**基于vue的个人博客的设计与实现**

**摘要**

**互联网成为了当今世界人们对外交流，快速获取、发布和传递信息的最常用也是最便捷的渠道，在人们的日常生活和工作学习的各方面互联网技术都在发挥着不可或缺的作用。所以顺应时代出现了博客。博客网站正好满足了互联网中个人的共享需求。**

**本文对博客的功能与需求进行了完整分析，设计出了一个简单、易用的个人博客系统。结构上分为前台与后台两部分。前台未登陆只能对文章进行浏览，登陆后可以对浏览文章同时也可以对文章进行评论。后台管理系统实现了对文章、用户、文章分类的管理与对应的增删查改操作。为了提高开发效率和代码复用率，本系统使用采用SPA（单页面应用）思想的vue.js进行系统的开发。后端将node.js技术与express结合使用，以创建符合RESTful API设计规范的接口。**

**关键词:博客,vue.js,,node.js ,API,vuex**

**Design and implementation of personal blog based on vue**

**Abstract**

**The Internet has become the most common and convenient channel for people in the world to communicate, quickly acquire, publish and transmit information. Internet technology plays an indispensable role in people's daily life and work and study. So in the era of the emergence of a blog. The blog website just meets the sharing needs of individuals in the Internet.**

**This article provides a complete analysis of the functions and needs of blogs and designs a simple, easy-to-use personal blogging system. The structure is divided into two parts: the foreground and the background. The front desk is not logged in. You can only browse the article. After logging in, you can browse the article and comment on the article. The background management system implements the management of the article, user, and article classification and the corresponding addition, deletion, and modification operations. In order to improve development efficiency and code reuse rate, the system uses vue.js using the SPA (single page application) idea for system development. The backend uses node.js technology in conjunction with express to create interfaces that conform to the RESTful API design specification.**

**Keywords:Blog,Vue.js,Node.js,API,mysql**

# **绪论**

## **1.1 博客的背景**

互联网对人们的生活、工作和学习方式进行了新的定义。Blog是互联网web2.0的产物，是跟继GEnie、BiX、早期的CompuServe、邮件列表、BBS和聊天室之后出现的又一种网络通信方式，是以互联网作为基础，使得人们可以快速，方便的分享个人想法并与他人建立沟通的综合的可以展示自己个人特性的的平台。本系统是按用户划分的个人博客。博客有以下特点：

（1）简单的操作是博客开发的动力。这是博客受到如此众多网友欢迎的最大特点。许多博客托管网站使用的口号就是“在一分钟内轻松拥有一个博客”。

（2）不断更新是博客壮大的催化剂。现代社会，信息传递的速度非常快速，博客的更新就如逆水行舟，在时代的洪流中没有原地不动，只用前进与退步，不及时更新的博客很快就会被淹没在技术前进的道路上。

（3）开放互动是博客传播的推动剂。互联网为博客提供了开放性，因此博客的空间界定变得模糊，不在只是一个简单的私有空间。访客和其他博主写给文章的留言或评论，如果我们回复并通过访客和其他博主留下的地址去返回给他们访问的结果，这就可以实现互动效果。因此，只要善于利用博客开放互动的特点，就可以将博客用于交流和推广，使之最终形成一个固定的圈子。

## **1.2系统设计的意义**

在科技快速进步的今天，互联网成为了当今世界人们对外交流，快速获取、发布和传递信息的最常用也是最便捷的渠道，互联网如今已经渗入人们的日常生活中方方面面。当今时代，技术日新月异，作为互联网技术从业者，需要不断的汲取新知识，走在技术的前沿，但是大部分新技术文档与书籍都是英文，对于英语能力有限的互联网技术从业者，不可避免的需要借助搜素引擎或者其他人的帮助，在搜索引擎的搜索结果中大部分的技术问题解决方案的贡献者都来自博客，所以此次毕业设计自己也想要做一个记录自己学习与工作过程的心得体会的博客网站，让自己对社会和他人可以有所贡献。

## **1.3 国内外研究现状**

Blog一词最早出现在Peter Merholz的日志类网站Peterme.com的侧边栏中，他将Jorn Barger于1997年12月17日创造的术语“网络日志”（Weblog）变成词组“we blog”。由于读者众多，“Blog”很快就被接受并被定义为名词和动词，可以表示编写博客也可以表示博客。

