Name:Sahil Motiramani Div:D15C Roll No.:35

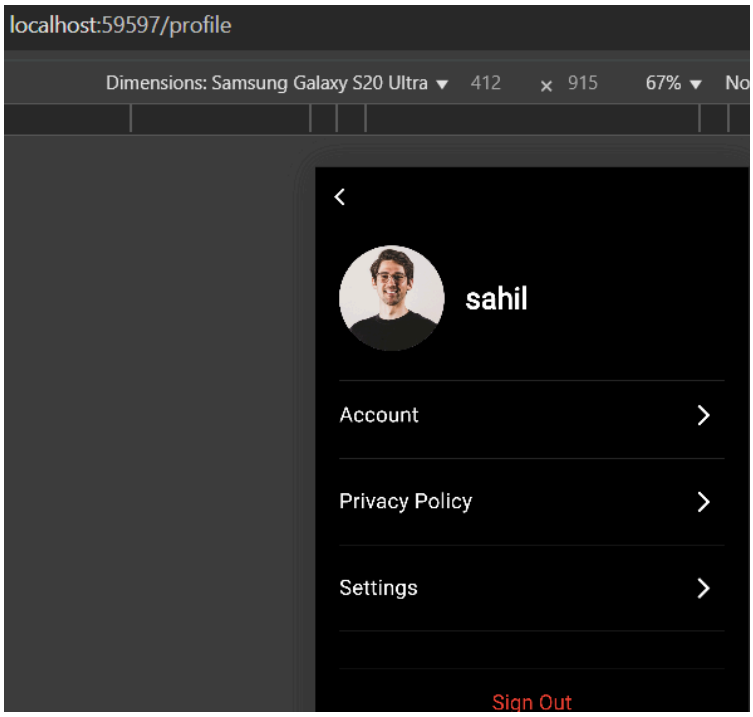
Experiment:5

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

Theory:

- Navigation and routing are essential concepts in Flutter to move between screens or pages, enabling a smooth user flow. Flutter uses a **Navigator** widget, which works like a stack where new routes (screens) are pushed and popped. Navigator.push() is used to navigate to a new screen, while Navigator.pop() is used to go back. Named routes (MaterialApp -> routes) are also supported for cleaner and scalable navigation structure, especially in large apps.
- Routing enables the creation of a multi-screen experience—vital for apps like workout planners, e-commerce platforms, and quizzes. Flutter's MaterialPageRoute is commonly used for simple transitions, but developers can also implement custom transitions for a richer experience.
- Gestures in Flutter are handled using the **GestureDetector** widget. It allows apps to respond to various user interactions like taps, swipes, double taps, and long presses. This is crucial for interactivity—for example, tapping on a workout card to open details, swiping to delete, or long-pressing for options.
- Combining routing and gestures gives the user a dynamic and fluid experience, enhancing both navigation and interactivity of the app.

Output:



Conclusion:

By implementing navigation, routing, and gesture detection, a Flutter app becomes more interactive, dynamic, and user-friendly. These features allow smooth transitions between screens, easy access to features, and natural responses to user actions. In my fitness app, they were essential in linking different pages and adding interactive behavior like tapping workout cards to open detail views.