**分类号：××× U D C：D10621-408-(2018)2941-0**

**密 级：公 开 编 号：2014121111**

**成都信息工程大学**

**学位论文**

**基于web的公司考勤系统**

|  |  |
| --- | --- |
| **论文作者姓名：** | **李寰民** |
| **申请学位专业：** | **网络工程** |
| **申请学位类别：** | **工学学士** |
| **指导教师姓名（职称）：** | **曾令明（讲师）** |
| **论文提交日期：** | **2018年6月 4日** |

**基于web的公司考勤系统**

**摘要**

随着Internet的不断发展和普及，互联网和传统的行业结合也越来越密切，对传统工作的改造也是越来越频繁。公司考勤是公司员工的一项重要考核指标，是评估每个员工的重要标准，传统的人工考勤有很多的弊端，基于web的公司考勤系统能够改善和解决这些弊端。所以本次毕业设计就是开发一款基于web的公司考勤系统。

此次毕业设计的系统使用的是Browser/Server（B/S）模式。在技术方面主要使用了Java语言开发、mysql数据库做数据持久化操作，还使用了Spring、SpringMVC、Mybatis框架。前端所使用的主要技术和语言是css，js，HTML。主要的功能包括打卡、个人中心、公司员工、公司部门、角色管理、部门管理、我的考勤、部门资料、请假管理。

本论文从介绍互联网的发展背景以及互联网和传统行业的结合入手，分析了基于web的考勤系统的可行性，阐述了设计的整体结构和各个功能模块，然后详细的介绍了各个模块的功能设计。

关键字：互联网、考勤、B/S模式、Java

**xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx**

**Abstract**

With the continuous development and popularization of Internet, the integration of the Internet and traditional industries is becoming more and more close, and the transformation of traditional work is becoming more frequent. The company attendance is an important evaluation index for the employees, and it is an important standard for evaluating each employee. There are many disadvantages in the traditional manual attendance. The company attendance system based on Web can improve and solve these disadvantages. Therefore, this design is to develop a company attendance system based on Web.

The system is a Browser/Server (B/S) mode attendance system. In terms of technology, Java language is used mainly, MySQL database is used to do data persistence, and Spring, SpringMVC and Mybatis frameworks are also used. The front end is mainly used in CSS, JS, HTML and other technologies. The main functions include punch card, personal center, company staff, company department, role management, department management, my attendance, departmental information and leave management.

This paper begins with the introduction of the development background of the Internet and the combination of the Internet and the traditional industry, analyzes the feasibility of the attendance system based on Web, expounds the overall structure of the design and various functional modules, and then introduces the functional design of each module in detail.

Keywords: Internet, attendance, B/S mode, Java

# 1.引言

## 1.1课题背景

互联网从诞生之后发展很快，各行各业基本上都被互联网颠覆和改造。到现在为止很多公司的管理互联网化和互联网紧密的结合了。市面上也出现了很多CRM软件来辅助公司进行管理和发展。理所当然公司考勤也由传统的人工方式转为了线上考勤。基于web的考勤系统是将公司的考勤从线下迁移到线上，在互联网上操作，将考勤记录的信息存持久化到数据库。相比如传统的考勤有如下优势，信息存储方便，降低了人工的工作量提高了效率，利用互联网强大的计算功能也使得统计分析信息更加容易，使原本枯燥的工作变得有趣了。同时保证了考勤信息的公开和透明。

## 1.2国内外研究现状

互联网起源于欧美国家，从2000年左右互联网才算真正意义上的走进了中国，所以中国的互联网的起步较为迟缓。互联网对传统行业的改造也落后于国外，国外公司考勤技术已经成熟，这一功能是归入了企业资源管理即ERP，公司的一切管理都实现互联网化。中国的互联网目前发展很快，有着巨大的潜力。近几年中国的互联网技术飞速发展，已经达到了世界顶尖水平，国内有很多设计开发公司考勤的服务提供商。中国的企业在自动化这方面做得越来越好。

## 1.3考勤系统的研究意义

21世纪是技术快速更新的，产品快速迭代的，各行各业和互联网快速的结合能提高工作效率。通过对基于web的公司考勤系统的研究，能够更加深入的了解和体会互联网和传统行业的结合方式。考勤是公司的日常运营的一个部分，考勤系统和互联网的结合也为其它方面和互联网结合提供了一个参考。

# 2需求分析

## 2.1系统定义

基于web的公司考勤系统的主要设计目的是帮助公司实现传统考勤向互联网的迁移，通过浏览器查看网页可以查询用户的考勤信息和一些数据的处理。

## 2.2功能需求

### 2.2.1基本功能需求

登录功能：员工使用自己的账号和密码进行登录操作，只有输入正确的账号和密码才能够登录成功进入系统进行其它操作。

打卡功能：用户在系统上实现打卡，签到操作记录出勤时间。

个人信息修改功能：对个人的基本信息实现修改。

考勤信息查看功能：对已经考勤的的信息进行查看。

请假功能：因故不能上班的可以向上级请假。

### 2.2.2部门管理员需求

部门管理功能：审核员工的请假，当有员工需要请假时，进行审核操作。同时还可以查看员工的考勤信息，对员工的出勤情况有大致的了解。

### 2.2.3公司管理员需求

公司员工：用于公司新入职员工和公司员工离职。

公司部门：部门信息的修改和新增加部门。

角色管理：添加新的角色，和角色的授权。

公司部门管理：对部门经理的管理包括请假审核，考勤信息的查看

考勤查看，查看整个公司的考勤信息

## 2.3技术需求

后台开发技术：

后台开发主要使用了Java语言，同时还是用了Mybatis，Spring，SpringMVC框架。其它的技术包括Quartz

前端技术：

前端主要使用了HTML语言来展示文字，辅助使用了css增加页面的样式，使得页面更加美观，好看。JavaScript语言以及对应的JQuery框架用于前后端的数据交互，和一些前端的动态操作。

