MAD and PWA Lab

Name: Vrushabh Ghuse **Class: D15A** **Roll no:22**

Aim: To design Flutter UI by including common widgets.

Theory:

We can split the Flutter widget into two categories:

1. Visible (Output and Input)
2. Invisible (Layout and Control)

1. Visible widget

The visible widgets are related to the user input and output data. Some of the important types of this widget are:

1. Text

A Text widget holds some text to display on the screen. We can align the text widget by using textAlign property, and style property allow the customization of Text that includes font, font weight, font style, letter spacing, color, and many more.

2. Button

This widget allows you to perform some action on click. Flutter does not allow
you to use the Button widget directly; instead, it uses a type of buttons like a FlatButton and a RaisedButton.

3. Image

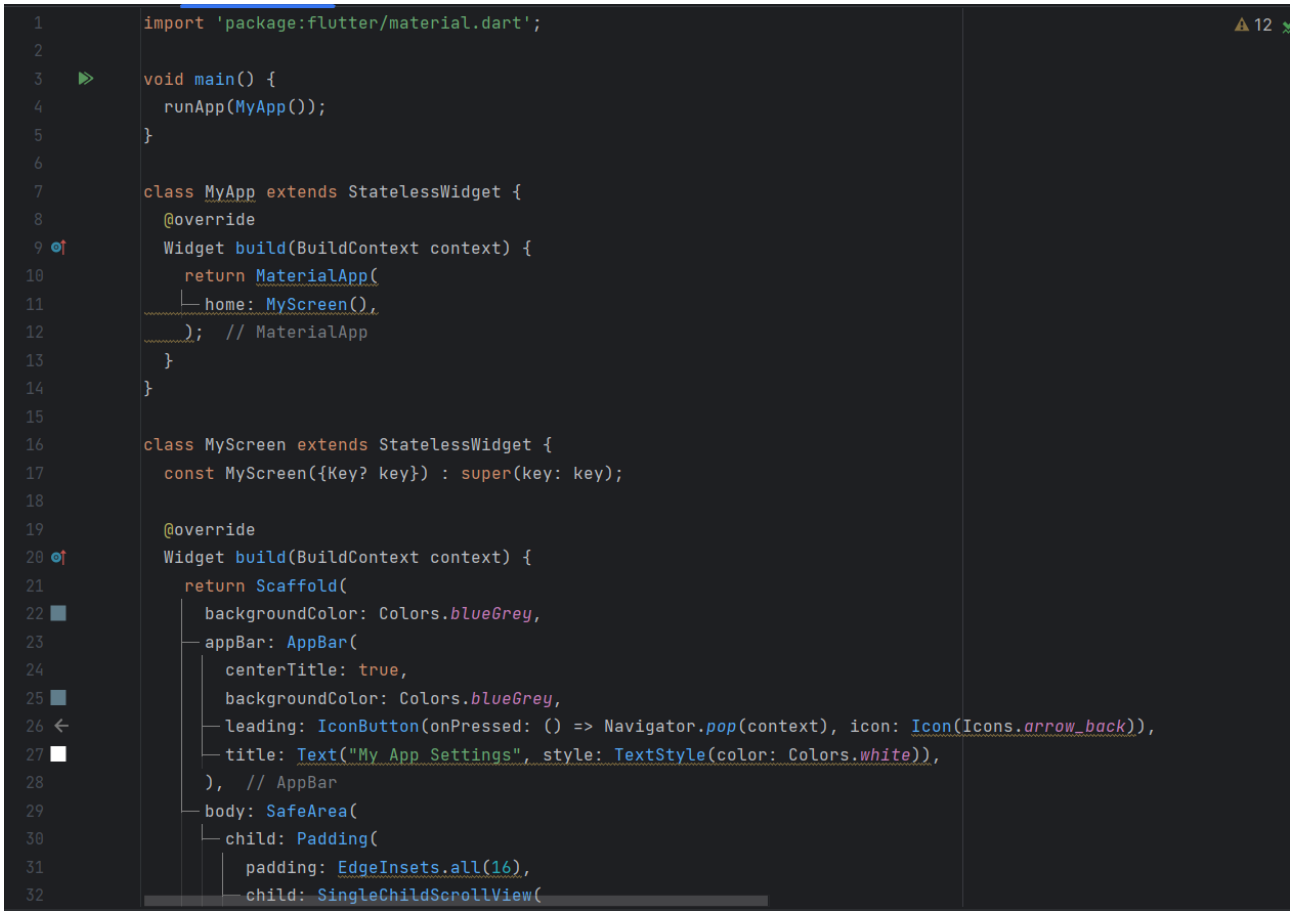
This widget holds the image which can fetch it from multiple sources like from the asset folder or directly from the URL. It provides many constructors for

loading image, which are given below: o Image: It is a generic image loader, which is used by ImageProvider. o asset: It load image from your project asset folder. o file: It loads images from the system folder.

o memory: It load image from memory. o

network: It loads images from the network.

