1. **a) Install Flutter and Dart SDK.**

**AIM:** To install Flutter and the Dart SDK, you can follow these steps:

a) Download Flutter: Visit the Flutter website's Get Started page and download the Flutter SDK for your operating system (Windows, macOS, or Linux).

b) Extract the Flutter SDK: After downloading, extract the contents of the compressed file to a location on your computer where you want to store the Flutter SDK. For example, you can extract it to C:\flutter on Windows, /Users/<your-username>/flutter on macOS, or ~/flutter on Linux.

c) Add Flutter to your PATH: Update your system's PATH variable to include the Flutter bin

directory. This step allows you to execute Flutter commands from any directory in your terminal or command prompt. The precise steps for updating the PATH vary depending on your operating system.

**Windows:** From the Start search bar, type 'env' and select 'Edit the system environment

variables'. Click on 'Environment Variables'. Under 'System Variables', find the 'Path' variable, select it, and click 'Edit'. Click 'New' and add the path to the bin directory insidethe Flutter directory (e.g., C:\flutter\bin). Click 'OK' on all open dialogs to save your changes.

**macOS and Linux:** Open a terminal window.

Run the following command to open the profile file associated with your terminal

(.bash\_profile, .bashrc, .zshrc, or similar):

nano ~/.bash\_profile Add the following line at the end of the file:

export PATH="$PATH:/path/to/flutter/bin"

Press Ctrl + X to exit, then Y to save changes, and Enter to confirm.

d) Verify the Flutter installation: Open a new terminal window, and run the following

command to verify that Flutter is properly installed: flutter –version. This comman should display the Flutter version and other relevant information if the installation was successful.

e) Install Flutter dependencies: Depending on your development environment, you may need to install additional dependencies, such as Android Studio to fully set up your Flutter development environment.

f) Download Dart SDK (if not bundled with Flutter): Flutter comes with the Dart SDK

bundled, so if you've installed Flutter, you should have the Dart SDK as well. However, if you need to install Dart separately, you can download it from the Dart "SDK archive".

**b) Write a simple dart program to understand the language basics.**

import 'package:flutter/material.dart';

void main() {

runApp(Abc());

}

class Abc extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

home: Def(),

);

}

}

class Def extends StatelessWidget {

const Def({super.key});

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text("Welcome"),

backgroundColor: Colors.purple,

),

body: Column(

children: [

//Widgets

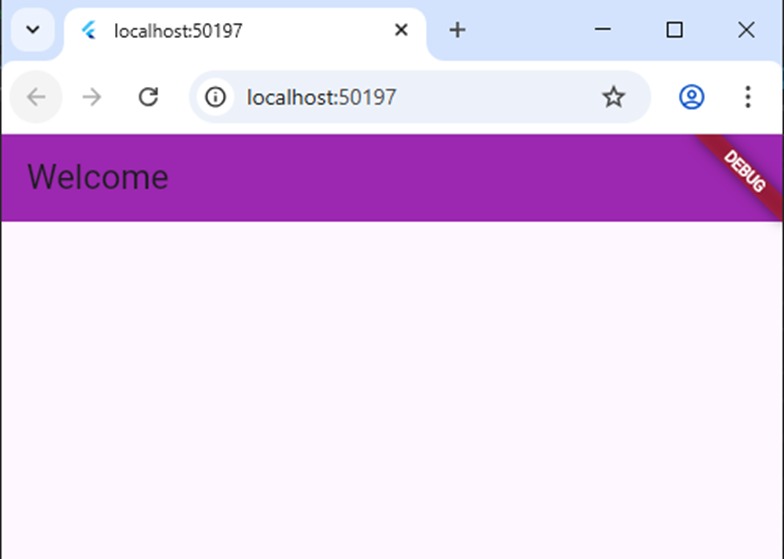
],

),

);

}

**OUTPUT:**



1. **a) Explore various Flutter widgets (Text, Image, Container, etc.).**

**Description:**

Flutter provides a rich set of widgets to build user interfaces for mobile, web, and desktop

applications. These widgets help in creating visually appealing and interactive UIs. Here are

some of the commonly used Flutter widgets categorized by their functionalities:

Layout Widgets:

**Container:** A versatile widget that can contain other widgets and provides options for alignment, padding, margin, and decoration.

**Row and Column:** Widgets that arrange their children in a horizontal or vertical line respectively.

**Stack**: Allows widgets to be stacked on top of each other, enabling complex layouts.

**ListView and GridView**: Widgets for displaying a scrollable list or grid of children, with support for various layouts and scrolling directions.

**Scaffold:** Implements the basic material design layout structure, providing app bars, drawers, and floating action buttons.

**Text and Styling Widgets:**

**Text:** Displays a string of text with options for styling such as font size, color, and alignment.

**RichText:** Allows for more complex text styling and formatting, including different styles within the same text span.

**TextStyle:** A class for defining text styles that can be applied to Text widgets.

**Input Widgets:**

**TextField:** A widget for accepting user input as text, with options for customization and validation.

**Checkbox and Radio:** Widgets for selecting from a list of options, either through checkboxes or radio buttons.

**DropdownButton**: Provides a dropdown menu for selecting from a list of options.

**Button Widgets:**

**ElevatedButton and TextButton:** Widgets for displaying buttons with different styles and customization options.

**IconButton:** A button widget that displays an icon and responds to user taps.

**GestureDetector:** A versatile widget that detects gestures such as taps, swipes, and drags, allowing for custom interactions.

**Image and Icon Widgets:**

**Image:** Widget for displaying images from various sources, including assets, network URLs, and memory.

**Icon:** Displays a Material Design icon.

**Navigation Widgets:**

**Navigator**: Manages a stack of route objects and transitions between different screens or pages in the app.

**PageRouteBuilder:** A customizable widget for building page transitions and animations.

**Animation Widgets:**

**AnimatedContainer:** An animated version of the Container widget, with support for transitioning properties over a specified duration.

**AnimatedOpacity, AnimatedPositioned, AnimatedBuilder:** Widgets for animating opacity, position,and custom properties respectively.

**Material Design Widgets:**

**AppBar:** A material design app bar that typically contains a title, leading and trailing widgets, and actions.

**BottomNavigationBar:** Provides a navigation bar at the bottom of the screen for switching betwee different screens or tabs.

**Card:** Displays content organized in a card-like structure with optional elevation and padding.

**Cupertino (iOS-style) Widgets:**

**CupertinoNavigationBar:** A navigation bar in the iOS style. CupertinoButton: A button widget with the iOS style.

**CupertinoTextField:** A text fieldwidget with the iOS style. These are just a few examples of the many widgets available in Flutter. Each widget comes with its set of properties and customization options, allowing developers to create highly customizable and responsive user interfaces.

**PROGRAM: TEXT**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

home: Abc()

);

}

}

class Abc extends StatelessWidget {

const Abc({super.key});

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(title: Text('Text Widget Example')),

body: Center(

child: Text(

'Hello, Flutter!',

style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),

),

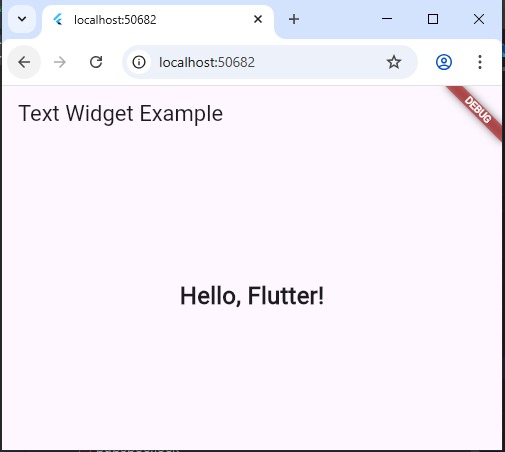
),

);

}

}

**OUTPUT:**

****

**PROGRAM: IMAGE**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

home: Abc(),

);

}

}

class Abc extends StatelessWidget {

const Abc({super.key});

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text("Image Widget"),

),

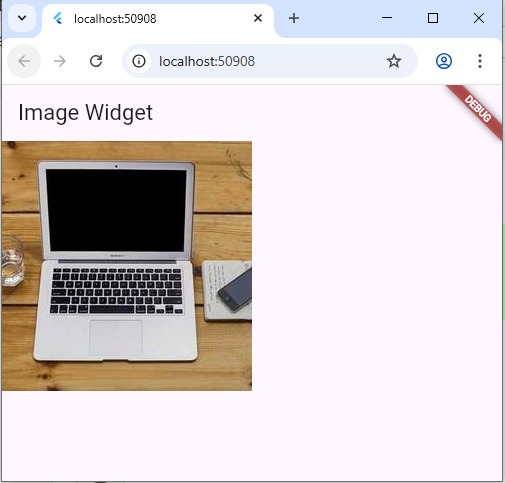
body: Image.network('https://picsum.photos/250?image=9'),

);

}

}

**OUTPUT:**

****

**PROGRAM: CONTAINER**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

home: Abc(),

);

}

}

class Abc extends StatelessWidget {

const Abc({super.key});

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(title: Text('Container Widget Example')),

body: Center(

child: Container(

width: 200,

height: 200,

padding: EdgeInsets.all(16),

margin: EdgeInsets.all(16),

decoration: BoxDecoration(

color: Colors.blue,

borderRadius: BorderRadius.circular(8),

boxShadow: [

BoxShadow(

color: Colors.black26,

blurRadius: 10,

offset: Offset(2, 2),

),

],

),

child: Center(

child: Text(

'Container',

style: TextStyle(color: Colors.white, fontSize: 24),

),

),

),

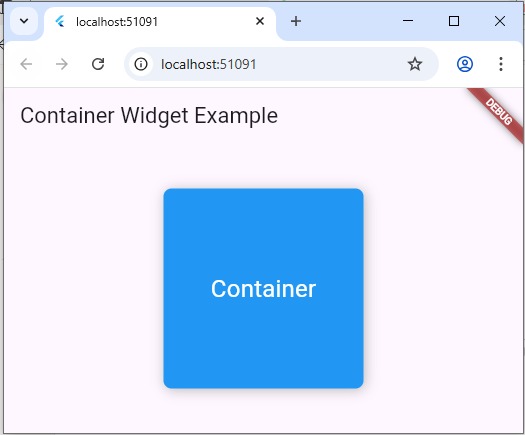
),

);

}

}

**OUTPUT:**

****

**b) Implement different layout structures using Row, Column, and Stack widgets.**

**PROGRAM: ROW WIDETS.**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp( home:

Scaffold(appBar: AppBar(

title: Text('Row Layout'),

),

body: Row(

mainAxisAlignment: MainAxisAlignment.spaceEvenly, children:

<Widget>[

Container(

color: Colors.red,

width: 100, height: 100,

), Container(

color: Colors.green,

width: 100, height: 100,

),

Container(

color: Colors.blue, width: 100,

height: 100,

),

],

),

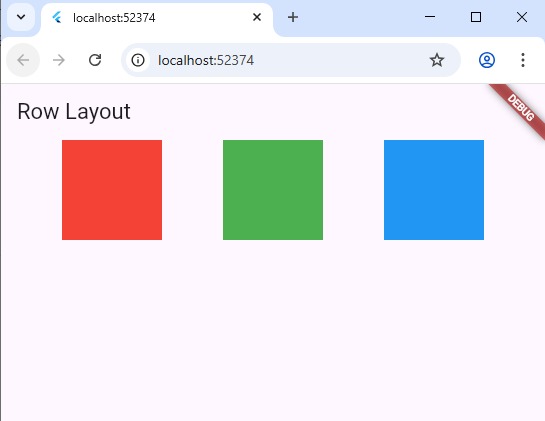
),

);

}

}

**OUTPUT:**

****

**PROGRAM: COLUMN WIDETS.**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp( home:

Scaffold(appBar: AppBar(

title: Text('Column Layout'),

),

body: Column(

mainAxisAlignment: MainAxisAlignment.spaceEvenly, children:

<Widget>[

Container(

color: Colors.red, width: 100,

height: 100,

), Container(

color: Colors.green, width: 100,

height: 100,

), Container(

color: Colors.blue, width: 100,

height: 100,

),

],

),

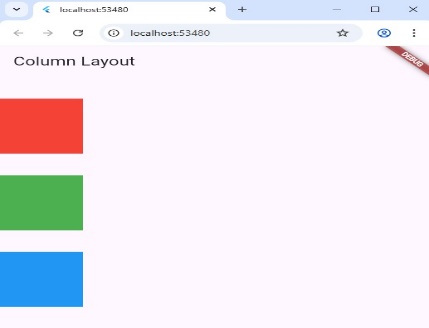
),

);

}

}

**OUTPUT:**

****

**PROGRAM: STACK WIDETS.**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp( home:

Scaffold(appBar: AppBar(

title: Text('Stack Layout'),

),

body: Stack(

alignment: Alignment.center, children:

<Widget>[

Container(

color: Colors.red, width: 200,

height: 200,

),

Container(

color: Colors.green, width: 150,

height: 150,

), Container(

color: Colors.blue, width: 100,

height: 100,

),

],

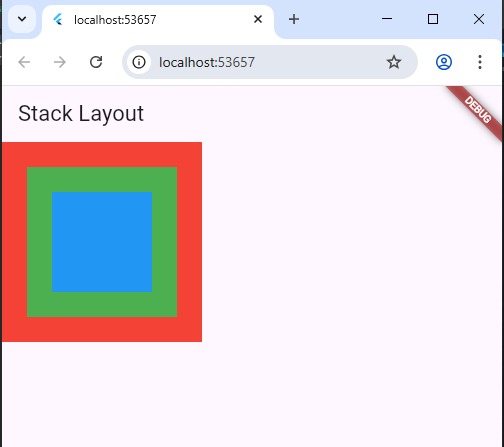
),

),

) ;

} }

**OUTPUT:**

****

**3. a) Design a responsive UI that adapts to different screen sizes.**

**PROGRAM:**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'Responsive UI Demo', theme:

ThemeData( primarySwatch:

Colors.blue,

),

home: ResponsiveHomePage(),

);

