



4 -

076 – Hanifidin Ibrahim – 10/02/2026

Link github :

https://github.com/Hanif13579/PY4_2ABC_D3_2024_Modul1_001

Daftar Isi

MODUL 1: Fondasi, Setup, dan Prinsip SRP	4
Tujuan Pembelajaran.....	4
Materi Esensial	4
Target Milestones	4
Materi untuk Praktikum	4
1. 1 Persiapan Alat dan Bahan	4
Trouble Shooting.....	7
1. 2 Menjalankan Proyek Pertama (Counter).....	11
Trouble Shooting Gradle	13
1.3 Refactoring: Belajar Menjadi Arsitek (SRP).....	15
A. Struktur Berkas & Folder Akhir (Minggu 1)	15
B. Membuat Si "Otak" (Controller).....	15
C. Membuat Si "Wajah" (View)	15
1.4 Rangkuman dan Trouble Shooting.....	17
Tugas Praktikum.....	18
Task 1: The Multi-Step Counter (Low Order Thinking - LOTS).....	18
Task 2: The History Logger (High Order Thinking - HOTS)	18
Homework (30%).....	19
Penilaian	20
Kuesioner Self-Assessment.....	20
Peer Assessment (Apresiasi Rekan Sejawat 1).....	21
Peer Assessment (Apresiasi Rekan Sejawat 2).....	22
Peer Assessment (Apresiasi Rekan Sejawat 3).....	Error! Bookmark not defined.
Template Lesson Learnt (Refleksi Akhir)	23
Log LLM: The Fact Check & Twist (Homework)	24
Contoh Cara Bertanya yang Benar (Prompting)	Error! Bookmark not defined.
Rubrik Penilaian Dosen Manajer	25
Rubrik Penilaian Log LLM (Integritas).....	25
Referensi	26

MODUL 1: Fondasi, Setup, dan Prinsip SRP

Di modul pertama ini, kita akan melakukan instalasi tools pemrograman mobile (Flutter) dan mengenal satu prinsip utama dalam dunia pengembangan perangkat lunak: **Single Responsibility Principle (SRP)**. Mari kita mulai!

Tujuan Pembelajaran

Setelah menyelesaikan modul ini, diharapkan Anda mampu:

1. Melakukan instalasi dan konfigurasi Flutter SDK secara mandiri.
2. Menghubungkan dan menjalankan aplikasi pada perangkat Android fisik.
3. Memahami dan menerapkan pola pikir SRP dalam memisahkan logika dan antarmuka.
4. Mengenal sintaks dasar bahasa pemrograman Dart (Class, Variable, Encapsulation).

Materi Esensial

Sebelum kita mulai menulis kode, pahami dahulu dua pilar utama hari ini:

1. Separation of Concerns (SoC): Konsep membagi program menjadi bagian-bagian kecil yang tidak saling tumpang tindih. Di Flutter, kita membagi mana yang bertugas "berpikir" (Controller) dan mana yang bertugas "tampil estetis" (View).
2. Reactivity: Flutter adalah reactive framework. Artinya, UI (tampilan) akan berubah secara otomatis jika ada perubahan pada state (data) melalui fungsi setState().

Target Milestones

Sebelum pulang dari lab hari ini, pastikan kalian sudah mencentang semua daftar di bawah ini:

TABEL 1.1: Daftar Target Milestones Modul 1 dan Indikator Keberhasilan.

Milestone	Indikator Keberhasilan
Environment Ready	flutter doctor tidak menunjukkan error fatal dan HP fisik terdeteksi.
First Run	Aplikasi Counter bawaan berhasil muncul di layar HP kalian.
The SRP Refactor	Kode aplikasi terpecah menjadi file Main, View, dan Controller.
Logic Extension	Berhasil menambah fungsi decrement (kurang) dan reset pada Controller.

Materi untuk Praktikum

1.1 Persiapan Alat dan Bahan

Flutter itu ibarat mesin balap; ia membutuhkan landasan yang tepat. Jangan asal melakukan instalasi!

1. Download cmd tools dari tautan <https://developer.android.com/studio>

Command line tools only

Platform	SDK tools package	Size	SHA-256 checksum
Windows	commandlinetools-win-13114758_latest.zip	143.0 MB	98b565cb657b012dae6794cefc0f66ae1efb4690c699b78a614b4a6a3505b003
Mac	commandlinetools-mac-13114758_latest.zip	143.3 MB	5673201e6f3869f418eed3b5cb6c4be7401502bd0aae1b12a29d164d647a54e
Linux	commandlinetools-linux-13114758_latest.zip	164.8 MB	7ec965280a073311c339e571cd5de778b9975026cfcbc79f2b1cdcb1e15317ee

Command-line tools are included in Android Studio. If you do not need Android Studio, you can download the basic Android command-line tools above. You can use the included [sdkmanager](#) to download other SDK packages.

GAMBAR 1.1: Unduhan Command Line (CMD) Tools Android (~140MB).

2. Download SDK flutter dari tautan berikut <https://docs.flutter.dev/install/manual> pilih sesuai sistem operasi anda, Fluttter yang digunakan pada praktikum ini adalah versi 3.38.7-stable.zip

Install and set up Flutter

To install the Flutter SDK, download the latest bundle from the SDK archive, then extract the SDK to where you want it stored.

- 1 Download the Flutter SDK bundle

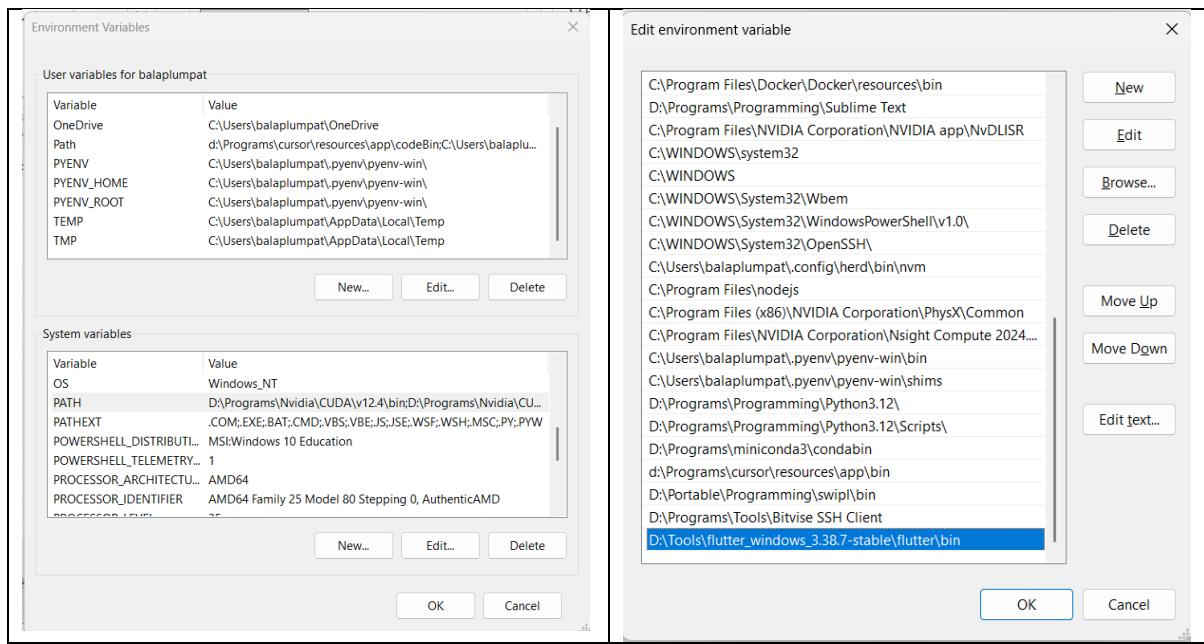
Download the following installation bundle to get the latest stable release of the Flutter SDK.

