WEEK-8

a) Add animations to UI elements using Flutter's animation framework.

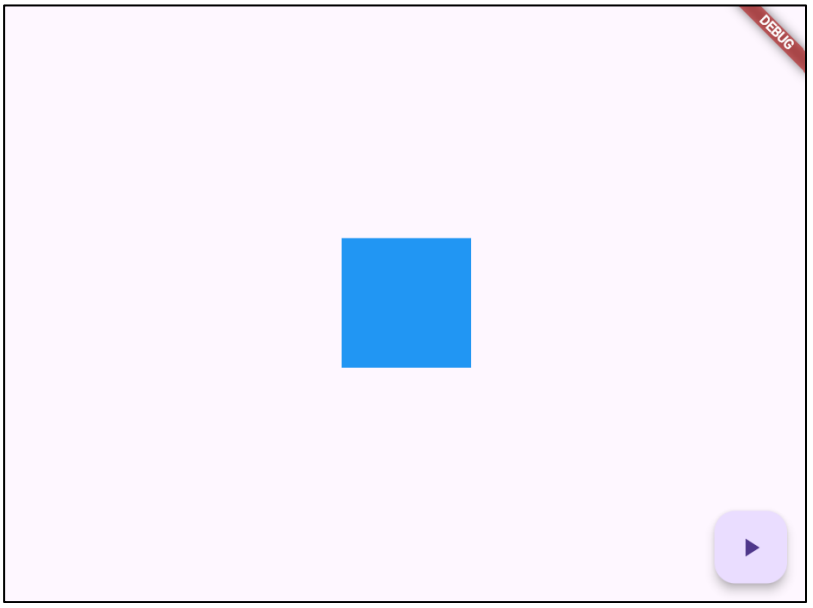
AIM: To add animations to UI elements using Flutter's animation framework.

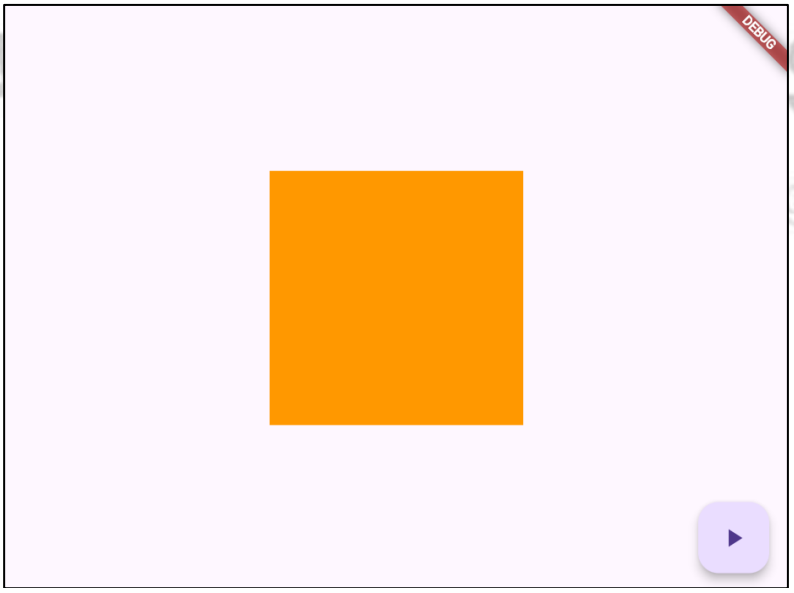
DESCRIPTION: Flutter's animation framework allows you to create smooth, interactive animations by controlling UI elements over time. It uses classes like **AnimationController** to manage animation duration and state, **Tween** to define value ranges, and **AnimatedBuilder** or **AnimatedWidget** to rebuild UI based on animation changes. You can animate properties such as size, color, position, and opacity to enhance user experience.

SOURCECODE:

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

void main() => runApp(MaterialApp(home: SimpleAnimation()));

class SimpleAnimation extends StatefulWidget{
  @override
  _SimpleAnimationState createState() =>
    _SimpleAnimationState();
}

class _SimpleAnimationState extends State<SimpleAnimation> {
  bool big = false; //one flag handles both size and colour

  @override
  Widget build(BuildContext context){
    return Scaffold(
    body: Center(
    child: AnimatedContainer(
    width: big ? 200 :100,
    height: big ? 200 :100,
    color: big ? Colors.orange : Colors.blue,
    duration: const Duration(seconds : 1),
    ),
    ),
    floatingActionButton: FloatingActionButton(
    onPressed: () => setState(() => big = !big),
    child: const Icon(Icons.play_arrow),
    ),
    );
  }
}
```

OUTPUT:





b) Experiment with different types of animations (fade, slide, etc.).

AIM: To experiment with different types of animations (fade, slide, etc.).

DESCRIPTION:

- **Fade** – Element appears/disappears smoothly.
- **Slide** – Moves in/out from a direction.
- **Zoom** – Grows or shrinks in size.
- **Rotate** – Spins around a point.
- **Bounce** – Moves with spring effect.
- **Flip** – Turns like a card.

SOURCECODE:

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

void main() => runApp(MaterialApp(home: SimpleAnimation()));

class SimpleAnimation extends StatefulWidget {
  @override
  _SimpleAnimationState createState() => _SimpleAnimationState();
}

class _SimpleAnimationState extends State<SimpleAnimation> {
  bool on = false; // one flag controls everything

  @override
  Widget build(BuildContext context) {
   return Scaffold(
     body: Center(
       child: Column(
         mainAxisAlignment: MainAxisAlignment.center,
         children: [
           AnimatedOpacity(
             opacity: on ? 1 : 0,
             duration: const Duration(seconds: 1),
             child: _box(Colors.blue),
           ),
           const SizedBox(height: 20),
           AnimatedSlide(
             offset: on ? Offset.zero : const Offset(-1, 0),
             duration: const Duration(seconds: 1),
             child: _box(Colors.green),
           ),
           const SizedBox(height: 20),
           AnimatedScale(
             scale: on ? 1 : 0.5,
             duration: const Duration(seconds: 1),
             child: _box(Colors.orange),
           ),
         ],
       ),
     ),
     floatingActionButton: FloatingActionButton(
```

```
      onPressed: () => setState(() => on = !on),
      child: const Icon(Icons.play_arrow),
    ),
  );
 }

 Widget _box(Color c) => Container(
   width: 80, height: 80, color: c);
}
```

OUTPUT: