خدمات فناوری اطلاعات اسپیرو رایانه

علی لفظی قاضی – کارشناس نرم افزار

وبسایت: www.spiro.ir

ایمیل: info@spiro.ir

چاپ شده در ۳ شماره از مجله ماهنامه وب

سیستم مدیریت فروش تحت فریم ورک cakePHP

فریم ورک چیست ؟

فریم ورک مجموعه ای از کدها، کتابخانه ها و کلاس ها می باشد که به برنامه نویس کمک می کند تا برنامه های تحت وب خود را سریع تر و با انعطاف بیشتر بسازد، فریم ورک ها با داشتن متد و توابع

مشخص و آماده این اجازه را به برنامه نویس ها می دهند تا با بکارگیری دستورات کمتر و نظم بیشتر سیستمی یکپارچه و استاندارد ارائه کنند.

ایده اصلی استفاده از فریم ورک این است که توسعه دهنده را مجبور کند تا بر اساس یک ساختار اولیه و با استفاده از عواملی که عموما مورد استفاده قرار می گیرند، برنامه کاربردی خود را بنا کند. اکثر برنامه نویسان کارکشته و با تجربه PHP، کتابخانه ها و ساختارهایی را برای گسترش سریع تر برنامه های خود تدارک می بینند و در پروژه های مختلف از آن ها بهره می گیرند.

برای درک بهتر، سایتی را در نظر بگیرید که قرار است امکان نظرسنجی به آن افزوده شود، توسعه دهنده سایت می تواند از میان سورس های رایگان موجود در وب بهره برده و سیستم نظرسنجی آنلاین را به راحتی به سایت خود بیافزاید، توسعه دهنده وب دراین گونه معماری مشکلاتی را در پیش رو خواهد داشت از جمله: پیچیده بودن تغییرات در سورس، پایگاه داده متفاوت، عمگرهای به کار رفته متفاوت، پردازش های اضافی و ...که می تواند کار را بسیار دشوار سازد. فریم ورک ها در واقع دارای یک هسته اصلی می باشند که هر گونه اضافه، توسعه و یا حذف اشیاء می تواند به راحتی و در قالب فایل های کتابخانه ای و وابسته به هسته اصلی این فریم ورک در نظر گرفته شوند.

همچنین ایجاد یک برنامه کاربردی تحت وب بر مبنای یک Framework ، باعث می گردد تا توسعه دهندگان و برنامه نویسان علاوه بر تجارب خود بتوانند از تجارب سایر برنامه نویسان نیز سود جسته و بکار بندند و مشخص بودن نوع معماری بکار رفته نیز باعث گردد تا توسعه دهندگان مختلف بتوانند به راحتی سیستم را توسعه دهند.

معرفی زبان برنامه نویسی PHP (Hypertext PreProcessor):

زبان برنامه نویسی پی اچ پی فقط جهت برنامه نویسی وب ساخته شده است، کدها بر روی سرور اجرا و به کاربر بصورت HTML نشان داده می شود.

مخصوص سرورهای لینوکس می باشد ولی بر روی ویندوز هم می توان آن را اجرا کرد.

فریم ورک کیک مانند سیستم عامل لینوکس و MySQL رایگان و Open Sourceاست.

زبان برنامه نویسی پی اچ پی فقط جهت برنامه نویسی وب ساخته شده است، کدها بر روی سرور اجرا و به کاربر بصورت HTML نشان داده می شود.
مخصوص سرورهای لینوکس می باشد ولی بر روی ویندوز نسخه خانگی هم می توان آن را اجرا کرد.
معرفی انواع فریم ورک ها جهت برنامه نویسی با PHP:

1. Zend framework
2. CakePHP
3. Symphony

با نصب این نرم افزار، نرم افزارهای: پایگاه داده MYSQL و مفسر Apache و PHPMyAdmin نیز بر روی سیستم شما نصب خواهد شد.

سرورHTTP ، ترجیاً Apache با mod_rewrite فعال (برای کاربرپسند کردنURL ها و کار با فایل‌های htaccessبهتر است قابلیت mod rewrite را فعال کنید. بدین منظور می‌بایستی فایل httpd.conf سرور آپاچی را ویرایش کنید.

برای راحت‌کار کردن با MySQL بهتر است از رابط تحت وب phpmyadmin استفاده کنید یا با استفاده از رابط گرافیکی MySQL GUI Tools دیتابیس را مدیریت کنید.

Cakephp چیست ؟

کیک‌پی‌اچ‌پی یک چارچوب (framework) نرم‌افزاری تحت وب آزاد با معماری MVC (Model View Control)برای تولید برنامه‌های وب است که به زبان پی‌اچ‌پی نوشته شده است مفهوم MVC براساس جداسازی بخشهای اساسی تعریف شده است. این تقسیم کردن باعث می شود تا کنترل و توسعه کدها به بهترین شکل ممکن انجام بشود. کیک یک فریم ورک ساخت یافته است که کاربران PHP در هر سطحی را قادر می سازد که بتوانند برنامه های قدرتمند تحت وب را به سرعت گسترش داده و بتوانند به سادگی اشیاء و متد های خود را تحت این فریم ورک بکار گیرند.

چرا از فریم ورک کیک باید استفاده کنیم؟

یکی از سوالاتی که ذهن برنامه نویس وب را به خودش مشغول خواهد ساخت این است که از میان فریم ورک های موجود برای زبان برنامه نویسی php همانند: CodeIgniter,Symphony و .... چرا باید از فریم ورک کیک برای پیاده سازی پروژهای خود برگزینیم؟

مهمترین دلیل استفاده از فریم ورک کیک را می توان به عدم نیاز به یادگیری و استفاده از دستورات پیچیده برنامه نویسی، آرایه ها، توابع و ... توسط توسعه دهنده را اشاره نمود. در واقع می توان گفت برای پیاده سازی برنامه های تحت وب تنها نیاز به ساختار کلی پایگاه داده نیاز است و روابط و ایجاد مدل ها، صفحات و کنترل کننده ها همگی بر اساس سوالاتی که توسط این فریم ورک و بر اساس تعلق داشتن و روابط جداول از کاربر پرسیده می شود، به راحتی و بدون دخالت کاربر ساخته می شود.

مهمترین مزایای استفاده از CakePHP در توسعه برنامه های کاربردی تحت وب و سایت ها را می توان موارد زیر برشمرد:

۱. فعال و جامعه دوستانه
۲. مجوز انعطاف پذیری کدها
۳. برنامه داربست شده
۴. معماری MVC
۵. درخواست توزیع کننده با آدرس های کوتاه ، سفارشی و مسیرهای
۶. قالب سریع و انعطاف پذیر (قواعد نحوی آسان)
۷. راهنما برای نمایش آژاکس، جاوا اسکریپت، فرم ها و امکانات شامل: ایمیل، کوکی، امنیت، جلسه، درخواست و جابجایی قطعات

۸. انعطاف پذیری توسط ACL (Access Control List)

۹. کش انعطاف پذیر

۱۰. محلی کردن

۱۱. بدون نیاز به پیکربندی آپاچی و کار کردن با استفاده از هر دایرکتوری وب

۱۲. پشتیبانی از الگوهای طراحی رایج (Design Patterns)

۱۳. توسعه سریع تر وب

۱۴. سازگاری با نگارش های ۴و ۵ PHP

۱۵. متن باز و مجانی بودن

۱۶. ساختارهای کد اصلاح شده

پشتیبانی از الگوهای طراحی رایج:

الگوی طراحی (Pattern Design) راه حلی کلی برای مشکلات رایج در برنامه نویسی وب است. الگوی طراحی کد کامل نیست بلکه راهکاری برای حل مشکل است که در شرایط گوناگون مورد استفاده قرار می گیرد. در برنامه نویسی تحت وب الگوهای طراحی زیادی برای حل مشکلات رایج که اکثرا تکراری نیز می باشند، وجود دارد. cakePHP تعداد زیادی از این الگوهای طراحی را به صورت توکار و یکپارچه شده پشتیبانی می نماید.

برخی از این الگوها که اشاره نیز شد عبارتند از :

رکورد فعال، کنترل جبهه ،بخش نقشه برداری داده ها و معماری MVC.

در بین موارد مذکور معماری MVC که بر اساس مدل (Model)، نما (View) و کنترل کننده (Controller) استوار است، در هسته cakePHP موجود بوده و در واقع ستون فقرات cakePHP را تشکیل می دهد.

توسعه وب به صورت سریع:

یکپارچگی الگوهای طراحی در cakePHP به این معنی است که توسعه دهندگان و برنامه نویسان دیگر درگیر حل مشکلاتی که عموماً در هنگام پیاده سازی پروژه های تحت وب بروز می کند، نخواهند گردید. چون همه این مسائل در cakePHP قبلا حل شده و توسعه دهندگان فقط بر روی منطق تجاری یا روند اصلی در هدف نرم افزار کاربردی تحت وب خود متمرکز گشته و بالطبع کارشان سریع تر پیش خواهد رفت.

سازگاری با نگارش های ۴ و ۵ PHP :

cakePHP با هر دو نگارش ۴ و ۵ PHP سازگاری کامل دارد، بنابراین برنامه کاربردی ایجاد شده با استفاده از cake به راحتی قابل حمل بوده و اجرا می شود.

cakePHP مجانی و Opensource است:

Cake به صورت مجانی و opensource تحت لیسانس MIT عرضه می گردد و تا به امروز به طور رسمی نگارش غیر مجانی از تیم توسعه دهنده عرضه نشده است. با این تفاصیل اگر شما یک PHP کار مبتدی هم که باشید با سهولت تمام می توانید با کمک سورس ها و کامنت های موجود منطق توابع، کلاس ها کتابخانه ها و ... را دریابید.

ساختار اصلاح شده کدها:

PHP یک زبان برنامه نویسی فوق العاده برای توسعه وب بوده و یادگیری آن نیز تقریبا آسان است و ساختاری شی گرا همانند C++ دارد. این زبان در هنگام کد نویسی بسیار انعطاف پذیر بوده و کد نویس یا توسعه دهنده را محدود و مقید به رعایت ساختار خاصی نمی کند. البته این مسئله در عین حال که مزیت است مشکل هم هست! برای برنامه نویسان کم تجربه و تازه کار ایجاد برنامه های بزرگ زیرا در

اکثر موارد کدهای غیر ساخت یافته دیباگ و رفع اشکال را بسیار مشکل می نماید. علاوه بر آن ایجاد تغییرات در برنامه نیز وقتی منطق در جایی تغییر کند می تواند باعث مشکلات زیادی گردد. البته این مشکلات مختص برنامه نویسان تازه کار نیست بلکه زمانی که یک برنامه پیچیده می شود مشکل حرفه ای ها نیز خواهد شد. اوضاع زمانی وخیم تر می شود که چند توسعه دهنده با هم بر روی یک پروژه کار می کنند و هر کدام از آن ها بر اساس سلیقه خودشان کد می نویسند، در نتیجه جمع آوری کدها، بهینه سازی و درنهایت یکپارچه سازی پروژه بر اساس اصول مهندسی نرم افزار خودش پروژه دیگری خواهد شد!

اما طراحان cake مشکل فوق الذکر را چنین حل کرده اند که معماری cake توسعه دهندگان را محدود به رعایت ساختاری خاص می کند، بنابراین کلیه کدها ساختار خاصی را رعایت نموده و از لحاظ ساختاری واحدند در نتیجه مدیریت،نگهداری و توسعه کدها را آسان خواهد کرد. جهت رعایت کردن ساختارهای استاندارد cake از الگوهای طراحی استفاده می کند که مهم ترین آن ها معماری MVC است.

الگو یا معماری MVC :

الگوی MVC سر نام مدل Model - نما View - کنترل کننده Controller ، به اعتقاد برخی رایج ترین الگوی جدید طراحی در توسعه نرم افزار است. در این الگو یا معماری همان طور که گفته شد کدها به سه گونه تقسیم می شوند : مدل ها ، نماها و کنترل کننده ها. مفهوم هر گونه، بستگی به نحوه پیاده سازی دارد به این معنی که انعطاف پذیر بوده و در فریم ورک های مختلف می تواند متفاوت باشد.

ساختار پوشه فریم ورک کیک:

هر کدام از اجزای فریم ورک کیک در داخل پوشه ای جداگانه قرار گرفته است که امر باعث سهولت جداسازی بخش های مختلف را امکان پذیر می نماید و شامل پوشه های:

Config: تنظیمات پیکربندی مربوط به پایگاه داده و فایل های پیکربندی هسته در این قسمت نگهداری می شوند.

Controllers: شامل کنترل کننده و کامپونت های مربوطه است.

Libs: به شما اجازه می دهد تا کتابخانه های داخلی سازمان را از کتابخانه های فروشندگان جدا کنید.

Locale: فایل های زنجیره ای را برای بین المللی سازی ذخیره می کند.

Models: شامل مدل، رفتار و منابع اطلاعاتی درخواست می باشد.

Plugins: شامل بسته های پلاگین می باشد.

Tmp: در این جا اطلاعات موقت ذخیره می شود.

Vendors: همه کلاسها و کتابخانه ها در اینجا ذخیره می شود.

Views: فایل های نمایشی در اینجا قرار می گیرد: عناصر، کمک کننده ها و فایل های نمایش

Webroot: برای شکل گیری پروژه، این پوشه به عنوان ریشه سند برای درخواست به کار می رود.همچنین برای نگهداری تصاویر و فایل های جاوا اسکریپت به کار می روند.

ساختار اصلی فریم ورک کیک:

کیک از بخش های مختلفی تشکیل شده است که در مجموع به برنامه نویس اجازه پیاده سازه پروژه را می دهد:

۱) کامپونت ها: امکان اضافه کردن توابع مختلف را به مدل فراهم می کندکه شامل:

Acl	این کامپونت برای ایجاد شکل ظاهری پایگاه داده و فایل دستیابی به کنترل بکار رود.
Auth	این کامپیونت بصورت خیلی ساده برای شناسایی هویت سیستم با استفاده از سیستم شناسایی با استفاده از انواع فرآیندهای خروج، از جمله ارتباط با کنترل کننده، ارتباط مجدد با شیء و ACL
Cookie	کامپوننت کوکی مانند کامپوننت رفتار جلسه یک پوشش حمایتی را برای PHP فراهم می کند.
Email	یک واسط است که می تواند برای فرستادن ایمیل ها به چندین بنگاه انتقال ایمیل که شامل ایمیل PHP و smtp است، به کار رود.
RequestHandler	نگهدارنده درخواست به شما اجازه می دهد تا درخواستها را به بازدیدکنندگان خود برگرداند و درخواست مربوط به انواع محتوا و اطلاعات درخواستی اطلاع دهد.
Security	کامپوننت امنیت به شما اجازه می دهد تا امنیت شدیدتری را ایجاد کنید و تصدیق HTTP را استفاده و مدیریت کنید.
Session	کامپوننت رفتار جلسه، یک پوشش مستقل ذخیره شده را برای رفتار جلسات PHP فراهم می کند.

۲) رفتارها که شامل:

ACL (Access Control List)

ACLو یا Access Control List برای مدیریت کنترل دسترسی در پروژه استفاده میشود ACL .از دو بخش درخواست کننده (ARO)و درخواست شونده (ACO) تشکیل می شود.

ARO ها همان کاربران و یا گروه های کاربری هستند که قرار است مجوز دسترسی به ACO ها را دریافت کنند و یا دسترسی آنها به ACO ها محدود شود.

ACO ها را شما انتخاب میکنید که چه چیزی باشند. درصورتی که کامپوننت ACL را به همراه کامپوننت Auth استفاده میکنید باید تمام کنترلر ها و اکشن های پروژه را به لیست ACO ها اضافه کنید.

ضمناً کامپوننت ACL باید قبل از کامپوننت Auth در کنترلر فراخوانی شود.

برای ایجاد خودکار ACO ها از روی پروژه میتوانید با ذخیره کد موجود در صفحه به آدرس:

http://book.cakephp.org/view/1549/An-Automated-tool-for-creating-ACOs

آن را ذخیره نموده و به طور مثال از طریق آدرس http://localhost/groups/build_acl ACO مربوط به پروژه خود را ایجاد نمایید.

نصب و بکارگیری کامپوننت توسط auth ساده‌ترین شکل ممکن برای داشتن یک سیستم ورود کاربر است. مانند همه کامپوننت‌ها، با افزودن 'Auth' به پارامتر$components کامپوننت به کنترلر شما افزوده می‌شود. از آنجا که قصد داریم از این کامپوننت در دیگر کنترلرها نیز استفاده کنیم برای جلوگیری از تکرار، آن را به AppController اضافه می‌کنیم. بنابر این در فایل app_controller.php در شاخه اصلی app کدهای مورد نظر را ایجاد می کنیم:

```
1.class AppController extends Controller {
2.var $components = array('Auth');
3.}
```

روشن است که چون دیگر کنترلرها فرزند AppController هستند خصوصیات آن را به ارث خواهند برد. بطور پیشفرض این کامپوننت انتظار دارد شما جدولی بنام users با فیلدهایی بنام username و passwordداشته باشید. اما در برخی موارد خاص، پایگاه داده بدلایل امنیتی اجازه نمی‌دهد که برای یک فیلد نام passwordرا انتخاب کنید.

پایگاه داده SQL مربوط به کاربران:

```
1.CREATE TABLE users (
2.id integer auto_increment,
3.username char(50),
4.password char(50),
5.PRIMARY KEY (id)
6.);
```

از username برای نام کاربری و password برای کلمه عبور استفاده کردیم. در اینجا لازم است با تابع beforeFilterدر کیک آشنا شوید CakePHP .یکسری توابع callbacks توکار دارد که با آن‌ها می‌توان منطقی را بعد یا قبل اجرای یک کنش اضافه کرد BeforeFilter .قبل از هر کنشی در کنترلر اجرا می‌شود. اما در اینجا چه استفاده ایی دارد؟ برای برخی از تنظیمات کامپوننت Auth نیاز داریم قبل از اجرای هر کنشی آن‌ها را مقدار دهی کنیم.

```
var $components = array('Session', 'Auth');

function beforeFilter()

}

$this->Auth->loginRedirect = array('controller' => 'invoices' , 'action' => 'dashboard');

$this->Auth->logoutRedirect = array('controller' => 'users' , 'action' => 'login');

$this->Auth->loginError = ' دوباره ، نام کاربری یا کلمه ی عبور اشتباه است. ';تلاش کنید

$this->Auth->authError='شما اجازه ی دسترسی به این بخش را ندارید' ;

{
```

به کنترلر users دو کنش login و logout را بصورت زیر اضافه کنید:

```
1.function login() {
2.//Auth componet does it automatically
3.}
4.function logout() {
5.$this->redirect($this->Auth->logout());
6.}
```

این کامپوننت بطور پیشفرض مسیر users/login کنترلری بنام users و کنشی بنام login را برای ورود در نظر می‌گیرد. می‌توانید با تکه کد زیر در BeforeFilter این مسیر را تغییر دهید:

```
1.$this->Auth->loginAction = array('admin' =>
false, 'controller' => 'members', 'action' => 'login');
```

در اینجا ما همان مسیر پیشفرض را بکار می‌گیریم. همچنین بطور پیشفرض نیازی نیست برای پردازش فرم login.ctp کنش login را کدنویسی کنید خود کامپوننت این کار را بطور خودکار انجام می‌دهد بنابراین آن را در بالا خالی گذاشتیم. حال فرم ورود را در login.ctp بصورت زیر تعریف کنید:

```
<?php echo $session->flash('auth');      echo $form->create('User', array('action' => 'login'));

        echo $form->input('username', array)

'label'=>'نام کاربری: '));

       echo $form->input('password', array )

'label'=>'رمزعبور: '));

       echo $form->end('ورود');?>
```

این کامپوننت برای رمزنگاری کلمه عبور از یک کلاس امنیتی استفاده می‌کند که این کلاس بصورت پیشفرض از SHA1 بهره می‌گیرد و رمزهای عبور ذخیره شده بصورت hash در پایگاه داده ذخیره می شوند.

فایل core.php و مقدار امنیتی Security.salt را یادتان هست که در ابتدای تنظیمات کیک مقداردهی می‌کردید؟! درست حدس زده‌اید، کلاس امنیتی برای رمزنگاری بیشتر، از این مقدار نیز استفاده می‌کند بنابراین دقت کنید اگر زمانی خواستید پروژه را از نو با مقادیر قدیمی رمزنگاری شده‌ی موجود در دیتابیس بنویسید این مقدار را مشابه پروژه قبل لحاظ کنید.

در اینجا برای تست نیاز دارید یک نام و کلمه عبور به دیتابیس اضافه کنید. ساخت یک فرم ثبت نام به منظور وارد کردن یک ردیف در جدول users ساده است اما ساده‌تر برای ما که تنها قصد تست برنامه را داریم چیست؟ اگر debug را در تنظیمات (core.php) دستکاری نکردید (برابر مقدار ۲ تنظیم شده است) در مرورگر مسیر users/login را مرور کنید و نام و کلمه عبور را انتخاب کنید. در نوار زیرین مربوط به debug کلمه عبوری که وارد کردید بطور هش شده (رمزنگاری شده) مشاهده می‌شود می‌توانید به phpMyAdmin بروید و این مقدار را بصورت دستی اضافه کنید. اگر هم برایتان دشوار است خط زیر را ایمپورت کنید. حال یک نام عبور ghazi با کلمه عبور ۱۲۳ دارید:

```
1.INSERT INTO 'users' ('id', 'username', 'password') VALUES
```

2.(1, 'ghazi', 'e1f0307e285b0446f4b66251b1611542636c4d39');

نوبت آن رسیده است که دسترسی به برخی از کنش‌ها را در برخی کنترلرها محدود کنیم. فرض کنید قسمت مدیریت بصورت users/account باشد. می‌خواهیم برنامه را طوری طراحی کنیم که کاربر با ورود موفق به سیستم به این بخش هدایت شود و همچنین کاربران دیگر بدون ورود مجاز به دیدن این کنش نباشند. کنش account را در کنترلر users ایجاد کنید.

از دو متد allow و deny بدین منظور استفاده می‌کنیم. فرض کنید می‌خواهیم دسترسی به کنش accountرا در کنترلر users محدود کنیم کافیست تکه کد زیر را به این کنترلر اضافه کنیم:

```
1.function beforeFilter(){
2.parent::beforeFilter();
3.$this->Auth->deny('account');
4.}
```

تکه کد بالا ابتدا خصوصیات تابع BeforeFilter کنترلر پدر (یعنی AppController را به ارث می‌برد) سپس کنش account را محدود می‌کند. با انجام مراحل فوق تلاش برای مرور users/account شما را به users/login منتقل خواهد کرد. اما بعد از ورود موفق به کجا هدایت خواهید شد؟ این کامپوننت بطور خوردکار پس از ورود موفق شما را به صفحه‌ایی که تلاش می‌کردید آن را ببینید هدایت می کند اما با کد زیر در BeforeFilter چنانچه نیاز باشد می‌توانید کاربر را به کنترلر و کنش دلخواه هدایت کنید:

```
1.$this->Auth->loginRedirect = array('controller' => 'users', 'action' => 'account');
```

حال می‌توان با مقداردهی مجدد $allowedActions و $deniedActionsدر ابتدای هر کنترلر کنش‌هایش را از لحاظ دسترسی تعریف کنم. برای مثال اگر بخواهم کنش register را در کنترلر users که بطور پیشفرض محدود است آزاد کنم به کنترلر users کد زیر را اضافه می‌کنم:

```
1.var $allowedActions = array('register');
```

حال تلاش برای نمایش users/register به صفحه users/login منتقل نخواهد شد. تا اینجای کار می‌توانید کنش‌ها را از لحاظ مجوز دسترسی به یک کاربر کنترل کنید اما این کامپوننت انعطاف پذیری خیلی بیشتری دارد در ادامه برخی از آن‌ها بصورت مختصر ذکر می‌گردد.

