

Easy Flutter

Buku Panduan

1. Putra Prima Arhandi

2. Annisa Taufika Firdausi

3. Dharma Yudistira Eka Putra

# DAFTAR ISI

[DAFTAR ISI 1](#_Toc107257024)

[DAFTAR GAMBAR 2](#_Toc107257025)

[DAFTAR TABEL 3](#_Toc107257026)

[ABSTRAK 4](#_Toc107257027)

[BAB 1 PENDAHULUAN 5](#_Toc107257028)

[1.1 Deskripsi Umum Aplikasi 5](#_Toc107257029)

[1.2 Pre-scaffolded Parsons Problem 5](#_Toc107257030)

[1.3 Kebutuhan Aplikasi 5](#_Toc107257031)

[1.3.1 Kebutuhan Minimum Perangkat Lunak 5](#_Toc107257032)

[1.3.2 Kebutuhan Minimum Perangkat Keras 5](#_Toc107257033)

[1.4 Proses Pengerjaan Latihan Soal 6](#_Toc107257034)

[BAB 2 PENGGUNAAN APLIKASI 7](#_Toc107257035)

[2.1 Cara Mengakses Aplikasi 7](#_Toc107257036)

[2.2 Mengerjakan Latihan Soal Pada Aplikasi 7](#_Toc107257037)

[BAB 3 PROJECT 14](#_Toc107257038)

[3.1 Struktur Project 14](#_Toc107257039)

[3.2 Source Code 14](#_Toc107257040)

[3.2.1 Menyediakan Latihan Soal 17](#_Toc107257041)

[3.2.2 Mengacak Blok Kode 17](#_Toc107257042)

[3.2.3 Mengecek Jawaban 17](#_Toc107257043)

[3.2.4 Menyimpan Log 18](#_Toc107257044)

# DAFTAR GAMBAR

[Gambar 1.1 Proses Pengerjaan Latihan Soal 6](#_Toc107225839)

[Gambar 2.1 Halaman Login 7](#_Toc107225850)

[Gambar 2.2 Halaman Login sebagai Mahasiswa 8](#_Toc107225851)

[Gambar 2.3 Halaman Utama Mahasiswa 8](#_Toc107225852)

[Gambar 2.4 Instuksi Fitur Code Reconstruction pada Halaman Utama Mahasiswa 9](#_Toc107225853)

[Gambar 2.5 Halaman Daftar Latihan Soal 9](#_Toc107225854)

[Gambar 2.6 Informasi Komponen pada Halaman Daftar Latihan Soal 10](#_Toc107225855)

[Gambar 2.7 Halaman Pengerjaan Latihan Soal 10](#_Toc107225856)

[Gambar 2.8 Informasi Komponen pada Halaman Pengerjaan Latihan Soal 11](#_Toc107225857)

[Gambar 2.9 Dialog Konfirmasi Pengerjaan Latihan Soal 11](#_Toc107225858)

[Gambar 2.10 Informasi Perubahan Halaman Setelah Konfirmasi Pengerjaan Latihan Soal 12](#_Toc107225859)

[Gambar 2.11 Dialog Penyelesaian Latihan Soal 13](#_Toc107225860)

[Gambar 3.1 Struktur Project 14](#_Toc107256986)

# DAFTAR TABEL

[Tabel 1.1 Kebutuhan Minimum Perangkat Lunak 5](#_Toc107225861)

[Tabel 1.2 Kebutuhan Minimum Perangkat Keras 5](#_Toc107225862)