数据库技术：

数据库使用了MySQL作为数据持久化。

## 2.4开发环境需求

### 2.4.1Eclipse

Eclipse是我们平时开发Java的主要的项目开发工具，它是一个继承开发的环境(IDE)。以及它的衍生产品Myeclipse。Eclipse提供了很多便捷的工具帮助我们开发，比如代码自动补全，代码格式化，还可以继承服务器用于发布我们的项目。所以用Eclipse开发很方便

### 2.4.2Maven

Maven是一个项目管理工具，使用Maven主要是管理项目。还有版本依赖，解决了项目的不同jar包的版本冲突问题，而且也不用再手动导入jar包，很方便。

### 2.4.3Git

Git是一个很好用的版本控制软件，不同于SVN的集中式版本控制软件，它是一个分布式的版本控制软件，每一个人的代码都是一个版本，无需联网就可以实现代码同步。而且还有github这样的托管平台，不用自己假设服务器，节省了很多的资源。

# 3系统设计

## 3.1系统结构设计

这次基于web的考勤系统，我是用的整体结构是B/S结构，用户可以通过浏览器访问架设的服务器。将项目放到服务器上，这样用户只需要浏览器就可以完成所有操作。当需要修改系统时，只需要修改服务器上的系统代码即可，这样就相当于对所有的用户都修改了。

整个系统采用mvc模式即controller-model-view，c层负责处理用户的请求，和服务器资源的返回，m层用于和数据库进行数据库交互和数据的封装，v层负责展示系统界面以供用户进行操作。

