# l23o6软件评审



## 1. 需求评审：

为满足我国人民铁路交通运输的需求，解决人们买票的问题，我们小组开发了一个类似于12306的产品。期望通过此产品实现用户买票，退票，退款，使用积分折扣付款，使用不同支付策略付款等功能，并实现铁路管理员站台管理，车次管理，路线管理等功能。预计在7月1日实现基本买票退票以及铁路管理功能等基础功能，在7月4日实现包括代码测试、评审以及完成所有功能。 本次讨论和评审的需求包括但不限于：

1. 用户与管理员的注册和登录：团队用两个不同的属性分别代表用户或者管理员并实现两个实例下的分别功能需求，在注册与登录时需要检测注册信息与登录信息。
2. 车票搜索和购买：团队会仔细审查车票搜索和购买的需求，包括用户可以输入出发地、目的地和日期进行搜索，以及购买车票的流程和支付方式。
3. 座位选择和座位类型：团队会检查座位选择和座位类型的需求，确认系统是否支持用户在购票时选择座位，并区分座位类型（如普通座、商务座）。
4. 班次和车次信息：需求评审还可能涉及班次和车次信息的需求，包括每个班次的出发时间、车型和站点信息，以及票价和余票信息。
5. 订单管理和退款：团队会检查订单管理和退款的需求，确保系统可以管理用户的订单，提供退票和退款的流程，并处理相关的退款事务。 通过对这些需求进行评审和讨论，团队可以确保需求的准确性、一致性和可行性，并为开发阶段提供一个清晰的指导。

## 2. 体系结构评审：

1. 分层体系结构（Layered Architecture）：

   用户界面层（Presentation Layer）：处理用户界面和用户交互，包括用户注册、登录、买票、退票、退款等功能。
   业务逻辑层（Business Logic Layer）：处理核心业务逻辑，包括用户身份验证、票务管理、支付策略、积分折扣计算等功能。
   数据访问层（Data Access Layer）：负责与数据库进行交互，处理数据的读取和存储，包括用户信息、车次信息、订单信息等。

2. 客户端-服务器体系结构（Client-Server Architecture）：将系统划分为客户端和服务器两个主要组件。

   客户端（Client）：提供用户界面和用户交互功能，如网页界面、移动应用等。用户可以通过客户端进行买票、退票、退款等操作。
   服务器（Server）：处理业务逻辑和数据存储，包括用户身份验证、票务管理、支付处理、数据存储等功能。服务器负责处理用户请求，与铁路系统进行交互，并响应客户端请求。

3. 微服务架构（Microservices Architecture）：将系统划分为多个独立的微服务，每个微服务负责一个特定的功能。

```
用户服务（User Service）：处理用户注册、登录、积分管理等功能。
车次服务（Train Service）：管理车次信息等功能。
路线服务（Route Service）：管理路线信息、增删改路线等功能。
站台服务（Station Service）：管理站台信息、增删改站台等功能。
订单服务（Order Service）：管理订单信息，包括创建订单、取消订单、支付订单等功能。
支付策略（payment Strategy）：实现用户使用微信或者支付宝支付的功能。
```

这种体系结构设计方案可以实现用户购票、退票、退款，使用积分折扣付款，使用不同支付策略付款等功能。同时，铁路管理员可以通过管理员服务管理站台、车次和路线等信息。 进而实现所有需求功能。

# 3. 代码评审：

为了确保代码的质量和可维护性，进行了代码评审工作。
本次代码评审的内容包括但不限于：

1. RouteServiceIpml.addRoute里应该加入关于路线名规范的检测功能以及识别是否加入了同一重复相同路线的功能。
2. RouteServiceIpml.editRoute与RouteServiceIpml.deleteRoute中应该加入如果已有车次使用次路线不能更改的检测功能。
3. StationServiceIpml.editStation与StationServiceIpml.deleteStation中应该加入识别是否加入了同名站点的功能以及如果已有车次使用次路线不能更改的检测功能。
4. TrainServiceIpml.addTrain与TrainServiceIpml.changeTrain中应该加入应该加入关于火车名规范的检测功能以及经停站时间处理的检测功能。
5. 在UserServiceImpl.register中模拟设置用户的积分以及余额以进行测试。
6. 在Payment Strategy里用两种支付策略满足策略模式。
7. 在积分抵扣策略中应该使用表驱动的方式阶梯型折扣付款。
8. 在前端与后端中应扩展orderDetailData对象的属性，使其可以存储和更新更多与订单详情相关的数据：比如bonus_points，consumption_points，left_points，use_points。
9. 在积分策略中应设置一个是否使用积分的按钮供用户选择。
10. 用户以及管理员应该得到不同的客户端界面。
11. RouteServiceImpl.addRoute 与 RouteServiceImpl.editRoute 中存在对信息合法性判断的重复代码，将其抽象为函数复用，提高可读性和易修改性。
12. StationServiceImpl.addStation/stationServiceImpl 和 TrainServiceImpl.addTrain/ trainServiceImpl.editTrain 同样使用11提到的方法提高可读性和易修改性。