LAPORAN PRAKTIKUM

PEMROGRAMAN PERANGKAT BERGERAK

MODUL IX

API PERANGKAT KERAS

DISUSUN OLEH :
MUHAMMAD FARIZ NUR HIDAYAT / 2211104069
KELAS : SE-06-2

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

API PERANGKAT KERAS

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.

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.

BUKTI

PRAKTIKUM

main.dart Source

Source Code

```dart
import 'package:flutter/material.dart';
import 'package:pert9/camera_screen.dart';
import 'package:pert9/image_picker_screen.dart';

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

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Guided_09',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
        useMaterial3: true,
      ),
      home: ImagePickerScreen(ImageSourceType.gallery),
      //MyCameraScreen(),
    );
  }
}

```

Deskripsi Program

Program ini adalah sebuah aplikasi Flutter sederhana yang dirancang untuk mendemonstrasikan pengambilan gambar menggunakan kamera atau memilih gambar dari galeri. Program memanfaatkan dua layar utama, yaitu:

1. ImagePickerScreen: Layar yang digunakan untuk memilih gambar dari galeri perangkat.
2. MyCameraScreen: Layar untuk mengambil gambar menggunakan kamera perangkat (komentar pada baris ini menunjukkan opsionalitas penggunaannya).

Fungsi Program

Kode ini adalah bagian dari aplikasi Flutter yang menavigasikan ke halaman pengambilan gambar. Program memiliki dua fitur utama:

1. **Gallery**: Menampilkan tampilan untuk memilih gambar dari galeri.
2. **Camera**: Menampilkan tampilan untuk mengambil gambar menggunakan kamera.

Image_picker_screen.dart

Source Code

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

class ImagePickerScreen extends StatefulWidget {
  final ImageSourceType type;

  const ImagePickerScreen(this.type);

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

enum ImageSourceType { camera, gallery } // Define enum for image source types

class _ImagePickerScreenState extends State<ImagePickerScreen> {
  File? image;
  late ImagePicker imagePicker;
  final ImageSourceType type;

  _ImagePickerScreenState(this.type) {
    imagePicker = ImagePicker(); // Initialize the ImagePicker
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(type == ImageSourceType.camera
            ? "Image from Camera"
            : "Image from Gallery"),
      ),
      body: Column(
        children: [
          SizedBox(height: 50),
          Center(
            child: GestureDetector(
              onTap: () async {
                // Choose image source (camera/gallery)
                final pickedFile = await imagePicker.pickImage(
                  source: type == ImageSourceType.camera
                      ? ImageSource.camera
                      : ImageSource.gallery,
                );

                if (pickedFile != null) {
                  setState(() {
                    image = File(pickedFile.path); // Update the image state
                  });
                }
              },
              child: Container(
                width: 100,
                height: 100,
                color: Colors.blue,
                child: Center(
                  child: Text(
                    'Pick Image',
                    style: TextStyle(color: Colors.white),
                  ),
                ),
              ),
            ),
          ),
          SizedBox(height: 20),
          image != null
              ? Image.file(
                  image!,
                  width: 100,
                  height: 100,
                  fit: BoxFit.cover,
                )
              : Container(), // Display the image if available
        ],
      ),
    );
  }
}

```

Deskripsi Program
Program di atas adalah implementasi sederhana dari halaman untuk memilih gambar menggunakan Flutter. Aplikasi ini memungkinkan pengguna untuk memilih gambar dari **galeri** atau mengambil gambar dengan **kamera** (berdasarkan parameter yang diberikan), lalu menampilkan gambar yang dipilih di layar.

Fitur Utama

1. **Memilih Sumber Gambar**:
    - **ImageSource.camera**: Mengambil gambar menggunakan kamera perangkat.
    - **ImageSource.gallery**: Memilih gambar dari galeri perangkat.
2. **Menampilkan Gambar**:
    - Setelah gambar dipilih, gambar tersebut akan ditampilkan dalam tampilan aplikasi.

Komponen Program

1. Library yang Digunakan

- **dart:io**:
    - Digunakan untuk bekerja dengan file sistem, termasuk gambar yang dipilih atau diambil.
- **image_picker**:
    - Library ini digunakan untuk memilih gambar dari galeri atau kamera perangkat.
- **material.dart**:
    - Library bawaan Flutter untuk membuat antarmuka pengguna.

2. ImageSourceType Enum

