

EXPERIMENT 2

Aim : To design Flutter UI by including common widgets.
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.

Category of Widgets:

There are mainly 14 categories in which the flutter widgets are divided. They are mainly segregated on the basis of the functionality they provide in a flutter application.

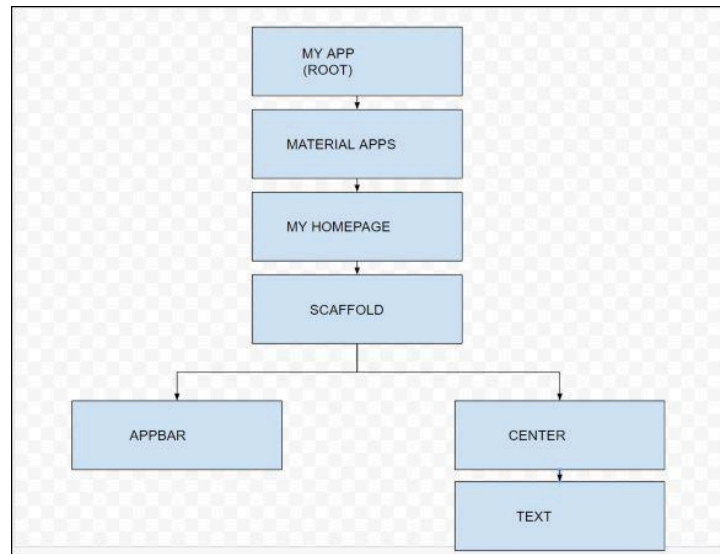
1. *Accessibility*: These are the set of widgets that make a flutter app more easily accessible.
2. *Animation and Motion*: These widgets add animation to other widgets.
3. *Assets, Images, and Icons*: These widgets take charge of assets such as display images and show icons.
4. *Async*: These provide async functionality in the flutter application.
5. *Basics*: These are the bundle of widgets that are absolutely necessary for the development of any flutter application.
6. *Cupertino*: These are the iOS designed widgets.
7. *Input*: This set of widgets provides input functionality in a flutter application.

8. *Interaction Models*: These widgets are here to manage touch events and route users to different views in the application.
9. *Layout*: This bundle of widgets helps in placing the other widgets on the screen as needed.
10. *Material Components*: This is a set of widgets that mainly follow material design by Google.
11. *Painting and effects*: This is the set of widgets that apply visual changes to their child widgets without changing their layout or shape.
12. *Scrolling*: This provides scrollability of to a set of other widgets that are not scrollable by default.
13. *Styling*: This deals with the theme, responsiveness, and sizing of the app.
14. *Text*: This displays text.

Types of Widgets:

There are broadly two types of widgets in the flutter:

1. Stateless Widget - These are immutable widgets that don't change over time.
 - The UI of a stateless widget is defined based on the configuration information passed to it during its creation.
 - Example: Container, Text, Icon.
2. Stateful Widget - These are mutable widgets that can change dynamically.
 - Stateful widgets maintain a mutable state that might change during the widget's lifetime.
 - Example: TextField, ListView, Form.



WIDGET

Code:

1. Login Page -

```
login_screen.dart X
news > lib > src > features > authentication > presentation > register > screens > login_screen.dart > _LoginScreen > build
12 import '../components/logIn_component.dart';
13
14 class LoginScreen extends StatelessWidget {
15   const LoginScreen({super.key});
16
17   static const routeName = '/login';
18
19   @override
20   Widget build(BuildContext context) {
21     return BlocProvider<RegisterBloc>(
22       create: (context) => RegisterBloc(context),
23       child: const _LoginScreen(),
24     ); // BlocProvider
25   }
26 }
27
28 class _LoginScreen extends StatelessWidget {
29   const _LoginScreen({Key? key}) : super(key: key);
30
31   @override
32   Widget build(BuildContext context) {
33     final registerBloc = BlocProvider.of<RegisterBloc>(context);
34     return BlocBuilder<AuthBloc, AuthState>(builder: (_, state) {
35       if (state is LoadingAuthState) {
36         return const Scaffold(
37           body: Center(
38             child: CircularProgressIndicator(),
39           ), // Center
40         ); // Scaffold
41       } else {
42         return Scaffold(
43           body: Padding(
```

