

## PANDUAN TUGAS 6

### Membuat Tampilan VideosPlayer

#### Capaian

1. Mahasiswa mampu mengimplementasikan interaksi pada video seperti memutar, menjeda, dan menghentikan video dengan memanfaatkan package `video_player`.
2. Mahasiswa dapat mengimplementasikan interaksi pada video baik untuk video yang bersumber dari asset lokal maupun dari internet.
3. Mahasiswa dapat menampilkan Text widget dengan style yang memanfaatkan package `google_fonts`.
4. Mahasiswa dapat mengimplementasikan animasi sederhana menggunakan `Animated Container Widget`.
5. Mahasiswa mampu menggunakan Image widget untuk menampilkan asset gambar serta memanfaatkan package `cached_network_image` untuk menampilkan gambar dari internet.

#### Spesifikasi Hardware dan Software

Memiliki komponen perangkat keras dan perangkat lunak yang benar sangat penting untuk memastikan keberhasilan pelaksanaan tugas yang diuraikan dalam panduan ini. Konfigurasi perangkat keras dan perangkat lunak yang diperlukan untuk menyelesaikan tugas panduan ini adalah sebagai berikut:

##### A. Spesifikasi Minimum Hardware

1. Minimum RAM 4 GB, disarankan RAM 8 GB
2. Minimum 15 GB ruang disk yang tersedia (2 GB untuk Flutter SDK, 8 GB untuk Android Studio, 4 GB untuk AVD, dan 1 GB untuk proyek)
3. Resolusi layar minimum 1280 x 800
4. Emulator Mobile (Android/IOS)

##### B. Software

1. Flutter SDK (versi 3.13.0 atau yang lebih baru) dan Dart (versi 3.1.5 atau yang lebih baru)
2. Android Studio
3. Visual Studio Code
4. Git dan Github

#### Sumber Daya

1. [Test File Tugas 4A](#)
2. [Test File Tugas 4B](#)
3. [Repository submission\\_media\\_player](#)

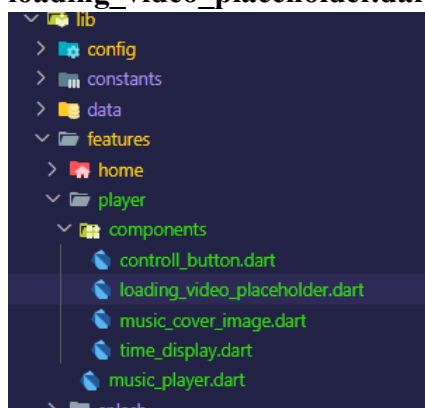
#### Deskripsi Tugas

Mahasiswa akan menuliskan kode pada proyek Flutter `media_player` untuk mengimplementasikan komponen UI dasar multimedia. Mahasiswa perlu memahami cara

pembuatan komponen yang dapat menampilkan gambar baik dari asset lokal maupun internet, teks, icon, serta berbagai dekorasi yang dapat ditambahkan. Dalam panduan ini mahasiswa akan membuat komponen `LoadingVideoPlaceholder`, `VideoIndicator`, `VideoInformation`, dan halaman `VideosPlayer`. Selain itu, mahasiswa juga akan mengimplementasikan interaksi multimedia pada video seperti memutar, menjeda, dan menghentikan video baik untuk video dari asset lokal maupun dari internet.

## Langkah Praktikum 1 (Membuat Komponen `LoadingVideoPlaceholder` dan `VideoIndicator`)

1. Buka proyek flutter `media_player` pada Visual Studio Code anda.
2. Di dalam folder `lib>features>player>components`, buat file baru bernama `loading_video_placeholder.dart`.



3. Berikut ini adalah gambar dari komponen yang akan anda buat



4. Buat sebuah stateless widget dengan nama `LoadingVideoPlaceholder` di dalam file `loading_video_placeholder.dart`.

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

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

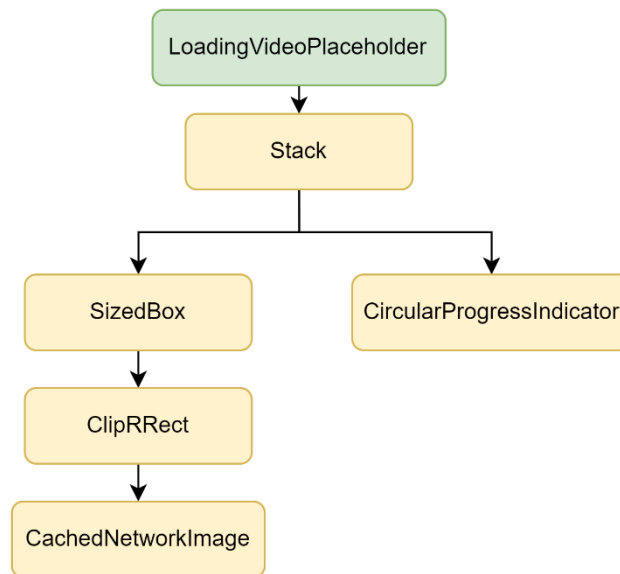
  @override
  Widget build(BuildContext context) {
    return const Placeholder();
  }
}
```

5. Tambahkan properti `sourceType` dan `cover` dengan tipe `String` ke dalam kelas `LoadingVideoPlaceholder`. Jangan lupa untuk menambahkannya juga ke dalam konstruktor.

```
const LoadingVideoPlaceholder({
  super.key,
  required this.sourceType,
  required this.cover,
});

final String sourceType;
final String cover;
```

6. Ubah widget kembalian dari `LoadingVideoPlaceholder` menjadi seperti berikut.



```
Widget build(BuildContext context) {
  return Stack(
    children: [
      SizedBox(
        child: ClipRRect(
          child: CachedNetworkImage(),
        ), // ClipRRect
      ), // SizedBox
      CircularProgressIndicator(),
    ],
  ); // Stack
}
```

7. Tambahkan properti widget dalam `LoadingVideoPlaceholder` tersebut sesuai dengan spesifikasi berikut.

Widget	Properti	Nilai
Stack	alignment	Alignment.center
SizedBox	width	double.infinity
	height	MediaQuery.sizeOf(context).width * (9 / 16)
CachedNetworkImage	imageUrl	cover
	fit	BoxFit.cover
CircularProgressIndicator	color	MainColor.purple5A579C

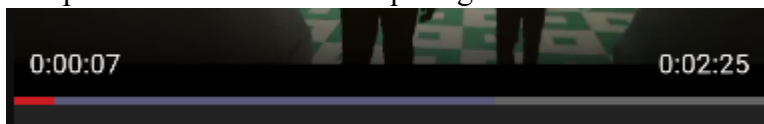
8. Agar komponen LoadingVideoPlaceholder juga dapat menampilkan gambar yang bersumber dari asset lokal, ubah child dari SizedBox menjadi seperti berikut ini.

