《java项目开发》期末作品文档

**基于javaweb的停车场管理系统**

目录

[一、 系统分析 1](#_Toc60488349)

[1.1 技术可行性 1](#_Toc60488350)

[1.2 经济可行性 1](#_Toc60488351)

[1.3 操作可行性 1](#_Toc60488352)

[二、 需求分析 2](#_Toc60488353)

[2.1 系统功能描述 2](#_Toc60488354)

[三、 系统设计 2](#_Toc60488355)

[3.1 USER类 3](#_Toc60488356)

[3.2 JOURNAL类 3](#_Toc60488357)

[3.3 数据库的实现 5](#_Toc60488358)

[四、 系统详细设计与实现 9](#_Toc60488359)

[4.1 用户登录 9](#_Toc60488360)

[4.2 系统信息管理 11](#_Toc60488361)

[4.3 车位信息管理 11](#_Toc60488362)

[4.4 IC卡管理 15](#_Toc60488363)

[4.5 固定车主管理 16](#_Toc60488364)

[4.6 临时车主管理 16](#_Toc60488365)

[4.7 系统功能操作 17](#_Toc60488366)

[五、 系统测试 17](#_Toc60488367)

[5.1 用户登录 18](#_Toc60488368)

[5.2 添加临时车主 19](#_Toc60488369)

[5.3 添加角色信息 20](#_Toc60488370)

[六、 总结 22](#_Toc60488371)

[七、 参考文献 23](#_Toc60488372)

1. 系统分析
2. 技术可行性

在停车场管理系统的设计中，MVC（Model-View-Control）设计模式贯穿了整系统，框架采用SpringMVC+JDBC组合结构。

* 模型层

模型层主要是逻辑处理。 在本系统中，提供处理数据持久化，JavaBean对业务逻辑Service封装。

* 视图层

停车场管理系统采用JSP来实现视图层。

* 控制层

采用SpringMVC技术来处理前台请求与业务逻辑层的交互。

在停车场管理系统的设计中应用MVC设计模式，便于开发人员设计代码，由于这三个逻辑可以同时进行，提高了效率，节省时间，且由于分层，将代码隔离开来，方便开发人员操作。且停车场管理系统采用RFID卡等等技术，方便车辆出入，增加智能化，便于管理。可以从这个角度得出，开发智能停车场是可行的。

1.2 经济可行性

主要是对开发本系统的经济效益进行评价。根据调查了解到，大多数公共场所出现停车难、车位少等问题，停车场的效率十分低下，导致停车场的收益降低，且不安全。针对这种情况，有必要对停车场低效率的原因进程分析，并为提高停车场的效率提出改进的对策。经过分析调查，我们得出的停车场管理系统相当的有用，从这个角度来说，开发智能停车场管理系统是可行的。根据花费的查阅资料时间和框架设计时间，基本的功能实现需要差不多3个月左右，从2017年1月到4月，实现再加上相关功能的测试以及代码实现要两个2星期左右，对于那些智能化功能写的接口，待技术引进再后期磨合，实现需待定。总的来说4月份可设计完毕。根据系统要求规定，系统需要配合摄像头、引进RFID卡技术等，需要花费一笔费用，再预估计范围之内，从此看来，开发智能停车场是可行的。

1.3 操作可行性

开发所用的开发工具是eclipse，这个软件在各种计算机上都可以支持运行，并且操作方便，用户易上手。操作员经过简单的讲解就可以使用此系统，从这个方面而言，开发智能停车场管理系统是可行的。

