

**LAPORAN PRAKTIKUM**  
**PEMROGRAMAN PERANGKAT BERGERAK**

**MODUL 9**  
**API PERANGKAT KERAS**



**Disusun Oleh :**

**Aji Prasetyo Nugroho / 2211104049**

**S1SE-06-2**

**Asisten Praktikum :**

**Muhammad Faza Zulian Gesit Al Barru**

**Aisyah Hasna Aulia**

**Dosen Pengampu :**

**Yudha Islami Sulistya**

**PROGRAM STUDI S1 SOFTWARE ENGINEERING**  
**FAKULTAS INFORMATIKA**  
**TELKOM UNIVERSITY PURWOKERTO**

**2024**

## A. GUIDED

- Camera Screen

Source code :

```
import 'package:camera/camera.dart';
import 'package:flutter/material.dart';
import 'package:pertemuan_9/display_screen.dart';

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

  @override
  State<CameraScreen> createState() => _CameraScreenState();
}

class _CameraScreenState extends State<CameraScreen> {
  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() {
    super.initState();
    _initializeCamera();
  }

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('Camera Flutter'),
        centerTitle: true,
        backgroundColor: Colors.greenAccent,
      ),
      body: FutureBuilder(
        future: _initializeControllerFuture,
        builder: (context, snapshot) {
          if (snapshot.connectionState == ConnectionState.done) {
            return CameraPreview(_controller);
          } else {
            return const 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);
          }
        },
        child: const Icon(Icons.camera_alt),
      ),
    );
  }
}
```

- Display Screen

Source Code :

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


class DisplayScreen extends StatelessWidget {
  final String imagePath;

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('Display Screen'),
        centerTitle: true,
        backgroundColor: Colors.greenAccent,
        actions: [
          IconButton(
            icon: const Icon(Icons.share),
            onPressed: () {},
          ),
        ],
      ),
      body: Column(
        children: [
          Expanded(
            child: Image.file(
              File(imagePath),
              fit: BoxFit.contain,
            ),
          ),
          Padding(
            padding: const EdgeInsets.all(16.0),
            child: Row(
              mainAxisAlignment: MainAxisAlignment.spaceEvenly,
              children: [
                ElevatedButton(
                  onPressed: () {
                    Navigator.pop(context);
                  },
                  child: const Text('Take Another Photo'),
                ),
                ElevatedButton(
                  onPressed: () {},
                  child: const Text('Save Photo'),
                ),
              ],
            ),
          ),
        ],
      ),
    );
  }
}
```

- Main

Source Code :



```
import 'package:flutter/material.dart';
import 'package:pertemuan_9/camera_screen.dart';
import 'package:pertemuan_9/image_picker.dart';

