Môn: Thương mại điện tử

SV:1312177\_Trần Thị Hạnh.

TÌM HIỂU CÔNG NGHỆ

I. Tech 1: Tìm hiểu ngôn ngữ lập trình: front end, back end và API.  
1. **Front end:** bao gồm quá trình xử lý các dữ liệu request nhận từ client và reponse gửi cho client. Cụ thể là các request của user sẽ được biên dịch thành gói dữ liệu như thế nào trên client trước khi gửi đến server, và các gói dữ liệu response nhận được từ server sẽ được brower trình diễn ra trang web trên máy client như thế nào. Để tạo ra được front end chúng ta cần sự kết hợp của 3 ngôn ngữ HTML, CSS và JavaScript đây là những ngôn ngữ mà brower dùng để biên dịch các dòng dữ liệu respose thô thành một trang web sống động nhiều màu sắc( nó là tất cả những gì người dùng nhìn thấy khi điều hướng trên internet từ front chữ, màu sắc cho tới menu xổ xuống và các thanh trượt.).

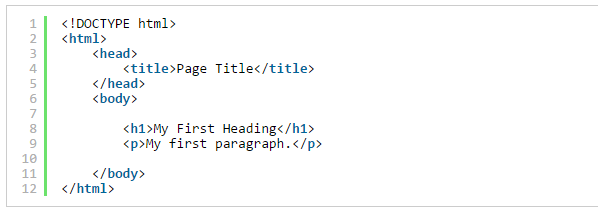
***- HTML:*** là chữ viết tắt của HyperText MarText Markup Language có nghĩa là ngôn ngữ đánh dấu siêu văn bản. Nó dùng để định dạng bố cục, các thuộc tính liên quan đến cách hiển thị của một đoạn text và được hiển thị trên một chương trình đặc biệt ta gọi là Brower. Hiện nay có khá nhiều Brower như Firefox, Chrome, Coccoc…. Tất cả Brower đề có điểm chung là giúp người dùng thao tác với website và nó đều có khả năng biện dịch những đoạn mã HTML, CSS và JavaScript. **HTML** là một định dạng đặc biệt của XML nên nó sẽ có thẻ mở và thẻ đóng với cú pháp:

<tagname></tagname>

Ngoài ra mỗi thẻ HTML còn có một số thuộc tính riêng và danh sách các thuộc tính sẽ nằm bên trong thẻ mở như sau:

<tagname pro1=”value1” pro2=”value2></tagname>

Bố cục HTML của một trang web:



Trong đó:

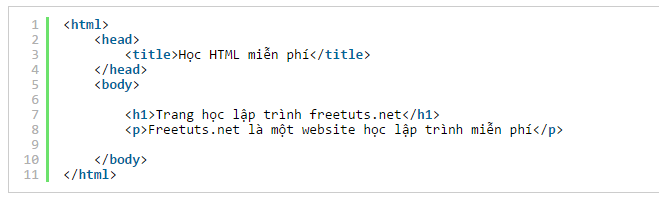
* <!DOCTYPE html> là phần khai báo kiểu dữ liệu hiển thị là html để trình duyệt biết.
* <html> và </html> là cặp thẻ nằm ngoài cùng và nó có nhiệm vụ là bao quát hết nội dung của trang web lại. Thẻ này là bắt buộc.
* <head> và </head> là phần khai báo thông tin của trang web.
* <body> và </body> là thành phần quan trọng nhất, nó chứa những đoạn mã html dùng đẻ hiển thị trên website.
* Các thẻ còn lại nằm trong thẻ ,body> chính là các thẻ định dạng dữ liệu.

Như vậy trong một website chúng chia làm 2 phần chính:

Phần 1: Là những khai báo thông tin cho trang web và đặt nó trong thẻ *head*.

Phần 2: Là phần hiển thị đinh dạng nội dung của trang web và ta đặt trong thẻ *body.*

Ví dụ:



Các vị trí thường thấy trong giao diện website thông thường sẽ được chia thành các phần sau:

Header, Footer, Menu, Sidebar, Main Content( mỗi trang khác nhau).