国际上，第一个博客，由Swarthmore大学的学生Justin Hall创建于1994年。那时还不叫博客，当时被称为个人主页。1998年，Jonathan Dube为The Charlotte Observer撰写关于Bonnie飓风的博客，这是传统新闻网站上的第一个博客。

在国内，我国学者孙坚华于1998年博客技术出现后的第一年开始写有关文章，此后国内的博客发展进入了繁荣阶段。

在博客进入发展阶段之前，国内外博客的索引方式是以内容为主的，以一篇一篇的文章作为检索的数据。在博客的发展阶段，提出了语义化博客的设计，建设语义Blog系统,利用语义网的思想解决传统Blog存在的问题。国内外对于博客内容的寻找、传播以及获取方式依然在进行不断。

# **系统相关技术介绍**

## **2.1 SPA（Single Page App）**

单页应用（英语：Single Page Application， SPA）是通过动态重写当前页面与用户交互的Web应用程序或网站的模型。避免了传统网页模型的打断用户体验的页面切换方式，也避免了不断的向服务器发送请求去加载整个新页面，减轻了服务器的负担。在单页应用中，通过加载该html时来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通过路由程序将它们添加到页面中（通常是响应用户操作）。网站所有的页面内容都包含在这个主页面中。但是在实际开发的时候，还是会分开写每一个页面片段。SPA支持多种多样的客户端功能，很少需要重新加载整个页面。因此当用户执行操作或在网站的不同页面之间跳转时，不必重新加载整个页面。另外单页应用它在后台提取数据，对单个用户操作的响应更快。

## **2.2 vue.js**

### 2.2.1 介绍

Vue.js是尤玉溪编写的用来构建用户界面的渐进式 JavaScript 框架 。与其它大型框架不同的是，Vue 被设计为可以自下而上应用。Vue.js的核心库只关注视图层，这种结构使其不仅易于上手，而且还便于与第三方库或现有项目整合。另一方面，当Vue.js与现代化工具链和各种支持的类库结合使用时可以为构建一个复杂的单页应用。

### 2.2.2 vue响应式原理

Vue 最独特的功能之一是其非侵入性的响应式系统。vue如何追踪变化呢？将普通的 JavaScript 对象传给 Vue 实例的 data 选项时，Vue 将迭代此对象所有的属性，并使用 Object.defineProperty 将所有这些属性转为 getter/setter。Vue.js不支持IE8及更低版本的浏览器，这是由于Object.defineProperty是ES5中无法填充的功能，无法用已有的方式组合去实现。这些 getter/setter 对用户是不可见的，但是内部 Vue 通过它们跟踪依赖关系，在属性被访问和修改时通知变化。每个组件实例都有一个相应的 watcher实例对象，该对象在组件呈现的过程将该属性记录为依赖项。稍后，当调用依赖项的 setter时，会通知 watcher重新计算，从而更新其关联的组件。Vue 响应式系统的底层的细节如图2-1所示。



**图 2-1 vue响应式原理**

### 2.2.3 Vue-cli

Vue- cli是基于 webpack构建开发的脚手架工具。拥有一个非常完善的开发生态，开发者只需要关注业务代码本身，而不必操心复杂的 webpack配置，可以说是对开发人员尤其是新手是相当友好的。

Vue CLI 提供了以下功能：(1)通过 @vue/cli 搭建交互式的项目脚手架;(2)通过 @vue/cli + @vue/cli-service-global 快速开始零配置原型开发;(3)一个运行时依赖 (@vue/cli-service)，该依赖：①可升级；②基于 webpack 构建，并带有合理的默认配置；③可以通过项目内存在的配置文件进行自主配置；④可以通过插件进行扩展;⑤丰富的官方插件集，集成了前端生态中的最佳工具;⑥用于创建和管理 Vue.js 项目的完全图形用户界面。

## **2.3 node.js**

### 2.3.1 简介

在node.js出现之前，应用的服务器端实现往往比较复杂。创建服务器的技术门槛比较高，需要有专业的技术知识才可以完成服务器的搭建。并且由于前后端开发语言的差异，使得开发者不得不使用多种编程语言，增加了开发的难度。

Google Chrome浏览器的v8引擎是一个开源的项目，通过简单的API就可以将其集成进浏览器中，v8解决了使用javascript作为服务端语言的性能与内存管理混乱两大问题。

Node.js的作者Ryan Dahl将V8引擎内嵌到了操作系统的集成层实现了将其带到服务器的目的。使得开发者可以使用同样的编程语言就可以完成客户端和服务器端的功能。

