

## 2019 年软件工程课程设计报告

课设题目： 基于树莓派人脸识别的员工考勤系统

姓名	学号
王鑫雨 (组长)	41601435
窦雪飞	41621058
傅正麒	41622229
徐茂源	41618055
徐韬	41624213

提交时间：2019 年 6 月 21 日

<b>一、总体说明</b>	<b>1</b>
1.1 开发团队	1
1.2 开发环境和工具	1
<b>二、软件需求</b>	<b>1</b>
2.1 需求陈述	1
2.2 可行性分析	2
2.2.1 技术可行性	2
2.2.2 经济可行性	3
2.2.3 操作可行性	3
2.2.4 法律可行性	3
2.3 需求分析	4
2.3.1 功能需求	4
2.3.2 性能需求	6
2.3.3 界面需求	6
2.3.4 资源使用需求	6
2.3.5 安全保密需求	7
2.3.6 软件成本消耗与开发进度需求	7
需求协商	7
<b>三、系统设计</b>	<b>11</b>
3.1 总体设计（或体系结构设计）	11
3.2 详细设计	13
<b>四、系统实现</b>	<b>14</b>
<b>五、系统测试</b>	<b>39</b>
5.1 测试方案	39

5.2 测试结果.....	40
5.3 分析.....	57
<b>六、部署和运行.....</b>	<b>57</b>
参考文献.....	59

## 一、总体说明

### 1.1 开发团队

前端开发：徐茂源、徐韬、王鑫雨

后端开发：傅正麒、窦雪飞、王鑫雨

树莓派：王鑫雨

### 1.2 开发环境和工具

开发环境：Windows 10 , Linux

开发工具：Eclipse, IntelliJ IDEA, Adobe Dreamweaver, Android Studio

## 二、软件需求

### 2.1 需求陈述

1. 经理能够向系统中添加、删除、管理用户信息，包括工号、所属部门、初始密码、是否为部门主管，一个部门只能有一个主管，经理能够任免主管。
2. 部门主管能够为本部门的员工安排工作班次(通常以月度为单位,但不限于月度),某员工的一个工作班次是每天的上下班时间。注意，不同员工的工作班次安排可能不同，一名员工每天的工作班次安排也可能不同。
3. 系统能够以月为单位展示 (1) 某一部门所有员工的工作班次安排、(2) 某一员工的工作班次安排。
4. 部门主管可以随时通过系统调整工作班次安排。
5. 员工可以通过系统查看自己的工作班次安排。

6. 员工可以通过系统进行请假和销假，系统能够自动提醒主管进行审批。如果员工的请假申请被批准，系统能够提醒主管调整该员工在请假期间的工作班次安排。请假申请包括请假的起始日期和终止日期、请假理由和类型（事假/病假）等。

7. 员工能够通过系统进行考勤记录，即系统能够记录员工实际上下班的时间（打卡）。

8. 如果员工加班，如下班时间超过计划时间一定阈值，系统提醒员工是否要申报加班。如果员工要申请加班，可以通过系统提交加班理由，等待部门经理审批。

9. 经理能够创建全单位的临时性加班活动，独立于部门经理的工作班次安排计划。系统能够记录员工是否参与了临时性加班（需打卡）。

10. 系统能够以月为单位展示每名员工的上班情况，部门主管能够查看本部门的员工情况，经理能够查看所有员工的情况。

11. 员工能够通过系统更新自己的账户信息

## 2.2 可行性分析

### 2.2.1 技术可行性

#### 1.1 资源分析

软件、硬件资源：要实现该系统，需要一个数据库服务器存储考勤系统的相关信息，一个后端服务器运行考勤查询与管理系统。用户通过手机 app 或 web 浏览器进入考勤系统。

系统开发所需的各类人员:Android（或 IOS）开发工程师，web 前端开发工程师，后端开发工程师，运维工程师，考勤系统系统管理员（以上各类人员可兼任）。

根据现有资源情况，现有资源可以满足以上需求。

#### 1.2 风险分析：

在给定约束条件下，可以在规定时间（8 周）内设计并实现系统所需的功能和性能。

1.3 技术分析：人工智能人脸识别可以通过百度云、阿里云等平台实现得；考勤系统相关信息可用 MySQL, oracle 等数据库进行存储；服务器后端系统可使用 java, python, node js , PHP 等语言完成。该系统搭建相关高级语言的环境，在其环境下系统，并且在正确连接数据库后可以正常运行。当前的科学技术完全可以支持系统开发的全过程。

综上：针对要求的功能、性能以及实现系统的各项约束条件进行分析，从技术角度上，预设的系统可以做成一个可接受的系统。

### 2.2.2 经济可行性

系统开发费用与服务器等硬件费用均在可接受范围之内。人脸识别功能使用可以通过学生账号在云端服务器购买，降低成本。在硬件设施上，只需要一台服务器即可完成该系统的搭建。

### 2.2.3 操作可行性

1. 该软件的开发过程可由相关人员在预计开发周期内完成。
2. 用户只需下载手机 app 或 web 应用即可通过连接该考勤系统服务器完成相关功能，该系统对用户来说也是可接受的。

### 2.2.4 法律可行性

该平台是作为学生学术设计与商业无关，又因为是自主开发设计，因此不会构成侵权，在法律上是可行的。

## 2.3 需求分析

基于以上调研结果，提出软件系统的以下需求。

### 2.3.1 功能需求

#### 1. 人脸识别相关功能

- (1) 基本需求：利用人脸识别技术，完成系统打卡看模块，要求能够借助 AI 开放平台（如百度云：<http://ai.baidu.com/tech/face/faceliveness>，阿里云：<https://data.aliyun.com/product/face>，腾讯云：<https://cloud.tencent.com/product/facerecognition>），对拍摄图像，进行人工智能人脸识别，并能够正确解析拍摄内容。
- (2) 活体检测：活体检测部分分为：动作配合式活体检测和在线图片活体检测。动作配合式活体检测是通过 SDK 给出指定动作要求，用户需配合完成，通过实时检测用户眼睛，嘴巴，头部姿态的状态，来判断是否是活体。支持多种预设动作，可自定义哪些生效以及检测顺序。在线图片活体检测包括基于图片中人像的破绽（摩尔纹、成像畸形等）来判断目标对象是否为活体，可有效防止屏幕二次翻拍等作弊攻击，可使用单张或多张判断逻辑的活体检测等动作配合式活体检测。
- (3) 人像照片的存储：上传到服务器相关文件目录，注意最好是大头照，否则影响识别。

#### 2. 考勤系统(WEB 端和手机 app 版本)

该系统是面向所有用户的，根据账户权限来显示不同功能页面。

- (1) 用户登录功能：用户登录账号后才能执行考勤系统的功能。要求用户记住账号密码，新建用户使用默认密码登录。
- (2) 查看班次功能：员工可以查看自己的班次，主管可以查看本部门的所有员工工作班次。
- (3) 更新账户信息功能：用户可以修改自己的个人信息，包括星座，手机

号，毕业学校，个人简介等。

- (4) 打卡功能：对镜头刷脸，检验此人是否存在。若不存在则提示用户重新刷脸或不存在此人若此人存在，检测是否在可打卡时间内。若在可打卡时间内，提示打卡成功，系统记录打卡时间；若不在可打卡时间内，提示打卡失败
- (5) 请假功能：用户在网页中提交请假申请，系统将申请发送给主管，自动提醒主管进行审批，并根据请假结果更新工作班次。如果请假申请被批准，系统提醒主管调整该员工在请假期间的工作班次安排，如果请假申请没有被批准，则提醒用户请假失败。
- (6) 销假功能：用户请假后申请销假，系统先审核是否此员工在请假状态。如果在终止日期前销假，系统更改用户状态，并返回销假成功。如果在终止日期后销假，记录销假时间，提示用户销假失败
- (7) 调整班次功能：选择一个员工或多个员工，更改工作班次，调整工作班次安排可以是临时调整或永久调整。合法输入，返回调整成功；反之则提示用户调整失败。
- (8) 批准功能：系统将员工申请发送给主管，可以说加班或者请假申请。部门主管选择批或不批。如果批准，发送批准消息给员工，考勤记录设置更新。如果不批准，发送不批准消息给员工。
- (9) 管理员工功能：编辑员工信息，保存并更新员工信息。若合法输入，提示修改成功；反之则提示用户修改失败。
- (10) 任免功能：列出所有主管，经理选定一位或多位主管，对他们的身份进行更改，改为员工，保存并更新员工信息到数据库，并返回修改成功。
- (11) 创建临时加班功能：创建全单位的临时性加班活动，并提交。系统记录员工工作信息，发送通知给加班时间不在请假的员工。



### 2.3.2 性能需求

**1.前端的性能需求：**网页需要有自适应，以免不同打开方式影响系统功能使用。将复杂多样的功能，通过目录等引导，在最短时间内能获取所需数据成为一个值得探讨的问题。

**2.后端系统性能：**要求该系统的并发能力要强，要求对峰值期间的大量用户查询或连接请求能够处理。另外，要求系统对用户的保存信息，上传文件等请求在较短时间内处理并返回，以免影响用户体验。

**3.服务器带宽：**服务器带宽要合理分配，尽量避免因为机器网络带宽不够而出现请求出错的情况。

**4.手机 app 性能需求：**手机 app 要避免出现卡顿感。

#### 5. 时间特性需求

- 1) 手机 app 登录界面的响应时间不超过 3s，相关功能响应时间不超过 5s。
- 2) 系统故障发生故障时，要求在一天（24 小时）内修复（与系统管理人员是否及时，以及程序员解决错误的专业能力有关）。
- 3) 系统采用 JDBC 连接数据库，保证较快的响应时间和更新处理时间，采用 JSP Servlet 技术，以满足用户对数据的转换和传送时间要。

### 2.3.3 界面需求

针对 android 版本需要遵循 google Material Design 设计规范，开发 Material Design 风格的 app。

### 2.3.4 资源使用需求

软件运行所需服务器：操作系统建议 Google Chrome，火狐等专业浏览器；

一台数据库服务器，数据库建议使用 MySQL，或 Microsoft SQL Server 或 Oracle。

### 2.3.5 安全保密需求

用户密码不能明文存储在服务器，需要 hash 加密后存储。

系统后台管理推荐使用 https 协议而非 http 协议。

其他安全需求：增强系统安全性能，防范各类网络攻击(如用户提交请求后，需要过进行过滤，防范 xss 跨站脚本攻击或 SQL 注入)

