成理网络人事管理系统

**详 细 设 计 说 明 书**

**成员：李强201813161327**

**秦帅201813161315**

**叶赟希201812030131**

目录

[一、引言](#_Toc10368)

[1.1 编写目的：](#_Toc22704)

[1.2项目背景：](#_Toc24086)

[1.3 参考资料：](#_Toc20967)

[二、数据库设计](#_Toc8)

[2.1 数据库表列表](#_Toc22301)

[2.2逻辑结构设计：](#_Toc5255)

[2.3物理结构设计：](#_Toc17025)

[2.4 数据表代码记录](#_Toc11069)

[三、程序模块描述](#_Toc29616)

[3.1 登录模块](#_Toc16910)

3.[2.系统管理模块](#_Toc4286)

[3.3 个人信息模块](#_Toc30584)

[3.4 员工薪资模块](#_Toc25550)

[3.5 薪资发放模块](#_Toc29132)

[四.登录体验](#_Toc1472)

[4.1 管理员登录](#_Toc11084)

[4.2 员工登录](#_Toc3643)

**一、引言**

人事管理系统是针对企业内部人事管理设计，分角色实现对公司系统及各部门员工的增、删、改、查以及对员工薪资招聘培训的管理。

* 1. **编写目的：**

在系统需求分析的基础上，对需求分析中产生的功能模块进行过程描述，设计功能模块的内部细节，包括算法和详细数据结构，为编写源代码提供必要的说明，并作为测试和维护的参考文档。

详细设计说明书的预期用户为所有与此项目相关的决策人员、开发人员和测试人员。

## 1.2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。

## 1.3 参考资料：

详细设计说明书模板资料

《人事管理系统需求分析说明书》

**二、数据库设计**

数据库名称:rsgl.sql

## 2.1 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 |

## 2.2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 |

## 2.3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** |

## 2.4 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';

1. **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';

1. **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;

1. **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;

1. **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;

1. **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;

1. **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`),

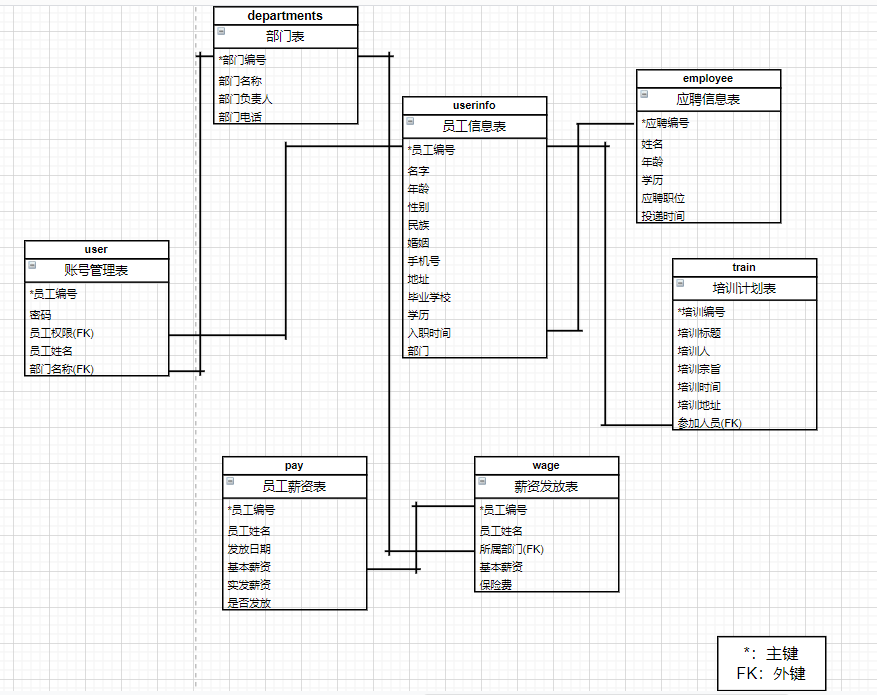
UNIQUE KEY `username` (`username`,`account`),

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

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

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

**2.5数据库ER图**



**三、程序模块描述**

## 3.1 登录模块

**1. 流程图：**

职工

输入用户名、密码

登录

登录 成功

普通用户主界面

管理员主界面

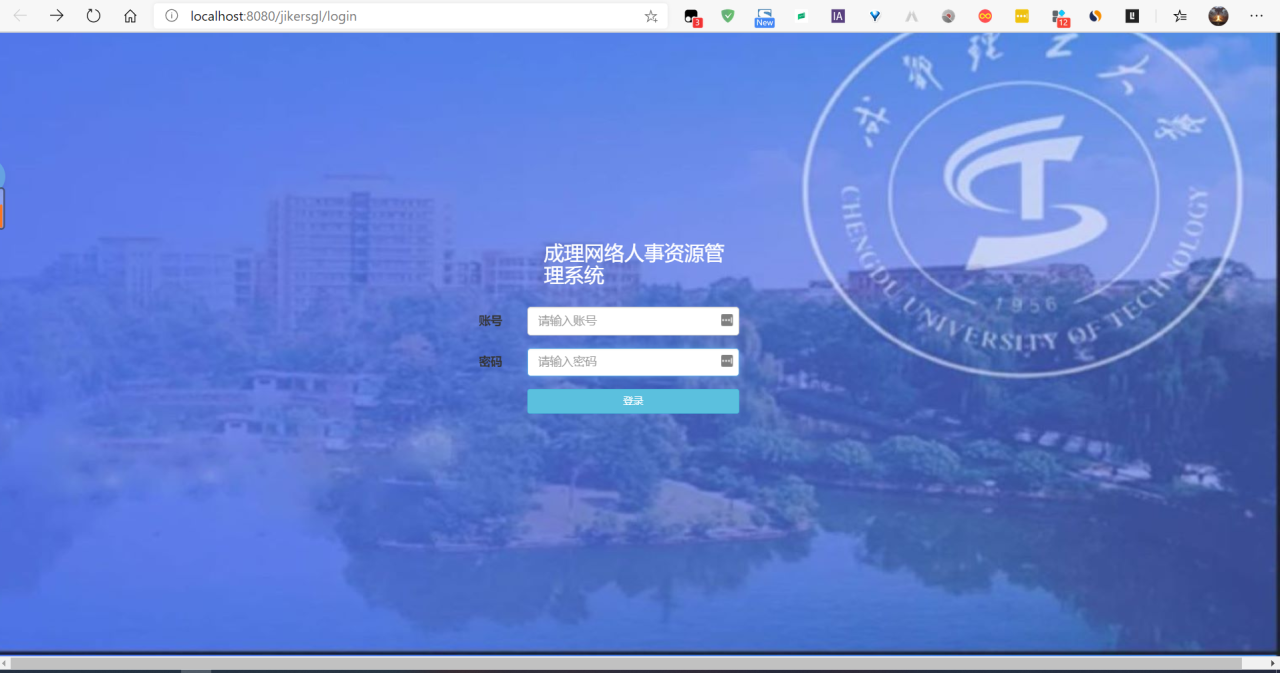
* 职工输入用户名、密码，会根据用户是否为管理员而加载出管理员/普通用户界面

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！”。

1. **界面设计及代码如下：**



* **用到的控件字段名、方法及变量有：**
* **控件：**

用户名文本框名称：账号

密码文本框名称：密码

登录标签名称：登录

* **变量:**

input type="text" class="form-control" name="username" id="username"//判断变量username是否正确

input type="text" class="form-control" name="password"//判断变量password是否正确

* **调用方法：**

uc.GetUser(user); //核对普通用户的用户名、密码；

uc.GetAdm(adm); //核对管理员的用户名、密码；

* **“登录”标签的单击事件：**
* <div class="row">  
  <form class="form-horizontal" method="post" role="form"  
  action="<c:url value='check\_action' />">  
    
