# 系统结构

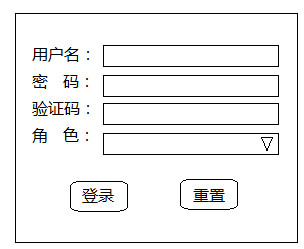
1. 前台展示页
   1. 部落简介
   2. 公告栏
   3. 交流会风采展
   4. 软件赛风采展
   5. 大创风采展
   6. 考核平台（考核人员权限）
2. 后台管理页
   1. 需登录方能进去，不同角色权限进去的页面不同
   2. 成员管理
   3. 部落活动管理

# 后台管理页结构

1. 登录
2. 成员管理
3. 部落活动管理

## 后台管理登录

1. 登录页面（不要角色）



1. 业务逻辑
   1. 需要用到权限管理框架
   2. 数据层
   3. 业务层
   4. realm（继承AuthorizingRealm））
      1. 设置realm的名称，程序员可自己硬编码
      2. 注入service
      3. 重写doGetAuthenticationInfo方法进行认证，调用数据库查询用户信息（密码，盐。。。。。）
   5. 控制层
      1. 验证码的生成
      2. 错误代码处理
   6. 视图层
      1. 表单参数限制
      2. 提交地址限制
   7. 由于此项目加了一个验证码校验，所有需要自定义校验过滤器（继承FormAuthenticationFilter），改写认证方法onAccessDenied，先校验验证码，再做账号密码校验
   8. 配置文件配置

## 后台管理页面

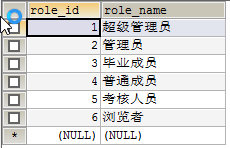
1. 用户登录后，进入后台首页，显示当前用户跟用户权限列表。
2. 后台管理页面包括两个板块：成员管理、部落活动管理。

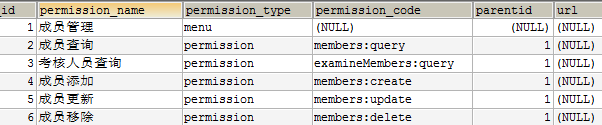
## 后台首页

1. 首页是用户登录后进入的页面，需要显示用户的名字和用户菜单列表及菜单下的权限列表。

需要在UserRealm中添加查找用户菜单列表和权限列表的功能。

1. 抽取出来的业务有两个
   1. 用户菜单列表
   2. 用户权限列表
2. 先设计好数据库测试数据





1. 数据库层操作
   1. 根据当前用户查用户角色进而查询用户拥有的菜单列表
   2. 根据当前用户查用户角色进而查询用户拥有的权限列表
2. 业务层操作
   1. 调用数据库层的相应方法，查询菜单列表
   2. 调用数据库层的相应方法，查询权限列表
3. 对UserRealm，需要做大改。
   1. 在认证方法中，之前是将从数据库中查询出来的用户信息TeamMembers，全部丢进SimpleAuthenticationInfo中。
   2. 而现在，因为除了存用户信息，还需要存用户的菜单和权限，而权限又是属于菜单的子元素。所以我们需要新建一个用户类User，专门存储用户的信息，包括用户主键，用户名，菜单列表。
   3. 在将用户信息设置在simpleAuthenticationInfo前，先将TeamMember中的信息取出来存在User类中，然后根据用户名去查用户菜单及权限，然后，对菜单和权限的子父关系做好相应的映射，再将用户菜单存储在User类中。
4. controller—UserController
   1. 在shiro的filter中，设置用户登录成功后请求/welcome，所以需要编写处理这个请求的方法
   2. 在UserController中编写一个welcome方法，从shiro中取出用户身份信息，传回页面。代码如下。
5. 后台首页
   1. 显示用户的姓名
   2. 显示用户拥有的菜单和权限列表
6. User类的设计：
   1. User类需要的内容是，用户标识，用户名，菜单列表。
   2. 而菜单列表除了要存储数据库中的相应内容外，还需要存储菜单列表下拥有的权限列表。所以需要新定义一个菜单类，使他继承现有的Permission的基础上，再加一个子权限列表的属性。

## 后台管理之成员管理

1. 不同的用户登录进后台系统时，权限内的菜单已经显示出来了，接下来就是对用户的访问做授权。
2. 不同角色拥有的访问权限不同，通过对用户访问的路径来做授权拦截。
3. 成员管理包括：成员查询，考核人员查询，成员添加，成员更新，成员移除。

### 成员查询（这里的成员是爪哇部落正式成员）

1. 查询出来的信息包括：

成员标识，年级，组别，姓名，手机，短号，微信，QQ，职务，角色

1. 由于在数据库中，team\_members表中，存储的年级、组别、职务、角色等都是标识引用，而在显示时候需要的是具体的内容，所以这里需要重新定义一个model类来存储成员信息。可以看出这里新定义的类里面，许多属性和TeamMembers中是重复的，所以这个类继承TeamMembers
2. 当然，这里查询的时候，涉及到多表查询，我们也需要重新定义mapper类。这里的查询有限定，查询的一定是正式成员。可以使用角色这个条件。
3. 我们可以发现，正式成员当中，仅有部分人是由职务的，如果进行多表查询，职务那个条件是会出现错误的，所以我们这里只差职务id，等在service层取出所有成员的信息后，再根据成员的职务标识查出职务名称。
4. 后期，需要对查询出来的数据做分页显示，所以还需要在数据库中更改查询操作，使用limit方言。

### 成员查询（这里的成员是考核人员）

1. 考核人员的查询跟上面正式成员的查询是差不多的，只是这里的条件改成角色是考核人员而已。而且考核人员暂不属于任何一个组，也没有职务可言。
2. 后期也需要做分页处理。
3. 显示的内容如下：

成员编号，姓名，手机，短号，微信，QQ，年级，角色

### 成员查询之分页显示

1. 分页显示需要记录
   1. 当前页，
   2. 总页数（从总记录数中得到），
   3. 每页记录数，
   4. 总记录数（从数据库中查出来），
   5. 当前页的数据集，
   6. 还需要记录URL。因为每次点击页码时，需要指定请求路径及请求参数还有当前的页码。
   7. 将这些信息设置成一个model
2. 数据库层
   1. 查询总记录数count函数
   2. 分页查询，需要传入当前页和每页记录数。直接用分页model
3. service层
   1. 提供一个接口，只接受controller中传入的当前页码
   2. 先实例化一个分页model出来
   3. 需要先查询出总记录数
   4. 然后自定义每页记录数，这样就可以计算出总页数
   5. 需要从controller中获取当前要查询的页码。
   6. 调用数据库层的分页查询，传入当前页和每页记录数，得到当前页的数据集。
   7. 将数据集设置在model中。service方法就结束了
4. controller层
   1. 从页面中先获取当前页码
   2. 然后调用service层的方法，得到分页model
   3. 解析controller层获取到的请求路径，去掉页面封装在分页model中。
   4. 然后把分页model传回给页面
5. 页面比较复杂
   1. 页面中的数据直接从分页model中取出显示。（成员标识，年级，组别，姓名，手机，短号，微信，QQ，职务，角色）
   2. 比较复杂的是页码列表的显示，下面一步一步分析。
   3. 最多显示10个页码！当前页，在页码列表中的位置，定为6；
   4. 计算公式：
      1. 如果总页数<=10（列表长度），那么begin=1，end=总页数
      2. 使用公式计算；begin=pc-5, end=pc + 4；
      3. 头溢出：当begin<1时，让begin=1
      4. 尾溢出：当end>${tp}时，让end=${tp}
   5. 页面的URL设为分页model的URL+当前页码

