过程管理

承诺需求与完成度列表

目标功能点	功能需求	完成度
登录	邮箱登录	全部完成
注册	密码重复性检验	全部完成
上传推荐	允许上传文字、图片	全部完成
展示推荐	1、搜索 2、记录点击量 3、点赞 4、按照点击量或点赞数排序	未完成：搜索
修改推荐	删除推荐	全部完成

任务分解与执行进度记录

任务	负责人	完成时间
上传推荐（前端）	赵文轩	4.25
密码重复性检验（前端）	赵文轩	4.26
邮箱登录（后端）	金辰哲	4.26
邮箱登录（前端）	赵文轩	4.27
上传推荐（后端）	朱家信、金辰哲	4.27
删除推荐（后端）	金辰哲	4.28
删除推荐（前端）	赵文轩	4.28
上传推荐（传输）	朱家信	4.29
记录点击量（后端）	金辰哲	4.30
排序（后端）	金辰哲	5.3
点赞（后端）	朱家信	5.3
排序（前端）	朱家信、赵文轩	5.5
个人主页（前端）	朱家信、赵文轩	5.6
绘图、整理文档	朱家信、赵文轩、金辰哲	5.9

开发过程关键技术介绍

前端

主要采用 HTML, CSS, Javascript 进行开发，使用了 bootstrap 5 框架，对于不同的浏览器，屏幕大小有比较好的适配能力。尽管 bootstrap 5 已经不再依赖 JQuery，但由于项目实际开发过程中仍然有很多地方需要 JQuery 的功能，因此项目仍然依赖 JQuery 进行开发。

发送http请求

主要借助的是ajax框架，完成打包数据，发送请求，接收后端数据，作出响应。

```
$.ajax({
        url: ,//目标资源
        type: ,//选择GET或者POST
        contentType: ,//类型是否需要经过默认处理
        processData: ,//数据格式是否需要经过默认处理
        async : ,//是否异步发送
        data: ,//附带的数据，只有当type="POST"时有效
        success:function (data) {
            //成功接收到响应，返回的数据存放在data中
        },
        error:function(data){
            //未成功接收到响应，后端可能会返回一些错误原因到data中
        },
        complete:function(){
            //发送请求这个动作完成后，会发生的事件；会在success或error前触发。
        }
    });
```

传参数有两种方法：

第一种是在url的末尾传参。

例如"/recommend/delete_recommend/?key=12324"这样一个请求，会在后端的路由表中通过正则表达式匹配到这样一个表项

```
re_path(r'^recommend/delete_recommend/$', recommend_app.views.delete_recommend),
```

在对应的处理函数delete_recommend(request)中，通过request.GET['key']就能获取到数据12324了。

第二种是在ajax框架中的data中传数据，通常采用json格式进行传输。

例如：

```
data : {
    usr : 'jason',
    pwd : '123456',
    img : files[0],//是一张图片
},
```

在后端通过request.POST中可以获得{usr:'json', pwd:'123456'}，而在request.FILES中可以获得图片，同样也可以是文件，文件是一个封装好的对象实体。

返回数据也有两种方法：

第一种是通过return HttpResponse(retdata)直接返回数据，然后交给前端js去操作；

第二种是通过return render('a.html', retdata)，渲染网页后返回，前端会默认进行一次网页跳转。

后端

1. 数据库生成

   在django项目下每个新的app内的models.py文件中定义模型的各个属性和属性的类别，再通过

   ```
   python manage.py makemigrations # 创建迁移文件
   python manage.py migrate # 将迁移文件写入数据库
   ```

   来保证数据库文件可以正常被调用

2. 获取前端数据：

   ```
   request.POST["key_name"]
   ```

3. 对数据库进行增删查改：

   ```
   MODEL.objects.create(key=key_value) # 添加条目
   MODEL.objects.get/filter(key=key_value).delete() # 删除条目
   MODEL.objects.get/filter(key=key_value).order_by('key') # 查找信息
   MODEL.objects.get/filter(key=key_value).key = new_value # 更改信息
   ```

4. 发送验证邮件：

   ```
   EmailMultiAlternatives(email_subject, text_content,
   settings.DEFAULT_FROM_EMAIL, [email]) # 定义邮件主题、内容、发送方，接收方等信息，完成邮件发送
   ```

5. 确认验证邮件是否超时：

   ```
   created_time = confirm.created_time
   now = datetime.datetime.now()
   now = now.replace(tzinfo=pytz.timezone('UTC'))
   cmp = created_time + datetime.timedelta(minutes=settings.CONFIRM_MINUTES,
   hours=settings.CONFIRM_UTC)
   if now > cmp:
       # code
   # 判断邮件生成时间与当前时间之间时间差少于设定时间差
   ```

