NAME: Payoshni Khekale ROLL NO:29 CLASS:D15B

EXPERIMENT 05

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

Theory:

Navigation and Routing:
Navigation in a Flutter app refers to the ability to move between different screens or "routes." Each screen typically represents a different UI state or functionality within the app.

Routing refers to the process of defining how the app navigates between these screens. In Flutter, routes are managed by a `Navigator` widget, which maintains a stack of routes and facilitates transitions between them.

Key Concepts:

Navigator:
- The `Navigator` widget manages a stack of `Route` objects.
- You can push a new route onto the stack to navigate to a different screen and pop a route off the stack to go back.

Routes:
- In Flutter, a route is an abstraction for a screen or page.
- Routes are typically represented as widgets that can be pushed onto or popped off the navigator's stack.

Named Routes:
- Named routes provide a way to define a mapping between route names and their corresponding screens.
- This makes it easier to navigate within the app by referencing routes by name rather than directly instantiating and pushing route widgets onto the stack.

MaterialPageRoute and CupertinoPageRoute:
- Flutter provides built-in route transitions through `MaterialPageRoute` (for Android-style transitions) and `CupertinoPageRoute` (for iOS-style transitions).
- These route classes define default animations and transitions when navigating between screens.

NavigatorObservers:
- `NavigatorObserver` objects can be used to observe and react to changes in the navigation stack.
- They can be used for analytics, logging, or implementing custom navigation behaviors.

NAME: Payoshni Khekale ROLL NO:29 CLASS:D15B

Gestures:

Gestures allow users to interact with the app using touch inputs such as taps, swipes, and drags. Flutter provides a rich set of gesture detectors and recognizers to handle various types of user interactions.

Key Concepts:

GestureDetector:

- The `GestureDetector` widget detects various gestures such as taps, drags, and long presses.
- It wraps its child widget and provides callback functions to respond to specific gestures.

Gesture Recognizers:

- Gesture recognizers are objects that interpret raw pointer events and recognize specific gestures.
- Examples include `TapGestureRecognizer`, `PanGestureRecognizer`, `LongPressGestureRecognizer`, etc.

Gesture Callbacks:

- GestureDetector provides callback properties like `onTap`, `onDoubleTap`, `onPanUpdate`, etc., to handle specific gestures.

Discrete Gestures vs. Continuous Gestures:

- Discrete gestures, like taps and double taps, occur at a specific point in time.
- Continuous gestures, like drags and scrolls, involve movement over time and can provide updates as the gesture progresses.

By understanding these concepts and implementing them effectively, you can create fluid and intuitive navigation experiences with rich gesture-based interactions in your Flutter app.

CODE:

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

void main() {
  runApp(MaterialApp(
    title: 'Navigation and Gestures Example',
    initialRoute: '/',
    routes: {
      '/': (context) => FirstRoute(),
      '/second': (context) => SecondRoute(),
    },
  ));
```

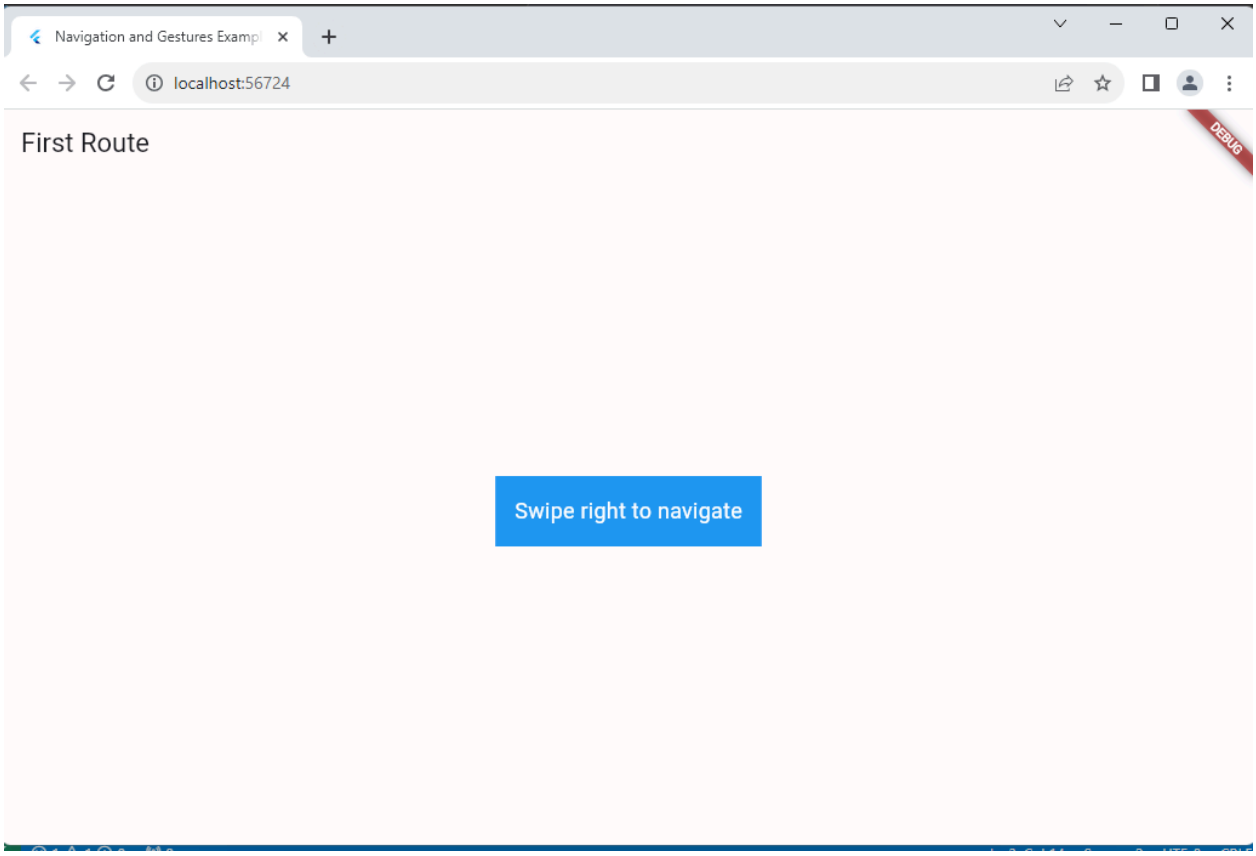
```
}

class FirstRoute extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('First Route'),
      ),
      body: Center(
        child: GestureDetector(
          onHorizontalDragEnd: (details) {
            Navigator.pushNamed(context, '/second');
          },
          child: Container(
            padding: EdgeInsets.all(16.0),
            color: Colors.blue,
            child: Text(
              'Swipe right to navigate',
              style: TextStyle(color: Colors.white, fontSize: 18.0),
            ),
          ),
        ),
      ),
    );
  }
}

class SecondRoute extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Second Route'),
      ),
      body: Center(
        child: ElevatedButton(
          onPressed: () {
            Navigator.pop(context);
          },
          child: Text('Go back!'),
        ),
      ),
    );
```

```
  }
}
```

OUTPUT:


Navigation and Gestures Exampl
localhost:56724
First Route
DEBUG
Swipe right to navigate

NAME: Payoshni Khekale ROLL NO:29 CLASS:D15B