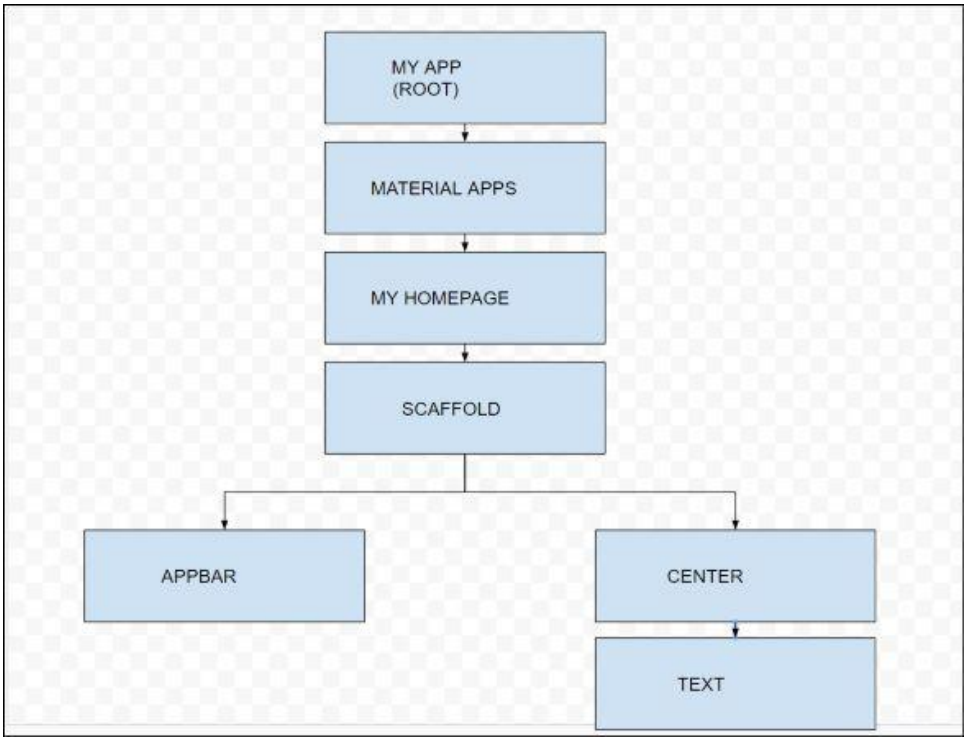
Types of Widgets

Types of Widgets:

There are broadly two types of widgets in the flutter:

1. Stateless Widget
2. Stateful Widget



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.

