**EXPERIMENT NO: - 05**

# Name:- Omkar Gholap Class:- D15A Roll:No: - 17

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

# Theory: -

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.

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.

Gestures enable the app to respond to user interactions, making the application more dynamic and responsive.

# Navigation and Routing in Flutter

Navigation is the process of moving between different screens or pages in an app. Flutter provides a simple and effective way to handle this through the use of the Navigator widget and routes.

## 1. Using Navigator Widget

The Navigator widget manages a stack of routes, allowing for pushing and popping routes on the stack.

* **Pushing a Route**: To navigate to a new screen, use Navigator.push().
* **Popping a Route**: To go back to the previous screen, use Navigator.pop().

ElevatedButton( onPressed: () {

Navigator.push(

context,

MaterialPageRoute(builder: (context) => SecondScreen()),

);},

);

## 2. Named Routes

Flutter also allows the use of named routes to navigate, which can make the routing process cleaner, especially in larger applications. MaterialApp(

initialRoute: '/', routes: {

'/': (context) => HomeScreen(),

'/second': (context) => SecondScreen(),

},

);

Navigate to the route using Navigator.pushNamed()

Navigator.pushNamed(context, '/second');

## Handling Gestures in Flutter

Gestures refer to user interactions with the app, such as taps, swipes, pinches, and drags. Flutter provides several widgets and gesture detectors to handle these interactions.

## Tap Gestures

The most common gesture is the tap, which can be handled using the GestureDetector widget or specific buttons like InkWell or ElevatedButton.

## Long Press Gesture

For long press gestures, Flutter provides the onLongPress callback in GestureDetector or InkWell.

## Swipe and Drag Gestures

Flutter also provides swipe and drag gesture handling. The onHorizontalDragUpdate and onVerticalDragUpdate callbacks are used for dragging gestures.

**Code: -**

## Home\_page.dart main.dart

import 'package:flutter/material.dart'; import 'pages/login\_page.dart'; import 'pages/register\_page.dart'; import 'pages/otp\_verification\_page.dart'; import 'pages/myaccountpage.dart'; import 'pages/home\_page.dart';

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

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) { return MaterialApp( debugShowCheckedModeBanner: false,

title: 'AgriApp', theme: ThemeData( primarySwatch: Colors.*green*, colorScheme: ColorScheme.fromSeed(seedColor:

Color(0xFF6A9A5B)),

),

initialRoute: '/login', // Set initial page routes: {

'/login': (context) => LoginPage(),

'/register': (context) => RegistrationPage(),

'/otp': (context) => OtpVerificationPage(),

'/myaccount': (context) => MyAccountPage(),

'/home': (context) => HomePage(),

},

);

} }

## Login\_page.dart

import 'package:flutter/material.dart';

class LoginPage extends StatefulWidget { @override

\_LoginPageState createState() =>

\_LoginPageState();

}

class \_LoginPageState extends State<LoginPage> {

final \_formKey = GlobalKey<FormState>();

@override

Widget build(BuildContext context) { return Scaffold( backgroundColor: Colors.*white*, body: SafeArea( child: SingleChildScrollView( child: Padding( padding: const EdgeInsets.all(16.0), child: Column( crossAxisAlignment: CrossAxisAlignment.stretch,

children: [ SizedBox(height: 50),

// Logo Center( child: Image.asset( 'assets/images/logo.png', height: 150,

),

),

SizedBox(height: 16),

// Title

Text(

'AgriApp: The mix of Agriculture &

Smart, Scientific, Sustainable, Modern Technology Methods for Precision Farming.', textAlign: TextAlign.center, style: TextStyle(

fontSize: 15, color: Colors.*black*,

),

),

SizedBox(height: 32),

// Form Form( key: \_formKey, child: Column( children: [ \_buildTextField( label: 'Email', hint: 'Enter your email', icon: Icons.*email*, validator: (value) { if (value == null ||

value.isEmpty) {

return 'Email is required';

} else if

(!RegExp(r'^[^@]+@[^@]+\.[^@]+').hasMatch( value)) {

return 'Enter a valid email address';

} return null;

},

),

SizedBox(height: 16), \_buildTextField( label: 'Password', hint: 'Enter your password', icon: Icons.*lock*, isPassword: true, validator: (value) { if (value == null ||

value.isEmpty) {

return 'Password is required'; } else if (value.length < 6) { return 'Password must be at least 6 characters';

} return null; },

),

],

),

),

SizedBox(height: 32),

// Login Button ElevatedButton( onPressed: () { if

(\_formKey.currentState!.validate()) {

Navigator.*pushReplacementNamed*(context,

'/home');

