**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH**



**BÁO CÁO BÀI TẬP LỚN GIỮA KỲ MÔN KIỂM THỬ PHẦN MỀM**

|  |  |  |
| --- | --- | --- |
| Họ Tên | : | Nguyễn Thị Mỹ Vân |
| MSSV | : | 2151053067 |
| Số thứ tự | : | 94 |
| Giảng viên hướng dẫn | : | Ths. Nguyễn Thị Ngọc Thanh |

Hồ Chí Minh, ngày 15 tháng 4 năm 2024

MỤC LỤC

[MỤC LỤC HÌNH ẢNH 5](#_Toc166418336)

[Phần A: KIỂM THỬ NUNIT TEST 9](#_Toc166418337)

[I. Khởi tạo project cho Nunit: 9](#_Toc166418338)

[1. Tạo Project: 9](#_Toc166418339)

[2. Mô tả hoạt động của project: 10](#_Toc166418340)

[II. Đặt tả chức năng: 10](#_Toc166418341)

[1. Chức năng 1: Kiểm tra giá trị nhập vào: 10](#_Toc166418342)

[2. Chức năng 2: Tính diện tích tam giác: 10](#_Toc166418343)

[III. Tạo giao diện: 10](#_Toc166418344)

[IV. Cài đặt sự kiện: 11](#_Toc166418345)

[1. Tạo class chứa các hàm chức năng: 11](#_Toc166418346)

[2. Viết code từng chức năng: 12](#_Toc166418347)

[a. Chức năng 1: Kiểm tra tam giác: 12](#_Toc166418348)

[b. Chức năng 2: Tính diện tích: 13](#_Toc166418349)

[c. Code của cả class: 13](#_Toc166418350)

[3. Cài đặt sự kiện cho từng button: 15](#_Toc166418351)

[a. Chức năng 1: Kiểm tra giá trị nhập vào: 15](#_Toc166418352)

[b. Chức năng 2: Tính diện tích tam giác: 16](#_Toc166418353)

[V. Chạy project kiểm tra kết quả: 17](#_Toc166418354)

[VI. Kiểm thử Nunit: 20](#_Toc166418355)

[1. Tạo project, cài đặt References, tạo Datasource: 20](#_Toc166418356)

[a. Tạo project kiểm thử: 20](#_Toc166418357)

[b. Cài đặt References: 20](#_Toc166418358)

[c. Tạo Datasource(Dữ liệu kiểm thử): 20](#_Toc166418359)

[2. Mô tả TestCase: 21](#_Toc166418360)

[a. Chức năng 1: Kiểm tra tam giác: 22](#_Toc166418361)

[b. Chức năng 2: Tính diện tích tam giác: 24](#_Toc166418362)

[3. Kết quả thu được: 25](#_Toc166418363)

[a. Chức năng 1: Kiểm tra tam giác: 25](#_Toc166418364)

[b. Chức năng 2: Tính diện tích: 26](#_Toc166418365)

[Phần B. SELENIUM WEB DRIVER 27](#_Toc166418366)

[I. Tạo project cho Test Selenium, cài đặt môi trường: 27](#_Toc166418367)

[1. Tạo project: 27](#_Toc166418368)

[2. Cài đặt môi trường: 29](#_Toc166418369)

[3. Môi trường cài đặt: 32](#_Toc166418370)

[II. Đặt tả chức năng: 32](#_Toc166418371)

[1. Chức năng 1: Tìm kiếm khóa học: 32](#_Toc166418372)

[a. Mục tiêu: 32](#_Toc166418373)

[b. Cách hoạt động của chức năng: 33](#_Toc166418374)

[2. Chức năng 2: Học IELTS: 33](#_Toc166418375)

[a. Mục tiêu: 33](#_Toc166418376)

[b. Cách hoạt động của chức năng: 33](#_Toc166418377)

[III. Bước thực hiện testcase của chức năng ‘Tìm kiếm khóa học’: 33](#_Toc166418378)

[1. Đặt tả testcase: 33](#_Toc166418379)

[2. Thực hiện test chức năng ‘Tìm kiếm khóa học’: 34](#_Toc166418380)

[a. Testcase 1: Tìm kiếm thành công, hoàn thành bài tập: 35](#_Toc166418381)

[b. Testcase 2: Nhập sai và không tìm kiếm ra kết quả khóa học: 43](#_Toc166418382)

[IV. Bước thực hiện testcase của chức năng ‘Học IELTS’: 47](#_Toc166418383)

[1. Đặt tả testcase: 47](#_Toc166418384)

[2. Thực hiện test chức năng ‘Học IELTS’: 48](#_Toc166418385)

[a. Testcase 1: Làm bài: 48](#_Toc166418386)

[b. Testcase 2: Học từ vựng: 53](#_Toc166418387)

[c. Testcase 3: Kiểm tra đề và đáp án: 57](#_Toc166418388)

[Phần C: API POSTMAN 62](#_Toc166418389)

[I. Tạo JSON Server: 62](#_Toc166418390)

[1. Cài đặt NodeJS: 62](#_Toc166418391)

[a. Môi trường cài đặt: 62](#_Toc166418392)

[2. Cài đặt gói sử dụng Node package manager(npm): 63](#_Toc166418393)

[3. Tạo file API trên server ảo: 67](#_Toc166418394)

[a. Tạo file sp\_94\_Van.json: 67](#_Toc166418395)

[b. Cập nhập file package.json: 69](#_Toc166418396)

[c. Chạy npm start: 70](#_Toc166418397)

[II. Kiểm thử API Postman: 72](#_Toc166418398)

[1. Tạo Workspace: 72](#_Toc166418399)

[a. Tạo một workspace mới: 72](#_Toc166418400)

[2. Phương thức Get: 74](#_Toc166418401)

[a. Tạo và lưu một request: Get: 74](#_Toc166418402)

[b. API testcase: Get: 77](#_Toc166418403)

[3. Phương thức Post: 82](#_Toc166418404)

[a. Tạo và lưu một request: Post: 82](#_Toc166418405)

[b. API testcase: Post: 84](#_Toc166418406)

[4. Phương thức Put/Patch: 89](#_Toc166418407)

[a. Tạo và lưu một request: Put: 89](#_Toc166418408)

[b. API testcase: Put: 91](#_Toc166418409)

[5. Phương thức Delete: 95](#_Toc166418410)

[a. Tạo và lưu một request: Delete: 95](#_Toc166418411)

[b. API testcase: Delete: 97](#_Toc166418412)

# MỤC LỤC HÌNH ẢNH

[Hình 1: Tạo Project Cho NUnit 9](#_Toc166417858)

[Hình 2: Đặt tên, nơi lưu trữ project 9](#_Toc166417859)

[Hình 3: Đặt tên Phần NUnit 10](#_Toc166417860)

[Hình 4: Giao diện project 11](#_Toc166417861)

[Hình 5: Đặt tên class 12](#_Toc166417862)

[Hình 6: Code chức năng kiểm tra tam giác 12](#_Toc166417863)

[Hình 7: Code chức năng xác định loại tam giác 13](#_Toc166417864)

[Hình 8: Code chức năng tính diện tích 13](#_Toc166417865)

[Hình 9: Code chức năng xử lý sự kiện Click nút Kiểm Tra 15](#_Toc166417866)

[Hình 10: Code chức năng xử lý sự kiện nút Tính 16](#_Toc166417867)

[Hình 11: Chạy project: Đây không phải tam giác 18](#_Toc166417868)

[Hình 12: Chạy project: Đây là tam giác thường 18](#_Toc166417869)

[Hình 13: Chạy project: Đây là tam giác cân 19](#_Toc166417870)

[Hình 14: Chạy project: Đây là tam giác đều 19](#_Toc166417871)

[Hình 15: Tạo project Unit test 20](#_Toc166417872)

[Hình 16: Đặt tên, nơi lưu trữ cho project Nunit test 20](#_Toc166417873)

[Hình 17: Add Reference cho NUnit 20](#_Toc166417874)

[Hình 18: DataSource cho chức năng kiểm tra tam giác 21](#_Toc166417875)

[Hình 19: DataSource cho chức năng tính diện tích 21](#_Toc166417876)

[Hình 20: Code tạo đối tượng test 22](#_Toc166417877)

[Hình 21: Kết quả Test của TestMethod kiểm tra loại tam giác 25](#_Toc166417878)

[Hình 22: Kết quả Test của TestMethod Kiểm tra có phải là tam giác 26](#_Toc166417879)

[Hình 23: Kết quả Test của TestMethod tính diện tích 26](#_Toc166417880)

[Hình 24: Tạo Project mới SeleniumWebdriver\_94\_Van 27](#_Toc166417881)

[Hình 25: Đặt tên, nơi luu trữ cho Project Selenium 28](#_Toc166417882)

[Hình 26: Hoàn tất tạo Project SeleniumWebdriver 29](#_Toc166417883)

[Hình 27: Dẫn đến NuGet Packages 30](#_Toc166417884)

[Hình 28: Các môi trường cần cài đặt 31](#_Toc166417885)

[Hình 29: Cài đặt Selenium.WebDriver 31](#_Toc166417886)

[Hình 30: Phiên bản Chrome trong thiết bị 32](#_Toc166417887)

[Hình 31: Cài đặt Selenium.WebDriver.Chrome 32](#_Toc166417888)

[Hình 32: Đặt tên cho file test chức năng "Tìm kiếm khoa học" 34](#_Toc166417889)

[Hình 33: Code điều hướng đến đường dẫn URL 35](#_Toc166417890)

[Hình 34:Inspect element tìm kiếm 35](#_Toc166417891)

[Hình 35: Element tìm kiếm 35](#_Toc166417892)

[Hình 36: Code phần tìm kiếm 36](#_Toc166417893)

[Hình 37: Inspect của nút tìm kiếm 36](#_Toc166417894)

[Hình 38: Code phần nút tìm kiếm 36](#_Toc166417895)

[Hình 39: Lấy element cho bài học 37](#_Toc166417896)

[Hình 40: Code phần chọn bài học 37](#_Toc166417897)

[Hình 41: Chọn Inspect cho nút Play 38](#_Toc166417898)

[Hình 42: Lấy element cho nút Play 38](#_Toc166417899)

[Hình 43: Code cho phần nút Play 39](#_Toc166417900)

[Hình 44: Chạy kết quả cho phần Play 39](#_Toc166417901)

[Hình 45: Lấy element cho ô nhập kết quả 39](#_Toc166417902)

[Hình 46: Code phần nhập kết quả 40](#_Toc166417903)

[Hình 47:Lấy element cho Script kết quả 40](#_Toc166417904)

[Hình 48: Code cho phần xem script xem kết quả 41](#_Toc166417905)

[Hình 49: Chạy kết quả cho phần nhập đáp án và xem script kết quả 41](#_Toc166417906)

[Hình 50: Kết quả testcase Tìm kiếm thành công 42](#_Toc166417907)

[Hình 51: Code điều hướng đến đường dẫn URL 43](#_Toc166417908)

[Hình 52:Inspect element tìm kiếm 43](#_Toc166417909)

[Hình 53: Element tìm kiếm 43](#_Toc166417910)

[Hình 54: Code phần tìm kiếm 44](#_Toc166417911)

[Hình 55: Inspect của nút tìm kiếm 44](#_Toc166417912)

[Hình 56: Code phần nút tìm kiếm 45](#_Toc166417913)

[Hình 57: Lấy element phần không tìm ra khóa học 45](#_Toc166417914)

[Hình 58: Code phần bắt Lỗi tìm kiếm khi nhập sai khóa học 46](#_Toc166417915)

[Hình 59: Kết quả testcase Nhập sai và không tìm kiếm ra kết quả khóa học 46](#_Toc166417916)

[Hình 60: Đặt tên cho file test chức năng "Học IELTS" 48](#_Toc166417917)

[Hình 61: Code điều hướng đến đường dẫn URL 49](#_Toc166417918)

[Hình 62: Inspect của 'Bài kiểm tra' 49](#_Toc166417919)

[Hình 63: Element của 'Bài kiểm tra' 50](#_Toc166417920)

[Hình 64: Code của phần chọn bài kiểm tra 50](#_Toc166417921)

[Hình 65: Inspect của 'Làm bài' 51](#_Toc166417922)

[Hình 66: Elements của 'Làm bài' 51](#_Toc166417923)

[Hình 67: Code phần Làm bài 51](#_Toc166417924)

[Hình 68: Kết quả testcase 1 Làm bài 52](#_Toc166417925)

[Hình 69: Code điều hướng đến đường dẫn URL 53](#_Toc166417926)

[Hình 70: Inspect của 'Bài kiểm tra' 53](#_Toc166417927)

[Hình 71: Element của 'Bài kiểm tra' 54](#_Toc166417928)

[Hình 72: Code của phần chọn bài kiểm tra 54](#_Toc166417929)

[Hình 73: Element của nút lựa chọn 55](#_Toc166417930)

[Hình 74: Code của nút lựu chọn 55](#_Toc166417931)

[Hình 75: Element của Từ vựng 56](#_Toc166417932)

[Hình 76: Code của Từ vựng 56](#_Toc166417933)

[Hình 77: Kết quả testcase 2 Học từ vựng 57](#_Toc166417934)

[Hình 78: Code điều hướng đến đường dẫn URL 57](#_Toc166417935)

[Hình 79: Inspect của 'Bài kiểm tra' 58](#_Toc166417936)

[Hình 80: Element của 'Bài kiểm tra' 58](#_Toc166417937)

[Hình 81: Code của phần chọn bài kiểm tra 59](#_Toc166417938)

[Hình 82: Element của nút lựa chọn 59](#_Toc166417939)

[Hình 83: Element của Đề và làm bài 60](#_Toc166417940)

[Hình 84: Code của Đề và đáp án 60](#_Toc166417941)

[Hình 85: Kết quả của testcase 3 Đề và đáp án 61](#_Toc166417942)

[Hình 86: Download Nodejs 62](#_Toc166417943)

[Hình 87: Phiên bản của Nodejs và npm 63](#_Toc166417944)

[Hình 88: Tạo folder lưu package.json 63](#_Toc166417945)

[Hình 89: Trỏ đường dẫn đến folder mới tạo 64](#_Toc166417946)

[Hình 90: Nhập npm init để tạo json 64](#_Toc166417947)

[Hình 91: Nhập các thông tin mong muốn 65](#_Toc166417948)

[Hình 92: Cấu trúc file package.json 66](#_Toc166417949)

[Hình 93: Hoàn tất tạo project package.json 66](#_Toc166417950)

[Hình 94: Kiểm tra folder 'jsonserver\_94\_Van' 66](#_Toc166417951)

[Hình 95: Tạo file 'sp\_94\_Van' trên notepad 67](#_Toc166417952)

[Hình 96: Dữ liệu của sp\_94\_Van.json 67](#_Toc166417953)

[Hình 97: Thêm dòng start vào package.json 70](#_Toc166417954)

[Hình 98: Nhập lệnh npm start 71](#_Toc166417955)

[Hình 99: Kiểm tra kết quả API vừa tạo 72](#_Toc166417956)

[Hình 100: Create Workspace 72](#_Toc166417957)

[Hình 101: Chọn Blank Workspace 73](#_Toc166417958)

[Hình 102: Đặt tên workspace 74](#_Toc166417959)

[Hình 103: Tạo collection 75](#_Toc166417960)

[Hình 104: Đặt tên collection 75](#_Toc166417961)

[Hình 105: Tạo request Get 76](#_Toc166417962)

[Hình 106: Đặt tên cho request 76](#_Toc166417963)

[Hình 107: Send request Get 77](#_Toc166417964)

[Hình 108: Tạo testcase cho getListAPI\_94\_Van 77](#_Toc166417965)

[Hình 109: Chọn Status code: Code name has string cho Get Method 78](#_Toc166417966)

[Hình 110:Get Method: testcase 1 Status Code Name 78](#_Toc166417967)

[Hình 111: Kết quả testcase 1 Status Code Name 79](#_Toc166417968)

[Hình 112: Get Method: testcase 2 Kiểm tra giá sản phẩm có id = 1 79](#_Toc166417969)

[Hình 113: Code tc2\_TestPriceId1\_94\_Van 80](#_Toc166417970)

[Hình 114: Kết quả testcase 2 Kiểm tra giá sản phẩm có id = 1 80](#_Toc166417971)

[Hình 115: Kiểm tra kết quả bằng collection 81](#_Toc166417972)

[Hình 116: Chọn Run getAPIpostman\_94\_Van 81](#_Toc166417973)

[Hình 117: Kết quả của getAPIpostman\_94\_Van 82](#_Toc166417974)

[Hình 118: Tạo Collection postAPIpostman\_94\_Van 82](#_Toc166417975)

[Hình 119: Tạo request postCreateAPIpostman\_94\_Van 83](#_Toc166417976)

[Hình 120: Các bước thực hiện Post 83](#_Toc166417977)

[Hình 121: Thêm dữ liệu bằng Post Method 83](#_Toc166417978)

[Hình 122: Send request Post 84](#_Toc166417979)

[Hình 123: Tạo testcase cho postCreateAPI\_94\_Van 85](#_Toc166417980)

[Hình 124: Chọn Status code: Successful POST request Post Method 85](#_Toc166417981)

[Hình 125: Post Method: testcase 1 Successful Post 86](#_Toc166417982)

[Hình 126: Kết quả testcase 1 Successful Post 86](#_Toc166417983)

[Hình 127: Post Method Kiểm tra đã thêm được sản phẩm mới chưa 87](#_Toc166417984)

[Hình 128: Code testcase 2 Kiểm tra đã thêm được sản phẩm mới chưa 88](#_Toc166417985)

[Hình 129: Kết quả testcase 2 Kiểm tra đã thêm được sản phẩm mới chưa 88](#_Toc166417986)

[Hình 130: Tạo Collection putAPIpostman\_94\_Van 89](#_Toc166417987)

[Hình 131: Tạo request putUpdateAPIpostman\_94\_Van 90](#_Toc166417988)

[Hình 132: Các bước thực hiện Put 90](#_Toc166417989)

[Hình 133: Send request Put 91](#_Toc166417990)

[Hình 134: Tạo testcase cho putUpdateAPIpostman\_94\_Van 91](#_Toc166417991)

[Hình 135: Chọn Response time is less than 200ms 92](#_Toc166417992)

[Hình 136: Put Method: testcase 1 Response time less 200ms 92](#_Toc166417993)

[Hình 137: Kết quả testcase 1 Response time less 200ms 93](#_Toc166417994)

[Hình 138: Code testcase 2 Kiểm tra dữ liệu có còn sản phẩm ban đầu không 94](#_Toc166417995)

[Hình 139: Kết quả của testcase 2 Kiểm tra dữ liệu có còn sản phẩm ban đầu không 95](#_Toc166417996)

[Hình 140: Tạo Collection delAPIpostman\_94\_Van 96](#_Toc166417997)

[Hình 141: Tạo request cho delAPIpostman\_94\_Van 96](#_Toc166417998)

[Hình 142: Send request Delete 96](#_Toc166417999)

[Hình 143: Kiểm tra kết quả Delete 97](#_Toc166418000)

[Hình 144: Tạo testcase cho DelDeleteAPIpostman\_94\_Van 98](#_Toc166418001)

[Hình 145: Chọn Status code: Code name has string cho Delete Method 98](#_Toc166418002)

[Hình 146: Delete Method: testcase 1 Status Code Name 99](#_Toc166418003)

[Hình 147: Kết quả testcase 1 Status Code Name 99](#_Toc166418004)

[Hình 148: Code testcase 2 Response time less 200ms 100](#_Toc166418005)

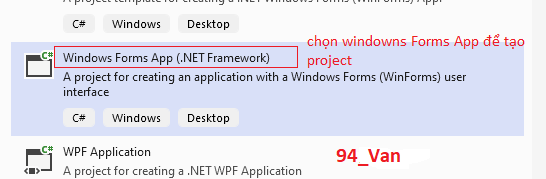
[Hình 149: Kết quả testcase 2 Response time less 200ms 101](#_Toc166418006)

# Phần A: KIỂM THỬ NUNIT TEST

## Khởi tạo project cho Nunit:

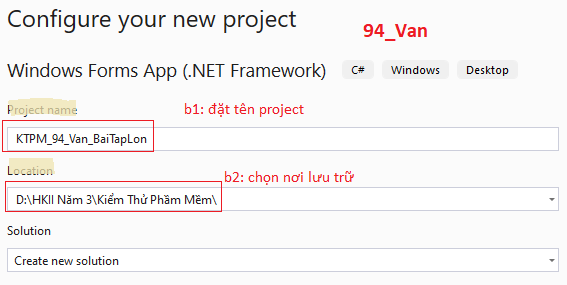
### Tạo Project:

* Tạo project ngôn ngữ là C#, platform là Windows, chọn loại project là Windows Forms App(.NET Framework).



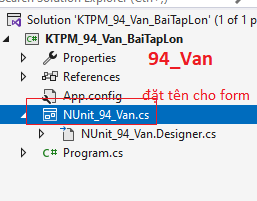
Hình 1: Tạo Project Cho NUnit

* Đặt tên project là ‘KTPM\_94\_Van\_BaiTapLon’, chọn nơi lưu trữ.



Hình 2: Đặt tên, nơi lưu trữ project

* Đặt tên cho phần A là Nunit\_94\_Van



Hình 3: Đặt tên Phần NUnit

### Mô tả hoạt động của project:

* Project được tạo ra hỗ trợ người dùng nhập ba giá trị số vào ô nhập liệu. Kiểm tra ba giá trị ấy có tạo nên tam giác hay không. Nếu là tam giác thì kiểm tra tam giác ấy là loại tam giác nào ( cân , đều, thường), bên cạnh đó là tính diện tích của tam giác đó.

## Đặt tả chức năng:

### Chức năng 1: Kiểm tra giá trị nhập vào:

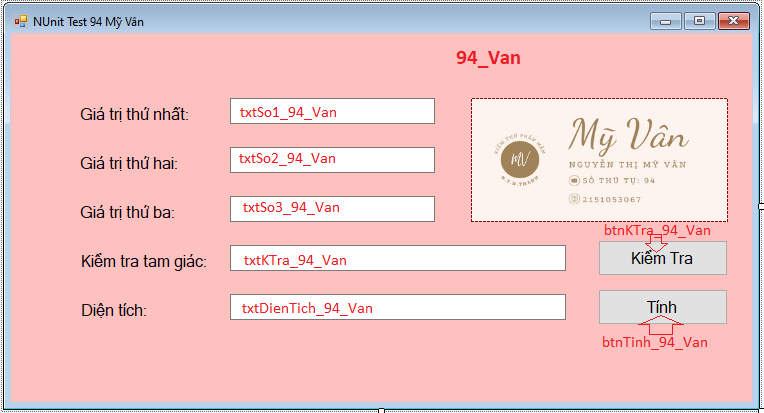
* Người dùng nhập vào 3 cạnh của một tam giác.
* Hệ thống kiểm tra xem ba cạnh đó có tạo thành một tam giác hay không dựa trên nguyên tắc: tổng hai cạnh bất kỳ luôn lớn hơn cạnh còn lại.
* Nếu ba cạnh tạo thành một tam giác, hệ thống xác định loại tam giác (đều, cân, thường).
* Nếu không, hệ thống thông báo rằng ba cạnh không tạo thành một tam giác.

### Chức năng 2: Tính diện tích tam giác:

* Nếu ba cạnh tạo thành một tam giác, hệ thống tính diện tích của tam giác đó bằng công thức Heron.
* Nếu ba cạnh không tạo thành một tam giác, gửi thông báo rằng “Đây không phải là tam giác”.

## Tạo giao diện:

* Sử dụng các công cụ trong Toolbox để tạo giao diện mong muốn. Gồm năm label lần lượt là ‘Giá trị thứ nhất’, ‘Giá trị thứ hai’, ‘Giá trị thứ ba’, ‘Kiểm tra tam giác’, ‘Diện tích’. Và năm textbox tương ứng với từng label với tên được đặt theo quy tắc như ‘txtSo1\_94\_Van’, ‘txtSo2\_94\_Van’, ‘txtSo3\_94\_Van’, ‘txtKTra\_94\_Van’, ‘txtDienTich\_94\_Van’. Và hai button với chức năng là kiểm tra các giá trị nhập vào, và tính diện tích nếu 3 giá trị tạo nên tam giác với tên được đặt theo quy tắc tương ứng là ‘btnKTra\_94\_Van’, ‘btnTinh\_94\_Van’.

