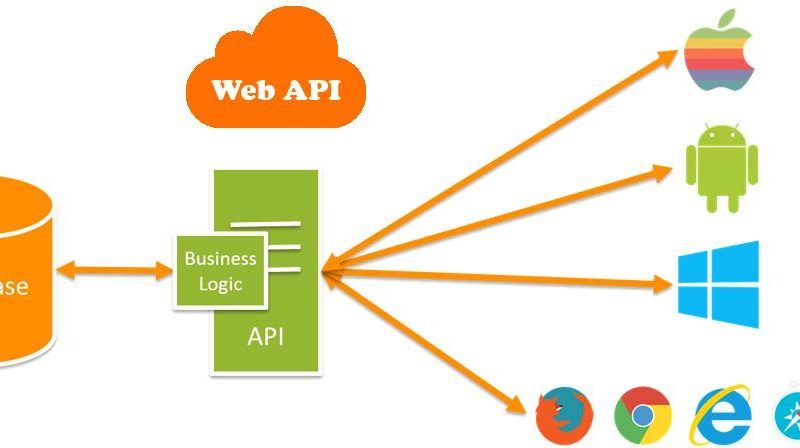
**CHỦ ĐỀ TÌM HIỂU: RESTFUL API**

1. **API**
2. ***Khái niệm:*** API (**A**pplication **P**rograming **I**nterface)

* API là giao diện lập trình ứng dụng. Giao diện ở đây không phải là giao diện của phần mềm, không phải là những khối màu, bố cục của phần mềm mà mắt bạn nhìn thấy. Giao diện ở đây giống như một chuẩn chung để kết nối vậy. Ví dụ như cái ổ cắm với cái phích cắm, mặc dù chúng có thể đến từ hai nhà sản xuất khác nhau nhưng khi cắm vào nhau thì chúng vẫn vừa khít, đấy là do chúng cùng tuân theo một giao diện kết nối.
* API có thể trả về dữ liệu mà bạn cần cho ứng dụng của mình ở những kiểu dữ liệu phổ biến như JSON hay XML.



1. ***Phân Loại***

* **API mở (Open API):** Có sẵn, công khai, không hạn chế quyền truy cập.
* **API đối tác (Partner API):** Cần có quyền hoặc giấy phép cụ thể mới truy cập được.
* **API nội bộ (Internal API):** Chỉ dùng cho các hệ thống nội bộ (chẳng hạn như công ty, tổ chức). Các đội ngũ phát triển nội bộ khác nhau có thể sử dụng chúng để cải thiện cho các sản phẩm hay dịch vụ chỉ phục vụ cho riêng nhân viên công ty.

