目 录

[1 绪 论](#_Toc31005)

[1.1 研究的背景及意义](#_Toc8515)

[1.1.1 选题的背景 1](#_Toc31371)

[1.1.2 国内外研究现状 1](#_Toc20278)

[1.1.3 研究的意义 2](#_Toc28316)

[1.2 系统目标](#_Toc15206)

[2 需求分析](#_Toc2694)

[2.1 业务需求 4](#_Toc6489)

[2.2 功能需求](#_Toc25147)

[2.2.1 角色分析](#_Toc10678)

[2.2.2 业务功能](#_Toc9813)

[2.3 非功能需求](#_Toc19188)

[2.3.1环境需求](#_Toc4753)

[2.3.2 性能需求](#_Toc19578)

[2.3.3 安全需求](#_Toc6382)

[3 总体设计](#_Toc32694)

[3.1系统设计的原则](#_Toc30437)

[3.2 系统体系结构设计](#_Toc7925)

[3.3 系统功能结构设计](#_Toc3423)

[4 数据库设计](#_Toc8199)

[4.1 概念结构设计](#_Toc22860)

[4.1.1 设计思路 27](#_Toc30962)

[4.1.2 E-R图 27](#_Toc6671)

[4.2 逻辑结构设计](#_Toc20286)

[4.2.1 设计思路 29](#_Toc23570)

[4.2.2 逻辑模型 30](#_Toc3744)

[4.3 物理结构设计](#_Toc4422)

[4.3.1 存取方式 30](#_Toc2580)

[4.3.2 存储结构 30](#_Toc28251)

[5 界面设计](#_Toc8864)

[5.1 界面关系图](#_Toc15087)

[5.2 界面设计成果](#_Toc2392)

[5.2.1 主界面 33](#_Toc29706)

[5.2.2 子界面 34](#_Toc7773)

[6 详细设计](#_Toc23453)

[6.1 系统主要功能模块介绍](#_Toc15589)

[6.2企业动态](#_Toc9442)

[6.2.1功能结构设计](#_Toc1180)

[6.2.2类图设计](#_Toc11469)

[6.2.3顺序图设计](#_Toc29394)

[6.2.4核心处理流程设计](#_Toc26347)

[6.3用户管理](#_Toc29268)

[6.3.1功能结构设计](#_Toc10859)

[6.3.2类图设计](#_Toc32488)

[6.3.3顺序图设计](#_Toc10264)

[6.3.4核心处理流程设计](#_Toc8639)

[6.4企业管理](#_Toc22935)

[6.4.1功能结构设计](#_Toc30297)

[6.4.2类图设计](#_Toc31526)

[6.4.3顺序图设计](#_Toc9137)

[6.4.4核心处理流程设计](#_Toc27816)

[6.5日志管理](#_Toc21422)

[6.5.1功能结构设计](#_Toc31687)

[6.5.2类图设计](#_Toc17193)

[6.5.3顺序图设计](#_Toc26737)

[6.5.4核心处理流程设计](#_Toc5890)

[6.6统计报表](#_Toc15872)

[6.6.1功能结构设计](#_Toc18213)

[6.6.2类图设计](#_Toc11972)

[6.6.3顺序图设计](#_Toc13934)

[6.6.4核心处理流程设计](#_Toc19664)

[7 编码](#_Toc1723)

[7.1 代码实现与核心算法](#_Toc4661)

[7.2 代码优化分析](#_Toc29380)

[8 测试](#_Toc31762)

[8.1 测试方案设计](#_Toc870)

[8.1.1 测试策略 58](#_Toc22304)

[8.1.2 测试进度安排 58](#_Toc25728)

[8.1.3 测试资源 58](#_Toc4516)

[8.1.4 关键测试点 58](#_Toc32158)

[8.2 测试用例构建](#_Toc17902)

[8.2.1 测试用例编写约定 58](#_Toc21755)

[8.2.2 测试用例设计 59](#_Toc2505)

[8.2.3 关键测试用例 59](#_Toc14251)

[8.2.4 测试用例维护 61](#_Toc12947)

[9 总结与展望](#_Toc14306)

[9.1 设计工作总结](#_Toc31436)

[9.2 未来工作展望](#_Toc23226)

[谢 辞](#_Toc7515)

[参考文献](#_Toc22759)

[附录A 外文翻译—原文部分](#_Toc23527)

[附录B 外文翻译—译文部分](#_Toc12849)

[附录C 软件使用说明书](#_Toc20711)

[附录D 主要源代码](#_Toc4534)

1 绪 论

## 1.1 研究的背景及意义

### 1.1.1选题的背景

随着互联网技术的快速发展，计算机科学领域愈发成熟，人们的生活和工作依赖于信息技术变得更加便利和高效。计算机的强大功能让人们认识到计算机在人类文明发展的各个领域里有着非常重要的作用。照着这样的发展趋势，人们的生活将变得越来越数字化和信息化。在计算机众多强大的功能里，不得不说的是计算机能快速并且高效的将庞大的信息进行统一的管理，并且能够快速的从庞大的数据中查询到想要信息，就这一强大的功能，就足以让计算机在很多领域都发挥耀眼的光彩。就算计算机拥有如此强大的功能，能将其充分功能利用也是一项不容易的挑战。从计算机被创造出来到现在，一直都有大批的人才进行深入的研究。到现在，国内外的信息化产业都经历了快速发展，计算机技术可以说已经相对成熟了，就是因为其的方便高效，计算机技术的应用范围也越来越广阔，现在各个领域都在使用计算机技术，上到国家的各种管理系统，下到每个人与每个人之间的往来通信。计算机技术无处不在。

信息管理系统在最初只是统计而已，仅仅只是把同一类的信息集中化，随着信息技术的不断提高，信息管理系统能做到的事情也越来越多，不仅仅只是信息统计而已了。现在的信息管理系统不仅仅只是对信息进行各种操作，还能对庞大的信息进行分析并发现其中的一些规律。

建立全国企业信息管理系统的目标是利用现代化的计算机技术管理手段，建立科学的企业信息管理系统，提高全国企业管理的整体工作效率。本系统是针对全国企业信息集中审核、统一查询管理的系统，本系统具有操作灵活、功能齐全等特点，是加强企业管理信息化、自动化管理所应必备的计算机系统之一。

### 1.1.2 国内外研究现状

目前我国信息系统工程的现状只能说是还不够成熟还有很大的发展空间，国内很多人才都被这个巨大的发展空间所吸引，他们纷纷对这个领域进行研究。不光是个人，各级政府也正在积极推进“电子政务”，国内的许多城市及各种企业也已经着手整合与强化其各自的信息化应用系统，从这可以看出，全国将有更多、更大的信息系统建设项目开始开发。但是在信息化技术的推进过程中，总是不可避免的遇到一些问题，主要是需求分析时制定的目标不够科学，对我国现在的计算机技术认识不准确，没有严格的项目管理，研发出来的系统运行效益不够明显等，就是因为这些原因，导致很多的信息化项目研发失败或未能达到预期的目标，从而浪费了大量的人力，物力。归根结底，这其中的主要原因之一是我国的信息化建设的第三方监管机制的标准还不健全。全世界的各种实践表明：进行信息化的风险很高，当需要研发的信息系统需求的规模越大，其功能越复杂，在研发时所要担任的风险也就越大。

目前，在国内的各个行业进行研发的信息化项目工程建设中，很多的项目负责人都无法组织好专业的团队，难以成功的担任从项目的可行性分析、项目的规划设计、甚至是招标、方案评审再到工程监理和工程验收等项目研发的全过程的管理与工作任务分配的工作。利用计算机技术来辅助审计信息和对各种数据进行处理，在信息技术的进一步发展与普及之后，使得大部分的企业越来越依赖基于计算机技术研发的信息系统。正因为这样，人们发现太多的信息存在互联网上，对信息的隐私性也越来越担忧，正是有了这样的担忧，企业也更加关注信息的保密性，出于这样的目的，相关人才一直往这方面研究，这才出现了真正意义的信息审计系统。随着电子商务的全球普及，信息系统的审计对象、范围及内容将已经越来越大，采用的技术也越来越复杂。到目前为止，信息系统审计在全球来看，还是一个新的业务，说明信息系统审计正逐渐受到重视。

### 1.1.3 研究的意义

计算机能快速高效的将庞大的信息进行统一的管理，并且能够快速的从庞大的数据中查询到想要信息，而目前我国企业的数量巨大，并且将越来越多，计算机能有效的提高对这些庞大的企业数据的管理。因此，需要一套计算机系统来实现对全国企业信息的管理。就目前我国的计算机和网络技术而言，相信全国企业信息管理系统将会对人们有很大的帮助。

本系统的重要意义就是能够方便人们去查询一个企业的详细信息，也可以让人们更加直观的了解我国企业的分布情况。就人们的需求而言，在很多情况下，本系统都能发挥重要的作用。就比如当一个人找工作时想要了解哪些企业是自己理想的企业时，就可以通过本系统来查询企业的详情信息。除此之外，还有很多情况下都可以，总之，本系统的开发很有必要。

## 1.2 系统目标

全国企业信息管理系统适合各种操作系统，并且可以网页浏览。系统有较快的响应，并且有较高的可靠性、可扩展性和健壮性，对于后期的维护也有较高的可维护性。

首先，了解本系统所要实现的功能，做出系统的需求分析和系统设计文档，在需求分析阶段，对系统的每个功能模块先进行业务需求分析，画出每个业务的活动图；再分析系统功能需求，画出其用例图，之后再对系统进行非功能需求。在系统设计阶段，基于需求分析文档，进行系统的结构设计和功能设计，在结构设计中分析每个实体的属性并建立相应的数据库，在功能设计中完成对系统的总体功能结构、界面、系统的输入/输出等设计。然后根据需求分析内容和系统设计内容进行编码，利用SSM（Spring+SpringMVC+MyBatis）框架、Java编程语言和web技术（如HTML、CSS等）实现系统的功能。之后对完成的系统进行功能测试，在测试中把系统的错误解决，提高系统的可靠性。

本系统的主要目标是实现系统用户的身份验证、账户设置、企业信息查询、企业动态查看、统计报表等功能，用户通过了身份验证之后，可以进行账户设置来查看个人信息或者修改密码；用户登录系统后还可以查看企业信息，企业动态和企业的统计报表。实现系统管理员对用户信息管理、企业信息管理、企业信息的报表展示、日志信息管理、企业动态管理等功能。也就是说系统管理员登录本系统之后，可通过系统来实现对系统用户的信息管理；也可以对企业信息进行管理：对企业信息的增删查改；还可以对企业动态进行管理；除此之外，管理员还可以查看系统日志信息，了解到用户通过本系统进行的各种操作。

# 2 需求分析

## 2.1 业务需求

（1）系统提供的多种功能需要用户通过账号和密码登录之后才能进行相应的操作，所以对于有要求的用户需要有登录和注册的功能。

（2）对于企业查询这个功能，系统应满足任何用户，所以系统对于游客也会开放企业信息查询这个功能。

（3）用户在注册时填写的信息之后若有更改，可以在系统的个人信息中进行相应的修改操作。

（4）用户可以通过系统来录入企业信息，录入企业信息之后，需要系统管理员对录入的信息进行审核。

（5）用户可以查看录入的企业信息的审核状态，审核通过的企业信息可以可以在用户或者游客登录的前台进行展示，同时，可以让使用本系统的任何用户查询到该企业信息。

（6）用户可以查看系统根据数据库中所有的企业信息按行业类别或者地区等其他标准生成的数据统计报表。

（7）用户还可以通过系统查看到各企业的动态信息，了解企业近况。

（8）用户在系统中进行的某些操作，系统会自动记录下来存入数据库中，系统管理员可以通过本系统对用户的操作日志进行查看，从而更好的管理。

（9）系统管理员可以对系统内的企业信息进行管理，不管是批量添加企业信息还是批量删除企业信息，系统都提供良好的界面体验。

（10）在用户录入了企业信息之后，这些信息都还是审核中的状态，系统管理员可以通过系统来确认企业信息，再决定审核通不通过。

（11）系统管理员可以对企业动态信息进行管理，管理员发布企业动态之后会在系统平台上展示，可供用户查看。

（12）系统管理员还可以对用户的信息进行管理。

系统的主要业务流程如下所示：

（1）用户的登录注册活动图如图2-1所示：

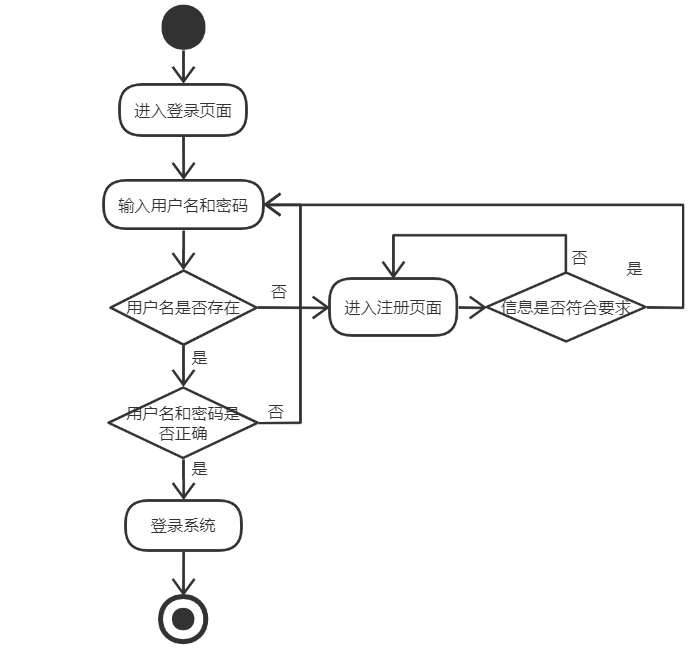


图2-1用户登录注册活动图

（2）企业信息查询活动图如下图2-2所示：

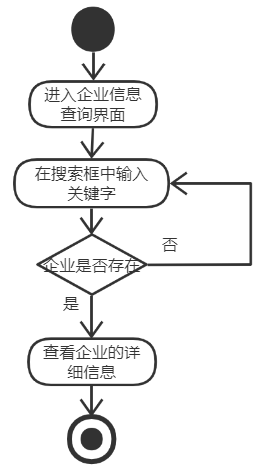


图2-2企业信息查询活动图

（3）企业信息审核活动图如下图2-3所示：

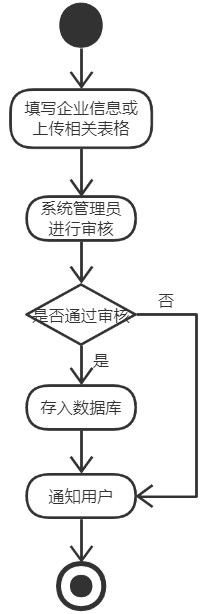


图2-3企业信息审核活动图

（4）企业动态信息发布活动图如下图2-4所示：

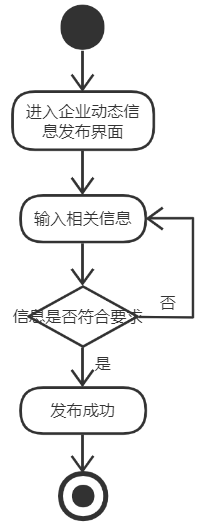


图2-4企业动态信息发布活动图

（5）用户查看企业信息统计报表的活动图如下图2-5所示：

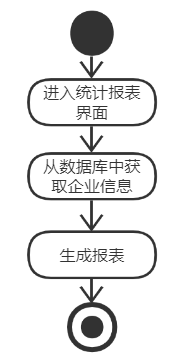


图2-5查看企业信息统计报表活动图

（6）查看用户信息活动图如下图2-6所示：

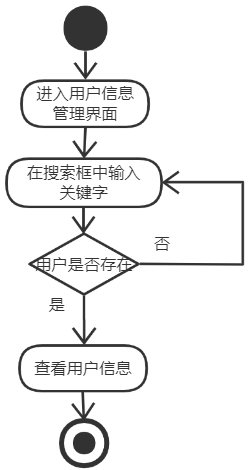


图2-6查看用户信息活动图

## 2.2 功能需求

### 2.2.1 角色分析

以下从全国企业信息管理系统的实际需求分析，系统涉及到的角色有三个，其具体的功能如下表2-1所示：

表2-1 系统角色表

|  |  |
| --- | --- |
| 角色 | 职责或功能 |
| 普通用户 | 进行企业信息的录入、查询、报表信息的查看等。 |
| 系统管理员 | 主要负责管理和维护整个系统的各种信息，负责对用户信息、企业信息、企业动态信息进行管理，并且可以对企业信息进行审核。 |
| 游客 | 可以进行企业信息的查询和查看企业动态信息。 |

### 2.2.2 业务功能

以下从业务角度出发，给出了系统的总体用例图，包含用户信息管理、企业信息录入、企业信息查询、各企业动态查看、全国企业信息的统计报表查看、查看审计流程、企业信息管理和企业信息审核等用例，如下图2-7所示：

