Experiment 2: Exploring Flutter Widgets

Aim: To design Flutter UI by including common widgets.

Theory:

Designing a Flutter UI involves combining and customizing a variety of common widgets to create a visually appealing and functional user interface.

Widgets

In Flutter, widgets are the building blocks of the UI. Widgets can be either stateless or stateful and can be combined to create complex UIs.

1. StatelessWidget:

● A basic building block in Flutter.
● Represents part of the user interface that can be described by a configuration that cannot change over time. Example: Container, Icon, Text.

2. StatefulWidget:

● Represents part of the user interface that can change dynamically.
● Has mutable state that affects its appearance. Example: TextField, Checkbox, Radio.

BASIC WIDGETS

- Container: Box model for layout and styling.
- Row and Column: Horizontal and vertical layout.
- Stack: Stacks widgets on top of each other. ListView: Scrollable list of widgets. GridView: Scrollable grid layout.
- Card: Material design card for grouping info.
- AppBar: Top app bar with title and actions.
- TextField: User text input field.
- Buttons: ElevatedButton, TextButton, OutlinedButton.
- Icon: Displays material design icons.
- Image: Displays images with various sources.
- Divider: Creates horizontal or vertical lines.
- SizedBox: Box with specified width and height.
- Expanded: Takes up remaining space.
- Flexible: Adjusts flex factor in a Flex widget.

- PageRouteBuilder: Customizable page transitions.
- ClipRRect and ClipOval: Clipping with rounded corners or oval shape.
- Sliver Widgets: Advanced scrolling in CustomScrollView.

Code:

Make a folder screens in the lib folder. Under screens folder make a home.dart,feed.dart,post_screen.dart,favorite_screen.dart,post_screen.dart,sear ch.dart,profile_screen.dart

Main.dart

```
import 'package:flutter/material.dart';
 import 'screens/login.dart'; void main() {
 runApp(const MyApp());

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

@override

Widget build(BuildContext context) {
return MaterialApp( title: 'Flutter Demo', theme: ThemeData(), home:const
LoginScreen(), );

} }
```

Home.dart

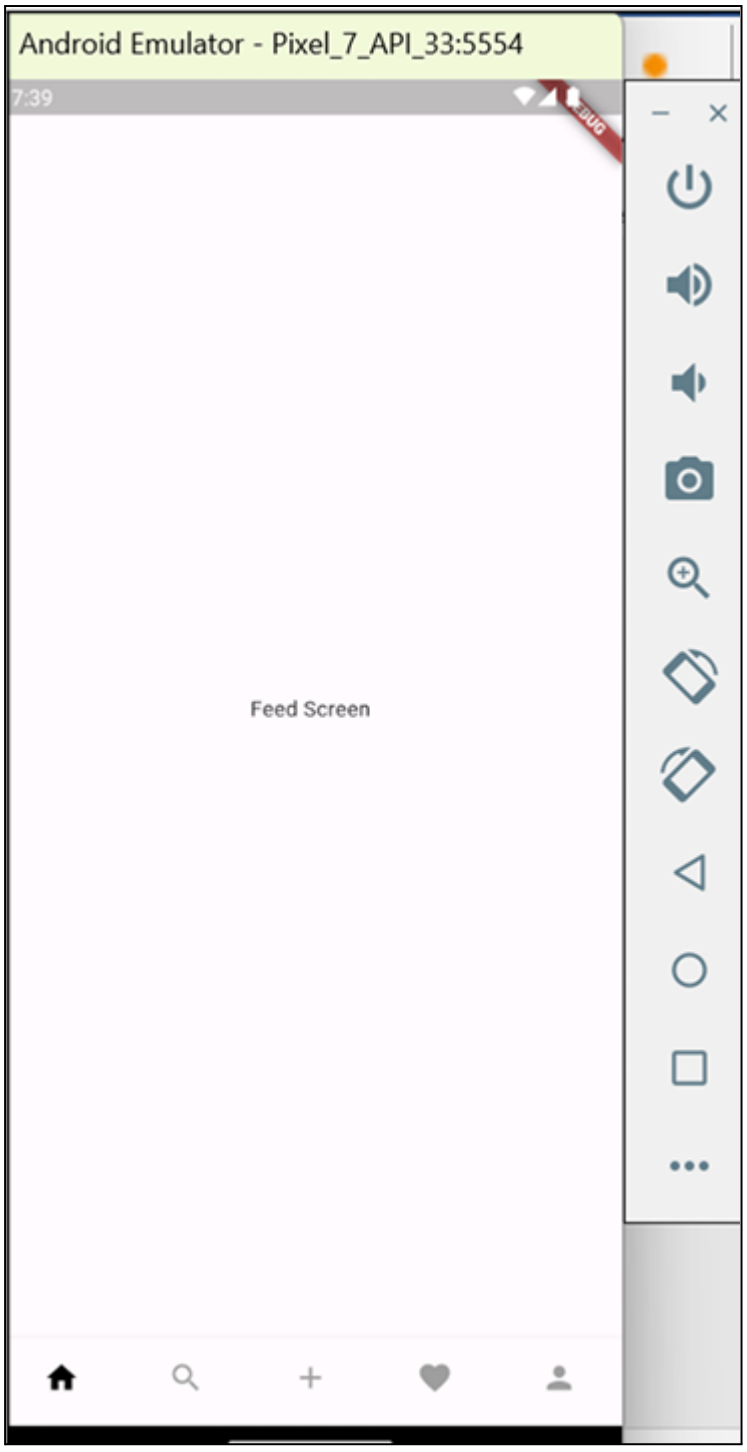
```
import 'package:flutter/material.dart';
import 'package:my_app/screens/favorite_screen.dart';
 import 'package:my_app/screens/feed.dart';
import 'package:my_app/screens/post_screen.dart';
import 'package:my_app/screens/profile_screen.dart';
import 'package:my_app/screens/search.dart';

class Home extends StatefulWidget {
```

```
const Home({Key? key}) : super(key: key);

@override

State createState() => _HomeState();
}

class _HomeState extends State {

int selectedIndex = 0; List pages = [ const FeedScreen(),
const SearchScreen(),
const PostScreen(),
const FavoriteScreen(),
const ProfileScreen(),
]; // You need to populate this list with your pages

@override

Widget build(BuildContext context) {

return Scaffold( body: pages[selectedIndex],
bottomNavigationBar: BottomNavigationBar(
currentIndex: selectedIndex, selectedItemColor: Colors.black,
unselectedItemColor: Colors.grey, showSelectedLabels: false,
showUnselectedLabels: false, onTap: (index){

 setState(() { selectedIndex=index; }); }, type: BottomNavigationBarType.fixed,
items: const [ BottomNavigationBarItem(icon: Icon(Icons.home),label: 'Feed',),
BottomNavigationBarItem(icon: Icon(Icons.search),label: 'Search',),
BottomNavigationBarItem(icon: Icon(Icons.add),label: 'post',),
BottomNavigationBarItem(icon: Icon(Icons.favorite),label: 'favorite',),
BottomNavigationBarItem(icon: Icon(Icons.person),label: 'profile',), ], ), );
 }
 }
```

