软件测试文档

一、系统界面

1. 小程序界面
2. 小程序首页

进入小程序,回显当前系统中未租售出的房屋信息。点击右下角按钮可以上传房屋信息，点击列表可以进入房屋详情。如果是未授权登陆用户或未认证用户，则会弹出提示框“您还没有认证！”。

1. 筛选过滤

列表上方可以进行常规筛选或进行细粒度筛选。

1. 房屋详情

点击列表进入房屋详情，可以看到有关房屋的信息。点击下方收藏按钮可以收藏心仪的房屋，联系他可以通过电话的方式联系对方。



1. 授权登陆及认证

未认证用户需要填写个人信息，上传学生卡或教工卡后点击认证。Web端的后台管理平台可以对用户信息和房屋信息进行审核，对于违规信息或非天大用户进行删除和驳回处理。



1. 个人中心

已微信授权登陆且认证后，可以点击我的收藏和我的房屋进行管理。



1. 我的收藏

添加至我的收藏后，所有信息在这里可以看到。

2．Web前端界面

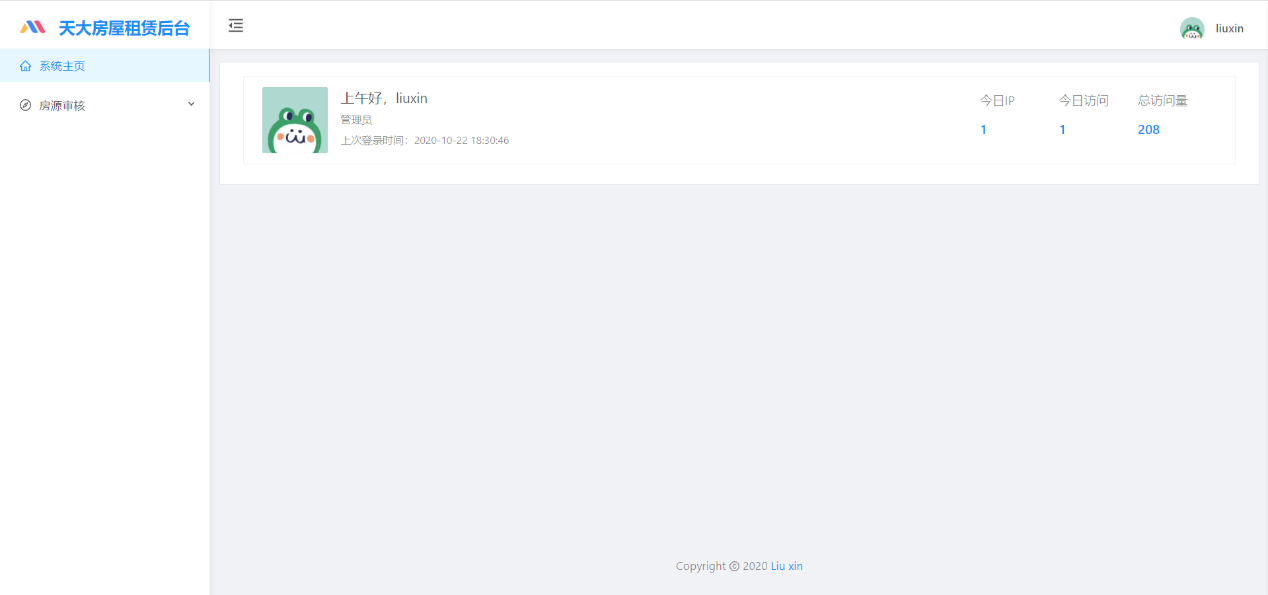
如图所示为系统的登录页面 (首页),页面采用单一色彩的方式,简洁明了。其中,登陆和注册按钮位于输入表单的下方,符合管理员用户使用习惯。



