دقیقه ی 0:

در این جلسه issue هارا مدیریت خواهیم کرد دقت کنیم که issue ها به project وابسته هستند یعنی ما نباید بتوانیم به طور مستقیم ان را ایجاد کنیم مطلب دیگر این که بین userها و issueها ارتباط برقرار است از طرف دیگر بین پروژه ها و userها هم ارتباط برقرار است پس ما در این جلسه یک سری relationshibخواهیم داشت ارتبااطات ما از این دست هستند:

User(1..\*)-(0..\*)project

User(1)-(0..\*)issue

project(1)-(0..\*)issue

در مایگریشنه create\_issue\_user\_assignment\_tables باید سه جدول بسازیم که یکی برای کاربران دیگری برای مساله ها و دیگری برای انتسابات.

چون ارتباط کاربران و پروژه ها چند به چند است از ساده ترین راه برای مدیریت ارتباط ان ها استفاده میکنیم که استفاده از یک جدول انتساب است

Issue ها به project وابسته هستند. فک کنم یعنی موجودیت وابسته هستند.

وقتی میگویی up او ان فایل جدیدی که ذخیره کرده ای را اجرا میکند اما اگر همان قبلی را تغییر داده باشی میگوید جدیدا گهی نخورده ای که حالا میخای اعمال کنی حالا فهمیدی migration جریانش چیست؟ و برای چه اعمالش میکنیم

دستورات ایجاد این جداول را در یک مایگریشن مینویسیم که در ان از دو متد safeup و safedown استفاده میکنیم این دو متد حالت ترنزکشن دارند یعنی اگر جدول اول را ایجاد کردند اما جدول دوم و سوم را نکردند (به هر دلیلی نتوانستند) جدول اول هم ذخیره نمیشود. حالت ترنزکشن یعنی عملیات ها به هم وابسته هستند یعنی اگر یکی انجام نشد بقیه هم اجرا نمیشوند یا اگر شده اند لغو میشوند

یک بار در main.php میگوییم نام جداول فلان پسوند را دارد.

درجدولissue:

Project\_id مربوز به پروژه ایست که این مساله به ان تعلق دارد

در status\_id وضعیت issue را تعیین میکنیم که ایا شروع شده است یا شروع نشده یا تمام شده.

در type\_id نوع issue را را تعیین میکنیم (نوعش میتواند سه چیز باشد که ما ان را از طریق منوی کشویی از کاربر دریافت میکنیم)

Owner\_idمالک ان مساله است

Requester\_id کسی است که درخواست ایجاد پروژه را داده.

فیلدهای دیگری هم گذاشته ایم که برای گزارش گیری ها به کارمان می ایند مثلا زمانی ایجاد شده کی تغییر کرده و اپدیت شده .

در کنترل در خط 73 یک متد قرار دارد که با ان میتوان نام labelها را تغییرداد منظورم از label ها ان نوشته هایی است که در کنار مستطیل ها نوشته میشود اسم این متد attributetables است.

دقیقه ی 20

وقتی میخواهیم safedown را بین جداولی که ارتباط با هم دارن لحاظ کنیم باید پس از خالی کردن ان حذفش کنیم. ما ابتدا جدول انتسابات را خالی و حذف میکنیم زیرا در ان کلید خارجی داریم این کار با دستور truncateTable انجام میشود. من کل دستوراتی که برای ایجاد سه جدول نامبرده و روابط شان امده را در فولدر 4 ذخیره خواهم کرد.

