# 博客系统代码文档

说明:本代码不包括css样式表.

前台开发使用.net的razor语法及vue，使用了布局页layout.

前台使用了一些js,css类库，主要包括bootstrap,jquery,layui,vue

字体类库使用：font-awesome.

文本编辑器使用：ueditor.

下拉树是将控件wdTree进行封装，将树形控件变成了comboboxTree.

基础表格使用：bootstrapTable.js库.

树型表格使用：FancyTree的treeGrid.

下拉列表：使用的vue-multiselect.

上传控件:使用的uploadify。经过封装后，能够上传多个控件.

后台使用mapper做实体与viewModel映射。Autofac做依赖注入。EntityFramework做ORM，采用code first ,Redis做缓存.

# 功能点如下

# 1系统管理

## 用户管理

\_index.cshtml布局页：

@{

Layout = null;

}

<!DOCTYPE html>

<html>

<head>

<title>@ViewBag.Title</title>

<meta name="viewport" content="width=device-width, initial-scale=1">

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<link href="~/Content/css/bootstrap.min14ed.css" rel="stylesheet" />

<link href="~/Content/css/font-awesome4.7.min12.css" rel="stylesheet" />

<link href="~/Content/css/animate.min2932.css" rel="stylesheet" />

<link href="~/Content/css/bootstrap-table.min12.css" rel="stylesheet"/>

<link href="~/Content/js/layui/css/layui.css" rel="stylesheet" />

@RenderSection("css",required:false)

<!-- 让IE8/9支持媒体查询，从而兼容栅格 -->

<!--[if lt IE 9]>

<script src="https://cdn.staticfile.org/html5shiv/r29/html5.min.js"></script>

<script src="https://cdn.staticfile.org/respond.js/1.4.2/respond.min.js"></script>

<![endif]-->

</head>

<body>

@RenderBody()

<script>

var \_hmt = \_hmt || [];

(function () {

var hm = document.createElement("script");

hm.src = "https://hm.baidu.com/hm.js?b74709fc39f4f8555f6437c174875680";

var s = document.getElementsByTagName("script")[0];

s.parentNode.insertBefore(hm, s);

})();

</script>

<script src="~/Content/js/jquery-2.1.4-min.js"></script>

<script src="~/Content/js/global.js?v=1"></script>

<script src="~/Content/js/common.js"></script>

<script src="~/Content/js/bootstrap.min.js"></script>

<script src="~/Content/js/bootstrap-table.min.js"></script>

<script src="~/Content/js/bootstrap-table-zh-CN.min.js"></script>

<script src="~/Content/js/vue/vue.min.js"></script>

<script src="~/Content/js/layui/layui.all.js"></script>

<script src="~/Content/js/plugins/validator.js"></script>

@RenderSection("scripts", required: false)

</body>

</html>

用户管理列表页

@{

Layout = "~/Views/Shared/\_Index.cshtml";

ViewBag.Title = "用户管理列表首页";

}

<!-- Content Header (Page header) -->

<section class="content-header">

<ol class="breadcrumb" style="margin-bottom:0px;">

<li><a href="#"><i class="fa fa-dashboard"></i> 系统管理</a></li>

<li class="active">用户信息管理</li>

</ol>

</section>

<section class="content">

<div class="panel panel-default">

<div class="panel-body" style="padding-top: -13px;padding-bottom:0" id="v-app">

<form id="formSearch" class="form-horizontal">

<div class="form-group" style="margin-top:5px">

<label class="control-label col-sm-1" for="txt\_search\_name">登录名</label>

<div class="col-sm-3">

<input type="text" class="form-control" v-model="LoginName">

</div>

<label class="control-label col-sm-1" for="txt\_search\_statu">邮箱</label>

<div class="col-sm-3">

<input type="text" class="form-control" v-model="Email">

</div>

<div class="col-sm-4" style="text-align:left;">

<button type="button" v-on:click="btn\_query" class="btn btn-primary"><i class="glyphicon glyphicon-search"></i> 查询</button>

<button type="button" v-on:click="btn\_clear" class="btn btn-danger"><i class="fa fa-undo"></i> 清空</button>

</div>

</div>

</form>

<div id="toolbar" class="btn-group">

<button id="btn\_refresh" type="button" class="btn btn-default" v-on:click="btn\_refresh">

<span class="glyphicon glyphicon-refresh" aria-hidden="true"></span>刷新

</button>

<button id="btn\_add" type="button" class="btn btn-default" v-on:click="btn\_add">

<span class="glyphicon glyphicon-plus" aria-hidden="true"></span>新增

</button>

<button id="btn\_edit" type="button" class="btn btn-default" v-on:click="btn\_edit">

<span class="glyphicon glyphicon-pencil" aria-hidden="true"></span>编辑

</button>

<button id="btn\_delete" type="button" class="btn btn-default" v-on:click="btn\_delete">

<span class="glyphicon glyphicon-remove" aria-hidden="true"></span>删除

</button>

<button id="btn\_reset\_pwd" type="button" class="btn btn-default" v-on:click="btn\_reset\_pwd">

<span class="fa fa-cog" aria-hidden="true"></span>重置密码

</button>

</div>

</div>

<div class="panel-body" style="padding-top: 0;padding-bottom:0">

<table id="dginfo"></table>

</div>

</div>

</section>

@section scripts{

<script type="text/javascript">

var vm = new Vue({

el: '#v-app',

data: {

LoginName: '',

Email: '',

},

mounted: function () {

this.init\_datagrid();

},

methods: {

btn\_clear: function () {

vm.LoginName = '';

vm.Email = '';

vm.btn\_refresh();

},

btn\_refresh: function () {

$('#dginfo').bootstrapTable('refresh');

},

btn\_add: function () {

this.editinfo('新增用户', 0);

},

btn\_edit: function () {

com.edit('#dginfo', 'boot', function (Id) {

vm.editinfo('编辑用户', Id);

});

},

btn\_delete: function () {

com.delete('/Plat/AppUser/Delete');

},

btn\_query: function () {

$('#dginfo').bootstrapTable('refresh');

},

btn\_clear: function () {

this.Email = '';

this.LoginName = '';

$('#dginfo').bootstrapTable("refresh");

},

btn\_reset\_pwd: function () {

com.edit('#dginfo', 'boot', function (Id) {

$.layerConfirm({

content: '你确定要将此用户的密码重置为初始值：1234',

callback: function () {

com.ajax({

url: '/Plat/AppUser/ResetPassword',

data: { Id: Id },

showMsg: true

});

}

})

});

},

editinfo: function (title, Id) {

$.layerOpen({

title: title,

width: '50%',

height: '70%',

content: '/Plat/AppUser/Form/' + Id,

yes: function (iframebody, iframeWin, index) {

iframeWin.vm.submit(function () {

vm.btn\_clear();

});

}

});

},

get\_status: function (code) {

if (code == null || code == undefined || code == "") return null;

var obj = { 'Enable': true, 'Forbidden': false };

return obj[code];

},

init\_datagrid: function () {

$('#dginfo').dataGrid({

url: '/Plat/AppUser/GetDataGrid', //请求后台的URL（\*）

sortOrder: "desc,asc", //排序方式

sortName: 'LoginName,Id',

queryParams: function (params) {

if (vm) {

params.Email = vm.Email;

params.LoginName = vm.LoginName;

}

return params;

},

uniqueId: "Id",

columns: [{ checkbox: true },

{ field: 'LoginName', title: '登录名', sortable: true },

{ field: 'Email', title: '邮箱', sortable: true },

{ field: 'NickName', title: '昵称', sortable: true },

{

field: 'Id', title: '配置角色', formatter: function (value, row) {

return String(row.UserRoleList)

}

},

{ field: 'LoginCount', title: '登录次数', sortable: true },

{

field: 'isBindQQ', title: '是否绑定QQ', sortable: true, formatter: function (value, row) {

var text = '数据为空', label\_class = 'warning';

if (value === true) {

text = '已绑定';

label\_class = 'success';

} else if (value === false) {

text = "未绑定";

label\_class = 'danger';

}

return com.format\_str('<span class="label label-{0}">{1}</span>', label\_class, text);

}

},

{

field: 'StatusCode', title: '状态', sortable: true, formatter: function (value, row) {

return com.format\_enable(vm.get\_status(value))

}

},

{ field: 'LastLoginTime', title: '最后登录时间', sortable: true }]

});

}

}

});

</script>

}

用户管理表单页

@{

ViewBag.Title = "用户新增编辑管理";

Layout = "~/Views/Shared/\_Form.cshtml";

}

<div id="v-app" class="container">

<form action="/" method="post">

<div class="form-group">

<label>登录名</label>

<input type="hidden" v-model="Id" value="" />

<input type="text" v-model="LoginName" class="form-control" />

</div>

<div class="form-group">

<label>邮箱</label>

<input type="text" v-model="Email" class="form-control" />

</div>

<div class="form-group">

<label>昵称</label>

<input type="text" v-model="NickName" class="form-control" />

</div>

<div class="form-group">

<label>状态</label>

<input data-value="Enable" type="radio" id="input-4" name="StatusCode" v-bind:checked="StatusCode=='Enable'">

<label for="input-4">启用 </label>

<input data-value="Forbidden" type="radio" id="input-3" name="StatusCode" v-bind:checked="StatusCode=='Forbidden'">

<label for="input-3">禁用</label>

</div>

<div class="form-group">

<label>角色配置</label>

<multiselect v-model="roleIds" :hide-selected="false" label="text" track-by="id" placeholder="--请配置角色--" :options="roleJson" :multiple="true"

:searchable="true" :clear-on-select="false"

:close-on-select="false">

<span slot="noResult">Oops! No elements found. Consider changing the search query.</span>

</multiselect>

</div>

</form>

</div>

@section css{

<link href="~/Content/css/vue-multiselect.min31f.css" rel="stylesheet" />

<link href="~/Content/css/plugins/skins/all.css" rel="stylesheet" />

}

@section scripts{

<script src="~/Content/js/vue/vue-multiselect.min.js"></script>

<script src="~/Content/js/plugins/icheck.min.js"></script>

<script>

var json = JSON.parse(`@Html.Raw(ViewBag.ViewModel)`);

Vue.component('Multiselect', VueMultiselect.Multiselect);

json.StatusCode == null && (json.StatusCode= 'Enable');

var vm = new Vue({

el: '#v-app',

data: json,

beforeCreate: function () {

json.roleJson = [];

json.roleIds == null && (json.roleIds = []);

},

mounted: function () {

this.get\_rolejson();

this.init\_form();

},

methods: {

submit: function (callback) {

var roleIds = [];

vm.roleIds.forEach(u => {

roleIds.push(u.id);

});

vm.roleIds = String(roleIds);

vm.roleJson = [];

com.ajax({

url: '/Plat/AppUser/SubmitForm',

data: vm.$data,

showMsg: true,

success: callback

})

},

get\_rolejson: function () {

com.ajax({

url: '/Plat/AppRole/GetRoleJson',

showLoading: false,

success: function (d) {

json.roleJson = d;

}

});

},

changeStatus: function () {

var pressed = $(this).attr('aria-pressed');

$(this).attr('aria-pressed', !pressed);

vm.Status = !pressed;

},

init\_form: function () {

$('form .form-group input[type=radio]').on('ifChecked', function (event) {

vm.StatusCode = $(event.target).attr('data-value');

}).iCheck({

labelHover: false,

cursor: true,

checkboxClass: 'icheckbox\_square-blue',

radioClass: 'iradio\_square-blue',

increaseArea: '20%'

});

}

},

});

</script>

}

用户控制器。此类继承CrudController

public class AppUserController : CrudController<AppUser, AppUserViewModel>

{

// GET: Plat/AppUser

IAppUserRepository \_service;

public AppUserController(IAppUserRepository service, IComponentContext container) : base(container)

{

this.\_service = service;

}

[HttpGet]

public override ActionResult Form(int id = 0)

{

if (id == 0)

{

ViewBag.ViewModel = this.GetFormJson(id).ToJson();

}

else

{

var userEntity = \_service.IQueryable(u => u.Id == id).Select(u => new {

u.Id,

u.LoginName,

u.NickName,

u.Email,

u.StatusCode,

roleIds = u.UserRoles.Select(r => new {

id = r.RoleId,

text = r.AppRole.RoleName

}).ToList()

}).FirstOrDefault();

ViewBag.ViewModel = userEntity.ToJson();

}

return View();

}

[HttpPost]

public ActionResult GetDataGrid(Pagination pag, string LoginName = "", string Email = "")

{

return Content(\_service.GetDataGrid(pag, LoginName, Email).ToJson());

}

public ActionResult SubmitForm(AppUser userEntity,string roleIds)

{

AjaxResult result = \_service.IsRepeat(userEntity);

//用户名或邮箱重复

if (result.state == ResultType.error.ToString())

{

return Content(result.ToJson());

}

\_service.SubmitForm(userEntity, roleIds);

return Success();

}

/// <summary>

/// 重置用户密码

/// </summary>

/// <param name="Id">主键</param>

/// <returns></returns>

public ActionResult ResetPassword(int Id)

{

\_service.ResetPassword(new AppUser { Id = Id }, "1234");

return Success();

}

}

CrudController 此类功能强大，使用缓存技术对表单数据缓存，此类实现统一增删改，开发者只需要定义Form表单对应的一个viewModel.使用autoMapper，实现实体与实体之间的转换，此类还实现了新增时间，新增人，修改时间，修改人，删除时间，删除人，软删除操作。

public class CrudController<TEntityModel, TViewModel> : BaseController where TEntityModel : class, new() where TViewModel : class, new()

{

private readonly IRepositoryBase<TEntityModel> \_ibase;

protected readonly IRedisHelper \_redisHelper;

public CrudController(IComponentContext container)

{

\_ibase = container.Resolve<IRepositoryBase<TEntityModel>>();

this.\_redisHelper = container.Resolve<IRedisHelper>();

}

public virtual ActionResult Form(int Id)

{

ViewBag.ViewModel = this.GetFormJson(Id);

return View();

}

#region 根据ID获取数据

/// <summary>

/// 此时表单中没有默认值，就可以用这个

/// </summary>

/// <param name="Id"></param>

/// <returns></returns>

public string GetFormJson(int Id = 0)

{

TViewModel viewModel = new TViewModel { };

if (Id != 0)

{

TEntityModel entity = \_ibase.FindEntity(Id);

if (entity != null)

{

viewModel = Mapper.Map<TEntityModel, TViewModel>(entity);

}

}

return viewModel.ToJson();

}

#endregion

#region Edit 统一的新增，编辑方法，每次用一次拆箱，装箱，不知道会不会影响速度

/// <summary>

/// author:luozQ

/// function:Edit 统一的新增，编辑方法，每次用一次拆箱，装箱，不知道会不会影响速度

/// </summary>

/// <param name="Id">主键</param>

/// <param name="viewModel">实体对象</param>

/// <returns></returns>

[HttpPost]

public virtual ActionResult Edit(TViewModel viewModel, int id = 0)

{

if (!ModelState.IsValid)

{

return Error(ModelState.Values.Where(u => u.Errors.Count > 0).FirstOrDefault().Errors[0].ErrorMessage);

}

var userid = OperatorProvider.Provider.CurrentUser.UserId;

//新增

if (id == 0)

{

TEntityModel entity = Mapper.Map<TViewModel, TEntityModel>(viewModel);

var d = entity as ICreationAudited;

if (d != null)

{

d.CreatorTime = DateTime.Now;

d.CreatorUserId = userid;

d.DeleteMark = false;

TEntityModel tEntityModel = d as TEntityModel;

if (tEntityModel != null && \_ibase.Insert(tEntityModel) > 0)

{

return Success("新增成功");

}

}

return Error("新增失败");

}

else

{

var tEntityModel = \_ibase.FindEntity(id);

if (tEntityModel == null)

return Error($"Id为{id}未找到任何类型为{viewModel.GetType().Name}的实体对象");

TEntityModel entity = Mapper.Map(viewModel, tEntityModel);

var d = entity as IModificationAudited;

if (d != null)

{

d.LastModifyTime = DateTime.Now;

d.LastModifyUserId = userid;

TEntityModel tUpdateEntityModel = d as TEntityModel;

if (tUpdateEntityModel != null && \_ibase.Update(tUpdateEntityModel) > 0)

{

return Success("编辑成功");

}

}

return Error("编辑失败");

}

}

#endregion

/// <summary>

/// 统一的删除功能，其类要继承IDeleteAduited这个类才行。

/// </summary>

/// <param name="Id">主键Id</param>

/// <returns></returns>

[HttpPost]

public ActionResult Delete(int Id)

{

var codetableEntity = \_ibase.FindEntity(Id);

if (codetableEntity == null)

return Error($"Id为{Id}未找到任何类型为{codetableEntity.GetType().Name}的实体对象");

IDeleteAudited deleteAudited = new DeleteAudited { Id = Id, DeleteMark = true, DeleteTime = DateTime.Now, DeleteUserId = OperatorProvider.Provider.CurrentUser.UserId };

TEntityModel entity = Mapper.Map(deleteAudited, codetableEntity);

if (\_ibase.Update(entity, "DeleteMark", "DeleteUserId", "DeleteTime") > 0)

{

return Success("删除成功！");

}

return Error("删除失败！");

}

public class AppUserViewModel

{

public int Id { get; set; }

public string Email { get; set; }

public string LoginName { get; set; }

public string NickName { get; set; }

public string StatusCode { get; set; }

public List<TreeSelectModel> roleIds { get; set; }

}

[Table("AppUser")]

public class AppUser : IEntity<AppUser>, ICreationAudited, IDeleteAudited, IModificationAudited

{

public AppUser()

{

this.UserRoles = new HashSet<UserRole>();

this.Articles = new HashSet<Article>();

this.GuestBooks = new HashSet<GuestBook>();

this.Uploads = new HashSet<Upload>();

this.AppUserAuths = new HashSet<AppUserAuth>();

this.Reviews = new HashSet<Review>();

this.GuestBooks = new HashSet<GuestBook>();

}

[Key]

public int? Id { get; set; }

[StringLength(50)]

public string LoginName { get; set; }

[StringLength(50,ErrorMessage ="最长为11")]

public string QQ { get; set; }

[StringLength(50)]

public string LoginPassword { get; set; }

[StringLength(20,ErrorMessage ="手机长度不能超过20位")]

public string Phone { get; set; }

[StringLength(50, ErrorMessage = "邮件长度不能超过50位")]

public string Email { get; set; }

public bool EmailIsValid { get; set; }

[StringLength(50)]

public string Birthday { get; set; }

[StringLength(300)]

public string PersonalWebsite { get; set; }

[StringLength(10)]

public string Gender { get; set; }

[StringLength(50)]

public string NickName { get; set; }

[StringLength(100,ErrorMessage ="个性签名不能超过100字")]

public string PersonSignature { get; set; }

[StringLength(50)]

public string UserSecretkey { get; set; }

[StringLength(100)]

public string Avatar { get; set; }

public int LoginCount { get; set; }

public DateTime? LastLoginTime { get; set; }

[StringLength(20)]

public string LastLogIp { get; set; }

[StringLength(50)]

public string StatusCode { get; set; }

public int? CreatorUserId { get; set; }

public DateTime? CreatorTime { get; set; }

public bool? DeleteMark { get; set; }

public int? DeleteUserId { get; set; }

public DateTime? DeleteTime { get; set; }

public int? LastModifyUserId { get; set; }

public DateTime? LastModifyTime { get; set; }

public virtual ICollection<UserRole> UserRoles { get; set; }

public virtual ICollection<Article> Articles { get; set; }

public virtual ICollection<Upload> Uploads { get; set; }

public virtual ICollection<AppUserAuth> AppUserAuths { get; set; }

public virtual ICollection<Review> Reviews { get; set; }

public virtual ICollection<GuestBook> GuestBooks { get; set; }

}

public static void RegisterMappings()

{

Mapper.Initialize(cfg => {

cfg.CreateMap<AppUser, AppUserViewModel>();

cfg.CreateMap<IDeleteAudited, AppUser>();

cfg.CreateMap<AppUserViewModel, AppUser>();

}

其他的类也同样处理。

## 用户注册

系统采用qq快速登录后，就注册了一个用户。

QQ互联使用了QConnectSDK，操作极其简单。

#region qq登录重定向页面

/// <summary>

/// QQ登陆页面

/// </summary>

[HttpGet]

public ActionResult Login(string returnUrl)

{

this.Session["ReturnUrl"] = returnUrl;

var context = new QzoneContext();

string state = Guid.NewGuid().ToString().Replace("-", "");

Session["requeststate"] = state;

string scope = "get\_user\_info,add\_share,list\_album,upload\_pic,check\_page\_fans,add\_t,add\_pic\_t,del\_t,get\_repost\_list,get\_info,get\_other\_info,get\_fanslist,get\_idolist,add\_idol,del\_idol,add\_one\_blog,add\_topic,get\_tenpay\_addr";

var authenticationUrl = context.GetAuthorizationUrl(state, scope);

return new RedirectResult(authenticationUrl);

}

#endregion

#region qq互联回调页面

/// <summary>

/// 回调页面

/// </summary>

public ActionResult QQConnect()

{

if (Request.Params["code"] != null)

{

QOpenClient qzone = null;

var verifier = Request.Params["code"];

var state = Request.Params["state"];

string requestState = Session["requeststate"]?.ToString();

if (state == requestState)

{

qzone = new QOpenClient(verifier, state);

var currentUser = qzone.GetCurrentUser();

if (this.Session["QzoneOauth"] == null)

{

this.Session["QzoneOauth"] = qzone;

