EXPERIMENT NO.5

Experiment No 5 **Exp 5 To apply navigation, routing, and gestures in Flutter App**	
ROLL NO	**35**
NAME	**Gaurang Milind Mapuskar**
CLASS	**D15-B**
SUBJECT	**MAD & PWA Lab**
LO-MAPPE D	

Theory:

Navigation:

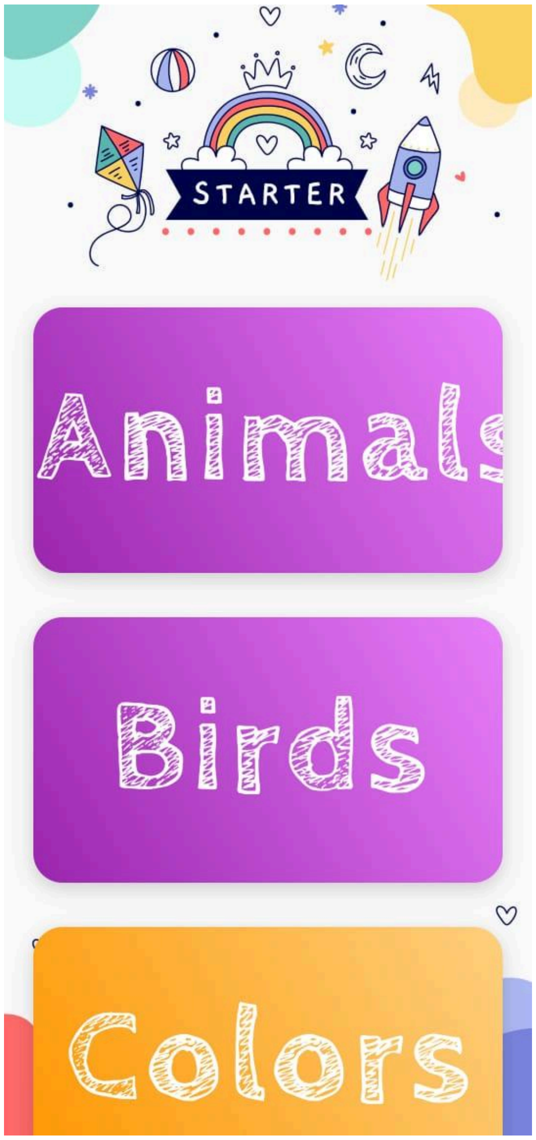
- Navigator Widget:In Flutter, the Navigator widget manages a stack of Route objects representing the screens/pages of the app.
- Push and Pop Navigation: Developers can navigate between screens by pushing new routes onto the stack and popping existing routes off the stack.
- Named Routes: Flutter allows developers to define named routes, making it easier to navigate between specific screens using a route's unique identifier.
- Bottom Navigation Bar: Implementing a bottom navigation bar provides users with quick access to different sections of the app, enhancing navigation efficiency.
- Drawer Navigation: Utilizing a drawer (or sidebar) navigation pattern offers users an additional way to access different sections or features of the app.

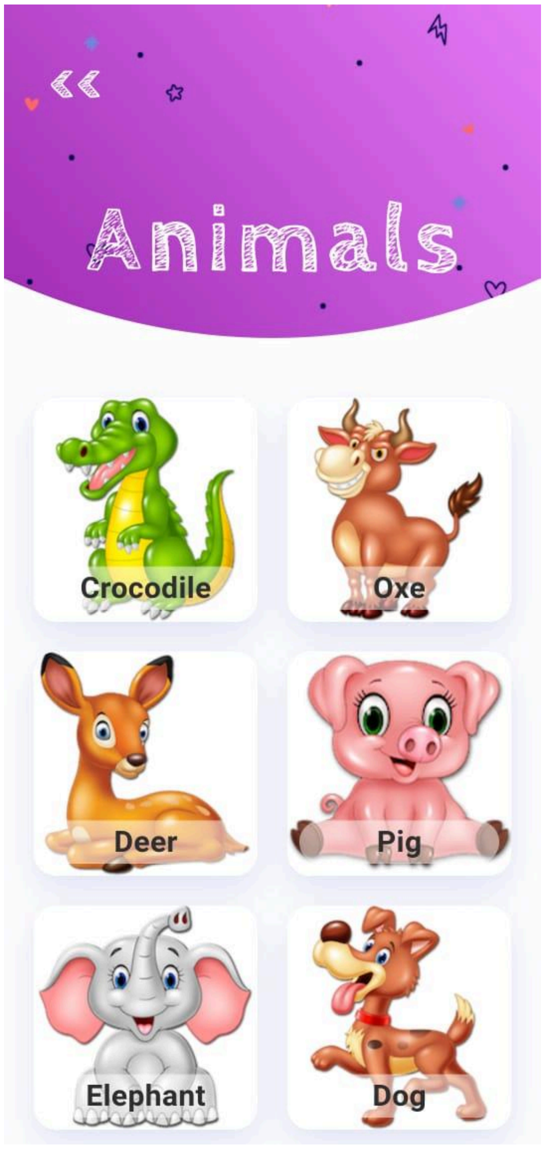
Routing:

- Hierarchical Routing: Flutter's routing system is hierarchical, enabling developers to organize app screens/pages in a structured manner.
- Route Management: Developers can manage routes dynamically based on user actions or application state changes, providing flexibility in app navigation.
- Route Transitions: Flutter allows developers to customize route transitions, such as slide, fade, or scale animations, to provide a polished user experience.
- Route Parameters: Passing parameters between routes enables the communication of data or context between different screens/pages within the app.
- Route Guards: Implementing route guards allows developers to control access to specific routes based on authentication status or other conditions.

Gestures:

- GestureDetector Widget: Flutter's GestureDetector widget allows developers to detect various user gestures such as taps, swipes, drags, and long presses.
- InkWell Widget: InkWell widget provides Material Design ripple effect and gesture detection, enhancing touch feedback for interactive elements.
- Draggable Widget: Flutter offers a Draggable widget to implement drag-and-drop interactions within the app, enabling users to move elements across the screen.
- Gesture Recognizers: Flutter provides built-in gesture recognizers like TapGestureRecognizer and LongPressGestureRecognizer to detect specific types of gestures accurately.
- Gesture Feedback: Providing visual or auditory feedback upon gesture detection improves user feedback and enhances the overall user experience.





Conclusion:
The integration of navigation, routing, and gestures in Flutter app development significantly enhances user experience and interaction. Leveraging Flutter's robust navigation system, hierarchical routing, and diverse gesture recognizers, developers can create intuitive, seamless, and engaging applications. By prioritizing user-centric design and functionality.