### 2.3.2 Node.js Express 框架

Express 是一个简洁而灵活的 node.js Web应用框架, 提供了一系列强大特性帮助开发者创建各种 Web 应用，和丰富的 HTTP 工具。使用 Express 可以快速地搭建一个完整功能的网站。

Express和js框架或者后台框架一样，把经常使用的东西比如方法放到一起，就形成了自己的一套东西，其实就是一个完善的库文件，里面写了大量的函数。Express的思想是在于在工程师的想法和服务器之间作为中介的简单的一层。这并不意味着Express不够健壮，或者没有足够的有用的特性，而是尽量少干预，让开发者充分表达自己的思想，同时提供一些有用的东西。Express 框架核心特性如下：

1. 强大的路由能力。
2. 强大的静态文件渲染能力。
3. 对模版引擎支持。可以通过向模板传递参数来动态渲染 HTML页面。
4. 性能更好。Express 在node.js 上扩展了Web应用所需的功能

安装 Express 并将其保存到依赖列表中：$npm install express --save

# **系统分析**

## **3.1 可行性分析**

可行性分析(Feasibility Analysis)的目的是以最小的代价在尽可能短的时间内确定问题是否能够解决。为了确定开发具有可行性，对本系统主要进行了以下几个方面的分析。

技术可行性分析：本人对于vue.js与node.js在实习中有所接触，对这方面有所了解，所以从技术方面看做个人博客是可行的。

时间可行性：现在博客技术已经非常成熟，多方面的问题都有成熟的解决方案，结合技术可行性，在预定的时间里可以完成此次设计。

## **3.2需求分析**

想要完成一个系统，需求分析是第一步，明确系统是什么方向，要干什么，要完成那些功能。

本系统需求如下：

博客的游客用户可以在网站上对文章进行常规访问。以及在通过注册登陆后成为注册用户，可以进行对已发表文章的评论。博主可以通过后台对用户进行管理，以及对文章与文章分类添加、删除、修改。

针对博客系统的以上需求，总结出如下信息：

1. 用户分为游客、普通用户和超级管理用户。

（2）超级管理用户员可以用户进行管理和设置权限。

（3）博客的超级管理员涉及对博客的文章类型管理、文章管理、评论管理、和用户管理。

（4）用户可以阅读文章、发表评论，游客只能进行文章的阅读。

（5）文章类型与文章之间形成一对多的关系，文章与评论之间也是一对多的关系。

# **系统概要设计**

个人博客结构上分为前台系统，后台系统和数据库。前台主要用作展示，后台进行具体管理。总体规划如图4-1所示。

**图 4-1 总体规划图**

## 4.1前台功能设计

在前台，游客浏览首页，文章详情页，文章列表与文章分类，注册用户在此基础上可以对文章进行评论。前台功能结构如图4-2所示。

**图 4-2 前台功能结构图**

## 4.2后台功能设计

系统的超级管理员通过预设的账号密码登陆后台。可以有所有的权限，对评论进行查看和删除；对注册用户进行管理，可以进行的操作有查看和封禁，对文章进行增删查改，对文章分类进行增删查改。后台功能结构如图4-3所示。

