Experiment No: 5

Name:- Vishal Gori
Batch: A
Roll No.: 18
Division:- D15B

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

THEORY:-

Navigation:

Navigation refers to the process of moving between different screens or pages within a mobile or web application. It allows users to explore different parts of the application, access different features, and interact with content.

In Flutter, navigation typically involves using the Navigator widget, which manages a stack of pages. Each page is represented by a widget, and you can push new pages onto the stack to navigate forward and pop them off the stack to navigate backward. The Navigator widget manages the transitions between pages, including animations and route management.

Routing:

Routing is the process of defining and managing the paths (or routes) that users can take through an application. In Flutter, routing involves defining a set of named routes and associating them with specific page widgets. This allows you to navigate to different pages by referencing their route names.

Flutter provides a routing mechanism using the MaterialApp widget's routes parameter, where you can define a map of route names to builder functions that return the corresponding page widgets.

Gestures:

Gestures are touch-based interactions with the user interface elements of an application. In Flutter, gestures are handled using gesture recognizer widgets like GestureDetector. These widgets detect various types of user input, such as taps, drags, swipes, and pinches, and allow you to respond to them with specific actions or behaviors.

Some common gestures in Flutter include:

- **onTap:** Responds to a tap on the screen.
- **onDoubleTap:** Responds to a double tap on the screen.
- **onLongPress:** Responds to a long press on the screen.
- **onHorizontalDragStart, onVerticalDragStart:** Responds to the start of a horizontal or vertical drag gesture.
- **onPanUpdate:** Responds to updates during a panning gesture.

You can wrap any widget with a GestureDetector and specify the gesture callbacks to handle user interactions.

