
**AMERICAN INTERNATIONAL UNIVERSITY–BANGLADESH (AIUB)**

**FACULTY OF SCIENCE AND TECHNOLOGY**

Course Name: SOFTWARE QUALITY AND TESTING

Course Code: CSC4271

Section: C

Semester: Spring 2024-25

Supervisor Name: ABHIJIT BHOWMIK

| SL | NAME | ID |
|---|---|---|
| 1 | MD. SHOHANUR RAHMAN SHOHAN | 22-46013-1 |
| 2 | FARJANA YESMIN OPI | 22-47018-1 |
| 3 | MD. ABU TOWSIF | 22-47019-1 |
| 4 | A. F. M. RAFIUL HASSAN | 22-47048-1 |

Submission date: May 19th, 2025

# Automated Testing using Selenium(Python)

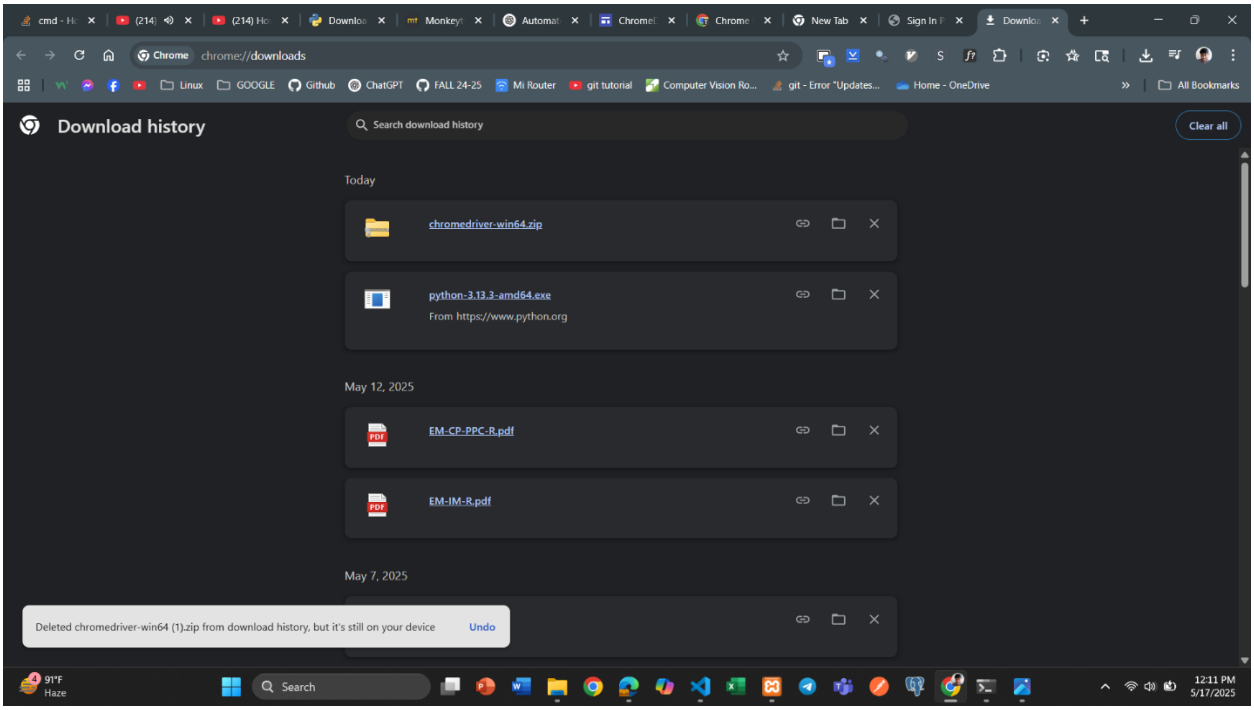


Figure 1: Downloading the correct version of Chrome Driver

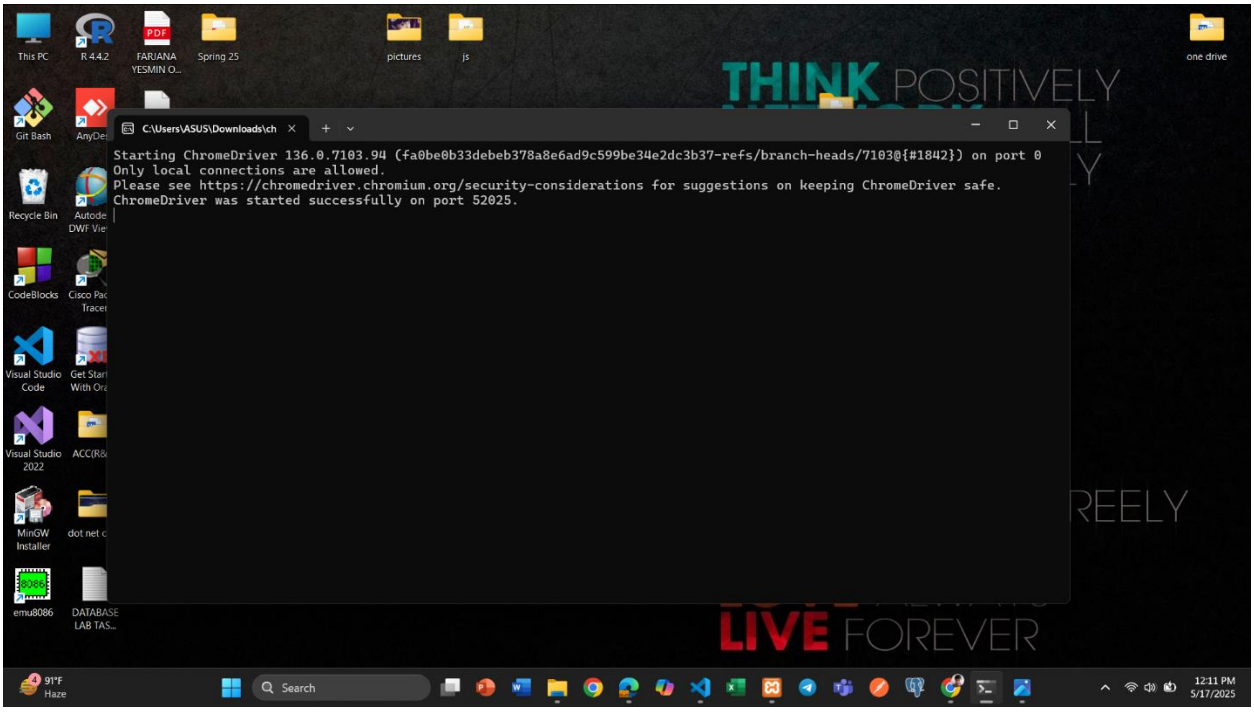


Figure 2: Chrome Driver installed and Chrome Driver started Selenium

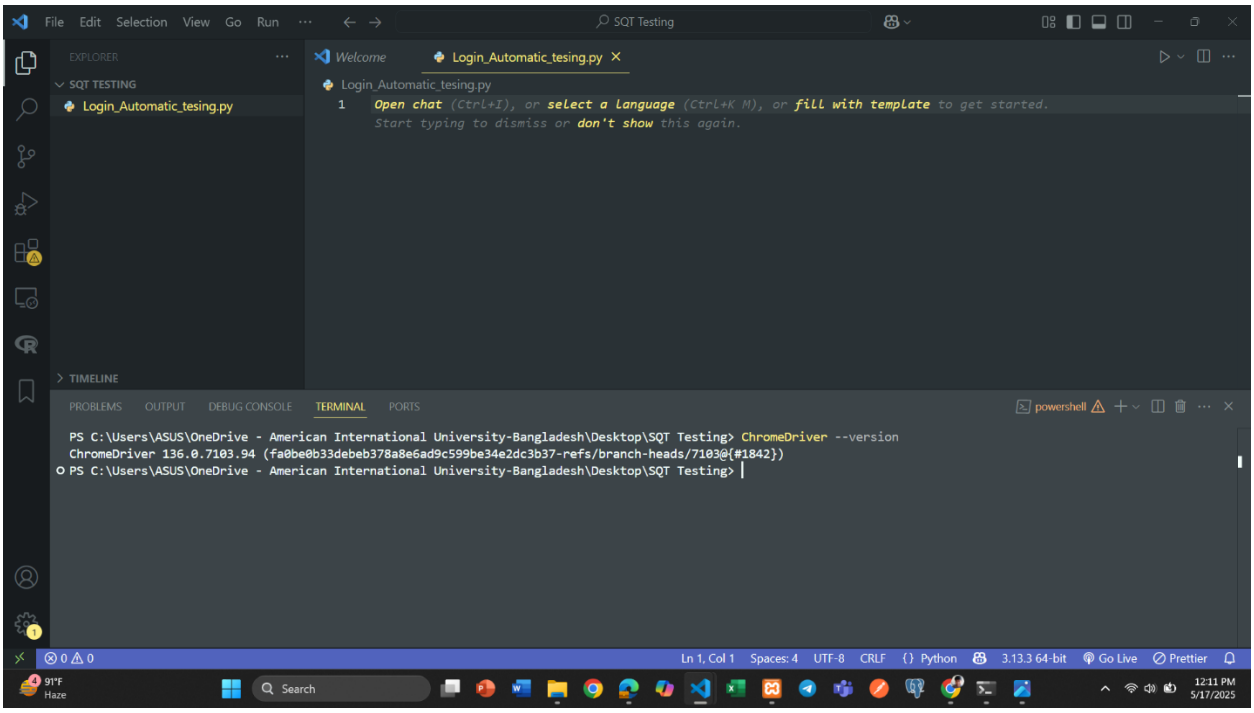


Figure 3: Checking the Chrome Driver in CLI

# Python Environment Setup

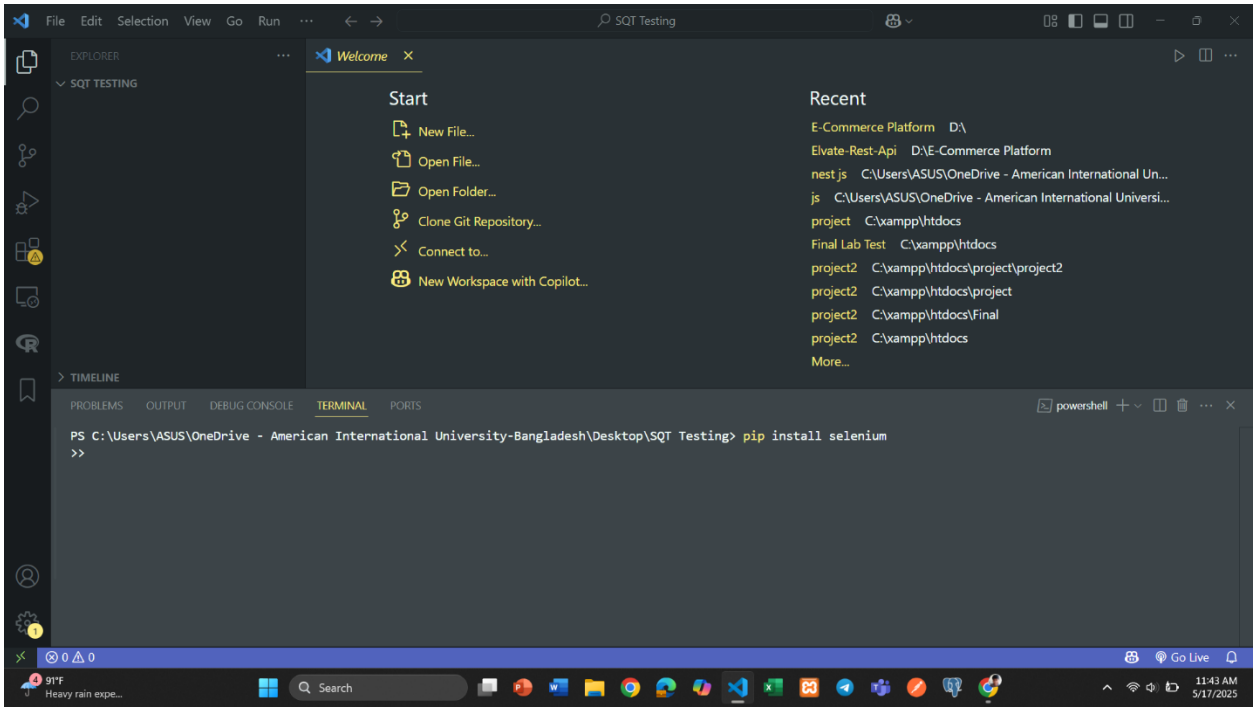


Figure 4: Installing required library in order to use Selenium

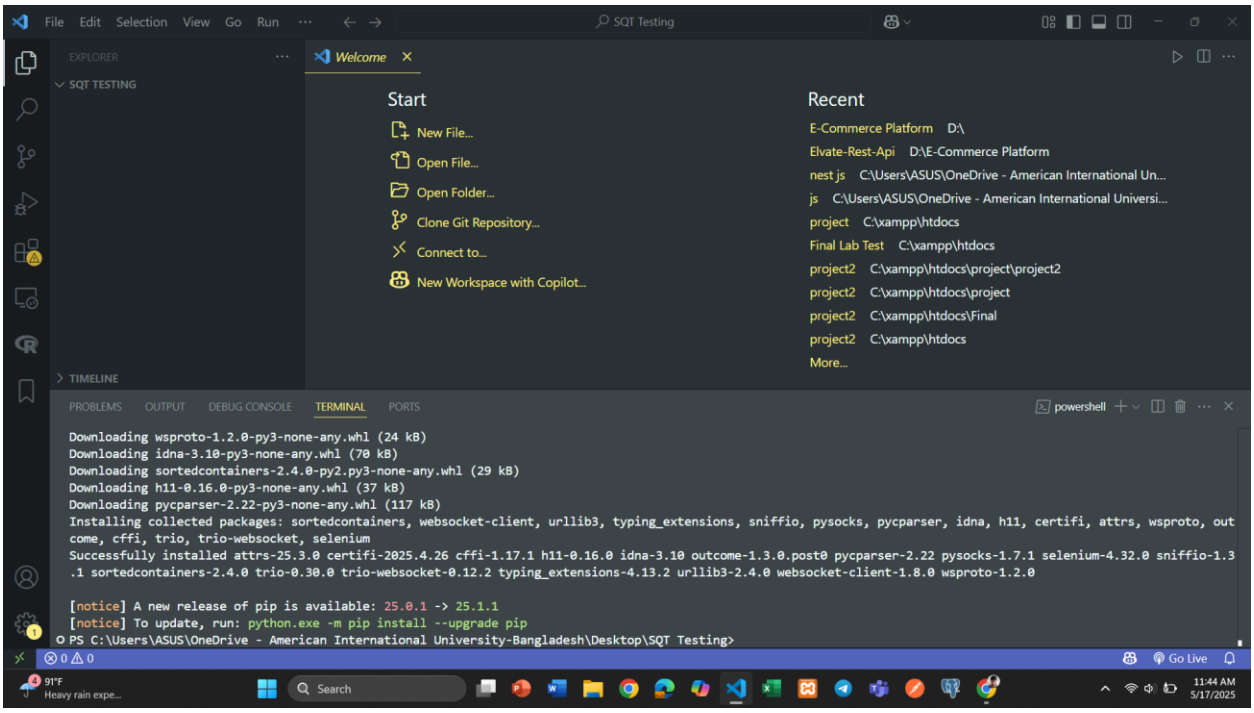


Figure 5: Required library has been installed in order to use Selenium

Test Case 1: Sign Up check

| | | | | |
|---|---|---|---|---|
| Project Name: Adventure | | | Test Designed by: A. F. M. RAFIUL HASSAN | |
