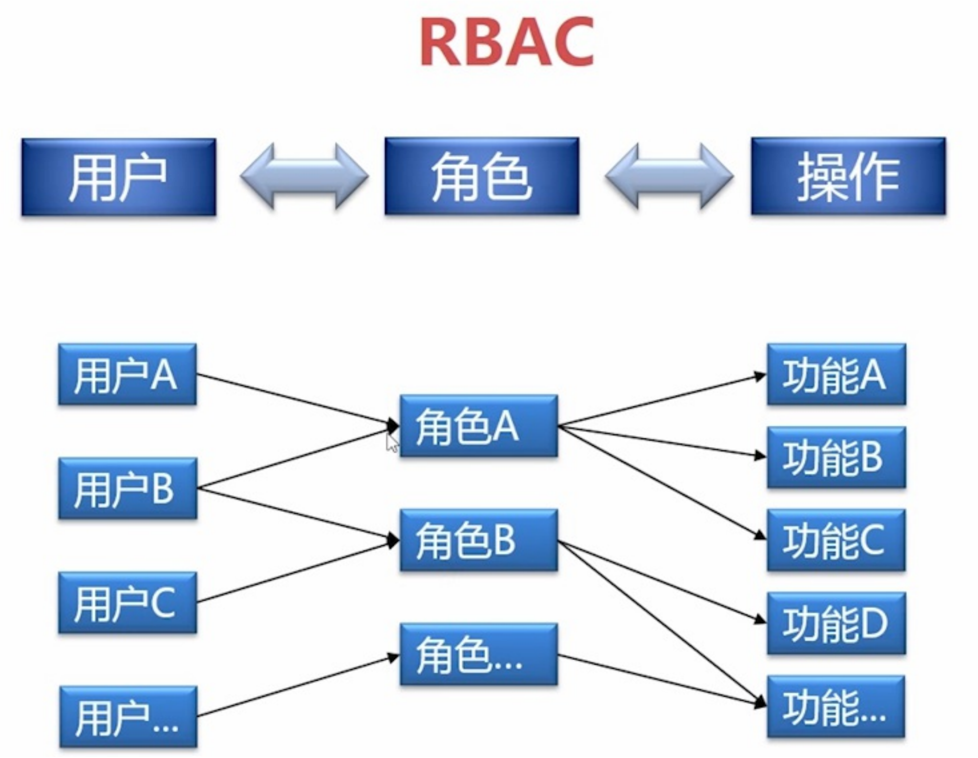
link: null
title: 珠峰架构师成长计划
description: null
keywords: null
author: null
date: null
publisher: 珠峰架构师成长计划
stats: paragraph=110 sentences=219, words=1701

1.RBAC

- 基于角色的权限访问控制（Role-Based Access Control）
- RBAC（Role-Based Access Control，基于角色的访问控制），就是用户通过角色与权限进行关联
- 一个用户拥有若干角色,每一个角色拥有若干权限。这样，就构造成 &#x7528;&#x6237;-&#x89D2;&#x8272;-&#x6743;&#x9650;的授权模型
- 在这种模型中，用户与角色之间，角色与权限之间一般是多对多的关系
- 在RBAC中最重要的概念包括：用户(User)，角色(Role)，权限(Permission)

1.1 示意图



1.2 表设计

1.2.1 用户表(user)

字段 字段名 类型 默认 id ID int(11) userName 用户名 varchar(255) password 密码 varchar(255)

```
CREATE TABLE `user`  (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `userName` varchar(255),
  `password` varchar(255),
  PRIMARY KEY (`id`)
)

INSERT INTO `user` VALUES (1, 'isadmin', '123456');
INSERT INTO `user` VALUES (2, 'isuser', '123456');
```

1.2.2 角色表(role)

字段 字段名 类型 默认 id ID int(11) name 名称 varchar(255) desc 描述 varchar(255)

```
CREATE TABLE `role`  (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(255) ,
  `desc` varchar(255) ,
  PRIMARY KEY (`id`)
) E

INSERT INTO `role` VALUES (1, 'admin', '管理员');
INSERT INTO `role` VALUES (2, 'user', '普通用户');
```

1.2.3 权限表(permission)

