完善项和占比安排情况

占比安排策略

孔馨怡初始80% ，其余4个人每人初始5%

后续完善功能多做一项 +5%（从孔馨怡的占比里面扣）

例如： 我多做一项 就是初始5%+完善5% = 10% 贡献比

初始分工详情

- zzy:前端优化、攻略内容填充
- yly:前端优化
- ysy:后端逻辑优化
- fjj:后端逻辑优化、评论功能完善
- kxy:前端框架、逻辑和样式，后端：略 诸多功能 我后面写

后面多做的补在后面

待办事项与流程要求细则

一些嘱咐

1. 对数据库进行操作的方法是mapper，有java和xml文件

- 大部分sql和简单的sql操作以及写了 看mapper的java文件去调用
- 如果你需要新的函数去操作sql，现在java里面创建函数，可以跳转xml去写，也可以用注解在上面直接写，如果是查询注意结果封装方式，最好采用xml并且加入resultMap来保证结果传递完整
- 传参要么传实体类，要么传多个参数，多个参数使用Param注解标注
- 如果出现xml疑义用CD包裹
- 如果使用数据库方法例如更新数据库原本内容时记得事务提交和回滚

2. 关于前端

- 所有的前端样式可以继承前面的页面的样式
- 前端总共有三部分构成，我第一次开发经验不足，这三部分写在一个文件里
- html写框架，css是样式，一部分在文件里一部分在前面的html文件里可复制
- js是逻辑，可以写函数把信息和json传给后端去处理

3. 关于后端处理

- 写一个servlet文件，web开头注解命名接口名称
- 重写doget和dopost方法
- 如果要使用到当前登录用户信息获取session
- 如果要使用前端传过来的数据要么从req.getParameter("名字");获取，要么从前端传回来的json获取
- 如果要返回信息给前端，要么自己编写json，要么使用对象方法如：

```
// 创建ObjectMapper对象用于JSON转换
ObjectMapper objectMapper = new ObjectMapper();

// 将查询结果转换为JSON并写入响应
String jsonResponse = objectMapper.writeValueAsString(你要传的东西);
resp.getWriter().write(jsonResponse);
```

- 处理结束以后的操作要么给前端返回json，如{success:信息} 要么你直接在response里写一个弹窗就行

4. 具体可以参考前面的，有问题问，我做了两个javaweb项目现在已经很清楚哪里有问题了 而且我们项目里的绝大多数文件都是写了很详细的注释的，特别是后端，很清晰

首页推送完善--fjj

- home.html页已经做好的图片展示给他换成动态的推送，也就是要去数据库里存储的
- 把攻略文字放一点上去 例如标题或者内容
- 点封面图片能跳转详情界面，这个后端接口已经写好，去看detail等的使用
- 数据库里guide表is_featured列是记录是否推荐的，默认全置0了 你想上推荐的就要在里面选择置1的推
- 写一个servlet或者就直接用之前其他返回攻略详细内容的后端接口也可以
- 写完以后其实这个逻辑在index.html也是同样适用的，直接粘过去就行，或者你想让index首页是别的介绍内容就随便写点网站的大致文字描述放着也行 or 放着不管他

退登--yly 有兴趣写下软注销

- 退登的后端servlet文件编写
- 所有涉及到要加退登按钮的前端修改

个人中心my.html展示更多信息

- 很简单的修改：加入男女/地区/生日/电话/邮箱，在现有的my.html上加显示就行了，对应查询用户信息的mapper查询的是用户完整信息，在接口profileServlet里json加入想要的项目即可
- 需要添加的功能：可以查看用户的关注和粉丝列表（逻辑完全一样就放在一起了），你可以用窗口展示或者直接再写一个html都行

管理员界面：主要是前端逻辑设计，后端我帮你

我想了下现在最简单的操作就是把现在的搜索界面的逻辑和页面直接复制粘贴，改一下成为管理员的界面。 这样不涉及其他前端界面的修改工作量最少 （问就是我们管理员的内网，设不设置密码都行，反正用户不会跳转到这里，只有我们管理员知道就方便打开操作）

要明确的事情：

- guide表有一列“status” 如果是published那么外界用户可见，如果是draft说明用户提交但是我们没有审核通过
- 管理员来做的事情就是草稿箱里的draft

具体要做的事情：

- 搜索界面粘过来抬头改一下，逻辑照旧

- 加一个选状态的框：draft/published 这样你可以去审核帖子进行通过/删除
- draft相当于你现在要审核他然后让他变成published其实就是一个查看详情然后更新他的状态的操作
- 如draft一样 区别在于看的是已经发布的 published 有没有违规 违规更新状态回draft
- 按照这样的逻辑把前端js的逻辑写好，传对应的数据给后端，参考前面的文件，其实也就是加一个更新状态的函数罢了

选择这一项可以联系我，我可以帮你写后端接口或mapper给你，你自己去调前端去，会快一点