**基于Java的**

**共享单车租赁系统的设计与实现**

|  |  |  |  |
| --- | --- | --- | --- |
| 学 院 | 计算机学院 | | |
| 专 业：  姓 名：  指导老师： | 软件工程 | | |
| 陈锦涛 | 学 号：  职 称： | 140202031036 |
| 王 琳 | 讲 师 |

中国·珠海

二○一八 年 五 月

**诚信承诺书**

**本人郑重承诺：**本人承诺呈交的毕业设计《基于Java的

共享单车租赁系统的设计与实现》是在指导教师的指导下，独立开展研究取得的成果，文中引用他人的观点和材料，均在文后按顺序列出其参考文献，设计使用的数据真实可靠。

本人签名：

日期： 年 月 日

**基于Java的共享单车租赁系统的设计与实现**

**摘 要**

共享单车租赁系统（Share Bike Lease System，简称SBLS）是一个校内共享单车的管理系统。在基于MVC的三层开发架构体系上，采用Java作为主要开发语言，结合SpringMVC+Spring+Mybatis三大开发框架进行该B/S系统的开发。一般来说，B/S系统的开发都基于MVC（Moel-View-Control）三层架构体系，使得前台和后台解耦，从而使得各自的业务逻辑互不影响。前台页面交给View展示，后台业务逻辑交给Model处理，Control负责控制两者的同步。这种开发架构体系能够很好的分离逻辑、数据与页面，有助于管理复杂的应用程序。本系统基于MVC体系开发，增加了系统的稳定性，同时也为管理校内共享单车提供了有效助力。

**关键词：**Java；共享单车；MVC；SSM

**Share Bike Lease System**

**Abstract**

(未完成)Continuously variab.

**Keywords:**  Continuously variable transmission;

**目 录**

[1前言 1](#_Toc512091089)

[1.1背景介绍 1](#_Toc512091090)

[1.2技术选择与基本思路 1](#_Toc512091091)

[1.2.1技术选择 1](#_Toc512091092)

[1.2.2基本思路 1](#_Toc512091093)

[1.3本系统应解决的主要问题 1](#_Toc512091094)

[2需求分析 2](#_Toc512091095)

[2.1功能清单 2](#_Toc512091096)

[2.1.1用户功能清单 2](#_Toc512091097)

[2.1.2管理员功能清单 2](#_Toc512091098)

[2.3用例文本 3](#_Toc512091099)

[2.4领域模型 16](#_Toc512091100)

[2.5非功能需求 17](#_Toc512091101)

[2.5.1性能和容量 17](#_Toc512091102)

[2.5.2安全性 17](#_Toc512091103)

[2.5.3易用性 18](#_Toc512091104)

[2.5.4可移植性 18](#_Toc512091105)

[2.5.5可扩展性 18](#_Toc512091106)

[2.5.6可靠性 18](#_Toc512091107)

[3概要设计 19](#_Toc512091108)

[3.1逻辑视图 19](#_Toc512091109)

[3.2开发视图 20](#_Toc512091110)

[3.3物理视图 21](#_Toc512091111)

[4数据库设计 22](#_Toc512091112)

[4.1E-R图 22](#_Toc512091113)

[4.2数据库表设计 22](#_Toc512091114)

[4.3数据字典 22](#_Toc512091115)

[5详细设计 25](#_Toc512091116)

[5.1登录 25](#_Toc512091117)

[5.2定位 26](#_Toc512091118)

[5.3租车 27](#_Toc512091119)

[5.4还车 29](#_Toc512091120)

[5.5金额充值 30](#_Toc512091121)

[5.6重置密码 32](#_Toc512091122)

[5.7单车数据统计 34](#_Toc512091123)

[5.8删除单车 35](#_Toc512091124)

[5.9修理单车 36](#_Toc512091125)

[5.10投放单车 37](#_Toc512091126)

[5.11修改站点信息 39](#_Toc512091127)

[5.12新增站点 40](#_Toc512091128)

[5.13删除站点 41](#_Toc512091129)

[6界面设计 43](#_Toc512091130)

[6.1登录 43](#_Toc512091131)

[6.2用户操作页面 43](#_Toc512091132)

[6.3管理员操作页面 44](#_Toc512091133)

[7代码实现 46](#_Toc512091134)

[7.1登录 46](#_Toc512091135)

[7.2定位 47](#_Toc512091136)

[7.3租车 47](#_Toc512091137)

[7.4还车 49](#_Toc512091138)

[7.4金额充值 50](#_Toc512091139)

[7.5重置密码 52](#_Toc512091140)

[7.6单车数据统计 54](#_Toc512091141)

[7.7单车管理 54](#_Toc512091142)

[7.8删除单车 55](#_Toc512091143)

[7.9修理单车 56](#_Toc512091144)

[7.10投放单车 57](#_Toc512091145)

[7.11站点管理 57](#_Toc512091146)

[7.12修改站点信息 58](#_Toc512091147)

[7.13新增站点 58](#_Toc512091148)

[7.14删除站点 59](#_Toc512091149)

[8测试 61](#_Toc512091150)

[8.1 测试方案与策略 61](#_Toc512091151)

[8.2 测试用例 61](#_Toc512091152)

[8.2.1登录 61](#_Toc512091153)

[8.2.2定位 61](#_Toc512091154)

[8.2.3租车 62](#_Toc512091155)

[8.2.4还车 62](#_Toc512091156)

[8.2.5金额充值 62](#_Toc512091157)

[8.2.6单车管理 63](#_Toc512091158)

[8.2.7站点管理 63](#_Toc512091159)

[9总结 64](#_Toc512091160)

# 1前言

1.1背景介绍

随着当代信息科学技术的高速发展，如今的计算机与IT领域，人工智能与大数据已经逐步迈进中国市场。在人们满足了普通物质需求后，开始追求更高层次的生活的同时，‘共享’经济应运而生。从一开始的共享单车，到现在的共享汽车，这些新型企业的诞生与如今高速发展的IT技术密切相关。就好比共享单车，如何利用地图进行单车定位，如何利用大数据来收集数据并分析。因此，利用这些新兴技术，结合所学知识所设计的共享单车租赁系统，是一个有挑战性也不失实际性的尝试。基于这种市场需求，本毕设设计并实现一个基于Java的共享单车租赁系统（以下简称本系统）。

1.2技术选择与基本思路

1.2.1技术选择

1. 系统开发平台：Windows10
2. 系统开发语言：Java
3. 系统开发数据库：MySQL5.7
4. 操作浏览器：360极速浏览器
5. Web服务器：Tomcat7.0
6. IDE工具：Eclipse Oxygen.3 Release (4.7.3)

1.2.2基本思路

本租赁系统基于B/S架构的开发，采用java的编程语言以及SSM的框架技术和MySQL数据库。采用面向对象的设计与分析的设计思想。在基于Java的面向接口的程序设计中，结合面向对象的设计思想。

1.3本系统应解决的主要问题

1. 用户对当前位置的定位；
2. 用户对当前位置范围内的站点查询；
3. 管理员对单车数据的统计；
4. 程序运行的稳定性与安全性。

# 2需求分析

2.1功能清单

2.1.1用户功能清单

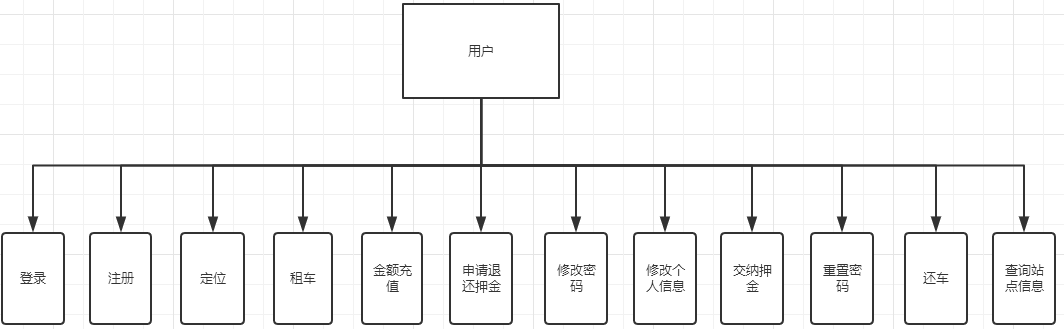
1. 注册与登录。
2. 定位当前位置。
3. 定位：用户定位自己当前所处位置。
4. 查询站点信息：用户在定位后，可以查询到附近半径范围500米内的单车站点信息。例如：该站点目前有多少单车可以使用。
5. 租车：用户可以在选定站点内，选中某辆单车并开始租赁使用。
6. 归还单车：用户在结束单车的租赁后，将单车归还至某一站点，并在系统中结算。
7. 金额充值：当用户第一次租车时，需交押金。用户在租赁单车时，系统会根据骑行距离和骑行时间产生费用，费用会在用户的余额扣除。余额不足时，用户需充值。
8. 申请退还押金：用户可以在没有租车的时候，申请退还押金。
9. 修改个人信息。
10. 修改密码。
11. 重置密码。

图2.1 用户功能清单图

2.1.2管理员功能清单

1. 登录。
2. 管理站点信息：管理员设定站点内有多少位置可供单车停放，某站点有多少辆单车可使用等信息。
3. 新增站点。
4. 管理单车：新增单车，删除单车，修改某辆单车的信息。
5. 投放单车：管理员将经过修理的单车投放到站点进行使用。
6. 更新单车状态：管理员维护单车的现有状态。单车的状态包括：使用中、可使用、报修等。
7. 统计单车的使用数据：管理员可以统计当前时间内有多少单车在使用，多少单车正在报修，多少辆单车是闲置状态。

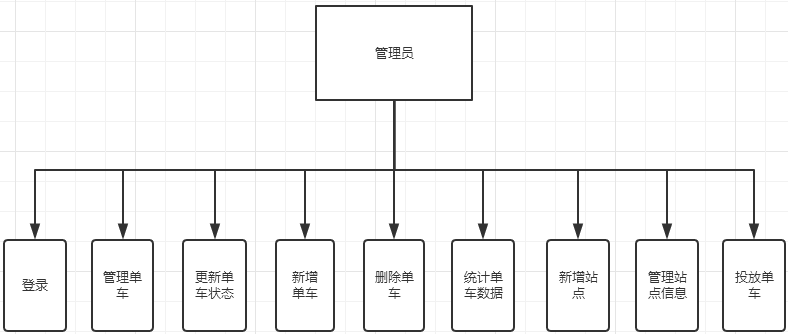


图2.2 管理员功能清单图

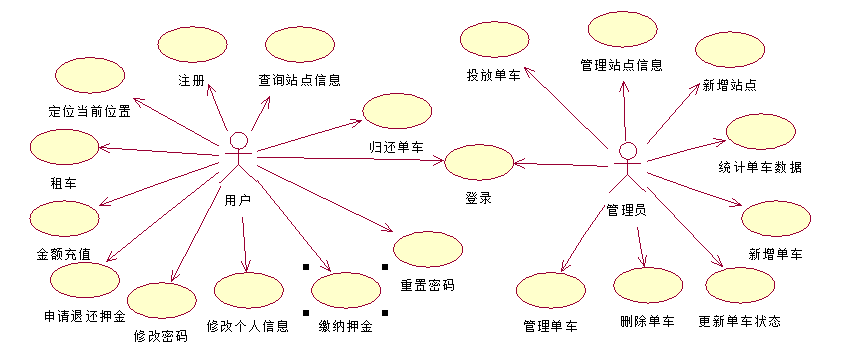
2.2用例图

图2.3 用例图

2.3用例文本

|  |
| --- |
| **用例名称：**注册 |
| **用例标识号：**DCZL001 |
| **参与者：**用户 |
| **简要说明：**  用户在进入单车租赁系统的首页时，选择注册操作。页面跳转到单车租赁系统的注册页面，在页面上输入账号的基本信息并提交，完成账号的注册。 |
| **前置条件:**  用户进入单车租赁系统的主页。 |
| **基本事件流：**   1. 用户进入单车租赁系统的主页面，选择注册操作。 2. 系统进入单车租赁系统的注册页面。 3. 用户在页面上输入基本信息，如账号、密码等，提交给系统。 4. 系统接收用户输入的信息，并在后台进行数据处理与逻辑判断。 5. 系统提示用户注册成功。 |
| **异常事件流：**  3a.用户输入的信息不完整   1. 注册失败，系统禁止用户提交本次请求。 2. 系统停留在当前页面，并提示用户补充完整信息。   2b.用户输入的信息不合法，无法通过校验   1. 注册失败，系统提示用户信息不合法，提示用户重新填写信息。 |
| **后置条件：**  系统提醒用户注册成功，并自动帮助用户登录系统，然后系统跳转到首页。 |

|  |
| --- |
| **用例名称：**登录 |
| **用例标识号：**DCZL002 |
| **参与者：**用户、管理员 |
| **简要说明：**  用户（或者管理员）进入单车租赁系统的首页，选择登录操作，页面跳转到系统的登陆页面，在页面的输入框中输入登录账号与登录密码并提交，从而完成登录。 |
| **前置条件:**  用户（或者管理员）进入单车租赁系统的登录页面，并已完成账号的注册。 |
| **基本事件流：**   1. 用户（或者管理员）进入单车租赁系统的登录页面。 2. 用户（或者管理员）在页面中输入账号、密码，并提交给系统。 3. 系统接收用户（或者管理员）输入的信息，并在后台进行账号与密码的验证。 4. 用户（或者管理员）登录成功，系统跳转至系统首页。 |
| **异常事件流：**  2a.用户（或者管理员）输入的账号密码的格式无法通过校验   1. 登录失败，系统禁止用户提交本次请求，并提示用户重新填写信息。   2b.用户（或者管理员）输入的信息不匹配   1. 登录失败，系统提示用户账号或密码错误，并提示用户重新填写信息。 |
| **后置条件：**  系统提醒用户（或者管理员）登录成功，并跳转到首页。 |

|  |
| --- |
| **用例名称：**定位当前位置 |
| **用例标识号：**DCZL003 |
| **参与者：**用户 |
| **简要说明：**  用户在系统的首页，选择定位操作。然后系统显示用户当前所在位置。 |
| **前置条件:**  用户成功登录系统，并进入系统首页。 |
| **基本事件流：**   1. 用户进入单车租赁系统的首页。 2. 用户选择定位操作。 3. 系统在地图上显示用户的当前位置信息。 |
| **异常事件流：**  无。 |
| **后置条件：**  系统显示用户当前所处的位置。 |