6. 获取图片：

   ```
   piclist = request.FILES.getlist("picture") # 获得所有传送图片文件
   pic_num = len(piclist)
   for i in range(pic_num):
       pic_file = piclist[i]
       # code
   # 获取列表中每个图片信息
   ```

7. 保存图片：

```
def recommend_path(instance, filename):
    ext = filename.split('.').pop()
    # filename = 'recommend_{0}_picture_{1}.
{2}'.format(instance.picture_key, instance.picture_id, ext)
    filename = instance.picture_id # 重命名图片
    return os.path.join('recommend', filename)  # 设置保存地址

class recommend_pic(models.Model):
    # code
    picture = models.ImageField('推荐图片', upload_to=recommend_path,
blank=True, null=True)  # 调用设置函数
```

阶段回顾

Sprint 2主要目标是完善基本功能，实现了上传、浏览、删除三个基本功能；再者，对之前得工作进行了一次回顾，绘制了uml图，理清了业务流程，代码结构等，这些都有助于后面的开发。

对于开发过程而言，我们针对Sprint 1存在的问题做出了一些变化，如任务细分，每个组员的分工不再严格地按照"前端、后端、传输"进行划分，而是尽量每个人都多学习一点，按照功能点进行联合开发。

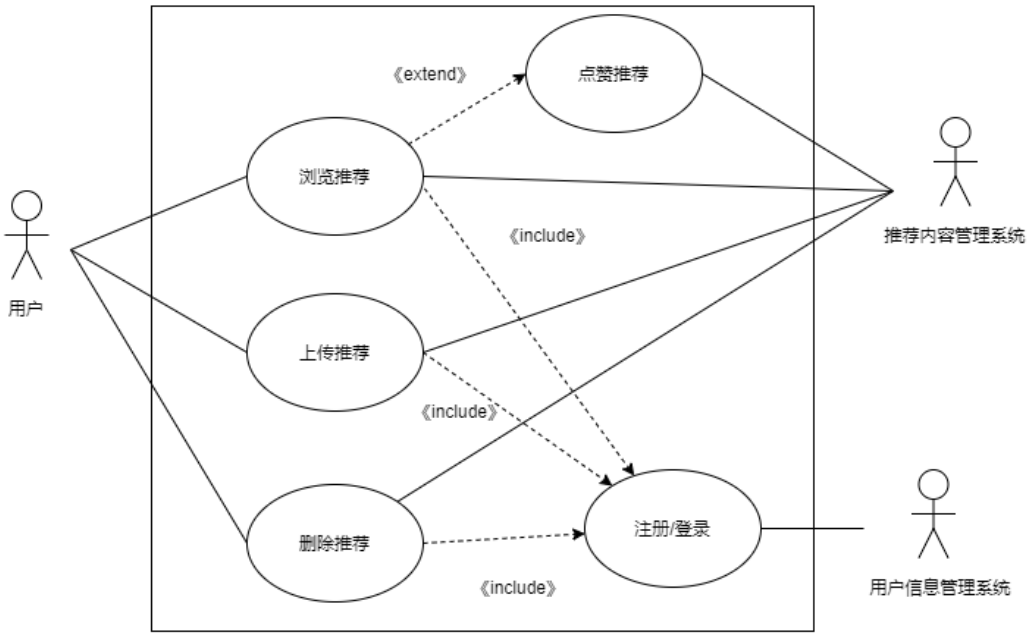
阶段二可能会存在的最大问题是，还未对系统进行很全面的测试，因为还未导入一些真实的数据，所以没办法对搜索功能进行很好的测试，以及没法测试我们的框架在数据大的情况下，是否可行和优秀。

