山东大学软件工程课程设计
易修哥校园服务类 APP

姓名：崔玉峰

学号：201600301079

班级：2016 级 5 班

团队成员：赵衍琛、陈怡凡、崔玉峰、张翰林

指导老师：史清华

个人完成部分：Android 前端需求分析、Android 前端设计和 Android 前端测试

目录

一、概述

1.1 开发背景

随着移动设备的快速普及，移动应用时代已经到来，4G 的普及与 5G 的即将推广，手机作为最便携的网络终端设备，使得高校信息化这一话题又有了新的生命。在数据中心不断增强的服务性能之外最显著的特征就是在用户移动端的精彩表现。单纯用 PC 的时代将一去不复返。以手机、平板电脑介质为代表的移动终端应用将为高校信息化带来巨大变革。移动应用不只是在手机上运行软件那么简单，它涉及到高校信息化应用场景的完善、扩展，让数据无所不在，通过广泛的产业联合作为用户提供低成本整体解决方案。

在高校及科研院所，一些公共设施经常需要定期维护和及时维修，不及时的检修和一些突发的事件可能会造成重大财产损害，甚至危害学生和职工的生命。如何防治和及时应对，成为大家普遍关心的热点问题。根据调查，比如在高校校园内，经常有公共设施:房屋、电力设备等的损坏，经常会出现教学用品：电脑、投影仪等的故障，这些情况大大影响了正常的教学工作活动，及时地维修和定期的维护是必要的。如果仅仅等待维修人员的定期检修和维护，一些重大突发的事故隐患可能无法及时发现。作为学校的学生和教工群体遇到这类事情，往往会发现不知如何处理，不知道向那个部门汇报的情况，传统的电话报修系统也不便于管理和维护，无法支持先进的数据分析和智能故障预测。并且，对于目前高校，如果大量增加维修保养人员无疑会增

加额外的支出。

在这样的现实需求中，学生和教工普遍使用的智能手机中开发手机应用程序，构造一个智慧校园应用系统将可以很好地解决这一问题。选择智能手机充当报修设备并不是偶然的，因为：智能手机具有一定的运算能力，基于主流智能手机系统的开发环境和应用的底层接口较为成熟；智能手机自身集成了摄像头、无线网络、各种传感器、2G\3G\4G 网络协议，语音录音通话等等模块，这些都为我们的校园数据展示与分析以及报修智能软件提供了硬件上的保障；另外，智能手机目前普及率高，基本目前人手一部，只需要安装智能移动应用软件及可以实现系统，无需其他的投资。

1.2 目标人群

学生：可以使用自己携带的智能手机，启动报修系统，选择相应的故障类型、具体所在地，并选择拍照提交给报修系统数据中心。
教师：可以使用自己携带的智能手机，启动报修系统，选择相应的故障类型、具体所在地，并可以选择拍照的形式提交给报修系统数据中心。并且可以选择是否对后勤的维修提出建议或评价。

维修工：可以查看主管派下来的维修任务，并进行及时的修理并且在手机端进行完成确认。

管理员：可以通过手机 APP 的管理员端查看空闲维修员及任务完

成情况，可以随时指派维修员去对某故障进行及时修理，并且可以过智能移动端，查看通过后台数据中心分析得出的分析报告，内容包括统计最近的问题类型，可能的问题预测等。

新增功能：班级二维码点名系统，老师可以通过主页的课堂点名按钮生成二维码，学生扫码签到后可以再次点击查看名单查看签到同学名单。

1.3 市场分析

1.3.1 产品及价值定位

①产品定位——主要面向广大学校师生，以报修功能与课堂点名起步积累平台用户，逐渐添加社区论坛等功能，最后可以添加旧物市场、跑腿办事等功能实现平台盈利。

②价值定位——为广大师生提供一个及时解决校园问题，并且在将来成为一个能够互帮互助的的平台，包含代取快递，有偿跑腿，传授知识等。

1.3.2 人群习惯分析

（1）行为习惯

越来越多的年轻人较早的拥有手机等电子设备，并且对手机的依

赖度也越来越高，用时习惯于通过 app 来获取信息、反馈交流信息，并且在线语音、支付功能愈发强大。

（2）消费习惯

随着网络的快速发展，让年轻人尤其是大学生群体早已习惯于网络消费，愿意在 app 中投资消费。本产品目前以及将来所提供的服务皆与师生日常生活的衣食住行有关，需求量大，市场容量大，可行性强。

1.3.3 可能的竞争对手分析

①目前市场上现有的校园跑腿 APP 如：俺来也等

②各高校学生自研内部使用的 APP

二、需求分析

2.1 总体需求概述

【问题说明】

整个解决方案基于以下典型的场景所暴露出的问题展开分析。

场景一：学生 A 发现自习室的门锁坏掉，学生 A 可以使用自己携带的智能手机，启动报修系统，选择相应的故障类型、具体所在地，并选择拍照提交给报修系统数据中心。

场景二:教工 B 发现教室的电脑和投影仪发生故障或操作问题,教工 B 可以使用自己携带的智能手机,启动报修系统,选择相应的故障类型、具体所在地,并可以选择拍照或者语音的形式提交给报修系统数据中心。在线的教室管理人员会选择直接通过手机语音通话帮助教工 B 或者直接维修,并在问题解决后将提交的任务设为以解决。教工 B 会选择是否对后勤的维修提出建议或评价。

场景三:主管 C 是一名设备维护部门的负责人,他的工作之一是负责排查学校可能发生的事故隐患,他手机安装有报修系统的智能移动端,某一天夜里,网络突发故障,造成校园网瘫痪,报修系统会检测到网络无法访问这一情况通过系统通知把这一情况及时通知到主管 C 的手机上。除此之外,主管 C 还可以通过智能移动端,查看通过后台数据中心分析得出的分析报告,内容包括统计最近的问题类型,可能的问题预测等。

综合上述三个场景,团队得到以下分析结论:

【用户期望】

应该尽可能多的支持并兼容符合以下各种配置要求的智能手机:具备摄像头和无线连接功能(WIFI 或蓝牙),可以选配 3G 或 4G 的 SIM 卡。

对于整个报修系统的角色组成和各个角色间的交互:

整个报修系统可以分为:数据存储中心和角色(权限)管理配置系统、智能管理移动端和智能用户移动端。

系统可以方便的增减智能管理移动端设备和智能用户移动端设备。可以采用数据存储中心（使用 SQL Server 等数据库，通过 Web 应用访问）或者智能管理移动端（App）配置，并获得实时报修数据。

智能管理移动端设备可以互相发送指令信息，并根据得到的信息内容，调用相应的功能。可以接收和理配置系统端发出的报修启动、关闭等控制指令，这些指令可以是通过本地 WIFI，或蓝牙等无线连接，也可以选择通过远程网络、电话语音或者短信发送（可选）。

智能用户移动端设备可以提交保修信息，并维护和查看自己的保修信息的状态，并能够通过本地 WIFI，或蓝牙等无线连接，也可以选择通过远程网络、电话语音或者短信发送（可选）。

2.2 功能需求

2.2.1 Android 师生端

1. 学生和老师可通过账号和密码登录本系统，只有登录的用户才能使用本系统的相应功能。

2. 学生用户可通过 Android 客户端并且输入正确的学号，姓名，电话号，密码直接进行注册；

3. 一个学生\老师用户，可以通过本系统填写故障类型，故障发生地，故障相关描述信息，和故障图片并提交给报修系统数据中心

4. 学生\老师可以在任何时刻查看自己提交的报修的处理状态，

（未处理，正在处理，已解决），并且查看提交的报修任务的详细信息（故障类型，故障发生地，故障相关描述信息，和故障图片，故障时间）

5. 对于未处理的报修任务用户可以取消;对于正在处理的的报修问题，可以显示维修人员姓名和电话号码方便与其联系。也可以确认该故障已被解决;对于已解决的报修任务,用户可以填写查看评价,

6. 老师可以发起课堂点名，通过本软件系统生成二维码，学生通过扫描二维码可以完成签到,并且老师可以看到签到成功的学生的名单。

2.2.2 Android 维修职工端

1. 维修工人和维修部门主管通过账户和密码登录入系统,系统可根据不同类型的用户显示不同数据。

2. 维修部门主管可以通过 Android 应用管理端,查看所有提交的报修任务。并且查看的状态，以及详细信息（故障类型，故障发生地，故障相关描述信息，和故障图片，故障时间，故障发起人，故障处理人）

3. 对于未处理的报修任务，主管可以为其分配维修工人；对于正在处理的任务和已完成的报修任务,主管可以查看全部的故障详情。

4. 维修部门主管可以查看维修工人的状态,是否空闲或者正在工作中。

5. 维修部门主管可以查看最近发生的故障的统计情况;包括各类别故障发生次数的统计表；发生故障的次数的时间表等。

6. 维修工人只能查看他被分配的报修任务,和他已经完成的报修任务，以及这些报修任务的详细信息（故障类型，故障发生地，故障相关描述信息，和故障图片，故障时间，故障发起人）

7. 对于维修工人正在处理的任务,维修工人可以确认该故障已被解决；对于自己已完成的任务,维修工人可以查看故障发起人的评价。

2.2.3 WEB 权限管理客户端

1.注册员工账号：输入主管或维修人员的工号、账号、密码、姓名、性别可进行注册。

2.查看权限：可以查询已注册人员的当前的工号、账号、权限、性别、姓 名与状态。

3.授予（更改）权限：需要对新注册的用户进行授权，也可以对已授权的用户更改权限。

2.2.4 Springboot 后端

能够把前端传输来的数据、发起的请求做到及时响应、及时处理,同数据库数据进行交互传输，并以 Json 的形式以安全的形式返回给

前端Android客户端和权限管理客户端。

2.3 设计约束

2.3.1 Android端

1. 物理环境约束：应该基于Android平台开发移动端应用，并且尽可能兼容低版本的Android系统。

2. 用户：应该设计四种类型的用户，学生，老师，维修主管，维修工人。

3. 应该提供用户端和管理端两个Android，用户端可以公开下载，给学生和老师进行使用；管理端由内部发放给维修人员和维修主管进行使用。

2.3.2 WEB端

1. 物理环境约束：基于PC端window、linux多系统，兼容ie8以及以上版本浏览器。

2. 用户：权限管理者。

3. 权限管理系统只对权限管理人员进行开放，通过浏览器访问。

2.4 过程约束

2.4.1 Android 端

1. 一个报修任务在被提交后还没有工人处理时状态为未处理;一个报修任务已经分配工人处理的状态为正在处理;一个报修任务在被维修工人或提交者标记为已完成时状态为已完成。

2. 对于状态为已完成的报修任务,提出者可以对完成情况,维修工人做出评价。

2.4.2 WEB 端

权限管理人员注册一个新的员工账号后,或原有账号更改权限时,需要使用授权功能对此账号进行授权。

2.5 非功能需求(质量需求)

2.5.1 安全性

Android 端:

通过严格的账号密码权限体系,以及双端分离的方式,不同的用户角色应该展示不同的界面,访问不同的数据,提供不同的功能。Android 端只提供学生注册功能,对于教师,维修工人的注册需要通

过账户管理平台进行访问。登录注册的逻辑实现，全部交给服务端处理，Android 前端，不会直接访问数据库，增加安全性。

WEB 端：

管理员访问登录界面，只有在账号密码正确的情况下，可以得到权限管理界面的访问权限，进行注册，查看，授权操作，在执行这些操作时会同时检查管理员的登录状态，保证系统的安全。

服务器后端及数据库：

1、 Springboot 自带 Security 框架和 token、sessionid

2、 数据库密码哈希加盐保存不用明文保存

3、 定时备份

4、 不同角色权限严格管理

5、 数据库自身口令加长并且定期更换

6、 采用方法级别的事务管理

2.5.2 可修改性

Android 端：

前后端分离开发，通过 Http 接口进行交互，整体系统耦合低。

WEB 端：

Html+js+css，最大的保证了 web 端的可修改性。

服务器后端：

采用Maven管理Springboot框架并且采取MVC架构，方便日后增加新功能时可以更加方便的导入所需第三方依赖,采用面向切面编程的形式,在以后增加功能时可以以模块化的形式整体加入而不影响其他代码，并且服务器系统整体支持热部署。

2.5.3 可维护性

1. 前后端均采用MVC架构进行设计，提高整体代码编写的可维护性，低耦合高内聚，

2. 对于项目使用的第三方库集中管理，使用通用的、经过广泛测试稳定好用，并且日后能够有持续维护更新的第三方库，方便日后的版本更新。

3. 前后端分离开发，减少整体系统的耦合度提高可维护性。

2.5.4 易使用性

Android端:

1. 整体Android客户端界面美观友好，交互流畅，并且采用标准化的Android移动端UI设计规范，符合移动端用户的操作习惯。

2. 对于一些表单的填写，能够提供可选项，尽量不需要用户手动输入。比如故障类型的选择，故障地址选择。

3. 文字说明配合图标使用方便用户找到相应功能。

WEB 端:

1. web 端界面设计简洁明了，功能模块划分明确。

2. 注册授权表单有明确提示，输入有误会具体提醒。

3. 操作简单，用户只需要点击和输入必要信息即可完成所有操作。

服务器后端:

代码编写好后直接整体用 maven 打包成 jar 包，然后对于新配置的服务器，只需要配置好 java 环境和 Mysql 数据库，然后在服务器中直接运行打包好的 jar 包文件即可。或者可以打包成 war 包，然后执行配置文件自动下载配置 tomcat 和 Mysql 数据库，然后把 war 包放置在 tomcat 目录下的 app 目录中即可。

2.5.5 高效性

Android 端:

为了提高使用流畅性以及交互性，应该采用 Android 原生开发的模式，而非 B/S 结构，有效提高了性能。

因为整体程序的界面较少，因此采用单 Acitivity+多 Fragment 的整体架构进行开发，可以使界面切换更加流畅。

WEB 端:

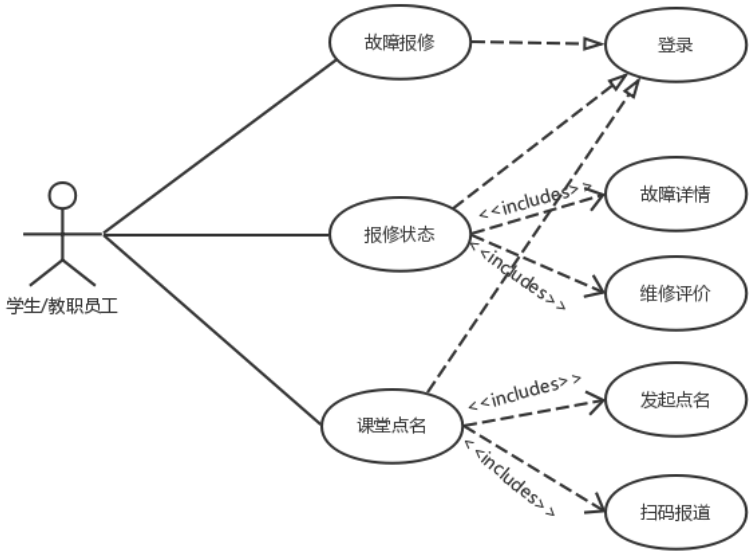
界面设计简洁，没有复杂 js 以及图片，网页加载速度较快。注册授权的数据较小，传输较快。权限查看在大量数据的情况下也能较快加载。总体性能较高。

服务器后端:

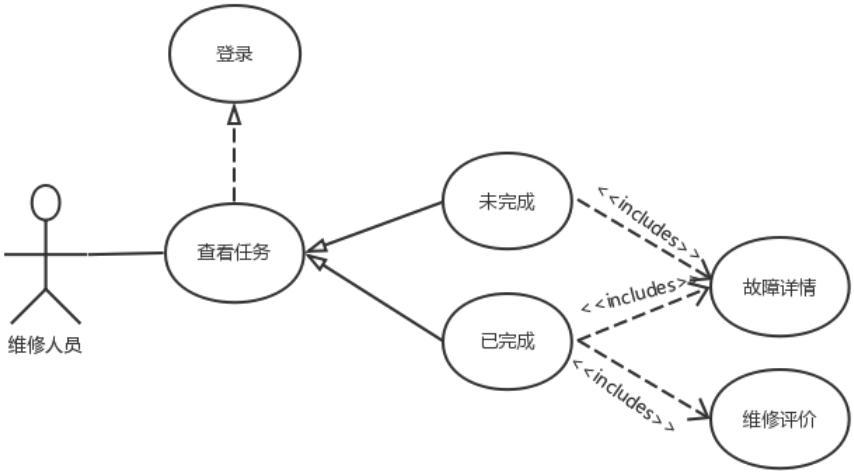
因为数据库在服务器本地，所以数据查询响应更快，并且采取 Mybaits 和 hibernate 处理数据和图片，加速和服务器的数据交换速度，能够做到对前端请求的快速响应，并且后端的高效性也取决于用户购买的云端服务器的配置，当云服务器的带宽和硬件配置越高时，整体后端系统性能也会大幅提升。

