Bài 9
Web Service and RESTful

Module: BOOTCAMP WEB-BACKEND DEVELOPMENT WITH LARAVEL 2.1

Mục tiêu

- Trình bày được khái niệm Web service
- Trình bày được các loại web service cơ bản
- Mô tả được RESTful Web service
- Triển khai được RESTful Web service
- Sử dụng được POSTMAN để kiểm thử web service

Thảo luận

Web service

Web Service (1)

- Web service (Dịch vụ web) là các thành phần ứng dụng dưới dạng các dịch vụ trên WWW.
- Dịch vụ
  - Là một thành phần phần mền
  - Chứa một vàu logic nghiệp vụ bên trong đó
  - Được hiển thị trên Web cho nhiều loại client
  - Được truy cập bởi các client tại các vị trí khác nhau
- Web Service có thể sử dụng để tích hợp với các ứng dụng được viết bằng các ngôn ngữ khác nhau và chạy trên các nền tảng khác nhau.
- Xây dựng các chuẩn mở và sử dụng các giao thức mở để giao tiếp.

Web Service (2)

- Web service hoạt động như một server trong mô hình ứng dụng client server sử dụng giao thức HTTP/HTTPs và chỉ thực hiện một tác vụ cụ thể.
- Với dữ liệu đầu vào xác định, web server xử lý và trả ra dữ liệu đầu ra theo chuẩn đảm bảo mọi ứng dụng có thể hiểu và sử dụng mà không quan tâm đến loại thiết bị, hệ điều hành, kiến trúc phần mềm hay ngôn ngữ được sử dụng.
- Kiểu dữ liệu đầu ra phổ biến của một web service thường là XML hoặc JSON.

Phân biệt Website và Web Service (1)

Website	Web Service
Có giao diện người dùng hoặc GUI	Không có giao diện người dùng
Nói đến Website được hiểu sử dụng bởi con người	Nói đến Web Service được hiểu sử dụng bởi các ứng dụng được tương tác với nhau qua internet
Website hoạt động đa nền tảng, vì chúng yêu cầu tinh chỉnh để hoạt động trên các trình duyệt hay hệ điều hành khác nhau.	Web Service độc lập về nền tảng và tất cả dạng truyền thông đều sử dụng giao thức chuẩn.
Website được truy cập bởi các thành phần trong giao diện người dùng như button, textbox, form ...	Webservice được truy cập bởi phương thức HTTP – PUT, GET, POST, DELETE ...

Phân biệt Website và Web Service (2)

Website	Web Service
Ví dụ. ArtOfTesting.com là trang web có bộ sưu tập các trang web có liên quan chứa hướng dẫn	Ví dụ: Google Maps API là một dịch vụ web có thể được sử dụng bởi các trang web để hiển thị bản đồ bằng cách chuyển tọa độ tới đó.
Website là một ứng dụng đầu cuối. Người sử dụng truy cập website qua URL qua đó nhận được những dữ liệu text, hình ảnh, âm thanh… có thể dễ dàng hiểu được.	Web service là một khái niệm rộng hơn, dữ liệu trả ra từ web service người sử dụng thông thường khó để hiểu. Dữ liệu đó được các ứng dụng (trong đó có web site) sử dụng và chế biến thành dữ liệu có thể đọc cho người sử dụng.

Ưu điểm của web service

- Hoạt động trên các ứng dụng, nền tảng, hệ điều hành, ngôn ngữ khác nhau.
- Khả năng tái sử dụng cao.
- Tạo mối quan hệ tương tác, mềm dẻo trong hệ thống phần mềm, dễ dàng cho việc phát triển ứng dụng phân tán.
- Giảm sự phức tạp của hệ thống, giảm thời gian phát triển hệ thống, hạ giá thành hoạt động, dễ dàng tương tác giữa các hệ thống với nhau.

