بررسی سطح دسترسی کاربران:

روش اول simple access control system

این متدهای filters و accesrules که در ادامه در موردشان صحبت میکنیم در contorolers قرار دارند

که از این روش تا حدودی در جلسات قبل استفاده کردیم (در انجا که متد فیلتر ررا تغییر دادیم)

public function filters()

{

return array(

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

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

);

}

می بینی نوشته شده 'accessControl' این یه معانی دارد که ان ها را در متد زیر مشخص میشوند:

public function accessRules()

{

return array(

array('allow', // allow all users to perform 'index' and 'view' actions

'actions'=>array('index','view'),

'users'=>array('\*'),

),

array('allow', // allow authenticated user to perform 'create' and 'update' actions

'actions'=>array('create','update'),

'users'=>array('@'),

),

array('allow', // allow admin user to perform 'admin' and 'delete' actions

'actions'=>array('admin','delete'),

'users'=>array('admin'),

),

array('deny', // deny all users

'users'=>array('\*'),

),

);

}

این قسمتش را ببین:

array(

array('allow', // allow all users to perform 'index' and 'view' actions

'actions'=>array('index','view'),

'users'=>array('\*'),

این یعنی همه ی کاربران به action های index و view دسترسی دارند.

قسمت دومش یعنی فقط کاربران ثبت نام کرده میتوانند index view کنند.

و قسمت سوم هم یعنی فقط admin میتواند admin delete کند.

روش دوم کنترل دسترسی مبتنی بر نقش:rbac

دقیقه ی ده:

Contorolers

Roles

Ips

Verbs

Expression

Actions

users

Yii به بزرگی و کوچکی حروف حساس نیست

از دقیقه ی 15 تا 25 متد accesrules را به طور کامل معرفی کرد

به نظرم ما برای فروشگاه باید همچین ارایه ای داشته باشیم:

Array(‘allow’,

‘actions’=>array(‘index’,’view’),

‘users’=>array(‘\*’)

)

و این را برای این میگویم تا همه ی کاربران اجازه ی دیدن محصولات و مشخصاتشان را داشته باشند.

من با نوشتن کد زیر تونستم اختیارات سابق admin رو در اختیار کاربر ali بگذارم:

array('allow', // allow admin user to perform 'admin' and 'delete' actions

'actions'=>array('admin','delete'),

'users'=>array('ali'),

),

البته اینکارو در کلاس contorolers که والد همه شونه و در components قرار داره انجام دادم. البته دقت کنیم وقتی داریم متد accesrules رو وقتی در contorolers کپی می کنیم همه ی کنترولرها حتی sitecontoroler هم از ان ارث بری می کند که ما این را نمیخواهیم زیرا sitecontoroler برای خودش accesrules دارد پس ما در متد accesrules کدی را مینویسیم که گفته باشیم این ها تنها برای فلان کنترولر ها هستند ما باید این دستور برای برای هرخانه از ارایه accesrules بنویسیم:

‘contorilers’=>array(‘issue’,’project’,’user’),

مثلا فرض کن دستور اول این باشد که اجازه ی create index را به کاربران ثبت نام کرده بدهیم این دستور به این شکل در میاید:

Array(

‘allow’,

‘contorilers’=>array(‘issue’,’project’,’user’),

‘actions’=>array(’create’,’index’),

‘users’=>array(‘@’),),

و پس از این دستور اگر بخواهیم اختیارات دیگری بدهیم یا بگیریم باز دوباره در array() ان چه میخواهیم را می دهیم یا میگیریم و در ان هم باید

‘contorilers’=>array(‘issue’,’project’,’user’),

را بنویسیم و الا برای همه لححاظ میشود.

),

دقیقه ی 30:

حالا میخواهیم نقش تعریف بکنیم و براساس این نقش ها بگوییم که کدام نقش چه سطح دسترسی هایی دارد.

نکته:mvc rbac معماری هایی هستند که ییی ان هارا پیاده سازی میکند و این طور نیست که فقط ییی ان هار داشته باشد هر سیستمی میتواند ان هارا پیاده سازی کند ییی هم یک از ان هایی است که این پیاده سازی را انجام میدهد

RBAC IN YII CONTAINS OF:

Role: owner--- member--- reader

Tasks: usermanagement--- isuuemanagement---projectmanagement

Operation:create user--- delete user,…

هر نقش میتواند شامل یک یا چند نقش باشد هم چنین یک نقش می تواند شامل نقش های دیگری هم باشد یعنی زیرنقش داشته باشد اگر این کار صورت گیرد تمام مجوز هایی که به یک نقش داده شده به تمام زیر نقش های ان هم داده میشود

وظایف هم میتوانند شامل عملیات هایی باشند و همچنین میتوانند وظایف دیگری هم داشته باشند اگر این کار صورت گیرد تمام عملیات هایی که برای یک وظیفه تعریف کرده ایم به زیروظیفه ی ان هم میرسد.

