大连理工大学软件学院 Java 课程设计

奥迪 4S 店汽车管理系统

School of Software, Dalian University of Technology

程新皓　软 1601　201693038　成绩

刘卉　软 1601　201692175　成绩

游传捷　软 1611　201692442　成绩

王之义　软 1611　201693138　成绩

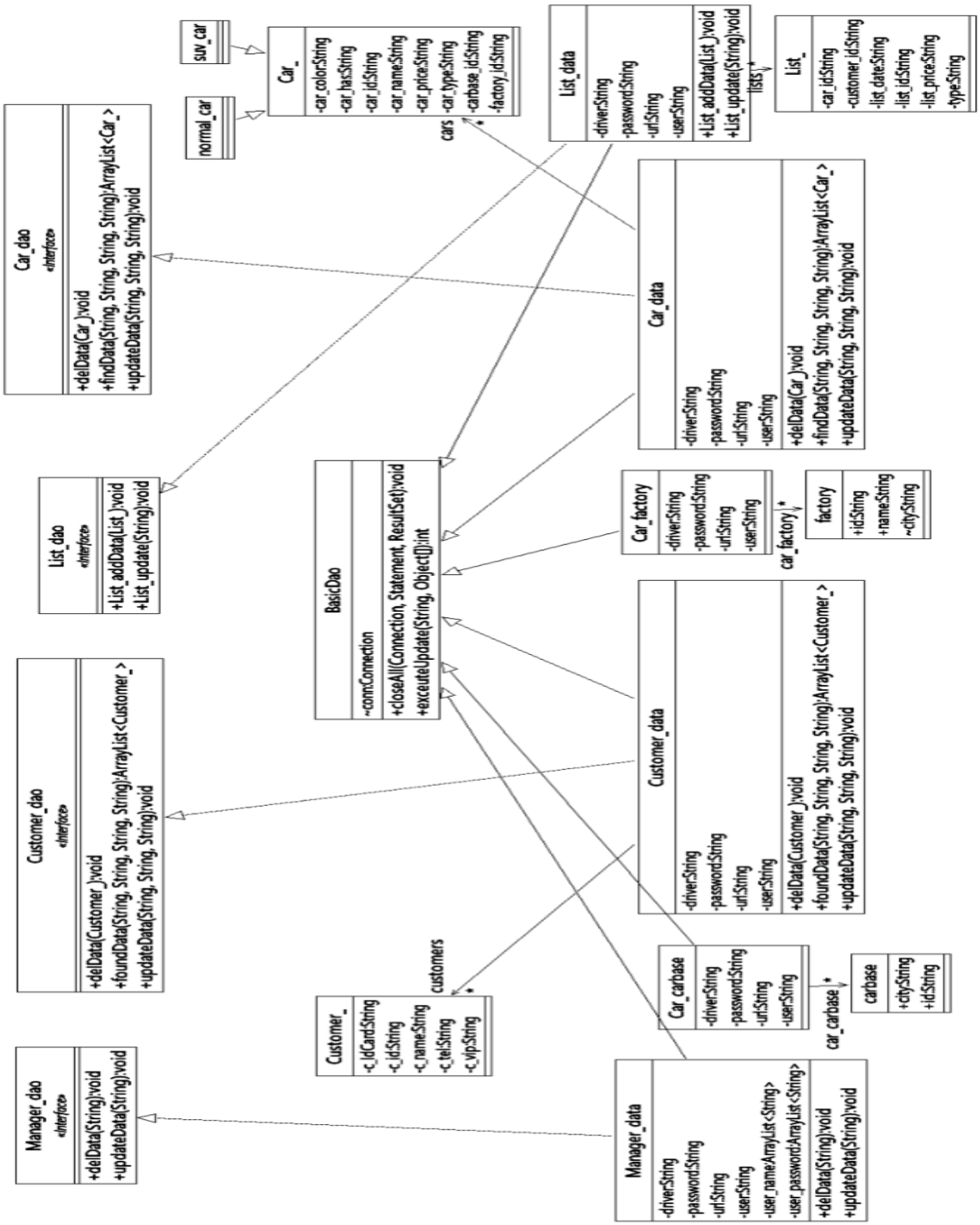
2018-7-19

目录

4S 店汽车管理系统

一、系统设计

1、系统主要类图:



我们将系统**分为三层**，针对实际应用情况设计数据库系统--4s_shop 系统，划分出 Car、CarBase、CarFactory、 Customer、List、Manager 关系模式，对应于相应的 Java 类，类的数据访问模式设定为私有来保证安全性，对应于数据库的属性集，对象则对应于数据库系统的实例。利用 ArrayList 结构加在 Database 与应用程序时间保证数据的**安全性与处理效率**。