1. 需求分析
2. 系统功能描述

系统信息管理：包括添加角色、管理角色、添加用户、管理用户、管理角色和管理用户模块可以对角色和用户进行增、删、改。

IC卡信息管理：添加IC卡类型，管理IC卡类型。管理IC卡信息可以对车主的IC卡信息进行增、删、改。

车位信息管理：管理车位、添加车位，管理车位信息模块可以对车主车位信息进行增、删、改。

固定车主停车管理：出入场设置、停车场信息管理。出入场设置可以设用户的入场和出场，相应的在停车信息中有一个展现。停车信息管理可以对车主的停车信息进行查询和删除。

临时车辆信息管理：车主入场信息，车主出场设置。设置车主入场，在相应的表中展现，在车主出场时进行收费，临时车主停车信息管理可以对临时车主的信息进行删除和查询。

收费管理：管理收费、添加收费信息。管理收费信息可以对车主的费用进行一个收费，可以对车主的收费信息进行查询和删除。

打印报表：查看报表信息，对固定车主和临时车主相关信息进行打印。

系统功能操作：修改密码、退出系统。

1. 系统设计

本章内容包括系统类分析、核心业务设计、数据库设计三个部分，其中系统类分析需要给出系统类图和类的说明，需要注意在给出类图的时候只需要给出核心业务类，对辅助类如界面类、数据库连接类这种类型的不需要给出，如果所涉及的类（包括类当中的内容）较多时，应首先给出类间关系图（图中的类只有名称没有属性和方法），之后在单个类说明中给出具体类的类图，类图同样需要用WORD进行绘制。

核心业务设计主要描述的是完成某项具体业务时，类之间的消息配合，用时序图进行表述，时序图可以用其他工具绘制，但不能够有其他的颜色出现。

数据库设计主要描述在系统中需要持久化的对象，需要给出ER图，如果系统所涉及的实体(或实体的属性)较多时同样采用分层绘制的方法，顶层图给出实体间关系，下层图针对每一个实体进行具体绘制。

1. USER类

USER类用于记录系统的用户信息，并完成登陆和注销的操作。具体说明如表4-1所示。

USER类功能表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 类功能描述 | 处理和用户相关的业务操作 | | | | | |
| 所在包名称 | DAL | | | | | |
| 继承对象 | 无 | | | | | |
| 实现对象 |  | | | | | |
| 类属性 | | | | | | |
| 保护属性 | 类型 | 名称 | 描述 | | 备注 | |
| Public | String | ID | 用户ID | | 长度限制20位 | |
| Public | String | Name | 姓名 | | 长度限制20位 | |
| public | String | Password | 密码 | | 长度限制20位 | |
| 主要实现方法 | | | | | | |
| 保护属性 | 方法名 | 输入参数 | | 输出参数 | | 方法功能描述 |
| Public | login | string id，string pwd | | String" | | 登陆系统，用户输入ID和PASSWORD后进行验证，验证通过返回用户名，并登记日志，验证失败返回空串。 |
| public | logout | string id | | Void | | 注销系统，注销成功清理用户的相关信息并登记日志。 |

1. JOURNAL类

JOURNAL类由5个类组成。分别是Fixed类、Temp类、Role类、Card类、Seat类 。

Fixed类记录固定车主信息，具体情况下如表4-2所示。

FIxed类功能表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 类功能描述 | 处理和固定车主的业务操作 | | | | | | |
| 所在包名称 | DAL | | | | | | |
| 继承对象 | 无 | | | | | | |
| 实现对象 | 表4-3FIxed类功能表 | | | | | | |
| 类属性 | | | | | | | |
| 保护属性 | 类型 | | 名称 | 描述 | | | 备注 |
| Public | String | | Fix\_id | 记录编号 | | | 长度限制20位 |
| Public | String | | Card\_id | IC卡号 | | | 长度限制20位 |
| 主要实现方法 | | | | | | | |
| 保护属性 | 方法名 | 输入参数 | | | 输出参数 | 方法功能描述 | |
| Public | Update | string Fix\_id，string pwd | | | int | 进入固定车主信息中，点击编辑，可修改固定车主信息 | |
| public | Delete | string Fix\_id | | | Void | 进入固定车主信息中，点击删除，可删除固定车主信息 | |

Temp类记录临时车主信息，具体情况如表4-3所示。

Temp类功能表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 类功能描述 | 处理临时车主的业务操作 | | | |
| 所在包名称 | DAL | | | |
| 继承对象 | 无 | | | |
| 实现对象 |  | | | |
| 类属性 | | | | |
| 保护属性 | 类型 | 名称 | 描述 | 备注 |
| Public | String | Temp\_id | 记录编号 | 长度限制20位 |
| 主要实现方法 | | | | |
| 保护属性 | 方法名 | 输入参数 | 输出参数 | 方法功能描述 |
| Public | Update | string Temp\_id | int | 进入临时车主信息中，点击编辑，可修改固定车主信息 |
| public | Delete | string Temp | Void | 进入固定车主信息中，点击删除，可删除固定车主信息 |