## 3.2系统功能设计

### 3.2.1基本功能结构设计

登录功能：用户在进行登录操作时使用自己的账号和密码登录系统，如果账号和密码都是正确的，进入打卡和进入系统界面。

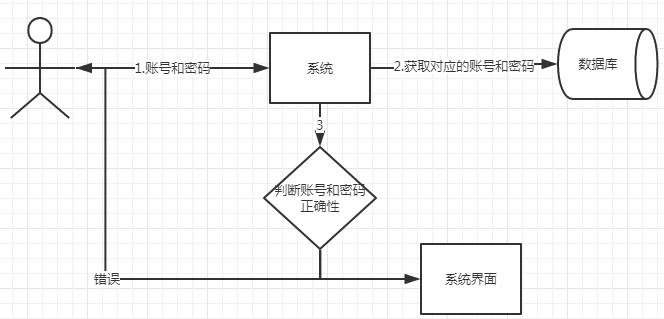


图3.1用户登录图

打卡功能：用户每天早晚两次打卡，如果在中午之前打卡则为上班时间，如果在中午之后打卡则为下班时间。

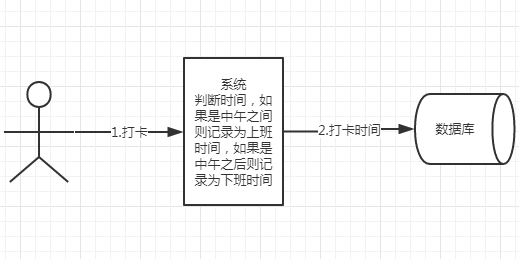


图3.2用户打卡

密码修改：用于用户需要修改自己的密码时使用。

信息管理：当用户第一次登陆系统时，除账号外个人信息是为空的，需要用户自己完善信息。以后如果用户的信息有所变更，可以再信息管理中进行修改个人信息。

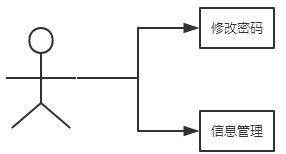


图3.3个人中心

部门资料上传：用于提供部门的公共资料上传。

部门资料下载：用于提供用户所需部门资料的下载。

部门员工查询：员工需要查看同一部门的其它员工的信息时，比如联系方式，邮箱之类的信息，可以在部门员工查询处获取。

部门信息查询：查看部门的基本信息，如员工人数，部门经理，部门的基本理念等。

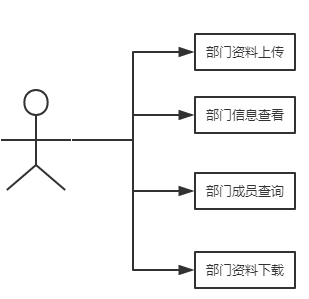


图3.4部门资料

我的考勤：展示用户已经考勤的情况，是否正常签到、旷工，请假信息。以供用户查看自己的考勤信息。

我要请假：提供给用户请假功能，当因事不能上班时，用户可以在此处提交请假申请。

查看请假：当用户提交了请假申请之后，可以在此处查看部门经理审批结果，是否已经同意请假，已经以前的请假历史信息。

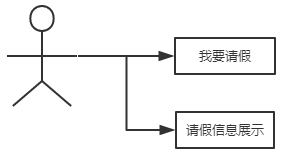


图3.5请假管理

### 3.2.2部门经理流程

员工请假：当员工请假时，部门经理审核员工请假，通过或者不通过。

员工请假记录：查看员工请假的历史记录。

员工考勤：部门经理来查看本部门的员工的考勤信息，对员工的出勤情况有一个大致的了解。

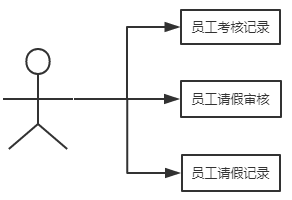


图3.6部门管理

### 3.2.3公司管理者流程

新入职员工： 新入职员工时，添加员工的基本信息到数据库中，包括员工账号，密码，角色，部门信息。

员工管理：用于修改员工的信息，包括员工的密码和基本信息。当员工离职的时候从数据库删除员工的信息，从而实现员工离职操作。

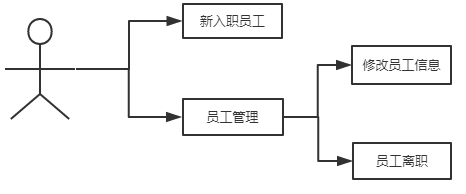


图3.7员工管理

添加部门：用于公司需要新增加部门，添加部门的信息包括部门的名字。

修改部门：修改部门的经理。

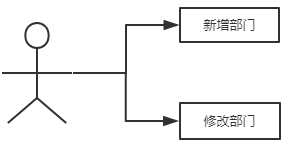


图3.8公司部门

添加角色：角色是用于给员工身份的标识，系统提供了三种角色普通员工，部门经理，CEO。不同的角色有着不同的权限。可以添加角色并且授予响应的权限资源。

修改角色：用于修改角色的基本信息和修改角色的权限资源。

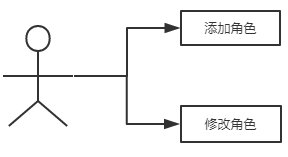


图3.9角色管理

## 3.4数据库概要设计表之间的关系呢？

用户基本信息表&用户标识表

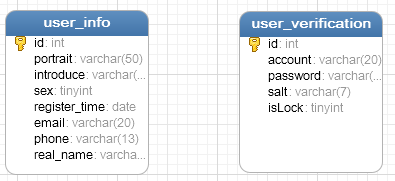


图3.10

角色表&用户角色表&角色菜单表

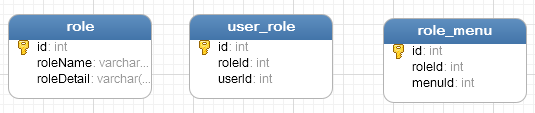


图3.11

部门表&用户部门表&部门文件表

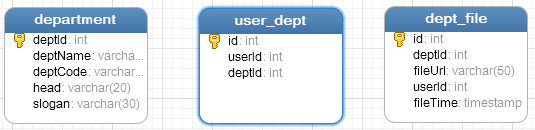


图3.12

请假表&考勤表&菜单表

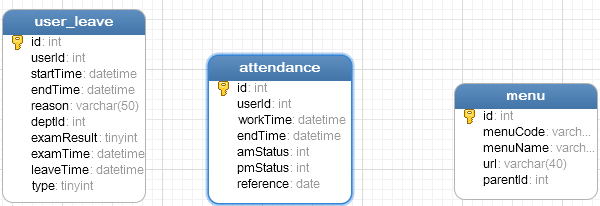


图3.13

# 4详细设计

## 4.1考勤模块

### 4.1.1考勤模块详细设计

考勤模块主要有考勤打卡和考勤信息查看两个功能。 考勤打开分为上午和下午两次，上午记录员工上班时间，下午记录员工下班时间。用户打卡时根据中午时间为基准判断是属于上班时间还是下班时间。考勤信息查看，用户只能查看自己的考勤信息但是不能修改考勤信息。

### 4.1.2考勤模块流程图

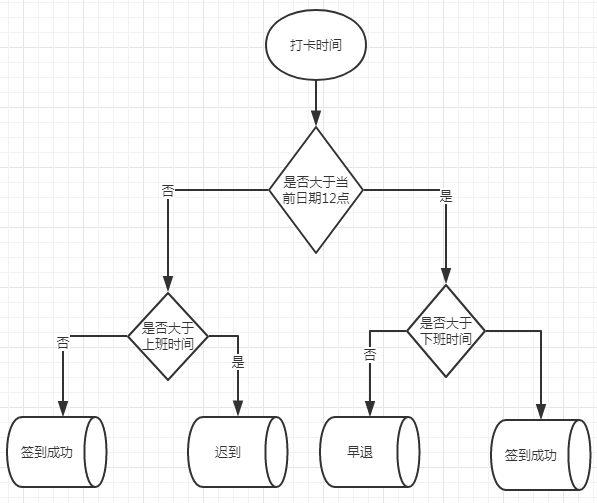


图4.1 xxx

### 4.1.3核心代码

Date date = new Date();

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

Attendance a = attendanceMapper.selectByUserId(userId, sdf.forma

t(date));

Date d = new Date();

SimpledDateFormat s = new SimpledDateFormat("yyyy-MM-dd HH:mm:ss");

String now = s.format(d);

ResultMsg msg = new ResultMsg();

//上午

if(now.compareTos(noon()) <= 0) {

if(a.getWorkTime()==null) {

a.setWorkTime(new Date());

//上午签到

if(now.compareTo(am()) <= 0) {

a.setAmStatus(2);

}else {

//上午迟到

a.setAmStatus(1);

}

}else {

return new ResultMsg(Boolean.FALSE, "上午已签到");

}

}else{//下午

if(a.getEndTime() == null) {

a.setEndTime(new Date());

//下午早退

if(now.compareTo(pm())<0) {

a.setPmStatus(1);