```

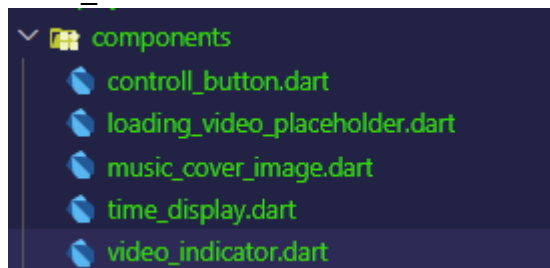
SizedBox(
  width: double.infinity,
  height: MediaQuery.sizeOf(context).width * (9 / 16),
  child: sourceType == "local"
    ? Image.asset(
      cover,
      fit: BoxFit.cover,
    ) // Image.asset
    : ClipRRect(
      child: CachedNetworkImage(
        imageUrl: cover,
        fit: BoxFit.cover,
      ), // CachedNetworkImage
    ), // ClipRRect
), // SizedBox

```

9. Setelah selesai membuat komponen LoadingVideoPlaceholder. Anda akan membuat komponen **VideoIndicator** seperti gambar di bawah ini.



10. Di dalam folder lib>features>player>components, buat file baru bernama **video\_indicator.dart**.



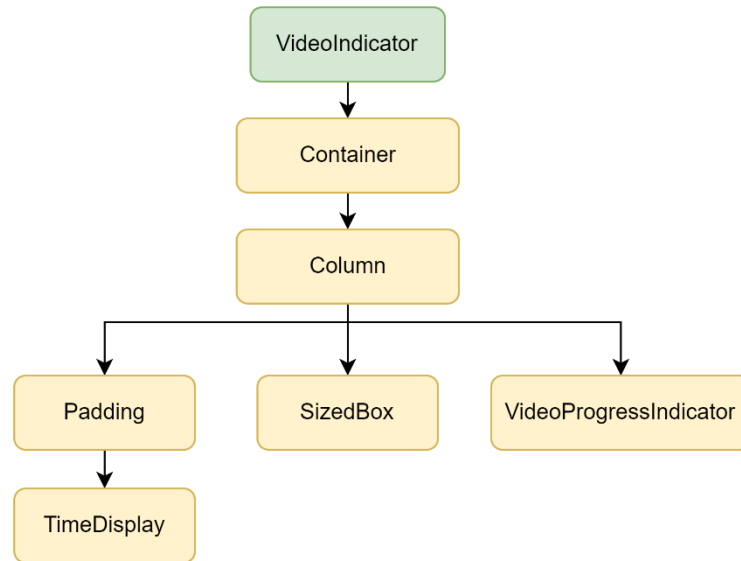
11. Buat sebuah stateless widget dengan nama **VideoIndicator**.

12. Tambahkan properti di bawah ini. Jangan lupa untuk menambahkannya juga ke dalam konstruktor.

```
final bool isVisible;
final Duration position;
final Duration duration;
final VideoPlayerController controller;
```

\*jangan lupa juga untuk melakukan import package video\_player dengan menambahkan kode `import 'package:video_player/video_player.dart';`

13. Ubah widget kembalian dari VideoIndicator menjadi seperti berikut.



```
@override
Widget build(BuildContext context) {
  return Container(
    child: Column(
      children: [
        Padding(
          child: TimeDisplay(),
        ), // Padding
        SizedBox(),
        VideoProgressIndicator(),
      ],
    ), // Column
  ); // Container
}
```

14. Tambahkan properti setiap widget yang ada dalam VideoIndicator sesuai dengan spesifikasi berikut.

Widget	Properti	Nilai
Container	padding	const EdgeInsets.only(top: 20)
	decoration	BoxDecoration(           gradient: LinearGradient(             begin: Alignment.bottomCenter,

		end: Alignment.topCenter, colors: [ MainColor.black000000, MainColor.black000000.withOpacity(0.5), MainColor.black000000.withOpacity(0.2), Colors.transparent, ], ) )
Column	mainAxisSize	MainAxisSize.min
Padding	padding	const EdgeInsets.symmetric(horizontal: 8)
TimeDisplay	position	position
	duration	duration
SizedBox	height	2
VideoProgressIndicator	controller	controller
	allowScrubbing	isVisible

## Langkah Praktikum 2 (Membuat Komponen VideoInformation)

1. Pada praktikum 2 ini, anda akan membuat sebuah komponen VideoInformation yang berisi teks deskripsi video. Pada komponen ini, akan ada handle untuk menampilkan keseluruhan deskripsi atau menyembunyikan sebagian deskripsi. Handle ini mirip seperti tombol show more dan hide pada deskripsi video yang ada di YouTube.
2. Pertama anda harus membuat file baru dengan nama **video\_information.dart** di dalam folder lib>features>player>components.
3. Buat sebuah stateful widget dengan nama **VideoInformation**.
4. Tambahkan properti video dengan tipe Video pada kelas VideoInformation. Jangan lupa untuk menambahnya juga ke dalam konstruktor.

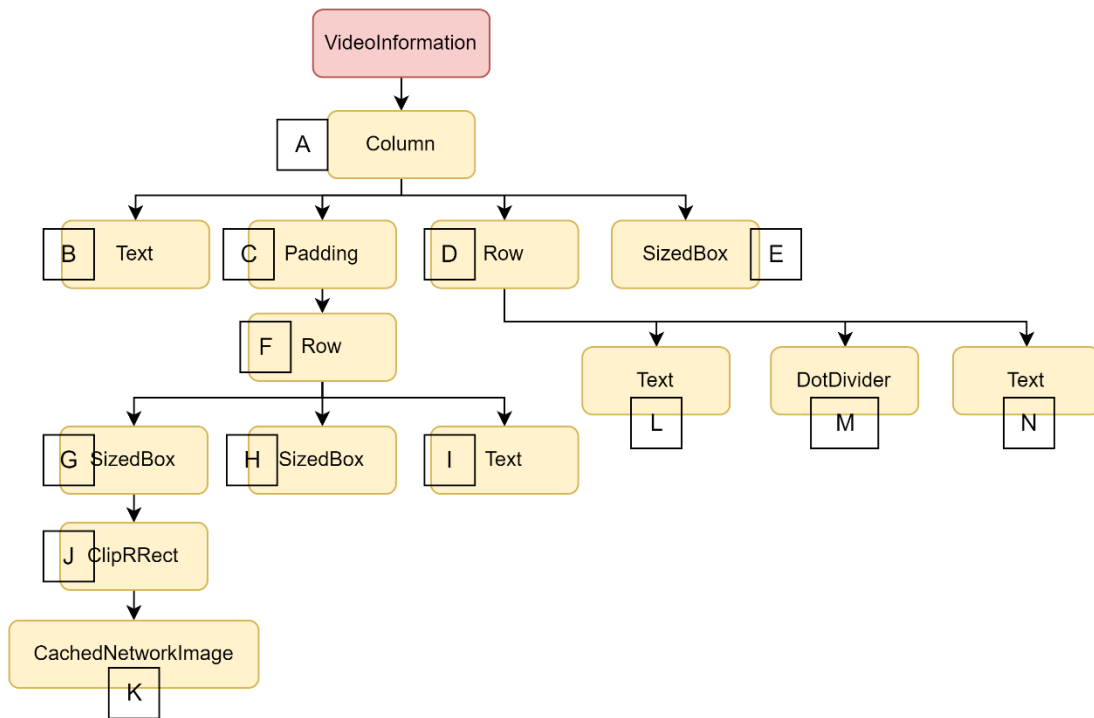
```
import 'package:flutter/material.dart';
import 'package:media_player/data/video_model.dart';

class VideoInformation extends StatefulWidget {
  const VideoInformation({
    super.key,
    required this.video,
  });

  final Video video;
  @override
  State<VideoInformation> createState() => _VideoInformationState();
}

class _VideoInformationState extends State<VideoInformation> {
  @override
  Widget build(BuildContext context) {
    return const Placeholder();
  }
}
```

5. Ubah widget kembalian dari VideoInformation menjadi seperti berikut



6. Untuk melengkapi kode sebelumnya, anda perlu menambahkan properti setiap widget yang ada pada komponen VideoInformation agar sesuai dengan spesifikasi berikut

[KODE] Widget	Properti	Nilai
[A] Column	crossAxisAlignment	CrossAxisAlignment.start
[B] Text	data	widget.video.title!
	maxLine	3
	overflow	TextOverflow.ellipsis
	style	MainTextStyle.poppinsW600.copyWith ( fontSize: 18, color: MainColor.whiteF2F0EB, )
[C] Padding	padding	const EdgeInsets.symmetric(vertical: 8)
[E] SizedBox	height	12
[G] SizedBox	width	36
	height	36
[H] SizedBox	width	8
[I] Text	data	widget.video.creator!
	style	MainTextStyle.poppinsW500.copyWith ( fontSize: 14, color: MainColor.whiteFFFFFF, )
[J] ClipRRect	borderRadius	BorderRadius.circular(18)

[K] CachedNetworkImage	imageUrl	widget.video.creatorPhoto!
	progressIndicatorBuilder	(context, url, progress) => const Padding( padding: EdgeInsets.all(8.0), child: CircularProgressIndicator( color: MainColor.purple5A579C, ), )
[L] Text	fit	BoxFit.cover
	data	"\${widget.video.viewsCount!.formatViewsCount()} x views"
[N] Text	style	MainTextStyle.poppinsW500.copyWith( fontSize: 13, color: MainColor.whiteFFFFFF, )
	data	widget.video.releaseDate!.toLocaleTime( )
[N] Text	style	MainTextStyle.poppinsW500.copyWith( fontSize: 13, color: MainColor.whiteFFFFFF, )

7. Untuk membuat handle show more / hide, tambahkan state `_canShowMore` dengan tipe bool dan initial value-nya adalah true. Selain itu tambahkan fungsi `switchShowMore()` yang digunakan untuk mengubah state `_canShowMore`.

