河南科技学院

2019 届本科毕业论文（设计）

基于 vue 的个人博客的设计与实现

学生姓名：	周会艳
所在院系：	信息工程学院
所学专业：	计算机科学与技术
导师姓名：	郑颖
完成时间：	2019 年 5 月

基于 vue 的个人博客的设计与实现

摘要

互联网成为当今世界人们对外交流、快速获取和发布信息的最常用也是最便捷的渠道，在人们的日常生活中扮演着不可或缺的作用。博客是互联网进入 web2.0 时出现的正好满足了互联网中人们信息交流与共享的需求。

本文对博客的功能与需求进行了完整分析，设计出一个简单、易用的个人博客系统。为了提高开发效率并进行组件复用，本系统前端使用采用 SPA（单页面应用）思想的 vue.js 进行系统的开发。后端使用 node.js 的 express 框架，以创建符合 RESTful API 设计规范的接口。结构上分为前台与后台两部分。前台实现文章浏览，文章评论。后台管理系统实现了文章的增加、删除与修改，用户的增加、删除与修改，评论的删除与封禁解封以及文章分类的增加、删除与修改等管理。

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 with each other, to quickly acquire and publish information, and plays an indispensable role in people's daily lives. Blogs appear when the Internet enters web 2.0 to meet the needs of people in the Internet for information exchange and sharing.

This article provides a complete analysis of the functions and needs of blogs and designs a simple, easy-to-use personal blogging system. In order to improve development efficiency and reuse components, Front end of the system uses vue.js using the SPA (single page application) idea for system development. The backend uses the express framework of node.js to create interfaces that conform to the RESTful API design specification. The structure is divided into two parts: the foreground and the background. The front desk implements article browsing and article review. The background management system implements the addition, deletion and modification of the article, the addition, deletion and modification of the user, the deletion and ban of the comment, and the addition, deletion and modification of the article classification.

Keywords:Blog,Vue.js,Node.js,API, Vuex

目录

第一章 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。博客是互联网web2.0的产物，是一种新的网络通信方式，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。

（2）不断更新是博客壮大的催化剂。互联网时代信息传递快速，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有前进与退步，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的特性，因此博客的空间界定变得模糊，不再只是一个简单的私有空间。博主与访客之间可以形成互动,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、快速获取和发布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日常生活中的方方面面。当今时代技术更新飞快，作为技术学习者与互联网技术从业者需要不断的汲取新知识走在技术进步的前端，但是如今大部分新技术文档与书籍都是英文，专业名词大都由国外技术人员命名，对于英语能力有限的技术学习者与互联网技术从业者，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网站形式都是博客,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

国际上第一个博客由Swarthmore大学的学生贾斯汀·霍尔创建于1994年。在国内，我国学者孙坚华在1998年即博客技术出现后的第一年开始写与博客相关的文章，此后中国的博客发展才进入了繁荣阶段。

在博客进入发展阶段之前，国内外博客的索引方式都是以内容为主的，即以

一篇一篇的文章作为检索的依据。在2005年，当博客进入发展阶段时，出现了语义化博客的设计概念,建设语义系统就是将博客的搜索元素从整篇文章进行了细分，例如：对文章进行分类，以技术划分，以内容深度划分，以时间划分等。直至目前，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地创新。

第二章 系统相关技术介绍

2.1 SPA

单页应用SPA从字面理解就是只有一个页面的应用。在加载时，会一次性加载这个页面所需要的JavaScript、CSS和HTML代码，以保证页面正常显示与使用。单页应用避免了传统网页构建模型的缺点,如：频繁的刷新页面，交互性容易被页面提交打断，页面的更新都依赖于服务器端的计算，服务器端压力过重。目前单页应用的页面跳转由路由管理思想进行控制，简化了页面制作的复杂程度，使得页面之间的跳转更加简单与方便。虽然说是只有一个页面，但是在实际的系统设计过程中使用组件模块化的思想,将一个大的页面按不同的需求分为了或大或小的多个页面文件，使得部分功能或样式可以在不同页面上进行复用。