```
enum ImageSourceType { camera, gallery }
```

- Mendefinisikan tipe sumber gambar:
    - **camera**: Kamera perangkat.
    - **gallery**: Galeri perangkat.

3. ImagePickerScreen Widget

- Widget ini adalah layar utama untuk memilih gambar.
- Parameter **type** menentukan apakah gambar diambil dari kamera atau galeri.

4. _ImagePickerScreenState Class

- **StatefulWidget** untuk mengelola perubahan keadaan aplikasi.
- Komponen utama:
    - **imagePicker**: Objek dari library **image_picker** untuk mengakses gambar.
    - **image**: Variabel untuk menyimpan gambar yang dipilih atau diambil.

5. Fungsi untuk Memilih Gambar

```
final pickedFile = await imagePicker.pickImage(
  source: type == ImageSourceType.camera
      ? ImageSource.camera
      : ImageSource.gallery,
);

if (pickedFile != null) {
```

```
    setState(() {
      image = File(pickedFile.path);
    });
  }
```

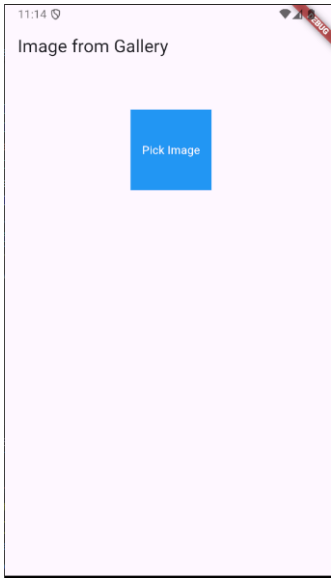
- Fungsi **pickImage** digunakan untuk mengambil gambar dari kamera atau galeri berdasarkan parameter **type**.
- Path gambar yang dipilih disimpan dalam variabel **image**, lalu UI diperbarui dengan **setState**.

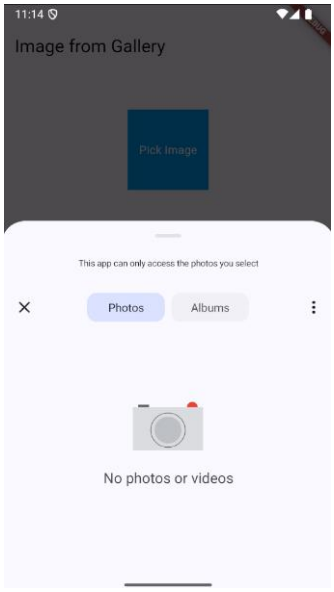
6. Tampilan Gambar

```
image != null
  ? Image.file(
      image!,
      width: 100,
      height: 100,
      fit: BoxFit.cover,
    )
  : Container(),
```

- Jika gambar telah dipilih, gambar ditampilkan menggunakan **Image.file**.
- Jika tidak ada gambar, area kosong ditampilkan.
-

Output




11:14
Image from Gallery
Pick Image
This app can only access the photos you select
Photos
Albums
No photos or videos

display_screen.dart

Source Code

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

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Display Screen'),
        centerTitle: true,
        backgroundColor: Colors.greenAccent[600],
      ),
      body: Image.file(File(imagePath)),
    );
  }
}

```

Deskripsi Program

Program di atas adalah implementasi dari sebuah halaman bernama DisplayScreen yang digunakan untuk menampilkan gambar dari file lokal (dalam bentuk path file). Program ini cocok digunakan untuk mempratinjau gambar yang telah diambil atau dipilih dari kamera atau galeri pada aplikasi berbasis Flutter.

