**INTERNSHIP**

PRODIGY\_ST\_03\_Automated Login Test for a Web Application

**CODE:**

import unittest

from selenium import webdriver

from selenium.webdriver.common.by import By

import time

class TestLogin(unittest.TestCase):

def setUp(self):

"""

This method is called before each test.

It initializes the WebDriver and navigates to the login page.

"""

self.driver = webdriver.Chrome()

self.driver.get("https://www.saucedemo.com/")

self.driver.maximize\_window()

time.sleep(2) # Allow time for the page to load

def test\_positive\_login(self):

"""

Test Case 01: Positive Login

This test verifies that a user with valid credentials can successfully log in.

"""

# Find the username and password fields and enter the credentials

self.driver.find\_element(By.ID, "user-name").send\_keys("standard\_user")

self.driver.find\_element(By.ID, "password").send\_keys("secret\_sauce")

self.driver.find\_element(By.ID, "login-button").click()

time.sleep(2)

# Verify that the login was successful by checking the URL

expected\_url = "https://www.saucedemo.com/inventory.html"

self.assertEqual(self.driver.current\_url, expected\_url, "Login was not successful.")

print("Positive Login Test: Passed")

def test\_negative\_invalid\_password(self):

"""

Test Case 02: Negative Login - Invalid Password

This test verifies that the system prevents login with a valid username but an invalid password.

"""

# Find the username and password fields and enter the credentials

self.driver.find\_element(By.ID, "user-name").send\_keys("standard\_user")

self.driver.find\_element(By.ID, "password").send\_keys("wrong\_password")

self.driver.find\_element(By.ID, "login-button").click()

time.sleep(2)

# Verify that an error message is displayed

error\_message = self.driver.find\_element(By.XPATH, "//h3[@data-test='error']").text

expected\_error = "Epic sadface: Username and password do not match any user in this service"

self.assertIn(expected\_error, error\_message, "Error message for invalid password was not displayed correctly.")

print("Negative Login Test (Invalid Password): Passed")

def test\_negative\_locked\_out\_user(self):

"""

Test Case 03: Negative Login - Locked Out User

This test verifies that a user who is locked out cannot log in.

"""

# Find the username and password fields and enter the credentials

self.driver.find\_element(By.ID, "user-name").send\_keys("locked\_out\_user")

self.driver.find\_element(By.ID, "password").send\_keys("secret\_sauce")

self.driver.find\_element(By.ID, "login-button").click()

time.sleep(2)

# Verify that an error message for a locked-out user is displayed

error\_message = self.driver.find\_element(By.XPATH, "//h3[@data-test='error']").text

expected\_error = "Epic sadface: Sorry, this user has been locked out."

self.assertIn(expected\_error, error\_message, "Error message for locked out user was not displayed correctly.")

print("Negative Login Test (Locked Out User): Passed")

def test\_negative\_empty\_credentials(self):

"""

Test Case 04: Negative Login - Empty Credentials

This test verifies that the system prompts for a username if the fields are left empty.

"""

# Click the login button without entering any credentials

self.driver.find\_element(By.ID, "login-button").click()

time.sleep(2)

# Verify that an error message for a required username is displayed

error\_message = self.driver.find\_element(By.XPATH, "//h3[@data-test='error']").text

expected\_error = "Epic sadface: Username is required"

self.assertIn(expected\_error, error\_message, "Error message for empty credentials was not displayed correctly.")

print("Negative Login Test (Empty Credentials): Passed")

def tearDown(self):

"""

This method is called after each test.

It closes the browser window.

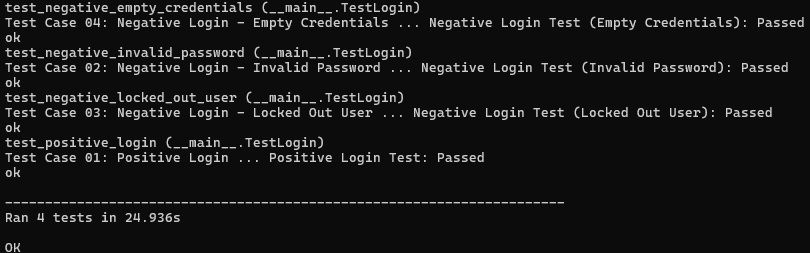
"""

self.driver.quit()

if \_\_name\_\_ == "\_\_main\_\_":

unittest.main()

**OUTPUT:**



Aritra Datta