## 3.3 各功能需求分析与设计

### 3.3.1 登陆



图3.1登陆失败图

1、shiro登陆用户验证和授权

登陆由用户名和密码登陆。运用shiro框架进行权限的控制，自定义一个SystemAuthorizingRealm类，当用户登陆时，接收一个由用户名和密码组成的token进行验证，若是验证失败就会出现图3.1的提示信息，若是验证成功就查询该用户角色并授权给该用户。在页面上结合<shiro></shiro>标签，没有授权的功能该用户是不可视的。如图3.2，图3.3所示，陈艳只是普通的置业顾问，没有合作伙伴的添加和删除的权限，因此添加和删除的按钮就不可视，而admin是管理员，他具备所有的权限，所以他可见所有的功能。



图3.2



图3.3

2、登陆超时与匿名访问

登陆系统后若是超过30分钟没有操作，会话就会失效，若是想再操作系统就要重新登陆。

代码：

<!-- 自定义会话管理配置 -->

<bean id="sessionManager" class="com.tospur.common.security.shiro.session.SessionManager">

<property name="sessionDAO" ref="sessionDAO"/>

<!-- 会话超时时间，单位：毫秒 -->

<property name="globalSessionTimeout" value="${session.sessionTimeout}"/>

<!-- 定时清理失效会话, 清理用户直接关闭浏览器造成的孤立会话 -->

<property name="sessionValidationInterval" value="${session.sessionTimeoutClean}"/>

<!-- <property name="sessionValidationSchedulerEnabled" value="false"/> -->

<property name="sessionValidationSchedulerEnabled" value="true"/>

<property name="sessionIdCookie" ref="sessionIdCookie"/>

<property name="sessionIdCookieEnabled" value="true"/>

</bean>

session.sessionTimeout=1800000

session.sessionTimeoutClean=120000

若是没有登陆就访问除登陆其他的页面，系统会跳回登陆页面。只有登陆才可以操作。

代码：

${adminPath}/login = authc

${adminPath}/logout = anon

${adminPath}/\*\* = user

Principal p = UserUtils.*getPrincipal*();

**if** (p != **null** && !p.isMobileLogin()){

WebUtils.*issueRedirect*(request, response, getSuccessUrl(), **null**, **true**);

