Bài 1. Tổng quan về tích hợp hệ thống phần mềm

- Mục đích, yêu cầu: Cung cấp cho sinh viên kiến thức Tổng quan về tích hợp hệ thống phần mềm và các thành phần trong tích hợp hệ thống phần mềm.

- Hình thức tổ chức dạy học: Lý thuyết, trực tuyến + tự học

- Thời gian: Lý thuyết (trên lớp: 0; online: 1) Tự học, tự nghiên cứu: 2

- Nội dung chính:

Tổng quan về tích hợp hệ thống phần mềm

1. Tích hợp hệ thống là gì?

Trong thế giới kết nối đa chiều, công nghệ đang dẫn đầu cho mọi nền tảng phát triển khác. "*Tích hợp hệ thống*" đã trở nên khá quen thuộc trong mọi tổ chức, không phân biệt quy mô. Vậy tích hợp hệ thống là gì?

1.1. Tích hợp hệ thống

Tích hợp hệ thống tiếng Anh là System Integration – SI. Trong kỹ thuật, nó được hiểu đơn giản là kết nối một chuỗi các hệ thống con với những tính năng khác nhau vào một hệ thống lớn. Những kết nối này đảm bảo các hệ thống con được gắn kết chặt chẽ với nhau như một thể thống nhất. Mỗi hệ thống được vận hành theo mục đích riêng của từng doanh nghiệp. Tích hợp hệ thống là giải pháp đáp ứng mọi yêu

cầu phức tạp nhất của doanh nghiệp. Đặc trong trong các vấn đề về công nghệ với yêu cầu tùy biến cao.

Trong công nghệ thông tin, SI giúp tích hợp các hệ thống con rời rạc, các phần mềm ứng dụng lại với nhau. Điều này được thực hiện bằng cách sử dụng các kỹ thuật kết nối. Ví dụ như mạng máy tính, tích hợp ứng dụng, quản lý quy trình, lập trình… Tích hợp hệ thống là quy trình giúp gia tăng giá trị và năng lực của hệ thống mẹ nhờ hợp lực tương tác giữa các hệ thống con.

1.2. Những lợi ích của việc tích hợp hệ thống

Tích hợp hệ thống giúp doanh nghiệp tối ưu chi phí. Sở dĩ như vậy là nhờ khả năng tích hợp linh hoạt khi được lựa chọn công nghệ, thiết bị, dịch vụ phù hợp. Hơn nữa, tích hợp hệ thống còn giúp tối ưu hóa nhu cầu sử dụng. Nó giúp doanh nghiệp hoạch định và đầu tư theo từng giai đoạn cụ thể. Tất nhiên điều này còn tùy vào khả năng và mức nhu cầu của họ trong giai đoạn đó. Đồng thời nó ngăn chặn các rủi ro từ những môi trường kinh doanh độc hại. Tích hợp hệ thống còn góp phần làm tăng sức cạnh tranh của doanh nghiệp trên thị trường.

Cốt lõi để tích hợp hệ thống thành công nằm ở năng lực của nhà tích hợp hệ thống. Năng lực triển khai của kỹ sư tích hợp là nghệ thuật kết nối hệ thống rời rạc này thành khối sức mạnh hợp nhất. Điều này càng đúng khi các yếu tố phần mềm, phần cứng là như nhau.