}else {

a.setPmStatus(2);

}

}else {

return new ResultMsg(Boolean.FALSE, "下午已签到");

}

}

### 4.1.4界面设计

考勤信息展示界面展示考勤信息的列表，在上面有一个日期的范围搜索，用于搜索日期范围内的考勤信息。同时提供分页功能，用于数据过多时分页展示。



图4.2

## 4.2请假模块

### 4.2.1请假模块详细设计

用户请假输入请假开始日期和结束日期以及请假原因，系统判断请假日期是否符合（即是否是以前的日期）。用户提交申请后等待上一级领导审批，同时将审批后的结果展示给用户。

### 4.2.2请假模块流程图

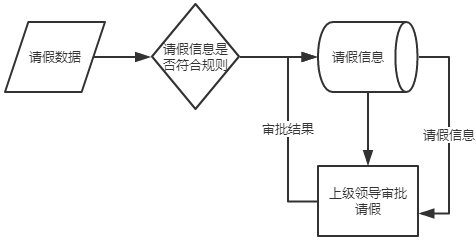


图4.3

### 4.2.3核心代码

UserLeave leave = new UserLeave();

//用户部门

int deptId = userDeptMapper.findDeptIdByUserId(userId);

//查询用户角色，根据角色判断用户请假类型

UserRole userRole = userRoleMapper.selectByUId(userId);

//角色id为1，类型为普通员工

if(userRole.getRoleId()==1)

leave.setType((byte)0);

//角色id为2角色为部门经理，类型为1部门经理

else if(userRole.getRoleId()==2){

leave.setType((byte)1);

}else {

//角色id为三角色为ceo，类型为2ceo

leave.setType((byte)2);

}

### 4.2.4界面设计

请假界面

请假界面是由三个输入框前两个是请假开始的时间，下面一个是请假的原因。

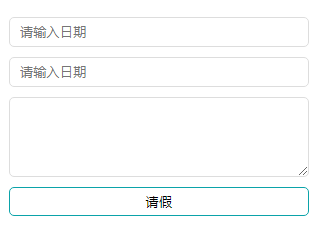


图4.4

请假展示列表

请假信息展示用于展示请假历史信息，包括请假日期范围以及是否审核等。在上面有一个用于提供日期范围搜索的功能，用于用户搜索日期范围内的请假信息。同时提供分页功能用于数据量过大时分页显示信息。



图4.5

审核请假

审核请假时用于领导审批员工请假，在每一条请假信息后面有两个按钮“同意”或“不同意”用于审核员工请假的操作。同时提供分页功能用于数据量过大时分页显示。

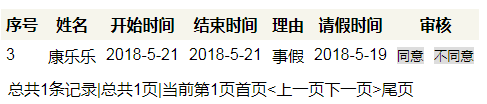


图4.6

## 4.3个人中心&登录模块

### 4.3.1个人中心模块详细设计

修改个人密码的时候需要用户输入旧密码，新密码，确认新密码。首先系统需要判断原密码是否正确，然后判断新密码和确认密码是否一致。

在修改个人信息界面，将用户的基本信息展示到界面，已提供用户查看修改，用户修改时需要判断用户输入的信息是否符合规则，需要对电话号码，邮箱进行校验。如果所有信息都准确无误则修改用户的个人信息。

登录功能设计，系统获取用户输入的账号和密码，根据用户输入的账号从数据库获取用户的加密密码，盐。然后将用户的密码和盐一起进行hash算法并和加密密码判断如果两者相同则用户登录成功进入主界面，否则返回登录界面并提示用户信息输入错误。

### 4.3.2个人中心流程图

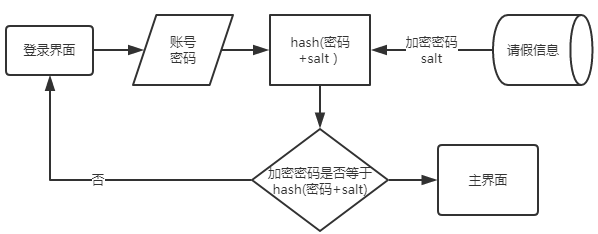


图4.7登录流程

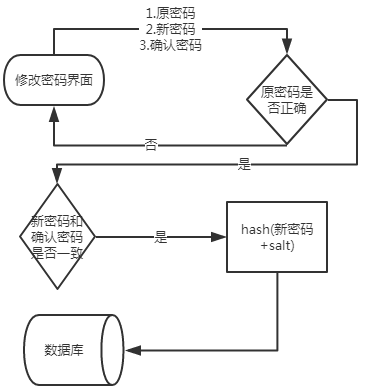


图4.8修改密码流

### 4.3.3核心代码

1. 登录核心代码

UserVerification userVerification = userVerificationMapper.selectUserByAccount(account);

if(userVerification == null)

return new ResultMsg(Boolean.*FALSE*, "用户名或密码错误");

if(userVerification != null&&userVerification.isLock())

return new ResultMsg(Boolean.*FALSE*, "你已被锁定，请联系管理员");

//将用户输入的密码和slat一起进行加密

String loginPasswd = MD5.*GetMD5Code*(userPasswd + userVerification.getSalt());

//判断数据库密码和用户输入的密码if(!loginPasswd.equals(userVerification.getPassword()))

return new ResultMsg(Boolean.*FALSE*, "用户名或密码错误");

HttpSession session = request.getSession();

session.setAttribute("userId",userVerification.getId());

session.setAttribute("userAccount", userVerification.getAccount());