}

}

class ResponsiveHomePage extends StatelessWidget {

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Responsive UI Demo'),

),

body: LayoutBuilder(

builder: (BuildContext context, BoxConstraints constraints) {

if (constraints.maxWidth < 600) {

return \_buildNarrowLayout();

} else {

return \_buildWideLayout();

}

},

),

);

}

Widget \_buildNarrowLayout() {

return Center(

child: Column(

mainAxisAlignment: MainAxisAlignment.center, children:

<Widget>[

FlutterLogo(size: 100),

SizedBox(height: 20), Text(

'Narrow Layout',

style: TextStyle(fontSize: 24),

),

SizedBox(height: 20),

ElevatedButton( onPressed:

() {},

child: Text('Button'),

),

],

),

);

}

Widget \_buildWideLayout() {

return Center(

child: Row(

mainAxisAlignment: MainAxisAlignment.center,

children: <Widget>[

FlutterLogo(size: 100),

SizedBox(width: 20),

Column(

mainAxisAlignment: MainAxisAlignment.center,

children: <Widget>[

Text(

'Wide Layout',

style: TextStyle(fontSize: 24),

),

SizedBox(height: 20),

ElevatedButton( onPressed: ()

{},

child: Text('Button'),

),

],

),

],

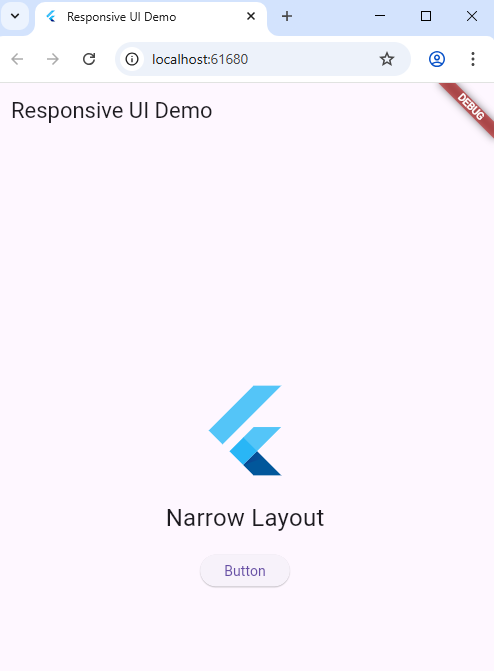
),

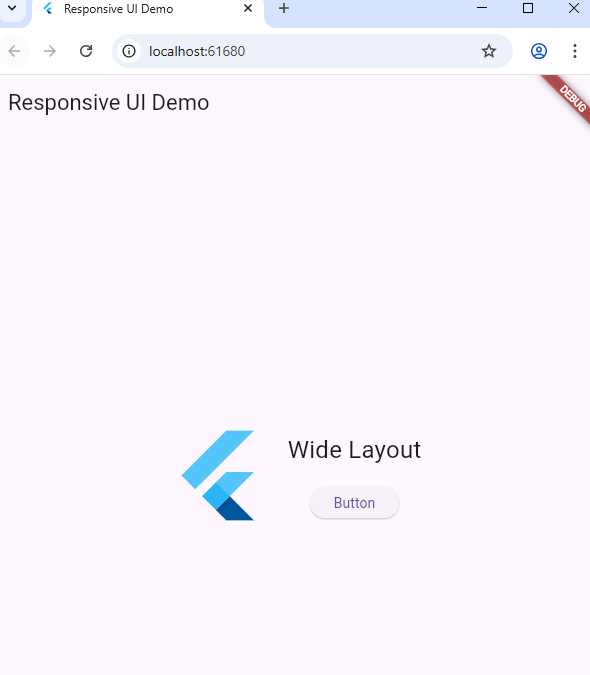
);

}

}

**OUTPUT:**





1. **b Implement media queries and breakpoints for responsiveness.**

**PROGRAM:**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends

StatelessWidget {

@override

Widget build(BuildContext

context) {

return MaterialApp(

title: 'Responsive UI with Media Queries', theme: ThemeData(

primarySwatch: Colors.blue,

),

home:

ResponsiveHomePage(),

);

}

}

class ResponsiveHomePage extends

StatelessWidget {

@override

Widget build(BuildContext context) {

return

Scaffold(

appBar: AppBar(

title: Text('Responsive UI with Media Queries'),

),

body: LayoutBuilder(

builder: (BuildContext context, BoxConstraints constraints) {

if (constraints.maxWidth < 600) {

return \_buildMobileLayout();

} else if (constraints.maxWidth < 1200) {

return \_buildTabletLayout();

} else {

return \_buildDesktopLayout();

}

},

),

);

}

Widget \_buildMobileLayout() {

return Center(

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: <Widget>[

FlutterLogo(size: 100),

SizedBox(height: 20),

Text(

'Mobile Layout',

style: TextStyle(fontSize: 24),

),

SizedBox(height: 20),

ElevatedButton(

onPressed: () {},

child: Text('Button'),

),

],

),

);

}

Widget

\_buildTabletLayout() {

return

Center(

child:

Row(

mainAxisAlignment: MainAxisAlignment.center,

children: <Widget>[

FlutterLogo(size: 100),

SizedBox(width: 20),

Column(

mainAxisAlignment: MainAxisAlignment.center,

children: <Widget>[

Text(

'Tablet Layout',

style: TextStyle(fontSize: 24),

),

SizedBox(height: 20),

ElevatedButton(

onPressed: () {},

child: Text('Button'),

),

],

),

],

),

);

}

Widget \_buildDesktopLayout() {

return

Center(

child: Row(

mainAxisAlignment: MainAxisAlignment.center,

children: <Widget>[

FlutterLogo(size: 100),

SizedBox(width: 20),

Column(

mainAxisAlignment: MainAxisAlignment.center,

children: <Widget>[

Text(

'Desktop Layout',

style: TextStyle(fontSize: 24),

),

SizedBox(height: 20),

ElevatedButton(

onPressed: () {},

child: Text('Button'),

),

],

),

],

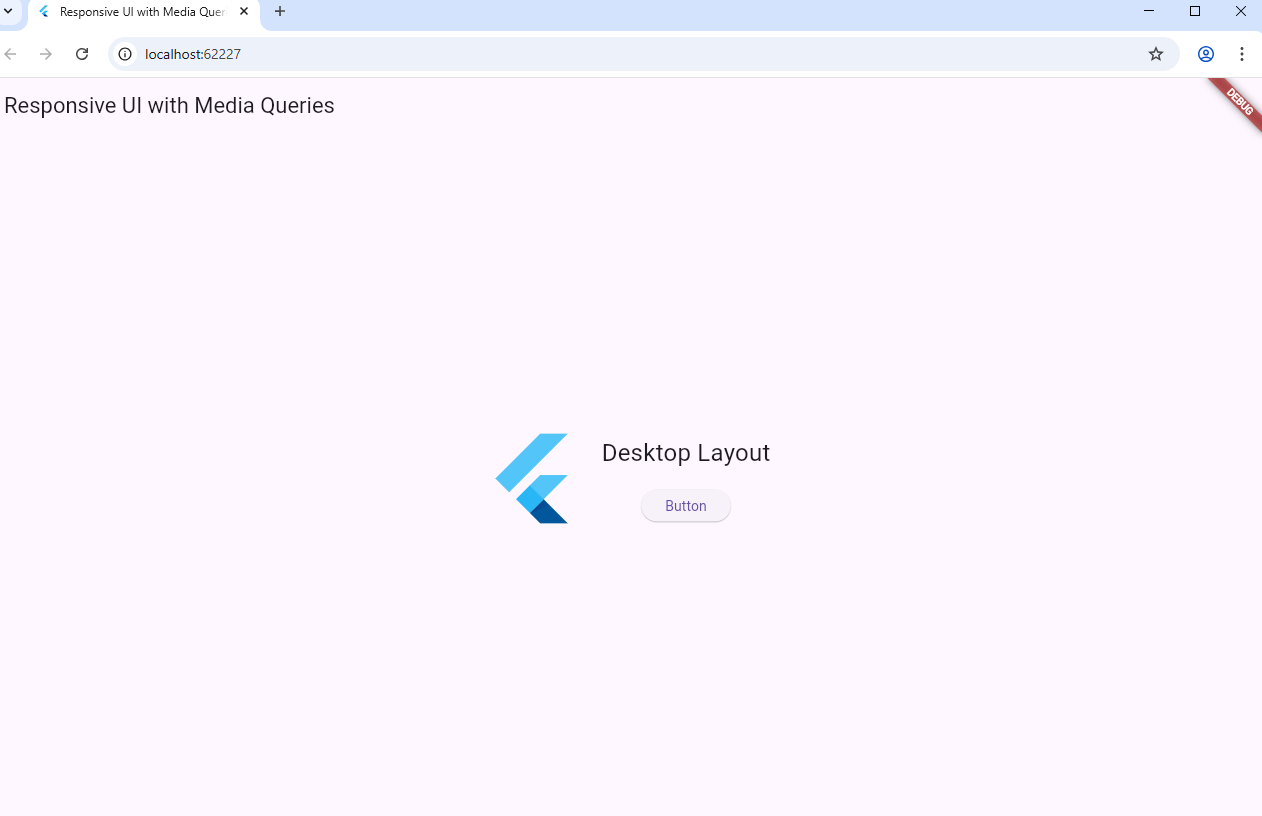
),

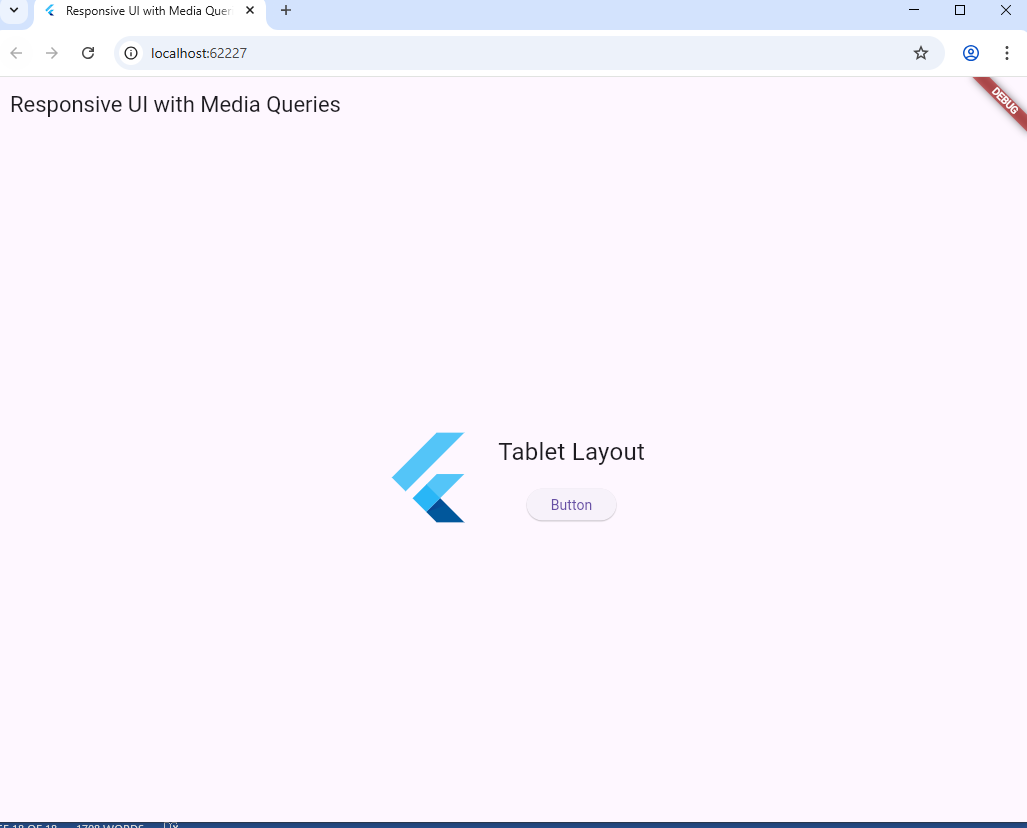
);

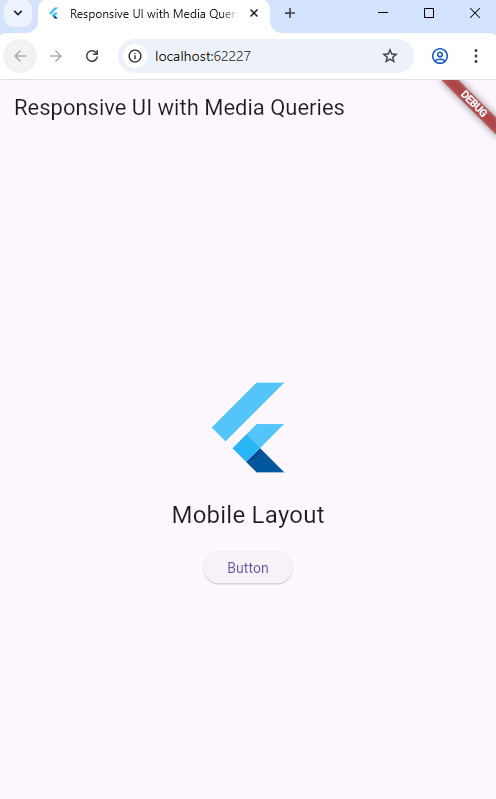
}

}

**OUTPUT:**







1. **a) Setup navigation between different screens using navigator**

**PROGRAM:**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'Navigation Example',

theme: ThemeData(

primarySwatch: Colors.blue,

),

home: FirstScreen(),

);

}

}

class FirstScreen extends StatelessWidget {

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('First Screen'),

),

body: Center(

child: ElevatedButton(

onPressed: () {

// Navigate to the second screen

Navigator.push(

context,

MaterialPageRoute(builder: (context) => SecondScreen()),

);

},

child: Text('Go to Second Screen'),

),

),

);

}

}

class SecondScreen extends StatelessWidget {

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Second Screen'),

),

body: Center(

child: ElevatedButton(

onPressed: () {

// Navigate back to the first screen

Navigator.pop(context);