**CSS** là chữ viết tắt của Cascading Stype Sheet nghĩa của nó tạm dịch CSS dùng để trình bày bố cục của một website dựa váo các thẻ html. Nói nôm na CSS sẽ tác động thay đổi cách hiển thị mặc định của các thẻ html nhằm thân thiên với người dùng hơn hay nhằm một mục đích cụ thể nào đó. Người ta thường hay ví: giả sử bạn xây dựng ngôi nhà. Nhưng đẻ trang trí cho ngôi nhà thì bạn sẽ sử dụng một số đồ nội thất khác như sơn màu….thì lúc này sơn màu sẽ tác động là thay đổi color mặc định của gạch. Màu sơn là CSS và gạch chính là html.

CSS đóng một vai trò rất quan trọng khi bạn xây dựng giao diện Website. Thông thường bạn sẽ  sử dụng một chương trình thiết kế như Photoshop để tạo giao diện, sau đó các Frontend Developer sẽ dựa vào file đó kết hợp với HTML & CSS để convert thanh một file có đuôi mở rộng là .html, file File này sẽ chứa code **HTML và CSS**.

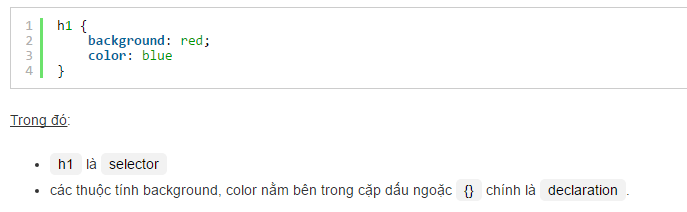
CSS có nhiều kiểu nhưng gom lại thì chúng ta sẽ có các thể loại CSS thông dụng như sau:

* **Background**: CSS tùy chỉnh hình nền
* **Text**: CSS tùy chỉnh cách hiển thị đoạn text
* **Font**: CSS tùy chỉnh kích thước, kiểu chữ
* **Link**: CSS tùy chỉnh link
* **List**: CSS tùy chỉnh danh sách
* **Table**: CSS tùy chỉnh bảng
* **Box model**: Mô hình box model kết hợp padding, margin, border.
* ...

Cú pháp của CSS Chúng ta chia một đoạn mã CSS ra gồm hai phần đó là:

* **selector**: selector sẽ trỏ đến những đối tượng (*html*) chịu ảnh hưởng bởi CSS.
* **declaration**: các thuộc tính CSS dùng để style cho thẻ selector

Ví dụ về thiết lập màu nền và chữ:



Chúng ta có 3 cách viết CSS:

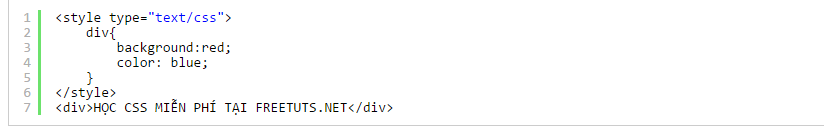
1. inline: viết trực tiếp lên thẻ thông qua thuộc tính *style.*
2. External: viết riêng một thẻ có phần đuôi .css rồi sau đó import vào bằng thẻ *link*.
3. Internal: viết tại file html hiện tại và nằm trong thẻ *style.*

# inline: Trong thẻ HTML chúng ta tạo một thuộc tính style = “code css”.

Ví dụ:



# internal : chúng ta sẽ code bên trong thể <style type=”text/css’> code css </type>



#External: trước hết chúng ta cần tạo một file có đuôi .css sau đó import vào file html qua thể link.





**JavaScript:**

**- Khái niệm:** JavaScript là một ngôn ngữ lập trình đa nền tảng (**cross-platform)**, ngôn ngữ lập trình kịch bản, hướng đối tượng. JavaScript là một ngôn ngữ nhỏ và nhẹ (small and lightweight). Khi nằm bên trong một môi trường (host environment), JavaScript có thể kết nối tới các Object của môi trường đó và cung cấp các cách quản lý chúng (Object).

