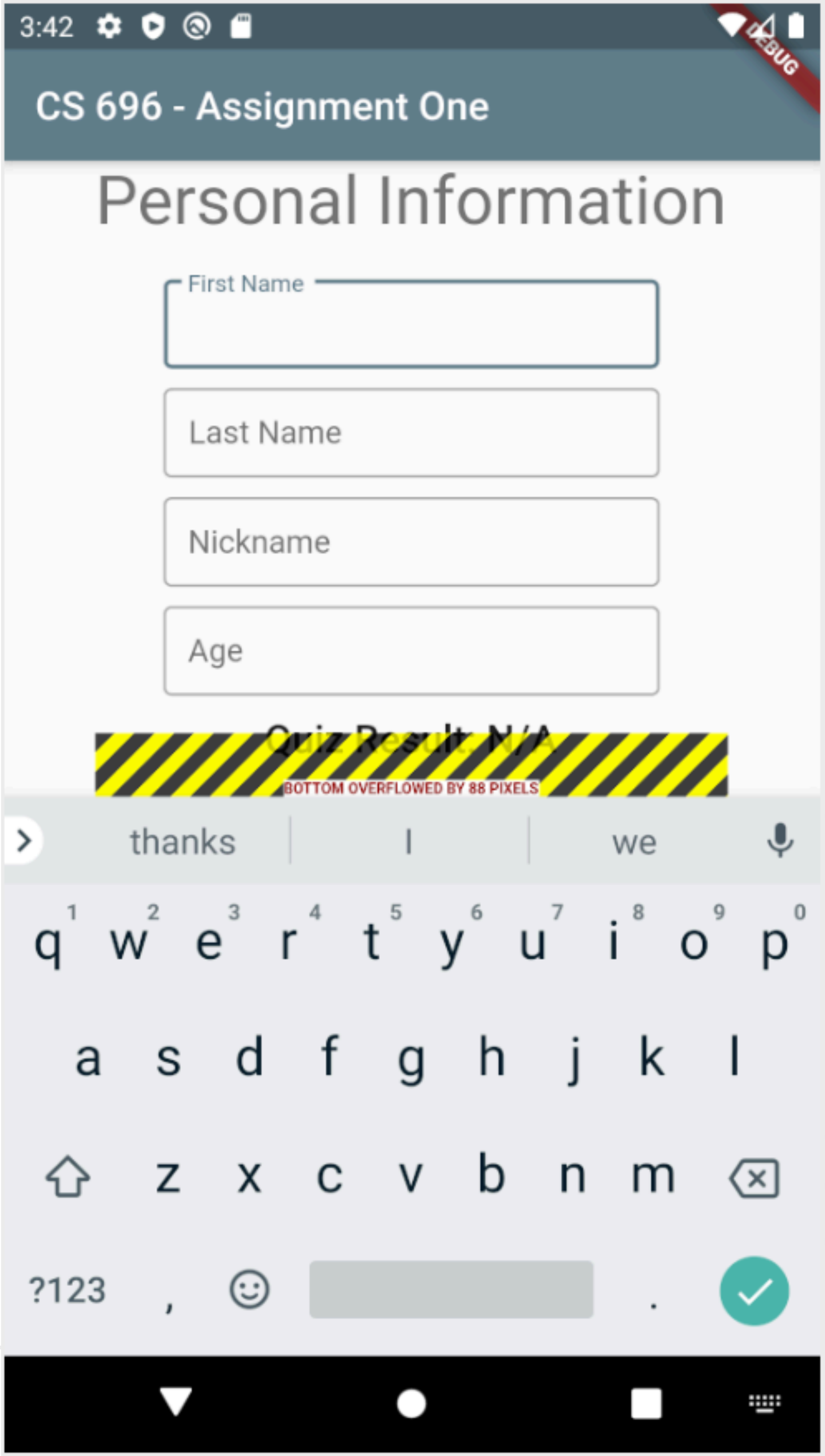
CS 696 Multi-platform Mobile App Development
Fall Semester, 2020
Doc 9 Assignment 1 Comments
Sep 29, 2020

Copyright ©, All rights reserved. 2020 SDSU & Roger Whitney, 5500 Campanile Drive, San Diego, CA 92182-7700 USA. OpenContent (http://www.opencontent.org/opl.shtml) license defines the copyright on this document.



3:42

CS 696 - Assignment One

Personal Information

First Name

Last Name

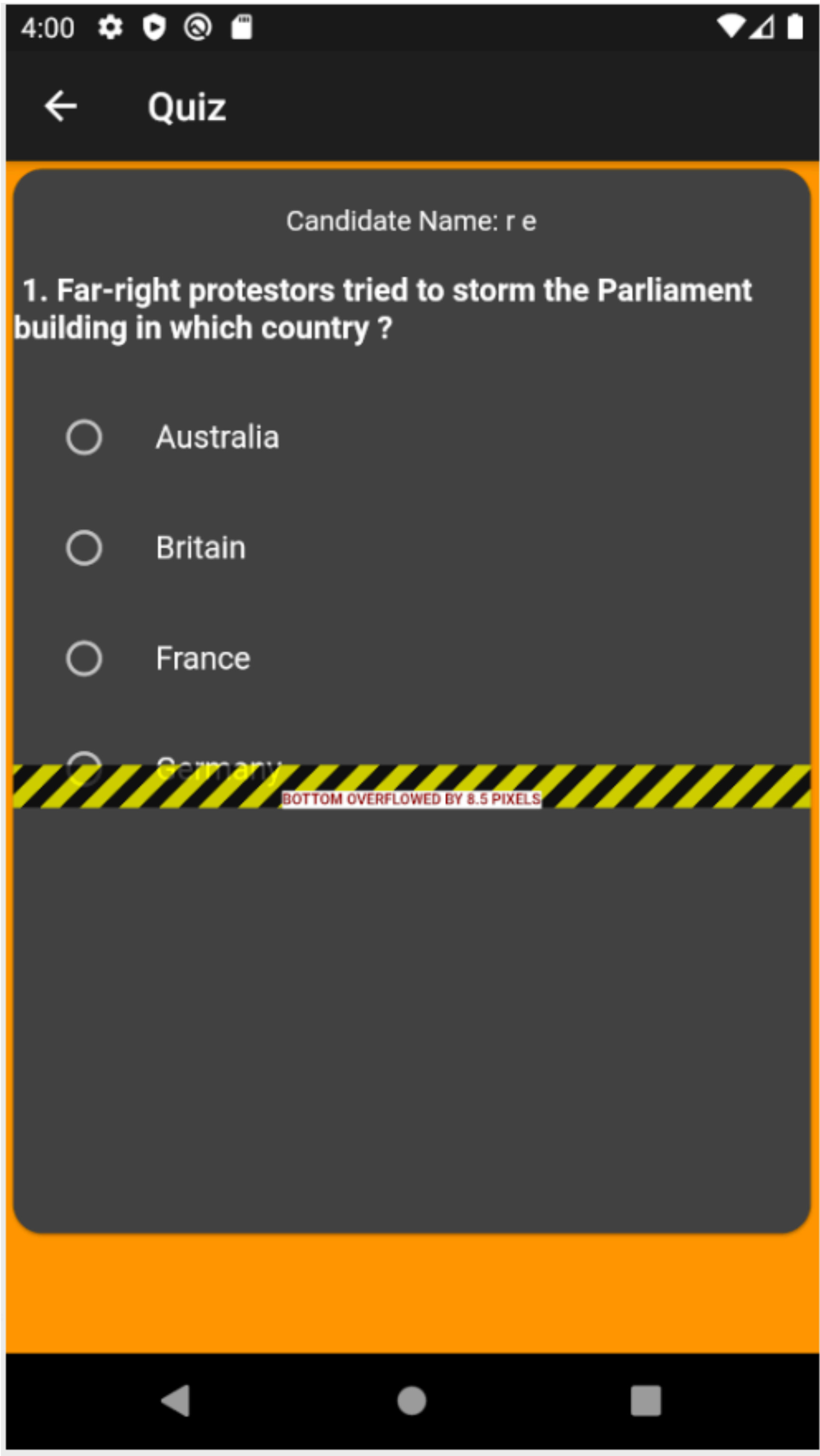
Nickname

Age

Quiz Result: N/A

BOTTOM OVERFLOWED BY 88 PIXELS

Consider applying a flex factor (e.g. using an Expanded widget) to force the children of the RenderFlex to fit within the available space instead of being sized to their natural size.