### 2.3.6 软件成本消耗与开发进度需求

该软件预计开发时间为 8 周。

### 2.3.7 数据需求

数据需求即在完成人脸识别签到过程中所需要的数据。数据需求应包括：数据采集范围及方法；静态数据；动态输入数据；动态输出数据。

数据采集范围及方法：用户上传。方法：在个人界面引导用户上传。

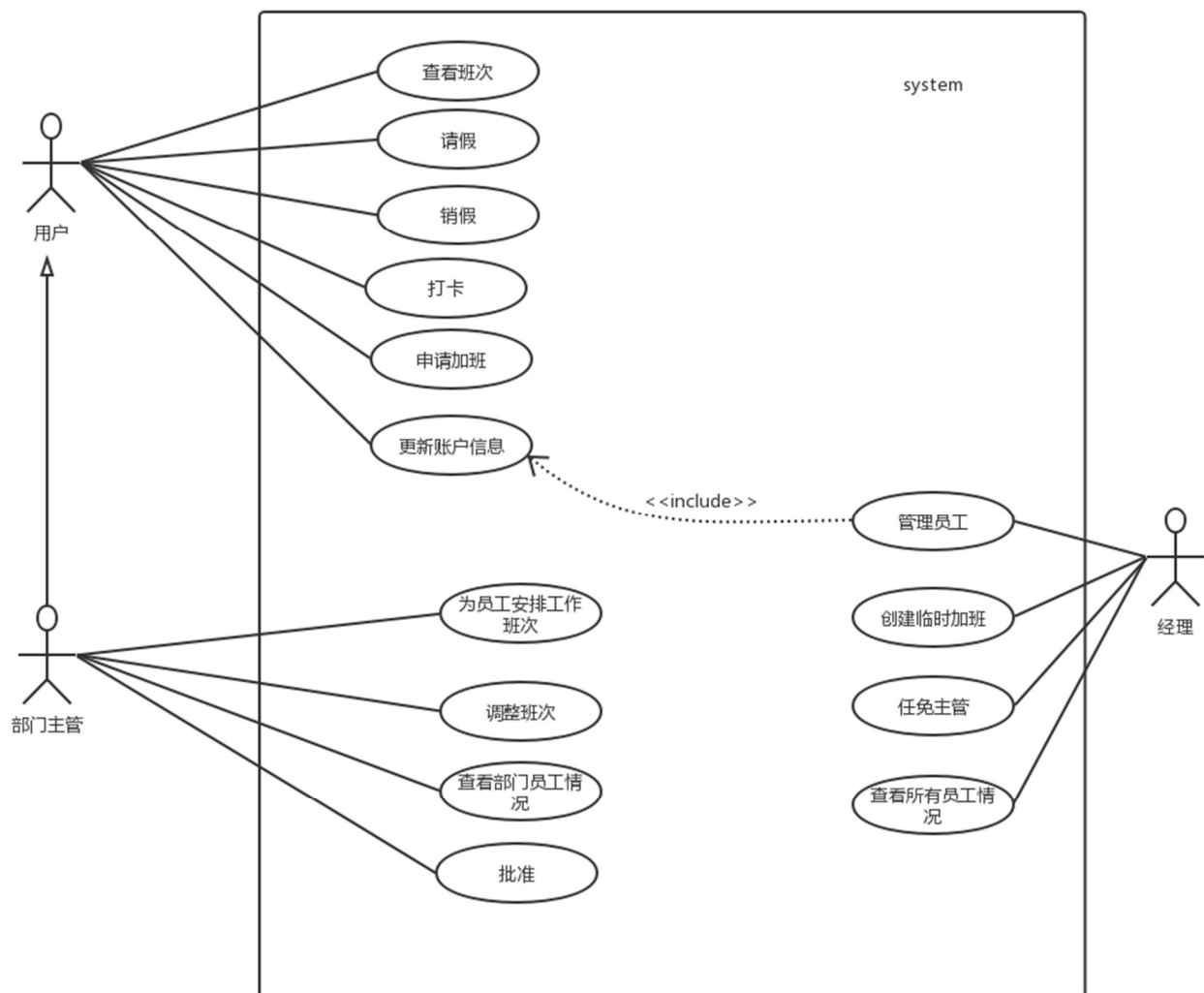
静态数据:用户账号，密码，个人照片，个人信息，考勤记录等。

动态输入输出数据:请假申请，销假申请。

## 需求协商

考虑到时间与精力不足的问题，优先完成需求中基本功能，没有实现批量上传，所设计的签到时刻表模块还没有实现交互。

## 用况模型



用况：查看班次

参与者：用户

1. [系统]：在网页中显示员工班次
2. [用户]：查看自己的班次
3. [系统]：在员工上班前提醒员工的上下班时间

用况：打卡

参与者：用户

1. [用户]：对镜头刷脸。
2. [系统]：检验此人是否存在，若不存在则提示用户重新刷脸或不存在此人。
3. [系统]：若此人存在，检测是否在可打卡时间内。

4. [系统]: 若在可打卡时间内, 提示打卡成功, 系统记录打卡时间
5. [系统]: 若不在可打卡时间内, 提示打卡失败

用况: 请假

参与者: 用户

1. [用户]: 在网页中发送请假申请
2. [系统]: 将申请发送给主管
3. [系统]: 系统自动提醒主管进行审批
4. [系统]: 如果请假申请被批准, 系统提醒主管调整该员工在请假期间的工作班次安排
5. [系统]: 如果请假申请没有被批准, 则提醒用户请假失败

用况: 销假

参与者: 用户

1. [用户]: 用户请假后申请销假
2. [系统]: 系统先审核是否此员工在请假状态
3. [系统]: 如果没有在请假状态, 则提示用户输入错误
4. [系统]: 如果在请假状态, 判断是否在终止日期前销假
5. [系统]: 如果在终止日期前销假, 系统更改用户状态, 并返回销假成功
6. [系统]: 如果在终止日期后销假, 记录销假时间, 提示用户销假失败

用况: 申请加班

参与者: 用户

1. [系统]: 如下班时间超过计划时间一定阈值, 系统提醒员工是否要申报加班
2. [用户]: 通过系统提交加班理由, 等待主管审批。
3. [系统]: 将加班申请发送给主管
4. [系统]: 如果主管同意, 提示申请成功, 并记录加班时间
5. [系统]: 如果主管不同意, 提示申请失败

用况: 更新账户信息

参与者: 用户

1. [用户]: 打开系统, 编辑个人信息, 并提交
2. [系统]: 保存更新数据
3. [系统]: 提示用户编辑成功

用况: 为员工安排班次

参与者: 部门主管

1. [部门主管]: 调整本部门员工工作班次安排。可选择批量设置或单独设置等。
2. [系统]: 更新员工安排班次。

用况：查看部门员工情况

参与者：部门主管

1. [系统]：统计并显示所有部门员工的情况，包括工作状态，打卡记录、请销假记录、加班记录等。
2. [部门主管]：查看本部门员工情况。

用况：调整班次

参与者：部门主管

1. [部门主管]：选择一个员工或多个员工，更改工作班次，调整工作班次安排可以是临时调整或永久调整。
2. [系统]：保存并更新员工工作班次，返回调整成功。

用况：批准

参与者：部门主管

1. [系统]：将员工申请发送给主管，可以说加班或者请假申请
2. [部门主管]：选择批或不批
3. [系统]：如果批准，发送批准消息给员工，系统更新考勤记录设置
3. [系统]：如果不批准，发送不批准消息给员工