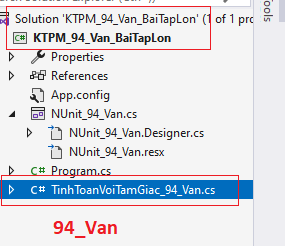


Hình 4: Giao diện project

## Cài đặt sự kiện:

### Tạo class chứa các hàm chức năng:

* Trong project, tạo một class mới có tên là ‘TinhToanVoiTamGiac\_94\_Van.cs’ chứa các hàm kiểm tra và tính toán.

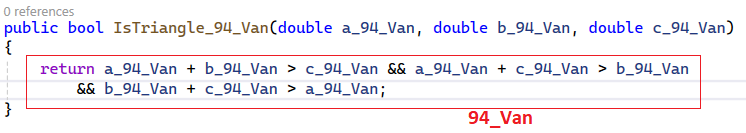


Hình 5: Đặt tên class

### Viết code từng chức năng:

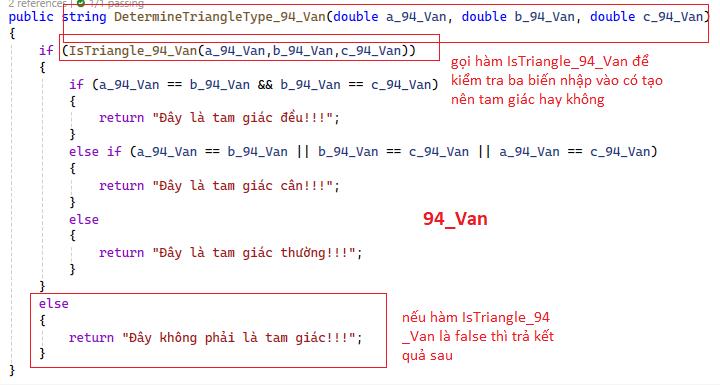
#### Chức năng 1: Kiểm tra tam giác:

* Kiểm tra ba giá trị nhập vào có là tam giác hay không, sử dụng định lý về tam giác là tổng hai cạnh luôn lớn hơn cạnh còn lại.



Hình 6: Code chức năng kiểm tra tam giác

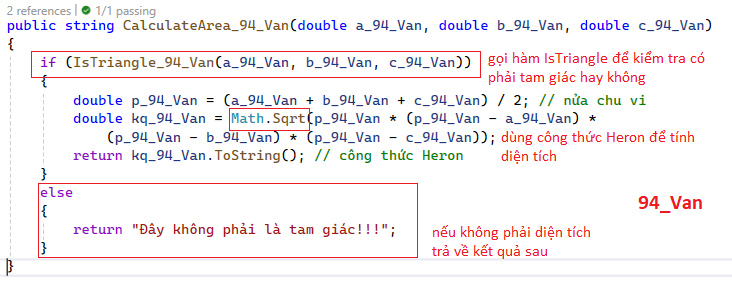
* Nếu đã là tam giác thì kiểm tra kết quả tam giác ấy là tam giác nào, nếu hai cạnh trong ba cạnh bằng nhau thì được gọi là tam giác cân trả về kết quả ‘Đây là tam giác cân!!!’. Còn nếu cả ba cạnh đều bằng nhau được gọi là tam giác đều và trả kết quả ‘Đây là tam giác đều!!!’. Ngược lại, nếu ba cạnh đều không bằng nhau thì nói ‘Đây là tam giác thường!!!’. Nếu không phải là tam giác thì trả về kết quả ‘Đây không phải là tam giác!!!’.



Hình 7: Code chức năng xác định loại tam giác

#### Chức năng 2: Tính diện tích:

* Nếu 3 giá trị nhập vào đã được xác định là tam giác, thì ta có thể tính kết quả diện tích tam giác khi biết 3 cạnh bằng công thức Heron. Sử dụng công thức Heron đã được chứng minh: Sqrt(p\_94\_Van \* (p\_94\_Van - a\_94\_Van) \* (p\_94\_Van - b\_94\_Van) \* (p\_94\_Van - c\_94\_Van)). Với p là nửa chu vi tam giác: (a\_94\_Van + b\_94\_Van + c\_94\_Van) / 2.



Hình 8: Code chức năng tính diện tích

#### Code của cả class:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace KTPM\_94\_Van\_BaiTapLon

{

public class TinhToanVoiTamGiac\_94\_Van

{

public bool IsTriangle\_94\_Van(double a\_94\_Van, double b\_94\_Van, double c\_94\_Van)

{

return a\_94\_Van + b\_94\_Van > c\_94\_Van && a\_94\_Van + c\_94\_Van > b\_94\_Van

&& b\_94\_Van + c\_94\_Van > a\_94\_Van;

}

public string DetermineTriangleType\_94\_Van(double a\_94\_Van, double b\_94\_Van, double c\_94\_Van)

{

if (IsTriangle\_94\_Van(a\_94\_Van,b\_94\_Van,c\_94\_Van))

{

if (a\_94\_Van == b\_94\_Van && b\_94\_Van == c\_94\_Van)

{

return "Đây là tam giác đều!!!";

}

else if (a\_94\_Van == b\_94\_Van || b\_94\_Van == c\_94\_Van || a\_94\_Van == c\_94\_Van)

{

return "Đây là tam giác cân!!!";

}

else

{

return "Đây là tam giác thường!!!";

}

}

else

{

return "Đây không phải là tam giác!!!";

}

}

public string CalculateArea\_94\_Van(double a\_94\_Van, double b\_94\_Van, double c\_94\_Van)

{

if (IsTriangle\_94\_Van(a\_94\_Van, b\_94\_Van, c\_94\_Van))

{

double p\_94\_Van = (a\_94\_Van + b\_94\_Van + c\_94\_Van) / 2;

// nửa chu vi

double kq\_94\_Van = Math.Sqrt(p\_94\_Van \* (p\_94\_Van - a\_94\_Van) \*

(p\_94\_Van - b\_94\_Van) \* (p\_94\_Van - c\_94\_Van));

return kq\_94\_Van.ToString(); // công thức Heron

}

else

{

return "Đây không phải là tam giác!!!";

}

}

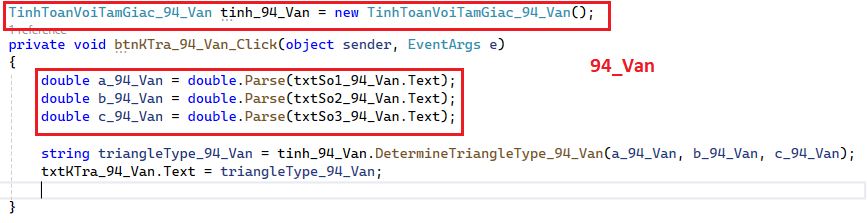
}

}

### Cài đặt sự kiện cho từng button:

#### Chức năng 1: Kiểm tra giá trị nhập vào:

* Tạo sự kiện Click cho button ‘Kiểm Tra’. Khai báo một đối tượng từ class ‘TinhToanVoiTamGiac\_94\_Van()’ là ‘tinh\_94\_Van’, ba số nguyên là 3 giá trị mà người dùng nhập vào textbox. Từ ba số nguyên gọi hàm ‘DetermineTriangleType\_94\_Van’ để kiểm tra tam giác ấy là tam giác gì và trả về cho textbox kết quả string ‘triangleType\_94\_Van’.



Hình 9: Code chức năng xử lý sự kiện Click nút Kiểm Tra

* Code

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using KTPM\_94\_Van\_BaiTapLon;

namespace KTPM\_94\_Van\_BaiTapLon

{

public partial class NUnit\_94\_Van : Form

{

public NUnit\_94\_Van()

{

InitializeComponent();

}

TinhToanVoiTamGiac\_94\_Van tinh\_94\_Van = new TinhToanVoiTamGiac\_94\_Van();

private void btnKTra\_94\_Van\_Click(object sender, EventArgs e)

{

double a\_94\_Van = double.Parse(txtSo1\_94\_Van.Text);

double b\_94\_Van = double.Parse(txtSo2\_94\_Van.Text);

double c\_94\_Van = double.Parse(txtSo3\_94\_Van.Text);

string triangleType\_94\_Van = tinh\_94\_Van.DetermineTriangleType\_94\_Van(a\_94\_Van, b\_94\_Van, c\_94\_Van);

txtKTra\_94\_Van.Text = triangleType\_94\_Van;

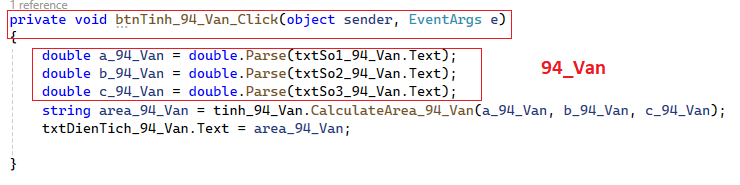
}

}

}

#### Chức năng 2: Tính diện tích tam giác:

* Tạo sự kiện Click cho button ‘Tính’. Khai báo một đối tượng từ class ‘TinhToanVoiTamGiac\_94\_Van()’ là ‘tinh\_94\_Van’, ba số nguyên là 3 giá trị mà người dùng nhập vào textbox. Từ ba số nguyên gọi hàm ‘IsTriangle\_94\_Van’ để kiểm tra ba giá trị có phải tam giác hay không, nếu không phải trả lại cho textbox diện tích là ‘Đây không phải là tam giác!!!’. Nếu nó tạo nên tam giác thì gọi hàm ‘CalculateArea\_94\_Van’ để tính diện tích tam giác và trả về kết quả vào textbox diện tích.



Hình 10: Code chức năng xử lý sự kiện nút Tính

* Code:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using KTPM\_94\_Van\_BaiTapLon;

namespace KTPM\_94\_Van\_BaiTapLon

{

public partial class NUnit\_94\_Van : Form

{

public NUnit\_94\_Van()

{

InitializeComponent();

}

TinhToanVoiTamGiac\_94\_Van tinh\_94\_Van = new TinhToanVoiTamGiac\_94\_Van();

private void btnKTra\_94\_Van\_Click(object sender, EventArgs e)

{

double a\_94\_Van = double.Parse(txtSo1\_94\_Van.Text);

double b\_94\_Van = double.Parse(txtSo2\_94\_Van.Text);

double c\_94\_Van = double.Parse(txtSo3\_94\_Van.Text);

if (tinh\_94\_Van.IsTriangle\_94\_Van(a\_94\_Van, b\_94\_Van, c\_94\_Van))

{

string triangleType\_94\_Van = tinh\_94\_Van.DetermineTriangleType\_94\_Van(a\_94\_Van, b\_94\_Van, c\_94\_Van);

txtKTra\_94\_Van.Text = triangleType\_94\_Van;

}

else

{

txtKTra\_94\_Van.Text = "Đây không phải là tam giác!!!";

}

}

private void btnTinh\_94\_Van\_Click(object sender, EventArgs e)

{

double a\_94\_Van = double.Parse(txtSo1\_94\_Van.Text);

double b\_94\_Van = double.Parse(txtSo2\_94\_Van.Text);

double c\_94\_Van = double.Parse(txtSo3\_94\_Van.Text);

string area\_94\_Van = tinh\_94\_Van.CalculateArea\_94\_Van(a\_94\_Van, b\_94\_Van, c\_94\_Van);

txtDienTich\_94\_Van.Text = area\_94\_Van;

}

}

}

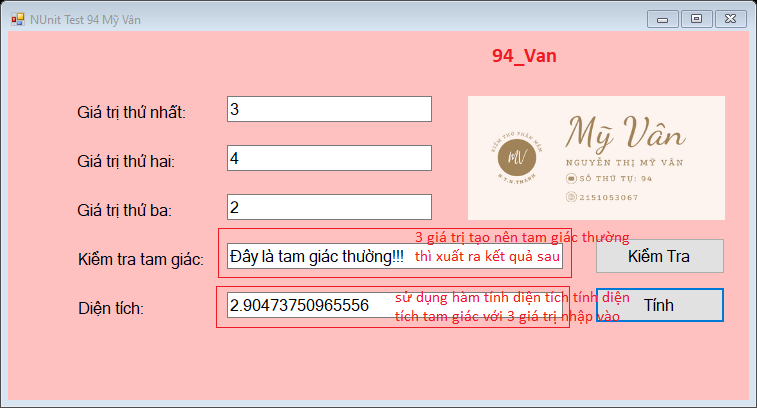
## Chạy project kiểm tra kết quả:

* Kết quả không phải là tam giác:



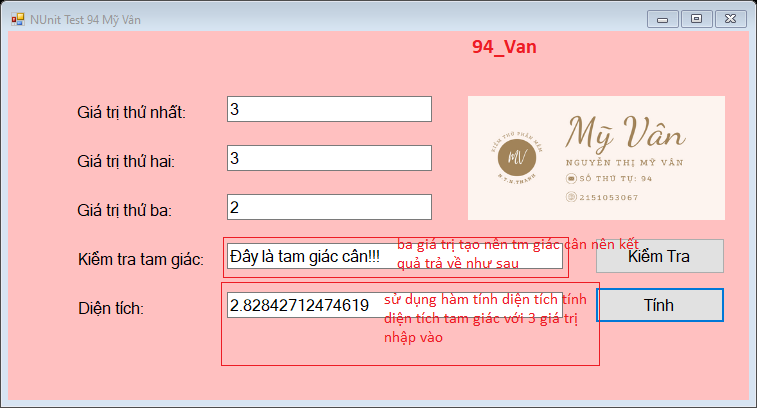
Hình 11: Chạy project: Đây không phải tam giác

* Kết quả là tam giác bình thường:



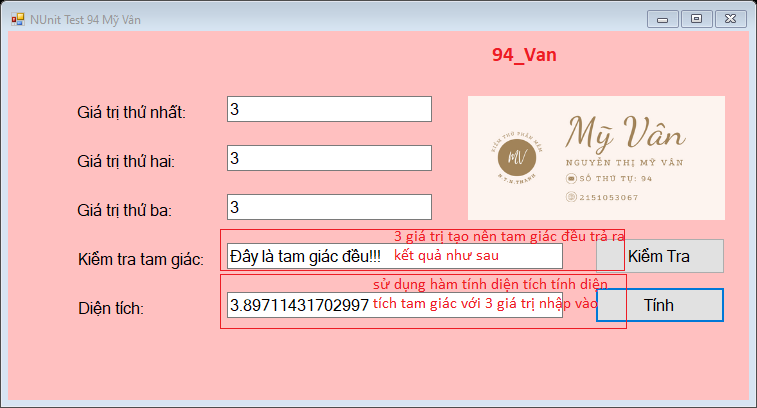
Hình 12: Chạy project: Đây là tam giác thường

* Kết quả là tam giác cân:



Hình 13: Chạy project: Đây là tam giác cân

* Kết quả là tam giác đều:



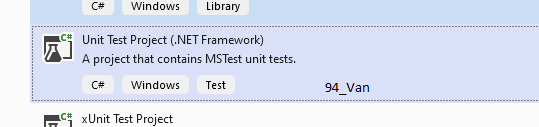
Hình 14: Chạy project: Đây là tam giác đều

## Kiểm thử Nunit:

### Tạo project, cài đặt References, tạo Datasource:

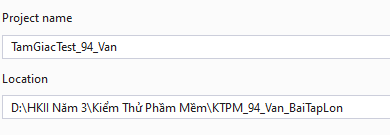
#### Tạo project kiểm thử:

* Tạo project trong solution project ‘KTPM\_94\_Van\_BaiTapLon’. Loại project là Unit Test Project(.NET Framework).



Hình 15: Tạo project Unit test

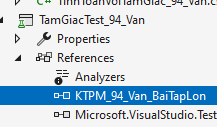
* Đặt tên là ‘TamGiacTest\_94\_Van’ với nơi lưu trữ nằm trong project.



Hình 16: Đặt tên, nơi lưu trữ cho project Nunit test

#### Cài đặt References:

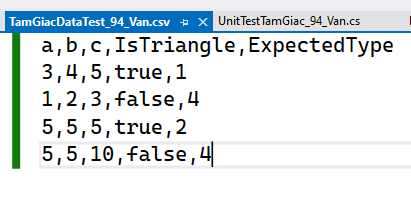
* Add reference là project tính toán ở trên:



Hình 17: Add Reference cho NUnit

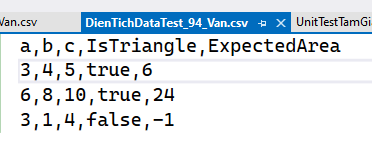
#### Tạo Datasource(Dữ liệu kiểm thử):

* Add reference là System.Data.
* Tạo một folder mới là ‘DataTest\_94\_Van’
* Tạo text file mới trong folder ‘DataTest\_94\_Van’ cho chức năng 1: Kiểm tra tam giác là ‘TamGiacDataTest\_94\_Van.csv’, cài đặt phần ‘Copy to Output Directory’ là ‘Copy always’. Nhập dữ liệu vào textfile.



Hình 18: DataSource cho chức năng kiểm tra tam giác

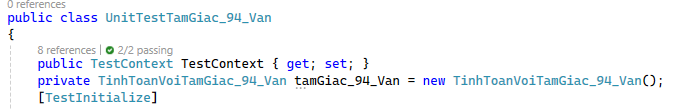
* Tạo Tạo text file mới trong folder ‘DataTest\_94\_Van’ cho chức năng 2: Tính diện tích tam giác là ‘TamGiacDataTest\_94\_Van.csv’, cài đặt phần ‘Copy to Output Directory’ là ‘Copy always’. Nhập dữ liệu vào textfile.



Hình 19: DataSource cho chức năng tính diện tích

### Mô tả TestCase:

* Đặt tên file Nunit Test là ‘UnitTestTamGiac\_94\_Van.cs’. Bên trong class tạo một đối tượng TestContext cho phép bạn truy cập các thông tin và thuộc tính liên quan đến quá trình thực thi của test. Tạo một đối tượng từ ‘TinhToanVoiTamGiac\_94\_Van’ để tính toán.
* Môi trường kiểm thử:
* Ngôn ngữ: C#
* Visual Studio 2022



Hình 20: Code tạo đối tượng test

#### Chức năng 1: Kiểm tra tam giác:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Test Case | Mô tả | Phương pháp | Tham số kiểm thử | | | Kết quả mong đợi | Kết quả thực tế | Kết quả |
| a | b | c |
| TC1 | Kiểm tra có phải tam giác hay không | * Lấy giá trị ba tham số, giá trị của expected trong datasource, sử dụng hàm ‘IsTriangle\_94\_Van’ để kiểm tra. Sử dụng phương thức của Assert là AreEqual để so sánh kết quả mong muốn với kết quả thực tế. * Tên hàm ‘TamGiacTest\_94\_Van()’ | 3 | 4 | 5 | true | true | Pass |
| TC2 | Kiểm tra tam giác đó là loại tam giác nào | * Lấy giá trị ba tham số, và giá trị mong đợi trong datasource. Định nghĩa giá trị ‘1’ là ‘ Đây là tam giác thường!!!’, ‘2’ là ‘ Đây là tam giác đều!!!’, ‘3’ là ‘ Đây là tam giác cân!!!’, ‘4’ là ‘ Đây không phải là tam giác!!!’. Sử dụng hàm ‘ DetermineTriangleType\_94\_Van’ để kiểm tra. Sử dụng phương thức của Assert là AreEqual để so sánh kết quả mong muốn với kết quả thực tế. * Tên hàm ‘LoaiTamGiacTest\_94\_Van()’ | 3 | 4 | 5 | 1 | Đây là tam giác thường!!! | Pass |

* Code:

[TestMethod]

[DataSource("Microsoft.VisualStudio.TestTools.DataSource.CSV",

@".\DataTest\_94\_Van\TamGiacDataTest\_94\_Van.csv", "TamGiacDataTest\_94\_Van#csv",

DataAccessMethod.Sequential)]

public void TamGiacTest\_94\_Van()

{

double a\_94\_Van = double.Parse(TestContext.DataRow[0].ToString());

double b\_94\_Van = double.Parse(TestContext.DataRow[1].ToString());

double c\_94\_Van = double.Parse(TestContext.DataRow[2].ToString());

bool isTriangle\_94\_Van = bool.Parse( TestContext.DataRow[3].ToString());

Assert.AreEqual(isTriangle\_94\_Van, tamGiac\_94\_Van.IsTriangle\_94\_Van(a\_94\_Van, b\_94\_Van, c\_94\_Van));

}

[TestMethod]

[DataSource("Microsoft.VisualStudio.TestTools.DataSource.CSV",

@".\DataTest\_94\_Van\TamGiacDataTest\_94\_Van.csv", "TamGiacDataTest\_94\_Van#csv",

DataAccessMethod.Sequential)]

public void LoaiTamGiacTest\_94\_Van()

{

double a\_94\_Van = double.Parse(TestContext.DataRow[0].ToString());

double b\_94\_Van = double.Parse(TestContext.DataRow[1].ToString());

double c\_94\_Van = double.Parse(TestContext.DataRow[2].ToString());

string expectedType\_94\_Van = TestContext.DataRow[4].ToString();

expectedType\_94\_Van = expectedType\_94\_Van == "1" ? "Đây là tam giác thường!!!" :

(expectedType\_94\_Van == "2" ? "Đây là tam giác đều!!!" :

(expectedType\_94\_Van == "3" ? "Đây là tam giác cân!!!" : "Đây không phải là tam giác!!!"));

Assert.AreEqual(expectedType\_94\_Van,

tamGiac\_94\_Van.DetermineTriangleType\_94\_Van(a\_94\_Van, b\_94\_Van, c\_94\_Van));

}

#### Chức năng 2: Tính diện tích tam giác:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Test Case | Mô tả | Phương pháp | Tham số kiểm thử | | | Kết quả mong đợi | Kết quả thực tế | Kết quả |
| a | b | c |
| TC1 | Tính diện tích tam giác | * Lấy giá trị ba tham số, giá trị của mong đợi trong datasource với số lớn hơn 0 thì lấy giá trị đó còn nếu giá trị nhỏ hơn 0 thì định nghĩa giá trị là ‘ Đây không phải là tam giác!!!’. Sử dụng hàm ‘CalculateArea\_94\_Van’ để kiểm tra. Sử dụng phương thức của Assert là AreEqual để so sánh kết quả mong muốn với kết quả thực tế. * Tên hàm ‘DienTichTest\_94\_Van()’ | 6 | 8 | 10 | 24 | 24 | Pass |
| 3 | 1 | 4 | -1 | Đây không phải là tam giác!!! | Pass |

* Code:

using Microsoft.VisualStudio.TestTools.UnitTesting;

using System;

using KTPM\_94\_Van\_BaiTapLon;

namespace TamGiacTest\_94\_Van

{

[TestClass]

public class UnitTestTamGiac\_94\_Van

{

public TestContext TestContext { get; set; }

private TinhToanVoiTamGiac\_94\_Van tamGiac\_94\_Van = new TinhToanVoiTamGiac\_94\_Van();

[TestInitialize]

[TestMethod]

[DataSource("Microsoft.VisualStudio.TestTools.DataSource.CSV",

@".\DataTest\_94\_Van\DienTichDataTest\_94\_Van.csv", "DienTichDataTest\_94\_Van#csv",

DataAccessMethod.Sequential)]

public void DienTichTest\_94\_Van()

{

double a\_94\_Van = double.Parse(TestContext.DataRow[0].ToString());

double b\_94\_Van = double.Parse(TestContext.DataRow[1].ToString());

double c\_94\_Van = double.Parse(TestContext.DataRow[2].ToString());

string expectedType\_94\_Van = double.Parse(TestContext.DataRow[4].ToString()) < 0 ? "Đây không phải là tam giác!!!" :

TestContext.DataRow[4].ToString();

Assert.AreEqual(expectedType\_94\_Van, tamGiac\_94\_Van.CalculateArea\_94\_Van(a\_94\_Van, b\_94\_Van, c\_94\_Van));

