Mục Lục

[NUNIT 3](#_Toc196206561)

[1.Tạo một project c# thực hiện phép toán luỹ thừa 3](#_Toc196206562)

[Bước 1 3](#_Toc196206563)

[Bước 2 4](#_Toc196206564)

[2.Tạo giao diện thao tác phép toán 5](#_Toc196206565)

[Bước 1.Tạo Giao Diện 5](#_Toc196206566)

[Bước 2. Viết code thực hiện phép toán trên giao diện 5](#_Toc196206567)

[3. Tạo project để kiểm thử để kiểm thử các phép toán trong chương trình trên 7](#_Toc196206568)

[Bước 1. Tạo project kiểm thử 7](#_Toc196206569)

[Bước 2. Tham chiếu đến project cần thực hiện Unit Test 9](#_Toc196206570)

[Bước 3. Tạo dữ liệu mẫu và thiết lập để kiểm thử từ DataSource 11](#_Toc196206571)

[Bước 4. Viết code kiểm thử phương thức 15](#_Toc196206572)

[4.Kết quả 18](#_Toc196206573)

[5.Test Case. 18](#_Toc196206574)

[Đặc tả yêu cầu 18](#_Toc196206575)

[Test case 1:cơ số x và số mũ n đều là số nguyên dương(TC1\_xLonHon0\_76\_phuc). 19](#_Toc196206576)

[Test case 2: cơ số x là số nguyên âm(TC2\_xNhoHon0\_76\_phuc). 19](#_Toc196206577)

[Test case 3: số mũ n bằng 0(TC3\_nBang0\_76\_phuc). 19](#_Toc196206578)

[Test case 4: số mũ n âm(TC4\_nNhoHon0\_76\_phuc). 19](#_Toc196206579)

[Test case 5: cơ số x bằng 0(TC5\_xBang0\_76\_phuc). 20](#_Toc196206580)

[Test case 6: cơ số x là số thực(TC6\_xLaSoThuc\_76\_phuc). 20](#_Toc196206581)

[Test case 7: với cơ số x bằng 0, số mũ n nhỏ hơn 0(TC7\_xBang0VanNhoHon0\_76\_phuc). 20](#_Toc196206582)

[WEB DRIVE-Facebook 21](#_Toc196206583)

[1. Cài đặt Selenium Web Driver( Selenium.WebDriver và Selenium.WebDriver.ChromeDriver) 21](#_Toc196206584)

[2.Thiết kế giao diện xử lý các chức năng. 23](#_Toc196206585)

[3.Code thực hiện chức năng. 23](#_Toc196206586)

[4.Các Chức năng. 27](#_Toc196206587)

[1.Chức năng đăng nhập. 27](#_Toc196206588)

[2.Chức năng gửi lời mời kết bạn. 27](#_Toc196206589)

[3.Chức năng bình luận bài viết. 28](#_Toc196206590)

[API POSTMAN 30](#_Toc196206591)

[1.API Json server. 30](#_Toc196206592)

[1. Cài đặt Node.js 30](#_Toc196206593)

[2.Cài đặt Json server. 30](#_Toc196206594)

[3.Tạo file json cho server ảo. 33](#_Toc196206595)

[4.Mở file package.json, thêm dòng start. 40](#_Toc196206596)

[2.Test API. 44](#_Toc196206597)

[1.Tạo workspace: 44](#_Toc196206598)

[2.Get API 47](#_Toc196206599)

[3.Post API 51](#_Toc196206600)

[4.Put API 55](#_Toc196206601)

[5.PATCH API. 58](#_Toc196206602)

[6.Delete API. 60](#_Toc196206603)

[3.Postman variable. 61](#_Toc196206604)

[1.Biến globals 61](#_Toc196206605)

[2. Biến collections 63](#_Toc196206606)

[4. Postman script 66](#_Toc196206607)

[Tạo script 66](#_Toc196206608)

[1.Get API. 67](#_Toc196206609)

[2.Post API. 68](#_Toc196206610)

[3.Put API. 70](#_Toc196206611)

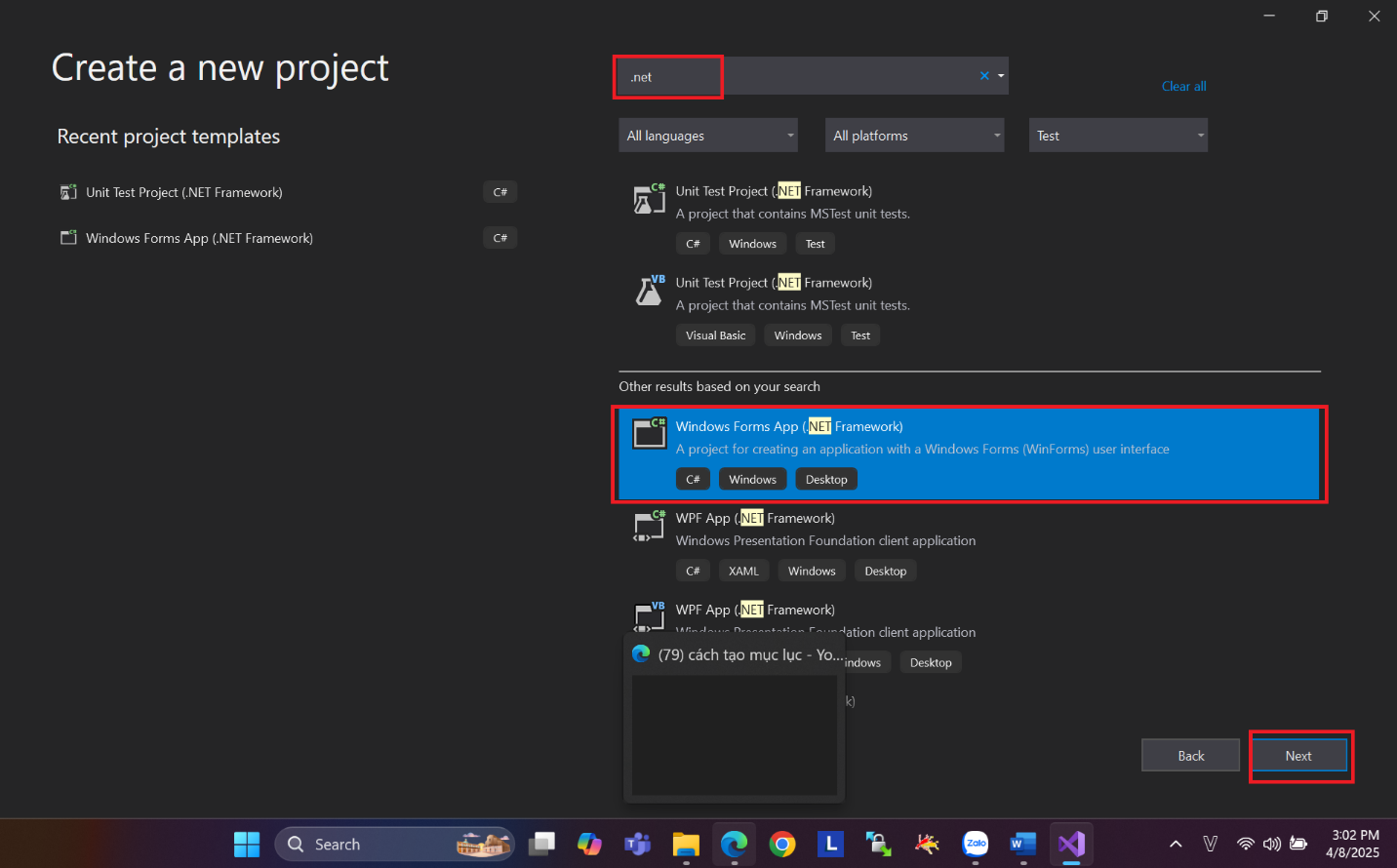
[4.Patch API. 71](#_Toc196206612)

[5.Delete API. 72](#_Toc196206613)

# NUNIT

## 1.Tạo một project c# thực hiện phép toán luỹ thừa

### Bước 1



### Bước 2



## 2.Tạo giao diện thao tác phép toán

### Bước 1.Tạo Giao Diện

A screenshot of a computer

AI-generated content may be incorrect.

### Bước 2. Viết code thực hiện phép toán trên giao diện

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;

namespace DoAn\_76\_phuc

{

public partial class Power : Form

{

public Power()

{

InitializeComponent();

}

private void label1\_Click(object sender, EventArgs e)

{

}

private void Form1\_Load(object sender, EventArgs e)

{

label3.Text = "x\u207F = ";

label4.Text = "Chương Trình Tính Luỹ thừa: x\u207F";

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

}

public static double power\_76\_phuc(double x\_76\_phuc, int n\_76\_phuc)

{

if (n\_76\_phuc == 0)

return 1.0;

else if (n\_76\_phuc > 0)

return x\_76\_phuc \* power\_76\_phuc(x\_76\_phuc, n\_76\_phuc - 1);

else

return power\_76\_phuc(x\_76\_phuc, n\_76\_phuc + 1) / x\_76\_phuc;

}

private void button1\_Click(object sender, EventArgs e)

{

double x;

int n;

double kq;

x = double.Parse(txtX.Text);

n = int.Parse(txtN.Text);

kq = power\_76\_phuc(x, n);

txtKQ.Text = kq.ToString();

}

}

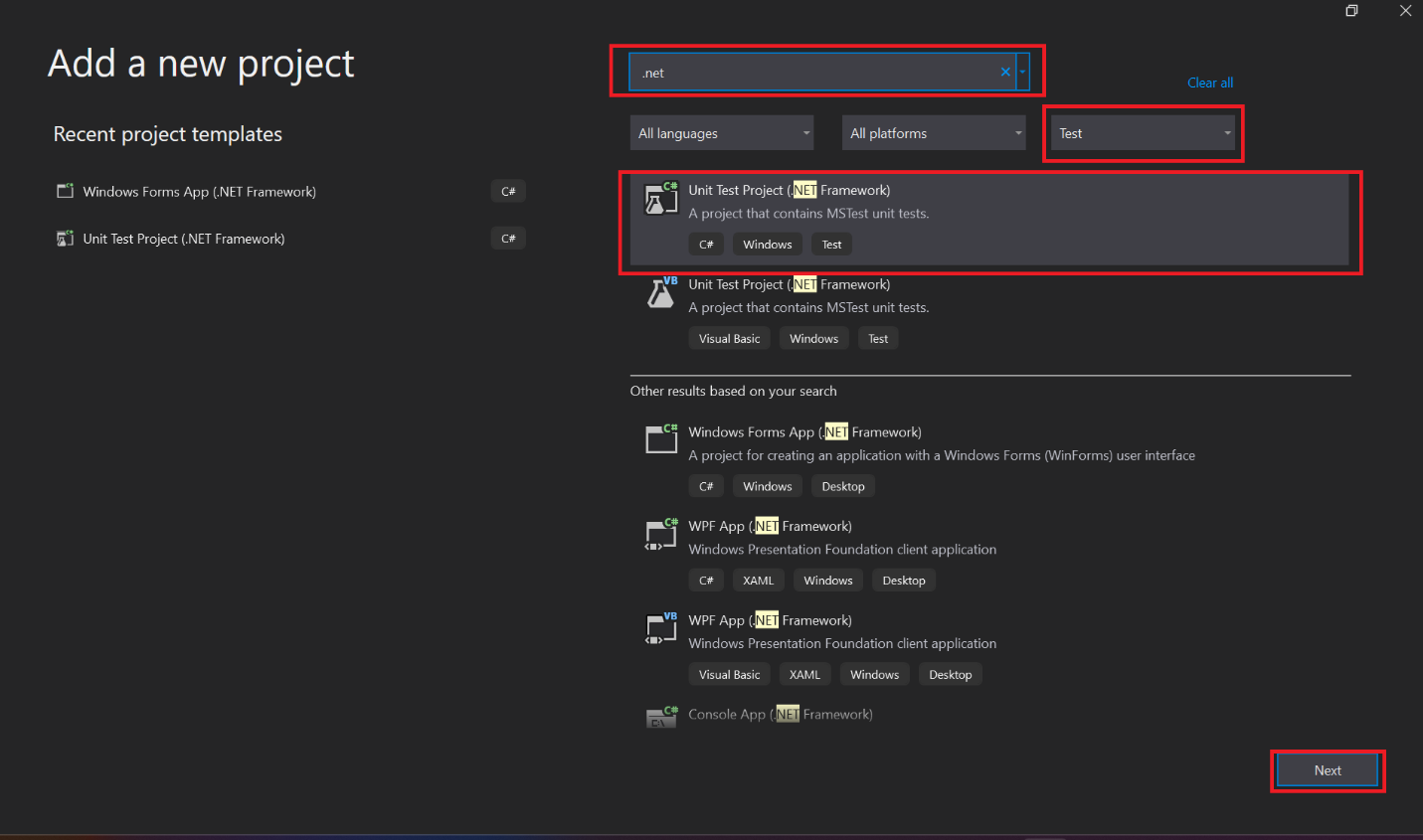
}

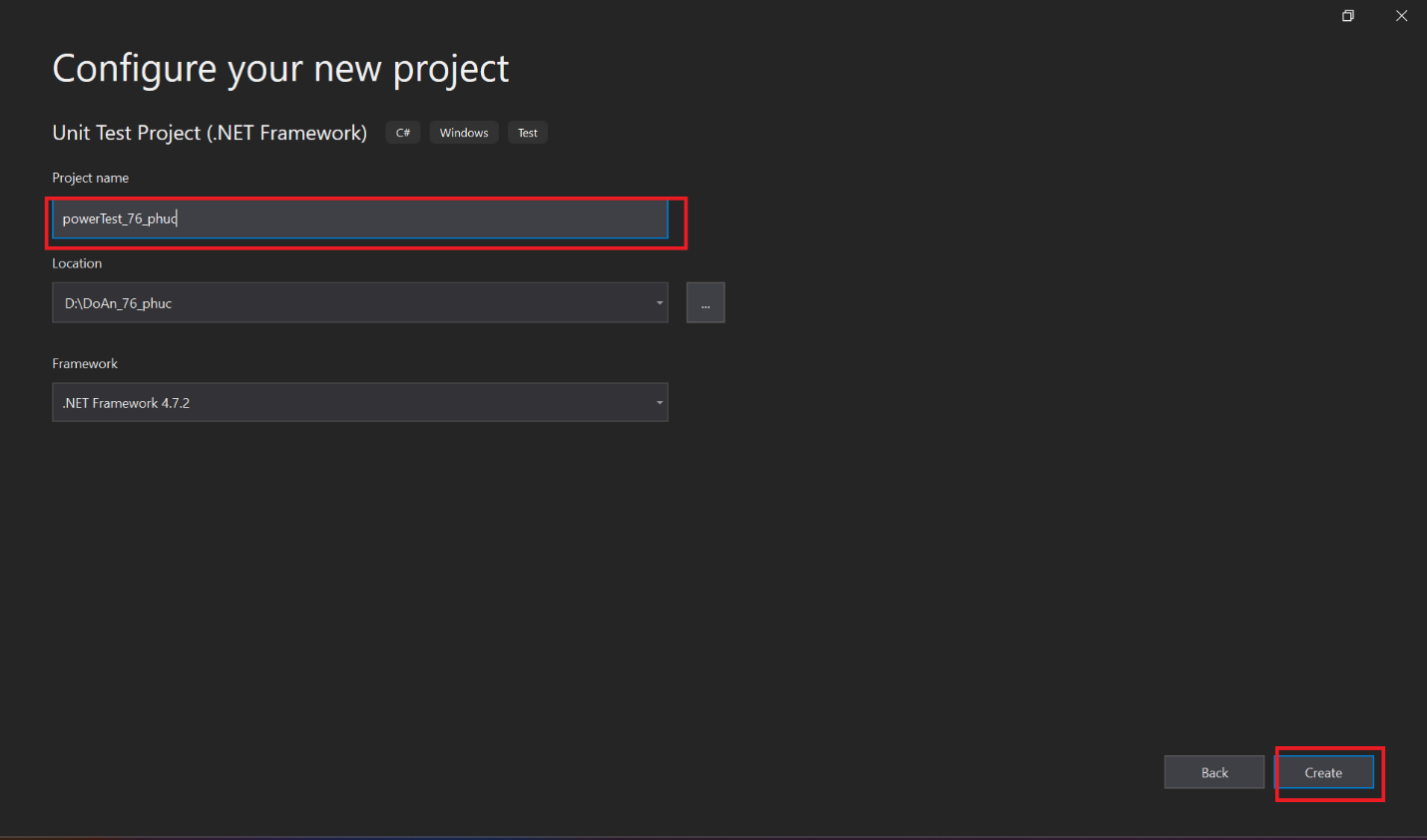
## 3. Tạo project để kiểm thử để kiểm thử các phép toán trong chương trình trên

### Bước 1. Tạo project kiểm thử

**Click chuột phải vào Solution**A screenshot of a computer program

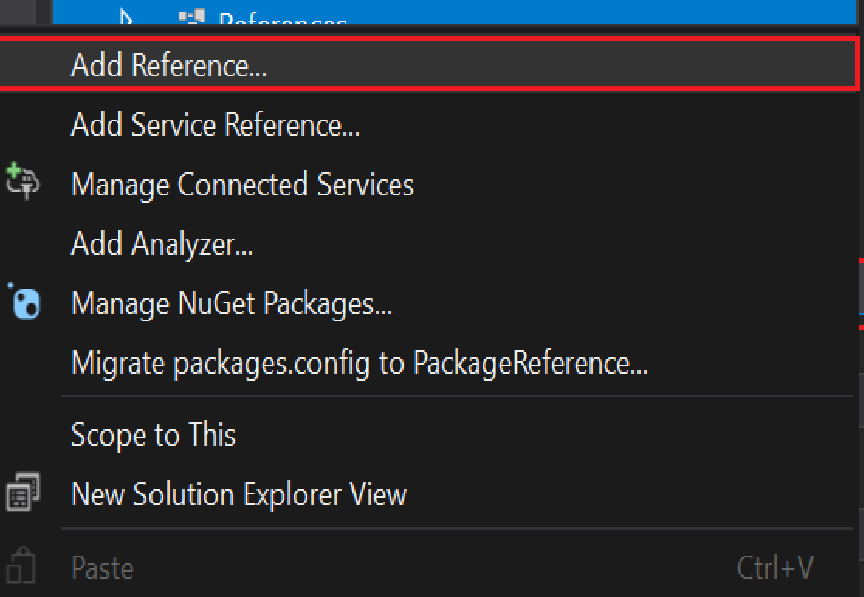
AI-generated content may be incorrect.

