Name: Mayuresh S. Pednekar Class:D15B Roll no: 53

EXPERIMENT NO. 3

Aim: To include icons, images, and fonts in the Flutter app

Theory:

Icons in Flutter:

Overview:
Icons in Flutter are visual symbols used to represent actions, objects, or concepts in the user interface. Flutter provides access to a wide range of icons from the Material Design and Cupertino icon sets.

Common Widgets and Classes:
1. `Icon` Widget:
   - The `Icon` widget is used to display icons in Flutter.
   - Example:
     Icon(Icons.star, color: Colors.yellow, size: 48.0);

2. `IconButton` Widget:
   - The `IconButton` widget is a button that contains an icon.
   - Example:
     IconButton(
       icon: Icon(Icons.add),
       onPressed: () {
         // Add your action here
       },
     );

Images in Flutter:

Overview:
Images are used to display visual content in a Flutter app. Flutter supports various image formats, including JPEG, PNG, GIF, and WebP. You can display images from the network, assets, or the device.

Common Widgets and Classes:
1. Image Widget:
   - The `Image` widget is used to display images in Flutter.
   - Example:
     Image.network('https://example.com/image.jpg');

2. Image.asset` Widget:

- The `Image.asset` widget is used to display images from the assets folder.
- Example:
  Image.asset('assets/image.png');

3. Image.network` Widget:
- The `Image.network` widget loads and displays images from the network.
- Example:
  Image.network('https://example.com/image.jpg');

Fonts in Flutter:

Overview:
Fonts in Flutter are used to define the style and appearance of text. Flutter supports custom fonts, allowing you to use a variety of typefaces and styles in your app.

Common Widgets and Classes:
1. TextStyle` Class:
- The `TextStyle` class is used to define the style of text, including font family, size, color, etc.
- Example:

```
TextStyle(
  fontFamily: 'Roboto',
  fontSize: 16.0,
  color: Colors.black,
);
```

2. Custom Fonts:
- To use custom fonts, you need to include the font files in your project and reference them in the `pubspec.yaml` file.
- Example:

```
fonts:
  - family: MyCustomFont
    fonts:
      - asset: fonts/my_custom_font.ttf
Text(
  'Custom Font Text',
  style: TextStyle(fontFamily: 'MyCustomFont'),
);
```

3. Google Fonts Package:
- The `google_fonts` package provides a convenient way to use Google Fonts in your Flutter app.
- Example:

```
GoogleFonts.openSans(
  textStyle: TextStyle(fontSize: 16.0, color: Colors.black),
```

```
    );
```

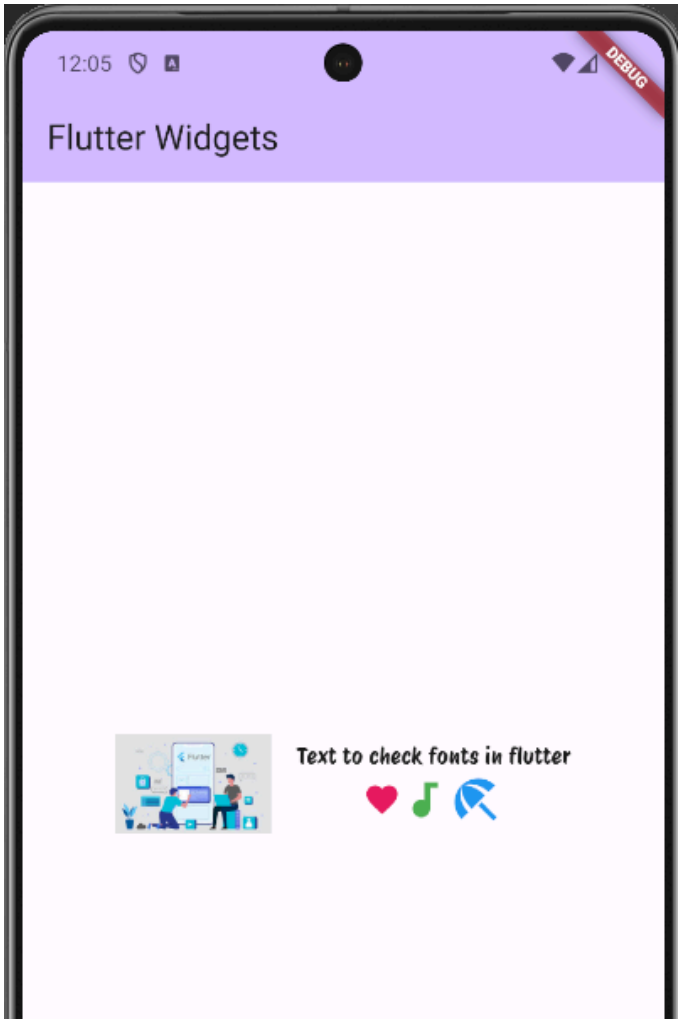
Code:

```
//expt no.3
import 'package:flutter/material.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: 'Flutter Demo',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
        useMaterial3: true,
      ),
      home: const MyHomePage(title: 'Flutter Widgets'),
    );
  }
}

class MyHomePage extends StatefulWidget {
  const MyHomePage({Key? key, required this.title}) : super(key: key);

  final String title;

  @override
  State<MyHomePage> createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        backgroundColor: Theme.of(context).colorScheme.inversePrimary,
        title: Text(widget.title),
      ),
```

```
body: Column(
  mainAxisAlignment: MainAxisAlignment.center,
  children: <Widget>[
    Row(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Container(
          width: 100,
          height: 100,
          child: Image.asset('assets/images/third.jpeg'),
        ),
        const SizedBox(width: 16), // Add spacing between image and text
        const Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: <Widget>[
            Text(
              'Text to check fonts in flutter', style: TextStyle(fontFamily: 'ProtestRiot'),
            ),
            Row(
              mainAxisAlignment: MainAxisAlignment.spaceAround,
              children: <Widget>[
                Icon(
                  Icons.favorite,
                  color: Colors.pink,
                  size: 24.0,
                  semanticLabel: 'Text to announce in accessibility modes',
                ),
                Icon(
                  Icons.audiotrack,
                  color: Colors.green,
                  size: 30.0,
                ),
                Icon(
                  Icons.beach_access,
                  color: Colors.blue,
                  size: 36.0,
                ),
              ],
            ),
          ],
        ),
      ],
    ),
  ],
```

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

Output:



Conclusion:

In conclusion, this experiment provided valuable insights into incorporating icons, images, and custom fonts within a Flutter application. By utilizing Flutter's versatile image and icon widgets, as well as integrating custom font files, we have expanded our understanding of enhancing the visual appeal and functionality of Flutter apps. These elements not only enrich the user experience but also offer greater flexibility and creativity in app development. Moving forward, this knowledge equips us with the necessary tools to create visually engaging and personalized applications in Flutter and add this in our flutter project.