[flutter_windows_3.38.7-stable.zip](#)

GAMBAR 1.2: Unduhan SDK Flutter (1.7GB).

3. Ekstrak SDK: Letakkan Flutter SDK di folder yang aman, contohnya D:\Tools\Flutter\flutter_windows_3.38.7-stable\flutter. Hindari folder Program Files karena Windows sering kali sangat protektif soal perizinan berkas (permission) yang bisa menghambat kinerja Flutter.
4. Pengaturan Path: Agar terminal Anda mengenali Flutter, daftarkan folder bin ke dalam Environment Variables di Windows.
5. Pada praktikum ini kita menggunakan Visual Studio Code, install plugin Flutter dan Dart dari dart code.

GAMBAR 1.3: Tampilan antarmuka Visual Studio Code yang telah terintegrasi dengan ekstensi Flutter dan Dart.



GAMBAR 1.4: Pengaturan Environment Variable PATH untuk mendaftarkan lokasi folder bin Flutter..

6. **Cek Kesehatan:** Buka terminal, lalu ketik mantra sakti ini:

```
flutter doctor
```

```
Building flutter tool...
Running pub upgrade...
Resolving dependencies... (1.0s)
Downloading packages... (9.6s)
Got dependencies.
Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, 3.38.7, on Microsoft Windows [Version 10.0.26200.7462], locale en-ID)
[✓] Windows Version (11 Education 64-bit, 25H2, 2009)
[✗] Android toolchain - develop for Android devices
    X Unable to locate Android SDK.
        Install Android Studio from: https://developer.android.com/studio/index.html
        On first launch it will assist you in installing the Android SDK components.
        (or visit https://flutter.dev/to/windows-android-setup for detailed instructions).
        If the Android SDK has been installed to a custom location, please use
        'flutter config --android-sdk' to update to that location.

[✓] Chrome - develop for the web
[✗] Visual Studio - develop Windows apps
    X Visual Studio not installed; this is necessary to develop Windows apps.
        Download at https://visualstudio.microsoft.com/downloads/.
        Please install the "Desktop development with C++" workload, including all of its default components
[✓] Connected device (3 available)
[✓] Network resources

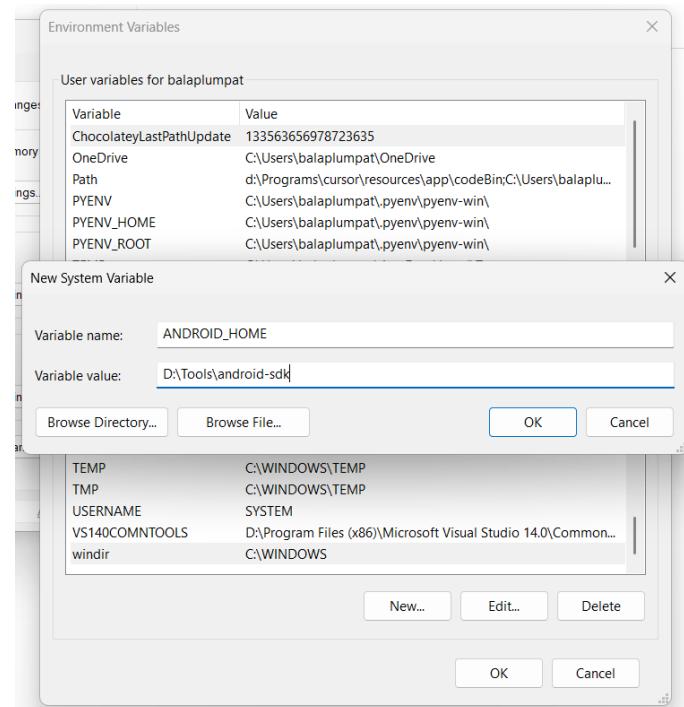
! Doctor found issues in 2 categories.
```

GAMBAR 1.5: Pengecekan ulang flutter doctor setelah Path Flutter terdaftar.

Pastikan Android Toolchain dan VS Code sudah bertanda centang hijau. Jika masih ada yang berwarna merah, jangan panik. Baca pesannya baik-baik, biasanya Flutter sudah memberikan instruksi cara memperbaikinya.

Trouble Shooting

1. Android SDK (<https://docs.flutter.dev/platform-integration/android/setup>)
 - Pertama buat folder android-sdk pada Komputer anda misal (D:\Tools\android-sdk)
 - Setup Environment dengan membuat variable ANDROID_HOME dengan nilai folder "D:\Tools\android-sdk"

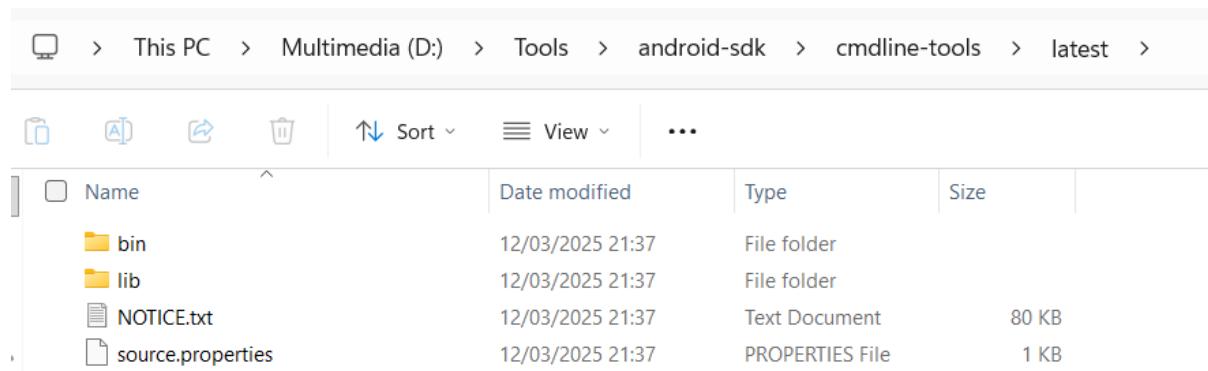


GAMBAR 1.6: Konfigurasi variabel lingkungan ANDROID_HOME pada System Variables Windows.

