# **Hướng Dẫn Triển Khai và Giải Thích Chi Tiết Automation Test cho Form Đăng Nhập Facebook bằng Selenium và Java**

## **Mục tiêu**

* Tự động hóa kiểm thử chức năng đăng nhập trên trang Facebook (https://www.facebook.com/) để:
  + Xác nhận đăng nhập thành công với thông tin hợp lệ (email/số điện thoại và mật khẩu đúng).
  + Xác nhận hiển thị thông báo lỗi khi đăng nhập với thông tin không hợp lệ.
* Sử dụng Selenium WebDriver, Java, TestNG, và ChromeDriver để thực hiện test.
* Quan sát chi tiết quá trình test qua log console và hành động trên trình duyệt Chrome.
* Giải thích từng dòng code để hiểu rõ cách hoạt động của automation test.

## **Các bước triển khai chi tiết**

### **Bước 1: Cài đặt môi trường**

1. **Cài đặt Java JDK**:
   * Truy cập https://www.oracle.com/java/technologies/javase-downloads.html hoặc <https://openjdk.java.net/>.
   * Tải và cài đặt Java JDK 17 (khuyến nghị để tương thích tốt với Selenium).
   * Cấu hình biến môi trường JAVA\_HOME:
     + Windows: Vào **Control Panel > System > Advanced System Settings > Environment Variables**.
     + Thêm biến JAVA\_HOME với giá trị là thư mục cài JDK (ví dụ: C:\Program Files\Java\jdk-11).
     + Thêm %JAVA\_HOME%\bin vào biến Path.
   * Kiểm tra cài đặt bằng lệnh trong Command Prompt:  
     java -version
   * **Cài đặt IntelliJ IDEA**:
   * Tải IntelliJ IDEA Community Edition từ https://www.jetbrains.com/idea/download/.
   * Cài đặt theo hướng dẫn mặc định.
   * Mở IntelliJ, vào **File > Project Structure > SDKs**, đảm bảo chọn J17.
2. **Cài đặt Maven**:
   * IntelliJ tích hợp sẵn Maven. Kiểm tra trong **File > Settings > Build, Execution, Deployment > Build Tools > Maven**.
   * Đảm bảo Maven Home Path trỏ đến phiên bản Maven đi kèm IntelliJ (thường là phiên bản mới nhất).
   * Nếu muốn cài Maven độc lập, tải từ https://maven.apache.org/download.cgi và cấu hình MAVEN\_HOME giống JAVA\_HOME.
3. **Tải và cài đặt ChromeDriver**:
   * Mở Google Chrome, vào **Settings > About Chrome**, xác nhận phiên bản là 137.0.7151.55.( vào phần 3. chỗ icon user -> dấu ? trợ giúp -> lấy phiên bản của anh )
   * Truy cập <https://googlechromelabs.github.io/chrome-for-testing/>.
   * Tìm phiên bản **137.0.7151.55 ( tìm đúng phiên bản của anh )**, tải file chromedriver-win64.zip từ:  
     <https://storage.googleapis.com/chrome-for-testing-public/137.0.7151.55/win64/chromedriver-win64.zip>
   * Giải nén file zip:
     + Mở bằng WinRAR hoặc Windows Explorer.
     + Vào thư mục chromedriver-win64, lấy file chromedriver.exe.( file đầu tiên ấy )
   * Lưu chromedriver.exe vào C:\drivers:
     + Tạo thư mục C:\drivers nếu chưa có (File Explorer > ổ C: > New Folder).
     + Sao chép chromedriver.exe vào C:\drivers.
   * Kiểm tra file tồn tại trong C:\drivers.

### **Bước 2: Tạo dự án Maven trong IntelliJ IDEA**

1. Mở IntelliJ IDEA, chọn **File > New > Project**.
2. Trong cửa sổ New Project:
   * Chọn **Maven** ở cột bên trái.
   * Đảm bảo **Project SDK** là JDK 11.( hoặc 17 hoăc 21 )
   * Nhấn **Next**.
3. Điền thông tin:
   * **GroupId**: com.example.
   * **ArtifactId**: selenium-login-test.
   * **Version**: 1.0-SNAPSHOT.
4. Nhấn **Next**, chọn thư mục lưu dự án (ví dụ: D:\Projects\selenium-login-test), nhấn **Finish**.

