# KTV综合管理系统项目说明书

**19** 级软件 **8**班

组长：曹伟

组员：张天浩 张津赫 杨旭

## 一、序

### 分工

曹伟：数据库的构建维护，员工管理系统、部分前台系统编写，文档编写

张天浩：网页端预订系统编写，文档编写

杨旭：包内管理系统，文档编写

张津赫：前台管理系统，文件编写

### 项目开发开发背景

本次综合实践项目所开发的系统为KTV综合管理系统。目的是为了满足KTV行业日常的基本需求，包括KTV员工管理、包内订单管理、前台系统管理、网页预约管理等实际要求。针对每一个要求，开发出一个子系统。部署在KTV的不同位置以满足日常要求。

### 项目开发环境

CS端：

* 编程平台：IntelliJ IDEA
* 开发平台：Windows10/11
* 数据库:oracle11g 11.2.0.1.0

BS端：

* 编程平台：IntelliJ IDEA
* 前端框架：Bootstrap 4.3.1
* 后端框架：springMVC
* js文件：javascript原生语言及jquery 3.2.1
* 数据库:oracle11g 11.2.0.1.0

### 前期工作

通过前期的资料查询，对当前市场的KTV行业的管理方式及需求有了大致了解。对于KTV行业进行了较为详细的需求分析，详细设计等工作。最终开发出这样的一个多端综合管理系统。

## 二、需求分析

### 问题描述

在实际的KTV系统运行当中，在不同场景的系统设备功能都是不相同的。例如前台管理系统尽量不要加入单个商品的点单信息、员工管理系统只能由经理的设备使用，不应出现在其他位置的设备、网页端主要是客户来使用，用来点单，注册充值等、包内的消费系统用于包厢内酒水等消费品的消费管理。本次综合实践系统也分为四种，对应四种场景下的四种使用情况。

### 可行性分析

技术可行性：小组成员均有较为扎实的编程基础，并且通过本学期Oracle数据库的学习。在能力上技术上完全可以胜任本次开发工作。

### 需求描述

#### BS端预定模块

1.网上客户在预约前进行注册。

2.网上用户在预约前进行登录。

3.网上用户进行预约取消预约、查看房间信息、查看已预定信息。

#### 前台模块：

1.员工在前台为客户进行开放预约、查询取消预约。

2.员工查看当前房间信息

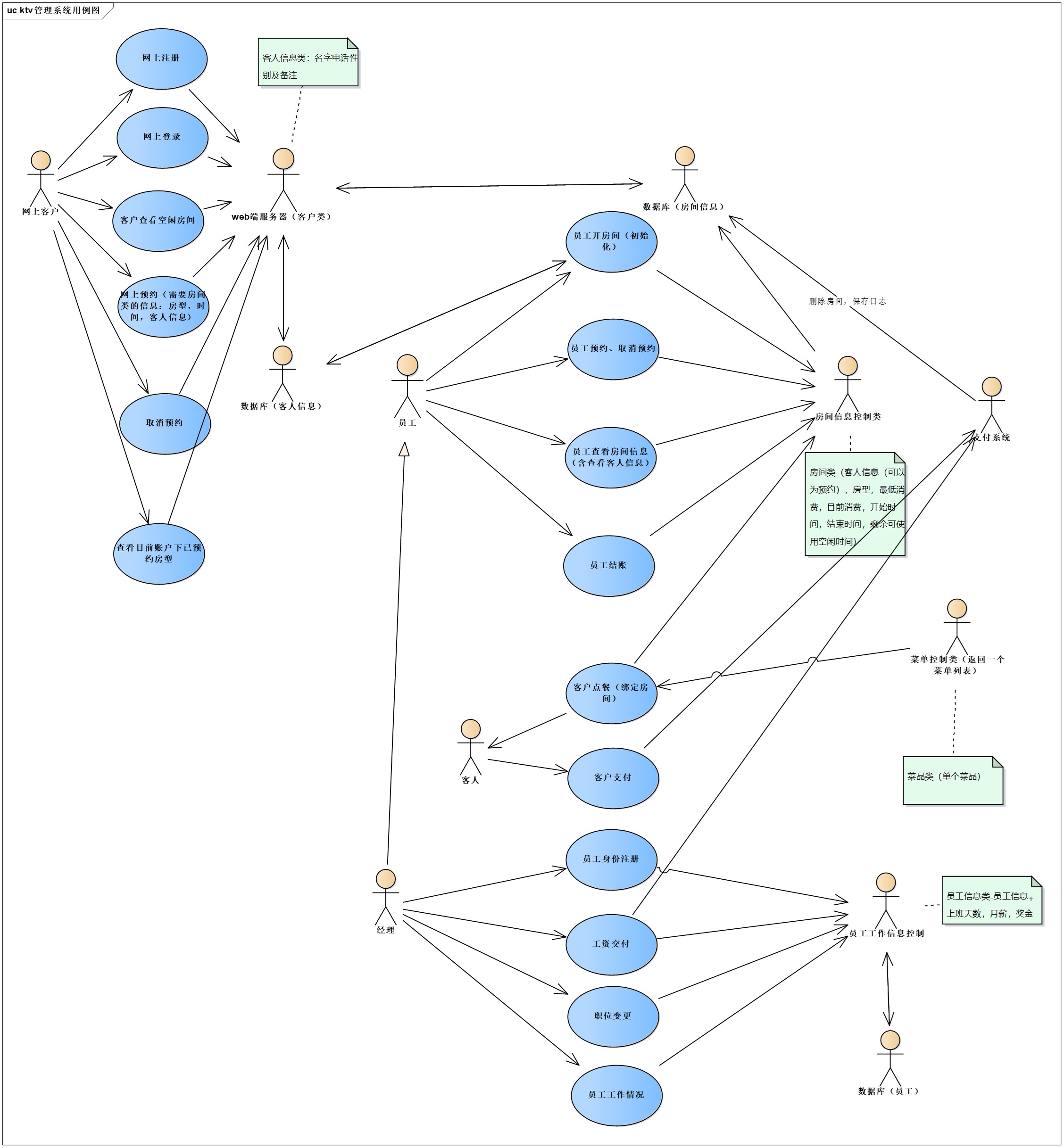
3.员工在前台进行收费结账

#### 包内点单模块

1.客户进行点单

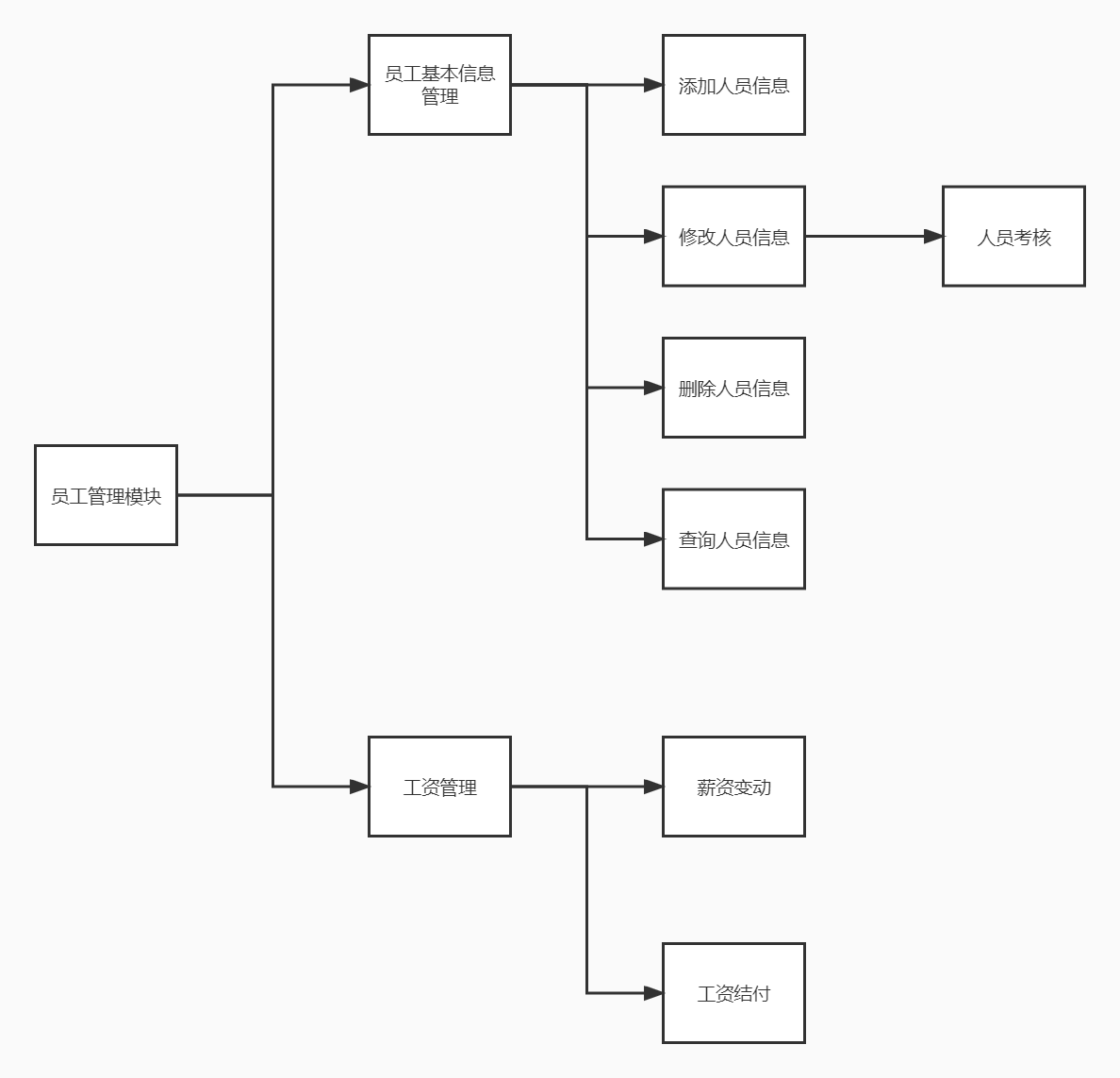
2.支付消费

### 需求分析用例图

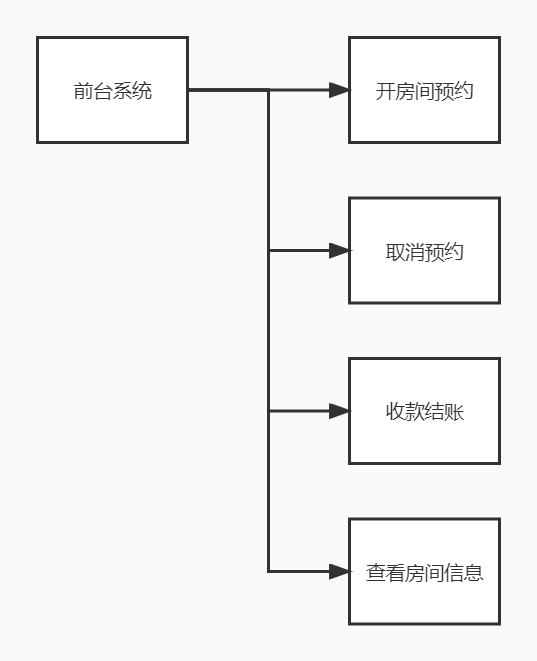


## 三、功能模块

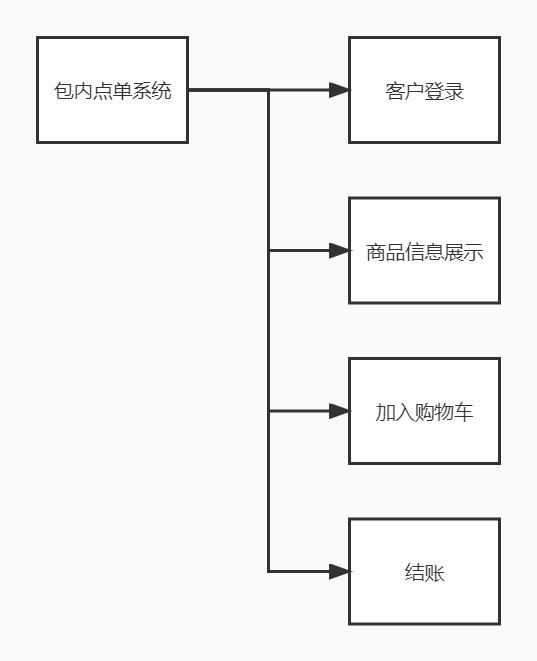
#### 员工模块组成



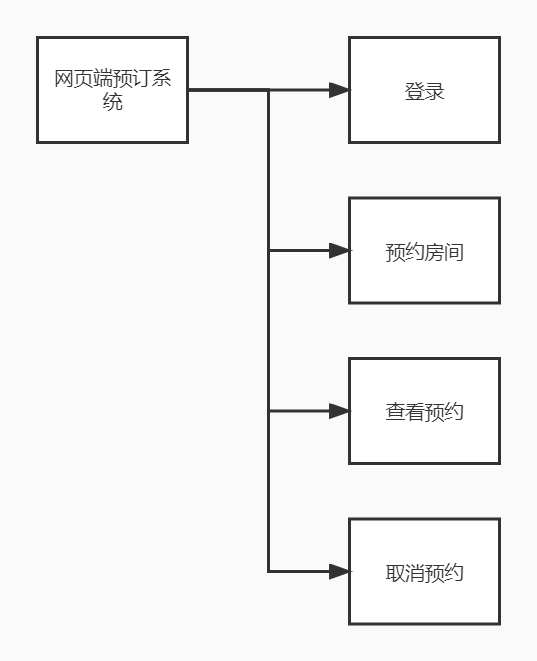
#### 前台模块



#### 包内点单模块

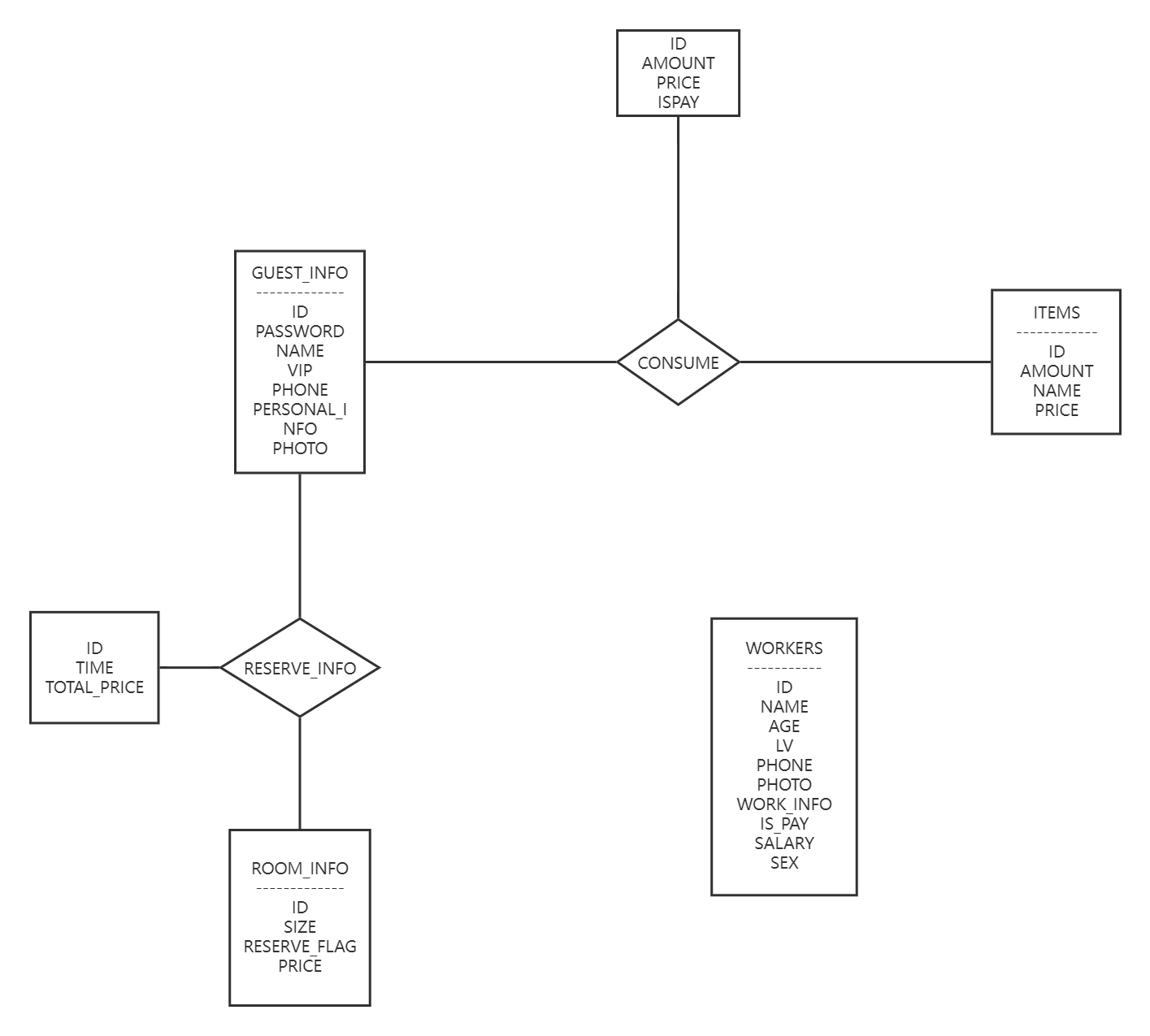


#### 网页端预约模块



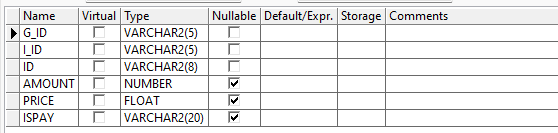
## 四、总体设计

### ER图



### 关系表

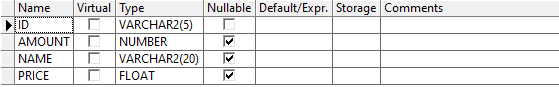
#### 消费品表(Consume)



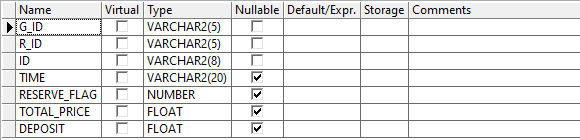
#### 客户信息表（Guest\_Info）



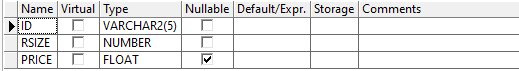
#### 库存信息（Items）



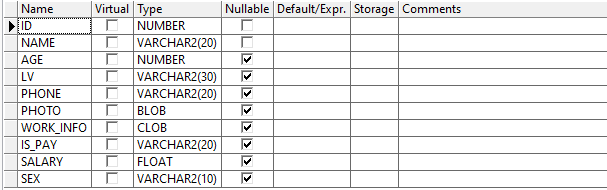
#### 预定信息（RESERVE\_INFO）



#### 房间信息（ROOM\_INFO）



#### 员工信息（WORKERS）



## 五、详细设计

### 数据库模块说明（负责人：曹伟）

#### 1.数据表的构建：

一共六张数据表，现只挑其中部份表的构建进行展示

-----消费品表   
-- Create table   
create table CONSUME   
(   
 g\_id VARCHAR2(5) not null,   
 i\_id VARCHAR2(5) not null,   
 id VARCHAR2(8) not null,   
 amount NUMBER,   
 price FLOAT,   
 ispay VARCHAR2(20)   
)   
tablespace MONITOR   
 pctfree 10   
 initrans 1   
 maxtrans 255   
 storage   
 (   
 initial 64K   
 next 1M   
 minextents 1   
 maxextents unlimited   
 );   
-- Create/Recreate primary, unique and foreign key constraints   
alter table CONSUME   
 add constraint PK\_CID primary key (ID)   
 using index   
 tablespace MONITOR   
 pctfree 10   
 initrans 2   
 maxtrans 255   
 storage   
 (   
 initial 64K   
 next 1M   
 minextents 1   
 maxextents unlimited   
 );   
alter table CONSUME   
 add constraint FK\_GCID foreign key (G\_ID)   
 references GUEST\_INFO (ID);   
alter table CONSUME   
 add constraint FK\_RIID foreign key (I\_ID)   
 references ITEMS (ID);   
   