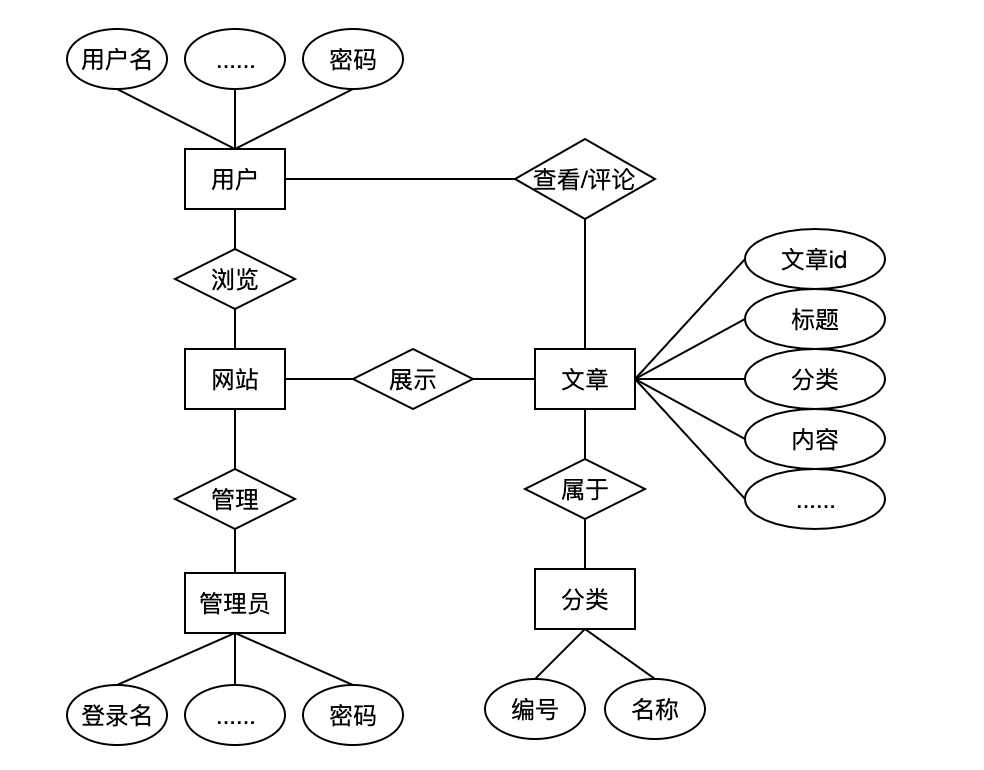
**图 4-3 后台功能结构图**

## 4.3数据库设计

数据库是个人博客设计的主要部分，本系统选用了mysql数据库，使用navicat premium可视化数据库管理工具进行数据库的设计与管理。

### 4.3.1 概念结构设计

对于属性比较多的实体，在E-R图中只列出了部分属性以做示意，具体的详细设计将体现在数据库表的结构中。如图4-4所示。



**图 4-4 数据库概念结构E-R图**

### 4.3.2逻辑结构设计

以各个实体之间的关系为基础，个人博客系统的表结构设计如下列表所示。

1. 用户表管理用户的信息。表结构设计如表4-1所示。

**表 4-1 用户表**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名称** | **数据类型** | **字段长度** | **说明** |
| id | int | 5 | 用户ID(主键) |
| account | varchar | 50 | 登陆名 |
| password | varchar | 50 | 密码 |
| pwd\_salt | int | 5 | 密码加盐随机数 |
| nickname | varchar | 50 | 用户名字 |
| birthday | date | 20 | 出生日期 |
| gender | tinyint | 1 | 性别 |
| introduce | varchar | 200 | 个人介绍 |
| status | tinyint | 1 | 用户状态 |
| role\_id | int | 5 | 角色id |

（2）评论表存储文章的评论，表结构设计如表4-2所示。

**表 4-2 评论表**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名称** | **数据类型** | **字段长度** | **说明** |
| id | int | 5 | 评论ID(主键) |
| art\_id | int | 5 | 文章ID |
| content | varchar | 255 | 评论内容 |
| author | varchar | 20 | 评论人 |
| state | tinyint | 1 | 评论状态 |

（3）文章表管理博客发表的文章，表结构设计如表4-3所示。

**表 4-3 文章表**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名称** | **数据类型** | **字段长度** | **说明** |
| id | int | 5 | 文章ID(主键) |
| type | int | 20 | 文章类型 |
| title | varchar | 20 | 文章标题 |
| content | mediumtext | 255 | 文章内容 |
| author\_id | int | 5 | 文章作者id |
| status | tinyint | 1 | 文章状态 |
| views | int | 20 | 文章浏览量 |
| Comments | int | 20 | 文章评论数 |
| top | tinyint | 1 | 文章是否置顶 |
| createtime | int | 20 | 文章创建时间 |
| updatetime | int | 20 | 文章更新时间 |
| dest | varchar | 100 | 对文章的简要描述 |

（5）角色表设置了不同角色对应的不同权限，表结构设计如表4-4所示。

**表 4-4 角色表**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名称** | **数据类型** | **字段长度** | **说明** |
| id | int | 5 | 角色ID(主键) |
| name | varchar | 20 | 角色名称 |
| dest | varchar | 20 | 角色描述 |
| permission\_list | varchar | 255 | 权限列表 |

（4）文章分类表管理文章分类的具体名称，表结构设计如表4-5所示。

**表 4-5 文章分类表**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名称** | **数据类型** | **字段长度** | **说明** |
| id | int | 5 | 分类ID(主键) |
| name | varchar | 20 | 分类名称 |