Cấu trúc thư mục dự án:  
selenium-login-test/

├── src/

│ ├── main/

│ ├── test/

│ └── java/

1. ├── pom.xml

### **Bước 3: Cấu hình file pom.xml**

1. Mở file pom.xml trong IntelliJ.

Thay nội dung bằng:

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>selenium-login-test</artifactId>

<version>1.0-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.seleniumhq.selenium</groupId>

<artifactId>selenium-java</artifactId>

<version>4.8.1</version>

</dependency>

<dependency>

<groupId>org.testng</groupId>

<artifactId>testng</artifactId>

<version>7.7.0</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.10.1</version>

<configuration>

<source>1.8</source>

<target>1.8</target>

</configuration>

</plugin>

</plugins>

</build>

1. </project>
2. Lưu file. IntelliJ sẽ tự động tải dependencies (Selenium và TestNG).
3. Nếu không tải, mở tab **Maven** (bên phải IntelliJ), nhấn **Refresh** hoặc chạy lệnh:  
   mvn clean install

### **Bước 4 : Viết code test**

1. Trong IntelliJ, nhấp chuột phải vào src/test/java/com/example/seleniumlogintest > **New > Java Class**.
2. Đặt tên class là SeleniumLoginTest, chọn **Kind: Class**.

Sao chép và dán code sau:  
package com.example.seleniumlogintest;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.testng.Assert;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

public class SeleniumLoginTest {

private WebDriver driver;

private String baseUrl = "https://www.facebook.com/";

@BeforeMethod

public void setUp() throws InterruptedException {

System.out.println("Bắt đầu thiết lập trình duyệt...");

System.setProperty("webdriver.chrome.driver", "C:\\drivers\\chromedriver.exe");

driver = new ChromeDriver();

driver.manage().window().maximize();

System.out.println("Trình duyệt Chrome đã mở và phóng to.");

Thread.sleep(1000);

}

@Test

public void testValidLogin() throws InterruptedException {

System.out.println("Bắt đầu test đăng nhập hợp lệ...");

System.out.println("Truy cập trang: " + baseUrl);

driver.get(baseUrl);

Thread.sleep(2000);

System.out.println("Tìm trường email/phone...");

WebElement emailField = driver.findElement(By.id("email"));

System.out.println("Tìm trường password...");

WebElement passwordField = driver.findElement(By.id("pass"));

System.out.println("Tìm nút login...");

WebElement loginButton = driver.findElement(By.name("login"));

Thread.sleep(1000);

System.out.println("Nhập email/phone: [your\_email\_or\_phone]");

emailField.sendKeys("your\_email\_or\_phone");

Thread.sleep(1000);

System.out.println("Nhập password: [your\_password]");

passwordField.sendKeys("your\_password");

Thread.sleep(1000);

System.out.println("Nhấn nút login...");

loginButton.click();

Thread.sleep(3000);

System.out.println("Kiểm tra trang chủ Facebook có hiển thị không...");

WebElement homePage = driver.findElement(By.cssSelector("div[role='feed']"));

Assert.assertTrue(homePage.isDisplayed(), "Login failed: Facebook home page not displayed");

System.out.println("Đăng nhập hợp lệ thành công! Trang chủ Facebook hiển thị.");

}

@Test

public void testInvalidLogin() throws InterruptedException {

System.out.println("Bắt đầu test đăng nhập không hợp lệ...");

System.out.println("Truy cập trang: " + baseUrl);

driver.get(baseUrl);

Thread.sleep(2000);

System.out.println("Tìm trường email/phone...");

WebElement emailField = driver.findElement(By.id("email"));

System.out.println("Tìm trường password...");

WebElement passwordField = driver.findElement(By.id("pass"));

System.out.println("Tìm nút login...");

WebElement loginButton = driver.findElement(By.name("login"));

Thread.sleep(1000);

System.out.println("Nhập email/phone: invalid\_user");

emailField.sendKeys("invalid\_user");

Thread.sleep(1000);

System.out.println("Nhập password: wrong\_password");

passwordField.sendKeys("wrong\_password");

Thread.sleep(1000);

System.out.println("Nhấn nút login...");

loginButton.click();

Thread.sleep(2000);