}

var openId = qzone.OAuthToken.OpenId;

var accessToken = qzone.OAuthToken.AccessToken;

var expiresAt = qzone.OAuthToken.ExpiresAt;

DateTime now = DateTime.Now;

AppUserAuth userAuth = \_appAuthRepository.IQueryable(r => r.OpenId == openId && r.DeleteMark == false).FirstOrDefault();

AppUser appUserEntity;

//如果未找到一个openid存在，说明当前用户未使用qq第三方登录

if (userAuth == null)

{

appUserEntity = new AppUser

{

NickName = currentUser.Nickname,

Avatar = currentUser.Figureurl,

Gender = currentUser.Gender,

CreatorTime = now,

LoginCount=1,

LastLoginTime=DateTime.Now,

LastLogIp=Net.Ip,

DeleteMark = false,

StatusCode = StatusCode.Enable.ToString(),

UserRoles = new List<UserRole> {

new UserRole{

RoleId=3

}

},

AppUserAuths = new List<AppUserAuth>

{

new AppUserAuth{

IdentityType=IdentityType.QQ.ToString(),

OpenId=openId,

AccessToken=accessToken,

ExpiresAt=expiresAt,

CreatorTime=now,

DeleteMark=false

}

}

};

\_appUserRepository.Insert(appUserEntity);

}

else

{

userAuth.AccessToken = accessToken;

userAuth.ExpiresAt = expiresAt;

userAuth.LastModifyTime = DateTime.Now;

\_appAuthRepository.Update(userAuth, "AccessToken", "ExpiresAt", "LastModifyTime");

appUserEntity = \_appUserRepository.IQueryable(r => r.Id == userAuth.UserId && userAuth.DeleteMark == false).FirstOrDefault();

appUserEntity.LoginCount += 1;

appUserEntity.LastLoginTime = now;

appUserEntity.LastLogIp = Net.Ip;

appUserEntity.LastModifyTime = now;

\_appUserRepository.Update(appUserEntity, "LoginCount", "LastLoginTime", "LastLogIp", "LastModifyTime");

}

var isPersistentCookie = true;

FormsAuthentication.SetAuthCookie(qzone.OAuthToken.OpenId, isPersistentCookie);

OperatorProvider op = OperatorProvider.Provider;

//保存用户信息

op.CurrentUser = new OperatorModel

{

UserId = appUserEntity.Id,

IsSystem = false,

LoginName = appUserEntity.LoginName,

LoginToken = accessToken,

LoginTime = now,

NickName = appUserEntity.NickName,

Avatar = appUserEntity.Avatar,

Email = appUserEntity.Email,

PersonalWebsite = appUserEntity.PersonalWebsite

};

}

else

{

LogFactory.GetLogger("QQ互联").Error("登录得到的state:"+ state+",session:"+requestState);

}

return Redirect(Url.Action("Index", "App"));

}

return View();

}

#endregion

## 用户登录

前台/Login/Index

@{

ViewBag.Title = "登录页面";

Layout = "~/Views/Shared/\_Index.cshtml";

}

<div class="login-container">

<div class="container">

<div class="row">

<div class="col-md-offset-4 col-md-4">

<form class="form-horizontal" id="v-app" v-on:keyup.enter="btn\_login">

<span class="heading">用户登录</span>

<div class="form-group">

<input type="text" class="form-control" v-model="loginName" isvalid="yes" checkexpession="NotNull" placeholder="用户名或电子邮箱" errormsg="用户名或电子邮箱">

<i class="fa fa-user"></i>

</div>

<div class="form-group help">

<input type="password" class="form-control" v-model="pwd" placeholder="密　码" isvalid="yes" checkexpession="NotNull" errormsg="密码">

<i class="fa fa-lock"></i>

<a href="#" v-on:click="forget\_pwd" class="fa fa-question-circle" data-toggle="tooltip" data-placement="top" title="忘记密码"></a>

</div>

<div class="form-group">

<button type="button" v-on:click="btn\_login" class="btn btn-primary" style="width:100%;"> 登录</button>

</div>

</form>

</div>

</div>

</div>

<div class="footer">

<p>

<a href="http://igeekfan.cn/" target="\_blank">

天上有木月博客<span>·</span>

</a><a> 2016~@DateTime.Now.Year ©</a><span>·</span>

<a href="https://github.com/luoyunchong/Mock.luo" target="\_blank">获取igeekfan博客代码</a>

</p>

<p>

</p>

</div>

</div>

@section scripts{

<script src="~/Content/js/plugins/jquery.md5.js"></script>

<script src="~/Content/js/plugins/icheck.min.js"></script>

<script>

var vm = new Vue({

el: "#v-app",

data: {

loginName: '',

pwd: '',

rememberMe: true

},

mounted: function () {

this.init\_check();

$('[data-toggle="tooltip"]').tooltip();

},

methods: {

init\_check: function () {

$('input[type=checkbox]').on('ifChanged', function (event) {

vm.rememberMe = $(event.target).prop('checked');

}).iCheck({

labelHover: false,

cursor: true,

checkboxClass: 'icheckbox\_square-blue',

increaseArea: '20%'

});

},

btn\_login: function () {

if (!$('#v-app').Validform()) {

return false;

}

var postdata = {

loginName: $.trim(vm.loginName),

pwd: $.md5($.trim(vm.pwd)),

rememberMe: vm.rememberMe

};

com.ajax({

url: '/Login/CheckLogin',

data: postdata,

success: function (jsondata) {

$.procAjaxMsg(jsondata, function () {

$.layerMsg(jsondata.message, jsondata.state, function () {

window.location.href = "/Home/Index";

});

}, function () {

$.layerMsg(jsondata.message, jsondata.state)

})

}

})

},

btn\_register: function () {

$.layerMsg('待完成！');

},

forget\_pwd: function () {

$.layerOpen({

title: "忘记密码",

width: '50%',

height: '63%',

content: '/Login/PwdReset',

shadeClose: false,

skin: 'layui-layer-lan',

title: '重置密码',

btn: []

});

}

}

})

</script>

}

## 找回密码

@{ Layout = null;}

<!DOCTYPE html>

<html lang="en">

<head>

<title>密码重置</title>

<meta http-equiv="content-type" content="text/html;charset=UTF-8" />

<link href="~/Content/css/bootstrap.min14ed.css" rel="stylesheet" />

<link href="~/Content/css/font-awesome4.7.min12.css" rel="stylesheet" />

<link href="~/Content/css/animate.min2932.css" rel="stylesheet" />

<link href="~/Content/js/layui/css/layui.css" rel="stylesheet" />

<!--[if lt IE 9]>

<script src="https://oss.maxcdn.com/html5shiv/3.7.3/html5shiv.min.js"></script>

<script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>

<![endif]-->

<style>

[v-cloak] { display: none }

#v-app > .container { margin-top: 20px; }

#v-app > .container > .row > h4 { margin-top: 10px; }

</style>

</head>

<body>

<div id="v-app">

<div class="container" v-if="sendcodeseen">

<div class="row text-center">

<h3>找回密码 </h3>

<h4>

验证码将会发送至您注册的邮箱

</h4>

</div>

<form>

<div class="form-group">

<label>邮箱</label>

<input type="email" class="form-control" id="email" name="email" v-model="email" placeholder="请输入邮箱">

</div>

<div class="form-group" style="position: relative;overflow:hidden;">

<label>验证码</label>

<input type="text" class="form-control" id="code" name="code" v-model="code" placeholder="请输入验证码">

<div style="position: absolute;right:2px;top: 27px;">

<img title="看不清楚？换一张" width="120" height="30" id="imgcode" :src="src"

alt="验证码" style="display: block;" v-on:click="changecode">

</div>

</div>

<div class="row text-center">

<button class="btn btn-primary" v-on:click="sendcode" type="button">获取验证码</button>

</div>

</form>

</div>

<div class="container" v-if="resetpwdseen" v-cloak>

<div class="row text-center">

<h3>重设密码</h3>

<h4>

验证码已经发送至您注册的邮箱

</h4>

</div>

<form>

<div class="form-group">

<label>邮箱</label>

<input type="hidden" name="pwdtoken" v-model="pwdtoken" />

<input type="email" class="form-control" name="account" v-model="account" disabled="disabled">

</div>

<div class="form-group">

<label>新密码</label>

<input type="password" class="form-control" name="newpwd" v-model="newpwd" placeholder="请输入新密码">

</div>

<div class="form-group" style="position: relative;overflow:hidden;">

<label>验证码</label>

<input type="text" class="form-control" name="emailcode" v-model="emailcode" placeholder="邮箱收到的6位数字验证码" maxlength="6">

<div style="position: absolute;right:0px;top: 25px;">

<button type="button" :class="btntimeclass" v-on:click="sendagin" v-bind:disabled="btntimedis">**{{** btntime **}}**</button>

</div>

</div>

<div class="row text-center">

<button class="btn btn-primary" v-on:click="resetpwd" type="button">重设密码</button>

</div>

</form>

</div>

</div>

<script src="~/Content/js/jquery-2.1.4-min.js"></script>

<script src="~/Content/js/global.js?v=1"></script>

<script src="~/Content/js/common.js"></script>

<script src="~/Content/js/bootstrap.min.js"></script>

<script src="~/Content/js/bootstrap-table.min.js"></script>

<script src="~/Content/js/bootstrap-table-zh-CN.min.js"></script>

<script src="~/Content/js/vue/vue.min.js"></script>

<script src="~/Content/js/layui/layui.all.js"></script>

<script src="~/Content/js/plugins/validator.js"></script>

<script>

var vm = new Vue({

el: '#v-app',

data: {

code: '',

email: '',

src: '/Login/GetAuthCode',

sendcodeseen: true,

resetpwdseen: false,

pwdtoken: '',

account: '',

newpwd: '',

emailcode: '',

btntime: '60秒后重发',

wait: 60,

btntimeclass: 'btn btn-default',

btntimedis: true

},

methods: {

changecode: function () {

this.src = '/Login/GetAuthCode?d=' + Math.random();

},

sendcode: function () {

if (this.code == '') {

$.layerMsg('验证码不能为空！');

return false;

}

if (this.email == '') {

$.layerMsg('邮箱不能为空！');

return false;

}

var data = vm.$data;

var data = {

code: this.code,

email: this.email

}

com.ajax({

data: data,

url: '/Login/SmsCode',

success: function (json) {

$.procAjaxMsg(json, function () {

$.layerMsg(json.message, 'ok');

vm.sendcodeseen = false;

vm.resetpwdseen = true;

vm.pwdtoken = json.data;

vm.account = vm.email

vm.btn\_timing();

}, function () {

$.layerMsg(json.message);

})

}

})

},

resetpwd: function () {

var data = {

pwdtoken: this.pwdtoken,

account: this.account,

newpwd: this.newpwd,

emailcode: this.emailcode

};

com.ajax({

data: data,

url: '/Login/ResetPwd',

success: function (json) {

$.procAjaxMsg(json, function () {

$.alertMsg(json.message, '提示', function () {

$.layerClose();

}, 'ok');

}, function () {

$.layerMsg(json.message);

});

}

})

},

sendagin: function () {

var account = this.account;

com.ajax({

url: '/Login/SmsCodeAgain',

data: { 'account': account },

success: function (json) {

$.procAjaxMsg(json, function () {

vm.pwdtoken = json.data;

vm.btn\_timing();

}, function () {

$.layerMsg(json.message);

});

}

})

},

btn\_timing: function () {

if (vm.wait == 0) {

//当计时器为0时，就将按钮设置为可点击

vm.wait = 60;

vm.btntimeclass = 'btn btn-primary';

vm.btntimedis = false;

vm.btntime = "点我重发";

} else {

//每过一秒按钮的字变一次

vm.btntimeclass = 'btn btn-default';

vm.btntime = vm.wait + "秒后重发";

vm.btntimedis = true;

vm.wait--;

window.setTimeout(function () {

vm.btn\_timing();

}, 1000);

}

}

}

});

</script>

</body>

</html>

两者后台在一个Controller中

后台：

首先会有一个BaseController

namespace Mock.Luo.Controllers

{

public abstract class BaseController : Controller

{

public OperatorProvider op = OperatorProvider.Provider;

public virtual ActionResult Index()

{

return View();

}

public virtual ActionResult Detail(int Id)

{

return View();

}

protected virtual ActionResult Success()

{

return Success("操作完成!");

}

protected virtual ActionResult Error()

{

return Error("操作失败！");

}

protected virtual ActionResult Success(string message)

{

return Content(new AjaxResult { state = ResultType.success.ToString(), message = message }.ToJson());

}

protected virtual ActionResult Success(string message, object data)

{

return Content(new AjaxResult { state = ResultType.success.ToString(), message = message, data = data }.ToJson());

}

protected virtual ActionResult Error(string message)

{

return Content(new AjaxResult { state = ResultType.error.ToString(), message = message }.ToJson());

}

protected virtual ActionResult Error(ResultType state, string message, object data)

{

return Content(new AjaxResult { state = state.ToString(), message = message, data = data }.ToJson());

}

/// <summary>

/// 简化返回json对象的包裹

/// </summary>

/// <param name="data">object对象</param>

/// <returns></returns>

protected virtual ActionResult Result(object data)

{

return Content(JsonHelper.SerializeObject(data));

}

}

}

BaseController中引用的AjaxResult类在这里，主要作用是前后台交互

public class AjaxResult

{

/// <summary>

/// 操作结果类型

/// </summary>

public string state { get; set; }

/// <summary>

/// 获取 消息内容

/// </summary>

public string message { get; set; }

/// <summary>

/// 获取 返回数据

/// </summary>

public object data { get; set; }

public static AjaxResult Info(string message, object data, string state)

{

return new AjaxResult { state = state, message = message, data = data };

}

public static AjaxResult Success(string message, object data = null)

{

return Info(message, data, ResultType.success.ToString());

}

public static AjaxResult Error(string message, object data = null)

{

return Info(message, data, ResultType.error.ToString());

}

}

/// <summary>

/// 表示 ajax 操作结果类型的枚举

/// </summary>

public enum ResultType

{

/// <summary>

/// 警告结果类型

/// </summary>

warning,

/// <summary>

/// 成功结果类型

/// </summary>

success,

/// <summary>

/// 异常结果类型

/// </summary>

error,

/// <summary>

/// 消息结果类型

/// </summary>

info,

/// <summary>

/// 未登录

/// </summary>

nologin,

/// <summary>

/// 没有权限

/// </summary>

nopermission

}

用户登录，注册在这个控制器中

namespace Mock.Luo.Controllers

{

[Skip]

public class LoginController : BaseController

{

// GET: Login

private readonly IAppUserRepository \_service;

public LoginController(IAppUserRepository \_service)

{

this.\_service = \_service;

}

public ActionResult GetAuthCode()

{

return File(new VerifyCode().GetVerifyCode(), @"image/Gif");

}

public ActionResult Mock()

{

OperatorProvider op = OperatorProvider.Provider;

op.CurrentUser = new OperatorModel

{

UserId = 2,

IsSystem = true,

LoginName = "admin",

LoginToken = Guid.NewGuid().ToString(),

UserCode = "1234",

};

HttpRuntime.Cache[op.CurrentUser.UserId.ToString()] = Session.SessionID;

return Success();

}

/// <summary>

/// 登录接口验证

/// </summary>

/// <param name="loginName">登录名/邮箱</param>

/// <param name="pwd">密码</param>

/// <param name="code">验证码</param>

/// <param name="token">唯一上下文token</param>

/// <returns></returns>

public ActionResult CheckLogin(string loginName, string pwd, bool rememberMe = false, string code = "", string token = "")

{

//if (code.IsNullOrEmpty())

//{

// return Error("验证码不能为空！");

//}

if (loginName.IsNullOrEmpty())

{

return Error("用户名不能为空！");

}

if (pwd.IsNullOrEmpty())

{

return Error("密码不能为空！");

}

//当token不为null,且等于当前上下文的token时，直接验证登录

if (!token.IsNullOrEmpty())

{

if (token == OperatorProvider.Provider.CurrentToken)

{

return Result(\_service.CheckLogin(loginName, pwd, rememberMe));

}

}

//if (code.IsNotNullOrEmpty() && code.ToLower() != OperatorProvider.Provider.CurrentCode)

//{

// return Error("验证码出错");

//}

return Result(\_service.CheckLogin(loginName, pwd, rememberMe));

}

public ActionResult PwdReset()

{

return View();

}

#region 重置密码，发送验证码 SmsCode

public ActionResult SmsCode(string email, string code)

{

if (string.IsNullOrEmpty(code))

{

return Error("验证码不能为空！");

}

else if (string.IsNullOrEmpty(email))

{

return Error("邮箱不能为空！");

}

else if (op.CurrentCode != code)

{

return Error("验证码错误，请重新输入");

}

else

{

return Result(\_service.SmsCode(email));

}

}

#endregion

#region 计时器60s过后，再次发送验证码

public ActionResult SmsCodeAgain(string account)

{

if (string.IsNullOrEmpty(account))

{

return Error("参数出错!");

}

else

{

return Result(\_service.SmsCode(account));

}

}

#endregion

#region 重置密码

public ActionResult ResetPwd(string pwdtoken, string account, string newpwd, string emailcode)

{

if (string.IsNullOrEmpty(pwdtoken))

{

return Error("token不能为空!");

}

if (string.IsNullOrEmpty(account))

{

return Error("帐号不能为空!");

}

if (string.IsNullOrEmpty(newpwd))

{

return Error("新密码不能为空!");

}

if (string.IsNullOrEmpty(emailcode))

{

return Error("邮箱验证码不能为空!");

}

AjaxResult amm = \_service.ResetPwd(pwdtoken, account, newpwd, emailcode);

return Content(amm.ToJson());

}

#endregion

}

}

用户操作的接口

public interface IAppUserRepository : IRepositoryBase<AppUser>

{

/// <summary>

/// 根据条件得到用户列表数据

/// </summary>

/// <param name="pag">分页条件</param>

/// <param name="LoginName">登录名</param>

/// <param name="Email">邮箱</param>

/// <returns></returns>

DataGrid GetDataGrid(Pagination pag, string LoginName, string Email);

/// <summary>

/// 新增用户，编辑用户信息

/// </summary>

/// <param name="userEntity">用户实体</param>

/// <param name="roleIds">角色id，以逗号分隔</param>

void SubmitForm(AppUser userEntity,string roleIds);

/// <summary>

/// 判断用户是否重复，用户的LoginName是否重复，Email是否重复

/// </summary>

/// <param name="userEntity"></param>

/// <returns></returns>

AjaxResult IsRepeat(AppUser userEntity);

/// <summary>

/// 验证登录

/// </summary>

/// <param name="loginName">登录名</param>

/// <param name="pwd">密码</param>

/// <returns>成功与否</returns>

AjaxResult CheckLogin(string loginName, string pwd,bool rememberMe);

/// <summary>

/// 重置密码

/// </summary>

/// <param name="userEntity">用户实体</param>

/// <param name="userPassword">用户密码</param>

void ResetPassword(AppUser userEntity, string userPassword);

/// <summary>

/// 根据登录的实体，通过session保存用户信息

/// </summary>

/// <param name="userEntity"></param>

void SaveUserSession(AppUser userEntity);

/// <summary>

/// 发送验证码

/// </summary>

/// <param name="email"></param>

/// <returns></returns>

AjaxResult SmsCode(string email);

/// <summary>

/// 重置密码

/// </summary>

/// <param name="pwdtoken">密码token，判断唯一性</param>

/// <param name="account">邮件</param>

/// <param name="newpwd">新密码</param>

/// <param name="emailcode">邮件验证码</param>

/// <returns></returns>

AjaxResult ResetPwd(string pwdtoken, string account, string newpwd, string emailcode);

}

在仓储实现层中 关于用户的操作都在 AppUserRepositroy

/// <summary>

/// 仓储实现层 AppUserRepositroy

/// </summary>]

public class AppUserRepositroy : RepositoryBase<AppUser>, IAppUserRepository

{

#region 构造方法

private readonly IAppModuleRepository \_ModuleService;

private readonly IRedisHelper \_iRedisHepler;

private readonly IMailHelper \_imailHelper;

public AppUserRepositroy(IAppModuleRepository \_ModuleService, IRedisHelper \_iRedisHepler, IMailHelper \_imailHelper)

{

this.\_ModuleService = \_ModuleService;

this.\_iRedisHepler = \_iRedisHepler;

this.\_imailHelper = \_imailHelper;

}

#endregion

#region 根据条件得到用户列表数据

public DataGrid GetDataGrid(Pagination pag, string LoginName, string Email)

{

Expression<Func<AppUser, bool>> predicate = u => u.DeleteMark == false && (LoginName == "" || u.LoginName.Contains(LoginName))

&& (Email == "" || u.Email.Contains(Email));

var dglist = this.IQueryable(predicate).Where(pag).Select(u => new

{

u.LoginName,

u.NickName,

u.Email,

u.Id,

u.LoginCount,

u.LastLoginTime,

u.StatusCode,

isBindQQ=u.AppUserAuths.Select(r=>r.IdentityType=="QQ").Count()>0,

UserRoleList = u.UserRoles.Select(r => r.AppRole.RoleName)

}).ToList();

return new DataGrid { rows = dglist, total = pag.total };

}

#endregion

#region 新增用户，编辑用户信息

public void SubmitForm(AppUser userEntity, string roleIds)

{

if (userEntity.Id == 0)

