/\*

\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/Project/Maven2/JavaApp/src/main/java/${packagePath}/${mainClassName}.java to edit this template

\*/

package com.mycompany.nkcubekopoepart1;

import java.util.ArrayList;

import javax.swing.JOptionPane;

/\*\*

\*

\* @author Nkcubeko

\*/

public class NkcubekoPOEpart1 {

public static class ToLogin {

private final String username;

private final String password;

private final String firstName;

private final String lastName;

private final ArrayList<Task> tasks;

private String[] developers;

private String[] taskNames;

private int[] taskIDs;

private int[] taskDuration;

private String[] taskStatus;

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

this.username = username;

this.password = password;

this.firstName = firstName;

this.lastName = lastName;

this.tasks = new ArrayList<>();

this.developers = new String[100]; // Adjust the size as needed

this.taskNames = new String[100]; // Adjust the size as needed

this.taskIDs = new int[100]; // Adjust the size as needed

this.taskDuration = new int[100]; // Adjust the size as needed

this.taskStatus = new String[100]; // Adjust the size as needed

}

public boolean checkUserName() {

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

}

public boolean checkPasswordComplexity() {

boolean hasCapitalLetter = false;

boolean hasNumber = false;

boolean hasSpecialCharacter = false;

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

if (Character.isUpperCase(c)) {

hasCapitalLetter = true;

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

hasNumber = true;

} else if (!Character.isLetterOrDigit(c)) {

hasSpecialCharacter = true;

}

}

return password.length() >= 8 && hasCapitalLetter && hasNumber && hasSpecialCharacter;

}

public String registerUser() {

if (checkUserName() && checkPasswordComplexity()) {

return "Username successfully captured\nPassword successfully captured";

} else {

StringBuilder errorMessage = new StringBuilder();

if (!checkUserName()) {

errorMessage.append("Username is not correctly formatted, please ensure that your username contains an underscore and is no more than 5 characters in length.\n");

}

if (!checkPasswordComplexity()) {

errorMessage.append("Password is not correctly formatted, please ensure that the password contains at least 8 characters, a capital letter, a number, and a special character.\n");

}

return errorMessage.toString();

}

}

public boolean loginUser(String inputUsername, String inputPassword) {

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

}

public String returnLoginStatus(boolean isLoggedIn) {

if (isLoggedIn) {

return "Welcome " + firstName + " " + lastName + ", it is great to see you again.";

} else {

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

}

}

public void displayTasksWithStatusDone() {

for (int i = 0; i < tasks.size(); i++) {

if (taskStatus[i].equalsIgnoreCase("done")) {

JOptionPane.showMessageDialog(null,"Developer: " + developers[i]

+ "Task Name: " + taskNames[i]

);

System.out.println("Task Name: " + taskNames[i]);

System.out.println("Task Duration: " + taskDuration[i]);

System.out.println();

}

}

}

public void displayTaskWithLongestDuration() {

int longestDuration = 0;

int longestDurationIndex = -1;

for (int i = 0; i < tasks.size(); i++) {

if (taskDuration[i] > longestDuration) {

longestDuration = taskDuration[i];

longestDurationIndex = i;

}

}

if (longestDurationIndex != -1) {

JOptionPane.showMessageDialog(null,"Developer: " + developers[longestDurationIndex]);

JOptionPane.showMessageDialog(null,"Task Duration: " + taskDuration[longestDurationIndex]);

}

}

public void searchTaskByName(String taskName) {

for (int i = 0; i < tasks.size(); i++) {

if (taskNames[i].equalsIgnoreCase(taskName)) {

JOptionPane.showMessageDialog(null, "Task Name: " + taskNames[i]

+ "Developer: " + developers[i]

+"Task Status: " + taskStatus[i]);

return;

}

}

System.out.println("Task not found.");

}

public void searchTasksByDeveloper(String developer) {

boolean foundTasks = false;

for (int i = 0; i < tasks.size(); i++) {

if (developers[i].equalsIgnoreCase(developer)) {

JOptionPane.showMessageDialog(null,"Task Name: " + taskNames[i] +"Task Status: " + taskStatus[i]) ;

foundTasks = true;

}

}

if (!foundTasks) {

System.out.println("No tasks found for the given developer.");

}

} return;

}

}