|  |
| --- |
| **用例名称：**查询站点信息 |
| **用例标识号：**DCZL004 |
| **参与者：**用户、管理员 |
| **简要说明：**  用户（或者管理员）在定位当前位置后，可以查询当前范围附近500米有多少单车租赁站点，并且可以查询某个站点的详细信息，包括该站点有多少单车可以使用等等。 |
| **前置条件:**  用户定位当前位置（参见用例DCZL003：定位当前位置）。 |
| **基本事件流：**   1. 用户（或者管理员）进入当前位置界面。 2. 系统显示当前位置附近500米有多少个站点。 3. 用户点击某个站点，选择【查询】操作。 4. 系统显示该站点的基本信息以及单车停靠情况。 |
| **异常事件流：**  1a.当前范围内没有站点   1. 系统提示用户重新定位。 |
| **后置条件：**  系统成功显示当前位置附近租车站点和单车停靠的信息。 |

|  |
| --- |
| **用例名称：**租车 |
| **用例标识号：**DCZL005 |
| **参与者：**用户 |
| **简要说明：**  用户在定位、并且选定某个站点的某辆单车后，选择页面上的【租赁】操作，此时便完成租车流程。 |
| **前置条件:**  用户查找到某个站点的单车使用情况（参见用例DCZL004：查询站点信息） |
| **基本事件流：**   1. 用户进入某个站点的单车使用情况界面。 2. 系统显示当前站点所有可供租赁的单车。 3. 用户选中页面上可使用的某一辆单车。 4. 用户选择租车。 5. 系统验证用户租赁信息通过，提示租车成功，开始计时。 |
| **异常事件流：**  5a.用户首次租车需缴纳押金   1. 系统提示租车失败，用户需缴纳押金后，方能成功租车。   5b.用户上次用车有欠费   1. 系统提示用户上次租车尚未还清使用金额，需先结算上次使用金额，才能进行新一轮租车。 |
| **后置条件：**  用户成功租借单车。 |

|  |
| --- |
| **用例名称：**归还单车 |
| **用例标识号：**DCZL006 |
| **参与者：**用户 |
| **简要说明：**  用户在完成租车后，选定某个站点，归还单车。 |
| **前置条件:**  用户已经成功租车（参见用例DCZL005：租车）。 |
| **基本事件流：**   1. 用户通过定位当前位置，选中要归还单车的站点，系统进入站点信息页面。 2. 用户选中站点信息页面上的空闲位置，选择【还车】操作。 3. 页面显示单车骑行时间，金额。 4. 用户完成付款，系统显示成功归还单车，之前选中的空闲位置显示为已被占。 |
| **异常事件流：**  2a.该站点没有空闲位置   1. 还车失败。系统提示用户重新选择站点进行还车。   4a.用户余额不足   1. 还车失败。系统提示用户余额不足，无法支付该次金额，建议充值金额。 |
| **后置条件：**  用户成功还车。 |

|  |
| --- |
| **用例名称：**金额充值 |
| **用例标识号：**DCZL007 |
| **参与者：**用户 |
| **简要说明：**  用户往自己账号内的金额进行充值，以确保用户有足够的金额支付在租车时产生的费用。 |
| **前置条件:**  用户登录成功。 |
| **基本事件流：**   1. 用户选择【充值】操作，系统进入金额充值界面。 2. 用户选中或输入所充值的金额，并提交。 3. 系统验证充值信息后，提示充值成功，页面显示用户的余额。 |
| **异常事件流：** 3a. 用户输入的金额错误  1)系统提示用户输入的金额不合法，并提示单次充值金额区间。 |
| **后置条件：**  用户充值成功。 |

|  |
| --- |
| **用例名称：**缴纳押金 |
| **用例标识号：**DCZL008 |
| **参与者：**用户 |
| **简要说明：**  用户在首次租车的时候，首先得先缴纳押金。 |
| **前置条件:**  用户已登录。 |
| **基本事件流：**   1. 用户在系统主界面，选择【押金缴纳】功能，系统进入押金缴纳页面。 2. 页面显示用户所需缴纳的押金的金额。 3. 用户选择【确定】操作。 4. 系统从用户的余额扣除相对应押金，提示缴纳押金成功，然后页面显示用户的余额。 |
| **异常事件流：**  4a.用户余额不足，无法缴纳押金   1. 缴纳押金失败。系统提示用户余额不足，无法支付该次金额，建议充值金额。 |
| **后置条件：**  用户成功缴纳押金。 |

|  |
| --- |
| **用例名称：**申请退还押金 |
| **用例标识号：**DCZL009 |
| **参与者：**用户 |
| **简要说明：**  用户在没有租车的时候，可以申请退还之前缴纳的押金，押金退还到用户的余额里。 |
| **前置条件:**  用户已登录系统。 |
| **基本事件流：**   1. 用户在系统主界面，选择【押金退还】功能，系统用户进入押金退还页面。 2. 页面显示用户所退还的押金的金额。 3. 用户确定退款操作。 4. 系统将要退还的押金退回支付的账户，并提示退还押金成功，页面显示用户的余额 |
| **异常事件流：**  2a.用户没有缴纳过押金   1. 退还押金失败。系统提示用户没有缴纳过押金。 |
| **后置条件：**  用户退还押金成功。 |

|  |
| --- |
| **用例名称：**修改个人信息 |
| **用例标识号：**DCZL010 |
| **参与者：**用户 |
| **简要说明：**  用户在个人信息页面，可以修改自己的信息。 |
| **前置条件:**  用户成功登录。 |
| **基本事件流：**   1. 用户进入个人信息页面。 2. 页面显示用户个人信息。 3. 用户选择修改个人信息。 4. 用户在可修改的属性的输入框中输入修改后的信息，并保存。 5. 系统验证修改后的信息，提示用户修改个人信息成功。 |
| **异常事件流：**  4a. 用户输入的信息不合法，无法通过校验   1. 修改失败，系统提示用户信息不合法，无法通过校验，并提示用户重新填写信息。 |
| **后置条件：**  用户成功修改了个人信息。 |

|  |
| --- |
| **用例名称：**修改密码 |
| **用例标识号：**DCZL011 |
| **参与者：**用户 |
| **简要说明：**  用户在个人信息页面，通过选择修改密码，从而进行密码的修改。 |
| **前置条件:**  用户成功登录。 |
| **基本事件流：**   1. 用户进入个人信息页面,选择修改密码。 2. 系统跳转修改密码页面。 3. 用户在输入框中输入旧密码与新密码两次，并提交系统。 4. 系统提示用户修改密码成功。 |
| **异常事件流：**  4a. 用户输入的旧密码错误，无法通过校验   1. 修改密码失败，系统提示旧密码错误，重新输入。   4b. 用户输入的新密码不符合要求，无法通过校验   1. 修改密码失败，系统提示新密码不符合要求，重新输入。次信息 种提示？不可能一种情况 |
| **后置条件：**  用户成功修改密码，系统要求重新登录。 |

|  |
| --- |
| **用例名称：**重置密码 |
| **用例标识号：**DCZL012 |
| **参与者：**用户 |
| **简要说明：**  用户如果忘记了密码，可以在登录页面选择忘记密码。此时系统会发邮件到用户注册的邮箱账号。用户此时通过点击邮箱上的链接，去到重置密码页面，完成密码的重置。 |
| **前置条件:**  用户成功注册。 |
| **基本事件流：**   1. 用户进入登录页面,并选择重置密码。 2. 系统发送邮件到用户注册的邮箱账号。 3. 用户收到系统发送的邮件，并点击里面的链接去到重置密码页面。 4. 用户重置密码。 |
| **异常事件流：**  2a. 系统发送的邮件被用户邮箱拦截   1. 系统重新发送邮件。   4a. 用户输入的新密码不符合要求，无法通过校验   1. 重置密码失败，系统提示新密码不符合要求，重新输入。 |
| **后置条件：**  用户成功重置密码，系统要求重新登录。 |

|  |
| --- |
| **用例名称：**管理站点信息 |
| **用例标识号：**DCZL013 |
| **参与者：**管理员 |
| **简要说明：**  管理员可以对单车站点进行管理，如：修改站点信息、修改某个站点的单车停放位置数等。 |
| **前置条件:**  管理员成功登录。 |
| **基本事件流：**   1. 管理员进入后台管理的站点管理页面，系统显示所有站点信息。 2. 管理员选中页面上显示的某个站点，并选择修改该站点信息。 3. 系统显示站点信息编辑界面。 4. 管理员更改要修改的站点信息，并保存。 5. 系统验证修改后的信息通过后，保存相应的信息，并提示修改站点信息成功。 |
| **异常事件流：**  1a. 系统内没有站点信息   1. 系统提示管理员无站点信息。   4a. 修改后的信息不符合格式要求  1)系统提示输入的信息不符合格式要求，请重新输入。 5a. 修改后的站点名称重复  1)系统提示输入的站点名称重复，请重新输入。 |
| **后置条件：**  站点信息被成功修改，系统显示更新后的站点信息。 |

|  |
| --- |
| **用例名称：**新增站点 |
| **用例标识号：**DCZL014 |
| **参与者：**管理员 |
| **简要说明：**  管理员可以在后台管理页面，新增单车停放站点。 |
| **前置条件:**  管理员成功登录。 |
| **基本事件流：**   1. 管理员进入后台管理的站点管理页面，系统显示所有站点信息。 2. 管理员选择新增站点操作，系统进入相应页面。 3. 管理员在页面上的输入新增站点的信息，然后保存。 4. 系统提示管理员成功新增站点，跳转到管理站点页面并显示所有站点信息，新增的站点信息显示在其中。 |
| **异常事件流：**  2a. 管理员输入的信息不合法，无法通过校验   1. 系统提示管理员输入信息不合法，重新输入。 |
| **后置条件：**  无。 |

|  |
| --- |
| **用例名称：**新增单车 |
| **用例标识号：**DCZL015 |
| **参与者：**管理员 |
| **简要说明：**  管理员可以在后台管理页面，新增单车。 |
| **前置条件:**  管理员成功登录。 |
| **基本事件流：**   1. 管理员进入后台管理页面的新增单车页面，系统显示新增单车界面。 2. 管理员在页面的输入框输入新增单车的信息，指定单车投放的站点，然后提交给系统。 3. 系统验证信息，提示用户新增单车成功。 4. 系统跳转至所有单车信息界面，新增的单车信息显示其中。 |
| **异常事件流：**  2a 站点无空闲位置可供放置单车   1. 系统提示该站点已无空闲位置可供停靠单车，提示重新选择站点。   3a 管理员输入单车编号重复   1. 系统提示该单车编号已存在，提示重新输入。 |
| **后置条件：**  无。 |

|  |
| --- |
| **用例名称：**管理单车 |
| **用例标识号：**DCZL016 |
| **参与者：**管理员 |
| **简要说明：**  管理员可以对单车进行管理，如：修改单车信息。 |
| **前置条件:**  管理员成功登录。 |
| **基本事件流：**   1. 管理员进入后台管理的单车管理页面，系统显示所有单车信息。 2. 管理员选中页面上显示的某辆单车，系统显示单车信息编辑界面。 3. 管理员在页面输入框中输入要更改的单车信息，并提交给系统。 |
| **异常事件流：**  3a. 单车处于使用状态无法更改   1. 系统提示管理员该单车正处于使用状态，无法更改其信息。 |
| **后置条件：**  无。 |

|  |
| --- |
| **用例名称：**删除单车 |
| **用例标识号：**DCZL017 |
| **参与者：**管理员 |
| **简要说明：**  管理员可以对单车进行删除操作。 |
| **前置条件:**  管理员成功登录。 |
| **基本事件流：**   1. 管理员进入后台管理的单车管理页面，系统显示单车列表。 2. 管理员根据站点或者单车编号等搜索条件选中页面上显示的某辆单车，并选择删除。 3. 系统提示确认删除操作，用户确认删除。 4. 管理员成功删除单车。 |
| **异常事件流：**  2a. 单车处于使用状态无法更改   1. 系统提示管理员该单车正处于使用状态，无法删除该辆单车。   3a. 用户取消删除操作  1) 系统取消删除操作，跳转回单车管理界面。 |
| **后置条件：**  无。 |

|  |
| --- |
| **用例名称：**更新单车状态 |
| **用例标识号：**DCZL018 |
| **参与者：**管理员 |
| **简要说明：**  管理员可以对单车的状态进行维护。例如将损坏的单车拿去修理。 |
| **前置条件:**  管理员成功登录。 |
| **基本事件流：**   1. 管理员进入后台管理的单车状态管理页面。 2. 管理员选中页面上显示状态为“损坏”的单车，并更改其状态。 3. 管理员成功更改单车状态。 |
| **异常事件流：**  无。 |
| **后置条件：**  无。 |

|  |
| --- |
| **用例名称：**投放单车 |
| **用例标识号：**DCZL019 |
| **参与者：**管理员 |
| **简要说明：**  管理员对修理好正处于闲置的单车，将其投放到单车站点。 |
| **前置条件:**  管理员成功登录。 |
| **基本事件流：**   1. 管理员进入后台管理的单车状态管理页面,并搜索状态为闲置的单车，系统显示符合条件的单车信息。 2. 管理员选中页面上显示状态为“闲置”的单车，并选择投放。 3. 系统跳转站点页面，显示所有站点信息。 4. 管理员选择某个站点，并进行单车投放站点操作。 5. 系统提示投放成功，选中的单车状态变更为空闲。 |
| **异常事件流：**  4a. 该站点没有空闲位置可以投放   1. 系统提示该站点没有空闲位置，提示重新选择站点 |
| **后置条件：**  原本闲置的单车成功投放到指定站点，单程状态修订为“使用中”。 |

|  |
| --- |
| **用例名称：**统计单车数据 |
| **用例标识号：**DCZL020 |
| **参与者：**管理员 |
| **简要说明：**  管理员统计当前时间或者某段时间内，有多少单车正在使用、多少单车正在修理等。 |
| **前置条件:**  管理员成功登录。 |
| **基本事件流：**   1. 管理员进入后台管理的单车数据管理页面。 2. 管理员在页面上输入查询条件（例如：站点名称、站点位置、单车状态等），然后系统根据条件搜索当前单车数据。 3. 页面显示当前时间段内的单车数据。 |
| **异常事件流：**  3a. 系统中没有符合条件的单车数据。   1. 系统在页面上提示没有符合条件的单车数据。 |
| **后置条件：**  系统成功显示符合统计条件的单车统计数据。 |

2.4领域模型

基于上一小节中对共享单车租赁系统的用例分析，我们可以构建出相对应的领域模型（如图2.4），来更好的分析本系统。通过对上述用例的详细描述，我们可以得知系统有用户、管理员、单车、站点、租赁情况、余额等领域对象。这些领域对象的具体说明如下：

* 用户：使用本系统的所有用户。用户可以在本系统上进行单车的查询，租赁-归还等一系列操作。
* 管理员：负责单车以及站点的维护。
* 单车：用户进行查询租赁和管理员进行维护的对象。
* 站点：管理员进行维护的对象。
* 租赁情况：负责记录用户的租赁信息。
* 余额：用户用来缴纳押金以及支付单车租赁而产生的金额。
* 押金：用户进行租赁操作的前提是缴纳押金。

