

# 数据库大作业报告

## 目录

---

- 目录
- 开发环境
- 题目要求
- 逻辑设计
  - ER图
  - 涉及到的数据库表
  - 实体间的关系
- 应用需求介绍
- 应用需求分析
- 实现思路
- 前后端功能实现&设计过程
  - 前端设计
  - 后端设计
- 数据库数据导入
- 实现成果
- 视频展示
- 总结
- 遇到问题&解决思路
- 心得体会

## 开发环境

---

编程语言: python3.6

框架: Django1.10.6

前端: HTML, CSS, JavaScript

数据库: sqlite3

软件: vscode

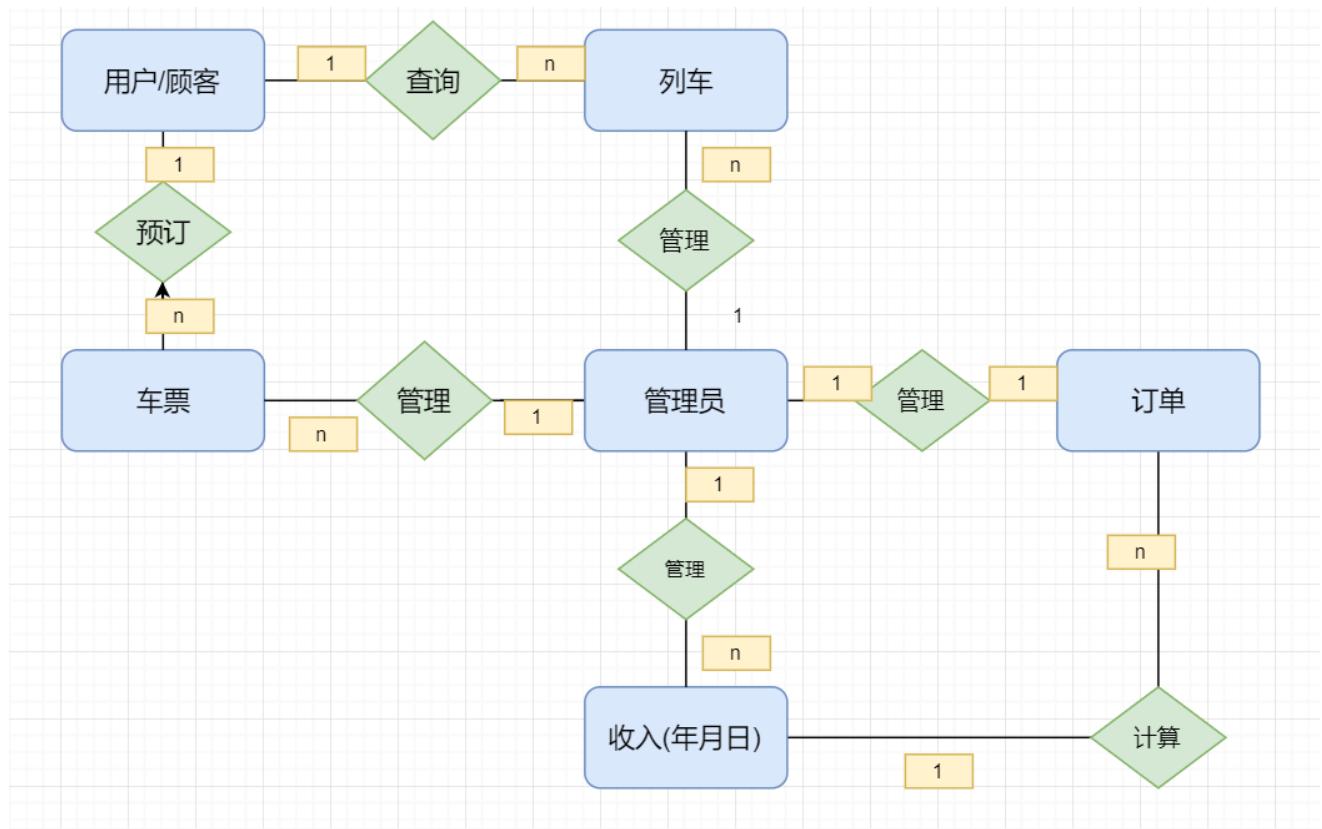
## 题目要求

---

- 设计和实现一个数据库应用系统
  - 实现一个railway ticket预订系统
- 至少包括五个以上实体; WEB界面
- 应用领域: 火车票预订或高铁票预订

# 逻辑设计

## ER图



注：图中矩形代表实体，其中1，n注明实体间的一对一或一对多关系。

## 涉及到的数据库表

- 用户/顾客信息表

key	datatype
id	interger
name	varchar(100)
password	varchar(100)
email	varchar(100)

注：这里的id为用户的id即passenger\_id

The screenshot shows a user management interface with the following details:

- Header:** WELCOME, ADMIN, VIEW SITE / CHANGE PASSWORD / LOG OUT.
- Search Bar:** Search (2 results (3 total)).
- Action Bar:** Action: -----, Go, 0 of 2 selected.
- Table Headers:** USERNAME, EMAIL ADDRESS, FIRST NAME, LAST NAME, STAFF STATUS.
- Data Rows:**
  - gzy, 893050234@qq.com, (empty), (empty), No
  - 郭梓煜, 893050234@qq.com, (empty), (empty), No
- Bottom Summary:** 2 users.
- Filter Sidebar:**
  - By staff status: All, Yes, No
  - By superuser status: All, Yes, No
  - By active: All, Yes, No
- Footer:** S 中 °, ⓘ, ⓘ, ⓘ, ⓘ, ⓘ, ⓘ, ⓘ.

这里展示两个用户。

- 管理员表

key	datatype
id	interger
name	varchar(100)
email	varchar(100)
status(active or not)	bool

注：这里的id为管理员的id即admin\_id

The screenshot shows a user management interface with the following details:

- Header:** WELCOME, ADMIN, VIEW SITE / CHANGE PASSWORD / LOG OUT.
- Search Bar:** Search (1 result (3 total)).
- Action Bar:** Action: -----, Go, 0 of 1 selected.
- Table Headers:** USERNAME, EMAIL ADDRESS, FIRST NAME, LAST NAME, STAFF STATUS.
- Data Rows:**
  - admin, 893050234@qq.com, (empty), (empty), Yes
- Bottom Summary:** 1 user.
- Filter Sidebar:**
  - By staff status: All, Yes, No
  - By superuser status: All, Yes, No
  - By active: All, Yes, No
- Footer:** S 中 °, ⓘ, ⓘ, ⓘ, ⓘ, ⓘ, ⓘ, ⓘ.

这里展示1个管理员。

- 个人预订车票表

key	datatype
id	interger
leave_city	varchar(100)
arrive_city	varchar(100)

key	datatype
leave_station	varchar(100)
arrive_station	varchar(100)
leave_time	datetime
arrive_time	datetime
price	interger

注：这里的id为列车id即train\_id

The screenshot shows a search interface for GZY Railway. The search parameters are: Departure Station (广州), Arrival Station (揭阳), and Date (2019/12/27). The results table has columns: 车次信息 (Train Information), 出发时间 (Departure Time), 到达时间 (Arrival Time), and 价格 (Price). The results are:

车次信息	出发时间	到达时间	价格
G1283	09:35 广州南站	13:35 潮汕站	满座率 2/100 ￥ 198.0 起 订票
G3623	09:55 广州东站	13:56 揭阳站	满座率 3/100 ￥ 198.0 起 订票
G2164	09:25 广州南站	10:32 潮汕站	满座率 3/100 ￥ 199.0 起 订票
G2378	09:23 广州东站	11:23 揭阳站	满座率 3/100 ￥ 250.0 起 订票

- 列车信息表

key	datatype
id	interger
leave_city	varchar(100)
arrive_city	varchar(100)
leave_station	varchar(100)
arrive_station	varchar(100)
leave_time	datetime
arrive_time	datetime
book_sum	interger
capacity	interger
price	interger

注：这里的id为列车id即train\_id

Action:	-----	Go	0 of 18 selected								
	NAME	LEAVE CITY	ARRIVE CITY	LEAVE STATION	ARRIVE STATION	LEAVE TIME	ARRIVE TIME	CAPACITY	PRICE	BOOK SUM	INCOME
<input type="checkbox"/>	G2378	广州	揭阳	广州东站	潮汕站	Jan. 1, 2020, 7:21 a.m.	Jan. 1, 2020, 9:21 a.m.	200	199.0	2	398.0
<input type="checkbox"/>	G1283	广州	揭阳	广州南站	潮汕站	Jan. 1, 2020, 10:20 a.m.	Jan. 1, 2020, 1:23 p.m.	200	200.0	2	400.0
<input type="checkbox"/>	D6947	广州	武汉	广州南站	武汉站	Dec. 27, 2019, 9:31 a.m.	Dec. 27, 2019, 2:17 p.m.	230	430.0	5	2150.0
<input type="checkbox"/>	G1260	广州	上海	广州南站	上海南站	Dec. 27, 2019, 6:13 p.m.	Dec. 27, 2019, 8:53 p.m.	500	400.0	6	2400.0
<input type="checkbox"/>	D3526	北京	天津	北京西站	天津站	Dec. 27, 2019, 6:09 a.m.	Dec. 27, 2019, 10:11 a.m.	100	350.0	2	700.0
<input type="checkbox"/>	Z3216	广州	昆明	广州南站	昆明南站	Dec. 27, 2019, 6:05 p.m.	Dec. 27, 2019, 8:05 p.m.	250	300.0	3	900.0
<input type="checkbox"/>	G4584	广州	石家庄	广州南站	石家庄北站	Dec. 27, 2019, 10:03 a.m.	Dec. 27, 2019, 12:13 p.m.	300	234.0	2	468.0
<input type="checkbox"/>	G3751	长沙	南宁	长沙南站	南宁西站	Dec. 27, 2019, 7:58 a.m.	Dec. 27, 2019, 9:52 a.m.	300	299.0	1	299.0
<input type="checkbox"/>	G3623	广州	揭阳	广州东站	揭阳站	Dec. 27, 2019, 9:55 a.m.	Dec. 27, 2019, 1:56 p.m.	199	198.0	4	795.0
<input type="checkbox"/>	Z3087	北京	济南	北京西站	济南北站	Dec. 27, 2019, 6:53 p.m.	Dec. 27, 2019, 10:53 p.m.	300	450.0	2	900.0
<input type="checkbox"/>	D3321	北京	上海	北京西站	上海南站	Dec. 27, 2019, 9:51 a.m.	Dec. 27, 2019, 1:51 p.m.	300	244.0	7	1708.0
<input type="checkbox"/>	D9242	北京	长沙	北京西站	长沙南站	Dec. 27, 2019, 6:34 a.m.	Dec. 27, 2019, 9:16 a.m.	300	400.0	2	800.0
<input type="checkbox"/>	Z3523	重庆	北京	重庆站	北京西站	Dec. 27, 2019, 7:33 a.m.	Dec. 27, 2019, 9:13 a.m.	250	300.0	4	1200.0
<input type="checkbox"/>	D3521	天津	北京	天津站	北京西站	Dec. 27, 2019, 9:31 a.m.	Dec. 27, 2019, 12:31 p.m.	300	400.0	1	400.0
<input type="checkbox"/>	D0241	上海	北京	北京西站	北京西站	Dec. 27, 2019, 1:27 a.m.	Dec. 27, 2019, 3:28 p.m.	200	300.0	0	0.0

