基于ThinkPHP8构建集成Auth验证的权限管理系统详细介绍

下面为你详细介绍如何基于 ThinkPHP8 构建一个集成 Auth 验证的权限管理系统，涵盖核心功能实现、数据表设计、关键代码示例等内容。

一、系统设计思路

权限管理系统通常包含**用户管理**、**角色管理**、**权限管理**三大核心模块，采用 **RBAC（基于角色的访问控制）** 模型设计，即：用户关联角色，角色关联权限，通过角色间接控制用户权限。

二、数据表设计

需要创建 4 张核心数据表：用户表、角色表、权限表、用户 - 角色关联表、角色 - 权限关联表。

1. 用户表（tp_user）

```
CREATE TABLE `tp_user` (
  `id` int(11) NOT NULL AUTO_INCREMENT COMMENT '用户ID',
  `username` varchar(50) NOT NULL COMMENT '用户名',
  `password` varchar(255) NOT NULL COMMENT '密码（哈希加密）',
  `nickname` varchar(50) DEFAULT NULL COMMENT '昵称',
  `status` tinyint(1) DEFAULT 1 COMMENT '状态：1正常，0禁用',
  `create_time` int(11) DEFAULT NULL COMMENT '创建时间',
  `update_time` int(11) DEFAULT NULL COMMENT '更新时间',
  PRIMARY KEY (`id`),
  UNIQUE KEY `username` (`username`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='用户表';
```

2. 角色表（tp_role）

```
CREATE TABLE `tp_role` (
  `id` int(11) NOT NULL AUTO_INCREMENT COMMENT '角色ID',
  `name` varchar(50) NOT NULL COMMENT '角色名称',
  `description` varchar(255) DEFAULT NULL COMMENT '角色描述',
```

```
  `status` tinyint(1) DEFAULT 1 COMMENT '状态：1正常，0禁用',
  `create_time` int(11) DEFAULT NULL COMMENT '创建时间',
  `update_time` int(11) DEFAULT NULL COMMENT '更新时间',
  PRIMARY KEY (`id`),
  UNIQUE KEY `name` (`name`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='角色表';
```

3. 权限表（tp_permission）

```
CREATE TABLE `tp_permission` (
  `id` int(11) NOT NULL AUTO_INCREMENT COMMENT '权限ID',
  `name` varchar(50) NOT NULL COMMENT '权限名称',
  `path` varchar(100) NOT NULL COMMENT '权限标识（如：admin/user/index）',
  `parent_id` int(11) DEFAULT 0 COMMENT '父级权限ID',
  `type` tinyint(1) DEFAULT 1 COMMENT '类型：1菜单，2按钮',
  `status` tinyint(1) DEFAULT 1 COMMENT '状态：1正常，0禁用',
  `create_time` int(11) DEFAULT NULL COMMENT '创建时间',
  `update_time` int(11) DEFAULT NULL COMMENT '更新时间',
  PRIMARY KEY (`id`),
  UNIQUE KEY `path` (`path`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='权限表';
```

4. 用户 - 角色关联表（tp_user_role）

```
CREATE TABLE `tp_user_role` (
  `id` int(11) NOT NULL AUTO_INCREMENT COMMENT 'ID',
  `user_id` int(11) NOT NULL COMMENT '用户ID',
  `role_id` int(11) NOT NULL COMMENT '角色ID',
  PRIMARY KEY (`id`),
  UNIQUE KEY `user_role` (`user_id`,`role_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='用户-角色关联表';
```

5. 角色 - 权限关联表（tp_role_permission）

```
CREATE TABLE `tp_role_permission` (
  `id` int(11) NOT NULL AUTO_INCREMENT COMMENT 'ID',
```

```
  `role_id` int(11) NOT NULL COMMENT '角色ID',
  `permission_id` int(11) NOT NULL COMMENT '权限ID',
  PRIMARY KEY (`id`),
  UNIQUE KEY `role_permission` (`role_id`,`permission_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COMMENT='角色-权限关联表';
```

三、核心功能实现

1. 安装 ThinkPHP8

```
composer create-project topthink/think tp8-auth
```

2. 配置数据库

修改`.env`文件：

```
APP_DEBUG = true
[DATABASE]
TYPE = mysql
HOSTNAME = 127.0.0.1
DATABASE = tp8_auth
USERNAME = root
PASSWORD = root
HOSTPORT = 3306
CHARSET = utf8mb4
PREFIX = tp_
```

3. 创建模型

分别创建用户、角色、权限模型：

app/model/User.php

```
<?php
namespace app\model;
use think\Model;
use think\model\concern\SoftDelete;
class User extends Model
```

```
{
    // 关联角色
    public function roles()
    {
        return $this->belongsToMany(Role::class, 'user_role', 'role_id', 'user_id');
    }
}
```

app/model/Role.php

```
<?php
namespace app\model;
use think\Model;
class Role extends Model
{
    // 关联权限
    public function permissions()
    {
        return $this->belongsToMany(Permission::class, 'role_permission', 'permission_id', 'role_
id');
    }
}
```

app/model/Permission.php

```
<?php
namespace app\model;
use think\Model;
class Permission extends Model
{
    // 关联角色
    public function roles()
    {
        return $this->belongsToMany(Role::class, 'role_permission', 'role_id', 'permission_id');
    }
}
```

