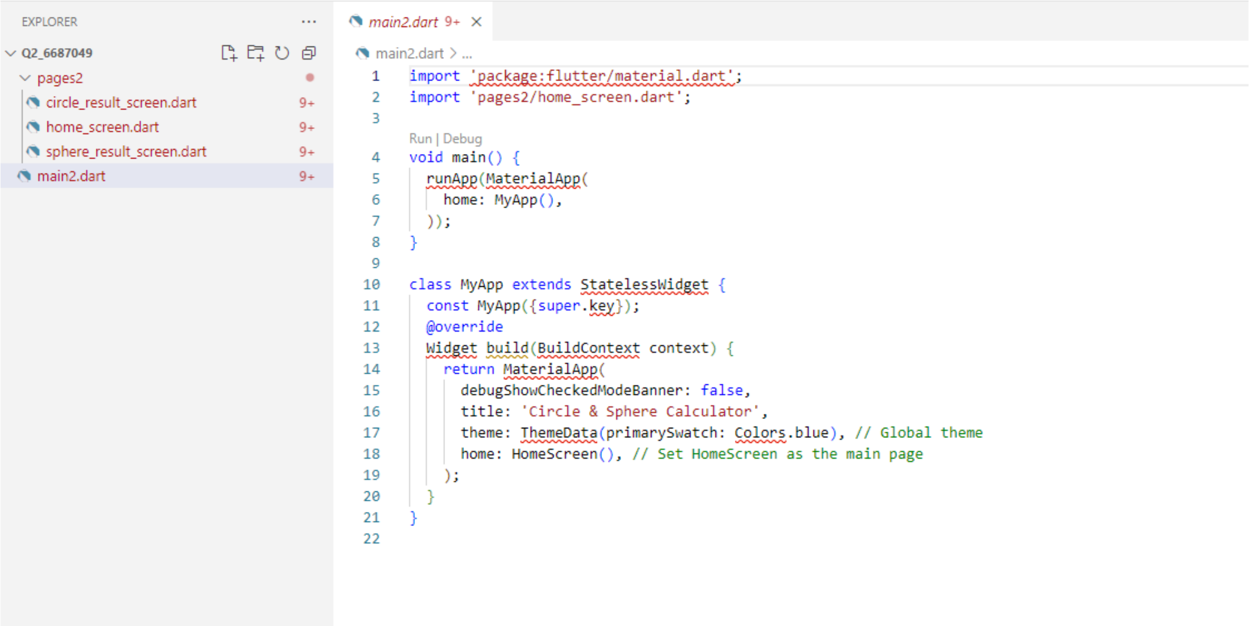


```dart
import 'package:flutter/material.dart';

class HomePage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text('Home')),
      drawer: Drawer(
        child: ListView(
          padding: EdgeInsets.zero,
          children: [
            DrawerHeader(
              decoration: BoxDecoration(color: Colors.blue),
              child: Text('Menu', style: TextStyle(color: Colors.white, fontSize: 24)),
            ),
            ListTile(
              title: Text('Page 1'),
              onTap: (){
                //Task03: using Navigator.pushNamed
                Navigator.pushNamed(context, '/page1');
              },
            ),
            ListTile(
              title: Text('Page 2'),
              onTap: (){
                Navigator.pushNamed(context, '/page2');
              },
            ),
            ListTile(
              title: Text('Page 3'),
              onTap: (){
                Navigator.pushNamed(context, '/page3');
              },
            ),
            ListTile(
              title: Text('Page 4'),
              onTap: (){
                Navigator.pushNamed(context, '/page4');
              },
            ),
            ListTile(
              title: Text('Page 5'),
              onTap: (){
                Navigator.pushNamed(context, '/page5');
              },
            ),
          ],
        ),
```

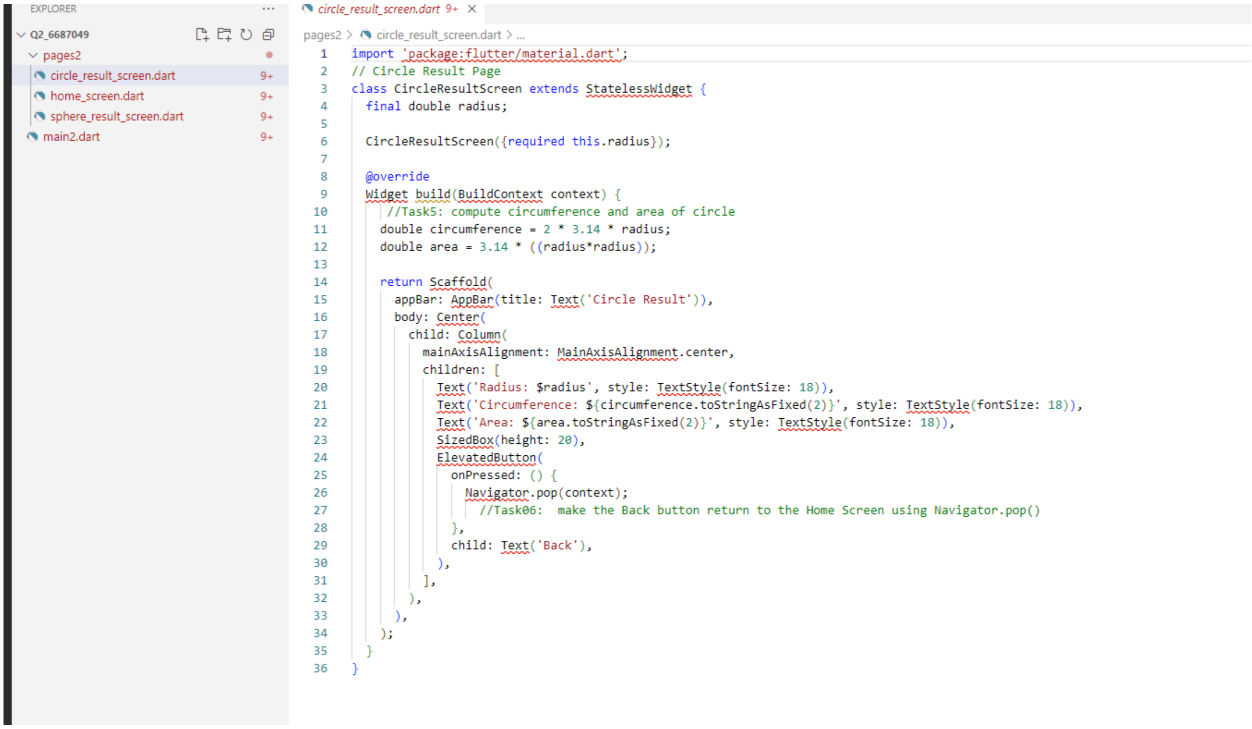


```
class HomePage extends StatelessWidget {
  Widget build(BuildContext context) {
            ListTile(
              title: Text('Page 1'),
              onTap: (){
                //Task03: using Navigator.pushNamed
                Navigator.pushNamed(context, '/page1');
              },
            ),
            ListTile(
              title: Text('Page 2'),
              onTap: (){
                Navigator.pushNamed(context, '/page2');
              },
            ),
            ListTile(
              title: Text('Page 3'),
              onTap: (){
                Navigator.pushNamed(context, '/page3');
              },
            ),
            ListTile(
              title: Text('Page 4'),
              onTap: (){
                Navigator.pushNamed(context, '/page4');
              },
            ),
            ListTile(
              title: Text('Page 5'),
              onTap: (){
                Navigator.pushNamed(context, '/page5');
              },
            ),
          ],
        ),
      ),
      body: Center(child: Text('Welcome to Home Page', style: TextStyle(fontSize: 24))),
    );
  }
}
```

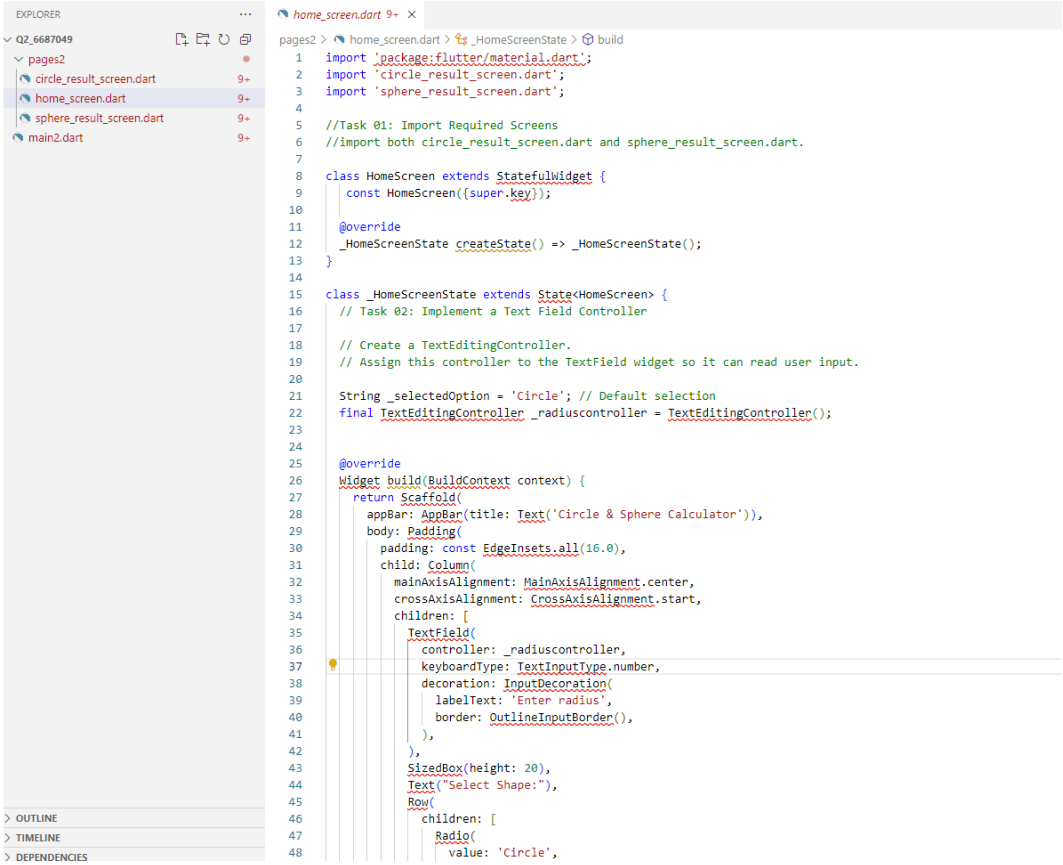
L9-Q2



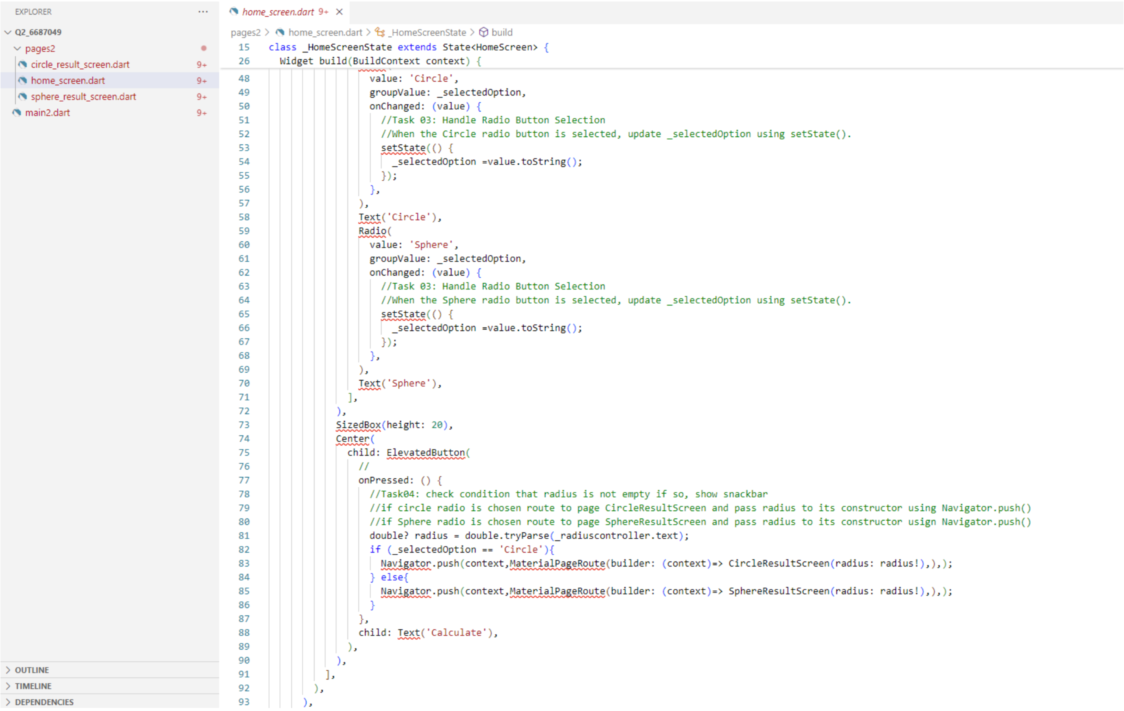
```dart
import 'package:flutter/material.dart';
import 'pages2/home_screen.dart';

void main() {
  runApp(MaterialApp(
    home: MyApp(),
  ));
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Circle & Sphere Calculator',
      theme: ThemeData(primarySwatch: Colors.blue), // Global theme
      home: HomeScreen(), // Set HomeScreen as the main page
    );
  }
}

```


EXPLORER
Q2_6687049
pages2
circle_result_screen.dart 9+
home_screen.dart 9+
sphere_result_screen.dart 9+
main2.dart 9+
circle_result_screen.dart 9+
pages2 > circle_result_screen.dart > ...

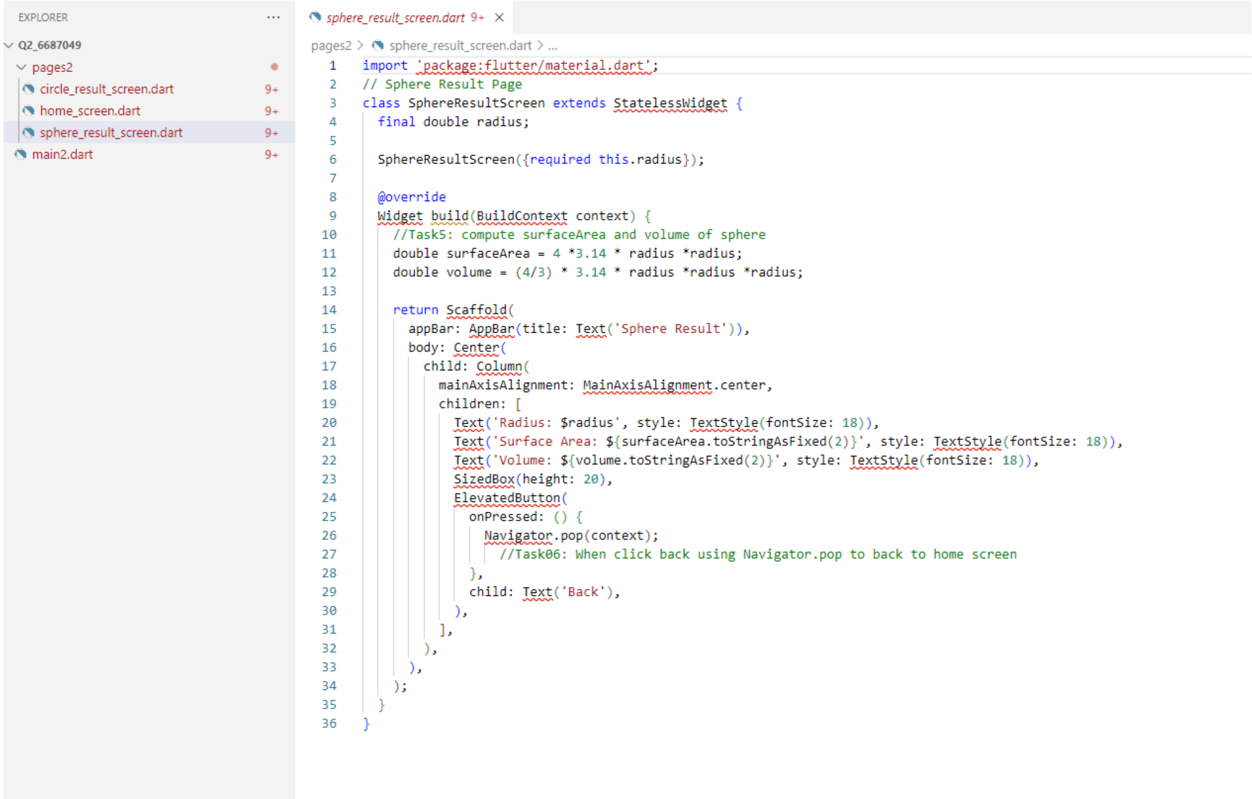
```dart
import 'package:flutter/material.dart';
// Circle Result Page
class CircleResultScreen extends StatelessWidget {
  final double radius;

  CircleResultScreen({required this.radius});

  @override
  Widget build(BuildContext context) {
    //Task5: compute circumference and area of circle
    double circumference = 2 * 3.14 * radius;
    double area = 3.14 * ((radius*radius));

    return Scaffold(
      appBar: AppBar(title: Text('Circle Result')),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Text('Radius: $radius', style: TextStyle(fontSize: 18)),
            Text('Circumference: ${circumference.toStringAsFixed(2)}', style: TextStyle(fontSize: 18)),
            Text('Area: ${area.toStringAsFixed(2)}', style: TextStyle(fontSize: 18)),
            SizedBox(height: 20),
            ElevatedButton(
              onPressed: () {
                Navigator.pop(context);
                  //Task06:  make the Back button return to the Home Screen using Navigator.pop()
              },
              child: Text('Back'),
            ),
          ],
        ),
      ),
    );
  }
}
```


EXPLORER
Q2_6687049
pages2
circle_result_screen.dart 9+
home_screen.dart 9+
sphere_result_screen.dart 9+
main2.dart 9+
OUTLINE
TIMELINE
DEPENDENCIES
home_screen.dart 9+
pages2 > home_screen.dart > _HomeScreenState > build

```dart
import 'package:flutter/material.dart';
import 'circle_result_screen.dart';
import 'sphere_result_screen.dart';

//Task 01: Import Required Screens
//import both circle_result_screen.dart and sphere_result_screen.dart.

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

  @override
  _HomeScreenState createState() => _HomeScreenState();
}

class _HomeScreenState extends State<HomeScreen> {
  // Task 02: Implement a Text Field Controller

  // Create a TextEditingController.
  // Assign this controller to the TextField widget so it can read user input.

  String _selectedOption = 'Circle'; // Default selection
  final TextEditingController _radiuscontroller = TextEditingController();


  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text('Circle & Sphere Calculator')),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            TextField(
              controller: _radiuscontroller,
              keyboardType: TextInputType.number,
              decoration: InputDecoration(
                labelText: 'Enter radius',
                border: OutlineInputBorder(),
              ),
            ),
            SizedBox(height: 20),
            Text("Select Shape:"),
            Row(
              children: [
                Radio(
                  value: 'Circle',
```

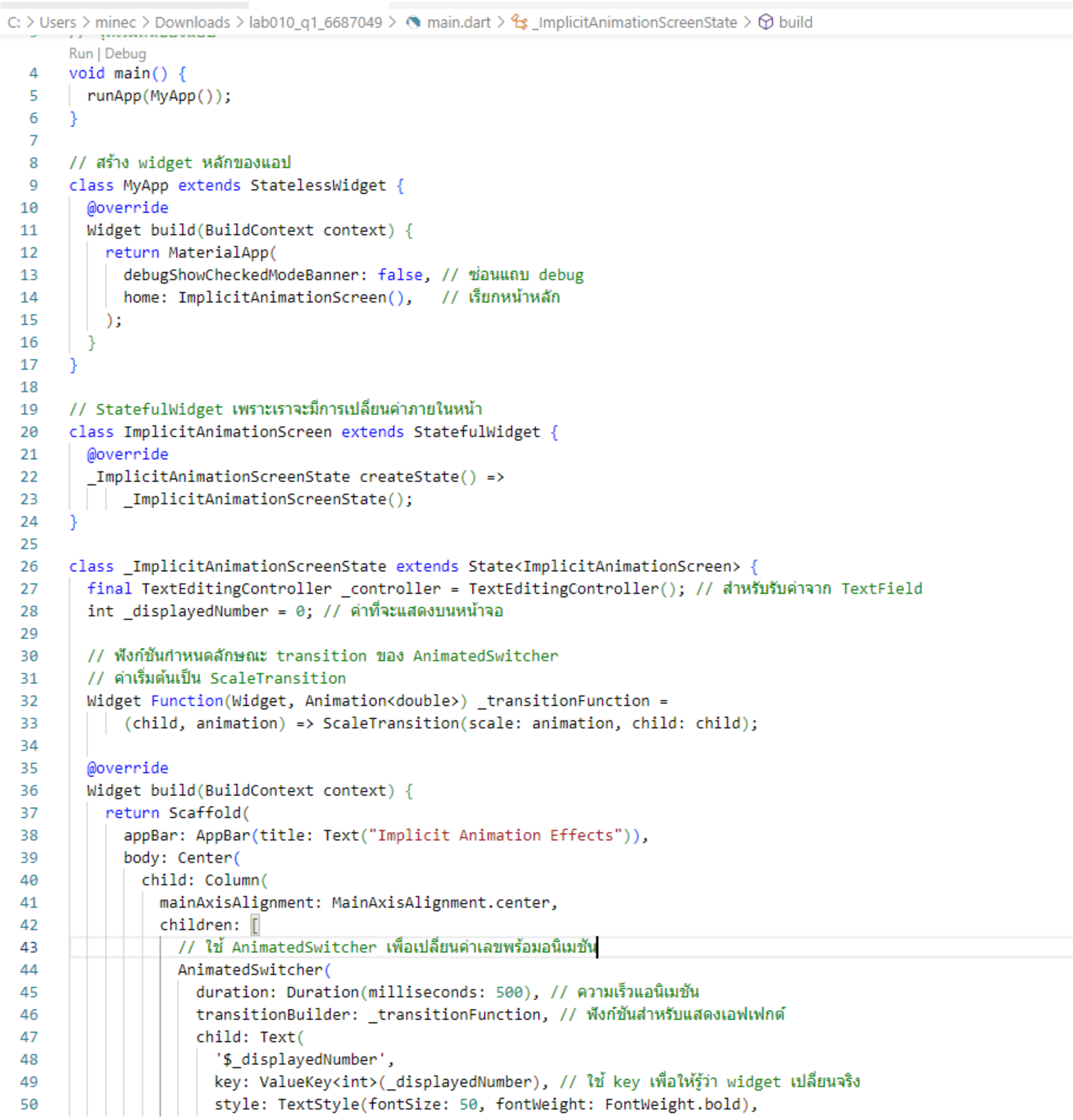

EXPLORER
Q2_6687049
pages2
circle_result_screen.dart 9+
home_screen.dart 9+
sphere_result_screen.dart 9+
main2.dart 9+
OUTLINE
TIMELINE
DEPENDENCIES
home_screen.dart 9+
pages2 > home_screen.dart > _HomeScreenState > build

```dart
class _HomeScreenState extends State<HomeScreen> {
  Widget build(BuildContext context) {
                  value: 'Circle',
                  groupValue: _selectedOption,
                  onChanged: (value) {
                    //Task 03: Handle Radio Button Selection
                    //When the Circle radio button is selected, update _selectedOption using setState().
                    setState(() {
                      _selectedOption =value.toString();
                    });
                  },
                ),
                Text('Circle'),
                Radio(
                  value: 'Sphere',
                  groupValue: _selectedOption,
                  onChanged: (value) {
                    //Task 03: Handle Radio Button Selection
                    //When the Sphere radio button is selected, update _selectedOption using setState().
                    setState(() {
                      _selectedOption =value.toString();
                    });
                  },
                ),
                Text('Sphere'),
              ],
            ),
            SizedBox(height: 20),
            Center(
              child: ElevatedButton(
                //
                onPressed: () {
                  //Task04: check condition that radius is not empty if so, show snackbar
                  //if circle radio is chosen route to page CircleResultScreen and pass radius to its constructor using Navigator.push()
                  //if Sphere radio is chosen route to page SphereResultScreen and pass radius to its constructor usign Navigator.push()
                  double? radius = double.tryParse(_radiuscontroller.text);
                  if (_selectedOption == 'Circle'){
                    Navigator.push(context,MaterialPageRoute(builder: (context)=> CircleResultScreen(radius: radius!),),);
                  } else{
                    Navigator.push(context,MaterialPageRoute(builder: (context)=> SphereResultScreen(radius: radius!),),);
                  }
                },
                child: Text('Calculate'),
              ),
            ),
          ],
        ),
      ),
```


EXPLORER
...
Q2_6687049
pages2
circle_result_screen.dart 9+
home_screen.dart 9+
sphere_result_screen.dart 9+
main2.dart 9+
sphere_result_screen.dart 9+
pages2 > sphere_result_screen.dart > ...

