个人博客管理系统的设计与实现

摘 要

Web 2.0出现以来，社交媒体上的内容和使用用户介入度急剧增长。博客已经成为与商业和个人生活的各个方面相关的记录平台。然而，没有适当的工具来正确地汇总和保存博客内容，以及有效地管理博客归档。鉴于博客日益重要，建立一个稳定运行的系统以促进博客保护至关重要，保护遗产的重要部分，这对当代和后代来说将是有价值的。在本文中，本作者将详细介绍在这个课题研究中所构建的一个具有体验流畅且稳定运行的博客展示和托管的平台，任何个人或组织都可以使用这个平台来保留他的博客。

系统功能主要包括博客内容的查看、编辑、删除、投稿、保存上传；博客用户的个人资料编辑修改和保存；管理员对投稿日志的筛选，以及将日志推送到首页。由于本系统的定位是一个博客社区，不仅仅是只有博客的基础管理，每个用户之间还有更多的互动：用户之间可以进行关注；用户可以对某篇文章进行评论、点赞等操作；当用户被关注或收到投稿相关的反馈时，该条内容会添加到通知模块，用户点击通知按钮时会弹出通知列表，点击某条通知即已读通知并从数据库里删除。

对于本博客管理系统，进行了系统化的测试。根据需求文档对比当前系统是否符合规定的要求；确定该系统是否处于高风险的范围，是否安全；用户是否需要花费成本；对每个模块进行单元测试；评估维护的成本等。本文针对个人博客管理系统进行了系统架构分析和实验研究，针对特点建立了个人博客管理的单页应用。本系统前后端分离，使系统耦合度更低，每个模块之间的关联更小，开发效率更高，维护的成本也大大降低。

**关键词：**SPA 博客社区 MVC

Design and Implementation of Personal Blog Management System

**ABSTRACT**

Since the advent of Web 2.0, content on social media and the use of user involvement have grown dramatically. The blog has become a recording platform associated with all aspects of business and personal life. However, we do not have the appropriate tools to properly summarize and save blog content, as well as effectively manage blog archives. Given the growing importance of blogging, it is important to establish a stable system to promote blog protection that is vital to protecting our heritage, which will be valuable to both contemporary and future generations. In this article, I will detail the platform in this research to build a smooth and stable blog display and hosting platform, any individual or organization can use this platform to retain his blog.

System functions include the contents of the blog to view, edit, delete, submit, save upload; blog user's personal data editor to modify and save; administrator on the submission of log selection, and push the log to the home page. As the positioning of the system is a blog community, not only the basic management of the blog, there is more interaction between each user: the user can be concerned about; users can comment on an article, Operation; when the user is concerned or receive feedback related to the submission, the article will be added to the notification module, the user clicks the notification button will pop up a notification list, click on a notice that is read and deleted from the database.

For this blog management system, a systematic test. According to the requirements of the document compared to the current system is consistent with the requirements; to determine whether the system is in a high risk range, whether the security; users need to spend costs; unit testing for each module.

**Key Words:** SPA Blog community MVC

目 录

[引 言 1](#_Toc16550)

[第一章 任务概述 2](#_Toc9125)

[1.1目标 2](#_Toc9266)

[1.2本系统的特点和目标用户 2](#_Toc3230)

[第二章 需求分析 4](#_Toc8466)

[2.1基本需求 4](#_Toc5820)

[2.1.1功能需求 4](#_Toc2081)

[2.1.2性能需求 4](#_Toc14966)

[2.2运行环境需求 5](#_Toc22233)

[2.2.1软件环境 5](#_Toc21179)

[2.2.2硬件环境 5](#_Toc27232)

[2.3 E-R图 5](#_Toc17989)

[2.3.1用户实体 5](#_Toc15370)

[2.3.2 通知实体 6](#_Toc10381)

[2.3.3文章实体 6](#_Toc31486)

[2.3.4评论实体 6](#_Toc30571)

[第三章 总体设计 8](#_Toc13273)

[3.1各模块流程图、流程说明及数据流图 8](#_Toc674)

[3.1.1.登录注册 8](#_Toc32384)

[3.1.2查看用户信息 8](#_Toc13547)

[3.1.3日志查看/编辑/删除 9](#_Toc11145)

[3.1.4通知模块 10](#_Toc29247)

[3.1.5首页投稿 10](#_Toc30176)

[3.1.6 关注作者动态模块 11](#_Toc7524)

[3.1.7评论功能 12](#_Toc8847)

[3.1.8 点赞功能 13](#_Toc10008)

[3.2 设计模式 14](#_Toc2005)

[3.2.1 MVC设计模式 14](#_Toc23364)

[3.2.2 MVVM设计模式 14](#_Toc32244)

[3.2.3系统交互模式设计 15](#_Toc1312)

[3.3技术选型 17](#_Toc3925)

[3.3.1 Vue 17](#_Toc2360)

[3.3.2 Webpack 17](#_Toc11003)

[3.3.3 Ajax 18](#_Toc19616)

[3.3.4 Apache+php+mysql 18](#_Toc29528)

[3.4 开发工具 20](#_Toc15079)

[3.4.1 WebStorm 20](#_Toc21764)

[3.4.2 PhpStorm 20](#_Toc2182)

[3.4.3 Sublime text3 20](#_Toc11232)

[第四章 详细设计 21](#_Toc32223)

[4.1 数据库设计 21](#_Toc12976)

[4.1.1 通知表（notification） 21](#_Toc14080)

[4.1.2 评论表（comment） 21](#_Toc6198)

[4.1.3 文章表（article） 21](#_Toc26319)

[4.1.4 关注表（attention） 22](#_Toc11846)

[4.1.5 用户信息表（userInfo） 22](#_Toc24650)

[4.1.6 用户登录表（userlog） 23](#_Toc2678)

[4.2 接口设计 23](#_Toc32504)

[4.2.1 登录页 23](#_Toc27989)

[4.2.2 注册页 24](#_Toc4037)

[4.2.3 文章编辑页 24](#_Toc11341)

[4.2.4 文章列表页 25](#_Toc9694)

[4.2.5 首页 26](#_Toc31643)

[4.2.6 关注页 28](#_Toc25416)

[4.2.7 文章详情页 29](#_Toc8450)

[第五章 系统实现 31](#_Toc15171)

[5.1 开发步骤 31](#_Toc7462)

[5.2 主要配置文件 31](#_Toc9402)

[5.2.1 package.json文件 31](#_Toc2227)

[5.2.2 vue-router配置文件 31](#_Toc9557)

[5.2.3 webpack配置文件 32](#_Toc3001)

[5.3 核心问题及解决方案 32](#_Toc8673)

[5.3.1用户的个性域名 32](#_Toc18658)

[5.3.2 编辑文章时防止意外退出，用local storage保存 33](#_Toc1831)

[5.3.3 每次切换模块时，加载资源时间过长的问题 34](#_Toc9073)

[5.3.4 markdown转码的问题 34](#_Toc13941)

[5.4程序的实现 34](#_Toc15920)

[5.4.1 开发流程 34](#_Toc15198)

[5.4.2部署流程 35](#_Toc17064)

[5.4.3 设计结果展示 35](#_Toc15481)

[参考文献 42](#_Toc15207)

[谢 辞 43](#_Toc26413)

# 

# 引 言

本课题利用这个的专门平台博客的特点，能够改进归档。 Web保存被定义为捕获，管理和保存网站和网页资源。 Web保存必须是一个从出生（即开始）到死亡（完成）的活动，它应该包括网络资源的一个完整生命周期。最著名的网络归档计划是Internet Archive自1996年以来一直在运作。另外还有各种各样的项目国家和国际组织正在开展网络维护工作相关活动。所有活跃的国家网络存档工作，作为以及一些学术网络档案是国际互联网的成员保存的。 但是，这是一项复杂的任务需要耗费很多资源。 因此，网络归档只有部分现有的网络被归档。与传统的媒体相反，如印刷业，使用网络可以极大地加快传播速度。 因此，博客的种类不仅仅是由规定主题的文档，还有其他参数，如每页的归档频率以及与页面请求相关的参数（例如浏览器，用户帐号，语言等）[1]。