### 成员添加

1. 成员添加需要添加的信息如下：

姓名，密码，盐（系统给），年级，手机，短号，微信，QQ，组别，角色，职务

1. 给出的添加页面中，年级、组别，角色，职务等是从数据库中查询出来的具体内容，放在下拉列表框中，id设为对应内容在数据库中的唯一标识，显示的内容即为具体的名称。
2. 从添加表单中，获取到的成员信息与team\_members数据表的表结构相似，不需要另写sql已经，直接用逆向工程出来的方法即可。int insertSelective(TeamMembers record);
3. 但还有几个问题，盐，需要系统自动生成，需要添加生成随机数的方法。密码在数据库中存的是加密后的，所有也需要写一个加密算法。
4. 在service层调用数据库层的方法前，需要先对盐进行填充，以及密码进行加密。再入库。
5. 具体实施之——添加页面的显示
   1. 页面中请求成员添加链接的时候，先用get方式请求，请求controller中的相应方法。
   2. controller中需要取年级，组别，角色，职务等信息，是一个集合列表。存储好传回给添加显示页面
   3. 在添加显示页面中制作好表单，年级，组别，角色，职务等信息需要从域中取出填充。
   4. model层用现有的年级，组别，角色，职务等对象
   5. mapper层，需要查询数据库中现存的年级，组别，角色，职务等，逆向工程中没有提供这样的方法，需要自己写。
   6. service层，逻辑比较简单，就只是调用mapper层的内容。
6. 具体实施之——成员添加
   1. 页面中输入具体的内容
   2. 在controller中接受页面内容，直接使用TeamMembers这个model即可。
   3. 然后调用service层完成添加操作。成功跳转至查询页面
   4. 输入参数的校验后期再做

### 成员添加的额外功能

1. 成员添加时，显示的年级，组别，角色，职务，是数据库中已经存在的。但会出现这样的情况，由于成员不断添加，现有的年级组别等信息不足以提供，所以，还需要提供一个添加年级、添加组别、添加角色、添加职务的功能。
2. 添加年级
3. 添加组别
4. 添加角色
5. 添加职务
6. 这四个添加操作，都可以直接使用逆向工程的代码，所以只需要在成员管理的业务类中，提供一个接口供外面调用，这个业务类里面再调用逆向工程的代码即可。

### 成员更新

1. 成员更新的内容包括

姓名，密码，年级，手机，短号，微信，QQ，组别，角色，职务（密码不在这里更新）

1. 更新的方法可直接用逆向工程生成的方法。（根据成员的标识）
2. 在service层，一样存在对密码进行加密的步骤，不过这里还增加了一步，就是要根据用户的标识，取出数据库中的盐，再做加密。
3. 具体实施之——成员更新表单的显示
   1. 根据成员编号，查询出成员更新需要的相关内容，然后显示出来
   2. 成员更新的发起是在查询页面中，点击修改，根据当前成员编号发起请求。
   3. controller层，取出成员编号，查询成员相关信息，同时还需要查询年级，组别，角色，职务等内容。存储在域中显示表单内容。
   4. service，根据成员编号查相应的内容。查年级，组别，角色，职务内容。
   5. 数据库层，根据成员编号查直接用逆向工程。
   6. 扩展功能：根据编号查询成员信息时，查询出来的密码是加密后的密码，能否做解密后再显示出来？因为当显示出来的信息经过更新后，密码即使没有更新，但由于显示的是加密后的密码，所以在更新的时候，把加密的密码再做一次加密，就出问题了。（MD5无法解密，换种方式做密码修改，不在成员更新中做密码修改。）

### 成员移除

1. 成员移除很简单，直接根据当前成员的标识删除，用逆向工程的方法即可。
2. 在成员列表页面中添加删除请求的URL，同时需要做友好提示，询问是否确定删除。为删除按钮添加onclick事件。onclick="return confirm('您是否真要删除该一级分类？')"
3. controller层接收删除请求，获取编号
4. service层，调用mapper层进行删除

## 后台管理之活动管理

1. 活动管理包括：公告管理、考核平台、交流会风采、软件赛风采、大创风采、蓝桥风采

### 完善后台首页关于活动管理部分的菜单显示

1. 采用一步一步添加的方式，由于活动管理相关的具体内容还不是太清楚，所以边分析边做，分析好一个功能，就添一个菜单。
2. 菜单显示需要做的无非几步
   1. 数据库中添加相关菜单和权限测试数据，做好角色权限映射。
   2. 后台首页中将菜单显示出来（这步好像可以不用做了，因为前面成员管理相关菜单的显示已经做好了遍历，各层的代码也写好了，只要数据库的测试数据一出来就可以显示出来了。）
3. 所以这一部分需要做的就只是数据库测试数据的设计和填充。

## 公告管理

1. 一个社团，每年会有许多活动或成就，有些是需要通过公告的形式来告知部落成员的。这里的公告定义如下：
   1. 非部落人，是可以看到这些公告的，但是看到的地方是在前台展示页
   2. 而部落人，权限范围是比较大的。
      1. 由于前台展示页是不需要条件的，任何人都可以看，部落人也可以通过前台展示页查看。前台的查看是有限制的，可能我们展示的只是部分公告，而且是公告列表的形式。而不能职能的进行检索查看。（后期有时间可以把前台展示页也完善一下，可以检索一些条件来查看公告）
      2. 当然查看公告的地方也可以是后台管理系统，而在后台管理系统中查看则需要先登录，获得了权限才可以查看，在后台查看，允许按一定的条件进行查看。
      3. 有部分部落人，系统是允许对公告进行更多操作的，分角色来授权

①毕业成员因为已经不再参与学校举办的这些活动，即他们不是活动筹划者，所以他们只能看公告而不能做其他事情。

②普通成员有些已可以接手部落的一些活动，所以允许他们做一些活动公告的发布，当然至于可以发布哪些类型的公告，由于系统无法做到如此细致的判断，这部分就交给人来处理，在部落明文规定哪些人可以发布哪些类型的公告才可行了。至于对已发布的公告做更改和删除，在普通成员里，系统只允许他们对自己发布的公告做删改操作。

