ATHARVA PATIL(39)
DIV: D15C BATCH: B

EXPERIMENT 02

Aim: To design Flutter UI by including common widgets.

Theory:

Flutter is an open-source UI toolkit by Google used to create natively compiled applications for mobile, web, and desktop from a single codebase. It uses the **Dart** programming language and follows a declarative UI approach, making it easy to build beautiful and highly customizable user interfaces.

Flutter UI Hierarchy

1. **MaterialApp**: The root of the Flutter application (for Material Design apps).
2. **Scaffold**: Provides the basic structure, including AppBar, Body, FloatingActionButton, Bottom Navigation, etc.
3. **Widgets**: The building blocks of the UI.

Types of Widgets

Widgets in Flutter are categorized into two types:

1. Stateless Widgets

- Immutable (does not change once created).
- Used when the UI does not need to update dynamically.

2. Stateful Widgets

- Can change dynamically during runtime.
- Useful for handling user interactions, animations, and real-time updates.

ATHARVA PATIL(39)
DIV: D15C BATCH: B

Syntax:

STATELESS WIDGET:

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: CounterScreen(),
    );
  }
}
```

STATEFUL WIDGET:

```
class CounterScreen extends StatefulWidget {
  @override
  _CounterScreenState createState() => _CounterScreenState();
}

class _CounterScreenState extends State<CounterScreen> {
  int _counter = 0;

  void _incrementCounter() {
    setState(() {
      _counter++;
    });
  }
```

ATHARVA PATIL(39)
DIV: D15C BATCH: B

Widget and properties:

Common widgets

1. Column: The Column widget arranges children vertically in a single column.

Properties of Column

- children → List of widgets to be placed in the column.
- mainAxisAlignment → Aligns children along the vertical axis. (MainAxisAlignment.start, center, end, spaceAround, spaceBetween, spaceEvenly)
- crossAxisAlignment → Aligns children along the horizontal axis. (CrossAxisAlignment.start, center, end, stretch, baseline)
- mainAxisSize → Controls how much space the column should take (MainAxisSize.min or max).
- verticalDirection → Defines the direction in which children are placed (VerticalDirection.down or up).
- textBaseline → Aligns text widgets based on the baseline.

2. Row: The Row widget arranges children horizontally in a single row.

Properties of Row

- children → List of widgets to be placed in the row.
- mainAxisAlignment → Aligns children along the horizontal axis.
- crossAxisAlignment → Aligns children along the vertical axis.
- mainAxisSize → Determines how much space the row should take.
- textBaseline → Aligns text widgets based on the baseline.

3. Stack: The Stack widget places widgets on top of each other in a layered fashion.

Properties of Stack

- children → List of widgets placed in the stack.
- alignment → Aligns children inside the Stack (Alignment.center, topLeft, bottomRight, etc.).
- fit → Controls the size of the non-positioned children (StackFit.loose, StackFit.expand).
- clipBehavior → Defines how content is clipped inside the Stack.

4. Container: The Container widget is a flexible box used to hold and style other widgets.

Properties of Container

- child → The widget inside the container.
- width & height → Dimensions of the container.
- color → Background color of the container.
- alignment → Aligns child inside the container (Alignment.center, topLeft, etc.).
- padding → Space inside the container around its child.
- margin → Space outside the container.
- decoration → Allows adding borders, shadows, gradients, etc.

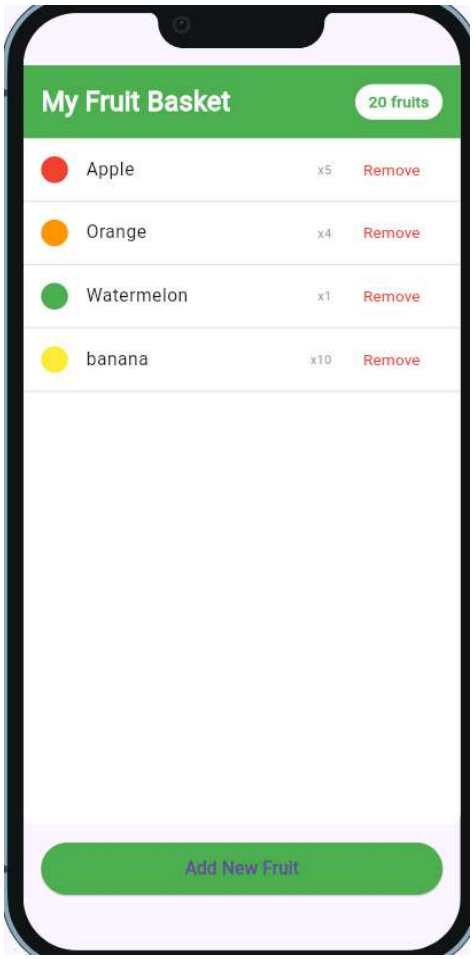
5. List View: The ListView widget displays a scrollable list of items.

Properties of ListView

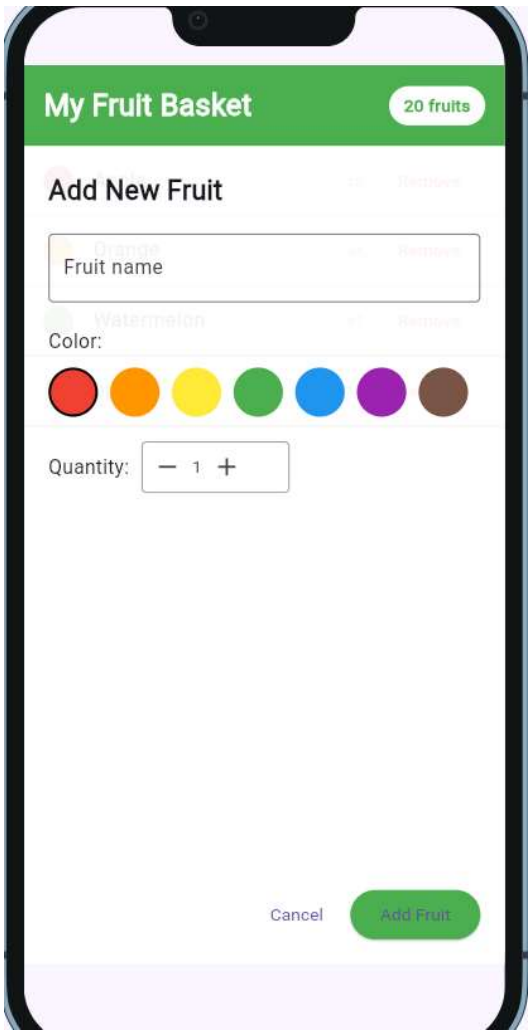
ATHARVA PATIL(39)
DIV: D15C BATCH: B

- children → List of widgets to display in the list (for ListView(children: [...])).
- scrollDirection → Defines the scrolling direction (Axis.vertical or Axis.horizontal).
- shrinkWrap → Adjusts the list size based on children (true or false).
- physics → Defines scrolling behavior (BouncingScrollPhysics, NeverScrollableScrollPhysics).
- padding → Adds padding around the list.
- separatorBuilder → Adds dividers between list items (ListView.separated).
- builder → Dynamically creates items (ListView.builder).

Output:



ATHARVA PATIL(39)
DIV: D15C BATCH: B



Conclusion: Hence we have successfully designed a flutter UI for creating a fruit basket along with adding and removing fruits and also included common widgets like rows, columns, containers, stack and Lists.