چگونه خطاها را تغییر دهید؟ اگر دقت کرده باشید وقتی تلاش می‌کنید یک کنش محدود شده را ببینید و به Login منتقل می‌شود خطای عدم مجوز یا هنگامی که نام و کلمه عبورتان درست نیست

خطای مربوطه در نمای login نمایش داده می‌شود با مقداردهی دو متغیر زیر در BeforeFilter می‌توانید این خطاها را تغییر دهید:

```
1.$this->Auth->authError = "Access denied!";
2.$this->Auth->loginError = "That's not the right email or password!";
```

گاهی اوقات نیاز است شرط دیگری را هم برای ورود موفق تعریف کنیم. مثلا کاربری که علاوه بر نام و کلمه عبورش اکانتش فعال هم شده باشد. یک فیلد دیگر بنام active به جدول users اضافه می‌کنیم. اگر Y بود کاربر فعال اگر N غیرفعال. حال با کد زیر می‌توانیم به کاربران فعال مجوز ورود بدهیم:

```
1.$this->Auth->userScope = array('User.active' => 'Y');
```

وقتی users/logout را برای خروج مرور می‌کنید مستقیم به users/login منتقل می‌شوید با مقداردهی کنترلر و کنش زیر می‌توایند هدایت کاربر بعد از logout را تغییر دهید:

```
1.$this->Auth->logoutRedirect = array(Configure::read('Routing.admin') =>
false, 'controller' => 'members', 'action' => 'logout');
```

سایر متغیرهای کامپوننت authentication را می‌توانید در مستندات کیک در اینجا دنبال کنید . در پست‌های آینده این کامپوننت را با ACL و Cookie ترکیب می‌کنیم تا سیستمی با مجوزها و گروه‌های مختلف بسازیم. هرچند با امکانات کامپوننت authentication به تنهایی می‌توان سیستمی با گروه‌های مختلف ایجاد کرد، نمونه‌اش را می‌توانید اینجا بیابید. اما انعطاف پذیری ACL بخصوص در پروژه‌های بزرگ که کاربران در گروه‌های مختلف قرار دارند فوق العاده قوی‌تر است.

مدل ها Models :

یک مدل نمایانگر یک جدول منحصر به فرد در پایگاه داده است هر جدول پایگاه داده که در برنامه کاربردی قصد استفاده از آن را داریم بالاجبار مدلی به عنوان نماینده باید داشته باشد. تمامی کدهای مربوط به دستیابی به داده ها، افزودن ، تغییر و اصلاح یا حذف رکورد ها از جدول در مدل واقع اند. علاوه بر آن مدل حاوی کدهایی می تواند باشد که امکان ارتباط و تعامل با سایر مدل ها را فراهم سازند. به زبان ساده تر همان مفهوم Relationship یا ارتباط را که بین جداول پایگاه داده استفاده می شود و رایج است به این طریق در سطح انتزاع به تصویر کشیده شده و استفاده می شود.

در مدل، می توان قواعدی را برای اعتبار سنجی داده ها (Data Validation) در هنگام عملیات بر روی داده ها ایجاد و مدیریت نمود. در واقع مدل ها مکان هایی هستند که منطق تجاری برنامه کاربردی در آنجا ایجاد می شود.

نکته : مدل را می توان به عنوان لایه ی داده ها (Data Layer) در برنامه کاربردی در نظر گرفت.

کنترل کننده ها Controllers:

کنترل کننده ها منطق یا جریان برنامه کاربردی را کنترل می کنند. برابر با هر درخواستی که به کنترل کننده ارجاع شود منطق کنترل کننده در مورد پاسخ تصمیم گیری نموده و آن را ایجاد می کند. در حالت عادی منطق کنترل کننده حاوی فراخوانی هایی به مدل جهت دستیابی به داده ها و مواردی نظیر کنترل ، دسترسی و ... است. در انتها کنترل کننده پاسخ یا خروجی را به نما (View) ارسال می کند.

نکته: کنترل کننده را به عنوان لایه کنترل منطق برنامه کاربردی (Logic Control) برنامه کاربردی در نظر گرفت.

نمایش View:

شامل کدهایی برای سر هم کردن اطلاعات و ساختن و نمایش خروجی می باشد. این بخش به هیچ عنوان کد عملیاتی ندارد و فقط اطلاعات را دریافت کرده، تحت قالب قرار داده و سپس جهت نمایش به Controller ارسال می کند.

مثالی از روند پردازش اطلاعات فرم ورود به سایت در معماری MVC به شرح زیر می باشد:

- کاربر با زدن دکمه ورود، فرم درخواست ورود اطلاعات را صادر می کند.

- Controller درخواست ورود را دریافت کرده، بررسی می شود که این درخواست از چه نوعی هست. نتیجه بررسی این خواهد بود که درخواست از نوع ورود به سایت بوده و سپس فرم نام کاربری و کلمه عبور از بخش view را درخواست خواهد کرد.
- Controller اطلاعات فرم را گرفته،سپس اطلاعات وارد شده را با اطلاعات موجود در دیتابیس مطابقت می دهد.
- Model دیتابیس را بررسی کرده و در صورت عدم تطابق پاسخ را به Controller برمی گرداند.
- Controller نتایج را پردازش می کند و تعیین می کند که باید فرم ورود و پیغام خطا را به کاربر برگرداند.
- Controller در صورت صحت ورود اطلاعات،کنترلر اطلاعات تعیین شده را به View ارسال می کند.
- View یک قالب خام هست که اطلاعات ارسال شده به آن را نمایش می دهد. این بخش با اطلاعاتی که از Controller دریافت می کند پر خواهد شد. Controller این پیغام رو برای View ارسال کرده و در فرم به نمایش درخواهد آمد.
- View فرم تحلیل شده را به Controller ارسال می کند.
- Controller نتیجه View را برای کاربر چاپ می کند.

:View

```
<form method="post">
    <?php echo (isset($message) ? $message : ''); ?>
    Username: <input type="text" name="username" value="" />
    Password: <input type="password" name="password" value="" />
    <input type="submit" name="action" value="login" />
```

```
</form>
```

:Model

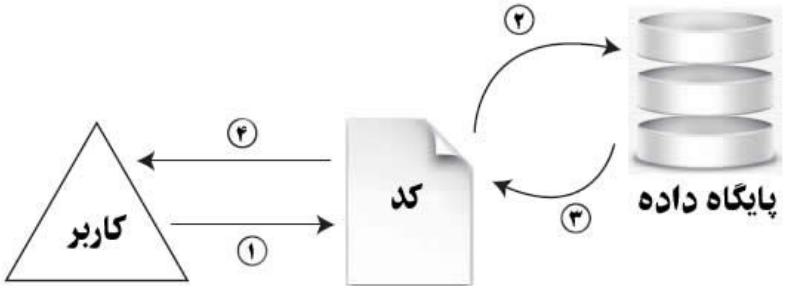
```
Class myModel {
    public function __construct() {
        $this->conn = mysql_connect('localhost', 'root', '');
        mysql_select_db('myDB');
    }

    public function getRow($sql) {
        $query = mysql_query($sql);
        return mysql_fetch_assoc($query);
    }

    public function escape($str) {
        return mysql_real_escape_string($str);
    }
}
```

:Controller

```
include './model/myModel.php';
$myModel = new myModel;

if(isset($_POST['action'])) {

    if($_POST['action'] == 'login') {
        $username = $myModel->escape($_POST['username']);
        $password = $myModel->escape($_POST['password']);
```

```
9.          $password = MD5($password);
10.
11.         $response = $myModel->getRow("
12.             SELECT * FROM users
13.             WHERE
14.                 `username`='$username'
15.                 AND `password`='$password'
16.         ");
17.
18.         if($response == false) {
19.             $message = 'Invalid Username or Password';
20.             $view = './view/login-form.php';
21.             return include $view;
22.         } else {
23.             // do something
24.         }
25.     }
26.
27.
28.     if($_POST['action'] == 'something') {
29.         // do something
30.     }
31. }
```

نحوه ارسال و دریافت اطلاعات در پی اچ پی:



تصویر۱ - جریان معمولی برای برنامه نویسی پی اچ پی

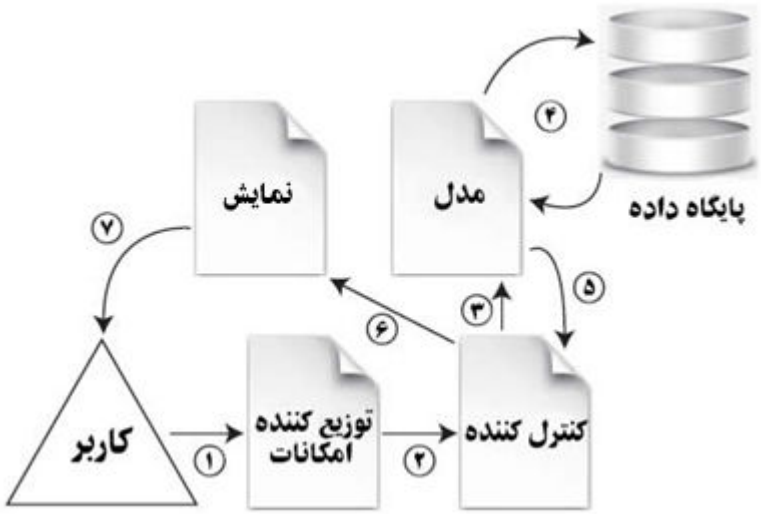
در معماری و کدنویسی عادی به طور خلاصه همه چیز در کدهای پی اچ پی خلاصه شده است.

توسعه دهندگان می توانند با بکارگیری تابع include() ،توابع مختلف را از سایر فایل ها فراخوانی کنند که و از افزونگی جلوگیری نمایند.MVC یک تکنیک موثر برای ساخت کلاس اشیا است. هدف اصلی در MVC این است تا مطمئن شوید که هر تابع از برنامه یک بار و تنها یک بار ، نوشته شده است در نتیجه با تولید کد های ساده افزونگی کاهش یابد این همان معماری ساده ولی کاربردی است که در فریم ورک ها بکار می رود.

۱- کاربر درخواستی را بصورت تایپ آدرس یا لینک به وب سرور ارسال می‌کند.
۲ – این کد اطلاعات را پردازش می کند و سپس درخواست های پایگاه داده را بصورت مستقیم به آن ارسال می کند.

۳ – این کد هر گونه خروجی مربوط به پایگاه داده و فرآیند داده ها را دریافت می کند.
۴ – خروجی کدها تولید شده و در مرورگر کاربر به نمایش در می آید.

نحوه ارسال و دریافت اطلاعات در فریم ورک کیک:



تصویر۲ – چگونگی استفاده فریم ورک کیک از معماری MVC

۱. کاربر صفحه درخواست را با تایپ کردن یک آدرس درخواست می کند.

آدرس های سایت های طراحی شده با این فریم ورک معمولا این ساختار را دارند:

http://{Domain}.com/{Application}/{Controller}/{Action}/{Parameter 1, etc}.

۲. توزیع کننده ساختار آدرس را تجزیه می کند و مشخص می کند که کدام کنترل کننده آن را به اجرا درآورده است. همچنین به موازات آن عمکرد ها و پارامترها را به کنترل کننده منتقل می کند.

۳. تابع در کنترل کننده ممکن است لازم باشد که مسئولیت رسیدگی به اطلاعات را بیش از پارامترهای فرستاده شده توسط توزیع کننده بر عهده داشته باشد و درخواست های پایگاه داده را به مدل اسکریپت ارسال می کند.

۴. مدل اسکریپت نحوه تعامل با پایگاه داده با استفاده از درخواست های ارسال شده توسط کنترلر تعیین می کند.این مدل تمام درخواستها را با پایگاه داده اجرا می کند. مرتب سازی بر اساس دستورالعمل ها انجام می شود.

۵. مدل، هر گونه اطلاعات را از پایگاه داده خارج می کند و یا به آن ارسال می کند و خروجی را به کنترل کننده برمی گرداند.

۶. کنترل کننده اطلاعات را پردازش کرده و خروجی آن را جهت نمایش ارسال می کند.

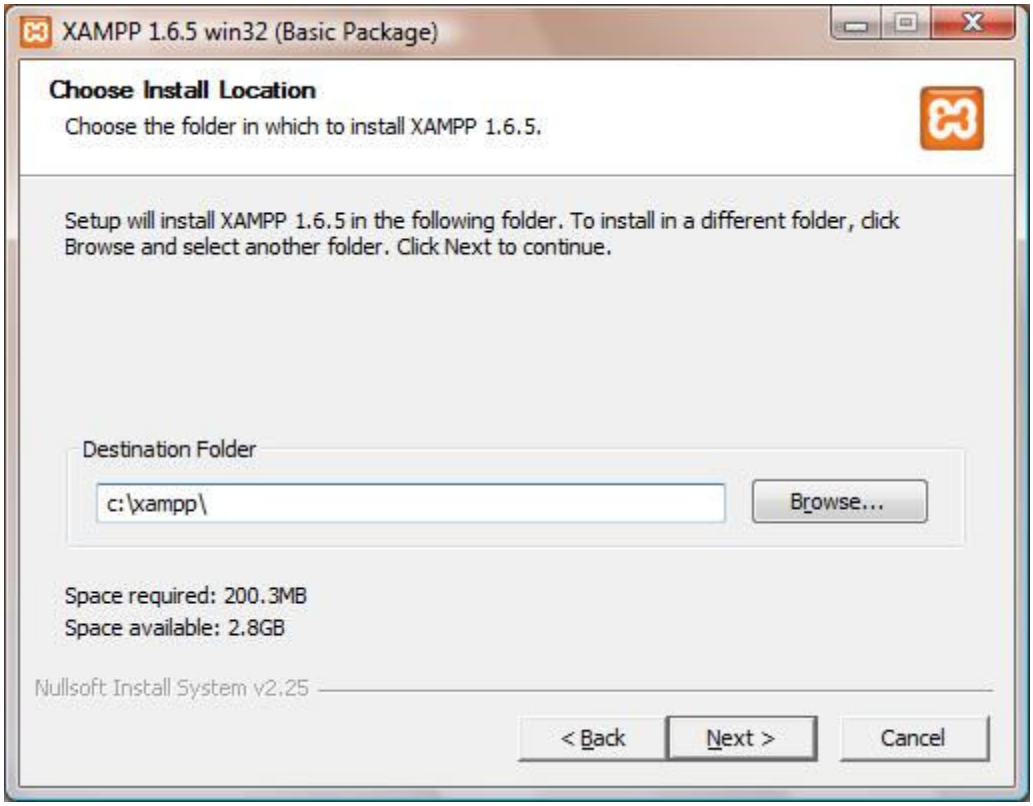
۷. طرح یا قالب داده های خروجی به کنترل کننده اضافه می شود و خروجی آن را به مرورگر کاربر می فرستد.

نصب مفسر php:

جهت تست و اجرای کدهای php و استفاده از فریم ورک ها نیاز است تا برنامه مخصوص به ان نصب گردد، بدین جهت باید مفسر xampp و یا wampp را از سایت های سازنده آن دانلود و مطابق شکل نصب و پیکربندی نمود.

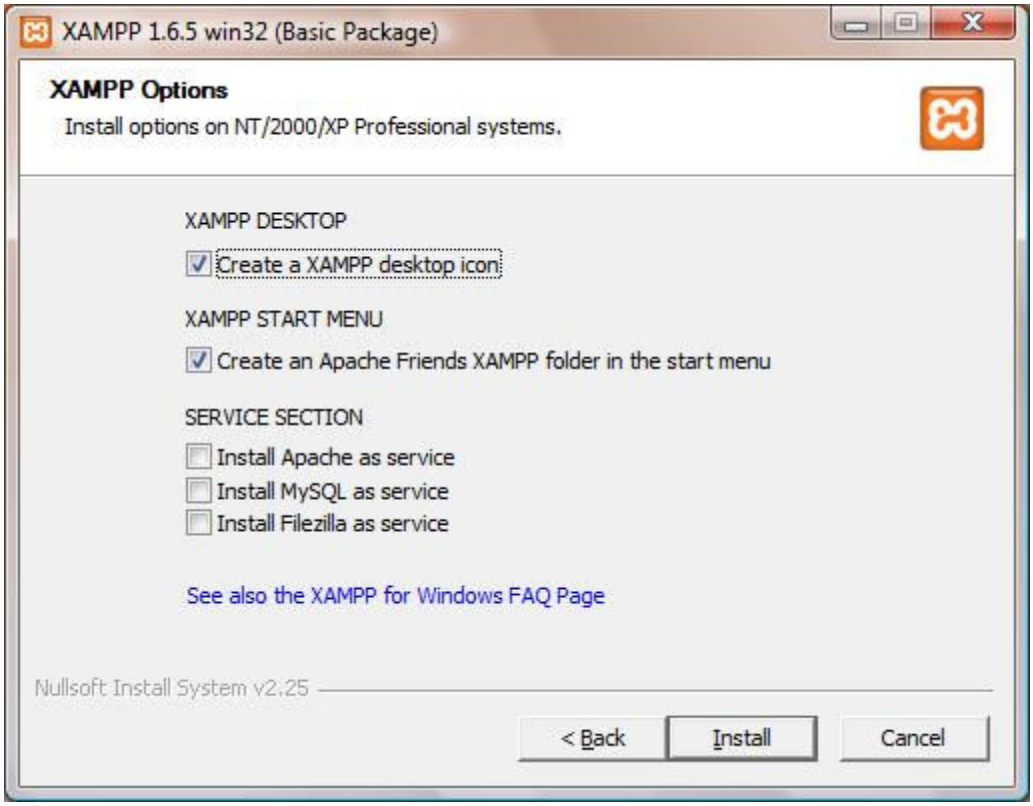
آموزش نصب Xampp:

شکل ۱. مراحل نصب زمپ



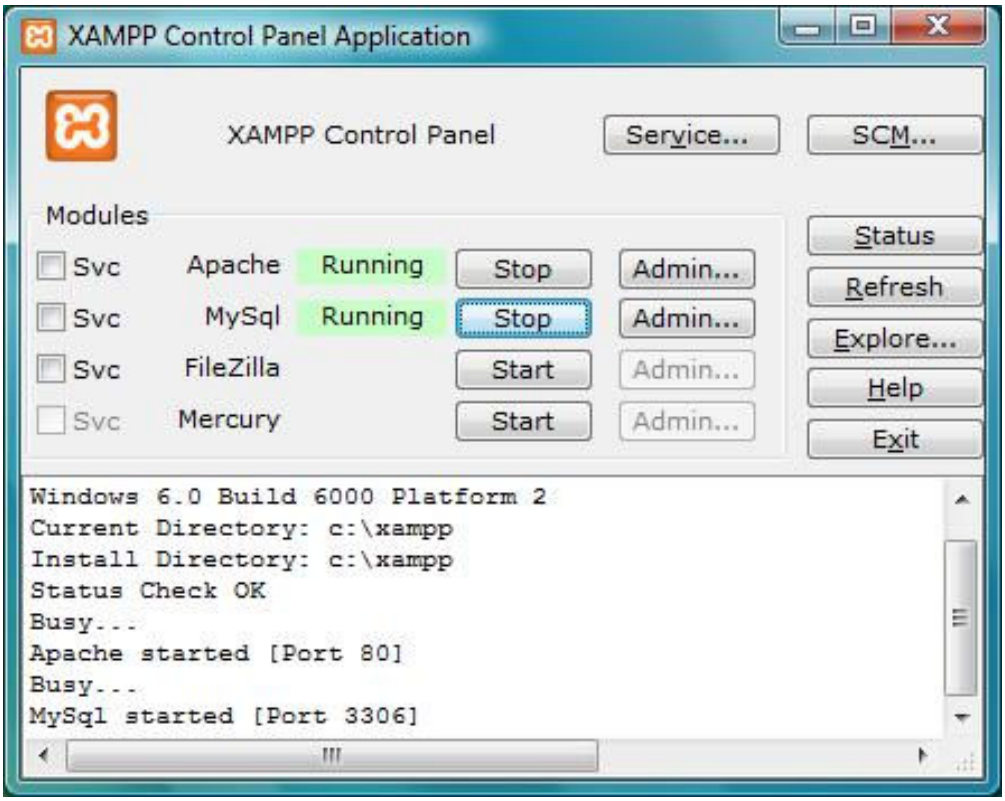
شکل ۲. انتخاب مسیر نصب

در صورتی که میخواهید Apache و MySQL به عنوان سرویس های ویندوز نصب بشند گزینه های Install Apache as Service و Install MySQL as Service رو تیک بزنید! انتخاب این گزینه ها اختیاریه. تنها نکته ای که هست اینه که در صورتیکه IIS رو سیستم تون نصبه و Apache رو نصب کنید باید حتما پورت IIS یا Apache رو تغییر بدید وگرنه Apache پیغام خطای Port Busy بهتون میده! بعد روش تغییر پروت IIS و Apache رو میبینیم. خوب تا اینجا کار تقریبا تمومه فقط باید دکمه Install رو بزنید.



شکل ۳. تنظیمات نصب

- بخش تنظیمات نصب Xampp که با آن میتونید Apache و MySQL رو Start یا Stop کنید! توجه کنید که اگر تیک Install as Service رو انتخاب باشید باید هر بار که می خواهید روی وب سایتتان کار کنید Apache و MySQL فعال باشند یعنی باید دکمه Start کنار آنها را کلیک کنید (Runningرو کنار Apache و MySQL نمایش بده)



شکل ۴. فعال کردن آپاچی و Mysql

حالا مرورگر خود را باز کنید و در آدرس بار مرورگر تایپ کنید localhost یا ۱۲۷,۰,۰,۱ ، اگر همه چیز خوب پیش رفته باشد باید صفحه زیر را ببینید که بهتر است در این صفحه زبان صفحات Xampp رو انتخاب کنید.