- 订单表

key	datatype
id	interger
train_id	interger
passenger_id	interger
leave_station	varchar(100)
arrive_station	varchar(100)
price	interger

注：这里的id为订单id即order\_id

- 后台收入表

key	datatype
id	interger
year/month/week	interger
order_id	interger
income	interger

The screenshot shows a dashboard with three tables:

- 周收入 (Weekly Income):**

编号	周次	车次	收入
1	00	2	798.0
2	51	15	14265.0
- 月收入 (Monthly Income):**

编号	月份	车次	收入
1	01	2	798.0
2	12	15	14265.0
- 年收入 (Annual Income):**

编号	年份	车次	收入
1	2019	15	14265.0
2	2020	2	798.0

## 订单列表

编号	旅客	车次	路线	出发时间	车票
1	gzy	G2378	广州 → 揭阳	Dec. 27, 2019, 9:23 a.m.	250.0
2	郭梓煜	G1283	广州 → 揭阳	Dec. 27, 2019, 9:35 a.m.	198.0
3	gzy	G3623	广州 → 揭阳	Dec. 27, 2019, 9:55 a.m.	198.0

## 实体间的关系

订单表中train\_id为参照列车信息表的外键，passenger\_id为参照用户表的外键。

后台收入表中order\_id为参照订单表的外键。

其中具体的关系可见上面的ER图，此处不多做介绍。

## 应用需求介绍

要求具备如下基本功能：

- 用户/顾客方面：
  - 可以根据出发地，目的地，时间获取列车信息；
  - 可以进行订票，取消订票，查看个人中心，订票记录；
- 预订系统方面：
  - 列车基本信息的管理；
  - 顾客预定车票、取消预约、付款取票、退票的管理；
  - 后台查询列车信息、列车预定情况、顾客信息，计算列车满座率。
  - 统计每周、每月，每年营业收入情况。

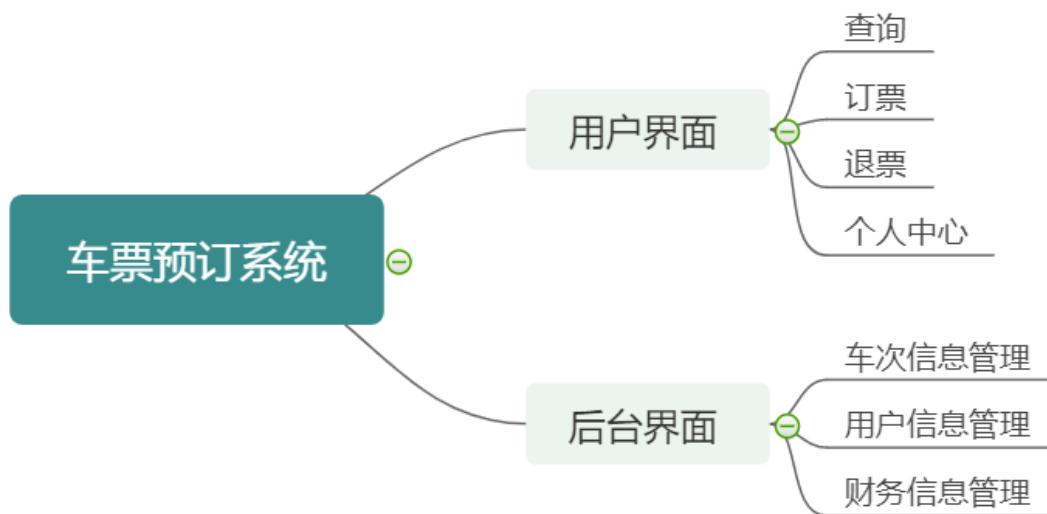
## 应用需求分析

### • 具体需求

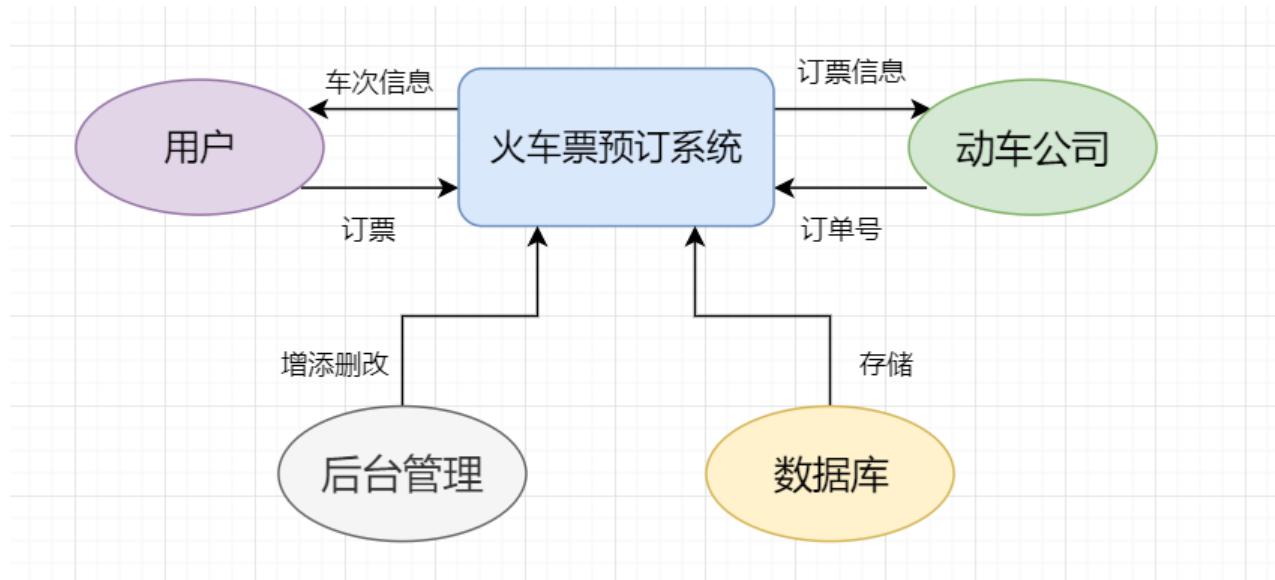
为方便用户出行，某火车运营公司（GZY railway）拟开发一个火车预定系统。用户可通过网上订票平台查询公司列车情况，通过输入出发地、目的地、出发时间等信息系统为用户安排列车，用户可根据自身需要，按照出发时间和机票价位来选择列车。订票成功之后，系统为用户生成订单信息，用户可以再个人信息页面查看自己的订票信息，并且可以向系统提出退票要求，系统针对具体情况计算手续费后进行相应退票处理。

### • 功能分析

- 用户界面
  - 查询：用户对列车信息进行查询操作；
  - 排序：用户根据自己的需求对查询结果进行排序筛选；
  - 订票：对用户订票需求进去处理并记录用户预定信息和更新数据库信息；
  - 退票：对用户退票需求进行处理并更新数据库；
  - 个人中心：用户查看自己的个人订票信息
  - 帮助：提供系统使用帮助文档；
  - 退出：关闭当前页面。
- 后台界面
  - 列车信息管理：可对列车信息进行增删改查操作；
  - 用户信息管理：可对用户信息进行增删改查操作；
  - 财务信息管理：可以统计动车公司每周、每月，每年营业收入情况。
  - 帮助：提供系统使用帮助文档；
  - 退出：关闭当前页面。
- 系统主功能图



- 系统数据流通图



## 实现思路

1. 首先，确定好实现所用的框架及环境。  
由于之前并未过多接触，所以先按照网上的教程学习基础实现。  
[学习教程](#)
2. 其次，在教程的基础上根据数据库设计来进行进一步实现。
3. 然后，将获取到的数据插入数据库中。
4. 实现后，在自己的电脑进行测试。

## 前后端功能实现&设计过程

我实现的这个火车票预订系统采用以 [Python3 + Django1.10.6 + Sqlite3](#) 为基础的设计方法，后台功能写在 views.py 文件中，通过 urls.py 文件中的正则匹配，调用 views.py 中写好的各种功能，以及 templates 文件夹下的各个 html 网页。

以下分前端设计和后端设计来介绍实现过程：

### 前端设计

前端的设计涉及到了html/css/JavaSricpt的使用，由于我对这些之前并没有过多的了解，所以在网上学习了一些[教程](#)后才开始自己的设计。

由于实现的网页较多，此处以最初始的欢迎页面的index.html为例介绍

```

<head>
    <meta charset="UTF-8">
    <title>火车票预订系统</title>
    <link rel="stylesheet"
        href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css">
        <script>

```

```
src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.0/jquery.min.js">
</script>
<script
src="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js">
</script>
{% load staticfiles %}
<style>
body {
    /*background: url("../static/img/flight-bg.jpg");*/
    background: url("{% static 'img/railway.jpg' %}");
    background-size: cover; /*自适应浏览器*/
}
</style>
</head>
```

这里我直接调用了现有的框架来实现，其中使用到了google所以如果没有科学上网可能打开速度会受影响，当然也可以注释掉，其实并无多大影响。

而网页的背景则可用background: url("{% static 'img/railway.jpg' %}");来指明，Django将会自动在该路径下的static文件夹进行搜索img文件夹的railway.jpg，然后加载到页面即可。

```
<div class="container-fluid" style="position: absolute; margin-top: 10%; width: 35%; left: 55%">
    <h1 style="color: white">Where would you go ?</h1>
    <br>
    <form class="form-horizontal" role="form" action="{% url 'booksystem:result' %}" method="post">
        {% csrf_token %}
        <div class="input-group">
            <label for="leave_city" class="input-group-addon">出发城市</label>
        <!--input 必须要指定name-->
            <input id="leave_city" type="text" name="leave_city" class="form-control" placeholder="City">
        </div>
        <br>
        <div class="input-group">
            <label for="arrive_city" class="input-group-addon">到达城市</label>
            <input id="arrive_city" type="text" name="arrive_city" class="form-control" placeholder="City">
        </div>
        <br>
        <div class="input-group">
            <label for="leave_date" class="input-group-addon">出发日期</label>
            <input id="leave_date" type="date" name="leave_date" class="form-control" placeholder="Date">
        </div>
        <br>
        <center>
            <button type="submit" class="btn btn-success">Let's Go</button>
        </center>
    </form>
</div>
```

```
</form>
</div>
```

以上代码则显示了文字以及输入框，按钮等部件，还有控制输入框的位置所在。前端的设计就介绍到这里，详见提交代码中的templates文件夹。

## 后端设计

后端的设计以python以及Django为主，依托于之前设计的功能，来进行后端的实现。

- 列车表的创建 在models.py中以类的形式创建表

```
class railway(models.Model):
    user = models.ManyToManyField(User, default=1) # 有了这个字段之后，默认的后台添加失效，必须要自定义Form，除去这个字段
    name = models.CharField(max_length=100) # 班次
    leave_city = models.CharField(max_length=100, null=True) # 离开城市
    arrive_city = models.CharField(max_length=100, null=True) # 到达城市
    leave_station = models.CharField(max_length=100, null=True) # 离开的车站
    arrive_station = models.CharField(max_length=100, null=True) # 到达的车站
    leave_time = models.DateTimeField(null=True) # DateTimeField包括了DateField信息，并且添加了时间
    arrive_time = models.DateTimeField(null=True)
    capacity = models.IntegerField(default=0, null=True) # 座位总数
    price = models.FloatField(default=0, null=True) # 价格
    book_sum = models.IntegerField(default=0, null=True) # 订票总人数
    income = models.FloatField(default=0, null=True) # 收入

    def __str__(self):
        return self.name
```

