

**专业实训2 课程设计**

**设计题目 人事管理系统**

**学院名称 信息科学与技术学院（网络安全学院·牛津布鲁克斯学院）**

**专业名称 计算机科学与技术**

**学生姓名 李强（组长）、叶赟希、秦帅**

**学生学号201813161327、201812030131、201813161315**

**任课教师 林杰**

**设计（论文）成绩**

**教务处 制**

2021年 1月 9日

填写说明

1. 专业名称填写为专业全称，有专业方向的用小括号标明；
2. 格式要求：格式要求：
3. 用A4纸双面打印（封面双面打印）或在A4大小纸上用蓝黑色水笔书写。
4. 打印排版：正文用宋体小四号，1.5倍行距，页边距采取默认形式（上下2.54cm，左右2.54cm，页眉1.5cm，页脚1.75cm）。字符间距为默认值（缩放100%，间距：标准）；页码用小五号字底端居中。
5. 具体要求：

**题目**（二号黑体居中）；

**摘要**(“摘要”二字用小二号黑体居中，隔行书写摘要的文字部分，小4号宋体)；

**关键词**（隔行顶格书写“关键词”三字，提炼3-5个关键词，用分号隔开，小4号黑体)；

正文部分采用三级标题；

**第1章** ××(小二号黑体居中，段前0.5行)

**1.1** ×××××小三号黑体×××××（段前、段后0.5行）

**1.1.1**小四号黑体（段前、段后0.5行）

**参考文献**（黑体小二号居中，段前0.5行），参考文献用五号宋体，参照《参考文献著录规则（GB/T 7714－2005）》。

目录

[1 引言 7](#_Toc61090375)

[1.1 编写目的 7](#_Toc61090376)

[1.2 文档概述 7](#_Toc61090377)

[1.3 参考资料 7](#_Toc61090378)

[2 项目概述 8](#_Toc61090379)

[2.1 项目名称 8](#_Toc61090380)

[2.2 项目背景 8](#_Toc61090381)

[2.3 项目内容 9](#_Toc61090382)

[3 需求分析 10](#_Toc61090383)

[3.1定义 10](#_Toc61090384)

[3.2 系统可行性分析 10](#_Toc61090385)

[3.2.1可行性分析前提 10](#_Toc61090386)

[3.2.2技术可行性分析 10](#_Toc61090387)

[3.2.3经济可行性分析 11](#_Toc61090388)

[3.2.4操作可行性分析 11](#_Toc61090389)

[3.2.5结论 11](#_Toc61090390)

[3.3系统目标 12](#_Toc61090391)

[3.4功能要求 12](#_Toc61090392)

[3.5安全要求 12](#_Toc61090393)

[3.6界面要求 12](#_Toc61090394)

[3.7 环境要求 13](#_Toc61090395)

[3.8 系统UML类图 13](#_Toc61090396)

[3.9 登录时序图 14](#_Toc61090397)

[4 概要设计 16](#_Toc61090398)

[4.1功能设计 16](#_Toc61090399)

[4.2外部接口 17](#_Toc61090400)

[4.3内部接口 17](#_Toc61090401)

[4.4登录模块 18](#_Toc61090402)

[4.5员工管理模块 20](#_Toc61090403)

[4.6账号管理模块 21](#_Toc61090404)

[4.7部门管理模块 22](#_Toc61090405)

[4.8数据库设计 22](#_Toc61090406)

[4.8.1数据库表列表 23](#_Toc61090407)

[4.8.2逻辑结构设计： 25](#_Toc61090432)

[4.8.3物理结构设计： 26](#_Toc61090457)

[4.8.4 数据表代码记录 28](#_Toc61090458)

[5 详细设计 32](#_Toc61090459)

[5.1 登录模块 32](#_Toc61090460)

[5.2 系统管理模块 32](#_Toc61090461)

[5.3 部门管理模块 35](#_Toc61090467)

[5.4 个人信息模块 38](#_Toc61090468)

[5.5 员工薪资模块 39](#_Toc61090469)

[5.6 薪资发放模块 39](#_Toc61090470)

[5.7 薪资发放模块 40](#_Toc61090471)

[5.8 培训计划 43](#_Toc61090472)

[6 设计展示 46](#_Toc61090473)

[6.1页面展示 46](#_Toc61090474)

[6.2登录体验 51](#_Toc61090475)

[6.2.1 管理员登录 51](#_Toc61090476)

[6.2.2 员工登录 52](#_Toc61090478)

[7 项目研发计划 53](#_Toc61090480)

[7.1 计划图解 53](#_Toc61090481)

[附录1：变更纪事 54](#_Toc61090482)

分工详情

|  |  |  |  |
| --- | --- | --- | --- |
|  | 李强 | 秦帅 | 叶赟希 |
| 项目文档分工 | 《需求分析》 | 《概要设计》 | 《详细设计》 |
| 项目功能模块分工 | 登录页面  员工管理功能模块 | 部门管理模块  招聘管理模块  培训计划模块 | 账号管理模块  员工薪资管理模块  薪资发放模块 |

# 引言

## 1.1 编写目的

本文编写目的在于规定和控制项目开发的内容，保证本项目的需求分析活动在受控状态下进行，在进行软件开发前，明确本系统应达到的目标，对系统目标做出完整、准确、清晰、具体的要求。保证本需求分析的结果能够完整、无遗漏地反映待开发系统的要求。

## 1.2 文档概述

本文档主要描述人事管理系统的功能定义，并对模块划分、业务流程进行了定义。所有设计人员、开发人员、测试人员以及其他团队成员都应该以该文档作为产品的功能定义，具体建设内容。

## 1.3 参考资料

《软件需求分析规格说明书》

《软件开发项目管理》

# 项目概述

## 项目名称

成理网络人事管理系统。

## 项目背景

如今各行各业竞争十分激烈，优胜劣汰。我国各行各业都齐头并进,竞争发展。由于信息化技术的快速发展，企业要想提高生产力和创新能力，必须加强对人力资源的研究、配置和投资,以此提升整个企业的人力资源优势。此外，为了留住人才,吸纳人才,将人力及人才资源的优势发挥更好，使企业更好、更快速的发展,必须改进和提升对自身的管理能力。

由于企业员工管理相对复杂,所以给员工管理人员带来了极大的不便。企业员工管理首先要建立人才信息库,然后再根据人才库进行管理、查询等一系列的工作,因此使企业在管理上面能够信息化、科学化、标准化和网络化。本系统是通过Java技术、mysql数据库进行开发的,本系统主要就是小型企业而开发的员工管理系统,开发本系统就是为了能够提高员工管理的工作效率和企业的管理效率。本文主要研究的企业员工管理系统,首先介绍了企业员工管理的选题背景和研究的意义;然后对系统进行需求分析,在需求分析阶段介绍系统的业务需求、非功能性需求、功能用例分析和可行性研究等内容;在系统设计过程中详细介绍了系统的功能模块、业务流程和系统数据库设计等内容;在系统实现与测试方面主要介绍了系统的界面实现,最后再进行了系统的测试。

随着我国现代经济的高速发展,各种企业的不断崛起,企业员工数量的不断增加、分工的不断细化,企业在员工管理方面对需求越来越大,伴随着计算机的迅速发展和广泛使用,企业员工信息管理转向电子化成为必然。实现员工信息管理系统的电子化、网络化,对企业的规范化、制度化、科学化管理具有突出意义。[2]越来越多的企业开始注重对员工管理系统的维护与升级,以此为背景,本文将对企业员工管理系统的设计与升级进行简要探讨。

## 项目内容

本次课题是开发一个小型企业员工管理系统的设计与开发，其主要功能模块如下所示：用户管理、部门管理、职位管理、员工管理、招聘管理、培训管理、公告管理、下载中心，系统管理八大管理模块，从各个方面记录、整合企业人员的信息。

八个功能模块：

（1）用户管理：本模块的主要分为用户查询和添加用户，用户查询可以搜索，删除用户，也可以更改用户信息。添加用户可以增添公司新成员。使用者为管理员，可以对系统的部门和人员信息情况进行管理，有着最高的权限。可以调整普通员工权限，增加或解除某项权利。

（2）部门管理：本模块的主要分为部门查询和添加部门，主要是对所有部门的信息进行管理，包括部门查询，删除，添加，修改部门机构的相关职能信息，完成对企业部门资源的整合工作，统一管理各个部门的信息，调度各部门人员及管理各部门的人员流动。

（3）职位管理：本模块的主要分为职位查询和添加职位，用来储存各职务信息。

（4）员工管理：管理员可以增加员工信息，可以对员工的基本信息进行查询，必要时可对员工的一些基本信息进行删除、修改。

（5）招聘管理：招聘是企业员工管理体系中的极为的重要环节，企业分布招聘岗位，招聘企业所需的人才。这是关键且十分重要的功能。

（6）培训管理：主要是对公司发布的培训方向及技能信息的管理和对所有在职员工进行过培训的信息进行管理，关系到每个员工的成长计划，及促进人才的培养。

（7）公告管理：本模块主要是公司发布一些公告，以便公司员工的查询。

（8）下载中心：本模块主要是管理员可以上传下载公司的文件，也可以删除文件，普通员工只能下载文件。

管理员登录：系统登录功能是系统中不可或缺的重要功能，从安全性方面考虑，企业员工管理系统设计有用户切换功能。

# 需求分析

## 3.1定义

事务流：数据进入模块后可能有多种路径进行处理。

主键：数据库表中的关键域。值互不相同。

外部主键：数据库表中与其他表主键关联的域。

ROLLBACK: 数据库的错误恢复机制。

参考资料:《软件工程导论第六版》

## 3.2 系统可行性分析

### 3.2.1可行性分析前提

本次设计的系统名为小型企业员工系统。目的是帮助各中小企业高效的管理企业员工数据并提高工作效率的员工管理系统。在该系统中，管理员可以对日常的员工系统信息进行相关操作。普通用户只能够查询部分员工系统信息，管理员才能对员工系统进行操作。管理员可以对员工系统进行全方位的业务需求操作，管理员能对所有的系统账号进行控制。系统分为两个权限：超级管理员、普通用户。执行权限操作既方便了操作也保证了系统数据的安全。

### 3.2.2 技术可行性分析

本套系统是一个小型企业员工管理系统，采用SSM框架，开发工具采用Eclipse,本系统开发语言为Java，数据库管理使用关系型数据库MySQL。SSM框架使系统开发工程中能更加关注系统的业务逻辑设计，能更好的提高用户的体验度。JAVA是目前国内比较流行的开发语言，具有功能强大、高效性的特点。JAVA 可以快速开发软件，安全、高效，还可以和xml进行很好的协作。本系统选用MySQL数据库，是因为它具有灵活且易于维护、适合中小企业的特点。MySQL数据库的使用加强了数据库本身的可用性，达到多平台的配适使用; “数据都存在于数据库中，需要系统的数据访问层与数据库交互后，将数据封装即JavaBean，业务逻辑层在调用数据访问层的方法的同时会重新封装JavaBean，这时，Web层的SpringMVC在得到业务逻辑层方法的返回值时，会拿到再次封装的数据即用户需要的数据，但是SpringMVC无法将数据以视图的形式输出，所以需要JSP”。这个系统操作方便，界面简单，容易维护。由此可见，本系统的开发具备了技术可行性。

### 3.2.3经济可行性分析

通过对本套系统的投入成本、后期运营以及维护费用的评估，它是一个较完整的系统并且各方面性价都比非常高。本系统所采用的技术均为目前流行且成熟的技术，系统需要投入开发的费用也不太多，并且系统运行的客户端只需使用Windows操作系统即可，同样不需要投入额外的费用。在国内，JAVA 所需的平台Eclipse是一个免费软件，该系统所使用的数据库软件MySQL也无需付费。MySQL数据库可用性较高，且多平台配适。因整个系统的设计是基于MVC架构，并应用MySQL数据库，降 低了系统运行维护费用。因此本系统运行所需维护费用也相对较低，不难看出，本系统具有一定的经济可行性

### 3.2.4操作可行性分析

该系统的使用操作简单，对不懂计算机知识的使用者也能快速上手。具有友好的操作界面，操作入口设置一目了然。数据录入迅速、规范、可靠;制表灵活、统计精确、适应性强、易于扩展。使用该系统的企业员工，并不需要掌握其他特别的技巧，只需要掌握在Windows平台上使用个人电脑的知识即可，花费很少的时间就可以做到熟练操作。本系统易用性、开放性、灵活性和可见性这些可以基本实现。在系统管理方面，系统是针对角色进行权限控制，所以安全性也有一定的保障，那么在操作可行性上本系统是可行的。

### 3.2.5结论

综上所述，因本系统在经济，技术和操作等方面都是可行的，所以本系统可立即进行开发。

### 3.3系统目标

本次设计的是一个小型企业员工管理系统。主要功能是方便企业员工部门可以对公司员工信息的统一规则化管理，系统管理员对员工系统的所有部门职位人员信息据进行综合的管理，其次本系统其他员工也可以查询个人信息以及查看公司的培训计划，可以进行参加和并查看培训结果，为员工管理人员和公司员工都带来了方便。因此，本设计系统的主要用户是员工系统管理员和本系统公司员工。

### 3.4功能要求

管理员可以在本系统查看个人信息，部门信息，职位信息，招聘信息，培训信息，公告信息。可以根据自己的实际业务需求查看相关的信息，不可以进行任何的增删改操作。管理员拥有更多的功能权限，不但可以查看系统的任何数据，还能对系统数据进行增加，删除和修改等操作。

### 3.5安全要求

员工只能查看个人信息，部门信息，职位信息，招聘信息，培训信息，公告信息，下载一些公司的文件等数据进行搜索和查看，管理员凭借相应的管理账号和密码登录不仅能查看和搜索公司所有数据，还可以改变系统的数据对其进行增加、修改和删除等操作。在本系统中，管理员拥有最高权限，他可以对一切系统信息进行管理。为了防止数据出现某些意外情况，系统会将以管理员身份登录的账号用日志进行详细保存。随时可通过日志来查看管理员的登录时间。

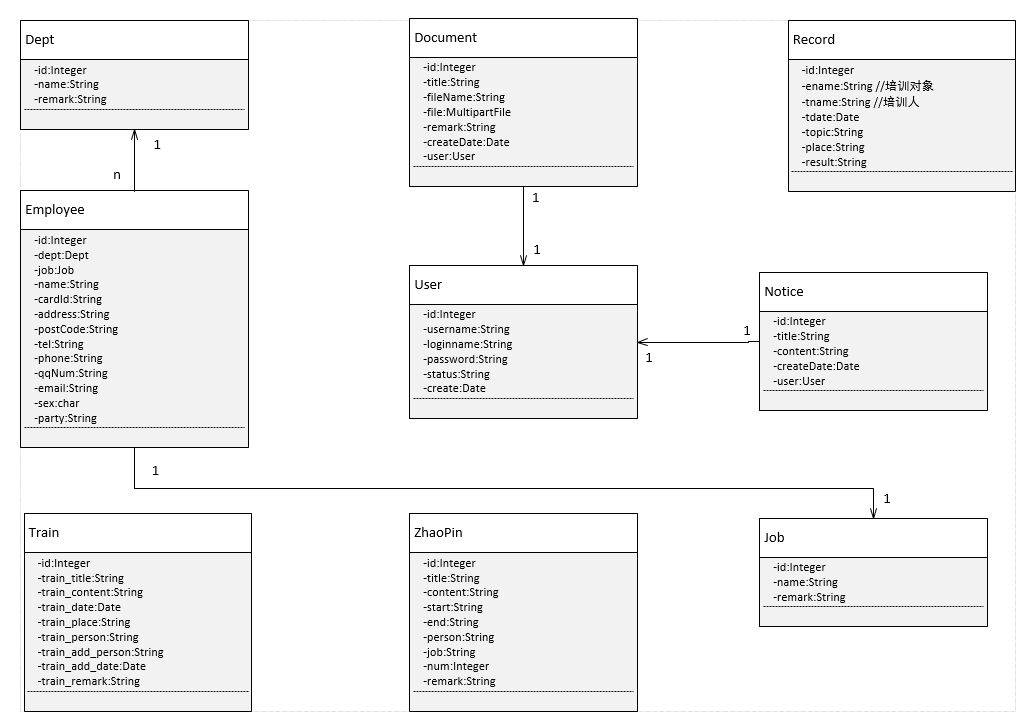
### 3.6界面要求

本系统使用浏览器界面操作。管理员和员工通过登录页面登录，系统的功能页会做出相应的应答。在各个功能页面中，管理员不但对部门信息，职位信息，招聘信息，培训信息，公告信息，进行搜索、修改和查看，查询结果将以分页的形式进行显示。此外，还可以根据需要删除某些信息。管理员员不仅还可以修改个人信息而且还能对员工信息进行更改的功能操作，可以对任意一个员工的相关信息进行更改。

