MAD PWA LAB 4

NAME: Bhavesh Wadhwani
Batch : C
Class : D15A
Roll No. : 69

Aim: To design Flutter UI by including common widgets.

Widgets:

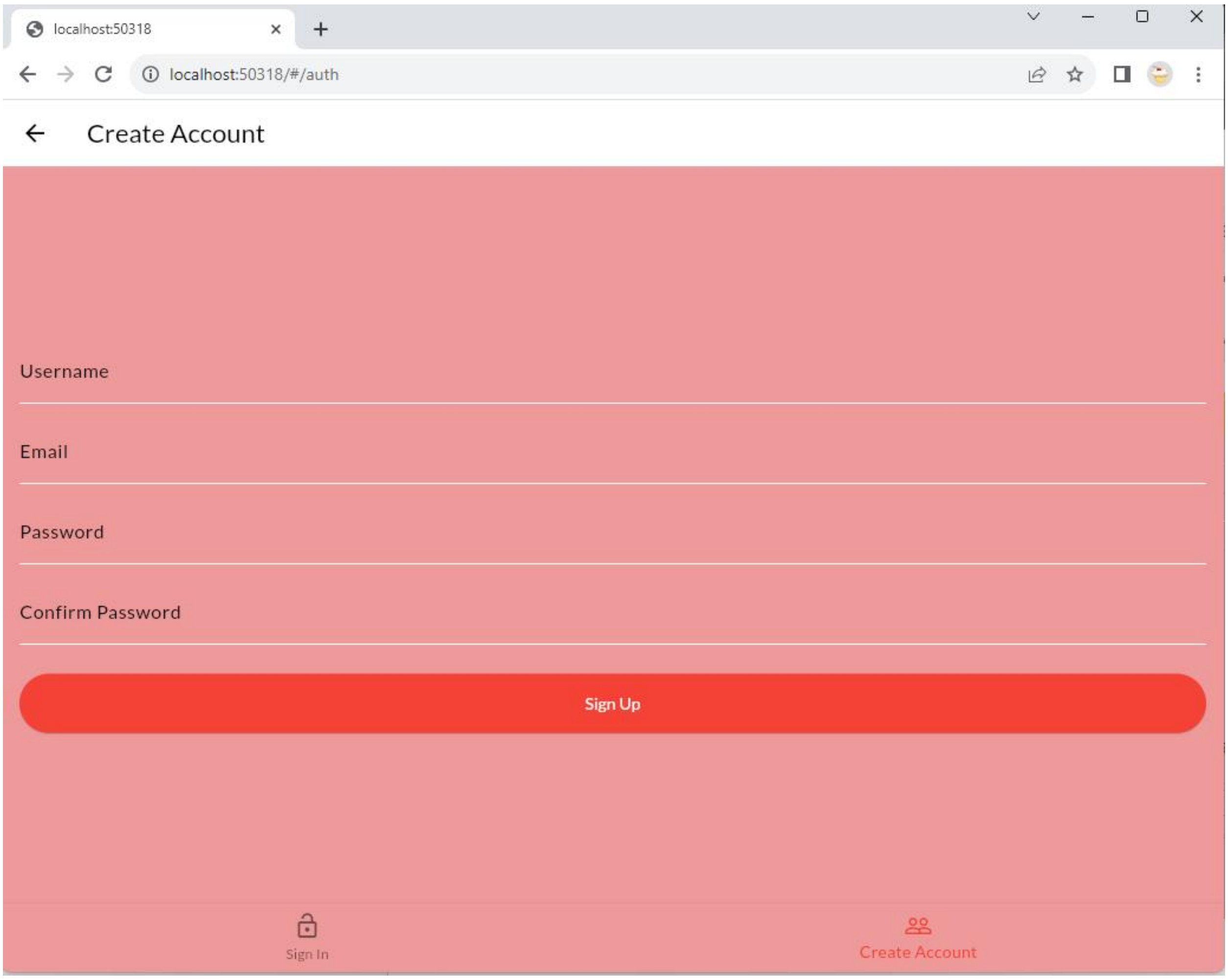
1. Column
2. Container
3. Row
4. Stack
5. Padding
6. Textfield
7. Button

