components/AccessRule.php file. Inside our AccessRule class, we have overridden the matchRole method on our own, where we get and check the current user role and match it with roles from our rules.

Finally, we need to deny access to a specific browser. For this recipe, we are denying only Internet Explorer 9. The rule itself is put on top, so it executes first, as follows:

[

'allow' => true,

'actions' => ['user'],

'matchCallback' => function ($rule, $action) {

return preg\_match('/MSIE 9/',$\_SERVER['HTTP\_USER\_AGENT'])!== false;

}

],

The detection technique that we are using is not very reliable, as MSIE is contained in many other user agent strings. For a list of possible user agent strings, you can refer to <http://www.useragentstring.com/>.

In the preceding code, we used another filter rule property named ' matchCallback'. This property will apply only when functions which are described in this property return true.

Our function checks if the user agent string contains MSIE 9.0 sting. Depending on your requirements, you can specify any PHP code.

See also

In order to learn more about access control and filters, refer to the following:

* [http://www.viiframework.com/doc-2.0/guide-structure-filters.html](http://www.yiiframework.com/doc-2.0/guide-structure-filters.html)
* <http://www.yiiframework.com/doc-2.0/yii-filters-accesscontrol.html>
* <http://www.yiiframework.com/doc-2.0/yii-filters-accessrule.html>
* [https://github.com/yiisoft/yii2/hloh/master/docs/guide/structure-filters.md](https://github.com/yiisoft/yii2/blob/master/docs/guide/structure-filters.md)
* [http://www.yiiframework.com/doc-2.0/guide-security-authorization.html#access-control-filter](http://www.yiiframework.com/doc-2.0/guide-security-authorization.html%23access-control-filter)
* The Using RBAC recipe