本科生课程设计报告

课程名称 软件工程课程设计

课程编号 19630360s

学生姓名 闫锡航

学 A10170210

班 微机 1707

提交日期 2020 年 3 月 2 日

2020 年 4 月 22 日

基于 SpringBoot 开发的个人博客项目

计算机科学与技术

微机 1707 闫锡航

A10170210

目录

一、 前言

1.1 个人概述

出于个人技术需要，以及本人对前端和后端开发技术较为好奇的因素，本次工程实践项目我选择自行开发一款个人博客的项目。此项目实现的博客功能虽不如 CSDN、博客园等知名博客网站功能齐全、美观大方，麻雀虽小，但五脏俱全，基本的功能都会有。此次工程项目实践意在对网站类项目形成一个初步的认知，意在了解各主流框架以及开发工具的使用。

纸上得来终觉浅，绝知此事要躬行。只有亲身实践才能更好地学习理论，即便刚开始的时候会有很多坎坷，但这是学习新知识必须要迈出的第一步。本项目选用 Sementic UI 前端框架，SpringBoot 后端框架，结合 MySQL 数据库，完成了一个简单的个人博客。代码量很多，此报告中只展示关键部分，项目源码我已上传至 GitHub，若有兴趣可以访问我的 GitHub 进行查看。

此报告利用 LaTeX 进行排版，部分名词有超链接部分，若有兴趣则可翻看电子版即 pdf 文件。流程图利用 MindMaster 制作。

1.2 系统主要功能简介

系统功能主要分为前端展示和后端管理。前端展示主要分为首页、分类、标签和归档，具体内容主要包括博客详情、展示标签分类、按时间序列展示内容、tag 分类等。后端管理主要分为管理员登录、博客管理、分类管理以及标签管理内容，具体内功主要包括发布、删除、查询（博客、标签以及分类）。

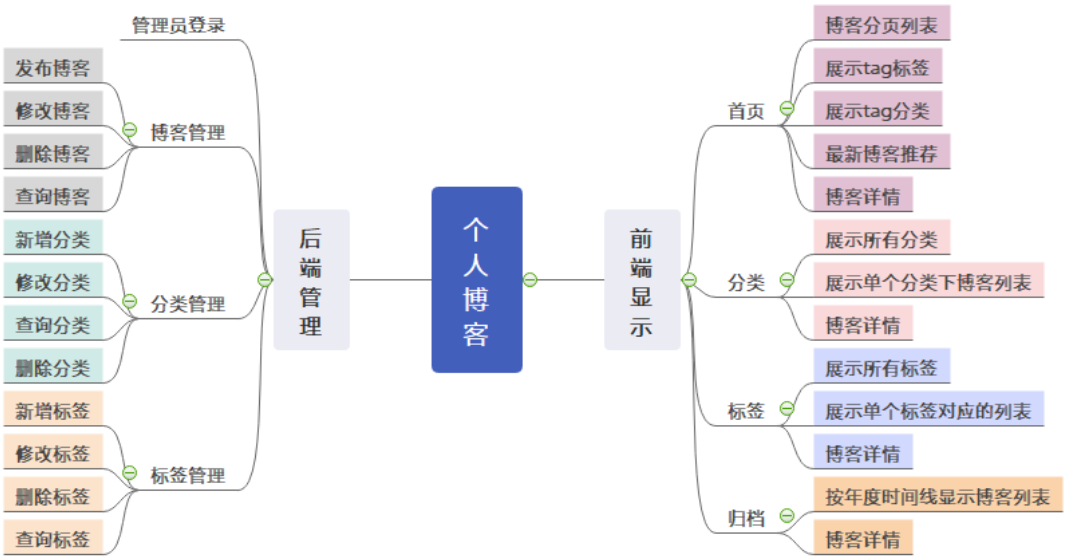

图 1.1 系统主要功能

二、 开发工具与环境

- **IntelliJ IDEA 2020.1 x64.IDEA** 全称 IntelliJ IDEA，是 java 编程语言开发的集成环境。IntelliJ 在业界被公认为最好的 java 开发工具，尤其在智能代码助手、代码自动提示、重构、J2EE 支持、各类版本工具 (git、svn 等)、JUnit、CVS 整合、代码分析、创新的 GUI 设计等方面的功能可以说是超常的。
- **WebStorm 2020.1 x64** WebStorm 是 jetbrains 公司旗下一款 JavaScript 开发工具。已经被广大中国 JS 开发者誉为“Web 前端开发神器”、“最强大的 HTML5 编辑器”、“最智能的 JavaScript IDE”等。与 IntelliJ IDEA 同源，继承了 IntelliJ IDEA 强大的 JS 部分的功能。
- **Navicat Premium 12** Navicat premium 是一款数据库管理工具, 是一个可多重连线资料库的管理工具，它可以让你以单一程式同时连线到 MySQL、SQLite、Oracle 及 PostgreSQL 资料库，让管理不同类型的资料库更加的方便。
- **MySQL Workbench 8.0 CE** MySQL 是一个关系型数据库管理系统，由瑞典 MySQL AB 公司开发，属于 Oracle 旗下产品。
- **Maven** Maven 项目对象模型 (POM)，可以通过一小段描述信息来管理项目的构建，报告和文档的项目管理工具软件。
- **JDK13** JDK 是 Java 语言的软件开发工具包，主要用于移动设备、嵌入式设备上的 java 应用程序。JDK 是整个 java 开发的核心，它包含了 JAVA 的运行环境（JVM+Java 系统类库）和 JAVA 工具。
- **Axure RP 8** Axure RP 是一款专业的快速原型设计工具。Axure（发音：Ack-sure），代表美国 Axure 公司；RP 则是 Rapid Prototyping（快速原型）的缩写。

三、 系统需求分析

3.1 用户故事

什么是用户故事? 用户故事是敏捷框架中的一种开发方法。可以帮助开发者转换视角，以用户的角度更好地把握需求，从而实现具有商业价值的功能。

用户故事模板：

- As a(role of user), I want (some feature) so that (some business value).
- 作为一个 (某个角色) 的使用者，我可以做 (某个功能) 事情，如此可以有 (某个商业价值) 的好处.

关键点：角色、功能、商业价值。

举例：

- 作为一个招聘网站的**注册用户**，我想**查看最近 3 天发布的招聘信息**，以便于**了解最新的招聘信息**。
- 作为公司，可以张贴新工作的信息。

由此，个人博客的用户故事可以描述为：
角色：访客，管理员 (我)

- 访客，可以分页查看所有的博客
- 访客，可以快速查看博客数最多的 6 个分类
- 访客，可以查看所有的分类
- 访客，可以查看某个分类下的博客列表
- 访客，可以快速标记博客最多的 10 个标签
- 访客，可以查看所有的标签
- 访客，可以查看某个标签下的博客列表
- 访客，可以查看某个标签下的博客列表
- 访客，可以快速查看最新的推荐博客
- 访客，可以用关键字全局搜索博客
- 访客，可以查看单个博客内容
- 访客，可以对博客内容进行评论
- 访客，可以赞赏博客内容
- 访客，可以微信扫码阅读博客内容
- 访客，可以在首页扫描公众号二维码关注我
- 我，可以管理博客：
  - 我，可以新增一个分类
  - 我，可以修改一个分类

