120A3051
Shreya Idate
Batch: E3

Experiment No: 6

AIM: To apply navigation, routing and gestures in Flutter App

THEORY:

Flutter Drawer

The mobile apps that use Material Design have two primary options for navigation. These navigations are Tabs and Drawers. A drawer is an alternative option for tabs because sometimes the mobile apps do nothave sufficient space to support tabs.

A drawer is an invisible side screen. It is a sliding left menu that generally contains important links in the application and occupies half of the screen when displayed.

Flutter uses a drawer widget to create a sliding left menu layout with a Material Design widget. The following steps are required to use a drawer in the app.

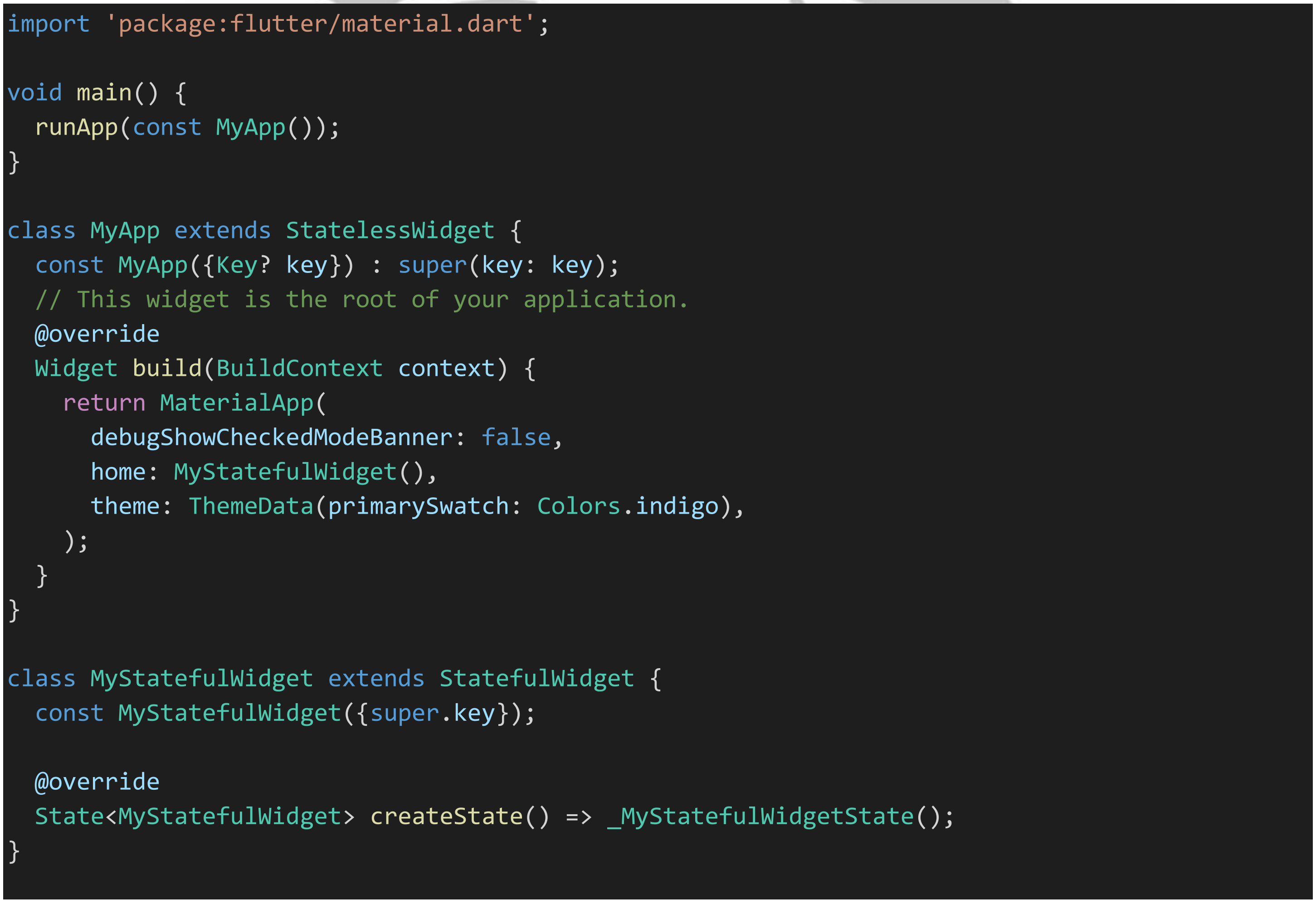
1. Create a Flutter Project.
2. Add drawer in scaffold widget
3. Populate the drawer by adding content
4. Close the drawer.

