

# Report on Database Project of SUSTech Railway 12305 System

Chi Xu 11912224 , Yunhao Luo 11813221, Haowei Kuang 11910710

December 2020

## 1 Introduction

Here we will show a fantastic system of railway, just like 12306, we call it SUSTech Railway 12305 System. In the following sections, we will introduce our system in these aspects: data crawling, table establishment, and our system functions with many extra points.

## 2 Design

Here is our design steps. First, we need to get authentic data of the trains, stations, time table, interval distance, order of stations, their relationships, etc. Second, we need to find their close and simple connection, then build several tables to connect them. Meanwhile, we must follow the normalization very strictly, since all the project is based on these diagram, and we almost spent one week to discuss the design of table. Third, distribute the workload and then everyone does his own part. However, since the distribution cannot be total fair and the competence is also different, almost everyone dose one third.

## 3 Process and bright-spot

### 3.1 Data

At the very beginning of the project, in the process of obtaining the information of real train numbers, we considered to crawl the network data by learning Python. After studying BeautifulSoup on our own, we started to try to get data by crawling the 12306 ticket platform. However, the complex login verification code of 12306 makes our simulated login operation unable to complete, resulting in the failure of data crawling. Without access to the most authoritative websites, we had to find a new way to get the information from other websites. After a search, we finally accepted the reference List of train numbers using the train ticket network (<http://search.huochepiao.com>), through Python to perform request URL, get response, call BS4 module to crawl the data. In the process of crawling, we adopted the crawling method according to the train number, starting from the train number in the reference table, and obtained the content through label comparison, and crawled the detailed data layer by layer. Through several methods, the operation of request URL, obtaining station name, obtaining train ticket information and so on was respectively performed, and the crawled data form was sorted and summarized according to the requirements of database construction, so as to obtain the final CSV master table, which was applied in the process of the final import into the database.

### 3.2 Table and relationship

### 3.3 Functions

When it comes to functions, I will list them one by one: the query of ticket, query of train, query of station, administrator station management, administrator train management, user management, privilege management, exception handling, hint of operations, and etc. Our project is divided into two general page: user and administrator. Here is the entrance of the system.

Then if we want to login the user page, we need provide a user name and password, just like this:

This web is somewhat useless, since in the requirement user is a general concept, however, this is not the point. Next I will show the system main function.