### 3.3.2 客户模块

客户管理模块分为购房客户和项目合作伙伴。

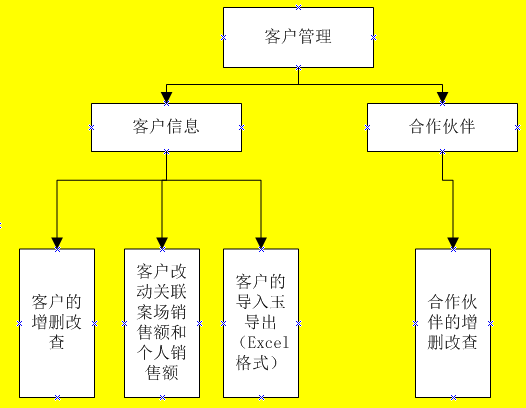


图3.3客户模块功能图

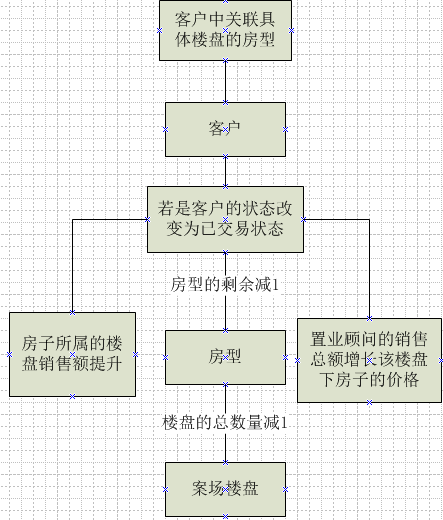


图3.4客户与目标户型关联图

一、客户信息

1、客户信息增删改查。结合shiro功能权限和数据权限不同的角色查询到客户数据有可能是不同的，增删改查的功能也不一定都可视。例如普通的案场顾问登陆进来只能查看到自己的客户的信息，而admin即管理员查询到的是所有的客户信息。另外客户的有三种状态：潜在客户、正在跟进客户、已交易客户。潜在客户：经过大数据的匹配的有意图卖房的客户；正在跟进客户：来案场咨询或是来电咨询客户；已交易客户：已进行金钱交易，有合同保障客户。一旦客户的状态改变为已交易客户，该客户的置业顾问的销售额会添加对应客户选中的案场房型的金额值，案场的房型数量也会减少1，本季度的案场销售额亦添加相应的金额。客户信息中包含的楼盘属性与房型和案场联系人有关联，若是楼盘的变更，其目标户型和案场负责人也会随之变化。

代码：

**function** changeCases(obj){

**var** option=$(obj).find("option:selected");

**var** item = option.text();

**var** temp = option.attr("value");

console.log(temp);

$("#address").val($(option[0]).attr("address"));

//发送请求

$.post('${ctx}/customer/customer/getHouses',{name:temp},**function**(data){

console.log(data);

console.log(data.typeList);

**var** typeList = data.typeList;

**var** teamList = data.teamList;

$("#target > \*").remove();

**for**(**var** i=0;i<typeList.length;i++){

**var** t = $("<option value="+typeList[i]+" >"+typeList[i]+"</option>")

$("#target").append(t);

}

$("#contacts > \*").remove();

**for**(**var** j=0;j<teamList.length;j++){

**var** e = $("<option value="+teamList[j]+" >"+teamList[j]+"</option>")

$("#contacts").append(e);

}

});

2、客户的导入导出功能。

只能由案场联系人也就是案场负责人和admin才拥有。合伙人所有置业顾问都可以查看，只能由项目人添加、修改和删除。

代码：

/\*\*

\* 导出excel文件

\*/

@RequiresPermissions("customer:customer:export")

@RequestMapping(value = "export", method=RequestMethod.***POST***)

**public** String exportFile(Customer customer, HttpServletRequest request, HttpServletResponse response, RedirectAttributes redirectAttributes) {

**try** {

String fileName = "客户"+DateUtils.*getDate*("yyyyMMddHHmmss")+".xlsx";

Page<Customer> page = customerService.findPage(**new** Page<Customer>(request, response, -1), customer);

**new** ExportExcel("客户", Customer.**class**).setDataList(page.getList()).write(response, fileName).dispose();

**return** **null**;

} **catch** (Exception e) {

addMessage(redirectAttributes, "导出客户记录失败！失败信息："+e.getMessage());

}

**return** "redirect:"+Global.*getAdminPath*()+"/customer/customer/?repage";

}

/\*\*

\* 导入Excel数据

\*/

@RequiresPermissions("customer:customer:import")

@RequestMapping(value = "import", method=RequestMethod.***POST***)

**public** String importFile(MultipartFile file, RedirectAttributes redirectAttributes) {

**try** {

**int** successNum = 0;

**int** failureNum = 0;

StringBuilder failureMsg = **new** StringBuilder();

ImportExcel ei = **new** ImportExcel(file, 1, 0);

List<Customer> list = ei.getDataList(Customer.**class**);

**for** (Customer customer : list){

**try**{

customerService.save(customer);

successNum++;

}**catch**(ConstraintViolationException ex){

failureNum++;

}**catch** (Exception ex) {

failureNum++;

}

}

**if** (failureNum>0){

failureMsg.insert(0, "，失败 "+failureNum+" 条客户记录。");

}

addMessage(redirectAttributes, "已成功导入 "+successNum+" 条客户记录"+failureMsg);

} **catch** (Exception e) {

addMessage(redirectAttributes, "导入客户失败！失败信息："+e.getMessage());

}

**return** "redirect:"+Global.*getAdminPath*()+"/customer/customer/?repage";

}