**大热篮球在线服务平台**

**源代码**

**（版本：V1.0）**

<?php

namespace app\frontend\controller;

use app\frontend\controller\Base;

use app\service\CampService;

use app\service\CoachService;

use app\service\ScheduleService;

use think\Db;

class Camp extends Base{

protected $CampService;

protected $CoachService;

public function \_initialize(){

parent::\_initialize();

$this->CampService = new CampService;

$this->CoachService = new CoachService;

}

public function index() {

// 最新消息

$member\_id = $this->memberInfo['id'];

$messageList = db('message')->page(1,10)->select();

$CampMember = new \app\model\CampMember;

$campList = $CampMember->where(['member\_id'=>$member\_id,'status'=>1])->select();

$this->assign('campList',$campList);

$this->assign('messageList',$messageList);

return view('Camp/index');

}

public function createCamp(){

$step = input('step', 1);

$view = 'Camp/createCamp'.$step;

$fast = input('fast', 0);

$campid = input('camp\_id');

$campS = new CampService();

$camp = $campS->getCampInfo($campid);

$this->assign('fast', $fast);

$this->assign('camp', $camp);

return view($view);

}

public function registerSuccess() {

return view('Camp/registerSuccess');

}

public function campInfo(){

$camp\_id = input('param.camp\_id');

$lessonList = db('lesson')->where(['camp\_id'=>$camp\_id,'status'=>1])->select();

$lessonCount = count($lessonList);

$commentList = $this->CampService->getCampCommentListByPage(['camp\_id'=>$camp\_id]);

$campInfo = $this->CampService->getCampInfo($camp\_id);

// 查询是否跟训练营有关系

$isPower = $this->CampService->isPower($camp\_id,$this->memberInfo['id']);

$this->assign('isPower',$isPower);

$this->assign('commentList',$commentList['data']);

$this->assign('lessonCount',$lessonCount);

$this->assign('lessonList',$lessonList);

$this->assign('campInfo',$campInfo);

return view('Camp/campInfo');

}

public function campList(){

// $map = input();

// $campList = $this->CampService->campListPage($map);

$campList = [];

$this->assign('campList',$campList);

return view('Camp/campList');

}

public function searchCamp(){

$keyword = input('keyword');

$province = input('province');

$city = input('city');

$area = input('area');

$map = ['province'=>$province,'city'=>$city,'area'=>$area];

foreach ($map as $key => $value) {

if($value == ''){

unset($map[$key]);

}

}

if($keyword){

$map['camp'] = ['LIKE',$keyword];

}

$campList = $this->CampService->getCampList($map);

$this->assign('campList',$campList);

return view('Camp/searchCamp');

}

public function searchCampApi(){

try{

$keyword = input('keyword');

$province = input('province');

$city = input('city');

$area = input('area');

$map = ['province'=>$province,'city'=>$city,'area'=>$area];

foreach ($map as $key => $value) {

if($value == ''){

unset($map[$key]);

}

}

if($keyword){

$map['camp'] = ['LIKE',$keyword];

}

$campList = $this->CampService->campListPage($map);

return json(['code'=>100,'msg'=>'OK','data'=>$campList]);

}catch(Exception $e){

return json(['code'=>200,'msg'=>$e->getMessage()]);

}

}

// 邀请学生入驻

public function inviteStudent(){

$data = input('param.');

$data['member'] = $this->memberInfo['member'];

$data['member\_id'] = $this->memberInfo['id'];

$data['type'] = 1;

$data['status'] = 1;

$is\_join = db('camp\_member');

return view('Camp/inviteStudent');

}

// 邀请教练入驻

public function inviteCoach(){

return view();

}

// 学生的训练营

public function campListOfStudent(){

$member\_id = $this->memberInfo['id'];

$actCampList = Db::view('camp\_member','camp\_id,member\_id')

->view('camp','camp,act\_member,finished\_lessons,star,province,city,area,logo','camp.id=camp\_member.camp\_id')

->where(['camp\_member.member\_id'=>$member\_id,'camp\_member.type'=>1,'camp\_member.status'=>1])

->order('camp\_member.id desc')

->select();

$restCampList = Db::view('camp\_member','camp\_id')

->view('camp','camp,act\_member,finished\_lessons,star,province,city,area,logo','camp.id=camp\_member.camp\_id')

->where(['camp\_member.member\_id'=>$member\_id,'camp\_member.type'=>1,'camp\_member.status'=>2])

->order('camp\_member.id desc')

->select();

$this->assign('actCampList',$actCampList);

$this->assign('restCampList',$restCampList);

return view('Camp/campListOfStudent');

}

// 教练身份的训练营

public function campListOfCoach(){

$coach\_id = input('param.coach\_id');

if(!$coach\_id){

$member\_id = $this->memberInfo['id'];

$coachInfo = db('coach')->where(['member\_id'=>$member\_id])->find();

$this->assign('coachInfo',$coachInfo);

$coach\_id = $coachInfo['id'];

}else{

$coachInfo = db('coach')->where(['id'=>$coach\_id])->find();

$member\_id = $coachInfo['member\_id'];

}

$campList = Db::view('camp\_member','camp\_id,member\_id,type,status')

->view('camp','camp,act\_member,finished\_lessons,star,province,city,area,logo,id','camp.id=camp\_member.camp\_id')

->where(['camp\_member.member\_id'=>$member\_id,'camp\_member.type'=>2,'camp\_member.status'=>1])

->order('camp\_member.id desc')

->select();

$this->assign('campList',$campList);

return view('Camp/campListOfCoach');

}

// 申请列表

public function applyListOfCoach(){

$camp\_id = input('param.camp\_id');

$applyListOfCoach = Db::view('camp\_member','member\_id,remarks')

->view('coach','star,coach,coach\_level,lesson\_flow,portraits','coach.member\_id=camp\_member.member\_id')

->view('member','sex,birthday','coach.member\_id=member.id')

->where(['camp\_member.camp\_id'=>$camp\_id,'camp\_member.type'=>2,'camp\_member.status'=>0])

->order('camp\_member.id desc')

->select();

// 计算年龄

foreach ($applyListOfCoach as $key => $value) {

$applyListOfCoach[$key]['age'] = ceil(( time() - strtotime($value['birthday']))/31536000) ;

}

$count = count($applyListOfCoach);

$this->assign('count',$count);

// dump($applyListOfCoach);die;

$this->assign('applyListOfCoach',$applyListOfCoach);

return view('Camp/applyListOfCoach');

}

public function studentInfo(){

return view('Camp/studentInfo');

}

// 没啥权限的campInfo菜单

public function indexCamp(){

return view('Camp/indexCamp');

}

// 管理员的camp菜单

public function powerCamp(){

$camp\_id = input('param.camp\_id');

$member\_id = $this->memberInfo['id'];

$power = $this->CampService->isPower($camp\_id,$member\_id);

$campInfo = $this->CampService->getCampInfo($camp\_id);

$gradeCount = db('grade')->where(['camp\_id'=>$camp\_id])->count();

$scheduleCount = db('schedule')->where(['camp\_id'=>$camp\_id])->count();

$lessonCount = db('lesson')->where(['camp\_id'=>$camp\_id])->count();

$this->assign('power',$power);

$this->assign('gradeCount',$gradeCount);

$this->assign('scheduleCount',$scheduleCount);

$this->assign('lessonCount',$lessonCount);

$this->assign('campInfo',$campInfo);

return view('Camp/powerCamp');

}

// 非管理员的camp菜单

public function clientOfcamp(){

$camp\_id = input('param.camp\_id');

$campInfo = $this->CampService->getCampInfo($camp\_id);

$GradeMember = new \app\model\GradeMember;

$objStudentList = $GradeMember->where(['member\_id'=>$this->memberInfo['id'],'camp\_id'=>$camp\_id])->select();

if($objStudentList){

$studentList = $objStudentList->toArray();

}else{

$studentList = [];

}

$Grade = new \app\model\Grade;

$leaderList = $Grade->where(['camp\_id'=>$camp\_id,'leader\_id'=>$this->memberInfo['id']])->select();

$teacherList = $Grade->where(['camp\_id'=>$camp\_id,'teacher\_id'=>$this->memberInfo['id']])->select();

$this->assign('teacherList',$teacherList);

$this->assign('leaderList',$leaderList);

$this->assign('studentList',$studentList);

$this->assign('campInfo',$campInfo);

return view('Camp/clientOfCamp');

}

// 教练的camp菜单

public function coachCamp(){

$camp\_id = input('param.camp\_id');

$member\_id = $this->memberInfo['id'];

$is\_power = $this->CampService->isPower($camp\_id,$member\_id);

if($is\_power == 0){

$this->error('您没有权限');

}

$campInfo = $this->CampService->getCampInfo($camp\_id);

$gradeCount = db('grade')->where(['camp\_id'=>$camp\_id])->count();

$scheduleCount = db('schedule')->where(['camp\_id'=>$camp\_id])->count();

$lessonCount = db('lesson')->where(['camp\_id'=>$camp\_id])->count();

$this->assign('gradeCount',$gradeCount);

$this->assign('scheduleCount',$scheduleCount);

$this->assign('lessonCount',$lessonCount);

$this->assign('campInfo',$campInfo);

return view('Camp/coachCamp');

}

/\* public function coachListOfCamp(){

$camp\_id = input('param.camp\_id');

$campInfo = $this->CampService->getCampInfo($camp\_id);

$type = input('param.type')?input('param.type'):2;

$status = input('param.status')?input('param.status'):1;

$map = ['camp\_member.camp\_id'=>$camp\_id,'camp\_member.type'=>$type,'camp\_member.status'=>$status];

$coachList = $this->CoachService->getCoachListOfCamp($map);

$this->assign('campInfo',$campInfo);

$this->assign('coachList',$coachList);

return view('Camp/coachListOfCamp');

}\*/

// 训练营设置

public function campSetting(){

$step = input('param.step', 1);

$view = 'Camp/campSetting'.$step;

$camp\_id = input('param.camp\_id');

$campInfo = $this->CampService->getCampInfo($camp\_id);

$is\_power = $this->CampService->isPower($camp\_id,$this->memberInfo['id']);

if($is\_power < 4){

$this->error('您没有权限');

}

// 营业执照

$campS = new CampService();

$campcert = $campS->getCampCert($camp\_id);

$this->assign('campInfo',$campInfo);

$this->assign('campcert', $campcert);

return view($view);

}

// 训练营设置成功

public function editsuccess() {

$edit = input('param.edit', 0);

$this->assign('edit', $edit);

return view('Camp/editsuccess');

}

// 训练营教练列表

public function coachListOfCamp() {

$camp\_id = input('camp\_id');

$status = input('status', 1);

if ($camp\_id) {

$map['camp\_member.camp\_id'] = $camp\_id;

} else {

$campS = new CampService();

$owncamp = $campS->getOnecamp(['member\_id' => $this->memberInfo['id']]);

// dump($owncamp);

if ($owncamp && $owncamp['check\_status']) {

$map['camp\_member.camp\_id'] = $owncamp['id'];

}

}

$map['camp\_member.status'] = $status;

$map['camp\_member.type'] = ['egt', 2];

$list= Db::view('camp\_member','camp\_id')

->view('coach','\*','coach.member\_id=camp\_member.member\_id')

->where($map)

->order('camp\_member.id desc')

->select();

// dump($list);

$this->assign('coachlist', $list);

$this->assign('camp\_id', $camp\_id);

$view = $status ? 'Camp/coachListOfCamp' : 'Camp/coachapplylist';

return view($view);

}

// 训练营教练加入申请

public function coachapply() {

$coach\_id = input('param.coach\_id');

$coachS = new CoachService();

if($coach\_id){

$coachInfo = $coachS->getCoachInfo(['id'=>$coach\_id]);

}

// 申请留言

$campmember = db('camp\_member')->where(['member\_id' => $coachInfo['member\_id'], 'camp\_id' => input('camp\_id'), 'type' => 2])->find();

// 全部班级

$gradeList = db('grade')->where(['coach\_id'=>$this->memberInfo['id']])->column('id');

$gradeCount = count($gradeList);

// 全部学员

$studentCount = db('grade\_member')->distinct(true)->field('member\_id')->where(['grade\_id'=>['in',$gradeList],'type'=>1,'status'=>1])->count();

// 执教过多少课时

$scheduleS = new ScheduleService();

$scheduleCount =$scheduleS->countSchedules(['coach\_id'=>$coachInfo['id'],'status'=>1]);

//教练评论

$commentList = db('coach\_comment')->where(['coach\_id'=>$coach\_id])->select();

//所属训练营

$campList = Db::view('camp\_member','camp\_id')

->view('camp','logo','camp.id=camp\_member.camp\_id')

->where(['camp\_member.member\_id'=>$coachInfo['member\_id'], 'camp\_member.type'=> 2,'camp\_member.status'=>1])

->order('camp\_member.id desc')

->select();

$this->assign('campmember', $campmember);

$this->assign('campList',$campList);

$this->assign('commentList',$commentList);

$this->assign('scheduleCount',$scheduleCount);

$this->assign('studentCount',$studentCount);

$this->assign('gradeCount',$gradeCount);

$this->assign('coachInfo',$coachInfo);

return view('Camp/coachapply');

}

// 训练营下教练档案

public function coachInfoOfcamp(){

$member\_id = input('param.member\_id')?input('param.member\_id'):$this->memberInfo['id'];

$coach\_id = input('param.coach\_id');

$camp\_id = input('param.camp\_id');

$coachS = new CoachService();

if(!$coach\_id){

// 获取教练档案

$coachInfo = $coachS->getCoachInfo(['member\_id'=>$member\_id]);

$coach\_id = $coachInfo['id'];

}else{

$coachInfo =$coachS->getCoachInfo(['id'=>$coach\_id]);

}

//教练的班级

$GradeService = new \app\service\GradeService;

$gradeOfCoachList = $GradeService->getGradeList(['coach\_id'=>$member\_id,'camp\_id'=>$camp\_id]);

// 教练的证件

$cert = db('cert')->where(['member\_id'=>$member\_id])->select();

$identCert = [];

$coachCert = [];

foreach ($cert as $key => $value) {

if($value['cert\_type'] == 1){

$identCert = $value;

}

if($value['cert\_type'] == 3){

$coachCert = $value;

}

}

// dump($identCert);die;

if(empty($identCert)){

$identCert['cert\_no'] = '未认证';

}

if(empty($coachCert)){

$coachCert = ['photo\_positive'=>'/static/frontend/images/uploadDefault.jpg','photo\_back'=>'/static/frontend/images/uploadDefault.jpg'];

}

//获取教练的课量

$m = input('m')?input('m'):date('m');

$y = input('y')?input('y'):date('Y');

$begin\_m = mktime(0,0,0,$m,1,$y);

$end\_m = mktime(23,59,59,$m,30,$y)-1;

$begin\_y = mktime(0,0,0,1,1,$y);

$end\_y = mktime(23,59,59,12,30,$y)-1;

$scheduleS = new ScheduleService();

$monthScheduleOfCoachList = $scheduleS->getScheduleList([

'coach\_id'=>$member\_id,

// 'type'=>1,

'camp\_id'=>$camp\_id,

'create\_time'=>['BETWEEN',[$begin\_m,$end\_m]]

]);

$monthScheduleOfCoach = count($monthScheduleOfCoachList);

$yearScheduleOfCoachList = $scheduleS->getScheduleList([

'coach\_id'=>$member\_id,

// 'type'=>1,

'camp\_id'=>$camp\_id,

'create\_time'=>['BETWEEN',[$begin\_y,$end\_y]]

]);

$yearScheduleOfCoach = count($yearScheduleOfCoachList);

//教练工资

$SalaryInService = new \app\service\SalaryInService($member\_id);

$salaryList = $SalaryInService->getSalaryInList(['member\_id'=>$member\_id,'type'=>1,'camp\_id'=>$camp\_id]);

// 平均月薪

$averageSalaryByMonth = $SalaryInService->getAverageSalaryByMonth($member\_id,$camp\_id);

// 平均年薪

$averageSalaryByYear = $SalaryInService->getAverageSalaryByYear($member\_id,$camp\_id);

// dump($salaryList);die;

$this->assign('monthScheduleOfCoachList',$monthScheduleOfCoachList);//教练当月的课量

$this->assign('monthScheduleOfCoach',$monthScheduleOfCoach);//当月课量数量

$this->assign('gradeOfCoachList',$gradeOfCoachList);//教练班级

$this->assign('yearScheduleOfCoachList',$yearScheduleOfCoachList);//当年课量

$this->assign('yearScheduleOfCoach',$yearScheduleOfCoach);//当年课量数量

$this->assign('y',$y);

$this->assign('m',$m);

$this->assign('salaryList',$salaryList);

$this->assign('coachInfo',$coachInfo);

$this->assign('identCert',$identCert);

$this->assign('coachCert',$coachCert);

$this->assign('averageSalaryByMonth',$averageSalaryByMonth);

$this->assign('averageSalaryByYear',$averageSalaryByYear);

return view('Camp/coachInfoOfCamp');

}

// 训练营的场地列表

public function courtListOfCamp(){

$camp\_id = input('param.camp\_id')?input('param.camp\_id'):0;

// $CourtService = new \app\service\CourtService;

// $courtList = $CourtService->getCourtList(['camp\_id'=>$camp\_id,'status'=>1]);

// $campInfo = $this->CampService->getCampInfo(['id'=>$camp\_id]);

// dump($courtList);die;

// $this->assign('campInfo',$campInfo);

// $this->assign('courtList',$courtList);

$this->assign('camp\_id',$camp\_id);

return view('Camp/courtListOfCamp');

}

// 训练营-教务人员列表

public function teachlistofcamp() {

$camp\_id = input('camp\_id', 0);

$status = input('status', 1);

$this->assign('camp\_id',$camp\_id);

$view = $status ? 'Camp/teachlistOfCamp' : 'Camp/teachapplylist';

return view($view);

}

}

类文件：use app\frontend\controller\Base;

<?php

namespace app\frontend\controller;

use think\Controller;

use app\service\SystemService;

use app\service\MemberService;

use think\Cookie;

use think\Request;

use app\service\WechatService;

class Base extends Controller{

public $systemSetting;

public $memberInfo;

public function \_initialize(){

// 从模板消息url进入 带有openid字段 保存会员登录信息

if ( input('?param.openid') ) {

$memberS = new MemberService();

$member = $memberS->getMemberInfo(['openid' => input('param.openid')]);

if ($member) {

cookie('mid', $member['id']);

cookie('member', md5($member['id'].$member['member'].config('salekey')) );

session('memberInfo', $member, 'think');

}

}

$url = cookie('url');

if(!$url){

$url = $\_SERVER["REQUEST\_URI"];

cookie('url', $url);

}

$pid = input('param.pid');

if($pid){

cookie('pid',$pid);

}else{

cookie('pid',0);

}

$this->systemSetting = SystemService::getSite();

$this->assign('systemSetting',$this->systemSetting);

$this->footMenu();

if ( !Cookie::has('mid') ) {

$this->nologin();

}

$this->memberInfo = session('memberInfo', '', 'think');

$this->assign('memberInfo', $this->memberInfo);

$fasturl = $this->is\_weixin() ? url('login/fastRegister') : url('login/login'); //提示完善信息对话框链接

$this->assign('fasturl', $fasturl);

$this->assign('mid', cookie('mid'));

}

protected function footMenu(){

define('CONTROLLER\_NAME',Request::instance()->controller());

define('MODULE\_NAME',Request::instance()->module());

define('ACTION\_NAME',Request::instance()->action());

$footMenu = [

[

'name'=>'首页',

'icon'=>'icon iconfont icon-m-Home',

'action'=>'index',

'controller'=>'Index'

],

[

'name'=>'消息',

'icon'=>'icon iconfont icon-m-news',

'action'=>'index',

'controller'=>'Message'

],

[

'name'=>'发现',

'icon'=>'icon iconfont icon-m-Find',

'action'=>'index',

'controller'=>'Find'

],

[

'name'=>'训练营',

'icon'=>'icon iconfont icon-m-TrainingCamp2',

'action'=>'index',

'controller'=>'Camp'

],

[

'name'=>'我的',

'icon'=>'icon iconfont icon-m-mine',

'action'=>'index',

'controller'=>'Member'

],

];

$this->assign('footMenu',$footMenu);

}

protected function nologin(){

// $this->redirect('Test/index');

/\*$result = $this->is\_weixin();

\* if($result){

$WechatService = new WechatService;

$callback = url('login/wxlogin','','', true);

$this->redirect($WechatService -> oauthredirect($callback));

}else{

$memberInfo = ['id'=>-1,'member'=>'游客','hp'=>0,'level'=>0,'avatar'=>'/static/default/avatar.png','nickname'=>'游客'];

unset($memberInfo['password']);

$cookie = md5($memberInfo['id'].$memberInfo['member'].'hot');

cookie('member',md5($memberInfo['id'].$memberInfo['member'].'hot'));

$re = session('memberInfo',$memberInfo , 'think');

$url = cookie('url');

if($re){

$this->redirect($url);

}else{

$this->redirect('Index/index');

}

}\*/

if ($this->is\_weixin()) {

$wechatS = new WechatService();

$callback = url('frontend/login/wxlogin', '', '', true);

$this->redirect( $wechatS->oauthredirect($callback) );

} else {

//echo 'other';

Cookie::set('mid', 0);

$member = [

'id' => 0,

'member' => '游客',

'nickname' => '游客',

'avatar' => '/static/default/avatar.png',

'hp' => 0,

'level' => 0,

'telephone' =>'',

'email' =>'',

'realname' =>'',

'province' =>'',

'city' =>'',

'area' =>'',

'location' =>'',

'sex' =>0,

'height' =>0,

'weight' =>0,

'charater' =>'',

'shoe\_code' =>0,

'birthday' =>'0000-00-00',

'create\_time'=>0,

'pid' =>0,

'hp' =>0,

'cert\_id' =>0,

'score' =>0,

'flow' =>0,

'balance' =>0,

'remarks' =>0,

'hot\_id'=>00000000,

];

cookie('mid', 0);

cookie('member', md5($member['id'].$member['member'].config('salekey')) );

session('memberInfo', $member, 'think');

if (session('memberInfo', '', 'think')) {

$url = cookie('url');

cookie('url',null);

$this->redirect( $url );

} else {

$this->redirect('frontend/Index/index');

}

}

}

// 判断是否是微信浏览器

function is\_weixin() {

if (strpos($\_SERVER['HTTP\_USER\_AGENT'], 'MicroMessenger') !== false) {

// 微信

return true;

} else {

return false;

}

}

}

类文件：use app\service\CampService;

<?php

namespace app\service;

use app\model\Camp;

use app\common\validate\CampVal;

use think\Db;

use app\common\validate\CampCommentVal;

use app\model\CampMember;

class CampService {

public $Camp;

public function \_\_construct()

{

$this->Camp = new Camp();

}

public function getCampList($map=[],$page = 1,$paginate = 10,$order='') {

$res = $this->Camp->where($map)->where(['status'=>1])->order($order)->page($page,$paginate)->select();

if($res){

$result = $res->toArray();

return $result;

}else{

return $res;

}

}

public function getCampListByPage( $map=[],$paginate=10, $order=''){

$res = $this->Camp->where($map)->where(['status'=>1])->order($order)->paginate($paginate);

if($res){

$result = $res->toArray();

return $result;

}else{

return $res;

}

}

/\*\*

\* 读取资源

\*/

public function getCampInfo($id) {

$res = $this->Camp->get($id);

if (!$res) {

return false;

}

return $res->toArray();

}

public function getOneCamp($map) {

$res = Camp::get($map);

if (!$res) {

return false;

}

$result = $res->toArray();

$result['check\_status'] = $res->getData('status');

return $result;

}

/\*\*

\* 更新资源

\*/

public function updateCamp($request) {

$model = new Camp();

$result = $model->validate('CampVal.edit')->isUpdate(true)->save($request);

if (false === $result) {

return ['code' => 100, 'msg' => $model->getError()];

} else {

return ['code' => 200, 'msg' => \_\_lang('MSG\_200'), 'data' => $request['id']];

}

}

public function SoftDeleteCamp($id) {

$res = $this->Camp->destroy($id);

return $res;

}

/\*\*

\* 创建资源

\*/

public function createCamp($request){

// 一个人只能创建一个训练营

if ( $this->hasCreateCamp($request['member\_id']) ){

return [ 'msg' => '一个会员只能创建一个训练营', 'code' => 100 ];

}

$model = new Camp;

$result = $model->validate('CampVal.add')->save($request);

if ( false === $result ) {

return ['code' => 100, 'msg' => $model->getError()];

}

// 保存camp\_member 关联记录

//dump($model->getLastInsID());

$lastInsId = $model->getLastInsID();

$campMemberDb = Db::name('camp\_member');

$campMemberData = [

'camp' => $request['camp'],

'camp\_id' => $lastInsId,

'member' => $request['realname'],

'member\_id' => $request['member\_id'],

'remarks' => '创建训练营',

'type' => 4,

'status' => 1,

'create\_time' => time(),

'update\_time' => time()

];

$addCampMember = $campMemberDb->insert($campMemberData);

if (!$addCampMember) {

Camp::destroy($lastInsId);

return ['msg' => \_\_lang('MSG\_403'), 'code' => 100];

}

return [ 'code' => 200, 'msg' => \_\_lang('MSG\_200'), 'data' => $lastInsId ];

}

/\*\*

\* 判断是否已创建训练营

\*/

public function hasCreateCamp($member\_id){

$res = Camp::get(['member\_id' => $member\_id]);

if ($res) {

return $res->toArray();

} else {

return false;

}

}

/\*\*

\* 返回权限

\*/

public function isPower($camp\_id,$member\_id){

$is\_power = db('camp\_member')

->where(['member\_id'=>$member\_id,'camp\_id'=>$camp\_id,'status'=>1])

// ->where(function ($query) {

// $query->where('type', 2)->whereor('type', 3)->whereor('type',4);})

->value('type');

return $is\_power?$is\_power:0;

}

// 获取训练营资质证明

public function getCampCert($campid) {

$certlist = db('cert')->where(['camp\_id' => $campid])->select();

$campCert = [

'cert' => '',

'fr' => ['cert\_no' => '', 'photo\_positive' => ''],

'cjz' => ['cert\_no' => '', 'photo\_positive' => ''],

'other' => ''

];

if ($certlist) {

foreach ($certlist as $val) {

switch ( $val['cert\_type'] ) {

case 1: {

if ($val['member\_id']) {

$campCert['cjz']['cert\_no'] = $val['cert\_no'];

$campCert['cjz']['photo\_positive'] = $val['photo\_positive'];

} else {

$campCert['fr']['cert\_no'] = $val['cert\_no'];

$campCert['fr']['photo\_positive'] = $val['photo\_positive'];

}

break;

}

case 4: {

$campCert['cert'] = $val['photo\_positive'];

break;

}

default: {

$campCert['other'] = $val['photo\_positive'];

}

}

}

}

return $campCert;

}

// 获取训练营评论列表

public function getCampCommentListByPage($map,$paginate = 10){

$CampCommentModel = new \app\model\CampComment;

$result = $CampCommentModel->where($map)->paginate($paginate);

if($result){

return $result->toArray();

}else{

return $result;

}

}

// 评论训练营

public function createCampComment($data){

$CampCommentModel = new \app\model\CampComment;

$validate = validate('CampCommentVal');

if(!$validate->check($data)){

return ['msg' => $validate->getError(), 'code' => 200];

}

$result = $CampCommentModel->save($data);

if($result){

return ['msg' => "评论成功", 'code' => 200];

}else{

return ['msg' =>"评论失败", 'code' => 100];

}

}

public function getCampMemberInfo($map){

$CampMemberModel = new CampMember;

$isMember = $this->CampService->where($map)->find();

return $isMember;

}

}

类文件：use app\service\CoachService;

<?php

namespace app\service;

use app\model\Coach;

use app\common\validate\CoachVal;

use app\model\Grade;

use app\model\GradeMember;

use think\Db;

class CoachService{

private $CoachModel;

public function \_\_construct(){

$this->CoachModel = new Coach();

$this->gradeMemberModel = new GradeMember;

}

/\*\*

\* 查询教练信息&&关联member表

\*/

public function coachInfo($map){

$res = Coach::with('member')->where($map)->find();

if($res){

$result = $res->toArray();

$result['status\_num'] = $res->getData('status');

return $result;

}else{

return $res;

}

}

/\*\*

\* 申请成为教练

\*/

public function createCoach($request){

$model = new Coach();

$result = $model->validate('CoachVal')->save($request);

if ($result === false) {

return ['code' => 100, 'msg' => $model->getError()];

}else{

return ['code'=>200,'msg'=>\_\_lang('MSG\_200'),'data'=> $model->getLastInsID() ];

}

}

/\*\*

\* 教练更改资料

\*/

public function updateCoach($request,$id)

{

$model = new Coach();

$result = $model->validate('CoachVal')->save($request, ['id' => $id]);

if ($result === false) {

return ['code' => 100, 'msg' => $model->getError()];

} else {

return ['code' => 200, 'msg' => \_\_lang('MSG\_200')];

}

}

// 教练列表 分页

public function getCoachListByPage( $map=[],$order='',$paginate = 10) {

$res = Coach::with('member')->where($map)->order($order)->paginate($paginate);

if($res){

$result = $res->toArray();

}else{

$res;

}

}

public function updateCoachStatus($request) {

$result = Coach::update($request);

if (!$result) {

return [ 'msg' => \_\_lang('MSG\_400'), 'code' => 100 ];

}else{

return ['msg' => \_\_lang('MSG\_200'), 'code' => 200, 'data' => $result];

}

}

public function SoftDeleteCamp($id) {

$result = Coach::destroy($id);

if (!$result) {

return [ 'msg' => \_\_lang('MSG\_400'), 'code' => 100 ];

} else {

return ['msg' => \_\_lang('MSG\_200'), 'code' => 200, 'data' => $result];

}

}

// 教练列表 分页

public function getCoachList($map=[],$page=1, $paginate = 10, $order='') {

$result = $this->CoachModel->where($map)->order($order)->page($page,$paginate)->select();

if($result){

$res = $result->toArray();

return $res;

}else{

return $result;

}

}

// 教练列表 分页

public function getCoachListOfCamp($map=[],$page = 1, $paginate = 10, $order='camp\_member.id desc') {

$result = Db::view('camp\_member','member\_id,type')

->view('coach','\*','camp\_member.member\_id=coach.member\_id')

->where($map)

->order($order)

->page($page,$paginate)

->select();

return $result;

}

// 教练列表 分页

public function getCoachListOfCampByPage($map=[], $order='camp\_member.id desc', $paginate = 10) {

$result = Db::view('camp\_member','member\_id,type')

->view('coach','\*','camp\_member.member\_id=coach.member\_id')

->where($map)

->order($order)

->paginate($paginate);

return $result;

}

public function getCoachInfo($map){

$res = $this->CoachModel->with('member')->where($map)->find();

if($res){

$result = $res->toArray();

return $result;

}else{

return $res;

}

}

// 教练所在在线班级列表

public function onlinegradelist($coach\_id, $camp\_id) {

$model = new Grade();

$res = $model->where(['camp\_id' => $camp\_id,'coach\_id' => $coach\_id, 'status' => 1])->select();

if ($res) {

return $res->toArray();

} else {

return $res;

}

}

}

类文件：use app\service\ScheduleService;

<?php

namespace app\service;

use app\model\Student;

use app\model\GradeMember;

use app\model\ScheduleMember;

use think\Db;

use app\common\validate\StudentVal;

class StudentService{

private $studentModel;

private $gradeMemberModel;

private $scheduleMemberModel;

public function \_\_construct(){

$this->studentModel = new Student;

$this->gradeMemberModel = new GradeMember;

$this->scheduleMemberModel = new ScheduleMember;

}

public function getStudentInfo($map){

$res = $this->studentModel->where($map)->find();

if($res){

$result = $res->toArray();

return $result;

}else{

return $res;

}

}

public function createStudent($data){

$validate = validate('StudentVal');

if(!$validate->check($data)){

return ['msg' => $validate->getError(), 'code' => 100];

}

$result = $this->studentModel->data($data)->save();

if($result){

return ['code'=>200,'msg'=>'ok','data'=>$result];

}else{

return ['code'=>100,'msg'=>$this->studentModel->getError()];

}

}

public function updateStudent($data,$id){

$validate = validate('CampVal');

if(!$validate->check($StudentVal)){

return ['msg' => $validate->getError(), 'code' => 100];

}

$result = $this->studentModel->save($data,['id'=>$id]);

if($result){

return ['code'=>200,'msg'=>'ok','data'=>$result];

}else{

return ['code'=>100,'msg'=>$this->studentModel->getError()];

}

}

/\*\*

\* 购买课程

\*/

public function buyLesson($request,$id = false){

//是否完善资料

$is\_student = $this->studentModel->where(['member\_id'=>$this->memberInfo['id']])->find();

if(!$is\_student){

return false;

}

if($id){

$result = $this->gradeMemberModel->save($request,['id'=>$id]);

}else{

$result = $this->gradeMemberModel->save($request);

}

if($result){

return true;

}else{

return false;

}

}

// 获取一个教练下的所有学生

public function getStudentListOfCoach($map,$page = 1,$paginate = 10){

$result = $this->gradeMemberModel->where($map)->page($page,$paginate)->select();

if($result){

$res = $result->toArray();return $res;

}

return $result;

}

// 获取学生班级?

public function getStudentGradeList($map,$page = 1,$paginate = 10){

$result = $this->gradeMemberModel->where($map)->page($page,$paginate)->select();

if($result){

$res = $result->toArray();return $res;

}

return $result;

}

// 获取课时学生列表

public function getStudentScheduleList($map,$page = 1,$paginate = 10){

$result = db('schedule\_member')->where($map)->page($page,$paginate)->select();

if($result){

$res = $result->toArray();return $res;

}

return $result;

}

// 获取学生?

public function getStudentListOfCamp($map,$page = 1,$paginate = 10){

$result = $this->gradeMemberModel->where($map)->page($page,$paginate)->select();

if($result){

$res = $result->toArray();

return $res;

}

return $result;

}

// 获取用户学生列表

public function getStudentList($map,$page = 1,$paginate = 10){

$result = $this->studentModel->where($map)->page($page,$paginate)->select();

if($result){

$res = $result->toArray();

return $res;

}

return $result;

}

// 批量更新学生数据

public function saveAllStudent($students,$grade\_id,$grade){

//重组上课学员

foreach ($students as $key => $value) {

$students[$key]['grade\_id'] = $grade\_id;

$students[$key]['grade'] = $grade;

}

// dump($students);die;

$result = $this->gradeMemberModel->saveAll($students);

if($result){

return ['code'=>200,'msg'=>\_\_lang('MSG\_200'),'data'=>$result];

}else{

return ['code'=>100,'msg'=>$this->gradeMemberModel->getError()];

}

}

// 班级学生变动

public function updateGradeMember($data,$id){

$result = $this->gradeMemberModel->save($data,['id'=>$id]);

if($result){

return ['code'=>200,'msg'=>'ok','data'=>$result];

}else{

return ['code'=>100,'msg'=>$this->gradeMemberModel->getError()];

}

}

// 已上课程+1

public function setIncFinishedTotal($num = 1,$student\_id){

$this->studentModel->where(['id'=>$student\_id])->setInc('finished\_total',$num);

}

public function getStudentListByPage($map=[], $order='', $paginate=10) {

$result = $this->studentModel->where($map)->order($order)->paginate($paginate);

if($res){

$res = $result->toArray();

return $res;

}else{

return $result;

}

}

}

类文件use app\service\SystemService;

<?php

// 系统service

namespace app\service;

use think\Lang;

class SystemService {

// 2017/7/18 获取站点数据

public static function getSite() {

return db('setting')->where('id', 1)->find();

}

// 2017/7/18 设置站点数据

public static function setSite($posts) {

$id = $posts['id'];

$data = [

'sitename' => $posts['sitename'],

'title' => $posts['title'],

'keywords' => $posts['keywords'],

'description' => $posts['description'],

'footer' => $posts['footer'],

'wxappid' => $posts['wxappid'],

'wxsecret' => $posts['wxsecret'],

'level1' => $posts['level1'],

'level2' => $posts['level2'],

'vip1' => $posts['vip1'],

'vip2' => $posts['vip2'],

'vip3' => $posts['vip3'],

'vip4' => $posts['vip4'],

'vip5' => $posts['vip5'],

'lrss' => $posts['lrss'],

'lrcs' => $posts['lrcs']

];

$result = db('setting')->where('id', $id)->update($data);

return $result;

}

// 获取公众号信息

public static function getWxApp() {

$res = db('setting')->where('id', 1)->field(['wxappid', 'wxsecret'])->find();

if (!$res)

return false;

return $res;

}

}

类文件use app\service\MemberService;

<?php

namespace app\service;

use app\model\Camp;

use app\model\Coach;

use app\model\Member;

use app\common\validate\MemberVal;

class MemberService{

private $memberModel;

public function \_\_construct(){

$this->memberModel = new Member;

}

// 获取会员

public function getMemberInfo($map){

$result = $this->memberModel->where($map)->find();

if($result){

$res = $result->toArray();

return $res;

}

return $result;

}

//获取资源列表

public function getMemberList($map){

$result = $this->memberModel->where($map)->select();

if($result){

$res = $result->toArray();

return $res;

}

return $result;

}

//修改会员资料

public function updateMemberInfo($request,$id){

//验证规则

$validate = validate('MemberVal');

if(!$validate->scene('edit')->check($request)){

return ['msg'=>$validate->getError(),'code'=>100];

}

$result = $this->memberModel->allowField(true)->save($request,['id'=>$id]);

if($result ===false){

return ['msg'=>$this->memberModel->getError(),'code'=>100];

}else{

return ['msg'=>\_\_lang('MSG\_200'),'code'=>200,'data'=>$result];

}

}

//新建会员

public function saveMemberInfo($request){

// 生成一个随机id

$hot\_id = $this->getHotID();

$MemberModel = new Member();

$request['password'] = passwd($request['password']);

$request['repassword'] = passwd($request['repassword']);

$request['hot\_id'] = $hot\_id;

if (!isset($request['avatar'])) {

$request['avatar'] = '/static/default/avatar.png';

}

//验证规则

$res = $MemberModel->validate('MemberVal.add')->allowField(true)->save($request);

if ($res === false) {

//dump($MemberModel->getError());

return [ 'code' => 100, 'msg' => $MemberModel->getError() ];

} else {

$login = $this->saveLogin($MemberModel->id);

if ($login) {

return ['code' => 200, 'msg' => \_\_lang('MSG\_200')];

} else {

return ['code' => 100, 'msg' => '请重新登陆'];

}

}

}

// 生成一个随机id

private function getHotID(){

$hot\_id = 10000000+rand(00000001,99999999);

$MemberModel = new Member();

$is\_hot = $MemberModel->where(['hot\_id'=>$hot\_id])->find();

if($is\_hot){

$this->getHotID();

}else{

return $hot\_id;

}

}

// 会员登录

public function login($username,$password){

$result = $this->memberModel

->where(['password'=> passwd($password) ])

->where(function($query) use ($username){

$query->where('member',$username)->whereOr('telephone',$username);

})

->find();

return $result['id'];

}

// 会员登录状态

public function saveLogin($id){

$member = $this->getMemberInfo(['id'=>$id]);

unset($member['password']);

cookie('mid', $member['id']);

cookie('member',md5($member['id'].$member['member'].config('salekey')));

session('memberInfo',$member,'think');

if ( session('memberInfo', '', 'think') ) {

return true;

} else {

return false;

}

}

// 获取组织列表

public function getMyGroup($member\_id){

$result = [];

$pidArr = $this->memberModel->where(['pid'=>$member\_id])->select();

if($pidArr){

$arr = $pidArr->toArray();

$result = $this->getGroupTree($arr,0);

}

return $result;

}

private function getGroupTree($arr,$times){

$times++;

if($times < 4) {

foreach ($arr as $key => $value) {

$result = $this->memberModel->where(['pid'=>$value['id']])->select();

$arr[$key]['groupList'] = $result->toArray();

$arr[$key]['count'] = count($result->toArray());

// if($times<3){

// foreach ($arr[$key]['groupList'] as $k => $val) {

// $result = $this->memberModel->where(['pid'=>$val['id']])->select();

// $arr[$key]['groupList'][$key]['groupList'] = $result->toArray();

// }

// }

$arr[$key]['groupList'] = $this->getGroupTree($arr[$key]['groupList'],$times);

}

}

return $arr;

}

public function isFieldRegister($field,$value){

$result = $this->memberModel->where([$field=>$value])->find();

return $result?1:0;

}

// 查询会员是否有教练身份数据

public function hasCoach($memberid) {

$model = new Coach();

$res = $model::get(['member\_id'=>$memberid]);

if ($res) {

$return = $res->toArray();

$return['check\_status'] = $model::get(['member\_id'=>$memberid])->getData('status'); //审核状态

} else {

$return = 0;

}

return $return;

}

public function hasCamp($memberid) {

$model = new Camp();

$res = $model::get(['member\_id' => $memberid]);

if ($res) {

$return = $res->toArray();

$return['check\_status'] = $model::get(['member\_id' => $memberid])->getData('status'); //审核状态

}else {

$return = 0;

}

return $return;

}

}

类文件use app\service\WechatService;

<?php

namespace app\service;

use wechatsdk\TPwechat;

class WechatService

{

public $options;

public function \_\_construct() {

$this->options = array(

'token' => config('token'),

//'encodingaeskey'=> config('queue.encodingaeskey'),

'appid' => config('appid'),

'appsecret' => config('appsecret')

);

}

//获取全局access\_token

public function authactoken() {

$weObj = new TPwechat($this->options);

$appid = $this->options['appid'];

$appsecret = $this->options['appsecret'];

$authname = 'wechat\_access\_token'.$appid;

$auth\_cache = cache($authname);

if (!$auth\_cache) {

$result = $weObj->checkAuth($appid, $appsecret);

return $result;

} else {

return $auth\_cache;

}

}

// 公众号绑定

public function mpbind() {

$weObj = new TPwechat($this->options);

$weObj->valid();

ob\_clean();

}

// 获取公众号菜单

public function getmenu() {

$weObj = new TPwechat($this->options);

$menu = $weObj->getMenu();

return $menu;

}

/\*\* 设置公众号菜单

\* @param $menu 菜单数据数组

\* @return bool

\*/

public function setmenu($menu) {

$weObj = new TPwechat($this->options);

return $weObj->createMenu($menu);

}

// 生成用户授权链接

public function oauthredirect($callback) {

$weObj = new TPwechat($this->options);

return $weObj->getOauthRedirect($callback,config('state'));

}

// 获取授权用户信息

public function oauthuserinfo() {

$weObj = new TPwechat($this->options);

$accesstoken = $weObj->getOauthAccessToken();

//dump($accesstoken);

if ($accesstoken) {

$userinfo = $weObj->getOauthUserinfo($accesstoken['access\_token'], $accesstoken['openid']);

if ($userinfo) {

return $userinfo;

}

}

}

/\*\*

\* 发送模板消息

\* @param array $data 消息结构

\* ｛

"touser":"OPENID",

"template\_id":"ngqIpbwh8bUfcSsECmogfXcV14J0tQlEpBO27izEYtY",

"url":"http://weixin.qq.com/download",

"topcolor":"#FF0000",

"data":{

"参数名1": {

"value":"参数",

"color":"#173177" //参数颜色

},

"Date":{

"value":"06月07日 19时24分",

"color":"#173177"

},

"CardNumber":{

"value":"0426",

"color":"#173177"

},

"Type":{

"value":"消费",

"color":"#173177"

}

}

}

\*/

public function sendTemplate($data) {

$weObj = new TPwechat($this->options);

return $weObj->sendTemplateMessage($data);

}

/\*\* jssdk 签名验证 用于分享页面

\* @param $url 分享页面url 不带传参

\* @return array|bool

\*/

public function jsapi($url) {

$appid = $this->options['appid'];

$weObj = new TPwechat($this->options);

return $weObj->getJsSign($url, 0, '', $appid);

}

}

类文件use app\model\Coach;

<?php

namespace app\model;

use think\Model;

use traits\model\SoftDelete;

class Coach extends Model {

use SoftDelete;

protected $deleteTime = 'delete\_time';

protected $autoWriteTimestamp = true;

protected $readonly = [

'create\_time',

'student\_flow',

'kps',

'lesson\_flow',

'coach\_rank',

'coach\_level'

];

public function getStatusAttr($value){

$status = [-1=>'禁用',0=>'待审核',1=>'已审核',2=>'未通过'];

return $status[$value];

}

public function member(){

return $this->hasOne('member','id','member\_id',[],'left join');

}

}

类文件use app\common\validate\CoachVal;

<?php

namespace app\common\validate;

use think\Validate;

class CoachVal extends Validate{

protected $rule = [

'coach' => 'require|max:60',

'member\_id' => 'require|max:60',

];

protected $message = [

'coach.require' => '用户名必须',

'member\_id' =>'用户名ID必须',

];

protected $scene = [

'add' => ['coach'],

'edit' => ['coach'],

];

}

类文件use app\model\Grade;

<?php

namespace app\model;

use think\Model;

use traits\model\SoftDelete;

class Grade extends Model {

use SoftDelete;

protected $deleteTime = 'delete\_time';

protected $autoWriteTimestamp = true;

protected $readonly = [

'create\_time',

];

public function getStatusAttr($value){

$status = [-1=>'预排班级',1=>'当前班级', 2 => '下架班级'];

return $status[$value];

}

public function member(){

return $this->hasOne('member','id','member\_id',[],'left join');

}

public function student(){

return $this->hasMany('student','grade\_id','member\_id',[],'left join');

}

}

类文件use app\model\GradeMember;

<?php

namespace app\model;

use think\Model;

use traits\model\SoftDelete;

class GradeMember extends Model {

use SoftDelete;

protected $deleteTime = 'delete\_time';

protected $autoWriteTimestamp = true;

protected $readonly = [

'create\_time',

'status',

'type',

];

public function getStatusAttr($value){

$status = [0=>'待审核',1=>'正常',2=>'退出',3=>'被开除',4=>'已毕业'];

return $status[$value];

}

public function getTypeAttr($value){

$type = [1=>'正式生',2=>'体验生'];

return $type[$value];

}

// 关联训练营

public function camp(){

return $this->hasOne('camp','id','camp\_id',[],'left join');

}

// 关联member

public function member(){

return $this->hasOne('member','id','member\_id',[],'left join');

}

// 关联班级

public function grade(){

return $this->hasOne('grade','id','grade\_id',[],'left join');

}

// 关联??多对多???

public function coach(){

return $this->belongsToMany('coach','grade\_member','coach\_id','');

}

// 关联学生

public function student(){

return $this->hasOne('student','id','student\_id',[],'left join');

}

}

类文件 use app\model\Student;

<?php

namespace app\model;

use think\Model;

use traits\model\SoftDelete;

class Student extends Model{

use SoftDelete;

protected $deleteTime = 'delete\_time';

protected $readonly = [

'member\_id',

'total\_lession',

'finished\_total',

];

protected $autoWriteTimestamp = true;

public function member(){

return $this->hasOne('member','member\_id','id',[],'left join');

}

public function grade(){

return $this->hasOne('grade','grade\_id','id',[],'left join');

}

}

类文件 use app\model\Camp;

<?php

namespace app\model;

use think\Model;

use app\validate\CampVal;

use traits\model\SoftDelete;

class Camp extends Model{

use SoftDelete;

protected $deleteTime = 'delete\_time';

protected $autoWriteTimestamp = true;

protected $readonly = [

//"id",

"max\_member",

"total\_coach",

"act\_coach",

"total\_member",

"act\_member",

"total\_lessons",

"finished\_lessons",

"star",

"total\_grade",

"act\_grade",

"camp\_base",

"type",

"status",

//"create\_time",

];

public function getStatusAttr($value)

{

$status = [0=>'待审核',1=>'已审核',2=>'关闭',3=>'未通过'];

return $status[$value];

}

}

类文件 use app\common\validate\CampVal;

<?php

namespace app\common\validate;

use think\Validate;

class CampVal extends Validate{

protected $rule = [

'camp' => 'require|max:60|unique:camp,camp',

'member\_id' => 'require',

'realname' => 'require',

];

protected $message = [

'camp.require' => '请输入训练营名称',

'camp.unique' =>'训练营名称被占用',

'member\_id.require' => '请先注册会员',

'realname.require' => '请输入创建者真实姓名',

];

protected $scene = [

'add' => ['realname','member\_id','camp'],

'edit' => ['id']

];

}

类文件 use app\common\validate\CampCommentVal;

<?php

namespace app\common\validate;

use think\Validate;

class CampCommentVal extends Validate{

protected $rule = [

'member\_id' => 'require',

'camp\_id' => 'require',

];

protected $message = [

'camp\_id.require' => '请输入训练营名称',

'member\_id.require' => '查不到会员信息',

];

protected $scene = [

'add' => ['member\_id','camp\_id'],

'edit' => ['member\_id','camp\_id'],

];

}

类文件 use app\model\CampMember;

<?php

namespace app\model;

use think\Model;

use traits\model\SoftDelete;

class CampMember extends Model {

use SoftDelete;

protected $table = 'camp\_member';

protected $deleteTime = 'delete\_time';

protected $autoWriteTimestamp = true;

// protected $readonly = [

// 'create\_time',

// 'status',

// 'type',

// ];

public function getStatusAttr($value){

$status = [0=>'待审核',1=>'正常',2=>'退出',-1=>'被辞退',3=>'已毕业'];

return $status[$value];

}

public function getTypeAttr($value){

$status = [-1=>'粉丝',1=>'学生',2=>'教练',3=>'管理员',4=>'创建者',5=>'其他'];

return $status[$value];

}

public function coach(){

return $this->hasOne('coach','member\_id','member\_id',[],'LEFT');

}

}