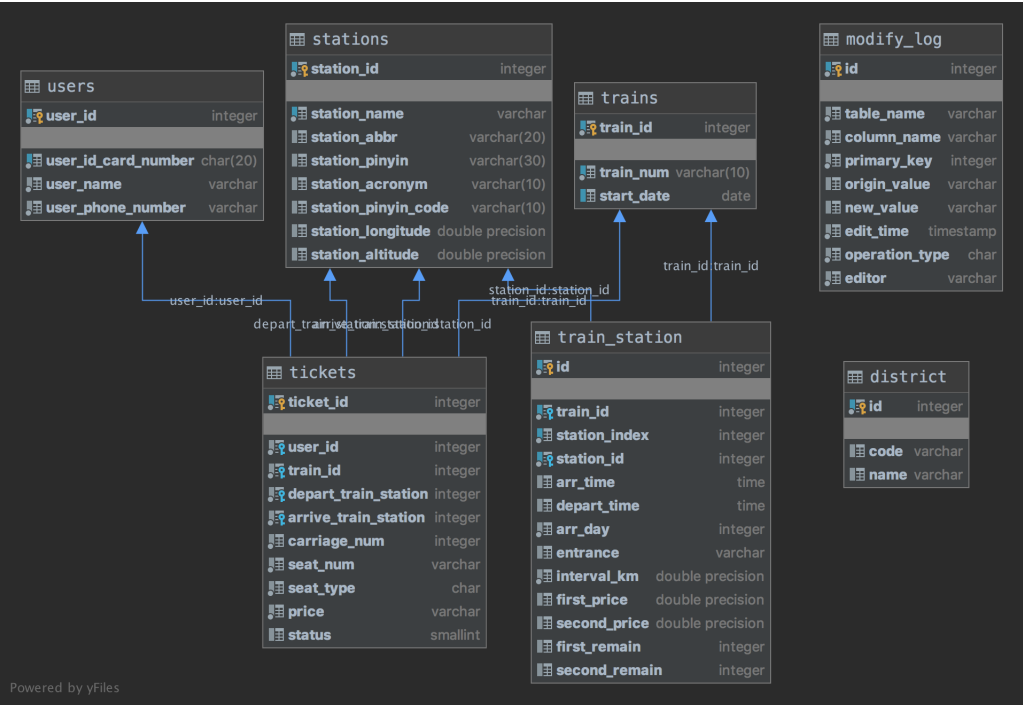


Figure 1: TABLE

train_station	
<b>id</b>	integer
<b>train_id</b>	integer
<b>station_index</b>	integer
<b>station_id</b>	integer
<b>arr_time</b>	time
<b>depart_time</b>	time
<b>arr_day</b>	integer
<b>entrance</b>	varchar
<b>interval_km</b>	double precision
<b>first_price</b>	double precision
<b>second_price</b>	double precision
<b>first_remain</b>	integer
<b>second_remain</b>	integer

