## 江西师大校徽2校徽标准（黑）(已经裁剪) 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：**

## 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。

关键词：博客；多人；Spring Boot；Spring MVC；ThymeLeaf

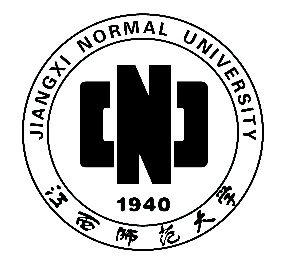
## Abstract

In recent years, the rapid development of the Internet has brought a lot of convenience and information. The ways of communication between people are more and more diversified, and the ways of information dissemination are numerous. Weibo, WeChat public account, blog, among which blogs are An important way to obtain all kinds of information is also an important carrier for recording a large amount of information in the information age.

This project is a multi-user blog system based on Spring Boot framework. It cooperates with Spring MVC framework and integrates Spring Security security framework, Spring Session user session management, Thymeleaf template engine, BootStrap front-end framework, HTML, CSS, JavaScript, MySQL. Manage projects with the Gradle project build tool and use Git to version control the project.

**Key words:** Blog; Multiplayer; Spring Boot; Spring MVC; ThymeLeaf

## 目录

 JIANGXINORMALUNIVERSITY 1

摘要 i

Abstract ii

目录 iii

第1章 绪论 1

1.1 开发背景 1

1.2 开发目的 2

第2章 需求分析 3

2.1 业务需求 3

2.2 用户需求 3

2.3 系统需求 4

第3章 开发技术 6

3.1 语言介绍 6

3.2 框架介绍 6

3.3 其他技术 8

第4章 开发平台介绍与环境搭建 10

4.1 开发环境 10

4.2 开发工具 11

第5章 系统实现详述 13

5.1 系统架构设计 13

5.2 数据库设计 14

5.3 后端实现 17

5.4 前端实现 20

参考文献 25

致谢 26

## 第1章 绪论

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。

### 1.1 开发背景

信息时代，人与人之间的沟通与信息交流逐渐多元化，博客可以说是占据了重要一席，博客完全可以按照自己的喜好记录各种各样的信息。同时也可以当成向别人展现自我想法的一扇窗口。

#### 1.1.1 博客历史

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。

#### 1.1.2 博客类别

从某种意义上说，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

一个优秀的博客能够成为传播各种正能量的载体，这也是开发本博客系统的初衷。

#### 1.2.1 博客的意义

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。

#### 1.2.2 博客的应用

博客作为一种信息的载体，记录着各种不同的信息，应用最广泛的就是作为知识分享的平台，多数人把自己学到的知识总结起来，发布到博客上，供他人进行参考，与他人进行交流讨论，不断提升自己。记录博客对于开发人员还是个必不可缺的好习惯，学习知识或者遇到问题解决后进行总结，可供以后快速参阅或者为后来人提供必要的经验。博客的应用途径可以说无所限制。

## 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 业务需求

业务需求表示客户或者组织高层次的目标。本项目致力于提供一个安全性高，用户体验佳的博客系统。

#### 2.1.1 业务目的

博客是一个公开的信息交流平台，最重要的是用户的数据安全与用户体验，注重这两方面，才能吸引更多用户。

#### 2.1.2 业务目标

在博客主页提供各个醒目清晰的模块，使用户能够快速定位到所想要的功能。同时提供各项安全措施确保用户不会进行误操作和被恶意攻击。在用户后台管理提供管理账号及各类信息的功能。

### 2.2 用户需求

用户目标是用户需要系统需要具备的功能。

#### 2.2.1 普通用户需求

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。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，系统提供了对个人文章，账号，评论进行统一管理。

#### 2.2.2 管理员需求

管理员除了可以进行普通用户的所有操作外，还需要具有对整个系统的控制权，如对所有文章进行操作，对所有账号进行操作，对所有评论进行操作。

### 2.3 功能需求

功能需求定义了开发人员开发中必须实现的功能。

#### 2.3.1 前台功能需求

在与用户交互时是前台是主要角色，在用户输入或者点击时就提醒给予用户明确提示，无需返回后台后再向用户报告错误信息，这样对用户的友好度更佳。如在注册时的用户名密码邮箱的相关字符字数限制，无需经由后台验证，直接在前台进行验证，不经由后台验证并非后台不需要验证机制。