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

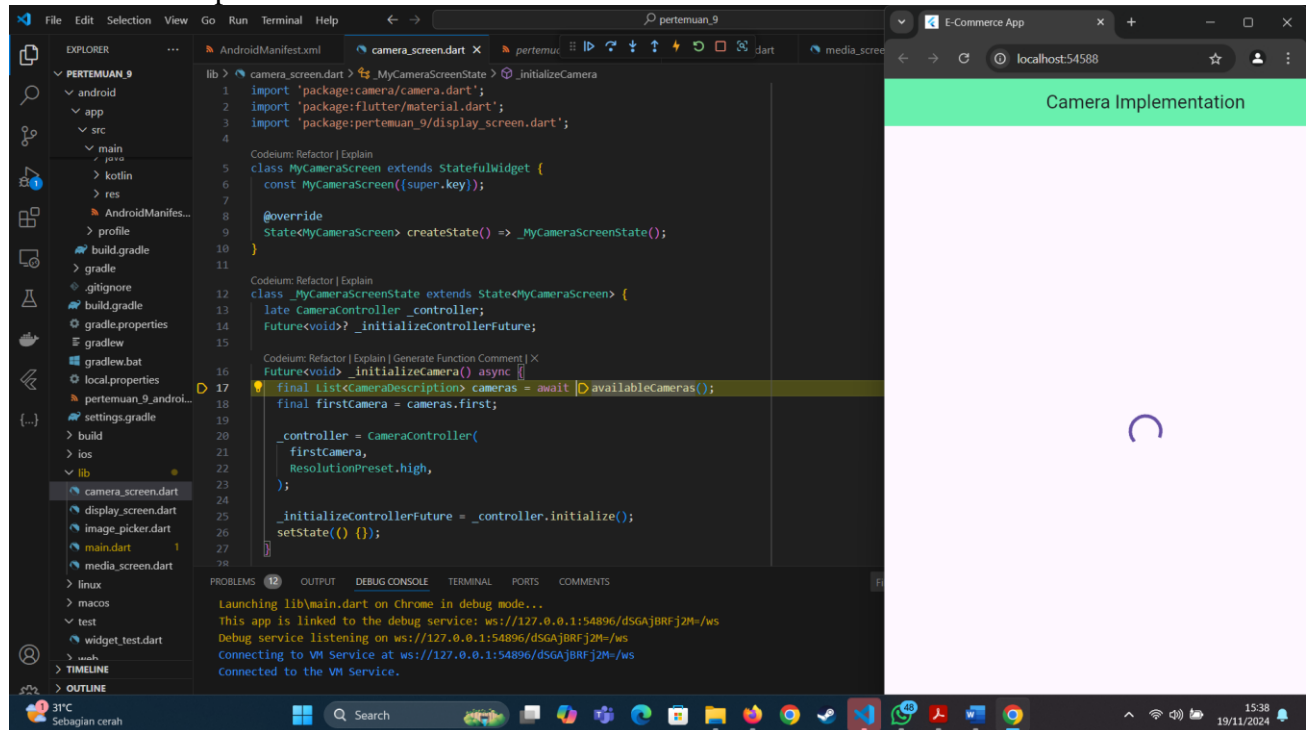
class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'E-Commerce App',
      theme: ThemeData(
        primarySwatch: Colors.blue,
      ),
      home: MyCameraScreen(

        //      ImagePickerScreen(
        //      ImageSourceType.gallery,
        //    ),