2.6 UML 建模

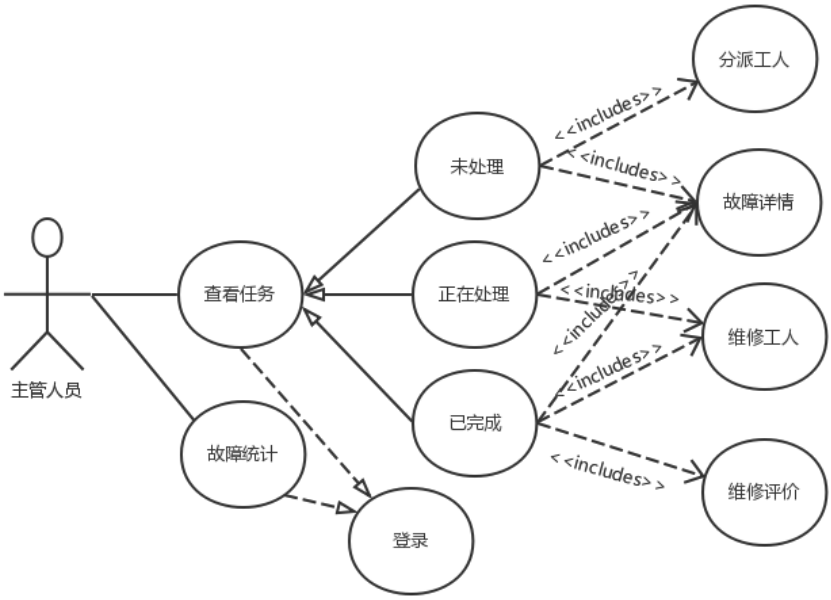
2.6.1 用例图



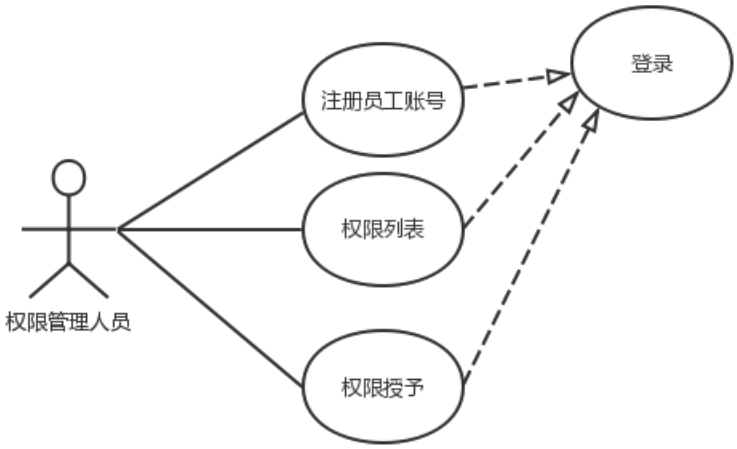
教师/学生



维修员工

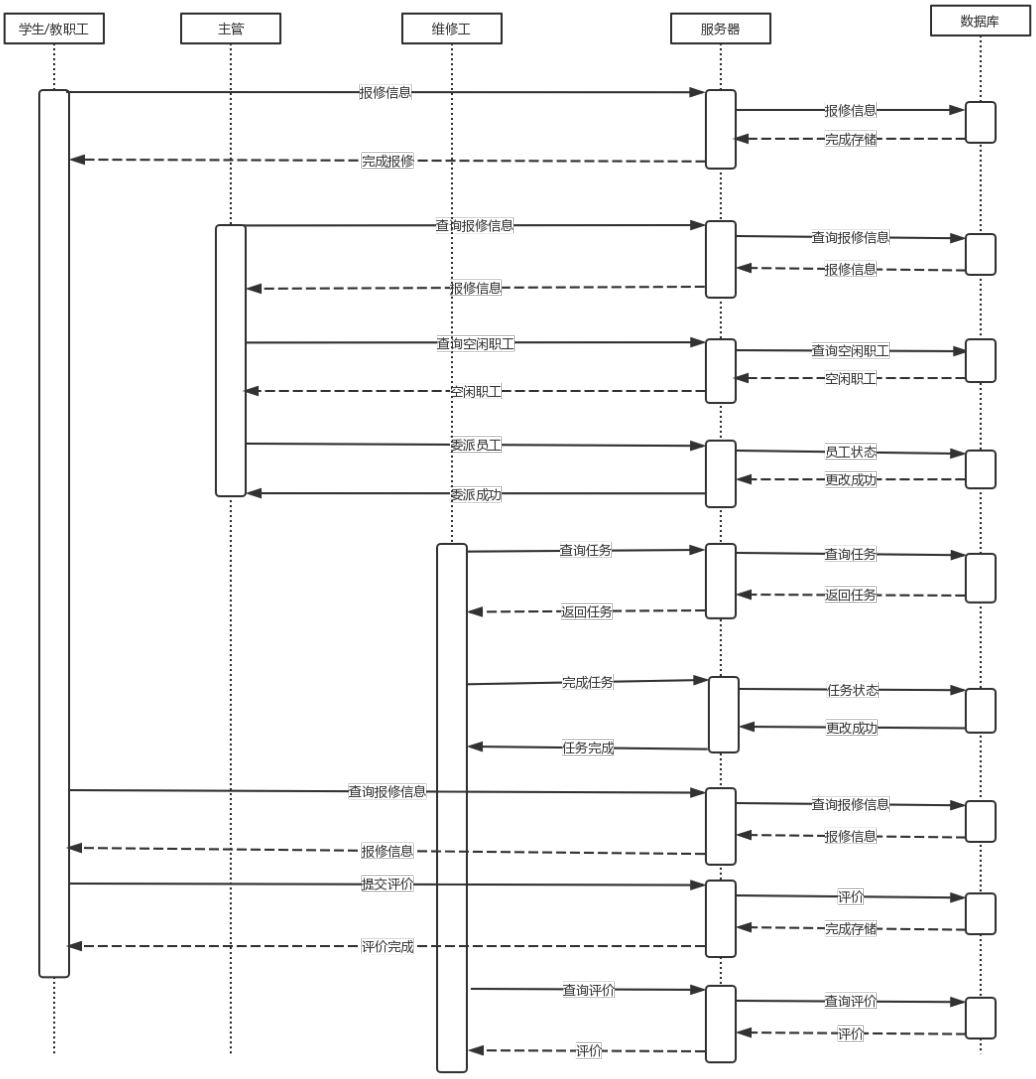

分派工人
未处理
<<includes>>
故障详情
查看任务
正在处理
维修工人
已完成
维修评价
故障统计
登录
主管人员

维修主管

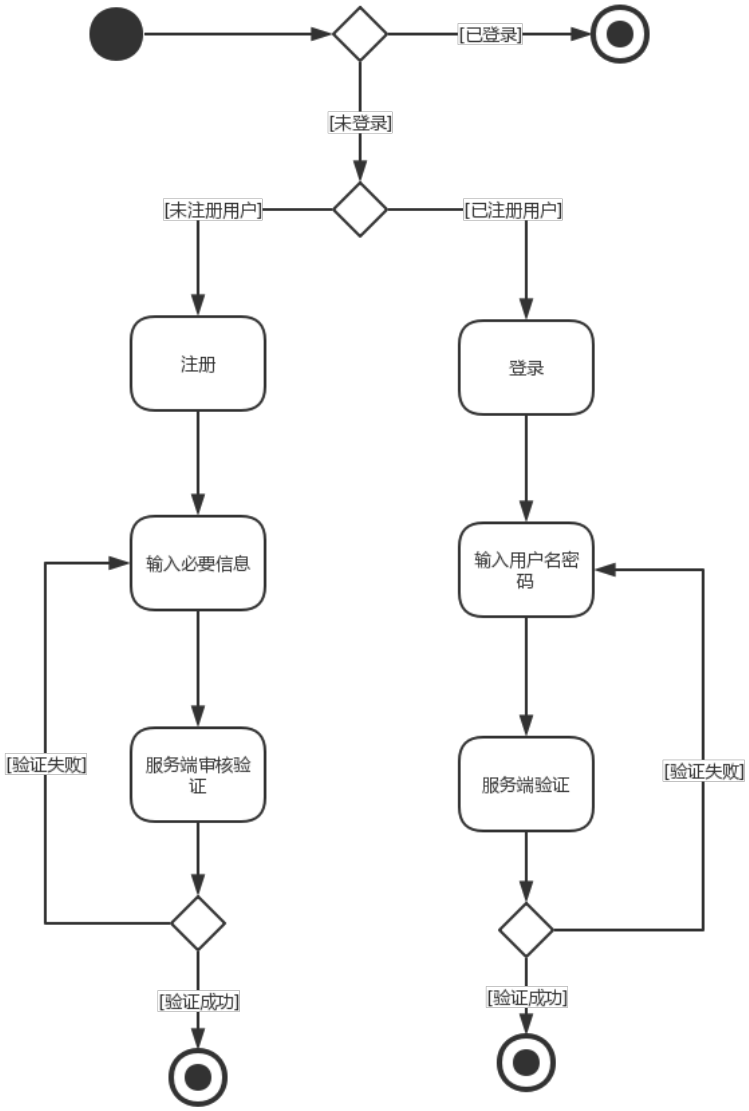

登录
注册员工账号
权限列表
权限授予
权限管理人员

系统管理员

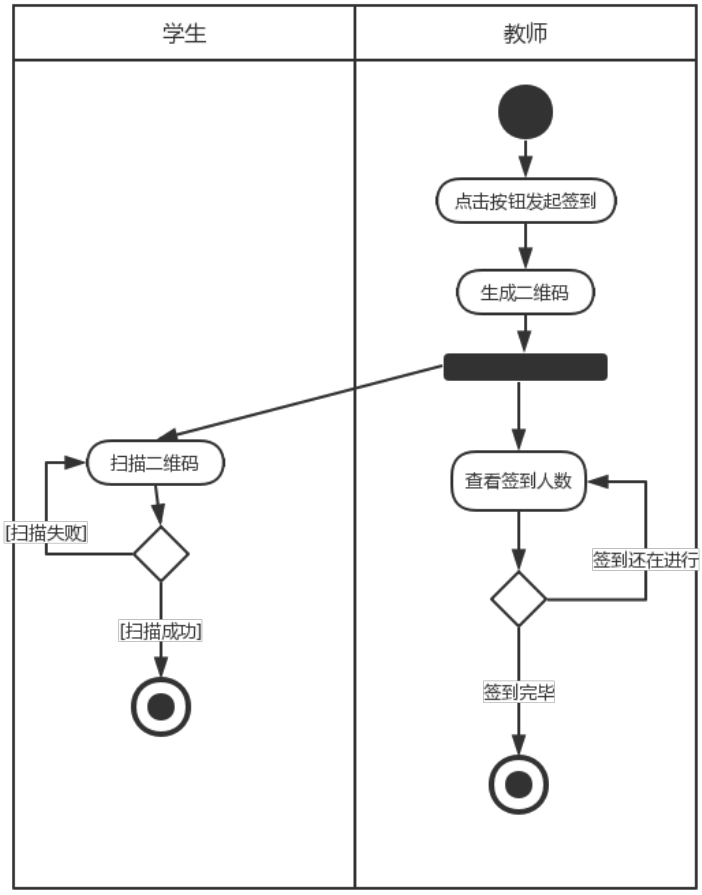
2.6.2 时序图


学生/教职工
主管
维修工
服务器
数据库
报修信息
报修信息
完成存储
完成报修
查询报修信息
查询报修信息
报修信息
报修信息
查询空闲职工
查询空闲职工
空闲职工
空闲职工
委派员工
员工状态
委派成功
更改成功
查询任务
查询任务
返回任务
返回任务
完成任务
任务状态
任务完成
更改成功
查询报修信息
查询报修信息
报修信息
报修信息
提交评价
评价
完成存储
评价完成
查询评价
查询评价
评价
评价

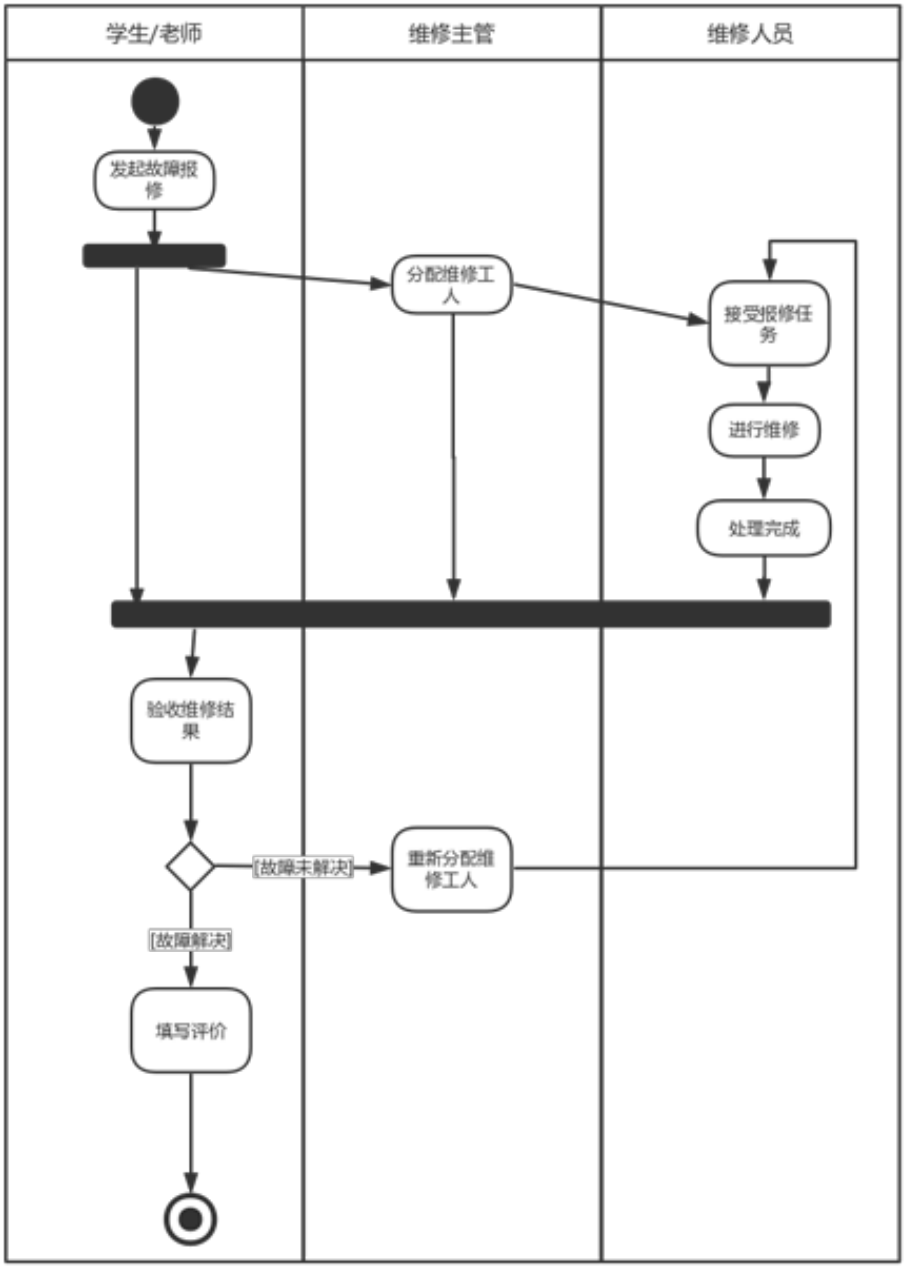
2.6.3 活动图



登录/注册活动图



签到活动图

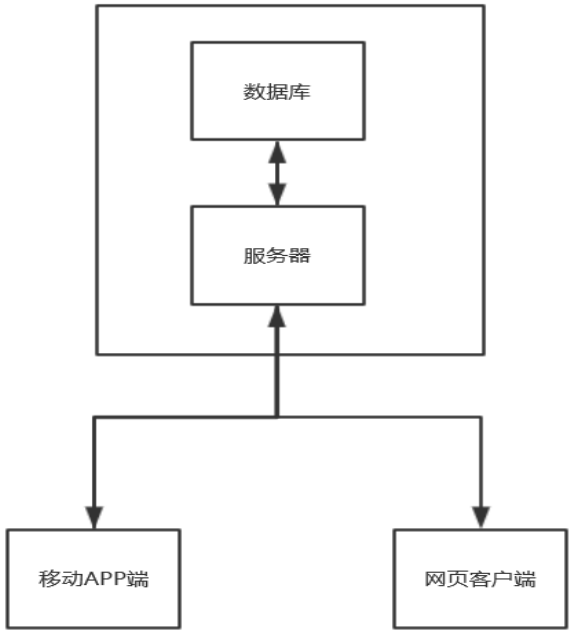


故障报修活动图

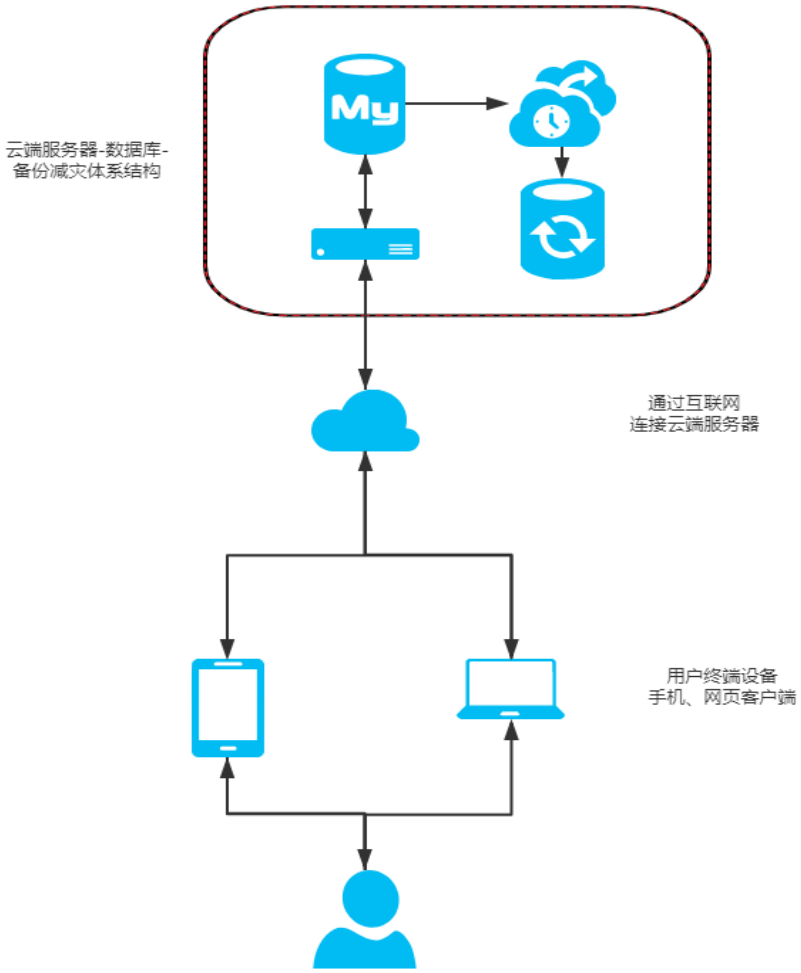
三、系统设计

3.1 系统架构

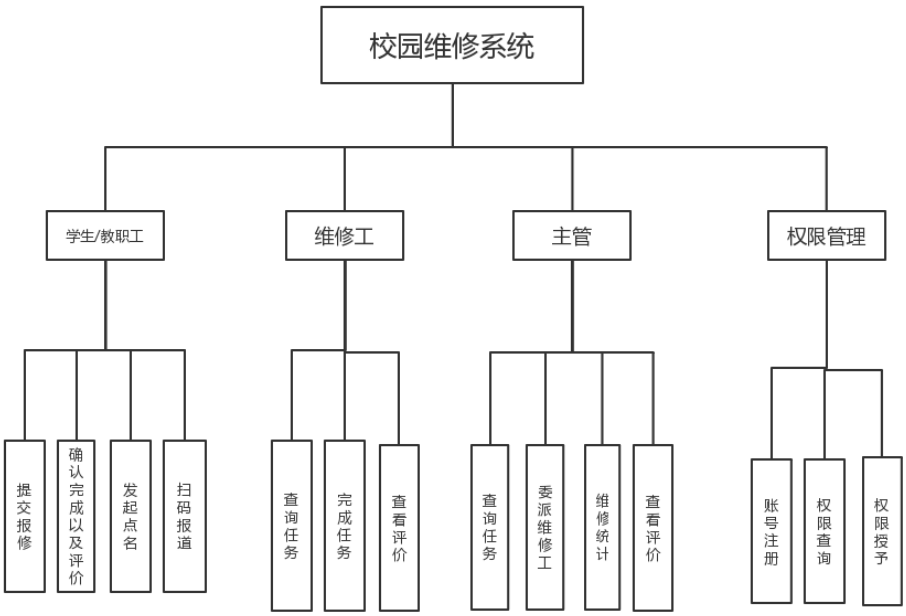
整体系统采用C/S架构，用户通过手机、电脑浏览器等终端途径来对服务器提交相应请求，然后服务器与服务器本地数据库进行数据交换，处理响应。



C/S 架构


云端服务器-数据库-
备份减灾体系结构
通过互联网
连接云端服务器
用户终端设备
手机、网页客户端

3.2 系统模块设计


校园维修系统
学生/教职工
维修工
主管
权限管理
提交报修
确认完成以及评价
发起点名
扫码报道
查询任务
完成任务
查看评价
查询任务
委派维修工
维修统计
查看评价
账号注册
权限查询
权限授予

3.3 前后端整体架构

3.3.1 硬件环境

操作系统：Windows、Linux 均可

手机系统：Android5.0+版本以及目前主流屏幕分辨率均可适配

3.3.2 软件环境

前端：Android 手机客户端、管理端

WEB 端：目前主流浏览器包括 IE 均可使用

服务器：腾讯云+tomcat 容器

后台：Springboot+MyBaits 主流后端框架，系统易扩展可移植性高

数据库：MySQL8.0+

3.4 系统初始建模

3.4.1 前端代码架构

概述：

Android 原生 APP 开发通过 java 编写，其本身就有一套相对完

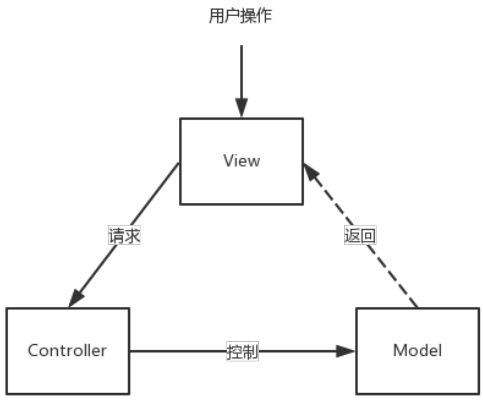
整的开发框架，以每个 Activity 界面作为中心进行编写，通过 XML 语言来编写界面样式。通过 Gradle 项目管理语言打包成 APK。这些是由 Google 官方提供给开发者的，所以在此的基础上进行项目整体架构上的设计：

