

**Aim: Implementation of stateful and stateless widget and flutter layout widgets in container.**

**Theory:**

### **Q1. Why Use Flutter?**

Flutter is a UI toolkit developed by Google for building natively compiled applications for mobile, web, and desktop from a single codebase. It provides fast development, expressive UI, and high performance.

### **Q2. List Flutter of Features.**

- Hot Reload - Allows developers to see changes instantly without restarting the app.
- Customizable Animation - Flutter provides rich animation APIs for smooth and flexible UI animations.
- Cross-Platform Development - Single codebase works on Android, iOS, Web, and Desktop.
- Integration - Easily integrates with existing native code, APIs, and third-party libraries

### **Q3. Components of Flutter**

- Dart Framework: Provides widgets, rendering, animation, and foundation libraries.
- Flutter Engine: Written in C++, responsible for rendering UI, handling input, and low-level graphics using Skia

### **Q4. Explain Categories of Flutter Widgets**

#### **a) Structural Widgets**

**Purpose:** Used to define the structure and layout of the UI.

**Common Widgets:**

- Scaffold
- AppBar
- Column, Row
- Center
- Stack
- ListView
- GridView

#### **b) Styling and Theming Widgets**

**Purpose:** Used for UI appearance, visual design, and theming.

**Common Widgets:**

- Theme
- Container

- TextStyle
- BoxDecoration
- Border
- Padding, Margin

### c) Interactive Widgets

**Purpose:** Respond to user interactions and gestures.

#### Common Widgets:

- ElevatedButton
- IconButton
- GestureDetector
- InkWell
- Draggable
- Dismissible

### d) Input / Output Widgets

**Purpose:** Used to accept user input and display data/output.

#### Common Widgets:

- TextField
- Checkbox
- Switch
- Radio
- Slider
- DropdownButton
- Text (output)
- Image (output)

**Q4. Create an app using Flutter that includes at least three screens, incorporating stateless widgets, stateful widgets, and animations.**

#### Code (main.dart):

##### Home Page:

```
// ===== HOME SCREEN (Screen 1)
// =====
class HomeScreen extends StatelessWidget {
  const HomeScreen({super.key});

  @override
```

```
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: const Text('Home Screen'),
      backgroundColor:
        Theme.of(context).colorScheme.primary,
      foregroundColor: Colors.white,
    ),
    body: Center(
```

```

        child: Column(
          mainAxisAlignment:
MainAxisAlignment.center,
          children: [
            // Animated Icon using
Implicit Animation
            const AnimatedHomeIcon(),
            const SizedBox(height: 30),
            // Custom Stateless Widget
            const FeatureCard(
              icon: Icons.countertops,
              title: 'Counter Screen',
              description: 'A stateful
counter with animations',
            ),
            const SizedBox(height: 15),
            const FeatureCard(
              icon: Icons.animation,
              title: 'Animation Screen',
              description: 'Various
Flutter animations',
            ),
            const SizedBox(height: 15),
            const FeatureCard(
              icon: Icons.palette,
              title: 'Theme Screen',
              description: 'Custom theme
customization',
            ),
            const SizedBox(height: 40),
            // Navigation Buttons
            ElevatedButton(
              onPressed: () {
                Navigator.push(
                  context,

MaterialPageRoute(builder: (context) =>
const CounterScreen()),
                ),
              child: const Text('Go to
Counter Screen'),
            ),
            const SizedBox(height: 15),
            ElevatedButton(
              onPressed: () {
                Navigator.push(
                  context,

MaterialPageRoute(builder: (context) =>
const AnimationScreen()),
                ),
              child: const Text('Go to
Animation Screen'),
            ),
          ],

```

```

        ),
      ),
    );
  }
}

```

### Counter Page:

```

// Another Stateless Widget
class CounterStatus extends
StatelessWidget {
  final int counterValue;

  const CounterStatus({super.key,
required this.counterValue});

  @override
  Widget build(BuildContext context) {
    String status;
    Color color;

    if (counterValue == 0) {
      status = 'Counter is at zero';
      color = Colors.grey;
    } else if (counterValue < 10) {
      status = 'Counter is small';
      color = Colors.blue;
    } else if (counterValue < 20) {
      status = 'Counter is medium';
      color = Colors.orange;
    } else {
      status = 'Counter is large!';
      color = Colors.red;
    }

    return Container(
      padding: const EdgeInsets.all(12),
      decoration: BoxDecoration(
        color: color.withOpacity(0.1),
        borderRadius:
BorderRadius.circular(10),
        border: Border.all(color: color,
width: 2),
      ),
      child: Row(
        mainAxisAlignment: MainAxisAlignment.min,
        children: [
          Icon(
            Icons.info,
            color: color,
          ),
          const SizedBox(width: 8),
          Text(
            status,
            style: TextStyle(
              fontSize: 16,
              fontWeight:

```

```

fontWeight.bold,
      color: color,
    ),
  ),
],
),
);
}
}

```

### Animation Page:

```

// Animated Stateless Widget using
Implicit Animation
class AnimatedHomeIcon extends
StatelessWidget {
  const AnimatedHomeIcon({super.key});

  @override
  Widget build(BuildContext context) {
    return AnimatedContainer(
      duration: const Duration(seconds:
1),
      curve: Curves.easeInOut,
      width: 150,
      height: 150,
      decoration: BoxDecoration(
        gradient: LinearGradient(
          colors: [
Theme.of(context).colorScheme.primary,
Theme.of(context).colorScheme.secondary,
],
        begin: Alignment.topLeft,
        end: Alignment.bottomRight,
      ),
      shape: BoxShape.circle,
      boxShadow: [
        BoxShadow(
          color:
Theme.of(context).colorScheme.primary.with
Opacity(0.5),
          blurRadius: 15,
          spreadRadius: 5,
        ),
      ],
    ),
    child: const Icon(
      Icons.flutter_dash,
      color: Colors.white,
      size: 80,
    ),
  );
}
}