شکل ۵. ورود به محیط اصلی زمپ

اطلاعات پیش فرض جهت ارتباط با پایگاه داده:

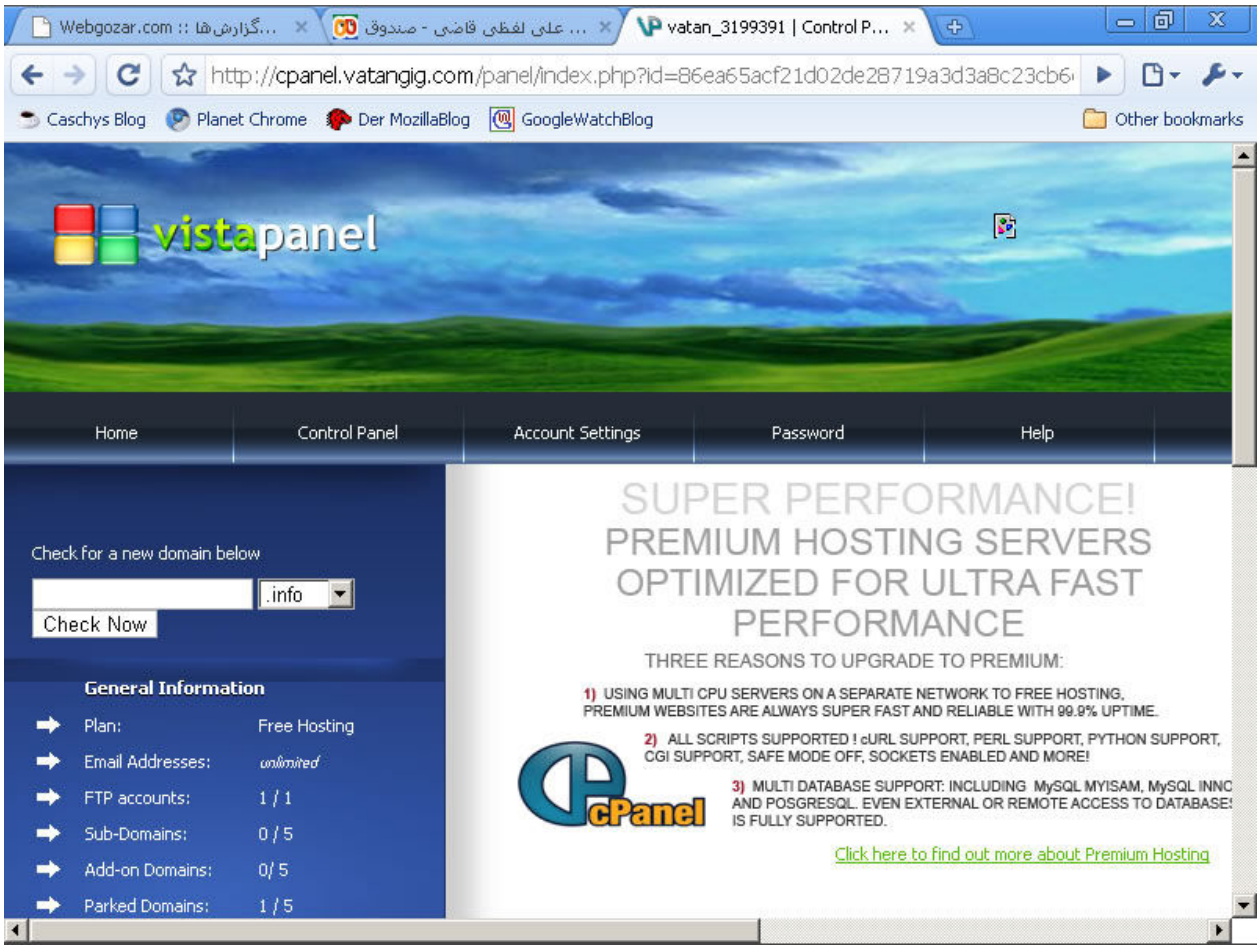
localhost : Host Name

root : Username

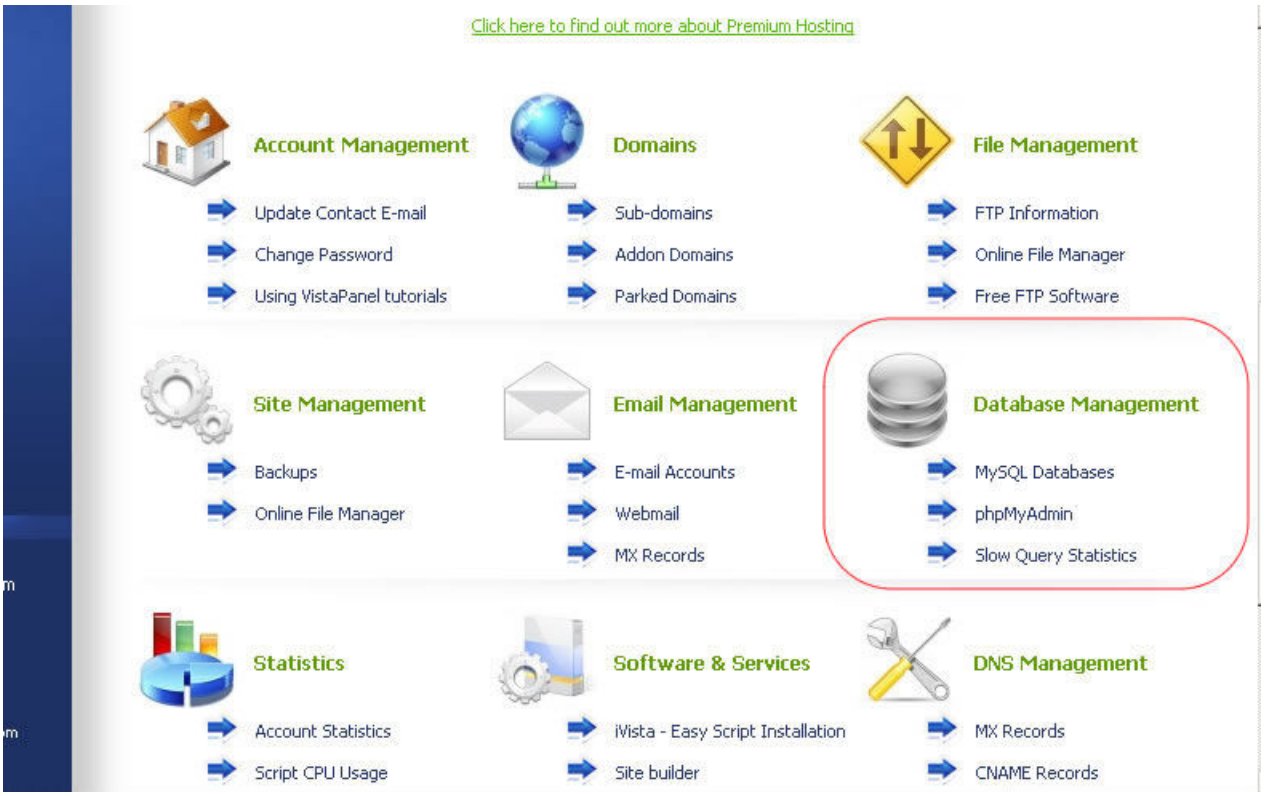
Password: ندارد (در صورتی هنگام نصب تعریف نکرده باشید)

نحوه ایجاد پایگاه داده بر روی سرور(هاست) هم همانند ایجاد آن بر روی لوکال است با این تفاوت که نام سرور، نام کاربر و همچنین رمز عبور آن ممکن است تفاوت هایی داشته باشد که البته این اطلاعات از طریق پنل هاست قابل دریافت و مشاهده است.

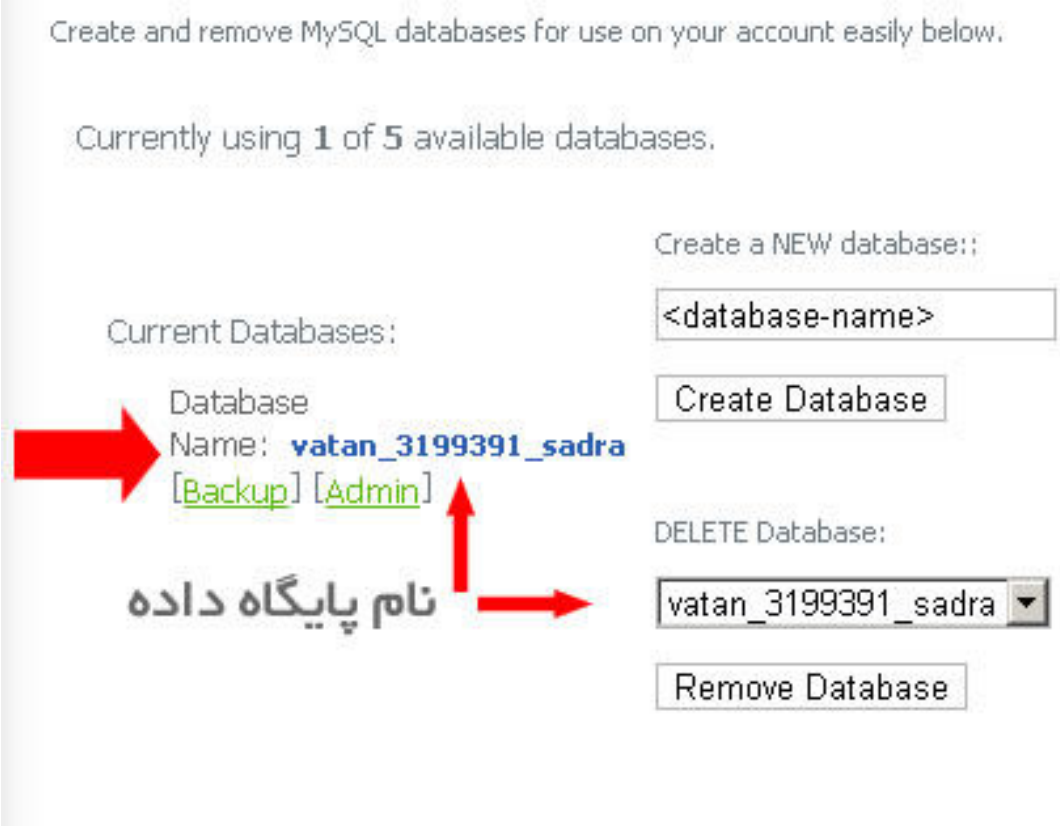
جهت آشنایی بیشتر با ایجاد پایگاه داده و اطلاعات مربوط به چگونگی اتصال به آن به تصاویر مربوطه توجه فرمایید.



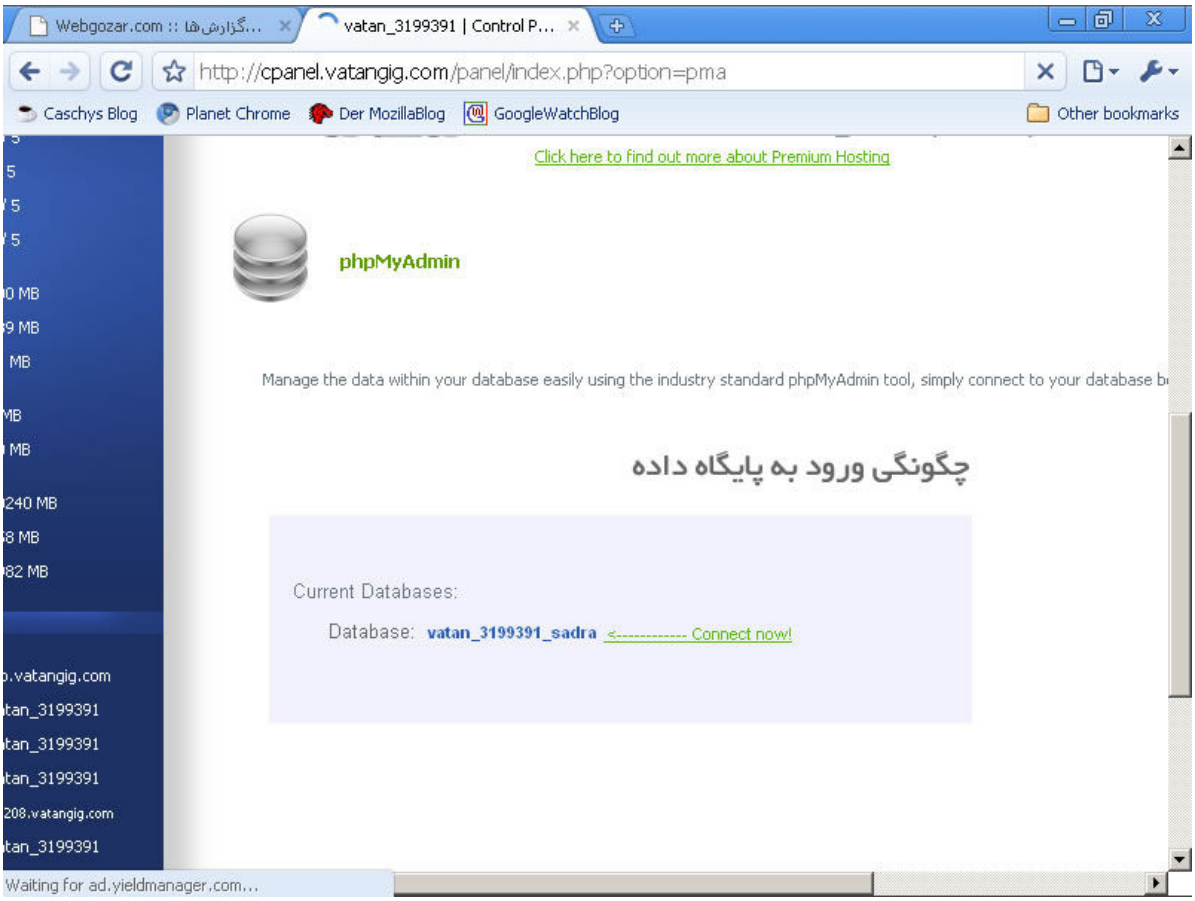
شکل ۶. ایجاد پایگاه داده Mysql در سرور وب



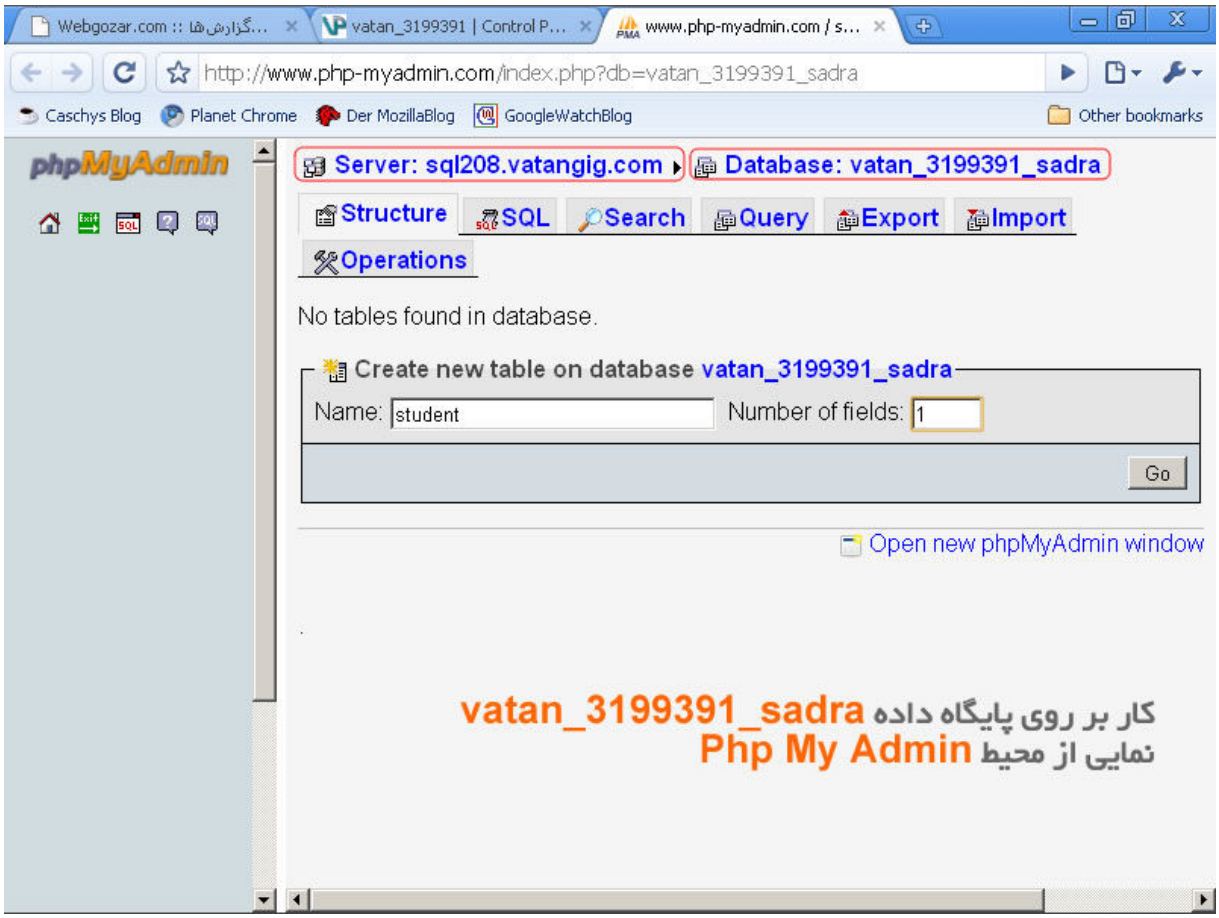
شکل ۷. انتخاب بخش مدیریت پایگاه داده ها



شکل ۸. نام دهی و ایجاد پایگاه داده جدید



شکل ۹. ورود به پایگاه داده



شکل ۱۰. ورود به بخش php Myadmin و دسترسی به محتویات پایگاه داده

نحوه نصب و پیکربندی CakePHP:

اگر مفهوم اصلی فریم ورک ها را به درستی درک کرده باشیم باید بدانیم که جهت اجرای کدهای برنامه نیاز به هسته اصلی فریم ورک داریم، در واقع این فریم ورک است که دستورات برنامه نویسی را کوتاه و به آن معنای واحد می بخشد. جهت تسریع نصب و پیاده سازی برنامه ها بهتر است تا آن ها را در بخش لوکال سیستم شخصی امتحان کنیم به همین جهت مراحل نصب فریم ورک بر روی لوکال خانگی در نظر گرفته شده است، در صورتی که بخواهیم کدها را بر روی فضای وب انتقال دهیم بایستی کلیه

کدهای دایرکتوری را آپلود نموده و فایل های پیکربندی را مطابق با نام کاربری، نام پایگاه داده و رمز عبور آن تنظیم نماییم.

جهت تبدیل سیستم شخصی به لوکال سروری که بتواند کدهای php را تفسیر نماید نیاز به برنامه Xampp داریم که با جستجو در اینترنت می توان این برنامه را نصب کرد. برنامه زمپ بصورت پیش فرض پایگاه داده مخصوص php یعنی phpmyadmin را نصب می کند پس از نصب این نرم افزار و راه اندازی آن می تواند به شاخه لوکال به آدرس http://localhost/phpmyadmin دست پیدا کرد، از این پس می توانید با انتقال فایل های php به آدرس روت در مسیر نصب: C:\xampp\htdocs و تایپ مسیر آن در لوکال هاست در مرورگر برنامه های نوشته شده خود را اجرا نمایید.

پس از نصب و راه اندازی سرور لوکال به وب سایت سازنده فریم ورک به آدرس www.cakephp.org مراجعه کرده و آخرین نسخه این فریم ورک را دریافت نماییم، سپس فایل را در مسیر C:\xampp از حالت فشرده خارج کرده می کنیم و نام فولدر را به cake تغییر نام می دهیم.

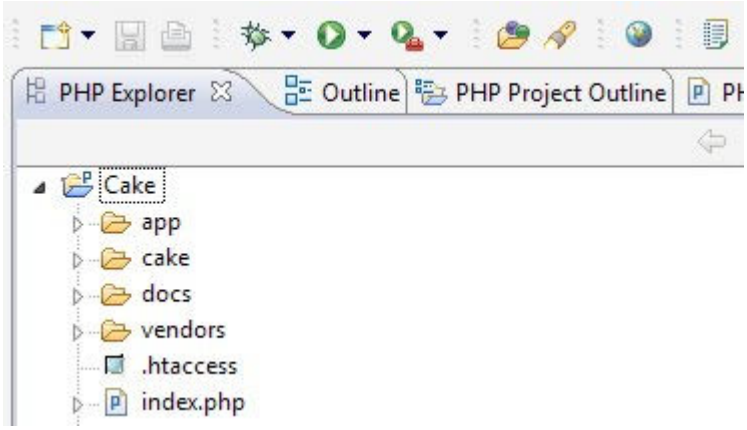
برای شناسایی فایل های سیستمی نیاز داریم تا تنظیماتی را نیز در آن بخش اعمال کنیم.

بر روی My computer کلیک راست کرده properties را بر می گزینیم. حال به بخش change setting می رویم و از آنجا زبانه Advanced را انتخاب می کنیم. در آنجا Environment variable را انتخاب کرده و سپس این کد را اضافه می کنیم. متغییر Path را ویرایش کرده و این مسیر را می افزاییم.

C:\xampp\cake\cake\console\;C:\xampp\php\;C:\xampp\mysql\bin;

این دستورات اجازه برقراری میان کنسول فریم ورک و پایگاه داده را ایجاد خواهد کرد.

پس از دریافت کیک آن را به وب سرور خود منتقل کنید ساختار فولدرها و فایل‌های پایه کیک به صورت زیر است:



از فولدر App شروع می‌کنیم، این همان فولدری است که بیشترین کار توسعه پروژه در آن انجام می شود. هر چند نام فایل‌ها و فولدرهای موجود در این شاخه گویای همه چیز است اما اجازه دهید نگاه دقیقتری به محتویات آن داشته باشیم:

Config:فایل‌های مربوط به پیکربندی برنامه در اینجا قرار دارند. دقت کنید این فایلها تنظیمات مربوط به هسته فریم ورک را شامل نمیشود (تنها تنظیمات مختصری در مورد پروژه‌ی کاری نظیر جزییات اتصال به پایگاه داده، محل قرار گیری فایل‌ها روی سرور و...) باید یادتان باشد هنگام پیاده سازی بر روی وب اطلاعات صحیح مسیر فریم ورک و فایل ها را به درستی تنظیم نمایید.

Controllers: کنترل‌گرها و مولفه‌های (کامپونت‌های) پروژه در این فولدر قرار خواهند گرفت. پیش از این گفتیم کنترل کننده (کنترلر) درخواست‌هایی که توسط کلاینت ایجاد شده را بررسی و به یک مسیر صحیح هدایت می کند.

Locale: برای فایل‌های مربوط به بومی‌سازی پروژه مورد استفاده قرار می‌گیرد. برای مثال قصد دارید پروژه‌ای را شروع کنید که به چند زبان مختلف موجود باشد، فایل‌های حاوی سایر زبان‌ها در اینجا قرار می‌گیرند.

Models: مدل‌های پروژه در این فولدر قرار خواهند گرفت. قبلاً گفتیم که که مدل‌ها به نوعی به داده برنامه اشاره دارد. معمولاً جداول پایگاه‌های داده در اینجا مورد بحث قرار می‌گیرند بطوریکه به ازای هر جدول پایگاه داده، یک فایل در این فولدر قرار خواهد گرفت که بدان جدول اشاره دارد.

Plugins: بسته های پلاگین چنانچه در صورت نیاز برای پروژه نوشته شود در اینجا قرار خواهد گرفت. پس از اتمام پروژه می توانید افزونه‌هایی بنویسید که بدون تغییر اساسی در پروژه قابلیت‌های جداگانه‌ای بدان اضافه کند.

Tmp: در این فولدر اطلاعات موقت ذخیره می‌شوند. نحوه ذخیره اطلاعات به تنظیمات کیک بستگی خواهد داشت، اما بطور معمول توضیحات مدل، فایل‌های ثبت وقایع و گاهی اوقات اطلاعات نشست‌ها در این فولدر ذخیره می‌شوند.

Vendors: کلاس‌ها یا کتابخانه‌هایی که توسط شخص برنامه‌نویس برای استفاده در برنامه نوشته می‌شوند باید در اینجا قرار بگیرند. جلوتر خواهید دید که با یک تابع توکار براحتی می‌توانید این محتویات را به پروژه الحاق و از آنها استفاده کنید.

Views: صفحات بصری که به نوعی با کاربر در ارتباط است شامل صفحات مربوط به طرح‌بندی، عناصر، خطاها، راهنما ها در اینجا قرار می‌گیرند.