```
bool _canShowMore = true;

switchShowMore() {
  setState(() {
    _canShowMore = !_canShowMore;
  });
}
```

8. Kemudian tambahkan juga fungsi `_hasMoreThanThreeLine(String? deskripsi)`. Fungsi ini berfungsi untuk mengembalikan nilai boolean apakah deskripsi dari video lebih dari 3 baris atau tidak.

```
bool _hasMoreThanThreeLine(String? deskripsi) {
  final span = TextSpan(
    text: deskripsi,
    style: MainTextStyle.poppinsW400.copyWith(
      fontSize: 12,
    ),
  );
}
```



```

final tp = TextPainter(
  text: span,
  maxLines: 3,
  textDirection: TextDirection.ltr,
)..layout(maxWidth: MediaQuery.of(context).size.width - 16);
return tp.didExceedMaxLines;
}

```

- Setelah anda berhasil mempersiapkan state dan fungsi yang dibutuhkan anda harus menambahkan kode berikut ini tepat di bawah widget SizedBox [E].

```

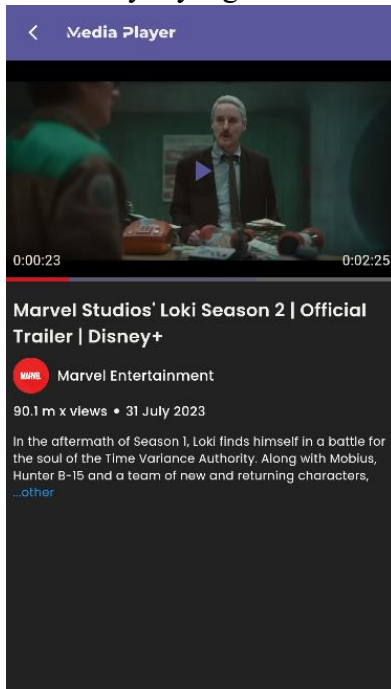
if (_hasMoreThanThreeLine(widget.video.description) &&
    _canShowMore) ...[
  Text(
    widget.video.description!,
    maxLines: 3,
    style: MainTextStyle.poppinsW400.copyWith(
      fontSize: 12,
      color: MainColor.whiteFFFFFF,
    ),
  ),
  InkWell(
    onTap: switchShowMore,
    child: Text(
      '...other',
      style: MainTextStyle.poppinsW400.copyWith(
        fontSize: 12,
        color: Colors.blue,
      ),
    ),
  ),
] else ...[
  Text(
    widget.video.description!,
    style: MainTextStyle.poppinsW400.copyWith(
      fontSize: 12,
      color: MainColor.whiteFFFFFF,
    ),
  ),
  const SizedBox(height: 12),
  if (_hasMoreThanThreeLine(widget.video.description))
    InkWell(
      onTap: switchShowMore,
      child: Text(
        'Less',
        style: MainTextStyle.poppinsW400.copyWith(
          fontSize: 12,
          color: Colors.blue,
        ),
      ),
    ),
]

```

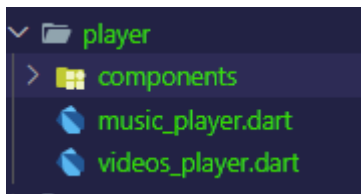
\*Kode ini akan mengevaluasi widget seperti apa yang harus ditampilkan berdasarkan deskripsi video (apakah lebih dari 3 baris atau tidak) dan state `_canShowMore`.

### Langkah Praktikum 3 (Membuat Halaman VideosPlayer)

1. Pada pratikum 3 ini, anda akan membuat halaman VideosPlayer yang tersusun dari komponen-komponen yang telah anda buat pada praktikum sebelumnya seperti CustomAppBar dengan BackButtonAppBarLeading, LoadingVideoPlaceholder, VideoIndicator, VideoInformation, dan ControllButton. Berikut adalah tampilan MediaPlayer yang akan anda buat.



2. Buat file baru di dalam folder lib>features>player dengan nama **videos\_player.dart**.



3. Pada file tersebut, buat sebuah stateful widget dengan nama **VideosPlayer**.

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

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

  @override
  State<VideosPlayer> createState() => _VideosPlayerState();
}

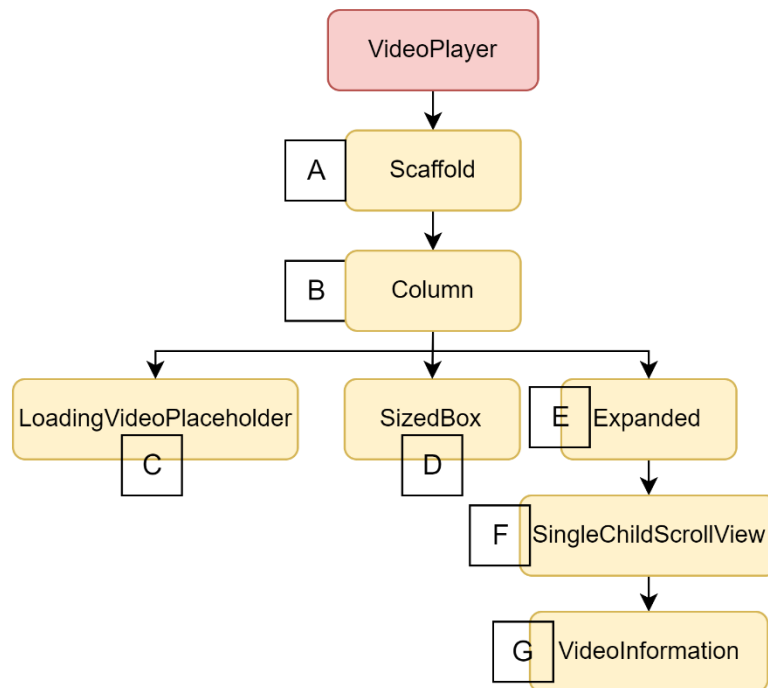
class _VideosPlayerState extends State<VideosPlayer> {
  @override
  Widget build(BuildContext context) {
    return const Placeholder();
  }
}
```

4. Sama seperti halaman MusicPlayer, tambahkan state dan fungsi tambahan untuk membuat data menjadi dinamis dengan mengambil objek Video yang akan dikirimkan argumen oleh Navigator. Untuk itu, tambahkan state `late Video video` dan fungsi `didChangeDependencies()` berikut ini pada kelas VideosPlayerState.

```
late Video video;

@override
void didChangeDependencies() {
  super.didChangeDependencies();
  video = ModalRoute.of(context)!.settings.arguments as Video;
}
```

5. Kemudian ubah widget kembalian dari VideosPlayer menjadi seperti berikut.



```
Widget build(BuildContext context) {
  return Scaffold(
    body: Column(
      children: [
        LoadingVideoPlaceholder(),
        SizedBox(),
        Expanded(
          child: SingleChildScrollView(
            child: VideoInformation(),
          ),
        ),
      ],
    ),
  );
}
```

6. Untuk melengkapi kode sebelumnya, anda perlu menambahkan properti setiap widget yang ada pada halaman VideosPlayer agar sesuai dengan spesifikasi berikut.