1.3. Giải pháp kết nối tích hợp hệ thống

1.3.1. Các giải pháp xây dựng hạ tầng IT bao gồm:

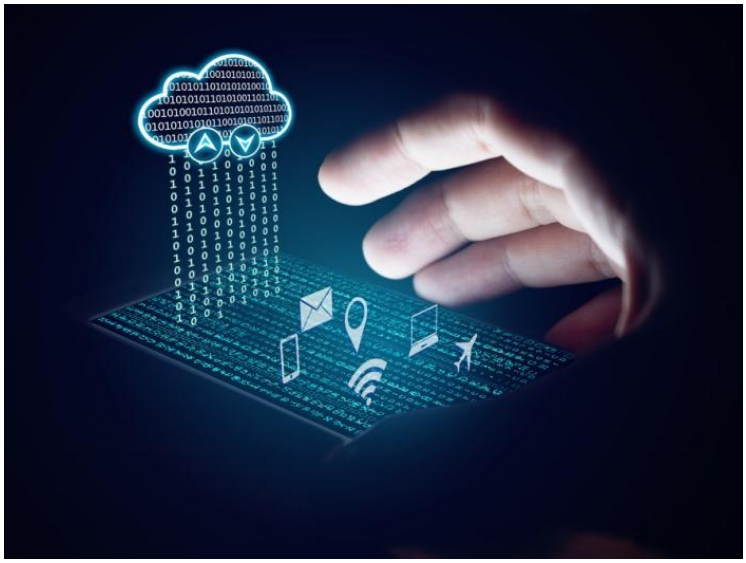
- Mạng LAN/WAN cho các doanh nghiệp.
- Giải pháp kết nối cho Trung tâm dữ liệu (Data Center)

1.3.2. Giải pháp cho các doanh nghiệp

Là gói giải pháp cung cấp khả năng kết nối cho các doanh nghiệp từ SME đến lớn. Các giải pháp loại này bao gồm cả phần hệ thống mạng (LAN/WAN), hệ thống tính toán, lưu trữ. Đồng thời chúng còn là các phần mềm hệ thống, các phương án bảo mật theo đúng yêu cầu của khách hàng.

1.3.3. Giải pháp lưu trữ máy chủ

Thế kỷ XXI được coi là thế kỷ bùng nổ thông tin. Cùng với đó là sự phát triển khái niệm "cloud computing" (điện toán đám mây). Giải pháp này mang lại cho con người nhiều tiện ích hơn nữa. Những lợi ích này thuộc về hệ thống mạng, hệ thống máy chủ và lưu trữ. Khi đó không cần phải phải đầu tư một máy tính đủ mạnh, có dung lượng lưu trữ lớn. Với chiếc máy tính hiện có, chúng ta có thể sử dụng những phần mềm mới nhất, chứa được nhiều dữ liệu hơn. Điện toán đám mây mang lại những tiện ích tuyệt vời. Chúng đến từ những hệ thống máy chủ của nhà cung cấp. Dữ liệu sẽ được chứa tại nhà cung cấp thay vì trên chiếc máy tính của mình. Khi đó, chỉ cần một máy tính có kết nối Internet, bạn hoàn toàn có thể truy xuất dữ liệu của mình mọi lúc mọi nơi.



Điện toán đám mây

1.3.4. Giải pháp kết nối cho Data Center (trung tâm dữ liệu)

Ngày nay nhu cầu tập hợp dữ liệu ngày càng cao. Giải pháp này giúp các doanh nghiệp xây dựng cơ sở hạ tầng cho các trung tâm dữ liệu an toàn. Chúng dễ dàng triển khai, mở rộng, quản lý cũng như tiêu hao năng lượng tối thiểu. Các trung tâm dự liệu theo tiêu chuẩn quốc tế đảm bảo an toàn và tối ưu hóa năng lượng và chi

phí. Không những vậy, chúng còn giúp khách hàng tập trung hóa cơ sở dữ liệu, quản lý từ xa dễ dàng. Hơn nữa đó còn làm nâng cao hiệu quả sản xuất kinh doanh.

Trung tâm dữ liệu

2. **Mối liên kết giữa các hệ thống**

Ngày nay, việc cho phép thao tác chuyển đổi giữa các tài nguyên thông tin khác nhau về hình dạng và nội dung là một trong những vấn đề then chốt của các cơ quan, tổ chức. Giữa người sử dụng và các ứng dụng có những nhu cầu về truy cập và trao đổi dữ liệu từ số lượng lớn ngày càng tăng. Tuy nhiên, các tài nguyên thông tin này được tạo ra và quản trị hoàn toàn độc lập về mặt vật lý, nguyên tắc và phương thức. Vì vậy, phương pháp tiếp cận đặt ra 03 hướng giải quyết:

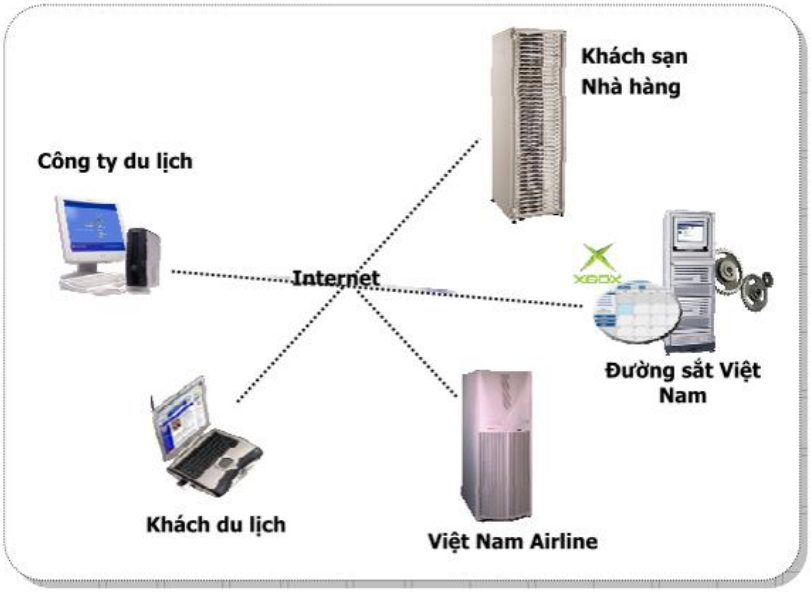
- Việc kết nối và chuyển đổi dữ liệu với nhiều hệ thống của nhà cung cấp khác nhau (về kỹ thuật giao thức và dạng nội dung dữ liệu);

- Hạn chế tối đa việc thay đổi giao thức, cơ sở dữ liệu và cách hoạt động của hệ thống sẵn có;

- Đảm bảo tính liên tục của các dịch vụ được cung cấp hiện nay.

Do vậy, cấu trúc trao đổi dữ liệu tuân theo lược đồ XML Schema, XML Schema được sử dụng để tạo ra dữ liệu, mô tả cấu trúc dữ liệu khi luân chuyển và kiểm tra cấu trúc dữ liệu, tính đúng đắn của các thành phần (về hình dạng và nội dung) của dữ liệu khi xử lý.

Việc quản lý thông tin, bao gồm cả thông tin trên các ngôn ngữ khác nhau cũng dựa trên nền tảng XML. Các cấu trúc thông tin dữ liệu trong các bảng và trường của cơ sở dữ liệu cần được mô tả lại bằng các cấu trúc XML thích hợp, các liên kết trực tiếp, gián tiếp hoặc tham chiếu của các dữ liệu bên trong đều có thể được mô tả qua các phần tử con, thuộc tính, hoặc các tham chiếu qua XLink. Vì thế, ngôn ngữ XML có thể mô tả rõ ràng và chính xác bất kỳ cơ sở dữ liệu SQL nào.