در ییی این امکان وجود دارد که به یک نقش هم به طور مستقیم یک عملیات نسبت دهیم یا ازوی بگیریم مثلا بگوییم فلان کاربر تو هرچند یک reader بیش نیستی اما میتوانی پروزه تعریف کنی یعنی عملیات create project را انجام دهی . یا میتوانیم یک عملیات را از کسی بگیریم مثلا بگوییم . این ها را برای این گفتیم تا بدانی rbac در ییی انعطاف پذیر است.

برای استفاهد از معماری rbac ما باید ان را در main.php اضافه کنیم . برای این کار یک کامپوننت اضافه میکنیم authManager را اضافه میکنیم که در فریم ورک ییی همان rbac است. متد main در کل یک ارایه برمیگرداند و هرکدام از خانه هایش هم میتوانند یک ارایه باشند به عنوان مثال یک از خانه هایش که نامش component است componentهایی را که در برنامه وجود دارند را مشخص میکند و البته که اگر ما هم بخواهیم یک componentاضافه کنیم باید در همین قسمت ان را بیفزاییم.

نکته ای مربوط به نیم ساعت اول این جلسه:

علاوه بر @ و \* یک ؟ هم بود که نمیدانم چه کاربرانی را معرفی میکرد اما در نیم ساعت گذشه ی این جلسه این را درسس داده

وقتی ما کامپوننت داشته باشیم دیگر هر جا که بخواهیم میتوانیم به شکل زیرر به ان دسترسی داشته باشیم:

Yii::app()->name of componenet

Example:Yii::app()->authManagement

مثال دیگر این که ما در شل با دستور زیر به فیلدی از کامپوننت db دسترسی پیدا کردیم:

Example:Yii::app()->db->connectionString

خوب پس ما هم میاییم و این مطلب را می نویسیم:

‘aouthMath’=>array(

‘class’=>’CDbAuthManager’,

‘connectionID’=>’db’,

),

که ما از connectionID’ زیاد استفاده نمیکنیم اما کلاس ما به جای ’CDbAuthManager’ میتواند’CPHPAuthManager’

باشد که ما از ان استفاده نمیکنیم ما از ’CDbAuthManager’ تا برایمان جداول را در پایگاه داده بسازد.

ییی خودش task را به rabc افزوده ما در این پروزه برای سادگی کار tastk را حذف میکنیم .

حال جداولی را پایگاه داده ایجاد میکنیم و این کار ار با نوشتن migration انجام میدهیم.

اما قبل از ان جالب است بدانیم جداولی که میسازیم یک ساختار ثابت و مشخص شده دارند یعنی نام فیلدها و این که از چه نوعی باشند از پیش تعریف شده است این ها در فایل schema-mysql.sql که در مسیر زیر قرار دارند هستند من خودم رفتم این فایل را ببا نوت پد باز کردم و صحت سخن فوق را مشاهده کردم:

C:\xampp\htdocs\yii\framework\web\auth

خوب پس اکنون میرویم یک مایگریشن حدید به نام create \_rbac میسازیم.

:

Authitam

Authitemchild

Authutemasignment

نام جداول باید یکی از سه نام فوق باشد ما نام های دیگری میدهیم اما بعد معالسازی میکنیم.

و در متد up ان دستورات زیر را مینویسیم :

در فیلد name باید نام item ی که میخواهیم را بنویسیم مثلا اگر بخواهیم یک نقش به اسم owner ایجاد کنیم کلمه ی owner در فیلد name قرار میگیرد یا اگر بخواهیم یک عملیات ایجاد کنیم که مثلا نامش create user باشد create user در نام قرار میگیرد.

گزینه ی دوم type هست که یک عدد است که نوعش را مشخص میکند اگر صفرباشد یک عملیات است اگر یک باشد یک task است و اگر دو باشد یعنی یک نقش اشت.

Description یک توضیح درباره ی ان است .

Bizrule مال زمانی است که ما بخواهیم این ساختار rbac راا گسترش دهیم مثلا در همین پروزه ی track star ما از این فیلد استفاده میکنیم تا شرطی بگذاریم که ایا این کاربر در این پروزه مجاز هست یا نه. زیرا نقش های ما برای هر پروزه تغییرمیکند.

جدول دوم:

Authitemchild

جدول معادلی که ما برای این جدول ایجاد میکنیم tbl\_auth\_item\_child است که رابطه ی پدرفرزندی یعنی ارث بری ایجاد میکند مثلا create user فرزند owner میشود. در ان صورت owner والد و create user فرزند میشود.

هردو فیلد جدول دوم به فیلد NAME در جدول اول ارجاع دارند.

جدول سوم:

این جدول قرار است معادل

Authutemasignment

است که ما ان را با نام tbl\_auth\_assignment ان را ایجاد میکنیم.

چون گمان میکنم که دستورات ایجاد این سه جدول همیشه تکرار میشوند ان ها را در فولدری به نام (جداول rbac) ذخیره میکنم.