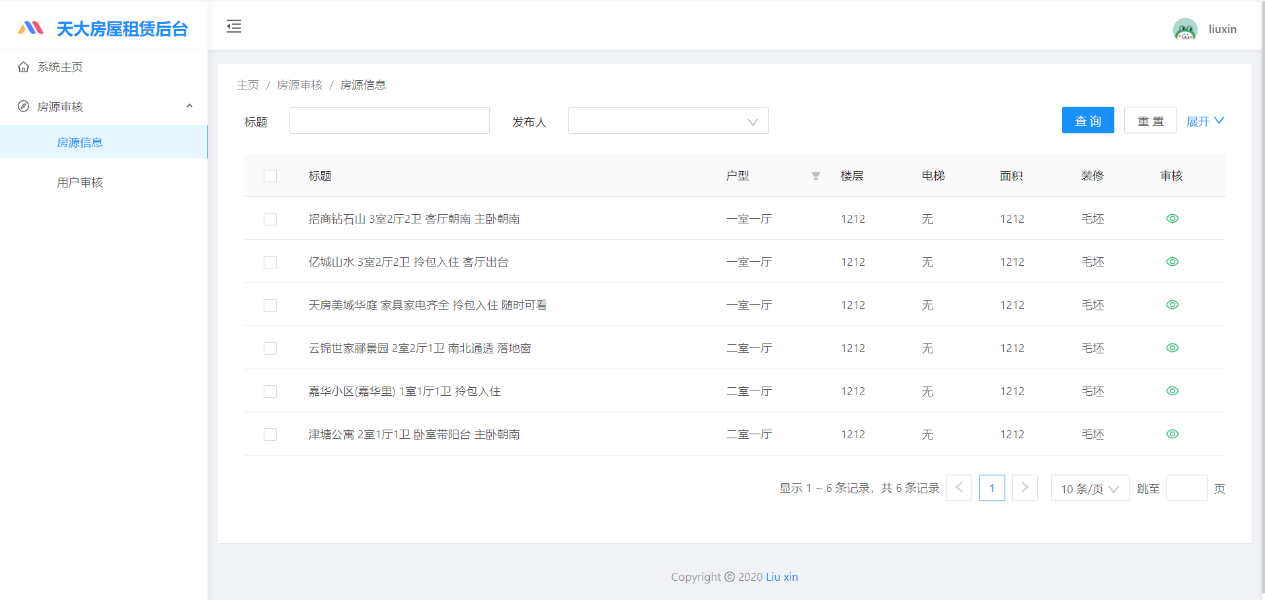
当管理员用户需要注册账户时，可以点击注册账户，之后会跳转到注册页面，如图所示为系统的注册页面。