4:00

Quiz

Candidate Name: r e

1. Far-right protestors tried to storm the Parliament building in which country ?

○ Australia

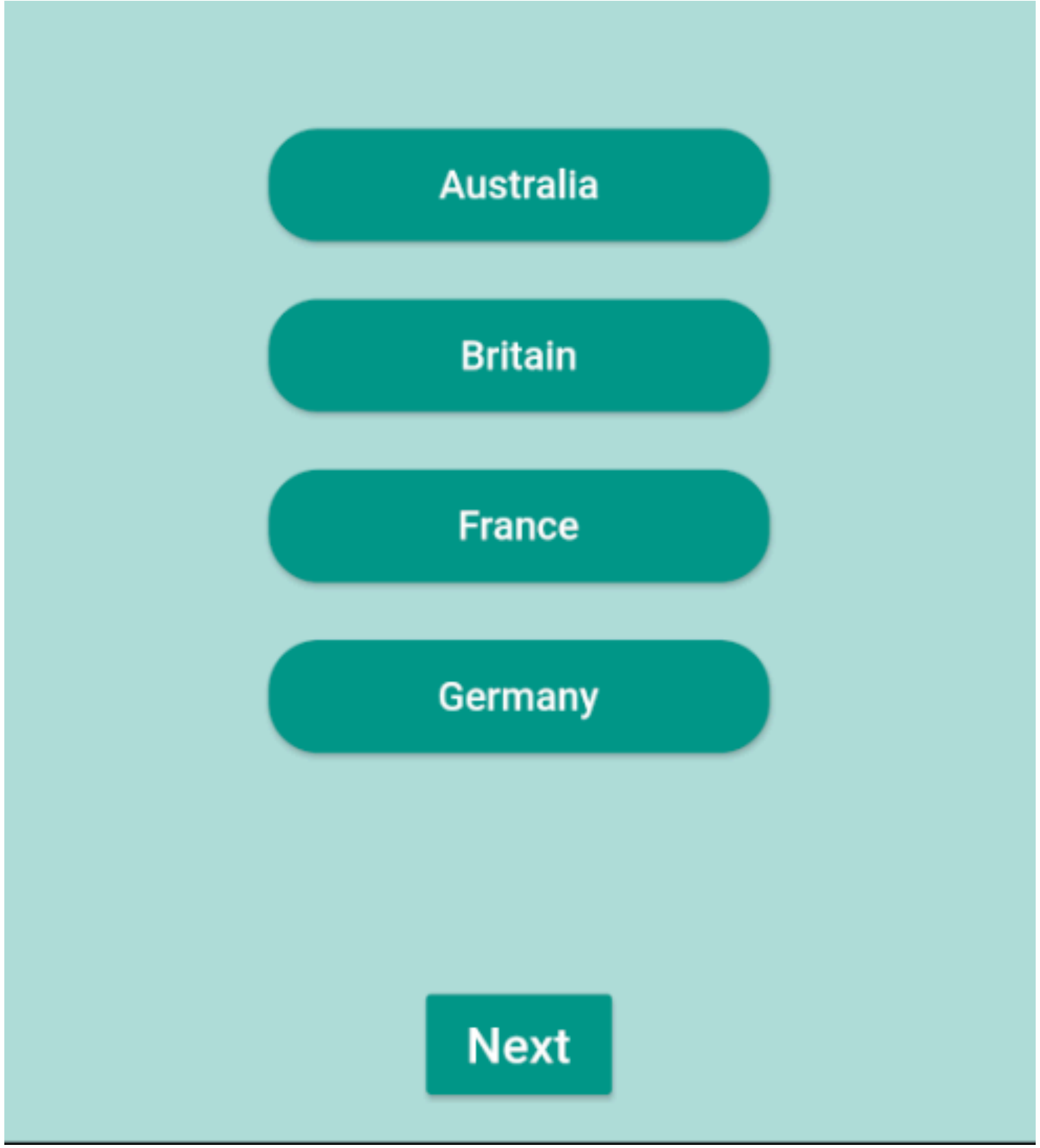
○ Britain

○ France

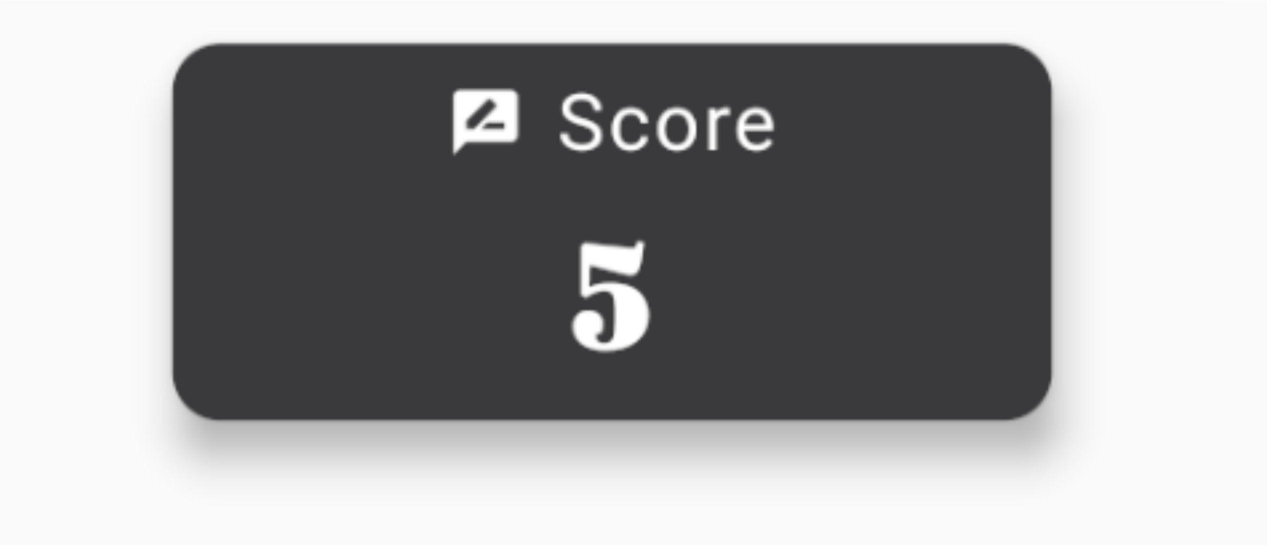
○ Germany

BOTTOM OVERFLOWED BY 8.5 PIXELS

Which one was Selected?

