需求规格说明文档

姓名	学号	分工
聂尔聪	211250243	用例一、二
王渤元	211250225	用例一、四
罗晗	211250242	用例一、五
曲廷锌	211250183	用例一、三

版本	修改时间	内容
v1.0	2023.5.9	初步完成
v2.0	2023.7.7	增加功能性需求中的订单支付与取消

目录

1. 引言

1.1 目的

随着工作和生活节奏的加快，人们差旅的需求迅速提升，但是传统的车票预定购买模式有着许多不便之处，为了方便人们购票我们决定搭建一个互联网购票乘车系统，这是一个为客户提供列车详细信息，帮助客户选择和购买，方便管理员管理车票和铁路信息的系统。

1.2 范围

此平台是我们小组开发的在线购票平台，开发目标是帮助客户和管理员更好的进行票务相关活动，主要业务包括：（管理员）管理车票和铁路信息、
（客户）浏览和购买车票。

1.3 参考文档

- IEEE标准
- 《软件开发的技术基础》 丁二玉、刘钦编著；
- 用例文档

2. 总体描述

2.1 项目前景

2.1.1 背景与机遇

1. 随着工作和生活节奏的加快，人们对差旅出行的需求不断增加
2. 传统车票预定购买方式有着效率低、信息不及时、抢票难度大等问题

2.1.2 业务需求

BR1： 系统使用后，用户购票效率提高 30%
BR2： 系统使用三个月后，管理人员工作效率提高 50%
BR3： 系统使用六个月后，运营成本降低 15%
BR4： 系统使用六个月后，售票情况提高 10%
最好情况： 20%
最可能情况： 10%
最坏情况： 5%

2.2 项目功能

帮助客户和管理员更好的进行票务相关活动，主要业务包括：（管理员）管理车票和铁路信息、（客户）浏览和购买车票、积分制度。

2.3 用户特征

用户	用户特征

用户	用户特征
客户	对系统了解不是很多，随机用户
管理员	对系统操作掌握程度好，固定用户

2.4 约束

CON1 : 使用java语言编写后端，应用springboot架构
CON2: 数据库使用postgresql
CON3: 《中华人民共和国铁路法》
CON4: 以过期的票不能销售
CON5: 开发者需要在3个月内完成开发任务和测试。
CON6: 在开发中，开发者需要提交软件需求规格说明文档、设计描述文档。

