Name: Shreyash Dhasade Expt_No:03 Roll No:14

Experiment No:03

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

Theory:

Including icons, images, and custom fonts in a Flutter app allows developers to enhance the visual appeal and functionality of their applications. Here's a brief overview of how to include these assets:

1. **Icons:**
   a. Flutter provides built-in support for icons through the Icons class, which includes a wide range of Material Design icons.
   b. You can use the Icon widget to display icons in your app. Simply specify the desired icon using the Icons class, along with properties like size and color.
2. **Images:**
   a. To include images in a Flutter app, you can add image files to the assets directory within your project.
   b. Use the Image widget to display images. Specify the image asset path using the Image.asset() constructor.
3. **Fonts:**
   a. Custom fonts can be added to a Flutter app by including font files (e.g., .ttf or .otf) in the project's fonts directory.
   b. Declare the custom fonts in the pubspec.yaml file under the flutter section using the fonts property.
   c. Once declared, you can apply the custom font to text in your app using the fontFamily property in the TextStyle widget.

Here's a summarized step-by-step guide:

1. **Add Icons:**
   a. Use the Icon widget with the desired icon from the Icons class.
   b. Customize the icon size and color as needed.
2. **Add Images:**
   a. Place image files in the assets directory of your Flutter project.
   b. Use the Image.asset() widget to load images from the asset bundle.
   c. Specify the image asset path as a parameter to the Image.asset() constructor.
3. **Add Fonts:**
   a. Place custom font files in the fonts directory of your Flutter project.
   b. Declare the custom fonts in the pubspec.yaml file under the flutter section using the fonts property.
   c. Apply the custom font to text using the fontFamily property in the TextStyle widget.

Code:

```dart
import "package:flutter/material.dart";
import "package:synchwrite/colors.dart";

class LoginScreen extends StatelessWidget {
  const LoginScreen({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Center(
        child: ElevatedButton.icon(
          onPressed: () {},
          icon: Image.asset(
            'assets/images/g-logo-2.png',
            height: 20,
          ),
          label: const Text(
            'Sign in with Google',
            style: TextStyle(color: kBlackColor),
          ),
          style: ElevatedButton.styleFrom(
            backgroundColor: kWhiteColor,
            minimumSize: const Size(150, 50),
          ),
        ),
      ),
    );
  }
}
```

```dart
import 'package:flutter/material.dart';
import 'package:synchwrite/screens/login_screen.dart';

void main() {
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
```

Name: Shreyash Dhasade Expt_No:03 Roll No:14

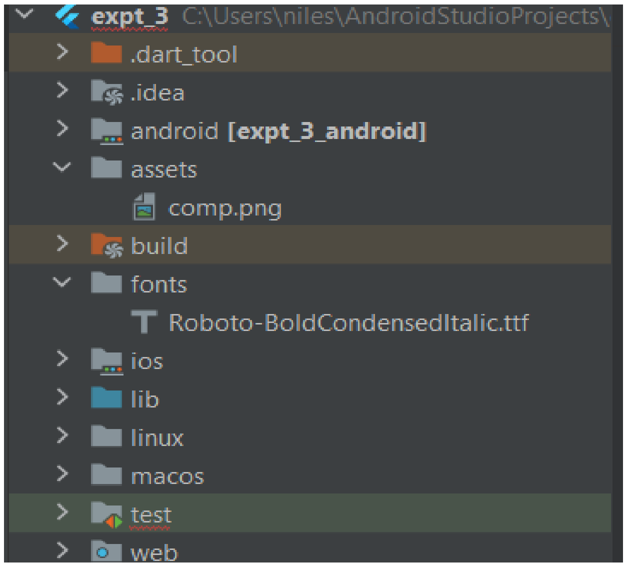
```
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
        useMaterial3: true,
      ),
      home: const LoginScreen(),
    );
  }
}
```

Pubsec.yaml:

```
assets:
 - assets/comp.png

fonts:
 - family: Roboto
   fonts:
     - asset: fonts/Roboto-BoldCondensedItalic.ttf
```