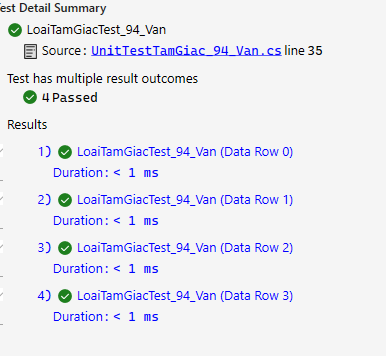
}

}

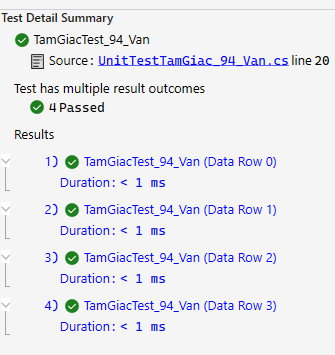
}

### Kết quả thu được:

#### Chức năng 1: Kiểm tra tam giác:

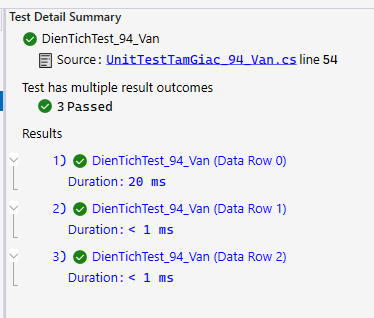


Hình 21: Kết quả Test của TestMethod kiểm tra loại tam giác



Hình 22: Kết quả Test của TestMethod Kiểm tra có phải là tam giác

#### Chức năng 2: Tính diện tích:



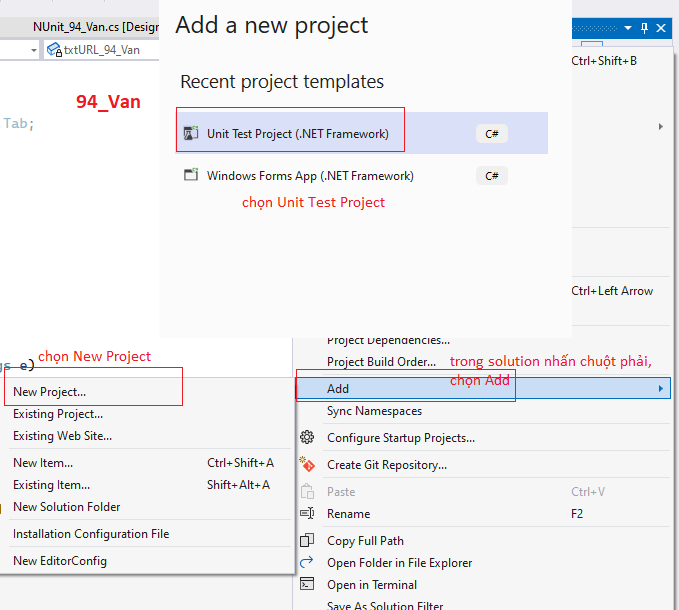
Hình 23: Kết quả Test của TestMethod tính diện tích

# Phần B. SELENIUM WEB DRIVER

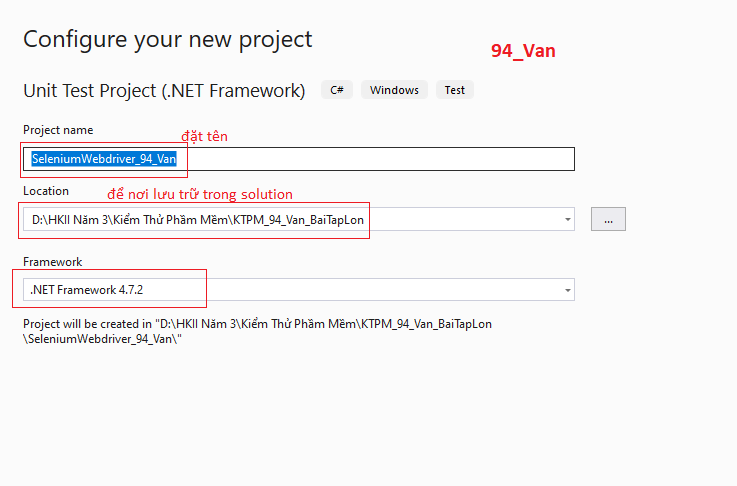
## Tạo project cho Test Selenium, cài đặt môi trường:

### Tạo project:

* Tạo một project mới để test Selenium Webdriver có tên ‘SeleniumWebdriver\_94\_Van’.

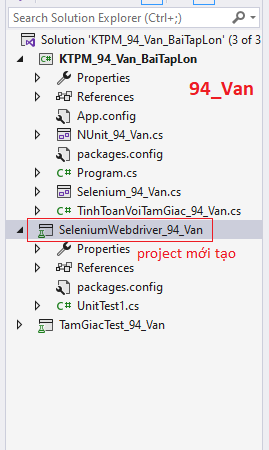


Hình 24: Tạo Project mới SeleniumWebdriver\_94\_Van



Hình 25: Đặt tên, nơi luu trữ cho Project Selenium

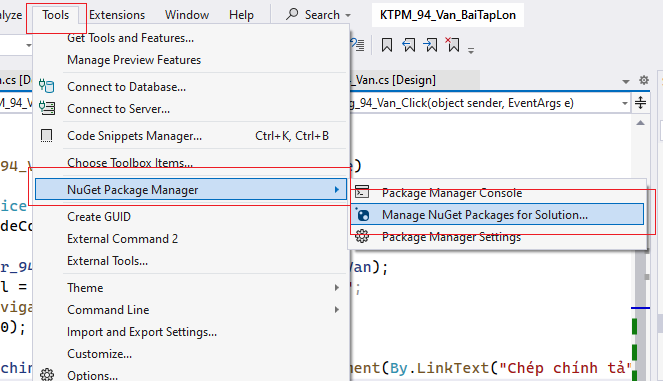
* Tiếp tục chọn create để hoàn tất tạo project.



Hình 26: Hoàn tất tạo Project SeleniumWebdriver

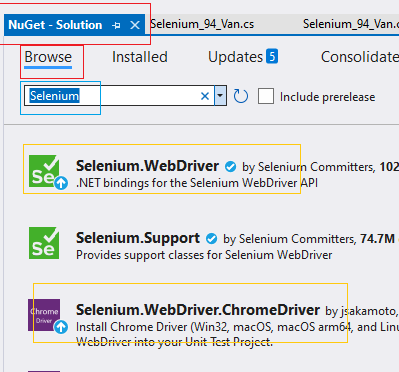
### Cài đặt môi trường:

* Chọn Tools => NuGet Package Manager => Manage Nuget Packages for Solution…



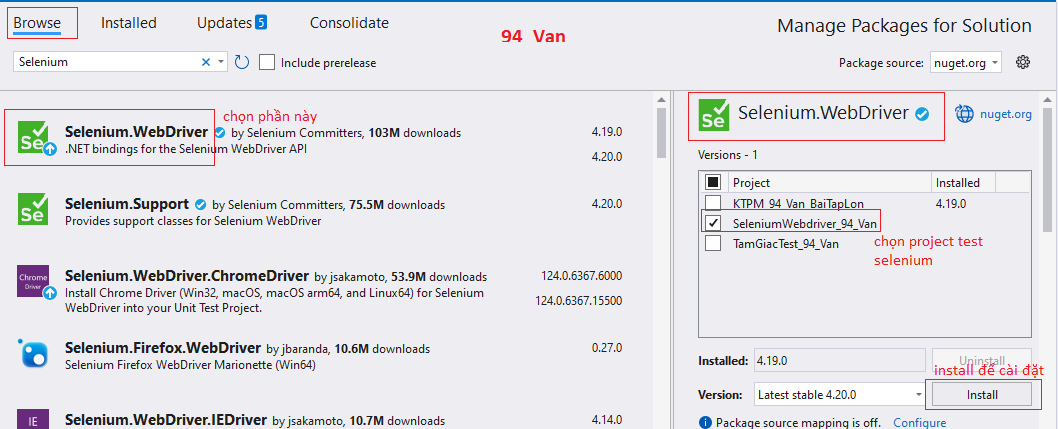
Hình 27: Dẫn đến NuGet Packages

* Tại Browse, tìm kiếm Selenium, sẽ xuất hiện các package có liên quan đến Selenium. Chọn Selenium.WebDriver và Selenium.WebDriver.ChromeDriver.



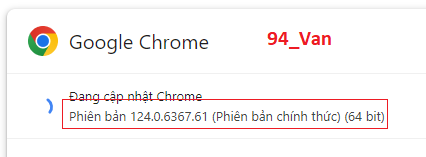
Hình 28: Các môi trường cần cài đặt

* Trong Selenium.WebDriver, chọn project cần thực hiện Selenium, install phiên bản thích hợp với thiết bị.

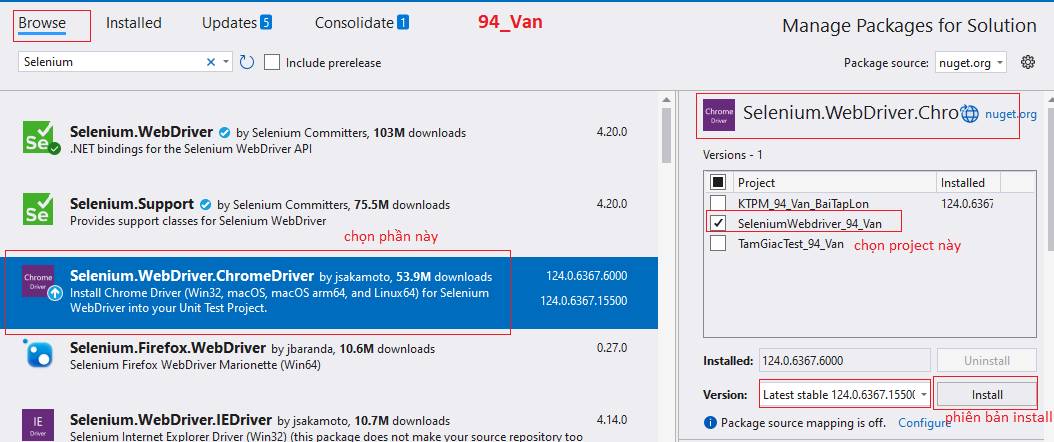


Hình 29: Cài đặt Selenium.WebDriver

* Trong Selenium.WebDriver.ChromeDriver, chọn project cần thực hiện Selenium, install phiên bản Chrome hiện tại thiết bị đang sử dụng.



Hình 30: Phiên bản Chrome trong thiết bị



Hình 31: Cài đặt Selenium.WebDriver.Chrome

### Môi trường cài đặt:

* Hệ điều hành: Window 10 Pro.
* Trình duyệt: Google Chrome version 124.0.6367.61
* Selenium: version 3.17.2.
* Ngôn ngữ lập trình: C#.
* Selenium Webdriver: C# version 4.19.0.

## Đặt tả chức năng:

### Chức năng 1: Tìm kiếm khóa học:

#### Mục tiêu:

* Kiểm tra tính năng "Tìm kiếm khóa học" trên trang web tuhocielts.dolenglish.vn/nghe-chep-chinh-ta-tieng-anh. Chức năng "Tìm kiếm khóa học" nhằm cung cấp cho người dùng tìm kiếm khóa học để hỗ trợ luyện tập và cải thiện kỹ năng ngữ pháp và chính tả trong tiếng Anh thông qua việc nghe và luyện viết.

#### Cách hoạt động của chức năng:

* Người dùng nhập từ khóa của khóa học cần luyện tập vào ô tìm kiếm.
* Sau đó, người dùng nhấn nút tìm kiếm để tìm các khóa học có liên quan.
* Sau khi tìm kiếm thành công, người dùng có thể chọn một khóa học bất kì và bắt đầu luyện tập.
* Người dùng có thể nhấn nút phát để nghe cách phát âm của từ hoặc đoạn văn đó.
* Người dùng sau đó cố gắng viết lại từ hoặc đoạn văn theo cách chính xác nhất có thể vào ô điền được cung cấp.
* Sau khi người dùng hoàn thành việc viết, họ có thể kiểm tra bài bằng cách xem đáp án.

### Chức năng 2: Học IELTS:

#### Mục tiêu:

* Kiểm tra tính năng bài kiểm tra IELTS Online trên trang web tuhocielts.dolenglish.vn. Cung cấp cho người học cơ hội thực hành và đánh giá kỹ năng ngôn ngữ tiếng Anh của họ, đặc biệt là trong khuôn khổ của kỳ thi IELTS (International English Language Testing System).

#### Cách hoạt động của chức năng:

* Người dùng truy cập vào trang web https://tuhocielts.dolenglish.vn/luyen-thi-ielts/free-ielts-online-test.
* Người dùng có thể chọn loại bài kiểm tra mong muốn, như "Reading", "Listening", "Writing" hoặc "Speaking".
* Sau khi chọn loại bài kiểm tra, người dùng sẽ được chuyển đến trang làm bài hoặc từ vựng hoặc đáp án tùy thích.
* Trang bài kiểm tra hay từ vựng sẽ cung cấp các câu hỏi, nhiệm vụ hoặc từ vựng liên quan đến kỹ năng đó.

Người dùng sẽ thực hiện bài kiểm tra theo hướng dẫn và chọn đáp án phù hợp.

## Bước thực hiện testcase của chức năng ‘Tìm kiếm khóa học’:

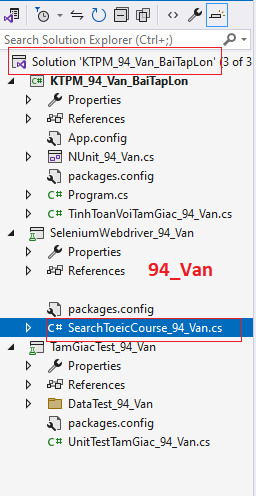
### Đặt tả testcase:

* Đặt tả các testcase của chức năng ‘Tìm kiếm khóa học’.

|  |  |  |  |
| --- | --- | --- | --- |
| TestCase | Mô tả | Phương pháp | Kết quả |
| TC1\_Search\_94\_Van | Nhập đúng khóa học | Nhập đúng khóa học vào SendKeys | Hoàn thành chép chính tả |
| TC2\_SearchError\_94\_Van | Nhập sai khóa học | Nhập sai khóa học vào SendKeys | Không tìm ra khóa học |

### Thực hiện test chức năng ‘Tìm kiếm khóa học’:

##### Đổi tên file class thành ‘SearchToeicCourse\_94\_Van.cs’ để thực hiện các testcase có liên quan



Hình 32: Đặt tên cho file test chức năng "Tìm kiếm khoa học"

#### Testcase 1: Tìm kiếm thành công, hoàn thành bài tập:

##### Bước 1: Vào trình duyệt

* Khởi tạo trình duyệt Chrome và điều hướng đến trang web ‘https://tuhocielts.dolenglish.vn/nghe-chep-chinh-ta-tieng-anh’.



Hình 33: Code điều hướng đến đường dẫn URL

* Code:

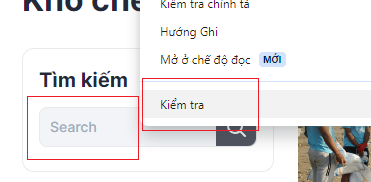
// Bước 1: Vào trình duyệt

driver\_94\_Van.Url = "https://tuhocielts.dolenglish.vn/nghe-chep-chinh-ta-tieng-anh";

driver\_94\_Van.Navigate();

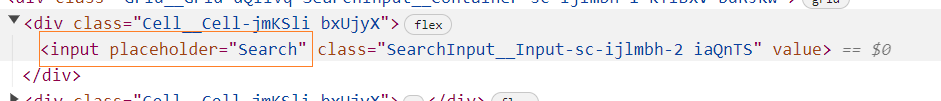
##### Bước 2: Nhập khóa học vào ô tìm kiếm

* Nhấn chuột phải, chọn Inspect(kiểm tra) để tìm kiếm element.



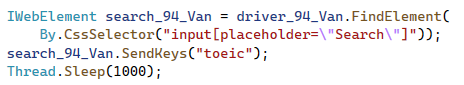
Hình 34:Inspect element tìm kiếm

* Trỏ đến element mong muốn, đối với ô nhập liệu tìm kiếm lấy element theo By.CssSelector. Copy phần [input placeholder="Search"]



Hình 35: Element tìm kiếm

* Nhập giá trị khóa học mong muốn vào element mới lấy bằng cách sử dụng SendKeys. Ví dụ, ta muốn tìm kiếm khóa học là toeic thì code sẽ là SendKeys(“toeic”).



Hình 36: Code phần tìm kiếm

* Code:

// Bước 2: Nhập khóa học vào ô tìm kiếm

IWebElement search\_94\_Van = driver\_94\_Van.FindElement(

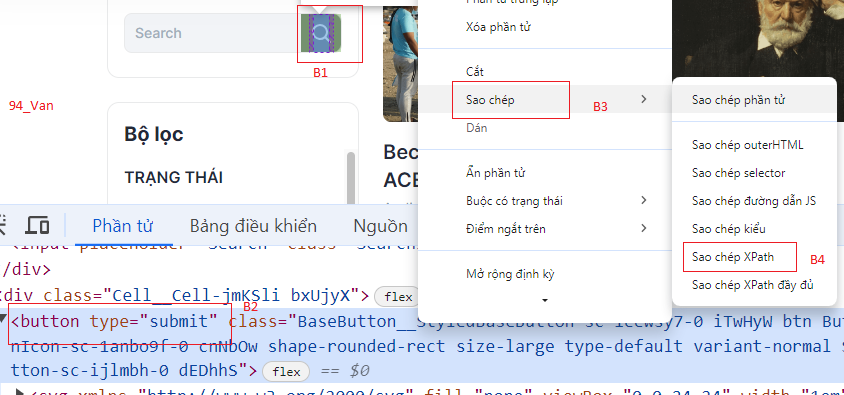
By.CssSelector("input[placeholder=\"Search\"]"));

search\_94\_Van.SendKeys("toiec");

Thread.Sleep(1000);

##### Bước 3: Nhấn tìm kiếm

* Tìm trong inspect element của nút tìm kiếm. Đối với nút tìm kiếm này lấy element theo By.Xpart. Các bước lấy Xpart theo thứ tự trong hình sau.



Hình 37: Inspect của nút tìm kiếm

* Tạo sự kiện cho element là Click().



Hình 38: Code phần nút tìm kiếm

* Code:

// Bước 3: Nhấn tìm kiếm

IWebElement submit\_94\_Van = driver\_94\_Van.FindElement(

By.XPath("//\*[@id=\"dictation-listing\"]/div[3]/div[1]/div/div/div[1]/div/form/div/div[3]/button"));

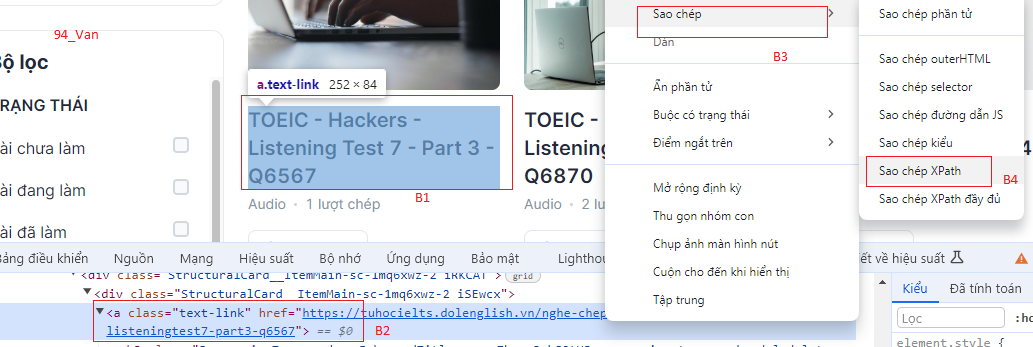
submit\_94\_Van.Click();

Thread.Sleep(1000);

* Sau khi sự kiện Click() tìm kiếm khóa học thành công, trình duyệt sẽ load các bài học theo giá trị đã nhập trong SendKeys.

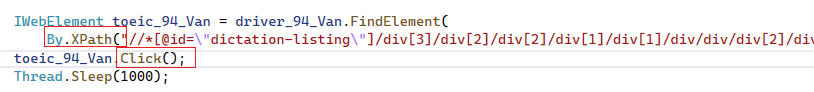
##### Bước 4: Chọn khóa học

* Chọn bài học đầu tiên bằng cách lấy Xpart của bài học đó trong Inspect(kiểm tra). Các bước lấy Xpart theo thứ tự trong hình sau.



Hình 39: Lấy element cho bài học

* Tạo sự kiện cho element này là Click()



Hình 40: Code phần chọn bài học

* Code

// Bước 4: Chọn khóa học

IWebElement toeic\_94\_Van = driver\_94\_Van.FindElement(

By.XPath("//\*[@id=\"dictation-listing\"]/div[3]/div[2]/div[2]/div[1]/div[1]/div/div/div[2]/div[1]/div[1]/a/h3"));

toeic\_94\_Van.Click();

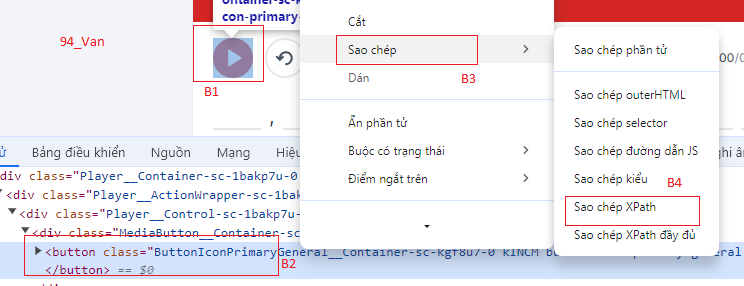
Thread.Sleep(1000);

##### Bước 5: Nghe scripts

* Nhấp chuột phải tại nút Play chọn Inspect(kiểm tra). Đối với nút Play chọn copy Xpart của element này. Các bước lấy Xpart theo thứ tự trong hình sau.

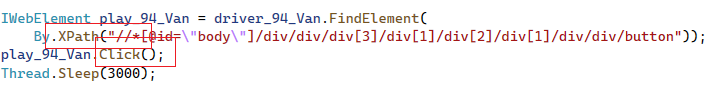


Hình 41: Chọn Inspect cho nút Play



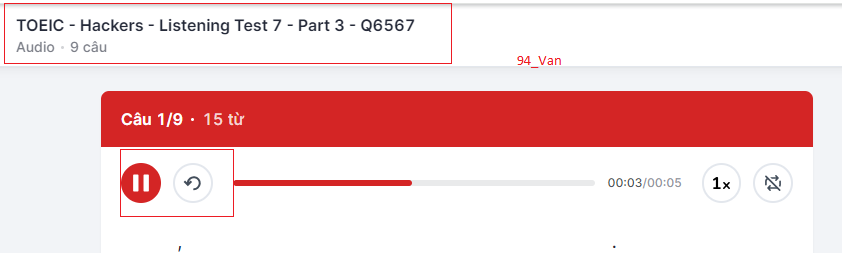
Hình 42: Lấy element cho nút Play

* Chọn sự kiện cho nút Play là Click().



Hình 43: Code cho phần nút Play

* Cài đặt thời gian 3000ms để đoạn script có thể chạy trong 3000ms.



Hình 44: Chạy kết quả cho phần Play

* Code

// Bước 5: Nghe scripts

IWebElement play\_94\_Van = driver\_94\_Van.FindElement(

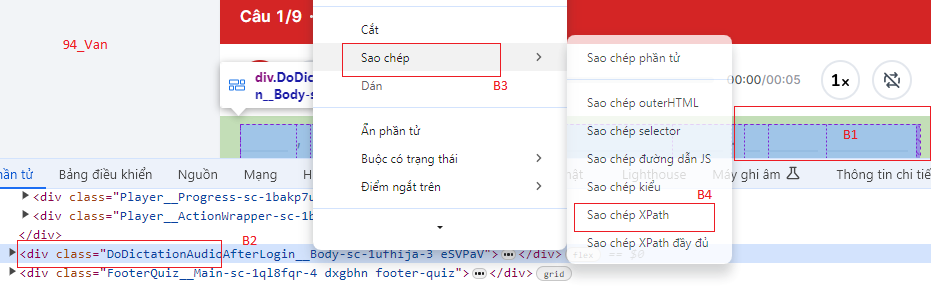
By.XPath("//\*[@id=\"body\"]/div/div/div[3]/div[1]/div[2]/div[1]/div/div/button"));

play\_94\_Van.Click();

Thread.Sleep(3000);

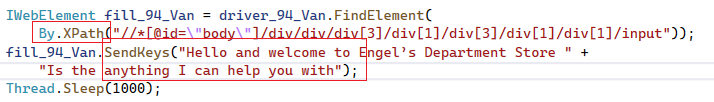
##### Bước 6: Nhập nội dung nghe được

* Nhấn chuột phải, chọn Inspect(Kiểm tra). Lấy Xpath của thẻ input để nhập kết quả nghe được. Các bước lấy Xpart theo thứ tự trong hình sau.