    );
  }
}
```

Output :



## Deskripsi Program

Kode tersebut adalah implementasi aplikasi Flutter untuk mengambil gambar menggunakan kamera perangkat. Aplikasi ini menggunakan paket camera untuk mengakses kamera, memulai CameraController, dan menampilkan pratinjau kamera melalui widget CameraPreview. Setelah inisialisasi kamera berhasil menggunakan FutureBuilder, pengguna dapat mengambil foto dengan tombol floating action button (FAB). Saat tombol ditekan, aplikasi mengambil gambar dan navigasi dilakukan ke layar DisplayScreen untuk menampilkan gambar yang diambil, dengan path gambar dikirim sebagai argumen. Kode juga menangani inisialisasi dan pelepasan sumber daya kamera di lifecycle widget (initState dan dispose) agar aplikasi berjalan lancar dan menghindari kebocoran memori.

- Image Picker  
Source code :

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

class ImagePickerScreen extends StatefulWidget {
  final ImageSourceType type;

  ImagePickerScreen(this.type);

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

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

  ImagePickerScreenState(this.type);

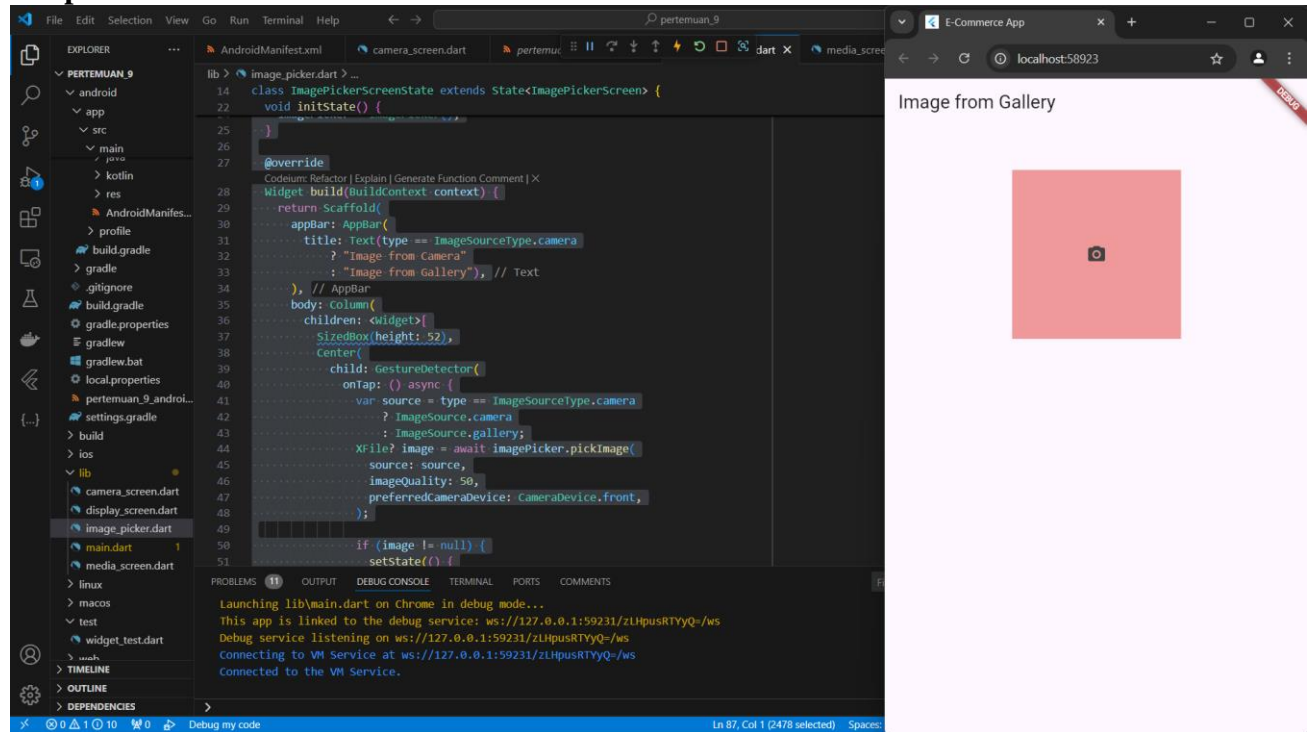
  @override
  void initState() {
    super.initState();
    imagePicker = ImagePicker();
  }

  @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(
            child: GestureDetector(
              onTap: () async {
                var source = type == ImageSourceType.camera
                  ? ImageSource.camera
                  : ImageSource.gallery;
                XFile? image = await imagePicker.pickImage(
                  source: source,
                  imageQuality: 50,
                  preferredCameraDevice: CameraDevice.front,
                );

                if (image != null) {
                  setState(() {
                    _image = File(image.path);
                  });
                } else {
                  ScaffoldMessenger.of(context).showSnackBar(
                    SnackBar(content: Text('No image selected!')),
                  );
                }
              },
            ),
          ),
          Container(
            width: 200,
            height: 200,
            decoration: BoxDecoration(
              color: Colors.red[200],
            ),
          ),
          child: _image != null
            ? Image.file(
                _image!,
                width: 200.0,
                height: 200.0,
                fit: BoxFit.fitHeight,
              )
            : Icon(
                Icons.camera_alt,
                color: Colors.grey[800],
              ),
        ],
      ),
    );
  }
}

enum ImageSourceType { camera, gallery }
```

## Output :



## Deskripsi Program :

Kode ini merupakan implementasi dari aplikasi Flutter yang memungkinkan pengguna memilih atau mengambil gambar dari kamera atau galeri pada perangkat. Aplikasi ini menggunakan paket `image_picker` untuk mengakses gambar dari dua sumber yang berbeda, yaitu kamera dan galeri. Jenis sumber gambar ditentukan oleh enum `ImageSourceType` yang dikirimkan ke widget `ImagePickerScreen` ketika diinisialisasikan. Pada tampilan utama terdapat sebuah area widget dimana diizinkan pengguna untuk mengetuk pada persegi yang berwarna merah, untuk memilih atau mengambil gambar. Jika gambar berhasil diambil dipilih, gambar tersebut akan tampil di area tersebut, jika gambar tidak di pilih maka akan menampilkan pesan pemberitahuan melalui `snackbar`. Selain itu, implementasi ini akan melakukan handling pengambilan gambar dari kamera yang baku dan pengambilan gambar harus di kompresi kualitasnya hingga 50%.