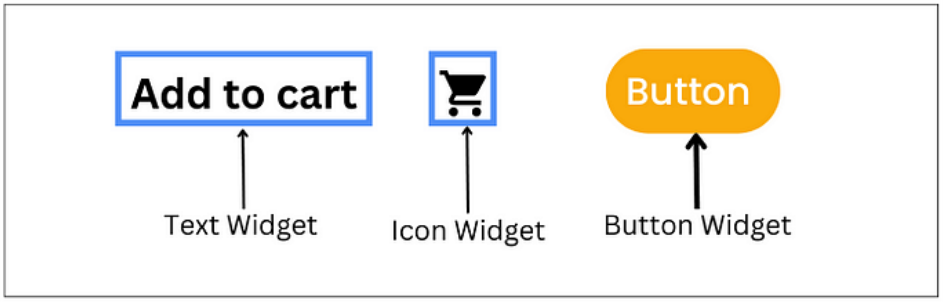


Fig 1: Individual inbuilt Widgets

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

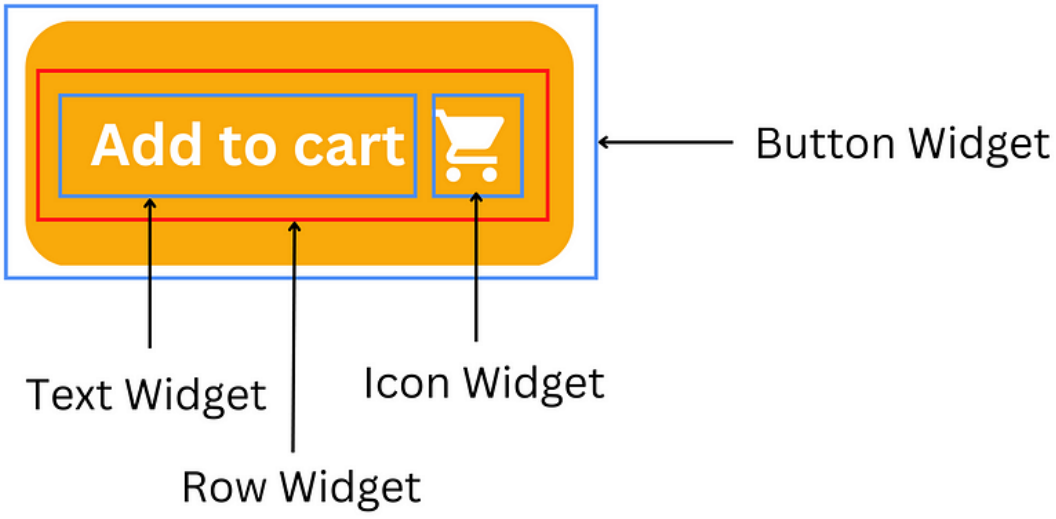


Fig 2: Custom Button

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.

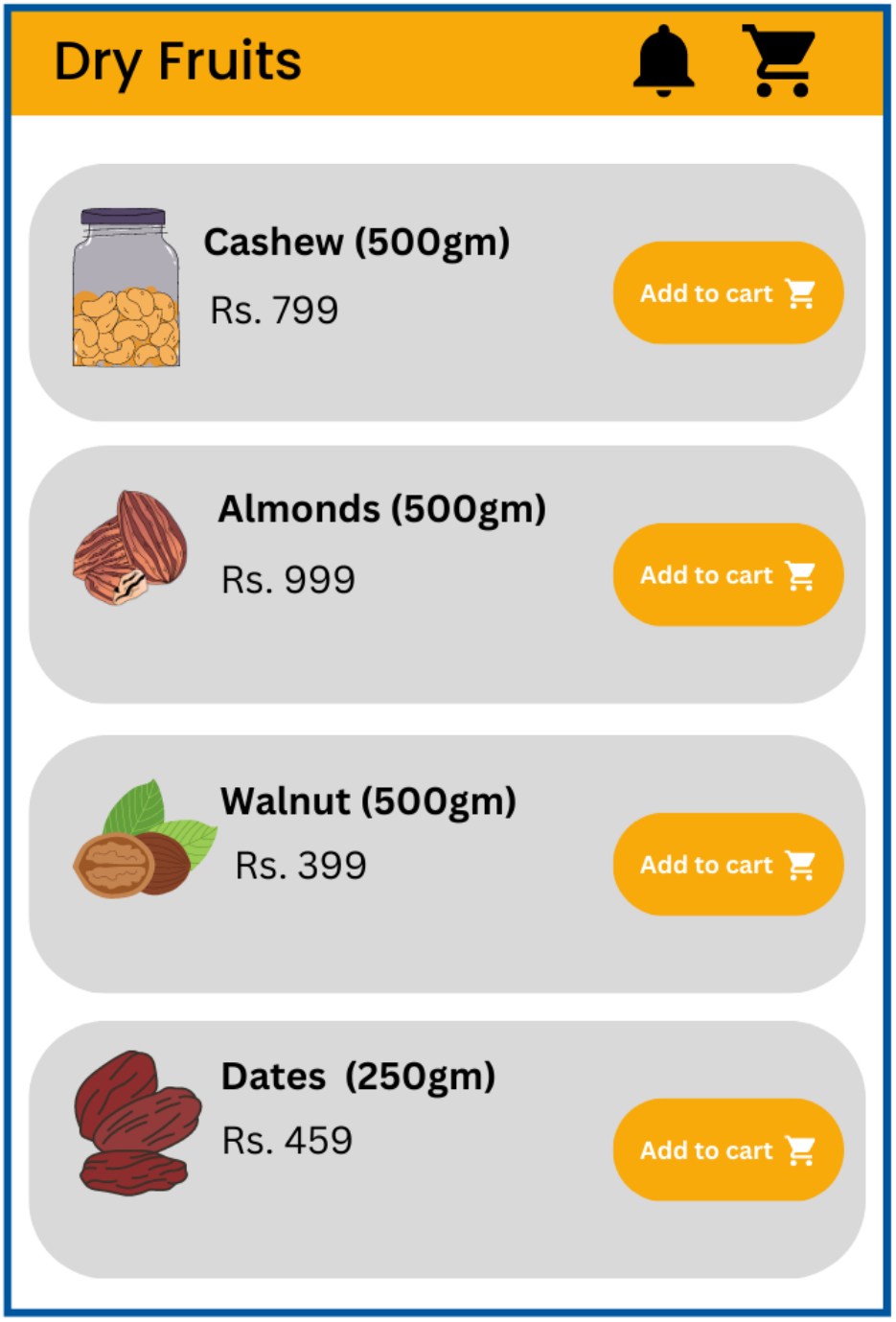


Fig 3: Products Screen

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.