博客归档是Web归档的子类别。博客的意义在商业和私人生活的各个方面都不能低估。 博客是拉丁美洲的教学物理学用来促进时尚讨论法国的年轻人的产物。所有博客的集体成果，它们的内容，相互联系和影响构成了他们的作品博客圈，具有重要意义的独特的社会技术文物。然而，当前的网络存档工具导致博客保护的几个问题。首先，采集和策划的工具使用基于时间表的方法确定内容应被捕获以便归档的时间点，导致信息丢失，如果它比其更新更频繁地更新被爬。另一方面，不必要的收获和存档的重复如果博客比爬网时间表更不频繁地更新，则会发生如果整个博客再次被收获，而不是选择性地收获新的页面。因此，考虑更新事件（例如新帖子，新评论等）作为爬行活动的触发器的方法将更为合适。

# 第一章 任务概述

## 1.1目标

当今快节奏的社会中，人们往往忽略了文字的沉淀。该系统旨在搭建一个博客社区，给用户提供一个更多元化的文字交流平台，不仅仅是发表文章，用户之间还可以进行更加现代化的互动。

本系统是用vue+php实现的一个完全基于浏览器的博客系统，注册用户拥有以下权限：

1. 登陆系统
2. 注册用户可以给某篇博客留言
3. 注册用户可以对某篇文章进行点赞操作
4. 注册用户可以拥有自己的头像昵称等信息
5. 注册用户可以编写自己的文章并保存到服务器上
6. 注册用户可以把自己的文章投稿到首页，让更多的用户查看或评论
7. 注册用户可以管理自己的文章列表，上传文章、删除指定文章、和编辑指定文章等操作
8. 注册用户可以管理自己的文章分类列表，编辑分类条目、添加一个分类、或删除某个分类等操作

管理员拥有以下权限：

管理员可以对用户投稿的文章进行筛选，并放到首页上。

非注册用户可以进行的操作：

1. 查看主页上的投稿文章
2. 注册账户

## 1.2本系统的特点和目标用户

该系统面向喜欢文字交流的年轻社群，尤其是能快速接受新鲜事物并且能够持续保持活跃状态的用户。

该系统特点：

1. 优雅。本系统使用基于markdown的编辑器。Markdown是一种具有纯文本格式化语法的轻量级标记语言。它的设计使其可以使用相同名称的工具转换为HTML和许多其他格式。Markdown通常用于设置自述文件的格式，用于在线论坛中写入消息，并使用纯文本编辑器创建丰富的文本。对比富文本编辑器：对于书写者来说，排版更加优雅，写作时更加专注于文字本身，而无需再考虑怎样排版更加好看；对于读者来说，使用统一的格式，阅读体验更佳，对于开发者来说，更加易于开发和维护。
2. 免费。用户不需要购买域名，不需要购买服务器，不需要花费任何的费用，只要注册一个用户名，就能拥有与自己用户名一样的基于本站的域名了。
3. 传播更广泛。用户可以通过投稿的方式，把自己的觉得满意的作品推送到首页，让更多的用户看到，传播更加广泛。而不仅仅像普通的个人主页那样：必须得进入目标用户的主页才能浏览信息，使信息流通变得更加闭塞。
4. 使用方便。使用该系统无需掌握任何的计算机知识，只要初步了解了markdown的基础语法，就可以写博客了。像使用电子邮箱一样简单可以对自己的博客进行管理：如：编辑、上传、删除等操作。

# 第二章 需求分析

## 2.1基本需求

### 2.1.1功能需求

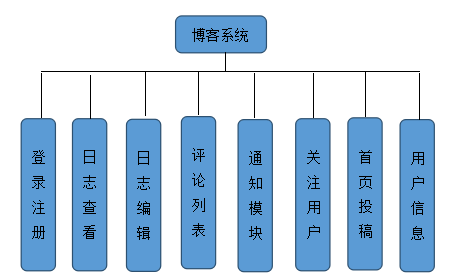


图2.1 功能模块图

本系统的用户有三种身份，分别为：游客（即未注册/未登录用户），注册用户和管理员。如图2.1所示，要求每种角色使用系统，操作时都方便快捷。游客在没有登录的情况下可以查看博客的详情；已经注册并且登陆系统的用户享有更多的特权，可以对自己的博客进行相关的操作等；管理员可以对用户投稿文章进行筛选。

### 2.1.2性能需求

1. 响应时间:

规定服务器响应时间不超过0.5s，所以在初次加载时不应该出现白屏现象，给用户一个更好的体验。每次客户端向服务端请求时应在异步状态下进行操作。

2. 系统输入输出精度需求

规定用户输入空值时给出警告提示。

每次请求后的响应字段必须为全部可用字段，需要什么就响应什么，不能冗余，造成输出变大，响应时间变长这一问题。

## 2.2运行环境需求

### 2.2.1软件环境

操作系统：windows、linux、macOS

服务器：apache 、node v4.4.4（开发环境）

数据库环境：mysql（innoDB）

浏览器：chrome、firefox、safari

### 2.2.2硬件环境

处理器：AMD A6-4400M APU with Radeon HD Graphics

内存：6.00GB

系统类型：64位操作系统

## 2.3 E-R图

### 2.3.1用户实体

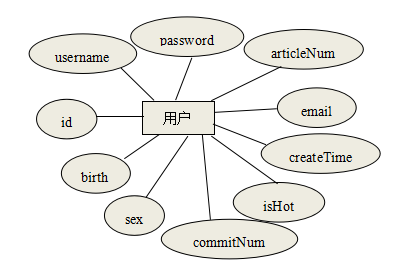


图2.2 用户实体图

如图2.2，每个用户实体包括用户id（id），用户名（username），登录密码（password），昵称（nickName）、性别（sex）、生日（birth）、常住地（place）、常用邮箱（email）、主页推荐标识符（isHot）、获赞数（likeNum）、获得评论数（commitNum）等属性。

### 2.3.2 通知实体

如图2.3，每个通知实体包括通知id、通知标题（title）、通知类型（即type，包含成功、失败和警告）、创建日期（createTime）等属性。

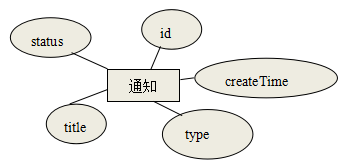


图2.3 通知实体图

### 2.3.3文章实体

如图2.4，每个文章实体包括文章id、文章标题（title）、内容（content）、创建时间（createTime）、作者名（writerId）、获赞数（likeNum）、摘要（shortCut）等属性。

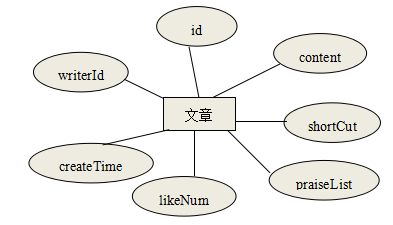


图2.4 文章实体图

### 2.3.4评论实体

如图2.5，评论实体包括评论的id、对应的文章id（articleId）、评论人的用户id（userId）、评论内容（content）、创建时间（createTime）等属性。

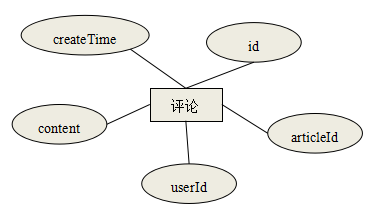


图2.5 评论实体图

# 第三章 总体设计

## 3.1各模块流程图、流程说明及数据流图

### 3.1.1.登录注册

如图3.1，从首页点击登录进入登录页，填写用户名和登录密码进行登录；未注册则填写相关字段进行注册。在用户输入的时候，验证空值和非法字段，减少请求次数。

操作成功后跳转首页，进行其他操作；失败则继续当前操作直至成功或仅浏览首页的文章或推荐关注的作者。

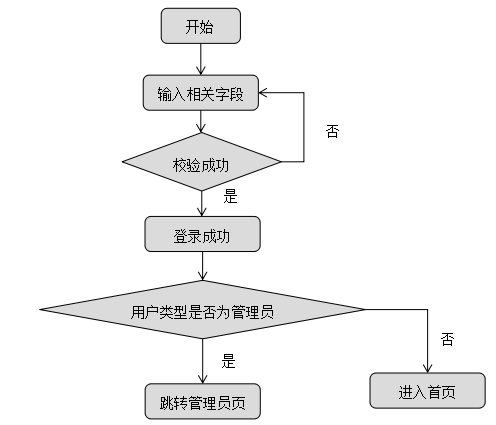


图3.1 登录流程图

### 3.1.2查看用户信息

如图3.2，点击用户头像，带着当前用户的id参数，跳转到个人信息页；判断所带参数是否为当前用户，是则编辑个人的资料；否则浏览信息。

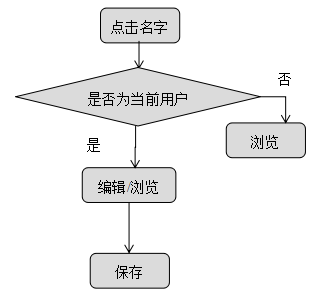


图3.2 用户信息编辑查看流程图

### 3.1.3日志查看/编辑/删除

如图3.3，点击日志，带着用户id这个参数跳转到详情页；进入这个详情页后，判断路由中的指定参数是否为当前的日志作者：是则可以进行修改，删除，评论等操作；否（即不是当前日志的作者）则只能评论点赞。

