 **JIANGXINORMALUNIVERSITY**

本科生毕业设计（论文）

**基于Spring Boot的博客系统的设计与实现**

**Design and Implementation of Blog System Based on Spring Boot**

**学院名称：计算机信息工程学院**

**专业名称： 网络工程**

**姓 名： 林臣哲**

**学 号： 201526203055**

**指导教师： 周新宇**

**完成日期：**

## 声明

本人郑重声明：

本人的毕业论文（设计）是在指导教师指导下独立进行研究取得的成果。其中引用的地方都加了标注和致谢，在法律规定允许的情况下，不包含任何其他个人或集体已经撰写完成或已经发表的以某种方式公开过的研究成果，不包含为了获得教育机构的证书或学位而作的材料。任何对本研究有贡献的个人和集体都在文中作了明确的说明并且表达了十分的谢意。本人完全意识到本声明的法律责任由本人承担。

本人的毕业论文（设计）成果在江西师范大学读书期间，在指导教师精心的指导下取得的，成果归江西师范大学计算机信息工程学院所有。  
 特此声明。

声明人（毕业设计（论文）作者）学号：  
声明人（毕业设计（论文）作者）签名：  
  
签名日期：　　　　年　　月　　日

**摘要**

近些年的网络急剧发展，带来了许多的便利与信息，人与人之间的沟通方式越来越多元化，信息传播的方式也数不胜数，微博、微信公众号、博客，其中博客是获取各种信息的重要途径，也是现在信息时代记录大量信息的重要载体。

本项目是一个基于Spring Boot框架的多用户博客系统，配合Spring MVC框架，主要集成了Spring Security安全框架、Spring Session用户会话管理、Thymeleaf模板引擎、BootStrap前端框架、HTML、CSS、JavaScript、MySQL，同时使用Gradle项目构建工具对项目进行管理，使用Git对项目进行版本控制管理。

**关键字：**Spring Boot ; Spring Security; MySQL; BootStrap; Thymeleaf;

1. **绪论**

**1.1开发背景**

博客是在互联网上发布信息或讨论的网站，由离散的，通常是非正式的日记式文本条目（帖子）组成。帖子通常以时间倒序显示，以便最新的帖子首先出现在网页的顶部。直到2009年，博客通常是一个人的工作，偶尔是一个小组，并且经常涉及一个主题。在21世纪初，出现了“多作者博客”（MABs），博客中包括多位作者的文章，有时还经过专业的编辑。来自报纸，媒体，大学，智囊团和类似机构的MAB占了越来越多的博客流量。 随着Twitter和其他“微博”系统的兴起，助于将MAB和单作者博客整合到新闻媒体中。博客也可以用作动词，意味着维护或添加博客内容。

20世纪90年代后期博客的出现和发展恰逢网络出版工具的出现，这些工具促进了对HTML或计算机编程经验不足的非技术用户发布内容。以前，需要了解HTML和FTP协议等技术才能在Web上发布文章，因此早期的Web用户往往是黑客和计算机爱好者。在21世纪初，大多数是互动的Web 2.0网站，允许访问者留下在线评论，正是这种互动使他们与其他静态网站区别开来。从这个意义上说，博客可以被视为社交网络服务的一种形式。事实上，博客作者不仅会在他们的博客上发布内容，还会经常与读者和其他博客建立社交关系。许多博客都提供有关特定主题或主题的评论，从政治到体育。部分人则当的是个人在线日记，而其他人则更多地充当特定个人或公司的在线品牌广告。典型的博客将文本，数字图像以及与其主题相关的其他博客，网页和其他媒体的链接相结合。读者留下公开可见评论并与其他评论者互动的能力是许多博客受欢迎的重要贡献。大多数博客主要是文本，但有些专注于艺术（艺术博客），照片（照片博客），视频（视频博客或“视频博客”），音乐（MP3博客）和音频（播客）。

**1.2开发目的**

开发的目的是为了提供记录个人想法和与他人分享知识的平台，有以下几方面：

1、博客可以作为一个见证，它既可以见证过去也可以见证未来看自己以前的文章，可以回顾自己的成长，就算是黑历史，也可以和现在的我打个照面，看看自己的改变，目前正在写的博文也将是未来我的“参考历史”，给自己留下一个悬念，期待一下现在的自己和未来的自己又会有怎样的不同。博客可以作为一个时间的媒介，它可以历尽沧海仍不变，供它的主人来了解其本身。