  - 我，可以删除一个分类
  - 我，可以根据分类名称查询分类

- 我，可以根据分类名称查询分类:
  - 我，可以新增一个标签
  - 我，可以修改一个标签
  - 我，可以删除一个标签
  - 我，可以根据名称查询标签

3.2 功能需求

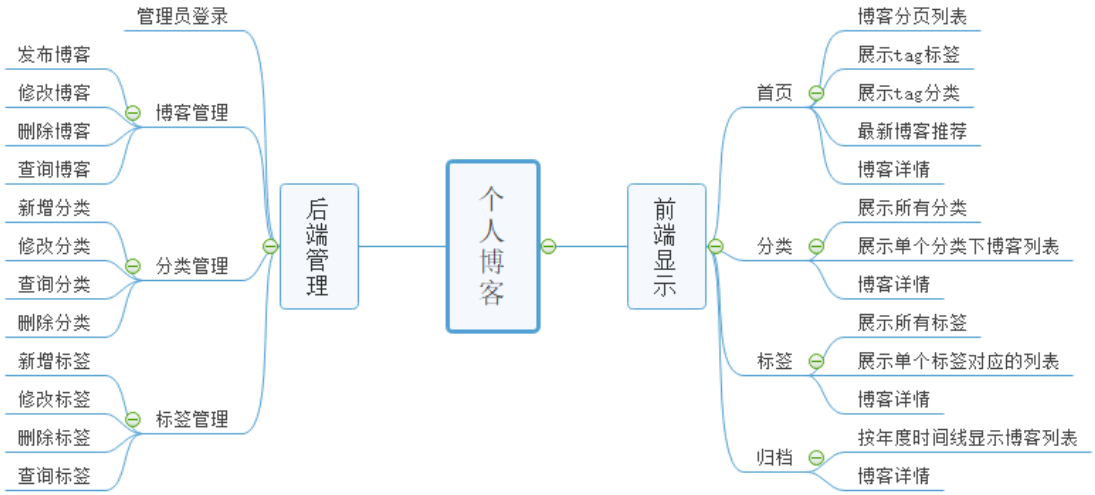

图 **3.1** 功能需求

四、 系统概要设计

4.1 页面设计与开发

4.1.1 设计

页面规划：

前端展示：首页、详情页、分类、标签、归档、AboutMe

后台管理：模板页

4.1.2 页面开发

- **Semantic UI 官网**
- **Semantic UI 中文官网**
- **WebStorm 下载与破解**
- **背景图片资源**

4.1.3 插件集成

- **编辑器 MarkDown**
- **内容排版 typo.css**
- **动画 animate.css**
- **代码高亮 prism**
- **目录生成 Tocbot**
- **滚动侦测 Waypoints**
- **平滑滚动 jQuery.scrollTo**
- **二维码生成 qrcode.js**

4.2 框架搭建

IDEA 下载 https://www.jetbrains.com/idea/

4.3 构建与配置

1. **引入 SpringBoot 模块**
    - web
    - JPA
    - Thymeleaf
    - MySQL
    - Aspects
    - DevTools

2. **application.yml 配置**

   - 使用 thymeleaf 3

     pom.xml:

     application.yml:
   - 数据库连接配置
   - 日志配置

     application.yml:

     logback-spring.xml:
   - 生产环境和开发环境配置
     - application-dev.yml
     - application-pro.ym

3. 异常处理

   (a) 定义错误界面

   - 404
   - 500
   - error

   (b) 全局处理异常

   - 统一处理异常
   - 错误页面异常信息显示处理

   (c) 资源找不到异常

4. 日志处理

   (a) 记录日志内容：

   - 请求 url
   - 访问者 ip
   - 调用方法 classMethod
   - 参数 args
   - 返回内容

   (b) 记录日志类：

5. 页面设计

(a) **静态页面导入 project**

(b) **thymeleaf 布局**

- 定义 fragment
- 使用 fragment 布局

(c) **错误页面美化**

4.4 设计与规范

4.4.1 实体设计

实体类：

- 博客 Blog
- 博客分类 Type
- 博客标签 Tag
- 博客评论 Comment
- 用户 User

4.4.2 应用分层

应用分层：

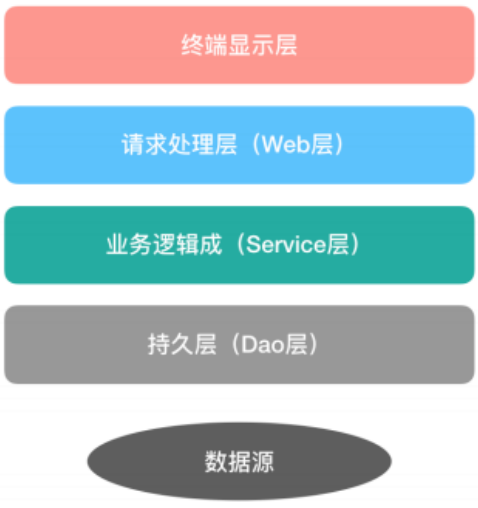

图 **4.1** 应用分层

4.4.3 命名约定

Service/DAO 层命名规定：

- 获取单个对象的方法用 get 做前缀
- 获取多个对象的方法用 list 做前缀
- 获取统计值的方法同 count 做前缀
- 插入的方法用 save(推荐) 或 insert 做前缀
- 删除的方法用 remove(推荐) 做前缀
- 修改的方法用 update 做前缀

4.5 后台管理功能实现

4.5.1 登录

1. 构建登录页面和后台管理首页
2. UserService 和 UserRepository
3. LoginController 实现登录
4. MD5 加密
5. 登录拦截器

4.5.2 分类管理

1. 分类管理页面
2. 分类列表分页
3. 分类新增、修改、删除

4.5.3 标签管理

4.5.4 博客管理

1. 博客分页查询
2. 博客新增
3. 博客修改
4. 博客删除

4.6 前端展示功能实现

4.6.1 首页展示

1. 博客列表
2. top 分类
3. top 标签
4. 最新博客推荐
5. 博客详情

MarkDown 转换 HTML

1. commonmark-java https://github.com/atlassian/commonmark-java
2. pom.xml 引用 commonmark 和扩展插件

评论功能：

- 评论信息提交与回复功能
- 评论信息列表展示功能
- 管理员回复评论功能

4.6.2 分类页

4.6.3 标签页

4.6.4 归档也

4.6.5 AboutMe

五、 系统详细设计

5.1 页面设计

在建立博客之前，我们应该明确页面的样式是什么样子的，这里我选择一种 teal 颜色的博客模板。页面的 head 分为首页、分类、标签、归档和 AboutMe。所有页面都有相同的 head 和 bottom，head 主要提供一级主题，bottom 主要描述一些博主个人信息以及维权相关内容。

1. 首页

在首页中，应该呈现所有博主所发布的博客内容 (这里出于便捷暂时发布一样的博客)，并且每个内容模块应该呈现博客博主 ID、发布时间、浏览数，以及封面配图等元素。页面右方展示一些分类和标签，与其他一级标题相对应，关于我的内容以二维码的形式给出。

