**Mobile Application Development (CEEL 311)  
First Semester, A.Y.2023-2024**

**Name: Score: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Year & Section: 3-CS3 Prof. Name: Prof. Ralph Laurence Visaya**

**Instructions:** This final project examination consists of dart and flutter code for a basic login page application. Each group will be given a time to adjust and design own login page with good application of Dart and Flutter Widgets. Each group may create their own login page if they create the whole program.

**Group Members:**John Bernard Cabantog

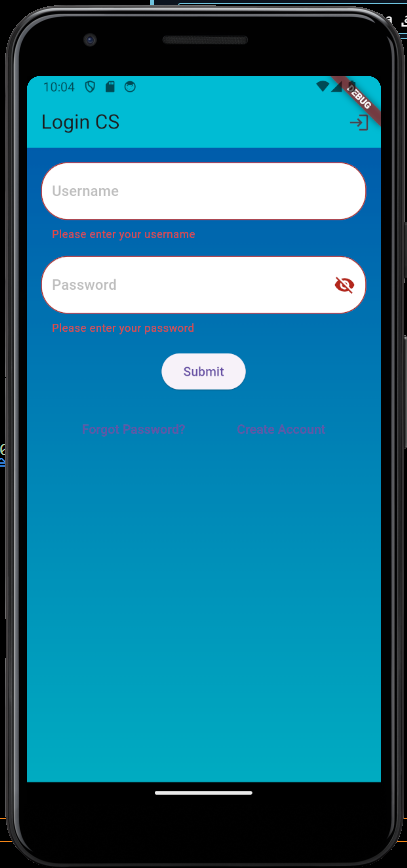
Renzy Gutierrez

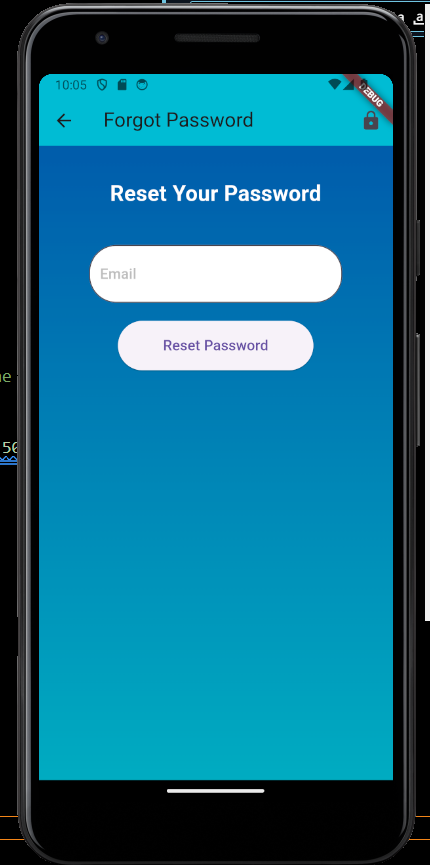
Denice Poblador

Marcos Reyes III.

John Simon Sapallo

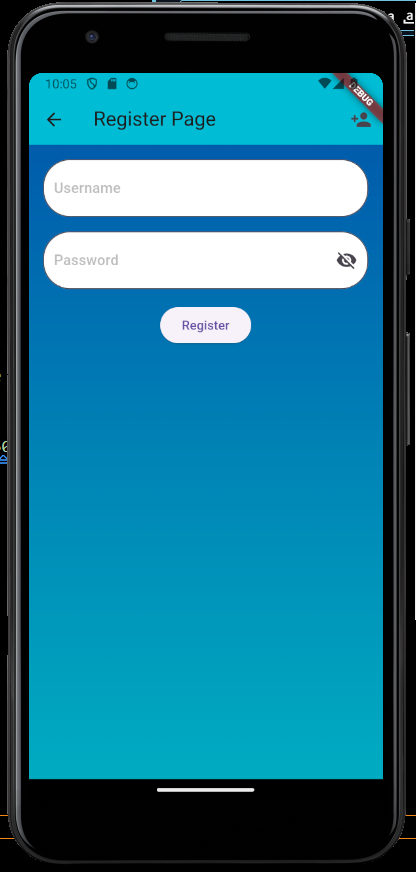
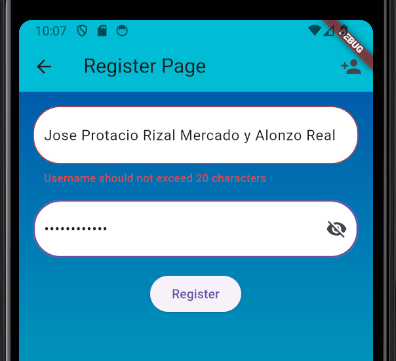
**Screenshots:**

