Experiment No: 05

Aim: To apply navigation, routing and gestures in Flutter Application.

THEORY:

Navigation and routing:

These are some of the core concepts of all mobile applications, which allows the user to move between different pages.Every mobile application contains several screens for displaying different types of information. For example, an app can have a screen that contains various products. When the user taps on that product, immediately it will display detailed information about that product.

In Flutter, the screens and pages are known as routes, and these routes are just a widget. In Android, a route is similar to an Activity, whereas, in iOS, it is equivalent to a ViewController.

In any mobile app, navigating to different pages defines the workflow of the application, and the way to handle the navigation is known as routing. Flutter provides a basic routing class MaterialPageRoute and two methods Navigator.push() and Navigator.pop() that shows how to navigate between two routes. The following steps are required to start navigation in your application.

Step 1: First, you need to create two routes.

Step 2: Then, navigate to one route from another route by using the Navigator.push() method.

Step 3: Finally, navigate to the first route by using the Navigator.pop() method.

Gestures:

Gestures are primarily a way for a user to interact with a mobile (or any touch based device) application. Gestures are generally defined as any physical action / movement of a user in the intention of activating a specific control of the mobile device. Gestures are as simple as tapping the screen of the mobile device to more complex actions used in different applications.

Some of the widely used gestures are mentioned here −

1.Tap − Touching the surface of the device with fingertip for a short period and then releasing the fingertip.

2.Double Tap − Tapping twice in a short time.

3.Drag − Touching the surface of the device with fingertip and then moving the fingertip in a steady manner and then finally releasing the fingertip.

4.Flick − Similar to dragging, but doing it in a speeder way.

5.Pinch − Pinching the surface of the device using two fingers.

6.Spread/Zoom − Opposite of pinching.

Gesture Detector:
Flutter provides excellent support for all types of gestures through its exclusive widget, GestureDetector. GestureDetector provides a way to recognize and handle gestures made by the user. It's a versatile widget that can be used to detect various types of gestures, such as taps, drags, long presses, and more. Here's an overview of the theory behind GestureDetector:

Basics of GestureDetector:

Gesture Detection: GestureDetector is used to detect various user interactions or gestures on the screen.It listens to touch events and translates them into higher-level gestures.

Gesture Recognizers: The core functionality of GestureDetector is based on gesture recognizers, which are objects that determine whether a sequence of pointer events represents a specific gesture.Examples of gesture recognizers include TapGestureRecognizer, LongPressGestureRecognizer, PanGestureRecognizer, etc.

Callbacks:GestureDetector uses callbacks to notify your application when a gesture is detected. Common callbacks include onTap, onDoubleTap, onLongPress, onPanUpdate, etc.
Common Properties and Callbacks:

1.onTap:The onTap callback is triggered when the user taps the screen with a single finger.
2.onDoubleTap: The onDoubleTap callback is triggered when the user quickly taps the screen twice with a single finger.
3.onLongPress: The onLongPress callback is triggered when the user presses and holds the screen with a single finger for an extended period.
4.onPanUpdate: The onPanUpdate callback is triggered when the user moves a single finger across the screen, indicating a dragging or panning gesture.
5.onScaleUpdate:The onScaleUpdate callback is triggered when the user performs a scaling gesture, typically involving two fingers moving apart or together.
onVerticalDragUpdate and onHorizontalDragUpdate:

These callbacks are triggered when the user performs vertical or horizontal dragging gestures, respectively.

CODE-

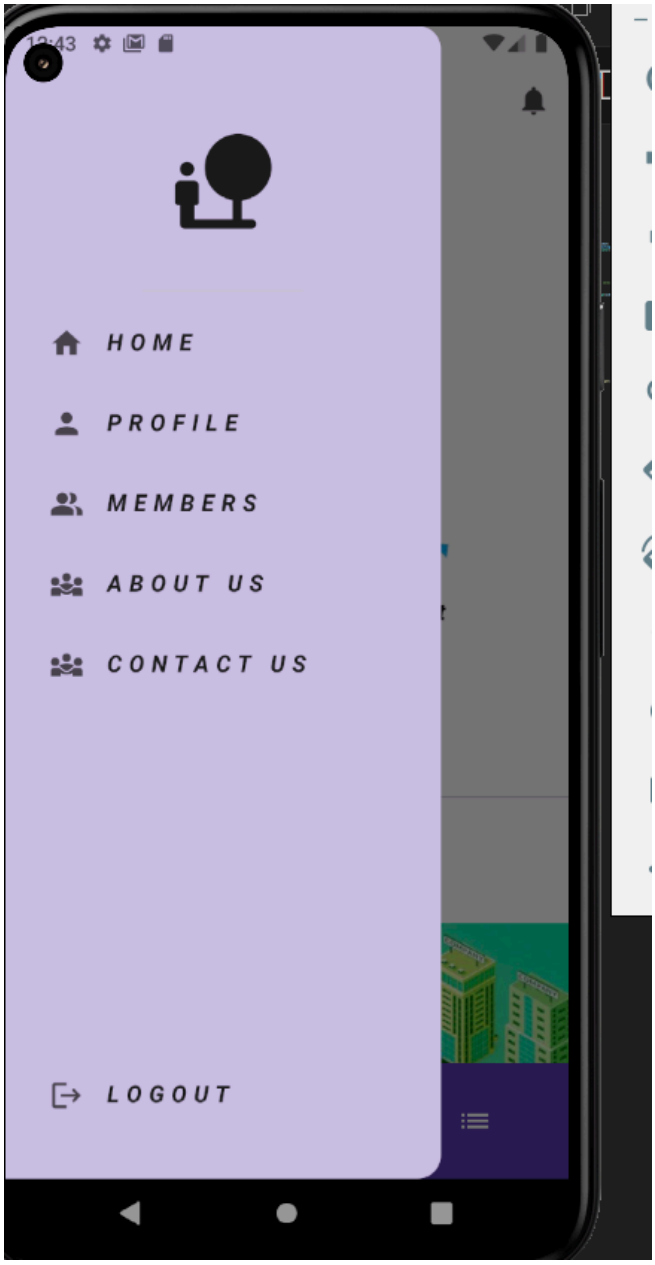
```dart
import 'package:demo_alumnet/services/auth/auth_service.dart';
import 'package:flutter/material.dart';
import 'package:provider/provider.dart';

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

  @override
  State<MyDrawer> createState() => _MyDrawerState();
}

class _MyDrawerState extends State<MyDrawer> {
  void signOut() {
    //get auth service
    final authService = Provider.of<AuthService>(context, listen: false);

    authService.signOut();
  }

  @override
```

```dart
Widget build(BuildContext context) {
  return Drawer(
    backgroundColor: Colors.deepPurple[100],
    child: Column(
      mainAxisAlignment: MainAxisAlignment.spaceBetween,
      children: [
        Column(
          children: [
            // Drawer header
            const DrawerHeader(
              child: Icon(
                Icons.nature_people_rounded,
                size: 80,
              ),
            ),

            //home tile
            Padding(
              padding: const EdgeInsets.only(left: 15),
              child: ListTile(
                leading: const Icon(Icons.home),
                title: const Text('H O M E'),
                onTap: () {
                  // Navigate to Home
                  Navigator.pop(context);
                },
              ),
            ),

            //profile tile
            Padding(
              padding: const EdgeInsets.only(left: 15),
              child: ListTile(
                leading: const Icon(Icons.person),
                title: const Text('P R O F I L E'),
                onTap: () {
                  //navigate to profile page
                  Navigator.pushNamed(context, '/profile_page');
                },
```