### 3.7环境要求

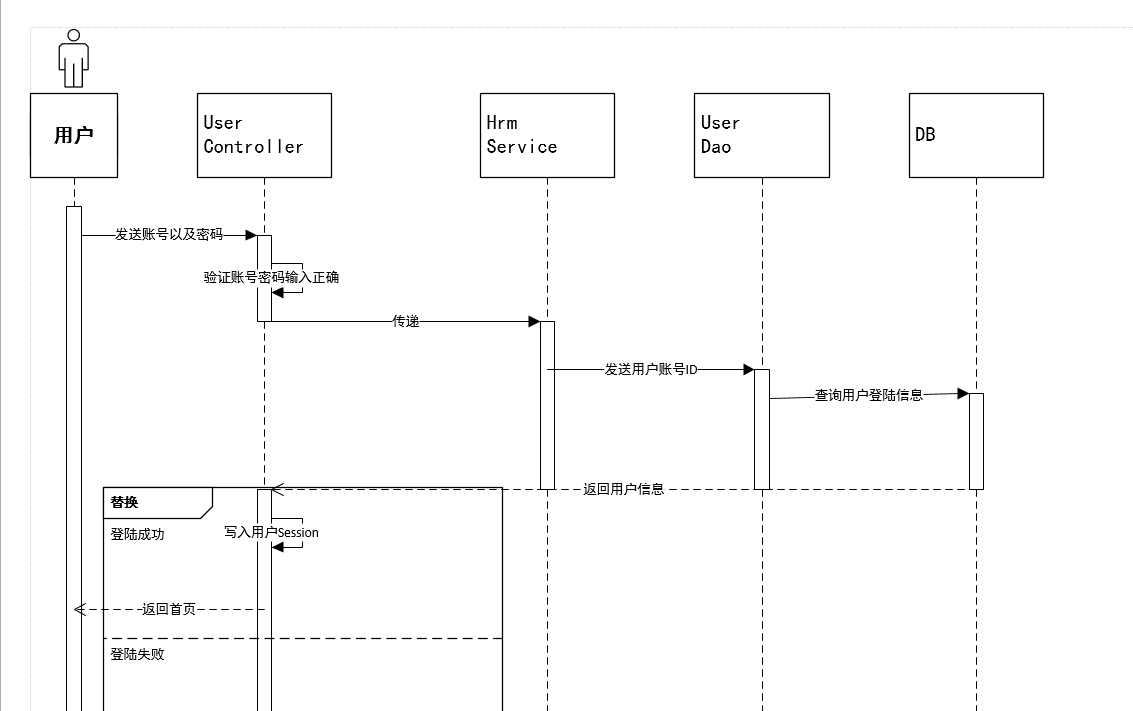
在操作系统环境中，Windows 10的最佳操作环境导致了在系统上运行的mySQL数据库软件的问题。为了使系统继续运行，管理员必须努力管理系统环境，并为解决方案创建必要的控制，以便维护数据系统的安全控制。

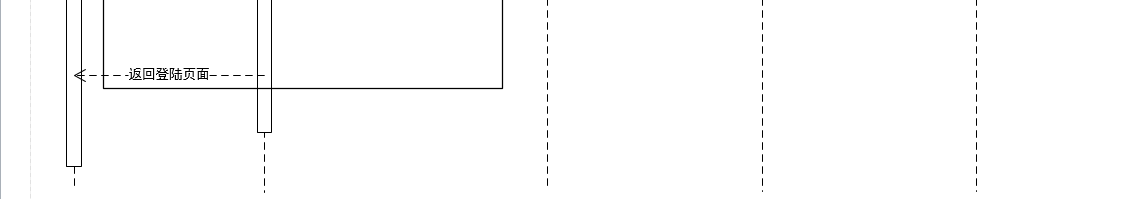
### 3.8 系统UML类图

**图3-1 系统主要类图**

如图为小型企业员工管理系统的主要类图。各个主要类相互独立并且功能明确，管理员从AdminServlet类（登录类）开始，可以进入到各个类，可以选择AdminBookServlet（后台图书管理类），AdminCategoryServlet（后台分类管理类），AdminOrderServlet（后台订单管理类，包含确认收货功能），AdminAddBookServlet（后台添加图书的图片类）。

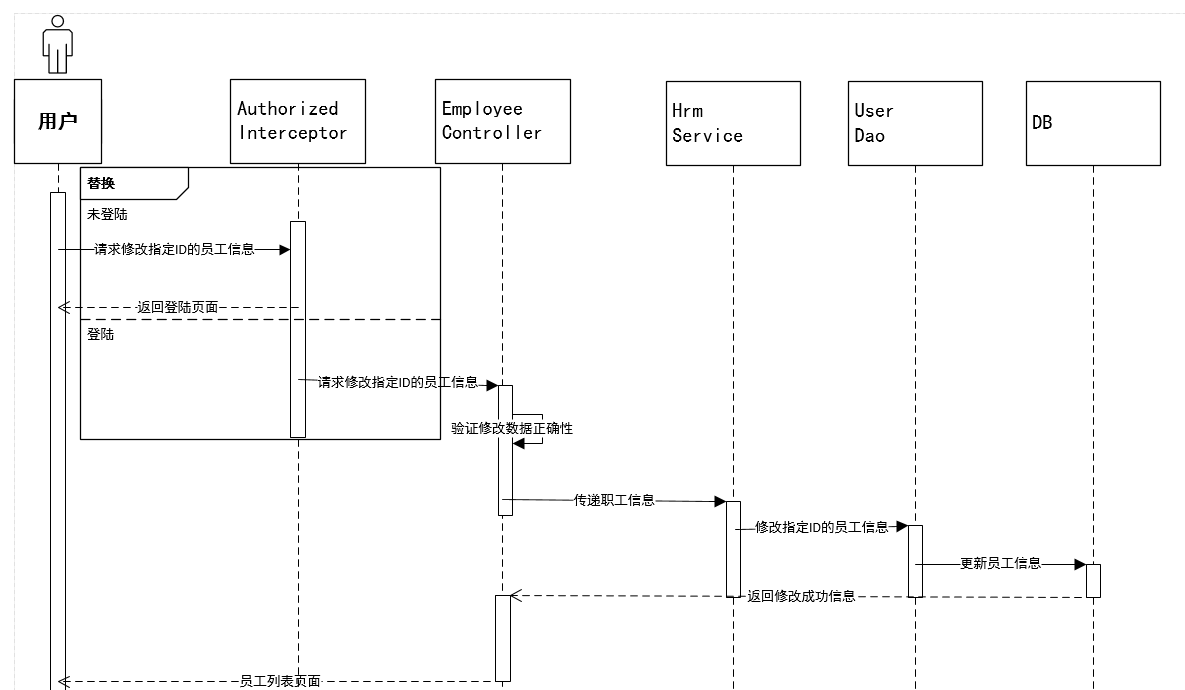
### 3.9 登录时序图





**图3-2 登录时序图**

登录时序图如图所示，用户登录系统后，登录账号会被系统验证用户是否为系统有效账号。若是有效账号系统又会对账号的角色进行检测，如果账号检测为管理员类型，那么用户就可以根据自身的权限对系统数据进行查询，增加，删除和修改等操作，操作后系统将会提示操作是否成功，不成功则返回页面。如果账号为普通用户，则用户只能根据需要查询相关的数据，而不能对系统做出任何增加，修改，删除的相关操作。



**图3-3 用户修改时序图**

用户修改时序图如图所示，管理员登录系统后，进入员工管理，在员工列表可以选中一个用户数据进行查看，可以对员工信息进行相关的修改。

# 概要设计

### 4.1功能设计

本系统主要是管理员，员工，部门信息，职位信息，招聘信息，培训信息，公告信息及其相关数据进行管理，共包含八个功能模块，分别是：用户管理功能，部门管理功能，职位管理功能，员工管理功能，招聘管理功能，培训管理功能，公告管理功能和文档管理功能。

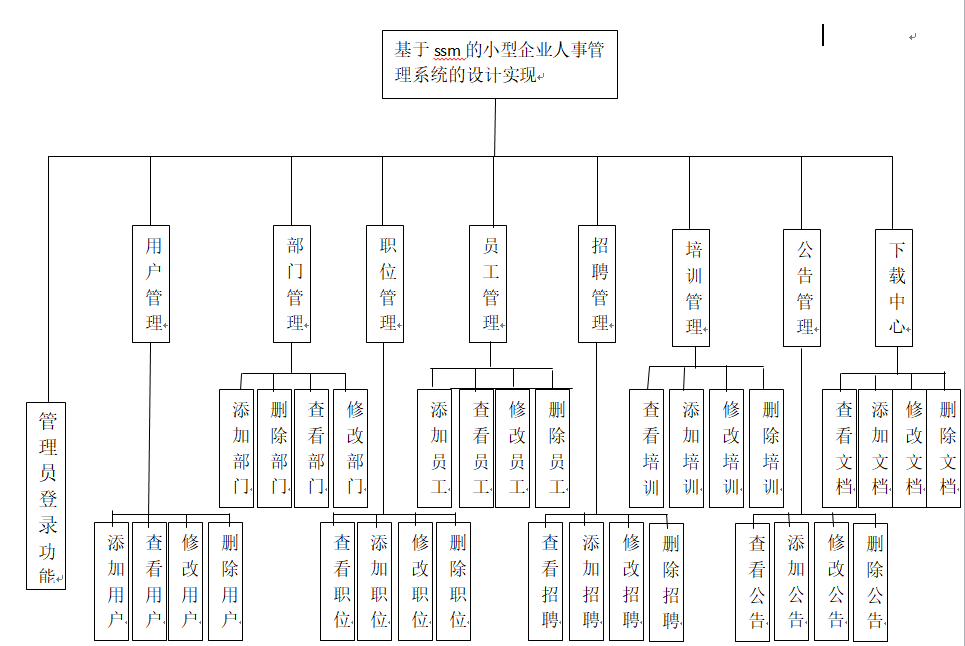


图4-1-1系统模块图（管理员）

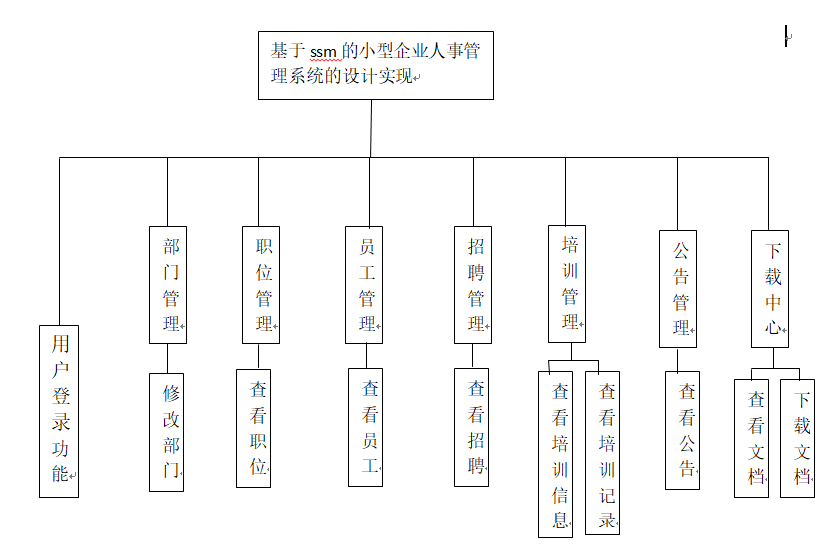


图4-1-2 系统模块图（管理员）

### 4.2外部接口

在用户界面部分，根据需求分析的结果，用户需要一个用户友善界面。在界面设计上，应做到简单明了，易于操作，并且要注意到界面的布局，应突出的显示重要以及出错信息，界面应设计美观，可加入适当的背景图片，减轻使用者的视觉疲劳，增加公司管理人员的工作效率。

### 4.3内部接口

身份认证模块的返回值决定当前用户是否可以进入人员信息模块，身份通过以后再根据用户本身的权限判定进入哪一个子功能模块。

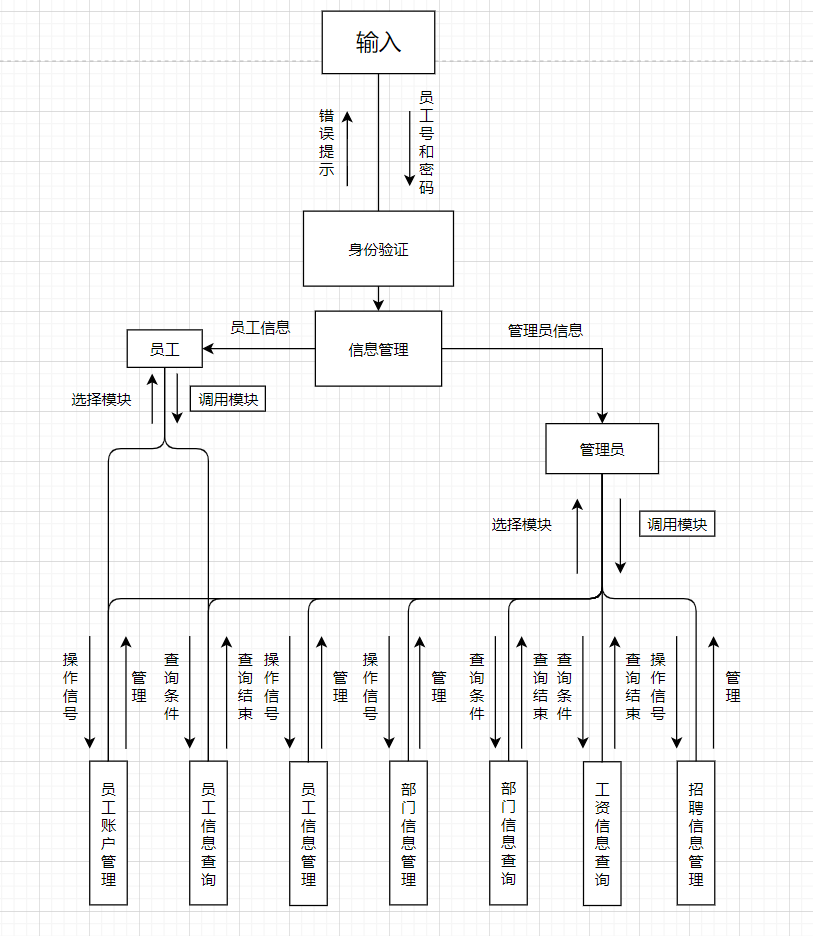


图4-2内部接口图

## 4.4登录模块

需要登陆的人员，对于不同的身份，他们的权限是不一样的。当用户输入ID 和密码时，查询数据库，若用户名和密码正确，则进入相应的员工信息页面，若不正确，则提示用户用户名或密码错误，仍显示当前页面

