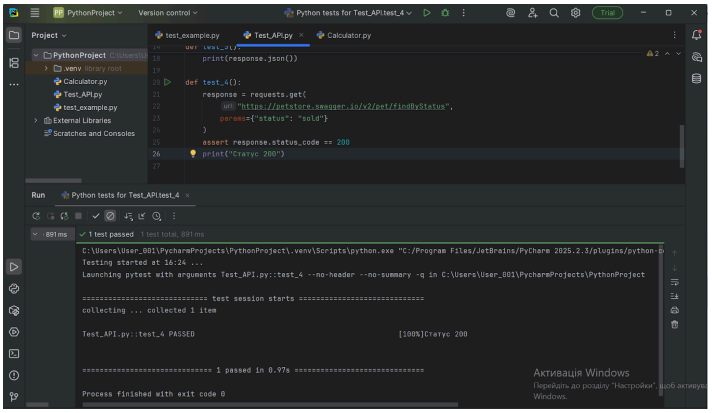
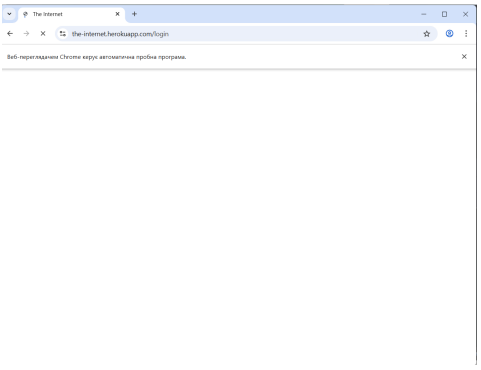
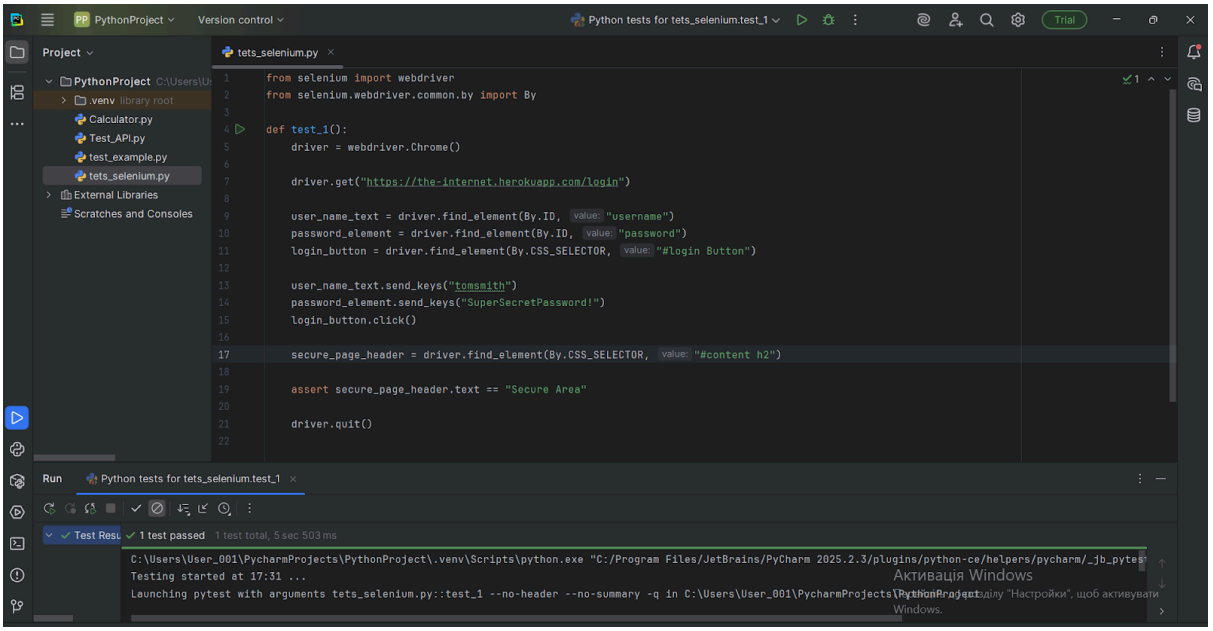
Автоматизація тестування веб-сторінок

