Experiment:2

Aim: To design Flutter UI by including common widgets.

Theory:

- MaterialApp
  Purpose: Serves as the root widget of your app when you want to use Material Design components. It wraps several widgets that are commonly required for material design applications, providing functionality such as navigation, themes, and so on.
  Usage in code: It is used at the top level of the app to ensure that Material Design is applied.

- Scaffold
  Purpose: Offers a framework for Material Design layouts. It provides default app bars, body text, and a structure for adding drawers, snack bars, and bottom sheets.
  Usage in code: It's used here to create a basic page layout with an AppBar and a body.
  AppBar
  Purpose: Displays a Material Design app bar at the top of the app. The app bar typically displays the app's title, and can also contain buttons for actions.
  Usage in code: Here, it's used to display the title 'Flutter Assets Example' at the top of the screen.

- Center
  Purpose: Centers its child within itself. This widget is often used to align widgets horizontally and vertically.
  Usage in code: It's used to center the Column widget on the screen.

- Column
  Purpose: Arranges its children in a vertical array. It's one of the most common layout widgets.
  Usage in code: It's used to layout its children (Text, SizedBox, Image.asset, another SizedBox, and Row) vertically in the center of the screen.

- Text
  Purpose: Displays a string of text with single style. It's a fundamental widget for displaying text in apps.

Usage in code: Used to display the 'Hello, Flutter!' message with a custom font and font size.

- TextStyle
  Purpose: Defines the style of text to be displayed. It includes properties like font size, color, font weight, etc.
  Usage in code: Here, it specifies the font family and font size for the 'Hello, Flutter!' text.

- SizedBox
  Purpose: A box with a specified size. It can be used to create space between widgets or to size a widget.
  Usage in code: Used twice to create vertical space of 20 pixels between the widgets in the column.

Code:

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

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: const Text('Experiment 1'),
        ),
        body: Center(
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: <Widget>[
              const Text(
                'Hello, Kaushik!',
                style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),
              ),
              Container(
```

```dart
          margin: const EdgeInsets.all(20),
          padding: const EdgeInsets.all(10),
          color: Colors.amber,
          child: const Text('Please enter your Message'),
        ),
        const Padding(
          padding: EdgeInsets.all(8.0),
          child: TextField(
            decoration: InputDecoration(
              border: OutlineInputBorder(),
              labelText: 'Input Field',
            ),
          ),
        ),
        Row(
          mainAxisAlignment: MainAxisAlignment.center,
          children: <Widget>[
            ElevatedButton(
              onPressed: () {},
              child: const Text('Submit'),
            ),
            const SizedBox(width: 10),
          ],
        ),
      ],
    ),
  ),
);
}
}
```
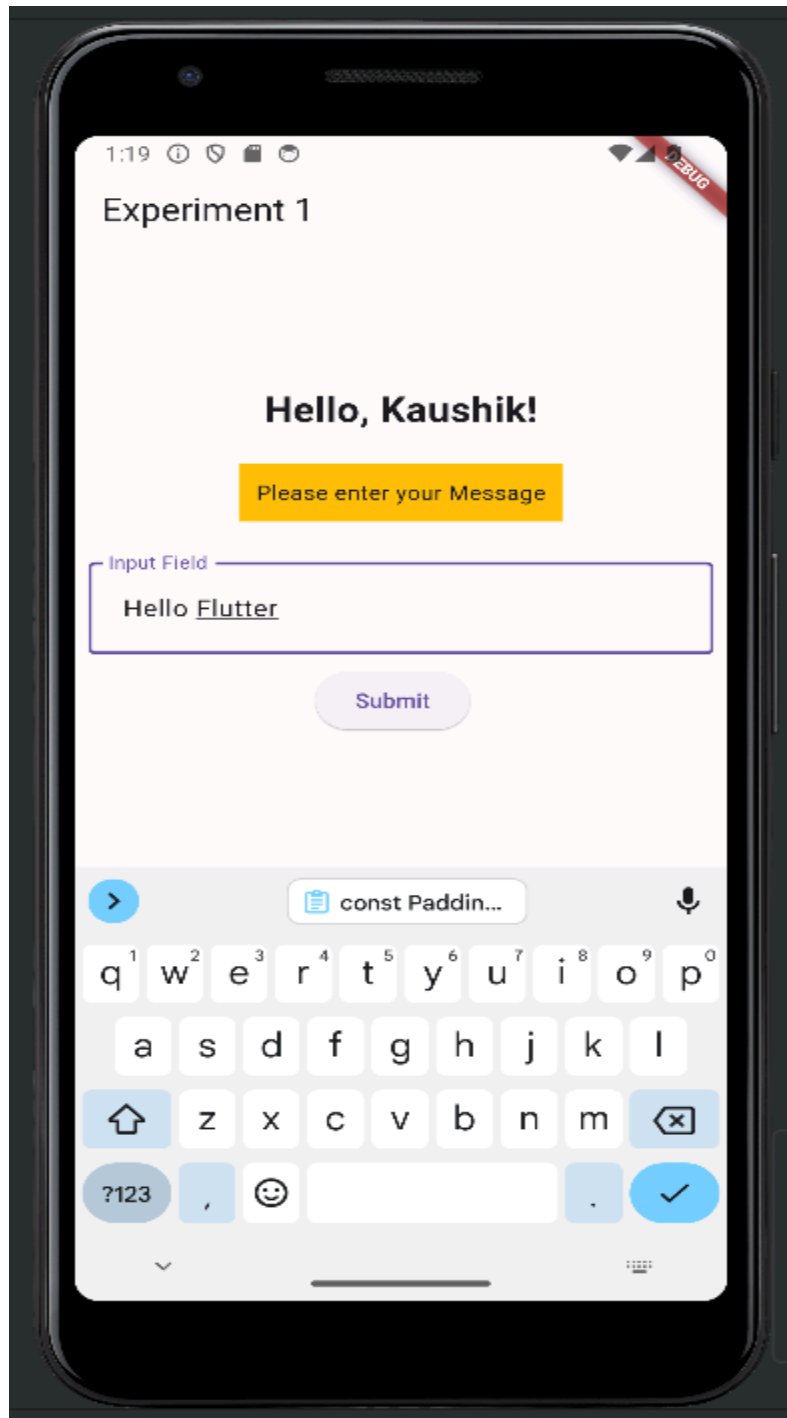
Conclusion: This experiment introduces you to basic Flutter UI design using common widgets. Flutter's widget-based architecture allows for building complex UIs with reusable components. Experiment with different widgets and properties to further enhance your UI designs.