2.2 vue.js

2.2.1 介绍

Vue.js是用来构建用户界面的渐进式 JavaScript 的框架 ，与其它种类的大型框架不同的是，它能够自下而上地进行应用。Vue.js的核心库只注重视图层，一方面，这种结构使其不但易于上手，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相结合使用时可以重构一个复杂的SPA。

2.2.2 vue响应式原理

响应式更新页面是vue的特点之一.那么vue是如何实现动态更新的呢？Vue响应式的实现采用了数据劫持配合发布者和订阅者模式的工作模式。Vue将data属性中所有的JavaScript对象都进行转换，基于es5新增的Object.defineProperty为属性添加了一个getter和一个setter。Getter用来劫持data对象的取值操作，setter用来劫持data对象的赋值操作。在此基础上引入发布者和订阅者模式。将data所有的属性均创建为发布者，在getter或者setter时进行发布操作，通知使用到这个属性的订阅者更新数据。此时使用到该属性的组件就会进行更新。Vue 响应式系统的底层细节如图2-1所示。

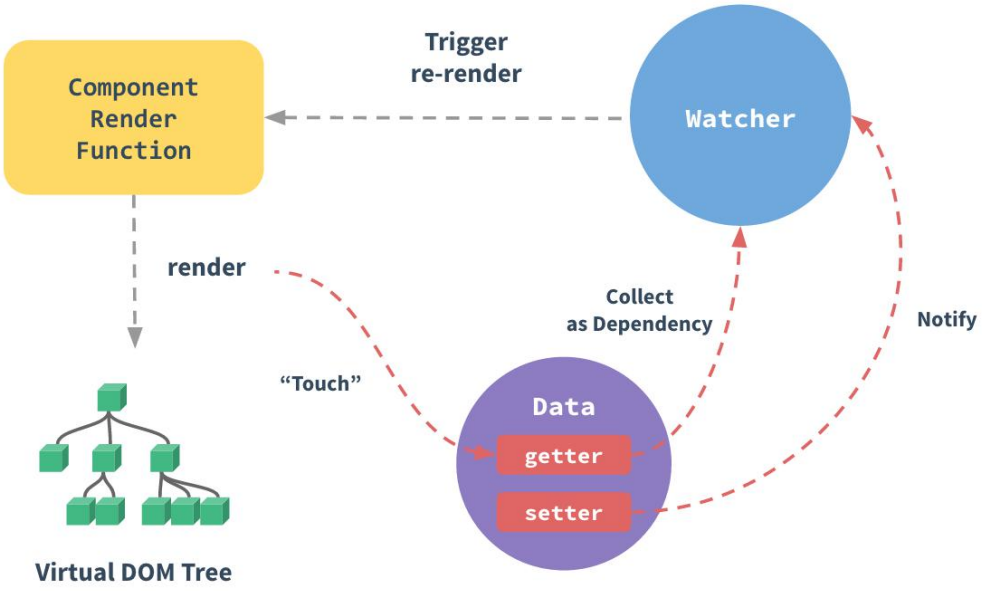

图 2-1 vue 响应式原理

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，让开发者可以只专注于代码与业务本身，而不必忧虑复杂的 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）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）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。

Node.js 的作者 Ryan Dahl 将 V8 引擎进行封装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。使得客户端和服务器端的开发者都可以使用

javascript 语言完成需要的功能。

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 的最大特点之一是单线程。就是在只有一个线程的情况下去实现多个事件的并发处理，采用的是 event loop 事件轮询技术。此外 node.js 采用事件驱动是一个非阻塞 IO 的脚本语言。

2.3.2 Express 框架

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 的思想是要做一个中间件，在工程师的想法和服务器之间作为一个中转层。在最少的干预下让开发者充分表达自己的思想。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）对前台的页面渲染的模版引擎支持。在不同的参数情况下模板可以显示不同的页面状态，实现了页面的动态变化。

（4）性能更好。Express 在 node.js 上扩展了 Web 应用所需的功能。

第三章 系统分析

3.1 可行性分析

可行性分析(Feasibility Analysis)是一个完整项目进行的第一步。进行可行性分析可以在一定程度上减少开发环节、开发复杂度、开发预算（人力、物力与财力）。项目前期的开发预算可以决定项目之后是否可以长久的进行下去。本系统的可行性分析如下所示。

技术可行性分析：本人对于 vue.js 与 node.js 在学习中有所接触，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