#### 2.2.2 后台功能需求

后台需要防止未经授权的访问以及恶意的数据提交，在接受前台的数据时需要进行必要的验证，防止类似于非法字符或者SQL注入等会使网站崩溃的数据。

在后台加入缓存功能提高请求响应速度。

## 第3章 开发技术

本博客系统涉及到许多目前主流的技术，大部分技术也被几乎市面上的互联网公司所采用，开发本次系统过程中所获得的经验也对未来工作有很大帮助。

### 3.1 语言介绍

开发一个系统首先必不可少的需要选择一种以上的语言进行开发，首先是后台开发语言，其次是前端页面所使用的语言

#### 3.1.1 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。也适用于浏览器 – 服务器的网站开发。所以本次系统开发选择Java作为博客系统的主要语言，主要是因为Java众多的第三方库，让开发变得更加高效。

#### 3.1.2 HTML、CSS、JavaScript

对于前端开发，必不可少的必须使用这三种语言，HTML、CSS、Javascript把网页分成三个层次，即：结构层(HTML)—决定网页的结构和内容，表示层(CSS)—设定网页的表现样式，行为层(Javascript)—控制网页的行为。

### 3.2 框架介绍

近些年为了编程更加简单高效，网上出现了各种各样的框架用于不同的场景，前端开发框架Bootstrap、Vue、React，或者这几年最后的后台开发框架Spring Framework，优秀的框架给我们带来了极高的开发效率以及

#### 3.2.1 Spring Boot框架

Spring Framework 是一个开源的Java／Java EE全功能栈，作为一个轻量级的控制反转的容器，与传统的Java EE开发方式相对比，使用Spring 框架更加容易使用，使得应用程序的组建更加快捷简易。而本系统基于了Spring Boot进行开发，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。

#### 3.2.2 Spring Security

Spring Security作为Spring家族的一员，是一个强大的安全框架，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

#### 3.2.3 BootStrap

Bootstrap是一个基于HTML、CSS、Javascript的免费的开源前端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，使开发人员能够快速构建整个应用程序。

#### 3.2.4 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。

#### 3.2.5 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更加高效快捷，同时Spring Session并不依赖于特定于应用程序容器。它还提供透明集成组件：

HttpSession组件：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组件：提供在接收WebSocket消息时保持HttpSession活动的功能。

WebSession组件：允许以应用程序容器中立的方式替换Spring WebFlux的WebSession。

### 3.3 其他技术

语言和框架固然是整个项目的核心，但其他的技术在项目中也扮演者不可缺少而角色。

#### 3.3.1 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。这有利于改善设计沟通，缩小了设计和开发团队之间的差距。

#### 3.3.2 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的编辑器外观以及功能有三种风格：Classic、Inline、Balloon。这里选用Classic风格作为文章编辑器更适合。

#### 3.3.3 MySQL数据库

本项目选择关系型数据库作为数据存储载体，具体部署的是MySQL社区版。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。

#### 3.3.4 Redis

为了提供用户体验，加快访问速度，降低MySQL数据库压力，为系统集成了Redis缓存功能。Redis是一个非关系型数据库，简称NoSQL。NoSQL存储系统为经典RDBMS提供了水平可扩展性和速度的替代方案。在实现方面，键值存储是NoSQL空间中最主要的成员之一。Redis实现了一个具有可选持久性的分布式内存键值数据库。 Redis支持不同类型的抽象数据结构，例如字符串，列表，映射，集合，有序集，HyperLogLog，位图，流和空间索引。Redis是高级键值存储形式。它与memcached类似，但数据集不是易失性的，值可以是字符串，与memcached完全相同，但也可以是列表，集合和有序集。所有这些数据类型都可以通过原子操作来操作，以压入/弹出元素，添加/删除元素，执行服务器端并集，交集，集合之间的差异等等。

## 第4章 开发平台介绍与环境搭建

开发平台与开发环境的的搭建是整个项目开发的根基，所有代码的运行都基于此。

### 4.1 开发环境

本项目前期开发的代码编写，测试运行在Windows上，后期转向使用Mac平台进行代码编写，同时将项目远程部署到Linux服务器上进行运行调试。

#### 4.1.1 开发环境

Gradle环境：

Gradle 5.2.1

Build time: 2019-02-08 19:00:10 UTC

Revision: f02764e074c32ee8851a4e1877dd1fea8ffb7183

Kotlin DSL: 1.1.3

Kotlin: 1.3.20

Groovy: 2.5.4

Ant: Apache Ant(TM) version 1.9.13 compiled on July 10 2018

Java环境：

java version "1.8.0\_202"