1. 修改个人密码

UserVerification user = userVerificationMapper.

selectUserById(userPassword.getUserId());

if(!user.getPassword().equals(MD5.*GetMD5Code*(userPassword.getOld()+user.getSalt())))

return new ResultMsg(Boolean.*FALSE*, "原密码不正确");

if(userPassword.getOld().equals(userPassword.getNewPassword()))

return new ResultMsg(Boolean.*FALSE*, "新旧密码一致");

if(!userPassword.getConfirm().equals(userPassword.getNewPassword()))

return new ResultMsg(Boolean.*FALSE*, "新密码和确认密码不一致");

UserVerification userVerification = new UserVerification();

userVerification.setSalt(userPassword.getSalt());

userVerification.setId(userPassword.getUserId());

userVerification.setPassword(MD5.*GetMD5Code*(userPassword.getNewPassword()+userPassword.getSalt()));

Int result = userVerificationMapper.updateByPrimaryKeySelective(userVerification);

### 4.3.4界面设计

修改密码界面设计为三个输入框，用户需要依次输入旧密码，新密码，确认密码。三个信息都是必填信息。



图4.9修改密码界面

修改个人信息界面首先是展示出用户的基本信息，第一个输入框为用户的账号信息，这个是不允许用户修改的只是提供展示，后面的基本信息用户都是可以修改的。

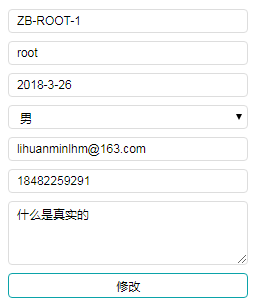


图4.10修改个人信息

登录界面需要用户输入登录的账号和密码，还提供了忘记密码功能，如果用户忘记了密码，可以店家忘记密码找回密码。

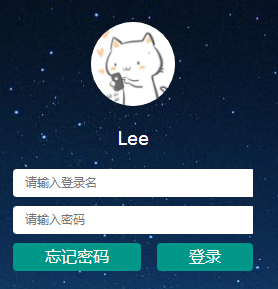


图4.11登录界面

## 4.4公司员工模块

### 4.4.1公司员工模块详细设计

公司员工模块包括新入职员工和员工离职，员工入职时直接添加员工信息，员工账号是员工部门拼音缩写加上员工姓名拼音缩写加上员工数量编号，员工密码默认的是123456。同时还需要添加员工的角色信息，员工所属部门信息。员工离职的时候需要删除员工的信息，员工角色信息，员工部门信息，员工考勤信息，员工请假信息等一切信息。

### 4.4.2公司员工模块流程图

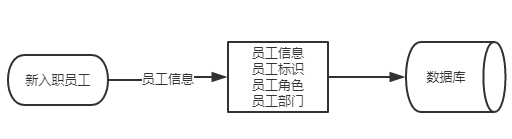


图4.12

### 4.4.3核心代码

UserVerification userVerification = new UserVerification();

String salt = Salt.*salt*();

userVerification.setSalt(salt);

userVerification.setPassword(MD5.*GetMD5Code*("123456"+salt));

userVerification.setAccount(user.getPrefixAccount()+"-"+user.getSuffixAccount()+"-"+user.getNumber());

int vRe = userVerificationMapper.

insertSelective(userVerification);

if(vRe == 0)

return new ResultMsg(Boolean.*FALSE*, "添加失败");

UserInfo userInfo = new UserInfo();

userInfo.setId(userVerification.getId());

userInfo.setPortrait("/attendance/static/img/background\_a.jpg");

System.*out*.println("sex"+user.getSex());

if("1".equals(user.getSex())){

userInfo.setSex(true);

}else {

userInfo.setSex(false);

}

userInfo.setRegisterTime(new Date());

userInfo.setEmail("");

userInfo.setPhone("");

userInfo.setIntroduce("");

userInfo.setRealName(user.getUserName());

if(userInfoMapper.insertSelective(userInfo)==0)

return new ResultMsg(Boolean.*FALSE*, "添加失败");

Department dept = departmentMapper.selectDeptById(new Integer(user.getDeptId()));

if(dept != null) {

UserDept userDept = new UserDept();

userDept.setDeptId(dept.getDeptId());

userDept.setUserId(userVerification.getId());

userDeptMapper.insertSelective(userDept);

} else

return new ResultMsg(Boolean.*FALSE*, "部门不存在");

Role role = roleMapper.selectByPrimaryKey(new Integer(user.getRole()));

if(role != null) {

UserRole userRole = new UserRole();

userRole.setUserId(userVerification.getId());

userRole.setRoleId(role.getId());

userRoleMapper.insertSelective(userRole);

}else

return new ResultMsg(Boolean.*FALSE*, "角色不存在");

### 4.4.4界面设计

1. 添加员工

添加员工界面第一个是选择用户的部门，然后输入用户的姓名，其中用户的账号是部门的拼音加上姓名的缩写加上员工人数组成的。

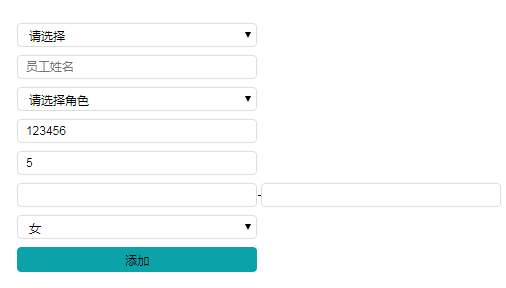


图4.13添加员工

1. 员工管理

员工管理用一个表格展示了员工的信息，其中提供了以用户姓名的模糊查询查询员工，同时提供了以部门名称查询员工功能。在每一个员工数据后面还提供了删除功能，用于员工离职的时候删除员工信息。



