北京工业大学 经管学院

火车票预订系统

颜雨祺 18110218 王慧 18110219 自来乐.白克力 18110260

颜雨祺 王慧 自来乐.白克力
2020/5/10

小组信息和软件信息

小组成员：颜雨祺 18110218
王慧 18110219
自来乐. 白克力 18110260

小组分工：

颜雨祺：报告——系统分析部分；整合报告；数据库设计和数据编写；主页和查询界面制作、程序编写；火车车次详情界面制作、程序编写；高铁动车车次详情界面制作、程序编写；订单详情界面制作、程序编写；订单详情 1 界面制作、程序编写；火车车次添加界面制作、程序编写；高铁动车车次添加界面制作、程序编写；修改密码界面制作、程序编写；系统错误修改与完善。

王慧：报告——需求分析部分；答辩视频；数据库设计和数据编写；登录界面制作、程序编写；乘客个人中心界面制作、程序编写；管理员界面制作、程序编写；退票界面制作、程序编写；火车车次添加程序编写；高铁动车车次添加程序编写；火车车次取消程序编写；高铁动车车次取消程序编写；系统错误修改与完善。

自来乐.白克力：报告——数据库设计部分；答辩 PPT；数据库设计和数据编写；留言界面制作、程序编写；火车车次取消界面制作、程序编写；高铁动车车次取消界面制作、程序编写；查看乘客信息界面制作、程序编写；管理乘客密码界面制作、程序编写；删除乘客信息界面制作、程序编写；发布公告界面制作、程序编写；留言显示界面制作、程序编写；美化界面。

开发工具及其版本说明：

Product	Version	Architecture	Quick Action
MySQL Server	8.0.19	X64	Reconfigure
MySQL Workbench	8.0.19	X64	
Connector/J	8.0.19	X86	

NetBeans IDE 8.1

目录

一、系统需求分析 4
1. 背景 4
2. 功能 4
3. 功能结构图 4
二、数据库分析 5
1. admin_ 6
2. hsorder 6
3. hsrailway 6
4. hsrailway_price 7
5. message 7
6. notice 7
7. passenger 8
8. torder 8
9. train 9
10. train_price 9
三、系统实现 10
1. 实体类的实现 10
1.1 database 类的实现 10
1.2 admin_类的实现 11
1.3 hsorder 类的实现 11
1.4 hsrailway 类的实现 14
1.5 hsrailway_price 类的实现 16
1.6 message 类的实现 18
1.7 notice 类的实现 18
1.8 passenger 类的实现 19
1.9 torder 类的实现 21
1.10 train 类的实现 24
1.11 train_price 类的实现 26
2. control 类的实现 28
3. 表单对象类的实现 46
4. 判断类的实现 46
5. 边界类的实现 46
5.1 登录界面 46
5.2 主页和查询界面 49
5.3 火车车次详情界面 50
5.4 高铁动车车次详情界面 56
5.5 乘客个人中心界面 60
5.6 修改密码 62
5.7 订单详情 1（<60 岁的乘客出现该界面） 64
5.8 订单详情（>=60 岁的乘客出现该界面） 66
5.9 退票 66
5.10 留言界面 71

5.11 管理员界面 72
5.12 火车车次添加界面 74
5.13 高铁车次添加界面 77
5.14 火车车次取消界面 77
5.15 高铁车次取消界面 79
5.16 查看乘客信息界面 80
5.17 乘客密码管理（管理乘客密码界面） 82
5.18 删除乘客信息界面 84
5.19 发布公告 87
5.20 留言管理（留言显示界面） 88
四、课程设计体会 90
颜雨祺 18110218： 90
王慧 18110219： 90
自来乐·白克力 18110260： 90

火车票预订系统

18110218 颜雨祺、18110219 王慧、18110260 自来乐.白克力

一、系统需求分析

1. 背景

在互联网高速发展的时代，网络预订火车票已经越来越常见。通过手机、电脑就可以轻松预定火车票，极大地方便了人们的生活。因此，我们小组决定开发设计一套火车票预订系统。

2. 功能

本系统主要功能为为乘客提供查询、预定、退票及老年人优先选择上下铺的服务。

从乘客角度看：

（1）注册：首次使用本系统需先进行注册

（2）登录：已有账号或已注册过的用户，可以进行登录

（3）查询：乘客可以根据出发日期、出发地点、到达地点、是否乘坐高铁动车进行查询

（4）预定：查询出符合要求的列车信息，点击方可预定，若年龄大于 60 岁的乘客，可以优先选择上下铺

（5）订单详情及退票：乘客预定后，可以进行订单查询及退票

（6）修改密码：乘客可在个人中心页面修改密码

（7）留言：在个人中心页面可以点击留言，进行反馈

从管理员角度看：

（1）登录：管理员可以通过登录权限进入管理员模式

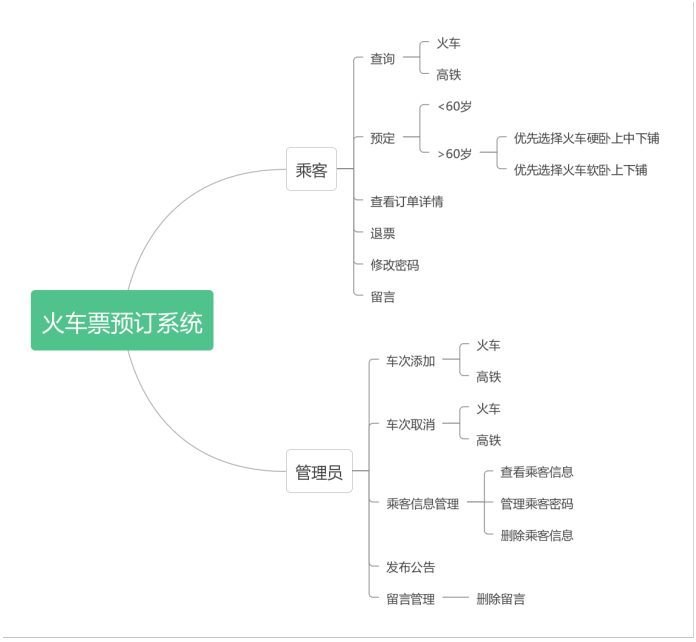
（2）车次添加及取消：管理员可以对火车车次和高铁车次进行添加或取消

（3）乘客信息管理：管理员可以查看乘客信息，管理乘客密码及删除乘客信息

（4）发布公告：管理员可以进行公告的发布，公告将在查询界面显示

（5）留言管理：管理员可以对留言进行删除

3. 功能结构图



通过上述功能，可以实现网上火车票查询、预定、退票功能，还可以针对老年人进行优选选座，足不出户便可预定车票，方便快捷，减少资源浪费。

二、数据库分析

本火车票预订系统中设有十张表，表设定如下：

1. admin_表中含有 A_id、A_password。A_id 是主键。
2. hsorder 表中含有 H_name、P_id、H_seattype、nowtime、H_date。H_name 是主键。
3. hsrailway 表中有 H_id、H_name、H_leavetime、H_arrivetime、H_destination、H_departure、H_date。H_id 是主键。
4. hsrailway_price 表中有 H_id、H_name、H_seattype、H_number、H_price、H_date。H_id 是主键。
5. message 表中有 M_message。
6. passenger 表中有 P_name、P_phone、P_address、P_id、P_password、P_email。P_id 是主键。
7. notice 表中有 content。
8. torder 表中有 T_name、P_id、T_seattype、nowtime、T_date、softtype、hardtype。T_name 是主键。
9. train 表中有 T_id、T_name、T_leavetime、T_arrivetime、T_departure、T_destination、T_date。T_id 是主键
10. train_price 表中有 T_id、T_name、T_seattype、T_price、T_number、T_date。T_id 是主键。

十张表具体设计：

1. admin_

字段名	含义
A_id	管理员 id（主键）
A_password	管理员密码

2. hsorder

字段名	含义
H_name	高铁动车车次（主键）
P_id	乘客身份证号（外键）
H_seattype	高铁动车座位类型
nowtime	现在的时间
H_date	高铁动车出发日期

3. hsrailway

#	名称	数据类型	长度/集合	无符号的	允许 N...	填零	默认	注释	校对规则
1	H_id	INT	10,0				AUTO_INCREME...		
2	H_name	VARCHAR	10				无默认值		utf8mb4_0900_ai_ci
3	H_leavetime	VARCHAR	20				无默认值		utf8mb4_0900_ai_ci
4	H_arrivetime	VARCHAR	20				无默认值		utf8mb4_0900_ai_ci
5	H_departure	VARCHAR	30				无默认值		utf8mb4_0900_ai_ci
6	H_destination	VARCHAR	30				无默认值		utf8mb4_0900_ai_ci
7	H_date	VARCHAR	50				''		utf8mb4_0900_ai_ci

字段名	含义
H_id	高铁动车序号
H_name	高铁动车车次
H_leavetime	高铁动车出发时间
H_arrivetime	高铁动车到达时间
H_destination	高铁动车出发地
H_departure	高铁动车到达地
H_date	高铁动车出发日期

4. hsrailway_price

#	名称	数据类型	长度/集合	无符号的	允许 N...	填零	默认	注释	校对规则
1	H_id	INT	10,0				AUTO_INCREME...		
2	H_name	VARCHAR	10				无默认值		utf8mb4_0900_ai_ci
3	H_seattype	VARCHAR	10				无默认值		utf8mb4_0900_ai_ci
4	H_price	VARCHAR	10				无默认值		utf8mb4_0900_ai_ci
5	H_number	VARCHAR	10				无默认值		utf8mb4_0900_ai_ci
6	H_date	VARCHAR	50				''		utf8mb4_0900_ai_ci

字段名	含义
H_id	高铁动车序号（主键）
H_name	高铁车次
H_seattype	高铁座位类型
H_number	每个座位余量
H_price	每个座位对应价格
H_date	高铁出发日期

5. message

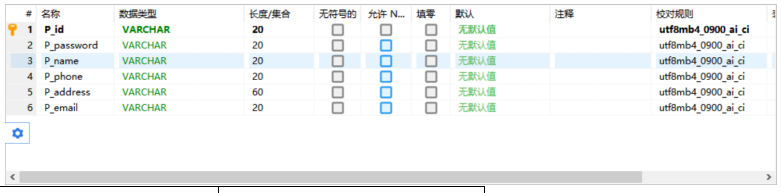
#	名称	数据类型	长度/集合	无符号的	允许 N...	填零	默认	注释	校对规则
1	M_message	VARCHAR	60		☑		NULL		utf8mb4_0900_ai_ci

字段名	含义
M_message	乘客留言内容

6. notice

字段名	含义
content	管理员发布公告内容

7. passenger



字段名	含义
P_name	乘客姓名
P_phone	乘客手机号码
P_address	乘客地址
P_id	乘客身份证号码（主键）
P_password	乘客密码
P_email	乘客电子邮箱

8. torder

字段名	含义
T_name	火车车次名（主键）
P_id	乘客身份证号码（外键）
T_seattype	火车座位类型
nowtime	现在的时间

T_date	火车出发日期
softtype	火车软卧类型（上、下铺）
hardtype	火车硬卧类型（上、中、下铺）

9. train

#	名称	数据类型	长度/集合	无符号的	允许 N...	填零	默认	注释	校对规则
1	T_id	INT	10,0	☐	☐	☐	AUTO_INCREME...		
2	T_name	VARCHAR	10	☐	☐	☐	无默认值		utf8mb4_0900_ai_ci
3	T_leavetime	VARCHAR	20	☐	☐	☐	无默认值		utf8mb4_0900_ai_ci
4	T_arrivetime	VARCHAR	20	☐	☐	☐	无默认值		utf8mb4_0900_ai_ci
5	T_departure	VARCHAR	30	☐	☐	☐	无默认值		utf8mb4_0900_ai_ci
6	T_destination	VARCHAR	30	☐	☐	☐	无默认值		utf8mb4_0900_ai_ci
7	T_date	VARCHAR	50	☐	☐	☐	''		utf8mb4_0900_ai_ci

字段名	含义
T_id	火车序号（主键）
T_name	火车车次名字
T_leavetime	火车出发时间
T_arrivetime	火车到达时间
T_departure	火车出发地
T_destination	火车到达地
T_date	火车出发日期

10. train_price

#	名称	数据类型	长度/集合	无符号的	允许 N...	填零	默认	注释	校对规则
1	T_id	INT	10,0	☐	☐	☐	AUTO_INCREME...		
2	T_name	VARCHAR	10	☐	☐	☐	无默认值		utf8mb4_0900_ai_ci
3	T_seattype	VARCHAR	10	☐	☐	☐	无默认值		utf8mb4_0900_ai_ci
4	T_price	VARCHAR	10	☐	☐	☐	无默认值		utf8mb4_0900_ai_ci
5	T_number	VARCHAR	10	☐	☐	☐	无默认值		utf8mb4_0900_ai_ci
6	T_date	VARCHAR	50	☐	☐	☐	''		utf8mb4_0900_ai_ci

字段名	含义
T_id	火车车次序号（主键）
T_name	火车车次名字
T_seattype	火车座位类型
T_price	各座位类型对应的价格
T_number	各座位类型余量
T_date	火车出发时间

三、系统实现

1. 实体类的实现

在系统中共设计了 11 个实体类，包括 database 类，admin_类，hsorder 类，hsrailway 类，hsrailway_price 类，message 类，notice 类，passenger 类，torder 类，train 类，train_price 类。database 类实现了数据的访问，admin_类，hsorder 类，hsrailway 类，hsrailway_price 类，message 类，notice 类，passenger 类，torder 类，train 类，train_price 类描述了管理员，高铁订单，高铁，高铁价格，留言，公告，乘客，火车订单，火车和火车价格的实体，分别与数据库中的 10 个表和字段一一对应。

1.1 database 类的实现

通过 database 类实现数据库访问。代码如下：

```
package newpackage;
import java.sql.Connection;
import java.sql.DriverManager;
public class database {

private String dbpath="jdbc:mysql://localhost:3306/火车票预订系
统?serverTimezone=GMT%2B8&useSSL=false";
private String dbUserName="root";
private String dbPassword="Yanyuqi0623";//s 输入你的数据库密码
private String jdbcName="com.mysql.cj.jdbc.Driver";
public database(){
}
public String GetDBPath()
{
return dbpath;
}
public void SetDBPath(String dbpath)
{
dbpath=dbpath;
}
public Connection getCon() throws Exception{
Class.forName(jdbcName);
//Connection con=DriverManager.getConnection(dbpath,dbUserName,dbPassword);
Connection con=DriverManager.getConnection(dbpath,dbUserName,dbPassword);
return con;
}
public void closeCon(Connection con) throws Exception{
if(con!=null){
con.close();
}
```

```
}
public void OpenDB()
{database db=new database();
try {
db.getCon();
System.out.println("数据库连接成功!");
} catch (Exception e) {
// TODO Auto-generated catch block
e.printStackTrace();
}
}
public    void main(String[] args) {
OpenDB();
}
}
```

1.2 admin_类的实现

```
package newpackage;
public class admin_{
private String A_id;
private String A_password;
public admin_()
{
super();
}

public String GetA_id(){
return A_id;
}
public void setA_id(String A_id)
{
this.A_id=A_id;
}

public String GetA_password(){
return A_password;
}
public void setA_password(String A_password)
{
this.A_password=A_password;
}
}
```

