BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC QUẢN LÝ VÀ CÔNG NGHỆ HẢI PHÒNG

ISO 9001:2015

ĐỒ ÁN TỐT NGHIỆP

NGÀNH : Công Nghệ Thông Tin

Sinh viên **:Đồng Văn Hiếu**

Giảng viên hướng dẫn **:TS. Đỗ Văn Chiểu**

HẢI PHÒNG – 2020

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC QUẢN LÝ VÀ CÔNG NGHỆ HẢI PHÒNG

TÌM HIỂU VỀ MỘT SỐ KỸ THUẬT LẬP TRÌNH THỜI GIAN THỰC BẰNG NGÔN NGỮ LẬP TRÌNH PHP

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY
NGÀNH: Công Nghệ Thông Tin

Sinh viên **:Đồng Văn Hiếu**
Giảng viên hướng dẫn **:TS.Đỗ Văn Chiểu**

HẢI PHÒNG – 2020

BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC QUẢN LÝ VÀ CÔNG NGHỆ HẢI PHÒNG

NHIỆM VỤ ĐỀ TÀI TỐT NGHIỆP

Sinh viên: **Đồng Văn Hiếu** **Mã SV**: 1512111016

Lớp : CT1901C

Ngành :Công Nghệ Thông Tin

Tên đề tài: **Tìm hiểu về một số kỹ thuật lập trình thời gian thực bằng ngôn ngữ lập trình php**

NHIỆM VỤ ĐỀ TÀI

1. **Nội dung và các yêu cầu cần giải quyết trong nhiệm vụ đề tài tốt nghiệp**

   Tìm hiểu một số kỹ thuật lập trình thời gian thực trong PHP

   - Jquery
   - Pusher
   - Socket.io
   - Ứng dụng demo

   ……

   ……

   ……

   ……

   ……

   ……

2. **Các tài liệu, số liệu cần thiết**

   ……

   ……

   ……

   ……

   ……

   ……

   ……

   ……

3. **Địa điểm thực tập tốt nghiệp**

   ……

CÁN BỘ HƯỚNG DẪN ĐỀ TÀI TỐT NGHIỆP

Họ và tên : Đỗ Văn Chiểu

Học hàm, học vị : Tiến Sĩ

Cơ quan công tác : Trường Đại học Quản lý và Công nghệ Hải Phòng

Nội dung hướng dẫn: Tìm hiểu về một số kỹ thuật lập trình thời gian thực bằng ngôn ngữ lập trình PHP

Đề tài tốt nghiệp được giao ngày 30 tháng 03 năm 2020

Yêu cầu phải hoàn thành xong trước ngày 30 tháng 06 năm 2020

Đã nhận nhiệm vụ ĐTTN	Đã giao nhiệm vụ ĐTTN
Sinh viên	*Giảng viên hướng dẫn*

Hải Phòng, ngày tháng năm 2020

HIỆU TRƯỞNG

LỜI CẢM ƠN

Lời đầu tiên em xin chân thành cảm ơn các thầy, cô trong khoa Công Nghệ Thông Tin cũng như toàn thể mọi người trong ngôi trường Đại học Dân lập Hải Phòng đã tạo điều kiện thuận lợi cho em trong suốt quá trình học tập tại trường cũng như trong thời gian thực hiện đồ án tốt nghiệp.

Đặc biệt, em muốn gửi lời cảm ơn tới Tiến Sĩ– Đỗ Văn Chiểu giảng viên trực tiếp hướng dẫn tận tình chỉ bảo giúp em khắc phục những khó khăn, thiếu sót để có thể hoàn thành các phần trong đồ án tốt nghiệp từ lý thuyết cho tới thực hành sử dụng công cụ.

Với hiểu biết tìm tòi của bản thân và sự chỉ bảo hướng dẫn tận tình của giảng viên em đã cố gắng hoàn thành đồ án một cách tốt nhất có thể nhưng cũng không thể tránh được thiếu sót. Kính mong nhận được sự đóng góp ý kiến từ thầy cô để em có thể nâng cao cũng như bổ sung thêm kiến thức cho bản thân, hoàn thiện đồ án với một kết quả tốt và hoàn chỉnh hơn.

Em xin chân thành cảm ơn!

Hải Phòng, ngày 21 tháng 06 năm 2019.

Sinh viên thực hiện

Đồng Văn Hiếu

MỤC LỤC

Mở Đầu

Với sự phát triển rất mau lẹ của Internet, ngời dùng ngày càng quan tâm hơn đến hình thức, tốc độ, tính năng của một trang web. Trước đây một trang web chỉ cần có banner, nội dung và ít footer hời hợt là đã được cho là một trang web hoàn chỉnh. Nhưng bây giờ trang web đó phải có banner bắt mắt, nội dung hay và còn nhiều hiệu ứng lạ mắt khác nữa đồng thời tốc độ xử lý và các tính năng phù hợp thì mới có thể thu hút được người dùng. Chính vì thế để xây dựng một trang web có đầy đủ các tính năng theo yêu cầu của người dùng các nhà phát triển, thiết kế web đã kết hợp việc sử dụng ngôn ngữ PHP - ngôn ngữ lập trình web phổ biến nhất hiện nay – với việc sử dụng các kỹ thuật lập trình thời gian thực . Trên cơ sở đó tôi hi vọng rằng việc tìm hiểu về PHP và một số kỹ thuật lập trình thời gian thực trong việc xây dựng các ứng dụng web sẽ giúp nâng cao kiến thức của bản thân trong việc lập trình và xây dựng các ứng dụng web. Đồng thời giúp các bạn sinh viên mới đi sâu về lĩnh vực này có thể hiểu rõ hơn về PHP và các kỹ thuật lập trình thời gian thực .

Đồ án gồm 3 chương:

Chương 1: Tổng hợp các kiến thức cơ bản về PHP. Đưa ra cái nhìn tổng quan về ngôn ngữ lập trình web phổ biến này.

Chương 2: Một số kỹ thuật lập trình thời gian thực trong PHP

Chương 3: Ứng dụng kỹ thuật lập trình thời gian thực trong PHP.

Cuối cùng là kết luận và tài liệu tham khảo.

Chương 1: TỔNG QUAN VỀ PHP

1.1. Lịch sử phát triển

1.1.1. PHP

Được phát triển từ một sản phẩm có tên là PHP/FI. PHP/FI do Rasmus Lerdorf tạo ra năm 1995, ban đầu được xem như là một tập con đơn giản của các mã kịch bản Perl để theo dõi tình hình truy cập đến bản sơ yếu lý lịch của ông trên mạng. Ông đã đặt tên cho bộ mã kịch bản này là 'Personal Home Page Tools'..

1.1.2. PHP3

PHP 3.0 là phiên bản đầu tiên cho chúng ta thấy một hình ảnh gần gũi với các phiên bản PHP mà chúng ta được biết ngày nay. Nó đã được Andi Gutmans và Zeev Suraski tạo ra năm 1997 sau khi viết lại hoàn toàn bộ mã nguồn trước đó. PHP 3.0 đã chính thức được công bố vào tháng 6 năm 1998, sau thời gian 9 tháng được cộng đồng kiểm nghiệm.

1.1.3. PHP4

Vào mùa đông năm 1998, ngay sau khi PHP 3.0 chính thức được công bố, Andi Gutmans và Zeev Suraski đã bắt đầu bắt tay vào việc viết lại phần lõi của PHP. Một động cơ mới, có tên 'Zend Engine' (ghép từ các chữ đầu trong tên của Zeev và Andi), đã đáp ứng được các nhu cầu thiết kế này một cách thành công, và lần đầu tiên được giới thiệu vào giữa năm 1999. PHP 4.0, dựa trên động cơ này, và đi kèm với hàng loạt các tính năng mới bổ sung, đã chính thức được công bố vào tháng 5 năm 2000, gần 2 năm sau khi bản PHP 3.0 ra đời

1.1.4. PHP5

Sự thành công hết sức to lớn của PHP 4.0 đã không làm cho nhóm phát triển PHP tự mãn. Cộng đồng PHP đã nhanh chóng giúp họ nhận ra những yếu kém của PHP 4 đặc biệt với khả năng hỗ trợ lập trình hướng đối tượng (OOP), xử lý XML, không hỗ trợ giao thức máy khách mới của MySQL 4.1 và 5.0, hỗ trợ dịch vụ web yếu. Những điểm này chính là mục đích để Zeev và Andi viết Zend Engine 2.0, lõi của PHP 5.0. Ngày 29 tháng 6 năm 2003, PHP 5 Beta 1 đã chính thức được công bố để cộng đồng kiểm nghiệm. Đó cũng là phiên bản đầu tiên của Zend Engine 2.0. Phiên bản Beta 2 sau đó đã ra mắt vào tháng 10 năm 2003 với sự xuất hiện của hai

tính năng rất được chờ đợi: Iterators, Reflection nhưng namespaces một tính năng gây tranh cãi khác đã bị loại khỏi mã nguồn. Ngày 21 tháng 12 năm 2003: PHP 5 Beta 3 đã được công bố để kiểm tra với việc phân phối kèm với Tidy, bỏ hỗ trợ Windows 95, khả năng gọi các hàm PHP bên trong XSLT, sửa chữa nhiều lỗi và thêm khá nhiều hàm mới. PHP 5 bản chính thức đã ra mắt ngày 13 tháng 7 năm 2004 sau một chuỗi khá dài các bản kiểm tra thử bao gồm Beta 4, RC 1, RC2, RC3. Mặc dù coi đây là phiên bản sản xuất đầu tiên nhưng PHP 5.0 vẫn còn một số lỗi trong đó đáng kể là lỗi xác thực HTTP.

1.1.5. PHP6

Hiện nay phiên bản tiếp theo của PHP đang được phát triển, PHP 6 bản sử dùng thử đã có thể được download tại địa chỉ http://snaps.php.net. Phiên bản PHP 6 được kỳ vọng sẽ lấp đầy những khiếm khuyết của PHP ở phiên bản hiện tại, ví dụ: hỗ trợ namespace (hiện tại các nhà phát triển vẫn chưa công bố rõ ràng về vấn đề này); hỗ trợ Unicode; sử dụng PDO làm API chuẩn cho việc truy cập cơ sở dữ liệu, các API cũ sẽ bị đưa ra thành thư viện PECL...

1.1.6. PHP7

Sau khi PHP 5 ra mắt vào năm 2004 thì từ đó đến 2015 không hề có một phiên bản nâng cấp lớn nào của PHP, chủ yếu là các bản vá lỗi, cải thiện hiệu suất và một vài tính năng mới mới như lập trình hướng đống tượng, … cộng đồng PHP luôn luôn mong chờ có một cái gì đó thật sự đổi mới đối với PHP và vào cuối tháng 12 năm 2015 PHP 7 đã ra mắt chính thức với hàng loạt tính năng mới sau một thời gian dài beta (PHPNG).

1.2. Cấu trúc cơ bản của PHP

1.2.1. Các cấu trúc cơ bản

PHP cũng có thẻ bắt đầu và kết thúc giống với ngôn ngữ HTML. Chỉ khác, đối với PHP chúng ta có nhiều cách để thể hiện.

Cách 1: Cú pháp chính:

<?php Mã lệnh PHP ?>

Cách 2: Cú pháp ngắn gọn

<? Mã lệnh PHP ?>

Cách 3: Cú pháp giống với ASP.

<% Mã lệnh PHP %>

Cách 4: Cú pháp bắt đầu bằng script

<script language=php>

.....

</script>

Mặc dù có 4 cách thể hiện. Nhưng đối với 1 lập trình viên có kinh nghiệm thì việc sử dụng cách 1 vẫn là lựa chon tối ưu.

Trong PHP để kết thúc 1 dòng lệnh chúng ta sử dụng dấu ";"

Để chú thích 1 đoạn dữ liệu nào đó trong PHP ta sử dụng dấu "//" cho từng dòng.
Hoặc dùng cặp thẻ "/*……..*/" cho từng cụm mã lệnh.

Ví dụ: <?php echo ”Hello ”; ?>

1.2.2. Xuất giá trị ra trình duyệt

Để xuất dữ liệu ra trình duyệt chúng ta có những dòng cú pháp sau:

- echo "Thông tin";

- printf "Thông tin";

Thông tin bao gồm: biến, chuỗi, hoặc lệnh HTML ….

```
1  <?php
2  Echo "Hello word";
3  Printf"<br><font color=red>Who Are You ?</font>";
4  ?>
5
```

Hình 1. Xuất dữ liệu

Nếu giữa hai chuỗi muốn liên kết với nhau ta sử dụng dấu "."

1.2.3. Biến, hằng, chuỗi và các kiểu dữ liệu

1.2.3.1 Biến

Biến được xem là vùng nhớ dữ liệu tạm thời. Và giá trị có thể thay đổi được. Biến được bắt đầu bằng ký hiệu "$". Và theo sau chúng là 1 từ, 1 cụm từ nhưng phải viết liền hoặc có gạch dưới.

Một biến được xem là hợp lệ khi nó thỏa các yếu tố:

- Tên của biến phải bắt đầu bằng dấu gạch dưới và theo sau là các ký tự, số hay dấu gạch dưới.

- Tên của biến không được phép trùng với các từ khóa của PHP.

Trong PHP để sử dụng 1 biến chúng ta thường phải khai báo trước, tuy nhiên đối với các lập trình viên khi sử dụng họ thường xử lý cùng một lúc các công việc, nghĩa là vừa khái báo vừa gán dữ liệu cho biến.

Bản thân biến cũng có thể gán cho các kiểu dữ liệu khác. Và tùy theo ý định của người lập trình mong muốn trên chúng.

```
<?
$a= 100 // biến a ở đây có giá trị là 100.
$a= "PHP is easy" // Biến a ở đây có giá trị "PHP Is easy".
Biena=123 //Có lỗi vì bắt đầu 1 biến phải có dấu "$"
$123a="PHP" //Có lỗi vì phần tên bắt đầu của biến là dạng số.
?>
```

Hình 2. Biến trong PHP

1.2.3.2 Hằng

Nếu biến là cái có thể thay đổi được thì ngược lại hằng là cái chúng ta không thể thay đổi được. Hằng trong PHP được định nghĩa bởi hàm define theo cú pháp:

define (string tên_hằng, giá_trị_hằng).

Cũng giống với biến hằng được xem là hợp lệ thì chúng phải đáp ứng 1 số yếu tố:

- Hằng không có dấu "$" ở trước tên.

- Hằng có thể truy cập bất cứ vị trí nào trong mã

lệnh + - Hằng chỉ được phép gán giá trị duy nhất 1 lần.

- Hằng thường viết bằng chữ in để phân biệt với biến

```
<?
define ("C", "COMPANY");
define ("YELLOW", "#ffff00");
echo "Gia tri cua C la". C;
?>
```

Hình 3. Hằng trong PHP

1.2.3.3 Chuỗi

Chuỗi là một nhóm các kỹ tự, số, khoảng trắng, dấu ngắt được đặt trong các dấu nháy.

Ví dụ:

„Hello“

Để tạo 1 biến chuỗi, chúng ta phải gán giá trị chuỗi cho 1 biến hợp lệ.

Ví dụ:

$fisrt_name= "Nguyen";

$last_name= „Van A“;

Để liên kết 1 chuỗi và 1 biến chúng ta thường sử dụng dấu "."

```php
<?php
$test="QHOnline.Info";
echo "welcome to".$test;
echo "<br><font color=red>welcome to".$test."</font><br>";
?>
```

Hình 4. Liên kết chuỗi và biến trong PHP

1.2.3.4 Kiểu dữ liệu

Các kiểu dữ liệu khác nhau chiếm các lượng bộ nhớ khác nhau và có thể được xử lý theo cách khác nhau khi chúng được theo tác trong 1 script.

Trong PHP chúng ta có 6 kiểu dữ liệu chính như sau:

Kiểu Dữ Liệu	Ví dụ	Mô Tả
Integer	10	Một số nguyên
Double	5.208	Kiểu số thực
String	"How are you ?"	Một tập hợp các ký tự
Boolean	True or False	Giá trị true hoặc false
Object	Hướng đối tượng trong PHP	
Array	Mảng trong PHP, chứa các phần tử.	

Hình 5. Kiểu dữ liệu trong PHP

Chúng ta có thể sử dụng hàm dựng sẵn GETTYPE() của PHP4 để kiểm tra kiểu của bất kỳ biến.

```php
<?php
$a= 5;
echo gettype($a); // Integer.
$a="qhonline.info";
echo gettype($a); //String
?>
```

Hình 6. Hàm GETTYPE

1.2.4. Các phương thức được sử dụng trong lập trình PHP

1.2.4.1 Phương thức GET

Phương thức này được dùng để lấy dữ liệu từ *form* nhập liệu. Tuy nhiên nhiệm vụ chính của nó vẫn là lấy nội dung từ trang dữ liệu từ web server.

Ví dụ:

Với url sau: shownews.php?id=50

Vậy với trang shownews ta dùng hàm $_GET[„id“] sẽ được giá trị là 50.

1.2.4.2 Phương thức POST

Phương thức này được sử dụng để lấy dữ liệu từ form nhập liệu. Và chuyển chúng lên trình chủ webserver.

```
<?php
    echo "Welcome ".$_POST['hoten']."!";
?>
<html>
    <form name="test" action="#" method="POST">
        Họ tên <input  type="text" name="hoten"/>
        <input type="submit" name="OK" value="OK"/>
    </form>
</html>
```

Hình 7. Phương thức POST

1.2.5. Cookie và Session trong PHP

Cookie và Session là hai phương pháp sử dụng để quản lý các phiên làm việc giữa người sử dụng và hệ thống

1.2.5.1 Cookie

Cookie là 1 đoạn dữ liệu được ghi vào đĩa cứng hoặc bộ nhớ của máy người sử dụng. Nó được trình duyệt gởi ngược lên lại server mỗi khi browser tải 1 trang web từ server.

Những thông tin được lưu trữ trong cookie hoàn toàn phụ thuộc vào Website trên server. Mỗi Website có thể lưu trữ những thông tin khác nhau trong cookie, ví dụ thời điểm lần cuối ta ghé thăm Website, đánh dấu ta đã login hay chưa, v.v...

Cookie được tạo ra bởi Website và gởi tới browser, do vậy 2 Website khác nhau (cho dù cùng host trên 1 server) sẽ có 2 cookie khác nhau gởi tới browser. Ngoài ra, mỗi browser quản lý và lưu trữ cookie theo cách riêng của mình, cho nên 2 browser cùng truy cập vào 1 Website sẽ nhận được 2 cookie khác nhau

- Để thiết lập cookie ta sử dụng cú pháp:

Setcookie("tên cookie","giá trị", thời gian sống) Tên cookie là
tên mà chúng ta đặt cho phiên làm việc.
Giá trị là thông số của tên cookie.
Ví dụ: setcookie("name","admin",time()+3600);

- Để sử dụng lại cookie vừa thiết lập, chúng ta sử dụng cú pháp:

Cú pháp: $_COOKIE["tên cookies"]
Tên cookie là tên mà chúng ta thiết lập phía trên.

- Để hủy 1 cookie đã được tạo ta có thể dùng 1 trong 2 cách sau:

- Cú pháp: setcookie("Tên cookie")
Gọi hàm setcookie với chỉ duy nhất tên cookie mà thôi
- Dùng thời gian hết hạn cookie là thời điểm trong quá khứ.
Ví dụ: setcookie("name","admin",time()-3600);

1.2.5.2 Session

Một cách khác quản lý người sử dụng là Session. Session được hiểu là khoảng thời gian người sử dụng giao tiếp với 1 ứng dụng. Một session được bắt đầu khi người sử dụng truy cập vào ứng dụng lần đầu tiên, và kết thúc khi người sử dụng thoát khỏi ứng dụng. Mỗi session sẽ có được cấp một định danh (ID) khác nhau.

- Để thiết lập 1 session ta sử dụng cú pháp: session_start()

Đoạn code này phải được nằm trên các kịch bản HTML. Hoặc những lệnh echo, printf. Để thiết lập 1 giá trị session, ngoài việc cho phép bắt đầu thực thi session. Chúng ta còn phải đăng ký 1 giá trị session. Để tiện cho việc gán giá trị cho session đó.

- Ta có cú pháp sau: session_register("Name")

Giống với cookie. Để sử dụng giá trị của session ta sử dụng mã lệnh sau:

-Cú pháp: $_SESSION[“name”]

Với Name là tên mà chúng ta sử dụng hàm session_register(“name”) để khai báo.

- Để hủy bỏ giá trị của session ta có những cách sau:

session_destroy()

// Cho phép hủy bỏ toàn bộ giá trị của session session_unset()//Cho phép hủy bỏ session

1.2.6. Cookie và Session trong PHP

Để giảm thời gian lặp lại 1 thao tác code nhiều lần, PHP hỗ trợ người lập trình việc tự định nghĩa cho mình những hàm có khả năng lặp lại nhiều lần trong Website.

Việc này cũng giúp cho người lập trình kiểm soát mã nguồn một cách mạch lạc. Đồng thời có thể tùy biến ở mọi trang. Mà không cần phải khởi tạo hay viết lại mã lệnh như HTML thuần

1.2.6.1 Hàm tự định nghĩa

Cú pháp:

function function_name() {

//Lệnh thực thi

}

Tên hàm có thể là một tổ hợp bất kỳ những chứ cái, con số và dấu gạch dưới, nhưng phải bắt đầu từ chứ cái và dấu gạch dưới.

1.2.6.2 Hàm tự định nghĩa với các tham số

Cú pháp:

function function_name($gt1,$gt2) {

//Lệnh thực thi

}

1.2.6.3 Hàm tự định nghĩa với giá trị trả về

Cú pháp:

function function_name(Có hoặc không có đối số) {

// Lệnh thực thi

return giatri; }.

1.2.6.4 Gọi lại hàm

PHP cung cấp nhiều hàm cho phép triệu gọi lại file. Như hàm include("URL đến file"), require("URL Đến file").

Ngoài hai cú pháp trên còn có include_once(), require_once(). Hai hàm này cũng có trách nhiệm gọi lại hàm. Những chúng sẽ chỉ gọi lại duy nhất 1 lần mà thôi.

CHƯƠNG 2: MỘT SỐ KỸ THUẬT LẬP TRÌNH THỜI GIAN THỰC TRONG PHP

2.1. Các cú pháp jQuery cơ bản

2.1.1. Cài đặt thư viện jQuery

- Hiện tại đang là phiên bản jQuery 3.5.1, ta có thể download từ trang chủ *www.jquery.com* có 2 phiên bản:

+ Phiên bản product(bản nén) dùng cho sản phẩm có dung lượng thấp khoảng 90,45KB
+ Phiên bản dùng cho nhà phát triển(bản chưa nén) có dung lượng khoảng 262,09 KB

- Cài đặt jQuery:

<script src="http://code.jquery.com/jquery-3.5.1.min.js"></script>

<script src="http://code.jquery.com/jquery-3.5.1.js"></script>

2.1.2. Gọi hàm jQuery

- Cú pháp: *$(selector).action()*
- ” $”: Ký hiệu của jQuery, có thể thay thế bằng từ khóa “jQuery”
- selector: là tên của các đối tượng cần truy cập hoặc con đường truy xuất đến các đối tượng cần truy cập

+ action: thực hiện các hành động trên các selector

2.1.3. jQuery Selector

jQuery selector giúp chúng ta dễ dàng truy vấn đến các phần tử DOM (Document Object Model – Mô hình đối tượng tài liệu) một cách nhanh nhất, code đơn giản và ngắn gọn nhất.

jQuery Selector có thể là : Thẻ html, id, class hoặc một đối tượng

- ID Selector: Bắt đầu bằng dấu “.” Và sau đó là Id element, cú pháp : *$(“#id_name”)*
- Class: Bắt đầu là dấu *”.”* và tên class, Cú pháp: *$(“.class_name”)*
- Một đối tượng:

VD: *var obj = $('#name_id')*

$(obj).html(');

- Element Selector: Tên của các element như : div, ul, li, table, tr, td , span dựa trên các thẻ html để hình thành con đường truy xuất

Cú pháp : VD: $(“tag_name”)

```
$(document).ready(function(){
$("button").click(function(){
  $("p").hide(); });
});
```

VD:

```
<div id = “firstID”>
<ul>
                        <li> <a class=”red”>red Link</a></li>
</ul>
</div>
```

Để gọi đến thẻ “a” ta có thể gọi : *$(„div#first ul li a.red")*

* jQuery Selector sử dụng cơ chế giống như css selector (cho phép lựa chọn các đối tượng DOM dựa trên nguyên tắc truy vấn CSS)

Một số ví dụ về jQuery

CÚ PHÁP	MÔ TẢ
$("")*	Chọn tất cả các thành phần
$(this)	Chọn thành phần HTML hiện tại

$("p.intro")	Chọn tất cả các thẻ <p> có class là “intro”
$("p:first")	Chọn thẻ <p> đầu tiên
$("ul li:first")	Chọn thẻ đầu tiên của thẻ đầu tiên
$("ul li:first-child")	Chọn thẻ đầu tiên của tất cả các thẻ
$("[href]")	Chọn tất cả các thành phần có thuộc tính là href
$("a[target='_blank']")	Chọn tất cả các thẻ <a> với thuộc tính target có giá trị: "_blank"
$("a[target!='_blank']")	Chọn tất cả các thẻ <a> với thuộc tính target có giá trị khác: "_blank"

$(":button")	Chọn tất cả các thẻ <button> và thẻ <input> có thuộc tính type="button"
$("tr:even")	Chọn tất cả các thẻ <tr> chẵn
$("tr:odd")	Chọn tất cả các thẻ <tr> lẻ

Ngoài ra còn có rất nhiều các selector khác

2.1.4. jQuery Attribute

Những phương pháp jquery nhận và thiết lập các thuộc tính của các yếu tố DOM. Các thuộc tính người lập trình jquery hay dùng.

2.1.4.1. Class

a) AddClass(class)

- Kiểu trả về: jQuery

- Thêm các class đã xác định vào mỗi tập phần tử phù hợp. Nếu có thêm nhiều class thì các class được cách nhau bởi khoảng trắng.

- Ví dụ: Thêm class "styleText" vào các thẻ p cuối cùng

```
<!DOCTYPE html>
<html>
<head>
  <style>
  p.styleText { text-decoration: underline}
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
</head>
<body>
  <p>Hello</p>
  <p>and</p>
  <p>then</p>
  <p>Goodbye</p>
<script>$("p:last").addClass("styleText");</script>
 </body>
</html>
```

Kết quả

Hello

and

then

Goodbye

b) RemoveClass(class)

- Kiểu trả về: jQuery
- Loại bỏ tất cả hoặc các class đã xác định khỏi tập phần tử phù hợp.
- Ví dụ: Loại bỏ lass “styleText” khỏi thẻ p cuối cùng.

```
<!DOCTYPE
html> <html>
<head>
<style>
p.styleText { text-decoration: underline}
</style>
<script src="http://code.jquery.com/jquery-latest.js"></script>
</head>
<body>
<p      class=”styleText”>Hello</p>
<p  class=”styleText”>and</p>  <p
class=”styleText”>then</p>        <p
class=”styleText”>Goodbye</p>
<script>$("p:last").addClass("styleText");</script
> </body>
</html>
```

Kết quả

Hello

and

then

Goodbye

c) toggleClass(class)

Kiểu trả về: jQuery

Thêm class nếu class chưa tồn tại hoặc loại bỏ nếu class đã tồn tại.

Ví dụ: Thêm class “styleText” vào thẻ p nếu class “styleText” chưa tồn tại trong thẻ p hoặc loại bỏ class “styleText” khỏi thẻ p nếu nó tồn tại.

```
<!DOCTYPE
html> <html>
```

```
<head>
<style>
 p.styleText { text-decoration:
underline} </style>
 <script src="http://code.jquery.com/jquery-
latest.js"></script> </head>
<body>
 <p
 class=”styleText”>Hello</p>
 <p class=”styleText”>and</p>
 <p class=”styleText”>then</p>
 <p>Goodbye</p>
<script>$("p”). toggleClass("styleText");</script>
 </body>
</html>
```

Kết quả

Hello
and
then
Goodbye

2.1.4.2. HTML, Text

a) html()

- Kiểu trả về: String
- Lấy nội dung html (innerHTML) của phần tử.
- Ví dụ: Mỗi khi click vào thẻ p lấy nội dung html của thẻ p đó và thông báo nội dung lấy được.

```
$("p").click(function() {alert($(this).html())});
```

b) html(val)

- Kiểu trả về: jQuery
- Thiết lậ nội dung html (innerHTML) cho phần tử.
- Ví dụ: Thiết lập nội dung html cho thẻ div.

```
$("div").html("<b>Chào các bạn!<i> Chúc buổi học hôm nay thú vị.</i></b>");
```

c) text()

- Kiểu trả về: String

- Lấy nội dung text (innerText) của phần tử.

- Ví dụ: Mỗi khi click vào thẻ p lấy nội dung text của thẻ p đó và thông báo nội dung lấy được.

$("p").click(function(){alert($(this).html())});

d) text(val)

- Kiểu trả về: jQuery

- Thiết lập nội dung text (innerText) cho phần tử.

- Ví dụ: Thiết lập nội dung text cho thẻ div.

$("div").text("Chào các bạn! Chúc buổi học hôm nay thú vị");

2.1.5 JQuery Events

JavaScript có một số cách được lập sẵn để phản ứng với những tương tác của người dùng và những sự kiện khác. Để làm cho trang web năng động và tương tác tốt, chúng ta cần phải tận dụng chức năng này, để vào những thời điểm phù hợp, chúng ta có thể sử dụng những kỹ thuật jQuery. Cũng có thể làm với JavaScript, nhưng jQuery nâng cao và mở rộng những cơ chế quản lý sự kiện cơ bản để giúp nó có cú pháp đẹp hơn, tiết kiệm thời gian hơn và tất nhiên cũng mạnh mẽ hơn.

jQuery cung cấp cho chúng ta khá nhiều các sự kiện đủ để thao tác với các thành phần trên website mà chúng ta mong muốn. Bên cạnh đó, jQuery còn loại bỏ hoàn toàn các event code ra khỏi mã HTML như cách viết javascript thông thường

Trong javascript, để xuất hiện thông báo với người dùng khi click vào liên kết ta phải viết như sau:

Click me

Ta thấy rằng chúng ta phải viết mã để gọi sự kiện onclick ngay trong liên kết <a>. Nhưng với jQuery thì chúng ta có thể sử dụng bộ selector để thao tác với các sự kiện như sau:

$("a.click").click(function() { alert("Welcome to Ewebvn.Com"); });

2.1.5.1. Cách viết các phương thức xử lý sự kiện và các sự kiện phổ biến

Javascript cung cấp Bộ quản lý sự kiện window.onload cho phép chúng ta thực thi một hàm nào đó. Trong khi đó jQuery cung cấp cho chúng ta bộ quả lý sự kiện $(document).ready() nhưng với những ưu điểm vượt trội hơn sự kiện window.onload

window.onload sẽ được thực thi sau khi trình duyệt tải xong toàn bộ tài liệu cần thiết bao gồm stylesheet, hình ảnh…

Trong khi đó $(document).ready() sẽ được thực thi ngay khi các phần tử DOM được sẵn sàng mà không cần đợi các tài liệu như stylesheet hay hình ảnh tải xong. Do đó quá trình xử lý vào thao tác sẽ diễn ra nhanh hơn.

- Cú pháp viết hàm sự kiện trong jQuery

```
$(document).ready(function() {
//Code và các chương trình thực thi
})
```

Hoặc

```
$().ready(function() {
//Code và các chương trình thực thi
})
```

Hoặc

```
$(function() {
//Code và các chương trình thực thi
}
```

- Một số sự kiện thường sử dụng trong jQuery

Mouse Events	Keyboard Events	Form Events	Document/Window Events
Click	*keypress*	*submit*	*load*
dblclick	*keydown*	*change*	*resize*
mouseenter	*keyup*	*focus*	*scroll*

a) click()

- Phương thức click() gắn với hàm xử lý sự kiện cho một thành phần của HTML
- Hàm sẽ được thực hiện khi người dùng nhấp đúp chuột vào một thành phần của HMTL

VD: Khi nhấp chuột vào thẻ p, thẻ này

mouseleave		*blur*	*unload*

b) dblclick()

- Phương thức dblclick() gắn với hàm xử lý sự kiện cho một thành phần của HTML
- Hàm sẽ được thực hiện khi người dùng nhấp chuột vào một thành phần của HMTL

VD: Khi nhấp đúp chuột vào thẻ p, thẻ này sẽ bị ẩn

đi *$("p").dblclick(function(){*

$(this).hide()

; });

c) mouseenter

- Phương thức dblclick() gắn với hàm xử lý sự kiện cho một thành phần của HTML
- Hàm sẽ được thực hiện khi người dùng trỏ chuột vào một thành phần của HMTL

```
$("#p1").mouseenter(function()
  { alert("You entered p1!");
});
```

d) mouseleave()

- Hàm sẽ được thực hiện khi người dùng dời con trỏ chuột khỏi một thành phần của HMTL

$("#p").mouseleave(function(){

```
  alert("Bye! You now leave p!");
});
```

e) mousedown():

- Hàm sẽ được thực hiện khi nhấn trái chuột vào một thành phần của HTML

```
$("#p1").mousedown(function(){
  alert("Mouse down over p1!");
});
```

f) mouseup()

- Hàm sẽ gọi sự kiện này khi nhả chuột trái

```
$("#p1").mouseup(function(){
  alert("Mouse up over p1!");
});
```

g) hoever()

- Hàm được gọi khi di chuyển con trỏ chuột đến một thành phần của HTML

```
$("#p1").hover(function(){
  alert("You entered p1!"); },
  function(){
  alert("Bye! You now leave p1!"); });
```

h) focus()

- Sự kiện xảy ra khi ta đang focus vào 1 element(con nháy trong textbox, viền trên button, radio, checkbox...).

```
$("input").focus(function(){
  $(this).css("background-color","#cccccc");
});
```

i)blur()

- Ngược lại với focus

```
$("input").blur(function(){
  $(this).css("background-color","#ffffff");
});
```

Ngoài ra còn có rất nhiều hàm khác, trong tùy từng các trường hợp cụ thể ta có thể sử dụng với các mục đích khác nhau. Ta có thể vào trang chủ của jQuery để tra cứu thêm.

2.2. JQUERY EFFECTS

Các hiệu ứng động của jQuery sẽ làm cho trang web thêm phần sinh động. Jquery cho phép ẩn hiện, trượt lên trượt xuống các thành phần của trang web. Bạn cũng có thể cho nó xảy ra cùng một lúc hoặc theo thứ tự định trước.

2.2.1. Hide/Show

2.2.1.1 hide() and show()

- Phương thức giúp ẩn, hiện các thành phần của HTML.

VD:

```
$("#hide").click(function()
  { $("p").hide();
});
$("#show").click(function()
  { $("p").show();
});
```

- Cú pháp

```
$(selector).hide(speed,callback);
$(selector).show(speed,callback);
```

Ta cũng có thể thêm vào một số tham biến để tùy chọn tốc độ việc ẩn đi hay hiển thị ra các thành phần, và giá trị của chúng có thể là “slow”, “fast” hoặc milliseconds

VD:

```
$("#hide").click(function(){
  $("p").hide(slow, alert);
});
$("#hide").click(function(){
  $("p").hide(1000);
});
```

2.2.1.2 toggle()

jQuery cung cấp thêm một phương thức giúp ta có thể ẩn đi và hiện lên các selector, ta có thể lựa chọn giữa việc sử dụng hàm này hoặc sử dụng phương thức hide() và show() đã giới thiệu ở trên. Nó giúp ta ẩn đi các selector đang hiện và hiện lại các selector đã bị ẩn đi

- Cú pháp: về cơ bản phương thức này cũng có cú pháp tương tự như hide() và show()

```
$(selector).toggle(speed,callbac
k);
```

Ta cũng có thể thêm vào một số tham biến để tùy chọn tốc độ việc ẩn đi hay hiển thị ra các thành phần, và giá trị của chúng có thể là “slow”, “fast” hoặc milliseconds. Đông thời cũng có thể thêm vào một hành động nào đó để thực hiện sau thực hiện xong *toggle()*

VD: Sau khi thực hiện vệc ẩn/hiện ra các thẻ p, sẽ xuất hiện một thông báo “OK”

```
<!DOCTYPE
html> <html>
<head>
<script
src="//ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js">
</script>
<script>
$(document).ready(functio
 n(){
 $("button").click(function
 (){
  $("p").toggle(“slow”, alert(„OK"));
 });
});
</script>
</head>
<body>
<button>Toggle</button>
```

* This is a paragraph with little content. This is another small paragraph. </body>*
</html>

2.2.2. Animate

- jQuery cung cấp phương thức giúp ta hỗ trợ việc tạo hiệu ứng cho các element như tăng kích cỡ, dịch chuyển vị trí bằng cách định dạng thêm các thuộc tính CSS

- Cú pháp:

$(selector).animate({params},speed,callback);

- Với phương thức này ta có thể thêm một hoặc nhiều thuộc tính CSS, giúp tạo thêm các hiệu ứng như ý muốn

VD:

```
<!DOCTYPE html>
<html>
<head>
<script src="//ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js">
</script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("div").animate({
      left:'250px',
      opacity:'0.5',
      height:'150px',
      width:'150px'
    });
  });
});
```

</script>

</head>

<body>

<button>Start Animation</button>

* By default, all HTML elements have a static position, and cannot be moved. To manipulate the position, remember to first set the CSS position property of the element to relative, fixed, or absolute! *

* *

* *

</body>

</html>

- Hoặc ta có thể sử dụng thêm các giá trị tương đối

```
$("button").click(function(){
  $("div").animate({
    left:'250px',
    height:'+=150px',
    width:'+=150px'
  });
});
```

- Ta cũng có thể sử dụng các thuộc tính để thực hiện các hiệu ứng : "show", "hide", or "toggle":

```
$("button").click(function(){
  $("div").animate({

    height:'toggle'
  });
});
```

- Ta cũng có thể thực hiện việc gọi ra nhiều hiệu ứng :

```
$("button").click(function(){
  var div=$("div");
  div.animate({height:'300px',opacity:'0.4'},"slow");
  div.animate({width:'300px',opacity:'0.8'},"slow");
  div.animate({height:'100px',opacity:'0.4'},"slow");
  div.animate({width:'100px',opacity:'0.8'},"slow");
});
```

2.2.3. Stop

- Khi phương thức này được gọi, các hiệu ứng đang xảy ra sẽ bị dừng lại. Giả sử nếu một element đang bị ẩn đi bởi phương thức slideUp(), thì khi phương thức stop() được gọi, element sẽ vẫn hiển thị ra nhưng chiều cao sẽ giảm đi.

- Cú pháp:

```
$(selector).stop(stopAll,goToEnd);
```

VD:

```
<!DOCTYPE html>
<html>
<head>
<script src="//ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js">
</script>
<script>
$(document).ready(function(){
  $("#flip").click(function(){
    $("#panel").slideDown(5000);
  });

  $("#stop").click(function(){
```

```
    $("#panel").stop();
  });
});
</script>
<style type="text/css">
#panel,#flip
{
padding:5px;
text-align:center;
background-color:#e5eecc;
border:solid 1px #c3c3c3;
}
#panel
{
padding:50px;
display:none;
}
</style>
</head>
<body>
<button id="stop">Stop sliding</button>
<div id="flip">Click to slide down panel</div>
<div id="panel">Hello world!</div>
</body>
</html>
```

Để tạo các hiệu ứng effect, jQuery đã cũng cấp thêm rất nhiều phương thức, ta có thể tìm hiểu thêm các phương thức này ở trang chủ của jQuery. Ngoài ra ta có

thể tham khảo thêm về jQuery UI đây là bộ thư viện JavaScripts viết dựa trên nền jQuery.

2.3. jQuery HTML

2.3.1. jQueryGet

2.3.1.1 Lấy ra nội dung:

Để lấy ra nội dung của các thành phần, jQuery cung cấp 3 phương thức cơ bản:

* text(): trả về nội dung của đoạn văn bản được lựa chọn

* html(): trả về nội dung của thành phần được lựa chọn (bao gồm cả các ký tự HTML)

VD:

```
<!DOCTYPE html>
<html>
<head>
<script src="//ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js">
</script>
<script>
$(document).ready(function(){
  $("#btn1").click(function(){
    alert("Text: " + $("#test").text());
  });
  $("#btn2").click(function(){
    alert("HTML: " + $("#test").html());
```

```
  });
});
</script>
</head>
<body>

<p id="test">This is some <b>bold</b> text in a paragraph.</p> <button id="btn1">Show Text</button>
<button id="btn2">Show HTML</button>
</body>
</html>
```

* val(): Trả ra giá trị của thành phần được chọn.

VD:

```
<!DOCTYPE html>
<html>
<head>
<script src="//ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js">
</script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    alert("Value: " + $("#test").val());
  });
});
</script>
</head>
<body>
<p>Name: <input type="text" id="test" value="Mickey Mouse"></p>
```

```
<button>Show Value</button>
</body>
</html>
```

2.3.1.2 Lấy ra thuộc tính:

* attr(): Phương thức giúp lấy ra nội dung thuộc tính của thành phần được chọn:

VD:

```
<!DOCTYPE html>
<html>
<head>
<script src="//ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js">
</script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    alert("$("#test").val(type));
  });
});
</script>
/head>
<body>
<p>Name: <input type="text" id="test" value="Mickey Mouse"></p>
<button>Show</button>
</body>
</html>
```

2.3.2. jQuery Set

jQuery cung cấp phương thức thêm vào một mội dung cho một đoạn văn bản (có thể đi kèm các ký thự cho HTML), đặt lại giá trị cho một thành phần nào đó trong HTML. Ngoài ra ta cũng có thể sử dụng thêm các hàm để tùy biến với mục đích sử dụng.

```
<!DOCTYPE html>
<html>
<head>
<script src="//ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js">
</script>
<script>
$(document).ready(function(){
  $("#btn1").click(function(){
    $("#test1").text("Hello world!");
  });
  $("#btn2").click(function(){
    $("#test2").html("<b>Hello world!</b>");
  });
  $("#btn3").click(function(){
    $("#test3").val("Dolly Duck");
  });
});
</script>
</head>
<body>
<p id="test1">This is a paragraph.</p>
<p id="test2">This is another paragraph.</p>
<p>Input field: <input type="text" id="test3" value="Mickey
Mouse"></p> <button id="btn1">Set Text</button> <button
id="btn2">Set HTML</button>
<button id="btn3">Set Value</button>
</body>
</html>
```

2.3.3. jQuery giúp người dùng thao tác với các thành phần của HTML

jQuery cung cấp các phương thức giúp ta có thể dễ dàng thêm hoặc bớt một nội dung, hoặc một thành phần trong các thẻ HTML

Một số hàm được cung cấp bởi thư viện jQuery như sau:

* append(): Chèn thêm một nội dung ở cuối của thành phần được chọn: *$("p").append("Some appended text.");*

* prepend(): Chèn thêm nội dung ở phía đầu phần tử được chọn *$("p").append("Some appended text.");*

* after(): Chèn thêm một nội dung ở phía trước của thành phần được chọn: *$("p").after("Some appended text.");*

* before(): Chèn thêm nội dung ở phía sau phần tử được chọn *$("p").before("Some appended text.");*

* remove(): Hàm giúp xóa đi các thẻ, sử dụng tùy theo mục đích người dùng *$("#div1").remove();*

* empty(): giúp loại bỏ các thành phần con của phần tử được lựa chọn *$("#div1").empty();*

2.3.4. jQuery giúp người dùng có thể thao tác, tác động tới CSS

jQuery cung cấp các phương thức giúp người dùng có thể thêm hoặc bớt các class CSS cho thẻ HMTL, hoặc thêm hay bớt một thuộc tính CSS cho một Class. Có một số phương thức như sau:

*addClass(): Thêm một class CSS .

```
$("button").click(function(){
  $("h1,h2,p").addClass("blue");
  $("div").addClass("important");
});
```

*removeClass(): Loại bỏ class khỏi đối tượng được chọn.

```
$("button").click(function(){
  $("h1,h2,p").removeClass("blue");
});
```

* toggleClass() : Phương thức giúp chuyển đổi giữa 2 thao tác add và remove *$("button").click(function(){*

$("h1,h2,p").toggleClass("blue

"); });

* css() : Trả về, hoặc cài đặt thêm thuộc tính CSS()

$("p").css("background-color"); <trả về thuộc tính của css>

$("p").css("propertyname","value"); <cài đặt thuộc tính cho css>

2.4. jQueryAjax

2.4.1. Giới thiệu

AJAX là kĩ thuật phát triển web có tính tương tác cao dựa trên cách kết hợp các công nghệ phát triển web với nhau: VD: HTML (hoặc XHTML) với Css, mô hình DOM thông qua JavaScript…, nó giúp trao đổi dữ liệu với server và cập nhật lại từng phần của trang web mà không phải tải lại toàn bộ trang web.

jQuery cung cấp một vài phương thức để thực hiện các chức năng cho Ajax. Với các phương thức này, có thể yêu cầu một đoạn văn bản, HTML, XML hoặc JSON từ máy Server cùng với việc sử dụng HTTP Get và HTTP Post và cũng có thể tại dữ liệu trực tiếp vào một thành phần của HMTL đã được chọn trên trang Web.

Việc viết code Ajax có thể đòi hỏi sự cẩn thạn và khéo léo, vì với mỗi trình duyệt khác nhau thì lại có các cú pháp Ajax khác nhau. Điều này cũng đồng nghĩa với việc ta sẽ phải viết code nhiều hơn bình thường để kiểm tra với những trình duyệt khác nhau, tuy nhiên thì các nhà phát triển jQuery đã lưu ý trước những điều này, do đó ta có thể viết những hàm Ajax với một vài dòng code

2.4.2. Các phương thức jQuery Ajax

2.4.2.1. jQuery load()

- Đây là phương thức rất đơn giản tuy nhiên nó lại là phương thức Ajax có sức ảnh hưởng rất lớn. Nó sẽ tải dữ liệu từ server về và đặt vào các yếu tố được chọn.
- Cú pháp:

$(selector).load(URL,data,callback);

URL : chỉ định vị trí đường dẫn đối tượng được load
data : chỉ định chuỗi truy vấn, giá trị hoặc từ khóa được gửi cùng khi gửi request
callback: tên hàm được thực hiện sau khi thao tác load được hoàn thành

- Ví dụ : ta có một file text có tên : “test.txt”, ví dụ sẽ thực hiện việc load nội dung từ đoạn văn bản đó vào một thẻ div được lựa chọn

```
<!DOCTYPE
html> <html>
<head>
<script src="//ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js">
</script>
<script>
$(document).ready(function(
 ){
```

```
$("button").click(function()
{
  $("#div1").load("demo_test.txt",function(responseTxt,statusTxt,xh
  r){ if(statusTxt=="success")
    alert("External content loaded
  successfully!"); if(statusTxt=="error")
    alert("Error: "+xhr.status+":
        "+xhr.statusText); });
            });
        });
</head>
<body>
<div id="div1"><h2>Let jQuery AJAX Change This
Text</h2></div> <button>Get External Content</button>
</body>
</html>
```

2.4.2.2 jQuery get() và jQuery post()

jQuery cung cấp phương thức yêu cầu lấy dữ liệu từ máy server cùng với HTTP GET hoặc POST

* get():

-Yêu cầu dữ liệu từ máy server cùng với giao thức HTTP GET

-Cú pháp:

$.get(URL,callback);

URL: Tham số quy định cụ thể địa chỉ mà người dùng yêu cầu.

callback: hàm được gọi khi phương thức get được thực hiện xong

Ví Dụ:

Ta có file “test.php” có chứa nội dung:

```
<?php echo ”DHDL HaiPhong” ?>
```

Ví dụ như sau:

```
<!DOCTYPE html>
<html>
<head>
<script src="//ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js">
</script>
<script>
```

```
$(document).ready(function(){
  $("button").click(function(){
    $.get("test.php",function(data,status){
      alert("Data: " + data + "\nStatus: " +
      status);
    });
  });
});
</script>
</head>
<body>
<button>Send an HTTP GET request to a page and get the
result
back</button>
</body>
</html>
```

>

*post():

- Yêu cầu dữ liệu lấy từ server theo giao thức HTTP POST

- Cú pháp:

$.post(URL,data,callback);

URL : chỉ định vị trí đường dẫn đối tượng được load

data: dữ liệu được gửi cùng với yêu cầu.

callback: hàm được gọi sau khi thao tác load được thực hiện xong.

VD: Ta có file “test.php”, có nội dung:

```
<?php
  echo 'Name: ' . $_POST["name"]."\n";
   echo 'City: '. $_POST["city"];
?>
```

Ta sẽ dùng jQuery để gửi dữ liệu bằng phương thức post

```
<!DOCTYPE html>

<html>
<head>

<script src="//ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js">
</script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $.post("test.php",
    {
```

```
    name:"Donald Duck",
    city:"Duckburg"
   },
   function(data,status){
    alert("Data: " + data + "\nStatus: " + status);
  });
 });|

});

</script>
</head>
<body>
<button>Send an HTTP POST request to a page and get the
            result
back</button>
</body>
</html>
```

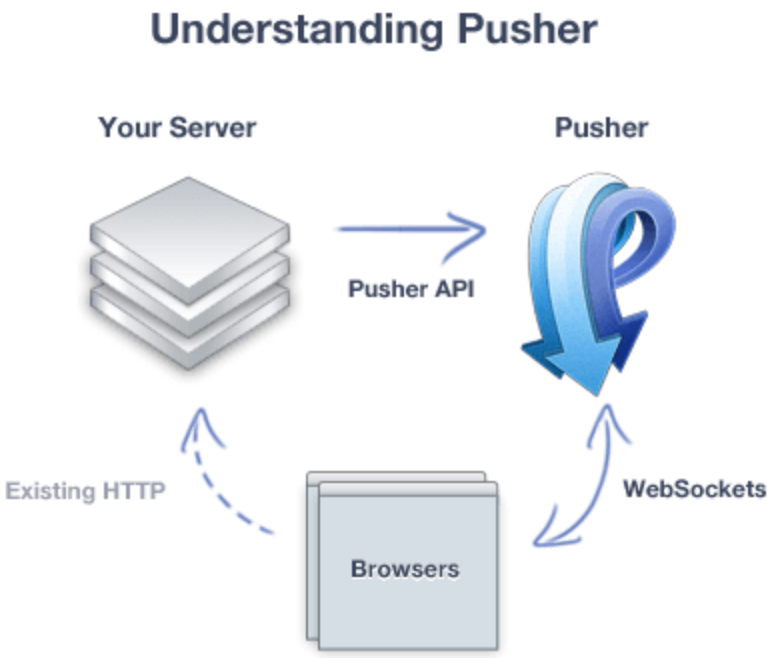
2.4. Pusher

2.4.1 Pusher là gì?

Pusher là một dịch vụ cloud, tạo ra một server trung gian giúp chúng ta có thể xử lý các tác vụ thời gian thực. Dữ liệu được gửi tới pusher, và pusher lại gửi nó đi tới các client đã subscribe (đăng ký) và các channel. Trong đó Pusher Channel cung cấp giao tiếp thời gian thực giữa các máy chủ, ứng dụng và thiết bị. Các kênh được sử dụng cho các biểu đồ thời gian thực, danh sách người dùng thời gian thực, bản đồ thời gian thực, chơi trò chơi nhiều người chơi và nhiều loại cập nhật giao diện người dùng khác. Nó có một thư viện hỗ trợ mọi thứ như trình duyệt web, ứng dụng iOS và Android, khung PHP, chức năng đám mây, tập lệnh bash, thiết bị IoT. Pusher Channel hoạt động ở mọi nơi vì nó sử dụng WebSockets và HTTP và cung cấp dự phòng cho các thiết bị không hỗ trợ WebSockets.

2.4.2 Quy trình hoạt động của pusher

Quy trình hoạt động của pusher được thể hiện qua sơ đồ sau:



Hình 8. Quy trình hoạt động của pusher

Client duyệt web - > dữ liệu sẽ được chuyển đến sever-> sever chuyển tiếp đến Pusher thông qua pusher API -> Pusher trả kết lại cho client (hoặc client khác).

Ví dụ: Lắng nghe sự kiện real-time trên Font-end

```
<!DOCTYPE html>
<head>
  <title>Pusher Test</title>
  <script src="https://js.pusher.com/5.0/pusher.min.js"></script>
  <script>
    // Enable pusher logging - don't include this in production
    Pusher.logToConsole = true;
    var pusher = new Pusher('app-key', {
      cluster: 'ap1',
      forceTLS: true
```

```
    });
    var channel = pusher.subscribe('my-channel');
    channel.bind('my-event', function(data) {
      alert(JSON.stringify(data));
    });
  </script>
</head>
<body>
  <h1>Pusher Test</h1>
  <p>
    Try publishing an event to channel <code>my-channel</code>
    with event name <code>my-event</code>.
  </p>
</body>
```

2.5. Socket.io

2.5.1 Socket.io là gì?

Socket.io là một module trong Node.js được phát triển vào năm 2010. Nó được phát triển để sử dụng các kết nối mở để tạo điều kiện giao tiếp thời gian thực, trả về giá trị thực ở tại thời điểm đó. Socket.io cho phép giao tiếp hai chiều giữa máy khách và máy chủ. Giao tiếp hai chiều được bật khi máy khách có Socket.io trong trình duyệt và máy chủ cũng đã tích hợp gói Socket.io

Nó được sử dụng trong việc xây dựng các ứng dụng web real-time cần tốc độ phản hồi ngay lập tức như: chat, trực tiếp bóng đá,.... Socket.io xây dựng dựa trên Engine.IO, đầu tiên nó sẽ thiết lập một kết nối long-polling, sau đó cố gắng nâng cấp lên các kết nối khác tốt hơn giống như WebSocket.

Ngoài Socket.io chúng ta còn có một vài kết nối khác như:

Trong long-polling, client sẽ gửi yêu cầu giống AJAX đến máy chủ. Với mỗi lần nhận được yêu cầu, máy chủ sẽ gửi phản hồi lại nếu & khi có cập nhật mới. Tại đây, clients sẽ liên tục & định kỳ yêu cầu cập nhật từ máy chủ, thông qua các kết nối TCP riêng biệt, làm tắc nghẽn lưu lượng mạng.

Trong short-polling, clients định kỳ gửi yêu cầu đến máy chủ để hỏi xem có gì mới không. Máy chủ không đợi, nhưng gửi lại nếu có cập nhật hoặc chỉ có tin nhắn trống. Ở đây, mạng thậm chí còn tắc nghẽn hơn với các yêu cầu liên tục này, ngay cả khi không có bản cập nhật.

Trong WebSockets, sẽ luôn có một kết nối TCP giữa clients và server. Có luồng dữ liệu hai chiều giữa clients và server cũng như tính chất thời gian thực do luôn kết nối TCP mở. Trong các phương thức, có tiềm năng rất lớn để tăng tốc độ trong WebSockets. Dung lượng phần header của giao thức HTTP là 100 byte, trong khi phần header của socket chỉ là 2 byte. Vì vậy, sau khi sử dụng HTTP ban đầu, Sockets có thể giao tiếp với tài nguyên ít hơn nhiều. Với nhiều số lượng yêu cầu được gửi đến thì nó cũng sẻ làm tăng thời gian phản hồi từ server tới clients.

Socket.io không phải là phát triển dựa trên WebSocket. Mặc dù Socket.io thực sự sử dụng WebSocket như một cách để giao tiếp trong một vài trường hợp, Socket.io sẽ bổ sung một số siêu dữ liệu cho mỗi gói: loại gói, không gian tên và id gói khi cần xác nhận thông báo. Đó là lý do tại sao máy khách WebSocket sẽ không thể kết nối thành công với máy chủ Socket.io và máy khách Socket.io cũng sẽ không thể kết nối với máy chủ WebSocket.

2.5.2 Cài đặt Socket.io

Để cài đặt Socket.io trong dự án của mình ban cần phải cài đặt ở 2 phía đó là server và client. Socket.io sẽ đảm nhận kết nối giữa 2 phía, thông thường các API của 2 phia sẽ tương tự giống nhau.

2.5.2.1 Socket.io trên server

Đối với server Node.js ta chỉ cần dùng npm đê cài đặt package có tên socket.io, truy cập vào thư mục dự án và mở terminal :

npm install --save socket.io

2.5.2.2 Socket.io trên client

Một bản dựng độc lập của clients được hiển thị theo mặc định bởi server tại /socket.io/socket.io.js

Ngoài ra, ta có thể cài đặt import thư viện này ở cdnjs.com , hoặc cài đặt thành các gói như webpack hoặc browserify bằng cách dùng npm:

npm install --save socket.io-client

2.5.3 Cấu trúc ứng dụng realtime sử dụng socket

Cấu trúc một ứng dụng realtime sử dụng socket bao gồm 2 phần: phía server, phía client.

2.5.3.1 Phía server

Đây là nơi sẽ cài đặt socket io. Ngôn ngữ để dựng server có thể là php,asp.net, nodejs,... Tuy nhiên, tùy vào ngôn ngữ lựa chọn mà cách cấu trúc server khác nhau. Ở đây, nếu được thì khuyến khích sử dụng nodejs để dựng server, vì như vậy có thể cài trực tiếp socketio vào cùng một server. Nếu sử dụng php thì phải cài thêm những package khác, hoặc phải chuẩn bị riêng server để chạy socketio.

Code khai báo sử dụng socketio ở server:

```
// build server, khai báo sử dụng socket io
var express = require("express");
var app = express();
app.use(express.static("public"));
app.set("view engine", "ejs");
app.set("views", "./views");

var server = require("http").Server(app);
var io = require("socket.io")(server);
server.listen(3000);
```

2.5.2.2 Phía client

Ở phía client sẽ xây dựng giao diện người dùng. Ở đây có thể sử dụng js, hoặc các thư viện của js như jquery,... Nói chung là ngôn ngữ gì cũng được.

Code khai báo sử dụng socketio ở phía client:

```
<html>
	<head>
		<title>Demo Socketio - Homepage</title>
		<script src="jquery.js"></script>
		<script src="socket.io/socket.io.js"></script>

		<script>
			var socket = io("http://localhost:3000");
		</script>
	</head>

	<body>
	</body>
</html>
```

Chương 3: Một số ứng dụng thời gian thực trong PHP

Như chúng ta đều đã biết thì realtime (thời gian thực) là một cụm từ không có gì xa lạ đối với các website nữa, nhất là trong thời đại công nghệ web phát triển như vũ bão hiện nay. Realtime ám chỉ răng việc phần mềm (ở đây mình nói tới chủ yếu là website) có thể phản hồi và tương tác người dùng một cách tức thì mà người dùng không cần chờ đợi lâu hoặc refresh lại ứng dụng hoặc trình duyệt. Chúng ta có thể nhìn thấy realtime ở khắp mọi nơi: thực tế nhất chính là qua các ứng dụng nhắn tin, hoặc bảng tin newsfeed trên Facebook.

Bạn có một website bán hàng? Bạn muốn thông báo về tình trạng đơn hàng cho khách hàng, hay xây dựng chương trình nhắn tin từ quản trị đến thành viên như thông báo khuyến mãi, biến động số dư....Hoặc bạn đang ấp ủ một website lấy kết quả xổ số,kết quả bóng đá theo thời gian thực, hoặc ứng dụng rất phổ biến là chát giữa các thanh viên với nhau,gửi các thông báo về máy. Thì lúc này bạn sẽ nghĩ đến realtime.Dưới đây là một số ứng dụng chat đơn giản có sử dụng kỹ thuật realtime.

3.1 Ứng dụng sử dụng jQuery

Chương trình ứng dụng web được xây dựng dựa trên nền ngôn ngữ PHP kết hợp với việc sử dụng jQuery. Bao gồm các chức năng :

-Đăng ký tài khoản,đăng nhập

-Đăng xuất.

-Trò chuyện (gửi và nhận tin nhắn ngay lập tức).

3.1.1. Xây dựng csdl cho ứng dụng:

3.1.1.1 Table users

```
CREATE TABLE `users` (
  `id_user` int(11) NOT NULL,
  `username` text NOT NULL,
  `password` text NOT NULL,
  `date_created` datetime NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
ALTER TABLE `users`
ADD PRIMARY KEY (`id_user`);
ALTER TABLE `users`
MODIFY `id_user` int(11) NOT NULL AUTO_INCREMENT;
```

Trong đó :

Field id_user chứa ID của người dùng, giá trị này sẽ tự động tăng.

Field username chứa tên đăng nhập của người dùng.

Field password chứa mật khẩu của người dùng.

Field date_created chứa thời gian mà người dùng đăng ký tài khoản.

	#	Tên	Kiểu	Bảng mã đối chiếu	Thuộc tính	Null	Mặc định	Ghi chú	Thêm	Hành động
☐	1	id_user	int(11)			Không	*Không*		AUTO_INCREMENT	Thay đổi Xóa Thêm
☐	2	username	text	utf8_general_ci		Không	*Không*			Thay đổi Xóa Thêm
☐	3	password	text	utf8_general_ci		Không	*Không*			Thay đổi Xóa Thêm
☐	4	date_created	datetime			Không	*Không*			Thay đổi Xóa Thêm

Theo dõi bảng *Lưu mục đã chọn* Duyệt Thay đổi Xóa Chính Duy nhất Chỉ mục Toàn văn

Hình 9. Table users

3.1.1.2 Table messages

```
CREATE TABLE `messages` (
  `id_msg` int(11) NOT NULL,
  `body` longtext NOT NULL,
  `user_from` text NOT NULL,
  `date_sent` datetime NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
ALTER TABLE `messages`
ADD PRIMARY KEY (`id_msg`);
ALTER TABLE `messages`
MODIFY `id_msg` int(11) NOT NULL AUTO_INCREMENT;
```

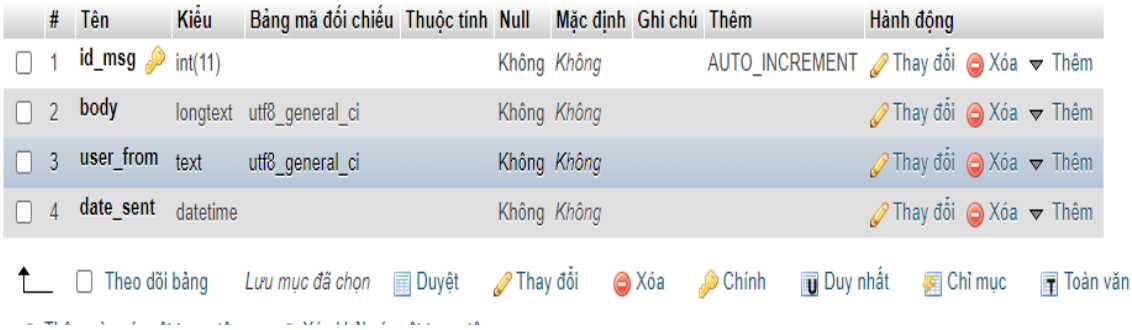
Trong đó :

Field id_msg chứa ID của tin nhắn, giá trị này sẽ tự tăng.

Field body chứa nội dung tin nhắn.

Field user_from chứa tên người gửi tin nhắn.

Field date_sent chứa thời gian gửi tin nhắn.



#	Tên	Kiểu	Bảng mã đối chiếu	Thuộc tính	Null	Mặc định	Ghi chú	Thêm	Hành động
1	id_msg	int(11)			Không	Không		AUTO_INCREMENT	Thay đổi Xóa Thêm
2	body	longtext	utf8_general_ci		Không	Không			Thay đổi Xóa Thêm
3	user_from	text	utf8_general_ci		Không	Không			Thay đổi Xóa Thêm
4	date_sent	datetime			Không	Không			Thay đổi Xóa Thêm

Theo dõi bảng Lưu mục đã chọn Duyệt Thay đổi Xóa Chính Duy nhất Chỉ mục Toàn văn

Hình 10. Table messages

3.1.3 Ứng dụng Chat Ajax - Đăng ký - đăng nhập - đăng xuất

3.1.3.1 Thiếu kế menu:

```
<!DOCTYPE html>
<html lang="en">
   <head>
      <meta charset="UTF-8">
      <title>Messenger</title>
      <meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no, minimum-scale=1.0, maximum-scale=1.0"/>
      <!-- Kết nối thư viện Font Awesome Icons -->
      <link rel="stylesheet" href="css/font-awesome-4.6.3/css/font-awesome.min.css">
      <!-- Kết nối file css/style.css -->
      <link rel="stylesheet" href="css/style.css">
   </head>
   <body>
      <?php
      // Nếu tồn tại $user
      if ($user) {
         // Hiển thị menu
         echo '<div class="main-menu">
                  <h1><i class="fa fa-commenting"></i> Messenger</h1>
                  <a href="logout.php"><i class="fa fa-sign-out"></i></a>
               </div>
               <div class="clearfix"></div>';
      }
      // Ngược lại
      else {
         // Hiển thị navbar
         echo '  <div class="main-navbar">
                  <h1><i class="fa fa-commenting"></i> Messenger</h1>
               </div><!-- div.main-header -->';
```

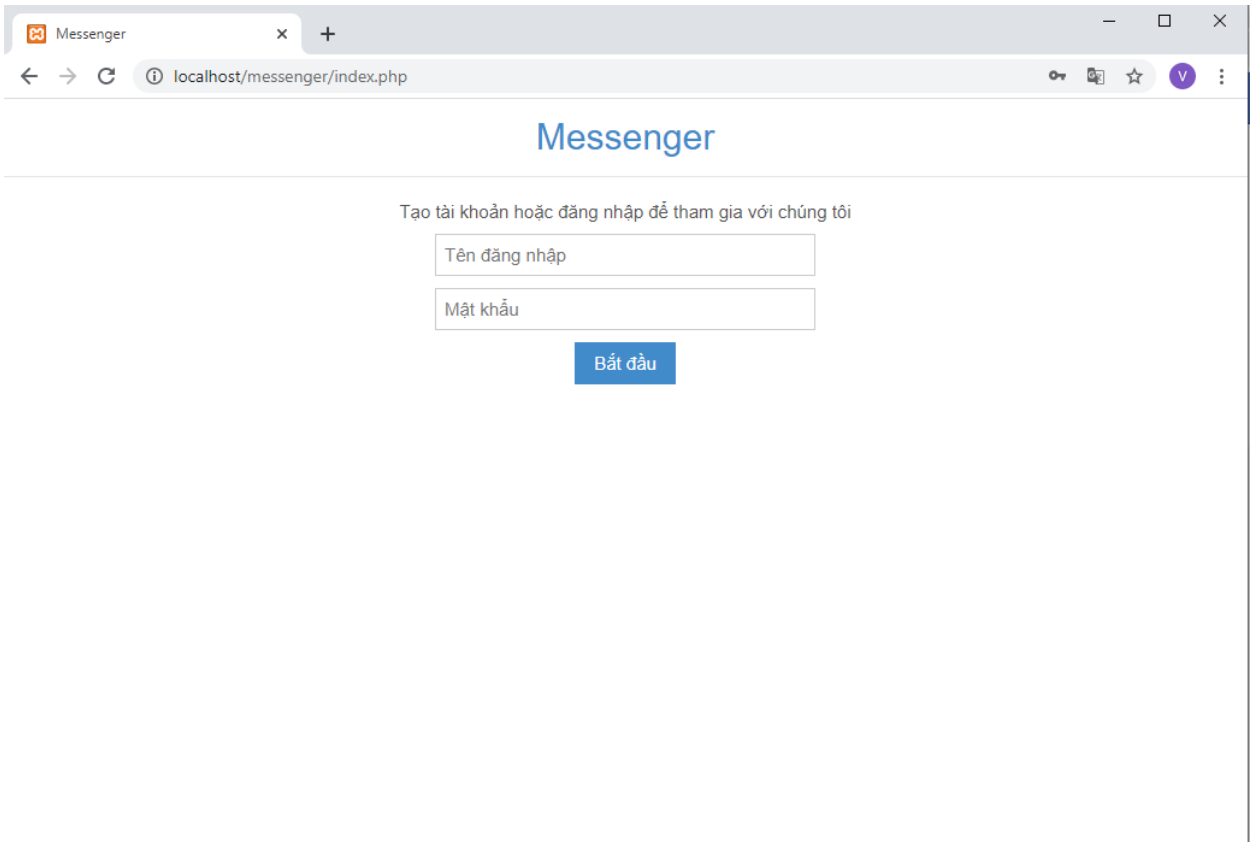
```
    }
    ?>
```

3.1.3.2 xây dựng giao diện trò chuyện

```
<?php

// Kết nối database
include('includes/general.php');
// Kết nối header
include('includes/header.php');

// Nếu tồn tại $user
if ($user) {
   // Hiển thị trò chuyện
   echo '<div class="main-chat">
         </div><!-- div.main-chat -->
         <div class="box-chat">
               <form method="POST" id="formSendMsg" onsubmit="return false;">
                     <input type="text" placeholder="Nhập nội dung tin nhắn ...">
               </form><!-- form#formSendMsg -->
         </div><!-- div.box-chat -->';
}
// Ngược lại
else {
   // Hiển thị form đăng nhập, đăng ký
   echo '<div class="box-join">
               <p>Tạo tài khoản hoặc đăng nhập để tham gia với chúng tôi</p>
               <form method="POST" id="formJoin" onsubmit="return false;">
                     <input type="text" placeholder="Tên đăng nhập" class="data-
input" id="username">
                     <input type="password" placeholder="Mật khẩu" class="data-
input" id="password">
                     <button class="btn-submit">Bắt đầu</button>
                     <div class="alert danger"></div>
               </form><!-- form#formJoin -->
         </div><!-- div.box-join -->';
}
// Kết nối footer
include('includes/footer.php');
?>
```



Hình 11. Giao diện ứng dụng

3.1.4.Gửi tin nhắn

3.1.4.1 Viết Ajax

```
// Hàm gửi tin nhắn
function sendMsg() {
    // Khai ba1oca1c biến trong form
    $body_msg = $('#formSendMsg input[type="text"]').val();

    // Gửi dữ liệu
    $.ajax({
        url: 'sendmsg.php', // đường dẫn file xử lý
        type: 'POST', // phương thức
        // dữ liệu
        data: {
            body_msg: $body_msg
                // thực thi khi gửi dữ liệu thành công
        }, success: function () {
```

```
            $('#formSendMsg input[type="text"]').val(''); // làm trống thanh trò chuyện
        }
    });
}

// Bắt sự kiện gõ phím enter trong thanh trò chuyện
$('#formSendMsg input[type="text"]').keypress(function () {
    var keycode = (event.keyCode ? event.keyCode : event.which);
    if (keycode == '13') {
        // Chạy hàm gửi tin nhắn
        sendMsg();
    }
});

// Bắt sự kiện click vào thanh trò chuyện
$('#formSendMsg input[type="text"]').click(function (e) {
    // Kéo hết thanh cuộn trình duyệt đến cuối
    window.scrollBy(0, $(document).height());
});
```

3.1.4.2 Viết PHP để xử lý dữ liệu

```
<?php

// Kết nối database, lấy dữ liệu chung
include('includes/general.php');

// Khai báo các biến gán với dữ liệu nhận được
$body_msg = @mysqli_real_escape_string($cn, $_POST['body_msg']);
// Xử lý chuỗi $body_msg
$body_msg = htmlentities($body_msg);
$body_msg = trim($body_msg);

// Nếu $body_msg khác rỗng
if ($body_msg != '') {
   // Thực thi gửi tin nhắn
   $query_send_msg = mysqli_query($cn, "INSERT INTO messages VALUES (
            '',
            '$body_msg',
            '$user',
            '$date_current'
         )");
}
?>
```

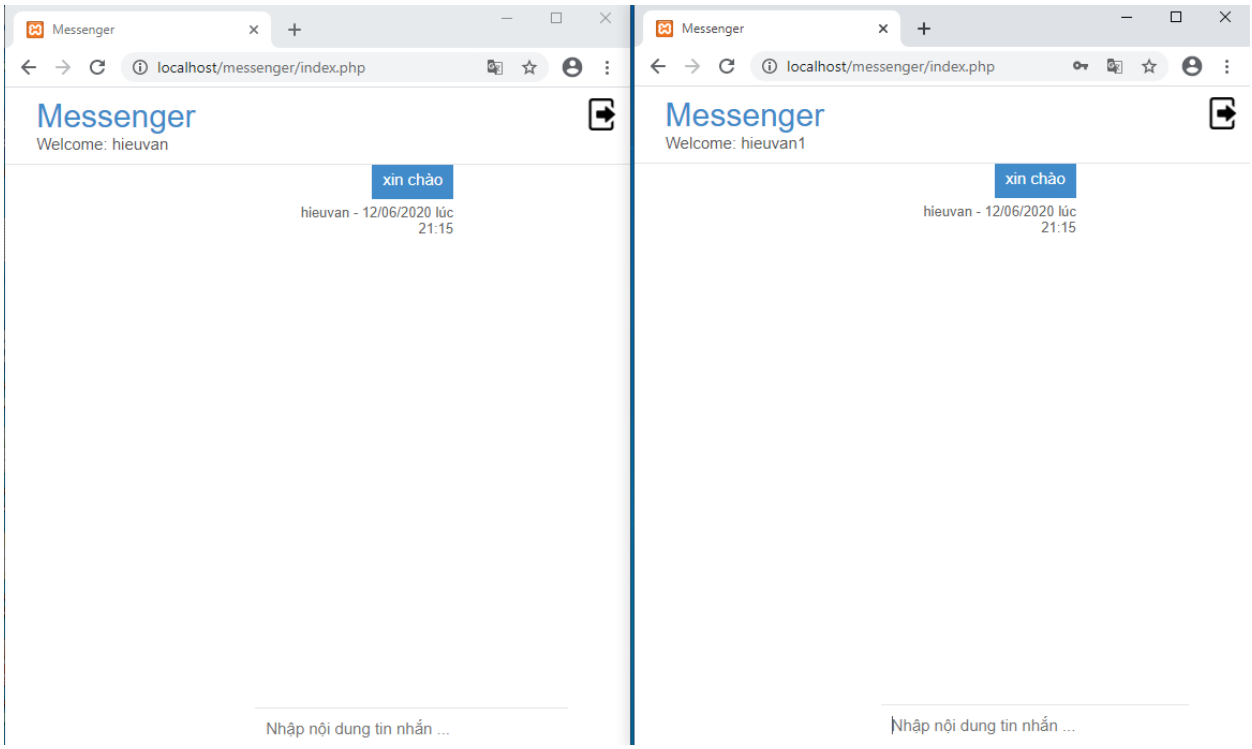
3.1.5 Thiết lập thời gian thực:

```
<?php

// Kết nối database, lấy dữ liệu chung
include('includes/general.php');

// Lấy dữ liệu từ table messages theo thứ tự id_msg tăng dần
$query_get_msg = mysqli_query($cn, "SELECT * FROM messages ORDER BY id_msg ASC");
// Dùng vòng lập while để lấy dữ liệu
while ($row = mysqli_fetch_assoc($query_get_msg)) {
    // Thời gian gửi tin nhắn
    $date_sent = $row['date_sent'];
    $day_sent = substr($date_sent, 8, 2); // Ngày gửi
    $month_sent = substr($date_sent, 5, 2); // Tháng gửi
    $year_sent = substr($date_sent, 0, 4); // Năm gửi
    $hour_sent = substr($date_sent, 11, 2); // Giờ gửi
    $min_sent = substr($date_sent, 14, 2); // Phút gửi
    // Nếu người gửi là $user thì hiển thị khung tin nhắn màu xanh
    if ($row['user_from'] == $user) {
        echo '<div class="msg-user">
                <p>' . $row['body'] . '</p>
                <div class="info-msg-user">
                    Bạn - ' . $day_sent . '/' . $month_sent . '/' . $year_sent . ' lúc ' . $hour_sent . ':' . $min_sent . '
```

```
                </div>
            </div>';
    }
    // Ngược lại người gửi không phải là $user thì hiển thị khung tin nhắn màu xám
    else {
        echo '  <div class="msg">
                <p>' . $row['body'] . '</p>
                <div class="info-msg">
                    ' . $row['user_from'] . ' - ' . $day_sent . '/' .
$month_sent . '/' . $year_sent . ' lúc ' . $hour_sent . ':' . $min_sent .
'
                </div>
            </div>';
    }
}
?>
```

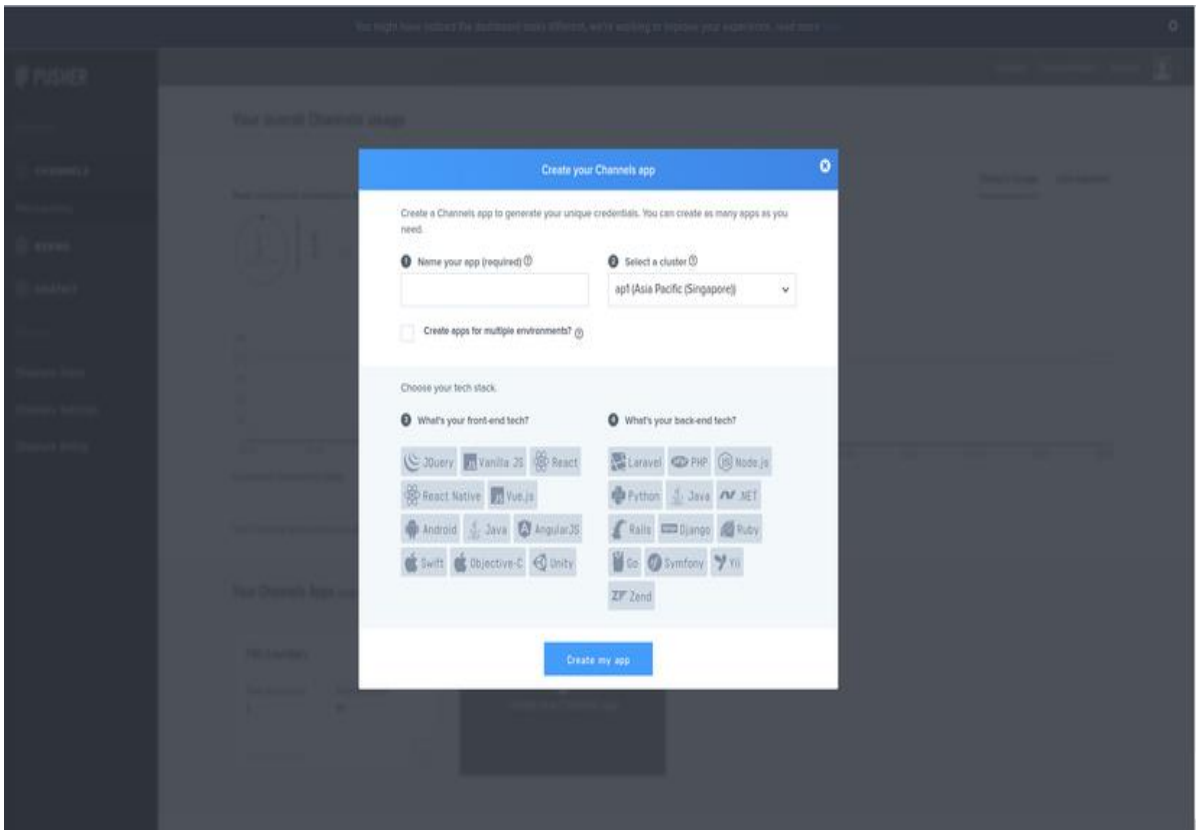


Hình 11. Ứng dụng chát

3.2 Ứng dụng sử dụng pusher

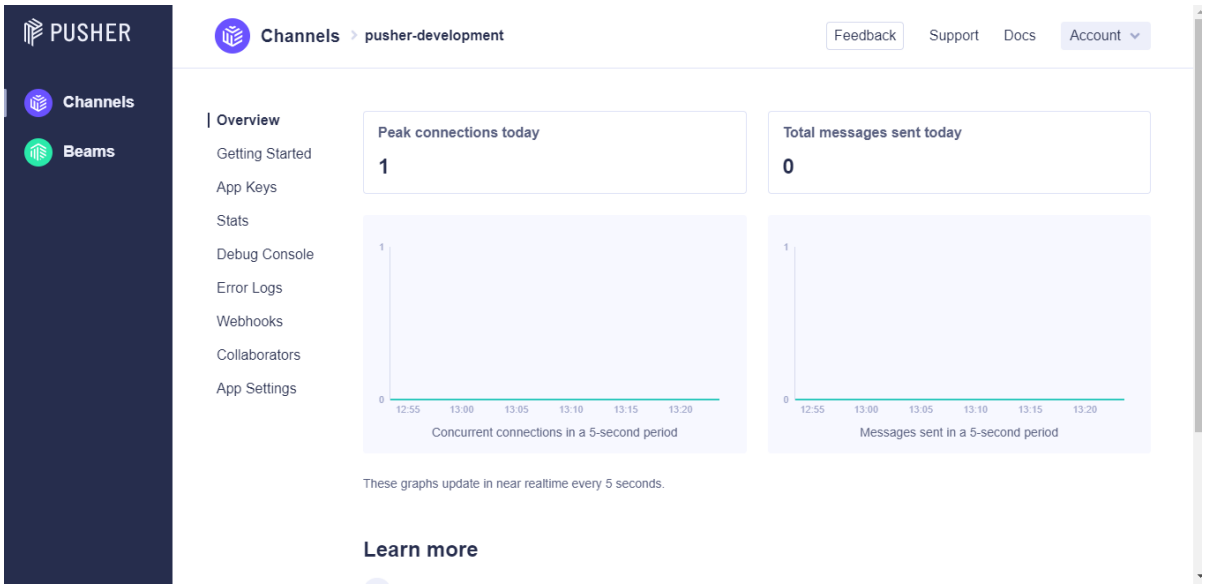
3.2.1 Tạo app Api

Đầu tiên để sử dụng Pusher, ta cần phải đăng nhập vào màn hình quản trị của Pusher và tiến hành khởi tạo thông tin về app của bạn:



Hình 12. Tạo App

Như trong hình chúng ta có thể thấy Pusher hỗ trợ trên Font-end và Back-end rất nhiều ngôn ngữ, bạn có thể thoải mái lựa chọn ngôn ngữ phù hợp với project mình đang dùng. Sau khi khởi tạo thành công app, chúng ta sẽ thấy một màn hình quản lý app của bạn như sau:



Hình 13. Quản lý App

Khi kéo xuống dưới sẽ thấy các thông tin app_id, app_secret, app_key, cluter. Đây là những thông tin để kết nối vào app

3.2.2 Đưa dữ liệu lên app api

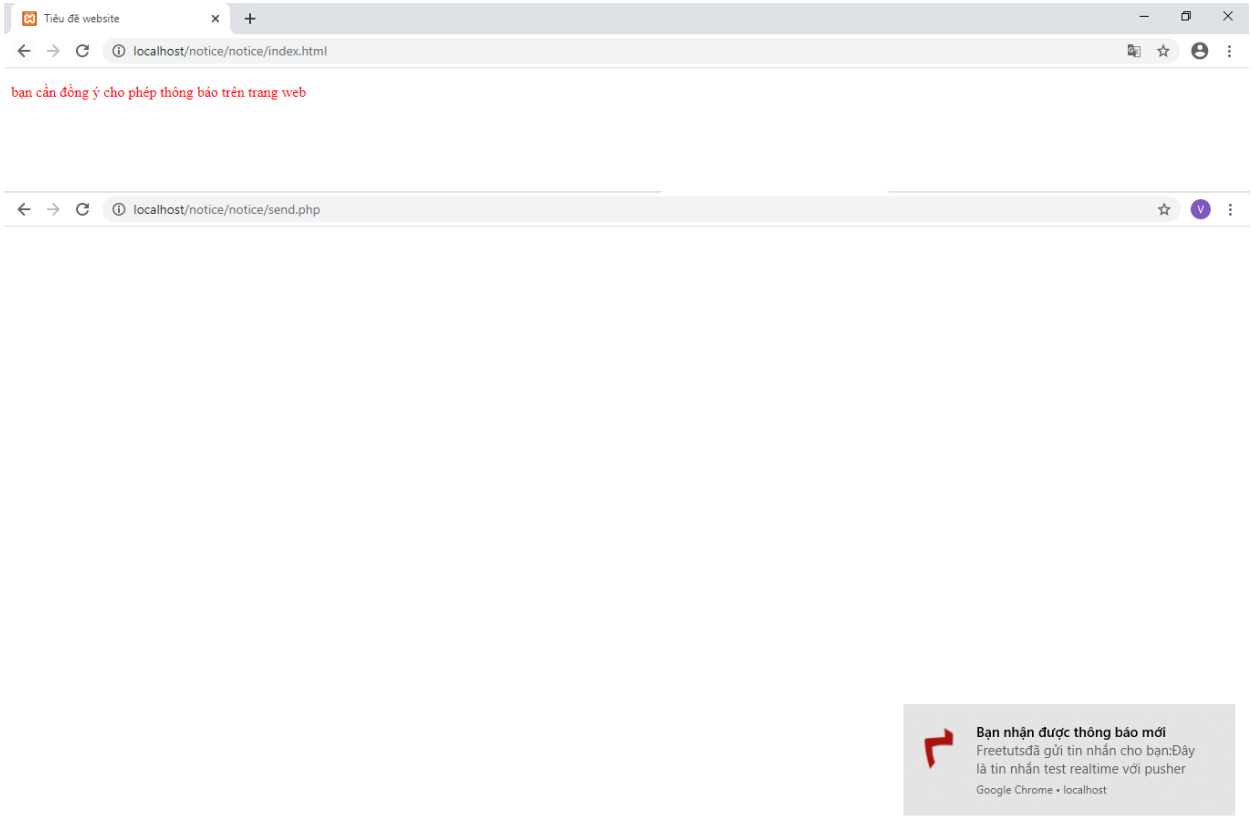
```
<?php

require('Pusher.php');
$options = array(
    'encrypted' => true
);
$pusher = new Pusher(
        '10d5ea7e7b632db09c72', 'a496a6f084ba9c65fffb', '234217', $options
);
$data['name'] = 'Freetuts.net';
$data['message'] = 'Đây là tin nhắn test realtime với pusher';
$pusher->trigger('Freetuts', 'notice', $data);
//Freetuts la ten kenh
//notice la su kien
//data la du lieu gui di
?>
```

3.2.3 Lấy dữ liệu từ api

```
<script type="text/javascript">
    Pusher.logToConsole = true;
    var pusher = new Pusher('10d5ea7e7b632db09c72', {
        encrypted: true
    });
```

```
    var channel = pusher.subscribe('Freetuts');
      channel.bind('notice', function (data) {
      //code xử lý khi có dữ liệu từ pusher
      n = new Notification(
            'Bạn nhận được thông báo mới',
            {
               body: data.name + 'đã gửi tin nhắn cho bạn:' + data.message,
               icon: 'https://freetuts.net/public/logo/icon.png', // Hình ảnh
               tag: 'https://freetuts.net/' // Đường dẫn
            });
      setTimeout(n.close.bind(n), 10000);
      // tự động đóng thông báo sau 10s
      n.onclick = function () {
         window.location.href = this.tag;
      }
      // kết thúc code xử lý thông báo
   });
</script>
```



Hình 14. Thông báo được gửi về

Khi ta ấn đồng ý thì lập tức sẽ có thông báo được gửi về máy

3.3.Ứng dụng sử dụng socket.io

3.3.1 Cài đặt socket.io

Ta cần phải cài đặt socket.io ở 2 phía đó là ở server và client:

3.3.1.1 Cài đặt socket.io trên server

Để cài đặt socket.io trong server ta tiến hành vào thư mục cần cài và mở terminal gõ dòng lệnh :
npm install ---save socket.io

3.3.1.2 Cài đặt socket.io trên client

Ta chỉ cần import thư viện socket.io vào clients, thư viện này sẽ được tạo tự động khi ta khởi tạo socket.io server:

<script src="/socket.io/socket.io.js"></script>

Ngoài socket.io ta cần phải cài đặt express, express cho phép chúng ta khởi tạo HTTP Server một cách đơn giản và dễ dàng hơn. Để cài đặt express bạn chỉ cần mở terminal lên và gõ dòng lệnh:

npm install ---save express

3.3.2. Xây dựng ứng dụng dù socket.io

Cấu trúc thư mục của chat app sẽ bao gồm:
chat-app
----app.js
----views/
---------index.html
Để chuẩn bị ta cần phải cài đặt một vài module như express, socket.io cho server bằng cách mở terminal và gõ dòng lệnh :
npm install ---save express socket.io

3.3.2.1 Khởi tạo HTTP Server kết hợp Socket.io:

Trong file app.js chúng ta sẽ khởi tạo HTTP Server bằng express và kết nối socket.io như sau :

```
var app = require("express")();
var http = require("http").createServer(app);
var io = require("socket.io")(http);

app.get("/", function(req, res) {
  res.sendFile(__dirname + "/views/index.html");
```

```
});

io.on("connection", function(socket) {
  //socket bên phía server ở đây
});

http.listen(3000, function() {
  console.log("listening on *:3000");
});
```

3.3.2.2 Xây dựng giao diện

Xây dựng một giao diện đơn giản, chúng ta sẽ vào thư mục views và tìm file index.html

```
<!doctype html>
<html>
  <head>
    <title>Socket.IO chat</title>
    <style>
      * { margin: 0; padding: 0; box-sizing: border-box; }
      body { font: 13px Helvetica, Arial; }
      form { background: #000; padding: 3px; position: fixed; bottom: 0; width: 100%;
}
      form input { border: 0; padding: 10px; width: 90%; margin-right: .5%; }
      form button { width: 9%; background: rgb(130, 224, 255); border: none; padding:
10px; }
      #messages { list-style-type: none; margin: 0; padding: 0; }
      #messages li { padding: 5px 10px; }
      #messages li:nth-child(odd) { background: #eee; }
      #messages { margin-bottom: 40px }
    </style>
  </head>
  <body>
    <ul id="messages"></ul>
    <form action="">
      <input id="m" autocomplete="off" /><button>Send</button>
    </form>
    <script src="/socket.io/socket.io.js"></script>
    <script src="https://code.jquery.com/jquery-1.11.1.js"></script>
    <script>
      //Chúng ta sẽ xử lý socket bên phía clients ở đây :))
    </script>
  </body>
</html>
```

3.3.2.3 Xử lý chat với Socket.io:

Tạo ra sự kiện có tên chat message khi người dùng click vào nút Send và bắt sự kiện trả về từ server :

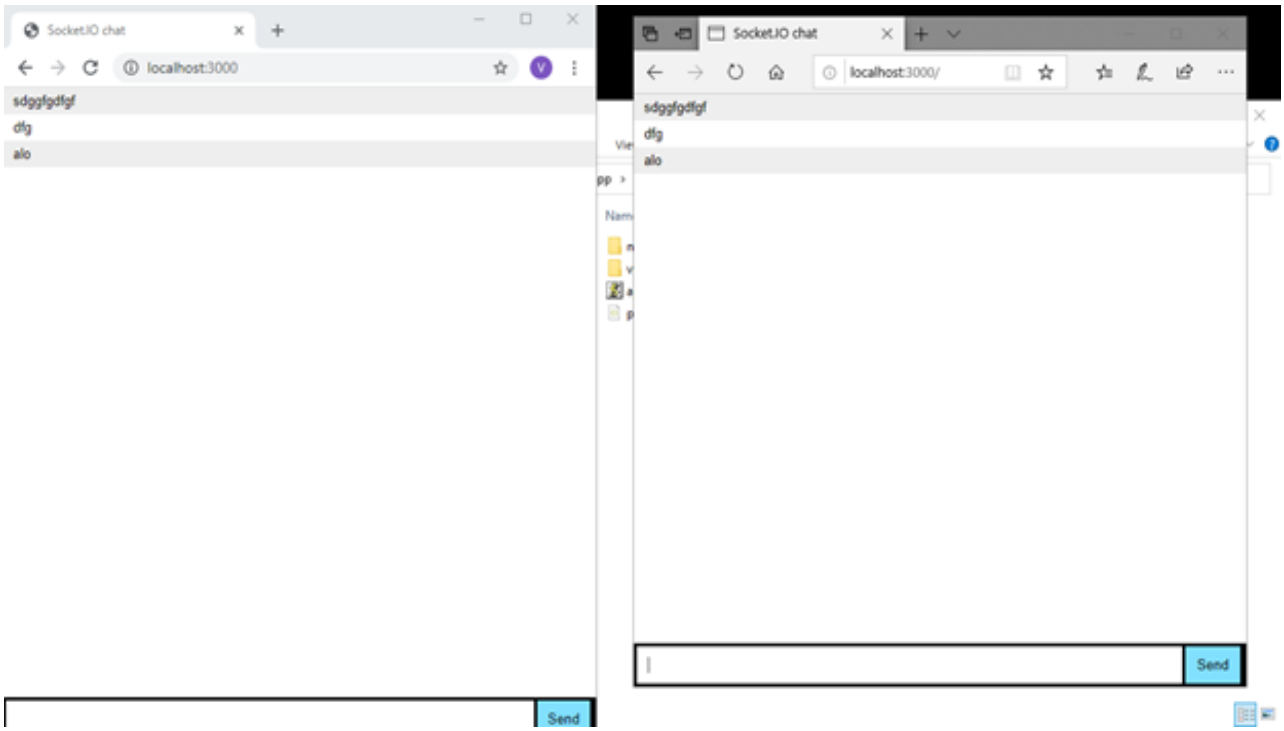
```
$(function() {
  var socket = io();
  $("form").submit(function() {
    //Tao sự kiện khi click vào nút send
    socket.emit("chat message", $("#m").val());
    $("#m").val("");
    return false;
  });

  //Băt sự kiện từ server
  socket.on("chat message", function(msg) {
    $("#messages").append($("<li>").text(msg));
    window.scrollTo(0, document.body.scrollHeight);
  });
});
```

Nhận sự lấy sự kiện chat message (bên phía server) và gửi trả về sự kiện chat message về cho clients:

```
io.on("connection", function(socket) {
  socket.on('chat message', function(msg){
    io.emit('chat message', msg);
  });
});
```

Tiến hành khởi chạy dự án bằng cách mở terminal và gõ dòng lệnh :
node app.js



Hình 15. Ứng dụng chát Socket

KẾT LUẬN

Đồ án đã tìm hiểu được các kiến thức cơ bản và chuyên sâu về một số kỹ thuật lập trình thời gian thực và PHP. PHP là một ngôn ngữ lập trình web phổ biến nhất vào thời điểm hiện tại, được sử dụng vào rất nhiều các trang web quản lý, tin tức, các cổng thông tin điện tử …

Việc kết hợp giữa các kỹ thuật lập trình thời gian thực và PHP được sử dụng trong rất nhiều các ứng dụng hiện nay bởi chúng không chỉ tạo ra những tiện ích cho người dùng mà còn cải thiện tốc độ xử lý cho website, tăng khả năng trải nghiệm người dùng, hiệu năng cho hệ thống.

Đồng thời, qua quá trình làm đồ án, em đã học thêm nhiều kiến thức thực tế và biết vận dụng kiến thức đã học để giải quyết một bài toán đặt ra. Tuy nhiên kết quả còn rất hạn chế, cần có sự hỗ trợ rất nhiều của thầy cô. Để có khả năng làm tốt việc vận dụng lý thuyết vào thực hành và có kỹ năng nhất định, em thấy cần phải thực hành nhiều hơn nữa.

Hướng phát triển tiếp theo của đồ án sẽ tiếp tục nghiên cứu sâu hơn về các kỹ thuật lập trình thời gian thực trong PHP, kết hợp thêm với việc thao tác với các hệ cơ sở dữ liệu như MySQL và SQLServer; Tìm hiểu thêm về sự kết hợp của PHP với các kỹ thuật lập trình thời gian thực khác như Slanger , pushcrew, ReactJs,…

TÀI LIỆU THAM KHẢO

[1] Tài liệu: *jQuery in Action* - Tác giả: Bear Bibeault và Yehuda Katz.

[2] https://www.w3schools.com/

[3] https://freetuts.net/