Webroot: در نهایت این فولدر در نصب برنامه می‌بایستی به عنوان (root شاخه اصلی) پروژه بکار گرفته شود. همچنین این فولدر فایل‌های مربوط به CSS ، تصاویر و جاوااسکریپت را در بر خواهد داشت.

جهت کار با کنسول کیک در command promp دستور cake bake foldername را تایپ می کنیم.

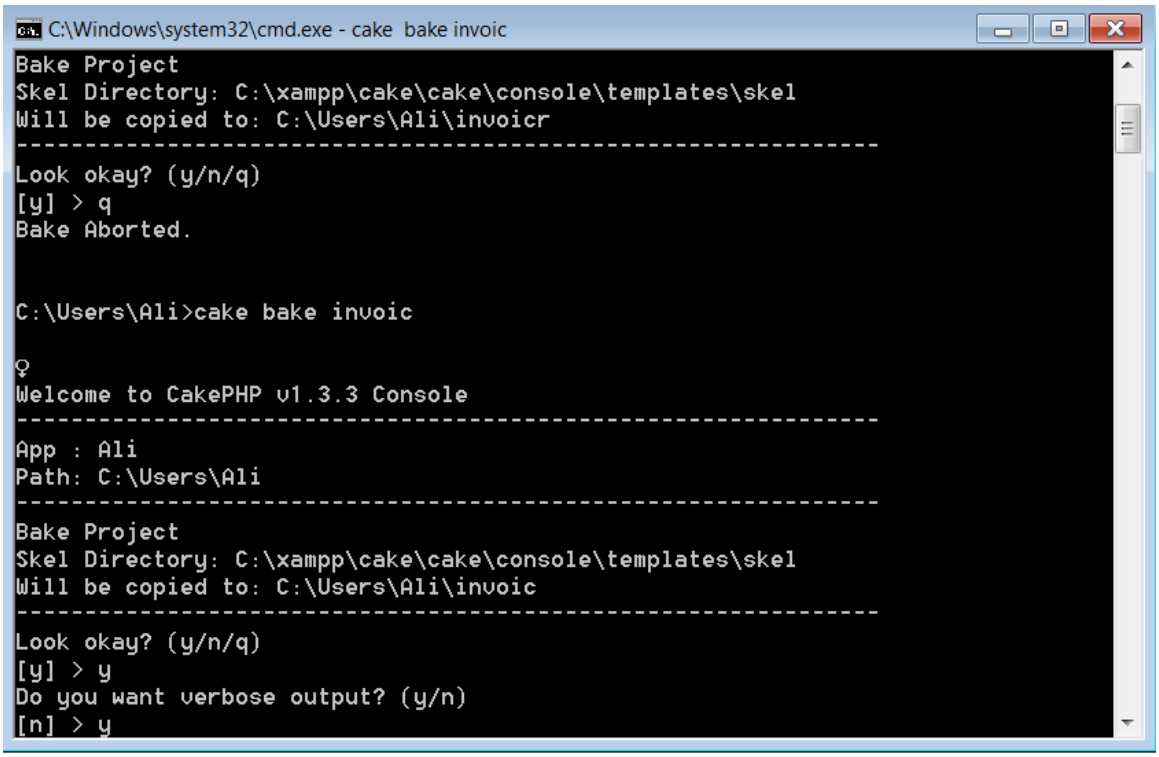
از آنجا که ما قصد ساخت یک سیستم مدیریت فروش را داریم به جای foldername نام invoice را تایپ می کنیم.

جهت ایجاد و ساخت شاخه invoice و فایل های اولیه مربوطه به فریم ورک به سوالات مطرح شده پاسخ می دهیم:

تائید سوال اول به منزله قبول ایجاد دایرکتوری invoice می باشد و تایید سوال دوم باعث ایجاد فایل های فریم ورک کیک در آن مسیر خواهد شد، جهت تایید و ایجاد فایل ها به هر دو سوال پاسخ مثبت می دهیم.

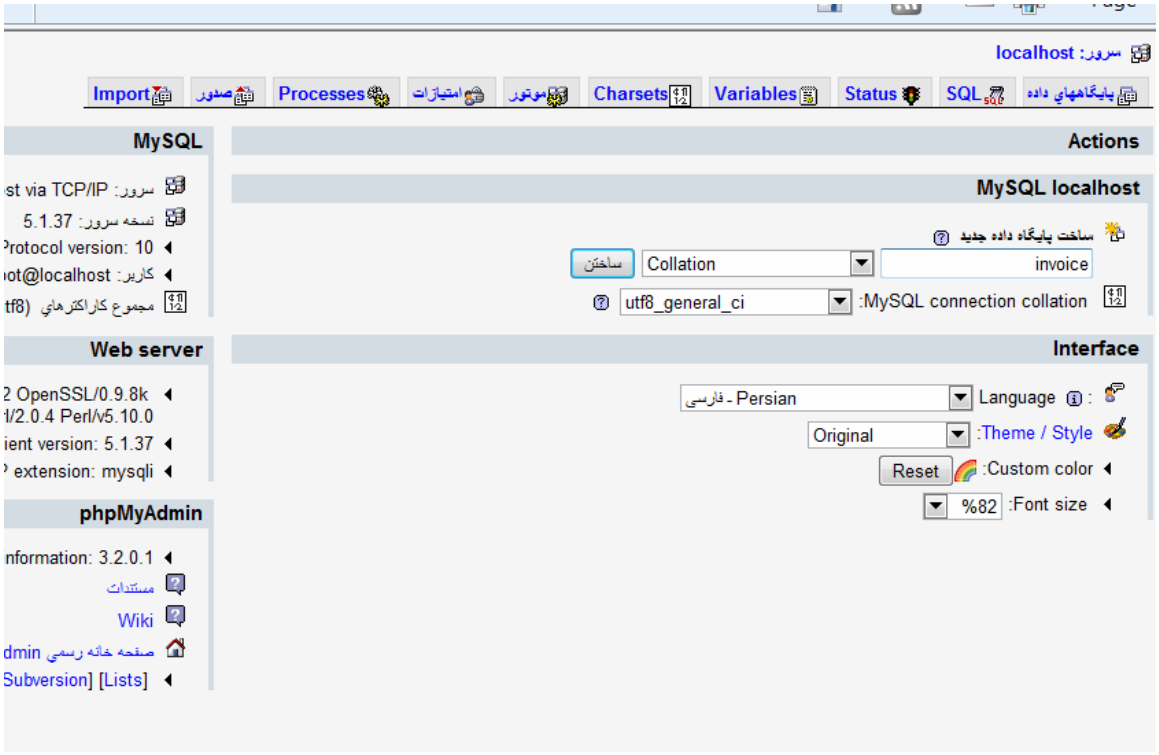
Look okey? Y =yes

Do you want verbose output? Y = yes



تنظیم پایگاه داده:

جهت ایجاد یک پروژه بایستی یک پایگاه داده داشته باشیم، برای ایجاد یک پایگاه داده جدید بهتر است آدرس http://localhost/phpmyadmin را در مرورگر تایپ کنیم و به بخش مدیریت پایگاه داده مراجعه کنیم.



نام پایگاه داده یعنی invoice را تایپ کرده و دکمه ساختن را انتخاب می کنیم.

حال پایگاه داده ساخته شده باید توجه کنیم که این پایگاه داده هیچ گونه جداول و اطلاعاتی را شامل نمی شود و باید پس از طراحی ملزومات اولیه جداول مورد نیز به پایگاه داده انتخاب شده import گردد.

طراحی پایگاه داده سیستم مدیریت فروش:

جهت طراحی پایگاه داده می توان از یک ویرایشگر ساده همانند نوت پد ویندوز استفاده کرد، طراحی پایگاه داده باید مناسب با نیازها و پارامترهای احتمالی در سیستم طراحی و در نظر گرفته شود، همچنین باید دید کلی نسبت به ملزومات این سیستم در نظر گرفت.

نیازهای سیستم مدیریت فروش آنلاین:

امکاناتی که قرار است برای این سیستم مدیریت فروش در نظر گرفته شود به شرح زیر می باشد:

۱. امکان عضویت کاربران و ایجاد موضوع، مشتریان و صدور فاکتور برای هر خرید ثبت شده
۲. تعیین وضعیت پرداخت و فروش کالاها شامل: در حال بررسی، انجام شده، تاریخ گذشته و در حال انجام...
۳. نمایش وضعیت پرداخت ها بصورت جداگانه در صفحه رومیزی
۴. نمایش تاریخ هجری شمسی
۵. امکان گرفتن خروجی از فاکتور در قالب PDF
۶. امکان بررسی قالب مقادیر ورودی و صحت اطلاعات درج شده در فرم ها
۷. امکان جستجوی الفبایی در اسامی مشتریان

مفاهیم اولیه برای طراحی پایگاه داده:

طراحی پایگاه داده و ایجاد نمودار ارتباط موجودیت ها (ERD) یکی از مهمترین بخش های چرخه حیات توسعه یک نرم افزار است که در برخی موارد از آن به عنوان مهمترین بخش نیز نام برده می شود. مدل صحیح و به هنگام (Up To Date) اطلاعات می تواند به عنوان مهمترین ابزار مرجع برای مدیران بانک اطلاعاتی (DBAs) ، پیاده کنندگان نرم افزار و سایر اعضاء تیم توسعه دهنده نرم افزار باشد . فرآیند ایجاد مدل داده به تیم توسعه دهنده کمک می کند تا به پرسش های مطرح شده توسط کاربران نهائی سیستم پاسخ دهند . همچنین طراحی کارا و موثر پایگاه داده به تیم توسعه دهنده این امکان را می دهد تا سیستم را از همان ابتدا در فرم مناسب پیاده سازی نمایند . ساخت سیستم با کیفیت فوق الذکر این امکان را به تیم توسعه دهنده خواهد داد تا زمان کلی انجام پروژه را کاهش دهند ، که در واقع این امر موجب کاهش هزینه های توسعه پروژه نیز خواهد شد .

طراحان خوب و خبره بانک های اطلاعاتی، مبانی و اصول نرمال سازی پایگاه داده را همواره در خلال طراحی به خاطر داشته و آن را به کار خواهند گرفت.

نرمال سازی فرآیندی در خلال طراحی پایگاه داده است که با چهار هدف عمده ذیل دنبال می شود :

- به حداقل رسانی افزونگی اطلاعات
- به حداقل رسانی تغییر ساختار اطلاعات
- به حداقل رسانی I/O سرور به منظور کاهش تعداد تراکنش ها (Transactions)
- و در نهایت حفظ یکپارچگی اطلاعات

برای طراحی بانک اطلاعاتی نرم افزار و مدل سازی آن می بایست اصول و تکنیک های ذیل را مد نظر داشت و از آنها استفاده نمود .

موجودیت (Entity) ، مجموعه ای از چیزهائی است که مربوط به بانک اطلاعاتی سیستم مورد نظر می باشد و یا به تعبیر دیگر هر آنچه که می خواهید در سیستم راجع به آن اطلاعات جمع آوری و نگهداری نمائید را شامل می شود . در مدل فیزیکی ، موجودیت تبدیل به جدول (Table) می شود .

خصلت (Attribute) یکی از مشخصه های توصیفی و یا مقداری موجودیت می باشد . در مدل فیزیکی یک خصلت به یک ستون (Column) و یا فیلد (Field) تبدیل می شود .

کلید اصلی (Primary Key) خصلت و یا ترکیبی از خصلت ها در یک موجودیت است که تضمین کننده یکتا بودن هر رخداد از موجودیت می باشد . خصلت یا خصلت های کلید اصلی نمی توانند فاقد ارزش باشند (NULL) و معمولا" کمتر تغییر می کنند . معمولا" سعی می شود جهت انتخاب کلید اصلی از خصلت هائی استفاده شود که کارائی بیشتری داشته و بهترین معرف موجودیت باشند (کارائی

یک فیلد از نوع Integer به مراتب بیشتر از فیلدی از نوع Char است) . در صورتیکه نتوان در یک موجودیت خصلت یا خصلت هائی برای کلید اصلی شدن یافت ، آنگاه کلیدهای دستی برای این کار را ایجاد می کنیم که به آنها کلید Artificial می گویند .

ارتباط (Relationship) ، ارتباط منطقی بین دو موجودیت است .

در فریم ورک کیک ارتباطات به چهار صورت امکان پذیر می باشد و با نام های زیر مطرح می گردد:

۱. یک به یک hasOne
۲. یک به چند hasMany
۳. چند به یک belongsTo
۴. چند به چندhasAndBelongsToMany

یک ارتباط در واقع نشان دهنده قوانین کاری حاکم بر پروژه و اطلاعات آن است که معمولا" به صورت جملات فعلی توصیف می گردد. مثل ارتباط بین موجودیت کارمند و دپارتمان که به صورت جمله ذیل بیان می شود :

"کارمند شاغل است در دپارتمان" در این مثال ارتباط بین موجودیت کارمند و دپارتمان با جمله "شاغل است" توصیف می گردد .

دو نوع ارتباط می تواند بین موجودیت ها وجود داشته باشد :

- ارتباط یک به چند (has Many) در این نوع ارتباط، هر رخداد از موجودیت والد با چندین رخداد در موجودیت فرزند ارتباط دارد . به عنوان مثال چندین کارمند می توانند در یک دپارتمان شاغل به کار باشند .

- ارتباط چند به چند (hasAndBelongsToMany) . در این نوع ارتباط ، چند رخداد از یک موجودیت با چند رخداد از موجودیت دیگر ارتباط دارند . به عنوان مثال اگر یک کارمند بتواند در چند دپارتمان شاغل به کار باشد ، آنگاه ارتباط بین موجودیت کارمند و دپارتمان یک ارتباط چند به چند است . ارتباط چند به چند در طراحی پایگاه داده پذیرفته شده نیست چراکه علاوه بر افزونگی اطلاعات موجب عدم یکپارچگی اطلاعات نیز می گردد ، از اینرو باید این ارتباط طبق فرم چهارم نرمال سازی تبدیل به دو ارتباط یک به چند شود . همانطور که در مقاله نرمال سازی بانک های اطلاعاتی اشاره گردید برای حل این مشکل کافی است یک موجودیت واسط که به آن موجودیت XREF می گویند ایجاد و خصلت های کلید اصلی هردو موجودیت را به این موجودیت رابط منتقل نمود . با این عمل هریک از موجودیت های اصلی به عنوان والد این موجودیت رابط تلقی شده و یک ارتباط یک به چند بین آنها برقرار خواهد شد. در نتیجه یک ارتباط چند به چند تبدیل به دو ارتباط یک به چند خواهد شد . لازم به ذکر است که بسیاری از سیستم های مدیریت بانک های اطلاعاتی رابطه ای (نظیر MS SQL Server) از ارتباط چند به چند پشتیبانی نمی کنند .

کلید خارجی (Foreign Key) . هرگاه خصلت(های) کلید اصلی موجودیت والد در موجودیت فرزند وجود داشته باشد (بر اساس ارتباط تعریف شده بین دو موجودیت) آنگاه این خصلت ها در موجودیت فرزند ، کلید خارجی نامیده می شوند . در واقع نمی توان هیچ رخدادی در موجودیت فرزند (که دارای کلید خارجی است) ایجاد نمود که رخداد مربوط به آن (بر اساس محتوای خصلت کلید خارجی) قبلا" در موجودیت والد ایجاد نشده باشد . آنگونه که از توصیف فوق استنباط می شود کلید خارجی تضمین کننده یکپارچگی اطلاعات در داخل پایگاه داده است چرا که باعث می شود که هیچ فرزند بدون والدی در بانک اطلاعاتی نداشته باشیم .

ارتباط (RelationShip) بین دو موجودیت به دو مدل ذیل دسته بندی می گردد:

- ارتباط تعریف شده (identifying Relationship) . اگر کلید اصلی جدول والد بخشی (یا تمام) از کلید اصلی جدول فرزند باشد و یا به تعبیر دیگر بخشی از کلید اصلی موجودیت فرزند کلید خارجی نیز باشد ، در این حالت ارتباط مابین این دو موجودیت از نوع تعریف شده است .
- ارتباط تعریف نشده (Non-Identifying Relationship) ، برخلاف مورد فوق اگر کلید اصلی جدول والد در جدول فرزند وجود داشته باشد اما نه به عنوان بخشی از کلید اصلی آن و صرفاً به عنوان یک خصلت غیر کلید ، در این حالت ارتباط بین این دو موجودیت از نوع تعریف نشده می باشد . ارتباط تعریف نشده خود دارای دو حالت متفاوت به شرح ذیل است : mandatory non-identifying relationship ، زمانی است که خصلت کلید خارجی در موجودیت فرزند نتواند فاقد ارزش باشد (Not Allow NULL) non-mandatory non-identifying relationship ، زمانی است که خصلت کلید خارجی در موجودیت فرزند بتواند فاقد ارزش باشد (Allow NULL)

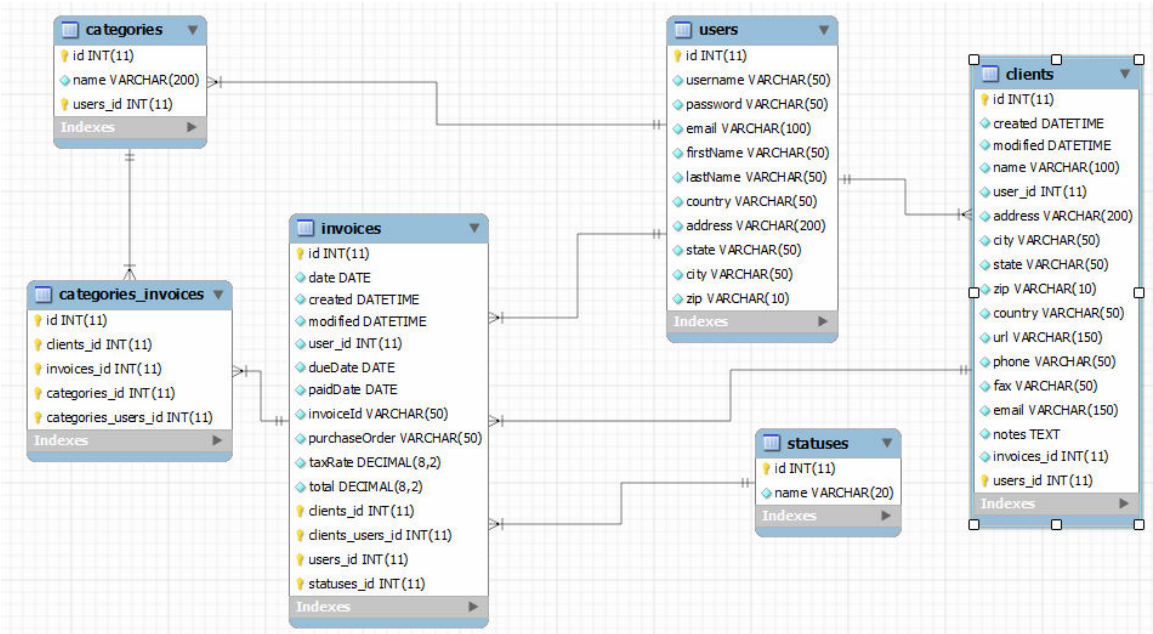
Cardinality ، به ما در فهم بیشتر ماهیت ارتباط مابین موجودیت والد و فرزند کمک می کند . جهت تشخیص Cardinality یک ارتباط کافی است به سئوال ذیل پاسخ داده شود : " چه تعداد رخداد از موجودیت فرزند مرتبط است با هر رخداد از موجودیت والد؟ " چهار نوع Cardinality مختلف به شرح ذیل وجود دارد :

- One To Zero or Many به این معنی که هر رخداد از موجودیت والد با هیچ و یا چند رخداد از موجودیت فرزند مرتبط است . به این نوع Common Cardinality می گویند.

- One To One Or Many به این معنی که هر رخداد از موجودیت والد با حداقل یک و یا چند رخداد از موجودیت فرزند مرتبط است . به این نوع P Cardinality می گویند .
- One To Zero Or One ، به این معنی که هر رخداد از موجودیت والد با هیچ و یا تنها یک رخداد از موجودیت فرزند مرتبط است . به این نوع Z Cardinality می گویند .
- One to Exactly N ، به این معنی که هر رخداد از موجودیت والد باید با N رخداد از موجودیت فرزند مرتبط باشد . به این نوع N Cardinality می گویند .

طراحی پایگاه داده سیستم فروش

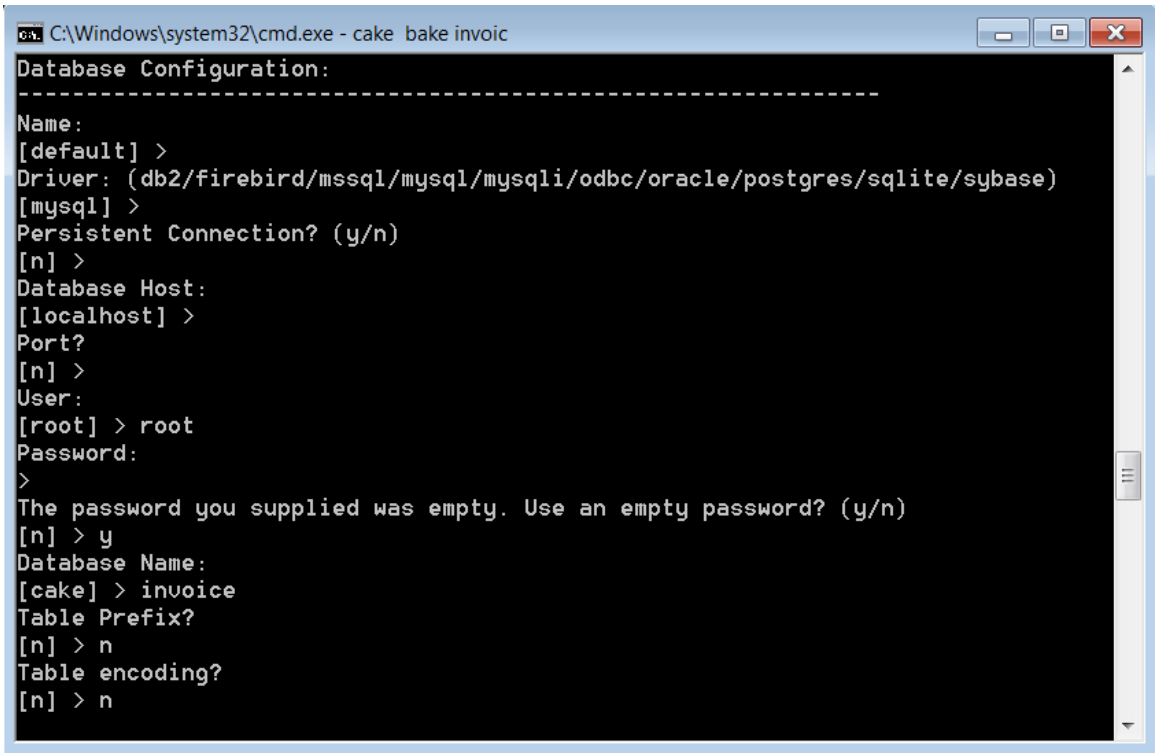
برای طراحی پایگاه داده سیستم از برنامه معروف MySQL workbench استفاده شده است، در این برنامه به راحتی می توان با ایجاد جداول و ایجاد رابطه بصورت گرافیکی به پایگاه داده دلخواه دست یافت، همچنین استفاده از این برنامه قابلیت کار گروهی و ایجاد شناخت کلی از برنامه را برای ما ایجاد خواهد نمود.