[KODE] Widget	Properti	Nilai
[A] Scaffold	backgroundColor	MainColor.black222222
	appBar	const CustomAppBar(leading: BackButtonAppBarLeading())
[B] Column	key	const Key('root_widget')
	crossAxisAlignment	CrossAxisAlignment.start
[C] LoadingVideoPlaceholder	sourceType	video.sourceType!
	cover	video.coverPath!
[E] SizedBox	height	4
[F] SingleChildScrollView	padding	const EdgeInsets.symmetric( vertical: 12, horizontal: 8, )
[G] VideoInformation	video	video

- Untuk membuat tampilan VideosPlayer muncul ketika anda menekan komponen CoverVideoCard yang ada di halaman Home, anda perlu menambahkan kode berikut sebagai nilai pada properti onTap GestureDetector yang ada di CoverVideoCard.

```
( ) {
  Navigator.pushNamed(
    context,
    MainRoute.videoPlayer,
    arguments: video,
  );
}
```

- Selain itu, agar halaman VideoPlayer yang anda buat dapat ditampilkan melalui route MainRoute.videoPlayer, buka file **main\_pages.dart** dan ubah widget kembalian dari widget function MainRoute.videoPlayer dari yang sebelumnya **Placeholder()** menjadi **VideosPlayer()**.
- Sekarang coba jalankan aplikasi anda. Seharusnya aplikasi akan terlihat [seperti ini](#).

### Langkah Praktikum 3 (Menambahkan Interaksi pada Halaman VideosPlayer)

- Sekarang anda akan belajar cara menambahkan interaksi multimedia pada video seperti memutar, menjeda, dan menghentikan video. Untuk melakukannya, anda perlu membuat sebuah objek VideoPlayerController dari package *video\_player* di dalam kelas VideoPlayerState.

```
Late VideoPlayerController controller;
```

- Selain itu, tambahkan juga state Duration animDuration, Duration duration, Duration position, bool isVisible, dan Future<void> initializeVideoPlayerFuture. Semua state tersebut akan digunakan untuk menyimpan nilai dari Video yang akan dimainkan dan perubahannya.

```
final Duration animDuration = const Duration(milliseconds: 300);
Duration duration = const Duration();
```

```

Duration position = const Duration();
late Future<void> initializeVideoPlayerFuture;
bool isVisible = true;

```

- Setelah itu, tambahkan fungsi `initVideoController()` untuk menginisiasi object controller dan state-state lainnya berdasarkan objek video yang akan dimainkan.

```

void initVideoController() {
    video.sourceType == "local"
        ? controller = VideoPlayerController.asset(video.source!)
        : controller =
            VideoPlayerController.networkUrl(Uri.parse(video.source!));
    initializeVideoPlayerFuture = controller.initialize().then((value) {
        setState(() {
            duration = controller.value.duration;
        });
    });
    controller.setLooping(true);
    controller.setVolume(1.0);

    controller.addListener(
        () => setState(
            () => position = controller.value.position,
        ),
    );
}

```

- Agar fungsi tersebut dijalankan tepat saat objek video berhasil diterima, tambahkan pemanggilan `initVideoController()`, pada body fungsi `didChangeDependencies()`.

```

class _VideosPlayerState extends State<VideosPlayer> {
    late Video video;

    late VideoPlayerController controller;
    final Duration animDuration = const Duration(milliseconds: 300);
    Duration duration = const Duration();
    Duration position = const Duration();
    late Future<void> initializeVideoPlayerFuture;
    bool isVisible = true;

    @override
    void didChangeDependencies() {
        super.didChangeDependencies();
        video = ModalRoute.of(context)!.settings.arguments as Video;
        initVideoController();
    }

    void initVideoController() {
        video.sourceType == "local"
            ? controller = VideoPlayerController.asset(video.source!)
            : controller =
                VideoPlayerController.networkUrl(Uri.parse(video.source!));
        initializeVideoPlayerFuture = controller.initialize().then((value) {
            setState(() {
                duration = controller.value.duration;
            });
        });
        controller.setLooping(true);
        controller.setVolume(1.0);

        controller.addListener(
            () => setState(
                () => position = controller.value.position,
            ),
        );
    }
}

```

- Setelah menambahkan kode untuk menginisiasi `VideoPlayerController`, sekarang anda akan menuliskan kode untuk mengatur state `isVisible` yaitu fungsi `offVisible` dan `onVisible`. `offVisible()` digunakan untuk mengubah nilai `isVisible` menjadi false sehingga nantinya komponen kontrol video (`ControllButton` dan `VideoIndicator`) dapat disembunyikan (`alpha = 0`), begitupun sebaliknya untuk `onVisible`.

```
offVisible() {
  setState(() {
    isVisible = false;
  });
}

onVisible() {
  setState(() {
    isVisible = true;
  });
}
```

- Tambahkan fungsi `switchControllVisibility()` yang digunakan untuk mengontrol visibilitas kontrol video. Jika saat ini tersembunyi (`!isVisible`), ia akan menampilkannya (`onVisible()`) dan menyetel pengatur waktu menggunakan `Debouncer` (fungsi khusus yang menunda eksekusi) untuk menyembunyikannya lagi setelah 2,5 detik.

```
void switchControllVisibility() {
  if (!isVisible) {
    onVisible();
    Debouncer(milliseconds: 2500).run(() {
      if (isVisible && controller.value.isPlaying == true) {
        offVisible();
      }
    });
  }
}
```

- Tambahkan lagi fungsi `playPause()` di bawah ini untuk mengatur interaksi memutar atau menjeda video.

```
Future<void> playPause() async {
  if (isVisible) {
    if (controller.value.isPlaying) {
      await controller.pause();
    } else {
      await controller.play();
      offVisible();
    }
  }
}
```

- Sedangkan untuk melakukan interaksi menghentikan video (menghapus video dari controller), tambahkan pemanggilan `controller.dispose()` pada fungsi override `dispose`. Sehingga ketika user menekan tombol kembali (men-trigger penghapusan halaman `VideosPlayer` dari `Stack Navigation`), baik dari `App Bar` maupun dari tombol fisik, video akan otomatis dihentikan.

```

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

```

9. Kini anda telah selesai menyiapkan semua fungsi agar halaman VideosPlayer dapat melakukan interaksi-interaksi pada video. Sekarang anda perlu memperbarui widget kembalian dari kelas VideosPlayerState, agar bisa menampilkan video dengan kontrol video yang dapat memudar dan muncul (memanfaatkan AnimatedOpacity). Untuk melakukannya, bungkus widget LoadingVideoPlaceholder dengan FutureBuilder seperti ini.

```

body: Column(
  key: const Key('root_widget'),
  crossAxisAlignment: CrossAxisAlignment.start,
  children: [
    FutureBuilder(
      future: initializeVideoPlayerFuture,
      builder: (context, snapshot) {
        return LoadingVideoPlaceholder(
          sourceType: video.sourceType!,
          cover: video.coverPath!,
        ); // LoadingVideoPlaceholder
      }
    ), // FutureBuilder
    const SizedBox(
      height: 4,
    ), // SizedBox
    Expanded