2.5 假设与依赖

AE1： 每天平台登录用户数不会有太大的波动，系统被访问次数稳定。
AE2： 用户有基本的网页使用知识

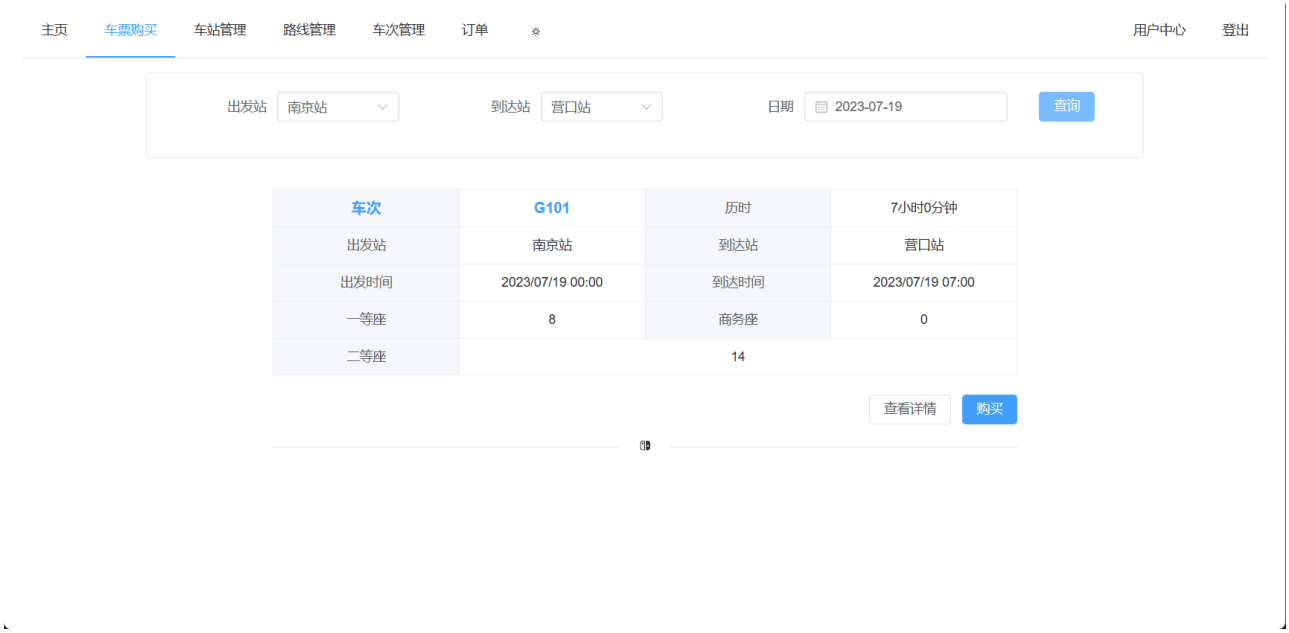
3. 需求详细描述

3.1 对外接口需求

3.1.1 用户界面

1. 登录注册界面

2. 查询车票界面
   用户输入出发地、目的地、日期来检索满足的车次



3. 支付/取消订单（选择支付策略和是否使用积分）

4. 显示当次订单详情

在用户完成支付或者支付失败或者取消订单时会显示本次订单详情

5. 展示个人所有订单
   用户在个人中心处点击订单信息可查看本人所有订单

6. 用户中心
   用户点击右上角的用户中心进入用户中心来查看和修改个人信息

3.1.2 通信接口

前后端通过***HTTP***通信

3.1.3 软件接口

使用***PostgreSQl***作为底层数据库存储各类数据

3.2 功能需求

3.2.1 个人基本信息管理

3.2.1.1 特征

客户打开个人信息界面时能方便快捷地维护个人基本信息
优先级=中

3.2.1.2 刺激/响应序列

刺激：用户点击登录界面上的“注册”按钮
响应：系统显示注册页面
刺激：用户在注册页面填写合法的身份证号，姓名，联系方式，用户名，密码,点击“确认”按钮
响应：系统返回到登录界面
刺激：用户在注册页面填写不合法的身份证号，姓名，联系方式，用户名，密码,点击“确认”按钮
响应：系统拒绝注册，并提示用户注册信息不合法，给出原因
刺激：用户在登录页面输入正确的用户名和密码，点击“登录”按钮
响应：系统跳转到用户主界面
刺激：用户在登录页面输入错误的用户名和密码，点击“登录”按钮
响应：系统拒绝登录，并提示用户登录信息错误
刺激：用户点击“个人基本信息”按钮
响应：系统显示姓名、性别、民族、籍贯、联系方式、身份证号
刺激：用户在个人基本信息页面修改姓名、性别、民族、籍贯、联系方式、身份证号，并点击“保存”按钮
响应：系统保存信息
刺激：用户点击“信用积分”按钮
响应：系统显示信用积分

3.2.1.3 相关功能需求

SignUp	注册
SignUp.showPage	显示注册页面
SignUp.success	注册成功
SignUp.invalid	注册信息不合法
LogIn	登录
LogIn.success	登录成功
LogIn.error	登录信息错误
BasicInformation	显示个人基本信息
BasicInformation.save	保存个人基本信息
CreditScore.show	显示信用积分

3.2.2 浏览车票详细信息

3.2.2.1 特征

用户点击进入车票查询界面，查看车票具体信息
优先级=中

3.2.2.2 刺激/响应序列

刺激：用户点击主界面上的“车票查询”按钮
响应：系统显示车票查询页面
刺激：用户在车票查询页面中输入合法的出发地、目的地和出行日期，并点击搜索按钮
响应：系统显示符合条件的车票列表，每一项包含车次、出发地、目的地、剩余票数
刺激：用户在车票查询页面中输入非法的出发地、目的地和出行日期，并点击搜索按钮
响应：系统拒绝查询并提示查询信息不合法
刺激：用户点击车票列表中的任意车票的“查看”按钮
响应：系统展示选中车票的出发时间、到达时间、价格、座位等级、车次、出发地、目的地、剩余票数
刺激：用户在车票详细信息界面点击“返回”按钮
响应：系统返回车票查询页面（车票列表）
刺激：用户在车票查询页面点击“返回”按钮
响应：系统返回主界面

