TUGAS PENDAHULUAN
PEMROGRAMAN PERANGKAT BERGERAK
MODUL IX
Api Perangkat Keras

Disusun Oleh :

Arwin Nabiel Arrofif

2211104057

SE 06 2

Asisten Praktikum :

Ayu susiloawati
Noviana Rizki Anisa putri

Dosen Pengampu :

Yudha Islami Sulistya, S.Kom., M.Cs.

PROGRAM STUDI S1 SOFTWARE ENGINEERING FAKULTAS

INFORMATIKA TELKOM UNIVERSITY PURWOKERTO 2024

API PERANGKAT KERAS

Tujuan Praktikum
Mahasiswa mampu memahami konsep layout pada Flutter Mahasiswa dapat mengimplementasikan desain user interface pada Flutter

1. Camera API

Camera API berfungsi untuk memungkinkan *developer* (pengembang) untuk mengakses dan mengontrol kamera perangkat. Flutter menyediakan paket camera yang memudahkan implementasi fitur kamera untuk mengambil foto, merekam video, dan mengakses umpan kamera secara langsung. Paket ini sangat berguna untuk membuat aplikasi yang membutuhkan pengambilan gambar atau video, seperti aplikasi media sosial atau e-commerce.

Pada bahasan kali ini, kita akan menggunakan packages atau plugin Camera supaya aplikasi yang dibuat dapat mengakses kamera yang ada pada device.

Cara instalasi:

a. Tambahkan paket *camera* yang ada pada Pub Dev di *pubspec.yaml*
b. Lalu jalankan perintah '*flutter pub get*'
c. Izinkan akses kamera pada *AndroidManifest.xml*

```
<uses-permission android:name="android.permission.CAMERA" />
<uses-feature android:name="android.hardware.camera" />
```

- Pada iOS, plugin ini dapat berjalan di versi iOS 10.0 atau lebih tinggi. Jika dijalankan di versi di bawah 10.0, pastikan bahwa ada program untuk cek versi dari iOS sebelum menggunakan fitur yang ada pada kamera. Plugin untuk cek versi iOS yaitu device_info_plus.

Tambahkan 2 baris pada ios/Runner/Info.plist:

- Satu baris dengan Privacy - Camera Usage Description
- Dan satu baris dengan Privacy - Microphone Usage Description

Atau dengan format teks penambahan key:

```
<key>NSCameraUsageDescription</key>
<string>Can I use the camera please?</string>
<key>NSMicrophoneUsageDescription</key>
<string>Can I use the mic please?</string>
```

Android

Ubah minimum versi Android sdk ke 21 (atau lebih tinggi) pada file **android/app/build.gradle.**

```
minSdkVersion 21
```

Hal tersebut penting karena MediaRecorder tidak bekerja dengan baik di emulator sesuai dengan [dokumentasi.]

d. Implementasi kamera pada halaman Flutter

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

class CameraScreen extends StatefulWidget
{
  @override
  _CameraScreenState createState() =>
  _CameraScreenState();
}

class _CameraScreenState extends State<CameraScreen>
{ late CameraController _controller;
  late Future<void> _initializeControllerFuture;

  @override void
  initState() {
  super.initState();
```

```
    // Initialize camera on startup
    _initializeCamera();
  }

  Future<void> _initializeCamera() async {
    // Ambil daftar kamera yang tersedia di perangkat
    final cameras = await availableCameras();
    final firstCamera = cameras.first;

    // Buat kontroler kamera dan mulai kamera
    _controller = CameraController(
      firstCamera,
      ResolutionPreset.high,
    );

    _initializeControllerFuture = _controller.initialize();
  }

  @override
  void dispose() {
    // Bersihkan kontroler ketika widget dihapus
    _controller.dispose();
    super.dispose();
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text('Camera Example')),
      body: FutureBuilder<void>(
        future: _initializeControllerFuture,
        builder: (context, snapshot) {
          if (snapshot.connectionState ==
ConnectionState.done) {
            return CameraPreview(_controller);
          } else {
            return Center(child:
CircularProgressIndicator());
```

```
          }
        },
      ),
      floatingActionButton: FloatingActionButton(
        onPressed: () async {
          try {
            // Pastikan kamera sudah diinisialisasi
            await _initializeControllerFuture;
            // Ambil gambar
            final image = await _controller.takePicture();
            // Tampilkan atau gunakan gambar
            Navigator.push(
              context,
              MaterialPageRoute(
                builder:            (context)            =>
DisplayPictureScreen(imagePath: image.path),
              ),
            );
          } catch (e) {
            print(e);
          }
        },
        child: Icon(Icons.camera_alt),
      ),
    );
  }
}

class DisplayPictureScreen extends StatelessWidget {
  final String imagePath;

  const     DisplayPictureScreen({Key?     key,     required
this.imagePath}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text('Display Picture')),
```

