**火车售票系统设计**

成员：葛星宏、李东星、唐华富、许浩俊

**项目大纲**

**一．设计大致思路：结构：前端、后端和数据库，**

从售票系统出发，先从票务的结构开始设计，建立好用户、站点、车次、座位和订单的表单，后端根据数据库结构来建立服务器。前端用户界面和后端通过http协议交换数据

1. **开发环境：**

后端：用c++在clion上，使用vamare虚拟机模拟linux服务器进行开发，用clion的ssh远程连接，方便进行调试。虚拟机使用ubantu发行版。用cmake管理项目

数据库：在linux系统上使用MySQL数据库进行开发

前端：使用QTcreator开发售票系统的用户界面，同样使用cmake管理项目

前端用户界面如图：



**三．已实现功能和运行截图：**

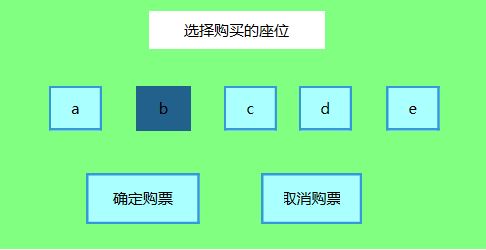
1.用户登陆功能，需要用户先登陆才能进行购票，否则会弹出提示



2.通过火车两个站点进行票务查询，会返回车次、出发时间和到达时间、价格数据。



3.双击后可以购买具体车次，然后弹出车次座位选择，点击购买后则成功购票

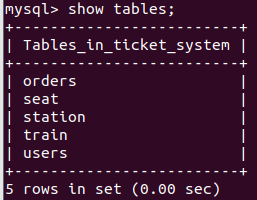


4.查询已购买订单

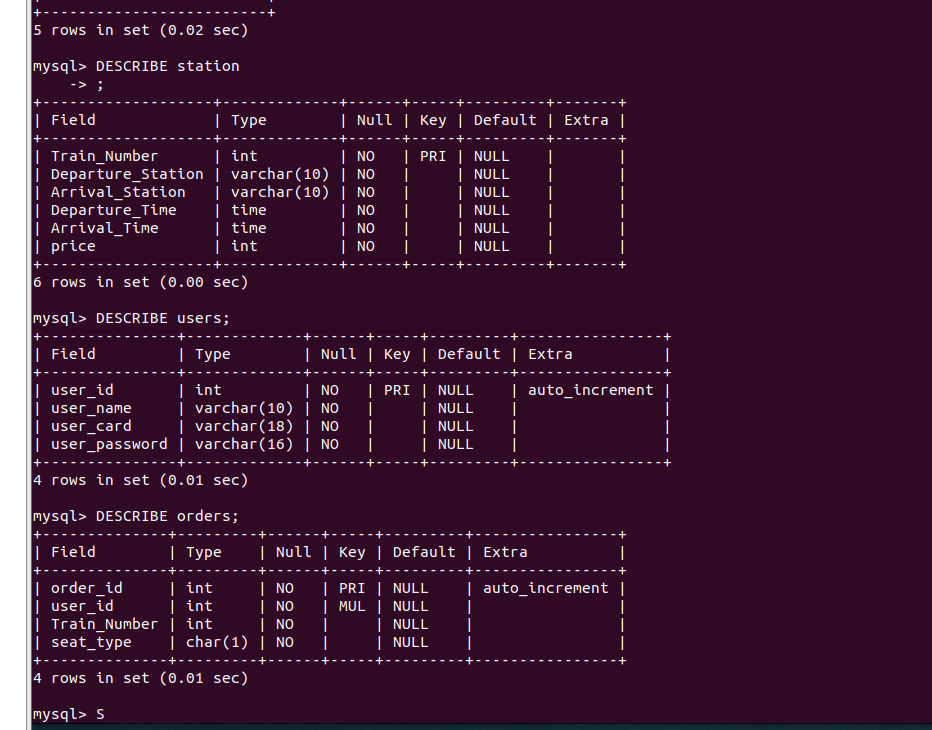
**四．详细设计和开发的过程**

**数据库设计思路**：

1.数据库分表，将用户、站点和车次信息、订单信息的表单分开，提高查询效率。当后续扩展和优化时只需要修改特定的表单即可。该项目数据库所有表单如图

****

**部分表单结构**

****

2.建库语句记录

**用户表单**：保存

create table users(

user\_id INT PRIMARY KEY AUTO\_INCREMENT,

user\_name varchar(10) NOT NULL,

user\_card varchar(18) NOT NULL,

user\_password varchar(16) NOT NULL

);