## B. Unguided

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 container akan dihapus.

### Source Code :

#### a. Main

```
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(
      title: 'Latihan Memilih Gambar',
      theme: ThemeData(
        primarySwatch: Colors.yellow,
      ),
      home: ImageSelectionScreen(),
    );
  }
}

class ImageSelectionScreen extends StatefulWidget {
  @override
  _ImageSelectionScreenState createState() => _ImageSelectionScreenState();
}

class _ImageSelectionScreenState extends State<ImageSelectionScreen> {
  File? _selectedImage;

  final ImagePicker _picker = ImagePicker();

  Future<void> _pickImageFromGallery() async {
    try {
      final pickedFile = await _picker.pickImage(source: ImageSource.gallery);
      if (pickedFile != null) {
```



```

        setState(() {
          _selectedImage = File(pickedFile.path);
        });
      } else {
        print('Tidak ada gambar yang dipilih.');
```

```

      }
    } catch (e) {
      print('Error memilih gambar dari galeri: $e');
    }
  }

Future<void> _pickImageFromCamera() async {
  try {
    final pickedFile = await _picker.pickImage(source: ImageSource.camera);
    if (pickedFile != null) {
      setState(() {
        _selectedImage = File(pickedFile.path);
      });
    } else {
      print('Tidak ada gambar yang diambil.');
```

```

    }
  } catch (e) {
    print('Error mengambil gambar dari kamera: $e');
  }
}

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

```

```

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('Latihan Memilih Gambar'),
      backgroundColor: Colors.yellow[700],
    ),
    body: Padding(
      padding: const EdgeInsets.all(16.0),
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          Container(
            height: 150,
            width: 150,
            decoration: BoxDecoration(
              color: Colors.grey[200],
              border: Border.all(color: Colors.grey),
              borderRadius: BorderRadius.circular(8),
            ),
            child: _selectedImage == null