从以下的领域模型图可以看出，管理员将单车投放到站点后（即管理员发布单车信息以及站点信息），用户在本系统查询相对应站点上的单车信息，然后根据押金情况进行租赁操作。然后当用户选择站点归还单车时，便使用余额进行支付操作。该次租赁操作便记录在租赁情况里。

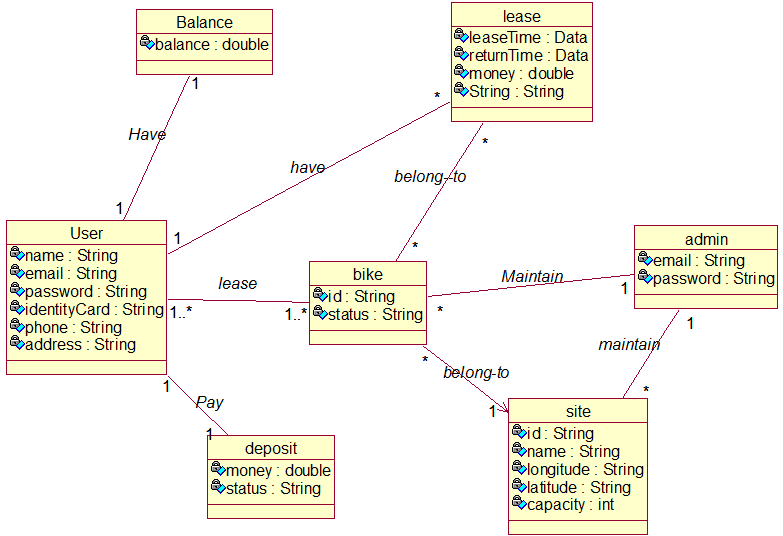


图2.4 领域模型

2.5非功能需求

2.5.1性能和容量

系统要求前台用户在发起请求后，系统响应的时间须在3秒内。其中一些复杂的操作响应时间可以在3秒以上。具体如下：

1. 租车受理响应时间须在6秒内。
2. 还车受理响应时间须在8秒内。
3. 系统登录响应时间须在5秒内。

2.5.2安全性

系统在每一步关键操作，与其相关联的上下层次的操作，都须记录在日志文件。同时，系统里面涉及金钱方面的操作，应具备安全监督、防病毒等功能；在数据方面须有严格的校验、核对及纠错功能；以及对系统可能发生的故障有相对应的故障解决方案。

2.5.3易用性

界面必须简洁、大方、清晰、友好、柔和，操作简单方便，使用户能轻快上手并使用本系统。在前端服务页面和后台管理页面，应以通俗易懂的操作栏向用户提供可视化操作。

2.5.4可移植性

系统应保证在更换操作系统时，可以做到平滑过渡。

2.5.5可扩展性

系统应保证在其基础版本上，可以往上迭代开发新功能的同时，不影响现版本功能的使用。

2.5.6可靠性

由于系统的可靠性依赖于操作系统、网络、数据库系统，所以系统采用的设备以及主要技术应该是成熟且可靠的。

# 3概要设计

3.1逻辑视图

共享单车租赁系统采取了MVC系统架构模式进行开发，本系统的逻辑架构共由4部分组成，分别是：用于展示前台页面的界面层（View），用于处理控制转发和业务逻辑处理的业务逻辑控制层（Controller），用于数据库交互的数据库访问层（Dao），以及业务模型层（Dto）。如图3.1所示。

界面层（View）：用于向系统的使用者展示其与系统进行交互后的数据显示界面，并且向系统的使用者提供可视化的系统功能操作。

业务逻辑控制层（Controller）：用于处理界面层与数据库访问层交互时的业务逻辑，并控制页面的跳转。包含了系统参与者的登录注册、单车的查询与维护、站点的查询与维护、单车的租赁与归还、余额的充值以及缴纳押金等用例的逻辑处理类。

数据库访问层（Dao）：用于处理系统向数据库发送或接收数据，是系统与数据库之间数据交互的桥梁。

业务模型层（Dto）：用于展示系统中的实体类，包含单车、站点、用户、租赁情况等实体类。

在本系统中，界面层（View）只用来在前台向系统使用者展示页面，而具体的业务逻辑处理以及控制页面的跳转和转发便在业务逻辑控制层（Controller）中实现。而一个完整的流程，则是界面层（View）发起功能请求到业务逻辑控制层（Controller），然后业务逻辑控制层（Controller）进行业务逻辑判断与处理后，向数据库访问层（Dto）发起请求，数据库访问层（Dto）再跟数据库交互后返回给业务逻辑控制层（Controller），然后业务逻辑控制层（Controller）再根据逻辑处理来决定具体转发到哪一个页面用于前台向系统使用者展示。

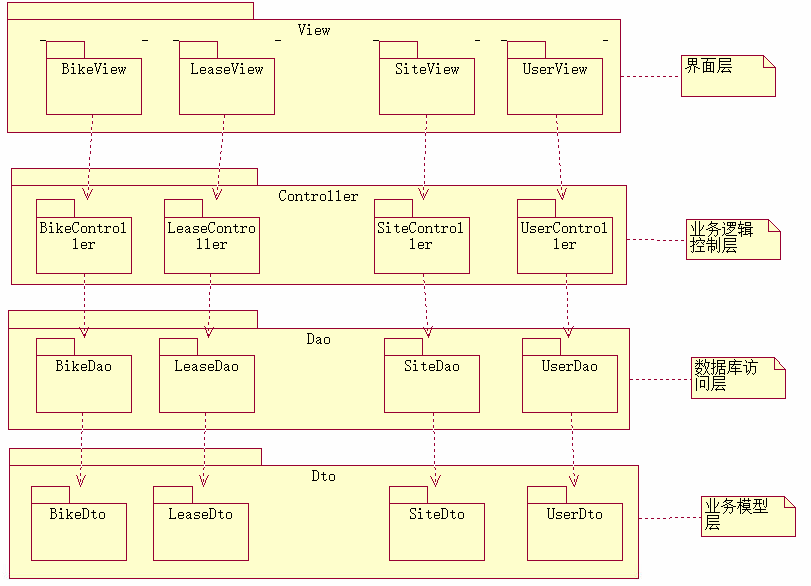


图 3.1 逻辑视图

3.2开发视图

根据上一小节中的逻辑结构设计，我们将本系统中各个层次进行划分后，可以得出如下的系统层次分析：

首先，用户在浏览器（View）访问系统，系统在接收用户的请求后，封装好相对应信息后，将该次请求发送到Controller（业务逻辑控制层），在Controller层中，系统进行业务逻辑的处理与判断，同时将相关信息数据封装后，发送到Dao（数据库交互层），在这一层中，系统将进行与数据库的交互操作，即与DB进行数据访问，同时将该次请求所封装的信息，持久化为Model（业务领域模型）对象。在与DB数据交互完成后，返回数据到 Controller层，Controller封装好相对应提示和数据后，再控制转发相对应的页面显示，从而将数据显示在相对应的页面上。

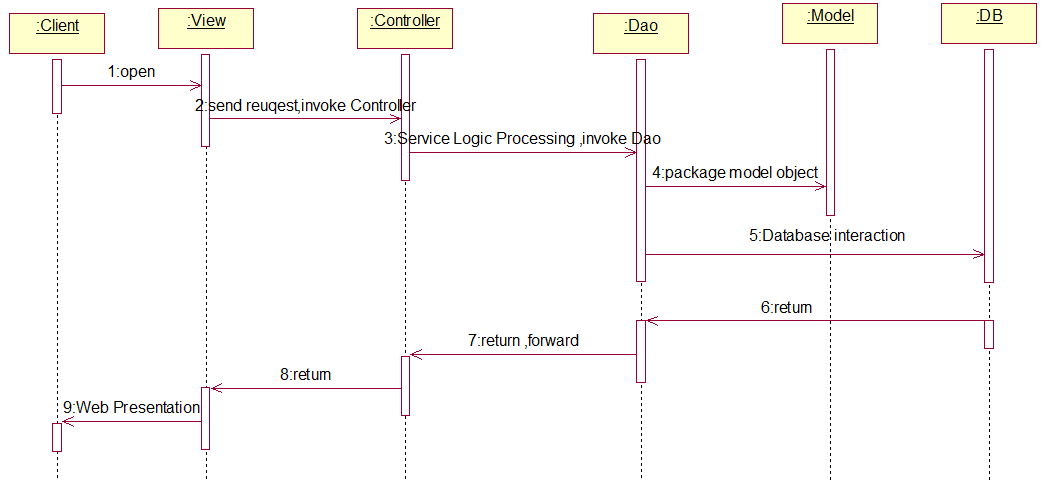
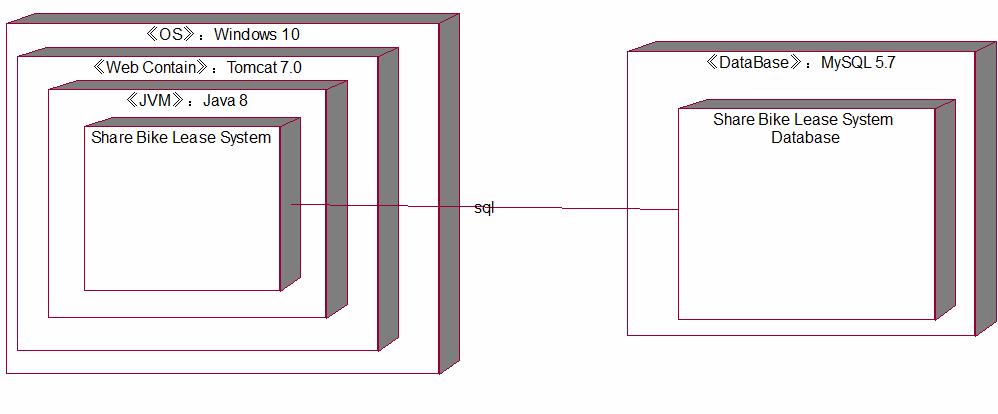


图3.2 开发视图

3.3物理视图

本系统在操作系统为Windows 10的基础上，采用Tomcat 7.0的Web服务器，搭配Java 8，以及MySQL5.7，进行系统的部署。如图3.3所示。

图3.3 部署图

# 4数据库设计

4.1E-R图

在本系统中，各个业务实体所对应的数据库表都相互依赖。具体如图4.1的E-R图所示。

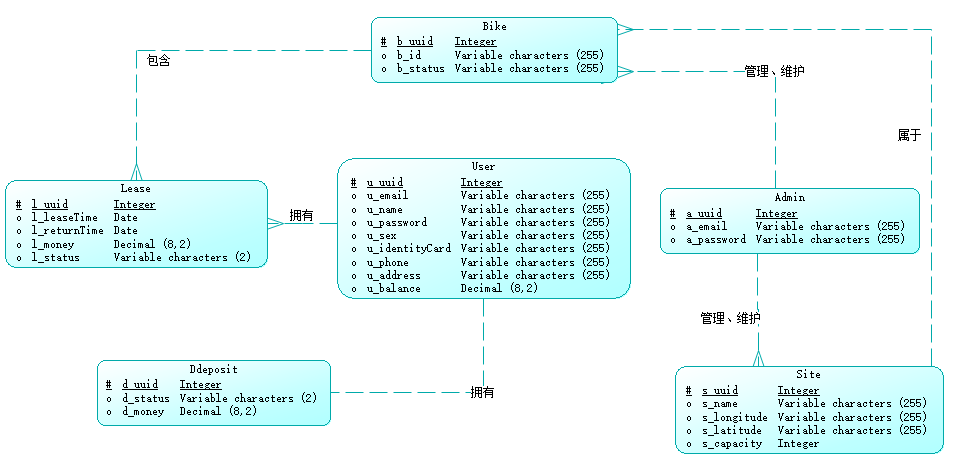


图4.1 E-R图

4.2数据库表设计

根据上一小节的E-R图分析，我们可以设计出如表4.1的数据库表结构。

表4.1 数据库总表

|  |  |  |
| --- | --- | --- |
| **表名** | **中文描述表名** | **功能说明** |
| Admin | 管理员 | 存放管理员的登录信息 |
| User | 用户 | 存放用户的信息 |
| Deposit | 押金 | 存放用户缴纳押金的相关信息 |
| Site | 站点 | 存放站点的相关信息 |
| Bike | 单车 | 存放单车的相关信息 |
| Lease | 租赁信息 | 存放用户租赁情况的相关信息 |

4.3数据字典

根据上两节的E-R图和数据库表设计，我们可以构建出相对应的数据字典。如下所示。表4.2 Admin表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | Admin | | | |
| 名称 | 数据类型 | 主要的 | 外来键 | 注释 |
| a\_uuid | int | TRUE | FALSE | 自增列 |
| a\_email | varchar(255) | FALSE | FALSE | 管理员账号 |
| a\_password | varchar(255) | FALSE | FALSE | 密码 |

表4.3 User表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | User | | | |
| 名称 | 数据类型 | 主要的 | 外来键 | 注释 |
| u\_uuid | int | TRUE | FALSE | 自增列 |
| u\_email | varchar(255) | FALSE | FALSE | 用户账号 |
| u\_name | varchar(255) | FALSE | FALSE | 姓名 |
| u\_password | varchar(255) | FALSE | FALSE | 密码 |
| u\_sex | varchar(255) | FALSE | FALSE | 性别 |
| u\_identityCard | varchar(255) | FALSE | FALSE | 身份证号 |
| u\_phone | varchar(255) | FALSE | FALSE | 电话 |
| u\_address | varchar(255) | FALSE | FALSE | 地址 |
| u\_balance | decimal(8,2) | FALSE | FALSE | 余额 |

表4.4 Deposit表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | Deposit | | | |
| 名称 | 数据类型 | 主要的 | 外来键 | 注释 |
| d\_uuid | int | TRUE | FALSE | 自增列 |
| d\_status | varchar(2) | FALSE | FALSE | 押金缴纳状态 |
| d\_money | decimal(8,2) | FALSE | FALSE | 押金数目 |
| u\_uuid | int | FALSE | TRUE | 用户id |

表4.5 Site表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | Site | | | |
| 名称 | 数据类型 | 主要的 | 外来键 | 注释 |
| s\_uuid | int | TRUE | FALSE | 自增列 |
| s\_name | varchar(255) | FALSE | FALSE | 站点名字 |
| s\_longitude | varchar(255) | FALSE | FALSE | 站点经度 |
| s\_latitude | varchar(255) | FALSE | FALSE | 站点纬度 |
| s\_capacity | int | FALSE | FALSE | 站点容量 |