4. 实现 Auth 验证逻辑

创建 Auth 服务类，封装权限验证方法：

app/service/Auth.php

```
<?php
namespace app\service;
use app\model\User;
class Auth
{
    // 当前登录用户
    protected $user;
    // 初始化
    public function __construct($userId)
    {
        $this->user = User::find($userId);
    }
    // 检查用户是否有指定权限
    public function check($path)
    {
        if (!$this->user) return false;
        // 获取用户所有角色
        $roles = $this->user->roles;
        if (empty($roles)) return false;
        // 获取角色所有权限
        $permissions = [];
        foreach ($roles as $role) {
            $perms = $role->permissions()->column('path');
            $permissions = array_merge($permissions, $perms);
        }
        // 检查权限是否存在
        return in_array($path, $permissions);
    }
    // 获取用户所有权限
    public function getPermissions()
    {
        if (!$this->user) return [];
        $roles = $this->user->roles;
        $permissions = [];
        foreach ($roles as $role) {
            $perms = $role->permissions()->select()->toArray();
            $permissions = array_merge($permissions, $perms);
```

```
        }
        return $permissions;
    }
}
```

5. 实现权限中间件

创建中间件，拦截请求并验证权限：

app/middleware/AuthCheck.php

```
<?php
namespace app\middleware;
use app\service\Auth;
use think\exception\HttpResponseException;
use think\Response;
class AuthCheck
{
    public function handle($request, \Closure $next)
    {
        // 获取当前登录用户ID（假设从session获取）
        $userId = session('user_id');
        if (!$userId) {
            throw new HttpResponseException(Response::create(['code' => 401, 'msg' => '未登录'],
 'json'));
        }
        // 获取当前请求的权限标识（模块/控制器/方法）
        $path = strtolower($request->module() . '/' . $request->controller() . '/' . $request->action(
));
        // 验证权限
        $auth = new Auth($userId);
        if (!$auth->check($path)) {
            throw new HttpResponseException(Response::create(['code' => 403, 'msg' => '无权限访
问'], 'json'));
        }
        return $next($request);
    }
}
```

6. 注册中间件

修改`app/middleware.php`，注册权限中间件：

```
return [
    // 全局中间件
    // \app\middleware\AuthCheck::class,
    // 路由中间件
    'auth' => \app\middleware\AuthCheck::class,
];
```

7. 路由配置

在路由中使用权限中间件：

route/app.php

```
<?php
use think\facade\Route;
// 无需权限的路由
Route::post('login', 'Login/index');
// 需要权限的路由组
Route::group('admin', function () {
    Route::get('user/index', 'Admin/User/index');
    Route::get('role/index', 'Admin/Role/index');
    Route::get('permission/index', 'Admin/Permission/index');
})->middleware('auth');
```

8. 登录功能实现

创建登录控制器，验证用户并记录 session：

app/controller/Login.php

```
<?php
namespace app\controller;
use app\model\User;
use think\Controller;
```

```
use think\facade\Session;
class Login extends Controller
{
    public function index()
    {
        $username = $this->request->post('username');
        $password = $this->request->post('password');
        // 查找用户
        $user = User::where('username', $username)->find();
        if (!$user || !password_verify($password, $user->password)) {
            return json(['code' => 0, 'msg' => '用户名或密码错误']);
        }
        // 记录用户ID到session
        Session::set('user_id', $user->id);
        return json(['code' => 1, 'msg' => '登录成功']);
    }
}
```

四、扩展功能

1. 菜单生成

根据用户权限动态生成菜单：

```
// 在控制器中获取用户权限并传递到视图
$auth = new Auth(session('user_id'));
$permissions = $auth->getPermissions();
$this->assign('menus', $permissions);
```

2. 角色与权限管理

实现角色的增删改查、权限分配功能，例如：

```
// 给角色分配权限
$role = Role::find(1);
$role->permissions()->attach([1, 2, 3]); // 分配权限ID为1、2、3的权限
```

```
// 取消角色权限
$role->permissions()->detach([1]);
```

3. 用户角色分配

```
// 给用户分配角色
$user = User::find(1);
$user->roles()->attach([1]); // 分配角色ID为1的角色
```

五、总结

以上实现了 ThinkPHP8 中基于 RBAC 模型的权限管理系统核心功能，包括用户认证、角色管理、权限验证等。你可以根据实际需求扩展更多功能，如权限缓存、菜单排序、数据权限控制等。

> （注：文档部分内容可能由 AI 生成）