用况：管理员工

参与者：经理

1. [经理]：导入文件更新员工信息
2. [系统]：保存并更新员工信息，并返回修改成功

用况：查看所有员工情况

参与者：经理

1. [系统]：统计并显示所有员工的情况，包括工作状态，打卡记录、请销假记录、加班记录等。
2. [部门主管]：查看所有员工情况。

用况：任免主管

参与者：经理

1. [系统]：列出所有主管
2. [经理]：选定一位或多位主管，对他们的身份进行更改，改为员工
3. [系统]：保存并更新员工信息，并返回修改成功

用况：创建临时加班

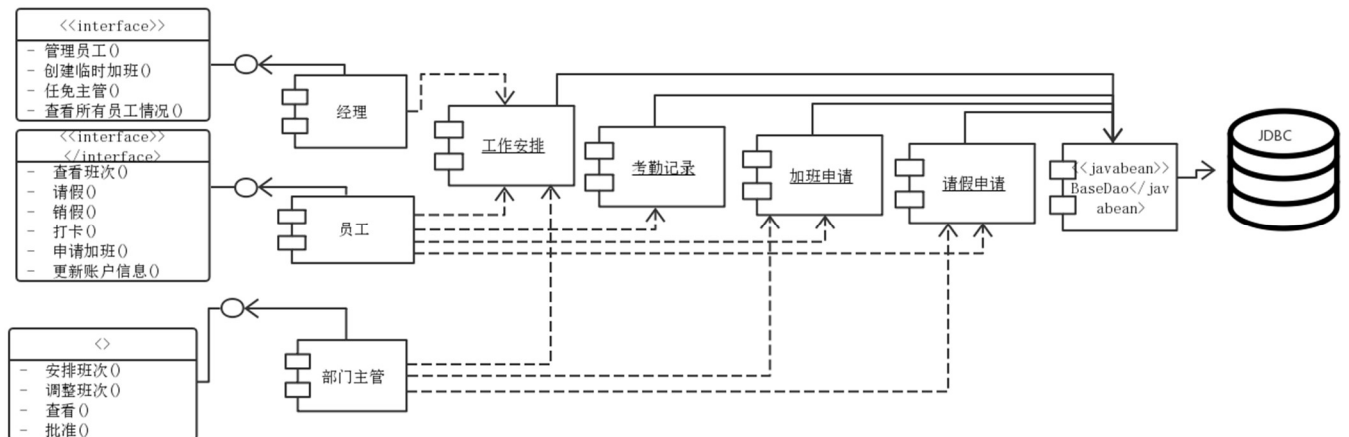
参与者：经理

1. [经理]：创建全单位的临时性加班活动，并提交
2. [系统]：系统记录员工工作信息，发送通知给加班时间不在请假的员工。

### 三、系统设计

#### 3.1 总体设计 (或体系结构设计)

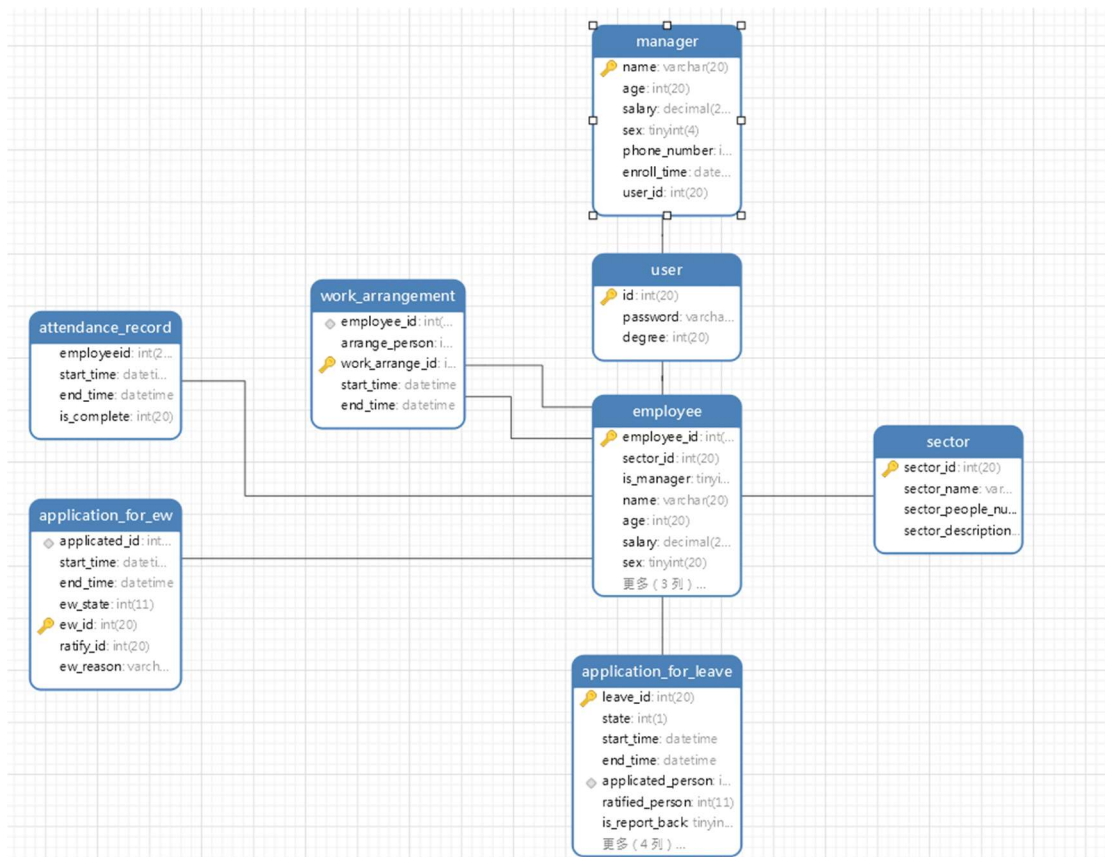
系统构件图：



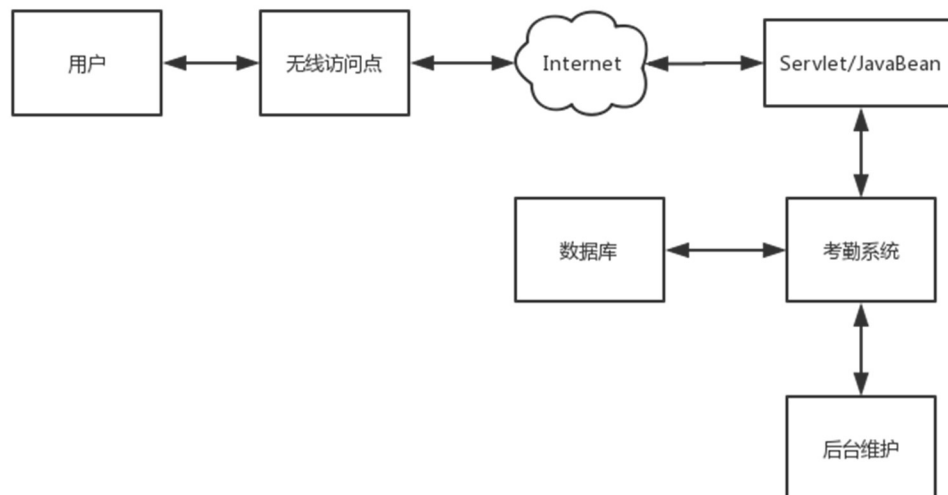
构件图说明：

- 1) 虚线表示依赖关系，部门主管依赖工作安排，加班申请，请假申请。  
员工依赖工作安排，考勤记录，加班申请，请假申请。经理依赖工作安排。
- 2) 这些类属于 BaseDao，BaseDao 与 JDBC 交互。
- 3) 调用功能后，通过和数据库的交互实现。

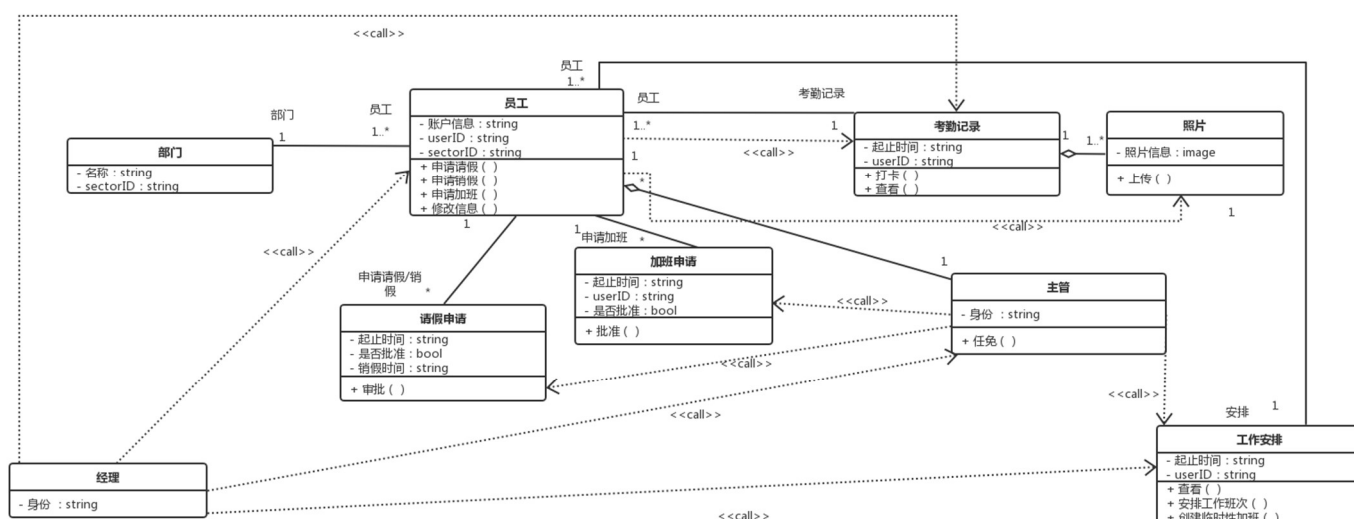
ER 图设计：



系统网络拓扑图：



### 3.2 详细设计



设计说明:

- 1) 员工打卡时会区分, 上班打卡, 下班打卡, 临时加班打卡, 计入考勤记录。
- 2) 主管有员工的行为, 因此他可继承 “员工” 类。
- 3) 用户对工作安排的操作权限有所不同, 所有人能查看工作安排, 但员工只能查看自己的工作安排。
- 4) 针对 “部门” 类, 一个部门有一个主管和多个员工。对于一个员工而言, 只能是一个部门的员工。主部门主管能够查看本部门的员工情况, 经理能够查看所有员工的情况。
- 5) “请假申请” 类里有销假时间, 通过用销假具体时间和请假规定时间做对比, 来判定是否按时销假。
- 6) 只有部门主管才能调用审批和批准功能。
- 7) 一个员工可以发送多个加班申请、请假申请, 一个加班申请、请假申请只



对应一个员工。

## 四、系统实现

本项目是前后端分离项目，打卡部分借助树莓派完成。

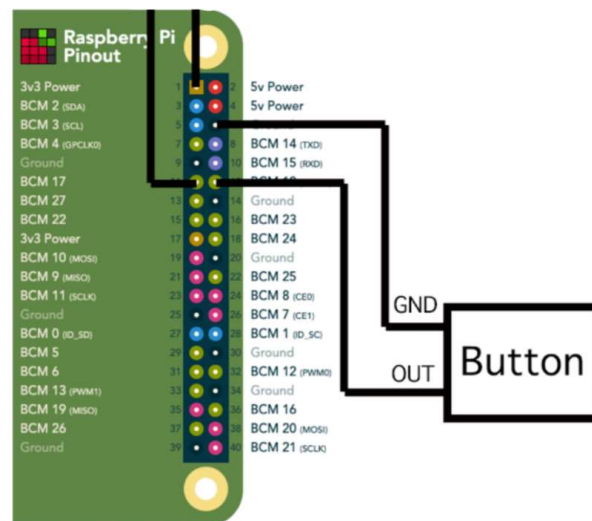
### 一、树莓派人脸识别部分：

使用原因：

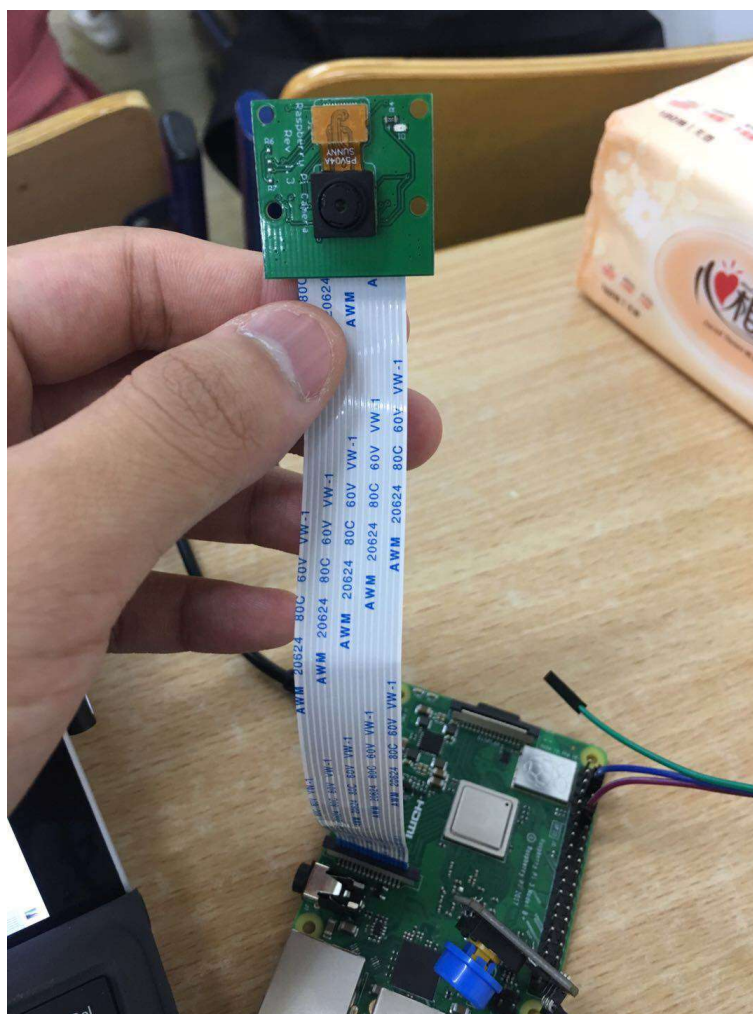
利用树莓派进行人脸识别打卡更符合实际应用场景。我们的想象的场景为，将树莓派安装在门禁上，打卡识别成功后则开放门禁给予通过，若失败则不开放。这种使用方式可以解决很多违法操作，比如员工打开后私自离开而系统无法对此做出有效的防范等。

技术分析：

树莓派是一款基于 ARM 的微型电脑主板，只有信用卡的大小但却具有电脑的所有基本功能。我们使用的树莓派的操作系统为 Linux 系统。树莓派本身不带有摄像头以及按键按钮，故我们购买了摄像头与按键，将二者与树莓派相联。这样树莓派就拥有了按下按钮拍照的硬件基础！是不是很炫酷！连接图如下：



图表 1 按键连接



图表 2 摄像头连接

硬件连接完成后，进行 Python 代码编写，从而完成：按下按钮，树莓派接收到信号后利用摄像头进行拍照，将拍出的照片通过网络发送给服务器上的打卡系统，打卡系统识别图片，并将识别结果返回给树莓派，树莓派根据返回结果来给出不同的反应，分别有三种反应，若打卡成功，树莓派播放米津玄师的 lemon！提醒员工他已打卡成功。若系统未能识别出此照片，树莓派喊出"No Face!"，提醒员工查无此脸。若有人尝试用手机照片进行打卡欺骗系统，我们的系统拥有活体检测功能，能够判断出是否为活体，树莓派则会喊出"Don't lie me!"。

源代码分析：

#导入所需包

```
#!/usr/bin/env python3
# encoding: utf-8
from picamera import PiCamera
import requests
import os
import sys
import re
import commands
import RPi.GPIO as GPIO
import time
import pygame
import pyttsx3
import json
```

#主函数

```
if __name__ == '__main__':
    setup()
    try:
        loop()
    except KeyboardInterrupt:
        destroy()
```

#loop 函数，检测按钮事件，说事件发生则调用 switchLed 函数，该函数调用图片识别函数 identifyPi

```
def loop():
    GPIO.add_event_detect(buttonPin,GPIO.FALLING,callback=switchLed)
    while(True):
        time.sleep(0.01)

def destroy():
```

```
def switchLed(ev = None):
    if (GPIO.input(buttonPin) == GPIO.LOW):
        identifyPic()
        while (GPIO.input(buttonPin) == GPIO.LOW):
            time.sleep(0.03)

def loop():
    GPIO.setup(buttonPin,GPIO.IN,GPIO.PULL_UP)
```