③超级管理员和管理员是拥有公告相关的所有权限的

1. 后台公告查看设置设置一些条件用来检索：
   1. 根据标题检索公告
   2. 根据发布者检索公告
   3. 根据发布时间检索公告
   4. 更加公告状态检索（用单选按钮）
2. 公告发布
   1. 发布的公告包括哪些：
      1. 交流会开展的通知，
      2. 考核开启的通知（附上考核要求文档，注明考核时间范围）
      3. 考核截止时期的公布（考核截止期临近时发公布起提醒作用）
      4. 考核作业批改人员安排的通知，
      5. 考核作业批改结束的通知（并附上结果）
      6. 软件赛启动的通知
      7. 软件赛比赛分组结果的公布。
      8. 赛事开题、指导分配结果的公布。
      9. 中期检查开启的通知
      10. 结赛相关事宜的通知。
      11. 成绩、项目成果的公布。
      12. 蓝桥杯开赛的通知
      13. 蓝桥杯报名指导的通知
      14. 蓝桥杯交流会开展通知（跟上面交流会的通知是一样的，只是这里规定的主题是蓝桥杯）
      15. 蓝桥杯比赛时间的通知
      16. 部落成员在蓝桥杯中取得的成绩公布。
   2. 允许进行公告发布的角色有：超级管理员、管理员、普通成员
   3. 在数据库中，存储公告相关的哪些内容
      1. 公告标识，公告标题，公告内容，发布时间、发布者
      2. 我们希望能够记录公告相关的历史情况，所以增加一个公告的状态信息。是int类型，目前我让他1为发布状态，0为删除状态，后期有变动再改。
      3. 根据上面发布公告类型的分析，可以看到，有时候是需要上传文件的，所以还需要添加附件这一栏（为此还需要设计一个附件的数据表，这里存的仅是附件数据id，具体的查看数据库设计文档）
3. 公告更改
   1. 超级管理员和管理员可以更改任何公告
   2. 普通成员只能更改自己发布过的公告。（这点可以通过控制程序逻辑来实现）
4. 公告删除
   1. 超级管理员和管理员可以删除任何公告
   2. 普通成员只能删除自己发布过的公告。（这点可以通过控制程序逻辑来实现）

### 后台首页先显示出相应的菜单和页面出来

1. 活动管理：一级菜单
2. 公告管理：二级菜单
3. 公告管理点开后，弹出一个页面，页面中是一个导航页，有两个页，分别是公告列表和发布公告。（待定）
4. 数据库测试数据
   1. 活动管理：菜单，
   2. 公告管理：
      1. 菜单，
      2. 父id是活动管理的id，
      3. 权限码：affiche
      4. url：
   3. 具有活动管理和公告管理的角色有超级管理员、管理员、普通成员、毕业成员、考核人员。
5. 公告管理点击后，弹出的页面开发，先做好导航页，后面，默认让他展示公告列表。（待定）

### 公告发布

1. 允许进行公告发布的角色有：超级管理员、管理员、普通成员
2. 公告发布的链接：在点击公告管理后弹出的那个页面中。
3. 公告发布包括公告发布表单的显示，和公告入库操作。
4. 公告发布实现之——公告发布表单的显示（这部分只涉及页面和controller）
   1. 发布链接中设置请求URL
   2. 首先需要先显示公告发布页面，是一个表单，还需要引入富文本编辑器。公告发布表单设计。



* 1. 富文本编辑器
     1. 下载富文本编辑器的包
     2. 在页面中引入富文本编辑器的CSS样式和JavaScript文件
     3. 在表单中创建一个textare，并定义一个id值。
     4. 写一个JavaScript脚本，用来创建富文本编辑器。
  2. 发布公告表单可能需要上传附件，所以需要修改表单，使支持文件上传
     1. enctype="multipart/form-data
     2. <input type="file" name=" "/>

1. 公告发布实现之——公告入库
   1. 测试工作
      1. 先看看在controller如何获取富文本编辑器的内容

通过测试，可以发现，可以直接想获取普通input内容那样来获取富文本编辑器的内容，而且获取的是带格式的内容。

* + 1. 引入上传组件，因为有时候很会需要上传附件

① 然而发现，获取普通表单项内容时，出现了乱码。解决方法如下

②普通表单项内容使用FileItem类的getString(“utf-8”)来处理编码。

* 1. 因为需要用到文件上传，所以需要使用上传组件。在maven中引入依赖，需要commons-fileupload.jar、commons-io.jar
  2. 页面：对表单有限制，需要做修改
  3. controller层
     1. 先用上传三步获取FileItem
     2. 然后判断是否有附件，如果有

① 获取上传文件的文件名称，由于浏览器差异，需要做一些处理

② 为防止文件重名，还需要为文件添加前缀

③ 文件最好不要放在一个目录下，根据部落的活动开展情况，最好用当前时间作为目录。

④ 文件需要存储在WEB-INF下

⑤ 将上传的文件保存在指定目录下。

⑥ 获取文件存储路径，封装到公告model对象中

这里还有其他额外的步骤，参见下面第6点

* + 1. 无附件，获取其他普通表单项，放入公告model对象中，需要注意编码问题
    2. 可以发现，数据表中还需要存储发布者，所以我们需要从session中获取用户
    3. 调用service层进行公告内容入库操作
  1. mapper层
     1. 因为我们对公告数据表做了处理，所以需要重新逆向工程，同时还添加了一个表，配置文件要先添加这个表的映射后再逆向工程。
     2. mapper层公告添加可以直接使用逆向工程的方法。
  2. service：需要先设置公告的status为1，然后再调用mapper层的方法。

1. 因为有附件的时候，我们需要记录附件的名称和存储路径，所以需要在controller层做附件入库处理，而公告中存储的是附件id，所以我们设计成如下。
   1. 附件的逆向工程生成的mapper可以提供添加功能，直接使用
   2. 在service层调用逆向工程的mapper，但主键是用UUID生成的，并且在service层，需要做判断，如果添加成功，返回UUID主键值
   3. 然而在controller层，就可以获取到UUID，放到公告中。

### 公告查看

1. 可以通过后台管理系统查看公告的成员有：考核人员、普通成员、毕业成员、管理员、超级管理员
2. 公告查看需要分页
3. 公告查看的高级实现——添加条件查询，同时还是需要做分页处理
   1. 根据标题检索公告
   2. 根据发布者检索公告
   3. 根据发布时间检索公告
   4. 更加公告状态检索（用单选按钮）
4. 公告查看的结果分两种实现
   1. 查询出数据库存在的所有已发布的公告（即控制好status），用表格形式展示出公告的大概内容，包括公告id，公告标题，公告发布时间，公告发布人，是否有附件
   2. 点击具体某条公告，展示该条公告的所有内容，包括公告标题，公告内容，发布时间和发布人。
5. 公告查看的发起页面是，点开公告管理后的那个页面，通过超链接访问，默认是查看全部公告，且是第一页，显示结果是时间倒序。
6. 在公告列表页面中，有高级查询的表单，和公告列表的表格，同时公告列表是分页的。
7. 公告无条件查看的实现
   1. 首先发起页是公告管理后的页面中的超链接，查询第一页公告。
   2. 数据库层
      1. 需要自己定义mapper，
      2. 需要查询出无条件的所有公告的记录数，同时查询出来的公告状态必须是1。
      3. 查询指定页数的公告，接收分页model，使用指定下标和当前记录数，查询数据库的公告内容。包括公告id，公告标题，公告发布时间，公告发布人。

① 从数据库中，我们发现，之前设计数据库时候，发布人我们设的是发布人标识，然而目前为止，我们发现这个设计有点麻烦，我们对数据库做修改，直接存发布人的名字，这样就免去关联操作了。