  <div class="form-group">  
    
  <label  class="col-sm-2 control-label"></label>  
  <div class="col-lg-3 col-sm-7" style="left:7%;color:White">  
  ${error}  
  </div>  
    
  </div>  
  <div class="form-group">  
    
  <label  class="col-sm-2 control-label">账号</label>  
  <div class="col-lg-3 col-sm-7">  
  <input type="text" class="form-control" name="username" id="username"  
  placeholder="请输入账号">  
  </div>  
  </div>  
  <div class="form-group">  
  <label  class="col-sm-2 control-label">密码</label>  
  <div class="col-lg-3 col-sm-7">  
  <input type="text" class="form-control" name="password"  
  placeholder="请输入密码">  
  </div>  
  </div>  
  <div class="form-group">  
  <label  class="col-sm-2 control-label"></label>  
  <div class="col-lg-3 col-sm-7">  
  <button type="submit" class="btn btn-info btn-sm btn-block">登录</button>  
  </div>  
  </div>  
  </form>  
  </div>**“退出”标签的单击事件：**

private void Login\_Esclab\_Click(object sender, EventArgs e)

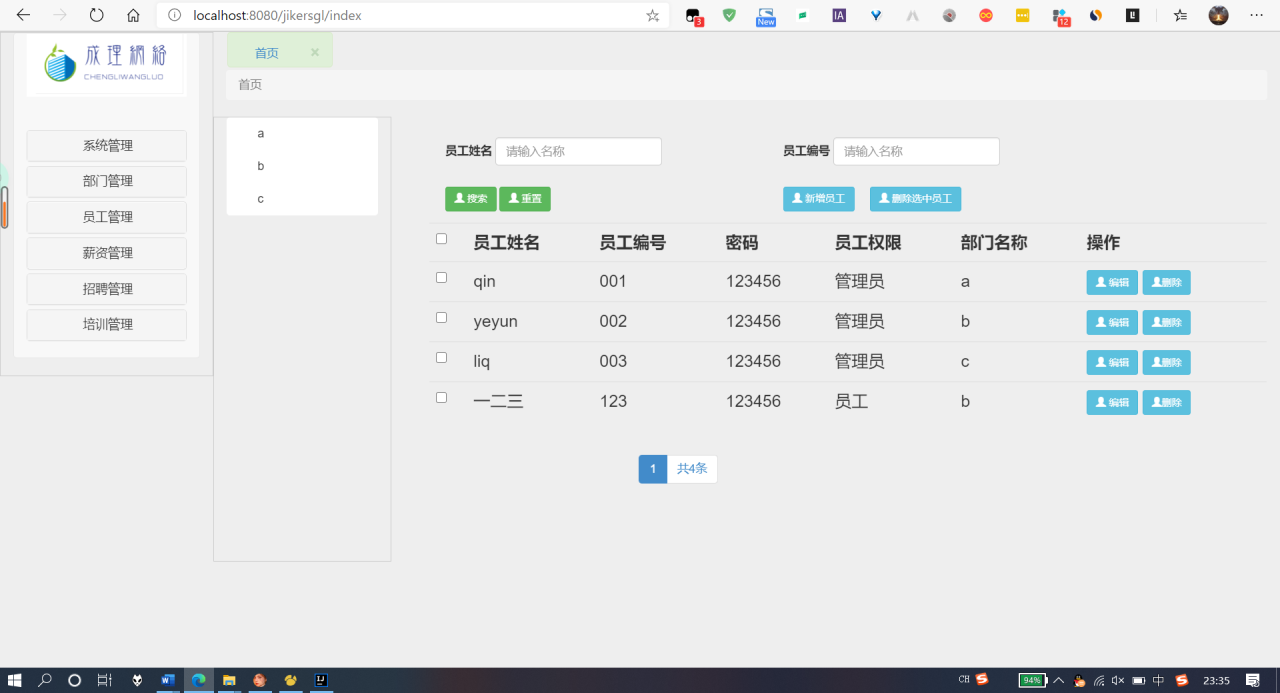
{

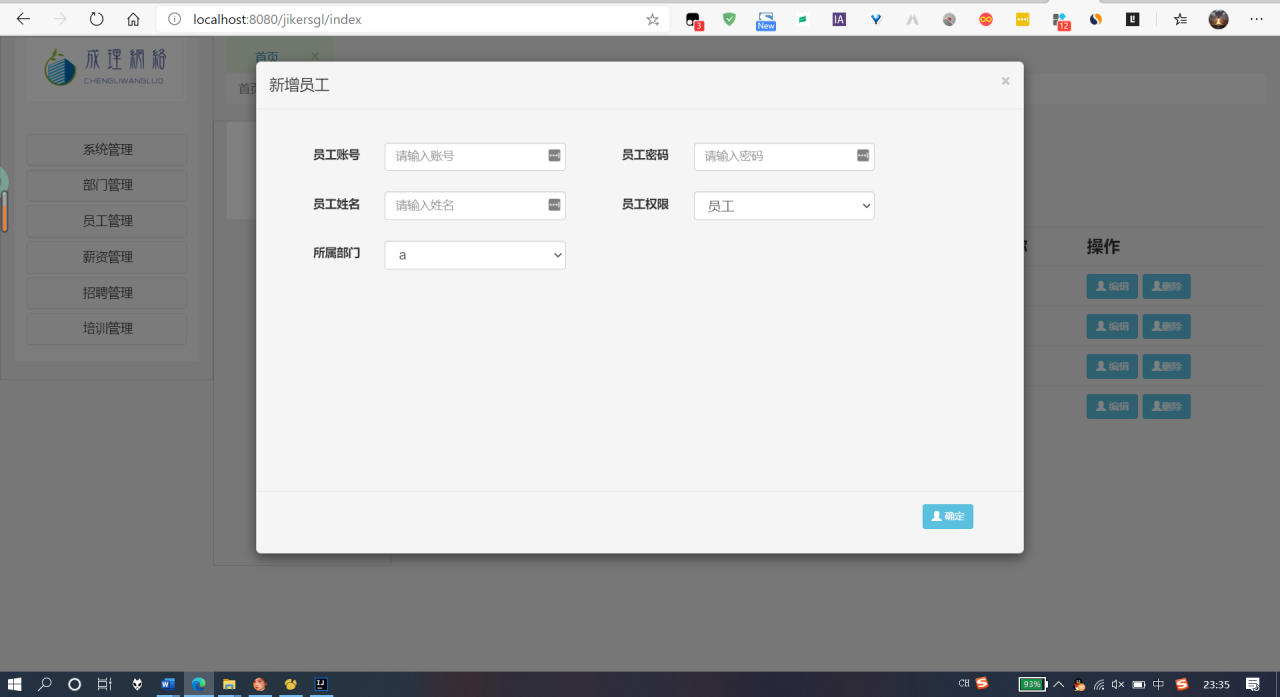
this.Close();