مجدد به command prompt باز می گردیم و تنظیمات مربوط به پایگاه داده را طبق شکل انجام می دهیم:

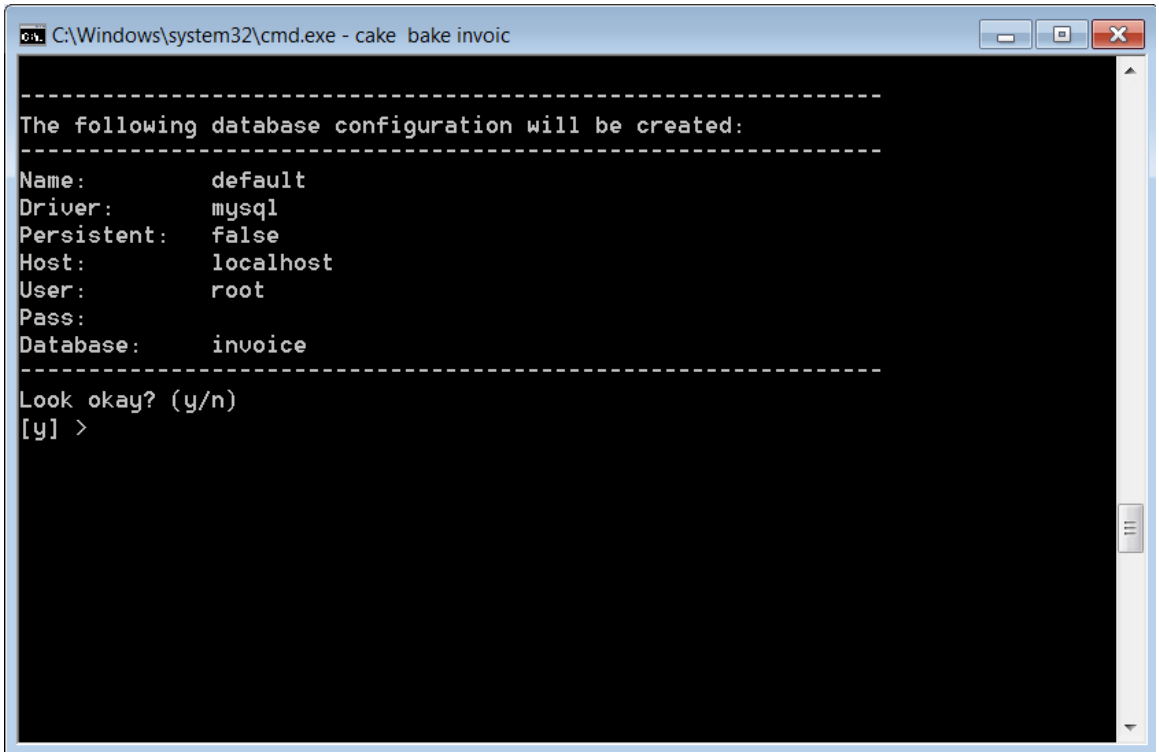
باید دقت کنیم که اطلاعات مربوط به نام کاربری مدیر، رمز و نام پایگاه داده در سیستم های مختلف بنا به انتخاب کاربران متفاوت است و باید در ورود اطلاعات صحیح دقت شود.

دستورات فوق اطلاعات پیش فرض فایل database.php در مسیر config را مورد ویرایش قرار می دهد در صورت تغییر اطلاعات سرور می تواند این فایل را مطابق با تنظیمات محلی ویرایش نمود.

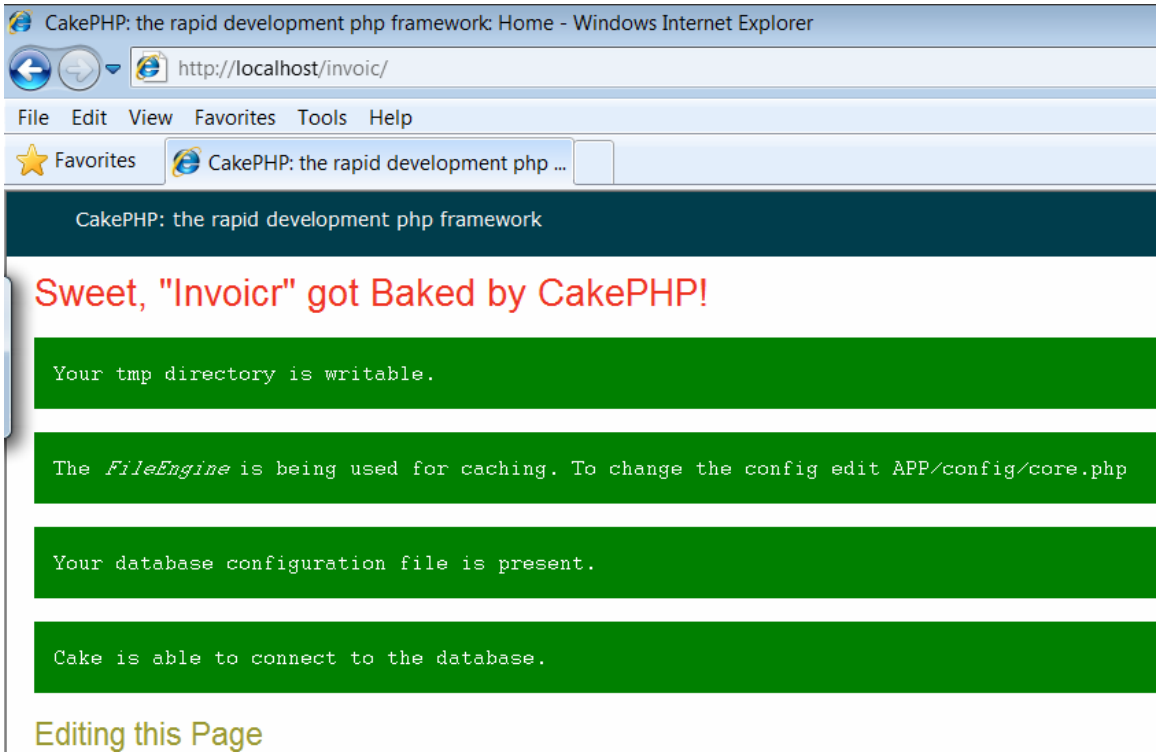


اطلاعات وارد شده سپس مورد بازبینی کاربر قرار خواهد گرفته شد.

در صورت اطمینان از صحت اطلاعات وارد شده آن را تایید می کنیم.



حال برای امتحان از برقراری ارتباط صحیح نرم افزار با پایگاه داده به http://localhost/invoice مراجعه می کنیم. در صورتی که اطلاعات درست تنظیم شده باشند باید صفحه مطابق با شکل زیر را مشاهده نماییم.



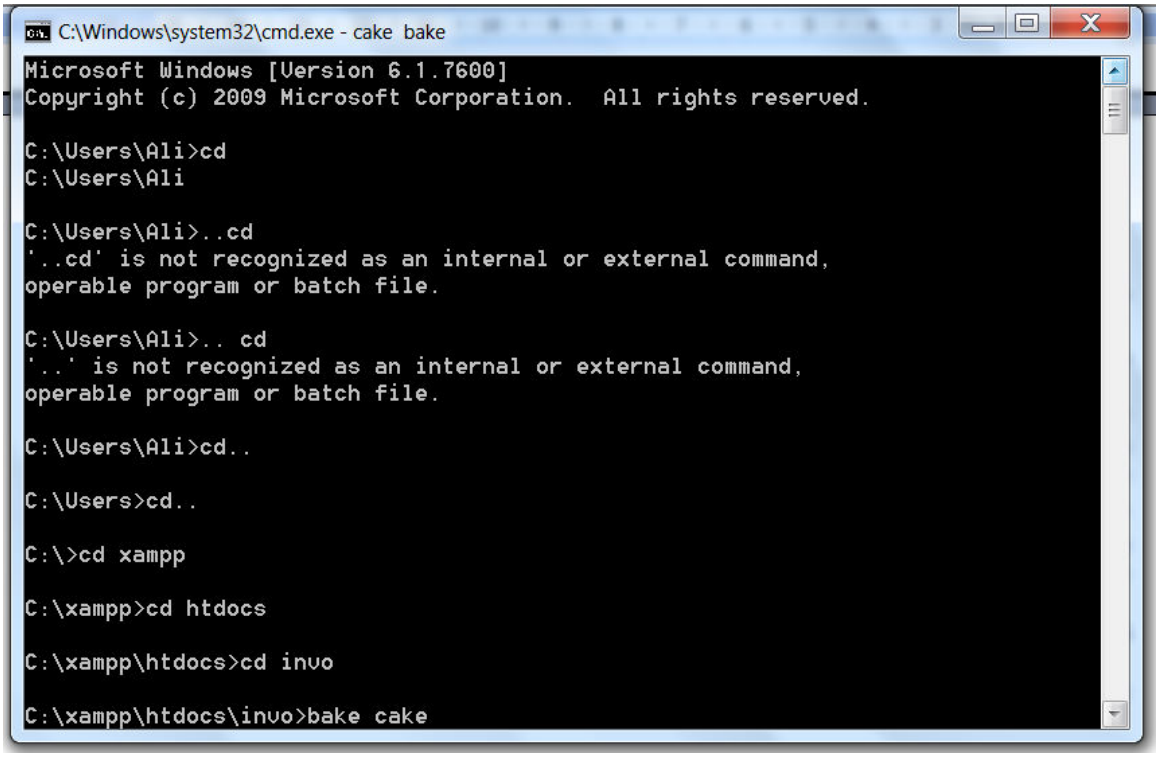
دسترسی به منوی اصلی کیک:

برنامه های تحت وب عموماً شامل بخش های تکراری شامل: درج اطلاعات، ویرایش، حذف، بخش مدیریت و ... می باشند، فریم ورک کیک با اطلاع داشتن از این ویژگی برنامه های تحت وب، از آن بهره جسته و با استفاده از ایجاد محیط ساده پرسش و پاسخ از کاربر بخش های مورد نیاز طراح را ایجاد می نماید، در واقع می توان گفت برای ایجاد یک برنامه ساده اولیه نیاز به دانستن هیچ گونه اطلاعات کدنویسی نمی باشد و فقط نیاز است تا بدانید به چه چیزی نیاز دارید و پروژه شما قرار است چگونه باشد.

همانطور که قبلاً گفته شد فریم ورک کیک بر مبنای معماری ۳ لایه MVC طراحی شده است و جهت طراحی پروژه باید با انتخاب گزینه مناسب به ایجاد ۲ بخش اصلی پروژه یعنی: Model – Control –

View پرداخته شود. ایجاد اجزاء داخلی با پاسخ مناسب به هر سوال مطرح شده توسط فریم ورک میسر خواهد گردید.

اعلان داس را باز کرده و به مسیر پوشه پروژه که قبلا در لوکال ایجاد کرده بودیم می رویم:



سپس در همان مسیر دستور اصلی کیک یعنی bake cake را اجرا می کنیم، سپس می توانیم به بخش منوی کیک دسترسی پیدا کنیم.

منوی این فریم ورک به شرح زیر می باشد:

[D]atabase Configuration – پیکربندی پایگاه داده و ویرایش تنظیمات

[M]odel - ایجاد فایل های مربوط به مدل

[V]iew - ایجاد فایل های مربوط به ظاهر سایت

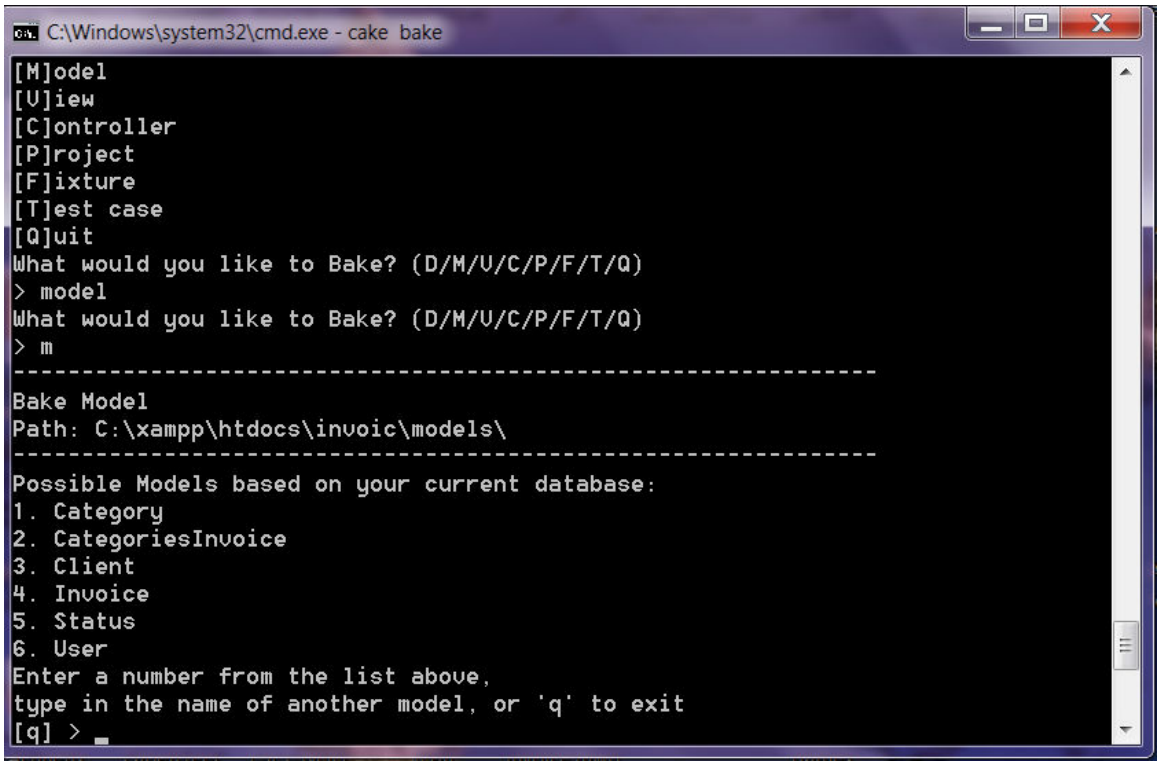
[C]ontroller – ایجاد فایل های مربوط به کنترلر

[P]roject– برای ایجاد فایل های app پروژه در صورت نیاز

[F]ixture – ایجاد نسخه جانبی برای برنامه و تست فایل ها

[T]est case - تست فایل های ایجاد شده در محیط دیباگ

[Q]uit



جهت ایجاد فایل های مربوط به model ، m را وارد می کنیم.

فریم ورک کیک با ایجاد ارتباط با جداول پایگاه داده مربوط به پروژه نام آن ها را لیست کرده و با اعلان شماره از طراح جهت ساخت و پیکربندی بصورت گزینشی از او سوال می کند.

در آغاز گزینه ۱ را جهت ایجاد فایل های مدل مربوط به موضوعات وارد می کنیم.

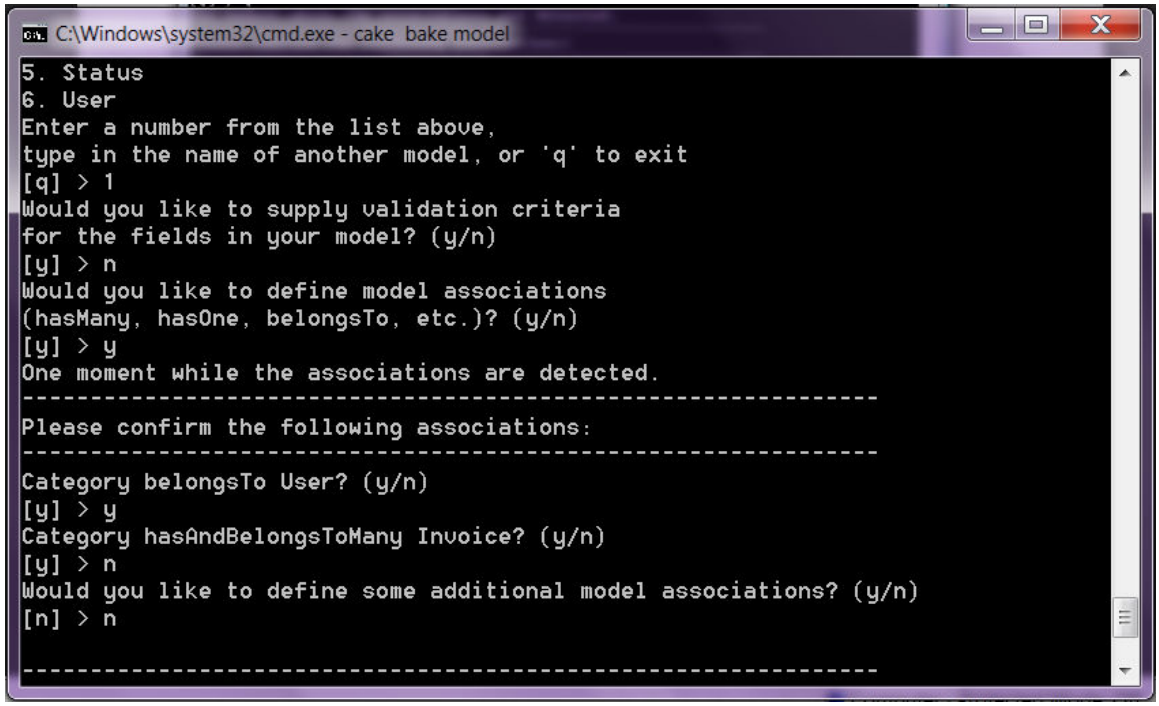
سوالاتی جهت روابط بین سیستم پرسیده می شود که باید به آن ها پاسخ صحیح ارائه داد:

سوال ۱: آیا تمایل دارید معیار ارتباط سنجی برای فیلد های موجود در مدل تهیه شود؟

پاسخ: خیر- زیرا برای موضوعات نیازی به اعتبار سنجی نیست.

سوال ۲: آیا تمایل دارید برای این مدل یک مشارکت (تکی، چند تایی، وابسته به و ...) تعریف شود؟

پاسخ: بله – به دلیل ارتباط بین موضوعات و فعالیت مشتریان ها



سوال ۳: آیا ارتباط موضوع با کاربر از نوع چند به یک است؟

پاسخ: بله - هر کاربر باید امکان ایجاد چند موضوع را داشته باشد.

سوال۴: آیا ارتباط موضوع با فاکتورها از نوع چند به چند است؟

پاسخ: خیر –هر کدام از فاکتورها باید به موضوعات تعلق داشته باشند و رابطه از نوع یک به چند فاکتور با موضوعات می باشد.

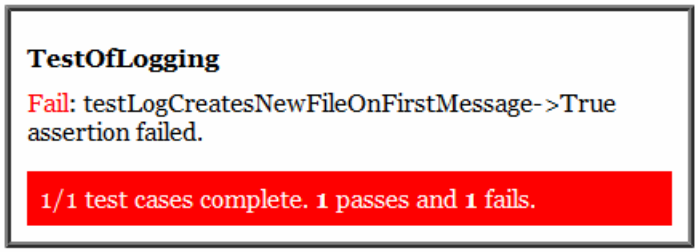
سوال۵: آیا تمایل دارید که برخی مشارکت های اضافی نیز تعریف گردد؟

پاسخ: خیر – با توجه به عملکرد سیستم نیاز نمی باشد.

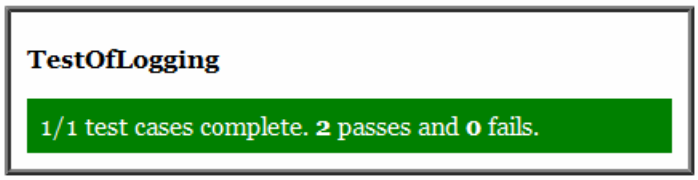
سپس جهت تایید در پاسخ به سوال: look okey? کلمه تایید یعنی y را تایپ کرده و گزارش فایل های ایجاد شده را مشاهده خواهیم کرد.

هم اکنون فایل category.php در مسیر تعیین شده ساخته شده است و سوالی مبنی بر تست این فایل پرسیده خواهد شد که به دلخواه می توان آن را تایید یا رد نمود.

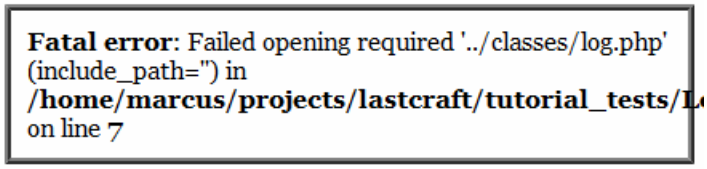
قابل ذکر است ویژگی تست فایل های php ساخته شده در ورژن 1.3 به بعد با عنوان fixture افزوده گردیده است و این امکان را می دهد تا با ایجاد شدن فایل های fixture.php_ و دانلود کردن برنامه ای آن از سایت http://simpletest.org و نصب آن بتوانیم خطاها ایجاد شده در برنامه را اشکال زدایی نماییم.



...and if it passes like this...

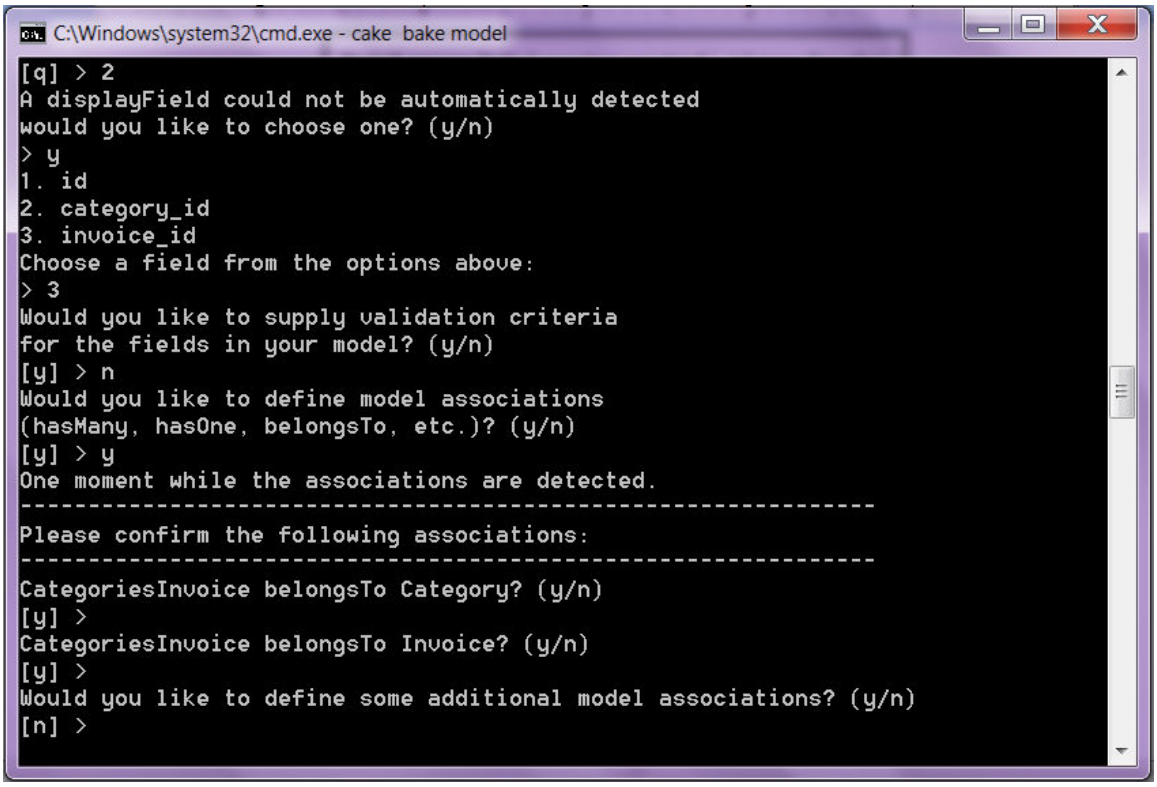


And if you get this...