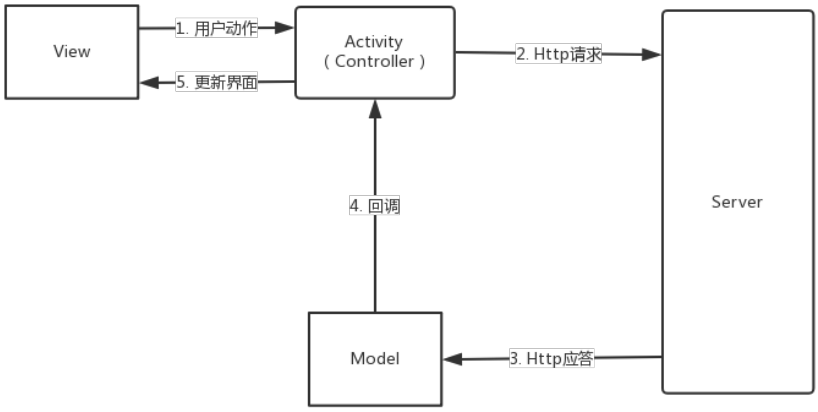
Android 整体设计架构为 MVC 架构，以每个界面为中心的设计方式：

Model 模型层负责数据的获取（JSON，Java Bean

View 视图层负责用户界面编写（XML，JAVA）

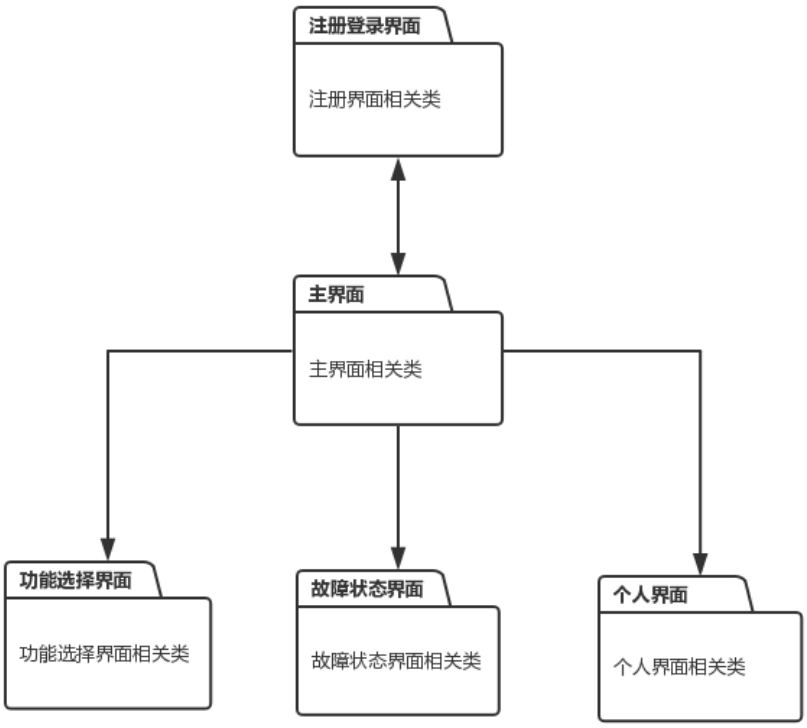
Control 控制层负责将视图层和模型层的交互。(Activity，Fragment)





用户通过与界面进行交互操作，交互信息会传递到控制层 Activity/Fragment 上，控制层根据交互内容通过 Http 请求访问服务器，服务器应答返回为 JSON 格式，通过 Model 层进行转换处理，将结果放回给控制层，控制层根据返回结果更新界面。

包图：



3.4.2 后端代码架构

概述：

开发工具：Intellij IDEA

后台框架：Spring Boot+Mybaits+MySQL+Maven+Thymeleaf

主要架构：MVC

其中，后台框架的部分介绍：

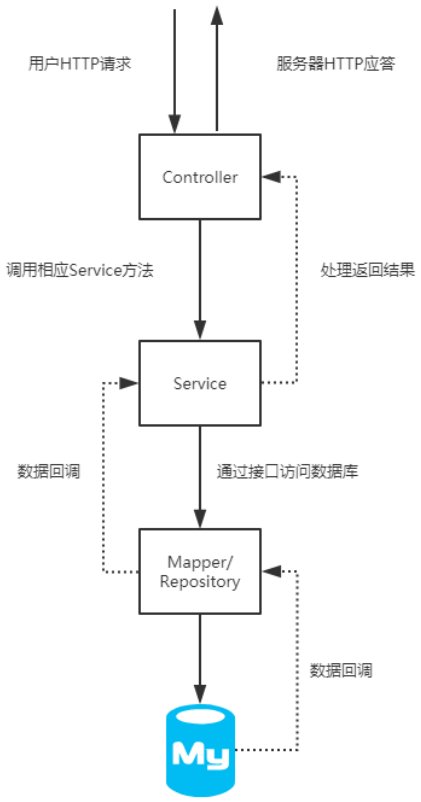
Spring Boot：全新后台框架快速应用开发领域的领导者。

Mybaits：优秀的持久层框架，它支持定制化 SQL、存储过程以及高级映射。便于 SQL 的统一管理和优化。将业务逻辑和数据访问逻辑分离，效率更高，更易维护。

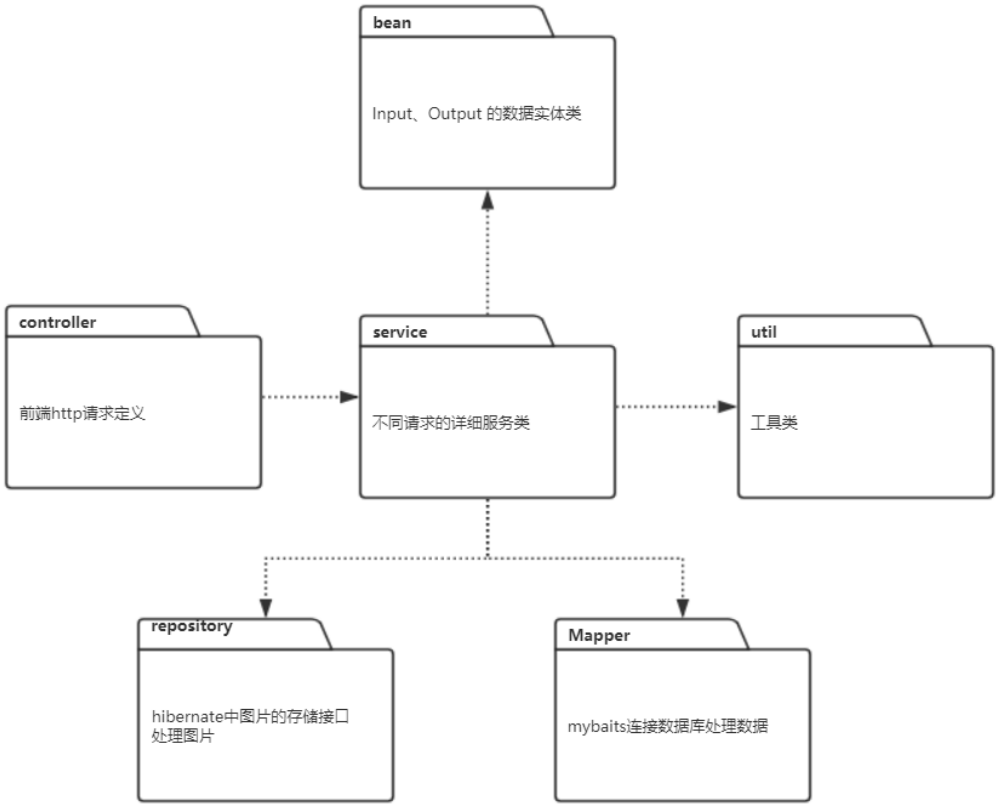
MySQL：最流行的关系型数据库管理系统之一。

Maven：项目对象模型(POM)，可以通过一小段描述信息来管理项目的构建，报告和文档的项目管理工具软件。

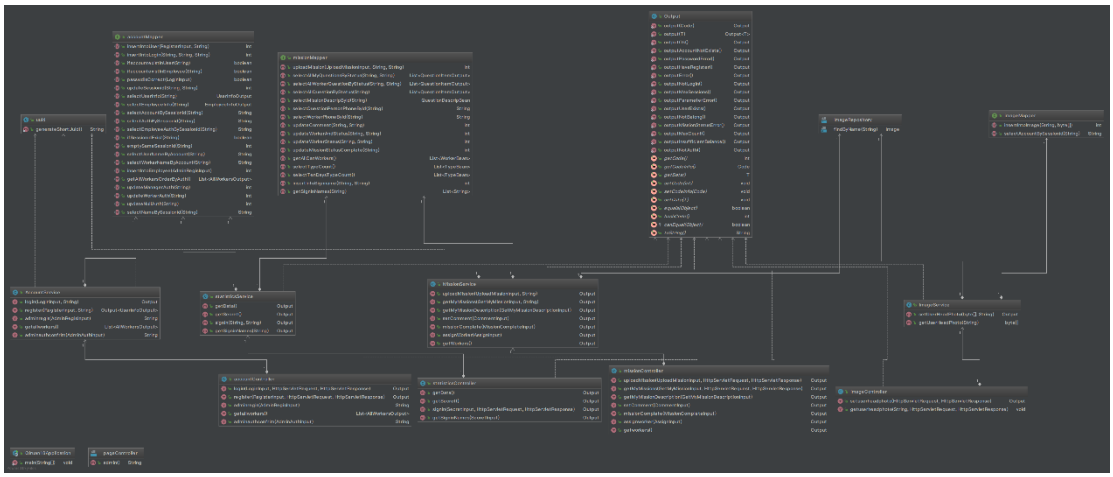
Thymeleaf：Thymeleaf 是 Web 和独立环境的现代服务器端 Java 模板引擎，能够处理 HTML，XML，JavaScript，CSS 甚至纯文本。主要负责返回给 WEB 端 HTML 页面。


用户HTTP请求
服务器HTTP应答
Controller
调用相应Service方法
处理返回结果
Service
数据回调
通过接口访问数据库
Mapper/
Repository
数据回调
My

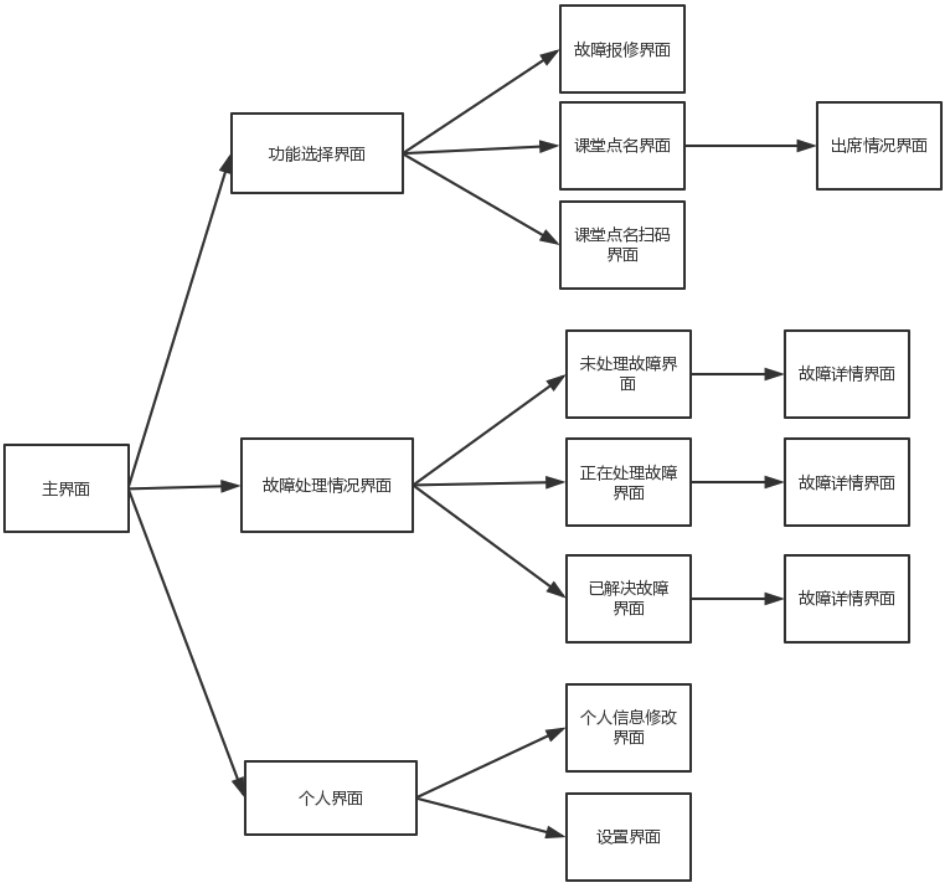
包图：


bean
Input、Output 的数据实体类
controller
前端http请求定义
service
不同请求的详细服务类
util
工具类
repository
hibernate中图片的存储接口
处理图片
Mapper
mybaits连接数据库处理数据