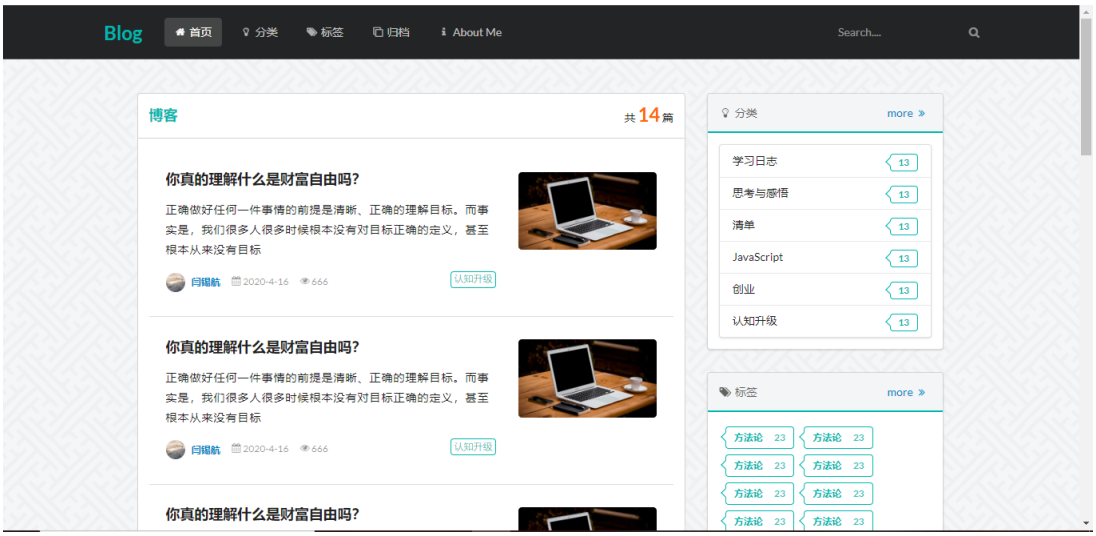

图 **5.1** 首页

2. 分类

分类页面中，以首页为基础添加了分类的内容，删除了右边详细部分。

图 **5.2** 分类

3. 标签

标签页面中，以首页为基础添加了标签内容，删除了右边详细部分。

图 **5.3** 标签

4. 归档

   归档页面中，应该呈现年份和博客摘要内容，并简要标记博客来源 (原创、转载、翻译)

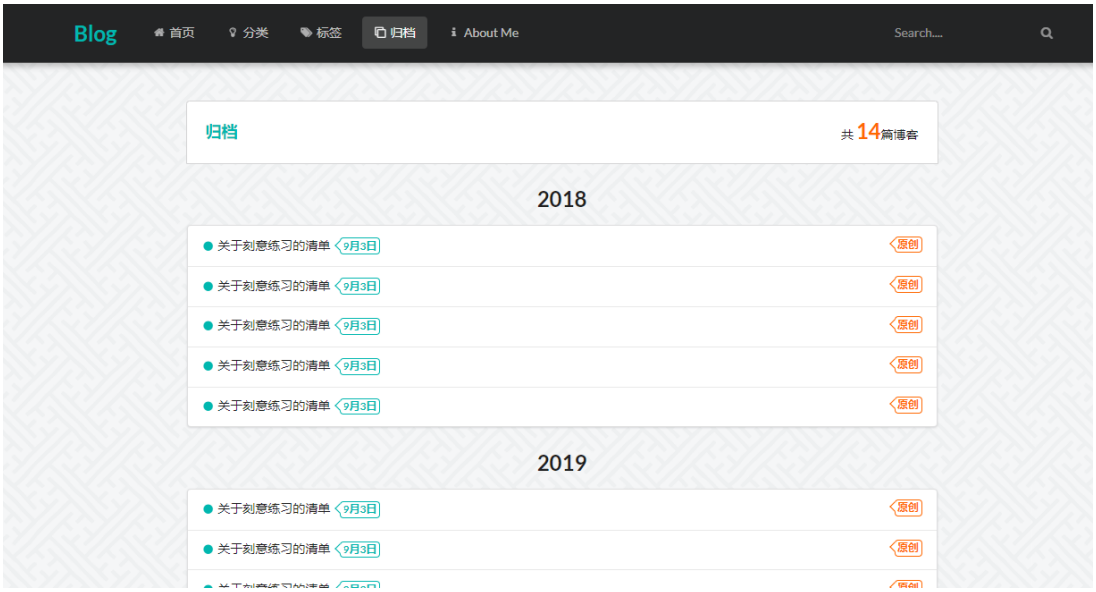

图 **5.4** 归档

5. AboutMe

   个人信息，包括个人简介等

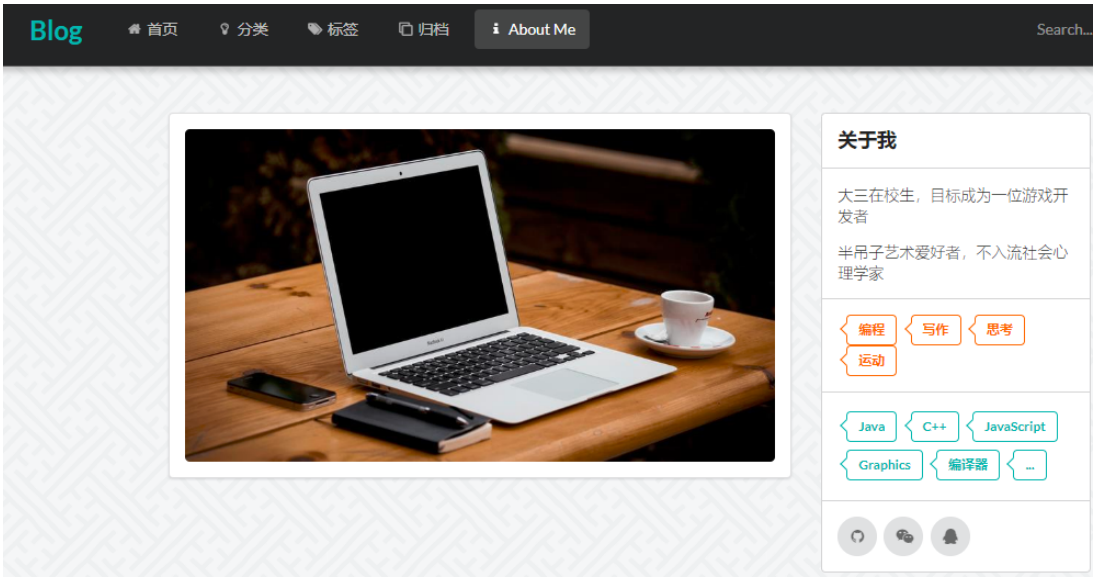

图 5.5 AboutMe

5.2 页面开发

5.2.1 首页开发

关键部分代码：

```
<div class="ui padded vertical segment m-padded-tb-large">
<div class="ui mobile reversed stackable grid">
<div class="ui eleven wide column">
<h3 class="ui header">你真的理解什么是财富自由吗？</h3>
<p class="m-text">正确做好任何一件事情的前提是清晰、正确的理解目标。
而事实是，我们很多人很多时候根本没有对目标正确的定义，甚至根本从来没
有目标</p>
<div class="ui grid">
<div class="eleven wide column">
<div class="ui mini horizontal link list">
<div class="item">
<img src="https://picsum.photos/id/1000/100/100" alt="" class="ui
    avatar image"><!--avatar小头像-->
<div class="content"><a href="#"
    class="header">闫锡航</a></div><!--图片后加一些文字content
    href="#"似乎是链接-->
</div>
<div class="item">
<i class="calendar icon"></i>2020-4-16<!--日期(日历图标)-->
</div>
```

```
<div class="item">
<i class="eye icon"></i>666<!--浏览次数图标eye-->
</div>
</div>
</div>
<div class="right aligned five wide column">
<a href="#" target="_blank" class="ui teal basic label m-padded-tiny
    m-text-thin">认知升级</a>
</div>
</div>
</div>

<div class="five wide column">
<a href="#" target="_blank">
<img src="https://picsum.photos/id/0/800/450" alt="" class="ui rounded
    image">
</a>
</div>

</div>
</div>
```

5.2.2 分类页面开发

关键部分代码：

```
<div class="ui attached segment m-padded-tb-large">
<div class="ui labeled button m-margin-tb-tiny">
<a href="#" class="ui basic teal button">思考与感悟</a>
<div class="ui basic teal left pointing label">24</div>
</div>

<div class="ui labeled button m-margin-tb-tiny">
<a href="#" class="ui basic button">开发者手册</a>
<div class="ui basic left pointing label">24</div>
</div>

<div class="ui labeled button m-margin-tb-tiny">
<a href="#" class="ui basic button">清单</a>
<div class="ui basic left pointing label">24</div>
</div>
```

```
        </div>
```

