# BÀI 2: CÔNG CỤ KIỂM TRA WEB API

Trong quá trình triển khai dự án, phần server và client làm độc lập với nhau nên có nhiều chỗ  
client chưa làm xong, mình không thể chờ client làm xong để test được dữ liệu mà test API bằng  
công cụ khác luôn –> Lúc này việc test hoàn toàn không phụ thuộc gì vào client.  
Kể cả khi client làm xong rồi, nếu mình test trên client mà thấy lỗi liên quan đến logic và dữ liệu  
thì cũng cần test thêm cả API để biết chính xác là server sai hay client sai –> fix lỗi sẽ nhanh hơn.  
Khi làm hệ thống web services, dự án của mình chỉ viết API cho bên khác dùng, mình sẽ không có  
client để test giống như các dự án khác –> phải test API hoàn toàn

## 2.1. Kiểm tra Web API với Swagger

### 2.1.1. Giới thiệu về Swagger

Swagger là một phần mềm mã nguồn mở được sử dụng để phát triển, thiết kế, xây dựng và làm tài liệu cho các RESTful Web Service.

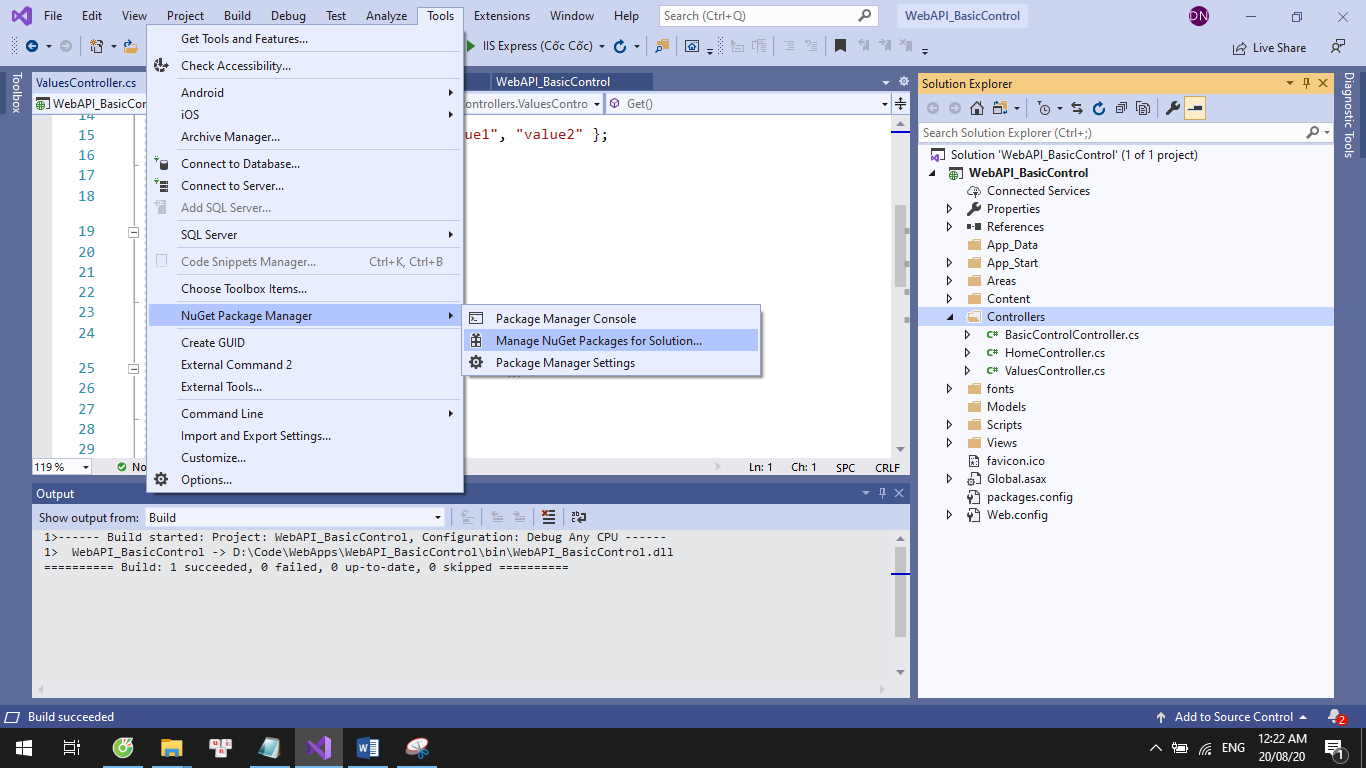
Swagger vừa help vừa test cho web API xây dựng, với Swagger ta có thể biết rõ được có những API nào, method và url tương ứng của nó

Với mỗi api ta lại biết được chi tiết input và output của như trường nào bắt buộc gửi lên, kết quả trả về có thể nhận những status nào…

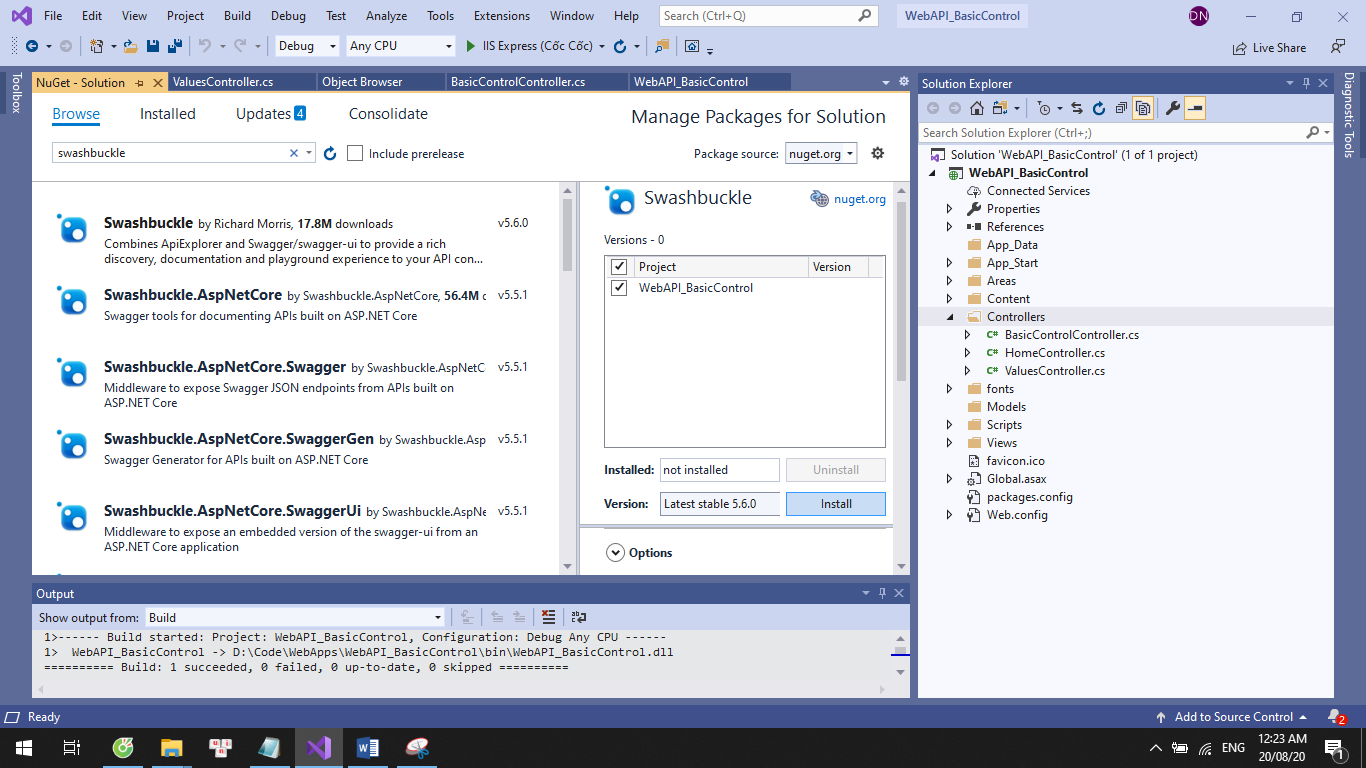
### 2.1.2. Cài đặt Swagger

Thường cài Swagger cho ứng dụng WebAPi bằng 2 cách

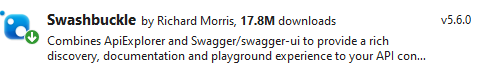
**Cách 1** Mở Manage NuGet Package



Search package swashbuckler

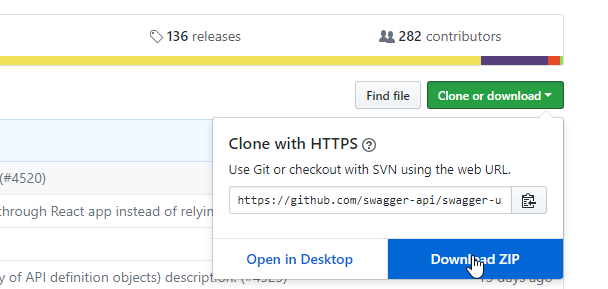


Chọn Install



Hiện biểu tượng như trên là việc cài package đã hoàn tất

**Cách 2** download nó tại github  [https://github.com/swagger-api/swagger-ui](https://github.com/swagger-api/swagger-ui" \t "_blank) về và cài đặt

[](https://stackjava.com/wp-content/uploads/2018/05/swagger-5.png)

Sau khi tải được file zip, bạn giải nén ra một folder, trong folder đó sẽ có thư dist, đây chính là bản Swagger UI mà ta sẽ sử dụng.