2、在学习到新知识时可以在博客上进行总结，并与他人进行交流。

**第2章 需求分析**

在系统工程和软件工程中，需求分析是指在创建新系统或更改现有系统或产品时确定新系统的目的，范围，定义和功能所需的所有工作。 需求分析是软件工程中的关键过程。 在此过程中，系统分析员和软件工程师确定客户的需求。 只有在确定了这些需求后，他们才能分析并寻求新系统的解决方案。

**2.1系统目标**

游客可浏览所有公开文章，如想自己发布文章或者与其他博主互动，需使用账号登录到系统中，对文章评论进行增删改查操作。在发布文章时同时指定文章所属分类，和相应的标签，在每篇文章下面提供了评论功能，可以与对文章进行评论。首页展示热门标签，分类，热门文章，最新文章等用户感兴趣的板块。在后台方面，系统提供了对分类，标签，文章，账号，权限进行统一管理。可对系统进行精细化管理。

**2.2系统需求**

**2.2.1软硬件需求**

硬件需求

PC机

(1)CPU：i7

(2)内存：8G

(3)固态硬盘：128G

(4)操作系统：Windows 10 Pro

软件需求：

数据库：MySQL

开发工具：IntelliJ IDEA

浏览器：Chrome

Web容器 :Tomcat

其他工具：Git

**2.2.2技术需求**

对系统开发来说，技术可行性分析至关重要，做技术可行性分析时需注意全面考虑系统开发过程所涉及的所有技术问题，尽可能采用成熟技术，慎重引入先进技术。

1、Java语言

Java是一种通用的计算机编程语言，它是并发的，基于类的，面向对象的，并且专门设计为具有尽可能少的实现依赖性。它旨在让应用程序开发人员“编写一次，随处运行”，这意味着编译的Java代码可以在支持Java的所有平台上运行，而无需重新编译。Java应用程序通常编译为可以在任何Java虚拟机（JVM）上运行的字节码，而不管计算机体系结构如何。截至2016年，Java是最常用的编程语言之一，特别适用于客户端 - 服务器Web应用程序。该语言从SmallTalk中获得了大部分原始特性，其语法类似于C和C ++。

2、MySQL

MySQL是一个关系型数据库管理系统，MySQL 是目前最流行的关系型数据库管理系统之一，在WEB应用方面，MySQL是最好的 RDBMS 应用软件。MySQL是一种关系数据库管理系统，关系数据库将数据保存在不同的表中，而不是将所有数据放在一个大仓库内，这样就增加了速度并提高了灵活性。MySQL所使用的 SQL 语言是用于访问数据库的最常用标准化语言。

3、Spring Boot

Spring Boot优势在于轻松创建可以独立运行的生产级的Spring的应用程序。Spring Boot对Spring平台和第三方库进行了整合，提供了更便利的使用方式。Spring Boot应用程序几乎不需要的Spring配置。因为Spring Boot遵循“约定大于配置”准则。 使用Spring Boot时可以创建Jar包或更传统的war包进行部署启动。Spring Boot还提供了一个运行“spring脚本”的命令行工具，使开发者能够更快且可广泛访问地使Spring开发。 Spring Boot还为大型项目（例如嵌入式服务器，安全性，度量标准，运行状况检查和外部化配置）提供一系列通用的非功能性功能。

4、Spring Security

Spring Security是一个安全框架，提供针对常见攻击的身份验证，授权和保护。通过对命令式和反应式应用程序的一流支持，它是保护基于Spring的应用程序的事实标准。Spring Security充分利用了Spring的 IoC（控制反转Inversion of Control），DI（依赖注入Dependency Injection）和AOP（面向切面编程）功能，Spring Security为应用系统提供了声明式的安全访问控制，使企业的安全控制更具灵活性。

5、Thymeleaf

Thymeleaf是一个现代服务器端Java模板引擎，适用于Web和独立环境，能够处理HTML，XML，JavaScript，CSS甚至纯文本。Thymeleaf的主要目标是提供一种优雅且高度可维护的模板创建方式。为实现这一目标，它以自然模板的概念为基础，将其逻辑注入模板文件，其方式不会影响模板被用作设计原型。这改善了设计沟通，缩小了设计和开发团队之间的差距。

6、BootStrap

Bootstrap是一个免费的开源前端Web框架。它包含用于排版，表单，按钮，导航和其他界面组件的HTML和基于CSS的设计模板，以及可选的JavaScript扩展。与许多早期的Web框架不同，它只关注前端开发。 BootStrap是上最受欢迎的前端组件库在Web上构建响应式，同时也是移动设备优先项目。 Bootstrap是一个用于使用HTML，CSS和JS进行开发的开源工具包。Bootstrap使用Sass变量和mixins，响应式网格系统，广泛的预构建组件以及基于jQuery构建的强大插件，能够快速构建整个应用程序。

