URL for task 1 and task 2:

<https://github.com/talhashahab786/NextBasketSecondTest.git>

**TASK 1 Part A**

from selenium import webdriver

from selenium.webdriver.common.by import By

from selenium.webdriver.common.keys import Keys

from selenium.webdriver.support.ui import WebDriverWait

from selenium.webdriver.support import expected\_conditions as EC

import unittest

import time

from selenium import webdriver

from selenium.webdriver.support.ui import Select

from selenium.webdriver.common.keys import Keys

from selenium.webdriver.common.keys import Keys

from selenium.webdriver.common.by import By

class TestVerifySale50Percent(unittest.TestCase):

def test\_ordeNonPromoItemCheckout(self):

driver = webdriver.Chrome()

try:

driver.get("https://teststoreforsouthafri.nextbasket.shop/")

driver.maximize\_window()

time.sleep(3)

cookiePopup = WebDriverWait(driver, 10).until(

EC.element\_to\_be\_clickable((By.XPATH, "//\*[@id=\"root\"]/div[1]/div/div[2]/button[2]"))

)

if cookiePopup is not None:

cookiePopup.click()

firstNonPromoProduct = WebDriverWait(driver, 10).until(

EC.element\_to\_be\_clickable((By.XPATH, "//\*[@id=\"root\"]/div/main/div/div/div/div[5]/div[1]/a/img"))

)

firstNonPromoProduct.click()

time.sleep(5)

add = WebDriverWait(driver, 10).until(

EC.element\_to\_be\_clickable((By.XPATH, "//\*[@id=\"root\"]/div/main/div/div[2]/div/div[2]/div/div[2]/div/button[1]"))

)

add.click()

# cancelPopup = WebDriverWait(driver, 10).until(

# EC.element\_to\_be\_clickable((By.XPATH, "/div/div/div/div/button/div/div/svg"))

# )

# if cancelPopup is not None:

# cancelPopup.click()

time.sleep(5)

bucket = WebDriverWait(driver, 10).until(

EC.element\_to\_be\_clickable((By.XPATH, "//\*[@id=\"headlessui-popover-button-:r4:\"]/button")))

bucket.click()

time.sleep(3)

goToBasket = WebDriverWait(driver, 10).until(

EC.element\_to\_be\_clickable((By.XPATH, "//\*[@id=\"headlessui-popover-panel-:r6:\"]/div/div[3]/a[1]")))

goToBasket.click()

time.sleep(6)

# proceedToCheckout = WebDriverWait(driver, 10).until(

# EC.element\_to\_be\_clickable((By.XPATH, "//\*[@id=\"root\"]/div/main/div/div[4]/div/div[2]/section/button")))

# proceedToCheckout.click()

writeEmail = WebDriverWait(driver, 10).until(

EC.element\_to\_be\_clickable((By.XPATH, "//\*[@id=\"root\"]/div/div/div[3]/div[1]/form/div/div[1]/div/div/input")))

writeEmail.send\_keys("talhashahab1@gmail.com")

submitEmail = WebDriverWait(driver, 10).until(

EC.element\_to\_be\_clickable((By.XPATH, "//\*[@id=\"root\"]/div/div/div[3]/div[1]/form/div/div[2]/button")))

submitEmail.click()

firstName = WebDriverWait(driver, 10).until(

EC.element\_to\_be\_clickable((By.XPATH, "//\*[@id=\"root\"]/div/div/div[3]/div[1]/div[1]/div/div/div[1]/div[2]/div[1]/div/div/input")))

firstName.send\_keys("Talha")

surName = WebDriverWait(driver, 10).until(

EC.element\_to\_be\_clickable((By.XPATH, "//\*[@id=\"root\"]/div/div/div[3]/div[1]/div[1]/div/div/div[1]/div[2]/div[2]/div/div/input")))

surName.send\_keys("Shahab")

phone = WebDriverWait(driver, 10).until(

EC.element\_to\_be\_clickable((By.XPATH, "//\*[@id=\"root\"]/div/div/div[3]/div[1]/div[1]/div/div/div[1]/div[2]/div[3]/div/div/input")))

phone.send\_keys("+923242080303")

time.sleep(7)

# countrySelect = Select(driver.find\_element(By.XPATH("//\*[@id=\"headlessui-combobox-input-:rm:\"]")))

# Locate multiple elements by XPath

elements = driver.find\_elements(By.XPATH, '//\*[@class="Autocomplete\_autocomplete\_\_input\_\_5yxYu Common\_field\_\_\_JumB"]')

if elements:

countrySelectEle = elements[0]

countrySelectEle.click()

countrySelectEle.clear()

countrySelectEle.send\_keys("South Africa")

countrySelectEle.send\_keys(Keys.RETURN)

time.sleep(5)

citySelect = elements[1]

citySelect.click()

# citySelect.clear()

time.sleep(2)

citySelect.send\_keys("Alberton")

time.sleep(2)

citySelect.send\_keys(Keys.RETURN)

time.sleep(8)