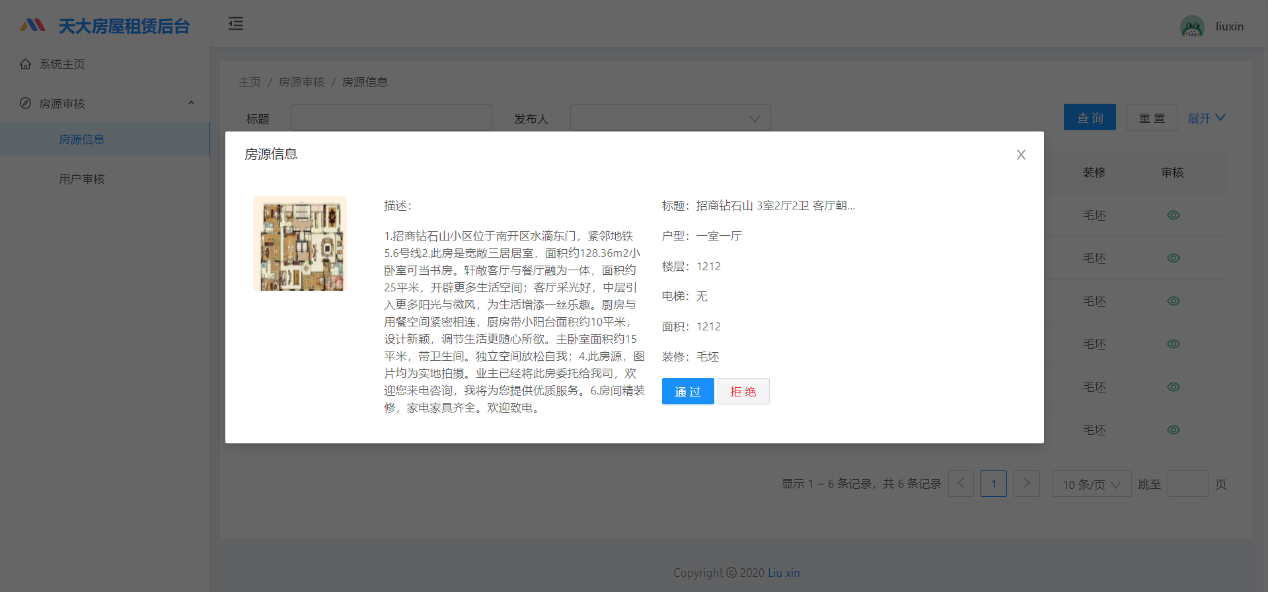
当管理员用户成功登录时，系统会跳转到系统的主页。显示用户的登录状态



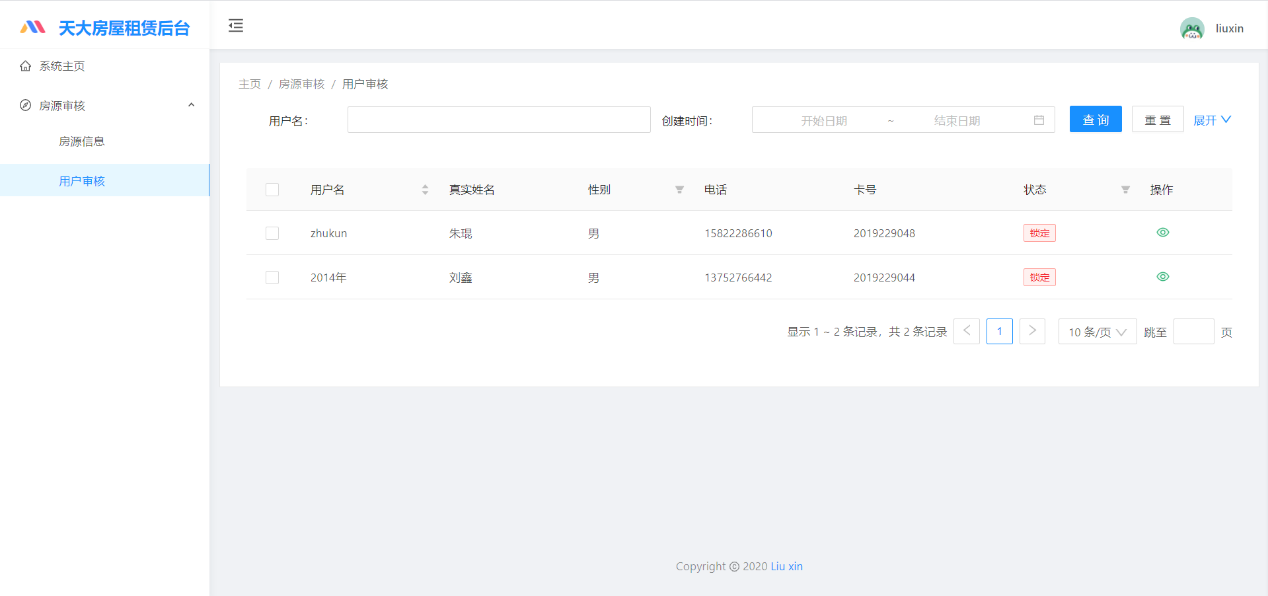
管理员用户可以在左侧列表中选择房屋审核，包括有房源信息和用户审核，点击房源信息会显示需要审核的房源信息，包括标题、户型、楼层、电梯、面积、装修和查看详细信息按钮。