Mối liên kết giữa các hệ thống

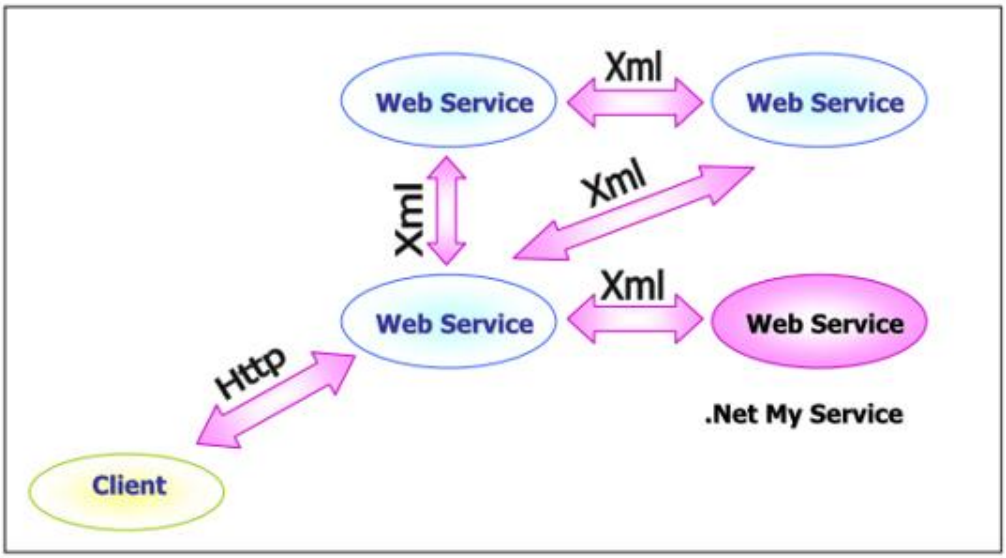
Dữ liệu trao đổi giữa các hệ thống

Trong thực tế XML/ JSON được sử dụng để đóng gói và trao đổi dữ liệu giữa các hệ thống. Khi có sự trao đổi dữ liệu giữa các hệ thống khác nhau thì dữ liệu đó được tổ chức dưới dạng XML/JSON.

Ví dụ: Hệ thống quản lý của **Nhà hàng** muốn lấy thông tin của khách du lịch từ hệ thống của **Công ty du lịch** thì giữa các hệ thống cần phải thực hiện các bước sau:

- Bước 1: Giữa các hệ thống phải thống nhất cấu trúc của tài liệu XML/JSON

- Bước 2: Công ty du lịch sẽ trích xuất dữ liệu từ hệ thống của mình, sau đó đóng gói dữ liệu dưới dạng XML/JSON theo cấu trúc đã thoả thuận ở bước 1.
- Bước 3: Hệ thống phần mềm của nhà hàng sẽ tiến hành phân tích và trích xuất dữ liệu từ tài liệu XML/JSON nhận được từ hệ thống của công ty du lịch.



Mối liên kết giữa các hệ thống

3. **Webservice**

Webservice là tập hợp các giao thức và tiêu chuẩn mở được sử dụng để trao đổi dữ liệu giữa các ứng dụng hoặc giữa các hệ thống. Tóm gọm

- Là phương thức giao tiếp giữa hai thiết bị qua mạng.
- Là ứng dụng hoặc thành phần ứng dụng để giao tiếp.
- Là tập hợp các tiêu chuẩn hoặc giao thức để trao đổi thông tin giữa hai thiết bị hoặc ứng dụng.

Các ứng dụng phần mềm được viết bằng các ngôn ngữ lập trình khác nhau và chạy trên các nền tảng khác nhau, có thể sử dụng các dịch vụ web để trao đổi dữ liệu qua mạng máy tính.

Web service hoạt động một cách độc lập không phụ thuộc bất kỳ ngôn ngữ nào. Các ứng dụng java, .net hoặc PHP… có thể giao tiếp với các ứng dụng khác thông qua web service.

3.1. Các thành phần của web service

Nền tảng web service cơ bản là XML HTTP. Tất cả các web service chuẩn đều hoạt động bằng các thành phần sau:

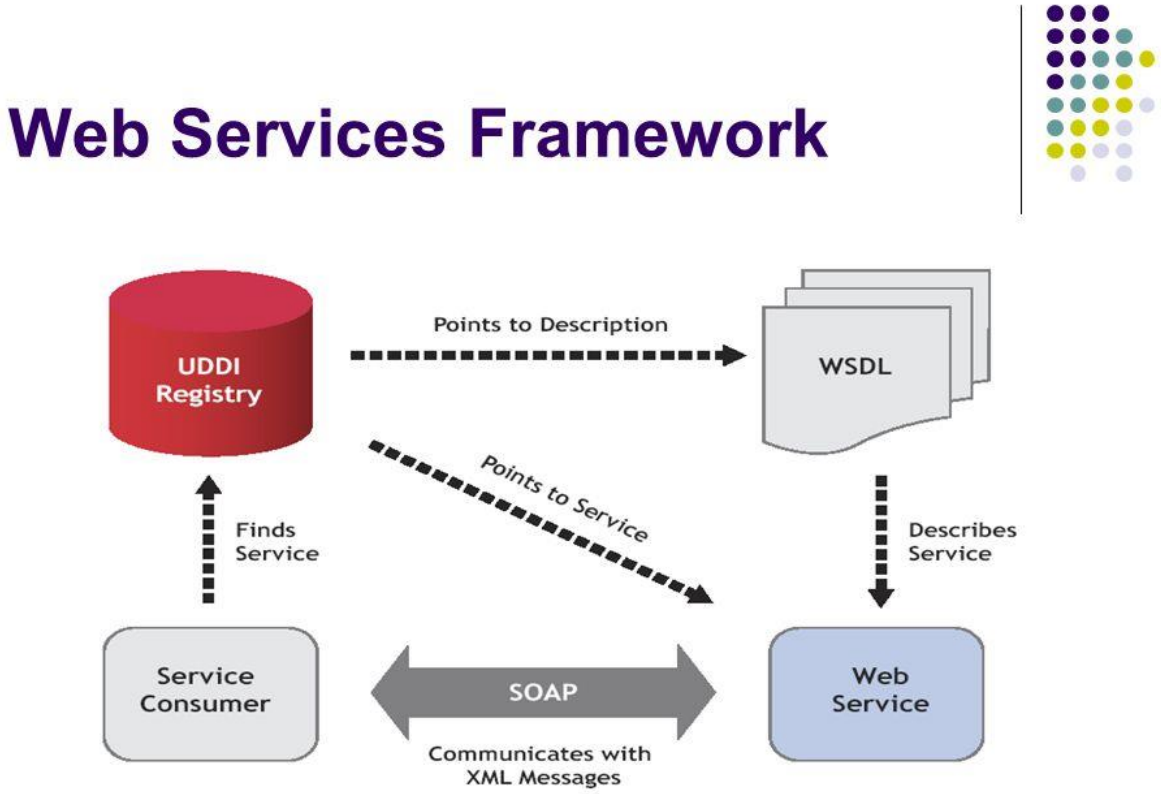
– **SOAP** (là viết tắt của Simple Object Access Protocol) – giao thức truy cập đối tượng đơn giản: SOAP là một giao thức dựa trên XML đơn giản cho phép các ứng dụng trao đổi thông tin qua HTTP.

– **UDDI** (Universal Description, Discovery and Integration): UDDI là một tiêu chuẩn dựa trên XML để mô tả, xuất bản và tìm kiếm các dịch vụ web.

– **WSDL** (Web Services Description Language) – ngôn ngữ định nghĩa web service: WSDL là một ngôn ngữ dựa trên XML để mô tả các dịch vụ web và cách truy cập chúng.

3.2. Các lợi ích mang lại từ Web service

Ngoài việc cho phép các ứng dụng được viết bằng các ngôn ngữ lập trình khác nhau giao tiếp với nhau, các dịch vụ web còn mang lại những lợi thế khác. Đầu tiên, họ cung cấp quyền truy cập vào các tính năng thông qua internet. Thật vậy, các tính năng được cung cấp bởi dịch vụ web cho ứng dụng khách được gọi thông qua giao thức HTTP. Do đó, chúng có thể được gọi qua internet. Tại thời điểm tất cả các ứng dụng được kết nối với internet, các dịch vụ web đã trở nên hữu ích hơn nhiều so với trước đây.



Web Services Framework

Ngoài ra, các dịch vụ web cho phép khả năng tương tác giữa các ứng dụng. Chúng cho phép các ứng dụng khác nhau giao tiếp với nhau và chia sẻ dữ liệu và dịch vụ. Do đó, thay vì phải viết mã cụ thể chỉ có thể được hiểu bởi các ứng dụng cụ thể, có thể viết mã chung có thể được hiểu bởi tất cả các ứng dụng.

Một ưu điểm khác của dịch vụ web là chúng sử dụng giao thức công nghiệp được tiêu chuẩn hóa để liên lạc. Bốn lớp (Service Transport, XML Messaging, Service Description và Service Discovery) sử dụng các giao thức được xác định rõ.

Cuối cùng, dịch vụ web có thể giảm chi phí liên lạc. Sử dụng SOAP thông qua giao thức HTTP, có thể sử dụng kết nối internet chi phí thấp để triển khai các dịch vụ web, các web service sử dụng an toàn và nhanh chóng.