# ABSTRAK

Flutter merupakan produk Google untuk mengembangkan aplikasi multi-platform. Namun, gaya penulisan kode yang deklaratif pada layout membuat penulisan kode Flutter berbeda dengan gaya penulisan kode di platform native pada umumnya. Dengan adanya perbedaan penulisan kode penyusunan layout pada Flutter, perlu adanya metode yang sesuai agar pelajar dapat memahami konsep penyusunan Flutter Layout. Penelitian ini mengenalkan pendekatan baru dalam pembelajaran Flutter Layout dengan menggunakan pendekatan Pre-scaffolded Parsons Problem. Dengan pendekatan ini, pelajar akan disediakan susunan blok kode yang telah diacak dan pelajar dapat menyusunnya ke dalam urutan yang benar. Dalam penerapannya, pendekatan ini diimplementasikan ke dalam sebuah aplikasi pengayaan berbasis web bernama EasyFlutter. Untuk melihat dampak penerapan aplikasi ini, peneliti melakukan eksperimen dengan menyediakan pre-test dan post-test pada 25 partisipan. Pada pre-test didapatkan hasil rata - rata 77,332 dan post-test 83,464. Secara deskriptif terdapat kenaikan rata - rata sebesar 6.132, selanjutnya perbedaan rata - rata tersebut dilakukan uji statistik Paired T-test dan mendapatkan hasil signifikansi sebesar 0,00. Berdasarkan hasil tersebut dapat diambil kesimpulan bahwa terdapat perbedaan rata - rata yang signifikan pada pre-test dan post-test. Dengan demikian, penggunaan aplikasi EasyFlutter memiliki dampak positif terhadap peningkatan pemahaman partisipan pada topik Flutter Layout.

**Kata kunci : Flutter Layout, Pre-scaffolded Parsons Problem, Paired T-test.**

# PENDAHULUAN

## Deskripsi Umum Aplikasi

EasyFlutter merupakan sebuah aplikasi pengayaan yang ditujukan untuk pelajar dalam memahami konsep pemrograman Flutter khususnya pada topik Flutter Layout. Fitur code reconstruction pada aplikasi ini akan memberikan beberapa latihan soal dengan topik Flutter Layout. Fitur ini mengimplementasikan pendekatan Pre-scaffolded Parsons Problem sebagai pendekatan baru dalam pembelajaran Flutter Layout. Pendekatan Pre-scaffolded Parsons Problem dapat mengurangi beban kognitif pelajar karena pelajar tidak perlu menuliskan baris kode serta dapat membatasi permasalahan yang dihadapi pelajar.

## Pre-scaffolded Parsons Problem

Parsons Problem merupakan salah satu teknik pembelajaran pemrograman dengan pendekatan penyelesaikan kode (code-completion), di mana pelajar harus menyusun ulang blok kode yang sudah diacak sebelumnya untuk mencapai solusi dari permasalahan program yang diinginkan.

Pre-scaffolded Parsons Problem merupakan jenis Parsons Problem yang memfokuskan pelajar untuk menyusun blok kode kedalam urutan yang benar. Hal ini dapat dicapai dengan cara menyiapkan kerangka dari solusi permasalahan atau menyiapkan blok-blok kode yang sudah ditulis dengan benar namun dalam urutan yang acak. Selain itu, Pre-scaffolded Parsons Problem terbukti sebagai alat pembelajaran yang efisien jika dibandingkan dengan menulis dan memperbaiki kode program.

## Kebutuhan Aplikasi

### Kebutuhan Minimum Perangkat Lunak

Tabel 1.1 Kebutuhan Minimum Perangkat Lunak

|  |  |  |
| --- | --- | --- |
| No | Nama Perangkat Lunak | Deskripsi |
| 1 | Windows 8 - 64bit | Sistem Operasi |
| 2 | Mac Sierra | Sistem Operasi |
| 3 | Google Chrome / Microsoft Edge / Mozila Firefox | Web Browser |

### Kebutuhan Minimum Perangkat Keras

Tabel 1.2 Kebutuhan Minimum Perangkat Keras

|  |  |  |
| --- | --- | --- |
| No | Nama Perangkat Keras | Deskripsi |
| 1 | 1366 x 768 | Resolusi Layar |
| 2 | Mouse | Peralatan Antaramuka |
| 3 | Keyboard | Peralatan Antarmuka |

## Proses Pengerjaan Latihan Soal

Diagram

Description automatically generated

Gambar 1.1 Proses Pengerjaan Latihan Soal

# PENGGUNAAN APLIKASI

## Cara Mengakses Aplikasi

Untuk mengakses aplikasi EasyFlutter, pengguna dapat melakukan langkah - langkah berikut ini :

1. Buka web browser yang akan digunakan, sebagai contoh Google Chrome.
2. Kemudian masukkan url "easy-flutter.github.io/code-reconstruction/".
3. Jika halaman login telah ditampilkan, maka pengguna telah berhasil mengakses aplikasi easyflutter.