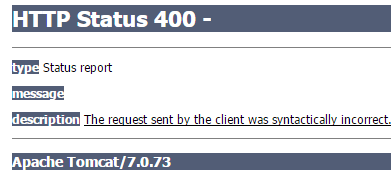
② 如果用的是发布人标识，不仅需要多次连数据库或做多表查询，连model层都得做封装，工作量比较大，所以决定直接存发布人姓名

③ 去修改数据库，改成发布人姓名，然后再逆向工程，只逆向公告表

④ 修改之前已经实现的公告发布的controller层，将发布人ID改成发布人姓名。

* + 1. 按时间倒序写出指定的分页sql查询语句，同时查询出来的公告状态必须是1
  1. service
     1. 向外提供的接口是，只接收当前页数，返回分页model的方法。
     2. 在service层需要实例化分页model，把当前页设置到model中。
     3. 调用数据库查询总记录数
     4. 调用常数工具类，获取每页公告的显示数
     5. 调用数据库查指定页的数据，封装在model中，返回。
  2. controller
     1. 获取页面的页码，根据页码调用service层方法，查指定页码的数据
     2. 实例化分页model，将页码、查询出来的数据、当前URL（去页码后的）封装在model中
     3. 封装在request域中返回给页面
  3. 页面
     1. 头部先定义一个高级查询表单
     2. 中间是公告列表，遍历出来request域中的分页model中的数据。
     3. 尾部是页码，仿照查询成员时候的分页页码实现。
  4. 设置公告查询的权限
     1. 在数据库中设置权限表相应内容
     2. 然后对角色权限做好映射

1. 条件查看公告的实现——添加条件查询，同时还是需要做分页处理
   1. 公告查询的条件有：标题、发布者、发布时间和公告状态
   2. 公告条件查询的发起页面是与公告列表同一个的那个页面
   3. 通过测试，发现公告类别和公告条件表单在同一个页面的时候，公告表单提交不了，不知道是不是<iframe>标签的问题，还是resulf风格的URL的问题，这种方式先放下，有时间再研究。
   4. 现在改变一下实现方式，把公告条件查询，另开一个页面
      1. 在公告管理打开后的那个页面添加一个高级查询公告连接，用来显示公告添加查询的表单。
      2. 用一个单独的查询表单来条件查询公告，同时默认查询的是第一页。
   5. 好像（3）中提到的问题解决了，因为我的公告管理controller之前是做过公告发布功能的，他需要做上传处理，所以我对controller做了修改，添加了上传需要的@MultipartConfig注解，这样在post请求中，就无法向之前那样使用了，不知道是不是这样原因，做一些测试
   6. 先把公告查询单独出来，写一个单独的controller。
   7. 对于（3）中的问题，（5）中的解释可能有问题，现在再仔细看，报的错误不是404找不到页面，而是400，百度的解释是：出现HTTP Status 400大多是因为，jsp的form表单提交的字段类型和后台接收字段类型不匹配造成的（例如，form中为String，后台接收为Date）。现在回想，确实如此，我的Affiche类中，发布时间是Date类型的。对于这类问题，解决的方法是自定义参数绑定将日期串转成java.util.Date类型



* 1. 自定义日期类型绑定
     1. 定义一个类实现Converter接口
     2. 在SpringMVC配置文件中配置自定义的Converter
  2. 通过测试，（7）的解释是正确的，确实是参数类型问题

1. 条件查看公告的实现——添加条件查询，同时还是需要做分页处理
   1. 公告条件查看还需要涉及分页，在controller层的实现比较麻烦，因为涉及到分页之后，第一页是带条件的，而且使用了springmvc，自动将条件封装在model上，但是当跳转到其他页后，条件就不存在了，虽然分页model中设置了URL，但是这个URL如何封装到model中，就是个问题了。
   2. 这个问题有点复杂，多条件的查询先放下，有空再回来做。
2. 公告内容查看
   1. 点击具体某条公告（点击标题），展示该条公告的所有内容，包括公告标题，公告内容，发布时间和发布人，显示出附件，可以下载附件。
   2. 公告内容查看的发起页面是，公告列表（表格）中公告标题，将标题设置为超链接。通过公告标识查找指定公告内容。
   3. mapper：因为公告详细内容需要提供附件下载功能，所以附件相关的内容需要查询出来，这里涉及到多表查询，需要自己定义mapper，根据公告ID查询公告相关的内容。
   4. model：通过分析，现有的model不支持公告和附件在同一类中，所以需要自己定义新的model，继承Affiche的同时，增加附件名和附件保存路径。同时在mapper中需要做对象的映射。观察AfficheMapper，由于affiche\_content是text类型，他对这个字段的映射做了处理，在上面自定义的mapper，仿照这做这个处理
   5. service：直接调用mapper，获取公告内容即可。
   6. controller：控制层逻辑比较复杂，除了要将内容设置在request域中，还需要提供下载附件的功能。我们拆分成两个controller来处理
      1. 第一个controller用来显示内容，即获取service中的公告内容，保存在request域中，然后转发至页面。将公告内容显示出来
      2. 第二个controller用来接受文件名和文件的存储路径，下载文件。
   7. 公告内容展示页。显示公告标题，公告发布时间，公告内容，发布者，附件名称。同时，将附件名称设置成超链接，请求上面分析的第二个controller，将附件名称和存储路径传过去。
   8. 开始打码后，发现上面分析的思路是行不通的，因为我们发布公告的时候，有时候是没有上传附件的，所以多表查询时使用主外键去重是不能得到想要结果的。现在决定改变上面的实现方式===========新实现方式如下=======================
   9. mapper层将公告和附件的查询拆开，直接用逆向工程生成的，上面扩展的mapper就不需要了，model层也不需要扩展了
   10. service层，也仅是调用mapper层的方法。
   11. 第一个controller就需要做分步处理了，先去公告内容，再判断公告model中的附件ID存不存在，存在再查附件，不存在就不需要查附件了。
   12. 问题出现：（6）中分析的，通过get方式传递附件存储路径是有问题的，我们可以发现，文件的存储路径是类似这样的，那将这个内容作为URL的一部分，会被解析成URL，这样跟请求映射的URL就匹配不上了。我们只能换种方式了。

E:\code\2017\IDEAProjects\Javatribe\javatribe\target\javatribe-1.0.0\WEB-INF\uploadFiles\20170330\3A4B80F77564406CA880719C26284FC5\_1idea.jpg

* 1. 附件的下载，需要传入附件的ID，然后查库，获取附件的相关内容，再下载。

### 公告修改

1. 公告修改的权限问题
   1. 超级管理员和管理员可以更改任何公告
   2. 普通成员只能更改自己发布过的公告。（这点可以通过控制程序逻辑来实现）
2. 公告修改的发起页面是公告列表。点击公告列表当中的修改，先从数据库中查询出该公告ID对应的内容，显示在一个表单中。此处还有一个限制，普通成员必须先判断是自己发布的公告才可以修改。然后修改是在从数据库取出的内容的基础上做的修改，对于附件的修改，需要先删除附件才可以显示出上传文件框。
3. 修改请求的发起
   1. 权限的设置，拥有修改公告权限的是普通成员和管理员超级管理员
   2. 首先需要在公告列表发起修改命令
   3. controller：需要先做角色判断。
      1. 首先从session中取出当前用户的角色信息，先分析现有session中的用户信息，发现没有用户角色这个信息，所以我们决定去修改之前的已实现的功能，在查找用户登录相关的内容的时候，把用户角色ID同时查出来，存到session中。
      2. 权限的限制，因为我们已经用粗颗粒度的权限对所有角色对修改公告的操作做了限制，所以细粒度下我们只需要对管理员和普通用户做限制即可。请求发过来后，先判断当前用户的角色信息