```

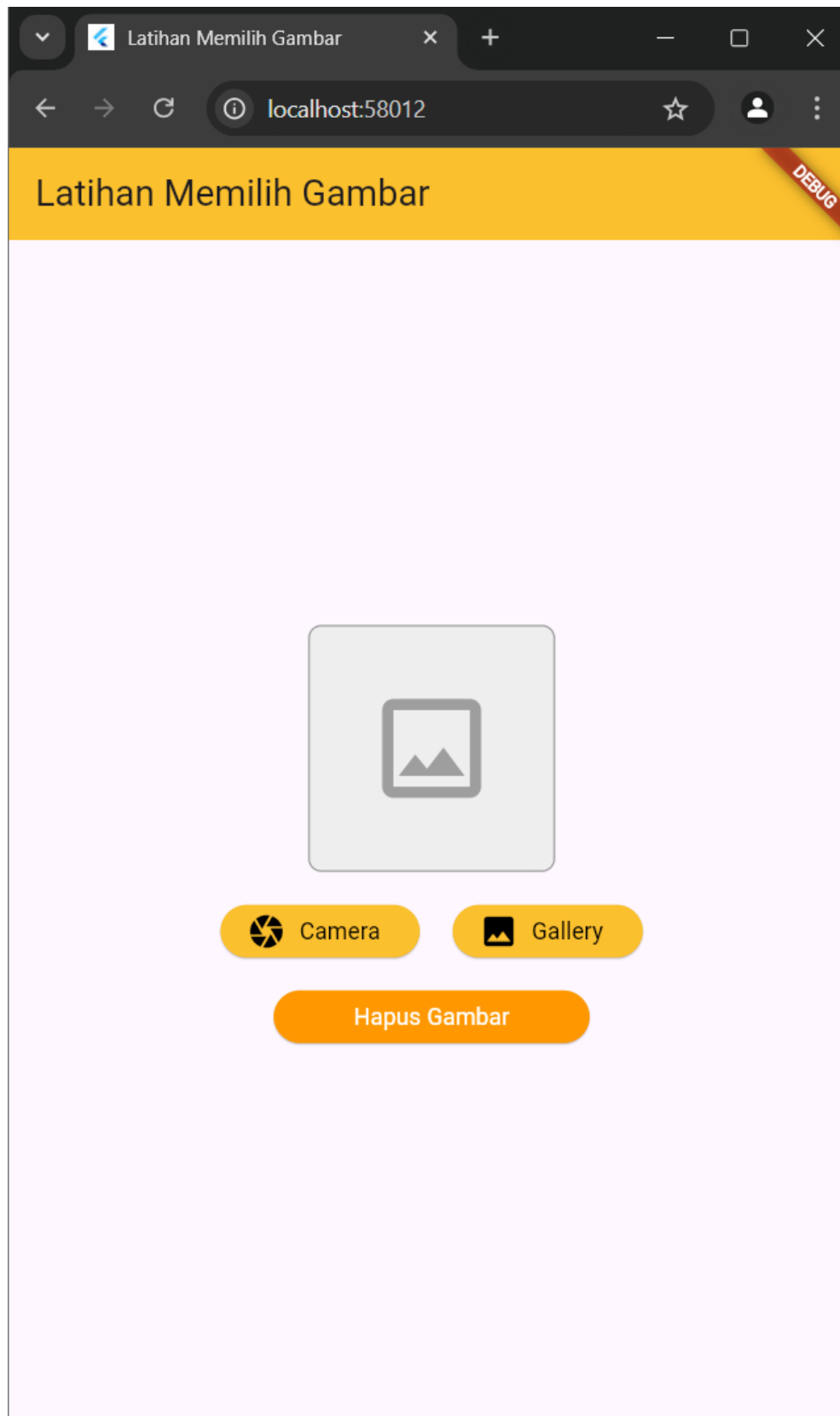
```

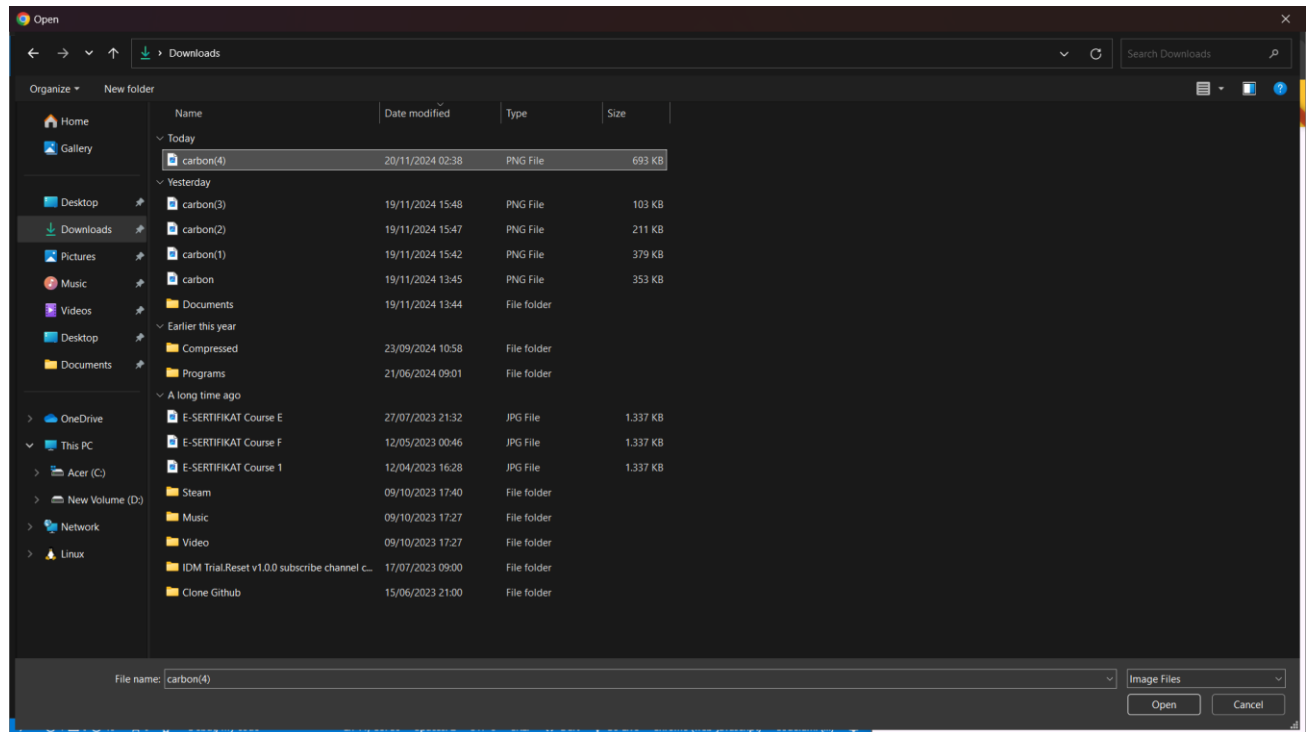
        ? Icon(
            Icons.image_outlined,
            size: 80,
            color: Colors.grey,
        )
        : Image.file(
            _selectedImage!,
            fit: BoxFit.cover,
        ),
    ),
    SizedBox(height: 20),
    Row(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
            ElevatedButton.icon(
                onPressed: _pickImageFromCamera,
                icon: Icon(Icons.camera),
                label: Text("Camera"),
                style: ElevatedButton.styleFrom(
                    backgroundColor: Colors.yellow[700],
                    foregroundColor: Colors.black,
                    shape: RoundedRectangleBorder(
                        borderRadius: BorderRadius.circular(20),
                    ),
                ),
            ),
            SizedBox(width: 20),
            ElevatedButton.icon(
                onPressed: _pickImageFromGallery,
                icon: Icon(Icons.photo),
                label: Text("Gallery"),
                style: ElevatedButton.styleFrom(
                    backgroundColor: Colors.yellow[700],
                    foregroundColor: Colors.black,
                    shape: RoundedRectangleBorder(
                        borderRadius: BorderRadius.circular(20),
                    ),
                ),
            ),