****

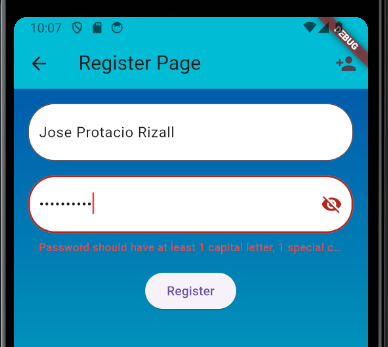
****

**// requires username and password**

**// reset password**

****

**// 20 max characters alphanumeric**

****

**// required to have a combination of 1 capital letter, 1 special character and alpanumeric**

**// registration**

**Code:**

|  |
| --- |
| import 'package:flutter/material.dart';  void main() {  runApp(LoginApp());  }  class LoginApp extends StatelessWidget {  @override  Widget build(BuildContext context) {  return MaterialApp(  title: 'Login Page',  theme: ThemeData(  primarySwatch: Colors.indigo,  ),  home: LoginPage(),  routes: {  '/register': (context) => RegisterPage(),  '/forgot\_password': (context) => ForgotPasswordPage(), // Added route for Forgot Password page  },  );  }  }  class LoginPage extends StatefulWidget {  @override  \_LoginPageState createState() => \_LoginPageState();  }  class \_LoginPageState extends State<LoginPage> {  final \_formKey = GlobalKey<FormState>();  final \_usernameController = TextEditingController();  final \_passwordController = TextEditingController();  bool \_isPasswordVisible = false; // Track password visibility  // Mock database  final List<String> \_users = [];  void \_togglePasswordVisibility() {  setState(() {  \_isPasswordVisible = !\_isPasswordVisible;  });  }  @override  Widget build(BuildContext context) {  return Scaffold(  resizeToAvoidBottomInset: false,  appBar: AppBar(  title: Text('Login CS'),  backgroundColor: Colors.cyan, // Background color suited for cyan background  actions: [  IconButton(  icon: Icon(Icons.login), // Icon for login page  onPressed: () {  // Add onPressed action for login icon if needed  },  ),  ],  ),  body: SizedBox.expand(  child: Container(  decoration: BoxDecoration(  gradient: LinearGradient(  begin: Alignment.topCenter,  end: Alignment.bottomCenter,  colors: [  Color.fromARGB(255, 0, 92, 171),  Color.fromARGB(255, 0, 172, 193),  ],  ),  ),  child: SingleChildScrollView(  child: Form(  key: \_formKey,  child: Padding(  padding: const EdgeInsets.all(16.0),  child: Column(  mainAxisAlignment: MainAxisAlignment.center,  children: <Widget>[  TextFormField(  controller: \_usernameController,  decoration: InputDecoration(  hintText: 'Username',  fillColor: Colors.white,  filled: true,  border: OutlineInputBorder(  borderRadius: BorderRadius.circular(30.0),  ),  hintStyle: TextStyle(color: Colors.grey[400]),  errorStyle: TextStyle(color: Colors.red),  ),  validator: (value) {  if (value!.isEmpty) {  return 'Please enter your username';  } else if (value.length > 20) {  return 'Username should not exceed 20 characters';  } else if (!RegExp(r'^[a-zA-Z0-9\s]+$').hasMatch(value)) {  return 'Username should contain only alphanumeric characters';  }  return null;  },  ),  SizedBox(height: 16.0),  TextFormField(  controller: \_passwordController,  decoration: InputDecoration(  hintText: 'Password',  fillColor: Colors.white,  filled: true,  border: OutlineInputBorder(  borderRadius: BorderRadius.circular(30.0),  ),  hintStyle: TextStyle(color: Colors.grey[400]),  errorStyle: TextStyle(color: Colors.red),  // Add suffixIcon for password visibility toggle  suffixIcon: IconButton(  icon: Icon(\_isPasswordVisible ? Icons.visibility : Icons.visibility\_off),  onPressed: \_togglePasswordVisibility,  ),  ),  obscureText: !