```

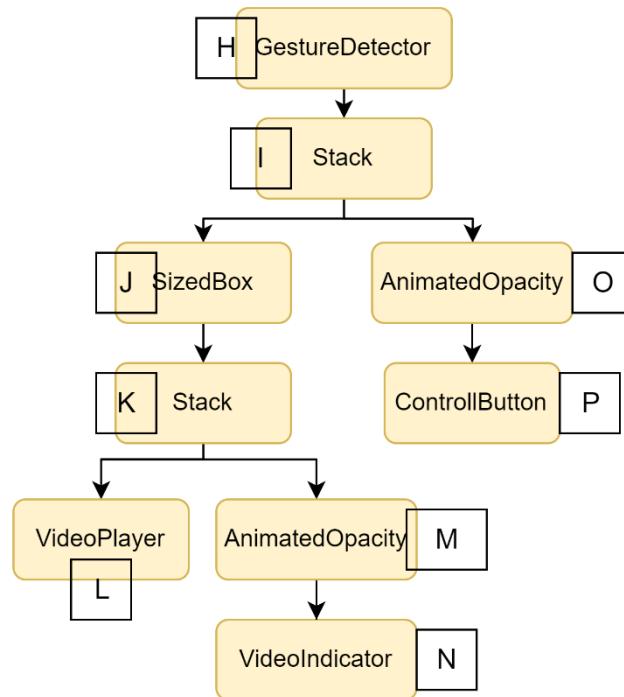
10. Pada body builder-nya, tambahkan if else seperti di bawah ini agar widget LoadingVideoPlaceholder ditampilkan ketika video memang belum selesai dimuat (`snapshot.connectionState != ConnectionState.done`).

```

FutureBuilder(
  future: initializeVideoPlayerFuture,
  builder: (context, snapshot) {
    if (snapshot.connectionState == ConnectionState.done) {
      return const Placeholder();
    } else {
      return LoadingVideoPlaceholder(
        sourceType: video.sourceType!,
        cover: video.coverPath!,
      );
    }
  },
),

```

11. Ubah widget Placeholder tersebut menjadi seperti berikut.



```

return GestureDetector(
  child: Stack(
    children: [
      SizedBox(
        child: Stack(
          children: [
            VideoPlayer(),
            AnimatedOpacity(
              child: VideoIndicator(),
            ),
          ],
        ),
      ),
      AnimatedOpacity(
        child: ControllButton(),
      ),
    ],
  ),
);
  
```

12. Untuk melengkapi kode diatas, anda perlu menambahkan properti widget-nya agar sesuai dengan spesifikasi berikut.

[KODE] Widget	Properti	Nilai
[I] Stack	alignment	Alignment.center
[J] SizedBox	width	double.infinity
	height	MediaQuery.sizeOf(context).width * 9 / 16
[K] Stack	alignment	Alignment.bottomLeft
[L] VideoPlayer	controller	controller
[M] AnimatedOpacity	duration	animDuration
	opacity	isVisible ? 1 : 0
[N] VideoIndicator	position	position



	duration	duration
	controller	controller
	isVisible	isVisible
[O] AnimatedOpacity	duration	animDuration
	opacity	isVisible ? 1 : 0
[P] ControllButton	icon	controller.value.isPlaying ? Icons.pause : Icons.play_arrow
	onPressed	playPause
	bgColor	MainColor.black000000.withOpacity(0.2)
	splashR	26
	icSize	36

13. Coba jalankan proyek anda pada emulator dengan menekan tombol run yang ada pada fungsi main di **main.dart**. Berikut adalah tampilan emulator jika anda telah berhasil menyusun tampilan VideosPlayer dan mengimplementasikan interaksi video. Lihat tampilan keseluruhan aplikasinya [di sini](#).

### Langkah Verifikasi Kode

1. Untuk keperluan testing, ubah kelas VideosPlayerState menjadi public dengan menghapus `_` yang ada pada nama kelasnya. Selain itu tambahkan anotasi `@visibleForTesting` tepat di atas kelas VideosPlayerState. Sehingga kelas VideosPlayerState akan menjadi seperti berikut.

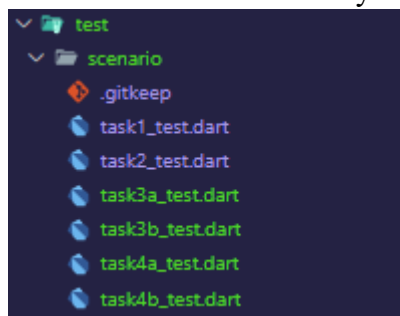
```
class VideosPlayer extends StatefulWidget {
  const VideosPlayer({super.key});

  @override
  State<VideosPlayer> createState() => VideosPlayerState();
}

@visibleForTesting
class VideosPlayerState extends State<VideosPlayer> {
  late Video video;

  late VideoPlayerController controller;
```

2. Unduh test file pada tautan [Test File Tugas 4A](#) dan [Test File Tugas 4B](#). Letakkan file ke dalam folder test>scenario yang ada pada proyek.



3. Buka file `task4a_test.dart`.

```

1 > import 'package:cached_network_image/cached_network_image.dart'; ...
19
20 Video? capturedVideo;
21
22 class MockNavigatorObserver extends Mock
23   implements NavigatorObserver, WidgetsBindingObserver {
24   @override
25   void didPush(Route<dynamic> route, Route<dynamic>? previousRoute) {
26     // print('pushed $route');
27     if (route.settings.arguments is Video) {
28       capturedVideo = route.settings.arguments as Video;
29     }
30   }
31 }
32
33 final routes = <String, WidgetBuilder>{
34   '/video-player': (_) => const VideosPlayer(),
35 };
36
37 void main() {
38   Video localSourceVideo = Video(...
51
52   void checkWhenCanShowMore(...
96
97   void checkWhenCanLess(...
127
128   testWidgets('Structur of LoadingVideoPlaceholder widget is built correctly', ...
193
194   testWidgets('LoadingVideoPlaceholder render local source', ...

```

4. Tekan tombol Run yang ada pada fungsi main dan tunggu proses hingga selesai.

5. Jika anda mengalami error, coba periksa pesan yang muncul dan perbaiki kesalahannya. Kesalahan yang sering terjadi biasanya karena adanya perbedaan nama key atau properti yang hilang (tidak sesuai dengan spesifikasi) dari widget tertentu.

```

✓ VideoInformation widget can handle show more action
═══════════ EXCEPTION CAUGHT BY FLUTTER TEST FRAMEWORK ════════════
The following TestFailure was thrown running a test:
Expected: exactly one matching node in the widget tree
  Actual: _KeyFinder:<zero widgets with key [<'video_section']> (ignoring offstage widgets)>
  Which: means none were found but one was expected
Expected found a widget with "video_section" key

When the exception was thrown, this was the stack:
#4 main.<anonymous closure> (file:///D:/00PENDIDIKAN/POLINEMA/PMB/Skripsi/PROJECT/media_player/test/scenario/task4a_test.dart:868:7)
<asynchronous suspension>
<asynchronous suspension>
(elided one frame from package:stack_trace)

This was caught by the test expectation on the following line:
file:///D:/00PENDIDIKAN/POLINEMA/PMB/Skripsi/PROJECT/media_player/test/scenario/task4a_test.dart line 868
The test description was:
VideosPlayer display ControllButton, VideoIndicator, and VideoInformation

Test failed. See exception logs above.
The test description was: VideosPlayer display ControllButton, VideoIndicator, and VideoInformation

✗ VideosPlayer display ControllButton, VideoIndicator, and VideoInformation

```

6. “Expected found a widget with "video\_section" key”. Maksud dari pesan error tersebut adalah seharusnya dalam struktur widget (widget tree) pada halaman VideosPlayer terdapat widget yang memiliki key "video\_section". Widget yang harus memiliki properti key tersebut adalah widget GestureDetector [H]. Sehingga saat anda mengalami error yang serupa, tambahkan properti **key** pada widget GestureDetector [H] sesuai dengan tabel spesifikasi berikut.