Graphical user interface, application

Description automatically generated

Gambar 2.1 Halaman Login

## Mengerjakan Latihan Soal Pada Aplikasi

1. Akses aplikasi EasyFlutter melalui web browser dengan memasukkan url "easy-flutter.github.io/code-reconstruction/".
2. Selanjutnya halaman login akan ditampilkan.
3. Masukkan username dan password kemudian pilih **Mahasiswa** pada menu "Masuk sebagai", selanjutnya tekan tombol "Masuk".

Graphical user interface, application

Description automatically generated

Gambar 2.2 Halaman Login sebagai Mahasiswa

1. Selanjutnya aplikasi akan menampilkan halaman utama ketika pengguna berhasil login sebagai mahasiswa.

Graphical user interface, application

Description automatically generated

Gambar 2.3 Halaman Utama Mahasiswa

1. Pilih fitur **Code Reconstruction** pada navigasi header di bagian atas halaman.

Graphical user interface, application

Description automatically generated

Gambar 2.4 Instuksi Fitur Code Reconstruction pada Halaman Utama Mahasiswa

1. Selanjutnya halaman daftar latihan soal akan ditampilkan, pada halaman ini terdapat 10 latihan soal dengan topik Flutter Layout.

A picture containing table

Description automatically generated

Gambar 2.5 Halaman Daftar Latihan Soal

1. Berikut adalah penjelasan dari komponen - komponen yang ditampilkan pada halaman daftar latihan soal :
   1. Judul halaman
   2. Tombol untuk navigasi ke fitur contoh soal. Fitur ini akan memberikan tutorial dengan langkah - langkah untuk mengenalkan fungsionalitas dari aplikasi
   3. Nama latihan soal
   4. ID latihan soal
   5. Tombol untuk mengerjakan latihan soal. Ketika pelajar belum pernah mengerjakan latihan soal maka tombol ini akan memiliki teks "Kerjakan" sedangkan jika pelajar sudah pernah mengerjakan latihan soal maka teks akan berubah menjadi "Kerjakan Lagi"

Background pattern

Description automatically generated with medium confidence

Gambar 2.6 Informasi Komponen pada Halaman Daftar Latihan Soal

1. Tekan tombol "Kerjakan" pada salah satu latihan soal yang ingin dikerjakan.
2. Selanjutnya halaman pengerjaan latihan soal akan ditampilkan.

Graphical user interface, table

Description automatically generated

Gambar 2.7 Halaman Pengerjaan Latihan Soal

1. Berikut adalah penjelasan dari komponen - komponen yang ditampilkan pada halaman daftar latihan soal :
   1. Nama latihan soal
   2. Instruksi latihan soal
   3. Susunan blok kode yang disediakan. Pada kotak merah, terdapat susunan blok kode yang sudah disediakan secara acak. Hal ini merupakan implementasi dari Pre-scaffolded Parsons Probelm. Dalam pengerjaanya, pelajar dapat mengurutkan blok - blok kode tersebut ke urutan yang benar dengan cara drag - drop dari atas ke bawa (atau sebaliknya)
   4. Informasi waktu pengerjaan
   5. Gambar output yang diharapkan. Gambar ini merupakan output yang diharapkan dari susunan blok kode, pelajar dapat memanfaatkan gambar ini untuk menyusun blok kode ke urutan yang benar
   6. Tombol kerjakan latihan. Tombol ini digunakan untuk memulai pengerjaan latihan soal.