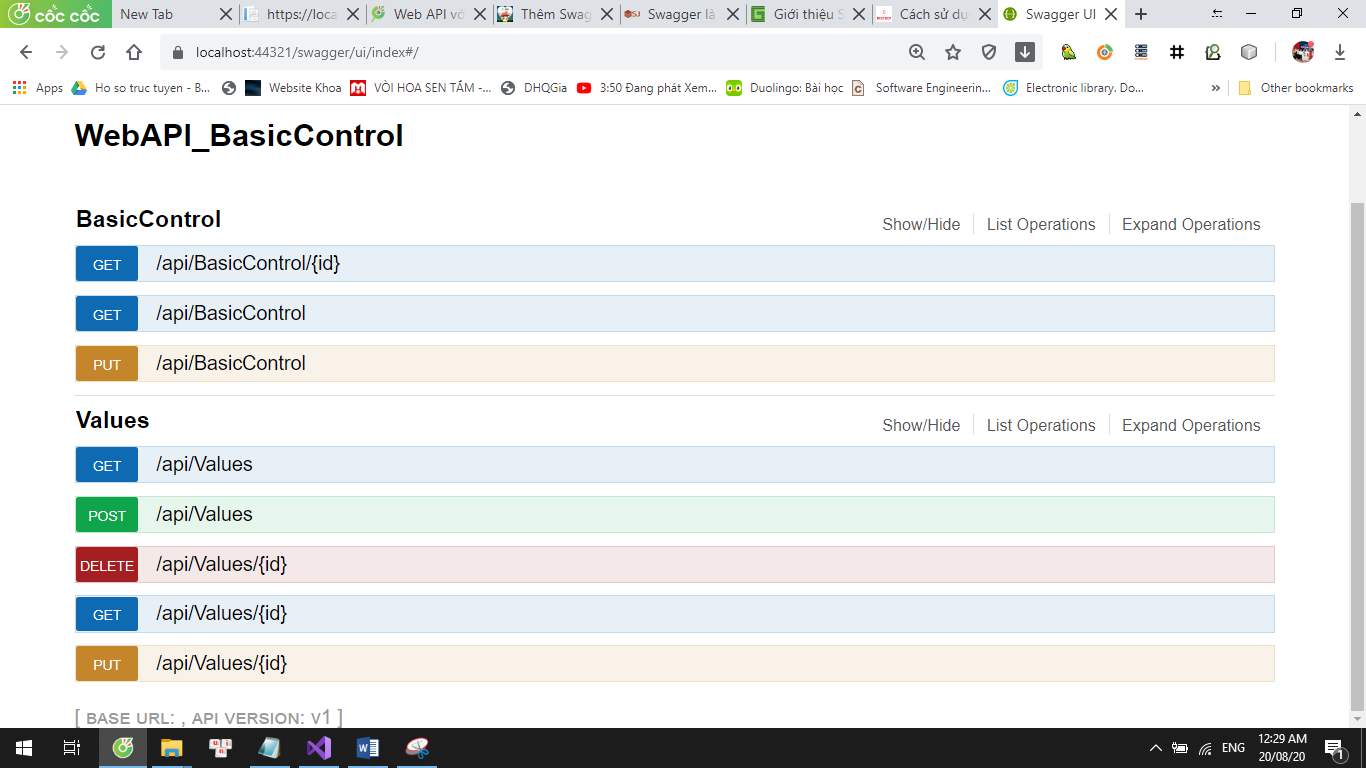
Ta có thể copy folder dist và deploy nó lên một server hoặc mở file index.html trên trình duyệt để chạy Swagger UI.

2.1.3. Cách sử dụng Swagger

Với cách 1 Swagger sẽ đi kèm với ứng dụng. Khi chạy ứng dụng, thêm Swagger vào url (giả sử url là <https://localhost:44321>) thì ta được là <https://localhost:44321/swagger>

Swagger UI sẽ hiển thị help và giao diện để test web API

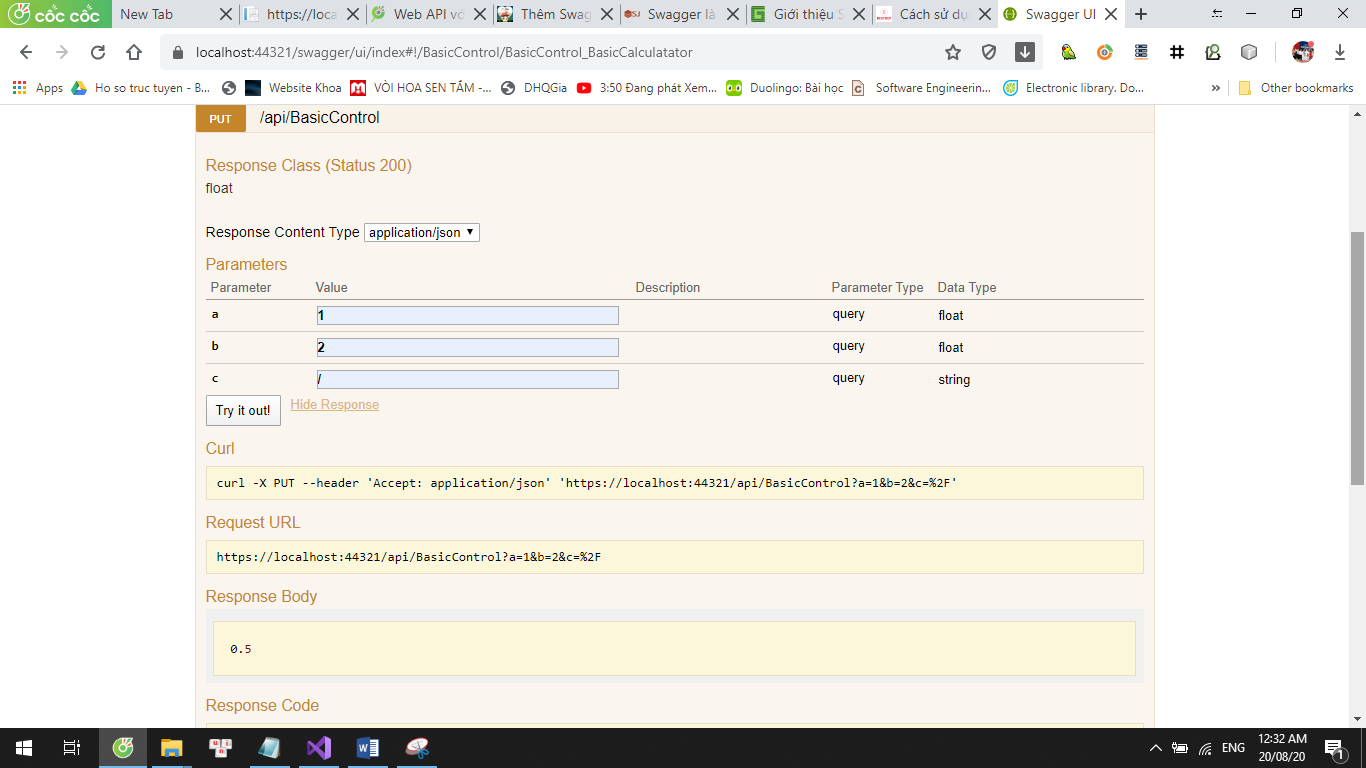
**Ví dụ** test WebAPi trong bài 1 như sau



Nhập giá trị input là các tham số Parameter

Chọn Try it out để gửi request tới webAPI

Để lấy kết quả đầu ra ta xem phản hồi response



Mở vào từng mục để test Web

Tham khảo thêm <http://petstore.swagger.io/>

## 2.2. Kiểm tra Web API với Postman

### 2.2.1. Giới thiệu về Postman

Postman là 1 công cụ để test API của cty Postdot Technologies được bắt đầu phát triển từ năm  
2012. Hiện tại Postman có 3 phiên bản: Postman, Postman Pro (2016) và Postman Enterprise  
(2017).