Java(TM) SE Runtime Environment (build 1.8.0\_202-b08)

Java HotSpot(TM) 64-Bit Server VM (build 25.202-b08, mixed mode)

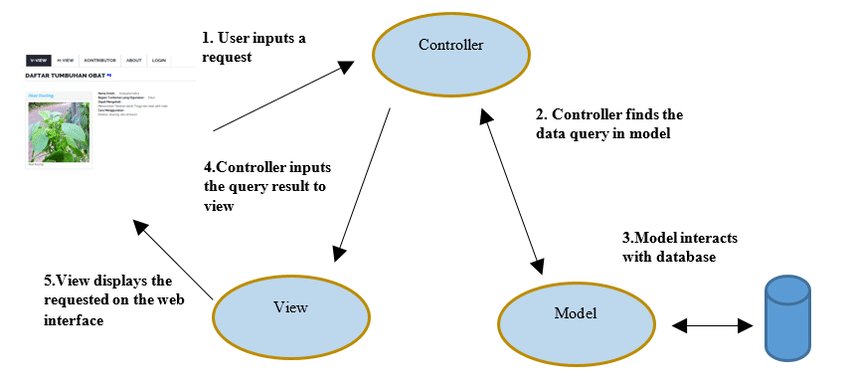
#### 4.1.2 MVC开发架构

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。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。



### 4.2 开发工具

开发工具的选择对系统开发效率影响甚大，一个优秀的开发工具给开发者的体验应该是愉快的。都说工欲善其事必先利其器，选择一款顺手的开发工具是必不可少的。

#### 4.2.1 Java 集成开发环境

现在主流的Java 集成开发环境有商业化的IntelliJ IDEA与开源的Eclipse和NetBeans。本项目使用的是IntelliJ IDEA 2018.3.5 (Ultimate Edition)进行开发与测试。

#### 4.2.2 Web容器

常见的Web容器有WebLogic、WebSphere、JBoss、Tomcat，前两者为商业平台，后两者为开源项目，Tomcat为Apache软件基金会中的Jakarta 项目中的一个核心项目。本项目就选用Tomcat作为博客系统的Web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。

## 第5章 系统实现详述

在平台，环境都搭建妥当后，系统实现就是整个项目的核心内容。

### 5.1 系统架构设计

在开发流程中，架构设计往往是最先开始的，对架构进行谨慎设计，为接下来的的开发工作奠定了基础。

#### 5.1.1 系统模块设计

#### 5.1.2 系统模块简述

**普通用户、管理员：**

个人账号管理：修改个人信息，支持修改密码，邮箱和头像

个人文章管理：管理个人文章，支持删除修改操作，和是否公开操作，可预览文章。

个人评论管理：管理个人发布的评论，支持删除和修改操作。

**管理员：**

分类管理：管理分类，支持增加新分类，增加子分类，删除分类，级联删除分类下的子分类以及分类和子分类下的所有文章。

标签管理：管理标签，增加标签以及删除标签，同时删除标签与文章的关联关系。

文章管理：管理所有文章，支持删除文章，预览文章，是否公开操作。

评论管理：管理所有评论，支持删除操作，可跳转到评论所在文章。

账号管理：管理所有账号，支持封禁用户，删除用户（同时级联删除账号下的所有文章以及发布的所有评论），分配用户权限。

### 5.2 数据库设计

数据库是整个系统最重要的底层支撑，用户对系统的一切的操作都是基于数据库的增删改查操作，所以，如何设计数据库表就成为了必须谨慎考虑的环节，数据库设计的优劣直接影响到系统的性能，也影响到用户的体验。

#### 5.2.1 数据库结构

开发中使用了Hibernate框架，Hibernate对Entity自动映射到数据库中，对每一个Entity生成对应的数据库表，对象间的各种关联关系，如单向或者双向的一对多，一对一，多对一映射关系。Hibernate在映射成表时加入外键或新建关联表来维护表见的关系。

#### 5.2.2 数据库表设计

系统中主要包括6个Entity，账号，文章，评论，标签，分类，权限，各自映射成表以及Entity之间的关联表。

**1、account（账号表）如表5.1所示**

表5.1 账号表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| account\_id | bigint | - | N | 主键，账号id |
| account\_non\_expired | bit | - | N | 账号是否过期 |
| account\_non\_locked | bit | - | N | 账号是否被锁定 |
| avatar | varchar | 255 | N | 头像地址 |
| create\_date | datetime | - | N | 创建日期 |
| credentials\_non\_expired | bit | - | N | 凭证是否过期 |
| email | varchar | 255 | N | 电子邮箱 |
| enabled | bit | - | N | 是否激活状态 |
| last\_modified\_date | datetime | - | N | 上次修改时间 |
| password | varchar | 16 | N | 密码 |
| username | varchar | 16 | N | 用户名 |

