package Package;

import javax.swing.JOptionPane;

public class Login {

// Private Fields for username, password, firstname and lastname of user

private String username;

private String password;

private String firstName;

private String lastName;

// Getters and Setters for username, password, firstName, and lastName fields

public String getUsername() { // Username

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) { // Password

this.password = password;

}

public String getFirstName() {

return firstName;

}

public void setFirstName(String firstName) { // First name

this.firstName = firstName;

}

public String getLastName() {

return lastName;

}

public void setLastName(String lastName) { // Last name

this.lastName = lastName;

}

// REGISTRATION:

/\*

\* Validation for username and password

\* Username must be <= 5 characters long and have at least 1 underscore

\* Password must be at least 8 characters long, contain 1 uppercase, 1 digit,

\* and 1 special character

\*/

public boolean checkUserName() {

return username.length() <= 5 && username.contains("\_");

}

public boolean checkPasswordComplexity() {

// These are like switches, they start false and flip to true when their

// condition is met, all switches are required to be true to validate password

boolean hasUppercase = false;

boolean hasDigit = false;

boolean hasSpecialCharacter = false;

char currentCharacterInPassword;

for (int i = 0; i < password.length(); i++) {

currentCharacterInPassword = password.charAt(i);

if (Character.isUpperCase(currentCharacterInPassword)) {

hasUppercase = true;

} else if (Character.isDigit(currentCharacterInPassword)) {

hasDigit = true;

} else if (!Character.isLetterOrDigit(currentCharacterInPassword)

&& !Character.isWhitespace(currentCharacterInPassword)) {

hasSpecialCharacter = true;

}

}

return password.length() >= 8 && hasUppercase && hasDigit && hasSpecialCharacter;

}

// Returns messages based on results of validation for username and password

public String registerUser() {

String usernameMessage;

String passwordMessage;

if (checkUserName()) {

usernameMessage = "Welcome " + firstName + " " + lastName + ", it is great to see you.";

} else {

usernameMessage = "Username is not correctly formatted, please ensure that your username contains an underscore and is at least 5 characters in length.";

}

if (checkPasswordComplexity()) {

passwordMessage = "Password successfully captured";

} else {

passwordMessage = "Password is not correctly formatted, please ensure that your password contains at least 8 characters, a capital letter, a number and a special character.";

}

return usernameMessage + " \n" + passwordMessage;

}

// LOGIN

// Verfies that login details entered match the details stored during

// registration

public boolean loginUser() {

// getting input from the user

String inputUsername = JOptionPane.showInputDialog(null, "Enter your username", "Username", 1);

String inputPassword = JOptionPane.showInputDialog(null, "Enter your password", "Password", 1);

// returning a boolean based on whether or not the username and password match

return inputUsername.equals(username) && inputPassword.equals(password);

}

// displaying messages based on the results of logging in

public String loginStatus(boolean login) {

String message;

if (login) {

message = "Logged in successfully";

} else {

message = "Unsuccessful Login, username or password is incorrect, please try again.";

}

return message;

}

}

package Package;

import javax.swing.JOptionPane;

public class RegisterOrLogin {

public static void main(String[] args) {

Login newUser = new Login();

String username;

String password;

String firstName;

String surname;

boolean registerUsername;

boolean registerPassword;

boolean correctLogin;

boolean cancel = false;

while (!cancel) {

String[] rolOption = { "Register", "Login", "Cancel" };

int registerOrLogin = JOptionPane.showOptionDialog(null,

"Register if you don't have an account\nOr Login to an existing account", "Register Or Login", 1, 1,

null, rolOption, rolOption[0]);

switch (registerOrLogin) {

case 0:

// Register

do {

// Prompting User for registration details

username = JOptionPane.showInputDialog("Register your username");

password = JOptionPane.showInputDialog("Register your password");

firstName = JOptionPane.showInputDialog("Register your first name");

surname = JOptionPane.showInputDialog("Register your surname");

// Setting inputted details

newUser.setUsername(username);

newUser.setPassword(password);

newUser.setFirstName(firstName);

newUser.setLastName(surname);

// Validating user inputs ---> username and password

registerUsername = newUser.checkUserName();

registerPassword = newUser.checkPasswordComplexity();

// Message displaying validation results

JOptionPane.showMessageDialog(null, newUser.registerUser(), "Registration Status",

registerUsername && registerPassword ? 1 : 0);

} while (!registerUsername || !registerPassword);

break;

case 1:

// Login

do {

correctLogin = newUser.loginUser();

JOptionPane.showMessageDialog(null, newUser.loginStatus(correctLogin), "Login Status",

correctLogin ? 1 : 0);

} while (!correctLogin);

break;

case 2:

cancel = true;

break;

default:

JOptionPane.showMessageDialog(null, "Please choose a valid option", "Error", 0);

break;

}

}

}

}