目 录

[1、引言 1](#_Toc197699655)

[**1.1**背景 1](#_Toc197699656)

[**1.2** 参考资料 1](#_Toc197699657)

[2、总体设计 1](#_Toc197699658)

[**2.1** 需求规定 1](#_Toc197699659)

[**2.2** 运行环境 1](#_Toc197699660)

[**2.2.1** 设备 1](#_Toc197699661)

[**2.2.2** 支持软件 2](#_Toc197699662)

[**2.3** 基本设计概念和处理流程 2](#_Toc197699663)

[**2.4** 功能模块设计 3](#_Toc197699664)

[3、数据库的逻辑设计 5](#_Toc197699665)

[**3.1** 主要数据表的结构 5](#_Toc197699666)

[4、关键功能实现 8](#_Toc197699667)

[**4.1** 用户登录功能实现 8](#_Toc197699668)

[**4.2** 考勤打卡功能实现 9](#_Toc197699669)

[**4.3** 请假卡片功能实现 11](#_Toc197699670)

[**4.4** 添加员工功能实现 12](#_Toc197699671)

[**4.5** 考勤数据统计功能实现 14](#_Toc197699672)

[5、系统不足与改进方向 16](#_Toc197699673)

[**5.1** 系统存在的不足 16](#_Toc197699674)

[**5.2** 未来改进方向 16](#_Toc197699675)

# **1、**引言

随着信息化的普及，企业管理信息化需求的不断提升，传统人工管理模式在员工信息管理方面存在诸多问题。这些问题不仅影响了企业的日常运营效率，还可能导致决策依据的偏差，这直接影响到企业的长期发展，因此加强企业的考勤管理十分重要。寻常的机器打卡虽然一定程度上解决了日常考勤的问题，但是面临排队打卡，定期维护设备的问题时，还是效率还是低下。

## **1.1**背景

待开发的软件的名称为：考勤打卡系统

本项目的任务提出者：全体开发人员

开发者：杨颖

用户：学生

## **1.2** 参考资料

《基于SSM框架的学生信息管理系统的设计与实现》现代信息科技 王建国, 张争, 汤来锋等

《基于SSM框架的企业人事管理系统的设计与实现》湖北工业大学 文鹏

《Utilization of Artificial Intelligence (AI) in Information Systems to Improve Business Efficiency》 ProBisnis: Jurnal Manajemen Nasrullah M

# **2、**总体设计

## **2.1** 需求规定

本设计结合人脸识别技术，实现了非接触式考勤打卡，提高了考勤的准确性和效率。请假处理模块支持员工在线提交请假申请，管理员在线审批，简化了请假流程，提高了审批效率。考勤数据实时存储和统计，确保数据的真实性和及时性，为管理层提供了准确的决策依据。

## **2.2** 运行环境

### **2.2.1** 设备

开发工具：IDEA

运行环境：笔记本

软件环境：Windows 11

### **2.2.2** 支持软件

本系统： Windows 11

数据库： MySQL 8

Web服务器： Tomcat 10

Java版本： 23

## **2.3** 需求分析

高层管理人员是系统的主要管理者，负责信息管理和数据统计分析，确保员工信息准确、考勤数据公正和审批部长的请假申请，必要时根据相应的流程修改员工的考勤状态解决漏打卡等问题。如图2.1。