System.out.println("Kiểm tra thông báo lỗi có hiển thị không...");

WebElement errorMessage = driver.findElement(By.id("error\_box"));

Assert.assertTrue(errorMessage.isDisplayed(), "Error message not displayed for invalid login");

System.out.println("Test đăng nhập không hợp lệ thành công! Thông báo lỗi hiển thị.");

}

@AfterMethod

public void tearDown() {

System.out.println("Đóng trình duyệt...");

if (driver != null) {

driver.quit();

}

System.out.println("Trình duyệt đã đóng.");

}

1. }
2. Thay your\_email\_or\_phone và your\_password trong testValidLogin bằng email/phone và mật khẩu của tài khoản test từ bước 4.
3. Lưu file.

### **Bước 6: Giải thích chi tiết code**

Dưới đây là giải thích từng dòng code trong SeleniumLoginTest.java:

#### **Package và Import**

package com.example.seleniumlogintest;

* **Giải thích**: Định nghĩa package cho class, tổ chức code trong thư mục com/example/seleniumlogintest.

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.testng.Assert;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.Test;

* **Giải thích**:
  + By: Lớp của Selenium để tìm phần tử trên trang web (dùng ID, name, CSS selector, v.v.).
  + WebDriver: Giao diện chính của Selenium để điều khiển trình duyệt.
  + WebElement: Đại diện cho một phần tử HTML (như input, button).
  + ChromeDriver: Lớp cụ thể để điều khiển trình duyệt Chrome.
  + Assert: Lớp của TestNG để kiểm tra kết quả (pass/fail).
  + AfterMethod, BeforeMethod, Test: Annotation của TestNG để định nghĩa các phương thức setup, test, và teardown.

#### **Class và biến instance**

public class SeleniumLoginTest {

private WebDriver driver;

private String baseUrl = "https://www.facebook.com/";

* **Giải thích**:
  + SeleniumLoginTest: Tên class chứa các test case.
  + driver: Biến instance kiểu WebDriver, lưu trữ đối tượng điều khiển trình duyệt.
  + baseUrl: URL của trang đăng nhập Facebook, dùng chung cho các test case.

#### **Phương thức @BeforeMethod**

@BeforeMethod

public void setUp() throws InterruptedException {

System.out.println("Bắt đầu thiết lập trình duyệt...");

System.setProperty("webdriver.chrome.driver", "C:\\drivers\\chromedriver.exe");

driver = new ChromeDriver();

driver.manage().window().maximize();

System.out.println("Trình duyệt Chrome đã mở và phóng to.");

Thread.sleep(1000);

}

* **Giải thích**:
  + @BeforeMethod: Annotation của TestNG, chạy trước mỗi test case (testValidLogin, testInvalidLogin).
  + throws InterruptedException: Cần thiết vì dùng Thread.sleep.
  + System.out.println: In log để biết đang thiết lập trình duyệt.
  + System.setProperty: Chỉ định đường dẫn đến chromedriver.exe để Selenium biết dùng ChromeDriver.
  + driver = new ChromeDriver(): Khởi tạo trình duyệt Chrome.
  + driver.manage().window().maximize(): Phóng to cửa sổ trình duyệt để dễ quan sát.
  + Thread.sleep(1000): Tạm dừng 1 giây để thấy trình duyệt mở.

#### **Test case: testValidLogin**

@Test

public void testValidLogin() throws InterruptedException {

System.out.println("Bắt đầu test đăng nhập hợp lệ...");

System.out.println("Truy cập trang: " + baseUrl);

driver.get(baseUrl);

Thread.sleep(2000);

* **Giải thích**:
  + @Test: Đánh dấu phương thức này là một test case của TestNG.
  + System.out.println: In log để biết test bắt đầu và URL được truy cập.
  + driver.get(baseUrl): Mở trang https://www.facebook.com/ trong Chrome.
  + Thread.sleep(2000): Chờ 2 giây để thấy trang tải.