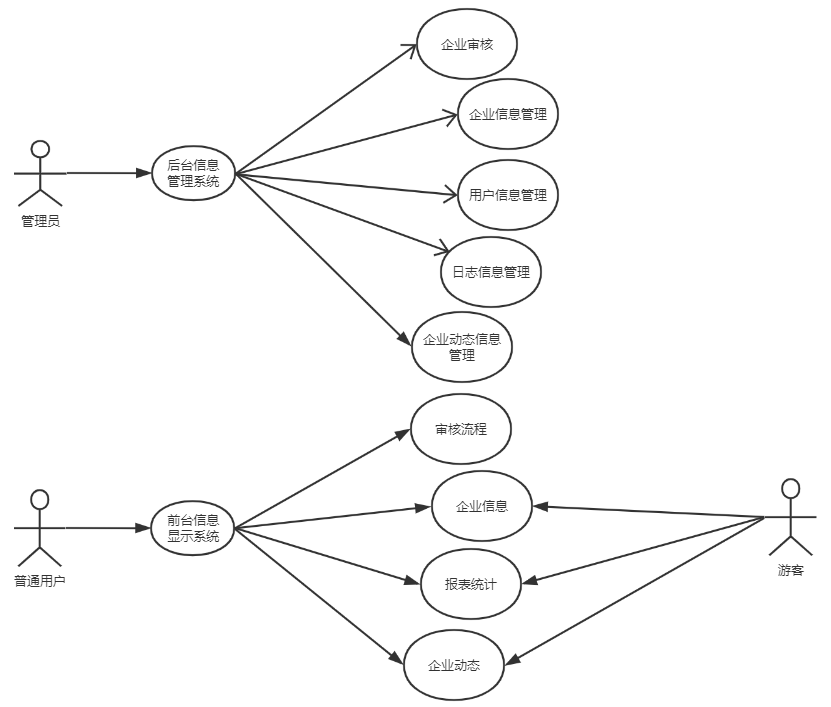


图2-7系统总体用例图

（1）普通用户

以下从业务角度出发，给出普通用户登录系统后的用例图，包含企业信息，审核流程，企业动态，查看统计报表等功能。如下图2-8所示：

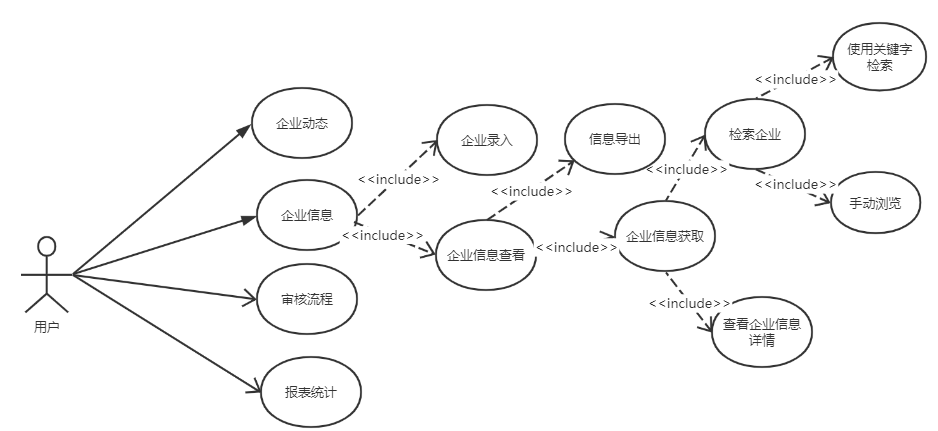


图2-8普通用户用例图

企业信息查看的用例描述如下表2-2所示：

表2-2企业信息查看用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 企业信息查看 |
| 标识符\* | 001 |
| 用例描述 | 描述了系统的普通用户使用本系统查看企业信息的整个过程。 |
| 参与者表 | 普通用户 |
| 优先级 | 1 |
| 状态\* | 进行中 |
| 前置条件 | 用户已登录系统 |
| 后置条件 | 系统给出操作成功提示 |
| 基本操作流 | 1. 用户在相应的系统主页上选择企业查询 2. 在企业信息列表查看界面提取符合条件的企业信息列表 3. 找到目标企业后点击“查看”按钮，进入企业详细信息查看页面。 |
| 可选操作流 | 1. 用户在企业查询界面可以选择导出为Excel来实现对企业信息的Excel表格文件导出。 2. 在搜索框中对要搜索的企业的关键信息进行检索 |
| 频率\* | 高 |

企业信息获取的用例描述如表2-3所示：

表2-3企业信息获取用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 企业信息获取 |
| 标识符\* | 002 |
| 用例描述 | 描述了用户使用本系统进行企业信息获取的过程 |
| 参与者表 | 普通用户 |
| 优先级 | 2 |
| 状态\* | 进行中 |
| 前置条件 | 用户已登录系统 |
| 后置条件 | 系统显示所获取到的企业信息 |
| 基本操作流 | 1.用户在系统主页上选择企业查询，进入企业列表查看界面。  2.输入关键字，展示和关键字相关企业列表信息。 |
| 被包含用例表 | 企业信息查看 |
| 频率\* | 较高 |

报表统计的用例描述如表2-4所示：

表2-4报表统计用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 报表统计 |
| 标识符\* | 003 |
| 用例描述 | 描述了用户通过本系统查看数据库中所有企业信息以不同标准（如：地区，类型等）展示的统计报表。 |
| 参与者表 | 普通用户 |
| 优先级 | 2 |
| 状态\* | 进行中 |
| 前置条件 | 用户已登录系统 |
| 后置条件 | 报表统计页面有图形展示出来 |
| 基本操作流 | 1.用户在系统主页面上选择报表统计就可以看到各类报表。 |
| 频率\* | 较高 |

信息导出的用例描述如表2-5所示：

表2-5信息导出用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 信息导出 |

续表2-5

|  |  |
| --- | --- |
| 标识符\* | 004 |
| 用例描述 | 描述了用户使用本系统将企业信息输出为Excel文件的整个过程。 |
| 参与者表 | 普通用户 |
| 优先级 | 2 |
| 状态\* | 进行中 |
| 前置条件 | 用户已登录系统 |
| 后置条件 | 文件下载成功 |
| 基本操作流 | 1．在系统的主页面上点击企业查询；  2. 点击导出为excel。 |
| 被包含用例表 | 企业信息查看 |
| 频率\* | 较低 |

搜索企业的用例描述如表2-6所示：

表2-6检索企业用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 检索企业 |
| 标识符\* | 005 |
| 用例描述 | 描述了用户使用本系统在企业信息查看时进行企业检索的整个过程。 |
| 参与者表 | 普通用户 |
| 优先级 | 2 |
| 状态\* | 进行中 |
| 前置条件 | 用户已登录系统 |
| 后置条件 | 系统给出用户所搜索的企业信息列表 |
| 基本操作流 | 1.用户点击在文本框中输入关键字  2.点击“搜索”按钮 |
| 被包含用例表 | 企业信息获取 |
| 被扩展用例表 | 手动浏览企业信息、使用关键字进行企业检索 |
| 频率\* | 较高 |

查看企业详情信息的用例描述如表2-7所示：

表2-7查看企业详细信息用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 查看企业详细信息 |
| 标识符\* | 006 |
| 用例描述 | 描述了用户使用本系统在企业信息查看时进行企业详细信息查看的整个过程 |
| 参与者表 | 普通用户 |
| 优先级 | 2 |
| 状态\* | 进行中 |
| 前置条件 | 用户已登录系统 |
| 后置条件 | 系统给出用户所查看企业详细信息 |
| 基本操作流 | 1.选择某一企业  2.点击“查看详情”按钮 |
| 被包含用例表 | 企业信息获取 |
| 频率\* | 较高 |

企业录入的用例描述如表2-8所示：

表2-8企业录入用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 企业录入 |
| 标识符\* | 007 |
| 用例描述 | 描述了用户使用本系统录入企业信息的整个过程 |
| 参与者表 | 普通用户 |
| 优先级 | 1 |
| 状态\* | 进行中 |
| 前置条件 | 用户已登录系统 |
| 后置条件 | 系统给将信息存入数据库，并等待管理员审核 |
| 基本操作流 | 1.用户在系统主页上点击企业录入  2.在页面上填写各企业的基本信息  3.点击添加 |
| 可选操作流 | 用户在录入企业界面可以选择导入文件来实现录入企业信息。 |
| 被包含用例表 | 企业信息 |
| 频率\* | 较高 |

审核流程的用例描述如表2-9所示：

表2-9审核流程用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 审核流程 |
| 标识符\* | 008 |
| 用例描述 | 描述了用户使用本系统查看企业信息审核状态的整个过程 |
| 参与者表 | 普通用户 |
| 优先级 | 2 |
| 状态\* | 进行中 |
| 前置条件 | 用户已登录系统 |
| 后置条件 | 系统给出还未审核的企业信息和未通过审核的企业信息 |
| 基本操作流 | 1.用户在主页点击审核流程即可查看到信息 |
| 频率\* | 较高 |

企业动态的用例描述如表2-10所示：

表2-10企业动态用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 查看企业动态信息 |
| 标识符\* | 009 |
| 用例描述 | 描述了用户使用本系统查看各企业的动态的过程 |
| 参与者表 | 普通用户 |
| 优先级 | 1 |
| 状态\* | 进行中 |
| 前置条件 | 用户已登录系统 |
| 后置条件 | 系统给出各企业的动态信息 |
| 基本操作流 | 1.登录系统  2.查看企业动态 |
| 可选操作流 | 1.查看企业详细信息 |
| 频率\* | 较高 |

（2）管理员

以下从业务角度出发，给出了系统管理员在登录本系统后的用例图，其中包含企业信息管理，用户信息管理，企业录入，日志信息管理，企业审核，企业动态信息管理等功能，如图2-9所示：

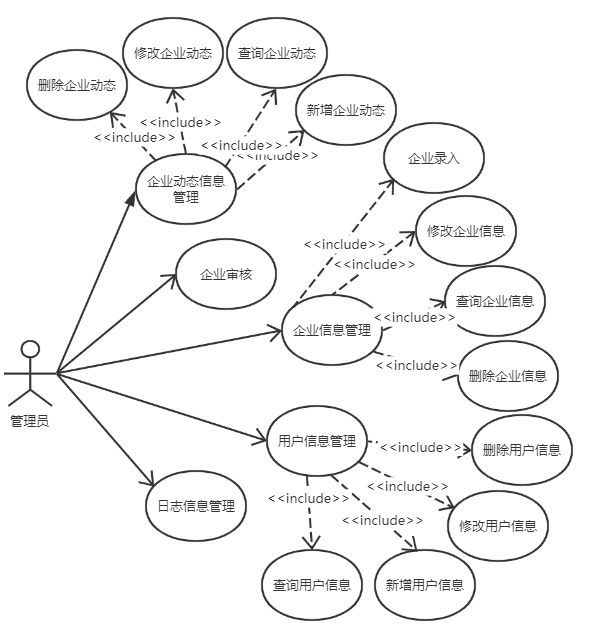


图2-9后台信息管理用例图

企业信息管理的用例描述如表2-11所示：

表2-11企业信息管理用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 企业信息管理 |
| 标识符\* | 010 |
| 用例描述 | 描述了管理员使用本系统进行企业信息的管理过程，包括查询企业信息，录入企业信息，修改企业信息，删除企业信息。 |
| 参与者表 | 系统管理员 |
| 优先级 | 1 |

续表2-11

|  |  |
| --- | --- |
| 状态\* | 进行中 |
| 前置条件 | 用户已登录系统 |
| 后置条件 | 系统显示操作成功 |
| 基本操作流 | 1.管理员在系统主页上选择企业信息管理，进入企业信息列表查看界面。  2.可以在输入框输入关键字，展示和关键字相关企业列表信息，还可以勾选某一企业，点击修改或删除来实现相应操作。 |
| 可选操作流 | 查询企业信息，修改企业信息，删除企业信息 |
| 频率\* | 较高 |

用户信息管理的用例描述如表2-12所示：

表2-12用户信息管理用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 用户信息管理 |
| 标识符\* | 011 |
| 用例描述 | 描述了管理员使用本系统进行用户信息的管理过程，包括增加用户信息，更新用户信息，删除用户信息。 |
| 参与者表 | 系统管理员 |
| 优先级 | 1 |
| 状态\* | 进行中 |
| 前置条件 | 用户已登录系统 |
| 后置条件 | 系统显示操作成功 |
| 基本操作流 | 1.管理员在系统主页上选择用户信息管理，进入用户信息列表查看界面。  2.可以在输入框输入关键字，展示和关键字相关用户列表信息，还可以勾选某一用户，点击修改或删除来实现相应操作。 |
| 可选操作流 | 查询用户信息，新增用户信息，修改用户信息，删除用户信息 |
| 频率\* | 高 |

日志信息管理的用例描述如表2-13所示：

表2-13日志信息管理用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 日志信息管理 |
| 标识符\* | 012 |
| 用例描述 | 描述了管理员使用本系统查看日志信息的整个过程。 |

续表2-13

|  |  |
| --- | --- |
| 参与者表 | 管理员 |
| 优先级 | 1 |
| 状态\* | 进行中 |
| 前置条件 | 用户已登录系统 |
| 后置条件 | 能查看到所有的日志信息 |
| 基本操作流 | 1．在系统的主页面上点击日志管理； |
| 频率\* | 高 |

企业审核的用例描述如表2-14所示：

表2-14企业审核用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 企业审核 |
| 标识符\* | 013 |
| 用例描述 | 描述了管理员使用本系统对企业信息进行企业信息审核的整个过程。 |
| 参与者表 | 管理员 |
| 优先级 | 1 |
| 状态\* | 进行中 |
| 前置条件 | 用户已登录系统，有需要审核的信息 |
| 后置条件 | 系统显示操作成功 |
| 基本操作流 | 1.管理员在系统主页上点击企业审核；  2.勾选需要某个或多个企业，点击“通过”或“不通过”。 |
| 频率\* | 较高 |

企业动态信息管理的用例描述如表2-15所示：

表2-15企业动态信息管理用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 企业动态信息管理 |
| 标识符\* | 014 |
| 用例描述 | 描述了管理员使用本系统对企业动态信息进行各种操作的过程。 |
| 参与者表 | 管理员 |
| 优先级 | 1 |
| 状态\* | 进行中 |
| 前置条件 | 用户已登录系统 |

续表2-15

|  |  |
| --- | --- |
| 后置条件 | 系统显示操作成功 |
| 基本操作流 | 1.管理员在系统主页上点击企业动态管理；  2.对信息进行相应操作。 |
| 可选操作流 | 新增企业动态，修改企业动态，删除企业动态，查询企业动态 |
| 频率\* | 较高 |

新增用户的用例描述如表2-16所示：

表2-16新增用户用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 新增用户 |
| 标识符\* | 015 |
| 用例描述 | 描述了管理员使用本系统新增用户的整个过程。 |
| 参与者表 | 管理员 |
| 优先级 | 1 |
| 状态\* | 进行中 |
| 前置条件 | 管理员已登录系统 |
| 后置条件 | 新增了用户信息 |
| 基本操作流 | 1. 在系统的主页面上点击用户信息管理； 2. 点击添加用户按钮； 3. 输入基本信息； 4. 点击添加。 |
| 被包含用例表 | 用户信息管理 |
| 频率\* | 高 |

修改企业信息的用例描述如表2-17所示：

表2-17修改企业信息用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 修改企业信息 |
| 标识符\* | 016 |
| 用例描述 | 描述了管理员使用本系统修改企业信息的整个过程。 |
| 参与者表 | 管理员 |
| 优先级 | 1 |
| 状态\* | 进行中 |
| 前置条件 | 管理员已登录系统 |

续表2-17

|  |  |
| --- | --- |
| 后置条件 | 企业信息修改成功 |
| 基本操作流 | 1. 在系统的主页面上点击企业信息管理； 2. 选中某一企业，点击修改按钮； 3. 修改需要修改的信息； |
| 被包含用例表 | 企业信息管理 |
| 频率\* | 高 |

企业动态信息删除的用例描述如表2-18所示：

表2-18企业动态信息删除用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 企业动态信息删除 |
| 标识符\* | 017 |
| 用例描述 | 描述了管理员使用本系统删除企业动态信息的整个过程。 |
| 参与者表 | 管理员 |
| 优先级 | 1 |
| 状态\* | 进行中 |
| 前置条件 | 管理员已登录系统 |
| 后置条件 | 选中的企业动态信息删除成功 |
| 基本操作流 | 1. 在系统的主页面上点击企业动态管理； 2. 选中某一企业或多个企业； 3. 点击删除按钮。 |
| 被包含用例表 | 企业动态信息管理 |
| 频率\* | 高 |

企业录入的用例描述如表2-19所示：

表2-19企业录入用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 企业录入 |
| 标识符\* | 018 |
| 用例描述 | 描述了管理员使用本系统录入企业信息的整个过程 |
| 参与者表 | 管理员 |
| 优先级 | 1 |
| 状态\* | 进行中 |
| 前置条件 | 管理员已登录系统 |

续表2-19

|  |  |
| --- | --- |
| 后置条件 | 系统给将信息存入数据库，并等待管理员审核 |
| 基本操作流 | 1.管理员在系统主页上点击企业录入  2.在页面上填写各企业的基本信息  3.点击添加 |
| 可选操作流 | 管理员在录入企业界面可以选择导入文件来实现录入企业信息。 |
| 被包含用例表 | 企业信息管理 |
| 频率\* | 较高 |

（3）游客

以下从业务角度出发，给出游客的用例图，包含企业信息，企业动态，查看统计报表等功能。如下图2-10所示：

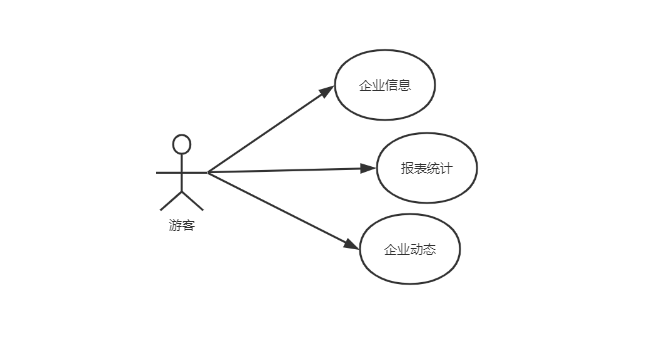


图2-10游客用例图

企业信息的用例描述如下表2-20所示：

表2-20企业信息用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 企业信息 |
| 标识符\* | 019 |
| 用例描述 | 描述了游客使用本系统查看企业信息的整个过程。 |
| 参与者表 | 游客 |
| 优先级 | 1 |
| 状态\* | 进行中 |
| 前置条件 | 已进入系统 |
| 后置条件 | 系统页面有信息显示 |

续表2-20

|  |  |
| --- | --- |
| 基本操作流 | 1. 在相应的系统主页上选择企业查询 2. 在企业信息列表查看界面提取符合条件的企业信息列表 3. 找到目标企业后点击“查看”按钮，进入企业详细信息查看页面。 |
| 可选操作流 | 用户在企业查询界面可以选择导出为Excel来实现对企业信息的Excel表格文件导出。 |
| 频率\* | 高 |

