Automating Login Functionality Validation

ICT 3215 - Software Quality Management

Name: I.D.I.L Senavirathna
Registration Number: ICT/19/20/140
Index: 5073

1. Introduction

In the modern digital world, **login** is a feature that becomes an essential requirement for any web application, hence providing users with security and personalization of access. Automating its validation reduces manual effort and increases the efficiency and reliability of testing. This assignment aims to use **Selenium WebDriver for automation testing of login functionality on a sample website**. The python script automates the login results for distinct sets of credentials and gives definite outputs for successful or failed logins.

This document summarizes how to perform the actual implementation step by step and describes what is expected in terms of results and requirements for submission.
[GitHub: https://github.com/lakmina456/automation_testing]

2. Objective

The key deliverables of this exercise are to have an automation script that performs the following operations:

1. Navigates to a sample website login page. **(https://practicetestautomation.com/practice-test-login)**
2. Enter login credentials.
3. Send the login form.
4. Verify the login status based on expected outcomes.

3. Configuration Steps

Tools and Libraries

- Programming Language: Python 3.8 installed
- Automation Tool: Selenium WebDriver package (pip install selenium)
- Browser Driver: Chrome Driver
- IDE: VS Code

Environment Setup

1. Install Selenium:

```
>pip install selenium
```

2. Download the ChromeDriver

4. Implementation Steps

Step 1: Open a Browser and Navigate to the Login Page

Code:

```
from selenium import webdriver

driver = webdriver.Chrome()
driver.get("https://example.com/login")
```

Screenshot:

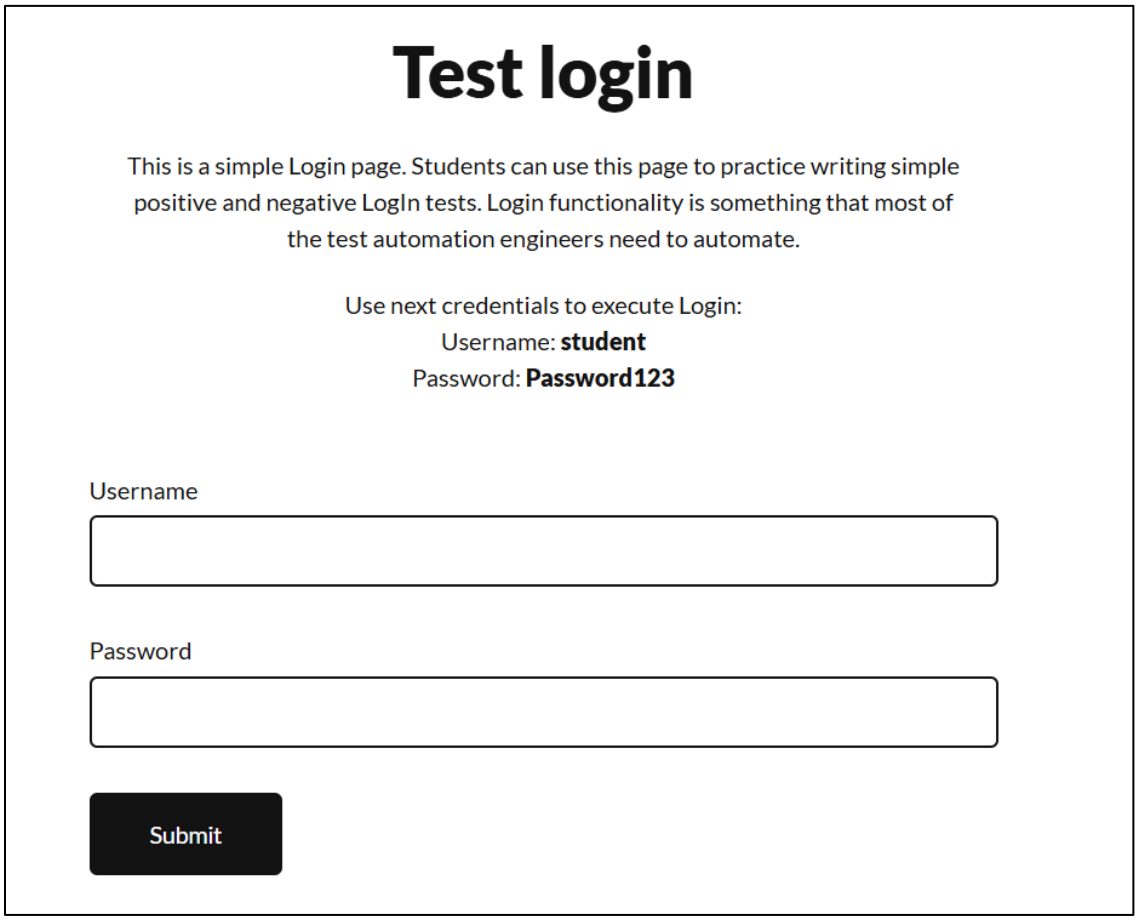

Figure 1: Landing page

Step 2: Locate Input Fields for Username and Password

- Identify the username and password input fields using locators (ID)

```
<div>
  <label for="username">Username</label>
  <input type="text" name="username" id="username" au
  f">
</div>
<div>
  <label for="password">Password</label>
  <input type="password" name="password" id="password
</div>
<button id="submit" class="btn">Submit</button>
/div>
```

Figure 2: Input fields IDs

- Input credentials:
    - Valid credentials: student/ Password123
    - Invalid credentials: user123 / password123

Code:

```
username_field = driver.find_element("id", "username")
password_field = driver.find_element("id", "password")

username_field.send_keys("user123")
password_field.send_keys("password123")
```

Screenshot:

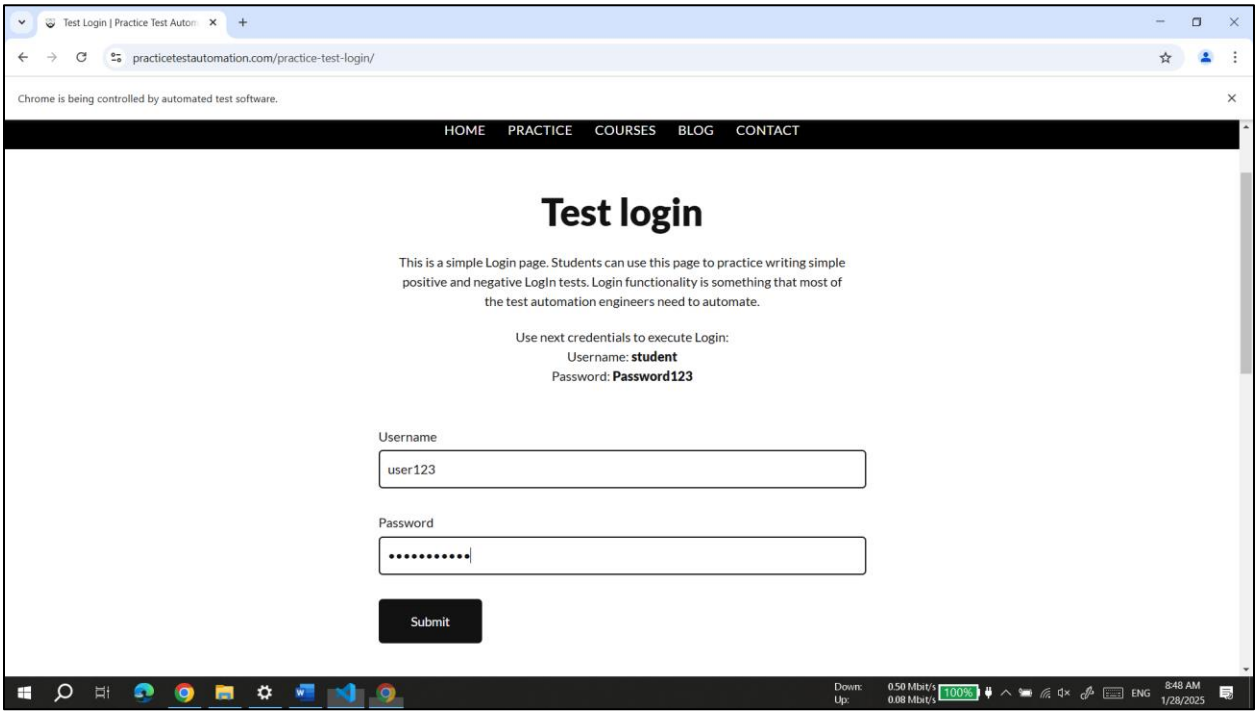

Figure 3: Browser In Auto Typing

Step 3: Click the Login Button

- Use Selenium to locate and click the login button.

Code:

```
login_button = driver.find_element("id", "submit")
login_button.click()
```

Step 4: Validate Login Status

- Check for successful login by verifying element “Welcome" message.
- Validate error messages for failed login attempts.

Code:

```
try:
    welcome_message = driver.find_element("xpath", "//div[@id='loop-
container']")
    print("Login Successful")
except:
    error_message = driver.find_element("xpath", "//div[@id='error']")
    print("Login Failed")
```

Success Case Screenshot:

- username: student and password: Password123

```
DevTools listening on ws://127.0.0.1:1182/devtools/browser/275437a8-b12b-4fee-81c3-d74d9829982e
Login Successful
```

Failure Case Screenshot:

- username: user123 and password: password123

```
DevTools listening on ws://127.0.0.1:12038/devtools/browser/43626c17-d8ae-43b4-8141-c8ba4cfbb7ee
Login Failed
```

5. Expected Output

✓ **Successful Login**:

- Message: "Login Successful"

Screenshot:

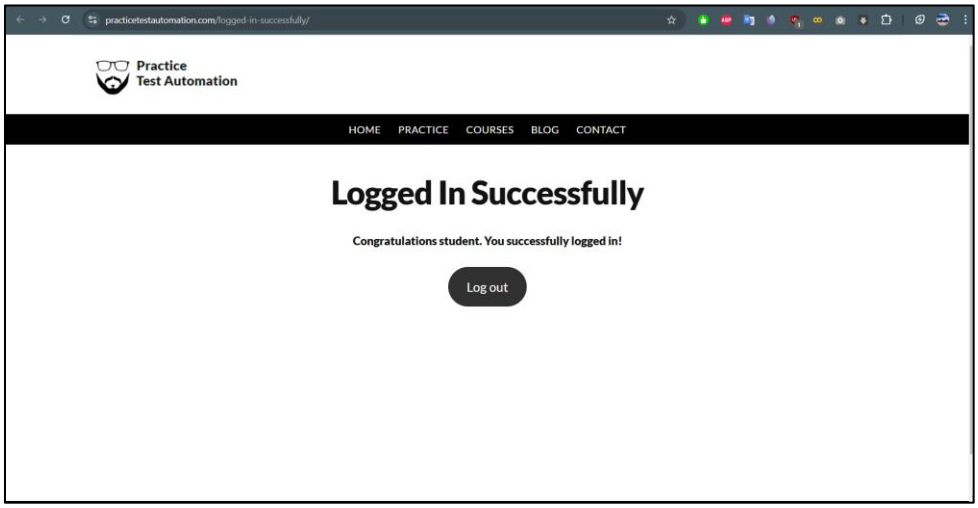

Figure 4: Login Success

✓ **Failed Login**:

- Message: "Login Failed"
  Screenshot:

Figure 5: Login Failed