| Test Case ID: FR_02 | | | Test Designed Date:5/17/2025 | |
| Test Priority (Low, Medium, High): Medium | | | Test Executed by: A. F. M. RAFIUL HASSAN | |
| Module Name: Signup Session | | | Test Execution Date:5/18/2025 | |
| Test Title: Verify signup with random valid credentials | | | | |
| Description: Test the website signup page with random valid data. | | | | |
| Precondition: User is on the signup page and database is ready to accept new users.<br>Dependencies: Internet connection, server running, form field IDs correctly set. | | | | |
| Test Steps | Test Data | Expected Results | Actual Results | Status (Pass/Fail) |
| 1. Open Chrome browser<br>2. Navigate to the signup page<br>3. Enter random valid test data<br>4. Select user type<br>5. Click the signup button | Name: testuser<br>Email: testXXXX@example.com (randomized)<br>Password: testpass123<br>User type: User (radio button) | User account should be created successfully. Page may redirect to sign in page. | As expected | Pass |
| Post Condition: User is registered successfully with randomized credentials. A new account is added to the database. The browser is closed properly. | | | | |

# Python code using Selenium Library

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time
import random

driver = webdriver.Chrome()

driver.get("http://localhost/project/view/opi_features/auth_feature/signup.html")
driver.maximize_window()

WebDriverWait(driver, 10).until(EC.presence_of_element_located((By.ID, "name")))

rand_num = random.randint(1000, 9999)
test_name = f"testuser"
test_email = f"test{rand_num}@example.com"
test_password = "testpass123"

driver.find_element(By.ID, "name").send_keys(test_name)
driver.find_element(By.ID, "email").send_keys(test_email)
driver.find_element(By.ID, "password").send_keys(test_password)
driver.find_element(By.ID, "confirm_password").send_keys(test_password)
driver.find_element(By.ID, "type1").click()

driver.save_screenshot("C:/Users/ASUS/Pictures/SS/signup_before_submit.png")
print('Screenshot saved: "signup_before_submit.png"')

submit_button = driver.find_element(By.XPATH, "//input[@type='submit' and
@name='signup']")
submit_button.click()

time.sleep(3)

driver.save_screenshot("C:/Users/ASUS/Pictures/SS/signup_after_submit.png")
print('Screenshot saved: "signup_after_submit.png"')