Nhược điểm của web service

- Khi một web service chết hoặc dừng hoạt động sẽ gây lỗi, thiệt hại lớn trên tất cả các hệ thống, thiết bị đang sử dụng web service đó.
- Cần quan tâm đến vấn đề an toàn và bảo mật nhiều hơn khi sử dụng web service.
- Việc có quá nhiều chuẩn cho web service dẫn đến người sử dụng khó nắm bắt.

Các loại web service cơ bản

- SOAP (Simple Object Access Protocol) là giao thức sử dụng XML để định nghĩa dữ liệu dạng thuần văn bản (plain text) và truyền dữ liệu thông qua HTTP.
- REST (Representational State Transfer) là một kiểu cấu trúc cung cấp các quy tắc để xây dựng web service.
  - REST định nghĩa dữ liệu dưới dạng XML hoặc JSON và truyền thông qua mạng internet sử dụng giao thức HTTP.
  - Các web service xây dựng dựa trên REST được gọi là RESTful, chúng chủ yếu nhằm xử lý các hoạt động CRUD (Create/ Read/ Update/ Delete) trên dữ liệu.

Thảo luận

RESTful web service

REST

- REST (Representational State Transfer) được sử dụng để tạo ra và giao tiếp với Web Service thay thế mô hình SOAP.
- REST có kiến trúc đón giản, định rõ các ràng buộc nhằm tạo ra ứng dụng Web Service đạt được những tính chất mong muốn về hiệu suất, khả năng mở rộng, khả năng điều chỉnh v.v..
- REST hướng tới việc xây dựng ứng dụng Web Service có khả năng làm việc tốt nhất trên môi trường WWW.
- Dữ liệu và các tính năng được coi như tài nguyên và được truy xuất thông qua các URI (Uniform Resource Identifier)
- Sử dụng 4 phương thức chính của HTTP là POST, GET, PUT và DELETE để thực hiện các hành động CRUD đối với cá tài nguyên.

RESful Web Service

- RESTful web service là một web service được xây dựng dựa trên cấu trúc REST.
- RESTful web service tuân thủ các quy tắc cơ bản của cấu trúc REST gồm:
  - Sử dụng các phương thức HTTP một cách rõ ràng
  - Phi trạng thái
  - Hiển thị cấu trúc thư mục như các URIs
  - Truyền tải **JavaScript Object Notation (JSON)**, **XML** hoặc cả hai.

RESful Web Service: Phương thức HTTP

- Sử dụng các phương thức HTTP một cách rõ ràng
  - Để tạo một tài nguyên trên máy chủ, sử dụng phương thức **POST**.
  - Để truy xuất một tài nguyên, sử dụng phương thức **GET**.
  - Để thay đổi trạng thái một tài nguyên hoặc để cập nhật nó, sử dụng phương thức **PUT**.
  - Để huỷ bỏ hoặc xoá một tài nguyên, sử dụng phương thức **DELETE**.

- Chú ý: Những quy tắc nêu trên là không bắt buộc nhưng REST khuyến nghị sử dụng chúng để tạo sự rõ ràng, dễ hiểu.

RESful Web Service: Stateless

- Phi trạng thái (Stateless)
  - RESTfull web service không lưu lại thông tin, phiên làm việc của client.
  - Mỗi lần gọi RESTful web service, client phải cung cấp đầy đủ các thông tin đầu vào. Web service sẽ xử lý và trả ra dữ liệu mà không quan tâm tới các lần request trước đó từ client.
- Ưu điểm:
  - Các thành phần máy chủ phi trạng thái sẽ có thiết kế ít phức tạp hơn.
  - Dễ dàng viết và phân bổ client đến server thông qua cân bằng tải.
- Nhược điểm:
  - Tiêu tốn tài nguyên băng thông hơn, do mỗi lần gọi web service, client phải truyền đầy đủ các thông tin.
  - Server thực thi web service phải hoạt động nhiều hơn do mỗi yêu cầu từ client đều phải xử lý như nhau, không tận dụng được tài nguyên, tính toán trước đó.