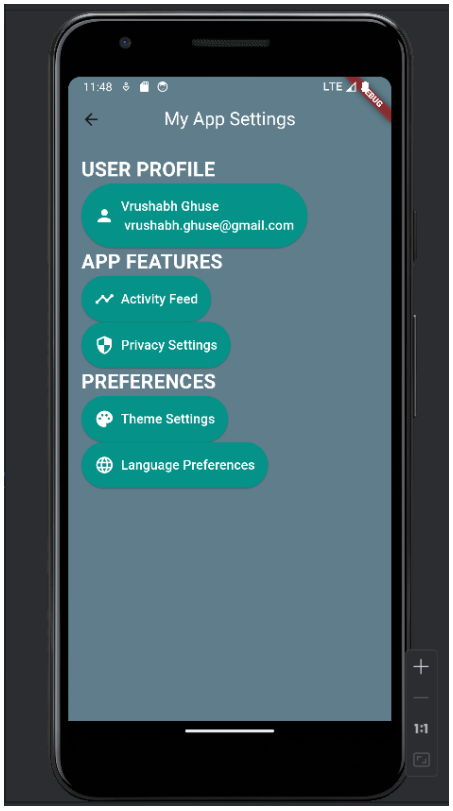
Code:

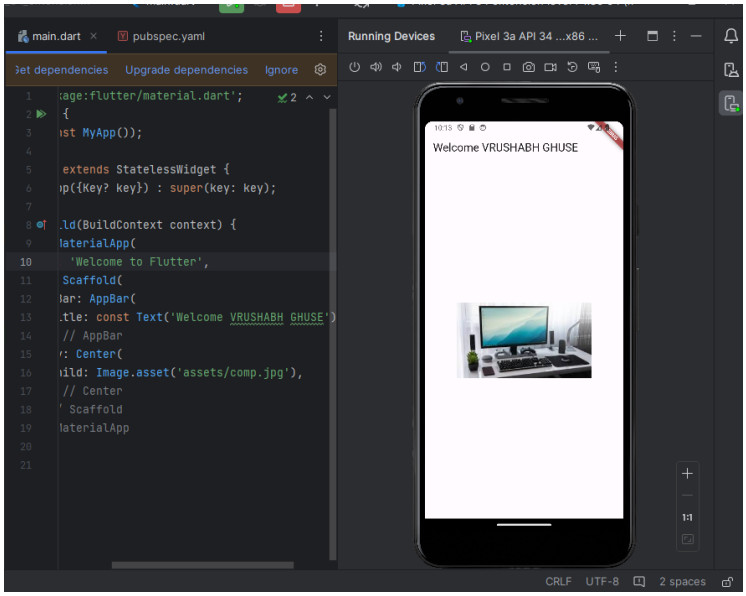


```dart
import 'package:flutter/material.dart';

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

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

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.blueGrey,
      appBar: AppBar(
        centerTitle: true,
        backgroundColor: Colors.blueGrey,
        leading: IconButton(onPressed: () => Navigator.pop(context), icon: Icon(Icons.arrow_back)),
        title: Text("My App Settings", style: TextStyle(color: Colors.white)),
      ),  // AppBar
      body: SafeArea(
        child: Padding(
          padding: EdgeInsets.all(16),
          child: SingleChildScrollView(
```

```dart
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          _buildSectionHeader("USER PROFILE"),
          _buildButton("Vrushabh Ghuse\n vrushabh.ghuse@gmail.com", Icons.person, "Profile"),
          _buildSectionHeader("APP FEATURES"),
          _buildButton("Activity Feed", Icons.timeline, "Activity Feed"),
          _buildButton("Privacy Settings", Icons.security, "Privacy Settings"),
          _buildSectionHeader("PREFERENCES"),
          _buildButton("Theme Settings", Icons.palette, "Theme Settings"),
          _buildButton("Language Preferences", Icons.language, "Language Preferences"),
        ],
      ),  // Column
    ),  // SingleChildScrollView
  ),  // Padding
),  // SafeArea
);  // Scaffold
}

Widget _buildSectionHeader(String title) => Row(children: [Text(title, style: TextStyle(fontSize: 24, fontWeight: Fo

Widget _buildButton(String label, IconData icon, String buttonLabel) {
  return Align(
    alignment: Alignment.topLeft,
    child: ElevatedButton.icon(
      onPressed: () => _handleButtonPress(buttonLabel),
      icon: Icon(icon, color: Colors.white),
      label: Text(label, style: TextStyle(color: Colors.white, fontSize: 16)),
      style: ElevatedButton.styleFrom(backgroundColor: Colors.teal, padding: EdgeInsets.all(16)),
    ),  // ElevatedButton.icon
  );  // Align
}
```

Output:





Conclusion:

Flutter’s widget architecture offers great flexibility for building complex UIs.
Understanding key widgets and concepts is essential for effective Flutter development.