- 自定义表单管理

```
class railwayAdmin(admin.ModelAdmin):
    list_display = (
        'name', 'leave_city', 'arrive_city', 'leave_station',
        'arrive_station', 'leave_time', 'arrive_time', 'capacity',
        'price', 'book_sum', 'income')
    form = railwayForm
```

- 网址的正则匹配

```
urlpatterns = [
    # 注册与登录
    url(r'^register/$', views.register, name='register'), # 注册
    url(r'^login_user/$', views.login_user, name='login_user'), # 登入
```

```

url(r'^logout_user/$', views.logout_user, name='logout_user'), # 登出

# 主要页面
url(r'^$', views.index, name='index'), # 欢迎页面
url(r'^result/$', views.result, name='result'), # 搜索结果
url(r'^user_info/$', views.user_info, name='user_info'), # 用户信息
url(r'^book/railway/(?P<railway_id>[0-9]+)/$', views.book_ticket,
name='book_ticket'), # 订票
url(r'^refund/railway/(?P<railway_id>[0-9]+)/$', views.refund_ticket,
name='refund_ticket'), # 退票
]

```

功能方面主要体现在views.py

- 订票功能

在用户订票的过程中，首先判定用户是否登录，如果没有登录，就加载登录页面；如果用户已经登陆了，通过前端用户选择的车次传入 railway.id，然后判断如果用户已经订过这次列车，就反馈冲突信息。如果没有，订票成功，数据库更新，显示订票成功页面。

```

def book_ticket(request, railway_id):
    if not request.user.is_authenticated(): # 如果没登录就render登录页面
        return render(request, 'booksystem/login.html')
    else:
        global Railway
        railway = Railway.objects.get(pk=railway_id)
        # 查看乘客已经订购的railways
        booked_railways = Railway.objects.filter(user=request.user) # 返回
QuerySet

        if railway in booked_railways:
            return render(request, 'booksystem/book_conflict.html')

        # book_railway.html 点确认之后, request为 POST 方法, 虽然没有传递什么值,
但是传递了 POST 信号
        # 确认订票, railway数据库改变

        # 验证一下, 同样的机票只能订一次
        if request.method == 'POST':
            if railway.capacity > 0:
                railway.book_sum += 1
                railway.capacity -= 1
                railway.income += railway.price
                railway.user.add(request.user)
                railway.save() # 一定要记着save
            # 传递更改之后的票务信息
            context = {
                'railway': railway,
                'username': request.user.username
            }
            return render(request, 'booksystem/book_railway.html', context)

```

- 查询功能

- 前端表单接收用户传入的出发地、目的地和出发时间，然后在列车数据库中寻找满足条件的列车，分两步：
  - 寻找出发地和目的地相同的列车；
  - 寻找列车出发日期与旅客出发日期相同的列车。
- 为了给用户良好的体验，满足条件的列车信息按照不同的 key 值（出发时间、到达时间、价格）进行升序排列。
  - 升序排列的实现只需使用sorted函数对不同的key进行排序即可。

```
def result(request):  
    if request.method == 'POST':  
        form = PassengerInfoForm(request.POST) # 绑定数据至表单  
        if form.is_valid():  
            passenger_lcity = form.cleaned_data.get('leave_city')  
            passenger_acity = form.cleaned_data.get('arrive_city')  
            passenger_ldate = form.cleaned_data.get('leave_date')  
  
            # 全设为aware比较  
            passenger_ltime = datetime.datetime.combine(passenger_ldate,  
datetime.time())  
            print(passenger_ltime)  
  
            # filter 可用车次  
            global Railway  
            all_railways = Railway.objects.filter(leave_city=passenger_lcity,  
arrive_city=passenger_acity)  
            usable_railways = []  
            for railway in all_railways: # off-set aware  
                railway.leave_time = railway.leave_time.replace(tzinfo=None)  
# replace方法必须要赋值。。笑哭  
            if railway.leave_time.date() == passenger_ltime.date(): # 只查  
找当天的车次  
                usable_railways.append(railway)  
  
            # 按不同的key排序  
            usable_railways_by_ltime = sorted(usable_railways,  
key=attrgetter('leave_time')) # 出发时间从早到晚  
            usable_railways_by_atime = sorted(usable_railways,  
key=attrgetter('arrive_time'))  
            usable_railways_by_price = sorted(usable_railways,  
key=attrgetter('price')) # 价格从低到高  
  
            # 转换时间格式  
            time_format = '%H:%M'  
  
            # 虽然只转换了一个list，其实所有的都转换了  
            for railway in usable_railways_by_price:  
                railway.leave_time = railway.leave_time.strftime(time_format)  
# 转成了str
```

```

        railway.arrive_time =
railway.arrive_time.strftime(time_format)

        # 决定 search_head , search_failure 是否显示
dis_search_head = 'block'
dis_search_failure = 'none'
if len(usable_railways_by_price) == 0:
    dis_search_head = 'none'
    dis_search_failure = 'block'
context = {
    # 搜索框数据
    'leave_city': passenger_lcity,
    'arrive_city': passenger_acity,
    'leave_date': str(passenger_ldate),
    # 搜索结果
    'usable_railways_by_ltime': usable_railways_by_ltime,
    'usable_railways_by_atime': usable_railways_by_atime,
    'usable_railways_by_price': usable_railways_by_price,
    # 标记
    'dis_search_head': dis_search_head,
    'dis_search_failure': dis_search_failure
}
if request.user.is_authenticated():
    context['username'] = request.user.username
return render(request, 'booksystem/result.html', context) # 最前面
如果加了/就变成根目录了, url 错误
else:
    return render(request, 'booksystem/index.html') # 在index界面提交的
表单无效, 就保持在index界面
else:
    context = {
        'dis_search_head': 'none',
        'dis_search_failure': 'none'
    }
return render(request, 'booksystem/result.html', context)

```

- 查询个人信息功能

由于管理员和用户共用一个个人信息窗口, 所以需要进行判断登录的身份。如果登录的用户是管理员, 则加载管理页面, 如果是普通用户, 则加载用户个人的订单页面。

```

def user_info(request):
global railway
if request.user.is_authenticated():
    # 如果用户是管理员, render公司车次收入统计信息页面 admin_finance
    print(request.user.id)
    if request.user.id == ADMIN_ID:
        context = admin_finance(request) # 获取要传入前端的数据
        return render(request, 'booksystem/admin_finance.html', context)
    # 如果用户是普通用户, render用户的机票信息 user_info
    else:
        booked_railways = Railway.objects.filter(user=request.user) # 从

```

```

booksystem_railway_user 表过滤出该用户订的车次
context = {
    'booked_railways': booked_railways,
    'username': request.user.username, # 导航栏信息更新
}
return render(request, 'booksystem/user_info.html', context)
return render(request, 'booksystem/login.html') # 用户如果没登录, render登录页面

```

- 退票功能

退票时需要更新数据库，更新车次的 (capacity, book\_sum, income) 字段，并且在订单表中删除这个订单

```

# 退票
def refund_ticket(request, railway_id):
    global Railway
    railway = Railway.objects.get(pk=railway_id)
    railway.book_sum -= 1
    railway.capacity += 1
    railway.income -= railway.price
    railway.user.remove(request.user)
    railway.save()
    return HttpResponseRedirect('/booksystem/user_info')

```

- 财务统计功能

```

def admin_finance(request):
    global Railway
    all_railways = Railway.objects.all()
    all_railways = sorted(all_railways, key=attrgetter('leave_time')) # 将所有车次按照出发时间排序

    # 将车次每天的输入打上不同的时间标签 [周, 月, 日]
    week_day_incomes = []
    month_day_incomes = []
    year_day_incomes = []

    # 用set存储所有的 周, 月, 年
    week_set = set()
    month_set = set()
    year_set = set()
    for railway in all_railways:
        if railway.income > 0: # 只统计有收入的车次
            # 打上周标签
            this_week = railway.leave_time.strftime('%W') # datetime获取周
            week_day_incomes.append((this_week, railway.income)) # 添加元组
            (week, income)
            week_set.add(this_week)
            # 打上月标签

```

```

        this_month = railway.leave_time.strftime('%m') # datetime获取月
        month_day_incomes.append((this_month, railway.income)) # 添加元组
        (month, income)
        month_set.add(this_month)
        # 打上年标签
        this_year = railway.leave_time.strftime('%Y') # datetime获取年
        year_day_incomes.append((this_year, railway.income)) # 添加元组
        (year, income)
        year_set.add(this_year)

    # 存储每周收入
    # 将每周的收入用 IncomeMetric 类型存储在 week_incomes List 中
    week_incomes = []
    for week in week_set:
        income = sum(x[1] for x in week_day_incomes if x[0] == week) # 同周次
        的income求和
        railway_sum = sum(1 for x in week_day_incomes if x[0] == week) # 同周
        次的车次总数目
        week_income = IncomeMetric(week, railway_sum, income) # 将数据存储到
        IncomeMetric类中, 方便jinja语法
        week_incomes.append(week_income)
    week_incomes = sorted(week_incomes, key=attrgetter('metric')) # 将List类型
    的 week_incomes 按周次升序排列

    # 存储每月收入
    # 将每月的收入用 IncomeMetric 类型存储在 month_incomes List 中
    month_incomes = []
    for month in month_set:
        income = sum(x[1] for x in month_day_incomes if x[0] == month)
        railway_sum = sum(1 for x in month_day_incomes if x[0] == month)
        month_income = IncomeMetric(month, railway_sum, income)
        month_incomes.append(month_income)
    month_incomes = sorted(month_incomes, key=attrgetter('metric')) # 将List类
    型的 month_incomes 按月份升序排列

    # 存储每年收入
    # 将每年的收入用 IncomeMetric 类型存储在 year_incomes List 中
    year_incomes = []
    for year in year_set:
        income = sum(x[1] for x in year_day_incomes if x[0] == year)
        railway_sum = sum(1 for x in year_day_incomes if x[0] == year)
        year_income = IncomeMetric(year, railway_sum, income)
        year_incomes.append(year_income)
    year_incomes = sorted(year_incomes, key=attrgetter('metric')) # 将List类型
    的 year_incomes 按年份升序排列

    # 存储order信息
    passengers = User.objects.exclude(pk=1) # 去掉管理员
    order_set = set()
    for p in passengers:
        railways = Railway.objects.filter(user=p)
        for f in railways:
            route = f.leave_city + ' → ' + f.arrive_city
            order = Order(p.username, f.name, route, f.leave_time, f.price)

```

```
order_set.add(order)

# 信息传给前端
context = {
    'week_incomes': week_incomes,
    'month_incomes': month_incomes,
    'year_incomes': year_incomes,
    'order_set': order_set
}
print(context)
return context
```

