***Experiment N0.05***

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

1. Create two routes. (In Flutter, screens and pages are called routes. )

2. Navigate to the second route using Navigator.push().

3. Return to the first route using Navigator.pop().

**Theory :**

**1. Routes (Screens/Pages):**

In Flutter, a "route" represents a screen or page in your app. Think of it as a distinct UI that the user can navigate to. Each route is typically associated with a widget that defines the content and layout of that screen. You don't explicitly "create" routes in the sense of a separate object. Instead, you define them implicitly through your widget hierarchy and how you navigate between them.

**2. Navigation:**

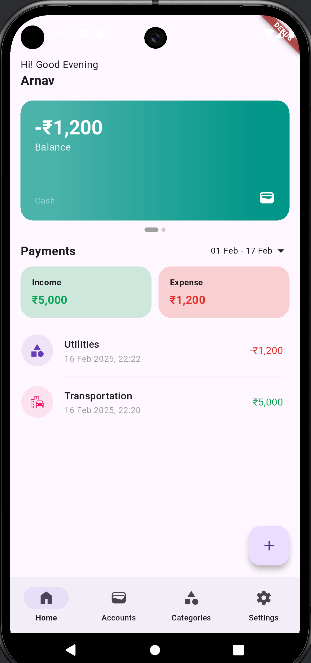
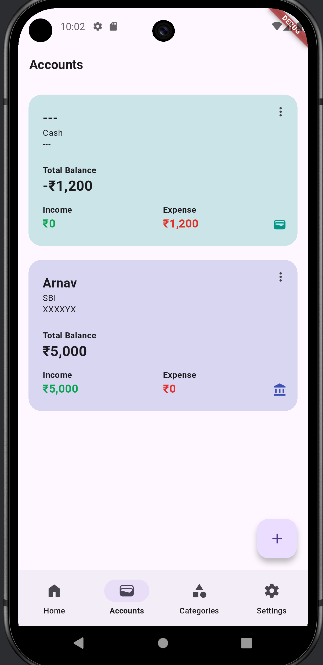
Navigation is the process of moving between these routes (screens). Flutter's Navigator class is the primary mechanism for managing navigation. It uses a stack-based approach, meaning that routes are pushed onto a stack and popped off when you go back.

**3. Navigator.push():**

The Navigator.push() method is used to navigate to a new route (screen). It takes a Route object as an argument, which defines the transition and the widget to display for the new screen. The most common way to create a Route is using MaterialPageRoute:

Code :

import 'package:fintracker/bloc/cubit/app\_cubit.dart';  
import 'package:fintracker/screens/main.screen.dart';  
import 'package:flutter/material.dart';  
import 'package:flutter/services.dart';  
import 'package:flutter\_bloc/flutter\_bloc.dart';  
import 'package:flutter\_localizations/flutter\_localizations.dart';  
  
class App extends StatelessWidget {  
 const App({super.key});  
 @override  
 Widget build(BuildContext context) {  
 SystemChrome.*setSystemUIOverlayStyle*(SystemUiOverlayStyle(  
 statusBarColor: Colors.*transparent*,  
 statusBarIconBrightness: MediaQuery.*of*(context).platformBrightness  
 ));  
 return BlocBuilder<AppCubit, AppState>(  
 builder: (context, state){  
 return MaterialApp(  
 title: 'Fintracker',  
 theme: ThemeData(  
 useMaterial3: true,  
 brightness: MediaQuery.*of*(context).platformBrightness,  
 navigationBarTheme: NavigationBarThemeData(  
 labelTextStyle: WidgetStateProperty.*resolveWith*((Set<WidgetState> states){  
 TextStyle style = const TextStyle(fontWeight: FontWeight.*w500*, fontSize: 11);  
 if(states.contains(WidgetState.selected)){  
 style = style.merge(const TextStyle(fontWeight: FontWeight.*w600*));  
 }  
 return style;  
 }),  
 )  
 ),  
 home: const MainScreen(),  
 localizationsDelegates: const [  
 GlobalWidgetsLocalizations.*delegate*,  
 GlobalMaterialLocalizations.*delegate*,  
 ],  
 );  
 }  
 );  
 }  
}



import 'package:flutter/material.dart';

import 'package:flutter\_bloc/flutter\_bloc.dart';

import 'package:material\_symbols\_icons/symbols.dart';

class MainScreen extends StatefulWidget {

const MainScreen({super.key});