#图片识别函数，调用系统命令，使用摄像头进行拍照，将拍摄出的照片使用 POST 请求发送给打卡系统，根据系统的返回值，使树莓派做出不同的反应

```
def identifyPic():
    commands.getoutput("raspistill -w 800 -h 600 -o image.jpg")
    url="http://39.105.38.34:8080/api/v1/clock/attendance"
    files={'image':('test.jpg',open('image.jpg','rb'),'image/jpg',{})}
    res=requests.request("POST",url,data={"type":"1"},files=files)
    print(res.text)
    print(json.loads(res.text)['state'])
    if(json.loads(res.text)['state']==0):
        print(1)
        #saysuccess()
        music()
    elif(json.loads(res.text)['state']==1):
        sayfail()
    else:
        saylie()
```

## 二、前端部分

前端采用的是传统的 H5 技术，并结合了目前一些流行的前端框架和组件，比如 bootstrap(整体页面布局，自适应)，jQuery(UI 设计，Ajax 等)，vue(数据驱动)，element(饿了么前端，主要使用日历组件)等，基本原理为通过 Ajax 调用后天 restful 风格 API 交换数据，再通过 vue 用数据驱动整个页面。前端工程由基本的 HTML,js,css 文件和图片，字体等资源构成。

在根目录的 webhtml 文件夹中分了四个文件夹，分别是 common, employee,

manager, master。顾名思义他们分别存放了公用页面，员工页面，经历页面，主管页面。

在根目录的 js 和 CSS 文件夹中分别存放了公用 js 代码和公用样式表。

在根目录的 img 文件夹存了公用图片资源。

根目录 templates 文件夹保存的是前端的模板文件，可以用于快速创建新页面。

关键逻辑：通过 Ajax 调用 restful API 和后端交换数据，用 vue 来驱动整个网页，由于整个工程页面逻辑都相似，在此列举三个

例一：登录功能的实现。根目录/index.html

```
<script type="text/javascript">
$(document).ready(function(){
    console.log("begining \n")
$.cookie('the_cookie1231', 'the_value131231', { expires: 10, path: '/js' });
    console.log(document.cookie)
    $("#logBtn").click(function(){
        $("#myDiv").html(getjson());
        if($('#username').val()!=" " && $('#password').val()!=""){
            $.ajax({
                type:"POST",
                url:"http://39.105.38.34:8080/api/v1/login/login",
                contentType:"application/json",
                data:JSON.stringify(getjson()),
                dataType:'json',
                xhrFields: {
                    withCredentials: true
                },
                crossDomain: true,
```

```

        success:function(data){
            getUserInfo(data.employeeId)
            if(data.degree=='2'){
                //主管
                window.location= ("webhtml/master/master.html");
            }
            else if(data.degree=='1'){
                //员工
                window.location=("webhtml/employee/employee.html");
            }
            else if(data.degree=='55'){
                //经理
                window.location=("webhtml/manager/manager.html");
            }
            else{
                alert("请检查用户密码!!! ");
            }
        },
        error:function(){
            alert("请检查用户密码!!! ");
        }

    });

    }

    else{
        $("#myDiv").html("请输入账号密码! ");
    }

});

function getjson(){
    var json = {
        id:$("#username").val(),
        password:$("#password").val()
    }
}

```

```

        return json;
    }
    function getUserInfo(UID){
        var that=this;
        jQuery.ajax({
            type: "GET",
            url: "http://39.105.38.34:8080/api/v1/employees/" + UID,
            contentType: "application/json",
            xhrFields: {
                withCredentials: true
            },
            async:false,
            crossDomain: true,
            success: function (data) {
                var userArray = JSON.parse(data)
                ///[{"sectorId":127,"enrollTime":"1970-01-
01","phoneNumber":1234567,"manager":true,"sex":true,"name":"张和",
"employeeId":5,"salary":3000,"userId":1003,"age":26}]
                if (typeof(Storage)!=="undefined")
                {
                    sessionStorage.userInfo=JSON.stringify(userArray[0]);
                }
                else
                {
                    alert("不支持 Storage,即将跳转到登录页");
                    window.location= ("../index.html");
                }
            },
            error: function (data) {
                alert("网络失败,即将跳转到登录页");
                //window.location= ("../index.html");
            }
        })
    }
}
</script>

```

可以看到，按钮 logBtn 绑定了函数 getjson()，一旦按钮被点击事件发生就会调用 getjson 函数。getjson 函数会先获取当前输入框的值，然后做一个判断，如果有空就会弹出为空提示。完成输入判断后，发起 Ajax 请求，将返回值存入会话，并根据返回值跳转页面，后面的页面就可以通过读取会话中的值来进行身份判断，所有页面被串在一起。因为工程简单，没有采用 cookies 的方式实现登录，如果考虑到安全性，改用 cookies 即可。

例二：工作班次安排页面的实现 根目录/webhtml/master/ workshiftr.html

```
<script>
    $(document).ready(function(){
        $('i-i-checks').iCheck({
            checkboxClass: 'icheckbox_square-green',
            radioClass: 'iradio_square-green'
        });
        $('#calendar').fullCalendar({
            header: {
                left: 'prev,next today',    //上一页、下一页、今天
                center: 'title',            //居中
                right: 'month,agendaWeek,agendaDay'    //右边：显示哪些视图
            },
            editable: true,
            droppable: true,
            drop: function() {
                if ($('#drop-remove').is(':checked')) {
                    $(this).remove();
                }
            },
            eventDrop: function(event) {
                if (!confirm("is this okay?")) {
```



```

        event.revert();
    }
    else {
        console.dir(event)
        ///employeeId,workShiftId,begin,end
        console.log("event.start_d",event.start._d)
        console.log(" event.end_d ",event.end._d)
        var d = new Date(event.start._d)
        var f = new Date(event.start._i)
        var startTime = d.getFullYear() + '-' + (d.getMonth() + 1) + '-' +
d.getDate() + ' ' + f.getHours() + ':' + f.getMinutes() + ':' + f.getSeconds();
        d = new Date(event.end._d)
        f = new Date(event.end._i)
        var endTime = d.getFullYear() + '-' + (d.getMonth() + 1) + '-' +
d.getDate() + ' ' + f.getHours() + ':' + f.getMinutes() + ':' + f.getSeconds();
        myvue.updateWorkShift(event.title,event.id,startTime,endTime)
    }
},
    eventResize: function(event) {
        alert(event.title + " end is now " + event.end.toISOString());

        if (!confirm("is this okay?")) {
            event.revert();
        }
    },

    eventClick: function(event) {
        if ($('#drop-remove').is(':checked')) {
            if (confirm("Are you sure about this change?")) {
                myvue.deleteWorkShift(event.id);
                //UiUpdate;
                //window.location.reload()
            }
        }
    },

    events:myvue.events

```

```

    });

});

//Init calendar

</script>
<script type="text/JavaScript">
var myvue = new Vue({
    el: '#app',
    data: {
        value1: "",
        events: [],
        workshifts: [],
        workshift: [],
        staffs: [],
        options: [],
        userInfo: []
    },
    methods: {
        ///新增一条工作班次(可用)
        newToWorkShift:function(employeeId){
            console.log("ajax beging \n")
            var that = this;
            console.log("value1 is "+that.value1+" value2 is ",that.value2);
            var jsono={};
            jsono["startTime"]=that.value1[0];
            jsono["endTime"]=that.value1[1];
            jsono["arrangePerson"]="1";
            jsono["employeeId"]= $("#staffSelc option:selected").attr("id");//
            jQuery.ajax({
                type:"POST",
                url:"http://39.105.38.34:8080/api/v1/workA/employees/2",

```

```

        contentType:"application/json",
        // xhrFields: {
        //     withCredentials: true
        // },
        // crossDomain:true,
        dataType:"json",
        data:JSON.stringify(jsono),
        success:function(data){
            console.log("success");
            //UiUpdate;
            window.location.reload()
        },
        error:function(data){
            console.log("fail");
        }
    });

},
///从数据库拉取某员工的工作班次(可用)
addEvent:function(employeeId){
    var that = this;
    jQuery.ajax({
type:"GET",
url:"http://39.105.38.34:8080/api/v1/workA/employees/"+employeeId,
contentType:"application/json",
xhrFields: {
    withCredentials: true
},
crossDomain:true,
async:false,
success:function(data){
    console.log("success");
    that.workshifts=JSON.parse(data);
    console.log(" workshifts is",that.workshifts)
}
    });
}

```

```

for(var i=0;i<that.workshifts.length;i++){
    var event={allDay: false}
    event["title"]=that.workshifts[i].employeeId;
    event["start"]=that.workshifts[i].startTime;
    event["end"]=that.workshifts[i].endTime;

    event["id"]=that.workshifts[i].workArrangedId;
    that.events.push(event)
}

},
error:function(data){
    console.log("fail");

}
});

},
///删除某条工作安排（可用）
deleteWorkShift:function(workShiftId){
    jQuery.ajax({
        type:"GET",
        url:"http://39.105.38.34:8080/api/v1/workA/delete/"+workShiftId,
        contentType:"application/json",
        xhrFields: {
            withCredentials: true
        },
        crossDomain: true,
        success:function(data){
            console.log("ajax succ");
        },
        error:function(data){
            console.log("fail");
        }
    });
}

```

```

    }
});

},

///更新某条工作安排(可用)
updateWorkShift:function(employeeId,workShiftId,begin,end){
    var jsono={};
    jsono["startTime"]=begin;
    jsono["endTime"]=end;
    jsono["arrangePerson"]="1";
    jsono["employeeId"]=employeeId;
    jsono["workArrangedId"]=workShiftId;
    jQuery.ajax({
type:"PUT",
url:"http://39.105.38.34:8080/api/v1/workA/"+1,
contentType:"application/json",
xhrFields: {
        withCredentials: true
    },
dataType:"json",
data:JSON.stringify(jsono),
crossDomain: true,
success:function(data){
        console.log("succ");
    },
error:function(data){
        console.log("fail");
    }
});
},

///查询某部门所有员工
getAllStaffByDepartId:function(departmentId){
    var that = this;
    jQuery.ajax({
        type:"GET",

```

```

url:"http://39.105.38.34:8080/api/v1/employees/sectors/"+departmentId,
    contentType:"application/json",
    xhrFields: {
        withCredentials: true
    },
    crossDomain: true,
    async:false,
    success:function(data){
        that.staffs=JSON.parse(data);
    },
    error:function(data){
        console.log("fail");
    }
})
},
///init options

},
created:function(){
    var that =this
    this.userInfo = JSON.parse(sessionStorage.userInfo)
    this.getAllStaffByDepartId(this.userInfo.sectorId)
    for(var i=0;i<this.staffs.length;i++){
        this.addEvent(this.staffs[i].employeeId);
    }
}
})
</script>

```

工作班次页面先用 bootstrap 完成布局，然后在 vue 中的 methods 钩子中，将本页会用到的后端 API 实现为具体函数，可以看见有如下函数，均带注释。最后在 vue 的 created 钩子中按照逻辑调用即可。

本页的逻辑为：先从会话中获得用户登录数据，然后调用获取某部门所有员工函数(把当前登录者的部门 id 传入) 最后调用添加日历事件函数，更新日历的显示。只要保证数据的正确性，vue 会自动的帮我们驱动整个页面正常显示。