}

},

style: ElevatedButton.*styleFrom*( backgroundColor: Colors.*green*, minimumSize: Size(double.*infinity*,

50), shape: RoundedRectangleBorder( borderRadius:

BorderRadius.circular(8),

), ), child: Text( 'Login', style: TextStyle(fontSize: 18, color:

Colors.*white*),

),

),

SizedBox(height: 16),

// Registration link Center( child: TextButton( onPressed: () {

Navigator.*pushNamed*(context,

'/register'); // Add registration route later

}, child: Text(

'Don’t have an account? Register', style: TextStyle(color:

Colors.*green*), // Green color applied

),

),

),

],

),

),

),

),

);

}

Widget \_buildTextField({ required String label, required String hint, required IconData icon, bool isPassword = false, required String? Function(String?) validator,

}) { return TextFormField( obscureText: isPassword, decoration: InputDecoration( labelText: label, hintText: hint, prefixIcon: Icon(icon, color: Colors.*green*), border: OutlineInputBorder( borderRadius: BorderRadius.circular(8),

),

focusedBorder: OutlineInputBorder( borderSide: BorderSide(color: Colors.*green*), // Green color applied borderRadius: BorderRadius.circular(8),

),

),

validator: validator,

);

} }

# add\_video.dart

import 'package:flutter/material.dart'; import 'package:http/http.dart' as http; import 'dart:convert';

class PillMate extends StatefulWidget { @override

\_PillMateState createState() =>

\_PillMateState();

}

class \_PillMateState extends State<PillMate> { final \_formKey = GlobalKey<FormState>(); final TextEditingController \_cityController =

TextEditingController();

String? \_city;

String? \_address;

String? \_updatedAt;

String? \_status;

String? \_temp;

String? \_tempMin;

String? \_tempMax;

String? \_windSpeed;

String? \_pressure;

String? \_humidity;

String? \_sunrise; String? \_sunset; bool \_isLoading = false; bool \_isError = false; bool \_isDataFetched = false; // Add this flag to control the visibility of weather details

Future<void> \_fetchWeather() async { setState(() { \_isLoading = true;

\_isError = false;

\_isDataFetched = false; // Reset this flag when fetching new data

});

final response = await http.get(Uri.*parse*(

'https://api.openweathermap.org/data/2.5/weathe r?q=$\_city&units=metric&appid=73cbebdd0322 acd49bda6ede059b2b18'));

if (response.statusCode == 200) { final data = jsonDecode(response.body);

setState(() { \_address = '${data['name']},

${data['sys']['country']}'; \_updatedAt = 'Updated At:

${DateTime.fromMillisecondsSinceEpoch(data[' dt'] \* 1000).toString()}'; \_status = data['weather'][0]['description'].toUpperCase(); \_temp = '${data['main']['temp']}°C'; \_tempMin = 'Min Temp: ${data['main']['temp\_min']}°C'; \_tempMax = 'Max Temp: ${data['main']['temp\_max']}°C'; \_pressure = 'Pressure: ${data['main']['pressure']} hPa'; \_humidity = 'Humidity:

${data['main']['humidity']}%'; \_windSpeed = 'Wind Speed:

${data['wind']['speed']} m/s';

\_sunrise =

DateTime.fromMillisecondsSinceEpoch(data['sy s']['sunrise'] \* 1000).toString();

\_sunset =

DateTime.fromMillisecondsSinceEpoch(data['sy s']['sunset'] \* 1000).toString();

\_isLoading = false;

\_isDataFetched = true; // Set the flag to true after data is fetched

}); } else { setState(() { \_isLoading = false;

\_isError = true;

\_isDataFetched = false; // Reset the flag if there is an error

});

}

}

@override

Widget build(BuildContext context) { return Scaffold( appBar: AppBar( title: Text('Weather Forecasting'), //backgroundColor: Colors.white,

),

//backgroundColor: Colors.white, body: SafeArea( child: SingleChildScrollView( child: Padding( padding: const EdgeInsets.all(16.0), child: Column( crossAxisAlignment: CrossAxisAlignment.stretch,

children: [ SizedBox(height: 50),

// Logo Center( child: Image.asset( 'assets/images/logo.png', // Replace with your logo

height: 150,

),

),

SizedBox(height: 16),

// Title

Text(

'Weather Forecasting', textAlign: TextAlign.center, style: TextStyle( fontSize: 18, color: Colors.*black*,