1.3 hsorder 类的实现

```
package newpackage;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Timestamp;
public class hsorder extends hsrailway{

private String H_name;
private String H_seattype;
private String P_id;
private Timestamp Nowtime;
private String H_date;

public hsorder(){
      super();
}
public hsorder(String H_id,String H_name){

}
public String GetP_id(){
return P_id;
}
public void setP_id(String P_id)
{
this.P_id=P_id;
}
public String GetH_seattype(){
return H_seattype;
}
public void setH_seattype(String H_seattype)
{
this.H_seattype=H_seattype;
}

public String GetH_name(){
return H_name;
}
public void setH_name(String H_name)
{
this.H_name=H_name;
}
public Timestamp GetNowtime(){
return Nowtime;
```

```
}
public void setNowtime(Timestamp Nowtime)
{
this.Nowtime=Nowtime;
}
public String GetH_date(){
return H_date;
}
public void setH_date(String H_date)
{
this.H_date=H_date;
}
public ResultSet PassengerList(Connection con, passenger passenger) throws SQLException {
StringBuffer sb = new StringBuffer("select
hsrailway.H_name,hsrailway.H_departure,hsrailway.H_destination,hsrailway.H_leavetime,hs
railway.H_arrivetime,hsorder.H_seattype,hsrailway_price.H_price,hsorder.H_date,hsrailway
_price.H_date from hsrailway_price,hsrailway,hsorder where
hsrailway.H_name=hsorder.H_name and hsrailway.H_id=hsrailway_price.H_id and
hsrailway_price.H_seattype=hsorder.H_seattype and
hsrailway_price.H_date=hsorder.H_date");

if (StringUtil.isNotEmpty(passenger.GetP_id())) {
sb.append(" and P_id = '" + passenger.GetP_id() + "'");
}
System.out.println(sb.toString());
PreparedStatement pstmt = con.prepareStatement(sb.toString());
return pstmt.executeQuery();
}
public int SelectionCancel(Connection con)throws Exception{
String sql="delete from hsorder where H_name=? and H_date=?";
PreparedStatement pstmt=con.prepareStatement(sql);

pstmt.setString(1, H_name);
pstmt.setString(2, H_date);
return pstmt.executeUpdate();
}

public int NumSelectedMinus(Connection con)throws Exception{
String sql="update hsorder set numSelected = numSelected - 1 where H_name=?";
PreparedStatement    pstmt=con.prepareStatement(sql);
pstmt.setString(1, H_name);
return pstmt.executeUpdate();
}
```

```
public int SelectionAdd(Connection con)throws Exception{
String sql="insert into hsorder (H_name,P_id,H_seattype,Nowtime,H_date)
values(?,?,?,?,?)";
PreparedStatement pstmt=con.prepareStatement(sql);
pstmt.setString(1,H_name);
pstmt.setString(2,P_id);
pstmt.setString(3,H_seattype);
pstmt.setTimestamp(4,Nowtime);
pstmt.setString(5,H_date);
return pstmt.executeUpdate();
}
public int H_numberMinus(Connection con,String H_name,String H_seattype,String
H_date)throws Exception{
String sql="update hsrailway_price set H_number = H_number - 1 where H_name=? and
H_seattype=? and H_date=?";
PreparedStatement    pstmt=con.prepareStatement(sql);
pstmt.setString(1, H_name);
pstmt.setString(2, H_seattype);
pstmt.setString(3, H_date);
return pstmt.executeUpdate();
}
public    ResultSet isExisted(Connection con) throws SQLException{
String sql = "select * from hsorder where P_id=? and H_name=? ";
PreparedStatement pstmt = con.prepareStatement(sql);
pstmt.setString(1, P_id);
pstmt.setString(2, H_name);
return pstmt.executeQuery();
}
public int H_numberAdd(Connection con,String H_name,String H_seattype,String
H_date)throws Exception{
String sql="update hsrailway_price set H_number = H_number    + 1 where H_name=? and
H_seattype=? and H_date=?";
PreparedStatement    pstmt=con.prepareStatement(sql);
pstmt.setString(1, H_name);
pstmt.setString(2, H_seattype);
pstmt.setString(3, H_date);
return pstmt.executeUpdate();
}
}
```

1.4 hsrailway 类的实现

```
package newpackage;
 import java.sql.Time;
 public class hsrailway extends hsrailway_price{
```

```
static void update(String sql, String time) {
 throw new UnsupportedOperationException("Not supported yet."); //To change body of
 generated methods, choose Tools | Templates.
 }
 private String H_id;
 private String H_name;
 private Time H_leavetime;
 private Time H_arrivetime;
 private String H_departure;
 private String H_destination;
 private String H_date;
 public hsrailway()
 {
     super();
 }
 public hsrailway(String H_id)
 {
       super();
       H_id=H_id;
 }
 public hsrailway(String H_id,String H_name) {
       super();
       H_id=H_id;
       H_name=H_name;
 }

 public String GetH_id(){
       return H_id;
 }

 public void setH_id(String H_id)
 {
       this.H_id=H_id;
 }

 public String GetH_name(){
       return H_name;
 }
 public void setH_name(String H_name)
 {
       this.H_name=H_name;
 }
 public Time GetH_leavetime(){
       return H_leavetime;
```

```
}
public void setH_leavetime(Time H_leavetime)
{
    this.H_leavetime=H_leavetime;
}
public Time GetH_arrivetime(){
    return H_arrivetime;
}
public void setH_arrivetime(Time H_arrivetime)
{
    this.H_arrivetime=H_arrivetime;
}
public String GetH_departure(){
    return H_departure;
}
public void setH_departure(String H_departure)
{
    this.H_departure=H_departure;
}
public String GetH_destination(){
    return H_destination;
}
public void setH_destination(String H_destination)
{
    this.H_destination=H_destination;
}
public String GetH_date(){
    return H_date;
}
public void setH_date(String H_date)
{
    this.H_date=H_date;
}
}
```

1.5 hsrailway_price 类的实现

```
package newpackage;
public class hsrailway_price {
private String H_id;
private String H_name;
private String H_seattype;
private String H_price;
private String H_number;
private String H_date;
```

```
public hsrailway_price()
{

}
public hsrailway_price(String H_id,String H_name,String H_seattype)
{     H_id=H_id;
       H_name=H_name;
       H_seattype=H_seattype;
}

public String GetH_name()
{
       return H_name;
}
public void setH_name(String H_name)
{
       this.H_name=H_name;
}
public String GetH_seattype()
{
       return H_seattype;
}
public void setH_seattype(String H_seattype)
{
       this.H_seattype=H_seattype;
}
public String GetH_id()
{
       return H_id;
}
public void setH_id(String H_id)
{
       this.H_id=H_id;
}
public String GetH_price()
{
       return H_price;
}
public void setH_price(String H_price)
{
       this.H_price=H_price;
}
  public String GetH_number()
{
```

```
        return H_number;
    }
    public void setH_number(String H_number)
    {
        this.H_number=H_number;
    }
    public String GetH_date(){
        return H_date;
    }
    public void setH_date(String H_date)
    {
        this.H_date=H_date;
    }
    }
```

1.6 message 类的实现

```
package newpackage;
public class message {
private String M_number;
private String M_message;
public String GetM_number(){
return M_number;
}
public void setM_number(String M_number)
{
        this.M_number=M_number;
}
public String GetM_message(){
        return M_message;
}
public void setM_message(String M_message)
{
        this.M_message=M_message;
}
}
```

1.7 notice 类的实现

```
package newpackage;
public class message {
private String M_number;
private String M_message;
public String GetM_number(){
        return M_number;
}
```

```
public void setM_number(String M_number)
{
        this.M_number=M_number;
}
public String GetM_message(){
        return M_message;
}
public void setM_message(String M_message)
{
        this.M_message=M_message;
}
}
```

1.8 passenger 类的实现

```
package newpackage;
public class passenger{
private String P_phone;
private String P_address;
private String P_id;
private String P_password;
private String P_email;
private String P_name;
private String P_age;
private String P_sex;
public passenger()
{

}
public passenger(String P_name)
{
      P_name=P_name;
}
public passenger(String P_name,String P_password) {
      super();
      P_name=P_name;
      P_password=P_password;
 }

 public String GetP_name(){
       return P_name;
 }
 public void setP_name(String P_name)
 {
       this.P_name=P_name;
```

```
}
public String GetP_password(){
	return P_password;
}
public void setP_password(String P_password)
{
this.P_password=P_password;
}
public String GetP_phone(){
	return P_phone;
}
public void setP_phone(String P_phone)
{
this.P_phone=P_phone;
}
public String GetP_address(){
	return P_address;
}
public void setP_address(String P_address)
{
	this.P_address=P_address;
}
public String GetP_id(){
	return P_id;
}
public void setP_id(String P_id)
{
	this.P_id=P_id;
}
public String GetP_email(){
	return P_email;
}
public void setP_email(String P_email){
	this.P_email=P_email;
}
public String GetP_age(){
	return P_age;
}
public void setP_age(String P_age)
{
	this.P_age=P_age;
}
public String GetP_sex(){
	return P_sex;
```

```
}
public void setP_sex(String P_sex)
{
        this.P_sex=P_sex;
}

void P_age(String string) {
throw new UnsupportedOperationException("Not supported yet."); //To change body of
generated methods, choose Tools | Templates.
}
}
```

1.9 torder 类的实现

```
package newpackage;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Timestamp;
public class torder extends train {
private String T_name;
private String P_id;
private String T_seattype;
private Timestamp Nowtime;
private String T_date;
private String softype;
private String hardtype;

public torder(){
            super();
}

public String GetP_id(){
      return P_id;
}
public void setP_id(String P_id)
{
      this.P_id=P_id;
}
public String GetT_seattype(){
      return T_seattype;
}
public void setT_seattype(String T_seattype)
{
```

```
        this.T_seattype=T_seattype;
    }

    public String GetT_name(){
        return T_name;
    }
    public void setT_name(String T_name)
    {
        this.T_name=T_name;
    }
    public Timestamp GetNowtime(){
        return Nowtime;
    }
    public void setNowtime(Timestamp Nowtime)
    {
        this.Nowtime=Nowtime;
    }

    public String GetT_date(){
        return T_date;
    }
    public void setT_date(String T_date)
    {
        this.T_date=T_date;
    }
    public String Getsoftype(){
        return softype;
    }
    public void setsoftype(String softype)
    {
        this.softype=softype;
    }
      public String Gethardtype(){
        return hardtype;
    }
    public void sethardtype(String hardtype)
    {
        this.hardtype=hardtype;
    }

    public ResultSet PassengerList(Connection con, passenger passenger) throws SQLException {
    StringBuffer sb = new StringBuffer("select
    train.T_name,train.T_departure,train.T_destination,train.T_leavetime,train.T_arrivetime,tord
    er.T_seattype,train_price.T_price,torder.T_date,train_price.T_date from
```

```
 train_price,train,torder where train.T_name=torder.T_name and train.T_id=train_price.T_id
 and train_price.T_seattype=torder.T_seattype and train_price.T_date=torder.T_date");

if (StringUtil.isNotEmpty(passenger.GetP_id())) {
      sb.append(" and P_id = '" + passenger.GetP_id() + "'");
}
System.out.println(sb.toString());
PreparedStatement pstmt = con.prepareStatement(sb.toString());
return pstmt.executeQuery();
 }

 public int SelectionCancel(Connection con)throws Exception{
 String sql="delete from torder where T_name=? and T_date=?";
 PreparedStatement pstmt=con.prepareStatement(sql);

 pstmt.setString(1, T_name);
       pstmt.setString(2, T_date);
 return pstmt.executeUpdate();
 }
 public int NumSelectedMinus(Connection con)throws Exception{
 String sql="update torder set numSelected = numSelected - 1 where T_name=?";
 PreparedStatement    pstmt=con.prepareStatement(sql);
 pstmt.setString(1, T_name);
 return pstmt.executeUpdate();
 }

 public int T_numberMinus(Connection con,String T_name,String T_seattype,String
 T_date)throws Exception{
 String sql="update train_price set T_number = T_number - 1 where T_name=? and
 T_seattype=? and T_date=?";
 PreparedStatement    pstmt=con.prepareStatement(sql);
 pstmt.setString(1, T_name);
 pstmt.setString(2, T_seattype);
 pstmt.setString(3, T_date);
 return pstmt.executeUpdate();
 }

 public    ResultSet isExisted(Connection con) throws SQLException{

 String sql = "select * from torder where P_id=? and T_name=? ";
 PreparedStatement pstmt = con.prepareStatement(sql);
 pstmt.setString(1, P_id);
 pstmt.setString(2, T_name);
 return pstmt.executeQuery();
```

```
}
public int SelectionAdd(Connection con)throws Exception{
String sql="insert into torder (T_name,P_id,T_seattype,Nowtime,T_date,softype,hardtype)
values(?,?,?,?,?,?,?)";
PreparedStatement pstmt=con.prepareStatement(sql);
pstmt.setString(1,T_name);
pstmt.setString(2,P_id);
pstmt.setString(3,T_seattype);
pstmt.setTimestamp(4,Nowtime);
pstmt.setString(5,T_date);
pstmt.setString(6,softype);
pstmt.setString(7,hardtype);
return pstmt.executeUpdate();
}

public int T_numberAdd(Connection con,String T_name,String T_seattype,String
T_date)throws Exception{
String sql="update train_price set T_number = T_number    + 1 where T_name=? and
T_seattype=? and T_date=?";
PreparedStatement    pstmt=con.prepareStatement(sql);
pstmt.setString(1, T_name);
pstmt.setString(2, T_seattype);
pstmt.setString(3, T_date);
return pstmt.executeUpdate();
}
}
```

1.10 train 类的实现

```
package newpackage;
import java.sql.Time;
public class train extends train_price{
static void update(String sql, String time) {
throw new UnsupportedOperationException("Not supported yet."); //To change body of
generated methods, choose Tools | Templates.
}
private String T_id;
private String T_name;
private Time T_leavetime;
private Time T_arrivetime;
private String T_departure;
private String T_destination;
private String T_date;

public train()
```

```
{
    super();
}
public train(String T_id)
{
      super();
      T_id=T_id;
}
public train(String T_id,String T_name) {
      super();
      T_id=T_id;
      T_name=T_name;
}

public String GetT_id(){
      return T_id;
}
public void setT_id(String T_id)
{
      this.T_id=T_id;
}

public String GetT_name(){
      return T_name;
}
public void setT_name(String T_name)
{
      this.T_name=T_name;
}
public Time GetT_leavetime(){
      return T_leavetime;
}
public void setT_leavetime(Time T_leavetime)
{
      this.T_leavetime=T_leavetime;
}
public Time GetT_arrivetime(){
      return T_arrivetime;
}
public void setT_arrivetime(Time T_arrivetime)
{
      this.T_arrivetime=T_arrivetime;
}
public String GetT_departure(){
```

```
        return T_departure;
    }
    public void setT_departure(String T_departure)
    {
        this.T_departure=T_departure;
    }
    public String GetT_destination(){
        return T_destination;
    }
    public void setT_destination(String T_destination)
    {
        this.T_destination=T_destination;
    }
    public String GetT_date(){
        return T_date;
    }
    public void setT_date(String T_date)
    {
        this.T_date=T_date;
    }
    }
```

1.11 train_price 类的实现

```
    package newpackage;
    public class train_price {
    private String T_id;
    private String T_name;
    private String T_seattype;
    private String T_price;
    private String T_number;
    private String T_date;
    public train_price()
    {

    }
    public train_price(String T_id,String T_name,String T_seattype)
    {     T_id=T_id;
          T_name=T_name;
          T_seattype=T_seattype;
    }

    public String GetT_name()
    {
        return T_name;
```

```
}
public void setT_name(String T_name)
{
	this.T_name=T_name;
}
public String GetT_seattype()
{
	return T_seattype;
}
public void setT_seattype(String T_seattype)
{
	this.T_seattype=T_seattype;
}
public String GetT_id()
{
	return T_id;
}
public void setT_id(String T_id)
{
	this.T_id=T_id;
}
public String GetT_price()
{
	return T_price;
}
public void setT_price(String T_price)
{
	this.T_price=T_price;
}
  public String GetT_number()
{
	return T_number;
}
public void setT_number(String T_number)
{
	this.T_number=T_number;
}
public String GetT_date(){
	return T_date;
}
public void setT_date(String T_date)
{
	this.T_date=T_date;
}
```

```
    }
```