driver.quit()
```

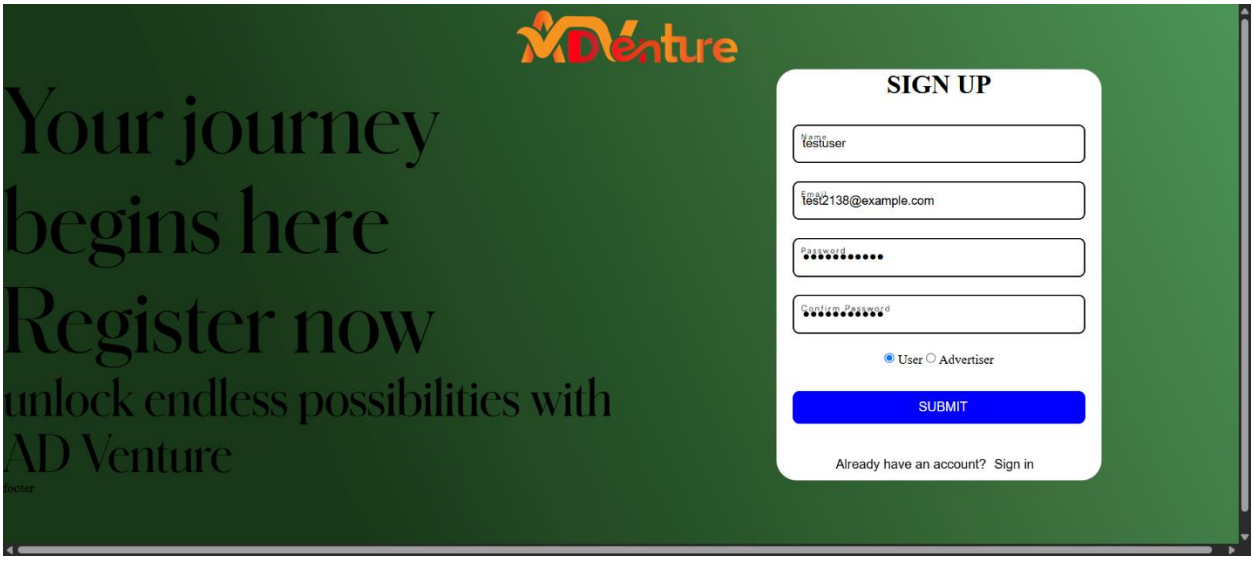


Figure 6: Sign Up before submit


Figure 7: Sign Up after submit

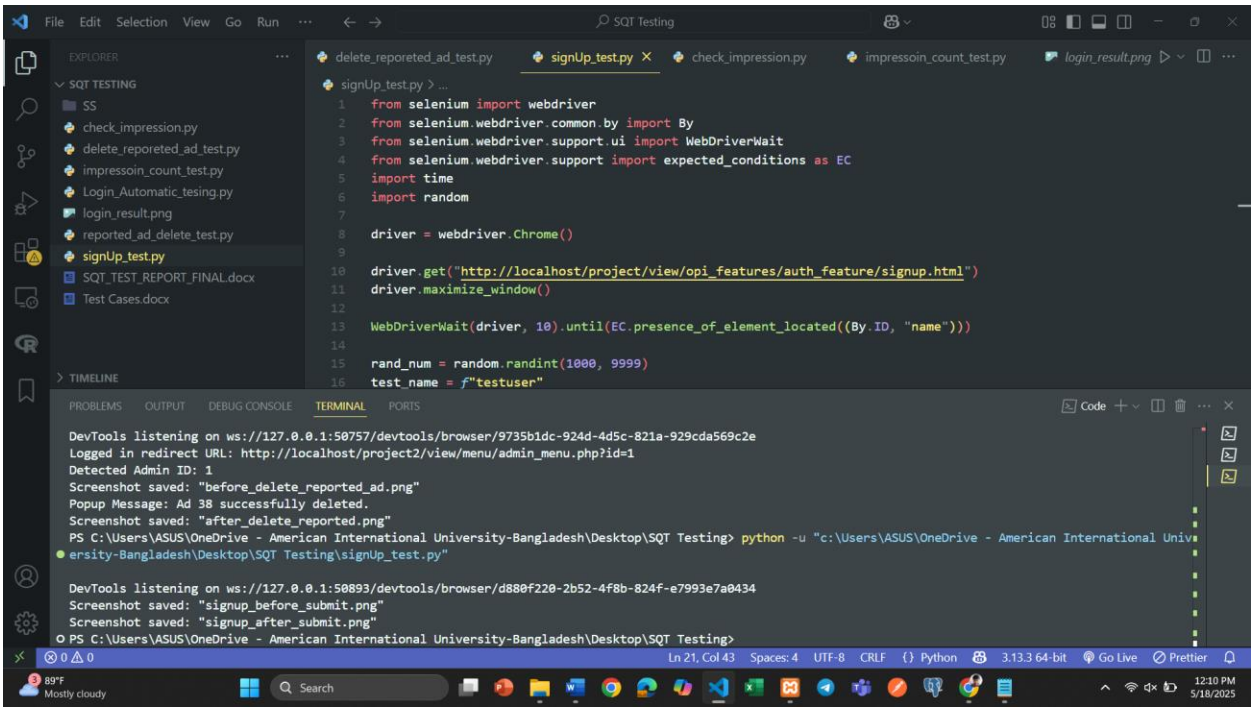


Figure 8: Terminal View

Test Case 2: login check

| | | | | |
|---|---|---|---|---|
| Project Name: Adventure | | | Test Designed by: MD. SHOHANUR RAHMAN SHOHAN | |
| Test Case ID: FR_01 | | | Test Designed Date: 5/17/2025 | |
| Test Priority (Low, Medium, High): Medium | | | Test Executed by: MD. SHOHANUR RAHMAN SHOHAN | |
| Module Name: login session | | | Test Execution Date: 5/18/2025 | |
| Test Title: verify login functionality with username and password using Selenium | | | | |
| Description: Test the website login page using Selenium automation script | | | | |
| Precondition: User has access to localhost login page with valid credentials<br>Dependencies: Chrome WebDriver installed and reachable via system path | | | | |
| Test Steps | Test Data | Expected Results | Actual Results | Status (Pass/Fail) |
| 1. Open Chrome browser<br>2. Navigate to login page<br>3. Enter valid username<br>4. Enter valid password<br>5. Click the submit button | Username: advertiser<br>Password: 123456 | User should be logged in successfully and redirected to the menu page. | As expected | Pass |
| Post Condition: User is validated and logged in successfully if credentials are correct. Browser session is closed automatically. | | | | |

# Python code using Selenium Library