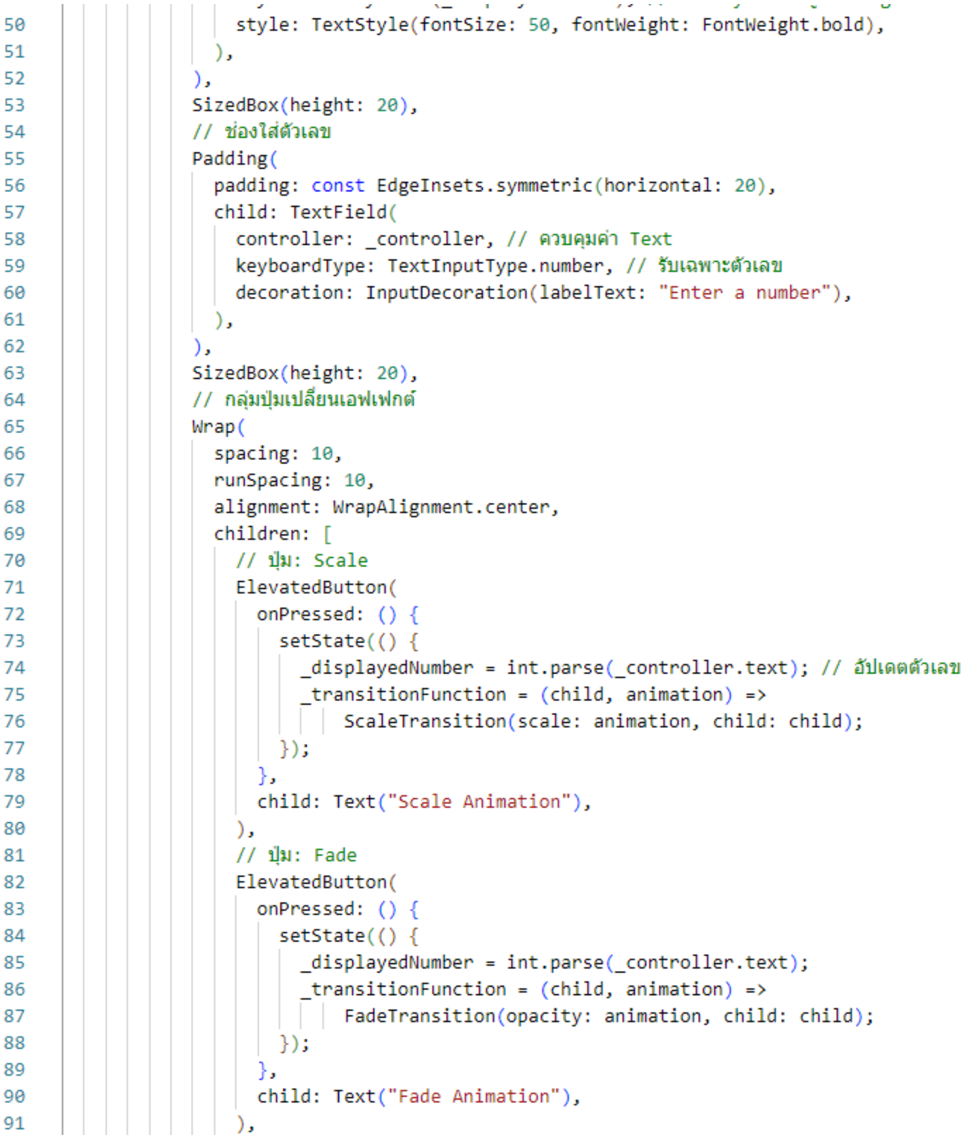
```dart
import 'package:flutter/material.dart';
// Sphere Result Page
class SphereResultScreen extends StatelessWidget {
  final double radius;

  SphereResultScreen({required this.radius});

  @override
  Widget build(BuildContext context) {
    //Task5: compute surfaceArea and volume of sphere
    double surfaceArea = 4 *3.14 * radius *radius;
    double volume = (4/3) * 3.14 * radius *radius *radius;

    return Scaffold(
      appBar: AppBar(title: Text('Sphere Result')),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Text('Radius: $radius', style: TextStyle(fontSize: 18)),
            Text('Surface Area: ${surfaceArea.toStringAsFixed(2)}', style: TextStyle(fontSize: 18)),
            Text('Volume: ${volume.toStringAsFixed(2)}', style: TextStyle(fontSize: 18)),
            SizedBox(height: 20),
            ElevatedButton(
              onPressed: () {
                Navigator.pop(context);
                  //Task06: When click back using Navigator.pop to back to home screen
              },
              child: Text('Back'),
            ),
          ],
        ),
      ),
    );
  }
}
```

L10-Q1 อนิเมชั่น

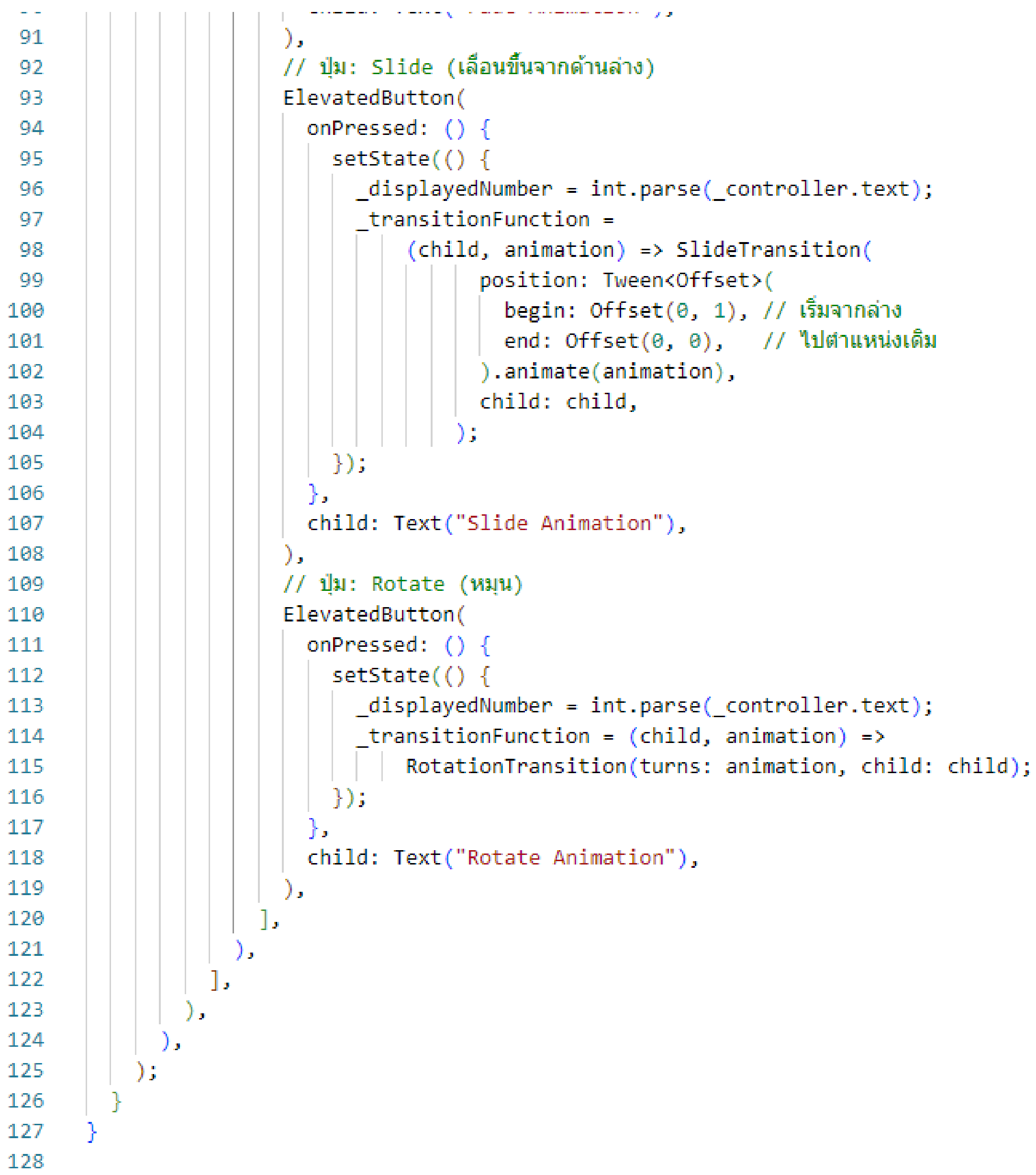


C: > Users > minec > Downloads > lab010_q1_6687049 > main.dart > _ImplicitAnimationScreenState > build

```dart
Run | Debug
void main() {
  runApp(MyApp());
}

// สร้าง widget หลักของแอป
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false, // ซ่อนแถบ debug
      home: ImplicitAnimationScreen(),   // เรียกหน้าหลัก
    );
  }
}

// StatefulWidget เพราะเราจะมีการเปลี่ยนค่าภายในหน้า
class ImplicitAnimationScreen extends StatefulWidget {
  @override
  _ImplicitAnimationScreenState createState() =>
      _ImplicitAnimationScreenState();
}

class _ImplicitAnimationScreenState extends State<ImplicitAnimationScreen> {
  final TextEditingController _controller = TextEditingController(); // สำหรับรับค่าจาก TextField
  int _displayedNumber = 0; // ค่าที่จะแสดงบนหน้าจอ

  // ฟังก์ชันกำหนดลักษณะ transition ของ AnimatedSwitcher
  // ค่าเริ่มต้นเป็น ScaleTransition
  Widget Function(Widget, Animation<double>) _transitionFunction =
      (child, animation) => ScaleTransition(scale: animation, child: child);

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text("Implicit Animation Effects")),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            // ใช้ AnimatedSwitcher เพื่อเปลี่ยนค่าเลขพร้อมอนิเมชัน
            AnimatedSwitcher(
              duration: Duration(milliseconds: 500), // ความเร็วแอนิเมชัน
              transitionBuilder: _transitionFunction, // ฟังก์ชันสำหรับแสดงเอฟเฟกต์
              child: Text(
                '$_displayedNumber',
                key: ValueKey<int>(_displayedNumber), // ใช้ key เพื่อให้รู้ว่า widget เปลี่ยนจริง
                style: TextStyle(fontSize: 50, fontWeight: FontWeight.bold),
```



```
    style: TextStyle(fontSize: 50, fontWeight: FontWeight.bold),
  ),
),
SizedBox(height: 20),
// ช่องใส่ตัวเลข
Padding(
  padding: const EdgeInsets.symmetric(horizontal: 20),
  child: TextField(
    controller: _controller, // ควบคุมค่า Text
    keyboardType: TextInputType.number, // รับเฉพาะตัวเลข
    decoration: InputDecoration(labelText: "Enter a number"),
  ),
),
SizedBox(height: 20),
// กลุ่มปุ่มเปลี่ยนเอฟเฟกต์
Wrap(
  spacing: 10,
  runSpacing: 10,
  alignment: WrapAlignment.center,
  children: [
    // ปุ่ม: Scale
    ElevatedButton(
      onPressed: () {
        setState(() {
          _displayedNumber = int.parse(_controller.text); // อัปเดตตัวเลข
          _transitionFunction = (child, animation) =>
              ScaleTransition(scale: animation, child: child);
        });
      },
      child: Text("Scale Animation"),
    ),
    // ปุ่ม: Fade
    ElevatedButton(
      onPressed: () {
        setState(() {
          _displayedNumber = int.parse(_controller.text);
          _transitionFunction = (child, animation) =>
              FadeTransition(opacity: animation, child: child);
        });
      },
      child: Text("Fade Animation"),
    ),
```



```
                ),
                // ปุ่ม: Slide (เลื่อนขึ้นจากด้านล่าง)
                ElevatedButton(
                  onPressed: () {
                    setState(() {
                      _displayedNumber = int.parse(_controller.text);
                      _transitionFunction =
                          (child, animation) => SlideTransition(
                                position: Tween<Offset>(
                                  begin: Offset(0, 1), // เริ่มจากล่าง
                                  end: Offset(0, 0),   // ไปตำแหน่งเดิม
                                ).animate(animation),
                                child: child,
                              );
                    });
                  },
                  child: Text("Slide Animation"),
                ),
                // ปุ่ม: Rotate (หมุน)
                ElevatedButton(
                  onPressed: () {
                    setState(() {
                      _displayedNumber = int.parse(_controller.text);
                      _transitionFunction = (child, animation) =>
                          RotationTransition(turns: animation, child: child);
                    });
                  },
                  child: Text("Rotate Animation"),
                ),
              ],
            ),
          ],
        ),
      ),
    );
  }
}

```

⚙️ รายละเอียดของอนิเมชันแต่ละแบบ

✅ 1. ScaleTransition (`Scale Animation`)

dart 🗍 คัดลอก ✎ แก้ไข

```dart
ScaleTransition(scale: animation, child: child);
```

- ขยาย/ย่อ Widget ตามค่า `animation`
- ค่า `animation` อยู่ในช่วง `0.0 → 1.0`
- ใช้สำหรับให้ widget ค่อยๆ โตขึ้นจากขนาด 0 → 100% (หรือหดลง)

ผลลัพธ์: ตัวเลขจะ "ซูมเข้ามา" หรือ "ซูมออกไป" อย่างนุ่มนวล

✅ 2. FadeTransition (`Fade Animation`)

dart 🗍 คัดลอก ✎ แก้ไข

```dart
FadeTransition(opacity: animation, child: child);
```

- ค่อยๆ เปลี่ยนค่าความโปร่งใส (`opacity`) ของ widget
- ค่า `animation` บอกระดับความทึบ (0 = มองไม่เห็น, 1 = มองเห็นเต็มที่)

ผลลัพธ์: ตัวเลขจะค่อยๆ จางหายไป และตัวใหม่จะค่อยๆ ปรากฏขึ้น

✅ 3. SlideTransition (`Slide Animation`)

dart ⧉ คัดลอก ✎ แก้ไข

```dart
SlideTransition(
  position: Tween<Offset>(
    begin: Offset(0, 1),
    end: Offset(0, 0),
  ).animate(animation),
  child: child,
);
```

- ขยับตำแหน่ง widget ตามแกน X/Y โดยใช้ค่า `Offset`
- `Offset(0, 1)` หมายถึง เริ่มจาก **ล่างขึ้นบน**
- `Offset(1, 0)` หมายถึง ขยับจากขวาไปซ้าย

ผลลัพธ์: ตัวเลขจะ "เลื่อนขึ้น" เข้ามาจากด้านล่างของหน้าจอ

✅ 4. RotationTransition (`Rotate Animation`)

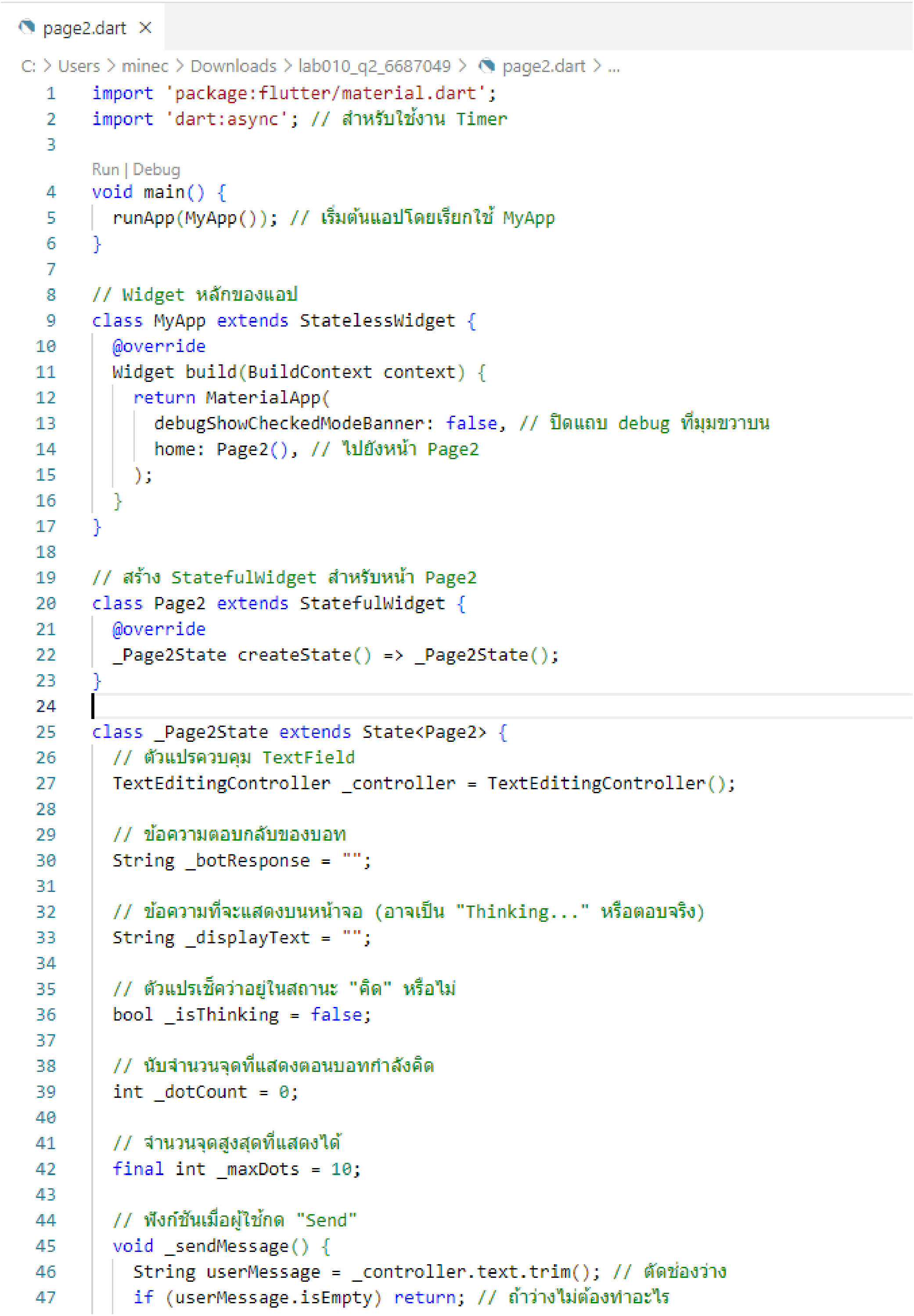
dart ⧉ คัดลอก ✎ แก้ไข

```dart
RotationTransition(turns: animation, child: child);
```

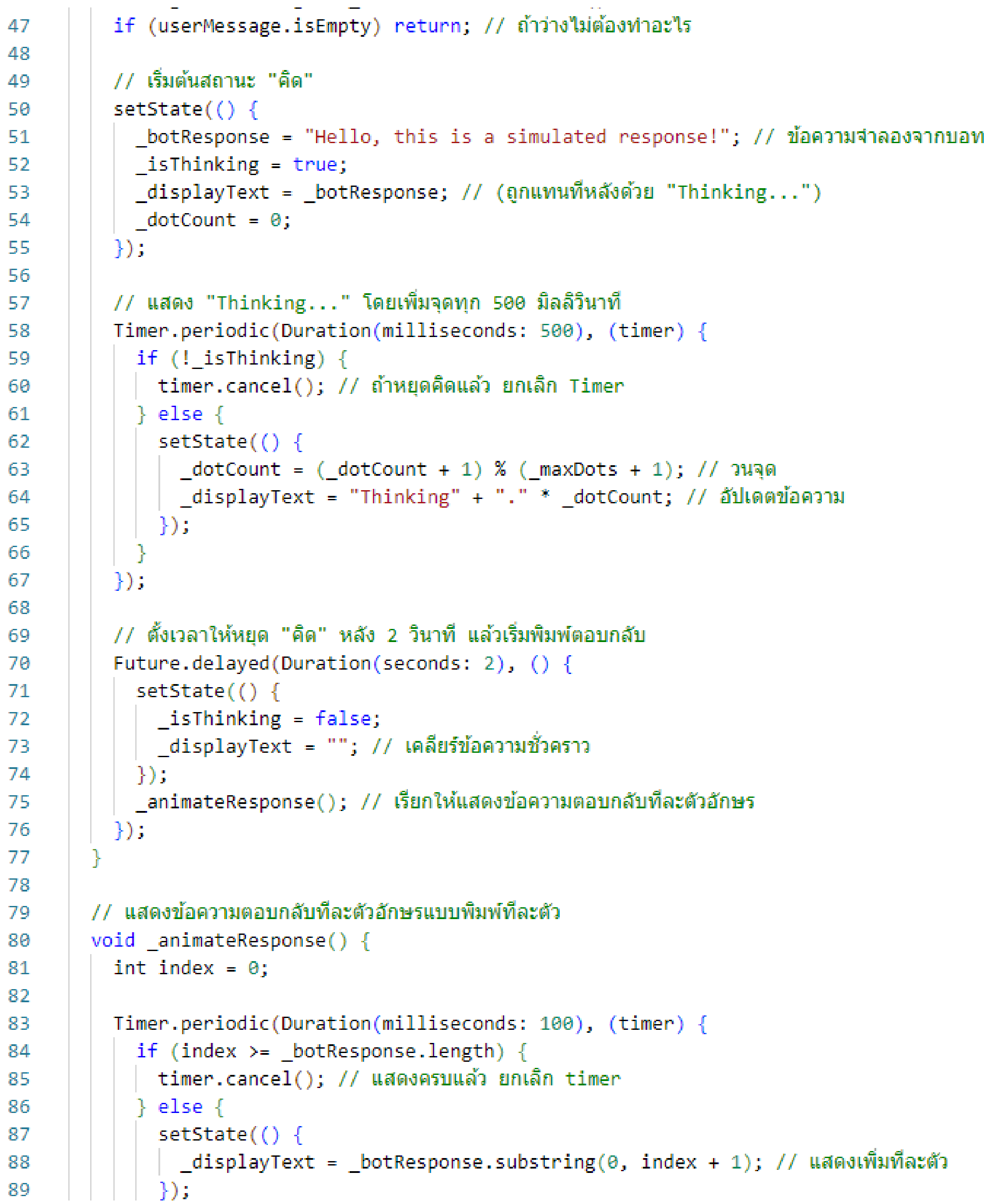
- หมุน widget ตามค่า `animation`
- 1.0 = หมุนหนึ่งรอบ (360°)