مجدد با تایپ cake bake model به منوی کیک بر می گردیم و انتخاب گزینه ۲ یعنی categoryInvoices را بر می گزینیم.

توجه: این جدول، با تکرار id به invoice متصل شده است و متعلق به آن می باشد.



سوال ۱: نمایش این فیلد بصورت اتوماتیک قابل شناسایی نمی باشد آیا می خواهید آن را انتخاب کنید؟

پاسخ: خیر

سوال۲: آیا تمایل دارید معیار ارتباط سنجی برای فیلد های موجود در مدل تهیه شود؟

پاسخ: خیر- زیرا برای موضوعات نیازی به اعتبار سنجی نیست و فقط تعیین ارتباط است.

سوال ۳: آیا تمایل دارید برای این مدل یک مشارکت (تکی، چند تایی، وابسته به و ...) تعریف شود؟

پاسخ: بله

سوال ۴: آیا ارتباط موضوعات فاکتور با موضوع از نوع چند به یک است؟

پاسخ: بله - به دلیل ایجاد ارتباط دو سویه پایگاه داده جداول بین آن ها

سوال۵: آیا ارتباط موضوعات فاکتور با فاکتور از نوع چند به یک است؟

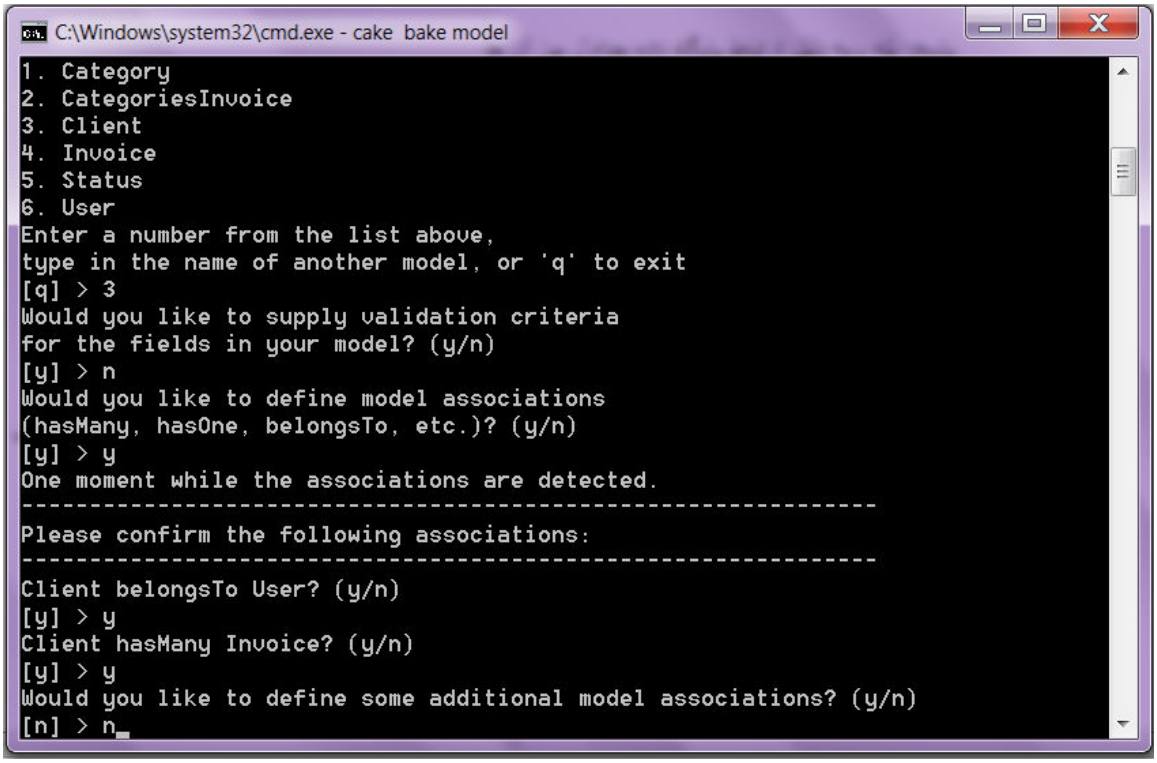
پاسخ: بله – به دلیل ارتباط پایگاه داده جداول بین آن ها و نمایش موضوعات فاکتور در فاکتور

سوال۶: آیا تمایل دارید که برخی مشارکت های اضافی نیز تعریف گردد؟

پاسخ: خیر – با توجه به عملکرد سیستم نیاز نمی باشد.

پس از تایید فایل category_invoice.php ساخته خواهد شد.

مجدد با تایپ cake bake model به منوی کیک بر می گردیم و انتخاب گزینه ۳ یعنی client را بر می گزینیم.



سوال ۱: آیا تمایل دارید معیار ارتباط سنجی برای فیلد های موجود در مدل تهیه شود؟

پاسخ: خیر- زیرا برای موضوعات نیازی به اعتبار سنجی نیست.

سوال ۲: آیا تمایل دارید برای این مدل یک مشارکت (تکی، چند تایی، وابسته به و ...) تعریف شود؟

پاسخ: بله

سوال ۳: آیا ارتباط مشتری با کاربر از نوع چند به یک است؟

پاسخ: بله - به دلیل اینکه هر کاربر عضو سیستم می تواند چند مشتری را بیا فزاید.

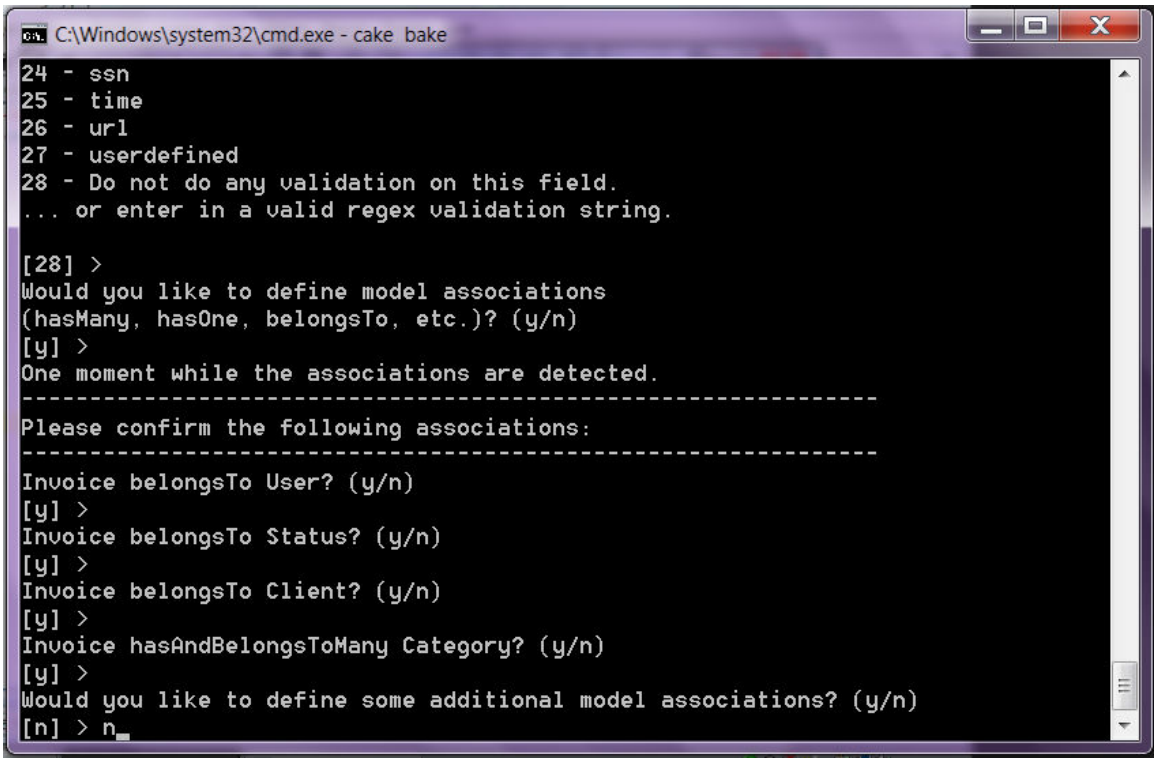
سوال ۴: آیا ارتباط مشتری با فاکتور از نوع یک به چند است؟

پاسخ: بله- هر مشتری می تواند به چندین فاکتور متعلق باشد.

سوال۵: آیا تمایل دارید که برخی مشارکت های اضافی نیز تعریف گردد؟

پاسخ: خیر – با توجه به عملکرد سیستم نیاز نمی باشد.

مجدد با تایپ cake bake model به منوی کیک بر می گردیم و انتخاب گزینه ۴ یعنی Invoice را بر می گزینیم.



سوال ۱: آیا تمایل دارید معیار ارتباط سنجی برای فیلد های موجود در مدل تهیه شود؟

پاسخ: خیر- در صورت نیاز بعداً خواهیم افزود.

سوال ۲: آیا تمایل دارید برای این مدل یک مشارکت (تکی، چند تایی، وابسته به و ...) تعریف شود؟

پاسخ: بله

سوال ۳: آیا ارتباط فاکتور با کاربر از نوع چند به یک است؟

پاسخ: بله - به دلیل اینکه هر کاربر می تواند چندین فاکتور فروش ایجاد کند.

سوال ۴: آیا ارتباط فاکتور با وضعیت پرداخت از نوع چند به یک است؟

پاسخ: بله – هر فاکتور شامل یک وضعیت پرداخت می باشد.

سوال ۵: آیا ارتباط فاکتور با مشتری از نوع چند به یک است؟

پاسخ: بله – هر فاکتور صادر شده متعلق به یک مشتری است.

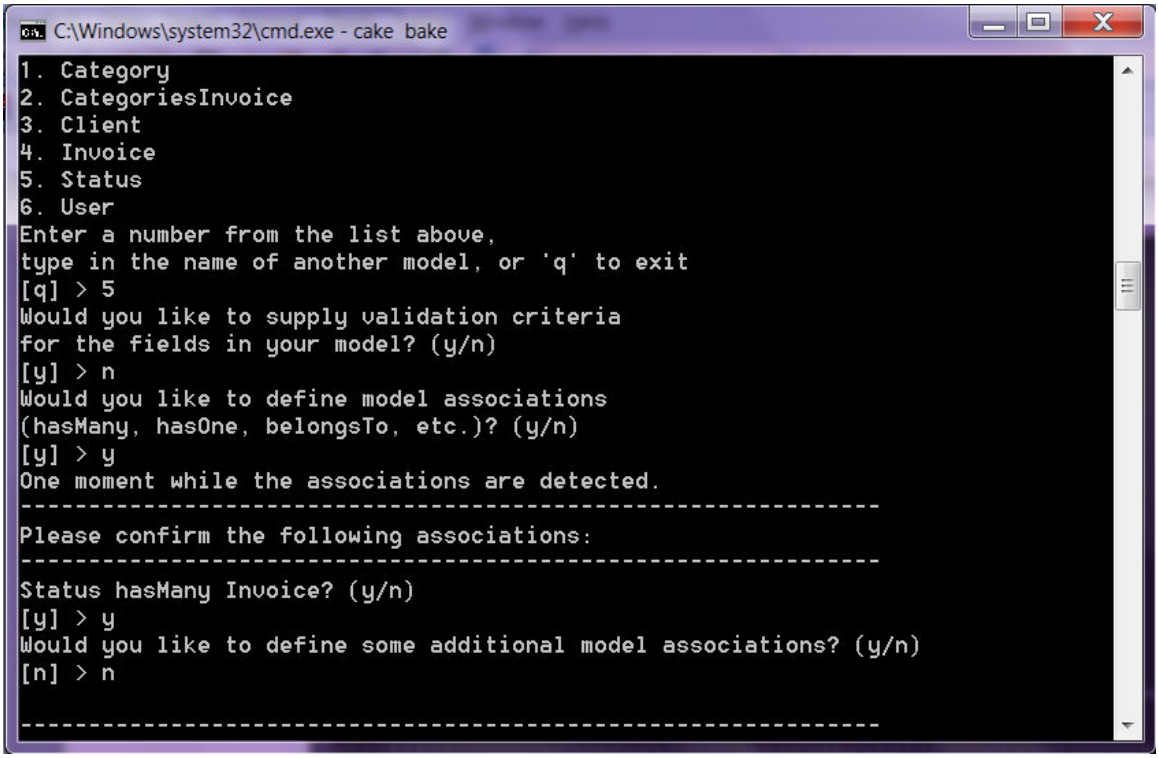
سوال ۶: آیا ارتباط فاکتور با موضوع از نوع چند به چند است؟

پاسخ: بله – از آنجا که جدولی با نام category_invoices وجود دارد ارتباط بین فاکتور با موضوع چند به چند خواهد بود.

سوال۷: آیا تمایل دارید که برخی مشارکت های اضافی نیز تعریف گردد؟

پاسخ: خیر – با توجه به عملکرد سیستم نیاز نمی باشد.

مجدد با تایپ cake bake model به منوی کیک بر می گردیم و انتخاب گزینه ۵ یعنی Status را بر می گزینیم.



سوال ۱: آیا تمایل دارید معیار ارتباط سنجی برای فیلد های موجود در مدل تهیه شود؟

پاسخ: خیر- در صورت نیاز بعداً خواهیم افزود.

سوال ۲: آیا تمایل دارید برای این مدل یک مشارکت (تکی، چند تایی، وابسته به و ...) تعریف شود؟

پاسخ: بله

سوال ۳: آیا وضعیت فروش با فاکتور از نوع یک به چند است؟

پاسخ: بله – هر فاکتور شامل یک وضعیت فروش می شود.

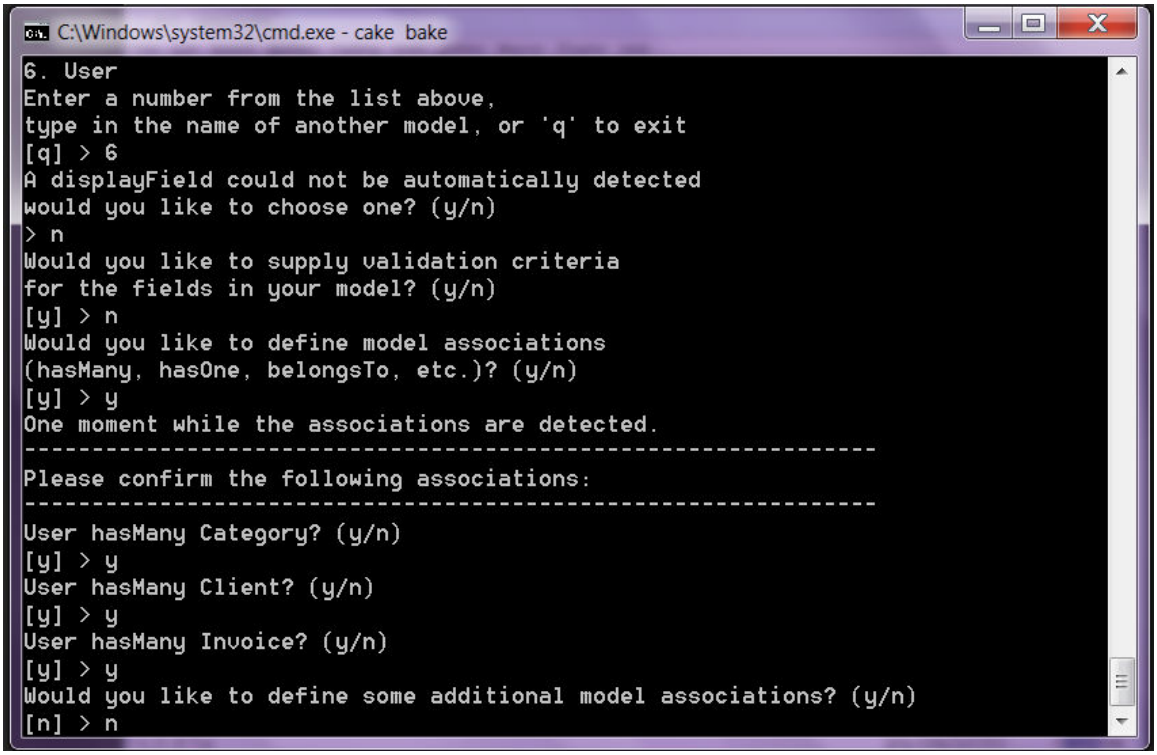
سوال ۴: آیا وضعیت فروش با فاکتور از نوع یک به یک است؟

پاسخ: بله – وضعیت هر سفارش مربوط به یک فاکتور جداگانه است.

سوال۵: آیا تمایل دارید که برخی مشارکت های اضافی نیز تعریف گردد؟

پاسخ: خیر – با توجه به عملکرد سیستم نیاز نمی باشد.

مجدد با تایپ cake bake model به منوی کیک بر می گردیم و انتخاب گزینه ۶ یعنی users را بر می گزینیم.



سوال ۱: آیا تمایل دارید معیار ارتباط سنجی برای فیلد های موجود در مدل تهیه شود؟

پاسخ: خیر- در صورت نیاز بعداً خواهیم افزود.

سوال۲: نمایش این فیلد بصورت اتوماتیک قابل شناسایی نمی باشد آیا می خواهید آن را انتخاب کنید؟

پاسخ: خیر

سوال ۳: آیا تمایل دارید برای این مدل یک مشارکت (تکی، چند تایی، وابسته به و ...) تعریف شود؟

پاسخ: بله

سوال ۴: آیا ارتباط کاربر با موضوع از نوع یک به چند است؟

پاسخ: بله – هر کاربر می تواند چند موضوع ایجاد کند.

سوال ۵: آیا ارتباط کاربر با مشتری از نوع یک به چند است؟

پاسخ: بله – هر کاربر می تواند چند مشتری ایجاد کند.

سوال ۶: آیا ارتباط کاربر با فاکتور از نوع یک به چند است؟

پاسخ: بله – هر کاربر می تواند چند فاکتور فروش ایجاد کند.

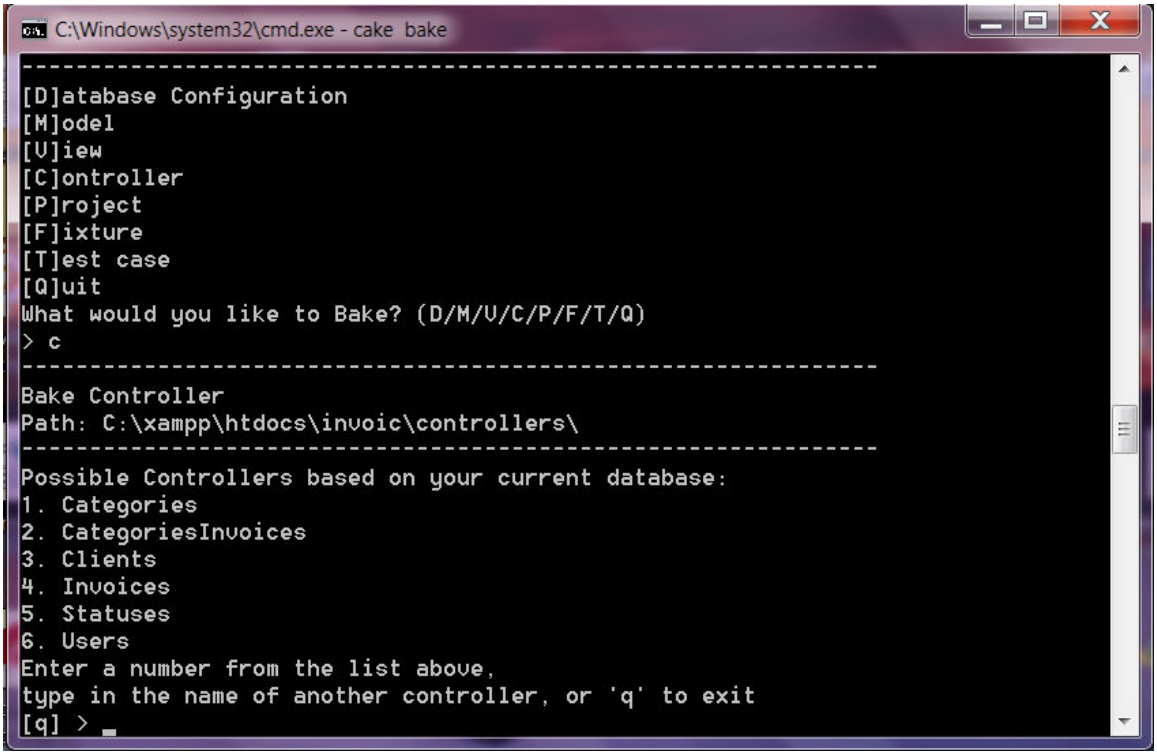
سوال۷: آیا تمایل دارید که برخی مشارکت های اضافی نیز تعریف گردد؟

پاسخ: خیر – با توجه به عملکرد سیستم نیاز نمی باشد.

ایجاد فایل های مربوط به کنترلر - Controller پروژه:

اکنون وقت آن است تا فایل های کنترلر را نیز ایجاد نماییم.

دستور cake bake controller را اجرا نموده و گزینه اول یعنی Categories را انتخاب می کنیم.