```
from selenium import webdriver
from selenium.webdriver.common.by import By
import time

driver = webdriver.Chrome()

driver.get("http://localhost/project2/view/auth_feature/signin.html")
driver.maximize_window()
time.sleep(2)

username_input = driver.find_element(By.ID, "username")
password_input = driver.find_element(By.ID, "password")
submit_button = driver.find_element(By.XPATH, "//input[@type='submit' and
@name='login']")

username_input.send_keys("advertiser")
password_input.send_keys("123456")

submit_button.click()
time.sleep(3)

current_url = driver.current_url
print("Current URL after login attempt:", current_url)

if "menu" in current_url:
   screenshot_name = "login_successful.png"
   print("Login successful")
else:
   screenshot_name = "login_failed.png"
   print("Login failed")

screenshot_path = f"C:/Users/ASUS/Pictures/SS/{screenshot_name}"
driver.save_screenshot(screenshot_path)
print(f"Screenshot saved as: {screenshot_path}")

driver.quit()
```

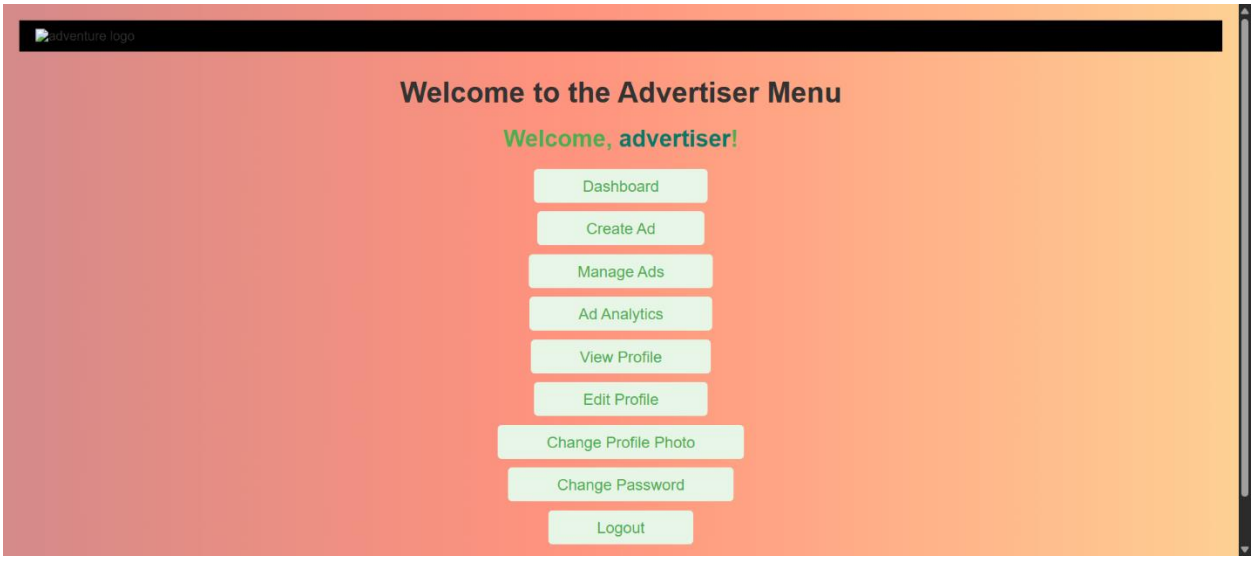


Figure 9: Login Successful

**Invalid Username and Password**

Figure 10: Login Failed

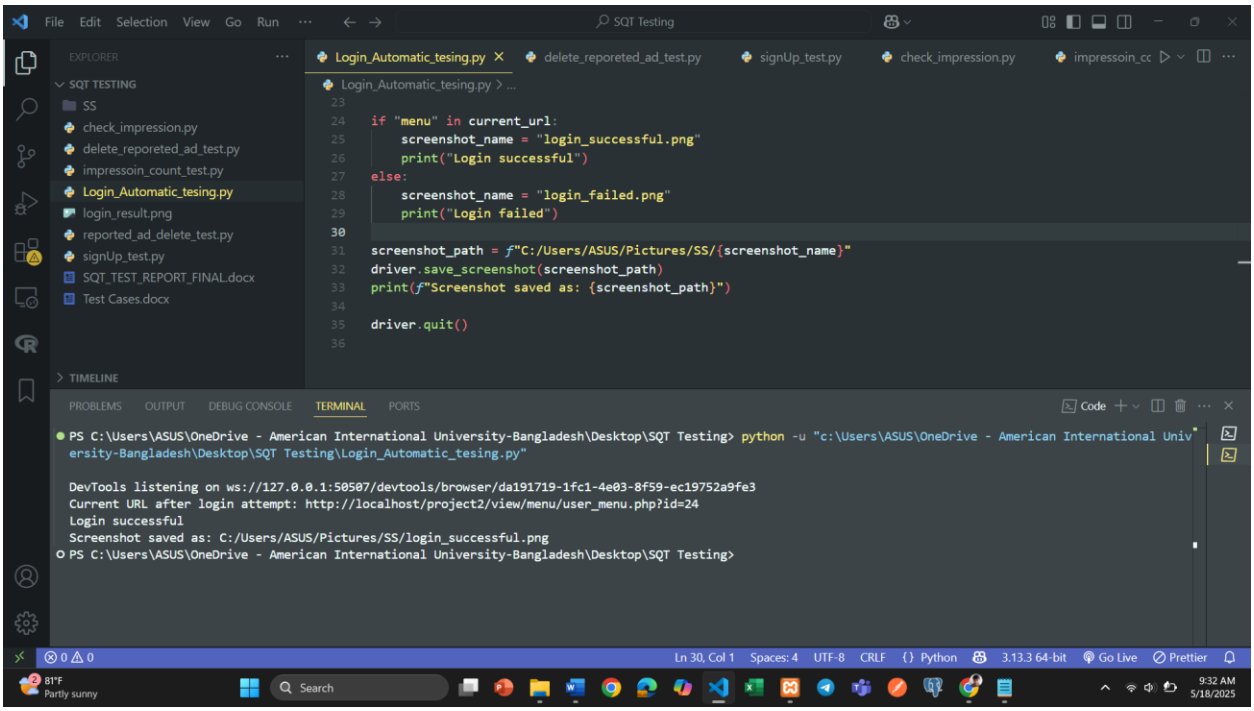


Figure 11: Terminal View For Login Successful

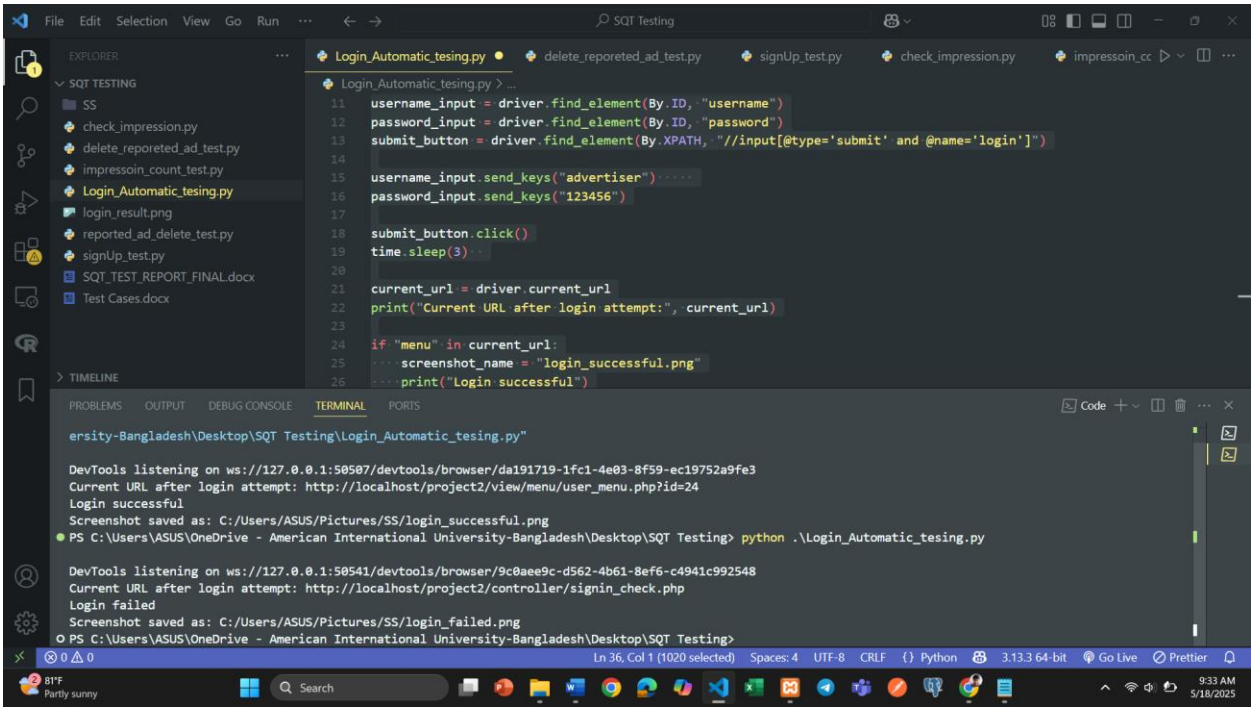


Figure 12: Terminal View For Login Failed

Test Case 3: Delete Reported Ad Functionality

| | | | | |
|---|---|---|---|---|
| Project Name: Adventure | | | Test Designed by: MD. ABU TOWSIF | |
| Test Case ID: FR_03 | | | Test Designed Date: 5/17/2025 | |
| Test Priority (Low, Medium, High): Medium | | | Test Executed by: MD. ABU TOWSIF | |
| Module Name: Reported Ads | | | Test Execution Date: 5/18/2025 | |
| Test Title: Verify deletion of a reported ad with confirmation popup using Selenium | | | | |
| Description: Test deleting a reported ad and confirm the success message via the confirmation popup. | | | | |
| Precondition: User must have access to the reported ads page with at least one reported ad present. Chrome WebDriver should be installed and accessible in the system path.<br>Dependencies: Chrome WebDriver installed and accessible via system path. | | | | |
| Test Steps | Test Data | Expected Results | Actual Results | Status (Pass/Fail) |
| 1.Open Sign in page<br>2. Sign in as Admin<br>2. Open browser and navigate to reported ads page<br>3. Wait for ads to load<br>4. Click first delete button<br>5. Confirm deletion<br>6. Verify success message<br>7. Close popup<br>8. Close browser | User Name: adminadmin<br>Password: 110918 | Page loads with ads. Deletion confirmed by success message. Browser closes. | As expected | Pass |
| Post Condition: The reported ad is deleted, and the browser session is closed. | | | | |

# Python code using Selenium Library