3.2.2.3 相关功能需求

SearchTicket 车票查询按钮	
SearchTicket.showList	显示符合条件的车票列表
SearchTicket.showDetail	显示所选车票详细信息
SearchTicket.showDetail.return	返回车票查询页面
SearchTicket.invalid	查询信息不合法
SearchTicket.return	返回主界面

3.2.3 购买车票

3.2.3.1 特征

用户进入到购票系统，进行购票操作，选择座位等级和位置，以及是否使用积分，系统显示价格，用户进行确认是否购买，系统检测购票超时和恶意订票
优先级=中

3.2.3.2 刺激/响应序列

刺激: 用户在车票详细信息界面点击购票按钮
响应：系统将该车票的余票减1，开始计时，并跳转到购买页面，显示价格、座位等级、座位、出发地，目的地，时间
刺激：用户在购买页面选择座位等级一二三
响应：所选座位等级方框变色
刺激：用户在购买页面选择座位ABCDEF
响应：所选座位方框变色
刺激：用户在购买页面选择是否使用积分抵扣车票金额
响应：所选价格方框变色
刺激：用户在购买页面点击“确认购买”按钮

响应：系统生成购票记录，并添加时间、成功状态到个人购票信息记录中，返回车票查询页面
刺激：购票超时
响应：系统提示超时并强制返回车票查询页面，车票加1，添加时间、超时状态到个人购票信息记录中
刺激：用户的个人购票信息记录构成恶意订票
响应：系统扣除该用户的信用积分10分

3.2.3.3 相关功能需求

buy.click	进入并显示购票界面，时钟开始计时，车票余票减1，显示车票信息
buy.score	选用积分抵扣
buy.seatGrade	选择座位等级
buy.seat	选择座位
buy.finish	购票完成，生成购票记录，添加到个人购票信息记录中，返回车票查询页面
buy.check.timeout	检测购票超时
buy.check.spite	检测恶意订票，扣除信用积分

3.2.4 铁路信息管理

3.2.4.1 特征

铁路管理员登录到铁路管理系统之后，进行查询铁路信息，新建铁路信息，修改铁路信息的操作
优先级=中

3.2.4.2 刺激/响应序列

刺激：铁路管理员在登录页面输入正确账号密码
响应：系统跳转到管理员主界面
刺激：铁路管理员在登录页面输入错误账号密码
响应：系统拒绝登录，并提示登录信息错误
刺激：管理员在主界面点击“查询路线信息”
响应：系统跳转到查询信息界面，显示现有路线信息列表，每一项包含线路的途经站点及到达、出发时间
刺激：管理员在查询信息界面点击“新增”
响应：系统显示新增路线界面。
刺激：管理员在新增路线界面输入路线信息并点击保存
响应：系统处理请求
刺激：管理员选择路线信息列表中的一项，点击“修改”
响应：系统显示路线信息修改界面
刺激：管理员在路线信息修改界面修改路线信息，并点击“保存”按钮
响应：系统处理请求
刺激：新增路线信息不合法
响应：系统提示不合法并拒绝保存
刺激：修改路线信息不合法
响应：系统提示修改不合法并拒绝保存

3.2.4.3 功能需求

RIM.login.input	系统应该允许管理员在登陆界面进行键盘输入
RIM.login.success	登陆成功，跳转
RIM.login.fail	登陆失败
RIM.new.check	检测新增路线信息的合法性
RIM.modify.check	检测修改路线信息的合法性
RIM.new	系统更新新建之后的信息并返回界面
RIM.modify	系统更新修改之后的信息并返回界面
RIM.query	系统显示查询界面

3.2.5 车票信息管理

3.2.5.1 特征

铁路管理员可以在车票信息管理界面修改更新车票信息，系统根据输入的信息判断是否合法，成功则更新车票信息并展示更新后的车票信息列表，失败则提示信息不合法要求重新输入

优先级=中

3.2.5.2 刺激/响应序列

刺激：铁路管理员在主界面点击“车票信息管理”按钮。
响应：系统展示当前车票信息列表。
刺激：铁路管理员点击“设置”按钮。
响应：系统显示车票设置界面
刺激：管理员设置车票信息，包括出车次、发地、目的地、时间、价格、票的数目，并点击“保存”按钮
响应：系统更新车票信息并展示更新后的车票信息列表
刺激：管理员设置车票信息非法
响应：系统提示信息不合法，拒绝保存

