Aim: To design Flutter UI by including common widgets.
Theory:
Flutter applications are crafted using widgets, which serve as the fundamental building blocks defining the layout, appearance, and behavior of the app's user interface (UI). These widgets range from basic elements like buttons and text fields to intricate layouts and animations. Flutter offers a diverse range of built-in widgets enabling developers to create highly personalized and interactive interfaces.

Widgets in Flutter fall into two primary categories:

Stateless Widgets: These widgets are immutable and lack internal state. They are suitable for UI components that remain static over time. Stateless widgets are created by extending the StatelessWidget class and defining the UI within the build method. Examples include text, buttons, icons, and images.

Stateful Widgets: Unlike stateless widgets, stateful widgets manage mutable state that can evolve over time in response to user actions, data updates, or other events. These widgets are created by extending the StatefulWidget class, which comprises both the widget itself and an associated state class, typically suffixed with State. The state class handles the widget's mutable state and triggers widget rebuilding upon state changes.

Here's a selection of commonly used Flutter widgets:

- Text: Displays text strings.
- Image: Renders images.
- Container: Offers versatility for layout and styling.
- Row: Organizes child widgets horizontally.
- Column: Organizes child widgets vertically.
- Stack: Layers child widgets.
- ListView: Presents a scrollable list of widgets.
- GridView: Exhibits a scrollable grid of widgets.
- AppBar: Implements a material design app bar.
- Scaffold: Establishes the basic material design layout structure.
- TextField: Allows user input of text.
- FlatButton, RaisedButton, IconButton: Variants of buttons.
- AlertDialog, BottomSheet: Dialog and modal widgets.
- GestureDetector: Detects gestures like taps, drags, and swipes.

- Opacity: Renders its child with partial transparency.
- AnimatedContainer, AnimatedOpacity, AnimatedBuilder: Facilitates animation creation.

These examples represent just a fraction of the available Flutter widgets. Developers can also craft custom widgets by composing existing ones or by subclassing StatelessWidget or StatefulWidget.

Code:

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

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

class PomodoroApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Pomodoro Timer',
      theme: ThemeData(primarySwatch: Colors.blue),
      home: PomodoroTimer(),
    );
  }
}

class PomodoroTimer extends StatefulWidget {
  @override
  _PomodoroTimerState createState() => _PomodoroTimerState();
}

class _PomodoroTimerState extends State<PomodoroTimer> {
  int _minutes = 25;
  int _seconds = 0;
  late Timer _timer;
  bool _isRunning = false;

  void _startTimer() {
    setState(() {
      _isRunning = true;
    });
    _timer = Timer.periodic(Duration(seconds: 1), (timer) {
      if (_minutes == 0 && _seconds == 0) {
        _resetTimer();
        return;
      }
      if (_seconds == 0) {
        setState(() {
          _minutes--;
          _seconds = 59;
        });
      } else {
        setState(() {
          _seconds--;
        });
      }
    });
  }

  void _stopTimer() {
    _timer.cancel();
    setState(() {
      _isRunning = false;
    });
  }

  void _resetTimer() {
    _timer.cancel();
    setState(() {
      _isRunning = false;
      _minutes = 25;
      _seconds = 0;
    });
  }

  String _formatTime(int time) {
    return time < 10 ? '0$time' : '$time';
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Pomodoro Timer'),
      ),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: <Widget>[
            Text(

