Experiment 5:-

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

Theory:

In Flutter, navigation, routing, and gestures are key features for building interactive and multi-screen applications. Navigation and routing allow users to move between different screens (widgets) using methods like Navigator.push() and Navigator.pop(). Named routes and route generators provide better organization and scalability in larger apps. Gestures, on the other hand, enable user interaction through actions like taps, swipes, drags, and long presses using widgets like GestureDetector or InkWell. These interactions enhance user experience by making the app feel dynamic and responsive. Together, navigation, routing, and gestures form the backbone of user interaction and app flow in Flutter development.

Code:-

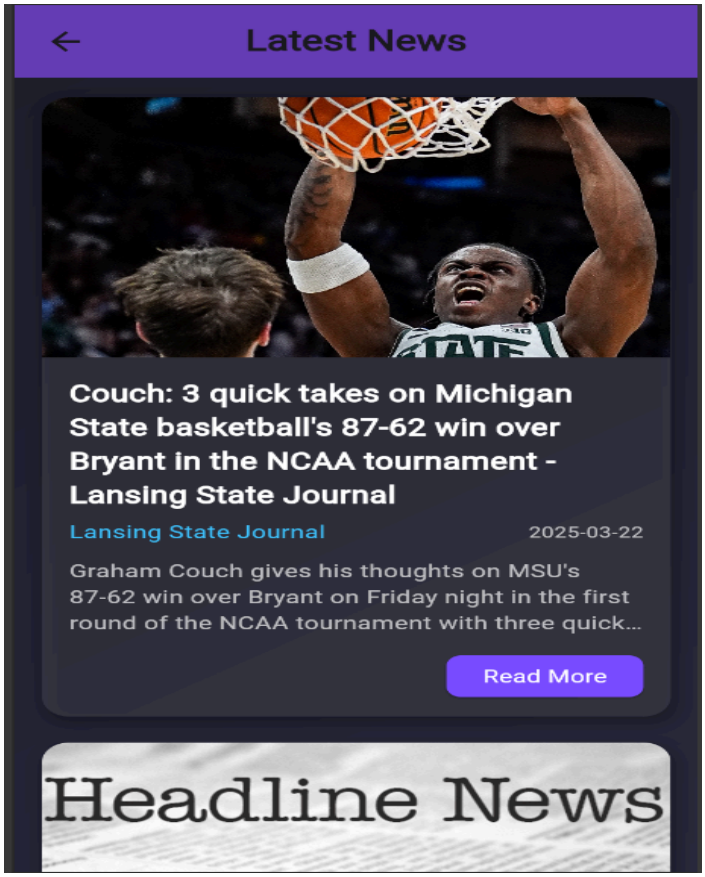
```
onPressed: () {
                Navigator.push(
                  context,
                  MaterialPageRoute(builder: (context) => const NewsPage()),
                );
              },
 onPressed: () {
                Navigator.push(
                  context,
                  MaterialPageRoute(
                      builder: (context) => const SearchPage()),
                );
              },
 onPressed: () async {
                await FirebaseAuth.instance.signOut();
                Navigator.pushReplacement(
                  context,
                  MaterialPageRoute(
                      builder: (context) => const LoginPage()),
                );
```

},

Github Link:-
https://github.com/Ishan2611/News_Edge/commit/d8f3b93eb931d1010863c5a5aa0c8080abc82bf6

Output:-

Here this is the home page and here there are various buttons like ‘Explore News’, ‘Search News’, ‘News Categories’, ‘Logout’. Now after clicking on ‘Search News’ button the user will be redirected to the News Page.



This is the redirected page.

Conclusion:-

In this experiment, we successfully implemented navigation, routing, and gesture features in a Flutter app. The app allows users to switch between screens using buttons, enhancing interactivity and user flow. Gesture detection was applied for smoother user engagement, making the app feel more responsive. Features like Navigator.push() helped in page redirection, demonstrating practical routing. Overall, this experiment highlights the importance of seamless navigation and gesture handling in building dynamic Flutter applications.