**12306余票查询系统**

**系统详细设计说明书**

**终稿**

**班级： 计科1602班**

**姓名： 麻锦涛**

**学号： 16281262**

目录

[系统详细设计说明书 4](#_Toc502444428)

[1. 系统功能概述 4](#_Toc502444429)

[1.1 用户模块 4](#_Toc502444430)

[1.2 管理员模块 6](#_Toc502444431)

[2. 系统功能模块结构 8](#_Toc502444432)

[2.1 前端功能 8](#_Toc502444433)

[2.2 WEB服务器 9](#_Toc502444434)

[2.3 数据库端 9](#_Toc502444435)

[3.系统界面设计说明 10](#_Toc502444436)

[3.1 用户登录界面 10](#_Toc502444437)

[3.2 余票查询界面 11](#_Toc502444438)

[3.3 管理员登录界面 12](#_Toc502444439)

[3.4 管理员操作界面 12](#_Toc502444440)

[4.系统物理模型说明 13](#_Toc502444441)

[4.1 表 13](#_Toc502444442)

[4.2 视图 16](#_Toc502444443)

[4.3 索引 17](#_Toc502444444)

[4.4 存储过程 17](#_Toc502444445)

[4.5 触发器 18](#_Toc502444446)

[5. 系统安全体系设计 19](#_Toc502444447)

[5.1 用户管理与控制 19](#_Toc502444448)

[5.2 数据存储与恢复 20](#_Toc502444449)

[6. 系统运行环境设计与部署结构 20](#_Toc502444450)

[6.1 开发环境 20](#_Toc502444451)

[6.2 数据库系统 20](#_Toc502444452)

[6.3 服务器 20](#_Toc502444453)

[6.4 开发平台 20](#_Toc502444454)

[6.5 运行平台 20](#_Toc502444455)

[7. 源代码列表及说明 21](#_Toc502444456)

[7.1源代码列表 21](#_Toc502444457)

[7.2 文件说明 23](#_Toc502444458)