ผลลัพธ์: ตัวเลขจะหมุนเข้ามาหรือหมุนออกไป

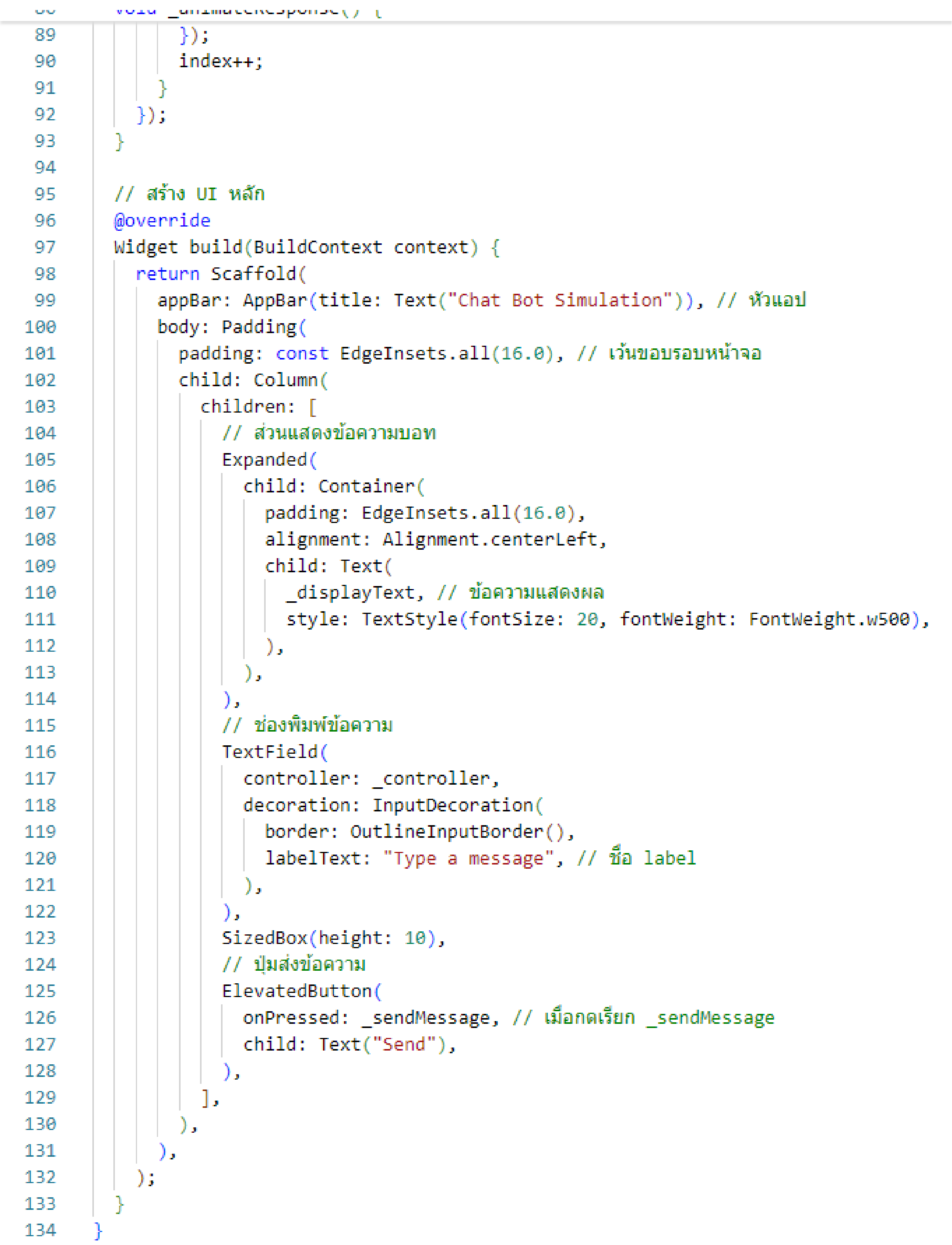
L10-Q2 อนิเมชั่น


page2.dart
C: > Users > minec > Downloads > lab010_q2_6687049 > page2.dart > ...
Run | Debug

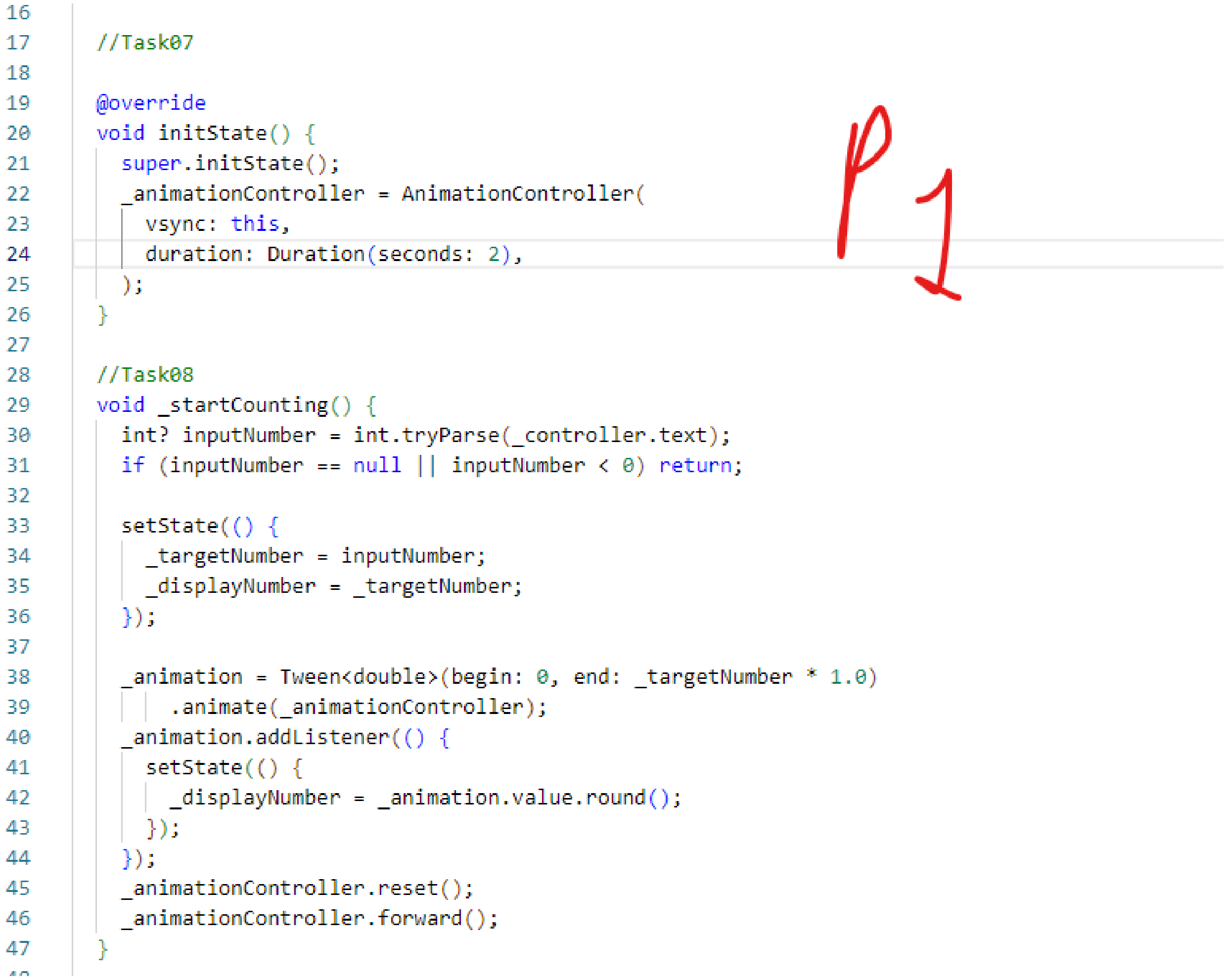
```dart
import 'package:flutter/material.dart';
import 'dart:async'; // สำหรับใช้งาน Timer

void main() {
  runApp(MyApp()); // เริ่มต้นแอปโดยเรียกใช้ MyApp
}

// Widget หลักของแอป
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false, // ปิดแถบ debug ที่มุมขวาบน
      home: Page2(), // ไปยังหน้า Page2
    );
  }
}

// สร้าง StatefulWidget สำหรับหน้า Page2
class Page2 extends StatefulWidget {
  @override
  _Page2State createState() => _Page2State();
}

class _Page2State extends State<Page2> {
  // ตัวแปรควบคุม TextField
  TextEditingController _controller = TextEditingController();

  // ข้อความตอบกลับของบอท
  String _botResponse = "";

  // ข้อความที่จะแสดงบนหน้าจอ (อาจเป็น "Thinking..." หรือตอบจริง)
  String _displayText = "";

  // ตัวแปรเช็คว่าอยู่ในสถานะ "คิด" หรือไม่
  bool _isThinking = false;

  // นับจำนวนจุดที่แสดงตอนบอทกำลังคิด
  int _dotCount = 0;

  // จำนวนจุดสูงสุดที่แสดงได้
  final int _maxDots = 10;

  // ฟังก์ชันเมื่อผู้ใช้กด "Send"
  void _sendMessage() {
    String userMessage = _controller.text.trim(); // ตัดช่องว่าง
    if (userMessage.isEmpty) return; // ถ้าว่างไม่ต้องทำอะไร
```



```
    if (userMessage.isEmpty) return; // ถ้าว่างไม่ต้องทำอะไร

    // เริ่มต้นสถานะ "คิด"
    setState(() {
      _botResponse = "Hello, this is a simulated response!"; // ข้อความจำลองจากบอท
      _isThinking = true;
      _displayText = _botResponse; // (ถูกแทนที่หลังด้วย "Thinking...")
      _dotCount = 0;
    });

    // แสดง "Thinking..." โดยเพิ่มจุดทุก 500 มิลลิวินาที
    Timer.periodic(Duration(milliseconds: 500), (timer) {
      if (!_isThinking) {
        timer.cancel(); // ถ้าหยุดคิดแล้ว ยกเลิก Timer
      } else {
        setState(() {
          _dotCount = (_dotCount + 1) % (_maxDots + 1); // วนจุด
          _displayText = "Thinking" + "." * _dotCount; // อัปเดตข้อความ
        });
      }
    });

    // ตั้งเวลาให้หยุด "คิด" หลัง 2 วินาที แล้วเริ่มพิมพ์ตอบกลับ
    Future.delayed(Duration(seconds: 2), () {
      setState(() {
        _isThinking = false;
        _displayText = ""; // เคลียร์ข้อความชั่วคราว
      });
      _animateResponse(); // เรียกให้แสดงข้อความตอบกลับทีละตัวอักษร
    });
  }

  // แสดงข้อความตอบกลับทีละตัวอักษรแบบพิมพ์ทีละตัว
  void _animateResponse() {
    int index = 0;

    Timer.periodic(Duration(milliseconds: 100), (timer) {
      if (index >= _botResponse.length) {
        timer.cancel(); // แสดงครบแล้ว ยกเลิก timer
      } else {
        setState(() {
          _displayText = _botResponse.substring(0, index + 1); // แสดงเพิ่มทีละตัว
        });
```



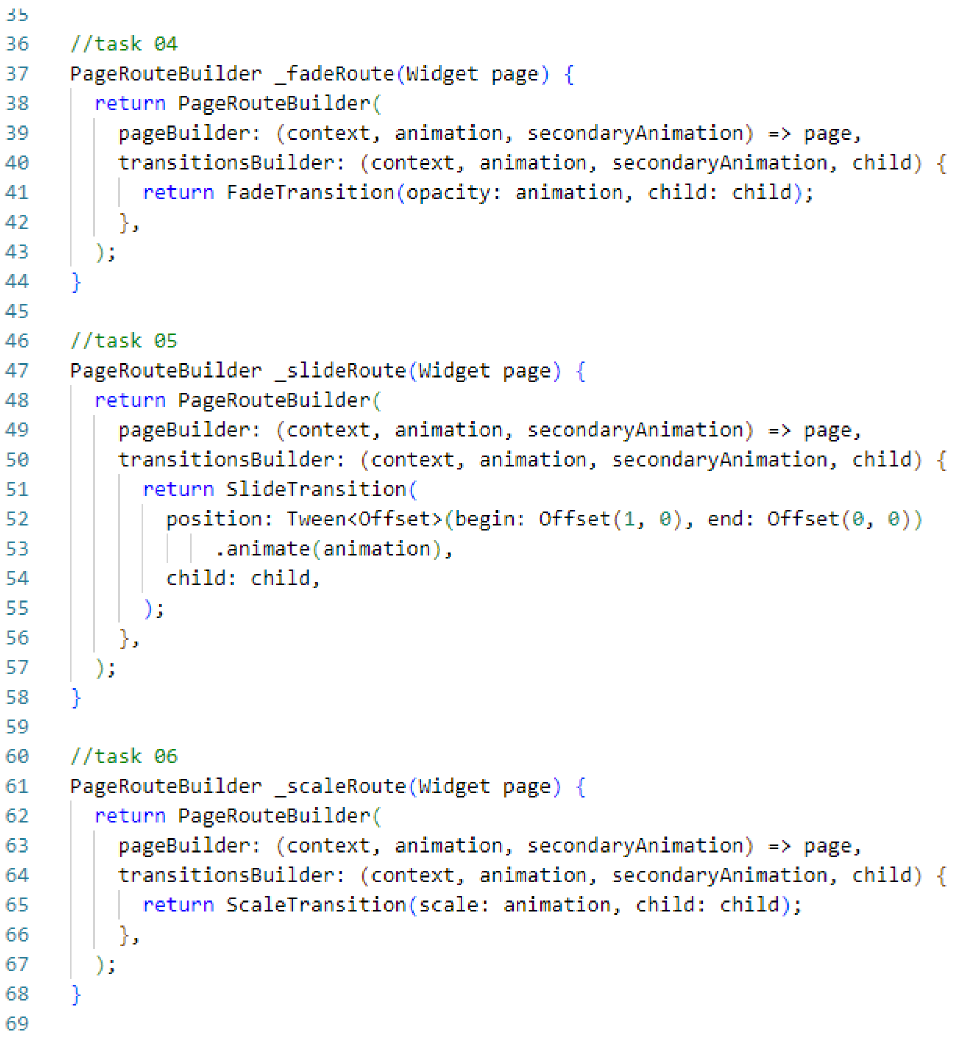
```
  void _animateResponse() {
        });
        index++;
      }
    });
  }

  // สร้าง UI หลัก
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text("Chat Bot Simulation")), // หัวแอป
      body: Padding(
        padding: const EdgeInsets.all(16.0), // เว้นขอบรอบหน้าจอ
        child: Column(
          children: [
            // ส่วนแสดงข้อความบอท
            Expanded(
              child: Container(
                padding: EdgeInsets.all(16.0),
                alignment: Alignment.centerLeft,
                child: Text(
                  _displayText, // ข้อความแสดงผล
                  style: TextStyle(fontSize: 20, fontWeight: FontWeight.w500),
                ),
              ),
            ),
            // ช่องพิมพ์ข้อความ
            TextField(
              controller: _controller,
              decoration: InputDecoration(
                border: OutlineInputBorder(),
                labelText: "Type a message", // ชื่อ label
              ),
            ),
            SizedBox(height: 10),
            // ปุ่มส่งข้อความ
            ElevatedButton(
              onPressed: _sendMessage, // เมื่อกดเรียก _sendMessage
              child: Text("Send"),
            ),
          ],
        ),
      ),
    );
  }
}
```



```

//Task07

@override
void initState() {
  super.initState();
  _animationController = AnimationController(
    vsync: this,
    duration: Duration(seconds: 2),
  );
}

//Task08
void _startCounting() {
  int? inputNumber = int.tryParse(_controller.text);
  if (inputNumber == null || inputNumber < 0) return;

  setState(() {
    _targetNumber = inputNumber;
    _displayNumber = _targetNumber;
  });

  _animation = Tween<double>(begin: 0, end: _targetNumber * 1.0)
      .animate(_animationController);
  _animation.addListener(() {
    setState(() {
      _displayNumber = _animation.value.round();
    });
  });
  _animationController.reset();
  _animationController.forward();
}
```

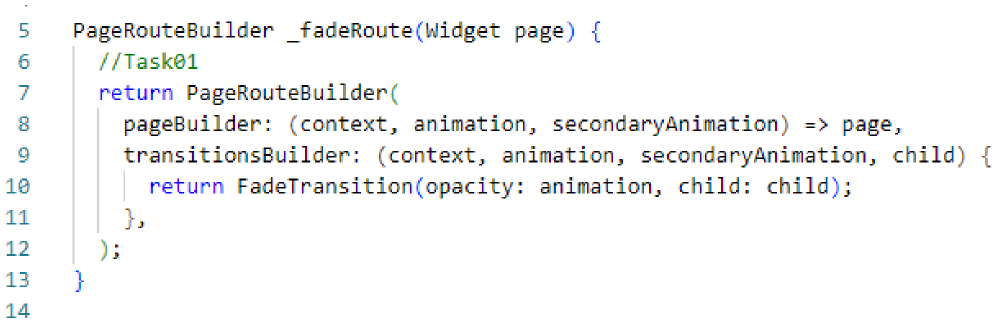
P1

h_p



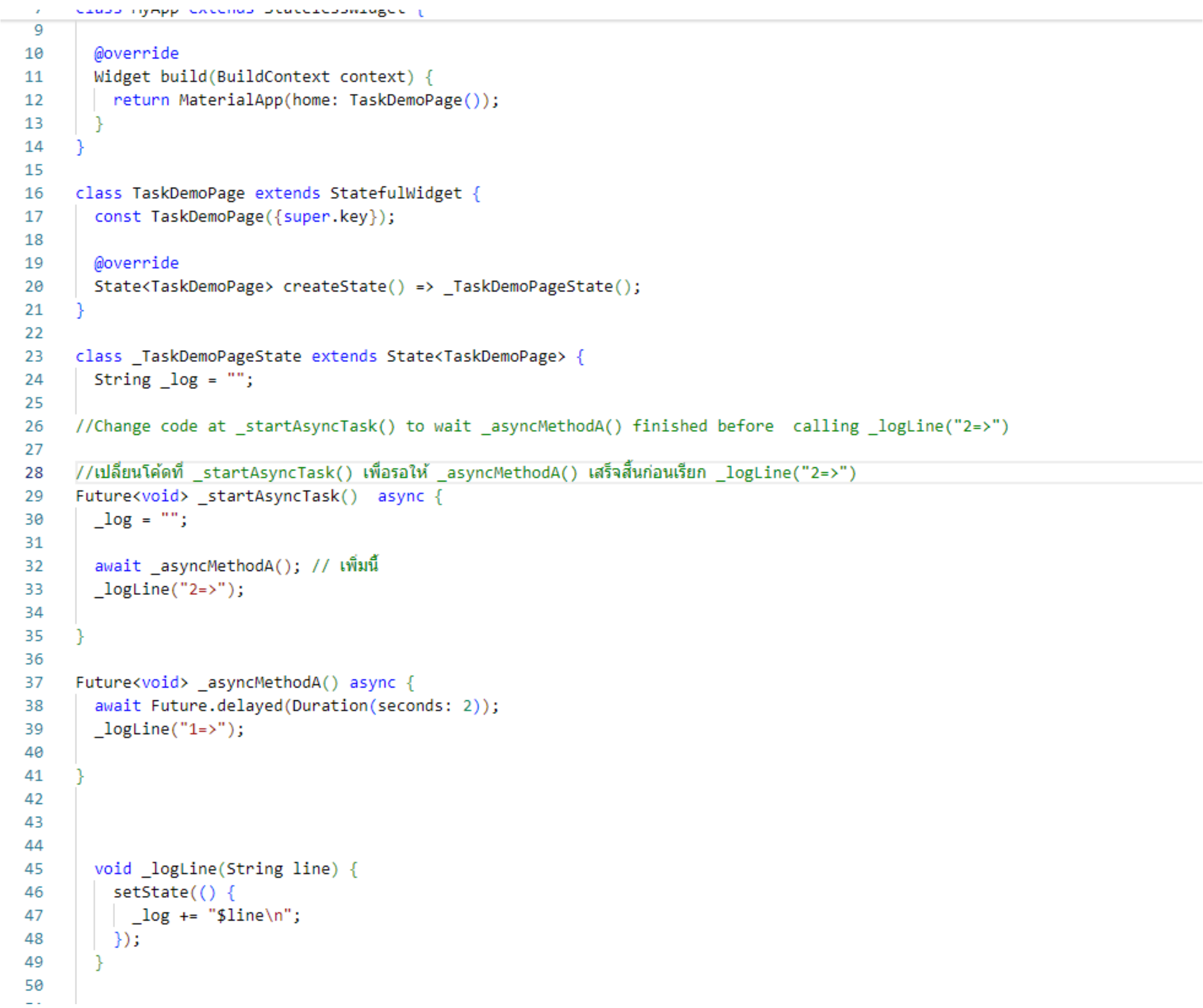
```
//task 04
PageRouteBuilder _fadeRoute(Widget page) {
  return PageRouteBuilder(
    pageBuilder: (context, animation, secondaryAnimation) => page,
    transitionsBuilder: (context, animation, secondaryAnimation, child) {
      return FadeTransition(opacity: animation, child: child);
    },
  );
}

//task 05
PageRouteBuilder _slideRoute(Widget page) {
  return PageRouteBuilder(
    pageBuilder: (context, animation, secondaryAnimation) => page,
    transitionsBuilder: (context, animation, secondaryAnimation, child) {
      return SlideTransition(
        position: Tween<Offset>(begin: Offset(1, 0), end: Offset(0, 0))
            .animate(animation),
        child: child,
      );
    },
  );
}

//task 06
PageRouteBuilder _scaleRoute(Widget page) {
  return PageRouteBuilder(
    pageBuilder: (context, animation, secondaryAnimation) => page,
    transitionsBuilder: (context, animation, secondaryAnimation, child) {
      return ScaleTransition(scale: animation, child: child);
    },
  );
}
```