),

),

SizedBox(height: 32),

// Form Form( key: \_formKey, child: Column( children: [ TextFormField( controller: \_cityController, decoration: InputDecoration( labelText: 'City', hintText: 'Enter the city name', prefixIcon:

Icon(Icons.*location\_city*, color: Colors.*green*), border: OutlineInputBorder( borderRadius:

BorderRadius.circular(8),

), focusedBorder:

OutlineInputBorder(

borderSide: BorderSide(color: Colors.*green*),

borderRadius:

BorderRadius.circular(8),

), ), validator: (value) { if (value == null ||

value.isEmpty) {

return 'City is required';

} return null;

},

),

SizedBox(height: 16), ElevatedButton( onPressed: () { if

(\_formKey.currentState!.validate()) {

setState(() {

\_city = \_cityController.text;

});

\_fetchWeather();

}

},

style: ElevatedButton.*styleFrom*( backgroundColor: Colors.*green*, minimumSize: Size(double.*infinity*, 50), shape:

RoundedRectangleBorder(

borderRadius:

BorderRadius.circular(8),

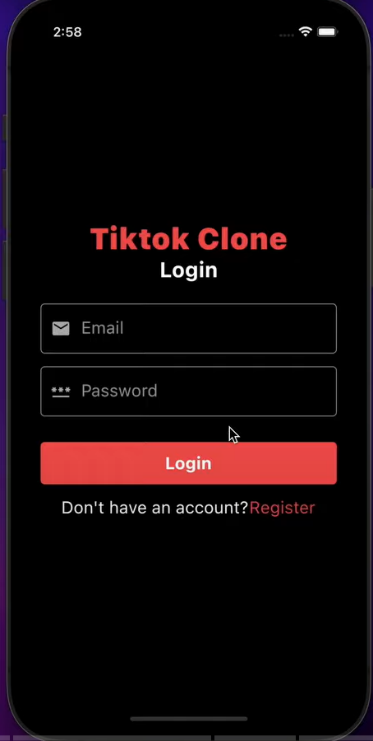
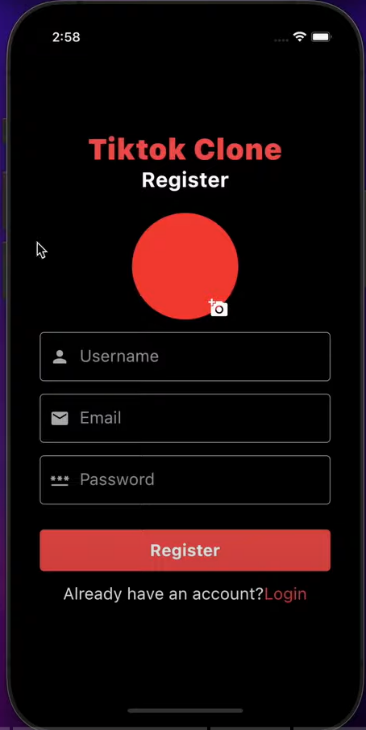
),

