“C++程序设计与训练”课程大作业

项目报告

项目名称：酒店预订与管理系统

姓名： 程晔安

学号： 2017011627

班级： 自 72

日期： 2018 年 8 月 3 日

目　录

1 系统功能设计

在科技日益发展的今天，许多的传统服务业都面临产业结构的转型和面对新市场的自我改变。其中，服务业面对的挑战尤为严重。市场特点的变化、人们需求的增长、自身技术转型的需求等等，都是服务业在科技发展的浪潮中所面对的挑战。因此，如何提高顾客消费时的自主性，提高服务效率和质量，带给顾客更好的体验，成为商家的一个重要关注点。

酒店预订是旅游出行的重要环节，然而当前酒店行业存在管理混乱、信息不够透明等诸多问题。另外，随着民宿不断火热，这一问题将会更加明显。建立一个有效的酒店信息管理平台，有助于规范酒店和民宿等的经营秩序，在给用户提供方便的同时，营造出各个酒店公平竞争、接受合理审核和信息公开的良好环境。其中，顾客和酒店在平台上交换必要的信息，由平台进行信息处理和传递，最后完成交易，起到了信息综合汇总、创造交易条件的作用。

本项目参考携程网的结构，创造了一个模拟网上预订酒店的信息平台。本系统将采用**单用户模拟多用户系统**，是的不同用户登录时能够使用自己的功能并且查看其它用户发来的消息。一再实现以上几点要求，同时顾客在界面上做到美观简洁，以带给用户更好地订购体验。平台管理员界面做到简单明晰，已达到操作的效率和管理上的方便。

1.1 总体功能描述

本部分将分几个模块，介绍本项目各个对象的功能及实现流程。

1.1.1 平台顾客各项功能及实现

图 1　按照规定关键字搜索酒店

图 2　带有图片显示的房间列表显示

（1） 注册

顾客设置账号，并提交手机号码作为之后获取订单之后的动向等信息用，即可登录系统。

（2） 登录

使用之前的账号信息，即可登录系统

（3） 查看酒店信息

用户输入所想要的酒店城市、地址、房型、订房日期、订购天数，经过查询返回相应的酒店信息。

（4） 查看房间信息

在显示了酒店信息以后，用户可对某酒店的详细信息进行进一步查询。包括该酒店所拥有的所有符合条件的房间信息。

（5） 订购房间

顾客可以构建订单，订单含有顾客信息和房间信息，并提交到平台，由酒店和平台管理员进行其他操作。

（6） 查看我的订单列表

可以查看所有订购人为该用户的订单、订单状态、退款审核状态等。

（7） 为预定的房间付款

可以在预订房间后进行付款

（8） 提出退款申请

在未入住前提出付款申请，并交与平台管理员审核

（9） 评价房间

在退房后可以对所住过的房间进行评价。评价的结果加入酒店和房间信息中，用于之后其他用户的订购时显示。

（10） 更改密码

在经过手机验证后，可以更改账号的密码。

1.1.2 平台酒店各项功能及实现

图 3　酒店所属房间的列表

图 4　创建房间界面

（1）　**提交创建酒店申请**

提交相关信息之后，可向平台管理员递交创建酒店申请。等待审核后方可登录。

（2）　**登录酒店管理系统界面**

在通过审核后，可以登录到酒店管理界面。

（3）　**新建房间**

可以输入新建房间信息，等待审核。审核结束后方可在用户的界面中显示，接受查询和预订。

（4）　**查看已有房间列表**

可以查看已有的房间的信息列表

（5）　**更改已有房间信息**

更改已有的房间信息，再次等待平台审核。

（6）　**上传房间图片**

可以在本地文件中上传房间对应的图片信息。

（7）　**更改房间图片**

更改房间已经有的图片信息

（8）　**查看所有已有房间相关订单**

查看自己所拥有的房间接受的订单信息。同时可以获得订单状态。

（9） **改变订单状态：确认入住**

可以确认订单状态为“入住”。

（10） **改变订单状态：确认退房**

可以确认订单状态为“退房”。

1.1.3 平台管理员各项功能及实现

	顾客账号	酒店姓名	房间类型	住房日期	住房天数	订单状态	是否允许退款
1	kevin	华清	总统套房	10.08.2018	2	已评论	否
2	kevin	华清	大床房	11.08.2018	3	退款	是
3	kevin	华清	双人房	07.08.2018	3	已评论	否
4	kevin	华清	双人房	14.08.2018	3	已预订	否