表4.2.6 Bike表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | Bike | | | |
| 名称 | 数据类型 | 主要的 | 外来键 | 注释 |
| b\_uuid | int | TRUE | FALSE | 自增列 |
| b\_id | varchar(255) | FALSE | FALSE | 单车编号 |
| b\_status | varchar(255) | FALSE | FALSE | 单车状态 |
| s\_uuid | int | FALSE | TRUE | 停靠站点 |
| b\_uuid | int | TRUE | FALSE | 自增列 |
| b\_id | varchar(255) | FALSE | FALSE | 单车编号 |
| b\_status | varchar(255) | FALSE | FALSE | 单车状态 |
| s\_uuid | int | FALSE | TRUE | 停靠站点 |

表4.2.7 Lease表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | Lease | | | |
| 名称 | 数据类型 | 主要的 | 外来键 | 注释 |
| l\_uuid | int | TRUE | FALSE | 自增列 |
| l\_leaseTime | date | FALSE | FALSE | 租车时间 |
| l\_returnTime | date | FALSE | FALSE | 还车时间 |
| l\_money | decimal(8,2) | FALSE | FALSE | 消费金额 |
| l\_status | varchar(2) | FALSE | FALSE | 状态：单车是否归还 |
| u\_uuid | int | FALSE | TRUE | 用户id |
| b\_uuid | int | FALSE | TRUE | 单车id |

# 5详细设计

5.1登录

该用例的时序图如图5.1所示，类图如图5.2所示。该用例的基本事件流如下：

1. 用户打开浏览器访问本系统，系统在浏览器上显示登录页面。
2. 用户在登录页面的输入框中分别输入邮箱账号、密码，然后提交。
3. 系统封装好用户的输入信息，将本次请求发送到LoginController进行逻辑处理。
4. LoginController进行逻辑处理后，将数据发送到LoginDao，从而进行数据库的数据交互。
5. LoginDao与DB进行数据交互后，将返回数据发送给LoginController。
6. LoginController封装好相对应数据后，再根据逻辑处理，控制转发相对应页面，进而将数据展示在页面上。

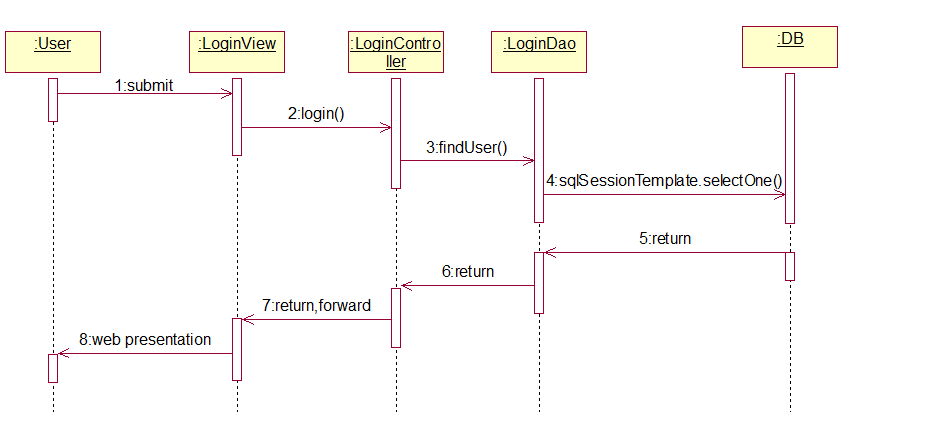
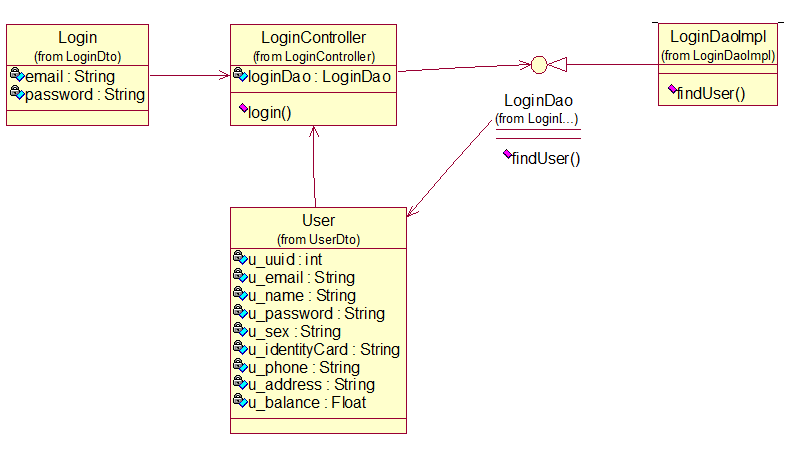
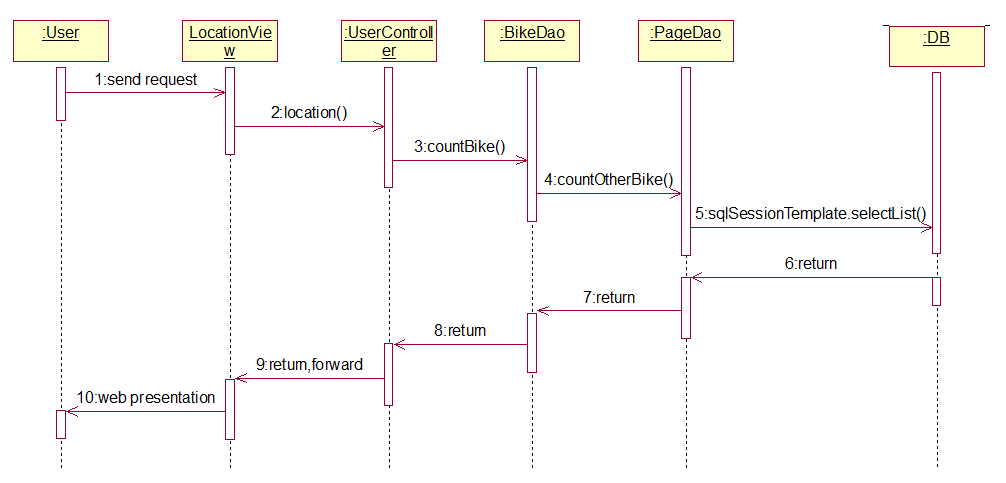
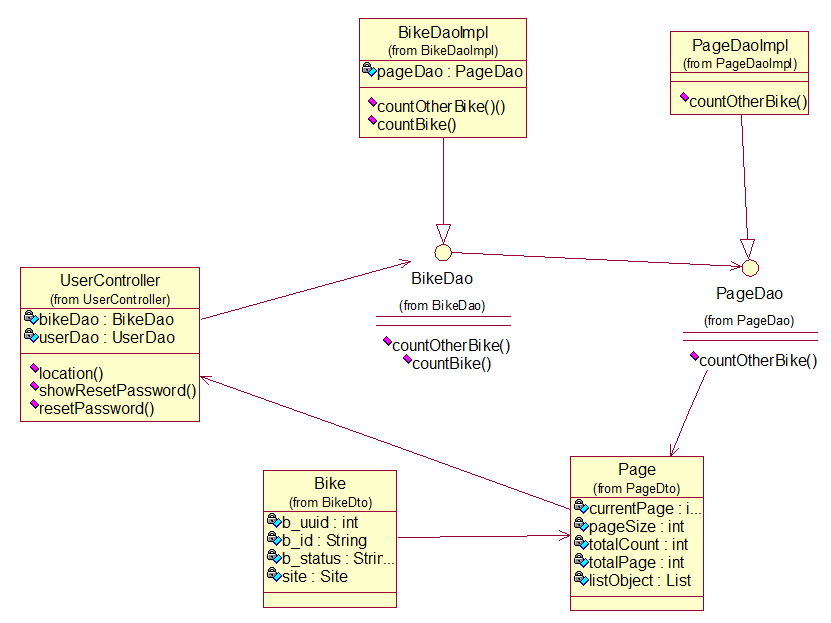
图5.1 登录用例的时序图

图5.2 登录用例的类图

5.2定位

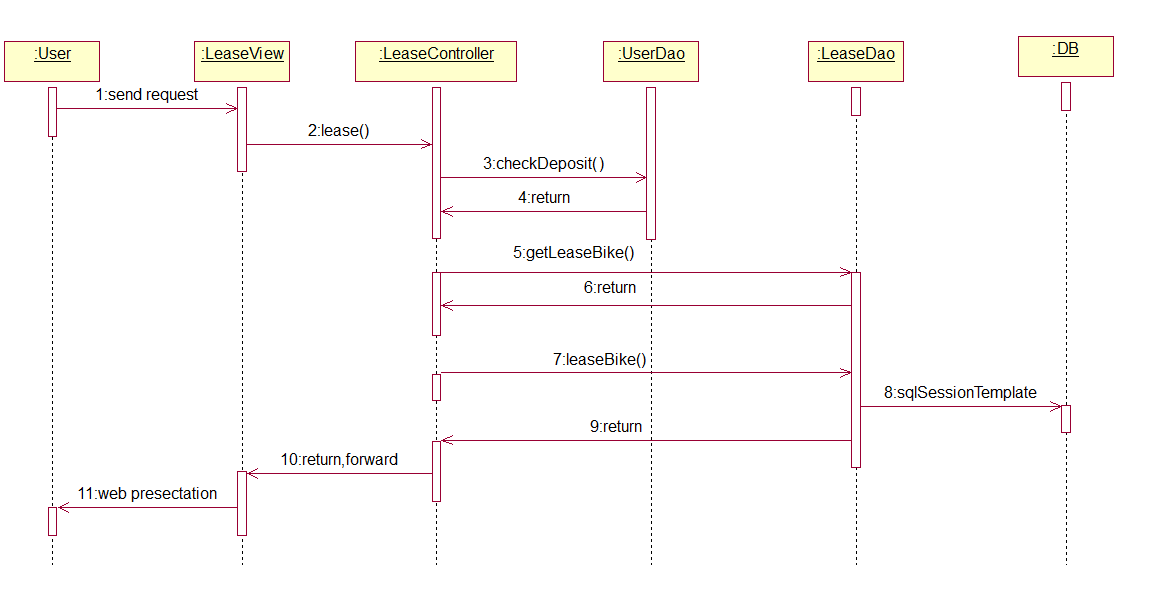
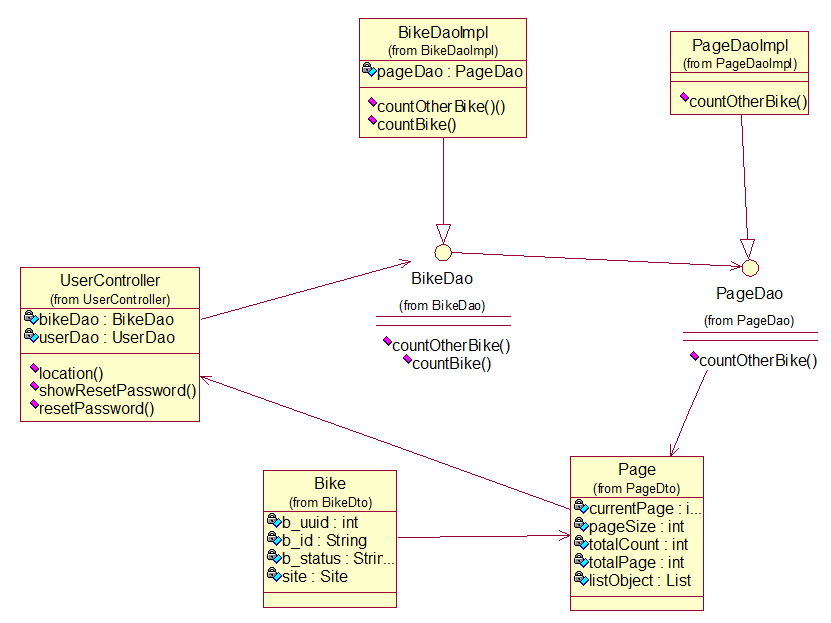
该用例的时序图如图5.3所示，类图如图5.4所示。该用例的基本事件流如下：

1. 用户在系统主页的左边操作栏上，选择【定位】按钮。
2. 系统接收用户该次请求，将请求发送到UserController。
3. UserController进行逻辑处理后，将地点信息等相关数据发送到BikeDao。
4. BikeDao接收数据后，将数据转发给PageDao进行分页数据的查询。
5. PageDao接收数据后，与数据库进行数据的交互访问，并返回结果集给PageDao。
6. PageDao再将数据封装到Page实体上，然后再一层层返回给UserController，最后UserController再封装相对应数据以及提示，控制转发到相对应页面，从而展示数据。图5.3 定位用例的时序图图5.4 定位用例的类图

5.3租车

该用例的时序图如5.5X所示，类图如图5.6所示。该用例的基本事件流如下：

1. 用户在定位后的单车显示页上，选择【租赁】按钮。
2. 系统接收用户该次请求，将请求发送到LeaseController。
3. LeaseController将数据发送到UserDao与LeaseDao，分别进行逻辑处理，并取得返回数据。
4. LeaseController取得返回数据后，再调用LeaseDao的lease()方法与数据库进行数据交互并取得返回结果集，从而进行租赁操作。
5. LeaseDao将返回的结果集封装后，返回给LeaseController。
6. LeaseController封装相对应数据以及提示，控制转发到相对应页面，从而展示数据。

图5.5 租车用例的时序图图5.6 租车用例的类图

5.4还车

该用例的时序图如5.7所示，类图如图5.8所示。该用例的基本事件流如下：

1. 用户在系统主页的左边操作栏上，选择【订单】按钮，从而进入订单页面。
2. 用户在订单页面中，选择【归还】按钮。
3. 系统接收用户请求，将该次请求数据发送到LeaseController。
4. LeaseController首先通过UserDao判断用户的余额是否足够支付该次操作所产生的费用。在通关验证后，再调用LeaseDao与数据库进行交互并取得返回数据后，从而完成归还单车这一操作。
5. LeaseDao将返回的结果集数据发送给LeaseController， LeaseController封装相对应数据以及提示，控制转发到相对应页面，从而展示数据。