```
 from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time
import re

driver = webdriver.Chrome()

driver.get("http://localhost/project2/view/auth_feature/signin.html")
driver.maximize_window()

WebDriverWait(driver, 10).until(EC.presence_of_element_located((By.ID, "username")))
driver.find_element(By.ID, "username").send_keys("adminadmin")
driver.find_element(By.ID, "password").send_keys("110918")
driver.find_element(By.XPATH, "//input[@type='submit' and @name='login']").click()

WebDriverWait(driver, 10).until(EC.url_contains("admin_menu.php"))

admin_url = driver.current_url
print("Logged in redirect URL:", admin_url)

match = re.search(r'id=(\d+)', admin_url)
admin_id = match.group(1) if match else "1"
print("Detected Admin ID:", admin_id)

reported_ad_url =
f"http://localhost/project/view/tishat_features/report_ads/reported_ads.php?id={admin_id}
"
driver.get(reported_ad_url)

WebDriverWait(driver, 10).until(
   EC.presence_of_element_located((By.CLASS_NAME, "reported_ad_info_container"))
)
driver.save_screenshot("C:/Users/ASUS/Pictures/SS/before_delete_fullflow.png")
print('Screenshot saved: "before_delete_fullflow.png"')

delete_button = driver.find_element(By.XPATH, "(//div[@class='delete_btn']/button)[1]")
delete_button.click()

WebDriverWait(driver, 5).until(EC.visibility_of_element_located((By.ID, "confirmPopup")))
WebDriverWait(driver, 5).until(EC.element_to_be_clickable((By.ID, "confirmYes"))).click()
```

```
popup_message = WebDriverWait(driver, 10).until(EC.visibility_of_element_located((By.ID,
"popupMessage")))
print("Popup Message:", popup_message.text)

time.sleep(1)
driver.save_screenshot("C:/Users/ASUS/Pictures/SS/after_delete_fullflow.png")
print('Screenshot saved: "after_delete_fullflow.png"')

