华东交通大学软件学院

2022 年 8 周软件工程实训总结报告

专业班级：软件（软件开发）2019-1　　学　　号：2019211001000311
学生姓名：郭瑄　　指导教师：付小军
所在项目：全国列车车次查询系统　　项目经理：曾雪斌
实训公司：北京传习公司
实训周期：2022 年 05 月 09 日至 2022 年 07 月 01 日

总 结 内 容

一、 团队基本情况

小组人数：8 人
小组组长：曾雪斌
小组成员：蒋梦依、曾聆聆、殷和建、郭瑄、胡子豪、刘贤洋、罗宇豪

二、 项目基本概况

项目名称：全国列车车次查询系统

主要内容：

系统开发的总体任务是实现旅客对列车车次查询的系统化、规范化和自动化。系统功能分析是在系统开发的总体任务的基础上完成。

网站的使用人员主要有游客人员和管理员人员，这些用户在具体操作的时候还会涉及到更多更细的分类，具体可参见功能设计部分的描述。全国列车车次查询系统需要完成的功能主要有如下几点：

(1) 管理员人员能对列车的车次信息（包括车次、始发地、始发时间、到达各站的时间、到达终点站的时间等）进行录入、修改和删除等操作。

(2) 管理员人员能对车站信息（包括车站名称、车站检票口等）进行录入和删除等操作。

(3) 管理员人员能对列车信息（包括列车名、列车座位等）信息进行录入、修改、删除等操作。

(4) 游客用户能根据出发地、目的地和出发时间查询车票信息。

(5) 游客用户能根据出发地、目的地、出发时间返回时间查询车票信息。

(6) 游客用户能根据出发地、目的地和出发时间查询车票换乘信息。

(7) 游客用户能根据车站和车次查询车票正晚点信息。

(8) 游客用户能根据出发时间、车次和乘车站查询检票口信息。

(9) 游客用户能根据起售时间和乘车站查询车票起售时间。

(10) 游客用户能根据乘车时间和车次查询列车时刻表。

(11) 游客用户能根据出发地、目的地和出发时间查询列车票价。

前台主要是让旅客了解各种列车信息以及车站最新动态情况，还有实现对旅客所需信息的查询和最新的新闻动态等功能。

项目开发的方式以 B/S 为主，采用三层架构或 MVC 架构模式，保证项目架构设计的合理有效，具有较强的可维护性，力求项目投入生产环境后运行高效稳定。

三、本人在项目中的主要任务及完成情况

1）主要任务简介及完成情况

本人在项目中主要负责后端代码编写，在本次的项目中完成了车票价格查询，正晚点查询，检票口查询，时刻表查询等网站基本功能的后端代码编写工作，所有任务均已完成。

正晚点查询功能实现：

控制器代码：

```
@RequestMapping("front_late")//查询列车是否晚点
public String TimeQuery(Model model,String train_id,String station_name) throws SQLException {
    String msg="查询为空，请输入正确信息";
    if (train_id != null && station_name!=null){
        //查找起始站晚点信息
        if (queryService.findTrainById(train_id)!=null){
            Train train=queryService.findTrainById(train_id);
            if (station_name.equals(train.getBeginStation())){
                if (train.getLateTime()==0){
                    msg="列车准时";
                }else {
                    msg="列车晚点";
                }
            }else if (station_name.equals(train.getEndStation())){
                if (train.getLateTime()==0){
                    msg="列车准时";
                }else {
                    msg="列车晚点";
                }
            }

        }
        //查找经停站晚点信息
        if (queryService.findTrainChainByTrainAndStation(train_id, station_name)!=null){
            TrainChain trainChain=queryService.findTrainChainByTrainAndStation(train_id, station_name);
            if (trainChain.getLateTime()==0){
                msg="列车准时";
            }else if (trainChain.getLateTime()==1){
                msg="列车晚点";
            }
```

```
            }
        }
        model.addAttribute("msg",msg);
        return "front_late";
    }
```