# html = driver.find\_element(By.TAG\_NAME, 'html')

# html.send\_keys(Keys.END)

# driver.find\_element(By.TAG\_NAME, 'body').send\_keys(Keys.PAGE\_DOWN)

# driver.execute\_script("window.scrollTo(0, document.body.scrollHeight);")

driver.execute\_script('window.scrollBy(0, 1000)')

# ActionChains(driver)\

# .scroll\_to\_element(iframe)\

# .perform()

postalCode = driver.find\_element(By.XPATH, "//\*[@name=\"shippingAddress.postCode\"]")

driver.execute\_script("arguments[0].scrollIntoView()", postalCode)

postalCode = WebDriverWait(driver, 10).until(

EC.element\_to\_be\_clickable((By.XPATH, "//\*[@name=\"shippingAddress.postCode\"]")))

postalCode.send\_keys("75290")

address = WebDriverWait(driver, 10).until(

EC.element\_to\_be\_clickable((By.XPATH, "//\*[@name=\"shippingAddress.address\"]")))

address.send\_keys("B=124, st-54, Gulistan-e-Jauhar, Karachi, Pakistan.")

totalSum = WebDriverWait(driver, 10).until(

EC.element\_to\_be\_clickable((By.XPATH, "//\*[@id=\"root\"]/div/div/div[3]/div[2]/div/div/div/div[2]/div[4]/p[2]/span")))

if totalSum is not None and len(totalSum.text) > 0:

print("[VALIDATION PASSED] Total Sum is added and shown")

else:

print("[VALIDATION FAILED] Total Sum is NOT added")

assert 1 == "2"

time.sleep(8)

confirmOrder = WebDriverWait(driver, 10).until(

EC.element\_to\_be\_clickable((By.XPATH, "//\*[@id=\"root\"]/div/div/div[3]/div[2]/div/div/div/button")))

confirmOrder.click()

time.sleep(12)

orderConfirmationValidation = WebDriverWait(driver, 10).until(

EC.element\_to\_be\_clickable((By.XPATH, "//\*[@id=\"root\"]/div/main/div/div/div[1]/p[1]")))

if orderConfirmationValidation is not None and orderConfirmationValidation.text == "Your order has been placed successfully.":

print("[VALIDATION PASSED] Order placement validated")

else:

print("[VALIDATION FAILED] Order placement validation failed")

assert 1 == 3

print("First in-stock non-promo product ordered successfully.")

else:

print("[VALIDATION FAILED] Validation failed for Country selection and Checkout process failed as without country selection, checkout is not possible")

assert 1 == 3

finally:

driver.quit()

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main()

**TASK 1 PART B**

import unittest

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

class TestVerifySale50Percent(unittest.TestCase):

def test\_verify\_sale50(self):

driver = webdriver.Chrome()

try:

driver.get("https://teststoreforsouthafri.nextbasket.shop/")

discount\_label\_minus\_50\_percent = WebDriverWait(driver, 10).until(

EC.visibility\_of\_element\_located((By.XPATH, "//\*[@id=\"root\"]/div/main/div/div/div/div[2]/div[1]/div[1]/div[2]/p"))

)

assert discount\_label\_minus\_50\_percent.is\_displayed(), "50% off label is not visible."

#print("50% off label is visible as: ", discount\_label\_minus\_50\_percent.text)

print("50% off label is visible.")

finally:

driver.quit()

if \_\_name\_\_ == '\_\_main\_\_':

unittest.main()

