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

Experimen:03

Aim: To include icons, images, fonts in Flutter app

Theory:

Including custom icons, images, and fonts enhances the look and feel of a Flutter app, making it more engaging and personalized.

- **Icons**:
  Flutter provides built-in Material and Cupertino icons, which can be used via the Icon widget. Custom icons can be added using packages like flutter_vector_icons or font_awesome_flutter.

- **Images**:
  Flutter supports asset images and network images using the Image.asset() and Image.network() widgets. Asset images must be placed in a dedicated folder (e.g., assets/images/) and declared in pubspec.yaml.

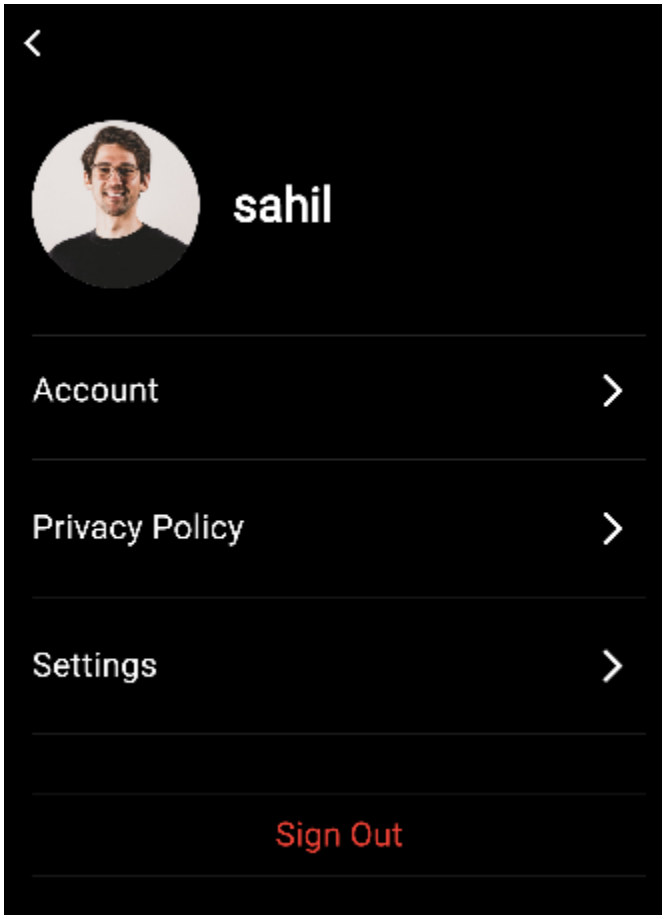
- **Fonts**:
  Custom fonts help define the app's branding. These are added in the assets/fonts/ folder and registered in pubspec.yaml. You can then apply them using the TextStyle class with fontFamily.

- **pubspec.yaml**:
  This configuration file is used to register all assets (icons, fonts, images) under the flutter: section to make them accessible in the app.

- **Best Practices**:
  Keep asset files organized, compress images to reduce app size, and use font weights/styles effectively for consistency.

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

Output:



Conclusion:

Adding custom icons, images, and fonts makes your Flutter app visually appealing and unique. By properly configuring these assets in the pubspec.yaml file and applying them through widgets like Icon, Image, and Text, you can create a polished, branded, and professional-looking UI.