        ],
    ),
    SizedBox(height: 20),
    ElevatedButton(
        onPressed: _clearImage,
        child: Text("Hapus Gambar"),
        style: ElevatedButton.styleFrom(
            backgroundColor: Colors.orange,
            foregroundColor: Colors.white,
            shape: RoundedRectangleBorder(
                borderRadius: BorderRadius.circular(20),
            ),
        ),
        minimumSize: Size(200, 40),
    ),

```

```
    },  
    ],  
  },  
  },  
);  
}  
}
```

Output :





## Deskripsi kode :

Kode tersebut merupakan aplikasi Flutter sederhana yang memungkinkan pengguna untuk memilih gambar dari galeri atau mengambil gambar menggunakan kamera, lalu menampilkannya di dalam sebuah container. Aplikasi ini menggunakan plugin `image_picker` untuk mengakses kamera dan galeri. Gambar yang dipilih atau diambil ditampilkan dalam widget `Image.file` di dalam container, sementara ikon default akan ditampilkan jika belum ada gambar yang dipilih. Terdapat tiga tombol utama: tombol "Camera" untuk membuka kamera, tombol "Gallery" untuk membuka galeri, dan tombol "Hapus Gambar" untuk menghapus gambar yang telah dipilih atau diambil. Fungsi ini dibangun menggunakan `stateful widget` untuk memastikan UI diperbarui setiap kali gambar dipilih, dihapus, atau diambil dari kamera. Aplikasi ini juga menangani error dengan logging sederhana saat terjadi kesalahan saat mengambil atau memilih gambar.