****



### Bước 2. Tham chiếu đến project cần thực hiện Unit Test

**Click chuột phải vào Reference**



**Click chọn project cần tham chiếu đến** A screenshot of a computer

AI-generated content may be incorrect.

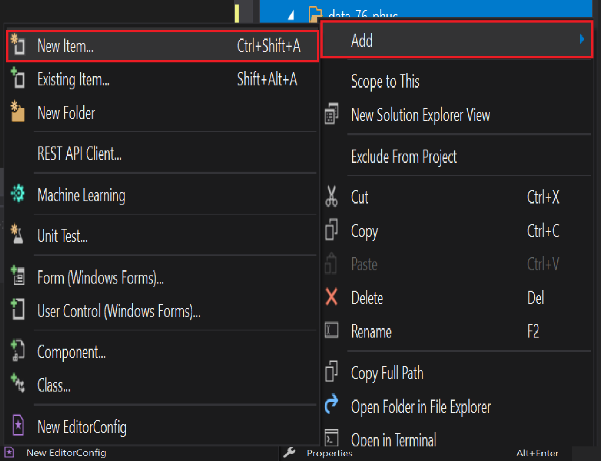
### Bước 3. Tạo dữ liệu mẫu và thiết lập để kiểm thử từ DataSource

#### Tạo data file

**Click chuột phải vào project kiểm thử** A screenshot of a computer program

AI-generated content may be incorrect.

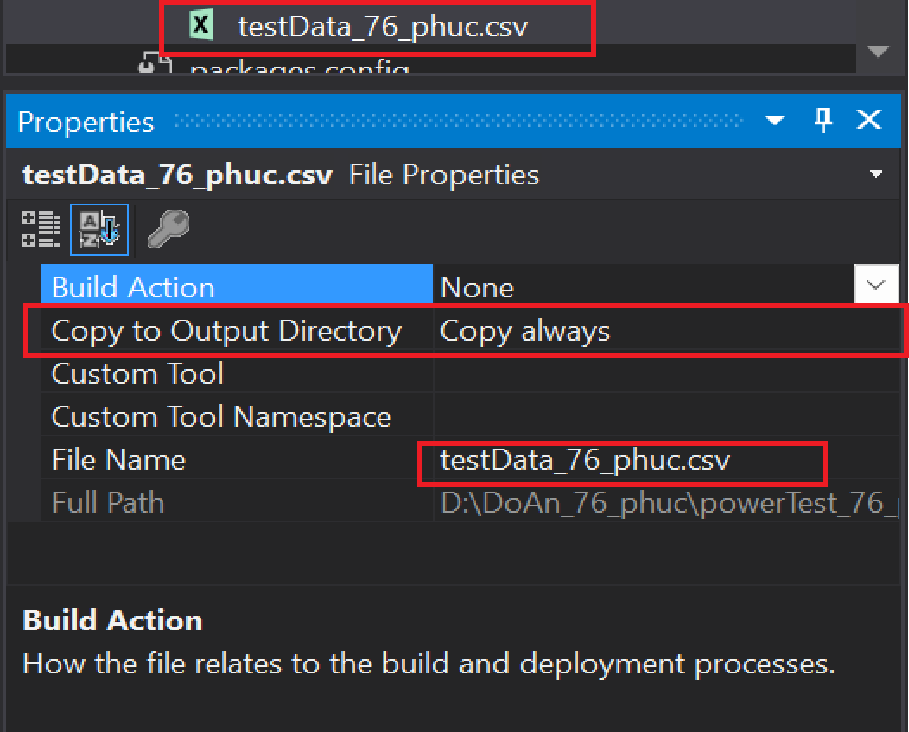
**Tạo tập tin csv trong file Data vừa tạo**

****

#### 2.Thay đổi thuộc tính cho file csv

**Chuột phải tập tin csv chọn Properties**

**Thiết lập thuộc tính “copy to output dierectory” thành “copy always”**

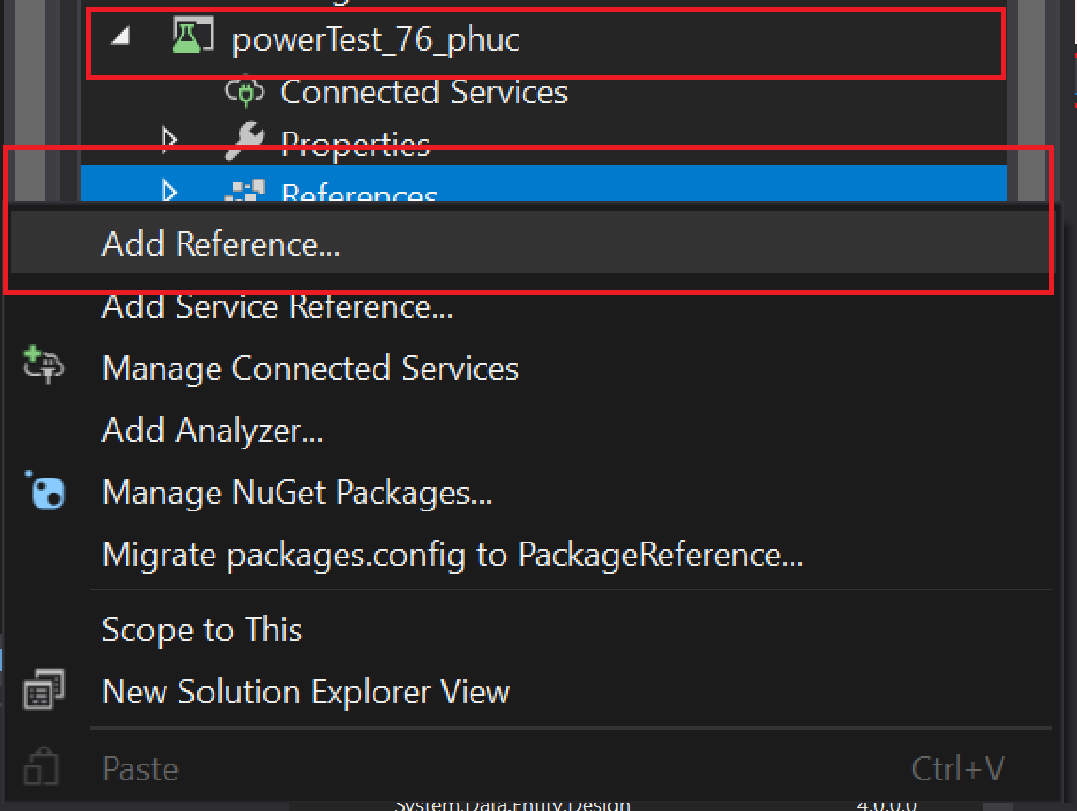
****

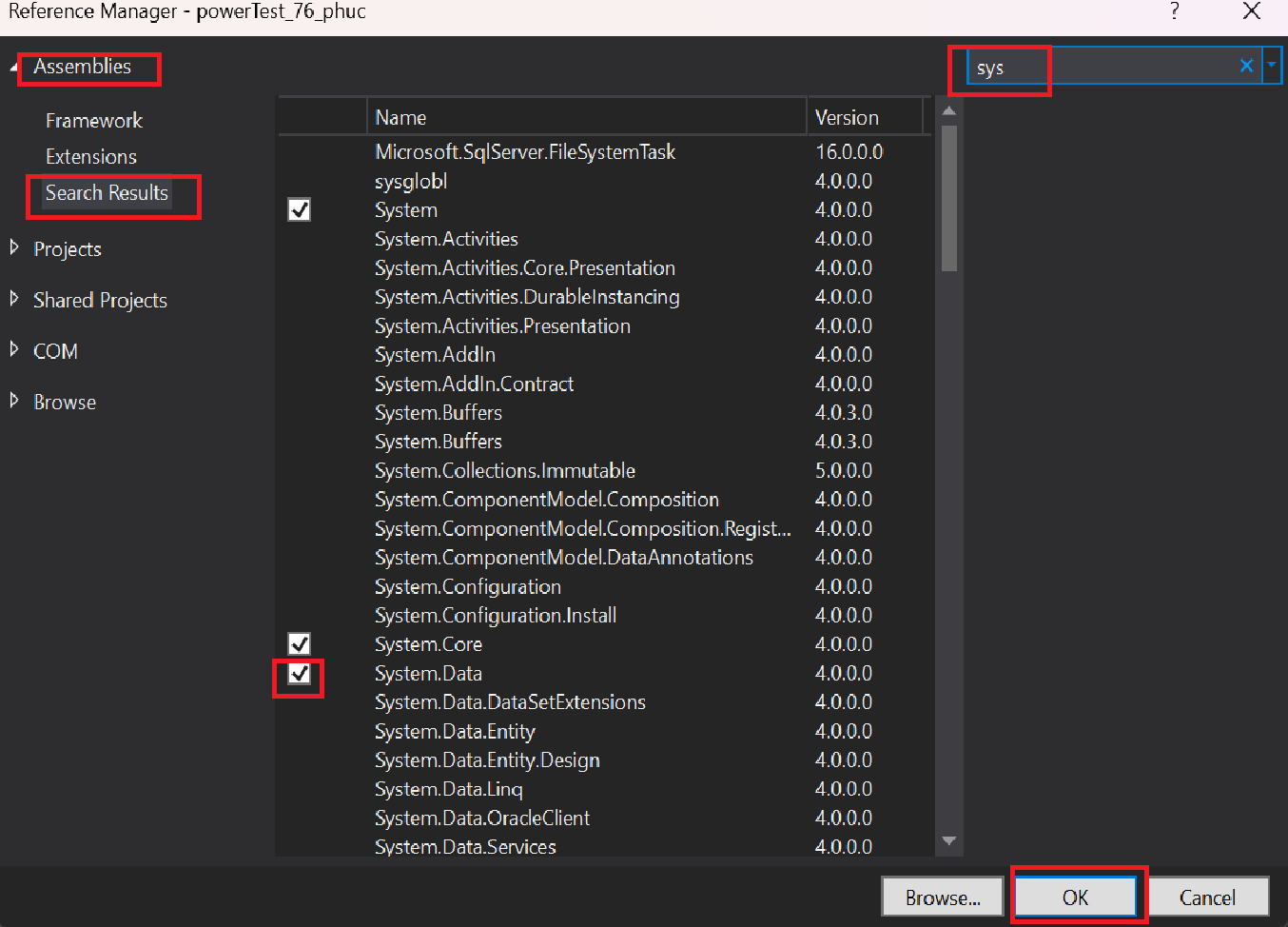
#### 3.Nhập dữ liệu test vào tập tin csv

A screenshot of a computer

AI-generated content may be incorrect.

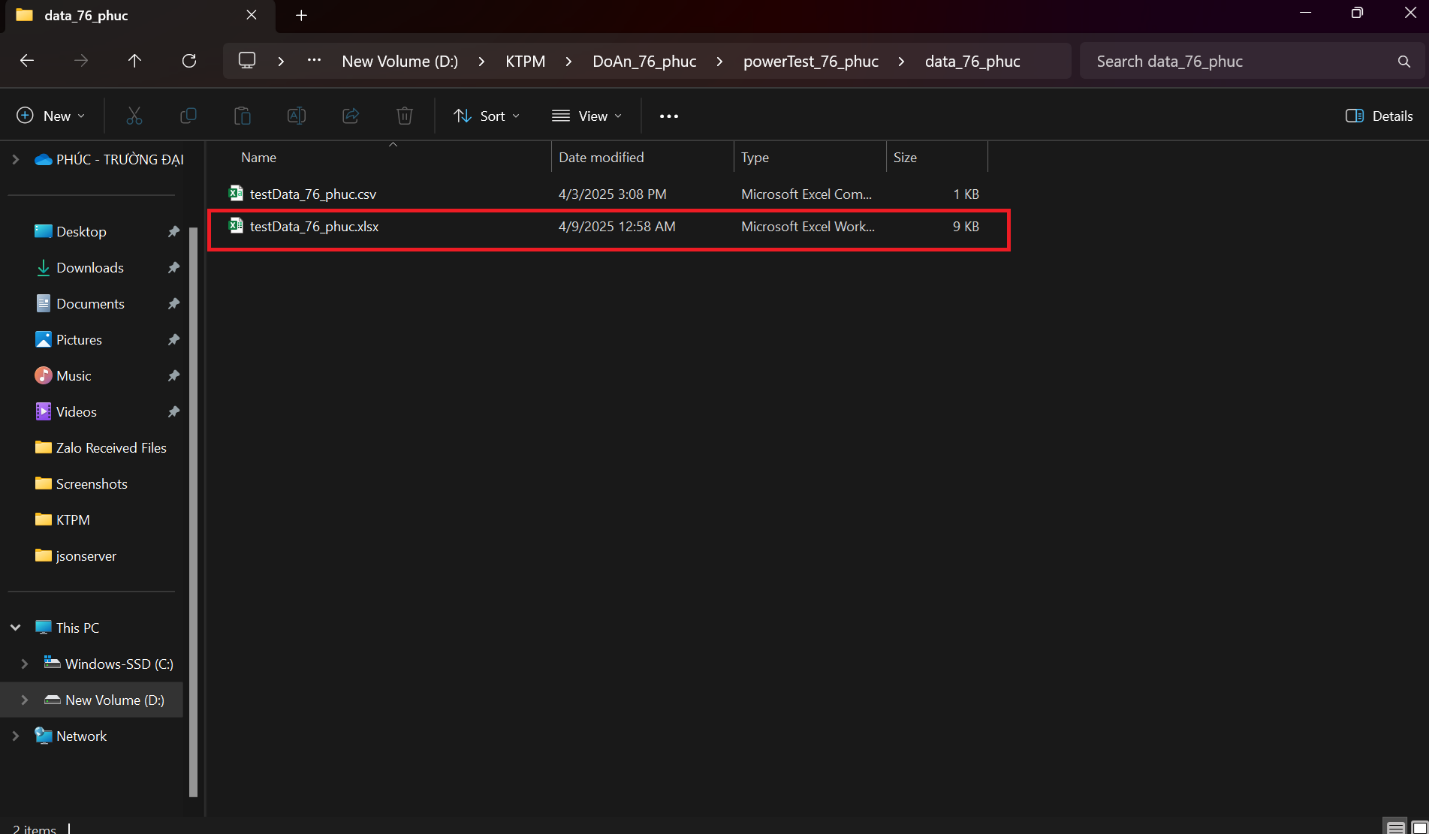
**Thêm Reference System.Data vào project test**





#### 4.Kiểm thử bằng tập tin xlsx

Tạo file xlsx trong file data\_76\_phuc



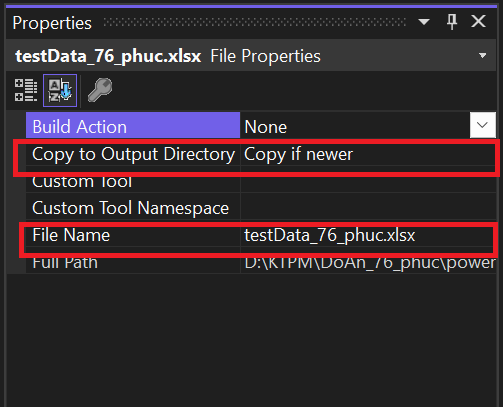
**Tạo dự liệu**

**A screenshot of a spreadsheet

AI-generated content may be incorrect.**

**Chuột phải tập tin csv chọn Properties**

**Thiết lập thuộc tính “copy to output dierectory” thành “copy if newer”**

****

Chú ý: Cần đảm bảo rằng hệ thống có Microsoft Access Database Engine (ACE.OLEDB). Nếu chưa có, hãy tải và cài đặt: [Download Microsoft Access Database Engine](https://www.microsoft.com/en-us/download/details.aspx?id=54920)

### Bước 4. Viết code kiểm thử phương thức

**Trong file UnitTest viết đoạn code có dùng data source và không dùng data source sau để kiểm thử các test case**

using Microsoft.VisualStudio.TestTools.UnitTesting;

using System;

using DoAn\_76\_phuc;

namespace powerTest\_76\_phuc

{

[TestClass]

public class UnitTest1

{

public TestContext TestContext { get; set; }//tạo đối tượng TestContext dùng để đọc dữ liệu

[TestMethod]

public void TC1\_xLonHon0\_76\_phuc()//test case với x và n là số nguyên dương

{

int n\_76\_phuc;

double actual\_76\_phuc;

double x\_76\_phuc, expected\_76\_phuc;

n\_76\_phuc = 2;

x\_76\_phuc = 2;

expected\_76\_phuc = 4;// kết quả mong muốn

actual\_76\_phuc = Power.power\_76\_phuc(x\_76\_phuc, n\_76\_phuc);// kết quả thực tế

Assert.AreEqual(expected\_76\_phuc, actual\_76\_phuc);// so sánh kết quả mong muốn và kết quả thực tế

}

[TestMethod]

public void TC2\_xNhoHon0\_76\_phuc()//test case với x là số nguyên âm n là số bất kì

{

int n\_76\_phuc;

double actual\_76\_phuc;

double x\_76\_phuc, expected\_76\_phuc;

n\_76\_phuc = 4;

x\_76\_phuc = -3;

expected\_76\_phuc = 81;

actual\_76\_phuc = Power.power\_76\_phuc(x\_76\_phuc, n\_76\_phuc);

Assert.AreEqual(expected\_76\_phuc, actual\_76\_phuc);

}

[TestMethod]

public void TC3\_nBang0\_76\_phuc()//với số mũ bằng 0,x là số bất kì

{

int n\_76\_phuc;

double actual\_76\_phuc;

double x\_76\_phuc, expected\_76\_phuc;

n\_76\_phuc = 0;

x\_76\_phuc = -3;

expected\_76\_phuc = 1;

actual\_76\_phuc = Power.power\_76\_phuc(x\_76\_phuc, n\_76\_phuc);

Assert.AreEqual(expected\_76\_phuc, actual\_76\_phuc);

}

[TestMethod]

public void TC4\_nNhoHon0\_76\_phuc()//với n âm, x là số bất kì

{

int n\_76\_phuc;

double actual\_76\_phuc;

double x\_76\_phuc, expected\_76\_phuc;

n\_76\_phuc = -2;

x\_76\_phuc = 5;

expected\_76\_phuc = 0.04;

actual\_76\_phuc = Power.power\_76\_phuc(x\_76\_phuc, n\_76\_phuc);

Assert.AreEqual(expected\_76\_phuc, actual\_76\_phuc);