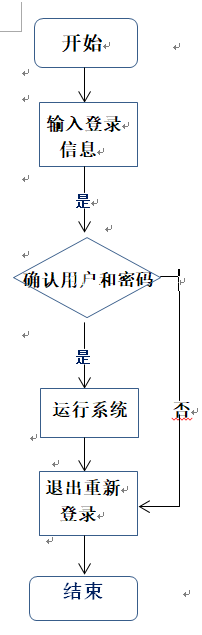


图4-3 登录页面流程图

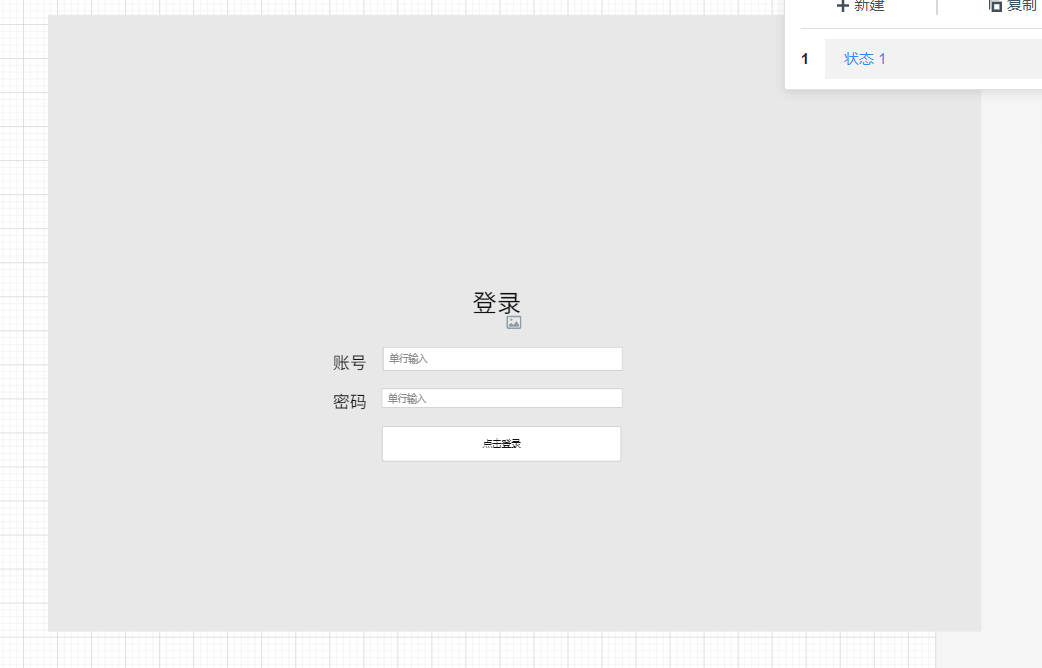


图4-4登录页面设计

## 4.5员工管理模块

员工登录成功后可以对自己的信息进行修改。查看个人信息，部门信息，职位信息，招聘信息，培训信息，公告信息，下载一些公司的文件。另外，管理员也是可以修改个人信息的，但在本系统中意义不大。如图所示：

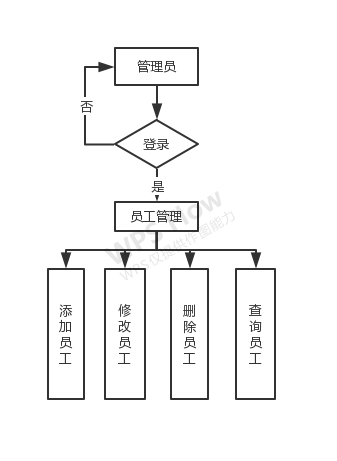


图4-5员工管理流程图



图4-6员工管理设计图

## 4.6账号管理模块

当管理员登录系统后可以对员工进行增加员工账号，删除员工账号，修改员工，查找员工等维护。

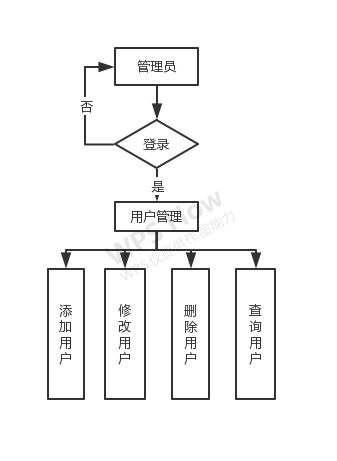


图4-7账号管理图

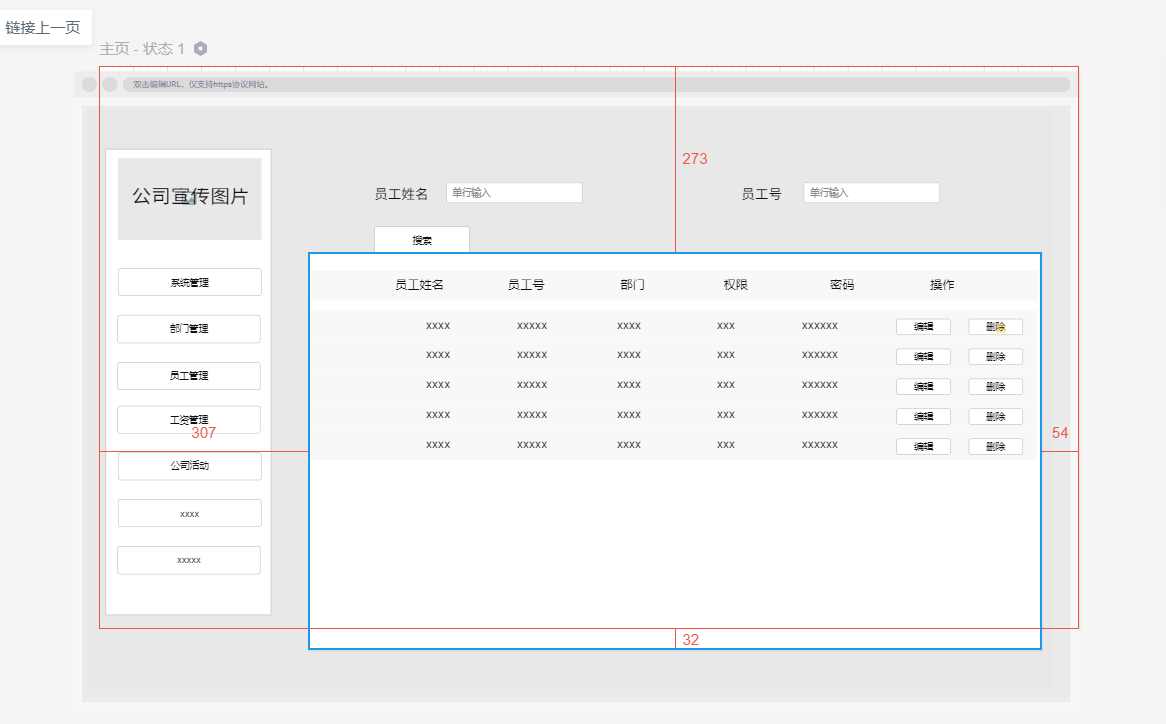


图4-8账号管理设计图

## 4.7部门管理模块

当管理员登录系统后可以对员工进行增加员工，删除员工，修改员工，查找员工等维护。

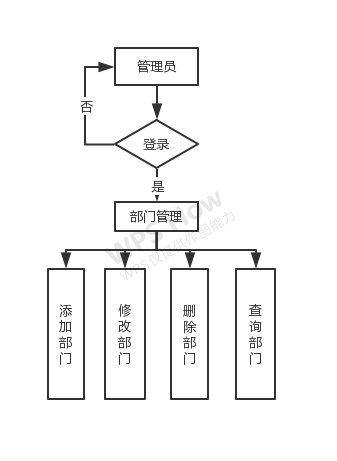


图4-9账号管理流程图

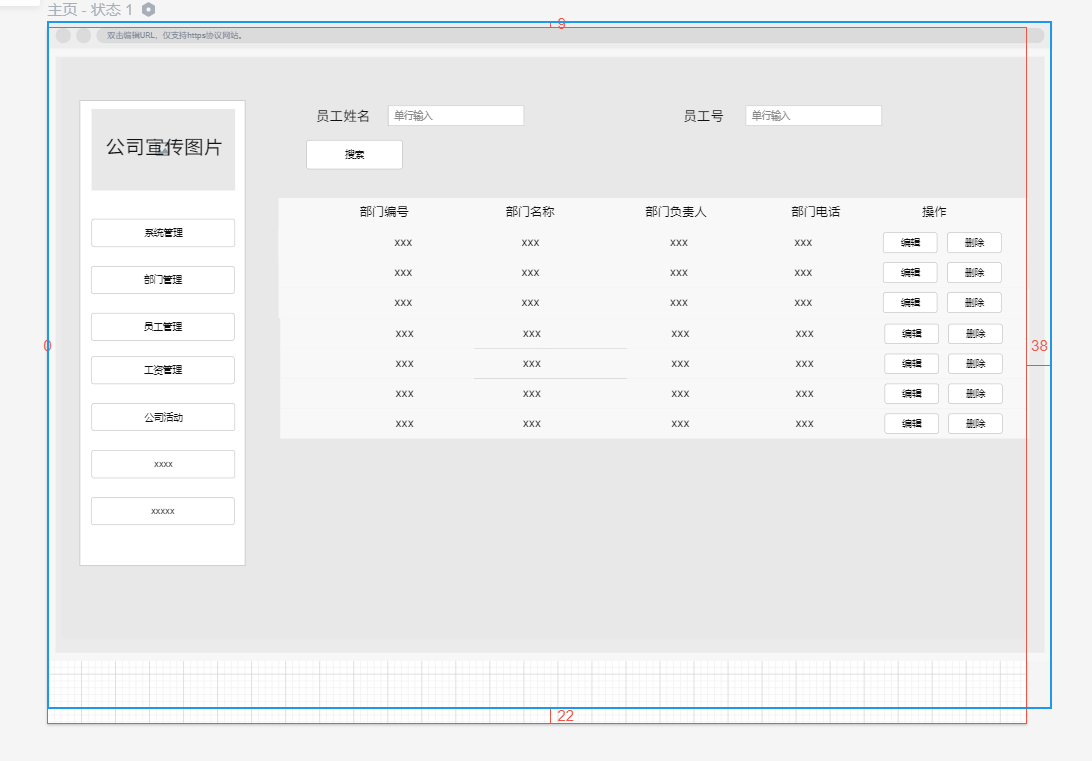


图4-10账号管理设计图

# 4.8数据库设计

数据库名称:rsgl.sql

## 4.8.1数据库表列表

|  |  |  |
| --- | --- | --- |
| 序号 | 关系名称 | 关系的属性 |
| 1 | 部门表 | 存储部门的编号、名称、负责人、电话 |
| 2 | 员工信息表 | 存储员工的编号，名字，年龄，性别，民族，婚姻，手机号，地址，毕业学校，学历，入职时间，部门 |
| 3 | 员工薪资表 | 存储员工的编号、姓名，基本薪资、实发薪资和发放日期以及是否发放 |
| 4 | 薪资发放表 | 存储员工的编号、姓名、所属部门、基本薪资、保险费 |
| 5 | 应聘信息表 | 存储应聘者的编号，姓名，年龄，学历，应聘职位，投递时间 |
| 6 | 培训计划表 | 存储培训的编号、标题、培训人、培训宗旨、培训时间和培训地址以及参加人员 |
| 7 | 账号管理表 | 存储员工编号、密码、姓名，标识部门名称和员工权限 |

## 4.8.2逻辑结构设计：

|  |  |  |
| --- | --- | --- |
| 序号 | 关系名称 | 关系的属性 |
| 1 | 部门表 | 部门编号，部门名称，部门负责人，部门电话 |
| 2 | 员工信息表 | 员工编号，名字，年龄，性别，民族，婚姻，手机号，地址，毕业学校，学历，入职时间，部门 |
| 3 | 员工薪资表 | 员工编号，员工姓名，发放日期，基本薪资，实发薪资，是否发放 |
| 4 | 薪资发放表 | 员工编号，员工姓名，所属部门，基本薪资，保险费 |
| 5 | 应聘信息表 | 应聘编号，姓名，年龄，学历，应聘职位，投递时间 |
| 6 | 培训计划表 | 培训编号，培训标题，培训人，培训宗旨，培训时间，培训地址，参加人员 |
| 7 | 账号管理表 | 员工编号，密码，员工权限，员工姓名，部门名称 |

## 4.8.3物理结构设计：

**部门表(departments)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **说明** | **类型** | **可空** | **备注** |
| **number** | **部门编号** | **varchar(20)** | **No** | **主键** |
| **name** | **部门名称** | **varchar(20)** | **No** |  |
| **manager** | **部门负责人** | **varchar(20)** | **No** |  |
| **telephone** | **部门电话** | **varchar(20)** | **No** |  |

**员工信息表(userinfo)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **说明** | **类型** | **可空** | **备注** |
| **usernumber** | **员工编号** | **varchar(10)** | **NO** | **主键** |
| **name** | **名字** | **varchar(20)** | **YES** |  |
| **age** | **年龄** | **varchar(10)** | **YES** |  |
| **sex** | **性别** | **varchar(10)** | **YES** |  |
| **nation** | **民族** | **varchar(10)** | **YES** |  |
| **wedding** | **婚姻** | **varchar(10)** | **YES** |  |
| **phonenumber** | **手机号** | **varchar(20)** | **YES** |  |
| **address** | **地址** | **varchar(50)** | **YES** |  |
| **school** | **毕业学校** | **varchar(50)** | **YES** |  |
| **education** | **学历** | **varchar(50)** | **YES** |  |
| **worktime** | **入职时间** | **date** | **YES** |  |
| **depnumber** | **部门** | **varchar(50)** | **YES** |  |

**员工薪资表(pay)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **说明** | **类型** | **可空** | **备注** |
| **account** | **员工编号** | **varchar(20)** | **No** | **主键** |
| **username** | **员工姓名** | **varchar(20)** | **No** |  |
| **paytime** | **发放日期** | **date** | **No** |  |
| **basewage** | **基本薪资** | **varchar(20)** | **No** |  |
| **paywage** | **实发薪资** | **varchar(20)** | **No** |  |
| **booleanpay** | **是否发放** | **varchar(20)** | **No** |  |

**薪资发放表(wage)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **说明** | **类型** | **可空** | **备注** |
| **account** | **员工编号** | **varchar(20)** | **No** | **主键** |
| **username** | **员工姓名** | **varchar(20)** | **No** |  |
| **departments** | **所属部门** | **varchar(20)** | **No** |  |
| **bsalary** | **基本薪资** | **varchar(20)** | **No** |  |
| **premium** | **保险费** | **varchar(20)** | **No** |  |

**应聘信息表(employee)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **说明** | **类型** | **可空** | **备注** |
| **eid** | **应聘编号** | **varchar(20)** | **No** | **主键** |
| **ename** | **姓名** | **varchar(20)** | **No** |  |
| **eage** | **年龄** | **varchar(20)** | **No** |  |
| **education** | **学历** | **varchar(20)** | **No** |  |
| **job** | **应聘职位** | **varchar(20)** | **No** |  |
| **sendtime** | **投递时间** | **date** | **No** |  |

