青岛理工大学

毕 业 设 计（论 文）

**题目：**基于SSM的企业公务用车管理系统的设计与实现

**院 别：** 信息与控制工程学院

**专业班级：** 软件工程164班

**学生姓名：** 周歆怡

**学 号：** 201607340

**指导教师：** 胡克勇

2020 年 2 月 24 日

摘 要

随着社会的进步、计算机技术的飞速发展，用计算机实现企业公务用车的管理势在必行。对于大中型企业来说，利用计算机支持企业高效率完成车辆管理的日常事务，是适应现代企业制度要求、推动企业劳动人事管理走向科学化、规范化的必要条件。而企业用车管理是一项琐碎、复杂而又十分细致的工作，每月每天需要用车的量很大，但买车加上后期养护车非常耗费金钱，如果派人租车，每月须进行各种用车记录，这就会耗费工作人员大量的时间和精力。

通过企业公务用车管理系统进行用车管理，不仅能够保证用车准确无误、快速输出，而且还可以利用数据库对有关车、人订单等各种信息进行统计，服务于用车公司其他方面的核算价格和订单处理等，同时它有着人工管理所无法比拟的优点，例如:检索迅速、查找方便、可靠性高、存储量大、保密性好、寿命长、成本低等。这些优点能够极大地提高企业用车时的效率,也是企业的科学化、正规化管理,与世界接轨的重要条件。所以基于SSM框架的企业公务用车管理系统应运而生，本文将详细地介绍设计此系统的详细思路和制作流程，对系统的主要功能和部分重要代码的设计也有较为详尽的说明。

关键词：企业公务用车，信息管理，计算机，SSM

ABSTRACT

With the progress of society and the rapid development of computer technology, it is imperative to realize the management of enterprise official car by computer. For large and medium-sized enterprises, the use of computers to support enterprises to efficiently complete the daily business of vehicle management is a necessary condition to meet the requirements of modern enterprise system and promote the enterprise labor and personnel management towards scientific and standardized. The enterprise car management is a trivial, complex and very detailed work, a large amount of car every day every month, but the car and car maintenance costs money, if sent to rent a car, all kinds of car records every month, it will consume a lot of staff time and energy.

Through the enterprise official vehicle management system, the vehicle management can not only ensure the accuracy and fast output of vehicle use, but also use the database to make statistics on various information such as vehicle and human orders, and serve other aspects of the accounting price and order processing of vehicle companies. At the same time, it has the advantages that cannot be compared with manual management, such as: fast retrieval, search party Convenient, high reliability, large storage, good confidentiality, long life, low cost. These advantages can greatly improve the efficiency of vehicle use, and are also the important conditions for the scientific and formal management of enterprises and their integration with the world. Therefore, the enterprise official vehicle management system based on the SSM framework came into being. This paper will introduce the detailed ideas and production process of the design of this system in detail, as well as the design of the main functions and some important codes of the system.

KEY WORDS: official vehicle，information management, computer，SSM

目录

[第1章 绪论 1](#_Toc6523)

[1.1研究背景及意义 1](#_Toc13105)

[1.2国内外研究现状 1](#_Toc12333)

[1.3存在的问题](#_Toc7825)

[1.4课题研究内容和目标 2](#_Toc7825)

[1.5 项目概述 2](#_Toc299636006)

[1.5.1产品介绍 2](#_Toc299636007)

[1.5.2 产品范围 2](#_Toc299636008)

[1.6论文组织结构](#_Toc6219)

1.6.1 [文档目的 1](#_Toc299636001)

1.6.2 [文档范围 1](#_Toc299636002)

1.6.3 [读者对象 1](#_Toc299636003)

[第2章 课题关键技术介绍 5](#_Toc390931801)

[2.1 Java语言 5](#_Toc390931802)

[2.2 SSM框架 5](#_Toc390931803)

[2.3 前端技术 8](#_Toc390931804)

[2.4 其他技术9](#_Toc390931805)

[2.5 开发工具 3](#_Toc299636010)

[2.6 本章小结 9](#_Toc390931806)

[第3章 系统需求分析 10](#_Toc390931807)

[3.1 目标设计要求 10](#_Toc390931808)

[3.2 用户组织管理及系统用例图](#_Toc390931810)

3.3 系统业务（数据）流程图 12

[3.4 功能性需求分类 4](#_Toc299636014)

3.4.1 [功能类别 A 4](#_Toc299636015)

3[.4.2 功能 X 4](#_Toc299636016)

3[.4.3 用例 Y 5](#_Toc299636017)

[3.4.4 功能类别 B 6](#_Toc299636018)

[3.4 可行性研究分析 12](#_Toc390931811)

[3.4.1 技术可行性 12](#_Toc390931812)

[3.4.2 经济可行性 13](#_Toc390931813)

[3.4.3 操作可行性 13](#_Toc390931814)

[3.5 系统功能需求分析 （不同用户操作权限表）](#_Toc390931809)

3.6 系统性能需求分析

[3.7 本章小结 13](#_Toc390931815)

[第4章 系统总体设计 14](#_Toc390931816)

[4.1 系统总体架构 （功能模块图）14](#_Toc390931817)

[4.2 系统主要功能模块设计 15](#_Toc390931818)

[4.3 系统功能流程 17](#_Toc390931819)

[4.4 系统数据库设计](#_Toc390931820)

[4.4.1](#_Toc390931820) 数据库逻辑设计（（E-R图，属性图）

[4.4.2](#_Toc390931820) 数据库设计实现（数据库名、表单（功能内容、表间的关系图）

[4.5 本章小结 21](#_Toc390931821)

[第5章 系统详细设计 22](#_Toc390931822)

[5.1 系统开发条件 22](#_Toc390931823)

[5.2 系统主要功能模块设计（分模块） 22](#_Toc390931824)

[5.2.1 登陆、注册功能 22](#_Toc390931825)（算法）

[5.2.2 首页功能 22](#_Toc390931826)

[5.2.7 管理员功能 23](#_Toc390931831)

[5.3 系统类图及交互图 26](#_Toc390931833)

[5.5 本章小结 30](#_Toc390931834)

[第6章 系统实现 31](#_Toc390931835)

[6.1 系统实现目标（环境、特点）](#_Toc390931836)

[6.2 系统实现结果（主要界面及功能，录入注意事项）](#_Toc390931837)

[6.3 系统主要代码](#_Toc390931838)

[6.4 帮助及出错处理 38](#_Toc390931839)

[6.9本章小结 57](#_Toc390931844)

[第7章 系统测试及维护 30](#_Toc21288)

6.1 测试目标及步骤

[6.2 单元测试 30](#_Toc14774)

[6.3 集成测试 32](#_Toc8286)

6.4 系统维护（系统部署，系统实施、系统维护定义、过程）

[6.5 本章小结 35](#_Toc11565)

1. 绪论

1.1研究背景及意义

1.1.1课题的研究背景

伴随着中国经济的高速发展,汽车逐渐成为现在主要的交通工具之一。由于官方车辆与人员的不断增加,企业公务车辆的管理问题也不断出现,因此对司机的管理、对车辆的管理、对订单的管理诸如此类的管理系统应运而生。公司一味购买车辆不仅成本较高，而且也不能满足实际的运输成本,无形之中增加了单位压力。公务用车管理系统的主要目的就是使车辆管理具有制度性的合理化和规模化,使得公务车辆的效益最优化,大幅度的节约成本。

1.1.2课题的研究意义

在此背景下,课题研究基于SSM架构的企业车辆管理信息系统,目的是设计和实现一款功能完备,界面简洁整齐易上手，以及具备一定的健壮性、易维护的企业公务用车管理系统,为的是较好地满足现在企业对于用车的各项业务需求。

1.2国内外研究现状

当下的公务用车管理系统已经发展到比较成熟的阶段，公务用车管理系统是一个专业性很强的小型数据库应用软件，国内外用车管理公司大部分已经拥有了适合自己的管理系统。采用计算机管理数据提高了数据的完整性和安全性，代替了过去工作人员复杂的手工操作。同时使用计算机管理软件数据显示清晰，功能强大，实现了过去人工管理所实现不了的功能。

国内外的信息化利用现代通信技术、网络技术、数据库技术为核心，不断发展深化的过程，将研究对象(用车公司、司机、管理员)等汇总于数据库，与各种行为相互结合，从而使得企业用车便利,为推动人类社会进步提供极大的技术支持。

1.3存在的问题

计算机管理系统开发的一个重要问题就是如何准确的管理用车数据，各个企事业单位的需求不同，关注的业务问题不同，采用的管理技术不同，也必然存在着不同的开发策略和方法。如何有效和迅速地响应公务车辆管理需求,如何解决目前的困难是现阶段所面临的主要问题。

1.4课题研究内容和目标

本论文研究的公务用车管理系统采用B/S模式,在实际工作过程中应用IDEA开发平台进行设计和开发。以页面登录权限验证和对企业用车途中各角色管理数据的设计为重点进行深入研究和探讨,满足实际应用对象的开发需求。整个系统使用了JSP动态网页技术、Javascript、CSS、Bootstrap等技术来实现美观、简洁、便于操作的用户界面。管理员管理数据使用Navicat for MySQL数据库作为数据存储和提取的平台,并对各个数据表的详细信息进行了说明。

1.5项目概述

1.5.1产品介绍

企业公务用车管理系统主要的用途是对企业的用车情况进行合理化管理，包括司机管理（姓名、车牌号、昵称、信用积分、年龄等）、车辆管理（名称、车牌号、颜色、品牌、类型、车型等）、订单管理、账号管理、投诉管理等模块。

该系统还具有一个对系统用户进行管理的，用户管理模块，包括用户添加，用户密码的修改，用户的删除。

1.5.2产品范围

本系统建成后将为企业提供电子化,信息化以及精细化的支持,不仅降低了公务车辆管理成本,也提升了政务管理效率,使管理员管理工作变得更加方便快捷。

1.5.3用户群体及角色

司机：为用车公司提供服务。

用车公司：租赁需要使用的车辆。

管理员：管理用户信息、车辆信息、订单信息等。

1.6论文组织结构

第一章为绪论部分，主要介绍企业公务用车的相关知识以及该领域在国内外的发展现状，最后对本文的整体组织架构进行说明。

第二章为课题关键技术介绍，主要介绍了制作本系统用到的语言、框架、技术及开发工具。

第三章为系统需求分析，通过结合系统需求情况，针对性地分析系统需求在非功能与功能两方面的体现，给出系统设计的整体框架，使用用例形式描述系统功能需求情况。

第四章为系统总体设计，主要介绍系统功能总体设计和系统框架，其中主要对系统总体、数据库和各种子模块功能设计进行说明，同时介绍了系统架构及其核心功能模块。

第五章为系统详细设计，主要分模块详细介绍了系统是如何设计的，对每个子功能进行详细说明，使用交互图和类图对系统进行详细阐述。

第六章为系统具体实现，对系统的具体功能实现进行介绍，并且介绍了系统的实现过程中 SSM 框架的架构和shiro框架的使用。详细介绍所有子模块的设计和具体的信息管理功能的实现过程，分析其不同模块间的功能实现流程，并介绍了代码设计思路。

第2章 课题关键技术介绍

2.1 Java语言

Java是一门面向对象编程语言，不仅吸收了C++语言的各种优点，还摒弃了C++里难以理解的多继承、指针等概念，因此Java语言具有功能强大和简单易用两个特征。Java语言作为静态面向对象编程语言的代表，极好地实现了面向对象理论，允许程序员以优雅的思维方式进行复杂的编程。Java具有简单性、分布式、健壮性、安全性、平台独立与可移植性、多线程、动态性等特点。

2.2 SSM框架介绍

本系统充分考虑用户的需求，服务端采用MVC软件架构模式，可将页面显示、业务逻辑和数据库访问彼此分离，是目前很成功的软件架构模式。为了提高应用程序的开发效率，减少系统的复杂度，因此系统采用SSM框架[1]（Spring MVC、Spring和MyBatis），SSM框架相对于传统的SSH（Struts、Spring和Hibernate）框架耦合度更低，更轻量级，效率更高。SSM框架是标准的MVC，将整个系统划分为表示层、控制层、服务层、数据库访问层4个大层，使用Spring MVC负责请求的转发和视图管理，Spring实现业务对象管理，MyBatis作为数据对象持久化引擎,对系统架构进行良好的设计、功能进行合理规划并实现。

2.2.1SSM框架

（1）Spring框架

Spring框架是Rod Johnson为解决EJB开发存在的一些问题所提出的一种轻量级框架。它是一个开源的企业应用开发框架，采用JavaBean来替换笨重繁杂、成本高、效率低的EJB，降低了企业级开发的复杂性，是一个轻量级的控制反转（IOC）和面向切面（AOP）的容器框架。Spring框架将之前复杂且难以维护的javaEE的开发变得简单而高效, 而且它可以与其他框架相结合来更好地实现每一层。

（2）Spring MVC框架

Spring MVC[2]属于SpringFrameWork的后续产品，已经融合在Spring Web Flow里。Spring MVC是Spring框架中用于WEB快速开发的一个模块，分离了控制器、模型对象、分派器以及处理程序对象的角色，这种分离让它们更容易进行定制。它是是基于DispatcherServlet (前端控制器) 来实现表示层与控制层的交互, 通过XML配置的方式将映射处理器、处理适配器、拦截器、视图解析器、文件上传解析器等注入到Spring容器中。为了简化开发还提供了注解方式, 只需要在java类上写上相应的注解名, SpringMVC就会对其进行扫描并自动注入。

（3） MyBatis-plus框架

与Hibernate一样，MyBatis是一个基于Java的数据持久层框架。有别于Hibernate的全自动化，MyBatis框架结合了Hibernate和JDBC的优点，使用较灵活，程序员自己写SQL语句，便于优化。

Mybatis-Plus（简称MP）是一个 Mybatis 的增强工具，在 Mybatis 的基础上只做增强不做改变，为简化开发、提高效率而生。它已经封装好了一些crud方法，我们不需要再写xml了，直接调用这些方法就行，就类似于JPA。

2.3前端技术介绍

1.2.1JSP动态网页技术

JSP (Java Server Page) 其实是一种运行在服务器端的java技术, 用来开发WEB动态网页。准确来讲, JSP是一种服务端的脚本语言, 因为系统表示层的视图使用的是jsp文件进行显示,还有许多其他的标签参考了文献[3]。

1.2.2 Bootstrap前端框架

Bootstrap是Twitter公司(www. twitter.com)开发的一个基于HTML , CSS , JavaScript的技术框架，符合HTML和 CSS规范，且代码简洁、视觉优美。该框架设计时尚、直观、强大，可用于快速、简单地构建网页或网站。 Bootstrap使用了最新的浏览器技术，为实现快速开发提供了一套前端工具包，包括布局、栅格、表格、按钮、表单、导航和提示等。使用Bootstrap不仅可以构建出非常优雅的前端界面，而且占用资源非常少.