企业动态的用例描述如表2-21所示：

表2-21企业动态用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 查看企业动态信息 |
| 标识符\* | 020 |
| 用例描述 | 描述了游客通过本系统查看各企业的动态的过程 |
| 参与者表 | 游客 |
| 优先级 | 1 |
| 状态\* | 进行中 |
| 前置条件 | 已进入系统 |
| 后置条件 | 系统给出各企业的动态信息 |
| 基本操作流 | 1.进入系统  2.查看企业动态 |
| 可选操作流 | 1.查看企业详细信息 |
| 频率\* | 较高 |

报表统计的用例描述如表2-22所示：

表2-22报表统计用例描述

|  |  |
| --- | --- |
| 描述项 | 说明 |
| 用例名称 | 报表统计 |
| 标识符\* | 021 |
| 用例描述 | 描述了游客通过本系统查看数据库中所有企业信息以不同标准（如：地区，类型等）展示的统计报表。 |
| 参与者表 | 游客 |
| 优先级 | 2 |
| 状态\* | 进行中 |
| 前置条件 | 已进入系统 |

续表2-22

|  |  |
| --- | --- |
| 后置条件 | 报表统计页面有图形展示出来 |
| 基本操作流 | 1.在系统主页面上选择报表统计就可以看到各类报表。 |
| 频率\* | 较高 |

## 2.3 非功能需求

### 2.3.1环境需求

系统的服务器端的环境需求如下表2-23所示：

表 2-23服务器端环境

|  |  |
| --- | --- |
| 系统所使用数据库 | MySQL 5.7 |
| 操作系统 | Windows 10专业版 |
| Web服务器 | Tomcat 9.0 |
| 开发环境 | IDEA 2019.1.3+Maven 3.0.5 |

客户端的环境如下表2-24所示：

表 2-24客户端环境

|  |  |
| --- | --- |
| 操作系统 | Windows10版本及以上操作系统 |
| 浏览器 | IE 6.0以及其以上版本浏览器，Chrome浏览器等主流浏览器 |
| 分辨率 | 最佳效果为1024×768像素 |

### 2.3.2 性能需求

1. 系统在运行时，能够正常运行，不会出现系统崩溃现象。
2. 用户在使用系统时，响应速度不能太慢，要尽量快速响应。
3. 系统的需求在使用过程中会慢慢增多，所以系统要有足够的可扩展性，以便今后有其他功能的添加。
4. 系统的各个功能模块的界面应该要尽量简洁，美观。让用户有较好的体验。
5. 考虑到管理员可能会误删信息，系统的信息删除操作之后应还可以还原信息。

### 2.3.3 安全需求

1. 容错性

用户在使用本系统时，总会出现不符合系统的基本操作流的操作，在这样的情况下就需要系统自身有一定的容错性，并且可以告知正在操作的用户。

1. 系统的外在环境安全

每个系统都有自己的数据库来存储有关的信息，这些信息都是非常重要的。网络上有很多黑客会想尽办法窃取某个系统的信息库，所以为了系统信息的安全考虑，系统应做好信息保密措施，主要方法是设立防火墙和把数据进行加密处理。

1. 系统内部安全

在确保了系统外部的环境安全的同时，系统的内部安全也特别重要，要确保登录的用户操作合法。

3 总体设计

## 3.1系统设计的原则

本系统的开发主要是为了实现全国企业信息管理的系统化、规范化和自动化，来达到全国企业信息管理高效率的目的。使得人们能够更加方便快捷地查询企业信息以及能够使得相关人员能够更加方便快捷地对全国企业信息进行管理，以此来提高工作效率和经济效益。计算机的功能是强大的，必须要充分利用其功能，才能设计出功能强大的系统，同时要确保尽可能地减少系统资源的占用，同时本系统应该要符合各种功能需求和非功能需求。

## 3.2 系统体系结构设计

本系统主要利用Java编程语言、web技术（如bootstrap、HTML、CSS等）、SSM（Spring+SpringMVC+MyBatis）框架等来实现系统的功能，整个设计过程利用面向对象的方法进行开发，在编码设计之前先进行系统的需求分析，再进行系统的详细设计两个步骤，确保系统能成功完成。系统结构图如3-1所示:

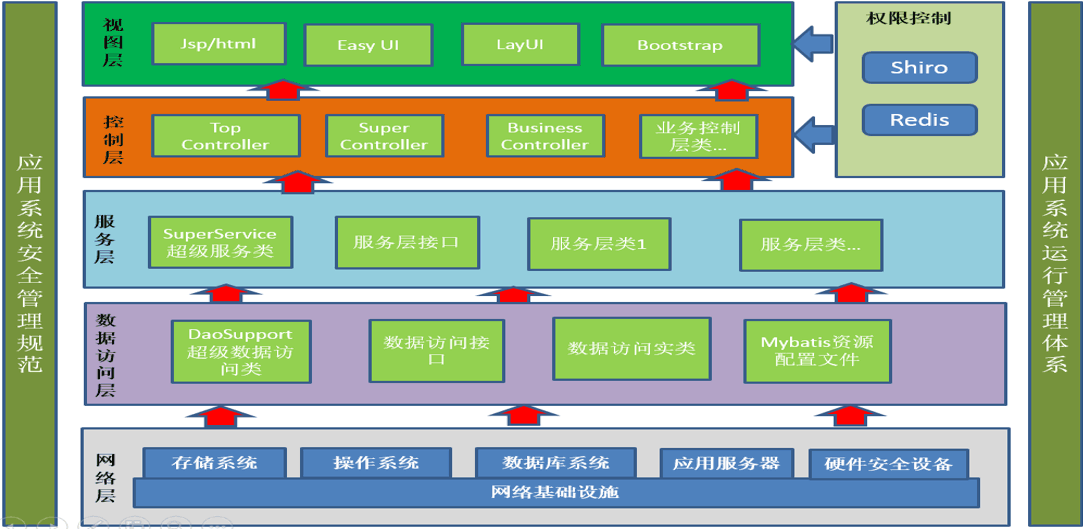


图3-1 系统结构图

SSM框架是由Spring和MyBatis两个开源的框架整合而成的，而Spring中又包含Spring和SpringMVC。SSM框架是目前web系统开发中运用较为普遍的框架，因为SSM框架是由Spring、SpringMVC和MyBatis整合而成的，所以SSM框架的优点就在于将三者的优点集中起来。

其中，Spring的创建是为了解决企业应用开发的复杂性的，简而言之，Spring是一个轻量级的控制反转（IoC）和面向切面（AOP）的容器框架；AOP，即面向切面编程，在AOP当中，实现了对service层的方法定义了切入点，从而在执行service层的方法的时候会自动加入事务；在本系统设计当中，就利用了这个来实现对用户的各种操作进行记录，例如，当用户进行企业查询的时候，系统会记录这个操作信息。这些信息形成日志信息提供给管理员查看。而SpringMVC是使用了MVC设计思想的轻量级web框架，对web层进行解耦，使我们开发更简洁与Spring无缝衔接。本系统在开发时充分利用其优势，将每个模块的业务模型和视图的代码分开，这样便于后期的维护。而MyBatis最大的特点是将数据库的操作(sql)采用xml文件配置，解除了sql和代码的耦合，提供映射标签。正因为这些优点，SSM框架则成为众多web项目开发的第一选择。

## 3.3 系统功能结构设计

系统的功能模块图如图3-2所示，全国企业信息管理系统就不同角色来说可以分为两个子系统，即前台子系统和后台信息管理子系统。其中，前台子系统主要是实现普通用户和游客在本系统中能够使用的功能，即企业动态信息查看、全国企业信息查询、企业信息审核流程查看、还有根据全国企业信息生成的报表统计共4个模块。而基于系统管理员实现的后台信息管理子系统又可分为用户信息管理、企业信息管理、日志管理、企业动态管理和企业审核共5个模块。

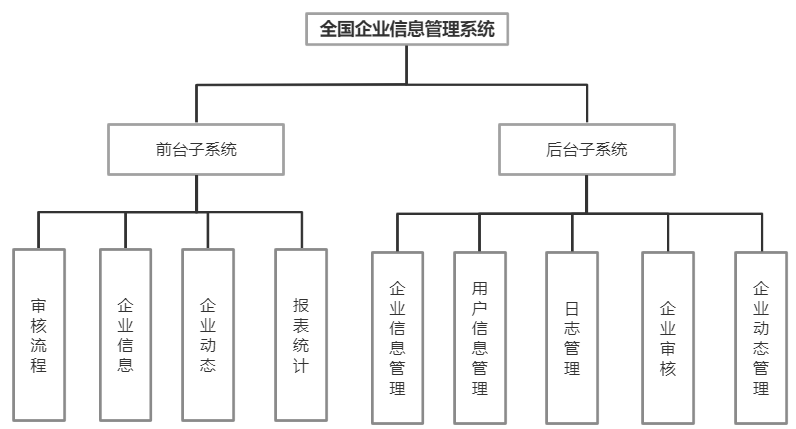


图3-2系统功能模块图

当使用本系统时，在登录界面输入账号和密码之后，系统根据数据库中的信息判断用户的角色，不同的角色进入不同的主页。在主页上可以选择进入不同的页面进行不同的操作，管理员进入系统后，可以选择企业信息管理，用户信息管理，企业动态管理，日志管理，企业审核进行相应的操作，普通用户登录界面后，可以选择企业信息，企业动态，审核流程，报表统计进行相应的操作，具体的系统流程图如图3-3所示：

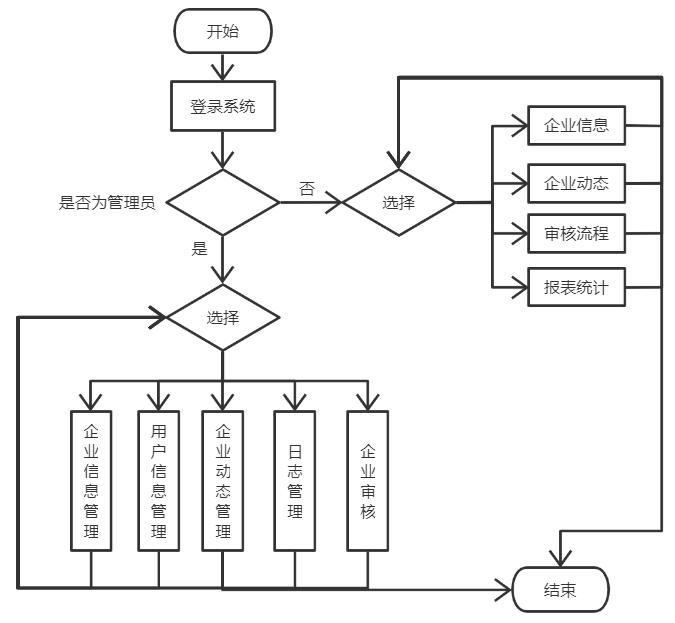


图3-3系统流程图

# 4 数据库设计

## 4.1 概念结构设计

### **4.1.1 设计思路**

本系统为一个企业信息管理网站，主要是用于管理企业信息的网站系统，首先用户登录系统必须要登录，所以就需要一个用户表，用于用户登录注册产生的用户数据，而企业信息管理网站当然首当重要的表就是企业信息表，企业信息表也是必须要有的，存储企业的详细地址，公司简介等信息，然后，企业动态的发布需要企业动态表；因为有监控用户操作的功能，所以也要有日志的信息，这就需要一个日志信息表。因为本系统分前后台来实现，普通用户和系统管理员因为角色不同在登录之后，分别有不同的登录主页面和功能，所以还需要一个角色表，然后用一个菜单表来辅助不同用户登录的菜单列表。

因此，数据库需要：用户信息表、企业信息表、企业动态表、角色表、日志信息表、菜单表。

### **4.1.2 E-R图**

系统的总体E-R图如4-1所示。

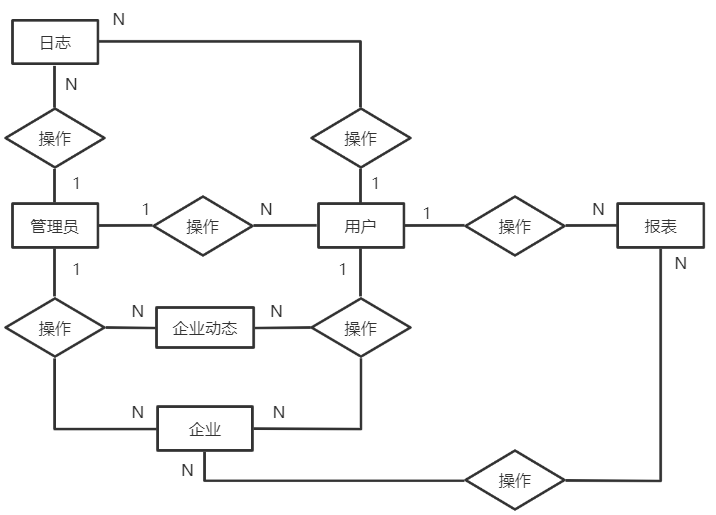


图4-1系统总体E-R图

用户实体包含的属性分别为：id，名字，账号，密码，性别，角色，电话，地址，邮箱；其E-R图如图4-2所示：

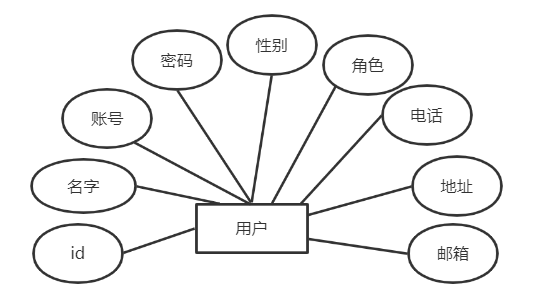


图4-2用户实体E-R图

企业实体包含的属性分别为：id，企业编码，成立日期，企业名称，经营状态，企业类型，法定代表人，电话，官网，邮箱，所属地区，详细地址，简介；其E-R图如图4-3所示：

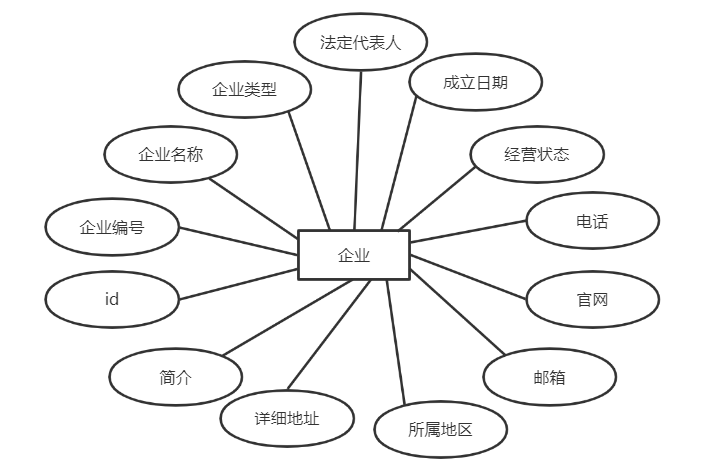


图4-3企业实体E-R图

日志实体包含的属性分别为：id，操作用户名，操作者IP，操作说明，操作时间，操作接口信息；其E-R图如图4-4所示：

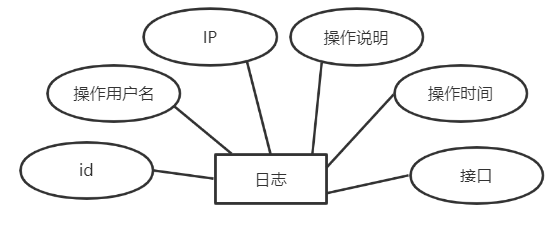


图4-4日志实体E-R图

企业动态实体包含的属性分别为：id，标题，时间，内容，相关企业；其E-R图如图4-5所示：

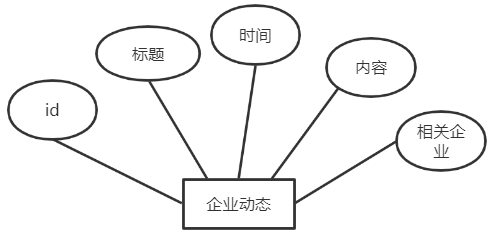


图4-5企业动态实体E-R图

菜单实体包含的属性分别为：id，菜单名，角色，路径；其E-R图如图4-6所示：

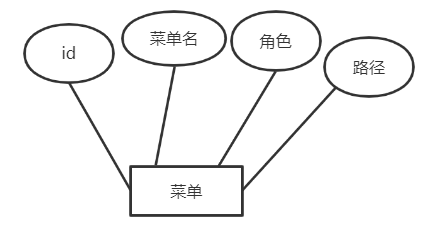


图4-6菜单实体E-R图

## 4.2 逻辑结构设计

### 4.2.1 设计思路

数据库的逻辑结构设计是将数据库概念结构设计中的E-R图转换为关系模型，然后根据具体的实体属性和限制来转化为符合要求的数据模型。

### 4.2.2 逻辑模型

在此数据库中，每个实体都设置了自增ID作为主键。具体属性如下：

1. 用户（ID，名字，账号，密码，性别，角色，电话，地址，邮箱）
2. 企业（ID，企业编码，成立日期，企业名称，经营状态，企业类型，法定代表人，电话，官网，邮箱，所属地区，详细地址，简介）
3. 日志（ID，操作用户名，操作者IP，操作说明，操作时间，操作接口信息）
4. 角色（ID，角色名）
5. 菜单（ID，菜单名，角色，路径）
6. 企业动态（ID，标题，时间，内容，相关企业）

## 4.3 物理结构设计

### 4.3.1 存取方式

本系统采用的数据库是MYSQL结构型数据库，并使用Navicate for MySQL作为辅助工具。服务器使用jdbc来将数据库连接起来,采用了长连接的方式以保证服务器能够有效的连接数据库,减少了数据丢失的可能性,也增强了数据的可读性和可写性。