L_p



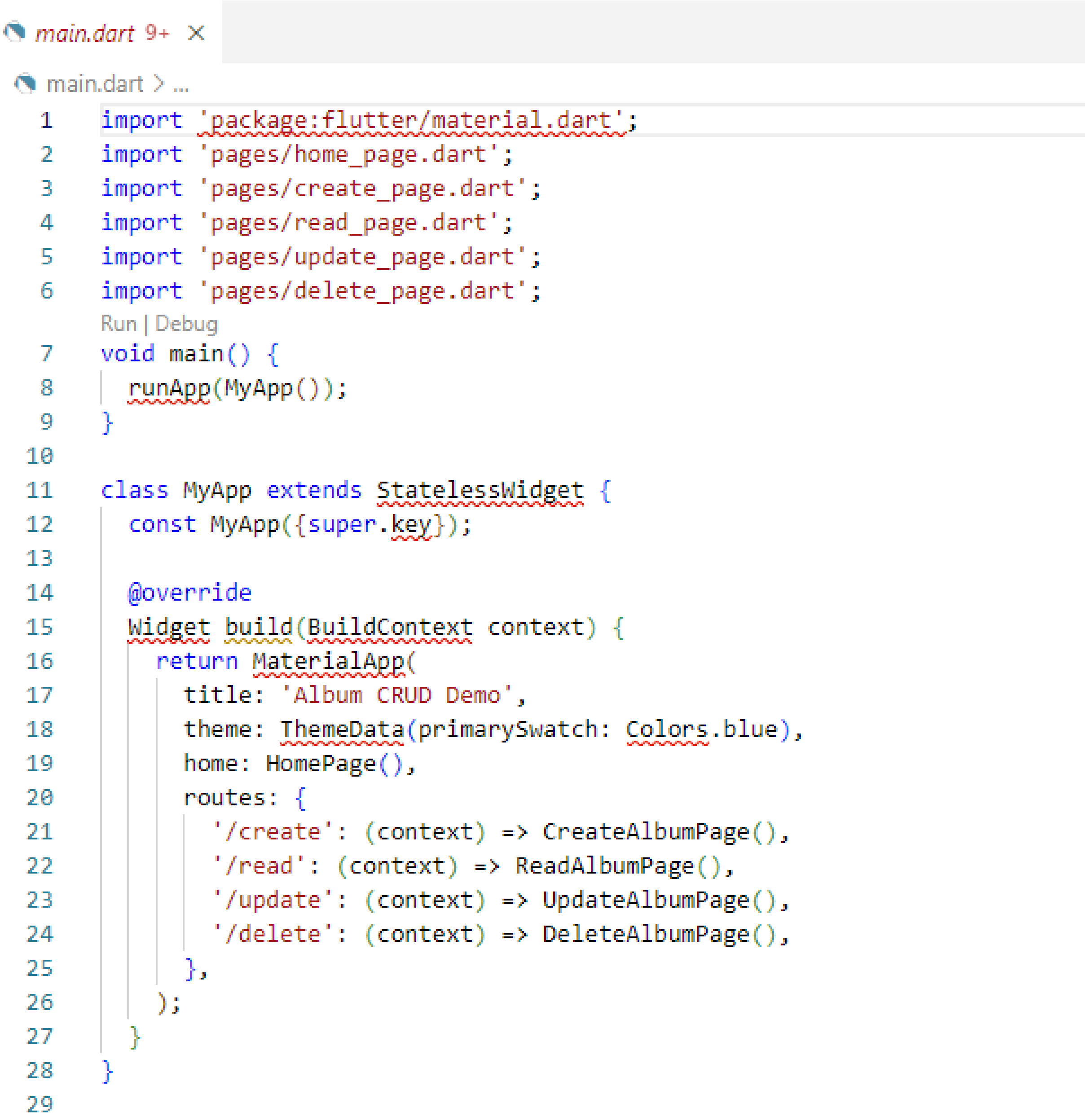
```
PageRouteBuilder _fadeRoute(Widget page) {
  //Task01
  return PageRouteBuilder(
    pageBuilder: (context, animation, secondaryAnimation) => page,
    transitionsBuilder: (context, animation, secondaryAnimation, child) {
      return FadeTransition(opacity: animation, child: child);
    },
  );
}
```

L11-Q1 **API calls**

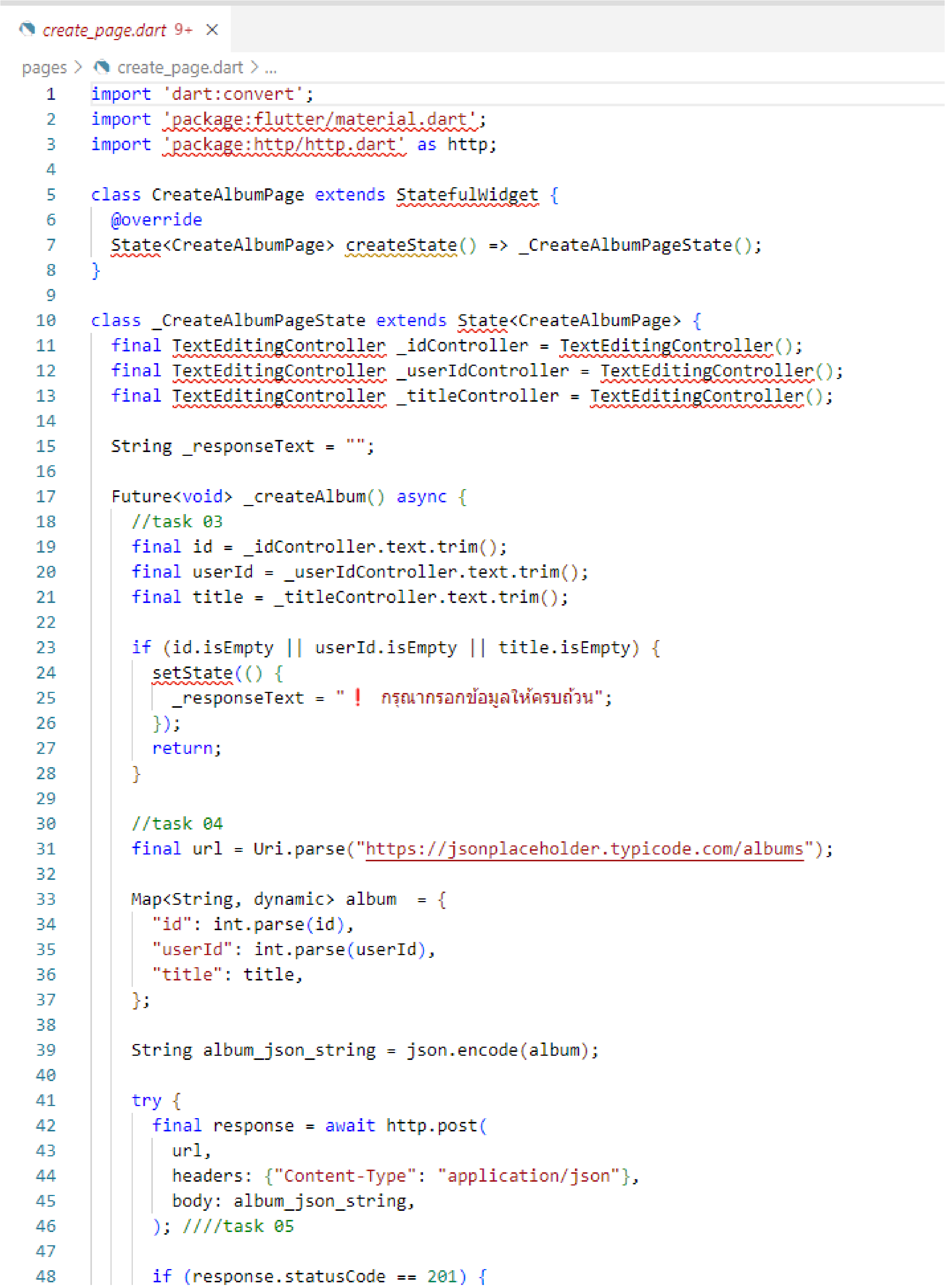


```
class MyApp extends StatelessWidget {

  @override
  Widget build(BuildContext context) {
    return MaterialApp(home: TaskDemoPage());
  }
}

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

  @override
  State<TaskDemoPage> createState() => _TaskDemoPageState();
}

class _TaskDemoPageState extends State<TaskDemoPage> {
  String _log = "";

//Change code at _startAsyncTask() to wait _asyncMethodA() finished before  calling _logLine("2=>")

//เปลี่ยนโค้ดที่ _startAsyncTask() เพื่อรอให้ _asyncMethodA() เสร็จสิ้นก่อนเรียก _logLine("2=>")
Future<void> _startAsyncTask()  async {
  _log = "";

  await _asyncMethodA(); // เพิ่มนี้
  _logLine("2=>");

}

Future<void> _asyncMethodA() async {
  await Future.delayed(Duration(seconds: 2));
  _logLine("1=>");

}



  void _logLine(String line) {
    setState(() {
      _log += "$line\n";
    });
  }

```

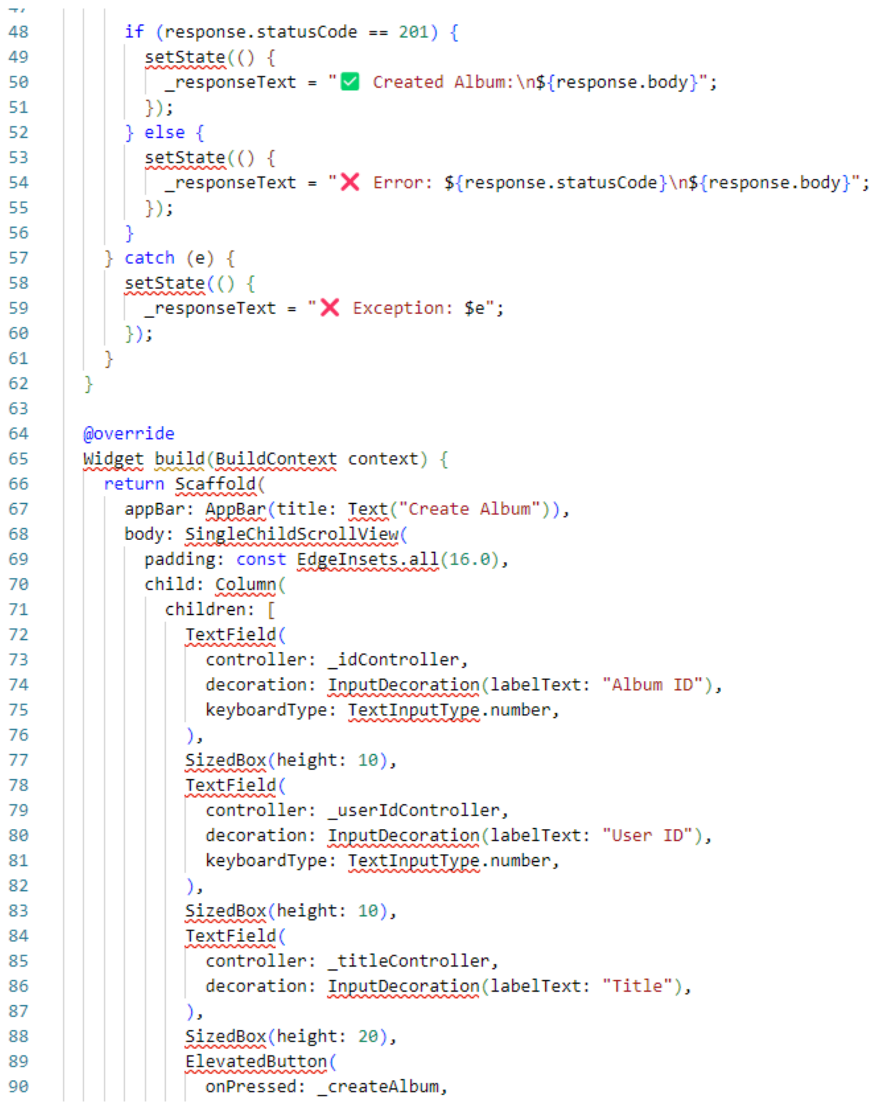
L11-Q2 **API calls**



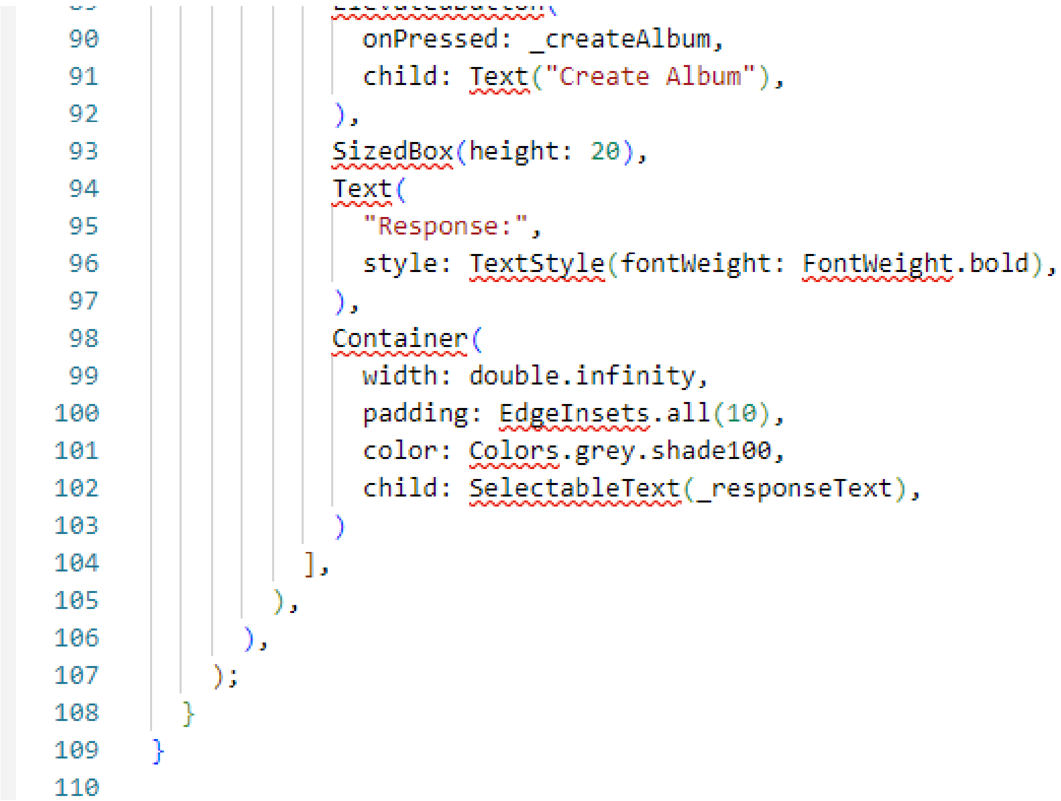
```dart
import 'package:flutter/material.dart';
import 'pages/home_page.dart';
import 'pages/create_page.dart';
import 'pages/read_page.dart';
import 'pages/update_page.dart';
import 'pages/delete_page.dart';
void main() {
  runApp(MyApp());
}

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

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Album CRUD Demo',
      theme: ThemeData(primarySwatch: Colors.blue),
      home: HomePage(),
      routes: {
        '/create': (context) => CreateAlbumPage(),
        '/read': (context) => ReadAlbumPage(),
        '/update': (context) => UpdateAlbumPage(),
        '/delete': (context) => DeleteAlbumPage(),
      },
    );
  }
}

```



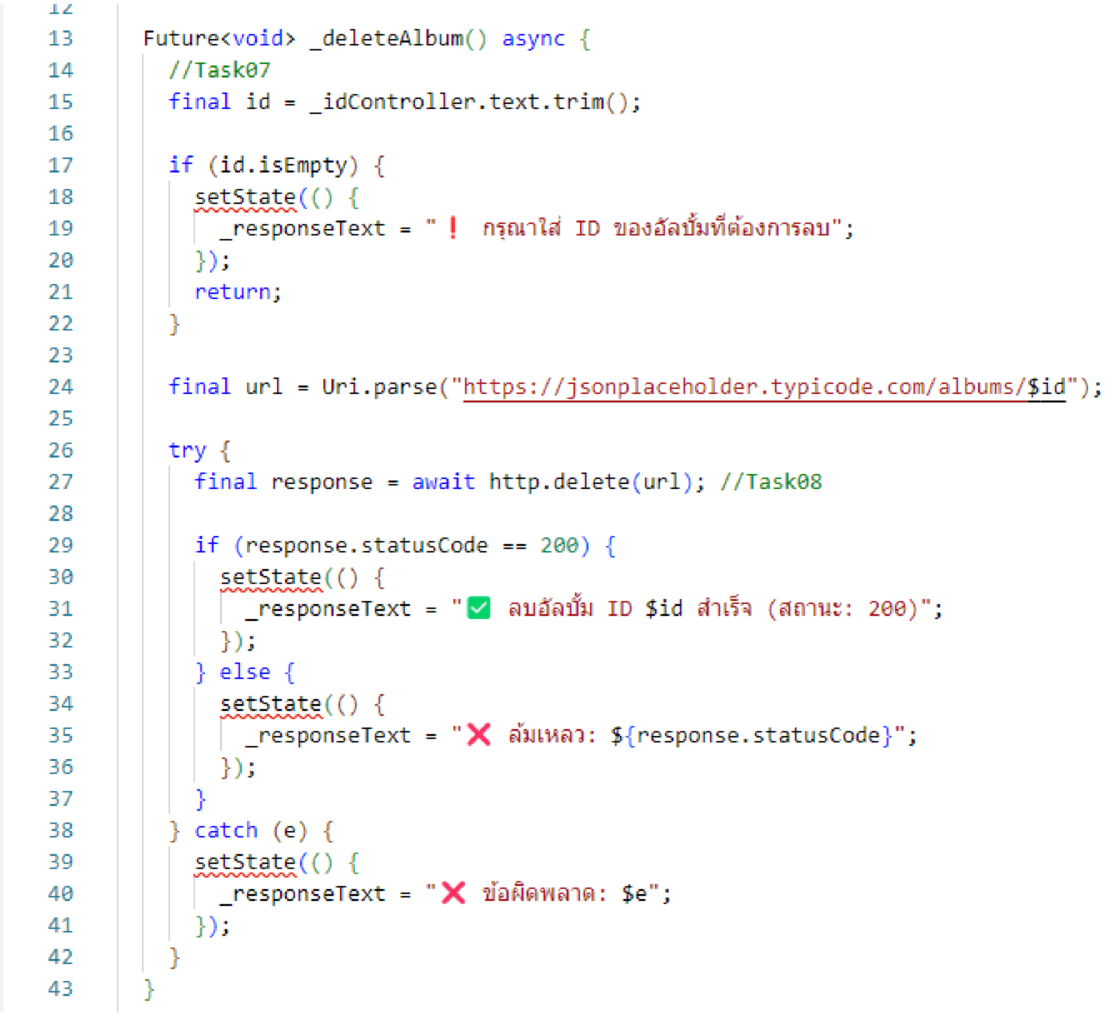
```dart
import 'dart:convert';
import 'package:flutter/material.dart';
import 'package:http/http.dart' as http;

class CreateAlbumPage extends StatefulWidget {
  @override
  State<CreateAlbumPage> createState() => _CreateAlbumPageState();
}

class _CreateAlbumPageState extends State<CreateAlbumPage> {
  final TextEditingController _idController = TextEditingController();
  final TextEditingController _userIdController = TextEditingController();
  final TextEditingController _titleController = TextEditingController();

  String _responseText = "";

  Future<void> _createAlbum() async {
    //task 03
    final id = _idController.text.trim();
    final userId = _userIdController.text.trim();
    final title = _titleController.text.trim();

    if (id.isEmpty || userId.isEmpty || title.isEmpty) {
      setState(() {
        _responseText = "❗ กรุณากรอกข้อมูลให้ครบถ้วน";
      });
      return;
    }

    //task 04
    final url = Uri.parse("https://jsonplaceholder.typicode.com/albums");

    Map<String, dynamic> album  = {
      "id": int.parse(id),
      "userId": int.parse(userId),
      "title": title,
    };

    String album_json_string = json.encode(album);

    try {
      final response = await http.post(
        url,
        headers: {"Content-Type": "application/json"},
        body: album_json_string,
      ); ////task 05

      if (response.statusCode == 201) {
```