**站点表单**：存储两站点的往返的所有车次数据

CREATE TABLE station(

Train\_Number INT PRIMARY KEY,

Departure\_Station VARCHAR(10) NOT NULL,

Arrival\_Station VARCHAR(10) NOT NULL,

Departure\_Time TIME NOT NULL,

Arrival\_Time TIME NOT NULL,

price INT NOT NULL

);

**座位表单**，存储某车次的所有座位的数量，通过外键和车次表单关联在一起

CREATE TABLE seat(

Train\_Number INT PRIMARY KEY,

Seat\_a INT NOT NULL,

Seat\_b INT NOT NULL,

Seat\_c INT NOT NULL,

Seat\_d INT NOT NULL,

seat\_e INT NOT NULL,

FOREIGN KEY (Train\_Number) REFERENCES station(Train\_Number)

)```

**订单信息表**`

CREATE TABLE orders (

order\_id INT PRIMARY KEY AUTO\_INCREMENT,

user\_id INT NOT NULL,

Train\_Number INT NOT NULL,

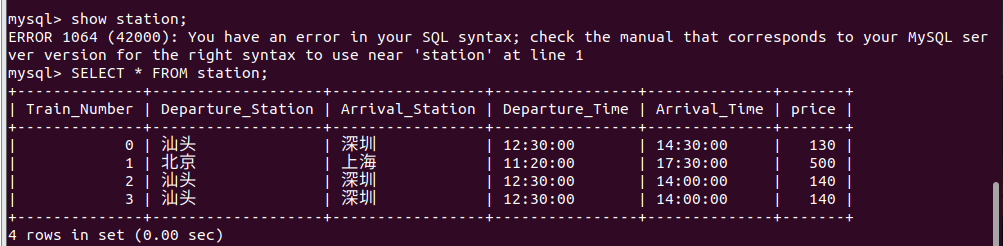
seat\_type CHAR(1) NOT NULL,

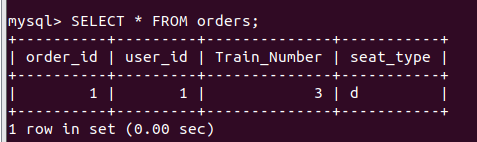
FOREIGN KEY (user\_id) REFERENCES users(user\_id)

);