*Ưu điểm:*– Dễ sử dụng, hỗ trợ cả chạy bằng UI và non-UI.  
– Hỗ trợ viết code cho assert tự động bằng Javascript.  
– Hỗ trợ cả RESTful services và SOAP services.  
– Có chức năng tạo API document.  
*Nhược điểm:*– Những bản tính phí mới hỗ trợ những tính năng advance: Làm việc theo team, support trực  
tiếp…

### 2.2.2. Cài đặt Postman

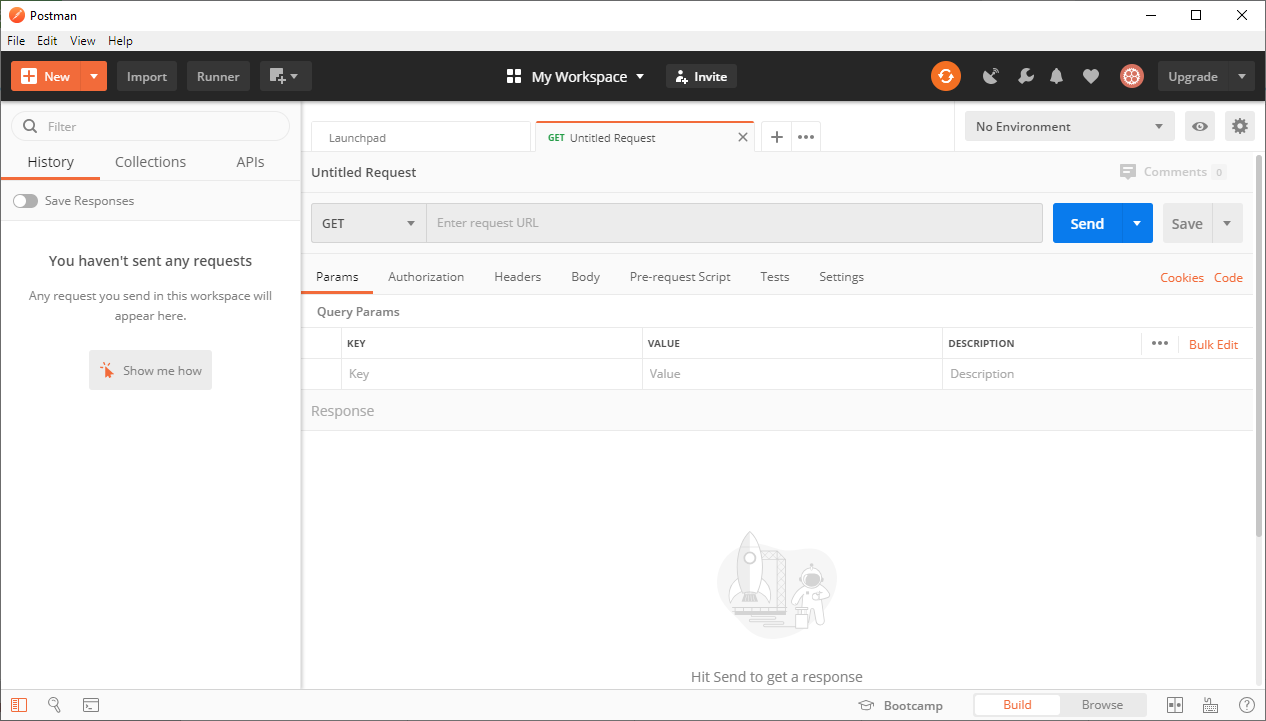
Download tại địa chỉ: <https://www.getpostman.com/>

Hoặc <https://www.postman.com/downloads/>

Sau đó, cài đặt như 1 phần mềm bình thường

Có thể đăng nhập vào tài khoản postman bằng tài khoản google or facebook

Cài đặt xong nó có giao diện thế này



Xem thêm Link hướng dẫn

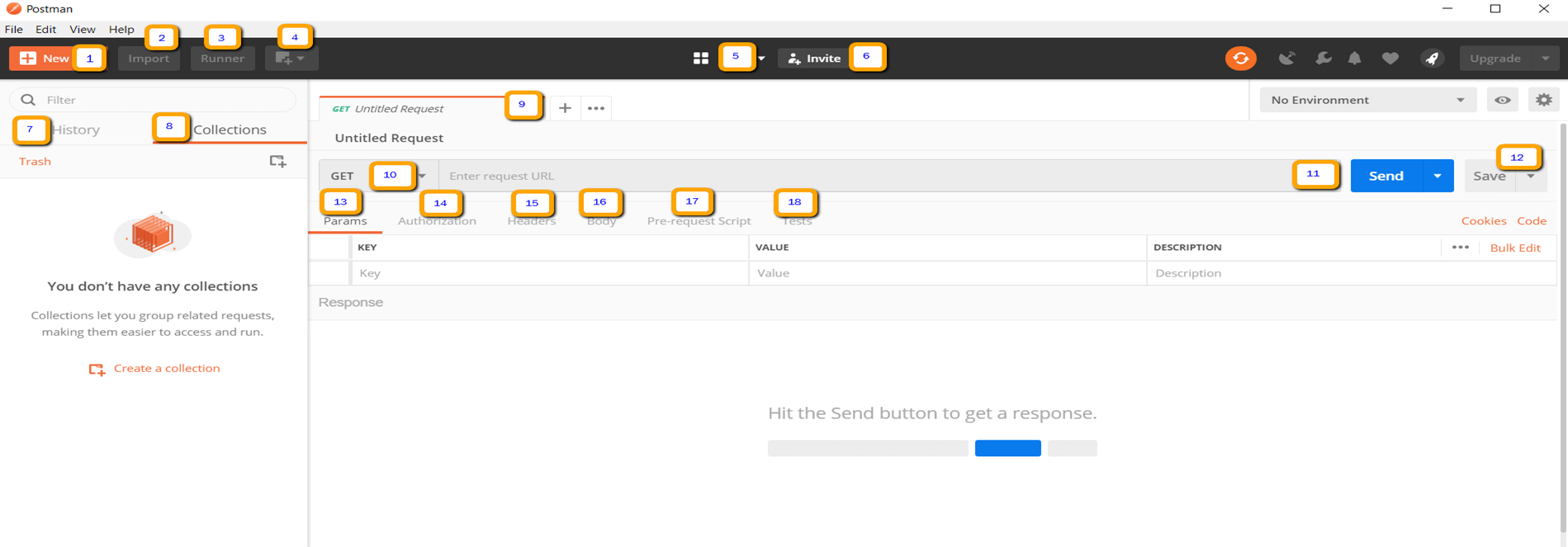
<https://www.guru99.com/postman-tutorial.html>

### 2.2.3. Cách sử dụng Postman

Bước 1 Tạo Collections

Bước 2 Tạo các requests trong Collections

Bước 3: Viết request tương ứng Http method, rồi nhấn send để gửi request, rồi chờ để nhận response từ web API

