

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:

```
import 'package:flutter/material.dart';

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Icons, Images, and Fonts Example',
      theme: ThemeData(
        primarySwatch: Colors.blue,
      ),
      home: MyHomePage(),
    );
  }
}

class MyHomePage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Expt-3'),
      ),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: <Widget>[
            Text(
              'Flutter-Expt:3  
Nilesh',
              style: TextStyle(
                fontSize: 40,
                color: Colors.red,
              ),
            ),
            Icon(
              Icons.favorite,
              size: 50,
              color: Colors.red,
            ),
            SizedBox(height: 20),
          ],
        ),
      ),
    );
  }
}
```

```

Image.asset(
  'assets/comp.png'
  , width: 100,
  height: 100,
),
 SizedBox(height: 20),
Text(
  'Custom Font
  Example', style:
  TextStyle(
    fontFamily: 'Roboto', // Custom font family
    fontSize: 24,
    fontWeight: FontWeight.bold,
  ),
),
],
),
),
);
}
}

```

Pubsec.yaml:

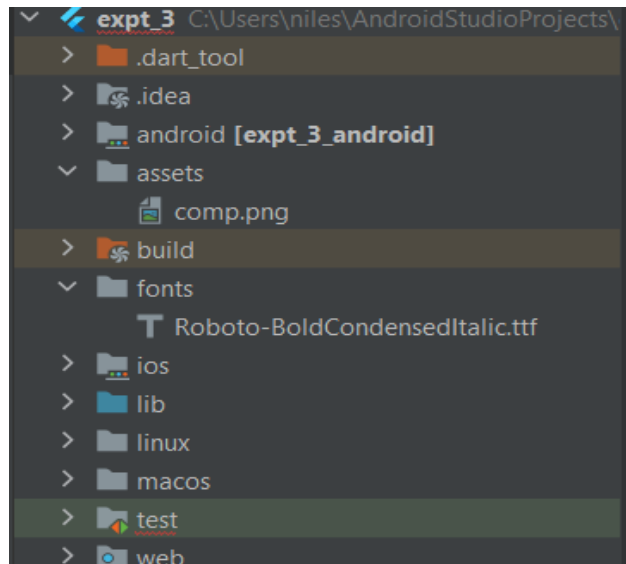
```

assets:
- assets/comp.png

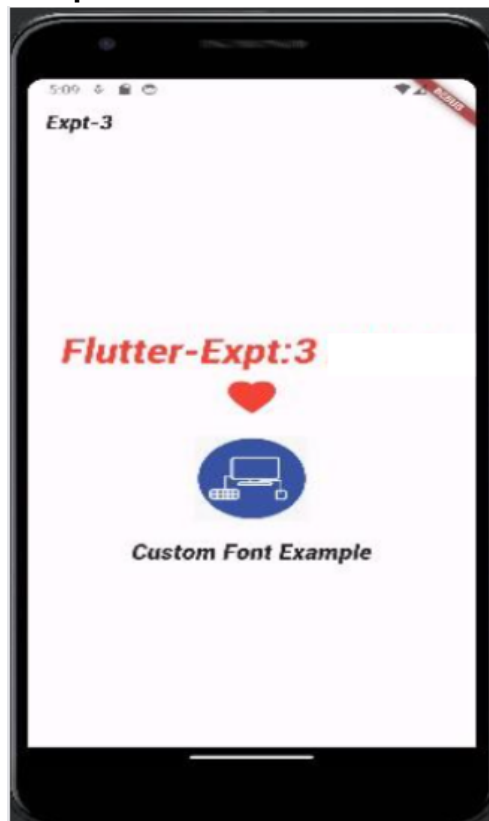
fonts:
- family: Roboto
  fonts:
    - asset: fonts/Roboto-BoldCondensedItalic.ttf

```

File Structure :



Output:



Conclusion:

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