LAPORAN PRAKTIKUM
PEMROGRAMAN PERANGKAT BERGERAK

MODUL IX

API PERANGKAT KERAS

Disusun Oleh :

Zivana Afra Yulianto / 2211104039

SE-06-02

Asisten Praktikum :

Muhammad Faza Zulian Gesit Al Barru
Aisyah Hasna Aulia

Dosen Pengampu :

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

PROGRAM STUDI S1 SOFTWARE ENGINEERING

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

BAB I
PENDAHULUAN

A. DASAR TEORI

API_Perangkat_Keras:
API (*Application Programming Interface*) perangkat keras adalah antarmuka yang memungkinkan pengembang memanfaatkan fitur-fitur perangkat keras seperti kamera, sensor, atau penyimpanan dalam aplikasi. Pada aplikasi ini, digunakan API `camera` dan `image_picker` untuk mengakses kamera dan galeri perangkat.

- Library `camera`:
  - Digunakan untuk memanfaatkan fungsi kamera secara langsung, seperti mengambil gambar atau menampilkan *live preview*.
  - Memberikan kontrol penuh atas kamera perangkat, termasuk resolusi dan orientasi.
- Library `image_picker`:
  - Memungkinkan pengguna memilih gambar dari galeri atau menangkap gambar menggunakan kamera tanpa memerlukan kontrol penuh atas perangkat keras.
  - Sangat mudah digunakan untuk implementasi sederhana.
- Flutter Framework:
  - Kerangka kerja berbasis Dart yang memungkinkan pengembangan aplikasi lintas platform dengan kinerja mendekati aplikasi asli.

B. MAKSUD DAN TUJUAN

Maksud:
Mengembangkan aplikasi berbasis Flutter yang memanfaatkan API perangkat keras untuk mengambil, memilih, dan menampilkan gambar dari perangkat pengguna.
Tujuan:

- Memahami penggunaan API kamera (`camera`) dan galeri (`image_picker`) pada Flutter.
- Membuat antarmuka yang intuitif dan fungsional untuk interaksi pengguna dengan perangkat keras.
- Menerapkan animasi sederhana untuk meningkatkan pengalaman pengguna.

.

BAB II
IMPLEMENTASI

A. GUIDED

Code camera_screen,dart :

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

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

  @override
  _CameraScreenState createState() => _CameraScreenState();
}

class _CameraScreenState extends State<CameraScreen> {
  late CameraController _controller;
  Future<void>? _initializeControllerFuture; // Ubah late menjadi nullable

  @override
  void initState() {
    super.initState();
    _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();
    setState(() {}); // Memperbarui UI setelah inisialisasi
  }

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('Camera Initialization'),
        centerTitle: true,
        backgroundColor: Colors.greenAccent[600],
      ),
      body: FutureBuilder<void>(
        future: _initializeControllerFuture,
        builder: (context, snapshot) {
          if (snapshot.connectionState == ConnectionState.done) {
            return CameraPreview(_controller);
          } else {
            return const 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: (_) => DisplayPictureScreen(imagePath: image.path),
              ),
            );
```

```
          } catch (e) {
            print(e);
          }
        },
        child: const Icon(Icons.camera_alt),
      ),
    );
  }
}

class DisplayPictureScreen extends StatelessWidget {
  final String imagePath;

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

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

Code display_screen.dart :

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

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: const Text('Display Picture'),
        centerTitle: true,
        backgroundColor: Colors.greenAccent[600],
      ),
      body: Center(
        child: Image.file(File(imagePath)),
      ),
    );
  }
}
```

Code image_picker_screen.dart :

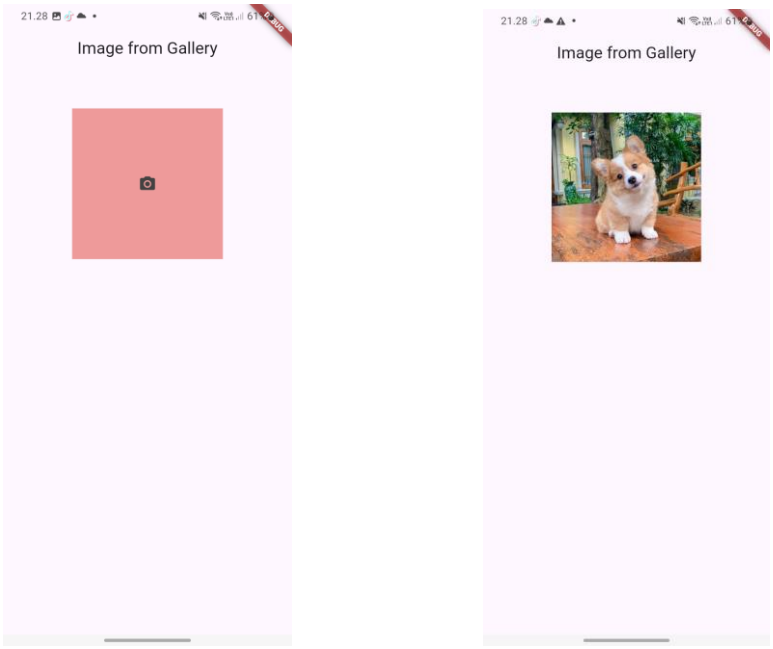
```
import 'dart:io';
import 'package:flutter/material.dart';
import 'package:image_picker/image_picker.dart';