},

child: Text('Go back to First Screen'),

),

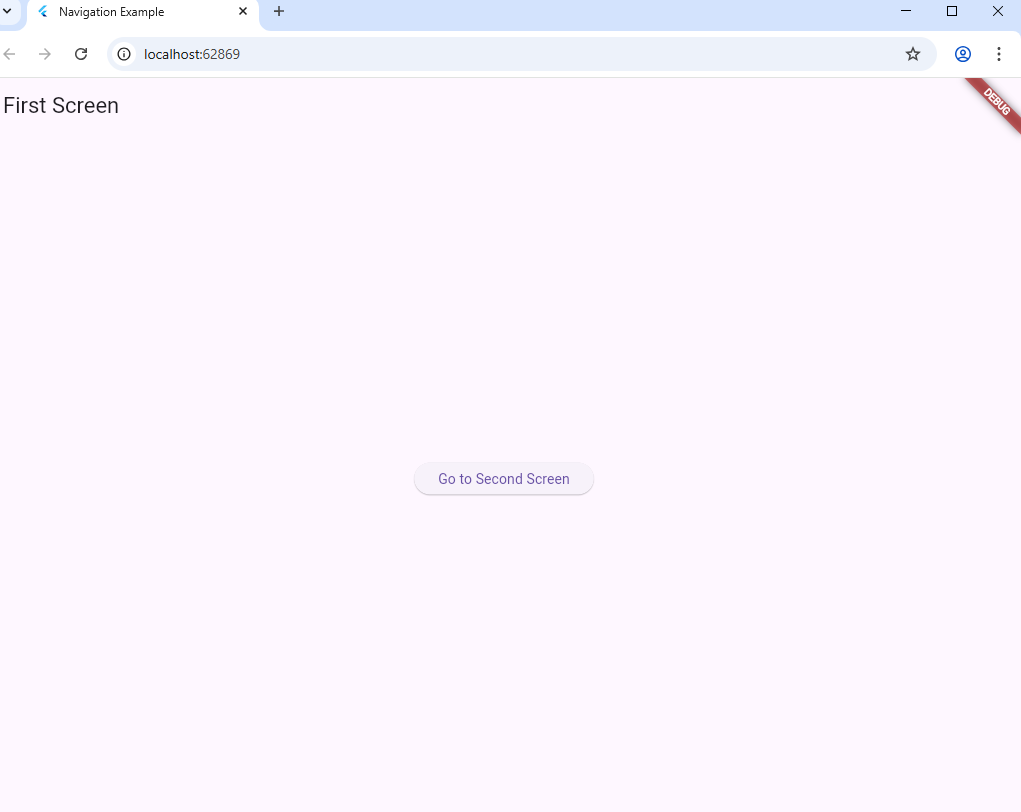
),

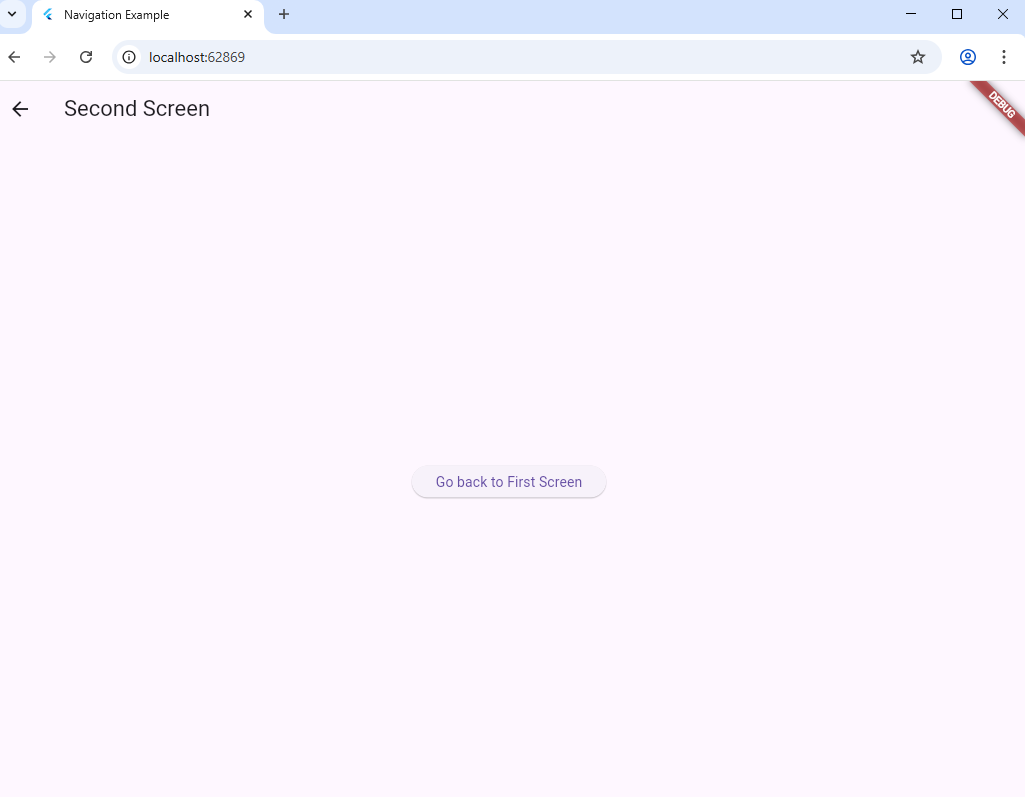
);

}

}

**OUTPUT:**





**4.b) Implement navigation with named routes**

**PROGRAM:**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends

StatelessWidget {

@override

Widget build(BuildContext

context) {

return MaterialApp(

title: 'Named RoutesDemo',

initialRoute: '/',

routes: {

'/': (context) => HomeScreen(),

'/second': (context) => SecondScreen(),

'/third': (context) => ThirdScreen(),

},

);

}

}

class HomeScreen extends StatelessWidget {

@override

Widget build(BuildContext context) {

return

Scaffold(

appBar: AppBar(

title: Text('Home Screen'),

),

body: Center(

child: ElevatedButton(

onPressed: () {

Navigator.pushNamed(context, '/second');

},

child: Text('Go to Second Screen'),

),

),

);

}

}

class SecondScreen extends StatelessWidget {

@override

Widget build(BuildContext context) {

return

Scaffold(

appBar: AppBar(

title: Text('Second Screen'),

),

body: Center(

child: ElevatedButton(

onPressed: () {

Navigator.pushNamed(context, '/third');

},

child: Text('Go to Third Screen'),

),

),

);

}

}

class ThirdScreen extends

StatelessWidget {

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Third Screen'),

),

body: Center(

child: ElevatedButton(

onPressed: () {

Navigator.popUntil(context, ModalRoute.withName('/'));

},

child: Text('Go Back to Home'),

),

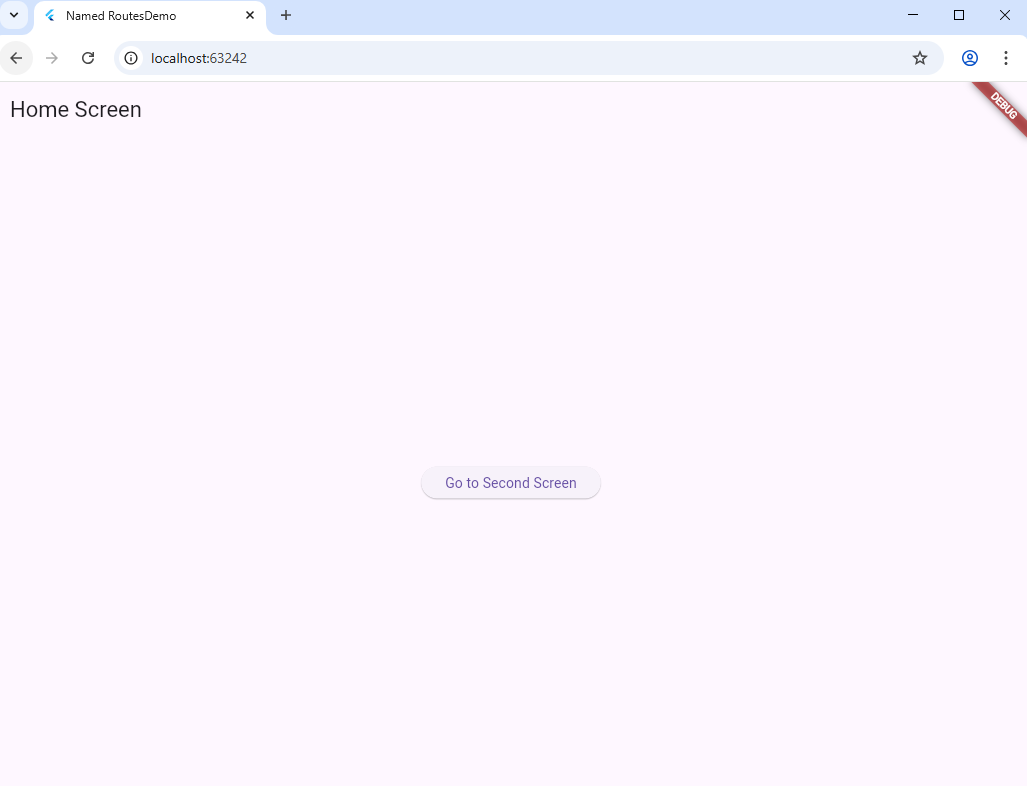
),

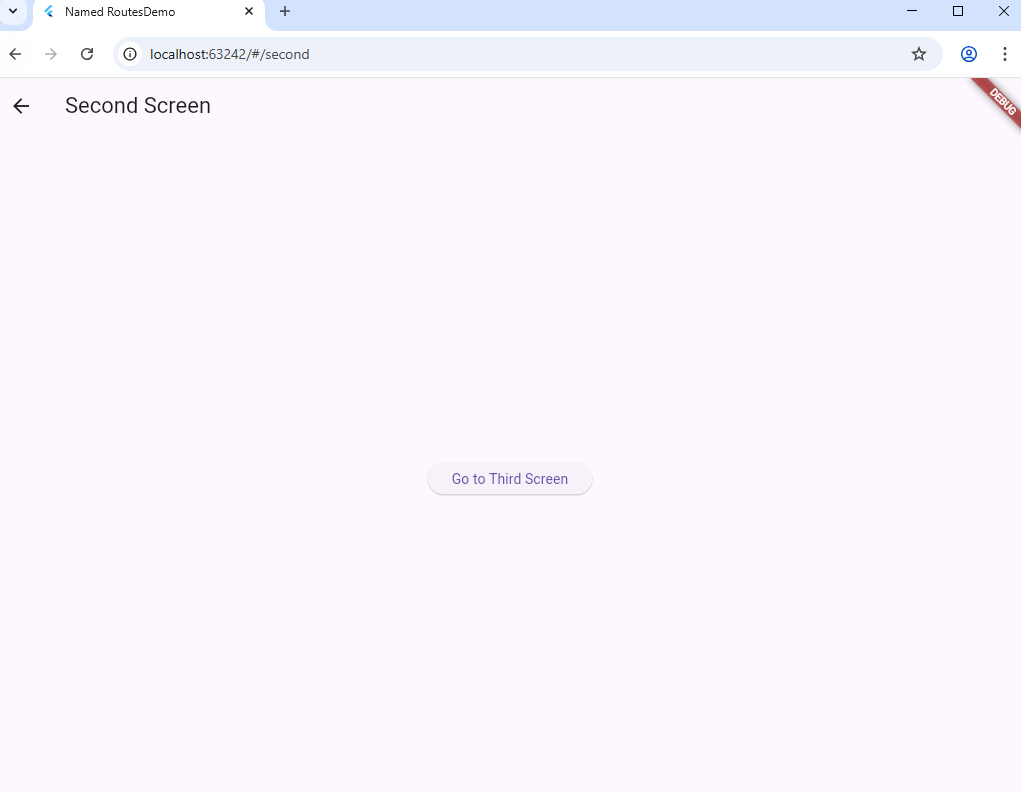
);

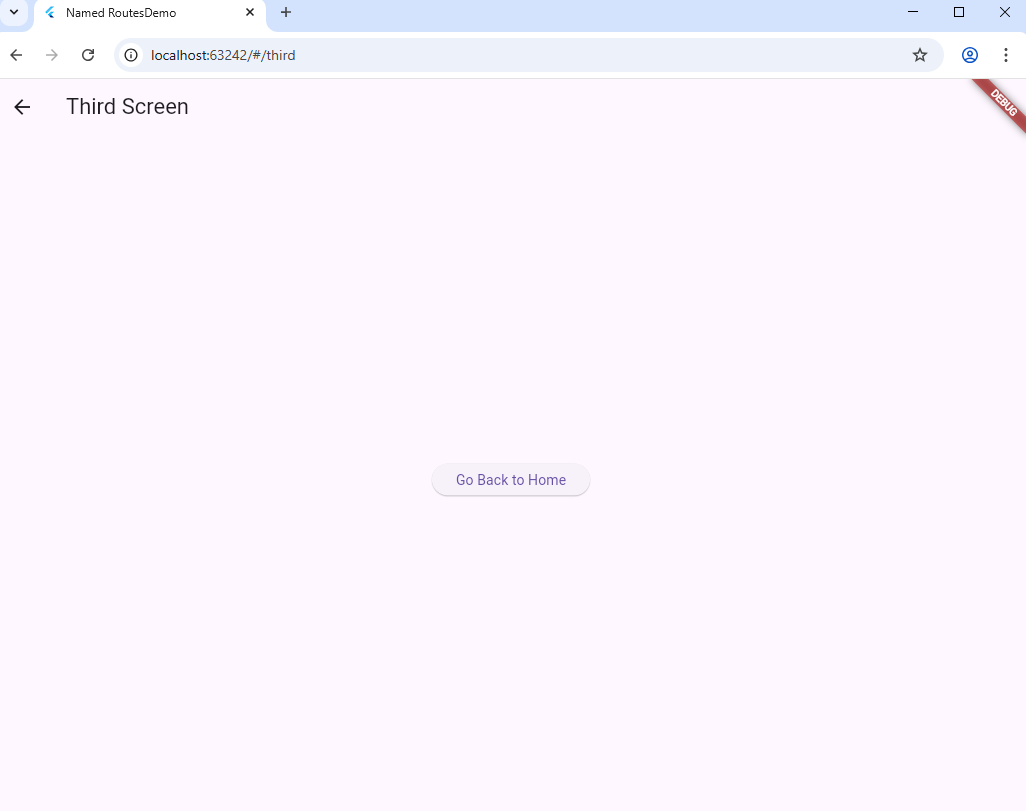
}

}

**OUTPUT:**







1. **a) Learn about stateful and stateless widgets**

**PROGRAM:**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

home: Scaffold(

appBar: AppBar(

title: Text('Cards Example'),

),

body: CardList(),

),

);