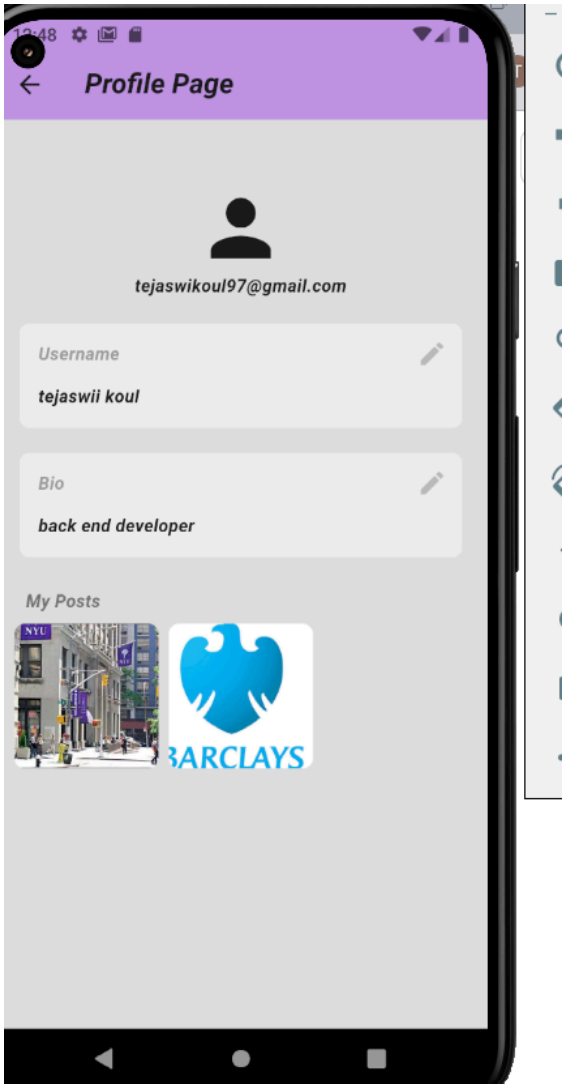
```
        ),
      ),

      //Members tile
      Padding(
        padding: const EdgeInsets.only(left: 15),
        child: ListTile(
          leading: const Icon(Icons.people_alt_sharp),
          title: const Text('M E M B E R S'),
          onTap: () {
            // Navigate to members
            Navigator.pushNamed(context, '/members_list');
          },
        ),
      ),

      // About Us
      Padding(
        padding: const EdgeInsets.only(left: 15),
        child: ListTile(
          leading: const Icon(Icons.diversity_3),
          title: const Text('A B O U T   U S'),
          onTap: () {
            // Navigate to about us
            Navigator.pushNamed(context, '/aboutus_page');
          },
        ),
      ),
      Padding(
        padding: const EdgeInsets.only(left: 15),
        child: ListTile(
          leading: const Icon(Icons.diversity_3),
          title: const Text('C O N T A C T   U S'),
          onTap: () {
            // Navigate to about us
            Navigator.pushNamed(context, '/contacts_page');
          },
        ),
      ),
```

```
            ],
          ),
          Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              // logout button
              Padding(
                padding: const EdgeInsets.only(left: 15, bottom: 30),
                child: ListTile(
                  leading: const Icon(Icons.logout),
                  title: const Text('L O G O U T'),
                  onTap: signOut,
                ),
              ),
            ],
          ),
        ],
      ),
    );
  }
}
```

OUTPUT:



Now we navigate to different pages from this drawer bar e.g profile page-



Conclusion : We have successfully understood and implemented the concepts of route navigation and gestures in flutter.