5.2.3 标签页面开发

关键部分代码：

```
        <!--header-->
        <div class="ui top attached segment">
        <div class="ui middle aligned two column grid">
        <div class="column">
        <h3 class="ui teal header">标签</h3>
        </div>
        <div class="right aligned column">
        共 <h2 class="ui orange header m-inline-block">14</h2> 个
        </div>
        </div>
        </div>

        <div class="ui attached segment m-padded-tb-large">
        <a href="#" target="_blank" class="ui teal basic left pointing large
            label m-margin-tb-tiny">
        方法论 <div class="detail">23</div>
        </a>

        </div>
```

5.2.4 归档页面开发

关键部分代码：

```
        <h2 class="ui center aligned header">2018</h2>
        <div class="ui fluid vertical menu">
        <a href="#" target="_blank" class="item">
        <span>
        <i class="circle icon teal mini"></i>关于刻意练习的清单
        <div class="ui teal basic left pointing label m-padded-mini
            ">9月3日</div>
        </span>
        <div class="ui orange basic left pointing label m-padded-mini
            ">原创</div>
        </a>
```

```
        </div>
        <h2 class="ui center aligned header">2019</h2>
        <div class="ui fluid vertical menu">
        <a href="#" target="_blank" class="item">
        <span>
        <i class="circle icon teal mini"></i>关于刻意练习的清单
        <div class="ui teal basic left pointing label m-padded-mini
            ">9月3日</div>
        </span>

        <div class="ui orange basic left pointing label m-padded-mini
            ">原创</div>
        </a>
```

5.2.5 AboutMe 页面开发

关键部分代码：

```
        <div class="five wide column">
          <div class="ui top attached segment">
          <div class="ui header">关于我</div>
          </div>
          <div class="ui attached segment">
          <p class="m-text-thin">大三在校生，目标成为一位游戏开发者</p>
          <p class="m-text-thin">半吊子艺术爱好者，不入流社会心理学家</p>
          </div>
          <div class="ui attached segment">
          <div class="ui orange basic left pointing label">编程</div>
          <div class="ui orange basic left pointing label">写作</div>
          <div class="ui orange basic left pointing label">思考</div>
          <div class="ui orange basic left pointing label">运动</div>
          </div>
        </div>
```

5.2.6 部分插件集成引入

MarkDown 引入，Semantic UI 引入，内容排版 typo.css，动画 Animate.css，代码高亮 prism，目录生成 Tocbot，滚动侦测 waypoints，平滑滚动 jquery.scrollTo，二维码生成 qrcode

```
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>博客详情</title>
<link rel="stylesheet"
    href="https://cdn.jsdelivr.net/semantic-ui/2.2.4/semantic.min.css">
<link rel="stylesheet" href="./static/css/typo.css">
<link rel="stylesheet" href="./static/css/animate.css">
<link rel="stylesheet" href="./static/lib/prism/prism.css">
<link rel="stylesheet" href="./static/lib/tocbot/tocbot.css">
<link rel="stylesheet" href="static/css/me.css">
</head>

<script
    src="https://cdn.jsdelivr.net/npm/jquery@3.2/dist/jquery.min.js">
</script>
<script
    src="https://cdn.jsdelivr.net/semantic-ui/2.2.4/semantic.min.js">
</script>
<script
    src="//cdn.jsdelivr.net/npm/jquery.scrollto@2.1.2/jquery.scrollTo
.min.js"></script><!--平滑地返回Top-->
<script src="./static/lib/prism/prism.js"></script>
<script src="./static/lib/tocbot/tocbot.min.js"></script>
<script src="./static/lib/qrcode/qrcode.min.js"></script>
<script src="./static/lib/waypoints/jquery.waypoints.min.js"></script>
```

5.3 框架搭建

1. **引入 SpringBoot 模块**

   - web
   - JPA
   - Thymeleaf
   - MySQL
   - Aspects
   - DevTools

   部分引入模块，篇幅有限，只展示部分，未显示部分只有少量差异。

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-aop</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
```

2. **application.yml 配置**

- 使用 thymeleaf 3

  pom.xml:

```
<thymeleaf.version>3.0.2.RELEASE</thymeleaf.version>
<thymeleaf-layout-dialect.version>2.1.1</thymeleaf-
layoutdialect.version>
```

  application.yml:

```
spring:
thymeleaf:
mode: HTML
```

- 数据库连接配置

```
spring:
datasource:
driver-class-name: com.mysql.jdbc.Driver
url: jdbc:mysql://localhost:3306/blog?
useUnicode=true&characterEncoding=utf-8
username: root
password: root
jpa:
hibernate:
ddl-auto: update
show-sql: true
```

- 日志配置 application.yml:

```
logging:
level:
```

```
root: info
com.imcoding: debug
file: log/imcoding.log
```

logback-spring.xml:

- 生产环境和开发环境配置
  - application-dev.yml
  - application-pro.ym

3. 异常处理

(a) 定义错误界面

- 404

404.html 部分源码

```
<!--导航-->
<nav th:replace="_fragments :: menu(0)"
    class="ui inverted attached segment
    m-padded-tb-mini m-shadow-small" >
<div class="ui container">
<div class="ui inverted secondary stackable
    menu">
<h2 class="ui teal header item">Blog</h2>
<a href="#" class=" m-item item
    m-mobile-hide"><i class="mini home
    icon"></i>首页</a>
<a href="#" class="m-item item
    m-mobile-hide"><i class="mini idea
    icon"></i>分类</a>
<a href="#" class="m-item item
    m-mobile-hide"><i class="mini tags
    icon"></i>标签</a>
<a href="#" class="m-item item
    m-mobile-hide"><i class="mini clone
    icon"></i>归档</a>
<a href="#" class="m-item item
    m-mobile-hide"><i class="mini info
    icon"></i>关于我</a>
<div class="right m-item item m-mobile-hide">
<div class="ui icon inverted transparent input
    m-margin-tb-tiny">
```

```
<input type="text" placeholder="Search....">
<i class="search link icon"></i>
</div>
</div>
</div>
</div>
<a href="#" class="ui menu toggle black icon
    button m-right-top m-mobile-show">
<i class="sidebar icon"></i>
</a>
</nav>

<div class="m-container-small
    m-padded-tb-massive">
<div class="ui error message m-padded-tb-huge"
    >
<div class="ui contianer">
<h2>404</h2>
<p>对不起，你访问的资源不存在</p>
</div>
</div>
</div>
```

- 500
- error

(b) 全局处理异常

- 统一处理异常:

```
@ControllerAdvice
public class ControllerExceptionHandler {
  private final Logger logger =
  LoggerFactory.getLogger(ControllerException
  Handler.class);
  @ExceptionHandler({Exception.class})
  public ModelAndView
      handleException(HttpServletRequest
      request,
  Exception e) throws Exception {
  logger.error("Request URL : {} , Exception
      : {}",
  request.getRequestURL(), e);
```

```
        if (AnnotationUtils.findAnnotation(e.
        getClass(),
        ResponseStatus.class) != null) {
        throw e;
        }
        ModelAndView mav = new ModelAndView();
        mav.addObject("url",
            request.getRequestURL());
        mav.addObject("exception", e);
        mav.setViewName("error/error");
        return mav;
        }
    }
```

- 错误页面异常信息显示处理:

```
<div>
<div th:utext="'&lt;!--'"
    th:remove="tag"></div>
<div th:utext="'Failed Request URL : ' +
    ${url}" th:remove="tag">
</div>
<div th:utext="'Exception message : ' +
    ${exception.message}"
th:remove="tag"></div>
<ul th:remove="tag">
<li th:each="st : ${exception.stackTrace}"
    th:remove="tag"><span
th:utext="${st}" th:remove="tag"></span></li>
</ul>
<div th:utext="'--&gt;'" th:remove="tag"></div>
</div>
```

(c) 资源找不到异常

```
@ResponseStatus(HttpStatus.NOT_FOUND)
public class NotFoundExcepiton extends
    RuntimeException {
  public NotFoundExcepiton() {
  }
  public NotFoundExcepiton(String message) {
    super(message);
```

```
    }
    public NotFoundExcepiton(String message, Throwable
        cause) {
      super(message, cause);
    }
  }
```

4. 日志处理

   (a) 记录日志内容：

   - 请求 url
   - 访问者 ip
   - 调用方法 classMethod
   - 参数 args
   - 返回内容

```
private class RequestLog {
  private String url;
  private String ip;
  private String classMethod;
  private Object[] args;

  public RequestLog(String url, String ip, String
      classMethod, Object[] args) {
  this.url = url;
  this.ip = ip;
  this.classMethod = classMethod;
  this.args = args;
}
```

   (b) 记录日志类：

```
@Aspect
@Component
public class LogAspect {
  private final Logger logger =
  LoggerFactory.getLogger(this.getClass());
}
```

5. 页面设计

(a) **静态页面导入 project**

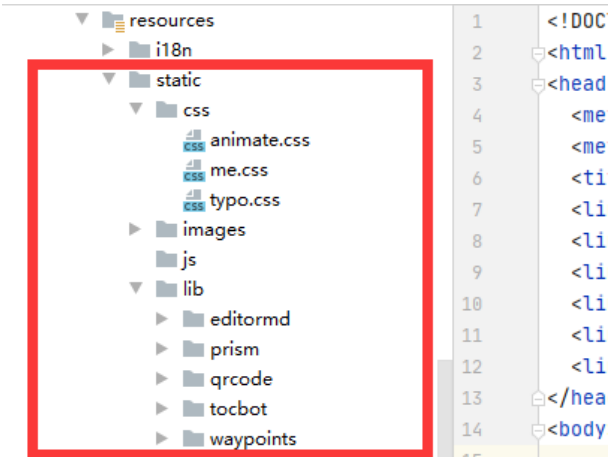

图 **5.6** 重新导入 **static**

(b) **thymeleaf 布局**

- 定义 fragment
- 使用 fragment 布局
  加入 th:href=

```
<!DOCTYPE html>
<html lang="en"
    xmlns:th="http://www.w3.org/1999/xhtml">
<head th:fragment="head(title)">
<meta charset="UTF-8">
<meta name="viewport"
    content="width=device-width,
    initial-scale=1.0">
<title th:replace="${title}">博客详情</title>
<link rel="stylesheet"
    href="https://cdn.jsdelivr.net/semantic-ui/
2.2.4/semantic.min.css">
<link rel="stylesheet"
    href="../static/css/typo.css"
    th:href="@{/css/typo.css}">
<link rel="stylesheet"
    href="../static/css/animate.css"
    th:href="@{/css/animate.css}">
<link rel="stylesheet"
    href="../static/lib/prism/prism.css"
    th:href="@{/lib/prism/prism.css}">
```

```
                <link rel="stylesheet"
                    href="../static/lib/tocbot/tocbot.css"
                    th:href="@{/lib/tocbot/tocbot.css}">
                <link rel="stylesheet"
                    href="../static/css/me.css"
                    th:href="@{/css/me.css}">
                </head>
                <body>
```

(c) 错误页面美化

图 5.7 error 页面美化

5.4 设计与规范

5.4.1 实体设计

实体关系:

- 评论类自关联关系
- Blog 类
- Type 类
- Tag 类
- Comment 类
- User 类

评论类自关联关系:

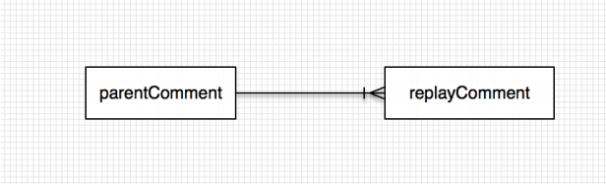

图 5.8 评论类自关联关系

Blog 类:

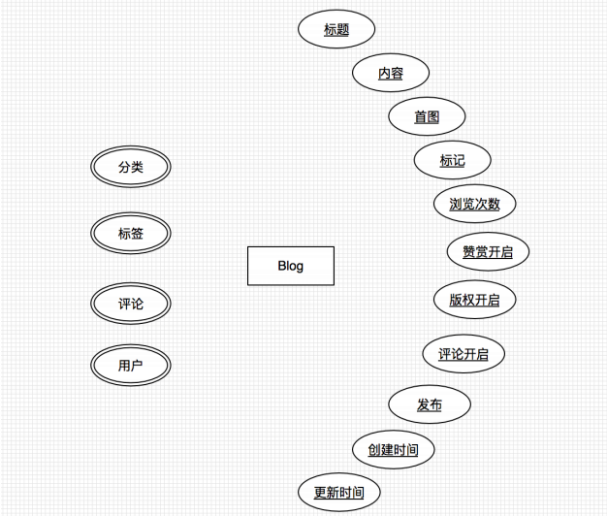

图 5.9 Blog 类

Type 类:

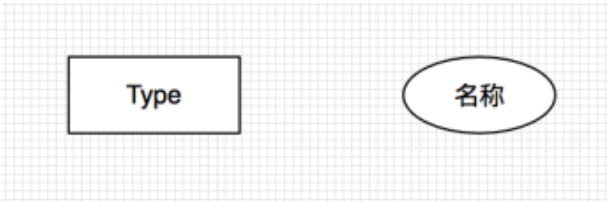

图 5.10 Type 类

Tag 类:

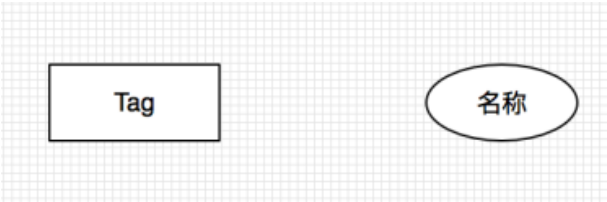

图 5.11 Tag 类

Comment 类:

图 5.12 Comment 类

User 类:

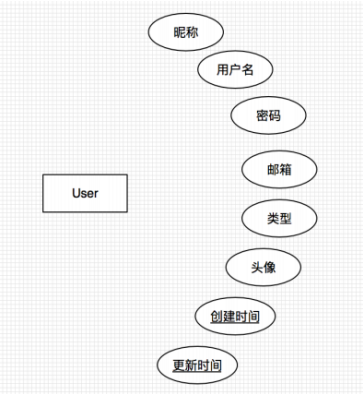

图 5.13 User 类

5.5 后台管理功能实现

5.5.1 登录

1. 构建登录页面和后台管理首页

   利用 semantic UI 构建登录页

```
6    <title>博客管理登录</title>
7    <link rel="stylesheet" href="https://cdn.jsdelivr.net/semantic-ui/2.2.4/semantic.min.css">
8    <link rel="stylesheet" href="../../static/css/me.css">
```

图 5.14 登录