明确博客系统的主要任务具体为要干什么要完成那些功能，这些就是需求分析的内容。

本系统需求如下：

可以在页面上展示文章列表，可以查看文章详情，对于不同权限的用户可以有不同操作。后台管理系统需要实现用户管理、评论管理、文章管理与文章分类管理。

个人博客的用户划分为游客、注册、超级管理员。游客仅可以在网站上对文章进行浏览这样的常规访问；注册用户是游客用户通过注册后就是注册用户，注册用户登录博客系统可以对已发表的文章进行评论；超级管理员可以通过个人博客的后台管理系统对用户管理、评论管理、文章管理、文章分类管理。

针对博客系统的以上需求，总结出如下信息：

（1）用户分为未注册游客、注册用户和超级管理用户。游客不计入系统。

（2）只有超级管理员用户拥有足够的权限，可以登录后台管理系统。

（3）博客的超级管理员的可操作功能为：博客的文章类型管理、文章管理、评论管理和用户管理。

（4）普通用户可以阅读文章、发表评论，游客只能进行文章的阅读。

第四章 系统概要设计

个人博客结构上分为前台子系统,后台子系统和数据库。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。

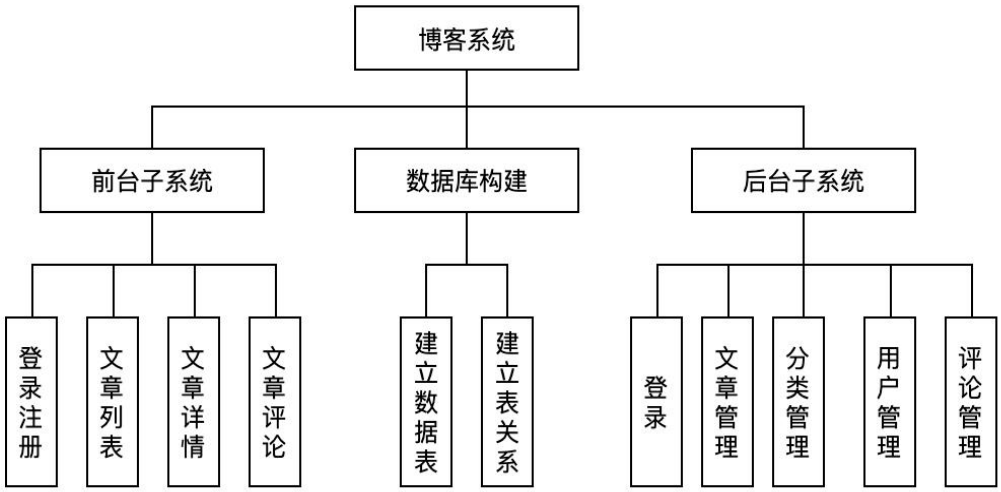

图 4-1 总体规划图

4.1 前台功能设计

博客系统的前台功能设计如下：

（1）游客可以浏览首页，文章详情页，文章列表与文章分类。

（2）游客在有进行文章评论的需求时可以通过注册提高权限。

（2）已注册用户进行登陆后浏览首页，文章详情页，文章列表与文章分类，可以对文章进行评论。

前台功能结构如图 4-2 所示。

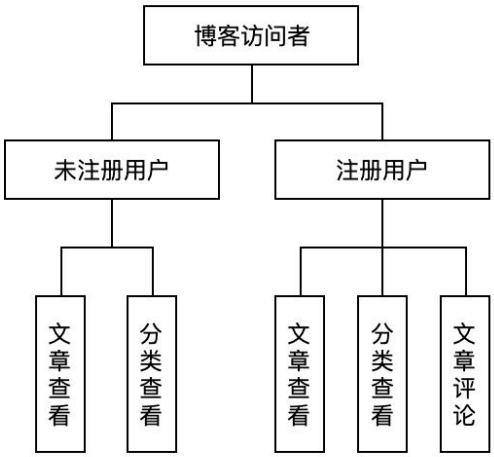

图 4-2 前台功能结构图

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。拥有后台管理系统所有的权限。可以对评论进行查看、解封和封禁；对注册用户进行管理，可进行的操作有添加、修改、查看、解封和封禁，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。