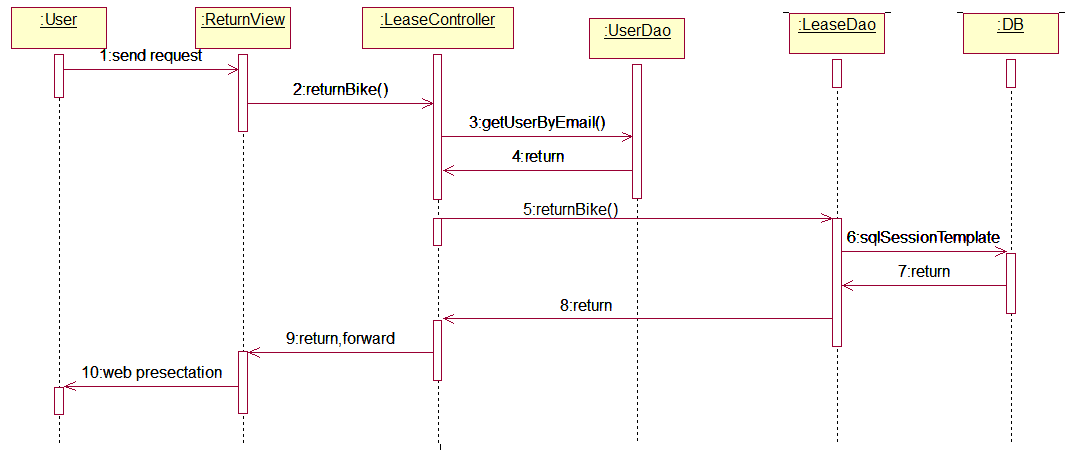


图5.7 还车用例的时序图

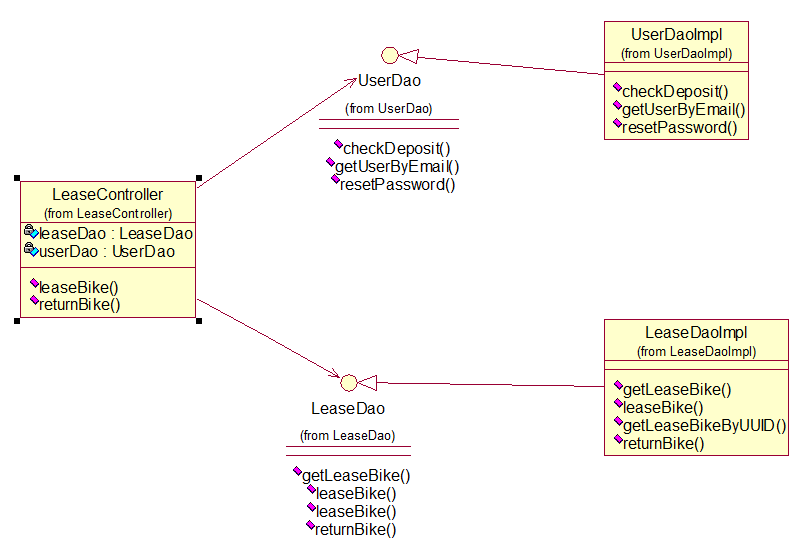
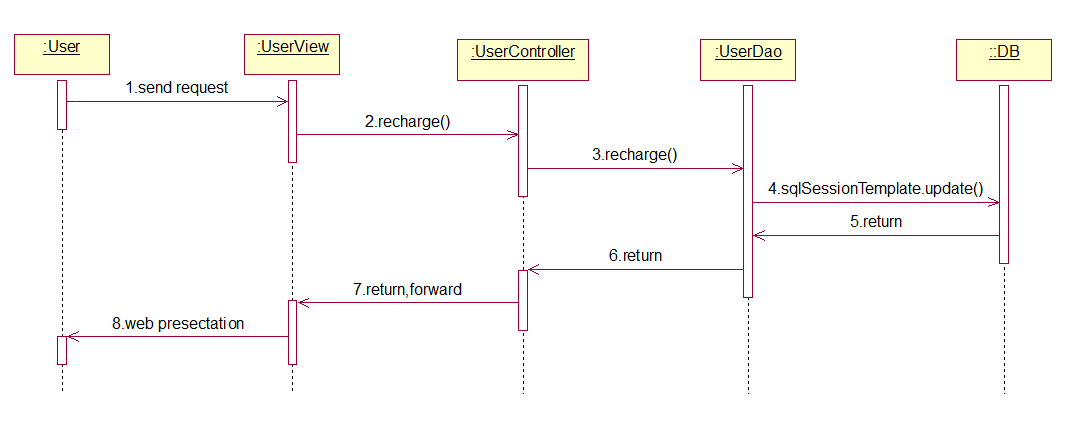
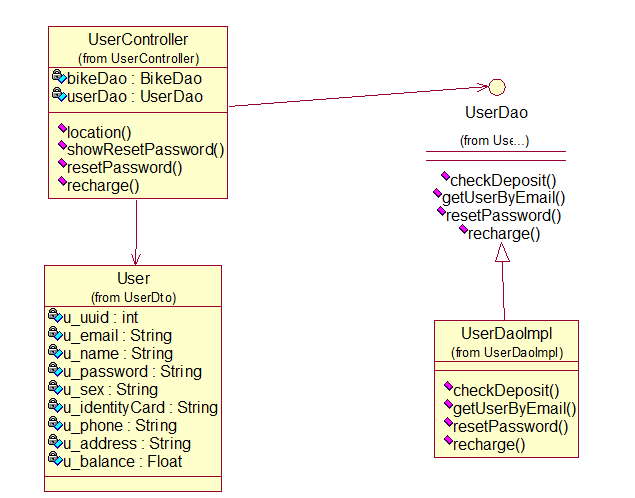


图5.8还车用例的类图

5.5金额充值

该用例的时序图如图5.9所示，类图如图5.10所示。该用例的基本事件流如下：

1. 用户在系统主页的左边操作栏上，选择【金额充值】按钮，从而进入充值页面。
2. 用户在该页面中输入或选择充值金额数目并提交给系统。
3. 系统接收用户请求，将该次请求数据发送到UserController。
4. UserController首先通过UserDao与数据库进行数据交互、取得返回结果集数据并转发给UserController后，从而完成金额的充值。
5. UserController在取得返回结果集数据后，封装相对应提示，控制转发到相对应页面，从而展示数据。

图5.9 金额充值用例的时序图图5.10 金额充值用例的类图

5.6重置密码

该用例的时序图如图5.11所示，类图如图5.12所示。该用例的基本事件流如下：

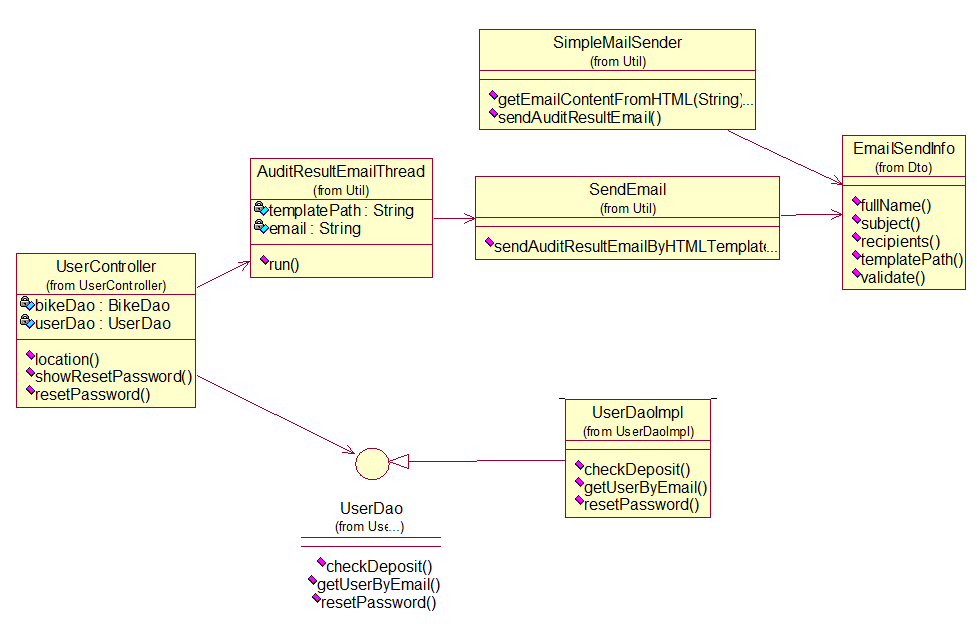
1. 用户在系统登录页面上，选择【重置密码】按钮，从而进入重置密码页面。
2. 用户在该页面中输入邮箱和身份证号并提交给系统。
3. 系统接收用户请求，将该次请求数据发送到UserController。
4. UserController首先通过UserDao判断用户的邮箱和身份证号是否匹配。验证通过后，UserController调用AuditResultEmailThread发送邮件到用户输入的邮箱账号。
5. 用户在接收到邮件后，打开邮件中的地址，在页面上输入新的密码并提交。
6. 系统接收用户该次请求后，将信息发送到UserController。
7. UserController通过UserDao与数据库进行交互，从而修改数据库中的密码并取得返回数据，进而完成重置密码操作。
8. UserController取得返回数据后，封装相对应提示，控制转发到相对应页面，从而展示数据。

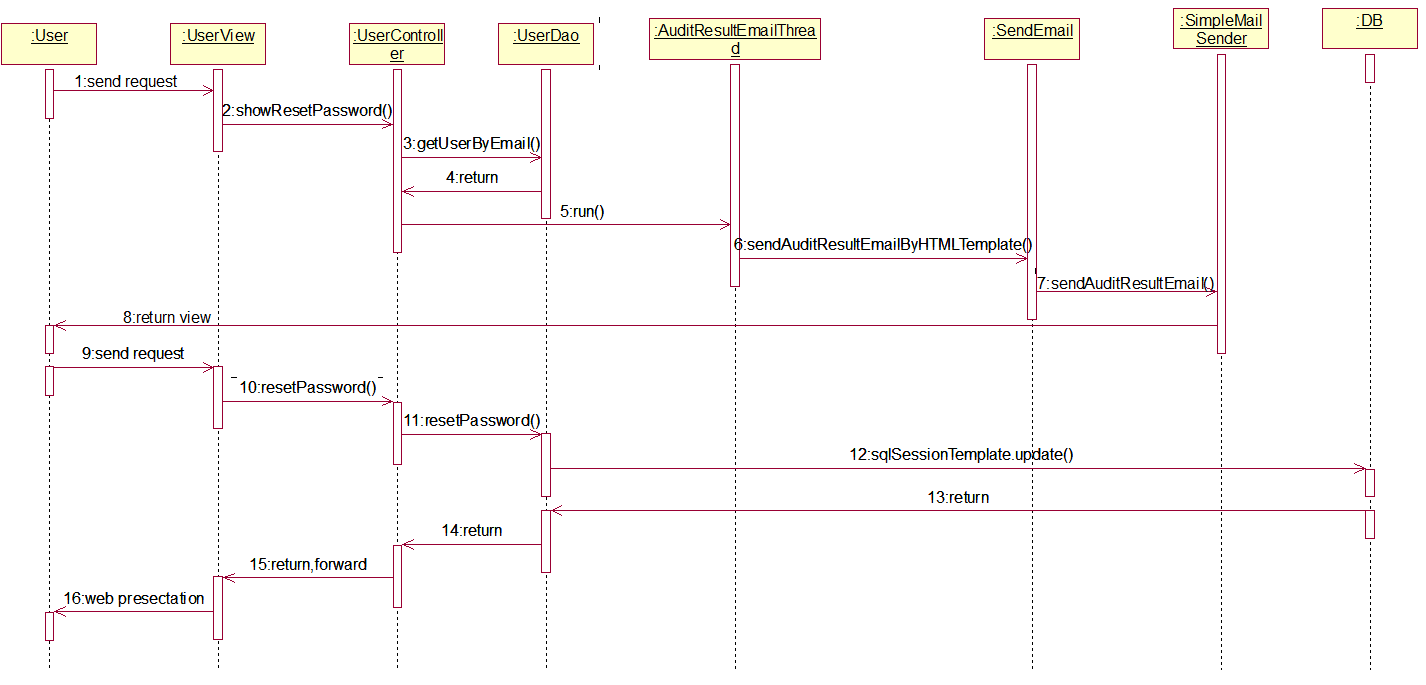
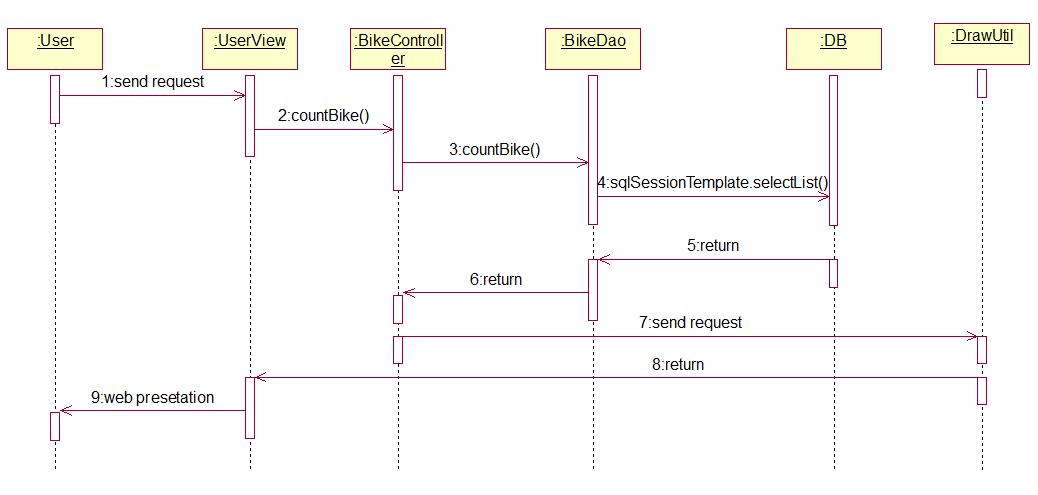
图5.11 重置密码用例的时序图 

图5.12 重置密码用例的类图

5.7单车数据统计

该用例的时序图如图5.13所示，类图如图5.14所示。该用例的基本事件流如下：

1. 管理员在后台管理页面上，选择【单车数据统计】按钮。
2. 系统接收用户请求，将该次请求发送到BikeController。
3. BikeController通过BikeDao与数据库交互进行数据访问并取得返回结果集后，调用DrawUtil进行饼状图的展示，并控制页面转化，将数据显示在前台。