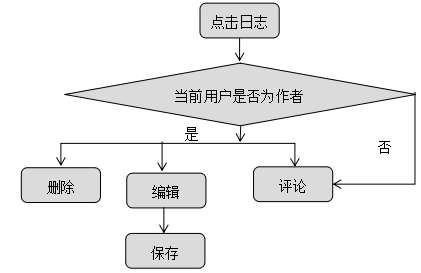


图3.3 日志查看/编辑流程图

### 3.1.4通知模块

如图3.4，用户被关注或日志被点赞和评论时，插入通知队列中，当用户从主页点击“查看通知”这个图标时，展示该用户当前的未读通知，点击指定通知，则关闭该通知并从数据库里删除。

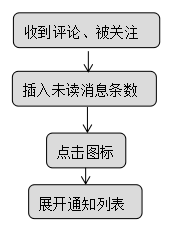
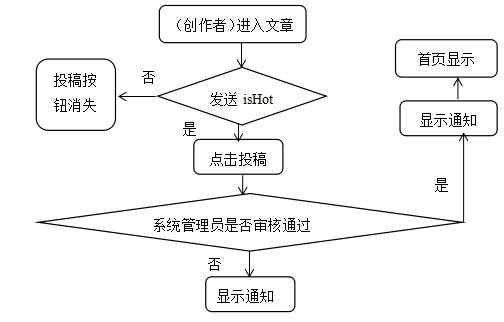


图3.4 通知模块流程图

### 3.1.5首页投稿



发送isHot

图3.5 首页投稿流程图

如图3.5和3.6，当用户进入详情页时，立即判断当前用户的id是否与文章作者的id一致：是则通过服务器发来的isHot字段判断投稿状态，进行投稿操作；否则不显示投稿按钮，用户无法进行投稿操作。管理员把投稿的文章进行筛选，将投稿的文章显示在首页。

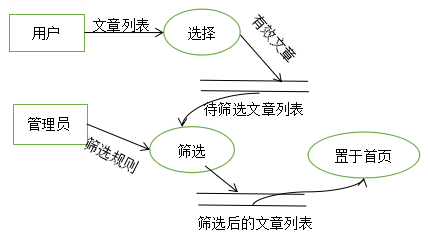


图3.6 首页投稿数据流图

### 3.1.6 关注作者动态模块

1. 关注作者

如图3.7，首页有系统推荐的作者，用户可以点击头像查看他的信息，对其进行关注。已关注的作者则报提示信息，确认是否取消关注。之后，他的动态（即发布的文章）将会被输出在关注页。

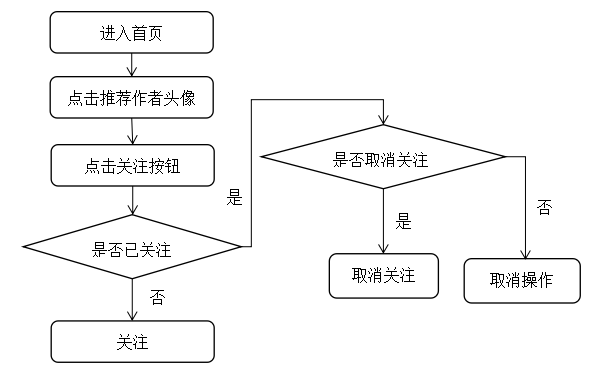


图3.7 关注作者流程图

1. 查看所关注作者的时间线

如图3.8，在关注页显示用户关注的作者列表，用户可以根据时间最新排序或者热度排序（被赞数）查看该作者的文章列表，点击则进入某篇文章的详情模块，查看文章具体信息，进行查看、评论、点赞等。

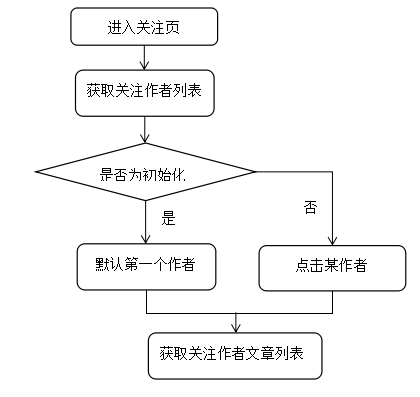


图3.8 关注列表文章流程图

### 3.1.7评论功能

如图3.9，用户进入详情页就可以对文章进行评论了，评论提交成功后，新增评论直接显示在评论列表头部，输出评论人、评论内容和评论时间。每次评论都会向文章作者的通知模块推送一条信息。

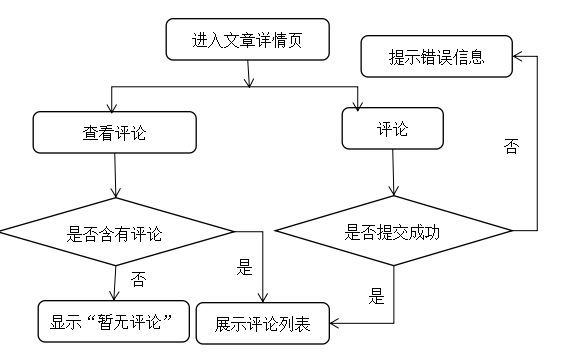


图3.9 评论功能流程图

### 3.1.8 点赞功能

用户进入详情页就可以对文章点赞，每个用户只能点赞一次，点过赞的高亮点赞图标，不能再点。每次进行点赞操作都会向文章作者的通知模块推送一条信息。

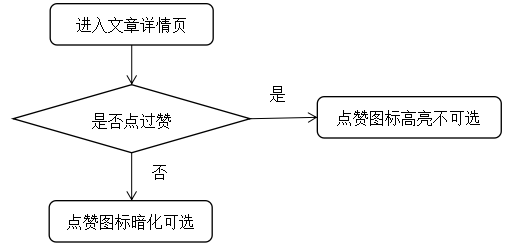


图3.10 点赞功能流程图

## 3.2 设计模式

### 3.2.1 MVC设计模式

对于MVC模式，大多数开发者都知道M是数据模型（Model），V是视图（View），C是控制器（Controller）。在MVC里，View是可以直接访问Model的。从而，View里会包含Model信息，不可避免的还要包括一些业务逻辑。 MVC模型关注的是Model的不变，所以，在MVC模型里，Model不依赖于View，但是 View是依赖于Model的[2]。不仅如此，因为有一些业务逻辑在View里实现了，导致要更改View也是比较困难的，至少那些业务逻辑是无法重用的。

UI开发中最广泛的引用模式可能是模型视图控制器（MVC），它也是最引人注目的。坦率地说，很多经典的基于MVC的框架对于这些富客户端来说并不真实[3]。

### 3.2.2 MVVM设计模式

(1) MVVM介绍

Model-view-viewmodel（MVVM）是一种软件架构模式。MVVM有助于通过开发业务逻MVVM的视图模型是一个值转换器：意味着视图模型负责从模型中显示（转换）数据对象，使得对象容易管理和呈现[4]。 在这方面，视图模型比视图更模型，并处理大部分（如果不是全部）视图的显示逻辑。 视图模型可以实现调解器模式，组织对视图支持的一组用例的后端逻辑的访问。辑或后端逻辑（数据模型），分离图形用户界面的开发（无论是通过标记语言还是GUI代码）。

MVVM是Martin Fowler的演示模型设计模式的一个变体[5]。MVVM以同样的方式提取视图的状态和行为，但是Presentation Model会以不依赖于特定用户界面平台的方式抽象视图（创建视图模型）。