# **第五章 系统详细设计**

## 5.1注册功能的实现

将注册作为一个公共的组件独立出来，成为一个.vue文件，用户通过填写简单的信息以及设置登陆密码即可完成注册。

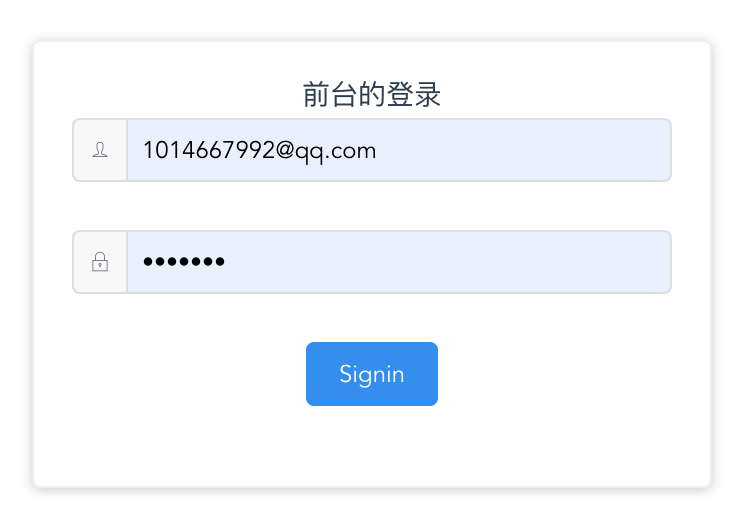
实现方法：

页面输入：前台用户输入完成提交表单，以post方式发送请求。

后端接口：后台接收到post请求，解析出账号与密码。首先将用户名与数据库中的登录账号存在的数据进行对比。若不存在就向用户表里添加一条数据，若账号存在则返回错误信息“账号已存在”。

## 5.2 **登陆功能的具体实现**

实现方法：登陆分为前台的登陆与后台的登陆，两个登陆是公用一个组件，通过传入参数的不同去执行不同的登陆方法跳转不同的页面。前台登录成功跳转到博客前台首页，显示登陆状态，可以对文章进行评论，登录界面如图5-1所示。



**图 5-1 前台登陆**

后台管理系统的登陆同前台登陆逻辑相同只是最后跳转的页面不同，后台管理系统登录成功进入后台管理系统首页。后台登陆界面如图5-2 所示。



**图 5-2 后台登陆**

实现方法：

页面输入：输入登录名和密码，传入区分前台与后台的参数。在页面里调用不同的登录方法。使用post方法发送请求。

后端接口：解析用户输入的账号和密码，首先查找用户是否在数据库的用户表里，若存在去判断输入的密码是否正确。若密码正确则返回用户信息与成功信息，当密码错误时返回错误信息“密码错误”，当用户不存在时返回错误信息“账号不存在”。

## 5.3 后台管理系统的实现

后台实现对文章的增删查改，对文章列表的增删查改，对用户的查看和封禁以及对评论的查看和封禁。文章管理，分类管理，用户管理，评论管理都通过axios获取数据，然后通过对状态的修改来表示是可用还是不可用，是存在还是删除。

后台管理系统的每一个分类下的大致逻辑都是相同的，现以文章管理和用户管理为例介绍设计思路。

1. 文章管理

①添加

添加与修改的大部分方法一样，只是SQL语句从insert改为了update。

页面输入：以post方式提交文章添加请求。

后端接口：解析post请求参数，对取得的参数进行处理，若参数类型和数量无误，则为文章表添加一条数据，向前端返回成功信息，反之返回对应的错误信息如“参数错误”。

②删除

页面输入：出入文章id，以post方式提交文章添加请求。

后端接口：解析post请求参数，取得传入的id，在表中查询是否有此id，若存在且参数类型和数量无误，则为此文章删除，向前端返回成功信息，反之返回对应的错误信息如“参数错误”。

