RBAC Authorization on an yii2-framework based system

2020/03/13 (Fri)

By Umbalaconmeogia

Table of Contents

Table of Contents

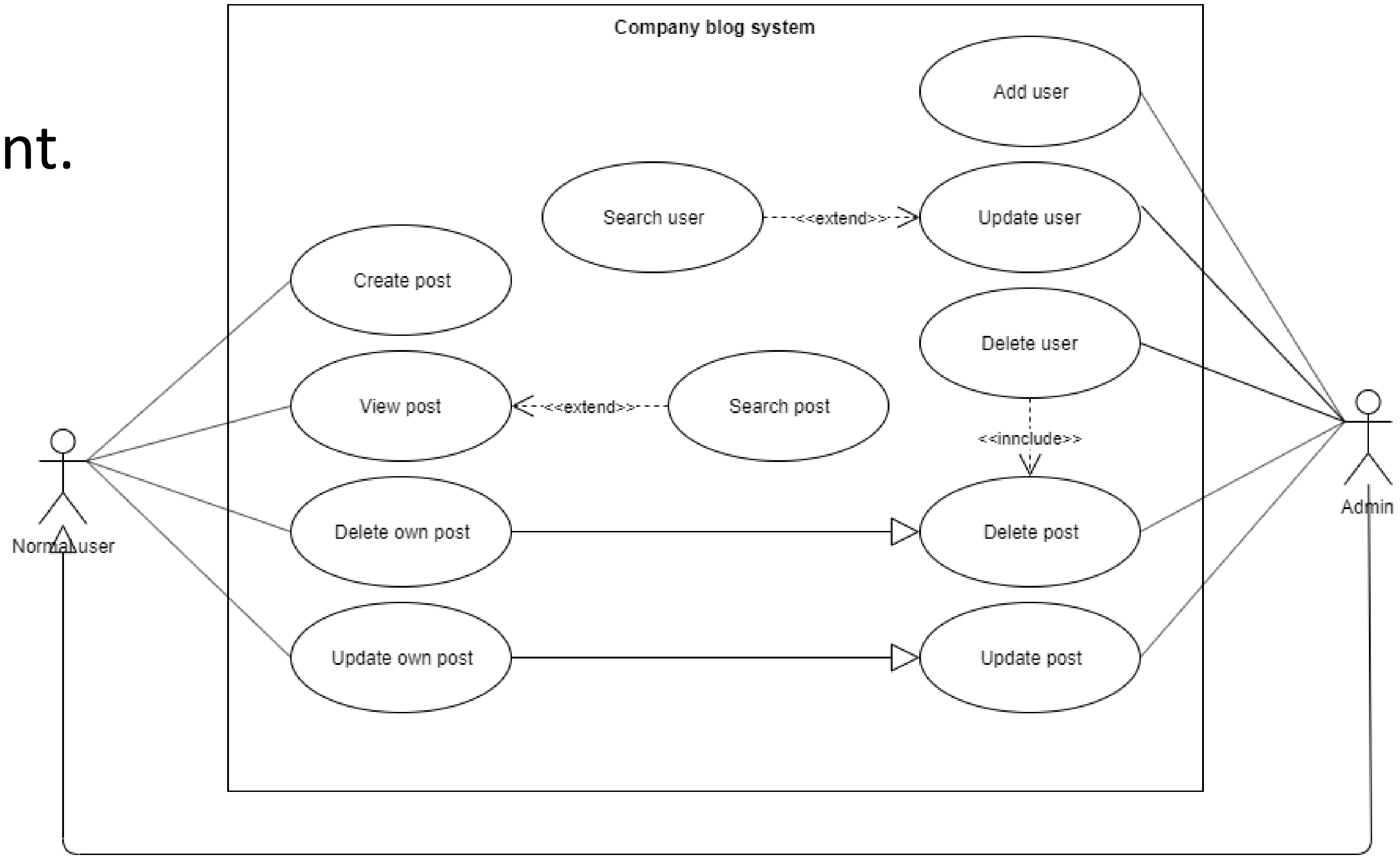
Overview

- Purpose of this document: Introduce about RBAC on yii2 framework, with some usage example.
- Prerequisite:
  - Audience should have basic experience of using yii2 framework.
  - Read about Authorization on yii2

Table of Contents

Sample project: Company blog system

- A simple system to post and view content.