MVVM和Presentation Model都来源于模型 - 视图 - 控制器模式（MVC）。

MVVM专门用于简化用户界面的事件驱动编程。此外，MVVM另一个重要特性，双向绑定。它更方便你同时维护页面上都依赖于某个字段的N个区域，而不用手动更新它们。

(2) MVVM模式的组件

A. 模型（Model）

模型是指代表真实状态内容（面向对象的方法）的域模型，也可以是代表内容（以数据为中心的方法）的数据访问层。

B. 视图（View）

与MVC和MVP模式一样，视图是用户在屏幕上看到的结构，布局和外观。

C. 视图模型（View model）

View model是暴露公共属性和命令的视图的抽象概念。代替MVC模式的控制器或MVP模式的演示者，MVVM具有绑定器。在视图模型中，粘合剂介导视图和数据绑定器之间的通信视图模型已被描述为模型中数据的状态。

D. 绑定器（Binder）

声明式数据和命令绑定在MVVM模式中是隐含的。在Microsoft解决方案堆栈中，Binder是一种名为XAML的标记语言。绑定器使开发人员不必编写繁杂的逻辑来同步视图模型和视图。当在Microsoft堆栈之外实现时，声明性数据绑定技术的存在是该模式的关键推动因素。

(3) 解释

MVVM旨在利用WPF（Windows Presentation Foundation）中的数据绑定功能，通过从视图层中删除几乎所有的GUI代码（“代码隐藏”），更好地促进了视图层开发与其余模式的分离。他们可以使用框架标记语言（例如，XAML）来创建数据绑定到视图模型，而不是用户体验（UX）开发人员编写GUI代码，而是由应用程序开发人员编写和维护。角色分离允许交互式设计人员专注于UX需求，而不是业务逻辑的编程。因此，应用程序的层可以在多个工作流中开发，以提高生产率。即使单个开发人员在整个代码库中工作，视图与模型的正确分离更有效率，因为用户界面通常根据最终用户反馈在开发周期中频繁更新。

MVVM模式试图获得由MVC提供的功能开发分离的两个优点，同时利用数据绑定的优点和框架，通过将数据绑定到尽可能接近纯应用模型。它使用绑定器，视图模型和任何业务层的数据检查功能来验证传入的数据。结果是模型和框架驱动尽可能多的操作，消除或最小化直接操纵视图（例如，代码隐藏）的应用程序逻辑。

### 3.2.3系统交互模式设计

开发环境下：

在开发环境下，vue文件需要被webpack编译，而webpack则依赖于node作为服务器来进行热加载，所以ajax发送的请求跨域，需要在服务端写上跨域头（即cors）

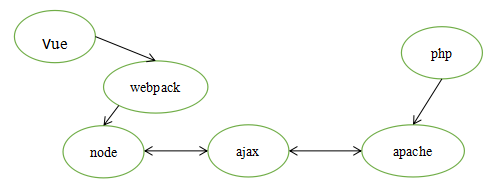


图3.7 开发环境下的技术交互模式

生产环境下：

通过webpack打包后的vue文件则不需要webpack-dev-server来实施热加载了，直接把他放到apache下就好，这时不存在跨域的问题。

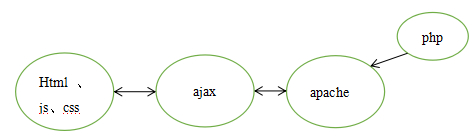


图3.8 生产环境下的技术交互模式

## 3.3技术选型

### 3.3.1 Vue

1.概念

vue是一套构建用户界面的渐进式框架[6]，在2016年与react、angular并称为三大框架之一的前端开发框架，凭借其详细的官方文档、简单灵活的设计模式和高效率的开发流程，深受我国前端开发者的热爱和推崇。

Vue压缩后只有17kb，它采用自底向上增量开发的设计，只关注视图层，不仅易于上手，还便于与第三方库或既有项目整合[7]。

1. 特性

(1) 模板

Vue使用基于HTML的模板语法，允许您将渲染的DOM声明性地绑定到底层的Vue实例的数据。所有Vue模板都是有效的HTML，可由规范兼容的浏览器和HTML解析器解析。在引擎盖下，Vue将模板编译成虚拟DOM渲染功能。结合反应性系统，Vue能够智能地找出要重新渲染的最小数量的组件，并在应用程序状态更改时应用最少量的DOM操作。

在Vue中，您可以使用模板语法或选择使用JSX直接编写渲染功能。为了这样做只需用render函数替换template选项。渲染功能为强大的基于组件的模式开辟了可能性 - 例如，新的过渡系统现在完全基于组件，在内部使用渲染功能。

(2) 活动

Vue最独特的功能之一是不引人注意的反应系统。模型只是简单的JavaScript对象。修改它们时，视图将更新。它使状态管理非常简单直观。 Vue提供了优化的重新渲染开箱即用，无需执行任何操作。每个组件在渲染过程中跟踪其反应依赖关系，因此系统会精确地知道何时重新呈现，以及哪些组件重新呈现。

(3) 组件

组件是Vue最强大的功能之一。在大型应用中，有必要将整个应用程序分成小型，独立的，经常可重复使用的组件，使开发变得易于管理。组件扩展基本的HTML元素以封装可重用的代码。在高层次上，组件是Vue编译器附加行为的自定义元素。在Vue中，组件本质上是一个具有预定义选项的Vue实例。

### 3.3.2 Webpack

Webpack是一个开源的JavaScript模块绑定器。 Webpack采用依赖关系的模块，并生成代表这些模块的静态资产[9]。它采用依赖关系，并生成依赖图，允许Web开发人员使用模块化方法来进行Web应用程序开发。 捆绑器可以从命令行使用，也可以使用名为webpack.config.js的配置文件进行配置。

需要Node.js才能安装webpack。 关于webpack的另一个重要方面是它可以通过使用装载机来高度可扩展[10]。 装载器允许开发人员在将文件捆绑在一起时编写他们想要执行的自定义任务。

Webpack使用标记代码分割来提供需要的代码。 ECMAScript的技术委员会39正在开发一个加载附加代码的函数的标准化：proposal-dynamic-import。

Webpack也是当下最流行的模块加载器和打包工具，他可以把js、jsx、css等前端资源文件打包为模块进行使用。在本系统中，因为使用到了vue文件，wepack的vue-loader将会自动将他编译成js，开发者不必进行各种配置。

### 3.3.3 Ajax

Ajax（“异步JavaScript和XML”的缩写）是一组Web开发技术，在客户端使用许多Web技术来创建异步Web应用程序。 使用Ajax，Web应用程序可以异步（在后台）将数据发送到服务器并从服务器检索，而不会影响现有页面的显示和行为。 通过将数据交换层与表示层分离，Ajax允许Web页面和扩展Web应用程序动态地更改内容，而无需重新加载整个页面。实际上，现代实现通常将JSON替换为XML，这是因为JavaScript本身的优点[11]。

Ajax在近些年来已经成为与服务端交互工具的不二之选，使用ajax能够使代码更明确，前后端分离更清晰；最大的特点是异步，使服务器减轻负担，可以给用户更好的体验。

Ajax不是一种技术，而是一组技术。 可以组合使用HTML和CSS来标记和样式信息。 使用JavaScript访问DOM以动态显示，并允许用户与所呈现的信息进行交互。[12] JavaScript和XMLHttpRequest对象提供了一种在浏览器和服务器之间异步交换数据以避免全页重新加载的方法。

### 3.3.4 Apache+php+mysql

这套技术栈已经流行了很多年，主要是因为php对mysql都有非常友好的扩展。

Php稳定高效，最大的优点是在apache上不用配置很多，开发速率很快，代码简单易读，很适合中小型项目。

Apache HTTP Server，俗称Apache，是基于Apache License 2.0的免费开源跨平台Web服务器软件。 Apache由Apache Software Foundation主持的开放社区开发和维护[13]。

最常用于Unix系统（通常是Linux），该程序也可用于Microsoft Windows。2.0版改进了对非Unix的支持，例如Windows和OS / 2（和eComStation）。Apache的旧版本被移植到例如OpenVMS，和NetWare。

最初基于NCSA HTTPd服务器，Apache的开发始于1995年初，NCSA代码停滞后工作。 Apache在万维网的初步发展中发挥了关键作用，迅速超越了NCSA HTTP作为主流的HTTP服务器，自1996年4月以来一直保持最受欢迎。2009年，它成为第一个提供更多服务的Web服务器软件超过1亿个网站。