دقیقه ی 30:

بعد از ایجاد جداول نوبت به ساخت مدل برای ان ها میرسد

هنگام ایجاد مدل برای یک جدول حتما دقت کنیم که build relation تیک خورده باشد تا ارتباطات بین موجودیت ها لحاظ شود. ما گفتیم که در issue کلید خارجی به project وجود دارد حال اگر این build relationتیک خورده باشد میتوانیم از طریق هر issue به project مربوطه وصل شوییم. تیک دار بودن این گزینه موجب میشود تا درکلاس مدل متدی به نام ralations ایجاد شود که ارتباطات ررا نگه میدارد

چون در جدول انتسابات قرار نیست چیزی به طور مستقیم نوشته شود پس برای ان جدول active record جدا نمی سازیم orm م yii نوشتن در ان را برایمان مدیریت میکند

دقیقه ی 40:

ساخت منوی کشویی برای یک فیلد:

در model generator نام جدول را میدهیم و یک model class برایمان ساخته میشود یه به قول خودشان generate میشود خوب این یعنی چی؟ یعنی کلاس مدلی خواهیم داشت. یعنی یک کلاس خواهیم داشت که پسوند .php دارد مثلا وقتی ما نام جدول را tbl\_user گذاشتیم در پوشه ی models دیدیم کلاسی به نام User ایجاد شده که وقتی درون ان را مشاهده کردیم دیدیم از کلاس cactivitionrecord ارث بری میکند یعنی یه همچین چیزی شد:

class User extends CActiveRecord

در این کلاس یک سری متد جالب وجود دارد که اولی ان ها نام جدول را نگه میدارد. دومی انها هم rules است که در جلسه ی سوم در مورد این متد سخن هایی گفته ایم و کارهایی کرده ایم.متدهای روابط جدول و برچسب های جدول هم درون همین فایل ایجاد میشوند. پس کلاس مدل شامل چنین متدهایی برای یک جدول است. این گونه که من توضیح میدهم شاید بگویید که داریم کلاس مدل میسازیم البته این حرف باید درست باشداما اقای شهرکی به این ها activerecord میگوید و همانگونه که میبینیم عنوان ان کلاس هم دارد از cActiveRecord ارث بری میکند. به عنوان مثال همانگونه که در صفحه ی بالایی اشاره شده میگوید برای جداول واسط activerecord نمیسازیم.(کلاس مدل همان activeRecord) است.

در crud generator نام کلاس مدل را میدهیم

برای ساخت منوی کشویی برای فیلدی از جدول وارد مدل ساخته شده روی ان جدول در فولدرmoddels میشویم. یعنی منظورم این است که وارد activereccord ان موجودیت میشویم و ما پس از این باید یاد بگیریم از این اصطلاح استفاده کنیم.

برای ایجاد منوی کشویی فایل مدل ان موجودیت را از پوشه ی models باز میکنیم

در ان سه ثابت اعلان میکنیم

const type\_bug=0;

const type\_feature=1;

const type\_task=2;

یاداورمیشوم که اعلان ثوابت در php با کلمه ی کلیدیه const انجام میشود.

حال در اداامه ی کار از ان ثوابت در متدی که در اکتیورکورد میسازیم استفاده میکنیم من کد ان را کپی میکنم:

public function getTypeoptions(){

return array(

self::type\_bug=>'nokia',

self::type\_feature=>'sony',

self::type\_task=>'s^amsung',

);

}

گورپدر نوشتن در کاغذ همین الان میخواستم نگاه کنم که <= ایا بررای دسترسی به عناصر ارایه بوده اما میبینم که چیزی دم دست نیست.(بله <= علامتی است که فقط در ارایه ها دیده شده است) اما درکل ایینگونه نیست که type\_bug مقدار nokia را گرفته باشد بلکه انها اندیس ارایه اند و این ها مقادیری که در اندیس قرار گرفته اند

نام تابع هایمان را ترجیحا با get شروع میکنیم چون yii کارهایش را با ثابت جادویی get انجام میدهد وقتی در اول نام تابع get بگذاریم دراینده این اختیارخواهیم داشت که با تابع مثل property رفتار کنیم. ان قسمتی را میبینی که نام شرکت ها را گذاشته ام؟ ان جا جایی است که چیزهایی که میخواهیم کاربر ببیند و یکی شان را انتخاب کند را میگذاریم اما درواقع ثوابت که اعدادی بیش نیستند ارسال میشوند او فکر میکند nokia را ارسال کرده بلکه 0 را ارسال کرده.