{

userEntity.Create();

string userPassword = "1234";//默认密码

userEntity.UserSecretkey = Md5.md5(Utils.CreateNo(), 16).ToLower();

userEntity.LoginPassword = Md5.md5(DESEncrypt.Encrypt(Md5.md5(userPassword, 32).ToLower(), userEntity.UserSecretkey).ToLower(), 32).ToLower();

userEntity.LoginCount = 0;

//新增时配置角色

if (roleIds.IsNotNullOrEmpty())

{

foreach (string id in roleIds.Split(','))

{

int.TryParse(id, out int result);

if (result == 0) continue;

UserRole userRoleEntity = new UserRole { RoleId = result };

userEntity.UserRoles.Add(userRoleEntity);

}

}

this.Insert(userEntity);

}

else

{

using (var db = new RepositoryBase().BeginTrans())

{

userEntity.Modify(userEntity.Id);

string[] modifystrs = { "LoginName", "StatusCode", "Email", "NickName", "LastModifyUserId", "LastModifyTime" };

this.Update(userEntity, modifystrs);

db.Delete<UserRole>(u => u.UserId == userEntity.Id);

if (roleIds.IsNotNullOrEmpty())

{

foreach (string id in roleIds.Split(','))

{

int.TryParse(id, out int result);

if (result == 0) continue;

UserRole userRoleEntity = new UserRole { RoleId = result, UserId = (int)userEntity.Id };

db.Insert(userRoleEntity);

}

}

db.Commit();

}

}

}

#endregion

#region 判断用户是否重复，用户的LoginName是否重复，Email是否重复

public AjaxResult IsRepeat(AppUser userEntity)

{

Expression<Func<AppUser, bool>> predicate = u => u.DeleteMark == false;

if (userEntity.Id == 0)

{

var loginNameExpression = predicate.And(r => r.LoginName == userEntity.LoginName);

if (this.IQueryable(loginNameExpression).Count() > 0)

{

return AjaxResult.Error("用户名已存在！");

}

else

{

var emailExpression = predicate.And(r => r.Email == userEntity.Email);

if (this.IQueryable(emailExpression).Count() > 0)

{

return AjaxResult.Error("此邮箱已存在！");

}

}

}

else

{

var loginNameExpression = predicate.And(r => r.LoginName == userEntity.LoginName && r.Id != userEntity.Id);

if (this.IQueryable(loginNameExpression).Count() > 0)

{

return AjaxResult.Error("用户名已存在！");

}

else

{

var emailExpression = predicate.And(r => userEntity.Email != "" && r.Email == userEntity.Email && r.Id != userEntity.Id);

if (this.IQueryable(emailExpression).Count() > 0)

{

return AjaxResult.Error("此邮箱已存在！");

}

}

}

return AjaxResult.Success("邮箱与用户名都可用！");

}

#endregion

#region 验证登录

public AjaxResult CheckLogin(string loginName, string pwd, bool rememberMe)

{

AjaxResult ajaxResult;

try

{

AppUser userEntity = new AppUser { };

userEntity = this.IQueryable().Where(t => (t.LoginName == loginName || t.Email == loginName) && t.DeleteMark == false).FirstOrDefault();

if (userEntity != null)

{

if (userEntity.UserSecretkey.IsNullOrEmpty())

{

throw new Exception("用户密钥丢失，请联系管理员！");

}

//登录成功

string dbPassword = Md5.md5(DESEncrypt.Encrypt(pwd.ToLower(), userEntity.UserSecretkey).ToLower(), 32).ToLower();

//登录成功

if (dbPassword == userEntity.LoginPassword)

{

//根据登录实体，去缓存用户数据

this.SaveUserSession(userEntity);

//记住密码

if (rememberMe == true)

{

}

ajaxResult = AjaxResult.Success("登录成功!");

DateTime now = DateTime.Now;

userEntity.LoginCount += 1;

userEntity.LastLoginTime = now;

userEntity.LastLogIp = Net.Ip;

userEntity.LastModifyTime = now;

this.Update(userEntity, "LoginCount", "LastLoginTime", "LastLogIp", "LastModifyTime");

}

else

{

throw new Exception("密码不正确，请重新输入");

}

}

else

{

throw new Exception("账户不存在，请重新输入");

}

}

catch (Exception ex)

{

ajaxResult = AjaxResult.Error(ex.Message);

Log log = LogFactory.GetLogger("登录日志记录");

log.Error(ex.Message);

}

return ajaxResult;

}

#endregion

#region 重置密码

public void ResetPassword(AppUser userEntity, string userPassword)

{

userEntity.UserSecretkey = Md5.md5(Utils.CreateNo(), 16).ToLower();

userEntity.LoginPassword = Md5.md5(DESEncrypt.Encrypt(Md5.md5(userPassword, 32).ToLower(), userEntity.UserSecretkey).ToLower(), 32).ToLower();

string[] modifstr = { "UserSecretkey", "LoginPassword", };

this.Update(userEntity, modifstr);

}

#endregion

#region 使用session暂存登录用户信息

/// <summary>

/// 使用session暂存登录用户信息

/// </summary>

/// <param name="userEntity"></param>

public void SaveUserSession(AppUser userEntity)

{

OperatorProvider op = OperatorProvider.Provider;

//保存用户信息

op.CurrentUser = new OperatorModel

{

UserId = userEntity.Id,

IsSystem = userEntity.LoginName == "admin" ? true : false,

LoginName = userEntity.LoginName,

LoginToken = Guid.NewGuid().ToString(),

UserCode = "1234",

LoginTime = DateTime.Now,

NickName = userEntity.NickName,

Avatar = userEntity.Avatar,

Email=userEntity.Email,

PersonalWebsite=userEntity.PersonalWebsite

};

//缓存存放单点登录信息

ICache cache = CacheFactory.Cache();

op.Session[userEntity.Id.ToString()] = userEntity.LoginName;//必须使用这个存储一下session，否则sessionid在每一次请求的时候，都会为变更

cache.WriteCache<string>(userEntity.Id.ToString(), op.Session.SessionID, DateTime.UtcNow.AddMinutes(60));

//登录权限分配,根据用户Id获取用户所拥有的权限，可以在登录之后的Home界面中统一获取。

//op.ModulePermission = \_ModuleService.GetUserModules(userEntity.Id);

op.ModulePermission = null;

}

#endregion

#region 重置密码，发送验证码 SmsCode

public AjaxResult SmsCode(string email)

{

AjaxResult amm;

int limitcount = 10;

int limitMinutes = 10;

if (!Validate.IsEmail(email))

{

return AjaxResult.Error("邮箱格式不正确");

}

AppUser userEntity = this.IQueryable(u => u.Email == email).FirstOrDefault();

if (userEntity == null)

{

amm = AjaxResult.Error("此邮箱尚未注册！");

}

else

{

string count = \_iRedisHepler.StringGet<string>(email);

//缓存十分钟，如果缓存中存在,且请求次数超过10次，则返回

if (!string.IsNullOrEmpty(count) && Convert.ToInt32(count) >= limitcount)

{

amm = AjaxResult.Error("没收到邮箱：请联系igeekfan@163.com");

}

else

{

#region 发送邮箱，并写入缓存，更新登录信息表的token,date,code

int num = 0;

if (!string.IsNullOrEmpty(count))

{

num = Convert.ToInt32(count);

}

string countplus1 = num + 1 + "";

\_iRedisHepler.StringSet<string>(email, countplus1, new TimeSpan(0, limitMinutes, 0));

ResetPwd resetpwdEntry = new ResetPwd

{

UserID = (int)userEntity.Id,

ModifyPwdToken = Utils.GuId(),

PwdCodeTme = DateTime.Now,

ModfiyPwdCode = Utils.RndNum(6),

LoginName = userEntity.LoginName,

NickName=userEntity.NickName

};

//将发送验证码的数据存入redis缓存中

\_iRedisHepler.StringSet(email + "sendcodekey", resetpwdEntry, new TimeSpan(0, limitMinutes, 0));

//发送找回密码的邮件

string body = UiHelper.FormatEmail(resetpwdEntry, "PwdReSetTemplate");

\_imailHelper.SendByThread(email, "[、天上有木月博客] 密码找回", body);

#endregion

//将修改密码的token返回给前端

amm = AjaxResult.Info("验证码已发送至你的邮箱！", resetpwdEntry.ModifyPwdToken, ResultType.success.ToString());

}

}

return amm;

}

#endregion

#region 重置密码 ResetPwd

public AjaxResult ResetPwd(string pwdtoken, string account, string newpwd, string emailcode)

{

AjaxResult amm;

var userEntity = this.IQueryable(u => u.Email == account).FirstOrDefault();

if (userEntity != null)

{

//从缓存中取出存放的验证码的键（邮箱+"sendcodekey"）得到重置密码的对象

ResetPwd rpwdEntry = \_iRedisHepler.StringGet<ResetPwd>(account + "sendcodekey");

if (rpwdEntry != null && rpwdEntry.ModifyPwdToken == pwdtoken)

{

if (rpwdEntry.ModfiyPwdCode != emailcode)

{

amm = AjaxResult.Error("验证码不正确！");

}

else

{

string cacheaccount = \_iRedisHepler.StringGet<string>(account + "modfiyPwdKey");

if (!string.IsNullOrEmpty(cacheaccount))

{

amm = AjaxResult.Error("10分钟内只可修改一次密码，请稍后再试!");

}

else

{ //重置密码成功之后将帐号写入cache中，10分钟内只可修改一次密码

ResetPassword(userEntity, newpwd);

amm = AjaxResult.Success("重置密码成功！");

\_iRedisHepler.StringSet<string>(account + "modfiyPwdKey", account, new TimeSpan(0, 10, 0));

}

}

}

else

{

amm = AjaxResult.Error("验证码过期了！");

}

}

else

{

amm = AjaxResult.Error("此帐号尚未注册");

}

return amm;

}

#endregion

}

在这个类中，使用了依赖注入autofac，在WEB项目中Global.asax

调用了此方法：

public class AutofacConfig

{

/// <summary>

/// 负责调用autofac框架实现业务逻辑层和数据仓储层程序集中的类型对象的创建 /// 负责创建MVC控制器类的对象(调用控制器中的有参构造函数),接管DefaultControllerFactory的工作

/// </summary>

public static void Register()

{

//实例化一个autofac的创建容器

ContainerBuilder builder = new ContainerBuilder();

var service = Assembly.Load("Mock.Domain");

builder.RegisterControllers(Assembly.GetExecutingAssembly());

builder.RegisterAssemblyTypes(service).AsImplementedInterfaces();

builder.RegisterType<RedisHelper>().As<IRedisHelper>().SingleInstance();

builder.RegisterType<MailHelper>().As<IMailHelper>().SingleInstance();

//创建一个Autofac的容器

var container = builder.Build();

//将MVC的控制器对象实例 交由autofac来创建

DependencyResolver.SetResolver(new AutofacDependencyResolver(container));

}

}

这里还用到了缓存redis，使用autofac，还能实现单例。

在这里就不放出iredis方法的实现。其主要实现一个缓存的功能。在这里不多介绍。

## 用户个人信息维护

本页面主要是对个人资料的维护，我的资料，头像，帐号和密码，帐号绑定。

<div class="layui-container user-main" id="v-app">

<ul class="layui-nav layui-nav-tree" lay-filter="test">

<li class="layui-nav-item"><a href="/App/Index" target="\_blank"><i class="layui-icon">&#xe68e;</i>瞧瞧主页</a></li>

<li class="layui-nav-item layui-this"><a href="/Account/Set"><i class="layui-icon">&#xe620;</i>基础设置</a></li>

<li class="layui-nav-item"><a href="/Account/Comment"><i class="layui-icon">&#xe611;</i>我的评论</a></li>

<li class="layui-nav-item"><a href="/Account/GuestBook"><i class="layui-icon">&#xe63c;</i>我的留言</a></li>

<li class="layui-nav-item"><a href="/Account/Collect"><i class="layui-icon">&#xe622;</i>收藏</a></li>

</ul>

<div class="panel-user">

<div class="layui-tab layui-tab-brief" lay-filter="docDemoTabBrief">

<ul class="layui-tab-title">

<li class="layui-this">我的资料</li>

<li>头像</li>

<li>帐号和密码</li>

<li>帐号绑定</li>

</ul>

<div class="layui-tab-content" style="padding: 20px 0;">

<div class="layui-tab-item layui-show">

<form class="layui-form layui-form-pane">

<div class="layui-form-item">

<div class="layui-inline">

<label class="layui-form-label">昵称</label>

<div class="layui-input-inline">

<input type="text" v-model="user.NickName" required lay-verify="required" placeholder="这里应该写昵称" autocomplete="off" class="layui-input">

</div>

</div>

<div class="layui-inline">

<label class="layui-form-label">QQ</label>

<div class="layui-input-inline">

<input type="text" v-model="user.QQ" required lay-verify="required" placeholder="这里应该写QQ号" autocomplete="off" class="layui-input">

</div>

</div>

</div>

<div class="layui-form-item">

<label class="layui-form-label">手机号</label>

<div class="layui-input-inline">

<input type="text" v-model="user.Phone" required lay-verify="required" placeholder="手机号在这填" autocomplete="off" class="layui-input">

</div>

</div>

<div class="layui-form-item">

<div class="layui-inline">

<label class="layui-form-label">性别</label>

<div class="layui-input-block">

<input type="radio" name="Gender" lay-filter="Gender" value="保密" title="保密" :checked="user.Gender=='保密'">

<input type="radio" name="Gender" lay-filter="Gender" value="男" title="男" :checked="user.Gender=='男'">

<input type="radio" name="Gender" lay-filter="Gender" value="女" title="女" :checked="user.Gender=='女'">

</div>

</div>

</div>

<div class="layui-form-item">

<label class="layui-form-label">个人网站</label>

<div class="layui-input-block">

<input type="text" v-model="user.PersonalWebsite" class="layui-input">

</div>

</div>

<div class="layui-form-item layui-form-text">

<div class="layui-form-label">

个性签名

</div>

<div class="layui-input-block">

<textarea placeholder="说点什么吧" class="layui-textarea" v-model="user.PersonSignature "></textarea>

</div>

</div>

<div class="layui-form-item">

<button type="button" class="btn-submit" v-on:click="save\_user\_info">确认修改</button>

</div>

</form>

</div>

<div class="layui-tab-item">

<div class="user-avatar">

<button class="btn btn-primary" id="upload-img"><i class="fa fa-cloud-upload"></i>上传头像</button>

<p>建议尺寸168\*168，支持jpg、png、gif，最大不能超过50KB</p>

<img :src="user.Avatar" />

</div>

</div>

<div class="layui-tab-item">

<form class="layui-form layui-form-pane">

<div class="layui-form-item">

<label class="layui-form-label">用户名</label>

<div class="layui-input-inline">

<input type="text" v-model="user.LoginName" required lay-verify="required" placeholder="只可设置一次" autocomplete="off" class="layui-input">

</div>

<div class="layui-form-mid layui-word-aux">

<a href="javascript:void(0)" v-on:click="set\_loginName" v-show="LoginNameIsShow" style="font-size: 12px; color: #4f99cf;">设置</a>

用于登录帐号，长度（4-20个字符）及内容（只能是汉字、字母、下划线、数字）

</div>

</div>

<div class="layui-form-item">

<label class="layui-form-label">邮箱</label>

<div class="layui-input-inline">

<input type="email" v-model="user.Email" required lay-verify="required" placeholder="" autocomplete="off" class="layui-input">

</div>

<div>

<div class="layui-form-mid layui-word-aux" v-if="user.EmailIsValid">

<a href="javascript:void(0)" v-on:click="set\_email\_address" style="font-size: 12px; color: #4f99cf;" title="修改邮箱需要验证邮箱验证码">修改邮箱</a>

<span class="layui-icon" title="已绑定成功">&#xe6c6;</span> 已绑定成功（可用于登录）

</div>

<div class="layui-form-mid layui-word-aux" v-else="user.Email">

<a href="javascript:void(0)" v-on:click="set\_email\_address" style="font-size: 12px; color: #4f99cf;" title="绑定邮箱">绑定邮箱</a>

（先填写好邮箱，再点击绑定邮箱）

</div>

</div>

</div>

<div class="layui-form-item">

<label class="layui-form-label">帐号密码</label>

<div class="layui-input-inline" style="height:38px">

<a href="javascript:void(0)" v-on:click="set\_use\_pwd" class="pwd-text" :title="user.PwdIsSet?'修改密码':'设置密码'" v-text="user.PwdIsSet?'修改密码':'设置密码'"></a>

</div>

<div>

</div>

</div>

</form>

</div>

<div class="layui-tab-item app-bind">

<blockquote class="layui-elem-quote layui-quote-nm">

<i class="fa fa-qq icon-qq"></i>

<template v-if="user.isBindQQ">

<span v-text="user.NickName"></span>

<a href="#" v-on:click="unbind\_qq">（解绑）</a>

</template>

<template v-else>

<a href="/Account/bindQQ" onclick="layer.msg('正在绑定QQ', {icon:16, shade: 0.1, time:0})">立即绑定</a>

<span>，即可使用QQ帐号登录本博客</span>

</template>

</blockquote>

</div>

</div>

</div>

</div> <div v-show="validate\_email\_show" class="layui-form-pane" id="validate-email">

<form class="layui-form">

<div class="layui-form-item">

<label class="layui-form-label">邮箱</label>

<div class="layui-input-inline">

<input type="text" name="Email" v-model="user.Email" class="layui-input" disabled="disabled" readonly="readonly">

</div>

</div>

<div class="layui-form-item">

<label class="layui-form-label">验证码</label>

<div class="layui-input-inline">

<input type="text" name="Code" class="layui-input" v-model="Code">

<a class="btn btn-primary" v-on:click="get\_active\_code">**{{**TimeKeeping**}}**</a>

</div>

</div>

</form>

</div>

<div v-show="validate\_pwd\_show" class="layui-form-pane" id="validate-pwd">

<form class="layui-form">

<div class="layui-form-item" v-show="user.PwdIsSet">

<label class="layui-form-label">旧密码<strong>\*</strong></label>

<div class="layui-input-inline">

<input type="password" name="LoginPassword" v-model="LoginPassword" class="layui-input">

</div>

</div>

<div class="layui-form-item">

<label class="layui-form-label">新密码<strong>\*</strong></label>

<div class="layui-input-inline">

<input type="password" name="newPwd" v-model="newPwd" class="layui-input">

</div>

</div>

<div class="layui-form-item">

<label class="layui-form-label">重复密码<strong>\*</strong></label>

<div class="layui-input-inline">

<input type="password" name="newPwd2" v-model="newPwd2" class="layui-input">

</div>

</div>

</form>

</div>

</div>

@section scripts{

<script src="~/Content/js/plugins/jquery.md5.js"></script>

<script>

var userJson = JSON.parse('@Html.Raw(ViewBag.viewModel)');

var vm = new Vue({

el: '#v-app',

data: {

validate\_email\_show: false,

validate\_pwd\_show:false,

user: userJson,

LoginNameIsShow: (userJson.LoginName == "" || userJson.LoginName == null) ? true : false,

wait: 60,

TimeKeeping: '获取验证码',

Code: '',

EmailToken: '',

LoginPassword: '',

newPwd:'',

newPwd2:''

},

mounted: function () {

this.init\_layui\_form();

$('.scroll-c').css('display', 'none');

},

methods: {

init\_layui\_form: function () {

layui.form.on('radio(Gender)', function (data) {

vm.user.Gender = data.value;

});

this.upload\_img();

layui.form.render();

},

set\_timekeeping: function () {

if (this.wait == 0) {

vm.TimeKeeping = "重发验证码";

this.wait = 60;

} else {

vm.TimeKeeping = this.wait + "秒后重发";

this.wait--;

setTimeout(function () {

vm.set\_timekeeping()

}, 1000)

}

},

get\_active\_code: function (callback) {

if (this.wait < 60) {

return;

}

com.ajax({

url: '/Account/GetActiveCode',

data: {

Email: vm.user.Email

},

success: function (d) {

$.layerMsg(d.message, d.state);

$.procAjaxMsg(d, function () {

vm.EmailToken = d.data;

vm.set\_timekeeping();

if ($.isFunction(callback)) {

callback(d);

}

}, function () {

});

}

});

},

set\_loginName: function () {

if (this.user.LoginName == "") {

$.layerMsg('不可为空');

return;

}

$.post('/Account/SetLoginName', { LoginName: this.user.LoginName }, function (d) {

$.layerMsg(d.message, d.state);

$.procAjaxMsg(d, function () {

vm.LoginNameIsShow = false;

}, function () {

});

}, 'json');

},

set\_email\_address: function () {

if (vm.user.Email == "") {

$.layerMsg('邮箱不能为空！');

return;

}

if (!com.isEmail(vm.user.Email)) {

$.layerMsg('邮箱格式不正确！');

return;

}

this.validate\_email\_show = true;

$.layerOpen({

title: '修改邮箱',

btn: '确定',

type: 1,

width: 300,

content: $('#validate-email'),

yes: function (index, layero) {

console.log(layero);

com.ajax({

url: '/Account/SetEmail',

data: {

Email: vm.user.Email,

Code: vm.Code,

EmailToken: vm.EmailToken

},

showMsg: true,

success: function (d) {

vm.Code = "";

vm.user.EmailIsValid = true;

layer.close(index);

vm.validate\_email\_show = false;

}

});

},

shadeClose: false,

cancel: function (index, layero) {

vm.validate\_email\_show = false;

layer.close(index);

}

});

},

save\_user\_info: function () {

com.ajax({

data: this.user,

url: '/Account/SaveUserInfo',

showMsg: true,

success: function (d) {

//console.log(d)