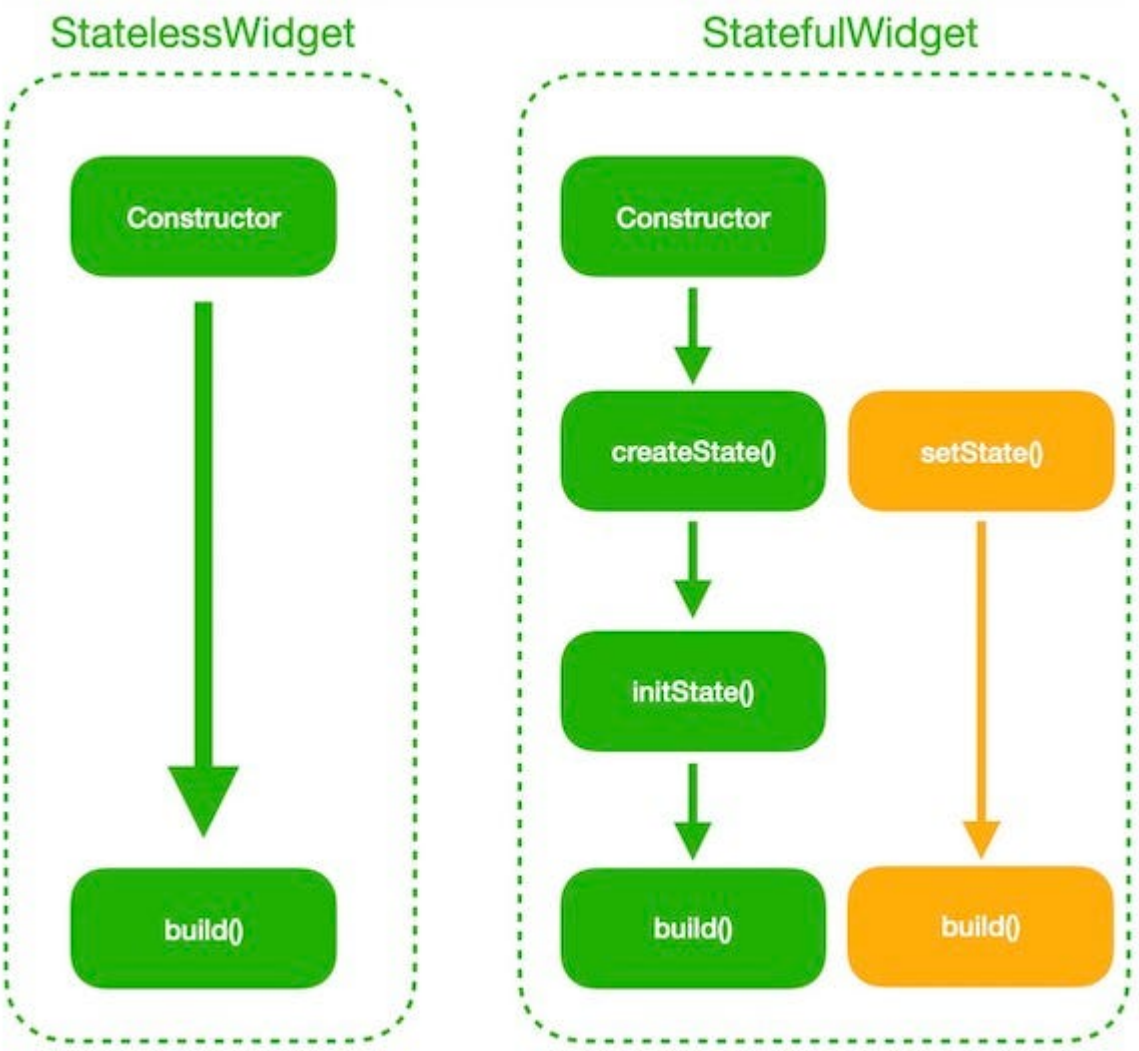


Fig 4: Stateless and Stateless Widget

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.