Code:

```
import 'dart:js_interop';

// import 'package:firebase_auth/firebase_auth.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:flutter_ecommerce_ui_kit/blocks/auth_block.dart';
import 'package:flutter_ecommerce_ui_kit/models/user1.dart';
import 'package:provider/provider.dart';

class SignUp extends StatefulWidget {
 @override
 _SignUpState createState() => _SignUpState();
}

class _SignUpState extends State<SignUp> {
 final _formKey = GlobalKey<FormState>();
 final User1 user = User1(password: '', username: '', email: '');
 late String confirmPassword;

 @override
 Widget build(BuildContext context) {
   return Center(
     child: Form(
       key: _formKey,
       child: SingleChildScrollView(
child: Padding(
  padding: const EdgeInsets.only(left: 15.0, right: 15.0),
  child: Column(
    mainAxisAlignment: MainAxisAlignment.center,
    children: <Widget>[
      Padding(
        padding: const EdgeInsets.only(bottom: 12.0),
        child: TextFormField(
          validator: (value) {
            if (value!.isEmpty) {
              return 'Please Enter Username';
            }
            return null;
          },
          onSaved: (value) {
            setState(() {
              user.username = value!;
            });
          },
          decoration: InputDecoration(
            hintText: 'Enter Username',
            labelText: 'Username',
          ),
        ),
      ),
      Padding(
        padding: const EdgeInsets.only(bottom: 12.0),
        child: TextFormField(
          validator: (value) {
            if (value!.isEmpty) {
              return 'Please Enter Email Address';
            }
            return null;
          },
          onSaved: (value) {
            setState(() {
              user.email = value!;
            });
          },
          decoration: InputDecoration(
            hintText: 'Enter Email',
            labelText: 'Email',
          ),
        ),
      ),
      Padding(
        padding: const EdgeInsets.only(bottom: 12.0),
        child: TextFormField(
            validator: (value) {
              if (value!.isEmpty) {
                return 'Please Enter Password';
              }
              return null;
            },
            onSaved: (value) {
              setState(() {
                user.password = value!;
              });
            },
            onChanged: (text) {
              user.password = text;
            },
            decoration: InputDecoration(
              hintText: 'Enter Password',
              labelText: 'Password',
            ),
            obscureText: true),
      ),
      TextFormField(
        validator: (value) {
          if (value!.isEmpty) {
            return 'Please Enter Confirm Password';
          } else if (user.password != confirmPassword) {
            return 'Password fields dont match';
          }
          return null;
        },
        onChanged: (text) {
          confirmPassword = text;
        },
                    decoration: InputDecoration(
                      hintText: 'Enter Same Password',
                      labelText: 'Confirm Password',
                    ),
                    obscureText: true,
                  ),
                  Padding(
                    padding: const EdgeInsets.only(top: 25.0),
                    child: SizedBox(
                      width: double.infinity,
                      height: 50,
                      child: Consumer<AuthBlock>(builder:
                          (BuildContext context, AuthBlock auth, Widget? child) {
                        return ElevatedButton(
                          style: ElevatedButton.styleFrom(
                            primary: Theme.of(context).primaryColor,
                          ),
                          child: auth.loading && auth.loadingType == 'register'
                              ? CircularProgressIndicator(
                                  valueColor:
                                      AlwaysStoppedAnimation<Color>(Colors.white),
                                )
                              : Text(
                                  'Sign Up',
                                  style: TextStyle(color: Colors.white),
                                ),
                          onPressed: () {
                            if (_formKey.currentState!.validate() &&
                                !auth.loading) {
                              _formKey.currentState!.save();
                              // If the form is valid, display a snackbar. In the real world,
                              // you'd often call a server or save the information in a
      database.
                              auth.register(user);
                            } else{
                              String email = user.email;
                              String password = user.password;

      FirebaseAuth.instance.createUserWithEmailAndPassword(email: email,
      password: password).then((credential) {
```

```
               // Handle successful user creation
               print('User ${credential.user!.uid} created successfully.');
              }).catchError((error) {
               // Handle error during user creation
               print('Error creating user: $error');
              });
              ScaffoldMessenger.of(context).showSnackBar(
               SnackBar(
                content: Text('Sign up successful'),
                backgroundColor: Colors.green, // Set color here
               ),
              );
             }
            },
           );
          }),
         ),
        )
       ],
      ),
     ),
    ),
   ),
  );
 }
}
```

Output:


localhost:50318
localhost:50318/#/auth
Create Account
Username
Email
Password
Confirm Password
Sign Up
Sign In
Create Account