JavaScript chứa các thư viện tiêu chuẩn cho các Object, ví dụ như:  Array, Date, và Math, và các yếu tố cốt lõi của ngôn ngữ lập trình như: toán tử (operators), cấu trúc điều khiển (control structures), và câu lệnh. JavaScript có thể được mở rộng cho nhiều mục đích bằng việc bổ sung thêm các Object; ví dụ:

* *Client-side JavaScript* - JavaScript phía máy khách, JavaScript được mở rộng bằng cách cung cấp các Object để quản lý trình duyệt và Document Object Model (DOM)  của nó. Ví dụ, phần mở rộng phía máy khách cho phép một ứng dụng tác động tới các yếu tố trên một trang HTML và phản hồi giống các tác động của người dùng như click chuột, nhập form, và chuyển trang.
* *Server-side JavaScript*- JavaScript phía Server, JavaScript được mở rộng bằng cách cung cấp thêm các đối tượng cần thiết để để chạy JavaScript trên máy chủ. Ví dụ, phần mở rộng phía server này cho phép ứng dụng kết nối với cơ sở dữ liệu (database), cung cấp thông tin một cách liên tục từ một yêu cầu tới phần khác của ứng dụng, hoặc thực hiện thao tác với các tập tin trên máy chủ.

Có 2 công cụ được xây dựng trong Firefox, nó rất hữu ích để chạy các 'thí nghiệm' với JavaScript, đó là: **Web Console** và **Scratchpad**.

### Web Console

### [Web Console](https://developer.mozilla.org/en-US/docs/Tools/Web_Console) cho phép bạn thấy thông tin về trang Web đang chạy, và kèm theo một [command line](https://developer.mozilla.org/en-US/docs/Tools/Web_Console#The_command_line_interpreter), với nó bạn có thể sử dụng để chạy một đoạn lệnh JavaScript trên trang Web hiện tại.

* Mở Web Console bằng cách chọn "Web Console" từ menu "Web Developer" (Ctrl + Shift + I),  "Web Developer" nằm trong Menu chính của Firefox, nó có hình cờ lê, tên: Developer (nếu vẫn không thấy nó, bạn có thể mở menu và nhấn Customize để kéo nó ra ngoài). Sau khi mở lên, nó sẽ là 1 bảng hiển thị nằm phía dưới của cửa sổ trình duyệt. Có 1 ô nhập chạy dài dưới cùng của cửa sổ Web console (khoanh vùng màu đỏ trong mình), nó chính là command line, với nó bạn có thể nhập vào đoạn JavaScript mà bạu muốn thực thi, và sau khi Enter thì trình duyệt sẽ chạy và trả về kết quả lên bảng Web Console nằm trên nó:

### Scratchpad

* Web Console có thể rất hiệu quả cho việc chạy đơn lẻ từng dòng lệnh của JavaScript, bạn cũng có thể chạy những đoạn lệnh nhiều dòng với nó (Ctrl + Enter)... Nhưng có vẻ nó không được tiện lợi cho lắm! Bạn không thể lưu lại code khi sử dụng Web Console. Với các 'thí nghiệm' dài và phức tạp thì [Scratchpad](https://developer.mozilla.org/en-US/docs/Tools/Scratchpad) là một công cụ hiệu quả.
* Để mở Scratchpad, chọn "Scratchpad" từ menu "Web Developer" (Ctrl + Shift + I),  "Web Developer" nằm trong Menu chính của Firefox, nó có hình cờ lê, tên: Developer (nếu vẫn không thấy nó, bạn có thể mở menu và nhấn Customize để kéo nó ra ngoài).  Nó sẽ mở lên trong một cửa sổ window riêng với trình duyệt và là một trình soạn thảo mà bạn có thể sử dụng để viết và chạy JavaScript trong trình duyệt. Bạn cũng có thế lưu lại hoặc mở lên các đoạn script đó lên từ ổ đĩa.

Ví dụ về một chương trình HelloWorld trong JavaScript:



***\*Ưu điểm của Javascript:***  
- Hoàn toàn miễn phí và dễ học.

- JavaScript được thiết kế độc lập với hệ điều hành. Nó có thể chạy trên bất kỳ hệ điều hành nào có trình duyệt hỗ trợ JavaScript.

- Dễ dàng tương tác, điều khiển và tránh bớt việc xử lý từ phía server

- Nắm vững kiến thức JavaScript bây giờ rất hữu dụng cho các bạn sau này để có thê tiếp thu những công nghệ mới mà nó được gói gọn vào những ngôn ngữ như : Ajax , Atlas ….

***\*Nhược điểm của Javascript***  
- Javascript không có trình biên dịch riêng mà được diễn dịch và chạy bởi trình duyệt hỗ trợ nó. Chính vì thế, nếu trình duyệt không hỗ trợ, hoặc không bật JS, nó sẽ không chạy được.