Bootstrap 自带以下特性：全局的 CSS 设置、定义基本的 HTML 元素样式、可扩展的 class，以及一个先进的栅格系统。基本组件有：网站导航、标签页、工具条、分页栏、提示标签、产品展示、提示信息块和进度条等。JavaScript 插件：Bootstrap 包含了十几个自定义的 jQuery 插件，这些jQuery插件主要用来帮助开发者实现与用户交互的功能

2.4其他技术

2.4.1shiro框架

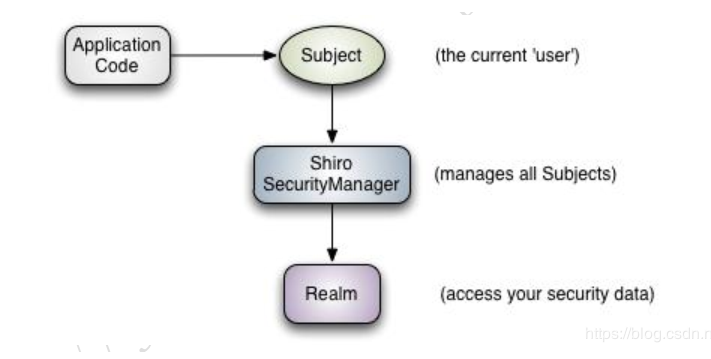


图3.3 shiro框架实现原理

Shiro[4]是一个开源的Java安全框架，如图3.3 shiro框架实现原理，它具有身份验证、授权、加密和会话管理等功能，它的三个核心组件Subject是指当前正在执行的用户；SecurityManager是Subject背后对应实际操作部分，管理着所有的安全操作，保证Subject和Shiro组件按照安全规则进行交互；Realms连接着Shiro和当前系统的安全数据，即当用户执行相应安全操作时，Shiro会从配置的一个或多个Realms中查找预存安全数据。

2.4.2Lombok插件

Lombok项目是一个Java库，它会自动插入编辑器和构建工具中，Lombok提供了一组有用的注释，用来消除Java类中的大量样板代码。仅五个字符(@Data)就可以替换数百行代码从而产生干净，简洁且易于维护的Java类。

2.5开发工具

2.5.1JDK

JDK，全称Java Development Kit,也叫做java开发工具包。它包括了Java编译器、JVM、不计其数的Java工具以及Java基础API里面的Java类库和Java的语言规范。有了JDK再加上一个编辑器(比如Eclipse等等) ，就可以使用java语言写程序。

2.5.2 IDEA

IDEA 全称IntelliJ IDEA，是java语言开发的集成环境，IntelliJ在业界被公认为最好的java开发工具之一，也是AES算法标准的主要竞争者，其安全性已经在国际密码年会上被证明。

2.5.3Tomcat

它是一个开源的web应用服务器，本项目中用于运行servlet和jsp页面。

2.5.4Mysql

MySQL[5]是一个关系型数据库管理系统，原本是一个开放源码的关系数据库管理系统，

在过去由于性能高、成本低、可靠性好，已经成为最流行的开源数据库，因此被广泛地应用在Internet上的中小型网站中。随着MySQL的不断成熟，它也逐渐用于更多大规模网站和应用。

2.6本章小结

本章主要介绍系统所使用的关键技术。首先对整个系统使用的Java语言进行介绍，然后对组成SSM框架的Spring 、SpringMVC 以及 MyBatis&MyBatis-plus进行详细的介绍，阐述这三种框架的基本组成以及各自的特点，之后介绍提供系统安全的 Shiro 技术和Lombok插件。最后简要说明了开发此系统用到的工具。

第3章 系统需求分析

本章内容分析了系统的需求情况并论述其架构组成，基于系统需求的视角为系统设计提供需求基础，同时介绍了不同模块功能的需求情况，奠定系统功能设计方面的理论根基。

2.1设计背景

随着社会上各个公司内部的发展持续性增强，企业的用车需求方面也逐渐增多，通过对各类车辆的监管，企业在用人管理车辆方面也有较大的需求，伴随着车辆的价格越来越可观，企业日常处理一些事务免不了要用车，管理车辆要用人，租用车辆也需要司机，借此管理人事、车辆的效率也在根据需求持续提高，管理方式和理念也逐渐得到发展，由此推动了国内用车管理系统的迅猛发展。

2.2目标设计要求

公务用车系统是一个利用网络通信技术、身份认证登录技术等，提供给有需求的人们的智能处理租赁用车数据的统一平台。相比于常规的信息管理系统，企业公务用车与之有着异曲同工之妙，但也具有其他截然不同的需求，需要在编写代码进程中始终结合企业自己的优缺点，完善在用车过程中所进行的一系列操作管理，并且要与日常信息管理系统区分开。企业用车系统应该以提高公司职工工作效能，实现信息的透明化，以及用户的系统化，不断推动企业效益增长，降低用车成本为目的。设计系统前查询了网络上的资料[6]，了解了了企业公务用车的实际需求，只有满足了公司用车的实际需求，才能更好地解决企业用车现存的问题。

此系统旨在为用户提供方便查看用车信息的平台，用户大致分为三类：用车公司、司机、管理员。

2.3用户组织管理及系统用例图

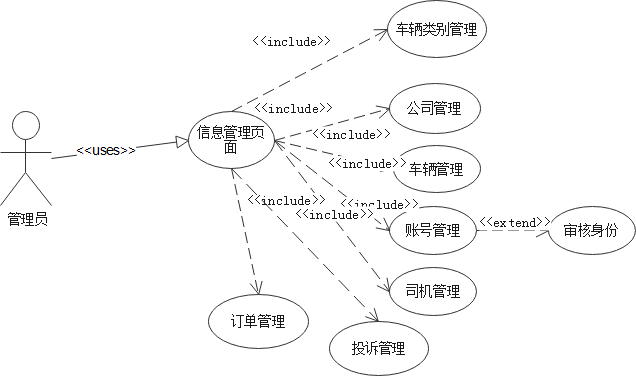


图3.1管理员用例图

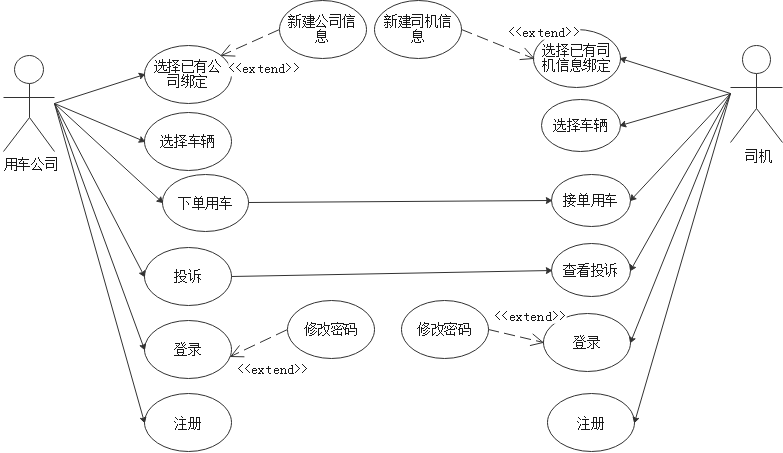


图3.2用车公司和司机用例图

如图所示，本系统的主要使用者有三个，分别为用车公司、司机、管理员，管理员负责管理所有用户的信息，包括订单信息投诉信息，个人信息等。用车过程主要是公司和司机的操作，用车结束后管理员根据用车情况处理后续的投诉。

3.3 系统整体数据流程图

企业公务用车系统的总体流程可描述为：用户登录系统进行相应权限的操作。具体为：当用户为用车公司或司机时，可进行个人信息提交，选择车辆等操作；当为管理员用户时，则可对后台数据进行相应的增、删、改等操作。如图3.3系统整体数据流程图。

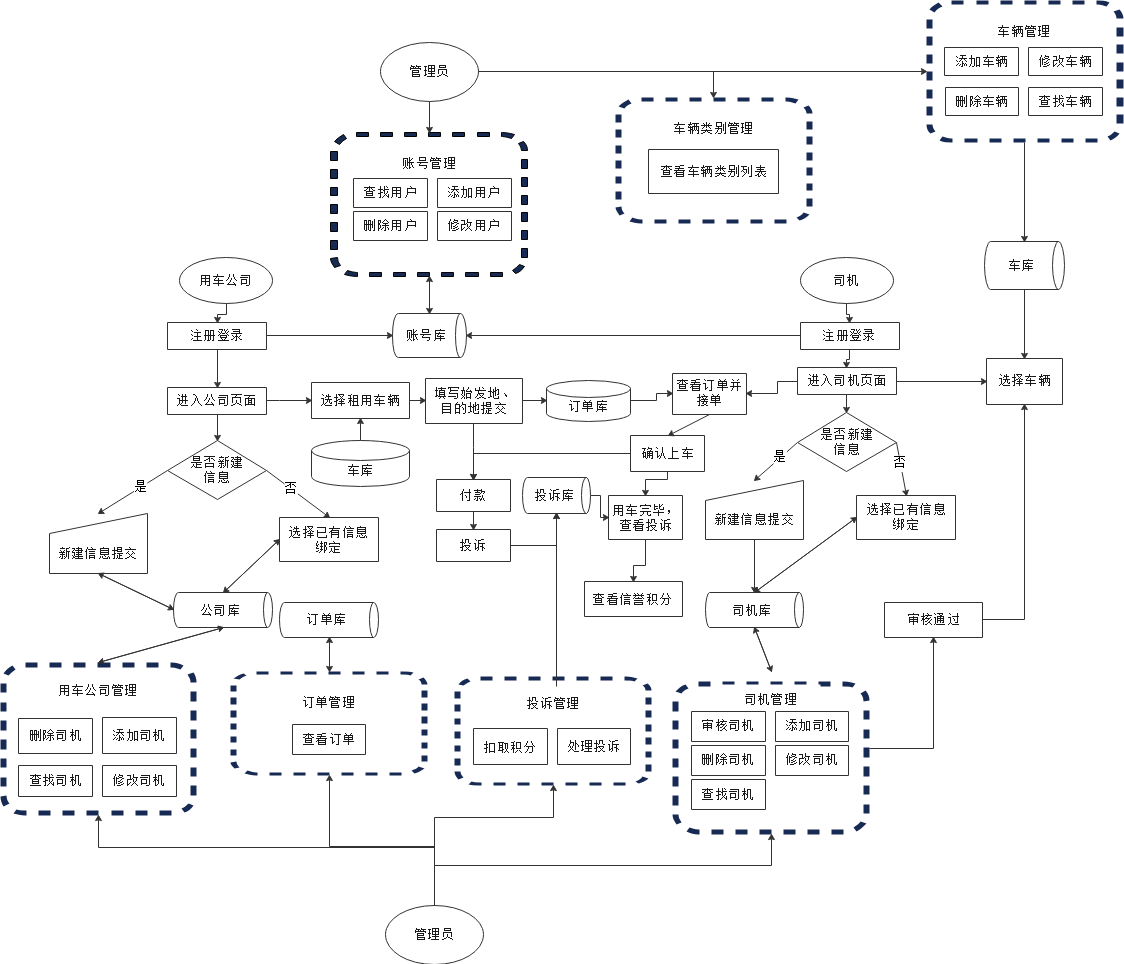


图3.3系统整体数据流程图

3.4系统功能性需求

3.4.1账号管理模块

账号管理模块为系统设计的基础部分，用户可以通过注册来选择自己的身份为司机或者是用车公司，然后登陆进入系统，shiro框架设计的部分参考的[9]。管理员可以登陆进入后台维护用户的信息，用户还可以不通过注册直接联系管理员添加新用户，管理员添加后用户可输入用户名和密码直接登录。当用户需要修改角色时也可以联系管理员修改，本系统规定用户自己不可随意更改角色。