例三：提醒功能的实现 根目录/webhtml/master/ master.html

由于服务器不能主动和浏览器通讯，因此提醒消息需要浏览器主动去查询，因此我们创建了一个函数用于发起 Ajax 请求获取提醒，通过 setTimeout，每隔 5 秒发起一次该请求，不断的查询。若返回新提醒就调用 toast 函数显示这个提醒。

```
<script type="text/javascript">
var myvue = new Vue({
  data: {
    notices:[],
    userInfo:[]
  },
  el: '#wrapper',
  methods:{
    getNotice:function(){
      var that = this;
      jQuery.ajax({
        type:"GET",
        url:"http://39.105.38.34:8080/api/v1/alert/find/"+that.userInfo.employeeld,
        contentType:"application/json",
        xhrFields: {
          withCredentials: true
        },
        crossDomain:true,
        success:function(data){
          that.notices = JSON.parse(data)
          for(var i=0 ;i<that.notices.length;i++){
```

```

        if(that.notices[i].state == 0){
            toastr.success('你有一个带审批的加班申请')
        }
        if(that.notices[i].state == 1){
            toastr.success('你有一个带审批的请假申请')
        }
    }
},
error:function(data){
    console.log("fail");
}
});
    setTimeout(this.getNotice, 5000);
}
},
created: function () {
    if(sessionStorage.getItem('userInfo')== null)
        alert("请先登录啊")
    else {
        this.userInfo = JSON.parse(sessionStorage.userInfo)
        this.getNotice();
    }
},
});
</script>

```

### 三、后端部分

#### 1、整体介绍：

本次开发后端采用 java 语言编写完成，主要框架采用当前 Java 开发中一种主流的 Spring Boot 框架，我们主要将后端分为三层进行代码编写，上层为 Controller 层，用于与前端进行数据交换，确定访问路径，主要存放路径代码，并调用中层接口。中层为 Service 层，用



于实现业务逻辑，为上层提供接口，并调用下层代码，实现业务逻辑的代码存放于此层。下层为 DAO 层，用于与数据库的交互，sql 语句均存放于此层，此层只是实现单一的增删改查操作，无其他操作，最大程度实现高耦合低内聚，方便代码的重构与调用。

与数据库的连接没有采用 mybaitis 框架，而是采用了 JDBCTemplate，对数据库进行了基本的底层封装，抛弃了 mybaitis 框架的复杂的 xml 文件配置。该技术秩序建好简单的 RowMapper 映射类就好，之后不用每次进行复写代码，只需调用类就好。同时本次开发，后端本着不重复代码，不复写代码原则，尽最大努力重构了 3 次代码，使最终代码条理清晰，方便后期维护，保证了一定的代码质量。同时命名规范也符合，没有乱起名字的现象出现，保证了代码一定程度上易读性。

本次后端开发采用数据库为 MySQL 数据库，这个没有什么原因，只是因为之前一直使用，习惯了用法，降低学习成本，减少开发时间，所以采用该数据库。同时本次后端进行人脸识别模块构建时，调用了百度的人脸识别接口，并采用了百度的人脸识别库，同时还进行了活体检测，通过上传图象，百度返回人脸对比度来实现员工打卡上班功能。

## 2、关键代码逻辑：

### ①登陆：

```
@RequestMapping(value="/login",method=RequestMethod.POST)

    public String login(@RequestBody User user ) {

        HttpServletRequest request =
        ((ServletRequestAttributes)RequestContextHolder.getRequestAttributes()).getRequest();

        HttpSession session = request.getSession();
```

```

        JSONObject ans=new JSONObject();
        if(userDao.searchUserByIdAndPasswd(user).isEmpty())                return
ans.toString();
        else {
            User loginuser = userDao.searchUserByIdAndPasswd(user).get(0);
            session.setAttribute("user", loginuser)
            if(employeeDao.findEmployeeById(loginuser.getId()).isEmpty()) {
                Manager                manager                =
managerDao.findManagerById(loginuser.getId()).get(0);
                session.setAttribute("manager", manager);
                ans.put("degree", 55);
                ans.put("employeeId", 9999);
            }else {
                Employee                employee                =
employeeDao.findEmployeeById(loginuser.getId()).get(0);
                if(employee.isManager()!=false) {
                    session.setAttribute("master", employee);
                    ans.put("employeeId", employee.getEmployeeId());
                    ans.put("degree", 2);
                }else{
                    session.setAttribute("employee", employee);
                    ans.put("employeeId", employee.getEmployeeId());
                    ans.put("degree", 1);
                }
            }
        }
        return ans.toString();
    }
}

```

登陆时首先通过 userService, 从用户表中查询时候有对应的 userId 和 password。如果存在对应的用户, 则根据用户的 id 去对应的表里面获取身份。然后, 新建一个 session, 将员工 id 和身份信息存放在 session 中, 方便前端的获取。由于登陆和返回员工 id 的功

能，使得前端对于不一样的人、不同身份的人展示不一样的界面。

## ②销假

```
@RequestMapping(value="/cancel/{leaveId}",method=RequestMethod.GET)
public String CancelLeave(@PathVariable int leaveId){
    return applicationForLeaveService.CancelLeave(leaveId);
}
```

员工具有销假的功能，当员工需要销假时，点击对应的按钮，前端会向后端发送对应假期安排的 id，然后后端通过更改数据库进行销假。

## ③审批功能

```
@RequestMapping(value="/ratify",method=RequestMethod.POST)
public String RatifyLeave(@RequestBody ApplicationForLeave applicationForLeave){
    String s=applicationForLeaveService.RatifyLeave(applicationForLeave);
    return s;
}
```

主管具有对员工的加班安排和请假安排进行审批的功能。是否同意在数据库中是一个字段，前端通过发送对应的请假 id 或者加班 id 与代表是否同意的数字，然后由后端进行茶操作。

## ④对 sector 的相关操作

```
/**
 * 获取所有的部门信息
 * @return
```

```

    */
@RequestMapping(value="",method=RequestMethod.GET)
public String LookForAllSectors() {

    return sectorService.findAllSector();
}

/**

* 获取特定的部门信息

* @param sectorId
* @return
*/
@RequestMapping(value="/{sectorId}",method=RequestMethod.GET)
public String findSpecificSector(@PathVariable int sectorId) {

    return sectorService.findSectorById(sectorId);
}

/**

* 添加一个 sector

* @param s
* @return
*/
@RequestMapping(value="",method=RequestMethod.POST)
public String AddSector(@RequestBody Sector s) {

    return sectorService.AddSector(s);
}

/**

* 修改一个 sector

* @param sectorId

```

```

    * @param s
    * @return
    */
    @RequestMapping(value="/{sectorId}",method=RequestMethod.PUT)
    public String ModifySector(@RequestBody Sector s) {

        return sectorService.ModifySector(s);
    }

    @RequestMapping(value="/{sectorId}",method=RequestMethod.DELETE)
    public String deleteSector(@PathVariable int sectorId) {

        return sectorService.DeleteSector(sectorId);
    }

    @GetMapping("/{sectorId}")
    public String findAllEmployeeBySectorId(@PathVariable int sectorId) {
        return sectorService.findAllEmployeeBySectorId(sectorId);
    }

    @RequestMapping(value="/peoplenum/{sectorId}",method=RequestMethod.GET)
    public String findSectorPeopleNumBySectorId(@PathVariable int sectorId) {
        return sectorService.findPeopleNumInSector(sectorId);
    }

    @RequestMapping(value="/peoplenum/all",method=RequestMethod.GET)
    public String findSectorPeopleNum() {
        return sectorService.findPeopleNumInCompany();
    }
}

```

通过 sector controller 可以实现对 sector 表的相关操作。通过不同的 url, 可以分别实现对 sector 的增删改查。返回对应的人数等一系列操作。

⑤返回对应部门的所有请假申请

```

@Override
public List<AttendanceRecord> findAttendanceRecordBySectorId(int sectorId)
{
    // TODO Auto-generated method stub
    String sql = "select * from attendance_record where employeeid in(select
employee_id from employee where sector_id = ?) ";
    return jdbcTemplate.query(sql, new Object[] {sectorId}, new
AttendanceRecordRowMapper() {});
}

```

使用了子查询的功能，首先从员工表中获取对应部门的员工 id，然后在打卡表中获取对应员工 id 的打卡记录。

#### ⑥查看对应不部门所有待审批的请假记录

```

@RequestMapping(value="/uncheck/sectors/{sectorId}",method=RequestMethod.GET)
T)
public String findUnratifyLeave(@PathVariable int sectorId) {
    return
applicationForLeaveService.findUnratifiedApplicationForleaveBySectorId(sectorId
);
}

```

通过部门 id 获得员工 id，再根据员工 id 查找等待审批的请假记录。将查找出来的请假记录存放到 JSON 中，再返回给前端。

#### ⑦查看所有请假离开的人数

```

@RequestMapping(value="/applicateEWNuAll",method=RequestMethod.GET)
public String applicatedNumberAll() {
    return applicationForEWService.applicatedNumberAll();
}