- JS có thể làm ứng dụng web của bạn trở nên nặng nề hơn.

- Bảo mật kém. Không có khả năng giấu mã.

**\* Nguyên tắc lập trình trong JavaScript**  
Bất cứ ngôn ngữ lập trình nào cũng đều phải có nguyên tắc của nó, và Javascript cũng ko ngoại lệ. Để bắt đầu học lập trình 1 ngôn ngữ nào, trước tiên chúng ta cần phải biết quy tắc của nó.

Một số quy tắc quan trọng trong JS mà chúng ta cần phải nhớ như sau:

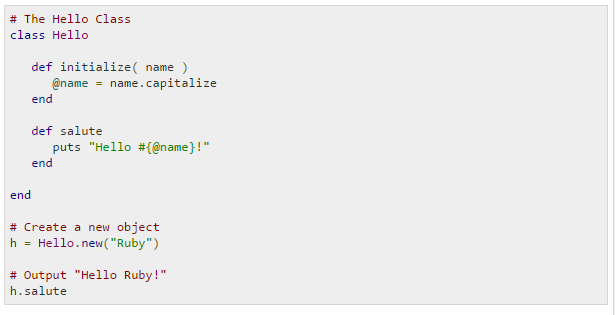
* Lệnh Javascript phân biệt chữ in hoa và chữ thường
* Mội câu lệnh Javascript đều kết thúc bằng dấu chấm phẩy “;”
* Các điều kiện phải được khai báo trong cặp dấu ngoặc đơn ().
* Khi sử dụng lệnh điều khiển, nếu sử dụng nhiều hơn 1 lệnh, bạn phải sử dụng cặp dấu ngoặc nhọn {}
* Javascript sử dụng dấu chấm “.” để tham chiếu đến 1 phương thức hay thuộc tính của đối tượng (Thay vì bên PHP là dấu “->”)

2. **back end**: bao gồm quá trình xử lý các dữ liệu request ở phía server và quá trình nhào nặn trên trình ứng dụng server( application) để tạo ra gói trả lời response gửi trả về cho client. Cụ thể là cần truy xuất thông tin phần nào từ database, xử lý thông tin đó như thế nào và sẽ ghi xuống database những gì, gửi trả về cho client những thông tin nào sau khi xử lý xong. Các ngôn ngữ lập trình cơ bản cho back end như C, Java, Python, Ruby, PHP, .NET, SQL….

**RUBY:** Hay còn hay gọi là Ruby on Rails. Rails là một web framework được thiết kế để làm việc bằng ngôn ngữ lập trình Ruby. Ruby on Rails là một cái tên ĐẦY ĐỦ cho framework Rails. Nhưng khi nói về nó thì các lập trình viên thường bỏ mất cụm từ "Ruby on" và chỉ cần gọi nó là Rails. Rails rất tốt cho người mới bắt đầu. Có một số lý do tại sao Rails là tuyệt vời cho người mới bắt đầu. Lý do đầu tiên chỉ đơn giản Ruby là thực sự tuyệt vời cho người mới bắt đầu. Học lập trình bằng Ruby dễ dàng hơn nhiều so với các ngôn ngữ lập trình khác bởi vì ngôn ngữ này cực kỳ linh hoạt, nó giúp bạn mất ít thời gian để nắm những nguyên tắc cơ bản trong lập trình và ít phải vò đầu bứt tóc khi gặp những khái niệm khó hiểu.

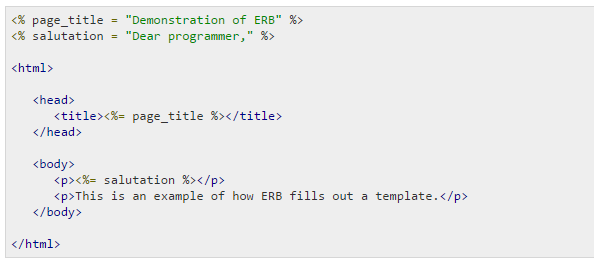
Ruby có nguồn gốc từ Nhật Bản và bây giờ nó đang trở nên phổ biến tại Mỹ và châu Âu. Các yếu tố sau đây góp phần vào việc phổ biến của nó :  
 - Dễ học  
 - Nguồn mở (giấy phép rất tự do)  
 - Thư viện phong phú  
 - Rất dễ dàng để mở rộng  
 - Quả thật hướng đối tượng  
 - Ít mã hóa với ít lỗi hơn.  
 - Cộng đồng hữu ích