- Pindahkan folder cmdline-tools yang sudah diekstrak ke dalam folder android-sdk

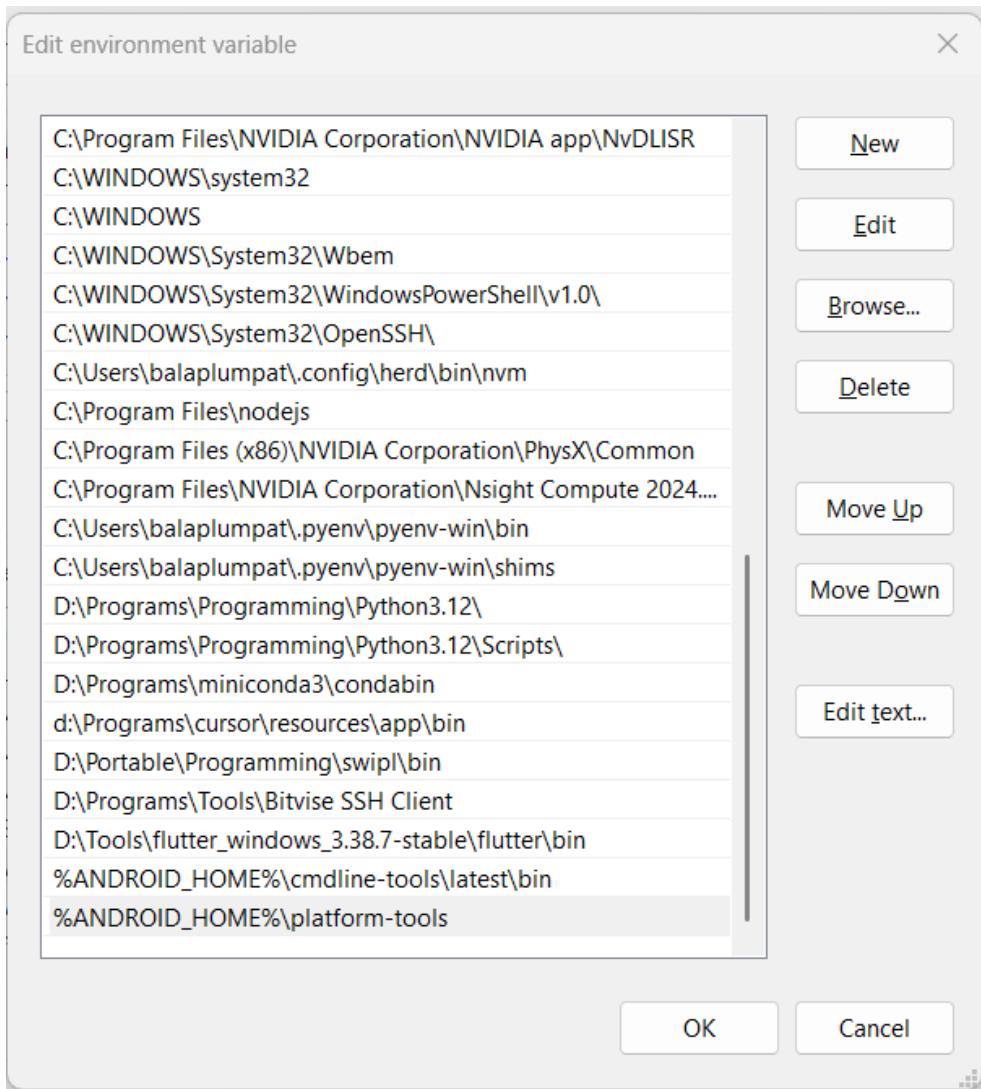
GAMBAR 1.7: Lokasi penempatan folder cmdline-tools di dalam direktori android-sdk.

- Selain itu perlu, penyesuaian folder karena android-sdk cenderung kaku dengan membuat new folder latest lalu pindahkan (bin, lib, dsb) ke folder latest yang baru sehingga memiliki struktur berikut.



GAMBAR 1.8: Struktur folder latest yang berisi binari utama Android SDK Command-line Tools.

- Setup Environment PATH untuk bin cmdline (tools cmd android studio)
%ANDROID_HOME%\cmdline-tools\latest\bin
%ANDROID_HOME%\platform-tools



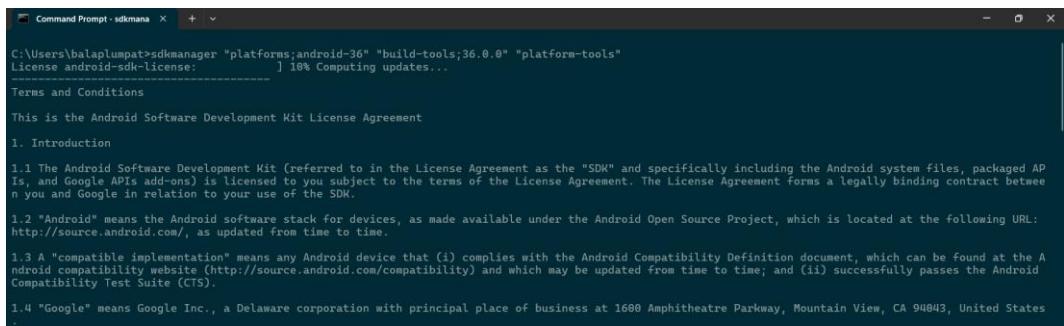
GAMBAR 1.9: Penambahan Path sistem untuk direktori bin pada cmdline-tools dan platform-tools.

- Check Env berjalan dengan sdkmanager --list

GAMBAR 1.10: Verifikasi koneksi SDK dengan menampilkan daftar paket yang tersedia melalui sdkmanager.