① 如果是普通成员，需要判断其发起的修改请求是不是自己发布的公告，不是则拒绝，是，则显示公告的内容。

② 如果是管理员或超级管理员，都可以修改。

* + 1. 这里同显示公告详情一样，需要将附件显示出来，所以多了一步查附件的内容。
  1. service：controller如果放行，发起的是根据公告ID查询公告相关的内容，所以service直接调用逆向工程的内容即可。
  2. mapper：逆向工程已经存在通过公告ID查公告内容的方法了。
  3. 页面：修改公告页面，需要对内容和附件做特殊处理
     1. 首先需要引入富文本编辑器。
     2. 附件：如果有附件，显示附件的内容和下载地址，没有附件，显示附件上传框
     3. 因为数据库中公告发布时间设置的是默认系统当前时间，当做了修改后时间也会做修改，所以修改公告的时候发布时间不需要显示出来了，即不需要用户去修改发布时间这个操作了。

1. 公告修改
   1. 首先分析公告修改的相关内容，公告修改先在页面上输入新公告的内容，点击修改发起修改操作。
      1. 但在页面上还有一步逻辑需要处理，如果是之前有附件的，同时又需要上传新的附件，则需要先删除现有的附件，才可以显示出上传文件框，再次上传新的内容上去。
      2. 在逻辑上的操作就有删除附件，上传新附件，修改公告内容。当然如果附件被修改了，需要手动获取新附件的ID修改到公告当中。
   2. 页面：如果要修改附件，先删除现有的附件再上传。先用JS写一个请求，获取当前附件ID，请求controller删除对应的附件和数据库中存储的附件信息。
   3. controller：
      1. 获取附件ID，执行删除附件操作，调用service中的方法。
      2. 修改公告内容跟发布公告很相似，也是用上传三步，获取公告的相关内容。普通表单项直接入model，文件表单项需要先做附件上传，再添加附件相关的内容入库。调用service的方法添加附件和修改公告。
   4. service：
      1. 删除附件文件和数据库中的附件信息，还要修改对应公告的附件ID为空。是同一个事务，要么同时成功，要么同时失败，需要引入事务。通过分析知道事务是真的数据库的，对删除附件文件这样的IO操作是不能实现事务的。所以将逻辑设计成，先用事务处理删除附件信息和修改对应公告的附件ID为空这两个数据库操作，如果这两个操作都成功再执行删除附件文件操作。

① 首先定义一个删除附件的方法，里面执行删除数据库中的附件信息和修改对应公告的附件ID为空，再删除附件文件

② 在spring-mybatis配置文件中创建事务管理器

③ 同时开启事务注解配置。

④ 然后在删除附件的方法上加入注解@Transactional

* + 1. 添加附件功能
    2. 修改公告，从controller获取到信息后还需要做额外的工作，需要将status设为1。
  1. mapper：根据ID删除附件，添加新附件，根据ID修改公告（分两种情况，一种是传入整个公告model，另一种是只修改公告的附件ID），前三个可以用逆向工程的，最后一个需要自己写mapper，专门写一个根据公告ID将附件ID改为null的

### 公告删除

(1) 超级管理员和管理员可以删除任何公告

(2) 普通成员只能删除自己发布过的公告。（这点可以通过控制程序逻辑来实现）

1. 公告删除的权限设置，只有上面这三个角色拥有删除权限。
2. 删除的发起页面是公告列表中的删除按钮，为防止误删，我们需要加入提示框。在公告类别的删除按钮加一个点击事件。
3. 在controller中，需要先判断发起删除操作的角色，对普通成员需要做限制。符合条件后调用service层执行删除操作
4. service层，将status设为0，然后调用mapper层执行删除操作。
5. mapper：由于公告被删除后，不删库，而是修改库中对应公告的状态信息。所以需要我们自己定义一个mapper，根据公告ID修改公告的status字段。

## 交流会风采

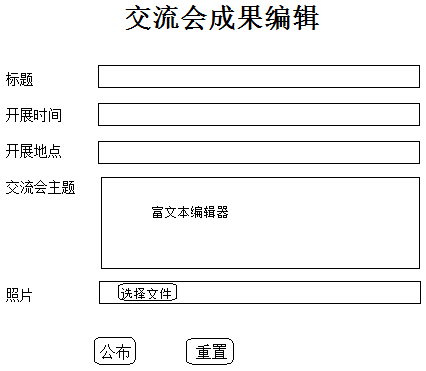
1. 爪哇部落是一个学习交流社团，常规活动中重中之重就是交流会的开展，通过开展交流会增进大家的了解，同时实现知识的共享和技术进步。每次交流会结束后，将成功展示出来以作纪念和记录。
2. 对交流会的风采展，系统提供阅读和编辑的功能。不同角色的人拥有的权限不同。
3. 毕业成员、考核人员、浏览者只可查看风采展。
4. 而超级管理员、管理员、普通成员可查看和编辑风采展。
   1. 风采展的编辑包括：风采展的公布、风采展的修改、风采展删除。
   2. 在细粒度下，超级管理员和管理员可以对任意风采展做编辑
   3. 而普通成员只可对自己编辑的风采展做修改和删除。

### 后台首页先显示出相应的交流会风采的菜单和页面

1. 直接做好数据库的测试数据即可。将权限和权限角色做好对应。
2. 权限设置
   1. 交流会风采 menu
   2. 拥有这个权限的角色有：超级管理员、管理员、普通成员、毕业成员、考核人员
3. 写一个页面，专门管理交流会风采的相关超链接。
   1. controller的名字是CommunicationShowController请求路径是/showCommunication.
   2. 页面名称是communicationShow.jsp

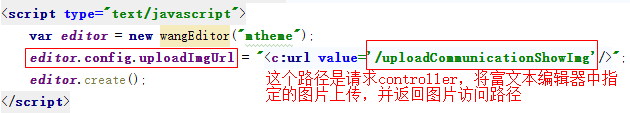
### 交流会风采公布

1. 交流会举办后需要部落成员将风采展公示出来，如何排版编辑由成员自行决定，但有些内容是必需的。
2. 需要编辑的内容包括：交流会标题，交流会开展时间，交流会开展地点，交流会主题，相关照片，编辑者，编辑时间。
3. 之前设计的数据库不太合理，我们需要做修改，将member\_id改成member\_name，同时加入编辑时间editTime，图片集的长度改成1024，将交流会开展时间换成字符型
4. 数据库做了修改后，需要对这个数据库做重逆向工程。
5. 权限设置，拥有这个权限的是超级管理员、管理员、普通成员
6. 交流会风采编辑页面示意图



* 1. 对这个页面的处理逻辑有点复杂。主要是在富文本编辑器中。交流会成果的编辑可能是图文并茂的内容。富文本编辑器默认插入的是网络上的图片，但我们一般是在本地中挑选图片做上传的。根据富文本编辑器的官方文档，我们先对富文本编辑器做扩展，加入上传本地图片的支持。