Figure 2: TABLE2

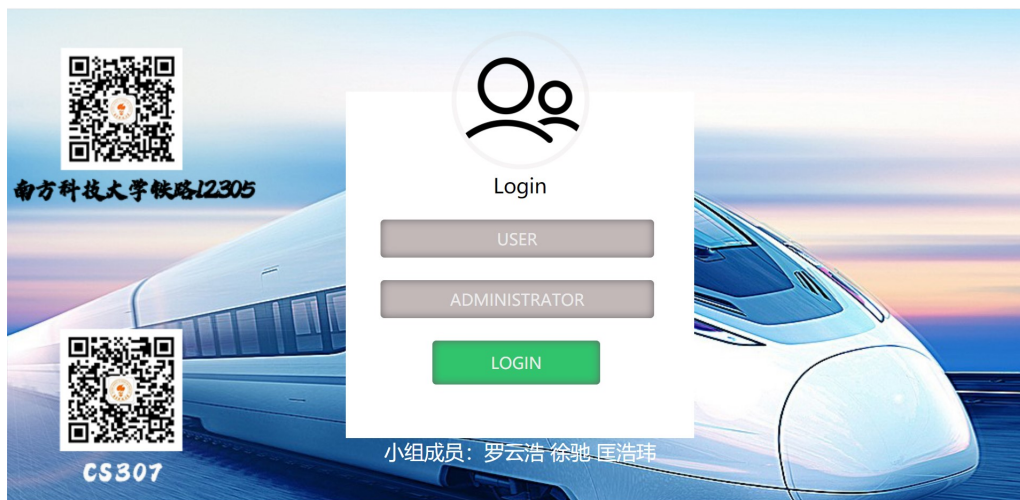


Figure 3: login

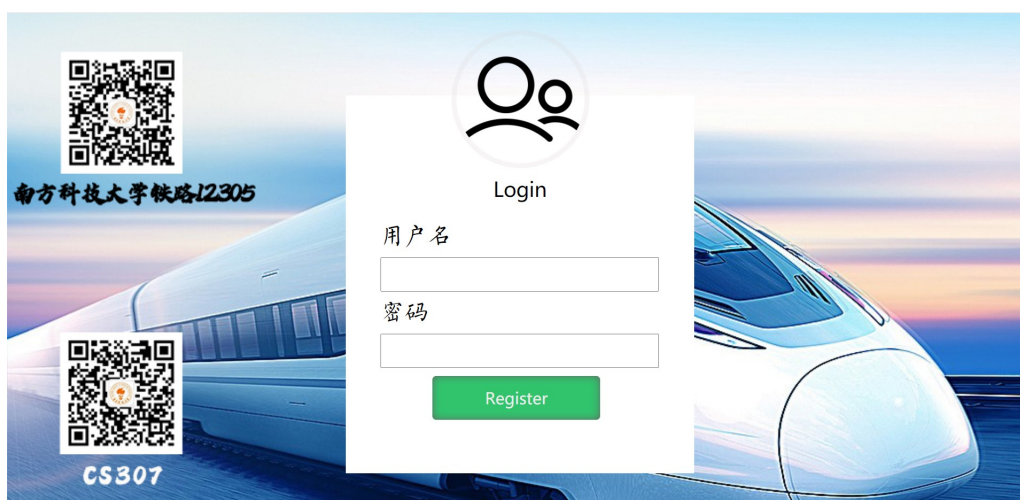


Figure 4: register

### 3.3.1 Query of ticket

When we enter the departing station, the arriving station and the date, the result will like the page below. NB: to make reader understand our system more easily, we choose BeijingXi to GuangzhouNan in 2020-12-30 as our example. Meanwhile, if we want know more information about a certain train, the button beside it could help us to get the hyper-reference.

车次号	出发时间	运行时间	到达时间	一等票价	一等余票	二等票价	二等余票	出发站序号	目的站序号
G65	10:33	09小时51分	20:24	1380	196	862	800	1	14
G67	12:13	10小时08分	22:21	1380	200	862	800	1	17
G69	13:07	09小时24分	22:31	1380	199	862	800	1	11
G71	07:26	09小时55分	17:21	1380	200	862	800	1	15
G79	10:00	08小时01分	18:01	1380	200	862	800	1	6

Figure 5: userquery

### 3.3.2 Query of train and station

Now that we got the the trainNum, we want to get all station this train will pass. So let us set G69 as an example. We enter G69 and 2020-12-30 in the text bar, and the result is as follows.

车站序号	车站名称	到达时间	出发时间	停留时间	一等余票	二等余票	里程数
1	北京西		07:26:00		200	800	0
2	涿州东	07:51:00	07:53:00	00:02:00	200	800	62
3	保定东	08:14:00	08:16:00	00:02:00	200	800	139
4	石家庄	08:52:00	08:58:00	00:06:00	200	800	281
5	邢台东	09:26:00	09:34:00	00:08:00	200	800	403
6	邯郸东	09:50:00	09:52:00	00:02:00	200	800	456
7	鹤壁东	10:19:00	10:21:00	00:02:00	200	800	562
8	郑州东	10:53:00	10:57:00	00:04:00	200	800	693

Figure 6: usertrainquery

Here we can see, all stations has been presented, including the interval time and ticket remain. If you want to but one ticket, just try to click the navigation bar option.

### 3.3.3 Buy ticket

Then it comes to the buying page, which is a little complex compared to the previous webs. However, its structure is really clear, which will help everyone buy a ticket without much effort. Here is the picture. In this

section, I will not show the process in the report, since that behavior may effect the structure and outline of this page.

Figure 7: userbuy

### 3.3.4 Query of order

When an user buy a ticket, he or she can query it by his or her own idcardnumber, which then can help he or she refund it. Let show it below.

46	北京西	高碑店东	320305200011200000	2020年12月30日	13:07	13:37	G69	一等座	1车10A	60.50元	已支付	<a href="#">退票</a>
47	北京西	涿州东	320305200011200000	2020年12月30日	07:26	07:51	G71	一等座	1车10A	45.50元	已退票	<a href="#">退票</a>

Figure 8: refund

### 3.3.5 Addition of station

New station can be added. For the sake of adding train, this is a must.

### 3.3.6 Subtraction of station

Station which is redundant can be deleted.

### 3.3.7 Addition of train

New train can be added, with a lot of new information added.

### 3.3.8 Subtraction of train

Any train can be deleted.