# 系统详细设计说明书

## 1. 系统功能概述

### 1.1 用户模块

#### 1.1.1 用户登录功能

操作：用户在登录界面输入用户名和密码，点击登录，即可跳转至登录界面，执行接下来的操作。

说明：必须输入匹配的用户名和密码方能登录。

若匹配不正确，系统会提示登录失败，此时用户需要重新登录。

同时，一个账户不能在同一台主机上登录两次，若一次登录之后未退出登录，又尝试再次登录，系统会提示：你的账户已登录，需要退出之后重新登录。

#### 1.1.2 用户注册功能

操作：若你是未注册的用户，则可在登录窗口点击进行注册，输入相应的信息之后即可完成注册功能。

#### 1.1.3 修改密码和个人信息

操作：若你要修改密码或者个人信息，则点击修改信息的链接即可执行修改。

说明：若你是已注册用户，忘记了密码，也可通过修改密码的方式重新设置登录密码，修改成功之后方可重新登录。

#### 1.1.4 查看车次信息

操作：用户登录成功后，在余票查询界面选择出发地、到达地、出发时间，点击查询按钮，系统则会返回满足条件的车次信息。

说明：用户只有在成功登录后，才能执行此项操作，若未登录，则点击查询会跳转至登录界面。

#### 1.1.5 查看账户信息

操作：用户登录成功后，在主界面点击查看账户信息的链接，则可进行查看；

说明：用户只有在成功登录后，才能执行此项操作，若未登录，则点击查询会跳转至登录界面。

#### 1.1.6 查看订单详情

操作：用户登录成功后，在主界面点击查看订单详情的链接，则可进行查看；

说明：用户只有在成功登录后，才能执行此项操作，若未登录，则点击查询会跳转至登录界面。

#### 1.1.7 买票功能

操作：用户在成功登录后，可以点击购票按钮进行购票，输入车次信息以及购票张数，即可完成购票功能。在购票过程中，要执行以下操作：

①调用随机函数生成一个订单号；

②根据用户的输入，向系统插入一条订购信息；

③执行更新操作，相应车次减去相应的票数；

④执行更新操作，买家账户余额减去相应的金额；

⑤生成一条订单详情，插入到用户订购表中，同时显示给用户。

说明：用户只有在成功登录后，才能执行此项操作，若未登录，则点击查询会跳转至登录界面。

#### 1.1.8 退票功能

操作：用户在成功登录后，可以执行退票操作，用户选择要退的订单，点击退票，即可执行退票操作。在退票过程中，要执行如下操作：

①执行更新操作，相应车次加上相应的票数；

②执行更新操作，用户账户余额加上相应的金额；

③删除之前的订单信息，同时删除与该订单相关的所有信息；

④提示退票成功。

说明：用户只有在成功登录后，才能执行此项操作，若未登录，则点击查询会跳转至登录界面。

### 1.2 管理员模块

#### 1.2.1 管理员登录功能

操作：输入匹配的用户名和密码，管理员即可进入管理员操作界面。

说明：由于管理员界面权限较高，只有特定权限的用户（管理员）方可登录，一般用户无法登录。

#### 1.2.2 增删改查基本表

操作：管理员在进入管理员操作界面后，可以对基本表进行增删改查操作。

具体如下：

①对账户表(account)进行操作

增添：管理员可向account表插入新的元组；

删除：管理员可删除account表存在的元组；

修改：管理员可以修改account表的各个属性的值；

查询：管理员可以对account执行查询操作，查看account表的内容；

②对用户表（User）进行操作

增添：管理员可向user表插入新的元组；

删除：管理员可删除user表中存在的元组；

修改：管理员可以修改user表的各个属性的值；

查询：管理员可以对user表执行查询操作，查看user表的内容；

③对订单表（booking）进行操作

增添：管理员可向booking表插入新的元组；

删除：管理员可删除booking表中存在的元组；

修改：管理员可以修改booking表的各个属性的值；

查询：管理员可以对booking表执行查询操作，查看booking表的内容；

④对列车表（train）进行操作

增添：管理员可向train表插入新的元组；

删除：管理员可删除train表中存在的元组；

修改：管理员可以修改train表的各个属性的值；

查询：管理员可以对train表执行查询操作，查看train表的内容；

⑤对票务表（information）进行操作

增添：管理员可向information表插入新的元组；

删除：管理员可删除information表中存在的元组；

修改：管理员可以修改information表的各个属性的值；

查询：管理员可以对information表执行查询操作，查看user表的内容；

⑥对映射表（map）进行操作

增添：管理员可向map表插入新的元组；

删除：管理员可删除map表中存在的元组；

修改：管理员可以修改map表的各个属性的值；

查询：管理员可以对map表执行查询操作，查看map表的内容；

说明：只有当管理员成功登录之后，管理员才能执行上述操作，否则，点击这些链接将会跳转到登录界面。

#### 1.2.3 修改登录密码

操作：管理员点击修改密码，即可修改登录密码。

说明：只有当管理员成功登录之后，管理员才能执行此操作，否则，点击链接将会跳转到登录界面。

#### 1.2.4 权限控制功能

操作：管理员可以执行授权和撤销权限的操作，权限有：{查询表，删除表，修改表，添加表}。管理员可以控制用户的权限。

说明：权限控制一般在数据库端实现。

#### 1.2.5 实时备份功能

操作：管理员可以实现对数据的转储和备份，并能在系统发生故障时，及时地恢复数据，让系统正常运作。

说明：要求管理员有很强的专业素养，需要经过专业的培训才能达到。

## 2. 系统功能模块结构

### 2.1 前端功能

该系统采用的是B/S架构，系统前端是采用jsp实现的，后台通过servlet处理用户请求。

无论是用户模块，还是管理员模块，所有操作的输入数据都是在前端完成的。具体如下：

用户模块

①用户登录

用户在前端界面输入用户名、密码，后台表单获取用户输入数据，通过查询账户表获取结果集，然后把用户输入数据与查询结果一一对比，若匹配，则跳转至余票查询界面；

用户注册、修改密码也是需要通过前端输入数据，交由后台处理。这些过程与用户登录过程类似。

②买票功能、卖票功能

用户需要在前端输入出发地、到达地、出发时间，才能实现查询；若要实现买票，也需要输入车次、购买张数等信息，这些信息的输入都是在前端完成的。

卖票功能与买票功能类似。

③查看订单信息、账户信息

当用户查看订单信息、账户信息时，此时前端需要提供一个超链接，供用户跳转。

管理员模块

①管理员登录

管理员在前端界面输入账号和密码，后台执行一定操作后方可实现跳转；

②实现对基本表的增删改查

管理员执行增删改查操作，需要在前端点击特定的链接或者输入特定的内容，方可执行成功。

简言之，前端主要是给用户提供一个操作界面，提供一些简单的组件（链接、按钮、文本框、下拉列表等）以便用户执行相应的操作，而具体的实现则在后台服务器端完成，这样不仅可以把功能层次化划分，而且可以更好地进行系统的维护。

