import java.util.ArrayList;

import java.util.Scanner;

//The User class holds details of a registered user.

class User {

private String username;

private String password;

private String firstname;

private String lastName;

public User(String username, String password, String firstName, String lastName) {

this.username = username;

this.password = password;

this.firstName = firstName;

this.lastName = lastName;

}

//Getters for user information.

public String getUsername() {return username; }

public String getPassword() { return password; }

public String getFirstName() { return firstName; }

public String getLastName() { return lastName; }

}

//The Login class encapsulates the registration and login logic.

class Login {

private User registeredUser;

//Checks that the username contains an underscore and is not more than 5 characters long.

public boolean checkUserName(String username) {

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

}

//Checks that the password meets the complexity requirement:

\*-At least 8 characters long

\*- Contains at least one capital letter

\*- Contains at least one digit

\*- Contains at least one special character (non-alphanumeric)

\*/

public boolean checkPasswordComplexity(String password) {

if(password.length() < 8) return false;

if (!password.matches(".\*[A-Z}.\*")) return false;

if(!password.matches(".\*\\d.\*")) return false;

if(!password.matches(".\*{^a-zA-Z0-9].\*")) return false;

return turn;

}

/\*\*

\* Attempts to register a user. Returns success if both the username and password meet the rules.

\* Otherwise returns an appropriate error message.

\*/

public String registerUser(String username, String password, String firstName, String lastName) {

boolean usernameValid = checkUserName(username);

boolean passwordValid = checkPasswordComplexity(password);

String error = "";

if(!usernameValid) {

error += "Username is not correctly formatted, please ensure that the password contains at least 8 charcters long; contain a capital letter; contain a number; contain a special vharacter.\n";

}

if(!passwordValid) {

error += "Password is not correctly please ensure that the password contains at least 8 characters long, contain a capital letter; contain a number; contains a special character.\n";

}

if(error.equals("")) {

registeredUser = new User(username, password, firstName, lastName);

return "The above two conditions have been met and user has been registered successfully.";

} else {

return error;

}

}

/\*\*

\* Checks entered login details against the stored user credentials.

\*/

public boolean loginUser(String username, String password) {

if(registeredUser != null &&

registeredUser.getUsername().equals(username)&&

registeredUser.getPassword().equals(password)) {

return true;

}

return false;

}

/\*\*

\* Returns a message that signals a successful or failed login attempt.

\*/

public String returnLoginStatus(boolean loginSuccess) {

if(loginSuccess) {

return "Welcome " + registeredUser.getFirstName() + "," + registeredUser.getLastName() + " it is great to see you again.";

}else {

return "Username or password incorrect, please try again.";

}

}

//The main class of the application holds main logic as well as simple unit tests.

public class QuickChatApp {

public static void main(String[] args) {

//Run the unit tests.

runTests();

Scanner scanner = new Scanner(System.in);

Login loginSystem = new Login();

//---Registration Phase ---

System.out.println("=== Registration ===");

System.out.print("Enter a username (must contain '\_' and be no more than 5 characters in length); ");

String username = scanner.nextLine();

System.out.print("Enter a password (minimum 8 characters, a captial letter, a number,a special character):");

String password = scanner.nextLine();

System.out.print("Enter your first name: ");

String firstName = scanner.nextLine();

System.out.print("Enter your last name: ");

String lastName = scanner.nextLine();

String registrationMessage= loginSystem.registerUser(username, password, firstName, lastName);

System.out.println(registrationMessage);

//--- Login Phase ---

if (registrationMessage.contains("registered successfully")) {

System.out.println("\n=== Login ===");

System.out.print("Username: ");

String loginUsername = scanner.nextLine();

System.out.print("Password: ");

string loginPassword = scanner.nextLine();

boolean loginSuccessful = loginSystem.loginUser(loginUsername, loginpassword);

String loginStatus = loginSystem.returnLoginStatus(loginSuccessful);

System.out.println(loginStatus);

//--- QuickChat Featre ---

if (loginSuccessful) {

chatMenu(scanner);

} else {

System.out.println("Login failed. Exiting application.");

} else {

System.out.println("Registration failed. Please correct the errors and try again.");

scanner.close();

}

/\*\*

\* Provides the Quickchat menu where a logged-in user can send messages.

\* Option 2 is a stub (Coming Soon) and the application continues running until quit.

\*/

public static void chatMenu(Scanner scanner){

ArrayList<String> messages = new ArrayList<>();

while (true) {

System.out.println("\n=== Welcome to QuicChat ===");

System.out.println("1. Send messages");

System.out.println("2. Show recently sent messages - Coming Soon");

System.out.println("3. Quit");

System.out.print("Choose an option: ");

}

int option = 0;

try {

option = Integer.parsenInt(scanner.nextLine());

} catch (NumberFormatException e) {

System.out.println("Please enter a valid number.");

continue;

}

if (option ==1) {

System.out.print("Enter your message: ");

String msg = scanner.nextLine();

messages.add(msg);

System.out.println("Message sent!");

} else if (option ==2) {

System.out.println("Coming Soon");

} else if (option ==3) {

System.out.println("Exiting QuickChat. Goodbye!");

break;

} else {

System.out.println("Invalid option,please try again.");

}

}

}

}

/\*\*

\* These simple unit tests check the login and registration methods.

\* Note: In a real project, you might use a testing framework like JUnit.

\*/

public static void runTests() {

System.out.println("Running unit tests...\n");

Login loginTest = new Login();

// Test 1: Username correctly formatted.

boolean test1 = loginTest.checkUsername("kyl\_1");

System.out.println("Test Username Valid (kyl\_"): " + (test1 ? "Passed" : "Failed"));"

//Test 2: Username incorrectly formatted.

boolean test2= loginTest.checkUserName("kyle!!!!!!!");

System.out.println("Test Username Invalid (kyle!!!!!!!): " + (!test2 ? "Passed" : "Failed"));

//Test 3: Passed correctly meets complexity requirements.

boolean test3 = loginTest.checkPasswordComplexity("Ch&&sec@ke99@!");

System.out.println("Test Password Valid (Ch&&sec@ke99!): " + (test3 ? "Passed" : "Failed"));

//Test 4: Passed that does not meet complexity requirements.

boolean test4 = loginTest.checkPasswordComplexity("password");

System.out.println("Test Password Invalid (password): " + (!test4 ? "Passed" : "Failed"));

//Test 5: Successful login test.

loginTest.registerUser("kyl\_1", "Ch&&sec@ke99!", "John", "Doe"); // registers user

boolean test5 = loginTest.loginUser("kyl\_1","Ch&&sec@ke99!");

System.out.println("Test Login Success (kyl\_1, Ch&&sec@ke99!"): + (test5 ? "Passed" : "Failed"));

//Test 6: Failed login test.

boolean test5 = loginTest.loginUser("kyl\_1","wrongpassword");

System.out.println("Test Login Fail (kyl\_1, wrongpassword): " + (!test6 ? "Passed" : "Failed"));

System.out.println("\nUnit tests completed.\n");