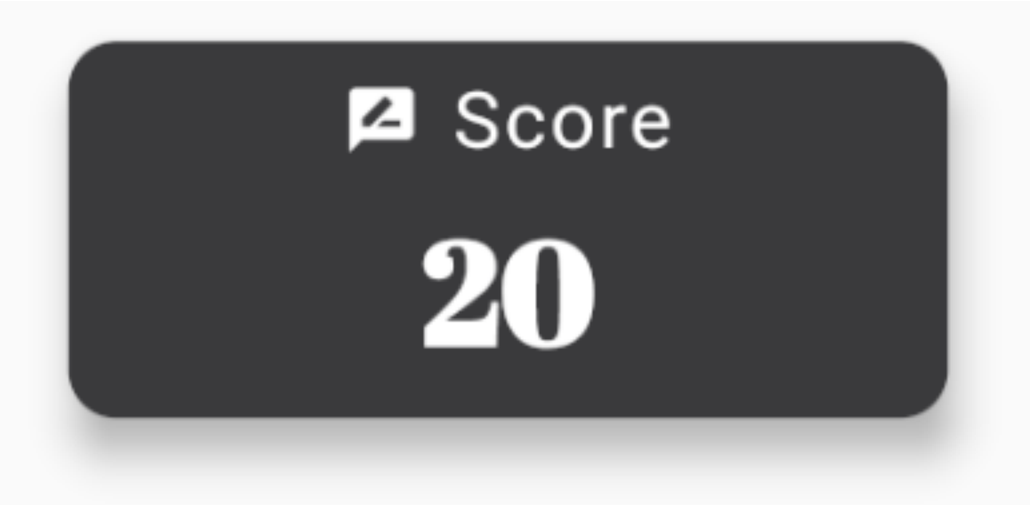


What Does 5 Mean



Score

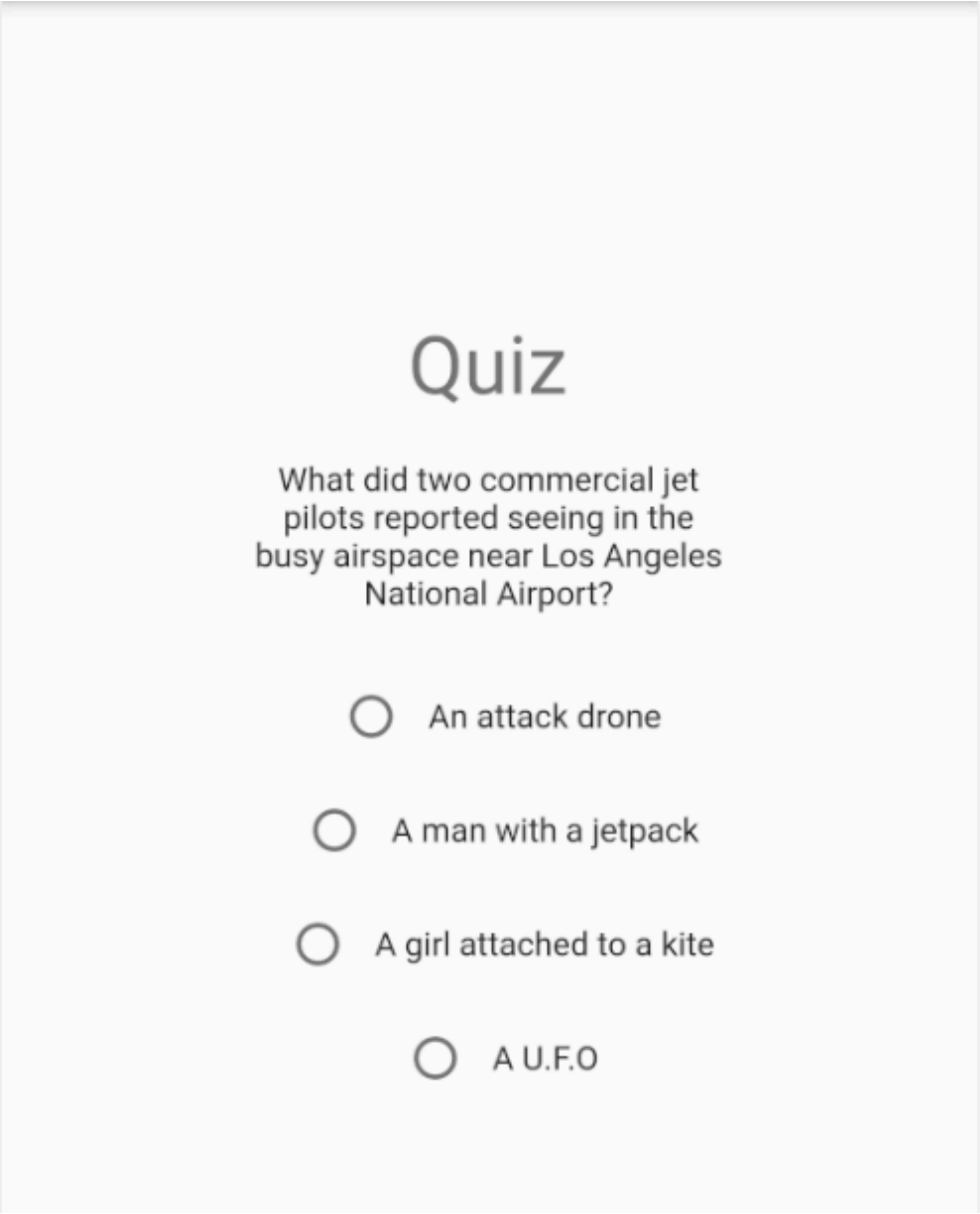
5



Score

20

Alignment



Quiz

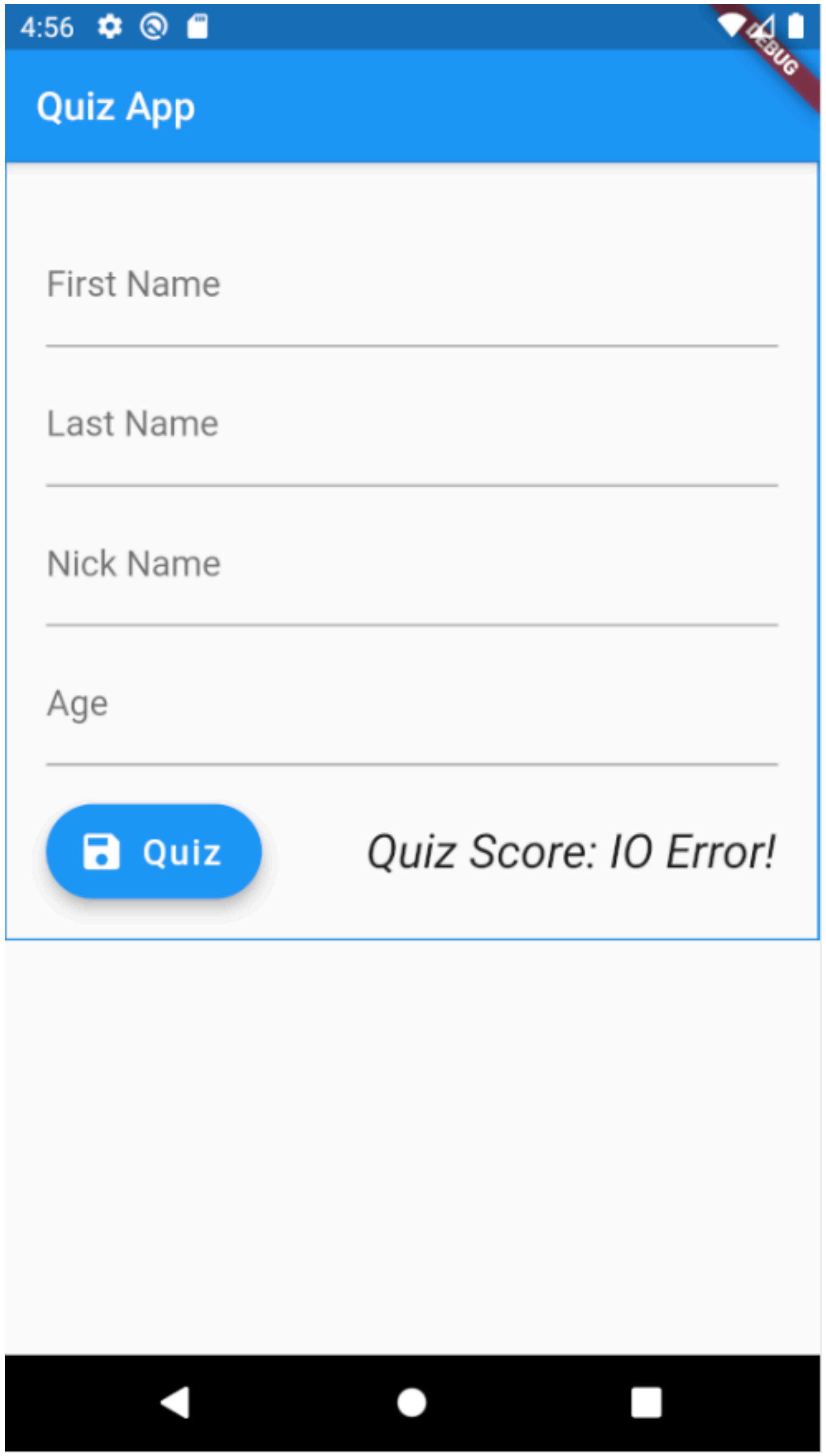
What did two commercial jet pilots reported seeing in the busy airspace near Los Angeles National Airport?

○ An attack drone

○ A man with a jetpack

○ A girl attached to a kite

○ A U.F.O



Quiz App

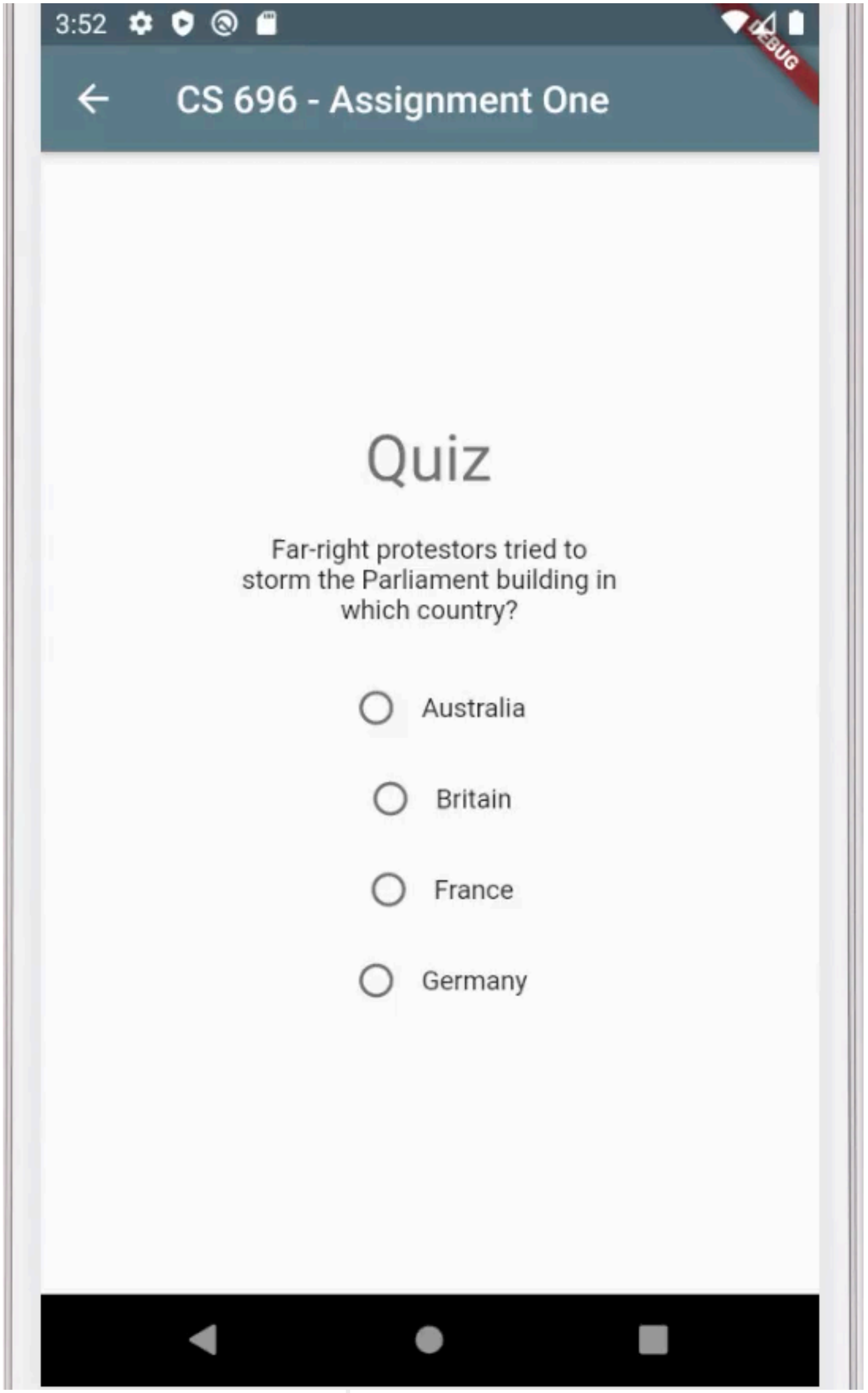
First Name

Last Name

Nick Name

Age

Quiz Score: IO Error!