مرحله ی بعد کارهایی است که باید در viewی مربوطه انجام دهیم :

فایل فرم ان را باز میکنیم نامش form.php است

نکته: از ان کلاس مدل در contorolers شی ساخته میشود و با همان شی به متدهای کلاسمان که گپی از انها در همین جلسه زدیم دسترسی خواهد داشت ما خودمان هم که داریم متد gettypeoptions را مینویسیم داریم یک متد به این کلاس میافزاییم بدیهی است ان شی میتواند از این کلاس استفاده کند.شهرکی در مورد labelEX و ارتباطش با متد attributelabels حرف هایی زد که از نظرمن بی ارزش بودند و ارزش نوشتن ندارند. اما میخواهم به این دقت کنی که type\_id را در قسمت rules جوری کرده ایم که حتما باید عددی و صحیح باشد و به همین دلیل ما هم ان موقع که ثوابت را تعریف کردیم خواستیم که سه عدد صحیح باشند. زیرا اعتبارات هنوز دارند اعمال میشوند:

<div class="row">

<?php echo $form->labelEx($model,'type\_id'); ?>

<?php echo $form->textField($model,'type\_id'); ?>

<?php echo $form->error($model,'type\_id'); ?>

</div>

که ما به این تغییرش دادیم:

<div class="row">

<?php echo $form->labelEx($model,'type\_id'); ?>

<?php echo $form->dropDownList($model,'type\_id',$model->TypeOptions); ?>

<?php echo $form->error($model,'type\_id'); ?>

</div>

اولین گزینه برای برچسب ها و سومی برای خطایی است که در صورت رد شدن اعتبار باید نشان داده شود

وبرایمان کارکرد ما تنها خط دوم را که textField بود به dropdowntable تغییر دادیم و متناظر با این تغییر پارامترهایش را هم تغییر دادیم و خلاص. خط سوم این هم خطایی را نشان میدهد که اگرrules رعایت نشد اتفاق میافتد.

پس از ان به همین روش برای status لیست کشویی ایجاد کردیم.

Ajax را میتوان فعال کرد.

در پوشه ی views که در پوشه ی protected قرار دارد به ازای هر جدول یک فایل هست که یکی از ان ها views است و یک فرم درون ان هست دران گزینه ای میبینم که نامش فرم است در ان گزینه ای را به حالت

'enableAjaxValidation'=>false,

که احتمالا مربوط به فعالسازی ajax است

دقیقه ی 50 این جلسه هستیم.

نکته: وقتی میخواهیم فرم مربوط به create کردن یک موجودیت را اصلاح کنیم در پوشه ی models کلاس ساخته شده برای ان جدول را یافته و متد هایی در ان تعریف میکنیم و سپس در پوشه ی views فایل مربوط به ان جدول را یافته و از ان متدها استفاده میکنیم

دقیقه ی 60:

نحوه ی استفاده از validator ر in و علت استفاده از ان:

وقتی می ایی یک منوی کشویی ایجاد کنی همواره این امکان وجود دارد که چاکشی با استفاده از inspect element مقداری را که میتواند دران قرار گیرد را تغییر دهد و مقداری بدهد که ما ان را نداریم و در برنامه پردازشی برای ان در نظر نگرفته ایم هرچند احتمال میدهم که استفاده از else و شرط های مناسب گذاشتن هم میتواند کار مارا پیش ببرد.اما فعلا که باید از in اتفاده کنیم:

سوال: اگر بخواهیم متدهایی تعریف کنیم که از ان ها در rules استفاده کنیم این متد ها را کجا تعریف میکنیم؟ ابله مگه نگفتیم کلاس مدل برای یک موجودیت ... اخه چه بگویم احمق؟

برای درک :