سوال ۱: آیا تمایل به ساخت یک کنترلر تعاملی دارید؟

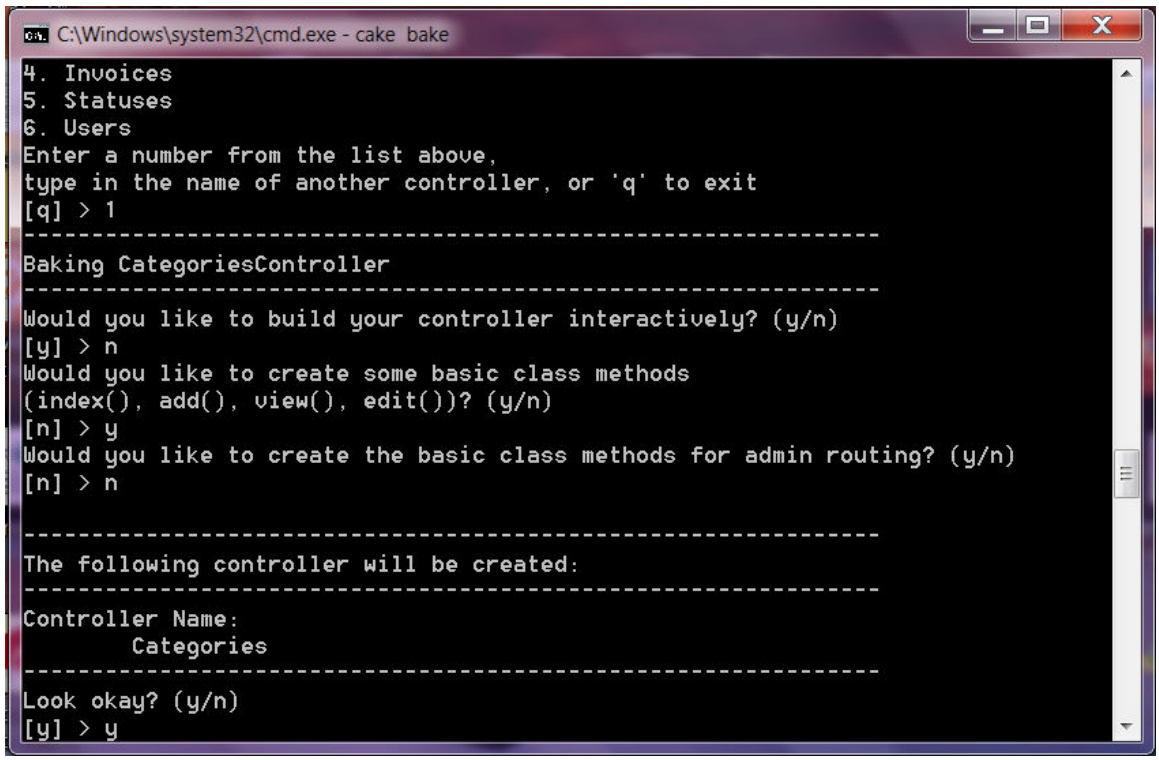
پاسخ: خیر

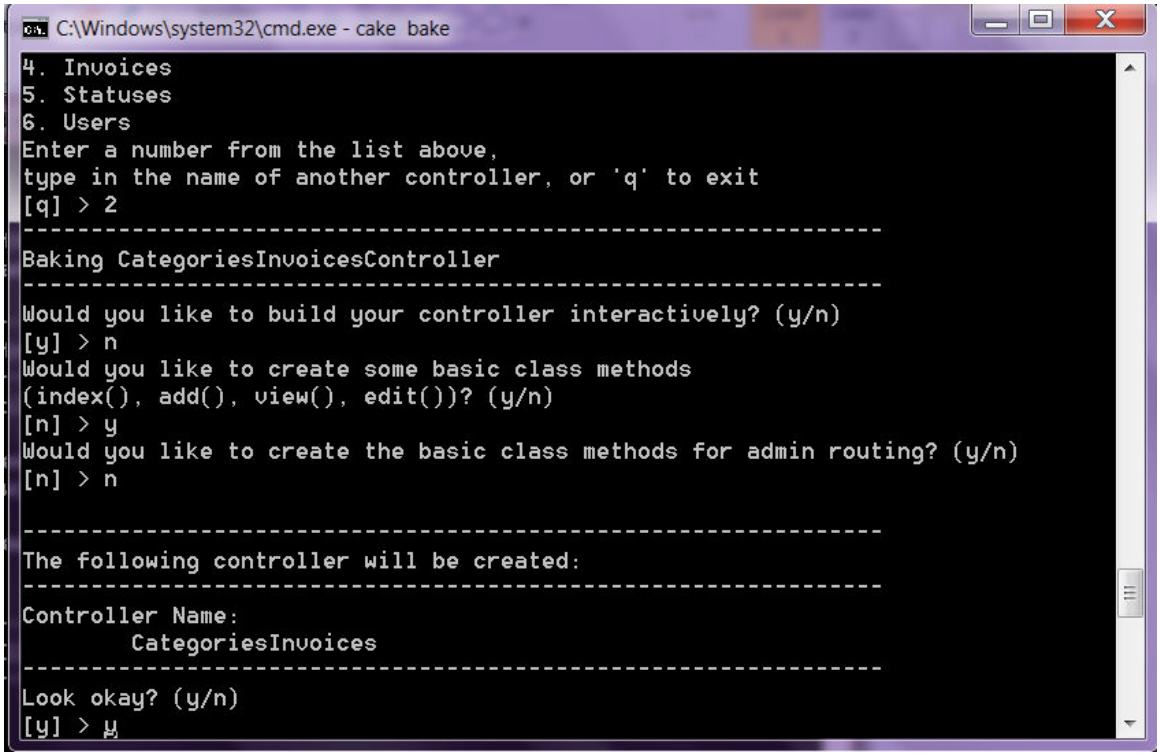
سوال۲: آیا تمایل به ایجاد کلاس هایی همچون index() , Add(), Edit() , View می باشید؟

بله – موضوعات باید قابلیت ایجاد و ویرایش و مشاهده داشته باشند.

سوال۳: آیا تمایل به ایجاد کلاس هایی برای دسترسی مدیر به این بخش هستید؟

پاسخ: خیر – با توجه به عملکرد سیستم هر کاربر عضو شده در سایت در واقع مدیر سیستم نیز می باشد.



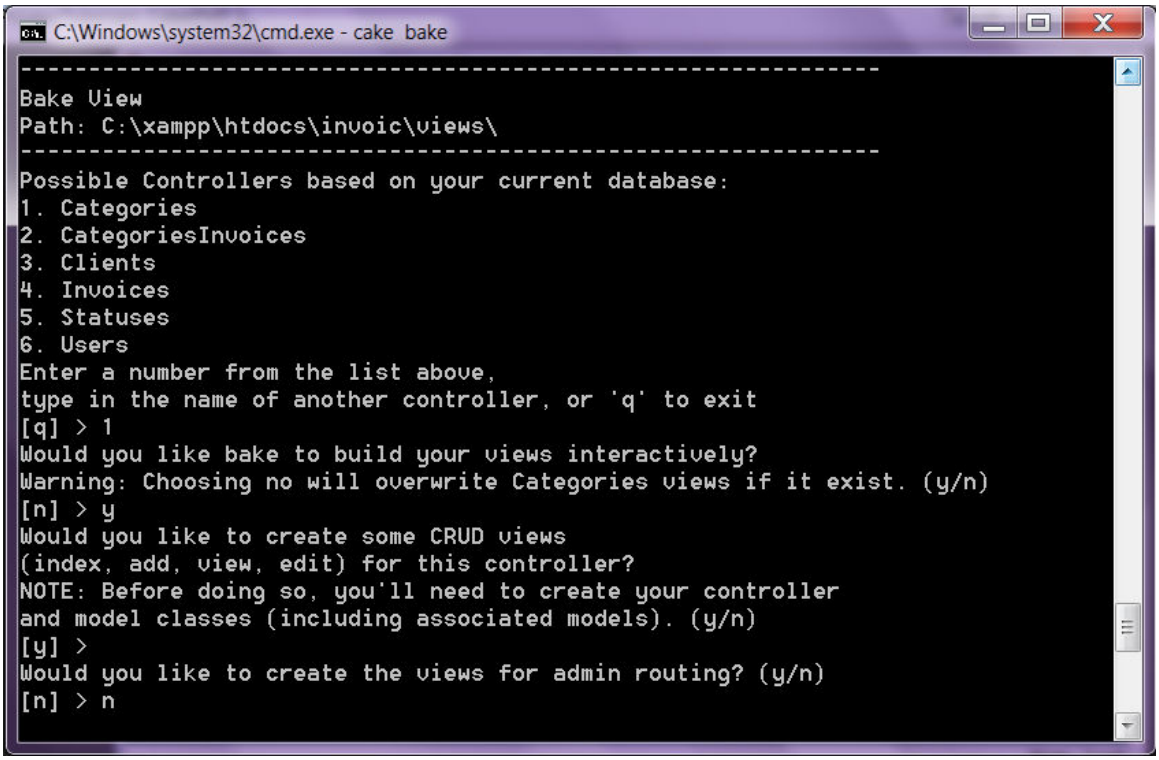


مجدد به بخش کنترلر منو باز می گردیم و برای تمام کنترلرها باقی مانده (Client,Invoices,Statuses,Users) پاسخ هایی مشابه Categories را به فریم ورک ارائه می کنیم.

حال برای اطمینان از ساخت فایل های کنترلر با مراجعه به شاخه کنترلر در پوشه پروژه می توانیم ۶ فایل ایجاد شده را مشاهده نماییم.

ایجاد فایل های مربوط به نمایش- View پروژه:

دستور cake bake view را اجرا نموده و گزینه اول یعنی Categories را انتخاب می کنیم.



سوال۱: آیا تمایل دارید مشاهدات بصورت تعاملی ایجاد شود؟

پاسخ: خیر

سوال۲: آیا تمایل به ایجاد بخش های نمایه، اضافه کردن، مشاهده، ویرایش دارید؟

پاسخ: بله

سوال۳: آیا تمایل یه ایجاد بخش مدیریتی دارید؟

پاسخ: خیر – با توجه به اینکه کنترلری برای آن تعریف نکردیم.

با توجه به اینکه گزینه ۲ categoryInvoices شامل موضوعات می شود و در بخش فروش نشان داده می شود از ایجاد بخش نمایش برای آن صرف نظر کرده و قسمت های بعدی را ایجاد می کنیم.

برای ایجاد بخش های باقی مانده یعنی Client , invoice , Status و user پاسخ هایی همانند پاسخ های گزینه ۱ یعنی Categories را ارائه می کنیم.

هم اکنون پروژه آماده شده است و با مراجعه به پوشه view و زیر شاخه های مربوطه می توانیم فایل های با پسوند ctp را مشاهده نماییم، این فایل ها در واقع قالب نمایشی سایت را به همراه کدها نمایش می دهند، بعد از برنامه نویسی سفارشی جهت کاهش حجم صفحات در آینده این فایل ها را به پوشه element منتقل خواهیم کرد.

حال با اجرای مفسر پی اچ پی همانند xampp و مراجعه به آدرس دایرکتوری پروژه بر روی سرور لوکال می توانیم نتیجه کار خود را تست کنیم.

آدرس پروژه:

http://localhost/invoice/invoices

با توجه به اینکه فایل صفحه اصلی هنوز ایجاد نشده است نمی توانیم فعلاً به ریشه اصلی سایت دسترسی داشته باشیم و باید آن را در آینده ایجاد کنیم.

ویرایش فایل های ایجاد شده و سفارشی کردن کدها:

فریم ورک کیک با ایجاد سوالاتی بسیار ساده بنده اصلی پروژه ما را طراحی نمود حال برای ایجاد قالب سفارشی و برنامه نویسی به زبان php به منظور سفارشی سازی کدها باید با استفاده از یک ویراشگر کدهای php همانند jedit و یا phpDesigner استفاده کنیم، ویرایشگرهای php با شناخت و خطا زدایی کدها ما را در امر برنامه نویسی یاری می کنند.

جهت برنامه نویسی کدهای این پروژه از ویرایشگر phpDesigner نسخه ۷ استفاده شده است.

۱) ایجاد اعتبار سنجی در اطلاعات ارسالی به فرم ها:

جهت جلوگیری از ورود اطلاعات اشتباه توسط کاربران، بهتر است در فرم های ثبت نام و .... از دستورات مرتبط به اعتبارسنجی استفاده نموده و کاربران را ملزم به رعایت قالب و مقدار صحیح از اطلاعات نماییم جهت افزودن اعتبارسنجی به فرم های این پروژه باید با مراجعه به پوشه پروژه و زیرشاخه Model و پیدا کردن فایل مورد نظر مربوط به فرم آن را توسط ویرایشگر باز و ویرایش نمود، در فریم ورک کیک افزودن اعتبارسنج ها به سادگی امکان پذیر می باشد، به عنوان مثال در صورتی که فیلدی با نام ایمیل داشته باشیم با افزودن یک آرایه تعریف شده در این فریم ورک که با rule مشخص می گردد می توانیم اعتبارسنجی مرتبط به ورود اطلاعات از نوع ایمیل را به فیلد گوش زد کنیم.

آرایه های اعتبار سنجی مدل در فریم ورک کیک:

در فریم ورک کیک به جهت تسریع در کار به جهت اعتبارسنجی از پیش آرایه هایی تعریف شده است که به سادگی با تعریف آن آرایه و انتصاب به نام فیلد فرم های پروژه به راحتی می توان این کار را انجام داد، در زیر لیستی از آرایه های اعتبارسنجی فریم ورک کیک آمده است همچنین با مراجعه به پایگاه اصلی کیک به آدرس می توانید دستورات بیشتری را نیز بیابید.

پذیرش فقط حروف الفبا اعداد:

```
'alphaNumeric' => array(
    'rule' => 'alphaNumeric',
    'required' => true,
    'message' => 'فقط حروف الفبا و اعداد'
```

پذیرش فقط حروف الفبا اعداد:

```
'alphaNumeric' => array(
    'rule' => 'alphaNumeric',
    'required' => true,
    'message' => 'فقط حروف الفبا و اعداد'
```

پذیرش بازه ای از اعداد:

```
'between' => array(
    'rule' => array('between', 5, 15),
    'message' => 'بین ۵ تا ۱۵ کراکتر باید باشد'
```

پذیرش حداقل طول تعریف شده:

```
'password' => array(
    'rule' => array('minLength', '8'),
    'message' => 'حداقل ۸ کراکتر باید وارد کنید'
```

پذیرش تاریخ:

```
'email' => 'email',
'born' => array(
    'rule' => 'date',
    'message' => 'فرمت تاریخ را صحیح وارد کنید',
    'allowEmpty' => true
```

پذیرش پست الکترونیک استاندارد:

```
'email' => 'email',
'born' => array(
    'rule'=> 'email',
    'message'=> 'پست الکترونیک را صحیح وارد کنید'
    'allowEmpty' => true
```

پذیرش آدرس وب :

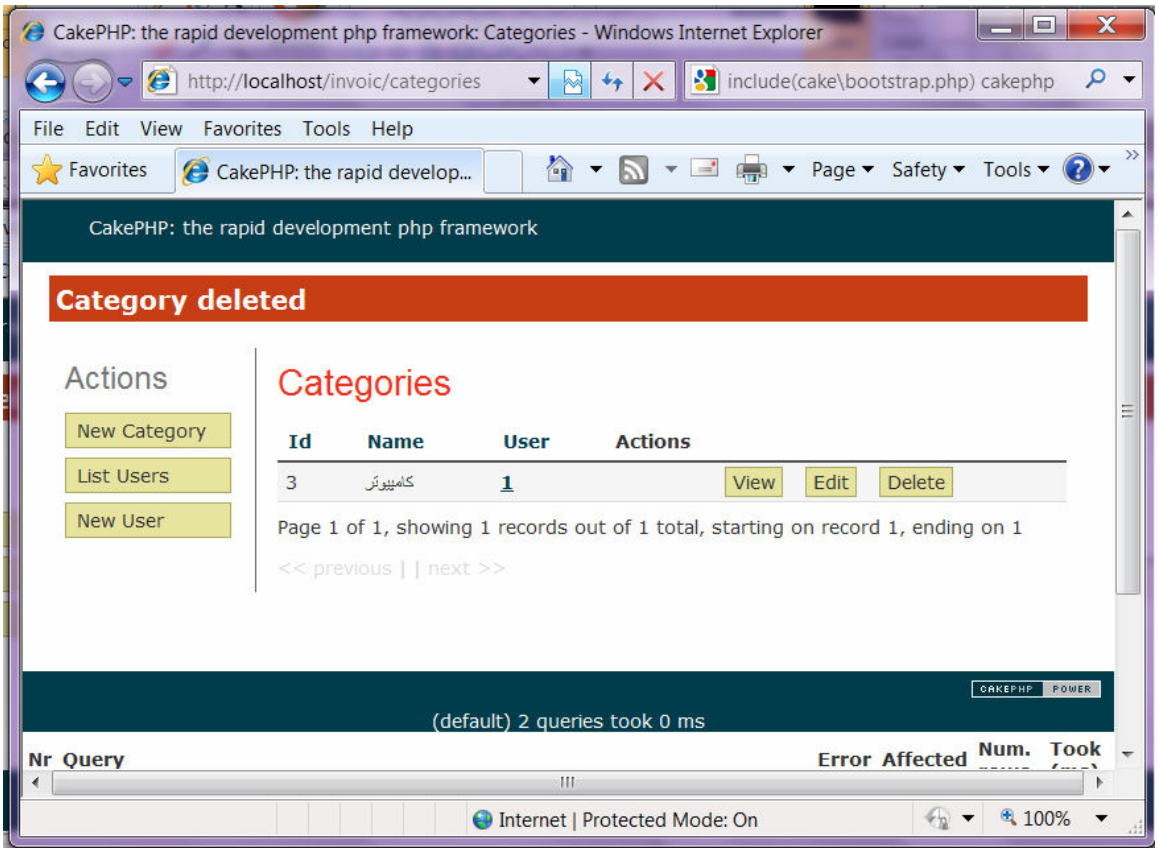
```
'web' => 'web',
'born' => array(
    'rule'=> 'url',
    'message'=> 'آدرس اینترنتی را صحیح وارد کنید'
    'allowEmpty' => true
```

اعتبارسنجی فایل Client:

به مسیر invoice » model » Client.php رفته و با ویرایشگر آن را باز می کنیم و طبق تصویر اطلاعات اعتبارسنجی مناسب برای هر فیلد را وارد می کنیم.

```php
<?php
class Invoice extends AppModel {
    var $name = 'Invoice';
    //The Associations below have been created with all possible key
var $validate=array(

'invoiceId'=> array(
'minLength'=> array(
'rule'=> array('minLength',1),
'allowEmpty'=> false,
'message'=> 'شماره فاکتور را وارد کنید'
)
),
'taxRate'=> array(
'decimal'=> array(
'rule'=> array('decimal',2),
'allowEmpty'=> false,
'message'=> 'نرخ مالیات را وارد کنید / 2 رقم اعشار'
)
),
'total'=> array(
'moneyFormat'=> array(
'rule'=> array('decimal',2),
'allowEmpty'=> false,
'message'=> 'مجموع هزینه ها را وارد کنید / 2 رقم اعشار'
),
'greaterThan'=> array(
'rule' => array('comparison', '>', 0.01),
'allowEmpty'=> true,
'message'=> 'باید بزرگتر از 0.01 باشد'
```

با توجه به یکسان بودن عملیات، برای سایر فایل های مرتبط یعنی users.php و invoices.php به شرح بالا عمل می نماییم، البته باید در نظر داشته باشیم که در صورتی که قرار است از تاریخ هجری شمسی استفاده نماییم، برای تاریخ اعتبارسنجی را فعال نکنیم.



همانطور که در تصویر مشاهده می کنید، فریم ورک کیک با در در نظر گرفتن قالبی پیش فرض اطلاعات درخواستی را نمایش می دهد، از آنجا که بصورت پیش فرض در قسمت پایین پروژه اطلاعات مربوط به پرس و جوهای درخواستی از پایگاه داده نمایش داده می شوند که با توجه به عدم نیاز ما برای نمایش در وب باید کدهای اضافی و برخی قسمت ها حذف گردند.

همچنین در حال حاظر بخش های ایجاد شده مربوط به قالب سایت در بخش view قرار گرفته اند که برای کاهش صفحات و کدهای اضافی و تکراری بهتر است، کدهای کوتاه و اصلاح شده را در پوشه element قرار داده و به فایل اصلی متصل نماییم.

طراحی و سفارشی سازی قالب:

جهت حذف اطلاعات اضافی به مسیر view » layouts می رویم و فایل default.ctp را توسط ویرایشگر باز می کنیم و دستورات زیر را حذف می کنیم:

اطلاعات مربوط به لینک سایت کیک ....<?php echo $this->Html->link(

حذف نمایش پرس و جوها <?php echo $this->element('sql_dump'); ?>

حذف عنوان تبلیغاتی فریم ورک و ...

<h1><?php echo $this->Html->link(__('CakePHP: the rapid development php framework', true), 'http://cakephp.org'); ?></h1>

حذف عنوان و جایگزینی آن با مقدار دلخواه:

<?php __('CakePHP: the rapid development php framework:'); ?>

حال با حذف این قسمت ها ظاهر سایت بهتر شده است، برای تغییرات فایل های تصاویر و قالب بندی CSS به پوشه webroot »CSS باید مراجعه کرد.

فریم ورک کیک بصورت هوشمندانه با استفاده از نام جداول پایگاه داده اسامی منوها و ....را انتخاب نموده است ولی با توجه به اینکه سیستم قرار است فارسی باشد فایل CSS را باز کرده و در هر کجا دستورهای text-align:left و float: left و margin-left بود جهت نمایش صحیح راست چین شدن آن ها را به right تغییر می دهیم، همچنین کلیه فونت ها را به Tahoma که فونت رایج وب فارسی است تغییر می دهیم.

با توجه به اینکه کیک از یک قالب ساده بصورت پیش فرض برای پروژه های ایجاد شده استفاده می کند و طبیعتاً در صورت استفاده از متون فارسی مشکل عدم نمایش صحیح و

چپ چین شدن می باشد، بایستی با اطلاع اولیه از نام های کلاس های قالب بندی کیک Css و استفاده از آن یک قالب طراحی نماییم، برای این کار باید به شاخه webroot موجود در پروژه رفته و فایل cake.generic.css را در مسیر css پیدا کرده و باید طراحی شده جایگزین نماییم، پوشه تصاویر نیز در همان مسیر بوده و برای مسیر دهی در فایل CSS باید بصورت ../img/name.jp آدرس دهی نماییم.

حال به شاخه invoice و ... مراجعه کرده و جهت حذف منوی کاربری و سفارشی کردن آن در کلیه فایل ها در بخش پایین کدها دستورات مربوط به نمایش منو را حذف می کنیم:

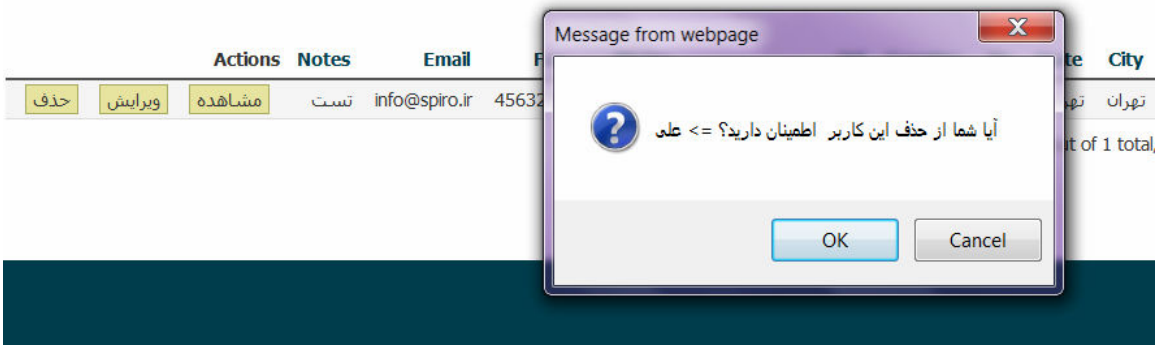
```
<div class="actions">
    <h3><?php __('Actions'); ?></h3>
    <ul>

        <li><?php echo $this->Html->link(__('List Invoices', true), array('action' => 'index'));?></li>
        <li><?php echo $this->Html->link(__('List Users', true), array('controller' => 'users', 'action' => 'index')); ?> </li>
        <li><?php echo $this->Html->link(__('New User', true), array('controller' => 'users', 'action' => 'add')); ?> </li>
        <li><?php echo $this->Html->link(__('List Statuses', true), array('controller' => 'statuses', 'action' => 'index')); ?> </li>
        <li><?php echo $this->Html->link(__('New Status', true), array('controller' => 'statuses', 'action' => 'add')); ?> </li>
        <li><?php echo $this->Html->link(__('List Clients', true), array('controller' => 'clients', 'action' => 'index')); ?> </li>
        <li><?php echo $this->Html->link(__('New Client', true), array('controller' => 'clients', 'action' => 'add')); ?> </li>
        <li><?php echo $this->Html->link(__('List Categories', true), array('controller' => 'categories', 'action' => 'index')); ?> </li>
        <li><?php echo $this->Html->link(__('New Category', true), array('controller' => 'categories', 'action' => 'add')); ?> </li>
    </ul>
</div>
```

و در فایل default.ctp کد سراسری منو را قرار می دهیم:

```
    <div id="container">
        <div id="header">
            <h1>سیستم مدیریت فروش</h1>
                <!-- start nav -->

        </div>

        <div id="content">
                    <?php echo $this->Session->flash(); ?>
      <div class="actions">
    <ul>
        <li> <?php echo $html->link('خانه', '/');?></li>
      <li> <?php echo $html->link('صفحه رومیزی', '/dashboard');?></li>
        <li>  <?php echo $html->link('فروشنده ها', '/invoices');?></li>
      <li> <?php echo $html->link('مشتری ها', '/Clients');?></li>
      <li> <?php echo $html->link('موضوعات', '/categories');?></li>
 <li><?php echo $html->link('خروج', 'users/Logout');?></li>
</ul>
    </div>


            <?php echo $content_for_layout; ?>
```

جهت تغییر پیغام های هشدار حذف به مسیر Client » index.ctp مراجعه کرده و سپس کد مربوطه را ویراش می کنیم، باید دقت داشته باشیم که بصورت پیشفرض آی دی مشتری نمایش داده می شود که با تغییر آن به نام وضعیت مطلوب تری حاصل می شود:

```
 ?>
 ?>
ull, sprintf(__('آیا شما از حذف این کاربر  اطمینان دارید؟ => %s', true), $client['Client']['name'])); ?>
```



ایجاد تاریخ شمسی:

عموماً در زبان برنامه نویسی php با include کردن فایلی با نام jdate و فراخوانی آن در تابع زمان می توان تاریخ را بصورت هجری شمسی نشان داد، ولی با توجه به اینکه فریم ورک کیک ملزم به رعایت اصول سلسله مراتبی خاص خودش است، باید جهت شمسی نمودن تاریخ به شیوه زیر عمل نمود:

۱) فایل persiandate.php را دریافت و در فولدر helper در مسیر view کپی کنید.