Role类管理角色信息，具体情况如表4-4所示。

Role类功能表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 类功能描述 | 角色管理 | | | | | |
| 所在包名称 | DAL | | | | | |
| 继承对象 | 无 | | | | | |
| 实现对象 |  | | | | | |
| 类属性 | | | | | | |
| 保护属性 | 类型 | 名称 | 描述 | | 备注 | |
| Public | String | Role\_id | 记录编号 | | 长度限制20位 | |
| 主要实现方法 | | | | | | |
| 保护属性 | 方法名 | 输入参数 | | 输出参数 | | 方法功能描述 |
| Public | Insert | string Role\_id | | int | | 增加角色 |
| public | Delete | string Role\_id | | Void | | 删除角色 |

1. 数据库的实现

根据系统需求确定系统有如下7张表。

用户表：储存用户信息。

角色表：储存角色信息。

车位管理表：管理车位信息以及车位状态。

IC卡管理表：管理IC卡信息。

临时车辆记录表：记录临时车辆的信息。

固定车主表：记录固定车主信息及出入场信息。

收费表：记录用户们的收费信息。

用户表用于记录用户的编号、姓名、电话等属性信息，如表4-5所示。

表4-5用户表

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 列名 | 数据类型 | 长度 | 小数位 | 标识 | 主键 | 允许空 | 外键 | 说明 |
| 1 | user\_id | Varchar2 | 50 | 0 | 是 | 是 | 否 |  |  |
| 2 | role\_id | Varchar2 | 50 | 0 |  |  | 否 | 是 |  |
| 3 | user\_name | Varchar2 | 50 | 0 |  |  | 否 |  |  |
| 4 | real\_name | Varchar2 | 50 | 0 |  |  | 否 |  |  |
| 5 | user\_pwd | Varchar2 | 20 | 0 |  |  | 否 |  |  |
| 6 | user\_phone | Varchar2 | 50 | 0 |  |  | 是 |  |  |

角色表储存角色的编号、名字等信息，如表4-6所示。

表4-6角色表

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 列名 | 数据类型 | 长度 | 小数位 | 标识 | 主键 | 允许空 | 外键 | 说明 |
| 1 | role\_id | Varchar2 | 50 | 0 | 是 | 是 | 否 |  |  |
| 2 | role\_name | Varchar2 | 50 | 0 |  |  | 否 |  |  |

车位管理表记录车位的编号、ID、状态等属性信息，如表4-7所示。

表4-7车位管理表

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 列名 | 数据类型 | 长度 | 小数位 | 标识 | 主键 | 允许空 | 外键 | 说明 |
| 1 | seat\_id | Varchar2 | 50 | 0 | 是 | 是 | 否 |  |  |
| 2 | seat\_num | Varchar2 | 50 | 0 |  |  | 否 |  |  |
| 3 | seat\_setion | Varchar2 | 50 | 0 |  |  | 否 |  |  |
| 4 | seat\_state | int | 11 | 0 |  |  | 否 |  |  |
| 5 | seat\_tag | Varchar2 | 50 | 0 |  |  | 是 |  |  |

IC卡表记录IC卡的ID、车位ID、姓名、车牌号等属性信息，如表4-8所示。

表4-8 IC卡表

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 列名 | 数据类型 | 长度 | 小数位 | 标识 | 主键 | 允许空 | 外键 | 说明 |
| 1 | card\_id | varchar2 | 50 | 0 | 是 | 是 | 否 |  |  |
| 2 | seat\_id | Varchar2 | 50 | 0 |  |  | 否 | 是 |  |
| 3 | user\_name | Varchar2 | 50 | 0 |  |  | 否 |  |  |
| 4 | user\_gender | Varchar2 | 1 | 0 |  |  | 否 |  |  |
| 5 | user\_addr | Varchar2 | 50 | 0 |  |  | 否 |  |  |
| 6 | car\_num | Varchar2 | 50 | 0 |  |  | 否 |  |  |