}

[TestMethod]

public void TC5\_xBang0\_76\_phuc()//với x = 0, n là số bất kì

{

int n\_76\_phuc;

double actual\_76\_phuc;

double x\_76\_phuc, expected\_76\_phuc;

n\_76\_phuc = 4;

x\_76\_phuc = 0;

expected\_76\_phuc = 0;

actual\_76\_phuc = Power.power\_76\_phuc(x\_76\_phuc, n\_76\_phuc);

Assert.AreEqual(expected\_76\_phuc, actual\_76\_phuc);

}

[TestMethod]

public void TC6\_xLaSoThuc\_76\_phuc()//với x là số thực, n là số bất kì

{

int n\_76\_phuc;

double actual\_76\_phuc;

double x\_76\_phuc, expected\_76\_phuc;

n\_76\_phuc = 2;

x\_76\_phuc = 2.5;

expected\_76\_phuc = 6.25;

actual\_76\_phuc = Power.power\_76\_phuc(x\_76\_phuc, n\_76\_phuc);

Assert.AreEqual(expected\_76\_phuc, actual\_76\_phuc);

}

[TestMethod]

[ExpectedException(typeof(DivideByZeroException))]// với x=0, n<0 sẽ không ném ra ngoại lệ vì code sai ở trường hợp 3, trong c# mẫu bằng 0 kết quả sẽ ra vô cực

public void TC7\_xBang0VanNhoHon0\_76\_phuc()

{

Power.power\_76\_phuc(0, -1);

}

//Test với file csv

[DataSource("Microsoft.VisualStudio.TestTools.DataSource.CSV", @".\data\_76\_phuc\testData\_76\_Phuc.csv", "testData\_76\_Phuc#csv", DataAccessMethod.Sequential)]//đường dẫn đến data source file csv

[TestMethod]

public void TestWithDataSourceCsv\_76\_Phuc()

{

int n\_76\_phuc;

double actual\_76\_phuc;

double x\_76\_phuc, expected\_76\_phuc;

n\_76\_phuc = int.Parse(TestContext.DataRow[1].ToString());// lấy giá trị n từ file csv

x\_76\_phuc = double.Parse(TestContext.DataRow[0].ToString());//lấy giá trị x từ file csv

expected\_76\_phuc = double.Parse(TestContext.DataRow[2].ToString());//lấy kết quả mong muốn từ file csv

actual\_76\_phuc = Power.power\_76\_phuc(x\_76\_phuc, n\_76\_phuc);

Assert.AreEqual(expected\_76\_phuc, actual\_76\_phuc);

}

//Test với file Excel

[DataSource("System.Data.OleDb", "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=|DataDirectory|\\data\_76\_phuc\\testData\_76\_phuc.xlsx;Extended Properties='Excel 12.0 Xml;HDR=YES';", "Sheet1$",

DataAccessMethod.Sequential)]

[TestMethod]

public void TestWithExcelDataSourceXlsx\_76\_Phuc()

{

int n\_76\_phuc = int.Parse(TestContext.DataRow[1].ToString());

double x\_76\_phuc = double.Parse(TestContext.DataRow[0].ToString());

double expected\_76\_phuc = double.Parse(TestContext.DataRow[2].ToString());

double actual\_76\_phuc = Power.power\_76\_phuc(x\_76\_phuc, n\_76\_phuc);

Assert.AreEqual(expected\_76\_phuc, actual\_76\_phuc);

}

}

}

## 4.Kết quả

A screenshot of a computer

AI-generated content may be incorrect.

## 5.Test Case.

### Đặc tả yêu cầu

* Chương trình sẽ nhập nhập vào giá trị của hai biến x và n sau đó dựa vào thuật toán để cho ra kết quả.
* Dữ liệu đầu vào: x và n.
* Không được để trống các ô dữ liệu.
* Giá trị trả về của ô kết quả : Giá trị của x mũ n hoặc “Vô Cực”(tức là lỗi của thuật toán).

### Bảng Test Case

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **TC\_id** | **x** | **n** | **expected** | **actual** | **Pass/Fail** |
| TC1\_xLonHon0\_76\_phuc | 5 | 4 | 625 | 625 | Pass |
| TC2\_xNhoHon0\_76\_phuc | -2 | 3 | -8 | -8 | Pass |
| TC3\_nBang0\_76\_phuc | 4 | 0 | 1 | 1 | Pass |
| TC4\_nNhoHon0\_76\_phuc | 5 | -2 | 0.04 | 0.04 | Pass |
| TC5\_xBang0\_76\_phuc | 0 | 2 | 0 | 0 | Pass |
| TC6\_xLaSoThuc\_76\_phuc | 0.5 | -3 | 8 | 8 | Pass |
| TC7\_xBang0VanNhoHon0\_76\_phuc | 0 | -1 | Lỗi |  | Fail |

# WEB DRIVE-Facebook

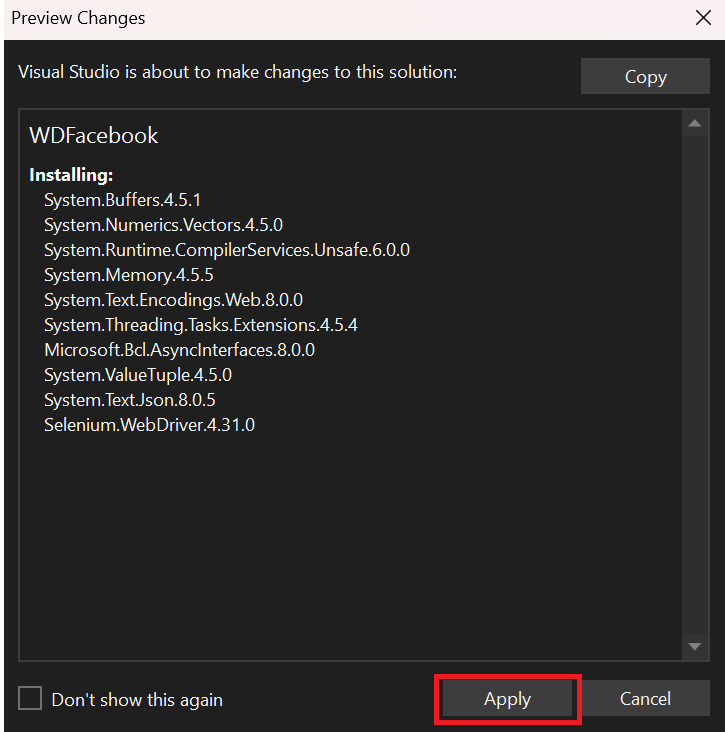
## Cài đặt Selenium Web Driver( Selenium.WebDriver và Selenium.WebDriver.ChromeDriver)

-Tại project C#, chọn menu Tools > NuGet Package Manager > Manage NuGet Packages for So lution…A screenshot of a computer

AI-generated content may be incorrect.

- Click tab Browse và tìm kiếm từ khóa Selenium và các web driver muốn kiểm thử để cài đặt. A screenshot of a computer

AI-generated content may be incorrect.



## 2.Thiết kế giao diện xử lý các chức năng.

A screenshot of a computer

AI-generated content may be incorrect.

## 3.Code thực hiện chức năng.

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.IO;

using System.Threading;

using OpenQA.Selenium;

using OpenQA.Selenium.Chrome;

using System.Windows.Forms;

using OpenQA.Selenium.Interactions;

using OpenQA.Selenium.Support.UI;

using SeleniumExtras.WaitHelpers;

namespace WDFacebook

{

public partial class Form1 : Form

{

private IWebDriver driverFacebook\_76\_phuc = null;

public object SeleniumExtras { get; private set; }

public Form1()

{

InitializeComponent();

}

// Khởi tạo trình duyệt Chrome nếu chưa có

private void InitializeDriverFacebook()

{

if (driverFacebook\_76\_phuc == null)

{

ChromeDriverService chromeService = ChromeDriverService.CreateDefaultService();

chromeService.HideCommandPromptWindow = true;

ChromeOptions options = new ChromeOptions();

options.AddArgument("--start-maximized");

driverFacebook\_76\_phuc = new ChromeDriver(chromeService, options);

}

}

// Hàm đăng nhập Facebook

private void LoginFacebook\_76\_phuc(string email\_76\_phuc, string password\_76\_phuc)

{

if (string.IsNullOrEmpty(email\_76\_phuc) || string.IsNullOrEmpty(password\_76\_phuc))

{

MessageBox.Show("Vui lòng nhập email và mật khẩu!", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

InitializeDriverFacebook();

driverFacebook\_76\_phuc.Navigate().GoToUrl("https://www.facebook.com/");

Thread.Sleep(3000);

try

{

// Điền thông tin đăng nhập và bấm nút login

driverFacebook\_76\_phuc.FindElement(By.Id("email")).SendKeys(email\_76\_phuc);

driverFacebook\_76\_phuc.FindElement(By.Id("pass")).SendKeys(password\_76\_phuc);

driverFacebook\_76\_phuc.FindElement(By.Name("login")).Click();

}

catch (NoSuchElementException)

{

MessageBox.Show("Không thể đăng nhập Facebook!", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

// Gửi lời mời kết bạn đến một profile cụ thể

private void SendFriendRequest\_76\_phuc(string profileUrl\_76\_phuc)

{

//InitializeDriverFacebook();

//string username\_76\_phuc = "plehoang641@gmail.com";

//string password\_76\_phuc = "Asdjklzxc..";

//LoginFacebook\_76\_phuc(username\_76\_phuc, password\_76\_phuc);

driverFacebook\_76\_phuc.Navigate().GoToUrl(profileUrl\_76\_phuc);

Thread.Sleep(5000);

try

{

// Tìm nút "Thêm bạn bè" và bấm vào

IWebElement addFriendButton = driverFacebook\_76\_phuc.FindElement(By.XPath("//div[@aria-label='Thêm bạn bè']"));

addFriendButton.Click();

}

catch (NoSuchElementException)

{

MessageBox.Show("Không tìm thấy nút 'Thêm bạn bè'!", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

// Bình luận vào một bài viết theo link

private void CommentOnPost\_76\_phuc(string postUrl\_76\_phuc, string comment\_76\_phuc)

{

driverFacebook\_76\_phuc.Navigate().GoToUrl(postUrl\_76\_phuc);

Thread.Sleep(5000);

try

{

// Dùng JavaScript để tương tác nâng cao với phần tử (ví dụ như focus vào phần tử)

IJavaScriptExecutor js = (IJavaScriptExecutor)driverFacebook\_76\_phuc;

// Scroll nhẹ xuống để chắc chắn render phần comment

js.ExecuteScript("window.scrollBy(0, 300);");

Thread.Sleep(1000);

// Tìm phần tử là ô viết bình luận – có role='textbox' và aria-label chứa 'Viết bình luận'

// Đây là phần tử dạng <div contenteditable="true">, không phải <textarea> hay <input>

IWebElement commentBox = driverFacebook\_76\_phuc.FindElement(

By.XPath("//div[@role='textbox' and contains(@aria-label, 'Viết bình luận')]"));

// Dùng JavaScript để focus vào ô bình luận (vì Click() có thể bị lỗi nếu bị che)

js.ExecuteScript("arguments[0].focus();", commentBox);

commentBox.Click();

Thread.Sleep(500);

commentBox.SendKeys(comment\_76\_phuc);

Thread.Sleep(500);

commentBox.SendKeys(OpenQA.Selenium.Keys.Enter);

}

catch (NoSuchElementException)

{

MessageBox.Show("Không tìm thấy ô bình luận trên trang!", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi comment: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void Button1\_Click(object sender, EventArgs e)

{

//string username\_76\_phuc = "plehoang641@gmail.com";

//string password\_76\_phuc = "Asdjklzxc..";

//LoginFacebook\_76\_phuc(username\_76\_phuc, password\_76\_phuc);

string username\_76\_phuc;

string password\_76\_phuc;

username\_76\_phuc = txtusername.Text;

password\_76\_phuc = txtpass.Text;

LoginFacebook\_76\_phuc(username\_76\_phuc, password\_76\_phuc);

}

private void Form1\_Load(object sender, EventArgs e)

{

}

private void label7\_Click(object sender, EventArgs e)

{

}

private void button1\_Click\_1(object sender, EventArgs e)

{

string linkpost\_76\_phuc;

string cmt\_76\_phuc;

linkpost\_76\_phuc = txtlinkcmt.Text;

cmt\_76\_phuc = txtcmt.Text;

CommentOnPost\_76\_phuc(linkpost\_76\_phuc,cmt\_76\_phuc);

}

private void button2\_Click(object sender, EventArgs e)

{

String link\_76\_phuc;

link\_76\_phuc= txtprofile.Text;

SendFriendRequest\_76\_phuc(link\_76\_phuc);

}

private void pictureBox1\_Click(object sender, EventArgs e)

{

}

}

}

## 4.Các Chức năng.

### 1.Chức năng đăng nhập.

private void LoginFacebook\_76\_phuc(string email\_76\_phuc, string password\_76\_phuc)

{

if (string.IsNullOrEmpty(email\_76\_phuc) || string.IsNullOrEmpty(password\_76\_phuc))

{

MessageBox.Show("Vui lòng nhập email và mật khẩu!", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

InitializeDriverFacebook();

driverFacebook\_76\_phuc.Navigate().GoToUrl("https://www.facebook.com/");

Thread.Sleep(3000);

try

{

// Điền thông tin đăng nhập và bấm nút login

driverFacebook\_76\_phuc.FindElement(By.Id("email")).SendKeys(email\_76\_phuc);

driverFacebook\_76\_phuc.FindElement(By.Id("pass")).SendKeys(password\_76\_phuc);

driverFacebook\_76\_phuc.FindElement(By.Name("login")).Click();

}

catch (NoSuchElementException)

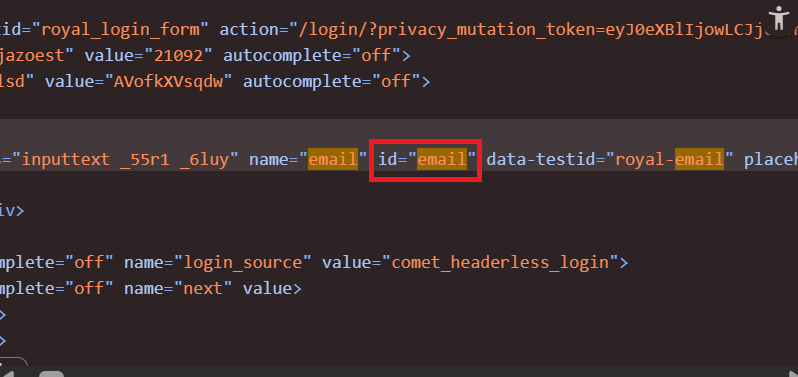
{

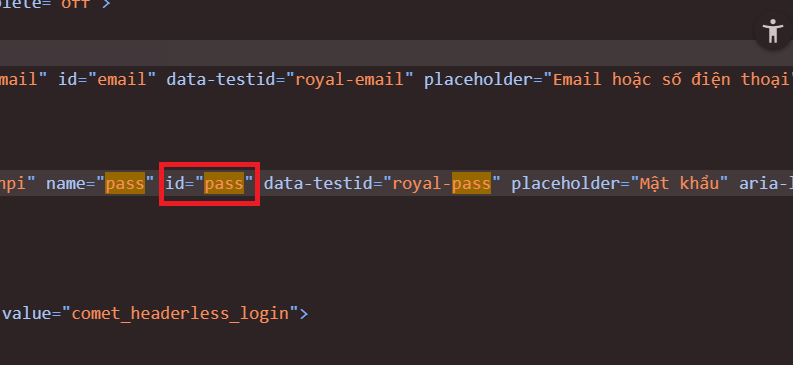
MessageBox.Show("Không thể đăng nhập Facebook!", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

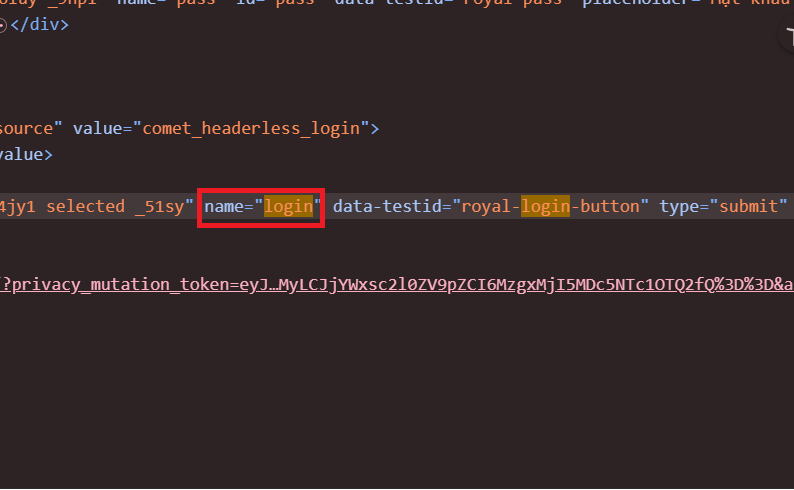
}

}

**Tìm và sử dụng các element sau:**







#### 1.1 Mô tả chức năng.

- Chức năng đăng nhập cho phép người dùng đã đăng ký truy cập vào tài khoản cá nhân của họ trên Facebook để sử dụng các dịch vụ như xem bài đăng,gửi lời mời kết bạn, bình luận, v.v.

#### 1.2 Test Case.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TC\_ID** | **Tên test case** | **username\_76\_phuc** | **password\_76\_phuc** | **output** |
| 1 | us\_mk\_dung | plehoang641@gmail.com | Asdjklzxc.. | Pass |
| 2 | us\_sai | lehoangphuc@gmail.com | Asdjklzxc.. | Fail |
| 3 | mk\_sai | plehoang641@gmail.com | 67896789 | Fail |

### 2.Chức năng gửi lời mời kết bạn.

private void SendFriendRequest\_76\_phuc(string profileUrl\_76\_phuc)

{

//InitializeDriverFacebook();

//string username\_76\_phuc = "plehoang641@gmail.com";

//string password\_76\_phuc = "Asdjklzxc..";

//LoginFacebook\_76\_phuc(username\_76\_phuc, password\_76\_phuc);

driverFacebook\_76\_phuc.Navigate().GoToUrl(profileUrl\_76\_phuc);

Thread.Sleep(5000);

try

{

// Tìm nút "Thêm bạn bè" và bấm vào

IWebElement addFriendButton = driverFacebook\_76\_phuc.FindElement(By.XPath("//div[@aria-label='Thêm bạn bè']"));

addFriendButton.Click();

}

catch (NoSuchElementException)

{

MessageBox.Show("Không tìm thấy nút 'Thêm bạn bè'!", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

#### 2.1 Mô tả chức năng.

- Chức năng cho phép người dùng gửi lời mời kết bạn đến người dùng khác trên Facebook để xem bài đăng, story, reel và các cập nhật khác của họ.