逻辑层代码：

```
@Override
public Train findTrainById(String train_id) {
    return queryDao.findTrainById(train_id);
}

@Override
public TrainChain findTrainChainByTrainAndStation(String train_id, String station_name) {
    return queryDao.findTrainChainByTrainAndStation(train_id,station_name);
}
```

数据持久层代码：

```
<resultMap id="train_model" type="Train">
    <id property="trainId" column="train_id"></id>
    <result property="trainType" column="train_type"></result>
    <result property="beginStation" column="begin_station"></result>
    <result property="endStation" column="end_station"></result>
    <result property="beginTime" column="begin_time"></result>
    <result property="endTime" column="end_time"></result>
    <result property="times" column="times"></result>
    <result property="lateTime" column="late_time"></result>
    <result property="note" column="note"></result>
    <result property="checkTicket" column="check_ticket_id"></result>
</resultMap>
<select id="findTrainById" resultMap="train_model">
    select *
    from t_train
    where train_id=#{train_id}
</select>

<resultMap id="TrainChain_model" type="TrainChain">
    <id property="trainId" column="train_id"></id>
    <result property="stationName" column="station_name"></result>
    <result property="tempBeginTime" column="temp_begin_time"></result>
    <result property="tempEndTime" column="temp_end_time"></result>
    <result property="sumTime" column="sum_time"></result>
    <result property="ticketMoney" column="ticket_id"></result>
    <result property="checkTicket" column="check_ticket_id"></result>
    <result property="lateTime" column="late_time"></result>
```

```
</resultMap>
<select id="findTrainChainByTrainAndStation" resultMap="TrainChain_model">
    select *
    from t_train_chain
    where train_id=#{train_id} and station_name=#{station_name}
</select>
```

2.时刻表查询功能实现：

控制器代码：

```
@RequestMapping("front_schedule")
public String scheduleQuery(Model model,String trainId,String date){
    String msg="当日无该车次";
    Train train = new Train();
    List<TrainChain> trainChainList=new ArrayList<TrainChain>();
    List<String> arriveDay = new ArrayList<>();
    if (trainId!=null&&date!=null){
        if(queryService.findTrainRunByDateAndTrainId(trainId,date)!=null){
                train = queryService.findTrainById(trainId);
                trainChainList=queryService.findTrainChainByTrainId(trainId);
                model.addAttribute("train",train);
                model.addAttribute("trainChainList",trainChainList);
                int stationCount = 2+trainChainList.size();
                model.addAttribute("stationCount",stationCount);
                msg="当日存在该车次";
        }
    }
    if (train.getBeginTime().compareTo(trainChainList.get(0).getTempBeginTime())<0){
        arriveDay.add("当日到达");
    }else {
        arriveDay.add("次日到达");
    }
    for (int i =1;i<trainChainList.size();i++){
        if
(trainChainList.get(i-1).getTempEndTime().compareTo(trainChainList.get(i).getTempBeginTime())<0){
            arriveDay.add("当日到达");
        }else {
            arriveDay.add("次日到达");
        }
    }
    if (trainChainList.get(trainChainList.size()-1).getTempEndTime().compareTo(train.getEndTime())<0){
        arriveDay.add("当日到达");
    }else {
        arriveDay.add("次日到达");
    }
    model.addAttribute("msg",msg);
```

```
        model.addAttribute("arriveDay",arriveDay);
        return "front_schedule";
    }
```

逻辑层代码：

```
@Override
public TrainRun findTrainRunByDateAndTrainId(String id, String date) {
        return queryDao.findTrainRunByDateAndTrainId(id,date);
    }
@Override
public List<TrainChain> findTrainChainByTrainId(String train_id) {
        return queryDao.findTrainChainByTrainId(train_id);
    }
```