WebDriverWait(driver, 5).until(EC.element_to_be_clickable((By.ID, "confirmOk"))).click()

driver.quit()
```

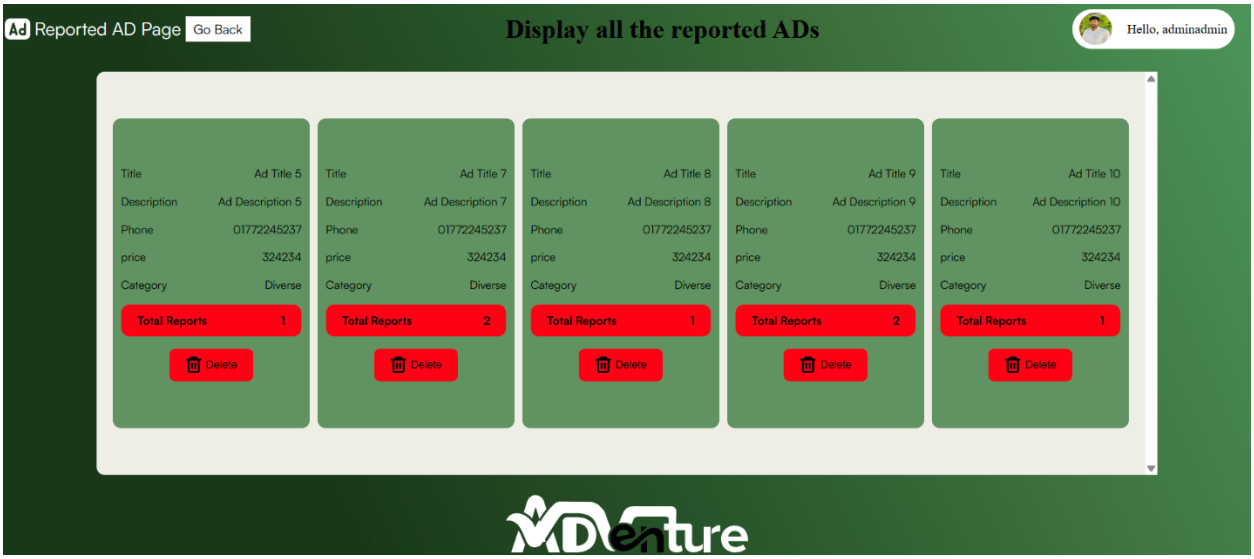


Figure 13: Before Delete reported ads

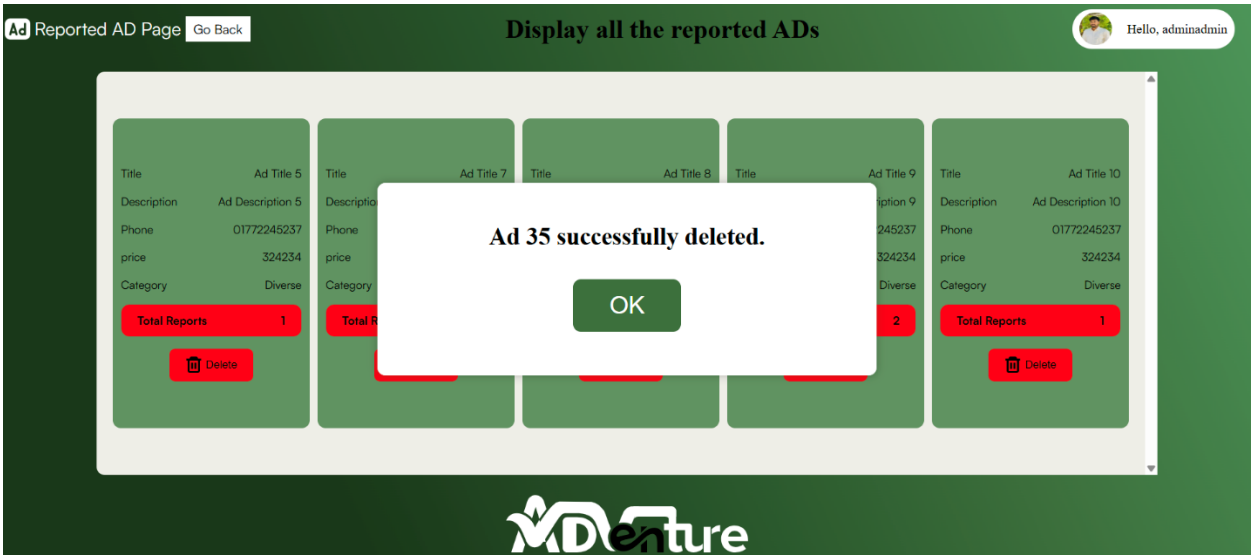


Figure 14: After Delete reported ads

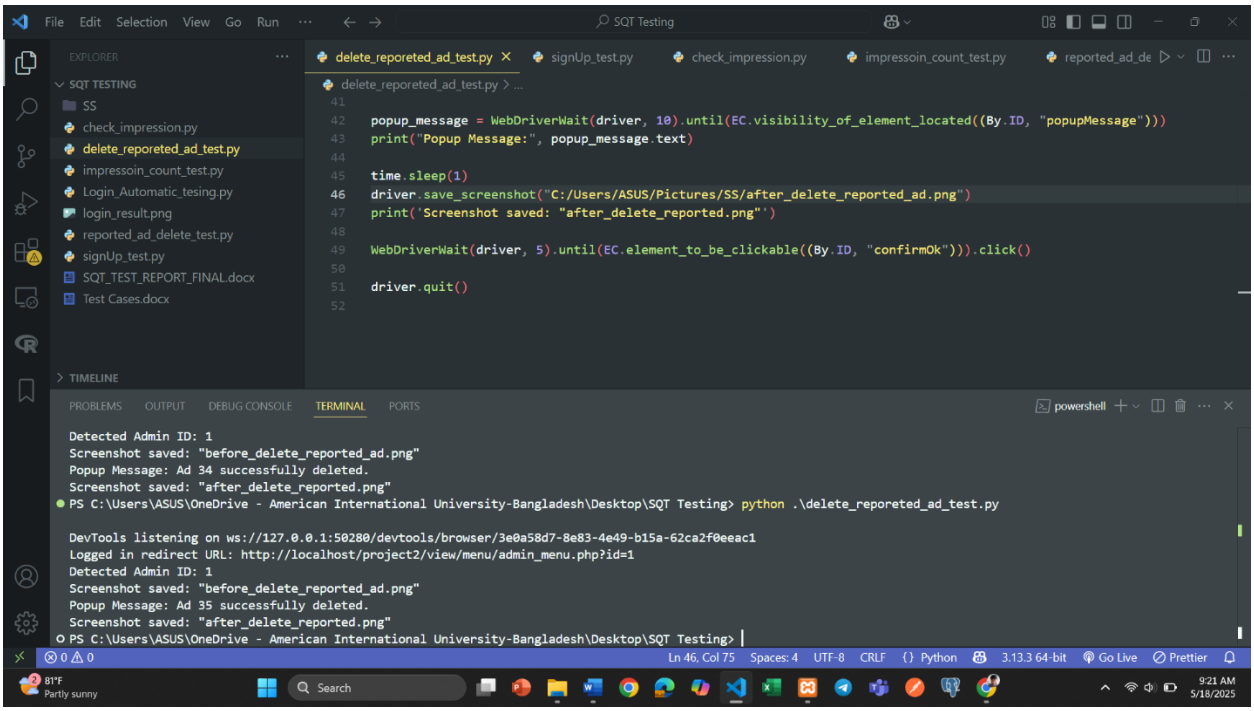


Figure 15: Terminal View for reported ad deletion test

Test Case 4: In landing page, impression button is working.

| | |
|---|---|
| Project Name: Adventure | Test Designed by: FARJANA YESMIN OPI |
| Test Case ID: FR_04 | Test Designed Date: 5/17/2025 |
| Test Priority (Low, Medium, High): Medium | Test Executed by: FARJANA YESMIN OPI |
| Module Name: Ad Impression Tracking | Test Execution Date: 5/18/2025 |
| Test Title: Validate Ad Impression Count Increment on Click | |
| Description: Ensure clicking the "Impression" button increases the ad impression count. | |

Precondition: Landing page is accessible; impression count is visible and functional.

Dependencies: Impression button must trigger backend update; internet/local server must be running.

| Test Steps | Test Data | Expected Results | Actual Results | Status (Pass/Fail) |
|---|---|---|---|---|
| 1. Open Chrome browser<br>2. Navigate to landing page<br>3. Capture current impression count<br>4. Click the impression button<br>5. Wait and verify count | URL: http://<br>localhost/project/ | Impression count should increase by 1 after clicking the button. | As expected | Pass |

Post Condition: Impression count is successfully incremented by one, verifying the click was registered and the browser session ends properly.

# Python code using Selenium Library