Graphical user interface, table

Description automatically generated

Gambar 2.8 Informasi Komponen pada Halaman Pengerjaan Latihan Soal

1. Tekan tombol "Kerjakan Latihan"
2. Selanjutnya dialog konfirmasi akan ditampilkan. Dialog ini digunakan untuk melakukan konfirmasi ulang apakah pelajar sudah yakin untuk mengerjakan latihan. Jika pelajar yakin untuk mengerjakan, maka pelajar dapat menekan tombol "Yakin", namun jika tidak maka pelajar dapat menekan tombol "Tidak". Ketika pelajar memilih "Yakin", maka waktu pengerjaan akan berjalan dan setiap pergerakan yang dilakukan oleh pelajar akan direkam oleh aplikasi.

Graphical user interface, text, application, email

Description automatically generated

Gambar 2.9 Dialog Konfirmasi Pengerjaan Latihan Soal

1. Jika telah memilih yakin, maka akan terdapat perubahan pada halaman di mana terdapat 2 tombol baru yakni tombol "Cek Jawaban" dan "Kirim Jawaban". Berikut merupakan penjelasan dari tombol tersebut :
   1. Tombol "Cek Jawaban". Tombol ini dapat dimanfaatkan oleh pelajar untuk mengecek apakah susunan blok kode yang telah dikerjakan telah sesuai dengan output yang diharapkan atau belum.
   2. Tombol "Kirim Jawaban". Tombol ini dapat digunakan oleh pelajar apabila susunan blok kode yang disusun telah sesuai dengan output yang diharapkan. Dengan menekan tombol ini maka telah menyelesaikan latihan soal yang dipilih.

Chart, waterfall chart

Description automatically generated

Gambar 2.10 Informasi Perubahan Halaman Setelah Konfirmasi Pengerjaan Latihan Soal

1. Jika pelajar telah menekan tombol "Kirim Jawaban" maka dialog informasi bahwa pelajar telah menyelesaikan latihan soal akan ditampilkan. Dan ketika pelajar memilik "OK", maka pelajar akan kembali ke halaman daftar latihan soal dan dapat melanjutkan ke soal berikutnya.

Graphical user interface, text, application

Description automatically generated

Gambar 2.11 Dialog Penyelesaian Latihan Soal

# PROJECT

## Struktur Project

Proses penerapan Pre-scaffolded Parsons Problem dilakukan pada module "code\_exercise" seperti pada (Gambar 3.1 Struktur Project. Pada penerapannya, struktur project dibagi menjadi 3 yakni views, controller, dan bindings. Folder **views** digunakan untk mengelola tampilan halaman, folder **controllers** digunakan untuk mengatur business logic dari halaman yang akan ditampilkan, dan folder **bindings** digunakan untuk binding controller ke dalam view. Pemisahan 3 folder ini diimplementasikan dengan tujuan memisahkan tanggung jawab dari setiap file (separation of concern), sehingga class view hanya akan bertanggung jawab untuk menampilkan tampilan tanpa harus khawatir akan business logic. Sebaliknya, class controller hanya akan bertanggung jawab untuk mengelola data yang dibutuhkan oleh view.

Graphical user interface

Description automatically generated

Gambar 3.1 Struktur Project

## Source Code

Penerapan Pre-scaffolded Parsons Problem dilakukan dengan menggunakan bahasa pemrograman *dart* dan menggunakan Visual Studio Code sebagai teks editor. Berikut merupakan code\_exercise\_controller.dart dimana pada file ini terdapat fungsi utama untuk menyediakan soal, mengacak blok kode, mengecek jawaban, dan menyimpan log.