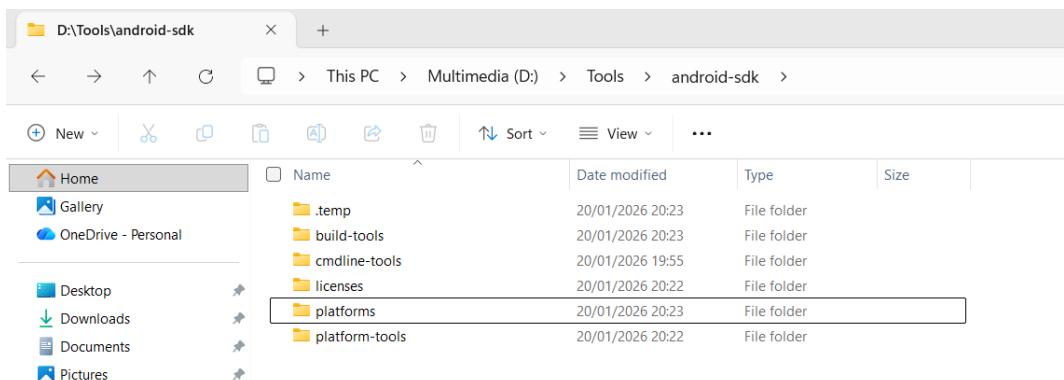
- Jalankan perintah berikut pada terminal / cmd dan pilih accept "Y"

```
sdkmanager "platforms;android-36" "build-tools;36.0.0" "platform-tools"
```



GAMBAR 1.11: Proses persetujuan lisensi (Terms and Conditions) melalui terminal sebelum instalasi SDK.

- Pastikan folder %ANDROID_HOME% anda sudah seperti ini



GAMBAR 1.12: Struktur akhir direktori Android SDK setelah komponen platform dan build-tools terinstal.

- Langkah terakhir aktifkan flutter dan android sdk dengan perintah konfigurasi

```
flutter config --android-sdks "D:\Tools\android-sdk"
flutter doctor --android-licenses
```

- Accept TOC

```
Microsoft Windows [Version 10.0.26200.7462]
(c) Microsoft Corporation. All rights reserved.

C:\Users\balaplumpat> flutter doctor --android-licenses
[=====] 100% Computing updates...
6 of 7 SDK package licenses not accepted.
Review licenses that have not been accepted (y/N)?
```

GAMBAR 1.13: Tahap akhir verifikasi lisensi Android menggunakan perintah flutter doctor -- android-licenses.

2. Visual Studio untuk Windows

- Karena kita tidak perlu membuat aplikasi windows, fokus pada android maka tidak perlu install Visual Studio Community yang besarnya 5-10GB.
- Jalankan perintah berikut

```
flutter config --no-enable-windows-desktop
```

3. Matikan telemetry yang default dijalankan oleh flutter

```
flutter config --no-analytics
```

4. Jalankan kembali “flutter doctor” dan pastikan semua checklist tercapai

GAMBAR 1.14: Tampilan output flutter doctor yang sukses menunjukkan lingkungan pengembangan telah siap.

1.2 📱 Menjalankan Proyek Pertama (Counter)

Kita tidak akan menggunakan emulator yang berat. Kita akan menggunakan ponsel fisik agar pengalaman pengembangan lebih nyata.

1. Aktifkan Developer Options dan USB Debugging pada ponsel Anda.
<https://youtube.com/shorts/jhZTiKM8IRw?si=JGUVoPZagVGJ6VZc>

GAMBAR 1.15: Langkah aktivasi fitur "USB Debugging" pada menu Developer Options di perangkat Android.

2. Buat proyek baru dengan perintah pada cmd atau terminal:

```
flutter create logbook_app_001
```

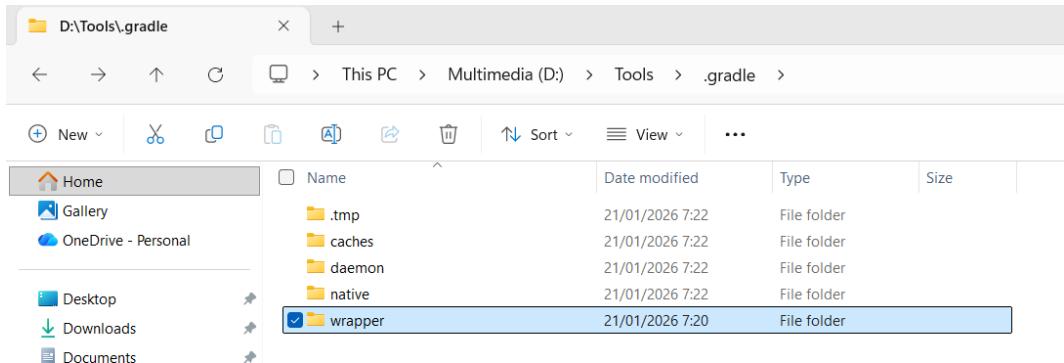
3. Sambungkan flutter dan device

```
flutter devices
```

GAMBAR 1.16: Daftar perangkat yang terhubung dan siap digunakan melalui perintah flutter devices.

4. Jalankan aplikasinya:

```
flutter run
```



GAMBAR 1.17: Proses inisialisasi dan pengunduhan dependensi oleh Gradle saat pertama kali aplikasi dijalankan.

```
Flutter run key commands.
r Hot reload.
R Hot restart.
h List all available interactive commands.
d Detach (terminate "flutter run" but leave application running).
c Clear the screen
q Quit (terminate the application on the device).

A Dart VM Service on Nokia 5 4 is available at: http://127.0.0.1:50576/qMLh3Yfb1qs=/
The Flutter DevTools debugger and profiler on Nokia 5 4 is available at:
http://127.0.0.1:50576/qMLh3Yfb1qs=/devtools/?uri-ws://127.0.0.1:50576/qMLh3Yfb1qs=/ws

I/choreographer(10781): Skipped 235 frames!  The application may be doing too much work on its main thread.
I/AdrenoGLES-0(10781): QUALCOMM build          : 2bd16eb911, I3687d7cf91
I/AdrenoGLES-0(10781): Build Date              : 02/10/21
I/AdrenoGLES-0(10781): OpenGL ES Shader Compiler Version: EV031.32.02.08
I/AdrenoGLES-0(10781): Local Branch           :
I/AdrenoGLES-0(10781): Remote Branch          :
I/AdrenoGLES-0(10781): Remote Branch          :
I/AdrenoGLES-0(10781): Reconstruct Branch     :

Ln 14, Col 78  Spaces: 2
```

GAMBAR 1.18: Status log terminal yang menunjukkan aplikasi berhasil dikompilasi dan dijalankan (Run).