  
   
 --------预定信息表   
   
 -- Create table   
create table RESERVE\_INFO   
(   
 g\_id VARCHAR2(5) not null,   
 r\_id VARCHAR2(5) not null,   
 id VARCHAR2(8) not null,   
 time VARCHAR2(20),   
 reserve\_flag NUMBER,   
 total\_price FLOAT,   
 deposit FLOAT   
)   
tablespace MONITOR   
 pctfree 10   
 initrans 1   
 maxtrans 255   
 storage   
 (   
 initial 64K   
 next 1M   
 minextents 1   
 maxextents unlimited   
 );   
-- Create/Recreate indexes   
create index INDEX\_GID on RESERVE\_INFO (G\_ID)   
 tablespace MONITOR   
 pctfree 10   
 initrans 2   
 maxtrans 255   
 storage   
 (   
 initial 64K   
 next 1M   
 minextents 1   
 maxextents unlimited   
 );   
-- Create/Recreate primary, unique and foreign key constraints   
alter table RESERVE\_INFO   
 add constraint PK\_REID primary key (ID)   
 using index   
 tablespace MONITOR   
 pctfree 10   
 initrans 2   
 maxtrans 255   
 storage   
 (   
 initial 64K   
 next 1M   
 minextents 1   
 maxextents unlimited   
 );   
alter table RESERVE\_INFO   
 add constraint FK\_GID foreign key (G\_ID)   
 references GUEST\_INFO (ID);   
alter table RESERVE\_INFO   
 add constraint FK\_RID foreign key (R\_ID)   
 references ROOM\_INFO (ID);

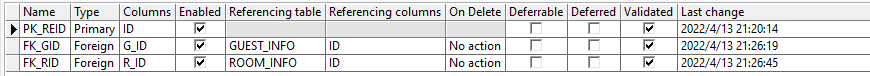
#### 2.用户和表空间的构建。

在创建数据库时，创建表空间MONITOR，和表空间的DBA角色HOMEWORK

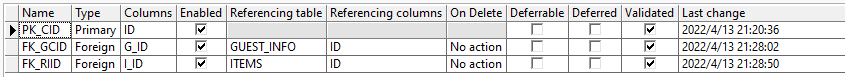
#### 3.主键和外键约束

每张表都有自己的主键。

RESERVE\_INFO的主键和外键。



CONSUME的主键和外键。



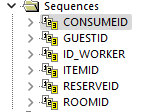
#### 4.视图

1.员工信息表视图   
create or replace view idnameorsal as select id,name,salary,is\_pay from workers;   
2.预定信息的视图   
create or replace view resguestroomid as select id,g\_id,r\_id from RESERVE\_INFO;   
3.预定信息表视图2   
create or replace view showreserve as select id,R\_ID,TIME,RESERVE\_FLAG from RESERVE\_INFO;

#### 5.索引

1.按预定信息的客人id创建的索引   
create index INDEX\_GID on RESERVE\_INFO (G\_ID)   
2.员工信息表，支付情况的索引   
create index INDEX\_ISPAY on WORKERS (IS\_PAY)

#### 6.序列



1.CONSUMEID序列   
-- Create sequence   
create sequence CONSUMEID   
minvalue 1   
maxvalue 9999999999999999999999999999   
start with 1000000   
increment by 1   
cache 20;   
   
2.RESERVEID序列   
-- Create sequence   
create sequence RESERVEID   
minvalue 1   
maxvalue 9999999999999999999999999999   
start with 10000020   
increment by 1   
cache 20;   
   
--其余序列略

#### 7.触发器

在本次综合实践中，每一个序列都对应一个触发器。用于完成对每张表主键的自增操作。

1.RESERVE\_INFO表主键自增触发器   
create or replace trigger autoincreaseRESID   
before   
insert on RESERVE\_INFO   
for each row   
declare   
begin   
select RESERVEID.nextval into :new.id from dual;   
end;   
   
   
2.WORKERS表主键自增触发器   
create or replace trigger autoincrease   
before   
insert on workers   
for each row   
declare   
begin   
select id\_worker.nextval into :new.id from dual;   
end;

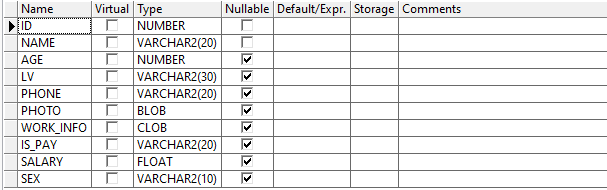
#### 8.函数

在本次综合实践中只实现了一个函数，用于根据一个预定信息的ID，计算出还需要补交的费用。在前台支付功能中用于查看是否有未结清订单。

create or replace function RoomIsPaid(RESID RESERVE\_INFO.ID%TYPE) RETURN FLOAT   
IS   
S FLOAT;   
BEGIN   
 SELECT TOTAL\_PRICE-DEPOSIT INTO S FROM RESERVE\_INFO R WHERE R.ID=RESID;   
 RETURN S;   
END;

#### 9.大对象存取

1.员工信息表的PHOTO字段，为BLOB字段，用于保存图片。WORK\_INFO，为CLOB类型，用于存放考核信息。



java中大对象操作函数：

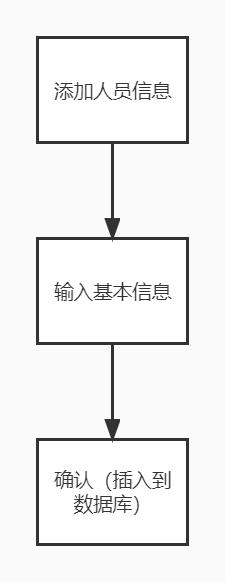
1.存储图片   
 public boolean insertBlob(int id,String filepath)//将路径下的图片存入数据库   
 {   
 PreparedStatement ps = null;   
   
 String sql = "update workers set photo=? where id=?";   
 try {   
 ps =conn.prepareStatement(sql);   
   
 File file=new File(filepath)   
 java.io.BufferedInputStream imageInput = new java.io.BufferedInputStream(   
 new java.io.FileInputStream(file));   
   
   
 ps.setBinaryStream(1, imageInput,(int) file.length());   
 ps.setInt(2,id);   
 ps.executeUpdate();   
 } catch (SQLException | FileNotFoundException e) {   
 e.printStackTrace();   
 }   
   
 return true;   
 };   
2.存储CLOB类型   
 public boolean insertClob(int id,String text)   
 {   
   
 PreparedStatement ps = null;   
   
 String sql = "update workers set work\_info=? where id=?";   
 try {   
 ps =conn.prepareStatement(sql);   
   
 Reader clobReader = new StringReader(text); // 将 text转成流形式   
 ps.setCharacterStream(1, clobReader, text.length());// 替换sql语句中的？   
 System.out.println("manager:199::"+text+" "+id);   
   
 //ps.setClob(1,clob);   
 ps.setInt(2,id);   
 ps.executeUpdate();   
 } catch (SQLException e) {   
 e.printStackTrace();   
 }   
   
   
 return true;   
 };   
   
3.从数据库读取BLOB类型并转成图片   
 Blob pic = p.pPhoto;   
 if(p.pPhoto!=null) {   
   
 Date day=new Date();   
 SimpleDateFormat df = new SimpleDateFormat("MM-dd\_HH@mm@ss");   
 System.out.println(df.format(day));   
   
 String filepath = "./" + df.format(day) + ".jpg";   
 System.out.println("输出文件路径为:" + filepath);   
 try {   
 InputStream in = pic.getBinaryStream(); // 建立输出流   
 FileOutputStream file = new FileOutputStream(filepath);   
 int len = (int) pic.length();   
 byte[] buffer = new byte[len]; // 建立缓冲区   
 while ((len = in.read(buffer)) != -1) {   
 file.write(buffer, 0, len);   
 }   
 file.close();   
 in.close();   
 } catch (Exception e) {   
 e.printStackTrace();   
 System.out.println("I/O Exception.");   
 }   
   
 icon = new ImageIcon(filepath);   
 icon.setImage(icon.getImage().getScaledInstance(210,280, Image.SCALE\_DEFAULT));   
 realPic.setIcon(icon);   
 }   
4.从数据库读取CLOB类型并转成图片   
 public String clobToString(Clob clob){   
 String re = "";   
 Reader is = null;   
 BufferedReader br = null;   
 try {   
 // 得到流   
 is = clob.getCharacterStream();   
 br = new BufferedReader(is);   
 String s = br.readLine();   
 StringBuffer sb = new StringBuffer();   
 // 执行循环将字符串全部取出付值给StringBuffer由StringBuffer转成STRING   
 while (s != null) {   
 sb.append(s);   
 s = br.readLine();   
 }   
 re = sb.toString();   
 } catch (Exception e) {   
 e.printStackTrace();   
 }finally {   
 if (is != null) {   
 try {   
 is.close();   
 } catch (IOException e) {   
 e.printStackTrace();   
 }   
 }   
 if (br != null) {   
 try {   
 br.close();   
 } catch (IOException e) {   
 e.printStackTrace();   
 }   
 }   
 }   
 return re;   
 }

### 员工管理模块（负责人：曹伟）

#### 功能说明（加流程图）

##### 1.添加人员信息

输入员工基本信息（姓名，性别，联系方式等)，点击添加。



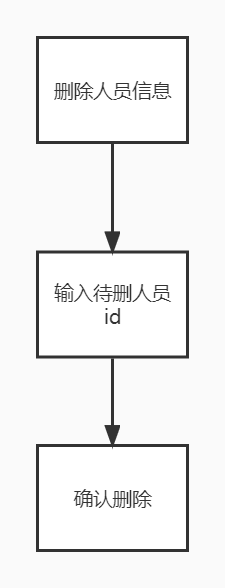
##### 2.修改员工信息

点击修改，修改基本信息后直接提交。若修改照片，考核信息则跳转到新界面进行操作



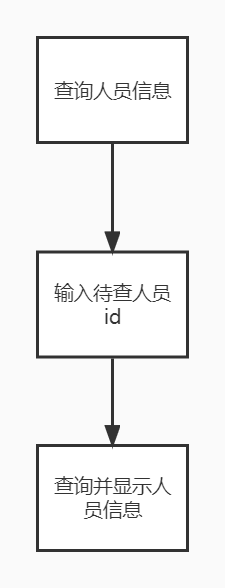
##### 3.删除人员

输入待删除的人员id，查询并显示基本信息，点击删除后进行删除。



##### 4.查询人员：

输入待查人员id，查询并输出员工信息



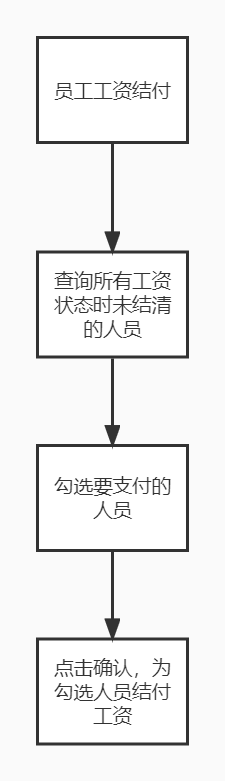
##### 5.修改劳资

输入待修改劳资人员id, 查询现有薪资并展示。输入新劳资。确认修改。



##### 6.员工工资结付

查询所有未结清工资的人员信息。操作者选择需要结清的人员。点击确认返回成功操作提示。



#### 部分代码展示

数据库交互类   
public class Manager{   
 oracle\_connect Dbconn=new oracle\_connect();   
 Connection conn=Dbconn.get\_Connection();   
 public boolean insert(Person p){   
 String sql;   
 sql = "insert into workers(name,age,lv,phone,salary,is\_pay,sex,photo,work\_info) values (?,?,?,?,?,?,?,?,?)";   
 PreparedStatement ps = null;   
 try {   
 ps = conn.prepareStatement(sql);   
 ps.setString(1,p.pName);//name   
 ps.setInt(2,p.pAge);   
 ps.setString(3,p.plv);   
 ps.setString(4,p.pPhone);   
 ps.setFloat(5,p.pSalary);   
 ps.setString(6,p.ispay);   
 ps.setString(7,p.pSex);   
 ps.setBlob(8,p.pPhoto);   
 ps.setClob(9,p.worker\_info);   
 ps.execute();   
 } catch (SQLException e) {   
 e.printStackTrace();   
 }   
 return true;   
 };   
 public boolean update(Person p)   
 {   
 String sql = "update workers set name=?,age=?,lv=?,phone=?,salary=?,is\_pay=?,sex=? where id=?";   
 PreparedStatement ps = null;   
 try {   
 ps = conn.prepareStatement(sql);   
 ps.setString(1,p.pName);//name   
 ps.setInt(2,p.pAge);   
 ps.setString(3,p.plv);   
 ps.setString(4,p.pPhone);   
 ps.setFloat(5,p.pSalary);   
 ps.setString(6,p.ispay);   
 ps.setString(7,p.pSex);   
 ps.setInt(8,p.pId);   
 ps.execute();   
 } catch (SQLException e) {   
 e.printStackTrace();   
 }   
 return true;   
 };   
 public Person searchById(int id)   
 {   
 Person p=new Person();   
 String sql="select \* from workers w where w.id="+String.valueOf(id);   
 try {   
 Statement stmt=conn.createStatement();   
 stmt.execute(sql);   
 ResultSet rs= stmt.getResultSet();   
 if(rs.next())   
 {   
 p.pId=id;   
 p.pName=rs.getString("NAME");   
 p.pAge=rs.getInt("AGE");   
 p.pSex=rs.getString("SEX");   
 p.plv=rs.getString("LV");   
 p.pPhone=rs.getString("PHONE");   
 p.pSalary=rs.getFloat("SALARY");   
 p.ispay=rs.getString("IS\_PAY");   
 p.pPhoto= rs.getBlob("PHOTO");   
 p.worker\_info=rs.getClob("WORK\_INFO");   
 }   
 rs.close();stmt.close();   
 } catch (SQLException e) {   
 e.printStackTrace();   
 }   
 return p;   
 }   
   
 public Person searchSal(int id)   
 {   
 Person p=new Person();   
 String sql="select \* from idNameOrSal w where w.id="+String.valueOf(id);   
 try   
 {   
 Statement stmt=conn.createStatement();   
 stmt.execute(sql);   
 ResultSet rs= stmt.getResultSet();   
 if(rs.next())   
 {   
 p.pId=id;   
 p.pName=rs.getString("NAME");   
 p.pSalary=rs.getFloat("SALARY");   
 p.ispay=rs.getString("IS\_PAY");   
 }   
 rs.close();stmt.close();   
 } catch (SQLException e) {   
 e.printStackTrace();   
 }   
 return p;   
 }  
 public boolean insertBlob(int id,String filepath)   
 {   
 PreparedStatement ps = null;   
 String sql = "update workers set photo=? where id=?";   
 try {   
 ps =conn.prepareStatement(sql);   
 File file=new File(filepath);   
 java.io.BufferedInputStream imageInput = new java.io.BufferedInputStream(   
 new java.io.FileInputStream(file));   
 ps.setBinaryStream(1, imageInput,(int) file.length());   
 ps.setInt(2,id);   
 ps.executeUpdate();   
 } catch (SQLException | FileNotFoundException e) {   
 e.printStackTrace();   
 }   
 return true;   
 };   
 public boolean insertClob(int id,String text)   
 {   
 PreparedStatement ps = null;   
 String sql = "update workers set work\_info=? where id=?";   
 try {   
 ps =conn.prepareStatement(sql);   
 Reader clobReader = new StringReader(text); // 将 text转成流形式   
 ps.setCharacterStream(1, clobReader, text.length());// 替换sql语句中的？   
 System.out.println("manager:199::"+text+" "+id);   
 //ps.setClob(1,clob);   
 ps.setInt(2,id);   
 ps.executeUpdate();   
 } catch (SQLException e) {   
 e.printStackTrace();   
 }   
 return true;   
 };   
}

部分界面类：

class search\_node   
{   
 int B\_type=1;//0   
 JLabel search = new JLabel("搜索");   
 JLabel pId = new JLabel("员工编号"); //员工编号   
 JLabel pName = new JLabel("员工姓名"); //员工姓名   
 JLabel pSex = new JLabel("员工性别"); //员工性别   
 JLabel pAge = new JLabel("员工年龄"); //员工性别   
 JLabel plv = new JLabel("员工等级"); //出生年月   
 JLabel pPhone = new JLabel("联系方式"); //所属部门编号   
 JLabel pPhoto = new JLabel("员工照片"); //出生年月   
 JLabel pWorkInfo = new JLabel("员工考核信息"); //所属部门编号   
 JLabel pSalary = new JLabel("工资"); //工资   
 JLabel pIspay = new JLabel("是否结清工资"); //是否考核   
 JTextField search\_ = new JTextField(10);   
 JTextField pId\_ = new JTextField(10); //员工编号   
 JTextField pName\_ = new JTextField(10); //员工姓名   
 JComboBox pSex\_ = new JComboBox(); //员工性别   
 JTextField pAge\_=new JTextField(10);   
 JComboBox plv\_ = new JComboBox(); //员工性别   
 JTextField pPhone\_ = new JTextField(10); //电话号   
 JButton pPhoto\_=new JButton("查看照片");   
 JTextField pSalary\_ = new JTextField(10); //工资   
 JButton pWorkInfo\_=new JButton("查看员工详细状况");   
 JComboBox pIspay\_ = new JComboBox(); //员工性别   
   
 public search\_node()   
 {   
 pPhoto\_.addActionListener(new ActionListener() {   
 public void actionPerformed(ActionEvent e)   
 {   
 Person pnow = new Person();   
 pnow.pId = Integer.parseInt(pId\_.getText());   
 //pnow.pPhoto=pblob;   
 Photo.show\_it(B\_type,pnow);   
   
 }   
 });   
 pWorkInfo\_.addActionListener(new ActionListener() {   
 public void actionPerformed(ActionEvent e)   
 {   
 Person pnow = new Person();   
 pnow.pId = Integer.parseInt(pId\_.getText());   
 //pnow.pPhoto=pblob;   
 WorkInfo.show\_it(B\_type,pnow.pId);   
   