4. RESTful Service

Sự quan trọng của API trong các ứng dụng ngày nay là điều ko thể bàn cãi. Một ứng dụng mà không có API thì như một cỗ máy tính không kết nối internet vậy. Và như một điều hiển nhiên, mọi thứ sau khi phát triển một thời gian sẽ hình thành những chuẩn mực chung và đối với API, nó chính là RESTful

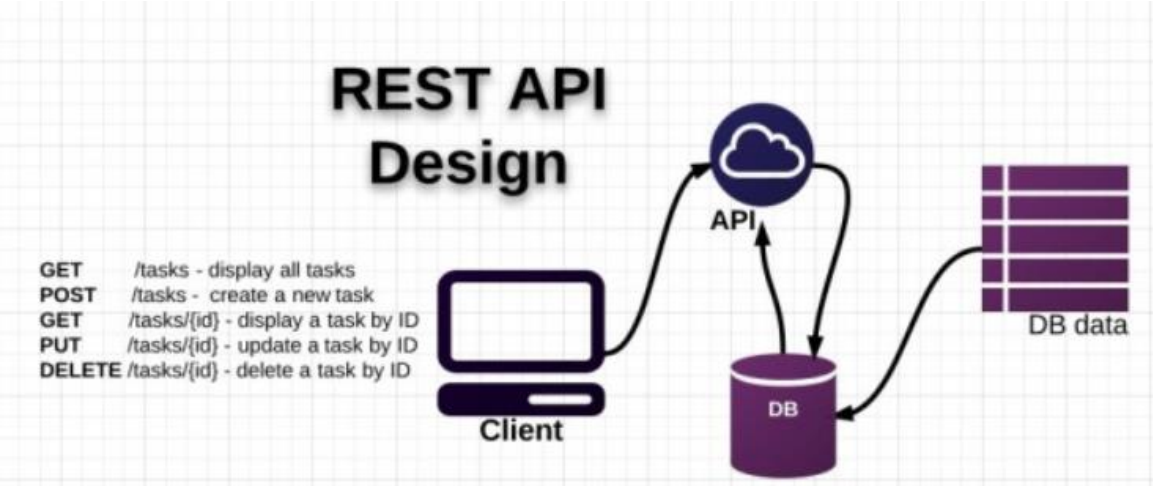
Có thể nói bản thân REST không phải là một loại công nghệ. Nó là phương thức tạo API với nguyên lý tổ chức nhất định. Những nguyên lý này nhằm hướng dẫn lập trình viên tạo môi trường xử lý API request được toàn diện.

Để hiểu rõ hơn về RESTful API ta sẽ đi lần lượt giải thích các khái niệm nhở API, REST hay RESTful.

4.1. RESTful API là gì?

Các lập trình viên web thường nhắc đến nguyên lý REST và cấu trúc dữ liệu RESTFUL bởi nó là một phần rất quan trọng trong sự phát triển của các ứng dụng web. Vậy RESTFUL API là gì ? Để hiểu rõ hơn chúng ta cùng nhau tìm hiểu nhé.

RESTful API là một tiêu chuẩn dùng trong việc thiết kế API cho các ứng dụng web (thiết kế Web services) để tiện cho việc quản lý các resource. Nó chú trọng vào tài nguyên hệ thống (tệp văn bản, ảnh, âm thanh, video, hoặc dữ liệu động…), bao gồm các trạng thái tài nguyên được định dạng và được truyền tải qua HTTP.



Các thành phần của RESTful

API (**A**pplication **P**rogramming **I**nterface) là một tập các quy tắc và cơ chế mà theo đó, một ứng dụng hay một thành phần sẽ tương tác với một ứng dụng hay thành phần

khác. 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.