}

}

class CardList extends StatelessWidget {

@override

Widget build(BuildContext context) {

return ListView.builder(

itemCount: 10,

itemBuilder: (context, index) {

return CardItem(

title: 'Card $index',

subtitle: 'Subtitle $index',

);

},

);

}

}

class CardItem extends StatelessWidget {

final String title;

final String subtitle;

const CardItem({

Key? key,

required this.title,

required this.subtitle,

}) : super(key: key);

@override

Widget build(BuildContext context) {

return Card(

margin: EdgeInsets.symmetric(horizontal: 16, vertical: 8),

child: ListTile(

title: Text(title),

subtitle: Text(subtitle),

leading: CircleAvatar(

child: Text(title.substring(0, 1)),

),

onTap: () {

// Handle card tap

},

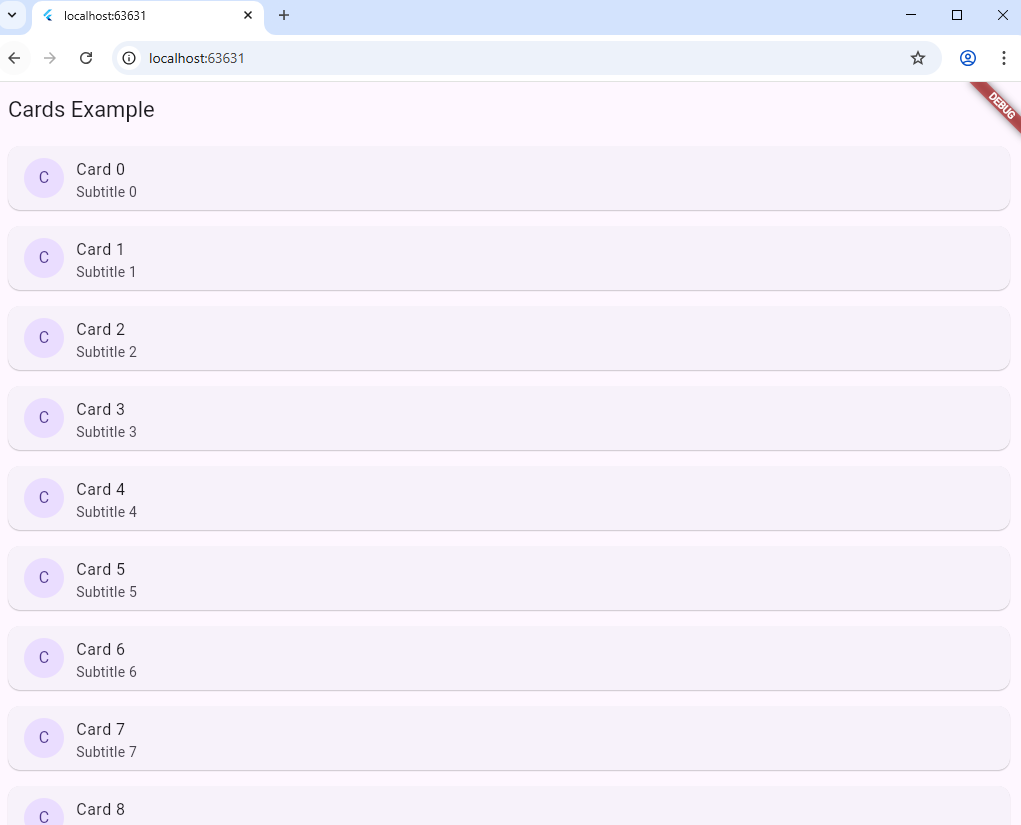
),

);

}

}

**OUTPUT:**



**5. b) Implement state management using set state and provider**

**Stateful Widgets:**

**Definition:** Stateful widgets are widgets that maintain state, allowing them to change and update over time in response to user actions, network events, or other factors. Characteristics: They have an associated mutable state that can change during the widget's lifetime. The state is stored in a separate class that extends State and is associated with the stateful widget. Changes to the state trigger a rebuild of the widget's UI, allowing dynamic updates. They are ideal for UI elements that need to change or react to user interactions, such as input forms, animations, or scrollable lists.

**PROGRAM:**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

home: CounterApp(),

);

}

}

class CounterApp extends StatefulWidget {

@override

\_CounterAppState createState() => \_CounterAppState();

}

class \_CounterAppState extends State<CounterApp> {

int \_counter = 0;

void \_incrementCounter() {

setState(() {

\_counter++;

});

}

@override

Widget build(BuildContext context) {

return

Scaffold

(

appBar: AppBar(

title: Text('Counter App'),

),

body: Center(

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: <Widget>[

Text(

'Counter:',

style: TextStyle(fontSize: 24),

), Text(

'$\_counter',

style: TextStyle(fontSize: 36, fontWeight: FontWeight.bold),

),

],

),

),

floatingActionButton: FloatingActionButton(

onPressed:

\_incrementCounter, tooltip:

'Increment',

child: Icon(Icons.add),

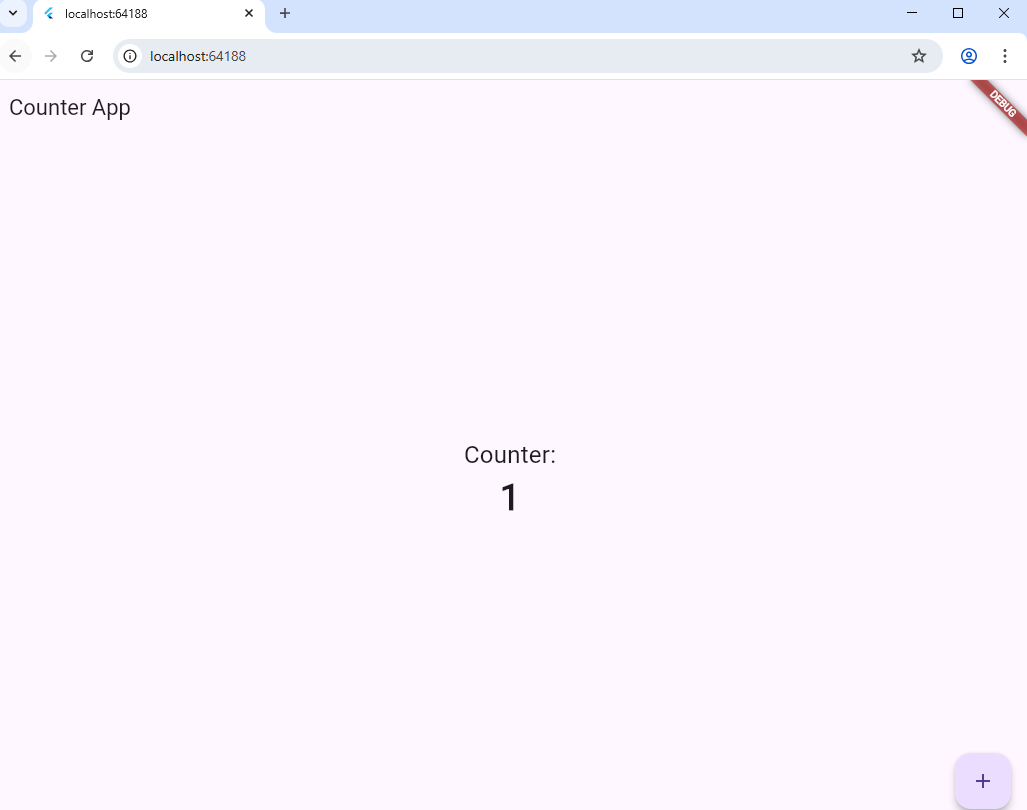
),

);

}

}

**OUTPUT:**



**Stateful widgets are composed of two classes:** the stateful widget itself (which extends

StatefulWidget) and its corresponding state class (which extends State). The state class is

responsible for maintaining the widget's mutable state and updating the UI accordingly

via the setState() method.

stateless widgets are static and immutable, while stateful widgets are dynamic and can

change over time by managing their internal state. Understanding the difference between

these two types of widgets is essential for designing and building efficient and

responsive Flutter UIs.

**State Management using setState():**

**PROGRAM:**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

home: CounterPage(),

);

}

}

class CounterPage extends StatefulWidget {

@override

\_CounterPageState createState() => \_CounterPageState();

}

class \_CounterPageState extends State<CounterPage> {

int \_counter = 0;

void \_incrementCounter() {

setState(() {

\_counter++;

});

}

@override

Widget build(BuildContext context) {

return

Scaffold(

appBar: AppBar(

title: Text('Counter Example (setState)'),

), body: Center(

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: <Widget>[

Text(

'Counter Value:',

), Text(

'$\_counter',

style: Theme.of(context).textTheme.headlineMedium,

),

],

),

),

floatingActionButton: FloatingActionButton(

onPressed: \_incrementCounter,

tooltip: 'Increment',

child: Icon(Icons.add),

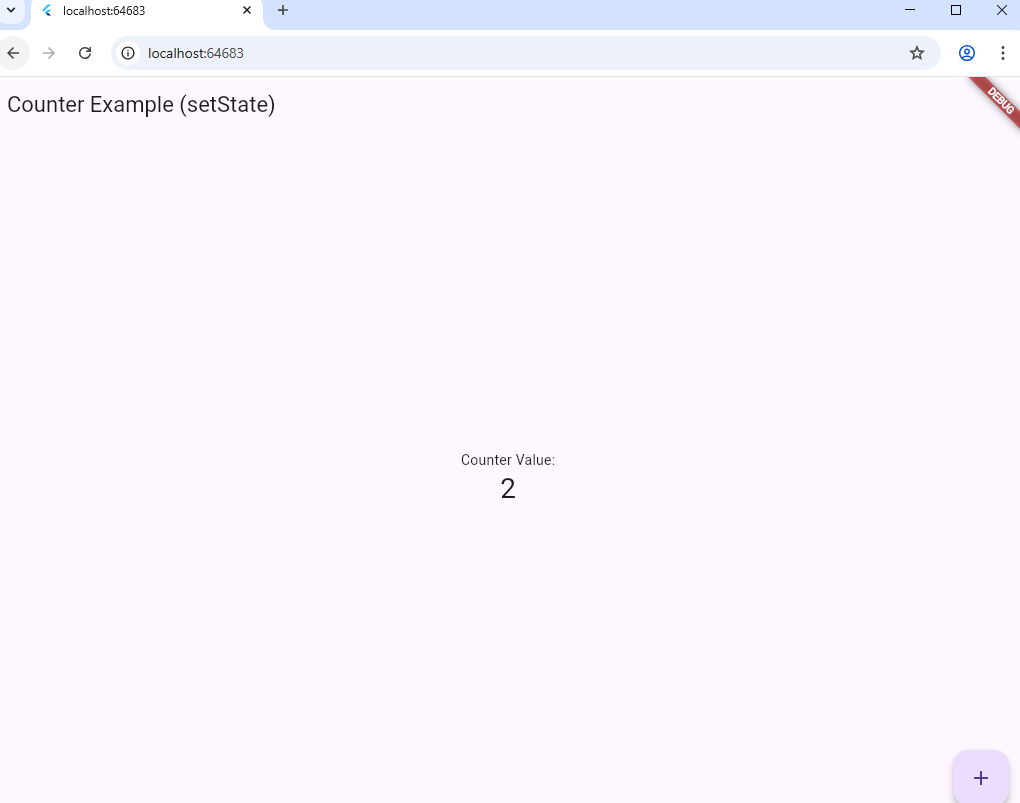
),

);

}

}

**OUTPUT:**



**State Management using setState():**

import 'package:flutter/material.dart';

void main() {

runApp(ShoppingApp());

}

class ShoppingApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

debugShowCheckedModeBanner: false,

home: ShoppingHomePage(),

);

}

}

class ShoppingHomePage extends StatefulWidget {

@override

\_ShoppingHomePageState createState() => \_ShoppingHomePageState();

}

class \_ShoppingHomePageState extends State<ShoppingHomePage> {

List<String> products = ["Product 1", "Product 2", "Product 3"];

List<String> cart = [];

void addToCart(String product) {

setState(() {

if (!cart.contains(product)) {

cart.add(product);

}

});

}

void viewCart() {

showDialog(

context: context,

builder: (BuildContext context) {

return AlertDialog(

title: Text("Shopping Cart"),

content: Column(

mainAxisSize: MainAxisSize.min,

children: cart.map((item) => Text(item)).toList(),

),

actions: [

TextButton(

child: Text("Close"),

onPressed: () {

Navigator.of(context).pop();

},

)

],

);

},

);

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text("Shopping App"),

backgroundColor: Colors.blue,

actions: [

Container(

margin: EdgeInsets.only(right: 10),

padding: EdgeInsets.all(5),

decoration: BoxDecoration(

color: Colors.red,

borderRadius: BorderRadius.circular(5),

),

child: Text(

"SALE",

style: TextStyle(color: Colors.white, fontWeight: FontWeight.bold),

),

)

],

),

body: ListView.builder(

itemCount: products.length,

itemBuilder: (context, index) {

return ListTile(

title: Text(products[index]),

trailing: IconButton(

icon: Icon(Icons.add\_shopping\_cart),

onPressed: () => addToCart(products[index]),

),

);