 }   
 });   
 }   
}

#### 测试及实现界面截图

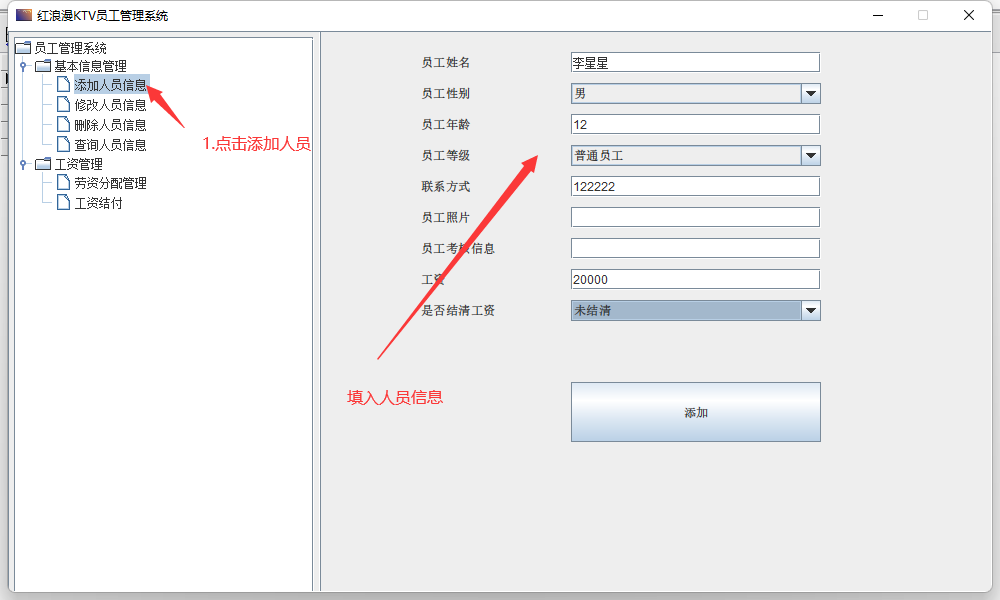
##### WORKERS表初始状态：

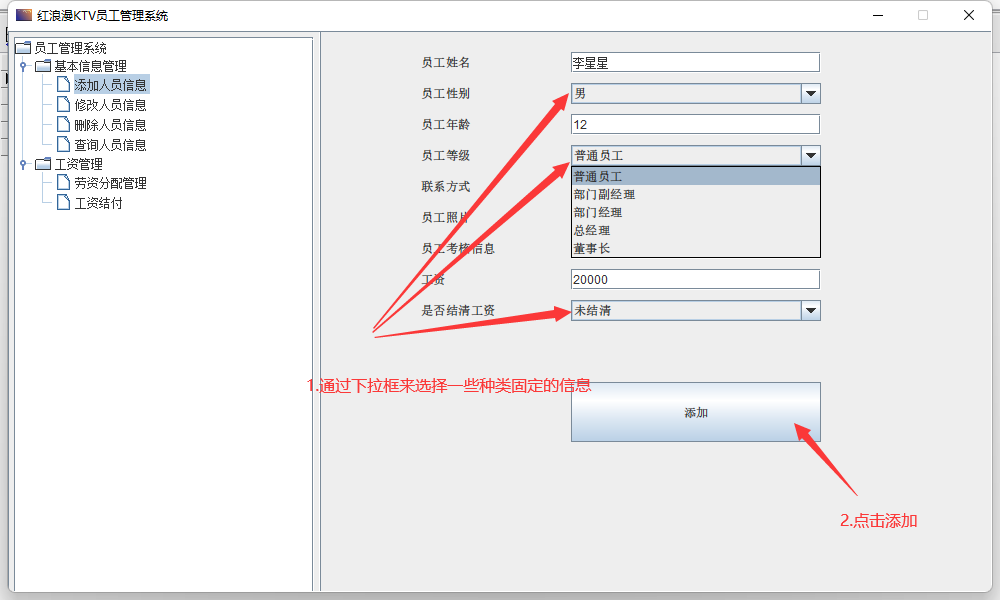


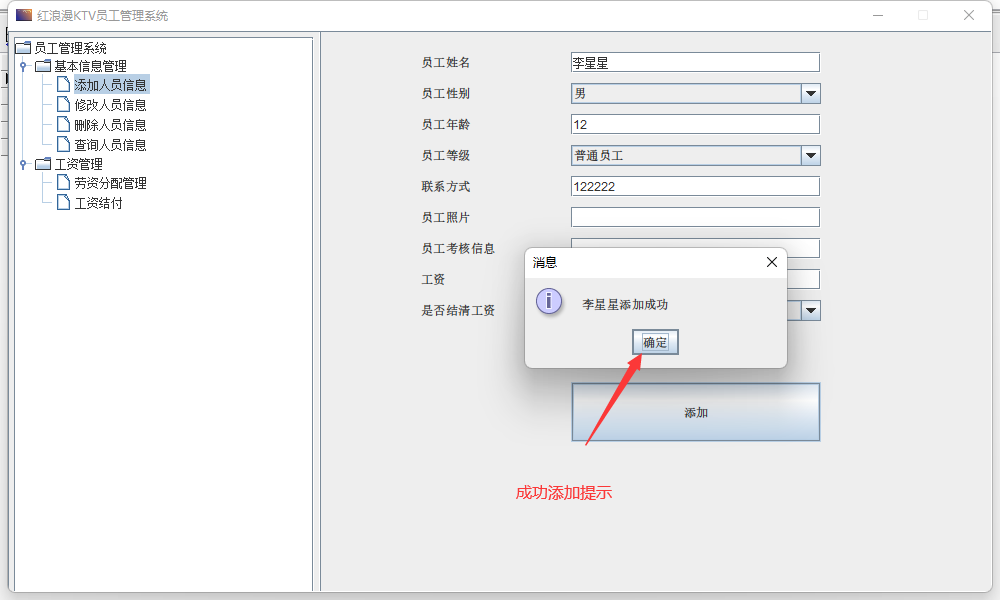
##### 1.主界面



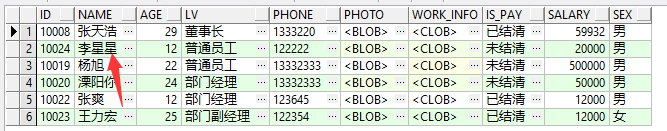
##### 2.人员添加功能



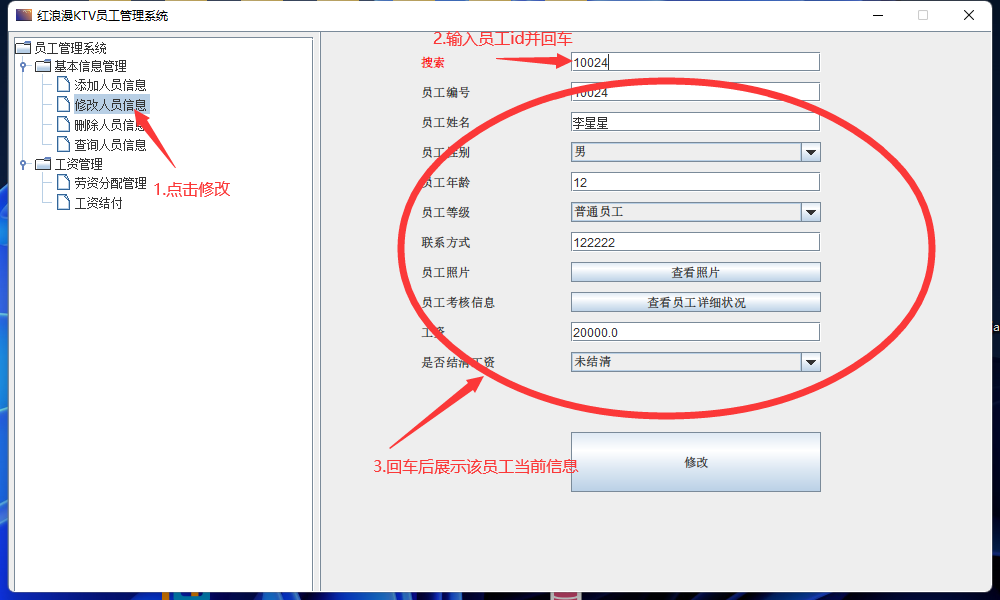


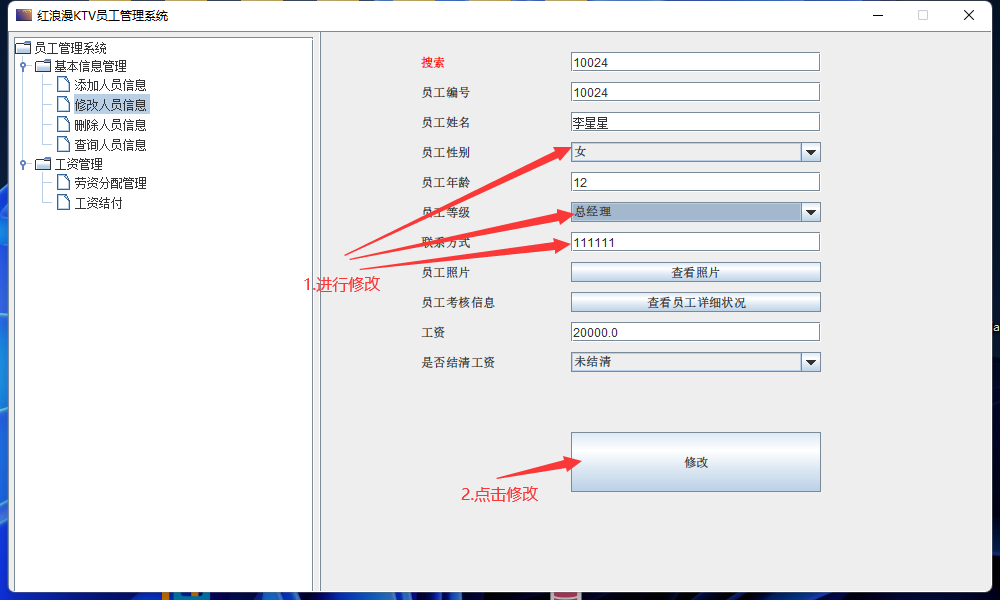


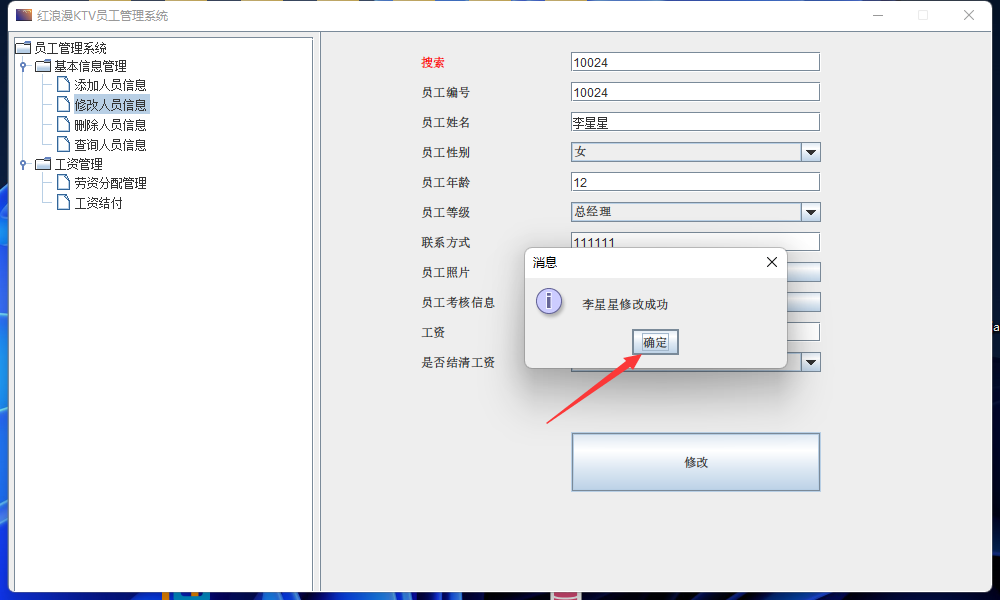
数据库展示：



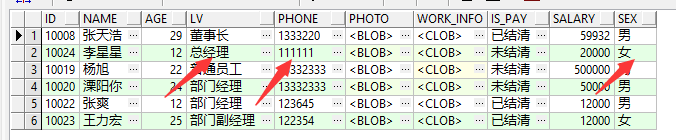
##### 3.修改人员信息



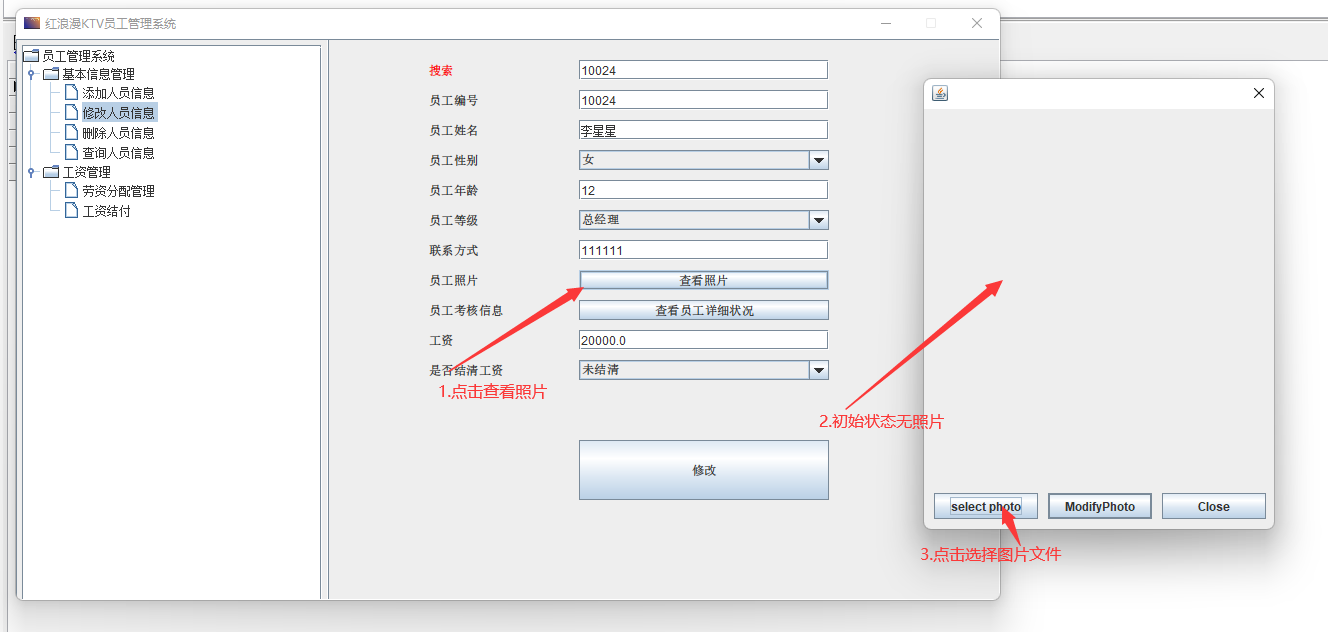




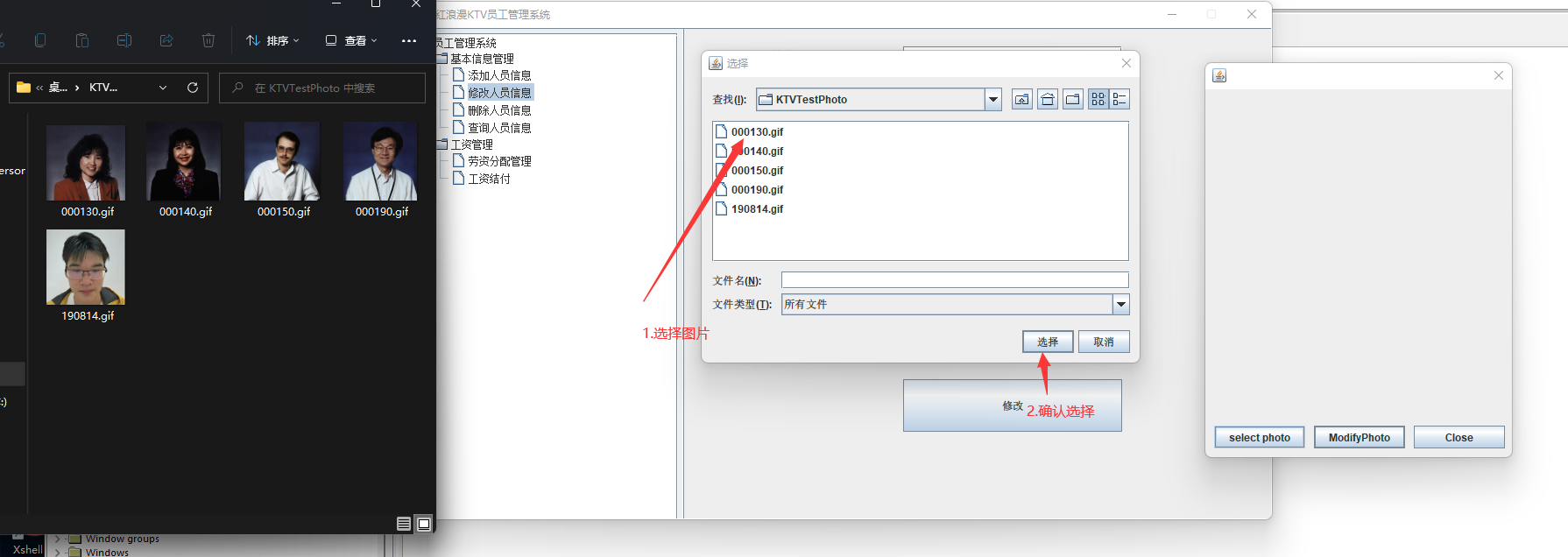
数据库展示：

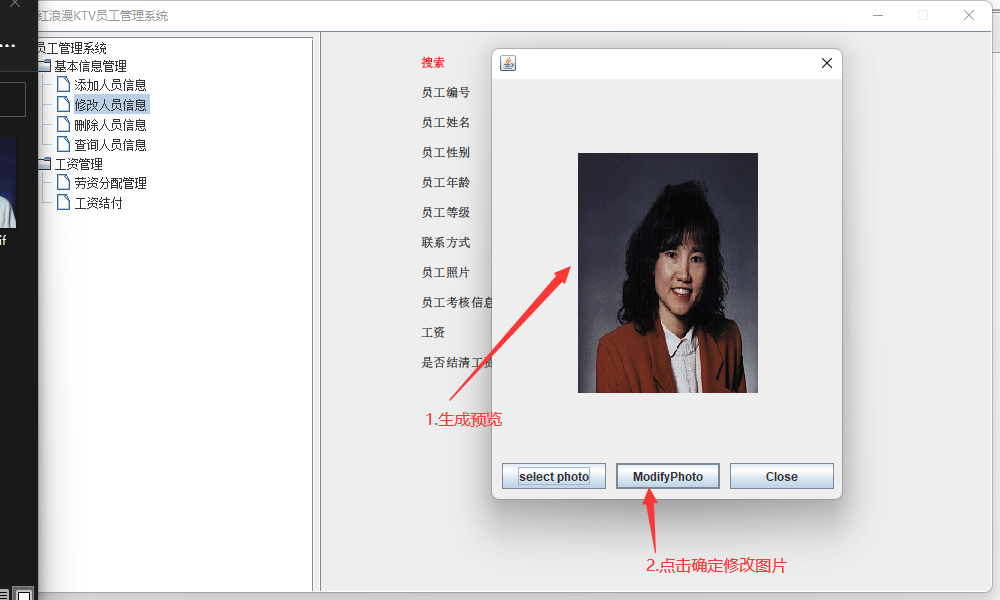


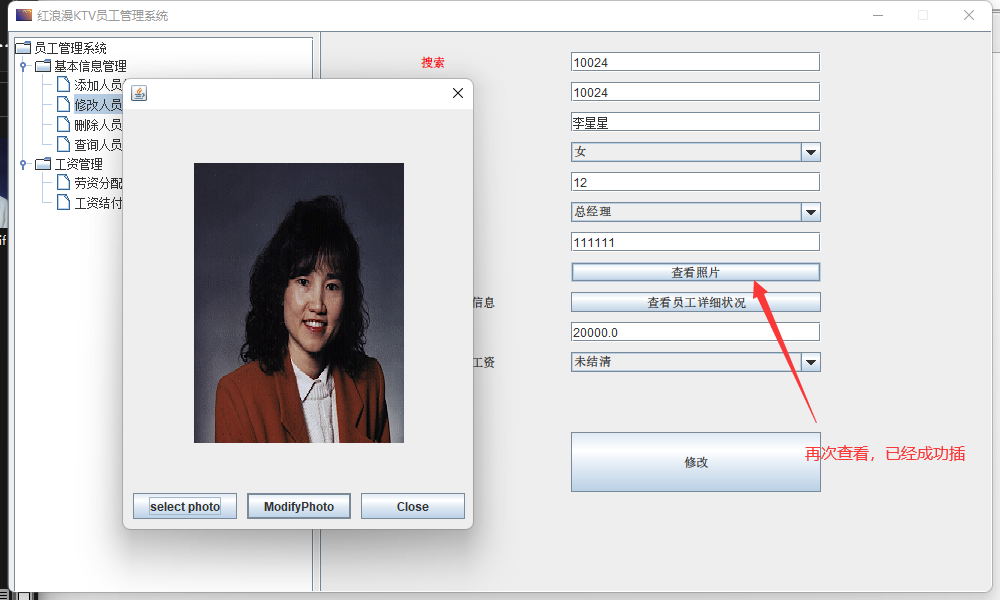
**员工图片插入：**

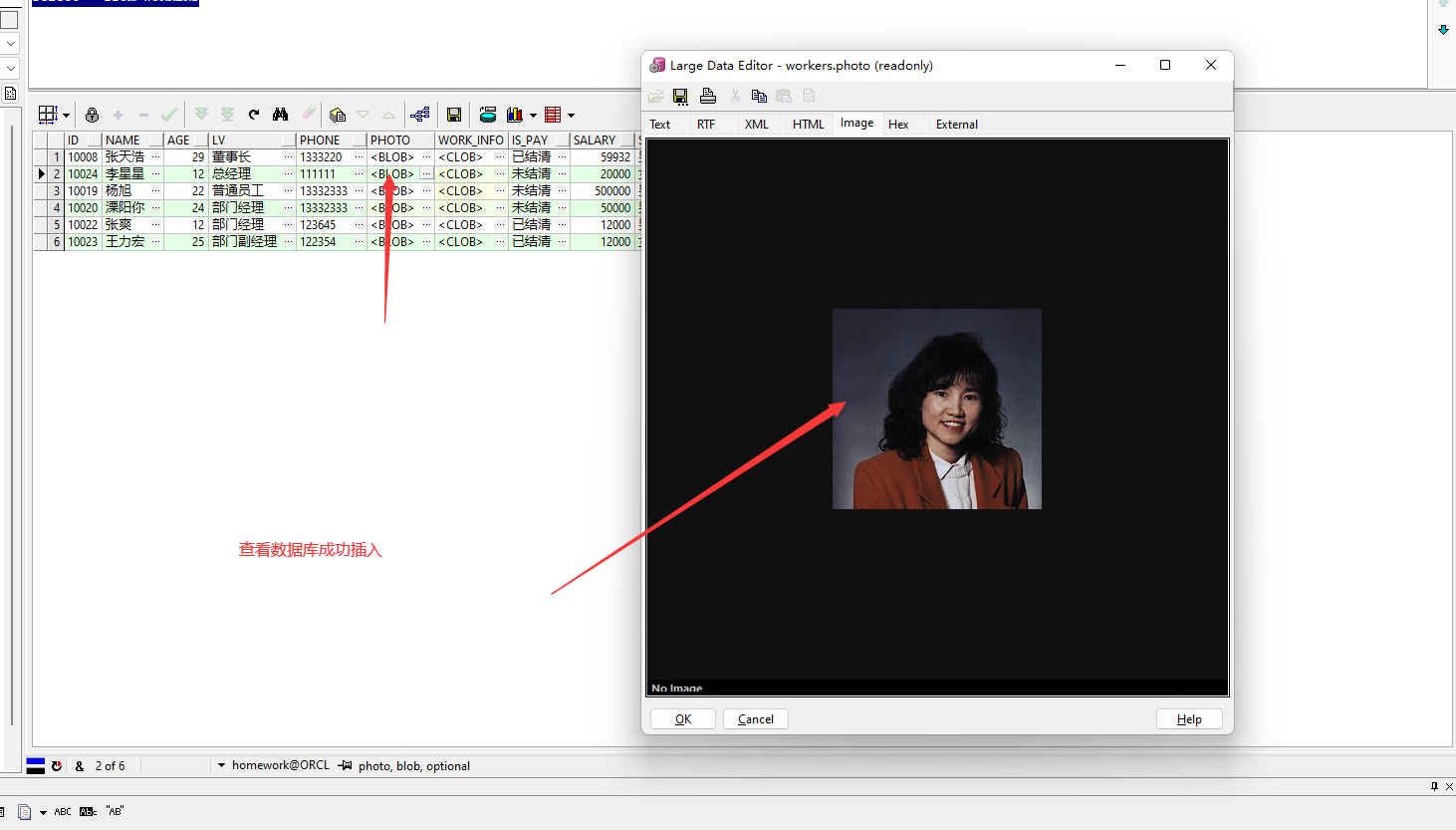


弹出文件资源管理器，选择图片。

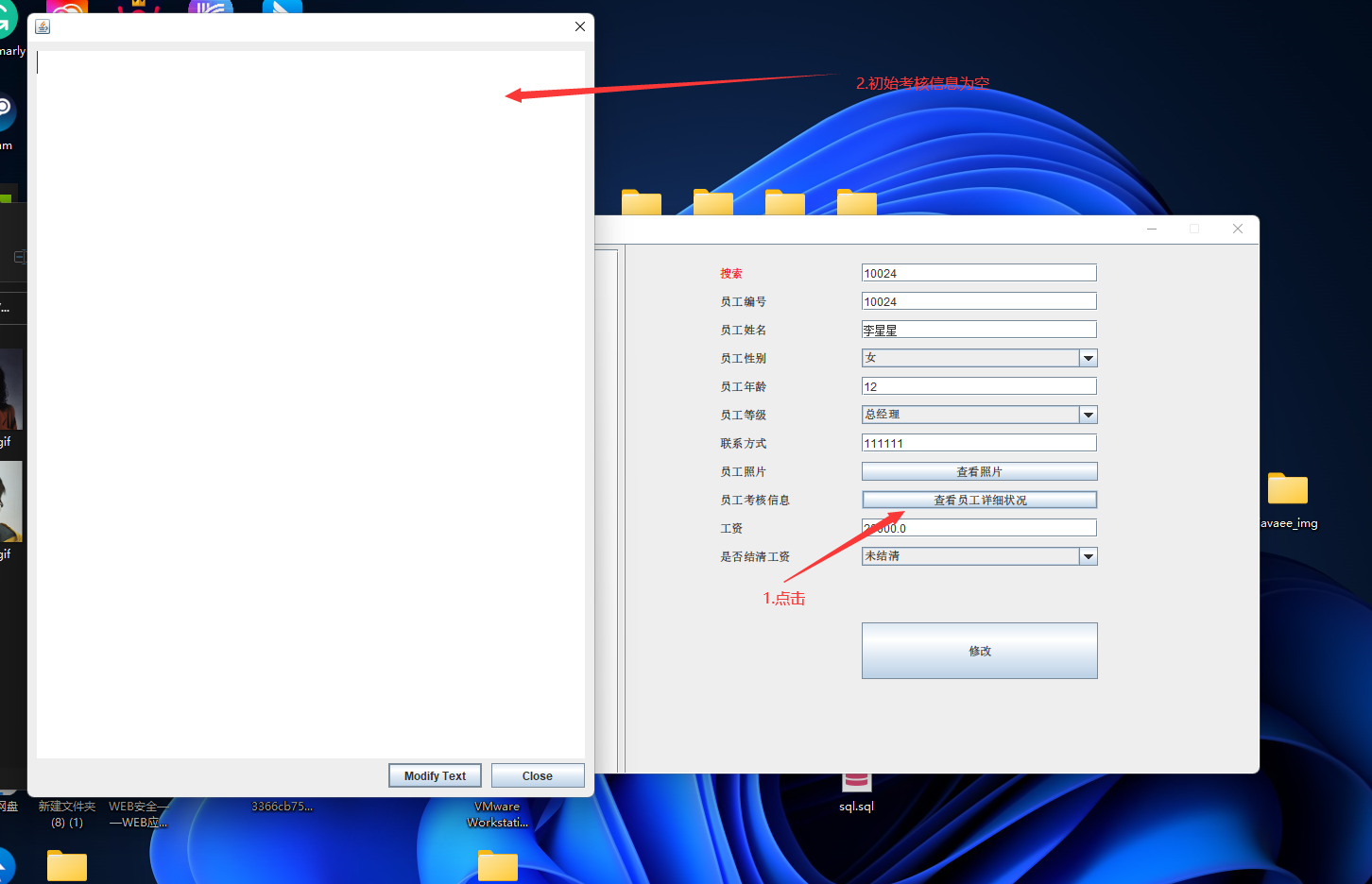






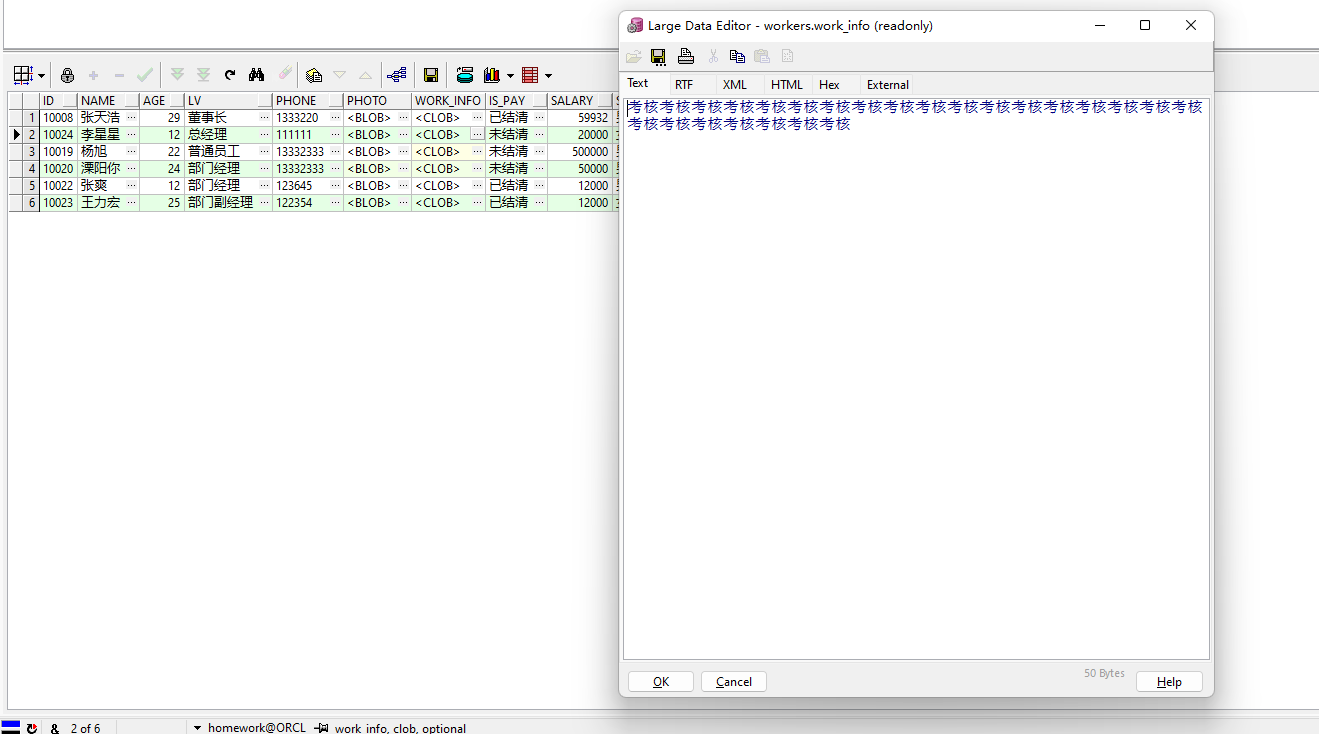


**员工考核信息插入：**



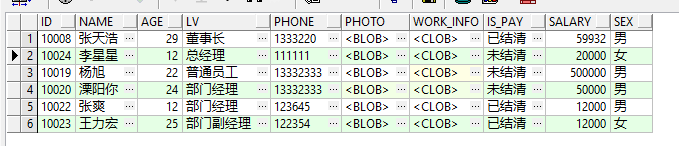


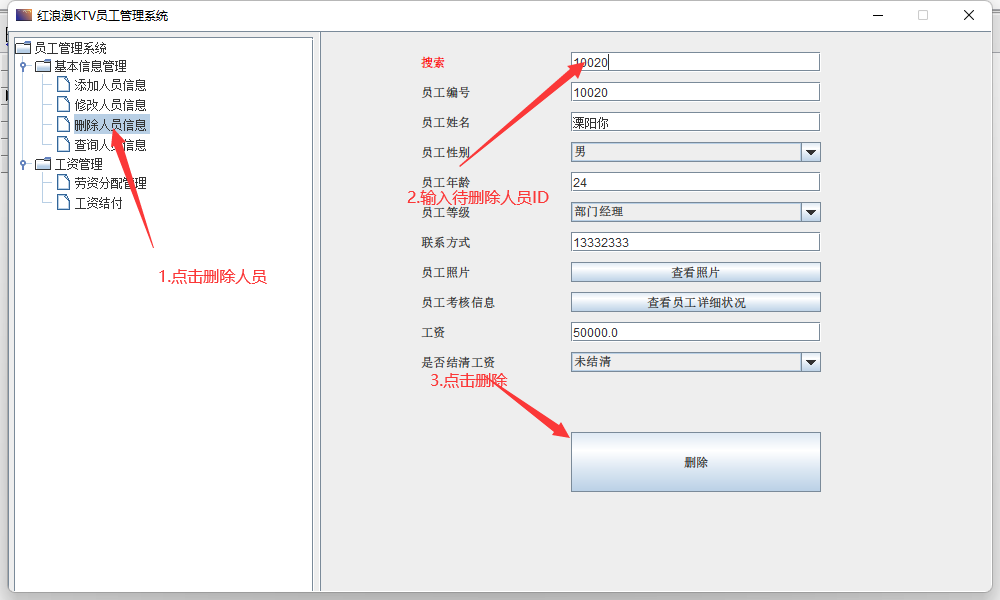
查看数据库：成功插入

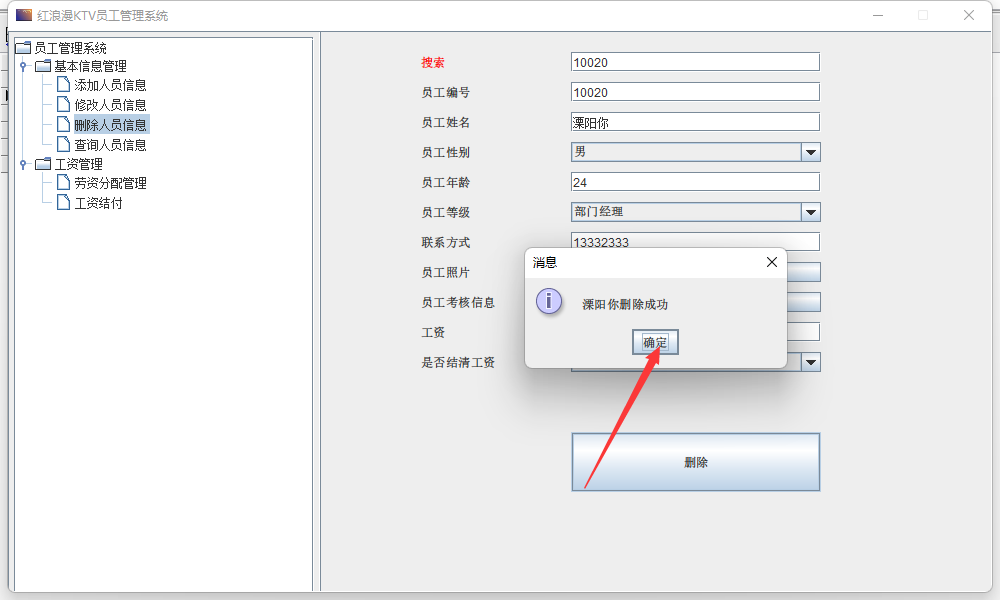


##### 4.删除人员

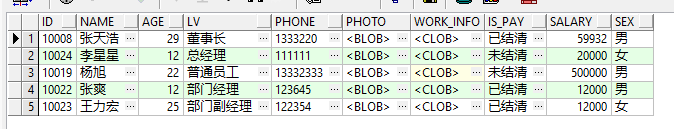
数据库初始状态：







再次查看数据库：成功删除



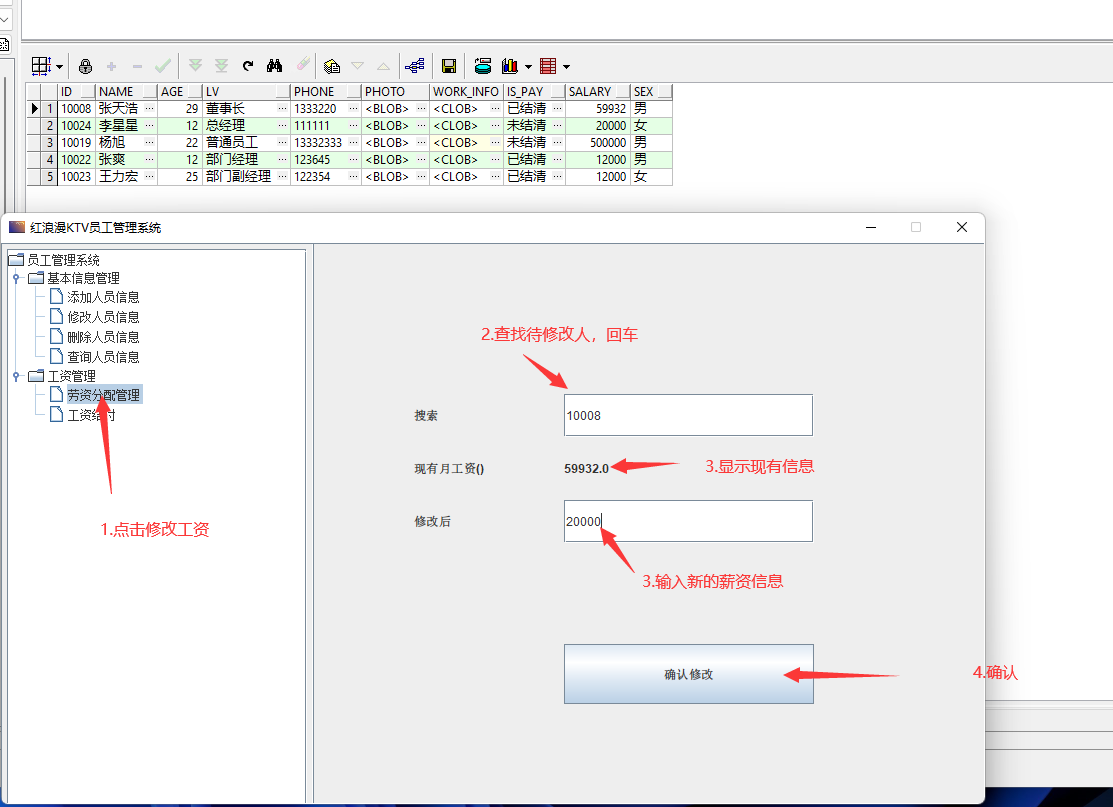
##### 5.查询



查询状态禁止更改信息。

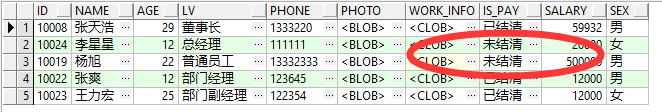


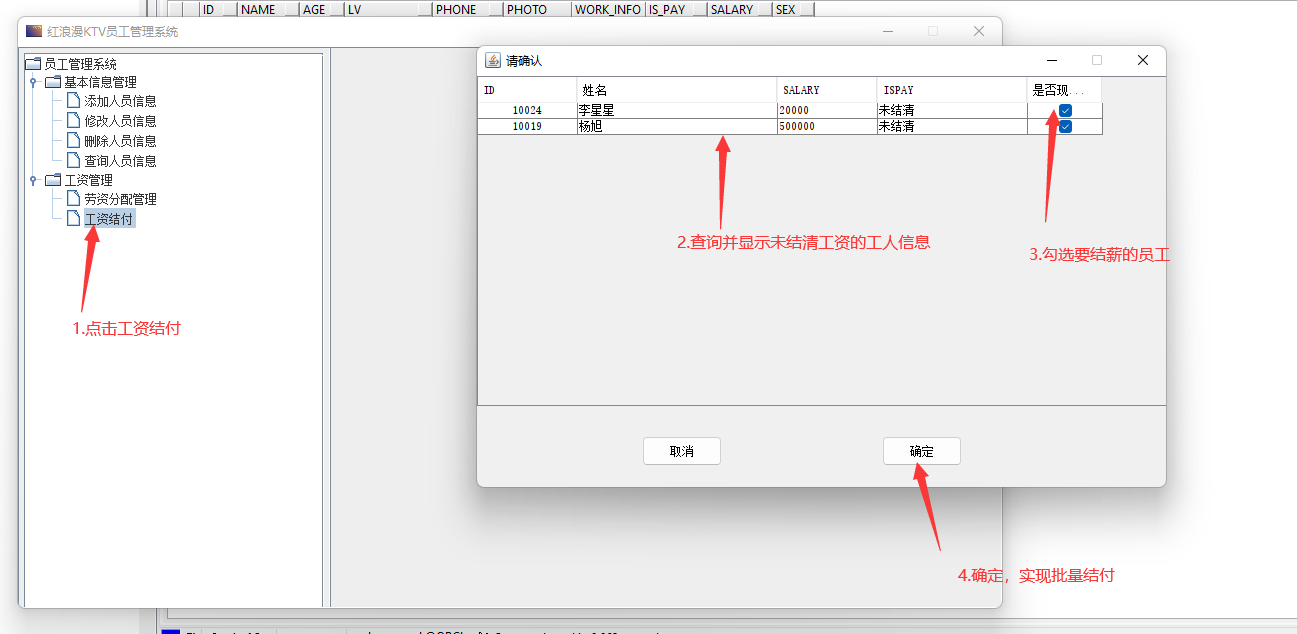