۲) به فایل کنترلری که می خواهید view آن نمایش پیدا کند مراجعه و دستور زیر را جهت include شدن دستورات اضافه نمایید:

```
var $helpers = array( 'Persiandate');
```

۳) به فایل view مربوطه مراجعه کنید و دستورات فوق را جایگزین نمایید:

۴) بصورت مثال:

```
<?php echo $invoice['Invoice']['created']; ?>
```

```
<?php echo $persiandate-
>pdate('Y/m/d', strtotime($invoice['Invoice']['created'])); ?>
```

۵) نکته: تابع strtotime فرمت تاریخ موجود را به شکل یک رشته قابل ترجمه (timestamps)تبدیل می کند.

```
echo $this->Form->input('date');
```

۶) کد زیر را جایگزین آن می کنیم:

```
echo $this->Form->input('date', array(
    'class' => 'date',
    'dateFormat' => 'YMD',
    'maxYear' => $this->Persiandate->pdate('Y')+10,
    'minYear' => $this->Persiandate->pdate('Y')-10,
    'monthNames' => $this->Persiandate->pdate('m'),
     'type' => 'date',
    ));
```

۷) نکته: جهت نمایش سال کمینه و بیشینه می توانید فاصله را به دلخواه تغییر دهید.

همچنین جهت نمایش فارسی ماه و روز و ساعت می توانید با مراجعه به فایل persiandateمعادل timestamps آن را پیدا کنید.

۸) برنامه نویسی پیشرفته برای دسترسی به اطلاعات:

۹) جهت سهولت در دسترسی به اطلاعات و نمایش آن ها لازم است تا تغییراتی را برای این کار در فایل ها داشته باشیم:

۱۰)در فایل index.php مربوط به موضوعات بایستی کدهای اضافی شامل: نمایش شناسه کاربرد و ... حذف گردد و فقط موضوع نشان داده شود:

```
<?php echo $this->Paginator->sort('user_id');?>
```

```
<?php echo $category['Category']['id']; ?>
```

همچنین جهت پیوند شدن موضوع به لینک ویرایش باید کد :

```
<?php echo $category['Category']['id']; ?>
```

با کد:

```
<?php echo $this->Html-
>link($category['Category']['name'], array('action' => 'edit', $category['Catego
ry']['id'])); ?>
```

جایگزین گردد، حال برای هر کدام از موضوعات یک لینک قابل ویرایش داریم.

ایجاد بخشی به نام "صفحه رومیزی":

از آنجا که قرار است سیستم مورد نظر ما دارای بخشی برای نمایش وضعیت درخواست های ثبت شده باشد باید برای این منظور مسیریاب مناسب برای ایجاد این صفحه ایجاد نماییم.

به مسیر config » routes.php مراجعه کرده و در پایین آرایه تعریف شده مربوط به صفحه اصلی کد زیر را می نویسیم:

```
Router::connect('/dashboard', array('controller' => 'invoices', 'action' => 'dashboard'));
```

حال باید کنترل این صفحه را ایجاد کنیم:

به مسیر controller رفته و فایل invoices_controller.php را باز می کنیم.

نیاز است تا تابع فراخوان قالب سایت در این صفحه نوشته شود:

```
function dashboard()
{

}
```

حالا کافیست فقط به بخش view » invoices مراجعه کنیم و صفحه ای خالی با نام dashboard.ctp ایجاد نمایم. به کنترلر invoice مراجعه کرده و یک تابع تعریف می کنیم، از آنجا که قرار است سیستم

ما قرار است وضعیت پرداخت ها را بصورت (معلق، درحال انجام،انجام شده و گذشته) باشد کدهای زیر را برای تعریف تابع وضعیت سفارش ها اضافه می کنیم:

```
  $draftInvoices = $this->Invoice-
>find('all', array('conditions' => "Invoice.status_id = '1'"));
      $openInvoices = $this->Invoice-
>find('all', array('conditions' => "Invoice.status_id = '2'"));
      $closedInvoices = $this->Invoice-
>find('all', array('conditions' => "Invoice.status_id = '3'"));
      $pastdueInvoices = $this->Invoice-
>find('all', array('conditions' => "Invoice.status_id = '4'"));

      $this-
>set(compact('draftInvoices', 'openInvoices', 'closedInvoices', 'pastdueInvoices' ));
```

حال با مراجعه به فایل مربوط به view کدهای زیر را اضافه می کنیم:

```
<h1>وضعیت پرداخت ها</h1>

  <h2 class="drafts">پرداخت های در حال بررسی</h2>

    <?php foreach ($draftInvoices as $invoice): ?>

      <?php echo $this->element('invoice', array('invoice' => $invoice)); ?>

    <?php endforeach; ?>

  <h2 class="pastdue">پرداخت های تاریخ گذشته</h2>
```

```
<?php foreach ($pastdueInvoices as $invoice): ?>
  <?php echo $this->element('invoice', array('invoice' => $invoice)); ?>
<?php endforeach; ?>
<h2 class="open">پرداخت های در حال انجام</h2>
<?php foreach ($openInvoices as $invoice): ?>
  <?php echo $this->element('invoice', array('invoice' => $invoice)); ?>
<?php endforeach; ?>
<h2 class="closed">پرداخت های انجام شده</h2>
<?php foreach ($closedInvoices as $invoice): ?>
  <?php echo $this->element('invoice', array('invoice' => $invoice)); ?>
<?php endforeach; ?>
```

نظم و دقت بیشتر با بکارگیری عناصر (elements):

از آنجا که برخی از دستورات به دلیل شباهت عملکرد یکسان می باشد، می توانیم کدها را با انتقال به پوشه خالی بنام element که مخصوص این کار در نظر گرفته شده است انتقال داده و با استفاده از دستور element مشابه دستور include در php بخش های تکراری را یکبار در فایل ضمیمه بیاوریم،

این کار باعث نظم بیشتر در کدنویسی خواهد شد و در صورت اعمال تغییرات کافیست فقط یک بار بر روی فایل اصلی اعمال گردد تا در تمامی صفحات نمایش پیدا کند.

برای مثال در فایل index موجود در invoices کدهای مربوط به صفحه بندی یعنی :

```
<p>
<?php
echo $this->Paginator->counter(array(
'format' => __('صفحه %page% از %pages% نمایش %current% رکورد از میان %count%, تا
: %end%', true)عداد نتایج
));
?>  </p>

<div class="paging">
    <?php echo $this->Paginator-
>prev('<< ' . __('previous', true), array(), null, array('class'=>'disabled'));?>
 |  <?php echo $this->Paginator->numbers();?>
|
    <?php echo $this->Paginator-
>next(__('next', true) . ' >>', array(), null, array('class' => 'disabled'));?>
</div>
</div></div>
```

این کدها را به صفحه ای جدید با نام paging.ctp در پوشه element منتقل کرده و آن را لینک به فایل اصلی می کنیم:

```
<?php echo $this->element('paging')?>
```

حالا با مراجعه به تمامی صفحاتی که شامل این کد بودند می توانیم دستور بالا را جایگزین آن کنیم.

محدودیت کراکترهای ورودی برای فرم ها:

از آنجا که ممکن است کاربران سهواً و یا عمداً اطلاعات را بصورت طولانی و با کراکترهای بالا وارد کنند، بهتر است از این جهت آرایه ای برای تعداد کراکتر های ورودی اعمال نماییم:

```
echo $this->Form->input('address' , array('size'=>30));
```

در این دستور برای فیلد ورودی آدرس حداکثر ۳۰ کراکتر ورودی پذیرفته خواهد گردید.

برچسب زدن به فیلد فرم ها:

از آنجا که فریم ورک کیک بصورت اتوماتیک و با استفاده از نام جداول پایگاه داده متنی را به عنوان نام در کنار فیلدهای فرم قرار می دهد باید برای تغییر آن از دستور برچسب استفاده نماییم.

مثال:

```
echo $this->Form->input('Category', array('label'=>'موضوع: '));
```

تغییر پیغام های هشدار:

در صورت فارسی بودن سیستم باید پیغام ها نیز متناسب با وظیفه ترجمه و تغییر داده شوند، برای این منظور باید به پوشه کنترلر مراجعه و فایل مربوطه را ویرایش کنیم.

مثال: تغییر پیغام های موفقیت و هشدار برای بخش موضوعات:

مراجعه به مسیر controllers » ویرایش فایل categories_controller.php

```
        if (!$id) {
            $this->Session->setFlash(__('شناسه نامعتبر برای موضوع', true));
            $this->redirect(array('action'=>'index'));
        }
        if ($this->Category->delete($id)) {
            $this->Session->setFlash(__('موضوع حذف گردید.', true));
            $this->redirect(array('action'=>'index'));
        }
        $this->Session->setFlash(__('موضوع حذف نمی شود!', true));
        $this->redirect(array('action' => 'index'));
    }
}
```

نکته خیلی مهم: در صورت فارسی بودن پیغام ها باید حتماً فایل با نو تپد و یا ویرایشگر باز شده و بصورت utf-8 ذخیره گردد و سپس حالت BOM آن برداشته شود، در غیر اینصورت متون فارسی بصورت ؟؟؟ ظاهر خواهند شد و ارسال و نمایش اطلاعات با خطا مواجه خواهد شد.

ایجاد لینک برای پیوندها و ایمیل:

برای ایجاد لینک سایت و ایمیل در بخش مشتریان می توان از دستورات زیر استفاده کرد:

دستور نمایش لینک به همراه متغییر:

```
<?php echo $html->link('سایت',$client['Client']['url']); ?>
```

دستور نمایش آدرس ایمیل به همراه متغییر:

```
<?php echo $html->link('سایت',$client['Client']['email],'mailto:'.$client['Client']['email']); ?>
```

ایجاد صفحه اصلی جهت ورود کاربران:

فایل موجود در ریشه اصلی سایت app_helper.php را باز کرده و کدهای زیر را می نویسم:

دستور شرطی با نام ifActive کنترلر مربوطه به آن را چک کرده و یک اکشن انتخاب می کند، این دستورات برای بررسی ورود کاربران و هدایت آن ها به صفحه مورد نظر می باشد.

همچنین برای جلوگیری از بروز خطا برای کوچک و بزرگ بودن نام فایل های دستور دیگری نیز افزودیم:

```
class AppHelper extends Helper {

 function isActive($controller, $actions = array())
    {
     foreach ($actions as $action)
     {
      if ($controller == $this->params['controller'] && $action == $this-
>params['action'])
       {
        return true;
       }
     }
     return false;
    }


       function formatCssName($name)
 {
  return strtolower(str_replace(' ', '', $name));
```

```
  }
}
```

اعمال حق دسترسی کاربران به بخش های مختلف:

برای مشاهده پیوندهای مختلف و دسترسی کاربران به بخش ها مربوطه با مراجعه به صفحه مربوط به منو یعنی defalt.ctp در پوشه layout کد منو را بصورت زیر اصلاح می کنیم:

```
        <li <?php if($html-
>isActive('pages', array('display'))) { echo 'class="active"'; }  ?>><?php echo $html-
>link('خانه', '/'); ?></li>

      <li <?php if($html-
>isActive('invoices', array('dashboard'))) { echo 'class="active"'; }  ?>><?php echo $html
->link('صفحه رومیزی', '/dashboard'); ?></li>

      <li <?php if($html-
>isActive('invoices', array('index', 'add', 'edit', 'view'))) { echo 'class="active"'; }  ?>><?p
hp echo $html->link('فاکتورها', '/invoices'); ?></li>

     <li <?php if($html-
>isActive('clients', array('index', 'add', 'edit'))) { echo 'class="active"'; }  ?>><?php echo $
html->link('مشتری ها', '/clients'); ?></li>

        <li <?php if($html-
>isActive('categories', array('index', 'add', 'edit'))) { echo 'class="active"'; }  ?>><?php ec
ho $html->link('موضوعات', '/categories'); ?></li>

</ul>
```

نمایش رنگی وضعیت فاکتورهای ثبت شده:

نمایش آخرین سفارش ها در اولین نتایج صفحه رومیزی:

با توجه به نمایش پیشفرض بصورت نزولی، لازم است تا نتایج را بصورت صعودی نمایش دهیم تا آخرین ثبت شده ها به اولویت پرداخت در صفحه رومیزی به نمایش درآید.

به بخش کنترلرها رفته و invoices_controller.php را باز کرده و تابع مربوط به dashboard را اصلاح می کنیم:

```
function dashboard()
{
      $draftInvoices = $this->Invoice-
>find('all', array('conditions' => "Invoice.status_id = '1'",'order'=> 'Invoice.id DESC'));
      $openInvoices = $this->Invoice-
>find('all', array('conditions' => "Invoice.status_id = '2'",'order'=> 'Invoice.id DESC'));
      $closedInvoices = $this->Invoice-
>find('all', array('conditions' => "Invoice.status_id = '3'",'order'=> 'Invoice.id DESC'));
      $pastdueInvoices = $this->Invoice-
>find('all', array('conditions' => "Invoice.status_id = '4'",'order'=> 'Invoice.id DESC'));

      $this-
>set(compact('draftInvoices', 'openInvoices', 'closedInvoices', 'pastdueInvoices' ));
}
```

برای کوتاه شدن دستورات این کدها را به بخش مدل ها و صفحه invoice.php انتقال داده و جهت نمایش ۵ وضعیت آخر و جلوگیری از شلوغ شدن در بخش صفحه رومیزی برای دستورات نمایش ۵ آخر را توسط دستور limit' => 5 تعریف می کنیم.

کدهای اضافه شده در model » invoice :

```
function draftInvoices()
{
 return $this->find('all', array('conditions' => array('Invoice.status_id' => 1),
                        'order' => 'Invoice.id DESC',
                                'limit' => 5)
            );
}

function openInvoices()
{
 return $this->find('all', array('conditions' => array('Invoice.status_id' => 2),
                        'order' => 'Invoice.id DESC',
                                'limit' => 5)
            );
}

function closedInvoices()
{
 return $this->find('all', array('conditions' => array('Invoice.status_id' => 3),
                        'order' => 'Invoice.id DESC',
                                'limit' => 5)
            );
}
```

```
    function pastDueInvoices()
    {
     return $this->find('all', array('conditions' => array('Invoice.status_id' => 4),
                                     'order' => 'Invoice.id DESC',
                                                'limit' => 5)
                    );
    }

}
```

حال برای فراخوانی تابع های تعریف شده نیاز است تا کدهای مربوط به کنترلر را تغییر دهیم:

مسیر controller » فایل invoices_controller.php :

```
function dashboard()
{
    $draftInvoices = $this->Invoice->draftInvoices();
      $openInvoices =  $this->Invoice->openInvoices();
      $closedInvoices = $this->Invoice->closedInvoices();
      $pastdueInvoices = $this->Invoice->pastDueInvoices();

      $this-
>set(compact('draftInvoices', 'openInvoices', 'closedInvoices', 'pastdueInvoices' ));
}
```

یافتن مشتریان با اول حروف آنها:

در برخی از سیستم های تحت وب بخشی جهت جستجوی نتایج بصورت الفبایی وجود دارد، مثلاً فرض کنید بخواهیم فقط لیست مشتریانی که اول نام آن ها با حرف "پ" شروع شده است را بازیابی کنیم.

در فریم ورک کیک برای این منظور می توانیم از روش زیر استفاده نماییم:

مراجعه به بخش کنترلر و ویرایش فایل Clients_controller.php

تابع index را به این صورت می نویسیم:

```
function index() {
    $this->Client->recursive = 0;
    $this->set('clients', $this->paginate());
        $this->layout = 'client_letters';


    $activeLetter = isset($this->params['named']['letter']) ? $this-
>params['named']['letter'] : '';
', چ', 'ج', 'ث', 'ت', 'پ', 'ب', 'ا','آ       $letters = array('

', ژ', 'ز', 'ر', 'ذ', 'د', 'خ', 'ح                    '

', ع', 'ظ', 'ط', 'ض', 'ص', 'ش', 'س                              '

',);ی','ه','و','ن','م','ل','گ', 'ک', 'ق', 'ف', 'غ                          '

    $urlArgs = array('url' => $this->params['named']);

    $clients = isset($activeLetter) ? $this-
>paginate('Client', array('Client.name LIKE' => $activeLetter.'%')) : $this->paginate();


    $this->set(compact('letters', 'clients', 'activeLetter', 'urlArgs'));
}
```

دستور Client->recursive = 0; جهت ایجاد نقطه توقف برای تابع در نظر گرفته است و در کل نیز می توان گفت تابع های بازگشتی همانند فاکتوریل، برای فاکتوریل صفر، ۱ را باید تعریف نمود که در واقع این به عنوان نقطه شروع و پایانی جهت جلوگیری از حلقه بی نهایت در نظر گرفته می شود.

```
$this->set('clients', $this->paginate());
        $this->layout = 'client_letters';
```

در این دستورات تابع صفحه بندی را تعریف کرده و لایه ای با نام client_letters تعریف می کنیم.

توجه: در فریم ورک کیک می توان برای هر صفحه یک لایه مجزا تعریف نمود که دستورات در آن لایه اجرا شوند برای ایجاد لایه جدید پس از تعریف نام آن در کنترلر آن باید در مسیر view » layout لایه جدید را ساخت.

در دستور isset همانن دستور Get صفحات وب بوده با این تفاوت که فقط برای مقادیری که وجود خارجی دارند در نظر گرفته می شود و صفر و مقادیر دیگر را بصورت پیش فرض null در نظر می گیرد:

در واقع در صورت عدم بکار گیری از دستور isset می بایستی ما برای تک تک حروف مقداری در نظر می گرفتیم:

```
$arr = array(
  'a' => false,
  'b' => 0,
  'c' => '',
  'd' => array(),
  'e' => null,
  'f' => 0.0,
);
```

دستور `$urlArgs = array('url' => $this->params['named']);` باعث می شود تا نام هایی که تعریف شده اند با حروف الفبا بصورت پروسه درخواست به آدرس ارسال شوند.

خط بعدی یعنی دستور:

```
$clients = isset($activeLetter) ? $this-
>paginate('Client', array('Client.name LIKE' => $activeLetter.'%')) : $this->paginate();
```

پروسه مربوط به MySQL را اجرا کرده و با دستور Like عنوان می کند که در جدول مشتریان نتایجی که اول آن ها با حرف $activeLetter جایگزین شده است را نمایش دهد. پارامتر ٪ فقط نتایج اول هر اسم مشتری را بررسی خواهد نمود.

در اینجا لایه ای با نام client_letters.ctp ساخته و از آنجایی که قرار است مشابه صفحات دیگر باشد کلیه کدهای صفحه اصلی default.ctp را کپی کرده و فقط کد نمایش المنت را در بالای بخش نمایش محتوا قرار می دهیم:

```
<?php echo $this->element('letters'); ?>
```

حالا کافیست در پوشه المنت ها صفحه ای با نام letters را ایجاد و دستورات زیر را جهت نمایش به آن بیافزاییم. view

```
<div id="letters">

  <h3 class="heading">جستجو بر اساس حروف الفبا</h3>


    <?php echo $html-
>link('نمایش همه', '/clients/index/letter:', array('class' => empty($activeLetter) ? 'active' : ''
)); ?>

    <?php foreach ($letters as $letter): ?>

      <?php echo $html-
```

```
>link($letter, '/clients/index/letter:'.$letter, array('class' => ($activeLetter == $letter) ? 'act
ive' : '')); ?>

    <?php endforeach; ?>

  <div class="clearfix"></div>

  </div>
```

ایجاد استایل جعبه ای:

به بخش webroot » css » رفته و فایل cake.generic.css را ویرایش می کنیم.

حال کدهای زیر را اضافه می کنیم:

```
#letters {
 margin-bottom: 20px;
}
#letters a {
  background-color: #33B1E9;
  color: #FFFFFF;
  text-decoration: none;
  display: block;
  float: right;
  margin-top: 3px;
  width: 18px;
  height: 18px;
  margin-right: 4px;
  text-align: center;
  font-size: 12px;
}
```

```
#letters a:hover, #letters a.active {
  color: #000;
  background-color: #F9D50F;
}
```

این کدها باعث می گردند تا کلاس تعریف شده با نام letters بصورت یک مربع و پس زمینه آی باشد که صفحه hover آن یعنی حرکت موس از آن رنگ پس زمینه نارنجی را داشته باشد و جعبه فعال صفحه نیز پس زمینه نارنجی با فونت مشکی پدیدار شود.

صفحه بندی بخش فاکتورها:

در صورتی که نتایج موجود در صفحه ها زیاد باشد لود صفحه طولانی شده و بررسی محتوا دشوار خواهد گردید، بدین منظور با استفاده از کنترل کننده ها می توانیم این قابلیت را ایجاد نماییم:

```
var $paginate = array('limit' => 10, 'order' => array('Invoice.invoiceId DESC'));
```

این دستورات باعث می شوند تا ۱۰ نتیجه آخر بصورت صفحه بندی شده به نمایش درآیند.

محاسبه میزان مجموع کل:

برای محاسبه میزان مجموع کل می توان بدون استفاده از بکارگیری تابع و با استفاده از بکار بردن علامت جمع در داخل کد echo میزان مجموع را چاپ نمود:

```
<?php echo $invoice['Invoice']['total']+

$invoice['Invoice']['taxRate']

?>
```