7、Spring Session

Spring Session提供了用于管理用户会话信息的API和实现，同时也使得支持群集会话管理变得易如反掌，同时不依赖于特定于应用程序容器的解决方案。它还提供透明集成组件：HttpSession允许以应用程序容器（即Tomcat）中立方式替换HttpSession，支持在头文件中提供会话ID以使用RESTful API。 WebSocket提供在接收WebSocket消息时保持HttpSession活动的能力。WebSession允许以应用程序容器中立方式替换Spring WebFlux的WebSession。

8、Hibernate

Hibernate是一个开放源代码的对象关系映射框架，它对JDBC进行了非常轻量级的对象封装， Hibernate负责从Java类到数据库表的映射，以及从Java数据类型到SQL数据类型的映射。此外，它还提供数据查询和检索功能。它可以显着缩短在SQL和JDBC中使用手动数据处理所花费的开发时间。 Hibernate的设计目标是通过消除使用SQL和JDBC进行手动数据处理的需求，使开发人员从95％的常见数据持久性相关编程任务中解脱出来。但是，与许多其他持久性解决方案不同，Hibernate不会隐藏SQL的强大功能，SQL中的功能Hibernate一样有相应实现。

9、富文本编辑器CKEditor 5

CKEditor 5能够快速，轻松地应用程序中使用，同时，它是一个用于创建定制的富文本编辑解决方案的框架。CKEditor 5提供了可以想象的所有类型的WYSIWYG编辑解决方案。从类似于Google Docs和Medium的编辑器到Slack或类似Twitter的应用程序，所有这些都可以在一个编辑框架内完成。CKEditor 5有三种风格：Classic、Inline、Balloon。这里选用Classic风格更适合博客编辑

2.2.3功能需求

在软件工程和系统工程中，功能需求定义了系统或其组件的功能，其中功能被描述为输出和输入之间的行为规范。功能需求可能涉及计算，技术细节，数据操作和处理，以及定义系统应该完成什么的其他特定功能。在用例中捕获描述系统使用功能需求的所有情况的行为要求。功能要求由非功能性要求（也称为“质量要求”）支持，这些要求对设计或实施（例如性能要求，安全性或可靠性）施加约束。通常，功能要求以“系统必须<要求>”的形式表示，而非功能要求采用“系统应<要求>”的形式。系统设计中详细说明了实现功能要求的计划。而非功能性要求在系统架构中有详细说明。

**普通用户界面：**

首页：展示用户发布的最新文章，最热门文章，分类以及热门标签

文章详情：展示文章详情，用户可以在文章下进行评论

用户主页：展示用户发布的所有文章以及评论，同时具有管理功能，支持增删改查操作，同时在账号信息中提供修改邮箱密码和头像的功能。

**管理员后台界面：**

分类管理：对分类进行管理，支持增删改操作。

标签管理：对标签进行管理，支持增删改操作。

文章管理：对所有文章进行管理，支持删除操作，和不公开操作。

权限管理：对用户权限进行管理，指定能否发布文章或者评论浏览之类的。

评论管理：对用户发布的评论进行管理，支持删除操作和查询操作。

**第3章 开发平台介绍与环境搭建**

**3.1 开发平台**

本博客系统前期基于Windows 10 64位平台开发，后期转向使用Mac平台开发，同时远程部署到Linux服务器进行远程运行调试。

**3.2 控制反转(IoC)**

控制反转（IoC）是一种编程原则。与传统控制流相比，IoC反转流量控制。在IoC中，计算机程序的自定义编写部分从通用框架接收控制流。与传统的过程编程相比，采用这种设计的软件架构颠倒了控制：在传统的编程中，表达目的的自定义代码调用可重用的库来处理通用任务，但是通过控制反转，它就是框架调用自定义或特定于任务的代码。

控制反转用于增加程序的模块性并使其可扩展,并且在面向对象编程和其他编程范例中具有应用。

控制反转与依赖性倒置原则有关，但不同之处在于依赖性倒置原则，该原理通过共享抽象将高层和低层之间的依赖关系解耦。一般概念也与事件驱动编程有关，因为它通常使用IoC实现，因此自定义代码通常仅涉及事件的处理，而事件/消息的事件循环和调度由框架处理或运行时环境。

