University of Engineering and Technology

Automation Testing Report

Submitted To:

- Ms. Arooj Arshad

Submitted By:

- Marryam Abid

Institute of Business and Management
Software Quality Assurance

Report on Automation Testing of LinkedIn Features

Introduction

Automation testing plays a pivotal role in modern software development, ensuring the reliability and functionality of applications with efficiency and precision. This report details the automation testing process conducted on a LinkedIn page, focusing on the login functionality and sending a connect request.

Objective

The primary objectives of this project were:

1. To automate the LinkedIn login process.
2. To automate navigation to the "My Network" section.
3. To identify and click the "Connect" button on a connection request.

Tools and Technologies Used

1. **Programming Language:** Python
2. **Libraries/Modules:** Selenium, random, time
3. **IDE:** Visual Studio Code
4. **Browser:** Google Chrome
5. **Environment:** Windows Operating System **Project Setup**

1. **Folder Creation**
   - A folder named **automation** was created to organize all files and dependencies related to the project.
2. **Python Installation**
   - Python was installed using the Command Prompt (CMD).
   - To confirm successful installation, the command python --version was executed.
3. **Virtual Environment Setup**
   - A virtual environment was created using CMD to ensure isolated project dependencies.
4. **Files Created**
   - **practice.py**: The main script for automation testing.

- **func.py**: A helper script containing reusable functions for random delays and human-like typing.

Code Implementation

func.py

This file included the following helper functions:

```
import time
import random


def random_delay(start=2, end=15):
    time.sleep(random.uniform(start, end))


def human_typing(element, text):
for char in text:
        element.send_keys(char)
random_delay(0.1, 0.3)
```

- **random_delay**: Simulates human-like delay between actions.
- **human_typing**: Mimics human typing behavior when entering text into fields.

practice.py

The following steps were implemented in this script:

1. **Initialization**

```
from selenium import webdriver from func import * from
selenium.webdriver.chrome.service import Service from
selenium.webdriver.support.ui import WebDriverWait from
selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.by import By import time

chrome_driver_path = r"C:\Users\marry\Downloads\Automation\chromedriver-win64\chromedriver.exe"
service = Service(chrome_driver_path) driver = webdriver.Chrome(service=service) wait =
WebDriverWait(driver, 10)
```

- Imported necessary modules.
- Set up the ChromeDriver path and initialized the WebDriver.

2. **Navigating to LinkedIn Login Page**

```
driver.get("https://www.linkedin.com/login/")
print(driver.title) time.sleep(5)
```

- Opened the LinkedIn login page. ▪ Paused to allow the page to load.

3. **Automating Login**

```
username_field = wait.until(EC.visibility_of_element_located((By.XPATH, "//input[@id='username']")))
human_typing(username_field, 'marium.abid02@gmail.com')

password_field = wait.until(EC.visibility_of_element_located((By.XPATH, "//input[@id='password']")))
human_typing(password_field, '12*Jan2002')

sign_in_button = wait.until(EC.element_to_be_clickable((By.XPATH, "//button[@type='submit']")))
sign_in_button.click() print("Login successful") time.sleep(20)
```

- Located username and password fields.
- Entered login credentials using the human_typing function. ▪ Clicked the login button.

4. **Navigating to "My Network" and Clicking "Connect"**

```
my_network = wait.until(EC.element_to_be_clickable((By.XPATH, "//a[contains(@href, '/mynetwork/')]")))
my_network.click()
print("Navigated to My Network")

connect_buttons = wait.until(EC.presence_of_all_elements_located((By.XPATH,
"//span[text()='Connect']/ancestor::button")))
if connect_buttons:
connect_buttons[0].click()     print("Clicked on
the first 'Connect' button") else:
    print("No 'Connect' buttons found")

time.sleep(5) driver.quit()
```

- Navigated to the "My Network" section.
- Located and clicked the "Connect" button.
- Closed the browser after completing the actions.

Workflow Summary

1. Installed Python and necessary libraries.
2. Set up a virtual environment to manage dependencies.
3. Implemented helper functions for simulating human interactions.
4. Automated:

   - Login process on LinkedIn.
   - Navigation to "My Network" section.
   - Clicking the "Connect" button for a connection request.

Challenges and Observations

- **Challenges:**

    - Identifying dynamic elements with XPath required trial and error.
    - Ensuring synchronization between page load times and script execution.

- **Observations:**

    - The random delays and human typing simulation improved the script's mimicry of human behavior.

Conclusion

This project successfully automated the testing of LinkedIn's login functionality and the connect request process. The structured approach and use of helper functions ensured code reusability and reliability. This testing framework can be extended to cover other features of LinkedIn or similar web applications.