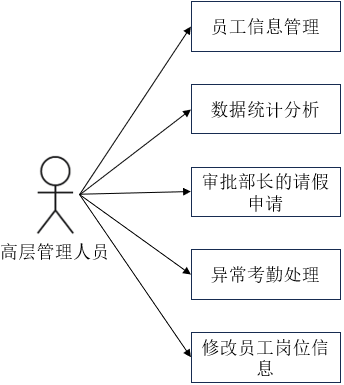


图2.1 高层管理人员用例图

部门部长需实时查看团队成员出勤，合理安排工作，及时审批请假申请，修改本部门员工基本信息。普通员工通过系统打卡，查看考勤统计，修改密码和联系方式，如图2.2。

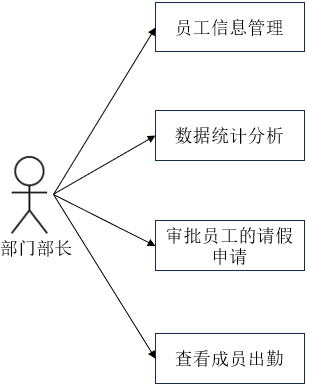


图2.2 部门部长用例图

普通员工通过系统完成日常考勤打卡，可查看个人考勤统计，修改登录密码和个人联系方式，如图2.3。

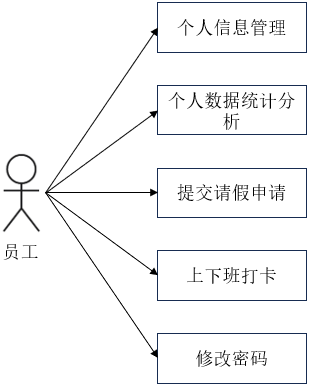


图2.3 员工用例图

## **2.4** 功能模块设计

根据之前的需求分析，本系统设计了员工模块，请假模块，登录模块，考勤模块，总共四个模块，具体功能由着四个模块分别实现以及相对应的详细功能实现。详细系统设计图如图2.4所示。

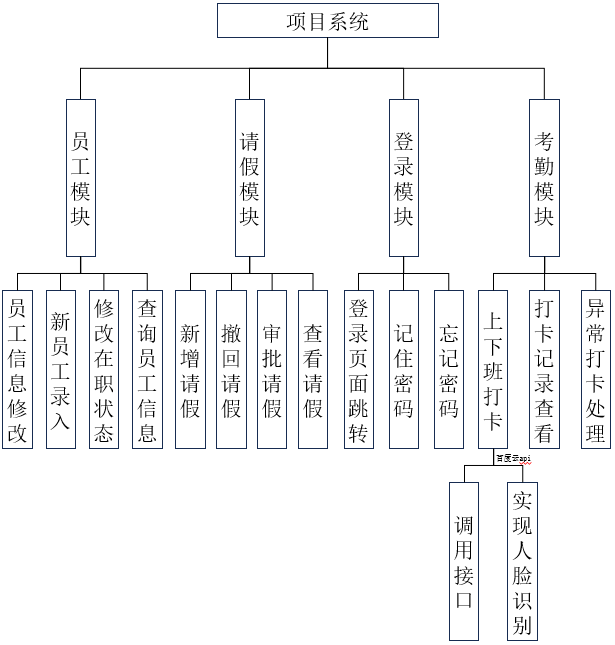


图2.4 系统模块图

1. 员工模块设计

管理员可在员工模块中对员工信息进行管理，包括添加、修改、查询及删除等操作。员工入职时，录入其姓名、性别、出生日期、联系方式、职位和部门等基本信息；信息变动时，及时更新以确保准确，如联系方式变更等。员工离职时，提供修改在职状态功能。管理员可按姓名、工号等条件查询员工信息，快速定位和查看详细资料。

1. 请假模块设计

员工在请假模块中可申请、撤回、审批请假。申请时，填写开始、结束时间和请假原因；未审批前可撤回修改。部门部长审核申请，决定批准或拒绝，并填意见。系统记录请假详情，供员工和管理人员查询，以便考勤管理和工作安排。

1. 登录模块设计

员工通过登录模块使用账号和密码登录系统，系统采用MD5+盐[34]加密确保密码安全。验证通过后，员工进入对应界面。员工按需修改密码，系统检测强度并经加密后存储。忘记密码时，通过人脸识别进行身份确认，重置后新密码同样经加密处理，保障访问权限安全。

1. 考勤模块设计

员工在打卡模块中可查询打卡记录，了解每日打卡时间和状态等信息，通过人脸识别功能进行上下班的考勤。其中人脸识别部分利用百度云api进行实现。如遇异常打卡，可申请修改。管理人员核实后修正记录，确保考勤数据准确。