数据持久层代码：

```
<resultMap id="TrainRun_model" type="TrainRun">
        <id property="trainId" column="fk_train_id"></id>
        <result property="runDate" column="run_date"></result>
    </resultMap>
    <select id="findTrainRunByDateAndTrainId" resultMap="TrainRun_model">
        select run_date,fk_train_id
        from t_train_run
        where run_date=#{date} and fk_train_id=#{id}
    </select>
    <resultMap id="TrainChain_model" type="TrainChain">
        <id property="trainId" column="train_id"></id>
        <result property="stationName" column="station_name"></result>
        <result property="tempBeginTime" column="temp_begin_time"></result>
        <result property="tempEndTime" column="temp_end_time"></result>
        <result property="sumTime" column="sum_time"></result>
        <result property="ticketMoney" column="ticket_id"></result>
        <result property="checkTicket" column="check_ticket_id"></result>
        <result property="lateTime" column="late_time"></result>
    </resultMap>
    <select id="findTrainChainByTrainId" resultMap="TrainChain_model">
        select *
        from t_train_chain
        where train_id=#{train_id}
    </select>
```

3.检票口查询功能实现

控制器代码：

```
@RequestMapping("front_checkIn")
public String CheckTicketQuery(Model model,String date,String stationId,String station) throws SQLException {
        String msg=" /";
        if(queryService.findTrainRunByDateAndTrainId(stationId,date)!=null){
```

```
            if (stationId != null && station!=null){
                TrainChain trainChain=queryService.findTrainChainByTrainAndStation(stationId, station);
                if (trainChain==null){
                    msg = " /";
                }else {
                    CheckTicket checkTicket = queryService.findCheckTicketById(trainChain.getCheckTicket());
                    msg=checkTicket.getName();
                }
            }
        }
        model.addAttribute("msg",msg);
        return "front_checkIn";
    }
```

逻辑层代码：

```
@Override
public TrainChain findTrainChainByTrainAndStation(String train_id, String station_name) {
    return queryDao.findTrainChainByTrainAndStation(train_id,station_name);
}

@Override
public CheckTicket findCheckTicketById(String id) {
    return queryDao.findCheckTicketById(id);
}

@Override
public TrainRun findTrainRunByDateAndTrainId(String id, String date) {
    return queryDao.findTrainRunByDateAndTrainId(id,date);
}
```

数据持久层代码：

```
<resultMap id="TrainChain_model" type="TrainChain">
    <id property="trainId" column="train_id"></id>
    <result property="stationName" column="station_name"></result>
    <result property="tempBeginTime" column="temp_begin_time"></result>
    <result property="tempEndTime" column="temp_end_time"></result>
    <result property="sumTime" column="sum_time"></result>
    <result property="ticketMoney" column="ticket_id"></result>
    <result property="checkTicket" column="check_ticket_id"></result>
    <result property="lateTime" column="late_time"></result>
</resultMap>
<select id="findTrainChainByTrainAndStation" resultMap="TrainChain_model">
    select *
    from t_train_chain
    where train_id=#{train_id} and station_name=#{station_name}
```

```
</select>
<resultMap id="CheckTicket_model" type="CheckTicket">
    <id property="name" column="check_ticket_name"></id>
    <result property="stationName" column="station_name"></result>
</resultMap>
<select id="findCheckTicketById" resultMap="CheckTicket_model">
    SELECT check_ticket_name,station_name
    FROM t_station s,t_check_ticket c
    WHERE s.station_id = c.fk_station_id AND c.check_ticket_id =#{id}
</select>
<resultMap id="TrainRun_model" type="TrainRun">
    <id property="trainId" column="fk_train_id"></id>
    <result property="runDate" column="run_date"></result>
</resultMap>
<select id="findTrainRunByDateAndTrainId" resultMap="TrainRun_model">
    select run_date,fk_train_id
    from t_train_run
    where run_date=#{date} and fk_train_id=#{id}
</select>
```

4.起售时间查询

控制器代码：

```
@RequestMapping("front_saleTime")
public String SaleTimeQuery(Model model,String cityName){
    List<Station> stationList=new ArrayList<Station>();
    stationList=queryService.findStationByCityName(cityName);
    model.addAttribute("stationList",stationList);
    return "front_saleTime";
}
```

逻辑层代码：

```
@Override
public List<Station> findStationByCityName(String cityName) {
    return queryDao.findStationByCityName(cityName);
}
```