在 Car、CarBase、CarFactory、Customer 添加 ID 保证了对象的唯一性，建立数据库时需要考虑参照完整性保证数据库的合理性。

我们按照不同的汽车类型，定义了 car_类的**继承类** normal_car，suv_car。对于每一个的对象，我们定义 set 和 get 方法来保证可用性。数据集对应于 UML 的最底层，对于数据库的访问我们封装了 **Dao 接口**方法，利用类的功能函数调用 Dao 接口实现对数据库的操作，这样既提高了程序运行的效率，又减少了重复书写代码可能出现的 bug。Dao 接口处于 UML 图的第二层，功能函数处于第一层。

对于用户友好性，我们着重考虑了 UI 的制作，根据不同用户相应，对类的功能函数予以**封装**，根据 manager 管理的不同对象,对功能函数实现了**多态**。对于系统安全性，我们采用了缓冲的方式，首先从数据库读出信息，置于 ArrayList 中，接下来基于 ArrayList 根据用户输入的信息与 ArrayList 中的信息进行比较，避免了 SQL 注入攻击。

2、系统数据库设计

2.1 概述

表名	功能说明
car(车辆基本表)	存储车辆基本信息
customer(客户信息表)	存储客户基本信息
factory(厂商基本信息表)	存储厂商基本信息
carbase(仓库表)	存储仓库基本信息
manager(管理员表)	存储管理员用户名和密码
list(订单信息表)	存储订单信息

2.2 车辆基本表

编号	名称	描述	数据类型	大小	备注
1	car_id	车辆编号	char	32	主键
2	car_price	车辆价格	demical		
3	car_name	车辆名称	char	100	
4	car_color	车辆颜色	char	100	
5	car_type	车辆类型	char	100	
6	factory_id	工厂编号	char	16	外键
7	carbase_id	车库编号	cahr	16	外键

2.3 客户信息表;

编号	名称	描述	数据类型	大小	备注
1	customer_id	客户编号	char	32	主键
2	customer_name	客户名字	char	100	
3	customer_vip	vip 状态	tinyint		1(0)
4	customer_birth	客户生日	char	100	
5	customer_tel	客户电话	char	11	

2.4 厂商基本信息表;

编号	名称	描述	数据类型	大小	备注
1	factory_id	工厂编号	char	16	主键
2	factory_name	工厂名称	char	100	
3	factory_city	工厂所在城市	char	100	

2.5 仓库表:

编号	名称	描述	数据类型	大小	备注
1	carbase_id	仓库编号	char	16	主键
2	carbase_city	仓库所在城市	char	100	

2.6 管理员表

编号	名称	描述	数据类型	大小	备注
1	manager_id	管理员编号	char	10	主键
2	password	管理员密码	char	10	

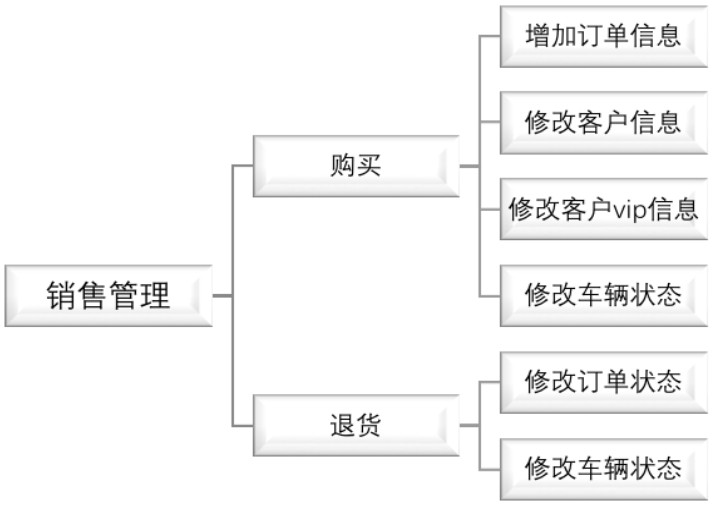
2.7 订单信息表

编号	名称	描述	数据类型	大小	备注
1	list_id	订单编号	char	32	主键
2	car_id	车辆编号	char	32	外键
3	customer_id	客户编号	char	32	外键
4	list_date	订单日期	char	100	
5	list_price	订单价格	demical		
6	list_type	订单类型	int		