```

通过执行聚合函数，获取所有的请假的人数。

#### 四、app 部分

通过下载 apk 文件，来获得手机 app



图标设计

会询问是否可以访问设备、媒体内容和文件，选择允许。否则会影响上传照片，二维码扫描等功能。



起始页面设计 (可用于公司活动等宣传)，可以手动跳过，或自动等待倒计时结束进入系统。



app 设计美观，易于操作。

在右上角同时有二维码扫描，和刷新按钮。

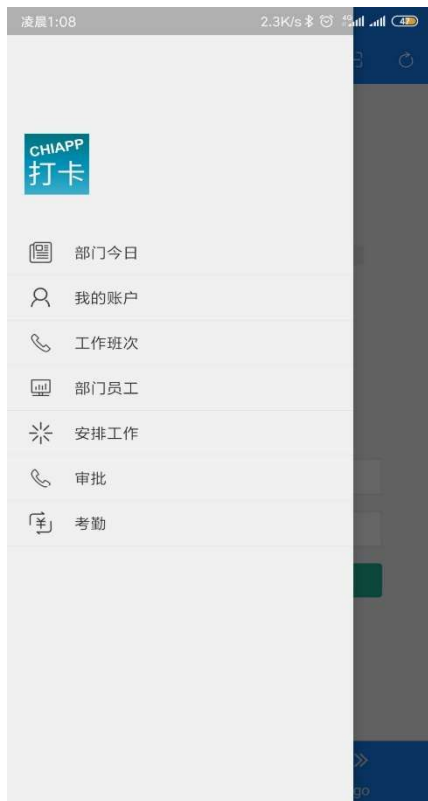


下侧放置前进，主菜单，返回上一级菜单等快捷操作，易于用户使用。

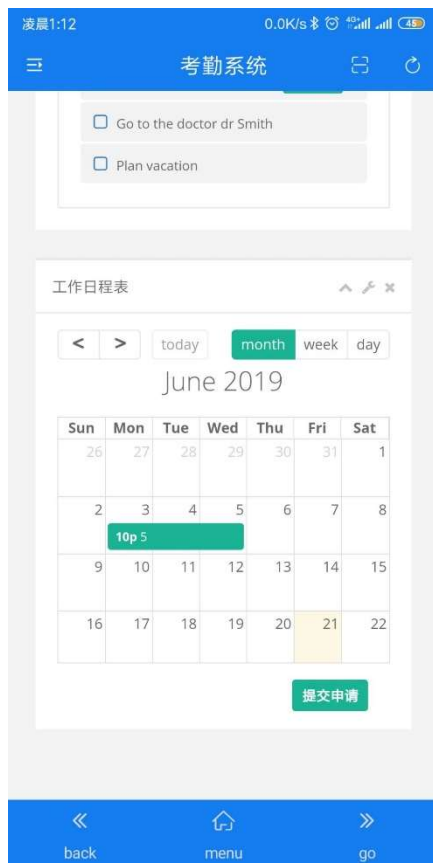


界面设计根据左侧分栏选择应用功能，功能模块安排合理，美观。





各项功能原理同前后端设计，提醒等功能完善。



代码存放地点： GitHub。网址： <https://github.com/softwareehw>

## 五、系统测试

### 5.1 测试方案

功能	输入	输出
用户登录	与数据库连接, 检查用户名和密码是否匹配	对于存在的用户名可以正常登录; 并能给用户正确的返回信息
更新账户信息	编辑个人信息, 检查合法性	能与数据库正常连接, 并即时更新数据库; 正确给出返回信息
查看班次	检查输入查询的班次条件	根据用户权限显示员工的上下班时间
打卡	将脸部与数据库连接, 检验用户是否存在和是否在用户可打卡时间内	能与数据库正常连接, 并即时更新数据库; 对于存在的用户可以正常打卡; 正确给出返回信息
请假	输入并发送请假申请	系统自动提醒主管进行审批, 正确给出返回信息
销假	与数据库连接, 检验用户是否在请假状态及其销假时间是否匹配	能与数据库正常连接, 如果在终止日期前销假, 系统更改用户状态, 并返回销假成功
申请加班	输入并发送加班理由	系统自动提醒主管进行审批, 正确给出返回信息
为员工安排班次	编辑工作班次安排, 检查合法性	能与数据库正常连接, 并即时更新数据库; 正确给出返回信息
调整班次	编辑所要更改员工的工作班次。	能与数据库正常连接, 并即时更新数据库; 正确给出返回信息
批准	输入信息	发送批准消息给员工, 即时更新数据库; 正确给出返回信息

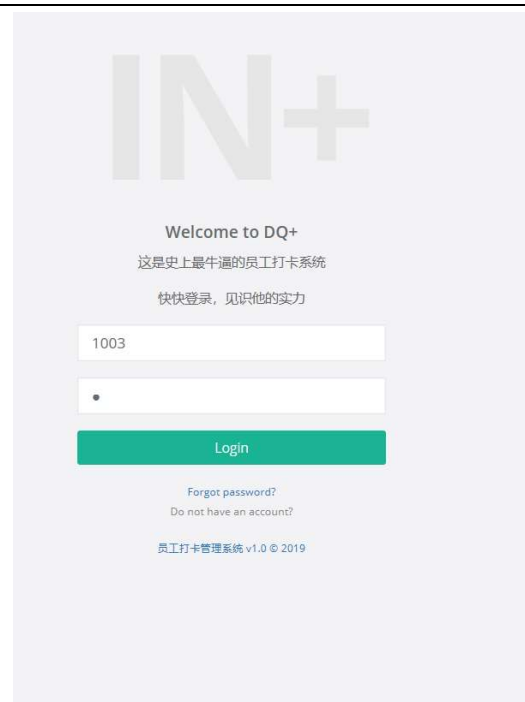
管理员工	导入文件更新员工信息	能与数据库正常连接, 并即时更新数据库; 正确给出返回信息
任免主管	选择用户, 编辑身份	能与数据库正常连接, 并即时更新数据库; 正确给出返回信息
创建临时加班	创建全单位的临时性加班活动, 并提交	能与数据库正常连接, 并即时更新数据库; 发送通知给加班时间不在请假的员工; 正确给出返回信息,

## 5.2 测试结果

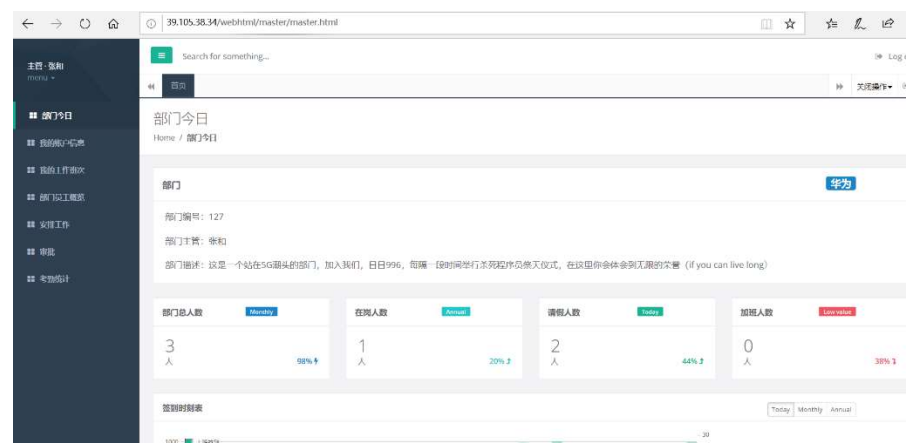
根据设计的测试用例, 填入结果

### 1、用户登录模块

测试用例序号	01	测试用例名称	管理员登录模块	被测试系统	考勤系统
测试功能描述	1: 运行登录对话框 2: 检验输入的帐号和密码 3: 检验输入的帐号和密码是否匹配				
测试用例描述					
测试步骤	1: 输入帐号和密码				
期待输出结果	1: 显示登陆对话框 2: 如果帐号和密码正确进则入系统 3: 反之则提示用户重新输入				
测试结果	1: 显示登陆对话框				

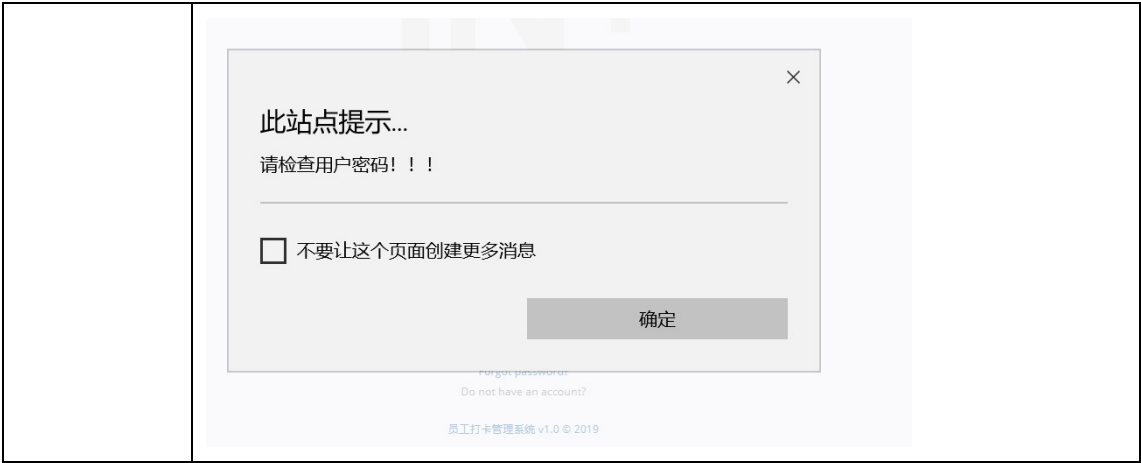


## 2: 如果帐号和密码正确则进入系统



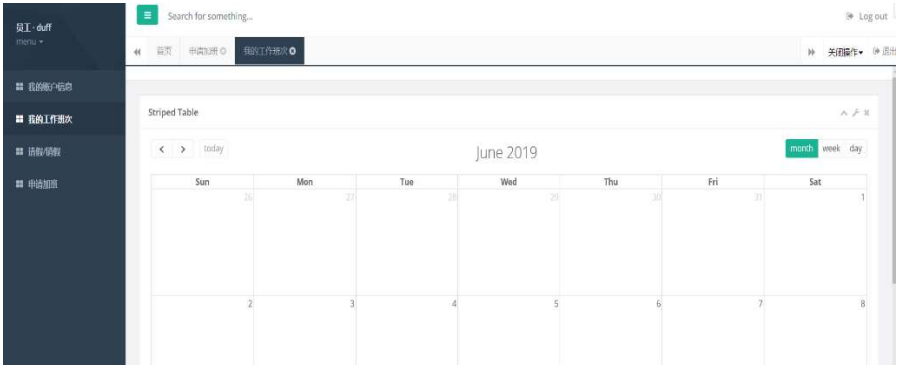
## 3: 反之则提示用户重新输入





2、查看班次模块

测试用例序号	02	测试用例名称	查看班次模块	被测试系统	考勤系统
测试功能描述	1：根据权限显示工作班次 2：可以用年月日三种方式查看 3：可以通过拖拽活动来修改日程安排				
测试用例描述					
测试步骤	1：选择查看我的工作班次				
期待输出结果	1：显示工作日程表				
测试结果	1：显示工作日程表 				

	
备注	需要分别对员工主管两个权限用户进行测试

### 3. 创建临时加班模块

测试用例序号	03	测试用例名称	创建临时加班	被测试系统	考勤系统
测试功能描述	1：创建全单位的临时性加班活动，并提交 2：统记录员工工作信息，发送通知给加班时间不在请假期的员工。				
测试用例描述					
测试步骤	1：查看工作安排 2：创建全单位的临时性加班活动 3：提交				
期待输出结果	1：发送临时加班活动 2：申请成功				
测试结果	<div><div><div>新增临时活动 New</div><div><div>活动名称</div><div>集体加班</div></div><div><div>活动时间</div><div><div><div></div></div>2019-06-21 03:00:00-2019-06-22 04:00:00</div></div><div><div>Cancel</div><div>Save changes</div></div></div><div>保存成功</div><div><div><div>此站点提示...</div><div>申请成功</div><div>确定</div></div></div></div>				

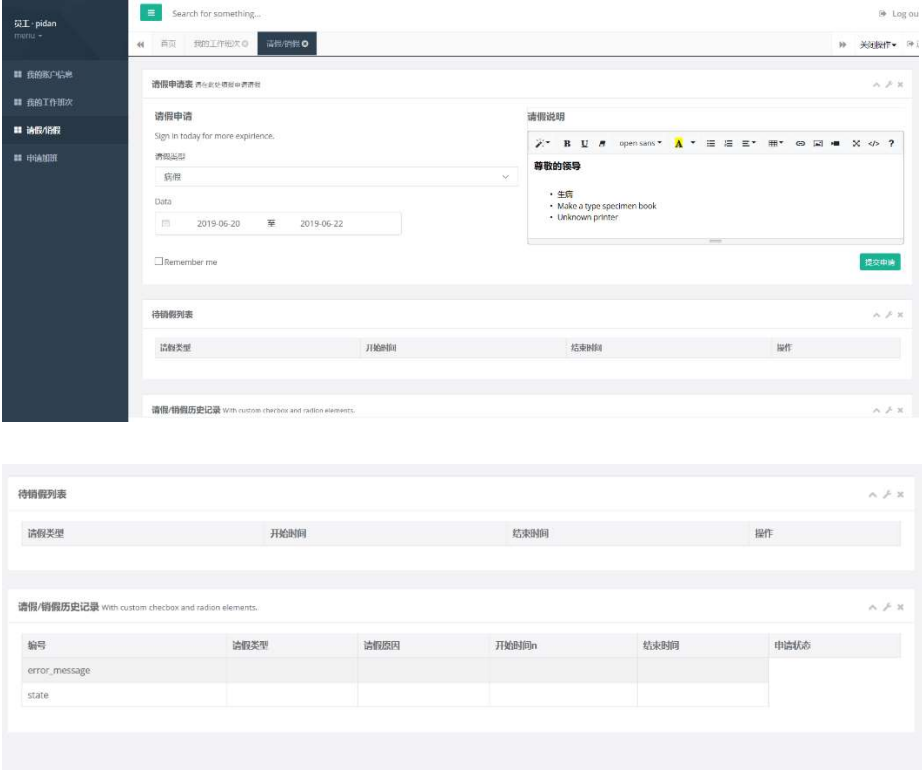
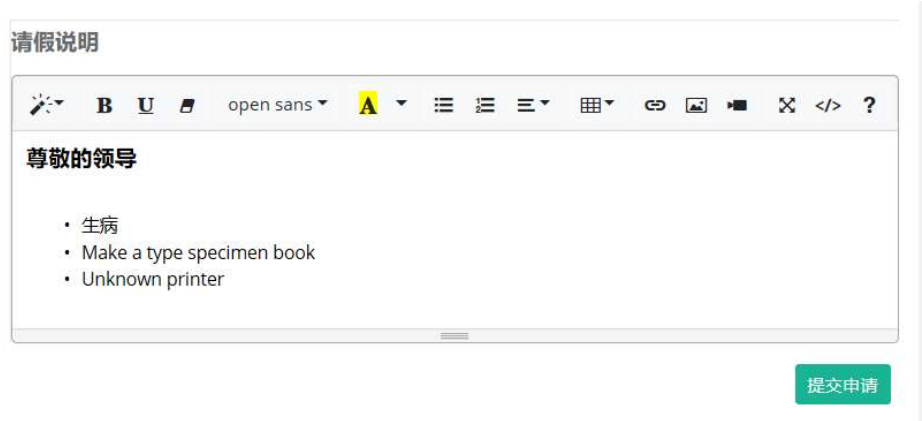
### 4、打卡模块