图4.14员工管理

## 4.5公司部门模块

### 4.5.1公司部门模块详细设计

公司部门主要用于新增加部门，部门编号为部门名称的拼音首字母，添加部门时获取输入的部门信息，部门经理设置为空，添加到数据库中。修改部门信息主要用于修改部门的基础信息，同时也可以修改部门的经理，通过获取部门的员工提供选择，修改原有经理的经理的角色为员工，并修改信任经理的角色为经理。然后保存到数据中。以保证部门的经理属于该部门的员工。

### 4.5.2公司部门流程图

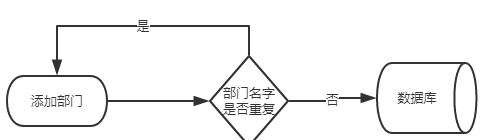


图4.15添加部门

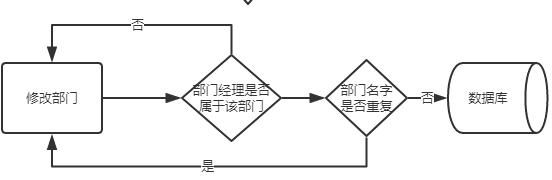


图4.16修改部门

### 4.5.3核心代码

// 查询名字是否重复

String reNameN = departmentMapper.getNameByName(

department.getDeptName());

if(null!=reNameN&&!"".equals(reNameN)) {

return new ResultMsg(Boolean.*FALSE*, "部门名字已重复，请重新输入");

}

//查询编号是否重复

String reNameC = departmentMapper.getNameByCode(department.getDeptCode());

if(null!=reNameC&&!"".equals(reNameC))

return new ResultMsg(Boolean.*FALSE*, "编号已经重复，请在原有基础上添加后缀或前缀");

//添加部门

int result = departmentMapper.insertSelective(

department);

//查询名字是否重复

String reNameN = departmentMapper.getNameByNameNoId(

department.getDeptName(), department.getDeptId());

if(null!=reNameN&&!"".equals(reNameN)) {

return new ResultMsg(Boolean.*FALSE*, "部门名字已重复，请重新输入");

}

// 查询编号是否重复

String reNameC = departmentMapper.getNameByCodeNoId(

department.getDeptCode(), department.getDeptId());

if(null!=reNameC&&!"".equals(reNameC))

return new ResultMsg(Boolean.*FALSE*, "编号已经重复，请在原有基础上添加后缀或前缀");

//查询部门员主管

Department dept = departmentMapper.selectDeptById(

department.getDeptId());

if(!"暂无".equals(dept.getHead())) {

UserInfo userInfo = userInfoMapper.selectByRealName(dept.getHead());

userRoleMapper.updateRoleByUserId(userInfo.getId());

}

if(department.getHead()!="暂无"){

UserInfo use = userInfoMapper.findUserBlDe(department.getHead(), department.getDeptId());

if(use==null)

return new ResultMsg(Boolean.*FALSE*, "该员工不属于该部门");

UserInfo userInfo = userInfoMapper.selectByRealName(

department.getHead());

//查询用户是否存在

if(userInfo==null)

return new ResultMsg(Boolean.*FALSE*,"该员工不存在");

UserRole userRole = new UserRole();

userRole.setUserId(userInfo.getId());

userRole.setRoleId(2);

int r = userRoleMapper.updateByUserId(userRole);

if(r <= 0)

return new ResultMsg(Boolean.*FALSE*, "修改角色失败");

}

int result = departmentMapper.updateByPrimaryKeySelective(

department);

### 4.5.4界面设计

1. 新增部门

新增部门界面需要输入部门的名称，部门的编号是由部门名称拼音的首字母组成的，在添加部门的时候不能添加部门主管，因为这个时候部门下面还没有员工。

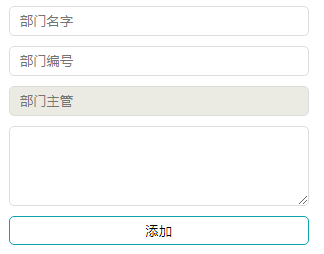


图4.17新增部门界面

1. 编辑部门

编辑部门信息界面展示了部门信息，然后用户可以进行修改。

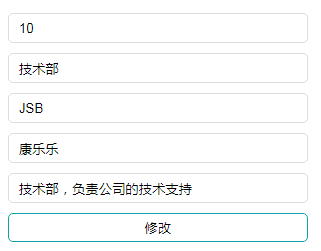


图4.18编辑部门信息界面

## 4.6角色模块

### 4.6.1角色模块详细设计

角色模块包括了添加角色和角色信息修改，在添加角色时输入角色名称和角色的描述信息以及角色相应的资源。判断角色名称是否已经重复，如果角色没有重复执行添加操作。

角色修改时，对角色的信息修改，同时删除角色原有的资源。判断角色名称是否重复，如果没有重复执行修改操作。

### 4.6.2角色模块流程图

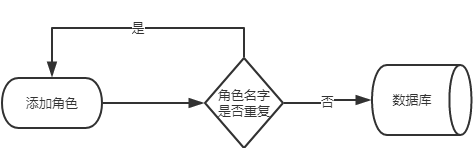


图4.19添加角色流程

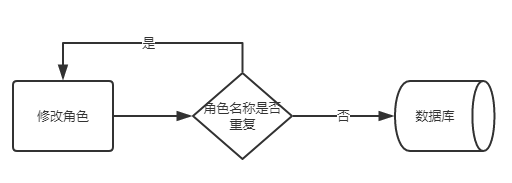


图4.20修改角色

### 4.6.3核心代码

Role role = new Role();

role.setId(roleMenuUpdate.getRoleId());

role.setRoleName(roleMenuUpdate.getRoleName());

role.setRoleDetail(roleMenuUpdate.getRoleDetail());

int r1 = roleMapper.updateByPrimaryKey(role);