二、主要功能设计

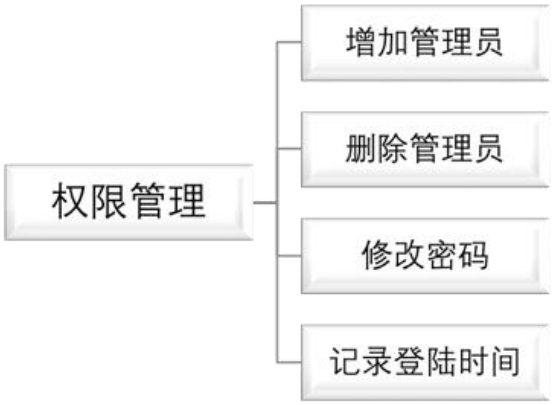
1、概述

该管理系统主要实现五个大功能，销售管理，权限管理，车辆管理，客户管理，订单管理



2、销售管理

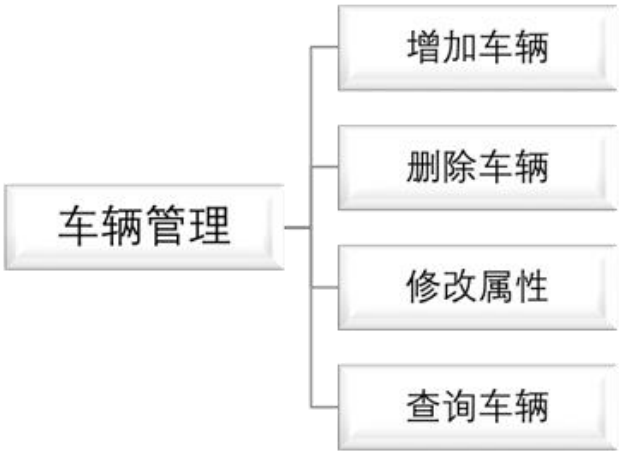
销售管理为客户提供购买和退货功能，购买时，需完成判断客户信息是否存在，不存在则增加客户信息。增加订单信息，将客户信息与车辆信息结合，生成订单信息。与此同时，查看客户购买总值，修改其 VIP 状态，修改所购买的车辆状态。



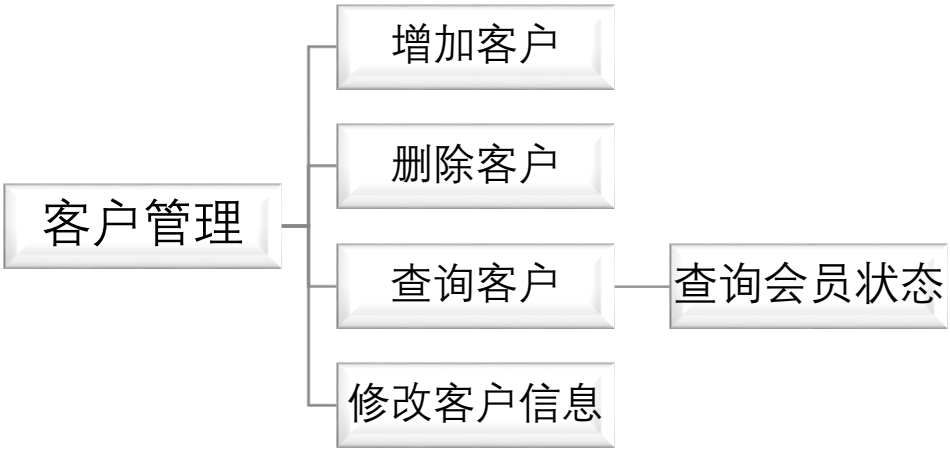
3、权限管理

权限管理为管理员提供了增加管理员，删除管理员，和对当前登陆管理员进行密码修改功能。管理员的 id 唯一，添加时不可重复。在管理员登陆的同时，记录管理员登陆信息。当出现问题时，可在后台查看管理员登陆信息。

4、车辆管理



车辆管理为管理员提供了增加车辆，删除车辆，修改车辆属性，查询某车辆功能，增加车辆时，id 随机生成，保证 id 不重复。查询车辆可根据两个属性值进行匹配。



5、客户管理

客户管理为管理员提供增加客户，删除客户，查询客户，修改客户信息功能，实现对客户的管理。增加客户自动生成客户随机 id，避免主键重复。

查询客户时，我们可以通过不同的选择标准，按照实际情况查询客户，考虑到客户 ID 为一串随机数码，用户友好性较差，我们提供了按名查找功能。可查看客户的会员状态。

6、订单管理

基于订单数据可，我们通过记录购买时间、购买金额、消费者、车辆 ID 来提供服务。我们为管理员提供查看所有订单状态，查询客户消费总额服务。当消费总额达到一定数目时，便于店家为消费者提供不同标准的服务例如 VIP 服务，多次买车优惠等活动。在车辆出现问题，需要保修或者返厂服务时，我们可以依据此订单管理功能提供销售凭据。