In the main.dart file, create a drawer in the scaffold widget as the code given below.

```
Scaffold(
  drawer: Drawer(
    child: // Populate the Drawer by adding content in the next step.
  )
);
```

Next, we need to add content in the drawer. In this example, we are going to use the Listview widget that allows the users to scroll through the drawer if the content does not fit in the screen supports. The following code explains it more clearly.

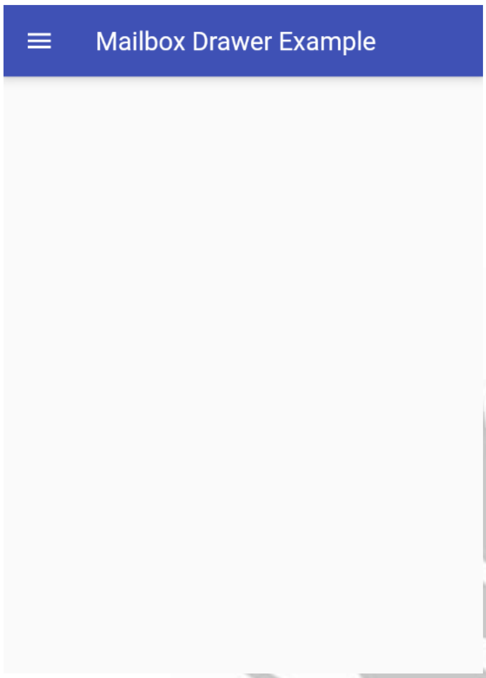
```
Drawer(
  child: ListView(
    padding:
    EdgeInsets.zero,
    children: <Widget>[
    DrawerHeader(
        child: Text('Drawer Header'),
        decoration:  BoxDecoration(
        color: Colors.blue,          ),
                                        ),
          ListTile(
            title: Text('Item
            1'),onTap: () {
              // Update the state of the app.
              // ...
            },
          ),
          ListTile(
            title: Text(
            onTap: ()
              // Updat
              // ...
            },
          ),
        ],
      ),
    );
```