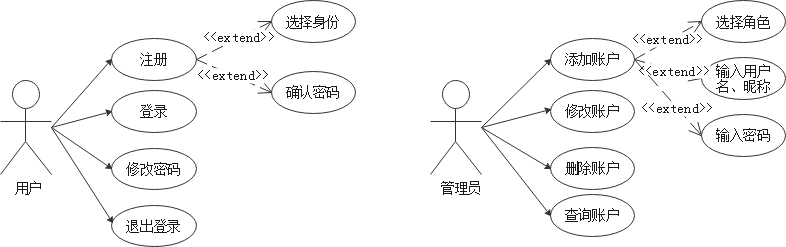


图3.4账号管理模块用例图

3.4.2审核司机模块

审核司机模块为下单模块做铺垫，如果司机无法审核通过就无法拥有车辆，用车公司也就无法选择车辆下单，只有司机提交的信息通过审核才可以正式进入司机库被选择。管理员可以增删改查司机的信息，无论是司机自己提交或者选择已有司机信息绑定都需要通过管理员审核才可以成为正式司机进行选车。

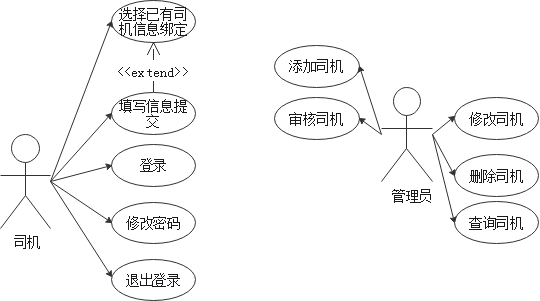


图3.5审核司机模块用例图

3.4.3下单模块

用车公司有意向用车时，下单，司机查看订单去指定地点接人，确认接到人后服务，服务完成后用车公司付款。管理员在用车全程中可查看订单，但无法操作。

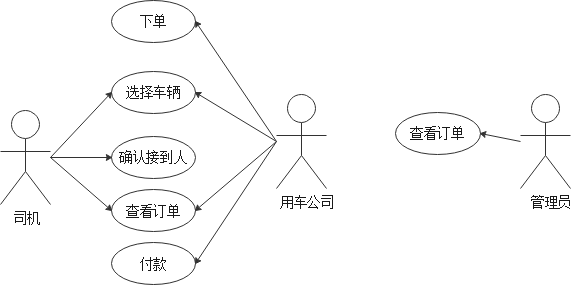


图3.6下单模块用例图

3.4.4投诉模块

用车公司用车完成后，可以根据实际情况投诉司机，投诉完成后可以查看投诉处理情况和投诉信息，管理员查看订单审核完毕后，按情况扣取司机的信誉积。司机可以查看本次订单的投诉情况和自己信誉积分扣除的情况。

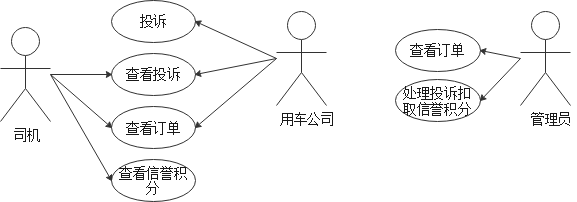


图3.7投诉模块用例图

3.5 系统非功能性需求

企业公务用车系统的非功能需求特指它在使用方面的便利性，以及它功能上的可扩展性，此后可以针对后续新功能进行拓展延伸，通过设置的接口拓展系统功能，有效地降低了研发所需要的额外的费用，同时可以增强管理体系的实用性，为现阶段先进系统的重要特征。易用性方面要求管理体系的使用过程简洁不复杂，且能实现强大的功能，人机交互界面友好，基于对功能的研发，以及对人机交互界面的设计。简化系统的操作，是现代化信息管理体系的重要特征之一。

（1）完整性特征

一个系统的重要构成部分是各种分门别类的信息，此管理系统需要对车辆和用车人的相关信息实现随时随刻都能查看的功能，由此可知在项目研发的初期，必须将信息完整性考虑在内，通过构建系统的自动弹窗以及提示预警功能，可以实现数据不完整性的报警，比如在使用了不合理的数据，系统会根据输入的数据规范法则自动报警，方便使用者及时纠正自己犯下的错误。

（2）实用性特征

企业公务用车管理系统的现代化水平和科学化水平能通过用户监测服务得以体现，能基于特定的用户提供不同的系统需求，通过计算机编程运行的高效性取代工作效率低下的传统管理模式，参考了文献[7]。该管理系统的实际操作者是提供租车平台的网络管理员，使用者是企业有用车意向的职工以及拥有车辆使用权的司机。使用人员为了使用方便起见，功能尽可能言简意赅，而管理人员为了管理方便，功能尽量全面，将使用人员的日常使用方式，以及操作人员的使用习惯纳入重点需要考虑的部分，这是系统编码与功能实现的过程中必须要严格实现的内容。

（3）功能的拓展性

企业公务用车管理系统管理数据的过程十分冗杂，参考了[8]，在系统编码研发阶段，需要充分了解当代企业不断发展会带来林林总总的需求，因此无论如何要将系统功能的可拓展性居于重要地位。为了契合用户的添加修改记录的功能，系统的拓展性应该可以轻松实现功能的添加和修改操作，而且在设计风格上能保证新增功能模块具有普遍性，同时根据人们的日常使用规范不断调整直到合适。

（4）先进性特征

企业公务用车管理系统的研发必须基于现阶段主流甚至前卫的数据库，只有这样才能维持各项功能的先进性，系统研发的主导地位充分体现在系统架构、软件的开发、设计思想、和后期数据测试维护等方面。

（5）安全性

鉴于系统中存放的不计其数的个人信息以及车牌号等无法被公开浏览的数据，系统的权限管理是必须纳入设计重点的部分，针对不同的用户数身份要设计不同的身份验证，区别对待不同的使用权限，不同层级的使用者只能被授予所述层级该有权限。

（6）便捷性

人机交互界面应该具备简便性和适用性，从而完成优良的人机交互设计，满足用户对操作简便和实用的功能的需求。

3.6系统用户权限需求

角色权限表

|  |  |
| --- | --- |
| 用户 | 操作权限 |
| 司机 | 注册、登录、选择车辆、完善个人信息、改查订单、查看投诉及后续处理 |
| 用车公司 | 注册、登录、选择车辆、增改查订单、完善个人信息、付款、投诉 |
| 管理员 | 登录、查看车辆分类、管理车辆信息、管理公司信息、管理司机信息（包含审核司机）、增删改查账号信息、查看订单、处理投诉 |

3.7本章小结

本章内容详细分析了企业公务用车系统的需求状况，同时深入探讨系统研发过程中可能出现的缺陷与不足，并分析各种系统功能详细需求，从而给出合理化的修改建议，分析的过程以及得出的结果会影响到系统的整体设计和实现。

第4章 系统总体设计

4.1系统总体架构

用车公司：通过选择车辆下单用车，完成订单后可以投诉司机。

司机：选择车辆完毕后，等待用车公司用车，完成订单后进行下一单，期间查看之前的投诉情况。

管理员：管理用户信息，车辆信息，订单信息等等，监控每一单用车的完成情况。



图4.1系统总体设计图

4.2 系统主要功能模块设计

4.2.1司机管理模块

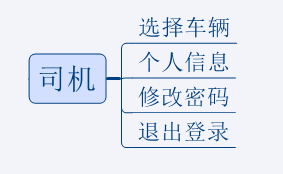


图4.2司机管理模块

司机管理模块主要包含选择车辆、个人信息、修改密码和退出登录，司机提交个人信息管理员审核通过后才可以选择车辆。

（1）选择车辆页面：司机通过注册后（注册时选择自己的角色是司机）选择车辆，若当前无车辆或车辆都已被选中，则超级管理员进入后台管理添加车辆，司机再次进入系统可选择车辆，每个司机只能选择一辆车，选择后不可更改。

（2）个人信息页面：

①注册登录：司机注册后登录进入个人信息界面，系统会根据登录的用户角色不同展示不同的界面，进入后首先需要完善自己的信息，或者从司机列表里选择自己的信息关联，提交后，后台管理员审核完毕才可以成为正式的司机进入企业可选司机列表。

②订单管理：若有企业选中司机拥有的该车并提交订单，司机可以进入系统查看订单并去指定地点接客，接到客人后点击接到客人，状态自动改为已接客。完成订单后，若用车公司投诉，可查看投诉。若管理员处理了投诉信息，可在此界面查看自己信誉积分。

③修改个人信息：司机可以修改自己的信息（除了自己的身份）以及修改密码。若想修改身份，联系管理员修改用户信息即可。

4.2.2用车公司管理模块

用车公司注册登录后可以修改密码，用车时选择服务，服务根据司机的状态分为三个模块，然后可以修改个人信息提交公司信息，提交后管理员才可以看到公司。

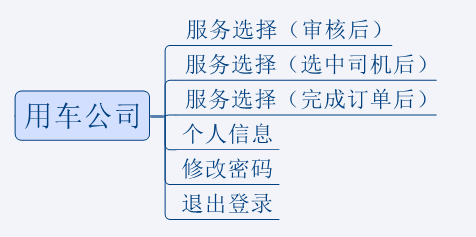


图4.3用车公司管理模块

4.2.3管理员管理模块

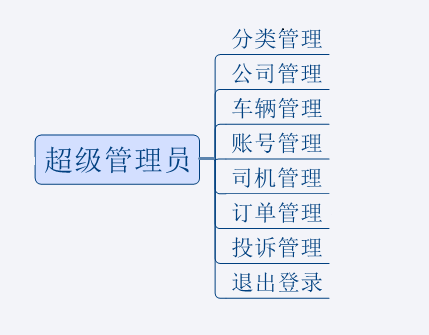


图4.4管理员管理模块

管理员负责统筹管理整个系统的所有信息，包括公司、司机的信息，还有车辆分类和车库信息，用车公司在用车时管理员还可以查看订单信息，用车结束管理员需要处理投诉信息发送给司机和用车公司。

（1）分类管理：此系统预设了5种车的类型，分别为入门级、消费级、普通级、豪华级、奢华级，分别有不同的价值和使用时的价格，可供用车公司比较选择。

（2）公司管理：若公司账号注册登录后未提交公司信息，在此模块无法看到。只有公司提交信息后，管理员方可在此看到此公司的信息，包括选车的状态等等。管理员还可以在此添加删除新公司以及修改用车公司的负责人和邮箱。

（3）车辆管理：此模块展示了所有车辆的各种信息，包括未选中的正在使用的等等，管理员可以增删改查车辆。若车辆过多，可以通过面板右上角的搜索功能来查找需要的车，可供查找的有品牌和颜色。

（4）账号管理：管理员可以增删改查所有用户的信息，每个用户可绑定一个司机或者一个用车公司（绑定后不可修改），也可赋予账号“管理员”的角色，只有admin超级管理员账号信息默认不可修改。

（5）司机管理：与公司管理相差无几，在此不做赘述。

（6）订单管理： 只能查看订单的信息，无法修改添加删除。

（7）投诉管理：查看用车公司投诉司机情况，根据描述管理员需扣取司机的信誉积分，处理完毕后，状态变为已处理。

4.3 系统功能流程

(1) 司机管理：司机进行注册，录入司机的个人信息，根据司机是否有车进行分类，无车司机需要选择系统里提供的车，有车司机则需联系管理员选择车辆信息绑定。提交后，发送至人工审核模块，成为可用状态的司机增加信誉积分栏目，管理员可根据派单模块和投诉模块的反馈对信誉积分进行增减操作。

(2) 人工审核：在审核员确定审核通过后，司机进入可用状态，生成司机编号添加到司机库（添加数据中含有审核员编号、审核时间和车辆层次）并发往司机管理模块。

(3) 用车公司：用车公司进行注册，注册成功后，进入管理界面进行设置其公司员工需要用车的时间、地点、车辆层次及员工的手机号，进行在线缴费并生成交易单号，发至派单模块，如司机有迟到、加价等违规行为则可进入投诉模块。

(4) 派单：针对车辆档次和司机信息发送派单信息，派单成功后，生成订单号发送至司机管理模块和用车公司模块，系统司机的手机号和车牌号码发送至用车人的手机上并将用车人的手机号发送至司机的手机上，司机前往指定地点进行接客，待用车人上车后，司机进行确认上车，订单成为进行状态，送至用车人指定地点后进行订单完成，订单此时成为完成状态，系统根据司机行程的公里数、路上使用的时间和信誉积分计算出司机应得的收益，并添加对应的信誉积分至司机管理模块。