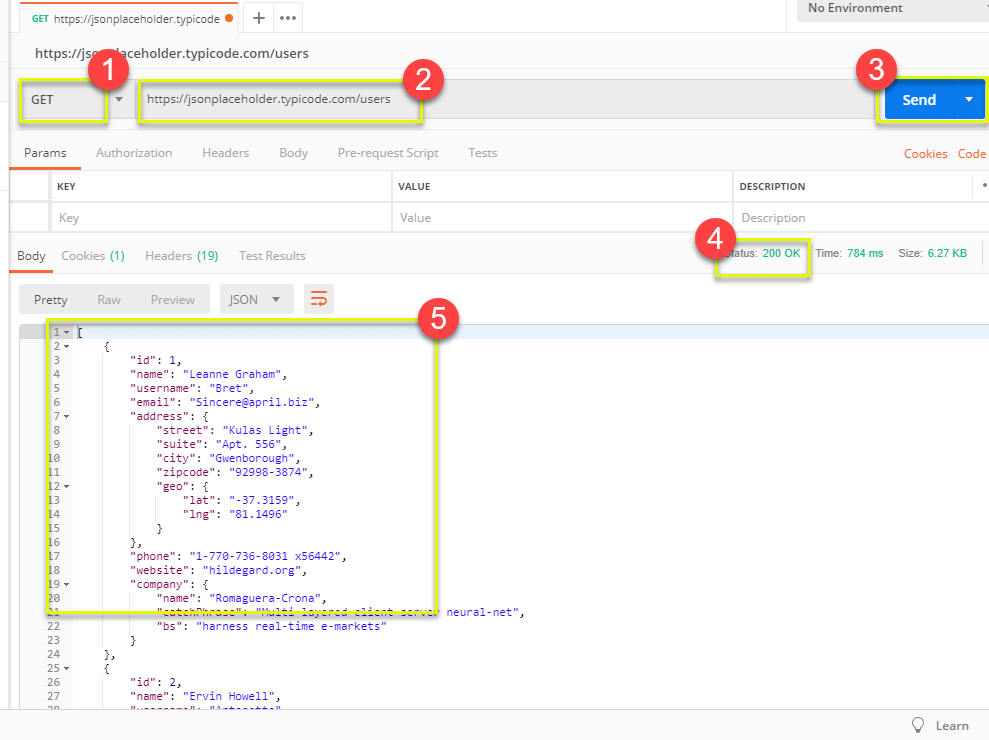


Giao diện Postman

1. **New** – Đây là nơi bạn sẽ tạo request, collection hoặc enviroment mới.
2. **Import** – Được sử dụng để import collection hoặc environment. Có các tuỳ chọn để import từ file, folder, link hoặc paste từ text thuần.
3. **Runner** – Kiểm tra tự động hóa có thể được thực hiện thông qua Runner cả collection. Điều này sẽ được thảo luận thêm trong bài học tiếp theo.
4. **Open New** – Mở một tab mới, cửa sổ Postman hoặc cửa sổ Runner bằng việc kích trên nút này.
5. **My Workspace** – Bạn có thể tạo sổ làm việc riêng hoặc như cho một nhóm.
6. **Invite** – Làm việc cộng tác với nhiều thành viên bằng việc mời các thành viên.
7. **History** – Các request đã thực hiện mà bạn đã thực hiện sẽ được hiển thị trong History. Giúp bạn có thể lần theo các hành động bạn đã làm.
8. **Collections** – Tổ chức bộ thử nghiệm của bạn bằng cách tạo collection. Mỗi collection có thể có các thư mục con và nhiều yêu cầu. Request hoặc thư mục cũng có thể được trùng lặp.
9. **Tab Request** – Hiển thị tiêu đề của requet mà bạn đang làm việc. Mặc định “Untitled Request” sẽ được hiển thị cho các request không có tiêu đề.
10. **HTTP Request** – Click vào đây sẽ hiển thị danh sách thả xuống với các request khác nhau như GET, POST, COPY, DELETE, v.v. Trong thử nghiệm, các yêu cầu được sử dụng phổ biến nhất là GET và POST.
11. **Request URL** – Còn được gọi là điểm cuối (endpoint), đây là nơi bạn sẽ xác định liên kết đến nơi API sẽ giao tiếp.
12. **Save** – Nếu có thay đổi đối với request, nhấp vào Save là bắt buộc để những thay đổi mới sẽ không bị mất hoặc bị ghi đè.
13. **Params** – Đây là nơi bạn sẽ viết các tham số cần thiết cho một request, ví dụ như các cặp key – value.
14. **Authorization** – Để truy cập API, cần được cấp quyền. Nó có thể ở dạng tên người dùng và mật khẩu, bearer token, v.v.
15. **Headers** – Bạn có thể thiết lập các header như nội dung kiểu JSON tùy theo cách tổ chức của bạn.
16. **Body** – Đây là nơi chúng ta có thể tùy chỉnh chi tiết trong request thường được sử dụng trong request POST.
17. **Pre-request Script** – Đây là các tập lệnh sẽ được thực thi trước request. Thông thường, script tiền request (pre-request) cho cài đặt môi trường được sử dụng để đảm bảo các kiểm tra sẽ được chạy trong môi trường chính xác.
18. **Tests** – Đây là các script được thực thi khi request. Điều quan trọng là phải có các thử nghiệm như thiết lập các điểm checkpoint để kiểm tra trạng thái là ok, dữ liêu nhận được có như mong đợi không và các thử nghiệm khác.

#### HttpGet Request

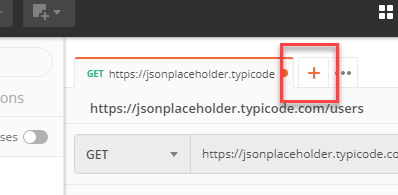
1. Thiết lập request HTTP của bạn là GET.
2. Trong trường URL yêu cầu, nhập vào link
3. Kích nút **Send**
4. Bạn sẽ nhìn thấy message ***200 OK***
5. Sẽ hiển thị kết quả 10 người dùng trong phần Body của bạn



**\*Chú ý:** Có thể có nhiều trường hợp request GET không thành công. Nó có thể thể do URL của request không hợp lệ hoặc do chứng thực không thành công

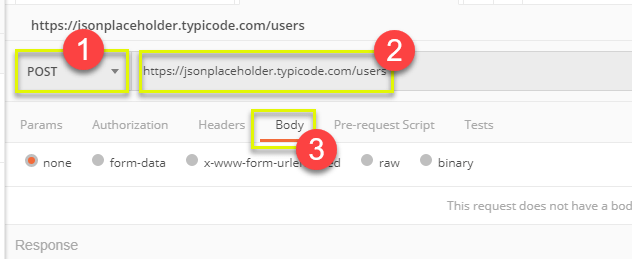
#### HttpPOST, HttpPut và HttpDelete Request

**Bước 1)** Kích dấu **+** để thêm mới một tab cho request mới.



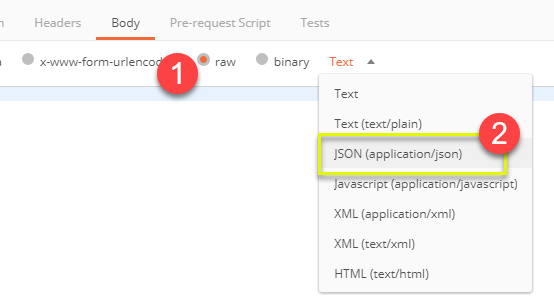
**Bước 2)** Trong tab mới

1. Thiết lập request HTTP là Post, Put hoặc Delete.
2. Nhập vào link với url: <https://jsonplaceholder.typicode.com/users>
3. Chuyển tới tab Body



**Bước 3)** Trong tab Body,

1. Kích chọn raw
2. Chọn JSON



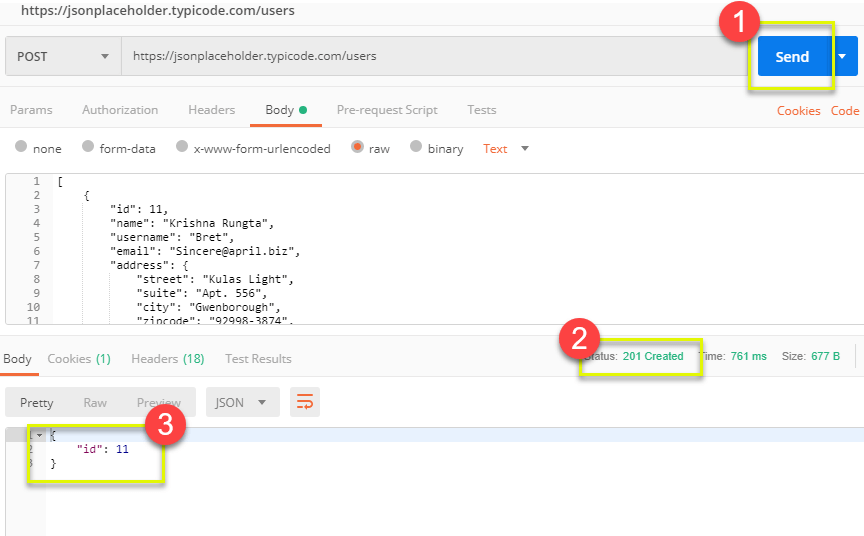
**Bước 4)** Copy và paste chỉ một user từ kết quả request trước như bên dưới. Đảm bảo rằng mã đã được sao chép chính xác với các dấu đóng mở. Thay đổi id thành 11 và đặt name bất kỳ tên nào bạn muốn. Bạn cũng có thể thay đổi các trường khác như address.