5. Perhatikan kode utama pada lib/main.dart

```
File: logbook_app_001\lib\main.dart
01: import 'package:flutter/material.dart';
02:
03: void main() {
04:   runApp(const MyApp());
05: }
06:
07: class MyApp extends StatelessWidget {
08:   const MyApp({super.key});
09:
10:   @override
11:   Widget build(BuildContext context) {
12:     return MaterialApp(
13:       title: 'Flutter Demo',
14:       theme: ThemeData(colorScheme: .fromSeed(seedColor:
Colors.deepPurple)),
15:       home: const MyHomePage(title: 'Flutter Demo Home Page'),
16:     );
17:   }
18: }
19:
20: class MyHomePage extends StatefulWidget {
21:   const MyHomePage({super.key, required this.title});
22:
23:   final String title;
24:
25:   @override
26:   State<MyHomePage> createState() => _MyHomePageState();
27: }
28:
```

```

29: class _MyHomePageState extends State<MyHomePage> {
30:   int _counter = 0;
31:
32:   void _incrementCounter() {
33:     setState(() {
34:       _counter++;
35:     });
36:   }
37:
38:   @override
39:   Widget build(BuildContext context) {
40:     return Scaffold(
41:       appBar: AppBar(
42:         backgroundColor: Theme.of(context).colorScheme.inversePrimary,
43:         title: Text(widget.title),
44:       ),
45:       body: Center(
46:         child: Column(
47:           mainAxisAlignment: .center,
48:           children: [
49:             const Text('You have pushed the button this many times:'),
50:             Text(
51:               '_$counter',
52:               style: Theme.of(context).textTheme.headlineMedium,
53:             ),
54:           ],
55:         ),
56:       ),
57:       floatingActionButton: FloatingActionButton(
58:         onPressed: _incrementCounter,
59:         tooltip: 'Increment',
60:         child: const Icon(Icons.add),
61:       ),
62:     );
63:   }
64: }
```

Trouble Shooting Gradle

Pernahkah Anda merasa flutter run pertama kali terasa sangat lama, seolah-olah komputer sedang membeku? Itu bukan komputer Anda yang rusak, melainkan Gradle sedang bekerja keras di balik layar. Mari kita atur strategi agar proses ini tidak menjadi hambatan.

1. Di mana posisi Gradle?

Gradle tidak termasuk di dalam Android SDK. Namun, Flutter menggunakan sistem yang disebut Gradle Wrapper.

- Saat mahasiswa membuat proyek dengan flutter create, Flutter otomatis menyertakan berkas "instruksi" di dalam folder android/gradle/wrapper/.

- Saat pertama kali flutter run dijalankan, si "instruksi" ini akan mendownload sendiri file Gradle yang dibutuhkan (versi yang spesifik dan paling cocok) ke folder %GRADLE_USER_HOME% atau D:\Tools\.gradle yang kita akan atur.
2. Memahami Cara Kerja Gradle (Offline & Online)
- Secara bawaan, Gradle akan menyimpan semua file "unduhan"-nya di folder C:\Users\NamaUser\.gradle. Masalahnya, folder ini sering membengkak hingga bergiga-giga dan membebani partisi sistem (C:). Solusi Arsitek: Pindahkan "Gudang" Gradle ke folder D:\Tools agar partisi sistem Anda tetap lega.
- Langkah Memindahkan Folder .gradle:
- Buat folder baru di D:\Tools\.gradle
 - Buka Environment Variables.
 - Tambahkan User Variable baru: **Variable name:** GRADLE_USER_HOME, **Variable value:** D:\Tools\.gradle
3. Standarisasi Versi Gradle
- Mahasiswa sering mengalami error karena versi Gradle yang digunakan antar-anggota kelompok berbeda-beda. Tips Populer: Selalu gunakan Gradle Wrapper. Jangan pernah menginstal Gradle secara manual di komputer mahasiswa. Biarkan setiap proyek Flutter menentukan versinya sendiri melalui file: android/gradle/wrapper/gradle-wrapper.properties
4. Menghadapi "First Run" yang Berat
- Proses flutter run pertama membutuhkan internet yang sangat stabil karena Gradle akan mengunduh semua kebutuhan Android (sekitar 500MB - 1GB).
- Best Practice Praktikum:
- Proses ini hanya terjadi satu kali di awal proyek, jalankan sebelum praktikum Proyek 4 dimulai.
 - **Copy-Paste Cache:** Mahasiswa yang berhasil bisa memberikan folder .gradle yang sudah "berisi" kepada teman anda melalui Flashdisk. Mereka cukup menyalinnya ke D:\Tools\.gradle. Dengan begitu, mahasiswa tidak perlu mengunduh lagi dari nol.
5. Mengatasi Masalah Ukuran File yang Besar
- Jika folder .gradle mulai memenuhi ruang disk:
- Perintah Pembersihan: Mahasiswa bisa menjalankan flutter clean untuk membersihkan file sementara di proyek.
 - Hapus Cache Lama: Folder .gradle/caches bisa dihapus secara berkala, namun ingat, setelah dihapus, proses unduh akan dimulai lagi dari awal saat build berikutnya.
6. Tabel Troubleshooting Gradle (Anti-Stuck)

Gejala	Penyebab	Solusi
Build stuck di Running Gradle task 'assembleDebug'...	Koneksi terputus saat mengunduh dependency.	Tutup terminal, hubungkan ke internet yang stabil, lalu jalankan ulang.
Error: Minimum SDK version mismatch	Versi Android di proyek terlalu rendah untuk package tertentu.	Ubah minSdkVersion di file android/app/build.gradle (biasanya ke 21 atau 23).

Connection Timeout	Firewall atau Antivirus memblokir Gradle.	Matikan sementara Antivirus atau berikan izin (Exception) untuk aplikasi Java/Gradle.
---------------------------	---	---

1.3 Refactoring: Belajar Menjadi Arsitek (SRP)

Pernahkah Anda melihat berkas main.dart bawaan Flutter? Sangat panjang dan campur aduk, bukan? Tampilan dan logika perhitungan menumpuk menjadi satu. Dalam dunia profesional, ini disebut Spaghetti Code.

Kita akan menerapkan prinsip SRP (Single Responsibility Principle). Intinya: Satu berkas hanya boleh memiliki satu tanggung jawab.

A. Struktur Berkas & Folder Akhir (Minggu 1)

```
logbook_app/
├── lib/
│   ├── counter_controller.dart    <-- [Logika: Angka, Step, List Riwayat]
│   ├── counter_view.dart         <-- [Antarmuka: Tombol, Teks, Dialog, SnackBar]
│   └── main.dart                 <-- [Entry Point: Menjalankan MyApp & CounterView]
└── pubspec.yaml                <-- [Konfigurasi proyek]
└── test/                        <-- [Tempat pengujian, abaikan sementara]
```

GAMBAR 1.20: Struktur folder proyek logbook_app setelah menerapkan prinsip SRP (Controller & View).

B. Membuat Si "Otak" (Controller)

Buat berkas baru lib/counter_controller.dart. Ia hanya bertugas menghitung, ia tidak peduli bagaimana bentuk tombolnya.

```
class CounterController {
  int _counter = 0; // Variabel private (Enkapsulasi)

  int get value => _counter; // Getter untuk akses data

  void increment() => _counter++;
  void decrement() { if (_counter > 0) _counter--; }
  void reset() => _counter = 0;
}
```

C. Membuat Si "Wajah" (View)

Buat berkas baru lib/counter_view.dart. Berkas ini bertugas menggambar layar.

