Department of Information Technology

Academic Year: 2022-23	**Name of Student:**
Semester: VI	**Student ID:**
Class / Branch / Div: TE- IT A/B	**Roll No.**
Subject: MAD & PWA Lab	**Date of Submission:**
Name of Instructor:	

Experiment No.:5

Aim: To apply navigation and routing in Flutter App.

Theory:

Flutter Navigation and Routing

Navigation and routing are some of the core concepts of all mobile application, which allows the user to move between different pages. We know that every mobile application contains several screens for displaying different types of information. **For example,** an app can have a screen that contains various products. When the user taps on that product, immediately it will display detailed information about that product.

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,** whereas, in iOS, it is equivalent to a **ViewController.**

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.

Step 1: First, you need to create two routes.

Step 2: Then, navigate to one route from another route by using the Navigator.push() method.

Step 3: Finally, navigate to the first route by using the Navigator.pop() method.

Let us take a simple example to understand the navigation between two routes:

Create two routes

Here, we are going to create two routes for navigation. In both routes, we have created only a **single button.** When we tap the button on the first page, it will navigate to the second page. Again, when we tap the button on the second page, it will return to the first page. The below code snippet creates two routes in the Flutter application.

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

The Navigator.push() method is used to navigate/switch to a new route/page/screen. Here, the **push()** method adds a page/route on the stack and then manage it by using the **Navigator.** Again we use MaterialPageRoute class that allows transition between the routes using a platform-specific animation. The below code explain the use of the Navigator.push() method.

```
// Within the `FirstRoute` widget
onPressed: () {
  Navigator.push(
    context,
    MaterialPageRoute(builder: (context) => SecondRoute()),
  );
}
```

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

Now, we need to use Navigator.pop() method to close the second route and return to the first route. The **pop()** method allows us to remove the current route from the stack, which is managed by the Navigator.

To implement a return to the original route, we need to update the **onPressed()** callback method in the SecondRoute widget as below code snippet:

```
// Within the SecondRoute widget
onPressed: () {
  Navigator.pop(context);
}
```

Conclusion: In this experiment we have designed a flutter application with navigation functionality.