# **3、**数据库的逻辑设计

根据以上需求分析，在确定了各个表主键字段的基础上，依据表与表之间相关字段之间的联系建立了各表之间的关系，对应的关系图如图3.1所示：

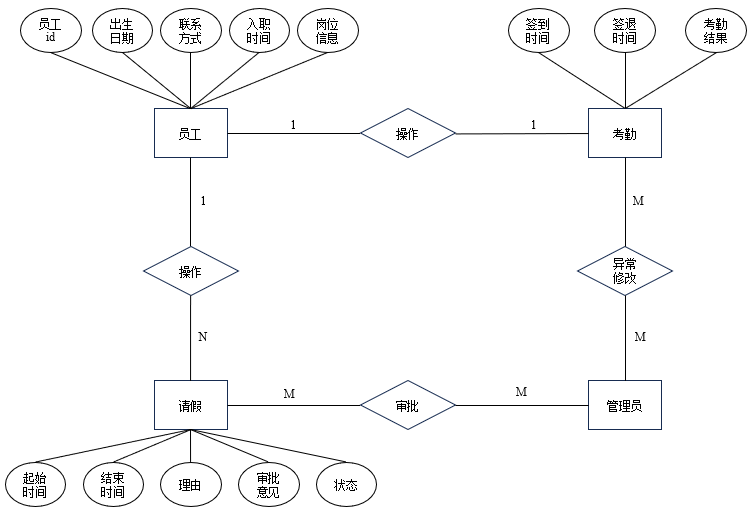


图3.1数据库逻辑图

## **3.1** 主要数据表的结构

（1）部门表（dapartment）如表3.1部门表所示：

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **数据类型** | **主键/外键** | **描述** |
| department\_id | INT | 主键 | 自增部门ID |
| department\_name | VARCHAR(255) |  | 部门名称 |

表3.1图书信息表

（2）员工部门职位关联表（emp\_dept\_pos）如表3.2员工部门职位关联表所示：

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **数据类型** | **主键/外键** | **描述** |
| id | INT | 主键 | 自增ID |
| emp\_no | VARCHAR(20) | 外键 | 员工编号 |
| department\_id | INT | 外键 | 部门ID |
| position\_id | INT | 外键 | 职位ID |

表3.2员工部门职位关联表

1. 员工详细信息表（emp\_detail）如表3.3员工详细信息表所示：

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **数据类型** | **主键/外键** | **描述** |
| id | INT | 主键 | 自增ID |
| photo | VARCHAR(255) |  | 照片路径 |
| emp\_no | VARCHAR(10) |  | 员工编号 |
| name | VARCHAR(255) |  | 姓名 |
| sex | VARCHAR(1) |  | 性别 |
| hire\_time | DATE |  | 入职时间 |
| phone | VARCHAR(255) |  | 联系电话 |
| email | VARCHAR(255) |  | 邮箱 |
| address | VARCHAR(255) |  | 住址 |
| birth | DATE |  | 出生日期 |
| status | TINYINT(1) |  | 在职状态（1/0） |
| isAdmin | INYINT(1) |  | 是否管理员（1/0） |

表3.3员工详细信息表

1. 员工登录表（emp\_login）如表3.4员工登录表所示：

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **数据类型** | **主键/外键** | **描述** |
| id | INT | 主键 | 自增ID |
| emp\_no | VARCHAR(255) |  | 员工编号 |
| password | VARCHAR(255) |  | 加密密码 |

表3.4员工登录表

1. 请假状态表（eave\_status）如表3.5请假状态表所示：

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **数据类型** | **主键/外键** | **描述** |
| status\_id | INT | 主键 | 状态ID |
| status\_name | VARCHAR(255) |  | 状态名称 |

表3.5请假状态表

1. 请假记录表（leaves）如表3.6请假记录表所示：

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **数据类型** | **主键/外键** | **描述** |
| id | BIGINT | 主键 | 自增ID |
| emp\_no | VARCHAR(255) |  | 员工编号 |
| start\_time | DATETIME |  | 请假开始时间 |
| end\_time | DATETIME |  | 请假结束时间 |
| status\_id | INT |  | 状态ID |
| reason | VARCHAR(255) |  | 请假原因 |
| review\_opinions | VARCHAR(255) |  | 审批意见 |

表3.6请假记录表

1. 职位表（positions）如表3.7职位表所示：

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **数据类型** | **主键/外键** | **描述** |
| position\_id | INT | 主键 | 自增职位ID |
| position\_name | VARCHAR(255) |  | 职位名称 |