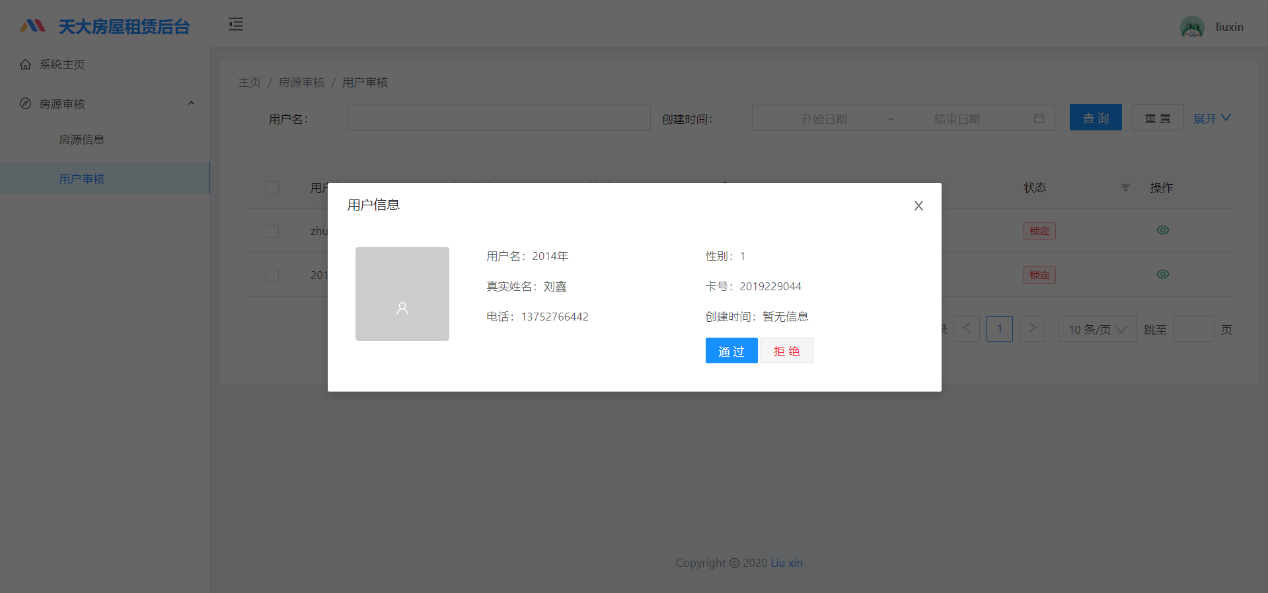
管理员用户点击详细信息后会显示房屋的详细信息。包括房屋的图片（如户型图等）。房屋的详细描述以及标题、户型、楼层、电梯、面积、装修等。此外还有通过和拒绝按钮，管理员用户可以在确认用户发布的房屋信息没有无用信息和有害信息后选择通过审核，否则需要拒绝，需要房源信息的发布者重新修改信息并再次审核。



管理员用户需要审核用户信息时可以选择用户审核。用户信息包括用户名、真实姓名、性别、电话、卡号、状态以及查看用户详细信息并操作等。



用户点击操作后，会显示用户的详细信息包括头像，用户名，真实姓名，电话，性别，卡号和创建信息。管理员用户在确认用户信息中没有无用信息和有害信息可以通过审核，否则应该拒绝审核并让用户重新提交信息。



1. 系统实现模块说明

系统本着高内聚低耦合的原则，主要划分为7个模块。

## 2.1 模块1（身份确认模块）

## 2.1.1 描述

身份确认模块分为微信授权、用户认证和管理员审核三个部分。微信授权点击确认即可，用户认证通过输入正确的学生/教工信息认证，后台管理员针对用户提交的信息确认是否为天大学生/教工采取同意、修改、驳回三个操作。

## 2.1.2 主要代码

/\*

\* @author liuxin、zhukun

\* 获取微信的session\_key 、openid等

\* \*/

@GetMapping(value = "/getSessionKey")

@PermitAll

public FebsResponse getSessionKey(String code) throws FebsException {

try {

String data = HttpUtil.sendPost(FebsConstant.SESSION\_KEY\_URL, "appid=" + FebsConstant.APP\_ID + "&secret="+FebsConstant.APP\_SECRET+"&js\_code="+ code + "&grant\_type=authorization\_code");

return new FebsResponse().data(data);

} catch (Exception e) {

String message = "SessionKey查询失败";

throw new FebsException(message);

}

}

/\*

\* @author liuxin、zhukun

\* 用户注册

\* \*/

@PostMapping("/userRegister")

public FebsResponse userRegister(@RequestBody User user) throws FebsException {

try {

Example example = new Example(User.class);

Example.Criteria criteria = example.createCriteria();

criteria.andEqualTo("openid",user.getOpenid());