图 5　订单大列表

图 6　待审核房间大列表

（1）　登录平台

平台管理员房间不可注册，有唯一账号。利用唯一账号可登录平台管理员界面。

（2）　管理所有的酒店列表

可以看到所有在平台注册的酒店列表。

（3）　管理所有的房间列表

可以看到所有酒店所属的房间大列表。

（4）　管理所有的订单列表

可以看到所有在酒店管理平台上产生的订单及其状态。

（5）　审批通过酒店

通过审核相关信息确认酒店可以在平台登录

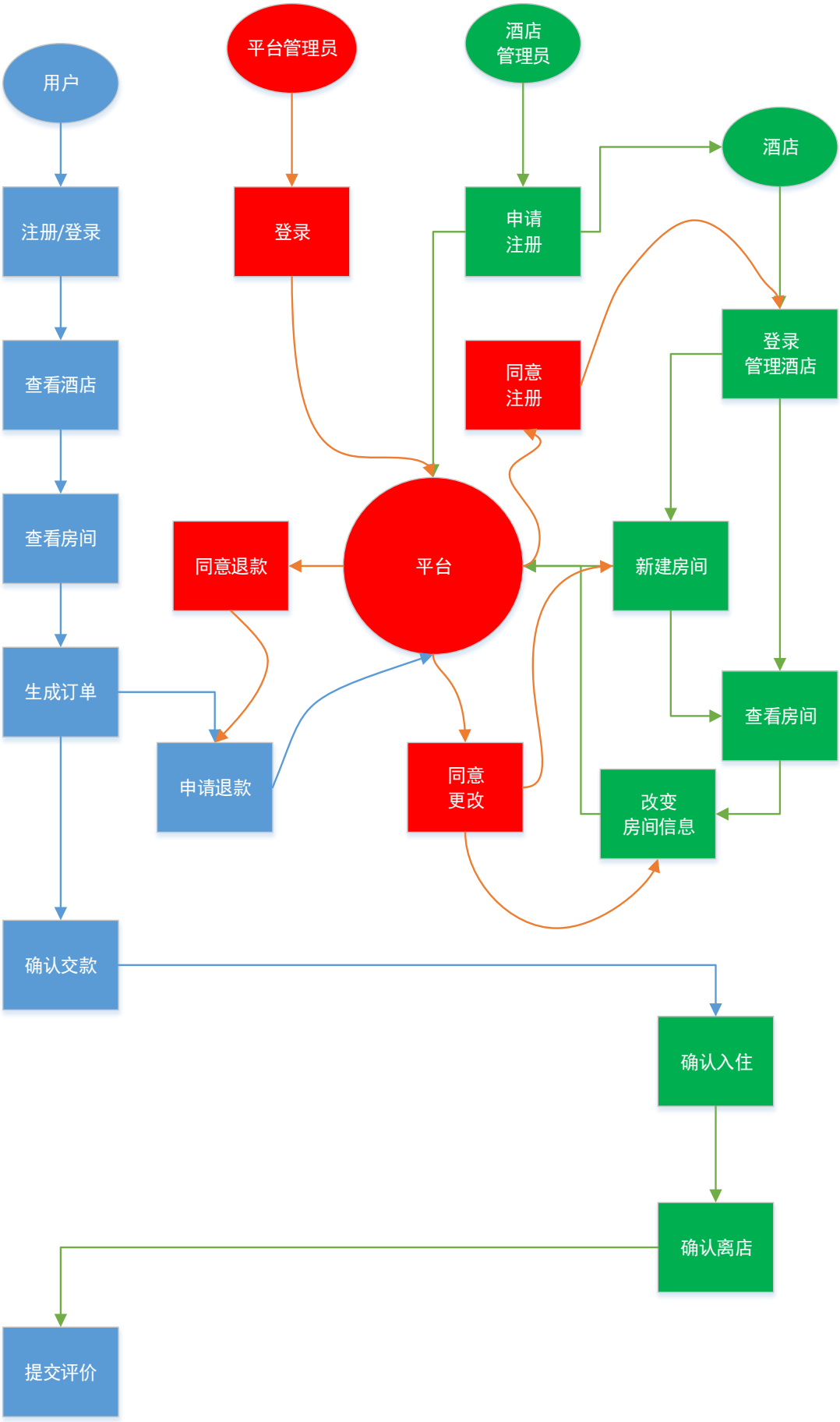
（6）　审批通过酒店房间

审核酒店上传的房间信息。包括首次上传和经过修改的。

（7）　审批通过退款信息

平台端有权决定是否审批退款操作。

1.2 功能流程描述


用户
平台管理员
酒店
管理员
酒店
注册/登录
登录
申请
注册
查看酒店
同意
注册
登录
管理酒店
查看房间
同意退款
平台
新建房间
生成订单
查看房间
申请退款
同意
更改
改变
房间信息
确认交款
确认入住
确认离店
提交评价

2 系统结构设计

2.1 整体架构设计

本项目的构造特点是采用了数据库进行存储数据，使用 Qt 进行编写图形界面。为了保证程序具有较高的可移植性，考虑到本次平台的搭建是一次模拟搭建，数据量不会太大，不用担心运行过程中的内存过大问题，又考虑到数据库仅作为存储数据用，本项目采用如下设计：程序启动时，从数据库中一次性读出所有数据，放在容器中，在程序结束的时候再将修改过的信息一次性从容器中写入。只利用 SQL 进行数据库的读、写功能。这样一来，程序与数据库的耦合性变得很小，因此程序得到了较高的可移植性。也为之后可能的接口变化（例如连接网络或者其他程序）留下充足的余地。

本项目的主体部分是自己编写的 C++代码和 Qt 基于模板（类似于）的第三方容器类编写而成；最后在装饰过程中，应用了一些基于 CSS 的 QSS 装饰表进行界面美化；在实现酒店房间照片存储时，为了贯彻加强程序可移植性，采用了将照片直接存在工程文件中的操作，不作只能在本地运行的路径调用；在界面累的功能转换时，尽量应用 Qt 特有的信号-槽机制。充分利用了各种知识点。

2.2 模块架构设计

由于本系统采用单用户模拟多用户系统，因此每个类至少有多个对象。要用到容器。

由于核心的类的接口和功能均为自己设计，不能使用 Qt 的信号-槽机制。本项目决定使用 QVector 全局变量来作为容器。

本项目主要分为两个模块：围绕酒店房间进行的信息上传、审核、查询模块；以订单为核心的存储多方信息，多方改变订单生命周期的管理和查询模块。

订单类与顾客、酒店和酒店房间都有密切的关系，酒店房间类更是整个平台所有操作的主要对象。因此需要各个模块、各个用户相互密切联系才能完成相应的操作。这些操作相互组合，最后构成了系统这一整体。

2.3 类总体设计

不包括各种界面类，本项目一共设计了 8 个基本类：AbtractUser 类（所有用户类的基类）， Customer 类（顾客类）， PlatformAdmin 类（平台管理员类），HotelManager 类（酒店管理员类）， myDatabase（数据库操作类）， Order（订单类）， Hotel 类（酒店类）， Room 类（房间类）。 UML 类图描述这些类如下：