表3.7职位表

1. 打卡记录表（punch\_records）如表3.8打卡记录表所示：

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **数据类型** | **主键/外键** | **描述** |
| id | BIGINT | 主键 | 自增ID |
| emp\_no | VARCHAR(20) |  | 员工编号 |
| punch\_date | DATE |  | 打卡日期 |
| sign\_in\_time | TIME |  | 签到时间 |
| sign\_out\_time | TIME |  | 签退时间 |
| result | INT |  | 打卡结果（1/2/3） |

表3.8打卡记录表

1. 打卡结果表（results）如表3.9打卡结果表所示：

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **数据类型** | **主键/外键** | **描述** |
| result\_id | INT | 主键 | 结果ID |
| result\_name | VARCHAR(255) |  | 结果名称 |

表3.9打卡结果表

# **4、**关键功能实现

## **4.1** 用户登录功能实现

登录功能界面的样式以及功能，是影响用户登录体验的重要因素，图 4.2 展示了用户登录界面的布局，利用简洁的布局，方便用户输入工号与密码进行登录和忘记密码等操作。



图4.2 用户登录界面

用户发送登录请求后，控制层调用LoginService中的私有方法login()方法,login()中先判断输入的用户名是否包含admin字段，若包含则调logginMapper中的getAdminPassword()方法来判断用户是否存在，若存在则查询数据库中的密码，并判断密码是否符合输入的密码，并返回相应的字段信息，再由控制层中的login方法获取到LoginService中login()方法传递过来的字段信息，根据字段信息跳转到相应的页面。

关键代码如下：

public String login(String username, String password) {  
 // 1. 非空校验  
 if(StringUtils.*isEmpty*(username)) return "工号不能为空!!";  
 if(StringUtils.*isEmpty*(password)) return "密码不能为空";  
 if (username.equals("admin")) {  
 // 从数据库中获取超级管理员的密码  
 AdminLogin adminLogin=loginMapper.getAdminPassword(username);  
 if (adminLogin.getPassword() == null) return "超级管理员未配置";  
 if(SaltMD5Util.*verifySaltPassword*(password,adminLogin.getPassword())){  
 return "超级管理员登录成功";  
 } else {  
 return "超级管理员密码错误";  
 }  
 }  
 // 2. 数据库查询

……  
 // 3. 密码验证  
 if(!SaltMD5Util.*verifySaltPassword*(password,empLogin.getPassword())) return "密码错误";  
 return "登录成功";  
}

## **4.2** 考勤打卡功能实现

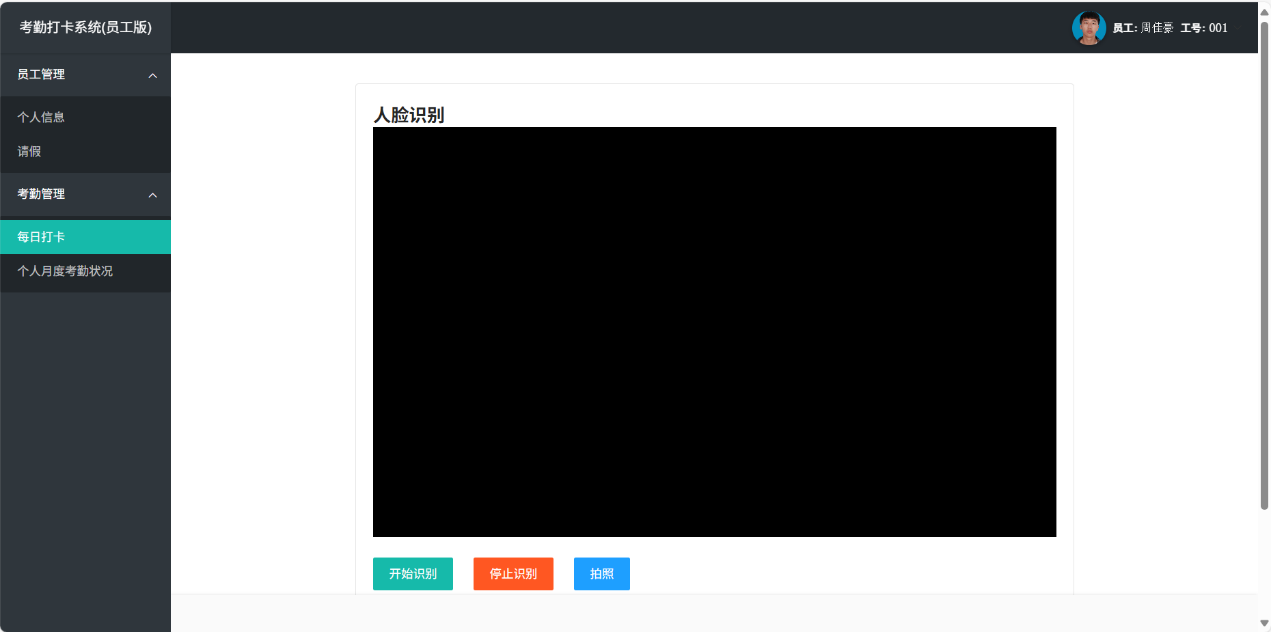
考勤打卡功能是系统的核心功能，为了使得用户更快速的进行打卡设计了如图4.3的打卡界面。