enum ImageSourceType { gallery, camera }

class ImagePickerScreen extends StatefulWidget {
  final ImageSourceType type;

  const ImagePickerScreen({Key? key, required this.type}) : super(key: key);

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

```dart
class _ImagePickerScreenState extends State<ImagePickerScreen> {
  File? _image;
  late ImagePicker imagePicker;

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

  Future<void> _pickImage() async {
    // Pilih sumber gambar berdasarkan tipe yang diberikan
    final source = widget.type == ImageSourceType.camera
        ? ImageSource.camera
        : ImageSource.gallery;

    final pickedFile = await imagePicker.pickImage(
      source: source,
      imageQuality: 50, // Mengatur kualitas gambar
      preferredCameraDevice:
          CameraDevice.front, // Kamera depan jika menggunakan kamera
    );

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(
          widget.type == ImageSourceType.camera
              ? "Image from Camera"
              : "Image from Gallery",
        ),
        centerTitle: true,
      ),
      body: Column(
        children: <Widget>[
          const SizedBox(height: 52),
          Center(
            child: GestureDetector(
              onTap: _pickImage,
              child: Container(
                width: 200,
                height: 200,
                decoration: BoxDecoration(
                  color: Colors.red[200],
                ),
                // Menampilkan gambar dari kamera atau galeri
                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],
                        ),
                      ),
              ),
            ),
          ),
        ],
      ),
    );
  }
}
```

Code main.dart :

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

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

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

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
        useMaterial3: true,
      ),
      home: const ImagePickerScreen(type: ImageSourceType.gallery),
    );
  }
}
```

Screenshoot Output :



Deskipsi :

Aplikasi Flutter ini memungkinkan pengguna mengambil atau memilih gambar dan menampilkannya:

- **`camera_screen.dart`**: Menangani pengambilan gambar langsung dari kamera dan menampilkan hasilnya.
- **`display_screen.dart`**: Menampilkan gambar yang diambil atau dipilih.
- **`image_picker_screen.dart`**: Memilih gambar dari galeri atau mengambilnya menggunakan kamera.
- **`main.dart`**: Mengatur *entry point* aplikasi dengan layar awal untuk memilih gambar dari galeri.

B. UNGUIDED

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

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

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

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

main.dart:

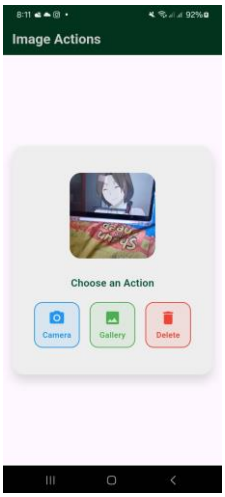
```
import 'package:flutter/material.dart';
import 'package:image_picker/image_picker.dart';
import 'dart:io';

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      home: ImageContainerScreen(),
    );
  }
}

class ImageContainerScreen extends StatefulWidget {
  @override
  _ImageContainerScreenState createState() => _ImageContainerScreenState();
}

class _ImageContainerScreenState extends State<ImageContainerScreen>
    with SingleTickerProviderStateMixin {
  late AnimationController _controller;
  File? _selectedImage;

  final ImagePicker _picker = ImagePicker();

  @override
  void initState() {
    super.initState();
    _controller = AnimationController(
      duration: Duration(seconds: 2),
      vsync: this,
    )..repeat(reverse: true);
  }

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

  Future<void> _pickImageFromGallery() async {
    final XFile? pickedFile =
        await _picker.pickImage(source: ImageSource.gallery);