**培训计划表(train)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **说明** | **类型** | **可空** | **备注** |
| **tid** | **培训编号** | **varchar(20)** | **No** | **主键** |
| **ttitle** | **培训标题** | **varchar(20)** | **No** |  |
| **name** | **培训人** | **varchar(20)** | **No** |  |
| **purpose** | **培训宗旨** | **varchar(20)** | **No** |  |
| **time** | **培训时间** | **date** | **No** |  |
| **address** | **培训地址** | **varchar(20)** | **No** |  |
| **participants** | **参加人员** | **varchar(20)** | **No** |  |

**账号管理表(user)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **说明** | **类型** | **可空** | **备注** |
| **account** | **员工编号** | **varchar(16)** | **No** | **主键，唯一** |
| **password** | **密码** | **varchar(16)** | **No** |  |
| **authority** | **员工权限** | **int(1)** | **No** | **默认为0** |
| **username** | **员工姓名** | **varchar(25)** | **No** |  |
| **depNumber** | **部门名称** | **varchar(20)** | **No** | **外键约束** |

## 4.8.4 数据表代码记录

**1部门表(departments)：number,name,manager,telephone**

CREATE TABLE `departments` (

`number` varchar(20) NOT NULL,

`name` varchar(20) NOT NULL,

`manager` varchar(20) NOT NULL,

`telephone` varchar(20) DEFAULT NULL,

PRIMARY KEY (`number`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='部门表';

**2.员工信息表(userinfo)：**

**usernumber,name,age,sex,card,birthday,nation,wedding,phonenumber,address,school,education,worktime,depnumber**

CREATE TABLE `userinfo` (

`usernumber` varchar(10) NOT NULL,

`name` varchar(20) DEFAULT NULL,

`age` varchar(10) DEFAULT NULL,

`sex` varchar(10) DEFAULT NULL,

`card` varchar(20) DEFAULT NULL,

`birthday` date DEFAULT NULL,

`nation` varchar(10) DEFAULT NULL,

`wedding` varchar(10) DEFAULT NULL,

`phonenumber` varchar(20) DEFAULT NULL,

`address` varchar(50) DEFAULT NULL,

`school` varchar(50) DEFAULT NULL,

`education` varchar(50) DEFAULT NULL,

`worktime` date DEFAULT NULL,

`depnumber` varchar(50) DEFAULT NULL,

PRIMARY KEY (`usernumber`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='员工信息表';

**3.员工薪资表(pay)：**

**account,username,paytime,basewage,paywage,booleanpay**

CREATE TABLE `pay` (

`account` varchar(20) NOT NULL,

`username` varchar(20) NOT NULL,

`paytime` date NOT NULL,

`basewage` varchar(20) NOT NULL,

`paywage` varchar(20) NOT NULL,

`booleanpay` varchar(20) NOT NULL,

PRIMARY KEY (`account`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

**4.薪资发放表(wage)：account,username,departments,bsalary,premium**

CREATE TABLE `wage` (

`account` varchar(20) NOT NULL,

`username` varchar(20) NOT NULL,

`departments` varchar(20) NOT NULL,

`bsalary` varchar(20) NOT NULL,

`premium` varchar(20) NOT NULL,

PRIMARY KEY (`account`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

**5.应聘信息表(employee)：eid,ename,eage,education,job,sendtime**

CREATE TABLE `employee` (

`eid` varchar(20) NOT NULL,

`ename` varchar(20) NOT NULL,

`eage` varchar(20) NOT NULL,

`education` varchar(20) NOT NULL,

`job` varchar(20) NOT NULL,

`sendtime` date NOT NULL,

PRIMARY KEY (`eid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

**6.培训计划表(train)：tid,ttitle,name,purpose,time,address,participants**

CREATE TABLE `train` (

`tid` varchar(20) NOT NULL,

`ttitle` varchar(20) NOT NULL,

`name` varchar(20) NOT NULL,

`purpose` varchar(20) NOT NULL,

`time` date NOT NULL,

`address` varchar(20) NOT NULL,

`participants` varchar(20) NOT NULL,

PRIMARY KEY (`tid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

**7．账号管理表(user)：account,password,authority,username,depNumber**

CREATE TABLE `user` (

`account` varchar(16) NOT NULL COMMENT '账号也是员工编号',

`password` varchar(16) NOT NULL COMMENT '密码',

`authority` int(1) NOT NULL DEFAULT '0' COMMENT '权限',

`username` varchar(25) NOT NULL COMMENT '姓名',

`depNumber` varchar(20) NOT NULL COMMENT '部门编号',

PRIMARY KEY (`account`),

UNIQUE KEY `username` (`username`,`account`),

KEY `FK\_user\_departments` (`depNumber`),

CONSTRAINT `user\_ibfk\_1` FOREIGN KEY (`depNumber`) REFERENCES `departments` (`number`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

**4.8.5数据库ER图**

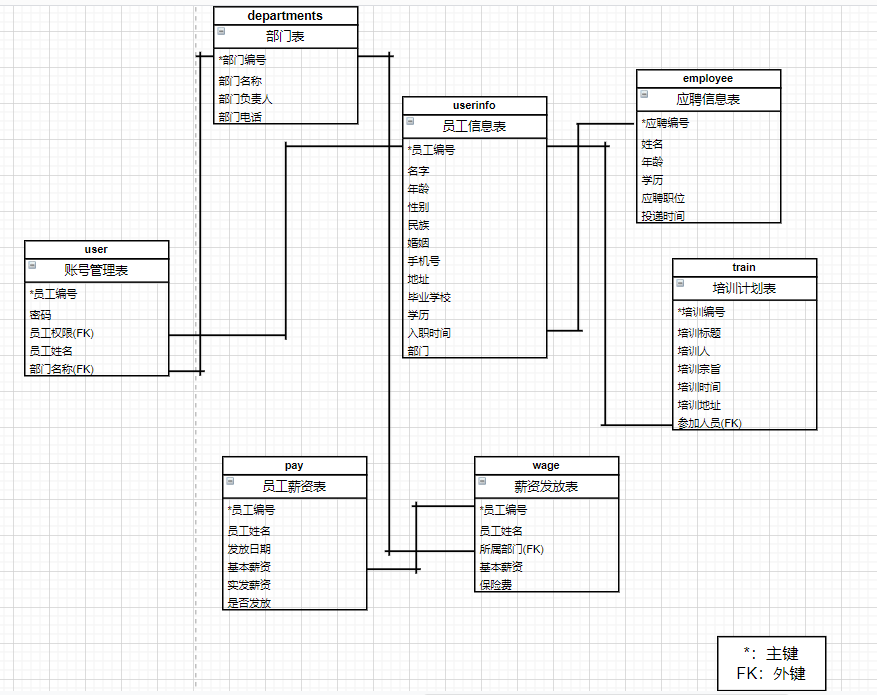


图4-11 E-R图

# 详细设计

**5.1 登录模块**

* 职工输入用户名、密码，会根据用户是否为管理员而加载出管理员/普通用户界面

1.若用户是普通用户，系统将查询数据库中的用户表Users的记录，用户名密码核对正确后加载用户主界面；

2.若用户是管理员，系统将查询数据库中的管理员表Adms的记录，用户名密码核对正确后加载管理员主界面；

3.两种身份，若用户名、密码核对不正确，弹出提示消息框：“用户名或密码错误！”。

**界面设计：**

**用到的控件字段名、方法及变量有：**

* **控件：**

用户名文本框名称：username

密码文本框名称：password

登录标签名称：submit

* **变量:**

private String username;

private String password

**调用方法：**

getUsername() //获取用户名

getPassword() //获取密码

login//判断用户名密码是否正确

loadUserByUsername//判断是否为管理员

**5.2 系统管理模块**

**界面设计**

**用到的控件字段名、方法及变量有：**

* **控件：**

selectByVal//搜索

resetting//重置

addUser//新加用户

deleteSelectUser//删除用户

username//员工名文本框

userid//员工号文本框

* **变量:**

private String username;

private String userid；

**调用方法：**

getUsername() //获取用户名

getUserid() //获取员工号

login//判断用户名密码是否正确

loadUserByUsername//判断是否为管理员

## 主要代码如下：

***UserDetailsService.java***

public class UserDetailsService implements org.springframework.security.core.userdetails.UserDetailsService {

@Autowired

private UserMapper userMapper;

private manage.model.User user;

public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {

UserDetails userDetails = null;

try {

user = userMapper.selectByPrimaryKey(username);

userDetails = new User(username, user.getPassword(), getAuthorities(user.getAuthority()));

} catch (Exception e) {

throw new UsernameNotFoundException("Error in retrieving user");

}

return userDetails;

}

private Collection<GrantedAuthority> getAuthorities(Integer access) {

List<GrantedAuthority> authorities = new ArrayList<GrantedAuthority>();

// 每个用户都具有ROLE\_USER权限

authorities.add(new SimpleGrantedAuthority("ROLE\_USER"));

// 如果参数access为1.则拥有ROLE\_ADMIN权限

if (access.compareTo(1) == 0) {

authorities.add(new SimpleGrantedAuthority("ROLE\_ADMIN"));

}

return authorities;

}

}

***UserServiceImpl.java***

public class UserServiceImpl extends BaseServiceImpl<User, UserMapper> implements UserService<User> {

private static Logger logger = LogManager.getLogger(UserServiceImpl.class);

@Autowired

private UserMapper userMapper;

public List<UserPo> selectAll(PageInfo page, String depNumber ,String userNum,String userName) {

List<UserPo> arrayList = userMapper.selectAllListPage(depNumber, page,userNum,userName);

for (UserPo userPo : arrayList) {

userPo.setDepName(userPo.getDepartments().getName());

}

// List<UserPo> arrayList = new ArrayList<UserPo>();

return arrayList;

}

// 验证功能，验证字段是否已经存在

public String validate(User user, String method) throws Exception {

if (BiSheUtil.notNull(user.getAccount())) {

if (!(method.equals("update"))) {

if (userMapper.selectByPrimaryKey(user.getAccount()) != null) {

return UserEnum.getName("account");

}

}

} else {

return UserEnum.getName("accountNull");

}

if (!BiSheUtil.notNull(user.getPassword())) {

return UserEnum.getName("passWord");

}

if (!BiSheUtil.notNull(user.getDepNumber())) {

return UserEnum.getName("depatrments");

}

if (!BiSheUtil.notNull(user.getUsername())) {

return UserEnum.getName("userName");

}

if (method == "update" || method.equals("update")) {

try {

userMapper.updateByPrimaryKey(user);

} catch (Exception e) {

logger.error("员工更新失败：" + e);

throw new Exception("员工更新失败：" + e);

}

return UserEnum.getName("updateSuccess");

} else {

try {

userMapper.insert(user);

} catch (Exception e) {

logger.error("员工增加失败：" + e);

throw new Exception("员工增加失败：" + e);

}

}

return UserEnum.getName("success");

}

## //批量删除

## public void deleteList(String[] deleteList) {

## userMapper.deleteList(deleteList);

## }

}

## 5.3 部门管理模块

**界面设计**

**用到的控件字段名、方法及变量有：**

* **控件：**

selectByVal//搜索

resetting//重置

addDepartmentname//新加部门

deleteDepartmentname//删除部门

Departmentename//部门名文本框

Departmentid//部门号文本框

* **变量:**

private String Departmentname;

private String Departmentid；

**调用方法：**

getDepartmentname() //获取部门名

getDepartmentid() //获取部门号

**代码如下：**

DepartmentsServiceImpl.java

public class DepartmentsServiceImpl extends BaseServiceImpl<Departments, DepartmentsMapper> implements DepartmentsService<Departments> {

private static Logger logger = LogManager.getLogger(DepartmentsServiceImpl.class);

@Autowired

private DepartmentsMapper departmentsMapper;

@Autowired

private UserMapper userMapper;

public List<Departments> selectAll(PageInfo page,String depNum,String depName) {

List<Departments> DepartmentsList = departmentsMapper.selectAllListPage(page,depNum,depName);

return DepartmentsList;

}

//删除部门，因为有外键

@SuppressWarnings("null")

public String deleteByPrimary(String num ) throws Exception {

try {

//先查询该部门下是否有员工

List<User> depList = userMapper.selectBydepNumber(num);

if (depList.size()!=0) {

//有员工就不删

return "该部门下尚有员工，请先将员工删除或转移后再删除该部门";

}

} catch (Exception e) {

logger.error("删除部门时，查询员工失败：" + e);

throw new Exception("删除部门时，查询员工失败：" + e);

}

try {

//没有员工，删除

departmentsMapper.deleteByPrimaryKey(num);

} catch (Exception e) {

logger.error("删除失败：" + e);

throw new Exception("删除失败：" + e);

}

return "删除成功";

}

* **控件：**

确定按钮名称：submit

* **变量:**

private String number;//部门编号

private String name;//部门名称

private String manager;//部门负责人

private String telephone;//部门电话

* **“确定”按钮的单击事件：**

public String validate(Departments dep,String method) throws Exception{

if (!(method.equals("update"))) {

if (departmentsMapper.selectByPrimaryKey(dep.getName()) != null) {

return "部门编号已经存在";

}

}

if (method == "update" || method.equals("update")) {

try {

departmentsMapper.updateByPrimaryKey(dep);

} catch (Exception e) {

logger.error("部门更新失败：" + e);

throw new Exception("部门更新失败：" + e);

}

return "更新成功";

} else {

try {

departmentsMapper.insert(dep);

} catch (Exception e) {

logger.error("部门增加失败：" + e);

throw new Exception("部门增加失败：" + e);

}

return "增加成功";

}

}

}

}

## 5.4 个人信息模块

**主要代码如下：**

***EmployeeServiceImpl.java***

public class EmployeeServiceImpl extends BaseServiceImpl<Employee, EmployeeMapper> implements EmployeeService<Employee> {

private static Logger logger = LogManager.getLogger(EmployeeController.class);

@Autowired

private EmployeeMapper employeeMapper;

public String validate(Employee employee, String method) throws Exception {

if (!(method.equals("update"))) {

if (employeeMapper.selectByPrimaryKey(employee.getEid())!=null) {

return "招聘编号已经存在";

}

try {

employeeMapper.insert(employee);

} catch (Exception e) {

logger.error("招聘员工信息增加失败：" + e);

throw new Exception("招聘员工信息增加失败：" + e);

}

return "增加成功";

}

if (method == "update" || method.equals("update")) {

try {

employeeMapper.updateByPrimaryKey(employee);

} catch (Exception e) {

logger.error("招聘员工信息更新失败：" + e);

throw new Exception("招聘员工信息更新失败：" + e);

}

return "更新成功";

}

return "success";

}

}

## 5.5 员工薪资模块

**主要代码如下：**

***WageServiceImpl.java***

public class WageServiceImpl extends BaseServiceImpl<Wage, WageMapper> implements WageService<Wage> {

@Autowired

private WageMapper wageMapper;

private static Logger logger = LogManager.getLogger(WageController.class);

public String validate(Wage wage, String method) throws Exception {

if (!(method.equals("update"))) {

if (wageMapper.selectByPrimaryKey(wage.getAccount())!=null) {

return "招聘编号已经存在";

}

try {

wageMapper.insert(wage);

} catch (Exception e) {

logger.error("招聘员工信息增加失败：" + e);

throw new Exception("招聘员工信息增加失败：" + e);

}

return "增加成功";

}

if (method == "update" || method.equals("update")) {

try {

wageMapper.updateByPrimaryKey(wage);

} catch (Exception e) {

logger.error("招聘员工信息更新失败：" + e);

throw new Exception("招聘员工信息更新失败：" + e);

}

return "更新成功";

}

return "success";