@override

State<MainScreen> createState() => \_MainScreenState();

}

class \_MainScreenState extends State<MainScreen> {

int \_selected = 0;

@override

Widget build(BuildContext context) {

return BlocBuilder<AppCubit, AppState>(

builder: (context, state) {

AppCubit cubit = context.read<AppCubit>();

if (cubit.state.currency == null || cubit.state.username == null) {

return const OnboardScreen(); // No change needed here

}

return Scaffold(

body: \_buildBody(\_selected), // Use a function to build the body

bottomNavigationBar: NavigationBar(

selectedIndex: \_selected,

destinations: const [

NavigationDestination(

icon: Icon(Symbols.home, fill: 1),

label: "Home",

),

NavigationDestination(

icon: Icon(Symbols.wallet, fill: 1),

label: "Accounts",

),

NavigationDestination(

icon: Icon(Symbols.category, fill: 1),

label: "Categories",

),

NavigationDestination(

icon: Icon(Symbols.settings, fill: 1),

label: "Settings",

),

],

onDestinationSelected: (int selected) {

setState(() {

\_selected = selected;

});

},

),

);

},

);

}

Widget \_buildBody(int index) {

switch (index) {

case 0:

return const HomeScreen(); // Or navigate:

// return Navigator.push(context, MaterialPageRoute(builder: (context) => const HomeScreen()));

case 1:

return const AccountsScreen(); // Or navigate:

// return Navigator.push(context, MaterialPageRoute(builder: (context) => const AccountsScreen()));

case 2:

return const CategoriesScreen(); // Or navigate:

// return Navigator.push(context, MaterialPageRoute(builder: (context) => const CategoriesScreen()));

case 3:

return const SettingsScreen(); // Or navigate:

// return Navigator.push(context, MaterialPageRoute(builder: (context) => const SettingsScreen()));

default:

return const HomeScreen(); // Default case

}

}

}

// Example of a screen (HomeScreen):

class HomeScreen extends StatelessWidget {

const HomeScreen({super.key});

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(title: const Text("Home Screen")),

body: Center(

child: ElevatedButton(

onPressed: () {

Navigator.push(

context,

MaterialPageRoute(builder: (context) => const DetailScreen()), // Navigate to detail screen

);

},

child: const Text("Go to Details"),

),

),

);

}

}

class DetailScreen extends StatelessWidget {

const DetailScreen({super.key});

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(title: const Text("Detail Screen")),

body: Center(

child: ElevatedButton(

onPressed: () {

Navigator.pop(context); // Go back

},

child: const Text("Go Back"),

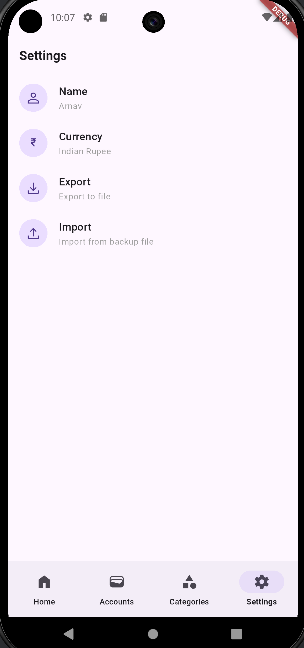
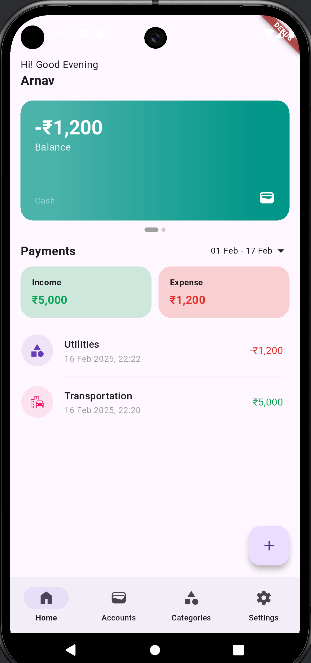
),

),

);

}

}



***Conclusion:***

This experiment successfully demonstrated the fundamental principles of navigation and routing in Flutter. We implemented two distinct routes, representing separate screens within the application. Using Navigator.push(), we enabled seamless transitions from the first route to the second, effectively adding the new screen to the navigation stack. Conversely, Navigator.pop() facilitated a smooth return to the previous screen by removing the top route from the stack.