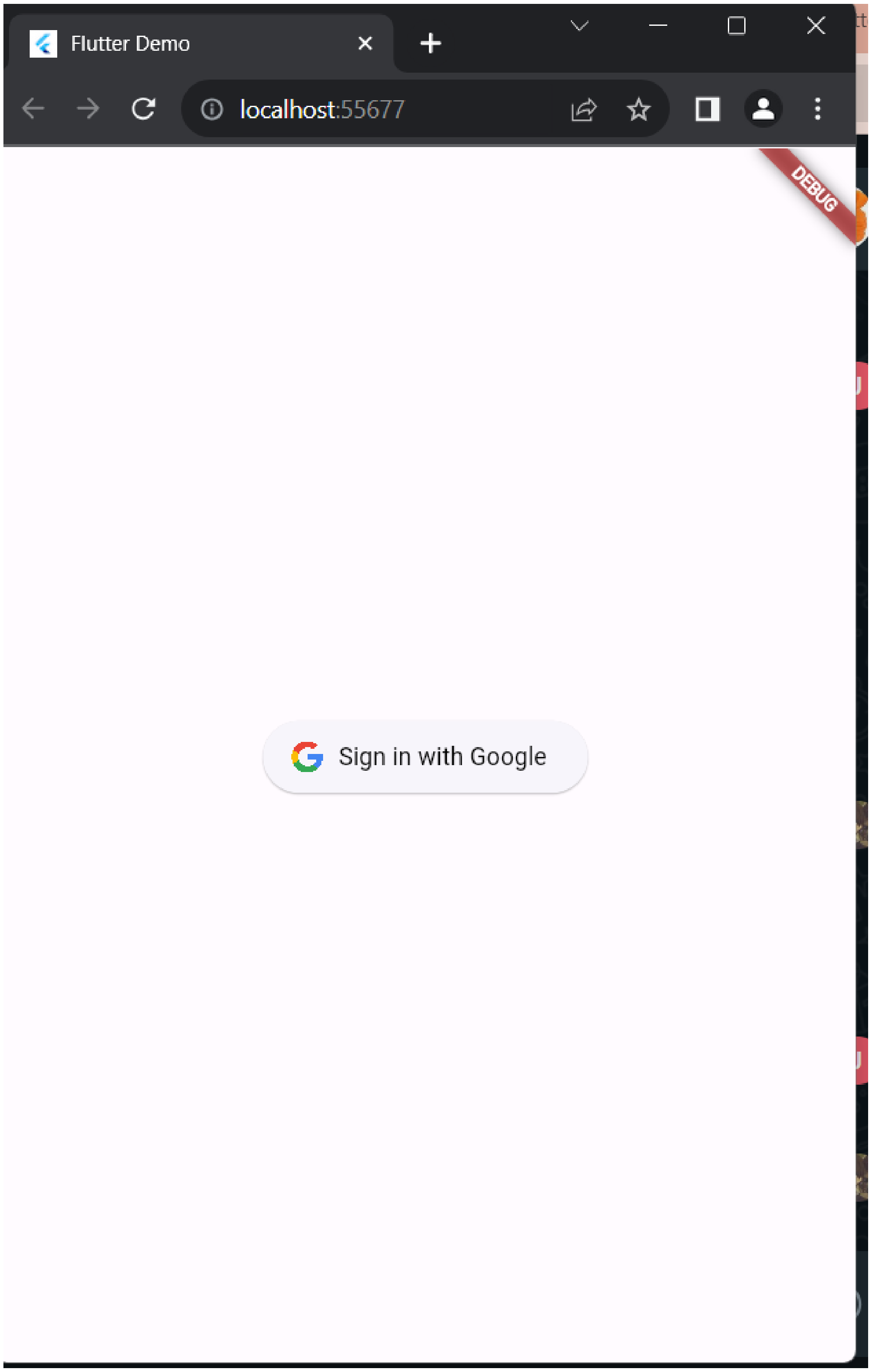
File Structure :



```
expt_3 C:\Users\niles\AndroidStudioProjects\
  .dart_tool
  .idea
  android [expt_3_android]
  assets
    comp.png
  build
  fonts
    Roboto-BoldCondensedItalic.ttf
  ios
  lib
  linux
  macos
  test
  web
```

Output:

Name: Shreyash Dhasade Expt_No:03 Roll No:14



Appended a logo of google as it'll be having a sign-in with google option.
And modified the text color accordingly

Name: Shreyash Dhasade Expt_No:03 Roll No:14

Conclusion:
I have successfully understood and implemented the images, fonts and Icons in a Flutter Application.