کلاس مدل برای یک موجودیت، کلاس ان موجودیت است و اشیایی که از نوع ان کلاس نمونه سازی میشوند به متدهایی که در ان کلاس یا کلاس های والدش است دسترسی دارند پس متدهایی که میخواهی استفاده کنی باید همه در این کلاس تعریف شوند.ضمنا اگر در views فولدر مربوط به یک موجودیت را باز کنی(منظورم از موجودیت جدول است) میبینی که فایل هایی در ان قرار دارند که پسوند .php دارند میدانی این یعنی چه؟یعنی همه به زبان php نوشته شده اند و تو در هرکدام از این ها که بخوایی میتوانی دستوور php بنویسی . و در هرکدام از این ها که بگویی $this به کلاس مدل ساخته شده ببرای ان موجودیت اشاره کرده ای. 000 میدانی این سه تا صفرو برای چی گذاشتم؟ تا بدونی من در این فاصله رفتم چن تا کار انجام دادم یکیشون این بود که در form.php محصولات یک دستور echo نوشتم که کار کرد دوم متدی در همان صفحه ایجاد کردم و صدا زدم که ان هم کار کرد پس به همین راحتی میتوان تغییرایجاد کرد. اما نکته ی ریزی هم اموختم که برایت مینویسم: در انجا <\br> کارنکرد بلکه <br> کار کرد و یاد گرفتم که </br> <br> درست هستند و استفاده از back slash اشتباه است.

خیر در مطالب فوق خطا و اشتباه وجود دارد ما این را با کدهایی که نوشتیم فهمیدیم. اولاخبر داشته باش در پوشه ی contorolers هم فایلی همنام با هر موجودیت وجود دارد و در ان متدهایی تعریف میشود و این ان کلاسی است که با ان فایل های php که در views هستند ارتباط تنگاتنگی دارد و this در واقع به این کلاس اشاره میکند . و من خودم رفتم در contorolers متدی تعریف کردم و در form.php با $this ان را صدا زدم(بدیهی است که وقتی میگویم form.php یا کلاسی که در contorilers است مربوط به یک موجودیت است نه این که بروی در form.php مربوط به یک موجودیت دیگر متدی را صدا بزنی که در contorilers ز کلاس دیگری ان را تعریف کرده ای) اما به متدهای کلاس مدل با $model دسترسی داریم و اکنون دیگر میدانم $model کجا نمونه سازی شده وی در contorolers نمونه سازی میشود.

مطالبی را میخواهم بگویم که شاید جایشان اینجا نباشد اما چند خط قبلی بستری مناسب برای این بحث ایجاد کرده اند.

هرچند $model در form.php در دسترس است اما همین متغیر در \_view.php تعریف نشده است من میخواستم در \_view.php(چون مربوط به view\_ه هر کاربر است) با نوشتن $model->id بتوانم شناسه ی هرکاربر را بدست بیاورم که متوجه مطلب فوق شدم.

حالا تلاش بعدی من برای دست یافتن به شناسهی کاربر موجود را ببین:

<?php echo 'what is your id?'.Yii::app()->user->id; ?>

هیچ فایده ای نداشت و هیچ ننوشت اما من فکر میکنم این کد در اینده کار خواهد کرد. زیرا وقتی ان را به این شکل تغییر دادم(کاملا اتفاقی):

<?php echo 'what is your id?'.Yii::app()->user->name; ?>

دیدم نوشت guest وقتی رفتم با admin به سایت log in کردم نوشت admin و برای دمو هم کار کرد و من انجا فهمیدم که دستور فوق مارا به مشخاصات کاربر log in کرده میرساند حال حدس میزنم اگر بتوانم بعدا بتوانم با کاربران موجود در پایگاه داده به محیط log in کنم این کد مرا به جدول کاربر میرساند.فعلا میروم تا بعد.

من برای این که کارکرد in را درک کنم خودم دارم کارهایی میکنم که چیزهای جانبی دارم یاد میگیرم من میخواهم برای username کاری کنم که تنها عددهای یک سه و هشت را قبول کند ابتدا با نوشتن این دستور ان را فقط عددصحیح کردم

چگونه بگوییم یک فیلد فقط عددی است ان هم عدد صحیح:

array('username','numerical','integerOnly'=>true),

یک مثال رای نوشتن in :

array('username','in','range'=>array(1,3,8,)),

میخواهم به چیزهایی که میگویم خوب دقت کنی:

استفاده از range لازم بوود و بدون ان کار نککرد

بعد از فلش باید مقادیری که میتوانند قرار بگیرند را بنویسیم و دلیل من برای نوشتن این مطلب هم بررسی همین بوده است. اما من میخواهم حالت های دیگری را هم بررسی کنم فعلا بدان همین کار کرده است البته برنامه های من برای درک خواهند بود نه کارایی.