1. 交流会风采编辑页面添加图片上传功能
   1. 在富文本编辑器中，可以插入本地图片。为实现这个功能，需要做如下工作
   2. 在页面中，在富文本编辑器创建前，需要先设置图片上传URL，指向后端的一个方法进行图片上传处理。



* 1. 在controller中，提供相应的方法，进行图片的上传，并把上传后的图片的访问路径返回给前端。这个controller方法就只对图片做上传处理，没有其他逻辑。
  2. 这样，前端得到了该图片的访问路径，就可在前端中将图片显示出来了。

1. 交流会风采编辑页面的显示
   1. 在communicationShow.jsp提供请求超链接
   2. 在controller中将交流会风采编辑页面显示出来。
   3. 按在上面示意图编写页面
2. 交流会风采公布
   1. mapper：直接使用逆向工程添加交流会相关内容
   2. service：直接调用mapper实现添加操作
   3. controller：前端接收部分数据，发布者从session中取出。
   4. 页面，type的name和model的属性名要对应。

### 交流会风采展览

1. 首先还是权限的设置
   1. 19 交流会风采展览 permission communicationShow:query 17

/queryCommunicationShow

* 1. 拥有这项权限的角色有1/2/3/4/5
  2. 设置对于数据库的测试数据

1. 交流会风采展览的方式有多种
   1. 列表方式，展示所有，按公布时间倒序显示，同时需要分页。
   2. 具体每一次交流会的风采详情。
   3. 条件检索方式：检索条件有按标题，按开展时间，按编辑者。查询出来的还是用列表方式显示出来，时间倒序，分页
   4. 第（3）点中带条件的分页目前还没有解决方案，先放下。
2. 列表展示所有交流会风采的实现
   1. 列表展示的内容有：交流会标识，交流会标题，交流会开展时间，交流会开展地点，编辑器
   2. 请求发起的页面是communicationShow.jsp，在这个页面中设置一个超链接，默认请求第一页。同时加入权限限制
   3. mapper：需要自己写，涉及到分页，还有一个就是查询总记录数
      1. 向外提供的接口是，接受分页model，需要该对象中的起始下标和每页显示数。
      2. sql语句，查询出上面要展示的内容，同时按发布时间倒序排列，分页查找出来。
      3. 引用的resultMap是CommunicationMeetingMapper逆向工程的ResultMapWithBLOBs
   4. service：对分页model做处理
      1. 向外提供的接口是接收当前页
      2. 创建一个分页model，为其设置当前页，每页记录数，总页数从mapper中查询出来再设置进去。
   5. controller：
      1. 获取当前页，调用对应的service获得带结果的分页model
      2. 为分页model设置URL
      3. 将结果存储，并转发到页面中显示
   6. 交流会风采展览
      1. 列表的页面是communicationShowList.jsp
      2. 用一个表格展示结果。
      3. 还需要引入分页展示。
3. 交流会风采详情展示的实现
   1. 发起页面是communicationShowList.jsp列表，点击某个标题，展示对应那次交流会的具体内容，是根据当前交流会标识查找具体信息。
   2. mapper：直接用逆向工程，根据ID查记录
   3. service：直接调用逆向工程的mapper即可。
   4. controller：接收指定的ID。
   5. 页面，将出来照片路径外的其他内容都显示出来。

### 交流会风采修改

1. 交流会风采修改的发起页面是交流会风采展览列表中的修改按钮。
2. 权限
   1. 超级管理员、管理员、普通成员都可修改交流会风采内容
   2. 但是普通成员只能修改自己公布的交流会风采内容。而管理员们则对所有的交流会风采内容都有修改权限。
   3. 这个细粒度的权限限制用代码来控制。
3. 数据库权限数据的填充

交流会风采修改 permission communicationShow:update 17 /updatecommunicationShow

1/2/4 20

1. 修改页面的显示
   1. communicationShowList.jsp列表的修改按钮中，设置请求URL，/updatecommunicationShow/{交流会风采标识}
   2. controller中，
      1. 需要先根据交流会风采标识查询交流会风采的内容，需要其中的编辑者
      2. 从session中取得当前用户的用户名和角色标识
      3. 如果角色标识是4，并且当前用户名和编辑者名不相等，则不具备修改权限，转跳至status页面，显示您没有修改交流会风采内容的权限提示。
      4. 如果是管理员们，则进入修改展示页面updateCommunicationShow.jsp，同时将取出的交流会风采内容存到Model中。
   3. service：调用逆向工程的根据ID查交流会风采内容的mapper。
   4. mapper：直接使用逆向工程生成的。
   5. 页面：是一个表单，显示标题，时间，地点，内容，编辑者。其中内容需要引入富文本编辑器，而且同样要提供上传图片的功能。
2. 交流会风采修改功能实现
   1. 修改的发起页面是updateCommunicationShow.jsp，页面的name和model的属性要对应好。ID用隐藏域传去controller
   2. controller：接收页面的参数，调用service的方法，修改内容
   3. service：调用逆向工程的修改交流会风采的mapper
   4. mapper：直接用逆向工程的。

### 交流会风采删除

1. 交流会风采删除采用的方式是直接删库。
2. 权限的设置
   1. 拥有删除权限的是管理员们和普通成员
   2. 设置数据库的测试数据
3. 请求发起页面是communicationShowList.jsp的删除按钮，想后端controller传入交流会风采id，同时加入一个确认提示框
4. controller
   1. 对删除权限做细粒度处理。
   2. 首先获取ID值查找对应的交流会风采详情。
   3. 如果当前用户角色标识是4，并且删除的不是当前用户编辑的，则拒绝访问。
   4. 否则，调用service删除交流会风采内容。
5. service：调用逆向工程生成的根据ID删除交流会风采内容。
6. mapper：直接使用逆向工程

### 权限配置

1. 权限相关的信息已经从数据库中取出并存在了session域的user中。
2. 直接使用shiro的注解即可。
3. 权限配置主要是controller的URL做拦截
4. 以及在交流会风采的起始页面做拦截