List<User> userList = userService2.selectByExample(example);

if (userList.size() == 1){

return new FebsResponse().data(userList.get(0));

}else if (userList.size() == 0){

user.setIsChecked(0);

user.setCreateAt(String.valueOf(System.currentTimeMillis()));

userService2.save(user);

return new FebsResponse().data(user);

}

throw new FebsException("failed");

}catch (Exception e){

e.printStackTrace();

String message = "注册失败！";

throw new FebsException(message);

}

}

/\*

\* @Author liuxin、zhukun

\* 更新用户信息

\* \*/

@PostMapping("updateUserInfo")

public FebsResponse updateUserInfo(

@RequestBody User user,

@RequestParam(value ="authPic", required = false) MultipartFile cardPic,

HttpServletRequest request) throws FebsException {

try {

if (cardPic!=null && !cardPic.isEmpty()){

String fileName = cardPic.getOriginalFilename();

String filePath = UPLOAD\_PATH;

File dest = new File(filePath);

try {

if (!dest.getParentFile().exists()) {

dest.getParentFile().mkdirs();// 新建文件夹

}

cardPic.transferTo(dest);

System.out.println("上传成功！");

return new FebsResponse().message("success").data(filePath+fileName);

} catch (IOException e) {

System.out.println("上传异常！" + e);

throw new FebsException("上传异常！" + e);

}

}

System.out.println(user);

userService2.updateNotNull(user);

return new FebsResponse().message("success").data(user);

}catch (Exception e){

e.printStackTrace();

String message = "更新失败！";

throw new FebsException(message);

}

}

/\*

\* @Author liuxin、zhukun

\* 管理员审核用户

\* \*/

@PutMapping("checkUser")

public int checkUser(@RequestBody User user){

try {

return userService2.updateNotNull(user);

}catch (Exception e){

e.printStackTrace();

return 0;

}

}

## 2.2模块2（房屋发布模块）

## 2.2.1 描述

房屋系统模块分为房屋发布模块、查询全部房屋任务、房屋审核部分。

房主能够填写自己想要发布和管理的房屋信息，包括租赁和售卖两种类型。

房主和用户都可查询、筛选所有已经发布的房屋信息。

管理员能够对房主发布的房屋信息进行审核。

## 2.2.2 主要代码

@PostMapping("/addRent")

public int addRent(@RequestBody Rent rent) throws FebsException {

try{

System.out.println(rent.toString());

rent.setIsDeleted(0);

rent.setCreateAt(String.valueOf(System.currentTimeMillis()));

rent.setIsChecked(0);

rent.setIsRented(0);

rent.setIsDisplayed(0);

return rentService.save(rent);

}catch (Exception e){

e.printStackTrace();

throw new FebsException("failed");

}

}

@PostMapping(value = "addSale")

public FebsResponse addSale(@RequestBody Sale sale) throws FebsException {

try{

sale.setCreateAt(String.valueOf(System.currentTimeMillis()));

sale.setIsDeleted(0);

sale.setIsSaled(0);

saleService.save(sale);

return new FebsResponse().data("success");

}catch (Exception e){

e.printStackTrace();

throw new FebsException("failed");

}

}

/\*

\* @Author liuxin、zhukun

\* 筛选出正在审核且没有租出去的房屋

\* \*/

@GetMapping(value = "rentCheckingList")

public List<Rent> querySaleCheckingList() throws FebsException {

try{

Example example = new Example(Rent.class);

Example.Criteria criteria = example.createCriteria();

criteria.andEqualTo("isChecked",0);

criteria.andEqualTo("isRented",0);

return rentService.selectByExample(example);

}catch (Exception e){

e.printStackTrace();

throw new FebsException("failed");

}

}

/\*

\* @Author liuxin

\* 筛选出审核通过，且没有出售的房屋

\* \*/

@GetMapping(value = "saleCheckedList")

public List<Sale> querySaleCheckedList(@RequestParam Map<String,String> params) throws FebsException {

try{

Example example = new Example(Sale.class);