数据持久层代码：

```
<resultMap id="Station_model" type="Station">
    <id property="name" column="station_name"></id>
    <result property="cityName" column="c_name"></result>
    <result property="ticketStartSell" column="ticket_start_sell"></result>
</resultMap>
<select id="findStationByCityName" resultMap="Station_model">
    SELECT station_name,ticket_start_sell,c_name
    FROM t_station s,t_city c
```

```
    WHERE s.fk_cid=c.c_id AND c.c_name = #{cityName}
</select>
```

5.车票价格查询

控制器代码：

```
    @RequestMapping("front_price")
    public String priceQuery(Model model,String fromPlace,String arrivePlace,String goDate){
        List<Station> fromStations= queryService.findStationByCityName(fromPlace);
        List<Station> arriveStations = queryService.findStationByCityName(arrivePlace);
        List<Train> trainList= new ArrayList<>();
        String msg = "当日无车次";
        List<String> arriveDay = new ArrayList<>();
//          double pricePercent=1.0;
        List<TicketMoney> ticketMoneyList = new ArrayList<>();

        int flag=0;

        for (int i=0;i<fromStations.size();i++){
            for (int j=0;j<arriveStations.size();j++){
                List<Train> trains =
queryService.findTrainByStation(fromStations.get(i).getName(),arriveStations.get(j).getName());

                if (trains!=null){
                    for (int z=0;z<trains.size();z++) {
                        if (queryService.findTrainRunByDateAndTrainId(trains.get(z).getTrainId(),
goDate) != null) {
                            if (trains.get(z).getBeginStation().equals(fromStations.get(i).getName())) {
                                if (trains.get(z).getEndStation().equals(arriveStations.get(i).getName())
== false) {
                                    TrainChain trainChain =
queryService.findTrainChainByTrainAndStation(trains.get(z).getTrainId(), arriveStations.get(j).getName());
                                    double pricePercent = TimePercent(trains.get(z).getBeginTime(),
trainChain.getTempBeginTime(), trains.get(z).getTimes());
                                    System.out.println();
                                    System.out.println("百分比" + pricePercent);
                                    System.out.println();

                                    trains.get(z).setEndStation(arriveStations.get(i).getName());
                                    trains.get(z).setEndTime(trainChain.getTempBeginTime());
                                    trains.get(z).setTimes(trainChain.getSumTime());
                                    TicketMoney ticketMoney =
queryService.findPriceByTrainId(trains.get(z).getTrainId());

                                    if
(trains.get(z).getBeginTime().compareTo(trains.get(z).getEndTime())<0){
```

```
                                        arriveDay.add("当日到达");
                                    }else {
                                        arriveDay.add("次日到达");
                                    }

                                    if (ticketMoney != null) ticketMoney = NewPrice(ticketMoney,
pricePercent);
                                    ticketMoneyList.add(ticketMoney);
                                } else {
                                    TicketMoney ticketMoney =
queryService.findPriceByTrainId(trains.get(z).getTrainId());
                                    ticketMoneyList.add(ticketMoney);

                                    if
(trains.get(z).getBeginTime().compareTo(trains.get(z).getEndTime())<0){
                                        arriveDay.add("当日到达");
                                    }else {
                                        arriveDay.add("次日到达");
                                    }
                                }
                            } else {
                                if (trains.get(z).getEndStation().equals(arriveStations.get(i).getName())) {
                                    TrainChain trainChain =
queryService.findTrainChainByTrainAndStation(trains.get(z).getTrainId(), fromStations.get(i).getName());
                                    double pricePercent = TimePercent(trainChain.getTempEndTime(),
trains.get(z).getEndTime(), trains.get(z).getTimes());

                                    trains.get(z).setBeginStation(fromStations.get(i).getName());
                                    trains.get(z).setBeginTime(trainChain.getTempEndTime());

                                    Date a =
QueryController.computationTime(trainChain.getTempEndTime(), trains.get(z).getEndTime());
                                    trains.get(z).setTimes(a);

                                    TicketMoney ticketMoney =
queryService.findPriceByTrainId(trains.get(z).getTrainId());
                                    if (ticketMoney != null) ticketMoney = NewPrice(ticketMoney,
pricePercent);
                                    ticketMoneyList.add(ticketMoney);

                                    if
(trains.get(z).getBeginTime().compareTo(trains.get(z).getEndTime())<0){
                                        arriveDay.add("当日到达");
                                    }else {
                                        arriveDay.add("次日到达");
```

```
                                        }
                                    } else {
                                        TrainChain trainChainFrom =
queryService.findTrainChainByTrainAndStation(trains.get(z).getTrainId(), fromStations.get(i).getName());
                                        TrainChain trainChainArrive =
queryService.findTrainChainByTrainAndStation(trains.get(z).getTrainId(), arriveStations.get(j).getName());
                                        double pricePercent =
TimePercent(trainChainFrom.getTempEndTime(), trainChainArrive.getTempBeginTime(),
trains.get(z).getTimes());

                                        trains.get(z).setBeginStation(fromStations.get(i).getName());
                                        trains.get(z).setEndStation(arriveStations.get(i).getName());

                                        trains.get(z).setBeginTime(trainChainFrom.getTempEndTime());
                                        trains.get(z).setEndTime(trainChainArrive.getTempBeginTime());

                                        Date a =
QueryController.computationTime(trainChainFrom.getTempEndTime(), trainChainArrive.getTempBeginTime());
                                        trains.get(z).setTimes(a);

                                        TicketMoney ticketMoney =
queryService.findPriceByTrainId(trains.get(z).getTrainId());
                                        if (ticketMoney != null) ticketMoney = NewPrice(ticketMoney,
pricePercent);
                                        ticketMoneyList.add(ticketMoney);

                                        if
(trains.get(z).getBeginTime().compareTo(trains.get(z).getEndTime())<0){
                                            arriveDay.add("当日到达");
                                        }else {
                                            arriveDay.add("次日到达");
                                        }
                                    }
                                }
                                trains.get(z).setBeginStation(fromStations.get(i).getName());
                                trains.get(z).setEndStation(arriveStations.get(j).getName());
                                trainList.addAll(trains);
                                flag++;
                                msg = "有车次";
                            }
                        }
                    }
                }
            }
            System.out.println(trainList);
```

```
        if (flag==0){
            System.out.println("test");
        }
        model.addAttribute("arriveDay",arriveDay);
        model.addAttribute("trainList",trainList);
        model.addAttribute("msg",msg);
        model.addAttribute("ticketMoneyList",ticketMoneyList);
        return "front_price";
    }
```