خوب ما توانستیم برنامه ی دیگری هم بنویسیم که ان هم کار میکند ان خط دیگری که در rules نوشتیم این است:

array('username','in','range'=>array('name'=>1,'family'=>3,'keshvar'=>8)),

که میبینی ارایه ی با اندیس رشته ی است.

حال مثالی که این مرد مینویسد بعد از <= یک متد را صدا میزند که یک ارایه ی اندیس رشته ای برمیگرداند و چون متد را در کلاس مدل صدا زده از از شی $model استفاده میکند

فیلتر کردن:

گفتیم issueحتما باید درقالب یک پروژه ایجاد شود پس نیاز داریم تا بستری برای اینکارفراهم کنیم. میخواهیم زمینه ای ای فراهم کنیم که ابتدا پروژه انتخاب شود بعد بتوان issue ایجاد کرد برای اینکار از فیلترینگ استفاده می کنیم فیلترهاکارهایی هستند که بلافاصله قبل یا بعداز یک عمل انجام میشوند مثلا همین که ما باید قبل از ایجاد پروژه log in کنیم یک فیلتر است یافرض کنیدبخواهیم پس از اینکه یک issueحذف شد پروژه ی مربوطه هم ویرایش شود این کارراهم باکمک فیلترانجام میدهیم

ساخت فیلترها:

برای ساخت فیلترهادوراه وجودداردیکی نوشتن یک متد در کلاس کنترولر است و دیگری نوشتن کلاسی جدا برای کنترولر.

اگرازروش ساخت متد در کلاس کنترولر استفاده کنیم باید نام متد با کلمه ی filter شروع شود و الگوی خاصی هم باید داشته باشد یک مثال میتواند این باشد:

Public function filterMyFilter($filterchain){

}

همان گونه که میبینی پارامتری به نام $filterchain دارد که اسم توابعی که باید فیلترشوند در ان قرار میگیرند.

روش ساخت فیلتر با کلاس:

Class myFilter extends cFilter{

Public function prefilter(){}

Public function postfilter(){}

}

باید از کلاس cFilter ارث بری کند و ان دو متد را پیاده سازی کند اولی فیلترهایی که قبل از action باید اجرا شوند و دومین متد برای فیلترهایی که قبل از actionباید اجرا شوند.

خوب ما برای نوشتن فیلتر از روش اول استفاده میکنیم پس وارد کلاس کنترولر میشویم. کلاس کنترولر کدام موجودیت؟ برای issueها چون میخواهیم این فیلتر روی دستور ایجاد issueنوشته شود.

پس متد زیر را در این کلاس ایجاد میکنیم فعلا این متد کار خاصی انجام نمیدهد

public function filterProjectContext($filterchain){

$filterchain->run();

}

در cactiverecord که کلاس کنترولر از ان مشتق شده متدی وجود دارد که باید حتما ان را پیاده سازی کنیم ما ممیتوانیم ان را یک متد انتزاعی فرض کنیم. ما در این متد باید actionهایی را که میخواهیم فیلترینگ رویشان اعمال شود را مشخص کنیم. هر چند ما تاکنون سراغ این متد نرفته ایم اما خودش به صورت پیش فرض overwrite شده و این شکلی هم نوشته شده:

public function filters()

{

return array(

'accessControl', // perform access control for CRUD operations

'postOnly + delete', // we only allow deletion via POST request

);

}

همانگونه که مشاهده میکنی این متد یک ارایه را برمیگردد که هرخانه از این ارایه یک محدودیت است در خانه ی اول کلمه ی accesControl نوشته شده که باعث می شود برای هریک از چهارعمل ازکاربر درخواست log in کند. اگر این کلمه را برداریم دیگر این را از کاربر نمی خواهد من فکر میکنم در فروشگاه اینترنتی نباید این فیلتر را فعال کنیم زیرا ما کاربرانی داریم که عضوسایت نیستند اما حق گشت گذار در سایت رو دارند. خط دوم گفته postOnly+delete که این دستور موجب میشود هنگام حذف اگر id به روش get ارسال شده باشد عمل نکند بلکه فقط به روش post ارسال شده باشد عمل کند.