- 用户信息管理 用户继承了 django.contrib.auth.User 类，我们只需要自定义用户表单需要输入的对象，其他默认生成的字段不用考虑。

```
class UserForm(forms.ModelForm):
    password = forms.CharField(widget=forms.PasswordInput)

    class Meta:
        model = User
        fields = ['username', 'email', 'password']
```

## 数据库数据导入

数据库的导入并不难，根据队友搜集到的数据，以空格隔开。然后参照[网上的教程](#)批量地导入数据。

示例代码如下：

```
import django
if django.VERSION >= (1, 7):#自动判断版本
    django.setup()

def main():
    from blog.models import Blog
    f = open('DBMS_source.txt')
    for line in f:
        title,content = line.split(' ')
        Blog.objects.create(title=title,content=content,...)
    f.close()

if __name__ == "__main__":
    main()
    print('导入成功!')
```

最终数据在db.sqlite3中存储形式如下：

以车次信息表为例

#	id	name	leave_city	arrive_city	leave_station	arrive_station	leave_time	arrive_time	capacity	price	book_sum	income
1	1	G1283	广州	揭阳	广州南站	潮汕站	2019-12-27 09:35:14	2019-12-27 13:35:20	198	198.0	2	396.0
2	2	G2378	广州	揭阳	广州东站	揭阳站	2019-12-27 09:23:22	2019-12-27 11:23:28	200	250.0	2	500.0
3	3	G2164	广州	揭阳	广州南站	潮汕站	2019-12-27 09:25:47	2019-12-27 10:32:57	200	199.0	3	597.0
4	4	D9241	长沙	北京	长沙南站	北京西站	2019-12-27 13:27:47	2019-12-27 15:28:11	200	200.0	0	0.0
5	5	D3521	天津	北京	天津站	北京西站	2019-12-27 09:31:19	2019-12-27 12:31:30	300	400.0	1	400.0
6	6	Z3523	重庆	北京	重庆站	北京西站	2019-12-27 07:33:11	2019-12-27 09:13:23	250	300.0	4	1200.0
7	7	D9242	北京	长沙	北京西站	长沙南站	2019-12-27 06:34:18	2019-12-27 09:16:35	300	400.0	2	800.0
8	8	D3321	北京	上海	北京西站	上海南站	2019-12-27 09:51:01	2019-12-27 13:51:07	300	244.0	7	1708.0
9	9	Z3087	北京	济南	北京西站	济南北站	2019-12-27 18:53:39	2019-12-27 22:53:46	300	450.0	2	900.0
10	10	G3623	广州	揭阳	广州东站	揭阳站	2019-12-27 09:55:56	2019-12-27 13:56:02	199	198.0	4	795.0
11	11	G3751	长沙	南宁	长沙南站	南宁西站	2019-12-27 07:58:14	2019-12-27 09:52:29	300	299.0	1	299.0
12	12	G4584	广州	石家庄	广州南站	石家庄北站	2019-12-27 10:03:24	2019-12-27 12:13:41	300	234.0	2	468.0
13	13	Z3216	广州	昆明	广州南站	昆明南站	2019-12-27 18:05:42	2019-12-27 20:05:54	250	300.0	3	900.0
14	14	D3526	北京	天津	北京西站	天津站	2019-12-27 06:09:24	2019-12-27 10:11:34	100	350.0	2	700.0
15	15	G1260	广州	上海	广州南站	上海南站	2019-12-27 18:13:16	2019-12-27 20:53:26	500	400.0	6	2400.0
16	16	D6947	广州	武汉	广州南站	武汉站	2019-12-27 09:31:19	2019-12-27 14:17:34	230	430.0	5	2150.0
17	17	G1283	广州	揭阳	广州南站	潮汕站	2020-01-01 10:20:11	2020-01-01 13:23:21	200	200.0	2	400.0
18	18	G2378	广州	揭阳	广州东站	潮汕站	2020-01-01 07:21:08	2020-01-01 09:21:33	200	199.0	2	398.0
19	19	G2632	广州	揭阳	广州东站	揭阳站	2019-12-27 21:30:53	2019-12-27 23:11:01	200	200.0	2	400.0

将数据库与系统连接

```
# Database
# https://docs.djangoproject.com/en/1.10/ref/settings/#databases
```

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),
    }
}
```

- 同步数据库步骤如下：

- 生成迁移文件

```
python3 manage.py makemigrations
```

- 应用到数据库

```
python3 manage.py migrate
```

## 实现成果

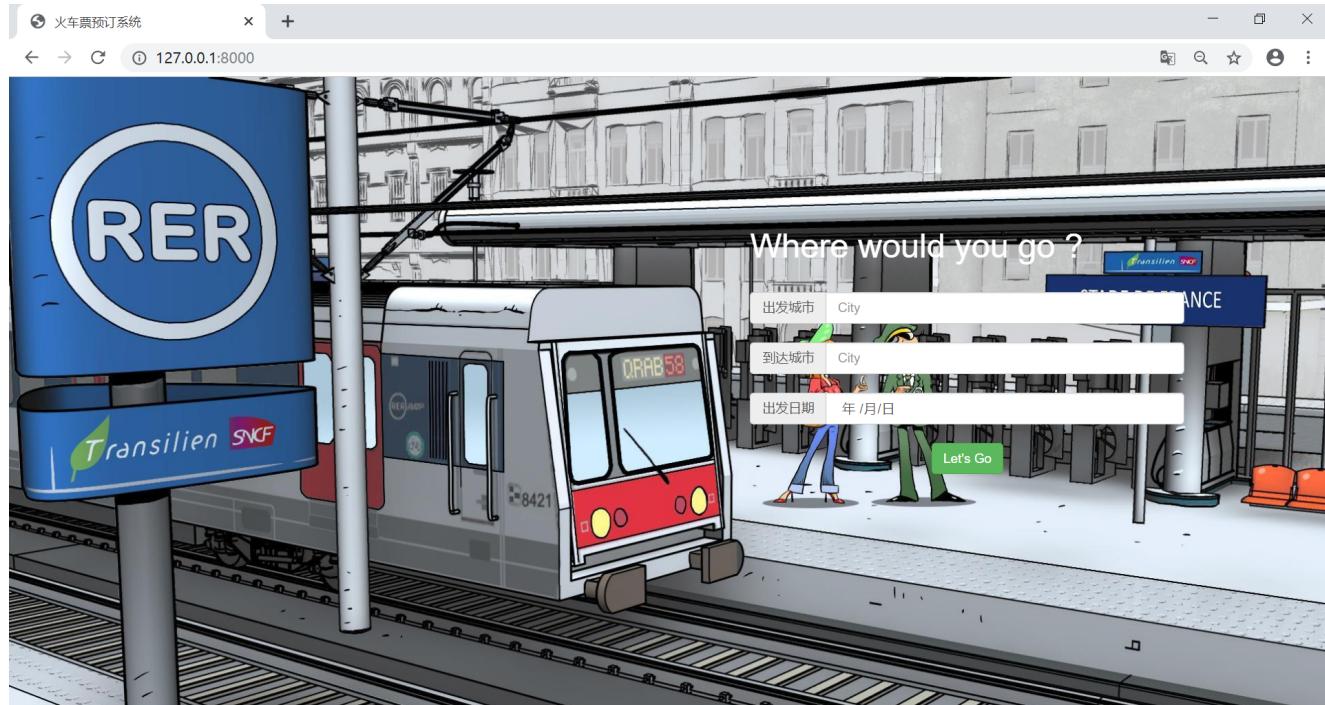
在命令行中cd到RAILWAY\_TICKET\_BOOKSYSTEM目录下，运行

