Darash Mishra 39 D15B

Experiment 05

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

THEORY:

Navigation:
Navigation is the action of moving between different screens or pages within a mobile or web application. It enables users to explore various sections of the app, access different functionalities, and engage with content seamlessly.

In Flutter, navigation is typically managed using the Navigator widget, which maintains a stack of pages. Each page is represented by a widget, and you can push new pages onto the stack to move forward and pop them off to go backward. The Navigator widget handles transitions between pages, including animations and route management.Navigation:
Navigation is the action of moving between different screens or pages within a mobile or web application. It enables users to explore various sections of the app, access different functionalities, and engage with content seamlessly.

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

Routing:
Routing involves defining and managing the paths (or routes) that users can follow through an application. In Flutter, routing requires specifying a set of named routes and linking them to specific page widgets. This enables navigation to different pages by referencing their route names.

Flutter facilitates routing through the MaterialApp widget's routes parameter. Here, you can establish a map of route names to builder functions that produce the corresponding page widgets.

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

Common gestures in Flutter include:

- onTap: Reacts to a single tap on the screen.
- onDoubleTap: Responds to a double tap on the screen.

- onLongPress: Reacts to a long press on the screen.
- onHorizontalDragStart, onVerticalDragStart: Reacts to the beginning of a horizontal or vertical drag gesture.
- onPanUpdate: Reacts to updates during a panning gesture.

You can enclose any widget with a GestureDetector and specify the appropriate gesture callbacks to manage user interactions effectively.

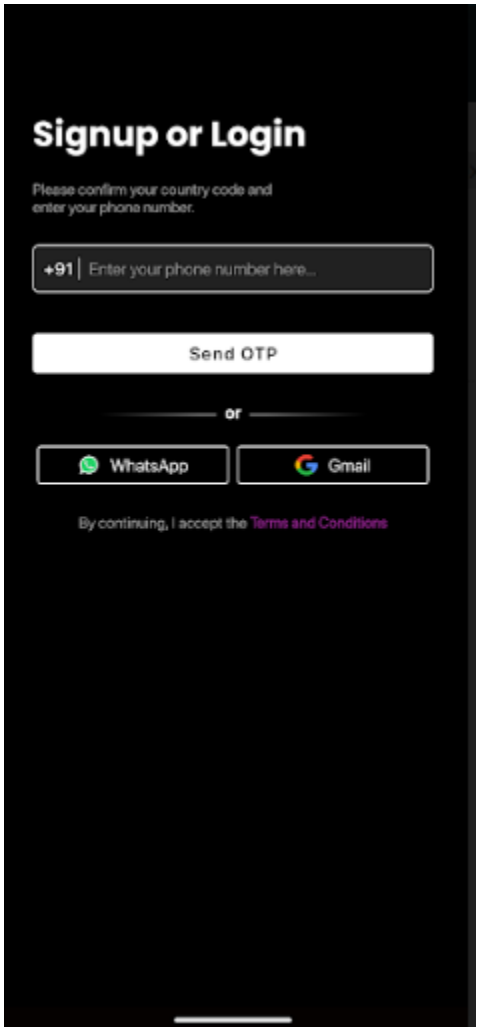
CODE:

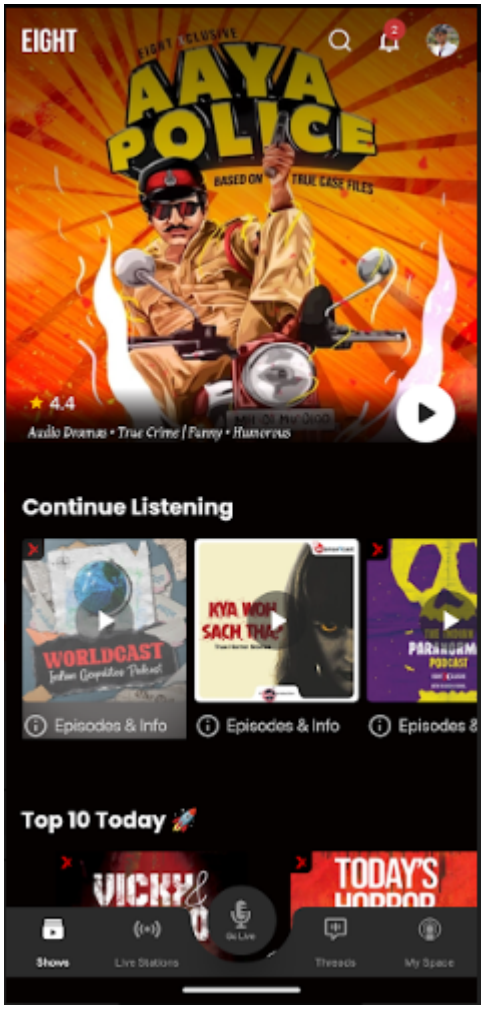
```
import 'package:flutter/material.dart';
void main() {
runApp(MyApp());
}
class MyApp extends StatelessWidget {
@override
Widget build(BuildContext context) {
return MaterialApp(
title: 'Attendance Manager',
theme: ThemeData(
primarySwatch: Colors.blue,
visualDensity: VisualDensity.adaptivePlatformDensity,
),
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(
```

Darash Mishra 39 D15B

```
'Welcome to Home Page!',
style: TextStyle(fontSize: 24.0),
),
SizedBox(height: 20.0),
ElevatedButton(
onPressed: () {
Navigator.pop(context);
},
child: Text('Logout'),
),
],
),
),
);
}
}
```





Darash Mishra 39 D15B

Conclusion:
In Flutter, effective app navigation, routing, and gesture implementation are crucial for delivering a seamless user experience. Navigation facilitates smooth transitions between screens, routing defines the paths users can take, and gestures enable intuitive touch-based interactions.