1. 用户管理

①查看

添加与修改的大部分方法一样，只是SQL语句从insert改为了update。

页面输入：以get方式提交文章添加请求。

后端接口：解析get请求参数，对取得的参数进行处理，若参数类型和数量无误，向前端返回用户列表与成功信息，反之返回对应的错误信息如“参数错误”。

②封禁

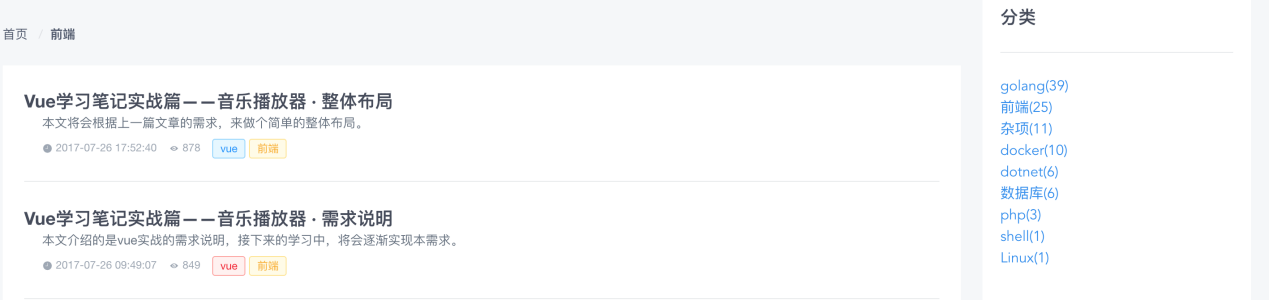
页面输入：出入用户id，以post方式提交文章添加请求。

后端接口：解析post请求参数，取得传入的id，在表中查询是否有此id，若存在且参数类型和数量无误，则为此用户的状态改为不可用，向前端返回成功信息，反之返回对应的错误信息如“参数错误”。

## 5.4 前台文章列表的实现

（1）前台文章列表的实现

通过不同的参数传递从api获得对应参数的文章数据，在前台进行展示。用户浏览文章与文章列表。通过对不同分类的选择改变api请求的参数，这样就可以获取不同分类下的文章。



**图 5-3 文章列表**

实现方法：

页面输入：使用get方法发送获取文章列表的请求。

后端接口：获取参数，当条件里的分类字段为空时，文章列表根据创建[时间](https://zh.wikipedia.org/wiki/%E6%99%82%E9%96%93" \o "时间)，以倒序方式由新到旧排列，当分类字段存在，先以进行分类id正序排，再以文章创建时间倒序排。若参数类型和数量无误，返回文章列表与成功信息，反之返回对应的错误信息如“参数错误”。

（2）前台文章详情的实现

实现方法：

页面输入：传入文章id，使用get方法发送获取文章详情的请求。

后端接口：获取传入的参数文章id，在数据库的文章表里查询id为传入id的数据，若存在且参数类型和数量无误，返回文章全部信息与成功信息，反之返回对应的错误信息如“参数错误”。

（3）前台文章评论的实现

前台登录成功后可以在文章详情页对文章进行评论。

实现方法：

页面输入：使用get方法发送评论请求。

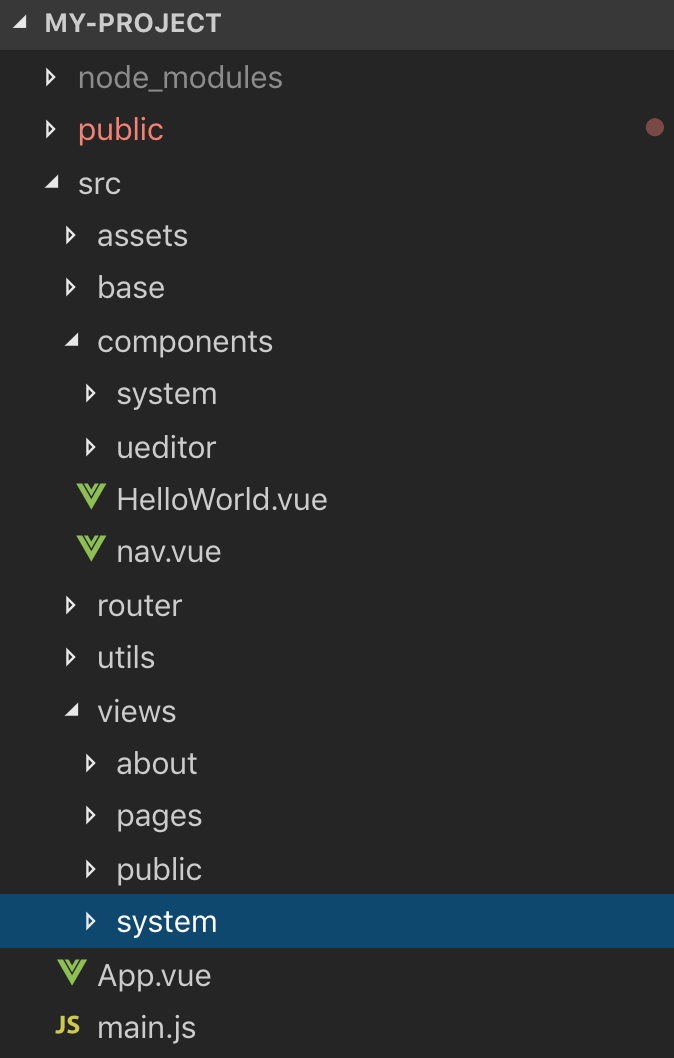
后端接口：获取参数，将参数进行处理。若参数类型和数量无误，则为评论表添加一条数据，向前端返回成功信息，反之返回对应的错误信息如“参数错误”。