1. ***Ưu , nhược điểm***

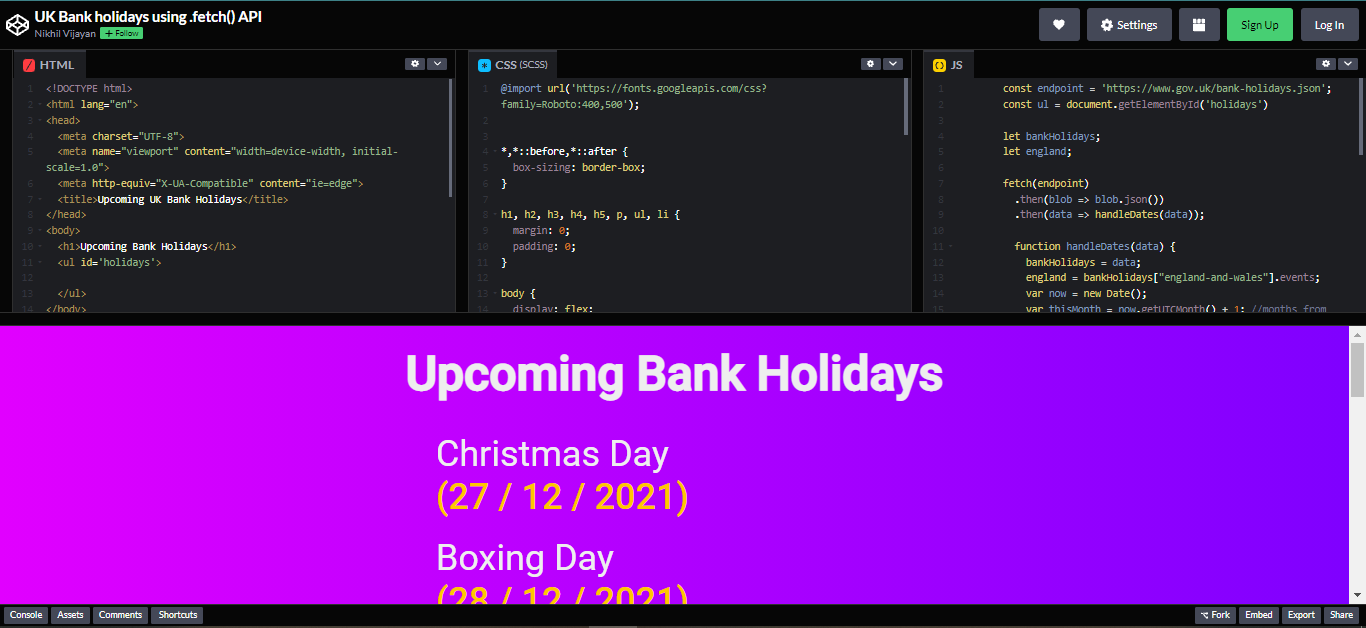
* Ưu điểm:
* **Tính tự động hóa rất cao:** API có thể thay thế chúng ta quản lý công việc cực kì hiệu quả. API giúp các cơ quan có thể cập nhật, xử lý hoàn thiện công việc nhanh và chất lượng hơn.
* **Ứng dụng cực kỳ linh hoạt:** API có thể truy cập vào các thành phần ứng dụng giúp việc cung cấp dịch vụ và thông tin linh hoạt hơn nhiều.
* **Khả năng thích ứng tốt:** API có chức năng thay đổi cũng như dự đoán thay đổi theo thời gian cho nên dữ liệu được truyền tốt hơn, thông tin được chọn lọc kĩ hơn, dịch vụ tốt hơn.
* **Cá nhân hóa:** Người dùng có thể tinh chỉnh API cho phù hợp nhu cầu sử dụng.
* **Dữ liệu có sẵn:** API cho phép mọi thông tin được tạo ở dạng chính chủ luôn có sẵn cho mọi người xem được.
* **Phạm vi:** Ngoài dữ liệu mới có sẵn được chia sẻ rộng rãi, người dùng còn có thể hiệu chỉnh web API để cung cấp dịch vụ, thông tin cá nhân hóa.
* **Nhược điểm:**
* API **chỉ hỗ trợ mặc định get, post** chưa hoàn toàn là restful service.
* Cấu hình cố hữu của wcf làm cấu hình của **API cực kì khó nhớ**. Nếu mới sử dụng, người dùng khó mà nhớ và dùng dễ dàng như các ứng dụng khác.
* **Web API tốn kha khá chi phí** vận hành, phát triển, hiệu chỉnh và đòi hỏi kiến thức chuyên sâu. Khi bị tấn công hệ thống, người dùng đôi lúc gặp rắc rối về bảo mật.

1. ***Ví dụ:***

* API bank holiday: https://www.gov.uk/bank-holiday.json



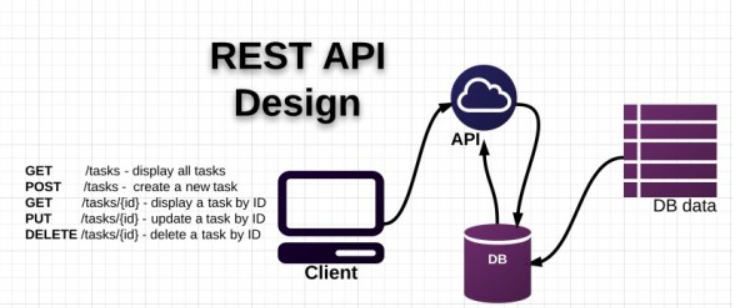
* Và đây là kết quả:



1. **Restful API là gì?**

* ***REST là gì?***
* **REST** (**RE**presentational **S**tate **T**ransfer) là một kiểu kiến ​​trúc phần mềm được tạo ra để hướng dẫn thiết kế và phát triển kiến ​​trúc cho World Wide Web. REST định nghĩa một tập hợp các ràng buộc đối với cách kiến ​​trúc của một hệ thống siêu phương tiện phân tán quy mô Internet, chẳng hạn như Web, sẽ hoạt động như thế nào.

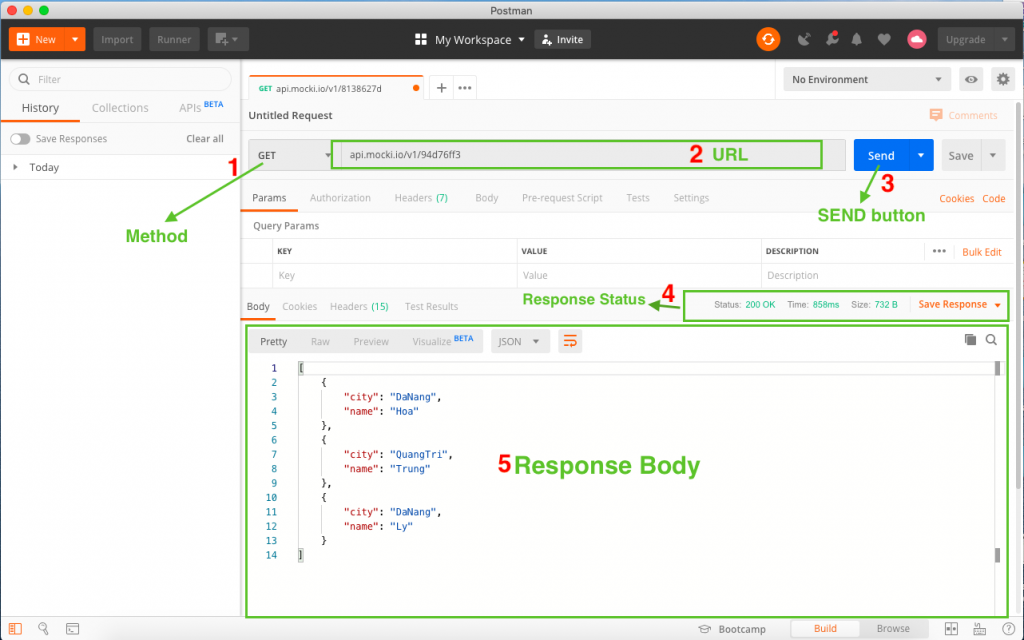
1. **Khái niệm:**



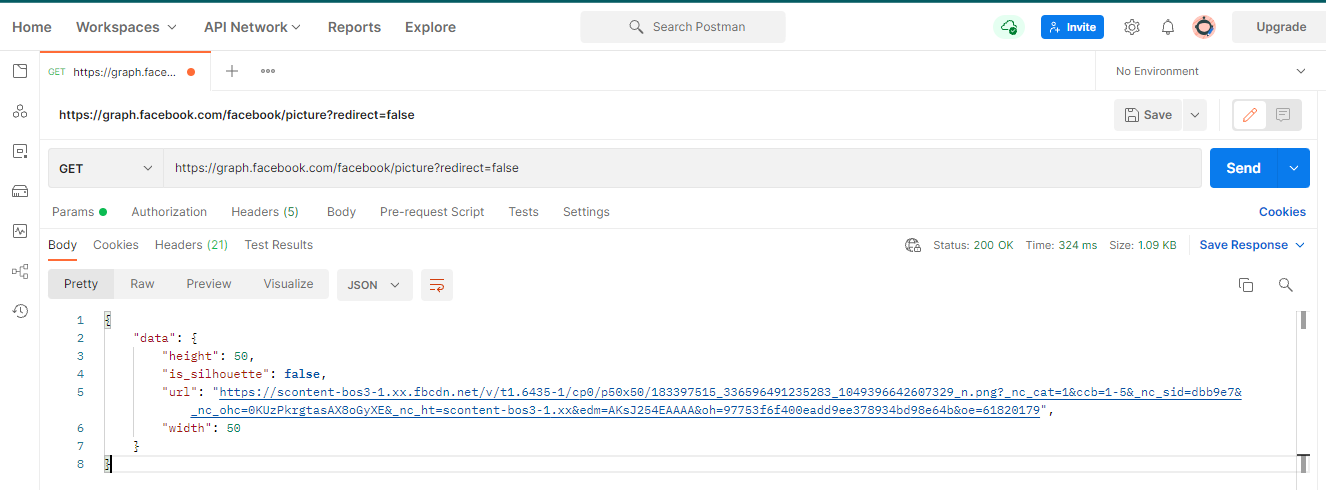
* RESTFUL (Representational State Transfer) là một chuẩn (phong cách) dùng trong việc thiết kế các API cho các ứng dụng web
* RESTFUL API được xây dựng trên giao thức HTTP, sử dụng các động từ HTTP tiêu chuẩn để làm việc với tài nguyên (resource) như GET, POST, PUT, PATCH, DELETE.
* Chức năng quan trọng nhất của **REST** là quy định cách sử dụng các HTTP method (như GET, POST, PUT, DELETE…) và cách định dạng các URL cho ứng dụng web để quản các resource. **RESTful** không quy định logic code ứng dụng và không giới hạn bởi ngôn ngữ lập trình ứng dụng, bất kỳ ngôn ngữ hoặc framework nào cũng có thể sử dụng để thiết kế một **RESTful API.**