\_isPasswordVisible, // Toggle password visibility  validator: (value) {  if (value!.isEmpty) {  return 'Please enter your password';  } else if (value.length < 8) {  return 'Password should be at least 8 characters long';  } else if (!RegExp(r'^(?=.\*?[A-Z])(?=.\*?[a-z])(?=.\*?[0-9])(?=.\*?[!@#$%^&\*()\_+]).{8,}$').hasMatch(value)) {  return 'Password should contain at least 1 uppercase letter, 1 lowercase letter, 1 digit and special character';  }  return null;  },  ),  SizedBox(height: 16.0),  ElevatedButton(  onPressed: () {  if (\_formKey.currentState!.validate()) {  if (\_users.contains(\_usernameController.text)) {  ScaffoldMessenger.of(context).showSnackBar(  SnackBar(content: Text('Processing Data')),  );  } else {  ScaffoldMessenger.of(context).showSnackBar(  SnackBar(content: Text('User does not exist')),  );  }  }  },  child: Text('Submit'),  ),  SizedBox(height: 16.0),  Row(  mainAxisAlignment: MainAxisAlignment.spaceEvenly, // Adjusted alignment  children: <Widget>[  TextButton(  onPressed: () {  Navigator.pushNamed(context, '/forgot\_password'); // Navigate to Forgot Password page  },  child: Text('Forgot Password?'), // Added "Forgot Password" link  ),  TextButton(  onPressed: () {  Navigator.pushNamed(context, '/register');  },  child: Text('Create Account'),  ),  ],  ),  ],  ),  ),  ),  ),  ),  ),  );  }  }  class RegisterPage extends StatefulWidget {  @override  \_RegisterPageState createState() => \_RegisterPageState();  }  class \_RegisterPageState extends State<RegisterPage> {  final \_formKey = GlobalKey<FormState>();  final \_usernameController = TextEditingController();  final \_passwordController = TextEditingController();  bool \_isPasswordVisible = false; // Track password visibility  // Access the mock database from LoginPage  final List<String> \_users = LoginPage().createState().\_users;  void \_togglePasswordVisibility() {  setState(() {  \_isPasswordVisible = !\_isPasswordVisible;  });  }  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: Text('Register Page'),  backgroundColor: Colors.cyan, // Background color suited for cyan background  actions: [  IconButton(  icon: Icon(Icons.person\_add), // Icon for register page  onPressed: () {  // Add onPressed action for register icon if needed  },  ),  ],  ),  body: SizedBox.expand(  child: Container(  decoration: BoxDecoration(  gradient: LinearGradient(  begin: Alignment.topCenter,  end: Alignment.bottomCenter,  colors: [  Color.fromARGB(255, 0, 92, 171),  Color.fromARGB(255, 0, 172, 193),  ],  ),  ),  child: SingleChildScrollView(  child: Form(  key: \_formKey,  child: Padding(  padding: const EdgeInsets.all(16.0), // Added padding around the form  child: Column(  mainAxisAlignment: MainAxisAlignment.center,  children: <Widget>[  // Styled TextFields for Username and Password  TextFormField(  controller: \_usernameController,  decoration: InputDecoration(  hintText: 'Username',  fillColor: Colors.white,  filled: true,  border: OutlineInputBorder(  borderRadius: BorderRadius.circular(30.0),  ),  hintStyle: TextStyle(color: Colors.grey[400]),  errorStyle: TextStyle(color: Colors.red),  ),  validator: (value) {  if (value!.isEmpty) {  return 'Please enter your username';  } else if (value.length > 20) {  return 'Username should not exceed 20 characters';  } else if (!RegExp(r'^[a-zA-Z0-9\s]+$').hasMatch(value)) {  