逻辑层代码：

```
    @Override
    public List<Train> findTrainByStation(String beginStationName, String endStationName) {
        List<String> TrainIds = new ArrayList<String>();
        TrainIds.add("0");
        //查找终点站车次
        List<String> FromTrainIds = queryDao.findTrainIdByBeginStationName(beginStationName);
        List<String> ArriveTrainIds =queryDao.findTrainIdByEndStationName(endStationName);
        //查找途径站车次
        List<TrainChain> FromTrainChains = queryDao.findTrainChainByStation(beginStationName);
        List<TrainChain> ArriveTrainChains = queryDao.findTrainChainByStation(endStationName);
        //将目的地为途径站的车次添加
        if (FromTrainChains.size()==0){
            for (int i=0;i<ArriveTrainChains.size();i++){
                ArriveTrainIds.add(ArriveTrainChains.get(i).getTrainId());
            }
        }
        //将出发地为途径站的车次添加
        if (ArriveTrainChains.size()==0){
            for (int j=0;j<FromTrainChains.size();j++){
                FromTrainIds.add(FromTrainChains.get(j).getTrainId());
            }
        }
        //添加出发站到达站均为途径站的车次
        for (int i=0;i<FromTrainChains.size();i++){
            for (int j=0;j<ArriveTrainChains.size();j++){
                if (FromTrainChains.get(i).getTrainId().equals(ArriveTrainChains.get(j).getTrainId())){
                    if
(FromTrainChains.get(i).getTempBeginTime().compareTo(ArriveTrainChains.get(j).getTempBeginTime())<0){
                        FromTrainIds.add(FromTrainChains.get(i).getTrainId());
                        ArriveTrainIds.add(ArriveTrainChains.get(j).getTrainId());
                    }
                }else {
                    FromTrainIds.add(FromTrainChains.get(i).getTrainId());
                    ArriveTrainIds.add(ArriveTrainChains.get(j).getTrainId());
```

```
                    }
                }
            }
            //判断可用车次
            for (int i=0;i<FromTrainIds.size();i++){
                for (int j=0;j<ArriveTrainIds.size();j++){
                    if (FromTrainIds.get(i).equals(ArriveTrainIds.get(j))){
                        TrainIds.add(FromTrainIds.get(i));
                    }
                }
            }
            //查找车辆信息
            List<Train> Trains = new ArrayList<Train>();
            int flag=0;
            for (int i=1;i<TrainIds.size();i++){
                Train train=queryDao.findTrainById(TrainIds.get(i));
                Trains.add(train);
                flag++;
            }
            if (flag!=0){
                return Trains;
            }else {
                return null;
            }
        }
        @Override
        public TicketMoney findPriceByTrainId(String trainId) {
            return queryDao.findPriceByTrainId(trainId);
        }
```