RESful Web Service: URIs

- Hiển thị cấu trúc thư mục như các URIs
  - REST đưa ra một cấu trúc để người dùng có thể truy cập vào tài nguyên của nó thông qua các URL.
  - Các tài nguyên bao gồm text, hình ảnh, âm thanh, video… được xác định thông qua định danh như một địa chỉ URI (*Uniform Resource Identifier) đảm bảo* rõ ràng, dễ hiểu và dễ đoán cho người sử dụng.
- Ví dụ:
  - URL của một web service lấy thông tin thời tiết của Hà Nội vào ngày 5/6/2018: http://myservice.com/weather/hanoi/2018-06-05

RESful Web Service: JSON hoặc XML

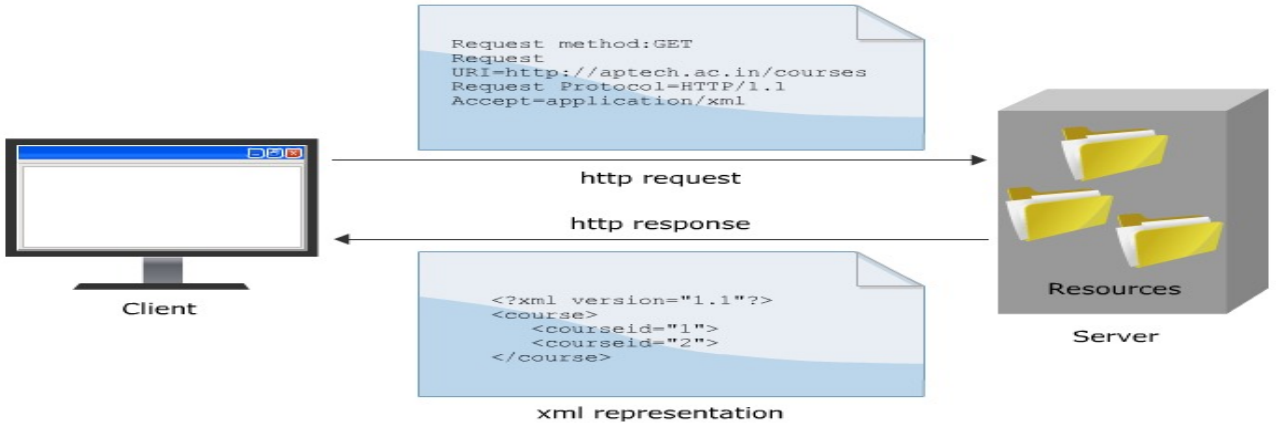
- Truyền tải **JavaScript Object Notation (JSON)**, **XML** hoặc cả hai.
  - Một client gửi yêu cầu đến web service được truyền tải dưới dạng XML hoặc JSON và dữ liệu trả về từ web service thông thường cũng tương tự.
  - Client có thể quyết định dữ liệu trả về từ web service bằng việc thêm chỉ định vào HEADER của request thông qua từ khóa Accept. Các chỉ định này được gọi là kiểu MINE.
- Ví dụ: Client chỉ định nhận dữ liệu trả về là kiểu JSON

  GET /weather/hanoi/2018-06-05 HTTP/1.1

  Host: myservice.com

  Accept: application/json

Giao tiếp Client – RESTful Web Service



1.Client gửi (request) yêu cầu về một tài nguyên cụ thể nào đó tới Web Service sử dụng các phương thức của HTTP

2.Web Service xác định phương thức, tài nguyên và thực hiện các hành động CRUD tương ứng

3.Web Service gửi (response) về cho client tài nguyên theo định dạng mà client yêu cầu

Thảo luận

Triển khai RESTful

RESTful API trong Laravel

Route api.php

- Các URI của RESTful API được khai báo trong routes/api.php
- Ví dụ:

```
Route::get('articles', 'ArticleController@index');
Route::get('articles/{id}', 'ArticleController@show');
Route::post('articles', 'ArticleController@store');
Route::put('articles/{id}', 'ArticleController@update');
Route::delete('articles/{id}', 'ArticleController@delete');
```