**TASK 1 PART C**

Feature: Checkout Process

Scenario: Successful checkout with valid credit card

Given the user is on the checkout page

When the user enters valid credit card details

And the user clicks the "Place Order" button

Then the order should be successfully placed

And a confirmation message should be displayed

Scenario: Checkout with gift card

Given the user is on the checkout page

When the user enters a valid gift card number

And the user clicks the "Apply Gift Card" button

Then the balance should be applied to the order

And the user should be able to place the order successfully

Scenario: Checkout with invalid CVV

Given the user is on the checkout page

When the user enters a credit card with an invalid CVV

And the user clicks the "Place Order" button

Then an error message "Invalid CVV" should be displayed

Scenario: Checkout with empty required fields

Given the user is on the checkout page

When the user leaves required fields empty

And the user clicks the "Place Order" button

Then an error message "Please fill out this field" should be displayed

Scenario: Checkout with expired credit card

Given the user is on the checkout page

When the user enters an expired credit card

And the user clicks the "Place Order" button

Then an error message "Card expired" should be displayed

Scenario: Checkout with saved credit card

Given the user is logged in

And the user has a saved credit card

When the user selects the saved credit card on the checkout page

And the user clicks the "Place Order" button

Then the order should be successfully placed

And a confirmation message should be displayed

Scenario: Checkout with insufficient funds

Given the user is on the checkout page

When the user enters a credit card with insufficient funds

And the user clicks the "Place Order" button

Then an error message "Insufficient funds" should be displayed

Scenario: Successful checkout with coupon code

Given the user is on the checkout page

When the user enters a valid coupon code

And the user clicks the "Apply Coupon" button

Then the discount should be applied

And the user should be able to place the order successfully

Scenario: Checkout with invalid coupon code

Given the user is on the checkout page

When the user enters an invalid coupon code

And the user clicks the "Apply Coupon" button

Then an error message "Invalid coupon code" should be displayed

Scenario: Checkout with different billing and shipping addresses

Given the user is on the checkout page

When the user enters a different billing address from the shipping address

And the user clicks the "Place Order" button

Then the order should be successfully placed

And a confirmation message should be displayed

**TASK 1 PART 4**

Feature: Checkout Process

Scenario: Incorrect shipping calculation for multiple items

Given the user has multiple items in the cart

When the user proceeds to the checkout page

And the user enters their shipping address

Then the shipping cost should be calculated correctly

But the shipping cost is inaccurately high

Scenario: Promo code not applied at checkout

Given the user has a valid promo code

When the user enters the promo code at checkout

And the user clicks the "Apply Promo Code" button

Then the discount should be applied to the total

But the promo code is not applied and no discount is shown

**TASK 2 PART a**

let response = pm.response.json();

if (Array.isArray(response) && response.length > 0) {

let lastBooking = response[response.length - 1];

let lastBookingId = lastBooking.id;

pm.collectionVariables.set("createdBookingId", lastBookingId);

pm.collectionVariables.set("createdBookingId", lastBookingId);

// Set variables based on response data

// pm.collectionVariables.set("bookingId", response.id); // Collection variable

pm.environment.set("createdBookingId", lastBookingId); // Environment variable

pm.globals.set("createdBookingId", lastBookingId); // Global variable

pm.variables.set("createdBookingId", lastBookingId); // Local variable

console.log("Last Booking ID:", lastBookingId);

}

pm.test("Response has booking id", function () {

var jsonData = pm.response.json();

pm.expect(jsonData).to.have.property("bookingid");

let response = pm.response.json();

if (Array.isArray(response) && response.length > 0) {

let lastBooking = response[response.length - 1];

let lastBookingId = lastBooking.id;

pm.collectionVariables.set("createdBookingId", lastBookingId);

// Set variables based on response data

// pm.collectionVariables.set("bookingId", response.id); // Collection variable

pm.environment.set("createdBookingId", lastBookingId); // Environment variable

pm.globals.set("createdBookingId", lastBookingId); // Global variable

pm.variables.set("createdBookingId", lastBookingId); // Local variable

console.log("Last Booking ID:", lastBookingId);

}

});

pm.test("Response has correct first name", function () {

var jsonData = pm.response.json();

pm.expect(jsonData.booking.firstname).to.eql("Jim");

});

pm.test("Response has correct last name", function () {

var jsonData = pm.response.json();

pm.expect(jsonData.booking.lastname).to.eql("Brown");

});

pm.test("Response has correct total price", function () {

var jsonData = pm.response.json();

pm.expect(jsonData.booking.totalprice).to.eql(111);

});

pm.test("Response has correct deposit paid", function () {

var jsonData = pm.response.json();

pm.expect(jsonData.booking.depositpaid).to.be.true;

});

pm.test("Response has correct booking dates", function () {

var jsonData = pm.response.json();

pm.expect(jsonData.booking.bookingdates.checkin).to.eql("2024-07-01");

pm.expect(jsonData.booking.bookingdates.checkout).to.eql("2024-07-02");

});

pm.test("Response has correct additional needs", function () {

var jsonData = pm.response.json();

pm.expect(jsonData.booking.additionalneeds).to.eql("Breakfast");

});

// pm.test("Response status code is 404", function () {

// pm.response.to.have.status(404);

// });

**TASK 2 PART B**

let createdBookingId = pm.collectionVariables.get("createdBookingId");

let createdBookingId2 = pm.environment.get("createdBookingId");

let createdBookingId3 = pm.globals.get("createdBookingId");

let createdBookingId4 = pm.variables.get("createdBookingId");

pm.test("New booking ID is present in the list", function () {

var jsonData = pm.response.json();

// pm.variables.set("createdBookingId", lastBookingId); // Local variable

// Get variables

var bookingIds = jsonData.map(booking => booking.bookingid);

pm.expect(bookingIds).to.include(parseInt(createdBookingId4));

});

**README.MD**

# Next Basket Second Test

Project description:

Project contains both Task 1 and Task 2 that are solved.

Technologies used:

Technologies used are Postman tool

Other valuable information:

Tasks took 4 hours maximum

Task 1:

Run Commands

1. cd Task1

2. python -m unittest 1\_ordernow\_test.py

3. python -m unittest 2\_verifysale\_test.py

**.env**

[EMAIL=talhashahab1@gmail.com](mailto:EMAIL=talhashahab1@gmail.com)

**.gitignore**

report.html

\_\_pycache\_\_

.pytest\_cache

assets