CS 696 - Assignment One

Quiz

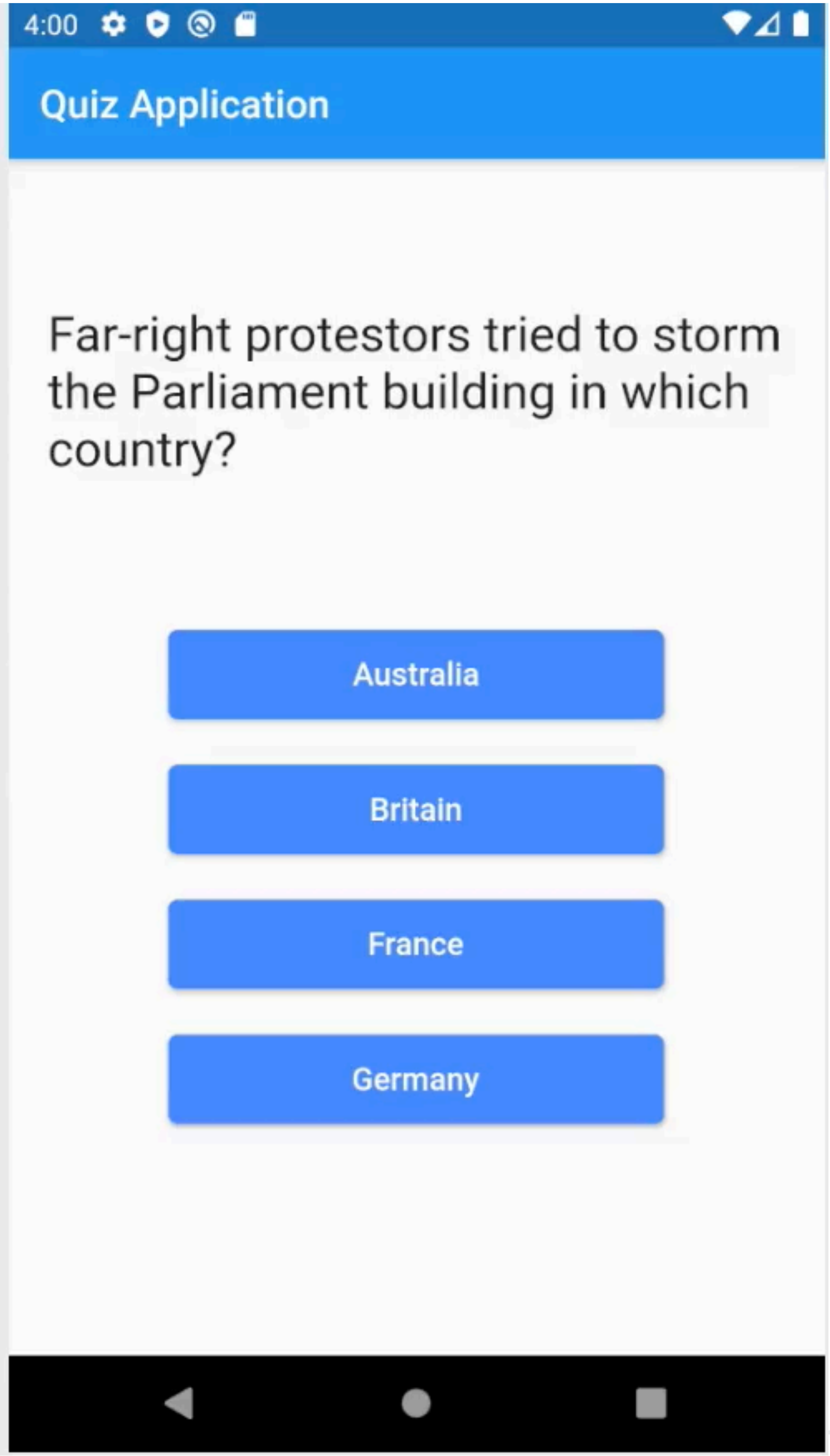
Far-right protestors tried to storm the Parliament building in which country?

○ Australia

○ Britain

○ France

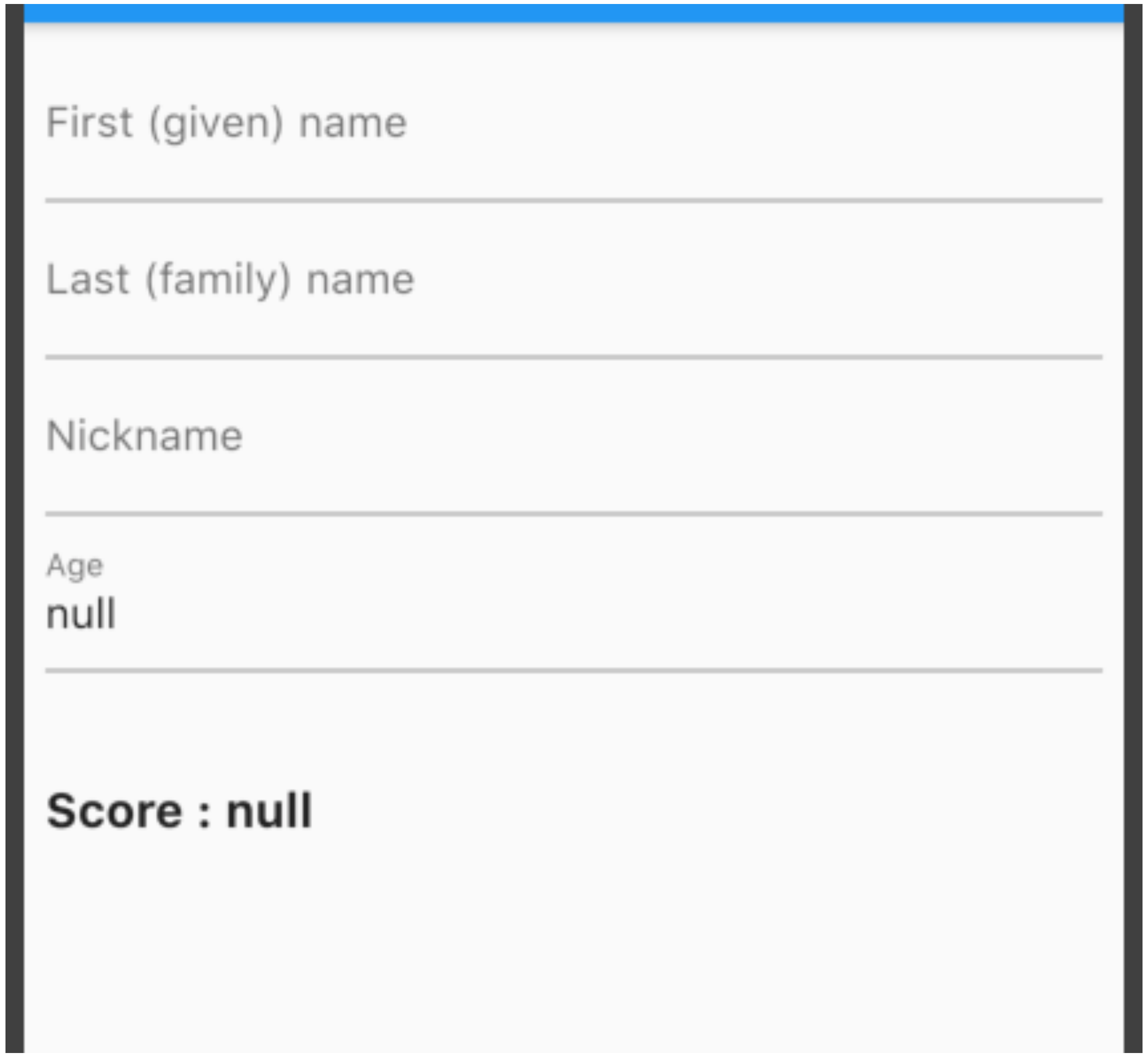
○ Germany



4:00

Quiz Application

Far-right protestors tried to storm the Parliament building in which country?



First (given) name

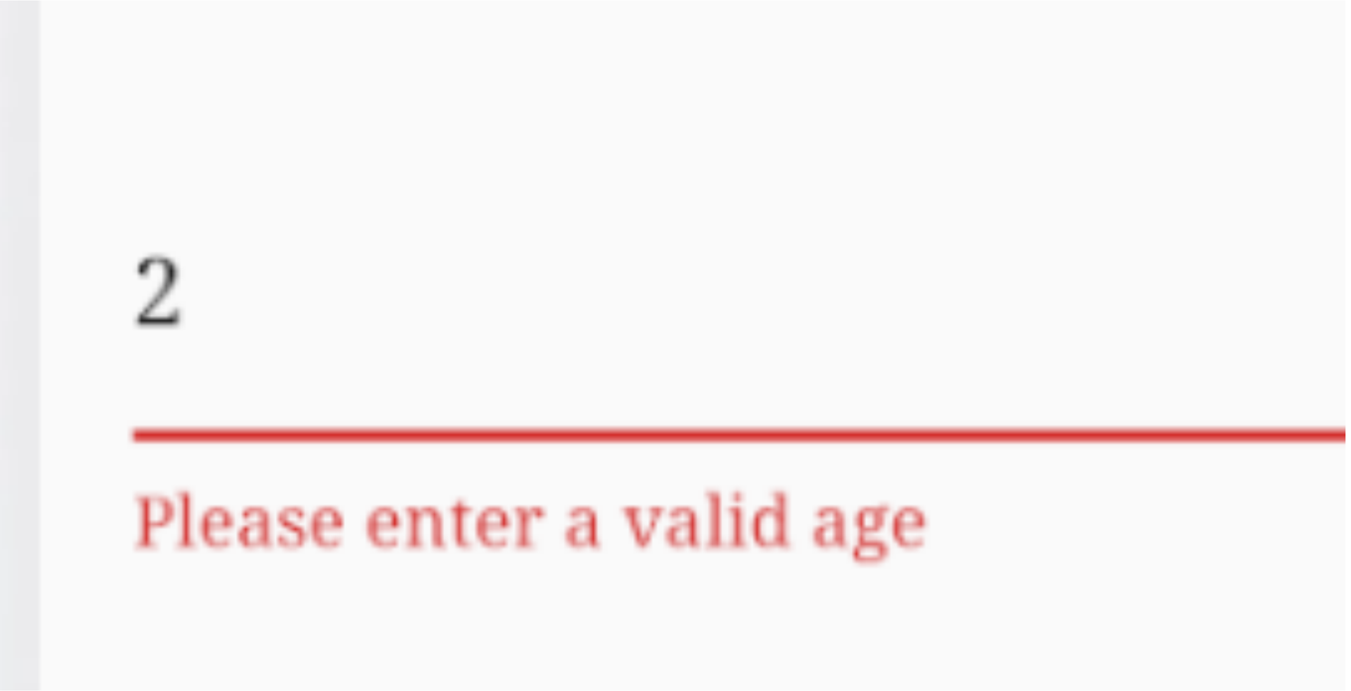
Last (family) name

Nickname

Age
null

Score : null

What is a Valid Age?



