Name: Soham H. Nimbalkar EXPT No: 02 Roll No: 49

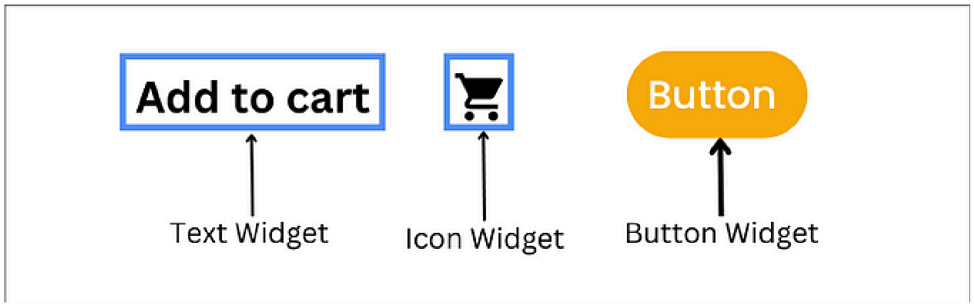
Aim: To design Flutter UI by including common widgets.

Theory:

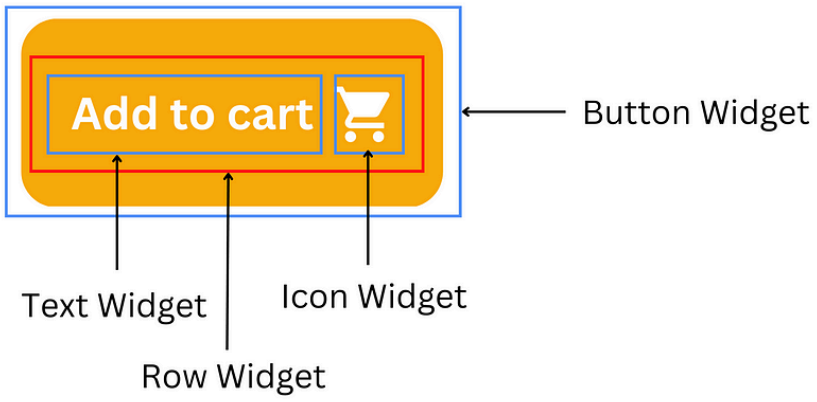
Flutter is a modern, open-source framework for developing cross-platform mobile applications. It was developed by Google and has become increasingly popular among developers due to its fast development cycle, easy-to-learn architecture, and fast-performing apps. One of the key features of Flutter is its use of widgets, which make up the building blocks of Flutter apps. **In this blog post, we will take a closer look at Flutter widgets and the architecture of Flutter apps.**

Widgets:

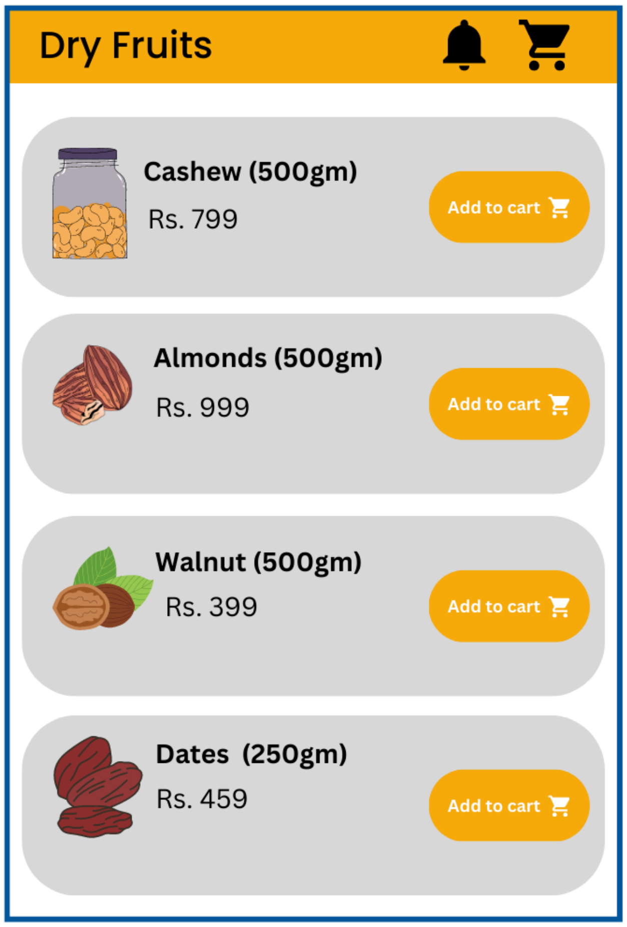
A Flutter widget is a basic unit of the user interface (UI) in a Flutter app. It can be as simple as a text box or as complex as a custom navigation drawer. The beauty of Flutter widgets is that they can be combined and nested to create a rich, dynamic UI. Flutter provides a wide range of built-in widgets, such as text boxes, buttons, and images, that can be used out-of-the-box, or customised to suit your needs.