return 'Username should contain only alphanumeric characters';  }  return null;  },  ),  SizedBox(height: 16.0),  TextFormField(  controller: \_passwordController,  decoration: InputDecoration(  hintText: 'Password',  fillColor: Colors.white,  filled: true,  border: OutlineInputBorder(  borderRadius: BorderRadius.circular(30.0),  ),  hintStyle: TextStyle(color: Colors.grey[400]),  errorStyle: TextStyle(color: Colors.red),  // Add suffixIcon for password visibility toggle  suffixIcon: IconButton(  icon: Icon(\_isPasswordVisible ? Icons.visibility : Icons.visibility\_off),  onPressed: \_togglePasswordVisibility,  ),  ),  obscureText: !\_isPasswordVisible, // Toggle password visibility  validator: (value) {  if (value!.isEmpty) {  return 'Please enter your password';  } else if (value.length < 8) {  return 'Password should be at least 8 characters long';  } else if (!RegExp(r'^(?=.\*?[A-Z])(?=.\*?[a-z])(?=.\*?[0-9])(?=.\*?[!@#$%^&\*()\_+]).{8,}$').hasMatch(value)) {  return 'Password should have at least 1 capital letter, 1 special character and alpanumeric';  }  return null;  },  ),  SizedBox(height: 16.0),  // Styled ElevatedButton for Register  ElevatedButton(  onPressed: () {  if (\_formKey.currentState!.validate()) {  \_users.add(\_usernameController.text);  ScaffoldMessenger.of(context).showSnackBar(  SnackBar(content: Text('User registered successfully')),  );  }  },  child: Text('Register'),  ),  ],  ),  ),  ),  ),  ),  ),  );  }  }  class ForgotPasswordPage extends StatelessWidget {  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: Text('Forgot Password'),  backgroundColor: Colors.cyan, // Background color suited for cyan background  actions: [  IconButton(  icon: Icon(Icons.lock), // Icon for forgot password page  onPressed: () {  // Add onPressed action for forgot password icon if needed  },  ),  ],  ),  body: SizedBox.expand(  child: Container(  decoration: BoxDecoration(  gradient: LinearGradient(  begin: Alignment.topCenter,  end: Alignment.bottomCenter,  colors: [  Color.fromARGB(255, 0, 92, 171),  Color.fromARGB(255, 0, 172, 193),  ],  ),  ),  child: SingleChildScrollView(  child: Padding(  padding: const EdgeInsets.all(16.0),  child: Column(  mainAxisAlignment: MainAxisAlignment.center,  children: <Widget>[  SizedBox(height: 20),  Text(  'Reset Your Password',  style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold, color: Colors.white),  ),  SizedBox(height: 40),  Padding(  padding: const EdgeInsets.symmetric(horizontal: 40),  child: TextFormField(  decoration: InputDecoration(  hintText: 'Email',  fillColor: Colors.white,  filled: true,  border: OutlineInputBorder(  borderRadius: BorderRadius.circular(30),  ),  hintStyle: TextStyle(color: Colors.grey[400]),  ),  validator: (value) {  // Add validation for email  if (value!.isEmpty) {  return 'Please enter your email';  } else if (!value.contains('@')) {  return 'Please enter a valid email';  }  return null;  },  ),  ),  SizedBox(height: 20),  ElevatedButton(  onPressed: () {  // Implement password reset logic here  // You may want to send a reset link to the provided email  },  style: ElevatedButton.styleFrom(  padding: EdgeInsets.symmetric(horizontal: 50, vertical: 16),  ),  child: Text(  'Reset Password',  style: TextStyle(fontSize: 16),  ),  ),  ],  ),  ),  ),  ),  ),  );  }  } |