

## Experiment No. 4

**AIM :** To create an interactive Form using form widget

### Theory:

#### 1. Form Widget:

- The Form widget is a container for managing form-related interactions. It allows for validation and saving the form data.
- It keeps track of the state of all the fields within the form (like TextFormField widgets) through a GlobalKey<FormState>.

#### 2. TextFormField:

- This is the main widget used for collecting user input (such as text). It integrates easily with form validation and submission.
- You can apply validators to the TextFormField to ensure the input meets specific criteria (e.g., required fields, correct format).

#### 3. GlobalKey:

- A GlobalKey<FormState> is essential for managing the form's state (e.g., validating fields, saving data). It's assigned to the Form widget and can be used to trigger actions like form validation or saving the data.

#### 4. Validation:

- You can define validation rules on each form field. The TextFormField widget has a validator property, which allows you to write logic that will run whenever the form is validated.
- A validator checks whether the input meets the required format (such as checking for valid email format or a non-empty field).

#### 5. Form Submission:

- When you're ready to submit the form, you can call formKey.currentState?.save() to trigger the save method for all fields or formKey.currentState?.validate() to check if all the fields pass the validation checks.

#### 6. Saving Data:

- After validation, data entered in the form fields can be saved to variables or used for further processing, such as sending it to a server.

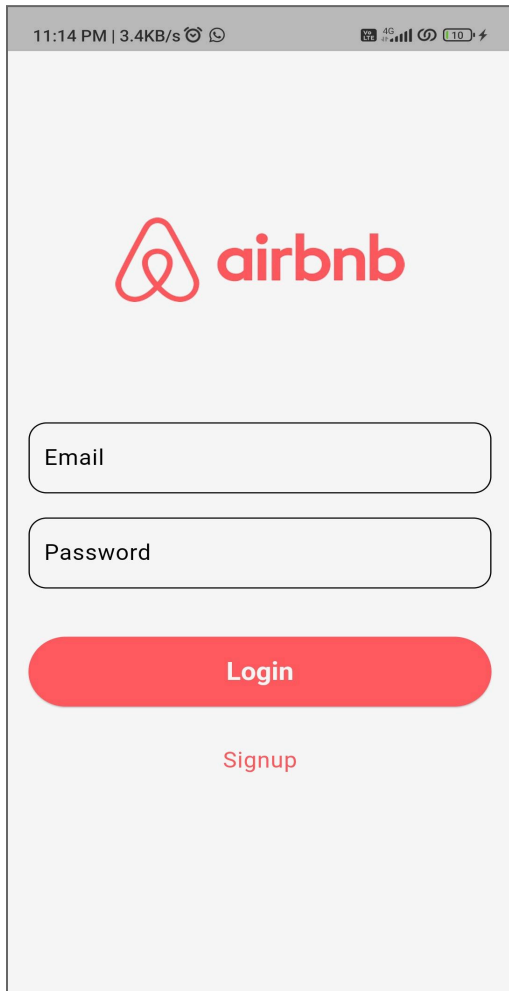
### Form Workflow:

1. **Create a form:** Use the Form widget to wrap all input fields.
2. **Add form fields:** Use widgets like TextFormField to collect data.
3. **Set up validation:** Add validation logic to the form fields.


4. **Submit the form:** Trigger validation and handle the form submission.
5. **Save data:** Capture the entered data once validation passes.

Flutter's form management tools are powerful, enabling you to handle complex forms with various input types, validations, and submissions effectively.

### Screenshots:



11:14 PM | 3.4KB/s



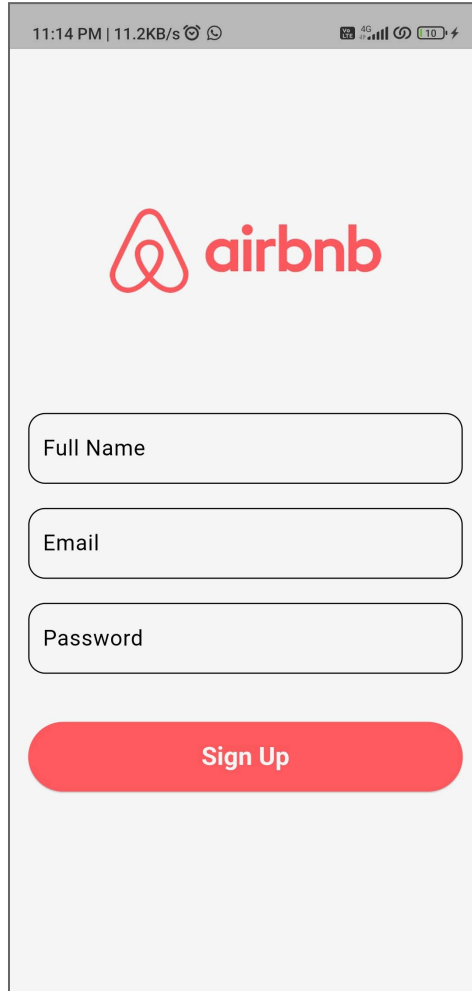
Email

Password


Login

Signup

This screenshot shows the Airbnb login interface. At the top, the status bar displays the time as 11:14 PM and a data speed of 3.4KB/s. The Airbnb logo is centered at the top. Below it, there are two input fields: 'Email' and 'Password'. A red 'Login' button is positioned below the password field, and a red 'Signup' link is located further down.