数据持久层代码：

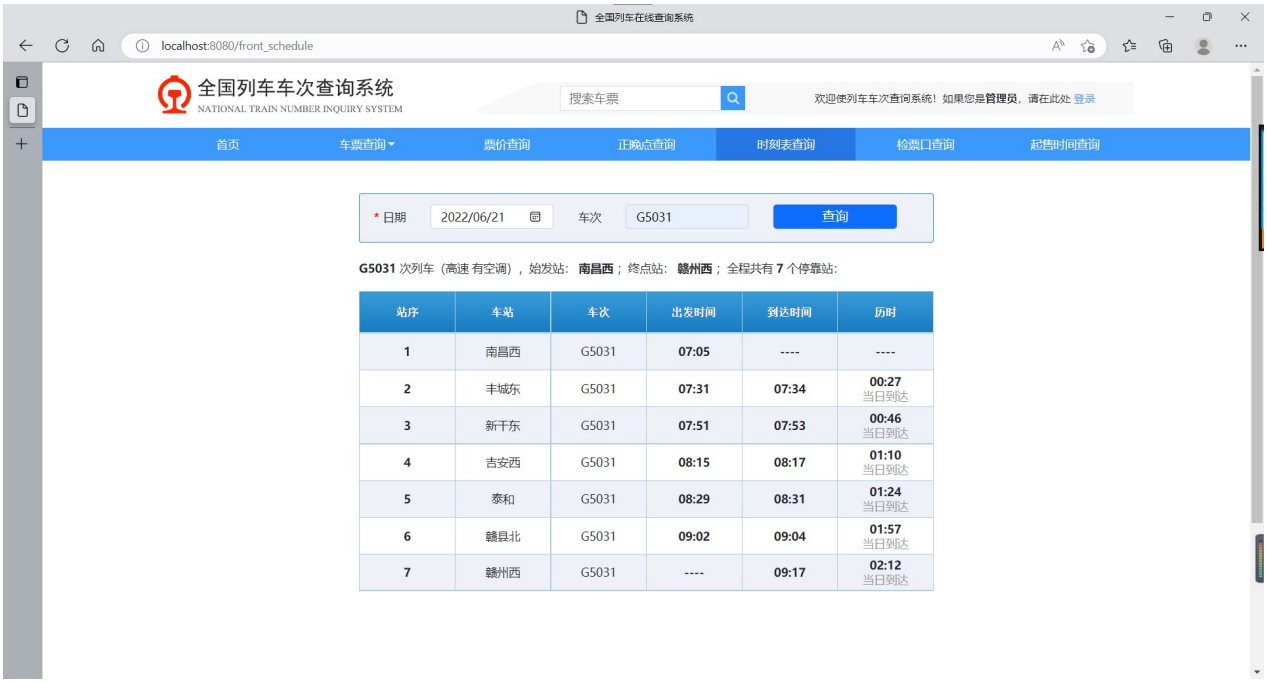
```
<select id="findTrainByBeginStationName" resultMap="train_model">
    select *
    from t_train
    where begin_station=#{stationName}
</select>
<select id="findTrainByEndStationName" resultMap="train_model">
    select *
    from t_train
    where end_station=#{stationName}
</select>
<select id="findTrainIdByBeginStationName" resultType="String">
    select train_id
    from t_train
    where begin_station=#{stationName}
```

```
</select>
<select id="findTrainIdByEndStationName" resultType="String">
    select train_id
    from t_train
    where end_station=#{stationName}
</select>
<select id="findTrainIdInChainByBeginStationName" resultType="String">
    select train_id
    from t_train_chain
    where station_name=#{stationName}
</select>
<select id="findTrainIdInChainByEndStationName" resultType="String">
    select train_id
    from t_train_chain
    where station_name=#{stationName}
</select>
<select id="findTrainChainByStation" resultMap="TrainChain_model">
    select *
    from t_train_chain
    where station_name=#{stationName}
</select>
<resultMap id="price_model" type="TicketMoney">
    <id property="trainId" column="train_id"></id>
    <result property="businessSeaMoney" column="business_seat_money"></result>
    <result property="firstSeatMoney" column="first_seat_money"></result>
    <result property="secondSeatMoney" column="second_seat_money"></result>
    <result property="softSleeperMoney" column="soft_sleeper_money"></result>
    <result property="hardSleeperMoney" column="hard_sleeper_money"></result>
    <result property="softSeatMoney" column="soft_seat_money"></result>
    <result property="hardSeatMoney" column="hard_seat_money"></result>
    <result property="noSeatMoney" column="no_seat_money"></result>
    <result property="otherMoney" column="other_money"></result>
</resultMap>
<select id="findPriceByTrainId" resultMap="price_model">
    select *
    from t_ticket_money
    where train_id=#{trainId}
</select>
```