2. control 类的实现

```
    public int passengerAdd(Connection con,passenger passenger)throws Exception {
    String sql="insert into sinfo values(?,?,?,?,?,?)";
        PreparedStatement    pstmt=con.prepareStatement(sql);
        pstmt.setString(1,passenger.GetP_name());
        pstmt.setString(2,passenger.GetP_phone());
        pstmt.setString(3,passenger.GetP_address());
                pstmt.setString(4,passenger.GetP_id());
                pstmt.setString(5,passenger.GetP_password());
                pstmt.setString(6,passenger.GetP_email());
        return pstmt.executeUpdate();
}
    public int psssengerModify(Connection con, passenger passenger)throws Exception{
        String sql="update sinfo set P_name = ? ,P_phone = ?, P_address = ?, P_password
= ?,P_email = ? where P_id = ?";
        PreparedStatement    pstmt=con.prepareStatement(sql);
        pstmt.setString(1,passenger.GetP_name());
        pstmt.setString(2,passenger.GetP_phone());
        pstmt.setString(3,passenger.GetP_address());
                pstmt.setString(4,passenger.GetP_id());
                pstmt.setString(5,passenger.GetP_password());
                pstmt.setString(6,passenger.GetP_email());
        return pstmt.executeUpdate();
    }
    public int passengerDelete(Connection con, String P_id) throws SQLException {
       //To change body of generated methods, choose Tools | Templates.
        String sql = "delete from passenger where P_id=?";
        PreparedStatement pstmt = con.prepareStatement(sql);
        pstmt.setString(1, P_id);
        return pstmt.executeUpdate();
    }
    public int trainAdd(Connection con,train train)throws Exception {
    String sql="insert into train values(?,?,?,?,?,?,?)";
        PreparedStatement    pstmt=con.prepareStatement(sql);
        pstmt.setString(1,train.GetT_id());
        pstmt.setString(2,train.GetT_name());
        pstmt.setTime(3,train.GetT_leavetime());
                pstmt.setTime(4,train.GetT_arrivetime());
                pstmt.setString(5,train.GetT_departure());
                pstmt.setString(6,train.GetT_destination());
                pstmt.setString(7,train.GetT_date());
```

```
            return pstmt.executeUpdate();
}
      public int trainModify(Connection con, train train)throws Exception{
            String sql="update train set T_name = ? ,T_leavetime = ?, T_arrivetime = ?, T_departure
= ?,T_destination = ?，T_seattype = ?,T_price = ?,T_number = ? where T_id = ?";
            PreparedStatement    pstmt=con.prepareStatement(sql);
            pstmt.setString(2,train.GetT_name());
            pstmt.setTime(3,train.GetT_leavetime());
                        pstmt.setTime(4,train.GetT_arrivetime());
                        pstmt.setString(5,train.GetT_departure());
                        pstmt.setString(6,train.GetT_destination());
                        pstmt.setString(7,train.GetT_seattype());
                        pstmt.setString(8,train.GetT_price());
                        pstmt.setString(9,train.GetT_number());
            return pstmt.executeUpdate();
      }
      public int trainDelete(Connection con,String T_id, String T_date,String T_name) throws
SQLException {
            //To change body of generated methods, choose Tools | Templates.
            String sql = "delete from train where T_id=? and T_date=? and T_name=?";
            PreparedStatement pstmt = con.prepareStatement(sql);
            pstmt.setString(1, T_id);
            pstmt.setString(2, T_date);
            pstmt.setString(3, T_name);
            return pstmt.executeUpdate();
      }
     public int hsrailwayAdd(Connection con,hsrailway hsrailway)throws Exception {
      String sql="insert into hsrailway values(?,?,?,?,?,?,?)";
            PreparedStatement    pstmt=con.prepareStatement(sql);
            pstmt.setString(1,hsrailway.GetH_id());
            pstmt.setString(2,hsrailway.GetH_name());
            pstmt.setTime(3,hsrailway.GetH_leavetime());
                        pstmt.setTime(4,hsrailway.GetH_arrivetime());
                        pstmt.setString(5,hsrailway.GetH_departure());
                        pstmt.setString(6,hsrailway.GetH_destination());
                        pstmt.setString(7,hsrailway.GetH_date());
                       return pstmt.executeUpdate();

}
      public int hsrailwayModify(Connection con, hsrailway hsrailway)throws Exception{
            String sql="update train set H_name = ? ,H_leavetime = ?, H_arrivetime = ?,
H_departure = ?,H_destination = ?,H_seattype = ?,H_price = ?,H_number = ? where H_id = ?";
            PreparedStatement    pstmt=con.prepareStatement(sql);
            pstmt.setString(2,hsrailway.GetH_name());
```

```
            pstmt.setTime(3,hsrailway.GetH_leavetime());
                        pstmt.setTime(4,hsrailway.GetH_arrivetime());
                        pstmt.setString(5,hsrailway.GetH_departure());
                        pstmt.setString(6,hsrailway.GetH_destination());
                        pstmt.setString(7,hsrailway.GetH_seattype());
                        pstmt.setString(8,hsrailway.GetH_price());
                        pstmt.setString(9,hsrailway.GetH_number());
            return pstmt.executeUpdate();
      }
        public int hsrailwayDelete(Connection con, String H_id,String H_date,String H_name)
throws SQLException {
            //To change body of generated methods, choose Tools | Templates.
            String sql = "delete from hsrailway where H_id=? and H_date=? and H_name=?";
            PreparedStatement pstmt = con.prepareStatement(sql);
            pstmt.setString(1, H_id);
            pstmt.setString(2, H_date);
            pstmt.setString(3, H_name);
            return pstmt.executeUpdate();
      }

      public int messageAdd(Connection con,message message)throws Exception {
      String sql="insert into message values(?)";
            PreparedStatement pstmt=con.prepareStatement(sql);
                        pstmt.setString(1,message.GetM_message());
                        return pstmt.executeUpdate();
      }

        public int messageDelete(Connection con, String M_message) throws SQLException {
            //To change body of generated methods, choose Tools | Templates.
            String sql = "delete from message where M_message=?";
            PreparedStatement pstmt = con.prepareStatement(sql);
            pstmt.setString(1, M_message);
            return pstmt.executeUpdate();
      }
        public passenger Verifypassenger(Connection con,passenger passenger) throws Exception{
            passenger resultpas=null;
      String sql="select * from passenger where P_id=? and P_password=?";
PreparedStatement pstmt=con.prepareStatement(sql);
            pstmt.setString(1,passenger.GetP_id());
            pstmt.setString(2, passenger.GetP_password());
            ResultSet rs=pstmt.executeQuery();
            if(rs.next()){

                  resultpas=new passenger();
```

```
                resultpas.setP_id(rs.getString("P_id"));
                resultpas.setP_password(rs.getString("P_password"));
            }

           return resultpas;
            }

public admin_ Verifyadmin_(Connection con, admin_ admin_) throws Exception{
            admin_ resultad=null;
      String sql="select * from admin_ where A_id=? and A_password=?";
            PreparedStatement pstmt=con.prepareStatement(sql);
            pstmt.setString(1,admin_.GetA_id());
            pstmt.setString(2, admin_.GetA_password());
            ResultSet rs=pstmt.executeQuery();
            if(rs.next()){
                resultad=new admin_();
                resultad.setA_id(rs.getString("A_id"));
                resultad.setA_password(rs.getString("A_password"));
            }
           return resultad;
            }

public String QueryPassword(Connection con,passenger passenger) throws SQLException
      {
            String sql ="select P_password from passenger where P_id = ? ";
            System.out.println(passenger.GetP_id());
            PreparedStatement pstmt=con.prepareStatement(sql);

            pstmt.setString(1,passenger.GetP_id());
            ResultSet rs=pstmt.executeQuery();
            if (rs.next()){
                  return rs.getString("P_password");
            }
                  else
                  {
                  return "";
                  }
      }
      public int ModifyPassword(Connection con,passenger passenger)throws Exception{
            String sql="update passenger set P_password = ? where P_id = ?";
            PreparedStatement    pstmt=con.prepareStatement(sql);

            pstmt.setString(1,passenger.GetP_password());
            pstmt.setString(2,passenger.GetP_id());
```

```
            return pstmt.executeUpdate();

        }

public String QueryPassword_(Connection con,admin_ admin_) throws SQLException
        {
            String sql ="select A_password from admin_ where A_id = ? ";
            System.out.println(admin_.GetA_id());
            PreparedStatement pstmt=con.prepareStatement(sql);

            pstmt.setString(1,admin_.GetA_id());
            ResultSet rs=pstmt.executeQuery();
            if (rs.next()){
                    return rs.getString("A_password");
            }
                    else
                   {
                   return "";
                   }
        }
        public int ModifyPassword_(Connection con,admin_ admin_)throws Exception{
            String sql="update admin_ set A_password = ?    where A_id = ?";
            PreparedStatement    pstmt=con.prepareStatement(sql);

            pstmt.setString(1,admin_.GetA_password());
            pstmt.setString(2,admin_.GetA_password());

            return pstmt.executeUpdate();

        }
        public Boolean CanpassengerAdd(Connection con,String tablename,String
columnname,String P_id) throws SQLException
        {

        String sql="select * from " + tablename + " where " + columnname    + " = ? ";
            PreparedStatement pstmt=con.prepareStatement(sql);
            pstmt.setString(1,P_id);
            ResultSet rs=pstmt.executeQuery();
            if (rs.next()){
                    return false;
            }
                    else
                    {
```

```
                return true;
                }
        }

    public Boolean CanpassengerModify(Connection con,String tablename,String
columnname,String P_id) throws SQLException
    {
        String sql="select * from " + tablename + " where " + columnname    + " = ? ";

        PreparedStatement pstmt=con.prepareStatement(sql);
        pstmt.setString(1,P_id);
        ResultSet rs=pstmt.executeQuery();
        if (rs.next()){
             return true;
        }
             else
             {
            return false;
            }
    }

    public Boolean CantrainAdd(Connection con,String tablename,String columnname,String
T_id) throws SQLException
    {

    String sql="select * from " + tablename + " where " + columnname    + " = ? ";
        PreparedStatement pstmt=con.prepareStatement(sql);
        pstmt.setString(1,T_id);
        ResultSet rs=pstmt.executeQuery();
        if (rs.next()){
             return false;
        }
             else
             {
            return true;
            }
        }

    public Boolean CantrainModify(Connection con,String tablename,String columnname,String
T_id) throws SQLException
    {
        String sql="select * from " + tablename + " where " + columnname    + " = ? ";

        PreparedStatement pstmt=con.prepareStatement(sql);
```

```
        pstmt.setString(1,T_id);
        ResultSet rs=pstmt.executeQuery();
        if (rs.next()){
                return true;
        }
                else
                {
               return false;
               }
    }

  public Boolean CanhsrailwayAdd(Connection con,String tablename,String columnname,String
H_id) throws SQLException
    {

    String sql="select * from " + tablename + " where " + columnname    + " = ? ";
        PreparedStatement pstmt=con.prepareStatement(sql);
        pstmt.setString(1,H_id);
        ResultSet rs=pstmt.executeQuery();
        if (rs.next()){
                return false;
        }
                else
                {
               return true;
               }
        }

    public Boolean CanhsrailwayModify(Connection con,String tablename,String
columnname,String H_id) throws SQLException
    {
        String sql="select * from " + tablename + " where " + columnname    + " = ? ";

        PreparedStatement pstmt=con.prepareStatement(sql);
        pstmt.setString(1,H_id);
        ResultSet rs=pstmt.executeQuery();
        if (rs.next()){
                return true;
        }
                else
                {
               return false;
               }
    }
```

```
    public Boolean CanmessageAdd(Connection con,String tablename,String
columnname,String M_number) throws SQLException
    {

    String sql="select * from " + tablename + " where " + columnname    + " = ? ";
        PreparedStatement pstmt=con.prepareStatement(sql);
        pstmt.setString(1,M_number);
        ResultSet rs=pstmt.executeQuery();
        if (rs.next()){
             return false;
        }
             else
             {
            return true;
            }
        }

    public Boolean CanmessageModify(Connection con,String tablename,String
columnname,String M_number) throws SQLException
    {
        String sql="select * from " + tablename + " where " + columnname    + " = ? ";

        PreparedStatement pstmt=con.prepareStatement(sql);
        pstmt.setString(1,M_number);
        ResultSet rs=pstmt.executeQuery();
        if (rs.next()){
             return true;
        }
             else
             {
            return false;
            }
    }
    public ResultSet QueryList(Connection con,String tablename,String columnname,String pno)
throws SQLException
    {
        String sql="";

    if (pno.isEmpty()){
        sql="select * from " + tablename;

        PreparedStatement pstmt=con.prepareStatement(sql);
```

```
            return pstmt.executeQuery();
        }
            else
        {
            sql="select * from " + tablename + " where " + columnname    + " = ? ";

            PreparedStatement pstmt=con.prepareStatement(sql);
            pstmt.setString(1,pno);
            return pstmt.executeQuery();
        }

        }
        public    ResultSet Querytrain(Connection con,String T_departure,String T_destination,String
T_date) throws SQLException
        {


            String sql = "select
train.T_name,T_leavetime,T_arrivetime,T_departure,T_destination,train.T_date,train_price.T_seat
type,T_price,T_number    from train,train_price where train.T_id=train_price.T_id and
train.T_departure=? and train.T_destination=? and train.T_date=?";
            PreparedStatement pstmt=con.prepareStatement(sql);
            pstmt.setString(1,T_departure);
            pstmt.setString(2,T_destination);
            pstmt.setString(3,T_date);


            System.out.println(pstmt.toString());
            return pstmt.executeQuery();
        }

public    ResultSet Queryhsrailway(Connection con,String H_departure,String
H_destination,String H_date) throws SQLException
        {

            String sql="";

            sql = "select
hsrailway.H_name,H_leavetime,H_arrivetime,H_departure,H_destination,hsrailway.H_date,hsrail
way_price.H_seattype,H_price,H_number    from hsrailway,hsrailway_price where
hsrailway.H_id=hsrailway_price.H_id and hsrailway.H_departure=? and hsrailway.H_destination=?
and hsrailway.H_date=?" ;
            PreparedStatement pstmt=con.prepareStatement(sql);
            pstmt.setString(1,H_departure);
```

```
            pstmt.setString(2,H_destination);
            pstmt.setString(3,H_date);
            System.out.println(pstmt.toString());
            return pstmt.executeQuery();
      }
public    ResultSet Queryallhsrailway(Connection con) throws SQLException
      {

            String sql="";

            sql = "select
hsrailway.H_name,H_leavetime,H_arrivetime,H_departure,H_destination,hsrailway.H_date,hsrail
way_price.H_seattype,H_price,H_number    from hsrailway,hsrailway_price where
hsrailway.H_id=hsrailway_price.H_id";
            PreparedStatement pstmt=con.prepareStatement(sql);
            System.out.println(pstmt.toString());
            return pstmt.executeQuery();
      }

public    ResultSet Queryalltrain(Connection con) throws SQLException
      {

            String sql="";

            sql = "select
train.T_name,T_leavetime,T_arrivetime,T_departure,T_destination,train.T_date,train_price.T_seat
type,T_price,T_number    from train,train_price where train.T_id=train_price.T_id" ;
            PreparedStatement pstmt=con.prepareStatement(sql);
            System.out.println(pstmt.toString());
            return pstmt.executeQuery();
      }

public int noticeAdd(Connection con,notice notice)throws Exception {
      String sql="insert into notice values (?)";
            PreparedStatement    pstmt=con.prepareStatement(sql);
            pstmt.setString(1,notice.Getcontent());
                  return pstmt.executeUpdate();
      }


      boolean CannoticeAdd(Connection con, String notice, String content) {
            throw new UnsupportedOperationException("Not supported yet."); //To change body
of generated methods, choose Tools | Templates.
      }
```

```
    public    ResultSet Querytrainorder(Connection con,passenger passenger) throws
SQLException
    {
        StringBuffer sb = new StringBuffer("select
train.T_name,torder.T_date,train.T_departure,train.T_destination,train.T_leavetime,train.T_arrivet
ime,torder.T_seattype,torder.hardtype,torder.softype,train_price.T_price,torder.Nowtime from
train_price,train,torder where train.T_name=torder.T_name and train.T_id=train_price.T_id and
train_price.T_seattype=torder.T_seattype and train_price.T_date=torder.T_date");

        if (StringUtil.isNotEmpty(passenger.GetP_id())) {
            sb.append(" and P_id = '" + passenger.GetP_id() + "'");
        }
        System.out.println(sb.toString());
        PreparedStatement pstmt = con.prepareStatement(sb.toString());
        return pstmt.executeQuery();
    }
    public    ResultSet Querytrainorder1(Connection con,passenger passenger) throws
SQLException
    {
        StringBuffer sb = new StringBuffer("select
train.T_name,torder.T_date,train.T_departure,train.T_destination,train.T_leavetime,train.T_arrivet
ime,torder.T_seattype,train_price.T_price,torder.Nowtime from train_price,train,torder where
train.T_name=torder.T_name and train.T_id=train_price.T_id and
train_price.T_seattype=torder.T_seattype and train_price.T_date=torder.T_date");


        if (StringUtil.isNotEmpty(passenger.GetP_id())) {
            sb.append(" and P_id = '" + passenger.GetP_id() + "'");
        }
        System.out.println(sb.toString());
        PreparedStatement pstmt = con.prepareStatement(sb.toString());
        return pstmt.executeQuery();
    }
    public    ResultSet Queryhsrailwayorder(Connection con,passenger passenger) throws
SQLException
    {
        StringBuffer sb = new StringBuffer("select
hsrailway.H_name,hsorder.H_date,hsrailway.H_departure,hsrailway.H_destination,hsrailway.H_le
avetime,hsrailway.H_arrivetime,hsorder.H_seattype,hsrailway_price.H_price,hsorder.Nowtime
from hsrailway_price,hsrailway,hsorder where hsrailway.H_name=hsorder.H_name and
hsrailway.H_id=hsrailway_price.H_id and hsrailway_price.H_seattype=hsorder.H_seattype and
hsrailway_price.H_date=hsorder.H_date");
```

```
            if (StringUtil.isNotEmpty(passenger.GetP_id())) {
                sb.append(" and P_id = '" + passenger.GetP_id() + "'");
            }
            System.out.println(sb.toString());
            PreparedStatement pstmt = con.prepareStatement(sb.toString());
            return pstmt.executeQuery();
    }
     public    ResultSet Querynotice(Connection con) throws SQLException
    {

            String sql="";

            sql = "select* from notice";
            PreparedStatement pstmt=con.prepareStatement(sql);
            System.out.println(pstmt.toString());
            return pstmt.executeQuery();
    }
       public    ResultSet Queryallpassenger(Connection con) throws SQLException
    {

            String sql="";

            sql = "select P_name,P_phone,P_address,P_id,P_password,P_email,\n" +
"timestampdiff(year,str_to_date(substring(P_id,7,8),\"%Y%m%d\"),str_to_date(now(),\"%Y-%m-
%d\"))as P_age,\n" +
"case if(length(P_id)=18, cast(substring(P_id,17,1) as UNSIGNED)%2, \n" +
"if(length(P_id)=15,cast(substring(P_id,15,1) as UNSIGNED)%2,3))\n" +
"when 1 then '男' \n" +
"when 0 then '女'\n" +
"else '未知'\n" +
"end as P_sex\n" +
"from passenger";
            PreparedStatement pstmt=con.prepareStatement(sql);
            System.out.println(pstmt.toString());
            return pstmt.executeQuery();
    }
       public    ResultSet Querypassenger(Connection con,String P_id) throws SQLException
    {

            String sql="";

            sql = "select P_name,P_phone,P_address,P_id,P_password,P_email,\n" +
"timestampdiff(year,str_to_date(substring(P_id,7,8),\"%Y%m%d\"),str_to_date(now(),\"%Y-%m-
%d\"))as P_age,\n" +
```

```
"case if(length(P_id)=18, cast(substring(P_id,17,1) as UNSIGNED)%2, \n" +
"if(length(P_id)=15,cast(substring(P_id,15,1) as UNSIGNED)%2,3))\n" +
"when 1 then '男' \n" +
"when 0 then '女'\n" +
"else '未知'\n" +
"end as P_sex\n" +
"from passenger\n" +
"where P_id =?";
            PreparedStatement pstmt=con.prepareStatement(sql);
            pstmt.setString(1,P_id);
            System.out.println(pstmt.toString());
            return pstmt.executeQuery();
      }
       public String QueryoldPassword(Connection con,passenger passenger) throws
SQLException
      {
            String sql ="select P_password from passenger where P_id = ? ";
            System.out.println(passenger.GetP_id());
            PreparedStatement pstmt=con.prepareStatement(sql);
            pstmt.setString(1,passenger.GetP_id());
            ResultSet rs=pstmt.executeQuery();
            if (rs.next()){
                  return rs.getString("P_password");
            }
                  else
                  {
                  return "";
                  }
      }

      public int train_priceAdd(Connection con,train_price train_price)throws Exception {
      String sql="insert into train_price values(?,?,?,?,?,?)";
            PreparedStatement    pstmt=con.prepareStatement(sql);
            pstmt.setString(1,train_price.GetT_id());
            pstmt.setString(2,train_price.GetT_name());
            pstmt.setString(3,train_price.GetT_seattype());
                        pstmt.setString(4,train_price.GetT_price());
                        pstmt.setString(5,train_price.GetT_number());
                        pstmt.setString(6,train_price.GetT_date());
                        return pstmt.executeUpdate();


}
```

```
        public int hsrailway_priceAdd(Connection con,hsrailway_price hsrailway_price)throws
Exception {
      String sql="insert into hsrailway_price values(?,?,?,?,?,?)";
            PreparedStatement    pstmt=con.prepareStatement(sql);
            pstmt.setString(1,hsrailway_price.GetH_id());
            pstmt.setString(2,hsrailway_price.GetH_name());
              pstmt.setString(3,hsrailway_price.GetH_seattype());
                        pstmt.setString(4,hsrailway_price.GetH_price());
                        pstmt.setString(5,hsrailway_price.GetH_number());
                        pstmt.setString(6,hsrailway_price.GetH_date());
            return pstmt.executeUpdate();

}

      public Boolean Cantrain_priceAdd(Connection con,String tablename,String
columnname,String T_id) throws SQLException
      {

      String sql="select * from " + tablename + " where " + columnname    + " = ? ";
            PreparedStatement pstmt=con.prepareStatement(sql);
            pstmt.setString(1,T_id);
            ResultSet rs=pstmt.executeQuery();
            if (rs.next()){
                  return false;
            }
                  else
                  {
                 return true;
                 }
            }

  public Boolean Canhsrailway_priceAdd(Connection con,String tablename,String
columnname,String H_id) throws SQLException
      {

      String sql="select * from " + tablename + " where " + columnname    + " = ? ";
            PreparedStatement pstmt=con.prepareStatement(sql);
            pstmt.setString(1,H_id);
            ResultSet rs=pstmt.executeQuery();
            if (rs.next()){
                  return false;
            }
                  else
                  {
```

```
                return true;
                }
        }

    public Boolean CannoticeAdd(Connection con,String tablename,String columnname,String
content) throws SQLException
    {

    String sql="select * from " + tablename + " where " + columnname    + " = ? ";
        PreparedStatement pstmt=con.prepareStatement(sql);
        pstmt.setString(1,content);
        ResultSet rs=pstmt.executeQuery();
        if (rs.next()){
            return false;
        }
            else
            {
            return true;
            }
        }


    public ResultSet passengerList(Connection con, passenger passenger) throws SQLException {
        /**StringBuffer sb = new StringBuffer("select * from slogin ");
        if (student.getSno() != "") {
            sb.append("where Sno='" + student.getSno() + "'");
        }
        PreparedStatement pstmt = con.prepareStatement(sb.toString());*/
        StringBuilder sb = new StringBuilder("select * from passenger");
        /**if (student.getSno() != "") {
            sb.append(" and Sno=" + student.getSno());
        }*/
        if (StringUtil.isNotEmpty(passenger.GetP_id())) {
            sb.append(" and jTextField1 ='").append(passenger.GetP_id()).append("'");
        }
        PreparedStatement pstmt = con.prepareStatement(sb.toString().replaceFirst("and",
"where"));
        return pstmt.executeQuery();

    }


    public int ModifyP_password(Connection con,passenger passenger)throws Exception{
        String sql="update passenger set spassword = ?    where P_id = ?";
```

```
        PreparedStatement    pstmt=con.prepareStatement(sql);

        pstmt.setString(1,passenger.GetP_password());
        pstmt.setString(2,passenger.GetP_id());

        return pstmt.executeUpdate();

    }



    public ResultSet messageList(Connection con, message message) throws SQLException {
        /**StringBuffer sb = new StringBuffer("select * from slogin ");
        if (student.getSno() != "") {
            sb.append("where Sno='" + student.getSno() + "'");
        }
        PreparedStatement pstmt = con.prepareStatement(sb.toString());*/
        StringBuilder sb = new StringBuilder("select * from message");
        /**if (student.getSno() != "") {
            sb.append(" and Sno=" + student.getSno());
        }*/
        if (StringUtil.isNotEmpty(message.GetM_message())) {
            sb.append(" and message ='").append(message.GetM_message()).append("'");
        }
        PreparedStatement pstmt = con.prepareStatement(sb.toString().replaceFirst("and",
"where"));
        return pstmt.executeQuery();

    }
public int DeleteM_message(Connection con, String M_message) throws SQLException
    {
        String sql = "delete from message where M_message=?";
        PreparedStatement pstmt = con.prepareStatement(sql);
        pstmt.setString(1,M_message);
        return pstmt.executeUpdate();
    }


    public Boolean Candelete(Connection con,String tablename,String columnname,String
M_message) throws SQLException
    {

    String sql="select * from " + tablename + " where " + columnname    + " = ? ";
        PreparedStatement pstmt=con.prepareStatement(sql);
```

```
            pstmt.setString(1,M_message);
            ResultSet rs=pstmt.executeQuery();
            if (rs.next()){
                    return false;
            }
                    else
                    {
                   return true;
                   }
            }


      public int ConfirmDelete(Component com,String str,String Sno)
      {
          int n= JOptionPane.showConfirmDialog(com, "确定要删除" + str + Sno +"吗？ ");
          return n;
      }
      public Boolean CanModify(Connection con,String tablename,String columnname,String P_id)
throws SQLException
      {
            String sql="select * from " + tablename + " where " + columnname    + " = ? ";

            PreparedStatement pstmt=con.prepareStatement(sql);
            pstmt.setString(1,P_id);
            ResultSet rs=pstmt.executeQuery();
            if (rs.next()){
                    return true;
            }
                    else
                    {
                   return false;
                   }
      }

     public Boolean Modifyseattype1(Connection con,String P_id) throws SQLException
      {
            String sql="select* from torder where T_seattype = '硬卧'and P_id=?";

            PreparedStatement pstmt=con.prepareStatement(sql);
            pstmt.setString(1,P_id);
            ResultSet rs=pstmt.executeQuery();
            if (rs.next()){
                    return true;
            }
```

```
                else
                {
               return false;
               }
    }
    public Boolean Modifyseattype2(Connection con,String P_id) throws SQLException
    {
        String sql="select* from torder where T_seattype = '软卧'and P_id=?";

        PreparedStatement pstmt=con.prepareStatement(sql);
        pstmt.setString(1,P_id);
        ResultSet rs=pstmt.executeQuery();
        if (rs.next()){
            return true;
        }
            else
            {
           return false;
           }
    }
    public int train_priceDelete(Connection con,String T_id, String T_date,String T_name)
throws SQLException {
        //To change body of generated methods, choose Tools | Templates.
        String sql = "delete from train_price where T_id=? and T_date=? and T_name=?";
        PreparedStatement pstmt = con.prepareStatement(sql);
        pstmt.setString(1, T_id);
        pstmt.setString(2, T_date);
        pstmt.setString(3, T_name);
        return pstmt.executeUpdate();
    }


  public int hsrailway_priceDelete(Connection con, String H_id,String H_date,String H_name)
throws SQLException {
        //To change body of generated methods, choose Tools | Templates.
        String sql = "delete from hsrailway_price where H_id=? and H_date=? and H_name=?";
        PreparedStatement pstmt = con.prepareStatement(sql);
        pstmt.setString(1, H_id);
        pstmt.setString(2, H_date);
        pstmt.setString(3, H_name);
        return pstmt.executeUpdate();
    }
}
```