[{…}…{..}]

**\*Chú ý:** Request Post/Put/Delete nên đúng định dạng để đảm bảo dữ liệu được yêu cầu sẽ được tạo. Kinh nghiệm để check dữ liệu JSON đúng định dang hay không, bạn có thể sử dụng tool như: <https://jsonformatter.curiousconcept.com/>

**Bước 5)** Tiếp theo,

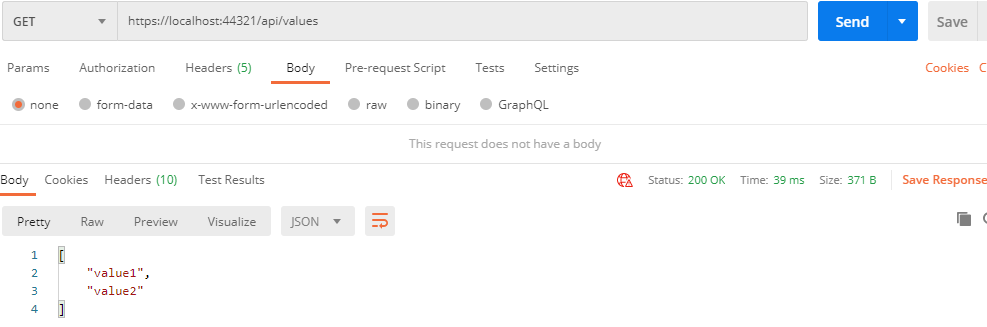
1. Kích nút Send.
2. Status: **201 Created** được hiển thị
3. Dữ liệu Post được hiển thị trong tab Body.

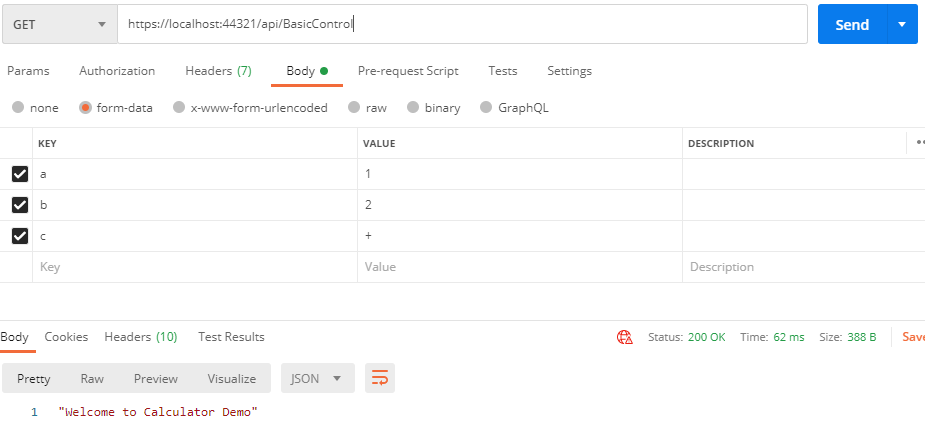


**Ví dụ** test WebAPi trong bài 1 như sau

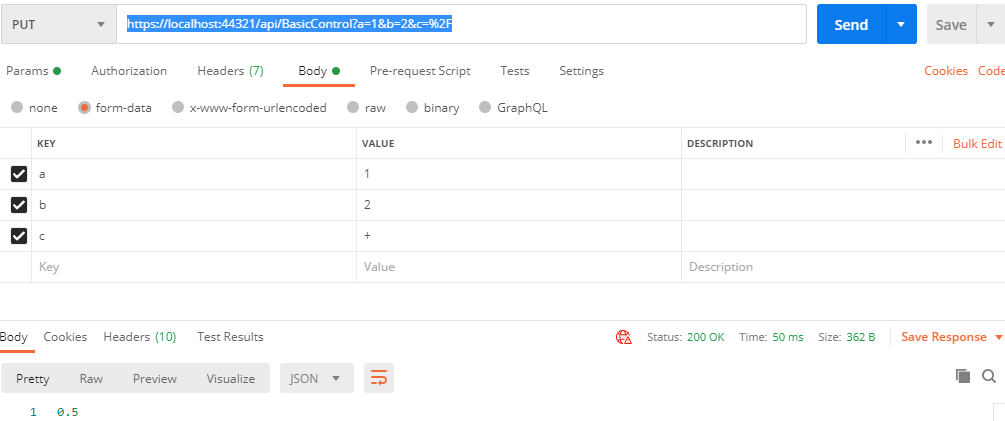
<https://localhost:44321/api/BasicControl>

<https://localhost:44321/api/values>





<https://localhost:44321/api/BasicControl?a=1&b=2&c=%2F>



## 2.3. Kiểm tra Web API với Fiddler

### 2.3.1. Giới thiệu về Fiddler

Fiddler là một tiện ích test resfull api và nghe, bắt các gói tin miễn phí được phát hành bởi hãng Telerik.