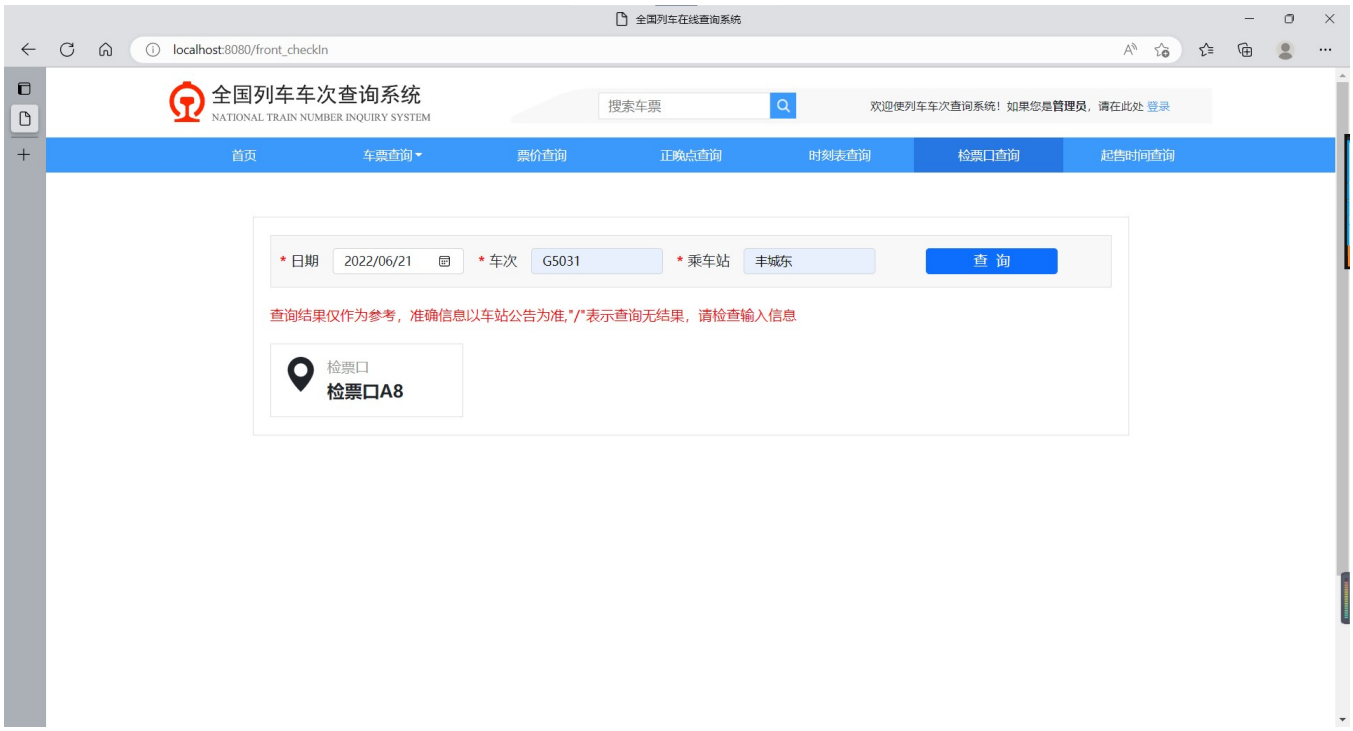
2）相关设计思路及实现成果

（1）正晚点查询：用户可以根据需要查询得车站和车次来进行对列车正晚点得查询，输入车站和车次后点击查询，显示正点到达或者晚点到达，实现结果如图：

（2）时刻表查询：用户可以根据需要查询得日期和车次来查询车站列车信息：车站、车次、出发时间和到达时间、历时，非常直观地显示始发站和终点站，以及全程的停靠站点。如图所示。



（3）检票口查询：用户可以根据输入需要查询的日期、车次以及乘车站来查询检票口信息，若无则显示“检票口/”，方便乘客快速找到对应的检票口进行检票乘车工作。如图所示。



（4）起售时间查询：用户可以根据输入需要查询的起售时间和起售车站来查询车站的车票起售时间，方便乘客购买车票。如图所示。

（5）车票价格查询：用户能根据出发地、目的地和出发时间查询列车票价。如图所示。

四、个人实训成果特点总结

特点 1：通过本门课掌握了一项重要的后端开发技术，看到了之前自己所 学的基础技术在实际中的作用。

特点 2：提高了自己解决问题的能力，这次课程中，有很多东西需要配置， 在配置这些程序的过程中难免遇到许多奇奇怪怪的错误,在解决这些问题的 过程中也是对自己解决问题能力的提升。

特点 3：学会了看文档，以前有问题总是百度，现在也慢慢开始从官方文 档中找解决办法。

五、个人实训成果缺陷总结

缺陷 1：对所学知识掌握的还是不够熟练。

产生原因：学习时间较短，缺少自学时间

改进措施：课程结束后，继续对 spring 知识进行学习

缺陷 2：敲代码速度较慢，每次老师布置作业，敲代码的速度要比其他同学慢一点。

产生原因：参与项目较少，对编程不够熟练

改进措施：在课余时间多花时间到编程上，提高编程速度

六、 心得与体会

这次的课程中，让我体会到了 spring 框架技术在项目构建中的巨大作用，比起我们之前课程设计的一些小项目，这次的项目更完善，也更趋近与一个真实项目，整体的代码也更多，但在使用 spring 框架技术后，给我的感觉是比之前做项目开发来的更加的容易，因为他的逻辑更为清楚，数据交互的逻辑也更清晰，比起之前所学的 jsp 等老 的技术，使用起来更加方便。这次的项目也让我有了不一样的编程体验，以往的比赛项目，我做的更多的都是一些前端的工作，这次的项目让我也体会了到后端开发者的快乐，而这些，让我对项目开发的流程体会更深， 对一个软件从无到有的过程有了更多的了解，也让我对未来的工作方向有了目标。

学生签名：郭瑄

2022 年 7 月 1 日