EXPERIMENT 3

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

Theory:

- Flutter is Google's UI toolkit for crafting beautiful, natively compiled iOS and Android apps from a single code base. To build any application we start with widgets – The building block of flutter applications.
- Widgets describe what their view should look like given their current configuration and state. It includes a text widget, row widget, column widget, container widget, and many more.
- Widgets: Each element on a screen of the Flutter app is a widget. The view of the screen completely depends upon the choice and sequence of the widgets used to build the apps. And the structure of the code of an app is a tree of widgets.

Icons :

- In Flutter, icons can be used to represent actions, categories, navigation, or any other visual element that conveys meaning in your application. Flutter provides a wide range of built-in icons that you can use out of the box.

Using Built-in Icons:

- Flutter includes a set of built-in icons that you can use directly. These icons are part of the Material Design and Cupertino Icon sets.
- To use a built-in icon, you can use the Icon widget and specify the icon using the Icons class.

Using Custom Icons:

- You can also use custom icons in Flutter. These icons can be loaded from image files or SVG files. You can use the Image.asset widget for

image files or the flutter_svg package for SVG files.

Images:

- In Flutter, you can display images using the Image widget. Flutter supports various image formats such as PNG, JPEG, GIF, WebP, and animated WebP. Here's how you can display images in Flutter:

1. **Displaying Local Images:** To display images from your Flutter project's assets, you need to add them to the project's pubspec.yaml file and specify their paths. Then, you can use the Image.asset widget to load and display them.

- First, add your images to the assets directory in your Flutter project. ● Update the pubspec.yaml file to include the paths to your image assets:

2. Displaying Network Images:

● You can also display images from URLs using the Image.network widget

Fonts:

In Flutter, you can use custom fonts to style your text. Flutter supports both system fonts and custom fonts. Here's how you can use custom fonts in your Flutter application:

1. Adding Custom Fonts: First, you need to add your custom font files (typically .ttf or .otf files) to your Flutter project. You can place these font files in a directory, often named fonts, in your project's directory structure.

2. Updating pubspec.yaml: Next, you need to update the pubspec.yaml file to include the paths to your custom font files.

3. Using Custom Fonts: Once you have added your custom font files and updated the pubspec.yaml, you can use your custom fonts in your Flutter application.

4. We use the fontFamily property to specify the name of the custom font family defined in the pubspec.yaml. We set other text properties such as fontSize and fontWeight as needed.

Code:

```
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:demo_alumnet/components/my_bottom_navbar.dart';
import 'package:demo_alumnet/components/my_drawer.dart';
import 'package:flutter/material.dart';
import 'package:firebase_auth/firebase_auth.dart';

class HomePage extends StatefulWidget {
  const HomePage({Key? key});

  @override
  State<HomePage> createState() => _HomePageState();
}

class _HomePageState extends State<HomePage> {
  String appTitle = "VESlumni";
  late String _userId;

  @override
  void initState() {
    super.initState();
    _getCurrentUser();
  }

  void _getCurrentUser() async {
    User? user = FirebaseAuth.instance.currentUser;
    if (user != null) {
      setState(() {
        _userId = user.uid;
      });
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(
```

```
        appTitle,
        style: const TextStyle(
          fontSize: 20,
          fontWeight: FontWeight.bold,
          fontFamily: 'Roboto', // Use Roboto font family
        ),
      ),
      actions: [
        IconButton(
          onPressed: () {
            Navigator.pushNamed(context, '/notification_page');
          },
          icon: const Icon(Icons.notifications),
        ),
      ],
    ),
    drawer: const MyDrawer(),
    body: Center(
      child: StreamBuilder(
        stream: FirebaseFirestore.instance.collection('posts').snapshots(),
        builder: (context, AsyncSnapshot<QuerySnapshot> snapshot) {
          if (snapshot.connectionState == ConnectionState.waiting) {
            return const Center(
              child: CircularProgressIndicator(),
            );
          }

          if (snapshot.hasError) {
            return Center(
              child: Text('Error: ${snapshot.error}'),
            );
          }

          if (!snapshot.hasData || snapshot.data!.docs.isEmpty) {
            return const Center(
              child: Text('No posts available.'),
            );
          }

          return SingleChildScrollView(
            child: Column(
              children: snapshot.data!.docs.map((DocumentSnapshot post) {
                return PostWidget(post: post, userId: _userId);
              }).toList(),
```

```
            ),
          );
        },
      ),
    ),
    bottomNavigationBar: const MyBottomNavBar(),
  );
 }
}

class PostWidget extends StatefulWidget {
 final DocumentSnapshot post;
 final String userId;

 const PostWidget({Key? key, required this.post, required this.userId})
     : super(key: key);

 @override
 _PostWidgetState createState() => _PostWidgetState();
}

class _PostWidgetState extends State<PostWidget> {
 bool isLiked = false;

 @override
 Widget build(BuildContext context) {
   Map<String, dynamic> data = widget.post.data() as Map<String, dynamic>;
   String postText = data['text'];
   String postImage = data['imagePath'] ?? '';
   String postUserId = data['userId'];
   int likesCount = data['likesCount'] ?? 0;

   return Column(
     crossAxisAlignment: CrossAxisAlignment.stretch,
     children: [
       Padding(
         padding: const EdgeInsets.all(8.0),
         child: ListTile(
           leading: CircleAvatar(
             radius: 20,
             backgroundImage: AssetImage('assets/default_profile_image.png'),
           ),
           title: Padding(
             padding: const EdgeInsets.only(left: 8.0),
```

```
            child: FutureBuilder(
              future: _getUsername(postUserId),
              builder: (context, AsyncSnapshot<String> usernameSnapshot) {
                if (usernameSnapshot.connectionState ==
                    ConnectionState.waiting) {
                  return const LinearProgressIndicator();
                }

                if (usernameSnapshot.hasError) {
                  return Text('Error: ${usernameSnapshot.error}');
                }

                return Text(
                  '${usernameSnapshot.data}',
                  style: const TextStyle(
                    fontWeight: FontWeight.bold,
                  ),
                );
              },
            ),
          ),
        ),
      ),
      if (postImage.isNotEmpty)
        Padding(
          padding: const EdgeInsets.symmetric(vertical: 8.0),
          child: AspectRatio(
            aspectRatio: 16 / 9,
            child: Image.network(
              postImage,
              fit: BoxFit.cover,
            ),
          ),
        ),
      Padding(
        padding: const EdgeInsets.fromLTRB(18.0, 8.0, 24.0, 16),
        child: Text(
          postText,
          style: const TextStyle(
            fontSize: 16,
            fontFamily: 'Roboto', // Use Roboto font family
          ),
        ),
      ),
```