|  |
| --- |
| class CodeExerciseController extends GetxController {  final \_storageHelper = Get.find<StorageHelper>();  final logReference = FirebaseFirestore.instance.collection("log");  final exerciseId = Get.arguments[0] as String;  final exerciseName = Get.arguments[1] as String;  final exerciseCaption = Get.arguments[2] as String;  final exerciseCodeBlocks = Get.arguments[3] as List<Map<String, Object>>;  final exerciseOutput = Get.arguments[4] as String;  final dialogMessage =  "Susunlah blok - blok kode yang sudah disediakan sesuai dengan output yang diharapkan. Ketika Anda menekan tombol \"Yakin\", maka waktu akan dimulai dan waktu akan secara otomatis berhenti ketika Anda telah menyusun blok kode dengan benar.";  final stopwatchTimer = StopWatchTimer();  var isStarted = false.obs;  var isCorrect = false.obs;  var isFinished = false.obs;  var steps = 0.obs;  var correctAnswer = <Map<String, Object>>[].obs;  var studentAnswer = <Map<String, Object>>[].obs;  @override  void onInit() {  super.onInit();  correctAnswer = exerciseCodeBlocks.obs;  studentAnswer = exerciseCodeBlocks.obs;  \_mixAnswer();  }  @override  void onClose() {  stopwatchTimer.dispose();  super.onClose();  }  void \_mixAnswer() {  while (CheckAnswerHelper.isDeepEqual(correctAnswer, studentAnswer)) {  studentAnswer.shuffle();  }  }  void onReorder(int oldIndex, int newIndex) {  //reordering list  final index = newIndex > oldIndex ? newIndex - 1 : newIndex;  final oldItem = studentAnswer.removeAt(oldIndex);  studentAnswer.insert(index, oldItem);  //setup log  var studentAnswerLog = "[";  studentAnswer.forEach((element) {  studentAnswerLog += "${element["keyValue"]},";  });  studentAnswerLog += "]";  steps.value++;  //make log model  final log = LogModel(  logId: "$exerciseId-${DateTime.now()}",  exerciseId: exerciseId,  studentId: \_storageHelper.getIdUser(),  studentName: \_storageHelper.getNameUser(),  answer: studentAnswerLog,  step: steps.value.toString(),  time: \_getStopwatchTime(),  timeStamp: DateTime.now().toString(),  );  // uploading log to the firestore  logReference.add({  "id\_log": log.logId,  "id\_mahasiswa": log.studentId,  "nama\_mahasiswa": log.studentName,  "id\_latihan": log.exerciseId,  "langkah": log.step,  "waktu": log.time,  "jawaban": log.answer,  "time\_stamp": log.timeStamp,  });  }  void checkAnswer(BuildContext context) {  final correctCodeValue = <String>[];  correctAnswer.forEach((element) {  correctCodeValue.add("${element['codeValue']}");  });  final studentCodeValue = <String>[];  studentAnswer.forEach((element) {  studentCodeValue.add("${element['codeValue']}");  });  isCorrect.value =  CheckAnswerHelper.isDeepEqual(correctCodeValue, studentCodeValue);  if (isCorrect.value) {  stopTimer();  isFinished.toggle();  SnackBarHelper.showFlushbarSuccess(  "Selamat",  "Blok kode yang Anda susun telah sesuai dengan output yang diharapkan",  )..show(context);  return;  } else {  SnackBarHelper.showFlushbarWarning(  "Peringatan",  "Blok kode yang Anda susun belum sesuai dengan output yang diharapkan",  )..show(context);  }  }  void startTimer() {  stopwatchTimer.onExecute.add(StopWatchExecute.start);  }  void stopTimer() {  stopwatchTimer.onExecute.add(StopWatchExecute.stop);  }  void dialogStartOnSuccess() {  isStarted.toggle();  Get.back();  startTimer();  }  String \_getStopwatchTime() {  final rawValue = stopwatchTimer.rawTime.value;  final displayTime = StopWatchTimer.getDisplayTime(rawValue, hours: true);  return displayTime;  }  void sendAnswer() {  SendAnswerHelper.updateExercise(exerciseId);  SendAnswerHelper.updateStudent(exerciseId);  }  } |

### Menyediakan Latihan Soal

Proses menyediakan latihan soal terjadi pada baris di bawah ini. Terdapat sebuah variable bernama **exerciseCodeBlocks**, variable ini digunakan untuk menyimpan array blok kode yang didapatkan dari argument.

|  |
| --- |
| final exerciseCodeBlocks = Get.arguments[3] as List<Map<String, Object>>; |