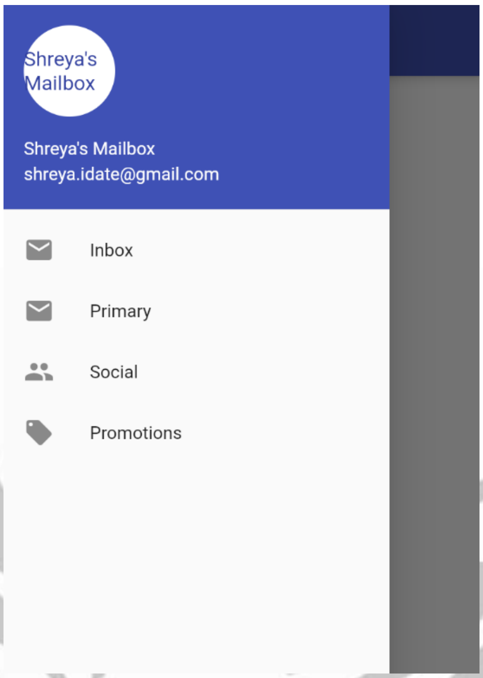


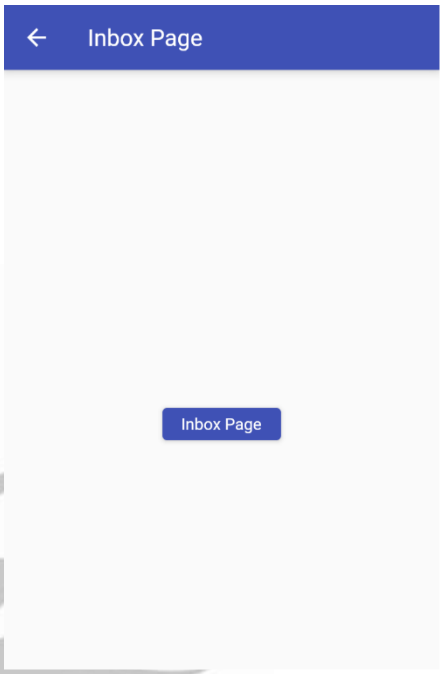
```
import 'package:flutter/material.dart';

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

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);
  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      home: MyStatefulWidget(),
      theme: ThemeData(primarySwatch: Colors.indigo),
    );
  }
}

class MyStatefulWidget extends StatefulWidget {
  const MyStatefulWidget({super.key});

  @override
  State<MyStatefulWidget> createState() => _MyStatefulWidgetState();
}
```

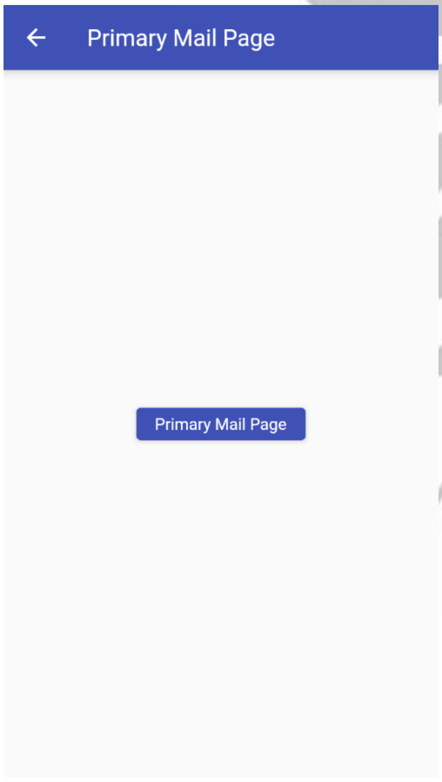
```dart
class _MyStatefulWidgetState extends State<MyStatefulWidget> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text(
          "Mailbox Drawer Example",
          style: TextStyle(color: Colors.white),
        ),
      ),
      drawer: Drawer(
        elevation: 20.0,
        child: Column(children: <Widget>[
          const UserAccountsDrawerHeader(
            accountName: Text(
              "Shreya's Mailbox",
              style: TextStyle(color: Colors.white),
            ),
            accountEmail: Text("shreya.idate@gmail.com"),
            currentAccountPicture: CircleAvatar(
              backgroundColor: Colors.white,
              child: Text("Shreya's Mailbox"),
            ),
          ),
          ListTile(
              title: const Text("Inbox"),
              leading: const Icon(Icons.mail),
              onTap: () {
                Navigator.pop(context);

                Navigator.push(
                  context,
                  MaterialPageRoute(builder: (context) => Mailpage()),
                );
              }),
          ListTile(
              title: const Text("Primary"),
              leading: const Icon(Icons.mail),
              onTap: () {
                Navigator.pop(context);

                Navigator.push(
                  context,
                  MaterialPageRoute(builder: (context) => Primarypage()),
                );
              }),
          ListTile(
              title: const Text("Social"),
              leading: const Icon(Icons.people),
              onTap: () {
                Navigator.pop(context);
                Navigator.push(
         context,
         MaterialPageRoute(builder: (context) => Socialpage()),
        );
       }),
     ListTile(
       title: const Text("Promotions"),
       leading: const Icon(Icons.local_offer),
       onTap: () {
        Navigator.pop(context);

        Navigator.push(
         context,
         MaterialPageRoute(builder: (context) => Promotionpage()),
        );
       }),
    ]),
   ),
  );
 }
}

class Mailpage extends StatelessWidget {
 const Mailpage({super.key});

 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    title: Text("Inbox Page"),
   ),
   body: Center(
     child: ElevatedButton(
     onPressed: () {
     Navigator.pop(context);
    },
    child: Text("Inbox Page"),
   )),
  );
 }
}

class Primarypage extends StatelessWidget {
 const Primarypage({super.key});

 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    title: Text("Primary Mail Page"),
   ),
   body: Center(
     child: ElevatedButton(
          onPressed: () {
          Navigator.pop(context);
        },
        child: Text("Primary Mail Page"),
      )),
    );
  }
}

class Socialpage extends StatelessWidget {
  const Socialpage({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("Social Mail Page"),
      ),
      body: Center(
          child: ElevatedButton(
          onPressed: () {
          Navigator.pop(context);
        },
        child: Text("Social Mail Page"),
      )),
    );
  }
}

class Promotionpage extends StatelessWidget {
  const Promotionpage({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("Promotion Mail Page"),
      ),
      body: Center(
          child: ElevatedButton(
          onPressed: () {
          Navigator.pop(context);
        },
        child: Text("Promotion Mail Page"),
      )),
    );
  }
}
```