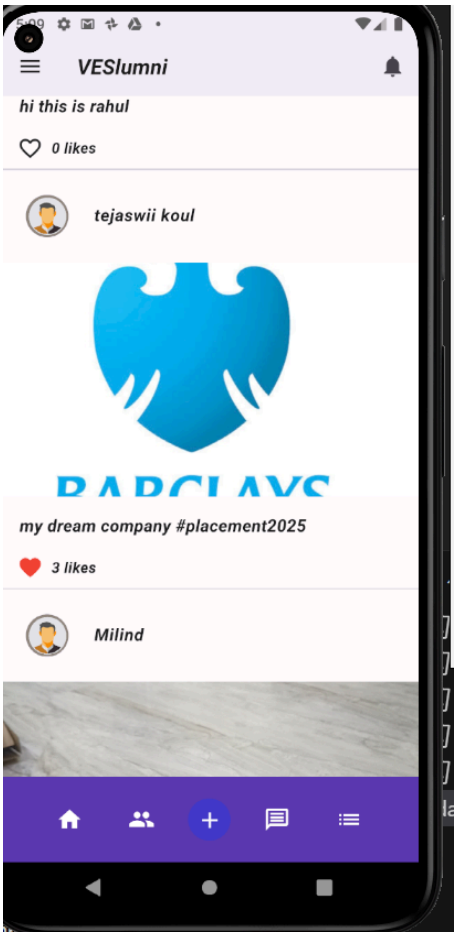
```
      Padding(
        padding: const EdgeInsets.symmetric(horizontal: 16),
        child: Row(
          mainAxisAlignment: MainAxisAlignment.spaceBetween,
          children: [
            Row(
              children: [
                GestureDetector(
                  onTap: () {
                    _toggleLike(widget.post.reference);
                  },
                  child: Icon(
                    isLiked ? Icons.favorite : Icons.favorite_outline,
                    size: 24,
                    color: isLiked ? Colors.red : null,
                  ),
                ),
                const SizedBox(width: 8),
                Text(
                  '$likesCount likes',
                  style: TextStyle(
                    fontWeight: FontWeight.bold,
                  ),
                ),
              ],
            )
          ],
        ),
      ),
      const Divider(),
    ],
  );
}

void _toggleLike(DocumentReference postRef) {
  setState(() {
    isLiked = !isLiked;
  });

  postRef.update({
    'likesCount':
        isLiked ? FieldValue.increment(1) : FieldValue.increment(-1),
  });
}
```

```
  Future<String> _getUsername(String userId) async {
    try {
      DocumentSnapshot userSnapshot = await FirebaseFirestore.instance
          .collection('users')
          .doc(userId)
          .get();
      return userSnapshot['username'];
    } catch (e) {
      print('Error fetching username: $e');
      return '';
    }
  }
}
```

Output:



conclusion: We have successfully understood and studied about the basic widgets in flutter and made use of image, icons and fonts in flutter. With the help of this we have designed a simple dashboard.