Example.Criteria criteria = example.createCriteria();

criteria.andEqualTo("isChecked",1);

criteria.andEqualTo("isSaled",0);

for (String key:params.keySet()) {

if (key.equals("title"))

criteria.andLike(key,params.get(key));

else if (key.equals("type"))

continue;

else if (key.equals("area"))

example.setOrderByClause("acreage " + params.get(key));

else if (key.equals("price"))

example.setOrderByClause("totalPrice " + params.get(key));

else

criteria.andEqualTo(key,params.get(key));

}

return saleService.selectByExample(example);

}catch (Exception e){

e.printStackTrace();

throw new FebsException("failed");

}

}

/\*

\* @Author liuxin

\* 租房详情

\* \*/

@GetMapping(value = "rentDetail")

public List<Rent> querySaleDetail(@RequestParam int rentId) throws FebsException {

try{

Example example = new Example(Rent.class);

Example.Criteria criteria = example.createCriteria();

criteria.andEqualTo("rentid",rentId);

return rentService.selectByExample(example);

}catch (Exception e){

e.printStackTrace();

throw new FebsException("failed");

}

}

/\*

\* @Author liuxin

\* 删除某个租房信息

\* \*/

@DeleteMapping(value = "deleteRent")

public int deleteSale(@RequestParam int rentId) throws FebsException {

try{

return rentService.delete(rentId);

}catch (Exception e){

e.printStackTrace();

throw new FebsException("failed");

}

}

/\*

\* @Author liuxin

\* 更新某个租房信息

\* \*/

@PutMapping(value = "updateSale")

public int updateSale(@RequestBody Rent rent) throws FebsException {

try{

return rentService.updateNotNull(rent);

}catch (Exception e){

e.printStackTrace();

throw new FebsException("failed");

}

}

/\*

\* @Author liuxin

\* 售卖详情

\* \*/

@GetMapping(value = "saleDetail")

public List<Sale> querySaleDetail(@RequestParam int saleId) throws FebsException {

try{

Example example = new Example(Sale.class);

Example.Criteria criteria = example.createCriteria();

criteria.andEqualTo("saleid",saleId);

return saleService.selectByExample(example);

}catch (Exception e){

e.printStackTrace();

throw new FebsException("failed");

}

}

/\*

\* @Author liuxin

\* 删除某个售卖信息

\* \*/

@DeleteMapping(value = "deleteSale")

public int deleteSale(@RequestParam int saleId) throws FebsException {

try{

return saleService.delete(saleId);

}catch (Exception e){

e.printStackTrace();

throw new FebsException("failed");

}

}

/\*

\* @Author liuxin

\* 更新某个售卖信息

\* \*/

@PutMapping(value = "updateSale")

public int updateSale(@RequestBody Sale sale) throws FebsException {

try{

return saleService.updateNotNull(sale);

}catch (Exception e){

e.printStackTrace();

throw new FebsException("failed");

}

}

/\*

\* @Author liuxin、zhukun

\* 筛选出正在审核且没有租出去的房屋

\* \*/

@GetMapping(value = "rentCheckingList")

public List<Rent> querySaleCheckingList() throws FebsException {

try{

Example example = new Example(Rent.class);

Example.Criteria criteria = example.createCriteria();

criteria.andEqualTo("isChecked",0);

criteria.andEqualTo("isRented",0);

return rentService.selectByExample(example);

}catch (Exception e){

e.printStackTrace();

throw new FebsException("failed");

}

}

/\*

\* @Author liuxin

\* 更新某个租房信息

\* \*/

@PutMapping(value = "updateRent")

public int updateRent(@RequestParam int rentid, @RequestParam int isChecked) throws FebsException {

try{

Rent rent = new Rent();

rent.setRentid(rentid);

rent.setIsChecked(isChecked);

return rentService.updateNotNull(rent);

}catch (Exception e){

e.printStackTrace();

throw new FebsException("failed");

}

}

/\*

\* @Author liuxin

\* 选出正在审核的房屋信息

\* \*/

@PostMapping(value = "saleCheckingList")

public List<Sale> querySaleCheckingList(@RequestBody Map<String,String> params) throws FebsException {

try{

Example example = new Example(Sale.class);