1、 AbtractUser：

AbstractUser
#Account:Qstring//账号 #Password:Qstring//密码 #Phone:Qstirng//手机号

+AbstarctUser(newAccount:Qstring, newPasssword:Qstring, newPhone)//构造函数
~AbstractUser()//析构函数
+GetAccount():Qstring//获取账号
+GetPassword():Qstring//获取密码
+GetPhone():QString//获取手机
+IsCorrect(QStirng aPassword):bool//判断密码是否正确
+SetPassword(QString aPassword):void//更改密码
+IsPhoneUsed(QString newPhone):bool//检查手机是否用过

Customer、PlatformAdmin、HotelMnanger 均继承自 AbstractUser 类

2、 PlatformAdmin

PlatformAdmin
#Account:Qstring//账号 #Password:Qstring//密码 #Phone:Qstirng//手机号

+PlatformAdmin(newAccount:Qstring, newPasssword:Qstring, newPhone)//构造函数
~PlatformAdmin()//析构函数
+GetAccount():Qstring//获取账号
+GetPassword():Qstring//获取密码
+GetPhone():QString//获取手机
+IsCorrect(QStirng aPassword):bool//判断密码是否正确
+SetPassword(QString aPassword):void//更改密码
+IsPhoneUsed(QString newPhone):bool//检查手机是否用过

3、 Customer

Customer
-totalCost:int//一共的消费

+Customer(QString newAccount, Qstring newPassword, QStirng Phone, int TotalCost)//构造函数，用于读入数据库中数据
+Customer(QStirng newAccount, QString newPassword, QString int TotalCost)//构造函数，用于程序中构造新的Customer类
+GetTotalCost():int//获取每个人的购物车总价
+SetTotalCost(int newTotalCost)//设置每个人的购物车总价

4、 HotelManager

HotelManager
-QString Lisence//注册酒店时所用经营执照号 -QString HotelName//酒店管理员所对应的酒店
+HotelManger(QString Account, QStirng Password, QString Phone, QString Lisence, QString HotelName)//构造函数 +GetLisence():Qstring//获取营业执照 +GetHotelName():Qstring//获取酒店名称

5、 myDatabase

myDatabase
+db:QSqlDataBase//创建链接用
-CreateConnetion():bool//创建与数据库的链接 -CreateTables():bool//创建数据库中的表 +myDatabase()//构造函数 +InitialCustomer():bool//初始化顾客容器 +InitialHotelnanager():bool//初始化酒店管理者容器 +InitialHotelAndRoom():bool//初始化酒店容器 +InitialOrder():bool//初始化订单大列表 +InitialRoom():bool//初始化房间大容器 +InitialPlatformAdmin():bool//初始化平台管理员列表 +Memory():bool//将容器中的信息写入数据库

除去其他基本类型，myDatabase 是本程序设计的一大亮点。在本类中，封装了所有与数据库相关的所有操作，在打开登录界面的时候将所有信息导入全局变量中，再在最后程序退出时从全局变量中将信息由 Memory（）函数导入数据库中，体现了功能的封装性。

6、 Order

Order
-QDate Date//订单开始日期 -int Days//订房天数 -int Type//订单的状态 -bool RefunApproved//是否退款被允许 -Room* pRoom//指向订单房间的指针 -Customer* pCustomer//指向订单顾客的指针

+Order(QDate newDate, int newDays, Room* new_pRoom, Customer* new_pCustomer)//构造函数
+Order(QDate newDate, int newDays, int newType, bool RefundApproved, Room* new_pRoom, Customer* new_pCustomer)//构造函数
+GetRoom():Room*//获取房间指针
+GetpCustomer():Customer*//获取顾客指针
+SetType(int newType):bool//设置订单种类
+GetDate():QDate//返回订单开始日期
+GetDays():int//返回订房天数
+GetType()int//返回订单种类
+GetRefundApproved():bool
+SetRefunApprevde(bool newApproved):void

作为一种基本类型，Order 的情况最为复杂。本程序设计其拥有指向相关房间和顾客的指针，这使类与类之间的关系更为密切，同时为获取数据提供了很大方便。也不用承担如果只存储字符串信息的相关时间复杂度。

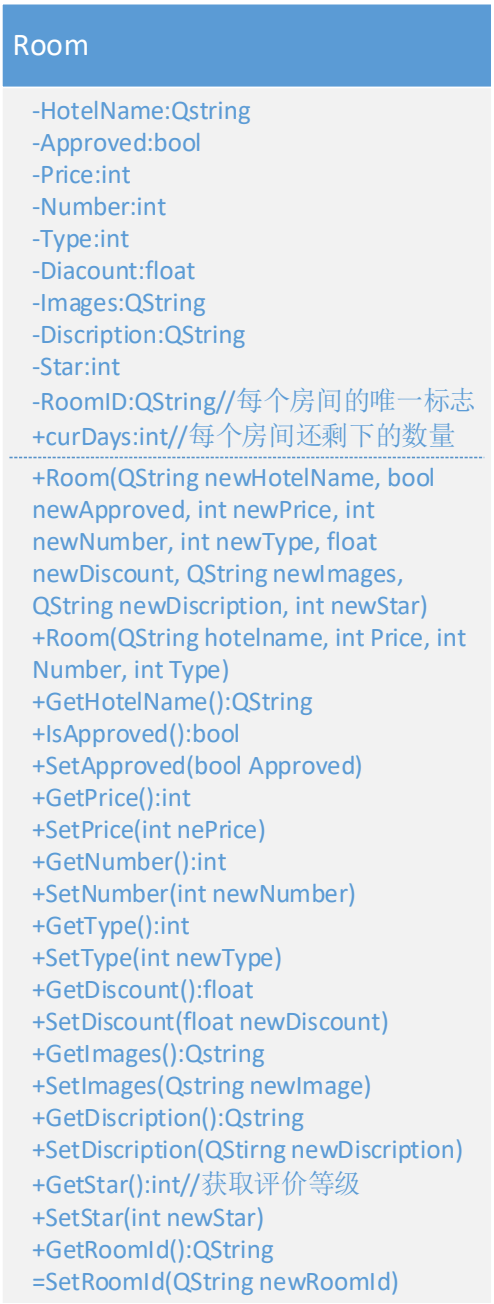
7、 Hotel

Hotel
-HotelName:QStirng//酒店名称 -Address:QString//酒店所在城市 -District:QString//酒店所在地区 -memberName -ConsolePhone:QString//酒店联系电话 -Approved:bool//酒店是否过审 +Roomlist:QVector<Room>//属于酒店的房间列表