### 2.2 WEB服务器

Web服务器也就是我们常说的“后台”，用户的业务基本上是在这个层次完成的。Web服务器扮演着桥梁的作用，“数据库端”和“前端”的交互，需要web服务器端来完成。在本工程中，采用Tomcat v7.0 作为web服务器。

在本系统中，web服务器主要实现如下功能：

①用户登录、管理员登录

用户在前端输入用户名和密码，交由后台的servlet进行处理，此时需要进行连接数据库操作、执行sql语句，以及处理返回的结果集的操作，这些，都需要web服务器来完成。

②用户查询余票

当用户在余票查询界面输入条件后，web服务器此时便会对用户请求发出响应，在后台，同样需要执行连接数据库操作、执行sql语句的操作，以及把查询结果重新返回至前端的操作，这些，也是web服务器完成的。

③增删改查

管理员的增删改查操作，都是需要交由web服务器进行处理。

至于本工程的web端到底需要做哪些处理，详情请见第7部分-源代码列表及说明。

### 2.3 数据库端

数据库端是用来保存基本表、视图、存储过程、触发器的地方，所有的用户数据都是通过数据库来保存的，同时，一切业务的开展都是通过数据库展开的。

在本工程中，数据库端是后台的后台，用户所要的数据最终都来源于数据库端，连上数据库的操作，是在web服务器端完成的，只有连上数据库，用户才能访问到数据。但是，由于操作数据库需要一定的权限，一般用户只有操作数据库的一些权限，那些更高的权限只有数据库管理员才拥有。

在本项目中，需要进行良好的封装，用户能看到的只能是视图层面的，而不能直接操纵基本表；只有数据库管理员才有权限直接对基本表进行操作。

## 3.系统界面设计说明

### 3.1 用户登录界面

用户登录界面主要实现的是用户登录，需要一定的控制机制才能保证用户成功登录，以及登录成功后实现对用户登录状态的维护，以保证用户的一系列操作是合法的。

用户登录界面如下图所示：



用户登录界面如上图所示，中间文本框部分就是用户登录时要输入的文本框和密码，用户只有输入正确的用户名和密码，同时，没有在其它终端登录，才能在端系统成功登录。用户输入用户名和密码，点击登录，即可跳转至余票查询界面进行接下来的操作。

“重置”按钮，点击之后，会把文本框的内容清空。

“用户注册”链接，是供那些未注册的用户使用的，这些用户只有通过此按钮进行成功注册后，方能进入余票查询系统。

“修改密码”链接，是供那些已注册的用户使用的，只有成功输入旧密码，才能实现修改密码的功能，若用户忘记密码，必须重新注册。

“关于我们”链接，是系统提供给用户的系统使用指导书，方便用户了解并使用此系统。

“网站声明”链接，向用户介绍了网站详情以及法律权限等。

### 3.2 余票查询界面

余票查询界面是用户模块的核心界面，用户的查票、买票、退票操作都是通过此操作界面来完成的。用户只有成功登录，才能进入此界面；若用户未登录便想访问此网页，则会跳转至登录界面。