测试用例序	04	测试用例名	打卡模块	被测系统	考勤系统
-------	----	-------	------	------	------

号		称			
测试功能描述	1: 对镜头刷脸, 检验此人是否存在。 2: 若不存在则提示用户重新刷脸或不存在此人若此人存在, 检测是否在可打卡时间内。 3: 若在可打卡时间内, 提示打卡成功, 系统记录打卡时间; 若不在可打卡时间内, 提示打卡失败				
测试用例描述					
测试步骤	1: 选择打卡 2: 刷脸				
期待输出结果	1: 显示检测结果 2: 若在可打卡时间内, 提示打卡成功 3: 反之则提示用户打卡失败				
测试结果	<p>显示检测结果, 识别失败时的的情况 并且会播放 “no face”</p>  <p>打卡检测成功时的反应 并且会播放 lemon(米津玄师的)</p>  <p>尝试用手机照片欺骗进行打卡的反馈 并且会说 “don't lie me”</p> 				

## 5、请假模块

测试用例序号	05	测试用例名称	请假模块	被测试系统	考勤系统
测试功能描述	1: 用户在网页中提交请假申请 2: 系统将申请发送给主管, 自动提醒主管进行审批				

	3: 根据请假结果更新工作班次
测试用例描述	
测试步骤	1: 输入请假申请 2: 发送请假申请
期待输出结果	1: 显示请假/销假页面 2: 实现输入文本字体选择 3: 如果请假申请被批准, 系统提醒主管调整该员工在请假期间的工作班次安排 4: 如果请假申请没有被批准, 则提醒用户请假失败
测试结果	<p>1: 显示请假/销假页面</p>  <p>2: 实现输入文本字体选择</p> 



发送请求:

请假申请表 请在右侧填写申请表

请假申请

Sign in today for more experience.

请假类型

Data

请假说明

尊敬的领导

• 生病

• Make a type specimen book

• Unknown printer

提交申请

最近一周

最近一个月

最近三个月

2019年6月

2019年7月

结束时间

操作

3: 系统提醒主管有一个待审批的申请

✓ 你有一个带审批的请假申请

4: 如果请假申请被批准, 申请状态变为批准; 如果请假申请没有被批准, 则提醒用户请假被驳回。

请假/销假历史记录 with custom checkbox and radion elements.

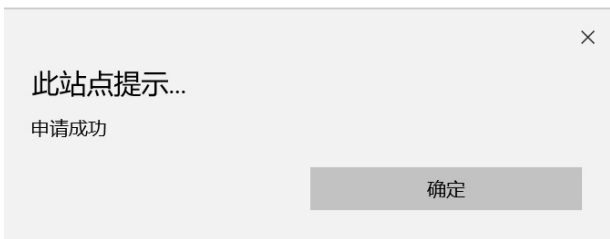
编号	请假类型	请假原因	开始时间n	结束时间	申请状态
0	true	jQuery("#summernoteContent").html()	2019-06-17 08:00:00.0	2019-06-19 08:00:00.0	驳回
1	true	jQuery("#summernoteContent").html()	2019-06-17 08:00:00.0	2019-06-19 08:00:00.0	批准
2	true	jQuery("#summernoteContent").html()	2019-06-17 08:00:00.0	2019-06-19 08:00:00.0	驳回
3	true	jQuery("#summernoteContent").html()	2019-06-12 08:00:00.0	2019-07-28 08:00:00.0	批准
4	true	jQuery("#summernoteContent").html()	2019-06-18 08:00:00.0	2019-07-31 08:00:00.0	批准
5	true	jQuery("#summernoteContent").html()	2019-06-19 08:00:00.0	2019-06-20 08:00:00.0	批准
6	true	jQuery("#summernoteContent").html()	2019-06-18 08:00:00.0	2019-06-20 08:00:00.0	批准


6、销假模块

测试用例序号	06	测试用例名称	销假模块	被测系统	考勤系统								
测试功能描述	1: 用户请假后申请销假 2: 系统先审核是否此员工在请假状态												
测试用例描述													
测试步骤	1: 选择申请销假												
期待输出结果	1: 查看请假状态 3: 如果在终止日期前销假, 系统更改用户状态, 并返回销假成功 4: 如果在终止日期后销假, 记录销假时间, 提示用户销假失败												
测试结果	1: 查看请假状态 <div><div>销假列表</div><table><thead><tr><th>请假类型</th><th>开始时间</th><th>结束时间</th><th>操作</th></tr></thead><tbody><tr><td>true</td><td>2019-06-18 08:00:00.0</td><td>2019-06-20 08:00:00.0</td><td>销假</td></tr></tbody></table></div>					请假类型	开始时间	结束时间	操作	true	2019-06-18 08:00:00.0	2019-06-20 08:00:00.0	销假
请假类型	开始时间	结束时间	操作										
true	2019-06-18 08:00:00.0	2019-06-20 08:00:00.0	销假										


46

2: 如果在终止日期前销假，系统更改用户状态，并返回销假成功

A screenshot of a success dialog box. The dialog has a title bar with a close button (X). The main text reads '此站点提示...' followed by '申请成功'. At the bottom center is a button labeled '确定'.

A screenshot of a table titled '待销假列表'. The table has four columns: '请假类型', '开始时间', '结束时间', and '操作'. The table is currently empty of data rows.

4: 如果在终止日期后销假，记录销假时间，提示用户销假失败

A screenshot of a failure dialog box. The dialog has a title bar with a close button (X). The main text reads '此站点提示...' followed by '网络原因，申请失败'. At the bottom center is a button labeled '确定'.

#### 7、申请加班模块

测试用例序号	07	测试用例名称	申请加班模块	被测试系统	考勤系统
测试功能描述	1：如下班时间超过计划时间一定阈值，系统提醒员工是否要申报加班 2：通过系统提交加班理由，等待主管审批。				
测试用例描述					
测试步骤	1：提交加班理由				
期待输出结果	1：提交加班申请成功 2：提醒主管进行审批				
测试结果	选择加班日期				

加班申请

Sign in today for more experience.

请假类型

2019-06-21

00:00:00

>

2019-06-22

00:00:00

« <

2019 年 6 月

> »

日

一

二

三

四

五

六

26

27

28

29

30

31

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

1

2

3

4

5

6

» <

2019 年 7 月

> »

日

一

二

三

四

五

六

30

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

1

2

3

4

5

6

7

8

9

10

清空

确定

加班说明

<

>

today

Sun

填写并提交加班理由

加班申请表 请在此处填报加班请假

加班申请

Sign in today for more experience.

请假类型

Meeting

申请加班日期

🕒

2019-06-21 00:03:00

-

2019-06-22 00:03:00

加班说明

🔍

**B**

U

🔗

open sans

▼

**A**

▼

☰

☰

☰

☰

☰

🔗

🖼️

🎥

✂️

</>

?

1

2

3

申请成功

×

此站点提示...

申请成功

确定

测试用例序号	07	测试用例名称	调整班次模块	被测试系统	考勤系统
测试功能描述	1: 选择员工, 更改工作班次, 调整工作班次安排可以是临时调整或永久调整				
测试用例描述					
测试步骤	1: 选择查看班次 2: 编辑班次 3: 保存				
期待输出结果	1: 合法操作后, 刷新后修改成功				
测试结果	<p>查看活动</p>  <p>将 21 号的 Meeting 班次, 通过拖拽, 修改移动到 5 号</p>  <p>拖拽后显示结果:</p>				

	5	6	7
	10:30a Meeting		
	12	13	14
	19	20	21
		12p Lunch	
	26	27	28
修改成功			

#### 8、批准模块

测试用例序号	08	测试用例名称	批准模块	被测试系统	考勤系统										
测试功能描述	1. 系统将员工申请发送给主管，可以说加班或者请假申请 2. 部门主管选择批或不批														
测试用例描述															
测试步骤	1：选择查看提醒 2：选择批准或不批准														
期待输出结果	1：如果批准，发送批准消息给员工，考勤记录设置更新 2：如果不批准，发送不批准消息给员工														
测试结果	<div>查看提醒</div> <div><div><div><div></div><div>你有一个带审批的请假申请</div></div><div></div></div></div> <div>审批</div> <div>部门员工请假申请</div> <div><div><div>刷新</div><div>1 同意</div><div>0 驳回</div></div><table><thead><tr><th></th><th>员工姓名</th><th>员工编号</th><th>开始时间</th><th>结束时间</th></tr></thead><tbody><tr><td><input type="checkbox"/></td><td>duff</td><td>12</td><td>2019-06-21 08:00:00.0</td><td>2019-06-22 08:00:00.0</td></tr></tbody></table></div> <div>审批结果</div> <div>同意：</div> <div><div><div>7</div><div>duff</div><div>12</div><div>2019-06-21 00:00:00.0</div><div>2019-06-22 00:00:00.0</div><div>0</div><div>同意</div></div></div> <div>同意成功</div> <div>驳回：</div> <div><div><div>8</div><div>duff</div><div>12</div><div>2019-06-19 00:00:00.0</div><div>2019-06-21 00:00:00.0</div><div>1</div><div>驳回</div></div></div> <div>驳回成功</div>						员工姓名	员工编号	开始时间	结束时间	<input type="checkbox"/>	duff	12	2019-06-21 08:00:00.0	2019-06-22 08:00:00.0
	员工姓名	员工编号	开始时间	结束时间											
<input type="checkbox"/>	duff	12	2019-06-21 08:00:00.0	2019-06-22 08:00:00.0											