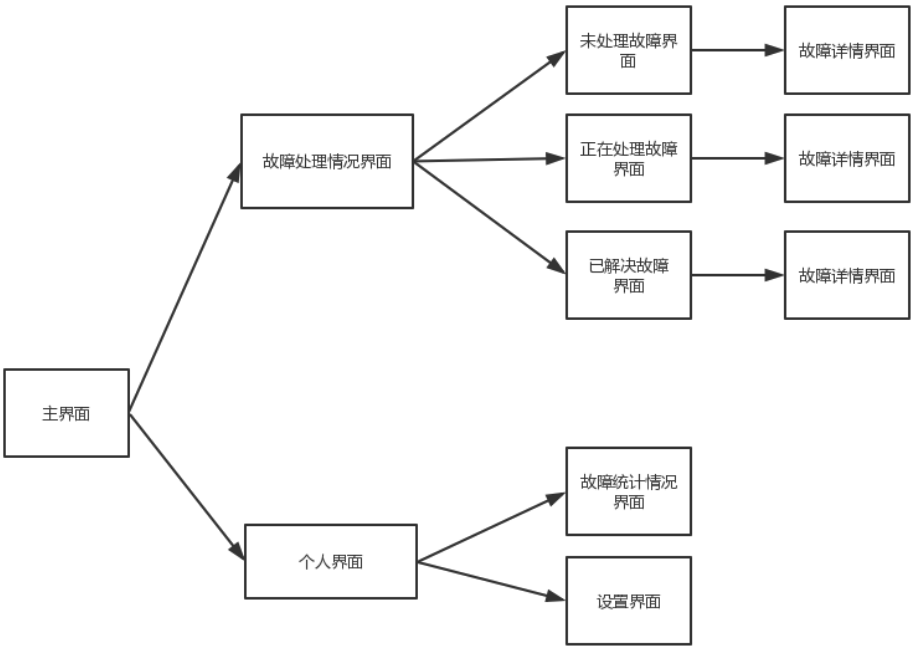
类图：



3.5 界面设计

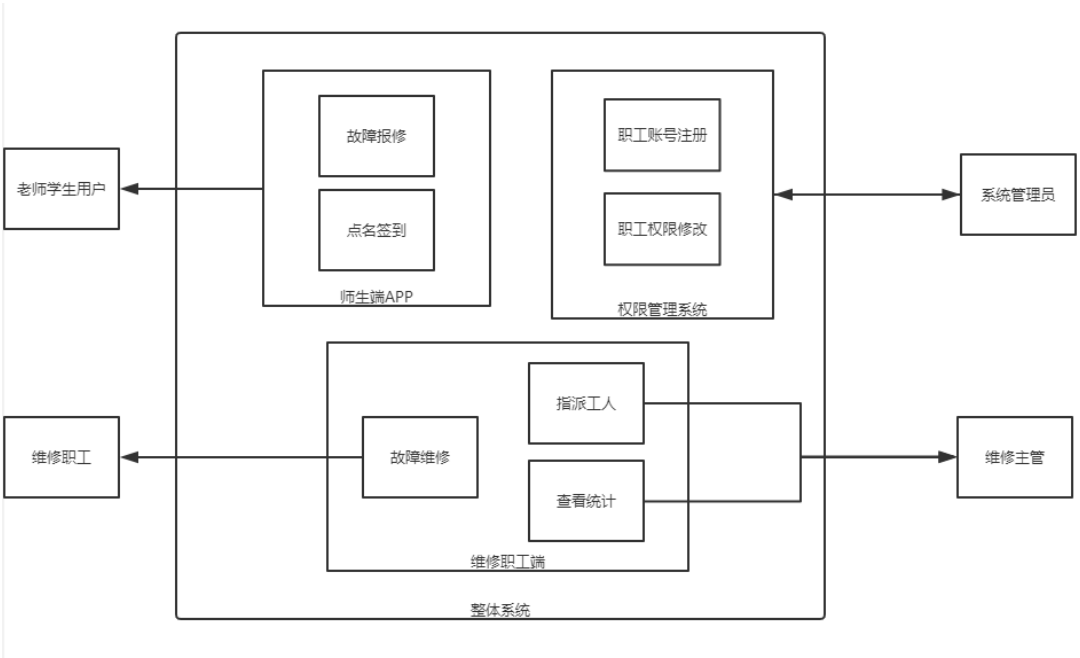


师生端界面设计



管理端界面设计

3.6 系统边界设计



3.7 工作计划

3.7.1 人员分工

WEB 端代码：张翰林

Android 端代码：崔玉峰

后端代码：赵衍琛

数据库设计、系统测试：陈怡凡

3.7.2 前后端 API 接口文档规定

例：

前端类别：Android 端　　请求方法：POST

用途：指派工人　　url：/api/mission/assign

请求体包含数据：　　返回值：

String questionId;　　{

String workerid;　　"code": 0,

"codeInfo": "OK",

"data": ""

}

3.7.3 总体工作流程

开发阶段	主要任务
第一阶段	主要开发框架确定、版本控制措施建立、API 文档确定、简单的接口测试。
第二阶段	迭代式开发，先开发出一个具备绝大多数功能的框架，然后前后端联调测试接口，全部通过后进一步完善具体实现。
第三阶段	服务器配置部署，将本地服务器转移至云端服务器进行最终系统测试，模拟真实使用环境测试。

3.8 数据库设计

3.8.1 数据库系统架构

数据库：MySQL8.0+（注：数据库位置位于服务器本地，非额外购买数据库）

数据库管理系统：Navicat For MySQL

3.8.2 数据库设计规划

3.8.2.1 任务陈述

根据系统需求，系统的主要任务是师生与主管、维修工人之间的维修任务分配完成、上课点名问题，数据库信息主要涉及到权限管理、任务多状态变换、人员信息存储、点名信息存，所以需要设计用户表、职工表、任务表、登录状态表、签到表，还有一个图片表，来存放用户头像和任务图片等图。

3.8.2.2 任务目标

目标级别	子目标
老师、学生	个人信息：查看个人信息（用户名、头像）、修改密码、修改个人信息、添加地址、意见反馈； 扫码签到：老师可以发起签到、查看签到名单、学生扫描二维码签到； 故障反馈提交：可以对校园中的故障添加标题、详情、地址、照片； 故障处理情况查看：对已提交的故障查看处理进度，并且能够进行处理情况的及时反馈，确认维修任务完成。
维修部门主管	个人信息：查看个人信息（用户名、头像）、修改个人信息、修改密码； 查看工人情况：查看工人当前工作状态； 查看意见反馈：查看故障报修人的意见反馈汇总； 查看统计情况：查看近十天内各种故障发生的数量、查看全部时间内各故障发生数量。 分配维修任务：为空闲的工人分配维修任务，查看报修的全部维修任务的进度。
维修职工	个人信息：查看个人信息（用户名、头像）、修改个

	人信息、修改密码； 查看维修任务：查看分配给自己的维修任务、 统一账号注册：能够为主管、职工注册账号。
系统管理员	权限查看列表：能够查看各员工的权限（主管还是普通员工）、为每位员工变更权限。

3.8.2.3 数据库系统需求

该数据库系统需要支持较强的数据处理功能，理论上应该能够容纳多所高校上万人的信息资料（包括头像）、能够容纳平均每人 50 条左右的报修信息历史记录（包括故障图片），并且在数据查询方面应该具有较快的响应速度，能够处理多方面的数据请求。权限设置清晰明了，能够有效防止越权操作，系统能够处理各种异常，具有较强的健壮性。

3.8.2.3.1 网络和共享需求

①保证带宽必须能支持至少 500 名用户同一时刻发起访问请求。

②保证同一账号同一时刻只能在一台设备上登录，不能重复登录。

③本系统通过网络和云端服务器连接，需要保证时刻有数据连接。

3.8.2.3.2 增删改查性能

①报修记录查询时间少于 1 秒，高峰期少于 3 秒。

②更新/新建报修信息时间少于 1 秒，高峰期少于 3 秒。

③扫码点名二维码扫码后的响应时间少于 2 秒，高峰期少于 4 秒

3.8.2.3.3 安全性

①数据库要有足够长度的口令进行保护，并且该数据库口令必须是无任何含义的随机字符组成，并且定期更换，避免暴力破解。

②对于用户的密码进行 hash 加盐存储保存密文，避免数据库不小心数据泄露导致的用户账户安全问题。

③对于用户权限一定要严格管理，用户只能操作他们权限下的功能，不能越权操作。

④设置数据库连接池最大连接数量，避免恶意连接重复请求占用查询带宽。

3.8.2.3.4 备份和恢复

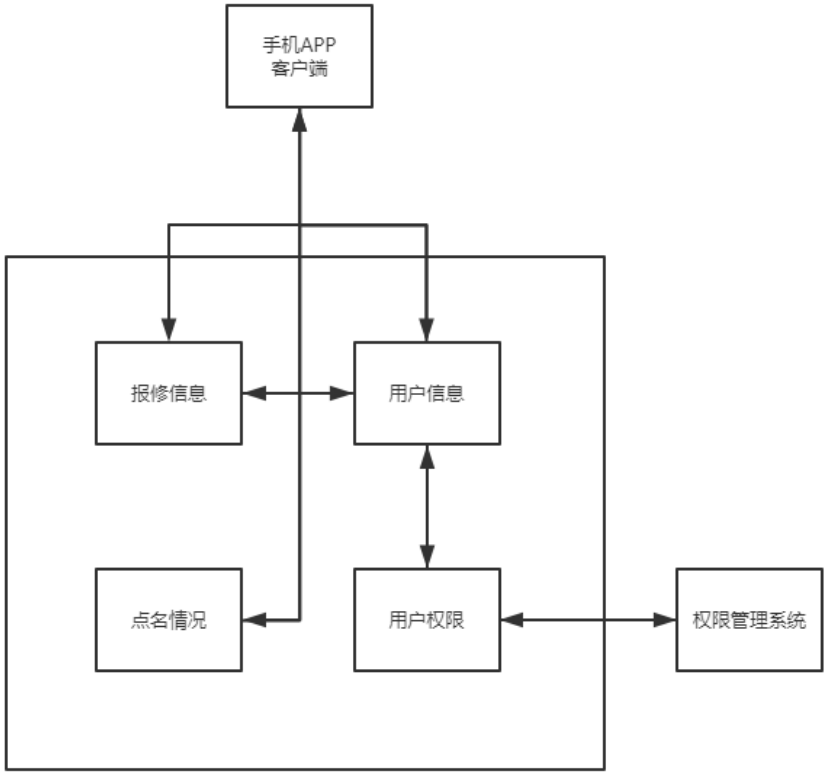
每天 0 点时对数据库信息进行及时的自动增量备份。

3.8.2.3.5 法律问题

对每位用户的信息进行妥善保管，密码密文存储，保证用户信息、照片、实名认证等信息不泄露、不外传，遵守相关法律法规。

3.8.3 数据库系统设计

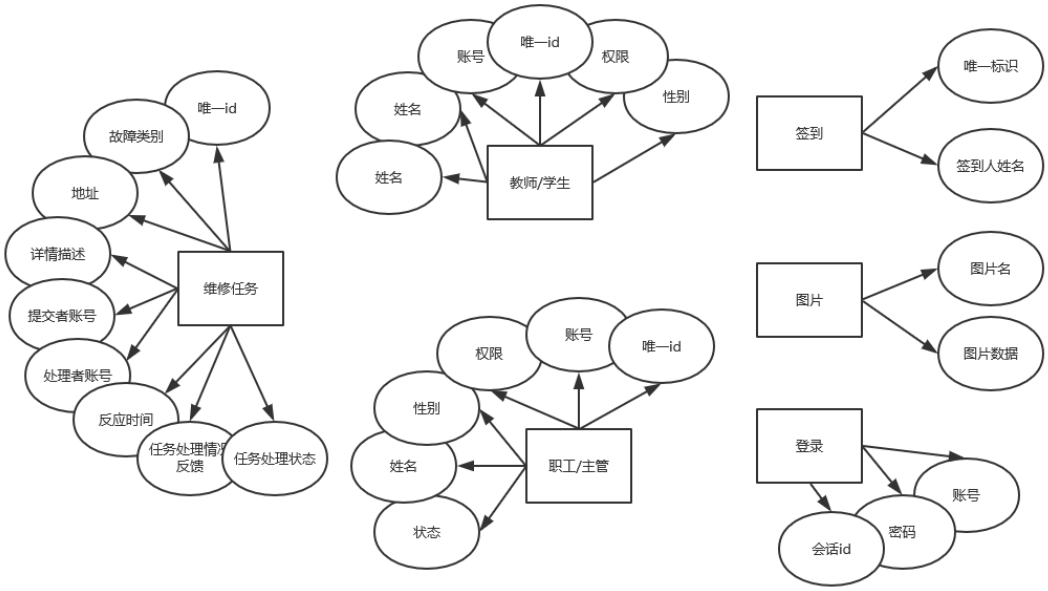
3.8.3.1 数据库系统边界



数据库系统边界图

3.8.3.2 数据库逻辑设计

3.8.3.2.1 ER 图



3.8.3.2.2 数据字典

①图片表（image）

字段	类型	是否主键	是否空值	说明
name	Varchar（50）	是	否	图片名
data	longblob	否	否	图片数据

②登录表（login）

字段	类型	是否主键	是否空值	说明
account	char（18）	是	否	用户账号
password	char（40）	否	否	图片数据
sessionid	char	否	否	记录登录