* Mặc dù chúng ta có nhiều lý do để sử dụng Ruby, có một vài hạn chế như tốt mà bạn có thể phải xem xét trước khi thực hiện Ruby.  
  **Các vấn đề hiệu suất** - Mặc dù đối thủ Perl và Python, nó vẫn còn là một ngôn ngữ giải thích và chúng ta không thể so sánh nó với các ngôn ngữ lập trình cấp cao như C hay C ++.
* **Threading mô hình** - Ruby không sử dụng đề bản địa. Đề Ruby được mô phỏng trong các máy ảo chứ không phải chạy theo chủ đề hệ điều hành bản địa.
* Ví dụ về Ruby: kết quả in ra là *Hello Ruby!*



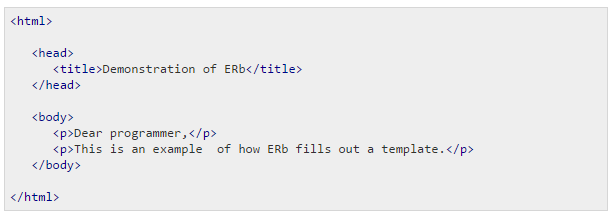
## Embedded của Ruby:

* Ruby cung cấp một chương trình gọi là ERB (Embedded Ruby), được viết bởi*Seki Masatoshi* . ERB cho phép bạn đặt mã của Ruby bên trong một tập tin HTML. ERB đọc cùng, từng chữ, và sau đó tại một thời điểm nhất định, khi nó gặp một mã Ruby nhúng trong tài liệu, nó bắt đầu thực thi mã Ruby.
* Bạn cần phải biết chỉ có hai điều cần chuẩn bị một tài liệu ERB -  
  Nếu bạn muốn có một số mã Ruby thực hiện, kèm theo nó giữa **<%** và**%>.**
  + Nếu bạn muốn kết quả của sự thực thi mã được in ra, như một phần của đầu ra, kèm theo mã giữa **<% =** và **%>** .
* Dưới đây là một ví dụ. Lưu mã trong tập tin erbdemo.rb. Lưu ý rằng một tập tin Ruby sẽ có một phần mở rộng **rb** –



Bây giờ, chạy chương trình bằng cách sử dụng dòng lệnh tiện ích **ERB** .



Điều này sẽ tạo ra các kết quả như sau -****

## **Rails là gì?**

## -Một suất rất cao khung ứng dụng web. - Viết bằng Ruby David Heinemeier Hansson. - Bạn có thể phát triển một ứng dụng web ít nhất mười lần nhanh hơn với - - - Rails hơn bạn có thể có một khuôn khổ Java điển hình. - Một của Ruby khuôn khổ nguồn mở để phát triển các ứng dụng web cơ sở dữ liệu sao lưu. - Cấu hình mã của bạn với lược đồ cơ sở dữ liệu. - Không có yêu cầu giai đoạn biên dịch.

## **Khung Full stack**

-Bao gồm tất cả mọi thứ cần thiết để tạo ra một ứng dụng web cơ sở dữ liệu điều khiển, sử dụng mô hình MVC.  
- Là một bộ khung hoàn chỉnh có nghĩa là tất cả các lớp được xây dựng để làm việc liền mạch với nhau với mã ít hơn.  
- Yêu cầu ít dòng mã hơn các khuôn khổ khác.

## **Ước về cấu hình**

* Rails tránh xa các tập tin cấu hình trong lợi của các công ước, suy tư, và phần mở rộng thời gian chạy năng động.
* mã ứng dụng của bạn và cơ sở dữ liệu của bạn đang chạy đã chứa tất cả mọi thứ mà Rails cần biết!

## **rails mạnh**

Rails được đóng gói với các tính năng mà làm cho bạn hơn, với rất nhiều các tính năng sau đây xây dựng vào những cái khác.

### Lập trình meta

Trường hợp các khuôn khổ khác sử dụng hệ mã mở rộng từ đầu, khung sắt sử dụng kỹ thuật Metaprogramming để viết chương trình. Ruby là một trong những ngôn ngữ tốt nhất cho Metaprogramming, và Rails sử dụng khả năng này tốt.Rails cũng sử dụng hệ mã nhưng dựa nhiều hơn vào Metaprogramming cho việc nâng hạng nặng.