Database design

- TBD

Table of Contents

Table of Contents

Access Control Filter

- Decide if current web user can access to a controller's action.
- Filter defined in controller's behaviors()
- Reference:
  - [Access Control Filter]()
  - [Request handling overview]()

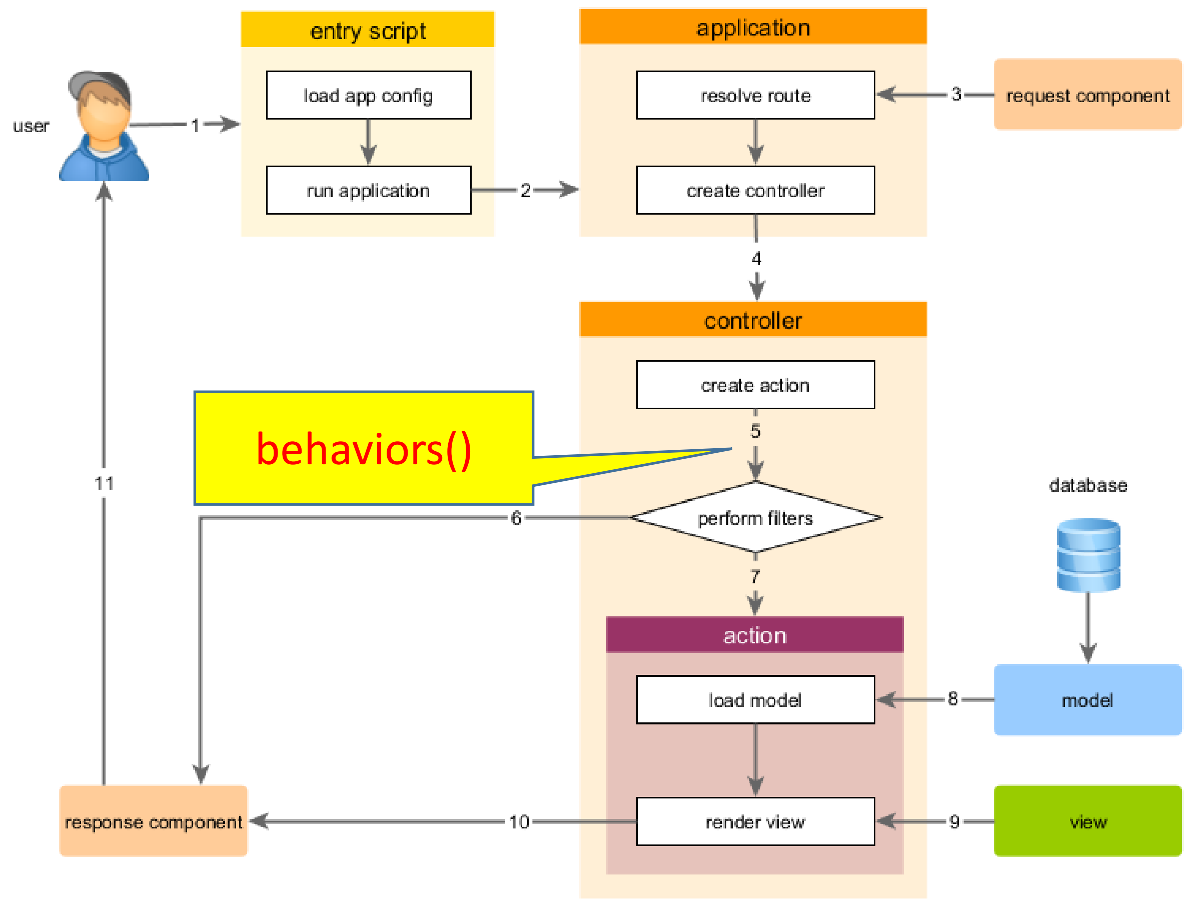


Table of Contents

RBAC

- RBAC is a method to build access control.
- It builds a way to check if current user have permission to do something.
  - RBAC can be used as filter in ACF.
  - RBAC can be used in any part of code.
- 2 steps to use RBAC:
  1. Build RBAC data (define permission/role/rule and assign permission).
  2. Use RBAC in code.
- Reference: Role Based Access Control

RBAC used in ACF

Simple roles defined in ACF

```
public function behaviors()
{
    return [
        'access' => [
            'class' => AccessControl::className(),
            'only' => ['login', 'logout', 'signup'],
            'rules' => [
                [
                    'allow' => true,
                    'actions' => ['login', 'signup'],
                    'roles' => ['?'],
                ],
                [
                    'allow' => true,
                    'actions' => ['logout'],
                    'roles' => ['@'],
                ],
            ],
        ],
    ];
}
```