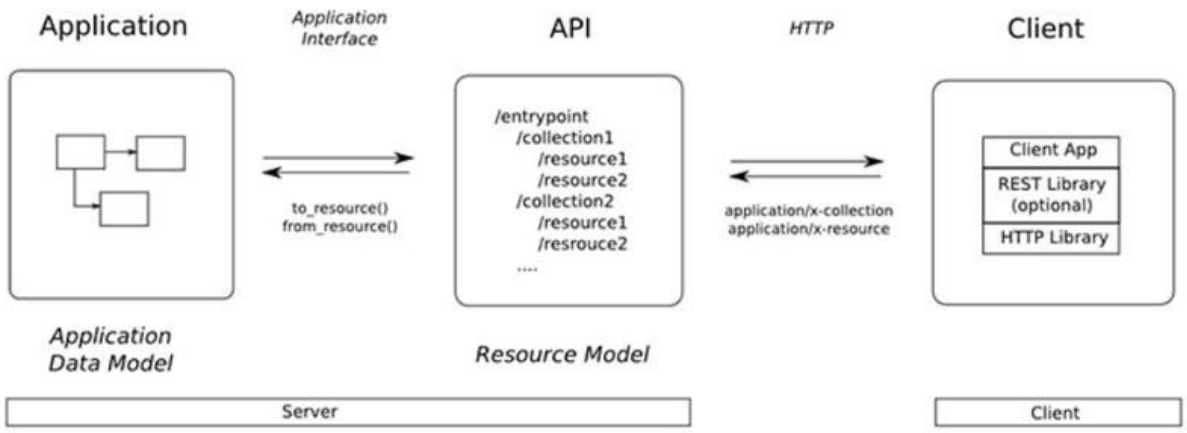
REST (**RE**presentational **S**tate** T**ransfer) là một dạng chuyển đổi cấu trúc dữ liệu, một kiểu kiến trúc để viết API. Nó sử dụng phương thức HTTP đơn giản để tạo cho giao tiếp giữa các máy. Vì vậy, thay vì sử dụng một URL cho việc xử lý một số thông tin người dùng, REST gửi một yêu cầu HTTP như GET, POST, DELETE, vv đến một URL để xử lý dữ liệu.

RESTful API là một tiêu chuẩn dùng trong việc thiết kế các API cho các ứng dụng web để quản lý các resource. RESTful là một trong những kiểu thiết kế API được sử dụng phổ biến ngày nay để cho các ứng dụng (web, mobile…) khác nhau giao tiếp với nhau.

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.**

4.2. RESTful API hoạt động như thế nào?

Sau khi chúng ta biết được RESTful API là gì thì trong phần này chúng ta cùng tìm hiểu nguyên lý hoạt động của nó nhé. Giống như các giao thức truyền thông hay cấu trúc dữ liệu khác. Để hiểu được bản chất vấn đề thì trước hết cần phải hiểu nguyên lý hoạt động của nó.



REST hoạt động chủ yếu dựa vào giao thức HTTP. Các hoạt động cơ bản nêu trên sẽ sử dụng những phương thức HTTP riêng.

- GET (SELECT): Trả về một Resource hoặc một danh sách Resource.
- POST (CREATE): Tạo mới một Resource.
- PUT (UPDATE): Cập nhật thông tin cho Resource.
- DELETE (DELETE): Xoá một Resource.

Những phương thức hay hoạt động này thường được gọi là CRUD tương ứng với Create, Read, Update, Delete – Tạo, Đọc, Sửa, Xóa.

Hiện tại đa số lập trình viên viết RESTful API giờ đây đều chọn JSON là format chính thức nhưng cũng có nhiều người chọn XML làm format, nói chung dùng thế nào cũng được miễn tiện và nhanh.

4.3. Authentication request và cấu trúc dữ liệu trả về

RESTful API không sử dụng session và cookie, nó sử dụng một access_token với mỗi request. Dữ liệu trả về thường có cấu trúc như sau:

```
{
   "status_code": 200,
   "data": [
      {
         "name": "ManhLD",
         "email": "manhld@example.com",
         "ny": "not found"
      },
      {
         "name": "Ahri",
```

```
            "email": "ahriKDA@lmht.com",
            "ny": "Ezreal"
        }
    ],
    error_messages: ""
}
```

Ở trên là ví dụ về cấu trúc trả về của api get một list users trong hệ thống.