图4.3 考勤打卡页面

用户发送人脸识别请求后，控制层调用PunchService中的 recognize()方法，recognize()方法中先通过HttpSession获取当前登录用户的员工工号。然后，调用EmployeeService中的findPhotoByEmpNo()方法，从数据库中查询该员工的注册照片路径。同时，获取当前系统时间和当前日期。接着，调用BaiDuFaceRecUtil中的faceComparison()方法，将用户上传的图片数据与数据库中的照片路径进行人脸识别对比。faceComparison()方法返回一个Response对象，从中提取比对分数。再判断分数，分数大于80，则根据findPunchSignIn()方法查询数据库中数据的情况以区分上班打卡还是下班打卡。在整个过程中，如果发生异常，系统会捕获异常并记录错误日志，同时返回一个包含错误信息的响应。最后，关闭Response对象以释放资源。

关键代码如下：

Map<String, Object> responseMap = new HashMap<>();  
 String empNo = (String) session.getAttribute("username");  
 try {  
 // 获取员工照片路径和当前时间

……  
 // 人脸识别对比  
 Response faceRecResponse=BaiDuFaceRecUtil.*faceComparison*(photoPath, dataUrl);  
 double score=BaiDuFaceRecUtil.*getScoreFromResponse*(faceRecResponse);  
 if (score > 80) {  
 responseMap.put("msg", "success");  
 int attendanceResult = determineAttendanceResult(currentTime);  
 Time signInTime = punchService.findPunchSignIn(empNo);  
 if (signInTime == null) {  
 punchService.insertPunchSignIn(empNo,currentDate,currentTime);  
 }else{

punchService.updatePunchSignOut(currentTime,attendanceResult,currentDate,empNo);  
 }  
 } else {  
 responseMap.put("msg", "fail");  
 }  
 faceRecResponse.close();  
 return responseMap;  
 }

## **4.3** 请假卡片功能实现

请假页面是用户查看请假信息的唯一方式，为了更好的满足用户请假管理需求设计了如图4.4的请假查看页面。

图4.4 请假查看页面

用户发送请假记录查询请求后，控制层调用LeaveService中findLeavings()方法，该方法先获取到登陆时保存于session中的登录信息，然后，根据请求参数中的status决定查询逻辑。如果状态为空，则调用findAllLeavings()方法查询所有状态的请假记录，并通过 countAllLeavings()方法获取总记录数，以此实现分页功能。如果状态不为空，则调用 findLeavings()方法以类似的逻辑实现查询。查询结果返回后，控制层遍历请假记录列表，调用getStatusColor()和getStatusText()方法获取状态的颜色和文本描述。然后，通过 DateFormatUtil工具类中的dateFormat()方法格式化请假的开始时间和结束时间，并将每条记录生成为一个HTML字符串，拼接成最终的HTML内容。