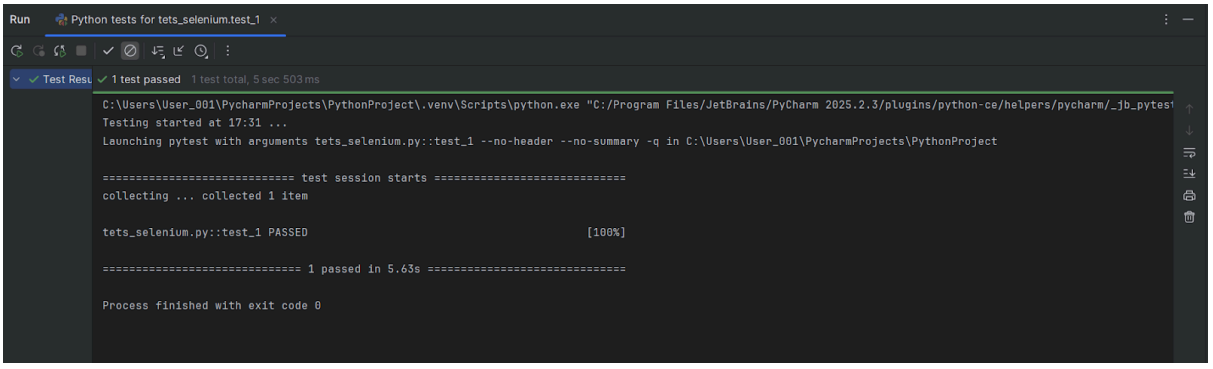
1. **Написання скрипта для відкриття сторінки:**
    - Використовуючи Selenium, реалізуємо скрипт, який відкриває сторінку: https://the-internet.herokuapp.com/login.

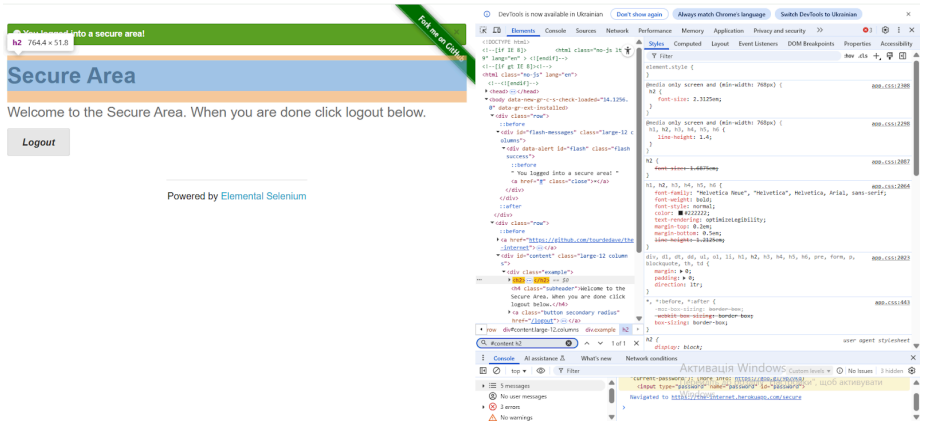




2. **Додавання перевірки заголовка сторінки:**
    - Перевірте, що заголовок на сторінці (#content h2) дорівнює тексту Login Page.