3. 表单对象类的实现

```
import java.awt.Component;
import javax.swing.JOptionPane;
public class FormObject {
public control thecontrol;
public train thetrain[];
public hsrailway thehsrailway[];

public FormObject()
{

}

public void ShowInfo(Component com,String strinfo)
{
    JOptionPane.showMessageDialog(com, strinfo);
}
public void ShowSuccessInfo(Component com,String strinfo)
{
    JOptionPane.showMessageDialog(com, strinfo);
}
}
```

4. 判断类的实现

```
public class StringUtil {
public static boolean isEmpty(String str){
if("".equals(str)||str==null){
    return true;
}else{
    return false;
}
}
public static boolean isNotEmpty(String str){
if(!"".equals(str)&&str!=null){
    return true;
}else{
    return false;
    }
}
}
```

5. 边界类的实现

5.1 登录界面

该界面主要有两个功能：管理员登录，乘客登录。管理员通过输入账号（A_id）和密码（A_password）与 Mysql 表中的 admin_进行比对，如果信息正确则进入管理员中心界面，如果所填内容为空或者错误则出现提示,“用户名不能为空”,“密码不能为空”,“登录失败”，无法登录。乘客通过输入账号（P_id）和密码（P_password）与 Mysql 表中的 passenger 进行比对，如果信息正确则进入主页和查询界面，如果所填内容为空或者错误则出现提示“用户名不能为空”，“密码不能为空”，“登录失败”。

该界面有三个按钮：登录，注册，重置

登录按钮下的程序如下所示：

```
private void jb_logOnActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String userName = userNameTxt.getText();
    String password = new String(passwordTxt.getPassword());
    if (StringUtil.isEmpty(userName)) {
        fo.ShowInfo(this, "用户名不能为空!");
        return;
    }
    if (StringUtil.isEmpty(password)) {
        fo.ShowInfo(this, "密码不能为空!");
        return;
    }
    Connection con=null;
    if (this.jrb_passenger.isSelected()) {
        passenger passenger = new passenger();
                    passenger.setP_id(userName);
                    passenger.setP_password(password);
```