```
python3 manage.py runserver
```

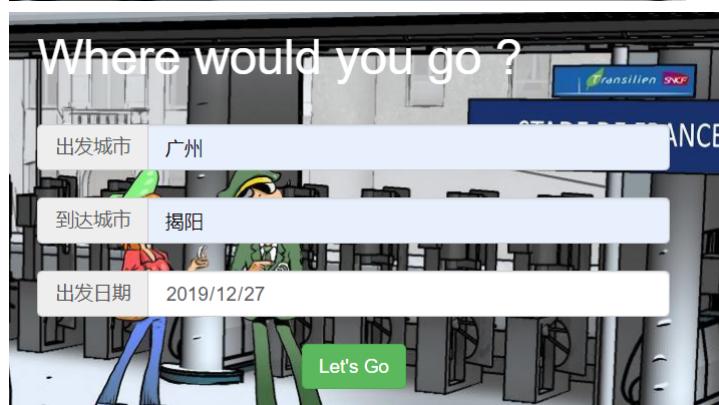
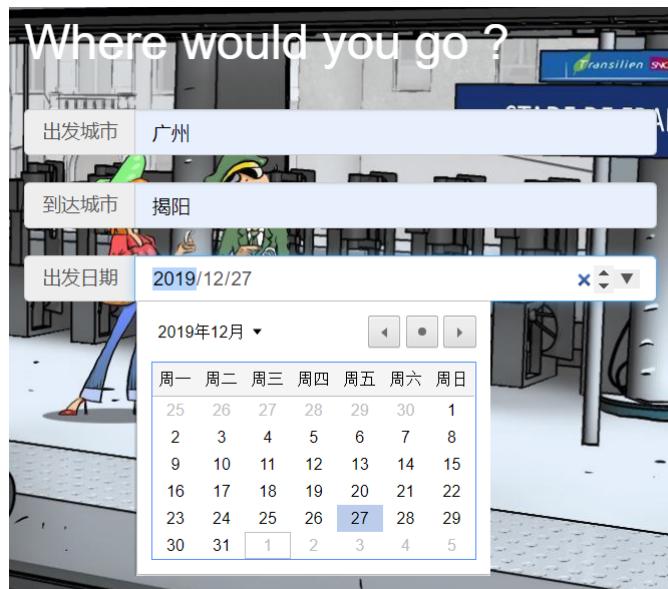
(前提是python3中已配置好了Django1.10.6, pytz等环境)

然后打开浏览器访问<http://127.0.0.1:8000>/将会来到预订系统的欢迎界面!

如图:

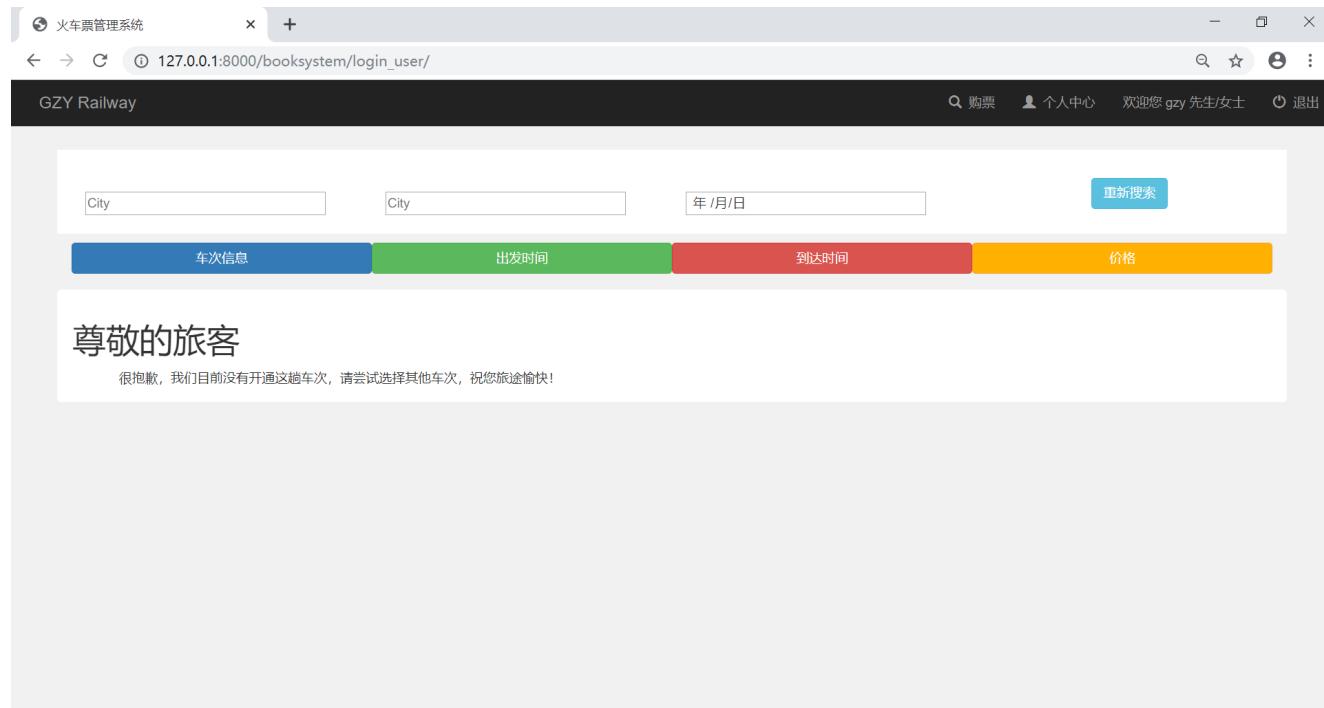


在输入框输入出发城市，到达城市以及日期



点击let's go按钮，即会跳转到查询结果页面。

若没有往返两地的列车，将显示以下页面



若有往返两地的列车，则显示以下页面

默认的车票信息按照价格升序排列，在价格旁边还有满座率等信息。

A screenshot of a web browser window titled "火车票管理系统" (Train Ticket Management System). The URL is 127.0.0.1:8000/booksystem/result/. The page header includes "GZY Railway", "购票" (Buy), "个人中心" (Personal Center), "欢迎您 先生/女士" (Welcome Mr./Ms.), and "退出" (Logout). Below the header is a search bar with three input fields: "广州" (Guangzhou) (出发地), "揭阳" (Jieyang) (目的地), and "2019/12/27" (Date). To the right of the date field is a blue "重新搜索" (Reset Search) button. Below the search bar is a horizontal navigation bar with four tabs: "车次信息" (Train Information) in blue, "出发时间" (Departure Time) in green, "到达时间" (Arrival Time) in red, and "价格" (Price) in yellow. The main content area displays a table of five train results:

车次	发车时间	到达时间	票价	满座率
G1283	09:35 广州南站	13:35 潮汕站	¥ 198.0 起	1/100
G3623	09:55 广州东站	13:56 揭阳站	¥ 198.0 起	4/100
G2164	09:25 广州南站	10:32 潮汕站	¥ 199.0 起	3/100
G2632	21:30 广州东站	23:11 揭阳站	¥ 200.0 起	2/100
G2378	09:23 广州东站	11:23 揭阳站	¥ 250.0 起	3/100

用鼠标点击绿色按钮出发时间，红色按钮到达时间，或者橙色按钮价格将会按照该列属性从小到大进行排序。

如图按照出发时间排序。

The screenshot shows a search interface for a train booking system. The search parameters are: Departure Station (广州), Arrival Station (揭阳), and Date (2019/12/27). The results are listed in a table with columns: Train Number (车次信息), Departure Time (出发时间), Arrival Time (到达时间), Occupancy Rate (满座率), and Price (价格). The results are as follows:

车次信息	出发时间	到达时间	满座率	价格
G2378	09:23 广州东站	11:23 揭阳站	3/100	¥ 250.0 起 <button>订票</button>
G2164	09:25 广州南站	10:32 潮汕站	3/100	¥ 199.0 起 <button>订票</button>
G1283	09:35 广州南站	13:35 潮汕站	3/100	¥ 198.0 起 <button>订票</button>
G3623	09:55 广州东站	13:56 揭阳站	4/100	¥ 198.0 起 <button>订票</button>
G2632	21:30	23:11	2/100	¥ 200.0 起 <button>订票</button>

可以看到右上角显示欢迎先生/女士，没有用户名，说明尚未登录。  
所以点击订票会直接跳转到登录页面。

The screenshot shows a login form titled "登录账号" (Login Account). It contains two input fields: "Username:" and "Password:", both with placeholder text. Below the fields is a green "Submit" button. At the bottom of the form, there is a link: "Don't have an account? Click here to register."

若还没有账号，点击下方的click here进行注册。

火车票管理系统

GZY Railway

注册账号

Username:

Email address:

Password:

Submit

Already have an account? [Click here to log in.](#)

输入username等信息点击submit即可进行注册。  
注册成功后即会直接跳转回刚刚的查询车票界面。

火车票管理系统

GZY Railway

广州 揭阳 2019/12/27 重新搜索

车次信息	出发时间	到达时间	票价	操作
G1283	09:35 广州南站	13:35 潮汕站	满座率 4/100 ￥ 198.0 起	<a href="#">订票</a>
G3623	09:55 广州东站	13:56 揭阳站	满座率 4/100 ￥ 198.0 起	<a href="#">订票</a>
G2164	09:25 广州南站	10:32 潮汕站	满座率 3/100 ￥ 199.0 起	<a href="#">订票</a>
G2632	21:30 广州东站	23:11 揭阳站	满座率 2/100 ￥ 200.0 起	<a href="#">订票</a>
G1243	12:39 广州南站	15:39 潮汕站	满座率 50/100 ￥ 200.0 起	<a href="#">订票</a>

再次点击订票，

The screenshot shows a web-based train ticket booking system. At the top, there's a header bar with the text "火车票管理系统" (Train Ticket Management System), a search icon, and user information "GZY Railway" and "欢迎您 gzy 先生/女士". Below the header is a main content area with a title "尊敬的旅客" (Dear Passenger). A message states "您选择了 G1283 次车次, 该车次还有余票 199 张。" (You have selected train G1283, which still has 199 available tickets). A table displays the following information:

车次	出发车站	到达车站	出发时间	到达时间	价格
G1283	广州	揭阳	Dec. 27, 2019, 9:35 a.m.	Dec. 27, 2019, 1:35 p.m.	¥198.0

At the bottom right of the table are two buttons: "确认" (Confirm) in blue and "返回" (Return) in red.

点击确认即可进行订票，并弹出订票成功信息。

This screenshot shows the same booking interface after a confirmation action. The message "您已成功订购了此次车次, 请保管好车次信息, 祝您旅途愉快!" (Your order for this train has been successfully placed. Please keep your train number for travel.祝您旅途愉快!) appears in green at the bottom of the main content area.

此时，点击右上角的个人中心可以查看自己的订票信息。

The screenshot shows a web browser window for a train ticket booking system. The URL is 127.0.0.1:8000/booksystem/user\_info/. The page title is "GZY Railway". The main content is a table listing three train trips:

车次架次	出发车站	到达车站	出发时间	到达时间	价格
G2378	广州广州东站	揭阳	Dec. 27, 2019, 9:23 a.m.	Dec. 27, 2019, 11:23 a.m.	¥ 250.0 <span style="background-color: red; color: white; padding: 2px;">退票</span>
G3623	广州广州东站	揭阳	Dec. 27, 2019, 9:55 a.m.	Dec. 27, 2019, 1:56 p.m.	¥ 198.0 <span style="background-color: red; color: white; padding: 2px;">退票</span>