}

});

},

upload\_img: function () {

layui.upload.render({

elem: '#upload-img',

url: '/Mock/Upload/LayuiImage',

done: function (res) {

//console.log(res);

if (res.code == 0) {

vm.user.Avatar = res.data.src;

$.post('/Account/SaveAvatar', { Avatar: vm.user.Avatar }, function (d) {

//console.log(d)

});

}

},

error: function () {

$.layerMsg('出错了！！')

}

})

},

set\_use\_pwd: function () {

this.validate\_pwd\_show = true;

$.layerOpen({

title: '修改密码',

btn: ['确定', '取消'],

type: 1,

width: 300,

content: $('#validate-pwd'),

yes: function (index, layero) {

console.log(layero);

if (vm.user.PwdIsSet==true&&vm.LoginPassword == "") {

$.layerMsg('旧密码不能为空！'); return;

}

if (vm.newPwd=="") {

$.layerMsg('新密码不能为空！'); return;

} if (vm.newPwd2 == "") {

$.layerMsg('重复密码不能为空！'); return;

}

if (vm.newPwd != vm.newPwd2) {

$.layerMsg('新密码与重复密码不一致！');

return;

}

com.ajax({

url: '/Account/ResetPwd',

data: {

LoginPassword:$.md5(vm.LoginPassword),

newPwd: vm.newPwd

},

showMsg: true,

success: function (d) {

vm.validate\_pwd\_show = false;

vm.user.PwdIsSet = true;

layer.close(index);

}

});

},

shadeClose: false,

cancel: function (index, layero) {

vm.validate\_pwd\_show = false;

}

});

},

unbind\_qq: function () {

$.layerMsg('暂不支持解绑，会在稍后的版本中更新！！！');

return;

$.layerConfirm('提示', '你确定要解绑此QQ吗，解绑后，无法使用此QQ')

}

}

})

</script>

}

}

public class AccountController : BaseController

{

// GET: Account

#region Constructr

private readonly IRedisHelper \_redisHelper;

private readonly IMailHelper \_imailHelper;

private readonly IAppUserRepository \_appUserRepository;

private readonly IReviewRepository \_reviewRepository;

private readonly IGuestBookRepository \_guestBookRepository;

public AccountController(IRedisHelper \_redisHelper, IMailHelper \_imailHelper, IAppUserRepository \_appUserRepository, IReviewRepository \_reviewRepository, IGuestBookRepository \_guestBookRepository)

{

this.\_redisHelper = \_redisHelper;

this.\_imailHelper = \_imailHelper;

this.\_appUserRepository = \_appUserRepository;

this.\_reviewRepository = \_reviewRepository;

this.\_guestBookRepository = \_guestBookRepository;

}

#endregion

/// <summary>

/// 帐号信息设置

/// </summary>

/// <returns></returns>

public ActionResult Set()

{

int? userid = op.CurrentUser.UserId;

var viewModel = \_appUserRepository.IQueryable(r => r.Id == userid).Select(r => new

{

r.LoginName,

r.Email,

r.QQ,

r.EmailIsValid,

r.Phone,

r.NickName,

r.Avatar,

r.Gender,

r.PersonSignature,

r.PersonalWebsite,

PwdIsSet = r.LoginPassword == null ? false : true,

isBindQQ = r.AppUserAuths.Select(u => u.IdentityType == "QQ").Count() > 0

}).FirstOrDefault();

ViewBag.viewModel = JsonHelper.SerializeObject(viewModel);

return View();

}

/// <summary>

/// 退出登录

/// </summary>

/// <returns></returns>

public ActionResult Logout()

{

OperatorProvider op = OperatorProvider.Provider;

op.RemoveCurrent();

FormsAuthentication.SignOut();

return RedirectToAction("Index", "App");

}

/// <summary>

/// 根据邮箱发送四位验证码，并将token返回给前端

/// </summary>

/// <param name="Email"></param>

/// <returns></returns>

public ActionResult GetActiveCode(string Email)

{

if (Email.IsNullOrEmpty())

{

return Error("Email不能为空!!!");

}

if (!Validate.IsEmail(Email))

{

return Error("Email格式不正确!!!");

}

int? Id = op.CurrentUser.UserId;

//bool EmailIsValid = \_appUserRepository.IQueryable(u => u.Id == Id).Select(r => r.EmailIsValid).FirstOrDefault();

//if (EmailIsValid)

//{

// return Error("您已经绑定成功了，请不要重复绑定！！！");

//}

//为了安全，一个用户IP10分钟内只能请求此接口3次

string IP = Net.Ip;

int count = \_redisHelper.StringGet<int>(IP);

if (count >= 3)

{

return Error("请求过于频繁，请稍后再试！");

}

count += 1;

\_redisHelper.StringSet(IP, count, new TimeSpan(0, 10, 0));

var amm = \_appUserRepository.IsRepeat(new AppUser

{

Id = Id,

LoginName=Utils.GuId(),

Email = Email

});

if (amm.state.Equals(ResultType.error.ToString()))

{

return Error("该邮箱已被绑定其他用户绑定!");

}

//缓存1小时

TimeSpan saveTime = new TimeSpan(1, 0, 0);

//生成token

string token = Utils.GuId();

string rand4Num = Utils.RndNum(4);

//redis缓存绑定邮箱随机token作为键，email作为值，随机u为键，当前登录id为值

\_redisHelper.StringSet(token, Email, saveTime);

\_redisHelper.StringSet(token + Email, rand4Num, saveTime);

EmailViewModel emailViewModel = new EmailViewModel

{

ToUserName = Email,

Code = rand4Num

};

string body = UiHelper.FormatEmail(emailViewModel, "SendCodeTemplate");

\_imailHelper.SendByThread(Email, "[、天上有木月博客] 邮箱激活通知", body);

return Success("邮箱发送成功，请查收", token);

}

/// <summary>

/// 生成一个token,发送邮箱给用户邮箱

/// /// </summary>

/// <returns></returns>

public ActionResult ActiveSendEmail(string Email)

{

if (Email.IsNullOrEmpty())

{

return Error("Email不能为空!!!");

}

if (!Validate.IsEmail(Email))

{

return Error("Email格式不正确!!!");

}

//缓存1小时

TimeSpan saveTime = new TimeSpan(1, 0, 0);

//生成token

string token = Utils.GuId();

string u = Utils.GuId();

//redis缓存绑定邮箱随机token作为键，email作为值，随机u为键，当前登录id为值

\_redisHelper.StringSet(token, Email, saveTime);

\_redisHelper.StringSet(u, op.CurrentUser.UserId, saveTime);

string rootUrl = Request.Url.Authority;

EmailViewModel emailViewModel = new EmailViewModel

{

ToUserName = Email,

Link = "http://" + rootUrl + "/App/ActiveEmail?t=" + token + "&u=" + u

};

string body = UiHelper.FormatEmail(emailViewModel, "ActiveEmailTemplate");

\_imailHelper.SendByThread(Email, "[、天上有木月博客] 邮箱激活通知", body);

return Success("邮箱发送成功，请查收");

}

/// <summary>

/// 修改email，需要上次缓存的验证码和token

/// </summary>

/// <param name="Email"></param>

/// <param name="EmailToken"></param>

/// <param name="Code"></param>

/// <returns></returns>

public ActionResult SetEmail(string Email, string EmailToken, string Code)

{

if (Code.IsNullOrEmpty())

{

return Error("验证码不能为空");

}

if (EmailToken.IsNullOrEmpty())

{

return Error("邮件标识符异常，请重新获取验证码！");

}

string email = \_redisHelper.StringGet(EmailToken);

if (email.IsNullOrEmpty())

{

return Error("您的操作已过期，请重试！");

}

if (Email.IsNullOrEmpty() || !email.Equals(Email))

{

return Error("邮箱参数异常");

}

string code = \_redisHelper.StringGet(EmailToken + email);

if (code.IsNullOrEmpty())

{

return Error("您的操作已过期，请重试！");

}

if (!Code.Equals(code))

{

return Error("您输入的验证码不正确，请重试!");

}

\_appUserRepository.Update(\_appUserRepository.IQueryable(u => u.Id == op.CurrentUser.UserId), u => new AppUser

{

Email = Email,

EmailIsValid = true

});

\_redisHelper.KeyDeleteAsync(EmailToken);

\_redisHelper.KeyDeleteAsync(EmailToken + Email);

return Success("绑定成功！");

}

/// <summary>

/// 设置用户名|仅一次，且不重复

/// </summary>

/// <param name="LoginName"></param>

/// <returns></returns>

public ActionResult SetLoginName(string LoginName)

{

bool nameValidate = Validate.IsValidUserName(LoginName);

if (!nameValidate)

{

return Error("您填写的用户名称不合法！");

}

int? UserId = op.CurrentUser.UserId;

//判断当前用户名是否存在重复

var amm = \_appUserRepository.IsRepeat(new AppUser

{

Id = UserId,

LoginName = LoginName

});

//重复直接返回

if (amm.state.Equals(ResultType.error.ToString()))

{

return Result(amm);

}

var tempIQueryable = \_appUserRepository.IQueryable(u => u.Id == UserId && u.DeleteMark == false);

string loginNameMs = tempIQueryable.Select(r => r.LoginName).FirstOrDefault();

//只有当前id查询出的用户名为空，才表明未设置过用户名

if (loginNameMs.IsNullOrEmpty())

{

\_appUserRepository.Update(tempIQueryable, u => new AppUser

{

LoginName = LoginName

});

}

else

{

return Error("您已经设置过用户名了，这个不能设置第二次");

}

return Success();

}

public ActionResult SaveUserInfo(AppUser userEntity)

{

if (!ModelState.IsValid)

{

return Error(ModelState.Values.Where(u => u.Errors.Count > 0).FirstOrDefault().Errors[0].ErrorMessage);

}

if (userEntity.NickName.IsNullOrEmpty())

{

return Error("昵称为必填项");

}

if (!Validate.IsValidUserName(userEntity.NickName))

{

return Error("昵称不合法，给我换、");

}

OperatorModel oUserModel = op.CurrentUser;

userEntity.Modify(oUserModel.UserId);

//\_appUserRepository.IQueryable(u => u.Id == oUserModel.UserId).Update(u => userEntity);

\_appUserRepository.Update(userEntity, "QQ", "Phone", "NickName", "Gender", "PersonSignature", "PersonalWebsite");

oUserModel.NickName = userEntity.NickName;

op.CurrentUser = oUserModel;

return Success();

}

/// <summary>

/// 保存用户头像| 用于上传头像的回调

/// </summary>

/// <returns></returns>

public ActionResult SaveAvatar(string Avatar)

{

int? id = op.CurrentUser.UserId;

\_appUserRepository.Update(\_appUserRepository.IQueryable(u => u.Id == id), r => new AppUser

{

Avatar = Avatar

});

OperatorModel oUserModel = op.CurrentUser;

oUserModel.Avatar = Avatar;

op.CurrentUser = oUserModel;

return Success();

}

[HttpPost]

public ActionResult ResetPwd(string newPwd, string LoginPassword = "")

{

int userId = (int)op.CurrentUser.UserId;

if (newPwd.IsNullOrEmpty())

{

return Error("新密码不能为空");

}

AppUser userEntity = \_appUserRepository.FindEntity(userId);

if (userEntity != null)

{

//当旧密码为空时，直接重置

if (userEntity.LoginPassword.IsNullOrEmpty())

{

\_appUserRepository.ResetPassword(userEntity, newPwd);

return Success("您的密码已经设置成功，请牢记你的密码噢！");

}

else

{

if (LoginPassword.IsNullOrEmpty())

{

return Error("旧密码不能为空");

}

if (userEntity.UserSecretkey.IsNullOrEmpty())

{

return Error("用户密钥丢失，请联系管理员重置密码");

}

string dbPwd = Md5.md5(DESEncrypt.Encrypt(LoginPassword.ToLower(), userEntity.UserSecretkey).ToLower(), 32).ToLower();

if (userEntity.LoginPassword.Equals(dbPwd))

{

\_appUserRepository.ResetPassword(userEntity, newPwd);

return Success("您的密码已经设置成功，请牢记你的密码噢！");

}

else

{

return Error("你的旧密码填写不对，无法重置密码！");

}

}

}

else

{

return Error("当前用户不存在");

}

}

## 登录权限过滤器

public class HandlerLoginAttribute : AuthorizeAttribute

{

/// <summary>

/// 登录过滤器，除了通有skip属性可以跳过登录验证的，其他操作都需要在已登录状态完成操作。

/// </summary>

/// <param name="filterContext"></param>

public override void OnAuthorization(AuthorizationContext filterContext)

{

//当有skip验证时，去除验证登录

bool skipignore = filterContext.ActionDescriptor.GetCustomAttributes(typeof(SkipAttribute),false).Length==1

|| filterContext.ActionDescriptor.ControllerDescriptor.IsDefined(typeof(SkipAttribute), false);

if (skipignore == true) return;

OperatorProvider op = OperatorProvider.Provider;

#region 判断用户是否登录|当前只可单个浏览器登录，即使是浏览器重开，也会导入sessionId变更，也就需要重新登录

if (op.CurrentUser == null)

{

StringBuilder sbScript = new StringBuilder();

sbScript.Append("<script type='text/javascript'>alert('请先登录!');top.location.href='/Login/Index?msg=noLogin'</script>");

filterContext.Result = new ContentResult() { Content = sbScript.ToString() };

}

else

{

//

bool loginOnce = Configs.GetValue("loginOnce").ToBoolean();

if (loginOnce)

{

//当前登录信息被覆盖，说明存在多个浏览器登录的情况

string userid = op.CurrentUser.UserId.ToString();

ICache cache = CacheFactory.Cache();

if (cache.GetCache<string>(userid) != op.Session.SessionID)

{

//清空Session

filterContext.HttpContext.Session.Remove(userid);

op.RemoveCurrent();

//跳转强制下线界面

this.HandleUnauthorizedRequest(filterContext);

}

}

}

#endregion

}

/// <summary>

/// 当重写验证返回为false时进入该重写方法

/// </summary>

/// <param name="filterContext"></param>

protected override void HandleUnauthorizedRequest(AuthorizationContext filterContext)

{

base.HandleUnauthorizedRequest(filterContext);

if (filterContext.HttpContext.Request.Url != null)

{

filterContext.Result = new RedirectResult("/Error/ReturnToLogin");

}

}

}

## 出错记录日志

public class HandlerErrorAttribute : HandleErrorAttribute

{

public override void OnException(ExceptionContext context)

{

base.OnException(context);

context.ExceptionHandled = true;

context.HttpContext.Response.StatusCode = 200;

ContentResult conResult = new ContentResult { Content = new AjaxResult { state = ResultType.error.ToString(), message = context.Exception.Message }.ToJson() };

context.Result = conResult;

this.WriteLog(context);

}

private void WriteLog(ExceptionContext context)

{

if (context == null)

return;

var log = LogFactory.GetLogger(context.Controller.ToString());

log.Error(context.Exception);

}

}

# 2.博客管理

## 文章管理

@{

Layout = "~/Views/Shared/\_Index.cshtml";

ViewBag.Title = "文章管理列表";

}

<section class="content-header">

<ol class="breadcrumb" style="margin-bottom:0px;">

<li><a href="#"><i class="fa fa-dashboard"></i> 博客管理</a></li>

<li class="active">文章管理</li>

</ol>

</section>

<section class="content">

<div class="panel panel-default">

<div class="panel-body" style="padding-top: -13px;" id="v-app">

<div id="toolbar" class="btn-group">

<button id="btn\_refresh" type="button" class="btn btn-default" v-on:click="btn\_refresh">

<span class="glyphicon glyphicon-refresh" aria-hidden="true"></span>刷新

</button>

<button id="btn\_add" type="button" class="btn btn-default" v-on:click="btn\_add">

<span class="glyphicon glyphicon-plus" aria-hidden="true"></span>新增

</button>

<button id="btn\_edit" type="button" class="btn btn-default" v-on:click="btn\_edit">

<span class="glyphicon glyphicon-pencil" aria-hidden="true"></span>编辑

</button>

<button id="btn\_delete" type="button" class="btn btn-default" v-on:click="btn\_delete">

<span class="glyphicon glyphicon-remove" aria-hidden="true"></span>删除

</button>

</div>

<table id="dginfo"></table>

</div>

</div>

</section>

@section css{

}

@section scripts{

<script type="text/javascript">

var vm = new Vue({

el: '#v-app',

mounted: function () {

this.init\_table();

},

methods: {

btn\_refresh: function () {

$('#dginfo').bootstrapTable('refresh');

},

btn\_add: function () {

this.editinfo('新增', 0);

},

btn\_edit: function () {

com.edit('#dginfo', 'boot', function (Id) {

vm.editinfo('编辑', Id);

});

},

btn\_delete: function () {

com.delete('/Plat/Article/Delete');

},

editinfo: function (title, Id) {

$.layerOpen({

title: title,

width: '80%',

height: '70%',

content: '/Plat/Article/Form/' + Id,

yes: function (iframebody, iframeWin, index) {

iframeWin.vm.submit(function () {

$('#dginfo').bootstrapTable("refresh");

});

}

});

},

init\_table: function () {

$('#dginfo').dataGrid({

url: '/Plat/Article/GetDataGrid',

pagination: true,

uniqueId: "Id",

search: true,

queryParams: function (params) {

return params;

},

columns: [{ checkbox: true },

{ field: 'Title', title: '标题', align: 'center' },

{ field: 'ItemName', title: '文章类型', align: 'center' },

{ field: 'LoginName', title: '作者', align: 'center' },

{

field: 'CommentQuantity', title: '评论量', align: 'center', formatter: function (value, row) {

if (value != null) {

return com.format\_str('<a href="javascript:vm.show\_review({0})" style="color: #2fa4e7;" title="点我查看评论" data-toggle="tooltip">{1}</a>', row.Id, value);

} else {

return '\_';

}

}

},

{

field: 'PointQuantity', title: '点赞量', align: 'center', formatter: function (value, row) {

if (value != null) {

return com.format\_str('<a href="javascript:vm.show\_point({0})" style="color: #2fa4e7;" title="点我查看点赞" data-toggle="tooltip">{1}</a>', row.Id, value);

} else {

return '\_';

}

}

},

{ field: 'ViewHits', title: '阅读次数', align: 'center' },

{

field: 'IsAudit', title: '关键字/来源/摘要/缩略图', align: 'center', formatter: function (value, row, index) {

var d = com.format\_str('<i class="fa fa-{0} fa-fw" ></i><i class="fa fa-{1} fa-fw" style="margin-left:10px;" ></i><i class="fa fa-{2} fa-fw" style="margin-left:10px;" ></i>', row.Keywords ? "check" : "close", row.Source ? "check" : "close", row.Excerpt ? "check" : "close")

if (row.thumbnail) {

var thumurl = com.format\_str('<a href="javascript:com.image\_preview\_dialog(\'' + row.thumbnail + '\')"; data-toggle="tooltip" data-original-title="点我查看" style="margin-left:10px;color:#2fa4e7;text-decoration:none;"> <i class="fa fa-photo fa-fw"></i> </a>');

return d + thumurl;

}

}

},

{

field: 'IsAudit', title: '状态', align: 'center', formatter: function (value, row, index) {

var isaudit = com.format\_str('<a data-toggle="tooltip" data-original-title="{0}"><i class="fa fa-{1}"></i></a>', row.IsAudit ? "已审核" : "未审核", row.IsAudit ? "check" : "close");

var isremd = com.format\_str('<a data-toggle="tooltip" style="margin-left:10px;" data-original-title="{0}"><i class="fa fa-{1}"></i></a>', row.IsStickie ? "置顶" : "未置顶", row.IsStickie ? "arrow-up" : "arrow-down");

var isstickie = com.format\_str('<a data-toggle="tooltip" style="margin-left:10px;" data-original-title="{0}"><i class="fa fa-{1}"></i></a>', row.Recommend ? "推荐" : "未推荐", row.Recommend ? "thumbs-up" : "thumbs-down");

return isaudit + isremd + isstickie;

}

},

{ field: 'CreatorTime', title: '发布时间', align: 'center' },

{

field: 'Id', title: '操作', align: 'center', formatter: function (value, row) {

return '<a href="/App/Detail/' + row.Id + '" target="\_blank" title="查看原文">查看原文</a>';

}

}],

callback: function () {

$("[data-toggle='tooltip']").tooltip();

$('.pull-right.search .form-control').attr('placeholder', '输入标题/作者');

}

});

}, show\_review: function (Id) {

$.layerOpen({

content: '/Plat/Review/Index?Id=' + Id,

width: '70%',

height: '69%',

btn:['关闭'],

yes: function (iframebody, iframeWin, index) {

top.layer.close(index);

}

});

}, show\_point: function (Id) {

$.layerOpen({

content: '/Plat/PointArticle/Index?Id=' + Id,

width: '70%',

height: '69%',

btn: ['关闭'],

yes: function (iframebody, iframeWin, index) {

top.layer.close(index);

}

});

}

}

});

</script>

}

@{

ViewBag.Title = "文章新增编辑管理";

Layout = "~/Views/Shared/\_Form.cshtml";

}

<form class="form-horizontal" id="v-app" style="margin-top:15px;">

<div class="container-fluid">

<div class="col-md-9 table-responsive">

<table class="table table-bordered">

<tr>

<th width="80">分类</th>

<td colspan="3">

<multiselect v-model="FId" :hide-selected="false" label="text" track-by="id" placeholder="--请选择分类--" :options="FTypeOptions" :multiple="false"

:searchable="true" :clear-on-select="false"