余票查询界面如下所示：



由于时间仓促，以及数据库的限制，只实现了部分按钮的功能，现说明如下：

①“退出登录”按钮：用户点此按钮即可退出，并跳转至登录界面；

②“出发地”、“到达地”、“出发日”按钮，用户选择输入之后，点击“查询”按钮，即可查出满足条件的车次信息；

③“车票预订”按钮：点击按钮，用户即可进行买票操作；

④在“我的12306”栏，是个下拉列表，点击“查看个人信息”，即可把用户的信



息详情显示出来；点击“已完成的订单（改/退）”，就会显示用户的历史订单；点击“账户安全”，即可查看自己的账户详情，查看密码以及登录余额等。

### 3.3 管理员登录界面

管理员登录界面，是管理员进入管理系统的通行证，管理员密码的安全级别必须要高。只有当管理员输入正确的账户名和密码时才能完成登录。

管理员登录界面如下所示：



“用户名”和“密码”框，以供用户输入，若输入信息与数据库中用户信息匹配，点击登录，即可跳转至管理员操作界面；

“重置按钮”，点击此按钮，会将文本框内容置空。

### 3.4 管理员操作界面

管理员操作界面，是管理员实现数据管理、安全管理、权限管理的核心平台。

在此界面，管理员可以实现如下功能：

①对账户表进行增删改查

②对用户表进行增删改查

③对订单表进行增删改查

④对车次表进行增删改查

⑤对票务表进行增删改查

⑥修改管理密码

⑦给用户授权、撤销权限操作

⑧实时备份

管理员操作界面如下所示：

## 4.系统物理模型说明

### 4.1 表

|- 账户表(account表)

建表语句如下所示：

|  |
| --- |
| #创建账户表  CREATE TABLE account(  id bigint,  password varchar(20),  money int,  flag varchar(20),  PRIMARY KEY(id),  FOREIGN KEY(id) REFERENCES user(id))ENGINE=InnoDB; |

说明：此表有4个基本属性（用户名、密码、账户余额、登录状态）

用户名：既是主键也是user表的外键，通过此属性可与user表相连；

登录状态：若flag=true，则表示用户成功登录；若flag=false，则表示用户已退出登录状态。

|- 用户表(user表)

建表语句如下所示：

|  |
| --- |
| #创建用户表  CREATE TABLE user (  id bigint PRIMARY KEY,  name varchar(50) NOT NULL,  gender enum('男','女') NOT NULL,  idcode varchar(100)NOT NULL,  telephone bigint NOT NULL,  mail varchar(100) )ENGINE=InnoDB; |

说明：此表有6个属性（用户名、用户姓名、性别、身份证号、手机号、邮箱）

用户名：该表的主键。

|- 订票表(booking表)

建表语句如下所示：

|  |
| --- |
| #创建订购表  CREATE TABLE booking(  no varchar(50),  id bigint NOT NULL,  bookingtime timestamp,  PRIMARY KEY(no),  FOREIGN KEY(id) REFERENCES user(id))ENGINE=InnoDB; |

说明：该表有3个属性（订单号、用户名、订票时间）

订单号：既是该表的主键，也是外键，通过此属性与用户表相连。

|- 映射表(map表)

建表语句如下所示：

|  |
| --- |
| #创建对应表  CREATE TABLE map (  no varchar(50),  trips varchar(50),  primary key(no))ENGINE=InnoDB; |

说明：该表只有2个属性（订单号、车次）

订单号：主键，通过该属性可以与订单表、车次表相连。

|- 车次表（train表）

建表语句如下所示：

|  |
| --- |
| #创建车次信息表  CREATE TABLE train (  trips varchar(50) PRIMARY KEY,  originating\_station varchar(50),  arrival\_station varchar(50),  departure\_time time,  arrival\_time time,  driven\_distance int,  driven\_time time,  route text)ENGINE=InnoDB; |

说明：该表有8个属性（车次、出发站、到达站、出发时间、到达时间、行车距离、行车时间、行车路线）

车次：为主键

|- 票务表（information表）

建表语句如下所示：

|  |
| --- |
| #创建订票信息表  CREATE TABLE information (  trips varchar(50),  type varchar(50),  price int,  amount int,  PRIMARY KEY(trips,type),  FOREIGN KEY(trips) REFERENCES train(trips))ENGINE=InnoDB; |

说明：该表有4个属性（车次、座位类型、价格、数量）

主键：（trips，type）

外键：trips，通过此属性可以与train表相连接。

|- 管理员账户表（admin表）

建表语句如下所示：

|  |
| --- |
| #创建管理员账户表  CREATE TABLE admin(  id bigint,  password varchar(20),  PRIMARY KEY(id))ENGINE=InnoDB; |

说明：该表有2个属性（管理员用户名、登录密码）

主键：用户名（id）

### 4.2 视图

处于安全考虑，需要给常见的用户查询建立视图，如：

行列子集视图view1:

|  |
| --- |
| #建立行列子集视图view1（查询用户10000的所有基本信息）  create view view1  as  select a.id,a.password,u.name,u.gender,u.native\_place,u.idcode,u.telephone,u.mail,a.money  from account a,user u  where a.id=u.id and a.id=10000; |