三、主要代码设计

1、分层:

将整个工程分为五个包及五个层次，分别是:

cn.jbit.shop.test

包含登陆界面及主界面，调用下层功能界面类;

cn.jbit.shop.service.impl

包含实现功能的界面及方法，实现相应接口;

cn.jbit.shop.service

定义一系列功能接口，例如购买，判断 id 有效应，查询等功能;

cn.jbit.shop.entity

实体类，临时存储数据，为上层实现下层修改提供数据，具有传送数据的功能;

cn.jbit.shop.dao.impl

DAO 接口的真实实现类，完成具体的数据库操作，但是不负责数据库的打开和关闭；

cn.jbit.shop.dao

定义一系列数据库的原子性操作，例如：增加、修改、删除、按 ID 查询等；

这种分层实现了低耦合性，不会出现牵一发而动全身的问题，且便于分工和维护，提供了组件的重用性。同时对于成员共同开发提供了便利，减少了错误出现的概率，大大提升了系统的安全性。

2、泛型：

在存储数据时，多次用到 **ArrayList<object>**来存储实体类对象。

ArrayList<carbase> car_carbase; 存储仓库实体类对象；

ArrayList<Car_> cars ;存储汽车实体类对象；

ArrayList<factory> car_factory;存储工厂实体类对象；

ArrayList<Customer_> customers;存储顾客实体类对象；

ArrayList<List_> lists ;存储订单实体类对象；

在创建 table 的 model 中，用 **Vector<object>**来存储表的属性，用 **Vector<Vector<Object>>**来存储表的元组；

Vector<String> columnNames

Vector<Vector<String>> data

泛型设计可以更方便的存储类对象，更重要的是消除源代码中的许多强制类型转换。这使得代码更加可读，并且减少了出错机会，为我们的编程提供了方便。

3、接口：

每个管理系统的功能均定义了相应的接口，以车辆管理系统为例：

public interface Car_ifind 定义了车辆管理的查找车辆功能；

public interface Car_update 定义了车辆管理的修改车辆功能；

public interface Car_add 定义了车辆管理的增加车辆功能；

public interface Car_del 定义了车辆管理的删除车辆功能；

public interface Car_has_id 定义了判断车辆 id 存在的功能；

public interface Car_has_cid 定义了判断仓库 id 存在的功能；

public interface Car_has_fid 定义了判断工厂 id 存在的功能；

接口使得系统可以支持开闭原则和依赖倒转原则，设计师可以设计出接口，而我们可以依照接口去写实现。可以更好的分工。

4、继承：

部分接口继承了一个功能的接口，实现的多功能

public interface Car_has_impl extends Car_has_cid , Car_has_id,Car_has_fid

该接口继承了两个判断外部表 id 的接口

```
public interface Car_able extends Car_add,Car_del,Car_update,Car_ifind
```

该接口继承了车辆管理的增、删、改、查功能

对功能的的集合，使功能的层次更加分明提高代码的重用性。

5、内部类&继承&线程：

存在Mainface里的内部类，继承Thread类，新增一个时间线程，1秒刷新一次；

```
class Time extends Thread {// 创建内部类
        public void run() {// 重构父类的方法
            while (true) {
                //方法实现
                try {

                    Thread.sleep(1000);// 令线程休眠1秒
                }
                catch (InterruptedException e) {
                    e.printStackTrace();
                }
```

```
            }
```

线程

每一个界面都是以线程的形式存在，且 MainFace 与 Time 类实现多线程

```
EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    MainFace frame = new MainFace();
                    frame.setVisible(true);

                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
```