[KODE] Widget	Properti	Nilai
[H] GestureDetector	key	const Key('video_section')

7. Jalankan kembali test file dengan menekan tombol Run yang ada pada fungsi main dan tunggu proses hingga selesai. Jika testing berhasil, maka akan muncul hasil berikut pada tab Debug Console Visual Studio Code anda.

```

PROBLEMS 2 DEBUG CONSOLE OUTPUT TEST RESULTS TERMINAL PORTS ... Filter (e.g. text, lex
✓ Struktur of LoadingVideoPlaceholder widget is built correctly
✓ LoadingVideoPlaceholder render local source
✓ LoadingVideoPlaceholder render network source
✓ Struktur of VideoIndicator widget is built correctly
✓ Struktur of VideoInformation widget is built correctly
✓ VideoInformation widget can handle show more action
✓ VideosPlayer display ControllButton, VideoIndicator, and VideoInformation

Exited.

```

8. Sekarang jalankan test file **task4b\_test.dart**. Jika anda mengalami error, coba periksa pesan yang muncul dan perbaiki kesalahannya.

```

PROBLEMS 2 DEBUG CONSOLE OUTPUT TEST RESULTS TERMINAL PORTS ... Filter (e.g. text, lexclude) Dart (Pixel 4
===== EXCEPTION CAUGHT BY FLUTTER TEST FRAMEWORK =====
The following TestFailure was thrown running a test:
Expected: 'Closure: () => void from Function \'switchControllVisibility\':..'
Actual: 'null'
Which is different.
  Expected: Closure: ( ...
    Actual: null
    ^
Differ at offset 0
GestureDetector widget with key "video_section" onTap callback should be switchControllVisibility

When the exception was thrown, this was the stack:
#4      main.<anonymous closure> (file:///D:/00PENDIDIKAN/POLINEMA/PMB/Skripsi/PROJECT/media_player/test/scenari
o/task4b_test.dart:168:7)
<asynchronous suspension>
<asynchronous suspension>
(elided one frame from package:stack_trace)

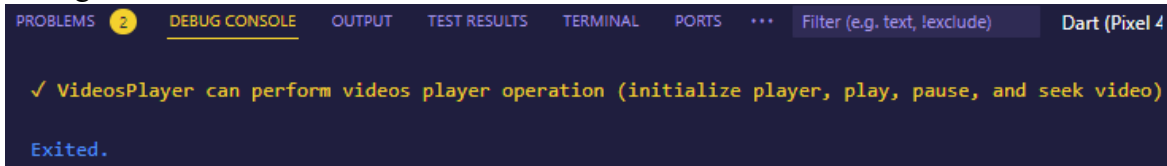
This was caught by the test expectation on the following line:
  file:///D:/00PENDIDIKAN/POLINEMA/PMB/Skripsi/PROJECT/media_player/test/scenario/task4b_test.dart line 168
The test description was:
  VideosPlayer can perform videos player operation (initialize player, play, pause, and seek video)

```

“GestureDetector widget with key "video\_section" onTap callback should be switchControllVisibility”. Untuk memperbaikinya, anda hanya perlu menambahkan properti berikut.

[KODE] Widget	Properti	Nilai
[H] GestureDetector	onTap	switchControllVisibility

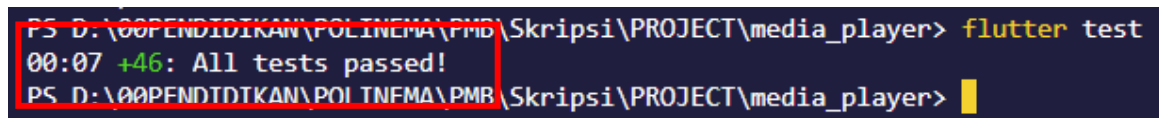
- Jalankan kembali test file dengan menekan tombol Run yang ada pada fungsi main dan tunggu proses hingga selesai. Jika testing berhasil, maka akan muncul hasil berikut pada tab Debug Console Visual Studio Code anda.



## Langkah Pengumpulan Proyek

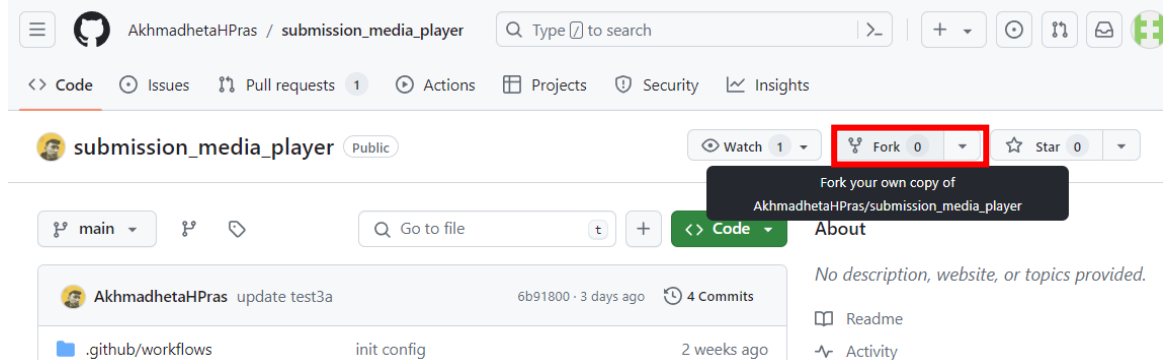
Pengumpulan proyek Flutter Media Player dilakukan melalui metode pull request pada repository github [submission\\_media\\_player](#). **Pastikan untuk mengubah title pull request anda dengan nama lengkap anda dan isi kolom deskripsi pull request sesuai format seperti pada langkah no 14!** Ikuti langkah-langkah berikut:

- Sebelum melakukan pengumpulan proyek flutter Media Player, anda dapat melakukan tinjauan ulang lagi apakah kode yang anda tuliskan dari Panduan Tugas A01 sampai A05 telah sepenuhnya berhasil saat proses verifikasi kode. Untuk melakukannya jalankan perintah `flutter test` pada terminal untuk menjalankan semua test file yang ada pada folder test sekaligus. Jika memang semua telah berhasil maka akan muncul hasil seperti berikut.



\* Jika anda mengalami error, coba periksa pesan yang muncul dan perbaiki kesalahannya.

- Buka [github.com](#) dan masuk menggunakan akun anda.
- Buka repository github [submission\\_media\\_player](#). Kemudian fork repository tersebut.



## Create a new fork

A *fork* is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project.

Required fields are marked with an asterisk (\*).

Owner \*  / Repository name \*

Copy the `main` branch only  
Contribute back to AkhmadhetaHPras/submission\_media\_player by adding your own branch. [Learn more.](#)

You are creating a fork in your personal account.



4. Buka terminal atau command prompt pada direktori lokal yang ingin Anda gunakan untuk menyimpan repository.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS D:\00PENDIDIKAN\POLINEMA\PMB\Skrripsi\projects archive> |
```

5. Jalankan perintah berikut untuk clone repository:

```
git clone https://github.com/<your_username>/submission_media_player.git
```

\*ubah `<your_username>` dengan nama username github anda.

```
PS D:\00PENDIDIKAN\POLINEMA\PMB\Skrripsi\projects archive> git clone https://github.com/Haflipside/submission_media_player.git
Cloning into 'submission_media_player'...
remote: Enumerating objects: 15, done.
remote: Counting objects: 100% (15/15), done.
remote: Compressing objects: 100% (9/9), done.
remote: Total 15 (delta 2), reused 15 (delta 2), pack-reused 0R
Receiving objects: 100% (15/15), 28.60 KiB | 1.30 MiB/s, done.
Resolving deltas: 100% (2/2), done.
```

6. Buka direktori repository yang di-clone.