3.部分数据在数据库的查询示例：

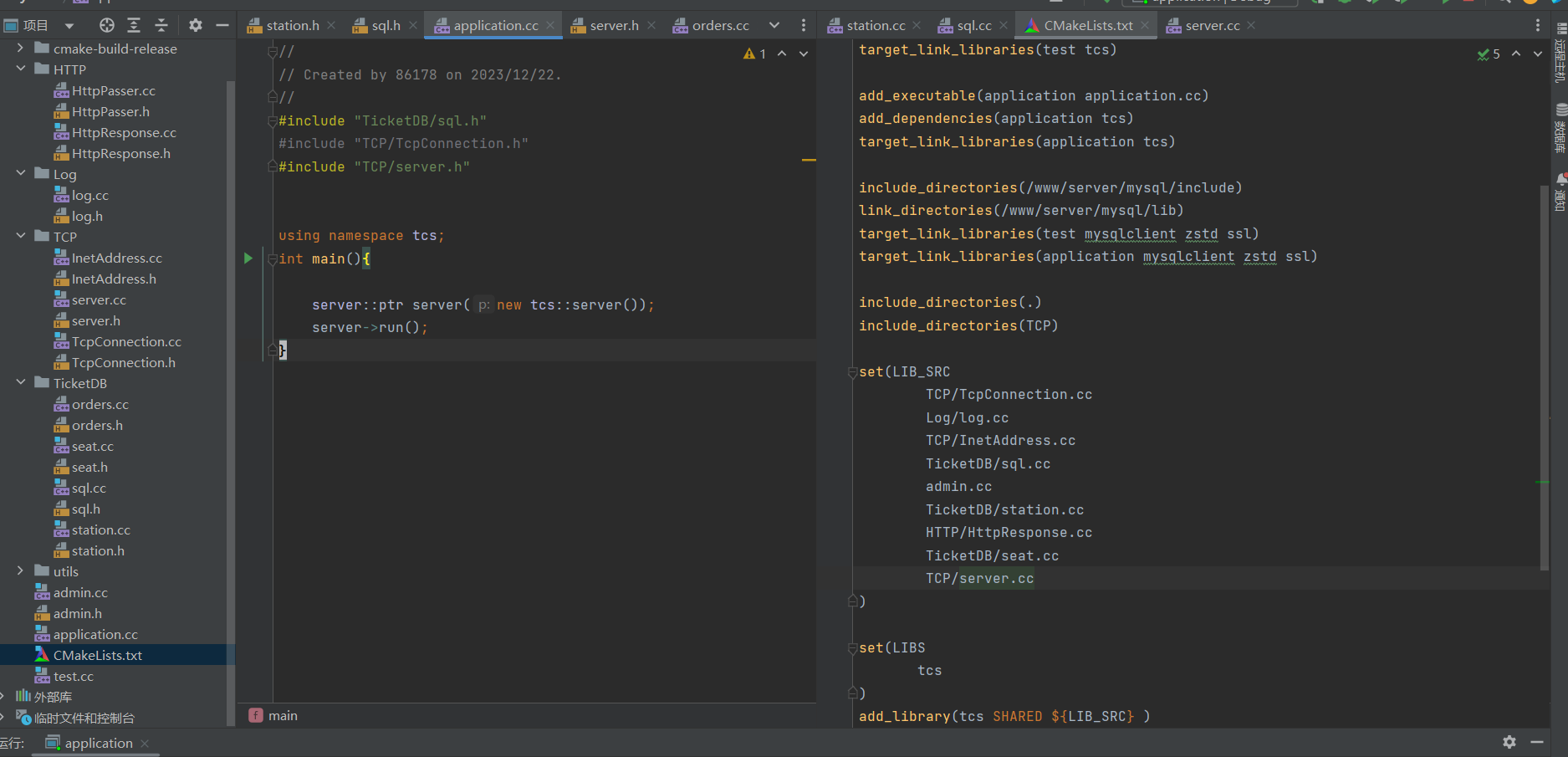
站点表单，记录车次，第一个数据为某车次的车次号

生成的订单数据在数据库显示，包含购票的用户、站点和价格信息：