}

**3.2 系统管理模块**

**1.界面设计截图**





## 2.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) {

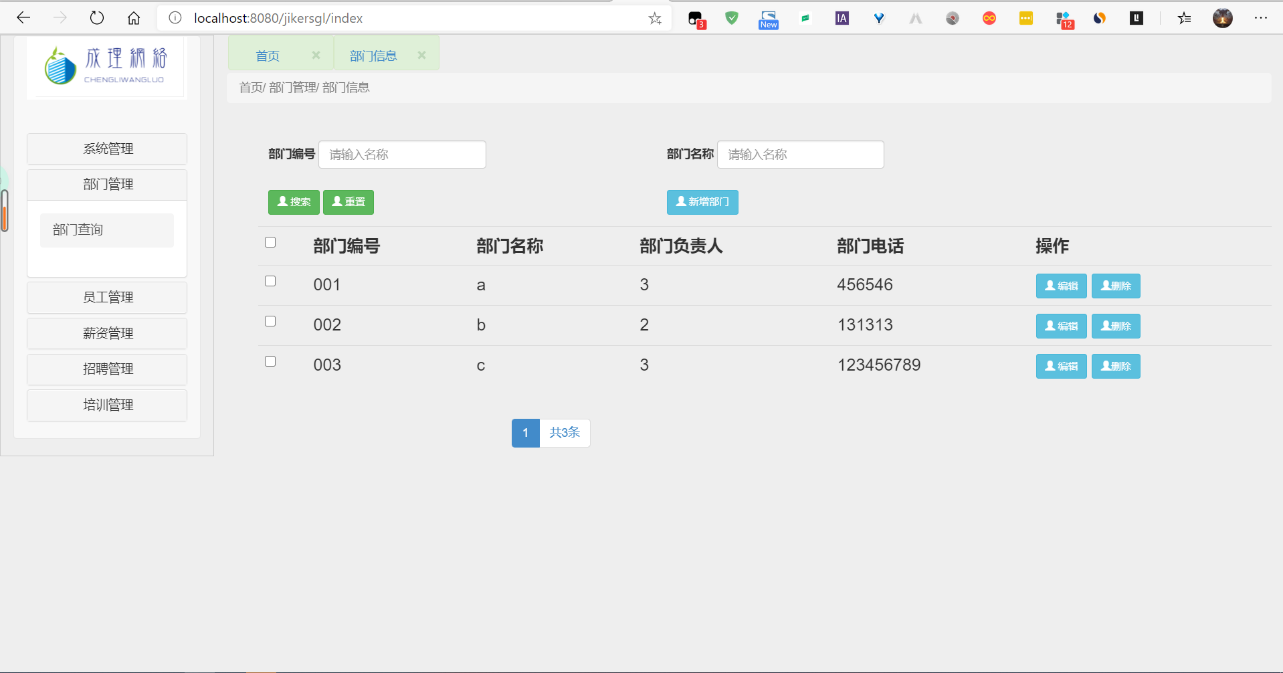
## userMapper.deleteList(deleteList);

## }

}

## 3.3 部门管理模块

**2. 界面设计截图及代码如下：**



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) {

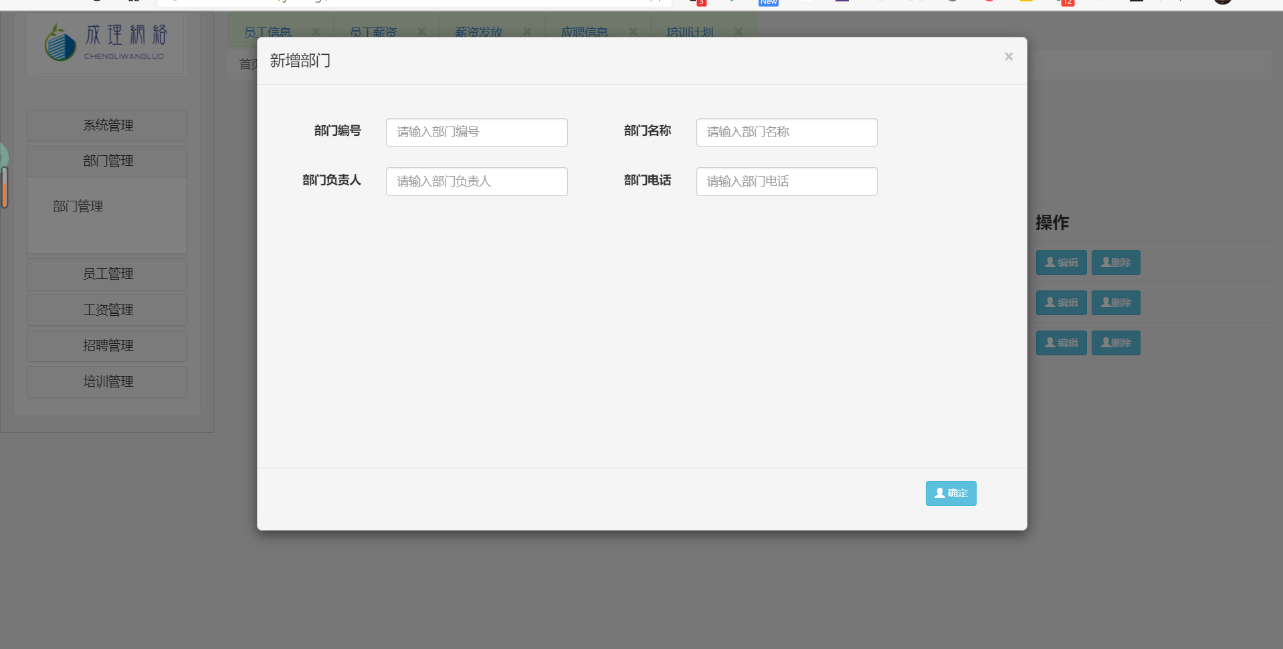
logger.error("删除失败：" + e);

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";