字段 字段名 类型 默认 id ID int(11) name 名称 varchar(255) parent_id 父ID int(11) icon 图标 varchar(255) key 路径 varchar(255) type 类型 varchar(32)

```
CREATE TABLE `permission`  (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(255) ,
  `parent_id` int(11) NULL DEFAULT NULL,
  `icon` varchar(255) ,
  `key` varchar(255) ,
  `type` varchar(255) ,
  PRIMARY KEY (`id`)
);

INSERT INTO `permission` VALUES (1, '授权平台', 0, 'desktop', '/api', 'menu');
INSERT INTO `permission` VALUES (2, '角色管理', 1, 'team', '/api/role', 'menu');
INSERT INTO `permission` VALUES (3, '用户管理', 1, 'user', '/api/user', 'menu');
INSERT INTO `permission` VALUES (4, '权限管理', 1, 'idcard', '/api/resource', 'menu');
INSERT INTO `permission` VALUES (5, '添加用户', 3, 'team', '/api/user/add', 'button');
INSERT INTO `permission` VALUES (6, '删除用户', 3, 'team', '/api/user/delete', 'button');
```

1.2.4 角色用户表(role_user)

字段 字段名 类型 role_id 角色ID int(11) user_id 用户ID int(11)

```
DROP TABLE IF EXISTS `role_user`;
CREATE TABLE `role_user`  (
  `role_id` int(11) NOT NULL,
  `user_id` int(11) NOT NULL,
  PRIMARY KEY (`user_id`, `role_id`)
)

INSERT INTO `role_user` VALUES (1, 1);
INSERT INTO `role_user` VALUES (2, 2);
```

1.2.5 角色权限(role_permission)

字段 字段名 类型 role_id 角色ID int(11) permission_id 资源ID int(11)

```
CREATE TABLE `role_permission`  (
  `role_id` int(11) NOT NULL,
  `permission_id` int(255) NOT NULL,
  PRIMARY KEY (`role_id`, `permission_id`)
)

INSERT INTO `role_permission` VALUES (1, 1);
INSERT INTO `role_permission` VALUES (1, 2);
INSERT INTO `role_permission` VALUES (1, 3);
INSERT INTO `role_permission` VALUES (1, 4);
INSERT INTO `role_permission` VALUES (1, 5);
INSERT INTO `role_permission` VALUES (1, 6);
INSERT INTO `role_permission` VALUES (2, 1);
INSERT INTO `role_permission` VALUES (2, 4);
```

1.2.6 数据库脚本

- cms.sql (http://img.zhufengpeixun.cn/cms.sql)

2. 后端

2.1.初始化项目

- egg.js (https://eggjs.org/zh-cn/intro/quickstart.html)
- api (https://eggjs.org/api/)

```
mkdir client-side
cd client-side
cnpm init egg --type=ts
cnpm i
cnpm run dev
```

2.2.支持MYSQL

- mysql (https://eggjs.org/zh-cn/tutorials/mysql.html)

2.2.1 安装

```
cnpm i --save egg-mysql
```

2.2.2 开启插件

- config\plugin.ts

```
import { EggPlugin } from 'egg';

const plugin: EggPlugin = {
+  mysql: {
+    enable: true,
+    package: 'egg-mysql'
+  }
};
export default plugin;
```

2.2.3 配置数据源

config\config.default.ts

```
import { EggAppConfig, EggAppInfo, PowerPartial } from 'egg';

export default (appInfo: EggAppInfo) => {
  const config = {} as PowerPartial;

  // override config from framework / plugin
  // use for cookie sign key, should change to your own and keep security
  config.keys = appInfo.name + '_1580620890875_8931';

  // add your egg config in here
  config.middleware = [];

+  config.mysql = {
+    // 单数据库信息配置
+    client: {
+      // host
+      host: 'localhost',
+      // 端口号
+      port: '3306',
+      // 用户名
+      user: 'root',
+      // 密码
+      password: 'root',
+      // 数据库名
+      database: 'cms'
+    },
+    // 是否加载到 app 上，默认开启
+    app: true,
+    // 是否加载到 agent 上，默认关闭
+    agent: false,
+  };
  // add your special config in here
  const bizConfig = {
    sourceUrl: `https://github.com/eggjs/examples/tree/master/${appInfo.name}`,
  };

  // the return config will combines to EggAppConfig
  return {
    ...config,
    ...bizConfig,
  };
};
```