## 考核平台

## 后台完善

### 问题提出

菜单列表显示需要重新设计数据库的测试数据

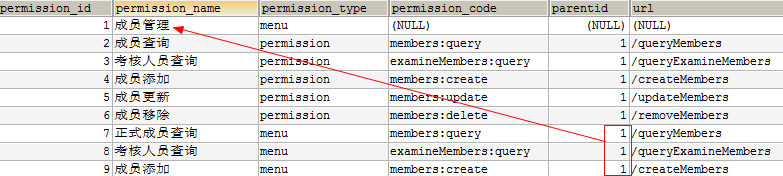
1. 目前后台首页所实现的功能还有一点小bug。



1. 可以看到，左侧的菜单栏中，是取出permission\_type为menu的来做一级菜单，而取permission\_type为permission来做二级菜单，这样是有问题的。permission是对请求URL做拦截的，而菜单只是用户能点击的可视的最粗粒度的内容。用户还有许多permission被隐藏在页面中，而不仅仅在菜单中显示。
   1. 就拿上面现有的内容来看，成员更新和成员移除，是在查询出来的成员当中来进行的，他们在没有做扩展之前，是依赖于某一个现有成员的，而不是凭空或凭操作者的一般记忆的。所以目前来说，将他们取出作为一个菜单是有问题的。
   2. 出现这样的情况原因在，我们取这部分内容的时候，先取数据库中的menu做一级菜单，而去permission做二级菜单，目前测试数据还很少，一旦permission的内容太多，是不是都要做为二级菜单了？这个设想是很可怕的，因为概念不清晰。
   3. 所以我们现在需要做区分，做分离。我们单独设用户可以点击的菜单，和用户所有的权限列表。
2. 菜单列表：成员管理，正式成员查询，考核人员查询，成员添加。这些才是菜单。当然他们有子父关系，我们看看如何去更改数据库的测试数据。如下
3. 权限列表：目前针对成员管理，用户拥有的权限有：正式成员查询，考核人员查询，成员添加，成员更新，成员移除。
4. 这样设计后，发现好像会出现冗余，但我觉得这种冗余是必需的，因为我们必需把菜单和权限做好分离。

### 后台完善的实施

1. 现在我们先开始设计数据库的测试数据，如下图。



1. 权限的测试数据设计好后，角色和权限的关系也要做更新。对于新加的这三个权限数据，需要添加他和角色的关系
2. 对mapper层，之前的mapper层是根据用户名查找用户拥有的权限菜单列表，不需要修改。
3. service层。
   1. 之前的service是没有做特殊处理的，直接调用mapper的方法获取菜单列表。因为他把一级和二级的菜单交给了realm来处理，当初那样设计是合理的，因为那时的一级二级关系是分别从菜单和权限中查的，不宜在service上做处理。
   2. 但现在因为已经分清楚了权限和菜单的关系，所以在service层中我们需要做好子父菜单的映射。
   3. 先对Menus做小修改，之前的Menus在继承Permission的基础上，还加了了一个List<Permission>属性，是用来装子菜单的，但当初的成员变量名可读性不再友好，所以做个小修改。跟数据库没影响，所以mapper不用修改。
      1. 改动前：



* + 1. 改动后：



* 1. service层，子父菜单的映射。从mapper中查询出来的是某个用户的所有菜单，没有分子父关系。这里因为父菜单会有多个，所以新定义一个父菜单集合，专门存放parentid为null的菜单内容。这样父菜单集合就找到了
  2. 然后遍历制作好的父菜单集合，和从数据库中查询出来的所有菜单做比较，如果父菜单列表中的权限标识和从数据库中取出的菜单的父标识是相同的，就找到了子父关系了。

1. realm：只需要修改菜单列表的获取，不再需要做映射，直接从service层取。
2. 页面welcome.jsp：菜单显示需要做小量修改，因为Menus类的成员变量名称做了小修改。

# 授权

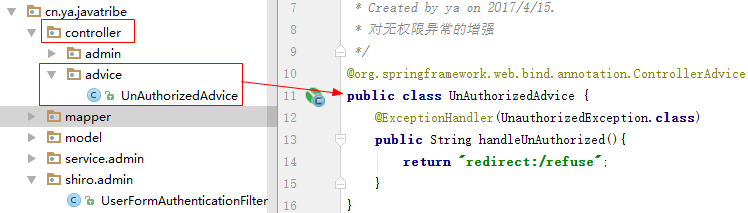
1. 现在我在页面中随便输入一个URL，当然前提是访问这个系统的有效URL，会出现这样的情况。



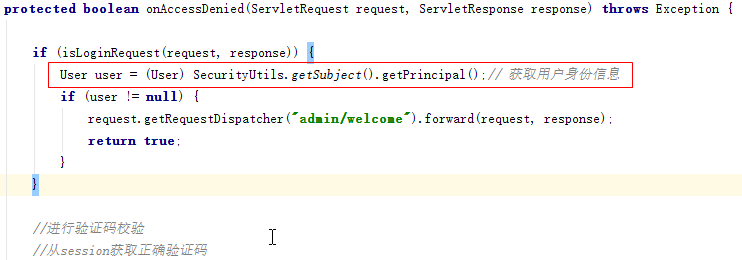
1. 发现不用登录，而且什么人都可以查看，这样的系统是不安全的，我们需要对他做权限拦截。
2. 授权的实现如下
3. 首先，需要从数据库中根据用户查找用户拥有的权限列表，然后service层需要将这些内容调出来。
4. 授权信息是设置在realm中的，重写doGetAuthorizationInfo方法进行授权。会在用户认证通过后，将用户的授权信息保存起来。
   1. 在这个方法中，需要先获取主身份信息principals.getPrimaryPrincipal();这个主身份信息是用户认证通过后设置给SimpleAuthenticationInfo的。
   2. 然后去service中的权限信息
   3. 因为做授权时，只需要权限代码，而从service中取出来的有许多多余的信息，所以我们单独去出权限代码，把他扔进一个List集合中。
   4. 将授权信息填充到simpleAuthorizationInfo对象中（simpleAuthorizationInfo.addStringPermissions(权限信息);）
5. 在springmvc.xml中配置，来开启controller类aop支持
6. 在controller中的方法当中加上权限注解，@RequiresPermissions("权限信息")，如：@RequiresPermissions("members:query")，表示执行某方法需要指定的权限。
7. 当没有权限被拒绝时，跳转到指定页面，通过spring-shiro.xml来设置。

# 后台管理遗留的问题

1. 授权拦截，当访问不存在权限的路径是，在spring-shiro.xml设了拒绝跳转页面，但是不起作用，直接报错了。做个改进。
   1. 直接抛了异常，而没有做跳转，可能是shiro的实现细节我们没弄懂，那不妨自己来做处理，也就是自己捕获异常，然后手动跳转。
   2. 我们再来分析下，为什么会抛异常，哪里的操作会导致抛这个异常。我们知道这个异常是无权限异常，而权限，主要是在controller中的，@RequestMapping对应的请求路径，如果没有权限，则会抛无权限异常。这样分析，我们可以看出，这个异常处理，是真的所有的controller，以及其中的所有带@RequestMapping的方法。
   3. 如何做到上面提出的统一处理呢？这个有点像AOP，再仔细回想，Spring中有一个叫控制器通知的。好像正符合我们的要求。
   4. 控制器通知，是定义一个类，这个类被@ControllerAdvice注解，这样controller层中的每一个controller都要扩展这个类，那我在这个定义的类里面，写一个方法，并且用注解@ExceptionHandler修饰，那么当controller抛出异常的时候，都会先执行这个类的扩展方法，再执行controller自己的方法。那我们就在这个扩展方法中，做异常处理，直接转跳页面即可。
   5. 实现如下