اکنون قصد داریم در متد فیلتر بگوییم که هنگام ایجاد issue ان متدی که نوشته ایم اجرا شود این را با یک خط دستور مینویسیم به این شکل:

public function filters()

{

return array(

'accessControl', // perform access control for CRUD operations

'postOnly + delete', // we only allow deletion via POST request

'projectContext + create',

);

}

شاید بپرسید ما که فقط گفتیم create و صحبتی از این که هنگام ایجاد کدام موجودیت این دستور ایجاد شود نکردیم پاسخ این است که ما رد کنترولر issue هستیم پس ان چه مینویسیم برای این موجودیت اعمال میشود.به ان + دقت کن میتوانستیم با کاما نام عملیات دیگری را هم بیاوریم \_- هم می توان استفاده کرد که به معنیه (به استثنای) هست. مثلا میتوانستیم بگوییم create,update

خوب برای ادامه ی کار یک فیلد private در کلاس کنترولر ایجاد میکنیم:

private $\_project=null;

حالا میخواهیم با این چه کار کنیم ؟ منطق ما این است که url را چک کنیم و نگاه کنیم ایا pid در ان قرار داده شده است یا نه؟ همین. اگر مقداردهی شده باشد فیلد فوق مقدار دهی میشود اگر نشده باشد که همان null باقی میماند.

درمرحله ی بعد می اییم تابعی دیگر مینویسیم:

protected function loadProject($projectId){

if($this->\_project===NULL)

{$this->\_project= Project::model()->findByPk($projectId);

if ($this->\_project===null)

throw new CHttpException(404,'the requested projected does not exists');

}

return $this->\_project;}

واز ان تابع در اینجا استفاده میکنیم:

public function filterProjectContext($filterchain){

if(isset($\_GET['pid']))

{

$this->loadProject($\_GET['pid']);

}

else {

throw new CHttpException(403,'must specify a project before performing yhis action');

}

$filterchain->run();

}

همانگونه که دیده میشود تابع filterProjectContext خودش ابتدا بررسی مکند که ایا به pid در url مقداردهی شده است یا نه؟ اگر شده باشد تابع loadproject را صدا میزند و میگوید حالا نگاه کن ببین ایا پروزه ای با این id وجود دارد یا نه ؟ در کل چون این دو متد در فیلتری که برای create کردن issue استفاده میکنیم صدازده میشوند اگر در url به pid مقداردهی شده باشد و ان pid در جدول پروژه ها وجود داشته باشد ان فیلد

$this->\_project که در ابتدای کلاس issueContoroler معرفی کردیم مقدار id پروژه ای که issue ها برایش ایجاد میشوند را نگه داری میکند. پس با این تحلیلی که ما کرده ایم ان چه در جلسه ی 5 گفته میشود و دستور زیر نوشته میشود در 10 دقیقه ی اولش:

$model->project\_id=$this->\_project->id;

باید اشتباه باشد زیرا $this->\_project خودش مقدار پروژه ی کنونی دارد به علاوه ان یک شی که نیست(حالا من اینهارا گفتم اما نمیدانم دارم درست میگویم یا نه) اما این را بدان هر کنترولری یک متد actionCreate دارد که در خط ابتدایی ان یک شی از کلاس معدل مربوط به همان موجودیت ساخته میشود و اسم ان هم همیشه مدل است مثلا در viewcontoroler در متد actioncreate دستور زیر نوشته می شود:

$model=new issue;

پس ان چه داری در چهارپنج سطربالاتر میبینی دارد به به فیلد project\_id مقدار id پروژه ی کنونی را نسبت می دهد .