行列子集视图view2:

|  |
| --- |
| #建立行列子集视图view2（查询用户10000的订单详情）  create view view2  as  select u.id,u.name,b.no,b.bookingtime  from user u,booking b  where u.id=b.id and u.id=10000; |

行列子集视图view3:

|  |
| --- |
| #建立行列子集视图view3(查询用户10000的订单号为A-0的车次详情)  create view view3  as  select u.id,u.name,b.no,m.trips,t.originating\_station,t.arrival\_station,t.departure\_time,t.arrival\_time,t.driven\_distance,t.route  from user u,booking b,map m,train t  where u.id=b.id and b.no=m.no and m.trips=t.trips and u.id=10000 and b.no='A-0'; |

行列子集视图view4:

|  |
| --- |
| #建立行列子集视图view4（查询车次为G1，座位类型为商务座的所有信息）  create view view4  as  select t.trips,i.type,t.departure\_time,i.price,i.amount  from train t,information i  where t.trips=i.trips and t.trips='G1' and i.type='商务座'; |

在实际应用中，根据特殊的需求可以添加不同的视图。

### 4.3 索引

处于系统优化的考虑，可以考虑给基本表建立索引，以提高查询效率。

给基本表建立索引如下所示：

|  |
| --- |
| #给account表建立唯一索引  create unique index account\_index on account(id);  #给user表建立唯一索引  create unique index user\_index on user(id);  #给booking表建立唯一索引  create unique index booking\_index on booking(no);  #给train表建立唯一索引  create unique index train\_index on train(trips);  #给information表建立唯一索引  create unique index information\_index on information(trips,type); |

### 4.4 存储过程

存储过程的设计，也是加快查询效率的方式之一。一般来说，可以给所有查询、增加、删除、修改操作建立存储过程。下面只给一些常用的操作建立存储过程：

存储过程select1:

|  |
| --- |
| #创建多表查询的存储过程 select1:连接查询账号和密码  delimiter //  create procedure select1(newid varchar(10))  begin  SELECT \*  FROM (SELECT a.id,a.password,u.name,u.gender,u.native\_place,u.idcode,u.telephone,u.mail  FROM account a,user u  WHERE a.id=u.id) t  WHERE t.id=newid;  end; // |

存储过程insert5:

|  |
| --- |
| #创建存储过程insert5:插入车次信息  delimiter //  create procedure insert5(trips varchar(10),originating\_station varchar(20),arrival\_station varchar(20),departure\_time varchar(20),arrival\_time varchar(20),driven\_distance int,driven\_time varchar(50),route varchar(200))  begin  insert into train values(trips,originating\_station,arrival\_station,departure\_time,arrival\_time,driven\_distance,driven\_time,route);  end;// |

在实际应用中，根据特殊的需求可以添加不同的存储过程。

### 4.5 触发器

触发器，是完整性控制的重要手段。此系统，给一些特定的操作建立触发器，以达到完整性控制目的和安全控制的目的。

触发器1：

|  |
| --- |
| #创建触发器(删除父表user表之前，必须先删除子表account表和booking表)  DROP TRIGGER IF EXISTS t\_afterdelete\_on\_tab1;  delimiter //  CREATE TRIGGER t\_afterdelete\_on\_tab1  BEFORE DELETE ON user  FOR EACH ROW  BEGIN  delete from account where id=old.id;  delete from booking where id=old.id;  END;// |

触发器2：

|  |
| --- |
| #创建触发器(删除父表train表之前，必须先删除子表information表)  DROP TRIGGER IF EXISTS t\_afterdelete\_on\_tab2;  delimiter //  CREATE TRIGGER t\_afterdelete\_on\_tab2  BEFORE DELETE ON train  FOR EACH ROW  BEGIN  delete from information where trips=old.trips;  END;// |

在实际应用中，根据特殊的需求可以添加不同的触发器。

## 5. 系统安全体系设计

### 5.1 用户管理与控制

此项目的用户管理与控制，可以采用session机制或者cookie机制来实现。

Session是服务器端使用的一种记录客户端状态的机制，使用上比Cookie简单一些，相应的也增加了服务器的存储压力。

#### 5.1.1 session介绍

Session是另一种记录客户状态的机制，不同的是Cookie保存在客户端浏览器中，而Session保存在服务器上。客户端浏览器访问服务器的时候，服务器把客户端信息以某种形式记录在服务器上。这就是Session。客户端浏览器再次访问时只需要从该Session中查找该客户的状态就可以了。