应用分析与设计

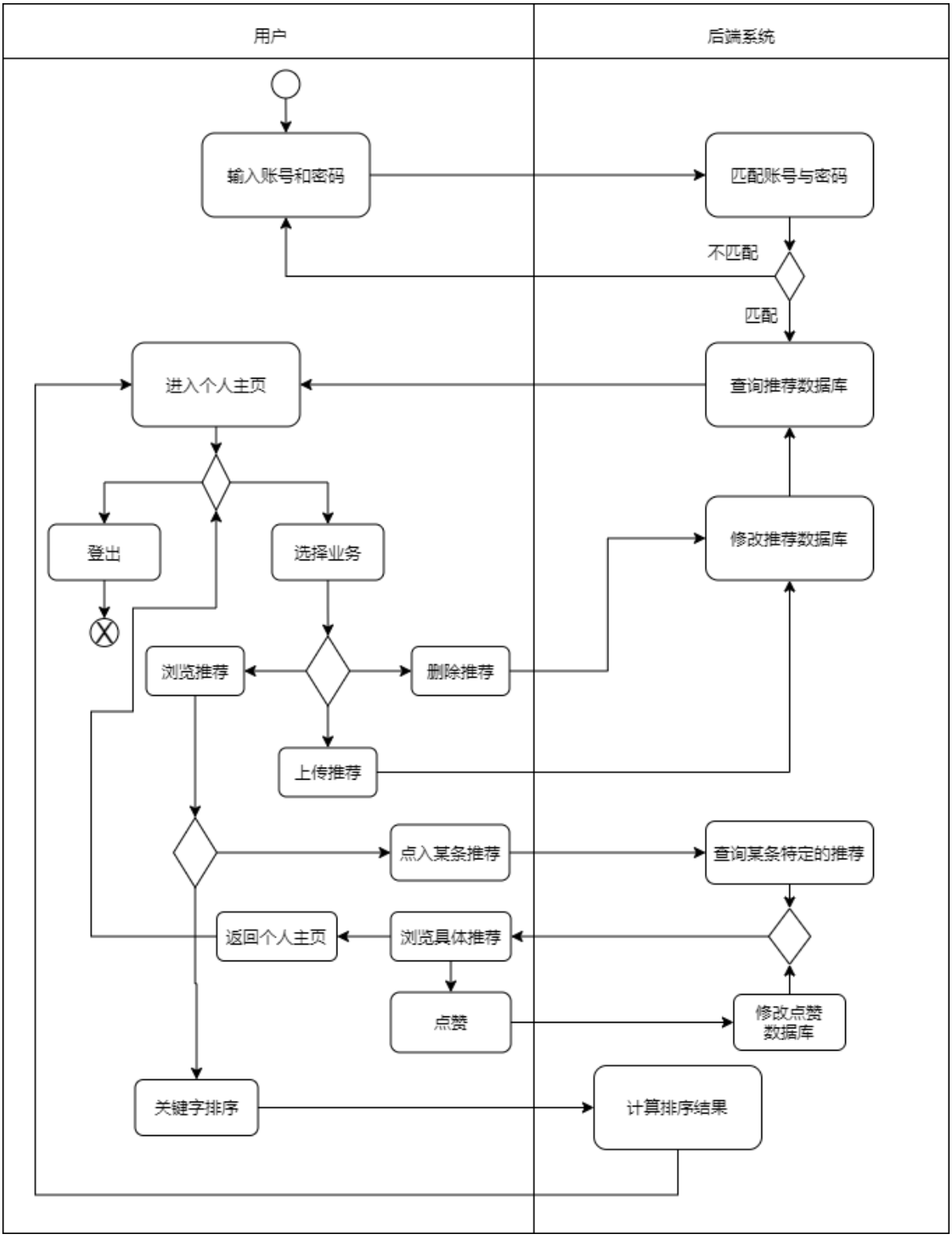
面向对象分析

核心用况	用况说明
浏览推荐	用户从网页进入某个推荐的详情页面，浏览页面不一定需要登录。网页会从后端的推荐内容管理系统获取推荐的具体内容。登录后可以点赞推荐，点赞后会反馈给后端的推荐内容管理系统。
上传推荐	上传推荐必须首先登录，输入推荐的标题，内容以及可选的图片，上传到后端的推荐内容管理系统进行储存。
删除推荐	登陆后可以对自己之前上传的推荐进行删除，发送删除请求到后端的推荐内容管理系统，系统自行删除。