(5) 投诉：根据生成的订单号查询司机，如有迟到、加价或车型不符等行为，则扣除司机的信誉积分，并发送警告通知给司机。

4.4系统数据库设计

4.4.1数据库设计实现

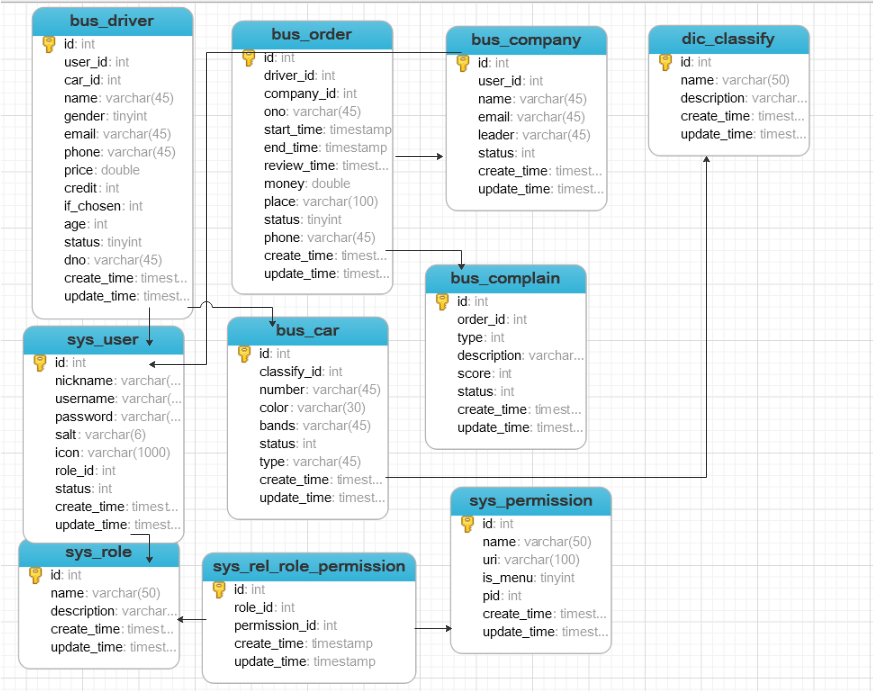


图4.5数据库整体设计

本系统针对每个角色模块功能的独特性, 为系统的每个角色的模块进行数据库的设计。如图3.2公务用车管理模块数据模型所示, 该模块中有10张表，彼此相互依存，为的是降低系统开发的复杂程度, 方便后期数据库的维护，实现了数据的持久化。它们分别存储企业在用车过程中涉及到的信息，下面会通过实体属性图详细介绍。

（1）sys\_user 用户表，存储了用户的id，用户名username，昵称nickname，对应的角色id，加密后的密码password，随机6位字符salt，头像路径icon，状态status等。

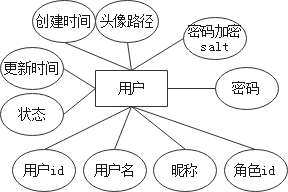
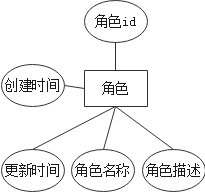


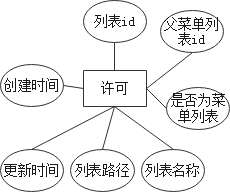
图4.6用户属性图

（2）sys\_role 角色表，分别存了三种用户的id、名称name、描述description等。



4.7角色属性图

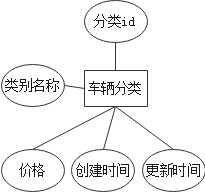
（3）sys\_permission许可表，存储了列表id，列表名称name，对应路径uri，是否为主菜单列表is\_menu，父菜单列表pid（若列表是主菜单为0）等。



4.7许可属性图

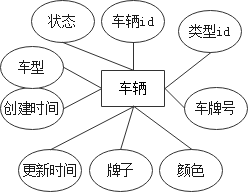
（4）sys\_rel\_role\_permission角色许可表，存储了角色许可id，角色id，对应每个角色对应许可表里列表pemission\_id等。

（5）dic\_classify车辆类型表（共5种类型），存储了类型id，类型名name，类型价格描述description等。



4.8车辆分类属性图

（6）bus\_car公务车辆表，存储了车辆id，类型classify\_id，车牌号number，颜色color，牌子bands，状态status，车型type等。



4.9车辆属性表

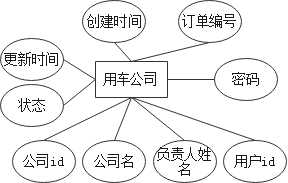
（7）bus\_driver司机表，存储了司机id，对应的用户id，车辆id，名字name，价格price，

信誉积分credit，是否被选中if\_chosen，状态status，编号dno等。



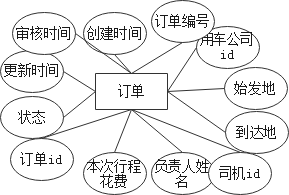
4.10司机属性表

（8）bus\_company用车公司表，存储了公司id，对应用户id，公司名name，负责人姓名leader，用车状态status，订单编号ono等。



4.11用车公司属性表

（9）bus\_order订单表，存储了订单id，司机id，用车公司id，订单号ono，开始用车时间start\_time，结束用车时间end\_time，审核时间review\_time，本次行程花费money，始发地phone，到达地place，订单状态status等。



4.12订单属性表

4.4.2 数据库逻辑设计

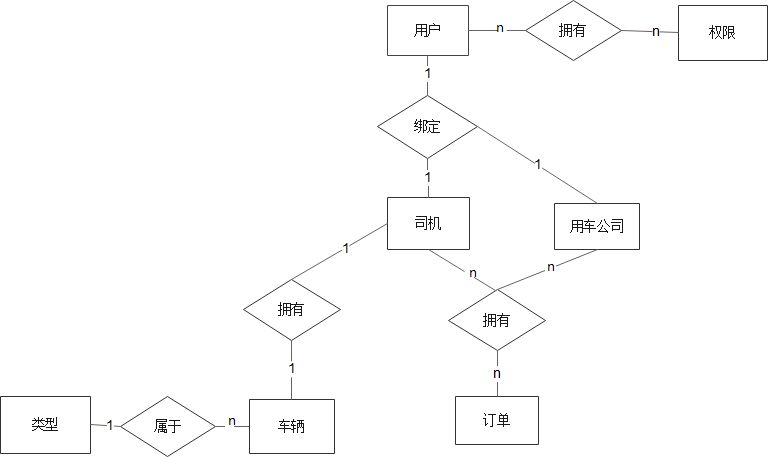


图4.6系统E-R图

4.5本章小结

本章从系统的整体设计开始，通过介绍主要模块来详细介绍系统的设计理念，最后通过数据库每个表间的联系和实体间的联系来建立整个系统数据的架构。

第5章 系统详细设计

本章内容根据系统的需求分析阐述了系统的功能并详细阐述每个功能用到的技术，基于系统需求的视角进行系统设计，同时介绍了不同模块之间的联系，指明了后期系统编码的大概方向。

5.1系统总体详细设计

本系统将充分考虑用户的需求, 并结合时下流行的SSM框架，(Spring MVC、Spring、Mybatis) 、JSP动态网页技术等对系统架构进行良好的设计、功能进行合理划分并实现。

系统主要涉及到用车公司、司机、超级管理员三个用户，针对每一个用户设计了不同的页面列表，这个页面根据登录用户的不同展示不同的列表，用车公司和司机在登录页面点击注册进入注册页面，注册完毕后可登陆进入系统，系统会根据登录的角色来展示不同的页面，同时页面也会有不同的功能供用户使用，整个系统的架构由图2.1系统的总体设计图所示。其中超级管理员的功能最为强大，它可以任意增删改查司机、用车公司、用户和车辆的信息，还可以查看用车过程中订单的状态、完成订单后的投诉情况等等，下面会一一讲解。

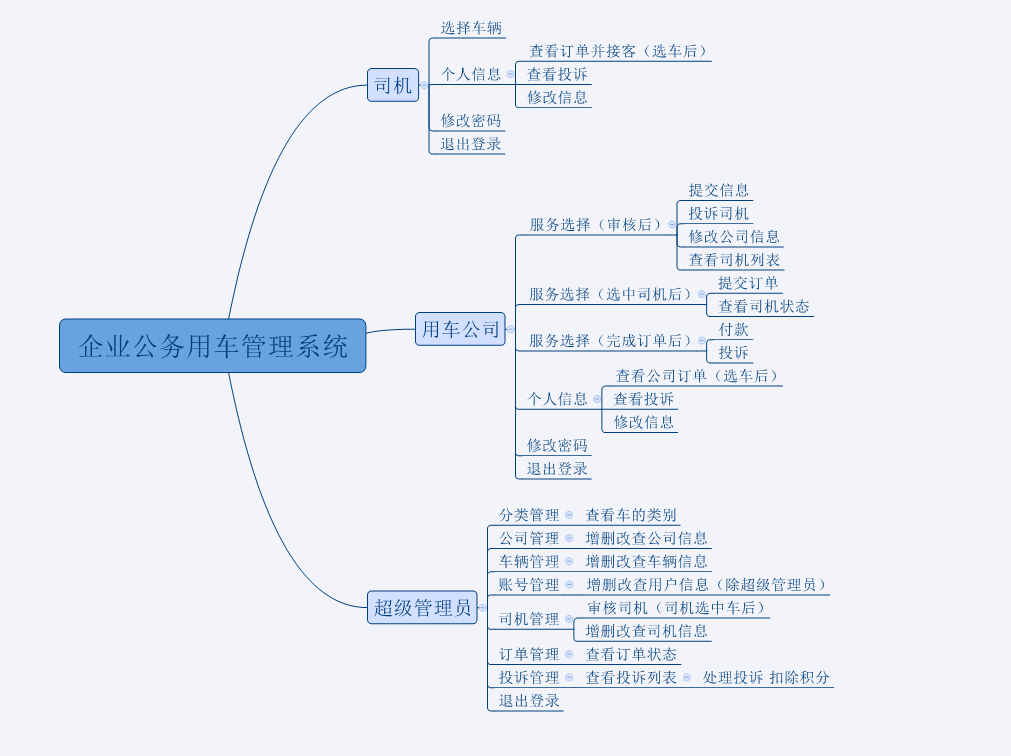


图5.1系统总体详细设计图

5.2系统主要功能模块设计

5.2.1注册登录模块设计

注册登录模块的功能是用户通过注册录入信息，转而进行登录进入系统。注册登录主要的核心类有：AuthController类、UserServiceImpl类、UserService接口、Service接口、ServiceImpl类、UserMapper接口、User类、Role类。注册登录模块核心类的类图如图5.2所示。

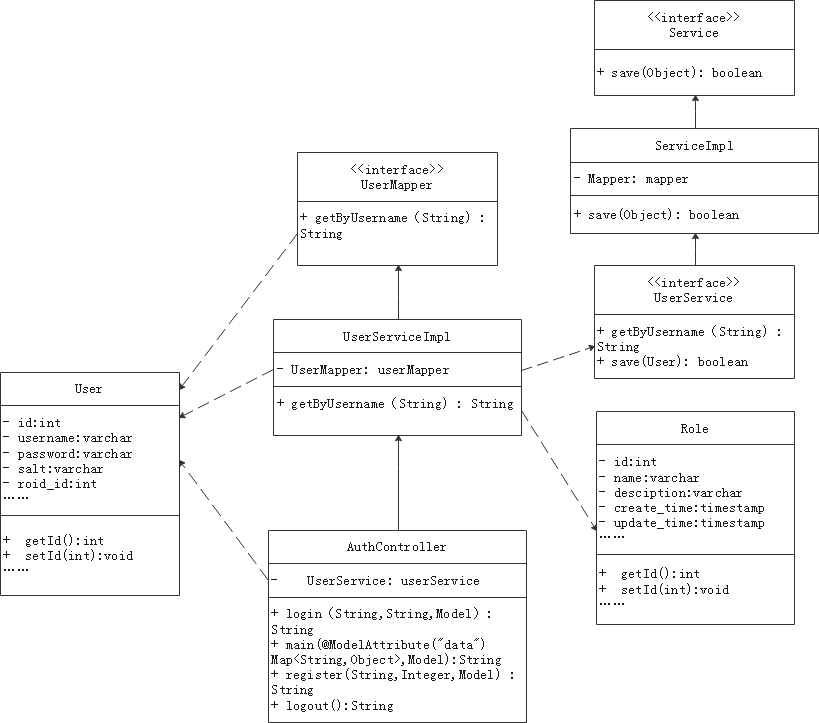


图5.2注册登录模块类图

AuthController类是属于控制层的类，用来处理登录注册进行查询添加的 HTTP 请求，通过调用相关的 Service 层处理具体的业务逻辑，并将处理结果返回给浏览器的前端页面进行展示。类中的主要方法有： 判断用户名密码是否正确login()，跳转到首页main()，注册完毕后保存数据register()，退出登录logout()。

UserServiceImpl 类是属于 Service 层（业务逻辑层）的类，用来处理登录注册相关的业务逻辑。类中的主要方法有：根据输入的用户名获取所有用户信息getByUsername，保存所有注册完毕的用户数据save()。Service和ServiceImpl均为Mybatics-plus提供的类，可直接继承使用它里面的方法。