#### 2.2 Test Case.

|  |  |  |  |
| --- | --- | --- | --- |
| **TC\_ID** | **Tên test case** | **profileUrl\_76\_phuc** | **output** |
| 1 | link\_dung | https://www.facebook.com/le.h.phuc.0602 | Pass |
| 2 | link\_sai | https://www.youtube.com/ | Fail |
| 3 | khong phai link | kkhgffhhjhjfjfhggcg | Fail |

### 3.Chức năng bình luận bài viết.

private void CommentOnPost\_76\_phuc(string postUrl\_76\_phuc, string comment\_76\_phuc)

{

driverFacebook\_76\_phuc.Navigate().GoToUrl(postUrl\_76\_phuc);

Thread.Sleep(5000);

try

{

// Dùng JavaScript để tương tác nâng cao với phần tử (ví dụ như focus vào phần tử)

IJavaScriptExecutor js = (IJavaScriptExecutor)driverFacebook\_76\_phuc;

// Scroll nhẹ xuống(mỗi lẫn 300px) để chắc chắn render phần comment

js.ExecuteScript("window.scrollBy(0, 300);");

Thread.Sleep(1000);

// Tìm phần tử là ô viết bình luận – có role='textbox' và aria-label chứa 'Viết bình luận'

// Đây là phần tử dạng <div contenteditable="true">, không phải <textarea> hay <input>

IWebElement commentBox = driverFacebook\_76\_phuc.FindElement(

By.XPath("//div[@role='textbox' and contains(@aria-label, 'Viết bình luận')]"));

// Dùng JavaScript để focus vào ô bình luận (vì Click() có thể bị lỗi nếu bị che)

js.ExecuteScript("arguments[0].focus();", commentBox);

commentBox.Click();

Thread.Sleep(500);

commentBox.SendKeys(comment\_76\_phuc);

Thread.Sleep(500);

commentBox.SendKeys(OpenQA.Selenium.Keys.Enter);

}

catch (NoSuchElementException)

{

MessageBox.Show("Không tìm thấy ô bình luận trên trang!", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi comment: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

#### 3.1 Mô tả chức năng.

- Chức năng bình luận cho phép người dùng để lại ý kiến, cảm xúc hoặc phản hồi trực tiếp dưới các bài viết (ảnh hoặc video) của người khác hoặc của chính mình. Đây là một phần quan trọng trong hệ thống tương tác mạng xã hội.

#### 3.2 Test Case.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TC\_ID** | **Tên test case** | **postUrl\_76\_phuc** | **comment\_76\_phuc** | **output** |
| 1 | link\_content | https://www.facebook.com/thongtinchinhphu/posts/pfbid0qYsRTj8mWDwWRMuzYXpbu52fUjW4fVtFxjT4nsPHGAAhvauqJvm81spwMVURKyP2l | aaaaaa | Pass |
| 2 | link\_sai | https://www.instagram.com/ | Asdjklzxc.. | Fail |
| 3 | no\_content | https://www.facebook.com/thongtinchinhphu/posts/pfbid0qYsRTj8mWDwWRMuzYXpbu52fUjW4fVtFxjT4nsPHGAAhvauqJvm81spwMVURKyP2l |  | Fail |

# API POSTMAN

## 1.API Json server.

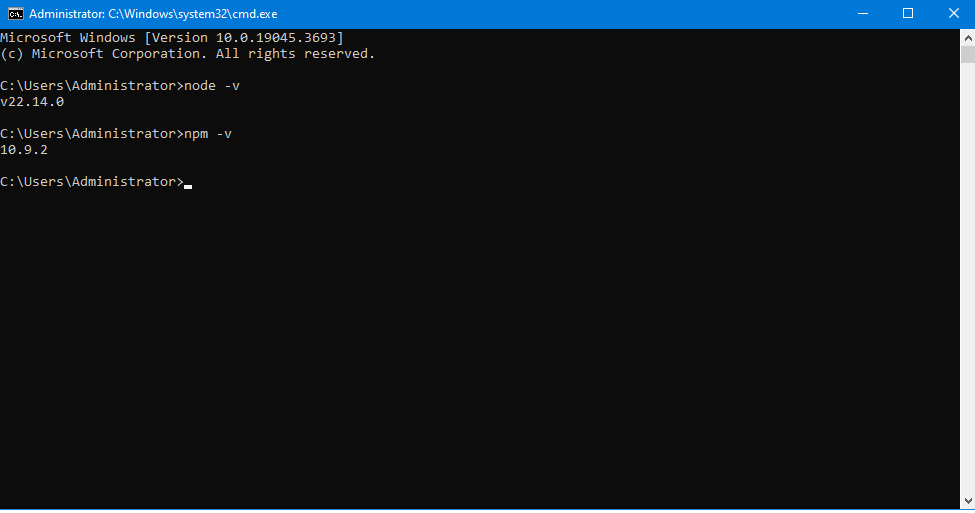
### 1. Cài đặt Node.js

Tải và cài đặt node.js link: [Node.js](https://nodejs.org/en/download)



Sau khi tải ở trang nhận được file này: next tới rồi finish là cài đặt thành công node.js

Sau khi cài đặt xong có thể check phiên bản của node – v dùng để kiểm tra phiên bản của máy và npm -v dùng để kiểm tra nm của máy (trình quản lý gói cho Node.js)



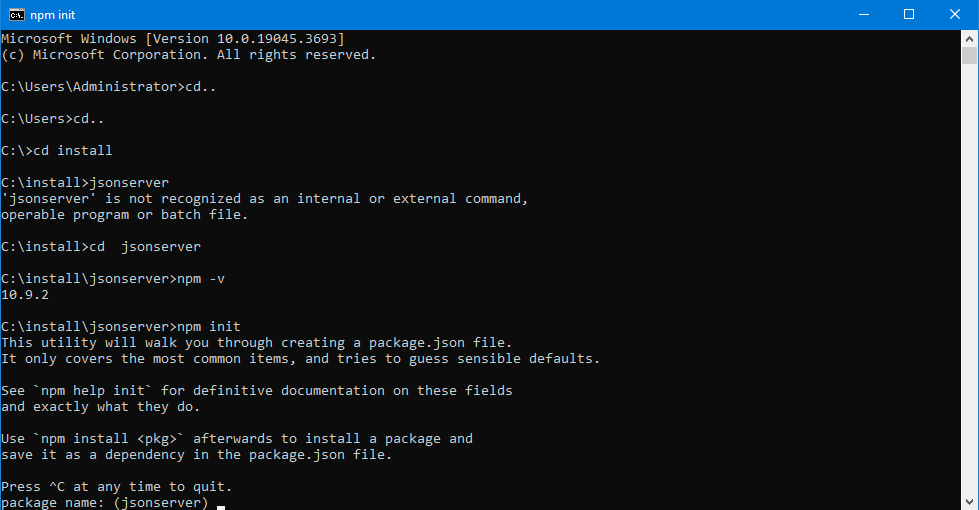
### 2.Cài đặt Json server.

Có thể dùng lệnh: npm install -g json-server

* Cài đặt JSON Server trên hệ thống của bạn một cách toàn cục (-g = global).
* JSON Server là một công cụ giúp tạo một REST API giả lập nhanh chóng từ một file JSON.

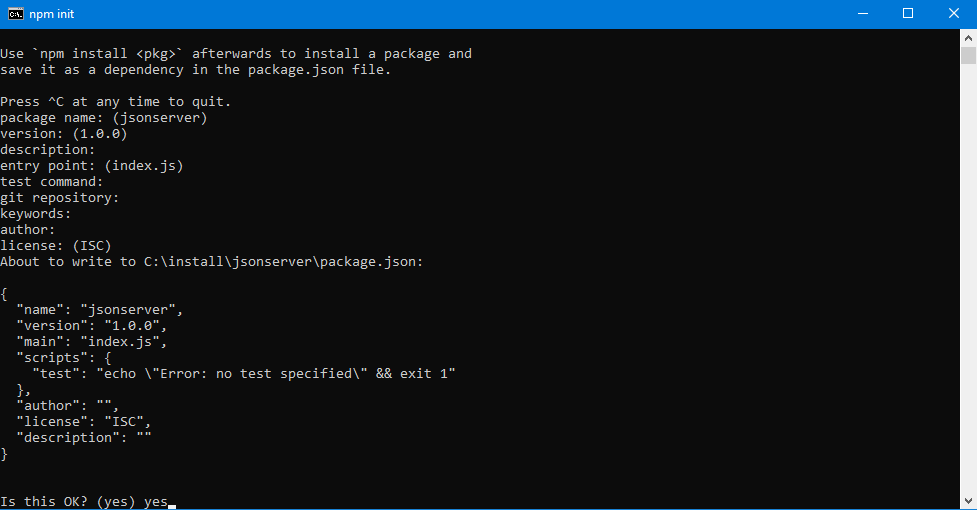
Hoặc có thể chỉ cài lên 1 folder: npm init

Chuyển tới thư mục cần tạo json server sau đó chạy lện npm init

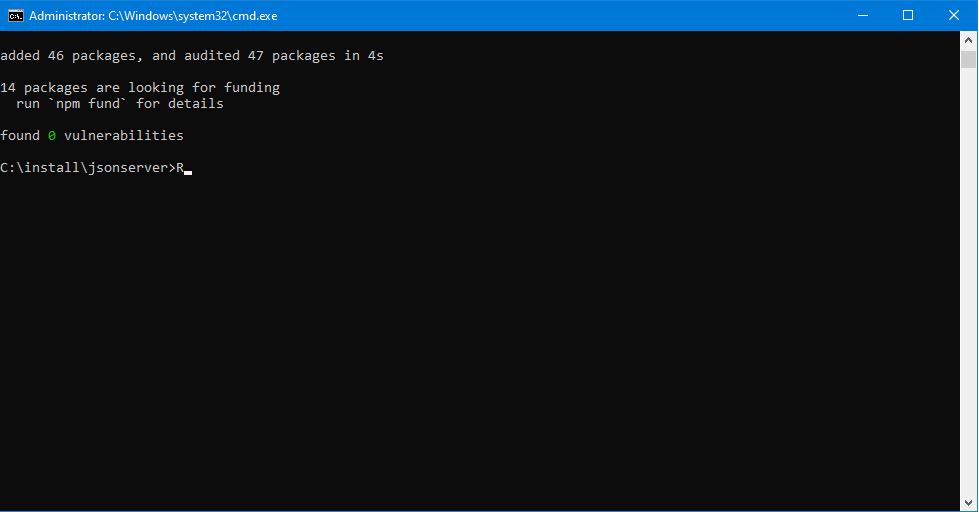


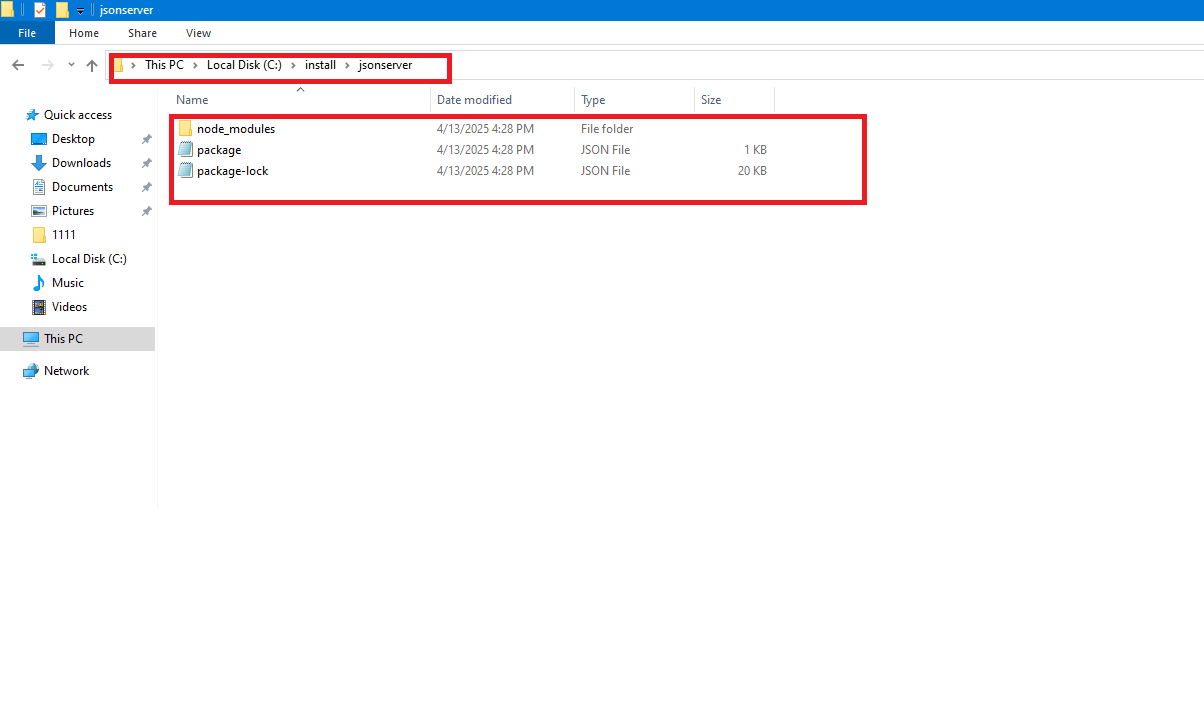
enter đến cuối chọn yes: ( để tạo 1 project)

Lệnh này sẽ tạo một package.json với các giá trị mặc định.



Sau đó gõ : npm install i json-server





* Giải thích về json

JSON là từ viết tắt của JavaScript Object Notation - Có thể hiểu là "Kí hiệu

Object trong Javascript".

JSON là một định dạng trao đổi dữ liệu chuẩn , nó là một ngôn ngữ độc lập và hỗ trợ các

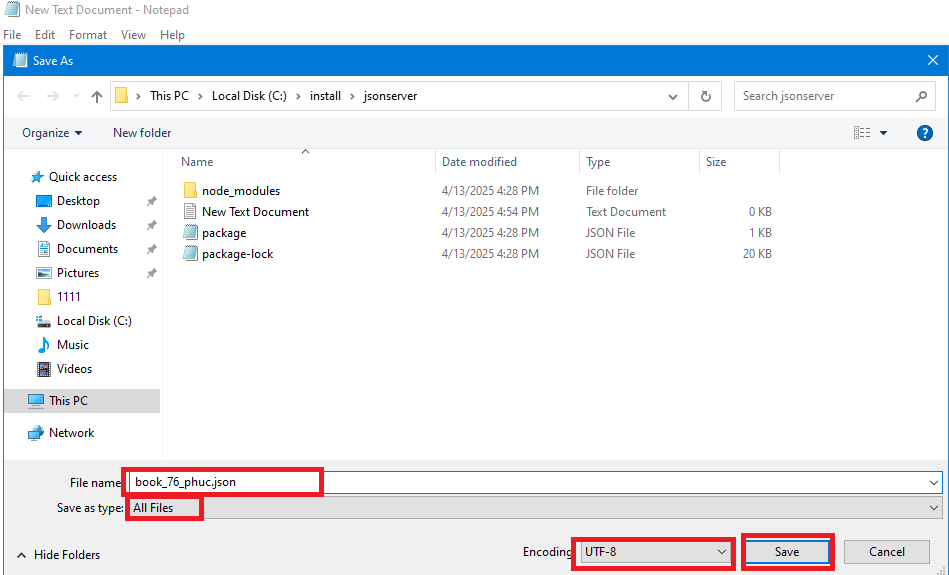
cấu trúc dữ liệu như mảng và đối tượng.

Nếu bạn đã biết về Object Javascript thì ví dụ về JSON sau đây cũng chính là để biểu diễn

cho một Object trong Js.

### 3.Tạo file json cho server ảo.

Tạo file json bằng Notepad.



Sau đó khi lưu cho phép lưu All file và Encoding là UTF-8 sau đó ấn save để lưu

Viết đoạn code này vào file vừa tạo.