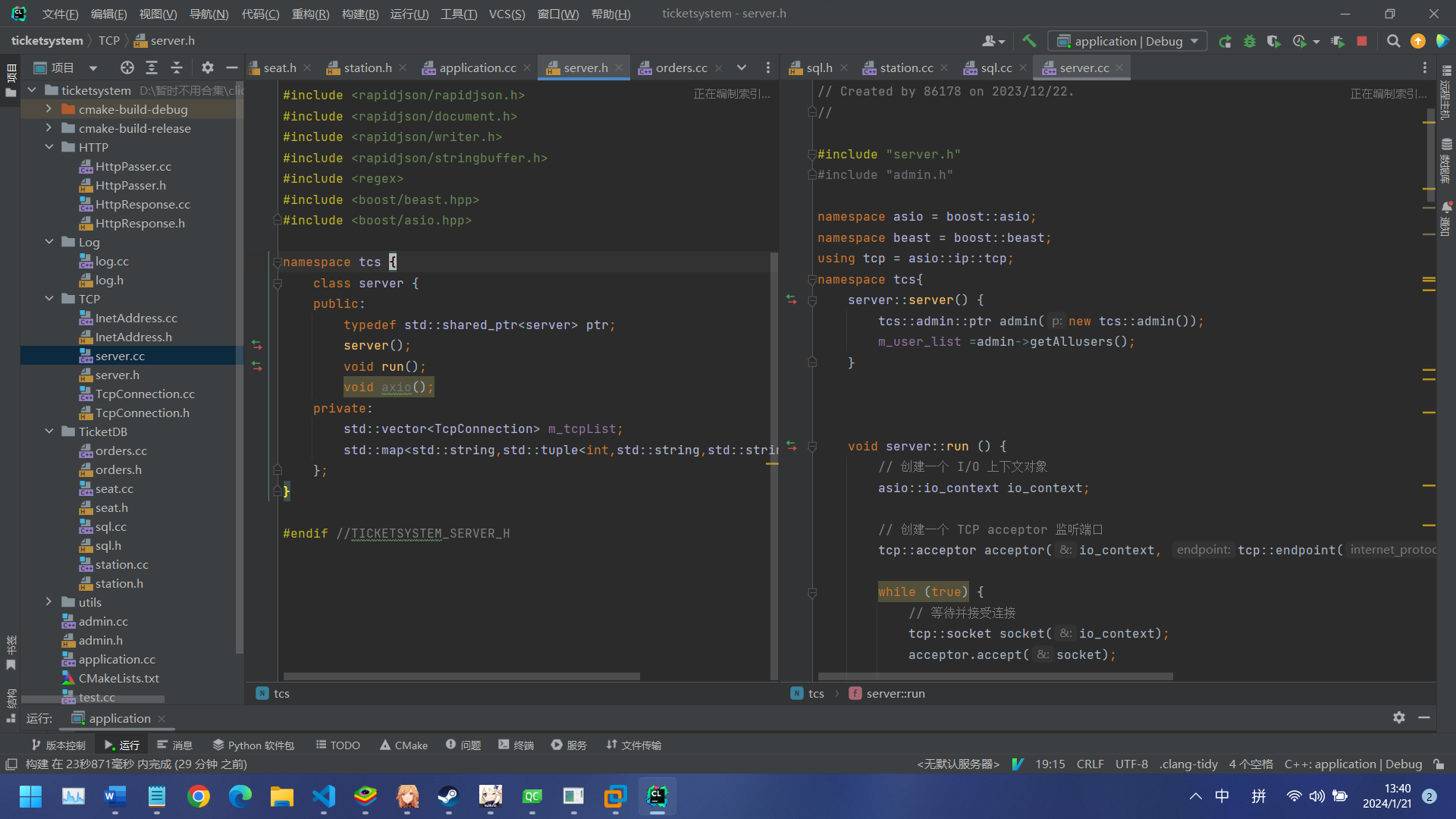
**后端服务器设计思路和实现：**

1. 使用cmake管理和链接项目，application类作为服务器程序启动入口类，运行后服务器启动。左图项目可见该项目创建了二十个类，代码量在两千行左右。代码较多，这里仅展示核心的部分