System.out.println("Tìm trường email/phone...");

WebElement emailField = driver.findElement(By.id("email"));

System.out.println("Tìm trường password...");

WebElement passwordField = driver.findElement(By.id("pass"));

System.out.println("Tìm nút login...");

WebElement loginButton = driver.findElement(By.name("login"));

Thread.sleep(1000);

* **Giải thích**:
  + driver.findElement(By.id("email")): Tìm trường nhập email/phone bằng ID email.
  + driver.findElement(By.id("pass")): Tìm trường nhập mật khẩu bằng ID pass.
  + driver.findElement(By.name("login")): Tìm nút đăng nhập bằng thuộc tính name="login".
  + WebElement: Lưu trữ các phần tử HTML để tương tác (nhập liệu, click).
  + Thread.sleep(1000): Chờ 1 giây để thấy các phần tử được tìm.

System.out.println("Nhập email/phone: [your\_email\_or\_phone]");

emailField.sendKeys("your\_email\_or\_phone");

Thread.sleep(1000);

System.out.println("Nhập password: [your\_password]");

passwordField.sendKeys("your\_password");

Thread.sleep(1000);

System.out.println("Nhấn nút login...");

loginButton.click();

Thread.sleep(3000);

* **Giải thích**:
  + emailField.sendKeys: Nhập email/phone test vào trường email.
  + passwordField.sendKeys: Nhập mật khẩu test vào trường mật khẩu.
  + loginButton.click(): Nhấn nút đăng nhập.
  + Thread.sleep(1000/3000): Chờ 1 giây sau mỗi hành động nhập liệu, 3 giây sau khi nhấn login để thấy kết quả.

System.out.println("Kiểm tra trang chủ Facebook có hiển thị không...");

WebElement homePage = driver.findElement(By.cssSelector("div[role='feed']"));

Assert.assertTrue(homePage.isDisplayed(), "Login failed: Facebook home page not displayed");

System.out.println("Đăng nhập hợp lệ thành công! Trang chủ Facebook hiển thị.");

}

* **Giải thích**:
  + driver.findElement(By.cssSelector("div[role='feed']")): Tìm phần tử news feed trên trang chủ Facebook.
  + Assert.assertTrue(homePage.isDisplayed(), ...): Kiểm tra phần tử news feed hiển thị, nếu không thì test fail với thông báo lỗi.
  + System.out.println: In log xác nhận test thành công.

#### **Test case: testInvalidLogin**

@Test

public void testInvalidLogin() throws InterruptedException {

System.out.println("Bắt đầu test đăng nhập không hợp lệ...");

System.out.println("Truy cập trang: " + baseUrl);

driver.get(baseUrl);

Thread.sleep(2000);

System.out.println("Tìm trường email/phone...");

WebElement emailField = driver.findElement(By.id("email"));

System.out.println("Tìm trường password...");

WebElement passwordField = driver.findElement(By.id("pass"));

System.out.println("Tìm nút login...");

WebElement loginButton = driver.findElement(By.name("login"));

Thread.sleep(1000);

System.out.println("Nhập email/phone: invalid\_user");

emailField.sendKeys("invalid\_user");

Thread.sleep(1000);

System.out.println("Nhập password: wrong\_password");

passwordField.sendKeys("wrong\_password");

Thread.sleep(1000);

System.out.println("Nhấn nút login...");

loginButton.click();

Thread.sleep(2000);

System.out.println("Kiểm tra thông báo lỗi có hiển thị không...");

WebElement errorMessage = driver.findElement(By.id("error\_box"));

Assert.assertTrue(errorMessage.isDisplayed(), "Error message not displayed for invalid login");

System.out.println("Test đăng nhập không hợp lệ thành công! Thông báo lỗi hiển thị.");

}

* **Giải thích**:
  + Tương tự testValidLogin, nhưng nhập thông tin sai (invalid\_user/wrong\_password).
  + Kiểm tra thông báo lỗi bằng By.id("error\_box").
  + Assert.assertTrue: Xác nhận thông báo lỗi hiển thị, nếu không thì test fail.

#### **Phương thức @AfterMethod**

@AfterMethod

public void tearDown() {

System.out.println("Đóng trình duyệt...");

if (driver != null) {

driver.quit();

}

System.out.println("Trình duyệt đã đóng.");

}

* **Giải thích**:
  + @AfterMethod: Chạy sau mỗi test case để dọn dẹp.
  + driver.quit(): Đóng trình duyệt và kết thúc phiên WebDriver.
  + if (driver != null): Kiểm tra để tránh lỗi nếu driver chưa khởi tạo.
  + System.out.println: In log để biết trình duyệt đã đóng.