**3.3 依赖注入(DI)**

依赖注入是一种技术，其中一个对象（或静态方法）提供另一个对象的依赖关系。依赖项是可以使用的对象（服务）。注入是将依赖项传递给将使用它的依赖对象（客户端）。该服务成为客户端状态的一部分。将服务传递给客户端，而不是允许客户端构建或查找服务，是模式的基本要求。

依赖注入的目的是将对象解耦到不需要更改客户端代码的程度，因为它所依赖的对象需要更改为不同的对象。这允许遵循开放/封闭原则。

依赖注入是更广泛的控制反转技术的一种形式。与其他形式的控制反转一样，依赖注入支持依赖性反转原则。客户端将其依赖关系的责任委托给外部代码（注入器）。客户端不允许调用注入器代码; 它是构建服务并调用客户端注入它们的注入代码。这意味着客户端代码不需要知道注入代码，如何构建服务，甚至不知道它正在使用哪些实际服务;客户端只需要知道服务的内在接口，因为它们定义了客户端如何使用服务。这分离了使用和构造的责任。

客户端接受依赖注入有三种常用方法：基于setter，interface和constructor的注入。 Setter和构造函数注入主要取决于它们何时可以使用。接口注入的不同之处在于依赖关系有机会控制其自身的注入。每个都要求单独的构造代码（注入器）负责引入客户端及其相互依赖关系。

**3.4面向切面编程(AOP)**

面向方面编程（AOP）是一种编程范例，旨在通过分离横切关注点来增加模块性。它通过在不修改代码本身的情况下向现有代码（建议）添加其他行为来实现，而是单独指定通过“切入点”规范修改哪些代码，例如“当函数名称以'set'开头时记录所有函数调用”。这允许将不是业务逻辑中心的行为（例如日志记录）添加到程序中，而不会使代码混乱，核心功能。 AOP构成了面向方面的软件开发的基础。

面向切面的编程需要将程序逻辑分解为不同的部分（所谓的关注点，功能的内聚区域）。通过提供可用于实现，抽象和组合这些问题的抽象（例如，函数，过程，模块，类，方法），几乎所有编程范例都支持某种程度的关注分组和封装到独立的独立实体中。一些问题“跨越”程序中的多个抽象，并且无视这些形式的实现。这些担忧被称为跨领域问题或横向关注。

所有AOP实现都有一些横切表达式，将每个关注点封装在一个地方。实现之间的区别在于所提供的构造的功能，安全性和可用性。例如，拦截器指定表达有限形式的横切的方法，而不太支持类型安全或调试。 AspectJ有许多这样的表达式，并将它们封装在一个特殊的类中，一个切面。例如，一个切面可以通过在称为切入点的量化或查询中指定的各种连接点（程序中的点）上应用建议（附加行为）来改变基本代码（程序的非方面部分）的行为（检测给定的连接点是否匹配）。切面还可以对其他类进行二进制兼容的结构更改，例如添加成员或父项。

**3.3MVC架构**

Model–View–Controller（通常称为MVC或MVVC变体的Model WC）是一种通常用于开发用户界面的体系结构模式，将应用程序划分为三个互连的部分。这样做是为了将信息的内部表示与信息呈现给用户和从用户接受的方式分开。MVC设计模式将这些主要组件分离，从而实现高效的代码重用和并行开发。 传统上用于桌面图形用户界面（GUI），这种架构已经成为设计Web应用程序甚至移动，桌面和其他客户端的首选。流行的编程语言，如Java，C＃，Python，Ruby，PHP都有MVC框架，可直接用于Web应用程序开发。

Model： 模式的核心组成部分。它是应用程序的动态数据结构，独立于用户界面。它直接管理应用程序的数据、逻辑和规则。

View： 多种信息表示，如图表，图表或表格。可以使用相同信息的多个视图，例如管理条形图和会计师的表格视图。

Controller： 接受用户请求并将其转换为模型或视图。 除了将应用程序划分为这些组件之外，Model–View–Controller设计还定义了它们之间的交互。该模型负责管理应用程序的数据。它从控制器接收用户输入。 视图意味着以特定格式呈现模型。 控制器响应用户输入并对数据模型对象执行交互。控制器接收输入，可选地验证它，然后将输入传递给模型。 与其他软件模式一样，MVC表达了问题的“核心解决方案”，同时允许它适用于每个系统。



**3.6开发环境与工具**

**3.6.1 Tomcat容器**

Apache Tomcat，通常称为Tomcat Server，是一个开源的Java Servlet容器。Tomcat实现了几个Java EE规范，包括Java Servlet，JavaServer Pages（JSP），Java EL和WebSocket，并提供了一个可以运行Java代码的“纯Java”HTTP Web服务器环境。