UserMapper 接口用来与数据库进行交互。接口中的方法与Service相同。

User类、Role类属于实体类，可以在数据库实体属性图中查看各个属性，在此不做赘述。

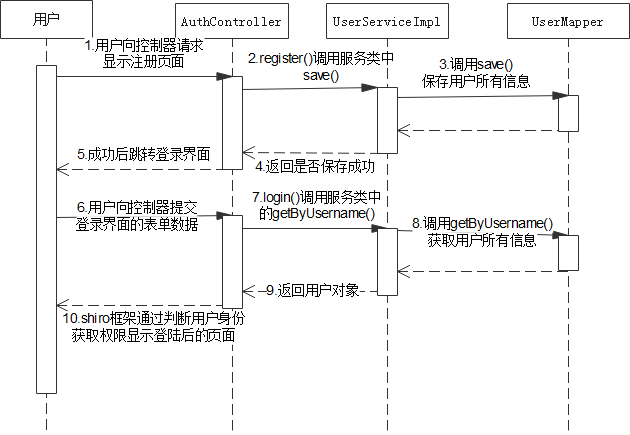


图5.3登录注册模块时序图

5.2.2审核司机模块设计

审核司机模块的功能是用户通过录入个人信息提交，然后管理员登录进行审核。审核司机模块主要的核心类有：JumpController类、BusDriverController类、UserController类、UserServiceImpl类、BusDriverServiceImpl类、BusDriverMapper接口、User类、BusDriver类。审核司机模块核心类的类图如图5.4所示。

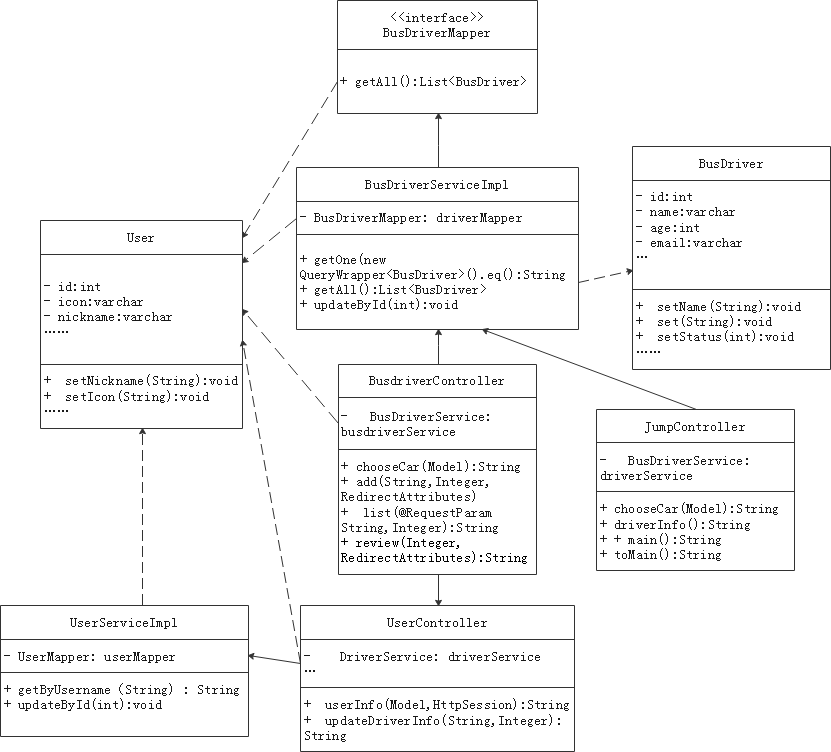


图5.4审核司机模块类图

JumpController类是属于控制层的类，用来处理登录后跳转到对应页面的请求，类中的主要方法有：跳转到选择车辆页面chooseCar、跳转到司机主页面driverInfo()。UserController类封装所有用户进行的操作，主要方法有：展示司机的操作列表userInfo。BusdriverController类主要用来管理员和司机更新司机不同阶段的信息，用到的方法有：选择车辆chooseCar、司机新建自己的信息add、展示司机列表给管理员list、审核司机review。

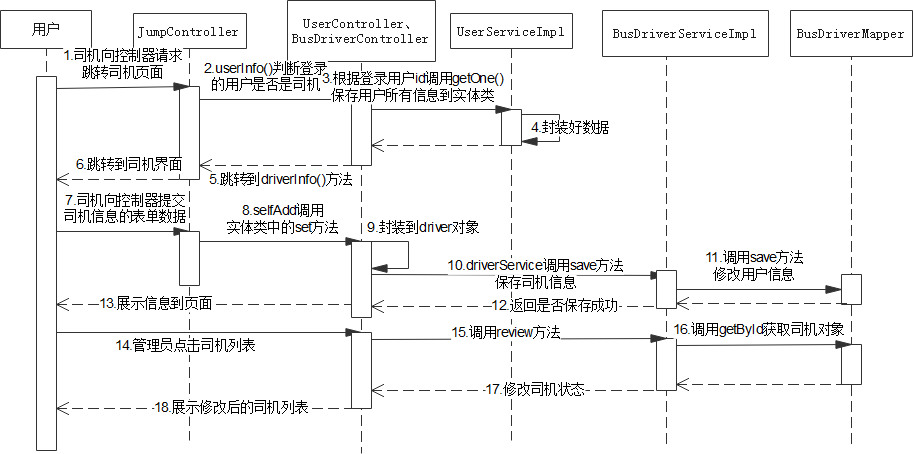
BusDriverServiceImpl 类是属于 Service 层（业务逻辑层）的类，用来处理司机信息相关的业务逻辑。类中的主要方法有： 根据登录用户的id获取司机对象getOne、获取所有司机的信息getAll()、审核后更新司机id updateById。

图5.5审核司机时序图

5.2.3下单模块

下单模块的功能是用车公司通过录入用车提交，然后司机进行接客人等操作完成订单。下单模块主要的核心类有：JumpController类、BusDriverController类、UserController类、GoodsController类、BusOrderServiceImpl类、BusDriverServiceImpl类、BusDriverMapper接口、BusOrderMapper接口、User类、BusDriver类、BusOrder类。审核司机模块核心类的类图如图5.6所示。

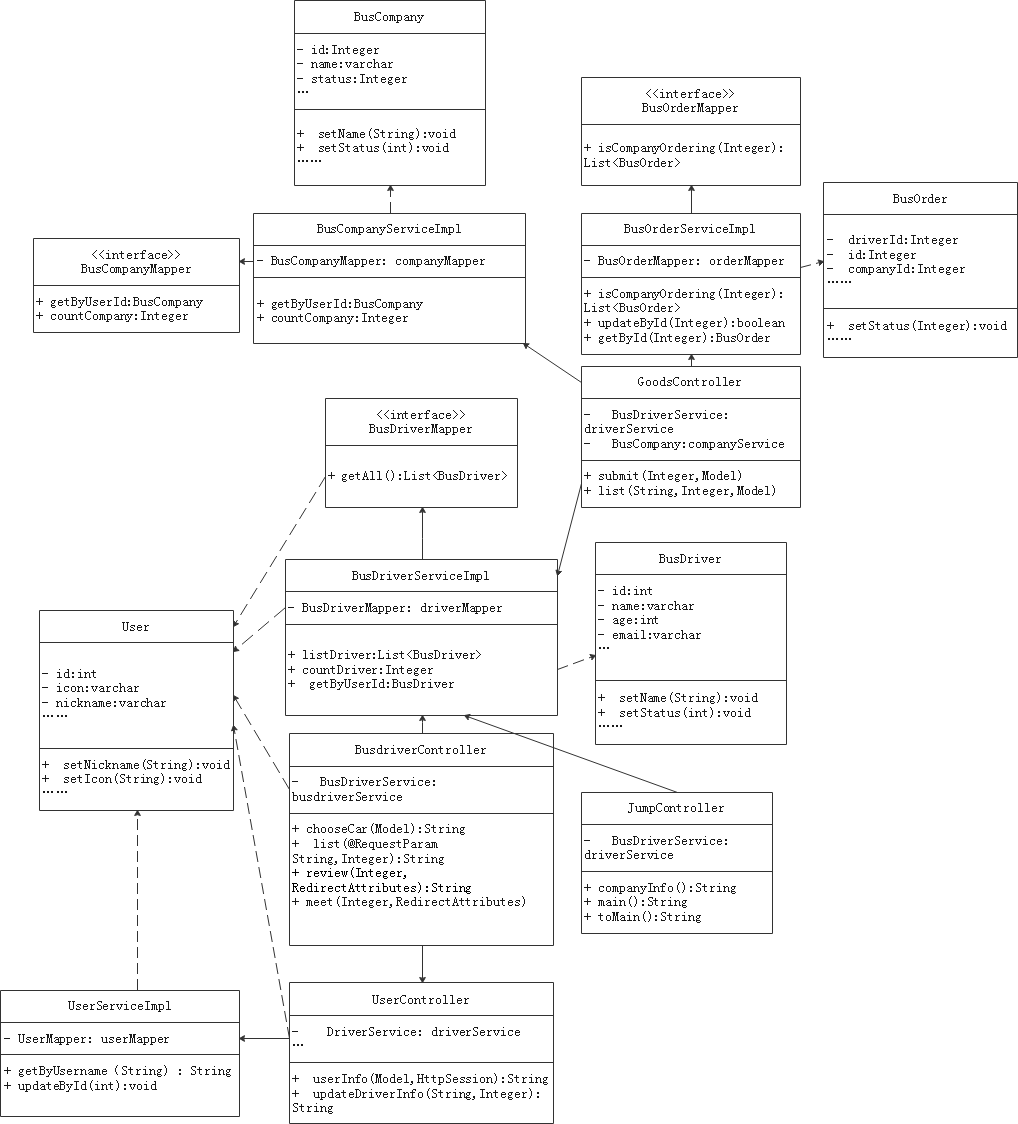


图5.6下单模块类图

JumpController类中的主要方法有：跳转到用车公司主页面companyInfo()、跳到管理员主页面main()、根据登录用户跳转到主页面tomain()。UserController类封装所有用户进行的操作，主要方法有：展示公司的操作列表userInfo、更新司机信息updateDriverInfo。BusdriverController类主要用来管理员和司机更新司机不同阶段的信息，用到的方法有： 展示司机信息列表list、更新用车状态meet。

BusDriverServiceImpl、BusOrderServiceImpl、BusCompanyServiceImpl类是属于 Service 层（业务逻辑层）的类，用来处理下单信息相关的业务逻辑。类中的主要方法有：展示列表listDriver、统计数量countDriver有订单则返回订单列表isCompanyOrdering、根据用户id更新order对象getByid。

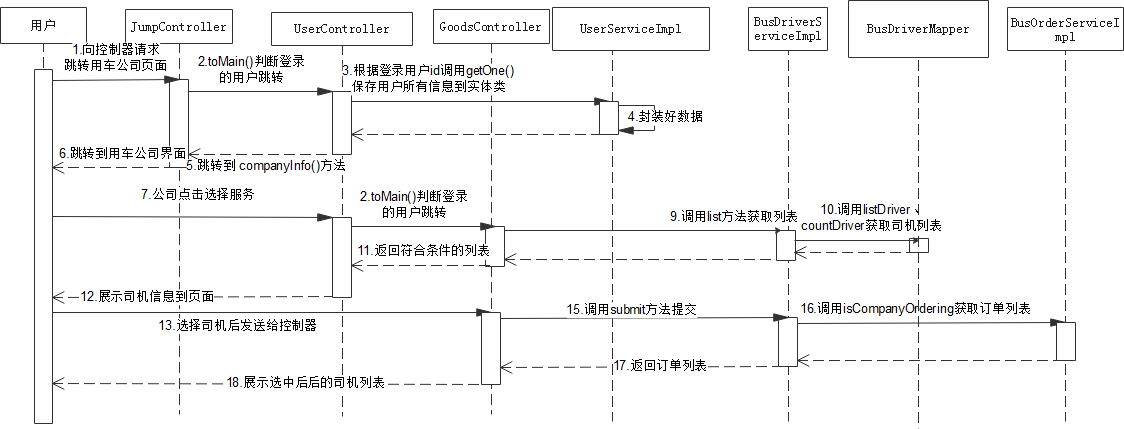


图5.7下单前选车时序图

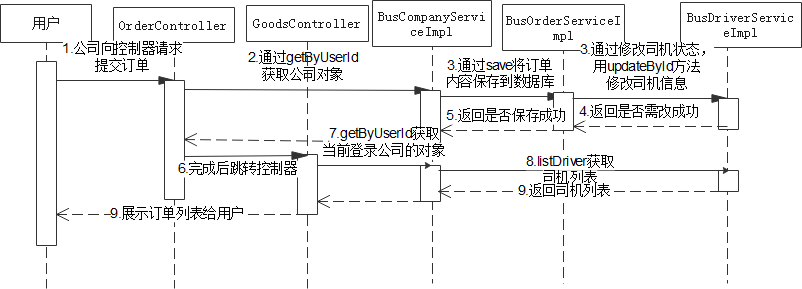


图5.8提交订单时序图

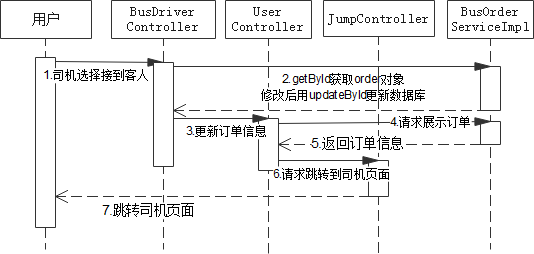


图5.9司机接人时序图

5.2.4投诉模块

投诉模块的功能是用车公司用车结束后投诉，然后管理员处理投诉。投诉模块主要的核心类有：BusDriverController类、UserController类、ComplainController、BusOrderServiceImpl类、BusDriverServiceImpl类、BusDriverMapper接口、BusOrderMapper接口、BusComplainMapperUser类、BusCompany类、BusOrder类。审核司机模块核心类的类图如图5.6所示。