关键代码如下：

// 生成 HTML 字符串

StringBuilder htmlBuilder = new StringBuilder();  
 for (Leave leave : list) {  
 String statusColor = getStatusColor(leave.getStatus\_name());  
 String statusText = getStatusText(leave.getStatus\_name());  
 try {  
 htmlBuilder.append("<div class=\"layui-col-md4\">")  
 .append("<div class=\"leave-card\" style=\"background-color: ")

.append(statusColor).append(";\">")  
 .append("<div class=\"layui-row\">")  
 .append("<div class=\"layui-col-md8\">")  
 .append("<p>工号: ").append(leave.getEmp\_no()).append("</p>")  
 .append("<p>")  
 .append("<span class=\"time-label\">时间: </span> ")  
 .append("<span class=\"start\_time\">")

.append(DateFormatUtil.*dateFormat*(leave.getStart\_time())).append("</span>")  
 .append("</p>")  
 ……

}

}

## **4.4** 添加员工功能实现

添加员工功能是企业进行新员工录入的主要功能，为此设计一个简介明了的页面如图4.6，以方便进行员工录入的工作。

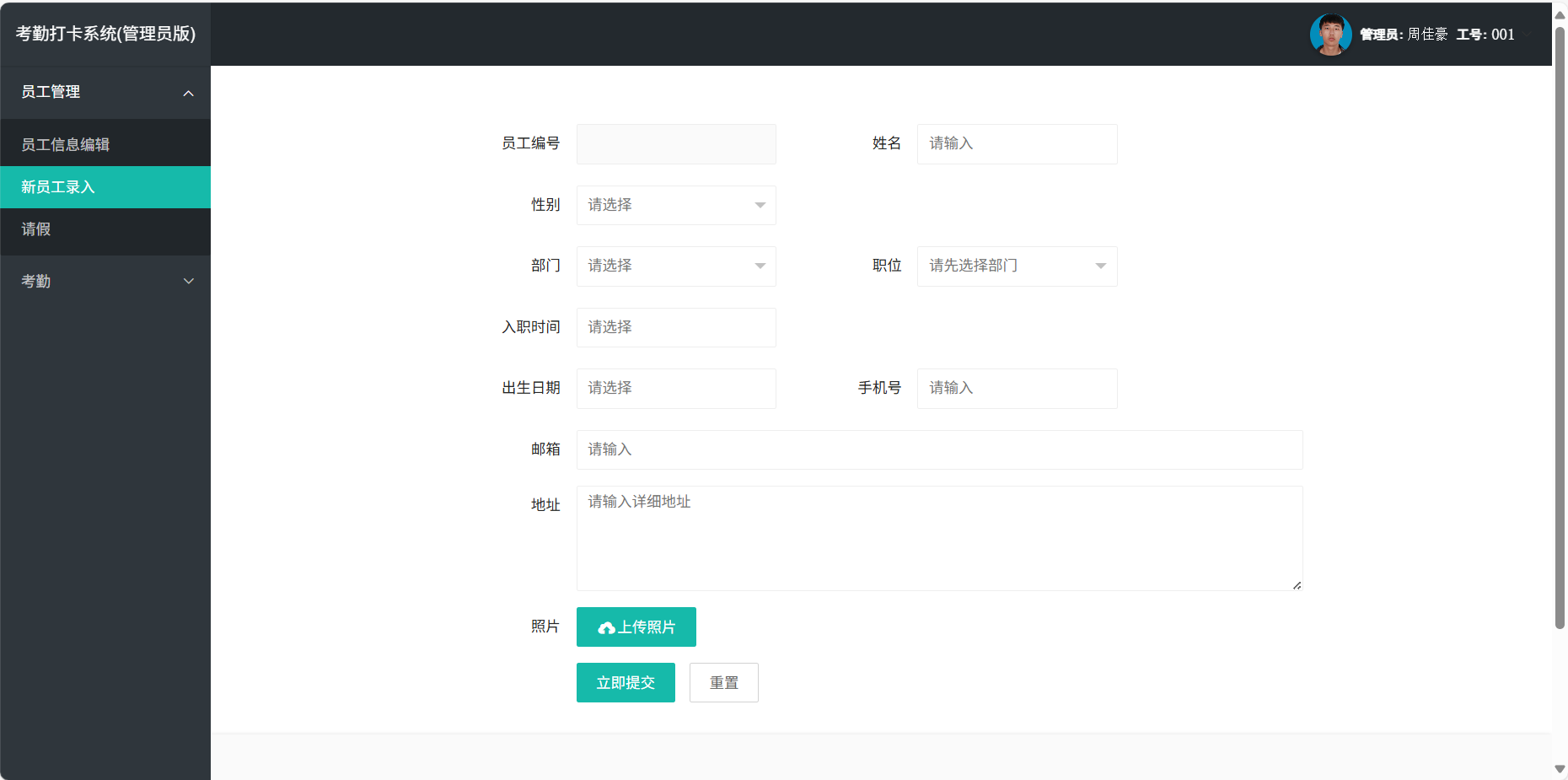


图4.6 新员工录入页面