临时车辆记录表，用于记录临时车主的编号、车牌号、出入场时间等属性信息，用户如表4-9所示

表4-9临时车辆表

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 列名 | 数据类型 | 长度 | 小数位 | 标识 | 主键 | 允许空 | 外键 | 说明 |
| 1 | temp\_id | Varchar2 | 50 | 0 | 是 | 是 | 否 |  |  |
| 2 | card\_id | Varchar2 | 50 | 0 |  |  | 否 |  |  |
| 3 | car\_num | Varchar2 | 50 | 0 |  |  | 否 |  |  |
| 4 | entry\_date | date | 0 | 0 |  |  | 否 |  |  |
| 5 | entery\_time | time | 0 | 0 |  |  | 否 |  |  |
| 6 | out\_date | date | 0 | 0 |  |  | 是 |  |  |
| 7 | out\_time | time | 0 | 0 |  |  | 是 |  |  |
| 8 | temp\_money | float | 0 | 4 |  |  | 是 |  |  |

固定车主表用于记录固定固定车主的编号、IC卡编号、出入场时间等属性信息，如表4-10所示。

表4-10固定车主表

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 列名 | 数据类型 | 长度 | 小数位 | 标识 | 主键 | 允许空 | 外键 | 说明 |
| 1 | fixed\_id | Varchar2 | 50 | 0 | 是 | 是 | 否 |  |  |
| 2 | card\_id | Varchar2 | 50 | 0 |  |  | 否 | 是 |  |
| 3 | entry\_date | date | 0 | 0 |  |  | 否 |  |  |
| 4 | entery\_time | time | 0 | 0 |  |  | 否 |  |  |
| 5 | out\_date | date | 0 | 0 |  |  | 是 |  |  |
| 6 | out\_time | time | 0 | 0 |  |  | 是 |  |  |

收费表用于记录收费的编号、车主姓名、车牌号、收费金额等属性信息，如表4-11所示。

表4-11收费表

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 列名 | 数据类型 | 长度 | 小数位 | 标识 | 主键 | 允许空 | 外键 | 说明 |
| 1 | role\_id | Varchar2 | 25 | 0 | 是 | 是 | 否 |  |  |
| 2 | role\_name | Varchar2 | 255 | 0 |  |  | 是 |  |  |
| 3 | role\_type | Varchar2 | 255 | 0 |  |  | 是 |  |  |
| 4 | car\_num | Varchar2 | 25 | 0 |  |  | 是 |  |  |
| 5 | car\_money | Varchar2 | 255 | 0 |  |  | 是 |  |  |
| 6 | car\_time | Varchar2 | 255 | 0 |  |  | 是 |  |  |

1. 系统详细设计与实现
2. 用户登录

登录模块界面如图4.1所示：

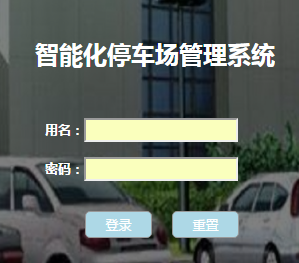


图4.1 用户登录

功能描述：登陆页面需要用户名、密码进行登录。登录的同时对输入的用户名和密码进行验证，首先我们要保证用户名和密码不能为空。其次是对数据库的验证，系统在数据库中搜索用户输入的用户名是否存在，若不存在，则提示出错。并且重新登录，系统不允许两个用户同时登录，该操作在一定程度上使系统的安全性有所提高。

登录主要代码位置：Parking/src/DAL/login.java。主要封装了对登录数据的操，该类中有三个方法分别用于检查用户登录信息是否合法，根据用户编号和角色编号获取用户名和角色信息。

publicclass Login {

publicboolean checkLogin(String user\_id,String user\_pwd)

{

String sqlCmd="select count(\*) from user where user\_id=? and user\_pwd=?";

Object[] objList=new Object[2];

objList[0]=user\_id;

objList[1]=user\_pwd;

String result=SQLUtil.excuteScalar(sqlCmd,objList).toString(); if(result.equals("1"))

{

returntrue;

}

else {

returnfalse;

}

}

public String getName(String user\_id)

{

String sqlCmd="select user\_name from user where user\_id='"+user\_id+"'";

String result=SQLUtil.excuteScalar(sqlCmd, null).toString();

return result;

}

public String getSysLevel(String user\_id)

{

String sqlCmd="select role\_id from user where user\_id='"+user\_id+"'";