### active Record

Rails giới thiệu Ghi khuôn khổ hoạt động, trong đó tiết kiệm đối tượng vào cơ sở dữ liệu. Các phiên bản Rails của Active Record phát hiện ra các cột trong một giản đồ cơ sở dữ liệu và tự động đính kèm chúng vào các đối tượng miền của bạn bằng lập trình meta.

### Công ước về cấu hình

Hầu hết các khuôn khổ phát triển web cho .NET hoặc Java buộc bạn phải viết các trang của mã cấu hình. Nếu bạn làm theo các quy ước đặt tên gợi ý, Rails không cần nhiều cấu hình.

### Đoạn đầu đài

Bạn thường tạo ra mã tạm thời trong giai đoạn đầu của sự phát triển để giúp có được một ứng dụng một cách nhanh chóng và xem như thế nào, những bộ phận làm việc cùng nhau. Rails tự động tạo ra nhiều của giàn giáo, bạn sẽ cần.

### Được xây dựng trong thử nghiệm

Rails tạo ra các bài kiểm tra tự động đơn giản thì bạn có thể mở rộng. Rails cũng cung cấp hỗ trợ mã được gọi là khai thác và đồ đạc mà làm cho trường hợp kiểm tra dễ dàng hơn để viết và chạy. sau đó Ruby có thể thực hiện tất cả các bài kiểm tra tự động của bạn với các tiện ích cào.

### ba môi trường

Rails cung cấp cho bạn ba môi trường mặc định: phát triển, thử nghiệm và sản xuất. Mỗi cư xử hơi khác nhau, làm cho toàn bộ chu trình phát triển phần mềm của bạn dễ dàng hơn. Ví dụ, Rails tạo ra một bản sao mới của cơ sở dữ liệu thử nghiệm cho mỗi lần chạy thử nghiệm.

3. **API** = Application Programming Interface là giao diện lập trình ứng dụng. Mỗi hệ điều hành, ứng dụng đề có những bộ API khác nhau. Nó cung cấp cho người lập trình các hàm tương tác với Database, lập trình thực hiện các thao tác với Hệ điều hành hay phàn mềm đó. Hầu hết các hàm trong API đều chứa trong các file DLL. Các hàm API cũng như các hàm bình thường mà chúng ta hay viết và viết trên từng ngôn ngữ khác nhau sẽ là các bộ API khác nhau.Ví dụ :

* Google API cho bộ .NET, PHP…

# II. Tech 2: Tìm hiểu hosting hệ thống: local, internet.

# **Hosting là gì ?**

Hosting là dịch vụ lưu trữ dữ và chia sẻ liệu trực tuyến, là không gian trên máy chủ có cài đặt các dịch vụ Internet như world wide web (www), truyền file (FTP), Mail… ,bạn có thể chứa nội dung trang web hay dữ liệu trên không gian đó.

Ví dụ: Bình thường bạn có 1 file trong máy tính, trong Localhost của bạn, giờ bạn muốn cho người khác xem thì bạn cần tải file đó lên mạng, nơi để lưu trữ file đó gọi là hosting.



## **2. Các loại hosting**

– Shared hosting: [Chia sẻ host](https://www.bkns.vn/hosting/hosting-cpanel.html)  
– Collocated hosting: [Thuê chỗ đặt máy chủ](https://www.bkns.vn/server/cho-dat-may-chu.html)  
– Dedicated Server: [Máy chủ dùng riêng](https://www.bkns.vn/server/may-chu-dung-rieng.html)  
– Virtual Private Server: VPS là [máy chủ riêng ảo](https://www.bkns.vn/server/may-chu-ao.html)

### ****3. Các thông số cần biết trong hosting****

– Hệ điều hành (OS) của máy chủ : hiện tại có hai loại OS thông dụng là Linux và Windows.