### 4.3.2 存储结构

1. 企业信息表如表4-1所示：

表4-1 企业信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否主键 | 说明 |
| ent\_id | int | 10 | 是 | 企业ID |
| ent\_code | varchar | 20 | 否 | 企业编码 |
| ent\_name | varchar | 20 | 否 | 企业名称 |
| ent\_type | varchar | 10 | 否 | 企业类型 |
| ent\_own | varchar | 20 | 否 | 企业法人 |
| ent\_date | date | 0 | 否 | 成立日期 |
| ent\_state | varchar | 10 | 否 | 经营状态 |
| ent\_tel | varchar | 20 | 否 | 联系电话 |
| ent\_www | varchar | 20 | 否 | 官网 |
| ent\_email | varchar | 20 | 否 | 电子邮箱 |
| ent\_area | varchar | 20 | 否 | 所属地区 |
| ent\_adress | varchar | 50 | 否 | 企业地址 |
| ent\_content | varchar | 200 | 否 | 企业简介 |
| ent\_showid | int | 10 | 否 | 信息状态 |

1. 菜单表如表4-2所示：

表4-2 菜单表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否主键 | 说明 |
| mnue\_id | int | 10 | 是 | 菜单ID |
| mnue\_name | varchar | 10 | 否 | 菜单名 |

续表4-2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| role\_id | int | 10 | 否 | 角色ID |
| mnue\_url | varchar | 50 | 否 | 菜单路径 |

1. 角色表如表4-3所示：

表4-3 角色表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否主键 | 说明 |
| role\_id | int | 10 | 是 | 角色ID |
| role\_name | varchar | 10 | 否 | 角色名 |

1. 日志表如表4-4所示：

表4-4 日志表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否主键 | 说明 |
| id | int | 10 | 是 | 日志ID |
| user\_name | varchar | 10 | 否 | 用户名 |
| optIp | varchar | 20 | 否 | 用户IP |
| operation | varchar | 50 | 否 | 操作接口 |
| content | varchar | 50 | 否 | 操作信息 |
| create\_time | date | 0 | 否 | 操作时间 |

1. 用户表如表4-5所示：

表4-5 用户表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否主键 | 说明 |
| user\_id | int | 10 | 是 | 用户ID |
| user\_name | varchar | 10 | 否 | 用户名 |
| user\_account | varchar | 10 | 否 | 账号 |
| user\_pwd | varchar | 20 | 否 | 密码 |
| role\_id | int | 10 | 否 | 角色ID |
| user\_showid | int | 10 | 否 | 信息状态 |
| user\_tel | varchar | 20 | 否 | 联系电话 |
| user\_adress | varchar | 20 | 否 | 地址 |
| user\_email | varchar | 20 | 否 | 电子邮箱 |

1. 企业动态表如表4-6所示：

表4-6 企业动态表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否主键 | 说明 |
| id | int | 10 | 是 | 动态ID |
| title | varchar | 50 | 否 | 动态标题 |
| time | date | 0 | 否 | 发布日期 |
| content | varchar | 2000 | 否 | 动态内容 |
| enterprise | int | 10 | 否 | 企业id |

5 界面设计

## 5.1 界面关系图

界面分为前台用户界面和后台管理员管理界面：

普通用户登录的前台界面：注册界面，登录界面，主页，个人信息界面，企业查询页面，企业录入页面，报表统计页面，审核流程页面，企业详细信息页面。如图 5-1 所示：

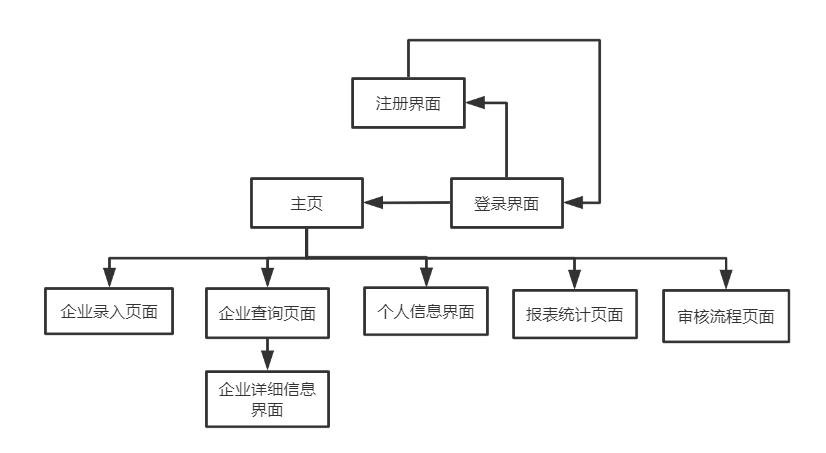


图5-1前台界面关系图

系统管理员登录的后台界面：登录界面，管理员登录主页，用户信息管理界面，企业审核界面，日志管理界面，企业信息管理界面，企业录入界面，新增（修改）用户信息界面，新增（修改）企业信息界面。如图 5-2所示：

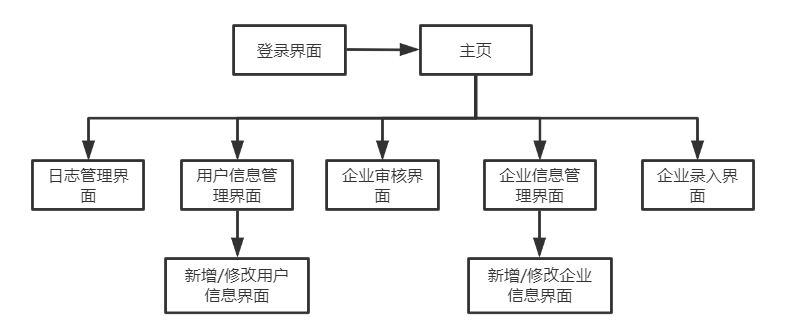


图 5-2 后台界面关系图

## 5.2 界面设计成果

### **5.2.1 主界面**

系统管理员登录系统后的主页面如下图5-3所示，页面提供给管理员对日志，用户，企业，企业动态等信息的管理接口，管理员可以通过这些接口进入相应的管理界面。

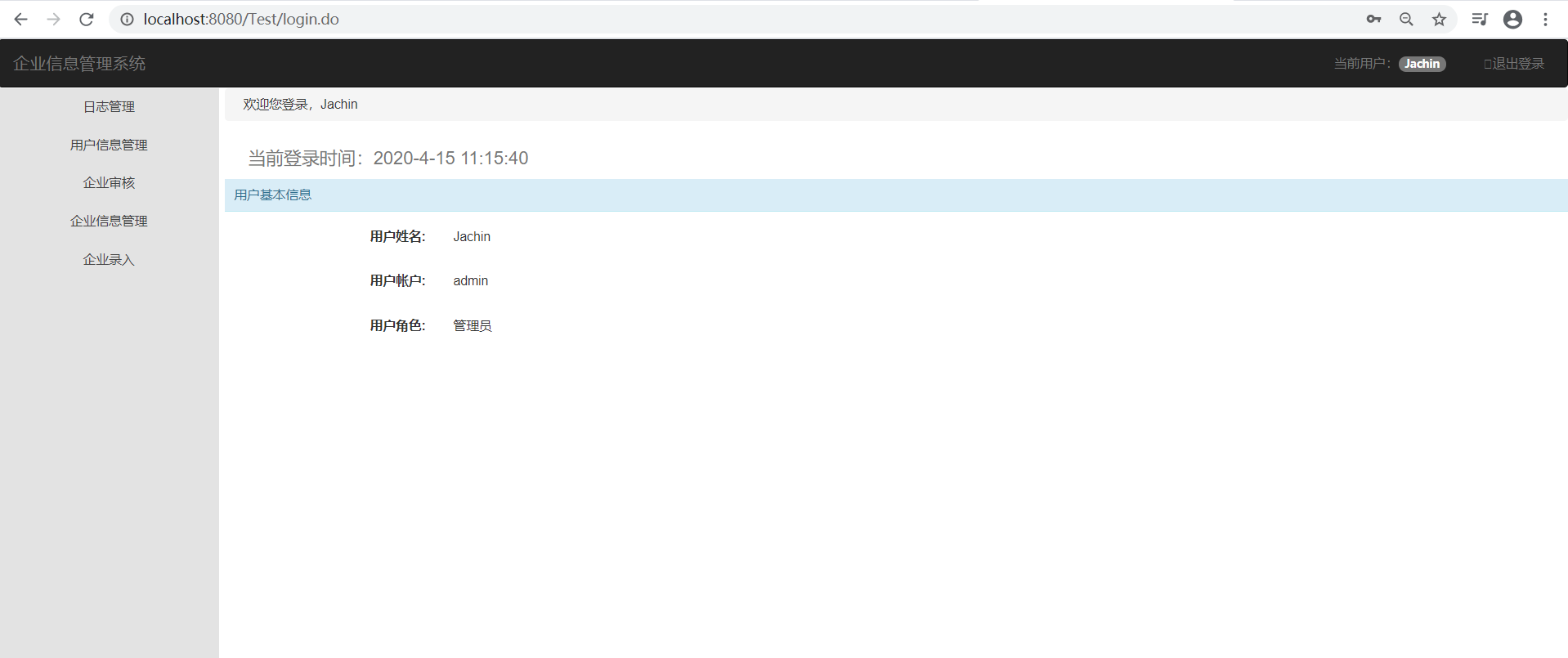


图 5-3 系统管理员主界面

普通用户登录系统后的主页面如下图5-4所示，页面提供个人信息查看，企业查询，企业录入，统计报表查看，审核流程查看，企业动态等接口，用户可以通过这些接口来使用相应的功能。

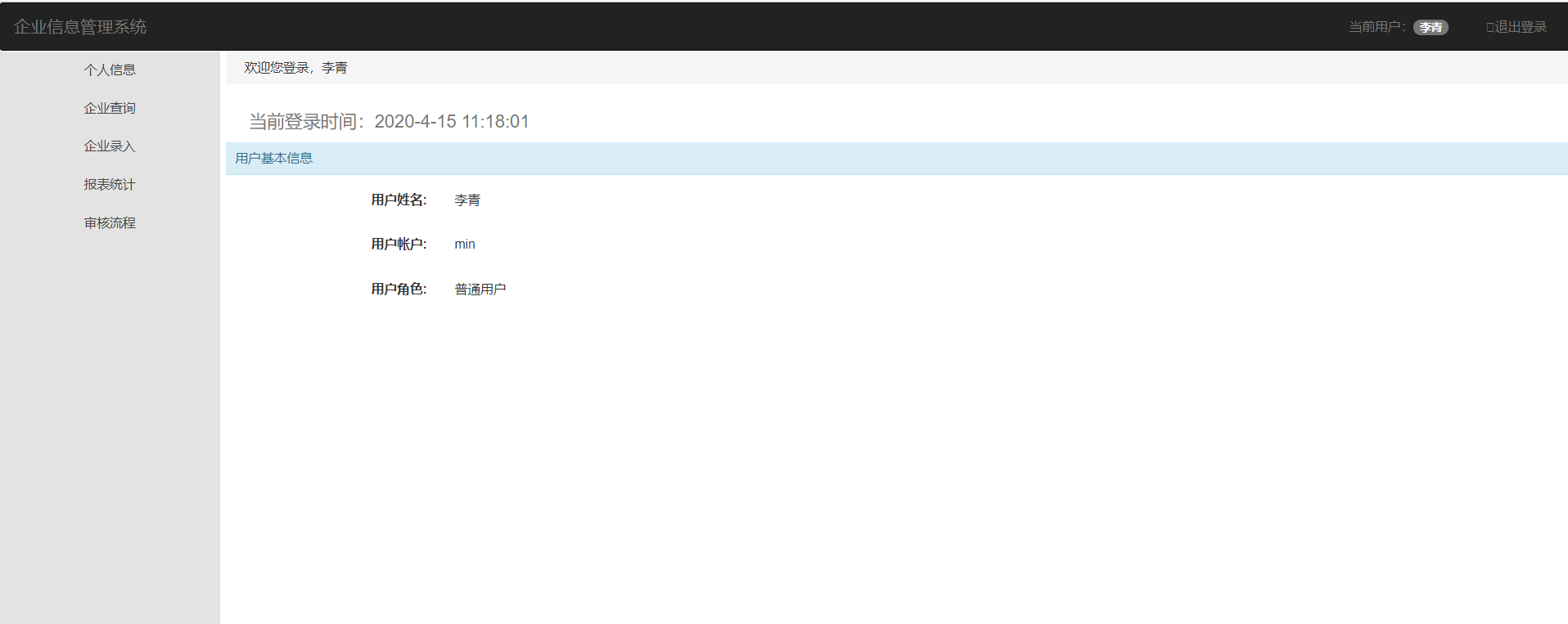


图 5-4 系统用户主界面

### **5.2.2 子界面**

系统的登陆注册界面如图5-5所示，页面提供用户登录，游客登录，用户注册功能，不同的角色登录系统进入不同的主页。



图 5-5 系统登录界面

管理员的日志信息管理界面如图5-6所示，管理员可以在页面上查看所有系统用户的操作日志信息。



图 5-6 日志信息管理界面

管理员的用户信息管理界面如图5-7所示，页面提供给管理员查询用户，添加用户，修改用户和删除用户的接口，管理员通过这些接口来实现对用户信息的管理。

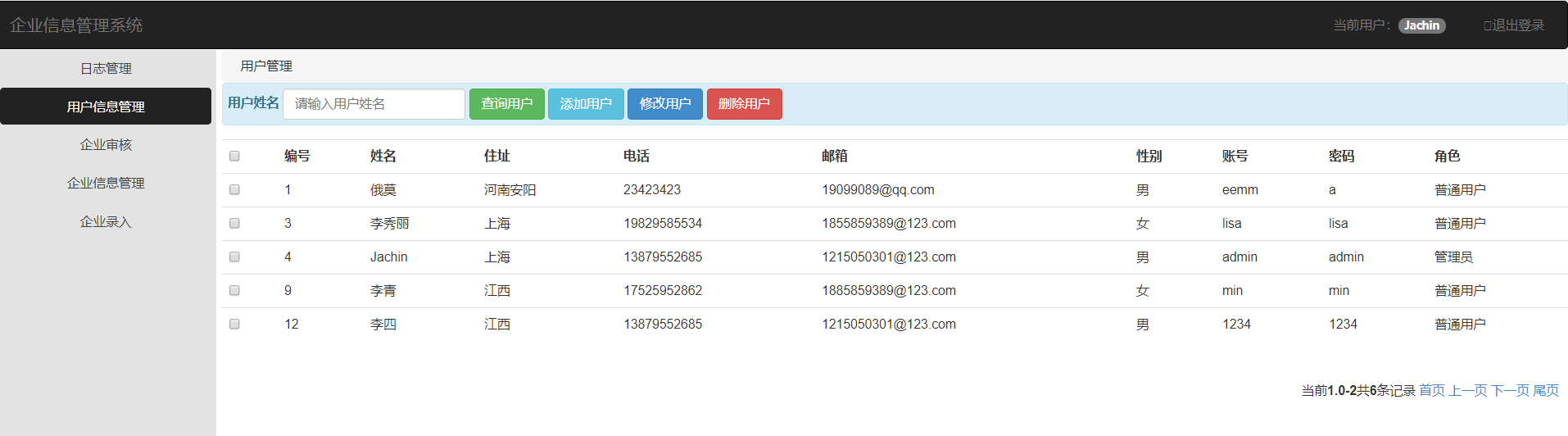


图 5-7 用户信息管理界面

管理员的企业审核界面如图5-8所示，管理员在本界面可以对用户录入的企业信息进行审核。

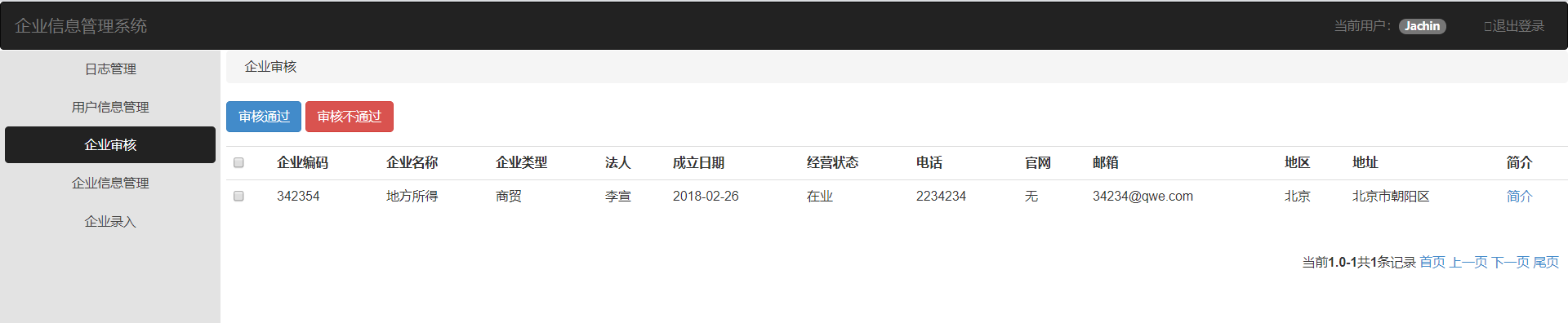


图 5-8 企业审核界面

管理员的企业信息管理界面如图5-9所示，本界面提供给管理员查询，修改，删除的接口，管理员可通过这些接口来实现对企业信息的管理。



图 5-9 企业信息管理界面

用户的个人信息界面如图5-10所示，本界面可以让登录用户看到自己的个人信息，并提供修改个人信息的接口。



图 5-10 用户个人信息界面

用户的企业信息查询界面如图5-11所示，本界面提供给用户查询企业，导出企业信息的接口，用户可以通过输入关键字实现对企业的查询，还可以将企业信息导出为excel。



图 5-11企业信息查询界面

用户的企业信息录入界面如图5-12所示，用户在本界面可以实现企业信息的录入，系统提供两种方式录入企业信息，一种是表单输入，一种是导入文件。



图 5-12企业信息录入界面

用户的报表统计界面如图5-13所示，本界面提供给用户以企业类型和企业地区为标准生成的统计报表。



图 5-13报表统计界面

用户的企业信息审核状态查看界面如图5-14所示，本页面提供用户查看审核中和审核不通过的企业信息。



图 5-14企业信息审核状态查看界面

6 详细设计

## 6.1 系统主要功能模块介绍

本系统的主要功能模块包括：

（1）用户管理：

用户管理模块具体功能主要包括管理员对系统中的用户信息进行管理；管理员可以对系统中的用户信息进行新增，修改，查看，删除等操作。

（2）企业动态管理：

企业动态是各个企业发生的重要信息，可供用户查看。企业动态管理的主要功能就在于系统管理员对前台显示的企业动态信息进行管理。管理员可对企业动态信息进行新增企业动态，查看企业动态，修改企业动态和删除企业动态等操作。