feed.dart

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

```
class FeedScreen extends StatefulWidget {
const FeedScreen({super.key});

@override

State createState() => _FeedScreenState();

}
class _FeedScreenState extends State {

@override

Widget build(BuildContext context) {
 return const Scaffold( body: Center( child: Text("Feed Screen"),
),
);
 }
}
```



search.dart

```
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:thread_clone_flutter/model/user.dart';

class SearchScreen extends StatefulWidget {
```

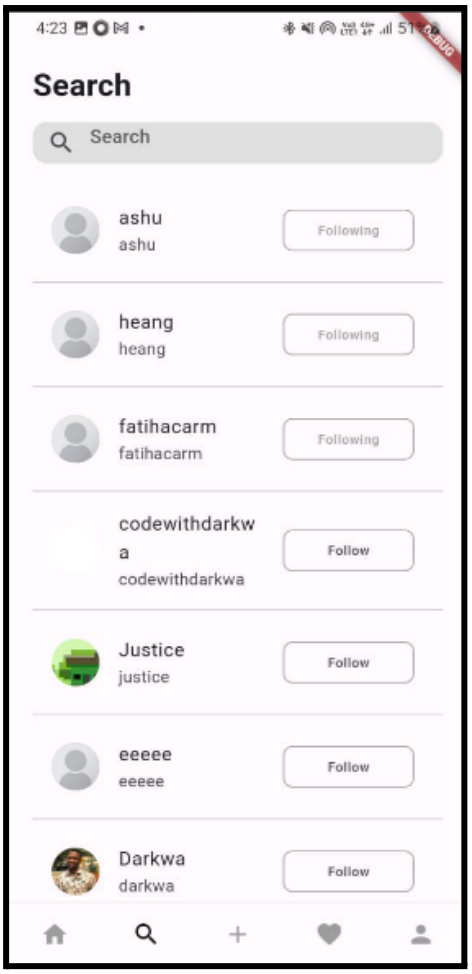
```
  const SearchScreen({super.key});

  @override
  State<SearchScreen> createState() => _SearchScreenState();
}

class _SearchScreenState extends State<SearchScreen> {
  final CollectionReference userCollection =
      FirebaseFirestore.instance.collection('users');

  final userId = FirebaseAuth.instance.currentUser!.uid;

  String searchQuery = "";
  final searchController = TextEditingController();
  List<UserModel> searchUsers(List<UserModel> users, String query) {
    return users.where((user) {
      return user.username.toLowerCase().contains(query.toLowerCase());
    }).toList();
  }

  Future<void> followUser(UserModel user) async {
    await userCollection.doc(userId).update({
      'following': FieldValue.arrayUnion([user.id])
    });
    await userCollection.doc(user.id).update({
      'followers': FieldValue.arrayUnion([userId])
    });
  }

  Future<void> unFollowUser(UserModel user) async {
    await userCollection.doc(userId).update({
      'following': FieldValue.arrayRemove([user.id])
    });
    await userCollection.doc(user.id).update({
      'followers': FieldValue.arrayRemove([userId])
    });
  }

  @override
  void initState() {
    searchController.addListener(() {
      setState(() {
        searchQuery = searchController.text;
      });
```

```
  });
  super.initState();
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    body: SafeArea(
      child: SingleChildScrollView(
        child: Padding(
          padding: const EdgeInsets.all(20.0),
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              const Text(
                'Search',
                style: TextStyle(fontSize: 26, fontWeight: FontWeight.bold),
              ),
              Padding(
                padding: const EdgeInsets.only(top: 12.0),
                child: Container(
                  width: double.infinity,
                  height: 35,
                  decoration: BoxDecoration(
                    color: Colors.grey[300],
                    borderRadius: BorderRadius.circular(12),
                  ),
                  child: TextFormField(
                    controller: searchController,
                    decoration: const InputDecoration(
                      hintText: 'Search',
                      border: InputBorder.none,
                      prefixIcon: Icon(Icons.search),
                    ),
                  ),
                ),
              ),
              const SizedBox(height: 20),
              StreamBuilder(
                stream: userCollection
                    .where('id', isNotEqualTo: userId)
                    .snapshots(),
                builder: (context, snapshot) {
                  if (!snapshot.hasData) {
```

```
                return const Center(
                  child: CircularProgressIndicator(),
                );
              } else if (snapshot.hasError) {
                return Center(
                  child: Text('Error: ${snapshot.error}'),
                );
              }
              final users = snapshot.data!.docs;

              final allUsers = users.map((doc) {