```
        <div class="ui segment">
          <div class="field">
```

```
            <div class="ui left icon input">
            <i class="user icon"></i>
            <input type="text" name="username" placeholder="用户名">
            </div>
            </div>
            <div class="field">
            <div class="ui left icon input">
            <i class="lock icon"></i>
            <input type="password" name="password" placeholder="密码">
            </div>
            </div>
            <button class="ui fluid large teal submit button">登 录</button>
          </div>
```

2. UserService 和 UserRepository

   UserServiceImpl.java

```
            @Service
            public class UserServiceImpl implements UserService {

               @Autowired
               private UserRepository userRepository;

               @Override
               public User checkUser(String username, String password) {
               User user =
                   userRepository.findByUsernameAndPassword(username,
                   MD5Utils.code(password));
               return user;
               }
            }
```

3. LoginController 实现登录

   LoginController.java

```
            public class LoginController {
               @Autowired
               private UserService userService;
               @GetMapping
               public String loginPage() {
                  return "admin/login";
```

```
    }
    @PostMapping("/login")
    public String login(@RequestParam String username,
    @RequestParam String password,
    HttpSession session,
    RedirectAttributes attributes) {
        User user = userService.checkUser(username, password);
        if (user != null) {
            user.setPassword(null);
            session.setAttribute("user",user);
            return "admin/index";
        } else {
            attributes.addFlashAttribute("message",
                "用户名和密码错误");
            return "redirect:/admin";
        }
    }
    @GetMapping("/logout")
    public String logout(HttpSession session) {
        session.removeAttribute("user");
        return "redirect:/admin";
    }
}
```

4. MD5 加密

MD5Utils.java

```
public static String code(String str){
    try {
    MessageDigest md = MessageDigest.getInstance("MD5");
    md.update(str.getBytes());
    byte[]byteDigest = md.digest();
    int i;
    StringBuffer buf = new StringBuffer("");
    for (int offset = 0; offset < byteDigest.length; offset++) {
    i = byteDigest[offset];
    if (i < 0)
    i += 256;
    if (i < 16)
    buf.append("0");
    buf.append(Integer.toHexString(i));
```

```
        }
        //32位加密
        return buf.toString();
        // 16位的加密
        //return buf.toString().substring(8, 24);
        } catch (NoSuchAlgorithmException e) {
        e.printStackTrace();
        return null;
        }
    }
```

5. 登录拦截器

LoginInterceptor.java

```
    public class LoginInterceptor extends HandlerInterceptorAdapter {
        @Override
        public boolean preHandle(HttpServletRequest request,
        HttpServletResponse response,
        Object handler) throws Exception {
            if (request.getSession().getAttribute("user") == null) {
                response.sendRedirect("/admin");
                return false;
            }
            return true;
        }
    }
```

5.5.2 分类管理

1. 分类管理页面

TypeService.java

```
    public interface TypeService {

        Type saveType(Type type);

        Type getType(Long id);

        Type getTypeByName(String name);
```

```
    Page<Type> listType(Pageable pageable);

    List<Type> listType();

    List<Type> listTypeTop(Integer size);

    Type updateType(Long id,Type type);

    void deleteType(Long id);
}
```

2. 分类列表分页

```
{
  "content":[
  {"id":123,"title":"blog122","content":"this is blog
      content"},
  {"id":122,"title":"blog121","content":"this is blog
      content"},
  {"id":121,"title":"blog120","content":"this is blog
      content"},
  {"id":120,"title":"blog119","content":"this is blog
      content"},
  {"id":119,"title":"blog118","content":"this is blog
      content"},
  {"id":118,"title":"blog117","content":"this is blog
      content"},
  {"id":117,"title":"blog116","content":"this is blog
      content"},
  {"id":116,"title":"blog115","content":"this is blog
      content"},
  {"id":115,"title":"blog114","content":"this is blog
      content"},
  {"id":114,"title":"blog113","content":"this is blog
      content"},
  {"id":113,"title":"blog112","content":"this is blog
      content"},
  {"id":112,"title":"blog111","content":"this is blog
      content"},
  {"id":111,"title":"blog110","content":"this is blog
      content"},
```

```
            {"id":110,"title":"blog109","content":"this is blog
                content"},
            {"id":109,"title":"blog108","content":"this is blog
                content"}],
            "last":false,
            "totalPages":9,
            "totalElements":123,
            "size":15,
            "number":0,
            "first":true,
            "sort":[{
            "direction":"DESC",
            "property":"id",
            "ignoreCase":false,
            "nullHandling":"NATIVE",
            "ascending":false
            }],
            "numberOfElements":15
        }
```


3. 分类新增、修改、删除

```
<div class="ui attached pointing menu">
  <div class="ui container">
    <div class="right menu">
      <a href="#" th:href="@{/admin/types/input}" class="active item">新增</a>
      <a href="#" th:href="@{/admin/types}" class="teal  item">列表</a>
    </div>
  </div>
</div>
```

图 **5.15** 分类管理

5.5.3 标签管理

与分类管理类似，详情请看源码

5.5.4 博客管理

```java
public interface BlogService {

    Blog getBlog(Long id);

    Blog getAndConvert(Long id);

    Page<Blog> listBlog(Pageable pageable,BlogQuery blog);

    Page<Blog> listBlog(Pageable pageable);

    Page<Blog> listBlog(Long tagId,Pageable pageable);

    Page<Blog> listBlog(String query,Pageable pageable);

    List<Blog> listRecommendBlogTop(Integer size);

    Map<String,List<Blog>> archiveBlog();

    Long countBlog();

    Blog saveBlog(Blog blog);

    Blog updateBlog(Long id,Blog blog);

    void deleteBlog(Long id);
}
```

图 5.16 博客管理

1. 博客分页查询
2. 博客新增

```java
public Blog updateBlog(Long id, Blog blog) {
    Blog b = blogRepository.findOne(id);
    if (b == null) {
        throw new NotFoundException("该博客不存在");
    }
    BeanUtils.copyProperties(blog,b, MyBeanUtils.getNullPropertyNames(blog));
    b.setUpdateTime(new Date());
    return blogRepository.save(b);
}
```

图 5.17 博客新增

3. 博客修改
4. 博客删除

```java
public void deleteBlog(Long id) { blogRepository.delete(id); }
}
```

图 5.18 博客删除

5.6 前端展示功能实现

5.6.1 首页展示

MarkDown 转换 HTML

1. commonmark-java https://github.com/atlassian/commonmark-java
2. pom.xml 引用 commonmark 和扩展插件

```
<dependency>
<groupId>com.atlassian.commonmark</groupId>
<artifactId>commonmark</artifactId>
<version>0.10.0</version>
</dependency>
<dependency>
<groupId>com.atlassian.commonmark</groupId>
<artifactId>commonmark-ext-heading-anchor</artifactId>
<version>0.10.0</version>
</dependency>
<dependency>
<groupId>com.atlassian.commonmark</groupId>
<artifactId>commonmark-ext-gfm-tables</artifactId>
<version>0.10.0</version>
</dependency>
```

评论功能：

- 评论信息提交与回复功能

  部分评论与回复功能的 js 代码：

```
$('#commentpost-btn').click(function () {
   var boo = $('.ui.form').form('validate form');
   if (boo) {
   console.log('校验成功');
   postData();
   } else {
   console.log('校验失败');
   }

});

function reply(obj) {
   var commentId = $(obj).data('commentid');
   var commentNickname = $(obj).data('commentnickname');
   $("[name='content']").attr("placeholder",
       "@"+commentNickname).focus();
   $("[name='parentComment.id']").val(commentId);
```

```
        $(window).scrollTo($('#comment-form'),500);
    }
```

- 评论信息列表展示功能

```
public class CommentServiceImpl implements CommentService {

    @Autowired
    private CommentRepository commentRepository;