图5.13 单车数据统计用例的时序图

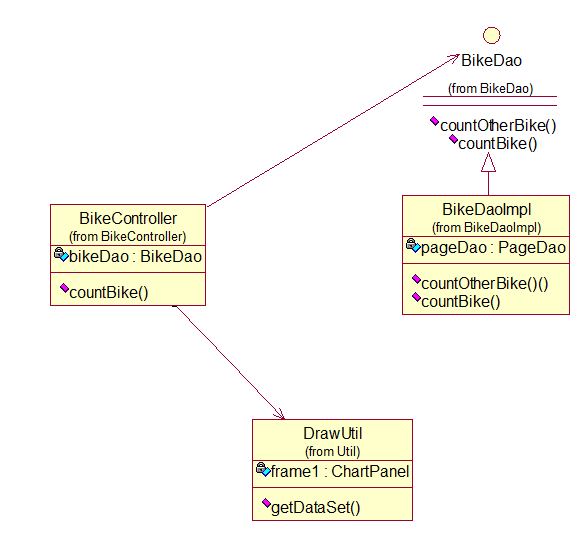
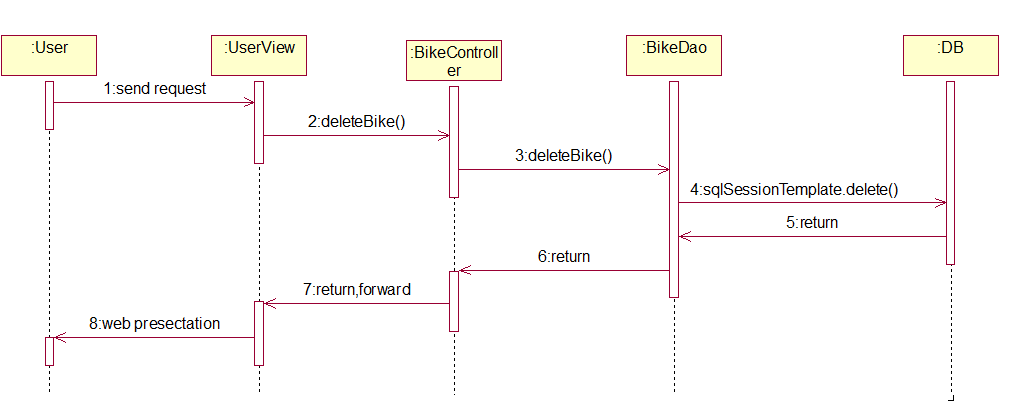


图5.14 单车数据统计用例的时序图

5.8删除单车

该用例的时序图如图5.15所示，类图如图5.16所示。该用例的基本事件流如下：

1. 管理员在后台的单车管理页上，选择某辆单车（或者根据搜索条件搜索）后的【删除】按钮。
2. 系统接收用户该次请求，将请求发送到BikeController。
3. BikeController进行逻辑处理后，将单车信息等相关数据发送到BikeDao。
4. BikeDao接收数据后，与数据库进行数据交互并返回结果集，从而完成删除操作。
5. BikeDao接收返回的数据后，将结果集转发给BikeController。
6. BikeController接收返回数据后，再封装相对应数据以及提示，控制转发到相对应页面，从而展示数据。

图5.15 删除单车用例的时序图

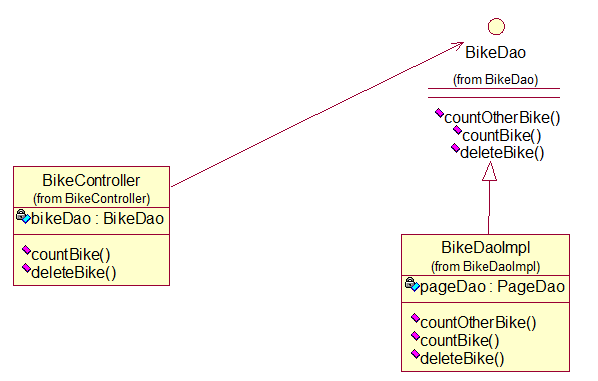


图5.16 删除单车用例的类图

5.9修理单车

该用例的时序图如图5.17所示，类图如图5.18所示。该用例的基本事件流如下：

1. 管理员在后台的单车管理页上，选择状态为损坏的单车，并选择【修理】操作。
2. 系统接收用户该次请求，将请求发送到BikeController。
3. BikeController进行逻辑处理后，将单车ID等相关数据发送到BikeDao。
4. BikeDao接收数据后，与数据库进行数据交互并返回结果集，从而完成修理操作。
5. BikeDao接收返回的数据后，将结果集转发给BikeController。
6. BikeController接收返回数据后，再封装相对应数据以及提示，控制转发到相对应页面，从而展示数据。

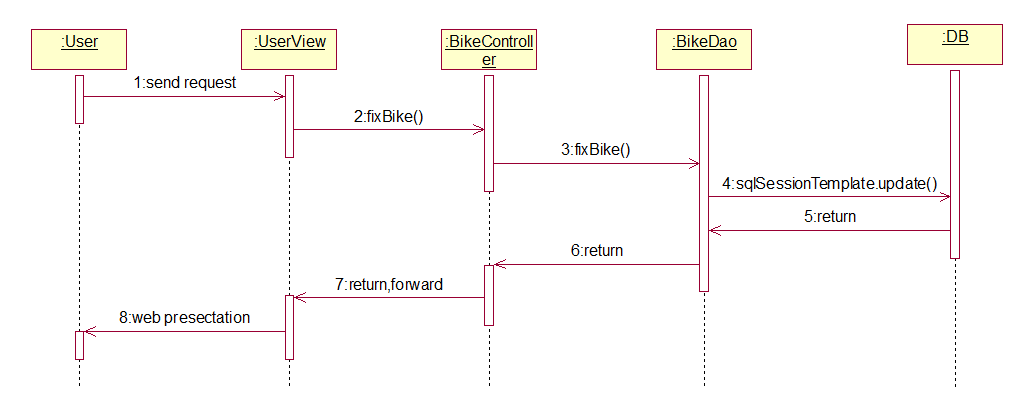


图5.17 修理单车用例的时序图

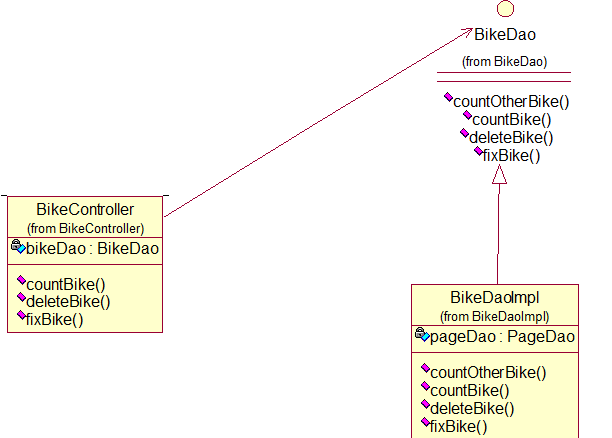
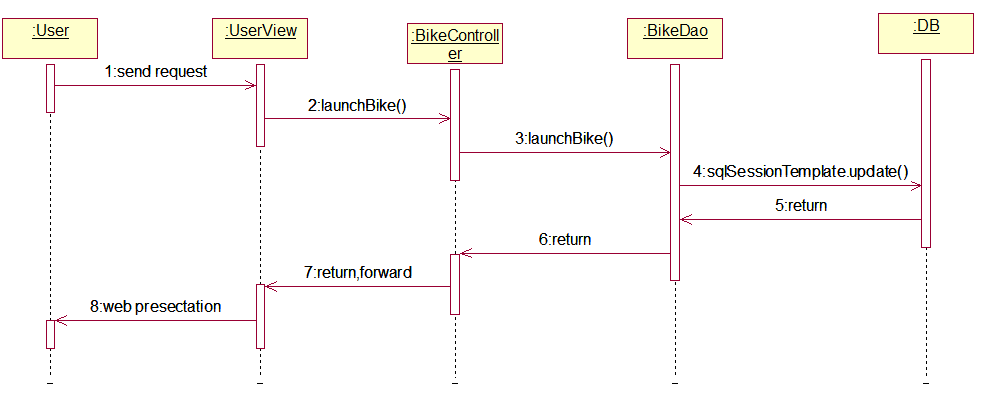


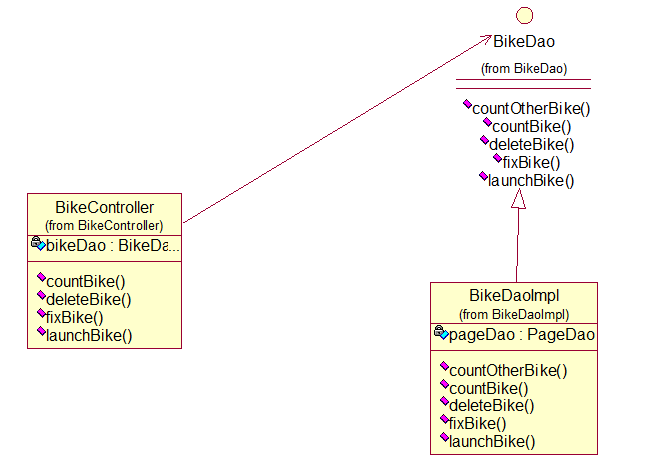
图5.18 修理单车用例的类图

5.10投放单车

该用例的时序图如图5.19所示，类图如图5.20所示。该用例的基本事件流如下：

1. 管理员在后台的单车管理页上，选择状态为闲置的单车，并选择【投放】操作。
2. 系统接收用户该次请求，将请求发送到BikeController。
3. BikeController进行逻辑处理后，将单车信息、站点信息等相关数据发送到BikeDao。
4. BikeDao接收数据后，与数据库进行数据交互并返回结果集，从而完成修理操作。
5. BikeDao接收返回的数据后，将结果集转发给BikeController。
6. BikeController接收返回数据后，再封装相对应数据以及提示，控制转发到相对应页面，从而展示数据。

图5.19 投放单车用例的时序图

图5.20 投放单车用例的类图

5.11修改站点信息

该用例的时序图如图5.21所示，类图如图5.22所示。该用例的基本事件流如下：

1. 管理员在后台的站点管理页上，选择【修改】操作，并在弹出的模态框中输入修改信息。
2. 系统接收用户该次请求，将请求发送到SiteController。
3. SiteController进行逻辑处理后，将站点信息等相关数据发送到SiteDao。
4. SiteDao接收数据后，与数据库进行数据交互并返回结果集，从而完成修改操作，并将结果集转发给SiteController。
5. SiteController接收返回数据后，再封装相对应数据以及提示，控制转发到相对应页面，从而展示数据。

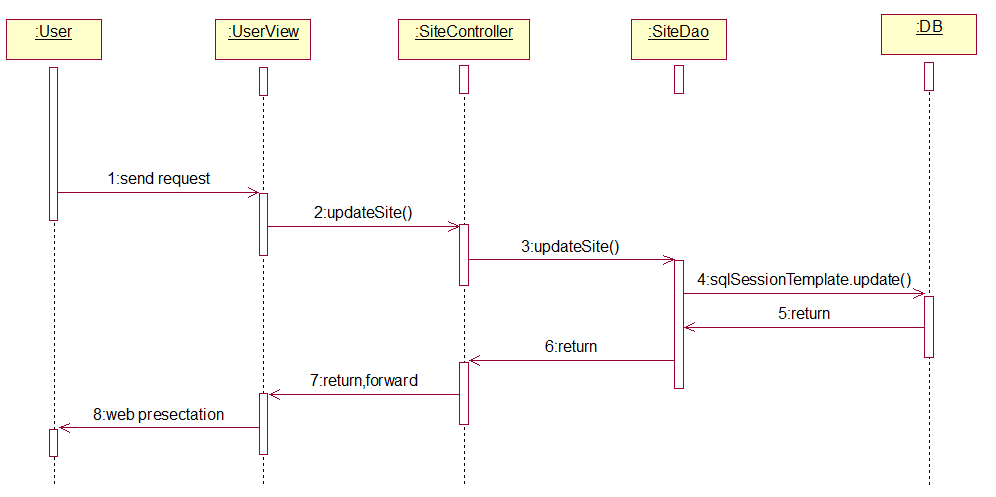
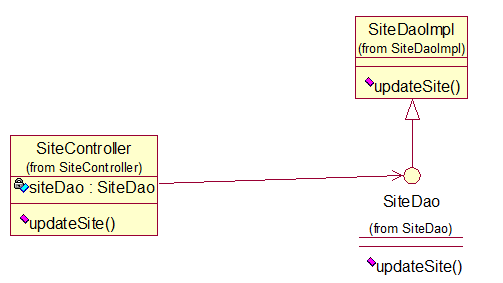
图5.21 修改站点信息用例的时序图

图5.22 修改站点信息用例的类图

5.12新增站点

该用例的时序图如图5.23所示，类图如图5.24所示。该用例的基本事件流如下：

1. 管理员在后台的新增站点页面上，在输入框中输入站点名字等相关信息。
2. 系统接收用户该次请求，将请求发送到SiteController。
3. SiteController进行逻辑处理后，将站点信息等相关数据发送到SiteDao。
4. SiteDao接收数据后，与数据库进行数据交互并返回结果集，从而完成新增站点操作，并将结果集转发给SiteController。
5. SiteController接收返回数据后，再封装相对应数据以及提示，控制转发到相对应页面，从而展示数据。

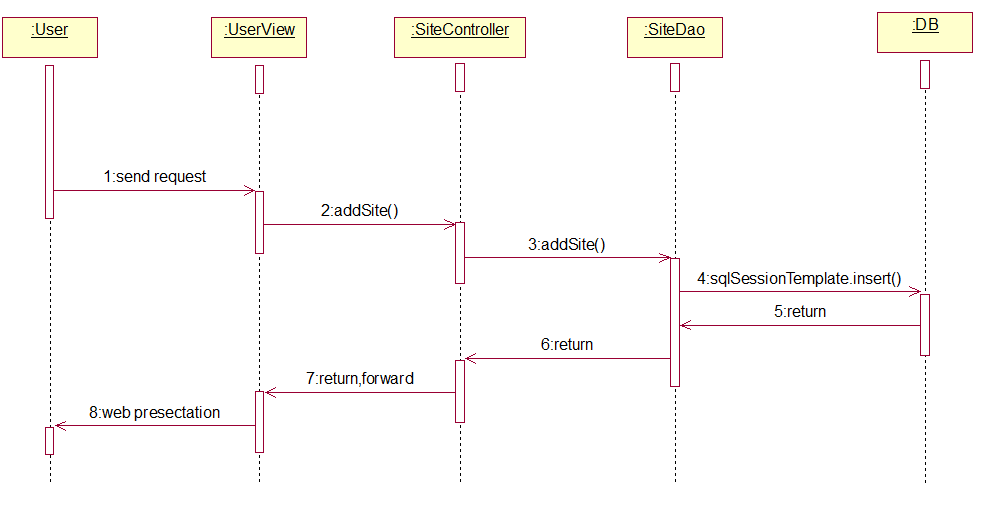


图5.23 新增站点用例的时序图

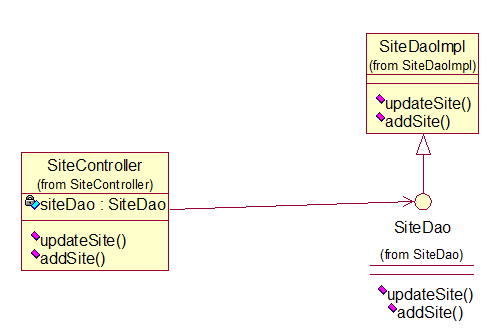


图5.24 新增站点用例的类图

5.13删除站点

该用例的时序图如图5.25所示，类图如图5.26所示。该用例的基本事件流如下：

1. 管理员在后台的站点管理页面上，选择【删除】操作。
2. 系统接收用户该次请求，将请求发送到SiteController。
3. SiteController进行逻辑处理后，将站点ID信息等相关数据发送到SiteDao。
4. SiteDao接收数据后，与数据库进行数据交互并返回结果集，从而完成删除站点操作，并将结果集转发给SiteController。
5. SiteController接收返回数据后，再封装相对应数据以及提示，控制转发到相对应页面，从而展示数据。