}

}

}

}

## 3.4 个人信息模块

**1. 流程图：**

个人信息

所有的用户登录系统

查看全部信息

修改部分信息

* 成功登录系统后所有的职工都可以对自己的全部信息进行查看、对部分信息进行修改。

**2. 界面设计截图及主要代码如下：**



***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";

}

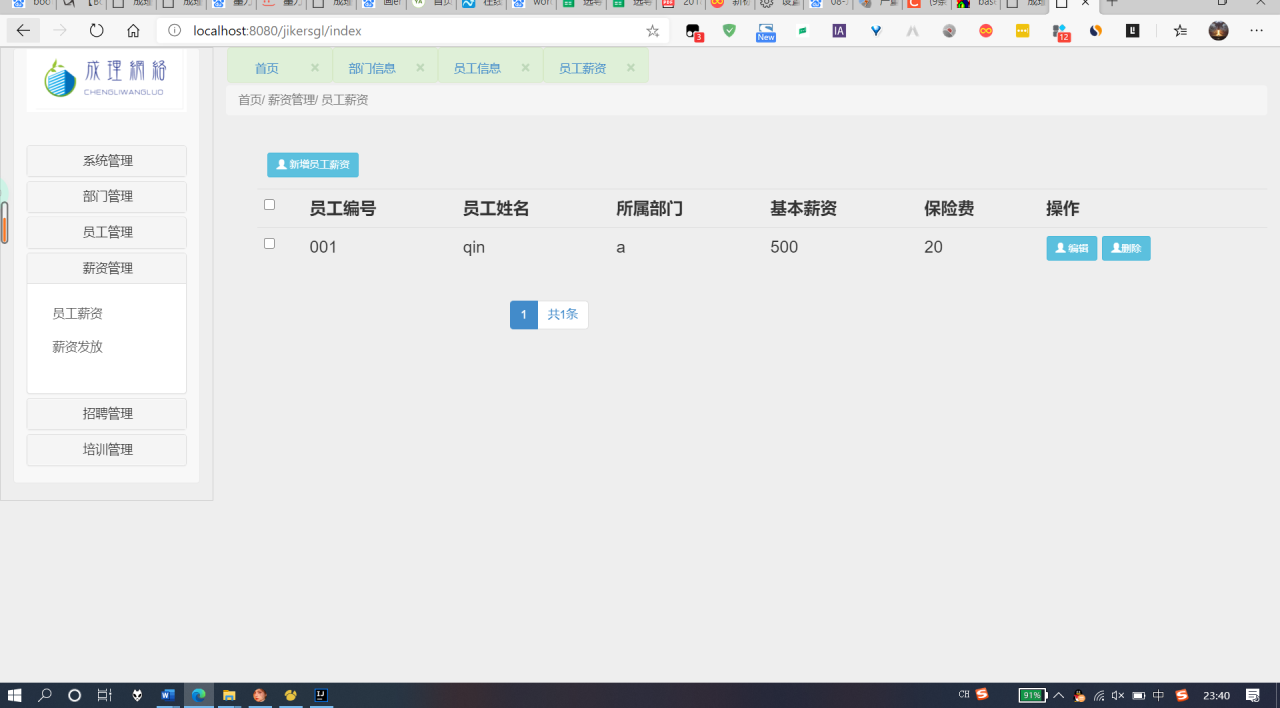
return "success";

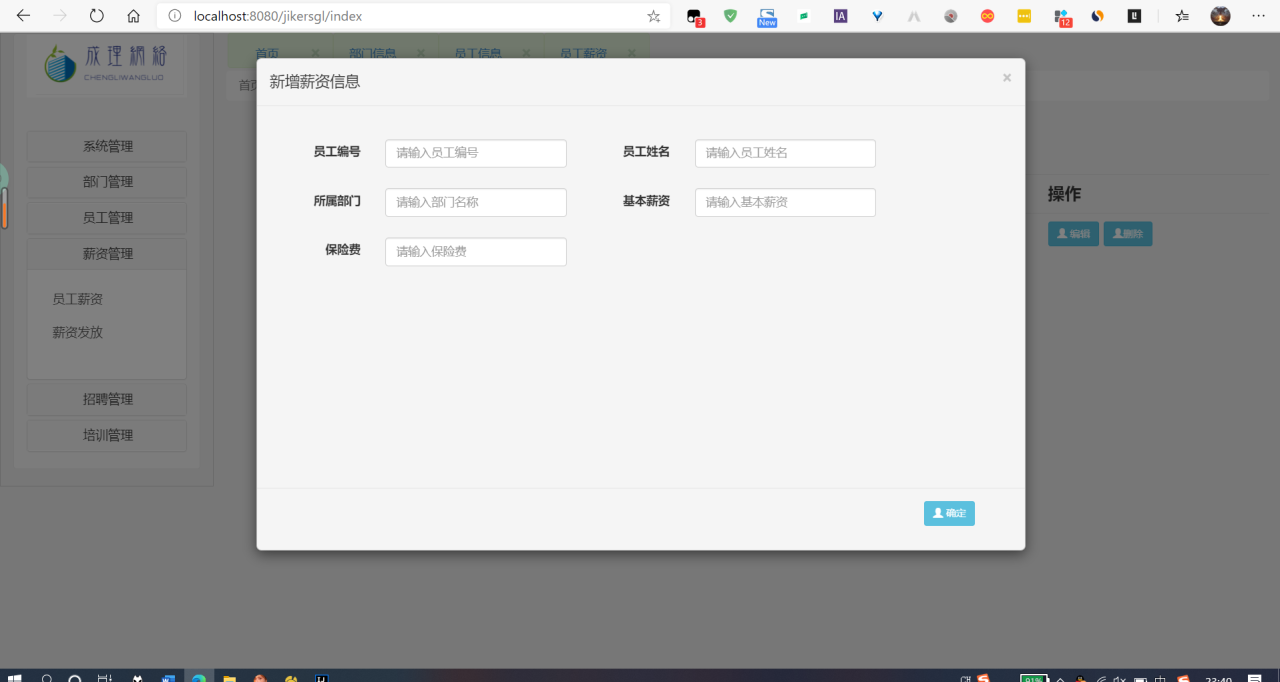
}

}

## 3.5 员工薪资模块

**1.界面设计截图**





**2. 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";

}

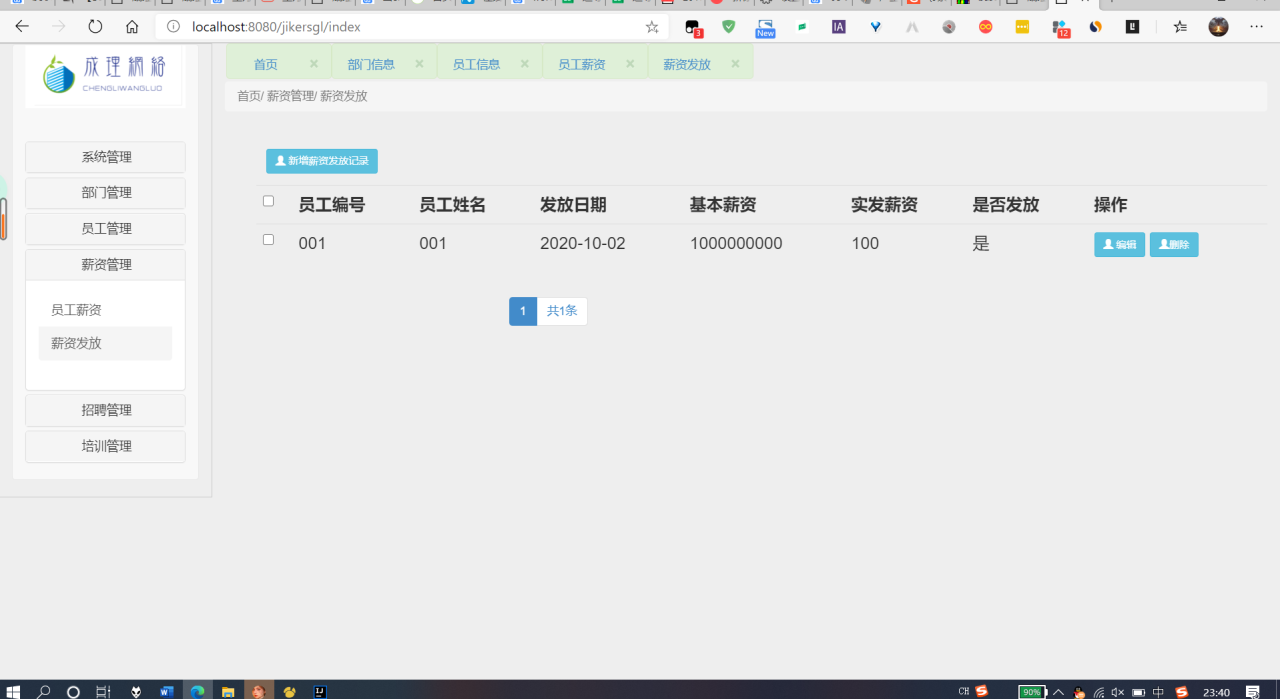
return "success";