2

Please enter a valid age

Naming Conventions

```
//variables
  bool clicked = false;
  List quizQuestion = [];
  List currentAns;
  int qIndex = 0;
  List currentScore;
  int total_Score = 0;
  String character;
```

What Data?

```
 // loading the question and answer json file
 Future<String> loadData() async {
   return await rootBundle.loadString('assets/qanda.json');
 }
```

```
Future<String> loadQuestions() async {
  return await rootBundle.loadString('assets/qanda.json');
}
```

Parse What Data?

```
//parsing the json file
 Future<List> parseData() async {
  String jsonString = await loadData();
  final jsonResponse = jsonDecode(jsonString);
  setState(() {
   quizQuestion = jsonResponse['questions'];
  });
  return jsonResponse['questions'];
 }
```

jsonResponse['questions']

Implies multiple questions

quizQuestion

Implies one question

Why setState & return value?

```
//parsing the json file
 Future<List> parseQuizQuestions() async {
  String jsonString = await loadData();
  final jsonResponse = jsonDecode(jsonString);
  setState(() {
   quizQuestion = jsonResponse['questions'];
  });
  return jsonResponse['questions'];
 }
```

```
Future<List> loadQuizQuestions() async {
 String jsonString = await loadData();
 final jsonResponse = jsonDecode(jsonString)
 setState(() {
  quizQuestion = jsonResponse['questions'];
 });
 return jsonResponse['questions'];
}
```

Why _ and not _?

```
var _controller_firstName = new TextEditingController();
var _controller_lastName = new TextEditingController();
var _controller_nickName = new TextEditingController();
var _controller_age = new TextEditingController();
var quizScore;
```

Names

```
//variables declaration
  bool readyFor = false;
  int  scoreVal = 0;
  bool scoreinFile = false;
```

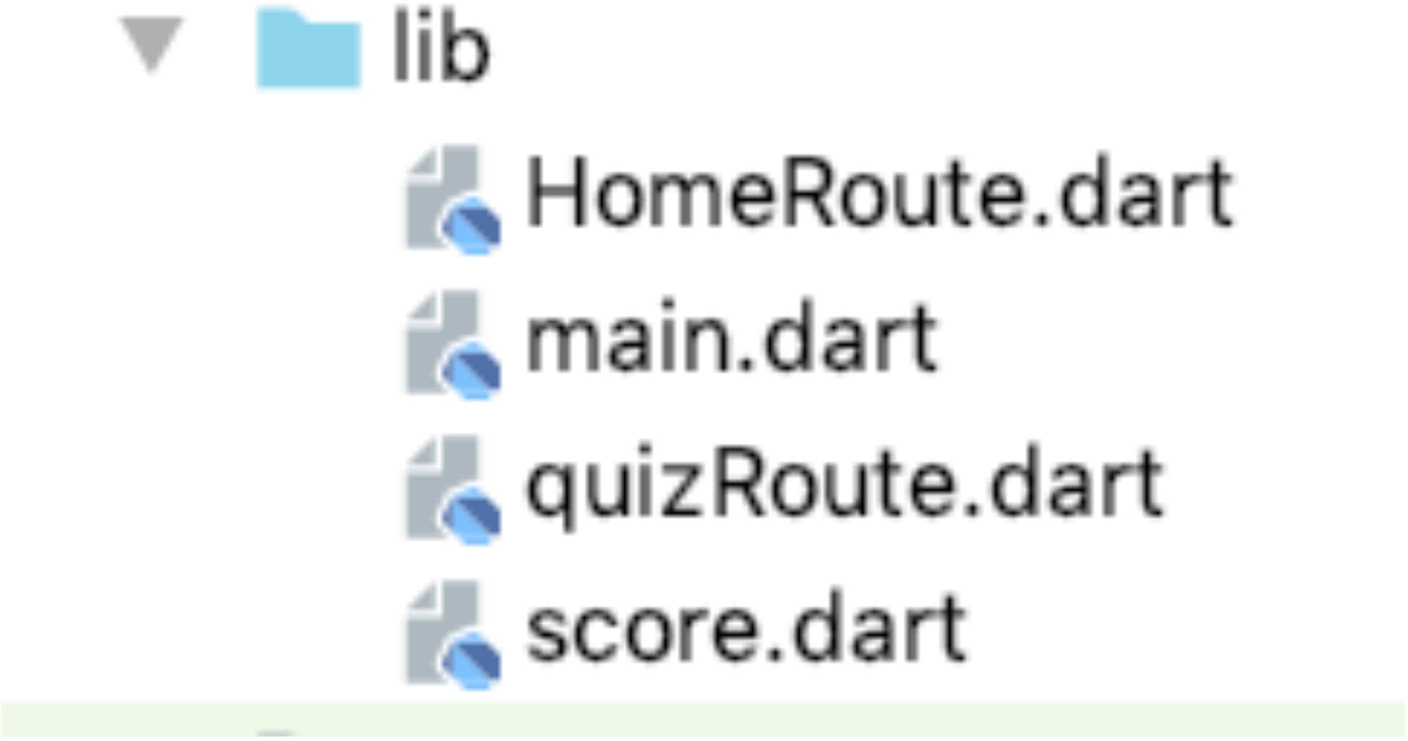
```
bool readyFor = false; // ready for what
int  score = 0;
bool scoreInFile = false;
```

tempvar, print, error handling

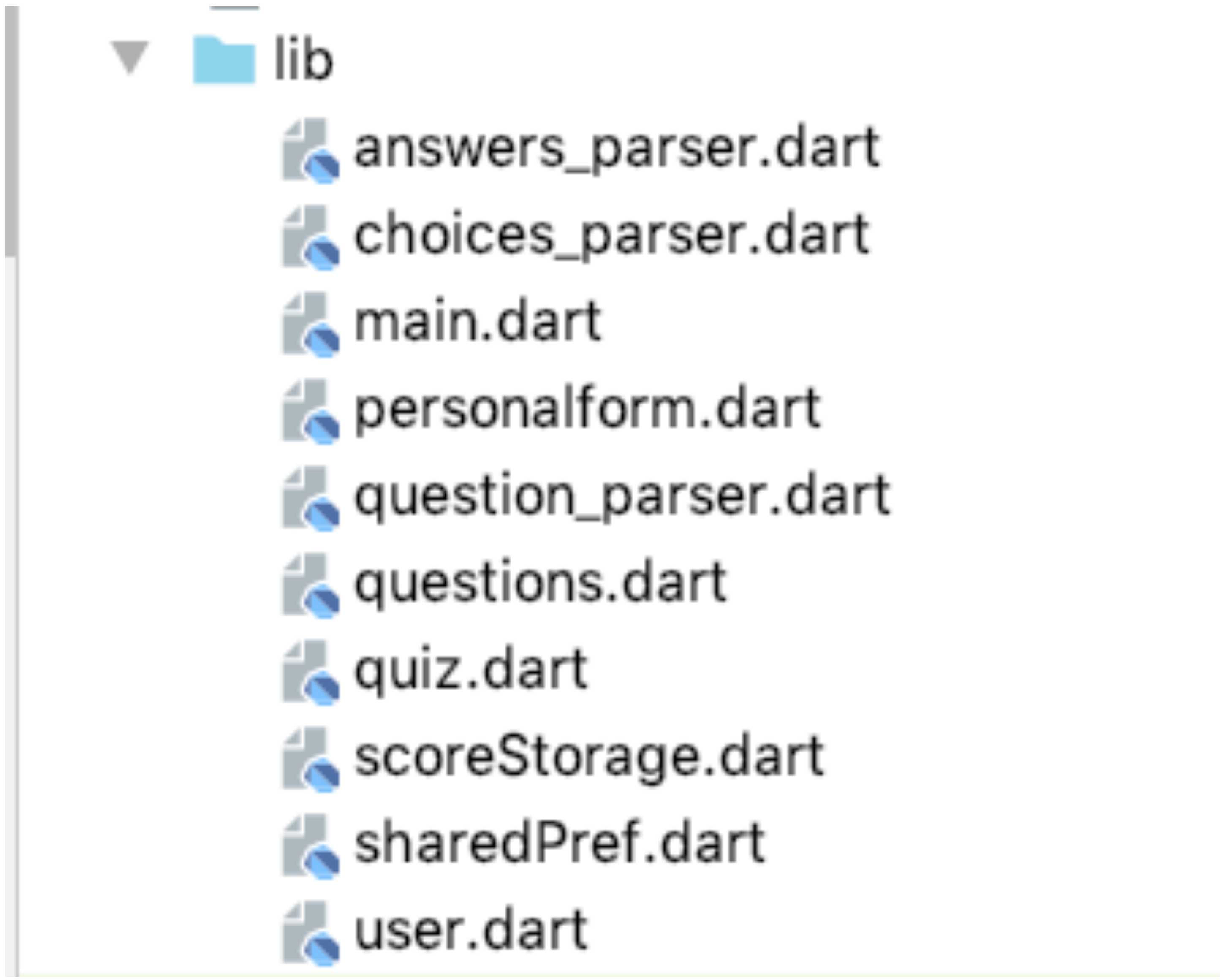
```
try {
  scoreFromFile = await _localFile.readAsString();
  if (scoreFromFile != null) {
    tempvar=true;
    print(scoreFromFile);
    print("File found. Reading the score from the file");
  }
} catch (e) {
  print("No file found, User haven't attempted any quiz yet.");
}
```

Which File data?

```
 Future<bool> loadFileData() async{
   Directory directory =  await getApplicationDocumentsDirectory() ;
   File file = File('${directory.path}/score.txt');
   isScoreAvailable =  await file.exists() ;
   if (isScoreAvailable) score =  await file.readAsString() ;
   if(score!="")
     return true;

   return false;
 }

Future<bool> loadScore() async{
```



```
▼ lib
    HomeRoute.dart
    main.dart
    quizRoute.dart
    score.dart
```



```
lib
    answers_parser.dart
    choices_parser.dart
    main.dart
    personalform.dart
    question_parser.dart
    questions.dart
    quiz.dart
    scoreStorage.dart
    sharedPref.dart
    user.dart
```

beginScore?