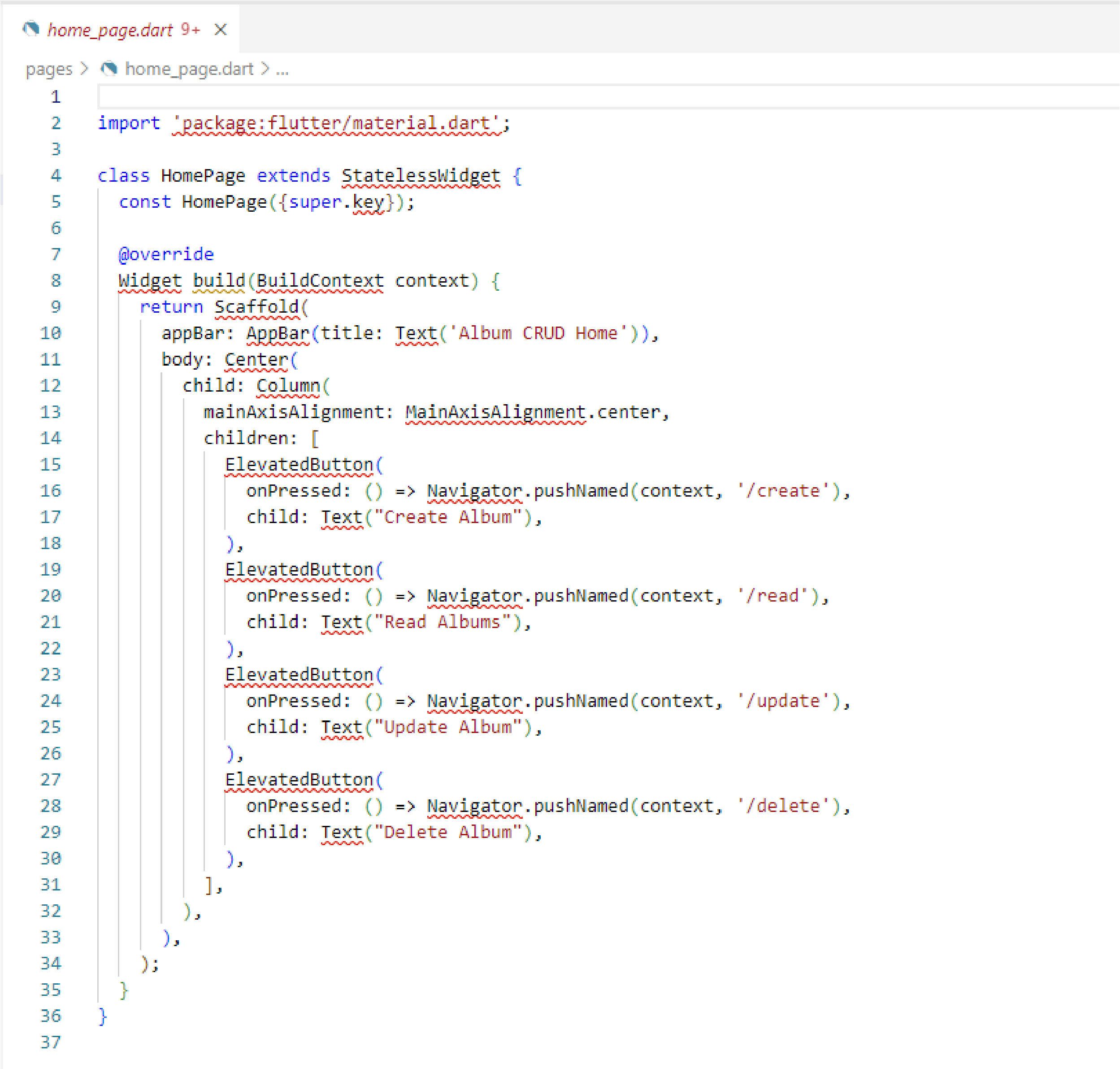
```
      if (response.statusCode == 201) {
        setState(() {
          _responseText = "✅ Created Album:\n${response.body}";
        });
      } else {
        setState(() {
          _responseText = "❌ Error: ${response.statusCode}\n${response.body}";
        });
      }
    } catch (e) {
      setState(() {
        _responseText = "❌ Exception: $e";
      });
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text("Create Album")),
      body: SingleChildScrollView(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          children: [
            TextField(
              controller: _idController,
              decoration: InputDecoration(labelText: "Album ID"),
              keyboardType: TextInputType.number,
            ),
            SizedBox(height: 10),
            TextField(
              controller: _userIdController,
              decoration: InputDecoration(labelText: "User ID"),
              keyboardType: TextInputType.number,
            ),
            SizedBox(height: 10),
            TextField(
              controller: _titleController,
              decoration: InputDecoration(labelText: "Title"),
            ),
            SizedBox(height: 20),
            ElevatedButton(
              onPressed: _createAlbum,
```



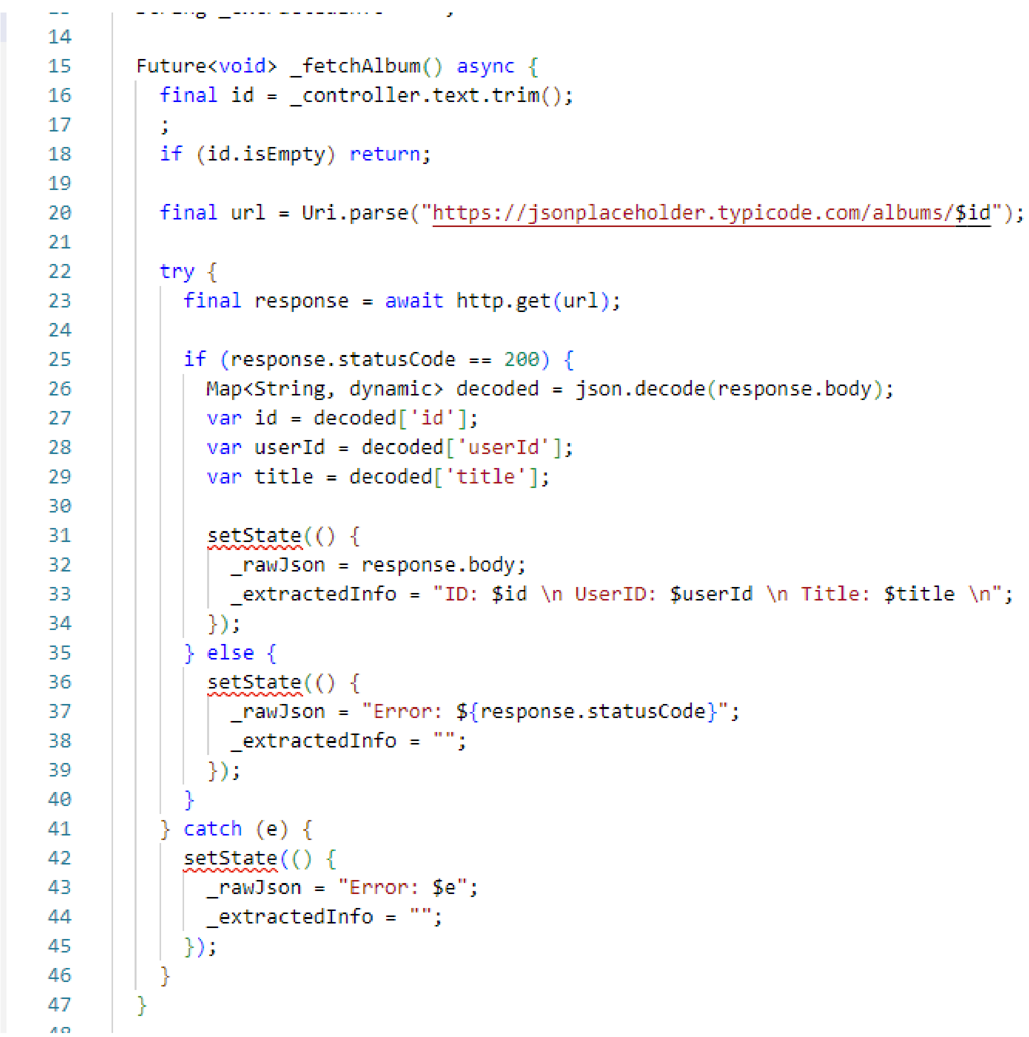
```
            ElevatedButton(
              onPressed: _createAlbum,
              child: Text("Create Album"),
            ),
            SizedBox(height: 20),
            Text(
              "Response:",
              style: TextStyle(fontWeight: FontWeight.bold),
            ),
            Container(
              width: double.infinity,
              padding: EdgeInsets.all(10),
              color: Colors.grey.shade100,
              child: SelectableText(_responseText),
            )
          ],
        ),
      ),
    );
  }
}

```



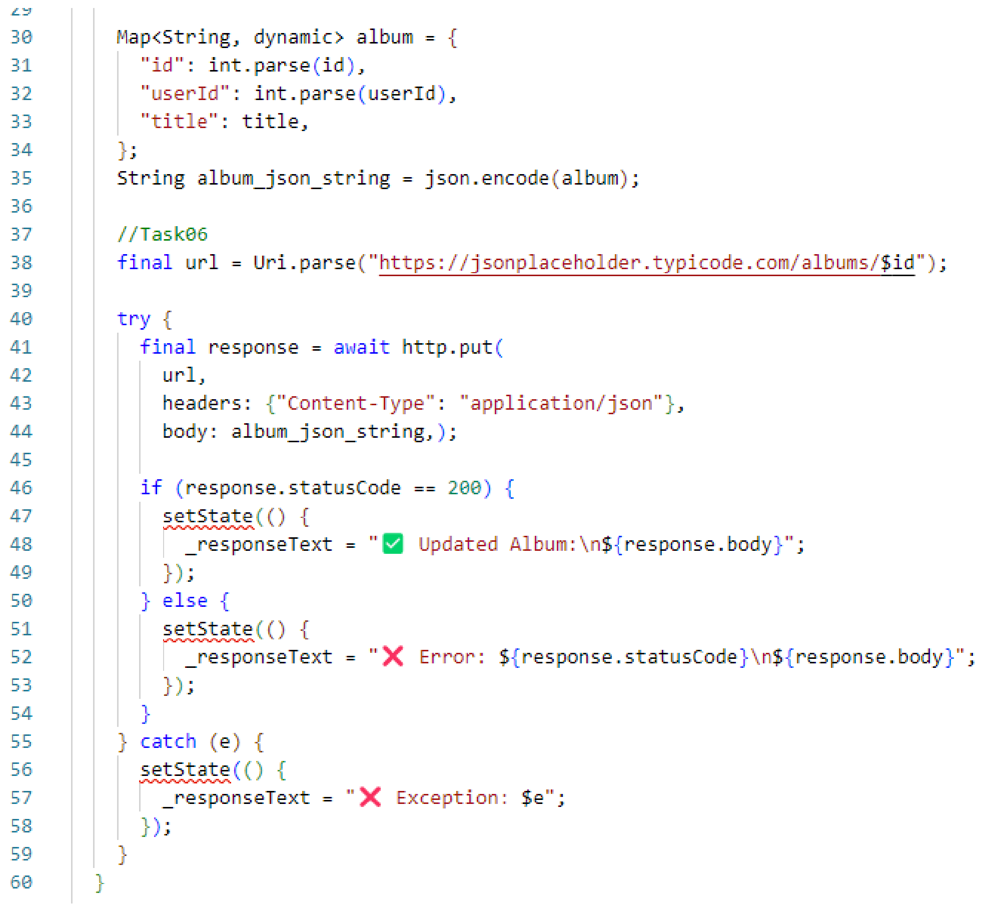
```
Future<void> _deleteAlbum() async {
  //Task07
  final id = _idController.text.trim();

  if (id.isEmpty) {
    setState(() {
      _responseText = " ❗ กรุณาใส่ ID ของอัลบั้มที่ต้องการลบ";
    });
    return;
  }

  final url = Uri.parse("https://jsonplaceholder.typicode.com/albums/$id");

  try {
    final response = await http.delete(url); //Task08

    if (response.statusCode == 200) {
      setState(() {
        _responseText = "✅ ลบอัลบั้ม ID $id สำเร็จ (สถานะ: 200)";
      });
    } else {
      setState(() {
        _responseText = "❌ ล้มเหลว: ${response.statusCode}";
      });
    }
  } catch (e) {
    setState(() {
      _responseText = "❌ ข้อผิดพลาด: $e";
    });
  }
}
```



```dart

import 'package:flutter/material.dart';

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text('Album CRUD Home')),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            ElevatedButton(
              onPressed: () => Navigator.pushNamed(context, '/create'),
              child: Text("Create Album"),
            ),
            ElevatedButton(
              onPressed: () => Navigator.pushNamed(context, '/read'),
              child: Text("Read Albums"),
            ),
            ElevatedButton(
              onPressed: () => Navigator.pushNamed(context, '/update'),
              child: Text("Update Album"),
            ),
            ElevatedButton(
              onPressed: () => Navigator.pushNamed(context, '/delete'),
              child: Text("Delete Album"),
            ),
          ],
        ),
      ),
    );
  }
}

```

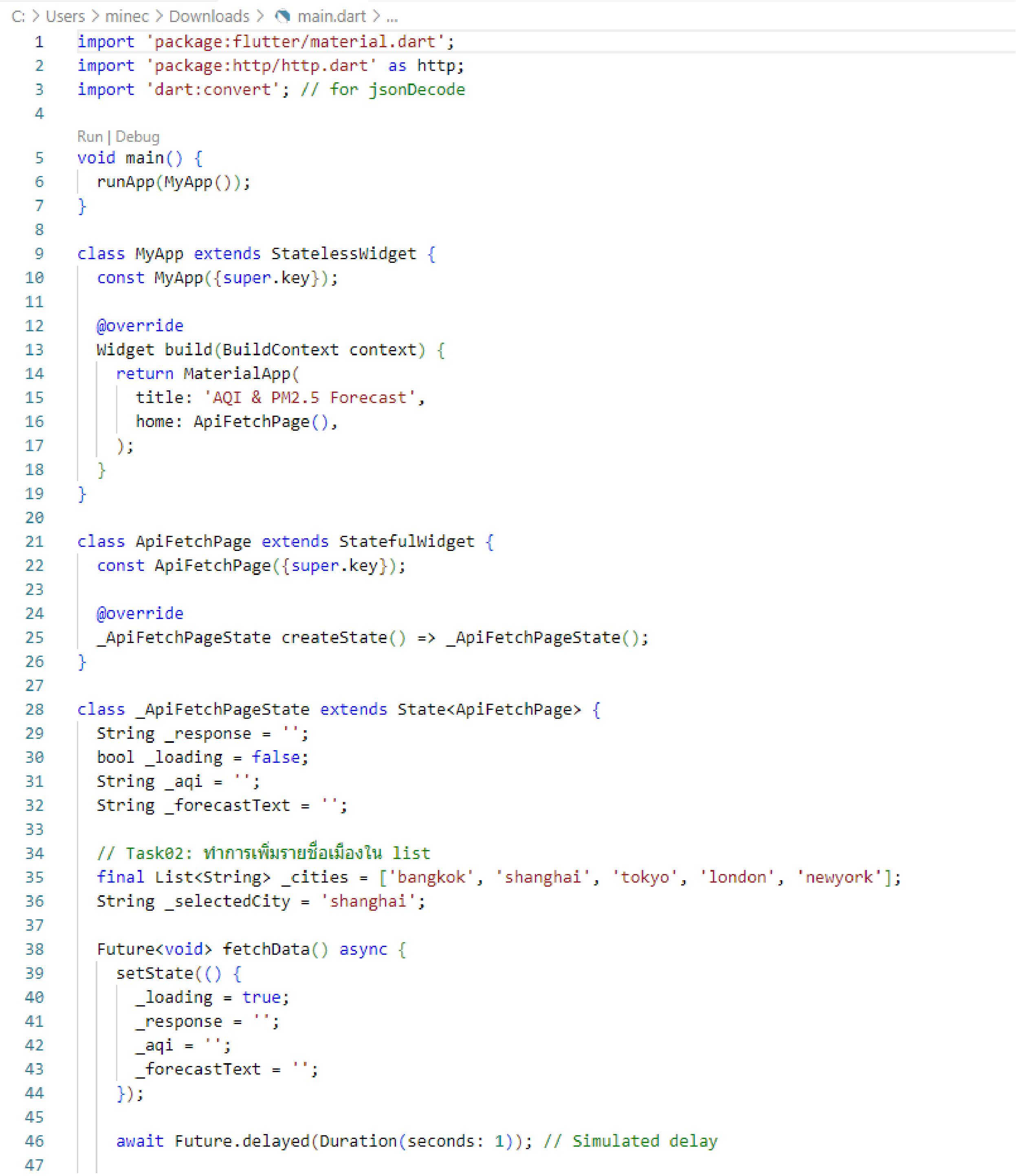


```
String _extractedInfo = "";

Future<void> _fetchAlbum() async {
  final id = _controller.text.trim();
  ;
  if (id.isEmpty) return;

  final url = Uri.parse("https://jsonplaceholder.typicode.com/albums/$id");

  try {
    final response = await http.get(url);

    if (response.statusCode == 200) {
      Map<String, dynamic> decoded = json.decode(response.body);
      var id = decoded['id'];
      var userId = decoded['userId'];
      var title = decoded['title'];

      setState(() {
        _rawJson = response.body;
        _extractedInfo = "ID: $id \n UserID: $userId \n Title: $title \n";
      });
    } else {
      setState(() {
        _rawJson = "Error: ${response.statusCode}";
        _extractedInfo = "";
      });
    }
  } catch (e) {
    setState(() {
      _rawJson = "Error: $e";
      _extractedInfo = "";
    });
  }
}
```

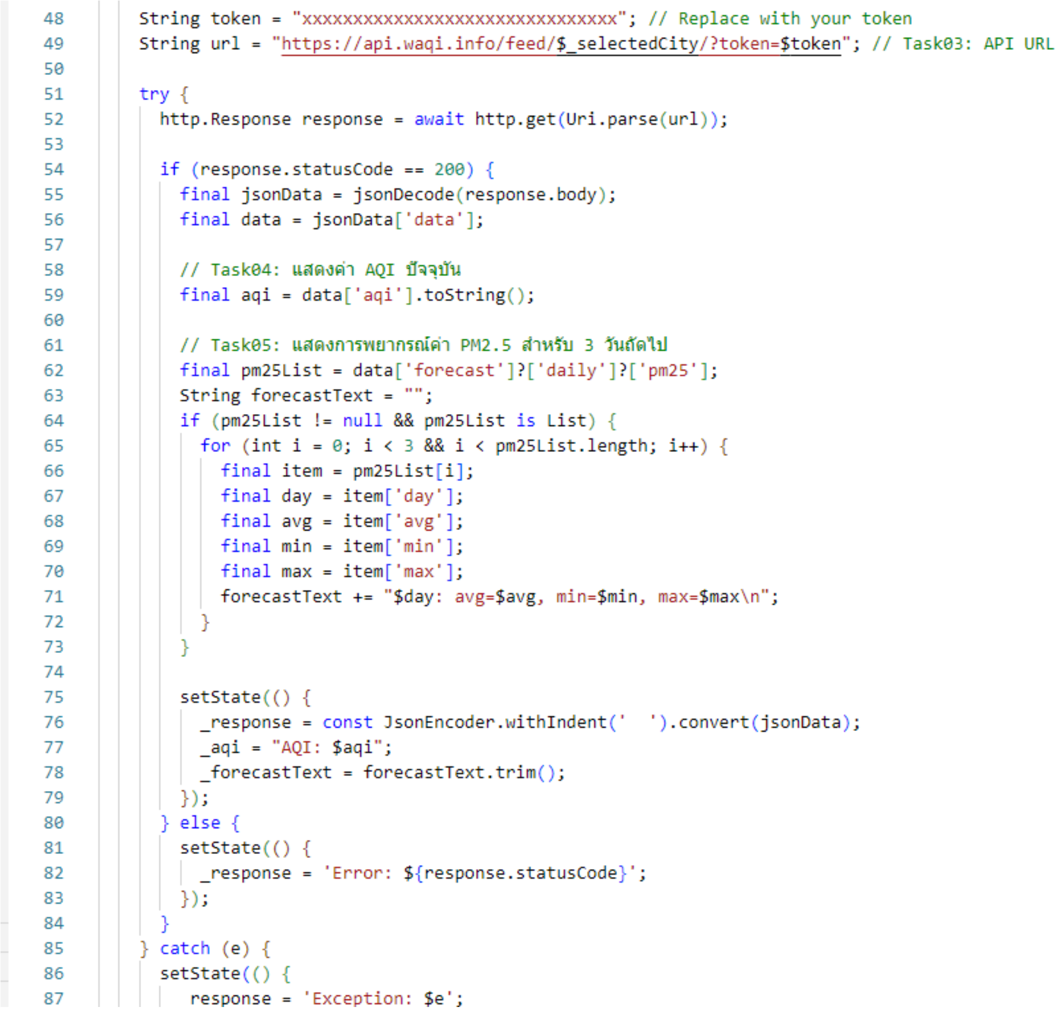


```
    Map<String, dynamic> album = {
      "id": int.parse(id),
      "userId": int.parse(userId),
      "title": title,
    };
    String album_json_string = json.encode(album);

    //Task06
    final url = Uri.parse("https://jsonplaceholder.typicode.com/albums/$id");

    try {
      final response = await http.put(
        url,
        headers: {"Content-Type": "application/json"},
        body: album_json_string,);

      if (response.statusCode == 200) {
        setState(() {
          _responseText = "✅ Updated Album:\n${response.body}";
        });
      } else {
        setState(() {
          _responseText = "❌ Error: ${response.statusCode}\n${response.body}";
        });
      }
    } catch (e) {
      setState(() {
        _responseText = "❌ Exception: $e";
      });
    }
  }
```