Hình 45: Lấy element cho ô nhập kết quả

* Nhập kết quả nghe vào element bằng cách sử dụng thông qua SendKeys. Ví dụ như giá trị nghe được là “Hello and welcome to Engel’s Department Store Is the anything I can help you with”.



Hình 46: Code phần nhập kết quả

* Code

// Bước 6: Nhập nội dung nghe được

IWebElement fill\_94\_Van = driver\_94\_Van.FindElement(

By.XPath("//\*[@id=\"body\"]/div/div/div[3]/div[1]/div[3]/div[1]/div[1]/input"));

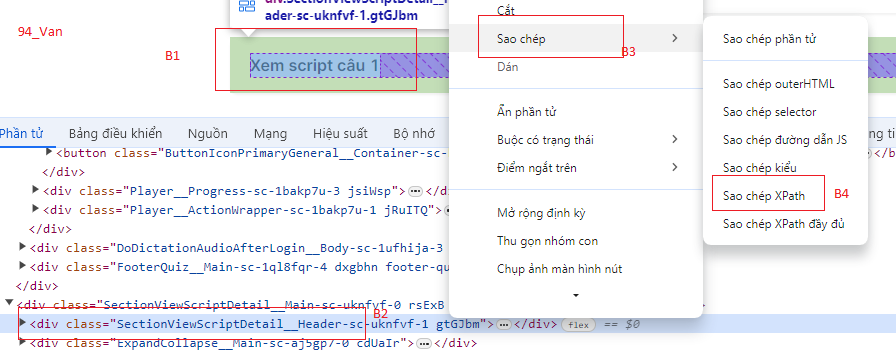
fill\_94\_Van.SendKeys("Hello and welcome to Engel’s Department Store " +

"Is the anything I can help you with");

Thread.Sleep(1000);

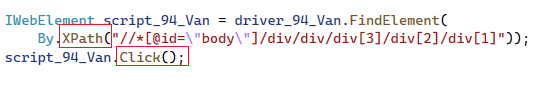
##### Bước 7: Kiểm tra đáp án

* Muốn kiểm tra kết quả, nhấn chuột phải vào ‘Xem script câu 1’. Copy Xpart của element đó. Các bước lấy Xpart theo thứ tự trong hình sau.



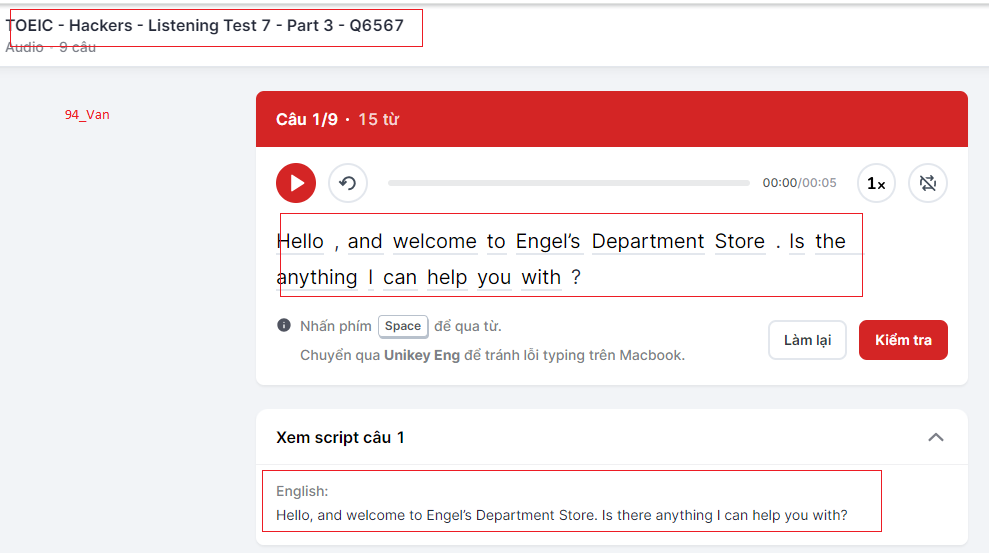
Hình 47:Lấy element cho Script kết quả

* Cài đặt element xem script kiểm tra kết quả là Click().



Hình 48: Code cho phần xem script xem kết quả

* Kết quả sau khi nhập giá trị và nhấn script để kiểm tra kết quả.



Hình 49: Chạy kết quả cho phần nhập đáp án và xem script kết quả

* Code

// Bước 7: Kiểm tra đáp án

IWebElement script\_94\_Van = driver\_94\_Van.FindElement(

By.XPath("//\*[@id=\"body\"]/div/div/div[3]/div[2]/div[1]"));

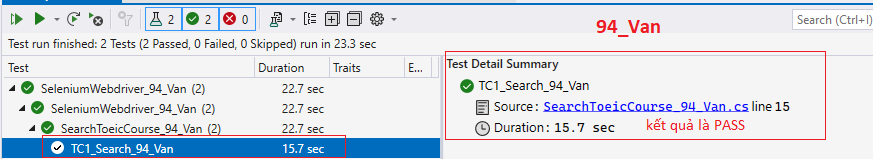
script\_94\_Van.Click();

##### Bước 8: Thoát trình duyệt

* Code

driver\_94\_Van.Quit();

##### **Tổng kết**: Kết quả testcase 1 Tìm kiếm thành công, hoàn thành bài tập



Hình 50: Kết quả testcase Tìm kiếm thành công

##### Code của cả hàm TC1\_Search\_94\_Van

//Bước 1: Vào trình duyệt

driver\_94\_Van.Url = "https://tuhocielts.dolenglish.vn/nghe-chep-chinh-ta-tieng-anh";

driver\_94\_Van.Navigate();

// Bước 2: Nhập khóa học vào ô tìm kiếm

IWebElement search\_94\_Van = driver\_94\_Van.FindElement(

By.CssSelector("input[placeholder=\"Search\"]"));

search\_94\_Van.SendKeys("toeic");

Thread.Sleep(1000);

// Bước 3: Nhấn tìm kiếm

IWebElement submit\_94\_Van = driver\_94\_Van.FindElement(

By.XPath("//\*[@id=\"dictation-listing\"]/div[3]/div[1]/div/div/div[1]/div/form/div/div[3]/button"));

submit\_94\_Van.Click();

Thread.Sleep(1000);

// Bước 4: Chọn khóa học

IWebElement toeic\_94\_Van = driver\_94\_Van.FindElement(

By.XPath("//\*[@id=\"dictation-listing\"]/div[3]/div[2]/div[2]/div[1]/div[1]/div/div/div[2]/div[1]/div[1]/a/h3"));

toeic\_94\_Van.Click();

Thread.Sleep(1000);

// Bước 5: Nghe scripts

IWebElement play\_94\_Van = driver\_94\_Van.FindElement(

By.XPath("//\*[@id=\"body\"]/div/div/div[3]/div[1]/div[2]/div[1]/div/div/button"));

play\_94\_Van.Click();

Thread.Sleep(3000);

// Bước 6: Nhập nội dung nghe được

IWebElement fill\_94\_Van = driver\_94\_Van.FindElement(

By.XPath("//\*[@id=\"body\"]/div/div/div[3]/div[1]/div[3]/div[1]/div[1]/input"));

fill\_94\_Van.SendKeys("Hello and welcome to Engel’s Department Store " +

"Is the anything I can help you with");

Thread.Sleep(1000);

// Bước 7: Kiểm tra đáp án

IWebElement script\_94\_Van = driver\_94\_Van.FindElement(

By.XPath("//\*[@id=\"body\"]/div/div/div[3]/div[2]/div[1]"));

script\_94\_Van.Click();

// Bước 8: Thoát trình duyệt

driver\_94\_Van.Quit();

#### Testcase 2: Nhập sai và không tìm kiếm ra kết quả khóa học:

##### Bước 1: Vào trình duyệt

* Khởi tạo trình duyệt Chrome và điều hướng đến trang web ‘https://tuhocielts.dolenglish.vn/nghe-chep-chinh-ta-tieng-anh’.



Hình 51: Code điều hướng đến đường dẫn URL

* Code:

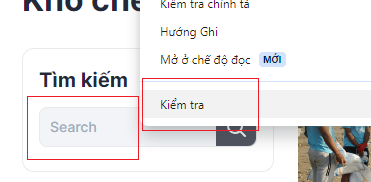
// Bước 1: Vào trình duyệt

driver\_94\_Van.Url = "https://tuhocielts.dolenglish.vn/nghe-chep-chinh-ta-tieng-anh";

driver\_94\_Van.Navigate();

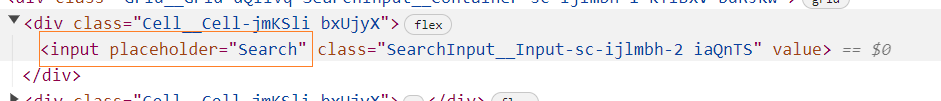
##### Bước 2: Nhập khóa học vào ô tìm kiếm

* Nhấn chuột phải, chọn Inspect(kiểm tra) để tìm kiếm element.



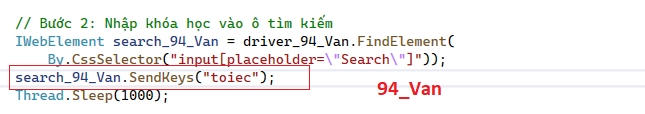
Hình 52:Inspect element tìm kiếm

* Trỏ đến element mong muốn, đối với ô nhập liệu tìm kiếm lấy element theo By.CssSelector. Copy phần [input placeholder="Search"].



Hình 53: Element tìm kiếm

* Nhập giá trị khóa học mong muốn vào element mới lấy bằng cách sử dụng SendKeys.
* Trong phần này, nhập một giá trị sai để test. Ví dụ nhập ‘toiec’ vào SendKeys.



Hình 54: Code phần tìm kiếm

* Code:

// Bước 2: Nhập khóa học vào ô tìm kiếm

IWebElement search\_94\_Van = driver\_94\_Van.FindElement(

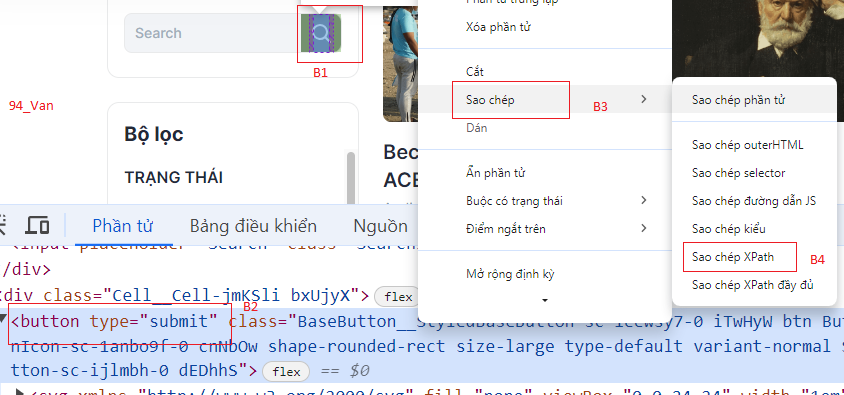
By.CssSelector("input[placeholder=\"Search\"]"));

search\_94\_Van.SendKeys("toiec");

Thread.Sleep(1000);

##### Bước 3: Nhấn tìm kiếm

* Tìm trong inspect element của nút tìm kiếm. Đối với nút tìm kiếm này lấy element theo By.Xpart. Các bước lấy Xpart theo thứ tự trong hình sau.



Hình 55: Inspect của nút tìm kiếm

* Tạo sự kiện cho element là Click().



Hình 56: Code phần nút tìm kiếm

* Code:

// Bước 3: Nhấn tìm kiếm

IWebElement submit\_94\_Van = driver\_94\_Van.FindElement(

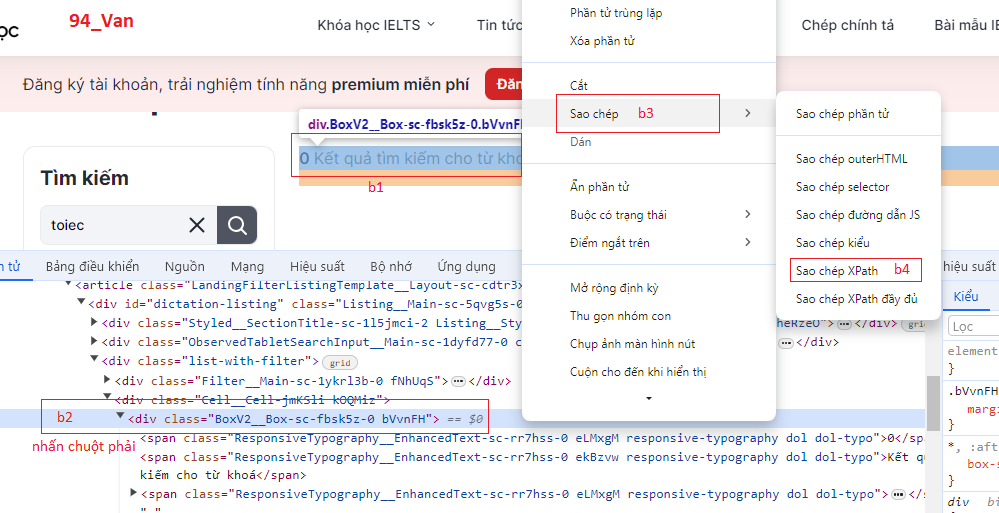
By.XPath("//\*[@id=\"dictation-listing\"]/div[3]/div[1]/div/div/div[1]/div/form/div/div[3]/button"));

submit\_94\_Van.Click();

Thread.Sleep(1000);

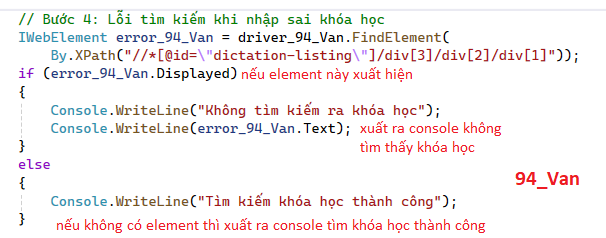
##### Bước 4: Lỗi tìm kiếm khi nhập sai khóa học

* Nhấn chuột phải chọn Inspect, tại dòng hiển thị ‘0 Kết quả tìm kiếm cho từ khoá “toiec”’, lấy element của đoạn ấy.



Hình 57: Lấy element phần không tìm ra khóa học

* Viết code xử lý bắt element tìm sai khóa học:



Hình 58: Code phần bắt Lỗi tìm kiếm khi nhập sai khóa học

* Code

// Bước 4: Lỗi tìm kiếm khi nhập sai khóa học

IWebElement error\_94\_Van = driver\_94\_Van.FindElement(

By.XPath("//\*[@id=\"dictation-listing\"]/div[3]/div[2]/div[1]"));

if (error\_94\_Van.Displayed)

{

Console.WriteLine("Không tìm kiếm ra khóa học");

Console.WriteLine(error\_94\_Van.Text);

}

else

{

Console.WriteLine("Tìm kiếm khóa học thành công");

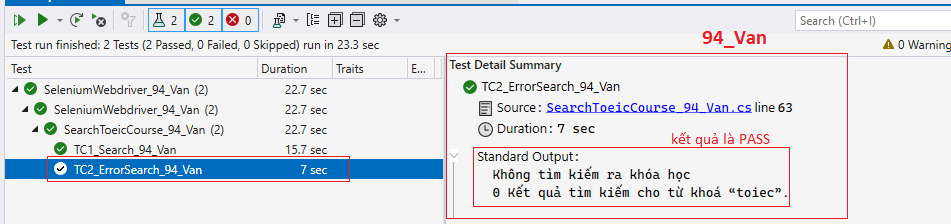
}

##### Bước 5: Thoát trình duyệt

* Code

driver\_94\_Van.Quit();

##### **Tổng kết**: Kết quả testcase 2 Nhập sai và không tìm kiếm ra kết quả khóa học



Hình 59: Kết quả testcase Nhập sai và không tìm kiếm ra kết quả khóa học

##### Code của cả hàm TC2\_ErrorSearch\_94\_Van

//Bước 1: Vào trình duyệt

driver\_94\_Van.Url = "https://tuhocielts.dolenglish.vn/nghe-chep-chinh-ta-tieng-anh";

driver\_94\_Van.Navigate();

// Bước 2: Nhập khóa học vào ô tìm kiếm

IWebElement search\_94\_Van = driver\_94\_Van.FindElement(

By.CssSelector("input[placeholder=\"Search\"]"));

search\_94\_Van.SendKeys("toiec");

Thread.Sleep(1000);

// Bước 3: Nhấn tìm kiếm

IWebElement submit\_94\_Van = driver\_94\_Van.FindElement(

By.XPath("//\*[@id=\"dictation-listing\"]/div[3]/div[1]/div/div/div[1]/div/form/div/div[3]/button"));

submit\_94\_Van.Click();

Thread.Sleep(1000);

// Bước 4: Lỗi tìm kiếm khi nhập sai khóa học

IWebElement error\_94\_Van = driver\_94\_Van.FindElement(

By.XPath("//\*[@id=\"dictation-listing\"]/div[3]/div[2]/div[1]"));

if (error\_94\_Van.Displayed)

{

Console.WriteLine("Không tìm kiếm ra khóa học");

Console.WriteLine(error\_94\_Van.Text);

}

else

{

Console.WriteLine("Tìm kiếm khóa học thành công");

}

// Bước : Thoát trình duyệt

driver\_94\_Van.Quit();

## Bước thực hiện testcase của chức năng ‘Học IELTS’:

### Đặt tả testcase:

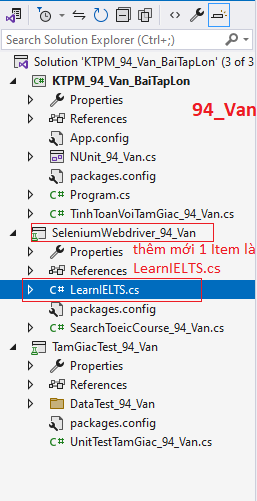
* Đặt tả các testcase của chức năng ‘Học IELTS’

|  |  |  |  |
| --- | --- | --- | --- |
| Testcase | Mô tả | Phương pháp | Kết quả |
| TC1\_Practise\_94\_Van | Làm bài | Chọn đúng vào nút làm bài | Bắt đầu làm bài IELTS |
| TC2\_Vocabulary\_94\_Van | Học từ vựng | Chọn vào phần từ vựng | Học từ vựng theo chủ đề |
| TC3\_Scripts\_94\_Van | Kiểm tra đáp án | Chọn vào phần đề và đáp án | Coi đề và đáp án tương ứng |

### Thực hiện test chức năng ‘Học IELTS’:

##### Tạo file class test chức năng Học IELTS

* Từ project ‘SeleniumWebdriver\_94\_Van’ thêm mới một Item. Đặt tên Item là ‘LearnIELTS.cs’ để lưu trữ các testcase của chức năng học IELTS.

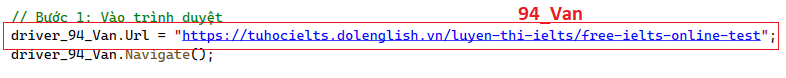


Hình 60: Đặt tên cho file test chức năng "Học IELTS"

#### Testcase 1: Làm bài:

##### Bước 1: Vào trình duyệt

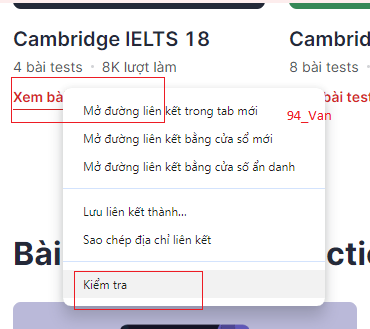
* Khởi tạo trình duyệt Chrome và điều hướng đến trang web ‘https://tuhocielts.dolenglish.vn/luyen-thi-ielts/free-ielts-online-test’.



Hình 61: Code điều hướng đến đường dẫn URL

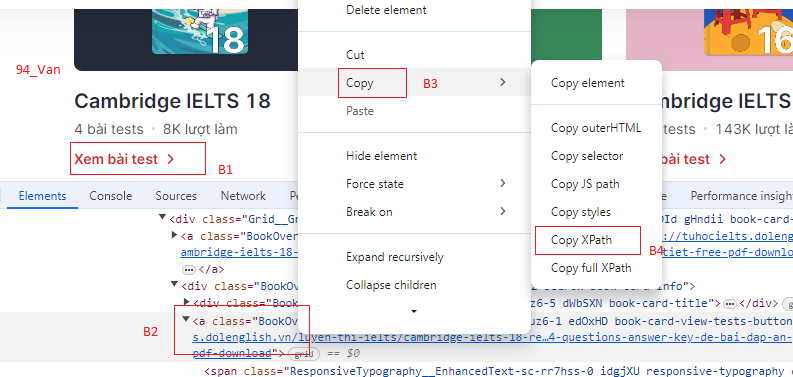
##### Bước 2: Chọn bài test

* Tới bài test muốn thực hiện, nhấn chuột phải, chọn Inspect(kiểm tra) để tìm kiếm element.



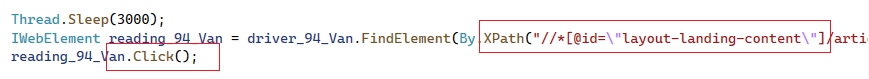
Hình 62: Inspect của 'Bài kiểm tra'

* Tìm trong inspect element của bài kiểm tra mong muốn. Đối với bài kiểm tra này lấy element theo By.Xpart. Các bước lấy Xpart theo thứ tự trong hình sau:



Hình 63: Element của 'Bài kiểm tra'

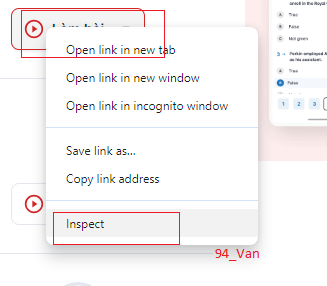
* Tạo sự kiện cho element là Click().



Hình 64: Code của phần chọn bài kiểm tra

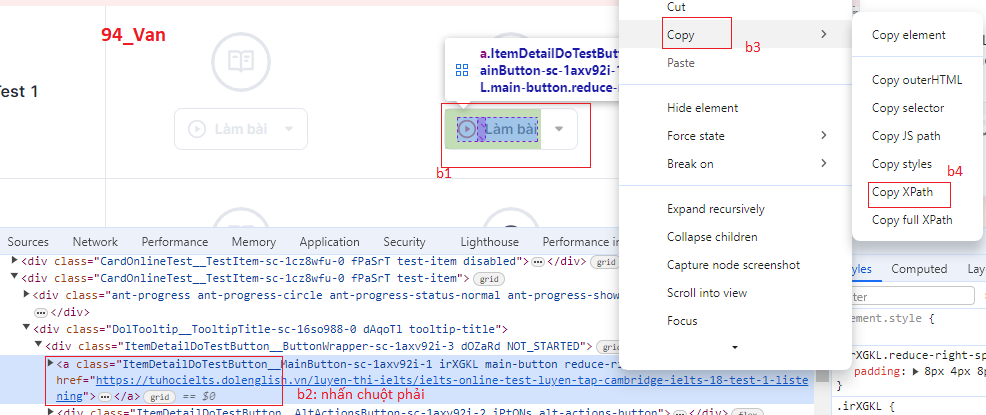
##### Bước 3: Chọn làm bài

* Nhấn chuột phải, chọn Inspect(kiểm tra) để tìm kiếm element.



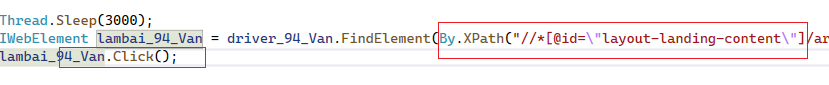
Hình 65: Inspect của 'Làm bài'

* Tìm trong inspect element của làm bài mong muốn. Đối với nút làm bài này lấy element theo By.Xpart. Các bước lấy Xpart theo thứ tự trong hình sau:



Hình 66: Elements của 'Làm bài'

* Chọn sự kiện cho nút Làm bài là Click().