**2、account\_authority（账号权限表）如表5.2所示**

表5.2 账号权限表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| account\_id | bigint | - | N | 主键，账号id |
| authority\_id | bigint | - | N | 主键，权限id |

**3、article（文章表）如表5.3所示**

表5.3 文章表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| article\_id | bigint | - | N | 主键，文章id |
| content | mediumtext | - | N | 富文本内容 |
| create\_date | datetime | - | N | 创建时间 |
| publish | bit | - | N | 是否公开 |
| stick | bit | - | N | 是否置顶 |
| last\_modified\_date | datetime | - | N | 上次修改时间 |
| page\_view | bigint | - | N | 浏览量 |
| raw\_content | mediumtext | - | N | 纯文本内容 |
| title | varchar | 255 | N | 标题 |
| account\_account\_id | bigint | - | N | 文章所属用户id |
| category\_category\_id | bigint | - | N | 所属分类id |

**4、article\_tag（文章标签表）如表5.4所示**

表5.4 文章标签表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| tag\_id | bigint | - | N | 主键，标签id |
| article\_id | bigint | - | N | 主键，文章id |

**5、authority（权限表）如表5.5所示**

表5.5 权限表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| authority\_id | bigint | - | N | 主键，权限id |
| authority\_name | Varchar | 255 | N | 权限名称 |
| create\_date | datetime | - | N | 创建时间 |
| last\_modified\_date | datetime | - | N | 上次修改时间 |

**6、category（分类表）如表5.6所示**

表5.6 分类表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| category\_id | bigint | - | N | 主键，分类id |
| category\_name | Varchar | 255 | N | 分类名称 |

**7、comment（评论表）如表5.7所示**

表5.7 评论表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| comment\_id | bigint | - | N | 主键，评论id |
| content | Varchar | 255 | N | 评论内容 |
| create\_date | datetime | - | N | 创建时间 |
| last\_modified\_date | datetime | - | N | 上次修改时间 |
| account\_account\_id | bigint | - | N | 评论所属账号id |
| article\_article\_id | bigint | - | N | 评论所属文章id |

8、favorites\_article（用户收藏文章表）如表5.8所示

表5.8 用户收藏文章表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| account\_id | bigint | - | N | 主键，账号id |
| article\_id | bigint | - | N | 主键，文章id |

9、tag（标签表）如表5.9所示

表5.9 标签表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| tag\_id | bigint | - | N | 主键，标签id |
| tag\_name | Varchar | 255 | N | 标签名称 |

10、spring\_session（Spring Session表）如表5.10所示

表5.10 Spring Session表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| PRIMARY\_ID | char | 36 | N | 主键id |
| SESSION\_ID | char | 36 | N | Session id |
| CREATION\_TIME | bigint | - | N | 创建时间 |
| LAST\_ACCESS\_TIME | bigint | - | N | 上次访问时间 |
| MAX\_INACTIVE\_INTERVAL | int | - | N | 最大不活跃间隔 |
| EXPIRY\_TIME | bigint | - | N | 过期时间 |
| PRINCIPAL\_NAME | varchar | 100 | N | 用户名 |

10、spring\_session\_attributes（Spring Session属性表）如表5.10所示

表5.10 Spring Session属性表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 是否为空 | 描述 |
| SESSION\_PRIMARY\_ID | char | 36 | N | session主键id |
| ATTRIBUTE\_NAME | varchar | 200 | N | 主键，属性名 |
| ATTRIBUTE\_BYTES | blob | - | N | 属性内容 |