{

"thuvien": {

"name": "Thư viện Tri thức Mở",

"address": "234 Đường Trần Hưng Đạo, Quận Hoàn Kiếm, Hà Nội",

"email": "thuvien@trithucmo.vn",

"phone": "024-8765-4321",

"establishmentYear": 2010,

"openingHours": {

"mondayToFriday": "8:00 - 21:00",

"saturday": "8:00 - 17:00",

"sunday": "9:00 - 16:00"

},

"website": "http://locallhost/ ",

"statistics": {

"totalBooks": 45678,

"totalMembers": 12540,

"borrowedBooks": 3254,

"newBooksThisMonth": 123

}

},

"loaisach": [

{

"code": "VHVN",

"name": "Văn học Việt Nam",

"location": "Khu A - Tầng 1",

"bookCount": 5623,

"id": "33e1"

},

{

"code": "VHNN",

"name": "Văn học nước ngoài",

"location": "Khu A - Tầng 2",

"bookCount": 7845,

"id": "7ffb"

},

{

"code": "KHTN",

"name": "Khoa học tự nhiên",

"location": "Khu B - Tầng 1",

"bookCount": 4532,

"id": "fee5"

},

{

"code": "CNTT",

"name": "Công nghệ thông tin",

"location": "Khu B - Tầng 2",

"bookCount": 3654,

"id": "10b2"

},

{

"code": "KHXH",

"name": "Khoa học xã hội",

"location": "Khu C - Tầng 1",

"bookCount": 6234,

"id": "3cfe"

}

],

"sach": [

{

"code": "S001",

"title": "Truyện Kiều",

"author": "Nguyễn Du",

"publishYear": 2020,

"publisher": "NXB Văn học",

"category": "VHVN",

"location": "A1-KE05-012",

"description": "Truyện Kiều là một truyện thơ của đại thi hào Nguyễn Du, được viết bằng chữ Nôm theo thể lục bát, gồm 3254 câu.",

"totalCopies": 5,

"availableCopies": 3,

"ISBN": "978-604-62-8574-3",

"image": "truyen\_kieu.jpg",

"rating": 4.8,

"borrowCount": 245,

"id": "6800"

},

{

"code": "S002",

"title": "Nhà giả kim",

"author": "Paulo Coelho",

"publishYear": 2019,

"publisher": "NXB Hội Nhà Văn",

"category": "VHNN",

"location": "A2-KE12-034",

"description": "Nhà giả kim là tiểu thuyết được xuất bản lần đầu ở Brasil năm 1988, và là cuốn sách nổi tiếng nhất của nhà văn Paulo Coelho.",

"totalCopies": 8,

"availableCopies": 2,

"ISBN": "978-604-94-5411-7",

"image": "nha\_gia\_kim.jpg",

"rating": 4.7,

"borrowCount": 367,

"id": "18bb"

},

{

"code": "S003",

"title": "Vũ trụ trong vỏ hạt dẻ",

"author": "Stephen Hawking",

"publishYear": 2021,

"publisher": "NXB Trẻ",

"category": "KHTN",

"location": "B1-KE03-045",

"description": "Vũ trụ trong vỏ hạt dẻ là cuốn sách khoa học phổ thông của nhà vật lý lý thuyết Stephen Hawking.",

"totalCopies": 6,

"availableCopies": 4,

"ISBN": "978-604-1-12345-6",

"image": "vu\_tru\_trong\_vo\_hat\_de.jpg",

"rating": 4.9,

"borrowCount": 183,

"id": "c1b2"

},

{

"code": "S004",

"title": "Clean Code",

"author": "Robert C. Martin",

"publishYear": 2022,

"publisher": "NXB Bách khoa Hà Nội",

"category": "CNTT",

"location": "B2-KE08-023",

"description": "Clean Code là cuốn sách về lập trình giúp các lập trình viên viết mã nguồn dễ đọc, dễ bảo trì và nâng cao hiệu quả làm việc.",

"totalCopies": 4,

"availableCopies": 1,

"ISBN": "978-604-2-67890-1",

"image": "clean\_code.jpg",

"rating": 4.8,

"borrowCount": 214,

"id": "10f8"

},

{

"code": "S005",

"title": "Sapiens: Lược sử loài người",

"author": "Yuval Noah Harari",

"publishYear": 2018,

"publisher": "NXB Thế giới",

"category": "KHXH",

"location": "C1-KE04-078",

"description": "Sapiens là một cuốn sách về lịch sử loài người từ thời kỳ đồ đá cũ đến thế kỷ 21.",

"totalCopies": 7,

"availableCopies": 3,

"ISBN": "978-604-77-3456-7",

"image": "sapiens.jpg",

"rating": 4.9,

"borrowCount": 328,

"id": "42be"

}

],

"hoivien": [

{

"code": "TV001",

"fullName": "Nguyễn Văn Minh",

"birthDate": "1995-06-12",

"email": "minh.nguyen@gmail.com",

"phone": "0912345678",

"address": "56 Trần Phú, Hà Đông, Hà Nội",

"cardType": "Thường niên",

"registrationDate": "2020-03-15",

"expiryDate": "2025-03-15",

"status": "Hoạt động",

"currentlyBorrowedCount": 2,

"borrowHistory": [

{

"bookCode": "S001",

"borrowDate": "2024-02-10",

"dueDate": "2024-03-10",

"returnDate": "2024-03-05",

"status": "Đã trả"

},

{

"bookCode": "S003",

"borrowDate": "2024-03-15",

"dueDate": "2024-04-15",

"returnDate": null,

"status": "Đang mượn"

},

{

"bookCode": "S005",

"borrowDate": "2024-03-15",

"dueDate": "2024-04-15",

"returnDate": null,

"status": "Đang mượn"

}

],

"id": "4ce8"

},

{

"code": "TV002",

"fullName": "Trần Thị Hoa",

"birthDate": "1990-08-25",

"email": "hoa.tran@outlook.com",

"phone": "0987654321",

"address": "72 Nguyễn Trãi, Thanh Xuân, Hà Nội",

"cardType": "VIP",

"registrationDate": "2019-05-20",

"expiryDate": "2024-05-20",

"status": "Hoạt động",

"currentlyBorrowedCount": 3,

"borrowHistory": [

{

"bookCode": "S002",

"borrowDate": "2024-01-05",

"dueDate": "2024-02-05",

"returnDate": "2024-01-30",

"status": "Đã trả"

},

{

"bookCode": "S004",

"borrowDate": "2024-03-01",

"dueDate": "2024-04-01",

"returnDate": null,

"status": "Đang mượn"

},

{

"bookCode": "S001",

"borrowDate": "2024-03-10",

"dueDate": "2024-04-10",

"returnDate": null,

"status": "Đang mượn"

},

{

"bookCode": "S003",

"borrowDate": "2024-03-10",

"dueDate": "2024-04-10",

"returnDate": null,

"status": "Đang mượn"

}

],

"id": "759c"

},

{

"code": "TV003",

"fullName": "Lê Quang Tuấn",

"birthDate": "1988-12-03",

"email": "tuan.le@hotmail.com",

"phone": "0965432109",

"address": "123 Giải Phóng, Hai Bà Trưng, Hà Nội",

"cardType": "Thường niên",

"registrationDate": "2021-01-10",

"expiryDate": "2026-01-10",

"status": "Hoạt động",

"currentlyBorrowedCount": 1,

"borrowHistory": [

{

"bookCode": "S005",

"borrowDate": "2024-02-20",

"dueDate": "2024-03-20",

"returnDate": "2024-03-18",

"status": "Đã trả"

},

{

"bookCode": "S002",

"borrowDate": "2024-03-20",

"dueDate": "2024-04-20",

"returnDate": null,

"status": "Đang mượn"

}

],

"id": "d937"

}

],

"sukien": [

{

"id": "SK001",

"name": "Gặp gỡ tác giả Nguyễn Ngọc Tư",

"time": "2024-04-15 18:30:00",

"location": "Phòng hội thảo - Tầng 3",

"description": "Buổi gặp gỡ và giao lưu với nhà văn Nguyễn Ngọc Tư, tác giả của 'Cánh đồng bất tận'",

"registeredCount": 120,

"maxCapacity": 150

},

{

"id": "SK002",

"name": "Workshop Lập trình Python cho người mới bắt đầu",

"time": "2024-04-20 09:00:00",

"location": "Phòng máy tính - Tầng 2",

"description": "Khóa học cơ bản về ngôn ngữ lập trình Python dành cho người mới bắt đầu",

"registeredCount": 25,

"maxCapacity": 30

},

{

"id": "SK003",

"name": "Đêm thơ Xuân Quỳnh",

"time": "2024-05-10 19:00:00",

"location": "Sân vườn - Tầng 1",

"description": "Đêm thơ nhạc tưởng nhớ nữ thi sĩ Xuân Quỳnh và những tác phẩm để đời",

"registeredCount": 85,

"maxCapacity": 100

}

],

"nhanvien": [

{

"code": "NV001",

"fullName": "Phạm Thị Mai",

"position": "Giám đốc thư viện",

"email": "mai.pham@trithucmo.vn",

"phone": "0912345670",

"hireDate": "2010-01-15",

"id": "43eb"

},

{

"code": "NV002",

"fullName": "Trần Văn Hùng",

"position": "Quản lý hệ thống",

"email": "hung.tran@trithucmo.vn",

"phone": "0912345671",

"hireDate": "2010-02-01",

"id": "825d"

},

{

"code": "NV003",

"fullName": "Lê Thị Nga",

"position": "Thủ thư",

"email": "nga.le@trithucmo.vn",

"phone": "0912345672",

"hireDate": "2011-03-10",

"id": "c892"

},

{

"code": "NV004",

"fullName": "Nguyễn Văn Toàn",

"position": "Thủ thư",

"email": "toan.nguyen@trithucmo.vn",

"phone": "0912345673",

"hireDate": "2012-05-20",

"id": "7b83"

},

{

"code": "NV005",

"fullName": "Hoàng Thị Linh",

"position": "Quản lý sự kiện",

"email": "linh.hoang@trithucmo.vn",

"phone": "0912345674",

"hireDate": "2015-09-15",

"id": "37c5"

}

]

}

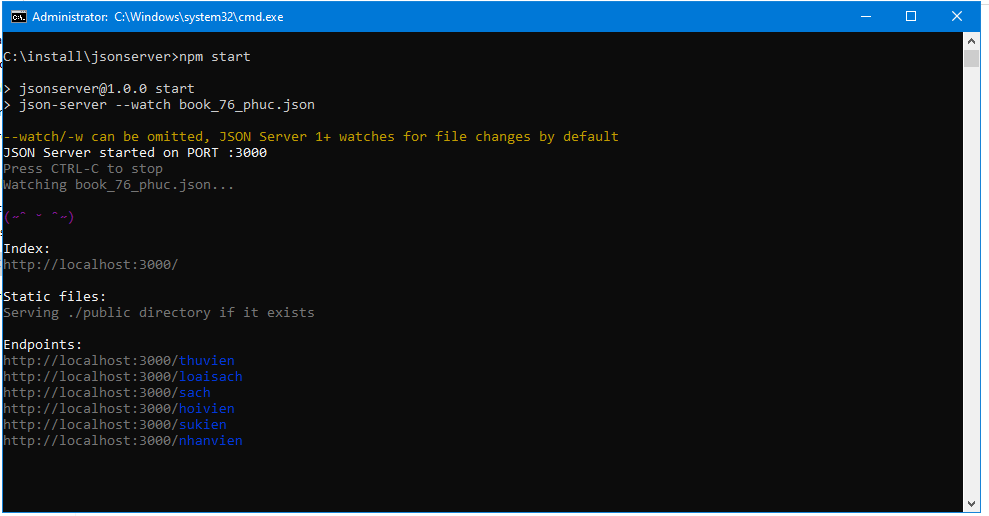
### 4.Mở file package.json, thêm dòng start.

"start": "json-server --watch book\_76\_phuc.json"

Khi khởi động JSON Server. Sử dụng file book\_76\_phuc.json làm cơ sở dữ liệu.



Mở cmd chạy lệnh npm start:



Đây là API của nó (API la url chứa dữ liệu mình muốn hiển thị ra)

http://localhost:3000/thuvien

http://localhost:3000/loaisach

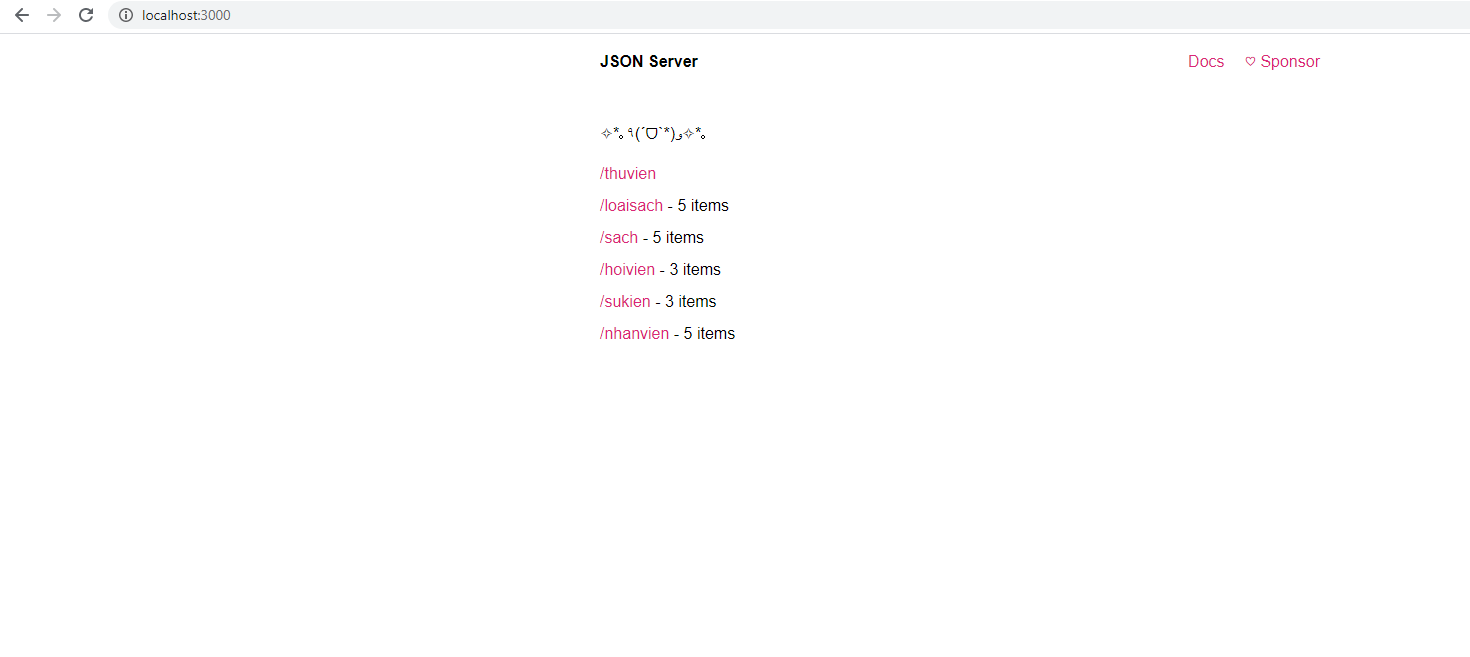
http://localhost:3000/sach

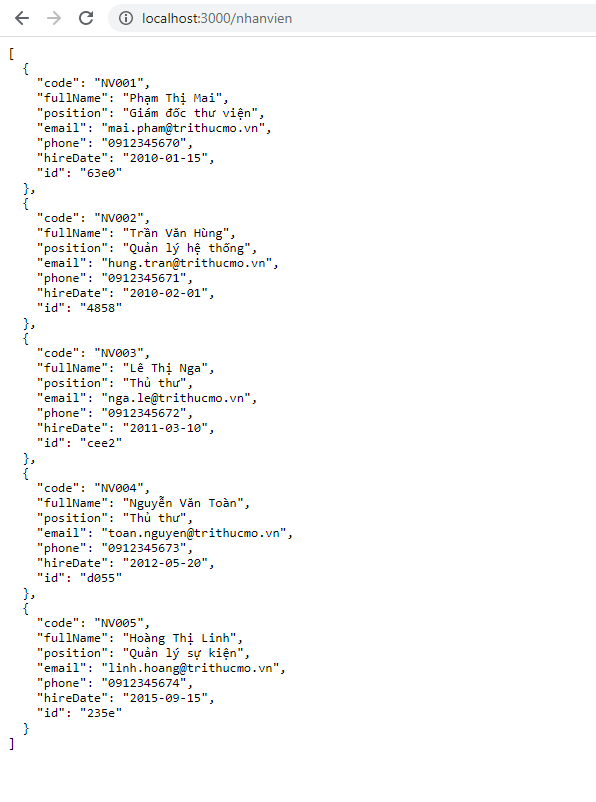
http://localhost:3000/hoivien

http://localhost:3000/sukien

http://localhost:3000/nhanvien

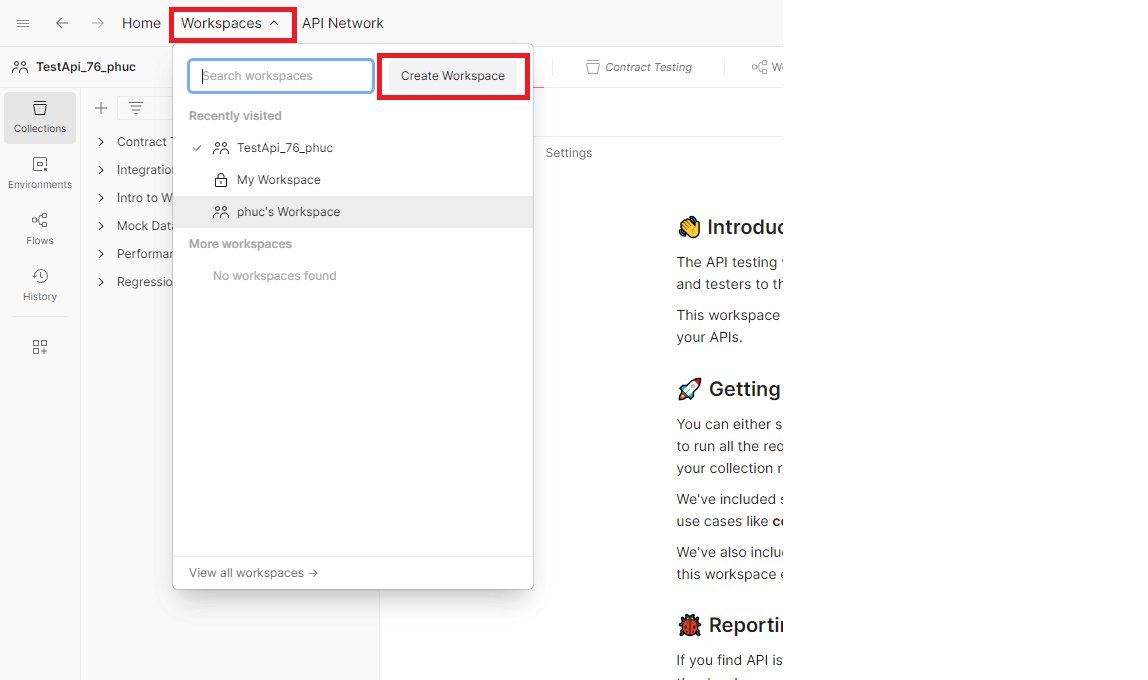
Mở trong chorme sẽ thấy.