As We can see in above example, Text, Icon and Button all are widgets which are the inbuilt widgets provided by Flutter.



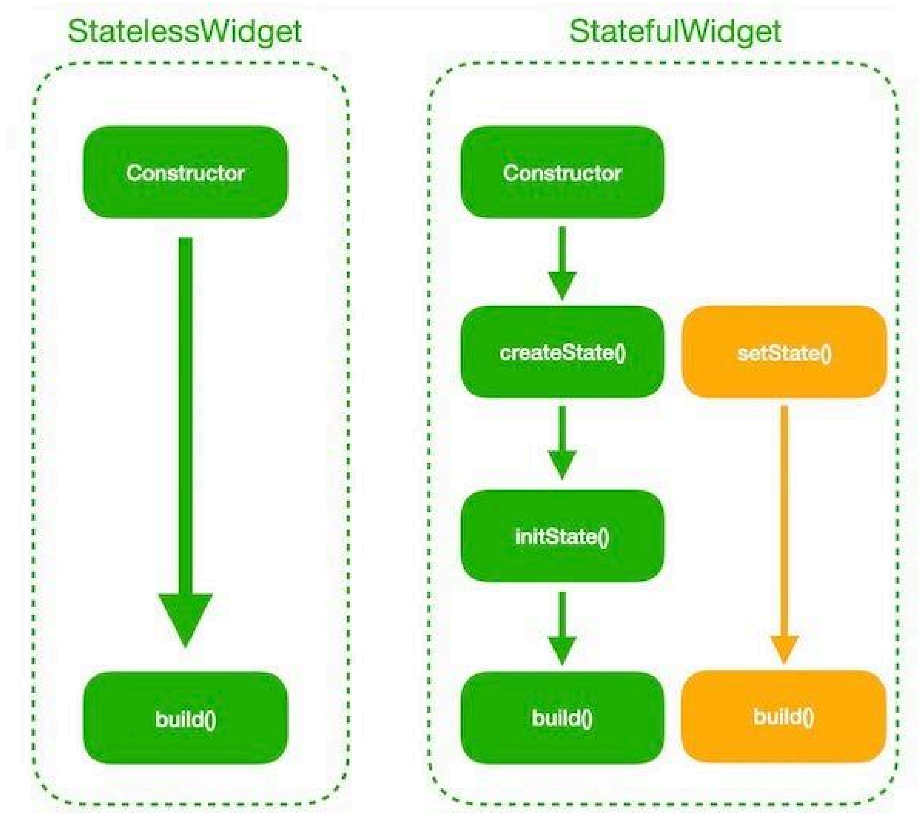
In Above example, we can see that Text, Icon, Row and Button widgets are combined / Nested to make a rich Custom Button. This is the beauty of flutter.



In Above Example, we can see that, Custom button made earlier has been used along with Image, Text, Rows, Columns and Cards to make a Product Tile, and multiple Product tiles have been used to make the entire Product Screen. **This is the best example to prove that Widgets are the building blocks of UI in Flutter.**
Flutter widgets are based on the concept of composition. This means that you can create a new widget by combining several smaller widgets. This allows you to create reusable UI elements that can be used across multiple screens in your app.

Stateless and stateful widgets in Flutter:

Flutter has majorly two types of widgets viz — Stateless and Stateful widgets.
Stateless widgets do not contain states hence they can be updated only when its parent changes. Whereas stateful widgets can hold the states internally, so it can be updated whenever its states changes and also whenever its parent changes.



In above example we can see

1. In case of Stateless widget — build function (Which is responsible to build the widget) is triggered only when the constructor is called.
2. In case of Stateful widget — build function is triggered when constructor is called and also when initState(), setState() and called.

We need to choose which widget to be used depending on requirements. I personally go for Stateful widget when dynamic widget is needed and stateless widget when static widget is needed. It is a good practice to limit the uses of stateful widgets wherever possible as it results into building entire widget instead of the desired part of widget.