3.2.5.3 相关功能需求

admin.show.all	系统显示车票信息列表
admin.set	系统显示车票信息设置界面
admin.set.check	系统检查输入信息是否合法
admin.set.update	系统更新车票信息

3.2.6 订单的支付与取消

3.2.6.1 特征

客户点击“订单提交”按钮时能选择不同支付策略和是或否使用积分来支付订单或者取消订单
优先级=中

3.2.6.2 刺激/响应序列

刺激：用户点击订单确认界面的提交按钮
响应：系统显示订单详情界面
刺激：用户在订单详情页面选择是否使用积分
响应：系统根据是否使用积分展示不同的按钮
刺激：用户选择支付策略（支付宝或者微信）
响应：系统展示下拉选择框
刺激：用户点击“支付订单”按钮
响应：系统展示支付详情
刺激：用户点击“取消订单”按钮
响应：系统展示取消成功弹窗并跳转到订单详情界面

3.2.6.3 相关功能需求

PayOrder	支付订单
PayOrder.setUsePoints	是否使用积分
PayOrder.setPaymentStrategy	选择支付策略
CancelOrder	取消订单
cancelOrder.returnSeat	归还座位
PaymentStrategy	支付策略
PaymentStrategy.pay	支付订单
PaymentStrategy.cancel	取消订单
PaymentStrategy.usePoints	使用积分
Order.payDetail	显示订单信息
orderDao.save	保存订单信息

3.3 非功能性需求

3.3.1 安全性

Safety1： 系统应该只允许经过验证和授权的用户访问
Safety2： 系统应该按照用户的身份验证用户相应的操作权限
Safety3： 用户只能查看和维护自己的信息
Safety4： 为防信息泄露，用户的密码必须密文存储

3.3.2 可维护性

Modifiability： 如果系统要增加新的支付功能，要能够在3人一天内完成

3.3.3 易用性

Usability： 用户不需要额外的培训，对于任意操作可以在10分钟内完成

3.3.4 可靠性

Reliability： 当客户端与服务端通信时，如果网络出现故障，系统不能出现故障

3.3.5 业务规则

获取订单后根据车次设置的价格与用户的**里程积分**计算订单价格

1. 里程积分达到1000，则该1000积分可以折 0.1%；
2. 1000到3000积分， 则该2000积分可以折0.15%；
3. 3000到10000积分， 则该7000积分可以折0.2%；
4. 10000到50000积分， 则该40000积分可以折0.25%；
5. 50000积分以上的积分，可以折0.3%

3.3.6 约束

CON1 : 使用java语言编写后端，应用springboot架构
CON2: 数据库使用postgresql
CON3: 《中华人民共和国铁路法》
CON4: 以过期的票不能销售
CON5: 开发者需要在3个月内完成开发任务和测试。
CON6: 在开发中，开发者需要提交软件需求规格说明文档、设计描述文档。

3.4 数据需求

3.4.1 数据定义

1. User类

- 包含用户的ID、用户名、密码、名字、电话号码、类别、身份证号、创建时间、修改时间。
- User类的属性如下

```
public class User {
  @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @NotNull
    @Column(unique = true)
    private String username;
    @NotNull
    private String password;

    private String name;
    private String phone;
    private String type;
    private String idn;
```

```
    @Column
    private String identity;
    @Column
    private double mileagePoints;
    @Column
    private double aliBalance;
    @Column
    private double wechatBalance;

    @CreationTimestamp
    private Date createdAt;

    @UpdateTimestamp
    private Date updatedAt;
}
```

```
CREATE TABLE "User" (
  id SERIAL PRIMARY KEY,
  username VARCHAR(255) NOT NULL UNIQUE,
  password VARCHAR(255) NOT NULL,
  name VARCHAR(255),
  phone VARCHAR(255),
  type VARCHAR(255),
  idn VARCHAR(255),
  identity TEXT,
  mileagePoints DOUBLE PRECISION,
  aliBalance DOUBLE PRECISION,
  wechatBalance DOUBLE PRECISION,
  createdAt TIMESTAMP,
  updatedAt TIMESTAMP
);
```