##### 6.修改劳资



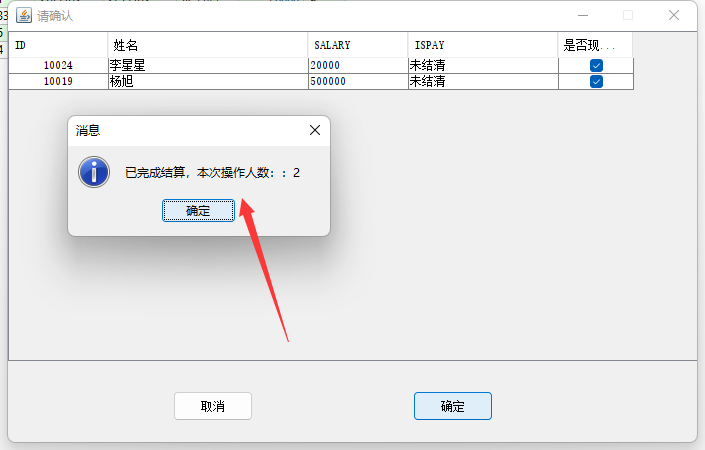
##### 7.结付工资

初始状态数据库：

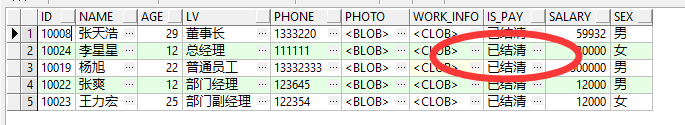




成功后返回，结清工资的人数：



数据库查看：



#### 心得体会

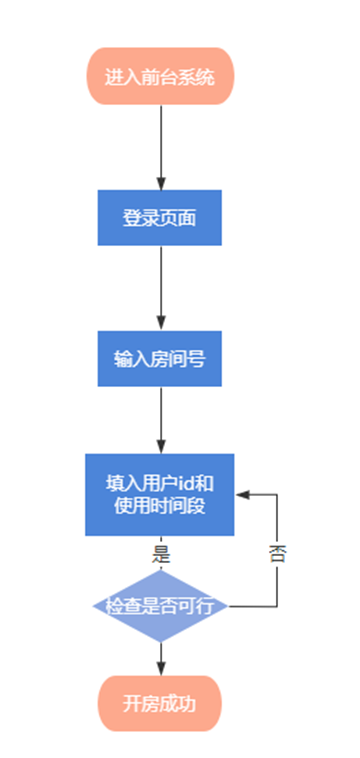
在本次综合实践中。真正意义上的做了一回数据库管理员。不光参与数据库的设计实现。还作为管理员对数据库进行维护。为数据库添加各种视图索引等。是第一次在数据库当中亲自实现这些功能。在完成人员管理模块实现的过程中。将数据库和java开发相结合，由于本次对对数据库使用了更多深入的操作。在完成功能时更加顺畅。与以往一知半解数据库只会建数据表的开发相比，有了巨大的进步。很感谢这次课程。让我亲身体验了当ORACLE DBA 的感觉。希望未来有机会真正参与DBA相关工作。

### 前台系统模块（负责人：张津赫、曹伟）

#### 功能说明（加流程图）

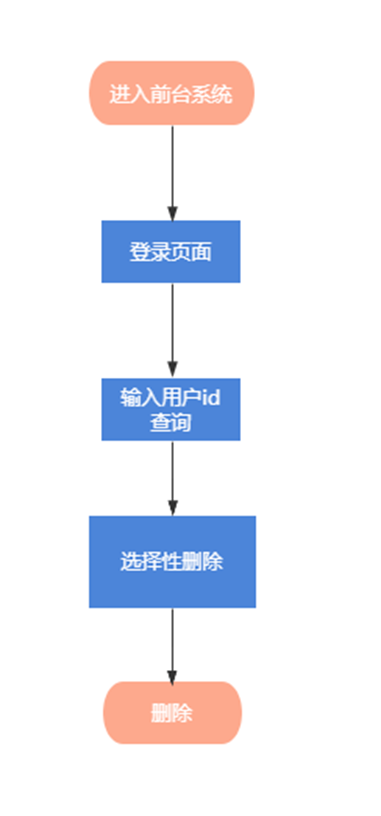
1） 开房间功能：

实现员工在前台系统中进行新房间开启功能。可根据用户输入的房间号查询该包房使用情况，再填入用户的id和使用时间段完成开房的功能。



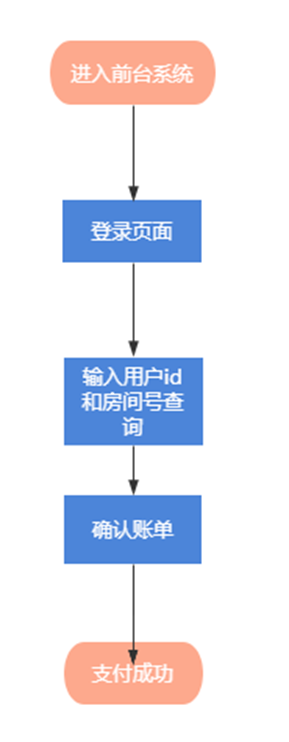
2） 取消预约功能：

通过用户id查询该用户房间预约情况，员工可选择性的对某条预约记录进行删除。



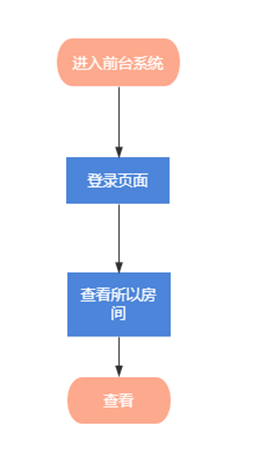
3） 结账功能：

根据客户id和房间号查询订单情况，查看账单无误后可选择确认支付。



4） 查看房间功能：

可查看所有房间的房型和价格

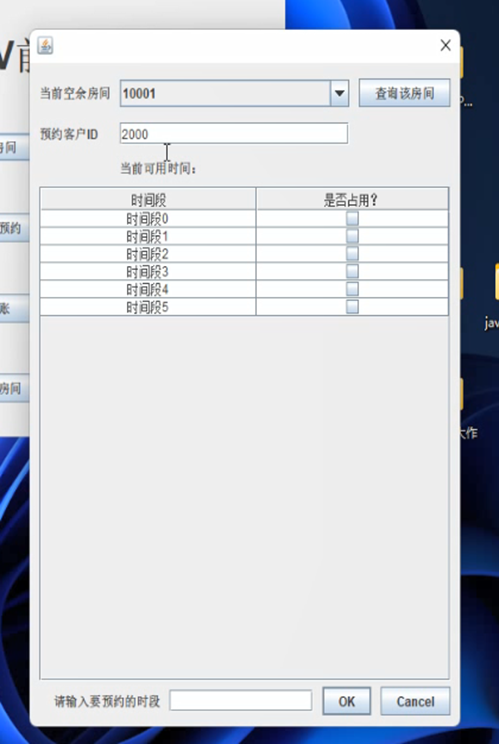


#### 测试及实现界面截图

1.前台系统整体界面

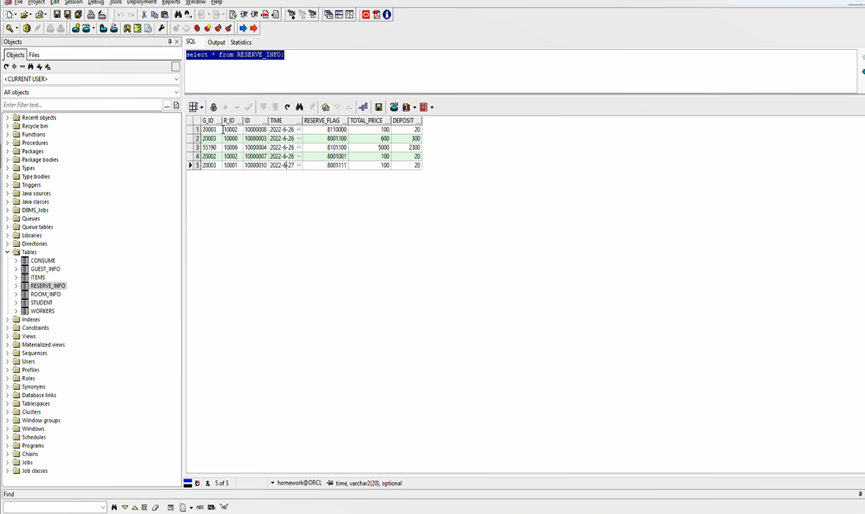


2.开房功能

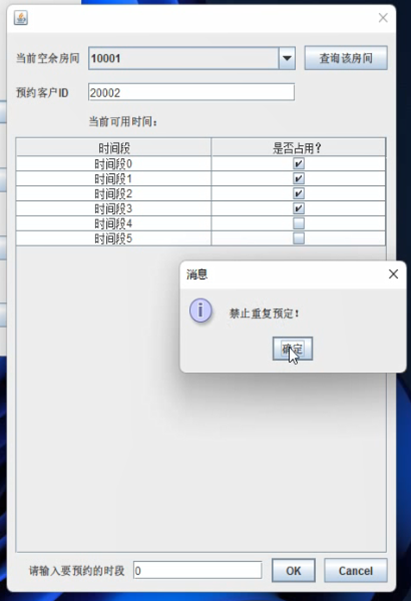




可以看到数据已加入到了数据库中



重复预订弹出提示：



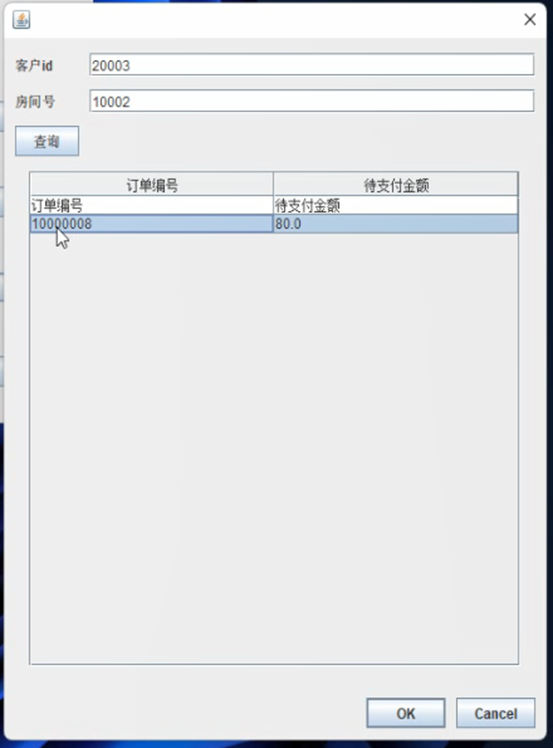
3.取消预约



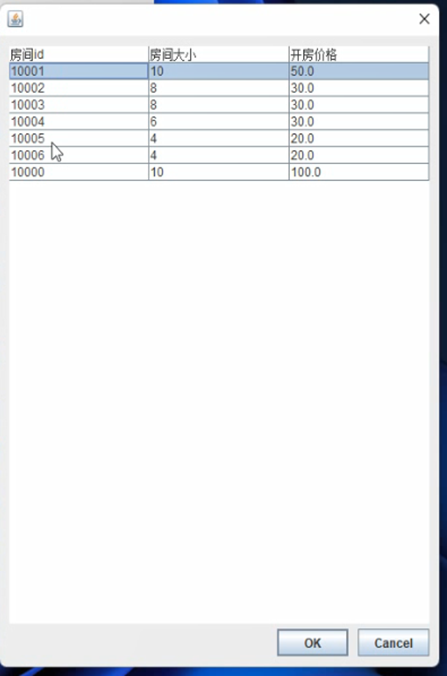
可以看到数据库中该条信息已被删除



4.结账功能



5.查看房间信息功能



#### 重点代码展示

1.RESERVE\_FLAG编码的转换

RESERVE\_FLAG的每一位表示一个时间段是否被预定的信息。在使用时要进行转码。