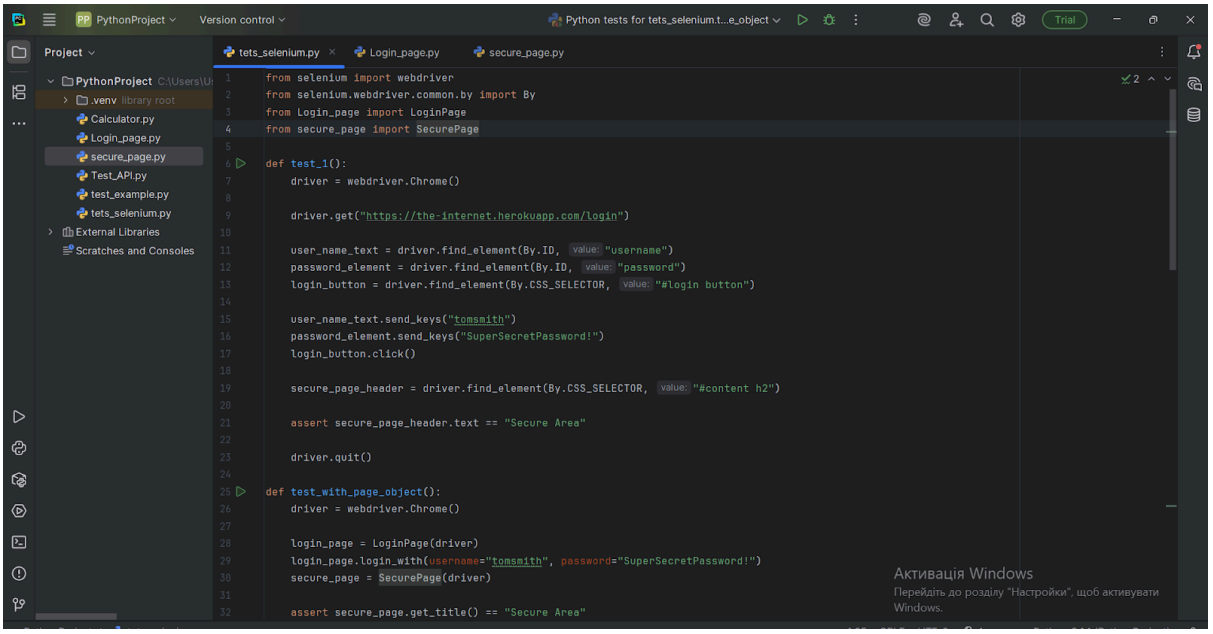
Page Object Model (POM)

test_selenium.py

Файл із тестами.

- test_1() – тест, де все в одному місці: відкриваємо браузер, вводимо логін/пароль, клікаємо кнопку й перевіряємо, що відкрилася сторінка з текстом "Secure Area".

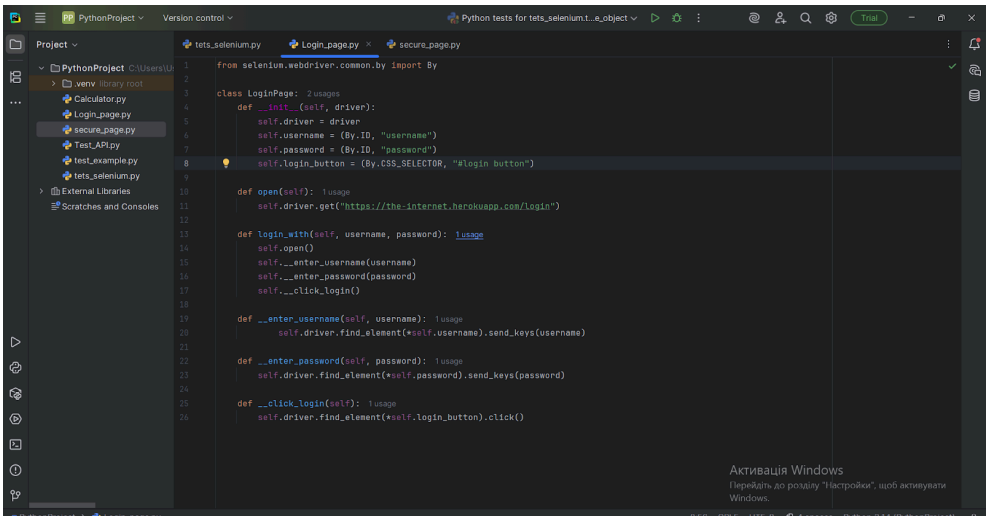
- test_with_page_object() – той самий тест, але вже з використанням **Page Object Model**. Тест виглядає чистіше: ми лише викликаємо методи з класів LoginPage і SecurePage.



login_page.py

Клас для роботи зі сторінкою логіну.

- В __init__ ми описали локатори (де шукати елементи: поле логіна, поле пароля, кнопку).
- Метод open() відкриває сторінку логіну.
- Метод login_with(username, password) виконує весь процес логіну (ввести логін, пароль, натиснути кнопку).
- Приватні методи __enter_username, __enter_password, __click_login інкапсулюють дії з кожним елементом.