+Hotel(QString hotelname, QString address, QString district, QString consolePhone, bool Approced)//用于数据库的构造函数
+Hotel(QString hotelname, QStirng address, QString district, QString consolephone)//用于程序内创建对象构造函数
+GetHotelName():QString //获取酒店名称
+GetAdress():QString// 获取酒店所在城市
+GetDistrict():QString //获取酒店所在地区
+GetConsolePhone():QString//获取酒店联系电话
+IsApproved():bool//获取酒店是否过审
+SetApproved():void//审查酒店
+SetRoomlist(QVector<Room>newRoomlist):void//重置酒店所拥有的房间容器
+GetStar():float//获得酒店综合评价
+GetMinPrice():int//获得酒店房间中的最低价格

Hotel 类大胆采用了 QVector 作为自己的共有数据成员的做法。这样的做法有利于直接获取酒店所拥有的房间的信息，同时从抽象的角度上来说，房间总是隶属于酒店，酒店的操作总是针对自己的酒店列表，符合现实抽象。
同时，本类的设计亮点是酒店本身不会被顾客评分，评分的主体是房间，在显示酒店的评分时，其实是在显示其所拥有的所有房间的评分的平均值。

8、 Room



本类的设计亮点是共有数据成员 RoomId 和私有数据成员 Image。RoomId 实际上是酒店名+房间类型名。有了这样的 Id，就形成了房间之间的唯一标识。也为数据库存储提供了主键。Image 其实储存的此酒店图片的相对地址。程序中真正将图片从本地复制到指定的目录下，将其路径存储之，既提高了程序的可移植性，又省却了将图片存入数据库的复杂度。

3 系统详细设计

3.1 信息存储方式设计

本项目采用一次性存储如内存的做法。因此对容器的选择是很重要的一步。本项目采取 QVector 全局变量作为容器。

```
//声明全局变量
QVector<Customer>g_Customerlist;
QVector<HotelManager>g_HotelManagerlist;
QVector<Hotel>g_Hotellist;
QVector<Order>g_Orderlist;
QVector<PlatformAdmin>g_PlatformAdminlist;
```

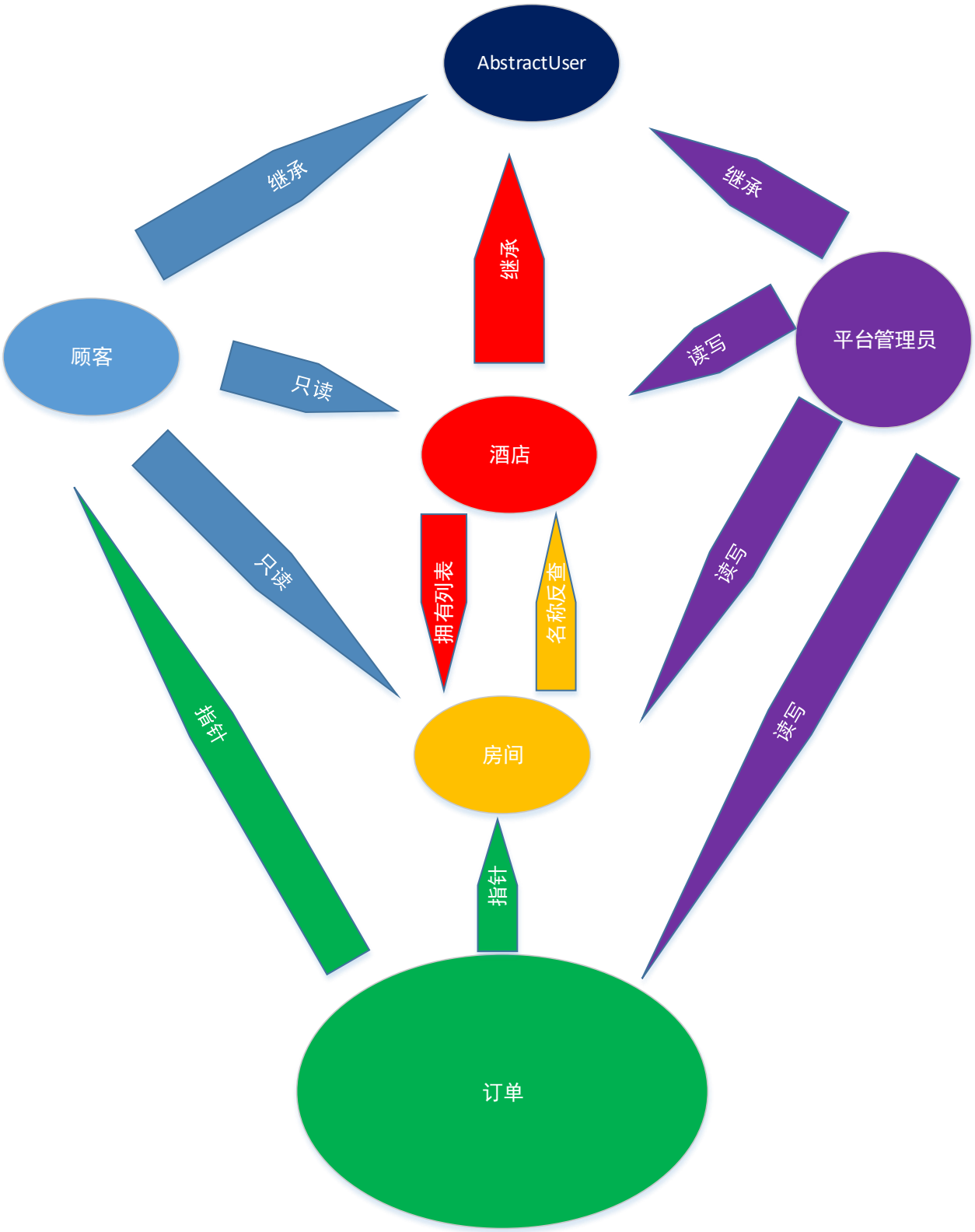
图 7　全局变量容器

为什么选择 Vector 容器而不选择 List 容器呢？在经过查阅手册，结合本项目对这些全局变量的功能要求来综合考虑，我最终选择了 Vector 容器。理由如下：