//将多条预定信息合并进行转码   
public static int put\_together(int flag1,int flag2)   
 {   
 return flag1+flag2-8000000;   
 }   
   
//将预定的时间段信息编码   
for(int i = 0;i < strs.length; i++){   
   
 if(table.getValueAt(Integer.parseInt(strs[i]), 1).toString().compareTo("true") == 0)   
 {   
 JOptionPane.showMessageDialog(null,"禁止重复预定！");   
 return ;   
 }   
 switch(Integer.parseInt(strs[i]))   
 {   
 case 0:   
 sum=sum+1;   
 break;   
 case 1:   
 sum=sum+10;   
 break;   
 case 2:   
 sum=sum+100;   
 break;   
 case 3:   
 sum=sum+1000;   
 break;   
 case 4:   
 sum=sum+10000;   
 break;   
 case 5:   
 sum=sum+100000;   
 break;   
 default:   
 break;   
   
 }   
   
 }

2.Oracle函数的调用：

//此函数作用，根据一个预定信息的ID，计算出还需要补交的费用。   
   
String sql = "{?=call RoomIsPaid(?)}";   
 CallableStatement pstmt = man.conn.prepareCall(sql);   
 pstmt.setString(2, resid\_arr.get(i));   
 pstmt.registerOutParameter(1, OracleTypes.FLOAT);   
 pstmt.execute();   
   
 if(Float.parseFloat(pstmt.getObject(1).toString())<=0)   
 {   
 resid\_arr.remove(i);   
 }   
 else   
 {   
 rowData[k]=new Object[]{resid\_arr.get(i),pstmt.getObject(1).toString()};   
 k++;   
 }   
 pstmt.close();