String result=SQLUtil.excuteScalar(sqlCmd, null).toString();

return result;

}

}

1. 系统信息管理

添加角色信息如图4.2所示：

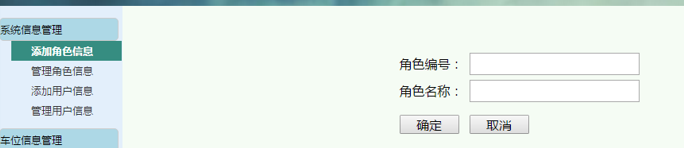


图4.2 添加角色信息

添加用户信息如图4.3所示：

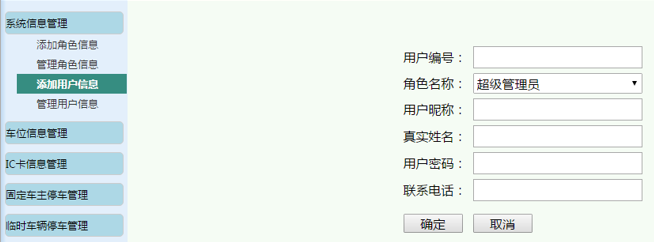


图4.3添加用户信

管理用户信息界面如图4.4所示：

图4.4管理用户信息

功能描述：该模块是对系统信息的一个管理，添加角色输入编号和名称，后台数据库进行验证，若角色没有存在则添加数据成功，跳转到管理界面。管理角色信息可以根据用户的编号，角色名称进行查询操作、编辑、和删除操作。添加用户信息输入用户编号、名称、昵称、姓名、密码电话点击确定进行后台数据库的验证，若成功添加则跳转到管理界面。管理用户界面可根据用户编号、角色名称、用户名称、真实姓名进行查询、编辑和删除操作。

1. 车位信息管理

添加停车位信息如图4.5所示：



图 4.5 添加车位信息

管理车位信息如图4.6所示：

图4.6管理车位信息页面

功能描述：该模块是对车位的一个管理。点击添加车位信息，输入编号选择是A区还是B区，添加成功则会跳转到管理界面。管理界面可根据车位ID、车位编号、所属区域、车位备注进行查询，还可对车位信息进行编辑和删除。

对车位数据的操作封装在：seat.java类中。该类中主要封装了分页和对车位信息的增、删、改、查等方法。首先获取车位信息列表，然后获取未分配的车位列表，根据查询条件获取获取分页后的信息列表，数据的更新、插入、删除，获取分页总数，根据查询条件获取分页总数。

publicclass Seat {

public List<Object> getEntity()

{

String sqlCmd="select \*from Seat";

return DBUtil.SQLUtil.executeQuery(sqlCmd, null);

}

public List<Object> getNoUseSeat()

{

String sqlCmd="SELECT \*FROM Seat WHERE seat\_id NOT IN(SELECT seat\_id FROM card)";

return DBUtil.SQLUtil.executeQuery(sqlCmd, null);

}

public List<Object> getEntity(int page)

{

int size=(page-1)\*15;

String sqlCmd="select \*from Seat limit "+size+",15";

return DBUtil.SQLUtil.executeQuery(sqlCmd, null);

}

public List<Object> getEntityByWhere(String sqlWhere,int page)

{

int size=(page-1)\*15;

String sqlCmd="select \*from Seat where "+sqlWhere+" limit "+ size+",15";

return DBUtil.SQLUtil.executeQuery(sqlCmd, null);

}

publicint deleteEntity(String seat\_id)

{

String sqlCmd="delete from Seat where seat\_id='"+seat\_id+"'";

return DBUtil.SQLUtil.executeNonQuery(sqlCmd, null);

}

public List<Object> getEntityById(String seat\_id)

{

String sqlCmd="select \*From Seat where seat\_id='"+seat\_id+"'";

return DBUtil.SQLUtil.executeQuery(sqlCmd, null);

}

publicint updateEntity(String seat\_id,String seat\_num,String seat\_section,String seat\_state,String seat\_tag)

{

String sqlCmd="Update Seat set seat\_num='" + seat\_num + "',seat\_section='" + seat\_section + "',seat\_state='" + seat\_state + "',seat\_tag='" + seat\_tag + "' where seat\_id='"+seat\_id+"'";

return SQLUtil.executeNonQuery(sqlCmd, null);