```
      body: Image.file(File(imagePath)),
    );
  }
}
```

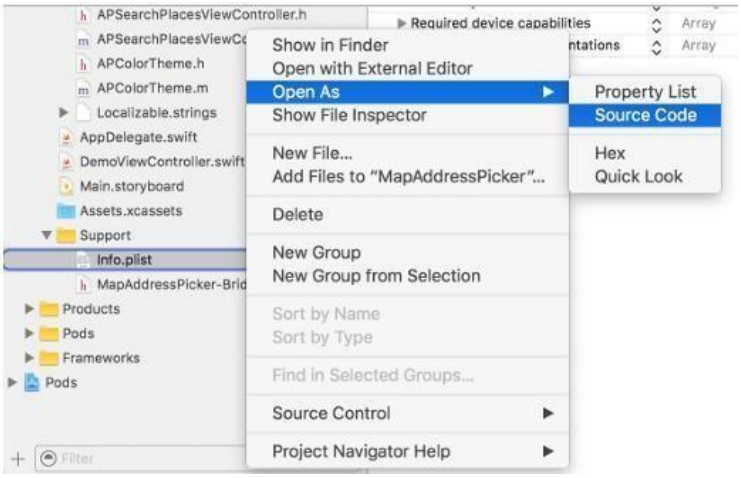
2. Media API

Media API adalah sekumpulan alat dan pustaka yang mendukung pengelolaan dan interaksi dengan berbagai jenis media, seperti gambar, video, dan audio. Flutter tidak memiliki API media bawaan untuk semua kebutuhan media, tetapi dapat menggunakan paket-paket tambahan untuk mengakses fitur media yang umum di aplikasi.

Pada bahasan kali ini, kita akan menggunakan packages atau plugin **Image Picker** supaya aplikasi dapat mengakses media galeri pada device. Pada platform iOS diperlukan konfigurasi tambahan, namun pada android tidak diperlukan konfigurasi tambahan. Berikut untuk konfigurasi tambahan:

A. iOS

Pada iOS diperlukan versi iOS 9.0.0 keatas, pastikan build version di set ke iOS 9.0.0 atau lebih tinggi. tambahkan konfigurasi pada file info.plist di Runner iOS. Kemudian ubah format tampilan ***.plist*** dengan tipe source code. Dapat dilihat pada gambar dibawah.



- Tambahkan potongan kode berikut pada ***info.plist***

```
                <key>NSCameraUsageDescription</key>
<string>Can I use the camera please?</string>
              <key>NSMicrophoneUsageDescription</key>
<string>Can I use the mic please?</string>
```

- Untuk pengunaan image picker, dapat dilihat pada potongan kode berikut :

```
import 'dart:io';

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

class ImageFromGalleryEx extends StatefulWidget {
  final ImageSourceType type;
  ImageFromGalleryEx(this.type);

  @override
  ImageFromGalleryExState       createState()       =>
ImageFromGalleryExState(this.type);
}

class          ImageFromGalleryExState          extends
State<ImageFromGalleryEx>{
  File? _image;
  late ImagePicker imagePicker;
  final ImageSourceType type;

  ImageFromGalleryExState(this.type);

  @override
  void initState() {
    super.initState();
    imagePicker = ImagePicker();
  }

  @override
```

```
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(type == ImageSourceType.camera
            ? "Image from Camera"
            : "Image from Gallery"),
      ),
      body: Column(
        children: <Widget>[
          SizedBox(height: 52),
          Center(
            //mengambil gambar dari camera atau gallery
            child: GestureDetector(
              onTap: () async {
                //operasi ternary untuk memilih sumber
gambar
                var      source      =      type      ==
ImageSourceType.camera
                    ? ImageSource.camera
                    : ImageSource.gallery;

                //menyimpan gambar pada variabel image
                XFile?       image       =       await
imagePicker.pickImage(
                    source: source,
                    imageQuality: 50,
                    preferredCameraDevice:
CameraDevice.front);

                if (image != null) {
                  setState(() {
                    _image = File(image.path);
                  });
                }
              },
              child: Container(
                width: 200,
                height: 200,
```

```
                decoration:       BoxDecoration(
                  color: Colors.red[200],
                ),

                // menampilkan gambar dari camera atau
gallery
                child:   _image   !=   null   ?
                    Image.file(       _image!,
                    width:   200.0,   height:
                    200.0,               fit:
                    BoxFit.fitHeight,
                      )