Code:

Main.dart:

```
import 'package:flutter/material.dart';
import 'package:weather_app/pages/home_page.dart';

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

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  // This widget is the root of your application.
  @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 Demo Home Page'),
    );
  }
}

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

  final String title;

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

class _MyHomePageState extends State<MyHomePage> {
  @override
  Widget build(BuildContext context) {
    return HomePage();
  }
}
```

home_page.dart:

```
import 'package:flutter/material.dart';
import 'package:intl/intl.dart';
import 'package:weather/weather.dart';
import 'package:weather_app/consts.dart';

class HomePage extends StatefulWidget {
  const HomePage({Key? key}) : super(key: key);

  @override
  State<HomePage> createState() => _HomePageState();
}

class _HomePageState extends State<HomePage> {
  final WeatherFactory _wf = WeatherFactory(OPENWEATHER_API_KEY);

  Weather? _weather;
```

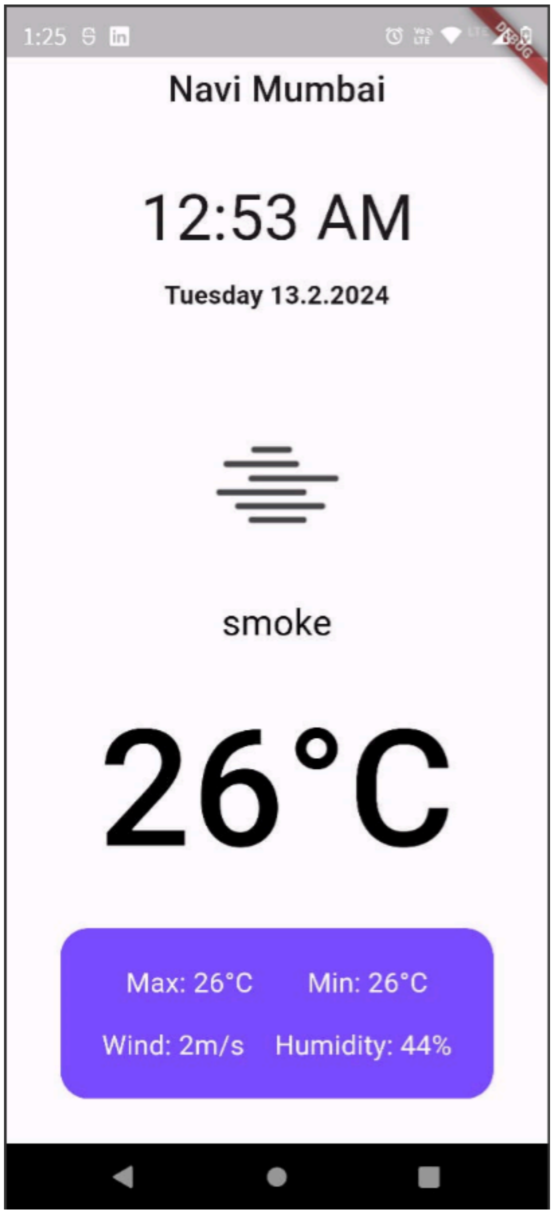
```
TextEditingController _locationController = TextEditingController();

@override
void initState() {
  super.initState();
  _getLocationWeather("Mumbai"); // Default location, you can change this
}

void _getLocationWeather(String location) {
  _wf.currentWeatherByCityName(location).then((w) {
    setState(() {
      _weather = w;
    });
  });
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    body: _buildUI(),
  );
}

Widget _buildUI() {
  if (_weather == null) {
    return const Center(
      child: CircularProgressIndicator(),
    );
  }

  return SingleChildScrollView(
    child: Column(
      mainAxisSize: MainAxisSize.max,
      mainAxisAlignment: MainAxisAlignment.center,
      crossAxisAlignment: CrossAxisAlignment.center,
      children: [
        SizedBox(
          height: MediaQuery.of(context).size.height * 0.05,
        ),
        _locationHeader(),
        SizedBox(
          height: MediaQuery.of(context).size.height * 0.05,
        ),
        _dateTimeInfo(),
        SizedBox(
          height: MediaQuery.of(context).size.height * 0.05,
        ),
        _weatherIcon(),
        SizedBox(
          height: MediaQuery.of(context).size.height * 0.02,
        ),
        _currentTemp(),
        SizedBox(
          height: MediaQuery.of(context).size.height * 0.02,
        ),
        _extraInfo(),
        SizedBox(height: 20),
        _locationInput(),
      ],
    ),
  );
}

Widget _locationHeader() {
  return Text(
    _weather?.areaName ?? "",
    style: const TextStyle(
      fontSize: 20,
      fontWeight: FontWeight.w500,
    ),
  );
}

Widget _dateTimeInfo() {
  DateTime now = _weather!.date!;
  return Column(
    children: [
      Text(
        DateFormat("h:mm a").format(now),
        style: const TextStyle(fontSize: 35),
      ),
      const SizedBox(
        height: 10,
      ),
      Row(
        mainAxisSize: MainAxisSize.max,
        mainAxisAlignment: MainAxisAlignment.center,
        crossAxisAlignment: CrossAxisAlignment.center,
        children: [
          Text(
            DateFormat("EEEE").format(now),
            style: const TextStyle(fontWeight: FontWeight.w700),
          ),
          Text(
            " ${DateFormat("d.M.y").format(now)}",
            style: const TextStyle(fontWeight: FontWeight.w700),