//删除角色菜单

int r2 = roleMenuMapper.deleteMenuById(roleMenuUpdate

.getRoleId());

//插入角色菜单

int r3 = roleMenuMapper.insertRoleMenu(roleMenuUpdate

.getRoleId(), roleMenuUpdate.getMenu());

### 4.6.4界面设计

添加角色界面需要用户输入角色名称和角色的详情，同时下面提供了角色的功能，需要勾选响应的功能，赋予角色响应的权限。

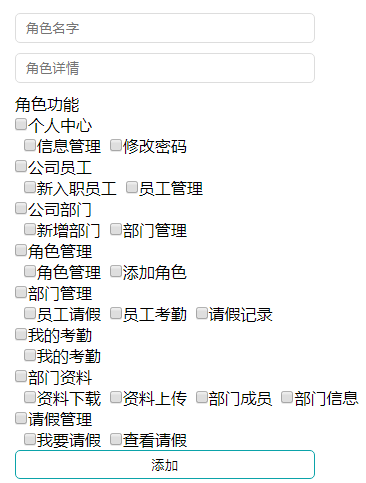


图4.21添加角色

修改角色界面展示角色的基本信息，以及响应的资源权限信息，方便用户修改。

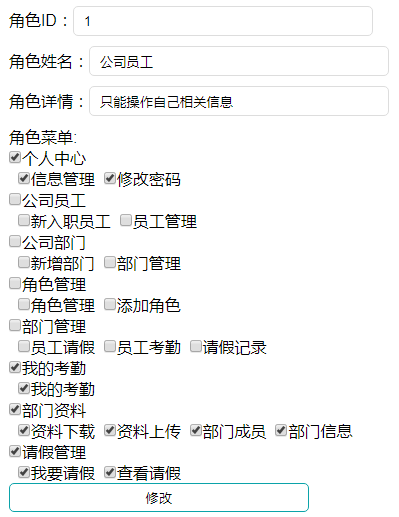


图4.22修改角色界面

## 4.7定时任务模块

### 4.7.1定时任务模块描述

定时任务包括每天考勤数据插入，请假判断。

考勤信息插入是在考勤表中插入所有员工当天的考勤数据，考勤状态全部设定为旷工，员工考勤后进行修改。

请假判断是查询旷工的员工，然后从请假表中获取数据判断员工是否处于请假状态，如果员工处于请假则将旷工状态修改为请假。

### 4.7.2核心代码

//查询上午旷工的人

List<Attendance> list = new ArrayList<Attendance>();

list = attendanceService.findDayAttendByDay(getYtd());

//查询该员工上午是否请假

if(list.size()>0) {

for(int i = 0; i < list.size(); i++) {

//根据用户id获取是否有请假的记录，并且这个时间在请假时间范围内

UserLeave userLeave = leaveService.findLeave(getDayStartTime(), getDayEndTime(), list.get(i).getUserId());

if(userLeave != null) { attendanceService.updateAmtatus(list.get(i).getId());

}

}

}else {

}

## 4.8数据库详细设计

### 4.8.1用户类表

表4.1用户信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 是否为空 | 字段解释 |
| Id | Int | Not | 唯一标识符 |
| Portrait | Varchar | Not | 头像 |
| Introduce | Varchar | Not | 个人介绍 |
| Sex | Tinyint | Not | 性别：0女，1男 |
| Register\_time | Date | not | 注册时间 |
| Email | Varchar | Not | 邮箱 |
| Phone | Varchar | Not | 电话 |
| Real\_name | Varchar | Not | 真实姓名 |

表4.2用户表示表

|  |  |  |  |
| --- | --- | --- | --- |
| Id | Int | Not | 唯一标识 |
| Account | Varchar | Not | 账号 |
| Password | Varchar | Not | 密码 |
| Salt | Varhcar | Not | 盐 |
| isLock | Tinyint | Not | 锁定0：未，1：锁定 |

表4.3用户角色表

|  |  |  |  |
| --- | --- | --- | --- |
| Id | Int | Not | 唯一标识 |
| roleId | Int | Not | 角色id |
| userId | Int | Not | 用户id |

表4.4用户部门表

|  |  |  |  |
| --- | --- | --- | --- |
| Id | Int | Not | 唯一标识 |
| userId | Int | Not | 用户id |
| deptId | Int | Not | 部门id |

### 4.8.2考勤表

表4.5考勤表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 允许为空 | 字段解释 |
| Id | Int | Not | 唯一标识 |
| userId | Int | Not | 用户id |
| workTime | Datetime |  | 上班时间 |
| endTime | Datetime |  | 下班时间 |
| amStatus | Int |  | 上班状态  0旷工，1迟到，2签到，3请假 |
| pmStatus | Int |  | 下班状态  0旷工，1早退，2签到，3请假 |
| Reference | Date | Not | 考勤日期 |