Fiddler hỗ trợ hầu hết các method resfull như get, post, delete, put, patch

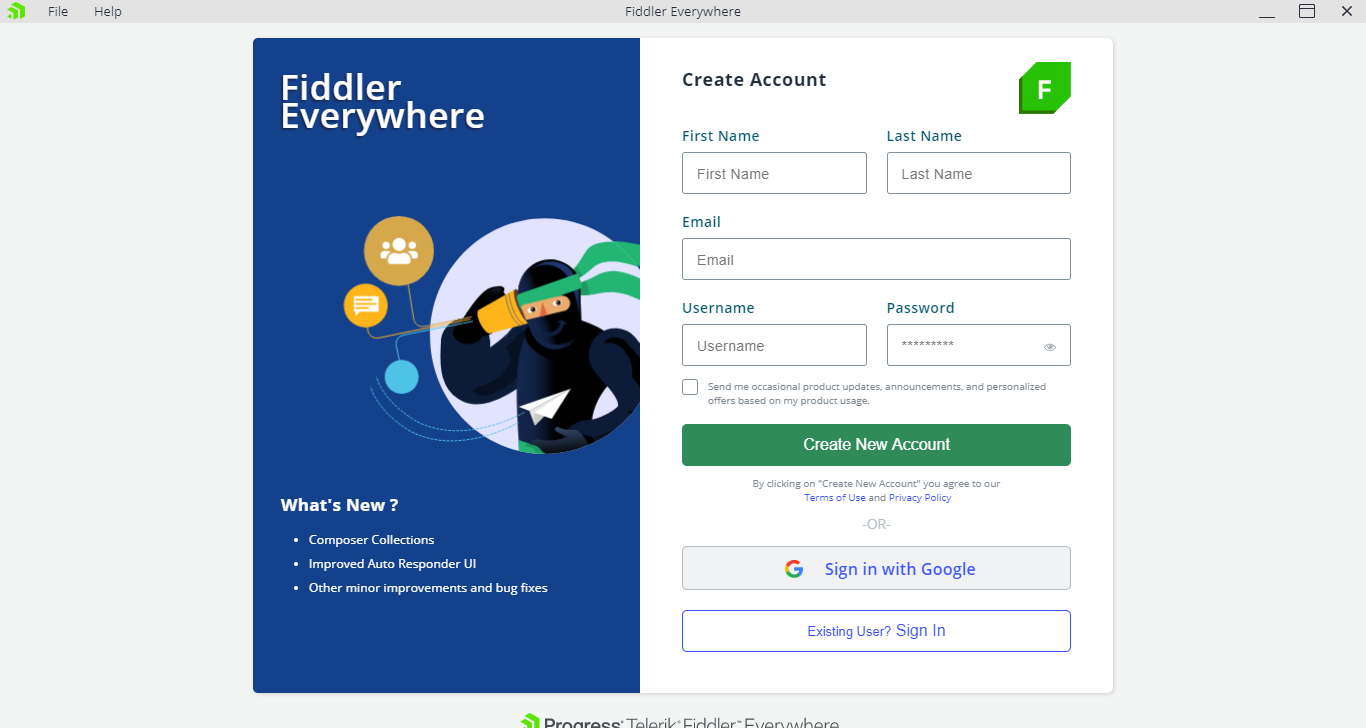
### 2.3.2. Cài đặt Fiddler

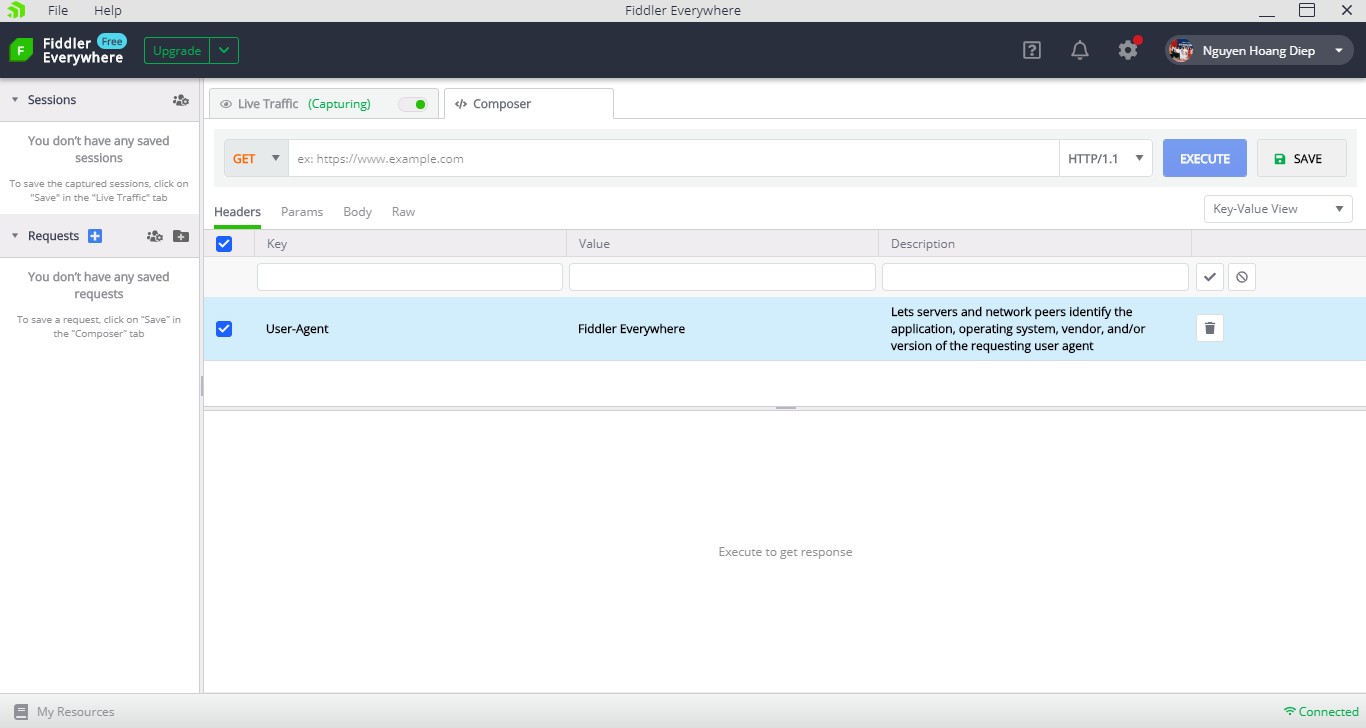
Download tại đường dẫn sau: <http://www.telerik.com/fiddler>

Sau đó, cài đặt như 1 phần mềm bình thường

Có thể đăng nhập vào tài khoản postman bằng tài khoản google or facebook

Cài đặt xong nó có giao diện thế này





### 2.3.3. Cách sử dụng Fiddler

**B1**: Chọn 1 result bất kỳ trong cửa số result

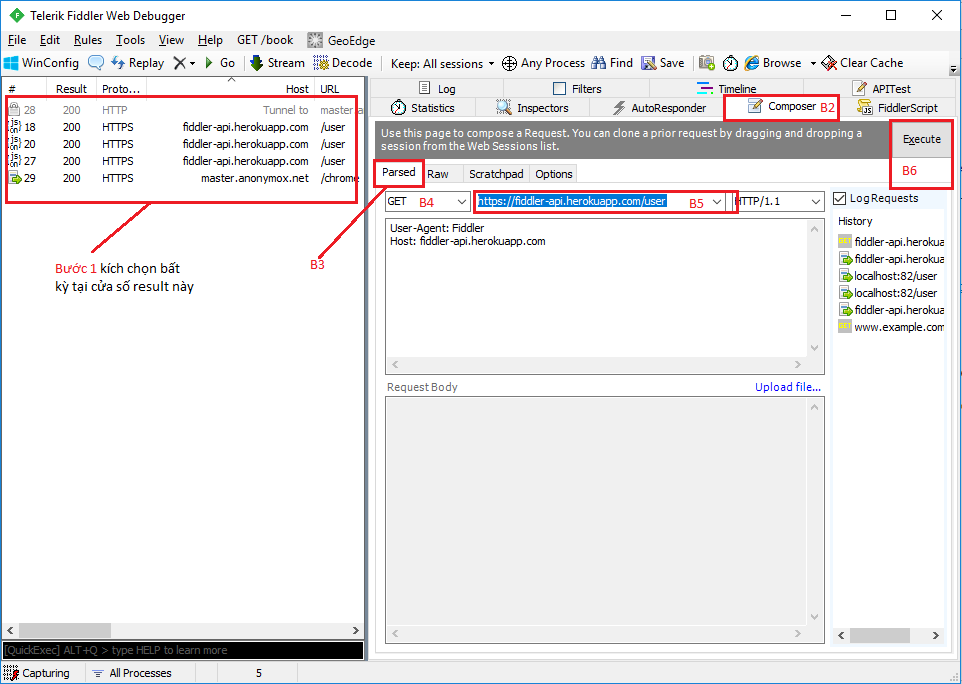
**B2**: Chọn Tab Composer

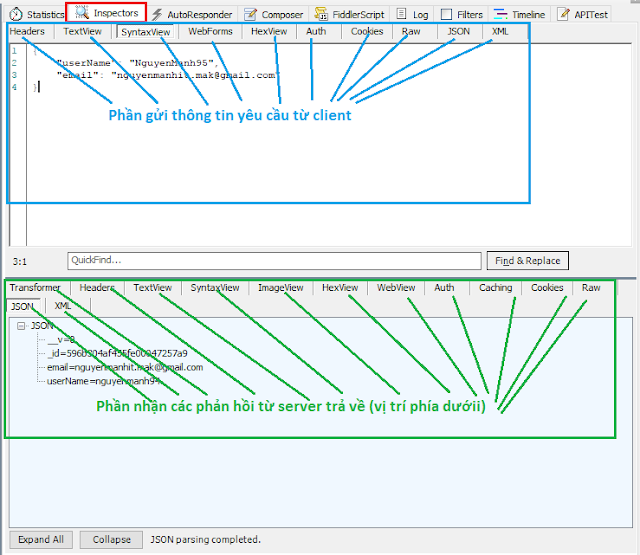
**B3**: Tab Parsed

**B4**: Chọn method sử dụng là GET

**B5**: Nhập url của bạn cần test vào; ở đây mình nhập URL: <https://fiddler-api.herokuapp.com/user>

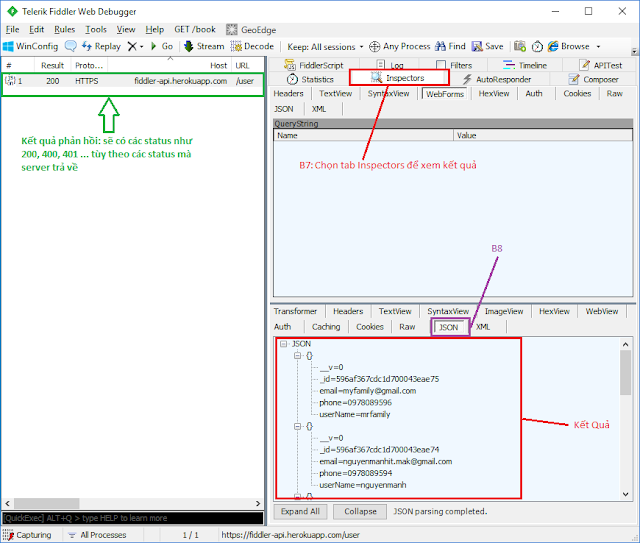
**B6**: Nhấn chọn execute.



[](https://1.bp.blogspot.com/-5rdZjuOzfX8/WWtAxrqo_8I/AAAAAAAACQY/9g6DAiJJ3CwQcU-JbNY2O80zyY3frgvzwCLcBGAs/s1600/Capture.PNG)

*Hình ảnh: Cửa sổ làm việc Inspectors*

#### HttpGet Request

[](https://3.bp.blogspot.com/-D1BaxLhinC0/WWs69-nZEEI/AAAAAAAACPs/sldFSSBRu4EXliWs2tMnryzfeHwM61X0wCEwYBhgL/s1600/Get2.png)

*Hình ảnh: Hiển thị kết quả sau khi thực thi method get*

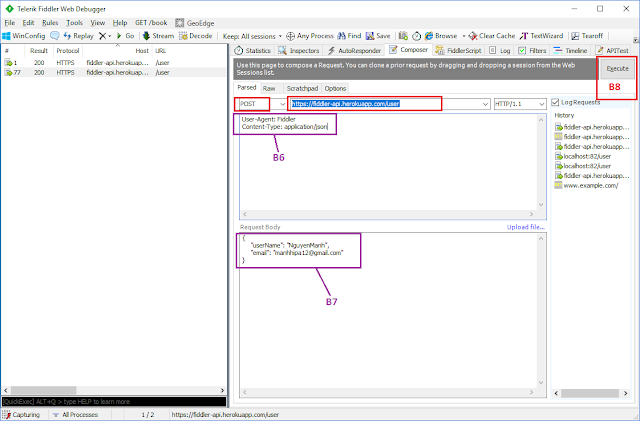
#### HttpPost, HttpPut và HttpDelete Request