}

publicint insertEntity(String seat\_id,String seat\_num,String seat\_section,String seat\_state,String seat\_tag)

{

String sqlCmd="Insert into Seat values('" + seat\_id + "','" + seat\_num + "','" + seat\_section + "','" + seat\_state + "','"+seat\_tag+"')";

return SQLUtil.executeNonQuery(sqlCmd, null);

}

publicboolean checkExist(String seat\_id)

{

String sqlCmd="select count(\*) from Seat where seat\_id='"+seat\_id+"'";

if(1==Integer.parseInt(SQLUtil.excuteScalar(sqlCmd, null).toString()) )

{

returntrue; }

returnfalse;

}

public Object getPageCount()

{

String sqlCmd="SELECT CEIL( COUNT(\*)/15.0) FROM Seat ";

return SQLUtil.excuteScalar(sqlCmd, null);

}

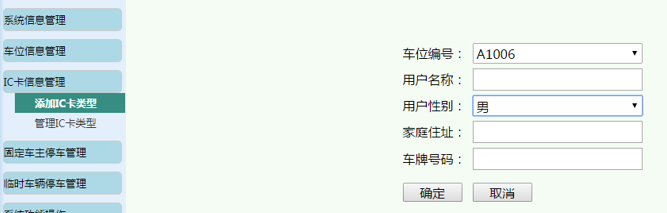
public Object getPageCountByWhere(String sqlWhere)

{

String sqlCmd="SELECT CEIL( COUNT(\*)/15.0) FROM Seat where "+sqlWhere;

return SQLUtil.excuteScalar(sqlCmd, null);

}}

1.  IC卡管理

添加IC卡界面如图4.7所示：

图4.7 添加IC卡界面

管理IC卡信息界面如图4.8所示：



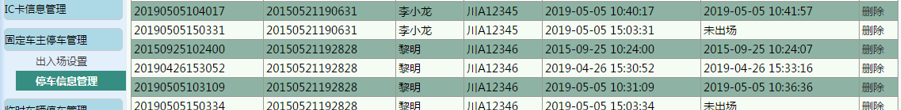
图4.8 管理IC卡界面

功能描述：该功能模块是对IC卡信息的管理，点击添加IC卡输入如图5.11所示的信息，添加成功后，跳转到图5.12的界面。管理界面可根据IC卡编号、车位编号、用户名称、车牌号码进行查询。也可进行相应的编辑和删除操作。

1. 固定车主管理

固定车主出入场设置界面如图4.9所示：



图4.9 固定车主出入场设置界面

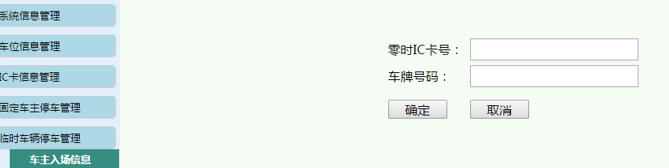
停车信息管理界面如图4.10所示：

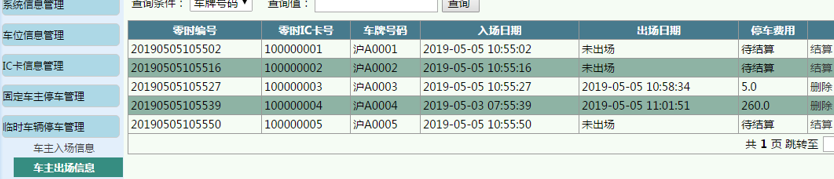
图4.10管理停车场信息界面

功能描述：该模块是对固定车主的出入场设置以及停车信息进行管理。如图5.11所示，对车主进行入场设置，点击确定后就会在下面的表格中显示，点击停车信息管理按钮，就会出现如图4.10所示。设置出场，出场时间就会出现在停车信息表中。