G1283	广州广州南站	揭阳	Dec. 27, 2019, 9:35 a.m.	Dec. 27, 2019, 1:35 p.m.	¥ 198.0 <span style="background-color: red; color: white; padding: 2px;">退票</span>

## 点击退票

The screenshot shows the same web browser window as the previous one, but with a modal dialog box overlaid. The dialog box contains the text "尊敬的顾客, 您确定要退票吗?" (Customer, are you sure you want to cancel the ticket?). It has two buttons: "确定" (Confirm) and "取消" (Cancel). The background of the main content area is dimmed.

## 点击确定

该条订票记录将直接消失

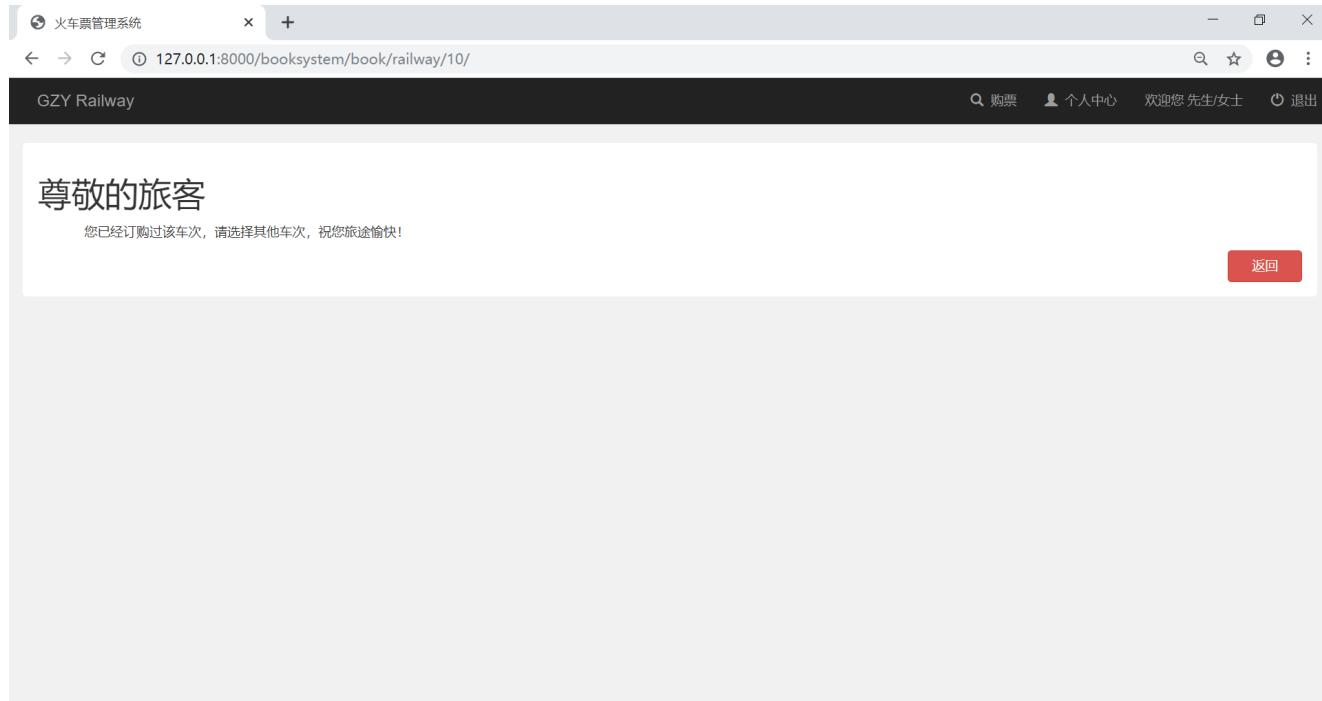
车次架次	出发车站	到达车站	出发时间	到达时间	价格
G3623	广州广州东站	揭阳	Dec. 27, 2019, 9:55 a.m.	Dec. 27, 2019, 1:56 p.m.	¥ 198.0 <span style="background-color: red; color: white; padding: 2px;">退票</span>
G1283	广州广州南站	揭阳	Dec. 27, 2019, 9:35 a.m.	Dec. 27, 2019, 1:35 p.m.	¥ 198.0 <span style="background-color: red; color: white; padding: 2px;">退票</span>

点击右上角的购票将跳转回查询页面

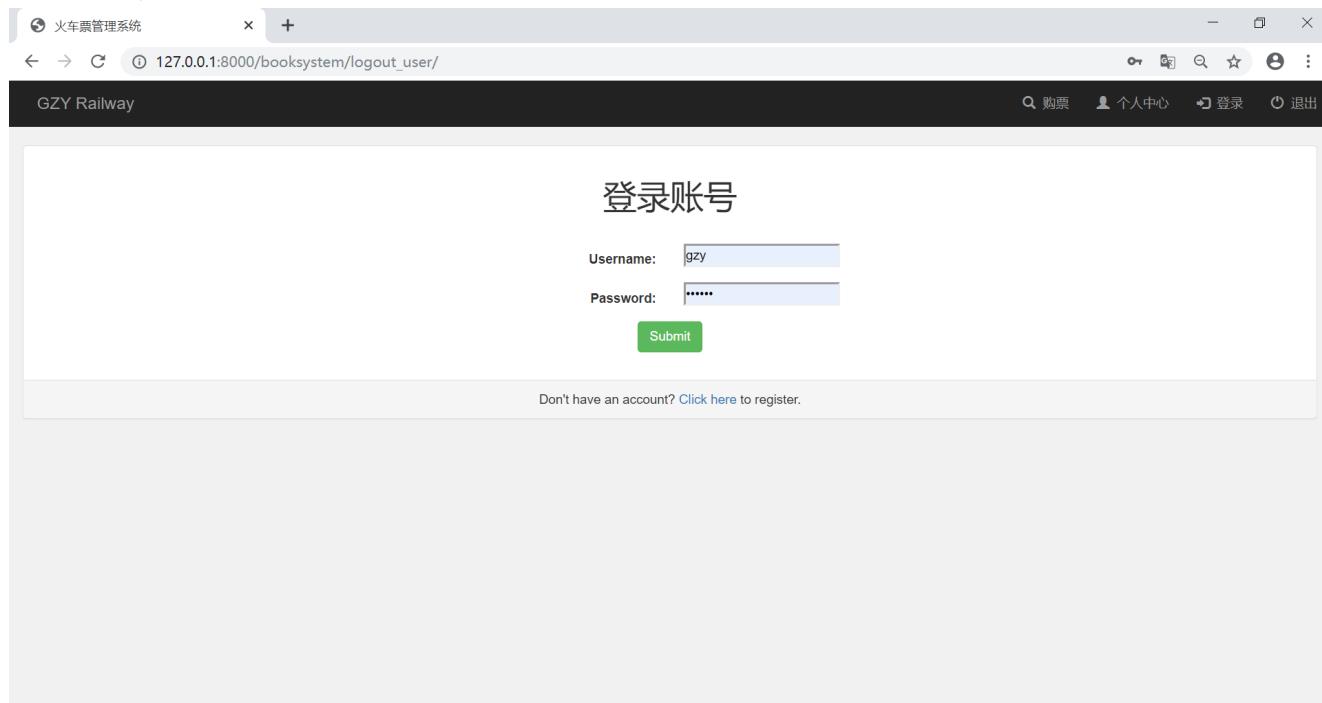
尊敬的旅客

很抱歉，我们目前没有开通这趟车次，请尝试选择其他车次，祝您旅途愉快！

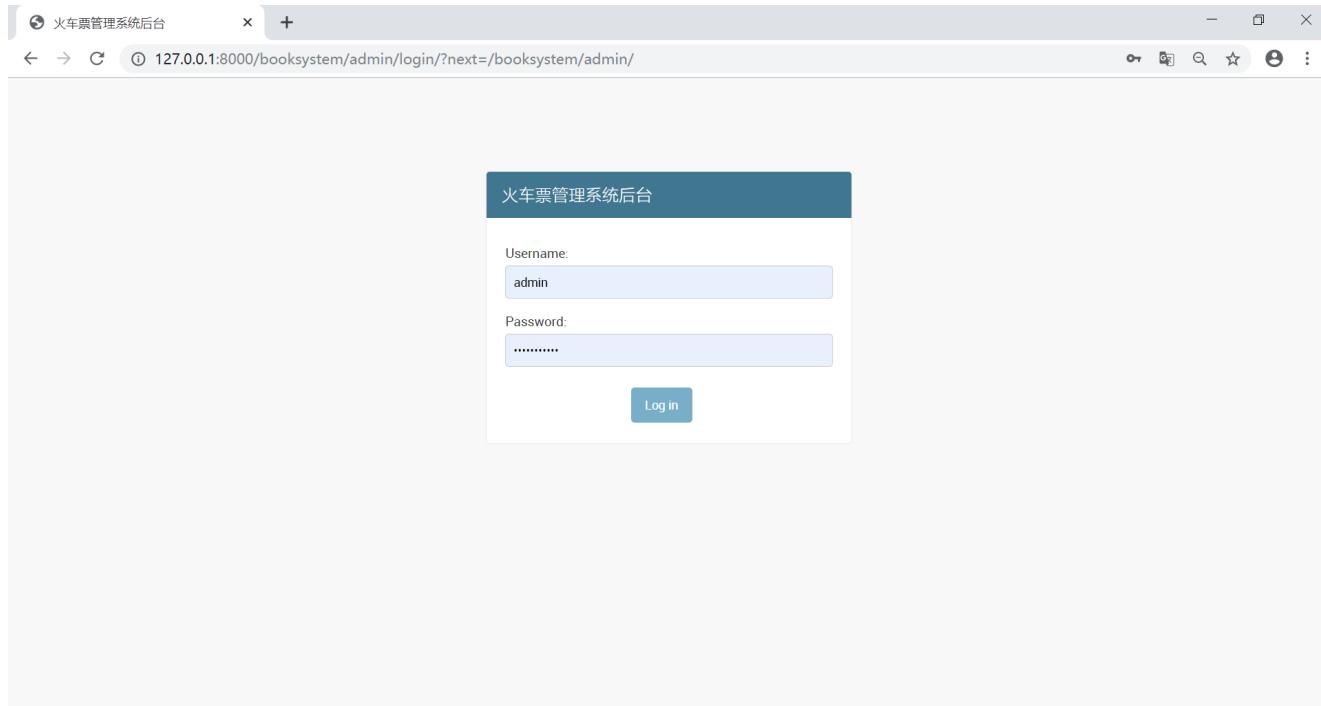
若预订已经预订过的车次，将会跳转至以下页面



点击右上角的退出可退出登录



修改网页地址栏在booksystem后加上admin，将进入本预订系统的后台



输入admin的账号密码即可登录后台

A screenshot of the "Train Ticket Management System" dashboard. The title bar says "火车票管理系统后台". The URL in the address bar is "127.0.0.1:8000/booksystem/admin/". The dashboard includes a "WELCOME, ADMIN" message and links for "VIEW SITE / CHANGE PASSWORD / LOG OUT". On the left, there's a sidebar with "Site administration" and sections for "AUTHENTICATION AND AUTHORIZATION" (Groups, Users) and "BOOKSYSTEM" (Railways). The "Users" and "Railways" items are highlighted with red boxes. On the right, there are two panels: "Recent actions" (listing recent operations like G1243 Railway, G1243 Railway, etc.) and "My actions" (listing recent operations like G1243 Railway, G1243 Railway, etc.).

点击users或者railways旁边的add, change可直接跳转到添加, 修改的页面

点击users可以查看本系统的用户以及管理员

点击Railways即可查看所有火车的信息, 并可进行增改删查的操作

右边的Recent actions可查看管理员的近期操作日志。

火车票管理系统后台

WELCOME, ADMIN VIEW SITE / CHANGE PASSWORD / LOG OUT

Home · Authentication and Authorization · Users

Select user to change

Action: ----- Go 0 of 1 selected

USERNAME	EMAIL ADDRESS	FIRST NAME	LAST NAME	STAFF STATUS
admin	893050234@qq.com			✓

1 user

**FILTER**

- By staff status
  - All
  - Yes
  - No
- By superuser status
  - All
  - Yes
  - No
- By active
  - All
  - Yes
  - No

如上为管理员表，点击右侧窗口按钮可进行切换为用户表或者查看用户以及管理员状态(是否active)

火车票管理系统后台

WELCOME, ADMIN VIEW SITE / CHANGE PASSWORD / LOG OUT

Home · Authentication and Authorization · Users

Select user to change

Action: ----- Go 0 of 3 selected

USERNAME	EMAIL ADDRESS	FIRST NAME	LAST NAME	STAFF STATUS
guoziyu	893050234@qq.com			✗
gzy	893050234@qq.com			✗
郭梓煜	893050234@qq.com			✗

3 users

**FILTER**

- By staff status
  - All
  - Yes
  - No
- By superuser status
  - All
  - Yes
  - No
- By active
  - All
  - Yes
  - No

如上为用户表

Select user to change

Action: **Delete selected users** | Go | 0 of 4 selected

USERNAME	EMAIL ADDRESS	FIRST NAME	LAST NAME	STAFF STATUS
admin	893050234@qq.com			✓
guoziyu	893050234@qq.com			✗
gzy	893050234@qq.com			✗
郭梓煜	893050234@qq.com			✗

4 users

**FILTER**

By staff status

- All
- Yes
- No

By superuser status

- All
- Yes
- No

By active

- All
- Yes
- No

选择actions栏选择用户或管理员点击go即可进行删除操作  
点击add user即可添加管理员

Add user

First, enter a username and password. Then, you'll be able to edit more user options.

Username: **admin**  
Required: 150 characters or fewer. Letters, digits and @/./+/-/\_ only.

Password: **\*\*\*\*\***

Password confirmation:   
Enter the same password as before, for verification.

**SAVE**



点击右上角的change password还可以跳转至修改密码界面

火车票管理系统后台

WELCOME, ADMIN CHANGE PASSWORD / LOG OUT

Home > Password change

Password change

Please enter your old password, for security's sake, and then enter your new password twice so we can verify you typed it in correctly.

Old password:

New password:

Your password can't be too similar to your other personal information.  
Your password must contain at least 8 characters.  
Your password can't be a commonly used password.  
Your password can't be entirely numeric.

New password confirmation:

**CHANGE MY PASSWORD**

点击左边的home回到首页 点击Railways查看所有车次信息表  
点击右上角按钮可添加车次

火车票管理系统后台

WELCOME, ADMIN VIEW SITE / CHANGE PASSWORD / LOG OUT

Home > Booksystm > Railways

Select railway to change

Action:  Go 0 of 20 selected

<input type="checkbox"/>	NAME	LEAVE CITY	ARRIVE CITY	LEAVE STATION	ARRIVE STATION	LEAVE TIME	ARRIVE TIME	CAPACITY	PRICE	BOOK SUM	INCOME