用户发送添加员工信息的请求后，控制层调用 EmployeeService 中的 addEmp()方法，addEmp()方法中接收用户输入的员工信息， 先根据用户输入的性别值，判断性别是否正确。1为男，2为女，否则返回错误响应，提示“性别不正确”。再使用 DateFormatUtil中的dateFormat()方法进行类型转换。如果转换失败或日期为空，返回错误响应，提示对应信息。然后调用 FileUtil中的moveFile()方法将用户上传的照片移动到指定目录，并获取新的照片路径。最后，创建Employee对象，包含照片路径、员工编号、姓名、性别、入职时间、手机号、邮箱、地址、生日等信息，并调用EmployeeService中的addEmp()方法将员工信息保存到数据库，同时删除临时文件夹中的照片信息。其中新增用户是会将新用户的密码设置为工号，并用SaltMD5Util中的generateSaltPassword()方法对密码进行加密。

关键代码如下：

Employee employee = new Employee(photo, emp\_no, name, sex, hire\_time, phone, email, address, birth, true, false);  
 String msg = employeeService.addEmp(employee);  
 employeeService.addDeptPos(emp\_no, department\_id, position\_id);  
 if (msg.contains("成功")) {  
EmpLogin empLogin=new EmpLogin(emp\_no,SaltMD5Util.generateSaltPassword (emp\_no));  
 loginService.addEmpLogin(empLogin);  
 result.put("code", 0);  
 result.put("redirect", "showEntry");  
 FileUtil.removeFile(photoPath);  
 } else {  
 result.put("code", 1);  
 result.put("msg", msg);  
 }

## **4.5** 考勤数据统计功能实现

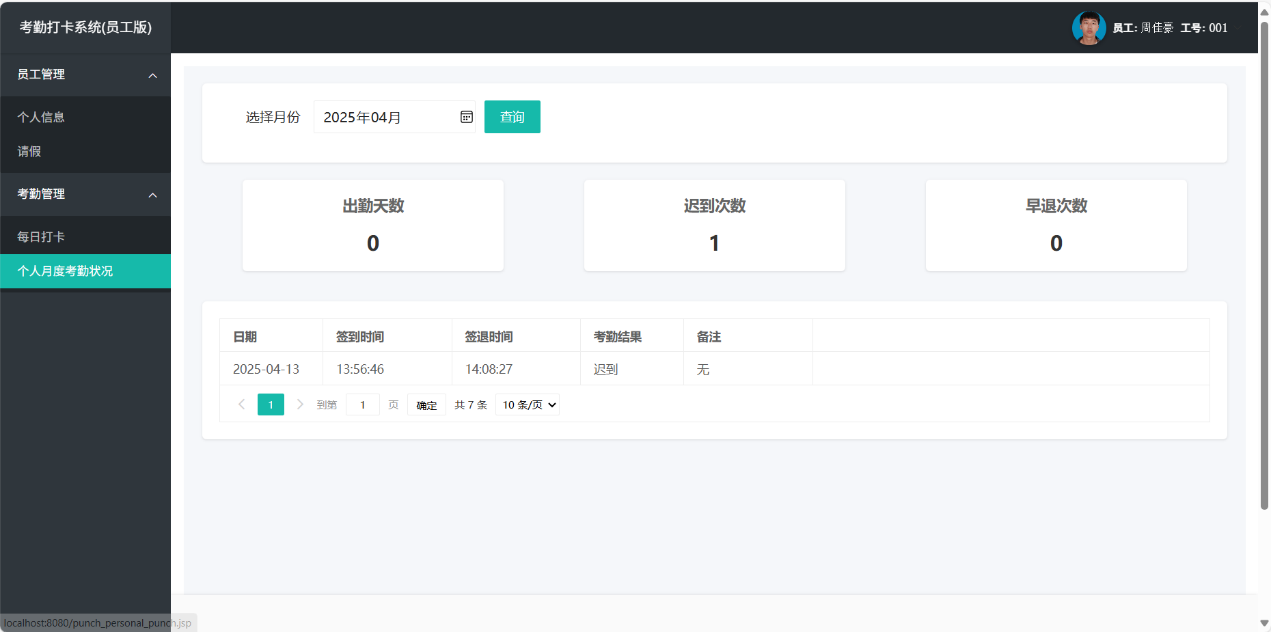
为了方便用户查看每月打卡情况更加方便和直观，为此设计了如图4.5的考勤数据统计页面。