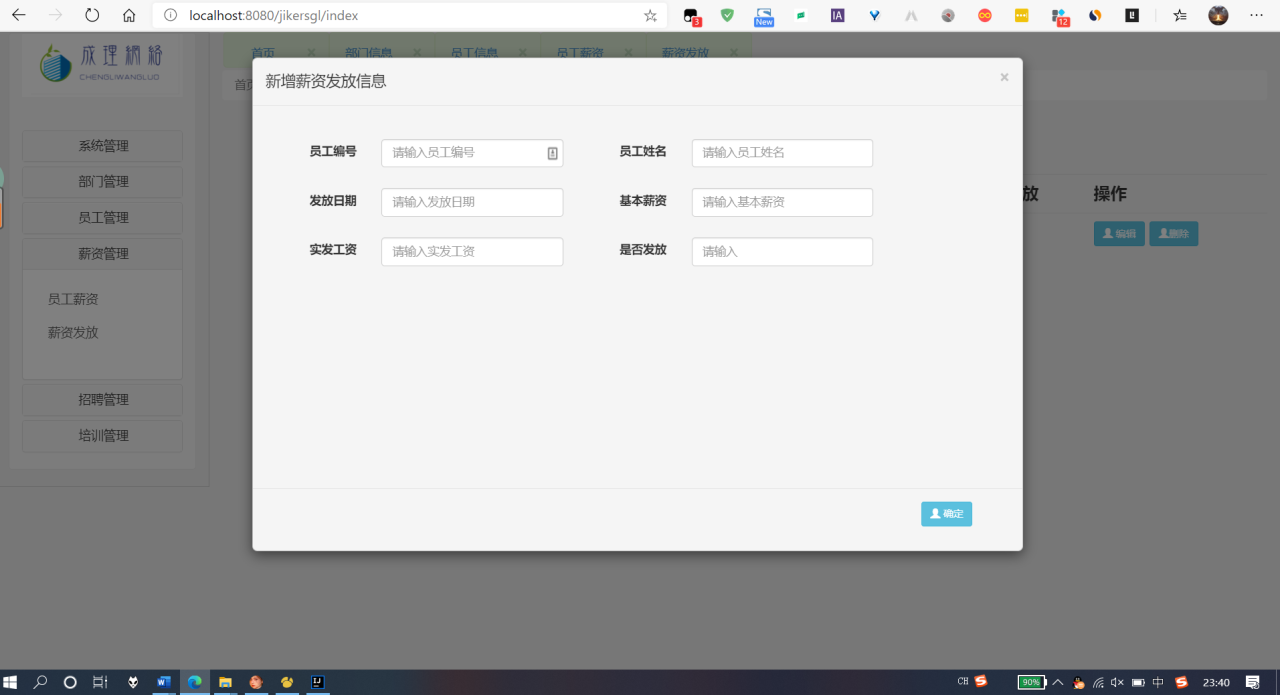
}

}

## 3.6 薪资发放模块

**1.界面设计截图**





**2. 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";

}

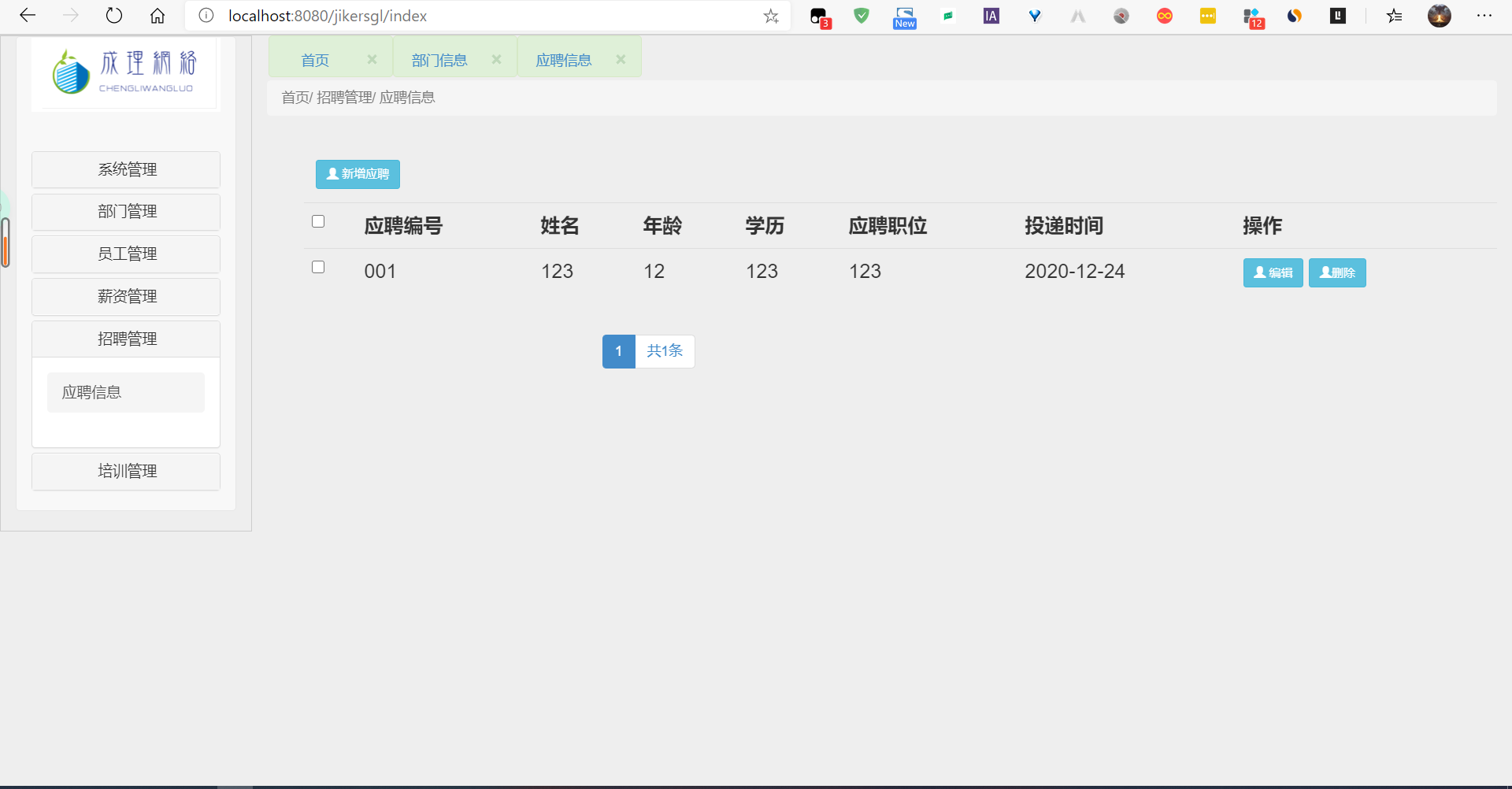
return "success";