	（40）		状态

③师生用户信息表（user）

字段	类型	是否主键	是否空值	说明
id	char（10）	是	否	用户唯一标识 id
account	char（18）	否	否	用户账号
auth	char（1）	否	否	用户权限等级
sex	char（1）	否	否	用户性别
name	char（5）	否	否	用户真实姓名
schoolnumber	char（15）	否	否	用户学号

④主管职工表（employee）

字段	类型	是否主键	是否空值	说明
id	char（10）	是	否	职工唯一标识 id
account	char（40）	否	否	职工账号
auth	char（11）	否	否	职工权限等级
sex	char（1）	否	否	职工性别
name	char（5）	否	否	职工真实姓名
status	char（5）	否	否	当前职工状态

⑤报修信息（mission）

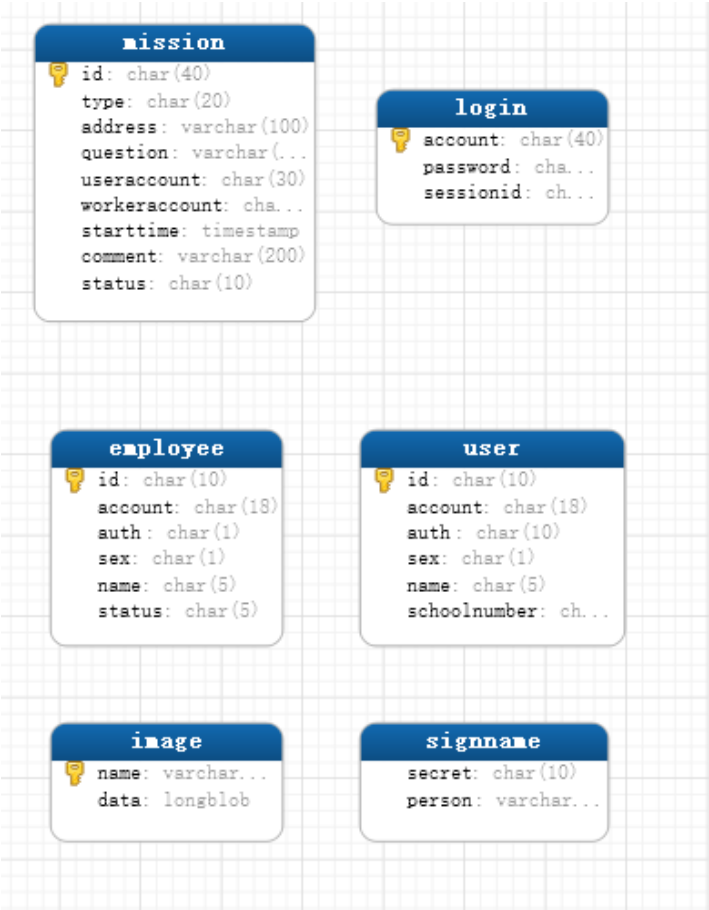
字段	类型	是否主键	是否空值	说明
id	char（40）	是	否	维修任务唯一标识 id
type	char（20）	否	否	维修任务故障类别
addres	varchar	否	否	维修任务的地

s	（100）			点
question	varchar（255）	否	否	维修任务详情
useraccount	char（30）	否	否	提交维修任务者的账号
workaccount	char（30）	否	否	负责维修的维修人员的账号
starttime	timestamp	否	否	维修任务开始时间
comment	varchar（200）	否	是	维修任务反馈
status	char（10）	否	否	当前维修状态

⑥点名表（signname）

字段	类型	是否主键	是否空值	说明
secret	char（10）	否	否	唯一标识一次签到任务的标识ID
person	varchar（255）	否	否	签到人的姓名

3.8.3.2.3 数据库模型图



3.8.3.3 数据库物理设计

3.8.3.3.1 索引

通过为各表的 id 和能够唯一标识该条数据的属性建立索引。

3.8.3.3.2 事务

事务作为单个逻辑工作单元执行的一系列操作，要么全部执行，要么都不执行。事务设计可以确保除非事务性单元内的所有操作都成功完成，或者报错都不执行，确保了数据的一致性。

在本校园故障报修系统中，对用户、职工、维修任务及各维修任务状态、人员权限设置等操作，几乎很多都运用到事务设计。

由于该系统有独立Spring Boot后台，可以利用Spring Boot中的@Transactional注解来进行方法级别上的事务管理，它保证了用户的每一次操作都是可靠的，即便出现了异常的访问情况，也能够通过统一的异常管理进行处理，也不至于破坏后台数据的完整性。

此处举例说明：

对于用户登录来说，首先要查询账号是否存在，如果存在，再查询密码是否正确，如果密码正确，再查询到用户的唯一ID进行return返回。对于这一系列的数据库查询操作，由于需要连续判断，任何一步的错误很可能导致用户登录失败或错登，严重影响使用。所以统一使用@Transactional事务管理。

3.8.3.3.3 安全机制

(1)数据传输安全：

①登录密码密码hash加盐对称加密：

使用通用唯一识别码（UUID）生成盐值，在用户注册时随信息一起加入数据库。并且在每次登陆时验证成功后重新生成一个新的盐值，不重复使用一个盐值。在认证的时候，根据传入的用户名取出以密文形式存储的密码及对应的 salt 盐值，对明文再次进行计算，结果与存储的 hash 结果进行比对，相同即通过验证。

②Cookie 与 SessionId 验证避免多次重复登录引起的错误，还可保障安全性。

③Spring Boot-Security 有效后台防护防止异常请求。

(2) 数据库系统安全:

①数据库密码口令保护

②用户密码 HASH 加盐存储

③每个用户只能查看自己权限所对应的窗口数据

④每晚 24 点进行数据库备份

四、测试文档

4.1 测试目的

易修哥的这一测试计划文档有助于实现以下目标：

1、列出推荐的测试需求

2、推荐可采用的测试策略，并对这些策略加以说明

3、确定所需的资源，并对测试的工作量进行估计

4、列出测试项目的可交付元素

4.2 测试需求

1、单元测试、集成测试

Android 前端界面显示情况、交互逻辑、输入域测试

WEB 权限管理客户端界面展示情况、交互逻辑、输入域测试

Springboot 后端同数据库数据交换存储测试

2、系统测试

Springboot 后端与手机端、WEB 端 API 接口联调测试

3、黑盒测试

模拟软件各角色用户与权限系统管理员对系统进行整体测试

4.3 测试策略

主要分为 Android 端、WEB 端、Springboot 后端三端分别编写测

试文档然后进行整合。因为采用了前后端分离的开发模式，首先应该定好前后端接口然后统一根据定好接口进行开发和测试：因此对于单元测试阶段，主要测试每个界面的运行情况，交互情况，并且所需要的数据是根据接口进行模拟的数据。

1、单元测试

Android 端：

①、登录/注册功能测试：

登录输入要求：

(1) 登录用户名为电话号码只能由数字组成，且不超过 11 位

(2) 登录密码在输入小于 18 位，数字字母组成

注册输入要求：

(1) 输入电话号码 真实姓名 性别和学号进行注册

(2) 电话号码只能由数字组成，且为 11 位

(3) 真实姓名为汉字组成，长度不超过 5

(4) 密码大于 10 位数字小于 18 位，字母组成

(5) 性别只能选择男/女

(6) 学号由数字组成，且为 12 位

根据上述要求对输入进行黑盒测试，给出测试用例：

登录测试用例：

测试数据	测试范围	期望结果	实际结果
手机号：18340018831	有效	有效	有效
手机号：A1234567890	包含非数字	无效	无效提示包含非法字符
手机号：12345678900	输入长度不足 11 位	无效	无效提示输入长度不够
密码：1fsda2f4as5	有效	有效	有效
密码：发生地方aaad	包含中文	无效	无效提示包含非法字符
密码：Fdsaxcvzv457a89fd5	输入长度超过 18 位	无效	无效提示密码长度过长。
不输入密码/手机号	必要元素缺失	无效	无效提示请输入密码/手机号

注册测试用例：（在登录测试用例的基础上增加这个测试用例）

测试数据	测试范围	期望结果	实际结果
真实名字：崔玉峰	有效	有效	有效
真实名字：崔玉峰 a	包含非法字符	无效	无效，提示包含非法字符
学号：1234567890	输入长度不足 12 位	无效	无效提示输入长度不够
学号：201600301079	有效	有效	有效
不输入性别/学号/姓名	必要元素缺失	无效	无效提示请输入性别/学号/姓名

②、提交故障报修测试：

提交故障数据要求：

① 输入故障地点，故障类型， 故障详情说明，故障图片

② 故障地点，输入长度小于 25 个字符

③ 故障类型，提供四个可选项，教室故障，物业，水电故障，其他故障。只能从四个选项中选择

④ 故障详情说明,输入长度小于 100 个字符说明文字

⑤ 故障图片，png 格式。

根据数据要求设计测试用例：

测试数据	测试范围	期望结果	实际结果
故障地点： 山东大学软件学院二号楼	**有效**	**有效**	**有效**
故障地点： 山东大学软件学院二号楼发士大夫打发山东高低撒发生大法师的	**长度大于 25**	**无效**	**无效，提示故障地点长度过长**
故障详情说明： 教室的投影仪出现了损坏	**有效**	**有效**	**有效**
故障详情说明： 教室的投影仪出现了损坏范德萨发达发热网球vhxcjvbzcxnbvmnxzfhjskdauirwueqfdsafsdafweqruiweqyfajkdxvbcxnvhzjhsduiarfyhesjkfcbxvj	**输入长度大于 100**	**无效**	**无效，提示故障详情输入过长。**

kehufsdahvczxhfui
weuyrqweiodzxbvnc
bxvcxzi

未填写/故障地点/故障详情说明/故障类型/	缺少必要元素	无效	无效，提示请完善信息
故障图片：格式为 jpg 不为 png	输入格式	无效	无效，提示图片加载失败。

③、获取故障信息列表数据测试：

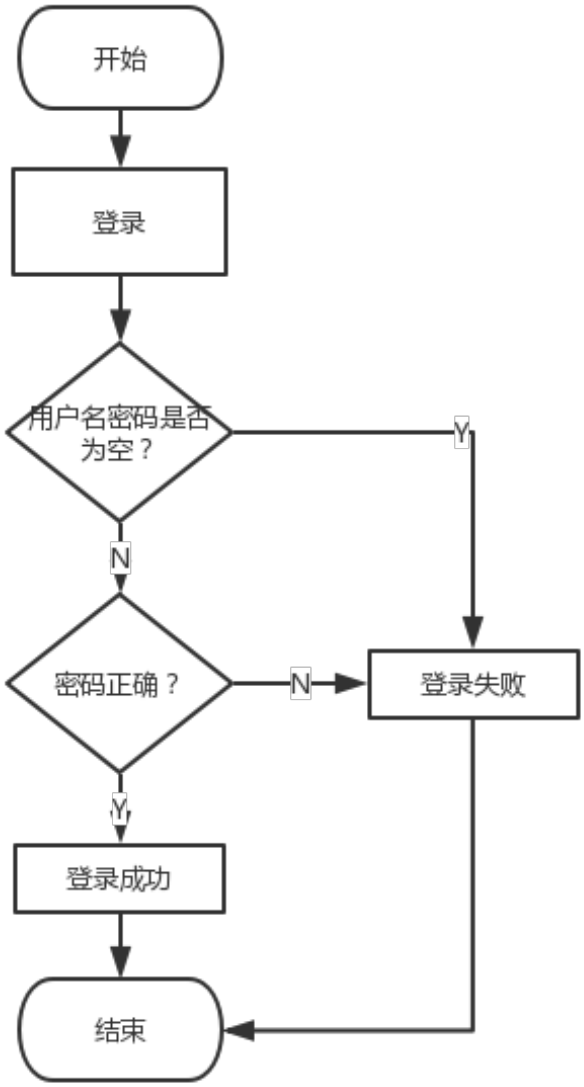
对于根据模拟服务端返回的 JSON 格式的故障信息数据进行显示到列表上。具体返回 JSON 已在前面列出，给出测试用例：

测试范围	期望结果	实际结果
返回 JSON 格式不满足接口要求	无效	无效 提示数据加载失败
返回 JSON 某个字段的值为 Null	无效	有效 程序正常运行，该数据不显示在界面上
返回 JSON 结果正常	有效	有效 数据正常显示

WEB 端：

① 权限管理员登录测试：

流程图：



关键代码：

```
if(userid.value==''){
        alert("用户名为空");
        userid.focus();
        return false;
    }
    if(password.value==''){
        alert("密码为空");
        password.focus();
        return false;
    }
    var xmlhttp;
    if (window.XMLHttpRequest){
        xmlhttp=new XMLHttpRequest();
```

```
}
else{
    xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
}
xmlhttp.onreadystatechange=function(){
    if (xmlhttp.readyState==4 && xmlhttp.status==200){
        var json=xmlhttp.responseText;
        var jsonObject = JSON.parse(json, null);
        if (jsonObject.data.auth=='3') {
            window.location.href='Homepage.html';
         }else{
            alert("WRONG");
         }
    }
}
```

