MOHIT R MANDHYANI **D15B-34**

MPL LAB EXPERIMENT-2

AIM:- To design Flutter UI by including common widgets.

THEORY:-

What are Widgets ?

Widgets in Flutter are the fundamental building blocks used to create user interfaces. They are either stateless or stateful.

● StatelessWidget: Represents static UI elements that do not change over time. Stateless widgets are immutable and only have a build method to describe their UI.
● StatefulWidget: Represents dynamic UI elements that can change over time based on user interactions or other factors. StatefulWidget maintains state that can change during the widget's lifetime. It consists of two classes: one for the widget itself and one for the mutable state.

Widgets can be composed hierarchically, allowing for the creation of complex UIs. They have a life cycle that includes creation, initialization, updating, and destruction. When the UI needs to be updated, Flutter efficiently rebuilds only the widgets that have Changed.

Flutter provides two sets of widgets for building UIs: Material Design widgets, which follow Google's Material Design guidelines, and Cupertino widgets, which mimic the iOS design language. Developers can choose the appropriate set of widgets based on the platform they are targeting.

Understanding widgets and how to use them effectively is essential for creating Flutter apps with dynamic and visually appealing user interfaces.

● Layout Widgets: - Column: Arranges its children widgets vertically. - SingleChildScrollView: Provides a scrollable view for its child widget.

● Material Design Widgets: - Scaffold: Provides a basic layout structure for the app, including app bar and body. - AppBar: Represents the app bar at the top of the screen. - Card: Represents a rectangular card with a shadow, used for containing information.

● Text Display Widgets: - Text: Displays a string of text with the specified style.

● Input and Interaction Widgets: - ElevatedButton: Represents a button that becomes elevated with a shadow when pressed.

● Utility Widgets: - Padding: Adds padding around its child widget. - SizedBox: Creates a box with a specified size.

● Styling and Theming: - ThemeData: Defines the overall theme for the app, including colors and text styles. - TextStyle: Defines the style of text, such as font size, font weight, etc.

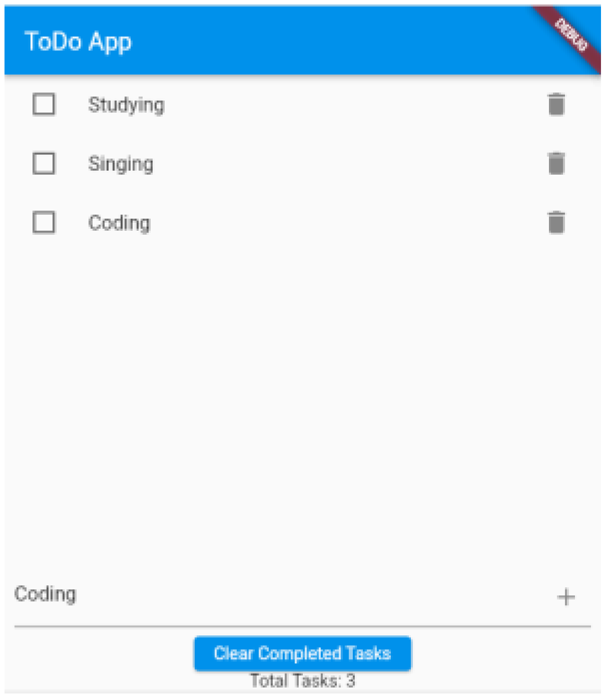
CODE:

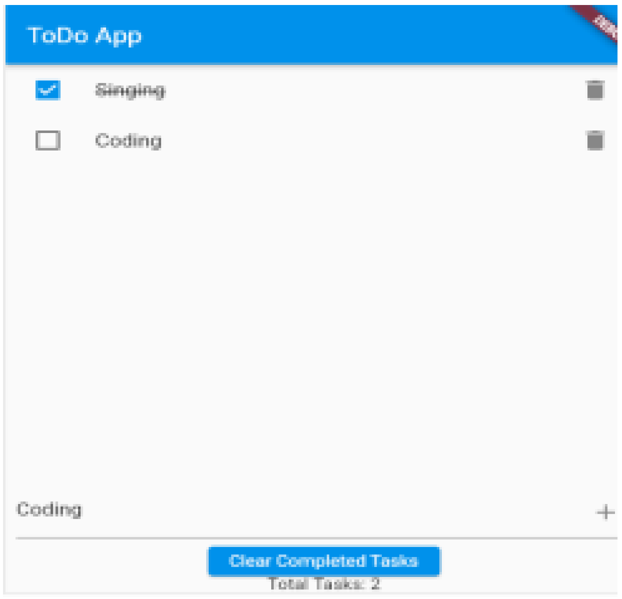
```
import 'package:flutter/material.dart';

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'ToDo App',
      home: ToDoScreen(),
    );
  }
}

class ToDoScreen extends StatefulWidget {
  @override
  _ToDoScreenState createState() => _ToDoScreenState();
}

class _ToDoScreenState extends State<ToDoScreen> {
  List<Task> tasks = [];

  @override
  Widget build(BuildContext context) {
```

```
return Scaffold(
  appBar: AppBar(
    title: Text('ToDo App'),
  ),
  body: Column(
    children: [
      Expanded(
        child: ListView.builder(
          itemCount: tasks.length,
          itemBuilder: (context, index) {
            return ListTile(
              leading: Checkbox(
                value: tasks[index].isCompleted,
                onChanged: (value) {
                  setState(() {
                    tasks[index].isCompleted = value!;
                  });
                },
              ),
              title: Text(
                tasks[index].taskName,
                style: TextStyle(
                  decoration: tasks[index].isCompleted
                      ? TextDecoration.lineThrough
                      : null,
                ),
              ),
              trailing: IconButton(
                icon: Icon(Icons.delete),
                onPressed: () {
                  // Handle task deletion
                  setState(() {
                    tasks.removeAt(index);
                  });
                },
              ),
            );
          },
        ),
      ),
      Padding(
        padding: const EdgeInsets.all(8.0),
        child: TextField(
          onSubmitted: (value) {
            // Handle task addition
            setState(() {
              tasks.add(Task(taskName: value));
            });
          },
          decoration: InputDecoration(
            hintText: 'Add a new task...',
            suffixIcon: Icon(Icons.add),
          ),
        ),
      ),
      ElevatedButton(
        onPressed: () {
          // Clear completed tasks
          setState(() {
            tasks.removeWhere((task) => task.isCompleted);
          });
        },
        child: Text('Clear Completed Tasks'),
      ),
      Text('Total Tasks: ${tasks.length}'),
    ],
   ),
  );
 }
}

class Task {
 String taskName;
 bool isCompleted;
```

```
  Task({required this.taskName, this.isCompleted = false});
}
```

OutPut:





CONCLUSION: Conclusion:Successfully implemented the Widgets such as Stateful Widget and Stateless Widget.