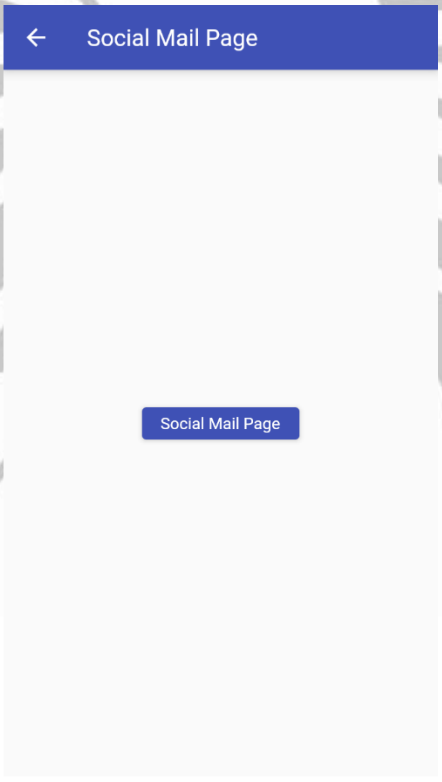
Demonstration of Navigation Drawer using Scaffold and List Tile Widget:

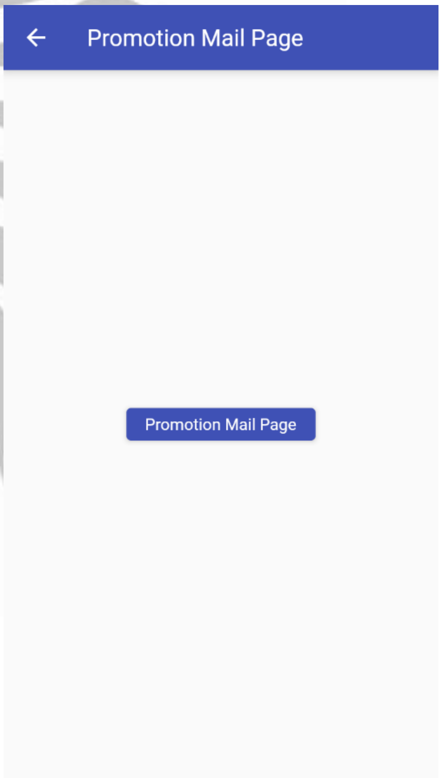












Bottom Navigation Bar:

The bottom navigation bar in Flutter can contain multiple items such as text labels, icons, or both. It allows the user to navigate between the top-level views of an app quickly. If we are using a larger screen, it is betterto use a side navigation bar.

In Flutter application, we usually set the bottom navigation bar in conjunction with the scaffold widget. Scaffold widget provides a Scaffold.bottomNavigationBar argument to set the bottom navigation bar. It is tonote that only adding BottomNavigationBar will not display the navigation items. It is required to set the BottomNavigationItems for Items property that accepts a list of BottomNavigationItems widgets.