},

),

bottomNavigationBar: Padding(

padding: const EdgeInsets.all(10.0),

child: ElevatedButton.icon(

style: ElevatedButton.styleFrom(

backgroundColor: Colors.blue,

padding: EdgeInsets.symmetric(vertical: 15),

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(30),

),

),

icon: Icon(Icons.shopping\_cart),

label: Text("View Cart"),

onPressed: viewCart,

),

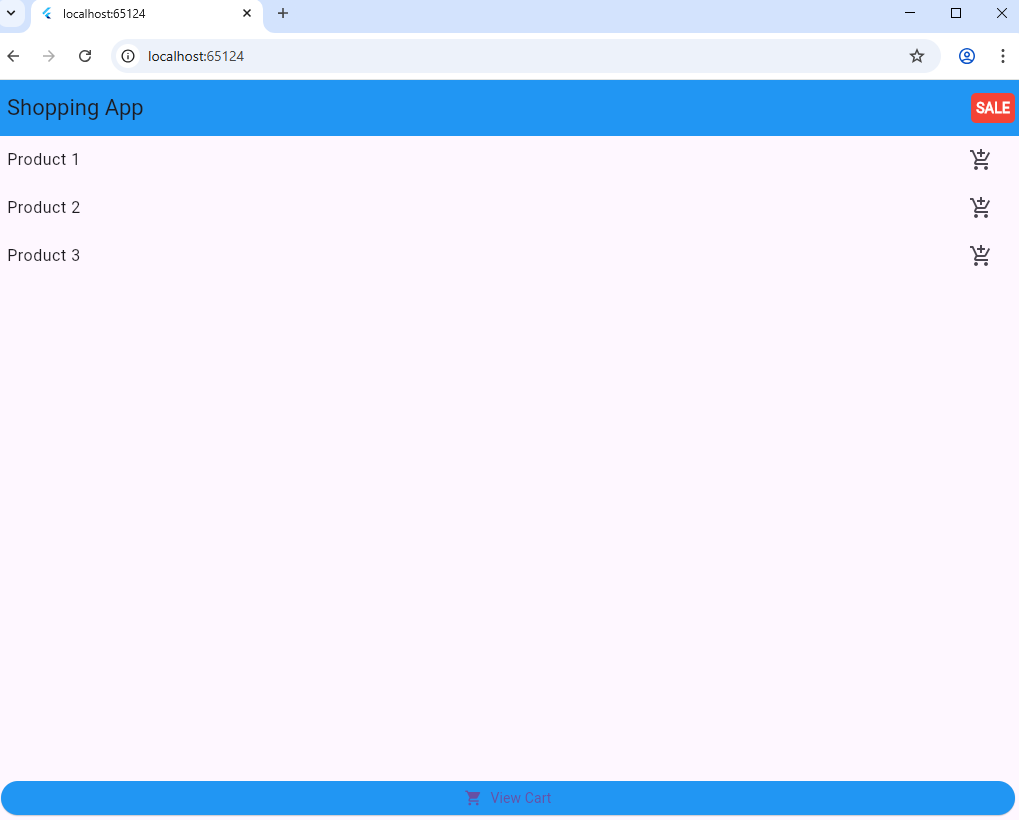
),

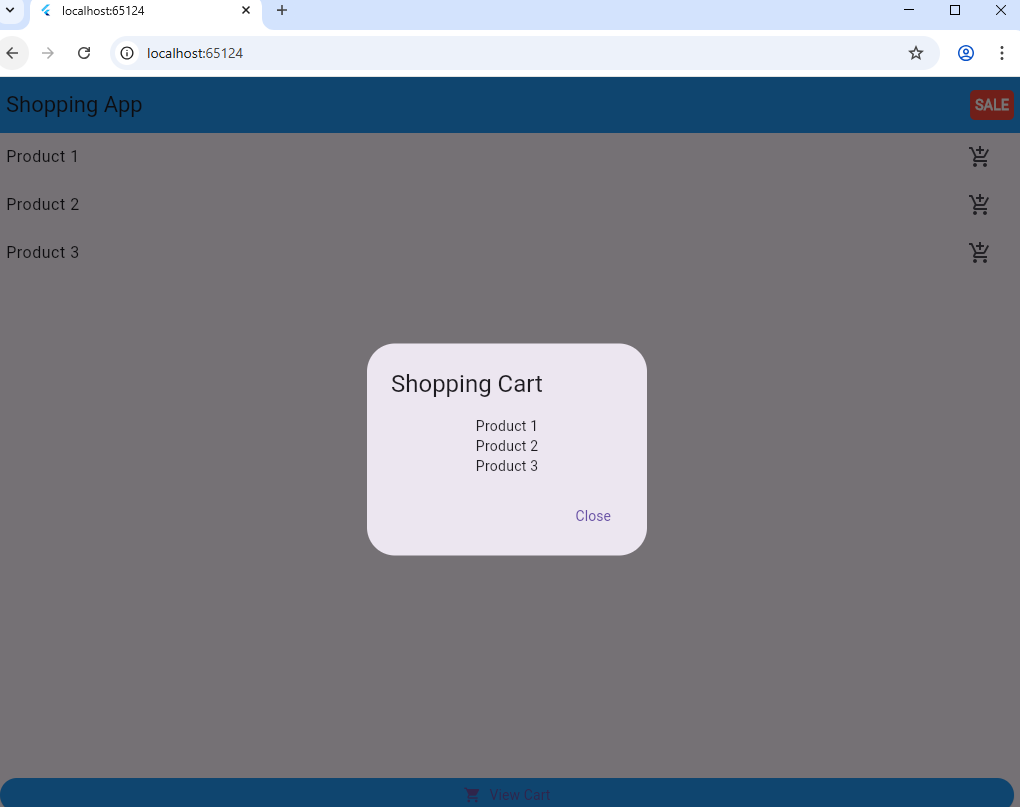
);

}

}

**OUTPUT:**





**State Management using provider package:**

**// main.dart**

import 'package:flutter/material.dart';

import'package:provider/provider.dart';

import 'provider/movie\_provider.dart';

import 'screens/home\_screen.dart';

void main() {

runApp(ChangeNotifierProvider<MovieProvider>( child: const MyApp(),

create: (\_) => MovieProvider(), // Create a new ChangeNotifier object

));

}

class MyApp extends

StatelessWidget {

const MyApp({Key? key}) : super(key: key);

@override

Widget build(BuildContext context) {

return MaterialApp(

// Remove the debug banner

debugShowCheckedModeBanner:

false, title: 'State Management usingprovider',

theme: ThemeData(

primarySwatch: Colors.indigo,

),

home: const HomeScreen(),

);

}

}

**create a movie folder and create file movie.dart**

class Movie {

final String title;

final String? runtime; // how long this movie is (in minute)

Movie({required this.title, this.runtime});

}

**Create a provider folder and create movie\_provider.dart inside the provider folder**

**// provider/movie\_provider.dart**

import 'package:flutter/material.dart';

import 'dart:math';

import '../movie/movie.dart';

// A list of movies

final List<Movie> initialData = List.generate(50,

(index) => Movie(

title: "Movie $index",

runtime: "${Random().nextInt(100) + 60} minutes"));

class MovieProvider with ChangeNotifier {

// All movies (that will be displayed on the Home screen)

final List<Movie> \_movies = initialData;

// Retrieve all movies

List<Movie> get movies => \_movies;

// Favorite movies (that will be shown on the MyList screen)

final List<Movie> \_myList = [];

// Retrieve favorite movies

List<Movie> get myList => \_myList;

// Adding a movie to the favorites

void addToList(Movie movie) {

\_myList.add(movie);

notifyListeners();

}

// Removing a movie from the favorites

void removeFromList(Movie movie) {

\_myList.remove(movie);

notifyListeners();

}

}

**Create a screens folder for screens Create home\_screen.dart for home screen page**

**// screens/home\_screen.dart**

import 'package:flutter/material.dart';

import 'package:provider/provider.dart';

import '../provider/movie\_provider.dart';

import 'my\_list\_screen.dart';

class HomeScreen extends StatefulWidget {

const HomeScreen({Key? key}) : super(key: key);

@override

State<HomeScreen> createState() => \_HomeScreenState();

}

class \_HomeScreenState extends

State<HomeScreen> {

@override

Widget build(BuildContext context) {

var movies = context.watch<MovieProvider>().movies;

var myList = context.watch<MovieProvider>().myList;

return

Scaffold(

appBar: AppBar(

title: const Text('State Management using provider'),

),

body:

Padding(

padding: const EdgeInsets.all(15),

child: Column(

crossAxisAlignment: CrossAxisAlignment.stretch,

children: [

ElevatedButton.icon( onPressed:

() { Navigator.of(context).push(

MaterialPageRoute(

builder: (context) => const MyListScreen(),

),

);

},

icon: const Icon(Icons.favorite),

label: Text(

"Go to my list (${myList.length})",

style: const TextStyle(fontSize: 24),

),

style: ElevatedButton.styleFrom(

backgroundColor: Colors.red,

padding: const EdgeInsets.symmetric(vertical: 20)),

),

const SizedBox(

height: 15,

), Expanded(

child: ListView.builder(

itemCount: movies.length,

itemBuilder: (\_, index) {

final currentMovie = movies[index];

return Card(

key: ValueKey(currentMovie.title), color:

Colors.amberAccent.shade100, elevation: 4,

child: ListTile(

title: Text(currentMovie.title),

subtitle:

Text(currentMovie.runtime ?? 'No information'), trailing:

IconButton(

icon: Icon( Icons.favorite,

color: myList.contains(currentMovie)

? Colors.red

: Colors.white,

size: 30,

),

onPressed: () {

if (!myList.contains(currentMovie)) {

context

.read<MovieProvider>()

.addToList(currentMovie);

} else {

context

.read<MovieProvider>()

.removeFromList(currentMovie);

}

},

),

),

);

}),

),

],

),

),

);

}

}

**create my\_list\_screen.dart inside the screens folder**

**// screens/my\_list\_screen.dart**

import 'package:flutter/material.dart';

import 'package:provider/provider.dart';

import '../provider/movie\_provider.dart';

class MyListScreen extends StatefulWidget {

const MyListScreen({Key? key}) : super(key:

key);

@override

State<MyListScreen> createState() => \_MyListScreenState();

}class \_MyListScreenState extends State<MyListScreen> {

@override

Widget build(BuildContext context) {

final myList = context.watch<MovieProvider>().myList;

return Scaffold(

appBar: AppBar(

title: Text("My List (${myList.length})"),

),

body: ListView.builder(

itemCount: myList.length,

itemBuilder: (\_, index) {

final currentMovie = myList[index];

return Card(

key: ValueKey(currentMovie.title),

elevation: 4,

child: ListTile(

title: Text(currentMovie.title),

subtitle: Text(currentMovie.runtime ?? ''),

trailing: TextButton(

child: const Text(

'Remove',

style: TextStyle(color: Colors.red),

),

onPressed: () {

context.read<MovieProvider>().removeFromList(currentMovie);

},

),

),

);