Example.Criteria criteria = example.createCriteria();

criteria.andEqualTo("is\_checked",0);

criteria.andEqualTo("is\_saled",0);

for (String key:params.keySet()) {

if (!key.equals("title"))

criteria.andEqualTo(key,params.get(key));

criteria.andLike(key,params.get(key));

}

return saleService.selectByExample(example);

}catch (Exception e){

e.printStackTrace();

throw new FebsException("failed");

}

}

/\*

\* @Author liuxin

\* 更新某个售卖信息

\* \*/

@PutMapping(value = "updateSale")

public int updateSale(@RequestBody Sale sale) throws FebsException {

try{

return saleService.updateNotNull(sale);

}catch (Exception e){

e.printStackTrace();

throw new FebsException("failed");

}

}

## 2.3模块3（房屋收藏模块）

## 2.3.1 描述

用户可以将心仪的房屋信息添加至我的收藏，方便日后寻找。

## 2.2.2 主要代码（略）

# 3．系统测试

首先进行测试方案说明，包括主要测试阶段划分，测试计划，每个阶段测试方法、测试工具等。

## 3.1 模块测试

## 3.1.1功能模块清单

如下表表3-1所示。微信授权、用户认证和管理员审核

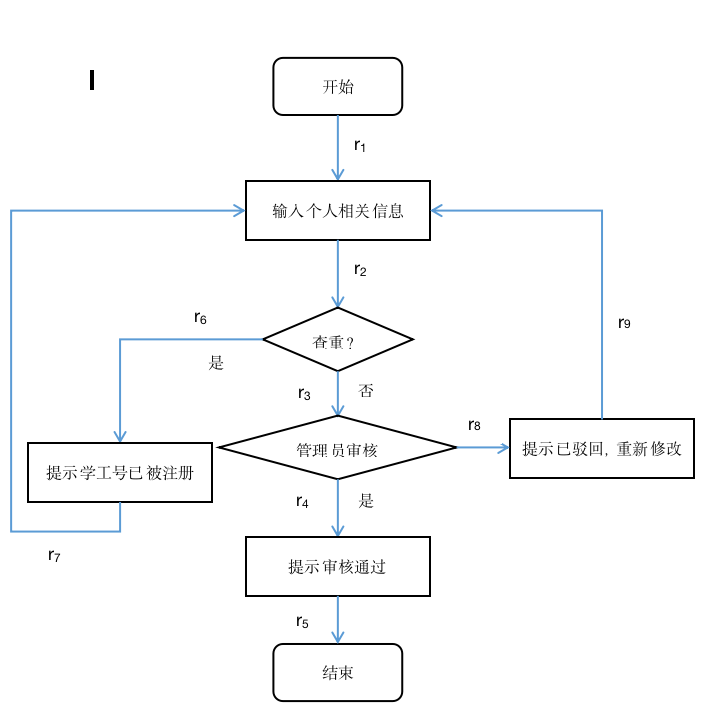
表3-1 模块清单

|  |  |  |  |
| --- | --- | --- | --- |
| 编 号 | 模块（类） | 模块（类）功能简述 | 接口简述 |
| M 1-1 | User\_wxauth | 微信授权登陆模块 | 小程序通过微信授权登陆获取唯一标识openID |
| M 1-2 | User\_auth | 用户填写相关信息及天大学工卡照片发起认证申请 | 与主界面有接口，通过session保持通话 |
| M 1-3 | User\_check | 管理员对认证的用户进行信息审核 | 与数据库的接口SQL |
| M 2-1 | Rent\_add | 房主能够发布自己的待租房源信息。 | 与数据库的接口SQL |
| M 2-2 | Rent\_check | 管理员对上传的房屋进行信息审核 | 与数据库的接口SQL |
| M 2-3 | Rent\_update | 房主能够修改自己的待租房源信息。 | 与数据库的接口SQL |
| M 2-4 | Rent\_delete | 房主能够删除自己的待租房源信息。 | 与数据库的接口SQL |
| M 3-1 | Sale\_add | 房主能够发布自己的待租房源信息。 | 与数据库的接口SQL |
| M 3-2 | Sale\_check | 管理员对上传的房屋进行信息审核 | 与数据库的接口SQL |
| M 3-3 | Sale\_update | 房主能够修改自己的待租房源信息。 | 与数据库的接口SQL |
| M 3-4 | Sale\_delete | 房主能够删除自己的待租房源信息。 | 与数据库的接口SQL |