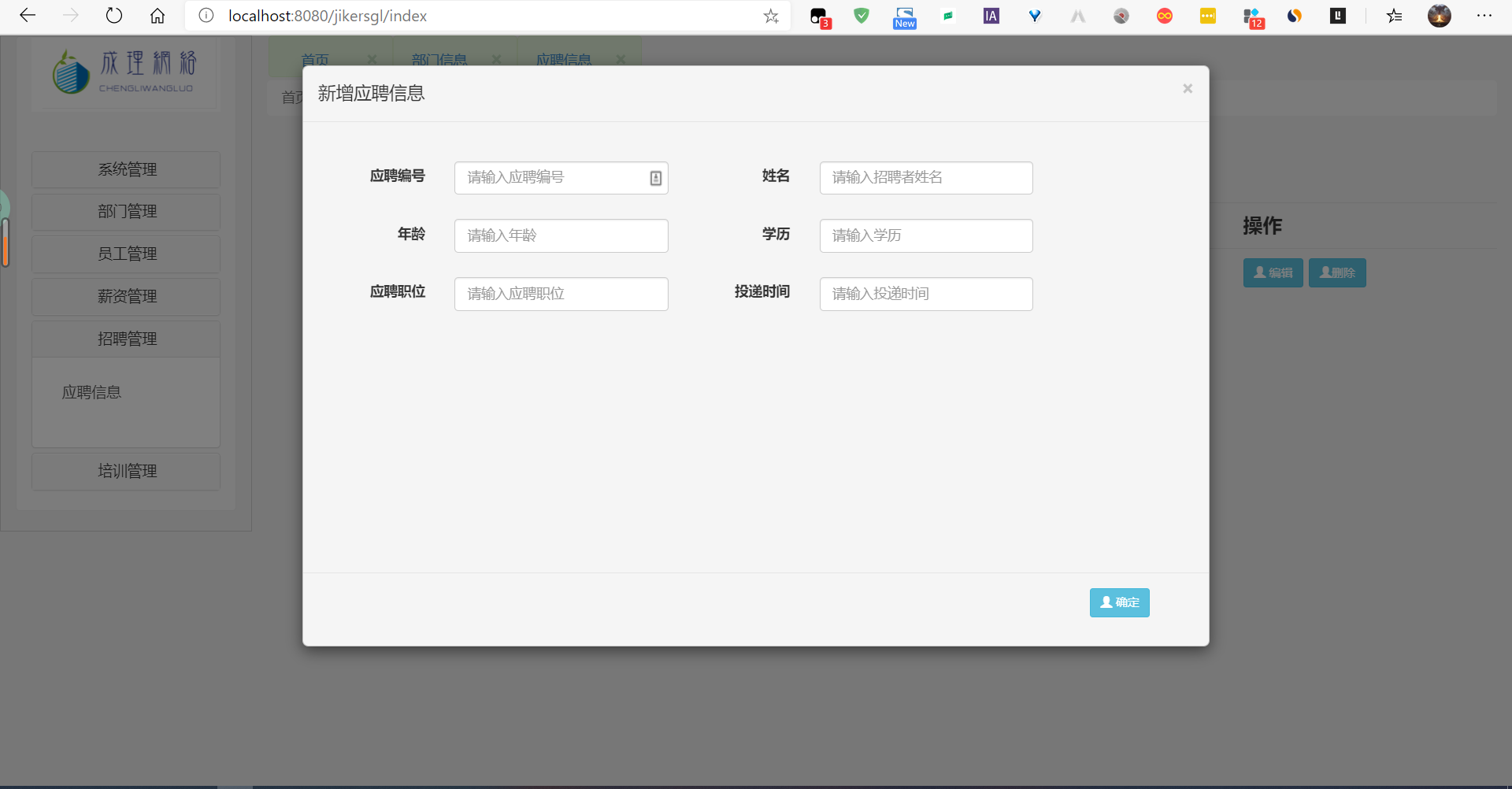
}

}

## 3.7 薪资发放模块

1.界面设计截图





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;

/\*

\* @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;

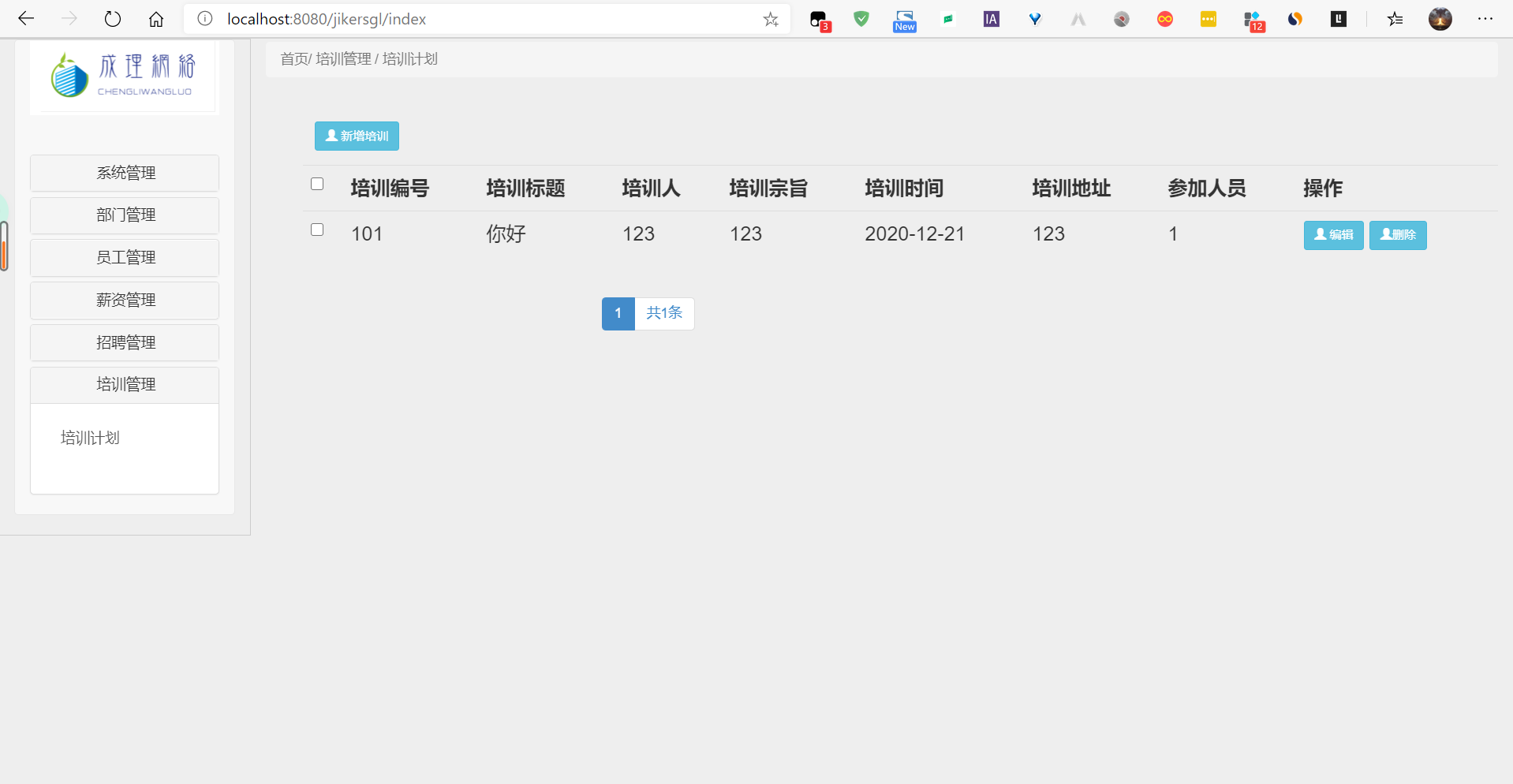
wageService.deleteByPrimaryKey(account);