Tomcat包含以下几个组件：

Catalina：Catalina是Tomcat的servlet容器。 Catalina实现了Sun Microsystems的servlet和JavaServer Pages（JSP）规范。

Coyote：Coyote是Tomcat的Connector组件，它支持HTTP 1.1协议作为Web服务器。这使得Catalina（名义上是Java Servlet或JSP容器）也可以充当普通的Web服务器，将本地文件作为HTTP文档提供。

Jasper：Jasper是Tomcat的JSP引擎。 Jasper解析JSP文件，将它们编译为Java代码作为servlet（可以由Catalina处理）。在运行时，Jasper检测JSP文件的更改并重新编译它们。这也提供了在开发时进行热部署，修改代码无需重启Tomcat的功能基础。

**3.6.2分布式版本控制系统**

Git是一个分布式版本控制系统，用于跟踪软件开发过程中源代码的变化。它旨在协调程序员之间的工作，但它可用于跟踪任何文件集中的更改。其目标包括速度，数据完整性，以及对分布式非线性工作流程的支持。与大多数其他分布式版本控制系统一样，与大多数客户端 - 服务器系统不同，每台计算机上的每个Git目录都是一个完整的存储库，具有完整的历史记录和完整的版本跟踪能力，独立于网络访问或中央服务器。

**第4章 系统实现详述**

**4.1系统设计**

**4.1.1 系统架构设计**

博客系统主要包括账号，文章，评论，标签，分类，权限模块

**4.1.2 数据库设计**

数据库设计采用Hibernate的对象关系映射，并提使用了对象持久化的功能。

1、account（账号表）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| account\_id | bigint | - | not null | 主键，账号id |
| account\_non\_expired | bit | - | not null | 账号是否过期 |
| account\_non\_locked | bit | - | not null | 账号是否被锁定 |
| avatar | varchar | 255 | not null | 头像地址 |
| create\_date | datetime | - | not null | 创建日期 |
| credentials\_non\_expired | bit | - | not null | 凭证是否过期 |
| email | varchar | 255 | not null | 电子邮箱 |
| enabled | bit | - | not null | 是否激活状态 |
| last\_modified\_date | datetime | - | not null | 上次修改时间 |
| password | varchar | 16 | not null | 密码 |
| username | varchar | 16 | not null | 用户名 |

2、account\_authority（账号权限表）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| account\_id | bigint | - | not null | 主键，账号id |
| authority\_id | bigint | - | not null | 主键，权限id |

3、article（文章表）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| article\_id | bigint | - | not null | 主键，文章id |
| content | mediumtext | - | not null | 富文本内容 |
| create\_date | datetime | - | not null | 创建时间 |
| is\_public | bit | - | not null | 是否公开 |
| last\_modified\_date | datetime | - | not null | 上次修改时间 |
| page\_view | bigint | - | not null | 浏览量 |
| raw\_content | mediumtext | - | not null | 纯文本内容 |
| title | varchar | 255 | not null | 标题 |
| account\_id | bigint | - | not null | 文章所属用户id |
| category\_id | bigint | - | not null | 所属分类id |

4、article\_tag（文章标签表）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| tag\_id | bigint | - | not null | 主键，标签id |
| article\_id | bigint | - | not null | 主键，文章id |

5、authority（权限表）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| authority\_id | bigint | - | not null | 主键，权限id |
| authority\_name | Varchar | 255 | not null | 权限名称 |
| create\_date | datetime | - | not null | 创建时间 |
| last\_modified\_date | datetime | - | not null | 上次修改时间 |

6、category（分类表）-----------------待定

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| category\_id | bigint | - | not null | 主键，分类id |
| category\_name | Varchar | 255 | not null | 分类名称 |

7、comment（评论表）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| comment\_id | bigint | - | not null | 主键，评论id |
| content | Varchar | 255 | not null | 评论内容 |
| create\_date | datetime | - | not null | 创建时间 |
| last\_modified\_date | datetime | - | not null | 上次修改时间 |
| account\_id | bigint | - | not null | 评论所属账号id |
| article\_id | bigint | - | not null | 评论所属文章id |

8、favorites\_article（用户文章收藏表）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| account\_id | bigint | - | not null | 主键，账号id |
| article\_id | bigint | - | not null | 主键，文章id |

9、tag（标签表）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| tag\_id | bigint | - | not null | 主键，标签id |
| tag\_name | Varchar | 255 | not null | 标签名称 |