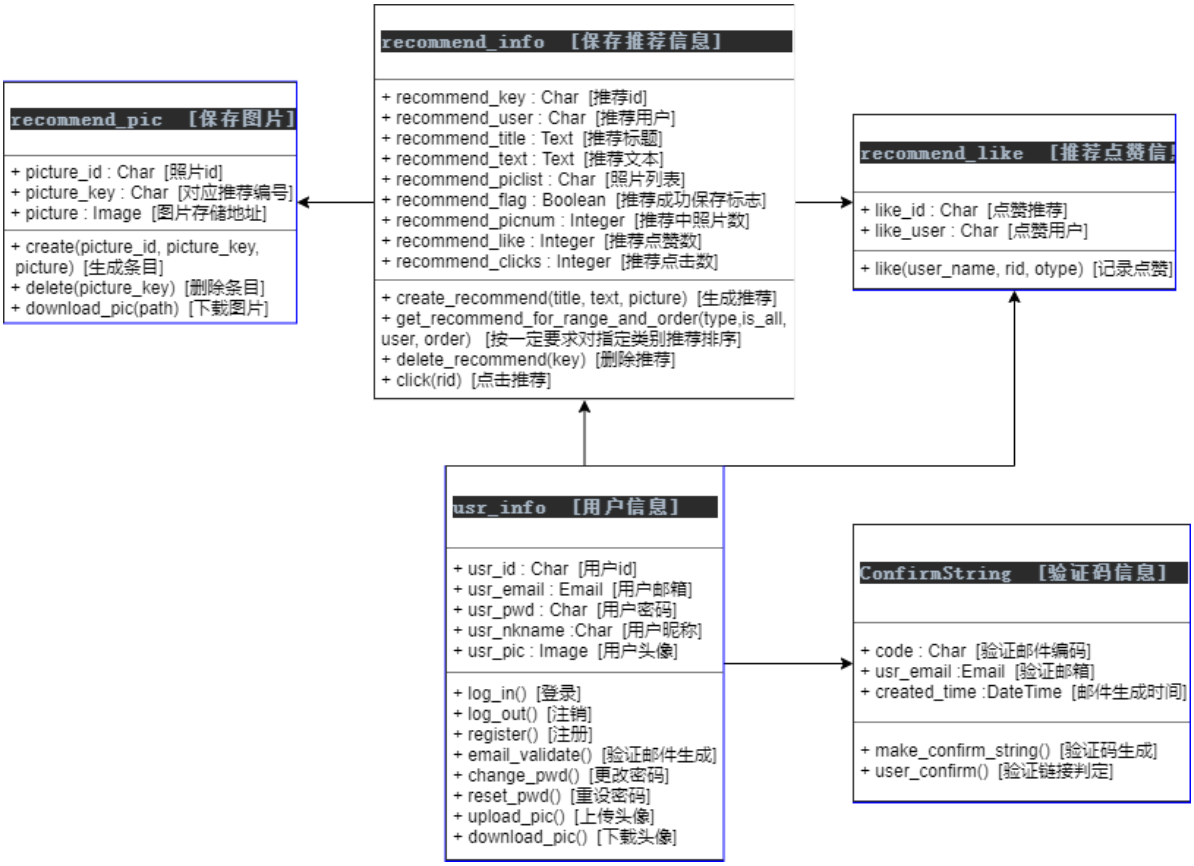
用况图


《extend》
点赞推荐
浏览推荐
《include》
推荐内容管理系统
用户
上传推荐
《include》
删除推荐
注册/登录
《include》
用户信息管理系统

