“技术生活交流社交Web应用”软件开发文档

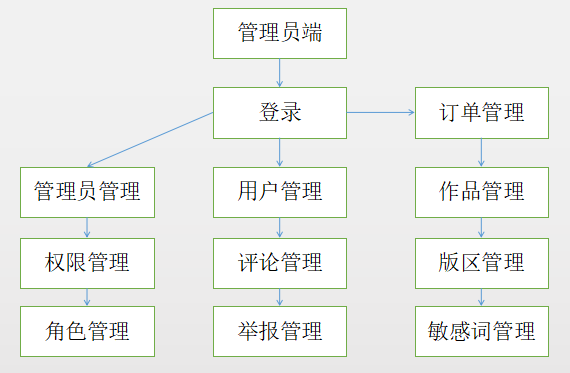
一、需求分析

本项目开发的目的：随着社会的进步和网络的快速发展，人们对精神生活的追求越来越高，渴望从外界获得技术生活方面的交流。我们的网站以网络为平台，用户可以在网站上查找、添加好友，上传自己的帖子（照片，知识技术分享，日志），发表对其他人帖子的评论、赞、收藏等。

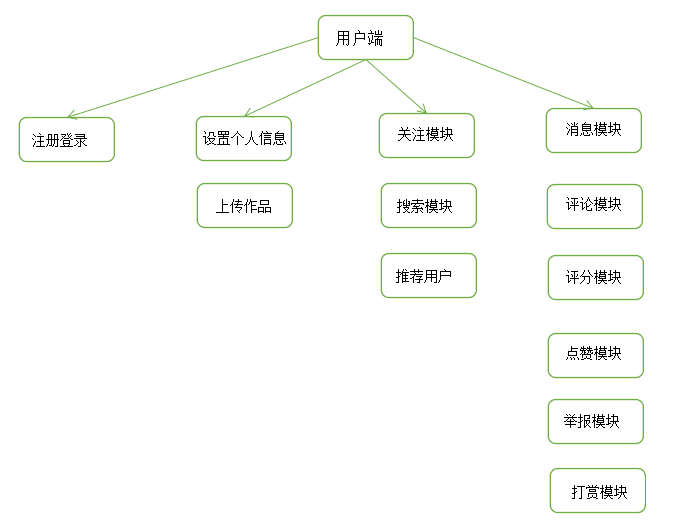
本项目具有的功能：本网站应用提供多个版区，用户根据喜爱的版区发布个人的技术分享或生活的作品，作品必须上传至少一张照片或视频(视频链接)，作品描述加上#号相当于加上标签。当然也可以浏览其他用户的作品，给作品打分、收藏作品、举报作品、评论作品、打赏作品(只作学习测试，不会开放接口)、点赞评论、举报评论，点赞高的评论会出现在流简介上，而作品的热度将会根据作品的发布时间、评分参与人数、评论参与人数、评分高低通过我们自定义的Feed流公式进行排序，在热流中优先显示。用户可以邮箱注册，或使用邮箱或注册ID和密码登录，完善和维护个人信息(上传头像、个人简介等)，修改密码，修改邮箱。可以关注/取消关注其他用户，查看个人关注帐号和被关注帐号，查看收藏的作品，查看站内信。同时提供搜索用户，搜索标签的功能。

二、概要设计

1.系统结构设计如图1图2



管理员端系统架构图-图2



用户端系统架构图-图2

2.数据设计：

数据库采用了MySQL关系型数据库和Redis非关系型缓存数据库，用户端和管理员端各自的前后端，用户端和管理员端共同操作MySQL和Redis数据库。

用户或管理员通过访问前端项目Frontend获取页面，Frontend通过用户和管理员的操作调用Backend的后端接口获取数据，Backend通过分层处理业务逻辑并根据Frontend的要求操作MySQL和Redis数据库且返回数据。系统结构设计图如图3：