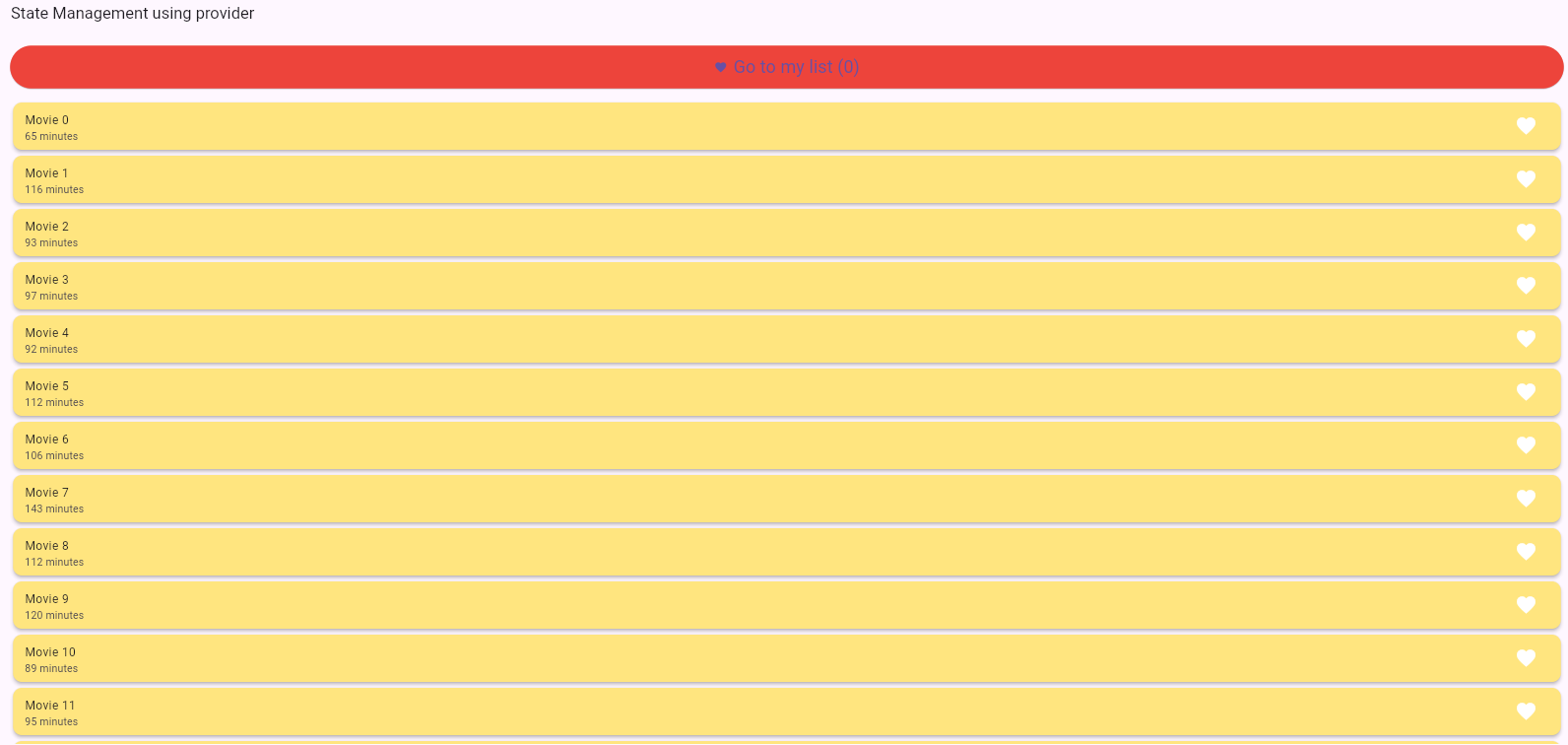
}),

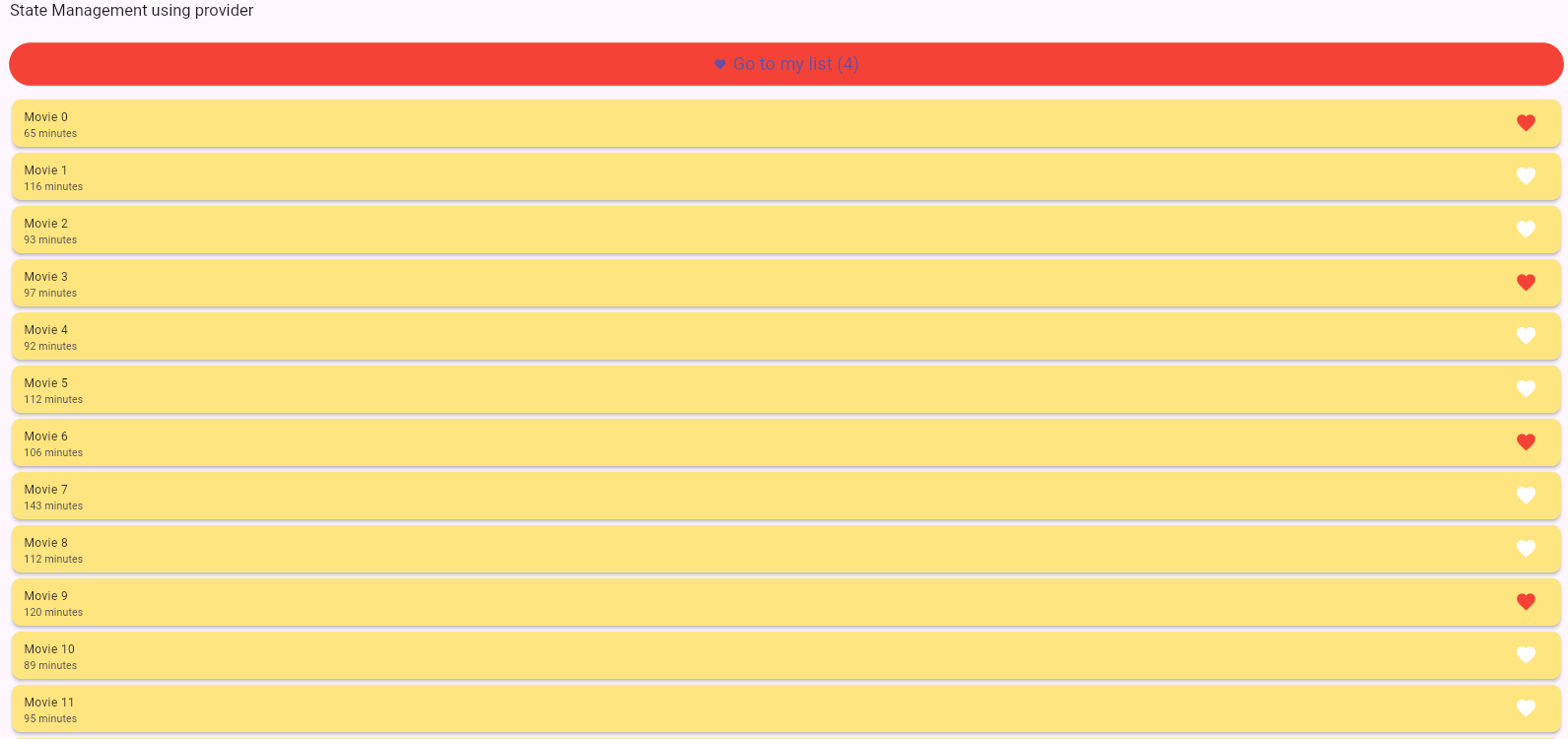
);

}

}

**OUTPUT:**







1. **a) Create custom widgets for specific UI elements**

**PROGRAM:**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp( home:

Scaffold( appBar: AppBar(

title: Text('Custom Widget Example'),

),

body: Column(

mainAxisAlignment: MainAxisAlignment.center, children:

<Widget>[

Padding(

padding: const EdgeInsets.all(8.0), child:

CustomTextField(

hintText: 'Enter your name', onChanged: (value) {

print('Name changed: $value');

},

),

),

SizedBox(height: 20), Padding(

padding: const EdgeInsets.all(8.0), child:

CustomTextField(

hintText: 'Enter Email', onChanged: (value) {

print('Name changed: $value');

},

),

),

SizedBox(height: 20), Padding(

padding: const EdgeInsets.all(8.0), child:

CustomTextField(

hintText: 'Enter Roll Number', onChanged: (value)

{ print('Name changed: $value');

},

),

),

SizedBox(height: 20), CustomButton(

text: 'Press Me',

onPressed: () {

print('Button pressed!');

},

),

],

),

),

);

}

}

class CustomButton extends StatelessWidget {

final String? text;

final VoidCallback? onPressed;

const CustomButton({ Key?

key,

@required this.text,

@required this.onPressed,

}) : super(key: key);

@override

Widget build(BuildContext context) {

return ElevatedButton( onPressed:

onPressed, child: Text(text!),

);

}

}

class CustomTextField extends StatelessWidget {

final String hintText;

final ValueChanged<String> onChanged;

const CustomTextField({ Key?

key,

required this.hintText, required

this.onChanged,

}) : super(key: key);

@override

Widget build(BuildContext context) {

return TextField( onChanged:

onChanged, decoration: InputDecoration(

hintText: hintText,

border: OutlineInputBorder(),

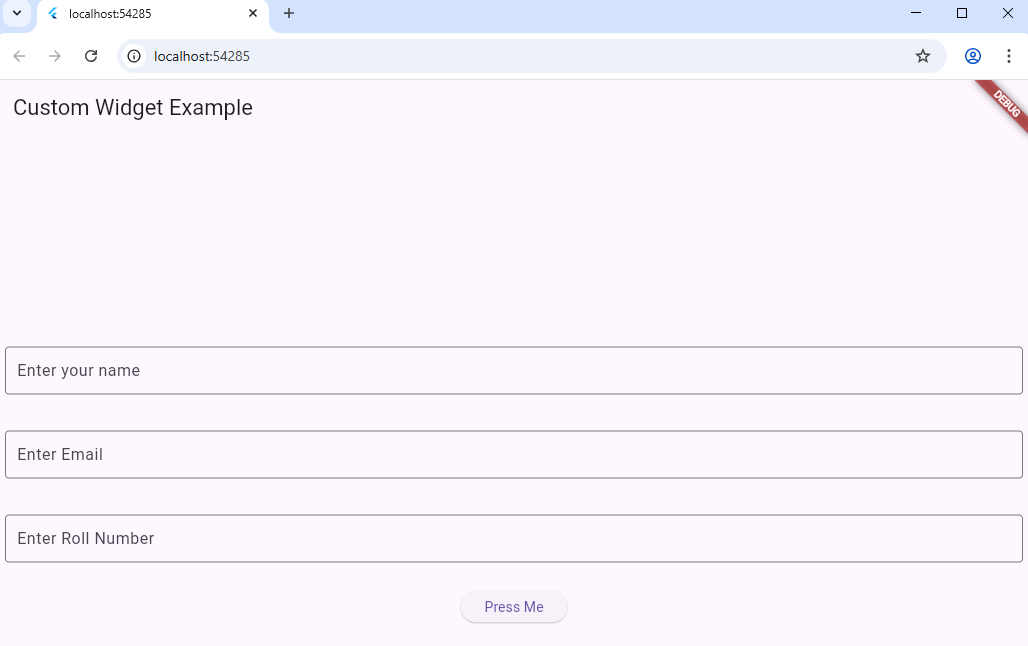
),

);

}

}

**OUTPUT:**



**6. b) Apply styling using themes and custom styles**

In Flutter, you can apply styling to your widgets using themes and custom styles to maintain consistency and make your UI more visually appealing.

**PROGRAM:**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

theme: ThemeData(

primaryColor: Colors.blue,

colorScheme: ColorScheme.fromSwatch().copyWith(secondary: Colors.orange),

fontFamily: 'Roboto',

textTheme: TextTheme(

displayLarge: TextStyle(fontSize: 24, fontWeight: FontWeight.bold), // for headings

bodyLarge: TextStyle(fontSize: 16), // for body text

),

elevatedButtonTheme: ElevatedButtonThemeData(

style: ElevatedButton.styleFrom(

backgroundColor: Colors.blue,

foregroundColor: Colors.white,

textStyle: TextStyle(fontSize: 18),

padding: EdgeInsets.symmetric(horizontal: 20, vertical: 15),

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(10),

),

),

),

),

home: HomePage(),

);

}

}

class HomePage extends StatelessWidget {

@override

Widget build(BuildContext context) {

return

Scaffold(

appBar: AppBar(

title: Text('Styling Example'),

),

body: Center(

child: Column(

mainAxisAlignment:

MainAxisAlignment.center, children:

<Widget>[

Text(

'Welcome to MyApp',

style: Theme.of(context).textTheme.displayLarge,

),

SizedBox(

height:

20),

ElevatedButton(

onPressed: () {

// Action here

},

style: ElevatedButton.styleFrom(

backgroundColor: Colors.blue,

foregroundColor: Colors.white,

padding: EdgeInsets.symmetric(horizontal: 20, vertical: 15),

textStyle: TextStyle(fontSize: 18),

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(10),

),

),

child: Text('Get Started'),

),

],

),

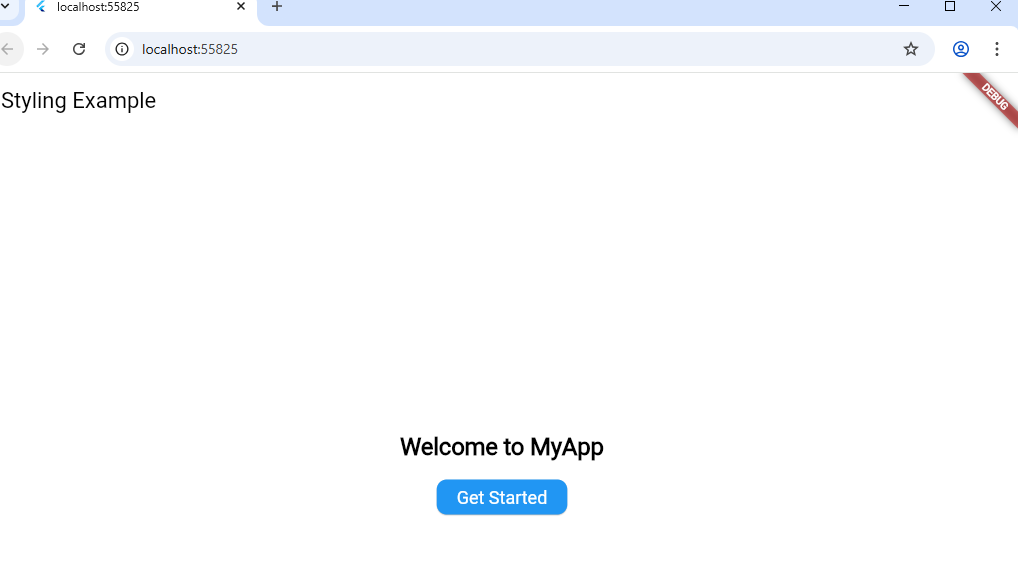
),

);

}

}

**OUTPUT:**



1. **a) Design a form with various input fields.**

Form with various input fields such as text fields, checkboxes, radio buttons, and a dropdown menu.

**PROGRAM:**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'Form Example',

theme: ThemeData(

primarySwatch: Colors.blue,

),

home: FormPage(),

);

}

}

class FormPage extends StatefulWidget {

@override

\_FormPageState createState() => \_FormPageState();

}

class \_FormPageState extends State<FormPage> {

final \_formKey = GlobalKey<FormState>();

String? \_name;

String? \_email;

bool \_subscribeToNewsletter = false;

String \_selectedCountry = 'USA';

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Form Example'),

),

body: Padding(

padding: EdgeInsets.all(20.0),

child: SingleChildScrollView(

child: Form(

key: \_formKey,

child: Column(

crossAxisAlignment: CrossAxisAlignment.start,

children: <Widget>[

// Name TextFormField

TextFormField(

decoration: InputDecoration(labelText: 'Name'),

onSaved: (value) {

\_name = value ?? '';

},

),

SizedBox(height: 20),

// Email TextFormField

TextFormField(

decoration: InputDecoration(labelText: 'Email'),

onSaved: (value) {

\_email = value ?? '';

},

),

SizedBox(height: 20),

// Subscribe Checkbox

Row(

children: <Widget>[

Checkbox(

value: \_subscribeToNewsletter,

onChanged: (bool? value) {

setState(() {

\_subscribeToNewsletter = value ?? false;

});

},

),

Text('Subscribe to Newsletter'),

],

),

SizedBox(height: 20),

// Country Dropdown

Row(

children: <Widget>[

Text('Country: '),

SizedBox(width: 20),

DropdownButton<String>(

value: \_selectedCountry,

onChanged: (String? value) {

setState(() {

\_selectedCountry = value ?? 'USA';

});

},

items: <String>['USA', 'Canada', 'UK', 'Australia']

.map<DropdownMenuItem<String>>((String value) {

return DropdownMenuItem<String>(

value: value,

child: Text(value),

);

}).toList(),

),

],

),

SizedBox(height: 20),

// Submit Button

ElevatedButton(

onPressed: () {

\_formKey.currentState!.save(); // Save form data

print('Name: $\_name');

print('Email: $\_email');

print('Subscribe to Newsletter: $\_subscribeToNewsletter');

print('Country: $\_selectedCountry');