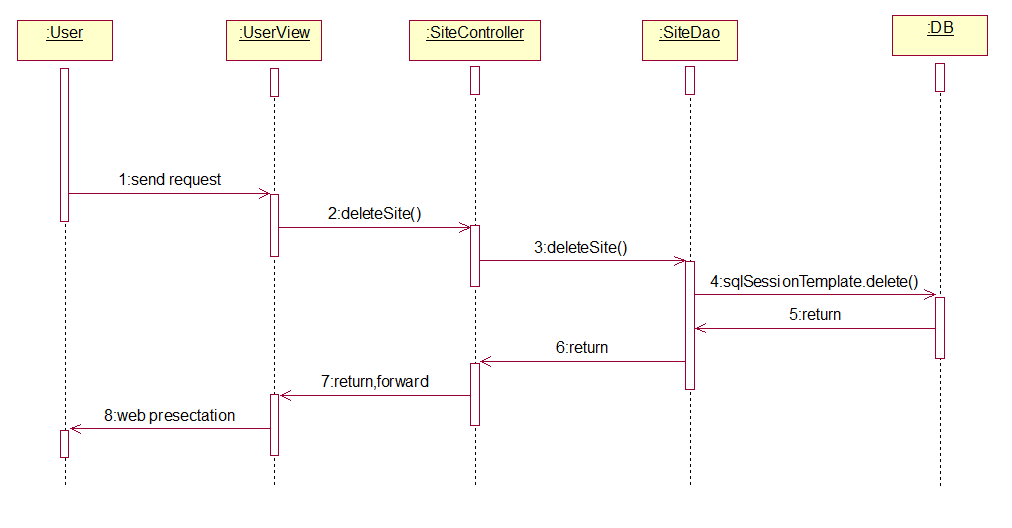


图5.25 删除站点用例的时序图

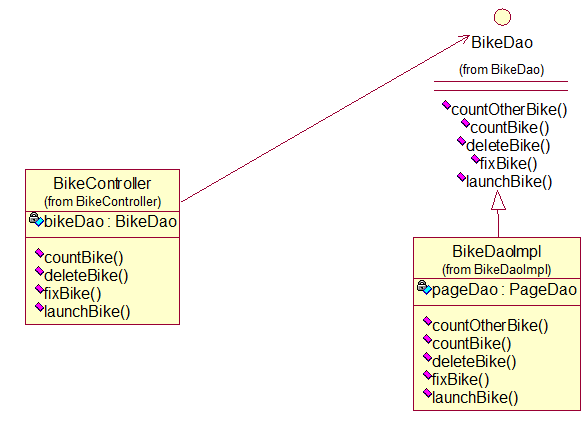


图5.26 删除站点用例的类图

# 6界面设计

在页面显示上，本系统采用了“厂”字型的页面布局。页面整体被划分为三部分，分别是：位于上方的用户信息栏、位于左边的操作导航栏，以及位于中间的页面显示部分。其中用户信息栏用于显示用户的登录登出操作和系统logo等相关信息；操作导航栏则是给用户提示相对应的操作；中间的页面显示部分则是用于显示用户与系统交互后返回的数据。

6.1登录

在登录页上，上面的导航栏是系统的logo。下方是登录框以及登录类型的选择。如图6.1所示。



图6.1 登录页面图

6.2用户操作页面

在本页面中，上面的导航栏是系统的logo以及系统欢迎语和登出操作。而在左方，用户可以选择相对应操作类型，页面的中间则是系统的主要数据显示页面。如图6.2所示。

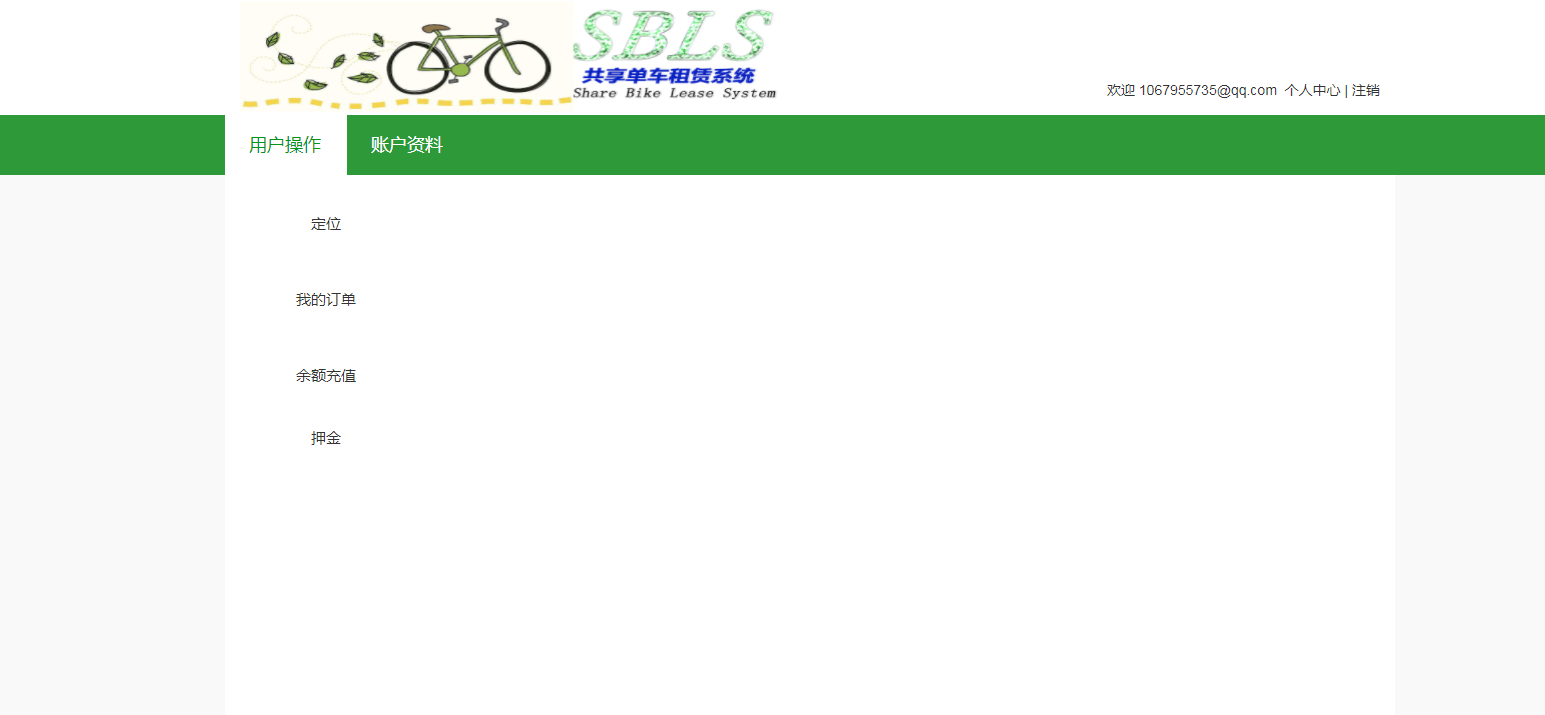


图6.2 用户操作图

6.3管理员操作页面

在本页面中，上面的导航栏是系统的logo以及系统欢迎语和登出操作。而在左方，管理员可以选择相对应操作类型，页面的中间则是系统的主要数据显示页面。如图6.3所示。



图6.3 管理员操作图

# 7代码实现

7.1登录

用户在登录SBLS系统时，首先前台的JS会对用户的输入信息做一个格式的验证。（如图7.1所示）若验证不通过，则会在具体的输入栏后面提示具体的错误原因。在通过验证后，用户填写的信息，经过SpringMVC转发到LoginController里面，LoginControler（如图7.2）封装好用户输入的信息后，通过LoginDao与数据库交互验证。若验证失败，则LoginController会封装好相对应的错误提示，然后返回到页面；若验证通过，则会将用户的信息放到session中，使用户在下一次有效期内登录可以不用重新输入用户名与密码，然后在封装好相对应的信息并跳转到用户首页。



图7.1 登录校验的JS



图7.2 UserController的login()方法

7.2定位

用户在定位页面上，选择SBLS系统给出的位置，此时系统会将用户所选的定位信息发送到UserController（如图7.3所示），UserController将用户所选择的信息封装并发送到UserDao中与数据库交互，从而取得该定位500米半径范围内的单车站点以及该站点内的单车信息情况。

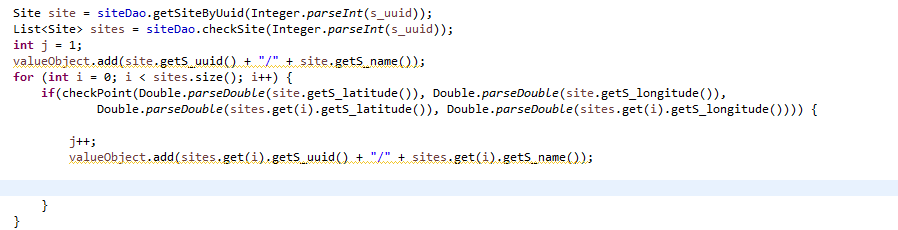


图7.3 UserController的location()方法

7.3租车

用户在单车信息显示页面上，选中某辆单车并进行租赁操作后，系统首先会通过JS（如图7.4）对用户进行验证，具体包括：用户是否有上一次租赁操作且未归还、用户是否缴纳了押金。通过验证后系统将用户所选择的单车编号等相关信息传送到LeaseController（如图7.5所示），LeaseController在取得单车编号，并封装用户ID、租赁时间等相应信息，然后发送给LeaseDao从而与数据库交互从而更改单车状态、站点状态、以及租赁记录。然后在将返回信息用JSONObject封装后返回给前台页面。



图7.4 租车的JS校验



图7.5 LeaseController的leaseBike()方法

7.4还车

用户在订单列表中，选择还车操作后，系统首先将通过JS（如图7.6）检测用户所选单车归还站点是否还有空余位置可供单车停靠。如果没有，则会提示用户选择其他站点；如果通过验证，系统会将单车ID、用户ID、站点ID等相关信息发送到LeaseController，LeaseController首先会计算出这次租赁操作所产生的费用。然后通过LeaseDao与数据库交互，判断用户当前的余额是否足够支付该次租赁操作所产生的费用。如果不够，LeaseController会封装相对应的错误提示信息并返回到前台页面，提示用户前往充值。若如果余额足够，则会修改数据库中关于单车、站点、租赁记录的相关信息。



图7.6 还车的JS校验

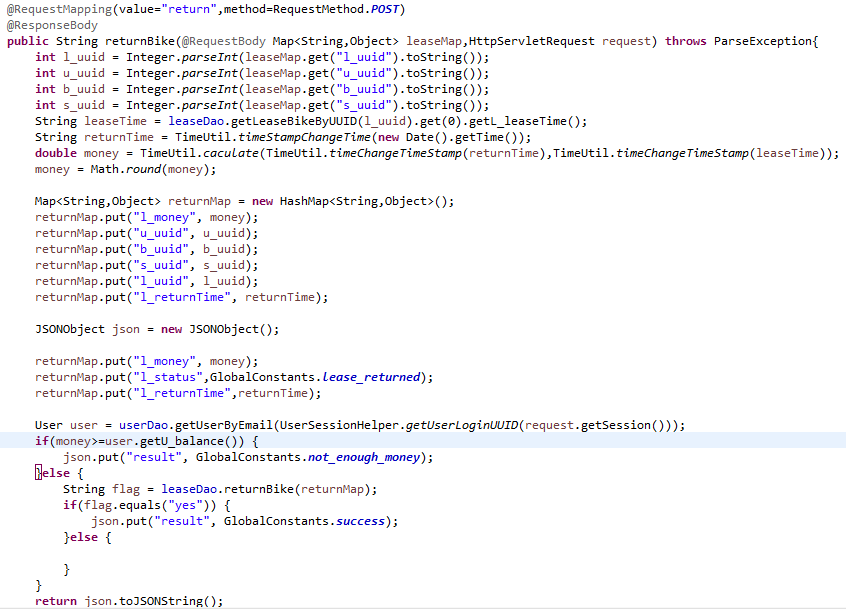


图7.7 LeaseController的returnBike()方法

7.4金额充值

用户在操作栏上选择金额充值后，输入所需充值的金额数目。系统首先通过JS（如图7.8）对金额数目作验证。若验证不通过（如金额数目为0或者负数，金额数目大于1000），则系统会给出相对应的错误提示；若验证通过，系统将金额数目发送到UserController（如图7.9），UserController会通过UserDao与数据库交互，从而修改用户的金额数目。然后再返回相对应信息到前台页面。



图7.8 金额充值的JS校验

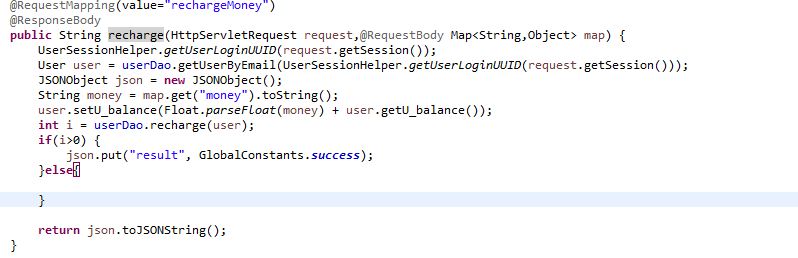


图7.9 UserController的recharge()方法

7.5重置密码

用户在登录页面上选择重置密码操作后，系统跳转到验证页面。用户输入身份证和邮箱后，系统首先通过JS（如图7.10）对用户所输入的身份证号进行验证。若验证不通过，则说明操作者不是本人，无法进行重置密码操作；若验证通过，系统将用户所输入的邮箱发送到UserController（如图7.11），UserController会调用AuditResultEmailThread（如图7.12）线程来发送邮件到用户输入的邮箱上，用户在接收到邮箱后点击邮箱中的地址从而跳转到重置密码的页面。此时用户在页面上输入新的密码。系统在经过验证后将密码发送到UserController，UserController将密码封装后通过UserDao与数据库交互，从而去修改用户存储在数据库中的密码。



图7.10 重置密码的JS校验



图7.11 UserController的showResetPassword()和resetPassword ()方法

图7.12 AuditResultEmailThread的sendEmail()方法

7.6单车数据统计

管理员在后台管理页面中，选择单车数据管理操作后，此时便相当于向BikeController（如图7.13）发送了一个请求，BikeController在接收该次请求后，根据相对应的URL路径转发，找到对应的getAllBike()方法，此时该方法会通过BikeDao去数据库查找所有的单车数据。与此同时，会通过countBike()方法取得数据，然后通过JfreeChat绘制相对应的饼状图来显示单车的数据情况。