- Vector 容器对随机存取的效率较高，可以下标操作，使用起来较方便。
- 本项目对全局变量的操作大多体现在对某实例的某个特征进行修改，或者就是对在尾部添加元素。不涉及在中间插入和删除的操作，规避了 Vector 容器缺点。
- 本项目使用指针较多，Vector 的地址存放是连续的，可能会降低出现野指针误操作的风险。

3.2 类结构设计

3.2.1 类间结构设计



3.2.2 类内结构设计

各个类几乎所 有的成员都以私有成员的形式出现。再在公有函数中提供相应的接口，以便信息的传递。而信息的读写属性的决定权主要由平台内部的操作

决定。因此，界面显示的时候每个用户主体都能且只能看到自己需要的信息。实现了信息的保护。

3.3 界面结构设计

界面设计大部分采用 QDesigner 进行可视化设计，同时应用了 QSS 添加了整体程序的风格特点。整体来看，初步达到了“酷雅扁平化”的设计风格初衷。本项目一共有 10 个窗口，每个窗口之间都有间接或直接的联系。同时设计了大量的“刷新界面”按钮，使用户的操作可以很快进行显示和更新。要注意不能直接关闭窗口退出应用程序，这样应用程序的全局变量信息可能来不及传入数据库中。应当使用保存并退出调用 mydatabase 类的 memory 函数，再能进行数据的保存。

对于界面的信号传递，本程序主要有三个关键点：

1. **应用 Qt 特有的信号-槽机制，使得不同的信号和输入指令都能得到相应的处理。**下图是一处信号槽声明的举例。

```
//接口
//tab1
//查找对应酒店
connect(ui->btnSearch, SIGNAL(clicked(bool)), this, SLOT(SearchARoomP1()));

//显示酒店对应房间
connect(ui->btnDetail, SIGNAL(clicked(bool)), this, SLOT(hide()));
connect(ui->btnDetail, SIGNAL(clicked(bool)), this, SLOT(ShowDetail()));

//保存并退出
connect(ui->btnExit, SIGNAL(clicked(bool)), this, SLOT(SaveAndLogout()));
//tab2
//刷新
connect(ui->btnRefresh, SIGNAL(clicked(bool)), this, SLOT(ShowOrder()));
//确认付款
connect(ui->btnPay, SIGNAL(clicked(bool)), this, SLOT(Pay()));
//申请退款，由平台管理员控制
connect(ui->btnRefund, SIGNAL(clicked(bool)), this, SLOT(Refund()));
//提交相应评价
connect(ui->btnComment, SIGNAL(clicked(bool)), this, SLOT(Comment()));
```

图 8　MainWindow 中的信号槽

2. **应用了局部变量---指针容器对相应数据进行存储和处理**
   在使用界面处理相应的数据时，有时候只是需要数据的副本进行传递操作，有时候需要更改全局变量中的数据。因此，本程序采用的方法是先将需要处理的数据传入局部变量中，建立一个局部变量对应的**指针 Vector**。这样处理可以在满足数据传输的基本要求的同时，尽量少占用储存空间。不同界面之间的数据传输也尽量都是用指针的形式。所有指针都是指向全局变量相应数据的位置的，指针之间的赋值也都是以浅复制的形式进行。这样做最大程度上保证了空间不被局部的类变量占用，不会产生赋值等操作时实际操作的是局部变量这种错误。同时还允许了两个容器以指针的方式传递数据的操作。这种处理方法充分利用了指针的灵活性。
3. **使用指针串联所有相关窗口。**

窗口之间的转换和跳转是以形参-局部变量的形势进行的。这样的处理首先可以明确看出各个窗口之间的关系，可以实现两个窗口之间的跳转。同时大大减小了野指针的产生可能性。图 9 为这种传递方式的一个例子。

```
ChooseRoom::ChooseRoom(MainWindow *newParent) :
    parent(newParent),
    ui(new Ui::ChooseRoom)
```

图 9 以参数形式记录窗口之间的关系

下面是界面具体实现。主要分为顾客界面、酒店界面、平台界面、登录界面四个大的主要界面介绍。每个界面中都有主要和自己相关的子界面，这里就不单独拿出来介绍了。

3.3.1 顾客界面设计与实现

- 顾客界面是本次界面的设计重点。
- 顾客界面继承自 QMainWIndow，接口转换情况是可以到达 wComment（评论界面）、ChooseRoom（选择房间界面）、loginWindow1（登录界面）。
- 顾客界面有私有容器如下：

```
//创建临时酒店指针列表
QVector<Hotel*>temp_pHotellist;
//创建临时订单列表
QVector<Order*>temp_pOrderlist;
```

分别记录针对单个顾客以及其搜索结果而生成的酒店和订单指针列表。

- 具体实现上来看，利用 QtabWidget 创建了三个 tab 页，其中信息的罗列使用 QTableWIdget 控件。分别用作酒店信息查询、订单查询和信息维护用。图 10 为实际界面。