Trả về JSON

- Sử dụng phương thức json() để trả về dạng JSON
- Ví dụ:

```
public function store(Request $request) {
    $article = Article::create($request->all());
    return response()->json($article, 201);
}
```

HTTP Status

- HTTP Status dùng để mô tả trạng thái của response
- Một số trạng thái thường gặp:

Mã trạng thái	Tên trạng thái	Ý nghĩa
200	OK	Thành công
201	Object created	Đối tượng đã được tạo
204	No content	Không có dữ liệu trả về
206	Partial content	Trả về một phần nội dung
400	Bad request	Request không hợp lệ
401	Unauthorized	Client cần phải xác thực
403	Forbidden	Client không có quyền truy cập
404	Not found	Không tìm thấy tài nguyên yêu cầu
500	Internal server error	Lỗi do server
503	Service unavailable	Dịch vụ yêu cầu không khả dụng

Quy ước sử dụng URI

- Sử danh từ, không sử dụng động từ
- Ví dụ:

URI	**GET** read	**POST** create	**PUT** update	**DELETE** delete
/cars	Trả về danh sách xe	Tạo một xe mới	Cập nhật nhiều xe	Xoá tất cả xe
/cars/711	Trả về một xe	Method not allowed (405)	Cập nhật một xe	Xoá một xe

URI: Không sử dụng động từ

- Ví dụ không tốt:

/getAllCars

/createNewCar

/deleteAllRedCars

URI: Không sử dụng GET tùy tiện

- Không sử dụng GET cho các thao tác có thay đổi trạng thái của đối tượng
- Ví dụ không tốt:

GET /users/711?activate

GET /users/711/activate

URI: Sử dung số nhiều

- Ví dụ:

Tốt	**Không tốt**
/cars	/car
/users	/user
/products	/product
/settings	/setting

URI: Tài nguyên con (sub-resource)

- Sử dụng sub-resource để truy cập đến các mối quan hệ
- Ví dụ:

URI	Ý nghĩa
GET /cars/711/drivers/	Trả về danh sách driver của car 711
GET /cars/711/drivers/4	Trả về driver có id là 4 của car 711