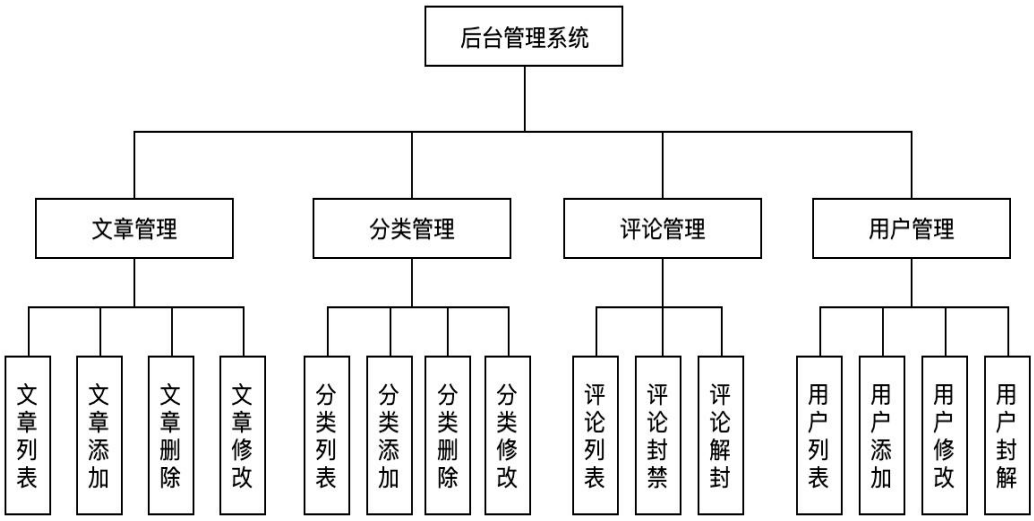

图 4-3 后台功能结构图

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，使用可视化数据库管理工具 navicat for MySQL 进行数据库的实现与管理。

4.3.1 概念结构设计

基于对系统的分析，本系统的数据库由用户表、文章表、评论表、文章分类表等组成。用户与博客系统是多对一的关系，即一个博客系统可以拥有多个用户；文章与评论之间是一对多的关系，一篇文章可以有多条评论；文章与文章分类形成多对一的关系，一个分类下能够有多篇文章，一篇文章属于一个分类。系统数据库的实体关系图如图 4-4 所示。