- 图 10 用户界面

具体功能如下：

- 搜索房间：用户输入城市、地区（选填）、房型（选填）、订房开始日期、入住天数、单击查询，就可以查询符合条件的酒店，并根据相应的排序方式排序显示，先不显示对应的房间。这样的设计是参考携程网的设计。其中，显示符合条件的酒店是这次涉及的难点。本项目的处理方法是：首先从酒店信息中得到符合要求的城市和地区要求的酒店，再遍历酒店所属的房间向量，如果有符合房型，就通过了第二遍筛选。最后，遍历

酒店对应的房间对应的订单，订单中记录了时间信息，从遍历中筛选酒店房间在该时间已经被预订了几间，最后获得房间剩余量大于零的才可显示。

- 查看详细信息：在 QTableWidget 中点击中意的酒店，再点击查看详细信息，就可以找到满足条件的房间的详细信息。这里本项目的闪光点是实现了房间图片的显示。如果用户在这里点击现在订购，那么就会产生生成订单，传入全局变量中。
  图 11 为实际的效果图。

图 11 房间列表

- 改变订单状态：
  可以在订单页面进行查看所有的订单，并且可以实现交款、申请退款等操作。这些操作可以改变订单的相应状态，传入其他用户进行操作。
- 评论
  在订单结束后可以进行评价。

图 12 评论界面

- 更改密码
  在进行密码验证之后进行更改账号的密码

3.3.2 酒店界面设计与实现

- 酒店界面继承自 QWidget， 接口情况是可以到达 loginwindow1（登录界面）， newRoom（新建房间界面）， Roomchange（更改房间属性）界面。
- 酒店有私有指针容器 QVector<Order*>temppOrderlist；以用来从全局变量中获取与此酒店相关的订单列表。
- 有通向其他界面的指针

具体功能如下：

- 在设计方面，有两个 tab 页分别可以显示酒店对应的房间列表和酒店对应的订单列表。
- 在房间显示页中，可以新建或者修改酒店的房间信息。如下图。

图 13 房间列表

- 新建房间：本项目的特点是可以实现照片的上传和保存。照片在保存时，其实保存的是文件的相对路径，再利用相对路径调用照片进行显示。值得注意的是，本程序真正实现了保存照片，即保存在/debug/photo 文件夹中，使程序的可移植性大大增强。同时输入信息的时候有很多默认选填选项，还有默认的图片。程序会分析在创建的酒店房间类型是否已经存在，如果已经存在则不能创建。“打开图片”按钮可以预览图片；“保存图片”按钮可以根据图片名称将图片保存在./debug/photo 目录下，实现了可移植性。创建房间后，房间状态变为未审核，交由平台管理员审核。下图为 newRoom 窗口界面。

图 14 NewRoom 界面

- 修改房间信息：与新建房间的功能特点相似，值得注意的是老的房间信息以 HolderText 的形式显示，任何的修改都会让房间状态变为未审核，重新经过审核。
  下图为更改酒店信息窗口。

图 15　更改房间属性

- 查看订单列表并管理订单一部分的生命周期：酒店可以查看和自己有关的所有订单，并且可以确认入住和确认退房。下图为订单列表窗口。

图 16 订单列表

3.3.3 平台管理界面设计与实现

- 平台类继承自 QWidget，接口情况是可以到达 loginWIndow（登录页面）。
- 平台类主要的作用为查看和处理信息，因此可以直接查看全局变量中的信息不加筛选。
- 平台管理界面分为三个 tab 页，分别为酒店列表、房间列表、订单列表。个人认为平台无权获取顾客的个人信息，因此除了顾客的信息以外其他信息都可以查看到。
- 查看审核酒店信息：可以看到酒店的信息和是否审核，可以显示待审核的酒店，如果信息正确即可通过审核。如下图：

图 17 待审核酒店

- 查看审核房间信息：与酒店信息类似。如下图：

图 18 待审核房间

- 查看所有订单列表：平台对订单类拥有最完整的读取权限。可以查看所有历史订单记录。并且可以审核退看操作。如下图：

图 19 订单列表

3.3.4 登录及注册界面设计与实现

- 登录界面继承自 QWidget， 接口情况是可以转至 registerWindow,（用户注册界面）、hotelregister（酒店注册界面）、MainWindow（用户主界面）、wHotel（酒店主界面）、platform（平台主界面）。
- 登录界面的主要功能有两个。第一是任何用户登录系统的必经界面。二是作为连接各个窗口的纽带，拥有各个主要窗口的指针，能够转向的界面比较丰富。界面如下图。

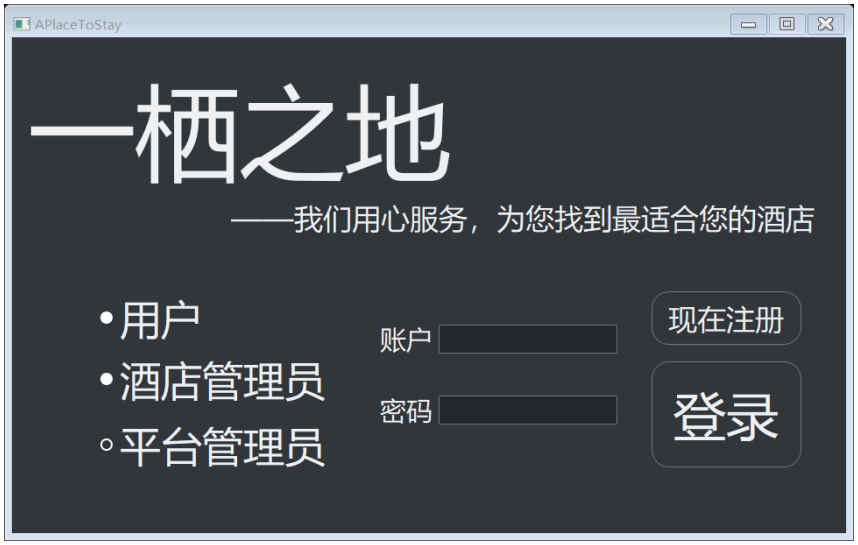


图 19 登录界面

- 登录界面的主要功能分为登录和注册。其中登录即为选择三种窗口进行登录，登录时遍历已有账号信息，给出能否登陆的判断。
- 登录界面可以注册用户界面和酒店管理员界面。平台管理员账号不能注册。因为平台中的信息安全和平台的操作权限。已有的平台管理员账号为：账号:kkk 密码：1234567890
- 用户登录界面其实主要的实现重点是容错的判断。同时注册前还会和已有的账号信息进行比对，防止重复注册。如下图：