<input type="checkbox"/>	G1243	广州	揭阳	广州东站	揭阳站	Dec. 27, 2019, 12:39 p.m.	Dec. 27, 2019, 3:39 p.m.	100	200.0	50	10000.0
<input type="checkbox"/>	G2632	广州	揭阳	广州东站	揭阳站	Dec. 27, 2019, 9:30 p.m.	Dec. 27, 2019, 11:11 p.m.	200	200.0	2	400.0
<input type="checkbox"/>	G2378	广州	揭阳	广州东站	潮汕站	Jan. 1, 2020, 7:21 a.m.	Jan. 1, 2020, 9:21 a.m.	200	199.0	2	398.0
<input type="checkbox"/>	G1283	广州	揭阳	广州南站	潮汕站	Jan. 1, 2020, 10:20 a.m.	Jan. 1, 2020, 1:23 p.m.	200	200.0	2	400.0
<input type="checkbox"/>	D6947	广州	武汉	广州南站	武汉站	Dec. 27, 2019, 9:31 a.m.	Dec. 27, 2019, 2:17 p.m.	230	430.0	5	2150.0
<input type="checkbox"/>	G1260	广州	上海	广州南站	上海南站	Dec. 27, 2019, 6:13 p.m.	Dec. 27, 2019, 8:53 p.m.	500	400.0	6	2400.0
<input type="checkbox"/>	D3526	北京	天津	北京西站	天津站	Dec. 27, 2019, 6:09 a.m.	Dec. 27, 2019, 10:11 a.m.	100	350.0	2	700.0
<input type="checkbox"/>	Z3216	广州	昆明	广州南站	昆明南站	Dec. 27, 2019, 6:05 p.m.	Dec. 27, 2019, 8:05 p.m.	250	300.0	3	900.0
<input type="checkbox"/>	G4584	广州	石家庄	广州南站	石家庄北站	Dec. 27, 2019, 10:03 a.m.	Dec. 27, 2019, 12:13 p.m.	300	234.0	2	468.0
<input type="checkbox"/>	G3751	长沙	南宁	长沙南站	南宁西站	Dec. 27, 2019, 7:58 a.m.	Dec. 27, 2019, 9:52 a.m.	300	299.0	1	299.0
<input type="checkbox"/>	G3623	广州	揭阳	广州东站	揭阳站	Dec. 27, 2019, 9:55 a.m.	Dec. 27, 2019, 1:56 p.m.	199	198.0	4	795.0
<input type="checkbox"/>	Z3087	北京	济南	北京西站	济南北站	Dec. 27, 2019, 6:53 p.m.	Dec. 27, 2019, 10:53 p.m.	300	450.0	2	900.0
<input type="checkbox"/>	D3321	北京	上海	北京西站	上海南站	Dec. 27, 2019, 9:51 a.m.	Dec. 27, 2019, 1:51 p.m.	300	244.0	7	1708.0
<input type="checkbox"/>	D9242	北京	长沙	北京西站	长沙南站	Dec. 27, 2019, 6:34 a.m.	Dec. 27, 2019, 9:16 a.m.	300	400.0	2	800.0

点击add railway后跳转至以下页面，填入信息点击下面save可进行保存

Add railway

Name:

Leave city:

Arrive city:

Leave station:

Arrive station:

Leave time: Date:  Today /  Time:  Now /

Arrive time: Date:  Today /  Time:  Now /

Capacity:

Price:

Book sum:

Income:

Save and add another Save and continue editing SAVE

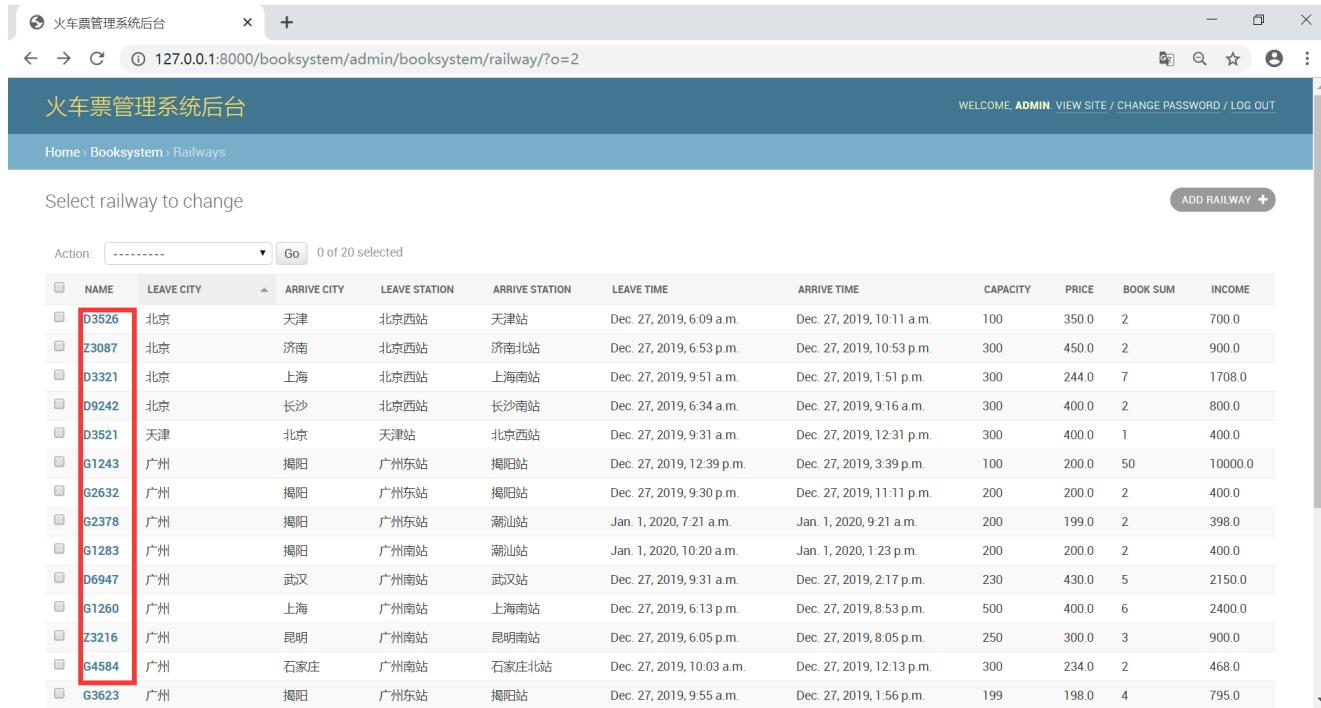
选择delete点击go可直接删除该车次

Select railway to change

Action: Delete selected railways  of 20 selected

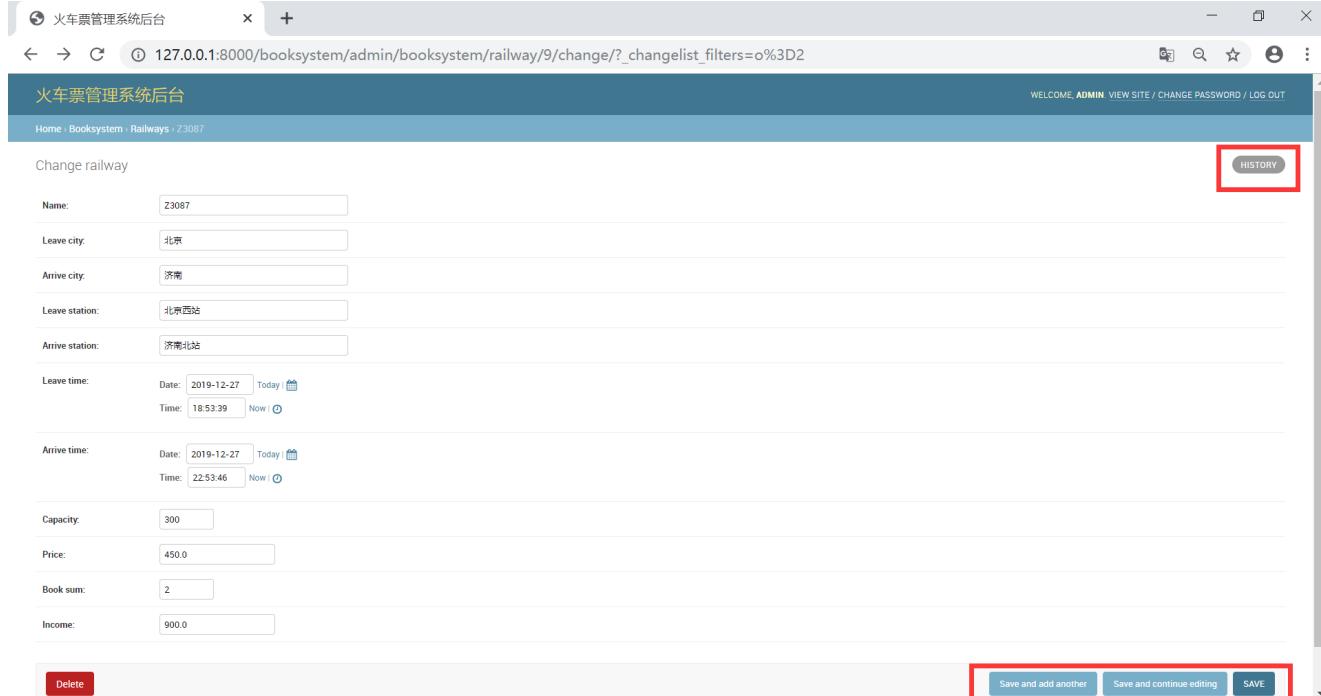
<input type="checkbox"/>	NAME	LEAVE CITY	ARRIVE CITY	LEAVE STATION	ARRIVE STATION	LEAVE TIME	ARRIVE TIME	CAPACITY	PRICE	BOOK SUM	INCOME
<input checked="" type="checkbox"/>	G1243	广州	揭阳	广州东站	揭阳站	Dec. 27, 2019, 12:39 p.m.	Dec. 27, 2019, 3:39 p.m.	100	200.0	50	10000.0
<input type="checkbox"/>	G2632	广州	揭阳	广州东站	揭阳站	Dec. 27, 2019, 9:30 p.m.	Dec. 27, 2019, 11:11 p.m.	200	200.0	2	400.0
<input type="checkbox"/>	G2378	广州	揭阳	广州东站	潮汕站	Jan. 1, 2020, 7:21 a.m.	Jan. 1, 2020, 9:21 a.m.	200	199.0	2	398.0
<input type="checkbox"/>	G1283	广州	揭阳	广州南站	潮汕站	Jan. 1, 2020, 10:20 a.m.	Jan. 1, 2020, 1:23 p.m.	200	200.0	2	400.0
<input type="checkbox"/>	D6947	广州	武汉	广州南站	武汉站	Dec. 27, 2019, 9:31 a.m.	Dec. 27, 2019, 2:17 p.m.	230	430.0	5	2150.0
<input type="checkbox"/>	G1260	广州	上海	广州南站	上海南站	Dec. 27, 2019, 6:13 p.m.	Dec. 27, 2019, 8:53 p.m.	500	400.0	6	2400.0
<input type="checkbox"/>	D3526	北京	天津	北京西站	天津站	Dec. 27, 2019, 6:09 a.m.	Dec. 27, 2019, 10:11 a.m.	100	350.0	2	700.0
<input type="checkbox"/>	Z3216	广州	昆明	广州南站	昆明南站	Dec. 27, 2019, 6:05 p.m.	Dec. 27, 2019, 8:05 p.m.	250	300.0	3	900.0
<input type="checkbox"/>	G4584	广州	石家庄	广州南站	石家庄北站	Dec. 27, 2019, 10:03 a.m.	Dec. 27, 2019, 12:13 p.m.	300	234.0	2	468.0
<input type="checkbox"/>	G3751	长沙	南宁	长沙南站	南宁西站	Dec. 27, 2019, 7:58 a.m.	Dec. 27, 2019, 9:52 a.m.	300	299.0	1	299.0
<input type="checkbox"/>	G3623	广州	揭阳	广州东站	揭阳站	Dec. 27, 2019, 9:55 a.m.	Dec. 27, 2019, 1:56 p.m.	199	198.0	4	795.0
<input type="checkbox"/>	Z3087	北京	济南	北京西站	济南北站	Dec. 27, 2019, 6:53 p.m.	Dec. 27, 2019, 10:53 p.m.	300	450.0	2	900.0
<input type="checkbox"/>	D3321	北京	上海	北京西站	上海南站	Dec. 27, 2019, 9:51 a.m.	Dec. 27, 2019, 1:51 p.m.	300	244.0	7	1708.0
<input type="checkbox"/>	D9242	北京	长沙	北京西站	长沙南站	Dec. 27, 2019, 6:34 a.m.	Dec. 27, 2019, 9:16 a.m.	300	400.0	2	800.0

点击左边车次name即可直接跳转至修改车次信息页面



	NAME	LEAVE CITY	ARRIVE CITY	LEAVE STATION	ARRIVE STATION	LEAVE TIME	ARRIVE TIME	CAPACITY	PRICE	BOOK SUM	INCOME
	D3526	北京	天津	北京西站	天津站	Dec. 27, 2019, 6:09 a.m.	Dec. 27, 2019, 10:11 a.m.	100	350.0	2	700.0
	Z3087	北京	济南	北京西站	济南北站	Dec. 27, 2019, 6:53 p.m.	Dec. 27, 2019, 10:53 p.m.	300	450.0	2	900.0
	D3321	北京	上海	北京西站	上海南站	Dec. 27, 2019, 9:51 a.m.	Dec. 27, 2019, 1:51 p.m.	300	244.0	7	1708.0
	D9242	北京	长沙	北京西站	长沙南站	Dec. 27, 2019, 6:34 a.m.	Dec. 27, 2019, 9:16 a.m.	300	400.0	2	800.0
	D3521	天津	北京	天津站	北京西站	Dec. 27, 2019, 9:31 a.m.	Dec. 27, 2019, 12:31 p.m.	300	400.0	1	400.0
	G1243	广州	揭阳	广州东站	揭阳站	Dec. 27, 2019, 12:39 p.m.	Dec. 27, 2019, 3:39 p.m.	100	200.0	50	10000.0