'${_formatTime(_minutes)}:${_formatTime(_seconds)}',
              style: TextStyle(fontSize: 60),
            ),
            SizedBox(height: 20),
            Row(
              mainAxisAlignment: MainAxisAlignment.spaceEvenly,
              children: <Widget>[
                ElevatedButton(
                  onPressed: _isRunning ? null : _startTimer,
                  child: Text('Start'),
                ),
                ElevatedButton(
                  onPressed: _isRunning ? _stopTimer : null,
                  child: Text('Stop'),
                ),
                ElevatedButton(
                  onPressed: _resetTimer,
                  child: Text('Reset'),
                ),
              ],
            ),
          ],
        ),
      ),
    );
  }
}
```

These are the widgets used in this code:
MaterialApp: This widget sets up the MaterialApp for the app, providing the title and theme.

Scaffold: This widget provides the basic layout structure for the app, including the app bar.

AppBar: This widget creates the app bar at the top of the screen, displaying the title "Pomodoro Timer".

Center: This widget centers its child widget vertically and horizontally within its parent.

Column: This widget lays out its children in a vertical column.

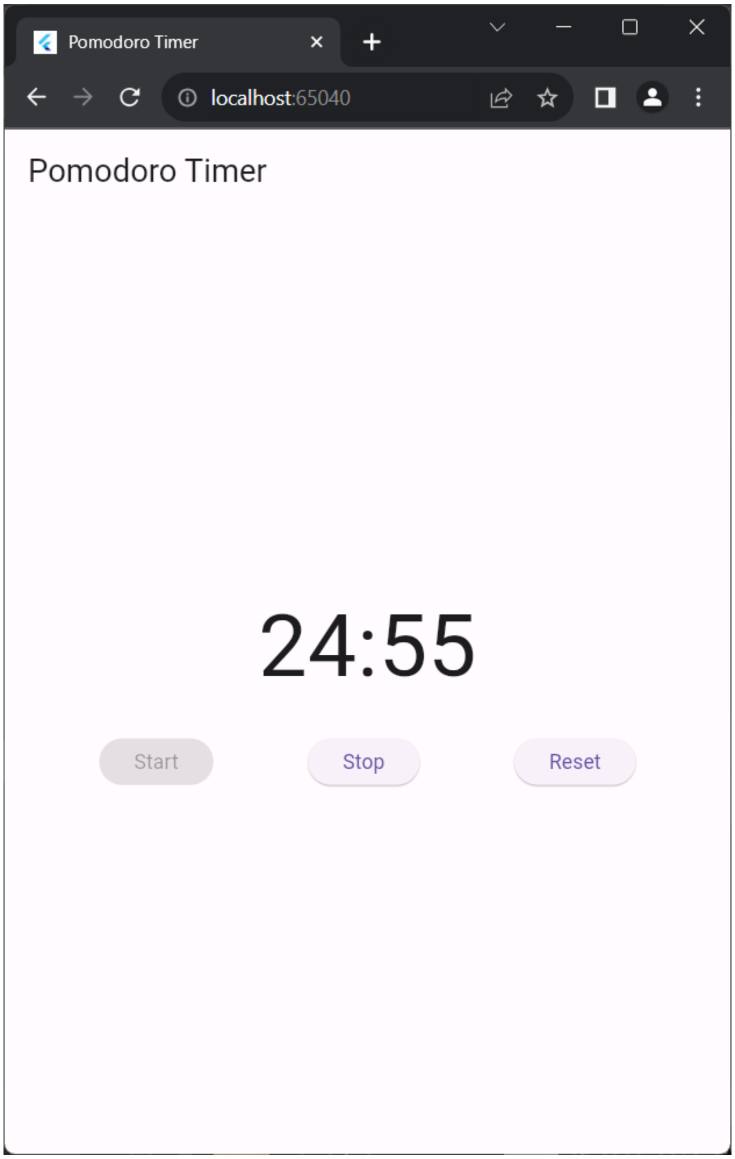
Text: This widget displays text on the screen, showing the current time remaining in the Pomodoro session.

SizedBox: This widget provides a fixed-size box to create space between widgets.

Row: This widget lays out its children in a horizontal row.

ElevatedButton: This widget creates a button with a raised appearance, allowing users to start, stop, and reset the timer.

Output:



Conclusion: Thereby we have implemented widgets in flutter application. These are the widgets that we have used in making of this app, MaterialApp, Scaffold, Text, Column, Center etc.