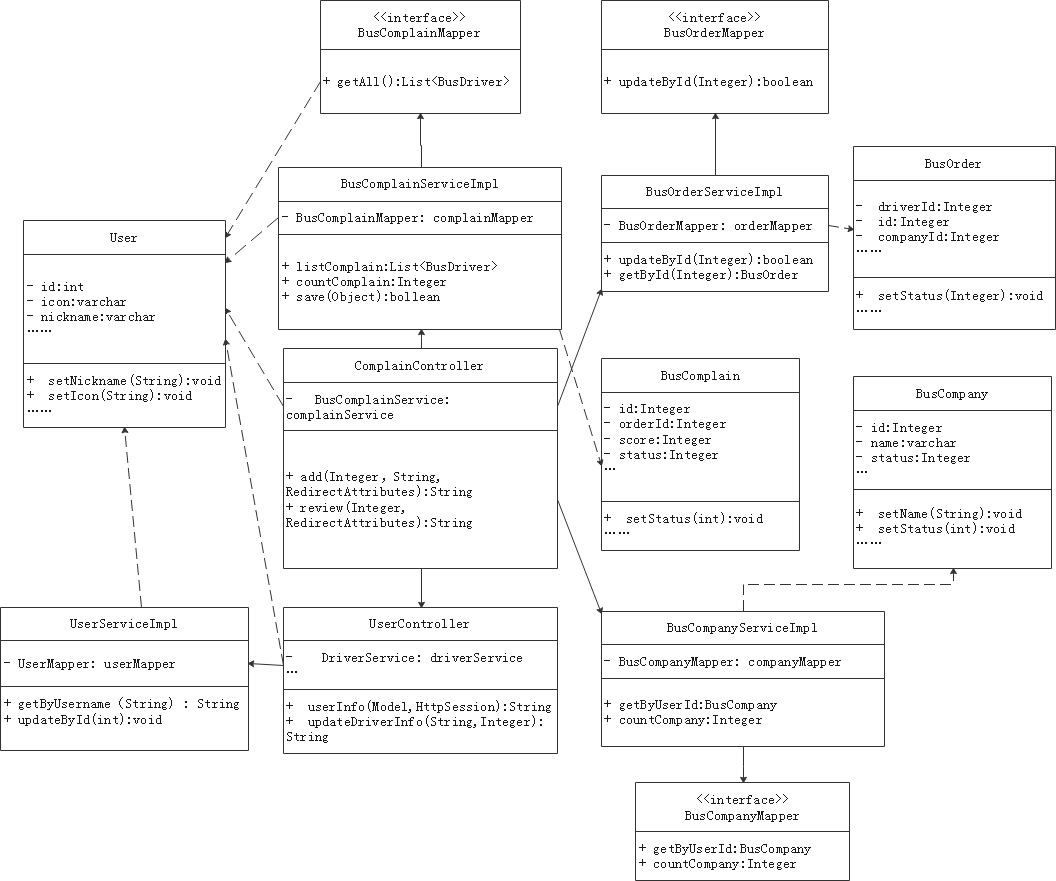
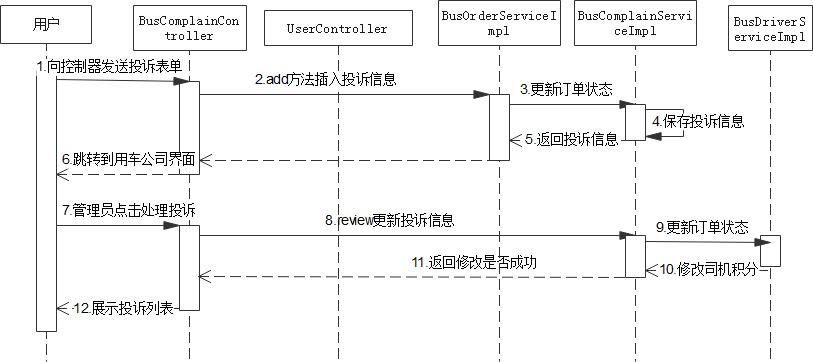


图5.10投诉模块类图

BusComplainController类中的主要方法有：添加投诉信息add、提交投诉信息review。

BusDriverServiceImpl、BusOrderServiceImpl、BusCompanyServiceImpl类是属于 Service 层（业务逻辑层）的类，用来处理投诉信息相关的业务逻辑，类中的主要方法之前相似。

5.11投诉模块时序图

5.3系统活动图设计

（1）审核信息活动图

审核信息是企业公务用车管理系统的重要组成部分，该部分的审核工作，尤其是针对司机信息的审核，均要通过管理员来操作。作为整个系统的起始点，同时也是关键且重要的功能，审核通过才可以依次进行后面的操作。活动图见图 3.4和图3.5。



图5.12审核司机信息活动图

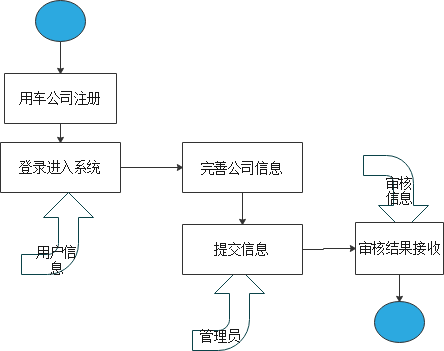


图5.13用车公司信息活动图

（2）企业用车活动图

用车是企业公务用车管理系统的核心功能，整个系统的其他功能都是为了用车功能来服务的，通过三类用户的共享管理数据模式，进而完成无缝传输数据功能，完成用车管理的统一化发展，如图3.6和图3.7。

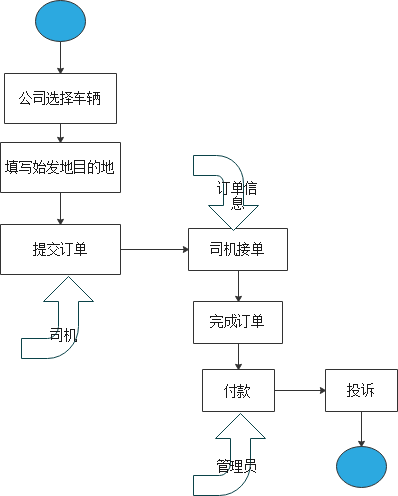


图5.14用车公司用车活动图

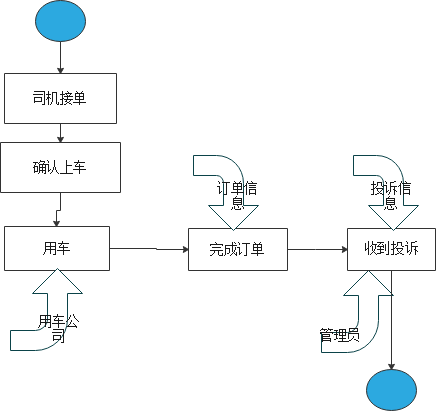


图5.15司机接单活动图

5.4本章小结

本章分模块详细介绍了系统的具体实现，通过类图、时序图和活动图从各个方面讲解了每个模块实现的过程，为接下来完善整个系统做铺垫。

第6章 系统实现

6.1系统环境架构

本系统主要使用了SSM框架架构进行设计，登录权限控制使用了java提供的shiro框架，前端使用了JSP动态网页技术、Javascript、CSS等技术来实现的，数据库使用Mysql数据库, WEB应用服务器使用Tomcat 9.0，如图6.1、图6.2。



图6.1 系统技术架构图

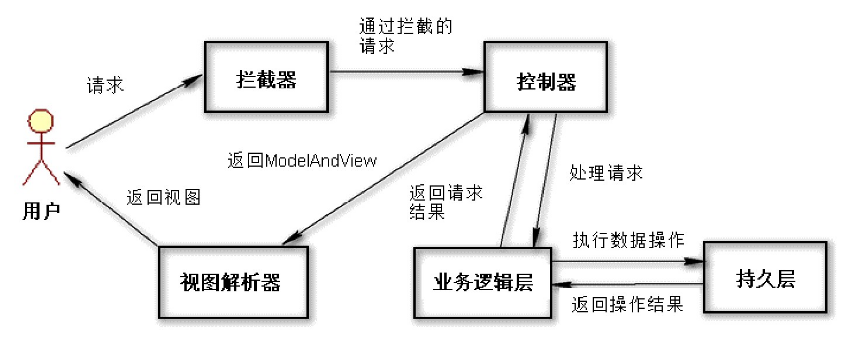


图6.2本系统的SpringMVC工作流程图

6.2系统实现要点

（1）配置shiro

由于企业公务用车系统使用的 Spring 框架，要将 Shiro 集成到Web 应用中，新建一个Shiro配置类，通过利用@Configuration注解在配置文件里，springAOP切面编程动态代理实现（在配置类上添加@EnableAspectJAutoProxy注解，开启注解版的AOP功能），配置Shiro最为核心的安全管理器SecurityManager和注册bean。再配置Shiro的过滤器工厂类，将上一步配置的安全管理器注入，并配置相应的过滤规则。

接下来创建最简单的shiro应用：应用代码通过 Subject 来进行认证和授权，而 Subject 又委托给 SecurityManager； 我们需要给 Shiro 的 SecurityManager 注入 Realm，从而让 SecurityManager 能得到合法 的用户及其权限进行判断。

（2）实现shiro,参考[10]

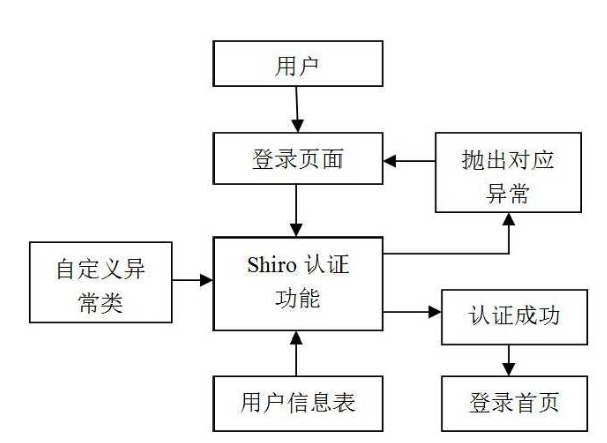


图6.3登录认证模块设计图

首先定义一个自定义标签menu，通过继承SimpleTagSupport类来重写doTag()方法，通过SecurityUtils.getSubject().getPrincipal()抓取登录的用户，根据角色许可表来获取这个角色拥有的列表路径，用getJspContext().getOut().write方法拼接出前台可以识别的列表代码。

（3）用户密码机制：

用户名密码采用DigestUtils实现md5码加密：这是一个非常好用的使用MD5+salt加密的工具类。Salt通过自己设置的随机字符类，通过产生随机数字，从数组里随机取字符来组成6位字符串。使用这个工具类， 从前台拿到密码password，直接MD5Util.getEncryptedPwd(passwd)就可以返回一个长度为56的字符串，可以用来保存到数据库中，相反，登录的时候，因为MD5加密是不可逆的运算，只能拿用户输入的密码走一遍MD5+salt加密之后，跟数据库中password比较，看是否一致，一致时密码相同，登录成功。

（4）实体类设计：

@Data: 注解在类上；提供类所有属性的 get 和 set方法，此外还提供了equals、hashCode、toString 方法.