#### 心得体会

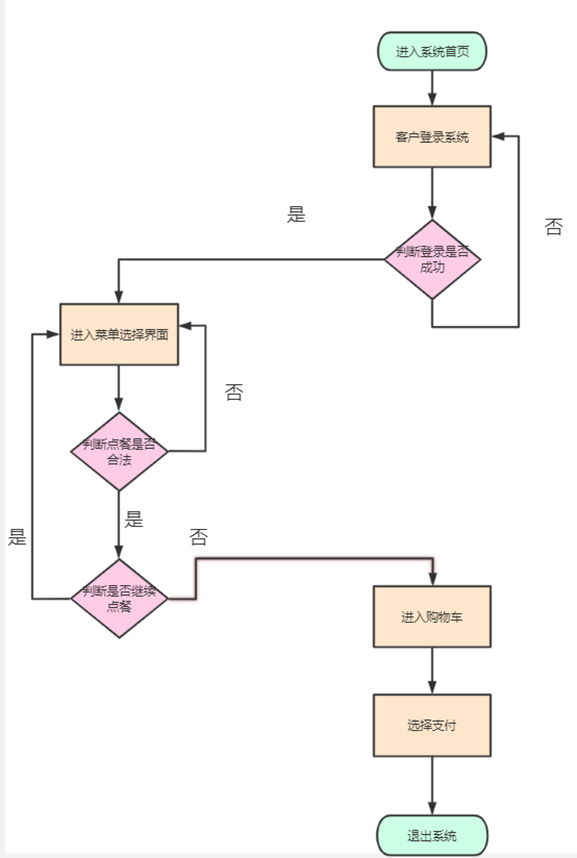
这次项目，把之前学到的 java 数据库操作从理论带到了实践，也学到了很多 oracle 相关的操作知识，收获满满。在队伍当中大家都很积极互相帮助，是一次宝贵的经历。

### 包内点餐模块（负责人：杨旭）

#### 功能说明（加流程图）

本模块用于 包间内客户点餐及付款结账。客户可以先通过登录系统，进入菜单展示界面，然后选择想要购买的菜品并放入购物车，客户可以随时进行购物车的查看。选择好菜品后，打开购物车可以进行餐品价格的计算，并能进行付款。

流程图：



#### 部分代码展示

登陆验证：

public int Judge(String id,String password) throws SQLException {   
 if(id==null||password==null)   
 return 0;//为空返回0   
 String sql="select \* from GUEST\_INFO ";   
 Statement stmt=conn.createStatement();   
 ResultSet rs = stmt.executeQuery(sql);   
 while(rs.next()){   
 String id1=rs.getString("id");   
 String password1=rs.getString("password");   
 if(id1.equals(id) && password1.equals(password))   
 return 1;//验证成功返回1   
 }   
 return 2;//错误返回2   
   
}

读取消费品信息：

public Object[][] getItems() throws SQLException {   
 Statement stmt=conn.createStatement();   
 String sql="select \* from items";   
 ResultSet rs=stmt.executeQuery(sql);   
 int row=rs.getRow();   
 int i=0;   
 Object [][]str=new String[row][4];   
 while(i<row){   
 str[i][0]=rs.getString("id");   
// System.out.println(obj[i][0]);   
 str[i][1]=rs.getInt("amount");   
 str[i][2]=rs.getString("name");   
 str[i][3]=rs.getFloat("PRICE");   
// System.out.println(obj[i][0].toString());   
 i++;   
 };   
 return str;   
 }

判断消费品选择是否合法：

public int Judge(String str) throws SQLException {   
 oracle\_connect orc = new oracle\_connect();   
 Connection conn = orc.get\_Connection();//获取连接   
   
 String sql="select id from items";   
 Statement stmt=conn.createStatement();   
 ResultSet rs=stmt.executeQuery(sql);   
 while(rs.next()){   
 String s=rs.getString("id");   
 if(s.equals(str))   
 return 1;   
 }   
 return 0;   
 }

查询和获取消费品对象信息：

public Item chaxun(String id, String order) throws SQLException {   
 oracle\_connect orc = new oracle\_connect();   
 Connection conn = orc.get\_Connection();//获取连接   
 String sql="select \* from items where id="+id;   
 Statement stmt=conn.createStatement();   
 ResultSet rs=stmt.executeQuery(sql);   
 while(rs.next()) {   
 String s = rs.getString("id");   
 int amount=rs.getInt("amount");   
 String name = rs.getString("name");   
 String price = rs.getString("price");   
 int order1=Integer.parseInt(order);   
 Item it=new Item(s,amount,name,price,order1);   
 return it;   
 }   
 return null;   
 }

购物车结算：

public float jisuan(){   
 int row=shopcar.size();   
 float sum=0;   
 for(int i=0;i<row;i++){   
 String s1=shopcar.get(i).getPrice();   
 int s2=shopcar.get(i).getorder();   
 float s=Float.valueOf(s1);   
 sum=sum+s\*s2;   
 }   
 return sum;   
 }

#### 实现界面截图

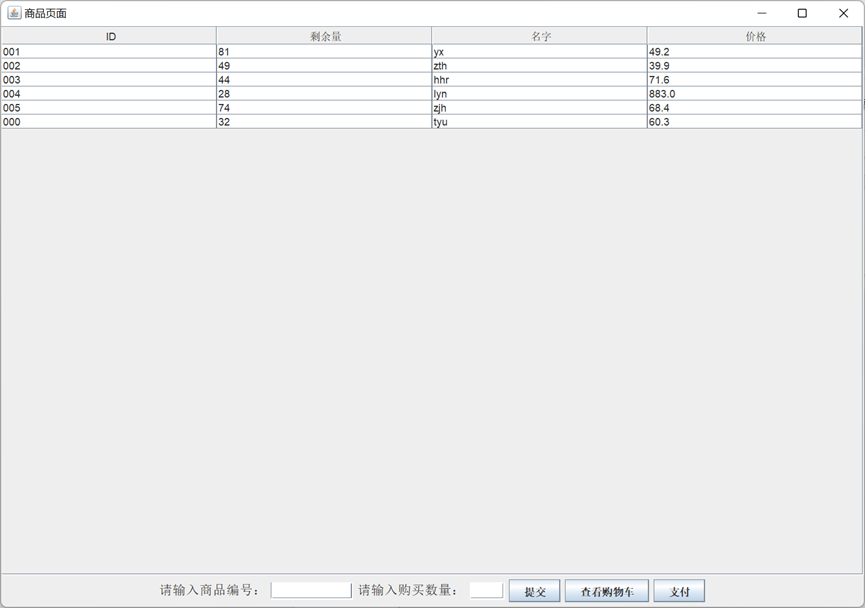
点餐系统登录界面：



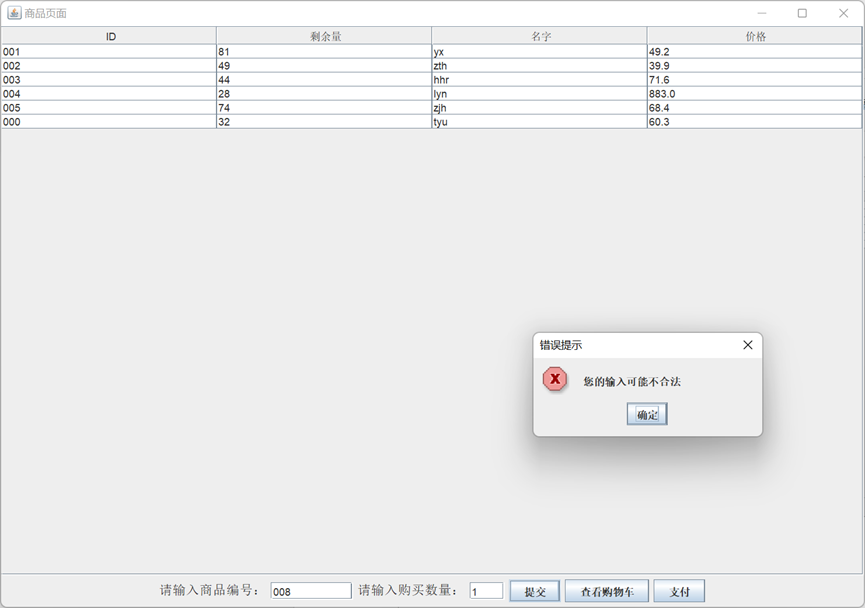
登录出现错误界面：

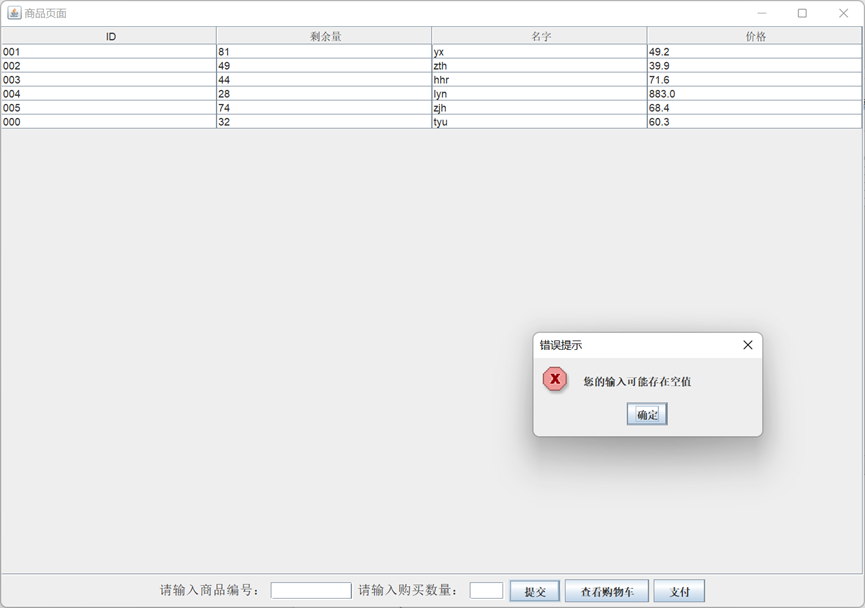


菜单显示界面：

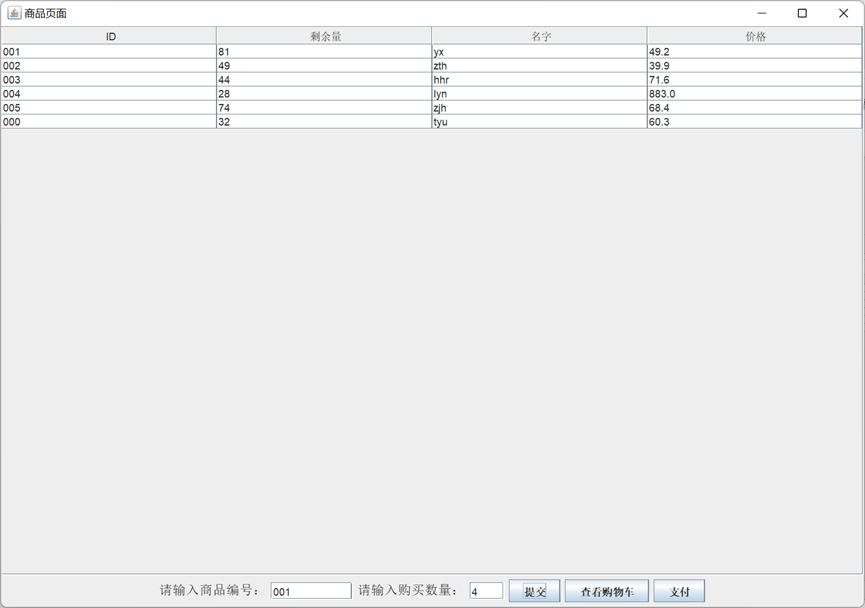


购买判断是否合法界面：





提交成功后界面：



购物车界面：



支付界面：

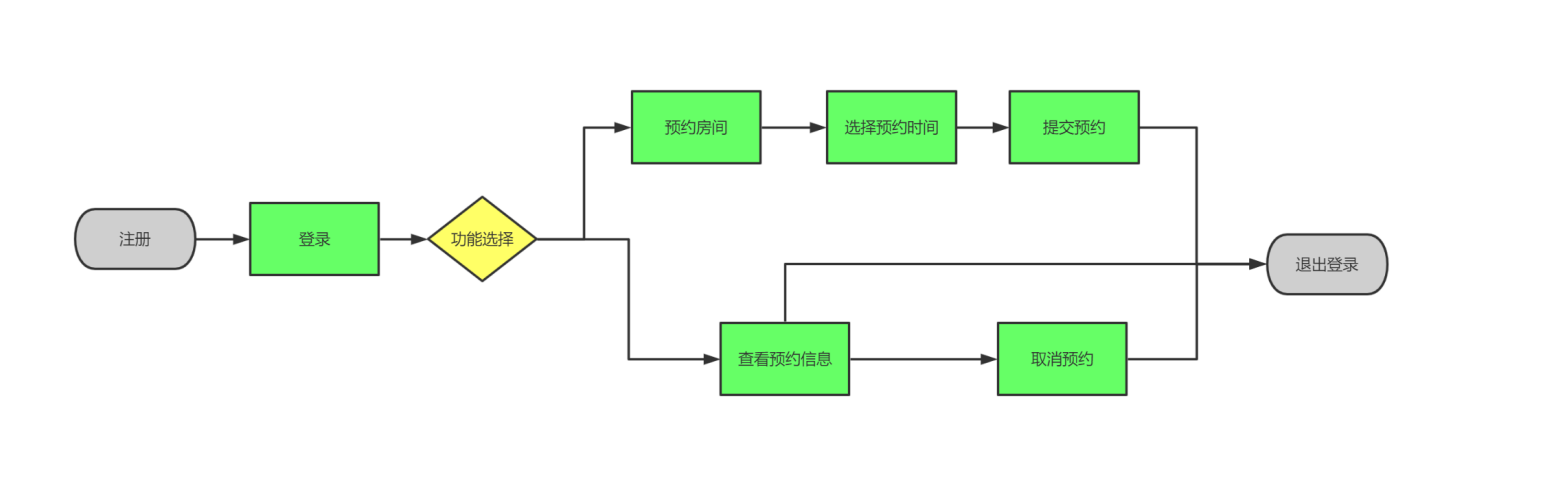


#### 心得体会

本次课程，我主要完成了后端的部分代码，通过与Oracle数据库的交互，完成了任务，更加熟悉了java语言和对数据库的操作，但是同时，我感觉我也有不足，我的界面比较简陋，点餐系统仅完成了最基本的功能，相对鲁棒性不强，很容易崩溃等等。但是这门课给我带来的影响是深远的，我相信他对我的未来是有益的！