1. 临时车主管理

临时车主入场信息界面如图4.11所示：



图4.11临时车主入场页面

车主出场信息界面如图4.12所示：

图4.12临时车主出场页面

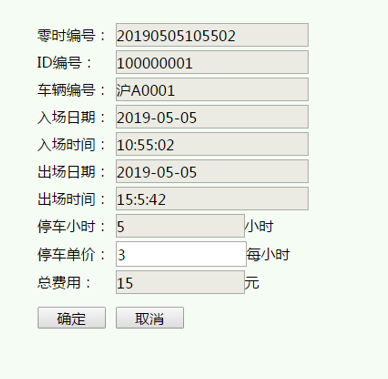
结算页面如图4.13所示

图4.13 结算页面

功能描述：该模块是对临时车主出入场的管理。点击入场管理，出现如图4.11所示的界面，输入信息，如果输入成功就会跳转到如图4.12的界面，出场日期，日期显示未出场，停车费用显示待结算。点击结算出现如图4.13界面。

4.7 系统功能操作

密码修改、退出系统界面如图4.15所示：

5.23 系统功能操作界面

功能描述：该模块是对系统功能的一个操作，可进行修改密码。点击退出系统则会跳转到如图4.1的登录界面。

1. 系统测试

停车场管理系统设计完之后需要进行一系列的测试，调试过程中内部环境和外界因素的变化，会影响系统的运行和操作，当系统适应这些变化后，慢慢趋于完善，达到预期想要的结果，这就是系统测试从头到尾一直要做的工作。

根据本系统的功能情况，以黑盒测试为主，白盒测试为辅。黑盒测试：将项目看成一个黑盒子，在不去考虑项目其他情况、只知道系统功能进行的测试。它的任务就是检测系统的每个功能是否可以正常运行以及操作结果是否正确。白盒测试：与黑盒测试相反，是将项目看成一个透明的白盒子，要求操作员必须知道项目流程、项目代码，按照规格说明书的规定检测功能是否符合要求，对操作员要求较高。

1. 用户登录

表 5-1 用户登录

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 功能测试 | | | | | | |
| 概述 | | | | | | |
| 测试编号 | | | ZL001 | | | |
| 功能描述 | | | 用户登录 | | | |
| 功能URL | | |  | | | |
| 用例目的 | | | 测试用户是否能登录成功 | | | |
| 前提条件 | | | 进入用户登录界面 | | | |
| 测试操作 | | | | | | |
| 编号 | 输入/动作 | 期望的输出响应 | | 实际情况 | 是否正确 | 错误编号 |
| 1 | 输入不存在的用户名，不填写密码 | 系统提示用户名或密码错误 | | 系统提示登录名或密码错误 | 正确 |  |
| 2 | 输入正确的用户名输入错误的密码 | 系统提示用户名或密码错误 | | 系统提示登录名或密码错误 | 正确 |  |
| 3 | 输入错误的用户名输入正确的密码 | 系统提示用户名或密码错误 | | 系统提示登录名或密码错误 | 正确 |  |
| 4 | 不填写用户名及密码 | 系统提示用户名或密码错误 | | 系统提示用户名或密码错误 | 正确 |  |

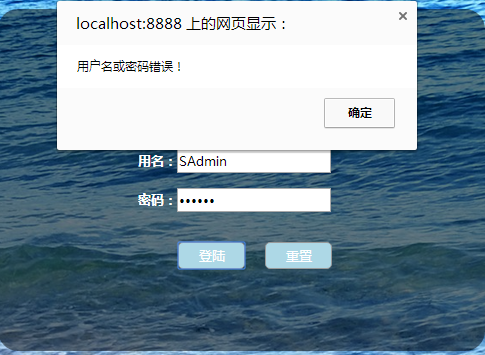
图5.2 登录密码错误功能

图5.3 登录账号错误功能图



1. 添加临时车主

表 5-2 用户登录

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 功能测试 | | | | | | |
| 概述 | | | | | | |
| 测试编号 | | | ZL002 | | | |
| 功能描述 | | | 添加临时车主 | | | |
| 功能URL | | |  | | | |
| 用例目的 | | | 测试添加临时车主是否能成功 | | | |
| 前提条件 | | | 进入添加临时车主界面 | | | |
| 测试操作 | | | | | | |
| 编号 | 输入/动作 | 期望的输出响应 | | 实际情况 | 是否正确 | 错误编号 |
| 1 | 不输入临时IC卡及车牌号 | 临时IC卡号或者车牌号不能为空 | | 临时IC卡号或者车牌号不能为空 | 正确 |  |
| 2 | 不输入临时IC卡卡号 | 临时IC卡号或者车牌号不能为空 | | 临时IC卡号或者车牌号不能为空 | 正确 |  |
| 3 | 输入的临时卡号重复 | 主键重复，数据添加失败！ | | 主键重复，数据添加失败！ | 正确 |  |

