package chatapp;

import java.io.FileWriter;

import java.io.IOException;

import java.util.Scanner;

import javax.swing.JOptionPane;

import java.util.\*;

import java.io.\*;

import java.util.regex.\*;

import javax.swing.JOptionPane;

public class ChatAPP {

static String registeredUsername;

static String registeredPassword;

static String registeredCellPhoneNumber;

static String registeredFirstName;

static String registeredLastName;

private static String loginPass;

static boolean checkMessageID;

static boolean checkMessageHash;

static boolean checkRecipient;

static boolean checkMessage;

static String storeMessaege;

private static Object msg;

private static Object message;

//Username :Contains an underscore and is no more than five characters long

public static boolean checkUserName(String username){

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

}

//Password

public static boolean checkPasswordComplexity(String password){

boolean hasUpper = false,hasDigit = false, hasSpecial = false,hasEight = false;

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

for (char ch : password.toCharArray()){

if (Character.isUpperCase(ch)) hasUpper = true;

else if (Character.isDigit(ch)) hasDigit = true;

else if (Character.isLetterOrDigit(ch)) hasSpecial = true;

}

return hasUpper &&hasDigit && hasSpecial;

}

//cellphone starts with +27 and

public static boolean checkCellPhoneNumber(String cell){

return cell.startsWith("+27") && cell.length() <= 13;

}

//REGISTRATION METHOD

public static String registerUser(String username,String password ,String cell, String email){

if (checkPasswordComplexity(password)){

return"Password is not correctly formatted, please ensure that the password contains at least eight characters, Capital letter and special character";

}

if (checkCellPhoneNumber(cell)){

return"Cellphone number incorectly formatted.Must start with +27 and max 13 digits";

}

return "All details valid. Registration successful!";

}

// login

public static boolean loginUser(String username, String password ){

if (username.equals(registeredUsername)) {

password.equals(registeredPassword);

}

return false;

}

public static String returnLoginStatus(boolean loginSuccess){

if (loginSuccess){

return "Welcome "+ registeredFirstName +"" +registeredLastName +" It is nice to see you again";

}else{

return "Username or password is incorrect, Please try again.";

}

}

// Display user profile

public static void showUserProfile(){

System.out.println("Full name: "+ registeredFirstName +""+ registeredLastName);

System.out.println("Username :"+ registeredUsername);

System.out.println("Cell No."+ registeredCellPhoneNumber);

}

//CheckMessageID

public static boolean checkMessageID(String MessageID){

return MessageID.length() <= 10;

}

//CheckRecipientCell

public static boolean checkRecipientCell(String RecipientCell){

return RecipientCell.startsWith("+27") && RecipientCell.length() <= 10;

}

//Message Hash

public static boolean sentMessage(String sentMessage){

return

}

public static final File FILE = new File("meassages.json");

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

//Step 1 : Registration

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

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

//Step 1 : Registration

System.out.println(" User Registration form");

JOptionPane.showInputDialog("Enter your first name: ");

registeredFirstName = input.nextLine();

JOptionPane.showInputDialog("Enter your last name: ");

registeredLastName = input.nextLine();

JOptionPane.showInputDialog("Create Username(underscore & max 5 characters): ");

registeredUsername = input.nextLine();

JOptionPane.showInputDialog("Create password (8+ characters, 1 Uppercase, 1 digit,1 special char: ");

registeredPassword = input.nextLine();

JOptionPane.showInputDialog("Enter Cellphone (+27..): ");

registeredCellPhoneNumber = input.nextLine();

if (registeredCellPhoneNumber()){

JOptionPane.showMessageDialog(null,"Registration successful");

}else{

JOptionPane.showMessageDialog(null,"Registration failed");

}

System.out.println("Enter Username:");

if (user.username()){

JOptionPane.showMessageDialog(null,"username is incorrectly formated");

return;

}

System.out.println("Enter Password: ");

if (user.checkPassword()){

JOptionPane.showMessageDialog(null,"password is incorrectly formated");

return;

String name = JOptionPane.showInputDialog("Enter your first name:");

String loginPass = JOptionPane.showInputDialog("Enter password to login:");

boolean loggedIn = loginUser(LoginUser, loginPass);

System.out.println(returnLoginStatus(loggedIn));

if(loggedIn){

showUserProfile();

}

input.close();

// Step 1: Show current contents (or that it's empty)

if (!file.exists() || file.length() == 0) {

JOptionPane.showMessageDialog(null, "No messages found in messages.json.", "Info",

JOptionPane.INFORMATION\_MESSAGE);

} else {

String content = readFromJsonFile(file);

JOptionPane.showMessageDialog(null, "Existing message:\n" + content, "Stored Message",

JOptionPane.INFORMATION\_MESSAGE);

}

// Step 2: Get new message from user

String userMessage = JOptionPane.showInputDialog(null, "Enter a message to store:", "Input

Message", JOptionPane.QUESTION\_MESSAGE);

if (userMessage != null && !userMessage.trim().isEmpty()) {

String jsonMessage = "{\n \"message\": \"" + escapeJson(userMessage) + "\"\n}";

writeToJsonFile(file, jsonMessage);

JOptionPane.showMessageDialog(null, "Message saved to messages.json", "Success",

JOptionPane.INFORMATION\_MESSAGE);

// Step 3: Show saved message

String content = readFromJsonFile(file);

JOptionPane.showMessageDialog(null, "Saved Message:\n" + content, "Stored Message",

JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(null, "No message entered.", "Cancelled",

JOptionPane.INFORMATION\_MESSAGE);

}