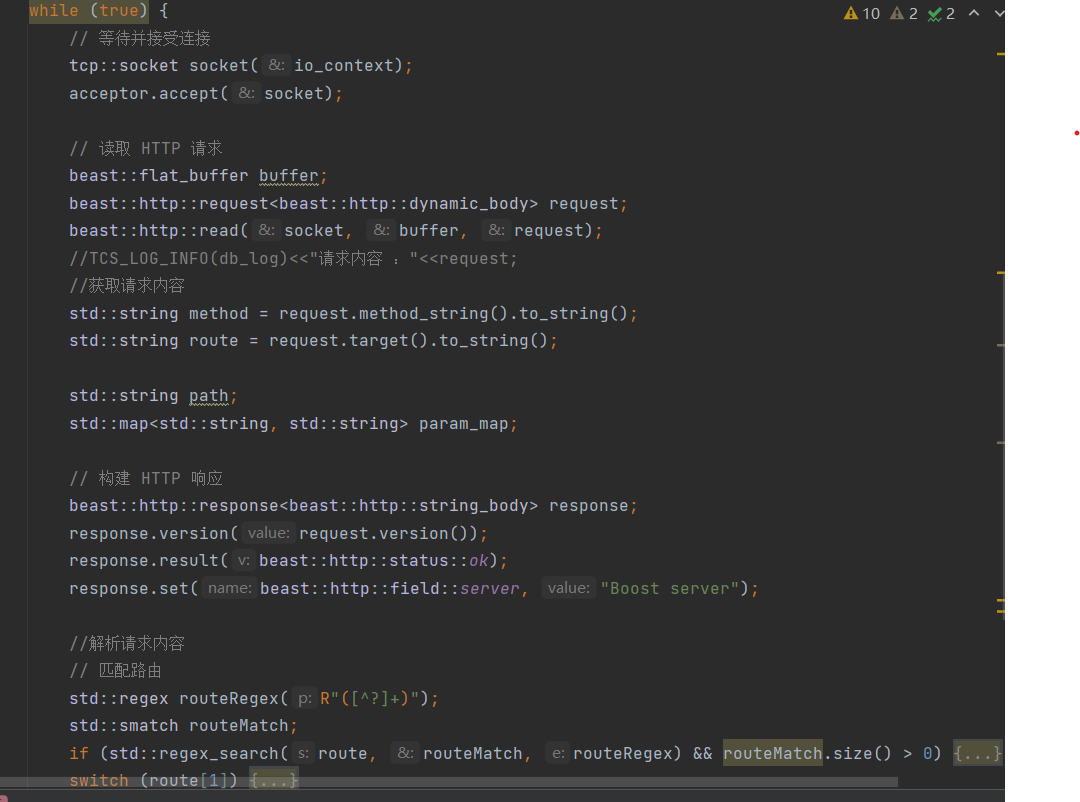
1. 服务器代码结构：日志系统类、数据库操作类、服务器类、工具类、测试单元类。在项目启动后，将创建服务器实例，服务器循环监听系统8080端口来自前端的数据http请求，通过状态机解析请求、查询或修改数据库，返回相对应的http响应报文到前端。服务器通过socket来在系统内核创建文件描述符，通过文件描述符在8080端口进行的