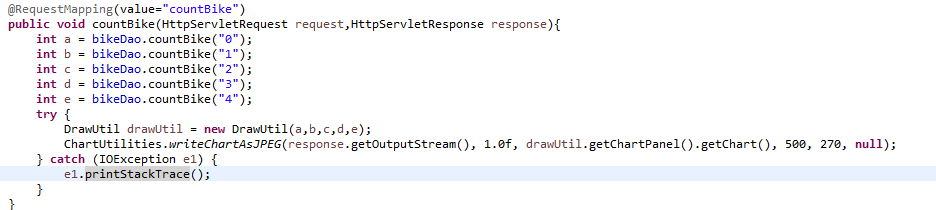


图7.13 BikeController的countBike()方法

7.7单车管理

管理员在后台管理页面中，选择单车管理操作后，此项系统接收请求，根据URL路径在BikeController（如图7.14）中找到了bikeManage()方法，然后通过BikeDao与数据库交互后返回了单车数据，再通过BikeController转发到前台显示页面。此时管理员可以在页面中对单车进行删除、维护、投放等操作。而这一系列操作都是通过BikeController来根据URL路径来找到相对应的处理方法，进而完成操作。

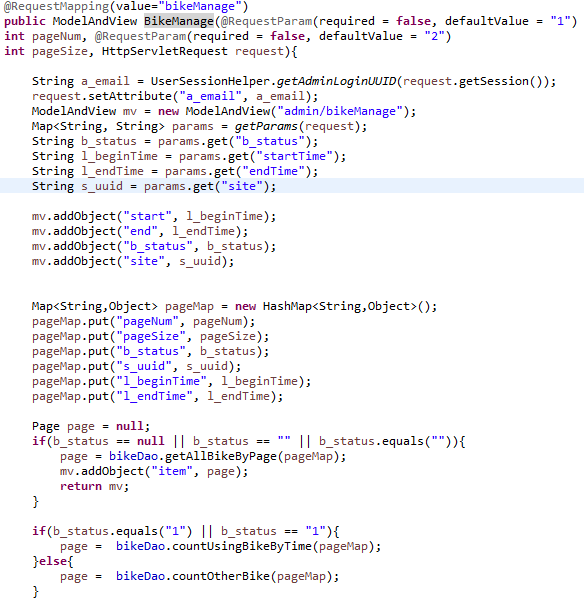


图7.14 BikeController的bikeManage ()方法

7.8删除单车

管理员在单车列表中，选择某辆单车后并点击修改按钮后，此事系统将该单车ID等相关信息发送到BikeController（如图7.15所示）的deleteBike()方法，此时该方法会先判断该辆单车是否处于使用状态。若是，则会封装相对应的错误提示并返回到前台；若不是，则通过BikeDao与数据库交互从而完成该辆单车的删除。

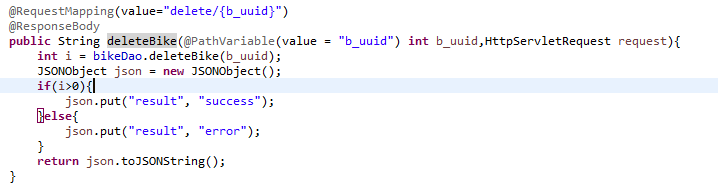


图7.15 BikeController的deleteBike()方法

7.9修理单车

管理员在单车列表中，选择某辆损坏的单车后并点击修理按钮，此时系统将该单车的ID等相关信息发送到BikeController（如图7.16）的fixBike()方法，该方法封装了单车信息后，通过BikeDao与数据库交互后，从而去修改单车状态。与此同时，系统中的定时任务（如图7.17、图7.18所示），会每隔5分钟去数据库查询状态为“待修理”的单车，从而去更改单车状态来达到修理单车的目的。

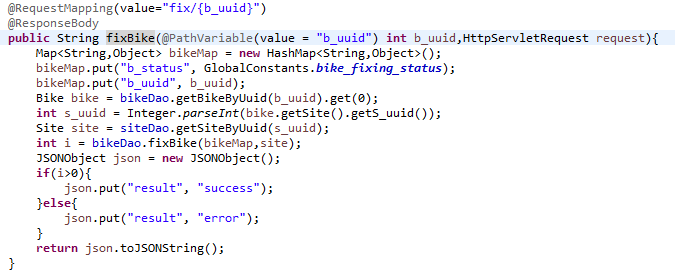


图7.16 BikeController的fixBike ()方法

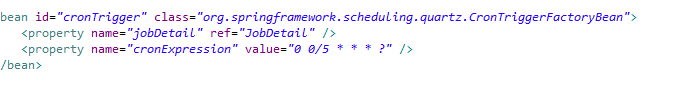


图7.17 定时任务的配置



图7.18 定时任务的具体处理方法

7.10投放单车

管理员在单车列表中，选择状态为“闲置”的单车后并点击投放按钮，此时系统会检测投放单车的站点是否有空余位置可供单车停靠。若没有，系统会给出相对应的错误提示；反之系统则将该单车的ID等相关信息发送到BikeController（如图7.19）的launch()方法，该方法封装了单车信息后，通过BikeDao与数据库交互，从而修改单车状态为空闲，达到投放单车的目的。

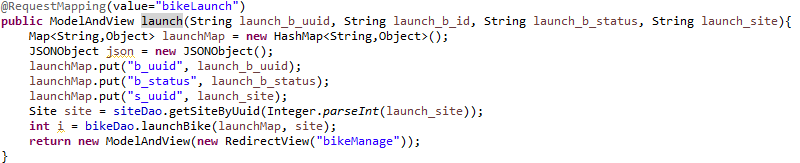


图7.19 BikeController的launch ()方法

7.11站点管理

管理员在后台管理页面中，选择站点管理操作后，此项系统接收请求，根据URL路径在SiteController（如图7.20）中找到了getAllSite ()方法，然后通过SiteDao与数据库交互后返回了站点数据，再通过SiteController转发到前台显示页面。此时管理员可以在页面中对站点进行修改、删除等操作。而这一系列操作都是通过SiteController来根据URL路径来找到相对应的处理方法，进而完成操作。



图7.20 SiteController的getAllSite ()方法

7.12修改站点信息

管理员在站点列表中，选择某个站点后并点击修改按钮，此时系统会先检测该站点是否处于使用状态（即：是否有单车停靠）。若是，则系统会给出相对应提示；反之系统则将该站点的ID以及管理员输入的数据等相关信息发送到SiteController（如图7.21所示）的updateSite()方法，该方法封装了站点信息后，通过SiteDao与数据库交互后，从而修改该站点信息。

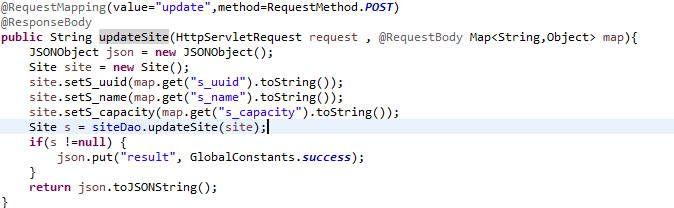


图7.21 SiteController的updateSite()方法

7.13新增站点

管理员在现在新增单车页中，在输入框输入站点名字，经纬度等相对应信息后，点击提交按钮。此时系统通过JS（如图7.22所示）会先检测用户输入信息是否合法。若不合法，则系统会给出相对应提示；若合法，系统此时则将用户输入的数据等相关信息发送到SiteController（如图7.23所示）的addSite()方法该方法封装了站点信息后，通过SiteDao与数据库交互后，从而新增站点。



图7.22 新增站点的JS校验

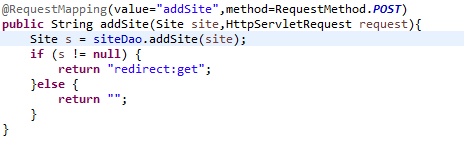


图7.23 SiteController的addSite ()方法

7.14删除站点

管理员在站点列表中，选择某个站点后并点击删除按钮，此时系统会先检测该站点是否处于使用状态（即：是否有单车停靠）。若是，则系统会给出相对应提示；反之系统则将该站点的ID等相关信息发送到SiteController（如图7.24）的deleteSite()方法，该方法封装了站点信息后，通过SiteDao与数据库交互后，从而删除该站点。

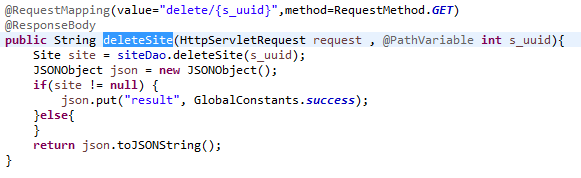


图7.24 SiteController的deleteSite()方法

# 8测试

8.1 测试方案与策略

|  |  |
| --- | --- |
| 测试功能模块 | 所采用的黑盒用例设计方法 |
| 登录 | 判定表法 |
| 定位 | 场景分析法 |
| 租车 | 场景分析法 |
| 还车 | 场景分析法 |
| 金额充值 | 等价类划分法  边界值分析法 |
| 单车管理 | 场景分析法 |
| 站点管理 | 场景分析法 |

8.2 测试用例

8.2.1登录

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | 账号 | 密码 | 预期输出 |
| SBLS-01 | 空 | 空 | 不执行登录，且在输入框右边提示不能为空 |
| SBLS-02 | 不合法的邮箱账号 | 正确密码 | 不执行登录，且在账号框右边提示邮箱账号不合法 |
| SBLS-03 | 合法但不存在的邮箱账号 | 正确密码 | 不执行登录，且在账号框右边提示邮箱账号不存在 |
| SBLS-04 | 合法且存在的账号 | 正确密码 | 成功登录 |
| SBLS-05 | 合法且存在的账号 | 错误密码 | 不执行登录，且在密码输入框右边提示密码错误 |

8.2.2定位

|  |  |  |
| --- | --- | --- |
| 用例编号 | 场景 | 预期结果 |
| SBLS-06 | 场景1：所选的定位周围500米内有站点 | 系统显示当前位置，且显示周围500米内站点的详细情况 |
| SBLS-07 | 场景2：所选的定位周围500米内没有站点 | 系统显示当前位置，且提示500米内没有站点，重新定位 |

8.2.3租车

|  |  |  |
| --- | --- | --- |
| 用例编号 | 场景 | 预期结果 |
| SBLS-08 | 场景1：用户已缴纳押金，且上一次租赁操作已归还单车 | 租赁成功 |
| SBLS-09 | 场景2：用户没有缴纳押金 | 租赁失败，系统提示用户尚未缴纳押金 |
| SBLS-10 | 场景：3用户已缴纳押金，但上一次租赁操作尚未归还单车 | 租赁失败，系统提示用户存在尚未结算的消费 |

8.2.4还车

|  |  |  |
| --- | --- | --- |
| 用例编号 | 场景 | 预期结果 |
| SBLS-11 | 场景1：用户所选择的还车站点没有空闲位置可供停靠 | 还车失败，系统提示该站点已无空闲位置，建议重新选择站点 |
| SBLS-12 | 场景2：用户余额不够支付本次消费 | 租赁失败，系统提示用户余额不足，请前往充值 |
| SBLS-13 | 场景3：用户选择正确的站点，且余额足够支付本次消费 | 还车成功 |

8.2.5金额充值

|  |  |  |  |
| --- | --- | --- | --- |
| 用例编号 | 金额数目 | 充值类型 | 预期输出 |
| SBLS-14 | 空 | 空 | 充值失败，系统提示应输入或选择金额数目和充值类型 |
| SBLS-15 | 含有特殊字符或字母的字符串 | 正确的类型 | 充值失败，系统提示金额数目不合法 |
| SBLS-16 | 小于0 | 正确的类型 | 充值失败，系统提示金额数目范围应在0-1000 |
| SBLS-17 | 0 | 正确的类型 | 充值失败，系统提示金额数目范围应在0-1000 |
| SBLS-18 | 1000 | 正确的类型 | 充值成功 |
| SBLS-19 | 大于1000 | 正确的类型 | 充值失败，系统提示金额数目范围应在0-1000 |

8.2.6单车管理

|  |  |  |
| --- | --- | --- |
| 用例编号 | 场景 | 预期结果 |
| SBLS-20 | 场景1：管理员所选择的单车处于使用中状态 | 操作失败，系统提示该单车正在使用中 |
| SBLS-21 | 场景2：管理员所选择的单车未处于使用中状态 | 操作成功 |

8.2.7站点管理

|  |  |  |
| --- | --- | --- |
| 用例编号 | 场景 | 预期结果 |
| SBLS-22 | 场景1：管理员所选择的站点内停放着单车 | 操作失败，系统提示该站点正在使用中 |
| SBLS-23 | 场景2：管理员所选择的站点内未停放着单车 | 操作成功 |

# 9总结

通过整个系统的设计与实现，再结合整个系统的需求分析和数据库设计，可以发觉系统的易用性比较高，同时安全性也可以满足。在功能业务方面，系统主要的定位、租赁、归还、统计等功能，都能得到实现，且响应时间也比较快。在涉及金钱交易方面，系统也做到了资金安全的保证，同时系统对并发用户时的情况也做了处理。于是可以得出，系统可以满足校园内共享单车的使用，可以为校园内共享单车的管理提供一个良好的帮助。

**参考文献**

[1] Kathy Sierra，Bert Bates.Head First Java[M].北京：中国电力出版社，2007：30-60.

[2] Nicholas C.Zakas.JavaScript高级程序设计第3版[M].北京：人民邮电出版社，2012：10-58.

[3] Freeman.E.Head First设计模式（中文版）[M].北京：中国电力出版社，2007:105-142.

[4] Budi,Kurniawan，Paul,Deck.Servlet、JSP和Spring MVC初学指南[M].北京：人民邮电出版社，2016：10-100.

[5] 王英瑛.JSP Web开发案例教程[M].北京：清华大学出版社，2013：52-89.

[6] 蒲子明，许勇，王黎.Struts2+Hibernate+Spring整合开发技术详解[M].北京：清华大学出版社，2010:1-500.

[7] 刘增杰.MySQL 5.7从入门到精通(视频教学版)[M].北京：清华大学出版社，2016年：100-120.

[8] 疯狂软件.Spring+MyBatis企业应用实战[M].北京：电子工业出版社，2017：100-280.

[9] 贾蓓.JavaWeb整合开发实践[M].北京：清华大学出版社，2013:150-241.

[10] 许令波.深入分析Java Web技术内幕(修订版)[M].北京：中国电力出版社，2014:37-69.

**谢 辞**

**附 录**

附录1

**程序源代码**

附录2

…….