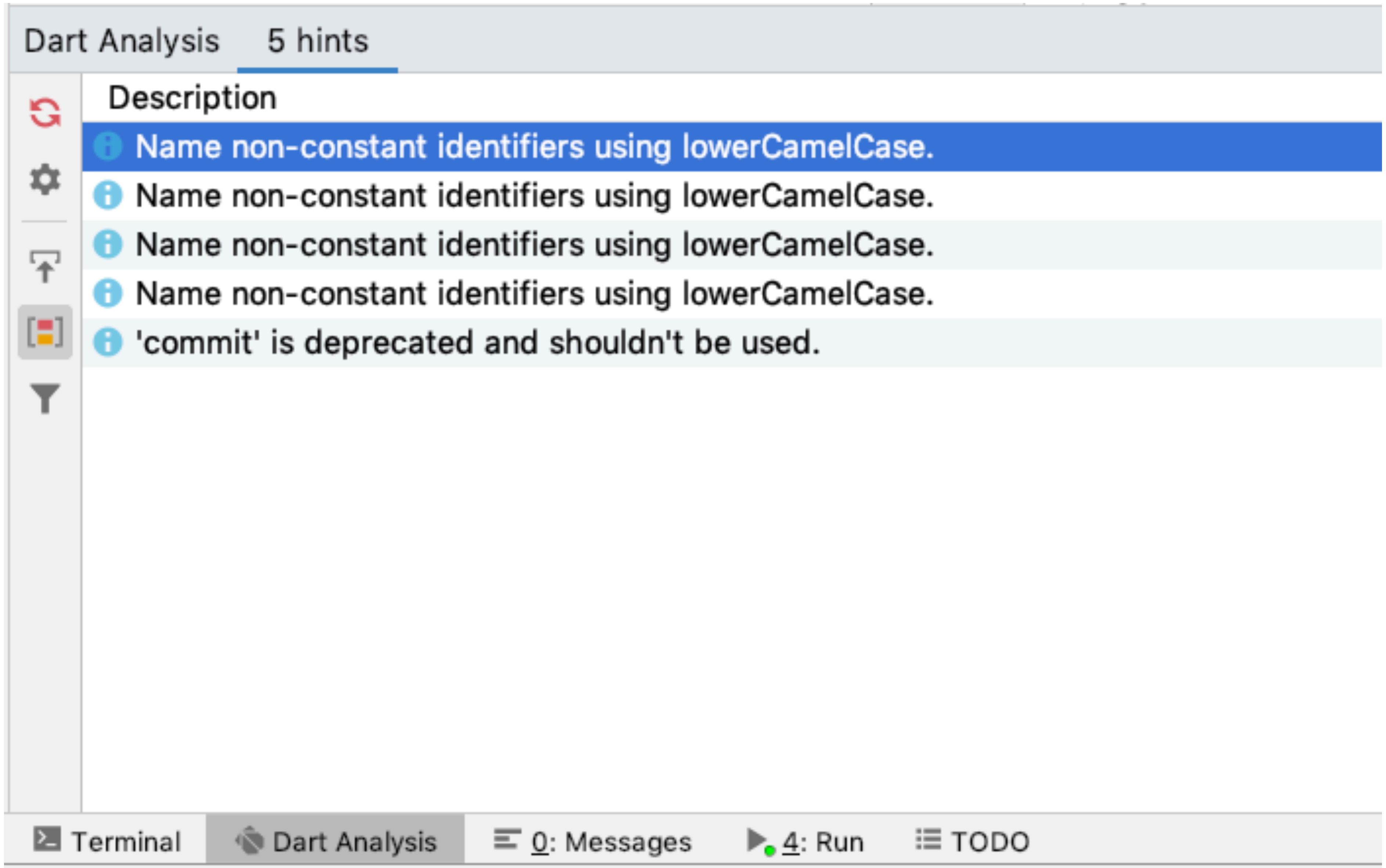
```
beginScore(BuildContext context) async {
  int returnScore = await Navigator.push(
      context,
      MaterialPageRoute(
        builder: (context) => QuizPage(),
      ));
  setState(() {
    if (returnScore != null) {
      scoreVal = returnScore;
      score.write(scoreVal);
      this.scoreinFile = true;
    } else {
      scoreVal = 0;
      score.write(scoreVal);
      this.scoreinFile = true;
    }
  });
}
```

takeQuiz

startQuiz

Dart Analysis

Helps you find issues



Name non-constant identifiers using lowerCamelCase.

Name non-constant identifiers using lowerCamelCase.

Name non-constant identifiers using lowerCamelCase.

Name non-constant identifiers using lowerCamelCase.

'commit' is deprecated and shouldn't be used.

Dart Naming Conventions

UpperCamelCase

Classes, enum types, typedefs, and type parameters

```
class SliderMenu { ... }

class HttpRequest { ... }

typedef Predicate<T> = bool Function(T value);
```

Dart Naming Conventions

lowercase_with_underscores

libraries, packages, directories, and source files

```
library peg_parser.source_scanner;

import 'file_system.dart';
import 'slider_menu.dart';
```

Dart Naming Conventions

lowerCamelCase

Class members, top-level definitions, variables, parameters, and named parameters

```
var item;

HttpRequest httpRequest;

void align(bool clearItems) {
  // ...
}
```

Dart Naming Conventions

PREFER using lowerCamelCase for constant names

Class members, top-level definitions, variables, parameters, and named parameters

```
const pi = 3.14;
const defaultTimeout = 1000;
final urlScheme = RegExp('^([a-z]+):');

class Dice {
  static final numberGenerator = Random();
}
```

SCREAMING_CAPS

When adding code to a file or library that already uses SCREAMING_CAPS.
When generating Dart code that’s parallel to Java code

Effective Dart: Style

https://dart.dev/guides/language/effective-dart/style

Describe What "flag" is Used For

✗

```
if (flag) {
    ...
}
```

✓

```
if (foundDuplicate) {
    ...
}
```

✗

```
flag
flagStatus
computeFlag
```

Do not help understand code

Variables 1 through N

✗

```
String s1;
String s2;
```

✓

```
String fileContents;
String pattern;
```

Who can remember the difference between s1 and s2?

Avoid Names With No Meaning

✗ MyLinkedList

Who are you?
What makes your LinkedList different?

✗ temp

All variables are temporary

```
swap = a;
a = b;
b = swap
```

```
(a, b) = (b, a)
```

Guidelines - Method/Function/Procedure Names

Describe what method does

Use verb to describe an action

    add(int index, E element)
    clear()

If returns a value name what it returns

    iterator()
    subList(int fromIndex, int toIndex)

If returns boolean value make it true/false statement

    isEmpty()
    contains(Object o)

Guidelines - Variables, Fields, Parameters

Use names that indicate role variable is playing

If declare variable types don't use type as name

Use plurals to indicate collections

Make boolean variable names true/false statement

isVisible, hasMultipleParts,

✗

```
public void execute(Vector vector) {
    Stack s;
}
```

✓

```
public void execute(Vector commands) {
    Stack commandsExecuted;
}
```

What Data?

```
class GetData extends StatelessWidget{
```

```
sampleFunction() async {
 int temp=currentQuestion;

 setState(() {
  if (currentQuestion < 4){
   currentQuestion = currentQuestion + 1;
  marks = current + marks;
  current = 0;
  isEnabled = false;
  btncolor.forEach((k, v) {
   btncolor[k] = Colors.indigo;
  });
 }
 });
 if(temp == 4){
  marks = current + marks;
  // used to store data in file
  storage.writeData(marks.toString());
  Navigator.pushReplacement(context , MaterialPageRoute
   (builder: (context) => MyApp()
  ),);
```