:close-on-select="true">

<span slot="noResult">Oops! No elements found. Consider changing the search query.</span>

</multiselect>

</td>

</tr>

<tr>

<th>关键词</th>

<td>

<input id="Keywords" name="Keywords" v-model="Keywords" title="请输入标题" class="form-control" type="text" checkexpession="LenStr" length="400">

</td>

<th>文章来源</th>

<td><input id="Source" name="Source" v-model="Source" class="form-control" type="text" checkexpession="LenStr" length="400"></td>

</tr>

<tr>

<th>文章标题<span style="color:red">\*</span></th>

<td colspan="3">

<input id="Title" name="Title" v-model="Title" data-placement="right" title="请输入标题" class="form-control" type="text" isvalid="yes" checkexpession="NotNull" length="200" errmsg="请输入标题">

</td>

</tr>

<tr>

<th>摘要</th>

<td colspan="3">

<textarea name="Excerpt" v-model="Excerpt" id="Excerpt" style="width: 98%; height: 80px;" class="form-control" isvalid="yes" checkexpession="LenStrOrNull" length="400"></textarea>

</td>

</tr>

<tr>

<th>标签</th>

<td colspan="3">

<div class="tagcloud">

<template v-for="item in Tag">

<a href="javascript:void(0);" v-bind:class="{ active: item.Active }" v-text="item.text" v-on:click="toogle\_tag(item.id)"></a>

</template>

</div>

</td>

</tr>

<tr>

<th>内容</th>

<td colspan="3">

<!-- 加载编辑器的容器 -->

<script id="Content" name="Content" type="text/plain" style="width:100%;">

</script>

</td>

</tr>

</table>

</div>

<div class="col-md-3">

<table class="table table-bordered">

<tbody>

<tr> <th> <b>缩略图</b></th></tr>

<tr>

<td>

<div style="text-align: center;">

<input type="hidden" v-model="thumbnail">

<a href="javascript:com.upload\_one\_image('图片上传','thumbnail');">

<img width="135" height="145" onerror="this.src='/Content/Images/default-thumbnail.png'" style="cursor:grab" :src="thumbnail" />

</a>

<input type="button" class="btn btn-default btn-sm" v-on:click="init\_img" value="取消图片">

</div>

</td>

</tr>

<tr> <th><b>评论</b></th> </tr>

<tr>

<td>

<label style="width: 88px"><a href="#" v-on:click="show\_review" style="color: #2fa4e7;">查看评论</a></label>

</td>

</tr>

<tr> <th><b>状态</b></th> </tr>

<tr>

<td>

<div class="radio">

<label><input type="radio" name="IsAudit" value="true" v-model="IsAudit">审核通过</label>

</div>

<div class="radio">

<label><input type="radio" name="IsAudit" value="false" v-model="IsAudit">待审核</label>

</div>

</td>

</tr>

<tr>

<td>

<div class="radio">

<label><input type="radio" name="IsStickie" value="true" v-model="IsStickie">置顶</label>

</div>

<div class="radio">

<label><input type="radio" name="IsStickie" value="false" v-model="IsStickie">未置顶</label>

</div>

</td>

</tr>

<tr>

<td>

<div class="radio">

<label><input type="radio" name="Recommend" value="true" v-model="Recommend">推荐</label>

</div>

<div class="radio">

<label><input type="radio" name="Recommend" value="false" v-model="Recommend">未推荐</label>

</div>

</td>

</tr>

<tr> <th><b>文章链接</b></th> </tr>

<tr>

<td>

<label style="width: 88px">

<a :href="linkUrl" target="\_blank" style="color: #2fa4e7;" v-text="linkUrl">

</a>

</label>

</td>

</tr>

</tbody>

</table>

</div>

</div>

</form>

@section scripts{

<script src="~/Content/js/vue/vue-multiselect.min.js"></script>

<script src="~/Content/js/ueditor/ueditor.config.js"></script>

<script src="~/Content/js/ueditor/ueditor.all.min.js"></script>

@\*<script src="~/Content/js/ueditor/toggletool.js"></script>\*@

<script>

var json = com.replaceSpce(`@Html.Raw(ViewBag.ViewModel)`);

var tagAcitve = JSON.parse('@Html.Raw(ViewBag.TagActive)');

Vue.component('Multiselect', VueMultiselect.Multiselect);

if (json.Id == null) {

json.IsAudit = true;

json.Recommend = false;

json.IsStickie = false;

json.ViewHits = 0;

json.CommentQuantity = 0;

json.PointQuantity = 0;

json.thumbnail = "/Content/Images/default-thumbnail.png";

json.linkUrl = "";

json.Archive ='@DateTime.Now.ToString("yyyy年MM")';

} else {

json.linkUrl = '/App/Detail/' + json.Id;

}

json.FId == null && (json.FId = -1);

json.Tag = [];

var ue;

var vm = new Vue({

el: '#v-app',

data: json,

beforeCreate: function () {

json.FTypeOptions = [];

json.FId == null && (json.FId = -1);

},

created: function () {

},

mounted: function () {

this.get\_ftypeoptions();

ue = UE.getEditor('Content', {

autoHeight: true

});

json.Content = decodeURIComponent(json.Content)

if (json.Id != null) {

ue.addListener("ready", function () {

// editor准备好之后才可以使用

ue.setContent(json.Content);

});

}

this.get\_tag\_options();

},

methods: {

submit: function (callback) {

if (!$('#v-app').Validform()) {

return false;

}

var content = ue.getContent();

vm.Content = encodeURIComponent(content);

vm.FId = vm.FId.id;

var TagArray = [];

$.each(vm.Tag, function (i, v) {

if (v.Active == true) {

TagArray.push(v.id);

}

});

vm.Tag = String(TagArray);

if (json.FId == -1) {

json.FId = null;

}

com.ajax({

url: '/Plat/Article/Edit/'+vm.Id,

data: vm.$data,

showMsg: true,

success: callback

})

},

init\_img: function () {

var imgsrc = '/Content/Images/default-thumbnail.png';

this.thumbnail = imgsrc;

return false;

},

get\_ftypeoptions: function () {

com.ajax({

url: '/Plat/ItemsDetail/GetCombobox?Encode=FTypeCode',

showLoading: false,

success: function (d) {

json.FTypeOptions = d;

//实现,编辑时，后台只传来类别的id时,将combobox选中的对象赋值给json.FId,这样才能实现选中效果,

//新增时，FId被赋值-1

$.each(d, function (i, v) {

if (json.FId ==v.id) {

json.FId = v;

return false;

}

});

}

})

},

show\_review: function () {

$.layerOpen({

content: '/Plat/Review/Index?Id=' + vm.Id,

width: '70%',

height: '64%'

})

},

get\_tag\_options: function () {

com.ajax({

url: '/Plat/ItemsDetail/GetCombobox?Encode=Tag',

showLoading: false,

success: function (data) {

if (data && data.length > 0) {

data.splice(0, 1);

$.each(data, function (i, d) {

data[i].Active = false;

});

}

if (tagAcitve && tagAcitve.length > 0) {

$.each(tagAcitve, function (i, d) {

$.each(data, function (j, v) {

if (v.id == d) {

data[j].Active = true;

return false;

}

});

});

}

vm.Tag = data;

}

})

},

toogle\_tag: function (Id) {

$.each(vm.Tag, function (i,d) {

if (Id == d.id) {

if (d.Active == true) {

vm.Tag[i].Active = false;

} else {

vm.Tag[i].Active = true;

}

return false;

}

})

}

}

});

</script>

}

文章管理的后台Controller

public class ArticleController : CrudController<Article, ArticleViewModel>

{

// GET: Plat/Article

IArticleRepository \_service;

IItemsDetailRepository \_itemsDetailRepository;

public ArticleController(IArticleRepository service, IItemsDetailRepository itemsDetailRepository, IComponentContext container) : base(container)

{

this.\_service = service;

this.\_itemsDetailRepository = itemsDetailRepository;

}

public override ActionResult Form(int Id)

{

//取出文章对应的多个标签Id

var tagActive = \_service.IQueryable(u => u.DeleteMark == false && u.Id == Id)

.Select(u => u.TagArts.Select(r => r.TagId)).FirstOrDefault();

ViewBag.TagActive = JsonHelper.SerializeObject(tagActive);

return base.Form(Id);

}

public ActionResult GetDataGrid(Pagination pag, string search = "")

{

return Content(\_service.GetDataGrid(pag, search).ToJson());

}

/// <summary>

/// 最新文章

/// </summary>

/// <returns></returns>

public ActionResult GetRecentArticle()

{

return Result(\_service.GetRecentArticle(5));

}

/// <summary>

/// 得到博客列表页面

/// </summary>

/// <param name="pag"></param>

/// <returns></returns>

[Skip]

public ActionResult GetIndexGird(Pagination pag, string category = "", string tag = "", string archive = "")

{

if (category.IsNullOrEmpty() && tag.IsNullOrEmpty() && archive.IsNullOrEmpty())

{

throw new ArgumentNullException("参数异常!!");

}

if (pag.sort.IsNullOrEmpty())

{

pag.sort = "Id";

}

if (pag.order.IsNullOrEmpty())

{

pag.order = "desc";

}

DataGrid dg = \_service.GetCategoryTagGrid(pag, category, tag, archive);

return Content(dg.ToJson());

}

/// <summary>

/// 文章新增，编辑，由于后期加了一个文章可以有多个标签，一个标签可对应多个文章，就是多对多的关系，

/// 所以，这里就需要重新对数据进行处理后，提交数据库

/// </summary>

/// <param name="viewModel"></param>

/// <param name="id"></param>

/// <returns></returns>

public override ActionResult Edit(ArticleViewModel viewModel, int id = 0)

{

if (!ModelState.IsValid)

{

return Error(ModelState.Values.Where(u => u.Errors.Count > 0).FirstOrDefault().Errors[0].ErrorMessage);

}

string TagIds = Request["Tag"].ToString();

List<TagArt> tagArtList = new List<TagArt> { };

if (TagIds.IsNotNullOrEmpty())

{

foreach (var i in TagIds.Split(',').Select(u => Convert.ToInt32(u)).ToList())

{

tagArtList.Add(new TagArt

{

TagId = i,

AId = id

});

}

}

Article entity;

if (id == 0)

{

entity = Mapper.Map<ArticleViewModel, Article>(viewModel);

entity.Create();

entity.TagArts = tagArtList;

entity.Archive = DateTime.Now.ToString("yyy年MM月");

\_service.Insert(entity);

}

else

{

var tEntityModel = \_service.FindEntity(id);

if (tEntityModel == null)

return Error($"Id为{id}未找到任何类型为{viewModel.GetType().Name}的实体对象");

entity = Mapper.Map(viewModel, tEntityModel);

entity.Modify(id);

using (var db = new RepositoryBase().BeginTrans())

{

db.Update(entity);

db.Delete<TagArt>(u => u.AId == id);

db.Insert(tagArtList);

db.Commit();

}

}

//要根据新增或修改的文章类型，来判断是否删除缓存中的数据

if (entity.FId != null)

{

string itemCode = \_itemsDetailRepository.IQueryable(u => u.Id == entity.FId).Select(r => r.ItemCode).FirstOrDefault();

if (itemCode.IsNotNullOrEmpty())

{

if (itemCode.Equals(CategoryCode.justfun))

{

\_redisHelper.KeyDeleteAsync(string.Format(ConstHelper.App, "JustFun"));

}

if (itemCode.Equals(CategoryCode.feelinglife))

{

\_redisHelper.KeyDeleteAsync(string.Format(ConstHelper.App, "FellLife"));

}

}

}

\_redisHelper.KeyDeleteAsync(string.Format(ConstHelper.Article, "GetRecentArticle"));

\_redisHelper.KeyDelete(string.Format(ConstHelper.Article, "archiveFile"));

return Success();

}

public interface IArticleRepository : IRepositoryBase<Article>

{

/// <summary>

/// 得到单个文章内容

/// </summary>

/// <param name="Id"></param>

/// <returns></returns>

ArtDetailDto GetOneArticle(int Id);

/// <summary>

/// 抽象取文章的列表数据

/// </summary>

/// <param name="artiQuaryable"></param>

/// <returns></returns>

List<ArtDetailDto> GetArticleList(IQueryable<Article> artiQuaryable);

/// <summary>

/// 后台管理的分页列表数据

/// </summary>

/// <param name="pag"></param>

/// <returns></returns>

DataGrid GetDataGrid(Pagination pag,string search);

/// <summary>

/// 得到最新的count篇文章 | 缓存

/// </summary>

/// <param name="count">条数</param>

/// <returns></returns>

List<ArtDetailDto> GetRecentArticle(int count);

/// <summary>

/// 根据评论量，点赞量，阅读次数得到最火文章 | 缓存

/// </summary>

/// <param name="count"></param>

/// <returns></returns>

List<Article> GetHotArticle(int count);

/// <summary>

/// 得到分类|标签 文章的分页数据

/// </summary>

/// <param name="pag"></param>

/// <param name="category"></param>

/// <param name="tag"></param>

/// <returns></returns>

DataGrid GetCategoryTagGrid(Pagination pag, string category, string tag,string archive);

/// <summary>

/// 缓存 得到站点统计数据

/// </summary>

/// <returns></returns>

SiteStatistics GetSiteData();

/// <summary>

/// 根据文章Id得到相关内容

/// </summary>

/// <param name="Id"></param>

/// <returns></returns>

ArtRelateDto GetRelateDtoByAId(int Id);

}

仓储的实现：

/// <summary>

/// 仓储实现层 ArticleRepositroy

/// </summary>]

public class ArticleRepositroy : RepositoryBase<Article>, IArticleRepository

{

#region Constructor

private readonly IRedisHelper \_iRedisHelper;

public ArticleRepositroy(IRedisHelper iRedisHelper)

{

\_iRedisHelper = iRedisHelper;

}

#endregion

#region private methods

private DataGrid GetGridByExpression(Expression<Func<Article, bool>> predicate, Pagination pag)

{

var dglist = this.IQueryable(predicate).Where(pag).Select(t => new

{

t.Id,

t.ItemsDetail.ItemCode,

t.ItemsDetail.ItemName,

t.CreatorTime,

t.Title,

t.Excerpt,

//t.Content,

t.ViewHits,

t.CommentQuantity,

t.Keywords,

t.Source,

t.thumbnail,

t.AppUser.LoginName,

t.IsAudit,

t.IsStickie,

t.PointQuantity,

t.Recommend

}).ToList();

return new DataGrid { rows = dglist, total = pag.total };

}

#endregion

public ArtDetailDto GetOneArticle(int Id)

{

ArtDetailDto artEntry = IQueryable(u => u.Id == Id && u.DeleteMark == false).Select(r => new ArtDetailDto

{

TypeName = r.ItemsDetail.ItemName,

TypeCode = r.ItemsDetail.ItemCode,

NickName = r.AppUser.NickName,

Avatar = r.AppUser.Avatar,

PersonSignature = r.AppUser.PersonSignature,

Id = r.Id,

Content = r.Content,

Title = r.Title,

CommentQuantity = r.CommentQuantity,

Excerpt = r.Excerpt,

CreatorUserId = r.CreatorUserId,

CreatorTime = r.CreatorTime,

ViewHits = r.ViewHits,

thumbnail = r.thumbnail,

PointQuantity = r.PointQuantity

}).FirstOrDefault();

artEntry.TimeSpan = TimeHelper.GetDateFromNow(artEntry.CreatorTime.ToDateTime());

return artEntry;

}

#region 抽象取文章 列表数据

public List<ArtDetailDto> GetArticleList(IQueryable<Article> artiQuaryable)

{

List<ArtDetailDto> artList = artiQuaryable.Select(r => new ArtDetailDto

{

TypeName = r.ItemsDetail.ItemName,

TypeCode = r.ItemsDetail.ItemCode,

NickName = r.AppUser.NickName,

Avatar = r.AppUser.Avatar,

PersonSignature = r.AppUser.PersonSignature,

Id = r.Id,

Title = r.Title,

CommentQuantity = r.CommentQuantity,

Excerpt = r.Excerpt,

CreatorUserId = r.CreatorUserId,

CreatorTime = r.CreatorTime,

ViewHits = r.ViewHits,

thumbnail = r.thumbnail,

}).ToList();

artList.ForEach(u =>

{

u.TimeSpan = TimeHelper.GetDateFromNow(u.CreatorTime.ToDateTime());

});

return artList;

}

#endregion

#region 后台管理的分页列表数据

public DataGrid GetDataGrid(Pagination pag, string search)

{

Expression<Func<Article, bool>> predicate = u => u.DeleteMark == false

&& (search == "" || u.Title.Contains(search) || u.AppUser.LoginName.Contains(search));

return this.GetGridByExpression(predicate, pag);

}

#endregion

/// <summary>

/// 最新的文章| 缓存

/// </summary>

/// <param name="count"></param>

/// <returns></returns>

public List<ArtDetailDto> GetRecentArticle(int count)

{

return \_iRedisHelper.UnitOfWork(string.Format(ConstHelper.Article, "GetRecentArticle"), () =>

{

IQueryable<Article> artiQuaryable = this.IQueryable(u => u.DeleteMark == false).OrderByDescending(r => r.Id).Take(count);

return this.GetArticleList(artiQuaryable);

});

}

#region 根据评论量，点赞量，阅读次数得到最火文章 | 缓存

public List<Article> GetHotArticle(int count)

{

return \_iRedisHelper.UnitOfWork(string.Format(ConstHelper.Article, "GetHotArticle"), () =>

{

return this.IQueryable(u => u.DeleteMark == false).Select(u => new

{

u.Id,

u.Title,

u.ViewHits,

u.CommentQuantity,

u.PointQuantity,

HotQuantity = u.ViewHits + u.CommentQuantity + u.PointQuantity

}).OrderByDescending(r => r.HotQuantity).Take(count).ToList().Select(u => new Article

{

Id = u.Id,

Title = u.Title,

}).ToList();

});

}

#endregion

/// <summary>

/// 得到分类|标签 文章的分页数据

/// </summary>

/// <param name="pag"></param>

/// <param name="category"></param>

/// <param name="tag"></param>

/// <returns></returns>

public DataGrid GetCategoryTagGrid(Pagination pag, string category, string tag, string archive)

{

Expression<Func<Article, bool>> expression = u => u.DeleteMark == false;

if (tag.IsNotNullOrEmpty())//tag:标签时，按照标签内容去查询列表数据

{

expression = expression.And(u => u.TagArts.Where(r => r.ItemsDetail.ItemCode == tag).Count() > 0);

}

else if (category.IsNotNullOrEmpty())

{

expression = expression.And(u => category == "" || u.ItemsDetail.ItemCode == category);

}

else

{

expression = expression.And(u => u.Archive == archive);

}

IQueryable<Article> iQueryableArt = base.IQueryable(expression);

//前台文章列表的特殊性，需要按照特定要求取出数据

List<ArtDetailDto> rows = this.GetArticleList(iQueryableArt.Where(pag));

return new DataGrid { total = iQueryableArt.Count(), rows = rows };

}

public SiteStatistics GetSiteData()

{

return \_iRedisHelper.UnitOfWork(string.Format(ConstHelper.Article, "GetSiteData"), () =>

{

var artQuery = base.IQueryable(u => u.DeleteMark == false);

return new SiteStatistics

{

ArticleCount = artQuery.Count(),

PointViewCount = artQuery.Sum(u => u.PointQuantity),

ArticleTypeCount = base.Db.Set<ItemsDetail>().AsNoTracking().Where(u => u.DeleteMark == false && u.Items.EnCode == EnCode.FTypeCode.ToString()).Count(),

TagCount = base.Db.Set<ItemsDetail>().AsNoTracking().Where(u => u.DeleteMark == false && u.Items.EnCode == EnCode.Tag.ToString()).Count(),

ReplyCount = base.Db.Set<Review>().AsNoTracking().Where(u => u.DeleteMark == false).Count(),

ViewHitCount = artQuery.Sum(r => r.ViewHits)

};

});

}

#region 根据文章Id得到相关内容

public ArtRelateDto GetRelateDtoByAId(int Id)

{

IQueryable<Article> iQ = base.IQueryable(u => u.DeleteMark == false);

//文章归档 | 可缓存

List<BaseDto> archiveFile = \_iRedisHelper.UnitOfWork(string.Format(ConstHelper.Article, "archiveFile"), () =>

{

return base.IQueryable(u => u.DeleteMark == false && u.IsAudit == true).GroupBy(u => u.Archive).Select(u => new

{

u.Key,

u.FirstOrDefault().Archive,

u.FirstOrDefault().CreatorTime,

count = u.Count(),

}).OrderBy(u=>u.CreatorTime).ToList().Select(u => new BaseDto

{

text = u.Archive,

code = u.count.ToString()

}).ToList();

});

//从文章列表中取出5条博主最后新增时间的置顶文章 | 可缓存

List<BaseDto> recommendArt = \_iRedisHelper.UnitOfWork(string.Format(ConstHelper.Article, "recommendArt"), () =>

{

return base.IQueryable(u => u.DeleteMark == false && u.IsAudit == true).OrderByDescending(u => u.CreatorTime).Take(5).ToList().Select(u => new BaseDto { Id = u.Id, text = u.Title }).ToList();

});

//取出分类目录:分类编码，分类名称

List<BaseDto> category = \_iRedisHelper.UnitOfWork(string.Format(ConstHelper.Article, "category"), () =>

{

return base.Db.Set<ItemsDetail>().AsNoTracking().Where(u => u.Items.EnCode == EnCode.FTypeCode.ToString()).Select(u => new BaseDto { Id = u.Id, text = u.ItemName, code = u.ItemCode }).ToList();