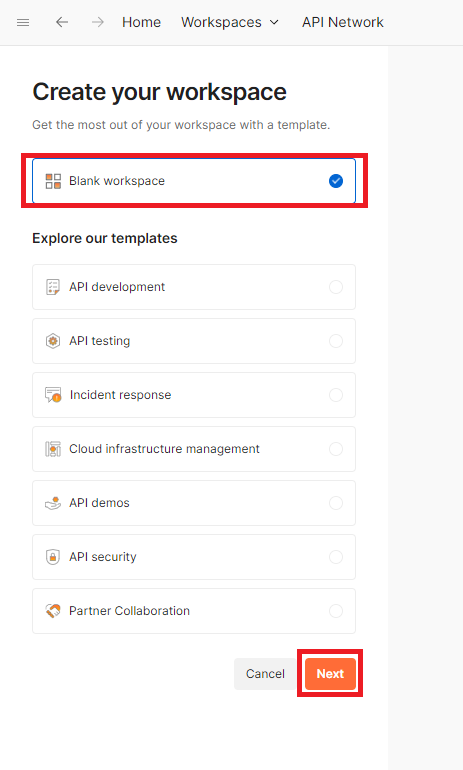


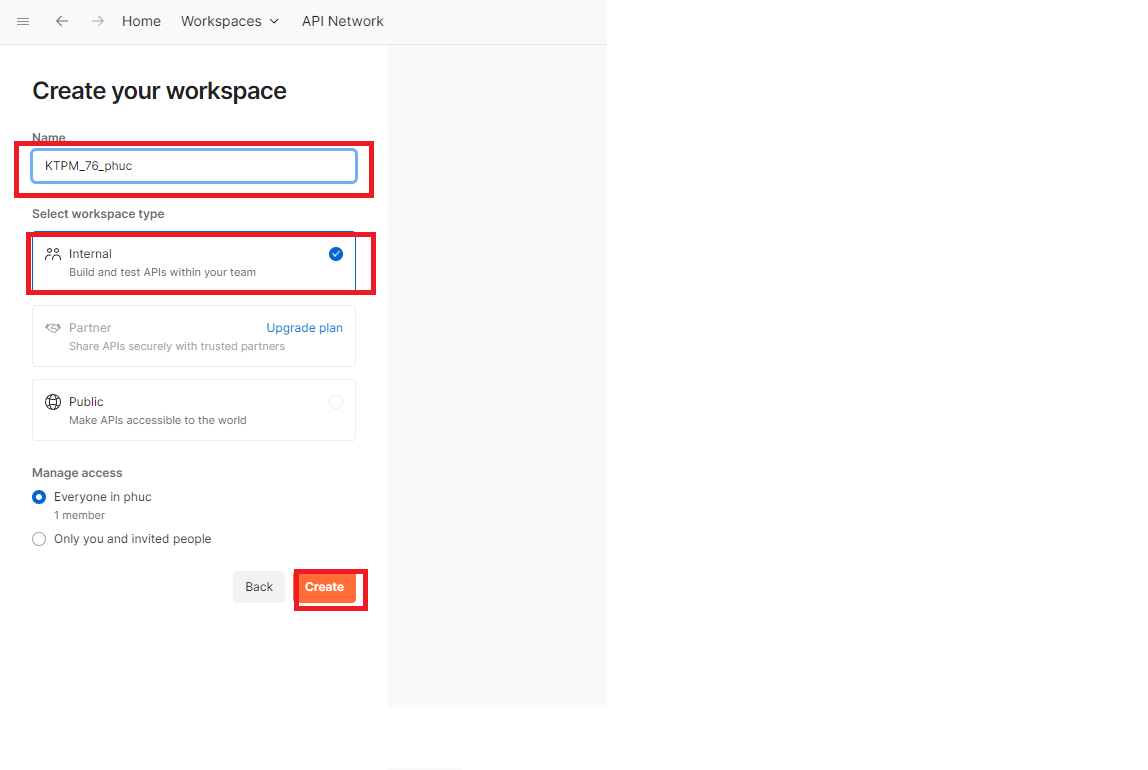


## 2.Test API.

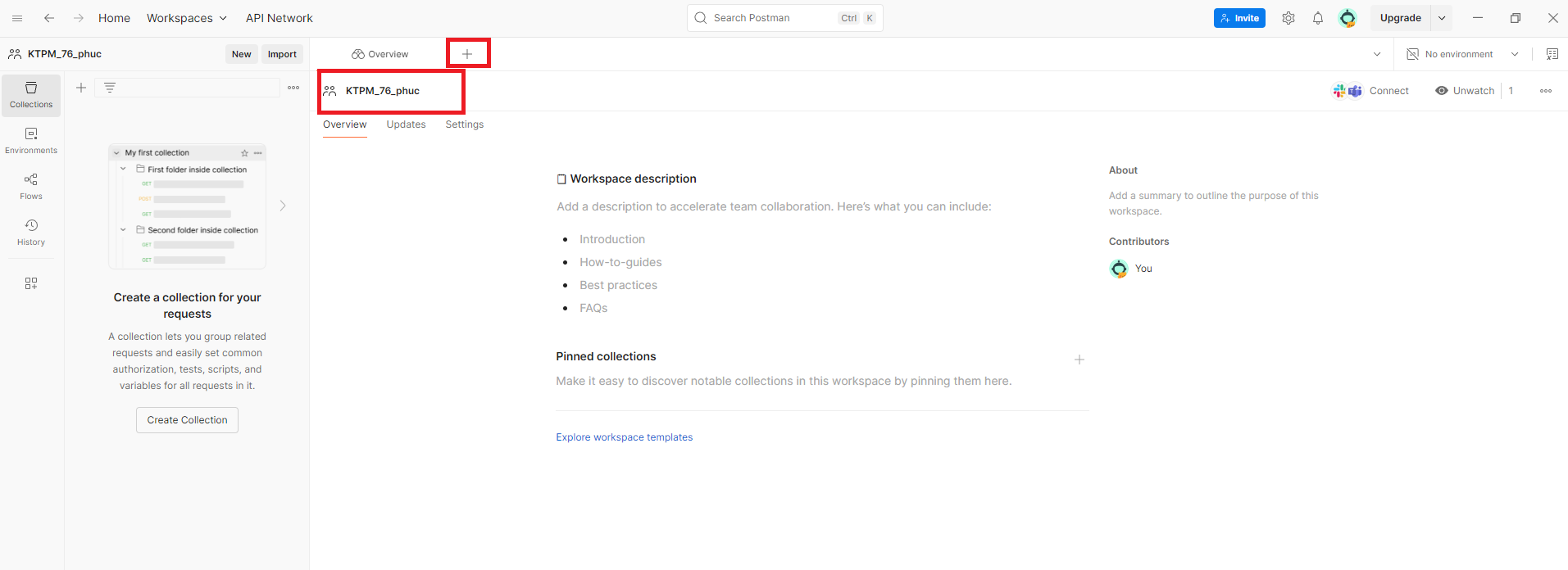
### 1.Tạo workspace:







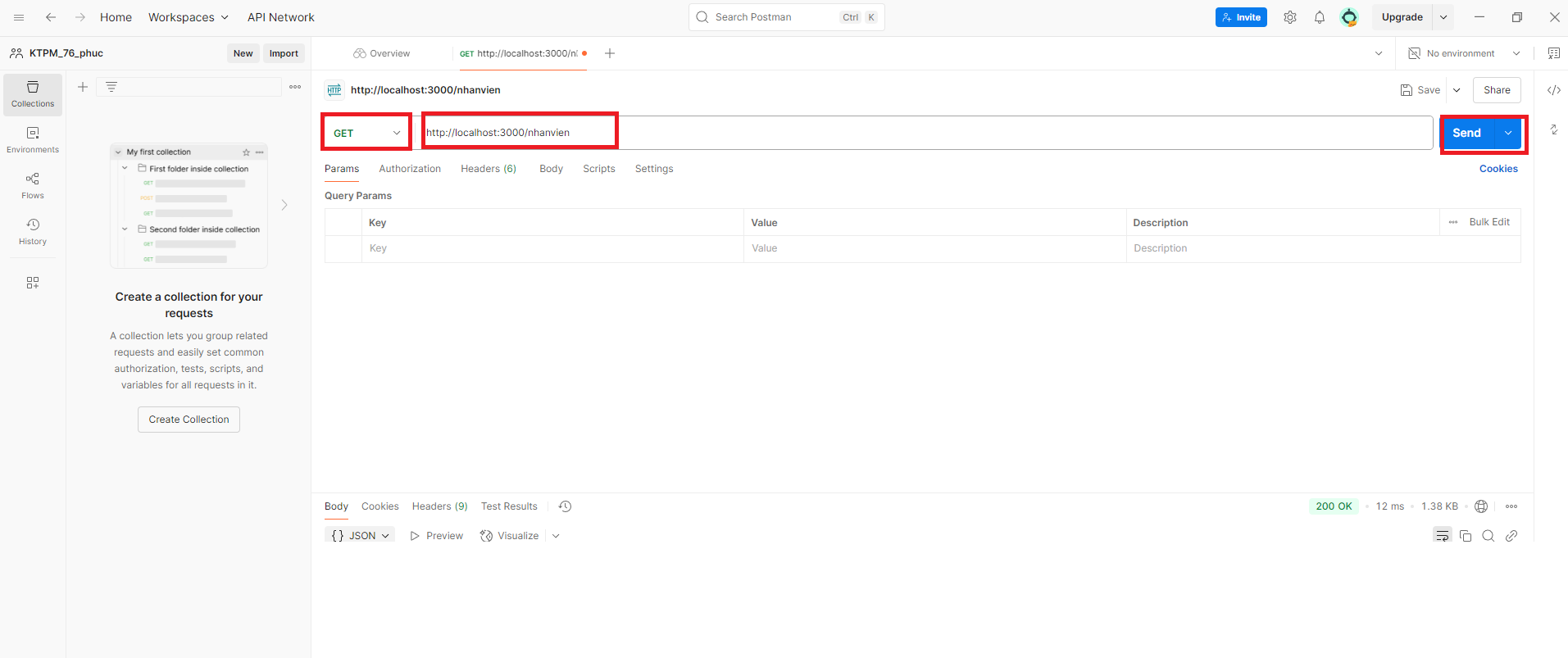
Ấn vào dấu + ở bên trên



### 2.Get API

GET: Phương thức GET trong API là phương thức phổ biến nhất, được dùng để lấy dữ liệu từ server

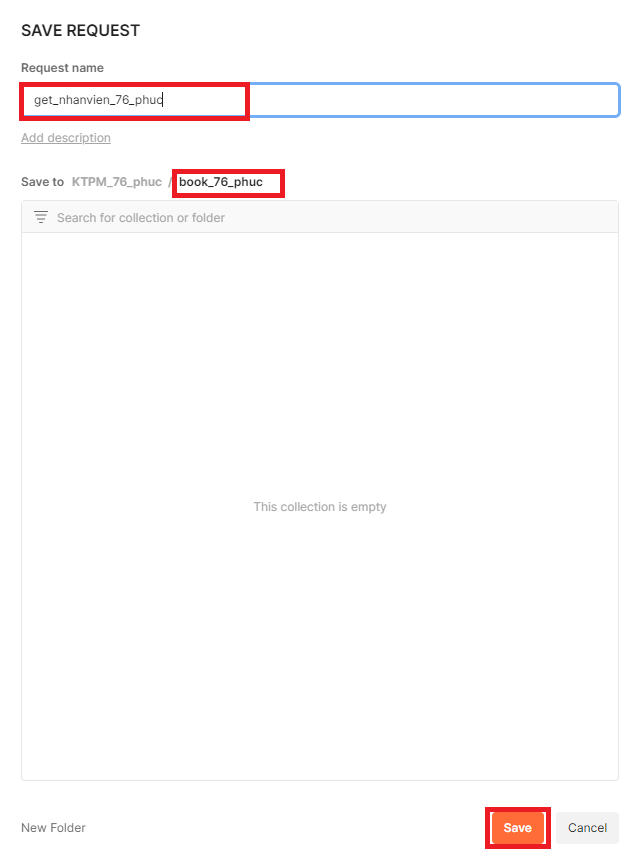
Chọn phương thúc và đường dẫn đến trang web api



Ta được kết quả trả về đúng với json.



Khi lưu đặt lại tên , chọn collection và nhấn save.

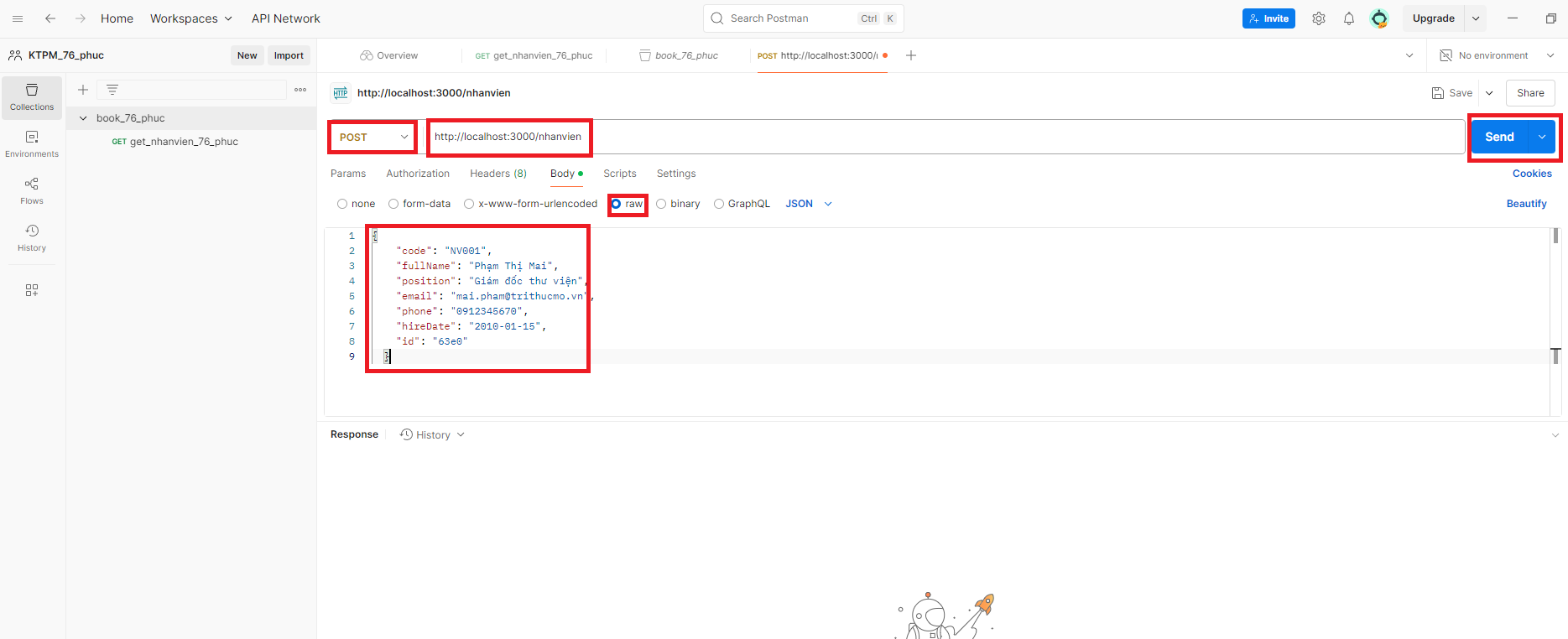


### 3.Post API

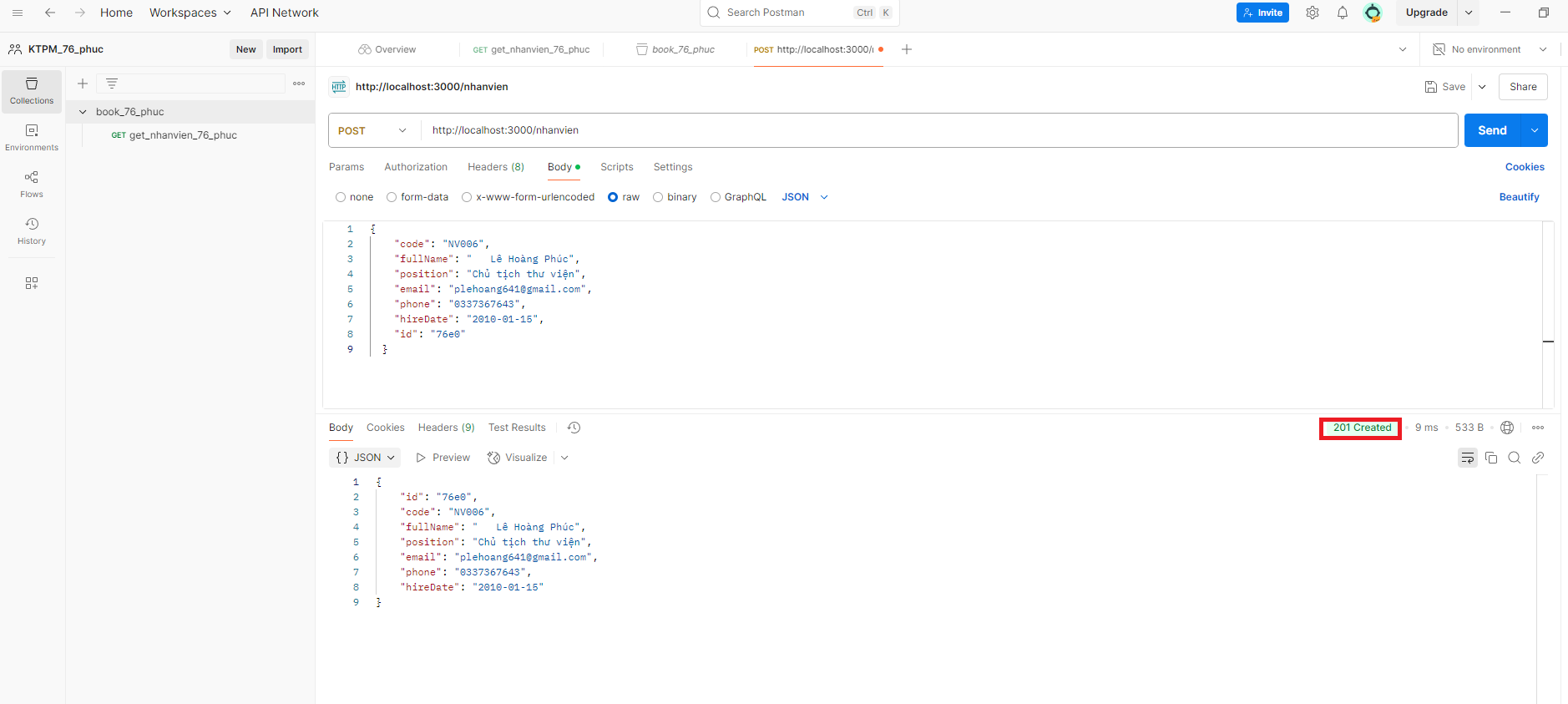
POST: Phương thức POST trong API được dùng để gửi dữ liệu từ client lên server nhằm tạo mới một tài nguyên hoặc thực hiện một hành động nào đó