（3）企业管理：

本系统的核心功能就是给用户提供查询企业信息的渠道，而除了查询企业信息以外，企业管理功能还提供给系统管理员管理企业信息的权限。管理员可以对企业信息进行新增企业信，查找企业信息，修改企业信息，删除企业信息和审核企业信息等操作，以此来实现对系统中的企业信息进行不断的更新和维护。

（4）统计报表：

统计报表功能是提供给用户查看企业分布情况的，统计报表中的所有图形都是系统根据数据库中所有的企业进行分析后自动生成的。

（5）日志管理：

日志管理模块主要是给系统管理员提供的，用户使用本系统时，会产生相应的日志信息，管理员可以查看所有的日志信息。从而达到对系统进行监控的目的。

## 6.2企业动态

### 6.2.1功能结构设计

“企业动态”功能主要包含管理员对企业动态信息的管理，其中包括企业动态信息的发布，修改，查询和删除。企业动态功能的功能结构图如图6-1所示。

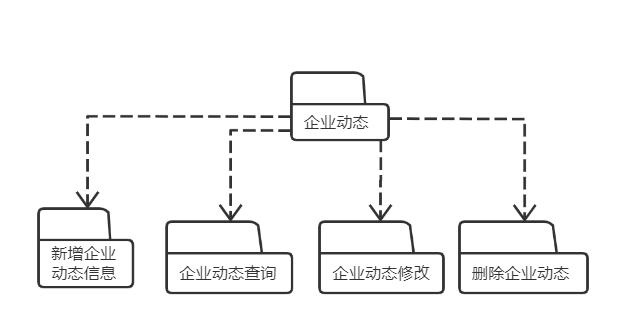


图 6-1企业动态功能结构图

### 6.2.2类图设计

企业动态的类图设计如图6-2所示：

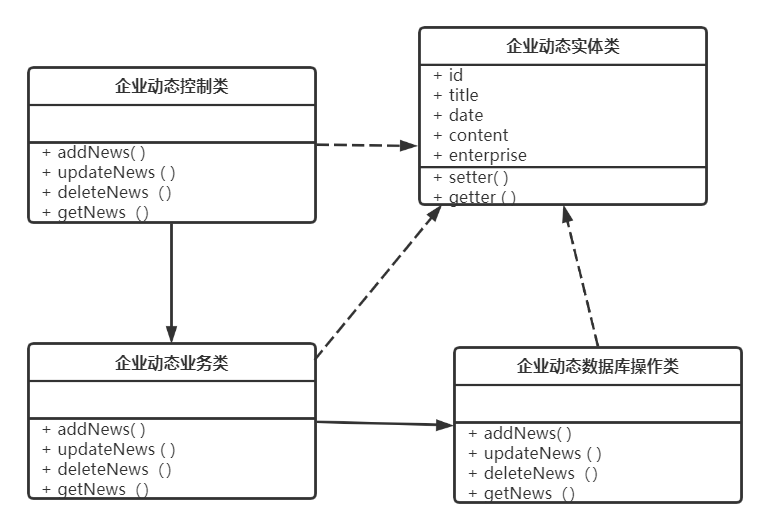


图 6-2企业动态类图

企业动态控制类的描述如下表6-1所示：

表6-1企业动态控制类描述

|  |  |  |
| --- | --- | --- |
| 企业动态控制类 | | |
| 类的描述：实现管理员对企业动态信息的各种操作。 | | |
| 操作描述 | addNews( ) | 添加企业动态信息 |
| updateNews ( ) | 更新企业动态信息 |
| deleteNews（） | 删除企业动态信息 |
| getNews（） | 获取企业动态信息 |

企业动态实体类的描述如下表6-2所示：

表6-2企业动态实体类描述

|  |  |  |
| --- | --- | --- |
| 企业动态实体类 | | |
| 类的描述：企业动态实体类 | | |
| 属  性  描  述 | Id | 企业动态Id |
| Title | 企业动态的标题 |
| Time | 企业动态信息的发布日期 |
| Content | 企业动态信息的内容 |
| enterprise | 相关企业 |
| 操作描述 | getter() | 获取企业动态信息 |
| setter() | 设置企业动态信息 |

### 6.2.3顺序图设计

发布企业动态的顺序图如图6-3所示：

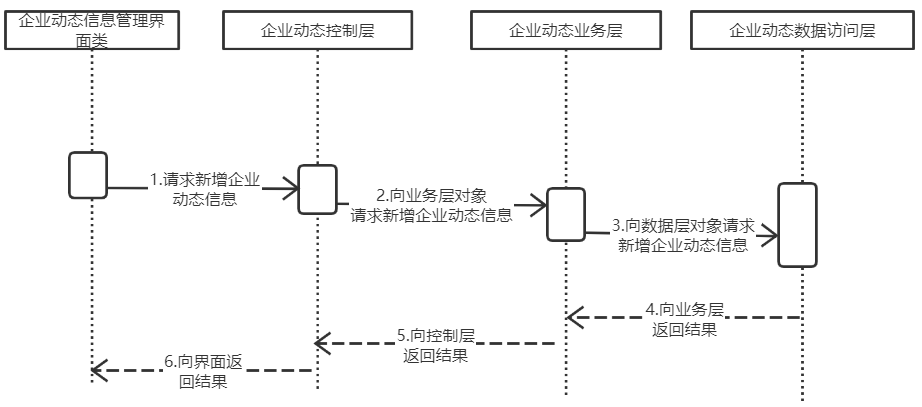


图 6-3发布企业动态顺序图

### 6.2.4核心处理流程设计

发布企业动态的主要流程如图6-4所示：

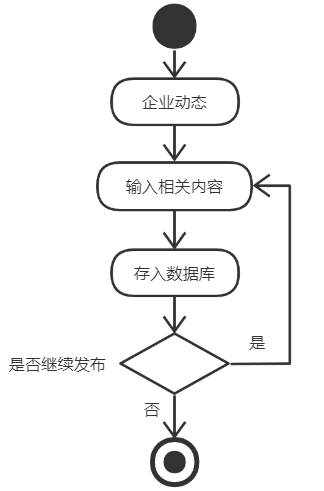


图 6-4发布企业动态活动图

## 6.3用户管理

### 6.3.1功能结构设计

“用户管理”功能主要包含管理员对系统用户的信息进行管理的功能；其中主要是对系统用户的信息进行新增用户信息，修改用户信息，查询用户信息，删除用户信息等操作。用户管理功能模块图如图6-5所示。

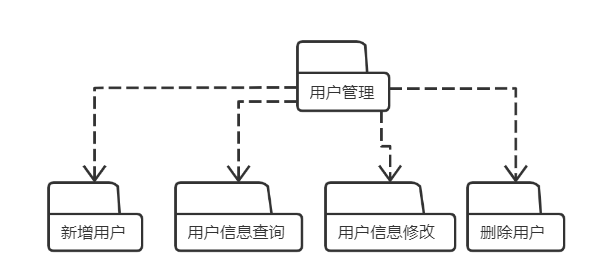


图 6-5用户管理功能结构图

### 6.3.2类图设计

用户管理的类图设计如图6-6所示：

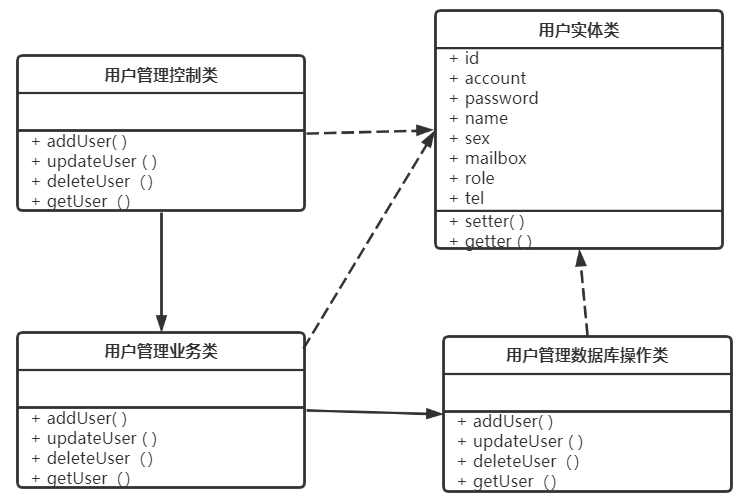


图 6-6用户管理类图

用户管理控制类的描述如表6-3所示：

表6-3用户管理控制类描述

|  |  |  |
| --- | --- | --- |
| 用户管理控制类 | | |
| 类的描述：负责响应系统管理员的各种操作。 | | |
| 操作描述 | addUser( ) | 添加用户信息到数据库 |
| updateUser ( ) | 更新数据库中的用户信息 |
| deleteUser（） | 删除用户信息 |
| getUser（） | 从数据库中获取用户信息 |

用户实体类的描述如表6-4所示：

表6-4用户实体类描述

|  |  |  |
| --- | --- | --- |
| 用户实体类 | | |
| 类的描述：用户实体类 | | |
| 属性  描述 | Id | 用户Id |
| Name | 用户姓名 |
|  | Account | 系统账号 |
| Password | 账号密码 |
| Sex | 性别 |
| Mailbox | 邮箱 |
| Tel | 联系电话 |

续表6-4

|  |  |  |
| --- | --- | --- |
|  | Role | 账号角色 |
| 操作描述 | getter() | 获取用户实体类的各个属性信息 |
| setter() | 设置属性信息 |

### 6.3.3顺序图设计

管理员可以通过本系统对用户的信息进行新增，修改，删除，查询操作，图6-7是管理员修改用户信息的顺序图：

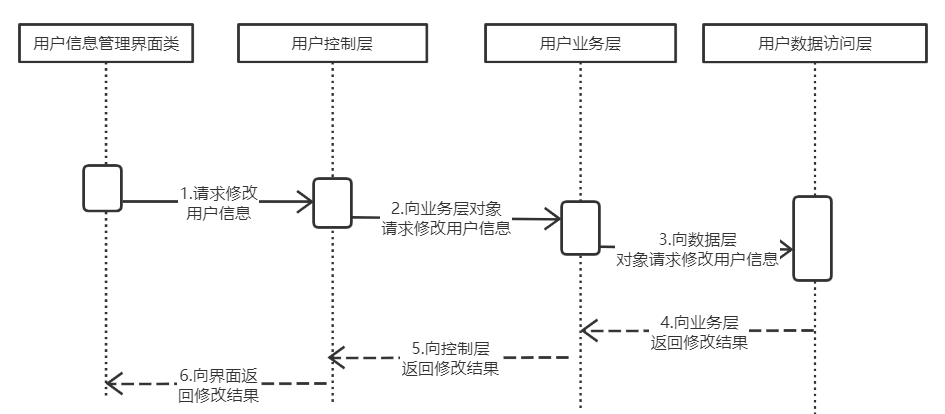


图 6-7修改用户信息顺序图

### 6.3.4核心处理流程设计

图6-8是管理员对用户信息进行修改的主要流程设计。

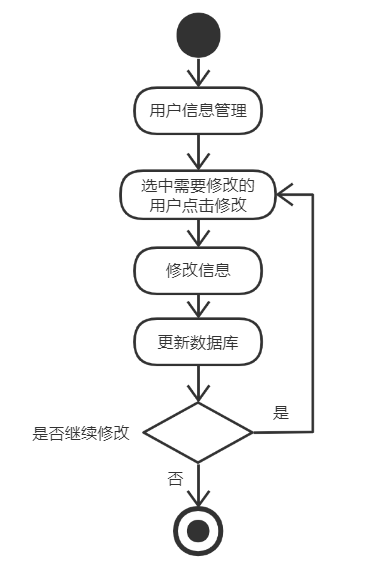


图 6-8修改用户信息活动图

## 6.4企业管理

### 6.4.1功能结构设计

“企业管理”功能主要目的是为管理员提供对企业信息进行管理的功能；其中主要是对企业的信息进行新增企业信息操作，修改企业信息操作，查询企业信息操作，删除企业信息操作。企业信息管理功能模块图如图6-9所示。

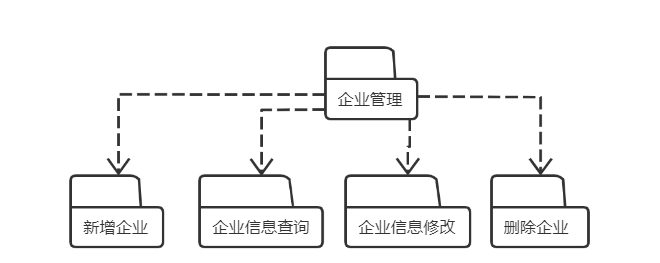


图 6-9企业管理功能结构图

### 6.4.2类图设计

企业信息管理的类图设计如图6-10所示：

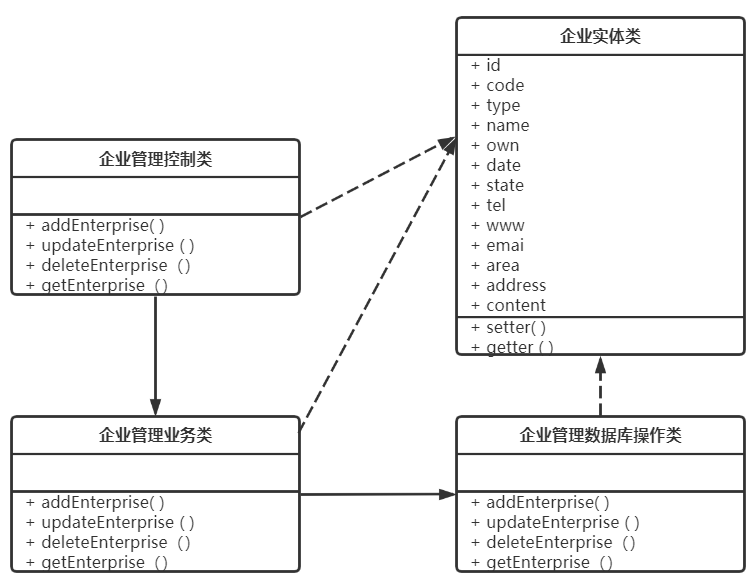


图 6-10企业管理类图

企业管理控制类的描述如表6-5所示：

表6-5企业管理控制类描述

|  |  |  |
| --- | --- | --- |
| 企业管理控制类 | | |
| 类的描述：负责响应系统管理员对企业信息进行的各种操作。 | | |
| 操作描述 | addEnterprise( ) | 添加企业信息到数据库 |
| updateEnterprise ( ) | 更新数据库中的企业信息 |
| deleteEnterprise（） | 删除企业信息 |
| getEnterprise（） | 从数据库中获取企业信息 |

企业实体类的描述如表6-6所示：

表6-6企业实体类描述

|  |  |  |
| --- | --- | --- |
| 企业实体类 | | |
| 类的描述：企业实体类 | | |
| 属性  描述 | Id | 企业Id |
| Code | 企业编码 |
| Name | 企业名称 |
| Type | 企业类型 |
| Own | 企业法定代表人 |
| Date | 成立日期 |
| State | 经营状态 |
| Tel | 企业联系电话 |
| Www | 企业官网 |
| Email | 企业邮箱 |
| Area | 所在地区 |
| Address | 详细地址 |
| Content | 企业简介 |
| 操作描述 | getter() | 获取企业实体类的各个属性信息 |
| setter() | 设置属性信息 |