### 9、管理员工模块

测试用例序号	09	测试用例名称	管理员工模块	被测试系统	考勤系统
测试功能描述	1. 编辑员工信息 2. 保存并更新员工信息				
测试用例描述					
测试步骤	1: 选择查看员工信息 2: 编辑员工信息 3: 保存更新员工信息				
期待输出结果	1: 若合法输入，提示修改成功 2: 反之则提示用户修改失败				
测试结果	<div> <div> <div>员工</div> <div>  <div> <div>张飞</div> <div>           Riviera State 32/106         </div> <div>           部门编号: 127            员工编号: 22            年龄: 26            性别: 女            工资: 5648 ¥/M            入职时间: 2019-06-17            手机号: (+86) 54687943         </div> </div> </div> <div>生活不止眼前的苟且</div> </div> <div> <div>修改信息</div> <div>删除员工</div> </div> </div> <p>1: 若合法输入，提示修改成功</p>				

姓名

张飞

所在部门

华为

是否为部门主管

否

年龄

20

工资


100

39.105.38.34 显示

success

确定

员工



张飞

📍 Riviera State 32/106

部门编号: 127  
员工编号: 22  
年龄: 20  
性别: 女  
工资: 100 ¥/M  
入职时间: 2019-06-17  
手机号: (+86) 334

生活不止眼前的苟且

2: 反之则提示用户修改失败

[illegible]

生活不止眼前的苟且

张飞

📍 Riviera State 32/106

部门编号: 127

员工编号: 22

年齡：20

性别: 女

工资: 100 ¥ /M

入职时间: 2019-06-17

手机号: (+86) 334

## 10、任免模块

测试用例序号	10	测试用例名称	任免模块	被测试系统	考勤系统
测试功能描述	1. 列出部门所有员工 2. 经理选定一位主管，对他们的身份进行更改，改为员工 3. 4. 保存并更新员工信息，并返回修改成功				
测试用例描述					
测试步骤	1：查看主管信息				



	<div>2: 选定一位主管，编辑身份为员工</div> <div>3: 保存修改</div> <div>4: 选择一名员工，编辑身份为主管</div>
期待输出结果	<div>1: 显示部门所有员工信息</div> <div>2: 选定一位主管，编辑身份为员工，保存修改成</div> <div>3: 选择一名员工，编辑身份为主管，保存修改成</div> <div>4: 将一个有主管部门的员工，升为主管，主管会自动降为员工，保存修改成功</div>
测试结果	<div>1: 查看主管信息</div> <div><div><div><div>主管</div><div><div><div><div><div></div><div>张和</div><div>生活不止眼前的苟且</div></div><div><div>张和</div><div>部门编号: 127</div><div>员工编号: 5</div><div>年龄: 26</div><div>性别: 男</div><div>工资: 3000 ¥/M</div><div>入职时间: 1970-01-01</div><div>手机号: (+86) 1234567</div></div></div></div></div></div><div><div>员工</div><div><div><div><div><div></div><div>duff</div><div>生活不止眼前的苟且</div></div><div><div>duff</div><div>部门编号: 127</div><div>员工编号: 12</div><div>年龄: 25</div><div>性别: 女</div><div>工资: 4567 ¥/M</div><div>入职时间: 2019-06-04</div><div>手机号: (+86) 848613</div></div></div></div></div></div></div><div>2: 选定一位主管，编辑身份为员工</div><div><div><div>基本信息</div><div><div>员工编号</div><div>5</div></div><div><div>入职时间</div><div>1970-01-01</div></div><div><div>姓名</div><div>张和</div></div><div><div>所在部门</div><div>华为</div></div><div><div>是否为部门主管</div><div>否</div></div><div><div>年龄</div><div>26</div></div></div></div><div>保存修改</div><div><div>Save changes</div></div><div><div>39.105.38.34 显示</div><div>success</div><div>确定</div></div></div>

员工



张和

Riviera State 32/106

部门编号: 127  
员工编号: 5  
年龄: 26  
性别: 男  
工资: 3000 ¥/M  
入职时间: 1970-01-01  
手机号: (+86) 1234567

3: 选择一名员工，编辑身份为主管

员工



duff

Riviera State 32/106

部门编号: 127  
员工编号: 12  
年龄: 25  
性别: 女  
工资: 4567 ¥/M  
入职时间: 2019-06-04  
手机号: (+86) 848613

修改信息

修改信息

删除员工

选择是

基本信息

员工编号

12

入职时间

2019-06-04

姓名

duff

所在部门

华为

是否为部门主管

是

年龄

25

保存成功:

39.105.38.34 显示

success

确定

主管



duff

📍 Riviera State 32/106

部门编号: 127

员工编号: 12

年龄: 25

性别: 女

工资: 4567 ¥/M

入职时间: 2019-06-04

手机号: (+86) 848613

生活不止眼前的苟且

4: 将一个有主管部门的员工, 升为主管, 主管会自动降为员工

员工



张和

📍 Riviera State 32/106

部门编号: 127

员工编号: 5

年龄: 26

性别: 男

工资: 3000 ¥/M

入职时间: 1970-01-01

手机号: (+86) 1234567

生活不止眼前的苟且

主管



duff

📍 Riviera State 32/106

部门编号: 127

员工编号: 12

年龄: 25

性别: 女

工资: 4567 ¥/M

入职时间: 2019-06-04

手机号: (+86) 848613

生活不止眼前的苟且

基本信息

员工编号

5

入职时间

1970-01-01

姓名

张和

所在部门

华为

是否为部门主管

是

保存成功

39.105.38.34 显示

success

确定

主管会自动降为员工



### 5.3 分析

除了基本要求操作以外，系统设计特点包括：提醒功能，树莓派人脸打卡，活体检测，上传图片，登陆界面设计，日历组件操控。经过测试，我们的界面拥有很强的自适应。

在第一次按照计划测试用例执行时，发现有许多模块实现有问题，需要调试更改。通过测试帮助我们完善了系统设计和功能实现。

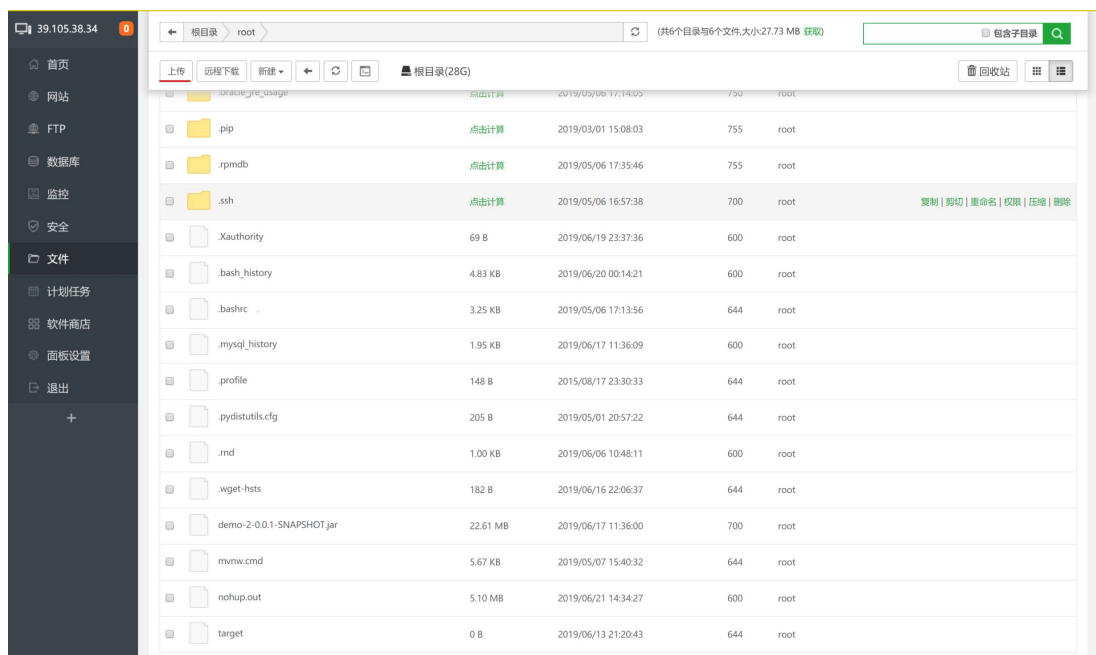
## 六、部署和运行

我们小组租用了阿里云的服务器，其系统为 Linux，我们利用宝塔一键式管理，大大降低了部署的复杂度。系统部署在服务器上，用 IP 就可以访问。服务器软件 Apache，端口号 80。

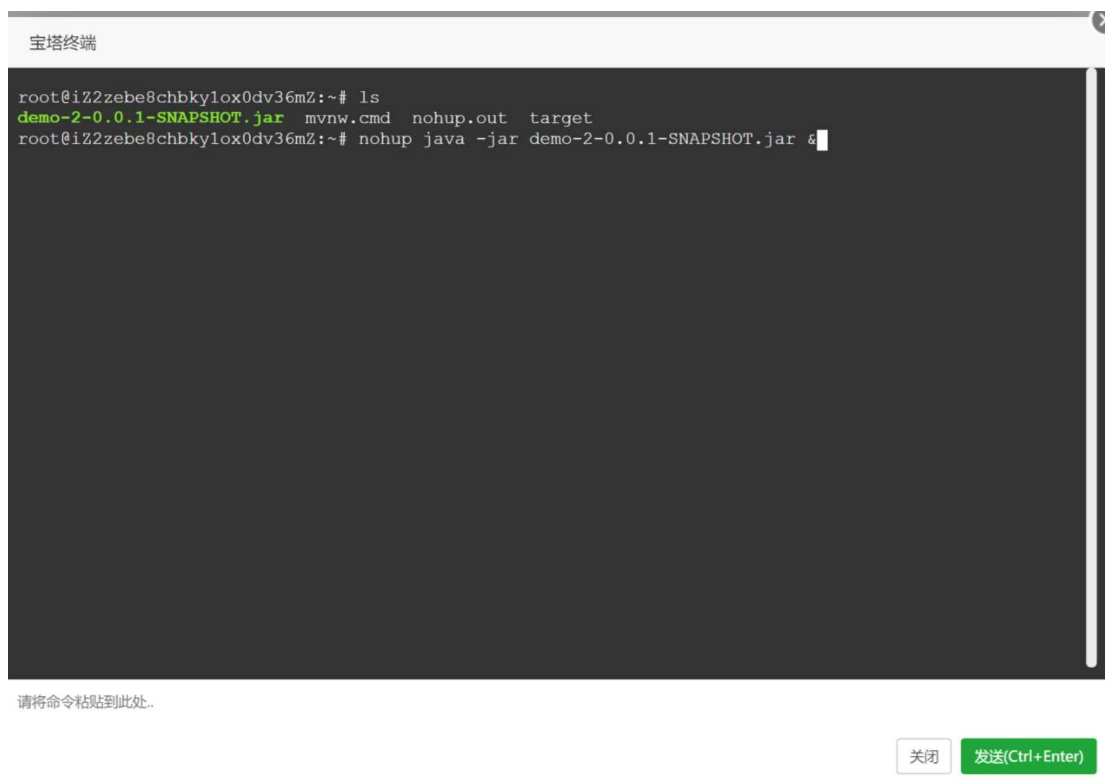
前端利用 JavaScript 脚本语言，边解释边执行，不需要编译。

后端的部署流程如下：

1. 上传 jar 包文件



2. 运行 jar 包文件,执行 `nohup java -jar demo-2-0.0.1-SNAPSHOT.jar &`,  
即可运行成功



## 参考文献

[1] 人脸搜索模块，人脸库管理，在线活体检测的三个参考文档

<http://ai.baidu.com/docs#/Face-Detect-V3/top>

[2] 程瑶.《软件工程》教学中 Android 移动学习 APP 的应用分析[J].电子测试,2019(11):119-120.

[3]曾丽兰.浅谈项目测试需求分析[J].计算机产品与流通,2019(07):283.