## **5.5 组件及路由设计**

组件系统是 Vue 的重要概念，它是一种抽象的概念，此设计允许开发者使用小型的、独立的和通常可复用的组件通过组合去构建大型应用。在WEB网页制作中几乎任意类型的应用界面都可以抽象为一个组件树。

本系统的项目结构如下图所示。所有的页面级的组件都放置在views文件夹下，一个文件夹或文件对应一个功能页面，公共组件或者每个页面所需要的组件放置在components文件夹下。

本系统的路由实现使用vue-router进行路由管理**。**Vue Router 是 [Vue.js](http://cn.vuejs.org/" \t "/Users/fisher_998/Documents\\x/_blank) 官方的路由管理器。路由设计使用了vue-router的history模式与路由嵌套，在router.js中需要将组件 (components) 与路由 (routes)进行映射，然后告诉在页面中子路由通过<router-view></router-view>标签呈现内容和实现路由变化使Vue Router知道在哪里渲染它们。



**图 5-5 项目结构图**

## **5.6公共状态管理**

Vuex是一个专为Vue.js设计的状态管理库，可利用Vue.js的细粒度数据响应机制实现高效的状态更新。本系统使用vuex来进行系统组件之前公共数据的管理。



**图 2-5 vuex状态管理流程**

所有 store 中 state 的改变，都放置在 store 自身的 action 中去管理。这种集中式状态管理能够被更容易地理解哪种类型的 mutation 将会发生，以及它们是如何被触发。当错误出现时，我们现在也会有一个 log 记录 bug 之前发生了什么。

## 5.7 axios封装

在JavaScript中发出HTTP请求的方法有很多，比如：Ajax，jQuery中的方法（$.ajax，$.get，$.post等），axios,Fetch等。本系统选用axios进行数据请求。

在vue项目中，和后台交互获取数据这块，我们通常使用的是axios库。axios是可运行在浏览器端和node.js中的，它是一个基于promise的http库。拥有很多优秀的特性，例如拦截请求和响应、取消请求、转换json、客户端防御XSRF等。

本次使用对axios进行了封装，封装为request.js本质上返回了一个Promise。Promise 是一种对异步操作的封装，是一个保存着未来将要结束的事件的对象，可以通过独立的接口添加在异步操作执行成功、失败时执行的方法。当数据有返回并且返回的状态码是成功时，进入Promise的then中执行相应的操作，或是不是成功或者没有数据返回时，在catch中处理错误的情况。

在系统中经常做的一件事就是对请求或响应进行拦截，希望在then或catch处理之前进行一些通用的操作。axios提供了请求拦截（interceptors.request）和响应拦（interceptors.response）。

当求请发送时在请求拦截中判断当前接口是否需要token的，若需要就添加上token。并在拦截里对get、post请求参数传递的方式进行区分。对于响应拦截主要是对于约定好的错误状态码或者常见的错误码进行统一处理，避免因为抛出异常导致的程序执行错误。

## 5.8 **登陆的路由拦截**

使用vue-router的router.beforeEach可以创建一个全局前置守卫。对于需要登陆的页面在路由定义时添加meta: {requiresAuth: true}项，来标识当前页面需要登陆才可以访问，在vuex里创建isLogined作为已登陆标志。然后在这个钩子里判断当前页面是否需要登陆，即判断是否存在meta的requireAuth标签，若requireAuth标签存在表明当前也是一个需要登陆的页面，若是不存在则无需登陆。然后判断登陆状态即isLogined为true还是为false,若是false则跳转到登陆页面login，若是true代表已登陆则不会执行去往登陆页面的代码。可以通过路由守卫实现简单的权限控制。

# **系统测试**

测试是对系统可靠性的检测，让系统的运行更加稳定，在自测过程中发现系统的问题并及时解决，不影响正常使用。

(1) 测试功能：前台用户登陆功能。

测试用例：进入博客主页，若用户没有登陆，在进入需要登陆的界面或者使用需要登陆才能使用的功能是登陆拦截会起作用，将用户被重定向到博客首页。 ① 填写正确的用户名及密码,用户名：zhy，密码：123456。

