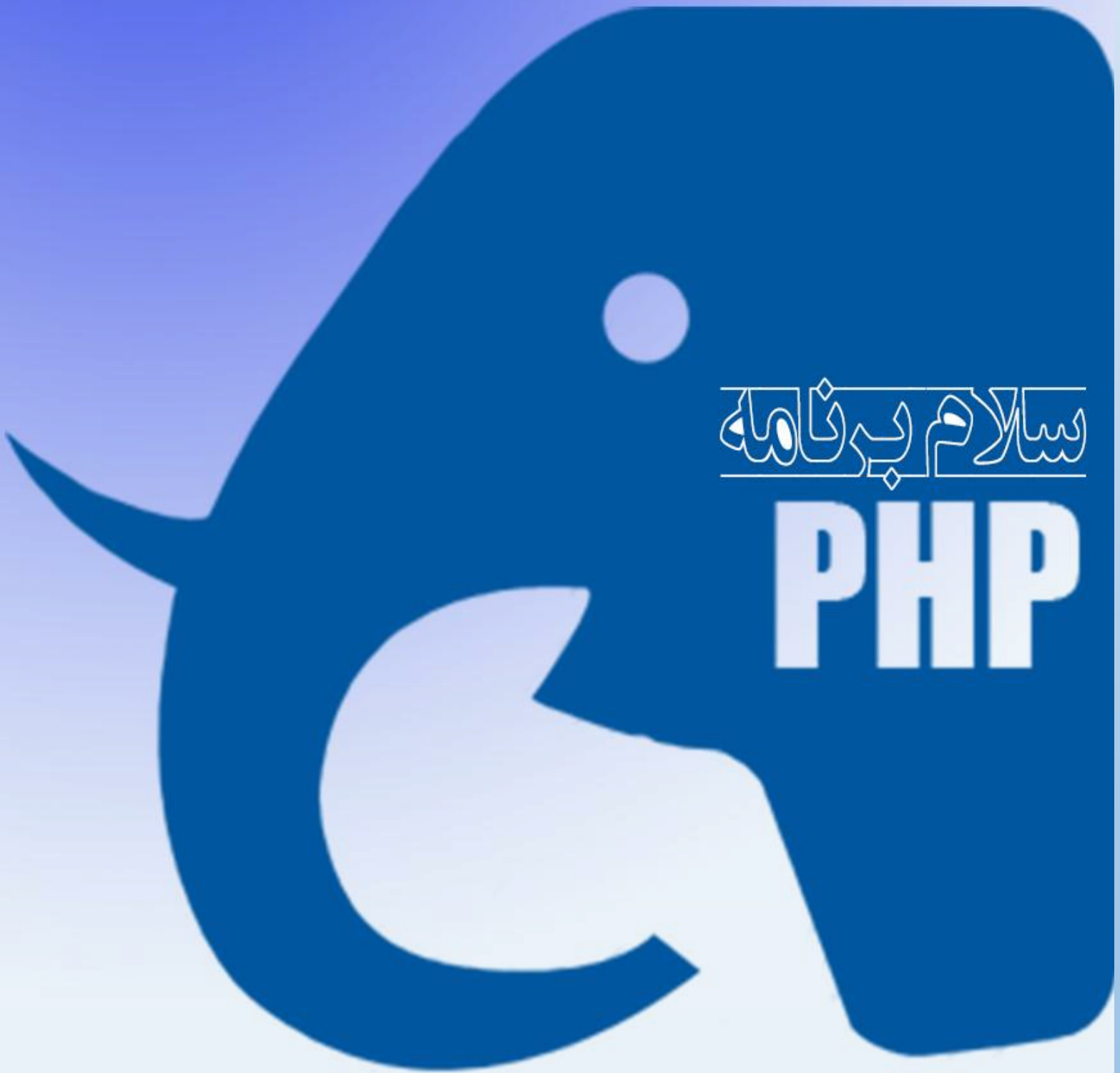
آموزش مقدماتی

PHP

HiProgram.ir



سلام برنامه

PHP

فهرست مطالب

سخن گرداورنده

سلام خدمت شما دوستان عزیز این کتاب آموزشی توسط تیم درسنامه نوشته و توسط تیم سلام برنامه و شخص امیر عباس سعدی گرداوری شده است تمامی مباحث مقدماتی زبان php در این کتاب آورده شده است از شما دوستان عزیز خواهشمندیم در صورت انتشار کتاب حتما منبع آن را قید کنید این کتاب توسط آدرس اینترنتی http://hiprogram.ir انتشار پیدا کرده است.

مقدمه

درود، به دوره آموزشی مقدماتی پی اچ پی (PHP) خوش آمدید!

در این دوره تلاش خواهیم کرد که برنامه نویسی را به زبانی ساده و همراه با لذت برای شما ارائه کنیم. شاید بعضی ها باشند که در ذهن یک غول از برنامه نویسی ساخته باشند اما اجازه بدهید که در اینجا اعلام کنیم این پیش فرض کاملا اشتباه است!

ساخت و تولید یک سیستم توسط زبان برنامه نویسی به هیچ عنوان سخت و صرفا برای آدم های تیزهوش نیست. هر کسی که اراده به یادگیری کند می تواد در مدت زمان مناسبی این کار شیرین و دوست داشتنی را فرا گرفته و به تولید سیستم های پویا اقدام نماید. پس ابتدایی ترین نوید ما به شما این خواهد بود که این کار را با هم و همراه لذت یاد خواهیم گرفت.

اگر با این دوره همراه شوید، ما تلاش خواهیم کرد که همراه هم قدم به قدم اصول ابتدایی پی اچ پی را یاد گرفته و پیاده کنیم. این قول را به شما می دهیم که پس از اتمام این دوره بدون اینکه خودتان متوجه شوید تبدیل به یک برنامه نویس خوب شوید. صد البته که این به تلاش و مهمتر از آن علاقه شما بستگی دارد.

در این دوره مفاهیم به صورت کاملا ساده و روشن، آهسته آهسته و همراه با پیاده سازی و گرفتن خروجی از آنها برای شما آموزش داده خواهند شد. **تنها خواهش ما این است که هیچگاه بدون اینکه درسی را کاملا یاد نگرفته اید به سراغ مبحث بعدی نروید.**

خیلی خب اگر آماده هستید به سراغ درس شیرین برنامه نویسی با زبان پی اچ پی برویم. امیدواریم که شما هم مثل ما مشتاق و آماده برای یک دوره همراه با لذت باشید.

تارنمای پویا چیست؟ (تارنما= وب سایت)
در گذشته اگر قرار بود یک شرکت، سازمان و یا هر کسی که اقدام به راه اندازی سیستم تارنما بر روی شبکه داخلی و یا اینترنت نماید، باید این کار را توسط زبان و یا دستورهای اچ تی ام ال (HTML) انجام می داد.

خب در آن زمان این کار انجام می شد اما مشکل این بود که برای هر صفحه ای باید تمام کدها دوباره نوشته می شد. مثلا اگر یک شرکت که دارای ۲۰۰ نوع محصول تولیدی بود، قرار بود تارنمایی راه اندازی کند، باید برای هر محصول یک برگه را اختصاص می داد و این یعنی ۲۰۰ بار کدنویسی یا حداقل رونویسی! تازه مشکل بزرگ تر بروزرسانی این صفحات بود که این کار باید توسط افرادی که با زبان برنامه نویسی آشنا بودند انجام می شد.

با پیدایش زبان هایی مانند پی اچ پی این قابلیت به تولید سیستم اضافه شد که از دوباره سازی این صفحات و کدها جلوگیری می کرد. حال همین شرکت خودمان با ۲۰۰ محصول کافی است که یک صفحه به عنوان الگو داشته باشد و تمام محصول ها را در همان یک برگه به نمایش بگذارد. جالب شد نه!

البته این کار به کمک ابزاری دیگر به نام پایگاه داده ها انجام می شود. پایگاه داده ها جایی است که اطلاعات شما به طور طبقه بندی شده در آن ثبت می شود و هر لحظه می توانید به آنها دسترسی داشته باشید. نگران این مورد هم نباشید چون در طول دوره آن را هم با لذت تمام یاد خواهیم گرفت.

خب در یک تعریف کوتاه باید بگوییم که سیستم پویا سیستمی است که به کمک پایگاه داده ها و توسط یک الگو برای یک صفحه می تواند کار ۱۰۰ یا ۲۰۰ و یا حتی بیشتر را انجام دهد.

یک سیستم پویا این قابلیت را دارد که با توجه به پارامترهای متفاوت، رفتار متفاوت از خود نشان دهد. اگر دقت کنید در سیستم هایی مانند تارنماهای خبری در طول یک روز بارها محتوای جدید به تارنما اضافه می شود. این دقیقا چیزی است که با کمک تارنمای پویا انجام می شود؛ یعنی یک الگو برای مثلا صفحه ابتدایی تارنما و داده هایی که به کمک پایگاه داده ها بر روی صفحه نقش می بندند.

از همه جالب تر این است که هر لحظه بدون هیچ نیازی به گسترش سیستم از نظر برنامه نویسی می توانید داده جدیدی را منتشر کنید. اگر کاملا متوجه موضوع نشدید اصلا ایرادی ندارد؛ این چیزی است که شما در طول دوره خود به خود فرا خواهید گرفت.

مفهوم سرور و کاربر CLIENT/SERVER

قبل از آشنایی با زبان برنامه نویسی لازم دانستیم که مفهوم سرور و کاربر را برای شما به طور خلاصه شرح دهیم.

به طور ساده این مفهوم وقتی پیدا می شود که در یک تعامل بین دو طرف یکی سرویس بدهد و دیگری سرویس بگیرد، همین :)

مثلا وقتی شما به فروشگاه محله خود برای خرید می روید، در این تعامل شما کسی هستید که تقاضای سرویس دارید یعنی کاربر، و یا همان مشتری خودمان و در آن طرف قضیه فروشنده، سرویس دهنده می شود. در دنیای رایانه به سیستمی که سرویس ارائه می کند سرور (Server) و به کسی که سرویس می گیرد کاربر (Client) می گویند. سرور همان رایانه خودمان است با این تفاوت که از قطعاتی قوی تر و سریع تر تشکیل شده است.

اما دلیل بیان این مطلب این است که در دنیای برنامه نویسی زبان های متفاوتی وجود دارند که ما براساس اینکه این زبان در طرف سرور کارایی دارد یا طرف کاربر، به آنها سمت کاربر و یا سمت سرور می گوییم. نمونه آن را بارها دیده اید؛ مثلا وقتی مرورگر خود را باز می کنید مرورگر شما ابتدا می لرزد و یا کلمه ای دنبال اشاره گر شما می دود. این دست کارها فقط در سمت کاربر انجام می گیرد و نیازی نیست که سرور را مشغول این کارها کنیم. از طرفی دیگر کارهایی مانند گرفتن داده ها و نمایش و تعامل با کاربر، کارهایی هستند که در سمت سرور انجام می شوند.

نکته مهم این است که زبان های سمت کاربر مانند HTML و JavaScript، وقتی در داخل رایانه کاربر فراخوانده شوند بدون نیاز سرور عملیاتی که در آنها نوشته شده را پیاده می کنند. اما زبان های سمت سرور باید در سرور نوشته و گذاشته شوند تا قابل اجرا باشند.

به طور خلاصه باید گفت که برای استفاده از زبان های سمت سرور نیاز به سرور و یا فراهم کردن این بستر داریم. مثلا اگر بخواهیم در رایانه های خانگی از زبان های سمت سرور استفاده کنیم باید بستر سرور را فراهم کنیم. این کار با بسته هایی مانند WampServer فراهم و به سادگی قابل پیاده سازی است.

WampServer برنامه ای است که بستر سرور را برای شما آماده کرده و به شما اجازه می دهد که مانند یک سرور از رایانه خود استفاده کنید.

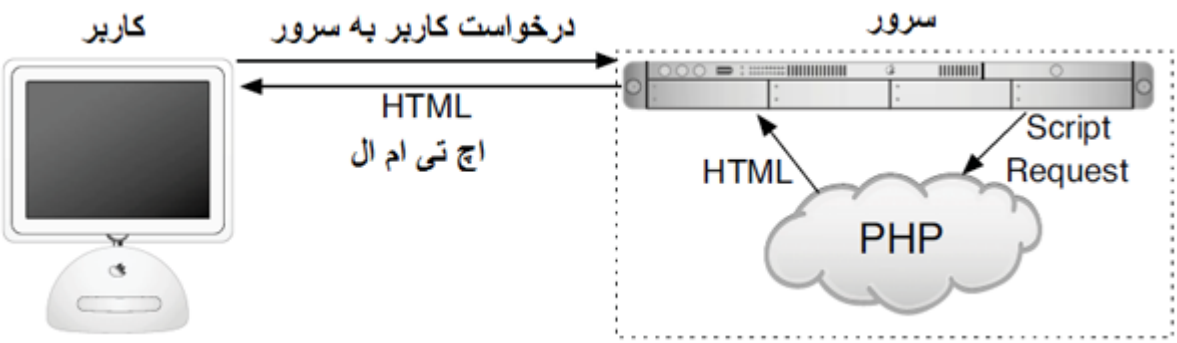
آشنایی با زبان برنامه نویسی پی اچ پی

پی اچ پی زبانی است برای تولید سیستم های پویا که در بستر اینترنت و که در داخل کدهای HTML کار می کنند. زبان پی اچ پی یک زبان سمت سرور است. بله، حالا خوب متوجه می شوید که منظورمان چیست.

منظور از این گفته این است که ما باید برای استفاده از این زبان در رایانه خانگی خود از بستر سرور به کمک WampServer استفاده کنیم. این زبان همیشه در حال بروزرسانی است، یعنی بوجود آورنده های پی اچ پی بعد از گذشت سال ها هنوز در فکر هر چه بهتر کردن این زبان هستند. این نکته یک عامل شده است که استفاده کننده های بسیاری به سمت پی اچ پی کشیده شده اند.

برای تولید سیستم های پویا، پی اچ پی یکی از بهترین زبان ها است چرا که سیستم ها با پی اچ پی معمولا سریع تر ، بهتر و راحت تر کار می کنند. از همه مهمتر ارتباط خوب پی اچ پی با پایگاه داده ها است. این زبان قابلیت ارتباط با انواع پایگاه داده ها را با کیفیت بالایی دارد.

اجازه بدهید کمی بیشتر در مورد سمت سرور بودن پی اچ پی توضیح دهیم. روال کار بدین ترتیب است که وقتی شما کدی را در تارنمایی با پی اچ پی می نویسید، این کد برای عمل به سرور متوسل می شود. یعنی کاربری که از تارنمای شما دیدن می کند درخواستی برای این دیدار می فرستد. بعد سرور درخواست را به پی اچ پی می دهد و پی اچ پی بر طبق کد نوشته شده عمل کرده و پاسخ می دهد؛ در این هنگام سرور پاسخ را برای کاربر می فرستد. فکر می کنیم که عکس زیر گویای توضیحاتی که گفتیم باشد.



منبع عکس: کتاب PHP and MySQL for Dynamic Web Sites: Visual QuickPro Guide نوشته Larry Ullman

در اینجا این درس را به پایان می بریم. در درس آینده کار را با معرفی سینتکس (Syntax) یا نحوه برنامه نویسی با پی اچ پی ادامه می دهیم.

فراموش نکنید که پی اچ پی زبان سمت سرور است؛ پس تا درس بعدی باید مقدمات لازم را فراهم کنید که همان نصب WampServer بر روی رایانه تان است. برای دریافت این نرم افزار به این آدرس مراجعه کنید.

به قسمت پایین صفحه رفته و برنامه مورد نظر خود را با توجه به نسخه سیستم عامل تان در بخش Downloads دریافت کنید. تا درس بعد بدرود :)

درس اول- سینتکس و متغیرها

نحوه کار زبان پی اچ پی

شما می‌توانید از زبان php در میان کدهای html استفاده کنید. برای آشنایی با نحوه کار این زبان ما این مهم را قدم به قدم با کدنویسی ساده انجام می دهیم. ابتدایی ترین چیز در استفاده کردن از این زبان قرار دادن کدهای خود در داخل برچسب (تگ یا Tag) زیر است:

```
<?php

محل قرار گرفتن کد

?>
```

هر کدی که در میان این دو برچسب قرار دهید به عنوان کد پی اچ پی قلمداد می شود. نکته بعدی پسوند فایل شما است که در زبان کدنویسی پی اچ پی بهتر آن است که از پسوند .php استفاده کنید.

به طور مثال نام index.php را برای ابتدایی ترین فایل خود برگزینید. البته برای نمایش یک برگه دارای کد پی اچ پی ما کماکان باید از برچسب های اچ تی ام ال هم استفاده کنیم. به مثال زیر توجه کنید:

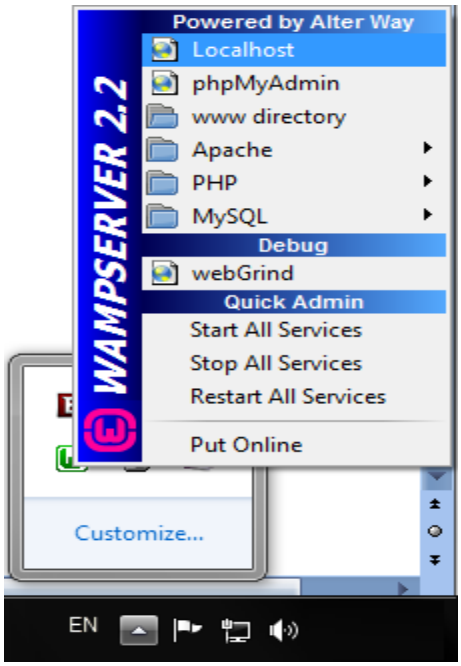
```
<html>

<head>
    <title>آموزش مقدماتی پی اچ پی</title>
</head>

    <body>
        <?php
            echo 'درود';

        ?>
    </body>

</html>
```

همانطور که می بینید برچسب پی اچ پی در داخل برچسب های اچ تی ام ال قرار گرفته است. تابع echo مورد نوشته شده را برای شما چاپ می کند. حال دست به کار شوید و فایلی با نام index.php درست کرده و برچسب های بالا را در آن بگذارید. این کار را با یک فایل متنی و تغییر پسوند آن انجام دهید. test.txt را به index.php تغییر دهید.

نکته: فراموش نکنید که برای نمایش کد در مرورگر خود حتما فایل مورد نظر را در پوشه www که توسط برنامه wamp server ایجاد شده قرار دهید. نشانی این پوشه می تواند چیزی مانند این باشد:

I:wampwww

البته I: می تواند نام درایو محل نصب برنامه باشد. در داخل www پوشه ای به نام PHP ایجاد کنید و فایل را در آن قرار دهید. برنامه wamp را اجرا کرده و بر روی آیکون این برنامه در قسمت راست نوار وظیفه کلیک کرده و مطابق عکس زیر Localhost را انتخاب کنید.



حال شما کلمه «درود» را در مرورگر خود مشاهده می کنید. شما ابتدایی ترین کد خود را در پی اچ پی نوشتید، به همین سادگی.

از این به بعد ما تمرین های این درس را در همین فایل انجام می دهیم و خروجی را در مرورگر می بینیم.

البته راه های بیشتری برای نوشتن برچسب پی اچ پی وجود دارد مانند:

```
<? ?> و یا <script language="php"> </script>
```

که همان ابتدایی بهترین است و پیشنهاد می شود. برای دسترسی مستقیم هم می توانید در محل نشانی مرورگر نشانی http://127.0.0.1/php/index.php را هم وارد کنید. php نام پوشه ای است که شما در پوشه www ساخته اید.

فرستادن داده ها به مرورگر

در بخش قبل دیدید که ما کلمه «درود» را در مرورگر نوشتیم. این کار را با کمک دستور echo انجام دادیم. برای فرستادن داده ها به مرورگر تابع های متفاوتی وجود دارد که معروف ترین آنها echo و print است. برای مثال می توانید به این طریق عمل کنید:

```
echo 'درود';   print "درود";
```

شما این کار را می توانید با کمک ' ' و یا " " انجام دهید که تفاوتی هم با یکدیگر دارند که در دروس بعدی آنها را ذکر خواهیم کرد.

نکته بعدی این است که در آخر تمام کدهای پی اچ پی باید علامت ; یا سمی کالن (Semicolon) قرار بگیرد. مورد مهم بعدی این است که دستورات پی اچ پی را می توانید با حروف بزرگ ECHO یا کوچک echo بنویسید که فرقی برای پی اچ پی نمی کند و همه را یکی در نظر می گیرد ولی بهتر آن است که از همین ابتدا یاد بگیرید که از حروف کوچک استفاده کنید.

نکته: شما نمی توانید از علامت ' ' یا " " در داخلی یکدیگر به طور تکراری استفاده کنید.

مثلا این مورد به شما خطا می دهد:

```
echo "، ویا  ;""درود' حال شما چطور است؟ ،' echo  درود" حال شما چطور است؟";
```

برای نوشتن دستور بالا شما باید به این صورت عمل کنید:

```
echo '، ویا  ;' "درود' حال شما چطور است؟ ،" echo  درود" حال شما چطور است؟" ';
```

در صورت تمایل برای استفاده کردن علامت های یکسان، باید یک علامت \ قبل از هر " که در داخل " قرار می دهید استفاده کنید. مانند:

```
echo "، ویا  ;"\"درود'\ حال شما چطور است؟ ،' echo  درود"\ حال شما چطور است؟"\";
```

```
<body>
    <?php
        echo 'درود';
        ECHO 'درود';
        print "درود";

        echo "درود ." ؟اصد طور چما لاح""";    بروز خطا
        echo 'درود .' ؟اصد طور چما لاح'';    بروز خطا

        echo '؟اصد طور چما لاح "درود ." ';
        echo "؟اصد طور چما لاح 'درود . ";
        echo "؟اصد طور چما لاح "درود\ .\"";
        echo '؟اصد طور چما لاح درود .\' \'';


    ?>
</body>
```

فضای خالی

پی اچ پی اصولا فضای خالی را در نظر نمی گیرد و اگر شما می خواهید به طور مثال یک فاصله یا یک خط فاصله بی اندازید، باید از کدهای اچ تی ام ال در کد پی اچ پی استفاده کنید. به طور مثال:

```
echo  'درود'  .  '<br />'  .  'حال شما چطور است' ;
```

همانطور که می بینید ما از برچسب اچ تی ام ال </ br> در داخل پی اچ پی برای انداختن خط فاصله استفاده کردیم.

نکته: برای چسباندن دو یا چند عبارت باید از علامت نقطه . بین هر عبارت استفاده کنید.

دیدگاه یا Comment

یکی از مهمترین اصول کدنویسی قرار دادن توضیحات یا دیدگاه در مورد کد نوشته شده است تا اگر در آینده خودتان و یا هر کس دیگری که از کد شما استفاده می کند به سراغ آن رفت، به راحتی با دیدن این توضیحات از نحوه عملکرد آن با خبر و در صورت لازم اقدامات مورد نظر خود را انجام دهد.

فرض کنید شما تابعی نوشته اید که یک عملیات ریاضی را انجام می دهد، حال بعد از چند ماه به سراغ آن می روید تا تغییراتی را اعمال کنید. چه اتفاقی خواهد افتاد اگر توضیحی وجود نداشته باشد؟

شما باید دوباره زمان زیادی را صرف فهمیدن و چگونگی رفتار آن کنید. حال فرض کنید شخص دیگری بخواهد این کار را انجام دهد، آن وقت اوضاع بدتر هم می شود. در صورتی که اگر توضیحات باشد یک راهنمای خوب برای ادامه کار می شود.

در پی اچ پی به چند صورت می توانید توضیحات یا دیدگاه بگذارید:

- استفاده از دو علامت // قبل از توضیح. این روش برای زمانی است که می خواهید تنها یک خط را از چشم برنامه دور کنید و به صورت دیدگاه قرار دهید.

```
print 'درود'; //کلمه درود را چاپ می کند
کلمه درود را چاپ می کند//
print 'درود';
```

- استفاده از علامت #: درست مانند نمونه قبلی عمل می کند.

```
print ";  # کلمه درود را چاپ می کند
```

- نوع سوم که با /* آغاز و با */ خاتمه می یابد برای گذاشتن توضیح در چند سطر است.

```
/*  این نوشته برای
توضیح این کد است
*/
```

متغیرها

متغیرها در اصطلاح فضاهایی هستند که مقداری را برای مدت کوتاهی در خود نگهداری می کنند که در صورت لازم از آن در طول مرحله اجرای کد استفاده شود. مقادیر داخل متغیرها می توانند اعداد، متن، کاراکتر و صورت های دیگری از مقدار باشند.

در زبان های برنامه نویسی دیگر مانند جاوا در هنگام تعریف متغیر نیاز است که نوع آن هم مشخص شود در صورتی که در پی اچ پی نیازی به این کار نیست. اما این متغیرها باید از قوانین ویژه ای پیروی کنند که شامل:

۱- یک متغییر باید با علامت $ آغاز گردد. مثلا $name

۲- نامی که برای متغیر انتخاب می کنید می تواند شامل حروف، اعداد و خط فاصله زیرین باشد. مانند $my_name

۳- کاراکتر ابتدایی نمی تواند عدد باشد. به همین دلیل این دو مثال خطا هستند: $name_3 یا $name5

۴- در مورد نام متغیرها بزرگی و یا کوچکی حروف در پی اچ پی مهم و با هم متفاوت هستند. پس $NAME با $name با $Name متفاوت است.

نکته ۱: برای مقدار دادن به متغیرها از علامت '=' استفاده می کنیم. مانند $ = name "darsameh" و برای چاپ نیازی به علامت ' ' یا " " نیست. مانند echo $name ; که کلمه darsnameh را چاپ می کند.

نکته ۲: در زمان قرار دادن یک متغیر در بین کلمات و یا با کلمات نمی توانید از علامت ' ' استفاده کنید و باید حتما از علامت " " استفاده کرد. خروجی این کد:

```
echo 'درود $name' ;
```

برابر با:

```
درود $name
```

می شود که حتما خواسته نویسنده کد نیست. حالت درست به شکل زیر است:

```
echo " درود$name  ";
```

که خروجی آن می شود:

```
درود darsameh
```

حال بیاید یک مثال کامل برای متغیرها درست کنیم. یک برنامه کوچک که نام، نام خانوادگی و سن ما را چاپ می کند. برای این کار سه متغییر نیاز داریم. کد را به شکل زیر می نویسم:

```
<?php
$first_name = 'پیمان';
$last_name  = 'ایرانی';
$age        = ۲۵ ;
print " درود $first_name  $last_name $age ساله ";
?>
```

خروجی می شود:

```
درود پیمان ایرانی ۲۵ ساله
```

با یک مثال دیگر چطورید؟

حالا می خوایم مشخصات مرورگر و آدرس فایل مورد استفاده شما را کشف کنیم. به کدهای زیر دقت کنید. لطفا شما هم این کار را در فایل خود بنویسید. دقت کنید که به هیچ وجه این نوشته ها را رونویسی و بازنویسی نکنید چرا که با نوشتن این کدها دست شما با برنامه نویسی آشنا می شود. خوب ما هم همین را می خوایم نه!

```
<?php
$file = $_SERVER['SCRIPT_FILENAME'];
$user = $_SERVER['HTTP_USER_AGENT'];
echo "<p>نشانی فایل شما در:<br/><b>$file</b>.</p> ";
// چاپ اطلاعات مرورگر
echo "<p>شما از مرورگر زیر استفاده می کنید:
<br /><b>$user</b></p> ";
?>
```

ما در این دو مثال از متغیرها استفاده کردیم و بعد آنها را چاپ نمودیم.

کار با رشته ها یا String

رشته در اصل همان مجموعه ای از کاراکترها است. به طور مثال نام شما یک رشته است. اما ارقام هم می توانند رشته باشند، مثلا '12345'. وقتی یک مقدار رشته ای را به متغیر می دهید، آن متغیر رشته ای می شود و آن مقدار را برای شما نگهداری می کند.

نکته این است که اگر می خواهید یک عدد را به طور رشته در متغیر قرار دهید، باید آنرا در داخل علامت " یا "" قرار دهید. برای چاپ رشته هم باید از همان تابع های echo() و print() استفاده کنید. در قسمت قبلی شما با رشته کار کردید پس در اینجا نیازی به مثال نیست.

نکته: در زمانی که به یک متغیر مقداری داده اید، اگر دوباره مقدار دیگری در آن قرار دهید، متغیر مقدار جدید را نگهداری می کند.

نکته: شما هیچ محدودیتی در طول رشته ندارید.

اتصال رشته

برای اتصال دو رشته به هم از علامت ' . ' استفاده می کنیم. به طور مثال فرض کنید ما دو مقدار رشته داریم و می خواهیم آنها را در کنار هم نمایش دهیم.

```
$f_name = ' پیمان ';
$l_name = ' ایرانی ' ;
$full_name = $f_name  . ' , ' . $l_name ;
print $full_name ;
```

خروجی این کد خواهد شد:

```
پیمان , ایرانی
```

مقدار عددی

از آنجا که در پی اچ پی نیاز ندارید نوع مقدار در متغیر را تعیین کنید، کار با اعداد و به طور کلی متغیرها بسیار آسان است. شما می توانید یک مقدار صحیح یا اعشاری در یک متغیر قرار داده و از آن استفاده کنید.

از جمله کارهایی که شما با اعداد می توانید انجام دهید گرد کردن عدد و یا دادن فرمت خاصی به آن است که از توابع round() و number_format() استفاده می شود.برای فهم این موضوع به مثال زیر توجه کنید:

```
$number = 2050.169 ;
$number = round ($number);
echo $number . "<br />"; // => 2050

$number = 2050.169 ;
```

```
$number = round ($number, 2) ;
echo $number . "<br />";// => 2050.17

$number = 2050.169 ;
$number = number_format($number);
echo $number . "<br />";// => 2,050

$number = 2050.169 ;
$number = number_format($number, 2);
echo $number . "<br />";// => 2,050.17
```

در این مثال ها چند نکته قابل توجه است

ابتدا که هر بار که به متغیر مقدار می دهیم، متغیر مقدار جدید را نگه می دارد.

دوم تاثیر توابع بر روی اعداد است که با توجه به اینکه پارامتری را به آن اضافه کنیم یا نه، این عملکرد متفاوت است. مثلا تابع round() بدون پارامتر یک عدد صحیح باز می گرداند در صورتی که همراه با پارامتر، بسته به مقدار پارامتر آن عدد را به طور اعشاری بر می گرداند.

تابع number_format() دو کار را همزمان انجام می دهد: ۱) به عدد از روی تعداد رقم ها سروسامانی می دهد و عدد را سه رقم سه رقم از هم جدا می کند. ۲) با وجود پارامتر، عدد با توجه به مقدار پارامتر اعشاری باز می گردد.

ثابت ها

ثابت ها در اصل همان متغیر هستند با این فرق که یک بار مقدار می گیرند و این مقدار تغییر نمی کند. همان طور که از نامش پیداست، ثابت است؛ یعنی در طول عملیات کد شما، این مقدار در ثابت ها تغییری نمی کند. برای تعریف کردن ثابت به مثال زیر توجه کنید:

```
define ( '  مقدار ثابت  ' , 'نام ثابت' );
define ( '  NAME' , '  پیمان' );
```

همانطور که می بینید ثابت ها نیاز به علامت $ ندارند و نکته بعدی این است که معمولا برای تعریف ثابت از حروف بزرگ استفاده می شود.

نکته: برای نمایش و چاپ ثابت نباید آن را در علامت های ' ' و یا " " قرار دهید. به مثال زیر توجه کنید:

```
نادرست: echo ' درود NAME'; // => درودNAME
درست: echo ' درود ' . NAME; // => درود پیمان
```

تفاوت علامت های ' ' و " "

در طول این درس ما از این دو علامت استفاده کردیم. حتما تا به حال متوجه فرق بین آنها شده اید. مورد اصلی تفاوت بین این دو طرز برخورد مقادیر در داخل این علامت ها است.

اگر مقداری را در بین علامت ' ' قرار دهید، پی اچ پی این مقدار را یک مقدار متنی در نظر می گیرد در صورتی که مقدار بین علامت " " مقداری است که پی اچ پی آن را ترجمه می کند.

در مثال های قبلی دیدیم که شما توانستید یک متغیر را در میان علامت " " قرار داده و مقدار داخل آن را چاپ کنید اما متغیر داخل ' ' به صورت همان نام متغیر چاپ شد.

نکته: با توجه به تفاوت ذکر شده، پیشنهاد می شود که اگر برای چاپ یک مقدار از متغیر استفاده نکردید، از علامت ' ' استفاده کنید زیرا با این کار به برنامه گفته اید که این یک مقدار متنی است و نیاز به ترجمه ندارد. پس برنامه شما دارای سرعت بیشتری خواهد شد چرا که عملیات ترجمه را انجام نخواهد داد و دنبال متغیر نخواهد گشت.

در درس بعدی ما از پی اچ پی برای ساختن فرم های انتقال داده HTML استفاده خواهیم کرد. پس تا درس بعدی بدرود.

درس دوم- ساخت فرم و انتقال داده از فرم به برگه نمایش

اکنون که شما با نحوه کار پی اچ پی آشنا شدید، وقت آن رسیده که از آن دانش پایه ای استفاده کرده و رفته به رفته برنامه نویسی با پی اچ پی را به طور جدی تر دنبال کنیم. در طول این درس ما تلاش می کنیم از موارد گفته شده استفاده کرده و قدم به قدم نکات جدید را به آن اضافه کنیم. این درس با ساخت فرم در اچ تی ام ال آغاز شده و یاد می گیریم که چطور می توانیم از پی اچ پی برای کار با مقادیر فرستاده شده توسط فرم استفاده کنیم.

ساخت فرم اچ تی ام ال

در ساخت یک تارنمای پویا یکی از مهمترین موارد کار با فرم های اچ تی ام ال و سامان دادن داده های ارسالی از آن توسط پی اچ پی است.

برای انجام این مهم دو کار ضروری است، یکی ساخت خود فرم و دیگری نوشتن کد پی اچ پی برای گرفتن و پردازش داده های ارسالی فرم.

از آنجایی که پیش نیاز این درس اچ تی ام ال و آگاهی کمی از سی اس اس است، ما وارد جزییات چگونگی ساخت فرم نخواهیم شد اما در یک مثال شما این فرم را در اختیار خواهید داشت. هر فرم اچ تی ام ال با برچسب <form method="post" action="form.php"> آغاز شده و با </form> پایان می پذیرد. در داخل این فرم دو ویژگی وجود دارد که در زیر به دلیل استفاده از آنها اشاره می کنیم:

action -: این ویژگی تعیین می کند که داده ها به کدام فایل که کد پی اچ پی در آن قرار دارد ارجاع داده شود.

method -: این نحوه عملکرد در ارسال داده ها را تعیین می کند که در زمان خود به آن بیشتر می پردازیم.

در داخل این برچسب فرم، المان های دیگری نظیر textbox و radio button و select menu و غیره قرار می گیرند. این المان ها براساس نیاز و در جای مشخص خود واقع می شوند. اما نکته مهم نامی است که شما برای آنها انتخاب می کنید که کد پی اچ پی با آن نام ها، آنها را شناسایی می کند.

مشخصات ویژگی های method در فرم

دو انتخاب برای این ویژگی وجود دارد که در زیر به آنها پرداخته می شود:

GET -: وقتی از این ویژگی استفاده می کنیم، داده ها به صورت نام = مقدار به مرورگر شما داده می شود به طوری که شما در نشانی مرورگر خود این داده ها را مشاهده می کنید.

موارد استفاده از این ویژگی زمانی است که شما به طور مثال یک فرم جستجو می سازید و کاربران از این فرم برای پیدا کردن اطلاعات خود استفاده می کنند. در این حالت، داده وارد شده توسط کاربر که در داخل کادر جستجو نوشته شده است به صورت نام = مقدار به مرورگر داده می شود که این در نشانی قابل مشاهده است.

خوبی این کار این است که کاربر می تواند این صفحه را برای خود نشان گذاری یا Bookmark کند و یا با کمک دکمه عقب (Back)، به عقب برگردد.

نگران نباشید! زمانی که فرم را با هم ساختیم، به صورت عملی این مورد را کاملا خواهید دید.

نکته ۱: به خاطر پدیدار شدن داده ها در محل نشانی مرورگر، ما از ویژگی get در مواردی که امنیت در درجه بسیار بالای باشد به هیچ وجه استفاده نمی کنیم.

نکته ۲: در اندازه مقدار داده شده به متد get محدودیت وجود دارد.

- POST: به طور کلی از این ویژگی برای زمانی استفاده می کنیم که امنیت بیشتری لازم داریم؛ چرا که این متد، داده ها را در مرورگر نشان نمی دهد. مثلا فرض کنید می خواهید مقداری را در پایگاه داده ها ثبت کنید و یا می خواهید رایانامه ای را ارسال کنید.

نکته: در استفاده از این متد هیچ محدودیتی برای اندازه داده ارسالی نداریم.

خوب زمان آن رسیده که ما با هم فرمی بسازیم و از دانش کسب شده تا به حال استفاده کنیم. آماده اید؟

ابتدا فایلی با نام form.php ساخته و در پوشه www/php که قبلا در پوشه برنامه WampServer ساخته اید قرار دهید. حال برنامه را اجرا کرده و به آدرس http://127.0.0.1/php/form.php بروید و یا از طریق آیکون برنامه، Localhost را انتخاب کنید. فعلا یک برگه سفید بیش نیست اما در ادامه این برگه را با فرم خود پر می کنیم.

نکته: همانطور که در بالا گفتیم برای این قسمت ما دو مرحله کار داریم: ساخت فرم و نوشتن کد پی اچ پی. نکته قابل توجه این است که ما می توانیم به دو شکل این کار را انجام دهیم. یکی گذاشتن هر دو قسمت یعنی فرم و کدهای پی اچ پی در داخل یک فایل و دیگری جدا کردن این دو قسمت به صورتی که فرم در یک فایل باشد و کد پی اچ پی در فایل دیگر. ابتدا ما به شکل دو فایل که کار را ساده تر می کند این کار را انجام می دهیم و در آینده به شکل تک فایل.

در فایل فرم خود کدهای زیر را وارد کنید:

```
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
     <title>آموزش ساخت فرم اچ تی ام ال</title>
<link rel="stylesheet" type="text/css" media="all" href="style.css" />
</head> <body>
<!-- کد برای ساخت فرم - form.html -->
<form action="form_handler.php" method="get">
<fieldset>
<legend align="right">لطفا فرم زیر را پر کنید</legend>
<label> نام</label>
```

```
<input type="text" name="name" />
<label> رایانامه </label>
<input type="text" name="email" />
<label>دیدگاه</label>
<textarea name="comments" ></textarea>
</fieldset>
<input type="submit" name="submit" value="ارسال" class="submit"/>
</form>
</body>
</html>
```

حال فرم خود را در مرورگر نظاره کنید.

قسمت بعدی فایلی است که کدهای پی اچ پی در آن قرار دارد. فایل دیگری با نام form_handler.php بسازید. این فایل کار دریافت و سازماندهی داده ها را انجام می دهد. اطلاعات زیر را در داخل آن بنویسید:

```
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title> پاسخ  </title>
<link rel="stylesheet" type="text/css" media="all" href="style.css" />
</head>
<body>
<div id="contain">
<?php  // چاپ داده های ارسالی
$name = $_GET['name'];
$email = $_GET['email'];
$comments = $_REQUEST['comments'];
echo 'با سپاس <br /><span> ' . $name . '</span><br />برای این نوشته <br />' ;
?>
<div class="comment"> <?php echo $comments ; ?> </div><br />
ما با شما تماس می گیریم توسط این رایانامه <br />
<div class="email">
<?php echo $email . '<br /><br />' ;?>
</div>
<a href="form.php">برگشت</a>
</div>
</body>
</html>
```

قبل از امتحان کدها یک کار دیگر هم باید انجام دهیم تا داده ها همان طور که ما می خواهیم نمایش داده شوند. فایل دیگری بسازید و نام آن را style.css بگذارید و کدهای زیر را در داخلش بنویسید.

```
body { background-color:#fff; text-align: center; color:#333;}
/* قسمت فرم */
form { width:700px; margin:100px auto auto;}
fieldset{border:1px solid #ddd;}
label, input, textarea{margin:5px 10px 0px auto; text-align: right; clear: both;
float:right;width:60%;}
input{margin:5px 10px 10px auto}
textarea{width:80%; height:150px;}
.submit{float: none; text-align: center; border:1px solid #ddd; color:#039;}

/* قسمت پی اچ پی */
#contain{ border:1px solid #ddd; width:60%; margin:150px auto auto auto;
overflow: hidden;}
span{color: GREEN;}
.email{color:#960;}
.comment{color:#09C; width:80%; border:1px solid #eee; margin: auto;}
```

در حال حاضر ما سه فایل داریم؛ یکی برای ساختن فرم، دیگری برای کار با داده های فرم و آخری برای ساختار نمایش داده ها در مرورگر. حال می توانید عملکرد کد خود را ببینید.

نکته ۱: هر سه این فایل ها باید در داخل یک پوشه قرار گیرند.

نکته ۲: ما از متد GET و REQUEST برای ارسال داده ها استفاده کردیم، لطفا نگاهی هم به نشانی مرورگر بعد از فشردن دکمه ارسال بیاندازید. چه می بینید؟ بله داده ها به صورت نام و مقدار در داخل نشانی مرورگر دیده می شوند. حال در دو فایل فرم و پی اچ پی هر جا که GET یا REQUEST می بینید، آنها را با POST عوض کنید. حال چه شد؟ اثری از داده ها در مرورگر نیست. بعد از استفاده از POST دیگر داده ها در مرورگر فرستاده نمی شوند.

نکته ۳: اگر در فایل form_handler.php دقت کنید ما از متغیری با نام $_REQUEST به صورت آنچه در زیر می بینید استفاده کردیم.

```
$comments = $_REQUEST['comments'];
```

به این نوع متغیر فرا جهانی می گویند که اطلاعات فرستاده شده از فرم را ذخیره می کند. در آینده در مورد این نوع متغیر بیشتر حرف می زنیم.

نکته ۴: کدهای داده شده در این درس قبلا امتحان شده است، پس اگر احیانا به خطایی برخورد کردید، ممکن است که شما کدها را اشتباه وارد کرده باشید. اما اگر از کار خود مطمئن هستید لطفا مورد خطا را در پرسشکده و در تالار مربوط به این دوره اطلاع دهید تا مشکل را حل کنیم.

برای یافتن خطای احتمالی فقط کافیست که به خط اشاره شده در خطا رجوع کنید و بعد در برگه مورد نظر، خط خطادار را پیدا کرده و با کد نوشته شده منطبق کنید.

نکته ۵: اگر به جای نوشته هایی مثل نام و... مطلبی نبود، این یعنی که متغیر خالی است و مشکلی در انتقال داده به متغیر اتفاق افتاده است.

خوب این درس هم به پایان رسید. امیدوارم که با تمرین بتوانید چنین کاری را در نمونه های دیگر انجام دهید. در درس بعدی به شرط ها در پی اچ پی می پردازیم که به ما کمک می کنند قبل از چاپ هر داده گرفته شده از فرم، آن را مورد ارزیابی قرار دهیم.

درس سوم- آشنایی با شرط ها و عملگرها

در درس قبلی با چگونگی ساخت فرم در اچ تی ام ال و ارتباط آن با پی اچ پی برای انتقال داده و کار با آن آشنا شدیم. نکته قابل توجه این است که اگر کاربر در داخل هر کدام از کادرهای تعیین شده در فرم مطلبی ننویسد، باز فرم ارجاع داده خواهد شد و جای آن کادر خالی باقی می ماند.

حال می خواهیم به آشنایی شرط ها بپردازیم و از آنها برای ارزیابی فرم و کادرهایش بهره ببریم به طوری که اگر کادری توسط کاربر خالی ماند، سیستم هشداری بدهد تا آن کادر پر شود.

برای انجام این مهم لازم است که از توابع شرطی و عملگرها استفاده کنیم که موضوع این درس است.

آشنایی با شرط ها و عملگرها

شرط در برنامه نویسی به زبان ساده، قرار دادن یک شرط برای انجام عملی است. ما در زبان روزمره بارها در موقعیت های متفاوت از آنها استفاده می کنیم. به طور مثال «اگر باران بیاید» (شرط) «ما به گردش نخواهیم رفت» (عمل مرتبط به شرط).

این یک شرط است که در صورت درستی شرط، عمل بعد از آن «نرفتن به گردش» اتفاق می افتد. البته این فقط یک شرط بود. ما می توانیم چند شرط را تعیین کنیم و یا از شرط های تو در تو استفاده کنیم. به مثال قبلی از این منظر نگاه کنید: «اگر باران بیاید» و «اگر ماشین نداشته باشیم» (دو شرط) «به گردش نخواهیم رفت».

خب با این مثال ها فکر می کنیم که به منظور استفاده از شرط ها پی بردید. نحوه کار با شرط ها در زبان برنامه نویسی هم دقیقا مثل همان مثال بالاست با این تفاوت که در زبان برنامه نویسی ما این «اگرها» را تبدیل به عبارتی می کنیم که برنامه آن را بفهمد.

مهمترین عباراتی که در ساخت شرط ها از آنها استفاده می کنیم if و else و else if هستند.

if: همان «اگر» خودمان است.

else: در غیر این صورت

else if: معنی آن می شود، در غیر این صورت اگر

شروع کار از عبارت if است، بعد از آن شرط ما قرار می گیرد البته در پرانتز، به دنبال آن هم عمل هایی که می خواهیم در صورت درستی یا نادرستی شرط اتفاق افتد.

```
if(شرط) {
عملی که در صورت درستی باید انجام شود//
}
```

خب حالا کمی با شرط ها آشنا شدیم اما قبل از بیشتر وارد شدن به بحث شرط ها لازم است که ما با عملگرها هم آشنا شویم.

عملگرها در اصل عباراتی هستند که درون پرانتز بعد از if قرار می گیرند تا شرط ما را پیاده سازی کنند. در زیر به لیست عملگرها و کارایی آنها توجه کنید:

- عملگرهای مقایسه ای: این دسته عملگرها برابری دو طرف عملگر را مقایسه می کنند:

- == : برای ارزیابی برابری دو طرف عملگر است که در صورت برابری پاسخ درست می شود.
- =! : این عملگر عدم برابری دو طرف را بررسی می کند که در صورت برابر نبودن پاسخ درست است.
- < : این علامت کوچک تر است، یعنی اگر طرف چپ کوچک تر از طرف راست عملگر است، پاسخ درست می شود.
- > : این درست برعکس کوچک تر یعنی بزرگ تر است. در صورتی که طرف چپ عملگر بزرگ تر باشد، پاسخ درست می شود.
- <= : کوچک تر یا مساوی، هر دو صورت بررسی می شود.
- >= : بزرگ تر یا مساوی.

عملگرهای منطقی: عملگرهای منطقی عبارات را براساس منطق بررسی می کنند:

- ! : این عملگر عبارت را منفی یا برعکس می کند مثلا درست می شود غلط
- && : این همان «و» خودمان است، زمانی این عملگر نتیجه درست می دهد که دو طرف آن بررسی شود. مثلا اگر x=5 و y=7 باشد، در صورت درستی هر دو عبارت، خروجی صحیح می شود.
- || : این همان «یا» می شود. در این عملگر هر کدام از دو طرف درست باشد، پاسخ صحیح می شود. مثلا اگر x=5 یا y=7 شود، هر کدام درست باشد، پاسخ صحیح می شود.
- XOR: خروجی این عملگر زمانی درست است که فقط یکی از دو طرف درست باشد. اگر هر دو طرف درست یا هر دو طرف نادرست باشند، خروجی می شود ناصحیح.

نگران نباشید؛ ما با مثال های فراوان هر کدام از عملگرها و شرط ها را به طور مفصل شرح خواهیم داد. حال اجازه بدهید این را با خود عبارات شرطی بیشتر توضیح دهیم. از تک شرط شروع می کنیم:

```
if(شرط) {
//عملی که در صورت درستی باید انجام شود
}
```

این یک وضعیت تک شرطی است که تنها اگر شرط درست باشد، انجام می شود. مثلا:

```
if(باران بیاید) {
ما به گردش نمی رویم
}
```

یا مثلا اگر مقدار x با مقدار y برابر بود، x را چاپ کن:

```
if($x== $y) {
print $x;
}
```

شرط همراه با else:

```
if (شرط) {
//عملی که در صورت درستی باید انجام شود
}
else {
// عمل دیگری اتفاق افتد
}
```

این یک عبارت شرطی با یک شرط همراه دو عمل است. این عبارت می گوید اگر شرط درست بود، این عمل اتفاق افتد، در غیر این صورت عمل دیگری اتفاق افتد. فرق این شرط با قبلی این است که ما از else برای زمانی که شرط نادرست بود استفاده کردیم. توجه کنید که در مثال قبلی اگر شرط نادرست بود، هیچ اتفاقی نمی افتد اما در این مثال در صورت نادرستی شرط، یک عمل دیگر اتفاق می افتد.

```
if (باران بیاید) {
به گردش نمیرویم
}
else {   // در غیر این صورت
به گردش می رویم
}
```

حال مثال را با متغیرها انجام می دهیم که اگر شرط درست بود یکی را چاپ کند و اگر شرط نادرست بود دیگری را.

```
if ($x == $y) {
print $x ;
}
else {
print $y ;
}
```

در مثال تک شرطی ما فقط به باران آمدن توجه کردیم که اگر بیاید به گردش نمی رویم یا اگر دو مقدار متغیرها برابر بود، یکی را چاپ کند. اما اینجا اگر باران بیاید نمی رویم و اگر نیاید (در غیر این صورت) به گردش می رویم. و در مثال اگر متغیرها برابر بودند، x را چاپ کن، اگر برابر نبودند، y را چاپ کن.

چند شرطی

حالا فرض کنید می خواهیم شرط هایی را به طور پشت سر هم قرار دهیم. توجه کنید:

```
if (شرط) {
عملی که در صورت درستی باید انجام شود//
}
else if ( شرطی دیگر) {
// عمل دیگری اتفاق افتد
}
else {
// در صورت اتفاق نیافتن هر کدام از شرطهای بالا این عمل صورت گیرد
}
```

در اینگونه شرط ها، ما دو یا بیشتر، شرط داریم که برنامه ابتدا شرط ابتدایی را بررسی می کند، اگر شرط درست بود، اتفاق می افتد و از شرط ها خارج می شویم؛ اما اگر شرط ابتدایی درست نبود، شرط دوم بررسی می شود که در صورت درستی عمل، آن شرط اتفاق می افتد. اما اگر هیچ کدام از شرط ها درست نبود آخرین عمل صورت می گیرد.

```
if (باران نیاید) {
به گردش می رویم
}
else if ( ماشین داشته باشیم) {
به گردش می رویم
}
else {   // در غیر این صورت
به گردش نمی رویم
}
```

در این مثال اگر باران نیاید ما به گردش می رویم و شرط تمام می شود ، اگر باران بیاید و اگر ما ماشین داشته باشیم باز به گردش می رویم اما اگر باران بیاید و ماشین هم نباشد نمی رویم.

```
if ($x == $y) {
print $x +$y ;
}
else  if ( $x > $y) {
print $x ;
}
else {
print $y ;
}
```

حال بیایید به شرط های تو در تو بپردازیم. نگران نباشید؛ ما برای هر کدام مثال می آوریم تا خوب برای شما مشخص گردد.

حالا می خواهیم برای اتفاق افتادن یک عمل دو شرط بگذاریم، مثال «اگر باران بیاید» و «اگر ماشین نداشتیم» یادتان هست!

```
if (شرط) {
if (شرط) {
//عملی که در صورت درستی باید انجام شود
  }
          }
else {
// عمل دیگری اتفاق افتد
}
```

به این گونه از شرط گذاری شرط تو در تو می گوییم، که در صورت درستی شرط ابتدایی، برنامه به شرط داخل شرط ابتدایی یعنی شرط دوم می رود، اما در صورت نادرستی شرط ابتدایی، شرط داخل آن اصلا بررسی نمی شود و برنامه به سراغ else می رود.

```
if (باران بیاید) {
if ( ماشین نداشته باشیم) {
به گردش نمی رویم
}
}
else {   // در غیر این صورت
به گردش می رویم
}
```

به مثال زیر توجه کنید که از چند شرط به طور تو در تو استفاده شده است. این مثال می گوید که اگر متغیرها با هم برابر نیستند، هر کدام که بزرگ تر است را چاپ کن. در غیر این صورت یعنی اگر برابرند، جمع آنها را چاپ کن.

```
if ($x != $y)
{
if ( $x > $y){
```

```
print $x;
}
else {
print $y;
    }
}
else {
print $x +$y ;
    }
```

طبق مثال بالا اگر شرط ابتدایی برقرار نباشد یعنی اگر متغیرها برابر باشند به قسمت else آخر می رود و جمع آنها را چاپ می کند. اما در صورت برابر نبودن یعنی نادرست بودن شرط به داخل شرط ابتدایی رفته و شرط های دیگر را بررسی می کند.

نکته ۱: در زمانی که عمل بعد از درستی شرط یکی باشد، نیازی به گذاشتن { } نیست. این کاراکتر زمانی ضروری است که شما بخواهید چند دستور را بعد از درستی شرط انجام دهید. اما پیشنهاد می شود که شما همیشه از این کاراکترها استفاده کنید.

نکته ۲: تلاش کنید همیشه در زمان تعریف یک شرط هر دو کاراکتر {} را قبل از هر چیزی بگذارید چرا که در صورت استفاده از چندین شرط ممکن است ترتیب کاراکترها به هم بخورد که باعث ایجاد خطا می شود.

Switch

نوع دیگر شرط گذاری استفاده کردن از switch است که کارایی خود را دارد. نحوه نوشتن آن به شکل زیر است:

```
switch (متغیر)
{
    case 'مقدار ابتدایی' :
    {
    // دستور
        break ;
}
     case 'مقدار دوم' :
{
    // دستور
    break;
}
    default:
{
    // دستور
```

```
    break;
}
}
```

در این نوع شرطی، switch مقدار متغیر را با هر کدام از عبارات مقابل case مقایسه می کند. وقتی این مقدار برابر با هر کدام از عبارات شد، دستورات داخل آن case انجام می شود و بعد با کمک دستور break از داخل شرط خارج می شود. در درس های آینده از این نوع شرطی بیشتر حرف خواهیم زد و بهتر آن است که فعلا روی if تمرکز کنیم.

کاربرد شرط ها در برنامه

در درس قبل ما یک فرم درست کردیم و به کمک کدهای پی اچ پی داده ها را از فرم گرفته و در صفحه دیگری نمایش دادیم. مشکلی که این فرم داشت این بود که در زمانی که کاربر کادری از فرم را پر نمی کرد باز فرم عمل کرده و جای آن کادر را خالی می گذاشت.

حال با فراگیری شرطی ها و عملگرها می خواهیم این نقض را برطرف کنیم به صورتی که اگر کاربر چیزی ننوشت یا موردی را خالی گذاشت در صورت نیاز به او تذکر داده تا برگردد و فرم را کامل کند.

لطفا برنامه WampServer خود را اجرا کرده و سه فایل با نام های form.php و form_handler.php و style.css را ساخته و در پوشه www/php خود قرار دهید.

فایل form.php را باز کرده و کدهای زیر را داخل آن قرار دهید.

```
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <title>آموزش شرطی با ساخت فرم اچتیامال</title>
    <link rel="stylesheet" type="text/css" media="all" href="style.css" />
</head>
<body>
<!-- کد برای ساخت فرم - form.html -->

<form action="form_handler.php" method="post">
<fieldset>
<legend align="right">لطفا فرم زیر را پر کنید</legend>
<label> نام</label>
<input type="text" name="name" />
<label> رایانامه </label>
<input type="text" name="email" />
<label>دیدگاه</label>
<textarea name="comments" ></textarea>
</fieldset>
<input type="submit" name="submit" value="ارسال" class="submit"/>
```

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

این دقیقا برابر همان فرمی است که در درس قبل ساختیم، البته با کمی از تغییرات مثلا تغییر متد.

حال ما نیاز به یک ارزیاب فرم با کدهای پی اچ پی داریم. فایل form_handler.php را باز کنید و کدهای پی اچ پی را در آن قرار دهید.

```
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <title> پاسخ  </title>
    <link rel="stylesheet" type="text/css" media="all" href="style.css" />
</head>
<body>
<div id="contain">
<?php // قرار دادن داده ها در متغیر
$name = $_POST['name'];
$email = $_POST['email'];
$comments = $_POST['comments'];
if ( !empty($_POST['name']) && !empty($_POST['comments']) &&
!empty($_POST['email']) )
{
echo 'با سپاس <br /><span> ' . $name . '</span><br />برای این نوشته <br />' ;
?>
<div class="comment"> <?php echo $comments ; ?> </div>
ما با شما تماس می گیریم توسط این رایانامه <br />
<div class="email">
<?php echo $email . '<br /><br />' ;?>
</div>
<?php
}
else
{
echo '<div class="error">لطفن برگردید و فرم را کامل پر کنید</div>';
}
?>
<a href="form.php">برگشت</a>
</div>
</body>
</html>
```

بررسی کدهای form_handler.php

در این کدها ابتدا ما داده های فرستاده شده از فرم را گرفته و در داخل متغیرها می گذاریم. سپس از شرطی ها برای بررسی داده ها استفاده می کنیم. کاری که انجام می شود این است که با یک «اگر» بررسی می کنیم که آیا همه کادرها پر شده یا نه.

```
$name = $_POST['name'];
$email = $_POST['email'];
$comments = $_POST['comments'];
```

این سه خط، داده ها را از فرم توسط متغیرهای پر شده $_POST گرفته و در متغیرهای تعریفی جدید می گذارند. حال می توانیم با بررسی مقدار داشتن یا نداشتن این متغیرها به خالی بودن یا پر بودن کادرها پی ببریم. این کار را با شرطی زیر انجام می دهیم:

```
if ( !empty($name) && !empty($_POST['comments']) && !empty($_POST['email']) )
```

این شرطی بررسی می کند که اگر متغیرهای نام، رایانامه و دیدگاه خالی نیست، دستورهای بعدی انجام شوند. در صورتی که پاسخ این شرط منفی باشد، به قسمت else می رود و هشدار خالی بودن را می دهد.

```
else
{
echo '<div class="error">لطفا برگردید و فرم را کامل پر کنید</div>';
}
```

همانطور که ملاحظه می کنید در داخل پرانتز if از هر دو نوع متغیر استفاده شده تا به شما گوناگونی این کاربرد را نشان دهیم. به طور مثال یکجا از خود متغیر ($_POST['comments']) استفاده شده و در جای دیگر متغیر $_POST['name'] را درون متغیر دیگری با نام $name ریخته و از آن استفاده کردیم.

در داخل این شرط از دو عملگر && و ! برای مطمئن شدن از خالی نبودن متغیرها استفاده شده است.

در زمانی که این شرط ها درست باشند، برنامه به داخل if رفته و دستورات چاپ متغیرها را اعمال می کند.

برای نمایش بهتر داده ها در صفحه مرورگر می توانید فایل سوم را هم به پوشه اضافه کنید. این فایل همان فایل سی اس اس است که در درس قبلی داشتیم.

نکته: ممکن است که شما نخواهید کاربر را مجبور کنید که برای همه کادرها داده ای وارد کند. در این صورت می توانید آن قسمت را از شرط حذف کنید.

به کاری که ما در این قسمت انجام دادیم ارزیابی داده ها می گویند. در طول دوره ما تلاش می کنیم که از تمام آموخته ها در موقعیت های مختلف استفاده کنیم تا شرایط بهتر و ساده تری برای یادگیری این مفاهیم بوجود آید.

در درس بعدی ما به یکی دیگر از انواع داده ها یعنی آرایه ها می پردازیم. آرایه ها دقیقا مانند دیگر متغیرها هستند با این تفاوت که آنها می توانند بیش از یک مقدار را برای ما نگهداری کنند.

درس چهارم- آشنایی با آرایه ها

تا اینجا در زمانی که خواستیم مقداری برای مان نگهداری شود تا در طول برنامه از آن استفاده کنیم، این مقدار را درون متغیرها که خانه های خالی حافظه هستند، قرار دادیم. اما این متغیرها تنها قادرند که یک مقدار را برایمان ذخیره کنند؛ پس اگر خواستیم چندین مقدار را ذخیره کنیم چی؟

اینجاست که آرایه ها به کمک ما می آیند. در این درس به شرح آرایه ها و چگونگی استفاده از آنها برای ذخیره چندین متغیر می پردازیم.

آرایه ها

آرایه ها یک نوعی از متغیرها هستند که می توانند چند داده را در قالب یک نام ذخیره کنند. فرض کنید که می خواهید اسامی کارمندان یک شرکت را به طور موقت ذخیره کنید تا در طول برنامه از آنها بتوانید استفاده کنید.

اگر برای این کار از متغیرهای آرایه ای استفاده نکنیم، باید برای هر کدام از این اسامی یک متغیر معمولی تعریف کنیم که اگر تعداد آنها زیاد باشد، چه بسا که کار نشدنی شود. حال آن که با آرایه، تمام این اسامی تحت یک نام آرایه ای ذخیره می شوند و در زمان نیاز با یک اندیس و نام آرایه به آن دسترسی پیدا می کنیم.

هر آرایه از دو بخش تشکیل شده: نام آرایه و اندیس.

نام آرایه که مشخص است و در زمان ایجاد آن، خود برنامه نویس انتخاب می کند. اما اندیس، کلید یا مقداری است که به یکی از مقادیر ذخیره شده در آرایه اشاره می کند. مثلا وقتی که پنج مقدار را در آرایه ریختیم، برای شناختن هر کدام از این مقادیر یک اندیس منحصر به فرد وجود دارد که به مقدار خاصی اشاره می کند.

ساختار آرایه

پی اچ پی دو نوع آرایه را حمایت می کند، یکی آرایه با اندیس عددی، یعنی اندیس اشاره گر

به مقدار عدد است و دیگری آرایه با اندیس کاراکتری یا کلمه ای که اندیس خود یک کاراکتر یا کلمه تعریف می شود. در زیر نمونه هایی از این دو نوع را می بینید:

```
$name[2]
$name[ ' staff ' ];
```

نکته ۱: قوانینی که برای نام گذاری متغیرها بیان شدند، برای آرایه ها هم برقرار هستند.

نکته ۲: به خاطر اینکه یک آرایه، مقادیر بیشماری را در خود نگه می دارد، طریقه چاپ آن با یک متغیر معمولی فرق دارد.

نکته ۳: پی اچ پی خود از آرایه های از قبل تعریف شده به نام متغیرهای فرا جهانی یا Super Global Variables استفاده می کند. نمونه هایی از این دست آرایه های از پیش تعریف شده را قبلا دیده اید. مثلا:

```
$_GET, $_POST, $_REQUEST ...
```

طرز ساخت آرایه

در زبان پی اچ پی راه های متفاوتی برای ایجاد یک آرایه وجود دارد، اما در کل دو راه اصلی است که این کار را برایمان مشخص می کند که با مثال برایتان شرح داده خواهد شد.

نوع اول

تعریف و دادن مقدار به آرایه در زمان ایجاد آن. به مثال زیر توجه کنید که ابتدا آرایه تعریف و همزمان بدان مقدار داده می شود:

```
$name[] = ' حمید ' ;
$name[] = ' سعید ' ;
$name[ ' staff3 ' ] = ' پیمان ' ;
```

در این حالت اندیس بطور خودکار از 0 تا ... به آرایه اضافه می شود، یعنی مقادیر برای فراخوانی باید با کلید خود خواسته شوند، و یا می توان از کلید کلمه ای استفاده کرد. توجه کنید:

```
$name[0]   // حمید
$name[1]  //   سعید
$name[' staff3 ']  //     پیمان
```

حال مقدار نخستین آرایه با کلید 0 می شود 'حمید'، دومی می شود 'سعید' و سومی هم 'پیمان'. برای چاپ کردن این مقادیر هم می توانید به صورت یکی از روش های زیر عمل کنید:

```
echo $name[0] ;
echo $name[1] ;
```

```
echo $name[' staff3 '] ;
echo $name[0] . ' ' . $name[1] . ' ' . $name[' staff3 ']  ;
```

نوع دوم

در نوع دوم به جای تعریف یک آرایه به طور یکی یکی، می توانید با کمک تابع array() کل مقادیر را یکجا در آرایه قرار دهید که این قابلیت هم به چند شیوه قابل اجرا است. مانند:

```
$name = array ( 'حمید ' , ' سعید ' , ' پیمان ' ) ;
```

خوب مقادیر به ترتیب با کلیدهای 0 و 1 و 2 وارد آرایه می شوند.

```
$name = array ( 1 => 'حمید ' , ' سعید ' , ' پیمان ' ) ;
```

در این حالت ما عدد کلید شروع شونده را از 0 به 1 تغییر دادیم تا مقادیر با کلید 1 شروع به ذخیره شدن کنند. پس مقادیر به ترتیب با کلیدهای 1 و 2 و 3 وارد آرایه می شوند.

```
$name = array (' staff1' => 'حمید  ' , ' staff2' => ' سعید ' , 'staff3' => ' پیمان ' ) ;
```

در این حالت ما از کلید کلمه ای استفاده کردیم که به جای عدد، مقادیر با کلیدهایی که تعریف می کنیم وارد شوند. توجه کنید که در این حالت برای فراخوانی مقداری از آرایه باید نام کلید را بدانید؛ مثلا فرض کنید ما می خواهیم نام سعید را چاپ کنیم که باید به شیوه زیر باشد:

```
echo $name[ ' staff2 ' ] ;
```

نکته ۱: از تابع array() می توانیم برای تعریف نخستین یک متغیر آرایه ای استفاده کنیم مانند:

```
$name = array();
$name[] = '  سعید ' ;
```

همانطور که می بینید در جمله ابتدایی، متغیر $name را به صورت یک آرایه تعریف کردیم و در جمله دوم بدان مقدار دادیم.

نکته ۲: از تابع range() می توانیم برای ایجاد آرایه ای متشکل از یک سری اعداد پشت سر هم استفاده کنیم.

```
$number = range ( 1 , 20 );
```

در این حالت ما آرایه ای تعریف کردیم که اعداد 1 تا 20 را با کلید 0 تا 19 در خود جا داده است.

```
echo $number [0] ; // 1  خروجی می شود
echo $number [10] ; // 11  خروجی می شود
echo $number [19] ; // 20  خروجی می شود
```

به خاطر شروع شدن کلید از 0، پس عدد 1 در آرایه ای با کلید 0 ذخیره می شود.

دسترسی به آرایه ها

در استفاده کردن از آرایه و چاپ آن دیدیم که این عمل چطور با دانستن کلید آرایه انجام شد. اما این راه فقط برای دسترسی به یک آرایه است. راه دیگری برای دسترسی همزمان به همه آرایه ها وجود دارد که از تابع foreach() استفاده می شود. ساختار این تابع به شکل زیر است:

```
foreach ( $array as $value ) {
//  دستورات برای انجام و استفاده از مقادیر
}
```

این تابع دارای دو پارامتر است که ابتدایی به نام آرایه شما اشاره دارد و دومی به تک تک مقادیر در آرایه. مثلا اگر بخواهیم اعدادی از 1 تا 20 را چاپ کنیم، می توانیم بنویسیم:

```
$number = range ( 1 , 20 ) ;
foreach ( $number as $num ) {
echo $num . '   ' ;
} //  1 2 3 4 5 ..... 20
```

دقت کنید که در زمان چاپ اعداد، بعد از هر عدد یک فاصله با کمک . ' ' گذاشتیم. علاوه بر این می توانیم در داخل ' </ br> ' بگذاریم که اعداد را در حالت ستونی چاپ کند:

```
echo $num . ' <br /> ' ;
```

نوع دیگر استفاده از این تابع گذاشتن کلید در آن است. توجه کنید:

```
$name = array (' staff1' => 'حمید  ' , ' staff2' => ' سعید ' , 'staff3' => ' پیمان ' ) ;
foreach ( $name as $key => $num ) {
echo $num . ' <br /> ' ;
}
```

مرتب کردن آرایه ها

برای مرتب کردن آرایه ها براساس حروف الفبا از تابع sort() استفاده می شود. ما در پی اچ پی قادر هستیم که آرایه ها را براساس چند فاکتور مرتب کنیم.

ابتدا اینکه آرایه را براساس مقدار داخل آن یا براساس کلیدهای آن مرتب کنیم.

```
$name = array (' staff1' => 'حمید  ' , ' staff2' => ' سعید ' , 'staff3' => ' پیمان ' ) ;
sort($name);
foreach ( $name as $key => $num ) {
```

```
echo $num . ' <br /> ' ;
}
```

نکته: در این حالت مرتب سازی با استفاده کردن از تابع sort()، این تابع کلیدهای وارد شده برای آرایه را پاک کرده و مجددا خود از 0 تا... کلیدگذاری می کند. پس اگر برای آرایه خود کلید کلمه ای انتخاب کردید، هرگز از این تابع استفاده نکنید. برای دیدن این تغییرات کافیست که مطابق زیر عمل کنید:

```
$name = array ('staff1' => 'سعید' , 'staff2' => ' حمید  ' , 'staff3' => '  پیمان  ' ) ;
sort($name);
foreach ( $name as $key => $num ) {
echo $key . ' => ' . $num . ' <br /> ' ;
}
```

مشاهده می کنید که کلیدهای چاپی آنهایی که تعریف کردیم نیستند؟

مرتب کردن با حفظ کلیدها

برای مرتب کردن آرایه و نگهداشتن کلیدها می توانید از تابع asort() استفاده کنید به گونه ای که این تابع کلیدهای شما را دست نخورده باقی می گذارد.

```
$name = array ('staff1' => 'سعید' , 'staff2' => ' حمید  ' , 'staff3' => '  پیمان  ' ) ;
asort($name);
foreach ( $name as $key => $num ) {
echo $key . ' => ' . $num . ' <br /> ' ;
}
```

مرتب کردن براساس کلیدها

این حالت زمانی استفاده می شود که ما بخواهیم آرایه را براساس کلیدها و نه براساس مقادیر مرتب کنیم:

```
$name = array ('staff' => 'سعید' , 'manager' => ' حمید  ' , 'customer' => '  پیمان  ' )
;
ksort($name);
foreach ( $name as $key => $num ) {
echo $key . ' => ' . $num . ' <br /> ' ;
}
```

نکته: برای نمایش یک جای یک آرایه در زمان هایی که می خواهید رفع اشکال کنید، می توانید مانند زیر از توابع اشاره شده استفاده کنید.

```
$name = array (  'staff1' => 'سعید' , 'staff2' => ' حمید  ' , 'staff3' => '  پیمان  ' ) ;
print_r($name);
echo "<br />";
var_dump($name);
```

ساختار for و while

حلقه ها ساختاری در زبان برنامه نویسی هستند که به شما اجازه دسترسی به همه مقادیر را به طور مرتب می دهند. در قسمت قبلی ما از یک نوع این حلقه ها استفاده کردیم: foreach() یادتان هست!

دو نوع دیگر از حلقه ها که در اینجا معرفی می کنیم for و while هستند.

حلقه for

این نوع حلقه دارای سه پارامتر است که شامل نقطه شروع حلقه، شرط پایانی حلقه و شمارش اندیس یا کلید حلقه می شود.

```
for ( شمارش اندیس;   شرط پایانی  ;   نقطه شروع   )
{
// دستور
}
```

اجازه بدهید با یک مثال کارکرد for را بیشتر شرح دهیم. فرض کنید می خواهیم که از شماره ۱ تا ۱۰ را چاپ کنیم:

```
for ($i = 1 ; $i <= 10; $i++)
{
echo $i . ' <br /> ' ;
}
```

جمله ابتدایی پارامتر این حلقه i = 1 است که همان اندیس شمارشگر ماست، بعد از آن شرط پایانی حلقه است که می گوید تا زمانی که اندیس کوچک تر یا برابر ۱۰ شود. در آخر هم شمارشگر که هر بار که حلقه اجرا می شود یکی به آن اضافه می کند.

نکته: مورد استفاده از این نوع حلقه زمانی است که ما تعداد تکرار حلقه را می دانیم. به طور مثال در قسمت قبل ما می دانستیم که می خواهیم از شماره 1 تا 10 را چاپ کنیم.

حلقه while

این حلقه هم مانند حلقه قبلی همان کار تکرار براساس شرط را انجام می دهد، ولی با یک تفاوت بزرگ و آن این است که در این نوع حلقه تعداد یا میزان تکرار را نمی دانیم. پس بر این اساس جایگاه استفاده از این نوع حلقه تغییر می کند. اجازه بدهید ابتدا مثال قبلی را توسط این حلقه بنویسیم تا تفاوت این دو مشخص تر شود.

```
$i =1;
while ( $i <= 10 ){
print $i . ' ';
$i++;
}
```

نکته: پی اچ پی نوع دیگری از حلقه دارد به نام do...while که شیوه کار آن درست مثل حلقه while است. با این تفاوت که در حلقه while در صورت درست نبودن شرط، دستورات داخل حلقه اجرا نمی شود اما در do...while حتی در صورت نادرستی شرط، باز هم اجرا می شود اما تنها برای یکبار.

```
$i = 1 ;
do {
print $i . ' ';
$i++;
} while ( $i <= 10 );   //  1 2 3 4 5 6 7 8 9 10
```

اگر در این مثال متغیر $i = 11 شود، یکبار این عدد چاپ می شود و بعد در شرط به خاطر برقرار نبودن شرط که باید کمتر از ۱۰ باشد، حلقه متوقف می شود.

```
$i = 11 ;
do {
print $i . ' ';
$i++;
} while ( $i <= 10 ); // 11
```

با به پایان رساندن این بخش ما مقدمات یادگیری اصول ابتدایی برای برنامه نویسی در زبان پی اچ پی را به پایان بردیم.

نکته ای که لازم می دانیم در اینجا ذکر کنیم این است که شما نیازی ندارید در این زمان تمام قسمت های قبلی را حفظ کنید، بلکه در طول دوره و با انجام مثال های بهتر و عملی و نزدیک تر به شیوه برنامه نویسی، در دنیای بیرون این اصول به طور خود به خود به مغز شما راه می یابند.

پس نگران نباشید، فقط تلاش کنید که مطالب را بفهمید و با تمرین لازم، خروجی های داده شده را بدست آورید که نشانه درستی عملکرد شما از دید برنامه نویسی است.

در درس بعدی ما به سراغ طراحی و ساخت تارنمای پویا رفته و از تکنیک های پی اچ پی در آن یاد خواهیم کرد. این تکنیک ها شامل چگونگی استفاده از چندین فایل برای ساخت یک تارنما و... می شود.

درس پنجم-استفاده از فایل های متفاوت(فایل های خارجی)

با اطلاعاتی که تا به اینجا کسب کردیم، حال زمان آن رسیده تا با مفاهیم مهمتر و جالب تر پی اچ پی آشنا شویم.

یکی از این مفاهیم، استفاده کردن یا وارد کردن فایل هایی خارجی به درون فایل اصلی خود، در پی اچ پی است. عبارت تارنمای پویا در اصل با بکارگیری این مفاهیم معنی پیدا می کند. حال اگر بخواهیم به طور حرفه ای تر تارنمای پویا را شرح دهیم، می گوییم: یک تارنمای پویا سیستمی است که از نظر نگهداری و تولید محتوا راحت تر و در مقابل کاربر به صورت تعاملی عمل می کند. این بدین معنی است که شما می توانید محتوا را براساس پاسخی به درخواست کاربر تعیین کنید.

قرار دادن چند فایل در یک فایل

تا به امروز همه کدها چه اچ تی ام ال و چه پی اچ پی در یک فایل قرار داشت. اما امروزه برای توسعه تارنماهای پیچیده تر این کار کمی اشتباه و نادرست است. شما با تقسیم فایل بزرگ تر به قسمت های کوچک تر می توانید کار را برای خود ساده تر و حرفه ای تر انجام دهید. اجازه بدهید با یک مثال این را بیشتر شرح دهیم:

فرض کنید شما سیستمی شامل ۱۰۰ برگه دارید که سیستم به صورت ایستا ساخته شده است، حال می خواهید یک قسمت در سربرگ صفحه ها را تغییر دهید. چه باید کرد؟ باید این کار را در تمام ۱۰۰ برگه انجام داد در صورتی که در سیستم پویا شما یک فایل برای سربرگ می سازید و آن را در بدنه فایل های دیگر وارد می کنید. حال برای تغییر تنها کافی است که آن فایل را تغییر دهید. البته در ادامه خودتان با انجام این کار، راز این نوع برنامه نویسی را کشف خواهید کرد.

نحوه قرار دادن فایل

کلا چهار تابع برای وارد کردن یک فایل در فایل دیگر وجو ددارند که شامل:

- include()
- include_once()
- require()
- require_once()

برای استفاده از این توابع و وارد کردن فایلی با آنها باید به طریق زیر عمل کرد:

```
include('نام فایل / نشانی فایل');
include(' files/header.html');
```

همانطور که مشاهده می کنید در داخل پرانتز ابتدا آدرس و بعد نام فایل آمده است. در اصل file پوشه ای است که شما فایل header.html را در آن قرار داده اید.

تمام توابع بالا عملیات قرار دادن را برای شما انجام می دهند اما با کمی تفاوت. فرق include() با require() این است که در هنگام بروز خطایی در برنامه، با استفاده کردن از include()، یک پیغام نشان داده می شود اما ادامه کدها اجرا و نشان داده می شوند در حالی که در require() ابتدا پیغام خطا نمایان و بعد اجرای کد قطع می شود.

نکته: معمولا در زمان هایی که می خواهیم اطلاعات مهمی را توسط این توابع وارد فایل دیگری کنیم، از تابع require() استفاده می کنیم. این بدین معنی است که ما می خواهیم فایل مورد نظر حتما وارد شده و در صورت بروز خطا یا درست وارد نشدن آن به ما هشدار دهد و از اجرای باقی کدها جلوگیری شود.

هر دوی این توابع نوع دیگری دارند که همراه با _once می آید. این توابع ضمانت می کنند که حتی در صورت دوباره فراخوانی فایل، آن فایل فقط یک بار وارد شود. once در زبان پارسی «یک بار» معنی می شود.

برای نوشتن نشانی فایل در قسمت ابتدایی پرانتز تابع، ما می توانیم به دو صورت عمل کنیم، یکی «مطلق» و دیگری «نسبی».

آدرس دهی مطلق: آدرس دهی مطلق نشانی فایل را از ریشه فهرست رایانه محاسبه و فراخوانی می کند. به طور مثال:

```
include (' C:/php/includes/header.html ');
```

همان طور که می بینید نشانی از فهرست C: آغاز شده و به خود فایل header.html رسیده است.

آدرس دهی نسبی: در این نوع آدرس دهی، نشانی از خود فایلی که در آن هستید و تقاضای قرار دادن فایل را می کند آغاز می شود. مثلا:

```
include(' ../includes/header.html ');
```

فرض کنید فایل فراخوان کننده در داخل پوشه scripts با آدرس

```
C:/php/scripts/
```

قرار دارد و می خواهد فایل header.html را فراخوانی کند. در این صورت ابتدا باید از پوشه scripts بیرون رفته و بعد وارد include شود و باقی آدرس ها.

علامت /.. کار به عقب رفتن از پوشه کنونی را انجام می دهند و بعد به پوشه دلخواه باید راهنمایی شوند. به این شیوه آدرس دهی که نشانی از خود فایل فراخوان کننده آغاز می شود، آدرس دهی نسبی می گوییم. حال زمان آن رسیده که از اندوخته های یادگرفته استفاده کنیم.

قرار دادن فایل خارجی

ابتدا به داخل پوشه php که قبلا ساخته اید بروید و پوشه جدیدی با نام files including بسازید. وارد آن شوید و فایلی با نام index.php و پوشه ای با نام includes بسازید. در داخل پوشه includes هم فایل هایی با نام های header.html و footer.html وstyle.css بسازید.
حال ما پوشه ای به نام including files داریم که در داخل آن فایل index.php و پوشه includes قرار دارند.

سربرگ یا header: در داخل این فایل کدهای زیر را بنویسید:

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" charset=utf-8" />
<title> فایل های خارجی</title>
<link rel="stylesheet" href="includes/style.css" type="text/css" media="screen" />
</head>
<body>
<div id="header">
<h1>  قسمت سربرگ</h1>
<div id="navigation">
<ul>
<li><a href="index.php"> خانه</a></li>
</ul>
</div> </div>
<div id="content"> <!-- قسمت محتوا -->
```

همان طور که مشاهده می کنید در این فایل قسمتی از کدهای اچ تی ام ال که شامل برچسب ها می شوند، قرار دارد. این بدین خاطر است که این فایل در داخل فایل اصلی فراخوانی می شود و در بالای باقی کدها قرار می گیرد. سربرگ را بسته و محتوای آن را ثبت کنید.

footer یا پایین برگ: فایل footer.html را باز کنید و کدهای زیر را در داخل آن قرار دهید:

```
</div><!-- پایان قسمت محتوا -->
<div id="footer">
<p>
   قسمت پایین برگه
</p>
</div>
</body>
</html>
```

در داخل این فایل برچسب های بستن html و body به همراه دیگر کدها قرار دارند. این فایل هم همانند فایل سربرگ در داخل فایل اصلی قرار داده می شود.

index.php: در این فایل ما باید دو فایل دیگر را فراخوانی کنیم و این کار را به صورت زیر انجام می دهیم:

```
<?php include ('includes/header.html'); ?>
<h1> قرار دادن فایل های دیگر در فایل اصلی</h1>
<p> در این قسمت محتوای تارنما قرار می گیرد. این قسمت همان فایل ایندکس
است.</p>
<?php include ('includes/footer.html'); ?>
```

همانطور که می بینید در ابتدا و پایان این صفحه دو برگه header و footer توسط تابع include وارد شده اند و هر آنچه در میان این دو تابع گذاشته شوند به صورت محتوا در داخل کادر اصلی نمایان می شوند.

حال فرض کنید شما ۱۰۰ برگه دارید که نیاز دارند این دو قسمت سربرگ و پایین برگ را نمایش بدهند. تنها کار وارد کردن این دو برگه به صورت بالا در داخل آنها است.

شیوه نمایش محتوا از طریق Style: بخش آخر که شیوه نمایش محتوا را تعیین می کند، مطابق معمول در فایل style.css قرار می گیرد.

```
body {
    background:#fafafa;
    color:#555;
    font:12px Tahoma,Arial, Helvetica, "bitstream vera sans", sans-serif;
}
/* header */
#header {
    border:1px solid #bbb;
    height:130px;
    margin:10px auto;
    width:751px;
}
#header h1 {
    color:#888;
    font-size:24px;
    text-align:right;
}
/* navigation */
#navigation {
    border-top:1px solid #ccc;
    margin:auto;
    width:740px;
}
```

```
#navigation ul{
    overflow:hidden;
    list-style:none;
    margin:0px;
}
#navigation li {
    border-left:1px solid #bbb;
    float:right;
    width:130px;
    height:40px;
    list-style:none;
    margin:0 auto;
}
    #navigation a {
    color:#555;
    display:block;
    line-height:40px;
    text-align:center;
    font-size:18px;
    text-decoration:none;
}
#navigation a:hover {
    background:#e3e3e3;
    color:#333;
}
#navigation .active {
    background:#e3e3e3;
    color:#777;
}
/* content */
#content {
    border:1px solid #ccc;
    height:auto;
    min-height:400px;
    margin:10px auto;
    padding:0 0 20px;
    width:751px;
    text-align:right;
}

#content h1 {
    font-size:16px;
    padding:20px 0 0;
}
```

```
#content p {
    padding:20px 20px 0;
}
/* footer */
#footer {
    border:1px solid #ccc;
    height:100px;
    margin:0 auto;
    padding:10px 0;
    text-align:center;
    width:751px;
}
```

این شیوه گذاری به ما اجازه می دهد که محتوا را هر طور که دوست داریم نمایش دهیم.

حال یک نگاه به خروجی کار بیاندازیم:

اتفاقی که در صورت استفاده از فایل های متفاوت در ساخت تارنما می افتد این است که مرورگر شما ابتدا به طور پیش فرض به سراغ فایل index می رود و شروع به خواندن کدهای داخل این فایل از بالای صفحه می کند.

نخست به تابع include() که فایل سربرگ را وارد می کند، برخورد می کند که با اجرای این تابع، محتوای فایل سربرگ در داخل فایل index آورده می شود.

سپس مرورگر محتوای زیرین این تابع را به نمایش می گذارد تا به تابع دیگر که باز هم include() است برسد.

این بار این تابع محتوای داخلی فایل پایین برگ را آورده و در فایل اصلی می گذارد. کاربر به هیچ عنوان از این انتقال ها با خبر نمی شود، چیزی که او می بیند همان چیزی است که در صورت داشتن تارنمای ایستا مشاهده می کرد.

فکر می کنیم که اکنون به معنی قرار دادن فایل های متفاوت در داخل یک فایل در طراحی یک تارنما آگاهی پیدا کرده اید.

نکته قابل توجه این است که شما می توانید به تعداد دلخواه از این شیوه استفاده کنید. مثلا فایل دیگری به نام ستون کناری ساخته و در داخل فایل اصلی آنرا وارد کنید. البته باید توجه داشته باشید که برای درست به نمایش در آمدن ستون کناری در صفحه، باید در فایل شیوه CSS، انتخابگرها و کدهای لازم را قرار دهید.

در ادامه دوره تلاش می کنیم که از این فن به صورت های متفاوتی استفاده کنیم. اما چیزی که شما باید در آخر این درس یاد گرفته باشید در ابتدا فهم دلیل این کار و درک چگونگی راحتی و مدیریت بهتر سیستم بعد از استفاده از این فن است.

در درس بعدی ما باز به سراغ ساخت فرم می رویم اما این بار کل روند کار را در داخل یک فایل انجام می دهیم. علاوه بر آن با فن های جدیدی در مورد کار با فرم ها آشنا می شویم که فرم های ما را به صورت حرفه ای تری در خواهند آورد.

درس ششم- ساخت و کار با فرم ها (پیشرفته)

در درس های قبلی با ساخت فرم آشنا شدیم و دیدیم که چطور می شود داده های فرم را به برگه دیگری انتقال داد.

در درس امروز می خواهیم این کار را به طور پیشرفته تری و با انجام این عمل در یک فایل انجام دهیم. منظور این است که گاهی شما نیاز دارید که داده ها را به برگه دیگری بفرستید اما زمانی هم است که بخواهید در یک برگه این داده ها را گرفته و نسبت به آن واکنشی نشان دهید. مهمتر از داشتن یک برگه یا دو برگه، شیوه واکنش به این داده ها براساس خود داده ها است و این هم می تواند تعریف دیگری از داشتن تارنمای پویا باشد.

کار با فرم در یک برگه

منظور از این نوع عملکرد در ساخت فرم، داشتن «فرم» و «بررسی کننده» آن در یک برگه است. این عمل فوایدی در بر خواهد داشت که در ادامه با آنها آشنا می شویم. نکته مهم در داشتن هر دو فاکتور در یک برگه استفاده از یک تابع شرطی برای ارزیابی عملکرد و نحوه کار با آن است. مثلا این تابع انتخاب می کند که داده ها را نمایش دهد یا خود فرم را و این انتخاب را تحت چه شرایطی بگیرد. در زیر به ساختار این تابع شرطی توجه کنید:

```
if ( /*  فرم ارسال شد */){
//  دستورات بررسی کننده فرم
} else {
//  نمایش فرم
}
```

همانطور که مشاهده می کنید این شرطی، دو گزینه دارد. نخست اینکه آیا فرم ارسال شده که اگر ارسال شده باشد آن را بررسی می کند و دستورات داخل قطعه ابتدایی را انجام می دهد؛ دوم در صورت عمل ارسال فرم، به سراغ قطعه دوم رفته و فرم را نمایش می دهد.
داشتن این شرطی برای ساخت فرم در یک برگه همراه بررسی و نمایش داده ها ضروری است.

شرط «فرم ارسال شد»

اگر یادتان باشد در درس های قبلی فرمی داشتیم که داده ها را به برگه دیگر انتقال می داد، اما چه زمانی این انتقال صورت می گیرد؟

در داخل هر فرم «دکمه ای» با نام ارسال یا Submit وجود دارد. «اگر فرم ارسال شد» یعنی اگر کاربر دکمه «ارسال» را بعد از پر کردن فرم فشار داد.

مثلا شرط مقابل را در نظر بگیرید:

```
if ( isset( $_POST['submit'] ) ) {
// دستورات کار با داده های فرم
} else {
// نمایش فرم }
```

تابع isset($_POST['submit']) بررسی می کند که آیا دکمه ارسال یا Submit فشار داده شده است یا نه. از این تابع برای بررسی انتخاب یا عدم انتخاب دکمه ارسال فرم استفاده می شود.

بهتر است با ساخت فرمی این مفهوم را بیشتر بررسی کنیم. این فرم درست مانند فرم های درس های قبلی است با این تفاوت که از این فن جدید در آن استفاده می کنیم.

برای اینکه از داشته هایمان تا به امروز هم استفاده کنیم، ساخت این فرم را در ادامه درس قبلی و در قالب اضافه کردن یک برگه که حامل فرم است انجام می دهیم. فرم ثبت نام در تارنما را حتما قبلا دیده اید؛ بیایید در اینجا یک فرم ثبت نام ساده بسازیم و به تارنمایی که در درس قبل داشتیم اضافه کنیم.

در درس قبل تارنمایی درست کردیم از چند بخش به شرح زیر:

- سربرگ header.html
- پایین برگ footer.html
- صفحه اصلی index.php
- فایل شیوه style.css

حال به این تارنما یک برگه دیگر به نام «ثبت نام» اضافه می کنیم و در آن برگه فرم ثبت نام را نمایش می دهیم. در کنار آن، بررسی کننده فرم ثبت نام را هم در همان برگه می گنجانیم تا پاسخ ها یا هشدارها در همان برگه ثبت نام در بالا یا پایین فرم، به نمایش درآیند.

پوشه جدیدی به نام ثبت نام = register بسازید و محتوای پوشه ای که در درس قبل با نام Including Files ساختیم را در آن رونویسی و بازنویسی کنید.

حال باید در پوشه register یک پوشه به نام include شامل سربرگ، پایین برگ و فایل شیوه، و فایلی به نام index داشته باشید. فایل جدیدی در کنار فایل index به نام register.php بسازید. در ادامه با کدهایی که باید در آن قرار دهیم و همین طور تغییراتی که باید به باقی فایل ها بدهیم، اشاره می کنیم.

برگه ثبت نام

ابتدا کدهای زیر را در داخل این فایل رونویسی و بازنویسی کنید:

```
<?php  $page_title = 'ثبت نام';
include ('includes/header.html'); ?>
<h1>  ثبت نام  </h1>
<?php
  if (isset($_POST['submit'])) {
$name = $_POST['name'] ;
```

```
$username = $_POST['username'];
$password = $_POST['password'];
$email = $_POST['email'];
if ( !empty($name) && !empty($username) && !empty($password) &&
!empty($email) ){
$respond = 'ثبت نام شما با موفقیت انجام شد'; }
else { $respond = ' لطفا تمام گزینه ها را پر کنید'; }
echo $respond ; }?>
<!-- کد برای ساخت فرم - form.html -->
<form action="register.php" method="post">
<fieldset> <legend align="right">لطفا فرم زیر را پر کنید</legend>
<label> نام کامل </label>
<input type="text" name="name" />
<label> شناسه </label> <input type="text" name="username" />
<label> رمز </label>  <input type="text" name="password" />
<label> رایانامه </label> <input type="text" name="email" />
</fieldset>
<input type="submit" name="submit" value="ارسال" class="submit"/>
</form>
<?php include ('includes/footer.html'); ?>
```

شرح کدهای این برگه

اکنون این برگه شامل هر دو بخش فرم و بررسی کننده آن است.

در ابتدا با ساختن متغیری نام هر برگه را در آن ذخیره کردیم تا در زمان وارد کردن سربرگ از آن استفاده کنیم. این کار به ما اجازه می دهد که نام هر برگه را با توجه به عملکرد آن برای تیتر برگه در مرورگر بفرستیم. این متغییر در سربرگ استفاده خواهد شد، تنها نکته این است که باید قبل از وارد کردن سربرگ در این برگه، این متغییر مقداردهی شود.

بعد از آن سربرگ را وارد کردیم که از تابع include استفاده شده است.

حال نوبت شرطی است که بررسی انتخاب یا عدم انتخاب دکمه ارسال از طرف کاربر را برایمان انجام می دهد. در داخل این شرط ابتدا داده های ارسالی را به داخل متغیرهایی می ریزیم که استفاده کردن از آنها را در ادامه برایمان راحت تر کنند.

حال شرط دیگری را در داخل شرط قبلی می گذاریم که از پر بودن تمام متغیرها مطمئن شویم. اگر تمام متغیرها پر بود، این یعنی تمام موارد توسط کاربر نوشته شده است و پیام «ثبت نام شما با موفقیت انجام شد» را در متغیری دیگری بنام respond می گذاریم.

در غیر این صورت در آن متغیر پیام «لطفا تمام گزینه ها را پر کنید» گذاشته می شود تا به کاربر هشدار دهد که باید تمام موارد را بنویسد و بعد از آن متغیر را چاپ می کنیم که بسته به نسبت به پر بودن یا نبودن موارد، مقدارش متفاوت است.

حال نوبت خود فرم است، البته این فرم مانند همان فرم های قبلی است با این تفاوت که مقدار گزینه action به نام همین فایل است. چرا که ما می خواهیم بررسی داده ها توسط همین فرم انجام شود.

در آخر هم فایل بالای برگه را وارد کردیم.

برگه سربرگ header

برای اینکه برگه جدید «ثبت نام» در تارنما قابل دیدن باشد، باید پیوند آن را به قسمت ناوبری تارنما «navigation bar» اضافه کنیم. پس تغییراتی در این فایل می دهیم.

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html"  charset= "utf-8" />
<title> <?php echo $page_title; ?> </title>
<link rel="stylesheet" href="includes/style.css" type="text/css" />
</head>
<body>
<div id="header">
<h1> درسنامه </h1>
<div id="navigation">
<ul>
<li><a href="index.php"> خانه</a></li>
<li><a href="register.php">  ثبت نام </a></li>
</ul>
</div>
</div>
<div id="content">
```

شرح برگه

این برگه درست مانند برگه درس قبلی است البته با چند تغییر.

ابتدا که در <title> <? ;php echo $page_title?> <title/> از متغییری که در قسمت قبلی شرح دادیم استفاده کردیم تا تیتر در مرورگر با توجه به عمل برگه نمایش یابد.

نکته بعدی اضافه کردن پیوند به برگه «ثبت نام» است که به صورت زیر انجام می شود.

```
<li><a href="register.php">  ثبت نام </a></li>
```

این تکه کد یک دکمه دیگر به ناوبری تارنما اضافه می کند که مقصد این گزینه به برگه ثبت نام ختم می شود.

بعد از انجام این تغییرات اگر به تارنما نگاه کنید این دکمه را خواهید دید.

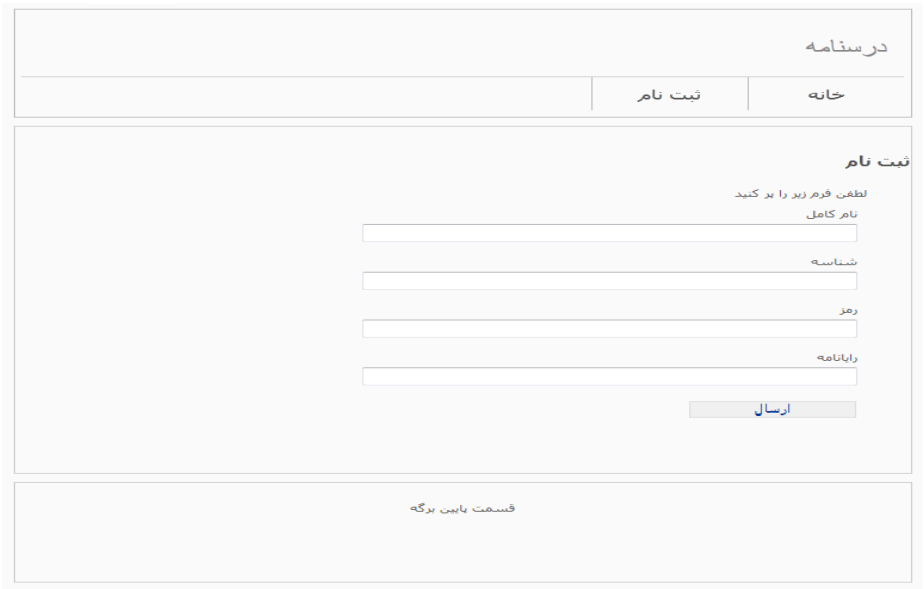
برگه شیوه

برچسب های لازم در این برگه قبلا در درس های دیگر آورده شده است. کافی است که آنها را در این برگه وارد کنید. برای این کار کدهای درس Including Files و کدهای درس Form را وارد این برگه کنید. البته تنها تغییر در انتخابگر ارسال است که در زیر می بینید.

```
.submit {text-align:center; border:1px solid #ddd; color:#039; width:20%;
margin:auto 20px auto auto;}
```

از آنجایی که هدف این درس آموزش سی اس اس نیست و پیش فرض ما هم دانش این شیوه نویسی از طرف کارآموزان است، در اینجا بیشتر به این بخش نمی پردازیم. اما شما می توانید شیوه نمایش را به سلیقه خود تغییر دهید. تنها کافیست که مقادیر انتخابگرها را تغییر دهید.

نتیجه کار باید چیزی شبیه تصویر زیر گردد.



فرم (چسبنده) حفظ کننده مقدار Sticky Form

فرم چسبنده در همان فرم استاندارد در اچ تی ام ال است منتها با این تفاوت که در این فرم قابلیت نگهداری مقادیر نوشته شده در کادر اضافه می شود.

فرض کنید که کاربر تارنمای شما در هنگام ثبت نام دچار خطا شود و سیستم پیغام بازگشت و درست کردن آن مقدار را بدهد. خب کاربر باید دوباره تمام موارد را بنویسد اما با استفاده از فرم چسبنده سیستم می تواند موارد را در خود حفظ کرده و در قسمت های نوشته شده دوباره نمایش دهد. آنگاه در صورت بروز خطا، کاربر فقط قسمت های مشکل دار را دوباره نویسی می کند.

بهتر است این کار را با فرم ساخته شده در همین درس انجام دهیم. به کدهای زیر توجه کنید:

```
<form action="register.php" method="post">
<fieldset>
<legend align="right">لطفا فرم زیر را پر کنید</legend>
<label> نام کامل </label>
<input type="text" name="name" value="<?php if (isset($_POST['name'])){ echo
$_POST['name'];} ?>" />
<label> شناسه </label>
<input type="text" name="username" value="<?php if (isset($_POST['name'])){
echo $_POST['username'];} ?>" />
<label> رمز </label>
<input type="text" name="password" />
<label> رایانامه </label>
<input type="text" name="email" value="<?php if (isset($_POST['name'])){ echo
$_POST['email'];} ?>" />
</fieldset>
<input type="submit" name="submit" value="ارسال" class="submit"/>
</form>
```

این قسمت فرم در برگه ثبت نام است. اگر توجه کنید می بینید که در داخل برچسب های input المان جدیدی با نام value وارد شده است. value در اصل مقدار پیش فرض است که در داخل کادر نمایان می شود. با کمک value و شرطی که داخل آن می گذاریم ما می توانیم در صورت نوشته شدن کلمه ای، آن را دوباره در داخل کادر نمایش دهیم تا کاربر نیاز به دوباره نویسی نداشته باشد.

نکته: معمولا این کار در مورد قسمت «رمز» به دلیل رعایت نکات امنیتی انجام نمی شود.

اجازه بدهید یک نگاه دقیق تری به یکی از این المان ها بی اندازیم.

```
<input type="text" name="name" value="<?php if (isset($_POST['name'])){ echo
$_POST['name'];} ?>" />
```

همانطور که مشاهده می کنید، در این کد که یک کادر متنی اضافه می کند، المان جدیدی اضافه شده است: value

در داخل این المان یک شرط نیز گذاشته شده است:

```
if ( isset($_POST['name']) ) {  echo $_POST['name']; }
```

که می گوید: اگر نوشته ای وارد این کادر شد، آن نوشته را در همین کادر چاپ کن. فکر می کنیم حال به قدرت و کاربری شرطی ها بیشتر اعتقاد پیدا کرده اید، نه؟

درس هفتم- توابع

مفهومی که در این درس با آن آشنا می شویم تابع نام دارد.

ساده ترین تعریفی که از یک تابع می توان گفت این است: تابع مجموعه ای از چند دستور است که برای انجام کار مشخصی نوشته شده است. به طور مثال مجموعه کدهایی را در نظر بگیرید که برای جمع دو عدد و چاپ نتیجه آن نوشته شده اند.

برای نوشتن یک تابع باید از قوانین ویژه ای پیروی کرد. اگر دلیل بوجود آمدن تابع یا دلیل استفاده کردن از آن مشخص شود، مفهوم تابع بهتر فهمیده خواهد شد. پس در این درس با مفهوم آن آغاز می کنیم.

تعریف تابع
تابع تعریفی است برای مجموعه ای از کدهای نوشته شده در کنار هم، به صورت یک قطعه یا بلوک، که با اجرای آن کار مشخصی، در هر زمان که نیاز باشد، انجام می گیرد. هر تابع یک نام مشخص دارد که کدهای نوشته شده تحت آن نام و در داخل بلوک آن تابع قرار می گیرند.

دلیل استفاده از تابع
دلیل اصلی ساخت و استفاده از تابع جلوگیری از تکرار در نوشتن کدها در برنامه است. فرض کنید شما برنامه ای می نویسید که در آن یک عمل مانند جمع دو عدد به صورت مداوم اتفاق می افتد.

یک راه انجام این کار این است که در هر زمان که به این عمل جمع نیاز بود، کدهای عمل کننده به این عملیات را بنویسید. این می شود تکرار یک مجموعه ای از کدها که کار این عملیات را برای شما انجام می دهند. راه دوم این کار نوشتن تابع است، یعنی مجموعه ای از کدها که کار جمع کردن دو عدد را در قالب یک تابع با نام مشخص انجام می دهند.

حال در هر زمان که نیاز به این عمل ریاضی باشد، فقط کافی است که تابع را فراخوانی کنید تا با اجرای دستورات در بلوک خود، نتیجه لازم را به شما بدهد. خب با این کار شما فقط یک بار یک تابع تعریف می کنید، اما هزاران بار می توانید از آنها در طول برنامه استفاده کنید.

طریقه نوشتن تابع
شیوه نوشتن تابع به صورت زیر است:

```
function نام تابع  () {
//  دستورات
}
```

ابتدا کلمه تابع (function) نوشته و بعد از آن نام تابع را می نویسید. دستورات تابع در داخل بلوک آن قرار می گیرند. به تابع زیر توجه کنید:

```
function add(){
$X = 3 + 2  ;
print   $x  ;  // 5
}
```

همان طور که می بینید تابعی با نام add() نوشته شده که در آن اعداد ۲ و ۳ با هم جمع و در متغیر $X ریخته می شوند، در آخر تابع هم متغییر $x چاپ می شود. تا اینجا تابعی نوشته شده است اما این تابع تا زمانی که فراخوانی نشود، به تنهایی کاری نمی کند.

فراخوانی تابع

همانطور که گفته شد تابع این توانایی را به ما می دهد که از دوباره نویسی کدها جلوگیری کنیم و یک بار با نوشتن، آن را در زمان های مورد نیاز استفاده کنیم. قسمت دوم تعریف تابع به استفاده از تابع در زمان های مورد نیاز به هر تعدادی که خواستیم اشاره می کند، این کار را فراخوانی تابع می گوییم. به مثالی که در بالا زدیم دوباره توجه کنید:

```
<?php
function add() {
$X = 3 + 2  ;
print   $x  ;  // 5
}
add(); //  فراخوانی تابع   =>   5
?>
```

فراخوانی تابع با عبارت add() که همان نام تابع است انجام گرفت و با فراخوانی آن کدهای تابع عمل کرده و نتیجه را چاپ می کند. نکته قابل توجه این است که شما می توانید به هر تعدادی که خواستید از این تابع استفاده کنید و یا به تعبیر دیگر آن را فراخوانی کنید و هیچ محدودیتی در آن نیست.

حال فکر می کنیم که به قدرت تابع پی بردید، یعنی نوشتن یک بار و استفاده کردن به تعداد دلخواه در هر زمان که خواستیم.

فرستادن پارامتر به تابع

پارامتر مقداری است که به تابع فرستاده می شود تا تابع در درون بلوک کدهای خود از آن استفاده کند. تابعی که در قسمت قبلی نوشتیم هیچ مقداری را به صورت پارامتر دریافت نمی کرد و فقط با مقادیر تعریف شده در داخل کدها کار می کرد، خروجی این تابع ثابت است و همیشه مقدار 5 را خواهد نوشت.

حال اگر بخواهیم تابع را طوری بنویسیم که مقادیر را به صورت پارامتر از ما بگیرد و بعد عملیات جمع را انجام دهد، ما می توانیم خروجی های متفاوت و با توجه به پارامترهای ارسالی بگیریم. مثلا به تابع زیر دقت کنید:

```
<?php
function add( $x , $y ) {
$z = $x + $y ;
print   $z  ;  // خروجی با توجه به پارامترهای ارسالی است
}
add(۵ , ۶); //  فراخوانی تابع با ارسال دو مقدار به صورت پارامتر
?>
```

در زمان فراخوانی تابع هر مقداری که به صورت پارامتر به تابع ارسال کنیم، حاصل جمع آن دو عدد را خواهیم داشت. این به ما اجازه می دهد که تابعی مستقل از مقادیر و تنها متمرکز به عملیات خاصی داشته باشیم. به طور مثال می توانیم فراخوانی را به شکل های زیر انجام دهیم:

```
add(۱ , ۲); // ۳
add(۳ , ۹);  // ۱۲
add(۱۱ , ۶۸);  // ۷۹
```

نکته: در هنگام فراخوانی تابع حتی می توان پارامترها را به صورت متغیر به تابع ارسال کرد:

```
add(  $a , $b );
```

در این حالت این دو پارامتر می توانند هر عددی را ارسال کنند. خوبی این کار به این است که پارامترها می توانند نتیجه یک عمل دیگر باشند و یا اعدادی باشند که توسط کاربر در کادرهای فرم نوشته شده باشند.

تارنمایی که در درس های قبلی درست کردیم، یادتان هست؟ بیایید برگه دیگری به آن اضافه کنیم و در آن برگه با تابعی که می نویسیم چند عملیات انجام دهیم.

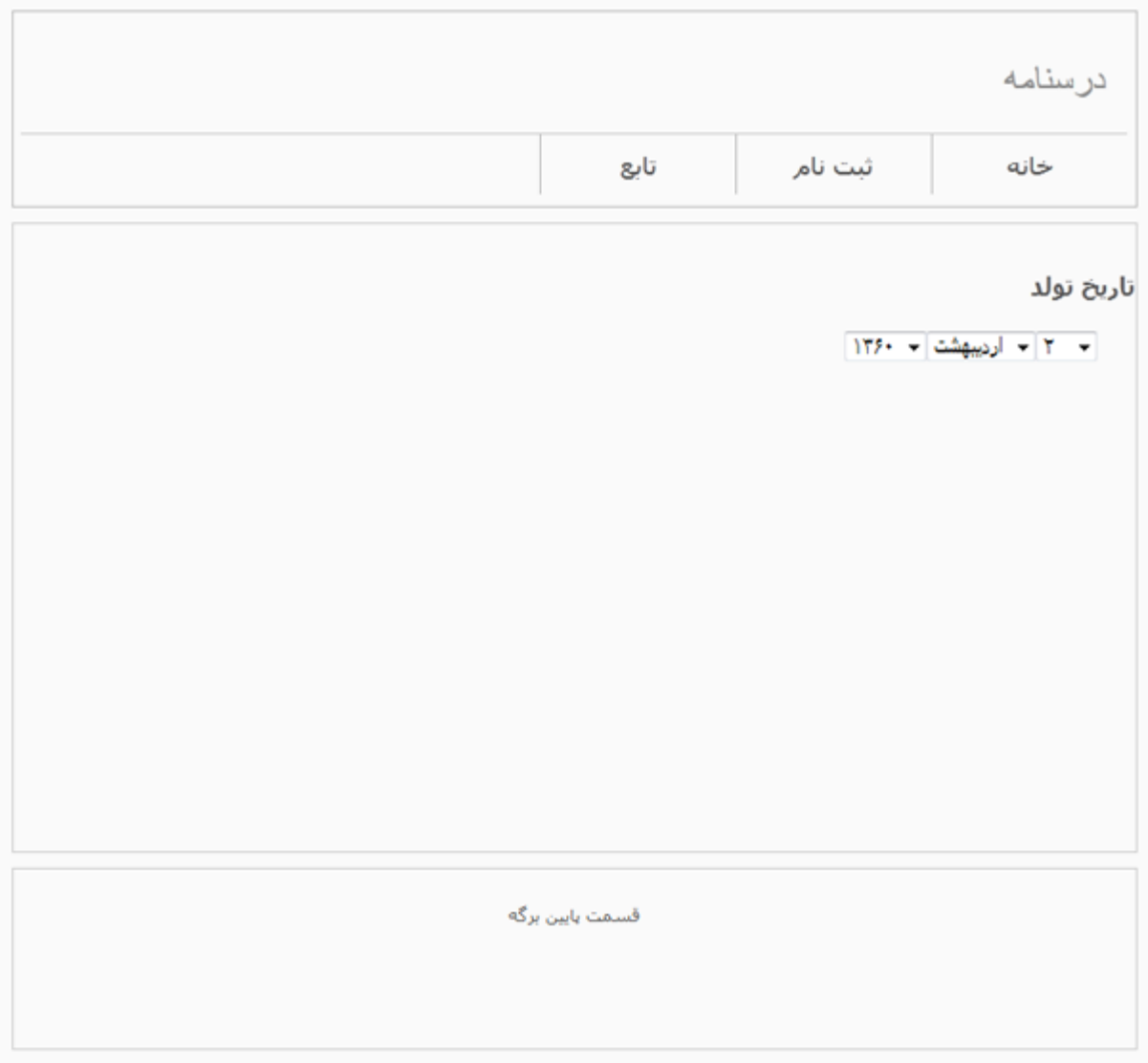
پوشه جدیدی با نام function بسازید و محتوای پوشه register را در آن قرار دهید. این کار برای این انجام می شود که شما در حین یادگیری مفاهیم، قدم به قدم یک تارنما داشته باشید با نسخه های متفاوت که در هر نسخه به مفهومی اشاره شده باشد.

برگه تابع function.php

در داخل پوشه function فایلی با نام function.php بسازید و کدهای زیر را در داخل آن بنویسید:

```
<?php
$page_title = 'تابع';
include ('includes/header.html'); ?>
<h1>  تاریخ تولد </h1>
<?php
/*  انگلیسی به فارسی  */
function convert($string) {
    $persian = array('۰', '۱', '۲', '۳', '۴', '۵', '۶', '۷', '۸', '۹');
    $num = range(0, 9);
    return str_replace($num, $persian,  $string);
}
function calendar() {
    echo '<select name="year">';
    for ($year = 1300; $year <= 1390; $year++) {
            echo "<option value="$year">" . convert($year) . "</option> "; }

    echo '</select>';
    $months = array (1 => ' ,'مهر' ,'شهریور' ,'مرداد' ,'تیر' ,'خرداد' ,'اردیبهشت' ,'فروردین
'اسفند' ,'بهمن' ,'دی' ,'آذر' ,'آبان");
    echo '<select name="month">';
    foreach ($months as $key => $value) {
            echo "<option value="$key">$value</option> "; }

    echo '</select>';
    echo '<select name="day">';
    for ($day = 1; $day <= 31; $day++) {
    echo "<option value="$day">" . convert($day) . "</option> "; }
    echo '</select>';
} // End of the function definition.
echo '<form action="function.php" method="post">';
// Call the function.
calendar();
echo '</form>';
?>
<?php include ('includes/footer.html'); ?>
```

اتفاقی که در اینجا می افتد این است که این برگه برای شما یک کادر تاریخ به همراه روز، ماه و سال می سازد. در درس قبلی ما یک فرم ثبت نام نوشتیم. خوب فرم ثبت نام نیاز به تاریخ زادروز هم دارد:



برای پیچیده نشدن امر ما این قسمت را به طور مجزا گذاشتیم اما در آینده در درس های پایانی به ساخت یک فرم کامل ثبت نام خواهیم پرداخت.

توضیح کدها

با قسمت بالای برگه قبلا آشنا شده ایم، پس یک راست سراغ کدهای داخل می رویم. در ابتدای این برگه تابعی نوشته شده که یک پارامتر را دریافت می کند.

convert($string function) کاری که این تابع می کند این است که با گرفتن یک عبارت اعدادی که به زبان انگلیسی نوشته شده اند را به زبان فارسی تبدیل می کند. این تابع از سه قسمت تشکیل شده است:

```
$persian = array('۰', '۱', '۲', '۳', '۴', '۵', '۶', '۷', '۸', '۹');
```

این خط کد، یک متغیر می سازد که به صورت آرایه تعریف می شود و در داخل آن اعداد را از ۰ تا ۹ قرار می دهد.

$num = range(0, 9);

این دقیقا همان کار بالا را برای اعداد انگلیسی انجام می دهد اما با کمک تابع range() یادتان هست؟ در قسمت آرایه ها در مورد این تابع حرف زدیم.

```
return str_replace($num, $persian,  $string);
```

در این قسمت از یکی از توابع از پیش تعریف شده پی اچ پی استفاده شده است.
تابع str_replace() سه پارامتر را قبول می کند. اعداد انگلیسی، اعداد فارسی و عبارتی که به تابع convert فرستاده شده و بعد اعداد انگلیسی را در عبارت فرستاده شده با اعداد فارسی عوض می کند. در اصل ما تابعی ساخته ایم که از تابع دیگری که از قبل تعریف شده، درون آن استفاده کرده ایم.

بعد از این تابع به تابع دیگری می رسیم با نام calendar(). این تابع هیچ ورودی به عنوان پارامتر دریافت نمی کند و فقط خروجی برای ما چاپ می کند. برای ساختن یک کادر زادروز نیاز به سه قسمت است: روز، ماه و سال؛ که در این تابع به طور مجزا هر کدام تعریف شده اند.

```
echo '<select name="year">';
   for ($year = 1300; $year <= 1390; $year++) {
         echo "<option value="$year">" . convert($year) . "</option> ";    }

echo '</select>';
```

این قسمتی است که با استفاده از حلقه for، بخش سال را برای مان چاپ می کند. این سال ها از ۱۳۰۰ شروع و به ۱۳۹۰ ختم می شود. دقت کنید که این اعداد به زبان انگلیسی هستند پس در زمان چاپ آن از تابع تبدیل کننده convert که قبلا نوشتیم استفاده می کنیم تا کار تبدیل را انجام دهد.

```
$months = array (1 => ' ,'فروردین', 'اردیبهشت', 'خرداد', 'تیر', 'مرداد', 'شهریور', 'مهر
اسفند', 'بهمن', 'دی', 'آذر', 'آبان");
echo '<select name="month">';
   foreach ($months as $key => $value) {
                echo "<option value="$key">$value</option> ";   }
echo '</select>';
```

در این قسمت به کمک تابع array() ماه های ایرانی را در آرایه ای قرار داده و با کمک حلقه آنها را چاپ می کنیم.

```
echo '<select name="day">';
      for ($day = 1; $day <= 31; $day++) {
                  echo "<option value="$day">" . convert($day) . "</option> "; }
echo '</select>';
```

این قسمت شبیه سال است که روز را برایمان چاپ می کند.

خب تا به اینجا تابع نوشته شده و آماده استفاده است، پس با یک فراخوانی از آن استفاده می کنیم:

```
echo '<form action="function.php" method="post">';
calendar();
echo '</form>';
```

نکته: برای ساخت این زاد روز انگار، ما از برچسب های اچ تی ام ال با نام select استفاده کردیم که یک کادر با قابلیت نگهداری مقادیر و به صورت کشویی به پایین درست می کند.

توابع با پارامتر پیش فرض

کار دیگری که در هنگام ساخت یک تابع می توانید انجام دهید، دادن یک مقدار پیش فرض به یک پارامتر یا متغیر است. با این کار در صورتی که هنگام فراخوانی، مقداری را به پارامتر نفرستید، از مقدار پیش فرض استفاده می کند و اگر مقداری را به عنوان پارامتر بفرستید، از مقدار شما استفاده می کند.

```
function say_hello( $name , $message = ' درود '){
echo "$message, $name" ; }
say_hello('پیمان ، درود      // ;('پیمان
echo '<br />';
say_hello('پیمان ، سلام // ;( 'سلام' , 'پیمان
```

در زمان فراخوانی ابتدایی فقط یک پارامتر فرستاده می شود و پیغام همراه مقدار پیش فرض چاپ می شود. اما در فراخوانی دوم که با دو مقدار تابع را می خواند، در زمان چاپ از این دو مقدار استفاده می کند و مقدار پیش فرض را در نظر نمی گیرد.

بازگشت مقدار از تابع

در توابعی که تا به اینجا مثال زدیم، به جز تابع convert()، همه آنها مقدار خروجی را چاپ می کردند. حال شرایطی مانند تابع convert() را در نظر بگیرید که نیاز داریم تابع، عملیاتی انجام دهد و بعد خروجی را به ما باز گرداند تا در جای دیگری از آن استفاده کنیم. برای این کار از عبارت return استفاده می کنیم. به مثال زیر توجه کنید:

```
function add ( $x , $y ){
$z = $x + $y ;
return $z ;
}
$num = add( 2 , 3 ) ;
echo $num * 3 ;
```

اینجا ابتدا مقدارهایی فرستاده شده، بعد در تابع آنها جمع و در درون متغیر دیگری $z ریخته می شوند. در آخر هم تابع مقدار را توسط متغیر باز می گرداند.

بعد از فراخوانی تابع، مقدار بازگردانده شده به داخل متغیر $num ریخته شده و در ادامه از آن استفاده می شود.

نکته ۱: برای اینکه چند مقدار را به عنوان خروجی باز گردانیم، باید از آرایه برای بازگرداندن استفاده کنیم:

```
return array  ($x , $y );
```

نکته ۲: در ساخت توابع باید به این نکته توجه کرد که توابع برعکس متغیرها به بزرگی و کوچکی نام تابع اهمیت نمی دهند. مثلا همه این فراخوانی ها یک تابع را صدا می زنند:

```
add( $x , $y);
ADD( $x , $y);
Add( $x , $y);
```

درس هشتم- پایگاه داده ها (مقدماتی)

در ساخت تارنمای پویا، یکی از پایه های این کار داشتن پایگاه داده ها است.

نحوه کار تارمای پویا همان طور که قبلا هم اشاره کردیم به این روش است که کل سیستم بر پایه دو فاکتور اساسی و تعامل آنها عمل می کند: زبان پی اچ پی و پایگاه داده ها. به این ترتیب که پی اچ پی داده ها را در اختیار گرفته و بعد آنها را با ارتباط با زبان SQL در پایگاه داده ها انبار می کند.

ارتباط پی اچ پی با پایگاه داده ها به دو عمل ثبت داده ها در آن، برای نگهداری، و خواندن داده ها، برای انجام عملیات های تعریف شده، و در آخر نمایش خروجی است. البته پی اچ پی عملیات نمایش را به کمک برچسب های اچ تی ام ال انجام می دهد.

پی اچ پی قابلیت کار با انواع پایگاه داده ها را دارد. به طور مثال MySQL و Oracle و... . در این دوره به خاطر رایگان بودن MySQL و تعامل خوب و راحت آن با پی اچ پی، ما از پایگاه داده های MySQL استفاده می کنیم.

پایگاه داده ها MySQL

این پایگاه داده ها بر طبق تارنمای MySQL، مشهورترین پایگاه داده های منبع باز (Open Source) است که عموما توسط پی اچ پی استفاده می شود. شیوه نگهداری داده ها در

MySQL به این طریق است که داده ها در آبجکت های (object) به نام جدول انبار می شوند. جدول یک مجموعه ای از داده های مرتبط با هم را در سطرها و ستون ها نگهداری می کند.

جدول Table

تعریف جدول همان است که در ذهن دارید؛ یک مستطیل که دارای سطر و ستون بسیاری است. هر پایگاه داده ها برای انبار کردن داده ها نیاز به ساخت جدول دارد که هر جدول با نام مشخصی باید ساخته شود. بعد از نامگذاری و ساخت جدول باید ستون ها را که معرف نام و نوع هر داده هستند، مشخص کرد. به جدول زیر توجه کنید:

ID	FirstName	LastName	UserName	Email
1	سامان	تهرانی	Saman2000	saman@yahoo.com
2	پویا	اشراقی	Poya1360	poya@yahoo.com

این یک نمونه از جدول همراه دو سطر و چهار ستون است. به سطر ابتدایی توجه کنید، نام هر ستون در بالای آن آورده شده است. در هر سطر داده های مربوط به یک کاربر نوشته شده است که به آن اصطلاحا رکورد (Record) گفته می شود و به هر خانه (Cell) از جدول، رشته (Field) گفته می شود که تک تک داده ها را در بر می گیرد.

پی اچ پی برای نوشتن و خواندن از یک پایگاه داده ها، نیاز به دانستن نام پایگاه داده ها، نام جدول و در آخر نام و نوع ستون دارد. داده ها در جدول به صورت رکوردوار و در قالب رشته ها قرار می گیرند.

نام گذاری پایگاه داده ها، جدول، سطر و ستون ها باید طبق قوانین از قبل نوشته شده صورت گیرد. در زمان نام گذاری متغیرها هم یک چنین قوانینی وجود داشت، به خاطر می آورید؟

- باید تنها از حروف، اعداد و علامت خط زیرین (_) استفاده شود.
- نباید از نام های از قبل تعریف شده در پایگاه داده ها استفاده کرد.
- بزرگی یا کوچکی حروف برای نامگذاری مورد توجه است و تفاوت ایجاد می کند.
- هر نام نمی تواند بیشتر از ۶۴ کاراکتر باشد.
- باید در محیط و حوزه خود منحصر به فرد باشد؛ دو جدول با یک نام نمی توانند ساخته شوند.

نکته: در هنگام ساختن جدول شما باید تعیین کنید که چه نوع داده ای در هر ستون قرار می گیرد؛ این کار با تعیین نوع داده ها در زمان معرفی ستون ها انجام می شود که این نوع می تواند: متن، عدد و یا زمان باشد. البته هر کدام هم از اندازه ها و متغیرهای بسیاری تشکیل می شوند.

بهتر آن است که قبل از ساخت پایگاه داده ها و متعلق هایش، یکبار برای خود بر روی کاغذی این جدول ها و سطر و ستون ها را به همراه نام و نوع آنها مشخص کنید، تا در زمان ساخت به مشکلی بر نخورید.

انواع داده ها

در پایگاه داده ها هم مانند تمام زبان های برنامه نویسی باید نوع داده مشخص باشد. این کار با تعیین نوع ستون انجام می شود. داده ها در MySQL به انواع زیر تقسیم می شوند:

- CHAR: این نوعی است به طور ثابت که رشته ای را از ۰ تا ۲۵۵ کاراکتر در خود نگهداری می کند.

- VARCHAR: درست مانند CHAR با این تفاوت که از ۰ تا ۶۵۵۳۵ کاراکتر را نگهداری می کند. فرق دیگر بین CHAR و VARCHAR در ثابت و متغیر بودن طول نوع است. VARCHAR دارای طول متغیر است و اندازه آن با توجه به طول داده تغیر می کند.

- TEXT: برای نگهداری متن از آن استفاده می شود. البته خود TEXT براساس طول نگهداری به انواع TINYTEXT و TEXT و MEDIUMTEXT و LONGTEXT تقسیم می شود.

- INT: برای نگهداری اعداد صحیح از آن استفاده می شود که مانند TEXT این نوع هم براساس طول به انواع مختلف تقسیم می شود.

- DATE: برای حفظ تاریخ در پایگاه کاربرد دارد.

به منظور آشنایی هر چه بیشتر با انواع داده ها می توانید به جدول زیر مراجعه کنید.

MySQL Data Types

Type	Size	Description
CHAR[Length]	*Length* bytes	A fixed-length field from 0 to 255 characters long
VARCHAR[Length]	String length + 1 or 2 bytes	A variable-length field from 0 to 65,535 characters long
TINYTEXT	String length + 1 bytes	A string with a maximum length of 255 characters
TEXT	String length + 2 bytes	A string with a maximum length of 65,535 characters
MEDIUMTEXT	String length + 3 bytes	A string with a maximum length of 16,777,215 characters
LONGTEXT	String length + 4 bytes	A string with a maximum length of 4,294,967,295 characters
TINYINT[Length]	1 byte	Range of –128 to 127 or 0 to 255 unsigned
SMALLINT[Length]	2 bytes	Range of –32,768 to 32,767 or 0 to 65,535 unsigned
MEDIUMINT[Length]	3 bytes	Range of –8,388,608 to 8,388,607 or 0 to 16,777,215 unsigned
INT[Length]	4 bytes	Range of –2,147,483,648 to 2,147,483,647 or 0 to 4,294,967,295 unsigned
BIGINT[Length]	8 bytes	Range of –9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 or 0 to 18,446,744,073,709,551,615 unsigned
FLOAT[Length, Decimals]	4 bytes	A small number with a floating decimal point
DOUBLE[Length, Decimals]	8 bytes	A large number with a floating decimal point
DECIMAL[Length, Decimals]	Length + 1 or 2 bytes	A DOUBLE stored as a string, allowing for a fixed decimal point
DATE	3 bytes	In the format of YYYY-MM-DD
DATETIME	8 bytes	In the format of YYYY-MM-DD HH:MM:SS
TIMESTAMP	4 bytes	In the format of YYYYMMDDHHMMSS; acceptable range ends in the year 2037
TIME	3 bytes	In the format of HH:MM:SS
ENUM	1 or 2 bytes	Short for *enumeration*, which means that each column can have one of several possible values
SET	1, 2, 3, 4, or 8 bytes	Like ENUM except that each column can have more than one of several possible values

منبع: PHP 6 and MySQL 5 for Dynamic Web Sites: Visual QuickPro Guide نوشته Larry Ullman

اتصال به پایگاه داده ها

برای اتصال به پایگاه و کار با آن دو روش موجود است:

روش اول: اتصال با کاربر پایگاه توسط Command Prompt یا خط فرمان که مسیر دسترسی به آن در سیستم عامل ویندوز به شرح زیر است:

```
Start > Programs > Accessories > Command Prompt
```

راه کوتاه تری هم وجود دارد که از طریق گرفتن دکمه «پنجره» در صفحه کلید و فشار دادن حرف R است. بعد در داخل کادری که باز می شود حروف cmd را بنویسید.

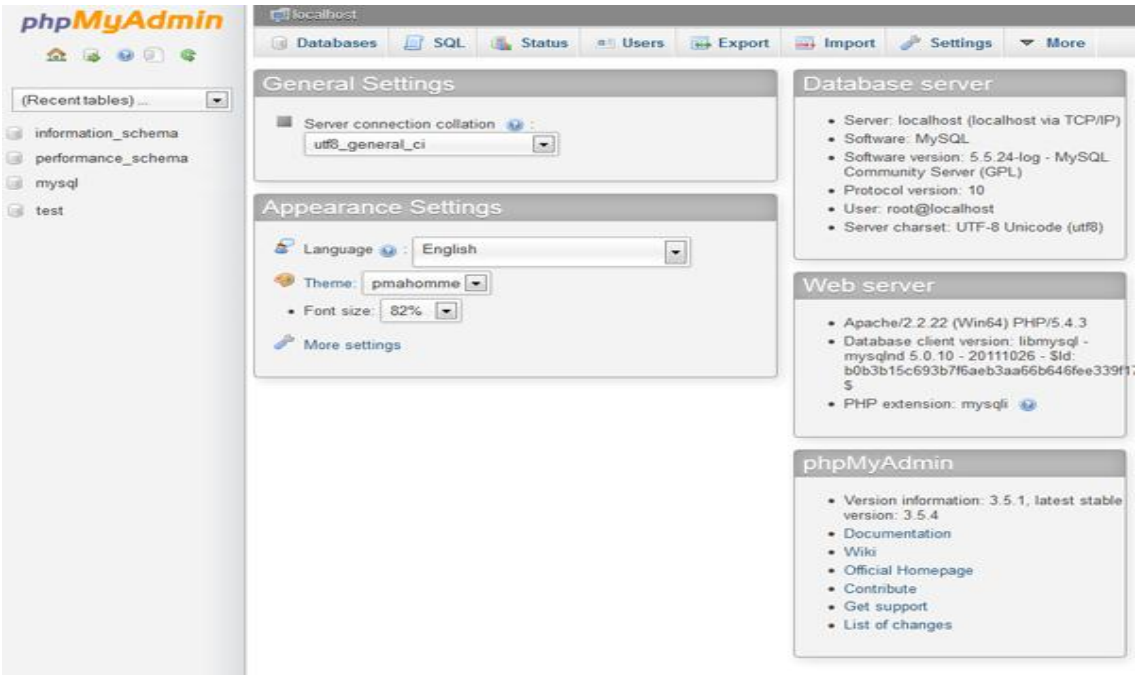
به محض فشردن دکمه «ورود»، CMD ظاهر می شود. حال در داخل این کادر مشکی رنگ باید آدرس محل نصب پایگاه داده ها را بدهید. البته در این حالت باید از دستورات تعریف شده استفاده کنید که کمی کار را مشکل می کند و فعلا نیازی به آن نیست؛ اما اگر می خواهید به طور حرفه ای این کار را انجام دهید، باید در آینده با آن بیشتر آشنا شوید.

در درس بعدی که مربوط به کار حرفه ای تر با پایگاه داده ها است، تمرکز کار به روش استفاده از CMD خواهد بود، فعلا در این درس به معرفی فضای پایگاه بسنده می کنیم.

روش دوم: راه ساده تر برای دسترسی، از طریق نوشتن نشانی در مرورگر و دسترسی به پایگاه توسط نرم افزار phpMyAdmin است که با نصب پایگاه یا WampServer نصب می شود. به طور مثال در حالت نصب WampServer در رایانه خود با آدرس زیر به پایگاه دسترسی پیدا می کنید:

```
http://localhost/phpmyadmin/
```

چیزی که مشاهده می کنید، یک صفحه کار با پایگاه به صورت GUI(Graphical User Interface) یا رابط کاربر گرافیکی است. این صفحه کار با پایگاه را بسیار ساده می کند که در ابتدای کار انتخاب خوبی است. شما با چیزی مانند تصویری زیر مواجهه می شوید:

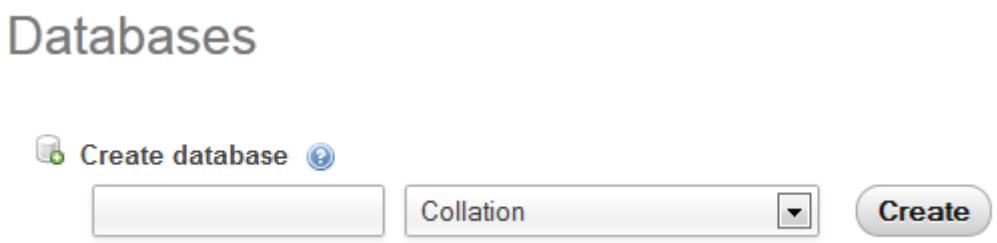


ستون چپ محل نمایش پایگاه های ساخته شده است و در نوار ابزار بالای صفحه می توانید انواع ابزار مورد نیاز برای ساخت و کار با پایگاه را مشاهده کنید. کار با این صفحه بسیار آسان است، تنها باید مدتی وقت صرف دیدن قسمت های مختلف و آشنایی با آنها بکنید.

نکته: در زمان کار بر روی اینترنت، سرویس دهنده شما این امکان دسترسی را برای شما فراهم می کند، کافیست که به قسمت پایگاه داده ها در تارنمای سرویس دهنده مراجعه فرمایید.

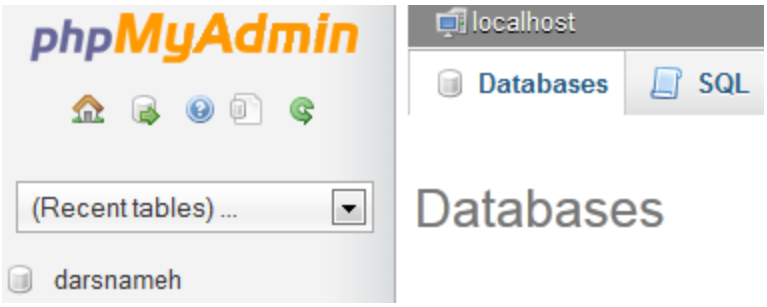
ساخت پایگاه درphpMyAdmin:

۱. برای ساخت پایگاه در این فضا کافی است که بر روی دکمه Databases واقع در بالای صفحه در نوار ابزار سمت چپ کلیک کنید. در صفحه ای که باز می شود به سراغ قسمت Create Database رفته و نام پایگاه خود را وارد کنید.

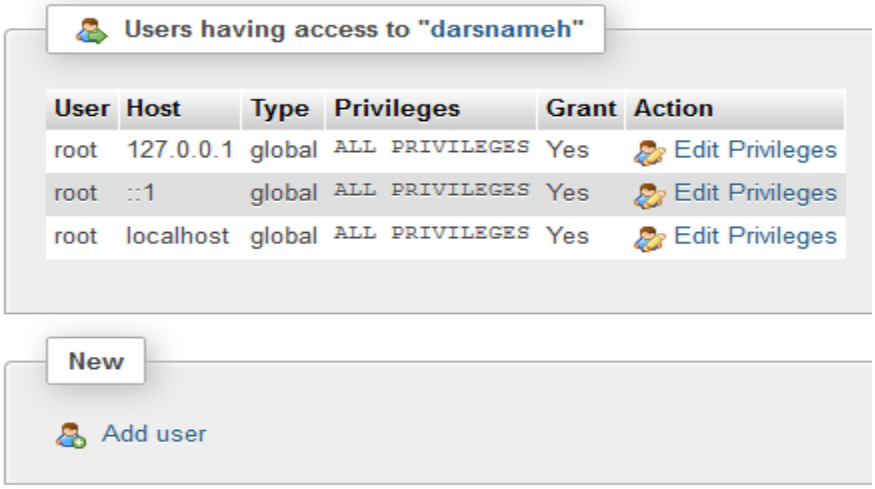


بخش ابتدایی مربوط به نام پایگاه و بخش دوم برای تعیین زبان نوشتداری است. به طور مثال برای ما فارسی زبان ها از utf8_general_ci استفاده می شود. این مربوط به مجموعه دستورات و مقررات برای تعیین شیوه و زبان نوشتن متن در پایگاه است. حال نام پایگاه را نوشته و قسمت Collation یا تطبیق را utf8_general_ci انتخاب کنید. به صورت زیر:

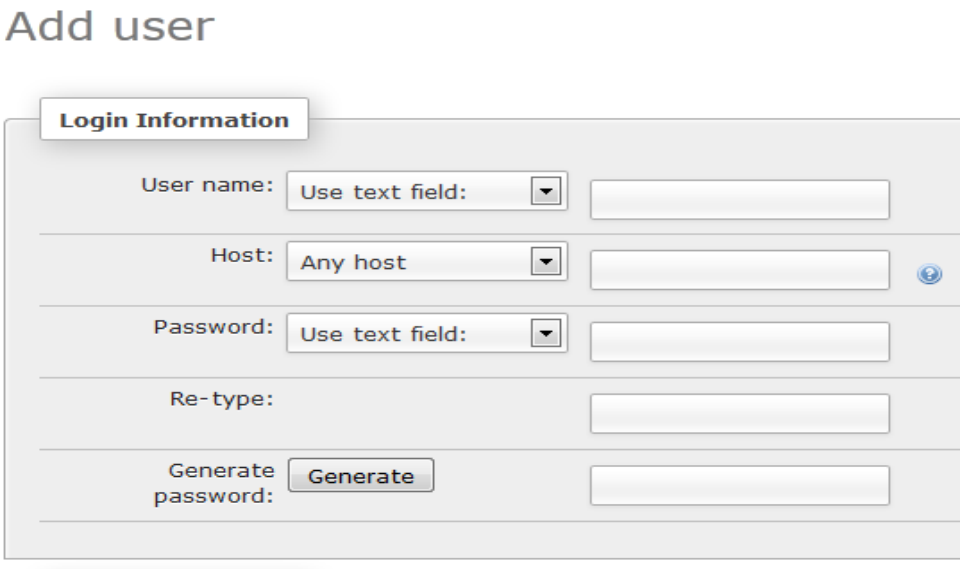
با فشردن دکمه Create، پایگاه داده ها ساخته شده و نشانه آن در قسمت چپ اضافه می شود.



۲. مرحله بعدی ساخت یک شناسه کاربری برای اتصال به پایگاه داده های ساخته شده است. البته شما می توانید از کاربر root استفاده کنید، اما بهتر آن است که نشانه کاربری برای خود بسازید. برای این کار بر روی نشان پایگاه ساخته شده در ستون چپ (اینجا darsnameh) کلیک کنید و از نوار ابزار بالای صفحه گزینه Privileges را انتخاب کنید.

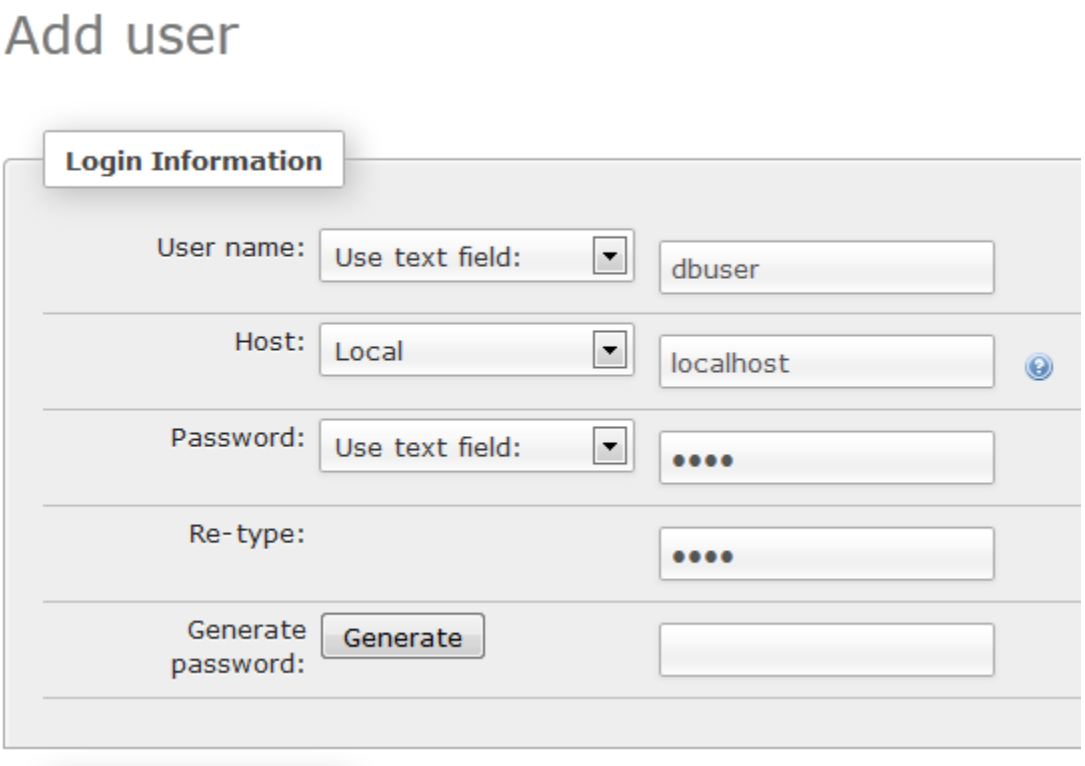


روی دکمه Add user کلیک کنید تا کادر اضافه کردن کاربر نمایان شود.

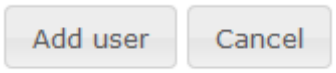


در این کادر سه مورد وجود دارد که باید نوشته شوند:

- نام کاربری
- محل اتصال کاربر
- رمز: دو بار



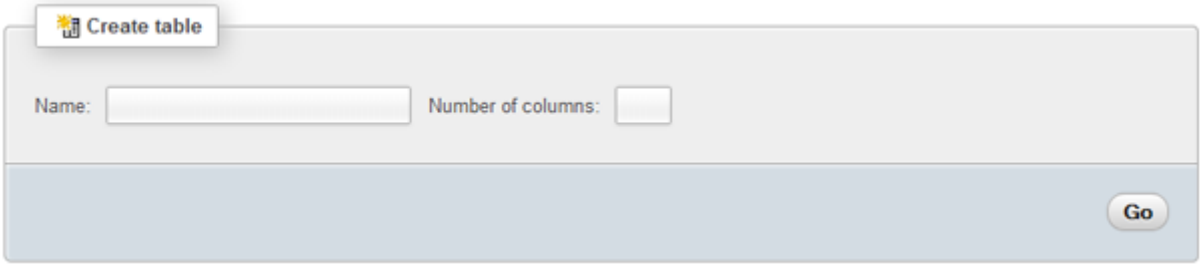
بعد از پر کردن موارد خواسته شده، دکمه Add User در قسمت پایین کادر را بزنید.



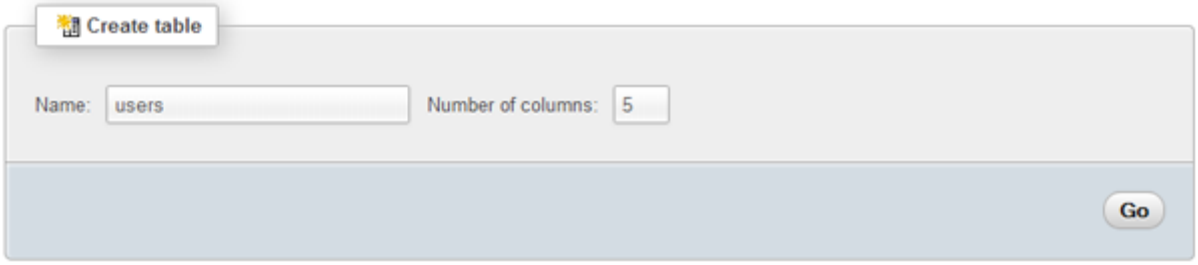
حال پایگاه داده ها به همراه کاربر ساخته شده و آماده اتصال است.

۳. در این قسمت باید جدول مورد نظر خود را بسازید. جدولی که در بخش توضیح جدول ساختیم یادتان هست؟ حال می خواهیم همان را در پایگاه داده ها پیاده کنیم.

بر روی گزینه Structure در سمت چپ نوار ابزار کلیک کنید تا به قسمت ساخت جدول بروید.



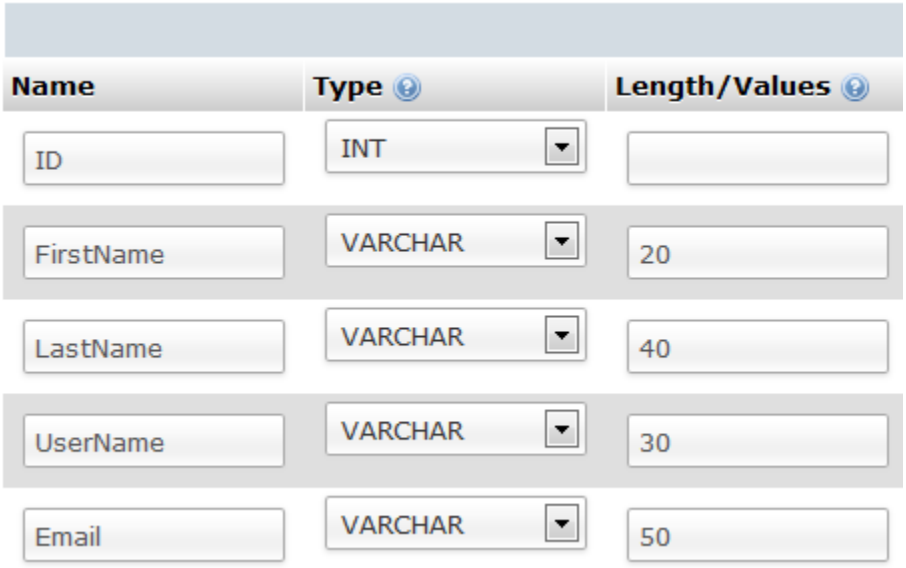
نام جدول و تعداد ستون ها را وارد کنید. با توجه به جدول معرفی شده، ما ۵ ستون نیاز داریم.



نام جدول را users و ۵ ستون انتخاب می کنیم و دکمه Go را فشار می دهیم.

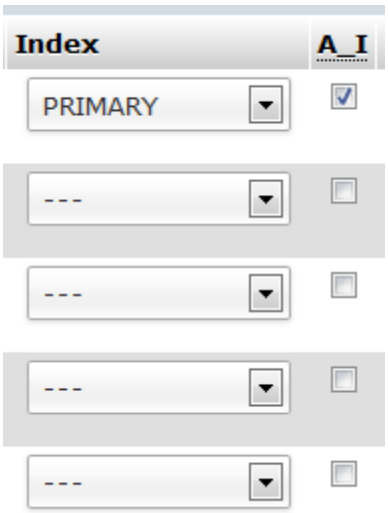
ID	FirstName	LastName	UserName	Email
1	سامان	تهرانی	Saman2000	saman@yahoo.com
2	پویا	اشراقی	Poya1360	poya@yahoo.com

با توجه به این جدول، ستون های داخل کادر را پر می کنیم. مواردی که باید نوشته شوند شامل Name و Type و Length و Attributes و Index هستند. البته Attributes و Index را فقط برای ستون ابتدایی یعنی ID می نویسیم.

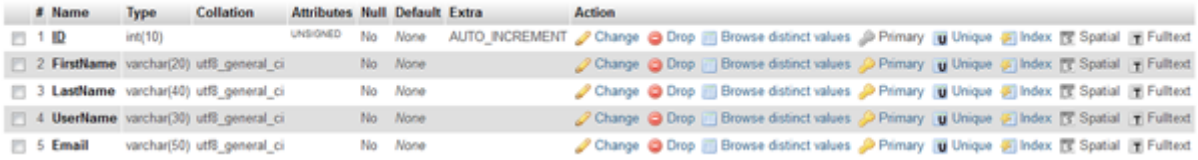


- Attributes: این انتخاب به ما کمک می کند که به پایگاه داده ها اطلاع دهیم ستون مورد توجه باید منحصر به فرد باشد و یا برای انجام عملیات هایی مانند جستجو مناسب است. برای این کار از گزینه Primary استفاده می کنیم، این بدان معناست که در این ستون هر رکورد به صورت منحصر به فرد دارای یک شماره است. در درس بعدی در مورد این گزینه بیشتر بحث خواهیم کرد.

- A_I یا Auto Increment: برای شمارنده ها مانند ستون ID که به صورت منحصر به فرد تعیین می شوند، استفاده می شود. این کار باعث ایجاد شماره از ۱ تا... برای هر رکورد به صورت یگانه می شود.



در درس بعدی با دیگر انتخاب ها بیشتر آشنا می شویم. بعد از انتخاب کلید Save، جدول ما ساخته می شود و همچنین نماد آن در ستون چپ به صورت نام جدول ظاهر می گردد. با انتخاب نام جدول می توانید آن را مشاهده کنید که البته هنوز خالی است.



#	Name	Type	Collation	Attributes	Null	Default	Extra	Action
1	ID	int(10)		UNSIGNED	No	None	AUTO_INCREMENT	Change Drop Browse distinct values Primary Unique Index Spatial Fulltext
2	FirstName	varchar(20)	utf8_general_ci		No	None		Change Drop Browse distinct values Primary Unique Index Spatial Fulltext
3	LastName	varchar(40)	utf8_general_ci		No	None		Change Drop Browse distinct values Primary Unique Index Spatial Fulltext
4	UserName	varchar(30)	utf8_general_ci		No	None		Change Drop Browse distinct values Primary Unique Index Spatial Fulltext
5	Email	varchar(50)	utf8_general_ci		No	None		Change Drop Browse distinct values Primary Unique Index Spatial Fulltext

در اینجا کار ساخت پایگاه داده ها به همراه جدول به اتمام می رسد. حالا زمان آن رسیده که با پی اچ پی به آن متصل شده و داده های خودمان را در آن انبار کنیم.

اما قبل از این کار بهتر است که طریقه دیگر که همان کار با CMD است را یاد بگیریم، چرا که دانستن آن برایمان ضروری و در نهایت بهتر است.

درس نهم- پایگاه داده ها (پیشرفته)

در درس قبل با پایگاه داده ها و مفاهیم مهم در این زمینه آشنایی پیدا کردیم. یاد گرفتیم که چطور می شود پایگاهی برای انبار کردن داده ها با استفاده از phpMyAdmin ساخت. با توجه به آسانی کار در محیط این برنامه که به صورت «رابط کاربر گرافیکی» است، ابتدا از این ابزار برای بیان چگونگی ساخت پایگاه و متعلقاتش بهره بردیم، اما نکته در این است که در زمان نوشتن برنامه ای به زبان پی اچ پی و اتصال به پایگاه از داخل برنامه، باید از کدهای زبان SQL استفاده کنیم.

پس این یک ضرورت است که با این زبان هم آشنا باشیم؛ البته کار بسیار راحتی است و دستورات بسیار محدود و ساده. در این درس با توجه به آشنایی مقدماتی درس قبلی با برنامه MySQL که از زبان SQL بهره می‌برد، تلاش می کنیم که زبان SQL را شرح دهیم.

MySQL و SQL

قبل از ورود به بحث لازم دیدیم یک بار دیگر در مورد این دو مفهوم توضیح کوتاهی بدهیم، باشد که تعریف واضع تر باعث فهم بهتر هر کدام از این دو گردد.

SQL یا Structured Query Language

این یک زبان برنامه نویسی ویژه ای است که به منظور مدیریت داده ها در سیستم های مدیریتی پایگاه داده های ارتباطی (RDBSM) نوشته شده است. سیستم های مدیریتی پایگاه داده های ارتباطی، به سیستم هایی گفته می شود که داده ها در داخل پایگاه براساس ارتباط بین شان انبار می شوند و در زمان خواندن از آنها، می توان با توجه به ارتباط داخلی، داده ها را فراخوانی کرد.

MySQL
این یک نرم افزار است که از زبان SQL استفاده می کند و ما معمولا آن را بر روی رایانه خود نصب می کنیم. می توان گفت که MySQL موتوری است که دستورات SQL را اجرا می کند. البته نرم‌افزارهای دیگری هم هستند که از زبان SQL برای ساخت و کار پایگاه داده ها استفاده می کنند. به طور مثال شرکت مایکروسافت نرم افزاری بنام Microsoft SQL Server دارد و یا شرکت Oracle نرم افزاری با همین نام دارد که همه آنها از زبان SQL در نرم افزار خود استفاده می کنند.

اگر هنوز تفاوت این دو برای تان روشن نیست، اشکالی ندارد؛ کمی صبر داشته باشید. همراه کار با پایگاه داده ها مفهوم مطالب بالا برای شما روشن تر خواهند شد.

دسترسی به MySQL Server
برای دسترسی به سرویس دهنده پایگاه داده ها و ساخت جدول و... ما نیاز به یک سطح دسترسی کاربری داریم. در درس قبل انجام این کار را توسط محیط phpMyAdmin دیدیم، اما در این درس این کار را توسط CMD انجام می دهیم.

کاربر MySQL
این کاربر هنگام نصب نرم‌افزار همراه بقیه موارد نصب می شود و برای استفاده از آن باید از محیط خط دستور یا همان Command Prompt Interface استفاده کرد. در شروع باید از طریق مسیر زیر (در سیستم عامل ویندوز) به CMD مراجعه کنید:

```
Start\ All Programs\Accessories\Command Prompt
```

راه دیگر هم دسترسی مستقیم به CMD از طریق گرفتن دکمه «پنجره» در صفحه کلید و انتخاب حرف R است. بعد در داخل کادری که باز می‌شود حروف cmd را بنویسید و دکمه Enter را بزنید.

هر دوی این ها به پنجره سیاه رنگ CMD در ویندوز ختم می‌شوند. در لینوکس نام این ابزار را Terminal می نامند که به خاطر گوناگونی نمای صفحه در نسخه های مختلف لینوکس هر کدام راه دسترسی خود را دارد.

نکته: برای کسانی که از Wamp Server استفاده می کنند، یک راه بسیار ساده وجود دارد. باید بر روی نشان برنامه در قسمت راست پایین صفحه کلیک کنید، بعد نشانگر را بر روی MySQL برده تا گزینه MySQL Console را مشاهده کنید. با این انتخاب مستقیما به پایگاه وصل می شوید.

حال فرض می کنیم که می خواهیم از داخل CMD وارد پایگاه شویم. پایگاه شما در داخل پوشه ای است که Wamp Server در آن نصب شده، پس باید به داخل آن پوشه رفت و دنبال پوشه ای به نام bin گشت. ما این برنامه را در دایرکتوری I نصب کرده ایم، پس آدرس ما به صورت زیر است:

```
I:\wamp\bin\mysql\mysql5.5.24\bin
```

این مسیر پوشه bin در رایانه ماست که آدرس ابتدایی به محل نصب بستگی دارد. به طور مثال اگر شما Wamp را در دایرکتوری C نصب کنید آدرس می شود:

```
C:\wamp\bin\mysql\mysql5.5.24\bin
```

این مسیر را باید از داخل cmd رفت. فرض کنید اکنون cmd ما را در داخل دایرکتوری C نشان می دهد.

لطفا به طرز انجام این کار دقت کنید:

```
C:\> I:     //  داخل دایرکتوری شو
I:\>    //  حال  اینجا هستیم
I:\> cd wamp\bin\mysql\mysql5.5.24\bin   //  ورود به آدرس جدید
I:\wamp\bin\mysql\mysql5.5.24\bin>    // bin  داخل پوشه
I:\wamp\bin\mysql\mysql5.5.24\bin> mysql -u root
```

این دستور ما را به پایگاه داده ها با شناسه کاربری root بدون هیچ روزی وصل می کند. اما این کار به دلایل امنیتی درست نیست و بهتر آن است که برای شناسه root یک رمز برگزینیم:

```
I:\wamp\bin\mysql\mysql5.5.24\bin> mysqladmin -u root password "darsnameh"
```

حال کاربر ریشه پایگاه رمز "darsnameh" را داراست. برای ورود باید به شیوه زیر عمل کرد:

```
>mysql -u root -p برای جلوگیری از تکرار، از نوشتن آدرس دیگر خودداری می کنیم
Enter Password: darsnameh
mysql>
```

این علامت ورود در پایگاه است. در حال حاضر به پایگاه داده ها وصل شدیم. البته برای ساختن کاربرهای جدید هم می توانید به طریق زیر عمل کنید:

```
>CREATE USER 'test' IDENTIFIED BY 'your_password';
```

نکته: در صورت تغییر رمز کاربر ریشه به هنگام باز کردن صفحه phpMyAdmin، به یک خطا می خورید. برای رفع این مشکل باید رمز را در داخل فایل تنظیمات وارد کنید. ابتدا به آدرس زیر رفته:

```
wamp\apps\phpmyadmin
```

و فایل config.inc.php را توسط ویرایشگر باز نموده و تغییرات زیر را انجام دهید:

```
$cfg['Servers'][$i]['verbose'] = 'localhost';
$cfg['Servers'][$i]['host'] = 'localhost';
$cfg['Servers'][$i]['port'] = '3306';
$cfg['Servers'][$i]['socket'] = '';
```

```
$cfg['Servers'][$i]['connect_type'] = 'tcp';
$cfg['Servers'][$i]['extension'] = 'mysqli';
$cfg['Servers'][$i]['auth_type'] = 'config';
$cfg['Servers'][$i]['user'] = 'root';
$cfg['Servers'][$i]['password'] = 'dars';
$cfg['Servers'][$i]['AllowNoPassword'] = true;
```

لطفا تغییرات را بر اساس نام کاربری و رمز آن وارد کنید. موارد مهمی که اغلب نیاز به تغییر دارند به شرح زیر هستند:

```
$cfg['Servers'][$i]['port'] = '3306';
$cfg['Servers'][$i]['user'] = 'root';
$cfg['Servers'][$i]['password'] = 'dars';
```

حال با دوباره راه اندازی برنامه همه چیز درست می شود. اگر مطابق چیز گفته شده عمل کنید، به مشکل دیگری بر نخواهید خورد. در صورت داشتن خطای دیگر می توانید با جستجوی متن خطا در گوگل دلیل آن را هم پیدا کنید.

خب حال ما به سرویس دهنده پایگاه دسترسی داریم. برای دیدن باقی پایگاه ها به شکل زیر عمل می کنیم:

```
> show datanases;
```

این کار لیستی از تمام پایگاه داده‌ها را برای مان نمایش می دهد.

اجازه بدهید با پاک کردن پایگاه هایی که دیروز ساختیم شروع کنیم:

```
> DROP DATABASE darsnameh;
```

حال فرض کنید می خواهیم پایگاه داده جدیدی ساخته و در آن جدول درس قبلی را هم اضافه کنیم:

```
> CREATE DATABASE darsnameh ;
```

قبل از ساخت جدول ابتدا باید مشخص کنیم که در کدام پایگاه می خواهیم جدول اضافه کنیم:

```
> USE darsnameh ;
Database changed //  پیام تغییر پایگاه
```

خب تا به اینجا پایگاهی با نام darsnameh ساخته شد و به آن متصل شدیم، مرحله بعدی ساخت جدول و ستون هایش است. جدول درس قبل یادتان هست؟

ID	FirstName	LastName	UserName	Email
1	سامان	تهرانی	Saman2000	saman@yahoo.com
2	پویا	اشراقی	Poya1360	poya@yahoo.com

ساختار درست کردن جدول به شکل زیر است:

```
CREATE  TABLE   table_name
(
column_name1    data_type,
column_name2    data_type,
column_name3    data_type,
....
);
```

نام جدول که users است و دارای ۵ ستون. ابتدا باید جدول و ستون هایش را ساخت، وارد کردن داده ها باید در مرحله دیگری صورت بپذیرد:

```
> CREATE TABLE  users
(
ID INT UNSIGNED NOT NULL AUTO_INCREMENT,
FirstName VARCHAR(20) NOT NULL,
LastName VARCHAR(40) NOT NULL,
Email VARCHAR(50),
PRIMARY KEY (ID)
) ;
```

- INT: نوع صحیح، نوع داده هایی که در این ستون قرار می گیرند.
- UNSIGNED: این داده ها بدون علامت و همه مثبت هستند.
- NOT NULL: این خانه نمی تواند خالی بماند و حتما باید مقدار داشته باشد.
- AUTO_INCREMENT: مقدار خانه ها را به طور اتوماتیک از 1 به بعد وارد می کند.
- VARCHAR(20): نوع رشته ای، نوع داده هایی که در این ستون باید وارد شوند.
- PRIMARY KEY (ID): انتخاب ستون ID به عنوان ستون کلیدی.
- KEY PRIMARRY: وقتی به ستونی این المان را اضافه می کنید، آن ستون تبدیل به ستون منحصر به فرد می شود و رکوردهایی که می خواهید وارد جدول کنید را منحصر به فرد می‌کند.

هر جدول باید یک کلید اصلی داشته باشد و هر جدول تنها می‌تواند یک کلید اصلی داشته باشد. ستون کلید اصلی هرگز نمی‌تواند مقدار NULL داشته باشد. NULL مقداری است به معنی هیچ اما با ستون خالی متفاوت است. مقداری که وارد ستون کلید اصلی می شود باید منحصر به فرد باشد.

اینها ویژگی‌های ستون کلید اصلی دار هستند که باعث ایجاد قابلیت هایی در مورد نوشتن، خواندن و جستجو در جدول می شوند. در ادامه دوره با استفاده از این ویژگی، مفهوم آن بیشتر روشن خواهد شد.

در بخش قبل ما یک پایگاه داده ها درست کردیم و در درون آن هم جدولی ساختیم. حال برای دیدن جدول کافی است که دستور زیر را وارد کنید:

```
> SHOW TABLES;
mysql> SHOW TABLES;
+--------------------+
| Tables_in_darsnameh |
+--------------------+
| users                |
+--------------------+
1 row in set (0.00 sec)
```

خروجی دستور به صورتی که مشاهده می کنید می شود. برای دیدن جزییات جدول دستور زیر را بزنید:

```
EXPLAIN users;
و یا
DESCRIBE users;
```

هر دوی این دستورات مشخصات جدول را برای شما نشان می دهند.

نکته: فراموش نکنید که قبل از هر کاری با پایگاه داده ها، باید ابتدا آن را انتخاب کنید:

```
> USE darsnameh;
```

خب تا به اینجا ما دو شیوه ساخت پایگاه داده ها و نحوه کار با آن را یاد گرفتیم. مرحله بعدی انبار کردن یا نوشتن داده ها در پایگاه است. برای نوشتن داده ای در پایگاه می توان به دو صورت عمل کرد.

۱- استفاده از محیط و دستورات SQL: با این شیوه می توان به طور مستقیم داده ها را وارد جدول پایگاه نمود.

۲- استفاده از پی اچ پی برای ثبت داده ها در پایگاه: در داخل کدهای پی اچ پی از دستورات SQL استفاده کرده و داده ها را در آن می گذاریم.

از آنجایی که هدف ما یادگیری MySQL یا همان پایگاه داده های MySQL برای استفاده و تعامل با پی اچ پی است، از مورد اول صرف نظر می کنیم. اما این نکته را هم لحاظ کنید که شیوه کار چه به صورت مستقیم و چه به صورت استفاده از پی اچ پی، یکی است؛ پس با یادگیری دستورات توانایی هر دو را خواهید داشت.

این درس را هم در اینجا به پایان می بریم، شیوه یادگیری زبان برنامه نویسی SQL در این دوره به صورتی خواهد بود که دستورات SQL را به طور موازی با پی اچ پی در زمان های لازم ارائه خواهیم داد. به این ترتیب شما با مفاهیم پیشرفته SQL به طور عملی و در حال بکارگیری آن بهتر آشنا خواهید شد.

درس دهم- استفاده از php با پایگاه داده ها

حال که از نظر دانش مقدماتی پی اچ پی و پایگاه داده ها به سطح خوبی رسیدیم، اجازه بدهید از تمامی این دانش بهره برده و آن را در تولید و ساخت یک برگه ثبت نام پیشرفته تر به کار بریم.

عملکرد خوب پی اچ پی و هماهنگی مورد قبول آن با پایگاه داده ها، این زبان را در سطح گسترده ای مورد قبول برنامه نویسان قرار داده است. حال در این درس ما هم از این قابلیت پی اچ پی استفاده کرده و یک فرم ثبت نام که داده ها را در پایگاه انبار می کند و در صورت لازم از آنها استفاده می کند، می سازیم.

قبل از ادامه شما باید مفاهیمی که تا به حال گفته شده را به خوبی درک کرده باشید تا در طول این درس به مشکل نخورید.

ارتباط با پایگاه داده ها

در زمان آموزش MySQL، طریقه ارتباط با پایگاه داده ها را از طریق phpMyAdmin و CMD به طور مستقیم و بدون بکارگیری پی اچ پی شرح دادیم. اما مسئله اصلی این است که ما می خواهیم از پایگاه داده ها توسط پی اچ پی و با یک تعامل دو طرفه استفاده کنیم.

برای بکارگیری این کار باید از طریق توابع از قبل تعریف شده در پی اچ پی عمل کرد. ابتدایی ترین کار برای ارتباط استفاده از تابع زیر است:

```
$dbc = mysqli_connect ( hostname , username , password , db_name );
```

این تابع عمل ارتباط با پایگاه را انجام می دهد. پارامترهای آن به شرح زیر هستند:

- hostname: در اصل همان مکان برای رجوع به پایگاه است. معمولا Localhost است ولی در شرایطی ممکن است متفاوت باشد که باید از سرویس دهنده در مورد آن جویا شد.
- username: این شناسه کاربری است که در زمان ساخت پایگاه درست کردید.
- password: رمزی که به شناسه کاربری برای اتصال به پایگاه داده اید.
- db_name: نامی که برای پایگاه داده ها هنگام ساخت آن انتخاب کردید.

ساخت فایل ارتباط با پایگاه

یک راه درست برای انجام این عمل ساختن یک فایل مجزا برای ارتباط با پایگاه داده است که با داشتن این فایل در هر زمان که نیاز باشد، کافیست که فایل را در محل مناسب وارد کنیم. حتما تابع وارد کردن فایل یادتان هست!

```
include(' db_connection.php ')
```

از دیگر تکنیک هایی که می شود به کار برد قرار دادن مشخصات پایگاه در ثابت ها است. این کار امنیت ارتباط با پایگاه را بالا می برد. برای ساخت فایل ارتباط با پایگاه ابتدا فایلی با نام db_connection.php بسازید و در داخل آن کدهای زیر را قرار دهید:

```
<?php # db_connection.php
// این فایل شامل مشخصات برای ارتباط با پایگاه داده ها است
// با وارد کردن این فایل پایگاه انتخاب و ارتباط با آن برقرار می شود.
DEFINE ('DB_USER', 'root');   //  نام کاربری را در ثابت قرار می دهد.
DEFINE ('DB_PASSWORD', 'dars');
DEFINE ('DB_HOST', 'localhost');
DEFINE ('DB_NAME', 'darsnameh');
$dbc = mysqli_connect (DB_HOST, DB_USER, DB_PASSWORD, DB_NAME)
OR die (' به پایگاه متصل نشد: ' . mysqli_connect_error() );
?>
```

همان طور که مشاهده می کنید، در ابتدا مشخصات لازم به داخل ثابت ها قرار داده شده است و بعد ارتباط توسط تابع mysqli_connect برقرار می شود.

```
$dbc = mysqli_connect (DB_HOST, DB_USER, DB_PASSWORD, DB_NAME)
OR die (' به پایگاه متصل نشد: ' . mysqli_connect_error() );
```

برای شرح این دستور باید به سه نکته اشاره کرد:

۱. عمل ارتباط در داخل متغیری با نام $dbc ذخیره می شود. این کار یک اشاره گر به منظور رجوع بعدی در داخل برنامه می سازد.

۲. mysqli_connect (DB_HOST, DB_USER, DB_PASSWORD, DB_NAME): این تابع که کار ارتباط را انجام می دهد، چهار پارامتر دارد که با هر چهارتایشان آشنا هستیم.

۳. die (' به پایگاه متصل نشد: ' . mysqli_connect_error()): این تابع در صورت بروز خطا و عدم ارتباط با پایگاه یک متن «به پایگاه متصل نشد» را به همراه متن خطا چاپ می کند.

نکته: در زمان انجام بررسی ابتدایی و قبل از انتشار تارنما در اینترنت، برای دیدن و رفع خطا، قسمت مورد سوم را اضافه کنید. اما در زمان انتشار به دلایل امنیتی از این کار خودداری کنید. اگر می خواهید این تابع متن خطا را چاپ نکند، کافیست که یک علامت @ در ابتدای کد اضافه کنید:

```
$dbc = @mysqli_connect (DB_HOST, DB_USER, DB_PASSWORD, DB_NAME)
OR die (' به پایگاه متصل نشد: ' . mysqli_connect_error() );
```

حال فایل ارتباط با پایگاه آماده است که در زمان لازم با تابع زیر آن را وارد فایل اصلی می کنیم:

```
require_once ('db_connect.php'); // اتصال به پایگاه
```

به خاطر اهمیت این فایل آن را با تابع require_once وارد می کنیم تا در زمان بروز خطا از ادامه اجرای برنامه جلوگیری شود.

از آنجایی که در این درس می خواهیم یک برگه ثبت نام کامل تری از دروس قبلی بسازیم، بنابراین یک جدول دیگر با ستون های بیشتر در داخل پایگاه درست می کنیم.

فراموش نکنید که برای اینکار ابتدا باید به پایگاه وصل شوید.

```
> mysql -u root -p
password: dars
```

ابتدا به جدول هایی که از قبل درست کردیم سری می زنیم:

```
mysql> show tables;
+---------------------+
| Tables_in_darsnameh |
+---------------------+
| users               |
+---------------------+
1 row in set (0.00 sec)
```

خروجی دستور نشان می دهد که از قبل جدولی با نام users وجود دارد. بهتر است ابتدا این جدول را پاک کرده و بعد جدول جدید را بسازیم:

```
mysql>DROP TABLE users ;
```

برای پاک کردن جدول از دستور بالا استفاده می کنیم.

حال جدول جدید را با توجه به نیاز برگه ثبت نام می سازیم.

دستورات زیر را برای ساخت این جدول وارد کنید:

```
CREATE TABLE users
(
ID INT UNSIGNED NOT NULL AUTO_INCREMENT,
first_name VARCHAR(20) NOT NULL,
last_name  VARCHAR(30) NOT NULL,
username  VARCHAR(20) NOT NULL,
password  CHAR(40) NOT NULL,
email VARCHAR(80) NOT NULL,
register_date   DATETIME NOT NULL,
PRIMARY KEY (ID)
) CHARSET utf8 ;    // نوع استاندارد کاراکتری جدول
```

تا به اینجا فایل ارتباط با پایگاه و جدول مورد نیاز ثبت داده ها را آماده کردیم. از این به بعد برای ساختن این برگه موارد دیگر را اضافه می کنیم.

پوشه ای به نام DatabaseConnecion در پوشه www برای این درس آماده کنید. در داخل آن محتوای پوشه Register که در درس های قبل داشتیم را وارد کنید.

از داخل پوشه includes فایل header.html را برای انجام بعضی تغییرات باز کنید و کدهای زیر را با کدهای قبلی تعویض کنید:

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html" charset="utf-8" />
<title> <?php echo $page_title; ?> </title>
<link rel="stylesheet" href="includes/style.css" type="text/css" media="screen" />
</head>
<body>
<div id="wrap">
<div id="header">
<h1> درسنامه </h1>
<div id="navigation">
<ul>
<li><a href="index.php"> خانه</a></li>
<li><a href="register.php">  ثبت نام </a></li>
</ul>
</div>
</div>
<div id="content">
```

کار با این فایل تمام است آن را ببندید. فایل بعدی style.css است. موارد تکراری را با کدهای قبلی جابه جا و موارد جدید را اضافه کنید.

```
#wrap{
    border:0px solid #ccc;
    width:751px;
    text-align:right;
    overflow:hidden;
    background-color:#fff;
    margin:auto;}
#header {
    border:0px solid #bbb;
    height:130px;
    margin:0 auto;
    width:100%;
}
#content {
    border:0px solid #ccc;
    border-top:1px dashed #ddd;
    border-bottom:1px dashed #ddd;
    width:100%;
```

```
    height:auto;
    min-height:400px;
    padding:0 0 20px 0;
}
.error{
    color : #F00 ;  margin : 5px 20px 5px auto;
}
.again { margin : auto 20px 0 auto; }
```

ساخت فایل register.php

در کنار فایل index.php، فایل دیگری به نام register.php بسازید و کدهای زیر را در داخل آن بنویسید:

```
<?php
$page_title = 'ثبت نام';
include ('includes/header.html'); ?>
<h1>  ثبت نام </h1>
<?php
if (isset($_POST['submit']))
{
require_once ('mysqli_connect.php'); // اتصال به پایگاه
$errors = array(); //  یک آرایه برای نگهداری خطاها می سازد
$fname = $_POST['f_name'] ;
$lname = $_POST['l_name'] ;
$username = $_POST['username'];
$pass1 = $_POST['pass1'];
$pass2 = $_POST['pass2'];
$email = $_POST['email'];
if ( !empty($fname) && !empty($lname) && !empty($username) && !empty($pass1)
&& !empty($pass2) && !empty($email) )
{
    $fn = mysqli_real_escape_string($dbc , trim($fname));
    $ln = mysqli_real_escape_string($dbc , trim($lname));
    $un = mysqli_real_escape_string($dbc, trim($username));
    $e  = mysqli_real_escape_string($dbc, trim($email));
    if($pass1 == $pass2)  {
        $pass= mysqli_real_escape_string($dbc, trim($pass1));
    }else {
        $errors[] = "لطفا هر دو رمز یکسان باشند.";
    }
} else {
    if(empty($fname))
```

```
    { $errors[] = ' لطفا نام خود را وارد کنید.'; }
    if(empty($lname))
    { $errors[] = ' لطفا نام خانوادگی خود را وارد کنید.'; }
    if(empty($username))
    { $errors[] = ' لطفا شناسه کاربری خود را وارد کنید.'; }
    if(empty($password))
    { $errors[] = ' لطفا رمز خود را وارد کنید.'; }
    if(empty($email))
    { $errors[] = ' لطفا رایانامه خود را وارد کنید.'; }
    }
if (empty($errors))  {
    $query = "INSERT INTO users (first_name, last_name, username, password,
email, register_date) VALUES ('$fn', '$ln', '$un', SHA1('$pass'),'$e',  NOW() )";
//  نوشتن درخواست و گذاشتن در متغیر
    $r = @mysqli_query ($dbc, $query); // اجرای درخواست
    if ($r) { // اگر درست اجرا شد
    echo ' از ثبت نام شما سپاسگزاریم. ' . '<br />' . $fn . '<br />' .  $ln  . '<br />'
.  $un . '<br />' . SHA1( $pass ) . '<br />' . $e ;
}else { // اگر درست اجرا نشد
    echo '<h1> خطا در ثبت نام </h1><p class="error"> نام برای شما به  امکان ثبت
نید. لطفا دوباره امتحان کنید دلیل وجود مشکلی وجود ندارد. </p>';
    echo '<p>' . mysqli_error($dbc) . '<br /><br />Query: ' . $query . '</p>'; }
    mysqli_close($dbc); // Close the database connection.
        include ('includes/footer.html');
        exit();
}else{
echo '<div class="error">:خطاهای زیر صورت گرفته<br />';
    foreach ($errors as $msg) { // Print each error.
        echo "  .$msg - <br />"; }
        echo '</div><span class="again"> دوباره امتحان کنید.</span>';
}  }// پایان شرط ارسال
?>
<!-- کد برای ساخت فرم - form.html -->
<form action="register.php" method="post">
<fieldset>
<legend align="right">لطفا فرم زیر را پر کنید</legend>
<label> نام </label>
<input type="text" name="f_name" value="<?php if (isset($_POST['f_name'])){
echo $_POST['f_name'];} ?>" />
<label> نام خانوادگی </label>
<input type="text" name="l_name" value="<?php if (isset($_POST['l_name'])){
echo $_POST['l_name'];} ?>" />
<label> شناسه </label>
<input type="text" name="username" value="<?php if
```

```
(isset($_POST['username'])){ echo $_POST['username'];} ?>" />
<label> رمز </label>
<input type="password" name="pass1" />
<label> رمز  تکرار</label>
<input type="password" name="pass2" />
<label> رایانامه </label>
<input type="text" name="email" value="<?php if (isset($_POST['email'])){ echo
$_POST['email'];} ?>" />
</fieldset>
<input type="submit" name="submit" value="ارسال" class="submit"/>
</form><?php include ('includes/footer.html'); ?>
```

توضیح کدها

بیشتر این کدها را از قبل دیده ایم، در اینجا فقط به قسمت های جدید اشاره می کنیم:

```
if (isset($_POST['submit']))
{
require_once ('mysqli_connect.php'); // اتصال به پایگاه
$errors = array(); //  یک آرایه برای نگهداری خطاها می سازد
```

قسمت ابتدایی شرط نخستین می گوید: اگر دکمه ارسال زده شد، فایلی که شامل ارتباط با پایگاه است را وارد کن. این کار باعث می شود که به پایگاه داده ها که ساختیم متصل شویم.

بعد یک آرایه برای نگهداری خطاها ساخته می شود، تا در صورت نیاز همه خطاها به طور منظم در یک آرایه نگهداری شوند.

```
if ( !empty($fname) && !empty($lname) && !empty($username) && !empty($pass1)
&& !empty($pass2) && !empty($email) )
{
    $fn = mysqli_real_escape_string($dbc , trim($fname));
    $ln = mysqli_real_escape_string($dbc , trim($lname));
    $un = mysqli_real_escape_string($dbc, trim($username));
    $e  = mysqli_real_escape_string($dbc, trim($email));
    if($pass1 == $pass2)
    {
    $pass= mysqli_real_escape_string($dbc, trim($pass1));
    }
    else
    {
        $errors[] = "لطفا هر دو رمز یکسان باشند";
    }
}
```

این قسمت چند شرط را شامل می شود که بعضی ها در حالت تو در تو هستند.

شرط ابتدایی بررسی می کند که اگر تمام متغیرهایی که بر دارنده داده های گرفته شده از کادرهای فرم هستند، پر باشند، آنها را پس از گذراندن از تابع دیگری که امنیت را مهیا می کند در متغیرهای جدید قرار دهد.

این متغییرهای جدید برای انتقال داده ها به پایگاه داده ها استفاده خواهند شد. شرطی که در داخل این شرط گذاشته شده است، وظیفه بررسی یکسان بودن هر دو کادر شامل رمز را دارا است. معمولا برای انتخاب رمز دو بار از کاربر خواسته می شود که رمز را وارد کند تا از بروز خطا جلوگیری شود.

```
mysqli_real_escape_string();
```

برای برقراری امنیت، باید هر متنی از داخل این تابع بگذرد. این تابع متن را بررسی می کند و در صورت داشتن کاراکترهای مشکل ساز، آن را خالی و تمیز می کند.

```
trim($fname)
```

این تابع فاصله های غیر ضروری میان متون را از بین می برد.

```
else  {
    if(empty($fname))
    { $errors[] = ' لطفا نام خود را وارد کنید.'; }
    if(empty($lname))
    { $errors[] = ' لطفا نام خانوادگی خود را وارد کنید.'; }
    if(empty($username))
    { $errors[] = ' لطفا شناسه کاربری خود را وارد کنید.'; }
    if(empty($password))
    { $errors[] = ' لطفا رمز خود را وارد کنید.'; }
    if(empty($email))
    { $errors[] = ' لطفا رایانامه خود را وارد کنید.'; }    }
```

در قسمت دوم شرط یا همان else، در زمانی که هر کدام از متغیرها خالی باشند، متنی را برای هشدار دادن به کاربر در داخل آرایه $errors می گذارد. این متن ها به صورت مرتب در خانه های آرایه جای می گیرند. در ادامه از این آرایه استفاده خواهد شد.

اتفاقی که تا به اینجا افتاده، این است که برنامه با یک شرط از زده شدن دکمه ارسال اطمینان حاصل می کند و بعد مقادیر داخل کادرها را به متغیرها سپرده تا در ارتباط با پایگاه در آن ذخیره کند.

```
if (empty($errors))
{
```

این شرط ابتدا بررسی می کند که متغیر $errors خالی است یا نه. اگر این متغیر خالی باشد، یعنی خطایی در آن ثبت نشده و تمام داده های گرفته شده از کادرهای فرم وارد متغیرهای اصلی شده اند.

حال که خطایی وجود ندارد، باید داده ها را وارد پایگاه داده ها کرد. در ابتدای فایل دیدیم که با وارد کردن فایل پایگاه به آن متصل شدیم. حال نوبت اندوختن داده ها در آن است. این کار را با چند تابع انجام می دهیم.

```
$query = "INSERT INTO users (first_name, last_name, username, password, email, register_date) VALUES ('$fn', '$ln', '$un', SHA1('$pass'),'$e',  NOW() )"; //  نوشتن درخواست و گذاشتن در متغیر
```

وارد کردن داده به پایگاه

برای وارد کردن داده ها به پایگاه از دستوری به شکل استفاده می کنیم:

```
INSERT INTO tablename (column1, column2  ...)
VALUES (value1, value2 ...)
```

ترجمه این دستور می شود:

در جدولی به نام «نام جدول» tablename

و با ستون های «نام ستون ها» (column1, column2)

مقادیر «مقادیر» VALUES را وارد کن.

نام فنی این کار نوشتن query (بخوانید کوئری) است که ما این query را در داخل متغیری ذخیره می کنیم.

```
$r = @mysqli_query ($dbc, $query); // اجرای درخواست
```

اجرای این "کوئری" با تابعی به نام mysqli_query انجام می گیرد. این تابع دارای دو پارامتر است.

- $dbc: این همان متغیری است که در فایل db_connection.php ارتباط با پایگاه را در آن قرار دادیم.
- $query: و این هم متغیری است که در همین فایل دستورات وارد کردن را در آن قرار دادیم. این تابع کار وارد کردن داده ها را در پایگاه انجام می دهد.

```
if ($r)  { // اگر درست اجرا شد
echo ' از ثبت نام شما سپاسگزاریم. ' . '<br />' . $fn . '<br />' .  $ln  . '<br />' .  $un . '<br />' . SHA1( $pass ) . '<br />' . $e ;
```

این شرط درستی انجام وارد کردن داده ها در پایگاه را بررسی می کند که در صورت درستی، پیغام موفقیت می دهد.

```
}else { // اگر درست اجرا نشد
echo '<h1> خطا در ثبت نام </h1><p class="error">  امکان ثبت نام برای شما به دلیل
وجود مشکلی وجود ندارد. لطفا دوباره امتحان کنید. </p>';
echo '<p>' . mysqli_error($dbc) . '<br /><br />Query: ' . $query . '</p>'; }
    mysqli_close($dbc); // Close the database connection.
    include ('includes/footer.html');
    exit();
}
```

اگر وارد کردن داده ها در پایگاه با موفقیت انجام نشود، وارد قسمت else شرط می شود که پیغام مشکل ایجاد شده را می دهد. بعد از آن ارتباط با پایگاه را توسط mysqli_close($dbc); قطع می کند.

```
else{
echo '<div class="error">:خطاهای زیر صورت گرفته<br />';
    foreach ($errors as $msg) { // Print each error.
echo "  .$msg - <br />"; }
     echo '</div><span class="again">.دوباره امتحان کنید</span>';
}
}// پایان شرط ارسال
```

این قسمت در اصل قسمت else شرط بررسی کننده متغیر $errors است. زمانی برنامه وارد این قسمت می شود که این متغییر دارای مقدار باشد و این یعنی کاربر یکی از موارد ثبت نام را خالی گذاشته و یا هر دو رمز را به طور یکسان وارد نکرده است. در اینجا تمام متن هایی که به صورت اخطار وارد این آرایه شده را با یک تابع حلقه چاپ می کند تا کاربر آنها را درست وارد کند.

بخش های عددی فایل را قبلا در درس های قبلی دیده ایم، تنها تفاوت این است که ما در اینجا چند کادر اضافه برای مواردی که اضافه کردیم داریم.

خلاصه این درس

در این درس ما یک برگه ثبت نام درست و محتوای آن را درون پایگاه داده ها ثبت کردیم. این درس شامل ساختن فایلی مجزا برای اتصال به پایگاه با مشخصات لازم برای ارتباط و همچنین چگونگی وارد کردن داده ها به درون پایگاه بود.

در این درس با چند تابع جدید آشنا شدیم که عمل اتصال به پایگاه توسط پی اچ پی با آنها انجام می شود. از موارد دیگر می توان به استفاده از یک آرایه برای نگهداری خطاها و چاپ خطاها در زمان لازم نام برد. همچنین با یک دستور دیگر زبان SQL، برای وارد کردن داده ها در جدول مورد نظر آشنا شدیم.

درس یازدهم- ارتباط با پایگاه داده ها و دریافت داده ها

در درس قبلی با چگونگی ارتباط با پایگاه داده ها و قرار دادن داده ها در آن آشنا شدیم. بعد از انبار کردن داده ها در پایگاه، باید برای دسترسی به آن داده ها هم شرایطی را فراهم کرد تا در زمان لازم این عمل انجام گیرد.

در این درس به گرفتن داده های ثبت شده در پایگاه و چاپ آن در برگه دیگری می پردازیم. البته بعد از دریافت داده ها چگونگی استفاده از آنها، بر حسب نیاز سیستم است و ما فقط یک راه استفاده از آنها را در این درس شرح می دهیم.

دریافت دادههای ثبت شده در پایگاه
در درس قبل در زمان ارسال داده ها به پایگاه، البته بعد از ارتباط با آن، از مفهومی به نام Query یا پرسوجو نام برده شد. برای دریافت داده ها از پایگاه هم دوباره باید از همین مفهوم استفاده کنیم، این بار دستوراتی که در پرسوجو نوشته می شوند بر طبق نیاز ما برای گرفتن داده ها خواهند بود. مسلما دستوراتی که به نام پرسوجو استفاده می کنیم در قالب دستورات زبان SQL و در بستر زبان پی اچ پی هستند.

Query پرسوجو
اجازه بدهید یک بار دیگر در مورد پرسوجو، توضیحی هر چند کوتاه بدهیم. این مفهوم در واقع زمانی معنی پیدا می کند که ما اقدام به ارتباط و ارسال یا دریافت داده ها از پایگاه می کنیم.

دستوراتی که در پرسوجو استفاده می شوند همان دستورات زبان SQL، که زبان برنامه نویسی پایگاه داده ها است، هستند. در درس قبلی با یکی از این دستورها آشنا شدیم، اگر یادتان باشد به ترتیب زیر بود:

```
$query = "INSERT INTO users (first_name, last_name, username, password, email, register_date) VALUES ('$fn', '$ln', '$un', SHA1('$pass'),'$e',  NOW() )"; //  نوشتن درخواست و گذاشتن در متغیر
```

این دستوری است که در قالب یک پرسوجو برای ارسال و ثبت داده ها در پایگاه نوشته شده است. این دستور با انتخاب نام جدول و ستون هایش، داده ها را که با متغیرها معرفی می شود را در ستون های مترادف می نشاند.

حال که داده ها در پایگاه قرار دارند، باید پرسوجو برای دریافت آنها آماده کرد؛ این پرسوجو از دستوری به شکل زیر استفاده می کند:

```
SELECT نام ستون ها FROM نام جدول
SELECT  *  FROM  نام جدول
```

*: منظور از این علامت انتخاب همه ستون ها است.

برای شرح بهتر این عمل اجازه بدهید برگه ای برای نمایش داده ها پس از رفتن آنها از پایگاه بسازیم و داده ها را در آن چاپ کنیم.

دریافت داده ها از پایگاه

پوشه DatabaseConnecion که در درس قبلی درست کردیم را باز کنید. فایل جدیدی به نام view.php در آن بسازید و کدهای زیر را در داخل آن وارد کنید:

```
<?php
$page_title = 'نمایش داده ها';
include ('includes/header.html');
echo '<h1> داده ها  </h1>';
require_once ('db_connection.php'); // ارتباط با پایگاه
// نوشتن پرسوجو
$q = "SELECT * FROM users ORDER BY register_date ASC";
$r = @mysqli_query ($dbc, $q); // اجرای پرسوجو
// شمارش تعداد دریافتی ها
$num = mysqli_num_rows($r);
if ($num > 0) { // اگر دریافتی وجود داشت
    // چاپ تعداد کاربرها
    echo "<p> کاربر  در حال حاضر $num وجود دارد </p> ";
    echo '<div id="view">';
    // سربرگ جدول
    echo '<div id="table-header">
    <div class="table-header"> نام </div>
    <div class="table-header">  نام خانوادگی </div>
    <div class="table-header"> نام کاربری </div>
    <div class="table-header"> رایانامه </div>
    <div class="table-header"> تاریخ ثبت نام </div>
    </div>';
    // دریافت و چاپ همه رکوردها
    echo '<div id="table-record">';
    while ($row = mysqli_fetch_array($r, MYSQLI_ASSOC)) {
            echo '<div class="table-record">' . $row['first_name'] . '</div>' ;
            echo '<div class="table-record">' . $row['last_name'] . '</div>' ;
            echo '<div class="table-record">' . $row['username'] . '</div>' ;
            echo '<div class="table-record">' . $row['email'] . '</div>' ;
            echo '<div class="table-record">' . $row['register_date'] . '</div>' ;
    }
    echo '</div>'; // پایان رکوردها
    echo '</div>'; // پایان جدول
    mysqli_free_result ($r); // آزادسازی منابع
```

```
} else { // اگر رکوردی نبود
echo '<p class="error"> در حال حاضر کوئری ای وجود ندارد </p>'; }
mysqli_close($dbc); // بستن ارتباط با پایگاه
include ('includes/footer.html');
?>
```

توضیح کدها

مطابق برگه ثبت نام این برگه هم با نام برگه، وارد کردن سربرگ و ارتباط با پایگاه آغاز می شود. در بخش بعد، پرسوجو برای دریافت داده ها از پایگاه، درون یک متغیر قرار می گیرد:

```
$q = "SELECT * FROM users ORDER BY register_date ASC";
```

این دستور SQL تمام رکوردهای جدول را انتخاب و بعد براساس تاریخ ثبت نام مرتب می کند:

```
ORDER BY register_date ASC
```

این قسمت که در انتهای دستور آمده می گوید که: داده ها را برحسب تاریخ ثبت نام به صورت ascending (صعودی) مرتب کن. برای مرتب سازی به صورت descending (نزولی) کافی است که جای ASC را با DESC تغییر دهید. با این عمل کار مرتب سازی از همان پایگاه انجام می شود.

```
$r = @mysqli_query ($dbc, $q); // اجرای پرسوجو
```

این خط اجرای دستور پرسوجو را انجام می دهد که حامل دو پارامتر است: نشانگر پایگاه و متغیر شامل دستور پرسوجو. خروجی دستور در متغیر r$ ذخیره می شود.

```
// شمارش تعداد دریافتی ها
$num = mysqli_num_rows($r);
```

در این خط کد، با استفاده از تابع mysqli_num_rows، تعداد رکوردهای ثبت شده در پایگاه را در داخل یک متغیر به نام num$ می ریزیم.

```
if ($num > 0) { // اگر دریافتی وجود داشت
    // چاپ تعداد کاربرها
    echo "<p> وجود دارد $num در حال حاضر  کاربر </p> ";
```

این شرط بررسی می کند که آیا متغیر num$ دارای مقدار هست یا نه؟ اگر این متغیر دارای مقدار باشد، این یعنی پایگاه دارای داده های ثبت شده است چرا که در قسمت قبل تعداد این رکوردها شمرده شده و در این متغیر ذخیره گشته است. با درست بودن این شرط، برنامه به داخل بلوک شرط وارد می شود و ابتدا تعداد رکوردها را چاپ می کند.

```
echo '<div id="view">';
    // سربرگ جدول
    echo '<div id="table-header">
    <div class="table-header"> نام </div>
    <div class="table-header">  نام خانوادگی </div>
    <div class="table-header"> نام کاربری </div>
    <div class="table-header"> رایانامه </div>
    <div class="table-header"> تاریخ ثبت نام </div>
    </div>';
    // دریافت و چاپ همه رکوردها
    echo '<div id="table-record">';
    while ($row = mysqli_fetch_array($r, MYSQLI_ASSOC)) {
        echo '<div class="table-record">' . $row['first_name'] . '</div>' ;
        echo '<div class="table-record">' . $row['last_name'] . '</div>' ;
        echo '<div class="table-record">' . $row['username'] . '</div>' ;
        echo '<div class="table-record">' . $row['email'] . '</div>' ;
        echo '<div class="table-record">' . $row['register_date'] . '</div>' ;
    }
    echo '</div>'; // پایان رکوردها
    echo '</div>'; // پایان جدول
```

در این قسمت با استفاده کردن از کدهای CSS جدولی برای نمایش داده ها می سازیم. سطر ابتدایی این جدول سربرگ آن است و در ادامه رکوردها را در سطرهای بعدی می گذاریم.

```
mysqli_free_result ($r); // آزادسازی منابع
```

در این قسمت در پایان نمایش همه رکوردها، از این تابع استفاده می کنیم تا متغیر $r را که قسمتی از حافظه را در اختیار گرفته آزاد کنیم. با این عمل با توجه به عدم نیاز ما به این متغیر، فضای گرفته شده از حافظه رایانه آزاد می شود.

```
} else { // اگر رکوردی نبود
echo '<p class="error"> در حال حاضر کاربری وجود ندارد </p>';
}
```

قسمت else برای شرط اینجا آغاز می شود؛ در صورتی که شرط درست نباشد این به این معنی است که رکوردی در پایگاه نیست و پیغام عدم وجود داده چاپ می شود.

```
mysqli_close($dbc); // بستن ارتباط با پایگاه
include ('includes/footer.html');
?>
```

قسمت پایانی، بستن ارتباط با پایگاه و وارد کردن پایین برگ است.

وظیفه برگه view.php ارتباط با پایگاه و دریافت داده ها و در نهایت چاپ آنها است. برای نمایش این برگه باید یک تغییر جزئی هم به فایل header.html بدهیم تا دکمه پیوند به این برگه در سربرگ نمایان شود.

ایجاد تغییرات در سربرگ

فایل سربرگ یا همان header.html که در پوشه includes قرار دارد را باز کنید. به قسمت نمایش دهند ناوبری رفته و تکه کد زیر را در آن وارد کنید.

```
<li><a href="view.php">  نمایش داده ها </a></li>
```

بعد از وارد کردن این تکه کد، قسمت ناوبری باید به شکل زیر در آید:

```
<div id="navigation">
<ul>
<li><a href="index.php"> خانه</a></li>
<li><a href="register.php">  ثبت نام </a></li>
<li><a href="view.php">  نمایش دادهها </a></li>
</ul>
</div>
```

حال مرورگر خود را دوباره بارگذاری کنید تا این دکمه پدیدار گردد. خروجی آن می شود:

تغییرات در فایل style.css

با اضافه شدن این برگه و نیاز به نمایش داده ها در آن باید یکسری کد سی اس اس به فایل آن اضافه کنیم:

```
/*      قسمت نمایش داده ها   */
#view{border:0px solid #ddd; overflow:hidden; width:90%; margin:auto;}
#table-header{ overflow:hidden; width:100%; margin:auto; background-
color:#f1f1f1;}
.table-header{border-right:1px solid #ccc;float:right; width:18.8%;
height:30px;padding:0 1% 0 0; line-height:2em; }
#table-record{ overflow:hidden; width:100%; margin:auto;}
.table-record{border-right:1px solid #ddd;float:right; width:18.8%;
height:25px;padding:0 1% 0 0; background-color:#f9f9f9;line-height:2em;}
```

این مجموعه کد را به قسمتی که می خواهید در فایل سی اس اس اضافه کنید. این کدها وظیفه نمایش بهتر و به طور دلخواه ما را دارا هستند.

حال برای امتحان درستی کدها و نحوه نمایش داده ها به برگه ثبت نام در تارنما رفته و فرم را پر کنید:

دکمه ارسال را زده تا پیغام موفقیت را ببینید.

حال پس از دریافت این پیغام با فشردن دکمه نمایش داده ها به قسمت نمایش بروید. شما باید جدولی که شامل کاربر ثبت نام شده است را مشاهده کنید:

در حال حاضر کاربر 2 وجود دارد

نام	نام خانوادگی	نام کاربری	رایانامه	تاریخ ثبت نام
پیمان	[redacted]	peyman	[redacted]@yahoo.com	2012-12-07 15:34:49
پویا	[redacted]	pooya	[redacted]@yahoo.com	2012-12-07 17:54:48

همان طور که مشاهده می کنید در خط ابتدایی، تعداد کاربران ثبت نام کرده را نمایش می دهد. در داخل جدول هم اطلاعات این کاربران به صورت یک جدول نمایان می شود.

برای مشاهده این اطلاعات می توانید مستقیما به پایگاه داده های خود رجوع کنید. کافی است که در محل نشانی مرورگر خود آدرس http://localhost/phpmyadmin/ را وارد کنید. بعد از وارد شدن به پایگاه باید ابتدا نام پایگاه داده ها که ساخته اید را انتخاب و سپس نام جدول خود را که در اینجا users است را برگزینید.

برای مشاهده ساختار جدول بر روی گزینه Structure در بالای صفحه کلیک کنید و برای دیدن داده های ثبت شده در جدول بر روی گزینه کناری آن با نام Browse کلیک کنید.

همان طور که می بینید phpMyAdmin قابلیت های بسیاری را برای شما ارائه می کند. به طور مثال شما می توانید یک رکورد را ویرایش، رونویسی و یا پاک کنید. البته استفاده از این قابلیت برای شرایط خاصی مفید است اما اینکه بتوانیم داده ها را تغییر و یا پاک کنیم باید چیزی باشد که توسط دستورات SQL و از طریق پی اچ پی انجام گیرد.

معمولا در سیستم هایی که کاربر جذب می کنند، ابزارهایی اینچنینی برای راحت تر کردن کار مدیر سیستم در نظر گرفته می شوند. به این طریق مدیر بدون نیاز مراجعه به پایگاه داده از طریق phpMyAdmin و یا هر نرم افزار دیگری، انجام عملیات های مورد نظر خود را از راه داشتن برگه هایی برای این کار در سیستم انجام می دهد.

درس دوازدهم- استفاده php با پایگاه داده ها در به روزرسانی داده ها

تا به حال در درس های گذشته ما یاد گرفتیم که چگونه فرم ساخته و داده های آن را در پایگاه داده ها ثبت کنیم و پس از آن آنها را در برگه دیگر نمایش دهیم.

در این درس با تکنیک دیگری آشنا می شویم که این قدرت را به ما می دهد که داده ها را تغییر دهیم و یا به قول معروف به روزرسانی کنیم. برای توضیح این مهم از تغییر رمز ثبت شده در پایگاه به عنوان یک مثال کارآمد استفاده می کنیم.

به روزرسانی داده ها در پایگاه

برای تغییر داده ها در پایگاه باید از دستور دیگری از SQL که برای این کار نوشته شده است استفاده کنیم. صورت کلی دستور به شرح زیر است:

```
UPDATE نام جدول SET نام ستون بعدی = مقدار, ستون = مقدار
```

این دستور با کلمه UPDATE یعنی به روزرسانی آغاز می شود و بعد باید نام جدول مورد نظر را داشته باشد.

قسمت بعدی کلمه SET است که باید بعد از آن نام ستون و مقداری که می خواهید در آن قرار گیرد را وارد کنید. به همین ترتیب ستون بعدی و مقدار آن.

البته این دستور شاید به تنهایی زیاد کاربرد نداشته باشد، به این دلیل که معمولا به روزرسانی داده هدفمند است و باید محل آن داده را در پایگاه مشخص کرد.

فرض کنید که می خواهید نام شخصی را در پایگاه به روزرسانی کنید و یا تغییر دهید. کاری که قبل از به روزرسانی باید انجام دهید دانستن نشانی آن شخص در پایگاه است. به مثال زیر دقت کنید که در آن ما رایانامه و شناسه کاربری شخصی را داریم و می خواهیم نام او را تغییر دهیم:

```
SELECT ID FROM users // انتخاب شماره کاربر از جدول
WHERE email = 'peyman@yahoo.com'
AND username = 'peyman' ; //    از جایی که رایانامه و شناسه آن برابر این مقدار است
```

تا به اینجا ما شماره ID آن کاربر را از جدول users از جایی که رایانامه او برابر 'peyman@yahoo.com' و شناسه کاربری او برابر peyman است بدست می آوریم. فرض کنید که این شماره ۵۲ است. قدم بعدی به روزرسانی نام از طریق همین شماره است.

```
UPDATE users  // به روزرسانی جدول
SET first_name = 'pejman'  //  مقدار نام را قرار بده
WHERE ID = 52 ; //  در جایی که شماره آن برابر مقدار روبرو است
```

نکته مورد توجه اینجا استفاده از عبارت WHERE است که در هر دو قسمت برای مشخص کردن ناحیه جستجو و یا به روزرسانی به کار برده شده است.

برای درک بیشتر این دستورات و نحوه کاربرد آن در پی اچ پی سراغ برگه دیگری می رویم.

به روزرسانی رمز کاربری

برای انجام این مهم یک برگه دیگر در پوشه خود درست می کنیم، نام برگه را password.php می گذاریم و کدهای زیر را در آن می نویسیم:

```
<?php
$page_title = ' بروزرسانی رمز ' ;
include ('includes/header.html' ) ;
// بررسی ارسال فرم
if (isset($_POST['submitted']))  {
require_once ('db_connection.php') ;
$errors = array();
// بررسی رایانامه
if (empty($_POST['email']))  {
$errors[] = 'لطفا رایانامه را وارد کنید';
} else {
$e = mysqli_real_escape_string($dbc, trim($_POST['email'])) ;
}
// بررسی رمز فعلی
if (empty($_POST['pass']))  {
$errors[] = ' رمز فعلی را وارد کنید';
} else {
$p = mysqli_real_escape_string($dbc, trim($_POST['pass']));
}
// برابری رمز تازه و تکرار آن
if (!empty($_POST['pass1']))  {
if ($_POST['pass1'] != $_POST['pass2'])  {
$errors[] = ' رمزهای تازه باید یکسان باشند ' ;
} else {
$np = mysqli_real_escape_string($dbc, trim($_POST['pass1'])) ;
}
} else {
$errors[] = 'رمز تازه را وارد نکردید ' ;
}
if (empty($errors)) {   // در صورت درستی یعنی اگر اشکالی نبود و آرایه خطاها خالی بود
```

```
// بررسی رایانامه و رمز وارد شده
$q = "SELECT ID FROM users WHERE (email='$e' AND password=SHA1('$p') )" ;
$r = @mysqli_query($dbc, $q) ;
$num = @mysqli_num_rows($r) ;
if ($num == 1) { // اگر برابر بودند مقدار یک درست می شود
// شماره کاربر را می گیرد
$row = mysqli_fetch_array($r, MYSQLI_NUM);
// پرسوجو برای بروزرسانی
$q = "UPDATE users SET password=SHA1('$np') WHERE ID = $row[0] " ;
$r = @mysqli_query($dbc, $q);
if (mysqli_affected_rows($dbc) == 1) { // اگر درست کار کرد
// خروجی موفقیت
echo '<h1> سپاسگزاریم</h1>
<p> رمز شما به روزرسانی شد </p><p><br /></p>';
} else { // اگر درست کار نکرد
// پیام خطا
echo '<h1> خطای سیستم </h1>
<p class="error">رمز شما به خاطر خطای سیستم تغییر نیافت  </p>';
// پیغام کد خطا
echo '<p>' . @mysqli_error($dbc) . '<br /><br /> پرسوجو : ' . $q . '</p>';
}
include ('includes/footer.html');
exit();
} else { // اگر روز و شناسه درست نبودند
echo '<h1> خطا </h1>
<p class="error"> رمز و رایانامه وارد شده یافت نشد </p>';
}
} else { // چاپ خطاها
echo '<h1> خطا </h1>
<p class="error">خطای زیر اتفاق افتاد  <br />';
foreach ($errors as $msg) {
echo " $msg - <br /> ";
}
echo '</p><p> لطفا دوباره امتحان کنید</p><p><br /></p>';
}
mysqli_close($dbc); // ارتباط با پایگاه بسته می شود
}
?>
</form>
<!-- کد برای ساخت فرم - form.html -->
<form action="password.php" method="post">
<fieldset>
<legend align="right">فرم به روزرسانی رمز</legend>
<label> رایانامه </label>
```

```
<input type="text" name="email" value="<?php if (isset($_POST['email'])) echo
$_POST['email']; ?>"  />
<label> رمز فعلی </label>
<input type="password" name="pass" />
<label> رمز تازه </label>
<input type="password" name="pass1" />
<label>  تکرار رمز </label>
<input type="password" name="pass2"  />
</fieldset>
<input type="submit" name="submit" value="ارسال" class="submit"/>
<input type="hidden" name="submitted" value="TRUE" />
</form>
<?php
include ('includes/footer.html');
?>
```

تغییر برگه سربرگ

برای آورده شدن این برگه لطفا ابتدا کد زیر را در برگه header.html وارد کنید:

```
<li><a href="password.php">  به روزرسانی رمز </a></li>
```

حتما می دانید که این کد باید در زیر کدهای دیگر فهرست ناوبری ما قرار گیرید:

```
<ul>
<li><a href="index.php">خانه</a></li>
<li><a href="register.php">  ثبت نام </a></li>
<li><a href="view.php">  کاربران </a></li>
<li><a href="password.php">  به روزرسانی رمز </a></li>
</ul>
```

حال باید پس از انتخاب دکمه «به روزرسانی رمز» در فهرست بالایی، برگه ای مانند تصویر زیر داشته باشید:

همانطور که مشاهده می کنید دکمه به روزرسانی رمز اضافه شده و فرم تغییر رمز در آن پدیدار گشته است.

شرح کد برگه بروزرسانی رمز password.php

ابتدایی ترین المان این برگه پس از علامت php?> که شروع کدهای پی اچ پی است، انتخاب نام برگه است و بعد وارد کردن برگه سربرگ.

```
$page_title = ' بروزرسانی رمز ';
include ('includes/header.html');
```

شرط ابتدایی بررسی می کند که آیا دکمه «ارسال» انتخاب شده است یا نه، که در صورت درستی شرط وارد بلوک آن می شود. حال باید با پایگاه ارتباط برقرار کرد و بعد آرایه ای برای ثبت خطاها. دقت کنید که از شرح بیشتر مواردی که قبلا توضیح داده شده خودداری می شود.

```
// بررسی ارسال فرم
if (isset($_POST['submitted'])) {
require_once ('db_connection.php');
$errors = array();
```

در داخل شرط اصلی چند شرط دیگر به طور پی در پی برای بررسی وارد شدن داده در کادرهای فرم آمده است. شرط ابتدایی و دومی بررسی می کنند که آیا «رایانامه» و «رمز فعلی» وارد شده یا نه؟

```
// بررسی رایانامه
if (empty($_POST['email'])) {
$errors[] = 'لطفا رایانامه را وارد کنید';
} else {
$e = mysqli_real_escape_string($dbc, trim($_POST['email']));
}
// بررسی رمز فعلی
if (empty($_POST['pass'])) {
$errors[] = ' رمز فعلی را وارد کنید';
} else {
$p = mysqli_real_escape_string($dbc, trim($_POST['pass']));
}
```

اما در شرط سوم به دلیل اینکه برای ثبت و به روزرسانی رمز تازه کاربر باید این رمز را دو بار بنویسد، یک شرط دیگر اضافه شده که برابری رمز تازه و تکرار آن را بررسی می کند.

```
// برابری رمز تازه و تکرار آن
if (!empty($_POST['pass1'])) {
if ($_POST['pass1'] != $_POST['pass2']) {
$errors[] = ' رمزهای تازه باید یکسان باشند ';
} else {
$np = mysqli_real_escape_string($dbc, trim($_POST['pass1']));
}
} else {
$errors[] = 'رمز تازه را وارد نکردید ';
}
```

تا به اینجای برنامه از ارسال داده ها توسط کاربر در کادرهای فرم اطمینان پیدا می کنیم، حال نیاز به شرطی داریم که از خالی بودن آرایه خطا اطمینان حاصل کند، چرا که اگر این متغیر خالی باشد، یعنی که خطایی رخ نداده و کاربر همه کادرهای فرم را به طور صحیح وارد کرده است.

```
if (empty($errors)) { // در صورت درستی یعنی اگر اشکالی نبود و آرایه خطاها خالی بود
// بررسی رایانامه و رمز وارد شده
$q = "SELECT ID FROM users WHERE (email='$e' AND password=SHA1('$p') )";
$r = @mysqli_query($dbc, $q);
$num = @mysqli_num_rows($r);
```

اگر متغیر آرایه ای خطاها خالی بود، ابتدا یک پرسوجو (Query) برای گرفتن شماره کاربر مورد نظر با توجه به رایانامه و رمز ابتدایی که وارد کرده می نویسیم. پرسوجو را اجرا و نتیجه آن را

در داخل متغیر $r می گذاریم. خط آخر این دسته کد، تعداد خروجی این پرسوجو پس از اجرا را در متغیر $num می گذارد.

```
if ($num == 1) { // اگر برابر بودند مقدار یک درست می شود
// شماره کاربر را می گیرد
$row = mysqli_fetch_array($r, MYSQLI_NUM);
```

از آنجایی که فقط یک کاربر با رایانامه مورد نظر ثبت نام کرده، خروجی $num باید برابر ۱ شود که در صورت درستی توسط دستور mysqli_fetch_array کلیه رشته هایی که در رکورد این رایانامه مورد نظر هست در آرایه $row ذخیره می شود. حال ما اطلاعات کاربر مورد نظر را داریم و می توانیم با توجه به این اطلاعات با رجوع به خانه ای که شماره کاربر را دارد، برای به روزرسانی دیگر قسمت ها دستور دیگری بنویسیم.

```
// پرسوجو برای به روزرسانی
$q = "UPDATE users SET password=SHA1('$np') WHERE ID = $row[0] " ;
$r = @mysqli_query($dbc, $q);
```

دستوری که به عنوان پرسوجو می نویسیم این است که رمز تازه را وارد ستون رمز کاربر، با شماره ای که ارائه می دهیم، بکند. یعنی می گوییم جدول users را انتخاب کن و رمز را در خانه رمز کاربر با شماره $row[0] قرار بده. $row[0] یک آرایه است و با توجه به کلید این آرایه که 0 است، این آرایه داده خانه ابتدایی که همان ID در جدول users است را در خود دارد.

```
if (mysqli_affected_rows($dbc) == 1) { // اگر درست کار کرد
// خروجی موفقیت
echo '<h1> سپاسگزاریم</h1>
<p> رمز شما به روزرسانی شد </p><p><br /></p>';
```

این قسمت کد با یک شرطی، اطمینان حاصل می کند که یکی از رکوردهای جدول به روزرسانی شده است. این عمل با تابع mysqli_affected_rows($dbc) انجام شده است که بررسی می کند آیا رکوردی تغییر کرده یا نه. اگر این تابع برابر ۱ باشد، یعنی رکوردی به روزرسانی شده و پیغام موفقیت صادر می شود.