图4.5 考勤数据统计页面

用户发送获取考勤汇总信息的请求后，控制层调用 PunchService中的getAttendanceSummary()方法。getAttendanceSummary()方法中先从 HttpSession 获取当前登录用户的员工工号，并获取当前日期。如果empNo不为空，方法将调用PunchService中的findAttendance()方法三次，分别获取正常打卡、迟到和早退的打卡记录列表。然后，通过 filterByMonth()方法筛选出对应月份的记录，并统计正常、迟到和早退的记录数量。统计结果被封装到一个map中，包含正常打卡天数、迟到次数和早退次数。然后，将summary添加到响应Map中，并设置响应码为0，表示查询成功。

关键代码如下：

Map<String, Object> map = new HashMap<>();  
 String empNo = (String) session.getAttribute("username");  
 LocalDate currentDate = LocalDate.now();  
 model.addAttribute("currentDate",currentDate);  
 try {  
 if (empNo != null) {  
 // 获取正常、迟到、早退的打卡记录  
 ……  
 // 筛选出对应月份的记录  
 ……  
 Map<String, Integer> summary = new HashMap<>();  
 summary.put("attendanceDays", normalCount);  
 summary.put("lateTimes", lateCount);  
 summary.put("earlyTimes", earlyCount);  
 map.put("data", List.of(summary));  
 map.put("code", 0);  
 map.put("msg", "查询成功");

}

# **5、**系统不足与改进方向

## **5.1** 系统存在的不足

人脸识别功能依赖于百度云API的网络连接，网络不稳定时可能影响打卡效率。在离线环境下，系统的功能受限，与本考勤系统的本地逻辑存在一定的集成复杂性。在上下班高峰期的时候，系统的响应时间略有增加，资源利用率较高，仍有进一步优化的空间。

当前系统主要面向PC端用户，移动端支持功能较少，无法满足用户随时随地进行考勤管理的需求。在页面显示方面，还可以进一步的进行有效的美化，没有实现对于不同视力情况人群的解决方法，所有的字体大小都无法修改。

考勤打卡异常情况下，没有提供在线的提交申请的接口，若出现断网等异常还是只能依赖于人工修改信息方式进行打卡的异常处理。

## **5.2** 未来改进方向

引入本地人脸识别技术，减少对网络的需求，提高打卡的稳定性和效率。在此基础上进一步优化人脸识别算法，以确保在绝大多数环境下都能稳定运行。对于人脸识别部分响应时间长问题，引入数据压缩功能，减少数据的大小，加快传输速度。

开发的移动端应用，提供与PC端相同的功能，支持用户随时随地进行考勤管理和查询。本系统暂时只支持在局域网之中的使用，后续与 移动端开发的同时也要将其发展为可以直接通过互联网进入的应用。

根据企业的具体需求，选择最适合的技术方案来进一步提升考勤系统的性能和用户体验，实现替换人脸识别的功能接口。增加弹性工作制度与异常打卡提交申请功能，允许企业根据自身需求灵活设置考勤规则，如扩展加班制度等。还要增加修改页面颜色、字体大小等功能，防止因为观察不清楚的问题导致工作效率以及员工体验下降的问题。