### Mengacak Blok Kode

Proses mengacak blok kode terjadi pada fungsi **onInit**() dengan melakukan override pada fungsi **onInit**() milik parent controller. Pada fungsi tersebut, terjadi pemanggilan fungsi \_**mixAnswer**(). Pada fungsi \_**mixAnswer**() terdapat looping dengan while dimana pada kondisi tersebut blok kode akan dilakukan .**shuffle**()

|  |
| --- |
| @override  void onInit() {  super.onInit();  correctAnswer = exerciseCodeBlocks.obs;  studentAnswer = exerciseCodeBlocks.obs;  \_mixAnswer();  }  void \_mixAnswer() {  while (CheckAnswerHelper.isDeepEqual(correctAnswer, studentAnswer)) {  studentAnswer.shuffle();  }  } |

### Mengecek Jawaban

Proses mengecek jawaban dilakukan pada fungsi checkAnswer(). Pada fungsi tersebut dilakukan pengecekan jawaban dengan membandingkan apakah variable **correctCodeValue** memiliki value yang sama dengan variable **studentCodeValue**. Jika sama, maka susunan blok kode yang dijawab oleh pelajar sudah sesuai dengan output yang diharapkan. Selanjutnya aplikasi akan menampilkan snackbar sebagai informasi.

|  |
| --- |
| void checkAnswer(BuildContext context) {  final correctCodeValue = <String>[];  correctAnswer.forEach((element) {  correctCodeValue.add("${element['codeValue']}");  });  final studentCodeValue = <String>[];  studentAnswer.forEach((element) {  studentCodeValue.add("${element['codeValue']}");  });  isCorrect.value =  CheckAnswerHelper.isDeepEqual(correctCodeValue, studentCodeValue);  if (isCorrect.value) {  stopTimer();  isFinished.toggle();  SnackBarHelper.showFlushbarSuccess(  "Selamat",  "Blok kode yang Anda susun telah sesuai dengan output yang diharapkan",  )..show(context);  return;  } else {  SnackBarHelper.showFlushbarWarning(  "Peringatan",  "Blok kode yang Anda susun belum sesuai dengan output yang diharapkan",  )..show(context);  }  } |

### Menyimpan Log

Proses menyimpan log terjadi pada fungsi onReorder. Pada fungsi ini aplikasi akan mengatur posisi blok kode sesuai dengan perubahan yang dilakukan oleh pelajar, selanjutnya aplikasi akan menyimpan informasi yang diperlukan sebagai log. Informasi yang disimpan yakni **logId, exerciseId, studentId, studentName, answer, step, time, dan timeStamp.**

|  |
| --- |
| void onReorder(int oldIndex, int newIndex) {  //reordering list  final index = newIndex > oldIndex ? newIndex - 1 : newIndex;  final oldItem = studentAnswer.removeAt(oldIndex);  studentAnswer.insert(index, oldItem);  //setup log  var studentAnswerLog = "[";  studentAnswer.forEach((element) {  studentAnswerLog += "${element["keyValue"]},";  });  studentAnswerLog += "]";  steps.value++;  //make log model  final log = LogModel(  logId: "$exerciseId-${DateTime.now()}",  exerciseId: exerciseId,  studentId: \_storageHelper.getIdUser(),  studentName: \_storageHelper.getNameUser(),  answer: studentAnswerLog,  step: steps.value.toString(),  time: \_getStopwatchTime(),  timeStamp: DateTime.now().toString(),  );  // uploading log to the firestore  logReference.add({  "id\_log": log.logId,  "id\_mahasiswa": log.studentId,  "nama\_mahasiswa": log.studentName,  "id\_latihan": log.exerciseId,  "langkah": log.step,  "waktu": log.time,  "jawaban": log.answer,  "time\_stamp": log.timeStamp,  });  } |

# KESIMPULAN

Berdasarkan hasil penerapan Pre-scaffolded Parsons Problem pada aplikasi EasyFlutter diperoleh kesimpulan bahwa aplikasi pengayaan berbasis web bernama EasyFlutter dapat digunakan sebagai inovasi pembelajaran Flutter Layout pada Pemrograman Mobile dan memiliki dampak positif yang signifikan pada uji statistik Paired T-test.