}

}

## 5.6 薪资发放模块

**主要代码如下：**

***PayServiceImpl.java***

public class PayServiceImpl extends BaseServiceImpl<Pay, PayMapper> implements PayService<Pay> {

@Autowired

private PayMapper payMapper;

private static Logger logger = LogManager.getLogger(PayController.class);

public String validate(Pay pay, String method) throws Exception {

if (!(method.equals("update"))) {

if (payMapper.selectByPrimaryKey(pay.getAccount())!=null) {

return "招聘编号已经存在";

}

try {

payMapper.insert(pay);

} catch (Exception e) {

logger.error("招聘员工信息增加失败：" + e);

throw new Exception("招聘员工信息增加失败：" + e);

}

return "增加成功";

}

if (method == "update" || method.equals("update")) {

try {

payMapper.updateByPrimaryKey(pay);

} catch (Exception e) {

logger.error("招聘员工信息更新失败：" + e);

throw new Exception("招聘员工信息更新失败：" + e);

}

return "更新成功";

}

return "success";

}

}

## 5.7 薪资发放模块

主要代码如下

WageControllerjava

public class WageController extends BaseController<Wage,WageService<Wage>> {

@Autowired

private WageService wageService;

/\*

\* @author

\* @description

\* @date

\*/

@RequestMapping(value = "wage", produces = "text/html;charset=UTF-8", method = RequestMethod.GET)

@ResponseBody

public String find(HttpServletRequest request) {

String pageSize = request.getParameter("pageSize");

String currentPage = request.getParameter("currentPage");

PageInfo page = getPage(pageSize, currentPage);

@SuppressWarnings("unchecked")

List<Wage> wageList = (List<Wage>) wageService.selectAll(page);

validate(page);

HashMap result = new HashMap();

result.put("arrayList", wageList );

result.put("pageInfo", page);

String returnInfo = JSONObject.toJSONString(result);

return returnInfo;

}

@RequestMapping(value = "wage", produces = "text/html;charset=UTF-8", method = RequestMethod.POST)

@ResponseBody

public String add(Wage wage) throws Exception {

String res = wageService.validate(wage, INSERT);

String returnInfo = JSONObject.toJSONString(res);

return returnInfo;

}

@RequestMapping(value = "wageSelect", produces = "text/html;charset=UTF-8", method = RequestMethod.POST)

@ResponseBody

public String selectByPrimaryKey(@RequestBody String account) {

Wage wage =(Wage) wageService.selectByPrimaryKey(account);

String json = JSONObject.toJSONString(wage);

return json;

}

@RequestMapping(value = "wage", produces = "text/html;charset=UTF-8", method = RequestMethod.PUT)

@ResponseBody

public String update(Wage wage) throws Exception {

String res = wageService.validate(wage, UPDATE);

String returnInfo = JSONObject.toJSONString(res);

return returnInfo;

}

@RequestMapping(value = "wage", produces = "text/html;charset=UTF-8", method = RequestMethod.DELETE)

@ResponseBody

public String delete(@RequestBody String number) throws Exception {

String account = number;

wageService.deleteByPrimaryKey(account);

return "删除成功";

}

}

WageController.java

public class WageController extends BaseController<Wage,WageService<Wage>> {

@Autowired

private WageService wageService;

@RequestMapping(value = "wage", produces = "text/html;charset=UTF-8", method = RequestMethod.GET)

@ResponseBody

public String find(HttpServletRequest request) {

String pageSize = request.getParameter("pageSize");

String currentPage = request.getParameter("currentPage");

PageInfo page = getPage(pageSize, currentPage);

@SuppressWarnings("unchecked")

List<Wage> wageList = (List<Wage>) wageService.selectAll(page);

validate(page);

HashMap result = new HashMap();

result.put("arrayList", wageList );

result.put("pageInfo", page);

String returnInfo = JSONObject.toJSONString(result);

return returnInfo;

}

@RequestMapping(value = "wage", produces = "text/html;charset=UTF-8", method = RequestMethod.POST)

@ResponseBody

public String add(Wage wage) throws Exception {

String res = wageService.validate(wage, INSERT);

String returnInfo = JSONObject.toJSONString(res);

return returnInfo;

}

@RequestMapping(value = "wageSelect", produces = "text/html;charset=UTF-8", method = RequestMethod.POST)

@ResponseBody

public String selectByPrimaryKey(@RequestBody String account) {

Wage wage =(Wage) wageService.selectByPrimaryKey(account);

String json = JSONObject.toJSONString(wage);

return json;

}

@RequestMapping(value = "wage", produces = "text/html;charset=UTF-8", method = RequestMethod.PUT)

@ResponseBody

public String update(Wage wage) throws Exception {

String res = wageService.validate(wage, UPDATE);

String returnInfo = JSONObject.toJSONString(res);

return returnInfo;

}

@RequestMapping(value = "wage", produces = "text/html;charset=UTF-8", method = RequestMethod.DELETE)

@ResponseBody

public String delete(@RequestBody String number) throws Exception {

String account = number;

wageService.deleteByPrimaryKey(account);

return "删除成功";

}

}

**5.8 培训计划**

**主要代码如下**

TrainController.java

public class TrainController extends BaseController<Train, TrainService<Train>> {

@Autowired

private TrainService trainService;

@RequestMapping(value = "train", produces = "text/html;charset=UTF-8", method = RequestMethod.GET)

@ResponseBody

public String find(HttpServletRequest request) {

String pageSize = request.getParameter("pageSize");

String currentPage = request.getParameter("currentPage");

PageInfo page = getPage(pageSize, currentPage);

@SuppressWarnings("unchecked")

List<Train> trainList = (List<Train>) trainService.selectAll(page);

validate(page);

HashMap result = new HashMap();

result.put("arrayList", trainList );

result.put("pageInfo", page);

String returnInfo = JSONObject.toJSONString(result);

return returnInfo;

}

@RequestMapping(value = "train", produces = "text/html;charset=UTF-8", method = RequestMethod.POST)

@ResponseBody

public String add(Train train) throws Exception {

String res = trainService.validate(train, INSERT);

String returnInfo = JSONObject.toJSONString(res);

return returnInfo;

}

@RequestMapping(value = "trainSelect", produces = "text/html;charset=UTF-8", method = RequestMethod.POST)

@ResponseBody

public String selectByPrimaryKey(@RequestBody String account) {

String tid = account;

Train train =(Train) trainService.selectByPrimaryKey(tid);

String json = JSONObject.toJSONString(train);

return json;

}

@RequestMapping(value = "train", produces = "text/html;charset=UTF-8", method = RequestMethod.PUT)

@ResponseBody

public String update(Train train) throws Exception {

String res = trainService.validate(train, UPDATE);

String returnInfo = JSONObject.toJSONString(res);

return returnInfo;

}

@RequestMapping(value = "train", produces = "text/html;charset=UTF-8", method = RequestMethod.DELETE)

@ResponseBody

public String delete(@RequestBody String number) throws Exception {

String tid = number;

trainService.deleteByPrimaryKey(tid);

return "删除成功";

}

}

TrainServiceImpl.java

public class TrainServiceImpl extends BaseServiceImpl<Train, TrainMapper> implements TrainService<Train> {

private static Logger logger = LogManager.getLogger(EmployeeController.class);

@Autowired

private TrainMapper trainMapper;

public String validate(Train train, String method) throws Exception {

if (!(method.equals("update"))) {

if (trainMapper.selectByPrimaryKey(train.getTid())!=null) {

return "培训编号已经存在";

}

try {

trainMapper.insert(train);

} catch (Exception e) {

logger.error("培训信息增加失败：" + e);

throw new Exception("培训信息增加失败：" + e);

}

return "增加成功";

}

if (method == "update" || method.equals("update")) {

try {

trainMapper.updateByPrimaryKey(train);

} catch (Exception e) {

logger.error("培训信息更新失败：" + e);

throw new Exception("培训信息更新失败：" + e);

}

return "更新成功";

}

return "success";