    @Override
    public List<Comment> listCommentByBlogId(Long blogId) {
    Sort sort = new Sort("createTime");
    List<Comment> comments =
        commentRepository.findByBlogIdAndParentCommentNull
    (blogId,sort);
    return eachComment(comments);
    ...
}
```

5.6.2 分类页

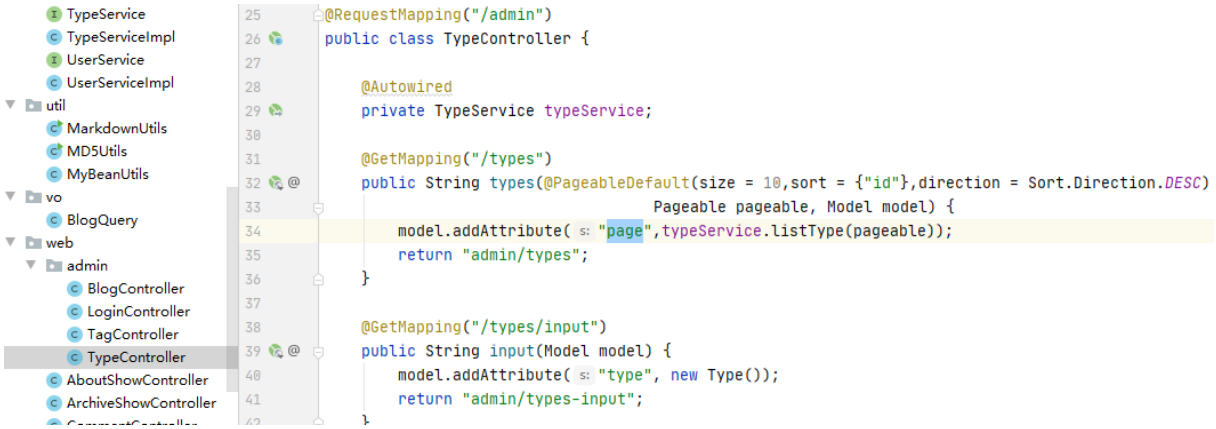

图 5.19 分类页动态化的关键部分代码 1

```
@PostMapping("/types")
public String post(@Valid Type type,BindingResult result, RedirectAttributes attributes) {
    Type type1 = typeService.getTypeByName(type.getName());
    if (type1 != null) {
        result.rejectValue( s: "name", s1: "nameError", s2: "不能添加重复的分类");
    }
    if (result.hasErrors()) {
        return "admin/types-input";
    }
    Type t = typeService.saveType(type);
    if (t == null ) {
        attributes.addFlashAttribute( s: "message",  o: "新增失败");
    } else {
        attributes.addFlashAttribute( s: "message",  o: "新增成功");
    }
    return "redirect:/admin/types";
}
```

图 5.20 分类页动态化的关键部分代码 2

5.6.3 标签页

部分关键代码：TagShowController.java

```
@Controller
public class TagShowController {
   @Autowired
   private TagService tagService;
   @Autowired
   private BlogService blogService;
   @GetMapping("/tags/{id}")
   public String tags(@PageableDefault(size = 8, sort = {"updateTime"},
       direction = Sort.Direction.DESC) Pageable pageable,
   @PathVariable Long id, Model model) {
   List<Tag> tags = tagService.listTagTop(10000);
   if (id == -1) {
   id = tags.get(0).getId();
   }
   model.addAttribute("tags", tags);
   model.addAttribute("page", blogService.listBlog(id,pageable));
   model.addAttribute("activeTagId", id);
   return "tags";
   }
}
```

5.6.4 归档页

部分关键代码：ArchiveShowController.java

```
@Controller
public class ArchiveShowController {

   @Autowired
   private BlogService blogService;

   @GetMapping("/archives")
   public String archives(Model model) {
   model.addAttribute("archiveMap", blogService.archiveBlog());
   model.addAttribute("blogCount", blogService.countBlog());
   return "archives";
   }
}
```

5.6.5 AboutMe

部分关键代码：ArchiveShowController.java