Hình 67: Code phần Làm bài

##### Bước 4: Thoát trình duyệt

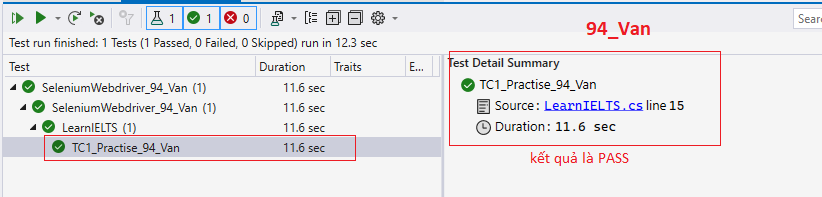
* Code

// Bước 4: Đóng trình duyệt

Thread.Sleep(3000);

driver\_94\_Van.Quit();

##### **Tổng kết**: Kết quả testcase 1 Làm bài



Hình 68: Kết quả testcase 1 Làm bài

##### Code của cả hàm TC1\_Practise\_94\_Van

// Bước 1: Vào trình duyệt

driver\_94\_Van.Url = "https://tuhocielts.dolenglish.vn/luyen-thi-ielts/free-ielts-online-test";

driver\_94\_Van.Navigate();

// Bước 2: Chọn bài test

Thread.Sleep(3000);

IWebElement reading\_94\_Van = driver\_94\_Van.FindElement(By.XPath("//\*[@id=\"layout-landing-content\"]/article/div[3]/div/div[2]/article[1]/div/section/div[3]/div[1]/div/div/a"));

reading\_94\_Van.Click();

// Bước 3: Chọn làm bài

Thread.Sleep(3000);

IWebElement lambai\_94\_Van = driver\_94\_Van.FindElement(By.XPath("//\*[@id=\"layout-landing-content\"]/article/div[4]/div/div[2]/div[1]/div/div[2]/div/div[2]/div[1]/div[3]/div[2]/div/a"));

lambai\_94\_Van.Click();

// Bước 4: Đóng trình duyệt

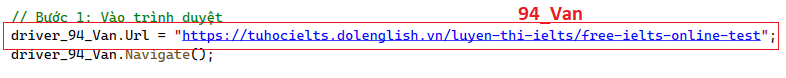
Thread.Sleep(3000);

driver\_94\_Van.Quit();

#### Testcase 2: Học từ vựng:

##### Bước 1: Vào trình duyệt:

* Khởi tạo trình duyệt Chrome và điều hướng đến trang web ‘https://tuhocielts.dolenglish.vn/luyen-thi-ielts/free-ielts-online-test’.



Hình 69: Code điều hướng đến đường dẫn URL

* Code

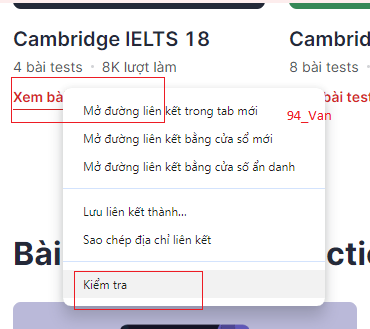
// Bước 1: Vào trình duyệt

driver\_94\_Van.Url = "https://tuhocielts.dolenglish.vn/luyen-thi-ielts/free-ielts-online-test";

driver\_94\_Van.Navigate();

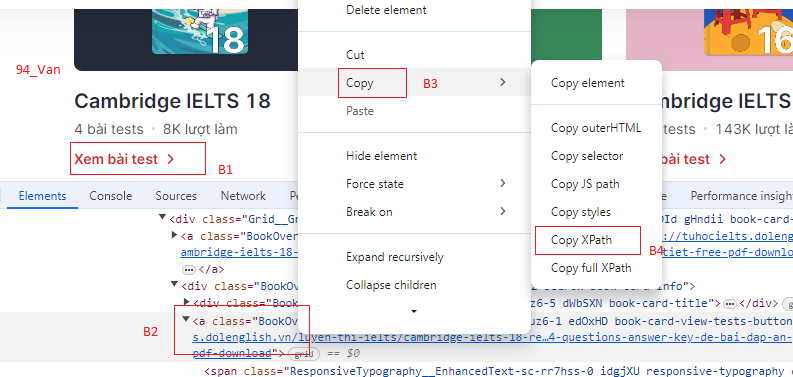
##### Bước 2: Chọn bài test

* Tới bài test muốn thực hiện, nhấn chuột phải, chọn Inspect(kiểm tra) để tìm kiếm element.



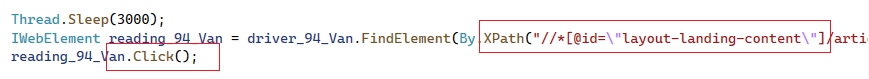
Hình 70: Inspect của 'Bài kiểm tra'

* Tìm trong inspect element của bài kiểm tra mong muốn. Đối với bài kiểm tra này lấy element theo By.Xpart. Các bước lấy Xpart theo thứ tự trong hình sau:



Hình 71: Element của 'Bài kiểm tra'

* Tạo sự kiện cho element là Click().



Hình 72: Code của phần chọn bài kiểm tra

* Code

// Bước 2: Chọn bài test

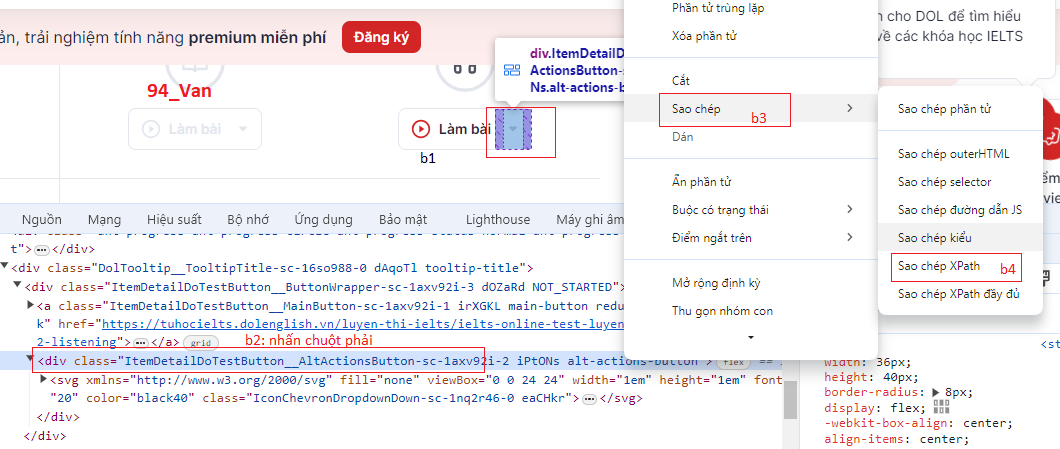
Thread.Sleep(3000);

IWebElement reading\_94\_Van = driver\_94\_Van.FindElement(By.XPath("//\*[@id=\"layout-landing-content\"]/article/div[3]/div/div[2]/article[1]/div/section/div[3]/div[1]/div/div/a"));

reading\_94\_Van.Click();

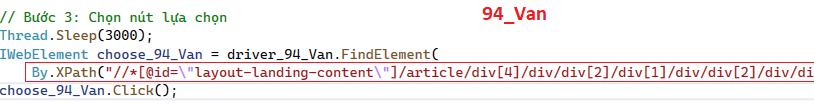
##### Bước 3: Chọn nút lựa chọn

* Lấy element của nút lựa chọn. Đối với nút lựa chọn này lấy element theo By.Xpart. Các bước lấy Xpart theo thứ tự trong hình sau:



Hình 73: Element của nút lựa chọn

* Tạo sự kiện cho nút lựa chọn là Click()



Hình 74: Code của nút lựu chọn

* Code

// Bước 3: Chọn nút lựa chọn

Thread.Sleep(3000);

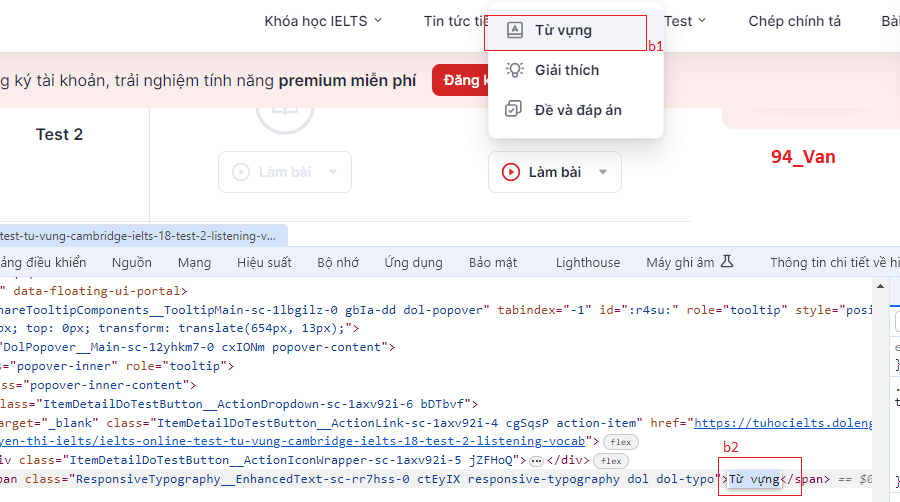
IWebElement choose\_94\_Van = driver\_94\_Van.FindElement(

By.XPath("//\*[@id=\"layout-landing-content\"]/article/div[4]/div/div[2]/div[1]/div/div[2]/div/div[2]/div[2]/div[3]/div[2]/div/div"));

choose\_94\_Van.Click();

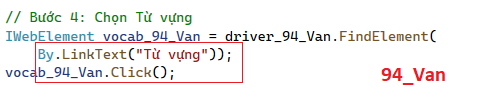
##### Bước 4: Chọn từ vựng

* Lấy element của Từ vựng. Đối với từ vựng này lấy element theo By.LinkTest.



Hình 75: Element của Từ vựng

* Chọn sự kiện cho nút Từ vựng này là Click()



Hình 76: Code của Từ vựng

* Code

// Bước 4: Chọn Từ vựng

IWebElement vocab\_94\_Van = driver\_94\_Van.FindElement(

By.LinkText("Từ vựng"));

vocab\_94\_Van.Click();

##### Bước 5: Thoát trình duyệt

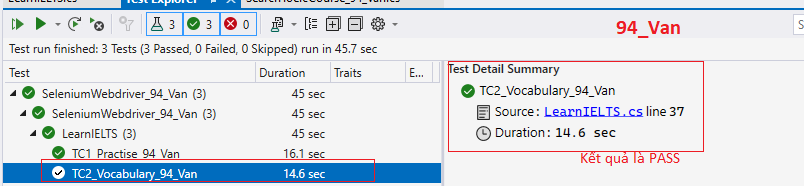
* Code

// Bước 5: Đóng trình duyệt

Thread.Sleep(3000);

driver\_94\_Van.Quit();

##### **Tổng kết**: Kết quả testcase 2 Học từ vựng



Hình 77: Kết quả testcase 2 Học từ vựng

##### Code của cả hàm TC2\_Vocabulary\_94\_Van

// Bước 1: Vào trình duyệt

driver\_94\_Van.Url = "https://tuhocielts.dolenglish.vn/luyen-thi-ielts/free-ielts-online-test";

driver\_94\_Van.Navigate();

// Bước 2: Chọn bài test

Thread.Sleep(3000);

IWebElement reading\_94\_Van = driver\_94\_Van.FindElement(By.XPath("//\*[@id=\"layout-landing-content\"]/article/div[3]/div/div[2]/article[1]/div/section/div[3]/div[1]/div/div/a"));

reading\_94\_Van.Click();

// Bước 3: Chọn nút lựa chọn

Thread.Sleep(3000);

IWebElement choose\_94\_Van = driver\_94\_Van.FindElement(

By.XPath("//\*[@id=\"layout-landing-content\"]/article/div[4]/div/div[2]/div[1]/div/div[2]/div/div[2]/div[2]/div[3]/div[2]/div/div"));

choose\_94\_Van.Click();

// Bước 4: Chọn Từ vựng

IWebElement vocab\_94\_Van = driver\_94\_Van.FindElement(

By.LinkText("Từ vựng"));

vocab\_94\_Van.Click();

// Bước 5: Đóng trình duyệt

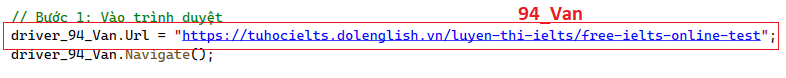
Thread.Sleep(3000);

driver\_94\_Van.Quit();

#### Testcase 3: Kiểm tra đề và đáp án:

##### Bước 1: Vào trình duyệt:

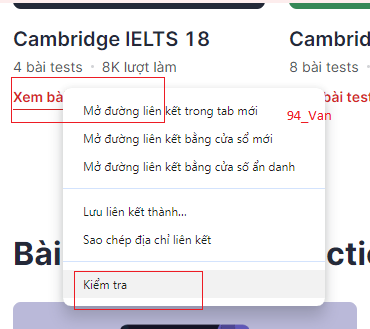
* Khởi tạo trình duyệt Chrome và điều hướng đến trang web ‘https://tuhocielts.dolenglish.vn/luyen-thi-ielts/free-ielts-online-test’.



Hình 78: Code điều hướng đến đường dẫn URL

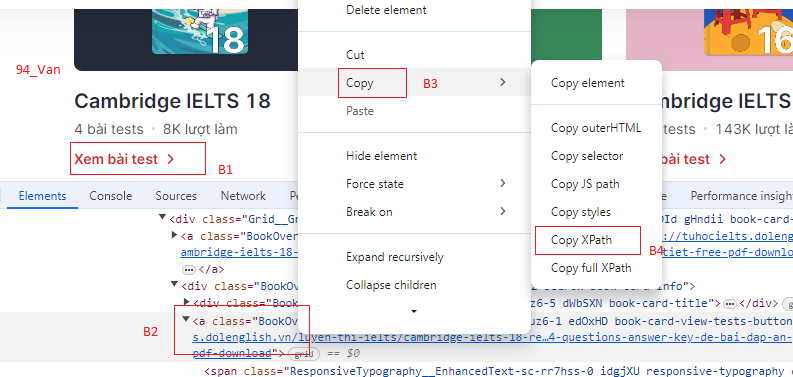
##### Bước 2: Chọn bài test

* Tới bài test muốn thực hiện, nhấn chuột phải, chọn Inspect(kiểm tra) để tìm kiếm element.



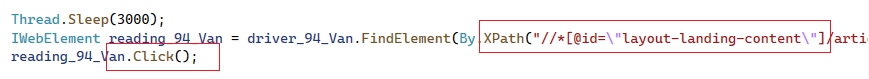
Hình 79: Inspect của 'Bài kiểm tra'

* Tìm trong inspect element của bài kiểm tra mong muốn. Đối với bài kiểm tra này lấy element theo By.Xpart. Các bước lấy Xpart theo thứ tự trong hình sau:



Hình 80: Element của 'Bài kiểm tra'

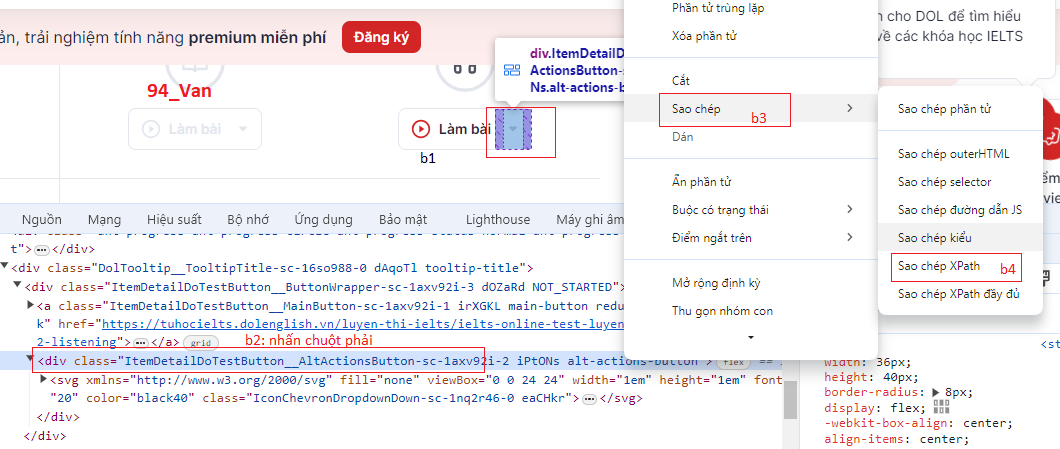
* Tạo sự kiện cho element là Click().



Hình 81: Code của phần chọn bài kiểm tra

##### Bước 3: Chọn nút lựa chọn

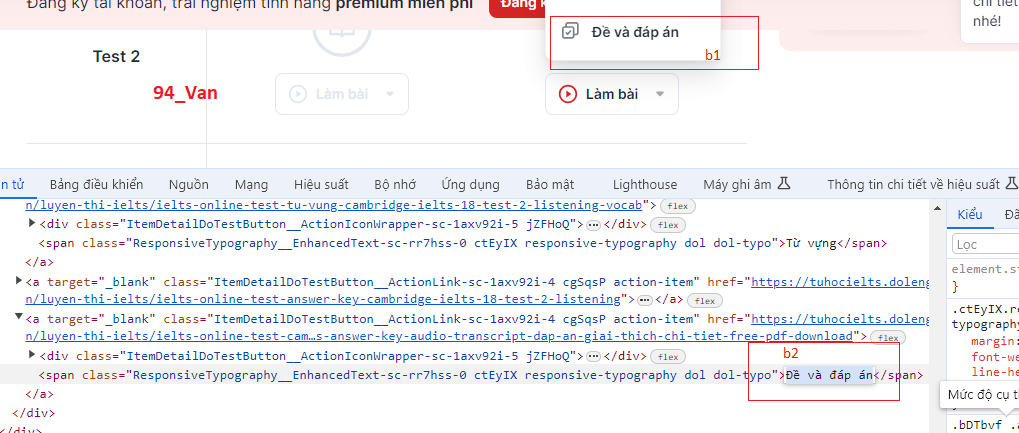
* Lấy element của nút lựa chọn. Đối với nút lựa chọn này lấy element theo By.Xpart. Các bước lấy Xpart theo thứ tự trong hình sau:



Hình 82: Element của nút lựa chọn

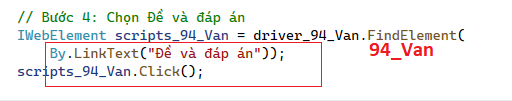
##### Bước 4: Chọn đề và đáp án

* Lấy element của Từ vựng. Đối với từ vựng này lấy element theo By.LinkTest.



Hình 83: Element của Đề và làm bài

* Chọn sự kiện cho nút Từ vựng này là Click()



Hình 84: Code của Đề và đáp án

* Code

// Bước 4: Chọn Đề và đáp án

IWebElement scripts\_94\_Van = driver\_94\_Van.FindElement(

By.LinkText("Đề và đáp án"));

scripts\_94\_Van.Click();

##### Bước 5: Thoát trình duyệt

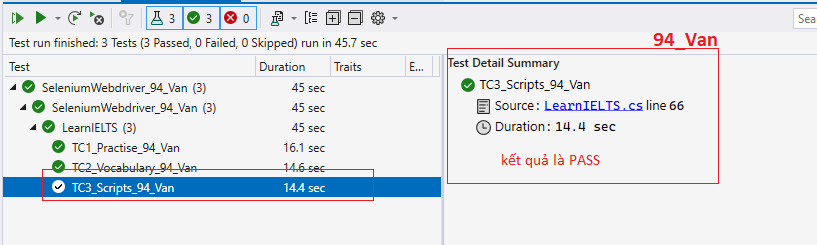
* Code

// Bước 5: Đóng trình duyệt

Thread.Sleep(3000);

driver\_94\_Van.Quit();

##### **Tổng kết**: Kết quả testcase 2 Học từ vựng



Hình 85: Kết quả của testcase 3 Đề và đáp án

##### Code của cả hàm TC3\_Scripts\_94\_Van

// Bước 1: Vào trình duyệt

driver\_94\_Van.Url = "https://tuhocielts.dolenglish.vn/luyen-thi-ielts/free-ielts-online-test";

driver\_94\_Van.Navigate();

// Bước 2: Chọn bài test

Thread.Sleep(3000);

IWebElement reading\_94\_Van = driver\_94\_Van.FindElement(By.XPath("//\*[@id=\"layout-landing-content\"]/article/div[3]/div/div[2]/article[1]/div/section/div[3]/div[1]/div/div/a"));

reading\_94\_Van.Click();

// Bước 3: Chọn nút lựa chọn

Thread.Sleep(3000);

IWebElement choose\_94\_Van = driver\_94\_Van.FindElement(

By.XPath("//\*[@id=\"layout-landing-content\"]/article/div[4]/div/div[2]/div[1]/div/div[2]/div/div[2]/div[2]/div[3]/div[2]/div/div"));

choose\_94\_Van.Click();

// Bước 4: Chọn Đề và đáp án

IWebElement scripts\_94\_Van = driver\_94\_Van.FindElement(

By.LinkText("Đề và đáp án"));

scripts\_94\_Van.Click();

// Bước 5: Đóng trình duyệt

Thread.Sleep(3000);

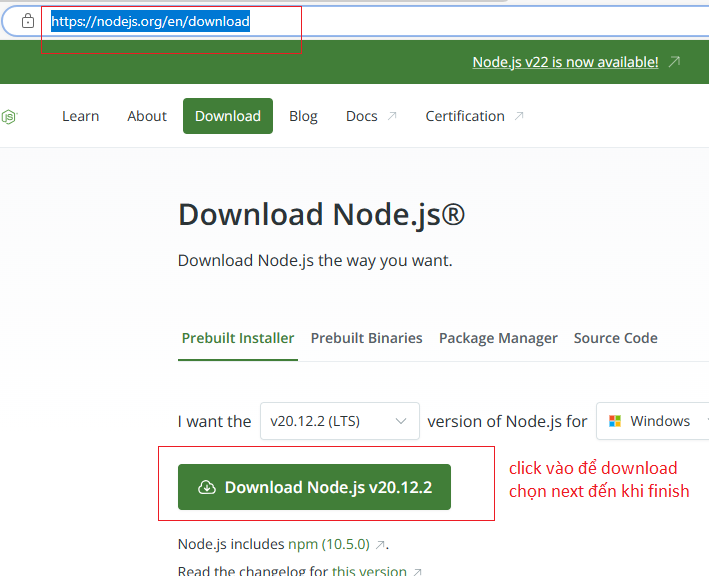
driver\_94\_Van.Quit();

# Phần C: API POSTMAN

## Tạo JSON Server:

### Cài đặt NodeJS:

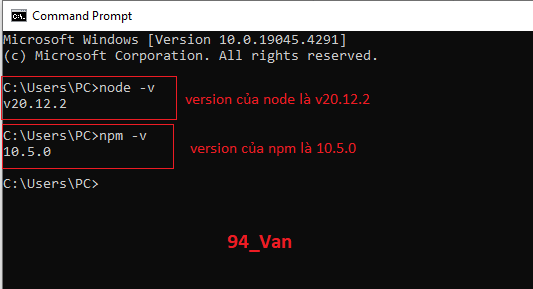
* Cài đặt, download nodejs trên google bằng liên kết <https://nodejs.org/en/download>



Hình 86: Download Nodejs

#### Môi trường cài đặt:

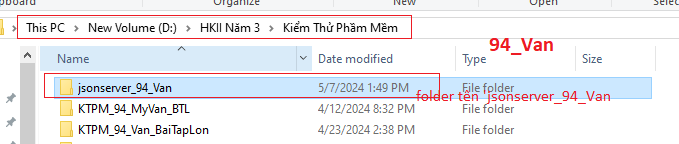
* Vào cmd trên máy tính, kiểm tra version của Nodejs và npm. Nhập node –v để kiểm tra version của Nodejs, nhập npm –v để kiểm tra version của npm.
* Phiên bản của Nodejs: v20.12.2
* Phiên bản của NPM: 10.5.0



Hình 87: Phiên bản của Nodejs và npm

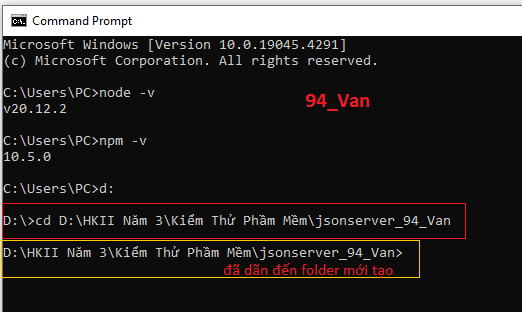
### Cài đặt gói sử dụng Node package manager(npm):

* Tạo một folder để lưu trữ package.json. Tên folder được đặt tên ‘jsonserver\_94\_Van’.