截至2016年7月，Apache仍然是使用最广泛的Web服务器软件，估计占所有活动网站的46％，百分之百的网站占43％。

在Web开发的服务器端脚本语言中，不得不提到PHP，他也可以用作通用的编程语言。最初由Rasmus Lerdorf于1994年创建，PHP参考实现现在由PHP开发团队生成。PHP最初代表个人主页，但它现在代表递归的首字母缩写PHP：超文本预处理器。

PHP代码可能嵌入到HTML或HTML5标记中，或者可以与各种Web模板系统，Web内容管理系统和Web框架结合使用。 PHP代码通常由作为Web服务器中的模块实现的PHP解释器或通用网关接口（CGI）可执行文件来处理。 Web服务器软件将解释和执行的PHP代码的结果组合在一起，该代码可以是生成的网页的任何类型的数据，包括图像。 PHP代码也可以使用命令行界面（CLI）执行，可用于实现独立的图形应用程序[14]。

由Zend Engine提供支持的标准PHP解释器是根据PHP许可证发布的免费软件。 PHP已经被广泛移植，可以免费部署在几乎所有操作系统和平台上的大多数Web服务器上。

PHP语言在没有书面形式规范或标准的情况下才会演变，直到2014年，将规范的PHP解释器作为事实上的标准。自2014年以来，工作已经开始创建一个正式的PHP规范。

MySQL是一个开源关系数据库管理系统（RDBMS），联合创始人Michael Widenius的女儿的名字和“SQL”，结构化查询语言的缩写。 MySQL开发项目已经根据GNU通用公共许可证的条款以及各种专有协议规定了其源代码。 MySQL由一家盈利性公司（由Oracle Corporation拥有的瑞典公司MySQL AB）所拥有和赞助。对于专有使用，可以使用几种付费版本，并提供附加功能。

MySQL是LAMP开源Web应用软件堆栈（和其他“AMP”堆栈）的核心组件。 LAMP是“Linux，Apache，MySQL，Perl / PHP / Python”的缩写。使用MySQL数据库的应用程序包括：TYPO3，MODx，Joomla，WordPress，phpBB，MyBB和Drupal[15]。 MySQL也用于许多高调的大型网站，包括Google （虽然不是搜索），Facebook， Twitter，Flickr，和YouTube。

## 3.4 开发工具

### 3.4.1 WebStorm

Webstorm是前端最优秀的ide，它集成了node、npm、webpack、eslint等各种js开发所用的插件的配置，自动纠错，代码高亮。它也集成了当前多种版本控制工具，大部分的功能都可以使用。最主要的是它支持node.js的调试，不需要繁杂的配置就可以进行多种操作。令人欣喜的是，webstorm内置ftp，直接就可以将他部署到远程服务器上。

### 3.4.2 PhpStorm

看名字就知道和WebStorm是来自一家公司，都是jet brain的旗舰产品，使开发过程更高效。经过简单的配置就可以和你的MYSQL连接到一起，实时纠错，语法高亮。可以直接在里面浏览数据库，浏览远程服务器上的文件，也是内置ftp，可以直接将程序部署到服务器上，可以在这个IDE上进行任何从开发到部署环节中的操作。

### 3.4.3 Sublime text3

Sublime是拥有非常多插件的一款文本编辑器，即使插件再多也不会造成卡顿的现象，使用手感非常流畅。Sublime Text是具有Python应用程序编程接口（API）的专有跨平台源代码编辑器。 它本身支持许多编程语言和标记语言，其功能可以由具有插件的用户扩展，通常由社区构建并在免费软件许可证下维护。

# 详细设计

## 4.1 数据库设计

### 4.1.1 通知表（notification）

通知表如表4.1所示。通知表是首页通知模块的通知列表的数据。其中type为通知类型，1代表成功，0代表失败，-1代表警告。

表4.1 通知表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 长度 | 主键 | 是否可为空 | 描述 |
| Id | Int | 20 | Y |  | 通知id |
| userid | Int | 11 |  |  | 用户id |
| Title | Char | 20 |  |  | 通知内容 |
| Type | Int | 1 |  |  | 通知类型 |
| createTime | timestamp |  |  |  | 创建时间 |

### 4.1.2 评论表（comment）

评论表如表4.2所示。评论表用于存储文章详情页的评论列表中的数据。

表4.2 评论表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 长度 | 主键 | 是否可为空 | 描述 |
| id | Int | 20 | Y |  | 评论id |
| articleId | Int | 11 |  |  | 文章id |
| userId | Int | 11 |  |  | 用户id |
| content | Varchar | 100 |  |  | 评论内容 |
| createTime | timestamp | / |  |  | 创建时间 |

### 4.1.3 文章表（article）

文章表如表4.3所示。其中isHot字段是推荐到首页的标识符，likeNum是文章的获赞数，每次点赞加一，评论数同理，每次获得评论都会加一，在关注页的文章列表的按热度查询会根据likenum这个字段进行排序。

表4.3 文章表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 长度 | 主键 | 是否可为空 | 描述 |
| id | Int | 20 | Y |  | 文章id |
| userid | Int | 11 |  |  | 作者id |
| username | Varchar | 10 |  |  | 作者昵称 |
| title | Varchar | 20 |  |  | 文章标题 |
| content | text | / |  |  | 文章内容 |
| categoryId | Int | 10 |  |  | 分类id |
| likeNum | Int | 3 |  |  | 喜欢人数 |
| commentNum | Int | 3 |  |  | 评论数 |
| isHot | Smallint | 1 |  |  | 是否热门 |
| createTime | timestamp | / |  |  | 创建时间 |

### 4.1.4 关注表（attention）

关注表如表4.4所示。这是个关系表，关注页的关注作者列表功能会从这个表中拿数据。

表4.4 关注表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 长度 | 主键 | 是否可为空 | 描述 |
| id | Int | 11 | Y |  | 关注id |
| userid | Int | 11 |  |  | 用户id |
| writerid | Int | 11 |  |  | 被关注用户id |
| writername | Varchar | 20 |  |  | 被关注用户名 |

### 4.1.5 用户信息表（userInfo）

用户信息表如表4.5所示。这个表的数据用来存储首页的推荐作者和作者详情页的数据。其中isHot为首页推荐作者的标识符，1为推荐，默认为0。

表4.5 用户信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 表名 | 主键 | 是否可为空 | 描述 |
| id | Int | 11 | Y |  | 用户id |
| nickname | Varchar | 10 |  |  | 用户昵称 |
| email | Varchar | 20 |  |  | 邮箱 |
| city | Varchar | 8 |  |  | 居住城市 |
| birth | date | / |  |  | 生日 |
| sex | Binary | 1 |  |  | 性别 |
| fansNum | Int | 10 |  |  | 粉丝数 |
| isHot | Smallint | 1 |  |  | 是否被推荐 |

### 4.1.6 用户登录表（userlog）

用户登录表如表4.6所示。这个表是用来存储用户登录页和注册页登录字段信息的数据表。其中type为用户类型，1代表管理员，默认为0，代表普通注册用户。

表4.6 用户登录信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 长度 | 主键 | 是否可为空 | 描述 |
| id | Int | 11 | Y |  | 用户id |
| username | Varchar | 10 |  |  | 用户登录名 |
| psd | Varchar | 15 |  |  | 用户密码 |
| nickname | Varchar | 8 |  |  | 用户昵称 |
| type | Int | 1 |  |  | 用户类型 |

## 

## 4.2 接口设计

### 4.2.1 登录页

登录页的接口如表4.7所示。

isLogin这个接口的功能是进行登录模块的校验，当用户登录系统点击“登录按钮”时请求这个接口，返回状态码，当状态码为1的时候代表请求成功，并返回用户类型，如果请求失败返回0。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 所在模块 | 接口功能 | 接口名称 | 请求字段 | 返回字段 |
| 登录页 | 登陆校验 | User/index/isLogin | Username：用户名 password：密码 | Type:1是管理员/0是用户 |

表4.7 登录页包含的接口

### 4.2.2 注册页

注册页的接口如表4.8所示。

regist这个接口的功能是用来提交注册信息，当用户进入登录模块，填写注册表单后，点击“注册”时请求这个接口，返回状态码，当状态码为1的时候代表请求成功，如果请求失败返回0，并携带失败信息。

表4.8 注册页包含的接口

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 所在模块 | 接口功能 | 接口名称 | 请求字段 | 返回字段 |
| 注册页 | 注册 | User/index/regist | Username:用户名  Psd：密码  Nickname:用户昵称  Email：邮箱  City:常住地  Birth：生日  Sex：性别（1男/0女） | msg：失败信息 |