2. Train类

- 包含车辆编号、名字、路线号、座位售卖情况、车辆类型、日期、到达时间、出发时间、额外信息、创建日期、更新日期。
- Train类如下

```
public class Train {
   @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @NotNull
    private String name;

    @NotNull
    private Long routeId;
```

```
    @NotNull
    @Type(BooleanArrayType.class)
    @Column(name = "seats", columnDefinition = "boolean[][]")
    private boolean[][] seats;

    @NotNull
    private TrainType trainType;

    @NotNull
    private String date;

    @NotNull
    private List<Date> arrivalTimes;

    @NotNull
    private List<Date> departureTimes;

    @NotNull
    private List<String> extraInfos;

    @CreationTimestamp
    private Date createdAt;

    @UpdateTimestamp
    private Date updatedAt;
}
```

```
CREATE TABLE train (
    id SERIAL PRIMARY KEY,
    name VARCHAR NOT NULL,
    route_id BIGINT NOT NULL,
    seats BOOLEAN[][] NOT NULL,
    train_type VARCHAR NOT NULL,
    date DATE NOT NULL,
    arrival_times TIMESTAMP[] NOT NULL,
    departure_times TIMESTAMP[] NOT NULL,
    extra_infos VARCHAR[] NOT NULL,
    created_at TIMESTAMP,
    updated_at TIMESTAMP
);
```

3. Station

- 包含车站编号、名字、创建时间、更新日期。
- Station类如下。

```
public class Station {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
```

```
    private Long id;

    @NotNull
    @Column(unique = true)
    private String name;

    @CreationTimestamp
    private Date createdAt;

    @UpdateTimestamp
    private Date updatedAt;
}
```

```
CREATE TABLE station (
    id SERIAL PRIMARY KEY,
    name VARCHAR(255) NOT NULL UNIQUE,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

4. Route

- 包含线路编号、名字、车站编号、创建日期、更新日期。
- Route类如下

```
public class Route{
     @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @NotNull
    @Column
    private String name;

    private List<Long> stationIds;

    @CreationTimestamp
    private Date createdAt;

    @UpdateTimestamp
    private Date updatedAt;
}
```

```
CREATE TABLE route (
    id SERIAL PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    station_ids INTEGER[],
```

```
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

5. Order类

- 包含订单号、用户编号、车辆编号、始发站终点站编号、价格、使用积分、奖励积分、消耗积分、订单状态、座位信息、创建时间、修改时间等。
- Order类

```
public class Order{
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @NotNull
    private Long userId;

    @NotNull
    private Long trainId;

    @NotNull
    private Long departureStationId;

    @NotNull
    private Long arrivalStationId;

    @Column
    private double price;
    @Column
    private boolean usePoints;
    @Column
    private double bonusPoints; //奖励积分
    @Column
    private double consumptionPoints; //消耗积分
    @Column
    private PaymentStrategy.strategy paymentStrategy;

    @NotNull
    private OrderStatus status;

    @NotNull
    private String seat;

    @CreationTimestamp
    private Date createdAt;

    @UpdateTimestamp
    private Date updatedAt;
}
```

```
CREATE TABLE orders (
    id SERIAL PRIMARY KEY,
    user_id BIGINT NOT NULL,
    train_id BIGINT NOT NULL,
    departure_station_id BIGINT NOT NULL,
    arrival_station_id BIGINT NOT NULL,
    price DOUBLE PRECISION,
    use_points BOOLEAN,
    bonus_points DOUBLE PRECISION,
    consumption_points DOUBLE PRECISION,
    payment_strategy VARCHAR,
    status VARCHAR,
    seat VARCHAR NOT NULL,
    created_at TIMESTAMP,
    updated_at TIMESTAMP
);
```

3.4.2 默认数据

1. 用户类型默认为普通用户
2. 用户余额默认为0
3. 座位默认为全部未售出
4. 订单积分默认为0
5. 操作人员默认为当前登录用户
6. 更新和创建时间默认为当天

3.4.3 数据格式要求

1. 日期的格式必须是yyyy-MM-dd
2. 时间的格式必须是hh-mm-ss
3. 用户余额必须是正数或0。
4. 里程积分必须是正数或0。

3.5 其他

1. 系统投入使用时，需要对管理人员进行一周的集中培训