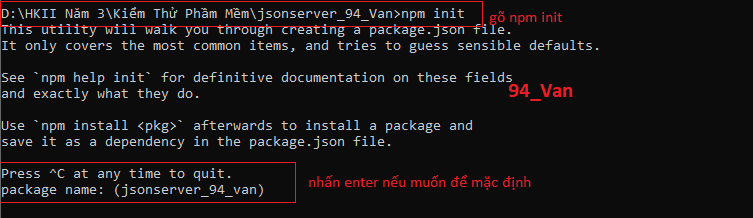
Hình 88: Tạo folder lưu package.json

* Trong cmd trỏ đến folder mới được tạo.



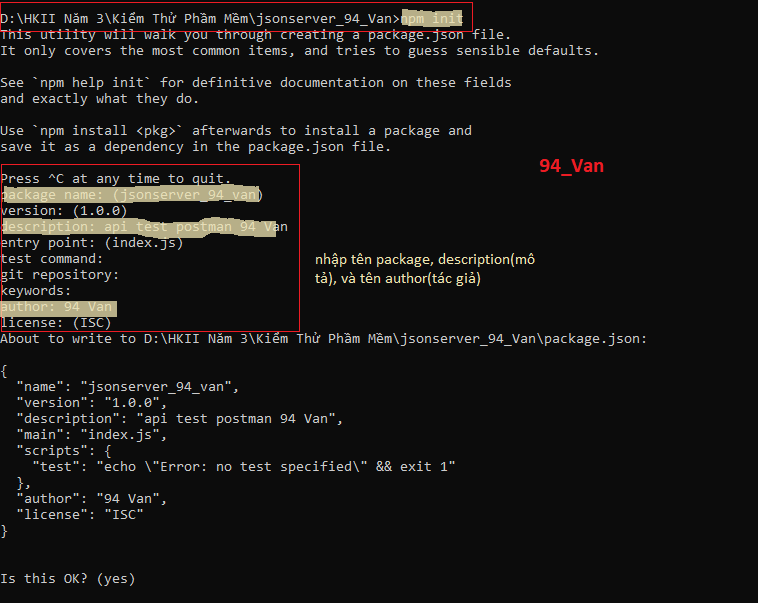
Hình 89: Trỏ đường dẫn đến folder mới tạo

* Bây giờ ta đang ở trong thư mục mới của mình, ta có thể khởi tạo dự án bằng lệnh init. Sau đó nhập dữ liệu tương với các dòng thông báo rồi nhấn Enter. Trường hợp bạn muốn để tên mặc định thì chỉ cần nhấn Enter là được.



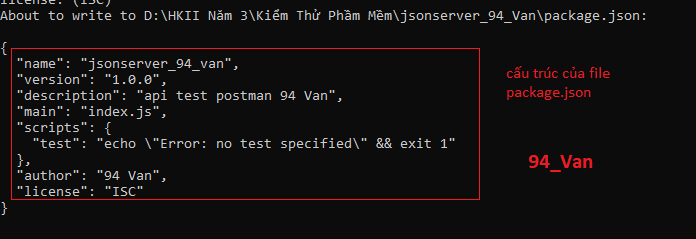
Hình 90: Nhập npm init để tạo json

* Trong đó:
* name: Tên của Project
* version: Version của Project
* description: Mô tả cho Project
* main: File chạy chính (chạy đầu tiên) của Project
* scripts: Danh sách các khai báo cấu hình bổ sung cho npm. Như trong chuỗi trên thì giá trị của test chính là câu thông báo và dừng chương trình khi bị lỗi.
* author: Tên tác giả của Project
* license: License của Project, giá trị mặc định là ISC.



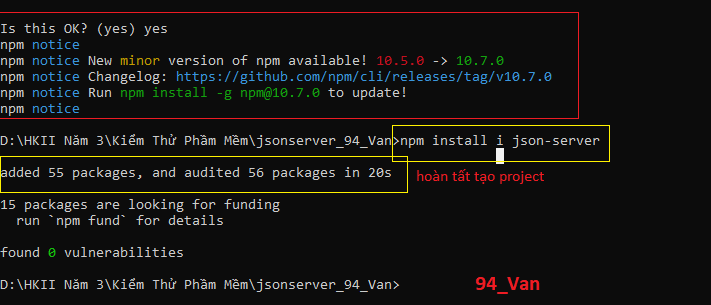
Hình 91: Nhập các thông tin mong muốn

* Một khi thiết lập đã hoàn tất, sẽ có một tập tin package.json có cấu trúc trông như thế này:



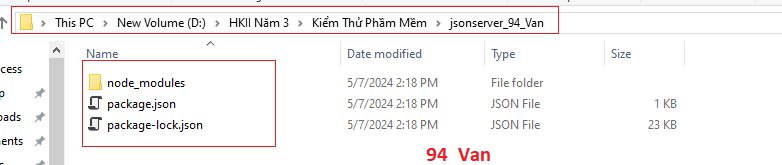
Hình 92: Cấu trúc file package.json

* Nhập yes để hoàn tất tạo project. Sau đó nhập npm install i json-server. Lệnh trên sẽ cài đặt json-server trên hệ thống. Sau khi cài đặt xong, ta có thể sử dụng json-server để tạo một REST API dựa trên tệp db.json.



Hình 93: Hoàn tất tạo project package.json

* Vào thư mục ‘jsonserver\_94\_Van’ để kiểm tra kết quả.

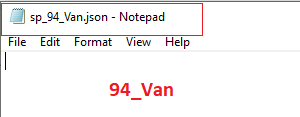


Hình 94: Kiểm tra folder 'jsonserver\_94\_Van'

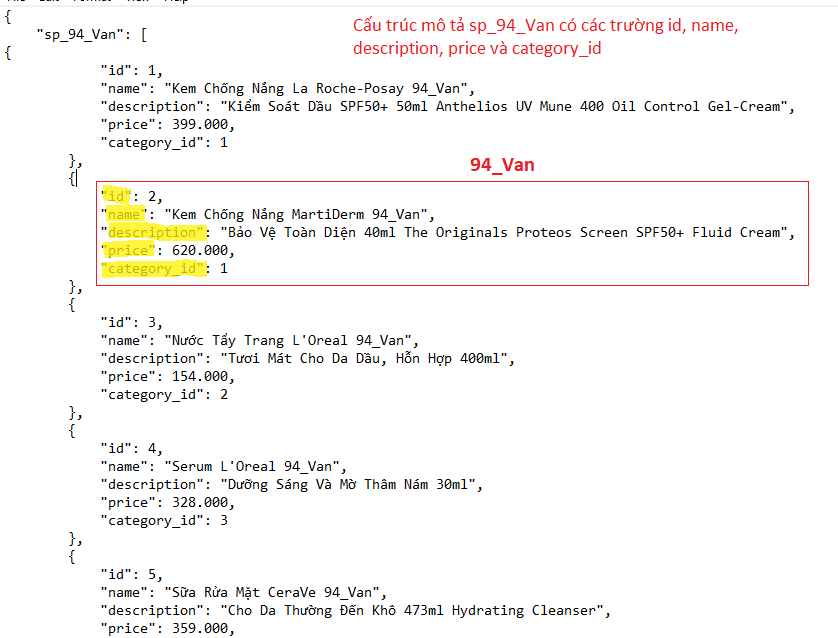
### Tạo file API trên server ảo:

#### Tạo file sp\_94\_Van.json:

* Trong Notepad tạo file sp\_94\_Van.json. Trong file này sẽ chứa những dữ liệu được trả về bởi REST API.



Hình 95: Tạo file 'sp\_94\_Van' trên notepad



Hình 96: Dữ liệu của sp\_94\_Van.json

* Cấu trúc trên mô tả sp\_94\_Van object với các trường id, name, desciption, price và category\_id.
* Dữ liệu

{

"sp\_94\_Van": [

{

"id": 1,

"name": "Kem Chống Nắng La Roche-Posay 94\_Van",

"description": "Kiểm Soát Dầu SPF50+ 50ml Anthelios UV Mune 400 Oil Control Gel-Cream",

"price": 399.000,

"category\_id": 1

},

{

"id": 2,

"name": "Kem Chống Nắng MartiDerm 94\_Van",

"description": "Bảo Vệ Toàn Diện 40ml The Originals Proteos Screen SPF50+ Fluid Cream",

"price": 620.000,

"category\_id": 1

},

{

"id": 3,

"name": "Nước Tẩy Trang L'Oreal 94\_Van",

"description": "Tươi Mát Cho Da Dầu, Hỗn Hợp 400ml",

"price": 154.000,

"category\_id": 2

},

{

"id": 4,

"name": "Serum L'Oreal 94\_Van",

"description": "Dưỡng Sáng Và Mờ Thâm Nám 30ml",

"price": 328.000,

"category\_id": 3

},

{

"id": 5,

"name": "Sữa Rửa Mặt CeraVe 94\_Van",

"description": "Cho Da Thường Đến Khô 473ml Hydrating Cleanser",

"price": 359.000,

"category\_id": 4

},

{

"id": 6,

"name": "Nước Tẩy Trang Garnier 94\_Van",

"description": "Dành Cho Da Dầu Và Mụn 400ml",

"price": 137.000,

"category\_id": 2

},

{

"id": 7,

"name": "Serum Garnier 94\_Van",

"description": "Giảm Mụn Mờ Thâm Cho Da Dầu, Mụn 30ml",

"price": 250000,

"category\_id": 3

},

{

"id": 8,

"name": "Kem Chống Nắng Skin1004 94\_Van",

"description": "Cho Da Nhạy Cảm SPF 50+ 50ml",

"price": 246.000,

"category\_id": 1

},

{

"id": 9,

"name": "Nước Tẩy Trang La Roche-Posay 94\_Van",

"description": "Dành Cho Da Nhạy Cảm 400ml",

"price": 374.000,

"category\_id": 2

},

{

"id": 10,

"name": "Serum GoodnDoc 94\_Van",

"description": "Dưỡng Ẩm, Hỗ Trợ Phục Hồi Da 30ml Hydra B5 Serum",

"price": 350000,

"category\_id": 3

},

{

"id": 11,

"name": "Gel Rửa Mặt Cosrx Tràm Trà 94\_Van",

"description": ".5% BHA Có Độ pH Thấp 150ml Low pH Good Morning Gel Cleanser",

"price": 125000,

"category\_id": 4

},

{

"id": 12,

"name": "Sữa Rửa Mặt Cetaphil 94\_Van",

"description": "Dịu Lành Cho Da Nhạy Cảm 500ml",

"price": 329.000,

"category\_id": 4

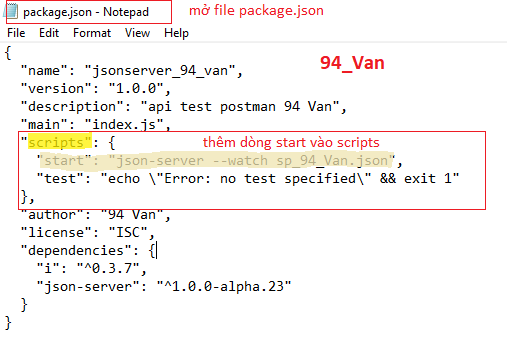
}

]

}

#### Cập nhập file package.json:

* Mở file package.json, thêm dòng sau vào scripts "start": "json-server --watch sp\_94\_Van.json".



Hình 97: Thêm dòng start vào package.json

* Dữ liệu trong package.json

{

"name": "jsonserver\_94\_van",

"version": "1.0.0",

"description": "api test postman 94 Van",

"main": "index.js",

"scripts": {

"start": "json-server --watch sp\_94\_Van.json",

"test": "echo \"Error: no test specified\" && exit 1"

},

"author": "94 Van",

"license": "ISC",

"dependencies": {

"i": "^0.3.7",

"json-server": "^1.0.0-alpha.23"

}

}

#### Chạy npm start:

* Khi nhập lệnh npm start, NPM sẽ thực thi lệnh được định nghĩa trong thuộc tính "start" của đối tượng "scripts" trong tệp package.json.
* Điều này có nghĩa npm sẽ chạy và tạo API từ dữ liệu được nhập vào sp\_94\_Van.json.



Hình 98: Nhập lệnh npm start

* Đây là API của nó (API là url chứa dữ liệu mình muốn hiển thị ra) <http://localhost:3000/sp_94_Van>
* Trên google chạy thử URL để kiểm tra kết quả



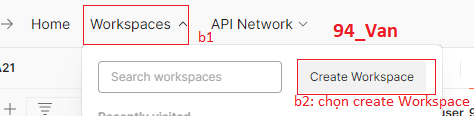
Hình 99: Kiểm tra kết quả API vừa tạo

## Kiểm thử API Postman:

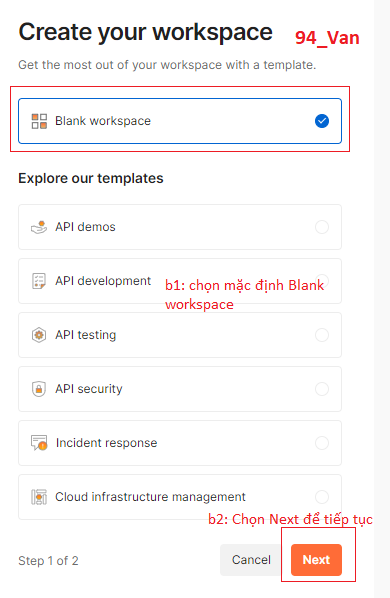
### Tạo Workspace:

#### Tạo một workspace mới:

* Tại Workspace chọn Create Workspace -> chọn mặc định blank workspace -> chọn Next để tiếp tục

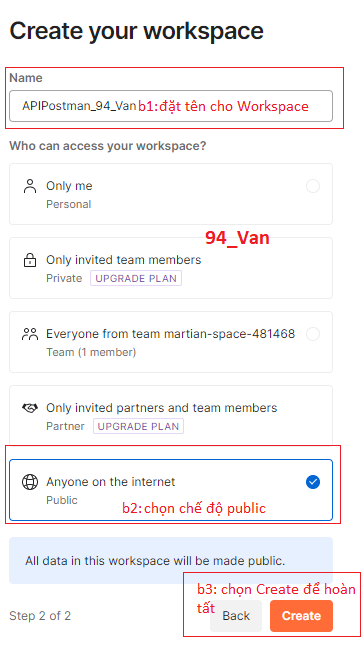


Hình 100: Create Workspace



Hình 101: Chọn Blank Workspace

* Đặt tên cho workspace, chọn chế độ là public(anyone on the Internet), chọn Create để hoàn tất tạo workspace.

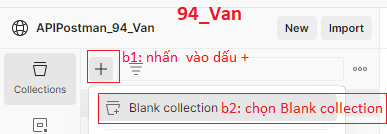


Hình 102: Đặt tên workspace

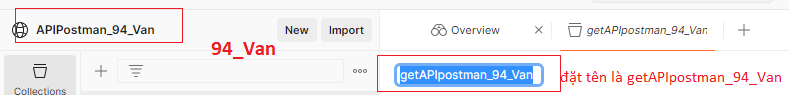
### Phương thức Get:

#### Tạo và lưu một request: Get:

* Tạo Collection tên là getAPIpostman\_94\_Van

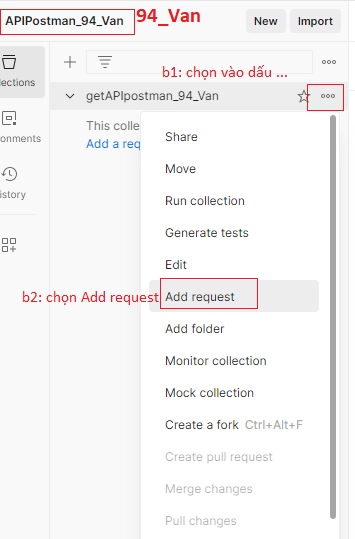


Hình 103: Tạo collection



Hình 104: Đặt tên collection

* Tạo một request có tên getListAPI\_94\_Van.



Hình 105: Tạo request Get



Hình 106: Đặt tên cho request

* Dán link API tạo ở trên(bằng npm) vào textbox, để phương thức là GET. Nhấn Send để gửi request lên server.
* Dữ liệu được gửi lên sẽ xuất hiện ở Body.

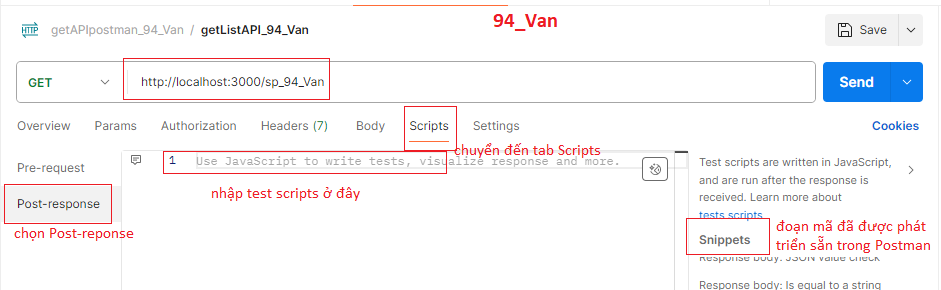


Hình 107: Send request Get

* Chọn Save(Ctrl+S) để lưu request.

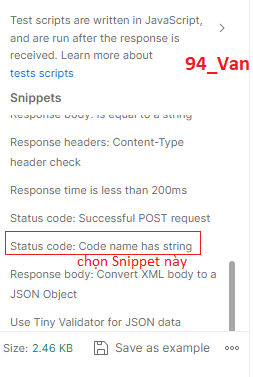
#### API testcase: Get:

* Chuyển đến tab Scripts, chọn post-response, bắt đầu viết testcase. Có thể sử dụng các Snippets để tạo các đoạn mã đã được phát triển sẵn trong Postman để kiểm tra API.



Hình 108: Tạo testcase cho getListAPI\_94\_Van

* Từ Snippets chọn Status name: Code name has string. Để kiểm tra xem tên mã trạng thái có chứa chuỗi “OK”(vì là phương thức get nên kết quả status code sẽ là OK) không.



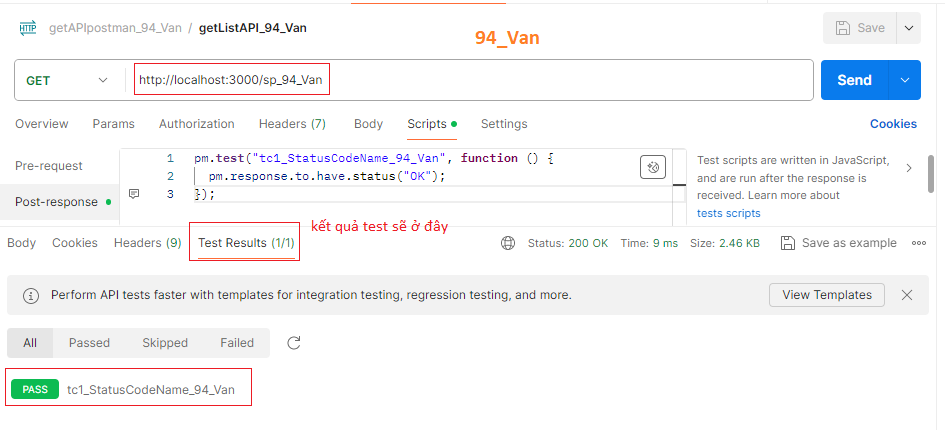
Hình 109: Chọn Status code: Code name has string cho Get Method

* Đặt lại tên testcase theo ý muốn, nhấn Save( Ctrl+S) để lưu trữ testcase.
* Có thể nhấn Send để kiểm tra kết quả testcase.



Hình 110:Get Method: testcase 1 Status Code Name

* Kiểm tra kết quả: Kết quả sẽ là ‘Pass’ tại vì phương thức Get trỏ ra kết quả status code là 200 OK, và ta kiểm tra giá trị status ấy là ‘OK’.



Hình 111: Kết quả testcase 1 Status Code Name

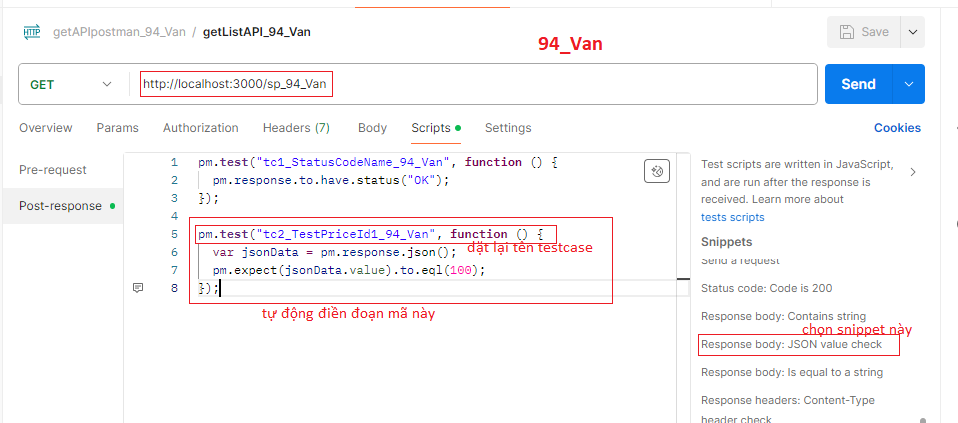
* Code:

pm.**test**("tc1\_StatusCodeName\_94\_Van", **function** () {

  pm.response.to.have.status("OK");

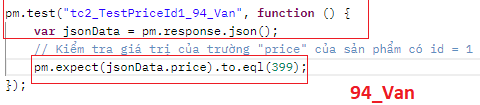
});

* Quay lại tab Scripts và thêm một bài kiểm tra khác. Từ phần Snippets, hãy nhấp vào Response body: JSON value check.



Hình 112: Get Method: testcase 2 Kiểm tra giá sản phẩm có id = 1

* Thay thế tên mặc định thành ‘tc2\_TestPriceId1\_94\_Van’, tìm sản phẩm có id = 1 trong mảng JSON. Kiểm tra giá trị của trường "price".



Hình 113: Code tc2\_TestPriceId1\_94\_Van

* Code:

pm.**test**("tc2\_TestPriceId1\_94\_Van", **function** () {

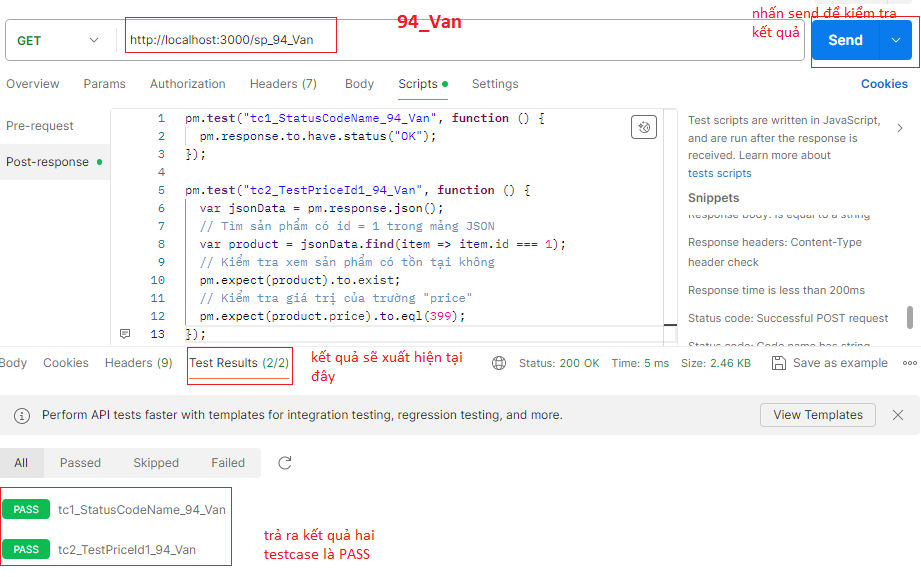
  var jsonData **=** pm.response.json();

  // Kiểm tra giá trị của trường "price"

  pm.expect(product.price).to.eql(399);

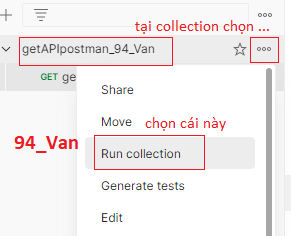
});

* Kiểm tra kết quả: Kết quả sẽ ra ‘Pass’ vì giá trị ‘price’ của sản phẩm id = 1 có giá trị đúng bằng 399.