可以提高资源利用率，使程序设计更加简单，程序响应更快

5、匿名内部类&接口

每一个按钮都实现了一个匿名内部类，创建了一个监听器，实现监听器接口

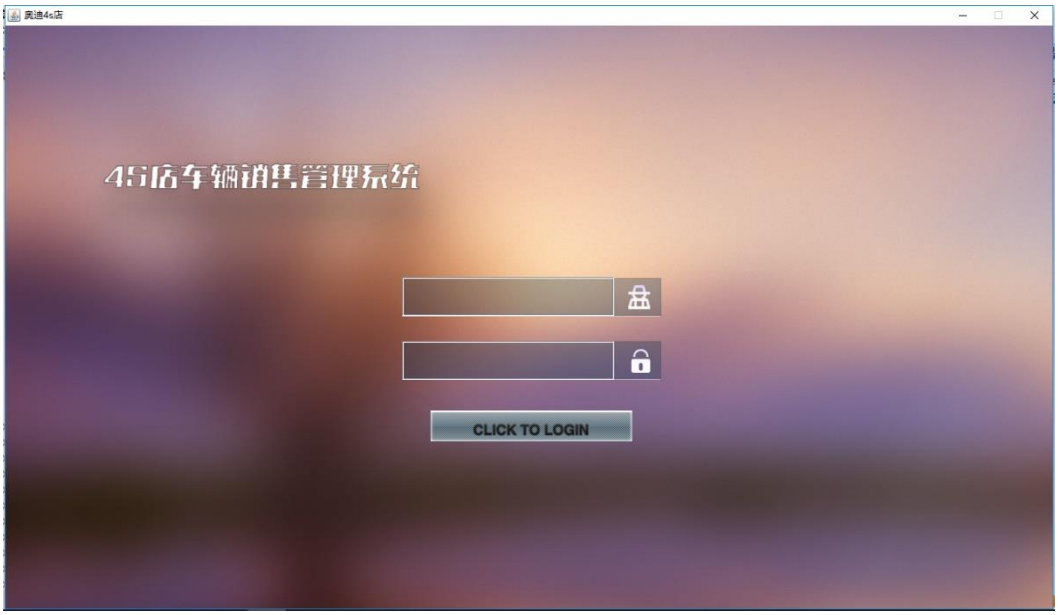
```
        btnNewButton_5.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                System.exit(0);
            }
        });
```

匿名内部类使代码更加简洁紧凑，可以减少代码量，直接在 button 上增加监听器实现简单的功能，而不用创建一个类。

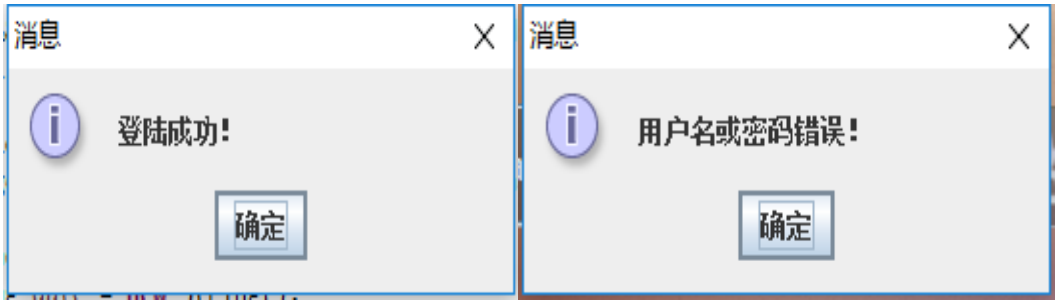
四、实现界面

1、登陆界面

实现管理员的登录，输入用户名和密码，点击登录按钮，系统判断该用户名和密码是否存在库中，若存在，则提示登陆成功，切换至主菜单界面。若不存在，则提示用户名或密码错误，需重新输入。