1. ***Các thành phần của 1 request HTTP***

* ***VD:***



* ***Ví dụ Cụ thể:***

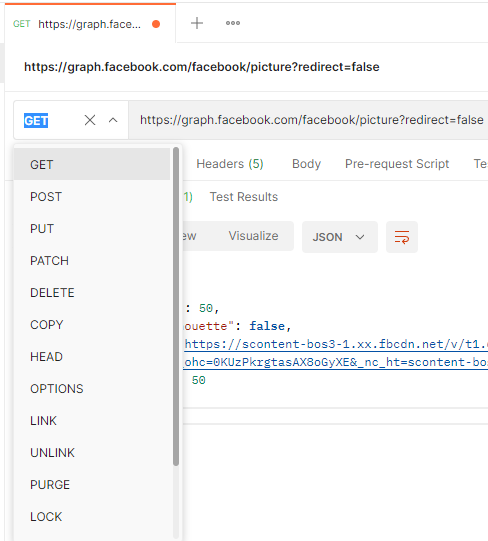


* endpoint (url ):

+ Là đường url mà client gửi yêu cầu lên

+ VD: https://graph.facebook.com/facebook/picture?redirect=false

* phương thức (method )



* GET: được sử dụng để lấy thông tin từ sever theo URI đã cung cấp.
* POST: gửi thông tin tới sever thông qua các biểu mẫu http (đăng kí chả hạn..)
* HEAD: giống với GET nhưng response trả về không có body, chỉ có header
* PUT: ghi đè tất cả thông tin của đối tượng với những gì được gửi lên
* PATCH: ghi đè các thông tin được thay đổi của đối tượng.
* DELETE: xóa tài nguyên trên server.
* CONNECT: thiết lập một kết nối tới server theo URI.
* OPTIONS: mô tả các tùy chọn giao tiếp cho resource.
* TRACE: thực hiện một bài test loop – back theo đường dẫn đến resource.
* Header