如果说Cookie机制是通过检查客户身上的“通行证”来确定客户身份的话，那么Session机制就是通过检查服务器上的“客户明细表”来确认客户身份。Session相当于程序在服务器上建立的一份客户档案，客户来访的时候只需要查询客户档案表就可以了。

#### 5.1.2 用session实现用户登录

Session对应的类为javax.servlet.http.HttpSession类。每个来访者对应一个Session对象，所有该客户的状态信息都保存在这个Session对象里。Session对象是在客户端第一次请求服务器的时候创建的。

Session也是一种key-value的属性对，通过getAttribute(Stringkey) 和setAttribute(String key，Objectvalue)方法读写客户状态信息。Servlet里通过request.getSession()方法获取该客户的Session，

例如：

HttpSession session = request.getSession(); // 获取Session对象

session.setAttribute("loginTime", new Date()); // 设置Session中的属性

out.println("登录时间为：" +(Date)session.getAttribute("loginTime")); // 获取Session属性

request还可以使用getSession(boolean create)来获取Session。区别是如果该客户的Session不存在，request.getSession()方法会返回null，而getSession(true)会先创建Session再将Session返回。

Servlet中必须使用request来编程式获取HttpSession对象，而JSP中内置了Session隐藏对象，可以直接使用。如果使用声明了<%@page session="false" %>，则Session隐藏对象不可用。