批注 [b1]: 登录按钮程序

```
            try {
                con=database.getCon();

                currentpassenger = co.Verifypassenger(con,passenger);
                if (currentpassenger != null) {
                    this.dispose();
                    new 主页和查询 JFrame().setVisible(true);
                } else {
                    JOptionPane.showMessageDialog(this, "登录失败!");
                }
            } catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
                JOptionPane.showMessageDialog(this, "登录失败!");
            }finally{
                try {
                    database.closeCon(con);
                } catch (Exception e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            }
        } else if (this.jrb_admin_.isSelected()) {
            admin_ admin_ = new admin_();
                        admin_.setA_id(userName);
                        admin_.setA_password(password);
            try {
                con=database.getCon();
                currentadmin_ = co.Verifyadmin_(con, admin_);
                if (currentadmin_!= null) {
                    this.dispose();
                    new 管理员 JFrame().setVisible(true);
                } else {
                    JOptionPane.showMessageDialog(this, "登录失败!");
                }
            } catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
                JOptionPane.showMessageDialog(this, "登录失败!");
            }finally{
                try {
                    database.closeCon(con);
                } catch (Exception e) {
```

批注 [b2]: 见 control 类：识别乘客：public passenger Verifypassenger(Connection con,passenger passenger)

批注 [b3]: 见 control 类：识别管理员：public admin_ Verifyadmin_(Connection con, admin_ admin_)

```
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            }
        }
        P_id = this.userNameTxt.getText();
    }
```

点击注册按钮，进入注册界面。
点击重置按钮，清空所输入的账号和密码。

5.2 主页和查询界面

该界面（主页和查询 JFrame）主要实现了两个功能：乘客对火车的查询，乘客对高铁动车的查询。乘客通过输入出发日期（T_date/H_date），出发地点（T_departure/H_daparture），到达地点（T_destination/H_destination），点击查询按钮查找 Mysql 表中 train/hsrailway 中的信息，进入火车车次详情进行火车的查询。在基础上点击是否乘坐高铁动车，可以进入高铁动车车次详情，实现对高铁动车的查询。
该界面有两个按钮：个人中心，查询。
查询按钮下的程序如下：

```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
if (this.jCheckBox1.isSelected()) {
    try {

        H_date= this.jTextField1.getText();
        H_departure = this.jTextField6.getText();
        H_destination = this.jTextField7.getText();
       System.out.println(主页和查询 JFrame.H_date);
```

批注 [b4]: 查询按钮程序

```
                System.out.println(主页和查询 JFrame.H_departure);
                System.out.println(主页和查询 JFrame.H_destination);
                this.dispose();
                高铁动车车次详情 JFrame1 b = new  高铁动车车次详情 JFrame1();
                b.setVisible(true);
            } catch (Exception ex) {
                Logger.getLogger(主页和查询 JFrame.class.getName()).log(Level.SEVERE, null, ex);
            }
            }
        else{
            try {
               T_date = this.jTextField1.getText();
            T_departure = this.jTextField6.getText();
            T_destination = this.jTextField7.getText();
              System.out.println(主页和查询 JFrame.T_date);
            System.out.println(主页和查询 JFrame.T_departure);
            System.out.println(主页和查询 JFrame.T_destination);
           this.dispose();
            火车车次详情 JFrame1 a = new  火车车次详情 JFrame1();
            a.setVisible(true);
            } catch (Exception ex) {
                Logger.getLogger(主页和查询 JFrame.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
        }
```

点击个人中心按钮，进入乘客个人中心界面。

该界面中的公告，会显示管理员发布的公告信息。从 Mysql 的 notice 表中获得数据。

5.3 火车车次详情界面

该界面主要实现一个功能：火车票预订。乘客在表中点击想要预定的车次，若预订成功会出

现“预订成功”的提示。若预订火车重复、同时选择多于一个车次或火车票没有剩余则会出现相应的提示。
当乘客年龄>=60 岁，系统自动识别，跳出软卧硬卧的铺位选择。

该界面有四个按钮：预订，全部火车信息，个人中心，返回。
预订按钮下的程序如下：

批注 [b5]: 预订按钮程序

```
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

        // TODO add your handling code here:
        Timestamp Nowtime =new Timestamp(System.currentTimeMillis());
        if (T_name== "" && T_seattype=="") {
            JOptionPane.showMessageDialog(this, "请选择一个车次！ ");
            return;
        }

        if (T_number== 0) {
            JOptionPane.showMessageDialog(this, "该车次没有票了！ ");
```

```
            return;
        }

        int n = JOptionPane.showConfirmDialog(this, "确定要选择这个车次吗?");
        if (n == 0) {
            Connection con = null;
            String currentP_id =  登录.currentpassenger.GetP_id();
            torder sc=new torder();
            sc.setT_name(T_name);
            sc.setT_seattype(T_seattype);
            sc.setNowtime(Nowtime);
            sc.setT_date(T_date);
            sc.setP_id(currentP_id);

            try {
                con = database.getCon();
                ResultSet rs = sc.isExisted(con);
                while (rs.next()) {
                    JOptionPane.showMessageDialog(this, "你已经选过这个车次，不
能重复选择！");
                    return;
                }
                int selectionNum = sc.SelectionAdd(con);

                int selectedNum = sc.T_numberMinus(con,T_name,T_seattype,T_date);
                System.out.println(selectionNum);
                System.out.println(selectedNum);
                if (selectionNum == 1 && selectedNum >=    1) {
                    JOptionPane.showMessageDialog(this, "预订成功!");


                } else {
                    JOptionPane.showMessageDialog(this, "预订失败!");
                }
            } catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
                JOptionPane.showMessageDialog(this, "预订失败!");
            } finally {
                try {
                    database.closeCon(con);
                } catch (Exception e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
```

批注 [b6]: 见 torder 类：把预订的火车票填入订单：public int SelectionAdd(Connection con)

批注 [b7]: 见 torder 类：车票减少一张：public int T_numberMinus(Connection con,String T_name,String T_seattype,String T_date)

```
                    }
                }
            }
            Connection con = null;
            try {
                con=database.getCon();
            } catch (Exception ex) {
                Logger.getLogger(火车车次详情 JFrame1.class.getName()).log(Level.SEVERE,
null, ex);
            }
            Boolean rs = null;
            try {
                rs = Queryage(con,登录.P_id);
                System.out.println(rs);
            } catch (Exception ex) {
                Logger.getLogger(火车车次详情 JFrame1.class.getName()).log(Level.SEVERE,
null, ex);
            }
            if(rs != false){
                Boolean modifyseattype1 = null;
                Boolean modifyseattype2 = null;
                try {
                    modifyseattype1 = co.Modifyseattype1(con,登录.P_id);
                    modifyseattype2 = co.Modifyseattype2(con,登录.P_id);
                } catch (SQLException ex) {
                    Logger.getLogger(火车车次详情
JFrame1.class.getName()).log(Level.SEVERE, null, ex);
                }

        System.out.println(T_seattype);
        System.out.println("车次为："+getname);

        if(gettype.equals("硬卧")){
                if(modifyseattype1 == true){
                    老年人硬卧  a = new  老年人硬卧();
        a.setVisible(true);

                }
        }
        if(gettype.equals("软卧")){
                if(modifyseattype2 == true){
                    老年人软卧  b = new  老年人软卧();
        b.setVisible(true);
            }
```

批注 [b8]: 见此界面中识别年龄函数：public Boolean Queryage(Connection con,String P_id)

批注 [b9]: 见 control 类：识别硬卧 public Boolean Modifyseattype1(Connection con,String P_id)

批注 [b10]: 见 control 类：识别软卧 public Boolean Modifyseattype2(Connection con,String P_id)

```
                }
                }


    }
```

全部火车信息按钮下的程序如下：

```
    private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
    fillTable1(train);
    }
```

点击个人中心按钮，进入乘客个人中心界面。

点击返回按钮，返回主页和查询界面界面。

鼠标功能的实现：

```
        private void jTable1MousePressed(java.awt.event.MouseEvent evt) {
            // TODO add your handling code here:
            int row = jTable1.getSelectedRow();
        T_name = (String) this.jTable1.getValueAt(row, 0);
            T_seattype = (String) this.jTable1.getValueAt(row, 6);
            T_date = (String) jTable1.getValueAt(row, 5);
        T_number = Integer.parseInt((String) this.jTable1.getValueAt(row, 8));
        gettype=(String) this.jTable1.getValueAt(row, 6);
            System.out.println(gettype);
            getname=(String) this.jTable1.getValueAt(row, 0);

        }
```

filltable 功能的实现：

```
private void fillTable(train train, String T_departure, String T_destination,String T_date) {
        DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();
        dtm.setRowCount(0);
        Connection con = null;
        try {
            con = database.getCon();
          try {

        } catch (Exception ex){
            Logger.getLogger(主页和查询 JFrame.class.getName()).log(Level.SEVERE, null, ex);
        }
            ResultSet rs = co.Querytrain(con,T_departure,T_destination,T_date);
            while (rs.next()) {
                Vector v = new Vector();
                v.add(rs.getString(1));
                v.add(rs.getString(2));
                v.add(rs.getString(3));
                v.add(rs.getString(4));
                v.add(rs.getString(5));
```

批注 [b11]: 见下文 filltable 功能的实现

批注 [b12]: 鼠标功能

批注 [b13]: Filltable 功能

批注 [b14]: 见 control 类：根据出发地，到达地，出发日期进行火车信息查询：public ResultSet Querytrain(Connection con,String T_departure,String T_destination,String T_date)

```
                v.add(rs.getString(6));
                v.add(rs.getString(7));
                v.add(rs.getString(8));
                v.add(rs.getString(9));
                dtm.addRow(v);

            }
        }catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } finally {
            try {
                database.closeCon(con);
            } catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }

private void fillTable1(train train){
        DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();
        dtm.setRowCount(0);
        Connection con = null;
        try {
            con = database.getCon();
            ResultSet rs = co.Queryalltrain(con);
            while (rs.next()) {
                Vector v = new Vector();
                v.add(rs.getString(1));
                v.add(rs.getString(2));
                v.add(rs.getString(3));
                v.add(rs.getString(4));
                v.add(rs.getString(5));
                v.add(rs.getString(6));
                v.add(rs.getString(7));
                v.add(rs.getString(8));
                v.add(rs.getString(9));
                dtm.addRow(v);

            }
        }catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
```

批注 [b15]: Filltable 功能

批注 [b16]: 见 control 类：查询全部火车信息：public ResultSet Queryalltrain(Connection con)

```
            } finally {
                try {
                    database.closeCon(con);
                } catch (Exception e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            }
        }
```

识别年龄功能的实现：

```
public Boolean Queryage(Connection con,String P_id) throws Exception{
            String sql="";
        sql="select
timestampdiff(year,str_to_date(substring(P_id,7,8),\"%Y%m%d\"),str_to_date(now(),\"%Y-%m-%
d\"))>= 60 as P_age from passenger where
timestampdiff(year,str_to_date(substring(P_id,7,8),\"%Y%m%d\"),str_to_date(now(),\"%Y-%m-%
d\"))>= 60 and P_id=?";
            PreparedStatement pstmt=con.prepareStatement(sql);
            pstmt.setString(1,P_id);
            System.out.println(pstmt.toString());
            ResultSet rs=pstmt.executeQuery();
            if (rs.next()){
                    return true;
            }
                    else
                    {
                    return false;
                    }
    }
```

5.4 高铁动车车次详情界面

该界面主要实现一个功能：高铁动车车票预订。乘客在表中点击想要预定的车次，若预订成功会出现“预订成功”的提示。若预订高铁动车重复、同时选择多于一个车次或高铁动车票没有剩余则会出现相应的提示。

该界面有四个按钮：预订，全部高铁动车信息，个人中心，返回。

预订按钮下的程序如下：

```
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    Timestamp Nowtime =new Timestamp(System.currentTimeMillis());
     if (H_name== "" && H_seattype=="") {
        JOptionPane.showMessageDialog(this, "请选择一个车次！ ");
        return;
    }

    if (H_number== 0) {
        JOptionPane.showMessageDialog(this, "该车次没有票了！ ");
        return;
    }

    int n = JOptionPane.showConfirmDialog(this, "确定要选择这个车次吗?");
    if (n == 0) {
        Connection con = null;
        String currentP_id =  登录.currentpassenger.GetP_id();
                        hsorder sc=new hsorder();
                        sc.setH_name(H_name);
                        sc.setH_seattype(H_seattype);
                        sc.setNowtime(Nowtime);
                        sc.setH_date(H_date);
                        sc.setP_id(currentP_id);

        try {
            con = database.getCon();
```

批注 [b17]: 预订按钮程序

```
                                    ResultSet rs = sc.isExisted(con);
                                    while (rs.next()) {
                                    JOptionPane.showMessageDialog(this, "你已经选过这个
车次，不能重复选择！");
                                    return;
                                    }
                    int selectionNum = sc.SelectionAdd(con);

                    int selectedNum = sc.H_numberMinus(con,H_name,H_seattype,H_date);
                                    System.out.println(selectionNum);
                                      System.out.println(selectedNum);
                    if (selectionNum == 1 && selectedNum >=    1) {
                        JOptionPane.showMessageDialog(this, "预订成功!");


                    } else {
                        JOptionPane.showMessageDialog(this, "预订失败!");
                    }
                } catch (Exception e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                    JOptionPane.showMessageDialog(this, "预订失败!");
                } finally {
                    try {
                        database.closeCon(con);
                    } catch (Exception e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                    }
                }
            }

    }
```

全部高铁动车信息程序如下：

```
        fillTable1(hsrailway);
```

点击个人中心按钮，进入乘客个人中心界面。

点击返回按钮，返回主页和查询界面界面。

鼠标功能的实现：

```
    private void jTable1MousePressed(java.awt.event.MouseEvent evt) {
        // TODO add your handling code here:
          int row = jTable1.getSelectedRow();
    H_name = (String) jTable1.getValueAt(row, 0);
        H_seattype = (String) jTable1.getValueAt(row, 6);
        H_date = (String) jTable1.getValueAt(row, 5);
```

批注 [b18]: 见 hsorder 类：把选择的高铁动车信息填入订单：public int SelectionAdd(Connection con)

批注 [b19]: 见 hsorder 类：对应预订车票减一：public int H_numberMinus(Connection con,String H_name,String H_seattype,String H_date)

批注 [b20]: 见下文 filltable 功能的实现

批注 [b21]: 鼠标功能

```
        H_number = Integer.parseInt((String) jTable1.getValueAt(row, 8));
        }
```

Filltable 功能的实现：

```
private void fillTable(hsrailway hsrailway, String H_departure, String H_destination,String H_date)
{
            DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();
            dtm.setRowCount(0);
            Connection con = null;
            try {
                con = database.getCon();
             try {

            } catch (Exception ex){
                Logger.getLogger(主页和查询 JFrame.class.getName()).log(Level.SEVERE, null, ex);
            }
                ResultSet rs = co.Queryhsrailway(con,H_departure,H_destination,H_date);
                while (rs.next()) {
                    Vector v = new Vector();
                    v.add(rs.getString(1));
                    v.add(rs.getString(2));
                    v.add(rs.getString(3));
                    v.add(rs.getString(4));
                    v.add(rs.getString(5));
                    v.add(rs.getString(6));
                    v.add(rs.getString(7));
                    v.add(rs.getString(8));
                    v.add(rs.getString(9));
                    dtm.addRow(v);

                }
            }catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            } finally {
                try {
                    database.closeCon(con);
                } catch (Exception e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            }
}
private void fillTable1(hsrailway hsrailway){
            DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();
```

批注 [b22]: Filltable 功能

批注 [b23]: 见 control 类：根据出发地，到达地，出发日期查询高铁信息：public ResultSet Queryhsrailway(Connection con,String H_departure,String H_destination,String H_date)