```
} else { // اگر درست کار نکرد
// پیام خطا
echo '<h1> خطای سیستم </h1>
<p class="error">رمز شما به خاطر خطای سیستمی تغییر نیافت  </p>';
// پیغام کد خطا
echo '<p>' . @mysqli_error($dbc) . '<br /><br /> پرسوجو : ' . $q . '</p>';
}
include ('includes/footer.html');
exit();
```

اگر رکوردی به روزرسانی نشود برنامه داخل else می شود که پیغام خطا همراه کد خطا را چاپ می کند. همان طور که گفتیم چاپ خطا تنها در زمان امتحان و برنامه نویسی درست است و پس از اتمام کار نباید هیچ خطایی را چاپ کنید. به دلایل امنیتی البته!

در انتهای این قسمت بدلیل عدم انتشار دوباره فرم، پایین صفحه را وارد و با دستور exit() از اجرای باقی صفحه جلوگیری می کنیم.

```
} else { // اگر روز و شناسه درست نبودند
echo '<h1> خطا </h1>
<p class="error"> رمز و رایانامه وارد شده یافت نشد </p>';
}
```

این قسمت مربوط به شرطی است که وجود رایانامه و رمز را در پایگاه بررسی می کرد که در صورت عدم وجود برنامه به این قسمت آمده و خطای مورد نظر این بخش را چاپ می کند.

```
} else { // چاپ خطاها
echo '<h1> خطا </h1>
<p class="error">خطای زیر اتفاق افتاد  <br />';
foreach ($errors as $msg) {
echo " $msg - <br /> ";
```

اگر آرایه خطاها خالی نبود، یعنی اینکه کاربر یکی از موارد کادر فرم را وارد نکرده و وارد این بخش می شود که منتهی به چاپ آن خطاها می شود.

```
mysqli_close($dbc); // ارتباط با پایگاه بسته می شود
.انتهای این برگه با بستن ارتباط با پایگاه داده ها و بعد چاپ فرم
```

به این ترتیب ما برگه ای به نام «بروزرسانی رمز» اضافه کرده و کار به روزرسانی پایگاه داده ها را با توجه به رایانامه کاربر مورد نظر انجام دادیم. برای اینکه این تغییرات صورت پذیرد، ما باید رایانامه کاربر مورد نظر را وارد کنیم.

درس سیزدهم- انواع دیگر فرستادن داده ها به php

تا این درس همه داده هایی که به کدهای پی اچ پی فرستاده شد، داده هایی بودند که کاربر در فرم وارد می کرد. البته دو راه دیگر هم برای این مهم وجود دارد که با کمک آنها ما می توانیم داده هایمان را به کدهای پی اچ پی برسانیم. در این درس به این راه ها می پردازیم.

انواع راه های فرستادن مقدار به کد اجرایی

روش اول: راه ابتدایی استفاده از برچسب پنهان و یا Hidden Input است. در زمان ساخت فرم با برچسب Input آشنا شدیم که توسط آن کادرهای یک فرم ساخته می شوند. شیوه اجرای این مهم به شرح زیر است:

```
<input type = " hidden "  name = " نام برچسب " value = "  مقدار " />
```

این برچسب چند خصوصیت دارد که شامل:

- type: نوع برچسب را تعیین می کند که اینجا پنهان است.
- name: نام برچسب برای رجوع و دسترسی به مقدار آن
- value: مقداری که توسط این برچسب فرستاده می شود.

دلیل پنهان یا "hidden" بودن نوع " type" این برچسب این است که دلیلی برای نمایش آن نیست و ما تنها می خواهیم با کمک این برچسب داده ای را انتقال دهیم. البته محل قرار گرفتن این برچسب هم داخل برچسب های فرم است.

نکته: در این نوع فرستادن داده ها، این عمل به طور پنهانی انجام شده و نام و مقدار از چشم پنهان است.

نکته: در این روش از هر دو متد یعنی POST و GET می توان استفاده کرد.

روش دوم: راه دوم فرستادن داده ها با اضافه کردن آنها به انتهای آدرس URL است که البته نام و مقدار در انتهای آدرس صفحه پدیدار می شوند. برای استفاده از این شیوه باید از متد GET در فرم استفاده شود.

در زمان تعریف خصوصیت های فرم به دو شیوه آن یعنی POST و GET اشاره کردیم. گفتیم که در POST داده ها در آدرس نمایان نمی شوند اما در GET داده ها براساس نام و مقدار در آدرس نمایان می شوند.

حال فرض کنید که می خواهید کاربری را پاک کنید، برای این کار می توان از داخل پایگاه عمل کرد، اما بهترین کار داشتن برگه ای برای انجام این همه است.

برگه پاک کردن کاربر delete.php

در داخل پوشه تارنمای خود، در کنار باقی فایل ها، فایل دیگری با نام delete.php بسازید و کدهای زیر را در آن قرار دهید:

```
<?php
$page_title = 'پاک کردن کاربر';
include ('includes/header.html');
echo '<h1> پاک کردن کاربر </h1>';
// بررسی شماره شناسه کاربری
if ( (isset($_GET['id'])) && (is_numeric($_GET['id'])) ) { // از view.php
$id = $_GET['id'];
} elseif ( (isset($_POST['id'])) && (is_numeric($_POST['id'])) ) { // از فرم
$id = $_POST['id'];
} else { // اگر مقدار درستی نبود باید برگه بسته شود.
echo '<p class="error"> این برگه به اشتباه باز شده </p>';
include ('includes/footer.html'); exit();  }
require_once ('db_connection.php');
// بررسی ارسال فرم
if (isset($_POST['submit'])) {
if ($_POST['sure'] == 'Yes') { // پاک کردن رکورد
$q = "DELETE FROM users WHERE ID = $id LIMIT 1";
$r = @mysqli_query ($dbc, $q);
if (mysqli_affected_rows($dbc) == 1) { // اگر درست کار کرد
echo '<p> کاربر پاک شد </p>';
} else { // اگر پرسوجو درست کار نکرد
echo '<p class="error"> بدلیل خطای سیستمی کاربر پاک نشد </p>'; // Public
message.
echo '<p>' . mysqli_error($dbc) . '<br /> پرسوجو: ' . $q . '</p>'; // Debugging
message.
} else { // پیغام پاک نشدن کاربر در صورت عدم تایید
echo '<p> کاربر پاک نشد </p>'; }
} else {
// گرفتن اطلاعات کاربر
$q = "SELECT CONCAT(last_name, ', ', first_name) FROM users WHERE ID=$id";
$r = @mysqli_query ($dbc, $q);
if (mysqli_num_rows($r) == 1) {
$row = mysqli_fetch_array ($r, MYSQLI_NUM);
echo '<form action="delete.php" method="post">
<h3> نام: ' . $row[0] . '</h3>
آیا مطمئن هستید که می خواهید این کاربر را پاک کنید؟ <br />
<div id="delete">
<label> بله  </label> <input type="radio" name="sure" value="Yes" />
<label> خیر  </label> <input type="radio" name="sure" value="No"
checked="checked" />
```

```
</div>
<div><input type="submit" name="submit" class="submit" value=" ارسال" /></div
<input type="hidden" name="submitted" value="TRUE" />
<input type="hidden" name="id" value="' . $id . '" />
 </form>';
} else { // شماره کاربر نادرست
echo '<p class="error"> این برگه اشتباهی باز شده است. </p>';
} }
mysqli_close($dbc);
include ('includes/footer.html');
?>
```

برای دسترسی به این برگه دیگر دکمه ای در فهرست ناوبری گذاشته نمی شود، بلکه این برگه از داخل برگه کاربران و با اضافه شدن پیوند «پاک کردن» قابل دسترسی می شود.

تغییرات در برگه کاربران view.php

تنها کاری که باید در این برگه صورت بپذیرد اضافه کردن یک گزینه دیگر در جدول نمایش کاربران است. پس در زیر گزینه «تاریخ ثبت نام» که در سرتیتر جدول است، کد زیر را قرار دهید:

```
<div class="table-header"> پاک کردن </div>
```

بعد از اضافه کردن این کد مجموعا قسمت تیتر جدول به شکل زیر باید باشد:

```
// سربرگ جدول
echo '<div id="table-header">
<div class="table-header"> نام </div>
<div class="table-header">  نام خانوادگی </div>
<div class="table-header"> نام کاربری </div>
<div class="table-header"> رایانامه </div>
<div class="table-header"> تاریخ ثبت نام </div>
<div class="table-header"> پاک کردن </div>
</div>';
```

قسمت بعدی در بخشی قرار می گیرد که مشخصات کاربران است. پس کد زیر را به آن قسمت اضافه کنید.

```
echo '<div class="table-record">' . '<a href="delete.php?id=' . $row['ID']. '">   پاک کردن </a>' . '</div>' ;
```

و نتیجه به شکل زیر می شود:

```
// دریافت و چاپ همه رکوردها
echo '<div id="table-record">';
while ($row = mysqli_fetch_array($r, MYSQLI_ASSOC)) {
    echo '<div class="table-record">' . $row['first_name'] . '</div>' ;
    echo '<div class="table-record">' . $row['last_name'] . '</div>' ;
    echo '<div class="table-record">' . $row['username'] . '</div>' ;
    echo '<div class="table-record">' . $row['email'] . '</div>' ;
    echo '<div class="table-record">' . $row['register_date'] . '</div>' ;
    echo '<div class="table-record">' . '<a href="delete.php?id=' . $row['ID']. '"> پاک کردن </a>' . '
</div>' ;
}
```

```
'<a href="delete.php?id=' . $row['ID']. ' "> پاک کردن </a>
```

همانطور که می بینید یک پیوندی با استفاده از برچسب اچ تی ام ال در این ستون قرار می گیرد با نام «پاک کردن» که این پیوند به برگه delete.php راهنمایی می شود. بعد از نام برگه یک قسمتی با علامت "؟" اضافه می شود که "id=" را به دنبال دارد.

این دقیقا شیوه ای است که توسط آن مقداری با نام id به برگه delete.php فرستاده می شود. در ادامه این خط، مقدار است که در متغیر $row['ID'] وجود دارد. این نام و مقدار در ادامه آدرس این برگه اضافه می شوند. به طور مثال شبیه زیر:

```
http://localhost/_Darsnameh-php/Delete/delete.php?id=9
```

تغییرات برگه شیوه style.css

کدهای زیر را دربرگه شیوه نامه خود اضافه کنید:

```
/*  برگه پاک کردن       */
#delete{ overflow: hidden; margin:5px auto 10px auto;}
#delete input, #delete label{ clear:none;width:20px; margin: auto 5px auto 5px;}
```

حال اگر دکمه «کاربران» را در فهرست کلیک کنید با چیزی همانند شکل زیر مواجه می شوید:

همان طور که از تصویر پیداست در داخل جدول اطلاعات کاربران، گزینه دیگری با نام «پاک کردن» اضافه شده است. از این به بعد با اضافه شدن هر نام کاربری به این جدول (در صورت ثبت نام کاربران) این پیوند برای پاک کردن هر کاربر به جدول اضافه می شود.

نکته مهم این است که این پیوند در برگه کاربران اضافه شده است و به طور مستقیم پیوندی برای آن قرار نمی دهیم زیرا ما نیاز داریم که در صورت انتخاب این گزینه، شماره شناسه کاربری که در پایگاه داده ها با نام ID ثبت شده را به کد پی اچ پی در در برگه delete.php بفرستیم.

حال اگر بر روی پیوند «پاک کردن» کلیک کنید به آن برگه راهنمایی می شوید:

شرح نکات مهم کدهای برگه delete.php

```
// بررسی شماره شناسه کاربری
if ( (isset($_GET['id'])) && (is_numeric($_GET['id'])) ) { // از view.php
    $id = $_GET['id'];
} elseif ( (isset($_POST['id'])) && (is_numeric($_POST['id'])) ) { // از فرم
    $id = $_POST['id'];
} else { // اگر مقدار درستی نبود باید برگه بسته شود
    echo '<p class="error"> این برگه به اشتباه باز شده </p>';
    include ('includes/footer.html');
    exit();
}
```

قسمت ابتدایی شرط بررسی می کند که اگر متد استفاده شده در فرم شیوه GET است، آیا مقدار به شکل عدد صحیح، با نام id ارسال شده یا نه، در صورت درستی شرط آن عدد یا مقدار داخل متغیر ذخیره می شود.

در بخش esleif احتمال دیگری بررسی می شود، با این فرض که فرم فرستنده مقدار از متد POST استفاده می کند، در این صورت باز نام و مقداری را از فرم دریافت می کند و در داخل متغیر قرار می دهد، البته با یک تفاوت که این بار این نام و مقدار توسط برچسب کادر پنهان فرستاده می شود.

بخش آخر else در صورتی عمل می کند که هیچ مقداری به این برگه ارسال نشده باشد.

البته در این صورت عددی که به عنوان شماره کاربر فرستاده می شود یافت نمی شود و این باعث بروز خطا و یا مشکل می شود. در این حالت پیغام اشتباه چاپ شده و برنامه متوقف می شود. یکی از حالت هایی که ممکن است این اتفاق بیافتد زمانی است که کاربری به طور مستقیم به این برگه دسترسی پیدا کند. مثلا با نوشتن نام برگه در قسمت آدرس دهی مرورگر:

```
http://localhost/_Darsnameh-php/Delete/delete.php
```

همانطور که می بینید در این شرایط شماره کاربر فرستاده نمی شود و پیغام اشتباه باز شدن برگه ظاهر می گردد. مسئله اصلی این است که برگه delete.php به تنهایی و به طور مستقیم کارکرد ندارد و تنها زمانی کار می کند که شماره شناسه یک کاربر به این برگه ارسال گردد.

تنها محل دسترسی به این برگه از داخل برگه «کاربران» است که در بخش قبلی دیدید که یک پیوند با نام «پاک کردن» به گزینه های برگه «کاربران» اضافه شد.

```
if (isset($_POST['submitted'])) {
    if ($_POST['sure'] == 'Yes') { // پاک کردن رکورد
        // پرسوجو
        $q = "DELETE FROM users WHERE ID = $id LIMIT 1";
        $r = @mysqli_query ($dbc, $q);
        if (mysqli_affected_rows($dbc) == 1) { // اگر درست کار کرد
                // پیغام موفقیت در پاک کردن
                echo '<p> کاربر پاک شد </p>';
        } else { // اگر پرسوجو درست کار نکرد

                echo '<p class="error"> به دلیل خطای سیستمی کاربر پاک نشد
</p>'; // Public message.
                echo '<p>' . mysqli_error($dbc) . '<br /> پرسوجو: ' . $q . '</p>'; //
Debugging message.
        }
    } else { // پیغام پاک نشدن کاربر در صورت عدم تایید
            echo '<p> کاربر پاک نشد </p>';
}
```

این یک شرطی تو در توی دیگر است که دو شرط را بررسی می کند. ابتدا، اگر دکمه «ارسال» انتخاب شده باشد و دوم، اگر گزینه «بله» برای پاک کردن زده شود. در این قسمت کاربر مورد نظر پاک می شود. قسمت else شرط داخل زمانی کار می کند که شما گزینه «خیر» را انتخاب کنید.

دستور جدیدی از SQL اینجا به کار برده شده است تا به توان رکوردی را در پایگاه پاک کرد. به شیوه بکار گیری دستور پاک کردن توجه کنید:

```
DELETE FROM users WHERE ID = $id LIMIT 1"
```

تعریف دستور این است که در جدول users رکوردی با شماره "ID" قرار گرفته که در داخل متغیر $id است را پاک کن. در آخر این دستور مورد دیگری با نام limit 1 قرار دارد که به دستور اضافه شده و می گوید فقط یک رکورد پاک شود. باقی کدهای این قسمت برای شما قبلا معرفی شده اند.

```
} else {
    // گرفتن اطلاعات کاربر
    $q = "SELECT CONCAT(last_name, ', ', first_name) FROM users WHERE ID=$id";
    $r = @mysqli_query ($dbc, $q);

if (mysqli_num_rows($r) == 1) {
    $row = mysqli_fetch_array ($r, MYSQLI_NUM);
    // فرم
```

این else متعلق به شرط ابتدایی است، یعنی تا زمانی که دکمه «ارسال» را فشار نداده باشید. اگر به تصویر توجه کنید؛

می بینید که نام و نام خانوادگی کاربر مورد نظر در بالای این قسمت آورده شده است. این کار را در این بخش کدها می توانید ببینید که چطور با یک پرسوجو این اطلاعات از پایگاه خوانده شده است.

اگر گزینه «خیر» را انتخاب کنید با پیغام زیر مواجه می شوید:

درس چهاردهم- ویرایش داده ها

در درس قبلی با چگونگی پاک کردن داده ها در پایگاه از طریق پی اچ پی و فرستادن شماره شناسه کاربر آشنا شدیم. حال می خواهیم با استفاده از تکنیک های دیگر داده ها را ویرایش کنیم.

ویرایش داده ها

برای اجرای این مهم از چند تکنیک که تا به امروز یاد گرفتیم استفاده می کنیم، مواردی مانند فرم با محتوای چسبان، برچسب های پنهان، ارزیابی داده ها و در آخر نوشتن پرسوجو.
توضیح این قسمت بسیار ساده خواهد بود زیرا که در درس قبلی چیزی مانند این کار را با اندکی تفاوت انجام دادیم.

برای این درس باید برگه دیگری با نام edit.php بسازیم و همانند درس قبل پیوند آن را در داخل برگه کاربران قرار دهیم.

ساخت برگه ویرایش

فایلی به نام edit.php در کنار دیگر فایل ها بسازید و کدهای زیر را در آن قرار دهید:

```
<?php
$page_title = ' ویرایش کاربران';
include ('includes/header.html');
echo '<h1>ویرایش کاربر  </h1>';
// بررسی شماره شناسه کاربری
if ( (isset($_GET['id'])) && (is_numeric($_GET['id'])) ) {
    $id = $_GET['id'];
} elseif ( (isset($_POST['id'])) && (is_numeric($_POST['id'])) ) {
    $id = $_POST['id'];
} else {
    echo '<p class="error"> این برگه به اشتباه باز شده </p>';
    include ('includes/footer.html');
    exit();
}
require_once ('db_connection.php');
// بررسی ارسال فرم
if (isset($_POST['submitted'])) {
    $errors = array();
    if (empty($_POST['first_name'])) {
        $errors[] = 'نام خود را فراموش کرده اید';
    } else {
        $fn = mysqli_real_escape_string($dbc, trim($_POST['first_name']));
    }
```

```
    if (empty($_POST['last_name'])) {
        $errors[] = ' نام خانوادگی فراموش شده است ';
    } else {
        $ln = mysqli_real_escape_string($dbc, trim($_POST['last_name']));
    }
    if (empty($_POST['email'])) {
        $errors[] = ' رایانامه نوشته نشده است  ';
    } else {
        $e = mysqli_real_escape_string($dbc, trim($_POST['email']));
    }
    if (empty($errors)) {
        //  آزمایش منحصر بودن رایانامه
        $q = "SELECT ID FROM users WHERE email='$e' AND ID != $id";
        $r = @mysqli_query($dbc, $q);
        if (mysqli_num_rows($r) == 0) {
        $q = "UPDATE users SET first_name='$fn', last_name='$ln', email='$e'
WHERE ID=$id LIMIT 1";
        $r = @mysqli_query ($dbc, $q);
        if (mysqli_affected_rows($dbc) == 1) { // If it ran OK.
            echo '<p> ویرایش انجام شد </p>';
        } else {
            echo '<p class="error"> خطای سیستمی! ویرایش انجام نشد </p>';
            echo '<p>' . mysqli_error($dbc) . '<br /> ' . $q . ' پرسوجو </p>';
        }
    } else { // قبلا با این رایانامه ثبت نام شده است.
            echo '<p class="error"> این رایانامه قبلا ثبت شده است.</p>';
        }

    } else { // Report the errors.
            echo '<p class="error"> خطاهای زیر انجام شده است. <br />';
            foreach ($errors as $msg) { // Print each error.
                echo " $msg - <br /> ";
            }
            echo '</p>';
    }
}
// گرفتن اطلاعات کاربر
$q = "SELECT first_name, last_name, email FROM users WHERE ID = $id";
$r = @mysqli_query ($dbc, $q);
if (mysqli_num_rows($r) == 1) {
    $row = mysqli_fetch_array ($r, MYSQLI_NUM);
    echo '<form action="edit.php" method="post">
<label>نام</label><input type="text" name="first_name" value="' . $row[0] . '" /
<label>نام خانوادگی </label><input type="text" name="last_name" value="' .
```

```
$row[1] . '" />
<label>رایانامه  </label><input type="text" name="email" value="' . $row[2] .
'"  />
<p><input type="submit" name="submit" value="ارسال" class="submit" /></p>
<input type="hidden" name="submitted" value="TRUE" />
<input type="hidden" name="id"  value="' . $id . '" />
</form>';
} else {
    echo '<p class="error"> این برگه به خطا باز شده است.</p>';
}
mysqli_close($dbc);
include ('includes/footer.html');
?>
```

قبل از اینکه به شرح کد بپردازیم ابتدا مراحل تغییرات در دیگر فایل هایی که با این برگه ارتباط دارند را شرح می دهیم.

تغییرات در برگه کاربران view.php

تغییری که باید صورت بپذیرد این است که در داخل جدول نمایش اطلاعات کاربران، یک گزینه دیگر اضافه شود؛ این گزینه در اصل پیوند به برگه «ویرایش» خواهد بود. در زیر دو تغییر را که یکی در تیتر جدول و دیگری اطلاعات کاربران است می آوریم:

```
<div class="table-header"> نام </div>
<div class="table-header">  نام خانوادگی </div>
<div class="table-header"> نام کاربری </div>
<div class="table-header" style="width:17%;"> رایانامه </div>
<div class="table-header"> تاریخ ثبت نام </div>
<div class="table-header"> پاک کردن </div>
<div class="table-header">ویرایش</div>
```

این مجموعه کدها بعد از اضافه کردن پیوند «ویرایش» است. تغییر دیگر در زیر آمده است، البته ما تمام خطوط کدها را آورده ایم تا مکان یابی آن برای شما راحت تر گردد.

```
echo '<div class="table-record">' . $row['first_name'] . '</div>' ;
echo '<div class="table-record">' . $row['last_name'] . '</div>' ;
echo '<div class="table-record">' . $row['username'] . '</div>' ;
echo '<div class="table-record" style="width:17%;">' . $row['email'] . '</div>' ;
echo '<div class="table-record">' . $row['register_date'] . '</div>' ;
echo '<div class="table-record">' . '<a href="delete.php?id=' . $row['ID']. '">  پاک
کردن </a>' . '</div>' ;
echo '<div class="table-record">' . '<a href="edit.php?id=' . $row['ID']. '">  ویرایش
</a>' . '</div>' ;
```

همانطور که از کدها پیداست خط آخر مربوط به برگه «ویرایش» است. در این مرحله هم مانند درس قبلی پیوندی به فایل edit.php داده شده است که همراه این پیوند شماره کاربر فرستاده می شود.

تغییر در فایل style.css

```
/*      قسمت نمایش داده ها   */
#view{border:0px solid #ddd; overflow: hidden; width:90%; margin: auto;}
#table-header{ overflow: hidden; width:100%; margin: auto; background-color:#f1f1f1;}
.table-header{border-right:1px solid #ccc; float: right; width:12.49%; height:30px;padding:0 1% 0 0; line-height:2em; }
#table-record{ overflow: hidden; width:100%; margin: auto;}
.table-record{border-right:1px solid #ddd; float: right; width:12.49%; height:25px;padding:0 1% 0 0; background-color:#f9f9f9;line-height:2em;}
/*  برگه پاک کردن       */
#delete{border:0px solid #ddd; overflow: hidden; margin:5px auto 10px auto;}
#delete input, #delete label{border:0px solid #ddd;clear:none;width:20px; margin: auto 5px auto 5px;}
```

این مجموعه کد که در بالا می بینید، شیوه های مورد استفاده برای برگه کاربران برای نسخه آخر نوشته شده است. با مراجعه به برگه کاربران چیزی همانند تصویر زیر را مشاهده می کنید که پیوند ویرایش را دارا است.

بعد از انجام این تغییرها اگر بر روی پیوند «ویرایش» که در برگه کاربران اضافه کرده ایم کلیک کنید، با صفحه زیر مواجه می شوید.

در این برگه سه کادر، در داخل فرم مشاهده می کنید که به طور پیش فرض حاوی اطلاعات کاربری که از برگه کاربران انتخاب کردید، است. در اینجا از تکنیک فرم چسبان (Sticky Form) برای گذاشتن این اطلاعات در فرم استفاده شده است. حال با تغییر هر کدام و فشردن دکمه ارسال، کار بروزرسانی این اطلاعات انجام می گیرد.

شرح کدها برگه ویرایش

درست مانند برگه «پاک کردن»، در برگه ویرایش هم باید شماره شناسه از طریق برچسب پنهان و یا GET از طریق آدرس انتقال یابد تا در این برگه اطلاعات کاربر مورد نظر قابل دسترسی باشد.

```
// بررسی شماره شناسه کاربری
if ( (isset($_GET['id'])) && (is_numeric($_GET['id'])) ) {
$id = $_GET['id'];
} elseif ( (isset($_POST['id'])) && (is_numeric($_POST['id'])) ) {
$id = $_POST['id'];
} else {   echo '<p class="error"> این برگه به اشتباه باز شده </p>';
include ('includes/footer.html');  exit(); }
```

اگر دقت کنید این کدها درست مانند برگه پاک کردن است.

```
// بررسی ارسال فرم
if (isset($_POST['submitted'])) {
    $errors = array();
```

```
    if (empty($_POST['first_name'])) {
        $errors[] = 'نام خود را فراموش کرده اید';
    } else {
        $fn = mysqli_real_escape_string($dbc, trim($_POST['first_name']));
    }
    if (empty($_POST['last_name'])) {
        $errors[] = ' نام خانوادگی فراموش شده است ';
    } else {
        $ln = mysqli_real_escape_string($dbc, trim($_POST['last_name']));
    }
    if (empty($_POST['email'])) {
        $errors[] = ' رایانامه نوشته نشده است  ';
    } else {
        $e = mysqli_real_escape_string($dbc, trim($_POST['email']));
    }
```

از آنجایی که این فرم دارای سه کادر است، باید پس از اطمینان از ارسال فرم، از پر بودن هر سه کادر هم اطمینان حاصل کرد. این قطعه کد همین کار را انجام می دهد.

```
if (empty($errors)) {
    //  آزمایش منحصر بودن رایانامه
    $q = "SELECT ID FROM users WHERE email='$e' AND ID != $id";
    $r = @mysqli_query($dbc, $q);
    if (mysqli_num_rows($r) == 0) {
        $q = "UPDATE users SET first_name='$fn', last_name='$ln', email='$e'
WHERE ID=$id LIMIT 1";
        $r = @mysqli_query ($dbc, $q);
        if (mysqli_affected_rows($dbc) == 1) { // If it ran OK.
            echo '<p> ویرایش انجام شد </p>';
        } else {
            echo '<p class="error"> خطای سیستمی! ویرایش انجام نشد </p>';
            echo '<p>' . mysqli_error($dbc) . '<br /> ' . $q . ' پرسوجو </p>';
        }
    } else { // قبلا با این رایانامه ثبت نام شده است
            echo '<p class="error"> این رایانامه قبلا ثبت شده است</p>';
    }
    } else { // Report the errors.
        echo '<p class="error"> خطاهای زیر انجام شده است <br />';
        foreach ($errors as $msg) { // Print each error.
            echo " $msg - <br /> ";
        }
        echo '</p>';
    }
```

در صورت پر بودن کادرهای فرم، متغیرهایی که این داده ها در آنها ذخیره می شوند، به همراه دستوری از SQL، محتوای خود را در داخل پایگاه قرار می دهند.

همانطور که قبلا گفتیم، وقتی شما با فشردن پیوند «ویرایش» در برگه کاربران به برگه ویرایش می آیید، با فرمی روبرو می شوید که سه کادر با محتوای از پیش پر شده دارد. این اطلاعات داده های فعلی ثبت شده در پایگاه هستند که با کمک تکنیک فرم چسبان گرفته شده و در کادرها قرار گرفته اند. برای انجام این مهم، فرم استفاده شده در این برگه کمی از فرم های دیگر متفاوت است. اگر به کدهای این قسمت توجه کنید شما هم این تفاوت را خواهید دید.

```
// گرفتن اطلاعات کاربر
$q = "SELECT first_name, last_name, email FROM users WHERE ID = $id";
$r = @mysqli_query ($dbc, $q);
if (mysqli_num_rows($r) == 1) {
    $row = mysqli_fetch_array ($r, MYSQLI_NUM);
    echo '<form action="edit.php" method="post">
<label>نام</label><input type="text" name="first_name" value="' . $row[0] . '" />
<label>نام خانوادگی </label><input type="text" name="last_name" value="' .
$row[1] . '" />
<label>رایانامه  </label><input type="text" name="email" value="' . $row[2] .
'"  />
<p><input type="submit" name="submit" value="ارسال" class="submit" /></p>
<input type="hidden" name="submitted" value="TRUE" />
<input type="hidden" name="id"  value="' . $id . '" />
</form>';
} else {
    echo '<p class="error"> این برگه به خطا باز شده است</p>';
}
mysqli_close($dbc);
include ('includes/footer.html');
```

در ابتدا با یک دستور پرسوجوی، داده ها از پایگاه گرفته شده و در کادرها ثبت می شوند و در آخر ارتباط با پایگاه قطع و پایین برگه آورده شده است.

با ساخت این برگه این درس هم به پایان رسید. در این درس ما برگه ای برای ویرایش داده های ثبت شده در پایگاه از طریق فرم و زبان پی اچ پی ساختیم.

این درس آخرین درس از این دوره یعنی دوره مقدماتی پی اچ پی و پایگاه داده ها بود. در طول این دوره ما از مفاهیم پایه ای زبان برنامه نویسی پی اچ پی شروع کرده و پله پله با تکنیک های بیشتری آشنا شدیم. همچنین در کنار پی اچ پی مفاهیمی از زبان SQL را بیان کردیم که به کمک پی اچ پی آمده تا ارتباط با پایگاه را امکان پذیر سازند.

در دوره بعدی که در سطح متوسط خواهد بود از تمام این مفاهیم استفاده خواهیم کرد و قدم به قدم با یادگیری مفاهیم، به صورت حرفه ای تر با زبان پی اچ پی آشنا خواهیم شد.

منابع:

درسنامه(https://darsnameh.com)

سلام برنامه(http://hiprogram.ir)

تابستان 96-95