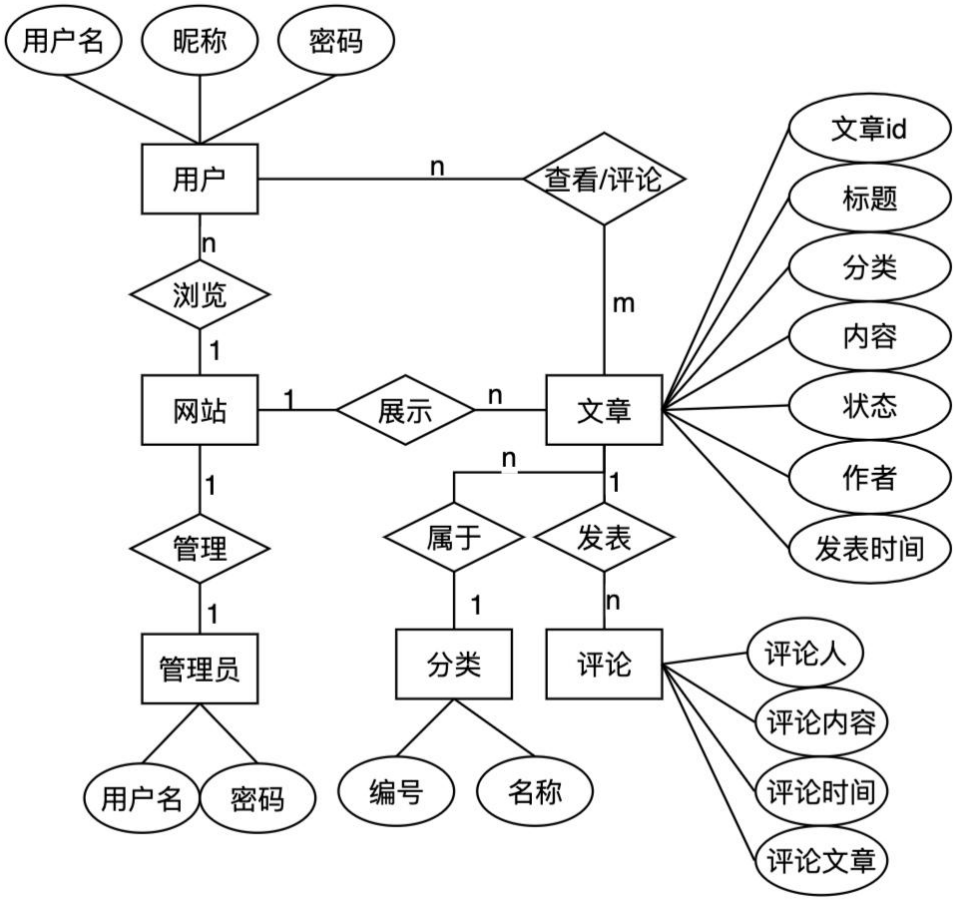

图 4-4 数据库概念结构 E-R 图

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。考虑本系统的数据量较少，所以 id 类字段的长度设置为 5。注册时间类字段设置为 int 型，字段长度为 20 用来存储时间戳。标题类型字段规定字段类型为 varchar 字段长度最大为 50。其他字段各自依据需要进行字段类型与长度的设

置。

（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_id	int	5	评论人 id
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

续表 4-3 文章表

字段名称	数据类型	字段长度	说明
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

数据库各表之间的主键与外键的关系图如图 4-5 所示。

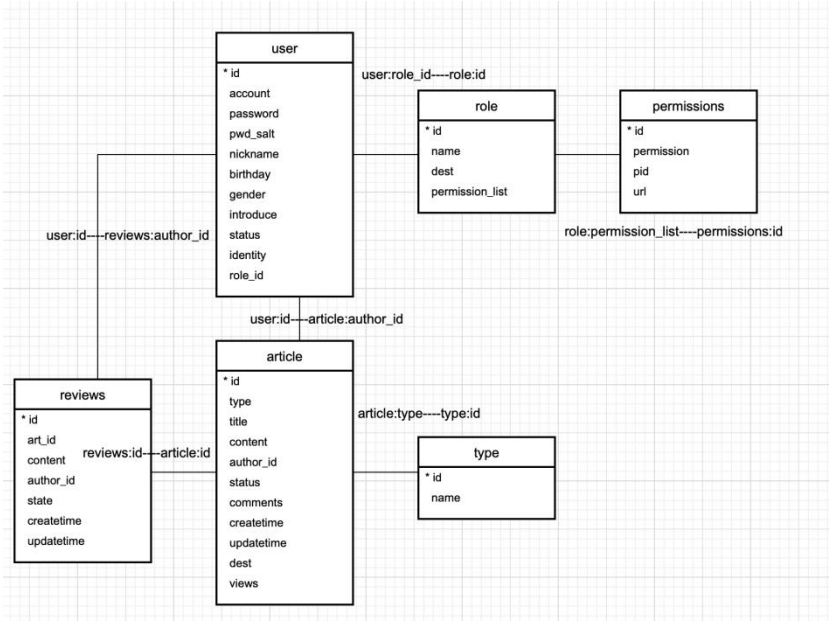

图 4-5 表之间主/外键关系

第五章 系统详细设计

5.1 前台功能的实现

5.1.1 注册功能的实现

将注册作为一个公共的组件独立出来，命名为 register.vue，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：

前台页面：接受用户输入后提交表单，将表单数据以 post 方式发送给后台”/register”接口。

后端接口：后台接收到 post 请求，解析出账号、密码等需要的数据。首先将用户名与数据库 user 表中 account 字段存储的登录账号的数据进行对比。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”。

5.1.2 前台文章列表的实现

通过不同的参数传递从“/art_list”接口获得对应参数的文章数据在前台进行展示。游客用户浏览文章与文章列表。通过对不同分类的选择改变“/type”接口请求的参数，这样就可以获取不同分类下的文章。系统截图如图 5-1 所示。