                    // jika tidak ada gambar yang
dipilih
                    :   Container(   decoration:
                        BoxDecoration(     color:
                        Colors.red[200],
                        ),  width:  200,  height:
                        200,     child:     Icon(
                        Icons.camera_alt,  color:
                        Colors.grey[800],
                        ),
                      ),
              ),
            ),
          ),
        ],
      ),
    );
  }
}

enum ImageSourceType { camera, gallery
}
```

Tugas Mandiri (Unguided)

1 (Soal) Modifikasi project pemilihan gambar yang telah dikerjakan pada Tugas Pendahuluan Modul 09 agar fungsionalitas tombol dapat berfungsi untuk mengunggah gambar.

2 Ketika tombol Gallery ditekan, aplikasi akan mengambil gambar dari galeri, dan setelah gambar dipilih, gambar tersebut akan ditampilkan di dalam container.

3 Ketika tombol Camera ditekan, aplikasi akan mengambil gambar menggunakan kamera, dan setelah pengambilan gambar selesai, gambar tersebut akan ditampilkan di dalam container.

4 Ketika tombol Hapus Gambar ditekan, gambar yang ada pada container akan dihapus.

Note: Jangan lupa sertakan source code, screenshoot output, dan deskripsi program.

Main.dart

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

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

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

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Image Picker Demo',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor:
Colors.deepPurple),
        useMaterial3: true,
      ),
      home: const HomeScreen(),
    );
  }
}

class HomeScreen extends StatefulWidget {
  const HomeScreen({Key? key}) : super(key: key);
```

```
  @override
  State<HomeScreen> createState() => _HomeScreenState();
}

class _HomeScreenState extends State<HomeScreen> {
  File? _image;
  final ImagePicker _picker = ImagePicker();

  Future<void> _pickImage(ImageSource source) async {
    final pickedFile = await _picker.pickImage(source: source);
    if (pickedFile != null) {
      setState(() {
        _image = File(pickedFile.path);
      });
    }
  }

  void _clearImage() {
    setState(() {
      _image = null;
    });
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text("Image Picker"),
      ),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            _image == null
                ? const Text("No image selected.")
                : Image.file(
                    _image!,
                    width: 200,
                    height: 200,
                    fit: BoxFit.cover,
```

```
                ),
            const SizedBox(height: 20),
            Row(
              mainAxisAlignment: MainAxisAlignment.center,
              children: [
                ElevatedButton(
                  onPressed: () =>
_pickImage(ImageSource.gallery),
                  child: const Text("Gallery"),
                ),
                const SizedBox(width: 10),
                ElevatedButton(
                  onPressed: () =>
_pickImage(ImageSource.camera),
                  child: const Text("Camera"),
                ),
                const SizedBox(width: 10),
                ElevatedButton(
                  onPressed: _clearImage,
                  child: const Text("Hapus Gambar"),
                ),
              ],
            ),
          ],
        ),
      ),
    );
  }
}
```

image_picker_screen.dart

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

enum ImageSourceType { gallery, camera }

class ImagePickerScreen extends StatefulWidget {
  final ImageSourceType type;
  const ImagePickerScreen({required this.type, Key? key}) :
super(key: key);

  @override
  State<ImagePickerScreen> createState() =>
_ImagePickerScreenState();
}

class _ImagePickerScreenState extends State<ImagePickerScreen> {
  File? _image;
  late final ImagePicker _imagePicker;

  @override
  void initState() {
    super.initState();
    _imagePicker = ImagePicker();
    _pickImage();
  }

  Future<void> _pickImage() async {
    try {
      final pickedFile = await _imagePicker.pickImage(
        source: widget.type == ImageSourceType.camera
            ? ImageSource.camera
            : ImageSource.gallery,
      );

      if (pickedFile != null) {
        setState(() {
          _image = File(pickedFile.path);
        });
      }
```