```
@Controller
public class AboutShowController {
   @GetMapping("/about")
   public String about() {
   return "about";
   }
}
```

六、 系统测试

各个页面的功能模块测试：

6.1 首页

图 6.1 首页部分 1 部分展示

图 6.2 页面底部 footer 测试

6.2 AboutMe

图 6.3 关于我页面测试插件 1: 鼠标 hover

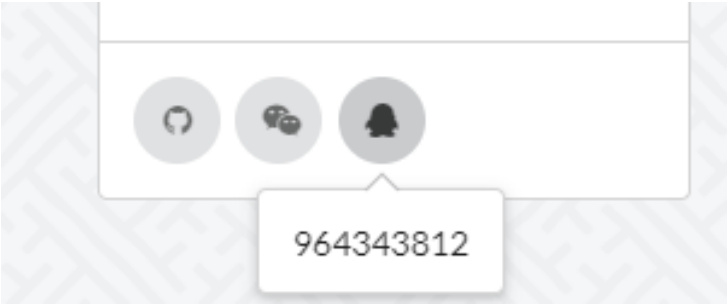

图 6.4 关于我页面测试插件 2: 鼠标 hover

6.3 博客详情页

图 6.5 MarkDown 编辑器测试

图 6.6 赞赏功能: 鼠标 hover 提供付款二维码

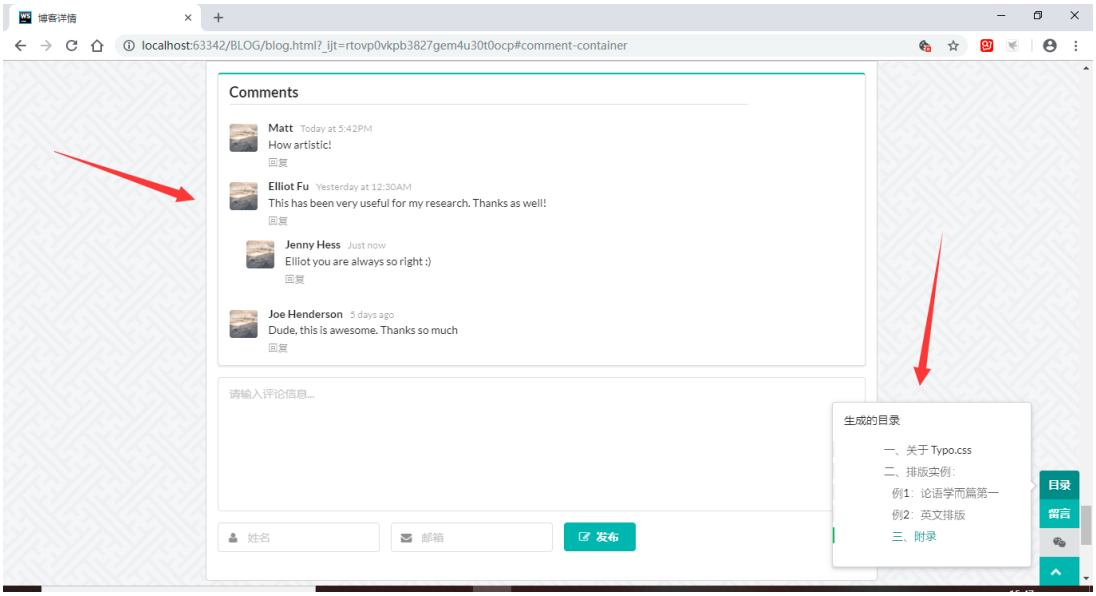

图 6.7 评论功能以及生成目录模块测试

6.4 博客管理

图 6.8 管理测试

6.5 博客发表

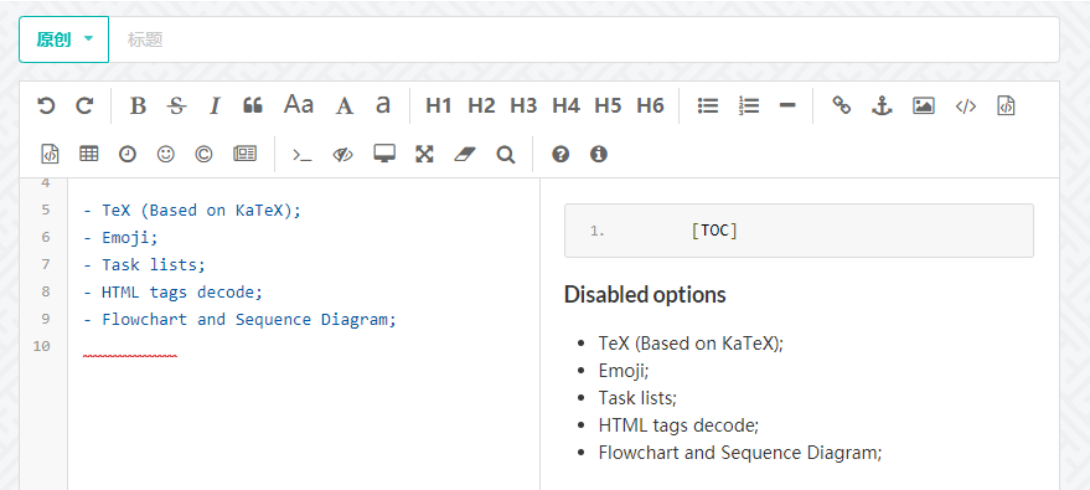

图 6.9 博客发表:MarkDown 编辑

图 6.10 博客发表: 保存、返回等

七、 结论

7.1 我学习到了什么

7.1.1 SpringBoot 的初步试用

Spring Boot 是由 Pivotal 团队提供的全新框架，其设计目的是用来简化新 Spring 应用的初始搭建以及开发过程。该框架使用了特定的方式来进行配置，从而使开发人员不再需要定义样板化的配置。用我的话来理解，就是 Spring Boot 其实不是什么新的框架，它默认配置了很多框架的使用方式，就像 Maven 整合了所有的 Jar 包，Spring Boot 整合了所有的框架。只需要非常少的几个配置就可以迅速方便的搭建起来一套 Web 项目或者是构建一个微服务。使用 Spring Boot 可以非常方便、快速搭建项目，使我们不

用关心框架之间的兼容性，适用版本等各种问题，我们想使用任何东西，仅仅添加一个配置就可以，所以使用 Spring Boot 非常适合构建微服务。

7.1.2 JavaScript 的初步认知以及简单使用

JavaScript（简称“JS”）是一种具有函数优先的轻量级，解释型或即时编译型的编程语言。虽然它是作为开发 Web 页面的脚本语言而出名的，但是它也被用到了很多非浏览器环境中，JavaScript 基于原型编程、多范式的动态脚本语言，并且支持面向对象、命令式和声明式（如函数式编程）风格。

7.1.3 HTML 和 CSS

html 是 HyperTextMark-upLanguage 的缩写，即超文本标记语言；html 是用来定义文档内容结构的，包含了用户需要浏览的内容，包括图文、视频，即构成网页的基本元素；html 是网页的结构（Structure），需要有多种框架和布局，比如 frameset 框架集、iframe 内联框架、div+css 布局、table 布局等，同时支持表单提交（HTML Form），包括基础表单、input 输入框、输入框类型、文本域、列表、label 等。html 的结构包括头部（Head）、主体（Body）两大部分，其中头部描述浏览器所需的信息，而主体则包含所要说明的具体内容。

css 是 Cascading Style Sheets 的缩写, 即层叠式样式表单，它是由 W3C 协会制定并发布的一个网页排版式标准, 是对 HTML 语言功能的补充。css 用于定义 html 文档的样式，即外观，比如网页上的动态文字、文字的色彩、字体、动画效果，都可以由 css 来实现。css 的主要的用途是对网页中字体、颜色、背景、图像及其他各种元素的控制, 使网页能够完全按照设计者的要求来显示。

简单来说，HTML 构建了网页的骨架，CSS 为丰富了骨架的内容和样式，js 提供网页各个组件之间的联系功能。

7.1.4 了解了前端与后端的区别

- 展示方式
  - 前端指的是用户可见的界面，网站前端页面也就是网页的页面开发，比如网页上的特效、布局、图片、视频，音频等内容。前端的工作内容就是将美工设计的效果图的设计成浏览器可以运行的网页，并配合后端做网页的数据显示和交互等可视方面的工作内容。
  - 后端是指用户看不见的东西，通常是与前端工程师进行数据交互及网站数据的保存和读取，相对来说后端涉及到的逻辑代码比前端要多的多，后端考虑的是底层业务逻辑的实现，平台的稳定性与性能等。

- 所用技术
  - 前端开发用到的技术包括但不限于 html5、css3、javascript、jquery、Bootstrap、Node.js 、Webpack，AngularJs，ReactJs，VueJs 等技术。
  - 后端开发以 java 为例主要用到的是包括但不限于 Struts spring springmvc Hibernate Http 协议 Servlet Tomcat 服务器等技术。
- 工作职责
  - 前端工程师主要的工作职责分为三大部分，分别是传统的 Web 前端开发，移动端开发和大数据呈现端开发。Web 前端开发主要针对的是 PC 端开发任务；移动端开发则包括 Android 开发、iOS 开发和各种小程序开发，在移动互联网迅速发展的带动下，移动端的开发任务量是比较大的，随着 5G 标准的落地，未来移动端的开发任务将得到进一步的拓展；大数据呈现则主要是基于已有的平台完成最终分析结果的呈现，呈现方式通常也有多种选择，比如大屏展示等。
  - 后端工程师的主要职责也集中在三大部分，分别是平台设计、接口设计和功能实现。平台设计主要是搭建后端的支撑服务容器；接口设计主要针对于不同行业进行相应的功能接口设计，通常一个平台有多套接口，就像卫星导航平台设有民用和军用两套接口一样；功能实现则是完成具体的业务逻辑实现。

7.2 此次个人实践中发现的问题和一些感触

1. 前端的开发相对于后端来说是比较容易的，至少从我这个初学者来说，因为可以实时的看到代码呈现的效果，有种“造物主”的感受。但是在开发过程中，随着代码量的加大，一旦落掉某个错误，排查起来会变得越来越困难，往往一个落掉的引号”, 就会使代码出错，页面样式得不到变化，排查起来十分困难。因此，这次第一次的 web 实践，使我了解到了代码分块、及时“复盘”差错、及时编译查找错误是开发过程中必不可少的步骤。

2. 后端的开发，我个人对于这次的实践的感受是比较“不友好的”，原因是没有扎实的 SpringBoot 基础，开发起来较为费力，查阅了大量资料和视频资源，尽管大部分模块功能得以实现，但仍有一些部分没有完成，这些部分在较短时间内可能暂时无法完成。

3. 博客现在仍处于测试阶段，一些后续工作 (如域名等) 等待完成，暂时无法上线。

7.3 写在最后

纸上得来终觉浅，绝知此事要躬行。只有亲自动手实践才能让学习变得丰富，让基础变得更扎实，在不断的实践中增进理论知识的吸取，同时也在不断的对理论知识进行实践。对于学习者来说，结果如何也许不是最终的目的，实践的过程是最为重要的。

最后,此次项目的源码已上传至个人的github: **https://github.com/YXH6620/MyBlog** 上，内容包括项目源码，报告书的 LaTeX 源码，书写素材等。

参考文献