The screenshot shows a web interface with three main sections:

- 增加车站 (Add Station):** A form with a text input for '车站名' (Station Name) and a '添加' (Add) button.
- 删除车站 (Delete Station):** A form with a text input for '车站名' (Station Name) and a '删除' (Delete) button.
- 增加车次 (Add Train Route):** A form with multiple text inputs: '车次编号' (Train Number) with example 'G123', '出发日期' (Departure Date) with '2020-12-30', '添加车站' (Add Station) with 'NKD', '出发时间' (Departure Time) with '12:00', '到达时间' (Arrival Time) with '15:00', and '行驶天数' (Travel Days) with '1'.

Figure 9: Admin

The screenshot shows a web interface with two main sections:

- 增加车次 (Add Train Route):** A form with text inputs for '到达时间' (Arrival Time) with example '15:00', '行驶天数' (Travel Days) with '1', '检票入口' (Ticket Entry) with 'S8', '行驶里程' (Travel Distance) with '100', '一等票价' (First Class Fare) with '199', '二等票价' (Second Class Fare) with '100', '一等余票' (First Class Remaining Tickets) with '99', and '二等余票' (Second Class Remaining Tickets) with '300'. There is a '添加' (Add) button.
- 删除车次 (Delete Train Route):** A form with text inputs for '车次编号' (Train Number) with example 'G123' and '出发日期' (Departure Date) with '2020-12-30', and a '删除' (Delete) button.

Figure 10: Admin2

### 3.3.9 High concurrency manipulation

We test 10,1000,10000,100000 users buying a ticket, here is the result. We submit 10,000 request to buy a ticket to our database. The output below shows that 1,000 tickets are sold correctly and we finish this test in 27.3 seconds, which is relatively fast, since our database is stored in a portable laptop rather than some high-performance servers.

```
高并发.txt
1 开始抢票: 1608496324287ms
2 2020-12-21 04:32:04.404 INFO 34991 --- [main] com.alibaba.druid.pool.
   DruidDataSource : {dataSource-1} inited
3 购票成功
4 购票成功
5 购票成功
6 购票成功
7 购票成功
8 购票成功
9 购票成功
10 购票成功
11 购票成功
12 购票成功
13 购票成功
14 购票成功
15 购票成功
16 购票成功
17 购票成功
18 购票成功
19 购票成功
20 购票成功
21 购票成功
22 购票成功
23 购票成功
24 购票成功
25 购票成功
26 购票成功
27 购票成功
28 购票成功
29 购票成功
30 购票成功
31 购票成功
32 购票成功
```

Figure 11: high

amount	time/s
10	0.592
1000	3.369
10000	27.327
100000	9997.971

Figure 12: result

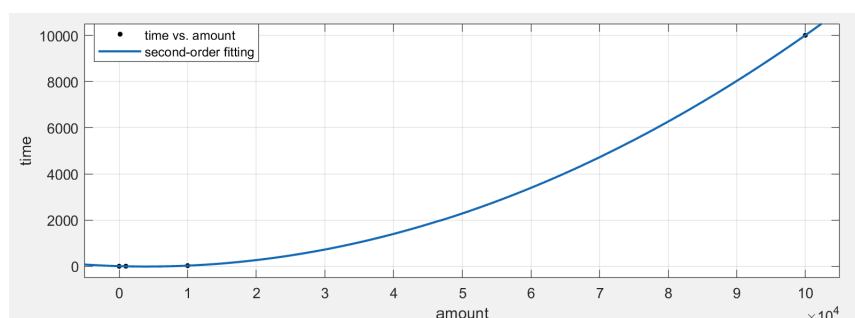


Figure 13: result1

## 4 Conclusion

In this project, we do and practice a lot. We learn a basic model for running a website. The whole procedure, from the client side, backend server side all the way to the lowest level database operation side is implemented. Although The SUSTech Railway 12305 System is to some degree great, though with some shortcoming. We will go!