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

  Future<void> _pickImageFromCamera() async {
    final XFile? pickedFile =
        await _picker.pickImage(source: ImageSource.camera);
```

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

  void _deleteImage() {
    setState(() {
      _selectedImage = null;
    });
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(
          'Image Actions',
          style: TextStyle(
            fontWeight: FontWeight.bold,
            fontSize: 24,
            color: const Color.fromARGB(255, 200, 201, 200),
          ),
        ),
        backgroundColor: const Color.fromARGB(255, 2, 52, 29),
      ),
      body: Center(
        child: Container(
          padding: EdgeInsets.all(20),
          decoration: BoxDecoration(
            color: Colors.grey[200],
            borderRadius: BorderRadius.circular(20),
            boxShadow: [
              BoxShadow(
                color: Colors.black.withOpacity(0.15),
                blurRadius: 15,
                offset: Offset(0, 10),
              ),
            ],
          ),
          width: 350,
          height: 400,
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              if (_selectedImage != null)
                ClipRRect(
                  borderRadius: BorderRadius.circular(20),
                  child: Image.file(
                    _selectedImage!,
                    width: 150,
                    height: 150,
                    fit: BoxFit.cover,
                  ),
                )
              else
                ScaleTransition(
                  scale: Tween<double>(begin: 0.8, end: 1.0).animate(
                    CurvedAnimation(
                        parent: _controller, curve: Curves.easeInOut),
                  ),
                  child: Icon(
                    Icons.image_outlined,
                    size: 100,
                    color: Colors.teal,
                  ),
                ),
              SizedBox(height: 30),
                "Choose an Action",
                style: TextStyle(
                  fontSize: 18,
                  fontWeight: FontWeight.w600,
                  color: const Color.fromARGB(255, 5, 85, 52),
                ),
              ),
              SizedBox(height: 20),
```

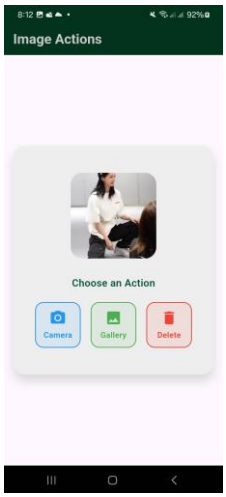
```
              Row(
                mainAxisAlignment: MainAxisAlignment.spaceEvenly,
                children: [
                  _buildActionButton(Icons.camera_alt, 'Camera', Colors.blue,
                      _pickImageFromCamera),
                  _buildActionButton(Icons.photo, 'Gallery', Colors.green,
                      _pickImageFromGallery),
                  _buildActionButton(
                      Icons.delete, 'Delete', Colors.red, _deleteImage),
                ],
              ),
            ],
          ),
        ),
      ),
    );
  }

  Widget _buildActionButton(
      IconData icon, String label, Color color, VoidCallback onTap) {
    return InkWell(
      onTap: onTap,
      borderRadius: BorderRadius.circular(10),
      child: Container(
        width: 80,
        height: 80,
        decoration: BoxDecoration(
          color: color.withOpacity(0.1),
          borderRadius: BorderRadius.circular(15),
          border: Border.all(color: color, width: 1.5),
        ),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Icon(icon, color: color, size: 30),
            SizedBox(height: 5),
            Text(
              label,
              style: TextStyle(
                color: color,
                fontWeight: FontWeight.w600,
              ),
            ),
          ],
        ),
      ),
    );
  }
}
```

Screenshoot Output

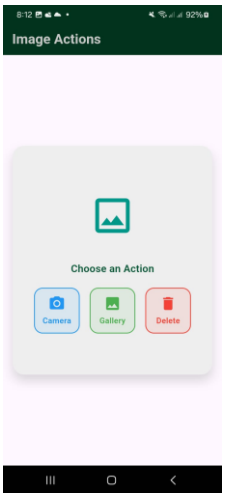
1. Home Page



(camera)



(gallery)



(delete)

Deskripsi Program
Aplikasi Flutter ini memungkinkan pengguna memilih gambar dari galeri, mengambil gambar menggunakan kamera, atau menghapus gambar yang dipilih.

- Fitur Utama:
  - Pilih gambar dari galeri.
  - Ambil gambar dengan kamera.
  - Hapus gambar yang dipilih.
- Tampilan:
  - Animasi ikon saat belum ada gambar.
  - Aksi dipilih melalui tombol dengan ikon dan label (*Camera*, *Gallery*, *Delete*).

BAB III
KESIMPULAN

A. KESIMPULAN

Dari implementasi dua aplikasi yang dibuat, dapat disimpulkan:

1. Fungsi API Perangkat Keras:
   Aplikasi berhasil memanfaatkan API perangkat keras untuk mengambil dan memilih gambar dengan memanfaatkan kamera dan galeri perangkat.
2. Penggunaan Library:
   - `camera` cocok untuk pengontrolan kamera tingkat lanjut seperti *live preview*.
   - `image_picker` sangat sederhana dan cocok untuk kebutuhan akses galeri dan kamera tanpa pengaturan kompleks.
3. Hasil Akhir:
   Aplikasi yang dihasilkan dapat menjalankan fitur-fitur utama dengan baik, termasuk mengambil gambar, menampilkan gambar, dan animasi pendukung untuk antarmuka. Hal ini menunjukkan pemanfaatan API perangkat keras pada Flutter dapat diimplementasikan secara efektif dan efisien.