部分代码如图：

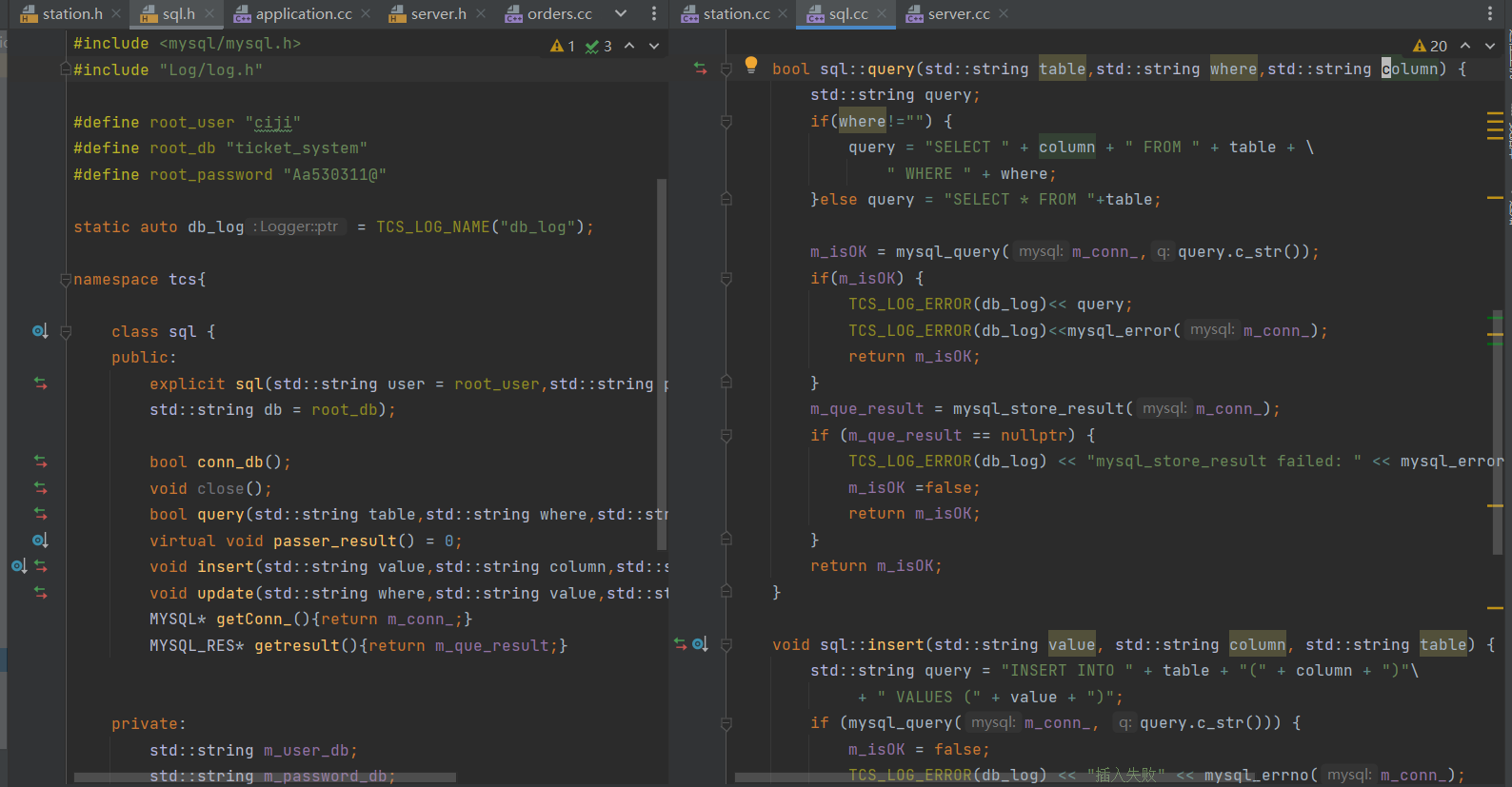


1. 具体实现：

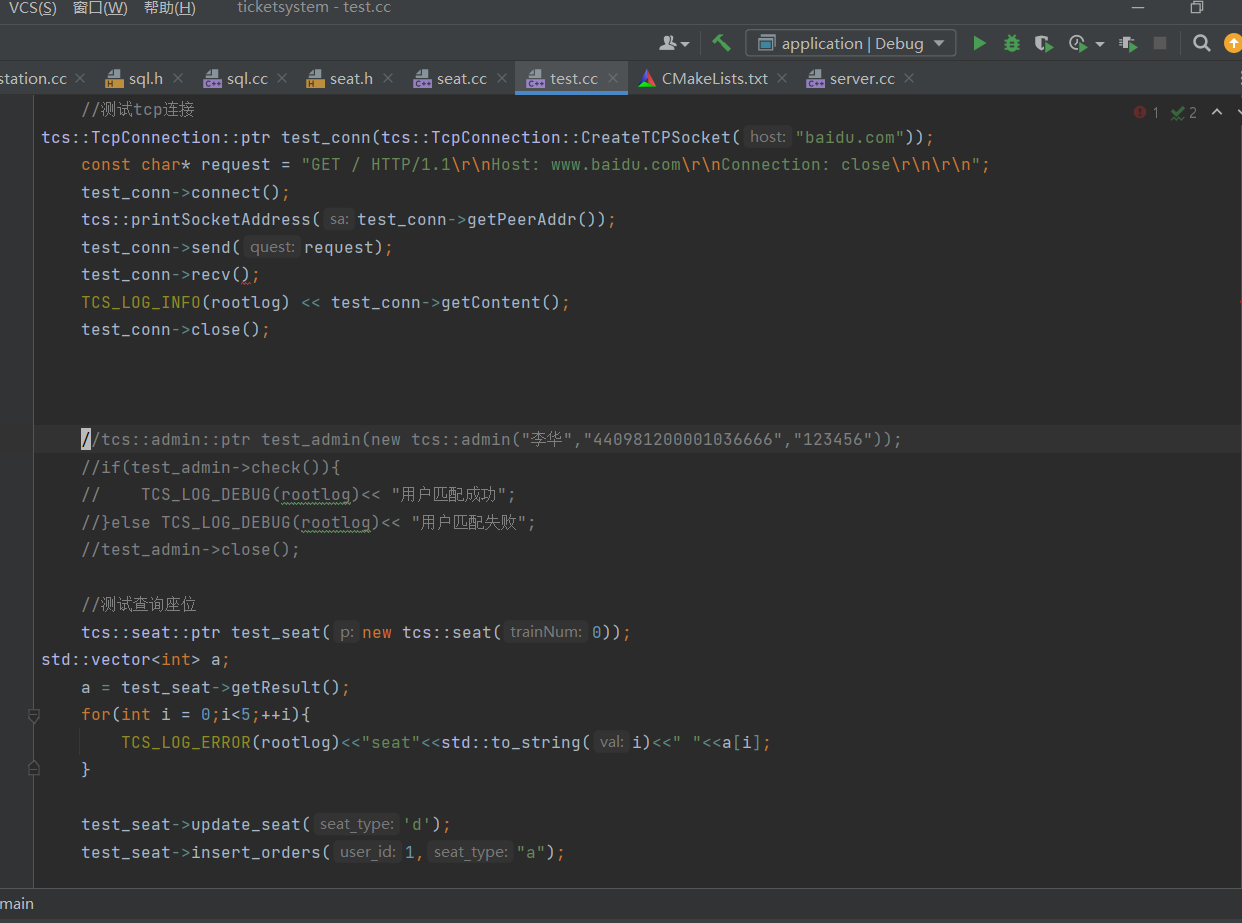
服务器部分：使用boost库来解析http请求，解析http头部部分，解析出具体查询内容，用switch语句跳转到相对应的处理方法。如果是查询请求将会用json格式将查询结果写入http响应，发送到前端。如果是修改内容，则会创建sql实例，修改数据库内容



