Assignment No.: 03
Title: Exploring Flutter Widgets – Text, Button, Image, and Single Child Widgets
Name: Harshit Raheja
Class: D15B
Roll Number: 45

Aim:
To understand and implement basic Flutter widgets such as Text, Button, Image, and a custom widget using single-child layout.

Theory:
Flutter provides rich pre-built widgets that are essential for building beautiful UIs. The fundamental ones include:

- Text: For displaying static or dynamic strings.
- ElevatedButton, TextButton: For user interaction.
- Image: For showing graphics from assets or networks.
- Container with styling: For custom single-child widget layout.

1. Text Widget Example

```
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(
   home: Scaffold(
    appBar: AppBar(title: const Text("Text Widget Example")),
    body: const Center(
     child: Text(
      'Hello, ALL!',
      textAlign: TextAlign.center,
      style: TextStyle(fontWeight: FontWeight.bold, fontSize: 24),
     ),
    ),
```

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

2. Button Widgets Example

```
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(
   home: Scaffold(
    appBar: AppBar(title: const Text("Button Widget Example")),
    body: Center(
     child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
       ElevatedButton(
        onPressed: () {
         print("Raised Button Clicked");
        },
        child: const Text("Click Me"),
       ),
       TextButton(
        onPressed: () {
         print("Flat Button Clicked");
        },
        child: const Text("Click Here"),
       ),
      ],
     ),
    ),
   ),
  );
```

```
 }
}
```

3. Image Widget Example

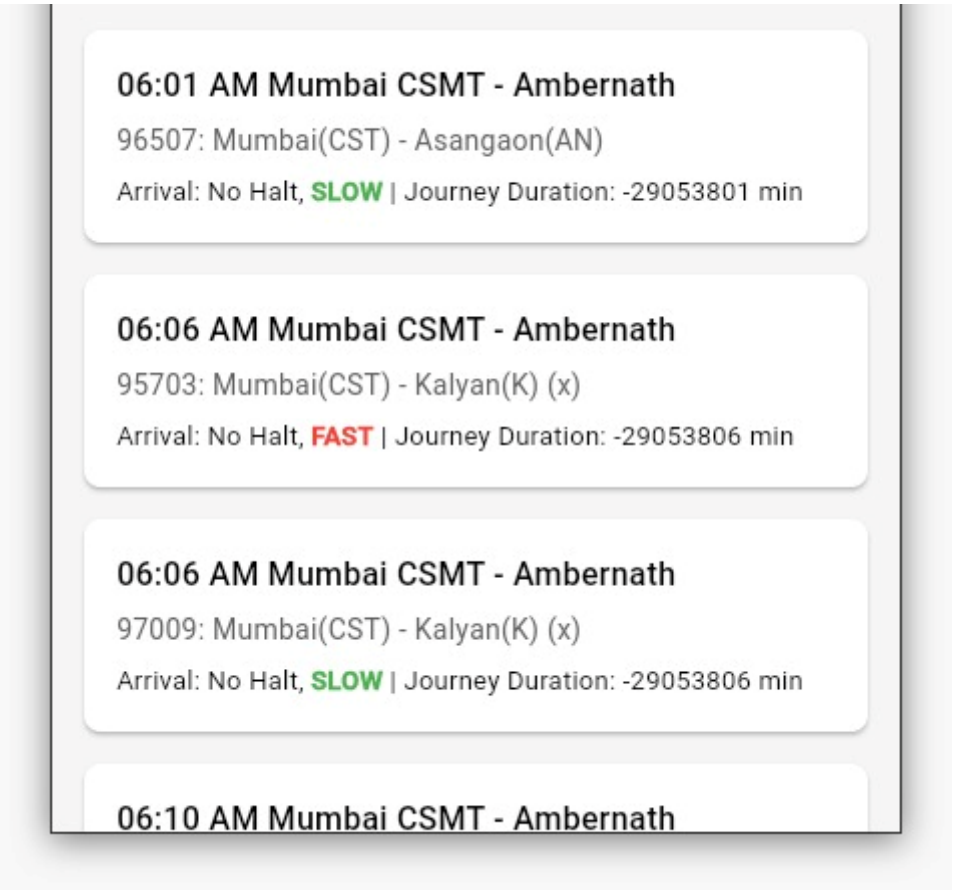
```
import 'package:flutter/material.dart';

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

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

 @override
 Widget build(BuildContext context) {
  return MaterialApp(
   home: Scaffold(
    appBar: AppBar(title: const Text("Image Widget Example")),
    body: Center(
     child: Image.network(
      'https://via.placeholder.com/150',
     ),
    ),
   ),
  );
 }
}
```

4. Single Child Custom Widget Example

```
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(
   home: Scaffold(
```

```
      appBar: AppBar(title: const Text("Single Child Widget Example")),
      body: Center(child: MyButton()),
    ),
   );
 }
}

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

 @override
 Widget build(BuildContext context) {
  return Container(
   decoration: const BoxDecoration(
    border: Border(
     top: BorderSide(width: 1.0, color: Color(0xFFFFFFFFF)),
     left: BorderSide(width: 1.0, color: Color(0xFFFFFFFFF)),
     right: BorderSide(width: 1.0, color: Color(0xFFFF000000)),
     bottom: BorderSide(width: 1.0, color: Color(0xFFFF000000)),
    ),
   ),
   child: Container(
    padding: const EdgeInsets.symmetric(horizontal: 20.0, vertical: 2.0),
    decoration: const BoxDecoration(
     border: Border(
      top: BorderSide(width: 1.0, color: Color(0xFFFFDFDFDF)),
      left: BorderSide(width: 1.0, color: Color(0xFFFFDFDFDF)),
      right: BorderSide(width: 1.0, color: Color(0xFFFF7F7F7F)),
      bottom: BorderSide(width: 1.0, color: Color(0xFFFF7F7F7F)),
     ),
     color: Colors.grey,
    ),
    child: const Text(
     'OK',
     textAlign: TextAlign.center,
     style: TextStyle(color: Colors.black),
    ),
   ),
  );
 }
}
```

Output:



Conclusion:

This assignment demonstrates practical usage of Flutter's basic UI widgets. The Text, Button, and Image widgets offer essential building blocks, while the Container wrapped custom widget showcases single-child layout customization.