},

child: Text('Submit'),

),

],

),

),

),

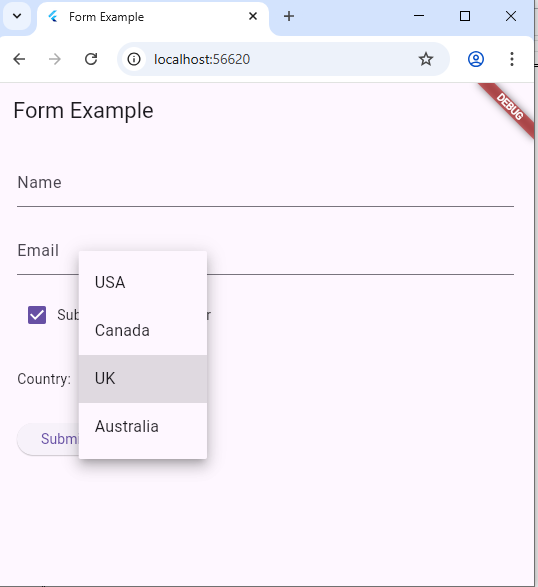
),

);

}

}

**OUTPUT:**



1. **b) Implement form validation and error handling.**

**PROGRAM:**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'Form Example',

home: Scaffold(

appBar: AppBar(

title: Text('Form Example'),

),

body: SingleChildScrollView(

padding: EdgeInsets.all(16),

child: FormWidget(),

),

),

);

}

}

class FormWidget extends StatefulWidget {

@override

\_FormWidgetState createState() => \_FormWidgetState();

}

class \_FormWidgetState extends State<FormWidget> {

final \_formKey = GlobalKey<FormState>();

String? \_name;

String? \_email;

String? \_password;

String? \_phone;

String? \_address;

@override

Widget build(BuildContext context) {

return Form(

key: \_formKey,

child: Column(

crossAxisAlignment: CrossAxisAlignment.start,

children: <Widget>[

// Name field

TextFormField(

decoration: InputDecoration(labelText: 'Name'),

validator: (value) {

if (value == null || value.isEmpty) {

return 'Please enter your name';

}

return null;

},

onSaved: (value) => \_name = value,

),

SizedBox(height: 16),

// Email field

TextFormField(

decoration: InputDecoration(labelText: 'Email'),

keyboardType: TextInputType.emailAddress,

validator: (value) {

if (value == null || value.isEmpty) {

return 'Please enter your email';

}

if (!RegExp(r'\S+@\S+\.\S+').hasMatch(value)) {

return 'Please enter a valid email';

}

return null;

},

onSaved: (value) => \_email = value,

),

SizedBox(height: 16),

// Password field

TextFormField(

decoration: InputDecoration(labelText: 'Password'),

obscureText: true,

validator: (value) {

if (value == null || value.isEmpty) {

return 'Please enter a password';

}

if (value.length < 6) {

return 'Password must be at least 6 characters';

}

return null;

},

onSaved: (value) => \_password = value,

),

SizedBox(height: 16),

// Phone field

TextFormField(

decoration: InputDecoration(labelText: 'Phone'),

keyboardType: TextInputType.phone,

validator: (value) {

if (value == null || value.isEmpty) {

return 'Please enter your phone number';

}

return null;

},

onSaved: (value) => \_phone = value,

),

SizedBox(height: 16),

// Address field

TextFormField(

decoration: InputDecoration(labelText: 'Address'),

maxLines: 3,

validator: (value) {

if (value == null || value.isEmpty) {

return 'Please enter your address';

}

return null;

},

onSaved: (value) => \_address = value,

),

SizedBox(height: 16),

// Submit button

ElevatedButton(

onPressed: \_submitForm,

child: Text('Submit'),

),

],

),

);

}

void \_submitForm() {

if (\_formKey.currentState!.validate()) {

\_formKey.currentState!.save();

// Print form data

print('Form submitted:');

print('Name: $\_name');

print('Email: $\_email');

print('Password: $\_password');

print('Phone: $\_phone');

print('Address: $\_address');

// Optional: Show confirmation on screen

ScaffoldMessenger.of(context).showSnackBar(

SnackBar(content: Text('Form submitted successfully!')),

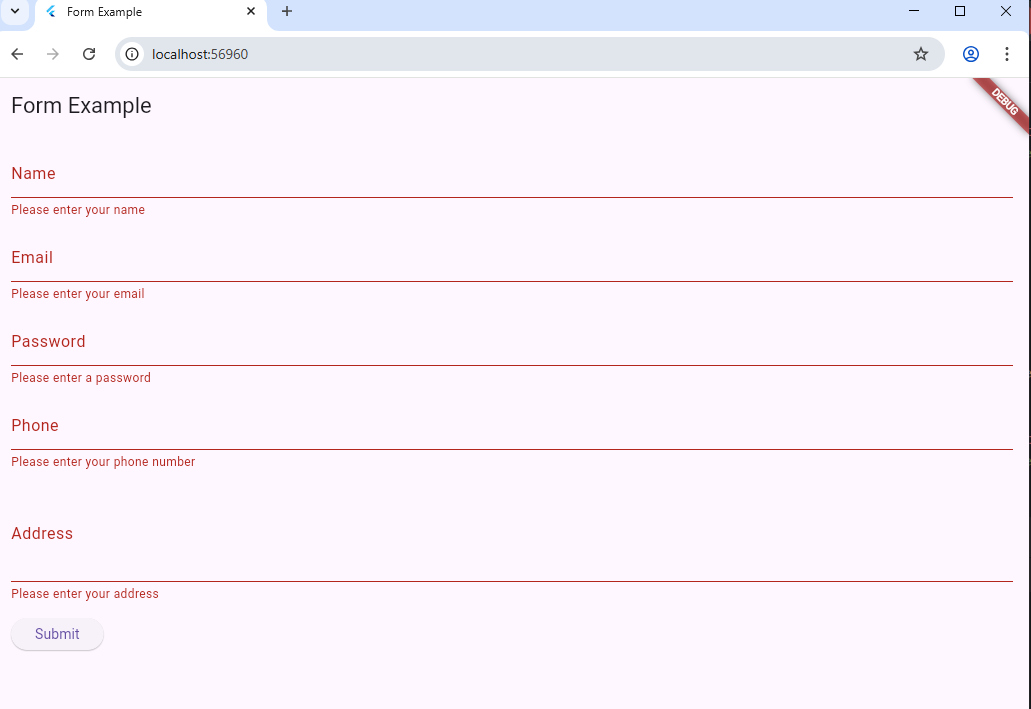
);

}

}

}

**OUTPUT:**



1. **a) Add animations to UI elements using flutter's animation framework.**

**PROGRAM:**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

home: Scaffold(

appBar: AppBar(

title: Text('Animation Example'),

),

body: AnimationWidget(),

),

);

}

}

class AnimationWidget extends StatefulWidget {

@override

\_AnimationWidgetState createState() => \_AnimationWidgetState();

}

class \_AnimationWidgetState extends State<AnimationWidget>

with SingleTickerProviderStateMixin {

late AnimationController \_controller;

late Animation<double> \_animation;

@override

void initState() {

super.initState();

\_controller = AnimationController(

duration: Duration(seconds: 1),

vsync: this,

);

\_animation = Tween<double>(begin: 0, end: 300).animate(\_controller)

..addListener(() {

setState(() {}); // Trigger rebuild when animation value changes

});

}

@override

Widget build(BuildContext context) {

return Center(

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: <Widget>[

Container(

width: \_animation.value,

height: \_animation.value,

color: Colors.blue,

child: FlutterLogo(size: 100),

),

SizedBox(height: 20),

ElevatedButton(

onPressed: () {

if (\_controller.status == AnimationStatus.completed) {

\_controller.reverse();

} else {

\_controller.forward();

}

},

child: Text(

\_controller.status == AnimationStatus.completed

? 'Reverse Animation'

: 'Start Animation',

),

),

],

),

);

}

@override

void dispose() {

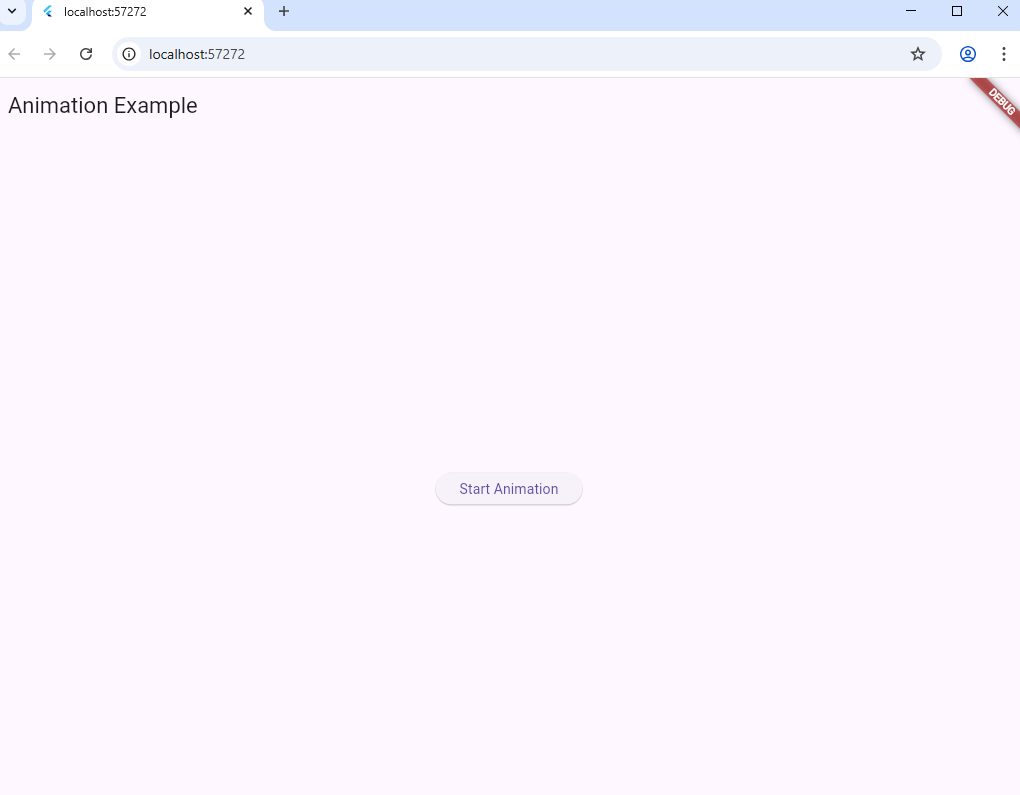
\_controller.dispose();

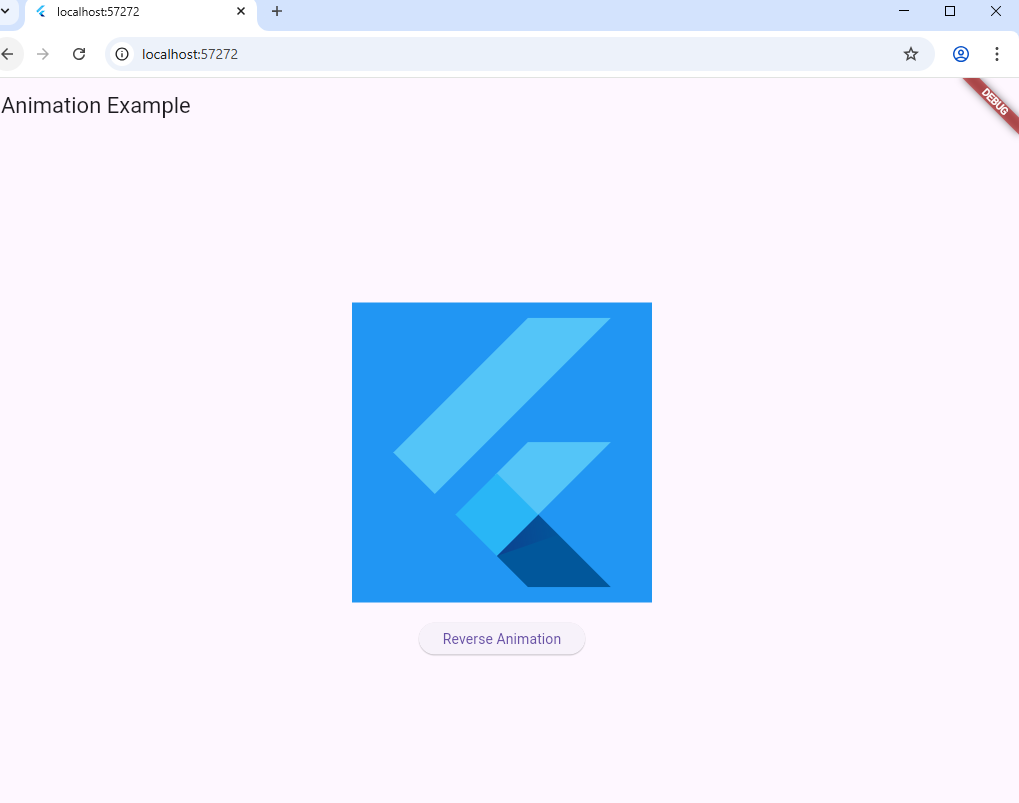
super.dispose();

}

}

**OUTPUT:**





**8. b) Experiment with different types of animations like fade,slide,etc.**

**PROGRAM:**

import 'package:flutter/material.dart';

void main() {

runApp(MyApp());

}

enum AnimationType { fade, slide, scale }

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'Combined Animations',

home: AnimationHome(),

);

}

}

class AnimationHome extends StatefulWidget {

@override

\_AnimationHomeState createState() => \_AnimationHomeState();

}

class \_AnimationHomeState extends State<AnimationHome>

with SingleTickerProviderStateMixin {

late AnimationController \_controller;

late Animation<double> \_fadeAnimation;

late Animation<Offset> \_slideAnimation;

late Animation<double> \_scaleAnimation;

AnimationType \_currentAnimation = AnimationType.fade;

@override

void initState() {

super.initState();

\_controller = AnimationController(

duration: Duration(seconds: 2),

vsync: this,

);

\_fadeAnimation = Tween<double>(begin: 0.0, end: 1.0).animate(\_controller);