@TableName（数据库名），声明此对象映射到数据库的数据表，通过它可以为实体指定表@TableId(type = IdType.AUTO)数据库字段注解，非数据库字段注解 @TableField(exist = false)

（5）Controller设计

每个控制器内的方法最初都要检验一下当前用户的角色是否拥有访问权限。

JumpController：通过@GetMapping注解到判断用户角色和提示需要完善信息的方法，大部分操作都通过这个控制器来跳转到相对应页面。

AuthController：封装用户登录注册退出时的方法，通过用前端输入的用户名和密码与数据库里用户表存的密码和salt同时加密后进行比对，结果一致后将用户的所有权限写成列表输出到页面上。

UserController：封装所有用户关于账户信息的操作，注解@Transactional(rollbackFor = RuntimeException.class)来控制出现异常时的数据回滚。

BusCompanyController：封装用车公司可进行的大部分操作。

BusDriverController：封装司机可进行的所大部分操作。

GoodsController、OrderController、ComplainController、DicClassifyController、BusCarController、OrderController：封装对他们各自的操作。

（6）Service设计

设置接口Service，通过继承mybatis-plus提供的IService<T>接口来获取许多便利的方法，然后设置ServiceImpl实现类，继承 ServiceImpl<Mapper, T>方法来具体实现。

（7）Dao层设计

设置接口Mapper，通过继承BaseMapper<T>类提供的方法可以很方便地封装实体对象。

6.3界面展示及对应部分代码

6.3.1登录界面



图6.4登录界面



图6.5注册界面

1. 前台

login.jsp登录页面

<form action="${pageContext.request.contextPath}/auth/login" method="post"></form>

Register.jsp注册界面

<form action="/auth/register" method="post"></form>

（2）控制器

JumpController.java

@RequestMapping("/auth/")

@GetMapping("/login")

public String toLogin(){

User sUser = (User) SecurityUtils.getSubject().getPrincipal();

if (sUser != null){

return "redirect:/main";//跳转到欢迎登录界面

}

return "auth/login"; }

@GetMapping("/main")

public String toMain(){//跳转到管理员管理的主页面

SecurityUtils.getSubject().checkPermission("/main");

User user = (User) SecurityUtils.getSubject().getPrincipal();

BusCompany company = companyService.getByUserId(user.getId());

BusDriver driver = driverService.getByUserId(user.getId());

if (company != null || driver != null){//否则就是管理员登录

return "redirect:/user/info";

}

return "main/index";

}

AuthController.java

@PostMapping("login")//验证用户名密码是否正确

public String login(String username, String password, Model model){

//省略部分代码

User user = userService.getByUsername(username);

password = DigestUtils.md5DigestAsHex((password+user.getSalt()).getBytes());

UsernamePasswordToken token = new UsernamePasswordToken(username,password);

Subject subject = SecurityUtils.getSubject();

try {

subject.login(token);//验证是否能登录

return "redirect:/main";

}catch (AuthenticationException ae){

model.addAttribute("msg","密码错误");

return "redirect:/login";

}

}

@PostMapping("register")

public String register(String nickname,String username, String password,String rePassword,Integer roleId,Model model){

//省略部分代码

User one = userService.getByUsername(username);

String salt = RandomUtil.getRandomString(6);//获取随机6位字符

User user = new User();

user.setSalt(salt);

user.setPassword(MD5Utils.getMD5(password+salt));//加密密码

boolean res = userService.save(user);//利用类中对象的属性信息来修改数据库

return "redirect:/login";

}

（3）UserServiceImpl.java

@Service

public class UserServiceImpl extends ServiceImpl<UserMapper, User> implements UserService {

@Autowired

private UserMapper userMapper;

@Override

public User getByUsername(String username) {

return userMapper.getByUsername(username);

}

6.3.2管理员管理界面



图6.6管理员界面

1. 前台页面

<aside class="main-sidebar">

<section class="sidebar">

<ul class="sidebar-menu" data-widget="tree">

<mytag:menu/>//展示侧拉列表的目录

</ul>

</section>

</aside>

1. 控制器

JumpController.java

@GetMapping("/")

public String main(){

return "redirect:/main";

}

@GetMapping("/main")

public String toMain(){

SecurityUtils.getSubject().checkPermission("/main");

User user = (User) SecurityUtils.getSubject().getPrincipal();

BusCompany company = companyService.getByUserId(user.getId());

BusDriver driver = driverService.getByUserId(user.getId());

if (company != null || driver != null){//否则就是管理员

return "redirect:/user/info";

}

return "main/index";

}

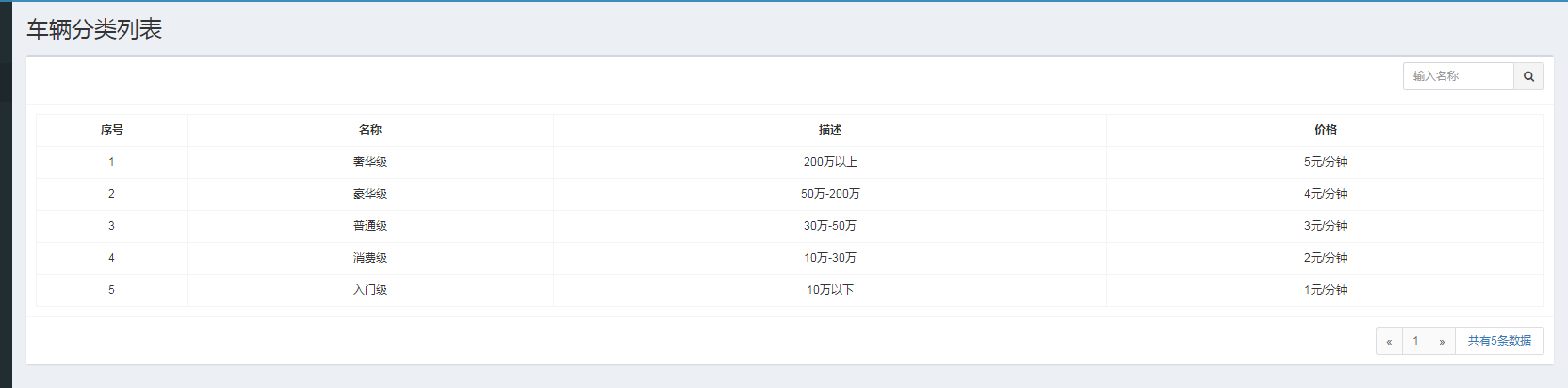


图6.7车辆管理界面

DicClassifyController.java

@RequestMapping("/classify/")

public class DicClassifyController {

@RequestMapping("list")//展示分类列表

public String list(@RequestParam(required = false,defaultValue = "")String name,

@RequestParam(required = false,defaultValue = "")String description,

@RequestParam(required = false,defaultValue = "1") Integer page,

@RequestParam(required = false,defaultValue = "10")Integer pageSize, Model model){

SecurityUtils.getSubject().checkPermission("/classify/list");//检查权限

SecurityUtils.getSubject().checkRole("admin");

// 筛选条件

DicClassify classify = new DicClassify();

if (StringUtils.isNotBlank(name)){

classify.setName(name);

}

if (StringUtils.isNotBlank(description)){

classify.setDescription(description);

}

if(!Objects.equals(0,pageSize)){

classify.setSkip((page-1)\*pageSize);

classify.setSize(pageSize);

}

List<DicClassify> list = dicClassifyService.listClassify(classify);

Integer total = dicClassifyService.countClassify(classify);

int pages = total/pageSize + (total % pageSize==0?0:1);

model.addAttribute("dataList",list);

model.addAttribute("total",total);

model.addAttribute("pages",pages);

return "classify/classifyList";

}



图6.8账号管理界面

UserController.java

@RequestMapping("list")

public String list(@RequestParam(required = false,defaultValue = "")String nickname,

@RequestParam(required = false,defaultValue = "1") Integer page,

@RequestParam(required = false,defaultValue = "10")Integer pageSize, Model model){

SecurityUtils.getSubject().checkPermission("/user/list");

SecurityUtils.getSubject().checkRole("admin");

// 筛选条件

User user = new User();

if (StringUtils.isNotBlank(nickname)) {

user.setNickname(nickname);

}

if(!Objects.equals(0,pageSize)){

user.setSkip((page-1)\*pageSize);

user.setSize(pageSize);

}

List<User> userList = userService.listUser(user);

Integer total = userService.countUser(user);

List<Role> roleList = roleService.list();

int pages = total/pageSize + (total % pageSize==0?0:1);

model.addAttribute("dataList",userList);

model.addAttribute("roleList",roleList);

model.addAttribute("total",total);

model.addAttribute("pages",pages);

return "user/userList";

@RequestMapping("update")//更新账户信息

public String update(Integer id,String nickname,String username,String password,Integer roleId,Model model){

SecurityUtils.getSubject().checkPermission("/user/update");

SecurityUtils.getSubject().checkRole("admin");

User one = userService.getOne(

new QueryWrapper<User>()

.eq("username",username)

.ne("id",id)//条件构造器：eq判断是否相等，ne判断不相等

);

//省略判断合理性代码

User user = userService.getById(id);

user.setNickname(nickname);

user.setUsername(username);

String salt = RandomUtil.getRandomString(6);

String md5 = DigestUtils.md5DigestAsHex((password + salt).getBytes());

user.setSalt(salt);

user.setPassword(md5);

user.setRoleId(roleId);

boolean res = userService.updateById(user);

model.addAttribute("msg","修改成功");

return "redirect:list";

}



图6.9用车公司管理

UserController.java

@RequestMapping("add")

public String add(String name, Integer userId, String leader, String email, RedirectAttributes attributes){

SecurityUtils.getSubject().checkPermission("/company/add");

BusCompany company = new BusCompany();

company.setName(name);

company.setLeader(leader);

company.setEmail(email);

boolean res = companyService.save(company);

if (!res){

attributes.addAttribute("msg","添加失败,请联系管理员");

return "redirect:list";

}

attributes.addAttribute("msg","添加成功");

return "redirect:list";

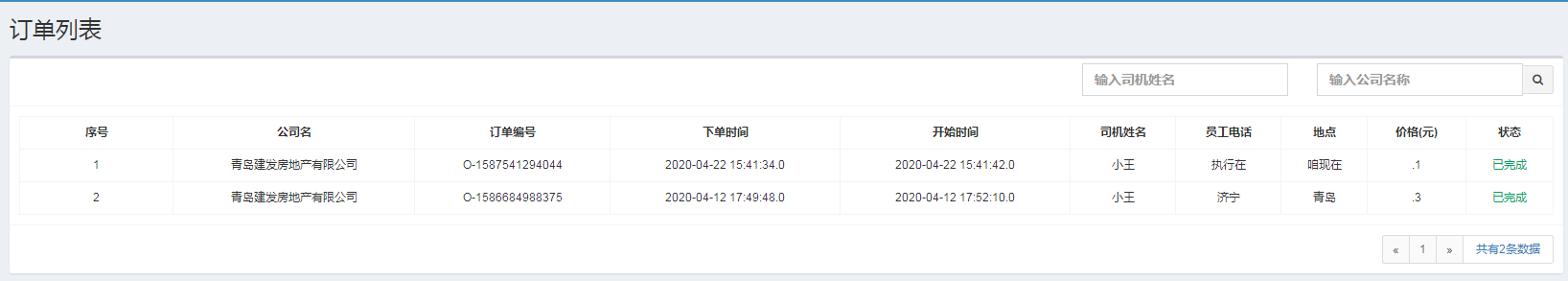
}



图6.10车辆管理



图6.11司机管理



6.12订单管理



6.13投诉管理

BusComplainController.java

@RequestMapping("/complain/")

@RequestMapping("review")//提交投诉

@Transactional(rollbackFor = RuntimeException.class)

public String review(Integer complainId,Integer score, RedirectAttributes attributes){

SecurityUtils.getSubject().checkPermission("/complain/review");

SecurityUtils.getSubject().checkRole("admin");

BusComplain complain = complainService.getById(complainId);

//set方法

boolean res = complainService.updateById(complain);

if (!res){

throw new RuntimeException("更新投诉信息失败");

}

BusOrder order = orderService.getById(complain.getOrderId());

BusDriver driver = driverService.getById(order.getDriverId());

if (driver.getCredit() < score){

attributes.addAttribute("msg","扣除积分已经大于当前积分");

return "redirect:list";

}

driver.setCredit(driver.getCredit() - score);//管理员传来扣除的信誉积分score，用司机现有的减去管理员扣除的

driver.setUpdateTime(null);

boolean b = driverService.updateById(driver);

if(!b){

throw new RuntimeException("更新司机信息失败");

}

attributes.addAttribute("msg","处理成功");

return "redirect:list";//跳到展示列表方法

}

}

6.3.3用车公司管理界面



图6.14选择已有信息或新建公司信息

UserController.java

@PostMapping("updateDriverInfo")

@Transactional(rollbackFor = RuntimeException.class)

public String updateDriverInfo(String nickname, String icon,String name,Integer gender,String age,String phone,String email,HttpSession session,RedirectAttributes attributes){

SecurityUtils.getSubject().checkRole("driver");

Subject subject = SecurityUtils.getSubject();

User user = (User) subject.getPrincipal();

BusDriver driver = (BusDriver) session.getAttribute("driver");

user.setNickname(nickname);

user.setIcon(icon);

driver.setName(name);

driver.setGender(gender);

int intAge;

try {

intAge = Integer.parseInt(age);

}catch (NumberFormatException e){

attributes.addAttribute("msg","年龄应是整数");

return "redirect:/driverInfo";

}

if (intAge <= 0 || intAge > 100){

attributes.addAttribute("msg","年龄应大于0小于100");

return "redirect:/driverInfo";

}

//set方法

boolean res1 = userService.updateById(user);

PrincipalCollection principals = subject.getPrincipals();

String realmName = principals.getRealmNames().iterator().next();

PrincipalCollection principalCollection

= new SimplePrincipalCollection(user,realmName);

subject.runAs(principalCollection);

BusCar car = carService.getById(driver.getCarId());

session.setAttribute("car", car);

session.setAttribute("driver", driver);

attributes.addAttribute("msg","修改成功");

return "redirect:/driverInfo";

}



图6.15用车公司主页面

UserController.java

@RequestMapping("info")

public String userInfo(Model model,HttpSession session){

User user = (User) SecurityUtils.getSubject().getPrincipal();

if(user.getRoleId() == 2) {//使用者为用车公司时

BusCompany company = companyService.getByUserId(user.getId());

if (company != null){

List<BusOrder> orderList = orderService.getByCompanyId(company.getId());

List<BusComplain> complainList = complainService.listByCompanyId(company.getId());

session.setAttribute("company", company);

session.setAttribute("complainList", complainList);

session.setAttribute("orderList", orderList);

return "redirect:/companyInfo";

}else {

List<BusCompany> companyList = companyService.getAll();

model.addAttribute("companyList",companyList);

return "company/companyChoose";