2. Sign Up Page:

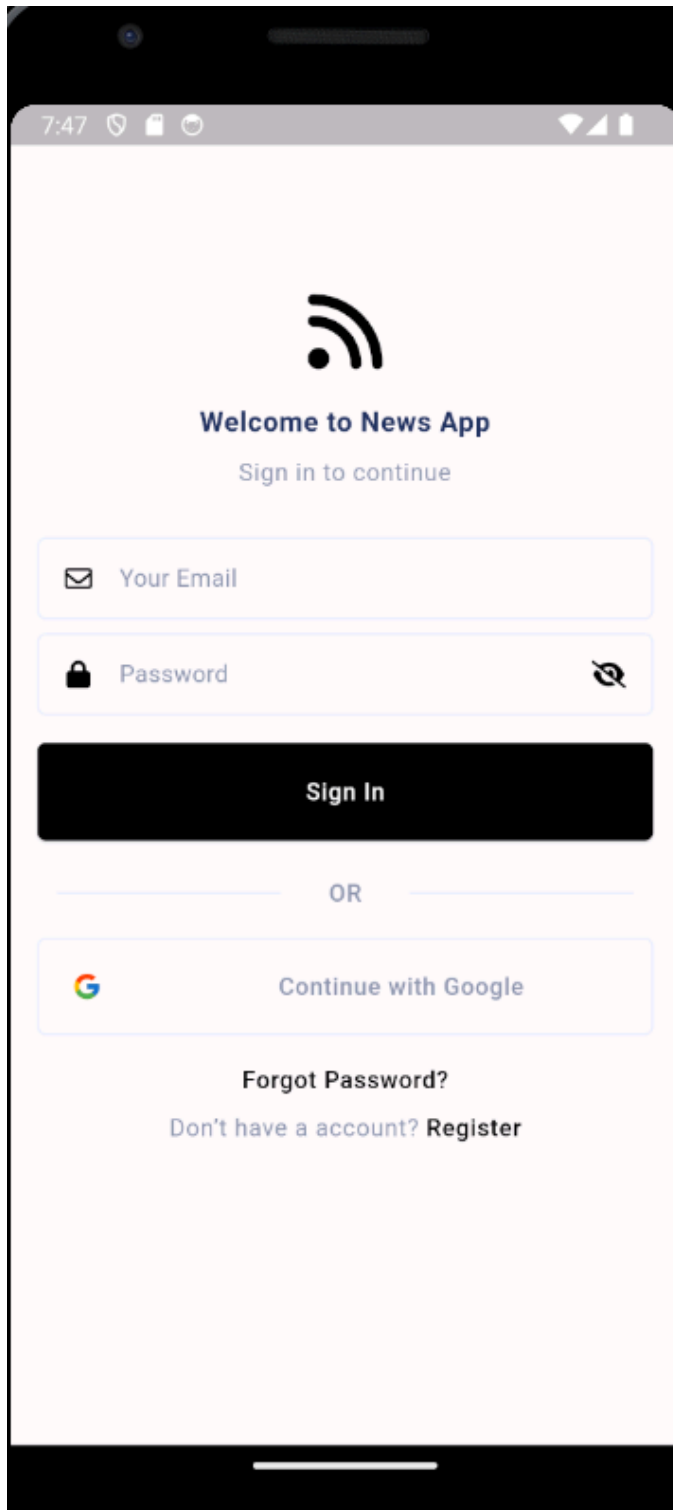
```
signup_screen.dart X
news > lib > src > features > authentication > presentation > register > screens > signup_screen.dart > SignUpScreen > SignUpScreen
10 import '../bloc/register_bloc.dart';
11 import '../components/logIn_component.dart';
12
13 class SignUpScreen extends StatelessWidget {
14   const SignUpScreen({super.key});
15
16   static const routeName = '/signup';
17
18   @override
19   Widget build(BuildContext context) {
20     return BlocProvider<RegisterBloc>(
21       create: (context) => RegisterBloc(context),
22       child: const _SignUpScreen(),
23     ); // BlocProvider
24   }
25 }
26
27 class _SignUpScreen extends StatelessWidget {
28   const _SignUpScreen({Key? key}) : super(key: key);
29
30   @override
31   Widget build(BuildContext context) {
32     final registerBloc = BlocProvider.of<RegisterBloc>(context);
33     return BlocBuilder<AuthBloc, AuthState>(builder: (_, state) {
34       if (state is LoadingAuthState) {
35         return const Scaffold(
36           body: Center(
37             child: CircularProgressIndicator(),
38           ), // Center
39         ); // Scaffold
40       } else {
41         return Scaffold(
```