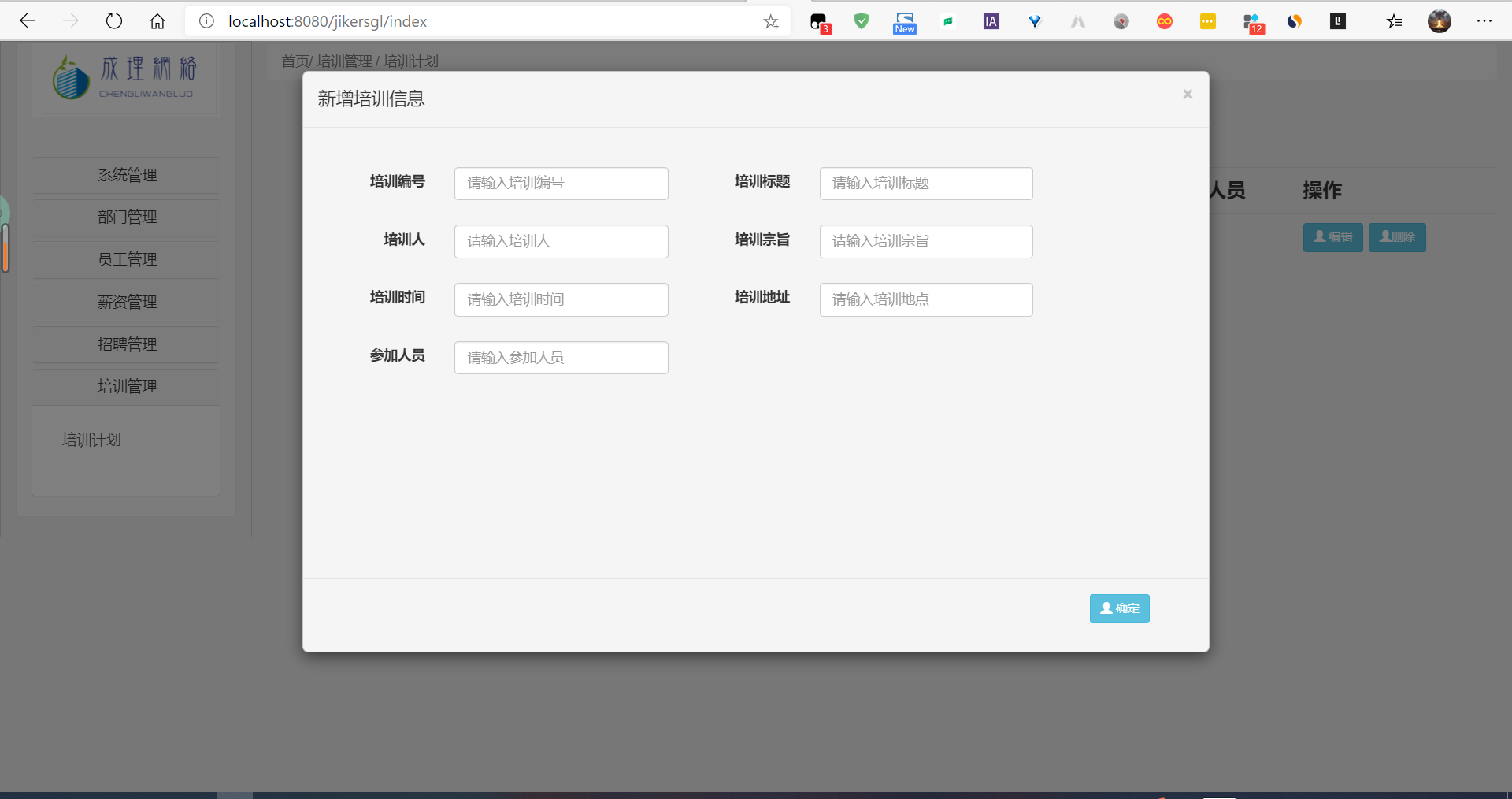
return "删除成功";

}

}

**3.8 培训计划**





TrainController.java

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

@Autowired

private TrainService trainService;

/\*

\* @author

\* @description

\* @date

\*/

@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";