     + [Hosting Linux](https://www.bkns.vn/hosting/hosting-cpanel.html): là Hosting chuyên hỗ trợ ngôn ngữ lập trình PHP, Joomla, các mã nguồn mở…

     + [Hosting Windows](https://www.bkns.vn/hosting/windows-hosting.html): Hosting Windows chuyên hỗ trợ về ngôn ngữ lập trình ASP, ASP.Net, HTML …. vì các Ngôn ngữ này, chạy chuyên trên Hosting Windows, do vậy khi load Web sẽ hỗ trợ tốt hơn, Hosting Windows có hỗ trợ ngôn ngữ PHP, nhưng chủ yếu, là hỗ trợ chính là ASP …

– Dung lượng: Bộ nhớ lưu trữ cho phép bạn tải file lên host

– Băng thông: Bandwidth (băng thông) là thông số chỉ dung lượng thông tin tối đa mà website được lưu chuyển qua lại mỗi tháng

– PHP: Phiên bản php hỗ trợ

– Max file: Số lượng file tối đa có thể upload lên host

– RAM: Bộ nhớ đệm

– Addon domain: Số lượng domain bạn có thể trỏ tới hosting

– Subdomain: Số lượng [tên miền](https://www.bkns.vn/ten-mien/dang-ky-ten-mien.html) phụ có thể tạo ra cho mỗi tên miền

– Park domain: Số lượng tên miền có thể parking

– [Email](https://www.bkns.vn/email/email-hosting-co-ban.html) accounts: Số lượng email đi kèm với hosting

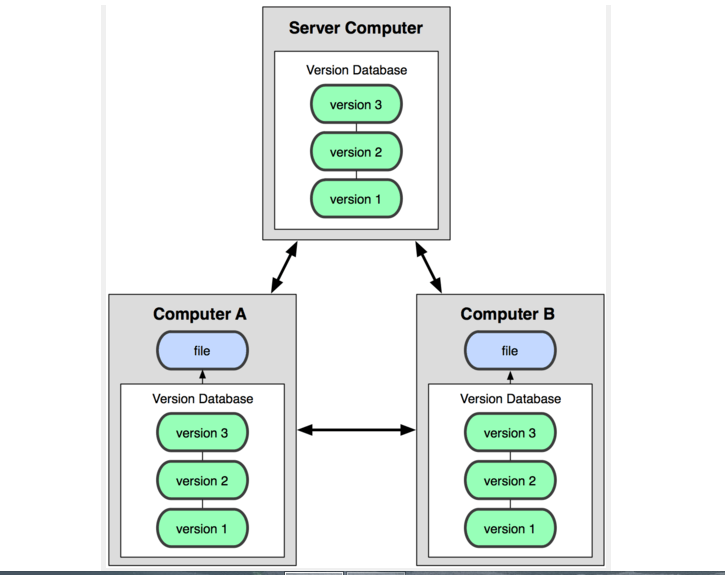
– FTP accounts: Số lượng FTP account bạn có thể tạo và dùng nó upload dữ liệu lên hosting



* Nếu không có hosting thì [website](http://www.bkweb.vn/) sẽ chỉ hoạt động trên máy tính bạn mà thôi, chỉ có mình bạn nhìn thấy, dữ liệu sẽ không được chia sẻ trên mạng. Vậy nên rất cần thiết để có một gói hosting.

III. Tech 3: Tìm hiểu công nghệ quản lý mã nguồn GIT: Tài khoản, SampleProject.

GITHub là hệ thống quản lý source phân tán( Distributed Version Control System – DVCS) là một trong những hệ thống quản lý phiên bản phân tán phổ biến nhất hiên nay. DVCS nghĩa là hệ thống iups mỗi máy tính có thể lưu trữ nhiều phiên bản khác nhau của một mã nguồn đucợ nhân bản( clone) từ một kho chứa mã nguồn( repository), mỗi thay đổi vào mã nguồn trên máy tính sẽ có thể ủy thác(commit) rồi đưa lên máy chủ nới đặt kho chứa chính và một máy tính khác ( nếu họ có ủy quyền truy cập) cũng có thể clone lại mã nguồn từ kho chứa hoặc clone lại một tập hợp các thay đổi mới nhất trên máy tính kia. Trong GIT thư mục làm việc trên máy tính gọi là Working Tree. Đại loại như dưới đây:



1. Tài khoản.

Để tạo một tài khoản GIT bạn cần có một tài khoản Gmail, sau đó truy cập tới trang chủ của GIT [*https://github.com*](https://github.com) điền thông tin cá nhân vào chọn create là xong.

2 . SampleProject.