登录模块测试用例

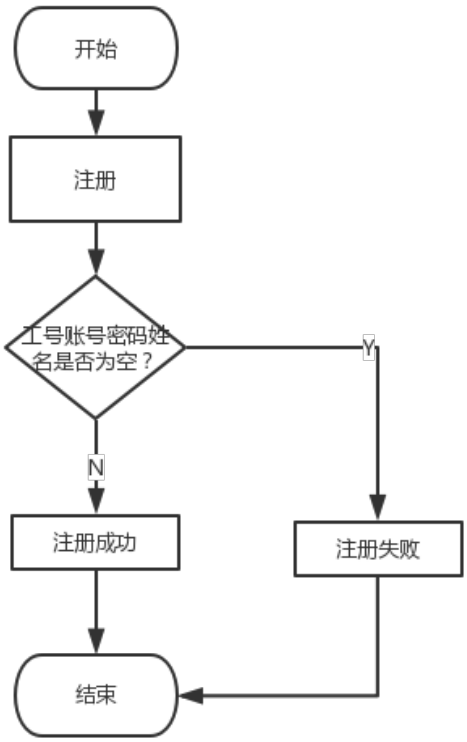
登录检测

输入：用户名 user 和密码 123456。

预期输出：登录成功。

② 注册员工账号测试：

流程图：



关键代码：

```
 if(id.value==''){
        alert("工号为空");
        id.focus();
        return false;
    }
    if(account.value==''){
        alert("账号为空");
        account.focus();
        return false;
    }
    if(password.value==''){
        alert("密码为空");
        password.focus();
        return false;
    }
```

```
    if(name.value==''){
        alert("姓名为空");
        name.focus();
        return false;
    }
    var xmlhttp;
    if (window.XMLHttpRequest){
      xmlhttp=new XMLHttpRequest();
    }
    else{
      xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
    }
    xmlhttp.onreadystatechange=function(){
      if (xmlhttp.readyState==4 && xmlhttp.status==200){
        if (xmlhttp.responseText=="admin regis success") {
          alert("注册成功");
        }
      }
    }
```

注册账号模块测试用例

输入：工号，账号，密码，姓名

预期输出：注册成功。

③ 查看权限功能测试：

关键代码：

```
if (xmlhttp.readyState==4 && xmlhttp.status==200){
    var json=xmlhttp.responseText;
    var node=eval("("+json+")");
```

```
    var str='';
      for (var i = 0; i < node.length; i++) {
        var id = node[i].id;
        var account = node[i].account;
        var auth = node[i].auth;
        var sex = node[i].sex;
        var name = node[i].name;
        var status = node[i].status;
        if(auth=='3')
          continue;
        if(sex=='1')
          sex='男';
        else
          sex='女';
        if(auth=='1')
          auth='职工';
        else
          auth='主管';
        if(status=='0')
          status='空闲';
        else
          status='工作中';
        if (i % 2 == 0) {
          str     +=     '<div     class="gradeContent"     style="background-
color:#F6F680;"><span class="id">' + id + '</span><span class="account">' +
account + '</span><span class="auth">' + auth + '</span><span class="sex">' +
sex + '</span><span class="name">' + name + '</span><span class="status">' +
status+'</span></div>' ;
        }else {
          str     +=     '<div     class="gradeContent"     style="background-
```

```
color:#F4F460;"><span class="id" >' + id + '</span><span class="account">' +
account + '</span><span class="auth">' + auth + '</span><span class="sex">' +
sex + '</span><span class="name">' + name + '</span><span class="status">' +
status+'</span></div>' ;
        }
      }
      document.getElementById("showGrade").innerHTML =str;
  }
```

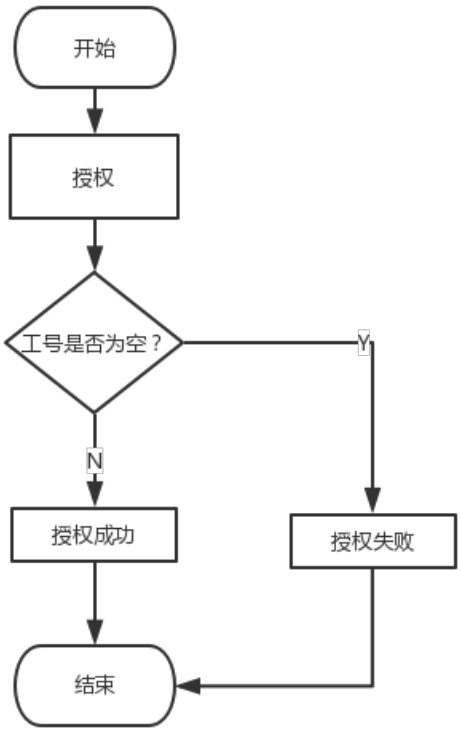
权限查看模块测试用例

输入：点击权限查看

预期输出：所以工作人员的权限列表

④ 授予（更改）权限功能测试：

流程图：



关键代码：

```
 if(id.value==''){
        alert("工号为空");
        id.focus();
        return false;
    }
    var xmlhttp;
    if (window.XMLHttpRequest){
      xmlhttp=new XMLHttpRequest();
    }
    else{
      xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
    }
    xmlhttp.onreadystatechange=function(){
      if (xmlhttp.readyState==4 && xmlhttp.status==200){
        if (xmlhttp.responseText=="success") {
          alert("授权成功");
        }
      }
    }
```

权限授予模块测试用例

输入：工号，权限

预期输出：授权成功

2、 集成测试

采用自底向上的测试方法，先对于每个模块分批组合进行测试，测试通过后模块进一步组合后进行进一步的测试,最后通过黑盒测试测试整个系统的运行情况。

3、 性能测试

(1) 并发测试:

因为能力有限,无法真实投入使用或者找到大量机器作并发测试,因此只测试是否拥有并发能力。同时用三个手机同时使用,一切正常。

(2) 响应速度测试:

因为能力有限,无法测试海量测试,只能生成一定量的测试数据(万级)进行数据响应测试:

网络环境在无线网络的情况下(10mbs),将响应速度统计精确 s,(<1s 则代表响应速度足够快用户无需等待)求出主要功能的平均响应时间:

	响应速度
提交故障申请	<3s(与图片大小有关)
获取故障列表	<1s
查看故障详情	<1s
登录注册	<1s

(3)压力测试

发现权限管理系统的性能在什么情况下会退化或失败。

测试目标：	确定所指定功能在以下情况下的性能行为： 大批量数据录入
方法：	1. 根据写好的测试计划对功能进行测试。 2. 通过修改数据的循环次数来改变一定时间段里数据输入的数量。对不同的数据量进行测试。记录下数据量和对应的资源占用率，响应时间。 3. 在同一台机器上重复测试，记录数据；在不同的机器上进行相同的操作，记录数据。
完成标准：	多个用户：在可接受的时间范围内成功地完成测试脚本，没有发生任何故障。

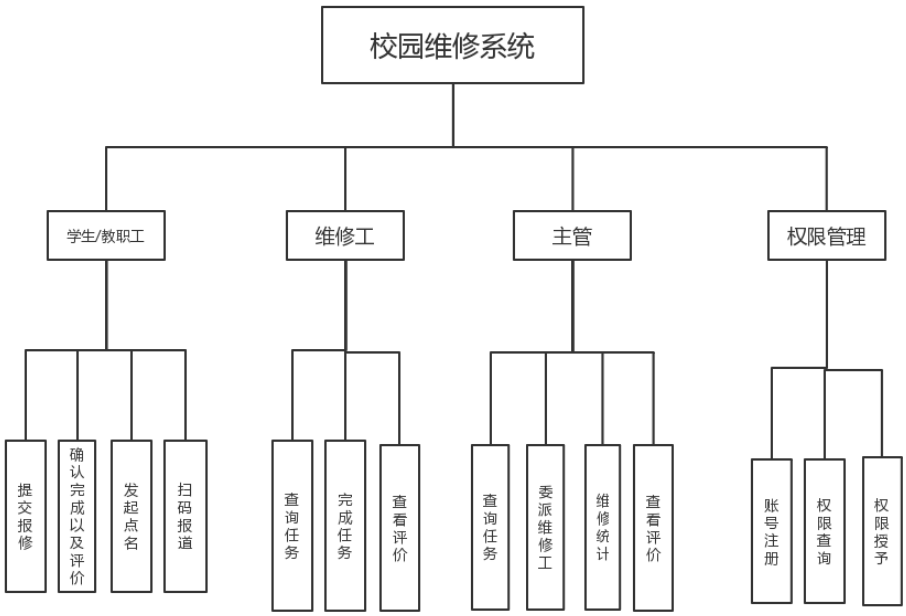
（4）安全性测试

对权限管理系统中的多方面可能出现的危险进行测试。

测试目标：	在系统中，确保不同级别的人员只能进行怎样的操作和数据处理。
方法：	1. 功能验证： 对注册功能、查看权限，授予权限等进行测试。 2. 模拟攻击： 用一组特殊的极端方法对功能进行测试。
完成标准：	1. 主要验证上述功能是否有效。 2. 验证系统的安全防护能力。

4、 功能验收

对于系统设计中的功能模块图中的所有功能都进行功能验收,做出平均与修改建议。



5、 安装测试

Android 端:

根据需求对不同分辨率,不同 Android 版本的手机进行了安装测试,对当前主流的系统版本和屏幕的手机进行了安装测试

测试成果表如下:

系统版本 屏占比	mdpi	hdpi	xhdpi	xxhdpi
Android 5.0	安装运行成功 字体略大	安装运行成功	安装运行成功	安装运行成功

Android 6.0	安装运行成功 字体略大	安装运行成功	安装运行成功	安装运行成功
Android 7.0	安装运行成功 字体略大	安装运行成功	安装运行成功	安装运行成功
Android 8.0	安装运行成功 字体略大	安装运行成功	安装运行成功	安装运行成功

在所有主流的版本中的手机中均安装并成功运行，不过在 mdpi 分辨率的屏幕下会有字体偏大的问题，可以后期进行优化。

服务器后端:

经过测试服务器后端代码打包适用于 jdk1.8 以上的 Java 环境，并且对于主流 tomcat 版本均适配，代码可兼容跨平台运行于 Linux 和 windows 系统之下，其中 Linux 版本支持 CentOS，Debian，Ubuntu 等多个 Linux 版本。