### 5.3 后端实现

后端层次结构为

#### 5.3.1 层次结构介绍

View：结合Controller层，向用户显示页面。

Controller：处理用户请求，处理业务逻辑，调用Service层接口。

Service：具体业务操作实现，调用Repository层。

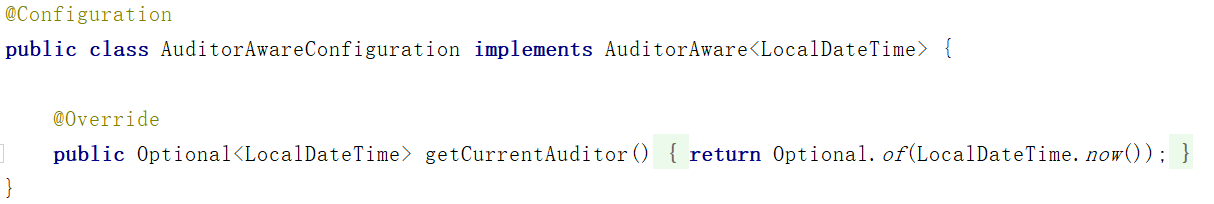
Repository：负责与数据库交互，数据业务处理。同时负责对象持久化

#### 5.3.2关键配置实现

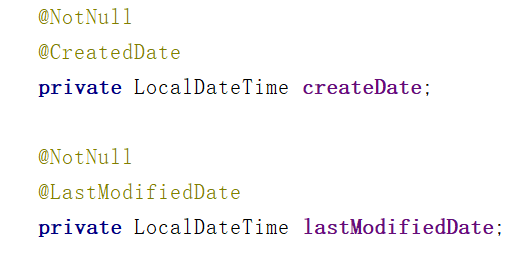
Spring Security作为集成到系统的安全框架，配置至关重要。在这里配置为开启表单登录功能，关闭Http Basic登录功能，开启Remember Me功能，关闭CSFR跨站请求伪造防御。同时开启了Sprin Session的Session限制，一个账号只能同时允许一个Session在线，如果再使用其他设备登录，上一个Session就自动失效。



Spring Data JPA审计功能能够自动对每个对象进行跟踪审计，可记录创建时间，最后修改时间，创建人和最后修改人，首先在新建审计配置类，在配置类中重写getCurrentAuditor方法。



使用时只需再实体类中里添加两个属性即可，这里使用@CreatedDate、@LastModifiedDate两个注解，能够自动记录对象首次创建时间和上次修改时间。



Spring Session能够轻松的管理用户会话，要开启Spring Session，需要新建配置类，使用注解@ EnableJdbcHttpSession即可继承Spring Session



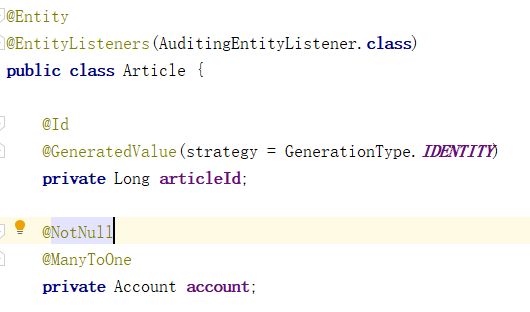
另外，我们还需要通过拓展类WebMvcConfigurationSupport，以配置WebMVC相关属性，如需要重写addArgumentResolvers方法，以避免在使用Spring Boot的分页功能时导入错误的Pageable类。同时需要重写addResourceHandlers方法，以处理资源文件映射问题。WebMvcConfigurationSupport还提供了addViewControllers方法，对能够减少多余的controller代码。



#### 5.3.4 实体类实现介绍

Spring Data JPA的默认实现库时Hibernate，Hibernate能够自动创建SQL数据库表，如果需要MySQL中的表互相关联关系，主键自增，表中字段限制等功能，只需要在实体类中的相应属性加上注解，在初次运行时即会根据注解自动处理表中定义以及表间关联。

需要开启实体映射功能，在类上加上@Entity注解，同时开启JPA审计功能，需要另外加上注解@EntityListeners(AuditingEntityListener.class)开启。



注解@Id定义字段为表中主键，将注解@OneToMany、@ManyToOne、@ManyToMany添加到字段上，Hibernate即可自动处理表间的关系，项目中均使用了双向关联，即双发都可以找到另一方。在注解后指定cascade、fetch、orphanRemoval属性的值，cascade可以定义表间的级联操作，如级联删除，级联更新，fetch可以定义在获取当前实体对象时加载子属性的数据，分为懒加载或者立即加载，orphanRemoval则指定了当对象间关系断开，是否删除下级对象。