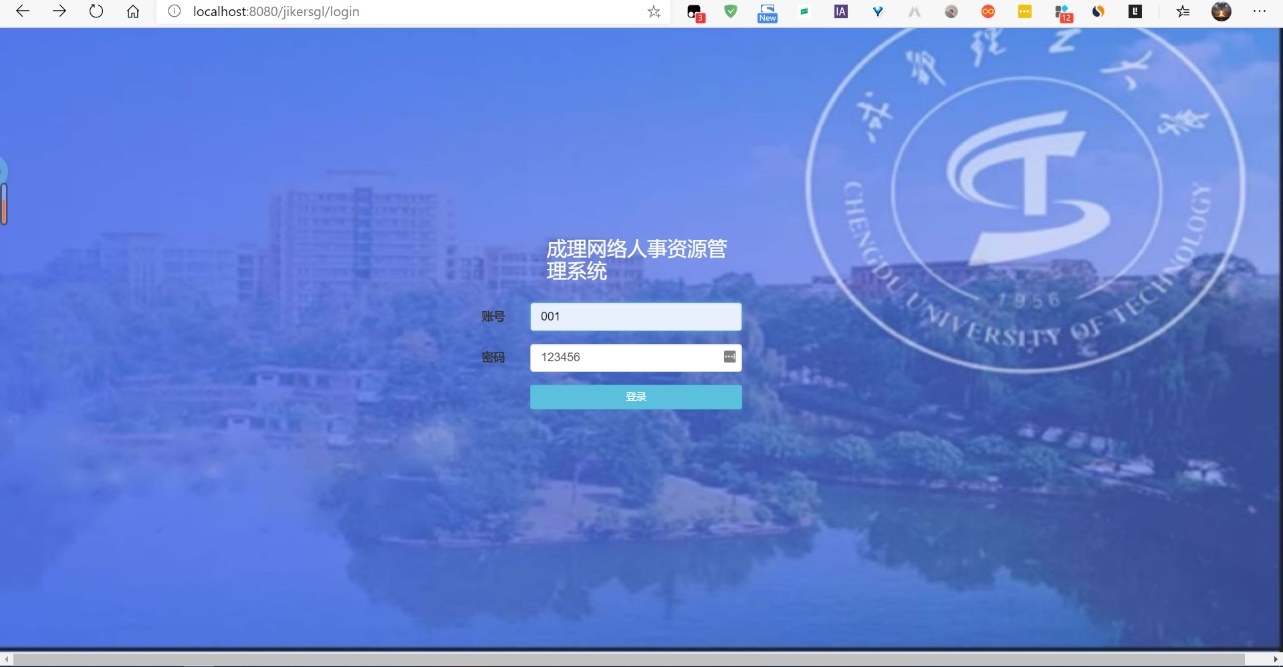
}

}

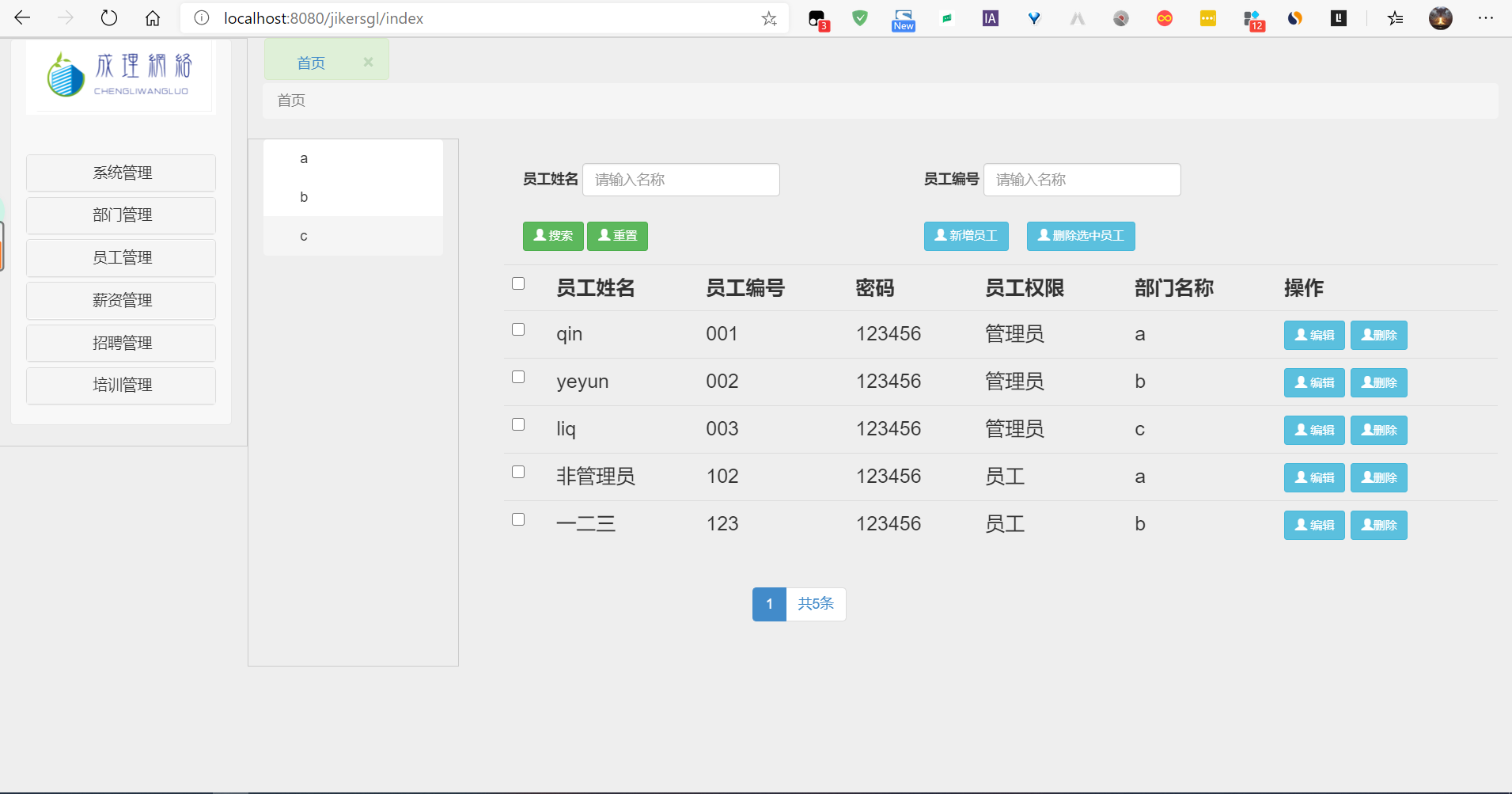
**四.登录体验**

**4.1 管理员登录**

**◆** 输入用户名：“ ”，密码：“ ”，点击登录

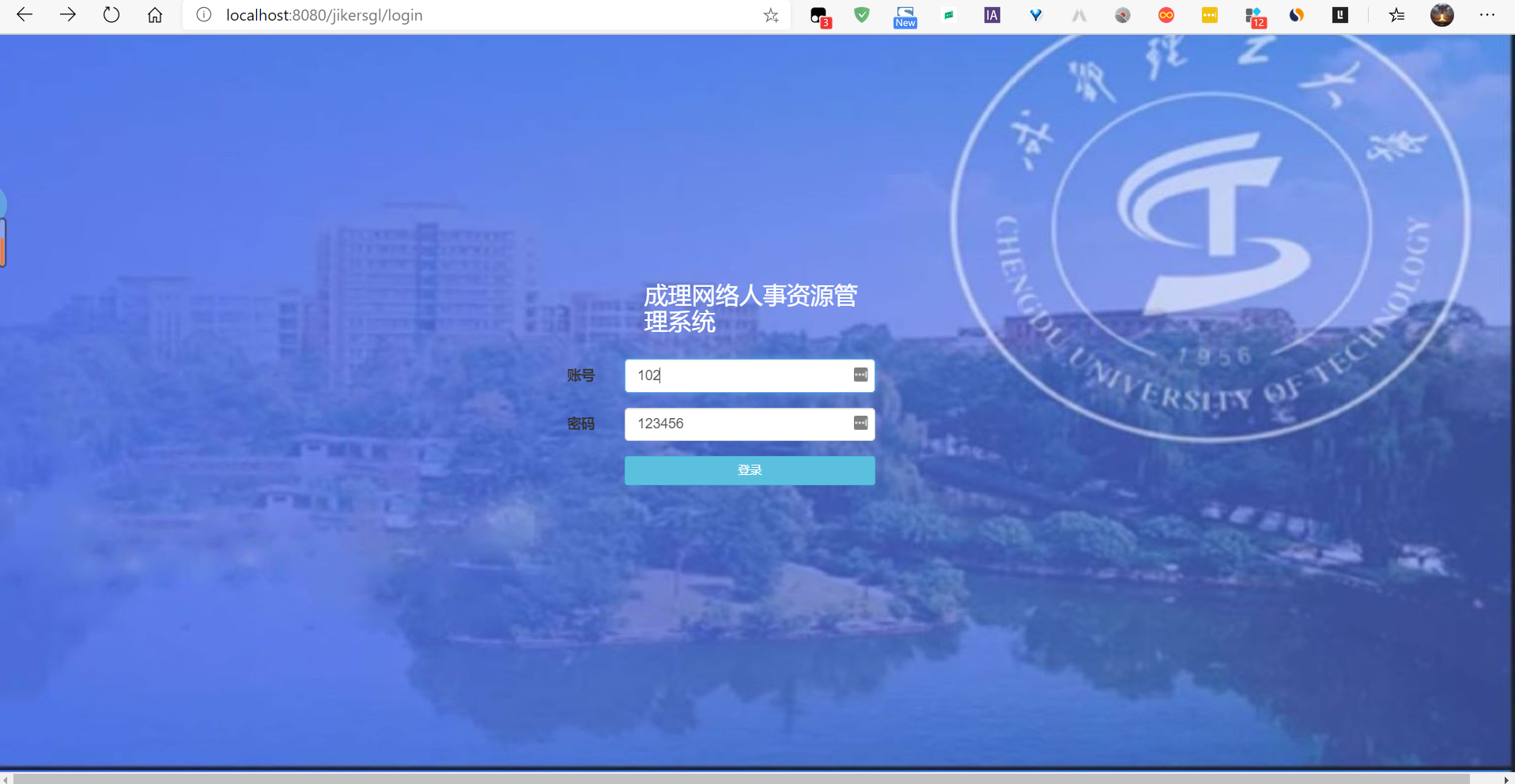


* 登录成功后进入管理员主界面，如下图：



**4.2 员工登录**

**◆** 输入用户名：“”，密码：“”，点击登录



* 登录成功后进入员工主界面，如下图：

