MPL Lab Exp 5

Name: Anurag Gaiwal Roll No. 17
Div: D15A Batch: A

Aim:
The aim of this experiment is to create an interactive form using Flutter's form widget. The form will include various form fields such as text fields, dropdown menus, and checkboxes, along with validation logic. Additionally, a button will be added to validate and submit the form data.

Theory:
Gesture:

In Flutter, gestures are user interactions with the application, such as taps, drags, and scrolls. The GestureDetector widget is used to detect various gestures and provides callbacks to handle these interactions. Here's a brief overview:

GestureDetector Widget: The GestureDetector widget wraps its child and detects various gestures applied to it. It provides properties like onTap, onDoubleTap, onLongPress, onPanUpdate, etc., to handle different types of gestures.

Gesture Detection Process: When a user interacts with the screen, the GestureDetector widget detects the gesture based on the user's input and invokes the appropriate callback function, allowing developers to respond to the gesture accordingly.

Navigation:

Navigation in Flutter refers to moving between different screens or routes within the application. Flutter's navigation system is built around the Navigator class, which manages a stack of routes. Here's how it works:

Routes: In Flutter, each screen or page is called a route. Routes are pushed onto and popped off the Navigator's stack to navigate between screens.

Navigator.push(): To navigate from one route to another, you use the Navigator.push() method. This method adds a new route to the stack, displaying the new screen on top of the current one.

Navigator.pop(): To return to the previous route, you use the Navigator.pop() method. This removes the top route from the stack, returning to the previous screen.

Steps to Implement Navigation and Gestures:

Create Two Routes: Define two separate screens or widgets to represent the two routes in your application.

Navigate to Second Route: Use Navigator.push() to navigate from the first route to the second route when a specific gesture or action is detected.

Return to First Route: Implement a mechanism, such as a button press or gesture detection, to trigger Navigator.pop() and return from the second route to the first route.

By combining gestures and navigation, you can create dynamic and interactive Flutter applications that provide a seamless user experience for navigating between different screens and responding to user input through gestures.

Code in main.dart:

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

import 'package:audioplayers/audioplayers.dart';

void main() {
  AudioPlayer player = AudioPlayer(); // Initialize audioplayers
  runApp(MyApp(player: player));
}

class MyApp extends StatelessWidget {
  final AudioPlayer player;
  const MyApp({Key? key, required this.player}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Spotify',
      theme: ThemeData(
```

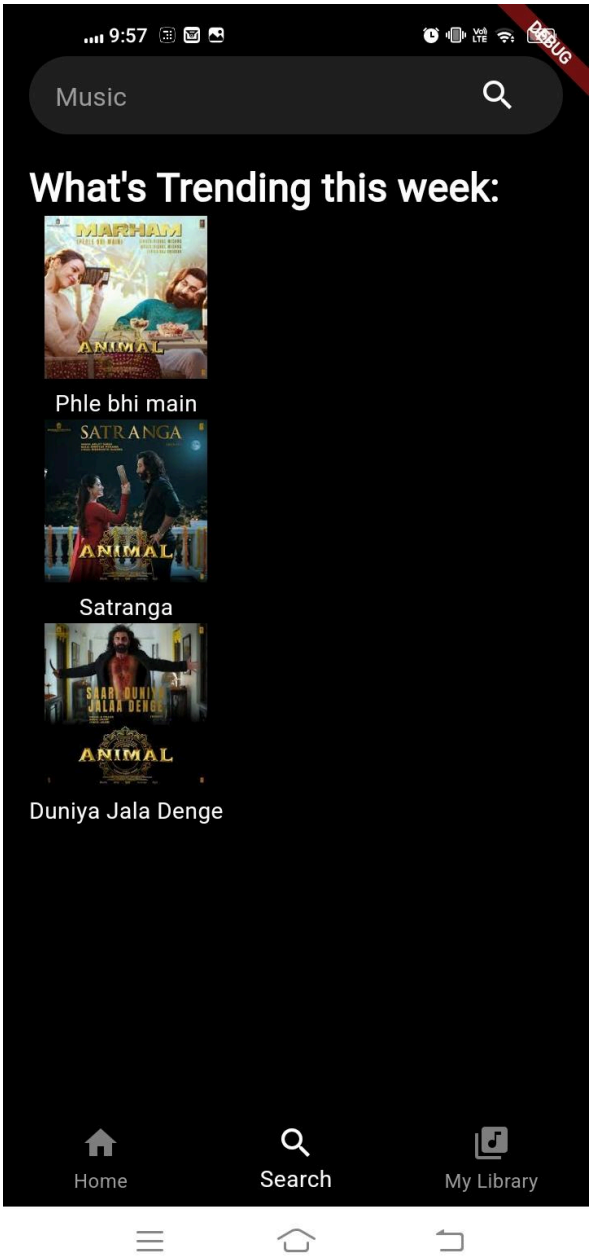
```
      scaffoldBackgroundColor: Colors.black,
      appBarTheme: AppBarTheme(
       backgroundColor: Colors.black,
       titleTextStyle: TextStyle(
         color: Colors.white,
         fontSize: 34.0,
         fontWeight: FontWeight.bold,
       ),
      ),
     ),
     home: SplashScreen(),
   );
  }
}
```

Code in navigation_bar.dart

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

class CustomBottomNavigationBar extends StatelessWidget {
 final int selectedIndex;
 final void Function(int) onItemTapped;
 final int currentIndex;

 CustomBottomNavigationBar({
  required this.selectedIndex,
  required this.onItemTapped,
  this.currentIndex = 0, // Provide a default value for currentIndex
 });

 @override
 Widget build(BuildContext context) {
  return BottomNavigationBar(
   currentIndex: selectedIndex,
   onTap: onItemTapped,
```

```
      items: [
        BottomNavigationBarItem(
          icon: Icon(Icons.home, color: _getColor(0)),
          label: 'Home',
        ),
        BottomNavigationBarItem(
          icon: Icon(Icons.search, color: _getColor(1)),
          label: 'Search',
        ),
        BottomNavigationBarItem(
          icon: Icon(Icons.library_music, color: _getColor(2)),
          label: 'My Library',
        ),
      ],
      backgroundColor: Color.fromARGB(31, 0, 0, 0),
      selectedItemColor: Color.fromARGB(255, 255, 255, 255),
      unselectedItemColor: Colors.grey,
    );
  }

  Color _getColor(int index) {
    return currentIndex == index
        ? Colors.white
        : const Color.fromARGB(255, 128, 128, 128);
  }
}
```

Switched to search screen



Home Screen