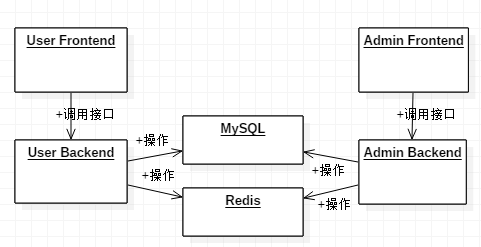


图3

由于引入Redis缓存数据库，根据业务逻辑在实现特定功能时会在Redis使用对应数据结构。

同样以用户模块作例子，在MySQL关系型数据库中，表的设计如下：

# User

CREATE TABLE `user`(

`user\_autoid` BIGINT AUTO\_INCREMENT,

`user\_id` VARCHAR(20),

`user\_email` VARCHAR(30),

`user\_status` INT,

`user\_pwd` VARCHAR(200),

`user\_gins` INT,

PRIMARY KEY (`user\_autoid`),

INDEX (`user\_email`),

INDEX (`user\_id`,`user\_pwd`),

INDEX (`user\_email`,`user\_pwd`)

)ENGINE=INNODB CHARSET utf8;

在Redis非关系型数据库中，涉及用户的结构如下：

Hash-

User:

key: user:[user\_id]

field: id, email, gins, avatar, follows, fans, posts, msgstatus//是否有未读消息

Set-

Follows-

key: follows:[user\_id]

value: //关注用户id

Fans-

key: fans:[user\_id]

value: //粉丝id

2.前后端交互接口设计：

本项目根据Restful　API风格设计前后端交互接口。根据业务逻辑管理员端涉及按条件的CRUD接口较多，用户端则设计了更多人性化接口。

管理员端大部分是直接增删改查的接口，此处列出用户管理接口作例子：

/user/add 新增用户

Data:{

id: String,

email: String,

pwd: String,

status: Integer,

gins: Integer

}

/user/del/{id} 删除用户

data:{

pathvariable:id:long

}

/user/get/{id} 查看用户

data:{

pathvariable:id:long

}

/user/list 条件查看用户列表

data: {

id: String,

email: String,

pwd: String,

status: Integer,

}

/user/mod 修改用户

data:{

id: String,

email: String,

pwd: String,

status: Integer,

gins: Integer

}

用户端接口偏向于业务逻辑设计，同样以举用户模块为例子：

/user/checkid/{id} 检查注册ID是否已被占用

data: {

pathvariable:id:String

}

/user/detail 查看个人详情页

data:{}

/user/moddetail 修改个人详情页

data:{

avatar:String,

brief:String,

website:String,

phone:String,

gender:char,

wechat:String

}

/user/modpwd 修改密码

data:{

oldPwd:String,

newPwd:String

}

/user/my 查看我的信息

data:{}

/user/register 注册

data:{

id:String,

email:String,

pwd:String

}

1. 详细设计
2. 功能设计

项目分为用户端模块和管理员端模块，本人具体实现用户端页面的以下内容：

1. 网站首页：设计属于我们小组的logo，头部的导航条随着鼠标的滚动总是停留在页面的最上方，提供搜索用户，搜索标签的功能。
2. 注册登录/注册页面：实现一个轮播图的特效。

var index=0;  
var timer=null;  
var pics=byId("banner").getElementsByTagName("div");  
var len=pics.length;  
function slideImg() {  
 /\*间歇调用\*/  
 timer=setInterval(function () {  
 index++;  
 if(index>=len){  
 index=0;  
 }  
 //切换图片  
 changeImg();  
 },3000);  
}  
function changeImg() {  
 //遍历banner下所有的div将其隐藏  
 for(var i=0;i<len;i++){  
 pics[i].style.display='none';  
 }

/\*索引跟图片的结合 通过pics这个数组\*/  
 pics[index].style.display='block';  
}  
slideImg();

验证用户输入的手机号和邮箱是否符合规范，并对用户设置的密码进行密码强度的检测。

//验证电话号码和邮箱  
function my$(id) {  
 return document.getElementById(id);  
}  
my$("phone-num").onblur=function () {

if(/(^[1][0-9]{10}$)|(^[0-9a-zA-Z\_.]+[@][0-9a-zA-Z\_.]+([.][a-zA-Z]+){1,2}$)/.test(this.value)){  
 //如果是true的话说明手机号或邮箱输入正确 把span隐藏了  
 my$("prompt").style.display="none";  
 //如果是false 显示span  
 }else{  
 my$("prompt").style.display="block";  
 }  
}