图 20 顾客注册界面

- 酒店管理员注册大体思路和用户差不多。注册完成后，会生成酒店管理员和酒店两个新实例。分别存储，等待审核。界面如下图。

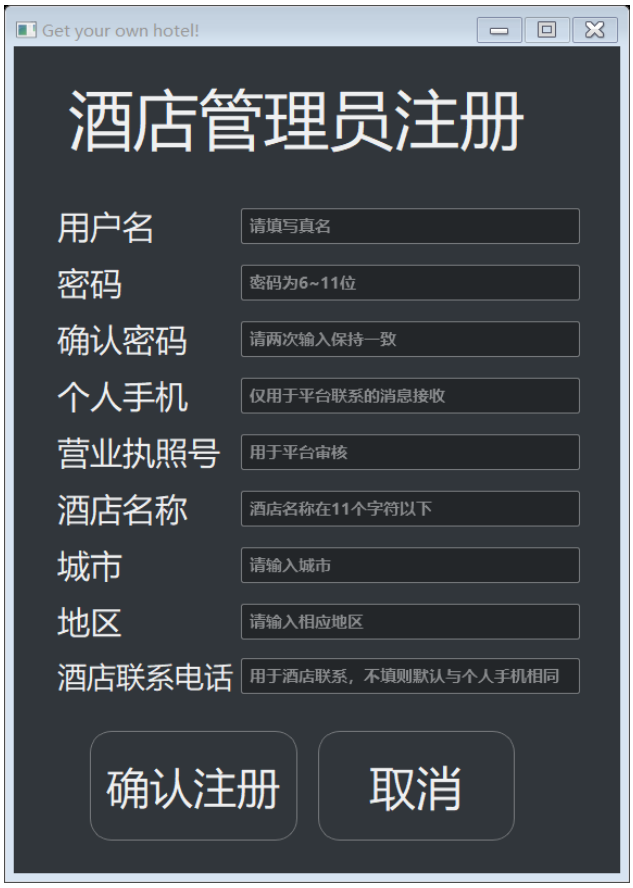


图 21 酒店管理员注册

3.4 关键设计思路

本项目的关键设计思路主要体现在整体架构等方面上，前面已经提到过，在此进行总结：

3.4.1 全局变量

本项目要应用的所有数据（包括用户信息、订单信息和酒店信息）都以全局变量的形式读入内存，放在 QVector 容器中。QVector 容器相对于 QList 在本项目中效率较高。在程序开始时一次性存入，再在程序结束的时候一次性存回。实现了较高的可移植性和程序可以脱开数据库运行的要求。数据库仅作为存储用。

3.4.2 面向对象的设计思维

本项目用 C++编写，将所有的对象都抽象为类进行处理，基本不使用数据库编程。所有对象均为数据抽象与行为抽象，同时各个类之间的关系也基本符合现实生活。例如所有的用户都继承于 AbtractUser 等。这是对现实的合理抽象，并具有一定的封装性。

3.4.3 指针容器的应用

在面对要将全局变量下的数据拿到局部来进行处理和显示的时候，本程序选择在局部建立指针 QVector 来存储数据。相当于建立局部的“指引”。这样处理不但可以完成实例 Vector 可以完成的所有功能，还能够保持数据的存储单一性，不容易产生赋值等更改操作误操作的情况。同时，由于存储的是地址，会减少内存的负担。

3.4.4 以状态刻画流程

本项目在处理订单的状态时，其实将状态和其生命周期一起考虑。订单的状态有已预订、已付款、退款、已入住、已退房（待评论）、已评价。这样的处理可以让不同的用户处理这个订的不同阶段的生命周期，同时让一个订单的流程十分清晰，易于管理。

3.4.5 指针连接操作

本项目大量利用指针进行不同平面、不同类之间的信息交流。这样处理的好处是首先指针存储的是地址，减少了储存的压力，其次可以直接访问到相关类相关数据，实现这种操作的另一种方法是进行友元声明，但是如果储存的只是相关类的关键信息，在应用时又要重新遍历 Vector 进行查找，这样做加大了运算负担。例如在订单类中找出房间所对应的酒店名称，就可以通过订单类的私有成员 pRoom->GetHotelName 很快完成。

其次是利用指针连接各个窗口之间的转换。以登录顾客界面为例：以形参形式传入顾客指针 pCustomer 和登录窗口指针 pLoginWindow 有利于保存登陆者的信息，同时可以轻松的跳回到登录界面中，准备其它用户的登录。

3.4.6 可移植性思想

前面介绍过的全局变量操作提高了程序的可移植性。同时，本项目在图片存储中也实现了可以脱离本机进行图片的显示并且不用数据库存储。实现方式是在数据库中以相对路径字符串的形式储存对应图片的地址，在进行打开图片、保存图片时，图片均在相对路径中。这样就提高了代码的重用性。