}

} else if(user.getRoleId() == 3) {//使用者为司机时

BusDriver driver = driverService.getOne(

new QueryWrapper<BusDriver>()

.eq("user\_id",user.getId())

);

if (driver != null){

BusCar car = carService.getById(driver.getCarId());

List<BusOrder> orderList = orderService.getByDriverId(driver.getId());

List<BusComplain> complainList = complainService.listByDriverId(driver.getId());

session.setAttribute("car", car);

session.setAttribute("driver", driver);

session.setAttribute("complainList", complainList);

session.setAttribute("orderList", orderList);

return "redirect:/driverInfo";

}else {

List<BusDriver> driverList = driverService.getAll();

model.addAttribute("driverList",driverList);

return "driver/driverChoose";

}

}else {

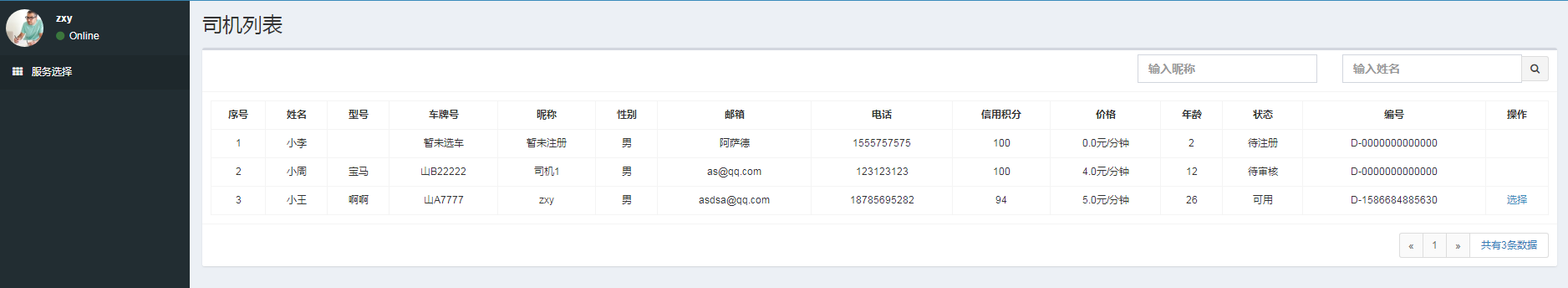
return "redirect:/main";

}

}



6.16我的投诉板块



6.17服务选择页面

6.3.4司机管理界面

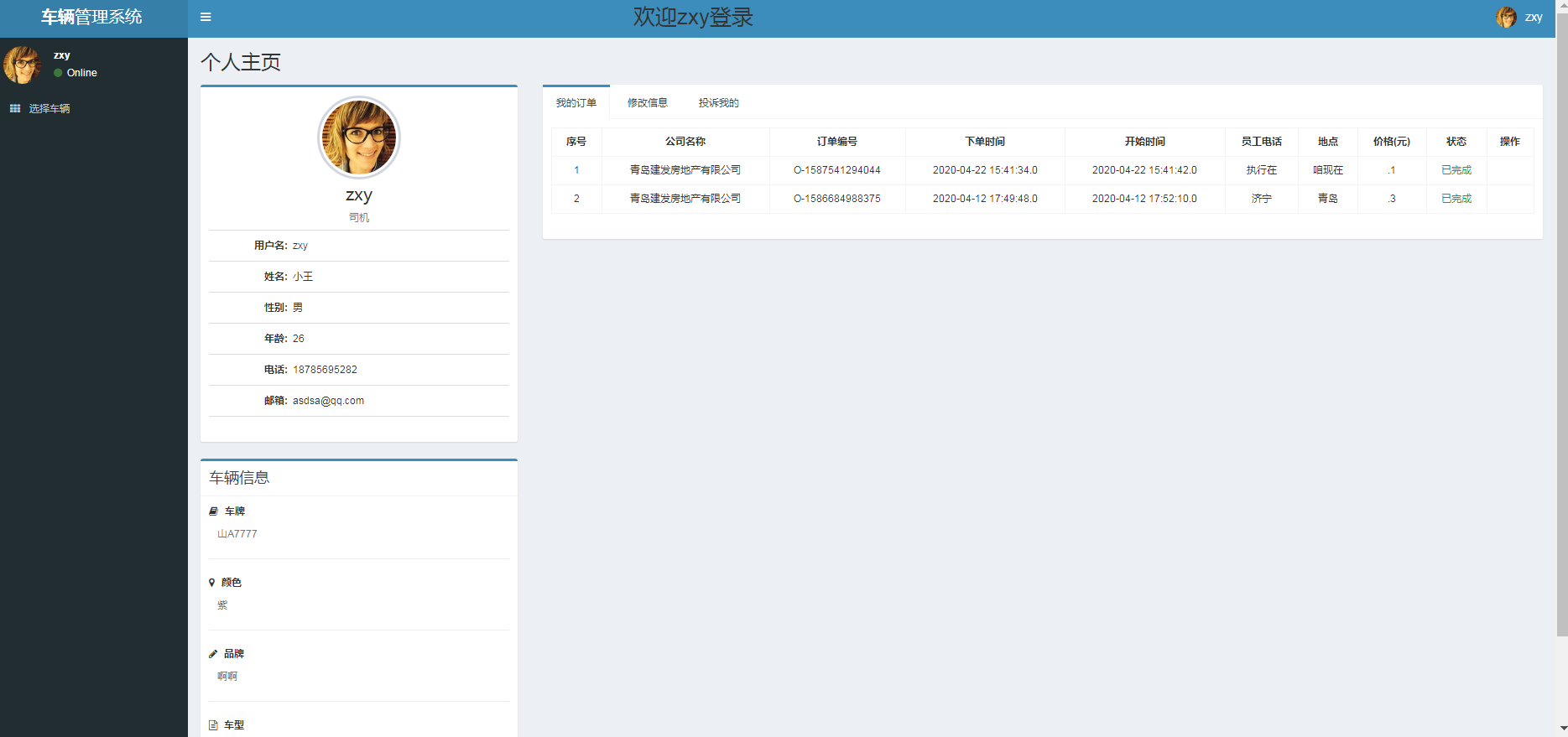


图6.18司机主页面

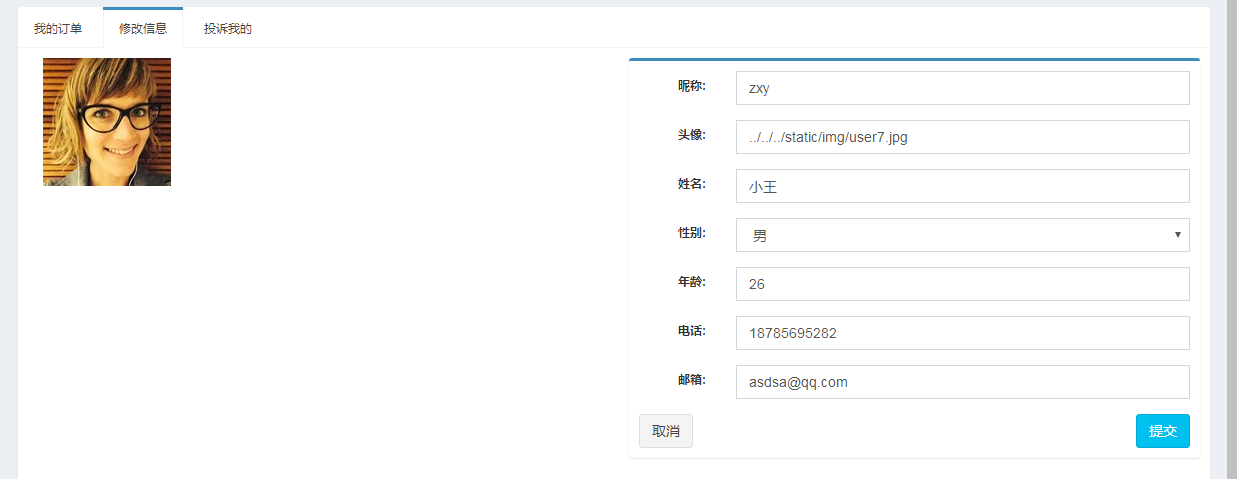


图6.19修改个人信息



图6.20选择车辆

BusDriverController.java

@RequestMapping("chooseCar")

@Transactional(rollbackFor = RuntimeException.class)

public String chooseCar(Integer carId, RedirectAttributes attributes) {

SecurityUtils.getSubject().checkPermission("/chooseCar/list");

SecurityUtils.getSubject().checkRole("driver");

BusCar car = carService.getById(carId);

//省略判断数据代码

car.setStatus(Config.CAR\_USING);

User user = (User) SecurityUtils.getSubject().getPrincipal();

BusDriver driver = driverService.getOne(

new QueryWrapper<BusDriver>()

.eq("user\_id", user.getId())

);

DicClassify classify = classifyService.getById(car.getClassifyId());

driver.setPrice((double)classify.getId());

driver.setStatus(Config.DRIVER\_NOT\_REVIEW);

driver.setCarId(carId);

boolean res = driverService.updateById(driver);

if (!res) {

attributes.addAttribute("msg", "更新失败,请联系管理员");

} else {

attributes.addAttribute("msg", "更新成功");

}

return "redirect:/chooseCar/list";

}



6.21投诉我的列表

BusDriverController.java

@RequestMapping("arrive")//用车完毕后，根据用车时间修改订单信息

@Transactional(rollbackFor = RuntimeException.class)

public String arrive(Integer orderId,RedirectAttributes attributes){

SecurityUtils.getSubject().checkPermission("/driver/arrive");

SecurityUtils.getSubject().checkRole("driver");

User user = (User) SecurityUtils.getSubject().getPrincipal();

BusDriver driver = driverService.getByUserId(user.getId());

BusOrder order = orderService.getById(orderId);

order.setStatus(Config.ORDER\_NOT\_PAY);//修改订单状态

order.setEndTime(new Timestamp(System.currentTimeMillis()));

long time = order.getEndTime().getTime() - order.getReviewTime().getTime();

double money = driver.getPrice() \* (time / 60000.0) \* driver.getCredit() / 100;//根据用车时间计算相对应的价格

order.setMoney(money);

order.setUpdateTime(null);//用车完毕要清零

boolean res1 = orderService.updateById(order);

if (!res1){

throw new RuntimeException("更新订单数据失败,请联系管理员");

}

driver.setStatus(Config.DRIVER\_NORMAL);

driver.setUpdateTime(null);

boolean res2 = driverService.updateById(driver);

if (!res2){

throw new RuntimeException("更新司机数据失败,请联系管理员");

}

attributes.addAttribute("msg", "更新成功");

return "redirect:/user/info";//根据登录用户，跳转到对应主页面

}

6.4系统总结

目前本系统应用的技术还达不到三个用户的页面同时存在，导致用车时候需要来回切换用户，若采用分布式系统可以解决这个问题。

6.5本章小结

本章主要展示了界面环境设计、登录权限设计和整个系统的界面设计的代码，可以清楚地看见每个模块对应的代码是如何设计的，由于篇幅不够只展示了部分关键代码。

致谢

首先，我要衷心感谢一直以来给予我无私帮助和关爱的老师们，特别是我的导师胡克勇老师，以及班主任安昭老师、李建萍老师等，谢谢您们这四年以来对我的关心和照顾，从您们身上，我学会了如何学习，如何工作，如何做人。其次，我还要认真地谢谢我身边所有的朋友和同学。最后，我要感谢我的父母及家人，没有人比你们更爱我，你们对我的关爱让我深深感受到了生活的美好，谢谢你们一直以来给予我的理解、鼓励和支持，你们的鼓励是我不断取得进步的永恒动力。

参考文献：

[1]李洋.SSM框架在Web应用开发中的设计与实现[J].计算机技术与发展,2016,26(12):190-194.

[2] 陈辉.软件工程中的解耦核心技术MVC和三层架构比较[J].廊坊师范学院学报 (自然科学版) .

[3]李春燕;李根.基于JSP技术的网络信息教学平台设计[J/OL].电子技术与软件工程，2017,（20）：9（2017-10-26）。

[4] 丁洁;.基于Shiro的Web应用安全框架设计研究[J].信息与电脑(理论版),2018 ,No.407(13):38-39.

[5] 韦美雁;段华斌;周新林;.大数据环境下的MySQL优化技术探讨[J].现代计算机(专业版),2018,No.630(30):68-72.

[6] 叶明.网络约租车监管系统技术方案分析[J].交通与港航,2016,3(04):43-45+76.DOI:10.16 487/j.cnki.issn2095-7491.2016.04.010

[7] 于明鹭,刘南杰,赵海涛,彭江琴.基于车联网的智能打车系统[J].计算机技术与发展,2016,26(02):118-123.

[8] 李营.汽车租赁信息系统的设计与实现[J].数字技术与应用,2014,(12):146.DOI:10.19695/ j.cnki.cn12-1369.2014.12.096

[9] 倪晓锋.基于SSM和Shiro的火电SIS系统的设计与实现[J].工业控制计算机,2019,32(09):131-132+134.

[10] 易文康,程骅,程耕国.Shiro框架在Web系统安全性上的改进与应用[J].计算机工程,2018,44(11):135-139.DOI:10.19678/j.issn.1000-3428.0048417