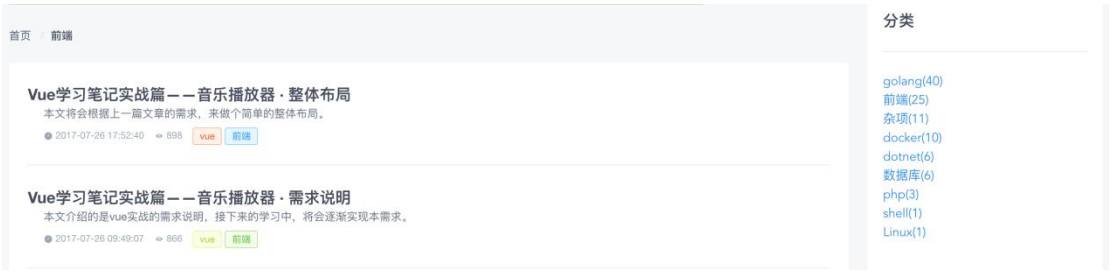

图 5-1 文章列表

实现方法：

页面输入：使用 get 方法向接口“/art_list”发送获取文章列表的请求。

后端接口：获取请求的参数，当条件里的分类字段 type 为空时，文章列表根据创建时间 createtime 以倒序方式由新到旧排列从文章表中返回数据，当分类字段 type 存在，先以 type 值正序排，再以文章创建时间倒序排。若参数类型和数量无误，返回得到的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”。

5.1.3 前台文章详情的实现

实现方法：

页面输入：获取要查看的文章 id，使用 get 方法向“/art_view”接口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”。

5.1.4 前台文章评论的实现

实现方法：

页面输入：使用 get 方法向“/review”接口发送评论请求。

后端接口：获取参数如评论人 author、评论时间 createtime、评论内容 content、评论的文章 ID，将参数进行解析。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,反之返回对应错误信息如“参数错误”。

5.2 后台管理系统的实现

后台管理系统实现对文章与文章列表的增删查改，对用户的添加、修改、查看和封禁以及对评论的查看和封禁。通过 axios 请求获取各个管理模块接口的数据。对于删除与封禁是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、用户管理和评论管理为例介绍设计思路。

5.2.1 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 方式向“/system/art/add”接口提交文章添加请求。

后端接口：从 res.body 中解析 post 请求的参数，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

文章的删除采用状态删除的方式。文章表添加 state 字段，此字段为 0 表示文章正常显示，为 1 表示文章被删除，将不会在前台博客页面中展示。

页面输入：传入文章 id，以 post 方式向“/system/art/edit_state”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,则将文章 id 为传入 id 的那条数据的 state 字段的值更新为 1 表示文章删除，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”。

5.2.2 用户管理

（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方式向“/system/users/add”接口提交文章添加请求。

后端接口：从res.body中解析post请求的参数，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）查看

页面输入：以get方式向“/system/users/list”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，将用户表数据按注册时间createtime降序排列组成用户数据列表，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”。

（3）封禁

用户的删除采用状态删除的方式。用户表的state字段为0表示用户正常显示，为1表示用户被封禁。被封禁的用户将不能成功登陆。

页面输入：传入用户id，以post方式向“/system/user/edit_state”接口提交文章添加请求。

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”。

5.2.3 评论管理

（1）查看

页面输入：以get方式向“/system/review/list”提交评论列表查看请求。

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，将评论表数据按评论时间createtime降序排列组成评论数据列表，向前端返回评论列表与成功信息，反之返回对应的错误信息如“参数错误”。

（2）封禁

评论的删除采用状态删除的方式。评论表的state字段为0表示该条评论正常显示，为1表示该条评论被封禁。被封禁的评论在前台文章评论区中不展示。

页面输入：传入用户id，以post方式向“/system/review/edit_state”接口提交文章添加请求。

后端接口：解析post请求参数，取得传入的id和状态state，在表中查询是否有此id，若存在且参数类型和数量无误，将review表中id为传入id的数

据的 state 的值更新为传入的 state，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”。

5.3 登陆功能的具体实现

登陆分为前台的登陆与后台的登陆，两个登陆使用一个公用组件，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首页。显示为已登陆状态，此时可以对文章进行评论，登录界面如图 5-2 所示。