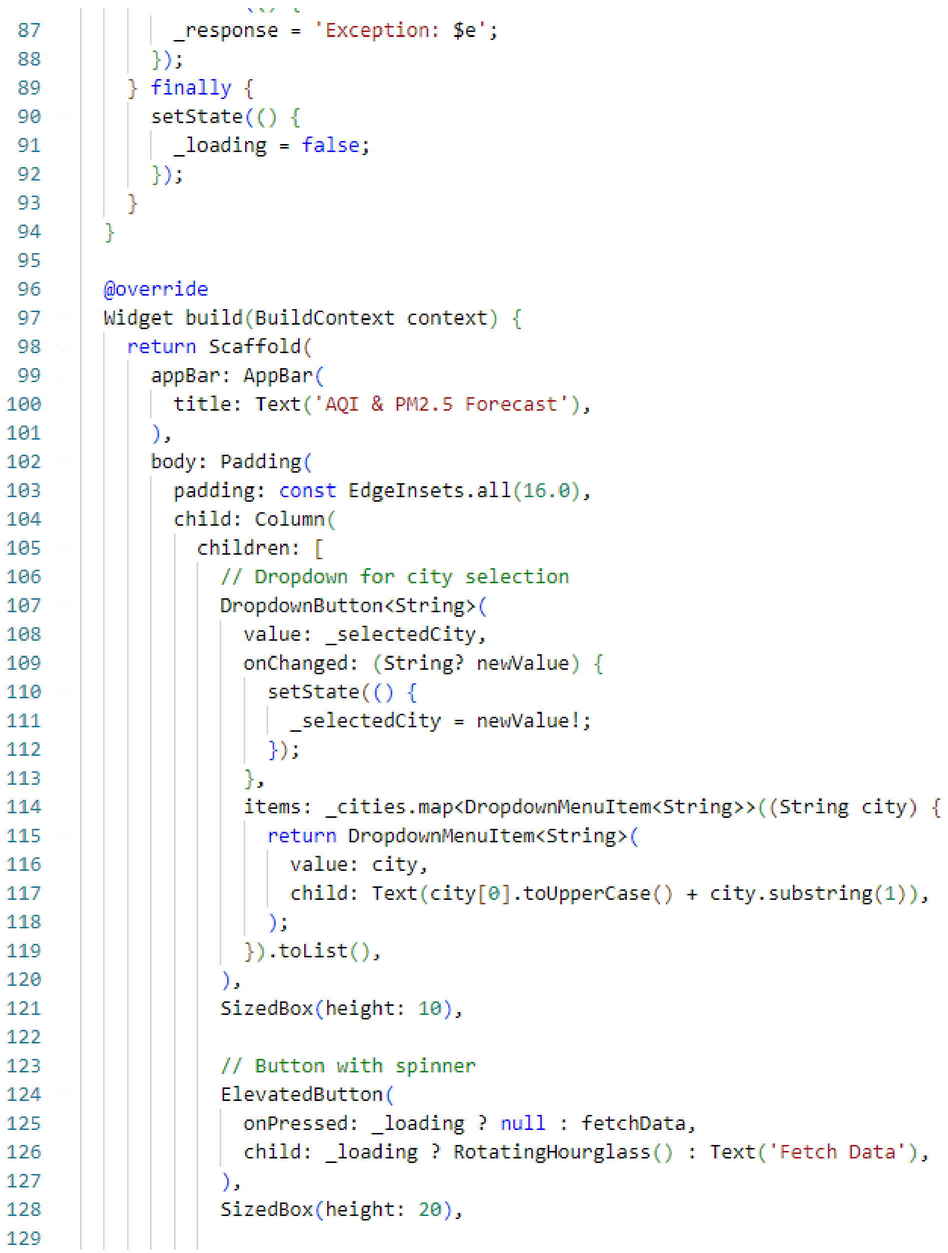
L11_ Challenge



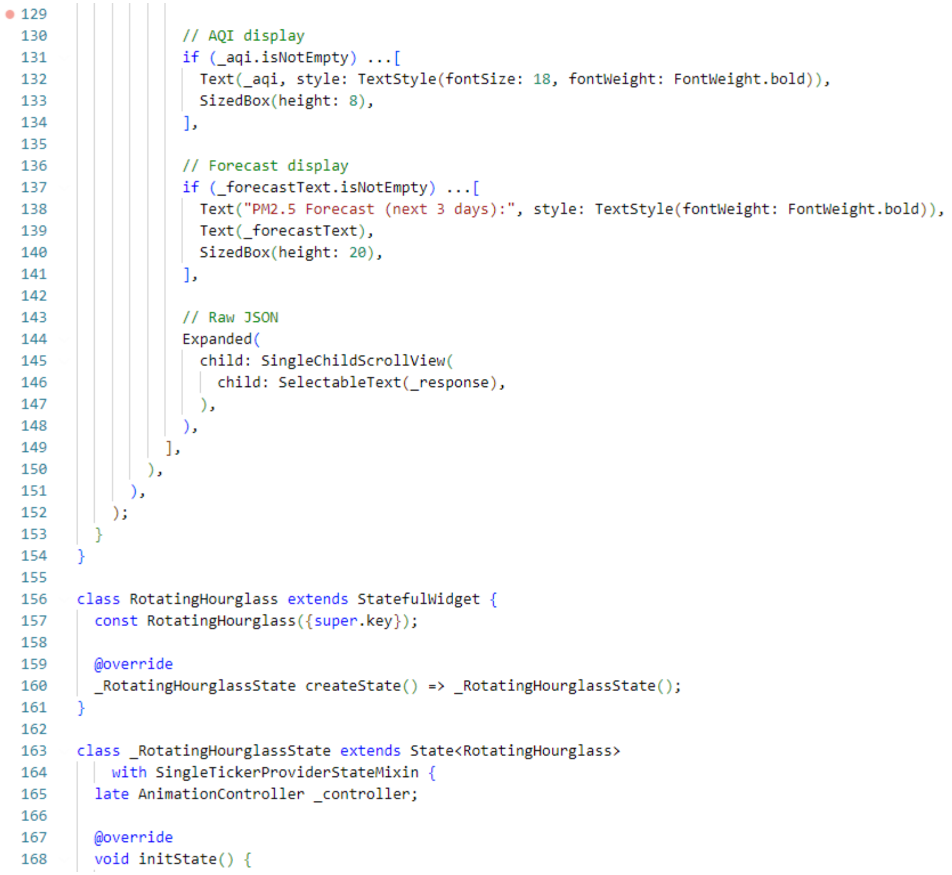
```dart
import 'package:flutter/material.dart';
import 'package:http/http.dart' as http;
import 'dart:convert'; // for jsonDecode

void main() {
  runApp(MyApp());
}

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

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'AQI & PM2.5 Forecast',
      home: ApiFetchPage(),
    );
  }
}

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

  @override
  _ApiFetchPageState createState() => _ApiFetchPageState();
}

class _ApiFetchPageState extends State<ApiFetchPage> {
  String _response = '';
  bool _loading = false;
  String _aqi = '';
  String _forecastText = '';

  // Task02: ทำการเพิ่มรายชื่อเมืองใน list
  final List<String> _cities = ['bangkok', 'shanghai', 'tokyo', 'london', 'newyork'];
  String _selectedCity = 'shanghai';

  Future<void> fetchData() async {
    setState(() {
      _loading = true;
      _response = '';
      _aqi = '';
      _forecastText = '';
    });

    await Future.delayed(Duration(seconds: 1)); // Simulated delay

```



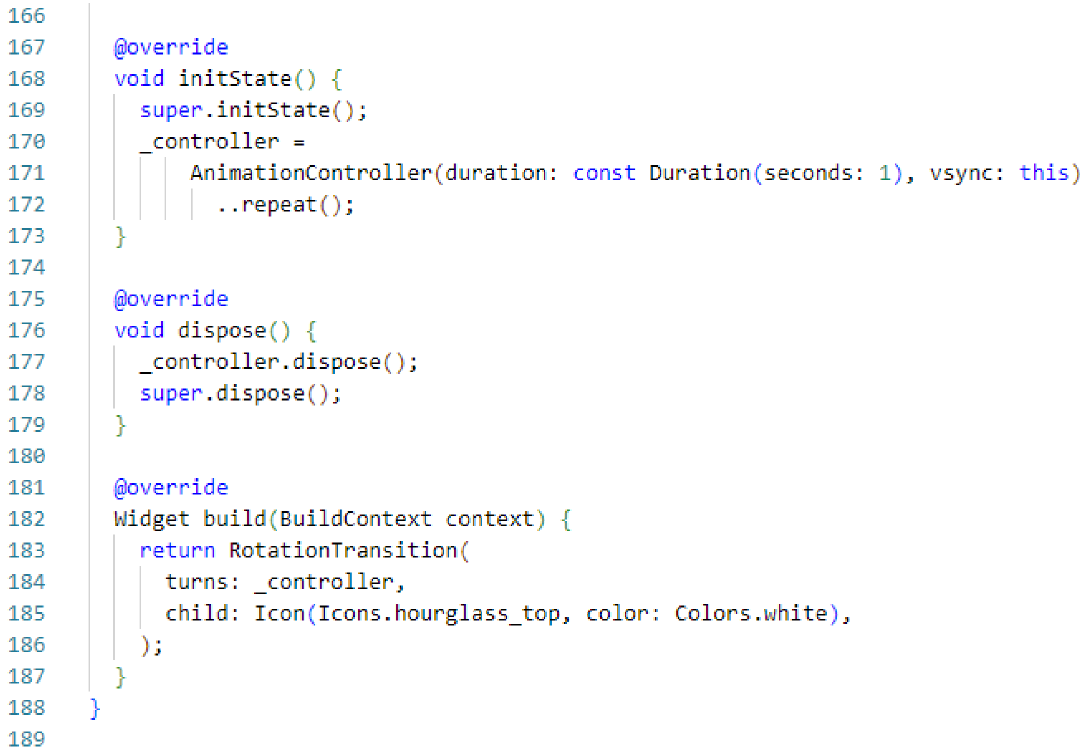
```
    String token = "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"; // Replace with your token
    String url = "https://api.waqi.info/feed/$_selectedCity/?token=$token"; // Task03: API URL

    try {
      http.Response response = await http.get(Uri.parse(url));

      if (response.statusCode == 200) {
        final jsonData = jsonDecode(response.body);
        final data = jsonData['data'];

        // Task04: แสดงค่า AQI ปัจจุบัน
        final aqi = data['aqi'].toString();

        // Task05: แสดงการพยากรณ์ค่า PM2.5 สำหรับ 3 วันถัดไป
        final pm25List = data['forecast']?['daily']?['pm25'];
        String forecastText = "";
        if (pm25List != null && pm25List is List) {
          for (int i = 0; i < 3 && i < pm25List.length; i++) {
            final item = pm25List[i];
            final day = item['day'];
            final avg = item['avg'];
            final min = item['min'];
            final max = item['max'];
            forecastText += "$day: avg=$avg, min=$min, max=$max\n";
          }
        }

        setState(() {
          _response = const JsonEncoder.withIndent('  ').convert(jsonData);
          _aqi = "AQI: $aqi";
          _forecastText = forecastText.trim();
        });
      } else {
        setState(() {
          _response = 'Error: ${response.statusCode}';
        });
      }
    } catch (e) {
      setState(() {
         response = 'Exception: $e';
```



```
        _response = 'Exception: $e';
      });
    } finally {
      setState(() {
        _loading = false;
      });
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('AQI & PM2.5 Forecast'),
      ),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          children: [
            // Dropdown for city selection
            DropdownButton<String>(
              value: _selectedCity,
              onChanged: (String? newValue) {
                setState(() {
                  _selectedCity = newValue!;
                });
              },
              items: _cities.map<DropdownMenuItem<String>>((String city) {
                return DropdownMenuItem<String>(
                  value: city,
                  child: Text(city[0].toUpperCase() + city.substring(1)),
                );
              }).toList(),
            ),
            SizedBox(height: 10),

            // Button with spinner
            ElevatedButton(
              onPressed: _loading ? null : fetchData,
              child: _loading ? RotatingHourglass() : Text('Fetch Data'),
            ),
            SizedBox(height: 20),

```



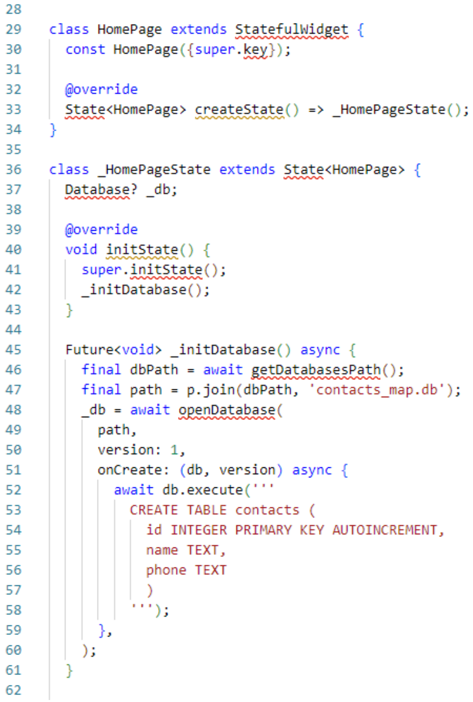
```dart

            // AQI display
            if (_aqi.isNotEmpty) ...[
              Text(_aqi, style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold)),
              SizedBox(height: 8),
            ],

            // Forecast display
            if (_forecastText.isNotEmpty) ...[
              Text("PM2.5 Forecast (next 3 days):", style: TextStyle(fontWeight: FontWeight.bold)),
              Text(_forecastText),
              SizedBox(height: 20),
            ],

            // Raw JSON
            Expanded(
              child: SingleChildScrollView(
                child: SelectableText(_response),
              ),
            ),
          ],
        ),
      ),
    );
  }
}

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

  @override
  _RotatingHourglassState createState() => _RotatingHourglassState();
}

class _RotatingHourglassState extends State<RotatingHourglass>
    with SingleTickerProviderStateMixin {
  late AnimationController _controller;

  @override
  void initState() {
```



```dart

  @override
  void initState() {
    super.initState();
    _controller =
        AnimationController(duration: const Duration(seconds: 1), vsync: this)
          ..repeat();
  }

  @override
  void dispose() {
    _controller.dispose();
    super.dispose();
  }

  @override
  Widget build(BuildContext context) {
    return RotationTransition(
      turns: _controller,
      child: Icon(Icons.hourglass_top, color: Colors.white),
    );
  }
}

```

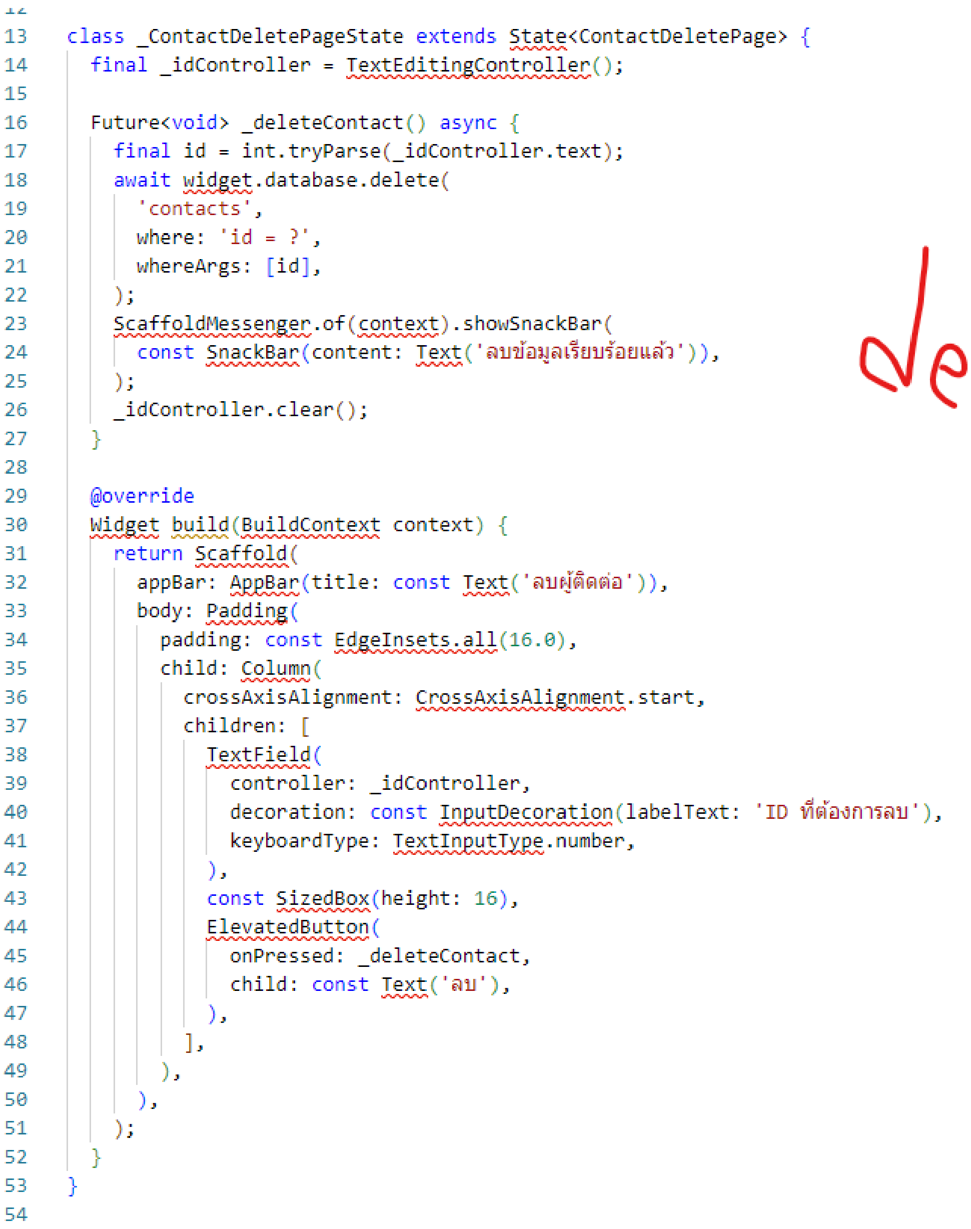
L_12 **Flutter and SQLite**

Q_1



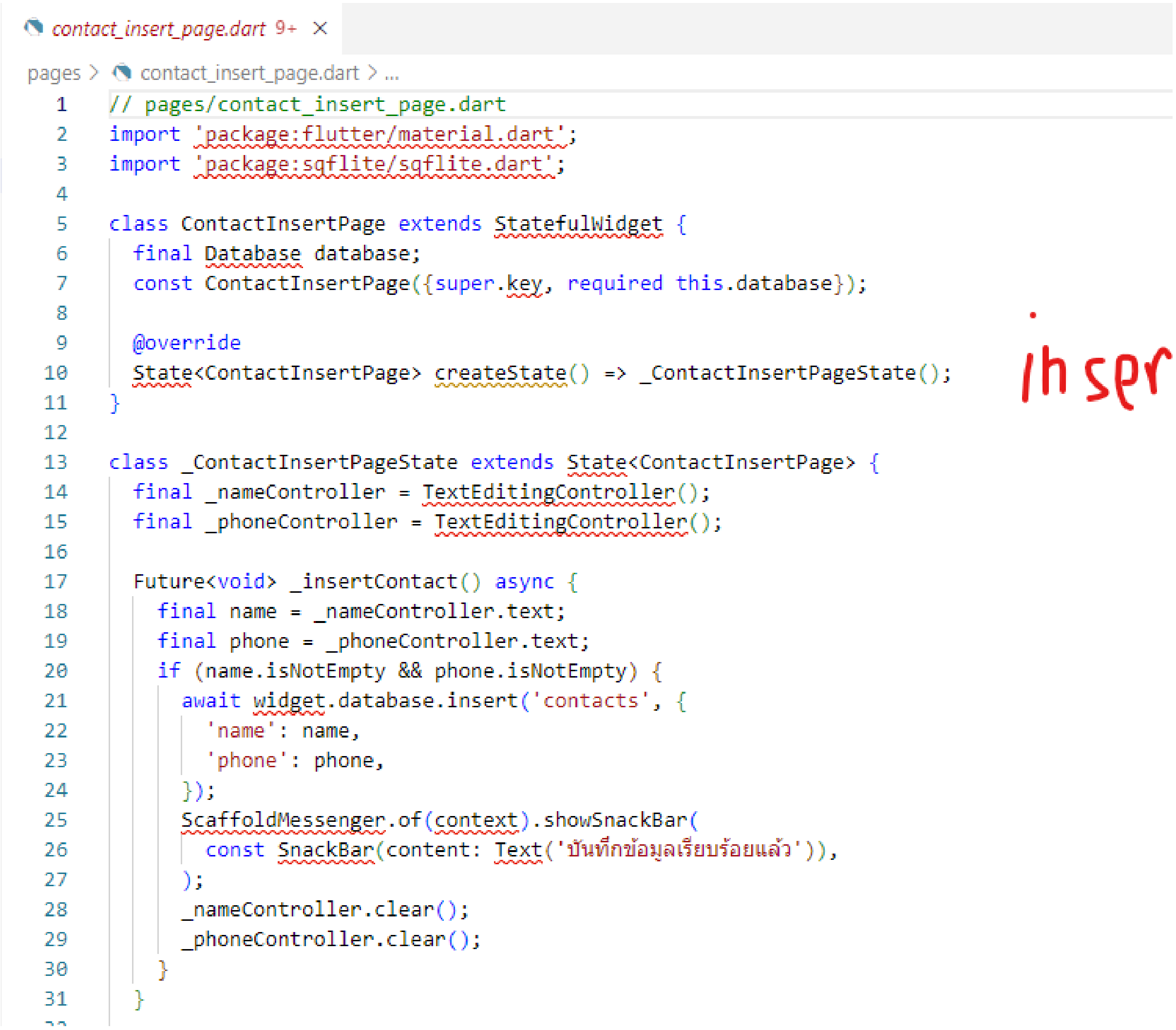
```dart

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

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

class _HomePageState extends State<HomePage> {
  Database? _db;

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

  Future<void> _initDatabase() async {
    final dbPath = await getDatabasesPath();
    final path = p.join(dbPath, 'contacts_map.db');
    _db = await openDatabase(
      path,
      version: 1,
      onCreate: (db, version) async {
        await db.execute('''
          CREATE TABLE contacts (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            name TEXT,
            phone TEXT
            )
          ''');
      },
    );
  }

```

Main



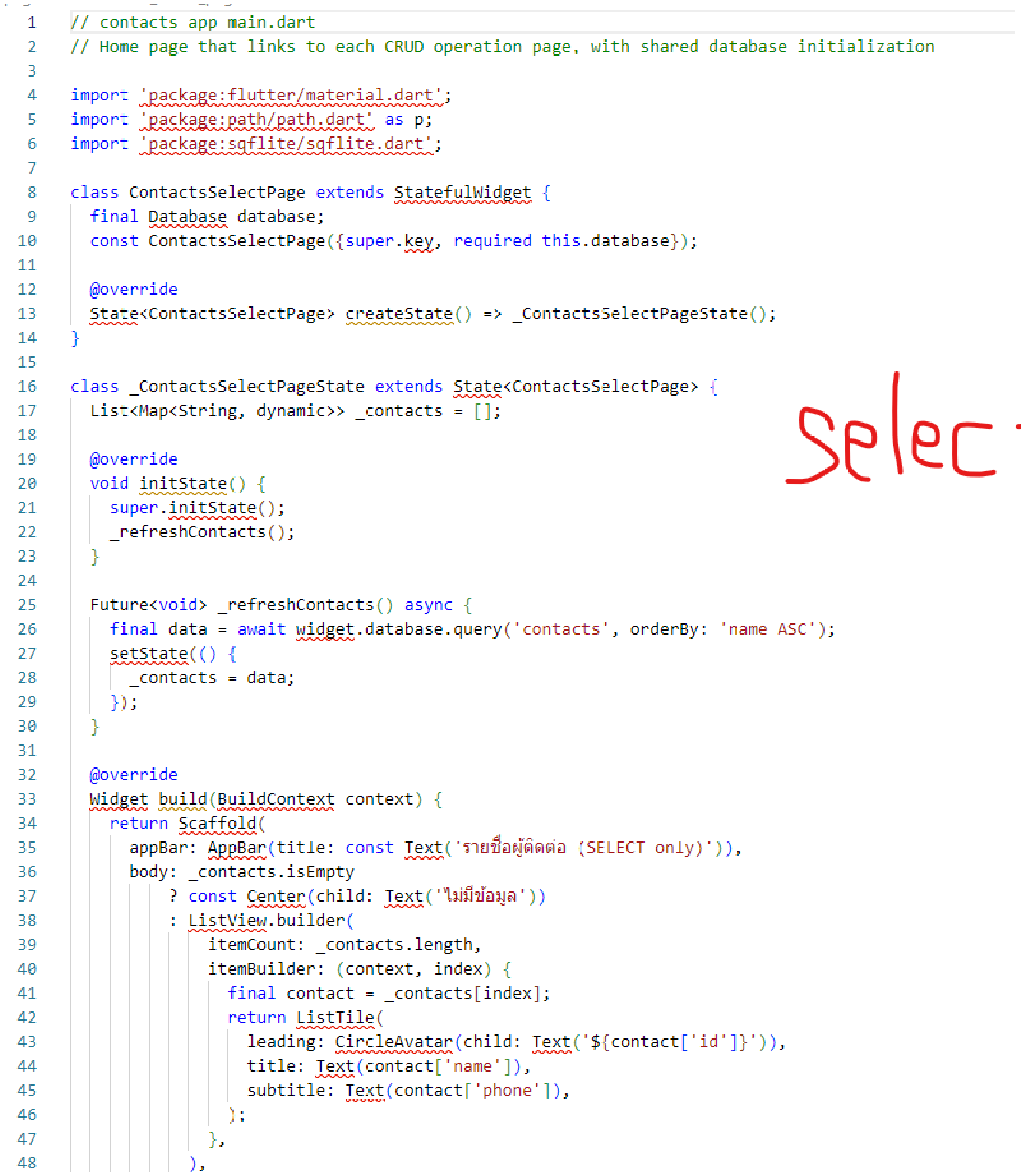
```dart
class _ContactDeletePageState extends State<ContactDeletePage> {
  final _idController = TextEditingController();

  Future<void> _deleteContact() async {
    final id = int.tryParse(_idController.text);
    await widget.database.delete(
      'contacts',
      where: 'id = ?',
      whereArgs: [id],
    );
    ScaffoldMessenger.of(context).showSnackBar(
      const SnackBar(content: Text('ลบข้อมูลเรียบร้อยแล้ว')),
    );
    _idController.clear();
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: const Text('ลบผู้ติดต่อ')),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            TextField(
              controller: _idController,
              decoration: const InputDecoration(labelText: 'ID ที่ต้องการลบ'),
              keyboardType: TextInputType.number,
            ),
            const SizedBox(height: 16),
            ElevatedButton(
              onPressed: _deleteContact,
              child: const Text('ลบ'),
            ),
          ],
        ),
      ),
    );
  }
}
```

del


contact_insert_page.dart 9+ ×
pages > contact_insert_page.dart > ...