### 5.2 数据存储与恢复

数据备份起着很重要的作用，当系统发生故障时，可以通过转储的数据与日志记录进行数据恢复，以保证系统再次正常运作。

在本项目中，可以借助服务器进行实时备份。

第一步：提前写好备份文件db\_bak.bat（详情参照db\_bak.bat文件）；

第二步：在计算机管理中，添加任务，在特定的时间执行此备份文件，即可实行备份。

## 6. 系统运行环境设计与部署结构

### 6.1 开发环境

此项目采用windows系统进行开发，操作系统版本推荐window7以上系统

### 6.2 数据库系统

此项目采用mysql5.18数据库软件进行开发

### 6.3 服务器

此项目采用tomcat v7.0作为web服务器

### 6.4 开发平台

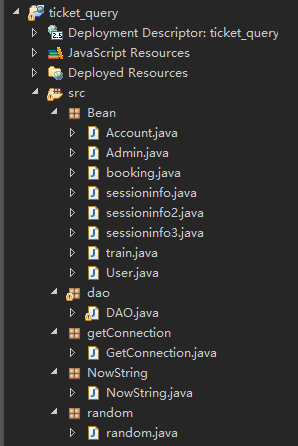
此项目采用myeclipse进行开发，JDK推荐使用jdk1.8以上版本

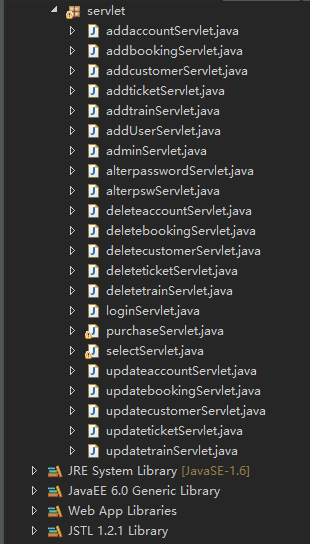
### 6.5 运行平台

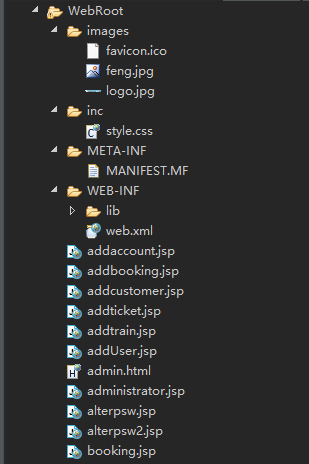
此项目可以运行在不同系统之上，推荐使用Linux系统

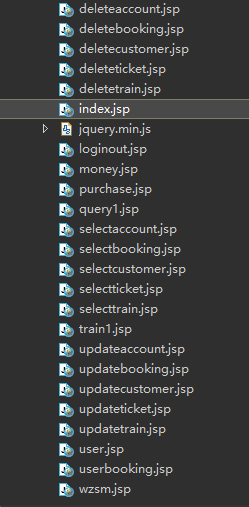
## 7. 源代码列表及说明

### 7.1源代码列表









### 7.2 文件说明

#### 7.2.1 Bean包

Account.java: 用来获取和设置account表的属性值

booking.java: 用来获取和设置booking表的属性值

train.java: 用来获取和设置train表的属性值

User.java: 用来获取和设置user表的属性值

Admin.java:用来获取和设置admin表的属性值

sessioninfo.java: 用来保存用户的登录信息

sessioninfo2.java: 用来保存管理员的登录信息

sessioninfo3.java: 用来保存余票查询界面的用户输入信息

#### 7.2.2 dao包

DAO.java: 基本的查询、修改、更新、删除的sql语句均在此类中实现

#### 7.2.3 getConnection包

GetConnection.java: 实现数据库的连接

#### 7.2.4 NowString包

NowString.java: 用于获取系统的当前时间

#### 7.2.5 random包

random.java: 随机产生长度为3的字符串，作为用户买票时的订单号

#### 7.2.6 servlet包

loginServlet.java: 用户登录的处理表单

adminServlet.java: 管理员登录的处理表单

selectServlet.java: 用户查票的处理表单

purchaseServlet.java: 用户买票的处理买单

alterpswServlet.java: 用户修改密码的处理表单

alterpasswordServlet.java: 管理员修改密码的处理表单

addaccountServlet.java: 添加账户信息的处理表单

addbookingServlet.java: 添加订购信息的处理表单

addUserServlet.java: 用户注册的处理表单

addcustomerServlet.java: 添加用户信息的处理表单

addticketServlet.java: 添加票务信息的处理表单

addtrainServlet.java: 添加车次信息的处理表单

deleteaccountServlet.java: 删除账户信息的处理表单

deletebookingServlet.java: 删除订单信息的处理表单

deletecustomerServlet.java: 删除用户信息的处理表单

deleteticketServlet.java: 删除票务信息的处理表单

deletetrainServlet.java: 删除车次信息的处理表单

updateaccountServlet.java: 修改账户信息的处理表单

updatecustomerServlet.java: 修改用户信息的处理表单

updatebookingServlet.java: 修改订购信息的处理表单

updateticketServlet.java: 修改票务信息的处理表单

updatetrainServlet.java: 修改车次信息的处理表单

#### 7.2.7 jsp文件

addaccount.jsp: 用于添加用户账户

addbooking.jsp: 用于添加订购信息

addcustomer.jsp: 用于添加用户信息

addticket.jsp: 用于添加票务信息

addtrain.jsp: 用于添加车次信息

addUser.jsp: 用于用户注册

admin.html: 管理员操作界面

administrator.jsp: 管理员登录界面

alterpsw.jsp: 用于用户修改密码

alterpsw2.jsp: 用于管理员修改密码

booking.jsp: 显示用户订单详情

deleteaccount.jsp: 用于删除账户信息

deletebooking.jsp: 用于删除订单信息

deletecustomer.jsp: 用于删除用户信息

deleteticket.jsp: 用于删除票务信息

deletetrain.jsp: 用于删除车次信息

index.jsp: 用户登录界面

jquery.min.js: admin.html的配套文件

loginout.jsp: 用于退出登录

money.jsp: 用于显示账户余额

purchase.jsp: 用于买票

query1.jsp: 余票查询界面

selectaccount.jsp: 用于查询账户信息

selectcustomer.jsp: 用于查询用户信息

selectticket.jsp: 用于查询票务信息

selecttrain.jsp: 用于查询车次信息

train1.jsp: 用于显示余票信息

updateaccount.jsp: 用于修改账户信息

updatebooking.jsp: 用于修改订购信息

updatecustomer.jsp: 用于修改用户信息

updateticket.jsp: 用于修改票务信息

updatetrain.jsp: 用于修改车次信息

user.jsp: 用于显示用户信息

userbooking.jsp: 用于显示用户订单详情

wzsm.jsp: 网站声明