11:14 PM | 11.2KB/s



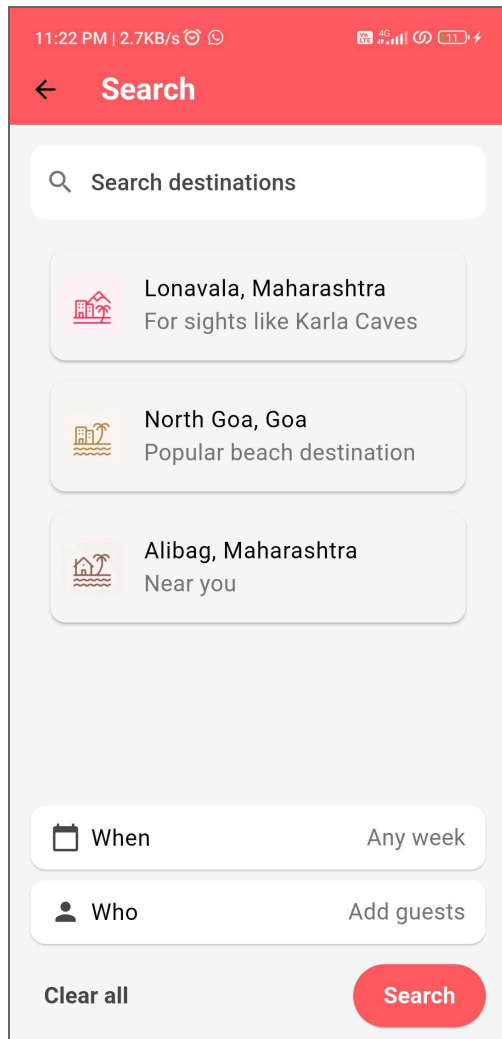
Full Name

Email

Password

Sign Up

This screenshot shows the Airbnb sign-up interface. The status bar at the top shows the time as 11:14 PM and a data speed of 11.2KB/s. The Airbnb logo is centered at the top. Below it, there are three input fields: 'Full Name', 'Email', and 'Password'. A red 'Sign Up' button is positioned below the password field.



## Code Snippets:

```
import 'package:airbnb/screens/signup.dart';
import 'package:flutter/material.dart';
import 'package:airbnb/theme.dart';
import 'home_page.dart';

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

  @override
  State<LoginPage> createState() => _LoginPageState();
}
```

```

class _LoginPageState extends State<LoginPage> {
  final TextEditingController _emailController = TextEditingController();
  final TextEditingController _passwordController = TextEditingController();

  String? _errorText;

  void _handleLogin() {
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: backgroundColor,
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Image.asset(
              "assets/images/logo-airbnb.png",
              height: 70,
              fit: BoxFit.cover,
            ),
            const SizedBox(
              height: 100,
            ),
            TextField(
              controller: _emailController,
              decoration: InputDecoration(
                labelText: 'Email',
                labelStyle: myTheme.inputDecorationTheme.labelStyle,
                hintText: 'Enter your email',
                hintStyle: myTheme.inputDecorationTheme.hintStyle,
                border: myTheme.inputDecorationTheme.border,
                focusedBorder:
                  Theme.of(context).inputDecorationTheme.focusedBorder,
              ),
              keyboardType: TextInputType.emailAddress,
            ),
            const SizedBox(height: 20),
            TextField(
              controller: _passwordController,
              decoration: InputDecoration(
                labelText: 'Password',

```

```

        labelStyle: myTheme.inputDecorationTheme.labelStyle,
        hintText: 'Password',
        hintStyle: myTheme.inputDecorationTheme.hintStyle,
        border: myTheme.inputDecorationTheme.border,
        focusedBorder:
            Theme.of(context).inputDecorationTheme.focusedBorder,
    ),
    obscureText: true,
),
if (_errorText != null) ...[
    const SizedBox(height: 10),
    Text(_errorText!, style: TextStyle(color: Colors.red)),
],
const SizedBox(height: 40),
SizedBox(
    width: double.infinity,
    child: ElevatedButton(
        onPressed: _handleLogin,
        style: ElevatedButton.styleFrom(
            backgroundColor: primaryColor,
            padding:
                const EdgeInsets.symmetric(vertical: 16, horizontal: 40),
        ),
        child: Text(
            'Login',
            style: Theme.of(context).textTheme.displaySmall?.copyWith(
                color: Theme.of(context).colorScheme.onPrimary,
            ),
        ),
    ),
),
const SizedBox(height: 20),
SizedBox(
    width: double.infinity,
    child: TextButton(
        onPressed: () {
            Navigator.push(
                context,
                MaterialPageRoute(builder: (context) => const SignUpPage()),
            );
        },
        child: Text(
            'Signup',
            style: Theme.of(context).textTheme.bodyLarge?.copyWith(

```

```

        color: primaryColor,
      ),
    ),
  ),
],
),
),
);
}
}

```

```

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

  @override
  State<SignUp> createState() => _SignUpState();
}

```

```

class _SignUpState extends State<SignUp> {
  @override
  Widget build(BuildContext context) {
    return const Placeholder();
  }
}

```