Demo

Triển khai RESTful

Thảo luận

Sử dụng POSTMAN

Giới thiệu POSTMAN

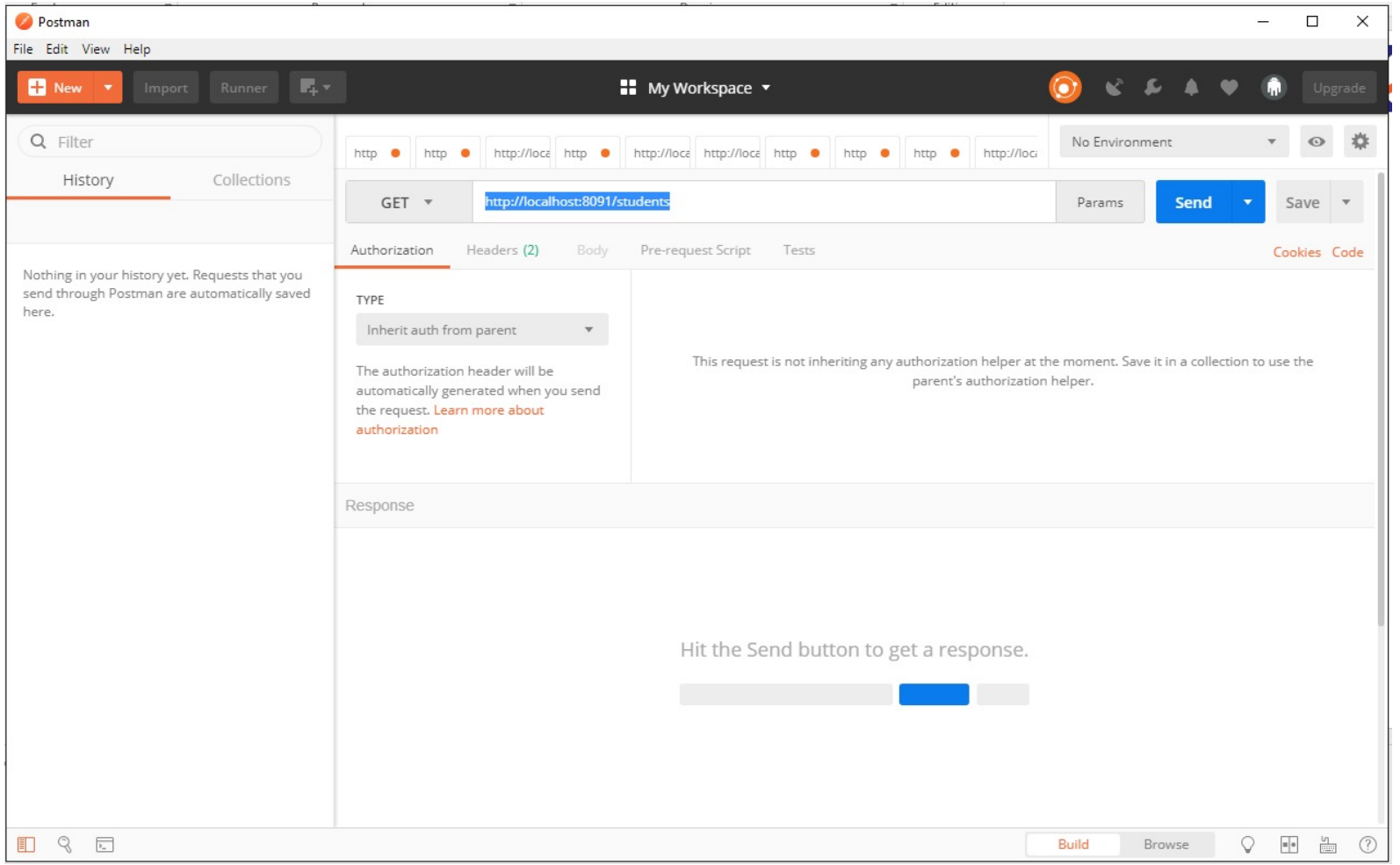
- POSTMAN là một App Extensions, cho phép làm việc với các API, điển hình là REST.
- Hỗ trợ tất cả các phương thức HTTP (GET, POST, PUT, DELETE, OPTIONS, HEAD ...)

Cài đặt POSTMAN

- POSTMAN được cung cấp miễn phí qua web site https://www.getpostman.com/

Sử dụng POSTMAN

- Giao diện chính



Sử dụng POSTMAN

- B1: Nhập URL của web service vào thanh địa chỉ của POSTMAN.
- B2: Chọn phương thức của web service.
- B3: Sửa, thêm các thông tin params phụ thuộc vào web service.
- B4: Nhấn SEND và theo dõi kết quả web service trả ra.

Sử dụng POSTMAN

- Với web service trên, khi kiểm thử với POSTMAN cho kết quả trả về với kiểu chỉ định JSON như sau:
- http://localhost:8080/students

Sử dụng POSTMAN

- Để kiểm thử web service sử dụng method POST, PUT…cần phải custom RequesHeader và RequestBody.
- Mục đích của việc custom là truyền thêm dữ liệu cần thiết cho web service.