### **Bước 7: Chạy test**

1. Trong IntelliJ, nhấp chuột phải vào SeleniumLoginTest.java > **Run 'SeleniumLoginTest'**.
2. Hoặc chạy riêng từng test:
   * Nhấp chuột phải vào testValidLogin > **Run 'testValidLogin()'**.
   * Nhấp chuột phải vào testInvalidLogin > **Run 'testInvalidLogin()'**.
3. Hoặc dùng terminal:  
   mvn test
4. Quan sát:
   * **Trình duyệt Chrome**:
     + Mở, phóng to, truy cập https://www.facebook.com/.
     + testValidLogin: Nhập email/phone và password đúng, nhấn login, hiển thị trang chủ (news feed).
     + testInvalidLogin: Nhập thông tin sai, nhấn login, hiển thị thông báo lỗi.
     + Trình duyệt đóng sau mỗi test.
   * **Console IntelliJ**:
     + Log chi tiết từng bước (truy cập, tìm phần tử, nhập liệu, nhấn nút, kiểm tra).

Ví dụ:  
Bắt đầu thiết lập trình duyệt...

Trình duyệt Chrome đã mở và phóng to.

Bắt đầu test đăng nhập hợp lệ...

Truy cập trang: https://www.facebook.com/

Tìm trường email/phone...

Tìm trường password...

Tìm nút login...

Nhập email/phone: [your\_email\_or\_phone]

Nhập password: [your\_password]

Nhấn nút login...

Kiểm tra trang chủ Facebook có hiển thị không...

Đăng nhập hợp lệ thành công! Trang chủ Facebook hiển thị.

Đóng trình duyệt...

* + - Trình duyệt đã đóng.
  + **TestNG**: Báo cáo "Tests passed: 2" trong tab **Run** nếu cả hai test pass.

### **Bước 8: Kiểm tra và xử lý lỗi**

1. **Kết quả mong đợi**:
   * testValidLogin: Trang chủ Facebook hiển thị (div[role='feed']).
   * testInvalidLogin: Thông báo lỗi hiển thị (div[id='error\_box']).
2. **Xử lý lỗi**:
   * **Lỗi "chromedriver.exe not found"**:
     + Kiểm tra file C:\drivers\chromedriver.exe tồn tại.
     + Tải lại từ https://storage.googleapis.com/chrome-for-testing-public/137.0.7151.55/win64/chromedriver-win64.zip.
   * **Lỗi "Session not created"**:
     + Xác nhận Chrome và ChromeDriver đều là 137.0.7151.55 (Chrome > Settings > About Chrome).
   * **Lỗi locator không tìm thấy**:
     + Mở https://www.facebook.com/, nhấp chuột phải > **Inspect**.
     + Kiểm tra ID (email, pass), name (login), hoặc CSS selector (div[role='feed'], div[id='error\_box']).
     + Cập nhật locator trong code nếu HTML thay đổi.
   * **Lỗi CAPTCHA/2FA**:
     + Dùng tài khoản test từ Facebook Developer.
     + Tắt 2FA trên tài khoản thật (Settings > Security and Login).

### **Lưu ý**

* **CAPTCHA/2FA**: Tài khoản test giúp tránh CAPTCHA. Nếu dùng tài khoản thật, tắt 2FA.
* **HTML thay đổi**: Facebook có thể cập nhật giao diện, kiểm tra locator bằng **Inspect Element**.

**WebDriverManager**: Để tránh tải ChromeDriver thủ công, thêm dependency:  
<dependency>

<groupId>io.github.bonigarcia</groupId>

<artifactId>webdrivermanager</artifactId>

<version>5.9.2</version>

<scope>test</scope>

* </dependency>  
  Sửa @BeforeMethod:  
  WebDriverManager.chromedriver().setup();

### **Kết quả đạt được**

* Hiểu cách tự động hóa kiểm thử form đăng nhập thực tế (Facebook).
* Nắm rõ từng dòng code và cách Selenium tương tác với trình duyệt.
* Biết cách thiết lập dự án, chạy test, và xử lý lỗi.
* Nền tảng để mở rộng test với các kịch bản phức tạp hơn.

phần chạy thì anh chạy aplocation trc sau rồi vào phần test để chạy test case