\_slideAnimation =

Tween<Offset>(begin: Offset(-1, 0), end: Offset(0, 0)).animate(\_controller);

\_scaleAnimation = Tween<double>(begin: 0.0, end: 1.0).animate(\_controller);

\_controller.forward();

}

void \_changeAnimation(AnimationType type) {

setState(() {

\_currentAnimation = type;

\_controller.reset();

\_controller.forward();

});

}

Widget \_buildAnimatedWidget() {

final container = Container(

width: 200,

height: 200,

color: Colors.blue,

child: FlutterLogo(size: 100),

);

switch (\_currentAnimation) {

case AnimationType.fade:

return FadeTransition(opacity: \_fadeAnimation, child: container);

case AnimationType.slide:

return SlideTransition(position: \_slideAnimation, child: container);

case AnimationType.scale:

return ScaleTransition(scale: \_scaleAnimation, child: container);

default:

return container;

}

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(title: Text('Combined Animations')),

body: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: [

Center(child: \_buildAnimatedWidget()),

SizedBox(height: 40),

Wrap(

spacing: 10,

children: [

ElevatedButton(

onPressed: () => \_changeAnimation(AnimationType.fade),

child: Text('Fade'),

),

ElevatedButton(

onPressed: () => \_changeAnimation(AnimationType.slide),

child: Text('Slide'),

),

ElevatedButton(

onPressed: () => \_changeAnimation(AnimationType.scale),

child: Text('Scale'),

),

],

),

],

),

);

}

@override

void dispose() {

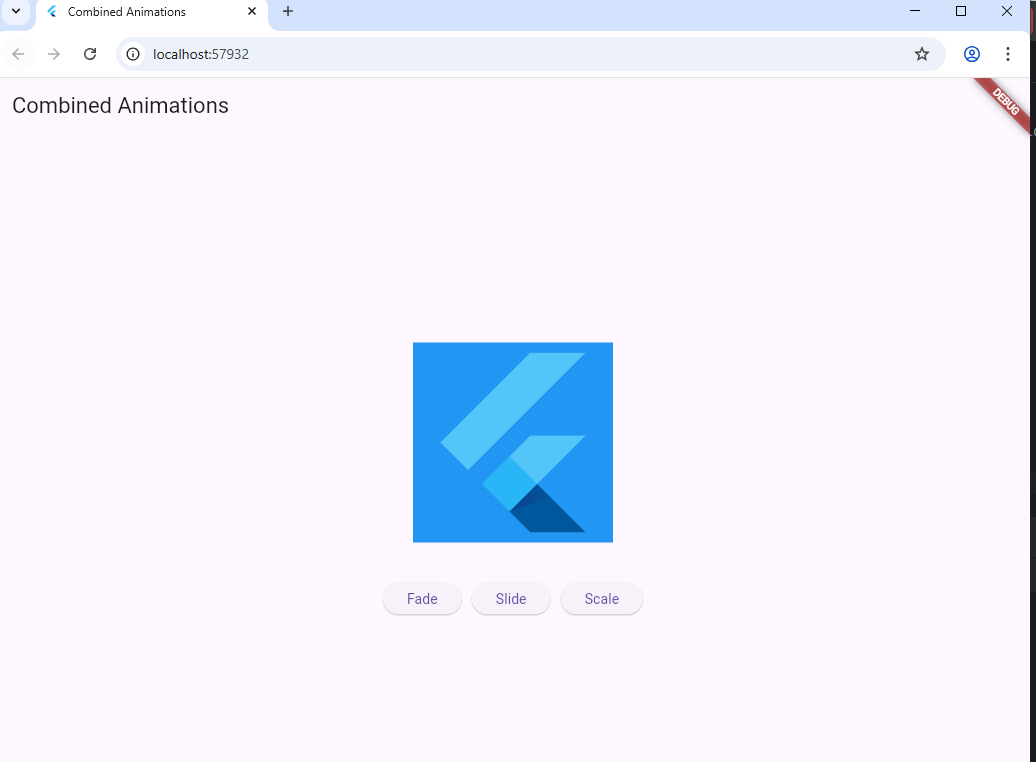
\_controller.dispose();

super.dispose();

}

}

**OUTPUT:**



1. **a) Fetch data from REST API**

add

**dependancy in pubspec.yaml:**

dependencies:

flutter:

sdk: flutter

http: ^1.1.0

**In your terminal, run:**

flutter pub get

**PROGRAM:**

import 'dart:convert';

import 'package:flutter/material.dart';

import 'package:http/http.dart' as http;

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'API Data Example',

home: HomePage(),

);

}

}

class HomePage extends StatefulWidget {

@override

\_HomePageState createState() => \_HomePageState();

}

class \_HomePageState extends State<HomePage> {

List<dynamic> \_data = [];

bool \_isLoading = true;

String? \_error;

@override

void initState() {

super.initState();

\_fetchDataFromApi();

}

Future<void> \_fetchDataFromApi() async {

try {

final response = await http

.get(Uri.parse('https://jsonplaceholder.typicode.com/posts'));

if (response.statusCode == 200) {

setState(() {

\_data = json.decode(response.body);

\_isLoading = false;

});

} else {

setState(() {

\_error = 'Failed to load data';

\_isLoading = false;

});

}

} catch (e) {

setState(() {

\_error = e.toString();

\_isLoading = false;

});

}

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('API Data Example'),

),

body: \_isLoading

? Center(child: CircularProgressIndicator())

: \_error != null

? Center(child: Text('Error: $\_error'))

: ListView.builder(

itemCount: \_data.length,

itemBuilder: (context, index) {

return ListTile(

title: Text(\_data[index]['title'] ?? ''),

subtitle: Text(\_data[index]['body'] ?? ''),

);

},

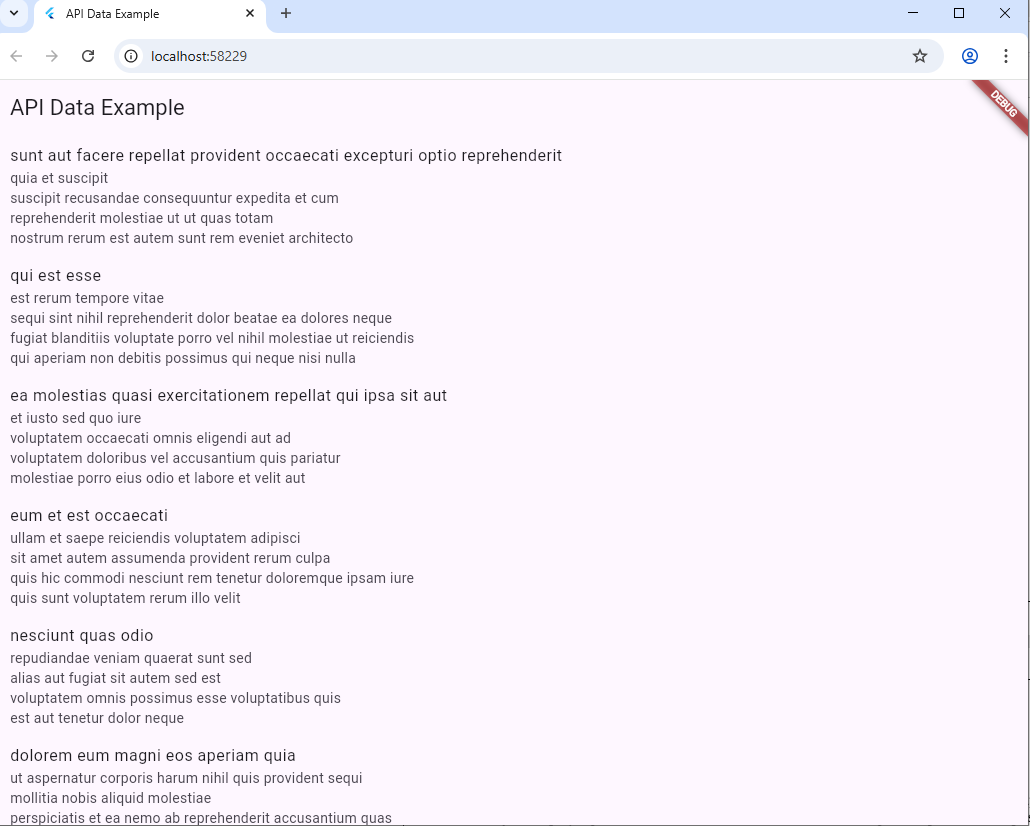
),

);

}

}

**OUTPUT:**



**9. b) Display the fetched data in a meaningful way in the UI.**

Display the fetched data in a meaningful way in the UI, we can use a more structured layout rather than just displaying the data in a list. We'll create a custom widget to represent each post fetched from the API, and display them in a scrollable list.

**PROGRAM:**

import 'dart:convert';

import 'package:flutter/material.dart';

import 'package:http/http.dart' as http;

void main() {

runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'API Data Example',

home: HomePage(),

);

}

}

class HomePage extends StatefulWidget {

@override

\_HomePageState createState() => \_HomePageState();

}

class \_HomePageState extends State<HomePage> {

List<dynamic> \_data = [];

bool \_isLoading = false;

@override

void initState() {

super.initState();

\_fetchDataFromApi();

}

Future<void> \_fetchDataFromApi() async {

setState(() {

\_isLoading = true;

});

try {

final response = await http

.get(Uri.parse('https://jsonplaceholder.typicode.com/posts'));

if (response.statusCode == 200) {

setState(() {

\_data = json.decode(response.body);

\_isLoading = false;

});

} else {

throw Exception('Failed to load data');

}

} catch (e) {

setState(() {

\_isLoading = false;

});

print('Error: $e');

}

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('API Data Example'),

),

body: \_isLoading

? Center(child: CircularProgressIndicator())

: ListView.builder(

itemCount: \_data.length,

itemBuilder: (context, index) {

return PostCard(

title: \_data[index]['title'] ?? '',

body: \_data[index]['body'] ?? '',

);

},

),

);

}

}

class PostCard extends StatelessWidget {

final String title;

final String body;

const PostCard({

Key? key,

required this.title,

required this.body,

}) : super(key: key);

@override

Widget build(BuildContext context) {

return Card(

margin: EdgeInsets.symmetric(horizontal: 16, vertical: 8),

child: Padding(

padding: EdgeInsets.all(16),

child: Column(

crossAxisAlignment: CrossAxisAlignment.start,

children: [

Text(

title,

style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),

),

SizedBox(height: 8),

Text(

body,

style: TextStyle(fontSize: 16),

),

],

),

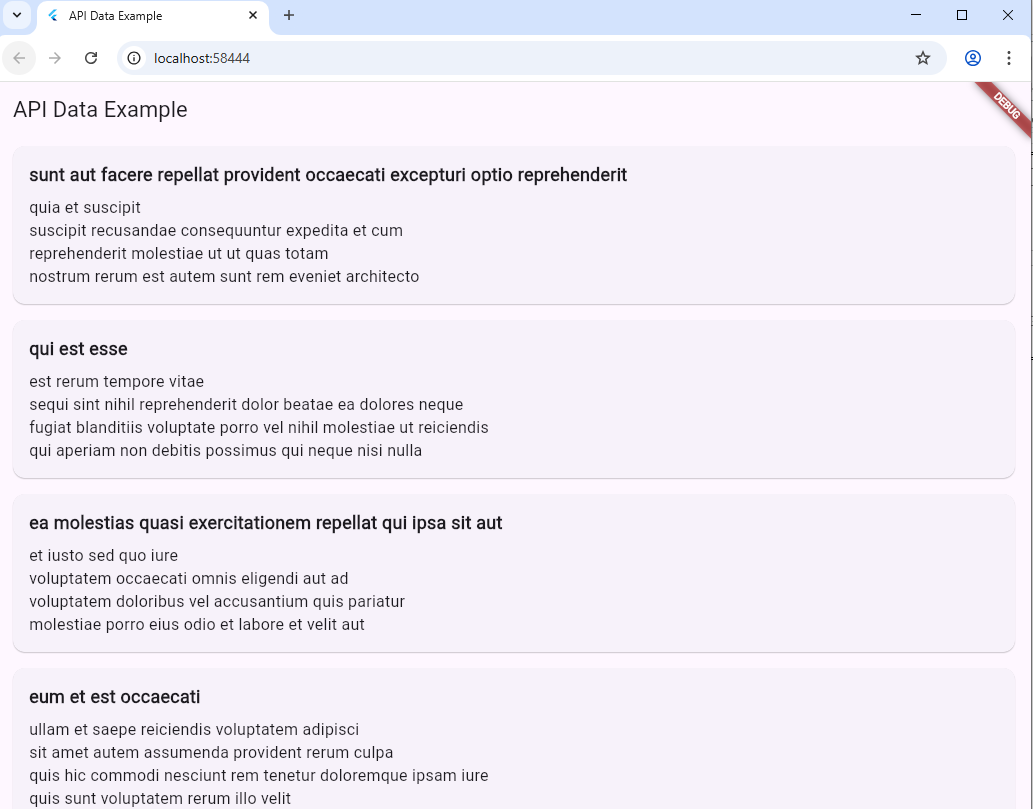
),

);

}

}

**OUTPUT:**



We've added a loading indicator (CircularProgressIndicator) to indicate when data is being fetched. The fetched data is displayed as a list of PostCard widgets, each representing a post from the API. The PostCard widget displays the title and body of each post in a structured manner using a Card layout

int addNumbers(int a, int b) {  
 return a + b;  
}  
  
void main() {  
 // 1. Print your name  
 String name = "Chandini"; // you can replace with your own name  
 print("My name is $name");  
  
 // 2. Check age with conditionals  
 int age = 22; // change value to test  
 if (age >= 18) {  
 print("You are an adult.");  
 } else {  
 print("You are a minor.");  
 }  
  
 // 3. Loop to count from 1 to 5  
 print("Counting from 1 to 5:");  
 for (int i = 1; i <= 5; i++) {  
 print(i);  
 }  
  
 // 4. Use the sum function  
 int x = 10, y = 20;  
 int result = addNumbers(x, y);  
 print("The sum of $x and $y is $result");  
}