1. 关于用户登录的问题（已解决）
2. 用户登录系统后，如果异常退出，即用户没有注销，其他用户登录不进去，为什么？
3. 同一台电脑，同一个浏览器，已经有一个用户登录后，当再打开一个标签页，想登录另一个人，登录页可以进入，但是登录无法成功。
   1. 分析原因是，因为之前已经有用户登录了，已经存在sessionID，再登一个，因为已经存在sessionID，所以不成功。而如果把上一个人登录后的sessionID删除后，那么后一个人是可以登录的。同时，如果上一个人登录后没有手动注销，也就是这上一个人还出在登录后的状态，后一个人要想登录成功而去删除了上一个人的sessionID，成功让自己登录进去后，此时，系统中上一个人和后一个人同时都是登录后的状态，这个情况更奇怪，浏览器中只保存一个sessionID，却同时有两个用户处于登录后的状态，并且都可以做登录后才能做的那些操作。。。额。。。这个问题好奇怪。
   2. 这样的设计本身就存在问题，在同一个浏览器中，已经有一个用户登录了，那么在sessionID没有过期的时候，就可以访问页面内所以登录后就可以访问的页面了，而且登录页面是不能再访问的了。也就是说，一个用户登录之后，登录页面是不能访问的，直接进入登录后的主页做其他操作即可。
   3. 但是引入shiro后，登录验证由shiro为我们做了，那么如何设置用户登录后登录页面不可访问呢？而直接进入登录后的主页？
   4. 突然想起，shiro提供了一个记住我的功能，也就是说，在一台电脑的一个浏览器中，某个用户登录后，系统记住这个用户，当用户再次用这台电脑的这个浏览器来访问这个系统时，直接进入登录后的主页了。加入这个功能看看，再测试，看能否避免上面那个问题。
   5. 但是再仔细想想，记住我这个功能是需要用户手动点击复选框后，才会开启这个功能的，如果用户手动开启这个功能，那么上面那个问题还是存在的，这种方法不能根本解决问题。
   6. 看看单点登录是怎样的，看能否找出一个灵感出来。
   7. 再不行，就看看能否对shiro的认证做改善，自己写个功能，让用户登录后，再次访问系统是，登录页是不可用的。
4. 对于（a）提出的问题，这里有新的解释，其实在（a）中的情况，并不是同时存在两个用户，其实那是同一个用户，只是因为上一个用户没刷新，导致页面中显示的是上一个用户的信息。如果将上一个用户所在的页面做个刷新，可以看到，此时页面中的用户信息是后一个人的，所以（a）中的情况奇怪在页面缓存上。
5. 针对上面问题的分析，和解决方案的提出，这里演示的是（g）中的解决策略
   1. 对shiro的认证做扩展。让用户登录后，直接访问登录后的页面，登录页不可再访问。
   2. shiro对认证的拦截，是使用FormAuthenticationFilter过滤器的，而加入验证码后，我用自己自定义的UserFormAuthenticationFilter继承了FormAuthenticationFilter过滤器。在自定义的UserFormAuthenticationFilter中，重写了onAccessDenied()方法，在进行表单验证前，做了验证码的校验。
   3. 而现在我要添加的功能是先看是否已有用户登录，有则直接转跳到登录后的成功页面。那如何知道是否已有用户登录呢？通过subject获取用户身份信息，有则表示有用户登录了，直接跳转到成功页。没有才做验证码校验和身份校验。
   4. 即在serFormAuthenticationFilter中的onAccessDenied()方法中添加如下逻辑



* 1. 然而再做测试，依然没法解决问题。

1. 第（4）中没法拦截用户登录URL。我们再去分析下源码，也许是我们对方法的增强增强错了。
   1. 我们知道，登录拦截器是FormAuthenticationFilter，也就是说，我们请求登录URL的时候，会先被过滤器拦截，做一些判断，能够访问才会被放行。回顾之前我们自己写的拦截器，是自定义一个类继承Filter，然后再复写Filter的onPreHandle()方法，现在，我们就看看，登录拦截器是如何对我们的请求做拦截的。也就是说，看FormAuthenticationFilter最顶层的哪个父类中有onPreHandle()方法做了哪些拦截处理。
   2. 我们查看源码，看我们自定义的UserFormAuthenticationFilter继承体系，看看他的哪个父类有onPreHandle()方法



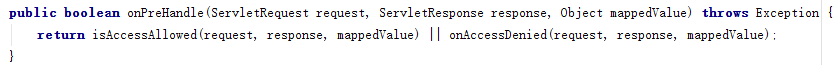




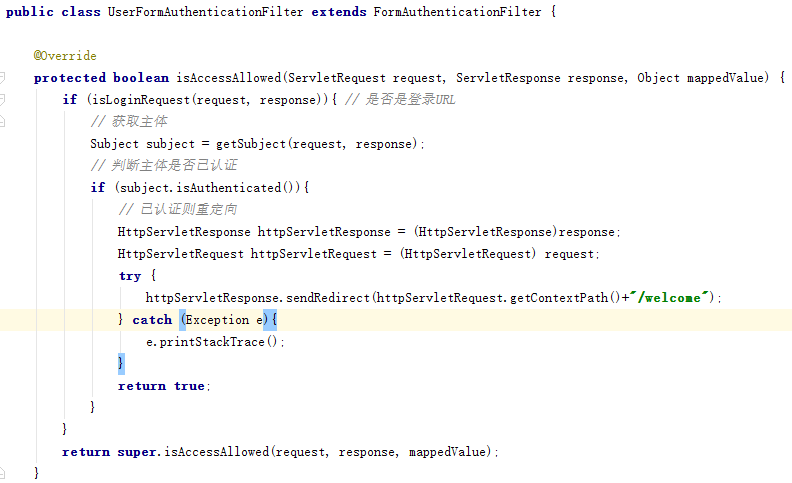




* 1. 查找到AccessControlFilter，我们发现这个类里面有个onPreHandle()方法，我们来看看这个方法的实现是怎样的



* 1. 我们可以看出拦截器的关键操作在isAccessAllowed()方法，如果这个方法返回true，那么就会放行，不会再走onAccessDenied()方法了。再反思我们目前项目的实现过程，在自定义的UserFormAuthenticationFilter中，我们只重写了onAccessDenied()方法，而根据onPreHandle()方法中的逻辑来看，onAccessDenied()方法可能根本不会执行，因为短路或中isAccessAllowed()方法做了限制。
  2. 根据方法名，我们可以很容易理解方法的作用。isAccessAllowed()方法表示访问是否允许。onAccessDenied()方法表示访问拒绝。从逻辑来看，onPreHandle()方法的判断逻辑设计是有原因的，因为我们做认证，本质上就是对访问的拒绝处理，如果认证不通过就解决，通过才放行。所以才把onAccessDenied()方法放在后面。
  3. 那如果我们要做的增强是，用户请求登录URL的时候，先不对他做拒绝处理，而是先看他是否已登录，如果已登录，是可以直接访问系统了，也就是允许访问，未登录才会对他做身份验证也就是拒绝处理。
  4. 所以，我们增强的逻辑是要写在isAccessAllowed()方法中的。在这个方法中，如果用户已经认证过了，就表示允许访问了，跳转到登录成功页中，并且返回的是false，其他情况就交由父类来处理吧。所以逻辑如下：UserFormAuthenticationFilter重写isAccessAllowed()方法。



1. 没有密码修改功能，后期加上。
2. 表单参数未做检

# 完善

1. 图片上传的时候做自动压缩
2. 交流会风采修改，如果把图片删除了，对应项目图片库里的图片文件也删除了他。

# 扩展

权限的分配也写一个后台管理出来