② 填写错误的用户名或密码,用户名：zhy，密码：1234567。

测试结果：① 成功登陆。

② 弹窗提示用户名或密码错误。

(2) 测试功能：后台用户登陆功能。

测试用例：点击后台登陆入口或者地址栏输入后台地址“http://localhost/#/system”。

① 填写正确的用户名及密码,用户名：admin，密码：123456a。

② 填写错误的用户名或密码,用户名：admin，密码：123456。

测试结果：① 成功登陆。

② 弹窗提示用户名或密码错误。

(3) 测试功能：前台文章展示功能。

测试用例：进入博客首页点击各个分类下的文章均能功能展示。

测试结果：文章详情展示无误。

(4) 测试功能：后台文章管理的删除功能。

测试用例：点击要删除的文章所在行的删除按扭。

测试结果：成功删除文章。

(5) 测试功能：后台文章管理的添加功能。

测试用例：① 填写正确的内容，不能为空的项全部填写。

② 填写正确的内容，为空的项全部不填写或部分不填写。

测试结果：① 成功添加文章。

② 弹窗提示未填字段为空请重新填写。

(6) 测试功能：后台文章管理的修改功能。

测试用例：① 填写正确的内容，不能为空的项全部填写。

② 填写正确的内容，为空的项全部不填写或部分不填写。

测试结果：① 成功修改文章。

② 弹窗提示未填字段为空请重新填写。

# **结论**

在今天这个时代博客的技术已经趋于成熟。创建一个博客越来越简单与方便，本系统完成了简单的前台浏览与登录功能，在后台进行整个博客的发文与具体管理。截止本文完成时系统的基本功能已经可以使用，但是由于技术与知识的局限，导致系统存在着一些不足。

具体为以下几个方面：

1. 由于使用了vue框架，不利于eso即搜索引擎对网站内容的抓取。但是vue对与这个问题已经有了解决方法，会在后期对系统进行此方面的优化。
2. 博客前台未增加丰富的操作功能，如文章发表，评论删除等。
3. 未加入详细的权限规划与分配，导致用户之间的界限不是很明显，对于用户是否拥有此功能，每次都需要前端页面进行复杂的判断，还有可能存在漏判的情况。

对于系统存在的不足，会在今后系统的不断更新维护中去一一解决。

# **参考文献**

[1]梁灏.Vue.js实战[M].清华大学出版社.2017.

[2]程桂花,沈炜,何松林,张珂杰.Node.js中Express框架路由机制的研究[J].工业控制计算机.2016,29(8):101-102.

[3]方晖, 蔡昭权.基于.NET的博客系统的设计与实现[J]. 惠州学院学报, 2007, 27(3):66-71.

[4]吉晓香, 张国华.基于B/S模式的博客系统[J]. 电脑知识与技术, 2010, 6(11):2561-2562.

[5]刘磊. 基于Web框架的博客管理系统设计与实现[J]. 计算机时代, 2017(5).

[6]奥尔波傅强, 陈宗斌. Node.js入门经典[M]. 人民邮电出版社, 2013.

[7]朱二华.基于Vue.js的Web前端应用研究[J]. 科技与创新, 2017(20):119-121.

[8]麦冬, 陈涛, 梁宗湾.轻量级响应式框架Vue.js应用分析[J]. 信息与电脑(理论版), 2017(7):58-59.

[9]王伶俐,张传国.基于NodeJS+Express框架的轻应用定制平台的设计与实现[J].计算机科学.2017,44(z2):596-599.

[10]聂鑫.前端编程与数据库设计的合理运用[J].信息与电脑(理论版).2011,(2):100.

[11]陈帅,关玉蓉.基于Java Web的奖助学金系统设计与实现[J].科技广场.2017,(3):190-192.

[12]李玉.Vue框架的前端交互性能优化解决方案的研究[D].华中科技大学.2017

[13]邹竞莹.Node.JS博客系统的设计与实现[D].黑龙江大学.2016.

[14]旷志光,纪婷婷,吴小丽.基于Vue.js的后台单页应用管理系统的研究与实现[J].现代计算机.2017,(30):51-55.

[15]邓雯婷.基于Vue.js构建单页面GIS应用的方法研究[J].科技创新与应用.2018，（14）：5-7，10.

[16]王志任.基于Vue.js的开发平台的设计与实现[D].广东工业大学.2018.