/\*获取密码框的id注册键盘抬起事件\*/  
my$("pwd").onkeyup=function() {  
 //显示密码强度  
 my$("pwd-strength").style.display="block";  
 //调用构造函数的同时 把把密码的值传给函数 this指的是my$("pwd")  
 //得到返回的等级  
 //根据等级改变类名 strength-level+0/1/2/3 my$("strength-level").className="strength-level"+(this.value.length>=6?getLel(this.value):0);/\*得加小括号\*/  
}  
//构造一个函数 给我一个密码 我返回相应的级别  
function getLel(pwd) {  
 var lvl=0;  
 if(/[0-9]/.test(pwd)){  
 lvl++;  
 }  
 if(/[a-zA-Z]/.test(pwd)){  
 lvl++;  
 }  
 if (/[^0-9a-zA-Z\_]/.test(pwd)){  
 lvl++;  
 }  
 return lvl;  
}

1. 个人主页：显示用户个人资料，查看我的关注者和我关注的人并显示用户发布的帖子和收藏的内容。其子页面用来让用户完善和维护个人信息，包括上传头像、更换邮箱、修改密码。
2. 作品发表页面：用户选择要发表的版区后，添加自己的描述并上传。

作品展示页面：分为体育、摄影和生活三个版区，分别展示出用户发布

//选择版区  
 <el-select v-model="value" placeholder="请选择版区" style="margin:20px 0 20px 20px">  
 <el-option  
 v-for="item in options"  
 :key="item.value"  
 :label="item.label"  
 :value="item.value">  
 </el-option>  
 </el-select>  
 <textarea style="resize:none;width:80%;margin: 20px 0 20px 20px;border-bottom: 1px solid #dbdbdb;border-right: 1px solid #dbdbdb;" placeholder="大胆发表你的作品吧.." rows="4"></textarea>  
 <el-upload style="margin: 0 0 0 20px;"  
 action="https://jsonplaceholder.typicode.com/posts/"  
 list-type="picture-card"  
 :on-preview="handlePictureCardPreview"  
 :on-remove="handleRemove">  
 <i class="el-icon-plus"></i>  
 </el-upload>  
 <el-dialog :visible.sync="dialogVisible">  
 <img width="100%" :src="dialogImageUrl" alt="">  
 </el-dialog>{{dialogImageUrl}}  
 <el-button style="margin:20px 0 20px 20px">发表</el-button>

methods: { //弹出计算机文档  
 handleRemove(file, fileList) {  
 console.log(file, fileList);  
 },  
 handlePictureCardPreview(file) {  
 this.dialogImageUrl = file.url;  
 this.dialogVisible = true;  
 }  
}

1. 页面设计

（1）Logo标志

logo2

（2）图标

图标1图标2图标3图标4

图标5

（3）页面结构

采用Vue组件化的页面结构，分别将<header><body><footer>组装成一个vue的component，实现一个单页面应用程序，页面采用居中对齐的方式，<header>部分随着鼠标或滑轮的滚动永远显示在页面的最顶端，页面中多处使用css的position属性，利用z-index实现多个遮罩层，并使用轮播的特效，给用户带来良好的交互体验。