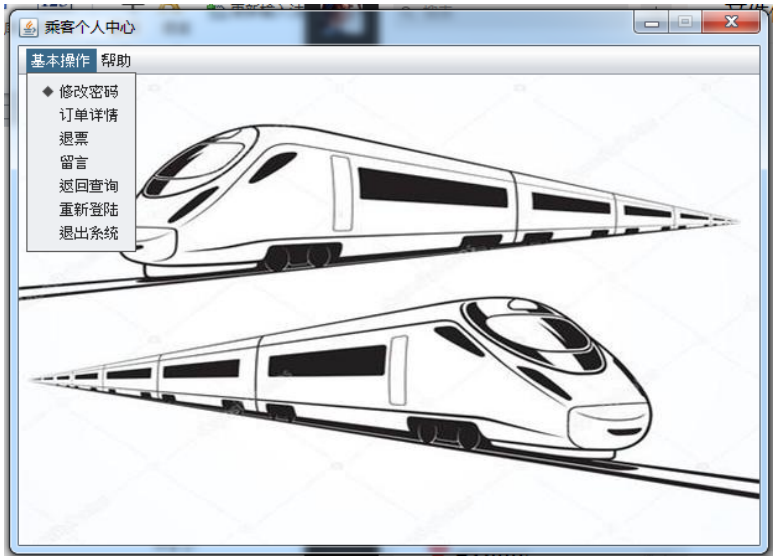
批注 [b24]: Filltable 功能

```
            dtm.setRowCount(0);
            Connection con = null;
            try {
                con = database.getCon();
                ResultSet rs = co.Queryallhsrailway(con);
                while (rs.next()) {
                    Vector v = new Vector();
                    v.add(rs.getString(1));
                    v.add(rs.getString(2));
                    v.add(rs.getString(3));
                    v.add(rs.getString(4));
                    v.add(rs.getString(5));
                    v.add(rs.getString(6));
                    v.add(rs.getString(7));
                    v.add(rs.getString(8));
                    v.add(rs.getString(9));
                    dtm.addRow(v);

                }
            }catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            } finally {
                try {
                    database.closeCon(con);
                } catch (Exception e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            }
}
```

批注 [b25]: 见 control 类：查询全部高铁信息：public ResultSet Queryallhsrailway(Connection con)

5.5 乘客个人中心界面



该界面中的基本操作下，有七个选项：修改密码，订单详情，退票，留言，返回系统，重新登录，退出系统。分别连接了修改密码，订单详情，退票，留言，返回系统，登录界面。点击退出系统，会出现提示。

其中，若>=60 岁的乘客点击订单详情，进入订单详情界面；若<60 岁的乘客点击订单详情，则进入订单详情 1。

订单详情按钮下的程序如下：

```
    private void jMenuItem5ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String P_id=  登录.currentpassenger.GetP_id();
        Boolean rs = null;
        Connection con = null;
    try {
        con=database.getCon();
    } catch (Exception ex) {
        Logger.getLogger(乘客个人中心 JFrame.class.getName()).log(Level.SEVERE, null, ex);
    }
            try {
                火车车次详情 JFrame1 a = new  火车车次详情 JFrame1();
                rs = a.Queryage(con,登录.P_id);
                System.out.println(rs);
            } catch (Exception ex) {
                Logger.getLogger(火车车次详情 JFrame1.class.getName()).log(Level.SEVERE,
null, ex);
            }
        if(rs == true){
                this.dispose();
```

批注 [b26]: 订单详情按钮程序

批注 [b27]: 见火车车次详情：识别年龄：public Boolean Queryage(Connection con,String P_id)

```
                new 订单详情 JFrame().setVisible(true);
            }
            else{
                this.dispose();
                new 订单详情 JFrame1().setVisible(true);
            }
        }
```

5.6 修改密码

该界面实现了一个功能：密码的修改。乘客输入原密码，新密码，再输入一遍新密码。
该界面有三个按钮：返回，修改，重置
修改按钮下的程序如下：

```
    private void jb_modifyActionPerformed(java.awt.event.ActionEvent evt) {
        try {
            // TODO add your handling code here:
            String oldPassword = new String(this.oldPasswordTxt.getPassword());
            String newPassword = new String(this.newPasswordTxt.getPassword());
            String passwordConfirm = new String(this.passwordConfirmTxt.getPassword());
            if (StringUtil.isEmpty(oldPassword)) {
                JOptionPane.showMessageDialog(this, "原密码不能为空!");
                return;
            }
            if (StringUtil.isEmpty(newPassword)) {
                JOptionPane.showMessageDialog(this, "新密码不能为空!");
                return;
```

批注 [b28]: 修改按钮程序

```
}
if (StringUtil.isEmpty(passwordConfirm)) {
    JOptionPane.showMessageDialog(this, "必须重复输入新密码!");
    return;
}
Connection con = null;
con=database.getCon();
String currentP_id=登录.currentpassenger.GetP_id();
passenger passenger = new passenger();
passenger.setP_id(currentP_id);
String rightOldPassword="";
try {
    rightOldPassword = co.QueryoldPassword(con, passenger);
} catch (SQLException ex) {
    Logger.getLogger(修改密码.class.getName()).log(Level.SEVERE, null, ex);
}
if (!oldPassword.equals(rightOldPassword)) {
    JOptionPane.showMessageDialog(this, "原密码错误，请重新输入!");
    return;
}
if (!newPassword.equals(passwordConfirm)) {
    JOptionPane.showMessageDialog(this, "两次输入的密码必须相同!");
    return;
}
passenger.setP_password(newPassword);
try {
    con = database.getCon();
    int modifyNum = co.ModifyPassword(con, passenger);
    if (modifyNum == 1) {
        JOptionPane.showMessageDialog(this, "密码修改成功!");
        this.resetValue();

    } else {
        JOptionPane.showMessageDialog(this, "密码修改失败!");
    }
} catch (Exception e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
    JOptionPane.showMessageDialog(this, "修改失败!");
} finally {
    try {
        database.closeCon(con);
    } catch (Exception e) {
        // TODO Auto-generated catch block
```

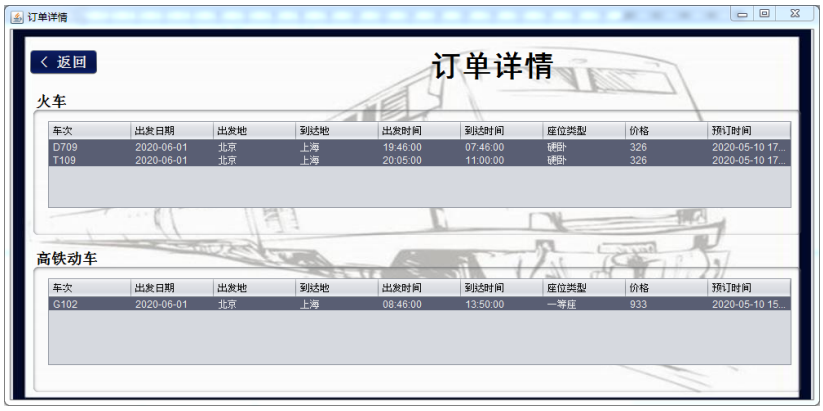
批注 [b29]: 见 control 类：识别乘客密码：public String QueryoldPassword(Connection con,passenger passenger)

批注 [b30]: 见 control 类：更新乘客密码：public int ModifyPassword(Connection con,passenger passenger)

```
                e.printStackTrace();
            }
        }
    } catch (Exception ex) {
        Logger.getLogger(修改密码.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

点击重置按钮，清空输入信息
点击返回按钮，返回乘客个人中心

5.7 订单详情 1（<60 岁的乘客出现该界面）



该界面实现两个功能：展示该乘客的火车订单和高铁动车订单
Filltable 程序的实现：

```
private void fillTable(train train) {
    DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();
    dtm.setRowCount(0);
    Connection con = null;
            torder sc=new torder();

    String P_id=  登录.currentpassenger.GetP_id();
            passenger.setP_id(P_id);
            System.out.println(passenger.GetP_id());
    try {
        con = database.getCon();
        ResultSet rs = co.Querytrainorder1(con,passenger);
        while (rs.next()) {
            Vector v = new Vector();
            v.add(rs.getString(1));
            v.add(rs.getString(2));
            v.add(rs.getString(3));
```

批注 [b31]: Filltable 功能

批注 [b32]: 见 control 类：查找该乘客的火车订单：public ResultSet Querytrainorder1(Connection con,passenger passenger)

```
                v.add(rs.getString(4));
                v.add(rs.getString(5));
                v.add(rs.getString(6));
                v.add(rs.getString(7));
                v.add(rs.getString(8));
                v.add(rs.getString(9));
                dtm.addRow(v);
            }
        }catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } finally {
            try {
                database.closeCon(con);
            } catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }
    private void fillTable1(hsrailway hsraiway) {
        DefaultTableModel dtm = (DefaultTableModel) jTable2.getModel();
        dtm.setRowCount(0);
        Connection con = null;
                hsorder sc=new hsorder();
        String P_id=  登录.currentpassenger.GetP_id();
                passenger.setP_id(P_id);
                System.out.println(passenger.GetP_id());
        try {
            con = database.getCon();
            ResultSet rs = co.Queryhsrailwayorder(con,passenger);
            while (rs.next()) {
                Vector v = new Vector();
                v.add(rs.getString(1));
                v.add(rs.getString(2));
                v.add(rs.getString(3));
                v.add(rs.getString(4));
                v.add(rs.getString(5));
                v.add(rs.getString(6));
                v.add(rs.getString(7));
                v.add(rs.getString(8));
                v.add(rs.getString(9));

                dtm.addRow(v);
```

批注 [b33]: Filltable 功能

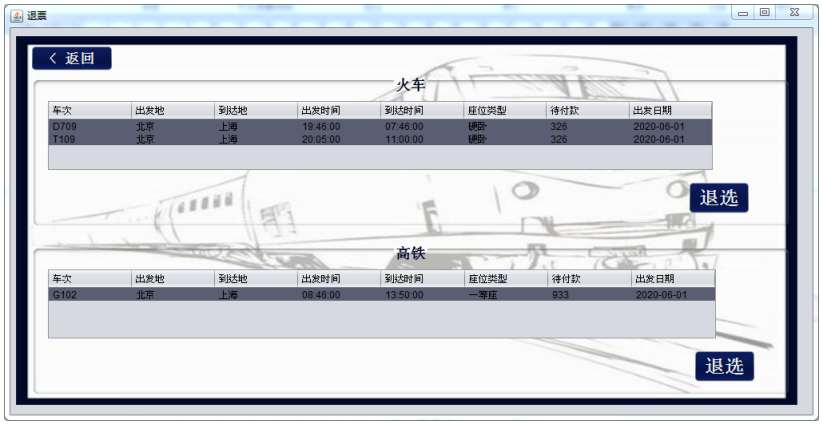
批注 [b34]: 见 control 类：查找该乘客的高铁动车订单：public ResultSet Queryhsrailwayorder(Connection con,passenger passenger)

```
            }
        }catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } finally {
            try {
                database.closeCon(con);
            } catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }
```

5.8 订单详情（>=60 岁的乘客出现该界面）

该界面实现两个功能：展示该乘客的火车订单和高铁动车订单
Filltable 程序与 5.6 的 filltable 差不多，少了软卧备注和硬卧备注，不赘述。

5.9 退票



该界面实现两个功能：火车和高铁动车退票。乘客通过点击表中数据进行退票

该界面中有三个按钮：退票，退票，返回。

第一个退票按钮下的程序：

```
private void jToggleButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
     if (T_name=="" && T_seattype=="") {
    JOptionPane.showMessageDialog(this, "请选择要退选的车次！ ");
    return;
    }
    int n = JOptionPane.showConfirmDialog(this, "确认要退选这个车次吗?");
    if (n == 0) {
        Connection con = null;
        String currentP_id =  登录.currentpassenger.GetP_id();
        torder sc= new torder();
                    sc.setT_name(T_name);
                    sc.setT_date(T_date);
                    sc.setT_seattype(T_seattype);
                    sc.setP_id(currentP_id);
        try {
            con = database.getCon();
            int selectionNum = sc.SelectionCancel(con);
            int selectedNum = sc.T_numberAdd(con,T_name,T_seattype,T_date);
            if (selectionNum == 1 && selectedNum == 1) {
                JOptionPane.showMessageDialog(this, "退选成功!");
                this.fillTable1(new torder());
            } else {
                JOptionPane.showMessageDialog(this, "退选失败!");
            }
        } catch (Exception e) {
            // TODO Auto-generated catch block
```

批注 [b35]: 第一个退票按钮程序

批注 [b36]: 见 torder 类：从订单删除所选火车票：public int SelectionCancel(Connection con)

批注 [b37]: 见 torder 类：该车次票数加一：public int T_numberAdd(Connection con,String T_name,String T_seattype,String T_date)

```
                e.printStackTrace();
                JOptionPane.showMessageDialog(this, "退选失败!");
            } finally {
                try {
                    database.closeCon(con);
                } catch (Exception e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            }
    }
    }
```

第二个退票按钮下的程序：

```
    private void jb_selectionCancelActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    if (H_name=="" && H_seattype=="") {
        JOptionPane.showMessageDialog(this, "请选择要退选的车次！ ");
        return;
        }
        int m = JOptionPane.showConfirmDialog(this, "确认要退选这个车次吗?");
        if (m == 0) {
            Connection con = null;
            String currentP_id =  登录.currentpassenger.GetP_id();
            hsorder sc= new hsorder();
                        sc.setP_id(currentP_id);
                        sc.setH_seattype(H_seattype);
                        sc.setH_name(H_name);
                          sc.setH_date(H_date);

            try {
                con = database.getCon();

                int selectionNum = sc.SelectionCancel(con);
                int selectedNum = sc.H_numberAdd(con,H_name,H_seattype,H_date);
                                    System.out.println(selectionNum);
                                    System.out.println(selectedNum);
                if (selectionNum == 1 && selectedNum >= 1) {
                    JOptionPane.showMessageDialog(this, "退选成功!");
                    this.fillTable2(new hsorder());
                } else {
                    JOptionPane.showMessageDialog(this, "退选失败!");
                }
            } catch (Exception e) {
                // TODO Auto-generated catch block
```

批注 [b38]: 第二个退票按钮程序

批注 [b39]: 见 hsorder 类：从订单删除所选高铁票：public int SelectionCancel(Connection con)

批注 [b40]: 见 hsorder 类：该车次票数加一：public int H_numberAdd(Connection con,String H_name,String H_seattype,String H_date)

```
                e.printStackTrace();
                JOptionPane.showMessageDialog(this, "退选失败!");
            } finally {
                try {
                    database.closeCon(con);
                } catch (Exception e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            }
    }
    }
```

鼠标功能的实现：

```
    private void tordertableMousePressed(java.awt.event.MouseEvent evt) {
        // TODO add your handling code here:
        int row = tordertable.getSelectedRow();
      T_name= (String) tordertable.getValueAt(row, 0);
           T_date = (String) tordertable.getValueAt(row, 7);
       T_seattype= (String) tordertable.getValueAt(row, 5);
    }

    private void hsordertableMousePressed(java.awt.event.MouseEvent evt) {
        // TODO add your handling code here:
        int row = hsordertable.getSelectedRow();
     H_name= (String) hsordertable.getValueAt(row, 0);
          H_date = (String) hsordertable.getValueAt(row, 7);
      H_seattype= (String) hsordertable.getValueAt(row, 5);
    }
```

Filltable 功能的实现：

```
        private void fillTable1(torder torder) {
        DefaultTableModel dtm = (DefaultTableModel) tordertable.getModel();
        dtm.setRowCount(0);
        Connection con = null;
                torder sc=new torder();

        String P_id=  登录.currentpassenger.GetP_id();
                passenger.setP_id(P_id);
                System.out.println(passenger.GetP_id());
        try {
            con = database.getCon();
            ResultSet rs = sc.PassengerList(con, passenger);
            while (rs.next()) {
                Vector v = new Vector();
                v.add(rs.getString(1));
```

批注 [b41]: 鼠标功能

批注 [b42]: Filltable 功能