### 网页端预定模块（负责人：张天浩）

#### 功能说明（加流程图）

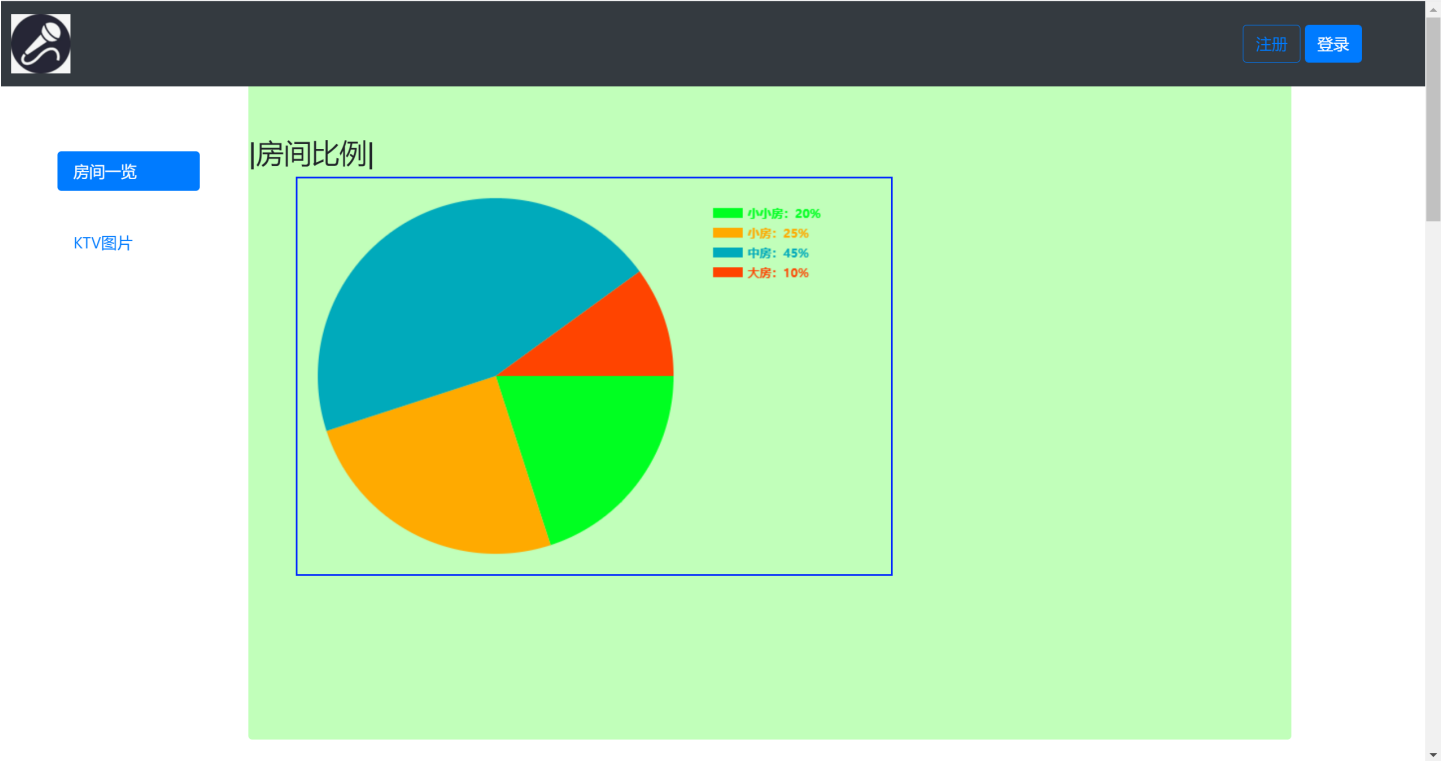


#### 部分代码展示

详细说明见，附件1。

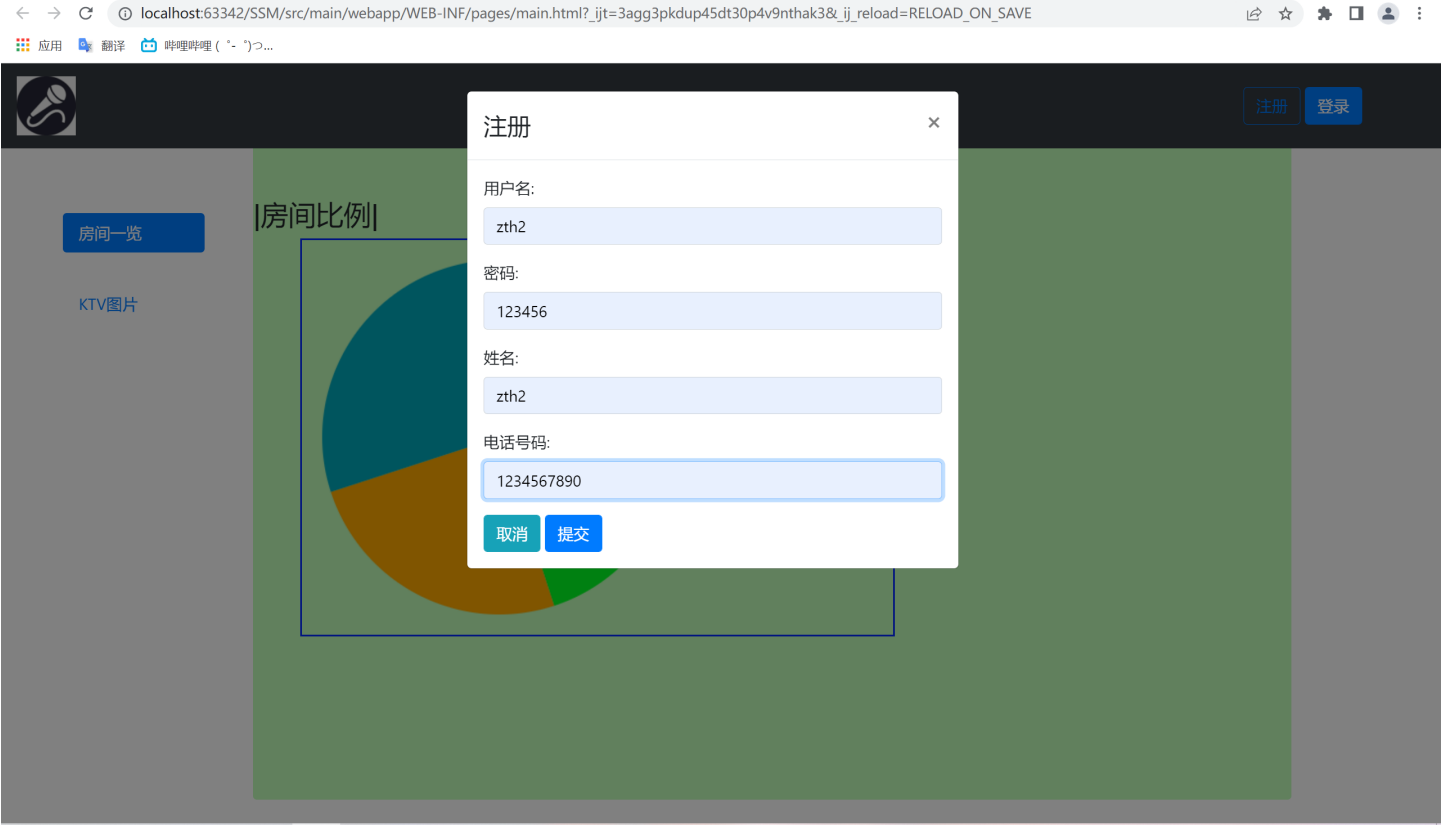
#### 测试及实现界面截图

起始界面

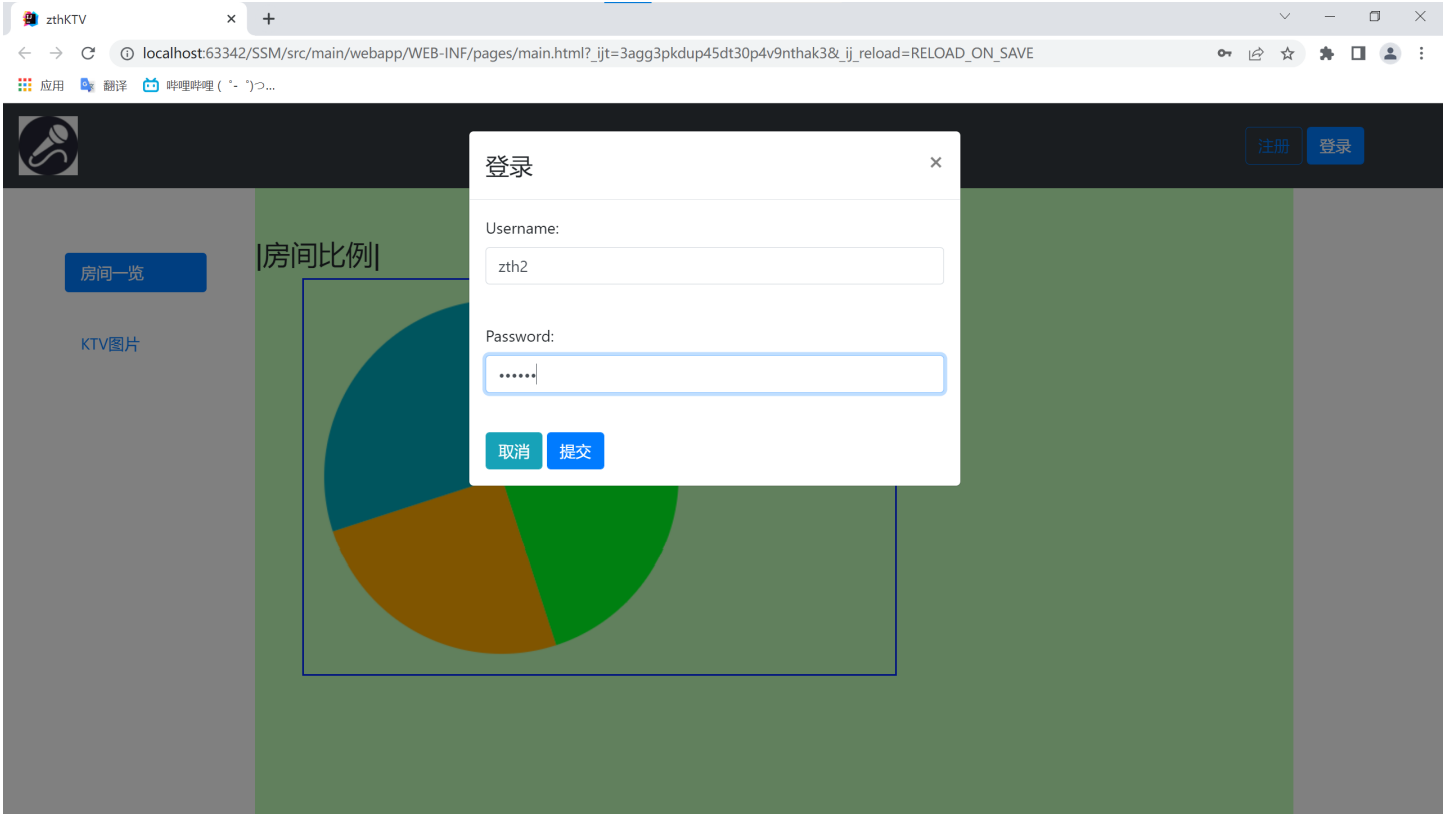


注册

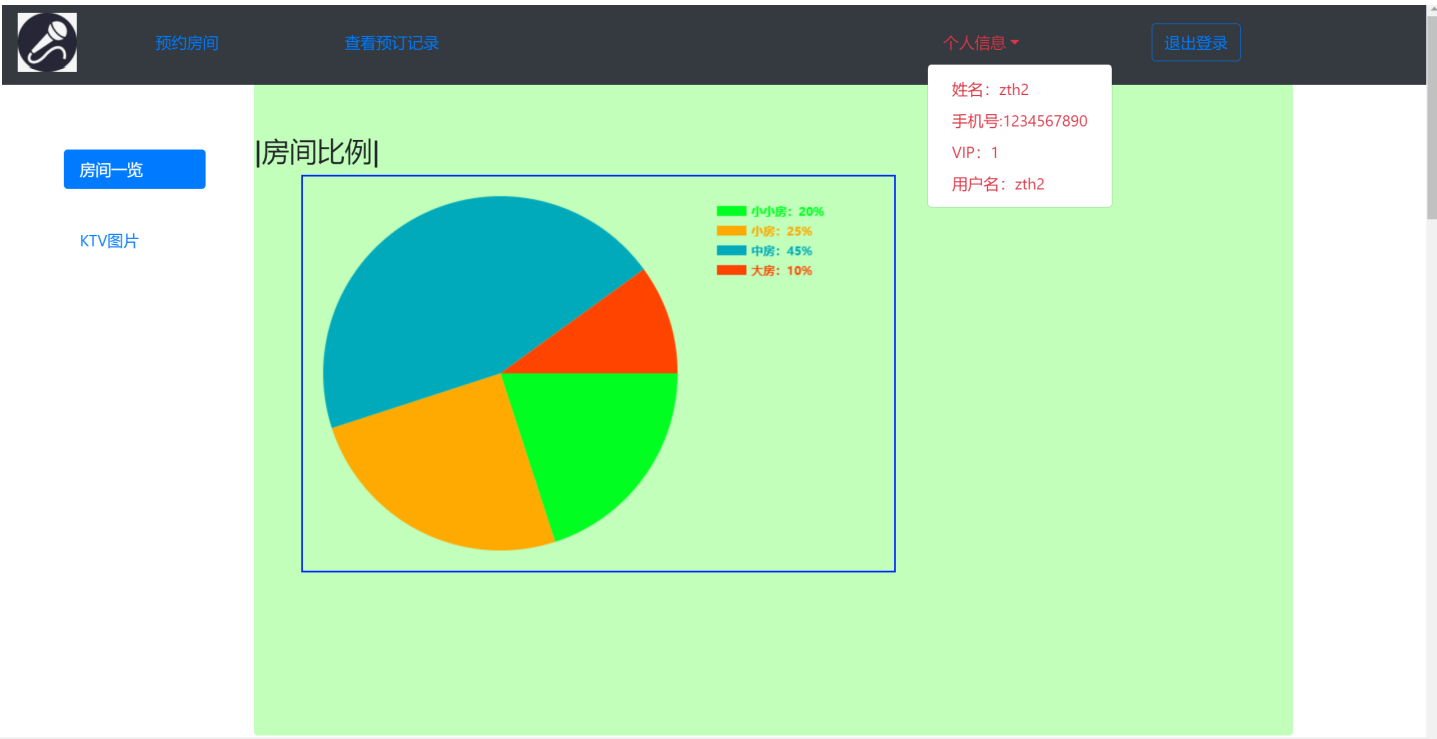
使用ZTH2展示



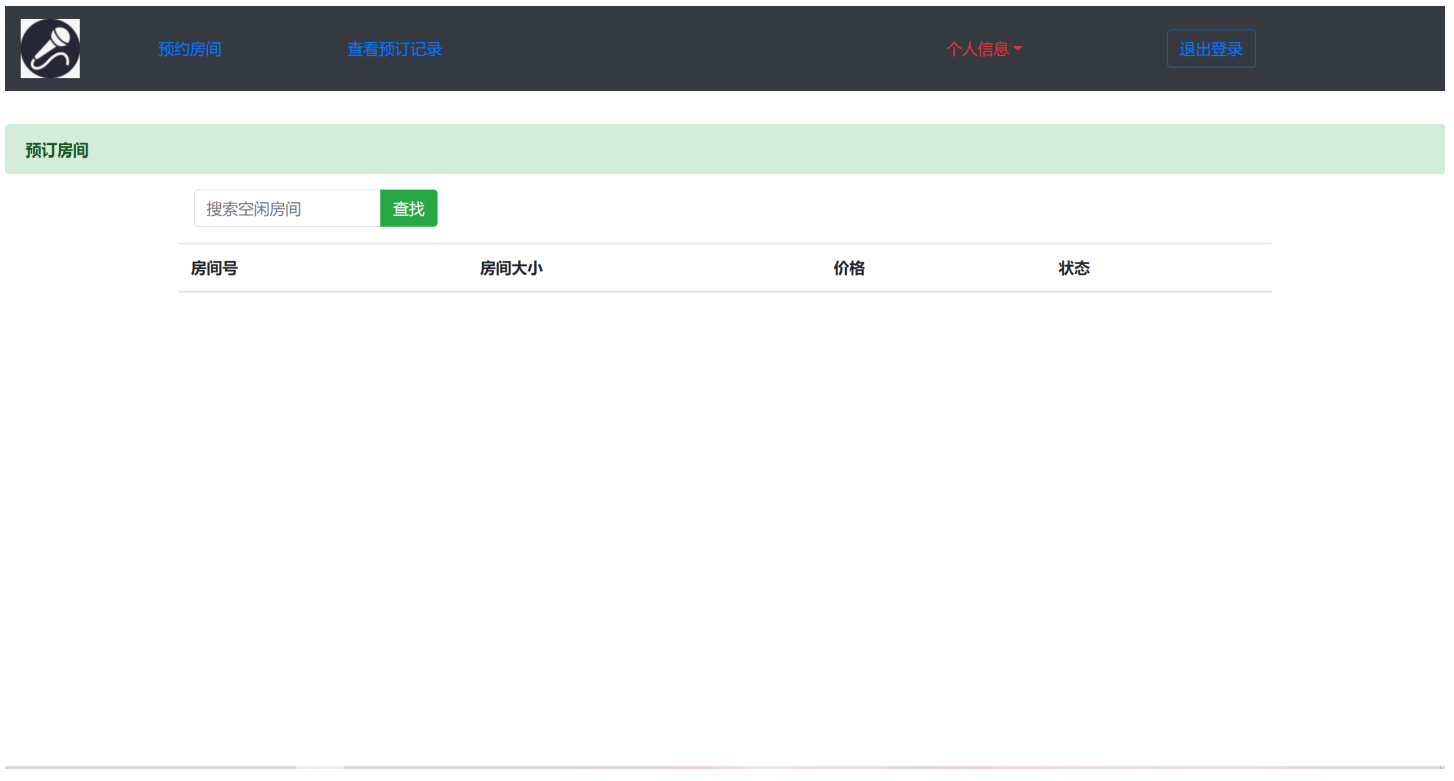
登录



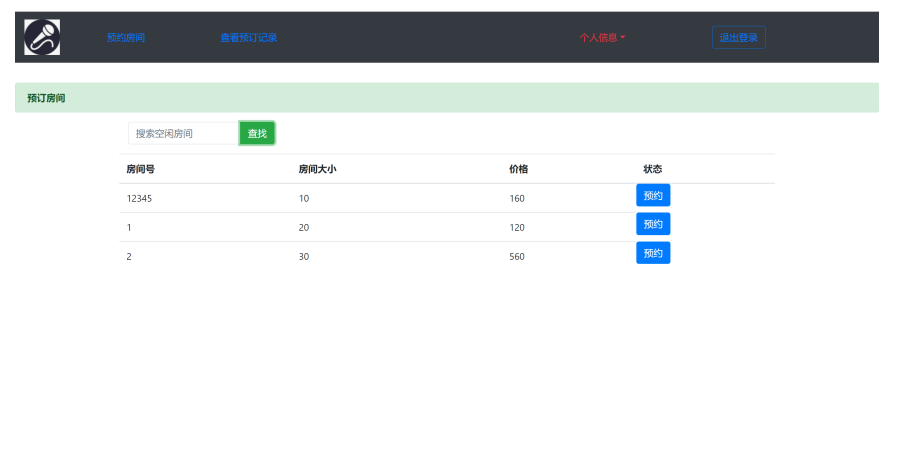
主界面



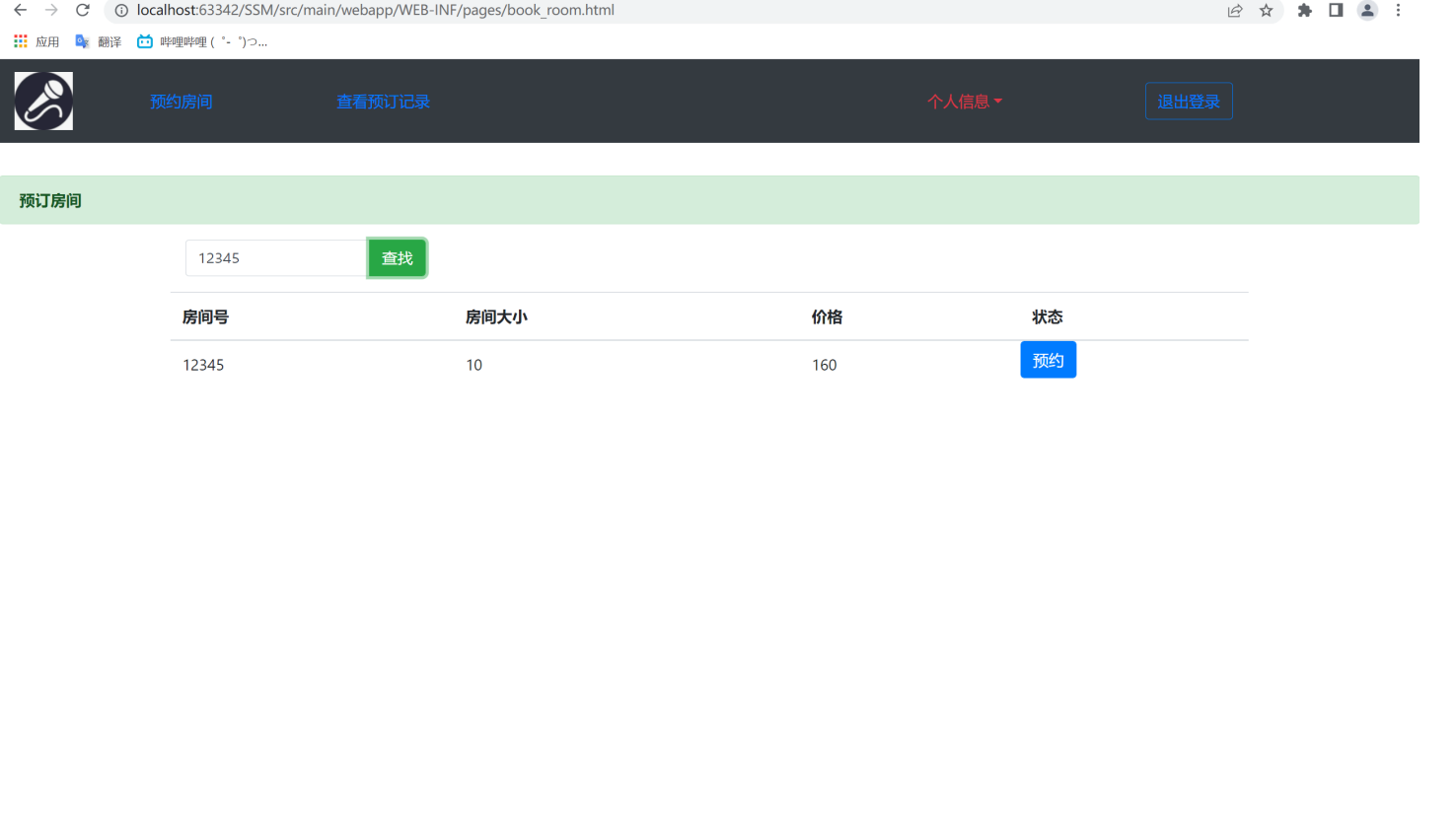
预约房间界面



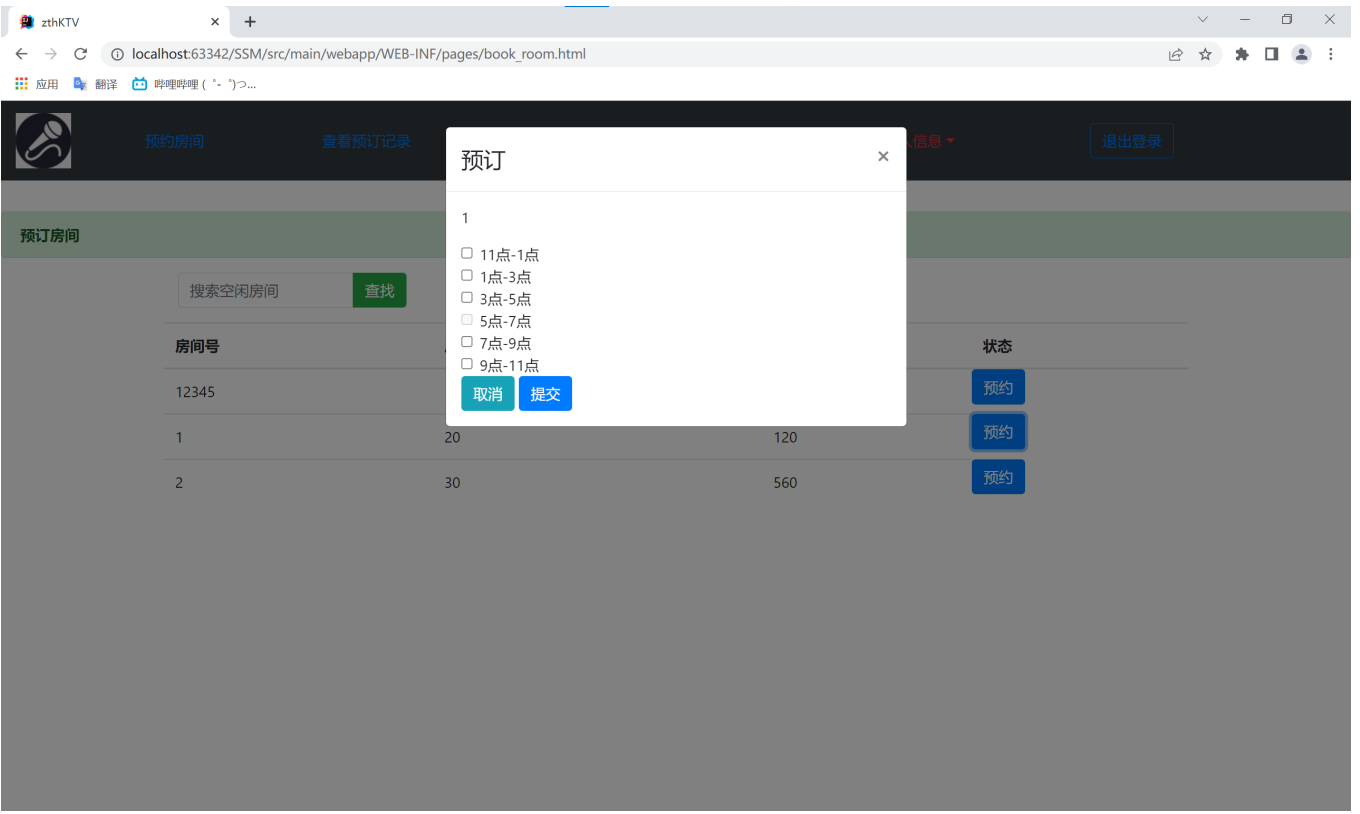
空白查找房间



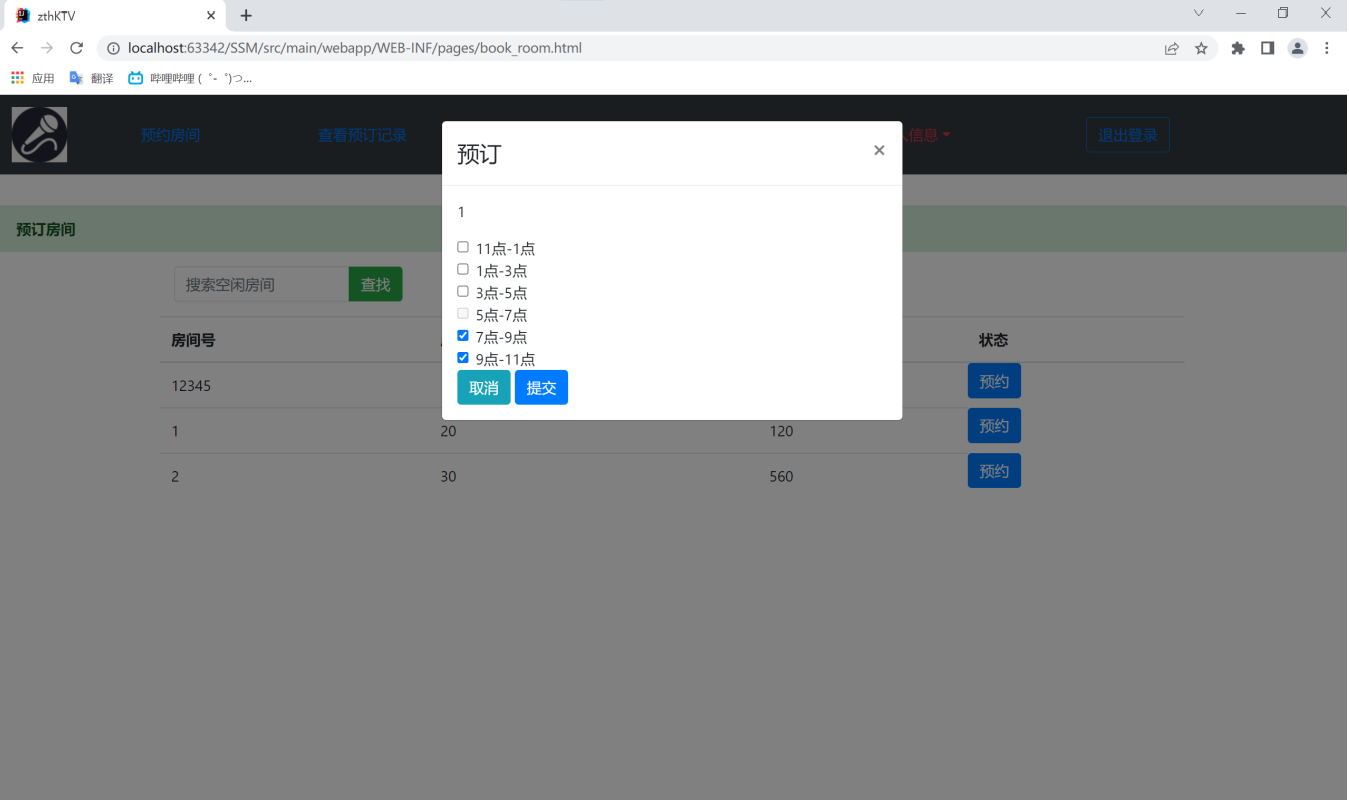
输入查找房间



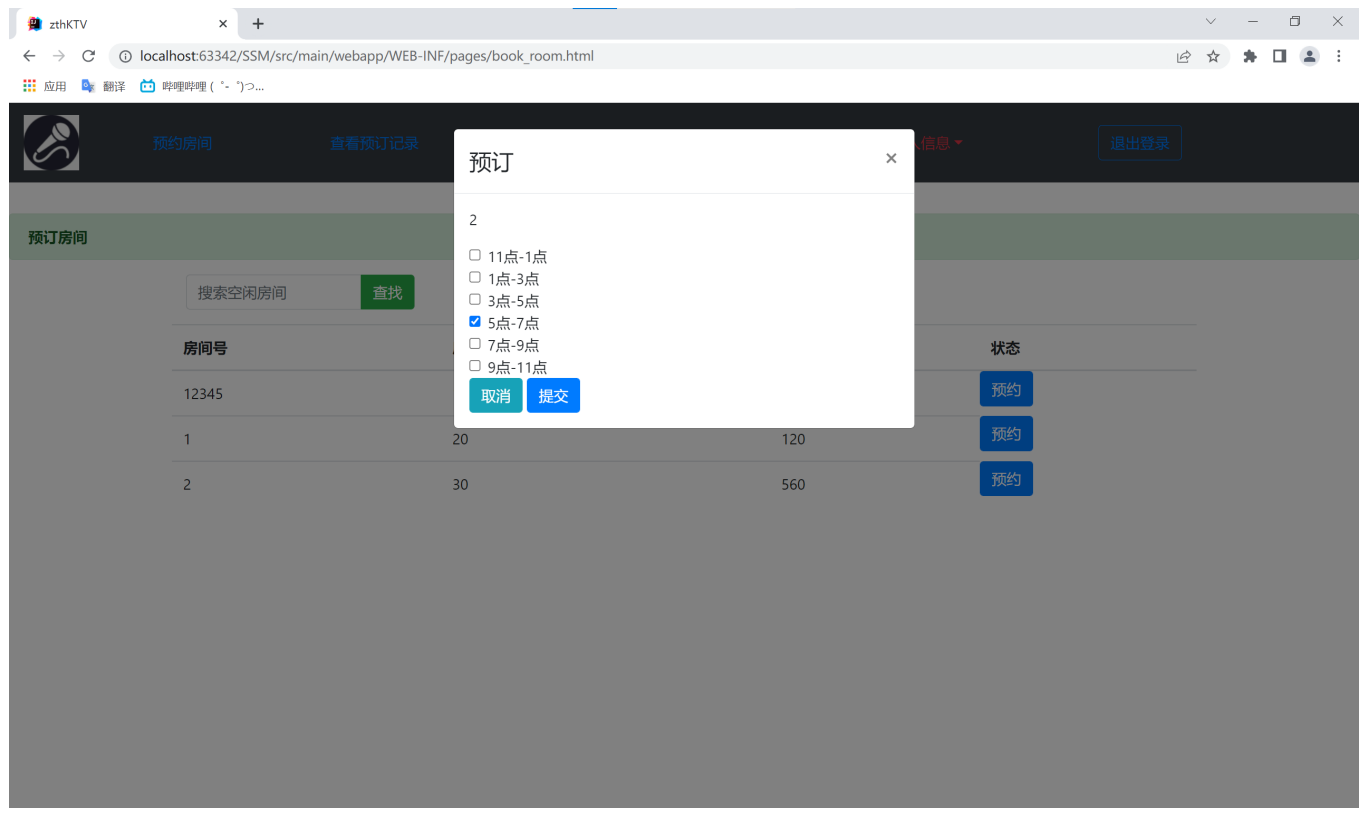
预约房间 无法选择的复选框代表已经被人占用



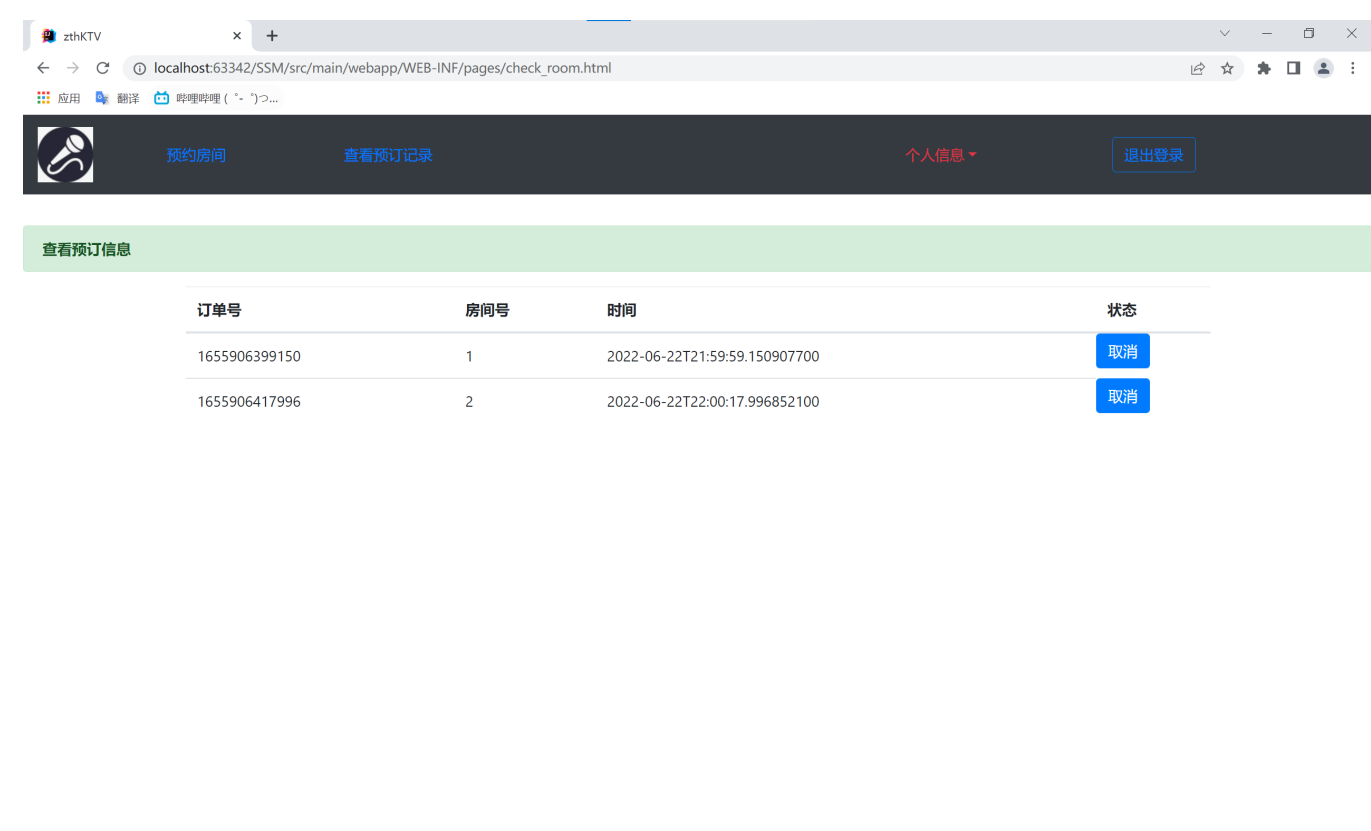
选择1号房间7-11点的框



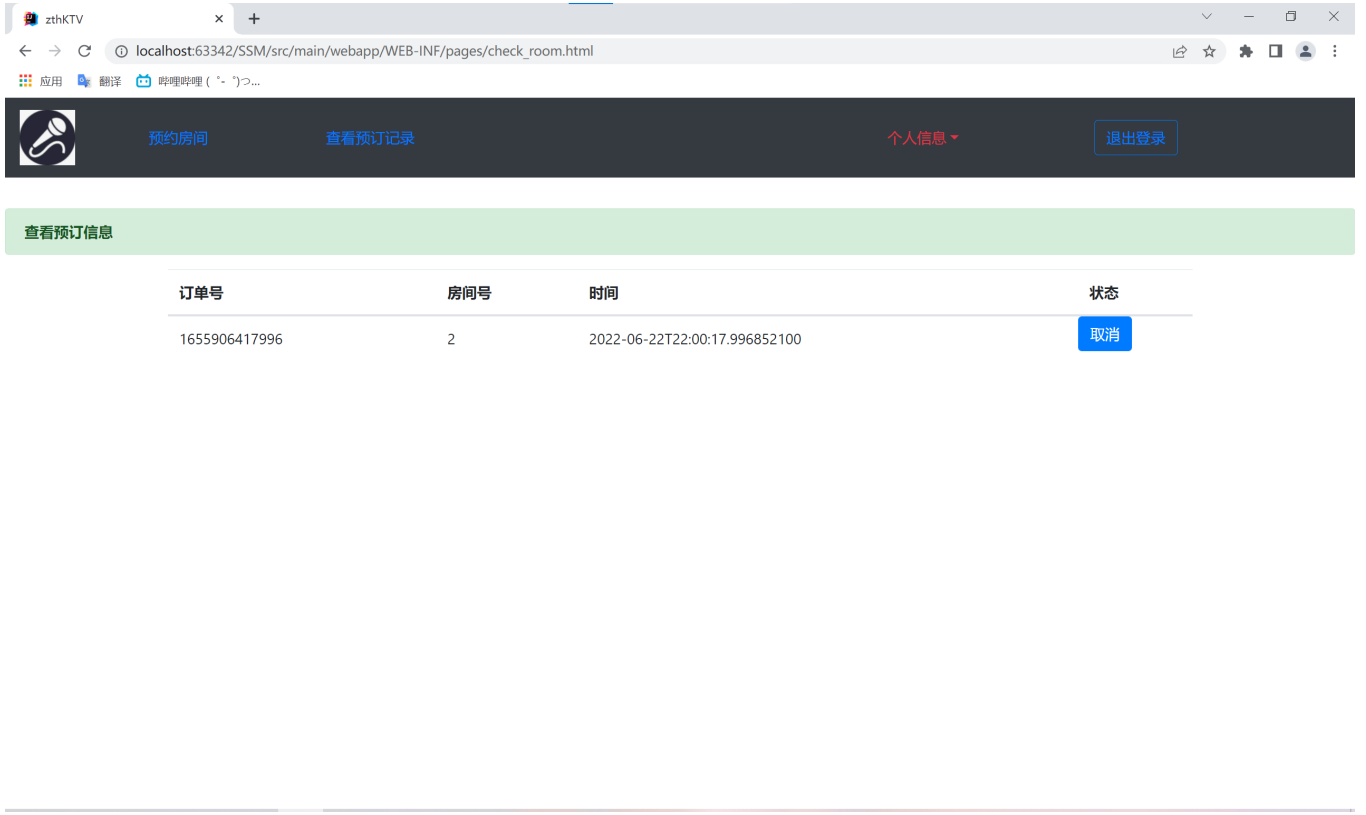
再选择2号房间5-7点的框



查看预约信息



进行预约取消 取消1号房间的预约