1. 添加角色信息

表 5-3 角色信息添加

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 功能测试 | | | | | | |
| 概述 | | | | | | |
| 测试编号 | | | ZL003 | | | |
| 功能描述 | | | 添加角色信息 | | | |
| 功能URL | | |  | | | |
| 用例目的 | | | 测试添加角色信息 | | | |
| 前提条件 | | | 进入添加角色信息界面 | | | |
| 测试操作 | | | | | | |
| 编号 | 输入/动作 | 期望的输出响应 | | 实际情况 | 是否正确 | 错误编号 |
| 1 | 不输入角色编号 | 请填写此字段。 | | 请填写此字段。 | 正确 |  |
| 2 | 不输入角色名称 | 请填写此字段。 | | 请填写此字段。 | 正确 |  |
| 3 | 输入重复的角色编号 | 主键重复，数据添加失败！ | | 主键重复，数据添加失败！ | 正确 |  |

图5.4 添加角色信息编号或者角色名称为空图

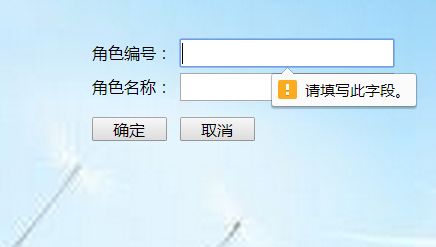
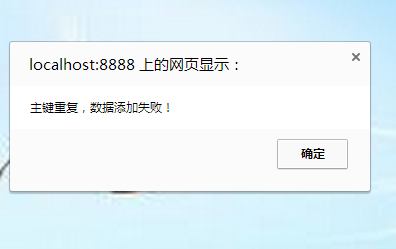


图5.5添加角色信息编号重复图



1. 总结

本系统采用MVC+J2EE+MYSQL实现。JSP用于页面的设计，Java用来处理后台跳转、对数据库操作,实现对数据库的增、删、改等操作。使用是Tomcat8.0作为WEB服务器，提供适应性强的Internet服务器功能，具有很高的执行效率。本次停车场管理系统能够满足用户的基本需求，方便快捷，既节省了时间，提高了效率，又节约了开发成本，给人们生活带来了极大的改善。通过本次课程设计我更加熟练的掌握了java编程，B/S交互以及jQuery使用。同时由于技术问题和知识面的局限，项目功能不是很完善，数据安全性和网页兼容性以及效率还有待提高。争取在以后的学习中进一步将其完善，做得更加完美。且将之前所学的东西可以更好地融合。在今后的日子里，我要将学无止境、不耻下问、敢于创新、坚持不懈的态度继续保持下去，做一个不让自己后悔的人。

# 参考文献

[1] 李平. 基于JSP的动态网页开发技术[J]. 微计算机信息, 2009, 25(21):108-110.

[2] 王珊, 萨师煊. 数据库系统概论[M]. 北京：高等教育出版社, 2007. 6-38

[3] 范新民. 基于JSP、JDBC技术动态访问Web数据库的实现方案[J]. 福建师范大学学报:自然科学版, 2002, 18(2):28-33.

[4] 曹春萍, 王志民. MVC设计模式的研究及其应用[J]. 现代电子技术, 2005, 28(20):80-82.

[5] 徐晓霞, 贝雨馨. B/S模式与C/S模式之比较[J]. 延边大学学报:自然科学版, 2002, 28(2):126-129.

[6]丁宝康：数据库实用教程 ，清华大学出版社，2004. 26-75.

[7]冯博琴等；面向对象分析与设计 ， 机械工业出版社，2004. 18-25.

[8]刘文利 .国内停车场管理系统的现状与发展趋势[J] . 中国新技术新产品，2011(01):20-22.

[9]吕英丽. 停车场车辆出入管理系统的研究[J].北京邮电工程,电子信息工程.2011(02).20-40