ACF using roles defined by RBAC

```
public function behaviors()
{
    return [
        'access' => [
            'class' => AccessControl::className(),
            'rules' => [
                [
                    'allow' => true,
                    'actions' => ['index'],
                    'roles' => ['managePost'],
                ],
                [
                    'allow' => true,
                    'actions' => ['view'],
                    'roles' => ['viewPost'],
                ],
                [
                    'allow' => true,
                    'actions' => ['create'],
                    'roles' => ['createPost'],
                ],
                [
                    'allow' => true,
                    'actions' => ['update'],
                    'roles' => ['updatePost'],
                ],
                [
                    'allow' => true,
                    'actions' => ['delete'],
                    'roles' => ['deletePost'],
                ],
            ],
        ],
    ];
}
```

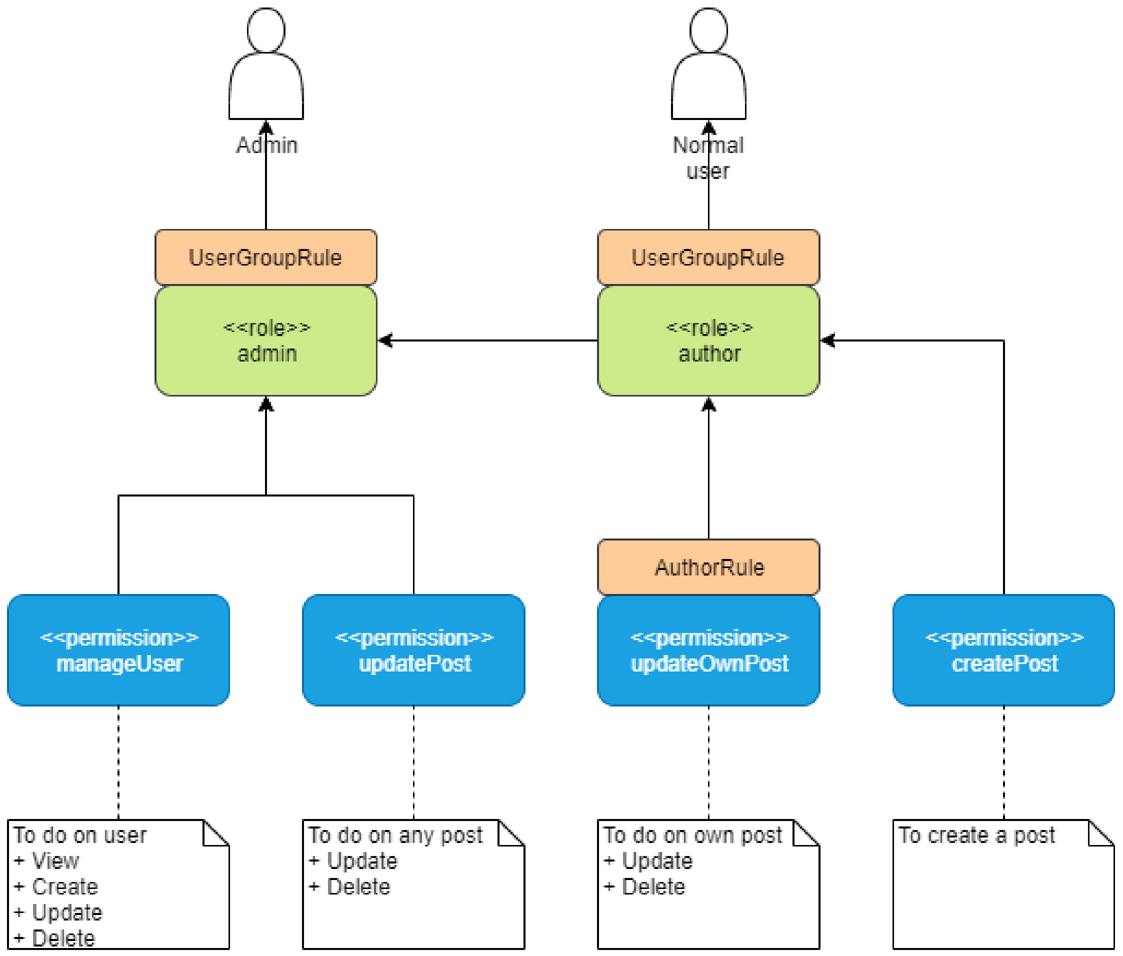
RBAC used in code

```
if (¥Yii::$app->user->can('createPost')) {
    // create post
}

if (¥Yii::$app->user->can('updatePost', ['post' => $post])) {
    // update post
}
```

RBAC data design

- RBAC data doesn't need relate to system function (it may be common mistake of access control design to create access permission same as user function)
- Any logged in user can view post, so it is unnecessary to define permission for viewing post (we have AFC definition for viewing post).



RBAC permission and role

Table of Contents