### 4.2.3 文章编辑页

文章编辑页的接口如表4.9所示。

saveContent这个接口的功能是用来保存编辑后的文章信息，当用户进行编辑指定文章或新增文章时点击“保存并上传”时请求这个接口，返回状态码，当状态码为1的时候代表请求成功，如果请求失败返回0，并携带失败信息。

getArticleCategory这个接口的功能是获取用户的文章分类的列表，当用户进行编辑或新增文章时请求这个接口，将数据渲染到分类列表上，返回状态码，当状态码为1的时候代表请求成功，并返回该用户分类列表的相关字段，如果请求失败返回0。

addCategory这个接口的功能是新建文章分类，当用户进入编辑页，下拉分类列表时，点击最后的“增加分类”时，打开新窗口输入新分类的名称后，点击“确定”按钮时，请求这个接口，返回状态码，当状态码为1的时候代表请求成功，如果请求失败返回0。

表4.9 编辑页包含的接口

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 所在模块 | 接口功能 | 接口名称 | 请求字段 | 返回字段 |
| 文章编辑页 | 文章保存 | User/edit/saveContent | textId：文章id（新增为空）  userId：作者id  Title：文章标题  Content：文章内容  categoryId：分类id | msg：失败信息 |
|  | 获取用户的文章分类 | User/edit/getArticleCategory | Userid：用户id | Id：分类的id  Title：分类的名称 |
|  | 创建分类 | User/edit/addCategory | Userid：用户id  Title:分类名称 | msg：失败信息 |

### 4.2.4 文章列表页

文章列表页的接口如表4.10所示。

isLogin这个接口的功能是进行登录模块的校验，当用户登录系统点击“登录按钮”时请求这个接口，返回状态码，当状态码为1的时候代表请求成功，并返回用户类型，如果请求失败返回0。

isLogin这个接口的功能是进行登录模块的校验，当用户登录系统点击“登录按钮”时请求这个接口，返回状态码，当状态码为1的时候代表请求成功，并返回用户类型，如果请求失败返回0。

表4.10 文章列表页包含的接口

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 所在模块 | 接口功能 | 接口名称 | 请求字段 | 返回字段 |
| 文章列表页 | 拉取文章列表 | User/edit/getArticleList | userId:用户登录名 | id：文章id  title：文章标题  content：文章内容  likeNum：点赞数  createTime：创建时间 |
|  | 删除文章 | User/edit/deleteArticle | Id:文章id | msg：失败信息 |

### 4.2.5 首页

首页的接口如表4.11所示。

goodWriters这个接口的功能是用来获取首页推荐作者列表，当用户进入首页时请求这个接口，返回状态码，当状态码为1的时候代表请求成功，并返回推荐作者列表，如果请求失败返回0。

goodArticle这个接口的功能是用来获取推荐文章的，当用户进入首页时请求这个接口，返回状态码，当状态码为1的时候代表请求成功，并返回推荐文章列表，如果请求失败返回0。

setHotArticle这个接口的功能是用来设置推荐文章的，当管理员进入推荐文章模块时，对指定文章点击“首页显示/取消显示”这个按钮时请求这个接口，返回状态码，当状态码为1的时候代表请求成功，如果请求失败返回0。

setHotWriter这个接口的功能是用来设置推荐作者的，当管理员进入推荐作者模块时，对指定作者点击“首页显示/取消显示”这个按钮时请求这个接口，返回状态码，当状态码为1的时候代表请求成功，如果请求失败返回0。

showArticleList这个接口的功能是用来获取所有投稿文章的，当管理员进入推荐文章模块时请求这个接口，返回状态码，当状态码为1的时候代表请求成功，并返回投稿文章列表，如果请求失败返回0。

showNotification这个接口的功能是用来显示通知列表的，当用户进入首页点击“显示通知”时请求这个接口，返回状态码，当状态码为1的时候代表请求成功，并返回该用户的通知列表，如果请求失败返回0。

followArticle这个接口的功能是用来关注首页推荐作者的，当用户进入首页的推荐作者模块对指定作者点击“关注”时请求这个接口，返回状态码，当状态码为1的时候代表请求成功，如果请求失败返回0。

表4.11 首页包含的接口

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 所在模块 | 接口功能 | 接口名称 | 请求字段 | 返回字段 |
| 首页 | 获取推荐作者 | User/index/goodWriters | null | userId:用户id  userAvatar：用户头像userName：用户名  likeNum：赞数  fansNum：粉丝数 |
|  | 获取推荐文章 | User/index/goodArticle | null | textId：文章id  userId：用户id  Title：标题  Content：内容  categoryId：分类id  likenum：点赞人数  isHot：是否推荐 |
|  | （管理员）设置推荐文章 | Admin/index/setHotArticle | articleId：文章id  Status：1设为推荐/0取消推荐 | true/false |
|  | （管理员）设置推荐作者 | Admin/index/setHotWriter | articleId：文章id  Status：1设为推荐/0取消推荐 | true/false |
|  | （管理员）获取所有文章 | Admin/index/showArticleList | null | articleId：文章id  Title：文章标题  Status：状态 |
|  | 显示通知 | Index/showNotification | userId：用户id | Id：通知id  Title：通知标题  type:通知类型 |
|  | 关注作者 | Index/followArticle | Id：当前用户  writerId：要关注的作者id  Writername：要关注的作者名字 | Status：1成功/0失败 |

### 4.2.6 关注页

关注页的接口如表4.12所示。

showWriterList这个接口的功能是用来获取关注列表的，当用户进入关注页时请求这个接口，返回状态码，当状态码为1的时候代表请求成功，并返回关注作者的列表，如果请求失败返回0。

showArticleList这个接口的功能是用来获取指定关注作者的文章列表的，当用户进入关注页时选定指定作者或默认请求第一个关注作者的id（前端处理）时请求这个接口，返回状态码，当状态码为1的时候代表请求成功，并返回指定作者的文章列表，如果请求失败返回0。

表4.12 关注用户页包含的接口

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 所在模块 | 接口功能 | 接口名称 | 请求字段 | 返回字段 |
| 关注页 | 获得关注列表 | Attention/showWriterList | userId：用户id | writerId：作者id  Name：作者名 |
|  | 读取关注作者的文章列表 | Attention/showArticleList | writerId：作者id  Type：1:按照热门排序/0：按照时间排序 | articleId：文章id  Title：文章标题  Content：文章内容 |

### 4.2.7 文章详情页

文章详情页的接口如表4.13所示。

showArticle这个接口的功能是用来显示文章详情的，当用户从文章列表选择某篇文章进行查看时请求这个接口，返回状态码，当状态码为1的时候代表请求成功，并返回该文章的详情，如果请求失败返回0。

showCommentList这个接口的功能是用来显示指定文章的评论列表的，当用户进入文章详情页，点击“展开评论”按钮时请求这个接口，返回状态码，当状态码为1的时候代表请求成功，并返回该文章的评论列表，如果请求失败返回0。

doComment这个接口的功能是用来提交评论内容的，当用户进入文章详情页，输入相关评论后，点击“评论”时请求这个接口，返回状态码，当状态码为1的时候代表请求成功，如果请求失败返回0。

表4.13 文章详情页包含的接口

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 所在模块 | 接口功能 | 接口名称 | 请求字段 | 返回字段 |
| 文章详情页 | 展示文章详情 | Detail/showArticle | articleId：文章id | writerId：作者id  writerName：作者名content：文章内容  createTime：创建时间  praiseNum：点赞人数 |
|  | 展示文章评论 | Detail/showCommentList | articleId：文章id | Id：评论id  userId：评论人id  username：评论人名  createTime：创建时间  content：评论内容  isPraise：当前用户是否点过赞 |
|  | 对文章进行评论 | Detail/doComment | articleId：文章id  userId：用户id  Content：内容 | Status：(1成功/0失败) |

# 第五章 系统实现

## 5.1 开发步骤

本作者是先写的前端部分，使用vue的ui库iview，采用了当下最流行的material design来进行构建初级架构；在每个页面的跳转上使用了vue的官方路由插件，vue-router，在视图层搭建一个路由；在和数据的交互方面，由于vue是mvvm架构，也就是数据驱动的，所以先构造json格式的假数据进行交互，等到后端写完接口再绑定数据，把假数据删了就好。

后端部分写的是原生的php，因为本系统有两个角色，分别是注册的用户和管理员，分成了两个文件夹，因为大多数都是注册用户在使用系统，每个分页都被分成了不同的文件，使维护的的时候可以更加明确方便。