Tiếp tục quay lại Tab Composer và thực hiện tương tự các bước từ 1 tới 5 như method GET, nhưng lựa chọn method là POST

B6: Thiết lập cho header request Content-type là application/json

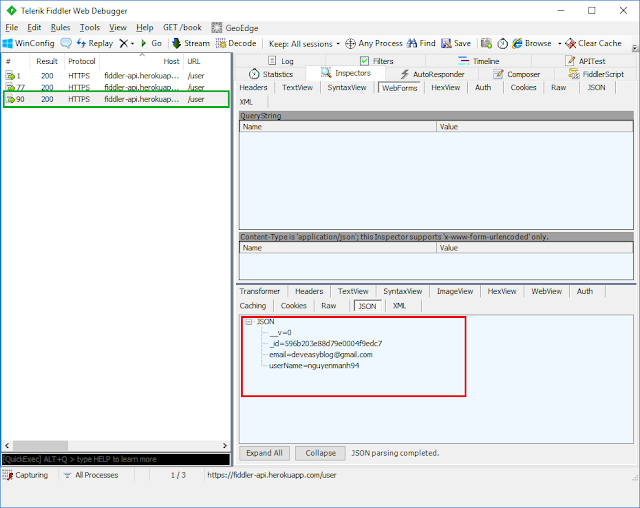
B7: Nhập vào chuỗi json data, chuỗi này chính là body gửi lên server.

B8: Nhấn Execute để test thử. (Xem thêm minh họa phần dưới)

[](https://4.bp.blogspot.com/-4pTs-s9RpV4/WWs690OInbI/AAAAAAAACP0/68hSuQwi0ggO_76oUzsWdXvw3OLQKbi3gCEwYBhgL/s1600/Post%2B1.png)

*Hình: Test post method*

    Bạn chuyển lại tab Inspectors để xem kết quả

[](https://2.bp.blogspot.com/-s8SAg6XbeVc/WWs6_OBIG5I/AAAAAAAACQI/l8y9gp3hoYocBkzNpgpnoSJPCFsCGUPPwCEwYBhgL/s1600/post2.png)

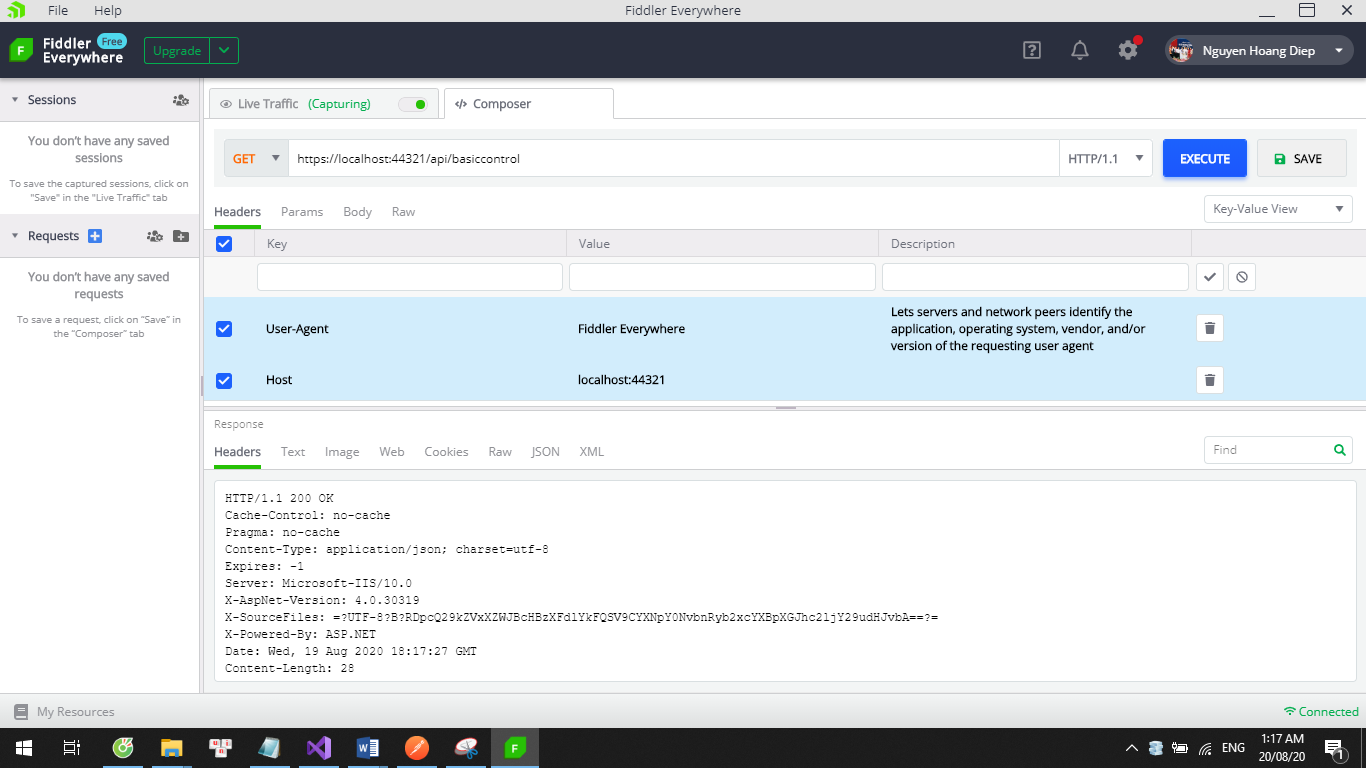
*Hình ảnh: Hiển thị kết quả sau khi thực thi post request*

    Các method *delete, path, put* cũng tương tự các bạn có thể tự test thử xem sao nhé.

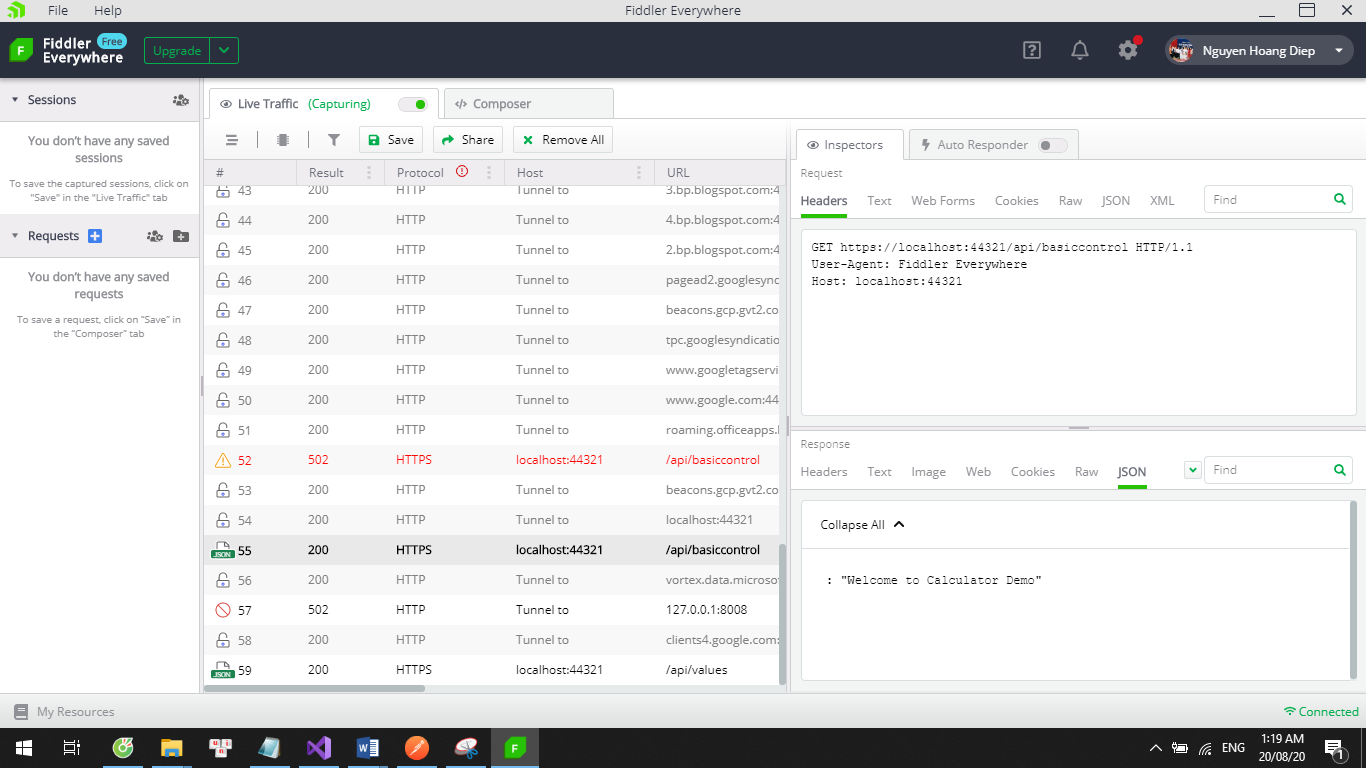
**Ví dụ** test WebAPi trong bài 1 như sau