```
                v.add(rs.getString(2));
                v.add(rs.getString(3));
                v.add(rs.getString(4));
                v.add(rs.getString(5));
                v.add(rs.getString(6));
                v.add(rs.getString(7));
                v.add(rs.getString(8));

                dtm.addRow(v);

            }
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } finally {
            try {
                database.closeCon(con);
            } catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }

    private void fillTable2(hsorder hsorder) {
        DefaultTableModel dtm = (DefaultTableModel) hsordertable.getModel();
        dtm.setRowCount(0);
        Connection con = null;
                hsorder sc=new hsorder();

        String P_id =  登录.currentpassenger.GetP_id();
                passenger.setP_id(P_id);
                System.out.println(passenger.GetP_id());
        try {
            con = database.getCon();
            ResultSet rs = sc.PassengerList(con, passenger);
            while (rs.next()) {
                Vector v = new Vector();
                v.add(rs.getString(1));
                v.add(rs.getString(2));
                v.add(rs.getString(3));
                v.add(rs.getString(4));
                v.add(rs.getString(5));
                v.add(rs.getString(6));
```

批注 [b43]: Filltable 功能

```
                    v.add(rs.getString(7));
                    v.add(rs.getString(8));

                    dtm.addRow(v);

                }
            } catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            } finally {
                try {
                    database.closeCon(con);
                } catch (Exception e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            }
        }
```

点击返回按钮，返回乘客个人中心界面。

5.10 留言界面

该界面实现了一个功能：留言。乘客把留言写在界面中，输入到 Mysql 的 message 表中。管理员在留言显示中可以查看并删除。

该界面有两个按钮：确定，返回。

确定按钮下的程序如下：

```
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        try{
            database database=new database();
            int m=0;
            String M_message = this.message.getText();

            message message=new message();
            message.setM_message(M_message);

            if(StringUtil.isEmpty(M_message)){
                fo.ShowInfo(this, "公告不能为空!");
            }

            Connection con=null;
            con=database.getCon();
            if (co.CanmessageAdd(con,"message","M_message",M_message)){
                m = co.messageAdd(con,message);

                if(m==1){
                    fo.ShowInfo(this, "发布成功!");
                    this.resetValue();
                }else{
                    fo.ShowInfo(this, "发布失败!");
                }
            } else {
            }
        } catch (Exception ex) {
            Logger.getLogger(留言.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
```

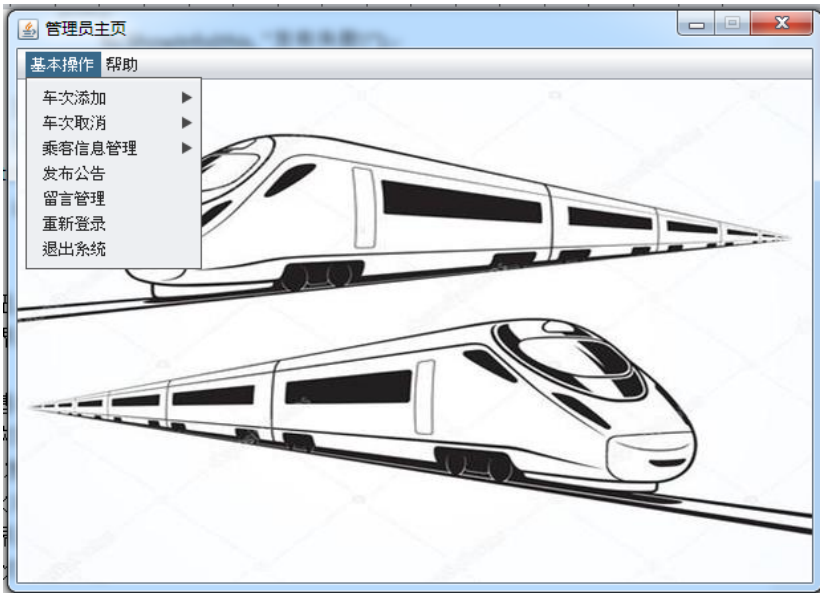
批注 [b44]: 确定按钮程序

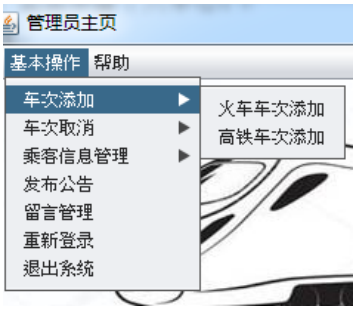
批注 [b45]: 见 control 类：判断该留言能否添加：public Boolean CanmessageAdd(Connection con,String tablename,String columnname,String M_number)

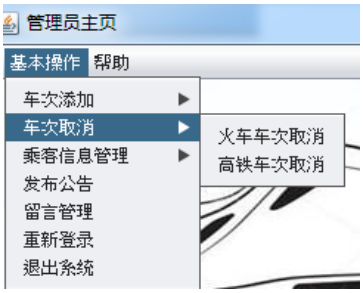
批注 [b46]: 见 control 类：往数据库表中输入留言：public int messageAdd(Connection con,message message)

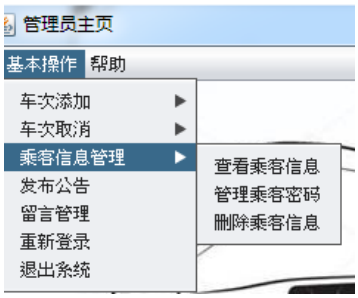
点击返回按钮，返回乘客个人中心。

5.11 管理员界面


管理员主页
基本操作
帮助
车次添加
车次取消
乘客信息管理
发布公告
留言管理
重新登录
退出系统

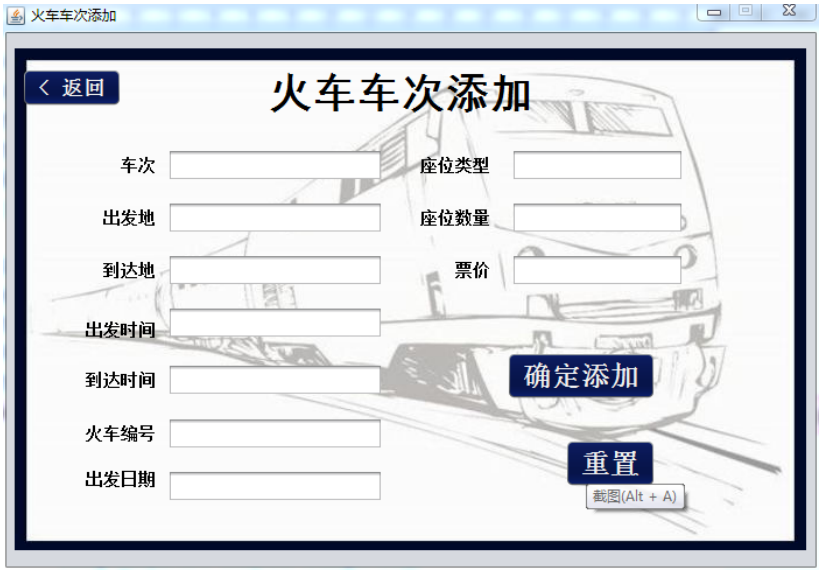

管理员主页
基本操作
帮助
车次添加
车次取消
乘客信息管理
发布公告
留言管理
重新登录
退出系统
火车车次添加
高铁车次添加


管理员主页
基本操作
帮助
车次添加
车次取消
乘客信息管理
发布公告
留言管理
重新登录
退出系统
火车车次取消
高铁车次取消



该界面中的基本操作下，有七个选项：车次添加（火车车次添加，高铁车次添加），车次取消（火车车次取消，高铁车次取消），乘客信息管理（查看乘客信息，管理乘客密码，删除乘客信息），发布公告，留言管理，重新登录，退出系统。分别连接了火车车次添加，高铁车次添加，火车车次取消，高铁车次取消，查看乘客信息，管理乘客密码，管理员删除乘客信息，登录界面。点击退出系统，会出现提示。

5.12 火车车次添加界面



该界面实现了车次添加的功能。

管理员依次填入火车相关信息，点击确定按钮，系统识别 Mysql 数据库的相关数据，若无重复则添加成功，数据加入到 Mysql 的 train 和 train_price 的表中。

该界面有三个按钮：确定添加，重置，返回。

确定添加按钮下的程序：

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    try{     // TODO add your handling code here:
        database database=new database();
```

批注 [b47]: 确定添加按钮程序

```
                int n=0;
                int m=0;
                String T_departure=this.jTextField2.getText();
                String T_destination=this.jTextField3.getText();
                String T_leavetime=this.jTextField4.getText();
                String T_arrivetime=this.jTextField6.getText();
                String T_name=this.jTextField1.getText();
                String T_id=this.jTextField5.getText();
                String T_seattype=this.jTextField7.getText();
                String T_price=this.jTextField9.getText();
                String T_number=this.jTextField8.getText();
                String T_date=this.jTextField10.getText();
SimpleDateFormat format = new SimpleDateFormat("hh:mm:ss");
java.util.Date d = null;
java.util.Date a= null;
try {
  d = format.parse(T_leavetime);
    a = format.parse(T_arrivetime);
} catch (Exception e) {
  e.printStackTrace();
}
     Time date = new java.sql.Time(d.getTime());
      Time c = new java.sql.Time(a.getTime());
                train train=new train();
                train_price train_price=new train_price();
                train.setT_departure(T_departure);
                train.setT_destination(T_destination);
                train.setT_leavetime(date);
                train.setT_arrivetime(c);
                train.setT_name(T_name);
                train.setT_id(T_id);
                train.setT_date(T_date);
                train_price.setT_id(T_id);
                train_price.setT_name(T_name);
                train_price.setT_seattype(T_seattype);
                train_price.setT_price(T_price);
                train_price.setT_number(T_number);
                train_price.setT_date(T_date);
                if(StringUtil.isEmpty(T_name)){
                      fo.ShowInfo(this, "车次不能为空!");
                      return;
                }
                if(StringUtil.isEmpty(T_departure)){
                      fo.ShowInfo(this, "出发地不能为空!");
```

```
                return;
            }
            if(StringUtil.isEmpty(T_destination)){
                fo.ShowInfo(this, "到达地称不能为空!");
                return;
            }
            if(StringUtil.isEmpty(T_leavetime)){
                fo.ShowInfo(this, "出发时间不能为空!");
                return;
            }
            if(StringUtil.isEmpty(T_arrivetime)){
                fo.ShowInfo(this, "到达时间不能为空!");
                return;
            }

            if(StringUtil.isEmpty(T_seattype)){
                fo.ShowInfo(this, "座位类型不能为空!");
                return;
            }
            if(StringUtil.isEmpty(T_number)){
                fo.ShowInfo(this, "座位余量不能为空!");
                return;
            }
            if(StringUtil.isEmpty(T_price)){
                fo.ShowInfo(this, "票价不能为空!");
                return;
            }
              if(StringUtil.isEmpty(T_date)){
                fo.ShowInfo(this, "出发日期不能为空!");
                return;
            }
            Connection con=null;
            con=database.getCon();
            if(co.Cantrain_priceAdd(con,"train_price", "T_name",
T_id)&&co.CantrainAdd(con,"train","T_name",T_id)){
                m = co.train_priceAdd(con, train_price);
                n = co.trainAdd(con,train);
                if(m==1&&n==1){
                    fo.ShowInfo(this, "添加成功");
                    this.resetValue();
                }else{
                    fo.ShowInfo(this, "添加失败!");
                }
            }
```

批注 [b48]: 见 control 类：判断火车车次对应的价格能否添加：public Boolean Cantrain_priceAdd(Connection con,String tablename,String columnname,String T_id)

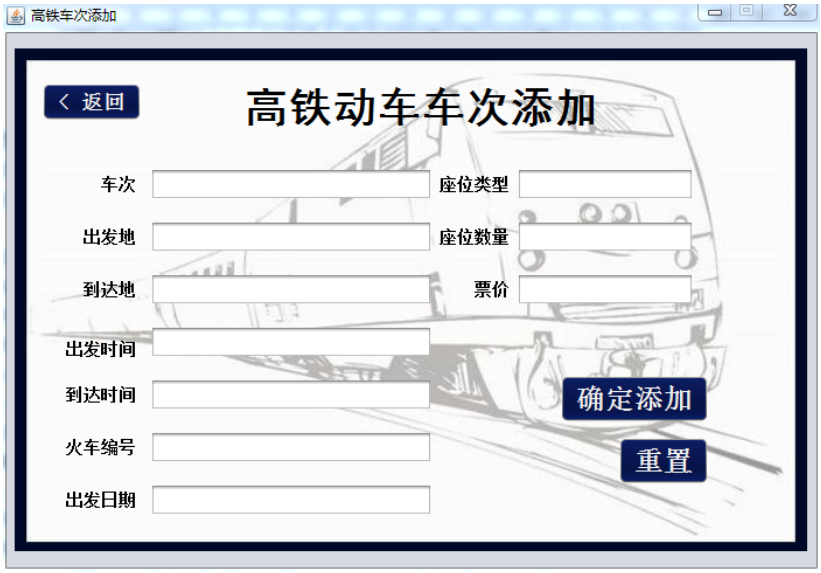
批注 [b49]: 见 control 类：判断火车车次能否添加：public Boolean CantrainAdd(Connection con,String tablename,String columnname,String T_id)

```
                else
                {
                    fo.ShowInfo(this, "火车编号重复!");
                }

            } catch (Exception ex) {
                Logger.getLogger(火车车次添加 JFrame.class.getName()).log(Level.SEVERE, null,
ex);
            }
        }
```

点击重置按钮，清空火车添加的信息。
点击返回按钮，返回管理员界面。

5.13 高铁车次添加界面



该界面实现了车次添加的功能。
管理员依次填入高铁动车相关信息，点击确定按钮，系统识别 Mysql 数据库的相关数据，若无重复则添加成功，数据加入到 Mysql 的 hsrailway 和 hsraiway_price 的表中。
该界面有三个按钮：确定添加，重置，返回。
高铁车次添加和火车车次添加异曲同工，在这里不赘述。

5.14 火车车次取消界面

该界面实现了车次取消的功能。
管理员输入需要取消的火车编号，日期和车次。点击删除按钮，系统识别 Mysql 数据库的相关数据，若识别到信息，则删除 Mysql 的 train 和 train_price 表中的相关信息。
该界面有两个按钮：删除，返回。
删除按钮下的程序如下：

```
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String T_id = this.id.getText();
        String T_name = this.t_name.getText();
          String T_date = this.date.getText();
        int deleteNum = 0;
        int deleteNum1 = 0;
        if (StringUtil.isEmpty(T_name)) {
            fo.ShowInfo(this, "车次不能为空!");
            return;
        }

        int n = co.ConfirmDelete(this,"车次", T_name);
        if (n == JOptionPane.YES_OPTION) {
            Connection con = null;
            try {
                con = db.getCon();
            } catch (Exception ex) {
                Logger.getLogger(火车车次取消 JFrame.class.getName()).log(Level.SEVERE,
null, ex);
```

批注 [b50]: 删除按钮程序

批注 [b51]: 见 control 类：判断能否删除：public int ConfirmDelete(Component com,String str,String Sno)

```
                }

                try {
                    deleteNum=co.trainDelete(con,T_id,T_date,T_name);
                    deleteNum1 = co.train_priceDelete(con,T_id,T_date,T_name);
                } catch (SQLException ex) {
                    Logger.getLogger(火车车次取消 JFrame.class.getName()).log(Level.SEVERE,
null, ex);
                }
            }
            else{
                return;
            }

            if (deleteNum == 1&&deleteNum1 == 1) {
                fo.ShowInfo(this, "删除成功!");
                this.resetValue();
            }
            else {
                fo.ShowInfo(this, "删除失败!");
            }
    }
```