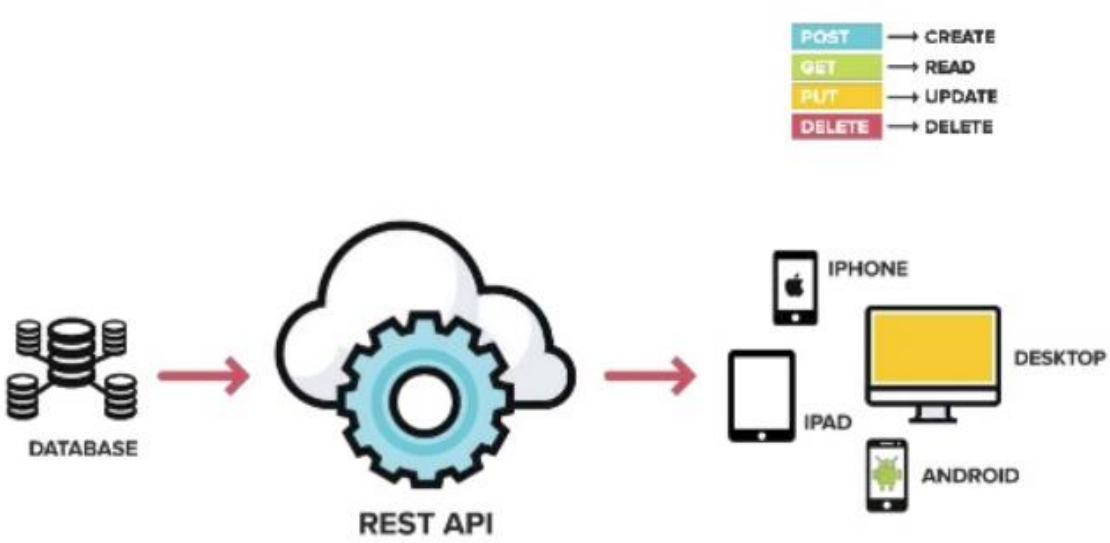
4.4. Status code

Khi chúng ta request một API nào đó thường thì sẽ có vài status code để nhận biết sau:

- 200 OK – Trả về thành công cho những phương thức GET, PUT, PATCH hoặc DELETE.
- 201 Created – Trả về khi một Resouce vừa được tạo thành công.
- 204 No Content – Trả về khi Resource xoá thành công.
- 304 Not Modified – Client có thể sử dụng dữ liệu cache.
- 400 Bad Request – Request không hợp lệ
- 401 Unauthorized – Request cần có auth.
- 403 Forbidden – bị từ chối không cho phép.
- 404 Not Found – Không tìm thấy resource từ URI
- 405 Method Not Allowed – Phương thức không cho phép với user hiện tại.
- 410 Gone – Resource không còn tồn tại, Version cũ đã không còn hỗ trợ.
- 415 Unsupported Media Type – Không hỗ trợ kiểu Resource này.
- 422 Unprocessable Entity – Dữ liệu không được xác thực
- 429 Too Many Requests – Request bị từ chối do bị giới hạn

4.5. Ưu điểm của RESTFUL API là gì ?

Như trình bày ở trên, việc sử dụng RESTFUL API mang lại những hiệu quả nhất định cho các lập trình viên. Vậy những lợi ích nó mang lại là gì ? So với các phương pháp khác nó sẽ có điểm gì vượt trội



Một số ưu điểm chính khi sử dụng RESTFUL API là:

- Giúp cho ứng dụng rõ ràng hơn
- REST URL đại diện cho resource chứ không phải hành động
- Dữ liệu được trả về với nhiều định dạng khác nhau như: xml, html, json….
- Code đơn giản và ngắn gọn
- REST chú trọng vào tài nguyên của hệ thống

Những trang web ngày nay thường sử dụng REST API để cho phép kết nối đến dữ liệu của họ. Trong đó, facebook cũng cung cấp các REST API để giúp các ứng dụng bên ngoài kết nối đến dữ liệu của họ

Tài liệu tham khảo:

- Sách, giáo trình chính:

[1]. Giáo trình dịch vụ web và ứng dụng/ Đại học KHTN – Đại học Quốc gia TP HCM, 2016.

[2]. Anura Guruge. Web service: Theory and Practice. Elsevier press 20017.
[3]. Scott Klein. Professional API- RESTfull Programming. Wiley Publishing, 2016.

- Sách, tài liệu tham khảo:

[1]. Alex Ferra, Mathew MacDonald. Programming .NET web services. O'Reilly Media, 2012.
[2]. Mark D. Hansen. SOA Using .NET web services. Prentice Hall, 2011.