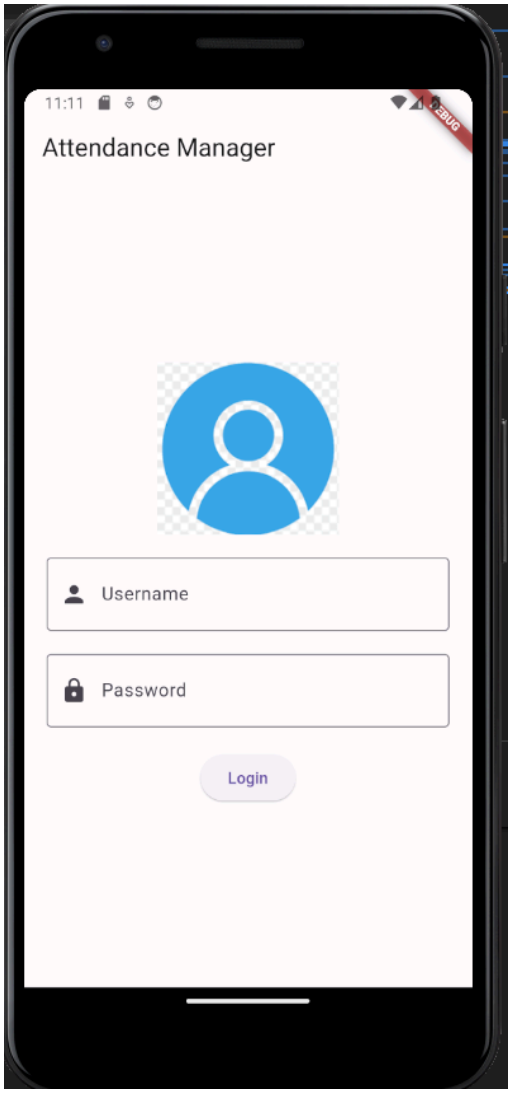
CODE:-

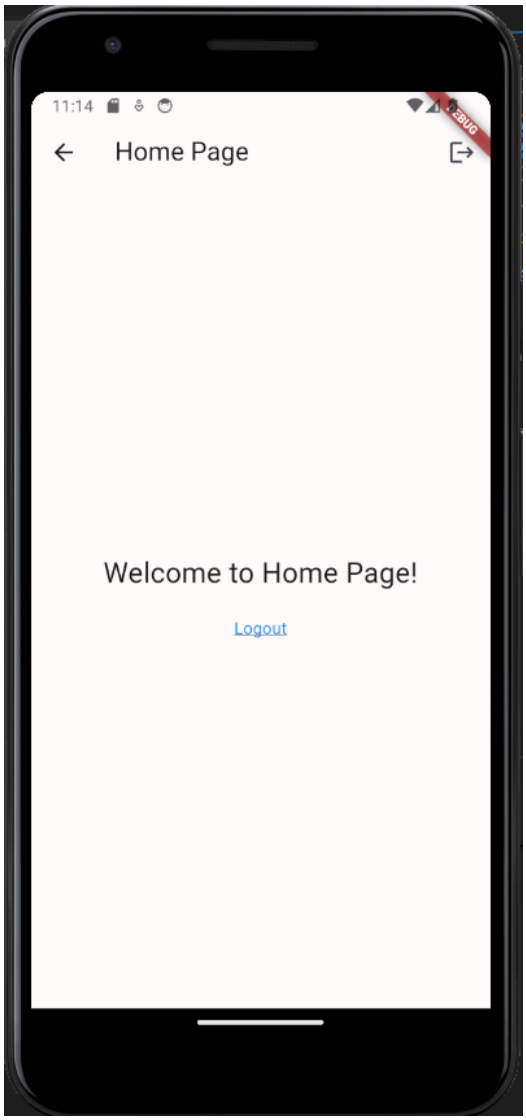
```
import 'package:flutter/material.dart';

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Attendance Manager',
      theme: ThemeData(
        primarySwatch: Colors.blue,
        visualDensity: VisualDensity.adaptivePlatformDensity,
      ),
      home: LoginPage(),
    );
  }
}
```

```
class LoginPage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Attendance Manager'),
      ),
      body: Center(
        child: Padding(
          padding: const EdgeInsets.all(20.0),
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: <Widget>[
              Image.asset(
                'assets/login_logo.png', // Your login logo path
                height: 150,
              ),
              SizedBox(height: 20.0),
              TextField(
                decoration: InputDecoration(
                  labelText: 'Username',
                  prefixIcon: Icon(Icons.person),
                  border: OutlineInputBorder(),
                ),
              ),
              SizedBox(height: 20.0),
              TextField(
                decoration: InputDecoration(
                  labelText: 'Password',
                  prefixIcon: Icon(Icons.lock),
                  border: OutlineInputBorder(),
                ),
                obscureText: true,
              ),
              SizedBox(height: 20.0),
```

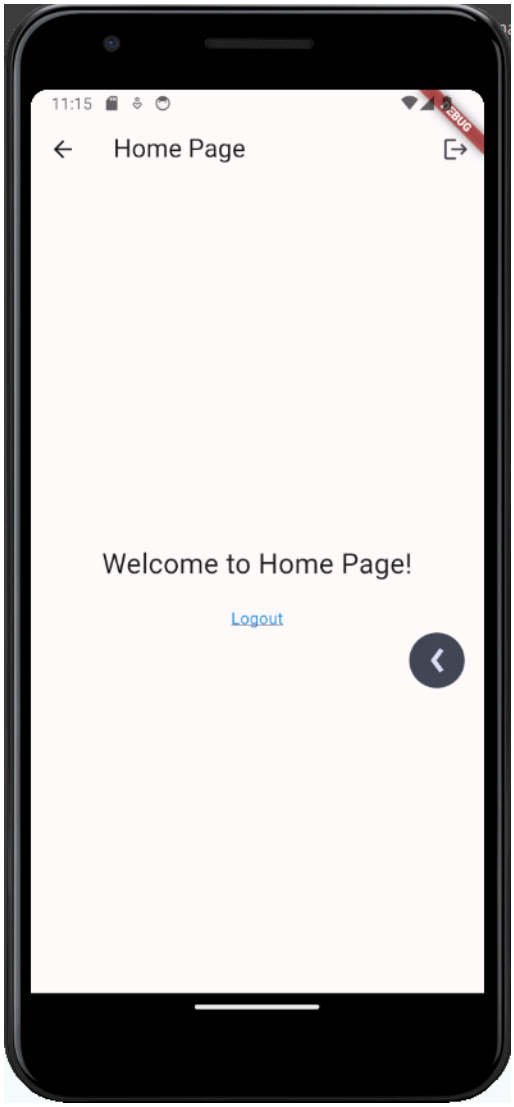
```
            ElevatedButton(
              onPressed: () {
                Navigator.push(
                  context,
                  MaterialPageRoute(builder: (context) => HomePage()),
                );
              },
              child: Text('Login'),
            ),
          ],
        ),
      ),
    ),
  );
 }
}

class HomePage extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
     title: Text('Home Page'),
    ),
    body: Center(
     child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: <Widget>[
       Text(
        'Welcome to Home Page!',
        style: TextStyle(fontSize: 24.0),
       ),
       SizedBox(height: 20.0),
       ElevatedButton(
        onPressed: () {
```

```
              Navigator.pop(context);
            },
            child: Text('Logout'),
          ),
        ],
      ),
    ),
  );
 }
}
```





Routing on logout button



Swipe back gesture add that take you back in previous page

Conclusion: In this experiment we learnt to use navigation , routing and gestures. For navigation we used the login button to navigate to homepage, added route '/' for loginpage and '/home' for HomePage , When clicked on logout set route to '/' i.e. loginpage. We added a gesture onHorizontaDragl gesture to go to the previous page.