#### 心得体会

这次软件开发实践课程我负责的是B/S部分。在完成KTV预约系统的过程中，我加强了对前端html 代码，css选择器，js，jquery的熟练程度，通过学习前端内容，我了解到了网页的组成，布局和控制是 如何完成的。后端使用的SpringMVC框架为开发后端提供了很大的便利，通过mybatis组件，我不再需 要对数据库表和实体类进行一一映射，只需要通过mybatis逆向工程就可以自动生成所有的实体类和数 据库的增删改查功能。与学校学习的内容相比，在实践中运用的软件与框架都更加高效便捷，但学校学 校的内容作为运用软件和框架的基础，能够使我们能够更快地理解如何使用软件和框架。 在这次课程中，小组内成员所有的交流均需要通过线上方式完成，在合作过程中，线上方式相比线 下方式有不少问题需要解决，比如交流问题，积极性问题等，我们通过各种方式，克服了线上方式的问 题，成功的完成了本次的项目

## 六、总结

本次综合实践开发，我们小组每个人都很积极的参与各个环节的开发阶段，从需求分析开始一步步开发，切身体验了类似企业的开发流程，领会到了本次综合实践课程的真正用意。每个人都在其中受益良多。但是在开发过程中我们还是遇到了许多困难。第一：一开始计划使用C++的OCCI来进行系统开发。但是由于数据库和OCCI兼容问题。一直出现无法使用数据库的情况。然而网络上关于OCCI开发的资料少之又少。最终不得已选择使用JAVA进行开发。在未来有机会会再次尝试使用OCCI。第二：疫情原因，沟通环节难度增加，多次出现开发中的歧义。但经过多次的共同探讨渐渐对开发有了一致的理解。沟通问题才得以解决。而在这个问题的解决过程中，小组每名成员的沟通能力都有了很大进步。

最后，我们小组在本门课程中受益良多，感谢老师这一学期的耐心教导。