3. Forgot Password:

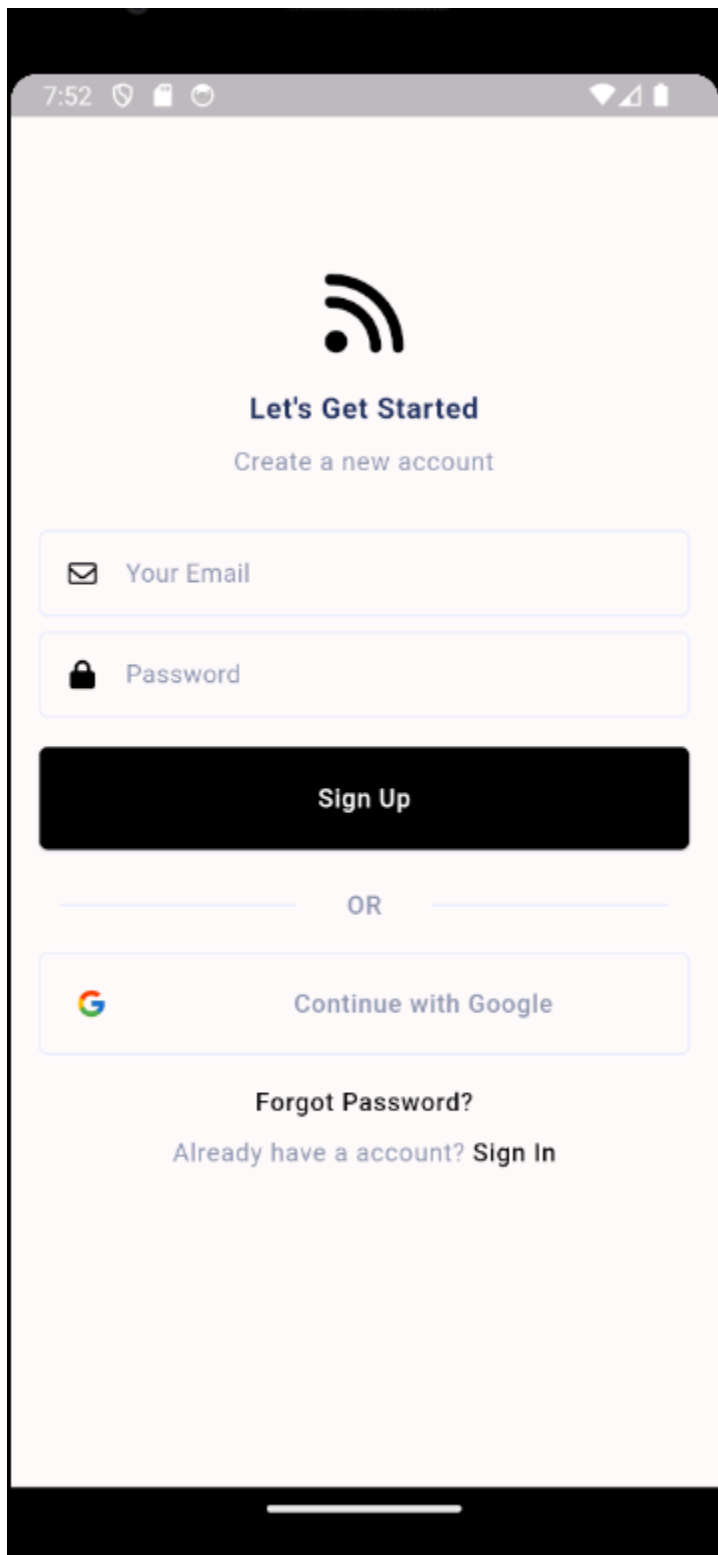
```
forget_password_screen.dart X
news > lib > src > features > authentication > presentation > register > screens > forget_password_screen.dart > ForgetPasswordScreen
9 import '...../bloc/auth/auth_state.dart';
10 import '../bloc/register_bloc.dart';
11
12 class ForgetPasswordScreen extends StatelessWidget {
13   const ForgetPasswordScreen({super.key});
14
15   static const routeName = '/forget-password';
16
17   @override
18   Widget build(BuildContext context) {
19     return BlocProvider<RegisterBloc>(
20       create: (context) => RegisterBloc(context),
21       child: const _ForgetPasswordScreen(),
22     ); // BlocProvider
23   }
24 }
25
26 class _ForgetPasswordScreen extends StatelessWidget {
27   const _ForgetPasswordScreen({Key? key}) : super(key: key);
28
29   @override
30   Widget build(BuildContext context) {
31     final registerBloc = BlocProvider.of<RegisterBloc>(context);
32     return BlocBuilder<AuthBloc, AuthState>(builder: (_, state) {
33       if (state is LoadingAuthState) {
34         return const Scaffold(
35           body: Center(
36             child: CircularProgressIndicator(),
37           ), // Center
38         ); // Scaffold
39       } else {
40         return Scaffold(
```

Output:

1. Login Page:




2. Sign Up Page:




A mobile application sign-up screen with a light pink background. At the top, a status bar shows the time 7:52 and various icons. Below the status bar is a black header bar. The main content area features a black RSS icon, the text "Let's Get Started" in bold, and "Create a new account" in a smaller font. There are two input fields: "Your Email" with an envelope icon and "Password" with a lock icon. A black "Sign Up" button is below the fields. A horizontal line with "OR" in the center separates the sign-up section from the sign-in section. The sign-in section includes a "Continue with Google" button with the Google logo and a "Forgot Password?" link. At the bottom, there is a link that says "Already have a account? Sign In". The screen is framed by a black border, and a white home indicator bar is visible at the very bottom.


7:52



Let's Get Started


Create a new account

 Your Email

 Password

Sign Up

OR

 Continue with Google

Forgot Password?

Already have a account? [Sign In](#)

Conclusion : Hence we have 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 login page.