+ Là tiêu đề của yêu cầu cung cấp thông tin cho client và server

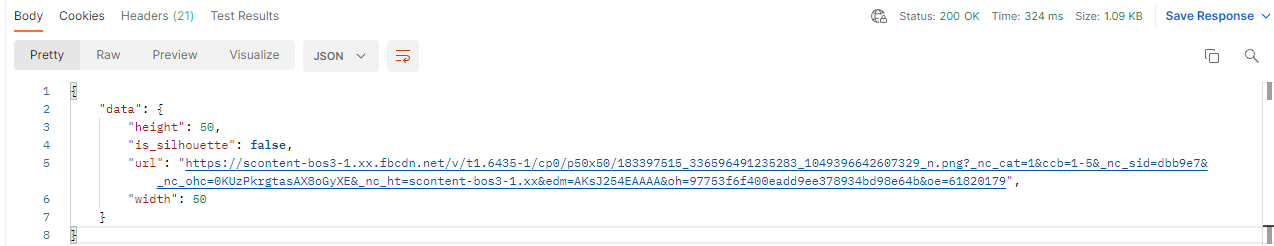
+ VD: Content-Type: application/json; charset= UTF-8



* body (data):

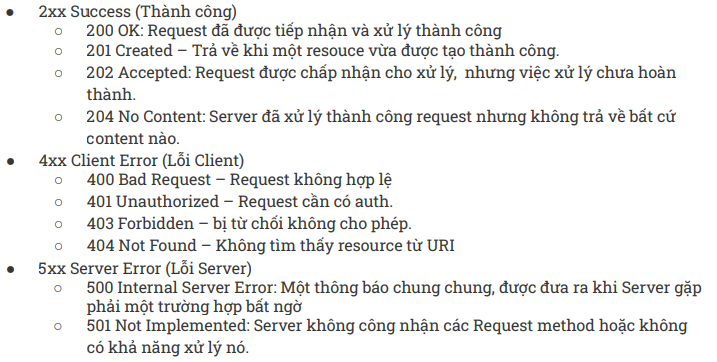
+ chứa tất cả thông tin mà client muốn gửi cho server

+ VD:



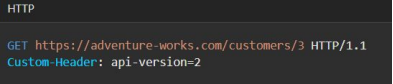
1. ***Tìm hiểu thêm***
   1. HTTP Status code

* HTTP status code là yếu tố quan trọng trong HTTP message response được server trả về cho client



* 1. Json
* JSON(JavaScript Object Notation) là kiểu định dạng dữ liệu chung.
* Clients tương tác với một service bằng cách trao đổi các tài nguyên dưới dạng JSON là phổ biến nhất.
* JSON sử dụng các cặp key – value
* VD: 
  1. Phân Trang
* Tránh dư thừa dữ liệu trả về, làm giảm việc lãng phí băng thông và hiệu năng xử lý.
* Nên thiết kế API giới hạn lượng data trả về là cần thiết cho mỗi request.
* VD: <https://api.instantwebtools.net/v1/passenger?page=1&size=10>
  1. Versioning
* Thuật tiện cho việc scale sản phẩm, thay đổi API theo thời gian
* Có 2 cách:

+ URL versioning: <http://shoseshop.com/api/v1/customers?page=1&limit=10>

+ Header versioning: 

* 1. Tips (Một số nguyên tắc thiết kế chính của RESTful API sử dụng HTTP)
* Một resource có một URL để định danh và nó là duy nhất.
* Tránh thiết kế URLs phức tạp hơn mẫu collection/item/collection
* Sử dụng danh từ thay vì động từ để biểu diễn tài nguyên (resource) trên URL
* Dùng danh từ số nhiều để biểu diễn 1 collection
* Dùng dấu gạch ngang (-) để ngăn cách giữa các cụm từ; Gạch dưới (\_) bị phụ thuộc nhiều vào font chữ hiển thị, trong một số trường hợp nó có thể bị che mất một phần, hoặc bị xóa, hoặc bị chuyển thành dấu cách trên một số trình duyệt. Điều này dễ dàng gây ra nhầm lẫn cho người sử dụng.
* Sử dụng chữ thường cho toàn bộ endpoint