ấn dấu + một http request > chọn phương thức là post

trong tab body truyền dữ liệu mà client muốn gửi lên server



Khi chạy ta được kết quả trả về 201 Created là đã chạy thành công

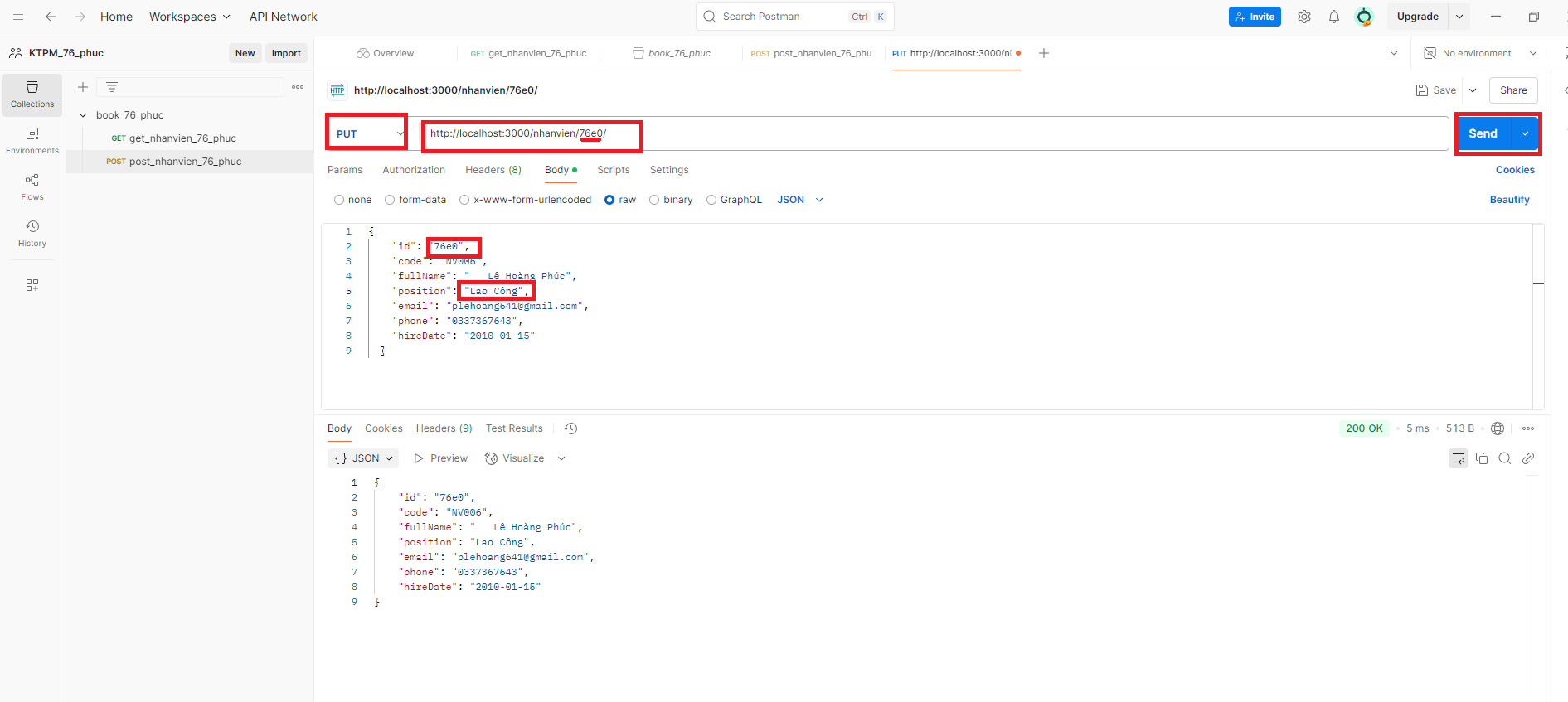


### 4.Put API

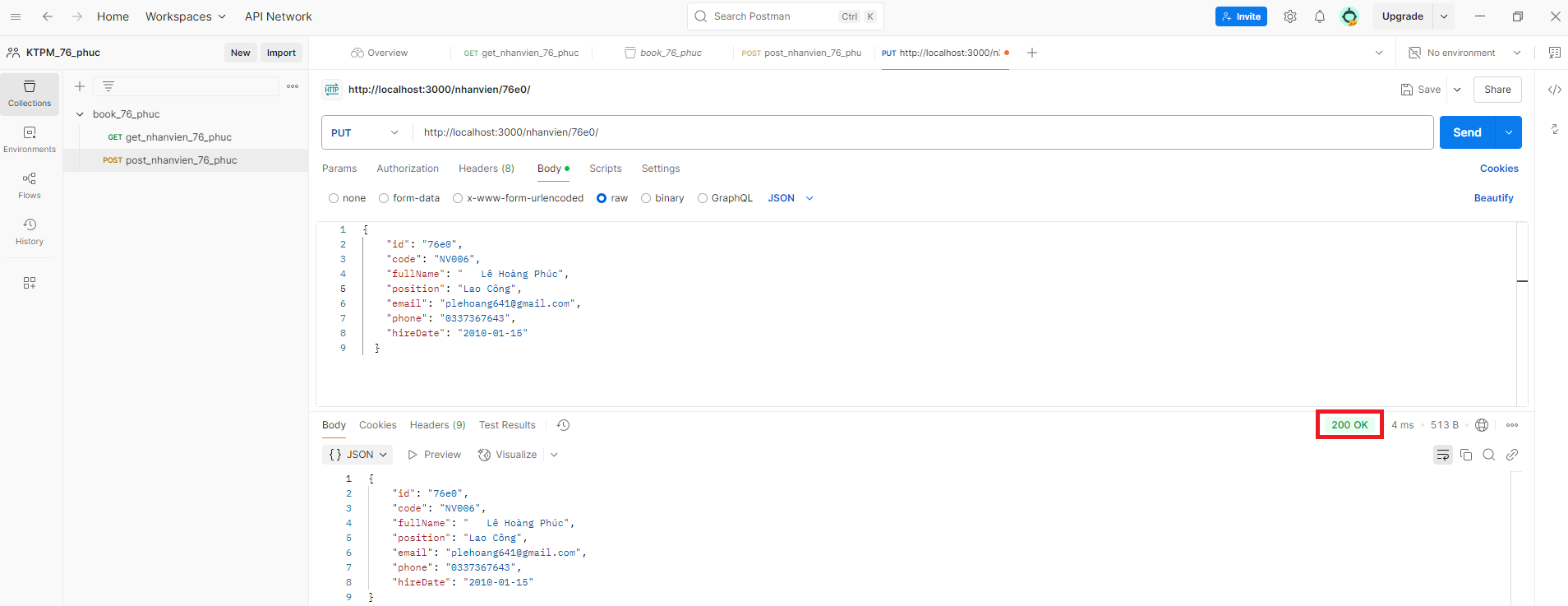
PUT: Phương thức PUT trong API được dùng để cập nhật toàn bộ một tài nguyên đã tồn tại trên server. Nó gửi dữ liệu mới thay thế toàn bộ dữ liệu cũ của tài nguyên đó.

ấn dấu + một http request > chọn phương thức là PUT

trong tab body truyền dữ liệu mà client muốn gửi lên server để tạo mới một tài nguyên.



Khi chạy ta được kết quả trả về 200 OK là đã chạy thành công

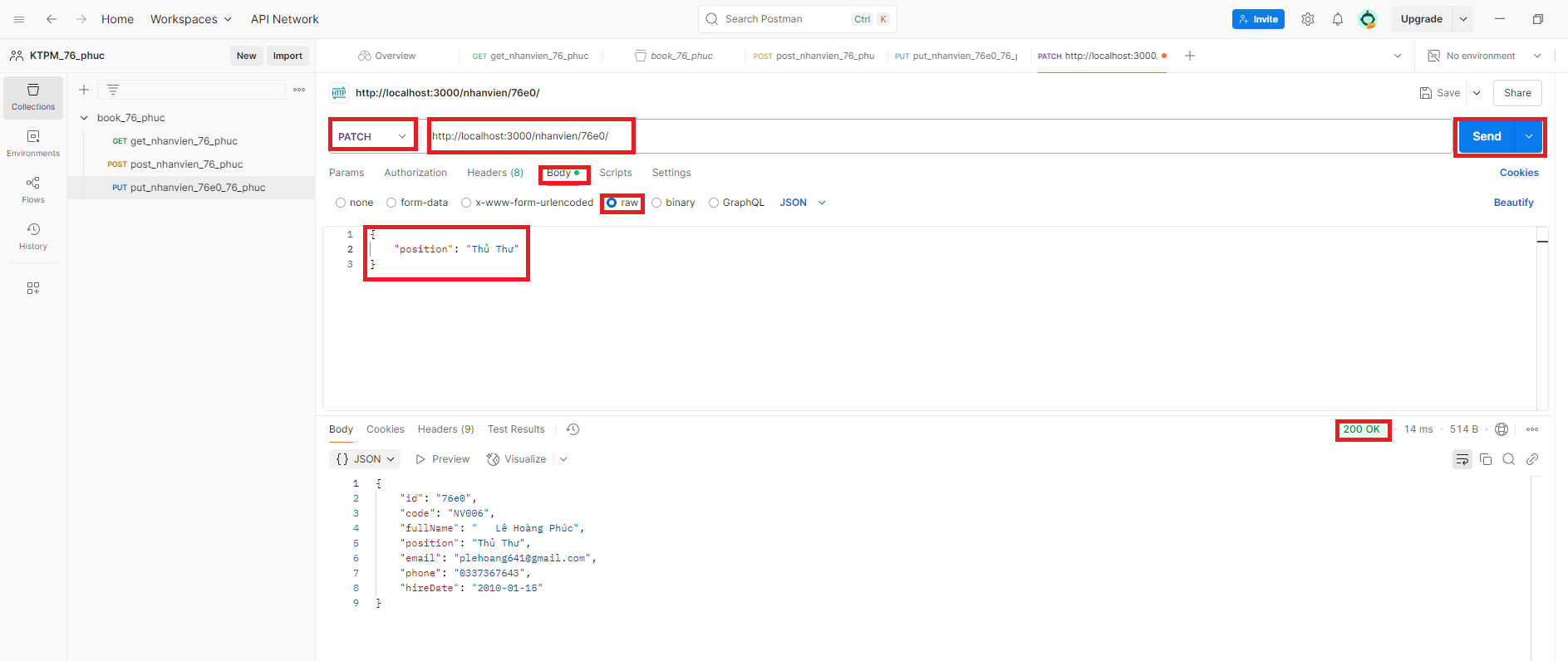


### 5.PATCH API.

Phương thức **PATCH** trong API được dùng để **cập nhật một phần** thông tin của một tài nguyên đã tồn tại trên server.

ấn dấu + một http request > chọn phương thức là PATCH

trong tab body truyền dữ liệu mà client muốn gửi lên server để tạo mới một tài nguyên.

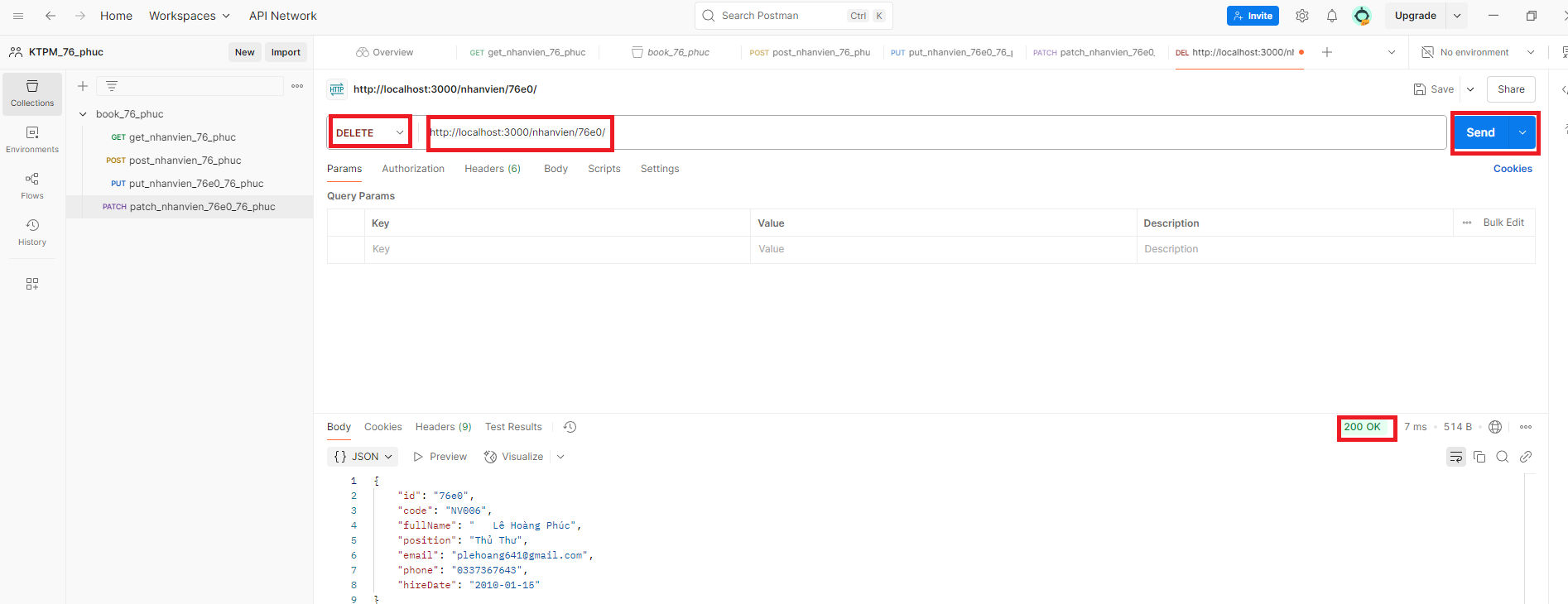


Khi chạy ta được kết quả trả về 200 OK là đã chạy thành công

### 6.Delete API.

Phương thức DELETE trong API được dùng để xóa một tài nguyên trên server, dựa theo định danh (ID hoặc khóa) mà client cung cấp.

ấn dấu + một http request > chọn phương thức là DELETE.



Khi chạy ta được kết quả trả về 200 OK là đã chạy thành công

## 3.Postman variable.

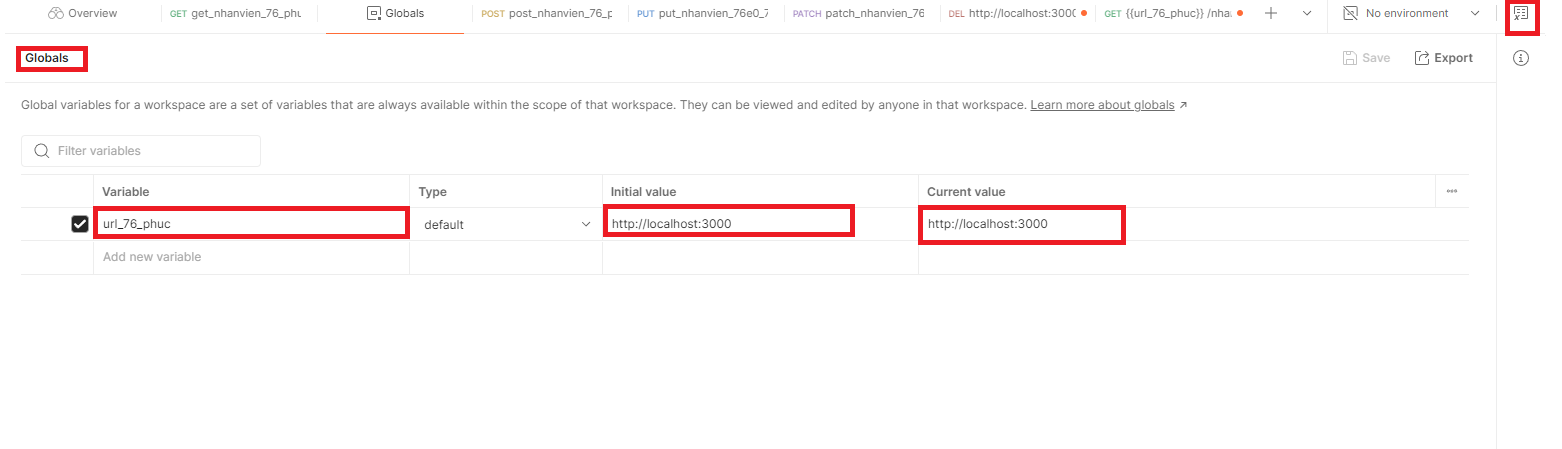
Việc tạo biến trong Postman giúp bạn tự động hóa, tái sử dụng và quản lý dữ liệu dễ dàng hơn khi test API. Nó đặc biệt hữu ích khi bạn làm việc với nhiều môi trường (dev, test, production) hoặc cần sử dụng lại dữ liệu (token, ID, URL, v.v.) giữa các request.