Sử dụng POSTMAN

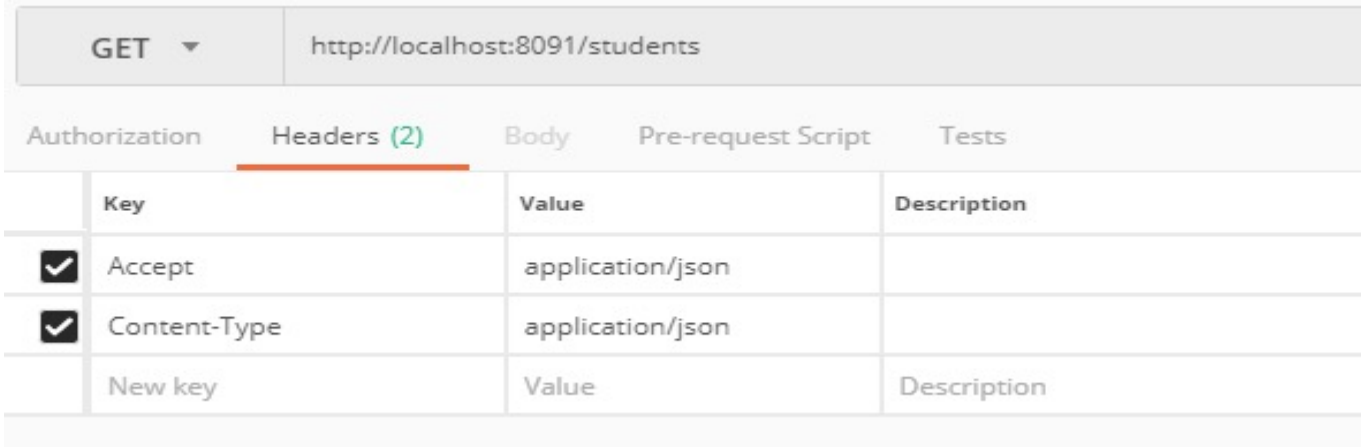
- Custom RequestHeader
  - Trên màn hình POSTMAN chuyển sang tab Header.
  - Để chỉ định kiểu dữ liệu trả về từ web service là JSON hoặc XML.

    Accept: Application/json

    Accept: Application/xml
  - Để chỉ định kiểu dữ liệu client gửi lên server là JSON, XML…