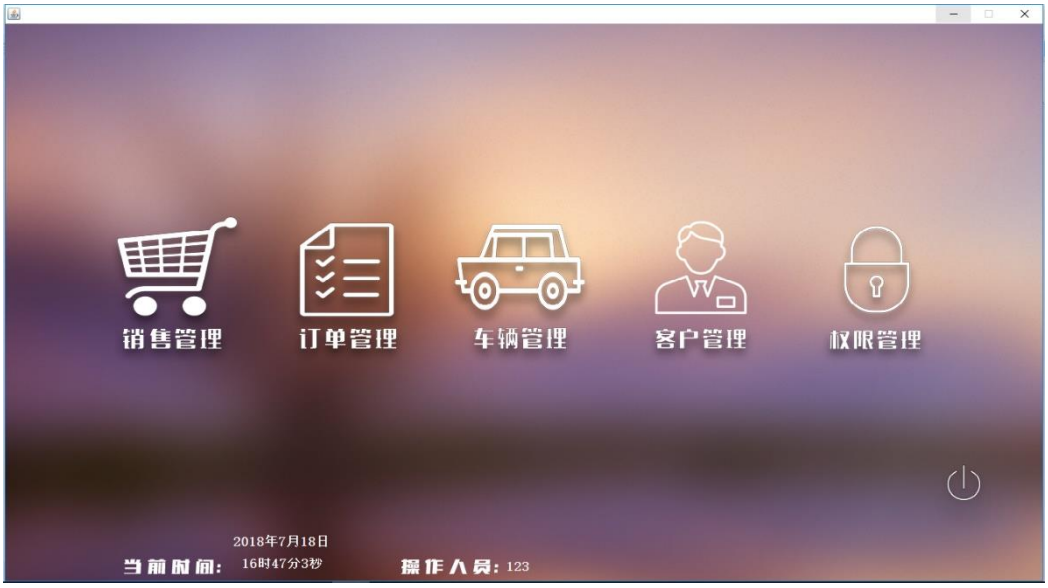


提示 1 登陆成功　　提示 2 用户名或密码失败



2、主菜单界面

实现通向销售管理、订单管理、车辆管理、客户管理、权限管理界面的按钮，可以出图系统，同时也显示当前时间，和当前操作人员。

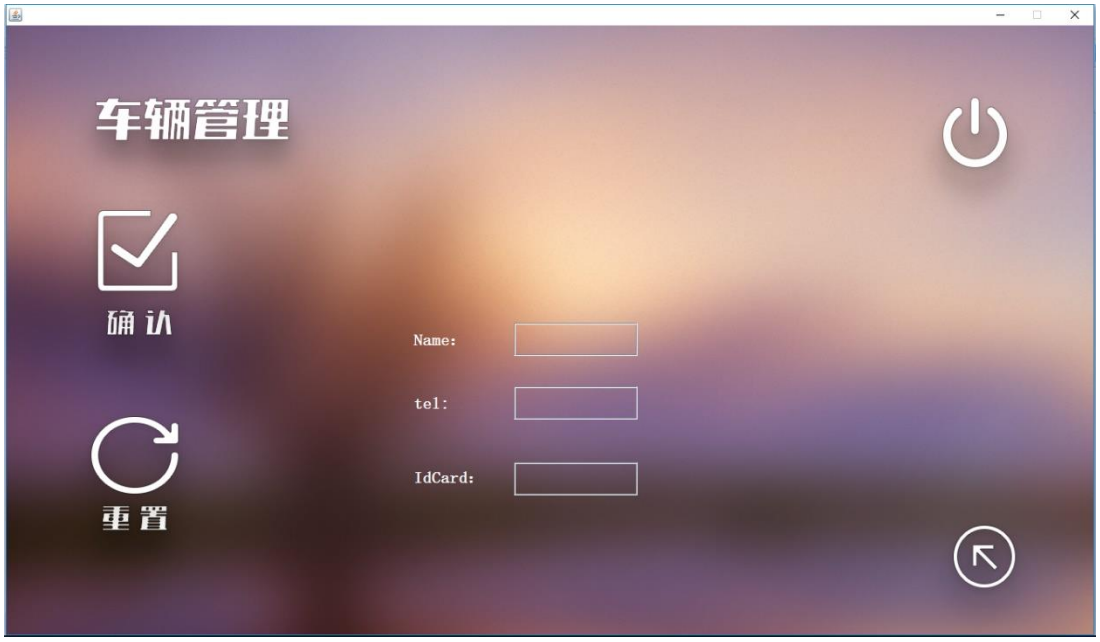


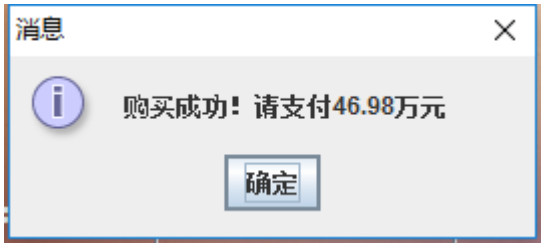
3、销售管理界面

实现购买和退货功能，购买则在购买框中输入想要购买的车辆 ID，若输入的购买车辆 ID 正确，则需要填写客户信息完成购买。需要退货则在退货框中输入购买时生成的退货 id 即可退货。若填写的信息有误，则弹出对话框提示。同时可退出系统，或返回主菜单。

4、客户信息填写界面

当填写正确车辆 ID 后，须填写客户姓名，电话号，IDCard，将自动生成客户 ID，填写成功后提示购买成功和需支付价格，vip 打九五折。点击重置按钮可实现重置输入内容，同时可退出或返回上一层。