### 6.4.3顺序图设计

管理员可以对企业信息进行新增企业信息操作，修改企业信息操作，删除企业信息操作，查询企业信息操作，图6-11是管理员删除企业信息的顺序图：

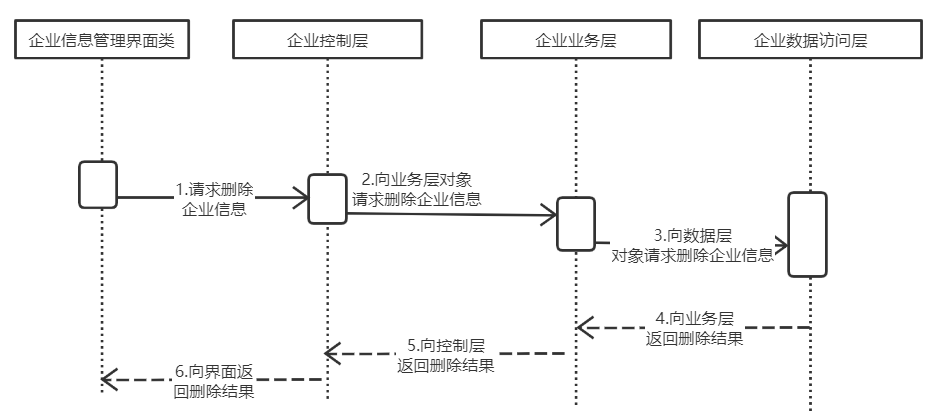


图 6-11删除企业信息顺序图

### 6.4.4核心处理流程设计

图6-12是管理员对企业信息进行删除的主要流程设计。

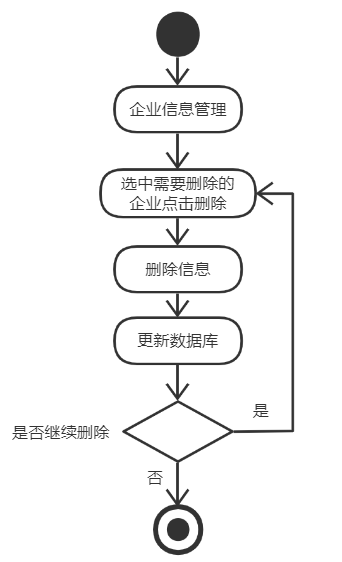


图 6-12删除企业信息活动图

## 6.5日志管理

### 6.5.1功能结构设计

“日志管理”是根据用户通过系统进行操作之后，系统将其记录下来以便让系统管理员进行查看的功能。其功能模块的设计如图6-13所示。

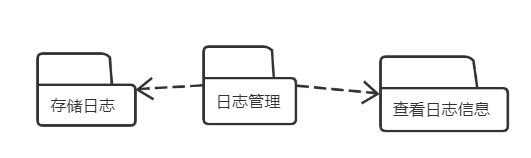


图 6-13日志管理功能结构图

### 6.5.2类图设计

日志管理的类图设计如图6-14所示：

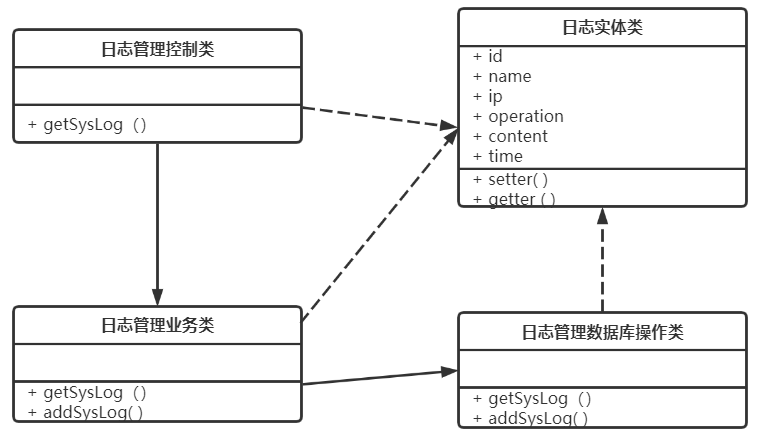


图 6-14日志管理类图

日志管理业务类描述如表6-7所示：

表6-7日志管理业务类描述

|  |  |  |
| --- | --- | --- |
| 日志管理业务类 | | |
| 类的描述：负责响应系统管理员的各种操作。 | | |
|  | getSysLog（） | 从数据库中获取日志信息 |
| addSysLog( ) | 向数据库中存入日志信息 |

日志信息类的描述如表6-8所示：

表6-8日志信息类描述

|  |  |  |
| --- | --- | --- |
| 日志信息类 | | |
| 类的描述：日志信息实体类 | | |
| 属性  描述 | Id | 日志Id |
| Name | 操作用户名 |
| Ip | 用户IP地址 |
| Operation | 操作接口 |
| Content | 操作信息 |
| Time | 操作时间 |
| 操作描述 | getter() | 获取日志实体类的各个属性信息 |
| setter() | 设置属性信息 |

### 6.5.3顺序图设计

管理员可以在该模块查看所有日志信息，图6-15是用管理员对日志信息进行查看的顺序图：

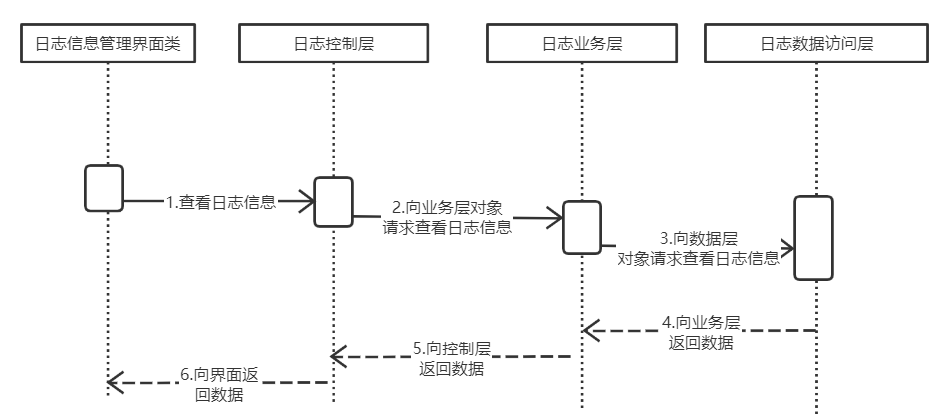


图 6-15查看日志信息顺序图

### 6.5.4核心处理流程设计

图6-16是管理员对日志信息进行查看的主要流程设计。

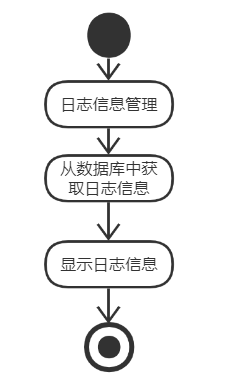


图 6-16查看日志信息活动图

## 6.6统计报表

### 6.6.1功能结构设计

“统计报表”是根据数据库中所有的企业信息来想用户展示企业所属地区分布情况和企业类型比例。其功能结构图如图6-17所示。

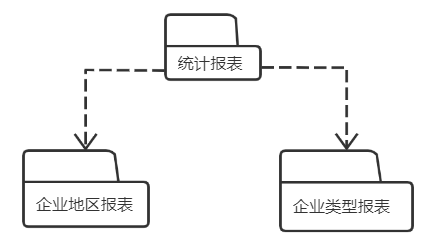


图 6-17统计报表功能结构图

### 6.6.2类图设计

统计报表的类图设计如图6-18所示：

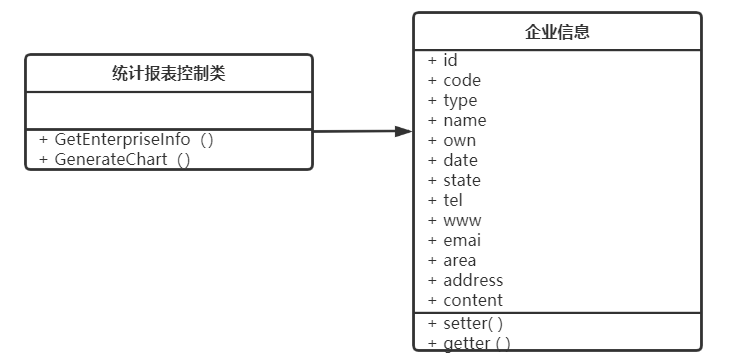


图 6-18统计报表类图

统计报表控制类的描述如表6-9所示：

表6-9统计报表控制类描述

|  |  |  |
| --- | --- | --- |
| 统计报表控制类 | | |
| 类的描述：负责向用户展示基于企业信息的各种报表。 | | |
| 操作描述 | GetEnterpriseInfo( ) | 获取所有的企业信息，用于形成报表 |
| GenerateChart ( ) | 形成报表 |

### 6.6.3顺序图设计

图6-19为用户查看统计报表的顺序图。

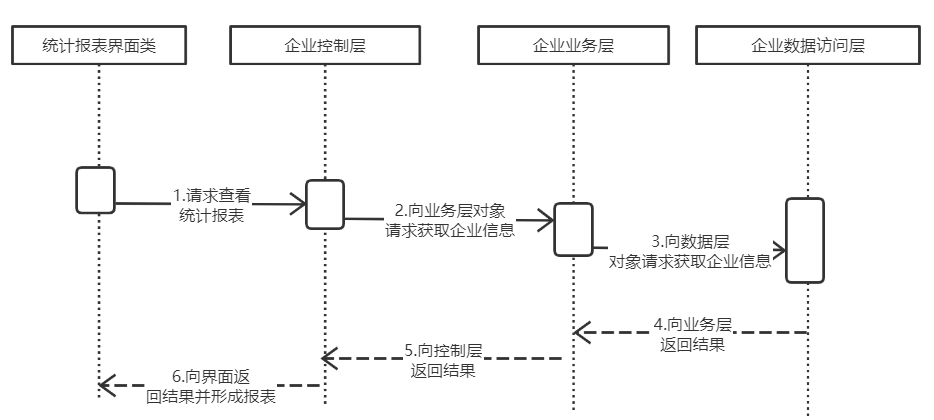


图 6-19查看统计报表顺序图

### 6.6.4核心处理流程设计

图6-20为报表统计的主要流程设计。

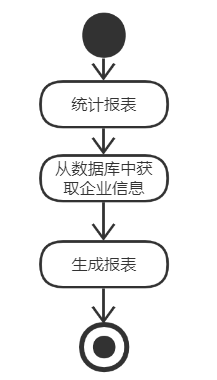
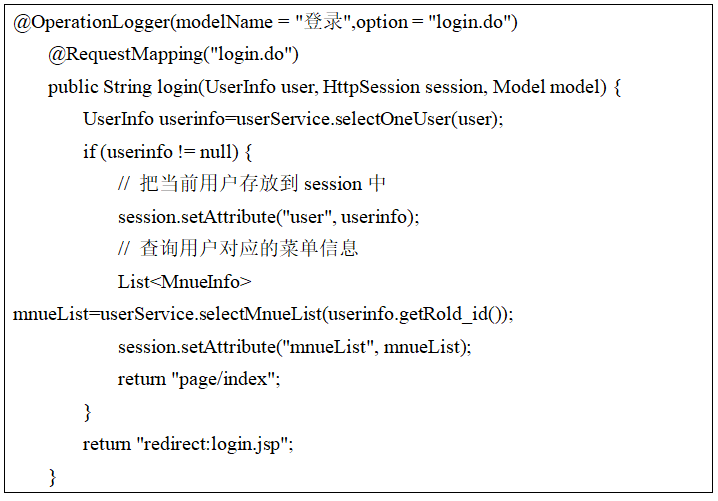


图 6-20查看统计报表活动图

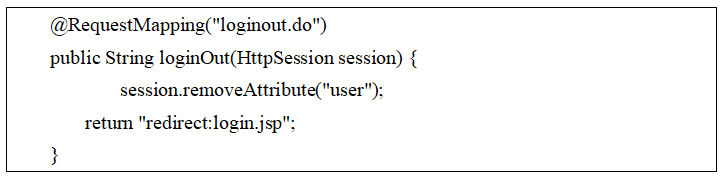
7 编码

## 7.1 代码实现与核心算法

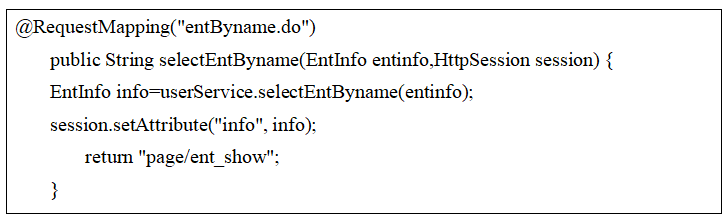
1. 用户登录系统时需要存储其信息来记录某些操作的信息，以下是用户登录系统时利用session来存储用户信息的主要代码：



1. 为避免浪费资源，用户在退出系统时应清除session信息，其主要代码如下：



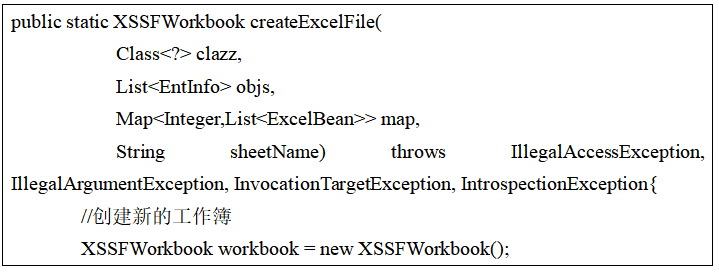
1. 用户在企业信息界面输入企业名查询企业信息的主要代码如下：

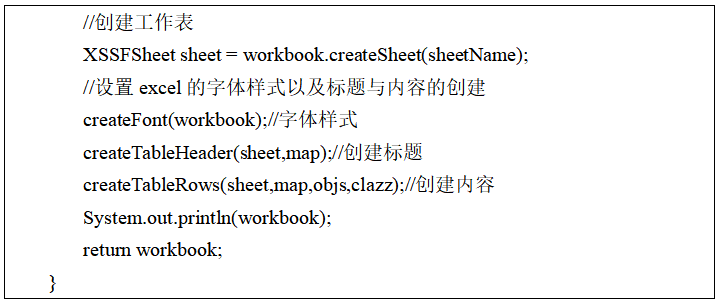


1. 用户可以选择将企业信息导出为Excel表格以便浏览，其主要代码如下：



其中，利用Apache POI插件来实现Excel表格的创建的具体方法如下：





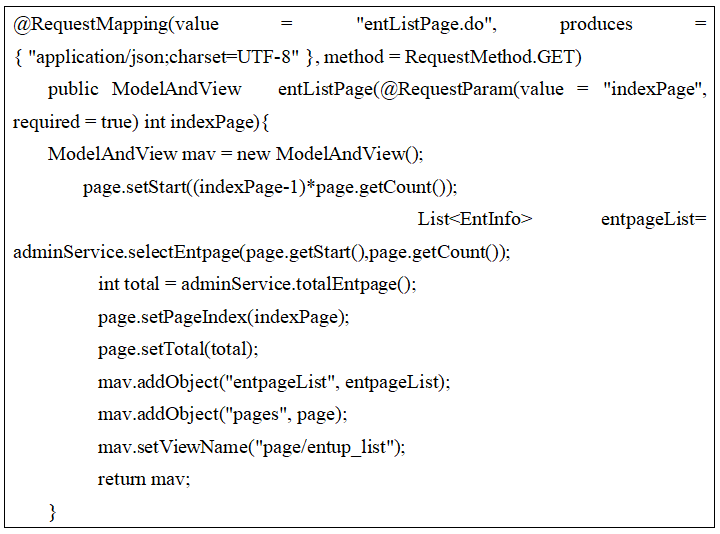
1. 系统根据数据库中的企业信息，利用jfree插件生成的统计报表功能的主要代码如下：



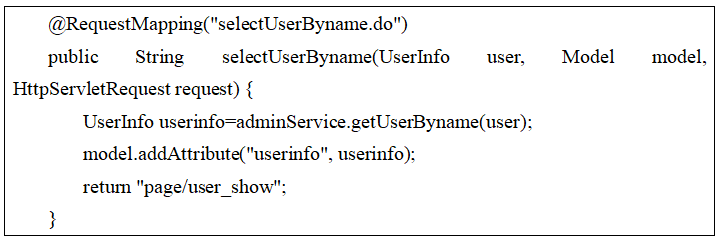




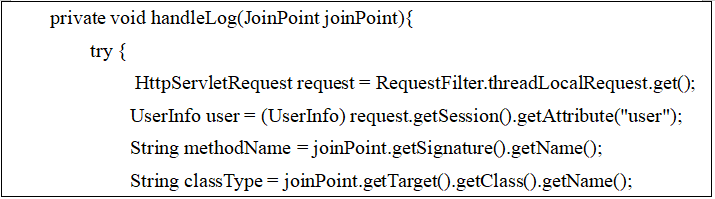
1. 系统管理员在企业信息管理界面显示企业信息的主要代码如下：

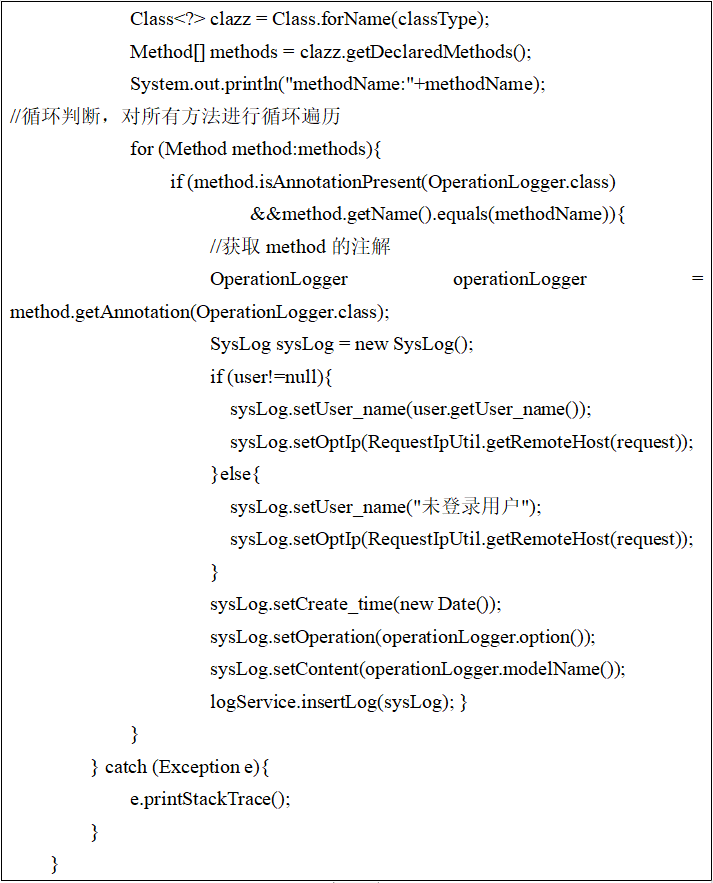


1. 系统管理员在用户信息管理界面查询用户信息的主要代码如下：



1. 系统将用户进行的某些操作信息记录下来形成日志信息并存储在数据库中，使系统管理员可以在日志信息管理界面查看日志信息的主要代码如下：





## 7.2 代码优化分析

1. 在代码模块化和接口开发之后，出现了许多规范代码。同时，注释也是模板化的。

请求输入、响应和输出参数都继承并实现了相应的基类，这给了人们一种整洁和专业化的感觉。与此同时，代码重复也在增加。在优化的后期，从添加基类、父类和接口的角度进一步提高代码的重用率。同时，应该提取前端的公共请求方法，以避免使用不同的请求参数重写请求方法，并且在每个请求上只使用请求接口。