2.2.4 配置声明文件

- config\config.default.ts

```
import 'egg';

declare module 'egg' {
+    interface Application {
+        mysql: any;
+    }
}
```

2.2.4 使用mysql

- app\controller\home.ts

```
import { Controller } from 'egg';

export default class HomeController extends Controller {
  public async index() {
    const { ctx } = this;
+    let users = await this.app.mysql.select('user');
+    ctx.body = {
+      success: true,
+      data: users
+    };
    //ctx.body = await ctx.service.test.sayHi('egg');
  }
}
```

2.3.Passport

- passport (https://eggjs.org/zh-cn/tutorials/passport.html)

2.3.1 安装

```
cnpm i --save egg-passport passport-local
```

2.3.2 config\plugin.ts

config\plugin.ts

```
import { EggPlugin } from 'egg';

const plugin: EggPlugin = {
  mysql: {
    enable: true,
    package: 'egg-mysql'
  },
+  passport: {
+    enable: true,
+    package: 'egg-passport'
+  },
+  passportLocal: {
+    enable: true,
+    package: 'egg-passport-local'
+  }
};

export default plugin;
```

2.3.3 typings\index.d.ts

typings\index.d.ts

```
import 'egg';

declare module 'egg' {
    interface Application {
         mysql: any;
+        passport: any;
    }
}
```

2.3.4 app.ts

app.ts

```
import { Application, IBoot } from 'egg';
import { Strategy } from 'passport-local';
export default class FooBoot implements IBoot {
    private readonly app: Application;

    constructor(app: Application) {
        this.app = app;
    }
    configDidLoad() {

        let { app } = this;
        app.passport.use(new Strategy({ usernameField: 'userName', passReqToCallback: true }, async (req, userName, password, done) => {
            const users = await this.app.mysql.select('user', { where: { userName, password }, limit: 1, offset: 0, });
            if (users && users.length > 0) {
                let user = users[0];
                let roles = await this.app.mysql.query(`SELECT role.* FROM user INNER JOIN role_user ON user.id = role_user.user_id inner JOIN role ON
role_user.role_id=role.id WHERE user.id=?`, [user.id]);
                user.currentAuthority = roles.map(role => role.name);
                return done(null, user);
            } else {
                req.ctx.isAuthenticated() && req.ctx.logout();
                return done(null, false);
            }
        }));
    }
}
```

2.3.5 app\controller\user.ts

app\controller\user.ts

```
import { Controller } from 'egg';

export default class userController extends Controller {
  public async loginCallback() {
    let { ctx } = this;
    if (ctx.isAuthenticated()) {
      ctx.body = {
        status: 'ok',
        type: ctx.user.type,
        currentAuthority: ctx.user.currentAuthority
      }
    } else {
      ctx.body = {
        success: false,
        error: '用户名或密码错误'
      }
    }
  }
  public async addUser2() {
    let { ctx } = this;
    if (ctx.isAuthenticated()) {
      let user = ctx.user;
      let url = ctx.url;
      let permissions = await this.app.mysql.query(`SELECT permission.key FROM user INNER JOIN role_user ON user.id = role_user.user_id inner JOIN
role_permission ON role_user.role_id=role_permission.role_id  INNER JOIN permission ON role_permission.permission_id = permission.id  WHERE user.id=?`,
[user.id]);
      let allowed = permissions.map(item => item.key).includes(url);
      if (allowed) {
        ctx.body = {
          success: true,
          error: '添加用户成功'
        }
      } else {
        ctx.body = {
          success: false,
          error: '用户未授权'
        }
      }
    } else {
      ctx.body = {
        success: false,
        error: '用户未授权'
      }
    }
  }
  public async addUser() {
    let { ctx } = this;
    ctx.body = {
      success: true,
      error: '添加用户成功'
    }
  }
}
```