5、订单管理界面

可查看所有订单，同时可查看订单的状态（是否退货），当在输入框输入姓名时，可实现查看该姓名客户消费总和。同时也可退出或返回上一层。

6、车辆管理界面

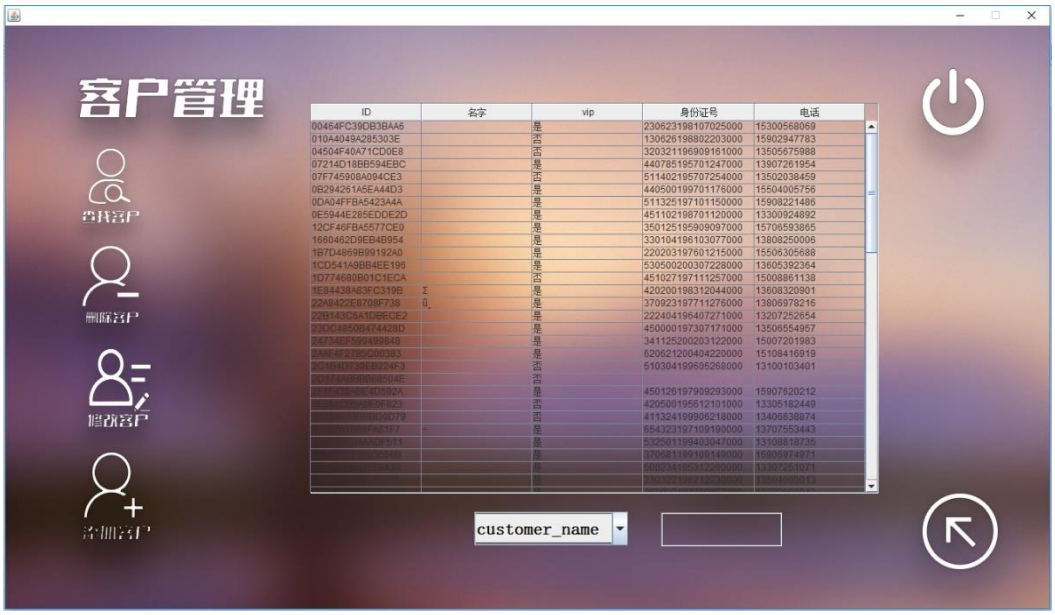
可查看所有车辆，卖出车辆则不显示。选中表中一行，下拉栏可选择属性，输入框内输入属性值，点击修改车辆可进行对该车属性进行修改。最下方输入框内输入属性值，点击添加车辆，可实现对车辆的增加。选中表中一行，点击删除车辆，可实现该车删除。点击查找按钮，进入车辆查找界面。同时实现了退出和返回上一层功能。

7、车辆查找界面

选择两个不重复的属性，并输入正确值（厂商 id，仓库 id 符合外键要求），点击确定即可查询符合两种属性值得车辆，重置按钮可重置输入内容，可实现返回上一层功能。

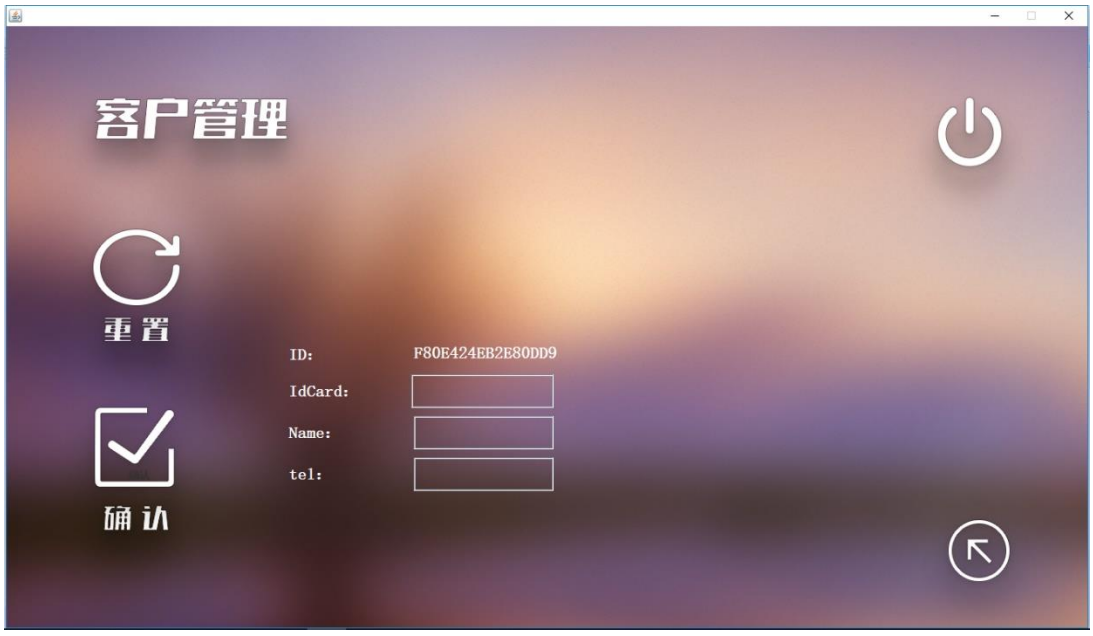
8、客户管理

显示所有客户信息，点击增加客户进入增加客户界面，选中表中一行，可实现对该客户的删除。选中一行且选择属性，输入属性值可实现对该客户的修改功能，点击查找客户进入查找客户界面。同时可退出和返回上一层。