## 3.1.2 模块测试报告

对用户认证模块，应用基本路径法设计测试用例，进行现场测试。要求画出程序流程图，表示每条基本路径，记录测试数据，评定测试结果。测试活动的记录格式，如表3-2所示。

在测试时，数据库中的用户信息有三个，他们对应的学工号分别是2019229066、096065、2019229044。



|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 编号 | 路径标识 | 输入 | 期望输出 | 输出内容 | 发现问题 | 测试结果 | 测试时间 | 测试人 |
| 1 | r1r2r3r4r5 | 学号：2019229045 | “审核通过” | “审核通过” | 无 | √ | 2020年10月28号 | 刘鑫 |
| 2 | r1r2r6 | 学号：2019229044 | “学工号已被注册” | “学工号已被注册” | 无 | √ | 2020年10月28号 | 刘鑫 |
| 3 | r1r2r6r7r2r3r4r5 | 第一次输入学号：2019229044  第二次输入学号：2019229046 | “学工号已被注册”  “审核通过” | “学工号已被注册”  “审核通过” | 无 | √ | 2020年10月28号 | 刘鑫 |
| 4 | r1r2r3r8 | 学号：asdqdqwedsa； | “已驳回，重新修改” | “已驳回，重新修改” | 无 | √ | 2020年10月28号 | 刘鑫 |
| 5 | r1r2r3r8r9r2r3r4r5 | 第一次输入学号：asdqdqwedsa；  第二次输入学号：2019229047； | “已驳回，重新修改”  “审核通过” | “已驳回，重新修改”  “审核通过” | 无 | √ | 2020年10月28号 | 刘鑫 |

## 3.2 功能测试

## 3.2.1 系统功能需求

功能需求点列表，如表3-3所示。

表3-3 功能需求点列表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 编号 | | 功能名称 | | 使用人 | 功能描述 | 输入内容 | 输出内容 |
| 1 | | 授权登陆 | 房主、用户 | 使用微信登陆 | 无 | 用户微信基本信息及openID |
| 2 | | 认证 | 房主、用户 | 审核是否天大人员 | 姓名、学工号、学工卡照片 | 认证通过、驳回认证 |
| 3 | | 查询所有房源信息 | 用户 | 查询在系统内所有已发布房源 | 点击小程序端首页 | 房源列表 |
| 4 | | 发布房源 | 房主 | 房主发布自己的房源信息等待被租售 | 房源的大小等信息 | 发布成功、发布失败 |
| 5 | | 收藏 | 用户 | 收藏自己心仪的房源 | 点击“收藏” | 收藏成功、收藏失败 |
| 6 | | 联系房主 | 用户 | 联系房主 | 点击“联系他” | 拨号界面 |
| 7 | | 管理员登陆 | 管理员 | 登陆后台审核系统 | 管理员账号密码 | 登陆成功、账号密码错误 |
| 8 | | 用户审核 | 管理员 | 对认证的用户进行审核 | 无 | 操作成功、操作失败 |
| 9 | | 房源信息审核 | 管理员 | 对于房源信息进行审核 | 无 | 操作成功、操作失败 |

## 3.3.验收测试

将系统挂到服务器上后，小组内成员用多种浏览器及不同手机型号同时对该系统进行检测，检查该系统的安全性、稳定性。最终发现，该系统可以支持不到10个用户同时稳定、安全地运行该系统，但超过10个用户，便可以感觉到明显的网络延迟、反应卡顿。我们用的是腾讯云服务器，到期时间为2020年11月31日，即2020年11月31日之后在不续费的情况下该系统无法支持网络使用。

## 3.4测试结论(Test Verdict)

测试日期：2020年10月28

测试地点：平园26斋327宿舍

测试环境：WINDOWS系统下的360浏览器、iPhone11

列出系统的强项：系统功能完善

列出系统的弱项：支持的并发数较少，数据库交互较慢。

列出不符合项的统计结果：无。

测试人员签字：刘鑫