Fitur Utama

1. Menampilkan Gambar dari Path:
    - Gambar ditampilkan berdasarkan path file yang diberikan sebagai parameter.
    - File gambar diakses menggunakan package Dart bawaan, dart:io.
2. Antarmuka Pengguna Sederhana:
    - Halaman memiliki judul di AppBar dan langsung menampilkan gambar di layar.
3. Parameter Dinamis:
    - Path gambar diterima melalui parameter imagePath, yang memungkinkan halaman ini menampilkan gambar apa pun sesuai dengan path yang diberikan.

---

1. **Konstruktor DisplayScreen**:

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

    - **imagePath**:

        - Parameter wajib yang berisi path dari gambar yang akan ditampilkan.
        - Parameter ini memungkinkan gambar dinamis untuk ditampilkan sesuai input yang diberikan.
2. **Scaffold**:
    - Digunakan untuk membuat struktur dasar halaman.
    - **AppBar**:
        - Judul halaman adalah "Display Screen".
        - Posisi judul berada di tengah (**centerTitle: true**).
        - Warna latar belakang **AppBar** adalah hijau muda (**Colors.greenAccent[600]**).
    - **Body**:
        - Menampilkan gambar yang diakses dari path menggunakan widget **Image.file**:

```
Image.file(File(imagePath))
```

        - **File** dari **dart:io** digunakan untuk membaca file gambar dari path yang diberikan.

camera_screen.dart

Source Code

```dart
import 'package:camera/camera.dart';
import 'package:flutter/material.dart';
import 'package:pert9/display_screen.dart';

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

  @override
  State<MyCameraScreen> createState() => _MyCameraScreenState();
}

class _MyCameraScreenState extends State<MyCameraScreen> {
  late CameraController _controller;
  Future<void>? _initializeControllerFuture;

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

    _controller = CameraController(
      firstCamera,
      ResolutionPreset.high,
    );

    _initializeControllerFuture = _controller.initialize();
    setState(() {});
  }

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Camera Implementation'),
        centerTitle: true,
        backgroundColor: Colors.greenAccent[600],
      ),
      body: FutureBuilder(
        future: _initializeControllerFuture,
        builder: (context, snapshot) {
          if (snapshot.connectionState == ConnectionState.done) {
            return CameraPreview(_controller);
          } else {
            return Center(
              child: CircularProgressIndicator(),
            );
          }
        },
      ),
      floatingActionButton: FloatingActionButton(onPressed: () async {
        try {
          await _initializeControllerFuture;
          final image = await _controller.takePicture();
          Navigator.push(
            context,
            MaterialPageRoute(
              builder: (_) => DisplayScreen(
                imagePath: image.path,
              ),
            ),
          );
        } catch (e) {
          print(e);
        }
      }),
    );
  }
}

```

Deskripsi Program

Program di atas adalah implementasi halaman **kamera** menggunakan Flutter. Halaman ini memungkinkan pengguna untuk mengambil gambar menggunakan kamera perangkat, kemudian menampilkan hasilnya di halaman **DisplayScreen**.

Fitur Utama

1. **Inisialisasi Kamera**:
   - Mengakses kamera perangkat yang tersedia.
   - Mengatur resolusi kamera ke kualitas tinggi (**ResolutionPreset.high**).
2. **Pratinjau Kamera**:
   - Menampilkan tampilan langsung dari kamera perangkat melalui widget **CameraPreview**.
3. **Pengambilan Gambar**:
   - Mengambil gambar menggunakan tombol **FloatingActionButton**.
   - Setelah gambar diambil, pengguna diarahkan ke halaman **DisplayScreen** untuk melihat hasilnya.

Komponen Utama

1. **Library yang Digunakan**:
   - **camera**:
     - Library ini digunakan untuk mengakses kamera perangkat, mengambil gambar, dan mengatur pratinjau.
   - **display_screen.dart**:
     - Halaman ini digunakan untuk menampilkan gambar yang diambil.
2. **Fungsi _initializeCamera**:

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

  _controller = CameraController(
    firstCamera,
    ResolutionPreset.high,
  );

  _initializeControllerFuture = _controller.initialize();
  setState(() {});
}
```

   - Mengambil daftar kamera yang tersedia menggunakan **availableCameras()**.
   - Memilih kamera pertama sebagai default.
   - Membuat dan menginisialisasi **CameraController** untuk mengatur kamera.
3. **Pratinjau Kamera**:

```
FutureBuilder(
  future: _initializeControllerFuture,
  builder: (context, snapshot) {
```

```
        if (snapshot.connectionState == ConnectionState.done) {
          return CameraPreview(_controller);
        } else {
          return Center(
            child: CircularProgressIndicator(),
          );
        }
      },
    )
```

- Menggunakan **FutureBuilder** untuk memastikan kamera telah diinisialisasi sebelum menampilkan pratinjau.
- Menampilkan **CameraPreview** jika kamera siap.
- Menampilkan **CircularProgressIndicator** saat kamera sedang diinisialisasi.

4. **Pengambilan Gambar**:

```
    floatingActionButton: FloatingActionButton(
      onPressed: () async {
        try {
          await _initializeControllerFuture;
          final image = await _controller.takePicture();
          Navigator.push(
            context,
            MaterialPageRoute(
              builder: (_) => DisplayScreen(
                imagePath: image.path,
              ),
            ),
          );
        } catch (e) {
          print(e);
        }
      },
    ),
```

- Tombol melayang (FloatingActionButton) digunakan untuk mengambil gambar.
- **takePicture()**:
  - Mengambil gambar dari kamera.
  - Menyimpan path gambar dalam variabel **image**.
- Mengarahkan pengguna ke **DisplayScreen** dengan path gambar sebagai parameter.

Output

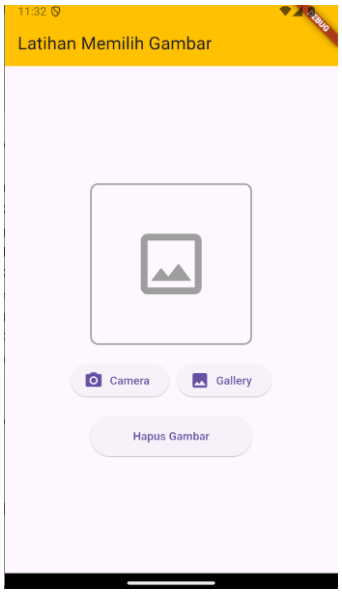
Tugas Mandiri (Unguided)

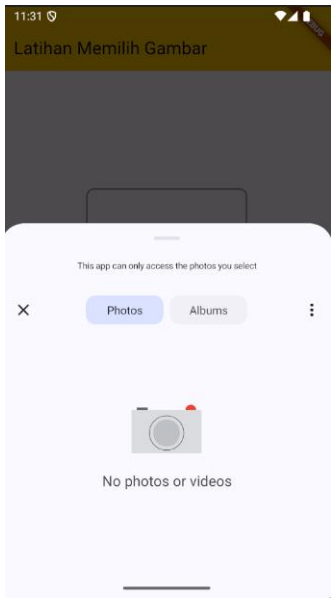
Main.dart

Source Code

```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(
      home: ImagePickerDemo(),
    );
  }
}

class ImagePickerDemo extends StatefulWidget {
  @override
  _ImagePickerDemoState createState() => _ImagePickerDemoState();
}

class _ImagePickerDemoState extends State<ImagePickerDemo> {
  File? _image; // Menyimpan gambar yang dipilih

  final ImagePicker _picker = ImagePicker();

  // Fungsi untuk mengambil gambar dari kamera
  Future<void> _pickFromCamera() async {
    final pickedFile = await _picker.pickImage(source: ImageSource.camera);
    if (pickedFile != null) {
      setState(() {
        _image = File(pickedFile.path);
      });
    }
  }

  // Fungsi untuk mengambil gambar dari galeri
  Future<void> _pickFromGallery() async {
    final pickedFile = await _picker.pickImage(source: ImageSource.gallery);
    if (pickedFile != null) {
      setState(() {
        _image = File(pickedFile.path);
      });
    }
  }