});

//取出分类FId,和文章对应的标签多个Id

var iTagFid = iQ.Where(r => r.Id == Id).Select(r => new

{

r.FId,

TagArts = r.TagArts.Select(u => new { u.AId, u.TagId, u.ItemsDetail.ItemCode, u.ItemsDetail.ItemName })

}).FirstOrDefault();

int? FId = iTagFid.FId;

List<int> tagIdList = iTagFid.TagArts.Select(u => u.TagId).ToList();

Expression<Func<Article, bool>> predicate = u => u.FId == FId;

//文章对应的多个标签

List<BaseDto> ArtTag = new List<BaseDto>();

if (tagIdList.Count > 0)

{

predicate = predicate.Or(u => u.TagArts.Select(r => tagIdList.Contains(r.Id)).Count() > 0);

ArtTag = iTagFid.TagArts.Select(u => new BaseDto

{

Id = u.TagId,

text = u.ItemName,

code = u.ItemCode

}).ToList();

}

//有关本文章的相关文章5条

List<BaseDto> relateArt = iQ.Where(predicate)

.OrderByNewId().Take(5).Select(u => new BaseDto { Id = u.Id, text = u.Title }).ToList();

//随机文章

List<BaseDto> randomArt = iQ.Select(u => new BaseDto { Id = u.Id, text = u.Title }).Take(5).ToList();

ArtRelateDto ardList = new ArtRelateDto

{

RelateArt = relateArt,

Category = category,

RecommendArt = recommendArt,

ArchiveFile = archiveFile,

RandomArt = randomArt,

ArtTag = ArtTag

};

return ardList;

}

#endregion

}

[Table("Article")]

[Serializable]

public class Article : IEntity<Article>, ICreationAudited, IDeleteAudited, IModificationAudited

{

public Article()

{

this.Reviews = new HashSet<Review>();

this.TagArts = new HashSet<TagArt>();

}

public int? Id { get; set; }

public int? FId { get; set; }

[ForeignKey("FId")]

public ItemsDetail ItemsDetail { get; set; }

[StringLength(200)]

public string Title { get; set; }

[StringLength(400)]

public string Keywords { get; set; }

[StringLength(400)]

public string Source { get; set; }

[StringLength(400)]

public string Excerpt { get; set; }

public string Content { get; set; }

public int? ViewHits { get; set; }

public int? CommentQuantity { get; set; }

public int? PointQuantity { get; set; }

[StringLength(400)]

public string thumbnail { get; set; }

public bool? IsAudit { get; set; }

public bool? Recommend { get; set; }

public bool? IsStickie { get; set; }

/// <summary>

/// 随笔档案 如2019年1月

/// </summary>

[StringLength(50)]

public string Archive { get; set; }

public int? CreatorUserId { get; set; }

[ForeignKey(" CreatorUserId")]

public AppUser AppUser { get; set; }

public DateTime? CreatorTime { get; set; }

public bool? DeleteMark { get; set; }

public int? DeleteUserId { get; set; }

public DateTime? DeleteTime { get; set; }

public int? LastModifyUserId { get; set; }

public DateTime? LastModifyTime { get; set; }

public virtual ICollection<Review> Reviews { get; set; }

public virtual ICollection<TagArt> TagArts { get; set; }

}

## 评论审核

@{

Layout = "~/Views/Shared/\_Index.cshtml";

ViewBag.Title = "文章审核";

}

<section class="content-header">

<ol class="breadcrumb" style="margin-bottom:0px;">

<li><a href="#"><i class="fa fa-dashboard"></i> 博客管理</a></li>

<li class="active">文章审核</li>

</ol>

</section>

<div class="layui-fluid" style="margin-top:20px;" id="v-app">

<div id="toolbar" class="btn-group" style="margin-bottom:15px;">

<button type="button" class="btn btn-default" v-on:click="btn\_refresh">

<span class="glyphicon glyphicon-refresh" aria-hidden="true"></span>刷新

</button>

<button type="button" class="btn btn-default" v-on:click="btn\_aduit">

<span class="glyphicon glyphicon-thumbs-up" aria-hidden="true"></span>审核

</button>

<button type="button" class="btn btn-default" v-on:click="btn\_black">

<span class="glyphicon glyphicon-bed" aria-hidden="true"></span>拉黑

</button>

<button type="button" class="btn btn-default" v-on:click="btn\_delete">

<span class="glyphicon glyphicon-remove" aria-hidden="true"></span>删除

</button>

</div>

<table id="dginfo"></table>

</div>

@section scripts{

<script>

var Id = $.getQueryString('Id');

var vm = new Vue({

el: '#v-app',

mounted: function () {

this.init\_datagrid();

},

methods: {

init\_datagrid: function () {

$('#dginfo').dataGrid({

url: '/Plat/Review/GetDataGrid',

sortOrder: "desc",

search: true,

sortName: 'Id',

queryParams: function (params) {

params.Email = params.search;

params.AId = Id;

return params;

},

columns: [{ checkbox: true },

{ field: 'Title', title: '文章标题',align: 'center' },

{

field: 'Text', title: '评论的内容', sortable: true, align: 'center', formatter: function (value, row) {

return com.content(value);

}

},

{ field: 'AuEmail', title: '邮箱', sortable: true },

{ field: 'AuName', title: '名字', sortable: true },

{

field: 'IsAduit', title: '状态', sortable: true, formatter: function (value, row) {

return com.format\_str('<span class="label label-{0}">{1}</span>', value == true ? "success" : "danger", value == true ? "审核通过" : "已拉黑");

}

},

{ field: 'CreatorTime', title: '评论时间', sortable: true },

{ field: 'Ip', title: 'Ip地址', sortable: true, align: 'center' },

{ field: 'Agent', title: '浏览器环境', align: 'center' },

],

callback: function () {

$('.pull-right.search .form-control').attr('placeholder','输入邮箱');

}

});

},

btn\_refresh: function () {

$('#dginfo').bootstrapTable("refresh");

},

btn\_aduit: function () {

var id = com.get\_selectid('#dginfo');

if (id == 0) {

$.layerMsg('请选择你要审核的评论！');

return;

}

this.editinfo(true, id);

},

btn\_black: function () {

var id = com.get\_selectid('#dginfo');

if (id == 0) {

$.layerMsg('请选择你要拉黑的评论！');

return;

}

this.editinfo(false, id);

},

btn\_delete: function () {

com.delete('/Plat/Review/Delete', '#dginfo');

},

editinfo: function (isaduit, id) {

com.ajax({

url: '/Plat/Review/Aduit',

data: { 'IsAduit': isaduit, 'Id': id },

showMsg: true,

success: function () {

vm.btn\_refresh();

}

})

}

}

});

</script>

}

namespace Mock.Luo.Areas.Plat.Controllers

{

public class ReviewController : BaseController

{

// GET: Plat/Review

#region Constructor

private readonly IReviewRepository \_reviewRepositroy;

private readonly IArticleRepository \_articleRepository;

private readonly IRedisHelper \_redisHelper;

private readonly IMailHelper \_imailHelper;

public ReviewController(IReviewRepository reviewRepositroy,

IArticleRepository articleRepository,

IMailHelper \_imailHelper,

IRedisHelper \_redisHelper)

{

this.\_reviewRepositroy = reviewRepositroy;

this.\_articleRepository = articleRepository;

this.\_redisHelper = \_redisHelper;

this.\_imailHelper = \_imailHelper;

}

#endregion

/// <summary>

/// 得到最新的10条评论的数据

/// </summary>

/// <returns></returns>

public ActionResult GetRecentReView()

{

var rows = \_reviewRepositroy.GetRecentReview(10);

return Result(rows);

}

/// <summary>

/// 后台：评论内容

/// </summary>

/// <param name="pag"></param>

/// <param name="Email"></param>

/// <param name="AId"></param>

/// <returns></returns>

public ActionResult GetDataGrid(Pagination pag, string Email = "", int AId = 0)

{

return Result(\_reviewRepositroy.GetDataGrid(ExtLinq.True<Review>(), pag, Email, AId));//所有

}

/// <summary>

/// 审核、拉黑评论

/// </summary>

/// <param name="IsAduit"></param>

/// <param name="Id"></param>

/// <returns></returns>

public ActionResult Aduit(bool IsAduit, int Id)

{

Review entity = new Review { Id = Id, IsAduit = IsAduit };

entity.Modify(Id);

\_reviewRepositroy.Update(entity, "IsAduit", "LastModifyUserId", "LastModifyTime");

return Success(IsAduit ? "审核成功！" : "拉黑成功！");

}

/// <summary>

/// 前端：文章对应的评论分页数据

/// </summary>

/// <param name="pag">分页</param>

/// <param name="AId">文章ID</param>

/// <returns></returns>

[Skip]

public ActionResult GetReviewGrid(Pagination pag, int AId)

{

if (pag.sort.IsNullOrEmpty())

{

pag.sort = "Id";

}

if (pag.order.IsNullOrEmpty())

{

pag.order = "desc";

}

if (pag.limit > 10)

{

pag.limit = 10;

}

//审核通过的

return Result(\_reviewRepositroy.GetDataGrid(u => u.IsAduit == true, pag, "", AId));

}

# region 前台用户评论文章

/// <summary>

/// 前台：用户评论文章

/// </summary>

/// <param name="reViewModel"></param>

/// <returns></returns>

[Skip]

public ActionResult Add(Review reViewModel)

{

if (reViewModel.AuEmail.IsNullOrEmpty())

{

return Error("Email不能为空！");

}

else if (!Validate.IsEmail(reViewModel.AuEmail))

{

return Error("邮箱格式不正确！");

}

if (reViewModel.AuName.IsNullOrEmpty())

{

return Error("用户昵称不能为空！");

}

if (!ModelState.IsValid)

{

return Error(ModelState.Values.Where(u => u.Errors.Count > 0).FirstOrDefault().Errors[0].ErrorMessage);

}

OperatorProvider op = OperatorProvider.Provider;

//未登录状态下，将生成一个随机头像

if (op.CurrentUser == null)

{

reViewModel.Avatar = "/Content/user/" + new Random(DateTime.Now.Second).Next(1, 361) + ".png";

}

else

{

}

reViewModel.Create();

reViewModel.System = Net.GetOSNameByUserAgent(Request.UserAgent);

reViewModel.UserHost = Net.Host;

reViewModel.Ip = Net.Ip;

reViewModel.GeoPosition = Net.GetLocation(reViewModel.Ip);

reViewModel.Agent = Net.Browser;

reViewModel.IsAduit = true;

\_reviewRepositroy.Insert(reViewModel);

var artEntity = \_articleRepository.IQueryable(u => u.Id == reViewModel.AId).FirstOrDefault();

artEntity.CommentQuantity += 1;

\_articleRepository.Update(artEntity, "CommentQuantity");

\_redisHelper.KeyDeleteAsync(string.Format(ConstHelper.Review, "GetRecentReview"));

return Success("吐槽成功");

}

#endregion

}

}

public interface IReviewRepository : IRepositoryBase<Review>

{

/// <summary>

/// 得到最新的count条回复信息 | 缓存

/// </summary>

/// <returns>最新回复消息</returns>

List<ReplyDto> GetRecentReview(int count);

/// <summary>

/// 根据条件得到文章的评论分页数据，默认是全部,可根据文章id查看评论内容

/// </summary>

/// <param name="pag">分页条件</param>

/// <param name="Email">邮箱</param>

/// <param name="AId">文章主键</param>

/// <returns>DataGrid实体</returns>

DataGrid GetDataGrid(Expression<Func<Review, bool>> predicate, Pagination pag, string Email, int AId);

}

/// <summary>

/// 仓储实现层 ReviewRepository

/// </summary>]

public class ReviewRepository : RepositoryBase<Review>, IReviewRepository

{

#region Constructor

private readonly IRedisHelper \_iRedisHelper;

public ReviewRepository(IRedisHelper iRedisHelper)

{

\_iRedisHelper = iRedisHelper;

}

#endregion

#region 根据条件得到文章的评论分页数据，默认是全部,可根据文章id查看评论内容

/// <summary>

/// 根据条件得到文章的评论分页数据,可根据文章id查看评论内容

/// </summary>

/// <param name="predicate"></param>

/// <param name="pag"></param>

/// <param name="Email"></param>

/// <param name="AId"></param>

/// <returns></returns>

public DataGrid GetDataGrid(Expression<Func<Review, bool>> predicate, Pagination pag, string Email, int AId)

{

predicate = predicate.And(u => u.DeleteMark == false

&& (Email == "" || u.AuEmail.Contains(Email))

&& (AId == 0 || u.AId == AId));

var reviewList = base.IQueryable(u => u.AId == AId).Select(u => new

{

u.PId,

u.Id,

u.AuName

}).ToList();

var dglist = base.IQueryable(predicate).Where(pag).Select(u=>new {

u.Article.Title,

Avatar = u.AppUser == null ? u.Avatar : u.AppUser.Avatar,

u=u

}).ToList().Select(r => new

{

r.Title,

r.Avatar,

r.u.Id,

r.u.AId,

PName = reviewList.Where(s => s.Id == r.u.PId).Select(s => s.AuName).FirstOrDefault(),

r.u.PId,

r.u.Text,

r.u.Ip,

r.u.Agent,

r.u.System,

r.u.GeoPosition,

r.u.UserHost,

r.u.AuName,

r.u.AuEmail,

r.u.IsAduit,

r.u.CreatorTime

}).ToList();

return new DataGrid { rows = dglist, total = pag.total };

}

#endregion

#region 得到最新的count条回复信息

public List<ReplyDto> GetRecentReview(int count)

{

return \_iRedisHelper.UnitOfWork(string.Format(ConstHelper.Review, "GetRecentReview"), () =>

{

return this.IQueryable().OrderByDescending(u => u.Id).Take(count).Select(u => new

{

u.Article.Title,

Avatar = u.AppUser != null ? u.AppUser.Avatar : u.Avatar,

u

}).ToList().Select(r => new ReplyDto

{

Id = r.u.Id,

AId = r.u.AId,

ArticleTitle = r.Title,

Text = r.u.Text,

NickName = r.u.AuName,

Avatar = r.Avatar

}).ToList();

});

}

#endregion

}

# 3.博客前台

## 首页

@{

Layout = "~/Views/Shared/\_LayoutBlog.cshtml";

ViewBag.Title = "、天上有木月 个人博客-小小程序员，信仰低调！";

}

@section css{

<style>

</style>

}

<div class="container" style="margin-top:80px;" id="v-app">

<div class="row">

<div class="col-xs-12 col-md-8">

<!--轮播图片-->

<div id="carousel-example-generic" class="carousel slide" data-ride="carousel">

<!-- Indicators -->

<ol class="carousel-indicators">

<li data-target="#carousel-example-generic" data-slide-to="0" class="active"></li>

<li data-target="#carousel-example-generic" data-slide-to="1"></li>

<li data-target="#carousel-example-generic" data-slide-to="2"></li>

</ol>

<!-- Wrapper for slides -->

<div class="carousel-inner" role="listbox">

<div class="item active">

<img src="~/Content/Images/1.jpg">

<div class="carousel-caption">

<div class="banner-word">

<h4>个人博客</h4>

<p>渡人如渡己，渡已，亦是渡人</p>

</div>

</div>

</div>

<div class="item">

<img src="~/Content/Images/2.jpg">

<div class="carousel-caption">

<div class="banner-word">

<h4> 、天上有木月</h4>

<p>-“当你发现自己的才华撑不起野心时，就请安静下来学习吧。”</p>

</div>

</div>

</div>

<div class="item">

<img src="~/Content/Images/3.jpg">

<div class="carousel-caption">

<div class="banner-word">

<h4>、天上有木月 </h4>

<p>-“你如今的气质里，藏着你走过的路，读过的书，和爱过的人。”</p>

</div>

</div>

</div>

</div>

<!-- Controls -->

<a class="left carousel-control" href="#carousel-example-generic" role="button" data-slide="prev">

<span class="glyphicon glyphicon-chevron-left" aria-hidden="true"></span>

<span class="sr-only">Previous</span>

</a>

<a class="right carousel-control" href="#carousel-example-generic" role="button" data-slide="next">

<span class="glyphicon glyphicon-chevron-right" aria-hidden="true"></span>

<span class="sr-only">Next</span>

</a>

</div>

<!--main-inner start-->

<div class="main-inner">

@{

List<ArtDetailDto> LatestArticles = ViewData["LatestArticles"] as List<ArtDetailDto>;

foreach (var item in LatestArticles)

{

<div class="flbox">

<article class="post">

<figure class="thumbnail box-hide box-show">

<a href="/App/Detail/@item.Id">

<img src="@item.thumbnail" alt="@item.Title">

</a>

</figure>

<span class="new-ico">NEW</span>

<header class="entry-header">

<h4 class="entry-title">

<a href="/App/Detail/@item.Id">@item.Title</a>

</h4>

</header>

<div class="entry-content">

<span class="entry-meta">

<span class="date" data-toggle="tooltip" title="@item.CreatorTime">

<i class="fa fa-clock-o"></i>

&nbsp;@item.TimeSpan

</span>

@if (item.TypeCode.IsNotNullOrEmpty())

{

<span class="cat">

<i class="fa fa-folder"></i>&nbsp;

<a href="/app/category?category=@item.TypeCode" target="\_blank">@item.TypeName</a>

</span>

}

<span class="views">

<i class="fa-user fa">

</i>&nbsp;

<a href="/@item.CreatorUserId/" data-toggle="tooltip" title="由 @item.NickName 发布">

@item.NickName

</a>

</span> <span class="comment">

<i class="fa fa-comment">

</i>&nbsp;

<a href="/App/Detail/@item.Id#author-comment">@item.CommentQuantity 条评论</a>

</span>

</span>

<br>

<div class="archive-content" >

@item.Excerpt

@\* 摘要 \*@

</div>

@\*<script>

var content = decodeURIComponent('@item.Content');

$('.content-@item.Id').html(content);

</script>\*@

<div class="clear">

</div>

</div>

</article>

</div>

}

}

</div>

<!--main-inner end-->

</div>

<div class="col-xs-12 col-md-4">

<div class="author-info-widget">

<div class="author-info-widget-img">

<img src="~/Content/Images/mo.jpg" class="img-responsive img-circle center-block" alt="侧栏个人头像" style="width:120px;">

</div>

<div class="text-center">

<h4>、天上有木月</h4>

<ul class="clearfix">

<li>

<a href="http://weibo.com/1820059022/profile?topnav=1&wvr=6" target="\_blank" data-toggle="tooltip" title="微博:、如莫" rel="external nofollow"><i class="fa fa-weibo fa-lg"></i></a>

</li>

<li>

<a href="https://github.com/luoyunchong" target="\_blank" data-toggle="tooltip" title="github：Mock.Luo"><i class="fa fa-github fa-lg"></i></a>

</li>

<li>

<a href="javascript:void(0);" target="\_blank" data-toggle="tooltip" v-on:click="show\_info('weixin.png')" title="微信:点我扫码"><i class="fa fa-weixin fa-lg"></i></a>

</li>

<li>

<a href="javascript:void(0);" target="\_blank" data-toggle="tooltip" v-on:click="show\_info('tim.jpg')" title="QQ:710277267"><i class="fa fa-qq fa-lg"></i></a>

</li>

</ul>

</div>

<div class="author-info-widget-words-main">

<span>-“你如今的气质里，藏着你走过的路，读过的书，和爱过的人”;</span>

</div>

<div class="author-info-widget-marker"></div>

</div>

@{Html.RenderPartial("TagHtml", ViewData["Tag"]); }

@using Mock.Data.Models;

<aside class="widget fadeInUp" data-wow-delay="0.3s">

<h3 class="widget-title">点击排行</h3>

<div id="hot\_article" class="hot\_article">

<ul>

@{

List<Article> HotArticles = ViewData["HotArticles"] as List<Article>;

string li\_number = "li-number";

int i = 0;

foreach (var item in HotArticles)

{

if (i == 3)

{

li\_number += "s";

}

i++;

<li>

<span class="hotp-title"><span class="@li\_number">@i</span><a href="/App/Detail/@item.Id" rel="bookmark" style="margin-left: 0px;">@item.Title</a></span>

</li>

}

}

</ul>

</div>

<div class="clear"></div>

</aside>

@{Html.RenderPartial("LatestArticles", ViewData["LatestArticles"]); }

@using Mock.Data.Dto;

<aside class="widget fadeInUp" data-wow-delay="0.3s">

<h3 class="widget-title">最新吐槽</h3>

<div class="hot\_article message-widget">

<ul>

@{

List<ReplyDto> SpitslotList = ViewData["SpitslotList"] as List<ReplyDto>;

foreach (var item in SpitslotList)

{

<li>

<a href="/App/Detail/@item.AId#anchor-comment-@item.Id" data-toggle='tooltip' title="@item.ArticleTitle" rel="external nofollow">

<img alt="" src="@item.Avatar" class="avatar avatar-64" width="64" height="64"> <span class="comment\_author"><strong>@item.NickName</strong></span>