Why Repeated

```
 //functions for getting the details
//first name
 getFirstname() async {
   SharedPreferences prefs = await SharedPreferences.getInstance();
   String firstName = prefs.getString('firstName') ?? '';
   setState(() {
     fNameController.text = firstName;
   });
 }

//family (last) name
 getLastname() async {
   SharedPreferences prefs = await SharedPreferences.getInstance();
   String lastName = prefs.getString('lastName') ?? '';
   setState(() {
     lNameController.text = lastName;
   });
 }
```

initialCondition - Does what

```
void initState() {
  super.initState();
  initialCondition();
  getFirstname();
  getLastname();
  getNickname();
  getAge();
  score.availability().then((bool val) {
    setState(() {
      scoreinFile = val;
    });
  });
  score.read().then((int value) {
    setState(() {
      scoreVal = value;
    });
  });
}
```

```
initialCondition() async {
  SharedPreferences prefs = await SharedPreferences.getInstance();
  bool present = prefs.containsKey('firstName');
  this.readyFor = present;
}
```

async functions are always executed later
if readyFor is used to determine when to show something - use setState

```
return RaisedButton(
                            onPressed: () {
                              if (this.readyFor) {
                                beginScore(context);
```

async issue

```
class FormScreenState extends State<FormScreen> {
  String _firstName;
  String _lastName;
  String _nickName;
  String _age;
  String _score = "";
  bool _scoreVisibility = false;
  bool _dataSavedFlag = false;

  // Shared Preferences get data if available
  Future<String> _getDataFromSharedPref() async {
    // getting the quiz score from last attempt if available in the text file
    try {
      final directory = await getApplicationDocumentsDirectory();
      final file = File('${directory.path}/score.txt');
      String text = await file.readAsString();
      _scoreVisibility = true; // setting the score visibility
      _score = text;
    } catch (e) {
      _scoreVisibility = false;
    }
```

async issue

```
_setDat() async {
  final prefs = await SharedPreferences.getInstance();

  prefs.setString('firstName', _firstName);
  prefs.setString('lastName', _lastName);
  prefs.setString('nickName', _nickName);
  prefs.setString('age', _age);
  _dataSavedFlag = true;
}
```

What is with the return?

```
readScore() async {
  try {
    final file = await _localFile;
    String scoreStr = await file.readAsString();
    setState(() {
      score = int.parse(scoreStr);
    });
    return int.parse(scoreStr);
  } catch (e) {
    return 0;
  }
}
```

```
@override
void initState() {
  _loadModel();
  readScore();
  // _loadAssets();
  return super.initState();
}
```

What Model?

```
_loadModel() async {
  SharedPreferences prefs = await SharedPreferences.getInstance();
  setState(() {
    _firstNameController.text = (prefs.getString('firstName') ?? '');
    _lastNameController.text = (prefs.getString('lastName') ?? '');
    _nickNameController.text = (prefs.getString('nickName') ?? '');
    _ageController.text = (prefs.getInt('age').toString() ?? '0');
  });
}
```

```
class QuizModel {
  int score;
  List<QnA> qnA;

  QuizModel({this.score, this.qnA});

  QuizModel.fromJson(Map<String, dynamic> json) {
    score = json['score'];
    if (json['QnA'] != null) {
      qnA = new List<QnA>();
      json['QnA'].forEach((v) {
        qnA.add(new QnA.fromJson(v));
      });
    }
  }

  Map<String, dynamic> toJson() {
    final Map<String, dynamic> data = new Map<String, dynamic>();
    data['score'] = this.score;
    if (this.qnA != null) {
      data['QnA'] = this.qnA.map((v) => v.toJson()).toList();
    }
    return data;
  }
}
```

```dart
import 'dart:io';

import 'package:path_provider/path_provider.dart';

class ScoreStorage {
  Future<String> localPath() async {
    final directory = await getApplicationDocumentsDirectory();
    return directory.path;
  }

  Future<File> localFile() async {
    final path = await localPath();
    return File('$path/score.txt');
  }

  Future<int> readScore() async {
    try {
      final file = await localFile();
      String contents = await file.readAsString();
      return int.parse(contents);
    } catch (e) {
      return null;
    }
  }

  Future<File> writeScore(int score) async {
    final file = await localFile();
    return file.writeAsString('$score');
  }
}
```

```
class Profile {
  String firstName;
  String lastName;
  String nickname;
  int age;
  String score;

  Profile({this.firstName, this.lastName, this.nickname, this.age, this.score});
}
```

Why not add to class

```
  bool save() {
      //write to shared preferences
      //write to file
      return if success
  }

  static Profile fromJSON(String JSONObject) {


  }

  static Future<Profile> load() {


  }
```

Model View Controller - MVC

Keep the model and how it is displayed separate

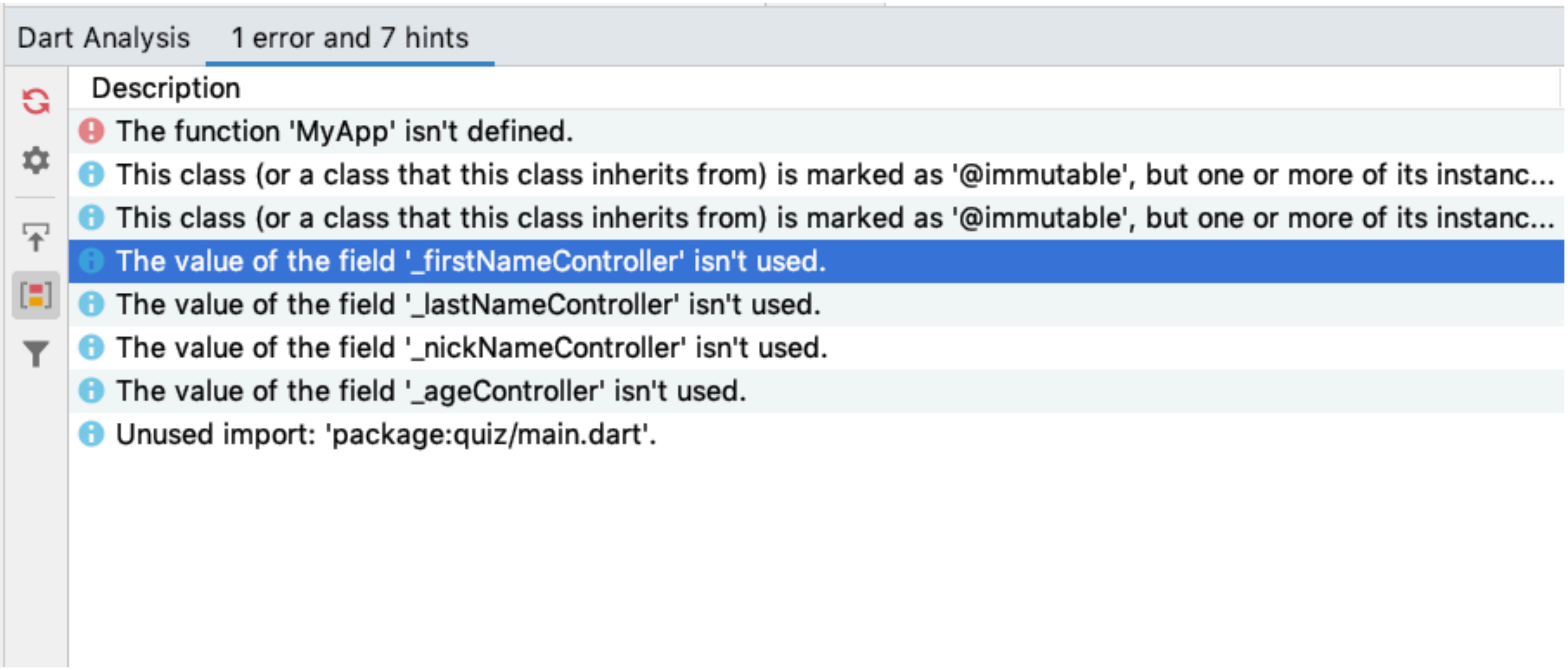
Assignment 1 Model

- Quiz
- Personal Data
- Reading/Writing data

Don't create Future in FutureBuilder

```
body: ListView(
    padding: EdgeInsets.symmetric(horizontal: 8.0, vertical: 24.0),
    children: <Widget>[
      FutureBuilder(
          future: DefaultAssetBundle.of(context).loadString(
              "assets/quiz.json", cache: false),
          builder: (context , snapshot){
```

Each time the widget is rebuild the future is recreated



The function 'MyApp' isn't defined.
This class (or a class that this class inherits from) is marked as '@immutable', but one or more of its instanc...
This class (or a class that this class inherits from) is marked as '@immutable', but one or more of its instanc...
The value of the field '_firstNameController' isn't used.
The value of the field '_lastNameController' isn't used.
The value of the field '_nickNameController' isn't used.
The value of the field '_ageController' isn't used.
Unused import: 'package:quiz/main.dart'.

Formating

```
          TextField(
            controller: _controller_data_two,
            obscureText: false,
            decoration: InputDecoration(
              border: OutlineInputBorder(),
              labelText: 'Age',
            ),
          ),
Row(mainAxisAlignment: MainAxisAlignment.center,
children: <Widget>[
  Padding(padding: const EdgeInsets.all(6.0),
    child: RaisedButton(
        onPressed: ()=> saveData(_controller_key.text,_controller_data.text, _controller_key_two.text,
_controller_data_two.text),
        child: Text('Save Data')
    ),

  ),

  Padding(padding: const EdgeInsets.all(6.0),
    child: RaisedButton(
        onPressed: ()=> readData(_controller_key.text, _controller_key_two.text )
```

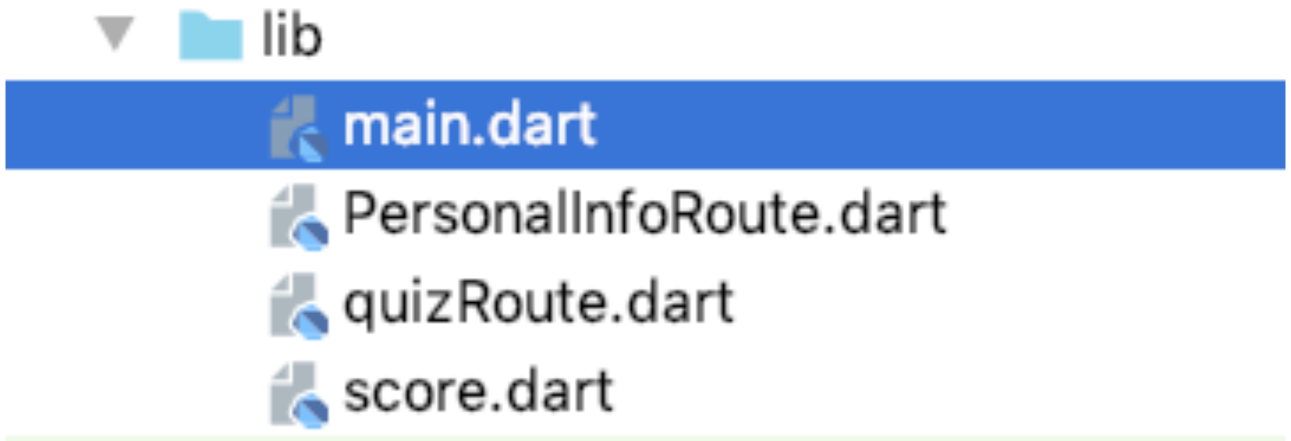
What is State?

```
class PersonalInformationFormState extends State<PersonalInformationForm> {

  bool isEnabled=false;
  final _formKey = GlobalKey<FormState>();
  TextEditingController firstNameController;
  TextEditingController lastNameController;
  TextEditingController nickNameController;
  TextEditingController ageController;

  String scoreText='';
  String firstName;
  String familyName;
  String nickName;
  String age;
```

Did you Dispose TextEditingController?