```
PS D:\00PENDIDIKAN\POLINEMA\PMB\Skrripsi\projects archive> cd .\submission_media_player\
PS D:\00PENDIDIKAN\POLINEMA\PMB\Skrripsi\projects archive\submission_media_player> |
```

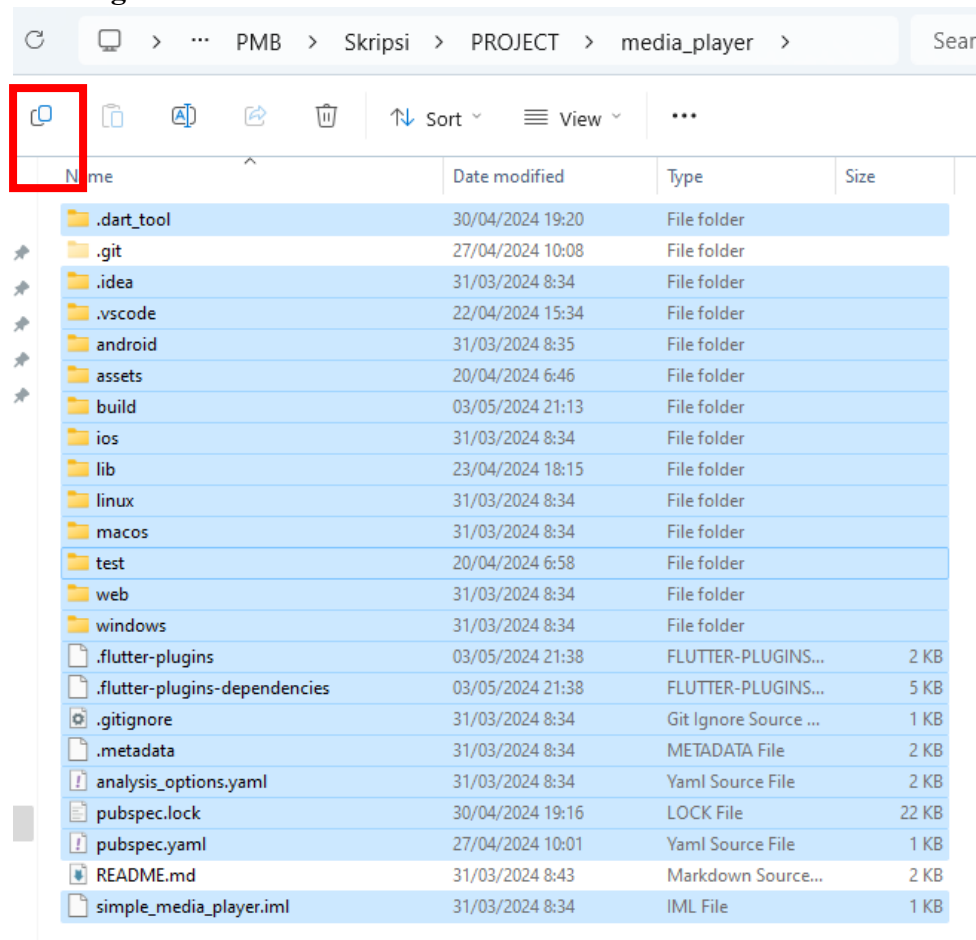
7. Jalankan perintah berikut untuk membuat branch baru:

```
git checkout -b <your_branch_name>
```

\*ubah `<your_branch_name>` dengan nama anda.

```
PS D:\00PENDIDIKAN\POLINEMA\PMB\Skrripsi\projects archive\submission_media_player> git checkout -b akhmadheta_hafid
Switched to a new branch 'akhmadheta_hafid'
PS D:\00PENDIDIKAN\POLINEMA\PMB\Skrripsi\projects archive\submission_media_player> |
```

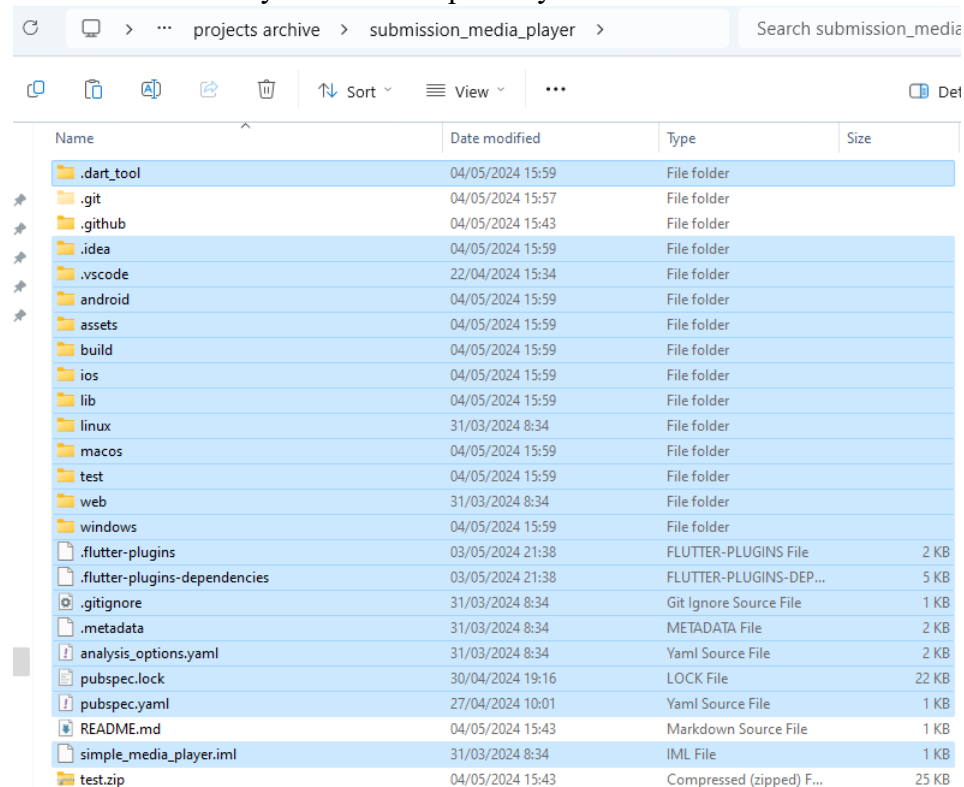
8. Salin semua file dan folder yang ada pada proyek flutter media\_player anda. **Kecualikan folder .git dan file README.md**



The screenshot shows a file explorer window with the path: `media_player`. The toolbar includes a 'Copy' icon (two overlapping sheets of paper) which is highlighted with a red box. Below the toolbar is a table listing the files and folders in the directory.

Name	Date modified	Type	Size
.dart_tool	30/04/2024 19:20	File folder	
.git	27/04/2024 10:08	File folder	
.idea	31/03/2024 8:34	File folder	
.vscode	22/04/2024 15:34	File folder	
android	31/03/2024 8:35	File folder	
assets	20/04/2024 6:46	File folder	
build	03/05/2024 21:13	File folder	
ios	31/03/2024 8:34	File folder	
lib	23/04/2024 18:15	File folder	
linux	31/03/2024 8:34	File folder	
macos	31/03/2024 8:34	File folder	
test	20/04/2024 6:58	File folder	
web	31/03/2024 8:34	File folder	
windows	31/03/2024 8:34	File folder	
.flutter-plugins	03/05/2024 21:38	FLUTTER-PLUGINS...	2 KB
.flutter-plugins-dependencies	03/05/2024 21:38	FLUTTER-PLUGINS...	5 KB
.gitignore	31/03/2024 8:34	Git Ignore Source ...	1 KB
.metadata	31/03/2024 8:34	METADATA File	2 KB
analysis_options.yaml	31/03/2024 8:34	Yaml Source File	2 KB
pubspec.lock	30/04/2024 19:16	LOCK File	22 KB
pubspec.yaml	27/04/2024 10:01	Yaml Source File	1 KB
README.md	31/03/2024 8:43	Markdown Source...	2 KB
simple_media_player.iml	31/03/2024 8:34	IML File	1 KB

9. Masukkan salinannya ke dalam repository hasil clone.



The screenshot shows a file explorer window with the path: `submission_media_player`. The toolbar includes a 'Copy' icon. Below the toolbar is a table listing the files and folders in the directory.

Name	Date modified	Type	Size
.dart_tool	04/05/2024 15:59	File folder	
.git	04/05/2024 15:57	File folder	
.github	04/05/2024 15:43	File folder	
.idea	04/05/2024 15:59	File folder	
.vscode	22/04/2024 15:34	File folder	
android	04/05/2024 15:59	File folder	
assets	04/05/2024 15:59	File folder	
build	04/05/2024 15:59	File folder	
ios	04/05/2024 15:59	File folder	
lib	04/05/2024 15:59	File folder	
linux	31/03/2024 8:34	File folder	
macos	04/05/2024 15:59	File folder	
test	04/05/2024 15:59	File folder	
web	31/03/2024 8:34	File folder	
windows	04/05/2024 15:59	File folder	
.flutter-plugins	03/05/2024 21:38	FLUTTER-PLUGINS File	2 KB
.flutter-plugins-dependencies	03/05/2024 21:38	FLUTTER-PLUGINS-DEP...	5 KB
.gitignore	31/03/2024 8:34	Git Ignore Source File	1 KB
.metadata	31/03/2024 8:34	METADATA File	2 KB
analysis_options.yaml	31/03/2024 8:34	Yaml Source File	2 KB
pubspec.lock	30/04/2024 19:16	LOCK File	22 KB
pubspec.yaml	27/04/2024 10:01	Yaml Source File	1 KB
README.md	04/05/2024 15:43	Markdown Source File	1 KB
simple_media_player.iml	31/03/2024 8:34	IML File	1 KB
test.zip	04/05/2024 15:43	Compressed (zipped) F...	25 KB

10. Jalankan perintah berikut untuk stage perubahan Anda:

```
git add .
```

11. Jalankan perintah berikut untuk commit perubahan Anda:

```
git commit -m "<commit_message>"
```

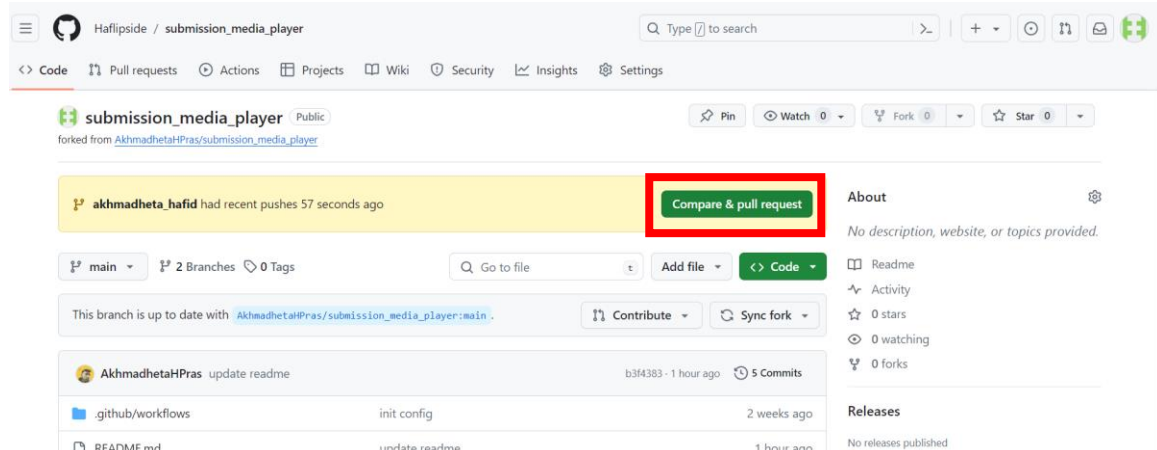
\*ubah <commit\_message> dengan deskripsi yang jelas tentang pengumpulan tugas anda.

```
PS D:\00PENDIDIKAN\POLINEMA\PMB\Skripsi\projects archive\submission_media_player> git commit -m "hafid submission #1"
[akhmadheta_hafid ec15249] hafid submission #1
184 files changed, 11017 insertions(+)
create mode 100644 .gitignore
create mode 100644 .metadata
create mode 100644 .vscode/settings.json
create mode 100644 analysis_options.yaml
create mode 100644 android/.gitignore
create mode 100644 android/app/build.gradle
create mode 100644 android/app/src/debug/AndroidManifest.xml
create mode 100644 android/app/src/main/AndroidManifest.xml
create mode 100644 android/app/src/main/kotlin/com/example/simple_media_player/MainActivity.kt
create mode 100644 android/app/src/main/res/drawable-v21/launch_background.xml
create mode 100644 android/app/src/main/res/drawable/launch_background.xml
create mode 100644 android/app/src/main/res/mipmap-hdpi/ic_launcher.png
create mode 100644 android/app/src/main/res/mipmap-mdpi/ic_launcher.png
```

12. Jalankan perintah berikut untuk push perubahan Anda ke repositori remote:

```
git push origin <your_branch_name>
```

13. Buka halaman repositori fork Anda di GitHub. Kemudian klik tombol "Compare & pull request".



14. Berikan judul dan deskripsi untuk pull request Anda.

- Pastikan anda memasukkan title pull request dengan **Nama Lengkap Anda**
- Gunakan format berikut untuk mengisi kolom deskripsi pull request
  - o **Tutorial Completion Time:** (How long did it take you to complete the tutorials? [e.g., 2 days, 1 week])
  - o **Tutorial Clarity:** (On a scale of 1 (Least Clear) to 5 (Most Clear), how clear and easy-to-follow were the tutorials?)
  - o **Project Difficulty:** (On a scale of 1 (Easiest) to 5 (Hardest), how challenging did you find developing this project?)
  - o **Project Satisfaction:** (On a scale of 1 (Least Satisfied) to 5 (Most Satisfied), how satisfied are you with your final project?)
  - o **Additional Feedback:** (Share any additional thoughts or feedback you have about the tutorials or the project itself)

### Contoh deskripsi pull request:

```
**Tutorial Completion Time:** 4 hours  
**Tutorial Clarity:** 4 (Clear and easy to follow)  
**Project Difficulty:** 3 (Moderately challenging)  
**Project Satisfaction:** 5 (Very satisfied, learned a lot!)  
**Additional Feedback:**
```

The tutorials were very helpful, especially the section on building the user interface. I would love to see more tutorials on advanced features like user authentication.

### Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#). [Learn more about diff comparisons here](#)

base repository: AkhmadhetaHPras/submission... base: main head repository: Hafliptime/submission\_media... compare: akhmadheta\_hafid

✓ Able to merge. These branches can be automatically merged.

Add a title

Akhmadheta Hafid Prasetyawan

Add a description

Write Preview H B I ≡ <> 🔗 ≡ ≡ 🗑️ 📎 @ 🗨️ ↶ 🗑️

```
**Tutorial Completion Time:** 4 hours  
**Tutorial Clarity:** 5 (Very Clear and easy to follow)  
**Project Difficulty:** 3 (Moderately challenging)  
**Project Satisfaction:** 5 (Very satisfied, learned a lot!)  
**Additional Feedback:**  
The tutorials were very helpful 😊
```

### 15. Klik tombol "Create pull request".

Write Preview H B I ≡ <> 🔗 ≡ ≡ 🗑️ 📎 @ 🗨️ ↶ 🗑️

```
**Tutorial Completion Time:** 4 hours  
**Tutorial Clarity:** 5 (Very Clear and easy to follow)  
**Project Difficulty:** 3 (Moderately challenging)  
**Project Satisfaction:** 5 (Very satisfied, learned a lot!)  
**Additional Feedback:**  
The tutorials were very helpful, 😊 🙌
```

Markdown is supported Paste, drop, or click to add files

Allow edits and access to secrets by maintainers

Create pull request

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).