### 4.8.3请假表

表4.6请假表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 类型 | 是否为空 | 字段解释 |
| Id | Int | Not | 唯一标识 |
| userID | Int | Not | 用户id |
| startTime | Datetime | Not | 开始时间 |
| endTime | Datetime | Not | 结束时间 |
| Reason | Varchar | Not | 请假原因 |
| deptId | Int | Not | 部门id |
| examResult | Tinyint | Not | 审批结果  0未审核，1未通过，2通过 |
| examTime | Datetime |  | 审批时间 |
| leaveTime | Datetime |  | 请假时间 |
| Type | Tinyint | Not | 0普通员工1，部门经理 |

### 4.8.4部门类表

表4.7部门表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 是否为空 | 字段解释 |
| deptId | Int | Not | 部门id |
| deptName | Varchar | Not | 部门名字 |
| deptCode | Varchar | Not | 部门编号 |
| Head | Varchar | Not | 部门主管 |

表4.8部门文件表

### 4.8.5角色类表

表5.9角色表

|  |  |  |  |
| --- | --- | --- | --- |
| Id | Int | Not | 唯一标识 |
| roleName | Varchar | Not | 角色名称 |
| roleDetail | Varchar | Not | 角色详情 |

表5.10角色菜单表

|  |  |  |  |
| --- | --- | --- | --- |
| Id | Int | Not | 唯一标识 |
| roleId | Int | Not | 角色id |
| menuId | Int | Not | 菜单id |

表5.11菜单表

|  |  |  |  |
| --- | --- | --- | --- |
| Id | Int | Not | 唯一标识 |
| menuCode | Varchar | Not | 菜单编号 |
| menuName | Varchar | Not | 菜单名称 |
| Url | Varchar | Not | 菜单地址 |
| parentId | Int | Not | 上级菜单 |

# 5测试

## 5.1请假功能测试

### 5.1.1期望结果

用户提交请假申请，部门经理收到用户的请假申请，并且在部门管理目录下能够审批用户请假申请。用户可以在请假管理中查看请假审批结果。

### 5.1.2测试结果

测试用户王瑞申请请假



图5.1

部门经理审批之前



图5.2

部门经理显示用户请假申请，并选择同意用户申请

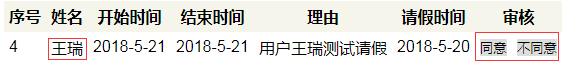


图5.3

用户王瑞的请假结果已经发生修改



图5.4

### 5.1.3结论

用户能够成功申请请假，部门经理能够收到用户的请假申请，审批后的结果能够及时展示给用户，测试功能合格。

## 5.2考勤功能测试

### 5.2.1期望结果

用户康乐乐能够打卡，并且能更具打卡时间判断打卡状态。

### 5.2.2测试结果

能够成功的打卡，并且能够根据打卡时间判断打卡状态。

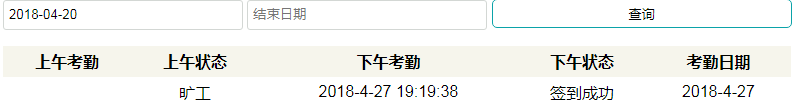


图5.5

### 5.2.3结论

测试功能合格

## 5.3添加员工功能测试

### 5.3.1期望结果

能够成功添加用户，用户能够实现登录。

### 5.3.2测试结果

添加员工

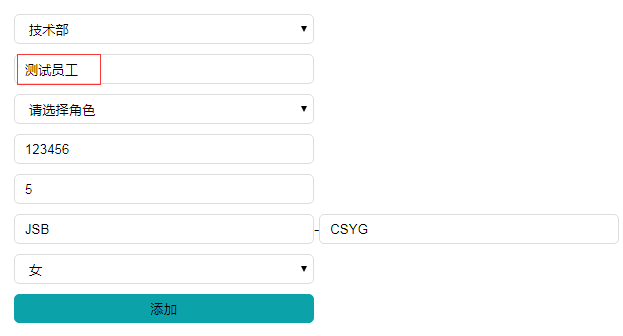


图5.6

新员工登录后可以修改个人信息

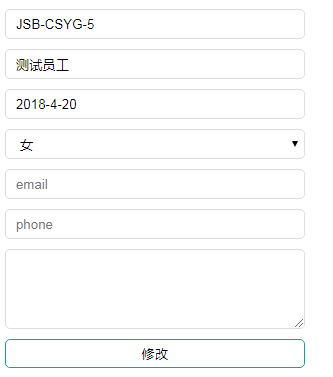


图5.7

### 5.3.3结论

测试功能合格

# 6结语

此次毕业设计前前后后大概花费了将近两个月的时间，付出了很多时间，同时也收获了很多东西。项目代码的完成和毕业论文文档的完成，也意味着大学的生活就要结束了。这次毕业设计的题目是基于web的公司考勤系统设计，前期设想了很多的功能，在实现这些功能的过程中也发现了很多设计上的缺陷，有些功能也推翻重来过，也遇到过一些困难，比如关于请假和考勤数据的统一，还有定时任务的设置，这些都是以前没有遇到过的，最后通过自己的思考和对老师的请教也解决了这些问题。本次设计的系统基本上是满足了公司的基本需要，但是如果是如果是高层次需求仍然需要继续修改和增加功能。这个系统目前只是一个出版，后面会继续迭代版本，当然不仅仅是为了毕业设计，一款产品应该不断的打磨，变得越来越好。考勤系统现在用的越来越多，很多传统的人工考勤也基本上被淘汰了。所以为了一款好的考勤系统应该有很大的市场前景。