),

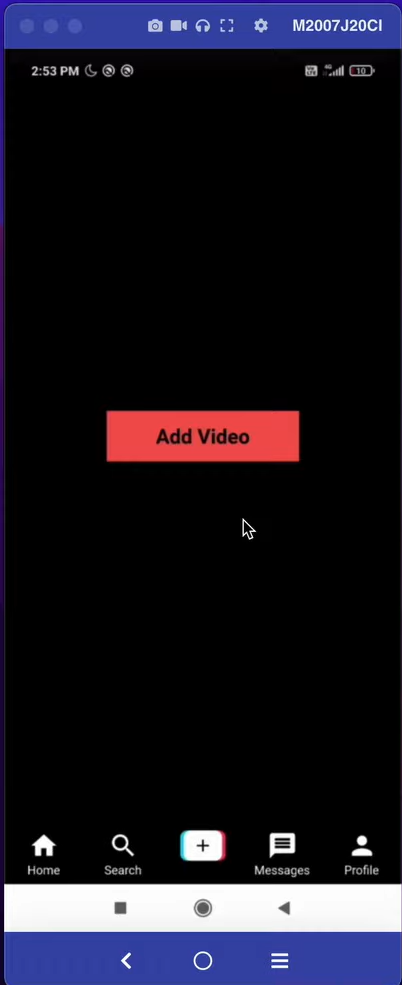
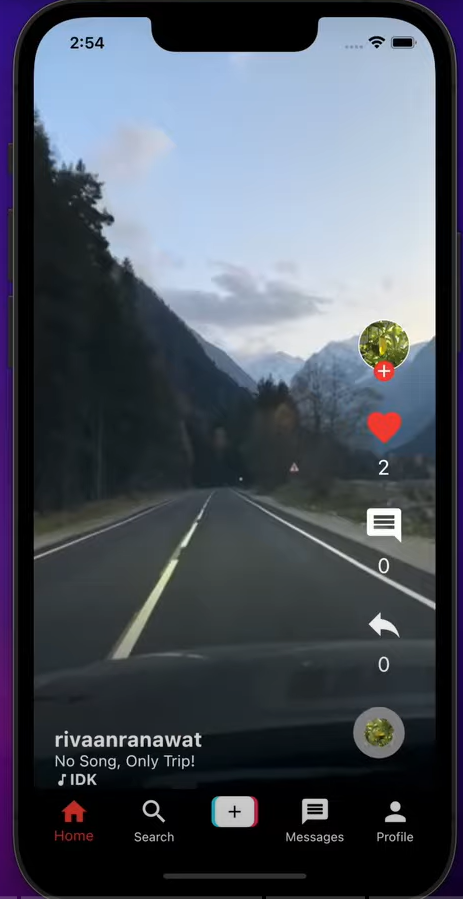
|  |  |
| --- | --- |
| child: Text( 'Get Weather', style: TextStyle(fontSize: 18, color: Colors.*white*),  ),  ),  ],  ),  ),  SizedBox(height: 32),  \_isLoading  ? CircularProgressIndicator()  : \_isError  ? Text(  'Error fetching weather data.', style: TextStyle(color: Colors.*red*), textAlign: TextAlign.center,  )  : \_isDataFetched  ? Column( crossAxisAlignment: CrossAxisAlignment.start,  children: [  Text(  'Location: $\_address', style: TextStyle(fontSize: 16, fontWeight: FontWeight.*bold*),  ),  SizedBox(height: 8),  Text('$\_updatedAt'),  SizedBox(height: 16),  Text('Status: $\_status'),  SizedBox(height: 8),  Text('Temperature: $\_temp'),  SizedBox(height: 8), Text('$\_tempMin'),  SizedBox(height: 8), Text('$\_tempMax'),  SizedBox(height: 8),  Text('$\_windSpeed'),  SizedBox(height: 8), Text('$\_pressure'),  SizedBox(height: 8), Text('$\_humidity'),  SizedBox(height: 8),  Text('Sunrise: $\_sunrise'),  SizedBox(height: 8), | Text('Sunset: $\_sunset'),  ],  )  : Container(),  ],  ),  ),  ),  ),  );  } } |

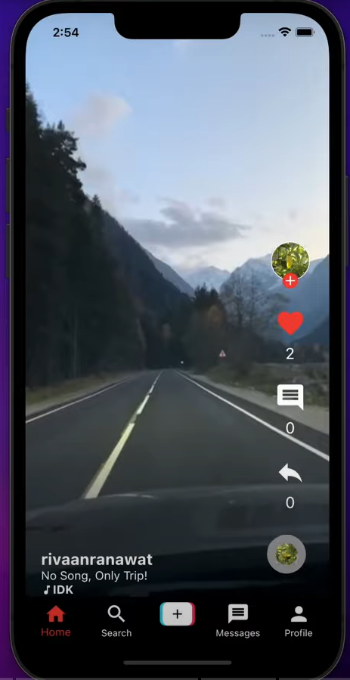
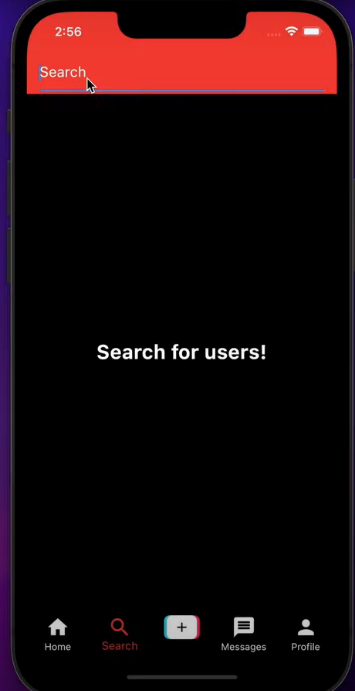
## OUTPUT: -

After clicking on Already have an account? it navigates to the registration page.



In home page, after clicking on “+” icon it navigates to the add video page.



In home page, after clicking on “🔍” icon it navigates to the search video page.

  
In home page, after clicking on “🙍‍♂️ ” icon it navigates to the profile page.