<span v-html="com.decodeText('@item.Text')"></span>

</a>

</li>

}

}

</ul>

</div>

<div class="clear"></div>

</aside>

<aside class="widget fadeInUp" data-wow-delay="0.3s" id="divStatistics">

<h3 class="widget-title">站点信息</h3>

@{ SiteStatistics site = ViewData["Site"] as SiteStatistics;}

<ul class="divStatistics">

<li>文章总数:@site.ArticleCount</li>

<li>点赞总数:@site.PointViewCount</li>

<li>类型总数:@site.ArticleTypeCount</li>

<li>标签总数:@site.TagCount</li>

<li>评论总数:@site.ReplyCount</li>

<li>浏览总数:@site.ViewHitCount</li>

</ul>

<div class="clear"></div>

</aside>

</div>

</div>

<div class="row">

@{ List<ArtDetailDto> FellLifeList = ViewData["FellLife"] as List<ArtDetailDto>;

ArticleDto fell = new ArticleDto { };

if (FellLifeList.Count > 0)

{

fell = FellLifeList[0];

}

}

@if (FellLifeList.Count > 0)

{

<div class="col-md-6">

<div class="cat-box fadeInUp" data-wow-delay="0.3s">

<h3 class="cat-title"><a href="/App/Category?category=@fell.TypeCode"><i class="fa fa-heartbeat"></i>@fell.TypeName<i class="fa fa-angle-right"></i></a></h3>

<div class="clear"></div>

<div class="cat-site">

<h4 class="entry-title"><i class="fa fa-angle-right"></i><a href="/App/Detail/@fell.Id" rel="bookmark">@fell.Title</a></h4> <figure class="thumbnail"><a href="/App/Detail/@fell.Id"><img src="@fell.thumbnail" alt="@fell.Title"></a></figure>

<div class="cat-main">

@Html.Raw(fell.Excerpt)

</div>

<div class="clear"></div>

<ul class="cat-list">

@{

for (int j = 1; j < FellLifeList.Count; j++)

{

<li class="list-date">@FellLifeList[j].CreatorTime.ToDateTime().ToString("yyyy/MM/dd")</li>

<li class="list-title"><i class="fa fa-angle-right"></i><a href="/App/Detail/@FellLifeList[j].Id">@FellLifeList[j].Title</a></li>

}

}

</ul>

</div>

</div>

</div>

}

@{ List<ArtDetailDto> funcList = ViewData["JustFun"] as List<ArtDetailDto>;

ArticleDto func = new ArticleDto { };

if (funcList.Count > 0)

{

func = funcList[0];

}

}

@if (funcList.Count > 0)

{

<div class="col-md-6">

<div class="cat-box wow fadeInUp" data-wow-delay="0.3s">

<h3 class="cat-title"><a href="/App/Category?category=@fell.TypeCode"><i class="fa fa-heartbeat"></i>轻松时刻<i class="fa fa-angle-right"></i></a></h3>

<div class="clear"></div>

<div class="cat-site">

<h4 class="entry-title"><i class="fa fa-angle-right"></i><a href="/app/detail/@func.Id" rel="bookmark">@func.Title</a></h4><figure class="thumbnail"><a href="/app/detail/@func.Id"><img src="@func.thumbnail" alt="@func.Title"></a></figure>

<div class="cat-main">

@Html.Raw(func.Excerpt)

</div>

<div class="clear"></div>

<ul class="cat-list">

@{

for (int j = 1; j < funcList.Count; j++)

{

<li class="list-date">@funcList[j].CreatorTime.ToDateTime().ToString("yyyy/MM/dd")</li>

<li class="list-title" ><i class="fa fa-angle-right"></i><a href="/App/Detail/@funcList[j].Id">@funcList[j].Title</a></li>

}

}

</ul>

</div>

</div>

</div>

}

</div>

</div>

@section scripts{

<script>

$(function () {

$('.scroll-c').css('display', 'none');

});

var vm = new Vue({

el: '#v-app',

data: {},

mounted: function () {

this.init\_tooltip()

},

methods: {

get\_article: function () {

com.ajax({

url: '/Plat/Article/GetDataGrid'

});

},

init\_tooltip: function () {

$("[data-toggle='tooltip']").tooltip();

},

show\_info: function (img) {

var img\_url = '/Content/Images/' + img ;

var json = {

"title": "", //相册标题

"id": 123, //相册id

"start": 0, //初始显示的图片序号，默认0

"data": [ //相册包含的图片，数组格式

{

"alt": img,

"src": img\_url, //原图地址

"thumb": img\_url //缩略图地址

}

]

};

layer.photos({

photos: json

, anim:0 //0-6的选择，指定弹出图片动画类型，默认随机（请注意，3.0之前的版本用shift参数）

});

}

}

});

</script>

}

TagHtml.

@{ Layout = null;}

@\* 标签分部视图 \*@

@using Mock.Code;

<aside class="widget fadeInUp" data-wow-delay="0.3s">

<h3 class="widget-title">热门标签</h3>

<div class="tagcloud">

@{

foreach (var i in ViewData["Tag"] as List<TreeSelectModel>)

{

<a href="/App/Category?tag=@i.data">@i.text</a>

}

}

</div>

<div class="clear"></div>

</aside>

Controller中。

#region 博客首页

public override ActionResult Index()

{

//标签

List<TreeSelectModel> tagItemsList = \_itemsDetailRepository.GetCombobox("Tag");

tagItemsList.RemoveAt(0);

ViewData["Tag"] = tagItemsList;

//最新文章

List<ArtDetailDto> LatestArticles = \_articleRepository.GetRecentArticle(5);

ViewData["LatestArticles"] = LatestArticles;

//最热文章

List<Article> HotArticles = \_articleRepository.GetHotArticle(8);

ViewData["HotArticles"] = HotArticles;

//吐槽

List<ReplyDto> SpitslotList = \_reviewRepository.GetRecentReview(8);

ViewData["SpitslotList"] = SpitslotList;

//统计

SiteStatistics site = \_articleRepository.GetSiteData();

ViewData["Site"] = site;

//轻松时刻 | 缓存

ViewData["JustFun"] = \_iredisHelper.UnitOfWork(string.Format(ConstHelper.App, "JustFun"), () =>

{

List<ArtDetailDto> justFunList = \_articleRepository.GetArticleList(\_articleRepository.IQueryable(u => u.DeleteMark == false && u.ItemsDetail.ItemCode == CategoryCode.justfun.ToString())).OrderByDescending(u => u.Id).Take(5).ToList();

if (justFunList.Count > 0)

{

justFunList[0].Content = Server.UrlDecode(justFunList[0].Content);

}

return justFunList;

});

//人生感悟 | 缓存

ViewData["FellLife"] = \_iredisHelper.UnitOfWork(string.Format(ConstHelper.App, "FellLife"), () =>

{

List<ArtDetailDto> feLifeList = \_articleRepository.GetArticleList(\_articleRepository.IQueryable(u => u.DeleteMark == false && u.ItemsDetail.ItemCode == CategoryCode.feelinglife.ToString())).OrderByDescending(u => u.Id).Take(5).ToList();

if (feLifeList.Count > 0)

{

feLifeList[0].Content = Server.UrlDecode(feLifeList[0].Content);

}

return feLifeList;

});

return base.Index();

}

#endregion

## 详情页+评论

@using Mock.Data.Dto

@{

ArtDetailDto entry = ViewData["ArticleDto"] as ArtDetailDto;

Layout = "~/Views/Shared/\_LayoutBlog.cshtml";

ViewBag.Title = @entry.Title;

}

@using Mock.Data;

<div class="container" style="margin-top:80px;" id="v-app">

<div class="row">

<div class="col-md-8 col-xs-12">

<div class="navcates">

<i class="fa fa-home"></i>当前位置：<a href="/" target="\_blank" title="entry.Title">首页</a><i class="fa fa-angle-right"></i>

@if (entry.TypeName.IsNotNullOrEmpty())

{

<a href="/App/Category?category=@entry.TypeCode" target="\_blank" title="@entry.TypeName">@entry.TypeName</a><i class="fa fa-angle-right"></i>

}

<a href="/App/Detail/@entry.Id" rel="bookmark" target="\_blank" title="@entry.Title">@entry.Title</a>

</div>

<div class="content single-content">

<header class="post-head">

<h4><a href="/App/Detail/@entry.Id" target="\_blank" title="正在阅读 @entry.Title">@entry.Title</a></h4>

<div class="info left">

<!-- 文章信息 -->

@if (entry.TypeName.IsNotNullOrEmpty())

{

<span class="category"><i class="fa fa-list-alt"></i><a href="/app/category?category=@entry.TypeCode">@entry.TypeName</a></span>

}

<span class="date"><i class="fa fa-clock-o"></i>@entry.CreatorTime</span>

<span class="view"><i class="fa fa-eye"></i>浏览(@entry.ViewHits)</span>

<span class="comment"><i class="fa fa-comment"></i><a href="#review-form">评论(@entry.CommentQuantity)</a></span>

</div>

</header>

<div class="entry">

@Html.Raw(entry.Content)

<div id="social" v-cloak>

<div class="social-main">

<span class="like">

<a href="javascript:void(0);" v-on:click="point\_art" title="点赞" class="favorite done">

<i class="fa fa-thumbs-up"></i>赞

**{{**pointQuantity**}}**

</a>

</span>

<div class="shang-p">

<div class="shang-empty"><span></span></div>

<span class="tipso\_style">

<span class="shang-s"><a title="赞助本站">赏</a></span>

</span>

</div>

<div class="share-sd">

<span class="share-s"><a href="javascript:void(0)" id="share-s" title="分享"><i class="fa fa-share-alt"></i>分享</a></span>

<div id="share" style="display: none;">

<ul class="bdsharebuttonbox bdshare-button-style1-16" data-bd-bind="1506151460593">

<li><a title="分享到QQ空间" class="fa fa-qq" data-cmd="qzone" onclick="return false;" href="#"></a></li>

<li><a title="分享到新浪微博" class="fa fa-weibo" data-cmd="tsina" onclick="return false;" href="#"></a></li>

<li><a title="分享到微信" class="fa fa-weixin" data-cmd="weixin" onclick="return false;" href="#"></a></li>

</ul>

</div>

</div>

<div class="clear"></div>

</div>

</div>

</div>

<div class="tags">

标签：

</div>

<div class="authorarea">

<div class="authorinfo">

<div class="author-avater">

<img class="avaterimg" alt="@entry.NickName" title="作者：@entry.NickName " src="@entry.Avatar" height="50" width="50">

</div>

<div class="author-des">

<div class="author-meta">

<span class="post-author-name"><a href="" title="由 @entry.NickName 发布" rel="author">@entry.NickName</a></span>

@\*<span class="post-author-tatus"><a href="/app/author-1" target="\_blank">258篇文章</a></span>\*@

<template v-for="item in ArtTag">

<span class="post-author-weibo"><a :href="'/App/Category?tag='+item.code" target="\_blank" v-text="item.text"></a></span>

</template>

</div>

<p class="author-description">@entry.PersonSignature </p>

</div>

</div>

</div>

@\* 上下一个文章 \*@

<div class="row nav-single">

@if (entry.PreviousPage != null)

{

<a href="/App/Detail/@entry.PreviousPage.Id">

<div class="col-md-6 col-xs-6 text-center meta-nav">

<span>

<i class="fa fa-angle-left"></i> 上一篇

</span>

<br />

@entry.PreviousPage.text

</div>

</a>

}

else

{

<a href="javascript:void(0);">

<div class="col-md-6 col-xs-6 text-center meta-nav">

<span>

<i class="fa fa-angle-left"></i>到起点了

</span>

<br />

你点我也没有用!

</div>

</a>

}

@if (entry.NextPage != null)

{

<a href="/App/Detail/@entry.NextPage.Id">

<div class="col-md-6 col-xs-6 text-center meta-nav">

<span>

下一篇<i class="fa fa-angle-right"></i>

</span>

<br />

@entry.NextPage.text

</div>

</a>

}

else

{

<a href="javascript:void(0);">

<div class="col-md-6 col-xs-6 text-center meta-nav">

<span>

已经到终点了<i class="fa fa-angle-right"></i>

</span>

<br />你点我也没有用!

</div>

</a>

}

</div>

<!--推荐文章-->

<div class="row">

<div class="col-md-6">

<div class="cat-box fadeInUp" data-wow-delay="0.3s">

<h3 class="cat-title"><a href="/App/Category?category=Relate" title=""><i class="fa fa-heartbeat"></i>相关文章<i class="fa fa-angle-right"></i></a></h3>

<div class="cat-site">

<ul class="cat-list">

<template v-for="item in RelateArt">

<li class="list-date" v-text="item.code"></li>

<li class="list-title"><i class="fa fa-angle-right"></i><a :href="'/App/Detail/'+item.Id" v-text="item.text"></a></li>

</template>

</ul>

</div>

</div>

</div>

<div class="col-md-6">

<div class="cat-box fadeInUp" data-wow-delay="0.3s">

<h3 class="cat-title"><a href="/App/Category?category=Random" title=""><i class="fa fa-heartbeat"></i>随机文章<i class="fa fa-angle-right"></i></a></h3>

<div class="cat-site">

<ul class="cat-list">

<template v-for="item in RandomArt">

<li class="list-date" v-text="item.code"></li>

<li class="list-title"><i class="fa fa-angle-right"></i><a :href="'/App/Detail/'+item.Id" rel="bookmark" v-text="item.text"></a></li>

</template>

</ul>

</div>

</div>

</div>

</div>

<!---评论-->

<div class="scroll-comments">

<div class="comment-respond">

<h3> <i class="fa fa-comments"></i>&nbsp;吐槽一下</h3>

<form id="review-form">

<input type="hidden" name="AId" v-model="reply.AId" />

<input type="hidden" name="PId" v-model="reply.PId">

<label>QQ</label><input class="form-control" maxlength="12" v-model="reply.QQ" name="QQ" type="text" placeholder="请输入您的QQ" />

<label for="AuName">昵称<strong>（必填)</strong></label> <input type="text" class="form-control" isvalid="yes" checkexpession="NotNull" name="AuName" v-model="reply.AuName" placeholder="输入你的昵称" errormsg="昵称" />

<label for="AuEmail">邮箱<strong>（必填)</strong></label><input type="text" class="form-control" isvalid="yes" checkexpession="EmailAndNotNull" name="AuEmail" v-model="reply.AuEmail" placeholder="请输入正确的email" errormsg="邮箱" />

<label for="PersonalWebsite">网址</label><input type="text" v-model="reply.PersonalWebsite" class="form-control" name="PersonalWebsite" placeholder="这里可以输入你的个人网站" />

<label>评论内容<strong>（必填)</strong></label>

<div class="">

<textarea class="layui-textarea layui-editor" name="layui\_edit" id="layui\_edit"></textarea>

</div>

</form>

</div>

<div class="text-right">

<input type="button" class="btn btn-default" v-on:click="cancel" v-show="reply.PId!=0" value="取消" />

<span>回复：</span>

<input type="text" v-model="reply.PName" value="" readonly="readonly" disabled class="form-control" style="width:100px;display:inline;" />

<input type="button" v-on:click="submit" value="提交吐槽" class="btn btn-submit" />

</div>

</div>

</div>

@\* 评论列表 \*@

<ol class="comment-list">

<li class="fadeInUp" v-if="review.length==0">

<div class="comment-body">

<div class="comment-author vcard">

<img class="avatar" src="~/Content/Images/120122x.jpg" style="display: block;">

<strong>

、天上有木月

</strong>

<span class="ua-info">

<span class="WB-OS">

<img src="http://www.99bsy.com/wp-content/themes/begin/show-useragent/browsers/chrome.png"

title="Chrome" alt="Chrome"> <img src="http://www.99bsy.com/wp-content/themes/begin/show-useragent/browsers/windows\_win7.png" title="windows 7"

alt="windows 7">

</span> 江苏 南京 联通

</span>

<span class="lv">博主</span>

<span class="comment-meta">

<a href="/App/Detail/1#comment-712"></a><br>

发表于 @DateTime.Now.ToString("")

<span class="floor">

&nbsp;

</span>

</span>

</div>

<p>没有人评论，好凄凉啊，快来评论吧！！！</p>

</div>

</li>

<template v-for="item in review">

<li class="fadeInUp">

<div v-bind:id="'author-comment-'+ item.Id" class="comment-body">

<div class="comment-author vcard">

<img class="avatar" v-bind:src="item.Avatar" style="display: block;">

<strong>

<a href="#">**{{**item.AuName**}}**</a>

</strong>

<span class="ua-info">

<span class="WB-OS">

<img src="" v-bind:title="item.Agent" v-bind:alt="item.Agent">

<img src="" v-bind:title="item.System" v-bind:alt="item.System">

</span>

**{{**item.GeoPosition**}}** **{{**item.UserHost**}}**

</span>

<span class="comment-meta">

<a href="/App/Detail/1#comment-712"></a><br>

<span class="reply"><a rel="nofollow" v-on:click="reply\_review(item.Id)" class="comment-reply-link" href="javascript:void(0)">回复</a></span>

发表于**{{**item.CreatorTime**}}**

<span class="floor">

&nbsp;

</span>

</span>

</div>

<p class="review-content">

<span class="at" v-show="item.PId!=0">@@<a v-bind:href="'#author-comment-'+ item.PId" v-text="item.PName"></a></span>

<span v-html="com.decodeText(item.Text)"></span>

</p>

</div>

</li>

</template>

</ol>

<div id="review-page" class="text-right"></div>

</div>

@{Html.RenderPartial("\_RightSidebar"); }

</div>

</div>

@section scripts{

<script src="~/Content/js/plugins/validator.js"></script>

<script>

$(function () {

$('.hot\_post a,.hot\_comment a,.hot\_article a,.random\_post a,.related\_post a,.post\_cat a,.widget\_recent\_entries a,.cat-list a,.tab-bd a').hover(function () {

$(this).stop().animate({ 'marginLeft': '20px' }, 0);

}, function () {

$(this).stop().animate({ 'marginLeft': '10px' }, 0);

});

$('#share-s').click(function () {

var type = $('#share').css('display');

if (type == 'none') {

type = 'block';

$('#share').css('display', type);

} else {

type = 'none';

$('#share').css('display', type);

}

});

});

var reply ={

AId: @ViewBag.AId,

PId: 0,

QQ: '',

AuName: '',

AuEmail: '',

PersonalWebsite: '',

Text: '',

PName: '博主',

isLogin:false

};

@{

OperatorProvider op = OperatorProvider.Provider;

string email="", website = "",auname="";

bool isLogin = false;

}

@if (op.CurrentUser == null)

{

}

else

{

email = op.CurrentUser.Email;

website = op.CurrentUser.PersonalWebsite;

auname = op.CurrentUser.NickName;

isLogin = true;

}

reply.AuEmail ='@email';

reply.PersonalWebsite = '@website';

reply.AuName = '@auname';

reply.isLogin = '@isLogin';

var vm = new Vue({

el: "#v-app",

data: {

AId: '@ViewBag.AId',

review: [],

limit: 5,

index: "",

reply: reply,

pointQuantity: '@entry.PointQuantity' == '' ? 0 : '@entry.PointQuantity',

RelateArt: [],

Category: [],

RecommendArt: [],

ArchiveFile: [],

RandomArt: [],

ArtTag: []

},

mounted: function () {

this.get\_relate\_list();

this.init\_datagrid();

this.init\_layedit();

},

methods: {

init\_layedit: function () {

window.setTimeout(function () {

vm.index = com.layEditor({

elem: '.layui-editor'

});

}, 1000);

},

init\_datagrid: function () {

var that = this;

this.get\_itemgrid({ 'limit': this.limit, 'offset': 0, 'AId': @ViewBag.AId }, function (json) {

that.layuipage(json.total);

});

},

layuipage: function (total) {

layui.laypage.render({

elem: 'review-page',

limit: vm.limit,

count: total,

curr: 1,

groups: 5,

layout: ['prev', 'page', 'next', 'skip'],

jump: function (obj, first) {

//首次不执行

if (!first) {

//1.跳转分页时也需要将查询参数携带给后台，才算真正的分页效果,

var data = {

'limit': vm.limit,

'offset': (obj.curr - 1) \* vm.limit,

'AId': '@ViewBag.AId'

};

vm.get\_itemgrid(data);

}

}

});

},

get\_itemgrid: function (data, callback) {

com.ajax({

url: '/Plat/Review/GetReviewGrid',

data: data,

success: function (json) {

vm.$data.review = json.rows;

vm.total = json.total;

if (typeof (callback) == 'function') {

callback(json);

}

vm.init\_hover();

}

})

},

init\_hover: function () {

window.setTimeout(function () {

$('.comment-list .fadeInUp').each(function () {

$(this).hover(function () {

$(this).find(".ua-info").css('display', 'inline');

}, function () {

$(this).find(".ua-info").css('display', 'none');

});

})

}, 1000);

},

submit: function () {

if (!$('#review-form').Validform()) {

return false;

}

var text = $('#layui\_edit').val();

if ($.trim(text) == "") {

$.layerMsg('请填写评论内容！', 'info');

return;

}

if ($.trim(text) != "" && $.trim(text).length > 500) {

$.layerMsg('评论内容太多了！', 'info');

return;

}

vm.reply.Text = com.escape(text);

com.ajax({

url: '/Plat/Review/Add',

data: vm.reply,

showMsg: true,

success: function (d) {

$('#layui\_edit').val("");

if (reply.isLogin == false) {

vm.reply.QQ = ""; vm.reply.AuName = ""; vm.reply.AuEmail = ""; vm.reply.PersonalWebsite = "";

}

vm.init\_datagrid();

}

});

},

reply\_review: function (Id) {

console.log(Id);

$('html,body').animate({

scrollTop: $('.scroll-comments').offset().top

}, 800);

$.each(vm.review, function (i, v) {

if (v.Id == Id) {

vm.reply.PName = v.AuName;

vm.reply.PId = v.Id;

return false;

}

});