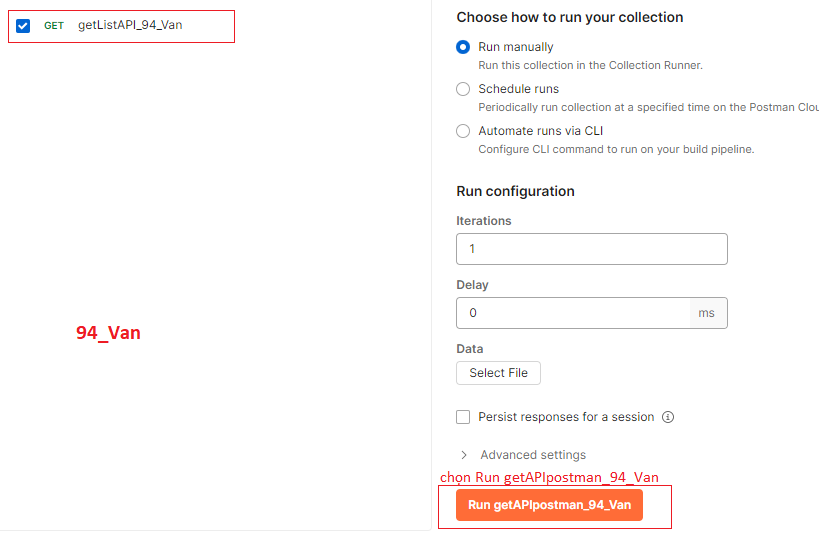
Hình 114: Kết quả testcase 2 Kiểm tra giá sản phẩm có id = 1

* Có thể kiểm tra kết quả bằng cách 2:
* Tại Collection, nhấn chọn dấu ba chấm, chọn Run Collection.



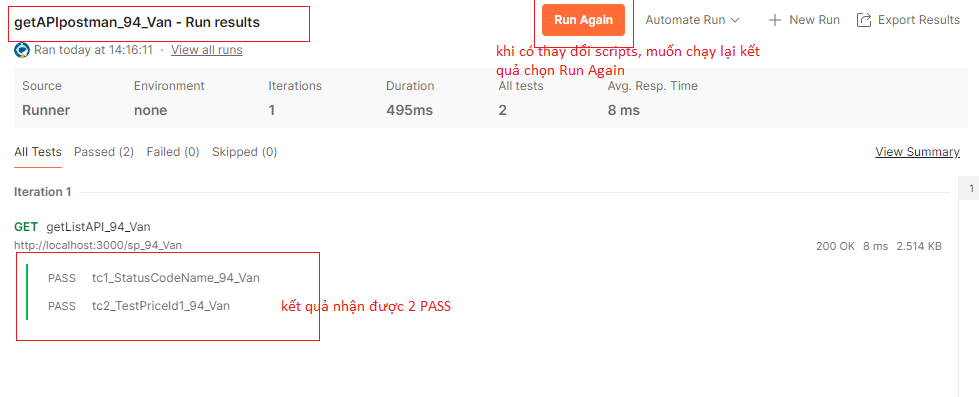
Hình 115: Kiểm tra kết quả bằng collection

* Chọn Run getAPIpostman\_94\_Van



Hình 116: Chọn Run getAPIpostman\_94\_Van

* Kết quả

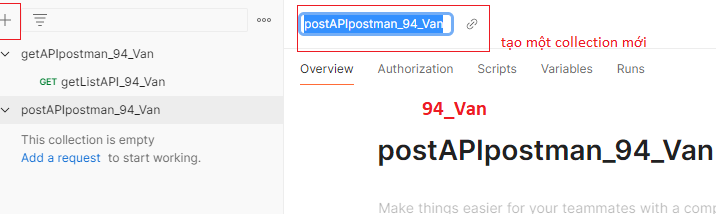


Hình 117: Kết quả của getAPIpostman\_94\_Van

### Phương thức Post:

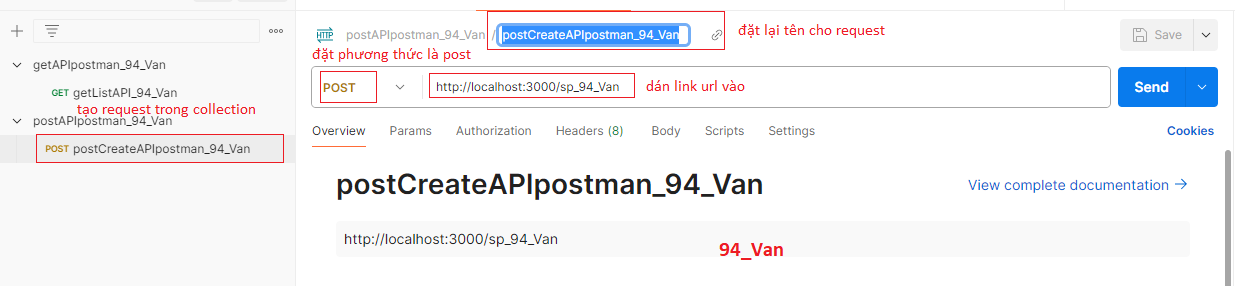
#### Tạo và lưu một request: Post:

* Tương tự như Get Method, tạo Collection mới có tên là postAPIpostman\_94\_Van.



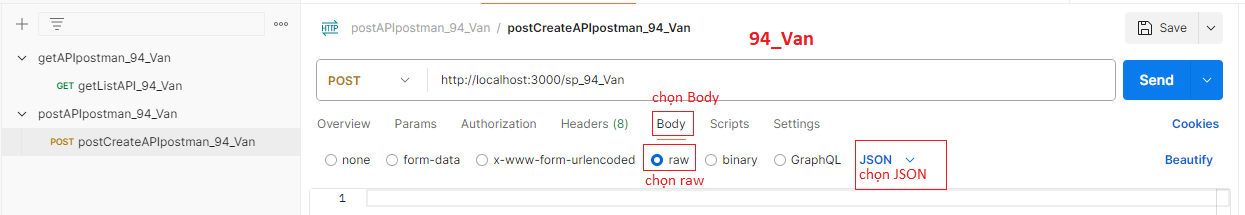
Hình 118: Tạo Collection postAPIpostman\_94\_Van

* Tạo request mới. Đặt phương thức là Post, nhập liên kết URL đã tạo bằng npm vào textbox.



Hình 119: Tạo request postCreateAPIpostman\_94\_Van

* Trong Body, chọn raw, tiếp theo chọn JSON.



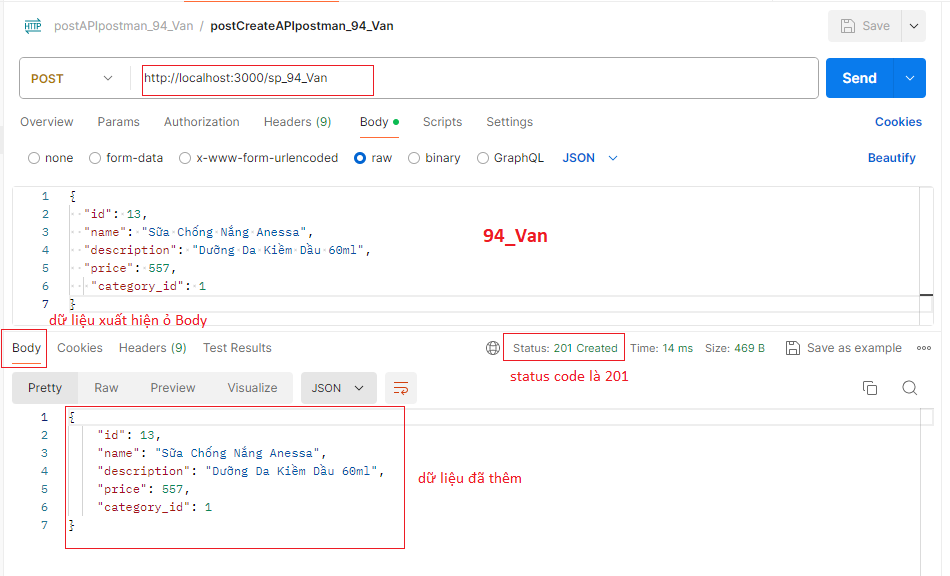
Hình 120: Các bước thực hiện Post

* Sao chép và dán chỉ một kết quả người dùng từ yêu cầu nhận trước đó như bên dưới. Đảm bảo rằng mã đã được sao chép chính xác với cặp dấu ngoặc nhọn và brackets. Thay đổi id thành 13 và đặt tên thành bất kỳ tên nào bạn muốn.



Hình 121: Thêm dữ liệu bằng Post Method

* Nhấn Send để gửi yêu cầu, xuất hiện trạng thái 201 Create, dữ liệu đã đăng xuất hiện ở phần Body.



Hình 122: Send request Post

* Dữ liệu nhập ở JSON:

{

  "id": 13,

  "name": "Sữa Chống Nắng Anessa",

  "description": "Dưỡng Da Kiềm Dầu 60ml",

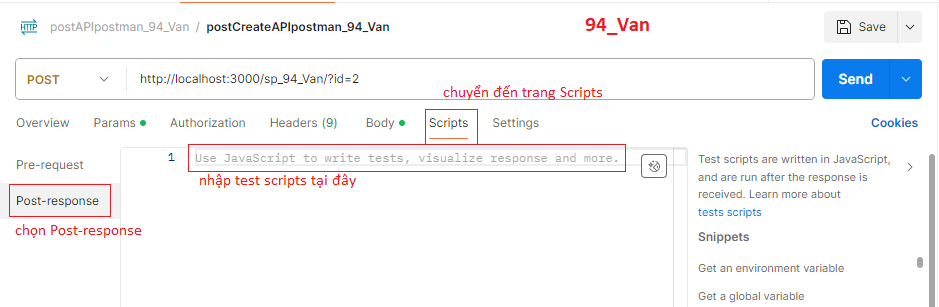
  "price": 557,

   "category\_id": 1

}

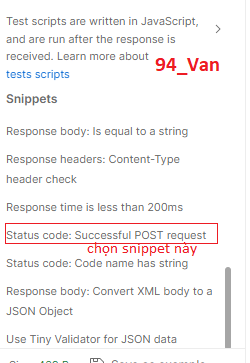
#### API testcase: Post:

* Chuyển đến tab Scripts, chọn post-response, bắt đầu viết testcase. Có thể sử dụng các Snippets để tạo các đoạn mã đã được phát triển sẵn trong Postman để kiểm tra API.



Hình 123: Tạo testcase cho postCreateAPI\_94\_Van

* Từ Snippets chọn Status code: Successful POST request, để kiểm tra request của post là thành công ( với code là 201 hoặc 202).



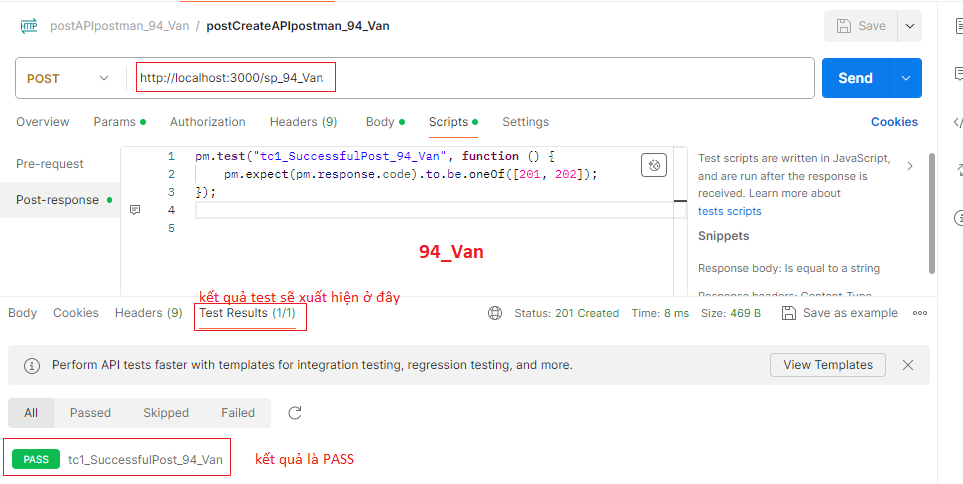
Hình 124: Chọn Status code: Successful POST request Post Method

* Đặt lại tên testcase theo ý muốn, nhấn Save( Ctrl+S) để lưu trữ testcase.
* Có thể nhấn Send để kiểm tra kết quả testcase.



Hình 125: Post Method: testcase 1 Successful Post

* Kiểm tra kết quả: Kết quả sẽ là ‘Pass’ vì gửi lên và thêm được dữ liệu thì sẽ hiện status code là 201 Create trùng với đoạn mã testcase.



Hình 126: Kết quả testcase 1 Successful Post

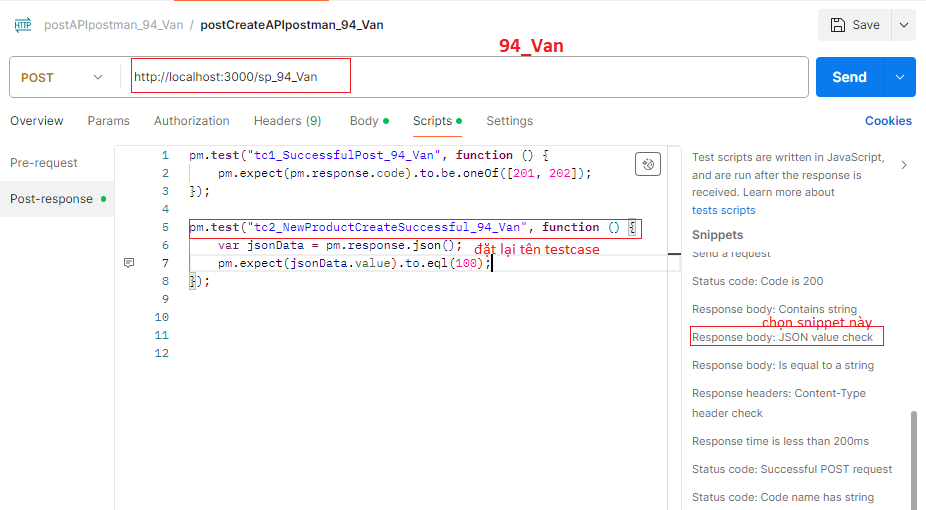
* Code

pm.**test**("tc1\_SuccessfulPost\_94\_Van", **function** () {

    pm.expect(pm.response.code).to.be.oneOf([201, 202]);

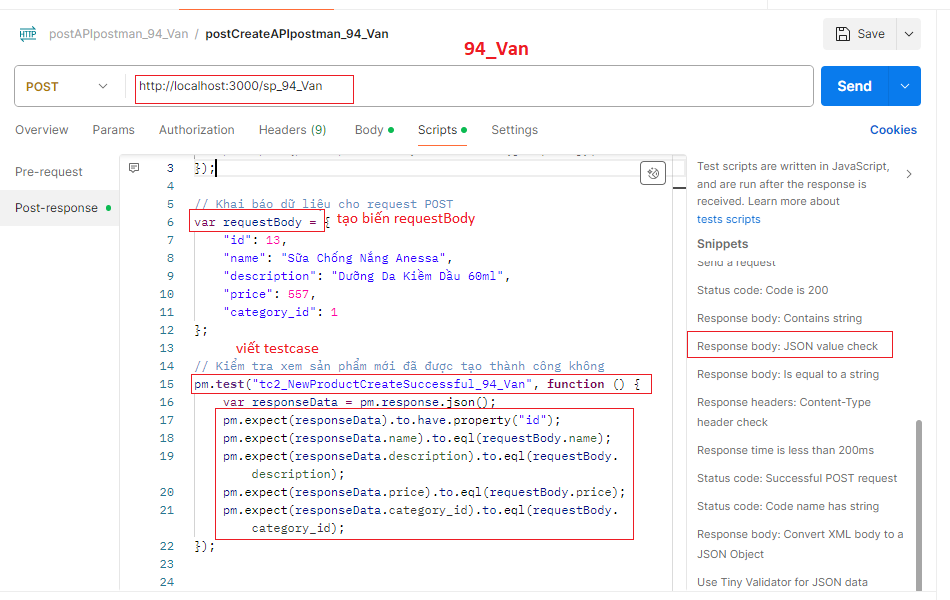
});

* Quay lại tab Scripts và thêm một bài kiểm tra khác. Từ phần Snippets, hãy nhấp vào Response body: JSON value check.



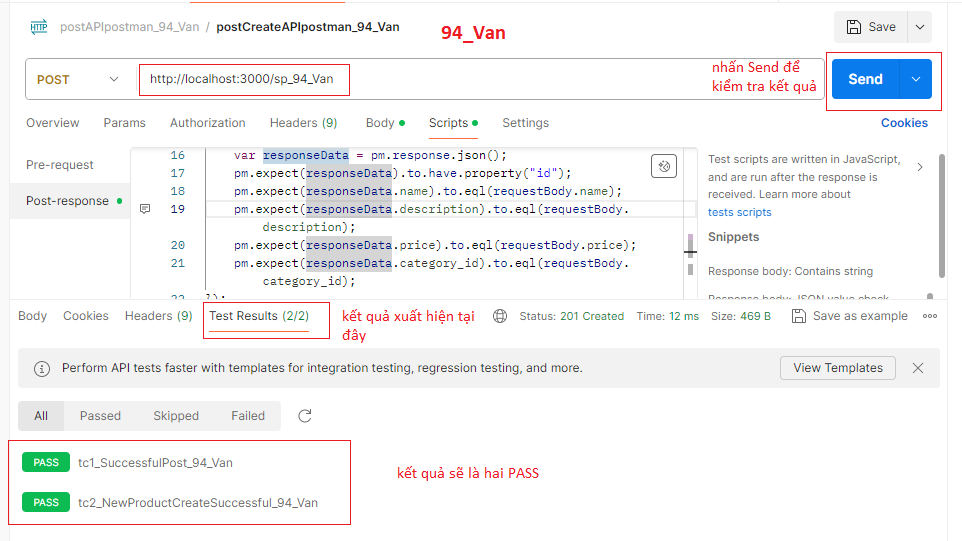
Hình 127: Post Method Kiểm tra đã thêm được sản phẩm mới chưa

* Thay thế tên mặc định thành ‘tc2\_NewProductCreateSuccessful\_94\_Van’. Thêm biến ‘requestBody’ để khai báo dữ liệu cho request POST, dữ liệu mới được thêm bởi method Post. Dùng pm.expect() mà postman hỗ trợ trong Snippets: ‘Response body: JSON value check’ để kiểm tra xem sản phẩm mới đã được tạo thành công không.



Hình 128: Code testcase 2 Kiểm tra đã thêm được sản phẩm mới chưa

* Kiểm tra kết quả: Kết quả sẽ ra ‘Pass’ vì dữ liệu trên đã được thêm bởi Post.



Hình 129: Kết quả testcase 2 Kiểm tra đã thêm được sản phẩm mới chưa

* Code:

// Khai báo dữ liệu cho request POST

**var** requestBody = {

    "id": 13,

    "name": "Sữa Chống Nắng Anessa",

    "description": "Dưỡng Da Kiềm Dầu 60ml",

    "price": 557,

    "category\_id": 1

};

// Kiểm tra xem sản phẩm mới đã được tạo thành công không

pm.test("tc2\_NewProductCreateSuccessful\_94\_Van", **function** () {

**var** responseData = pm.response.json();

    pm.expect(responseData).to.have.property("id");

    pm.expect(responseData.name).to.eql(requestBody.name);

    pm.expect(responseData.description).to.eql(requestBody.description);

    pm.expect(responseData.price).to.eql(requestBody.price);

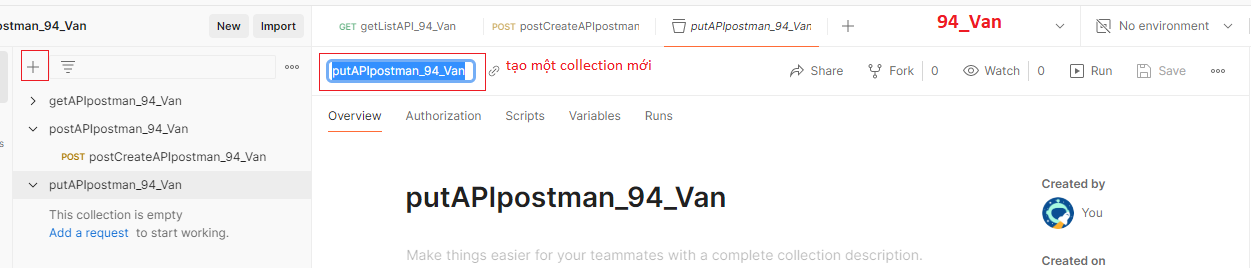
    pm.expect(responseData.category\_id).to.eql(requestBody.category\_id);

});

### Phương thức Put/Patch:

#### Tạo và lưu một request: Put:

* Tương tự tạo Collection tên là putAPIpostman\_94\_Van.



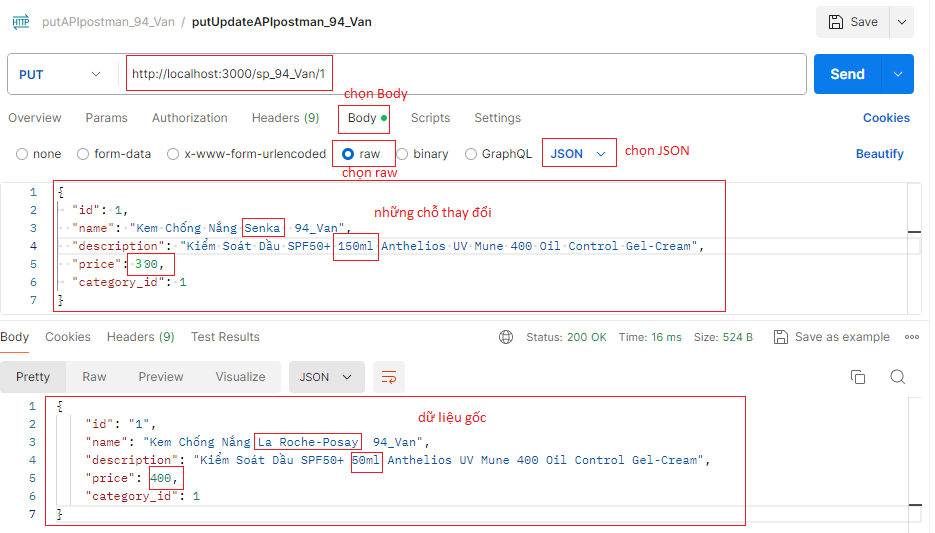
Hình 130: Tạo Collection putAPIpostman\_94\_Van

* Tạo request mới. Đặt phương thức là Put, nhập liên kết URL đã tạo bằng npm vào textbox.



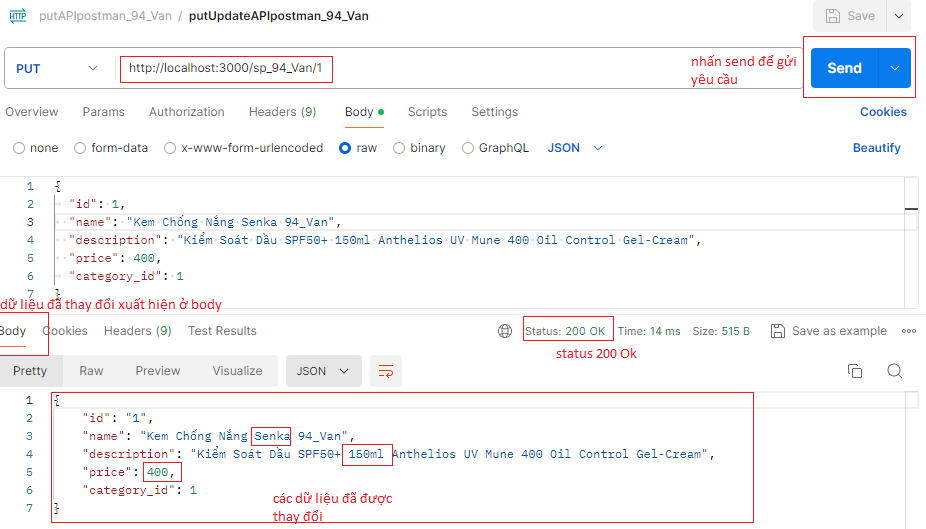
Hình 131: Tạo request putUpdateAPIpostman\_94\_Van

* Trong Body, chọn raw, tiếp theo chọn JSON.
* Sao chép và dán một đoạn dữ liệu có id = 1 và thay đổi dữ liệu theo mong muốn.