                final user = doc.data() as Map<String, dynamic>;
                return UserModel(
                  id: user['id'],
                  username: user['username'],
                  profileImageUrl: user['profileImageUrl'],
                  name: user['name'],
                  followers: [],
                  following: [],
                );
              }).toList();
              final filteredUsers = searchUsers(allUsers, searchQuery);
              return ListView.builder(
                shrinkWrap: true,
                itemCount: filteredUsers.length,
                itemBuilder: (contex, index) {
                  final user = filteredUsers[index];

                  return SuggestedFollowerWidget(
                    user: user,
                    follow: () => followUser(user),
                    unFollow: () => unFollowUser(user),
                  );
                },
              );
            },
          )
        ],
      ),
    ),
  ),
);
}
```

```
}

class SuggestedFollowerWidget extends StatefulWidget {
  const SuggestedFollowerWidget({
    super.key,
    required this.user,
    required this.follow,
    required this.unFollow,
  });

  final UserModel user;
  final VoidCallback follow;
  final VoidCallback unFollow;

  @override
  State<SuggestedFollowerWidget> createState() =>
      _SuggestedFollowerWidgetState();
}

class _SuggestedFollowerWidgetState extends State<SuggestedFollowerWidget> {
  @override
  Widget build(BuildContext context) {
    final userId = FirebaseAuth.instance.currentUser!.uid;
    return Column(
      children: [
        ListTile(
          leading: CircleAvatar(
            backgroundImage: NetworkImage(widget.user.profileImageUrl ??

"https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcRz8cLf8-P2P8GZ0-KiQ-O
XpZQ4bebpa3K3Dw&usqp=CAU"),
            backgroundColor: Colors.white,
          ),
          title: Text(widget.user.username),
          subtitle: Text(widget.user.username.toLowerCase()),
          trailing: StreamBuilder(
              stream: FirebaseFirestore.instance
                  .collection('users')
                  .doc(userId)
                  .snapshots(),
              builder: (context, snapshot) {
                if (!snapshot.hasData) {
                  return const SizedBox.shrink();
                }
```

```
            final currentUser = UserModel.fromMap(
                snapshot.data!.data() as Map<String, dynamic>);
            final isFollowing =
                currentUser.following.contains(widget.user.id);
            return Row(
              mainAxisSize: MainAxisSize.min,
              children: [
                InkWell(
                  onTap: isFollowing ? widget.unFollow : widget.follow,
                  child: Container(
                    alignment: Alignment.center,
                    width: 110,
                    height: 35,
                    decoration: BoxDecoration(
                      border: Border.all(color: Colors.grey),
                      borderRadius: BorderRadius.circular(8),
                    ),
                    child: isFollowing
                        ? const Text(
                            'Following',
                            style: TextStyle(color: Colors.grey),
                          )
                        : const Text('Follow'),
                  ),
                )
              ],
            );
          }),
      ),
      const Divider(),
    ],
  );
 }
}
```



CONCLUSION

In conclusion, the process of designing a Flutter UI by incorporating common widgets has proven to be effective and user-friendly.I learned many new concepts in flutter widgets.