### 1.Biến globals

Tạo biến globals

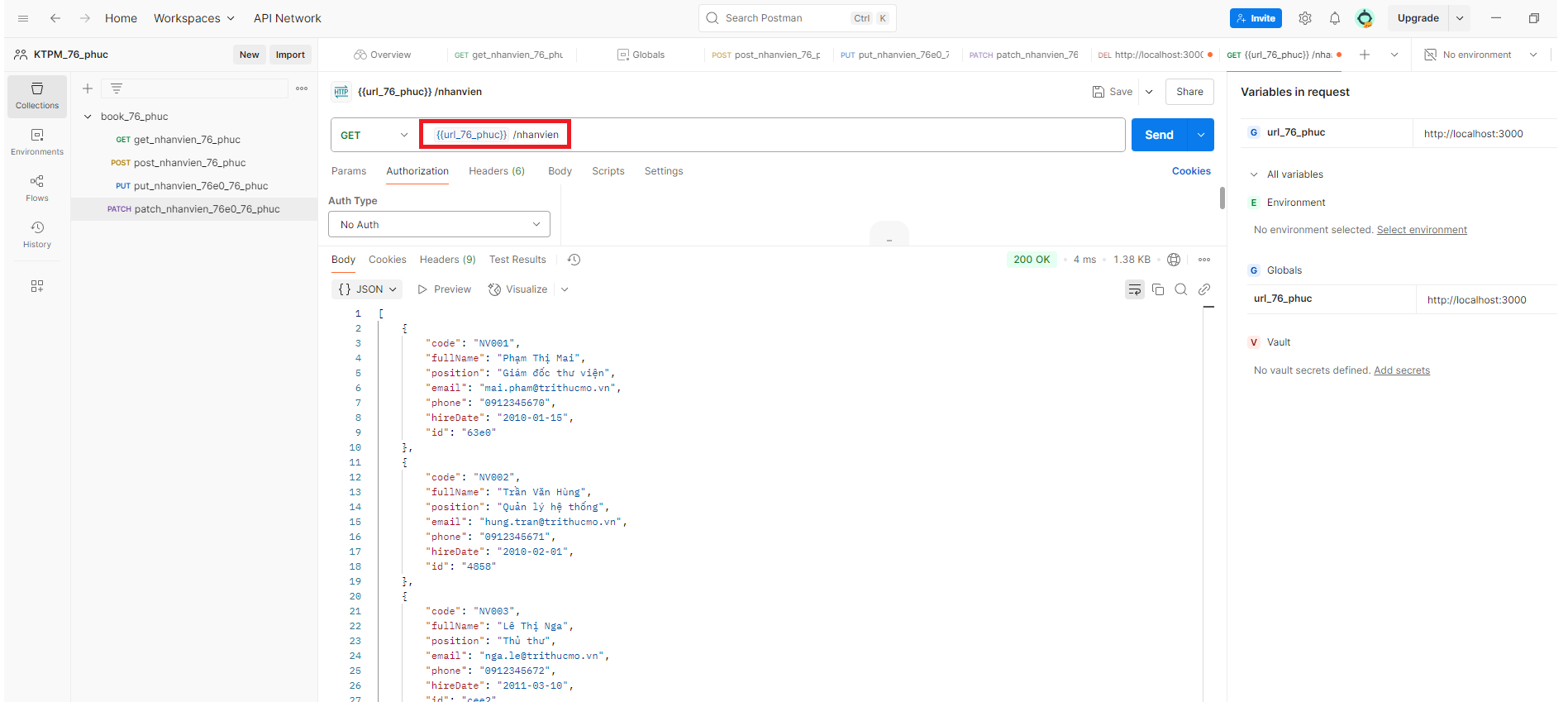
Nhấn vào biểu tượng variable ở góc trên cùng:

Sau đó thiết lập các thông số để tạo ra biến global:



Dùng p {{ url\_76\_phuc}} thay cho http://localhost:3000

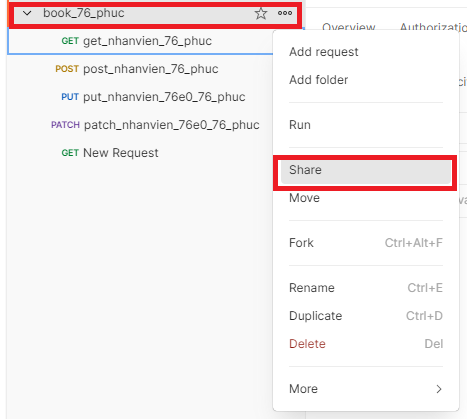
Dữ liệu sẽ trả về với kết quả tương tự:



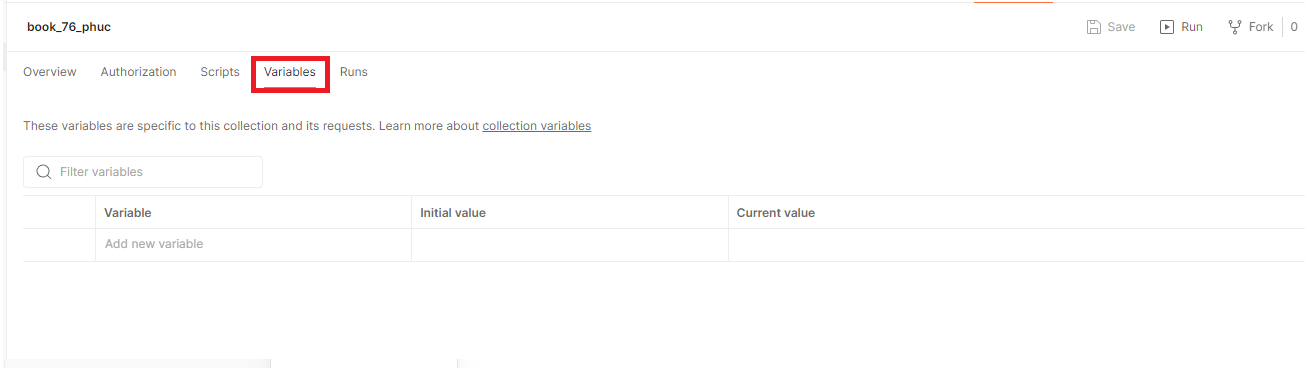
### Biến collections

##### Tạo biến collections trong API postman:

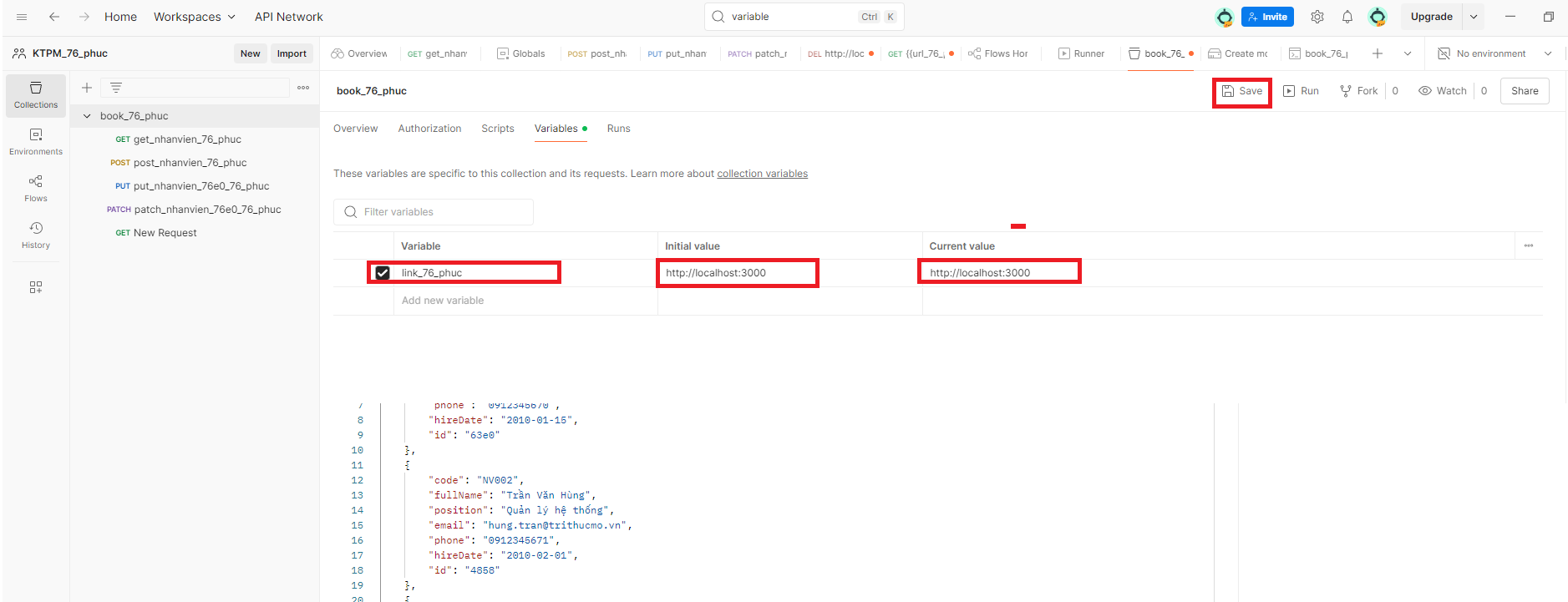
Chuột phải ở collection chọn share.



Sau đó tick qua tab variables

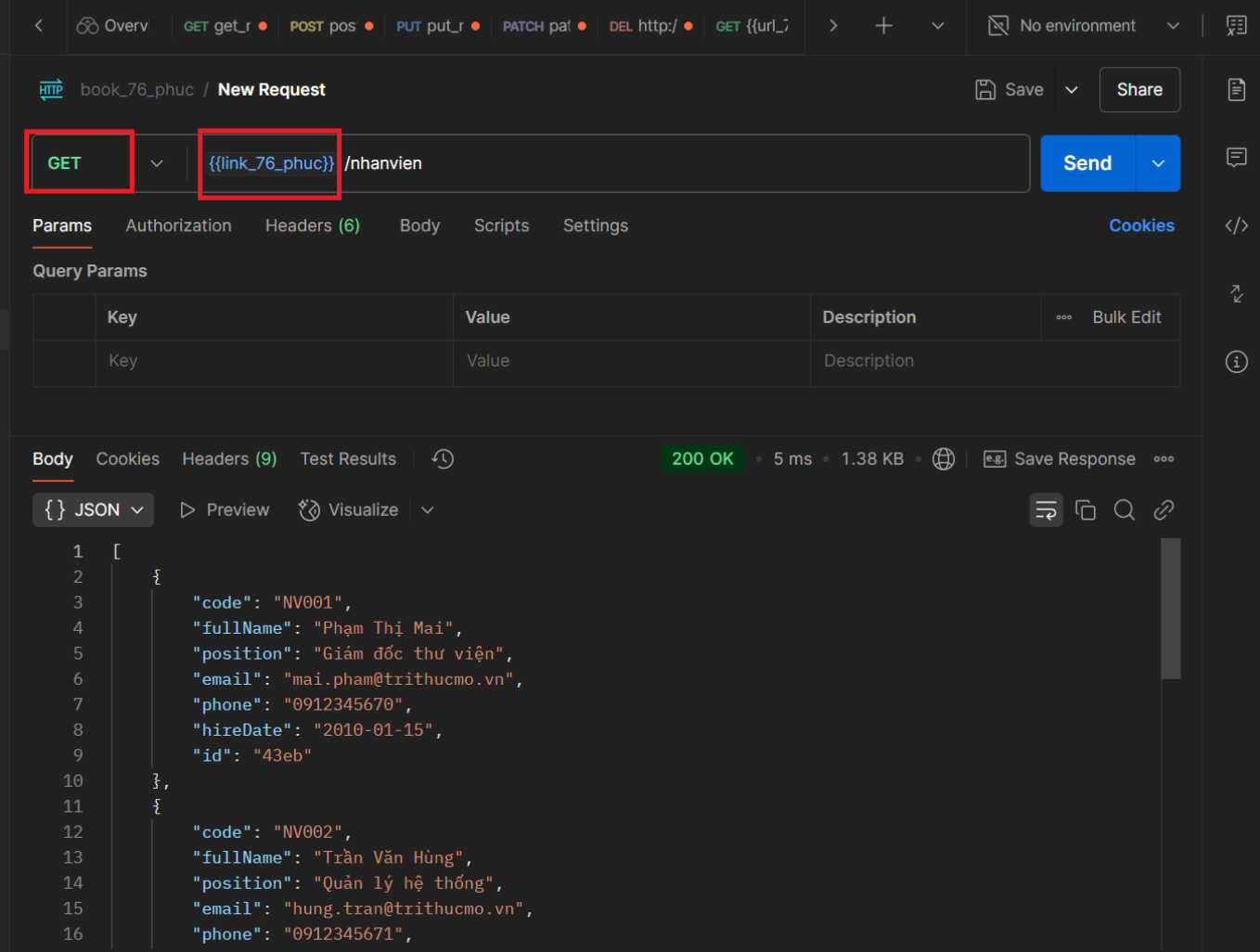


Thiết lập thuộc tính biến tương tự như biến globals rồi nhấn save:



Dùng {{ link\_76\_phuc}} thay cho http://localhost:3000

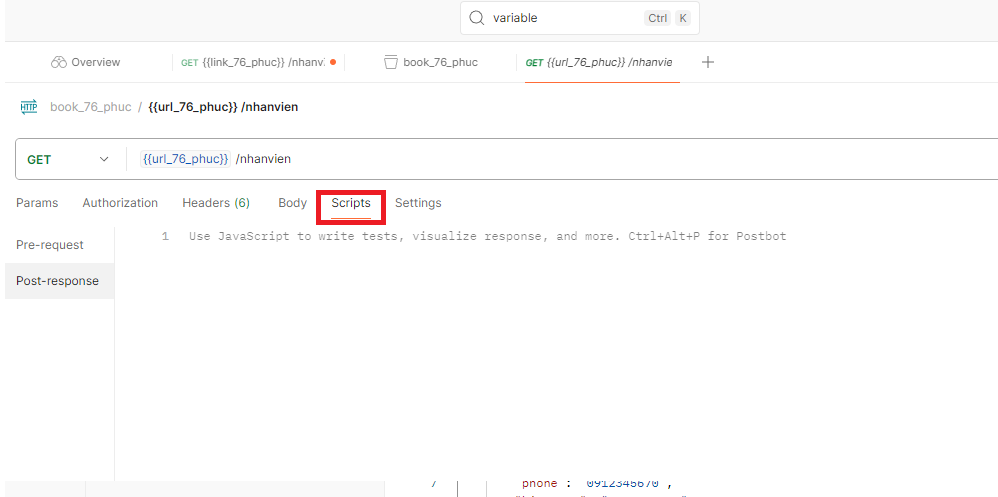
Dữ liệu sẽ trả về với kết quả tương tự:



## 4. Postman script

### Tạo script

*Ở góc dưới, ấn vào tab script*

**

### 1.Get API.

Script test:

TC1\_76\_phuc:Lấy toàn bộ dữ liệu thành công nên kết quả là pass.

TC2\_76\_phuc:Dữ liệu trả về là json nên kết quả là pass.

TC3\_76\_phuc:Kiểm tra xem trong dữ liệu trả về có nhân viên nào tên Lê Hoàng Phúc hay không. Vì không có lên kết quả là fail.

// 1. Kiểm tra status code là 200

pm.test("TC1\_76\_phuc - Status code is 200", *function*() {

    pm.response.to.have.status(200);

});

// 2. Kiểm tra response có phải là JSON không

pm.test("TC2\_76\_phuc - Response is JSON", *function*() {

    pm.response.to.be.withBody;

    pm.response.to.be.json;

});

//3 tc faild vì không có nhan vien nào là Lê Hoàng Phúc

pm.test("TC3\_76\_phuc - Contains nhan vien fullName 'Lê Hoàng Phúc'(fail)", *function*() {

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

*const* exists = jsonData.some(*nhanvien* *=>* nhanvien.fullName === "Lê Hoàng Phúc");

    pm.expect(exists).to.be.true;

});

Kết quả khi chạy:

A screenshot of a computer

AI-generated content may be incorrect.

### 2.Post API.

Script test:

TC1\_76\_phuc: Post một nhân viên mới thành công nên kết quả trả về là pass.

TC2\_76\_phuc:Kiểm tra xem sau cập nhập có nhân viên nào tên Lê Hoàng Phúc không. Vì nhân viên mới post có tên là Lê Hoàng Phúc nên kết quả là pass.

TC3\_76\_phuc: Kiểm tra xem sau cập nhập có nhân viên nào tên Hoàng Thị Linh không. Vì không có nhân viên nào tên là Hoàng Thị Linh nên kết quả là fail.

// Kiểm tra mã trạng thái là 201 (Created)

pm.test("TC1\_76\_phuc-Status code is 201", *function*() {

    pm.response.to.have.status(201);

});

//kiểm tra có nhan vien nào là Lê Hoàng Phúc

pm.test("TC2\_76\_phuc - Giá trị sau khi cập nhật", *function*() {

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

    pm.expect(data.fullName).to.eql("Lê Hoàng Phúc");

});

//kiểm tra có nhan vien nào là Hoàng Thị Linh.

pm.test("TC3\_76\_phuc - Giá trị fullName sai sau khi cập nhật", *function*() {

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

    pm.expect(data.fullName).to.eql("Hoàng Thị Linh");

});

Kết quả khi chạy:

A screenshot of a computer

AI-generated content may be incorrect.

### 3.Put API.

Script test:

TC1\_76\_phuc:Cập nhật position thành công nên kết quả trả về là pass.

TC2\_76\_phuc:Kiểm tra xem position đã được cập nhật đúng hay chưa.Vì position trả về là Lao Công nên kết quả là pass.

TC3\_76\_phuc:Kiểm tra xem position đã được cập nhật đúng hay chưa.Vì position trả về là Chủ Tịch nên kết quả là fail.

// Kiểm tra cập nhật nhân viên thành công (trả về status 200)

pm.test("TC1\_76\_phuc - Cập nhật nhân viên thành công", *function*() {

    pm.response.to.have.status(200); // json-server thường trả về 200

});

// Kiểm tra position nhân viên được cập nhật đúng

pm.test("TC2\_76\_phuc - position được cập nhật đúng(pass)", *function*() {

*const* data = pm.response.json();

    pm.expect(data.position).to.eql("Lao Công");

});

// Kiểm tra position nhân viên được cập nhật.

pm.test("TC3\_76\_phuc - position được cập nhật(fail)", *function*() {

*const* data = pm.response.json();

    pm.expect(data.position).to.eql("chủ tịch");

});

Kết quả khi chạy:

A screenshot of a computer

AI-generated content may be incorrect.

### 4.Patch API.

Script test:

TC1\_76\_phuc: Cập nhật position thành công nên kết quả trả về là pass.

TC2\_76\_phuc: Kiểm tra xem position đã được cập nhật đúng hay chưa.Vì position trả về là Thủ Thư nên kết quả là pass.

TC3\_76\_phuc:Kiểm tra xem position đã được cập nhật đúng hay chưa.Vì position trả về là Chủ Tịch nên kết quả là fail.

pm.test("TC1\_76\_phuc - Cập nhật nhân viên thành công", *function*() {

    pm.response.to.have.status(200); // json-server thường trả về 200

});

//kiểm tra xem giá trị position đã cập nhật đúng chưa.

pm.test("TC2\_76\_phuc - Giá trị position đúng sau khi cập nhật(pass)", *function*() {

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

    pm.expect(data.position).to.eql("Thủ Thư");

});

//kiểm tra giá trị position sai.

pm.test("TC3\_76\_phuc - Giá trị position sai sau khi cập nhật(fail)", *function*() {

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

    pm.expect(data.position).to.eql("chủ tịch");

});

Kết quả khi chạy:

A screenshot of a computer

AI-generated content may be incorrect.

### 5.Delete API.

Script test:

TC1\_76\_phuc:Xoá Thành công nên kết quả là pass.

TC2\_76\_phuc: Kiểm tra xem có tìm thấy id khi xoá hay không. Vì tìm thấy id 76e0 nên kết quả trả về là pass.

TC3\_76\_phuc: Kiểm tra xem có tìm thấy id khi xoá hay không. Vì tìm thấy id 76a0 nên kết quả trả về là fail.

pm.test("TC1\_76\_phuc - Xóa sách thành công", *function* () {

    pm.response.to.have.status(200); // hoặc 204 tùy vào backend

});

//kiểm tra xem có tìm thấy id cần xoá

pm.test("TC2\_76\_phuc - tìm thấy giá trị id khi xoá(pass)", *function* () {

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

    pm.expect(data.id).to.eql("76e0");

});

//kiểm tra có tìm thấy id sai không

pm.test("TC3\_76\_phuc - Giá trị id sai khi xoá(fail)", *function*() {

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

    pm.expect(data.id).to.eql("76a0");

});

Kết quả khi chạy:

A screenshot of a computer

AI-generated content may be incorrect.