接口方面，ajax请求chose字段暴露接口内容，后端根据chose的接口进行对数据的处理，每次响应字段包含status状态码，1为响应成功，0为失败。

## 5.2 主要配置文件

### 5.2.1 package.json文件

Package.json实际上是由node本身来使用的。当你需要引用一个外部的库或框架时，node中的npm模块会将这个库的依赖库或文件写入package.json，安装的依赖在node\_modules文件夹下；当你想根据package.json安装依赖时，只需要运行命令行npm install即可。

### 5.2.2 vue-router配置文件

vue-router是Vue.js的官方路由器。它与Vue.js核心深入整合，使Vue.js构建单页应用程序变得轻而易举。特点包括：

1. 嵌套路由/视图映射
2. 路由参数，查询，通配符
3. 模块化的基于组件的路由器配置
4. 查看Vue.js转换系统提供的转换效果
5. 细粒度导航控制
6. 与自动活动CSS类的链接
7. HTML5历史记录模式或哈希模式，在IE9中自动回退
8. 可定制的滚动行为

### 5.2.3 webpack配置文件

在webpack的配置中，有三个核心文件，分别是webpack.base.conf.js，webpack.dev.con.js，webpack.prod.conf.js。

看文件的名字就知道他们是要处理什么的配置文件了。Webpack.base.conf.js是基础配置，主要包括唯一入口的文件路径，打包后的出口文件路径，各种静态资源的loader等。Webpack.dev.conf.js是在开发环境中用到的，主要包括wepack-dev-server的配置。Webpack.prod.conf.js是在生产环境中使用的，主要是来配置打包后的生成文件路径和打包方式等。

## 5.3 核心问题及解决方案

### 5.3.1用户的个性域名

本系统要求每个已经注册的用户都有包含自己登录用户名（非昵称）的个人的基于本站的独立域名。在本系统中，为了实现路由的跳转，即每个单页面的切换，使用的是vue官方的切换url的工具，vue-router。

在单页应用程序（SPA）中，一个最明显的缺点是无法分享到特定网页中确切“子”页面的链接，即无法从一个路径带着参数跳转到下一个链接路径。由于SPA只为其用户提供了一个来自服务器的基于URL的响应（通常会为index.html或index.vue提供服务），因此保存书签或者与特定文章共享链接是不可能的。为了解决这个问题，前端路由器提供了最初由hashbangpage.com拆分的人造基于哈希的URL，大多数现代浏览器支持路由，而不使用hashbang。像Vue这样的JavaScript库可以提供一个简单的界面来根据当前的URL路径来更改页面上显示的内容。无论是通过电子邮件链接，刷新还是页内链接进行更改。另外，使用前端路由器允许在某些浏览器事件（即点击）发生在按钮或链接上时有意地转换浏览器路径。 Vue本身并没有带有前端散列路由。但是，开源的“vue-router”包提供了一个更改浏览器URL的API，使用后退按钮（哈希历史），以及使用URL中提供的验证参数的电子邮件密码重置或电子邮件验证链接。它支持将嵌套路由映射到嵌套组件，并提供细粒度的转换控制。使用Vue + vue-router创建前端路由单页面应用程序变得简单。使用Vue，开发人员已经在构建较大组件的小型构建块中构建应用程序。

用vue-router实现在路由中拥有当前用户名，首先，需要在路由的配置文件及main.js中添加如下代码：

'/:userId': {

name:'index',

component: index

},

这句的意思是，在跳转到首页（index）的时候提供一个名为userId的参数，在每次跳转的时候，使用$router.go(‘index’,params)这句，params代表需要传递的参数，即完成了带参跳转。

那么，如何才能知道当前的用户名呢。

用户在使用该系统需要进行登陆操作，所以在登陆成功之后，既后台返回响应参数status为1后，得到了该用户的用户名，此时，使用$router.go，把用户名传递到该路径上，既实现了带参跳转。

### 5.3.2 编辑文章时防止意外退出，用local storage保存

在编辑页写内容的时候，常常会发生一些难以预料的操作，比如说停电，或是突然断网需要刷新等，这时候刚刚编辑的内容还没来得及保存，就会消失。但每次更改，编辑某些字段内容时频繁发送请求就会使服务器繁忙，所以为了解决这个问题，本系统使用local storage来保存每次变更的内容。

Web存储提供两个不同的存储区域，本地存储和会话存储。其范围和使用寿命不同。放置在本地存储器中的数据是每个来源（同源策略中定义的协议，主机名和端口号的组合）（该数据可用于从先前存储数据的同一来源的页面加载的所有脚本）浏览器关闭后仍然存在。 会话存储是根据每个窗口或标签，并且仅限于窗口的生命周期。会话存储旨在允许同一Web应用程序的单独实例在不同的窗口中运行，而不会彼此干扰，Cookie的使用情况并不好。与服务器和客户端都可以访问的Cookie不同，Web存储完全属于客户端脚本的范围。

Web存储数据在每个HTTP请求中都不会自动发送到服务器，而Web服务器无法直接写入Web存储。 然而，这些效果中的任何一个都可以通过显式的客户端脚本来实现，从而允许微调所需的与服务器的交互。

支持Web存储的浏览器具有在窗口级别声明的全局变量localStorage。 这些浏览器可以使用以下JavaScript代码来触发Web存储行为：

localStorage.setItem('key', 'value');

### 5.3.3 每次切换模块时，加载资源时间过长的问题

使用传统即原生的架构，会出现当切换到某个页面的时候，会从服务器请求并加载，这需要一定的时间和网络资源，并且如果重复切换这个页面仍然会耗费资源和时间。所以本系统为了优化这个问题，采用的是vue这一JavaScript框架，来解决这些问题。

单页应用程序（SPA）是一个适合单个网页的Web应用程序或网站，目的是提供与桌面应用程序类似的用户体验。 在SPA中，通过单个页面加载检索+所有必需的代码（HTML，JavaScript和CSS），或者根据需要动态加载适当的资源并将其添加到页面，通常是响应于用户操作。 该页面不会在进程的任何时间重新加载，也不会将传输转移到另一个页面，尽管位置哈希或HTML5历史记录API可用于提供应用程序中单独的逻辑页面的感知和导航。

在初始页面加载中完全加载SPA，然后根据需要从服务器加载的新页面片段替换或更新页面区域。

### 5.3.4 markdown转码的问题

在编辑模块时，所使用的编辑器为基于markdown的编辑器。由于直接写markdown文法不能直观地查看和审阅整个文章的样式和结构，本系统使用基于vue的mardown编辑器[vue-simplemde](https://github.com/F-loat/vue-simplemde)来实现这个功能。用户在编辑的时候，可以边写边预览了。

在查看详情页的时候也需要将数据库中的markdown语法的文本，转换成带有格式的文本，这里引入了markdown自身的编译器，来解决这一问题。

## 5.4程序的实现

### 5.4.1 开发流程

先配置好开发环境，写前端的界面，然后是编写假数据写业务逻辑；局部测试完成后，编写服务端；用一个test页进行单元测试，测试通过后，在接口文档上标注完成标识；全部接口调通后，前端绑定接口，删除假数据。

### 5.4.2部署流程

运行npm run build，webpack会自动把所有前端代码打包好，先本地测试一下是否在不发请求的状态下能正常运行，由于开发环境是跨域的，所以在生产环境下先更改请求信息，如果不打算二次开发或进行维护，服务端可以删除用于跨域请求的代码，来保证安全性，之后把所有打包好的程序放到远程服务器上就可以了。

