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

Theory:

In Flutter, navigation allows users to move between different screens (pages). There are two types of navigation: direct navigation (using Navigator.push) and named routing (using Navigator.pushNamed). Named routes make navigation cleaner and easier to manage.
Gestures in Flutter are used to detect user interactions like taps, swipes, and long presses. The GestureDetector widget helps in handling these gestures, allowing us to add custom interactions.

Implementation in Our Code

In our Woman Safety App, we implemented:

- Named Routes – For smooth navigation between Login, Home, and Settings pages.
- Gesture Detection – Added a double-tap gesture on the home screen to send a help message.
- Settings Page Features – Users can update an emergency contact, toggle dark mode, and log out.

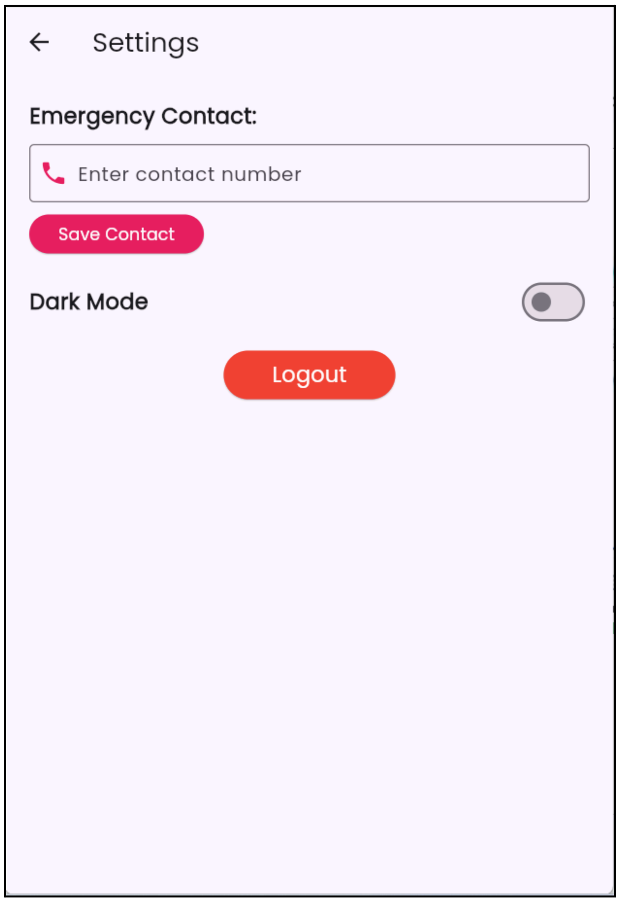
Code :

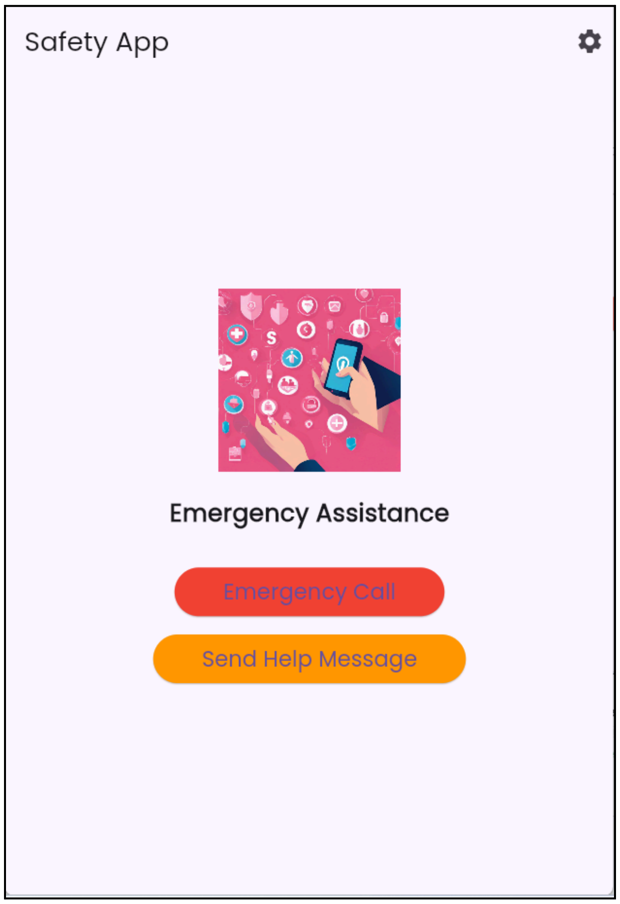
settings.dart

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

class SettingsScreen extends StatefulWidget {
  @override
  _SettingsScreenState createState() => _SettingsScreenState();
}

class _SettingsScreenState extends State<SettingsScreen> {
  bool isDarkMode = false; // Dark mode toggle
  final TextEditingController _contactController = TextEditingController();

  void _saveContact() {
    String contact = _contactController.text;
    if (contact.isNotEmpty) {
      ScaffoldMessenger.of(context).showSnackBar(
        SnackBar(content: Text("Emergency contact saved: $contact")),
      );
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text('Settings')),
      body: Padding(
        padding: EdgeInsets.all(20),
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            // Change Emergency Contact
            Text("Emergency Contact:", style: TextStyle(fontSize: 18,
fontWeight: FontWeight.bold)),
            SizedBox(height: 10),
            TextField(
              controller: _contactController,
              keyboardType: TextInputType.phone,
              decoration: InputDecoration(
                labelText: "Enter contact number",
                border: OutlineInputBorder(),
                prefixIcon: Icon(Icons.phone, color: Colors.pink),
              ),
            ),
            SizedBox(height: 10),
            ElevatedButton(
              onPressed: _saveContact,
              style: ElevatedButton.styleFrom(backgroundColor: Colors.pink),
              child: Text("Save Contact", style: TextStyle(color:
Colors.white)),
            ),
            SizedBox(height: 20),

            // Dark Mode Toggle
            Row(
              mainAxisAlignment: MainAxisAlignment.spaceBetween,
              children: [
                Text("Dark Mode", style: TextStyle(fontSize: 18, fontWeight:
FontWeight.bold)),
                Switch(
                  value: isDarkMode,
                  onChanged: (value) {
                    setState(() {
                      isDarkMode = value;
                      print("Dark Mode: $isDarkMode");
                    });
                  },
                ),
              ],
            ),
            SizedBox(height: 20),

            // Logout Button
            Center(
              child: ElevatedButton(
                onPressed: () {
                  Navigator.pushReplacementNamed(context, '/'); // Go to Login
Page
```

```
                },
                style: ElevatedButton.styleFrom(
                  backgroundColor: Colors.red,
                  padding: EdgeInsets.symmetric(horizontal: 40, vertical: 15),
                ),
                child: Text("Logout", style: TextStyle(fontSize: 18, color:
Colors.white)),
              ),
            ),
          ],
        ),
      ),
    );
  }
}
```

Screenshot:





Conclusion:
In this experiment, we successfully implemented navigation using named routes and gesture detection to enhance user interaction. Initially, we faced issues with incorrect route navigation and gesture recognition, but we resolved them by properly defining routes in MaterialApp and using GestureDetector correctly.