```

```

// Stateful Widget for Bouncing Animation
class BouncingBall extends StatefulWidget {
  const BouncingBall({super.key});

  @override
  State<BouncingBall> createState() =>
_BouncingBallState();
}

```

```

class _BouncingBallState extends
State<BouncingBall>
  with SingleTickerProviderStateMixin {
  late AnimationController
_bounceController;
  late Animation<double>
_bounceAnimation;

```

```

  @override
  void initState() {
    super.initState();
    _bounceController =
AnimationController(
      duration: const
Duration(milliseconds: 800),
      vsync: this,
    )..repeat(reverse: true);

    _bounceAnimation =
Tween<double>(begin: 0, end: 30).animate(
      CurvedAnimation(
        parent: _bounceController,
        curve: Curves.elasticOut,
      ),
    );
  }

```

```

  @override
  void dispose() {
    _bounceController.dispose();
    super.dispose();
  }

```

```

  @override
  Widget build(BuildContext context) {
    return Column(
      children: [
        const Text(
          'Bouncing Ball Animation',
          style: TextStyle(fontSize: 16,
fontWeight: FontWeight.bold),
        ),
        const SizedBox(height: 10),
        AnimatedBuilder(
          animation: _bounceController,
          builder: (context, child) {
            return Transform.translate(

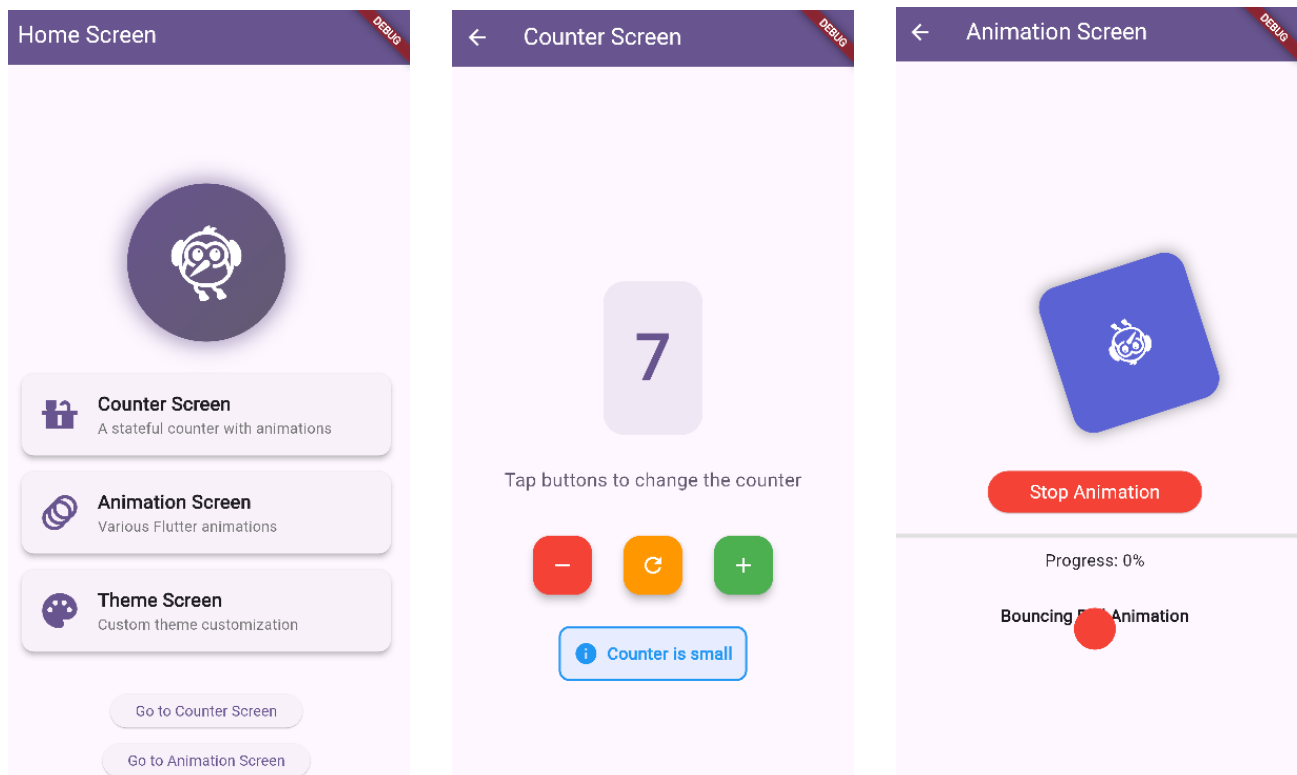
```

```

        offset: Offset(0, -
_bounceAnimation.value),
        child: Container(
          width: 40,
          height: 40,
          decoration: const
BoxDecoration(
          color: Colors.red,
          shape: BoxShape.circle,
        ),
      ),
    ),
  ),
);

```

**Output:**



## Conclusion

This experiment successfully demonstrated the use of stateless widgets, stateful widgets, and Flutter layout widgets in building a multi-screen Flutter application. Navigation between screens and state management were implemented effectively using Flutter's widget system.

The experiment also showcased the use of animations to enhance user interaction. Overall, it helped in understanding Flutter's core concepts and UI development approach.