```
from selenium import webdriver
from selenium.webdriver.common.by import By
import time

driver = webdriver.Chrome()

try:
   driver.get('http://localhost/project/')
   time.sleep(7)


driver.save_screenshot("C:/Users/ASUS/Pictures/SS/before_impression_increase.png")
   print(f'Screenshot saved: "before_impression_increase.png"')

   impression_stat_div = driver.find_element(By.CLASS_NAME, 'ad_impression_class')
   impression_count_elem = impression_stat_div.find_element(By.TAG_NAME, 'p')
   count_before = int(impression_count_elem.text.strip())

   impression_button = driver.find_element(By.CLASS_NAME, 'impression')
   impression_button.click()

   time.sleep(3)

   driver.save_screenshot("C:/Users/ASUS/Pictures/SS/after_impression_increase.png")
   print(f'Screenshot saved: "after_impression_increase.png"')

   updated_stat_div = driver.find_element(By.CLASS_NAME, 'ad_impression_class')
   updated_count_elem = updated_stat_div.find_element(By.TAG_NAME, 'p')
   count_after = int(updated_count_elem.text.strip())

   print("Impression count before click:", count_before)
   print("Impression count after click: ", count_after)

   if count_after > count_before:
      print("Success: Impression count increased.")
   else:
      print("Failed: Impression count did not increase.")

finally:
   driver.quit()
```


Figure 16: Before Impression Increase


Figure 17: After Impression Increase

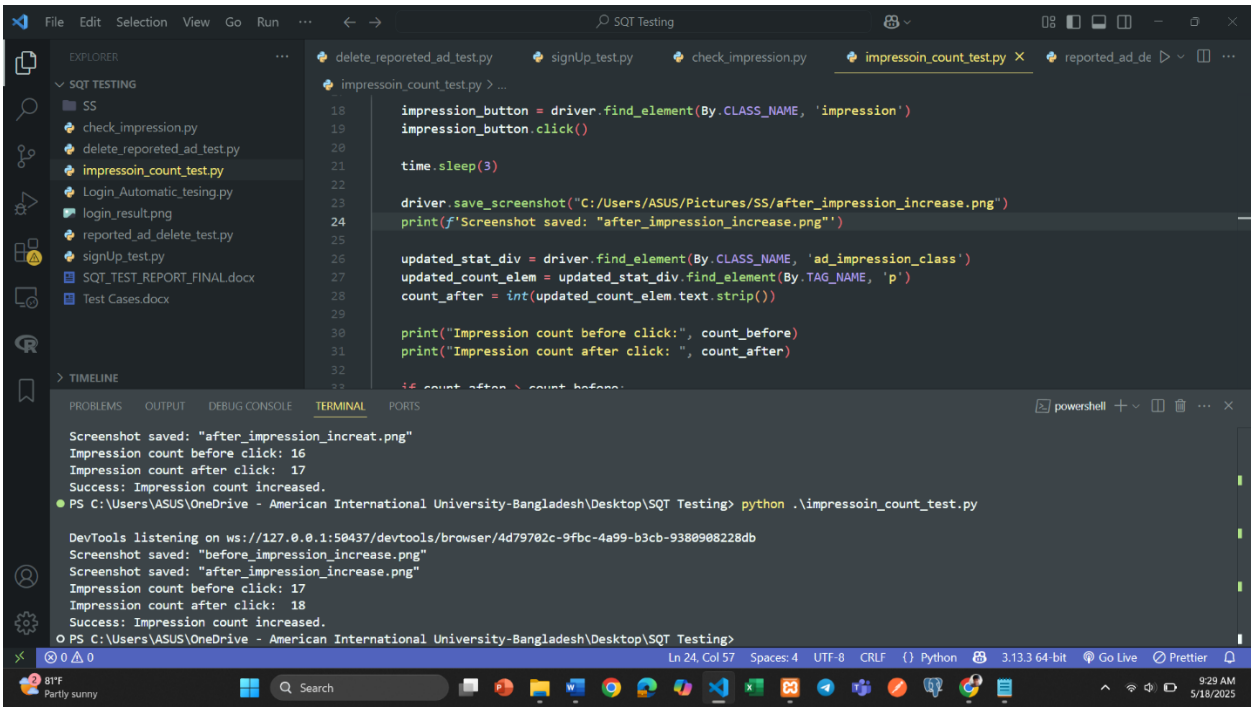


Figure 18: Terminal View for impression button test