2.3.6 middleware\auth.ts

app\middleware\auth.ts

```
module.exports = (_options, _app) => {
    return async function (ctx, next) {
        if (ctx.isAuthenticated()) {
            let user = ctx.user;
            let url = ctx.url;
            let permissions = await ctx.app.mysql.query(`SELECT permission.key FROM user INNER JOIN role_user ON user.id = role_user.user_id inner JOIN
role_permission ON role_user.role_id=role_permission.role_id  INNER JOIN permission ON role_permission.permission_id = permission.id  WHERE user.id=?`,
[user.id]);
            let allowed = permissions.map(item => item.key).includes(url);
            if (allowed) {
                await next();
            } else {
                ctx.body = { success: false, error: '用户未授权' };
            }
        } else {
            ctx.body = { success: false, error: '用户未授权' };
        }
    }
}
```

2.3.7 app\router.ts

- middleware (https://eggjs.org/zh-cn/basics/middleware.html#%E9%85%8D%E7%BD%AE)

app\router.ts

```
import { Application } from 'egg';
export default (app: Application) => {
  const { controller, router } = app;
  router.get('/', controller.home.index);
+  const localStrategy = app.passport.authenticate('local', { successRedirect: '/server/api/loginCallback', failureRedirect: '/server/api/loginCallback' });
+  router.post('/api/login/account', localStrategy);
+  const auth = app.middleware.auth();
+  router.post('/api/user/add', auth, controller.user.addUser);
+  router.get('/api/loginCallback', controller.user.loginCallback);
};
```

2.3.8 config\config.default.ts

- config\config.default.ts
- csrf (https://eggjs.org/zh-cn/core/security.html#web-%E5%AE%89%E5%85%A8%E6%A6%82%E5%BF%B5)

```
+  config.security = {
+    csrf: false
+  }
```

2.3.9 测试

- insomnia (https://insomnia.rest/)
- window (http://img.zhufengpeixun.cn/Insomnia.Setup.7.0.6.zip)
- debug (https://eggjs.org/zh-cn/core/development.html#%E8%B0%83%E8%AF%95)

ctrl+shift+p

```
Debug: Toggle Auto Attach
```

3. 前端

3.1. 启动项目

- create-umi (https://github.com/umijs/create-umi)

```
umi -v
umi ui
```

3.2. 设置代理

3.2.1 config\config.ts

config\config.ts

```
+  proxy: {
+    '/server/api/': {
+      target: 'http://localhost:7001/',
+      changeOrigin: true,
+      pathRewrite: { '^/server': '' },
+    },
+  },
```

3.3. 前端项目

3.3.1 config\config.ts

config\config.ts

```
  routes: [
    {
      path: '/user',
      component: '../layouts/UserLayout',
      routes: [
        {
          name: 'login',
          path: '/user/login',
          component: './user/login',
        },
      ],
    },
    {
      path: '/',
      component: '../layouts/SecurityLayout',
      routes: [
        {
          path: '/',
          component: '../layouts/BasicLayout',
          authority: ['admin', 'user'],
          routes: [
            {
              path: '/',
              redirect: '/welcome',
            },
            {
              path: '/welcome',
              name: 'welcome',
              icon: 'smile',
              component: './Welcome',
            },
            {
              path: '/admin',
              name: '权限平台',
              icon: 'crown',

              routes: [
                {
                  name: '角色管理',
                  icon: 'smile',
                  path: '/admin/role',
                  component: './admin/Role',
                  authority: ['admin']
                },
                {
                  name: '用户管理',
                  icon: 'smile',
                  path: '/admin/user',
                  component: './admin/User',
                  authority: ['admin', 'user']
                },
                {
                  name: '权限管理',
                  icon: 'smile',
                  path: '/admin/permission',
                  component: './admin/Permission',
                  authority: ['admin']
                },
              ],
            },
            {
              component: './404',
            },
          ],
        },
        {
          component: './404',
        },
      ],
    },
    {
      component: './404',
    },
  ],
```

3.3.2 User\index.tsx

src\pages\admin\User\index.tsx

```
+import { queryRule, updateRule, addUser, removeRule } from './service';
+import Authorized from '@/utils/Authorized';
+
+              handleModalVisible(true)}>
+               新建
+
+
```