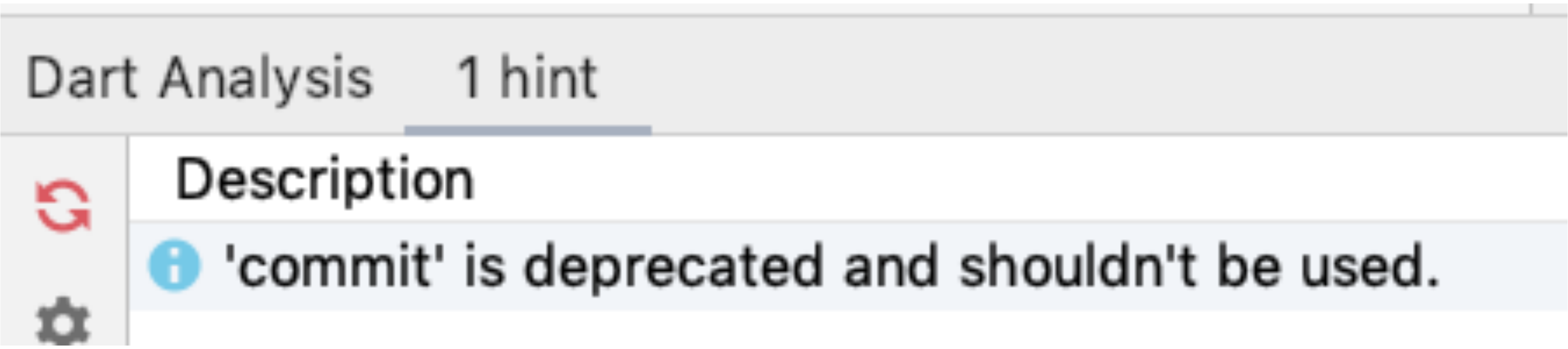
Where would you find InformationForm?

```
class MyApp extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  final appTitle = 'Quiz App';

  return MaterialApp(
   title: appTitle,
   theme: ThemeData(
    primarySwatch: Colors.cyan,
    visualDensity: VisualDensity.adaptivePlatformDensity,
   ),
   home: Scaffold(
    appBar: AppBar(
     title: Text(appTitle),
    ),
    body: InformationForm(),
   ),
  );
 }
}
```



```
▼ lib
    main.dart
    PersonalInfoRoute.dart
    quizRoute.dart
    score.dart
```

```
Future<bool> saveData(
    String firstName, String lastName, String nickName, String age) async {
  SharedPreferences prefs = await SharedPreferences.getInstance();
  prefs.setString("firstName", firstName);
  prefs.setString("lastName", lastName);
  prefs.setString("nickName", nickName);
  prefs.setString("age", age);
  return prefs.commit();
}
```



'commit' is deprecated and shouldn't be used.

The Entire File

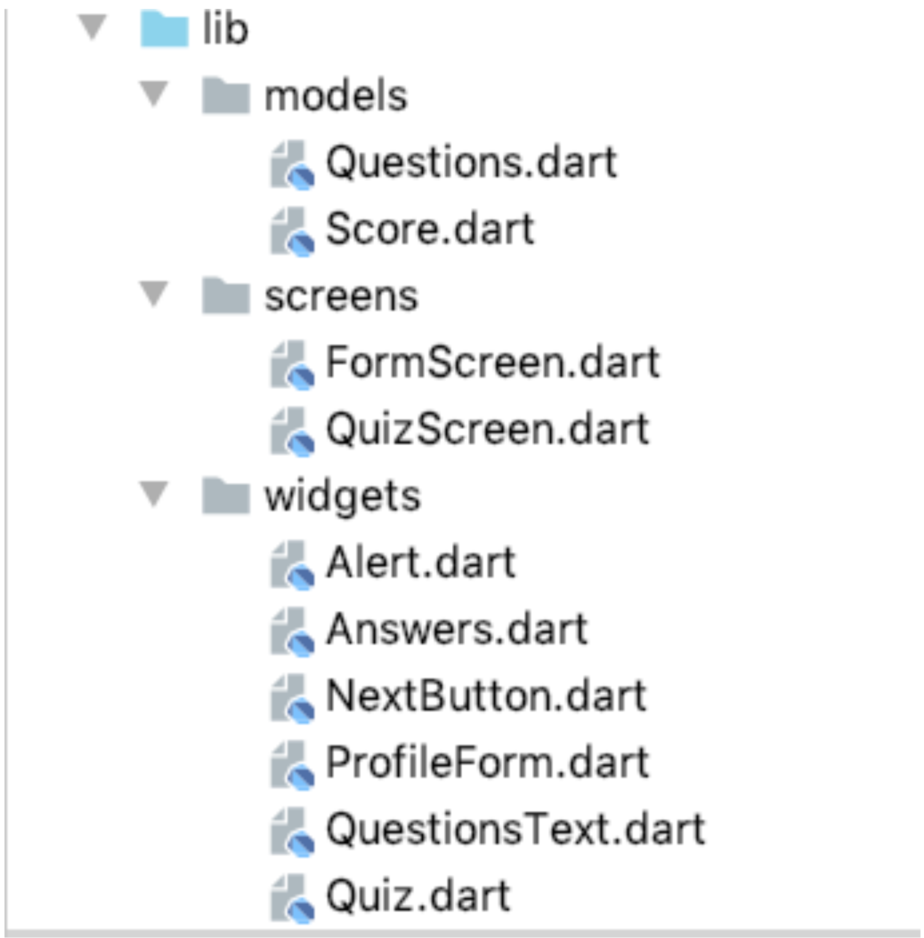
```
import 'package:json_annotation/json_annotation.dart';
```

async again

```
void initState() {
  super.initState();
  loadQuestion();
  loadChoices();
  loadAnswers();
  widget.storage.readScore().then((int value) {
    setState(() {
      _score = value;
    });
  });
}
```

```
Future<String> _loadQuestions() async {
  return await rootBundle.loadString('assets/questions.json');
}

Future loadQuestion() async {
  String jsonQuestion = await _loadQuestions();
  var questions = jsonDecode(jsonQuestion)['questions'];

  return questions;
}
```

Providing Some Structure



```
lib
  models
    Questions.dart
    Score.dart
  screens
    FormScreen.dart
    QuizScreen.dart
  widgets
    Alert.dart
    Answers.dart
    NextButton.dart
    ProfileForm.dart
    QuestionsText.dart
    Quiz.dart
```

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Quiz App',
      theme: ThemeData(
        primarySwatch: Colors.purple,
        accentColor: Colors.deepOrange,
        brightness: Brightness.light,
        fontFamily: 'Lato',
      ),
      initialRoute: FormScreen.routeName,
      //Navigation route table
      routes: {
        FormScreen.routeName: (ctx) => FormScreen(),
        QuizScreen.routeName: (ctx) => QuizScreen(),
      },
    );
  }
}
```

Why setState & return?

```
Future<Map<String, String>> _fetchInitialData() async {
  final SharedPreferences prefs = await SharedPreferences.getInstance();
  String val = await Score.readScore();
  setState(() {
    _fileScore = val;
  });
  // return initial values data from shared preferences  and  return a map of initial values
  return {
    'name': prefs.getString('name') ?? '',
    'lastName': prefs.getString('lastName') ?? '',
    'age': prefs.getString('age') ?? '',
    'nickName': prefs.getString('nickName') ?? '',
  };
}
```

Why score in setState and rest in return?

Where is the State?

```
class _MyHomePageState extends State<MyHomePage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(widget.title),
      ),
      body: HomePage(),
    );
  }
}
```

```
The following NoSuchMethodError was thrown building FutureBuilder<String>(dirty, state:
_FutureBuilderState<String>#1dd2b):
The method '[]' was called on null.
Receiver: null
Tried calling: []("Questions")

The relevant error-causing widget was:
  FutureBuilder<String> file:///Users/whitney/Courses/696/Fall20/assignments/submissions-2/Quiz/lib/
app_screens/quiz_screen.dart:37:15
When the exception was thrown, this was the stack:
#0      Object.noSuchMethod (dart:core-patch/object_patch.dart:51:5)
#1      _QuizScreenState.build.<anonymous closure> (package:assignment1/app_screens/
quiz_screen.dart:43:42)
#2      _FutureBuilderState.build (package:flutter/src/widgets/async.dart:740:55)
#3      StatefulElement.build (package:flutter/src/widgets/framework.dart:4663:28)
#4      ComponentElement.performRebuild (package:flutter/src/widgets/framework.dart:4546:15)
```