### 5.4.3 设计结果展示

如图5.1，是登录界面，如果输入为空值会提醒。

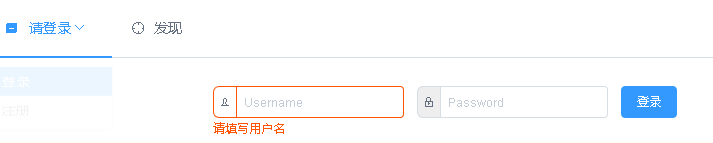


图5.1 登录界面

如图5.2为注册界面，如果输入为空值或非法字段会提醒。



图5.2 注册页展示

如图5.3为未登录的首页（发现页），只能进行登录/注册操作或查看首页推荐文章。



图5.3 未登录首页展示

如图5.4为登录后的首页（发现页），登录之后的用户可以进行查看关注列表和写日志等操作。



图5.4 （已登录状态）首页展示

如图5.5为文章新增页，用户通过tab栏“写日志”进入。

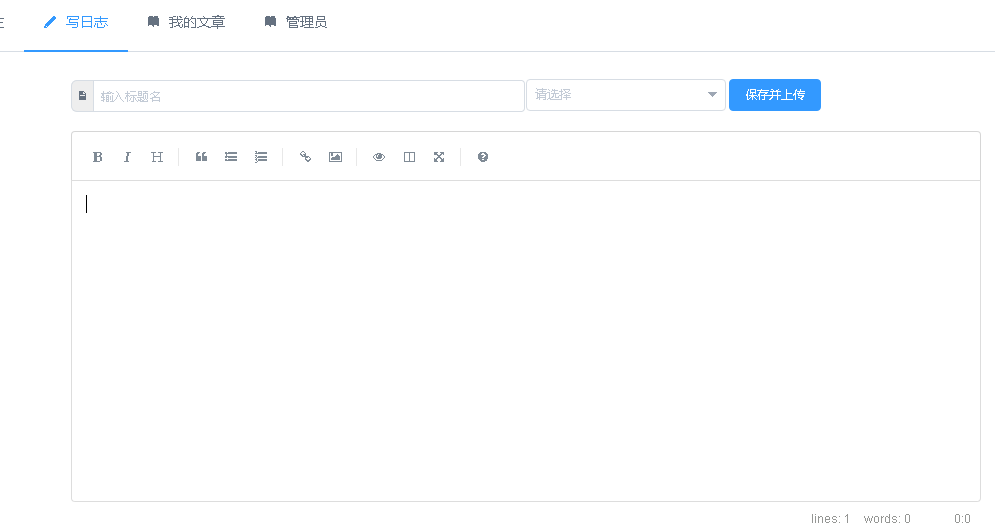


图5.5 文章新增页展示

如图5.6和图5.7分别为选择分类列表和新建分类。



图5.6 选择分类列表展示

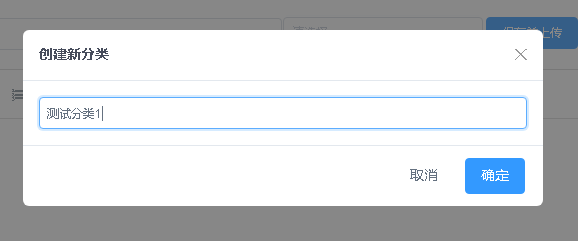


图5.7 新建分类展示

如图5.8，左列为关注作者列表，右列为该作者的文章列表，可以按热度和时间顺序排。



图5.8 关注页展示

如图5.9，展示用户自己的文章列表，右边为分类列表。在这页可以对文章进行编辑（如图5.10）和删除（如图5.11）的操作。



图5.9 我的文章页展示

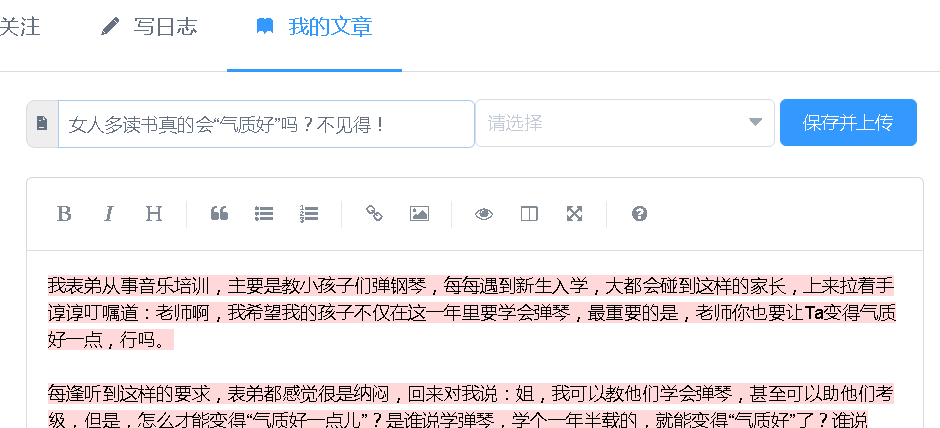


图5.10 我的文章页编辑操作展示



图5.11 我的文章页删除操作展示

如图5.12，展示文章详情。用户可以查看评论、进行点赞和评论。如图5.13点击“展开评论”可以查看该文章的评论列表。

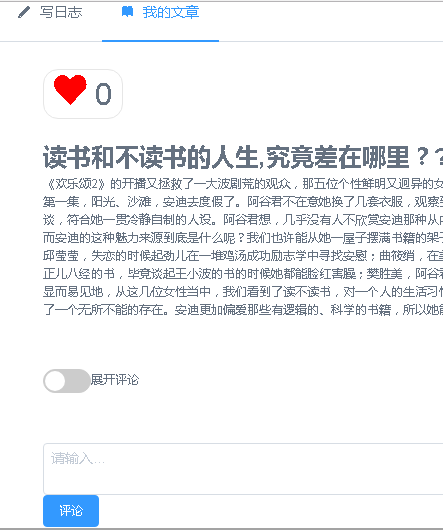


图5.12 文章详情页展示

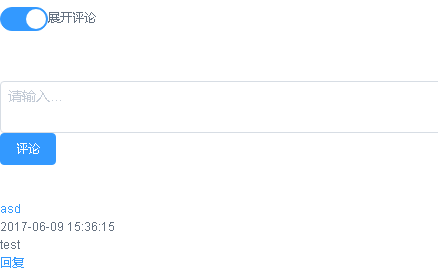


图5.13 评论列表展示

如图5.14，管理员登录后，可以把指定文章显示到首页。



图5.14 管理员登录后的文章推送页展示

# 参考文献

[1] Nikos Kasioumis．Towards building a blog preservation platform[J]．World Wide Web, 2014, Vol.17 (4)：799-825

[2] 李展飞．Web软件系统开发框架设计在MVC模式的实现[J]．电子技术与软件工程，2017．(08)：61

[3] WikiPedia．Model–View–Controller History[EB/OL]．维基百科，http://wiki.c2.com/

?Model ViewControllerHistory．2013-12-09

[4] 陈涛．MVVM设计模式及其应用研究[D]．计算机与数字工程, 2014, (10)：1982-1985.

[5] Msdn．The MVVM Pattern[EB/OL]．Msdn.microsoft.com．https://msdn.microsoft.com/en- us/library/hh848246.aspx．2016-08-29

[6] Evan You．Introduction of Vue.js[EB/OL]．Vue.js,https://vuejs.org/v2/guide/#

What-is-Vue- js.html．2017-03-11

[7]  Evan You．[First Week of Launching Vue.js](http://blog.evanyou.me/2014/02/11/first-week-of-launching-an-oss-project/index.html)[EB/OL]．Evan You’s Blog, http://blog.evanyou.me /2014/02/11/first-week-of-launching-an-oss-project/index.html, 2017-03-11

[8] 易剑波．基于MVVM模式的WEB前端框架的研究[D]．信息与电脑(理论版), 2016, (19)：76-77+84

[9] [Saransh Kataria](https://www.wisdomgeek.com/author/saranshkataria/)．Webpack: An Introduction[EB/OL]．Wisdom Geek．https://www.

　　wisdomgeek. com/web-development/webpack-introduction, 2017-01-12

[10] 彭娜．基于Node.JS博客系统的设计与实现[D]．大连理工大学，2013

[11] 温立辉． AJAX异步交互技术浅析[J]．山东工业技术，2017，(04)：213

[12] Douglas Crockford．[JSON: The Fat-Free Alternative toXML](http://www.json.org/xml.html)[EB/OL]．http://www.

json.org /xml.html．2017-02-17

[13] 周奎，王超，黄连丽．基于PHP与MySQL的教务管理系统设计[J]．软件导刊，2017，(05)：89-90

[14] 李华明．基于PHP和MySQL的网上购物系统设计与实现[D]．电子科技大学，2014.

[15] 赵红霞，王建．基于PHP+MySQL结构的微课在线学习系统设计与实现[J]．信息通信，2017，(03)：84-85

# 谢 辞

经过大学这四年的熏陶，我对编程这方面的兴趣从0到1，不要小看这只是一个小小的字节变动，但其实这是个布尔值，学校的教育打开了我的兴趣开关，感谢老师的悉心栽培，我希望自己可以在这条道路上越走越远，将来可以报答母校，报效国家！