批注 [b52]: 见 control 类：删除对应车次：public int trainDelete(Connection con,String T_id, String T_date,String T_name)

批注 [b53]: 见 control 类：删除火车车次对应的价格：public int train_priceDelete(Connection con,String T_id, String T_date,String T_name)

点击返回按钮，返回管理员界面。

5.15 高铁车次取消界面

该界面实现了车次取消的功能。
管理员输入需要取消的高铁编号，日期和车次。点击删除按钮，系统识别 Mysql 数据库的相关数据，若识别到信息，则删除 Mysql 的 hsrailway 和 hsraiway_price 表中的相关信息。
该界面有两个按钮：删除，返回。
高铁车次取消和火车车次取消异曲同工，在这里不赘述。

5.16 查看乘客信息界面

该界面实现了查看乘客信息的功能。
管理员输入想要查看的乘客信息（P_id），点击查询。系统识别数据，查询 Mysql 数据库 passenger 表中的数据。
该界面有两个按钮：查询，返回。
查询按钮下的程序如下：

批注 [b54]: 查询按钮程序

```
private void jb_searchActionPerformed(java.awt.event.ActionEvent evt) {
    {
        String P_id = this.SnoTxt.getText();
        Connection con=null;
        passenger passenger=new passenger();
        this.fillTable1(passenger,P_id);

        if (StringUtil.isEmpty(P_id)) {
            fo.ShowInfo(this, "信息为空!");
            return;
        }
    }
}
```

Filltable 功能的实现：

批注 [b55]: Filltable 功能

```
private void fillTable(passenger passenger) {
    DefaultTableModel dtm = (DefaultTableModel) studentTable.getModel();
    dtm.setRowCount(0);
    Connection con = null;
```

```
        try {
            con = database.getCon();
            ResultSet rs = co.Queryallpassenger(con);


            while (rs.next()) {
                Vector v = new Vector();
                v.add(rs.getString("P_id"));
                v.add(rs.getString("P_name"));
                v.add(rs.getString("P_password"));
                v.add(rs.getString("P_phone"));
                v.add(rs.getString("P_address"));
                v.add(rs.getString("P_email"));
                v.add(rs.getString("P_age"));
                v.add(rs.getString("P_sex"));
                dtm.addRow(v);
            }
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } finally {
            try {
                database.closeCon(con);
            } catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }
    private void fillTable1(passenger passenger,String P_id) {
        DefaultTableModel dtm = (DefaultTableModel) studentTable.getModel();
        dtm.setRowCount(0);
        Connection con = null;

        try {
            con = database.getCon();
            ResultSet rs = co.Querypassenger(con,P_id);

            while (rs.next()) {
                Vector v = new Vector();
                v.add(rs.getString("P_id"));
                v.add(rs.getString("P_name"));
```

批注 [b56]: 见 control 类：查询所有乘客全部信息：public ResultSet Queryallpassenger(Connection con)

批注 [b57]: Filltable 功能

批注 [b58]: 见 control 类：查询某乘客的全部信息：public ResultSet Querypassenger(Connection con,String P_id)

```
                v.add(rs.getString("P_password"));
                v.add(rs.getString("P_phone"));
                v.add(rs.getString("P_address"));
                v.add(rs.getString("P_email"));
                v.add(rs.getString("P_age"));
                v.add(rs.getString("P_sex"));
                dtm.addRow(v);
            }
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } finally {
            try {
                database.closeCon(con);
            } catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }
```

点击返回按钮，返回管理员界面。

5.17 乘客密码管理（管理乘客密码界面）

管理员点击表格内容或者输入乘客身份证号和乘客密码。系统识别 Mysql passenger 中的（P_id），再对（P_password）进行修改。

该界面有两个按钮：修改，返回。
修改按钮下的程序如下：

```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    String P_id = this.jTextField2.getText();
    String P_password = this.jTextField3.getText();
    passenger.setP_id(P_id);
    passenger.setP_password(P_password);
    if (StringUtil.isEmpty(P_id)) {
        JOptionPane.showMessageDialog(this, "身份证号不能为空!");
        return;
    }
    if (StringUtil.isEmpty(P_password)) {
        JOptionPane.showMessageDialog(this, "用户密码不能为空!");
        return;
    }
    Connection con = null;
    try {
        con=database.getCon();
    } catch (Exception ex) {
        Logger.getLogger(管理乘客密码.class.getName()).log(Level.SEVERE, null, ex);
    }
    int num =0;
try {
    if (co.CanModify(con, "passenger", "P_id", P_id)) {
        try {
            num = co.ModifyPassword(con, passenger);
        } catch (Exception ex) {
            Logger.getLogger(管理乘客密码.class.getName()).log(Level.SEVERE, null, ex);
        }
        if (num == 1 ){
            fo.ShowInfo(this, "修改成功!");
            this.resetValue();
            passenger passenger =new passenger();
            this.fillTable(passenger);
        }
        else {
            fo.ShowInfo(this, "修改失败!");
            this.resetValue();
        }
    }
    else
    {
        fo.ShowInfo(this, "没有这个用户");
    }
```

批注 [b59]: 修改按钮程序

批注 [b60]: 见 control 类：判断是否存在该乘客密码：public Boolean CanModify(Connection con,String tablename,String columnname,String P_id)

批注 [b61]: 见 control 类：修改该乘客的密码：public int ModifyPassword(Connection con,passenger passenger)

```
    } catch (SQLException ex) {
        Logger.getLogger(管理乘客密码.class.getName()).log(Level.SEVERE, null, ex);
    }
    }
```

Filltable 功能的实现

```
private void fillTable(passenger passenger) {
        DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();
        dtm.setRowCount(0);
        Connection con = null;

        try {
            con = database.getCon();
            ResultSet rs = co.passengerList(con, passenger);
            while (rs.next()) {
                Vector v = new Vector();
                v.add(rs.getString("p_name"));
                v.add(rs.getString("p_id"));
                v.add(rs.getString("p_password"));
                dtm.addRow(v);
            }
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } finally {
            try {
                database.closeCon(con);
            } catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }
```

批注 [b62]: Filltable 功能

鼠标功能的实现：

```
    private void jTable1MousePressed(java.awt.event.MouseEvent evt) {
        // TODO add your handling code here:
            int row = jTable1.getSelectedRow();
            this.jTextField2.setText((String) jTable1.getValueAt(row, 1));
            this.jTextField3.setText((String) jTable1.getValueAt(row, 2));
    }
```

批注 [b63]: 鼠标功能实现

点击返回按钮，返回管理员界面。

5.18 删除乘客信息界面

该界面中，实现了删除乘客信息的功能。

管理员点击表格或者输入乘客身份证号码（P_id）和乘客姓名（P_name），点击删除，系统识别 Mysql 中 passenger 表的 P_id 和 P_name，进行删除。

该界面有两个按钮：删除，返回。

删除按钮下的程序如下：

```
    private void deleteActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String P_id = this.jTextField2.getText();
        String P_name = this.jTextField3.getText();
        passenger.setP_id(P_id);
        passenger.setP_name(P_name);
        int deleteNum = 0;
    if (StringUtil.isEmpty(P_id)) {
        fo.ShowInfo(this, "身份证号不能为空!");
        return;
        }

        int n = co.ConfirmDelete(this,"乘客", P_name);
        if (n == JOptionPane.YES_OPTION) {
    Connection con = null;
            try {
                con = database.getCon();
            } catch (Exception ex) {
                Logger.getLogger(管理员删除乘客信息.class.getName()).log(Level.SEVERE,
null, ex);
```

批注 [b64]: 删除按钮程序

批注 [b65]: 见 control 类：判断是否存在该乘客：public int ConfirmDelete(Component com,String str,String Sno)

```
                }

                try {
                    deleteNum=co.passengerDelete(con,P_id);
                } catch (SQLException ex) {
                    Logger.getLogger(管理员删除乘客信息.class.getName()).log(Level.SEVERE,
null, ex);
                }
                }
            else{
                return;
            }

        if (deleteNum == 1) {
                fo.ShowInfo(this, "删除成功!");
                this.resetValue();
                try {
                    this.fillTable();
                } catch (Exception ex) {
                    Logger.getLogger(管理员删除乘客信息.class.getName()).log(Level.SEVERE,
null, ex);
                }
            }
         else {
            fo.ShowInfo(this, "删除失败!");
                };
      }
```

鼠标功能的实现：

```
            private void jTable1MousePressed(java.awt.event.MouseEvent evt) {
                // TODO add your handling code here:
                  int row = jTable1.getSelectedRow();
                    this.jTextField2.setText((String) jTable1.getValueAt(row, 0));
                    this.jTextField3.setText((String) jTable1.getValueAt(row, 1));
            }
```

Filltable 功能的实现：

```
private void fillTable() {
            DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();
            dtm.setRowCount(0);
            Connection con = null;

            try {
                con = database.getCon();
                ResultSet rs = co.passengerList(con, passenger);
                while (rs.next()) {
```

批注 [b66]: 见 control 类：删除该乘客信息：public int passengerDelete(Connection con, String P_id)

批注 [b67]: 鼠标功能实现

批注 [b68]: Filltable 功能

```
                Vector v = new Vector();
                                v.add(rs.getString("P_id"));
                v.add(rs.getString("P_name"));
                v.add(rs.getString("P_phone"));
                dtm.addRow(v);
            }
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } finally {
            try {
                database.closeCon(con);
            } catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }
```

点击返回按钮，返回管理员界面。

5.19 发布公告

该界面实现管理员发布公告功能。管理员输入公告，添加进 Mysql notice 表中。在主页和查询界面表格中显示出来。

该界面与留言界面类似，不做赘述。

5.20 留言管理（留言显示界面）

该界面实现了留言的删除功能。管理员点击表格内容，点击删除按钮，进行删除。
该界面有两个按钮：删除，返回。

```
private void deleteActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String M_message = this.jTextField1.getText();
        int deleteNum = 0;
if (StringUtil.isEmpty(M_message)) {
        fo.ShowInfo(this, "留言内容不能为空!");
        return;
        }

        int n = co.ConfirmDelete(this,"message", M_message);
        if (n == JOptionPane.YES_OPTION) {
Connection con = null;
            try {
                con=database.getCon();
            } catch (Exception ex) {
                Logger.getLogger(留言显示.class.getName()).log(Level.SEVERE, null, ex);
            }

            try {
                deleteNum=co.messageDelete(con,M_message);
            } catch (SQLException ex) {
                Logger.getLogger(留言显示.class.getName()).log(Level.SEVERE, null, ex);
```

批注 [b69]: 删除按钮程序

批注 [b70]: 见 control 类：在提示中加入想要删除的留言内容：public int ConfirmDelete(Component com,String str,String Sno)

批注 [b71]: 见 control 类：删除所选留言：public int messageDelete(Connection con, String M_message)

```
                }
                }
            else{
                return;
            }

        if (deleteNum == 1) {
                fo.ShowInfo(this, "删除成功!");
                this.resetValue();
                try {
                    this.fillTable();
                } catch (Exception ex) {
                    Logger.getLogger(留言显示.class.getName()).log(Level.SEVERE, null, ex);
                }
            }
         else {
            fo.ShowInfo(this, "删除失败!");
                };
        }
```

鼠标功能实现：

```
            private void jTable1MousePressed(java.awt.event.MouseEvent evt) {
                // TODO add your handling code here:
                int row = jTable1.getSelectedRow();
                this.jTextField1.setText((String) jTable1.getValueAt(row, 0));
            }
```

批注 [b72]: 鼠标功能实现

Filltable 功能实现：

```
        private void fillTable() {
                DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();
                dtm.setRowCount(0);
                Connection con = null;

                try {
                    con = database.getCon();
                    ResultSet rs = co.messageList(con,message);
                    while (rs.next()) {
                        Vector v = new Vector();
                                    v.add(rs.getString("M_message"));
                        dtm.addRow(v);
                    }
                } catch (Exception e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                } finally {
                    try {
```

批注 [b73]: Filltable 功能

```
                database.closeCon(con);
            } catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }
```

点击返回按钮，返回管理员界面。

四、课程设计体会

颜雨祺 18110218：

本学期的课设真的让我成长了很多，无论是心态还是知识获得，都让我有了一个飞跃。作为组长我认为有责任去承担责任，尽量让小组气氛积极和谐。这三个月里，我们小组三个人共同经历了探讨系统功能、系统开发、错误探讨。从最开始的逻辑不对，每个人都很崩溃到后来心平气并且积极面对一个个难题。尤其是最后几周，让我真的很感动，三个人每天上完课晚上编写程序，相互鼓励。虽然在写程序的时候会有一些分歧与矛盾，但我们积极沟通，让组内更加团结。

通过这次的课设我对实践出真知这句话有了全新的体会。从最开始的什么也不会，对于程序功能开发一点概念都没有，到后来通过看书或者问老师，慢慢变得有了想法也有了信心与能力写下这一串串的代码。最后看到自己写出来的系统能够正常运行，真的很自豪。

最后，我很感谢老师的帮助。谢谢老师在我们每次问一些很简单的问题的时候耐心的讲解。甚至帮我一起找程序里面的错误。老师的鼓励让我变得更加有信心去写好每一个代码，实现每一个功能。

王慧 18110219：

为期三个月的课设，令我感到体会满满，收获满满！心理素质也得到了磨炼，突然就接近尾声了，写着这个个人体会竟然还有一丝感动，从起初的不敢想象如何完成，到一个个界面成功运行，内心的激动和喜悦是用那些感叹词都无法描述的。

我第一个做的界面是登录，当时我认为代码都没有问题，但就是一直报错，后来问老师才知道，就是查询数据库的 sql 语句出现了问题，而这个问题就是少了一个空格，知道问题所在的我真的既难过又兴奋，难过自己找了好几天的错竟然就是一个空格的问题，兴奋这个登录功能实现了！这个登录功能的实现给了我信心，让我有决心做好后面的工作。但其实每个功能的实现都是充满坎坷的，每次点击运行文件时都会有那么点紧张与期盼，总是会遇到各种小问题，小组内进行讨论解决，找老师寻求帮助，每次问题得以解决，都会觉得自己又进步了一点，在各种报错中成长，其实感觉也很好！

课设虽然接近尾声了，但是做课设的这股劲儿要继续保持，课设带给我的不仅是知识储备增加，更是一种克服困难的力量！过程艰辛，但一切都值得！

自来乐·白克力 18110260：

这个学期经历了“艰难险阻”的对象编程与数据库技术这门课要结束了，经过这三个月来的历练，我的收获颇丰。

首先，对于上个学期数据库没有好好学的人来说，这无疑是巨大的挑战。无论是具体的步骤设计，还是从代码开发，都是补足上个学期遗憾的好机会。从最开始编写代码的迷惘，

和到现在能够独立自主完成某一个界面的开发，这其中的历练只有自己懂（跪了。）

其次，团队合作也是一种新的挑战。团队合作并非易事，特别是之前从来没有这么长时间的共同工作的经历，这次课程设计也是给了我们磨合到完美配合的机会。我相信如果有下次相同的课程设计作业，小组成员之间一定能够更加完美的配合。

最后，我想说的是，此次课程设计给了我一次磨炼自己和重新认识自己专业的机会。让我从排斥计算机、排斥编程，到慢慢接受，并且不断提升自己，这个过程是充实的。看到成品的心情，不是买一两件衣服或者看两个小时电影能够比拟的。

这次课程设计，让我有了很多收获，也让我懂了团队合作的重要性以及计算机行业的深奥。我们接触的只是皮毛，希望自己以后能够拥有更多的学习机会。