1. 应提高代码的注释量与注释的表述方式的专业程度，提高代码的可读性。
2. 公共参数与标识等固定值应该要抽离出来，放在公共模块，用枚举或常量的方式呈现，

避免写错或者下次引用的时候不清楚其的具体属性。

8 测试

## 8.1 测试方案设计

### 8.1.1 测试策略

为保障软件的质量与系统的可靠性，应该尽全力检测各个方面。本测试为了保证测试的效果，全部为手动测试，利用错误推断法的方法，对系统中开发人员认为可能出现的错误进行测试，对每个功能模块逐一选择测试用例进行测试，并将其记录下来以便查看。

### 8.1.2 测试进度安排

在每个功能模块一完成后就马上进行功能测试，每个模块完成时进行功能测试与UI测试，项目完成后进行验收测试。测试进度安排表如表8-1所示：

表8-1测试进度安排表

|  |  |  |
| --- | --- | --- |
| 测试任务 | 用时（天） | 执行人员 |
| 制定测试计划 | 1 |  |
| 登录功能测试 | 1 |  |
| 用户管理功能测试 | 1 |  |
| 企业动态管理功能测试 | 2 |  |
| 企业信息管理功能测试 | 2 |  |
| 日志管理功能测试 | 2 |  |
| 报表统计功能测试 | 1 |  |
| 系统总体测试 | 3 |  |

### 8.1.3 测试资源

因测试资源有限，测试人员就是本系统的开发人员，测试工具则为Google Chrome，Microsoft Edge等较为普遍的浏览器。

### 8.1.4 关键测试点

1. 用户登录测试
2. 用户查看企业信息测试
3. 用户上传企业信息测试
4. 用户查看企业动态测试
5. 管理员登录测试
6. 管理员管理信息测试

## 8.2 测试用例构建

### 8.2.1 测试用例编写约定

1. 在划分功能或过程时，必须简单明了。测试用例只检查一个功能点或流程。否则，测试用例将被混淆并降低可读性。
2. 测试用例的步骤描述应该一步一步地简单明了。这有助于提高用例的可操作性。
3. 描述要清晰，包括具体的场合、对象和术语，不要有模糊的概念和笼统的描述。尽量避免不确定的词语。
4. 测试用例应当全面，包括一些会使系统出现异常的用例。
5. 对于无输入的操作，应该详细描述其具体的操作步骤和结果。

### 8.2.2 测试用例设计

测试用例的编写必须包括测试用例编号、用例名称、前置条件、操作步骤、预期结果、实际结果、问题描述、测试结果与结论、测试执行者这几项，此次测试围绕下面几个方面：各个不同角色的用户登录、用户录入企业信息、用户修改自己的信息、查看企业信息的报表统计、管理员对各种信息的管理等。详见表8-2。

表 8-2测试用例表

|  |  |  |
| --- | --- | --- |
| 序号 | 测试模块 | 内容 |
| 1 | 用户管理 | 用户的登录注册，用户的信息管理等 |
| 2 | 企业动态管理 | 用户查看企业动态信息，管理员管理企业的动态信息 |
| 3 | 企业信息管理 | 用户查看，录入企业信息，管理员管理企业信息。 |
| 4 | 日志管理 | 查看日志信息 |
| 5 | 统计报表 | 查看企业的数据报表 |

### 8.2.3 关键测试用例

表8-3为用户登录测试用例：

表 8-3用户登录测试用例

|  |  |
| --- | --- |
| 用例编号 | CS 1-1 |
| 用例名称 | 用户登录 |
| 前置条件 | 有本系统有效的账号 |
| 操作步骤 | 1. 在文本框中输入账号和密码。 2. 点击登录按钮。 |
| 预期结果 | 登录成功，并跳转到主页面。 |
| 实际结果 | 与预期结果相同 |
| 问题描述 | 暂无 |
| 测试结果与结论 | 测试通过 |
| 测试执行者 |  |

表8-4为企业查询测试用例：

表 8-4企业查询测试用例

|  |  |
| --- | --- |
| 用例编号 | CS 1-2 |
| 用例名称 | 企业查询 |
| 前置条件 | 成功登录本系统 |
| 操作步骤 | 1. 在企业信息页面的搜索框内输入企业名称。 2. 点击查询按钮。 |
| 预期结果 | 页面显示查询到的企业信息。 |
| 实际结果 | 与预期结果相同 |

续表8-4

|  |  |
| --- | --- |
| 问题描述 | 暂无 |
| 测试结果与结论 | 测试通过 |
| 测试执行者 |  |

表8-5为企业录入测试用例：

表 8-5企业录入测试用例

|  |  |
| --- | --- |
| 用例编号 | CS 1-3 |
| 用例名称 | 企业录入 |
| 前置条件 | 成功登录本系统 |
| 操作步骤 | 1. 在企业录入页面输入企业基本信息。 2. 点击添加按钮。 |
| 预期结果 | 添加成功。 |
| 实际结果 | 与预期结果相同 |
| 问题描述 | 暂无 |
| 测试结果与结论 | 测试通过 |
| 测试执行者 |  |

表8-6为企业动态测试用例：

表 8-6企业动态测试用例

|  |  |
| --- | --- |
| 用例编号 | CS 1-4 |
| 用例名称 | 企业动态 |
| 前置条件 | 成功登录本系统 |
| 操作步骤 | 1. 在企业动态信息页面点击一条企业动态信息。 |
| 预期结果 | 显示该动态的详情信息。 |
| 实际结果 | 与预期结果相同 |
| 问题描述 | 暂无 |
| 测试结果与结论 | 测试通过 |
| 测试执行者 |  |

表8-7为日志管理测试用例：

表 8-7日志管理测试用例

|  |  |
| --- | --- |
| 用例编号 | CS 1-5 |
| 用例名称 | 日志管理 |
| 前置条件 | 成功登录本系统 |
| 操作步骤 | 1.系统管理员主页点击日志管理。 |
| 预期结果 | 显示所有的日志信息。 |
| 实际结果 | 与预期结果相同 |
| 问题描述 | 暂无 |
| 测试结果与结论 | 测试通过 |
| 测试执行者 |  |

表8-8为统计报表测试用例：

表 8-8统计报表测试用例

|  |  |
| --- | --- |
| 用例编号 | CS 1-6 |
| 用例名称 | 统计报表 |
| 前置条件 | 成功登录本系统 |
| 操作步骤 | 1.用户登录后的主界面点击统计报表。 |
| 预期结果 | 显示根据企业信息形成的统计图。 |
| 实际结果 | 与预期结果相同 |
| 问题描述 | 暂无 |
| 测试结果与结论 | 测试通过 |
| 测试执行者 |  |

### 8.2.4 测试用例维护

1. 若在后期增加或修改了相应的功能点，则应该相应的新增或修改测试用例；例如：在企业信息管理模块中，录入企业信息从原本的仅支持页面表单录入到现在的也支持Excel文件录入，相应的就需要增加Excel文件录入企业信息的测试用例。
2. 必要时应删除因新功能增加而多余的用例。随着系统不断地优化，系统的功能逐渐增加，很多新的功能在实现后所设计的测试用例包含了之前设计的测试用例，而以前的测试用例就显得很多余，应该将这些多余的测试用例删除。

9 总结与展望

## 9.1 设计工作总结

本系统主要通过idea开发工具进行开发，利用java语言、SSM框架、Bootstrap、Ajax等技术进行实现。具有对庞大的企业信息进行管理的功能，能实现用户在庞大的数据中进行精确查询的功能，本系统也不仅仅只是局限于查询企业信息这一种功能，本系统还具有企业统计报表展示的功能，还支持查看企业动态的功能，系统还会记录用户的各种操作形成日志信息存储在数据库中，其中用到了AOP（面向切面编程）来实现，并且本系统的后台支持管理员对所有的信息进行统一管理，值得一提的是，本系统支持文件上传来录入企业信息，其中用到了POI插件，Apache POI是Apache软件基金会的开源项目，POI提供API给Java程序对Microsoft Office格式档案读和写的功能。

## 9.2 未来工作展望

由于时间和能力的关系，本系统还有许多的问题和缺陷有待完善。要想让本系统成为真正的全国企业信息管理系统，还有很多事情要做，例如，可以优化前台信息展示的页面，让页面更加美观。除此之外，本系统的功能模块还比较少，不能完全满足人们的需求，需要增加一些也同样实用的模块。一个系统在经过使用后，往往会产生更多的需求，也会暴露很多的问题，这需要一段时间来将其进行完善。

谢 辞

参考文献

[1] 罗如为.Java Web开发技术与项目实战[J].水利水电出版社,2019.

[2] 马建红.JSP应用与开发技术.第三版[J].清华大学出版社,2018.

[3] 明日科技.Java从入门到精通.第五版[J].清华大学出版社,2019.

[4] 李刚.疯狂Java讲义.第四版[J].电子工业出版社,2018.

[5] 尤鸿霞.基于HTML5的企业微网站建设[M].电脑知识与技术,2018.

[6] 王勇,王松,张红英.基于B/S构架的网络结构可视化系统设计与实现[M].计算机工程与应用,2020.

[7] 黄子轩.计算机软件工程的维护措施和方法[J].电子技术与软件工程,2018.

[8] 高惠勒.计算机软件工程的现代化技术研究[J].中国新通信,2018.

[9] 林树青.大学计算机基础教学的新思路探讨[J].当代教育实践与教学研究,2020(08):84-85.

[10] 邓磊.计算机应用技术在工程项目管理中的应用[J].现代农村科技,2020(04):114.

[11] 佘金龙.计算机信息技术在机关单位信息化管理中的融合[J].科技风,2020(12):105.

[12] 杨军.计算机云计算的数据存储技术[J].科技风,2020(12):107.

[13] 朱婷.web前端技术在计算机相关专业教学中的应用研究[J].科技风,2020(12):113.

[14] 陈小华.事业单位管理中计算机信息管理系统的应用[J].科技风,2020(12):114.

[15] 周静,唐利平.“大数据”时代下计算机信息处理技术[J].科技风,2020(12):117+140.

[16] 赵学栋.计算机网络数据库的安全管理研究[J].计算机产品与流通,2020(04):67.

[17] 王永锋.计算机应用软件开发中编程语言的选择研究[J].计算机产品与流通,2020(05):24+38.

[18] 张清涛.计算机软件开发中的主要障碍与应对措施研究[J].计算机产品与流通,2020(05):27.

[19] 林惠然.科研院所实验动物管理系统的设计与开发[J].上海畜牧兽医通讯,2020(02):39-41.

[20] 白祎花.基于网络分布式文件系统设计与实现路径[J].微型电脑应用,2020,36(04):115-117.

[21] 张峰,刘迪,仇丽青,赵卫东.面向软件开发实践能力的《Java程序设计》教学资源建设[J].软件工程,2020.

[22] Casio Computer Co. Ltd.; "Exercise Information Acquisition Equipment" in Patent Application Approval Process (USPTO 20200100730)[J]. Computer Technology Journal,2020.

[23] Brother Kogyo Kabushiki Kaisha; Patent Issued for Communication Device, And Method And Computer Readable Medium For The Same (USPTO 10,613,802)[J]. Computer Technology Journal,2020.

[24] Manfred Kerber,Colin Rowat,Neels Vosloo. Using formal verification to develop higher assurance, more maintainable financial software[J]. Journal of Risk Management in Financial Institutions,2020

[25] Science - Computer Science; Studies from University of Naples Federico II Have Provided New Information about Computer Science (Alternating-time temporal logics with linear past)[J]. Computer Technology Journal,2020.

[26] Ranjith K. Soman,Jennifer K. Whyte. Codification Challenges for Data Science in Construction[J]. Journal of Construction Engineering and Management,2020,146(7).

[27] Casio Computer Co. Ltd.; Patent Issued for Phosphor Wheel (USPTO 10,613,317)[J]. Computer Technology Journal,2020.

[28] EMC IP Holding Company LLC; "Method, Device And Computer Program Product Of Storage Management" in Patent Application Approval Process (USPTO 20200104051)[J]. Computer Technology Journal,2020.

[29] Science - Computer Science; Researchers at Shanghai University Release New Data on Computer Science (Entity-related Paths Modeling for Knowledge Base Completion)[J]. Computer Technology Journal,2020.

[30] Bahador Sharifzadeh,Rasool Kalbasi,Mehdi Jahangiri,Davood Toghraie,Arash Karimipour. Computer modeling of pulsatile blood flow in elastic artery using a software program for application in biomedical engineering[J]. Computer Methods and Programs in Biomedicine,2020,192.

[31]Quang A Dang,Manh Tuan Hoang. Positivity and global stability preserving NSFD schemes for a mixing propagation model of computer viruses[J]. Journal of Computational and Applied Mathematics,2020,374.

附录A 外文翻译—原文部分

外文原文出处：出自<<Effective Java>>第15章。

**Minimize the accessibility of classes and members**

The single most important factor that distinguishes a well-designed component from a poorly designed one is the degree to which the component hides its internal data and other implementation details from other components. A well- designed component hides all its implementation details, cleanly separating its API from its implementation. Components then communicate only through their APIs and are oblivious to each others’ inner workings. This concept, known as information hiding or encapsulation, is a fundamental tenet of software design [Parnas72].

Information hiding is important for many reasons, most of which stem from the fact that it decouples the components that comprise a system, allowing them to be developed, tested, optimized, used, understood, and modified in isolation. This speeds up system development because components can be developed in parallel. It eases the burden of maintenance because components can be understood more quickly and debugged or replaced with little fear of harming other components. While information hiding does not, in and of itself, cause good performance, it enables effective performance tuning: once a system is complete and profiling has determined which components are causing performance problems (Item 67), those components can be optimized without affecting the correctness of others. Information hiding increases software reuse because components that aren’t tightly coupled often prove useful in other contexts besides the ones for which they were developed. Finally, information hiding decreases the risk in building large systems because individual components may prove successful even if the system does not.

Java has many facilities to aid in information hiding. The access control mechanism [JLS, 6.6] specifies the accessibility of classes, interfaces, and members. The accessibility of an entity is determined by the location of its declaration and by which, if any, of the access modifiers (private, protected, and public) is present on the declaration. Proper use of these modifiers is essential to information hiding.

The rule of thumb is simple: make each class or member as inaccessible as possible. In other words, use the lowest possible access level consistent with the proper functioning of the software that you are writing.

For top-level (non-nested) classes and interfaces, there are only two possible access levels: package-private and public. If you declare a top-level class or interface with the public modifier, it will be public; otherwise, it will be package-private. If a top-level class or interface can be made package-private, it should be. By making it package-private, you make it part of the implementation rather than the exported API, and you can modify it, replace it, or eliminate it in a subsequent release without fear of harming existing clients. If you make it public, you are obligated to support it forever to maintain compatibility.

If a package-private top-level class or interface is used by only one class, consider making the top-level class a private static nested class of the sole class that uses it (Item 24). This reduces its accessibility from all the classes in its package to the one class that uses it. But it is far more important to reduce the accessibility of a gratuitously public class than of a package-private top-level class: the public class is part of the package’s API, while the package-private top-level class is already part of its implementation.

For members (fields, methods, nested classes, and nested interfaces), there are four possible access levels, listed here in order of increasing accessibility:

private—The member is accessible only from the top-level class where it is declared.

package-private—The member is accessible from any class in the package where it is declared. Technically known as default access, this is the access level you get if no access modifier is specified (except for interface members, which are public by default).

protected—The member is accessible from subclasses of the class where it is declared (subject to a few restrictions [JLS, 6.6.2]) and from any class in the package where it is declared.

public—The member is accessible from anywhere.

After carefully designing your class’s public API, your reflex should be to make all other members private. Only if another class in the same package really needs to access a member should you remove the private modifier, making the member package-private. If you find yourself doing this often, you should reexamine the design of your system to see if another decomposition might yield classes that are better decoupled from one another. That said, both private and package-private members are part of a class’s implementation and do not normally impact its exported API. These fields can, however, “leak” into the exported API if the class implements Serializable (Items 86 and 87).

For members of public classes, a huge increase in accessibility occurs when the access level goes from package-private to protected. A protected member is part of the class’s exported API and must be supported forever. Also, a protected member of an exported class represents a public commitment to an implementation detail (Item 19). The need for protected members should be relatively rare.

There is a key rule that restricts your ability to reduce the accessibility of methods. If a method overrides a superclass method, it cannot have a more restrictive access level in the subclass than in the superclass [JLS, 8.4.8.3]. This is necessary to ensure that an instance of the subclass is usable anywhere that an instance of the superclass is usable (the Liskov substitution principle, see Item 15). If you violate this rule, the compiler will generate an error message when you try to compile the subclass. A special case of this rule is that if a class implements an interface, all of the class methods that are in the interface must be declared public in the class.

To facilitate testing your code, you may be tempted to make a class, interface, or member more accessible than otherwise necessary. This is fine up to a point. It is acceptable to make a private member of a public class package-private in order to test it, but it is not acceptable to raise the accessibility any higher. In other words, it is not acceptable to make a class, interface, or member a part of a pack-age’s exported API to facilitate testing. Luckily, it isn’t necessary either because tests can be made to run as part of the package being tested, thus gaining access to its package-private elements.

Instance fields of public classes should rarely be public (Item 16). If an instance field is nonfinal or is a reference to a mutable object, then by making it public, you give up the ability to limit the values that can be stored in the field. This means you give up the ability to enforce invariants involving the field. Also, you give up the ability to take any action when the field is modified, so classes with public mutable fields are not generally thread-safe. Even if a field is final and refers to an immutable object, by making it public you give up the flexibility to switch to a new internal data representation in which the field does not exist.

The same advice applies to static fields, with one exception. You can expose constants via public static final fields, assuming the constants form an integral part of the abstraction provided by the class. By convention, such fields have names consisting of capital letters, with words separated by underscores (Item 68). It is critical that these fields contain either primitive values or references to immutable objects (Item 17). a field containing a reference to a mutable object has all the disadvantages of a nonfinal field. While the reference cannot be modified, the referenced object can be modified—with disastrous results.