}

}

# 设计展示

**6.1页面展示**

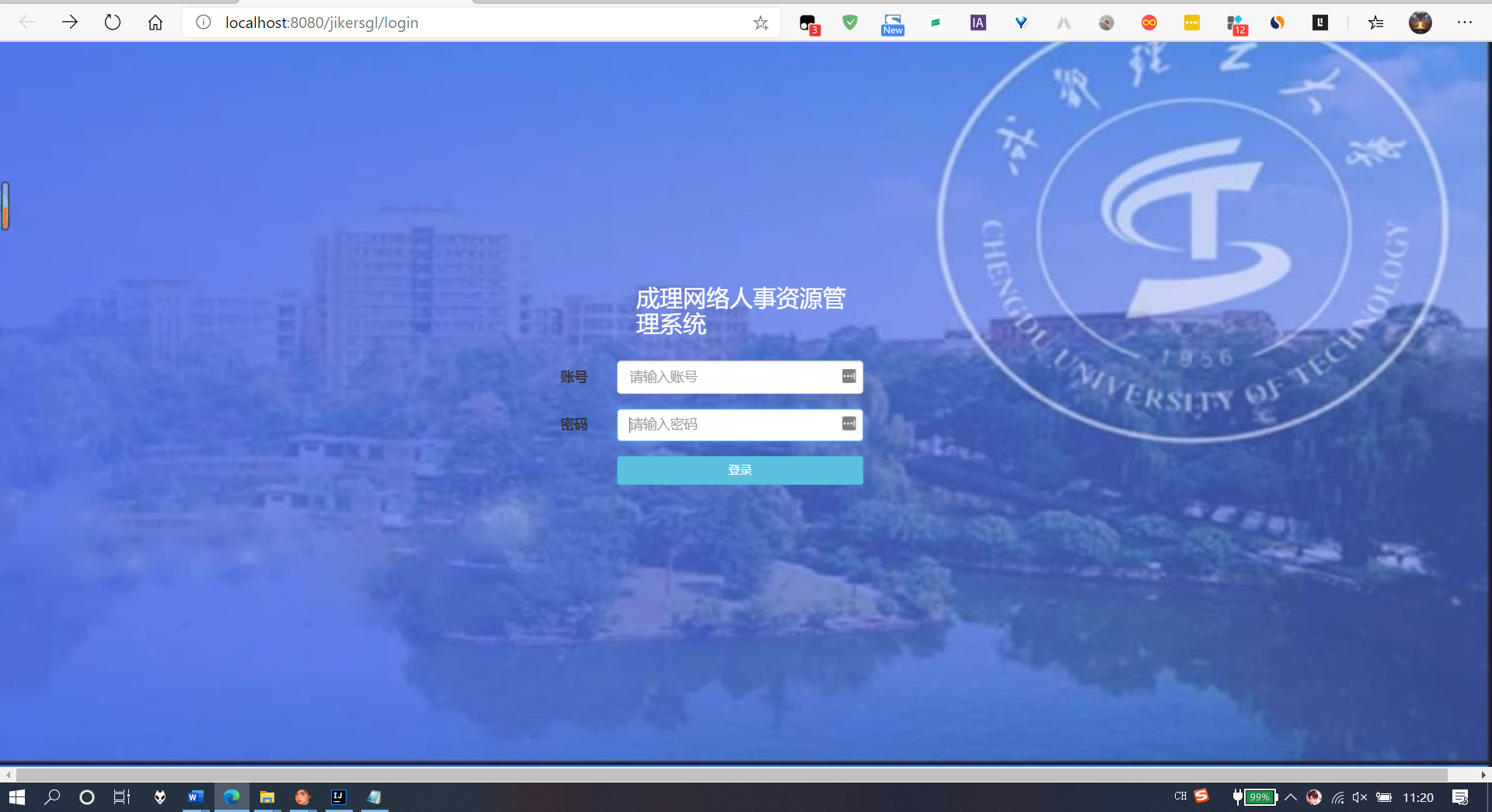


图6-1登录页面

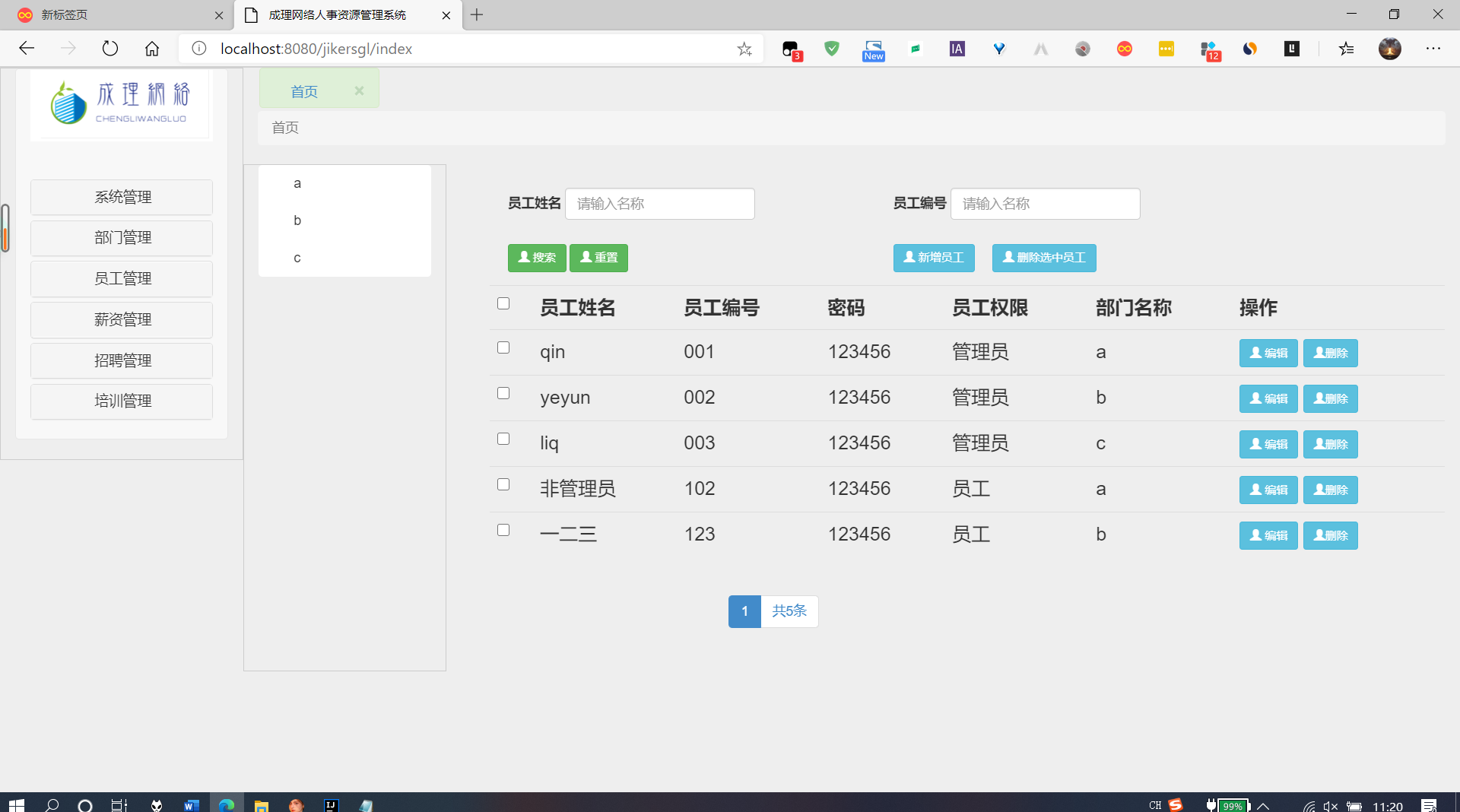


图6-2系统管理页面

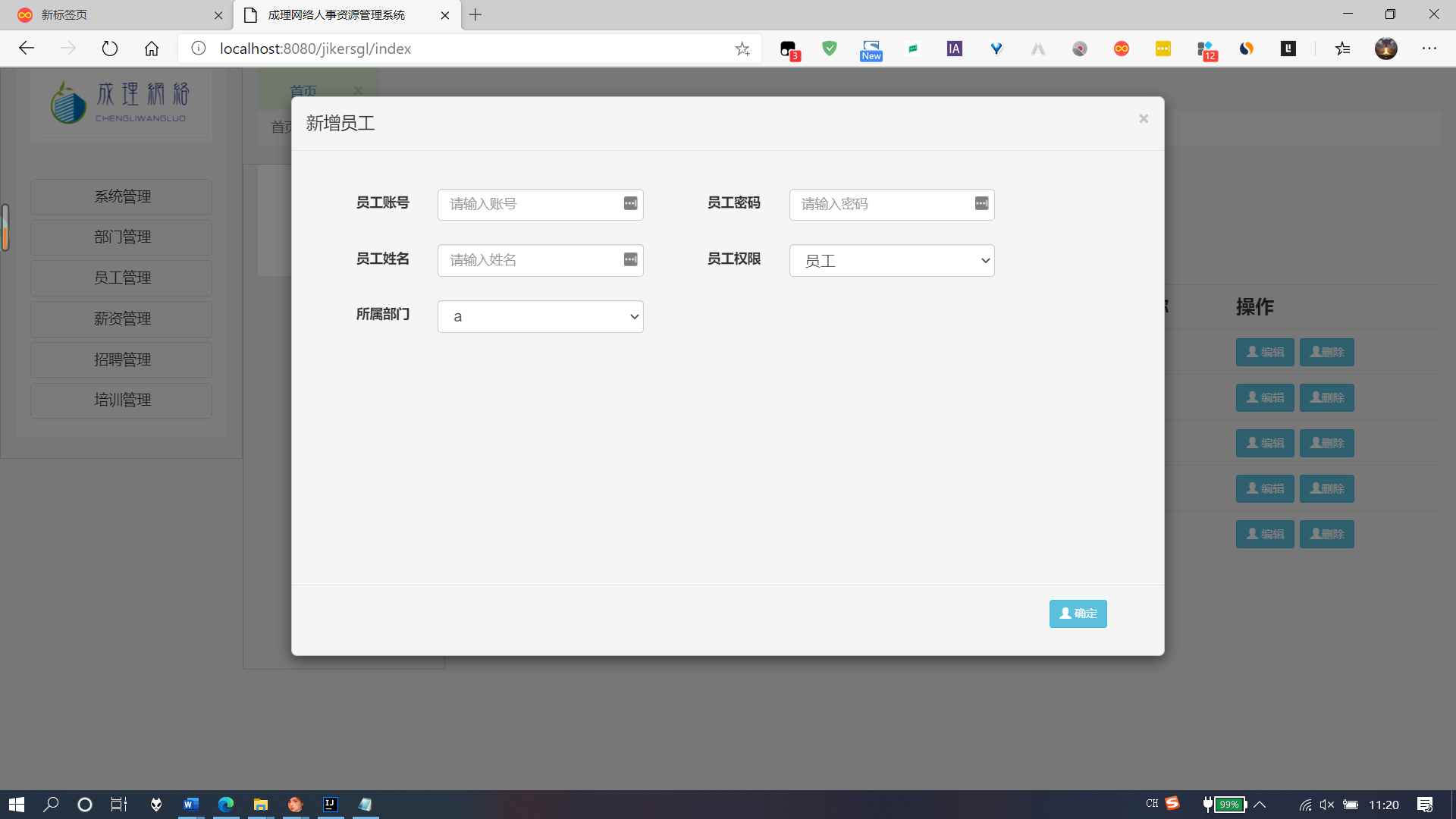


图6-3账户添加

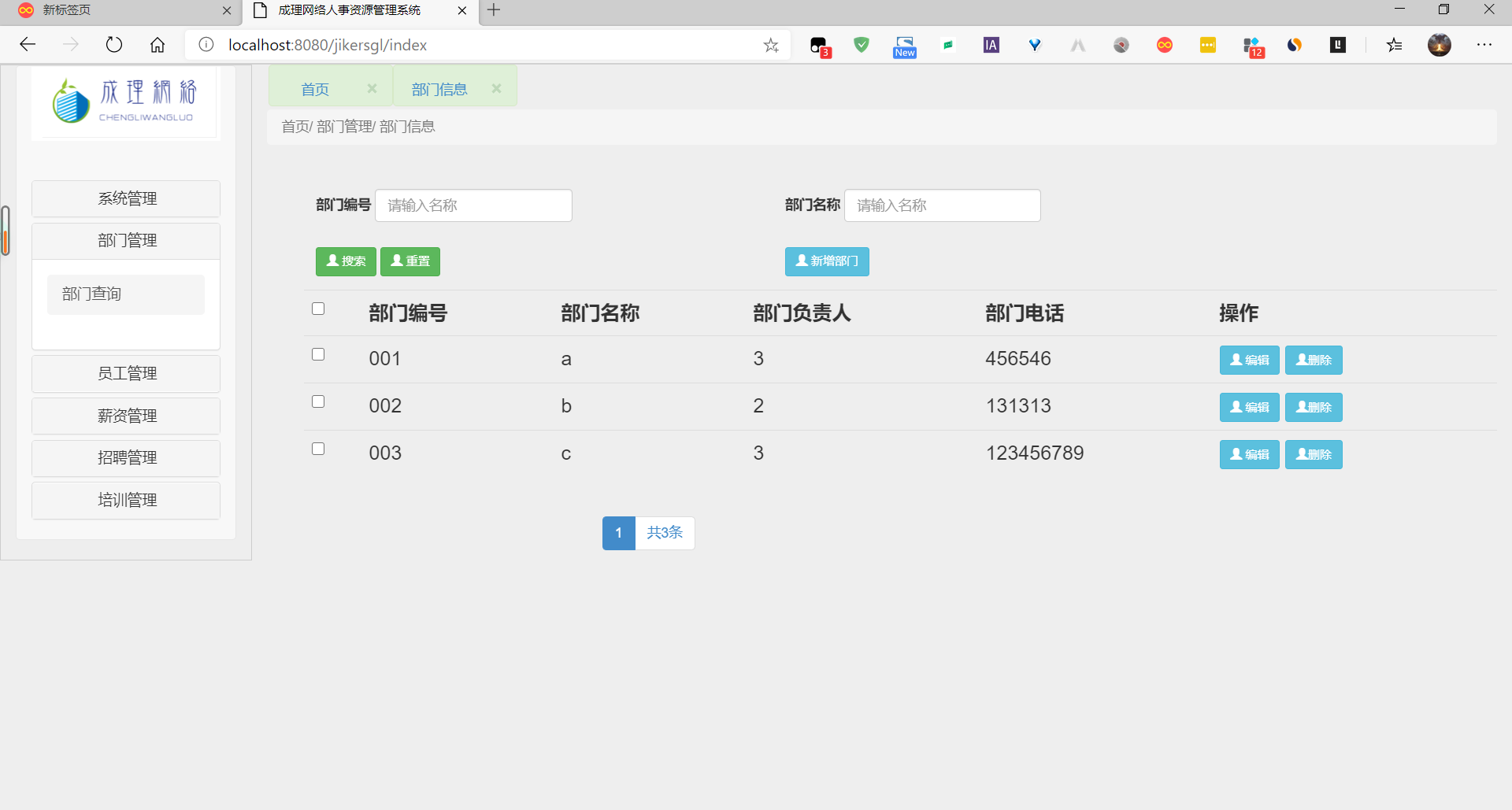


图6-4部门信息

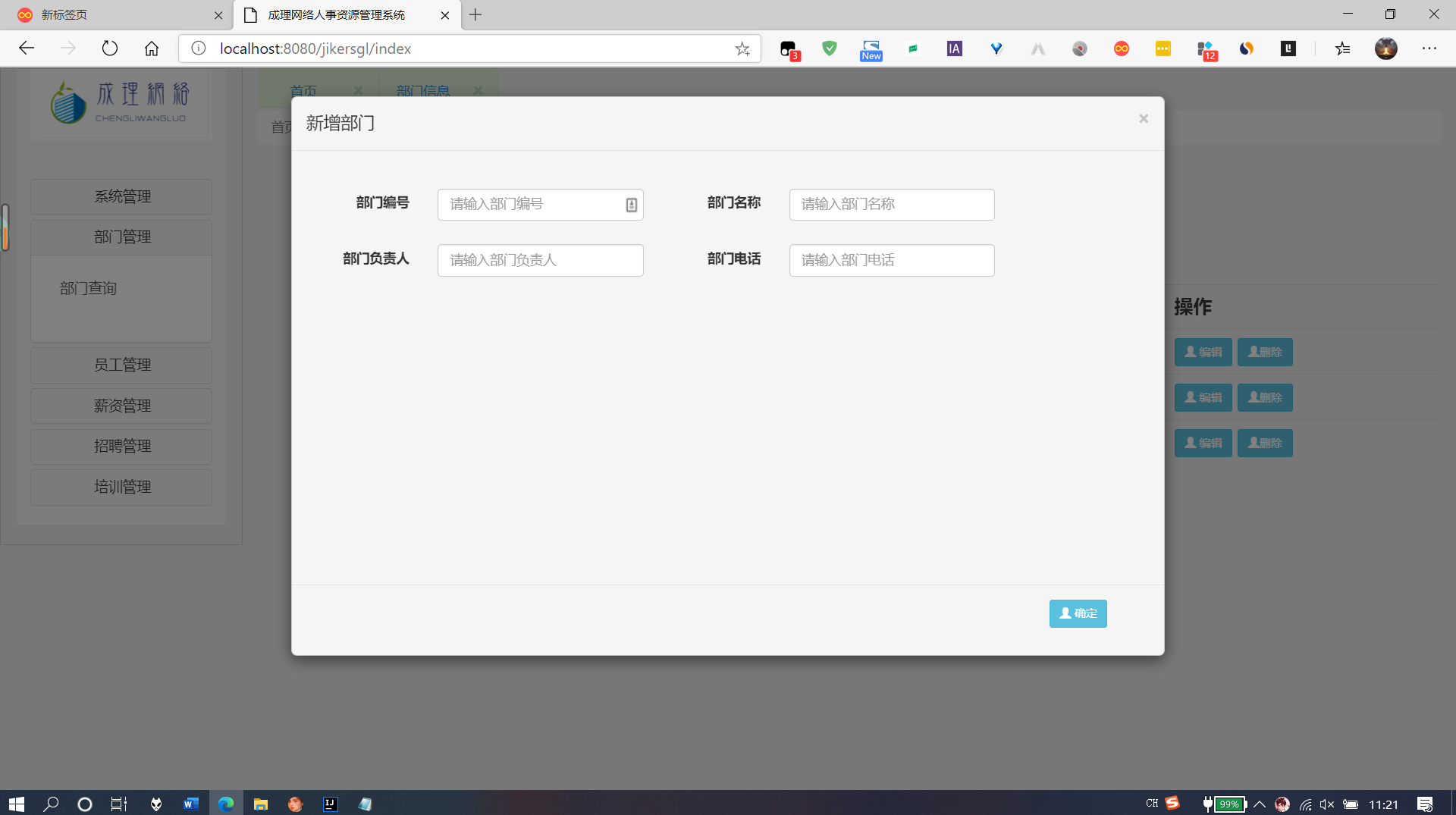


图6-5部门添加