  // Fungsi untuk menghapus gambar
  void _clearImage() {
    setState(() {
      _image = null;
    });
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Latihan Memilih Gambar'),
        backgroundColor: Colors.amber,
      ),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: <Widget>[
            Container(
              width: 200,
              height: 200,
              decoration: BoxDecoration(
                border: Border.all(color: Colors.grey, width: 2),
                borderRadius: BorderRadius.circular(10),
              ),
              child: _image != null
                  ? ClipRRect(
                      borderRadius: BorderRadius.circular(10),
                      child: Image.file(
                        _image!,
                        fit: BoxFit.cover,
                      ),
                    )
                  : Icon(
                      Icons.image_outlined,
                      size: 100,
                      color: Colors.grey,
                    ),
            ),
            SizedBox(height: 20),
            Row(
              mainAxisAlignment: MainAxisAlignment.center,
              children: [
                ElevatedButton.icon(
                  onPressed: _pickFromCamera,
                  icon: Icon(Icons.camera_alt),
                  label: Text('Camera'),
                ),
                SizedBox(width: 10),
                ElevatedButton.icon(
                  onPressed: _pickFromGallery,
                  icon: Icon(Icons.photo),
                  label: Text('Gallery'),
                ),
              ],
            ),
            SizedBox(height: 20),
            ElevatedButton(
              onPressed: _clearImage,
              child: Text('Hapus Gambar'),
              style: ElevatedButton.styleFrom(
                minimumSize: Size(200, 50),
              ),
            ),
          ],
        ),
      ),
    );
  }
}
```

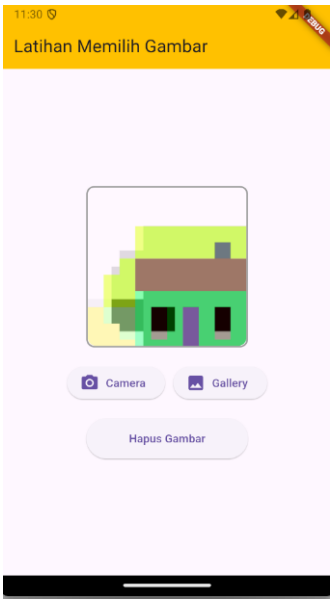
1. (Soal) Modifikasi project pemilihan gambar yang telah dikerjakan pada Tugas Pendahuluan Modul 09 agar fungsionalitas tombol dapat berfungsi untuk mengunggah gambar.
    - Ketika tombol Gallery ditekan, aplikasi akan mengambil gambar dari galeri, dan setelah gambar dipilih, gambar tersebut akan ditampilkan di dalam container.



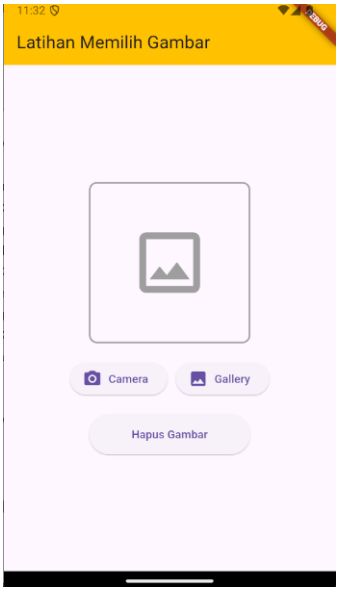


    - Ketika tombol Camera ditekan, aplikasi akan mengambil gambar menggunakan kamera, dan setelah pengambilan gambar selesai, gambar

tersebut akan ditampilkan di dalam container.



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



Deskripsi Program

Program di atas adalah aplikasi sederhana berbasis Flutter yang memungkinkan pengguna untuk memilih gambar dari **kamera** atau **galeri**, menampilkan gambar yang dipilih di layar, dan menyediakan opsi untuk **menghapus gambar**.

Fitur Utama

1. **Memilih Gambar dari Kamera**:

    - Menggunakan kamera perangkat untuk mengambil gambar dan menampilkannya di aplikasi.
2. **Memilih Gambar dari Galeri**:
    - Mengambil gambar dari galeri perangkat untuk ditampilkan.
3. **Menampilkan Gambar**:
    - Gambar yang dipilih atau diambil akan ditampilkan di area khusus di layar.
4. **Menghapus Gambar**:
    - Memberikan opsi untuk menghapus gambar yang telah dipilih, sehingga area tampilan gambar kembali kosong.