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

class CounterView extends StatefulWidget {
```

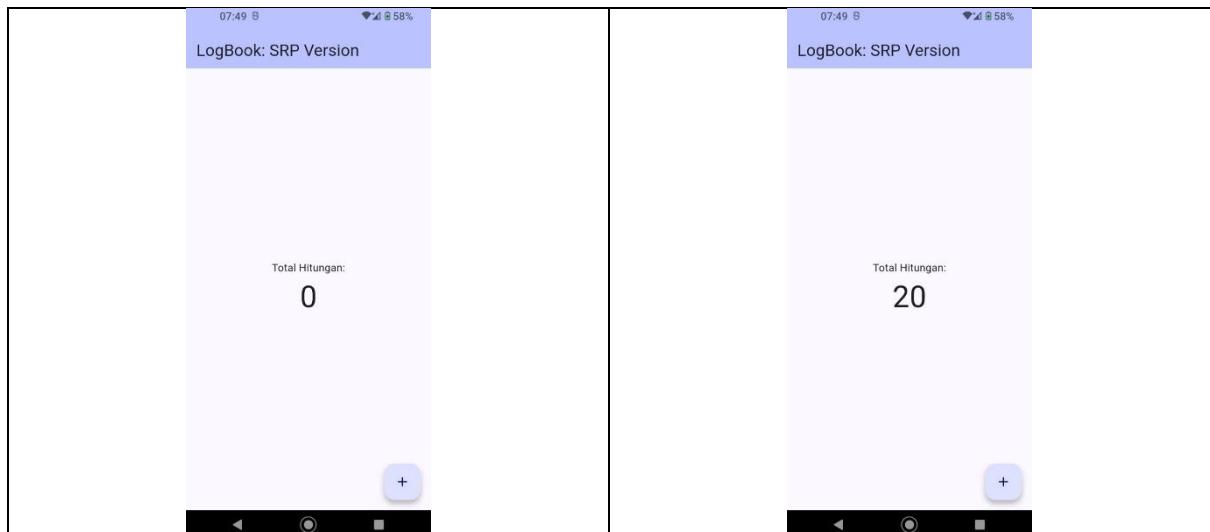
```

const CounterView({super.key});
@Override
State<CounterView> createState() => _CounterViewState();
}

class _CounterViewState extends State<CounterView> {
final CounterController _controller = CounterController();