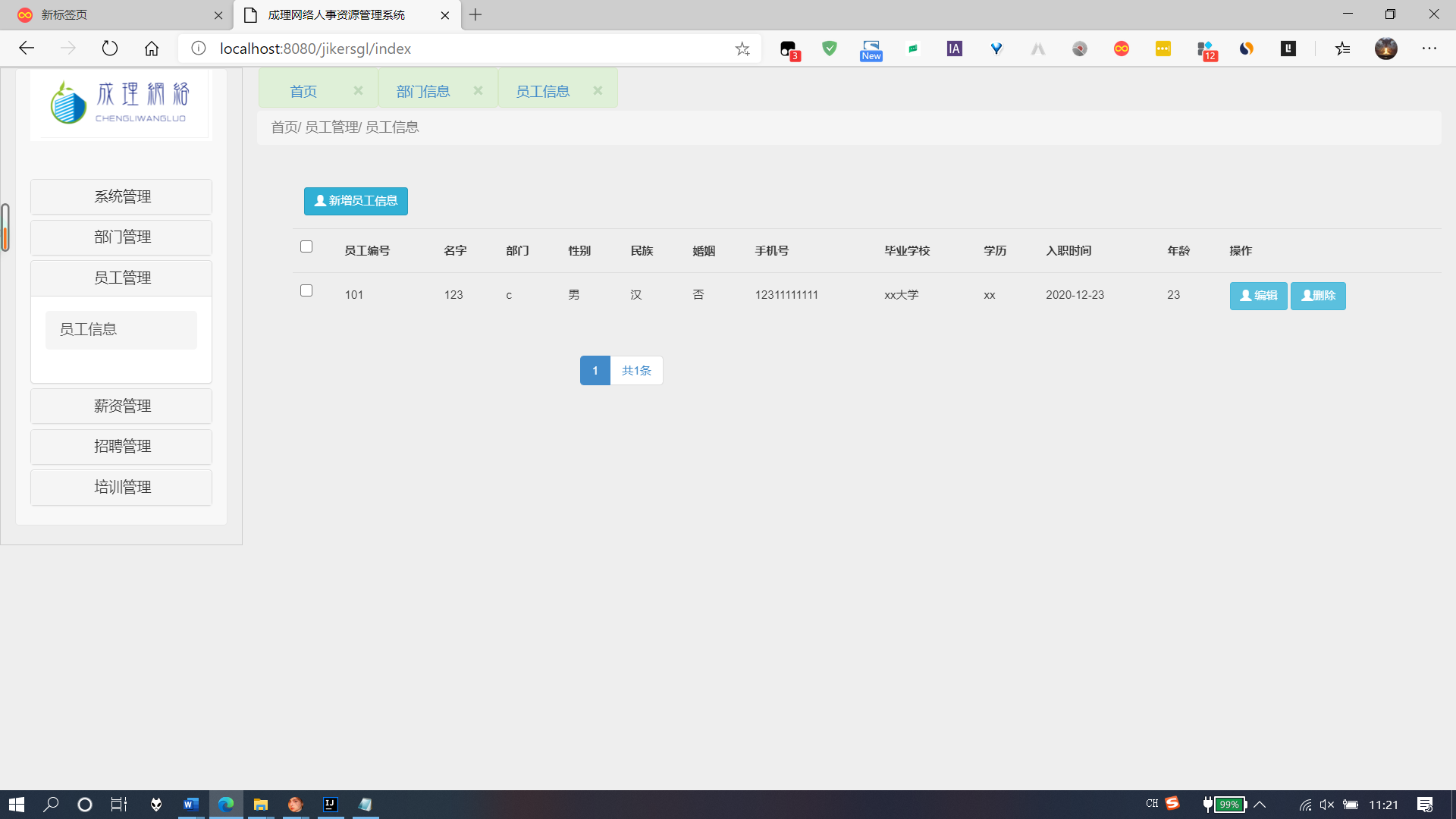


图6-6员工信息

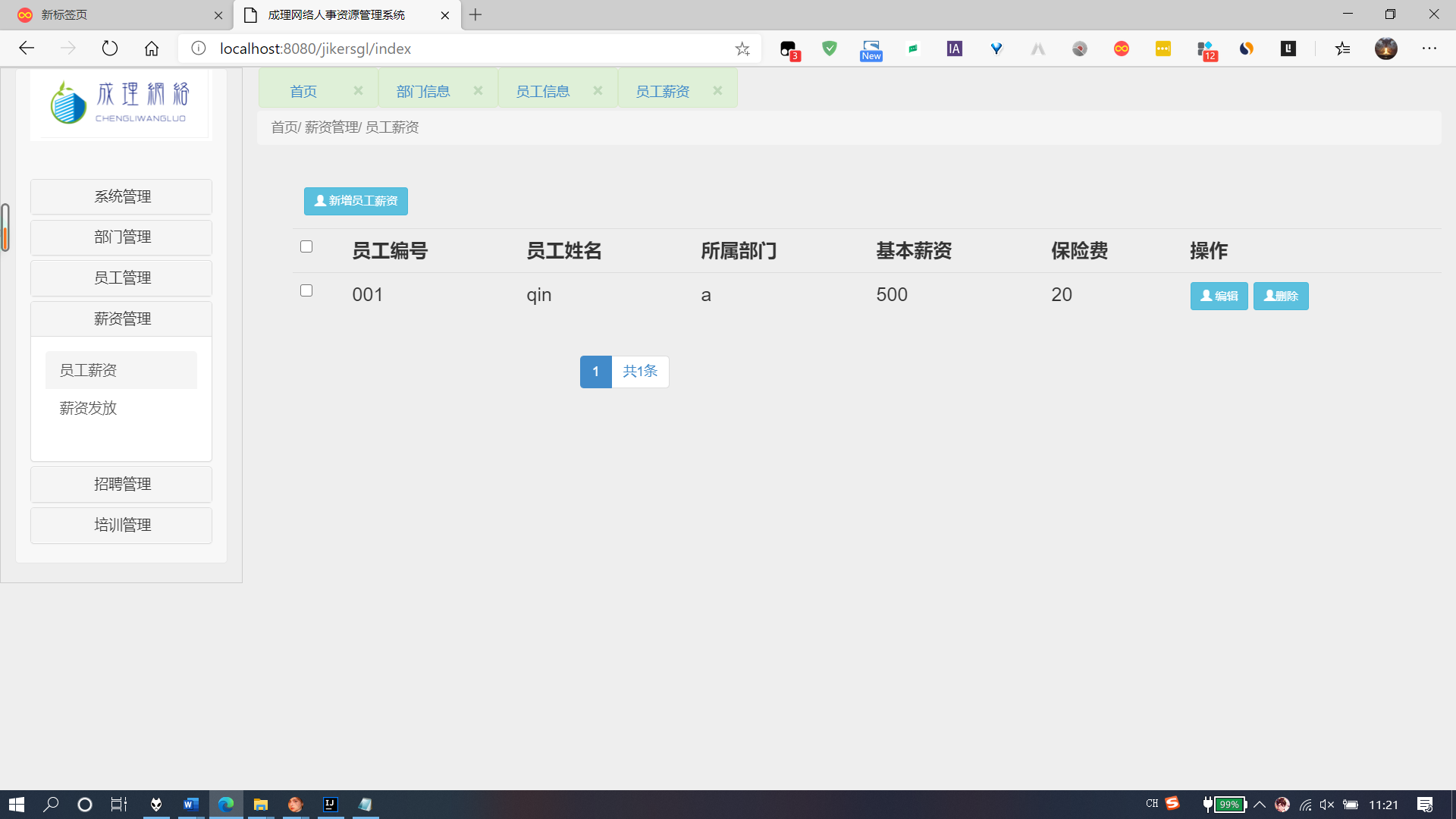


图6-7员工薪资

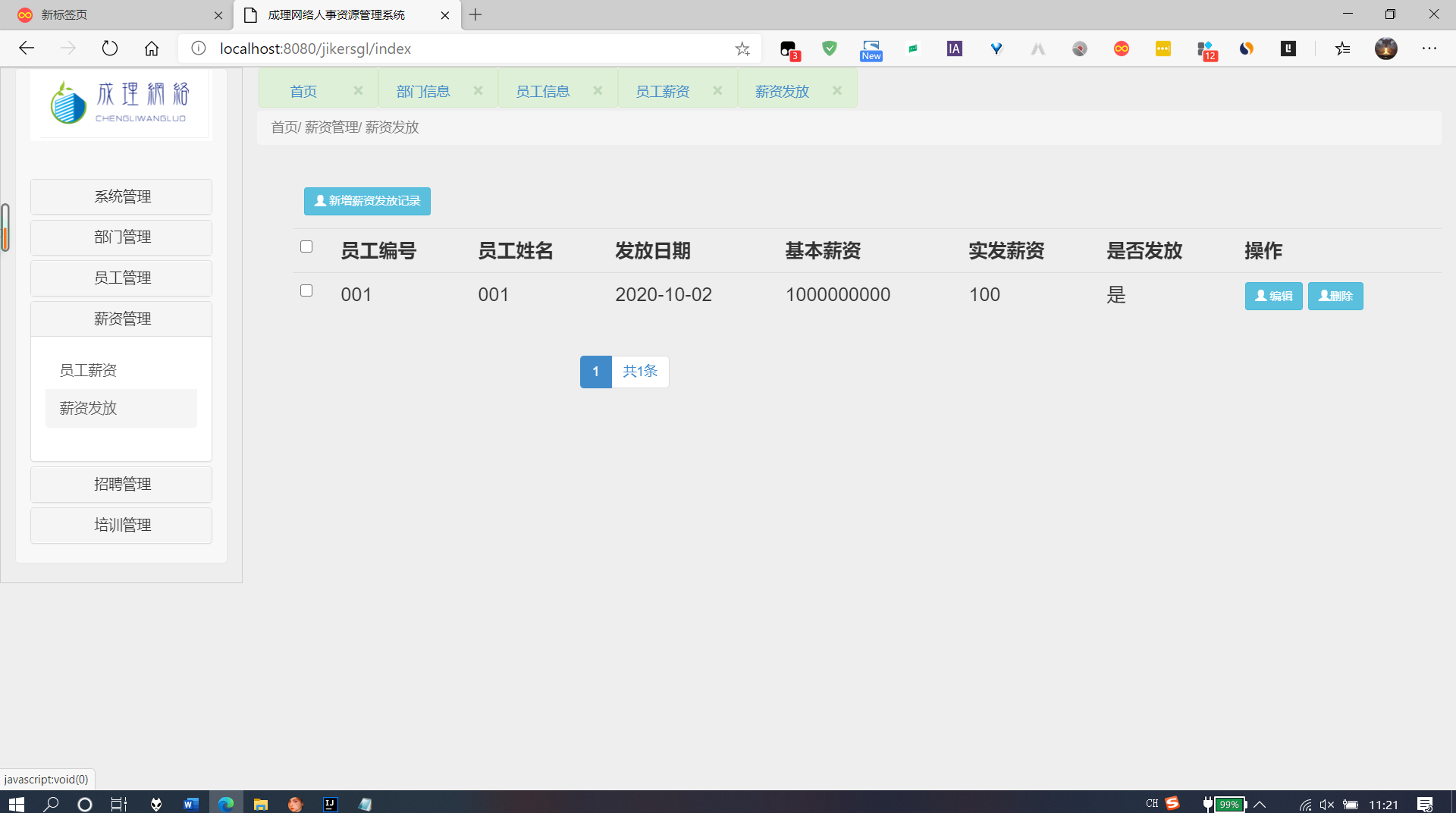


图6-8薪资发放

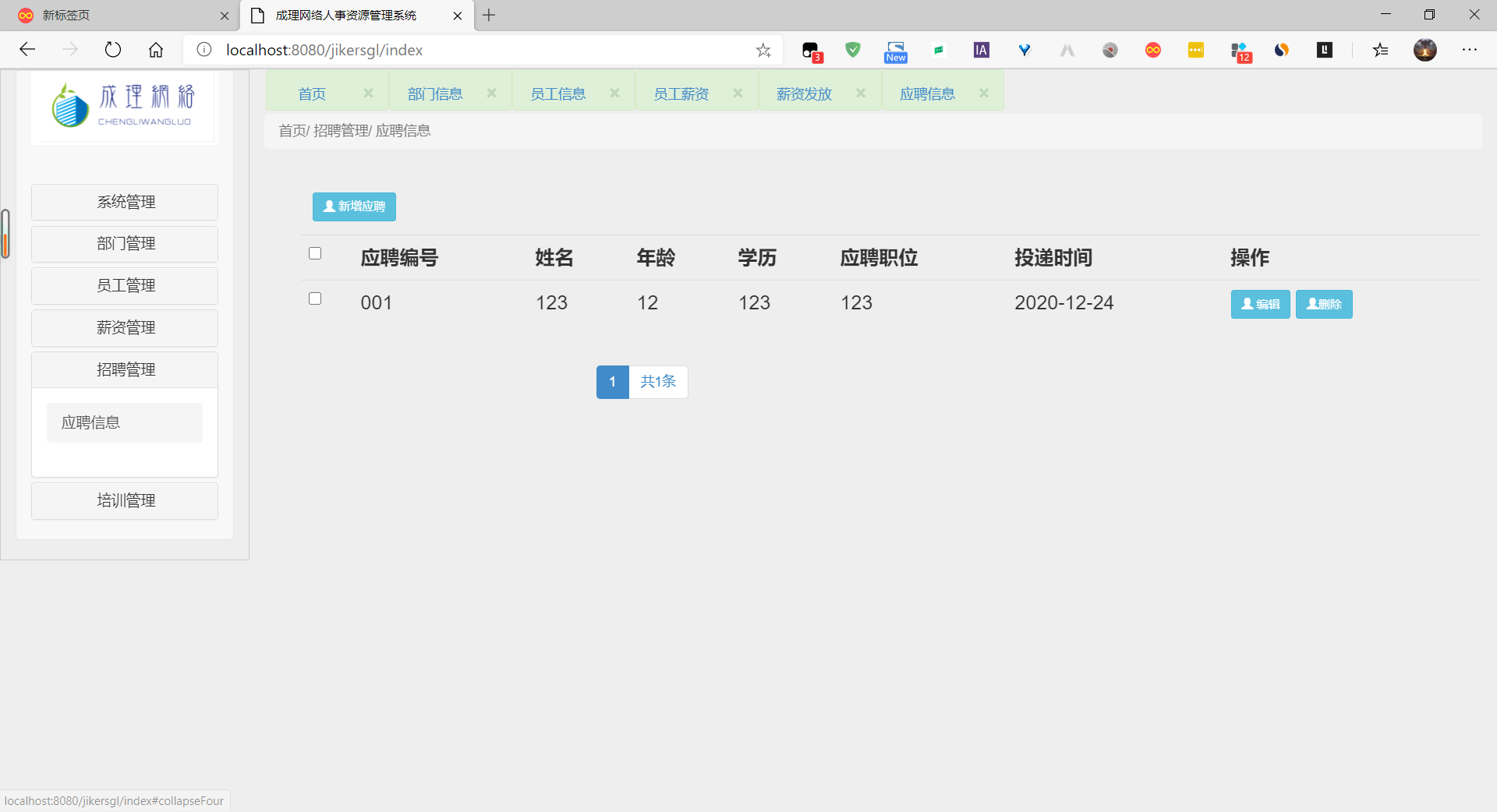


图6-9应聘信息

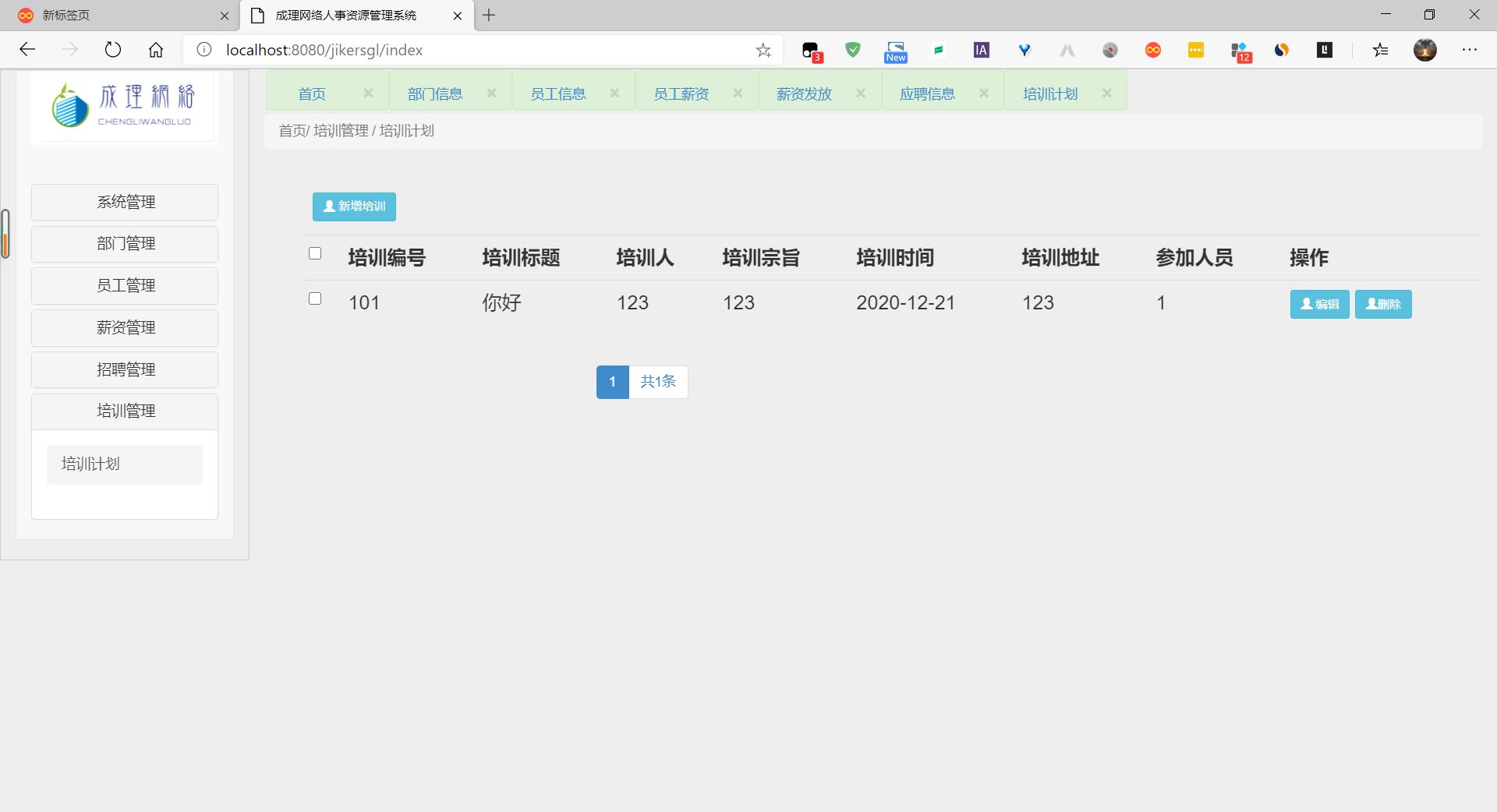
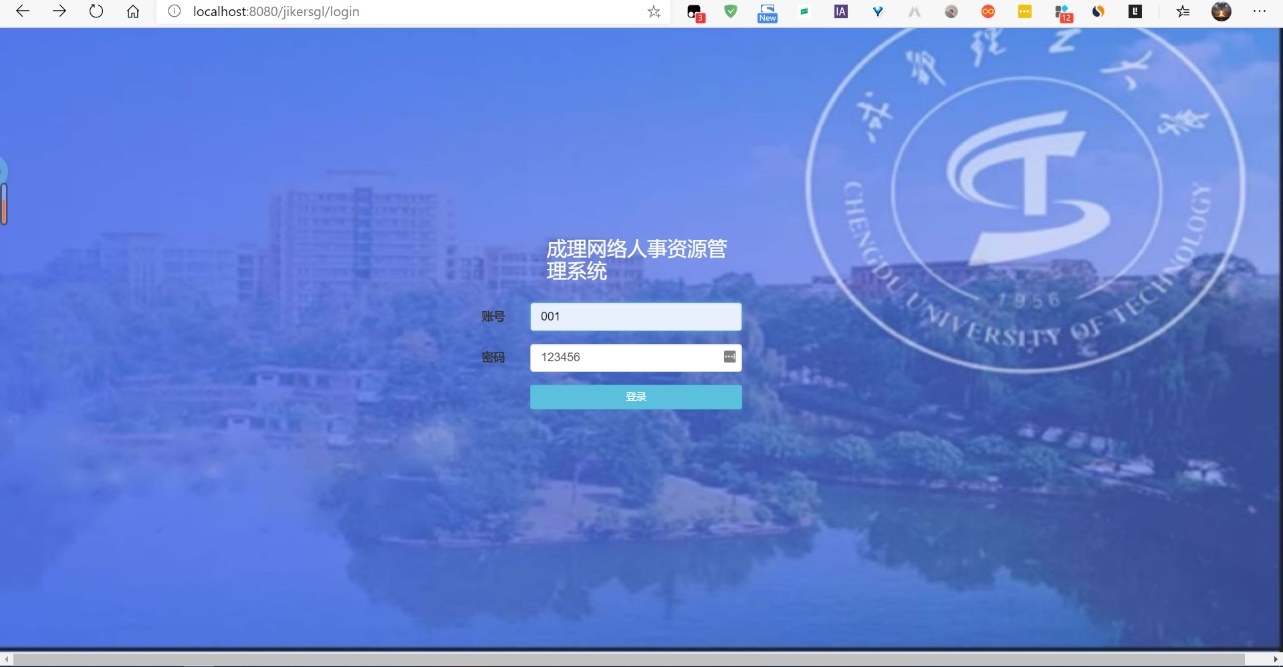