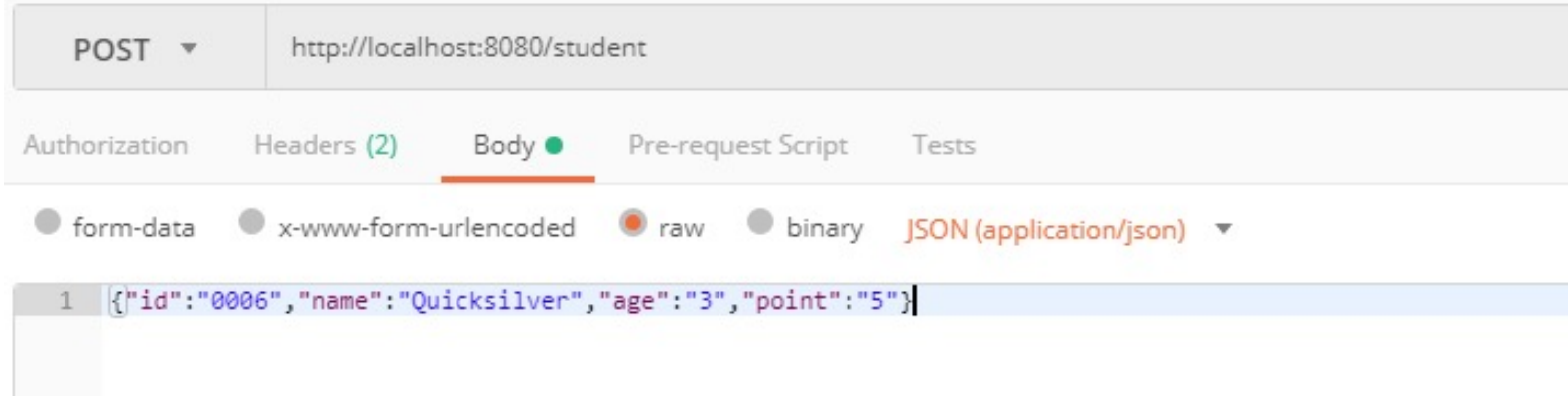
    Content-Type: Application/json

    Content-Type: Application/xml



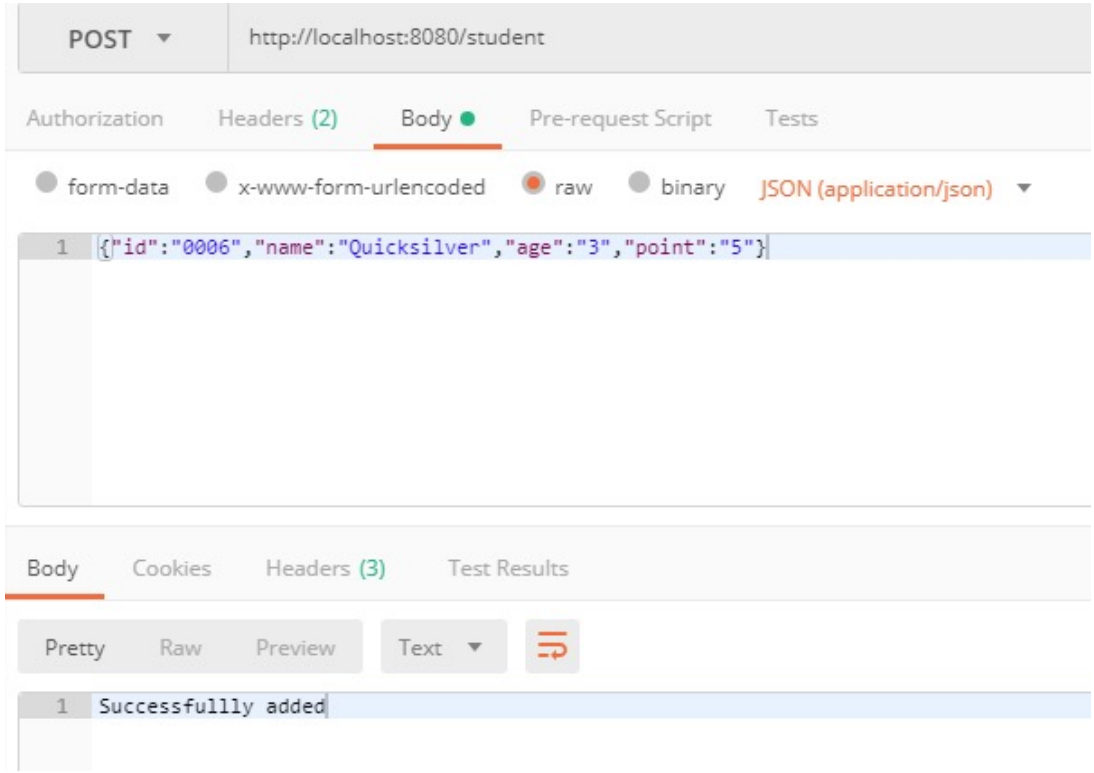
Sử dụng POSTMAN

- Custom RequestBody
  - Trên màn hình POSTMAN chuyển sang tab Body.
  - Xác định kiểu dữ liệu của request body, trong ví dụ là kiểu dữ liệu application/json. Nội dung của request body là chuỗi json của một đối tượng student.



Sử dụng POSTMAN

- Sau khi hoàn tất việc custom nhấn SEND để gọi web service thêm 1 đối tượng student mới.



- Message báo thành công, sử dụng web service getAllStudent để kiểm tra danh sách.

Demo

Sử dụng POSTMAN

Tóm tắt bài học

Hướng dẫn

- Hướng dẫn làm bài thực hành và bài tập
- Chuẩn bị bài tiếp: ***Internationalization***