9、客户增加界面

输入客户 IDCard，姓名，电话，可实现对客户的增加，与上面不同的是，此界面随机生成 ID，而购买车辆中客户信息添加则考虑是否存在客户信息而选择生成 ID。重置按钮则重置输入值。同时可退出或返回上一层。



10、客户查找界面

选择两个不重复的属性，并输入值，点击确定即可查询符合两种属性值得车辆，重置按钮可重置输入内容，可实现返回上一层功能。

11、权限管理

可查看当前所有管理员，点击添加管理员，删除管理员，修改密码可进入相应界面，进行对管理员的增加、删除、修改密码操作。增加时 id 不能重复，删除时，id 一定要存在当前库中，修改时原密码要跟新密码相同，否则将会出现提示信息。

五、课程体会

1、 刘卉

第一，技术方面。这次的个人项目老师要求的需求初看完全不明白怎么回事以往也没有接触过 java，最初是十分迷茫的，后来慢慢参考别人的想法，在网上去查相关资料，同时结合老师上课的内容，慢慢地了解了如何去完成项目的每个模块。

第二，学习态度法方面。起初刚拿到项目时其实毫无头绪，因为很多东西都没学过，而老师直接提出了这些需求，但是某种程度上说压力也是动力，最终在这样一次任务中我还是学到了一些新的东西，并且更懂得借用网络资源，不了解的东西就主动查阅学习，我觉得这样的学习态度也是很重要的收获。

2、游传捷

通过这个项目，我大致了解了后台是什么，后台到底是怎么和数据库和前端交互的。了解了一个项目从开始到做成原型的整个过程。这是我以前一直想要知道却一直无从得知的东西，说明真正的动手实践真的能得到比纯看书本找资料学到的东西更多，因为你不仅在做，也在不断查阅自己不懂的东西。第二，一定要从真实的项目中去学知识，这样才能学的快，有了需求就会去不断的学习。对于不懂得东西查阅各种资料，可以套用别人的，可以去研究别人是怎么做的，学以致用。各种网站，知乎，各种源码网站，设计网站，网上学习视频与资料、百度云盘都要好好利用，网络有无穷的资源，就看你会不会找。真正去学，把理想变为实践。

3、王之义

这个项目，老师给我们的要求是用 java 来实现。我主要负责 UI 和数据库连接部分，当时对 *java* 的知识是非常贫乏的，只有简单的语法基础，不知道从哪里开始，于是就研究老师给的一个非常简单的演示项目，以及老师的课程小作业，我从这些小的项目中循序渐进，获益匪浅。有困惑再有解决，是掌握技术的十分有效的而且巩固的方法，而且很有创造性，是属于归纳的思考方式。所以碰到问题，首先需要想想按常理该怎么去解决，再去寻找别人怎么解决的，这样自己提高会十分迅速。

4、程新皓

在做项目的过程中，我有点拖泥带水，导致的后果是，有那么2－3次我是通宵。也就是说，到了事情的最后，我才去匆匆忙忙“赶”项目。导致的结果是，我的

项目作的不够完善，也作的不好，并且自己也提心吊胆的，害怕项目出错。所以以后总结经验，早早把事情作完。

最后，我谈谈用户体验问题。能实现用户的需求是最基本的功能，最主要的是要让用户用的舒服和满意。我奉行的原则是，简单就是美。有句广告词叫做简约而不简单，我想就是这个道理，用最少的操作步骤达到操作目的。这就是我所要做的。

在整个项目的进行过程中，前期需求阶段，该阶段尽量多花时间，同时进行预研工作。因为有的需求可能要做实验才知道。在此阶段，对于需求，尽量保持各方理解都一样，在需求文档中，尽可能详细做备注。因为确定的东西越多越利于开发。第二，验收指标。建议写清验收方案，验收对比，验收指标是多少。如果自己不清楚，第一件事问人。在前期一定要明确好。第三，设计文档。先进行设计，再编码如果没有一个指导性的东西，自己想到哪就写哪，这样是耗时耗人的。在动手之前，要先想一遍涉及的模块。如果后期修改了机制，还要及时更新文档。如果个别人不写，要么强迫写，要么宁可延长项目时间。——匆忙的东西，总要延期的。