@Override
Widget build(BuildContext context) {
return Scaffold(
appBar: AppBar(title: const Text("LogBook: Versi SRP")),
body: Center(
child: Column(
mainAxisAlignment: MainAxisAlignment.center,
children: [
const Text("Total Hitungan:"),
Text('${_controller.value}', style: const TextStyle(fontSize:
40)),
],
),
),
floatingActionButton: FloatingActionButton(
onPressed: () => setState(() => _controller.increment()),
child: const Icon(Icons.add),
),
);
}
}

```



GAMBAR 1.21: Penampakan aplikasi "Counter" modifikasi SRP pada perangkat fisik Android.

1.4 Rangkuman dan Trouble Shooting

Untuk kalian yang baru kenal Dart, berikut adalah "kamus kecil" dari kode yang kita tulis tadi:

TABEL 1.3: Glosarium kata kunci dan sintaks dasar Dart pada Modul 1

Keyword	Penjelasan Sederhana
Final	Variabel yang nilainya tidak akan berubah setelah pertama kali diisi.
_ (Underscore)	Membuat variabel menjadi <i>Private</i> (Hanya bisa diakses di dalam filenya sendiri).
setState()	Perintah ke Flutter: "Ada perubahan data load ulang scene"
Class	Cetakan atau blueprint untuk membuat objek.

TABEL 1.4: Troubleshooting: Modul 1

Kendala / Pesan Error	Kemungkinan Penyebab	Solusi
'flutter' is not recognized...	Folder bin Flutter belum daftarkan di <i>Environment Variables</i> (Path).	Masuk ke <i>Edit the system environment variables</i> > Path > Tambahkan alamat folder bin Flutter Anda.
Android license status unknown	Lisensi SDK Android belum disetujui.	Jalankan perintah flutter doctor --android-licenses di terminal dan ketik y untuk semua pilihan.
No connected devices	Driver HP belum terinstal atau <i>USB Debugging</i> belum aktif.	Pastikan kabel data berkualitas baik, cek <i>Developer Options</i> di HP, dan aktifkan <i>USB Debugging</i> .
The method 'setState' isn't defined...	Mencoba memanggil setState di dalam kelas yang <i>StatelessWidget</i> .	Pastikan kelas View menggunakan <i>StatefulWidget</i> . Logic UI harus berada di dalam kelas <i>_NameState</i> .
Underscore variables (_) error	Mencoba mengakses variabel private dari luar berkas (file).	Variabel bertanda _ hanya bisa diakses di file yang sama. Gunakan getter jika ingin membaca nilainya dari file lain.
Gradle task assembleDebug failed	Koneksi internet tidak stabil saat pertama kali <i>build</i> atau kekurangan ruang penyimpanan.	Pastikan internet lancar (untuk unduh dependensi pertama kali) dan ruang disk mencukupi (min. 5GB bebas).

Tugas Praktikum

Untuk menguji sejauh mana pemahaman kalian tentang SRP dan Dart Dasar, selesaikan dua tantangan berikut. Ingat aturan mainnya: 70% harus selesai di Lab (hingga fungsi berjalan), dan 30% sisanya (perapian UI/UX) bisa kalian jadikan homework untuk mempercantik portofolio kalian.

Task 1: The Multi-Step Counter (Low Order Thinking - LOTS)

Fokus: Implementasi logika dasar dan manipulasi variabel.

Pada tugas ini, kalian diminta untuk memodifikasi CounterController agar tidak hanya bertambah satu demi satu, tapi memiliki "Langkah" (Step).

Spesifikasi Tugas:

1. Controller: Tambahkan variabel baru `_step` (default: 1). Buat fungsi untuk mengubah nilai `_step` tersebut.
2. Logic: Fungsi increment dan decrement harus menggunakan nilai `_step` tersebut (Contoh: jika `step = 5`, maka sekali klik "+" nilai bertambah 5).
3. View: Tambahkan sebuah TextField atau Slider di halaman CounterView untuk menentukan besarnya nilai step.

Kriteria Selesai di Lab (70%):

- [x] Nilai counter bertambah/berkurang sesuai angka yang diinput di step.
- [x] Logika perhitungan tetap berada di file CounterController.

Isi x warna hijau pada kurung siku jika berhasil.

Task 2: The History Logger (High Order Thinking - HOTS)

Fokus: Analisis struktur data dan manajemen state.

Aplikasi LogBook kita nantinya akan menyimpan banyak aktivitas. Sebagai pemanasan, mari kita buat fitur "Riwayat" sederhana pada aplikasi Counter ini.

Spesifikasi Tugas:

1. Analysis: Jika kita ingin mencatat setiap perubahan nilai (misal: "+1", "-5", "Reset"), tipe data apa yang paling cocok digunakan di dalam CounterController? (Petunjuk: Gunakan `List<String>`).
2. Controller: Buat sebuah List private untuk menampung riwayat aktivitas. Setiap kali fungsi increment, decrement, atau reset dipanggil, otomatis tambahkan catatan ke dalam List tersebut (Contoh: "User menambah nilai sebesar 1 pada jam 10:00").
3. View: Tampilkan daftar riwayat tersebut di bawah angka Counter menggunakan widget ListView atau Column.
4. The Twist: Tambahkan logika agar riwayat hanya menampilkan 5 aktivitas terakhir saja (Lakukan manipulasi List).

Kriteria Selesai di Lab (70%):

- [x] Setiap tombol diklik, daftar riwayat di layar bertambah secara real-time.

- [x] Mahasiswa mampu menjelaskan mengapa menggunakan List dan bagaimana cara membatasinya menjadi 5 data.

Homework (30%)

Setelah logika di atas berjalan lancar di lab, gunakan waktu di rumah untuk:

1. UI Polishing: Berikan warna yang berbeda untuk riwayat "Tambah" (hijau) dan "Kurang" (merah).
2. UX Improvement: Tambahkan Snackbar atau Dialog konfirmasi jika user menekan tombol Reset, agar data tidak hilang secara tidak sengaja.
3. Self-Reflection: Tuliskan di file readme.md proyek kalian: "Bagaimana prinsip SRP membantu kalian saat harus menambah fitur History Logger tadi?"

Penilaian

Kuesioner Self-Assessment

Jawablah dengan jujur sesuai dengan apa yang kalian rasakan setelah menyelesaikan Modul 1. Berikan tanda centang (✓) pada kolom angka yang paling sesuai:

Keterangan Skala: 1: Sangat Tidak Setuju (STS) | 2: Tidak Setuju (TS) | 3: Netral (N) | 4: Setuju (S) | 5: Sangat Setuju (SS)

No	Pernyataan Refleksi	1	2	3	4	5
1	Saya mampu menjelaskan alasan teknis mengapa View dan Controller harus dipisahkan (Prinsip SRP).					✓
2	Saya memahami fungsi tanda underscore (_) untuk keamanan data (Encapsulation) di Dart.					✓
3	Saya mampu mengatasi kendala instalasi "Path" secara mandiri tanpa bantuan penuh asisten.				✓	
4	Saya memahami kaitan antara fungsi setState() dengan perubahan tampilan pada layar HP.				✓	
5	Saya merasa percaya diri untuk memodifikasi logika perhitungan di dalam file Controller.				✓	

Apa bagian yang paling menantang atau membingungkan bagi kalian di modul ini?

Untuk bagian teknis saat setup nya menurut saya yang paling menantang adalah bagian ketika mengkoneksikan laptop dengan hp karena saya tidak punya USB untuk transfer file jadi saya menggunakan wireless debugging di mana laptop dan hp saya harus terhubung ke jaringan wifi yang sama.

Bagian paling menantang atau membingungkan pada saat perancangan aplikasinya bagi saya adalah bagian membuat UI karena saya masih belum terlalu terbiasa dengan membuat UI dan juga bagian memanggil logika controller di view juga lumayan menantang.

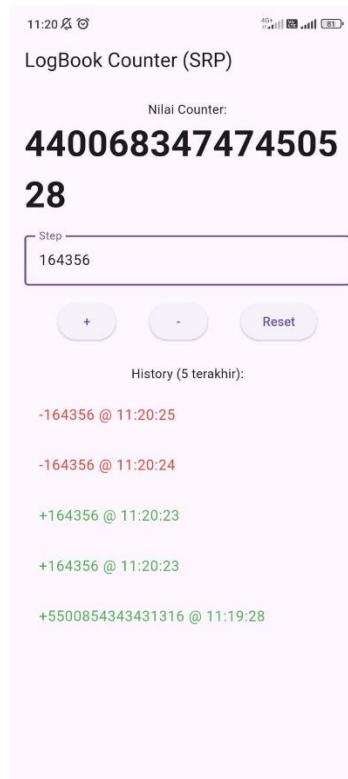
Peer Assessment (Apresiasi Rekan Sejawat 1)