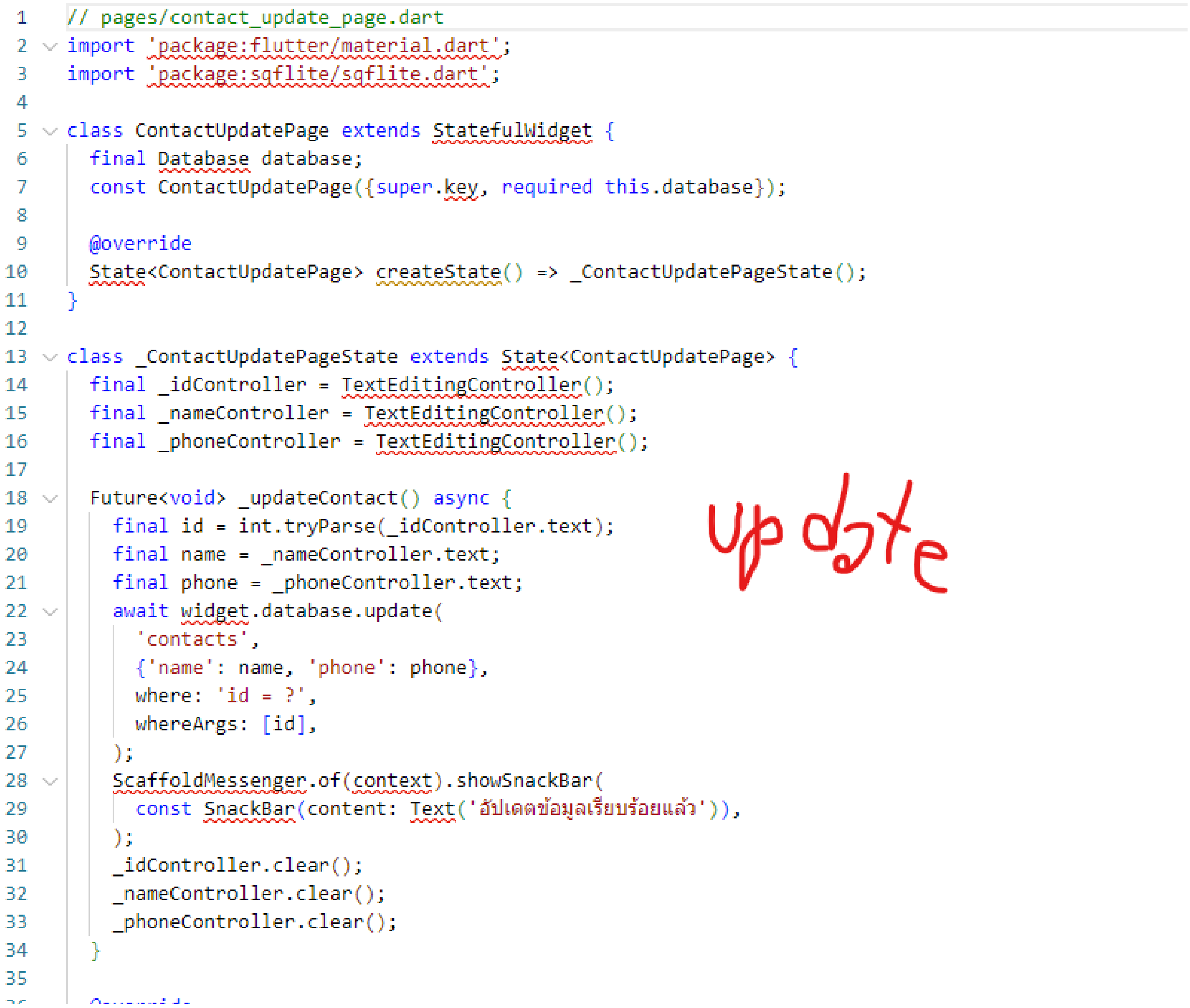
```dart
// pages/contact_insert_page.dart
import 'package:flutter/material.dart';
import 'package:sqflite/sqflite.dart';

class ContactInsertPage extends StatefulWidget {
  final Database database;
  const ContactInsertPage({super.key, required this.database});

  @override
  State<ContactInsertPage> createState() => _ContactInsertPageState();
}

class _ContactInsertPageState extends State<ContactInsertPage> {
  final _nameController = TextEditingController();
  final _phoneController = TextEditingController();

  Future<void> _insertContact() async {
    final name = _nameController.text;
    final phone = _phoneController.text;
    if (name.isNotEmpty && phone.isNotEmpty) {
      await widget.database.insert('contacts', {
        'name': name,
        'phone': phone,
      });
      ScaffoldMessenger.of(context).showSnackBar(
        const SnackBar(content: Text('บันทึกข้อมูลเรียบร้อยแล้ว')),
      );
      _nameController.clear();
      _phoneController.clear();
    }
  }
```

insert



```dart
// contacts_app_main.dart
// Home page that links to each CRUD operation page, with shared database initialization

import 'package:flutter/material.dart';
import 'package:path/path.dart' as p;
import 'package:sqflite/sqflite.dart';

class ContactsSelectPage extends StatefulWidget {
  final Database database;
  const ContactsSelectPage({super.key, required this.database});

  @override
  State<ContactsSelectPage> createState() => _ContactsSelectPageState();
}

class _ContactsSelectPageState extends State<ContactsSelectPage> {
  List<Map<String, dynamic>> _contacts = [];

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

  Future<void> _refreshContacts() async {
    final data = await widget.database.query('contacts', orderBy: 'name ASC');
    setState(() {
      _contacts = data;
    });
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: const Text('รายชื่อผู้ติดต่อ (SELECT only)')),
      body: _contacts.isEmpty
          ? const Center(child: Text('ไม่มีข้อมูล'))
          : ListView.builder(
              itemCount: _contacts.length,
              itemBuilder: (context, index) {
                final contact = _contacts[index];
                return ListTile(
                  leading: CircleAvatar(child: Text('${contact['id']}')),
                  title: Text(contact['name']),
                  subtitle: Text(contact['phone']),
                );
              },
            ),
```

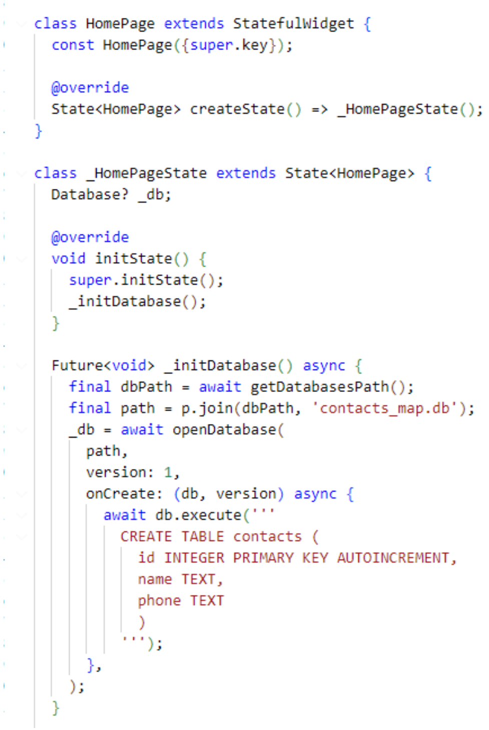
select



```dart
// pages/contact_update_page.dart
import 'package:flutter/material.dart';
import 'package:sqflite/sqflite.dart';

class ContactUpdatePage extends StatefulWidget {
  final Database database;
  const ContactUpdatePage({super.key, required this.database});

  @override
  State<ContactUpdatePage> createState() => _ContactUpdatePageState();
}

class _ContactUpdatePageState extends State<ContactUpdatePage> {
  final _idController = TextEditingController();
  final _nameController = TextEditingController();
  final _phoneController = TextEditingController();

  Future<void> _updateContact() async {
    final id = int.tryParse(_idController.text);
    final name = _nameController.text;
    final phone = _phoneController.text;
    await widget.database.update(
      'contacts',
      {'name': name, 'phone': phone},
      where: 'id = ?',
      whereArgs: [id],
    );
    ScaffoldMessenger.of(context).showSnackBar(
      const SnackBar(content: Text('อัปเดตข้อมูลเรียบร้อยแล้ว')),
    );
    _idController.clear();
    _nameController.clear();
    _phoneController.clear();
  }

  @override
```

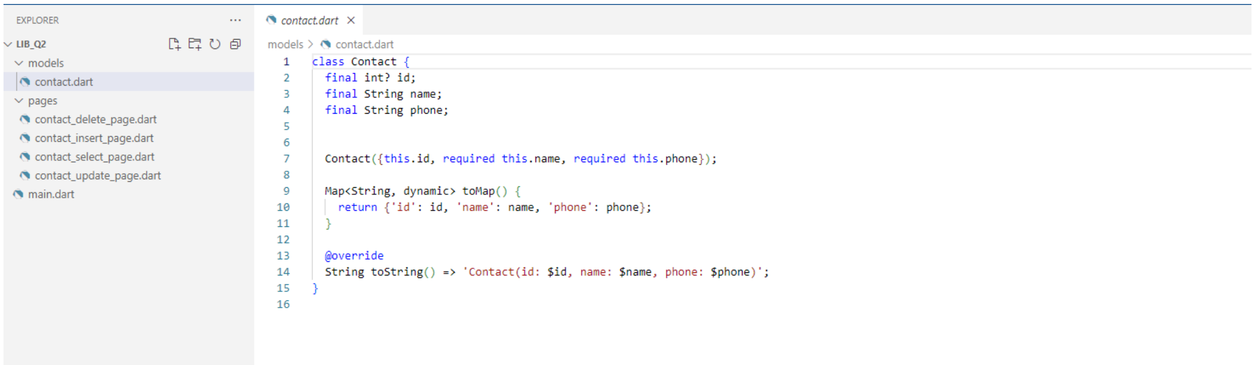
update

Q_2

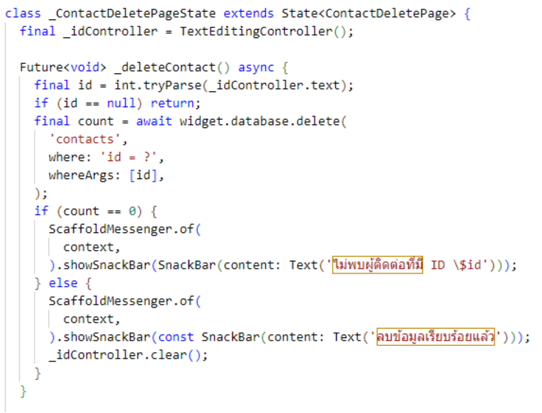


```dart
class HomePage extends StatefulWidget {
  const HomePage({super.key});

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

class _HomePageState extends State<HomePage> {
  Database? _db;

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

  Future<void> _initDatabase() async {
    final dbPath = await getDatabasesPath();
    final path = p.join(dbPath, 'contacts_map.db');
    _db = await openDatabase(
      path,
      version: 1,
      onCreate: (db, version) async {
        await db.execute('''
          CREATE TABLE contacts (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            name TEXT,
            phone TEXT
            )
          ''');
      },
    );
  }
```

Main

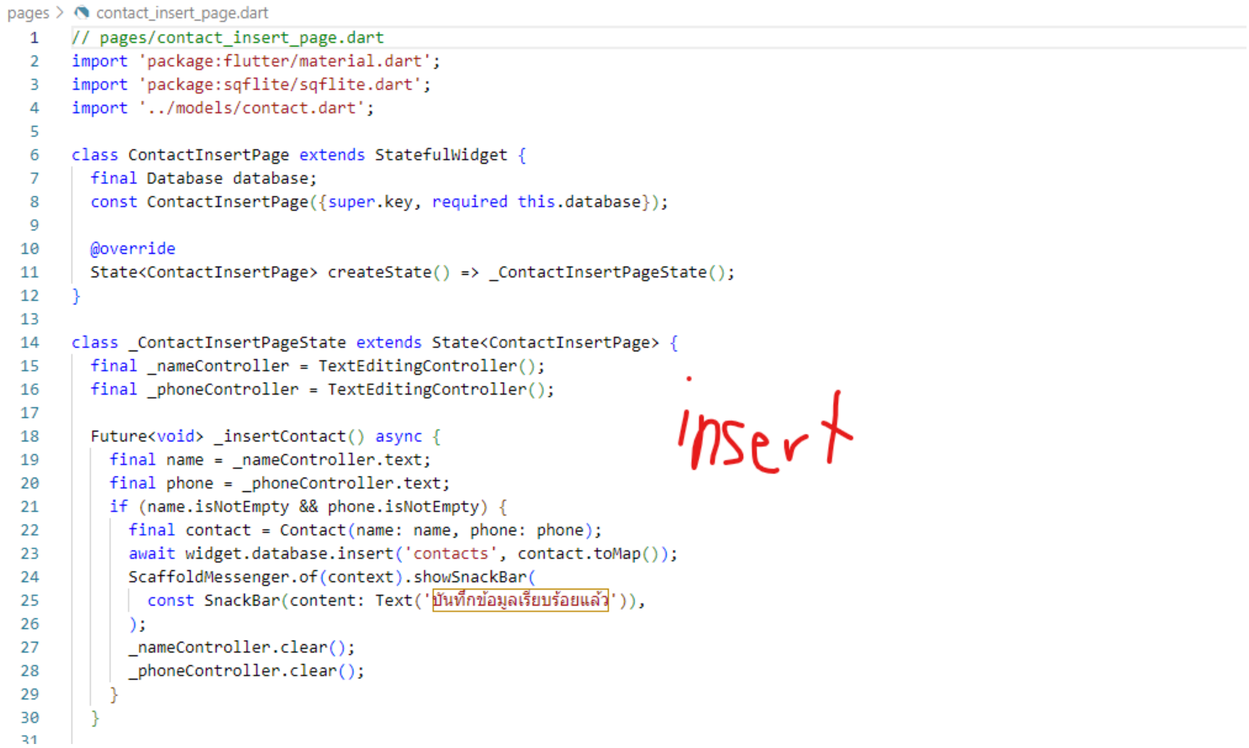


```dart
class Contact {
  final int? id;
  final String name;
  final String phone;


  Contact({this.id, required this.name, required this.phone});

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

  @override
  String toString() => 'Contact(id: $id, name: $name, phone: $phone)';
}

```



```dart
class _ContactDeletePageState extends State<ContactDeletePage> {
  final _idController = TextEditingController();

  Future<void> _deleteContact() async {
    final id = int.tryParse(_idController.text);
    if (id == null) return;
    final count = await widget.database.delete(
      'contacts',
      where: 'id = ?',
      whereArgs: [id],
    );
    if (count == 0) {
      ScaffoldMessenger.of(
        context,
      ).showSnackBar(SnackBar(content: Text('ไม่พบผู้ติดต่อที่มี ID \$id')));
    } else {
      ScaffoldMessenger.of(
        context,
      ).showSnackBar(const SnackBar(content: Text('ลบข้อมูลเรียบร้อยแล้ว')));
      _idController.clear();
    }
  }
```

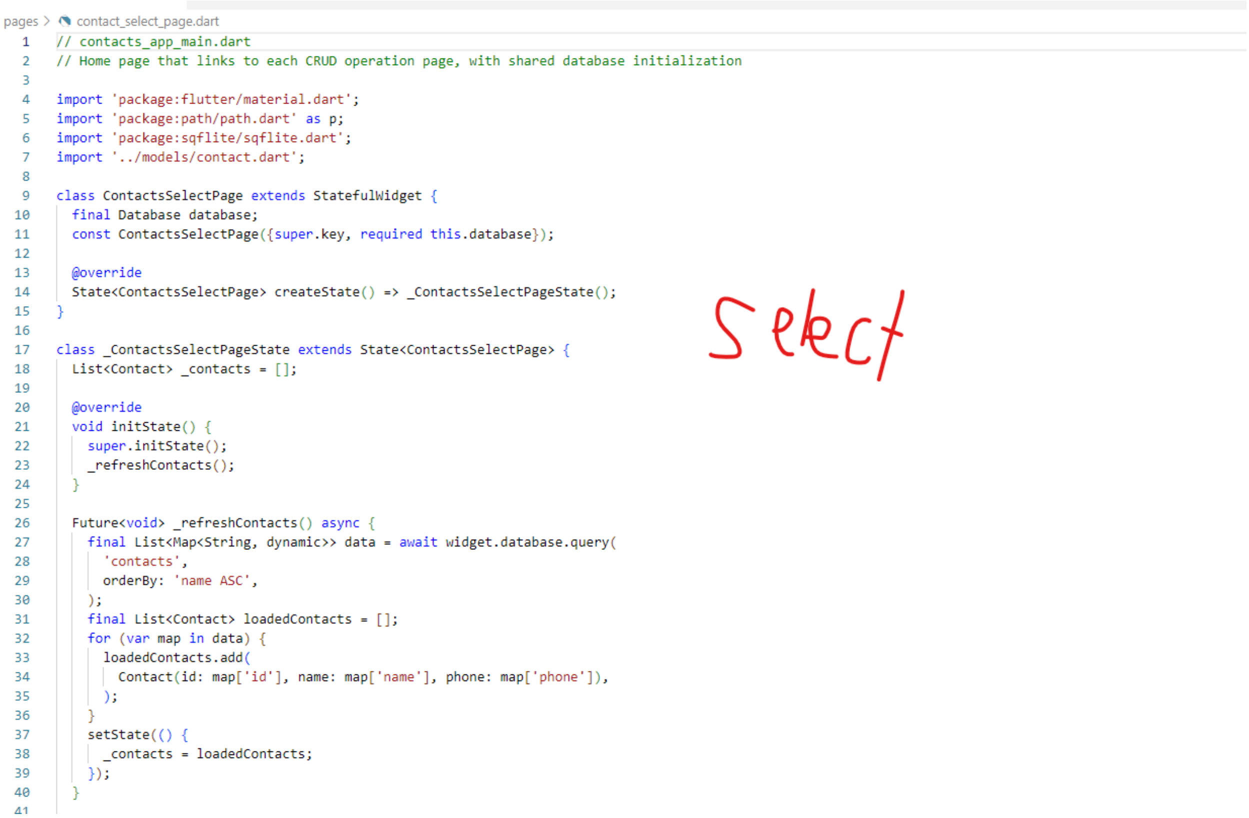
del



pages > contact_insert_page.dart

```dart
// pages/contact_insert_page.dart
import 'package:flutter/material.dart';
import 'package:sqflite/sqflite.dart';
import '../models/contact.dart';

class ContactInsertPage extends StatefulWidget {
  final Database database;
  const ContactInsertPage({super.key, required this.database});

  @override
  State<ContactInsertPage> createState() => _ContactInsertPageState();
}

class _ContactInsertPageState extends State<ContactInsertPage> {
  final _nameController = TextEditingController();
  final _phoneController = TextEditingController();

  Future<void> _insertContact() async {
    final name = _nameController.text;
    final phone = _phoneController.text;
    if (name.isNotEmpty && phone.isNotEmpty) {
      final contact = Contact(name: name, phone: phone);
      await widget.database.insert('contacts', contact.toMap());
      ScaffoldMessenger.of(context).showSnackBar(
        const SnackBar(content: Text('บันทึกข้อมูลเรียบร้อยแล้ว')),
      );
      _nameController.clear();
      _phoneController.clear();
    }
  }
```