- We can display only a small number of widgets in the bottom navigation that can be 2 to 5.
- It must have at least two bottom navigation items. Otherwise, we will get an error.
- It is required to have the icon and title properties, and we need to set relevant widgets for

them.Properties of the BottomNavigationBar Widget

The following are the properties used with the bottom navigation bar widget:

items: It defines the list to display within the bottom navigation bar. It uses argument BottomNavigationBarItem that contains sup-properties given below:

```
const
   BottomNavigationBarItem({
   @required this.icon,
   this.title,
   Widget activeIcon,
   this.backgroundCol
   or,
 })
```

currentIndex: It determines the current active bottom navigation bar item on the

screen.onTap: It is called when we tapped one of the items on the screen.

iconSize: It is used to specify the size of all bottom navigation item icons.

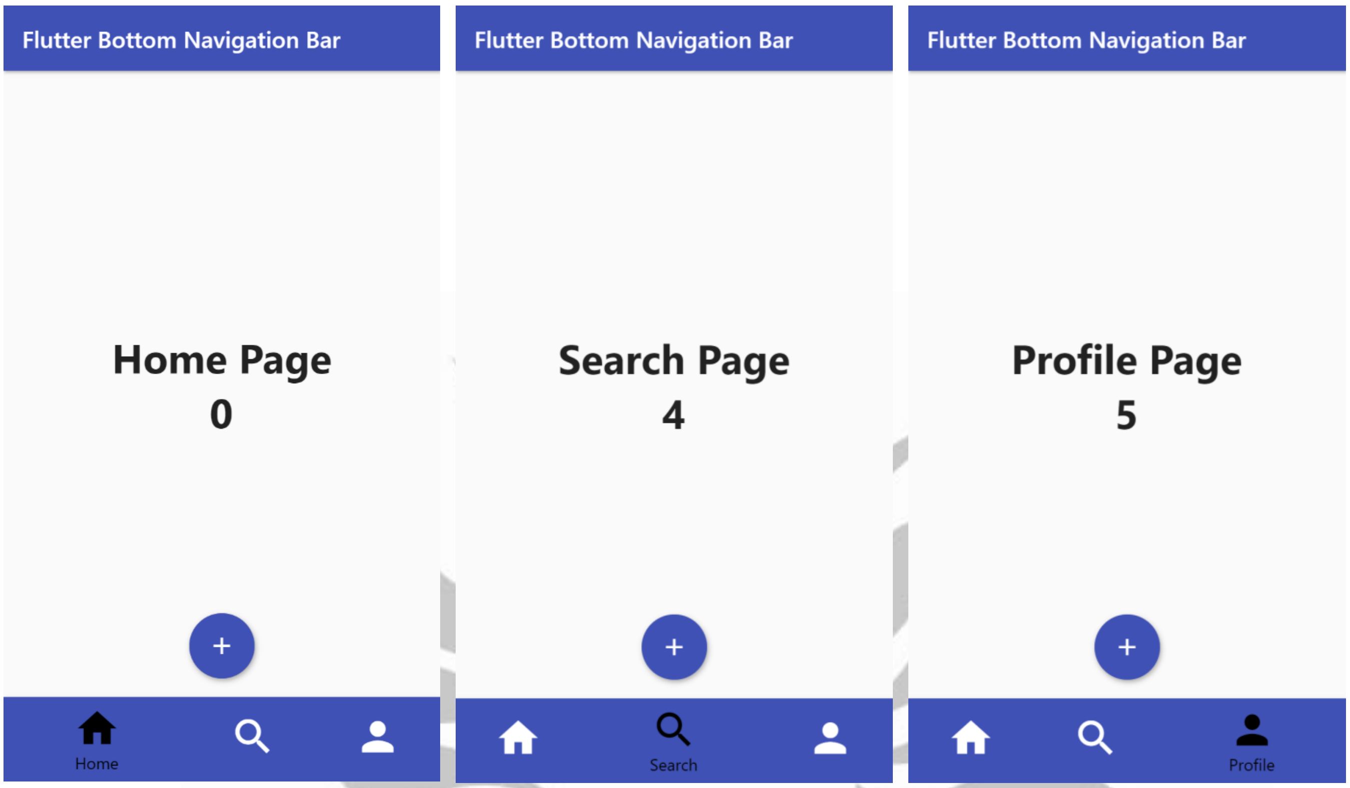
fixedColor: It is used to set the color of the selected item. If we have not set a color to the icon or title, it willbe shown.

type: It determines the layout and behavior of a bottom navigation bar. It behaves in two different ways that are: fixed and shifting. If it is null, it will use fixed. Otherwise, it will use shifting where we can see an animation when we click a button.