五、软件使用说明书

5.1 Android（师生端）实现情况及使用说明图文介绍

5.1.1 登录注册引导界面

学生用户可以自行注册和登录
老师应使用分配的账号直接登录

5.1.2 登录界面

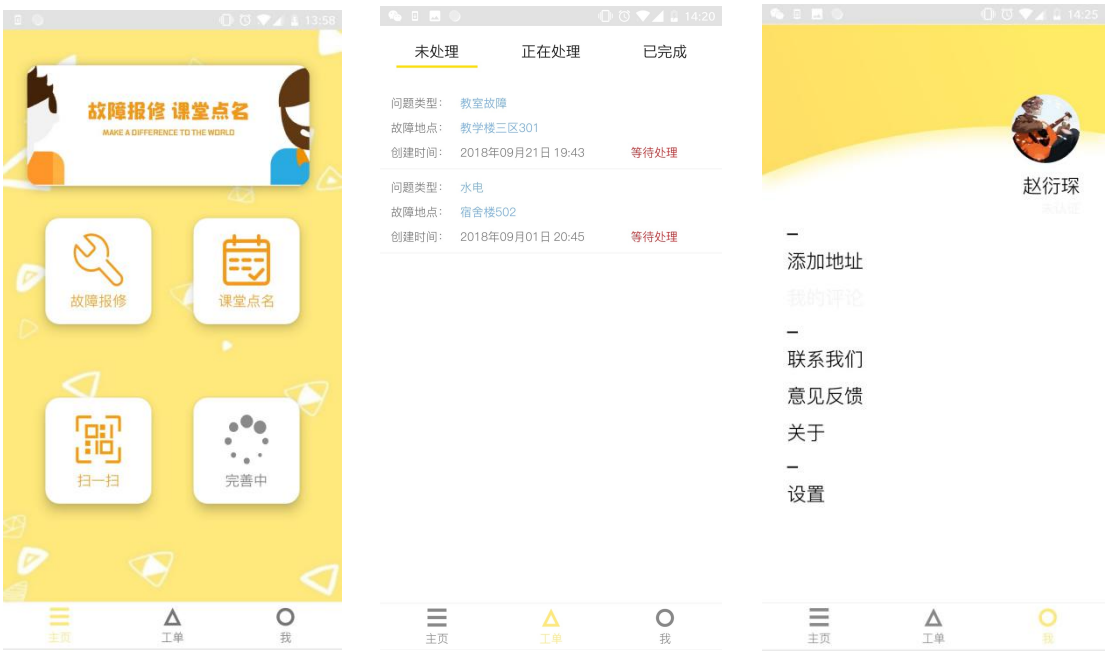
用户根据提示输入用户名和密码登录入系统。

5.1.3 注册界面

学生用户可通过系统自行完成注册，根据提示依次输入信息，账号为个人手机号，方便维修人员进行联系。

5.1.4 主界面

App 主界面预览，主要功能可完成故障报修，课堂点名等功能，并且可以对报修状态进行跟踪，实时查看故障状态，与报修人员沟通。



5.1.5 故障报修界面

填写故障地址，故障类型，并且填写故障详细情况，提交故障图片，点击提交可以将故障信息上传到后台数据中心。

5.1.6 查看报修处理情况界面

处理情况分为未处理，正在处理，已完成三种状态，点击进入查看故障详情。

未处理：报修系统接到报修请求，但还未有维修人员进行处理

正在处理：已分配维修人员进行处理的问题

已完成：维修提出者或维修人员确认已经解决的问题

5.1.7 报修详情界面

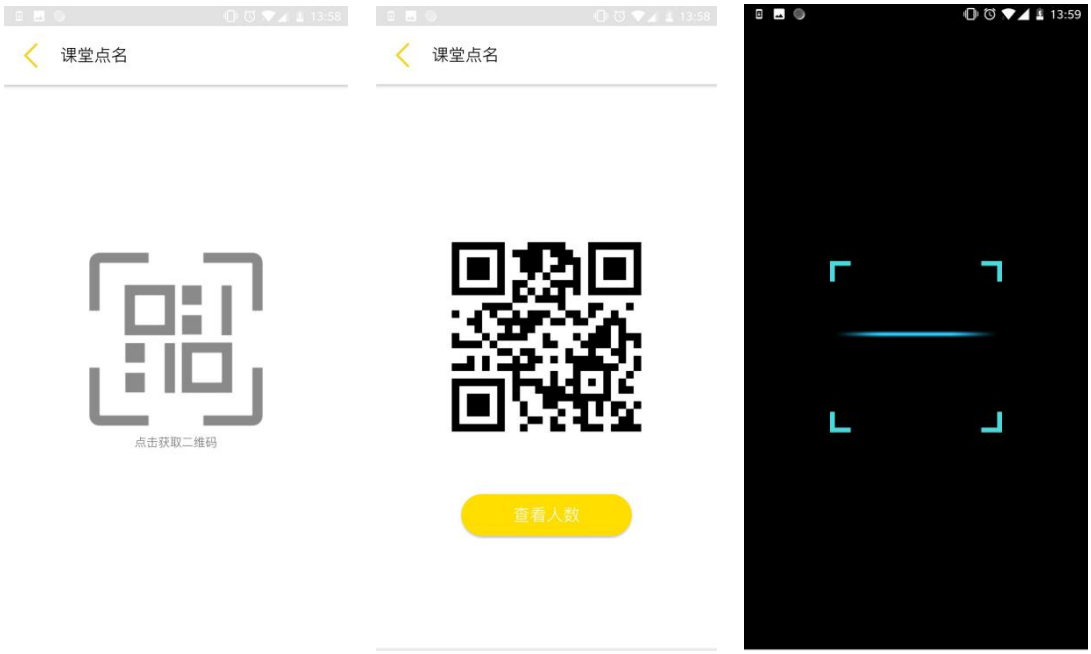
对于未处理的报修问题，还未分配工人，仅显示报修的信息。

对于正在处理的的报修问题，可以显示维修人员姓名和电话号码方便与其联系。也可以在确认故障解决后点击“确认解决”按钮，可以确认完成该次报修。

已成功处理的故障，用户可以填写评论，对故障处理结果进行评价。

5.1.8 课堂点名界面

点击获取二维码，可进行二维码的获取，其他用户扫描二维码可以被记录，点击查看人数可获得详细的名单。



5.2Android（管理员维修员端）实现情况及使用说明图文介绍

5.2.1 登录界面

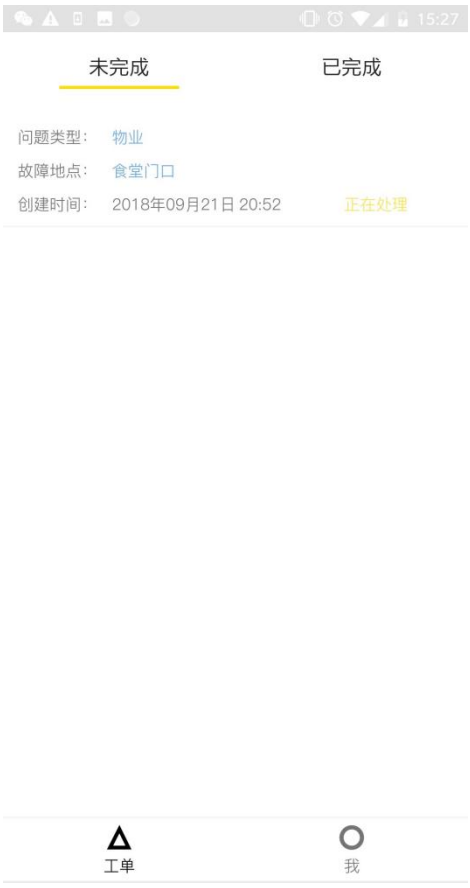
维修人员和维修主管可根据后台分配的的账号和密码进行登录，登入系统后会跟据不同身份显示不同的界面和数据。

维修人员：学校中负责处理故障事故的员工，通过 WEB 端后台权限管理系统进行注册，可通过系统查看其要完成的维修任务情况等信息。

维修主管：学校维修部门主管，通过后台权限管理系统进行注册，允许查看所有状态下的维修任务，并且可以为未处理的任务分配维修工人，可以查看工人状态以及近期维修情况数据图表分析统计。

5.2.2 维修人员界面

维修人员仅可以查看，他被分配的维修任务，以及他已完成的维修任务



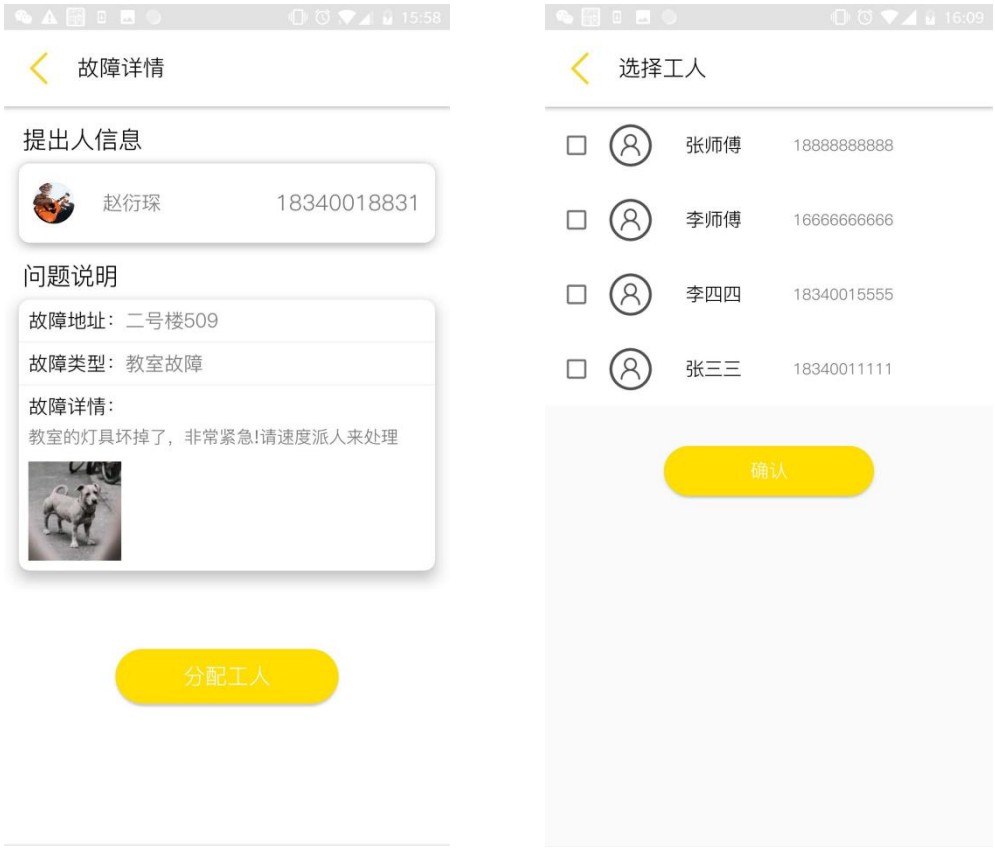
对于他正在处理的报修任务可显示故障详情，以及提出人的信息，并且可以点击确认解决按钮，告知系统已解决该问题。

已完成报修任务可以显示提出人信息，问题详情，以及提出人写的评价。

5.2.3 主管人员界面

维修主管可以查看全部的报修信息，未处理，正在处理和已完成全部报修工单

未处理故障，显示提出人和问题说明信息，并且可为此问题分配工人处理。



正在处理的报修任务显示故障提出人，故障详情，故障的处理者

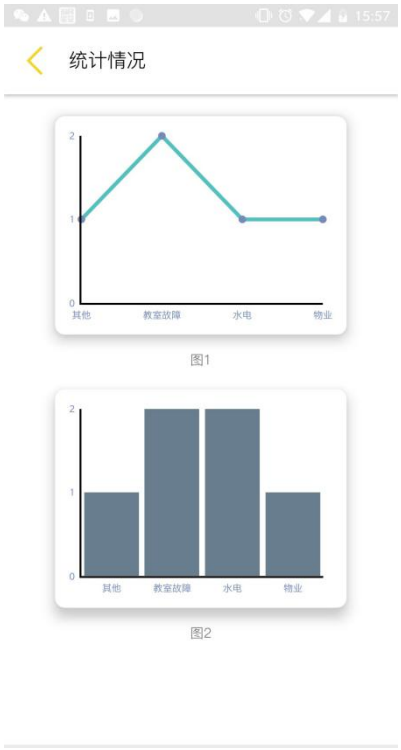
已完成的报修任务显示提出人、问题详情，处理者以及提出者给出的评价。

5.2.4 主管的个人界面

5.2.5 统计情况界面

可显示最近十天各类型故障发生的数量

以及全部时间内各故障发生数量的统计情况



5.3 WEB 端实现情况及使用说明图文介绍

5.3.1 登录界面

权限管理系统的登录界面，使用后台管理员账号登录

5.3.2 主界面

登录后的权限管理系统，菜单第一栏为账号注册，输入主管或维修人员的工号、账号、密码、姓名、性别即可进行注册

菜单栏第二栏权限列表栏，在此表中可以知道已注册人员的工号、账

号、权限、性别、姓名与当前状态

第三栏为授权系统，可以给新注册的用户进行授权，也可以对已授权的用户更改权限

六、安装部署说明书

1、手机 APP 端直接安装用户端与管理端 APP 即可

2、后台服务器部署

① 首先购买服务器之后运行以下 init.sh 文件

```
#!/usr/bin/env bash
mysql_passwd=v9jOBbSSWNQzMZEY

apt update
apt upgrade -y
apt dist-upgrade -y
apt install -y apt-transport-https

wget https://get.docker.com -O get.sh
chmod +x get.sh

./get.sh --mirror Aliyun

mkdir -p /etc/docker
tee /etc/docker/daemon.json <<-'EOF'
{
  "registry-mirrors": ["https://dpzpmu1h.mirror.aliyuncs.com"]
}
EOF

systemctl daemon-reload
systemctl restart docker

tee /home/data/nginx/conf.d/nginx.conf <<-'EOF'
server_tokens off;

server{
    listen 80 default_server;

    keepalive_timeout   60;
```

```
    location / {
        proxy_pass   http://tomcat:8080/;
        proxy_redirect  off;
        proxy_set_header  Host $host:$server_port;
        proxy_set_header  X-Real-IP $remote_addr;
        proxy_set_header  X-Forwarded-For $proxy_add_x_forwarded_for;
    }
}
EOF

docker run -dp 3306:3306 --restart=always -e MYSQL_ROOT_PASSWORD=$mysql_passwd
-v /home/data/mysql:/var/lib/mysql --name mysql mysql
docker run -dp 8080:8080 --restart=always --link mysql:mysql -v
/home/data/tomcat/webapps:/usr/local/tomcat/webapps --name tomcat tomcat:alpine
docker run -dp 80:80 -p 443:443 --restart=always --link tomcat:tomcat -v
/home/data/nginx/conf.d:/etc/nginx/conf.d -v /home/data/nginx/home:/home --name
nginx nginx:alpine
```

② 通过电脑上 Navicat for mysql 连接云端服务器数据库

账号 root 密码 v9jOBbSSWNQzMZEY

然后以下运行 SQL 语句

```
CREATE TABLE user (
    id CHAR(10) NOT NULL PRIMARY KEY,
    account CHAR(18) NOT NULL,
    auth CHAR(10) NOT NULL,
    sex CHAR(1) NOT NULL,
    name CHAR(5) NOT NULL,
    schoolnumber CHAR(15) NOT NULL
);

CREATE TABLE employee(
    id CHAR(10) NOT NULL PRIMARY KEY,
    account CHAR(18) NOT NULL,
    auth CHAR(18) NOT NULL,
    sex CHAR(1) NOT NULL,
    name CHAR(5) NOT NULL,
    status CHAR(5) NOT NULL
);

CREATE TABLE mission(
```

```
    id CHAR(40) PRIMARY KEY,
    type CHAR(20),
    address VARCHAR(100),
    question VARCHAR(255),
    useraccount CHAR(30),
    workeraccount CHAR(30),
    starttime TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    comment VARCHAR(200),
    status CHAR(10)
);

CREATE TABLE login(
    account CHAR(40) PRIMARY KEY,
    password CHAR(40),
    sessionid CHAR(40)
);


CREATE TABLE signname(
    secret CHAR(10),
    person VARCHAR(255)
);
```

③将后台 SpringBoot 打包成 War 包 放入服务器 /home/data/tomcat/webapps 之中稍等。

④可以通过浏览器访问 http://XXXXXXX:8080/admin 获取后台管理页面，手机 APP 可直接使用。