图6-10培训计划

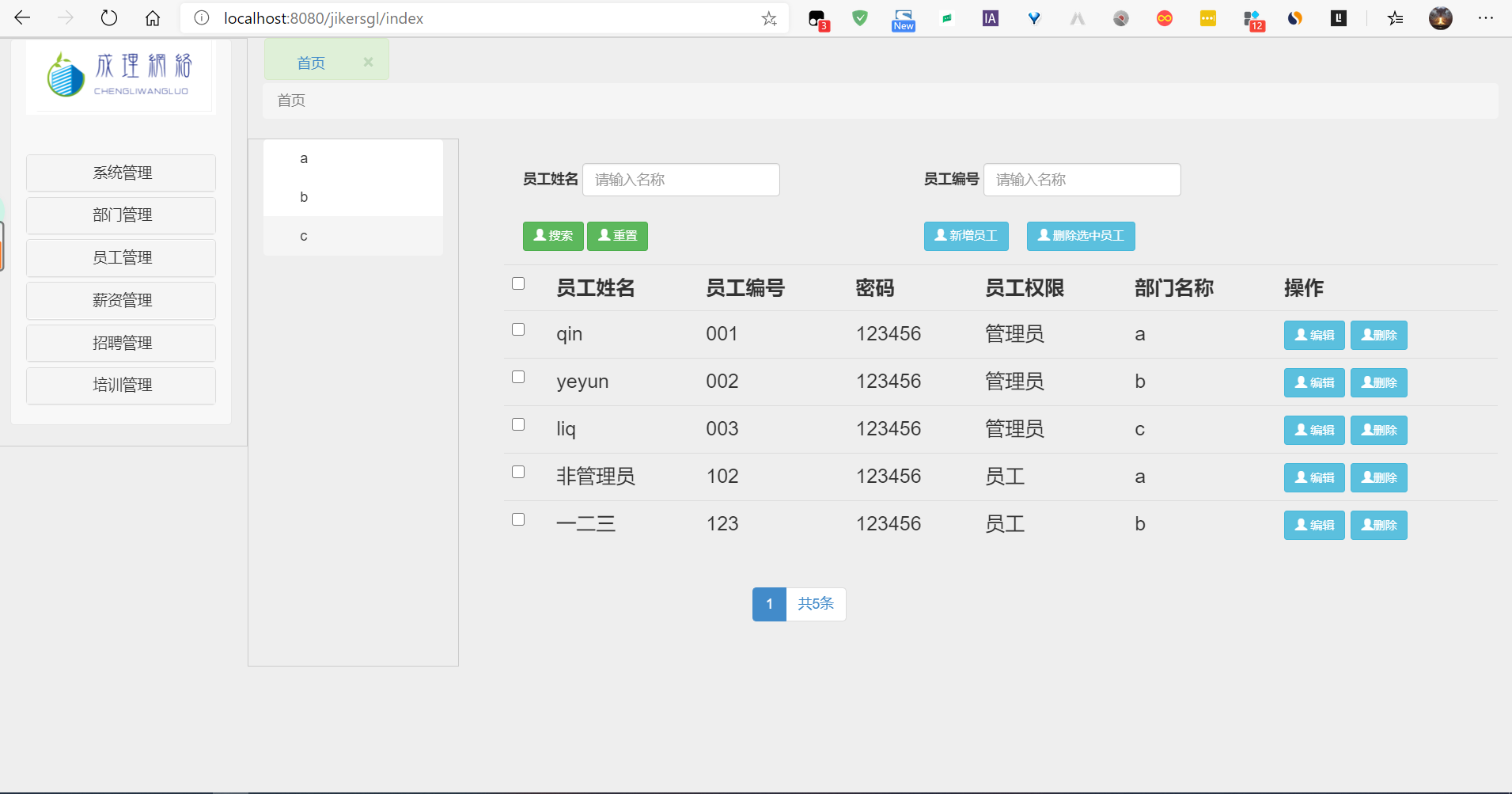
**6.2登录体验**

**6.2.1 管理员登录**

**◆** 输入用户名：001，密码：123456，点击登录

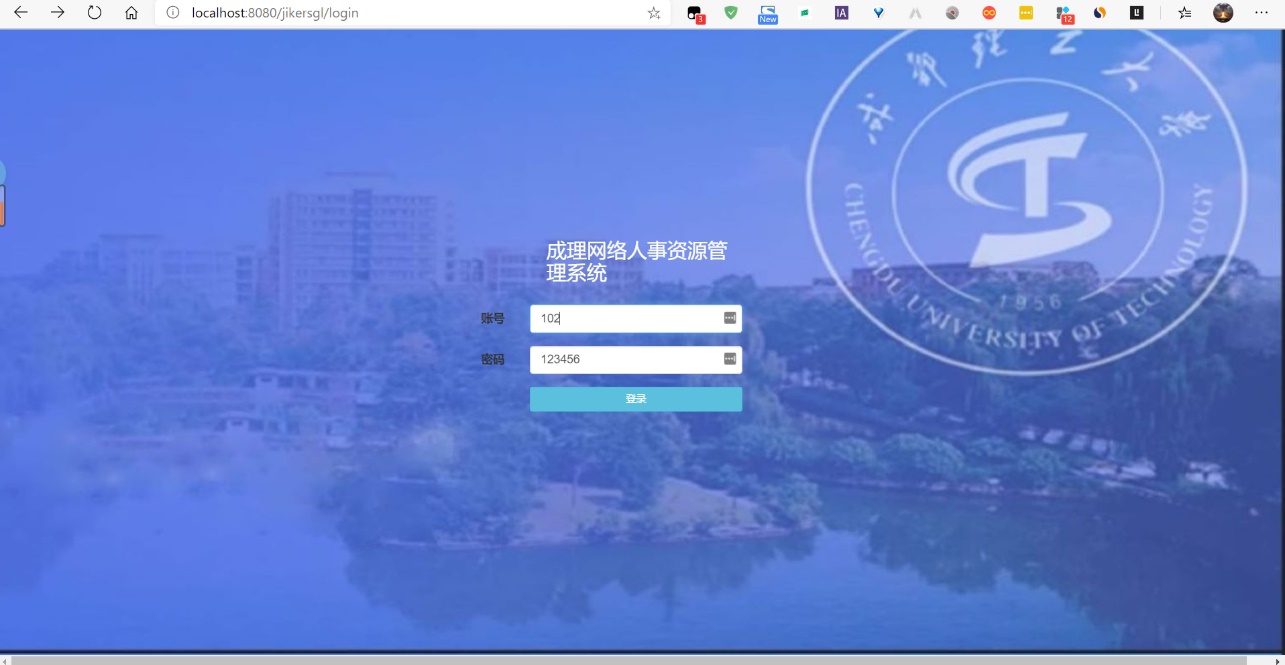
**图6-****11管理员登录**

* 登录成功后进入管理员主界面，如下图：



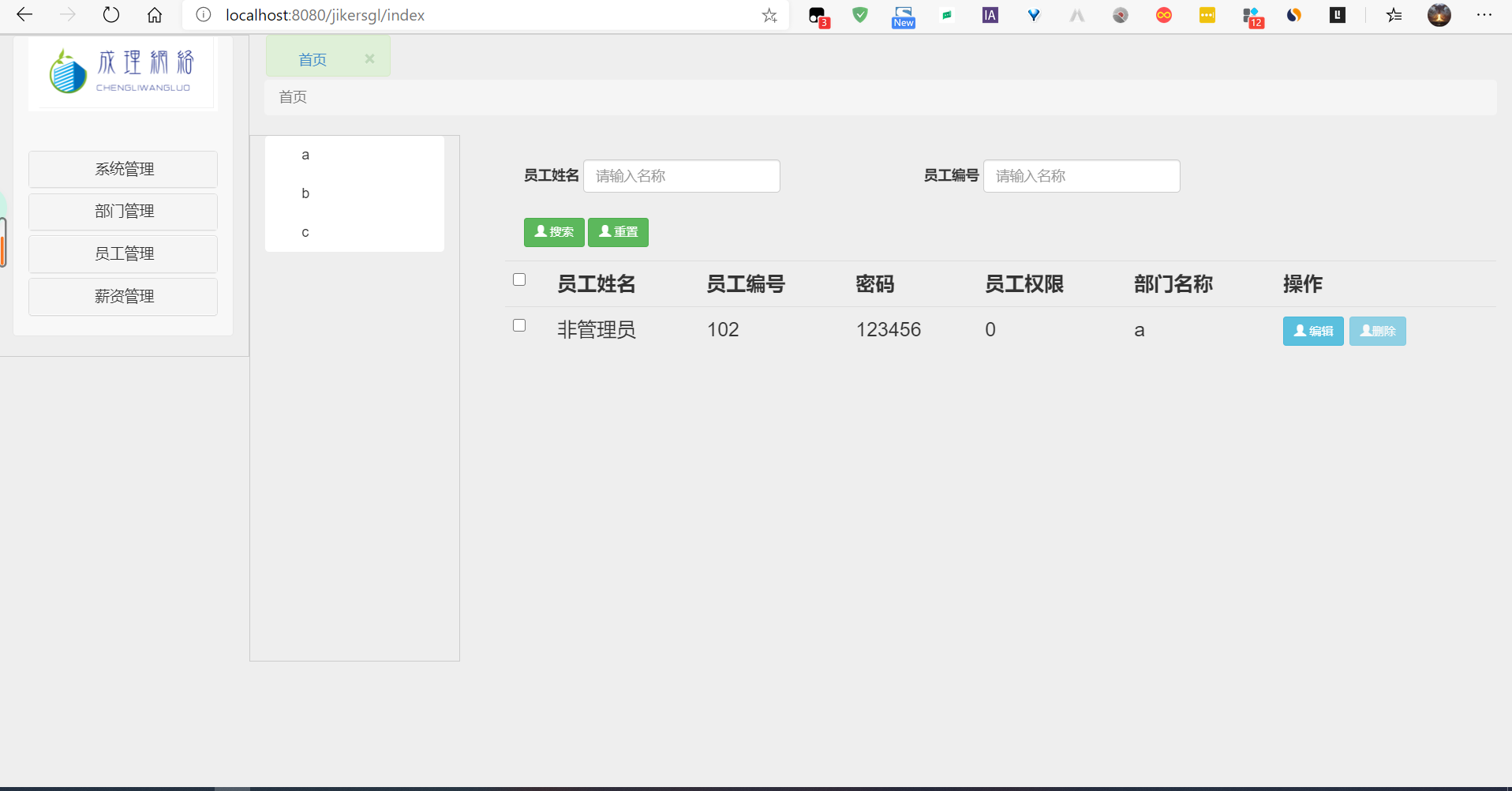
**图6-12管理员界面**

**6.2.2 员工登录**

**◆** 输入用户名：102，密码：123456，点击登录

**图6-13员工登录**

**◆** 登录成功后进入员工主界面，如下图：



**图6-14员工界面**

# 项目研发计划

## 7.1 计划图解



**图7-1**

**附录1：变更纪事**

**变更纪录**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **变更时间** | **变更描述** | **变更事由** | **批准者** | **变更者** |
| 1.1 | 需求分析修改 | 格式错误 | 李强 | 李强 |
| 1.3 | 概要设计修改 | 设计页面修改 | 李强 | 秦帅 |
| 1.6 | 详细设计修改 | 代码实现修改 | 李强 | 叶赟希 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |
| --- | --- |
| **学生实训 心得** | 这次实训学习了SSM框架，我们做了一个成理网络人事管理系统SSM项目，本次实训中我负责登录页面以及员工管理两大模块，巩固了我脑海中有关spring、springMVC、MyBatis的知识，并且锻炼了自己做项目的能力以及和队友配合、一起分工合作这些方面的能力，加强了自己团队协作、协同配合队友一起完成整个项目各大功能模块的能力，最后感谢老师的教导，谢谢老师的辛苦付出！  学生（签名）：  2021 年 1 月 9 日 |
| **学生实训 心得** | 这次实训是做的ssm项目，我们小组做的是一个成理网络人事管理系统，我负责的是部门管理模块、招聘信息模块、培训计划模块，在这次实训中发现了很多问题，也学到了很多，经过实训大概了解了ssm框架的使用流程，有了框架之后编写代码的效率提高了，因为框架封装了普通项目中程序员需要重复书写的代码和简化了调用过程，使页面更加美观，这次实训使我受益匪浅，实践和理论知识同样重要，在项目最后感谢老师的教导，谢谢老师的辛苦付出！  学生（签名）：  2021 年 1 月 9 日 |
| **学生实训 心得** | 这次实训做的是SSM项目，我们做了一个成理网络人事管理系统，本次实训中我负责账号管理页面以及员工薪资和薪资发放三大模块，在账号管理的界面撰写中，我通过查找资料学会了如何在spring security中创建基于数据库自定义UserDetailsService的认证服务;  我的感想是先想好代码逻辑，再去布局代码架构，最后在进行细节优化。通过完成这个SSM项目，我深入理解了JVM和MYSQL。  非常感谢老师的辛勤讲解和耐心的报告指导！  学生（签名）：  2021 年 1 月 9 日 |
| **诚信承诺书** | 本人郑重声明所呈交的课程报告是本人在指导教师指导下进行的研究工作及取得的研究成果。据我所知，除了文中特别加以标注的地方外，论文中不包含其他人已经发表或撰写过的研究成果。与我一同工作的同学对本文研究所做的贡献均已在报告中作了明确的说明并表示谢意。  学生（签名）： |
| **指导**  **教师**  **评语** | 成绩评定：  指导教师（签名）：  年 月 日 |