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

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

```
class MyApp extends StatelessWidget {
  const MyApp({super.key});
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      home: MyStatefulWidget(),
    );
  }
}

class MyStatefulWidget extends StatefulWidget {
  const MyStatefulWidget({super.key});

  @override
  State<MyStatefulWidget> createState() => _MyStatefulWidgetState();
}

class _MyStatefulWidgetState extends State<MyStatefulWidget> {
  int _count = 0;
  int _selectedindex = 0;

  static const List<Widget> _widgetOptions = <Widget>[
    Text(
      'Home Page',
      style: TextStyle(fontSize: 35, fontWeight: FontWeight.bold),
    ),
    Text(
      'Search Page',
      style: TextStyle(fontSize: 35, fontWeight: FontWeight.bold),
    ),
    Text(
      'Profile Page',
      style: TextStyle(fontSize: 35, fontWeight: FontWeight.bold),
    ),
  ];
  void _onItemTapped(int index) {
    setState(() {
      _selectedindex = index;
    });
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('Flutter Bottom Navigation Bar Example'),
        backgroundColor: Colors.indigo,
      ),
      body: Center(
          child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: <Widget>[
          _widgetOptions.elementAt(_selectedindex),
          Text(
            '$_count',
            style: TextStyle(fontSize: 35, fontWeight: FontWeight.bold),
          ),
        ],
      )),
      bottomNavigationBar: BottomNavigationBar(
        items: const <BottomNavigationBarItem>[
          BottomNavigationBarItem(
            icon: Icon(Icons.home),
            label: 'Home',
            backgroundColor: Colors.indigo,
          ),
          BottomNavigationBarItem(
            icon: Icon(Icons.search),
            label: 'Search',
            backgroundColor: Colors.indigo,
          ),
          BottomNavigationBarItem(
            icon: Icon(Icons.person),
            label: 'Profile',
            backgroundColor: Colors.indigo,
          ),
        ],
        type: BottomNavigationBarType.shifting,
        currentIndex: _selectedindex,
        selectedItemColor: Colors.black,
        iconSize: 40,
        onTap: _onItemTapped,
        elevation: 5,
      ),
      floatingActionButton: FloatingActionButton(
        onPressed: () => setState(() {
          _count++;
        }),
        backgroundColor: Colors.indigo,
        tooltip: 'Increment Counter',
        child: const Icon(
          Icons.add,
          // color: Colors.indigo,
        ),
      ),
      floatingActionButtonLocation: FloatingActionButtonLocation.centerFloat,
    );
  }
}
```

Demonstration of Bottom Navigation Bar:


Flutter Bottom Navigation Bar
Home Page
0
Home
Flutter Bottom Navigation Bar
Search Page
4
Search
Flutter Bottom Navigation Bar
Profile Page
5
Profile

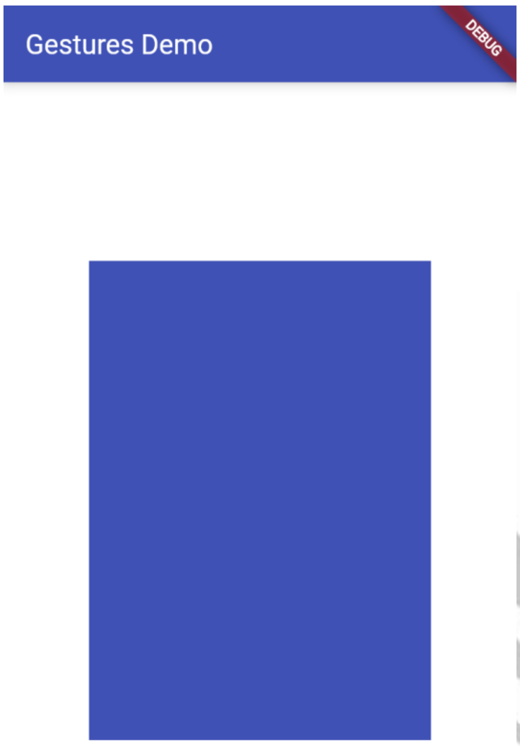
CONCLUSION: - Hence we have successfully designed and applied navigation and routing in Flutter