数据库操作类部分：使用MySQL官方c++操作库，，先创建sql类作为通用查询语句，然后创建admin用户登陆类，station站点查询类，seat座位数量类，order订单生成类，这几个类继承sql类避免重复地写操作语句，减少开发工作量。Sql类如图



测试单元类，用于测试数据库查询语句是否成功查询、服务器启动、tcp链接等类，在测试一个类成功后再开发下一个单元，循序渐进，一步一步开发完善服务器类

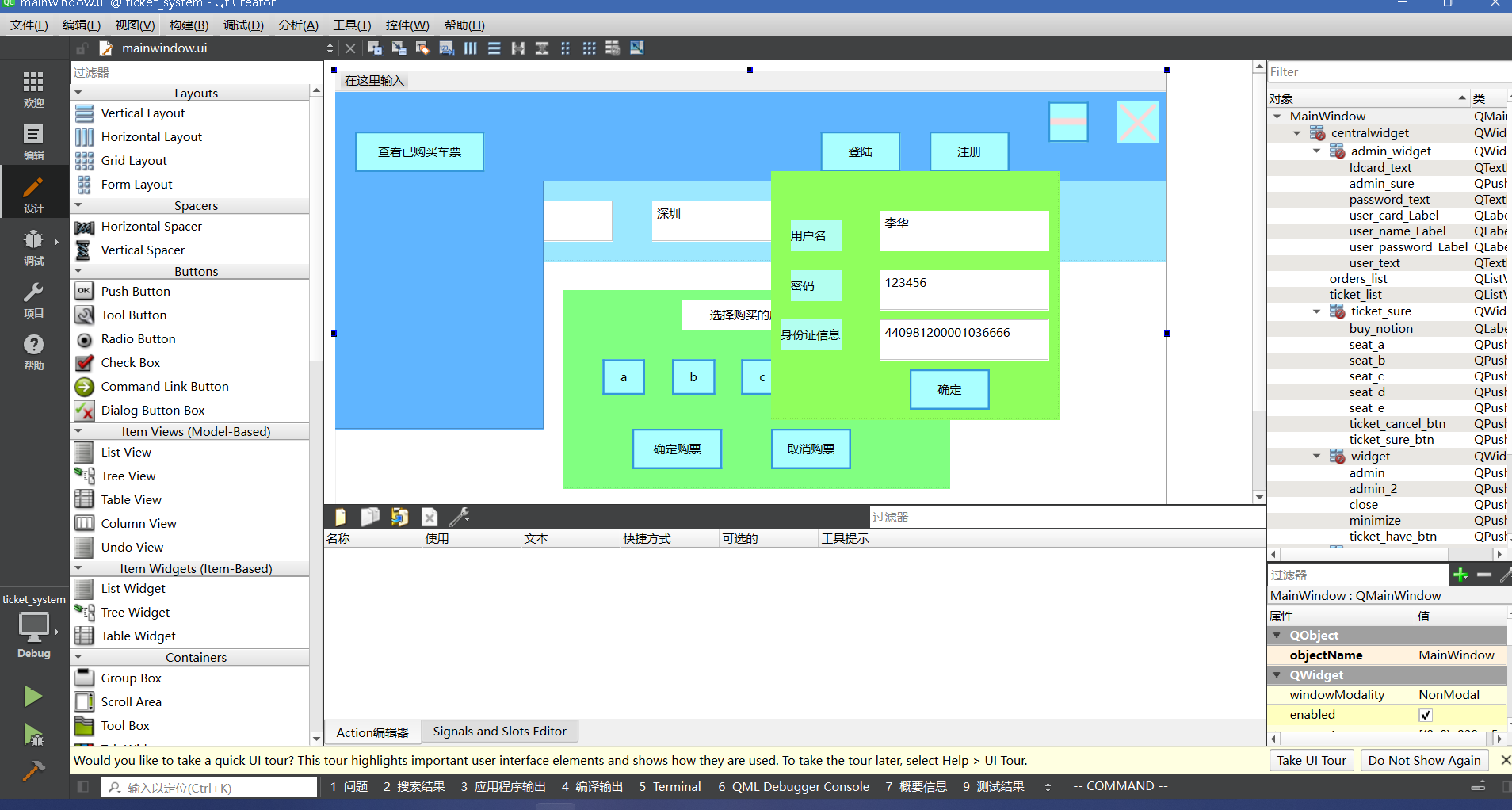


后端用于增加站点、增加座位数量等管理数据库的类已经完成，由于前端尚未实现后台管理端功能，便不在这里展示。所有测试都在Linux虚拟机上通过ssh远程连接进行，模拟实际的Linux服务器

其余代码篇幅限制不一一示例。

**前端代码设计和实现：**

1. 使用QTcreator提供的QTnetwork、QTWidget等组件便捷开发前端用户界面，先用QT提供的ui编辑器设计好一个基本的用户界面，再根据用户界面的使用逻辑顺序进行开发对应的操作代码。在程序启动时会创建admin实例用于保存用户个人信息的数据。



1. 登陆部分的实现：用户输入用户名、密码和身份证号，点击确定后，前端即客户端将发送用户名和密码到服务端进行验证，使用http的get方法进行发送，用json格式传输用户名和密码。实现代码如图  
   
2. 购票实现：分为查询站点、选择站点和座位、发送购票信息到后端三个部分。

查询站点信息部分：输入查询信息，点击查询后调用该函数



选择座位购买部分代码：选择座位后点击购买，调用该函数，该函数内部代码为发送请求到后端的代码



1. 前端程序主要函数如图



**五．项目总结**

**完成部分：**该项目设计好售票系统数据库，顺利完成前端查询站点和购票功能，并额外增加了用户登陆功能和订单保存和查询功能。使用数据库保存和Linux运行和测试服务器，接近真实生产环境。数据库。数据库将后面所需的管理端、具体票务服务的数据打下了基础，后续可以很好的扩展。前端由于教学进度没到web开发，所以用qt创建软件作为代替，后续可根据需求制作网页端。后端完成了数据库修改和监听前端数据请求功能，类的实现遵循可扩展设计模式，方便后续售票系统的进一步扩展功能。

**不足之处：**由于http协议具体内容和数据库内容尚未深入学习，在本项目中只能凭借自学的部分计算机网络和数据库原理来进行编写代码。部分缺点如https加密处理、数据库查询优化和redis缓存等尚不能处理。