بدان:هر سطر از جدول یک شی از کلاس مدل مربوطه است.

البته مطالبی که در 10 خط گذشته گفته ام ربطی به این جلسه کمتردارد و بیشتر به جلسه ی پنج مربوط است.

چرا هنگامی که میخواهیم نگاه کنیم که در url نگاه کنیم که id مشخص شده یا نه pid را چک میکنیم در حالیکه در هنگام تعریف جدول ما نام فیلدرا idگذاشته بودیم.

هرچند من هنوز از درک منطق کلی برنامه فیلترکردن عاجزم اما این قدر اموختم که که از حالا به بعد وقتی بگوییم create باید شماره ی id را هم مشخص کرده باشیم. به این صورتcretae&pid=1 نمیدانم چرا میگوییم pid اما این را یاد بگیر که از علامت & استفاده کرده ایم.

لابد تاکنون متوجه شده ایم که وقتی وارد یک موجودیت میشوی ان چه در ابتدا مشاهده میشود فایلindexان است برای issueفایل index را باز میکنیم در قسمت $this->menu نام ان منوهایی می اید که در سمت چپ قابل مشاهده هستند مثلا برای create کردن یک گزینه وجود دارد و گزینه ی دیگر که manage است . در انجا میتوانیم تغییراتی را که میخواهیم اعمال کنیم مثلا مردک میگوید.:

array('label'=>'Create Issue', 'url'=>array('create','pid'=>1)),

که این دستور موجب میشود هروقت روی create کلیک کنیم id=1 را برایمان باز کند یعنی مساله ای که برای پروژه ی id=1 وجود دارد.یعنی جوری میشود که ما هرگاه روی ساخت issue کلیک کنیم برای پروؤه ی شماره 1 issue ساخته شود.

ما از درک انچه در پایان جلسه درس داده شده عاجزیم و ان قدر برایش پیام میفرستیم تا کمکمان کند. فعلا ناچار از این جلسه میگذریم هرچند تا پایان ان زیاد نمانده

در شروع جلسه ی 5 هم توضیحاتی در مورد فیلتر ها میدهد که ان را هم یاد نگرفتیم.

یک نکته که در پایان این جلسه می افزاییم یادت باشد این نکته در پایان جلسه 4 امده:

هر record در view.php یک صفحه دارد:

توجه شما را به این نکته جلب میکنم که ما در view از شی ای که در کنترولرساخته بودم استفاده کردیم و متد را صدازدیم. ما در کلاس مدل مربوط به issue یک متد ایجاد کردیم و در فرم که در viewی مربوط به issue قرار داشت با $model از ان استفاده کردیم. این نکته که در این جا نوشته ام نکته ی ارزش مندی است من خودم خوب انچه این جا اموخته ام را درک نکرده بودم تا به کدی که علیرضا نوشت دقت کردم(اصلا اونو علی رضا ننوشته خود yii اینو واسه همه مینویسه. ان شی که در contoroler از کلاس مدل ساختیم فقط یک بار ساخته نمی شود بلکه برای هرسطر از جدول یک شی که yii به صورت پیش فرض نامش را $model می گذارد نمونه سازی میشود. این شی در فایل view.php قابل دسترسی است. این مطلب را در خاطر نگاه دار که view.php خصوصی است منظورم از خصوصی این است که برای هرکدام از سطرها ایجاد میشود. یعنی هرسطر از جدول ما یک فایل view.php برای خودش دارد. مثلا اگر ما کلاسی برای کاربرها داریم که فیلدهایش نام و نام خانوادگی و شناسه و شماره تلفن هستند و در form.php چند کاربر ایجاد کنیم هرکدام ازاین اشخاص یک صفحه خواهند داشت که مشخصاتشان را نشان می دهد. این که این صفحه چه سرو شکلی دارد در view.php مشخص میشود و به شکل زیر میتوان به فیلدهای جدول برای ان کاربر دسترسی داشت:

$model->نام فیلد