          )
        ],
      )
    ],
  );
}

Widget _weatherIcon() {
  return Column(
    mainAxisSize: MainAxisSize.min,
    mainAxisAlignment: MainAxisAlignment.center,
    crossAxisAlignment: CrossAxisAlignment.center,
    children: [
      Container(
        height: MediaQuery.of(context).size.height * 0.20,
        decoration: BoxDecoration(
          image: DecorationImage(
            image: NetworkImage(

"https://openweathermap.org/img/wn/${_weather?.weatherIcon}@4x.png",
            ),
          ),
        ),
      ),
```

```dart
      Text(
        _weather?.weatherDescription ?? "",
        style: const TextStyle(
          color: Colors.black,
          fontSize: 20,
        ),
      )
    ],
  );
}

Widget _currentTemp() {
  return Text(
    "${_weather?.temperature?.celsius?.toStringAsFixed(0)}°C",
    style: const TextStyle(
      color: Colors.black,
      fontSize: 90,
      fontWeight: FontWeight.w500,
    ),
  );
}

Widget _extraInfo() {
  return Container(
    height: MediaQuery.of(context).size.height * 0.15,
    width: MediaQuery.of(context).size.width * 0.80,
    decoration: BoxDecoration(
      color: Colors.deepPurpleAccent,
      borderRadius: BorderRadius.circular(20),
    ),
    padding: const EdgeInsets.all(8.0),
    child: Column(
      mainAxisAlignment: MainAxisAlignment.spaceEvenly,
      crossAxisAlignment: CrossAxisAlignment.center,
      children: [
        Row(
          mainAxisAlignment: MainAxisAlignment.spaceEvenly,
          mainAxisSize: MainAxisSize.max,
          crossAxisAlignment: CrossAxisAlignment.center,
          children: [
            Text(
              "Max: ${_weather?.tempMax?.celsius?.toStringAsFixed(0)}°C",
              style: const TextStyle(
                color: Colors.white,
                fontSize: 15,
              ),
            ),
            Text(
              "Min: ${_weather?.tempMin?.celsius?.toStringAsFixed(0)}°C",
              style: const TextStyle(
                color: Colors.white,
                fontSize: 15,
              ),
            ),
          ],
        ),
        Row(
          mainAxisAlignment: MainAxisAlignment.spaceEvenly,
          mainAxisSize: MainAxisSize.max,
          crossAxisAlignment: CrossAxisAlignment.center,
          children: [
            Text(
              "Wind: ${_weather?.windSpeed?.toStringAsFixed(0)}m/s",
              style: const TextStyle(
                color: Colors.white,
                fontSize: 15,
              ),
            ),
            Text(
              "Humidity: ${_weather?.humidity?.toStringAsFixed(0)}%",
              style: const TextStyle(
                color: Colors.white,
                fontSize: 15,
              ),
            ),
          ],
        )
      ],
    ),
  );
```

```
  }

  Widget _locationInput() {
    return Padding(
      padding: const EdgeInsets.all(16.0),
      child: TextField(
        controller: _locationController,
        decoration: InputDecoration(
          labelText: 'Enter Location',
          suffixIcon: IconButton(
            icon: Icon(Icons.search),
            onPressed: () {
              _getLocationWeather(_locationController.text);
            },
          ),
        ),
      ),
    );
  }
}
```

Name: Soham H. Nimbalkar EXPT No: 02 Roll No: 49

Output:



1:25

Navi Mumbai

12:53 AM

Tuesday 13.2.2024

smoke

26°C

Max: 26°C Min: 26°C

Wind: 2m/s Humidity: 44%