	G2632	广州	揭阳	广州东站	揭阳站	Dec. 27, 2019, 9:30 p.m.	Dec. 27, 2019, 11:11 p.m.	200	200.0	2	400.0
	G2378	广州	揭阳	广州东站	潮汕站	Jan. 1, 2020, 7:21 a.m.	Jan. 1, 2020, 9:21 a.m.	200	199.0	2	398.0
	G1283	广州	揭阳	广州南站	潮汕站	Jan. 1, 2020, 10:20 a.m.	Jan. 1, 2020, 1:23 p.m.	200	200.0	2	400.0
	D6947	广州	武汉	广州南站	武汉站	Dec. 27, 2019, 9:31 a.m.	Dec. 27, 2019, 2:17 p.m.	230	430.0	5	2150.0
	G1260	广州	上海	广州南站	上海南站	Dec. 27, 2019, 6:13 p.m.	Dec. 27, 2019, 8:53 p.m.	500	400.0	6	2400.0
	Z3216	广州	昆明	广州南站	昆明南站	Dec. 27, 2019, 6:05 p.m.	Dec. 27, 2019, 8:05 p.m.	250	300.0	3	900.0
	G4584	广州	石家庄	广州南站	石家庄北站	Dec. 27, 2019, 10:03 a.m.	Dec. 27, 2019, 12:13 p.m.	300	234.0	2	468.0
	G3623	广州	揭阳	广州东站	揭阳站	Dec. 27, 2019, 9:55 a.m.	Dec. 27, 2019, 1:56 p.m.	199	198.0	4	795.0

可在左侧修改后点击下面的save保存



Name: Z3087

Leave city: 北京

Arrive city: 济南

Leave station: 北京西站

Arrive station: 济南北站

Leave time: Date: 2019-12-27 Today:  Time: 18:53:39 Now:

Arrive time: Date: 2019-12-27 Today:  Time: 22:53:46 Now:

Capacity: 300

Price: 450.0

Book sum: 2

Income: 900.0

也可点击右上角的history查看历史修改记录

The screenshot shows a browser window titled "火车票管理系统后台" (Railway Ticket Management System Backend). The URL is 127.0.0.1:8000/booksystem/admin/booksystem/railway/20/history/. The page displays a table of change history for train G1243. The table has columns for DATE/TIME, USER, and ACTION. The actions listed are: "Added.", "Changed book\_sum and income.", "Changed capacity.", and "Changed book\_sum and income.". The date and time for all entries is Jan. 1, 2020, at 10:39 p.m.

DATE/TIME	USER	ACTION
Jan. 1, 2020, 10:39 p.m.	admin	Added.
Jan. 1, 2020, 10:41 p.m.	admin	Changed book_sum and income.
Jan. 1, 2020, 10:41 p.m.	admin	Changed capacity.
Jan. 1, 2020, 10:43 p.m.	admin	Changed book_sum and income.

点击log out即可退出管理员登录

The screenshot shows a browser window titled "火车票管理系统后台" (Railway Ticket Management System Backend). The URL is 127.0.0.1:8000/booksystem/admin/logout/. The page displays a message: "Logged out". Below the message, there is a note: "Thanks for spending some quality time with the Web site today." and a link: "Log in again".

点击右上角的view site可回到<http://127.0.0.1:8000/>  
重新查询后，点击个人中心

The screenshot shows a search interface with input fields for departure station (广州), arrival station (揭阳), and date (2019/12/27). Below the search bar are four tabs: 车次信息 (Train Information), 出发时间 (Departure Time), 到达时间 (Arrival Time), and 价格 (Price). The results list four train entries:

车次	出发时间	到达时间	票价
G1283	09:35 广州南站	13:35 潮汕站	满座率 2/100   ￥198.0 起 [订票]
G3623	09:55 广州东站	13:56 揭阳站	满座率 4/100   ￥198.0 起 [订票]
G2164	09:25 广州南站	10:32 潮汕站	满座率 3/100   ￥199.0 起 [订票]
G2632	21:30 广州东站	23:11 揭阳站	满座率 2/100   ￥200.0 起 [订票]

可看到此时用户为admin管理员身份

点击个人中心，即可查看本系统的年，月，星期利润以及所有的订单列表

The screenshot displays three tables under the '个人中心' (Personal Center) section:

- 周收入 (Weekly Income):**

编号	周次	车次	收入
1	00	2	798.0
2	51	17	24613.0
- 月收入 (Monthly Income):**

编号	月份	车次	收入
1	01	2	798.0
2	12	17	24613.0
- 年收入 (Annual Income):**

编号	年份	车次	收入
1	2019	17	24613.0
2	2020	2	798.0

下方是 '订单列表 (Order List)'，显示了三笔订单：

编号	旅客	车次	路线	出发时间	车票
1	gzy	G3623	广州 → 揭阳	Dec. 27, 2019, 9:55 a.m.	198.0
2	gzy	G1283	广州 → 揭阳	Dec. 27, 2019, 9:35 a.m.	198.0
3	郭梓煜	G1283	广州 → 揭阳	Dec. 27, 2019, 9:35 a.m.	198.0

## 视频展示

视频展示点击[这里](#)

视频中展示了访问身份为用户或管理员的情况，由于疏忽，没有演示使用管理员身份查看利润以及订单列表的情况，这一部分的情况可在上面的截图(就在视频展示的上一张图片)或者在videos中查看利润订单视频。

此外，针对每一个步骤，还录制了详细的视频以及制作了PPT引导，可在videos文件夹中查看，具体所有的实现成果也在以上[实现成果](#)已展出。

# 总结

---

根据组内成员的多次测试以及修改，本系统运转界面功能相对完善，面对用户的各种操作甚至误操作都设计了错误提示，在[实现成果](#)已有说明，而网页的跳转也较为灵敏快捷，界面的显示以及功能都很稳定。

## 遇到问题&解决思路

---

### 1. 数据库使用问题

一开始我准备使用数据库是 MySQL，可由于数据量不大，而且Django可以便捷地生成并链接sqlite3数据库，所以这次我用的是 Django 默认的数据库 Sqlite，这是一个轻量级的数据库，除了自带的一些指令与其他数据库有差异，大部分 SQL 语句与主流数据库都相同，但是 Sqlite 是一个本地型的数据库，无需安装和管理配置，并且占用空间非常小，用来做小型的网站开发完全够用。

### 2. 前端实现问题

原本我对写网站，设计web界面并不熟悉，布置了这个作业后，才开始了解。我便开始学习相关的基础的 html, css, js，对网站前端有了基本认识，又结合我目前常用的 Python 语言，学习了 Django 网页开发框架，对网站后端处理有了深层认识。另外，我现在用 web 开发的时候，对数据库的操作已经不是 SQL 语句了，而是通过高级语言（如 Python）的语法来完成对数据库的增删改查操作。比如说在查询一个表中的信息时，sql中是`select * from table`而 Django中是`Flight.objects.all()`。

### 3. 后端开发问题

后端的开发主要依赖于Django的框架，Django的版本问题属实需要注意一下。在写代码的时候要注意Django1和Django2版本的不同语法，我在网上参考了[教程](#)，获益匪浅。

## 心得体会

---

由于在这次大作业之前，我并没有接触到 html, css, js等方面的知识，对网页前端的了解也较为浅薄，所以要完成这次大作业，就需要我去另外地学习新的知识，也是相当地不容易。好在大作业的布置时间较早，有充足的时间学习。

经过了一段时间的学习，我结合python语言，学习了Django的网页开发框架，也对网站的后端处理有了较深的了解，开始完成这次大作业。结果这次大作业的实现，尽管对网站的前后端如何进行交互运作存在一些疑问，但对python, Django等语言进行开发变得更加地熟练了。

总而言之，这次数据库大作业不仅让我数据库表的设计更加熟悉，对书本上的知识复习了一波，同时也让我学习到了这门课以外的新知识，对网页前后端的实现都有了较深入的理解。