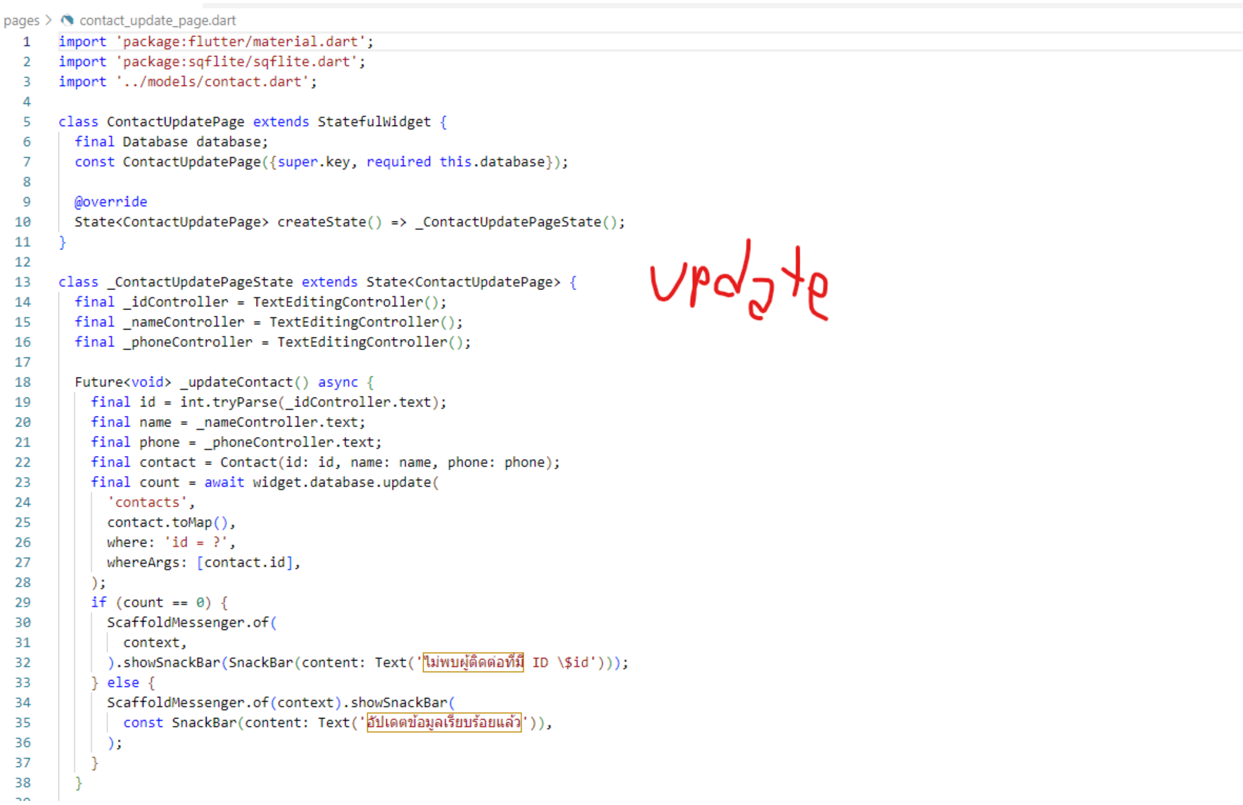
insert



pages > contact_select_page.dart

```dart
// contacts_app_main.dart
// Home page that links to each CRUD operation page, with shared database initialization

import 'package:flutter/material.dart';
import 'package:path/path.dart' as p;
import 'package:sqflite/sqflite.dart';
import '../models/contact.dart';

class ContactsSelectPage extends StatefulWidget {
  final Database database;
  const ContactsSelectPage({super.key, required this.database});

  @override
  State<ContactsSelectPage> createState() => _ContactsSelectPageState();
}

class _ContactsSelectPageState extends State<ContactsSelectPage> {
  List<Contact> _contacts = [];

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

  Future<void> _refreshContacts() async {
    final List<Map<String, dynamic>> data = await widget.database.query(
      'contacts',
      orderBy: 'name ASC',
    );
    final List<Contact> loadedContacts = [];
    for (var map in data) {
      loadedContacts.add(
        Contact(id: map['id'], name: map['name'], phone: map['phone']),
      );
    }
    setState(() {
      _contacts = loadedContacts;
    });
  }
```

Select

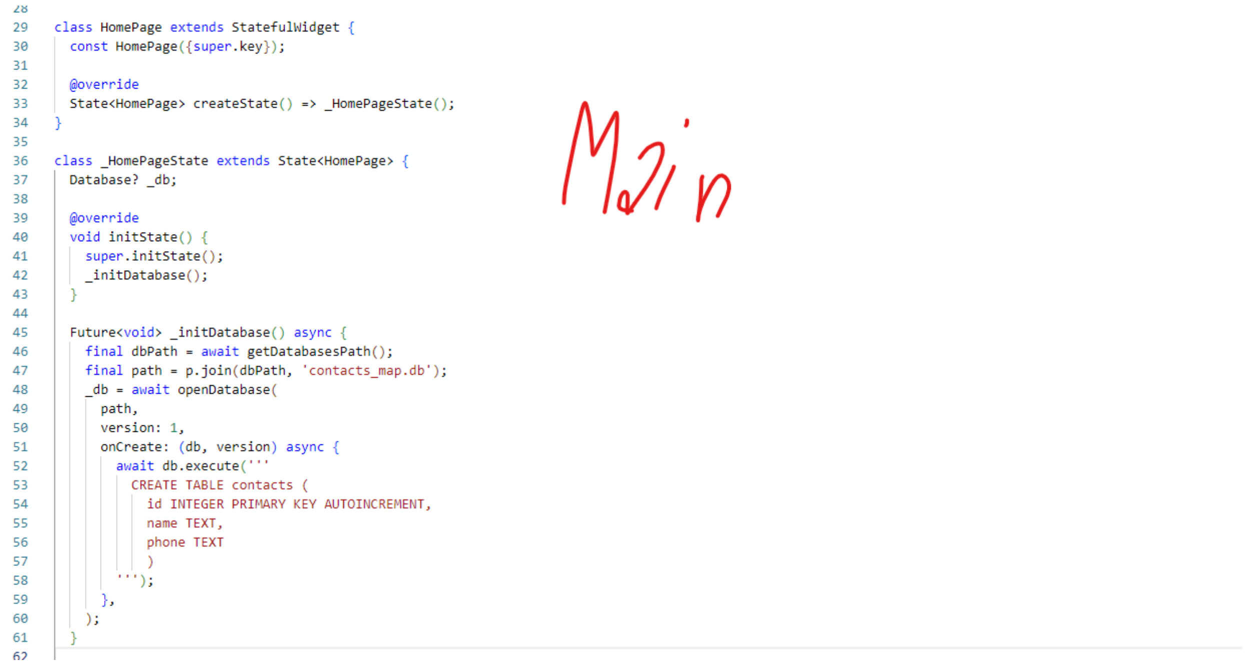


pages > contact_update_page.dart

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

class ContactUpdatePage extends StatefulWidget {
  final Database database;
  const ContactUpdatePage({super.key, required this.database});

  @override
  State<ContactUpdatePage> createState() => _ContactUpdatePageState();
}

class _ContactUpdatePageState extends State<ContactUpdatePage> {
  final _idController = TextEditingController();
  final _nameController = TextEditingController();
  final _phoneController = TextEditingController();

  Future<void> _updateContact() async {
    final id = int.tryParse(_idController.text);
    final name = _nameController.text;
    final phone = _phoneController.text;
    final contact = Contact(id: id, name: name, phone: phone);
    final count = await widget.database.update(
      'contacts',
      contact.toMap(),
      where: 'id = ?',
      whereArgs: [contact.id],
    );
    if (count == 0) {
      ScaffoldMessenger.of(
        context,
      ).showSnackBar(SnackBar(content: Text('ไม่พบผู้ติดต่อที่มี ID \$id')));
    } else {
      ScaffoldMessenger.of(context).showSnackBar(
        const SnackBar(content: Text('อัปเดตข้อมูลเรียบร้อยแล้ว')),
      );
    }
  }
```

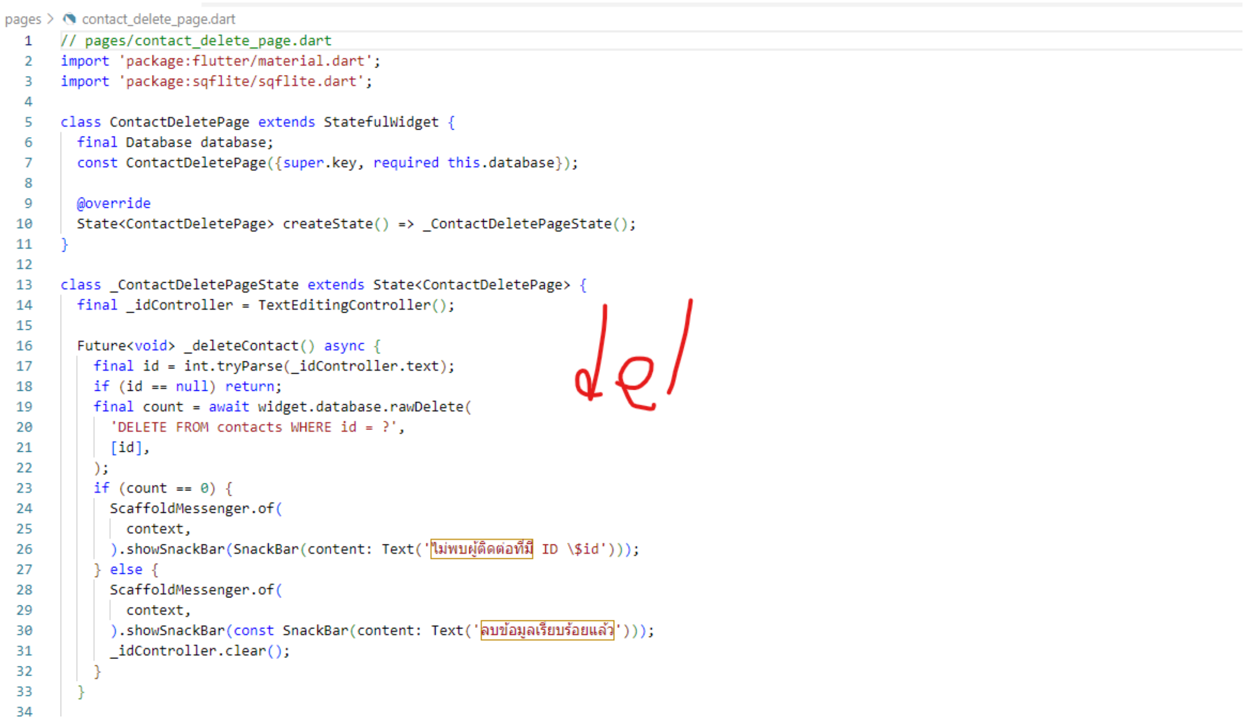
update

Q_3



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

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

class _HomePageState extends State<HomePage> {
  Database? _db;

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

  Future<void> _initDatabase() async {
    final dbPath = await getDatabasesPath();
    final path = p.join(dbPath, 'contacts_map.db');
    _db = await openDatabase(
      path,
      version: 1,
      onCreate: (db, version) async {
        await db.execute('''
          CREATE TABLE contacts (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            name TEXT,
            phone TEXT
            )
        ''');
      },
    );
  }

```

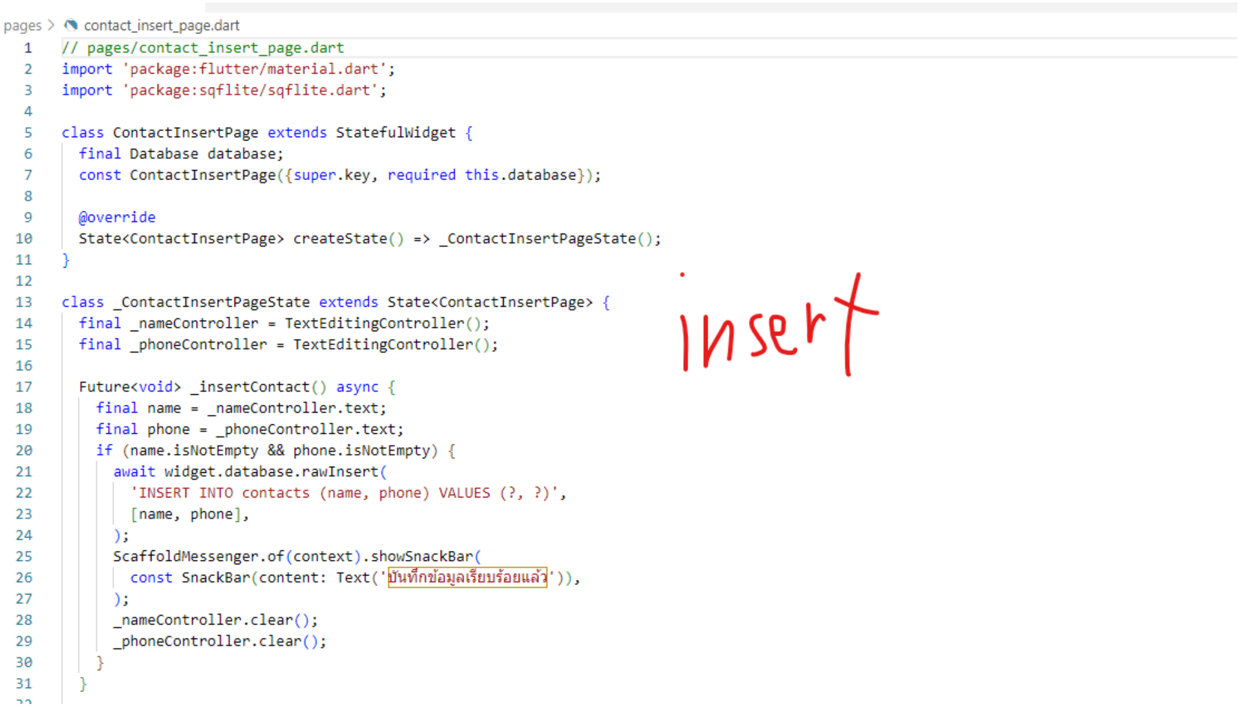
Main



pages > contact_delete_page.dart

```dart
// pages/contact_delete_page.dart
import 'package:flutter/material.dart';
import 'package:sqflite/sqflite.dart';

class ContactDeletePage extends StatefulWidget {
  final Database database;
  const ContactDeletePage({super.key, required this.database});

  @override
  State<ContactDeletePage> createState() => _ContactDeletePageState();
}

class _ContactDeletePageState extends State<ContactDeletePage> {
  final _idController = TextEditingController();

  Future<void> _deleteContact() async {
    final id = int.tryParse(_idController.text);
    if (id == null) return;
    final count = await widget.database.rawDelete(
      'DELETE FROM contacts WHERE id = ?',
      [id],
    );
    if (count == 0) {
      ScaffoldMessenger.of(
        context,
      ).showSnackBar(SnackBar(content: Text('ไม่พบผู้ติดต่อที่มี ID \$id')));
    } else {
      ScaffoldMessenger.of(
        context,
      ).showSnackBar(const SnackBar(content: Text('ลบข้อมูลเรียบร้อยแล้ว')));
      _idController.clear();
    }
  }

```

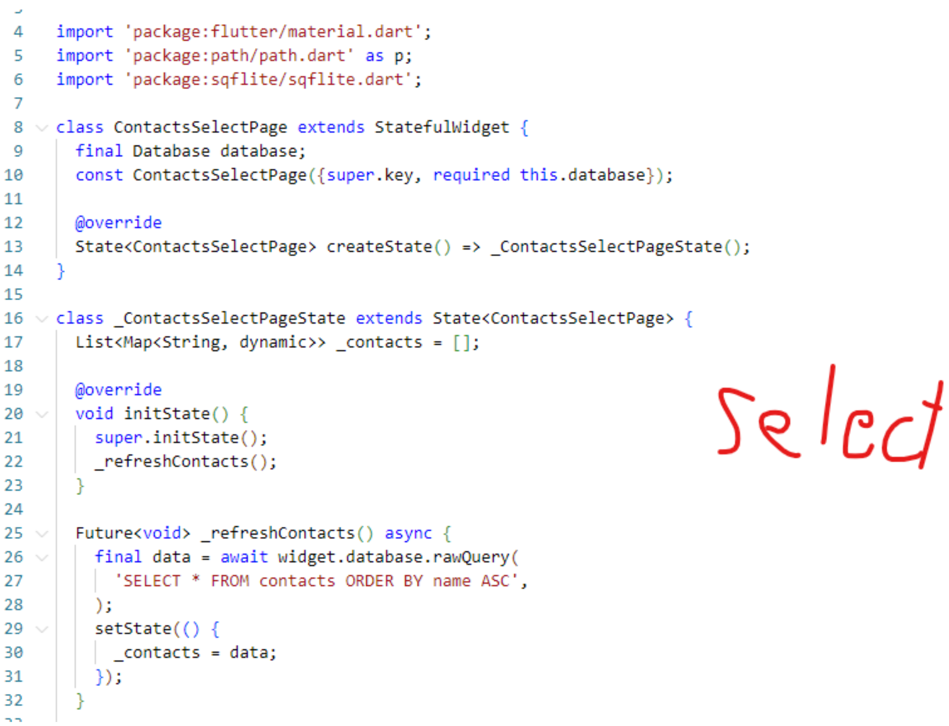
del



pages > contact_insert_page.dart

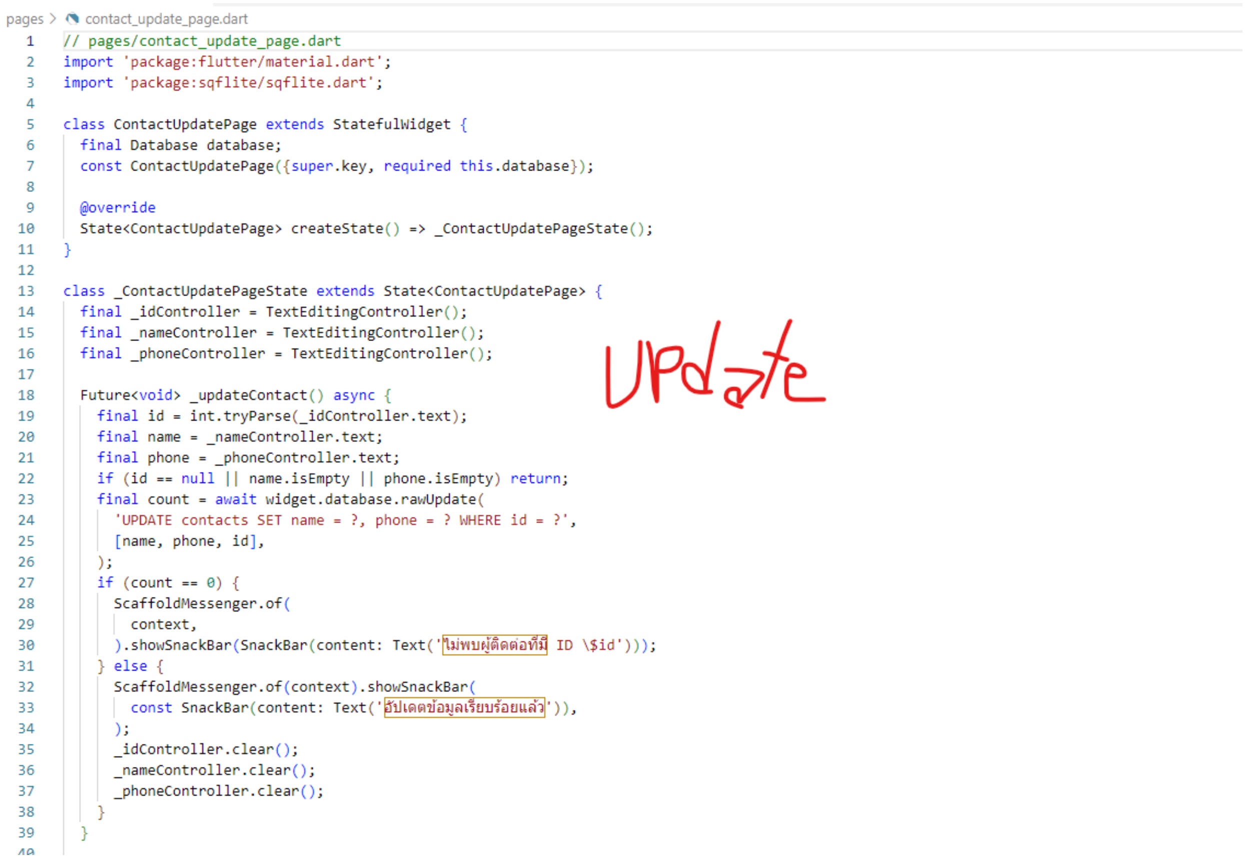
```dart
// pages/contact_insert_page.dart
import 'package:flutter/material.dart';
import 'package:sqflite/sqflite.dart';

class ContactInsertPage extends StatefulWidget {
  final Database database;
  const ContactInsertPage({super.key, required this.database});

  @override
  State<ContactInsertPage> createState() => _ContactInsertPageState();
}

class _ContactInsertPageState extends State<ContactInsertPage> {
  final _nameController = TextEditingController();
  final _phoneController = TextEditingController();

  Future<void> _insertContact() async {
    final name = _nameController.text;
    final phone = _phoneController.text;
    if (name.isNotEmpty && phone.isNotEmpty) {
      await widget.database.rawInsert(
        'INSERT INTO contacts (name, phone) VALUES (?, ?)',
        [name, phone],
      );
      ScaffoldMessenger.of(context).showSnackBar(
        const SnackBar(content: Text('บันทึกข้อมูลเรียบร้อยแล้ว')),
      );
      _nameController.clear();
      _phoneController.clear();
    }
  }
```

insert



```dart
import 'package:flutter/material.dart';
import 'package:path/path.dart' as p;
import 'package:sqflite/sqflite.dart';

class ContactsSelectPage extends StatefulWidget {
  final Database database;
  const ContactsSelectPage({super.key, required this.database});

  @override
  State<ContactsSelectPage> createState() => _ContactsSelectPageState();
}

class _ContactsSelectPageState extends State<ContactsSelectPage> {
  List<Map<String, dynamic>> _contacts = [];

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

  Future<void> _refreshContacts() async {
    final data = await widget.database.rawQuery(
      'SELECT * FROM contacts ORDER BY name ASC',
    );
    setState(() {
      _contacts = data;
    });
  }
```

Select



pages > contact_update_page.dart

```dart
// pages/contact_update_page.dart
import 'package:flutter/material.dart';
import 'package:sqflite/sqflite.dart';

class ContactUpdatePage extends StatefulWidget {
  final Database database;
  const ContactUpdatePage({super.key, required this.database});

  @override
  State<ContactUpdatePage> createState() => _ContactUpdatePageState();
}

class _ContactUpdatePageState extends State<ContactUpdatePage> {
  final _idController = TextEditingController();
  final _nameController = TextEditingController();
  final _phoneController = TextEditingController();

  Future<void> _updateContact() async {
    final id = int.tryParse(_idController.text);
    final name = _nameController.text;
    final phone = _phoneController.text;
    if (id == null || name.isEmpty || phone.isEmpty) return;
    final count = await widget.database.rawUpdate(
      'UPDATE contacts SET name = ?, phone = ? WHERE id = ?',
      [name, phone, id],
    );
    if (count == 0) {
      ScaffoldMessenger.of(
        context,
      ).showSnackBar(SnackBar(content: Text('ไม่พบผู้ติดต่อที่มี ID \$id')));
    } else {
      ScaffoldMessenger.of(context).showSnackBar(
        const SnackBar(content: Text('อัปเดตข้อมูลเรียบร้อยแล้ว')),
      );
      _idController.clear();
      _nameController.clear();
      _phoneController.clear();
    }
  }
```

Update