Hình 132: Các bước thực hiện Put

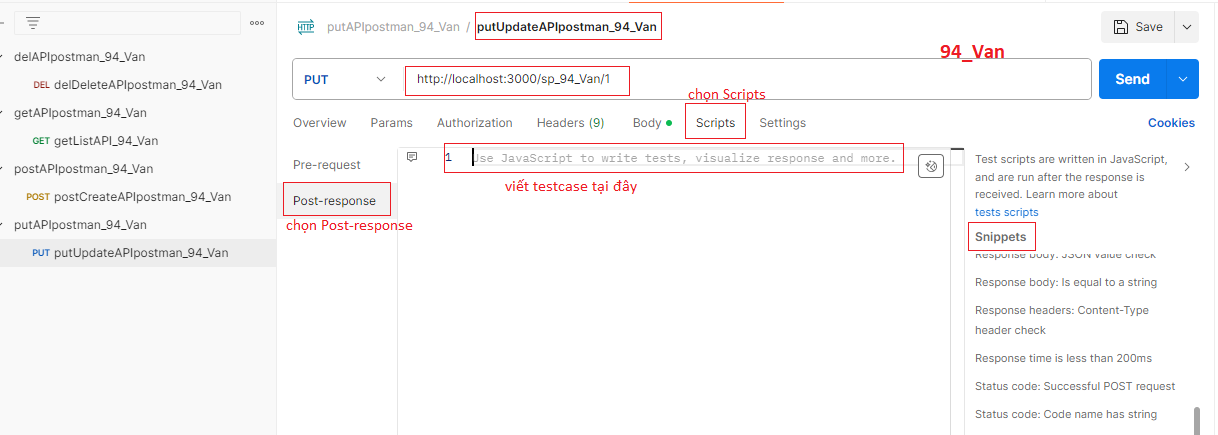
* Nhấn Send để gửi yêu cầu, xuất hiện trạng thái 201 OK, dữ liệu đã thay đổi xuất hiện ở phần Body.



Hình 133: Send request Put

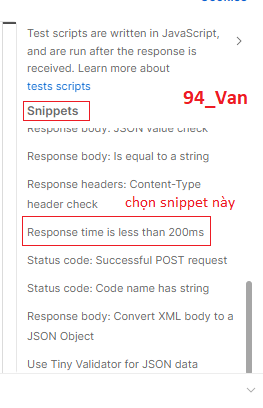
#### API testcase: Put:

* Chuyển đến tab Scripts, chọn post-response, bắt đầu viết testcase. Có thể sử dụng các Snippets để tạo các đoạn mã đã được phát triển sẵn trong Postman để kiểm tra API.



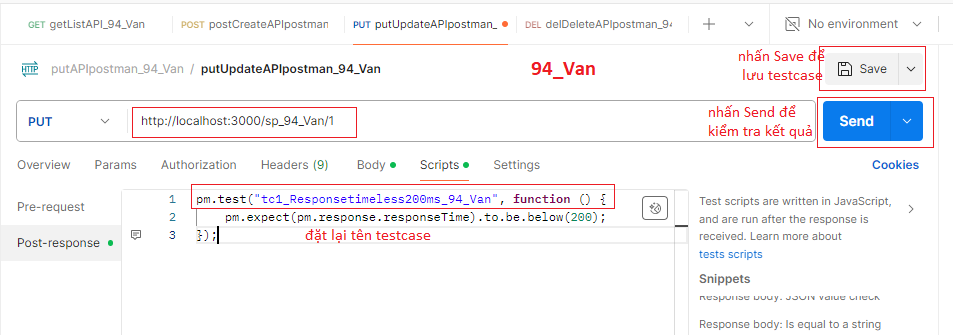
Hình 134: Tạo testcase cho putUpdateAPIpostman\_94\_Van

* Từ Snippets chọn Response time is less than 200ms, để kiểm tra thời gian phản hồi của request có nhỏ hơn 200ms hay không.



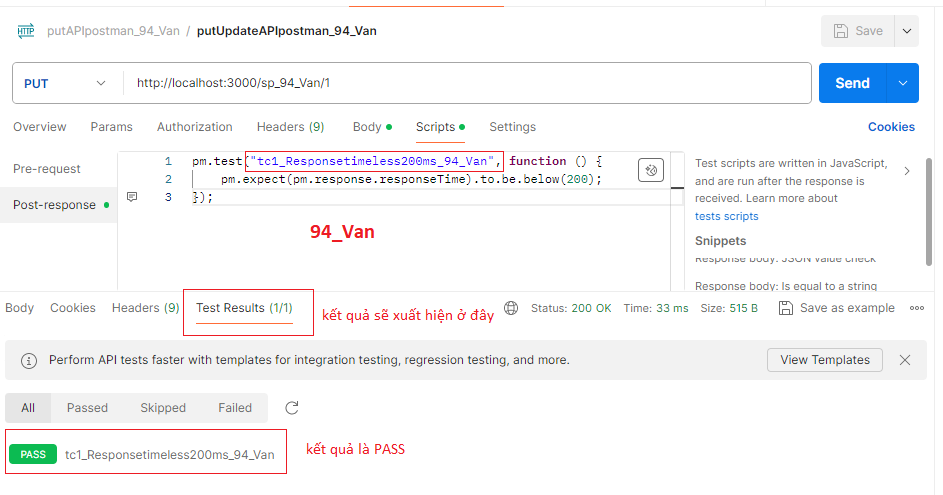
Hình 135: Chọn Response time is less than 200ms

* Đặt lại tên testcase theo ý muốn, nhấn Save( Ctrl+S) để lưu trữ testcase.
* Có thể nhấn Send để kiểm tra kết quả testcase.



Hình 136: Put Method: testcase 1 Response time less 200ms

* Kiểm tra kết quả: Kết quả sẽ là ‘Pass’



Hình 137: Kết quả testcase 1 Response time less 200ms

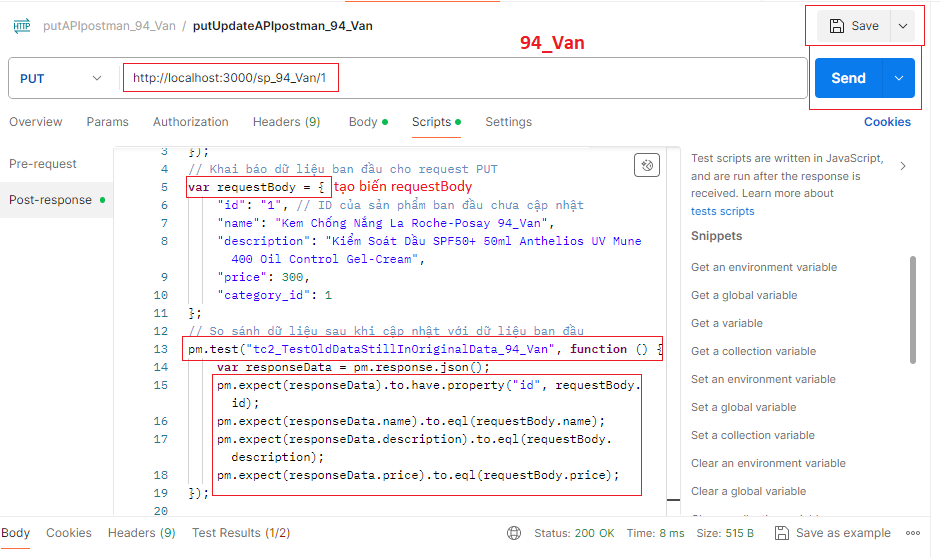
* Code:

pm.test("tc1\_Responsetimeless200ms\_94\_Van", **function** () {

    pm.expect(pm.response.responseTime).to.be.below(200);

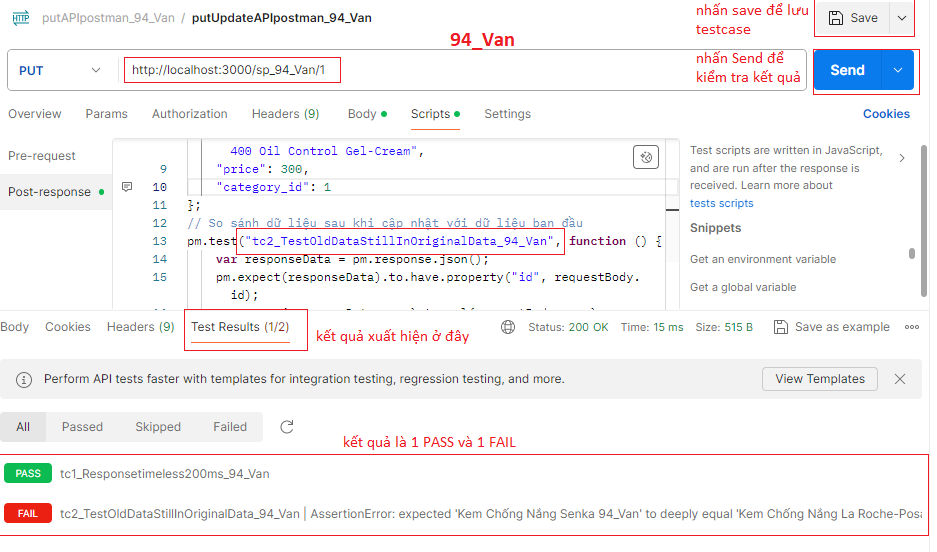
});

* Quay lại tab Scripts và thêm một bài kiểm tra khác. Từ phần Snippets, hãy nhấp vào Response body: JSON value check. Tạo một biến ‘requestBody’ chứa dữ liệu là sản phẩm ban đầu chưa được thay đổi. Dùng pm.expect() mà postman hỗ trợ trong Snippets: ‘Response body: JSON value check’ để so sánh giá trị của ‘requestBody’ có nằm trong data hay không.



Hình 138: Code testcase 2 Kiểm tra dữ liệu có còn sản phẩm ban đầu không

* Thay thế tên mặc định thành ‘tc2\_TestOldDataStillInOriginalData\_94\_Van’.
* Kiểm tra kết quả: Kết quả sẽ ra ‘Fail’.



Hình 139: Kết quả của testcase 2 Kiểm tra dữ liệu có còn sản phẩm ban đầu không

* Code:

// Khai báo dữ liệu ban đầu cho request PUT

**var** requestBody = {

    "id": "1", // ID của sản phẩm ban đầu chưa cập nhật

    "name": "Kem Chống Nắng La Roche-Posay 94\_Van",

    "description": "Kiểm Soát Dầu SPF50+ 50ml Anthelios UV Mune 400 Oil Control Gel-Cream",

    "price": 300,

    "category\_id": 1

};

// So sánh dữ liệu sau khi cập nhật với dữ liệu ban đầu

pm.test("tc2\_TestOldDataStillInOriginalData\_94\_Van", **function** () {

**var** responseData = pm.response.json();

    pm.expect(responseData).to.have.property("id", requestBody.id);

    pm.expect(responseData.name).to.eql(requestBody.name);

    pm.expect(responseData.description).to.eql(requestBody.description);

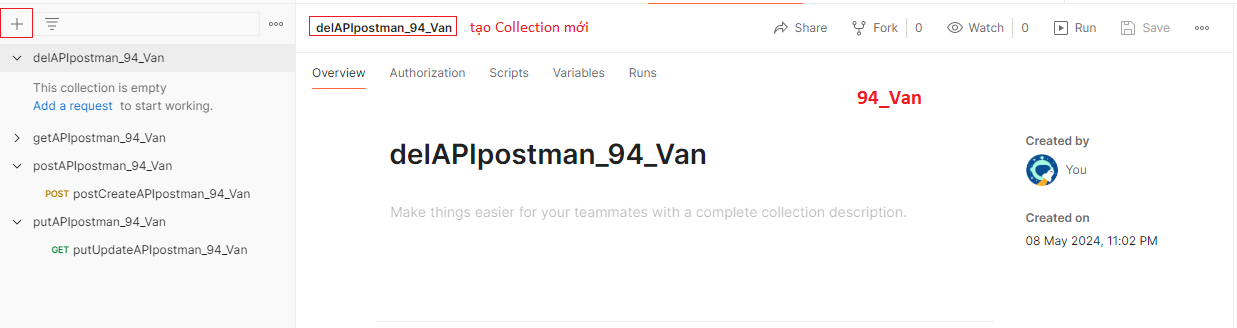
    pm.expect(responseData.price).to.eql(requestBody.price);

});

### Phương thức Delete:

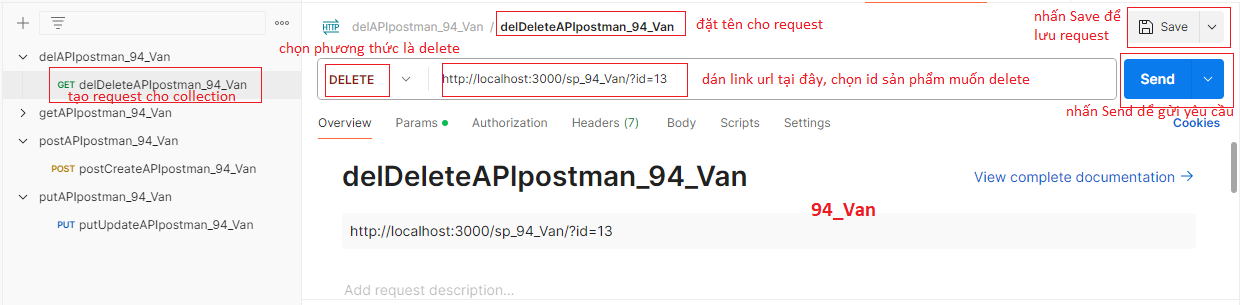
#### Tạo và lưu một request: Delete:

* Tương tự tạo Collection tên là delAPIpostman\_94\_Van.



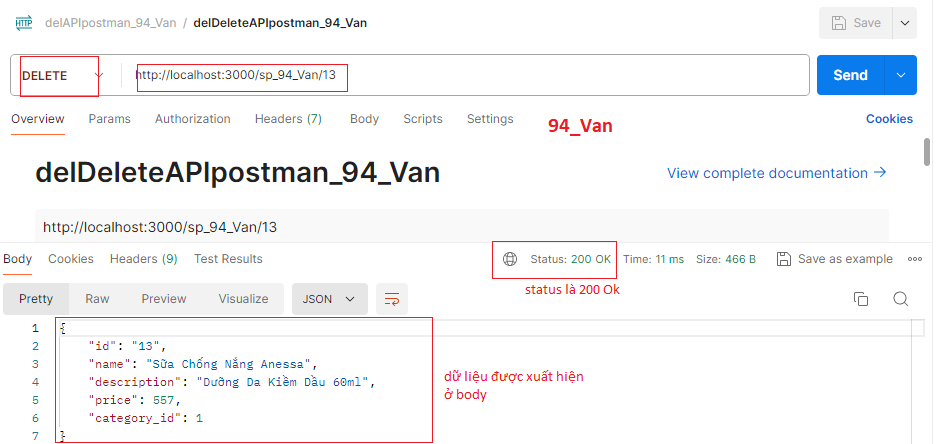
Hình 140: Tạo Collection delAPIpostman\_94\_Van

* Tạo request mới. Đặt phương thức là Delete, nhập liên kết URL đã tạo bằng npm vào textbox.



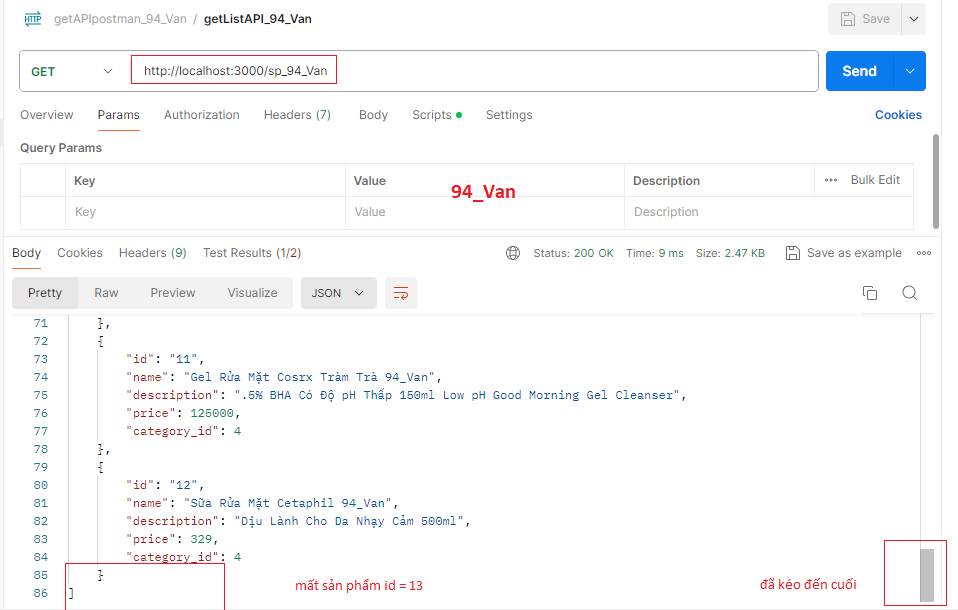
Hình 141: Tạo request cho delAPIpostman\_94\_Van

* Nhấn Send để gửi yêu cầu, xuất hiện trạng thái 200 OK, dữ liệu đã xóa xuất hiện ở phần Body.



Hình 142: Send request Delete

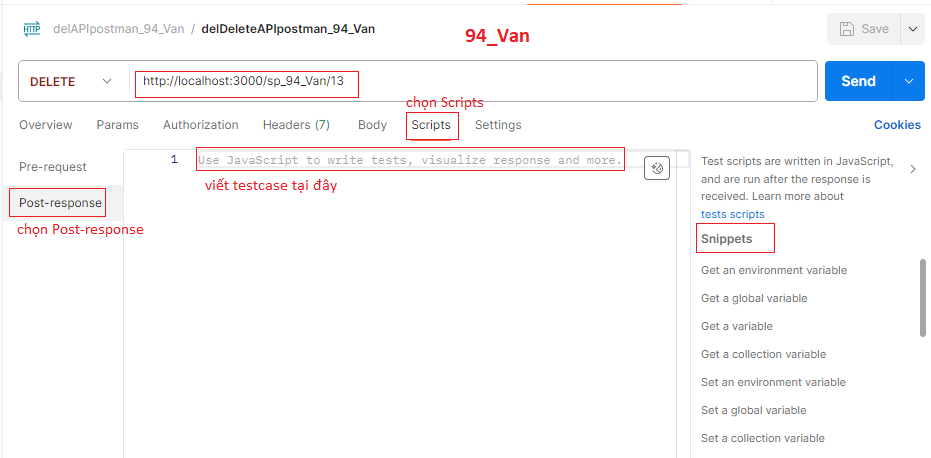
* Vào phương thức Get ‘getAPIpostman\_94\_Van’ để kiểm tra kết quả. Đã mất sản phẩm có id = 13.



Hình 143: Kiểm tra kết quả Delete

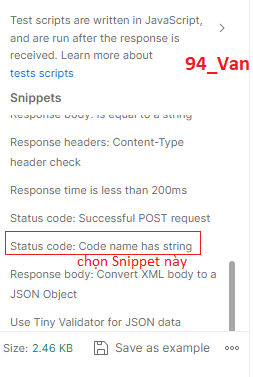
#### API testcase: Delete:

* Chuyển đến tab Scripts, chọn post-response, bắt đầu viết testcase. Có thể sử dụng các Snippets để tạo các đoạn mã đã được phát triển sẵn trong Postman để kiểm tra API.



Hình 144: Tạo testcase cho DelDeleteAPIpostman\_94\_Van

* Từ Snippets chọn Status name: Code name has string. Để kiểm tra xem tên mã trạng thái có chứa chuỗi “OK”(vì là phương thức Delete thành công nên kết quả status code sẽ là OK) không.



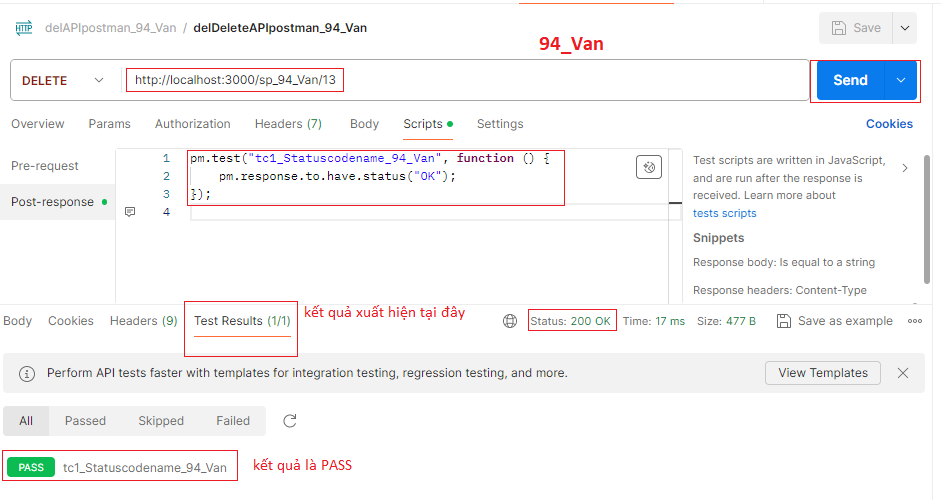
Hình 145: Chọn Status code: Code name has string cho Delete Method

* Đặt lại tên testcase theo ý muốn, nhấn Save( Ctrl+S) để lưu trữ testcase.
* Có thể nhấn Send để kiểm tra kết quả testcase.



Hình 146: Delete Method: testcase 1 Status Code Name

* Kiểm tra kết quả: Kết quả sẽ là ‘Pass’.



Hình 147: Kết quả testcase 1 Status Code Name

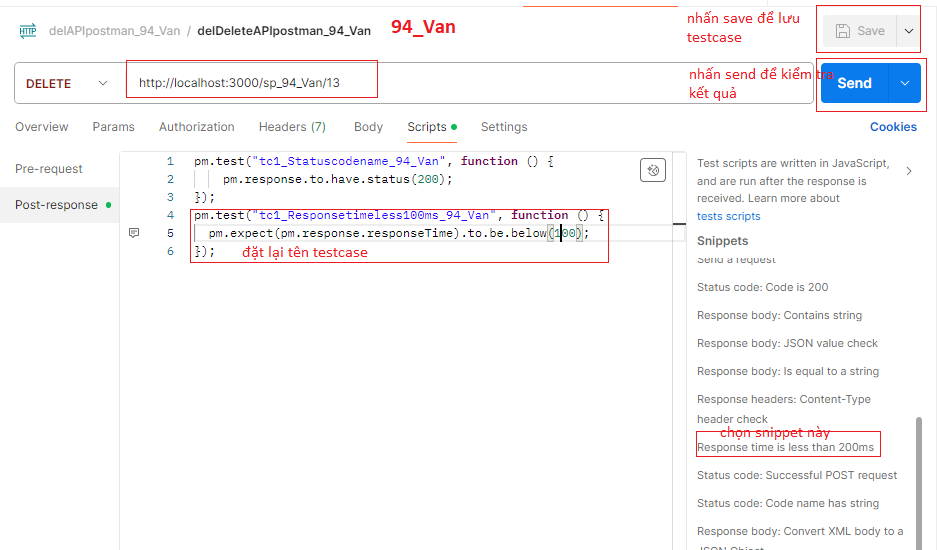
* Code:

pm.test("tc1\_Statuscodename\_94\_Van", **function** () {

    pm.response.to.have.status(200);

});

* Quay lại tab Scripts và thêm một bài kiểm tra khác. Từ phần Snippets, chọn Response time is less than 100ms, để kiểm tra thời gian phản hồi của request có nhỏ hơn 100ms hay không.



Hình 148: Code testcase 2 Response time less 200ms

* Kiểm tra kết quả: Kết quả sẽ ra ‘Pass’



Hình 149: Kết quả testcase 2 Response time less 200ms

* Code:

pm.test("tc1\_Responsetimeless100ms\_94\_Van", **function** () {

  pm.expect(pm.response.responseTime).to.be.below(100);

});