### 5.4 前端实现

前台使用Bootstrap框架，使用前端框架的好处是能够快速构建页面，避免了开发人员编写大量繁杂的重复代码，使减轻前端工作负担，能够更专注于后端开发。

#### 5.4.1 首页UI介绍

博客整体风格简约为主，首页头部设计为分类导航栏，在不同宽度下可隐藏部分分类，滑动可见全部分类，在首页放置了醒目的置顶文章轮播幻灯片

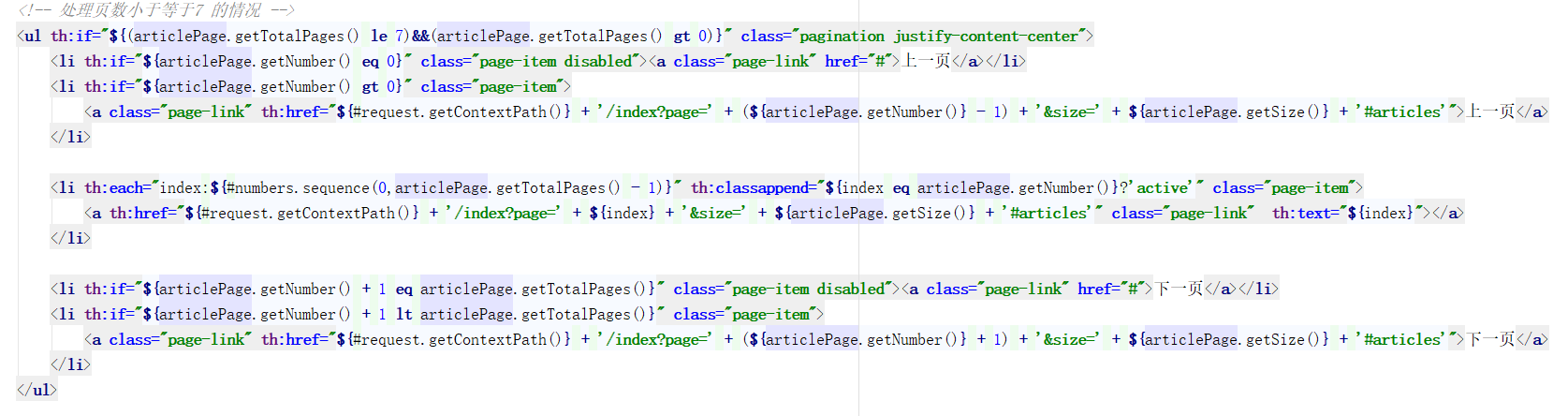


下面文章部分采用分页处理，每页显示10篇文章。右侧放置标签栏，关于此站栏和友情链接栏。



在分页功能中，根据总页数和当前页的不同，分页按钮也会有不同显示。以下是分页代码实现。

当总页数小于7时：



当总页数大于7时：





#### 5.4.2 文章详情页面UI设计

在文章详情页面，引用了相同的头部分片，减少了重复代码带来的维护难度了，文章详情页在主要部分显示了文章信息，作者头像，以及创建时间和上次修改时间。



在评论区，可以对文章进行评论，与博主进行互动，右侧同样带有标签栏，关于栏和友情链接。



## 参考文献

[1] 陈文伟.决策支持系统及其开发[M].北京:清华大学出版社, 2000.12-15.

[2] 杨南昌.基于多元智能(MI)的个性化学习研究[D].南昌：江西师范大学，2003.

[3]张朝晖,陆玉昌,张钹.利用神经网络发现分类规则[J].计算机学报,1999,(1):108-111.

[4] OU J P，SOONG T T，etal. Recent advance in research on applications of passive energy dissipation systems［J］.EarthquackEng,1997,38(3):358-361.

[5] 钟文发.非线性规划在可燃毒物配置中的应用［A］.赵炜.运筹学的理论与应用--中国运筹学会论文集［C］.西安：西安电子科技大学出版社，1996.468.

[6] Yang Hong-wei and Li Guang-jie.An adaptive ICI self cancellation scheme to compensate the frequency offset for OFDM system [A].Vehicular Technology Conference,2003.VTC 2003-Spring[C].The 57th IEEE Semiannual, 2003,4,22-25.

[7] 谢希德.创造学习的新思路［N］.人民日报，1998-12-25（10）.

[8] 王明亮.关于中国学术期刊标准化数据库系统工程的进展［EB/OL］. http://www.cajcd.edu.cn/pub/wml.html ，1998-08-16/1998-10-01.

[9] Eric Plotnick. Concept Mapping: A Graphical System for Understanding the Relationship Between Concepts [EB/OL]. http://ericit.org/digests/EDO-IR- 1997-05.shtml,2002-06-05.

**(参考文献要求10篇以上)**

## 致谢

感谢江西师范大学