3.5 容错设计

由于本项目中需要用户输入的地方很多，包括注册信息时等等，使用不规则输入可能会导致存储的信息错误，也不利于平台上信息的整合。同时在之后很多列表操作中都有先选定列表中的一行在进行详细操作的要求。因此需要大量的输入和操作容错性检查。以顾客注册界面为例，如下图：

```
//判断
if(Password.isEmpty())
{
    QMessageBox::information(this, "entering Error", "输入密码不能为空", QMessageBox::Ok);
    flag = false;
}
if(Password.length()<6||Password.length()>11)
{
    QMessageBox::information(this, "length Error", "密码长度应在6~11位以内", QMessageBox::Ok);
    flag = false;
}
if(Password != Password2)
{
    QMessageBox::information(this, "different Entering", "两次密码输入不同", QMessageBox::Ok);
    flag = false;
}
if(Account.isEmpty())
{
    QMessageBox::information(this, "entering Error", "输入账户不能为空！",QMessageBox::Ok);
    flag = false;
}
if(Phone.isEmpty())
{
    QMessageBox::information(this, "entering Error", "输入电话不能为空！",QMessageBox::Ok);
    flag = false;
}
if(Phone.length()!=11)
{
    QMessageBox::information(this, "entering Error", "输入电话号码应为11位！",QMessageBox::Ok);
    flag = false;
}
return flag;
```

据不完全统计，整个程序中防止容错设计提醒的 MessageBox 约有 160 处左右。充分提高了平台的鲁棒性和容错性。

3.6 数据库设计

本项目利用数据库的第二范式进行设计，一共有六个 table，分别为 Customer、Hotel、HotelName、Order、PlatformAdmin、Room。具体设计如下：

Customer:

Account VARCHAR PRIMARY KEY;
Password VARCHAR;
Phone VARCHAR;
TotalAmount VARCHAR;//记录当前用户的消费总额

Hotel:

HotelName VARCHAR PRIMARY KEY;
Address VARCHAR ;
District VARCHAR;
Consolephone VARCHAR;
Approved BOOLEAN//标记当前酒店是否通过审核

HotelManager:

Account VARCHAR PRIMARY KEY;
Password VARCHAR;
Phone VARCHAR;
Lisence VARCHAR;
Hotelname VARCHAR;

Order：
    Date VARCHAR
    Days INT
    Types INT//订单状态
    RefundApproved BOOLEAN//是否允许退款
    CustomerName VARCHAR//顾客名字
    RoomID VARCHAR PRIMARY KEY
PlatformAdmin
    Account VARCHAR PRIMARY KEY
    Password VARCHAR
    Phone VARCHAR
Room
    Hotelname VARCHAR
    Approved BOOLEAN
    Pirce INT
    Number INT
    Type INT
    Discount FLOAT
    Image VARCHAR//文件存储路径
    Discription VARCHAR
    Star INT
具体可参见工程文件夹中的.db 文件。

4 项目总结

4.1 遇到的问题及解决方法

在开发过程中遇到的问题很多很多，甚至从安装开始，我就安装了十几次 Qt 都没有能够成功运行，每次编译的时候都会提示没有合适的 Kit。我在网络上找到了所有相关的帖子，这个过程大概持续了两周左右，一直没有解决问题。最后终于在又一次绝望的搜索中找到了问题所在。在各种配置环境、添加 Path 之后，终于可以开始编程了。

对我来说最有难度的地方还是项目刚开始。从书本上的知识到真正的应用项目编程，一开始总有无从下手的感觉。很多库函数和信号槽机制都需要学习。最后还是在同学们互相讨论和学习的过程中才慢慢起步，边学边编，一旦上手就容易一些。

还有比较有难度的地方是创建类。也就是基础架构上的问题。又一次因为没有思考成熟就下手取编程，结果后面发现 Room 类少了一个必要的元素——RoomId 即为 Room 的唯一标识。还要花很多精力去重新改写。

就在提交截止日期的前一天晚上，我还遇到了程序突然崩溃，无法编译的问题。结果最后不得不新建一个工程文件，再把所有的.p.cpp.ui 文件一个个重新添

加，再更改 Pro 文件中的设置，一个一个查找匹配，最后才解决了问题。

4.2 项目难点

我认为项目的难点是照片的选择存储以及在已有订单中由时间因素进行检索筛选的过程。订单的房间筛选我在本项目实现的过程中用了四重循环，才解决了问题。

4.3 项目亮点

- 功能强大。本项目实现了题目所有要求，为管理员添加了审核每次酒店信息更改后的审核权利和对于退款的审核权利；为用户添加了修改密码功能。这些功能虽然难度不大，但是对项目的完整性和流程上还是很有帮助的。还添加了很多刷新、查询功能。虽然添加了工作量和代码量，但是我认为是值得的。
- 充分利用所学知识点。本项目体现了 OOP 的抽象、封装、继承等多项设计思想和关键技术，充分利用课程内所学的 C++面向对象的知识：所有的用户类都继承自 AbstractUser 类， 使用了 Qt 基于模板的容器类，并且对多个类的构造函数进行了重载。在这里不用默认参数构造而利用重载函数的原因是防止缺省一部分参数。构造函数在数据库读入时不能少参数；在程序内构造时参数的个数也是固定的。如果使用默认参数可能会产生实例声明构造时参数的混乱。
- 本项目也应用了很多 Qt 的知识点。用了很多信号槽机制，同时利用各种控件进行设计和传入数据。
- 利用了少量数据库操作，实现了断电数据的存储。
- 用户友好、界面美观。我的程序没有采用整张的背景图片，因为我挑选了很多背景图片都没有纯色的效果好。参考携程网的设计，纯色的背景更有利于让用户的注意力集中在数据上。

4.4 心得体会

这次的大作业让我收获颇丰。第一次经历从控制台应用程序到真正好看的界面的设计过程十分有趣，在无尽的编程和调试过程中，也遇到了各种各样的问题。学会了自己解决问题和上网搜集问题解决方案等的解决方法。在这里感谢舍友一直以来在宿舍的陪伴。在凌晨四点的紫荆宿舍，大家的敲打键盘声此起彼伏，仿佛达成了一种完美的默契。

5 相关问题的说明

附出已有账号信息：
顾客：
Account : kevin
Password :1234567890
酒店管理员：
Account : 程晔安
Password : 1234567890
Account:于济川
Password:1234567890
平台管理员：唯一账号，不可注册
Acccount:kkk
Password:1234567890