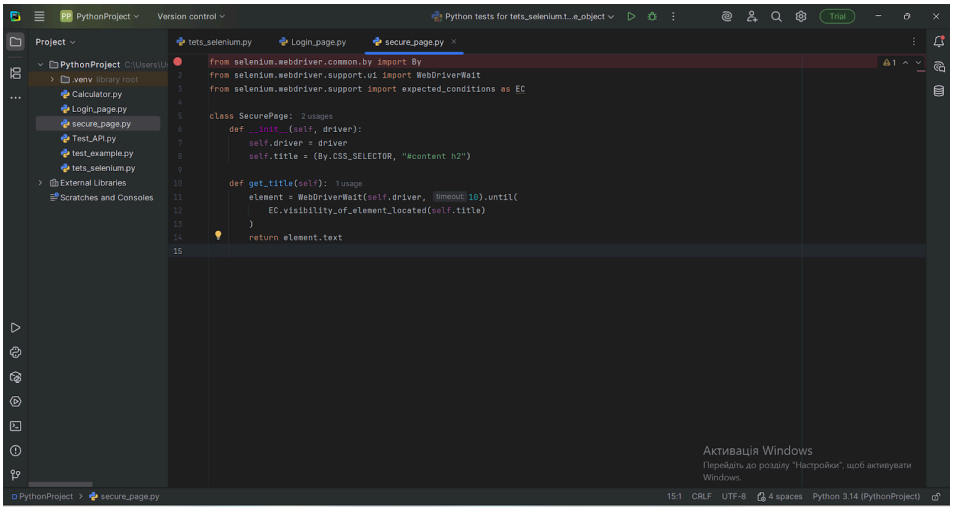


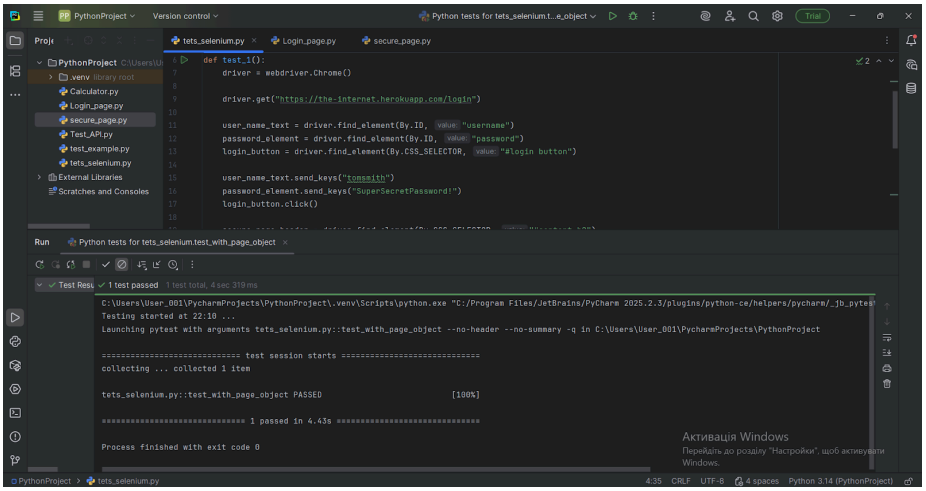
secure_page.py

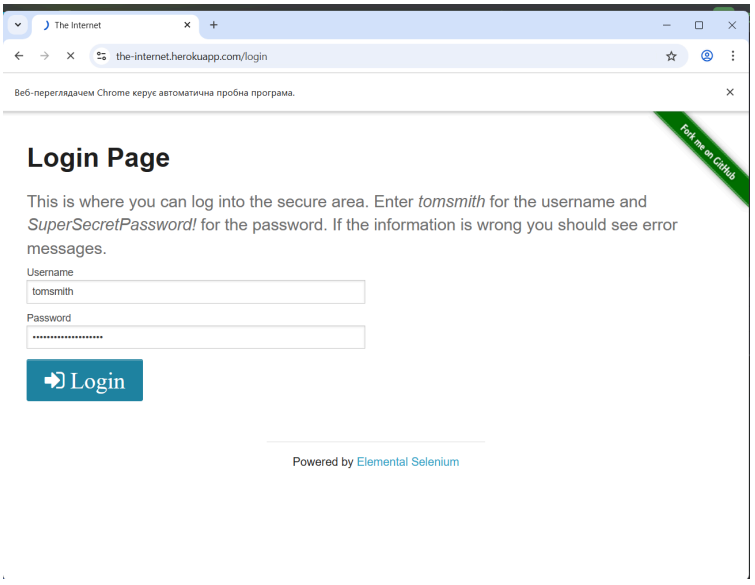
Клас для роботи зі сторінкою, куди потрапляє користувач після логіну.