（4）页面字体

项目字体为了避免特殊字体的不普遍性给用户带来不便，使用了系统默认的微软雅黑，页面中需要突出的文字给予一定的着重效果。

四、软件测试

1.页面测试



图4

作品上传页面：



图5

个人主页：

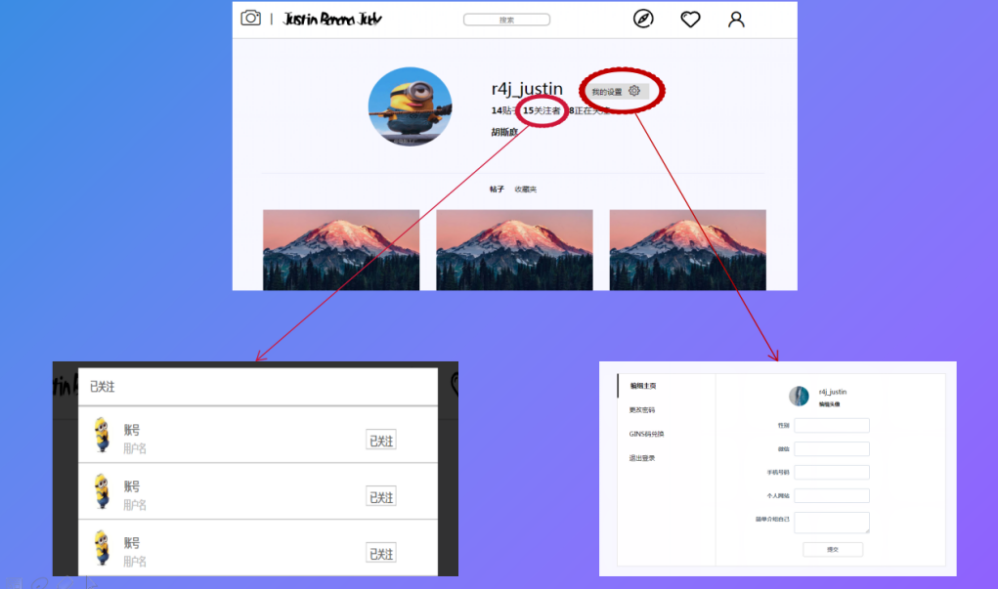


图6



图7

2.分辨率兼容性测试

项目布局多采用使用百分比的形式，当分辨率过大过小的时候都不会出现页面布局混乱的情况，通过在不同分辨率下访问项目可以测试页面的渲染情况，如下图所示：

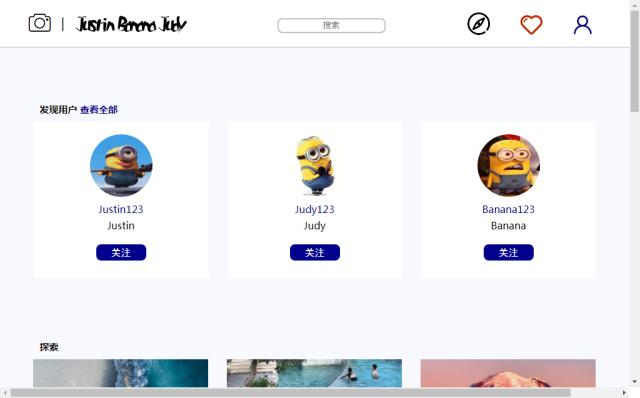


图8

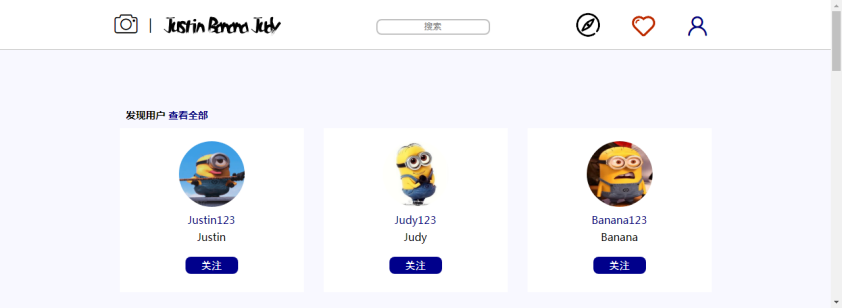


图9

3.浏览器兼容性测试

由于本项目多处使用了css的float属性，又设置了外边距，导致在ie6显示的外边距比设置的大，使得其中一块div顶到下一行。

解决方案：在float的标签样式中加入display:inline;将其转换成内行属性，避免了IE6的不兼容问题。

第五、软件完成后的总结汇报

通过完成这次项目，学习过的书本知识得到了充分的实践，HTML盒子模型和css的掌握更加熟练，又学习了一些vue和node.js的相关知识，让我有了继续学习这个专业的动力，收获颇丰。