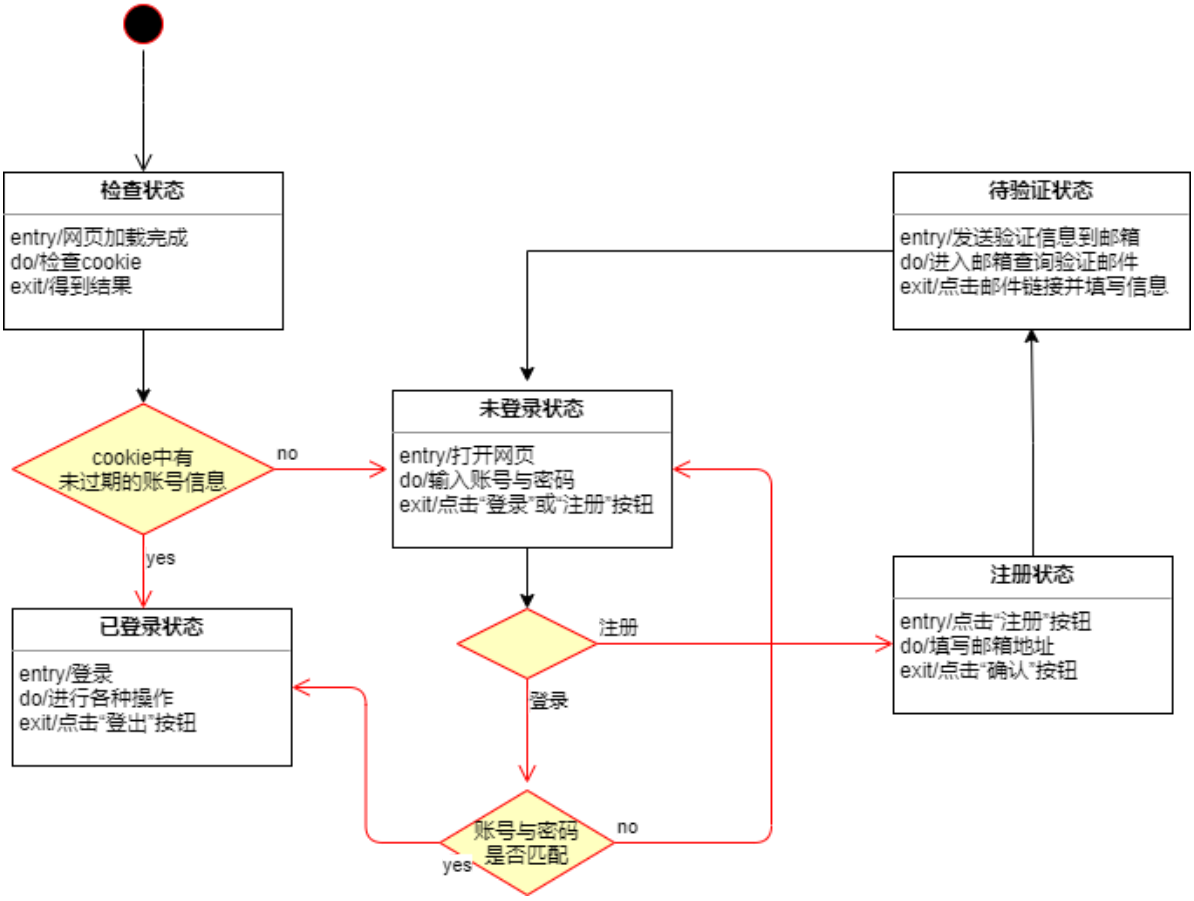
活动图


用户
后端系统
输入账号和密码
匹配账号与密码
不匹配
匹配
进入个人主页
查询推荐数据库
登出
选择业务
修改推荐数据库
浏览推荐
删除推荐
上传推荐
点入某条推荐
查询某条特定的推荐
返回个人主页
浏览具体推荐
点赞
修改点赞
数据库
关键字排序
计算排序结果

需求类设计


recommend_info [保存推荐信息]
+ recommend_key : Char [推荐id]
+ recommend_user : Char [推荐用户]
+ recommend_title : Text [推荐标题]
+ recommend_text : Text [推荐文本]
+ recommend_piclist : Char [照片列表]
+ recommend_flag : Boolean [推荐成功保存标志]
+ recommend_picnum : Integer [推荐中照片数]
+ recommend_like : Integer [推荐点赞数]
+ recommend_clicks : Integer [推荐点击数]
+ create_recommend(title, text, picture) [生成推荐]
+ get_recommend_for_range_and_order(type,is_all, user, order) [按一定要求对指定类别推荐排序]
+ delete_recommend(key) [删除推荐]
+ click(rid) [点击推荐]
recommend_pic [保存图片]
+ picture_id : Char [照片id]
+ picture_key : Char [对应推荐编号]
+ picture : Image [图片存储地址]
+ create(picture_id, picture_key, picture) [生成条目]
+ delete(picture_key) [删除条目]
+ download_pic(path) [下载图片]
recommend_like [推荐点赞信]
+ like_id : Char [点赞推荐]
+ like_user : Char [点赞用户]
+ like(user_name, rid, otype) [记录点赞]
usr_info [用户信息]
+ usr_id : Char [用户id]
+ usr_email : Email [用户邮箱]
+ usr_pwd : Char [用户密码]
+ usr_nkname :Char [用户昵称]
+ usr_pic : Image [用户头像]
+ log_in() [登录]
+ log_out() [注销]
+ register() [注册]
+ email_validate() [验证邮件生成]
+ change_pwd() [更改密码]
+ reset_pwd() [重设密码]
+ upload_pic() [上传头像]
+ download_pic() [下载头像]
ConfirmString [验证码信息]
+ code : Char [验证邮件编码]
+ usr_email :Email [验证邮箱]
+ created_time :DateTime [邮件生成时间]
+ make_confirm_string() [验证码生成]
+ user_confirm() [验证链接判定]

状态机图


检查状态
entry/网页加载完成
do/检查cookie
exit/得到结果
cookie中有
未过期的账号信息
no
yes
未登录状态
entry/打开网页
do/输入账号与密码
exit/点击"登录"或"注册"按钮
已登录状态
entry/登录
do/进行各种操作
exit/点击"登出"按钮
注册
登录
账号与密码
是否匹配
no
yes
待验证状态
entry/发送验证信息到邮箱
do/进入邮箱查询验证邮件
exit/点击邮件链接并填写信息
注册状态
entry/点击"注册"按钮
do/填写邮箱地址
exit/点击"确认"按钮