},

cancel: function () {

vm.reply.PId = 0;

vm.reply.PName = "博主";

},

get\_relate\_list: function () {

com.ajax({

url: '/App/GetRelateList',

data: { Id: '@ViewBag.AId' },

showLoading: false,

success: function (data) {

vm.RelateArt = data.RelateArt;

vm.Category = data.Category;

vm.RecommendArt = data.RecommendArt;

vm.ArchiveFile = data.ArchiveFile;

vm.RandomArt = data.RandomArt;

vm.ArtTag = data.ArtTag;

}

});

},

point\_art: function () {

com.ajax({

url: '/Plat/PointArticle/Edit',

data: {

AId: '@ViewBag.AId'

},

success: function (d) {

$.procAjaxMsg(d, function () {

vm.pointQuantity++;

}, function () {

$.layerMsg(d.message, 'info');

});

}

});

}

}

});

</script>

}

#region 博客文章详情页

public override ActionResult Detail(int Id)

{

ArtDetailDto entry = \_articleRepository.GetOneArticle(Id);

if (entry == null) throw new ArgumentNullException("根据Id,我去查了，但文章就是未找到！");

//找到当前的上一个，下一个的文章

Expression<Func<Article, bool>> nextlambda = u => u.Id > Id && u.DeleteMark == false;

var next = \_articleRepository.IQueryable(nextlambda).FirstOrDefault();

if (next != null)

{

entry.NextPage = new BaseDto

{

Id = next.Id,

text = next.Title

};

}

Expression<Func<Article, bool>> previouslabmda = u => u.Id < Id && u.DeleteMark == false;

var pre = \_articleRepository.IQueryable(previouslabmda).OrderByDescending(u => u.Id).FirstOrDefault();

if (pre != null)

{

entry.PreviousPage = new BaseDto

{

Id = pre.Id,

text = pre.Title

};

}

IQueryable<Article> queryable = \_articleRepository.IQueryable(u => u.Id == Id);

\_articleRepository.Update(queryable, u => new Article

{

ViewHits = entry.ViewHits + 1

});

entry.Content = Server.UrlDecode(entry.Content);

ViewData["ArticleDto"] = entry;

ViewBag.AId = Id;

return base.View();

}

#endregion

## 用户点赞

public class PointArticleController : BaseController

{

// GET: Plat/PointArticle

private readonly IPointArticleRepository \_service;

private readonly IArticleRepository \_articleRepository;

public PointArticleController(IPointArticleRepository service, IArticleRepository \_articleRepository)

{

this.\_service = service;

this.\_articleRepository = \_articleRepository;

}

//根据文章id得到点赞人信息

public ActionResult GetDataGrid(int AId)

{

return Result(\_service.GetDataGrid(AId));

}

//用户点赞

public ActionResult Edit(PointArticle entry)

{

entry.AddTime = DateTime.Now;

entry.UserId = (int)op.CurrentUser.UserId;

var pointCount = \_service.IQueryable(u => u.UserId == entry.UserId && u.AId == entry.AId).Count();

//点当前文章已经被点赞时，用户再次点赞，

if (pointCount > 0)

{

return Error("你已经点过赞了！");

//\_service.Delete(u => u.UserId == entry.UserId && u.AId == entry.AId);

}

else

{

\_service.Insert(entry);

int? PointQuantity=\_articleRepository.IQueryable(r => r.Id == entry.AId).Select(r => r.PointQuantity).FirstOrDefault();

\_articleRepository.Update(new Article { Id = entry.AId, PointQuantity = PointQuantity + 1 }, "PointQuantity");

}

return Success("成功点赞");

}

}

## 用户评论

public class ReviewController : BaseController

{

// GET: Plat/Review

#region Constructor

private readonly IReviewRepository \_reviewRepositroy;

private readonly IArticleRepository \_articleRepository;

private readonly IRedisHelper \_redisHelper;

private readonly IMailHelper \_imailHelper;

public ReviewController(IReviewRepository reviewRepositroy,

IArticleRepository articleRepository,

IMailHelper \_imailHelper,

IRedisHelper \_redisHelper)

{

this.\_reviewRepositroy = reviewRepositroy;

this.\_articleRepository = articleRepository;

this.\_redisHelper = \_redisHelper;

this.\_imailHelper = \_imailHelper;

}

#endregion

/// <summary>

/// 得到最新的10条评论的数据

/// </summary>

/// <returns></returns>

public ActionResult GetRecentReView()

{

var rows = \_reviewRepositroy.GetRecentReview(10);

return Result(rows);

}

/// <summary>

/// 后台：评论内容

/// </summary>

/// <param name="pag"></param>

/// <param name="Email"></param>

/// <param name="AId"></param>

/// <returns></returns>

public ActionResult GetDataGrid(Pagination pag, string Email = "", int AId = 0)

{

return Result(\_reviewRepositroy.GetDataGrid(ExtLinq.True<Review>(), pag, Email, AId));//所有

}

/// <summary>

/// 审核、拉黑评论

/// </summary>

/// <param name="IsAduit"></param>

/// <param name="Id"></param>

/// <returns></returns>

public ActionResult Aduit(bool IsAduit, int Id)

{

Review entity = new Review { Id = Id, IsAduit = IsAduit };

entity.Modify(Id);

\_reviewRepositroy.Update(entity, "IsAduit", "LastModifyUserId", "LastModifyTime");

return Success(IsAduit ? "审核成功！" : "拉黑成功！");

}

/// <summary>

/// 前端：文章对应的评论分页数据

/// </summary>

/// <param name="pag">分页</param>

/// <param name="AId">文章ID</param>

/// <returns></returns>

[Skip]

public ActionResult GetReviewGrid(Pagination pag, int AId)

{

if (pag.sort.IsNullOrEmpty())

{

pag.sort = "Id";

}

if (pag.order.IsNullOrEmpty())

{

pag.order = "desc";

}

if (pag.limit > 10)

{

pag.limit = 10;

}

//审核通过的

return Result(\_reviewRepositroy.GetDataGrid(u => u.IsAduit == true, pag, "", AId));

}

# region 前台用户评论文章

/// <summary>

/// 前台：用户评论文章

/// </summary>

/// <param name="reViewModel"></param>

/// <returns></returns>

[Skip]

public ActionResult Add(Review reViewModel)

{

if (reViewModel.AuEmail.IsNullOrEmpty())

{

return Error("Email不能为空！");

}

else if (!Validate.IsEmail(reViewModel.AuEmail))

{

return Error("邮箱格式不正确！");

}

if (reViewModel.AuName.IsNullOrEmpty())

{

return Error("用户昵称不能为空！");

}

if (!ModelState.IsValid)

{

return Error(ModelState.Values.Where(u => u.Errors.Count > 0).FirstOrDefault().Errors[0].ErrorMessage);

}

OperatorProvider op = OperatorProvider.Provider;

//未登录状态下，将生成一个随机头像

if (op.CurrentUser == null)

{

reViewModel.Avatar = "/Content/user/" + new Random(DateTime.Now.Second).Next(1, 361) + ".png";

}

else

{

}

reViewModel.Create();

reViewModel.System = Net.GetOSNameByUserAgent(Request.UserAgent);

reViewModel.UserHost = Net.Host;

reViewModel.Ip = Net.Ip;

reViewModel.GeoPosition = Net.GetLocation(reViewModel.Ip);

reViewModel.Agent = Net.Browser;

reViewModel.IsAduit = true;

\_reviewRepositroy.Insert(reViewModel);

var artEntity = \_articleRepository.IQueryable(u => u.Id == reViewModel.AId).FirstOrDefault();

artEntity.CommentQuantity += 1;

\_articleRepository.Update(artEntity, "CommentQuantity");

\_redisHelper.KeyDeleteAsync(string.Format(ConstHelper.Review, "GetRecentReview"));

return Success("吐槽成功");

}

#endregion

}

## 分类页面

@{

Layout = "~/Views/Shared/\_LayoutBlog.cshtml";

ViewBag.Title = ViewBag.TypeName;

}

<div class="container" style="margin-top:80px;" id="v-app">

<div class="row">

<div class="col-md-8 col-xs-12">

<div class="navcates">

<i class="fa fa-home"></i>当前位置：<a href="/" target="\_blank">首页</a>

<i class="fa fa-angle-right"></i>

<span>@ViewBag.TypeName</span>

<i class="fa fa-angle-right" v-show="items&&items.length>0"></i>

<span v-show="items&&items.length>0">正文</span>

</div>

<!--main-inner start-->

<div class="main-inner">

<div class="error-404 not-found page" v-if="items&&items.length==0" v-cloak>

<h3 class="text-center">亲，该网页可能搬家了！<i class="fa fa-smile-o"></i></h3>

<div class="searchbar">

<span>

<input type="text" name="s" value="" class="form-control col-md-8" placeholder="输入搜索内容" />

<button type="button" class="btn btn-primary col-md-4">搜索</button>

</span>

</div>

</div>

<div class="flbox" v-for="item in items" v-show="items&&items.length>0" v-cloak>

<article class="post">

<figure class="thumbnail box-hide box-show">

<a :href="'/App/Detail/'+item.Id">

<img :src="item.thumbnail" :alt="item.Title">

</a>

</figure>

<span class="new-ico">NEW</span>

<header class="entry-header">

<h4 class="entry-title">

<a :href="'/App/Detail/'+item.Id" v-text="item.Title"></a>

</h4>

</header>

<div class="entry-content">

<span class="entry-meta">

<span class="date" data-toggle="tooltip" :title="item.CreatorTime">

<i class="fa fa-clock-o"></i>

&nbsp;**{{**item.TimeSpan**}}**

</span>

<span class="cat">

<i class="fa fa-folder"></i>&nbsp;

<a :href="'/App/Category?category='+item.TypeCode" v-text="item.TypeName" target="\_blank"></a>

</span>

<span class="views">

<i class="fa-user fa">

</i>&nbsp;

<a :href="'/App/User/'+item.CreatorUserId" v-text="item.NickName" data-toggle="tooltip" :title="'由'+item.NickName +'发布'">

</a>

</span> <span class="comment">

<i class="fa fa-comment">

</i>&nbsp;

<a :href="'/App/Detail/'+item.Id+'#author-comment'" v-text="item.CommentQuantity+'条评论'"> </a>

</span>

</span>

<br>

<div class="archive-content" v-text="item.Excerpt"> </div>

<div class="clear"></div>

</div>

</article>

</div>

</div>

<!--main-inner end-->

<div id="review-page" class="text-right" v-show="items&&items.length>0"></div>

</div>

@{Html.RenderPartial("\_RightSidebar"); }

</div>

</div>

@section scripts{

<script>

$(function () {

$('.scroll-c').css('display', 'none');

})

var vmJson = com.replaceSpce('@Html.Raw(ViewBag.ViewModel)');

var request = com.get\_params();

var vm = new Vue({

el: "#v-app",

data: {

limit: 10,

total: vmJson.total,

items: vmJson.rows,

RelateArt: [],

Category: [],

RecommendArt: [],

ArchiveFile: [],

RandomArt: [],

ArtTag:[]

},

mounted: function () {

this.init\_datagrid();

this.get\_relate\_list();

},

methods: {

init\_datagrid: function () {

var that = this;

that.layuipage(vmJson.total);

},

layuipage: function (total) {

var that= this;

layui.laypage.render({

elem: 'review-page',

limit: that.limit,

count: total,

curr: 1,

groups: 5,

layout: ['prev', 'page', 'next', 'skip'],

jump: function (obj, first) {

//首次不执行

if (!first) {

//1.跳转分页时也需要将查询参数携带给后台，才算真正的分页效果,

var data = {

'limit': that.limit,

'offset': (obj.curr - 1) \* that.limit,

'archive':request.archive,

'category': request.category,

'tag': request.tag

};

that.get\_itemgrid(data);

}

}

});

},

get\_itemgrid: function (data,callback) {

com.ajax({

url: '/Plat/Article/GetIndexGird',

data: data,

showLoading:false,

success: function (json) {

vm.$data.items = json.rows;

if (typeof (callback) == 'function') {

callback(json);

}

}

})

},

get\_relate\_list: function () {

com.ajax({

url: '/App/GetRelateList',

showLoading: false,

data: { Id:1 },

success: function (data) {

vm.RelateArt = data.RelateArt;

vm.Category = data.Category;

vm.RecommendArt = data.RecommendArt;

vm.ArchiveFile = data.ArchiveFile;

vm.RandomArt = data.RandomArt;

vm.ArtTag = data.ArtTag;

}

})

}

}

})

</script>

}

#region 分类文章|标签视图

/// <summary>

/// 分类文章|标签

/// </summary>

/// <param name="category"></param>

/// <param name="tag"></param>

/// <returns></returns>

public ActionResult Category(string category = "", string tag = "", string archive = "")

{

if (category.IsNullOrEmpty() && tag.IsNullOrEmpty() && archive.IsNullOrEmpty())

{

throw new ArgumentNullException("参数不正确！！!");

}

Pagination pag = new Pagination

{

sort = "Id",

order = "desc",

limit = 10,

offset = 0

};

DataGrid dg = \_articleRepository.GetCategoryTagGrid(pag, category, tag, archive);

string TypeName = "";

if (category.IsNotNullOrEmpty() || tag.IsNotNullOrEmpty())

{

TypeName = \_itemsDetailRepository.IQueryable(r => r.ItemCode == category || r.ItemCode == tag).Select(r => r.ItemName).FirstOrDefault();

}

else

{

TypeName = archive;

}

ViewBag.TypeName = TypeName.IsNullOrEmpty() ? "亲，您迷路了啊！|、天上有木月" : TypeName;

ViewBag.ViewModel = dg.ToJson();

return View();

}

#endregion

## 留言互动

@{

Layout = "~/Views/Shared/\_LayoutBlog.cshtml";

ViewBag.Title = "留言板";

}

<div class="container" style="margin-top:80px;" id="v-app">

<div class="row">

<div class="col-md-8 col-xs-12">

<div class="navcates">

<i class="fa fa-home"></i>当前位置：<a href="/" target="\_blank" title="entry.Title">首页</a><i class="fa fa-angle-right"></i>

<span title="留言互动">留言互动</span>

</div>

@\* 留言列表 \*@

<ol class="comment-list">

<template v-show="review.length==0">

<li class="fadeInUp" v-if="review.length==0">

<div class="comment-body">

<div class="comment-author vcard">

<img class="avatar" src="~/Content/Images/120122x.jpg" style="display: block;">

<strong>

、天上有木月

</strong>

<span class="ua-info">

<span class="WB-OS">

<img src="http://www.99bsy.com/wp-content/themes/begin/show-useragent/browsers/chrome.png"

title="Chrome" alt="Chrome"> <img src="http://www.99bsy.com/wp-content/themes/begin/show-useragent/browsers/windows\_win7.png" title="windows 7"

alt="windows 7">

</span> 江苏 南京 联通

</span>

<span class="lv">博主</span>

<span class="comment-meta">

<a href="/App/Detail/1#comment-712"></a><br>

发表于 @DateTime.Now.ToString("")

<span class="floor">

&nbsp;

</span>

</span>

</div>

<p>没有人留言，好凄凉啊，快来留言吧！！！</p>

</div>

</li>

</template>

<template v-for="item in review">

<li class="fadeInUp">

<div v-bind:id="'author-comment-'+ item.Id" class="comment-body">

<div class="comment-author vcard">

<img class="avatar" v-bind:src="item.Avatar" style="display: block;">

<strong>

<a href="#">**{{**item.AuName**}}**</a>

</strong>

<span class="ua-info">

<span class="WB-OS"><img src="http://www.99bsy.com/wp-content/themes/begin/show-useragent/browsers/chrome.png" v-bind:title="item.Agent" v-bind:alt="item.Agent"> <img src="http://www.99bsy.com/wp-content/themes/begin/show-useragent/browsers/windows\_win7.png" v-bind:title="item.System" v-bind:alt="item.System"> </span>**{{**item.GeoPosition**}}** **{{**item.UseNet**}}**

</span>

<span class="comment-meta">

<a href="/App/Detail/1#comment-712"></a><br>

<span class="reply"><a rel="nofollow" v-on:click="reply\_review(item.Id)" class="comment-reply-link" href="javascript:void(0)">回复</a></span>

发表于**{{**item.CreatorTime**}}**

<span class="floor">

&nbsp;

</span>

</span>

</div>

<p class="review-content">

<span class="at" v-show="item.PId!=0">@@<a v-bind:href="'#author-comment-'+ item.PId" v-text="item.PName"></a></span>

<span v-html="com.decodeText(item.Text)"></span>

</p>

</div>

</li>

</template>

</ol>

<div id="review-page" class="text-right"></div>

<div class="content single-content">

<!---留言-->

<div class="scroll-comments">

<div class="comment-respond">

<h3 class="text-center"> <i class="fa fa-comments"></i>&nbsp;快来留言吧</h3>

<form id="review-form">

<input type="hidden" name="AId" v-model="reply.AId" />

<input type="hidden" name="PId" v-model="reply.PId">

<label>QQ</label><input class="form-control" maxlength="12" v-model="reply.QQ" name="QQ" type="text" placeholder="请输入您的QQ" />

<label for="AuName">昵称<strong>（必填)</strong></label> <input type="text" class="form-control" isvalid="yes" checkexpession="NotNull" name="AuName" v-model="reply.AuName" placeholder="输入你的昵称" errormsg="昵称" />

<label for="AuEmail">邮箱<strong>（必填)</strong></label><input type="text" class="form-control" isvalid="yes" checkexpession="EmailAndNotNull" name="AuEmail" v-model="reply.AuEmail" placeholder="请输入正确的email" errormsg="邮箱" />

<label for="PersonalWebsite">网址</label><input type="text" v-model="reply.PersonalWebsite" class="form-control" name="PersonalWebsite" placeholder="这里可以输入你的个人网站" />

<label>留言内容<strong>（必填)</strong></label>

<div class="">

<textarea class="layui-textarea layui-editor" name="layui\_edit" id="layui\_edit"></textarea>

</div>

</form>

</div>

<div class="text-right">

<span v-show="reply.PId!=0">

<input type="button" class="btn btn-default" v-on:click="cancel" value="取消" />

<span>回复：</span>

<input type="text" v-model="reply.PName" value="" readonly="readonly" disabled class="form-control" style="width:100px;display:inline;" />

</span>

<input type="button" v-on:click="submit" value="提交留言" class="btn btn-submit" />

</div>

</div>

</div>

</div>

</div>

</div>

@section scripts{

<script src="~/Content/js/plugins/validator.js"></script>

<script>

var vmJson = com.replaceSpce('@Html.Raw(ViewBag.ViewModel)');

var vm = new Vue({

el: "#v-app",

data: {

limit: 10,

total: vmJson.total,

review: vmJson.rows,

index: "",

reply: {

PId: 0,

QQ: '',

AuName: '',

AuEmail: '',

PersonalWebsite: '',

Text: '',

PName: '博主'

}

},

mounted: function () {

this.init\_datagrid();

this.init\_layedit();

},

methods: {

init\_layedit: function () {

window.setTimeout(function () {

vm.index = com.layEditor({

elem: '.layui-editor'

});

}, 1000);

},

init\_datagrid: function () {

var that = this;

that.layuipage(vmJson.total);

},

layuipage: function (total) {

var that = this;

layui.laypage.render({

elem: 'review-page',

limit: that.limit,

count: total,

curr: 1,

groups: 5,

layout: ['prev', 'page', 'next', 'skip'],

jump: function (obj, first) {

//首次不执行

if (!first) {

//1.跳转分页时也需要将查询参数携带给后台，才算真正的分页效果,

var data = {

'limit': that.limit,

'offset': (obj.curr - 1) \* that.limit

};

that.get\_itemgrid(data);

}

}

});

},

get\_itemgrid: function (data, callback) {

com.ajax({

url: '/Plat/GuestBook/GetDataGrid',

data: data,

success: function (json) {

vm.$data.review = json.rows;

vm.total = json.total;

if (typeof (callback) == 'function') {

callback(json);

}

vm.init\_hover();

}

})

},

cancel: function () {

vm.reply.PId = 0;

vm.reply.PName = "博主";

},

submit: function () {

if (!$('#review-form').Validform()) {

return false;

}

var text = $('#layui\_edit').val();

if ($.trim(text) == "") {

$.layerMsg('请填写您的留言！', 'info');

return;

}

if ($.trim(text) != "" && $.trim(text).length > 500) {

$.layerMsg('留言的内容太多了！我这里接收不下啊！！', 'info');

return;

}

vm.reply.Text = com.escape(text);

var that = this;

com.ajax({

url: '/Plat/GuestBook/Add',

data: vm.reply,

showMsg: true,

success: function (d) {

$('#layui\_edit').val("");

vm.reply.QQ = ""; vm.reply.AuName = ""; vm.reply.AuEmail = ""; vm.reply.PersonalWebsite = "";

var data = {

'limit': that.limit,

'offset': 0

};

vm.get\_itemgrid(data);

}

});

},

init\_hover: function () {

window.setTimeout(function () {

$('.comment-list .fadeInUp').each(function () {

$(this).hover(function () {

$(this).find(".ua-info").css('display', 'inline');

}, function () {

$(this).find(".ua-info").css('display', 'none');

});

})

}, 1000);

},

reply\_review: function (Id) {

console.log(Id);

$('html,body').animate({

scrollTop: $('.scroll-comments').offset().top

}, 800);

$.each(vm.review, function (i, v) {

if (v.Id == Id) {

vm.reply.PName = v.AuName;

vm.reply.PId = v.Id;

return false;

}

});

},

}

})

</script>

}