Architecture:

The architecture of a Flutter app is based on the Model-View-Controller (MVC) design pattern. In MVC, the model represents the data that your app is working with, the view represents the UI of the app, and the controller manages the interaction between the model and the view. In Flutter, the model and the controller are combined into a single entity called the “StatefulWidget”. The StatefulWidget is responsible for both managing the data and controlling the interaction with the view.

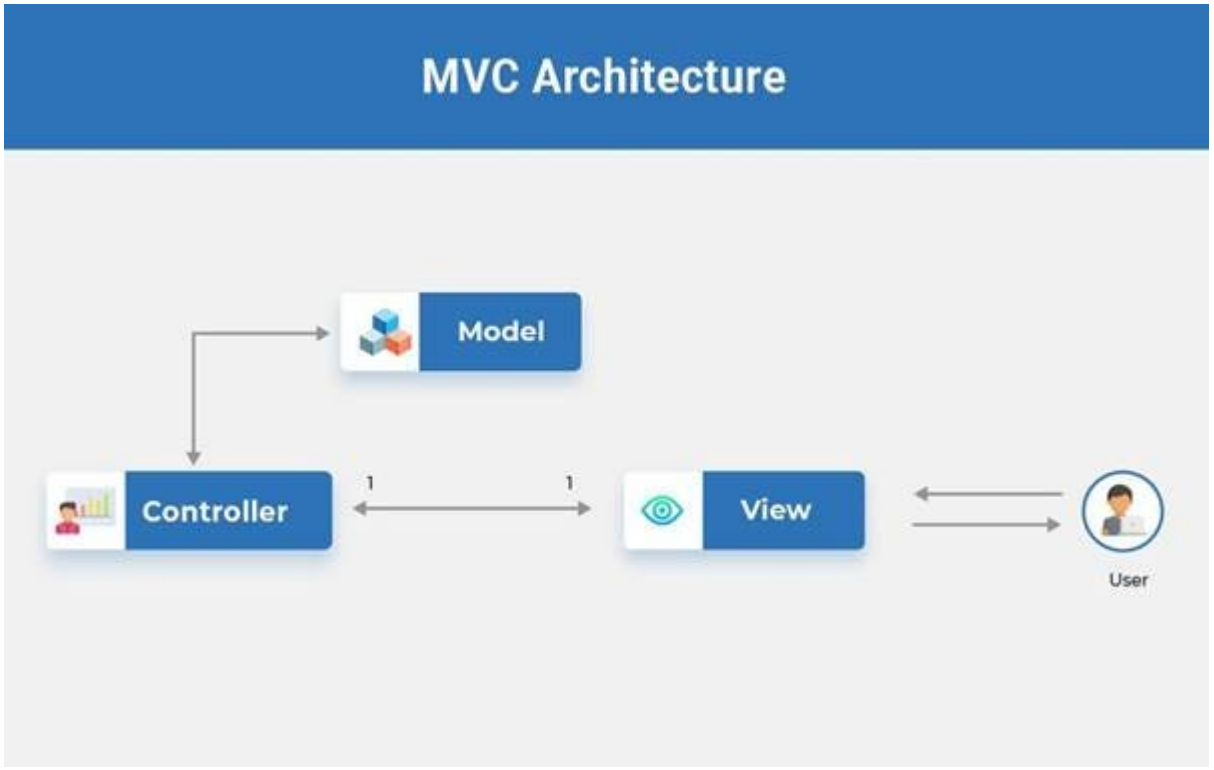


Fig 5: MVC Architecture

The view in a Flutter app is represented by the widgets. The widgets are built using the build method of the StatefulWidget. This method returns a widget tree that defines the structure of the UI. The widgets in the widget tree can be combined and nested to create a rich, dynamic UI.

In conclusion, Flutter’s use of widgets and its architecture based on the MVC design pattern make it an excellent choice for developing fast, responsive mobile apps. The wide range of built-in widgets and the ability to create custom widgets make it easy to create rich, dynamic UIs, and the reactive programming model makes it easy to create fast, responsive apps that are always up-to-date. If you’re looking to build a mobile app, consider using Flutter and start exploring the world of widgets and architecture today!