Setelah selesai, tukarkan HP kalian dengan teman sebangku. Cobalah aplikasi mereka dan berikan penilaian sejajar mungkin. Ingat, tujuannya bukan menjatuhkan, tapi saling memperbaiki!

Nama Penilai: Hanifidin Ibrahim **Nama Pemilik Aplikasi:** Dava Ramadhan

No	Kriteria Pengalaman Pengguna (UX)	Skor (1-5)	Catatan "Hal Keren/Saran"
1	Responsivitas: Apakah tombol terasa enak saat ditekan?	5	Terdapat getaran pada saat menekan tombol + - dan reset
2	Kejelasan: Apakah angka counter dan riwayat terbaca dengan jelas?	5	ya terbaca dan menampilkan 5 terakhir
3	Ketahanan: Apakah aplikasi crash saat kamu menekan tombol berkali-kali?	4	Aplikasi tidak crash hanya saja jika menambahkan angka terlalu banyak pada step hanya bertambah 1
4	Kejutan: Apakah fungsi Reset dan Snackbar muncul sesuai ekspektasi?	5	Ya, muncul ketika ingin mereset sebagai konfirmasi

Print Screen Aplikasi Teman Anda:



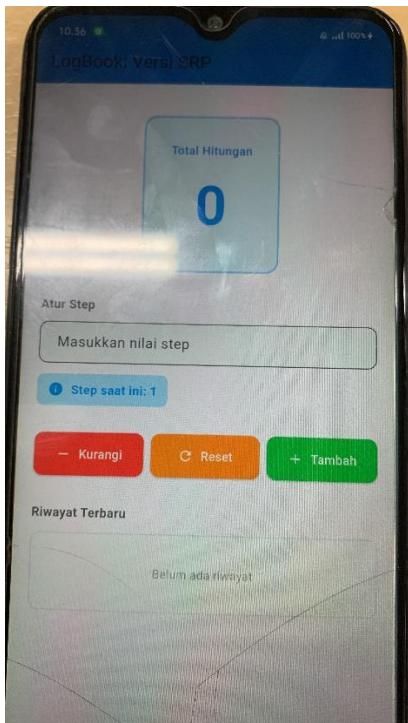
Peer Assessment (Apresiasi Rekan Sejawat 2)

Setelah selesai, tukarkan HP kalian dengan teman sebangku. Cobalah aplikasi mereka dan berikan penilaian sejajar mungkin. Ingat, tujuannya bukan menjatuhkan, tapi saling memperbaiki!

Nama Penilai: Hanifidin Ibrahim **Nama Pemilik Aplikasi:** Fairuz Sheva

No	Kriteria Pengalaman Pengguna (UX)	Skor (1-5)	Catatan "Hal Keren/Saran"
1	Responsivitas: Apakah tombol terasa enak saat ditekan?	5	Tombol berjalan dengan baik tidak ada masalah
2	Kejelasan: Apakah angka counter dan riwayat terbaca dengan jelas?	5	Ya terbaca dengan jelas dan riwayat juga sudah menampilkan warna yang sesuai dengan instruksi
3	Ketahanan: Apakah aplikasi crash saat kamu menekan tombol berkali-kali?	4	Tidak crash hanya saja bisa menambahkan 0 dan angkanya tetap bertambah
4	Kejutan: Apakah fungsi Reset dan Snackbar muncul sesuai ekspektasi?	4	Tombol reset sudah ada hanya saja belum ada konfirmasi ketika melakukan reset

Print Screen Aplikasi Teman Anda:



Lesson Learnt (Refleksi Akhir)

Jangan biarkan ilmu hari ini menguap begitu saja. Tuliskan 3 poin utama yang baru kalian sadari atau pahami hari ini.

1. Saya baru memahami kalau cara agar membuat variable pada class menjadi private di dart itu menggunakan hanya menggunakan _(underscore) saja di depan nama variabelnya dan tidak perlu menambahkan keterangan private di depan variabel seperti di java.
2. Saya baru tahu kalau di Dart ada konsep yang namanya null safety, di mana kita wajib mengisi suatu variabel kalau variabel tersebut dideklarasikan seperti biasa (contoh: int _step). Variabel seperti itu tidak boleh bernilai null, jadi harus langsung diberi nilai atau diisi sebelum digunakan. Kalau variabelnya boleh bernilai null, bisa pakai tanda ? setelah tipe data (contoh: int? _step). Opsi lainnya bisa menggunakan late (contoh: late int _step), tapi dengan catatan variabel tersebut tetap harus diisi sebelum dipakai saat program dijalankan.
3. Target Berikutnya: Pengen tahu cara pindah halaman dan ingin memperbaiki UI/UX aplikasinya

Log LLM: The Fact Check & Twist (Homework)

Menggunakan AI itu boleh, yang tidak boleh adalah menjadi "Zombi AI" (Copy-Paste tanpa mikir). Gunakan template ini setiap kali kalian meminta bantuan ChatGPT/Gemini/Claude.

Komponen	Isian Mahasiswa
Pertanyaan (Prompt)	<i>"Bagaimana cara menggunakan variable bertipe list jika variabelnya tidak ingin langsung diisi nilainya?"</i>
Jawaban AI (Intisari)	AI menyarankan untuk menginisialisasi list tersebut sebagai list kosong (<code>list<String> history = []</code>) atau bisa juga menggunakan late di depan list(<code>late list<String> history</code>).
The Fact Check	Saya coba di DartPad, dan ternyata berhasil dan saya tidak mendapat pesan error untuk mengisi nilai variabel history tersebut.
The Twist (Modifikasi)	Saya pun menambahkan message ke dalam history dan juga membuat logika agar hanya menampilkan 5 message aktivitas terbaru pada getter history sesuai instruksi.

Komponen	Isian Mahasiswa
Pertanyaan (Prompt)	"Saya memiliki class CounterController yang sudah berisi logika counter dan riwayat aktivitas. Bagaimana cara menampilkan nilai counter, mengatur step melalui TextField, serta menampilkan 5 riwayat aktivitas terakhir dengan tampilan berbeda berdasarkan jenis aksinya?"
Jawaban AI (Intisari)	AI menyarankan menggunakan StatefulWidget agar UI bisa diperbarui dengan <code>setState()</code> . Nilai counter ditampilkan menggunakan getter dari controller. Untuk input step, digunakan TextEditingController dan <code>onChanged</code> dengan <code>int.tryParse()</code> agar aman dari error. Riwayat ditampilkan menggunakan Column dan map() dari list history, lalu diberi kondisi untuk membedakan warna dan ikon berdasarkan isi teks.
The Fact Check	Saya menerapkan saran tersebut di project Flutter saya dan berhasil menampilkan nilai counter yang berubah setiap tombol ditekan. Input step juga berjalan dengan baik dan tidak error ketika user memasukkan angka. Riwayat aktivitas dapat ditampilkan dan otomatis update setiap ada aksi.
The Twist (Modifikasi)	Saya menambahkan dialog konfirmasi sebelum reset menggunakan <code>showDialog()</code> agar tidak langsung mereset tanpa persetujuan user. Saya juga menambahkan Snackbar setelah reset berhasil serta memberi warna berbeda pada setiap jenis aktivitas (hijau untuk tambah, merah untuk kurang, oranye untuk reset) agar tampilan lebih informatif dan tidak monoton.

Rubrik Penilaian Dosen Manajer

Tabel 1.5: Rubrik Penilaian Praktikum Modul 1.

Kriteria	Skor 0-50 (Kurang)	Skor 51-80 (Cukup)	Skor 81-100 (Sangat Baik)	Bobot
Prinsip SRP	Logika masih menumpuk di file View atau Main.	Logika sudah dipisah, tapi masih ada variabel UI di Controller.	Pemisahan sempurna antara Logic (Controller) dan UI (View).	30%
Task 1 (Step)	Fungsi tidak jalan atau step tidak dinamis.	Step jalan tapi logika perhitungan ditulis di View.	Step dinamis dan perhitungan dilakukan sepenuhnya di Controller.	20%
Task 2 (History)	Riwayat tidak muncul atau list tidak terupdate.	Riwayat muncul tapi tidak ada batasan (limit) 5 data.	Riwayat muncul rapi, real-time, dan limit 5 data bekerja.	25%
Task 3 (Reset/UX)	Reset langsung terjadi tanpa konfirmasi atau snackbar.	Ada dialog konfirmasi tapi Snackbar tidak muncul.	Alur UX sempurna: Klik -> Dialog -> Reset -> Snackbar.	25%

📋 Rubrik Penilaian Log LLM (Integritas)

Tabel 1.6: Rubrik Penilaian Integritas Log LLM.

Kriteria	Skor 1	Skor 3	Skor 5
Kejujuran	Mengaku tidak pakai AI padahal kode sangat kompleks.	Mencantumkan prompt tapi jawaban AI disalin bulat-bulat.	Terbuka soal bantuan AI dan mencantumkan prompt dengan jelas.
Analisa (Fact Check)	Tidak ada pengecekan ulang.	Mengecek apakah kode jalan atau tidak saja.	Mampu menemukan celah atau kelemahan dari saran AI.
Kreativitas (Twist)	Kode 100% sama dengan saran AI.	Mengubah nama variabel saja.	Mengintegrasikan saran AI ke dalam struktur SRP yang sudah dibuat.

Referensi