<https://localhost:44321/api/basiccontrol>

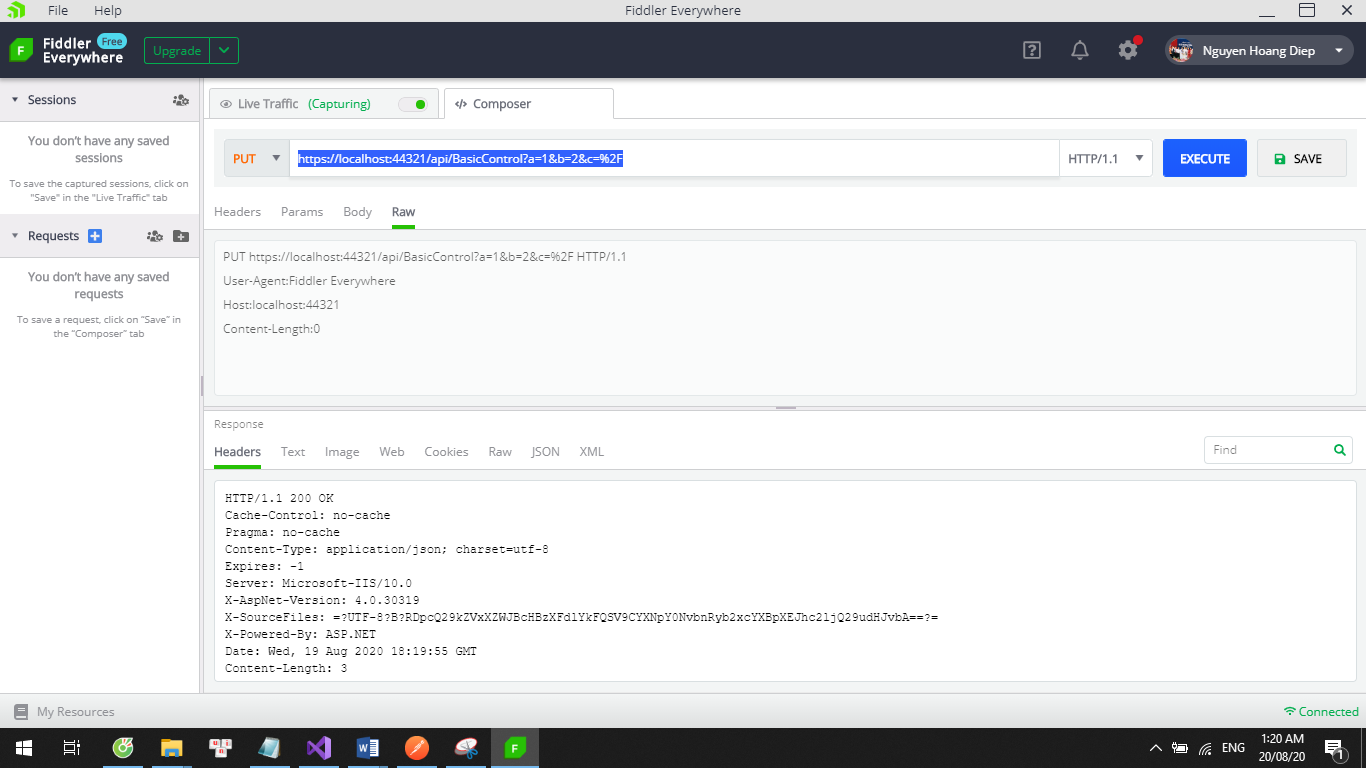
<https://localhost:44321/api/values>

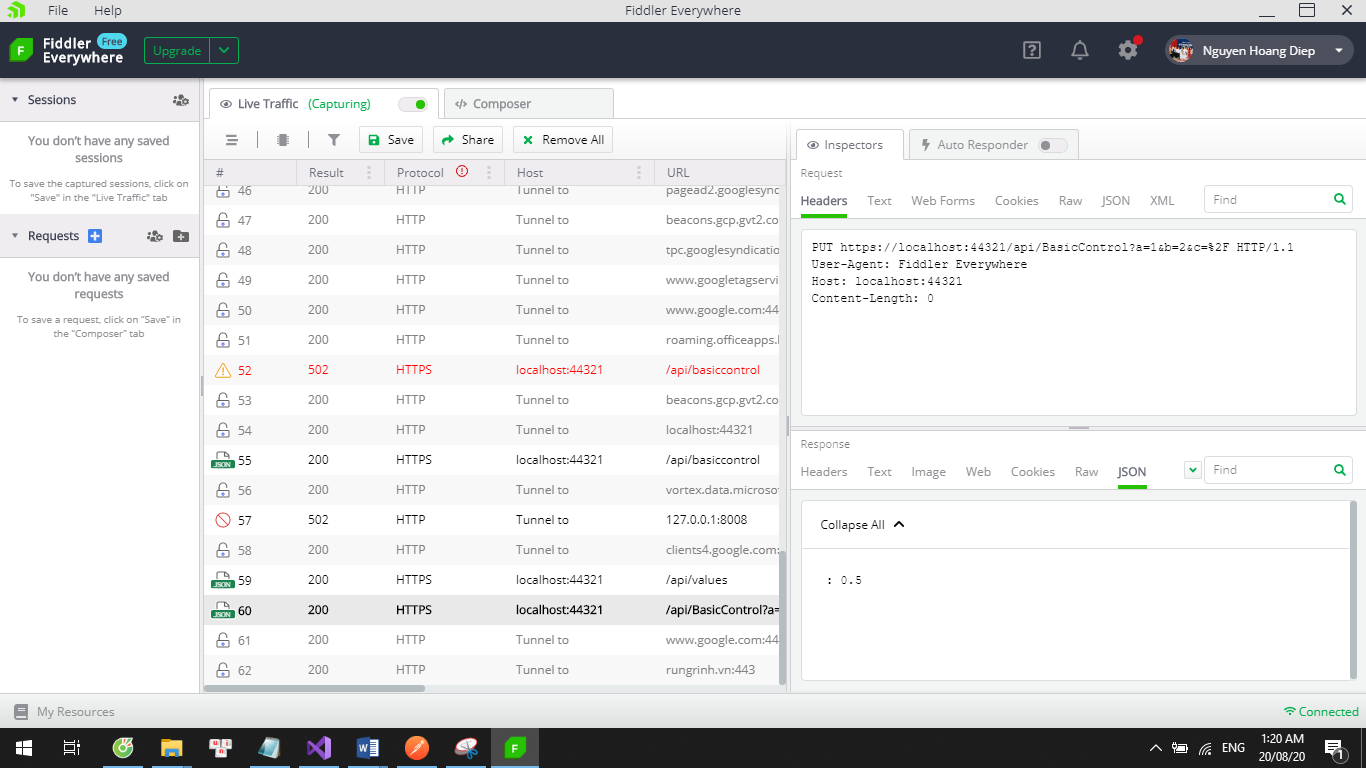


Kết quả



<https://localhost:44321/api/BasicControl?a=1&b=2&c=%2F>





Contents

[BÀI 2: CÔNG CỤ KIỂM TRA WEB API 1](#_Toc48777950)

[2.1. Kiểm tra Web API với Swagger 1](#_Toc48777951)

[2.1.1. Giới thiệu về Swagger 1](#_Toc48777952)

[2.1.2. Cài đặt Swagger 1](#_Toc48777953)

[2.2. Kiểm tra Web API với Postman 5](#_Toc48777954)

[2.2.1. Giới thiệu về Postman 5](#_Toc48777955)

[2.2.2. Cài đặt Postman 5](#_Toc48777956)

[2.2.3. Cách sử dụng Postman 6](#_Toc48777957)

[2.3. Kiểm tra Web API với Fiddler 12](#_Toc48777958)

[2.3.1. Giới thiệu về Fiddler 12](#_Toc48777959)

[2.3.2. Cài đặt Fiddler 12](#_Toc48777960)

[2.3.3. Cách sử dụng Fiddler 12](#_Toc48777961)