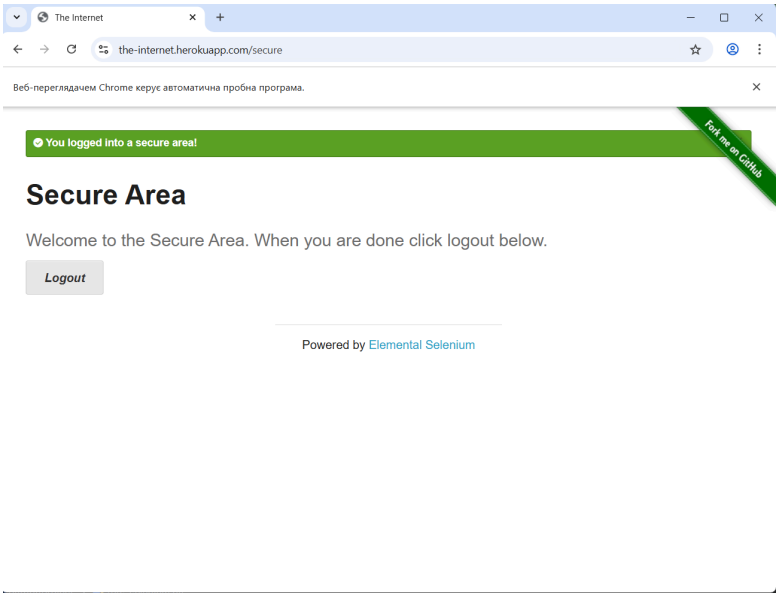
- Тут ми описали тільки один локатор – заголовок сторінки.
- Метод get_title() повертає текст цього заголовка.

Завдяки цьому в тесту ми можемо просто перевірити, що користувач реально потрапив у "Secure Area".




PythonProject
Version control
Python tests for tets_selenium.t...e_object
Trial
tets_selenium.py
Login_page.py
secure_page.py
PythonProject C:\Users\U
.venv library root
Calculator.py
Login_page.py
secure_page.py
Test_API.py
test_example.py
tets_selenium.py
External Libraries
Scratches and Consoles
def test_1():
driver = webdriver.Chrome()
driver.get("https://the-internet.herokuapp.com/login")
user_name_text = driver.find_element(By.ID, value: "username")
password_element = driver.find_element(By.ID, value: "password")
login_button = driver.find_element(By.CSS_SELECTOR, value: "#login button")
user_name_text.send_keys("tomsmith")
password_element.send_keys("SuperSecretPassword!")
login_button.click()
Run
Python tests for tets_selenium.test_with_page_object
Test Resu
1 test passed
1 test total, 4 sec 319 ms
C:\Users\User_001\PycharmProjects\PythonProject\.venv\Scripts\python.exe "C:/Program Files/JetBrains/PyCharm 2025.2.3/plugins/python-ce/helpers/pycharm/_jb_pytest
Testing started at 22:10 ...
Launching pytest with arguments tets_selenium.py::test_with_page_object --no-header --no-summary -q in C:\Users\User_001\PycharmProjects\PythonProject
============================ test session starts ============================
collecting ... collected 1 item
tets_selenium.py::test_with_page_object PASSED [100%]
============================= 1 passed in 4.43s =============================
Process finished with exit code 0
Активація Windows
Перейдіть до розділу "Настройки", щоб активувати Windows.
PythonProject
tets_selenium.py
4:35
CRLF
UTF-8
4 spaces
Python 3.14 (PythonProject)


The Internet
the-internet.herokuapp.com/login
Веб-переглядачем Chrome керує автоматична пробна програма.
Fork me on GitHub
Login Page
This is where you can log into the secure area. Enter tomsmith for the username and SuperSecretPassword! for the password. If the information is wrong you should see error messages.
Username
tomsmith
Password
Login
Powered by Elemental Selenium


The Internet
the-internet.herokuapp.com/secure
Веб-переглядачем Chrome керує автоматична пробна програма.
Fork me on GitHub
You logged into a secure area!
Secure Area
Welcome to the Secure Area. When you are done click logout below.
Logout
Powered by Elemental Selenium