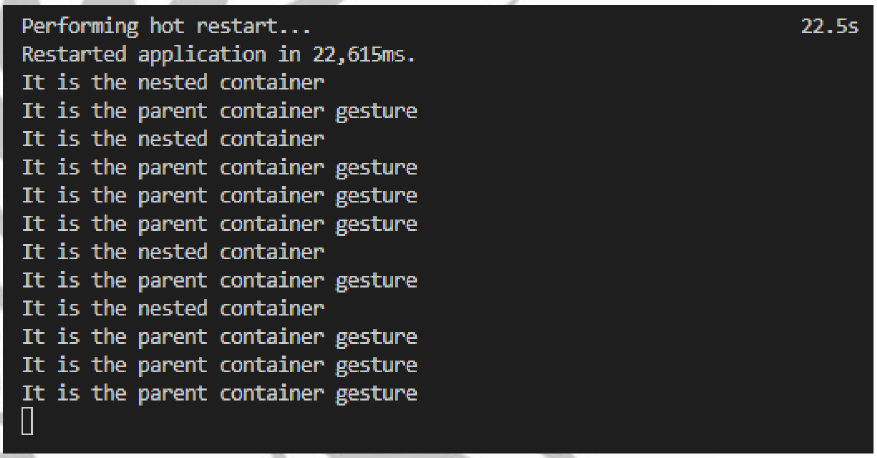
POSTLAB EXERCISE:

Applying gestures in Flutter App

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

//It is the entry point for your Flutter app.
void main() {
  runApp(
    MaterialApp(
      title: 'Multiple Gestures Demo',
      home: Scaffold(
        appBar: AppBar(
          backgroundColor: Colors.indigo,
          title: const Text('Gestures Demo'),
        ),
        body: DemoApp(),
      ),
    ),
  );
}
```

```dart
class DemoApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return RawGestureDetector(
      gestures: {
        AllowMultipleGestureRecognizer: GestureRecognizerFactoryWithHandlers<
            AllowMultipleGestureRecognizer>(
          () => AllowMultipleGestureRecognizer(),
          (AllowMultipleGestureRecognizer instance) {
            // ignore: avoid_print
            instance.onTap = () => print('It is the parent container gesture');
          },
        ),
      },
      behavior: HitTestBehavior.opaque,
      //Parent Container
      child: Container(
        color: Colors.white,
        child: Center(
          //Now, wraps the second container in RawGestureDetector
          child: RawGestureDetector(
            gestures: {
              AllowMultipleGestureRecognizer:
                  GestureRecognizerFactoryWithHandlers<
                      AllowMultipleGestureRecognizer>(
                () => AllowMultipleGestureRecognizer(), //constructor
                (AllowMultipleGestureRecognizer instance) {
                  //initializer
                  // ignore: avoid_print
                  instance.onTap = () => print('It is the nested container');
                },
              )
            },
            //Creates the nested container within the first.
            child: Container(
              color: Colors.indigo,
              width: 250.0,
              height: 350.0,
            ),
          ),
        ),
      ),
    );
  }
}

class AllowMultipleGestureRecognizer extends TapGestureRecognizer {
  @override
  void rejectGesture(int pointer) {
    acceptGesture(pointer);
  }
}
```





```
Performing hot restart...                                              22.5s
Restarted application in 22,615ms.
It is the nested container
It is the parent container gesture
It is the nested container
It is the parent container gesture
It is the parent container gesture
It is the parent container gesture
It is the nested container
It is the parent container gesture
It is the nested container
It is the parent container gesture
It is the parent container gesture
It is the parent container gesture
```