Note that a nonzero-length array is always mutable, so it is wrong for a class to have a public static final array field, or an accessor that returns such a field. If a class has such a field or accessor, clients will be able to modify the contents of the array. This is a frequent source of security holes.

Beware of the fact that some IDEs generate accessors that return references to private array fields, resulting in exactly this problem. There are two ways to fix the problem. You can make the public array private and add a public immutable list.Alternatively, you can make the array private and add a public method that returns a copy of a private array.

To choose between these alternatives, think about what the client is likely to do with the result. Which return type will be more convenient? Which will give better performance?

As of Java 9, there are two additional, implicit access levels introduced as part of the module system. A module is a grouping of packages, like a package is a grouping of classes. A module may explicitly export some of its packages via export declarations in its module declaration (which is by convention contained in a source file named module-info.java). Public and protected members of unexported packages in a module are inaccessible outside the module; within the module, accessibility is unaffected by export declarations. Using the module system allows you to share classes among packages within a module without making them visible to the entire world. Public and protected members of public classes in unexported packages give rise to the two implicit access levels, which are intramodular analogues of the normal public and protected levels. The need for this kind of sharing is relatively rare and can often be eliminated by rearranging the classes within your packages.

Unlike the four main access levels, the two module-based levels are largely advisory. If you place a module’s JAR file on your application’s class path instead of its module path, the packages in the module revert to their non- modular behavior: all of the public and protected members of the packages’ public classes have their normal accessibility, regardless of whether the packages are exported by the module [Reinhold, 1.2]. The one place where the newly introduced access levels are strictly enforced is the JDK itself: the unexported packages in the Java libraries are truly inaccessible outside of their modules.

Not only is the access protection afforded by modules of limited utility to the typical Java programmer, and largely advisory in nature; in order to take advantage of it, you must group your packages into modules, make all of their dependencies explicit in module declarations, rearrange your source tree, and take special actions to accommodate any access to non-modularized packages from within your modules [Reinhold, 3]. It is too early to say whether modules will achieve widespread use outside of the JDK itself. In the meantime, it seems best to avoid them unless you have a compelling need.

To summarize, you should reduce accessibility of program elements as much as possible (within reason). After carefully designing a minimal public API, you should prevent any stray classes, interfaces, or members from becoming part of the API. With the exception of public static final fields, which serve as constants, public classes should have no public fields. Ensure that objects referenced by public static final fields are immutable.

附录B 外文翻译—译文部分

**使类和成员的可访问性最小化**

将设计良好的组件与设计不良的组件区分开的最重要因素是该组件从其他组件中隐藏其内部数据和其他实现细节的程度。设计良好的组件会隐藏其所有实现细节，从而将其API与实现完全分开。然后，组件仅通过其API进行通信，而忽略了彼此的内部工作方式。 这个概念称为信息隐藏或封装，是软件设计的基本原则。

信息隐藏之所以重要，有许多原因，其中大部分原因是它使构成系统的组件脱钩，从而使它们可以独立开发，测试，优化，使用，理解和修改。由于可以并行开发组件，因此可以加快系统开发速度。由于可以更快速地理解组件并进行调试或更换，而几乎不担心会损坏其他组件，因此它减轻了维护负担。尽管信息隐藏本身并不能带来良好的性能，但是它可以进行有效的性能调整：一旦系统完成并且性能分析确定了哪些组件导致了性能问题，则可以在不影响正确性的情况下优化这些组件。别人的。信息隐藏可以提高软件的复用率，因为紧密耦合的组件通常在开发它们所针对的其他环境中也很有用。最后，信息隐藏降低了构建大型系统的风险，因为即使系统不成功，单个组件也可能被证明是成功的。

Java具有许多辅助信息隐藏的功能。访问控制机制[JLS，6.6]指定类，接口和成员的可访问性。实体的可访问性取决于其声明的位置以及声明中是否存在访问修饰符（私有，受保护和公共）（如果有）决定。正确使用这些修饰符对于隐藏信息至关重要。

经验法则很简单：使每个类或成员尽可能不可访问。换句话说，使用与您正在编写的软件的正常功能相一致的最低访问级别。

对于顶级（非嵌套）类和接口，只有两种可能的访问级别：包私有和公共。如果用public修饰符声明一个顶级类或接口，它将是public；否则，它将是包私有的。如果可以将顶级类或接口设置为包私有，则应该这样做。通过使其成为程序包私有，可以使其成为实现的一部分，而不是导出的API，并且可以在以后的发行版中对其进行修改，替换或删除，而不必担心会损害现有客户端。如果将其公开，则有义务永远支持它以保持兼容性。

如果包私有顶级类或接口仅由一个类使用，请考虑使顶级类成为使用它的唯一类（项24）的私有静态嵌套类。这就减少了它从包中的所有类到使用它的一个类的可访问性。但是，与包私有顶级类相比，减少免费公共类的可访问性要重要得多：公共类是包的API的一部分，而包私有顶级类已经是其实现的一部分。

对于成员（字段、方法、嵌套类和嵌套接口），这里列出了四种可能的访问级别，以提高可访问性：

private—成员只能从声明它的顶级类访问。

package-private—可以从包中声明该成员的任何类访问该成员。技术上称为默认访问，这是在未指定访问修饰符时获得的访问级别（除了默认为公共的接口成员）。

protected—可以从声明该成员的类的子类（受一些限制[JLS，6.6.2]）和声明该成员的包中的任何类访问该成员。

public—成员可以从任何地方访问。

在仔细设计了类的公共API之后，您的反思应该是将所有其他成员设为私有。仅当同一包中的另一个类确实需要访问成员时，才应删除private修饰符，使该成员变为包私有。 如果您发现自己经常这样做，则应重新检查系统的设计，以查看另一分解是否可以产生更好地彼此分离的类。也就是说，私有成员和程序包私有成员都是类实现的一部分，通常不会影响其导出的API。但是，如果该类实现了Serializable（项目86和87），则这些字段可以“泄漏”到导出的API中。

对于公共类的成员来说，当访问级别从包私有变为受保护时，可访问性会大大增加。受保护的成员是类导出API的一部分，必须永远受支持。此外，导出类的受保护成员表示对实现细节的公共承诺（项19）。对受保护成员的需求应该相对较少。

有一个关键规则限制您减少方法可访问性的能力。如果一个方法重写一个超类方法，那么它在子类中的访问级别不能比在超类[JLS，8.4.8.3]中的访问级别更严格。这对于确保子类的实例在超类的实例可用的任何地方都可用是必要的（Liskov替换原则，见第15项）。如果违反此规则，编译器将在您尝试编译子类时生成错误消息。此规则的一个特殊情况是，如果类实现接口，则接口中的所有类方法都必须在类中声明为public。

为了方便测试代码，您可能会尝试使类、接口或成员比其他必要的更易访问。这在一定程度上是可以的。将公共类包的私有成员设为私有以便对其进行测试是可以接受的，但是将可访问性提高到更高的级别是不可接受的。换句话说，将类、接口或成员作为pack age导出API的一部分以方便测试是不可接受的。幸运的是，也没有必要这样做，因为测试可以作为被测试包的一部分运行，从而获得对其包私有元素的访问。

公共类的实例字段很少应该是公共的（项16）。如果实例字段是非最终的或是对可变对象的引用，那么通过将其公开，您就放弃了限制可以存储在该字段中的值的能力。这意味着您放弃了强制执行涉及字段的不变量的能力。此外，当字段被修改时，您将放弃执行任何操作的能力，因此具有公共可变字段的类通常不是线程安全的。即使一个字段是final并且引用了一个不可变的对象，通过公开它，您就放弃了切换到字段不存在的新的内部数据表示的灵活性。

同样的建议也适用于静态字段，只有一个例外。您可以通过公共静态final字段公开常量，假设这些常量构成类提供的抽象的一个组成部分。按照惯例，这些字段的名称由大写字母组成，用下划线隔开（项目68）。这些字段必须包含原语值或对不可变对象（项17）的引用。包含对可变对象的引用的字段具有非最终字段的所有缺点。虽然不能修改引用，但可以修改被引用的对象，从而导致灾难性的结果。

注意，非零长度数组总是可变的，所以类拥有公共静态final数组字段或返回此类字段的访问器是错误的。如果类具有这样的字段或访问器，则客户端将能够修改数组的内容。这是安全漏洞的常见来源。

注意，一些ide生成的访问器返回对私有数组字段的引用，这正是导致这个问题的原因。有两种方法可以解决这个问题。您可以将公共数组设为私有并添加一个公共的不可变列表，也可以将数组设为私有并添加一个返回私有数组副本的公共方法。

要在这些备选方案中进行选择，请考虑客户可能会对结果做什么。哪种退货方式更方便？哪一个会有更好的表现？

从Java9开始，有两个额外的隐式访问级别作为模块系统的一部分引入。模块是包的分组，就像包是类的分组一样。模块可以通过其模块声明中的导出声明（按照惯例，该声明包含在名为module info.java的源文件中）显式导出其部分包。模块中未报告包的公共成员和受保护成员在模块外部不可访问；在模块内部，可访问性不受导出声明的影响。使用模块系统，您可以在模块内的包之间共享类，而不会使它们对整个世界可见。未报告包中的公共类的Public和protected成员产生两个隐式访问级别，这两个级别在模块内类似于普通的Public和protected级别。这种共享的需求相对较少，通常可以通过在包中重新排列类来消除。

与四个主要访问级别不同，这两个基于模块的级别主要是咨询级别。如果将模块的JAR文件放在应用程序的类路径而不是模块路径上，模块中的包将恢复其非模块化行为：包的公共类的所有公共成员和受保护成员都具有其正常的可访问性，而不管包是否由模块导出[Reinhold，1.2]。新引入的访问级别被严格执行的地方之一是JDK本身：Java库中未报告的包在其模块之外是真正不可访问的。

对于典型的Java程序员来说，功能有限的模块不仅提供了访问保护，而且在本质上也提供了很大的建议；为了利用它，您必须将包分组到模块中，在模块声明中明确它们的所有依赖关系，重新排列源树，并采取特殊的操作来适应任何从模块内访问非模块化的包[Reinhold，3]。现在说模块是否能在JDK之外实现广泛使用还为时过早。同时，除非你有迫切的需要，否则最好避免它们。

总而言之，您应该尽可能减少程序元素的可访问性（在合理的范围内）。在仔细设计了一个最小的公共API之后，您应该防止任何杂散的类、接口或成员成为API的一部分。除了作为常量的public static final字段外，public类不应该有public字段。确保公共静态最终字段引用的对象是不可变的。

附录C 软件使用说明书

网站本地访问地址为：http://localhost:8181/Test

普通用户和系统管理员在登录系统时，都在系统的登录页面进行账号和密码的输入，系统会根据不同角色进入不同的页面。

用户在登录时应确保在本系统中有账户存在，若是新用户则可以进行注册，登录之后用户可以看到如图1所示的界面。

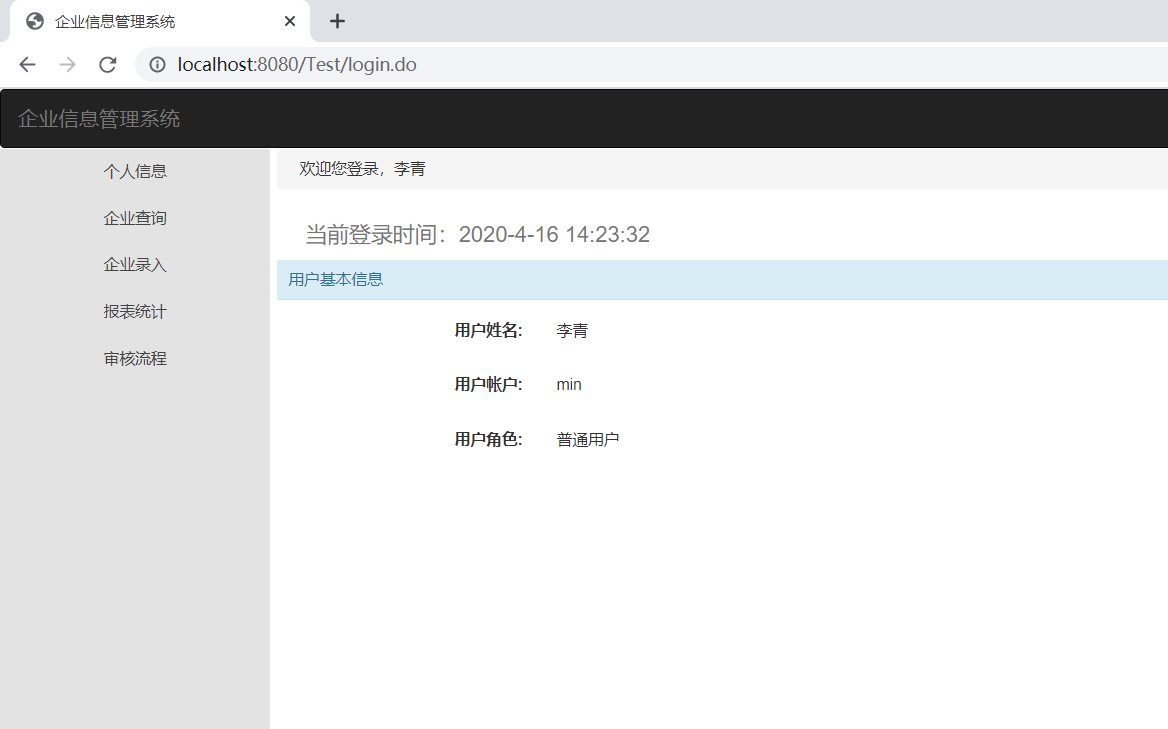


图 1普通用户登录主页展示图

普通用户登录成功之后可以使用如图1所示的功能：个人信息查看，企业信息查询，企业信息录入，报表统计，查看审核流程。其中，企业录入功能不仅支持页面输入各项信息，还支持文件导入，具体步骤为：在主页左栏点击企业录入，当页面跳转到企业信息录入页面后，点击选择文件，效果如图2所示。在企业查询也提供信息导出功能，如图3。

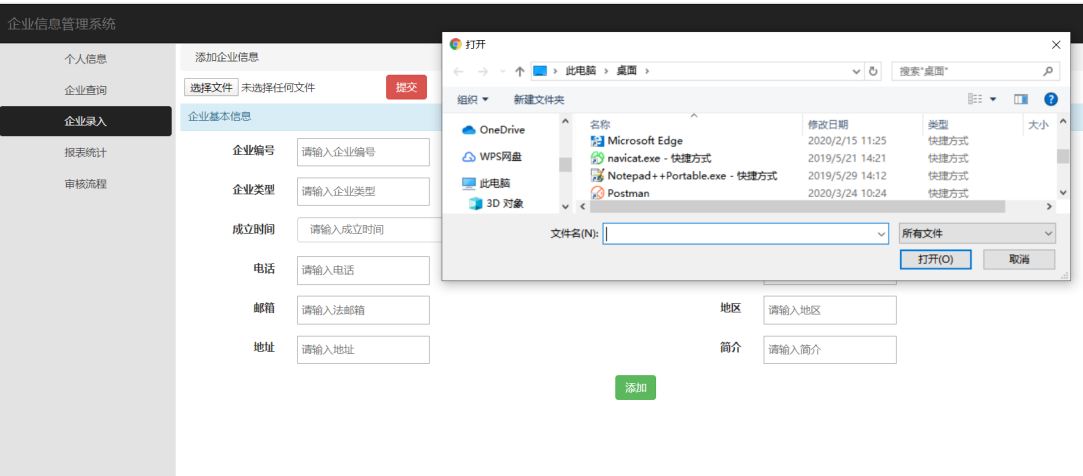


图 2企业信息录入展示图



图 3企业信息导出展示图

系统管理员在登录界面输入正确的账号和密码登录后，页面就会跳转到如图4所示的界面。

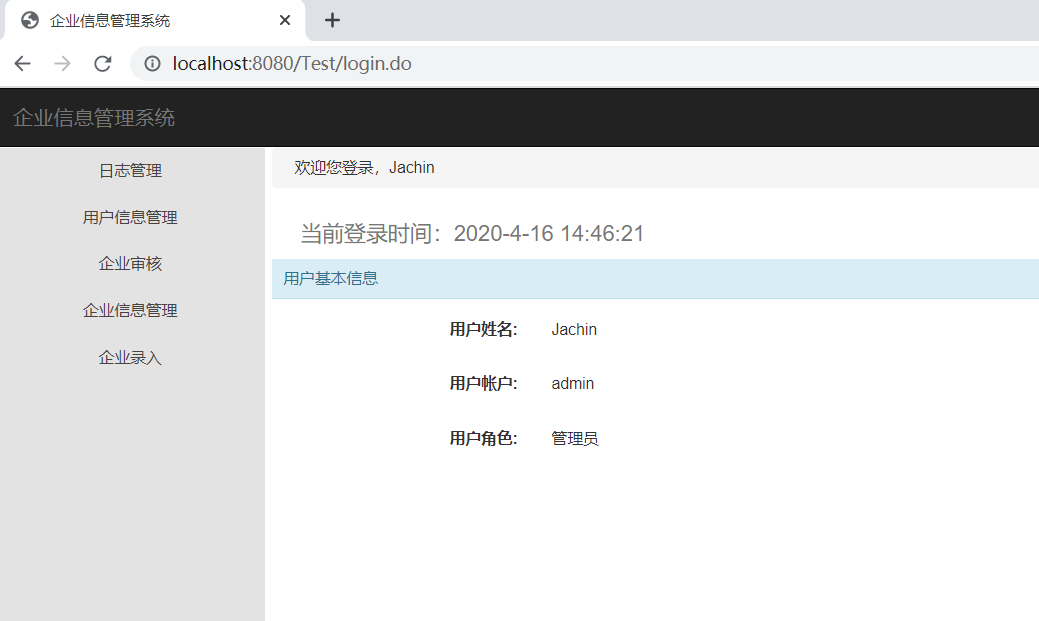


图 4管理员登录主页展示图

如上图所示，管理员可以使用日志管理，用户信息管理，企业审核，企业信息管理，企业录入功能。

附录D 主要源代码