```
    } catch (e) {
      print("Error picking image: $e");
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: const Text("Image Picker")),
      body: Center(
        child: _image == null
            ? const Text("No image selected.")
            : Image.file(_image!),
      ),
    );
  }
}
```

display_screen.dart

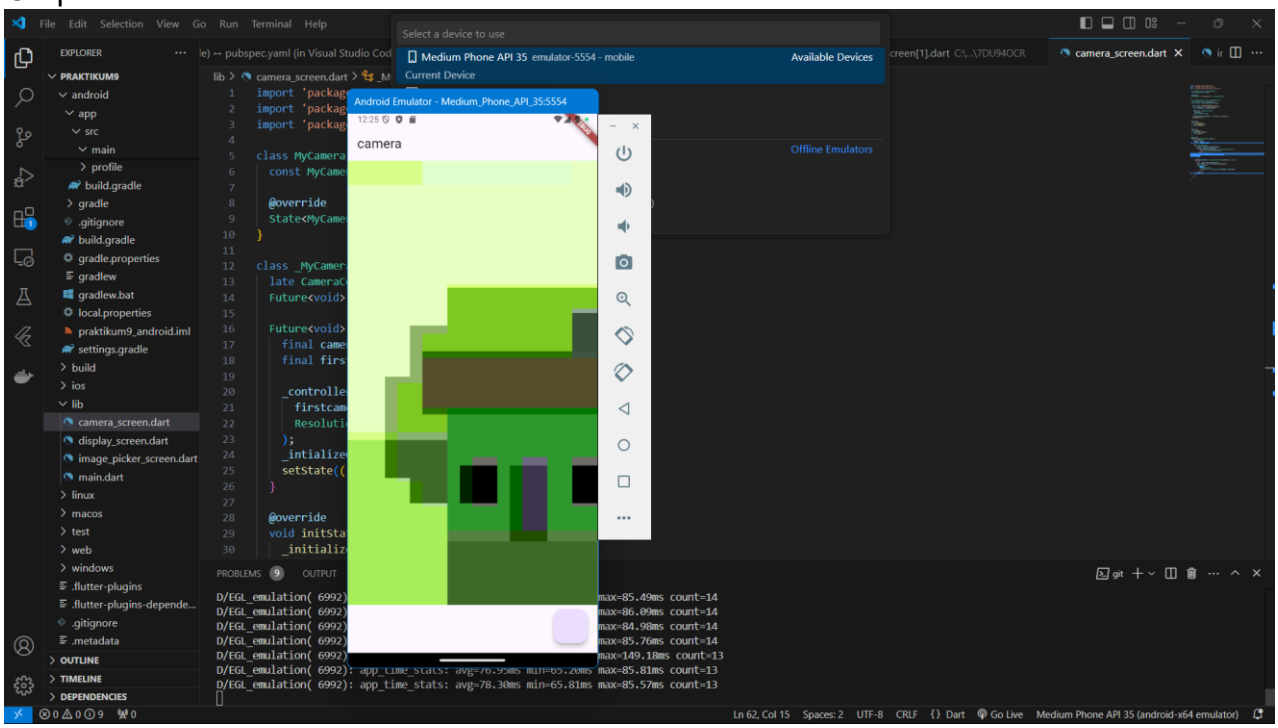
```
 import 'dart:io';
import 'package:flutter/material.dart';

class DisplayScreen extends StatelessWidget {
  final String imagePath;
  const DisplayScreen({Key? key, required this.imagePath}) :
super(key: key);

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Display'),
      ),
      body: Image.file(File(imagePath)),
    );
  }
}
```

camera_screen.dart

```
 import 'package:camera/camera.dart';
import 'package:flutter/material.dart';
import 'package:praktikum9/display_screen.dart';

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

  @override
  State<MyCamera> createState() => _MyCameraState();
}

class _MyCameraState extends State<MyCamera> {
  late CameraController _controller;
  Future<void>? _intializeControllerFuture;

  Future<void>? _initializeCamera() async {
    final cameras = await availableCameras();
    final firstcamera = cameras.first;

    _controller = CameraController(
      firstcamera,
      ResolutionPreset.high,
    );
    _intializeControllerFuture = _controller.initialize();
    setState(() {});
  }

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

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

  @override
```

```
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("camera"),
      ),
      body: FutureBuilder(
          future: _intializeControllerFuture,
          builder: (context, snapshot) {
            if (snapshot.connectionState == ConnectionState.done)
{
              return CameraPreview(_controller);
            } else {
              return Center(
                child: CircularProgressIndicator(),
              );
            }
          }),
      floatingActionButton: FloatingActionButton(onPressed: ()
async {
        try {
          await _intializeControllerFuture;
          final image = await _controller.takePicture();
          Navigator.push(
              context,
              MaterialPageRoute(
                builder: (_) => DisplayScreen(imagePath:
image.path),
              ));
        } catch (e) {
          print(e);
        }
      }),
    );
  }
}
```

Output:


Select a device to use
Medium Phone API 35 emulator-5554 - mobile
Available Devices
Current Device
Offline Emulators
Android Emulator - Medium_Phone_API_35:5554
camera
camera_screen.dart
PROBLEMS
OUTPUT
D/EGL_emulation(6992): app_time_stats: avg=78.30ms min=65.81ms max=85.57ms count=13