图 5-2 前台登陆

本系统对登录进行了路由拦截。使用 vue-router 的 router.beforeEach 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。然后对 isLogined 进行判断即登陆状态的判断,若是 false 则跳转到登陆页面 login 进行登录操作，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。

实现方法：

页面输入：提交用户名和密码，密码是经过 2 次 md5 加密后的秘文保证账号的安全。以 post 方法向“/login”接口发送请求。

后端接口：解析出用户输入的用户名和密码，首先查找此用户是否存在于数据库的用户 user 表中，若存则在去判断获得的登录密码是否正确，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”。

5.4 组件及路由设计

组件系统是 Vue 的一种抽象的概念，这种抽象的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。

本系统使用了 vue-cli 自动生成项目。在原有的项目结构上结合自身的需求进行了一些调整，调整后的的项目结构如图 5-3 所示。本项目中所有的页面级的组件都被放置在 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。在原基础上新增 utils 文件件，用来放置用到的公共配置、封装之后的 axios 请求与进行请求的 API 配置文件。

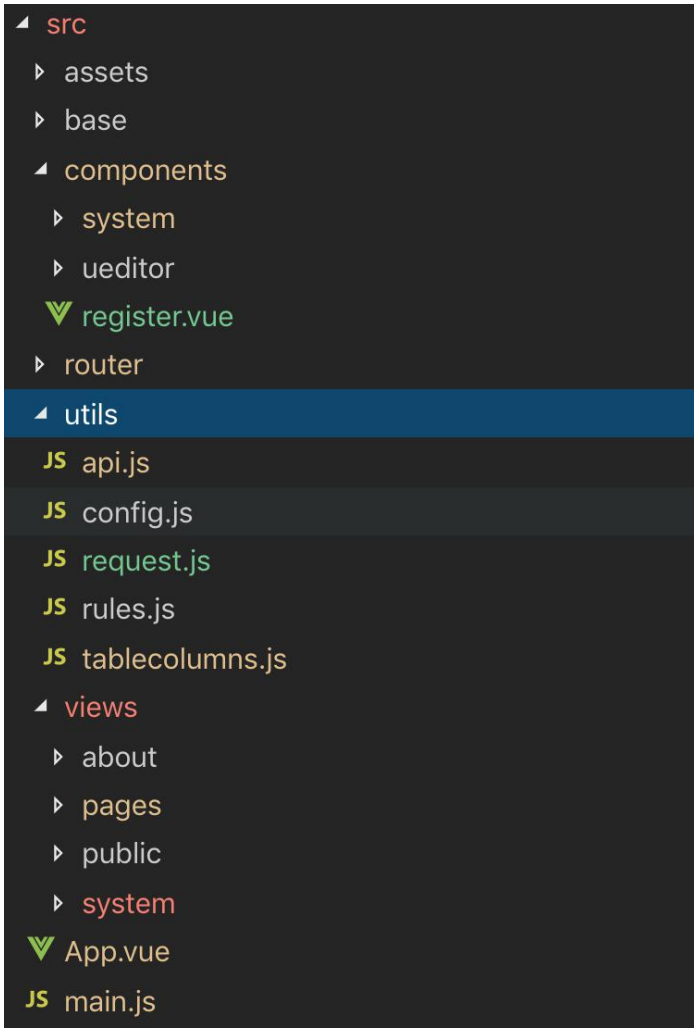

图 5-3 项目结构图

本系统的路由实现使用 vue-router 进行路由管理。Vue Router 是 Vue.js 官方的路由管理工具。系统的路由设计使用了 vue-router 的路由嵌套，在 router.js 中使用 component:组件路径,将组件 (components) 与路由 (routes) 之间的关系进行映射，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。

5.5 公共状态管理

本系统组件之间的公共状态通过 vuex 进行管理。它是 Vue.js 专用的状态管理库。

在 state 中进行公共变量或常量的注册，在组件中可以通过 this.$store.state.xxx 访问要得到的数据。state 的改变必须通过 action，将数据操作代码写在 action 中，组件中需要改变 state 的值时调用 commit 方法就可以调用 action 中的代码。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 出现之前发生了什么。vuex 状态管理流程如图 5-4 所示。

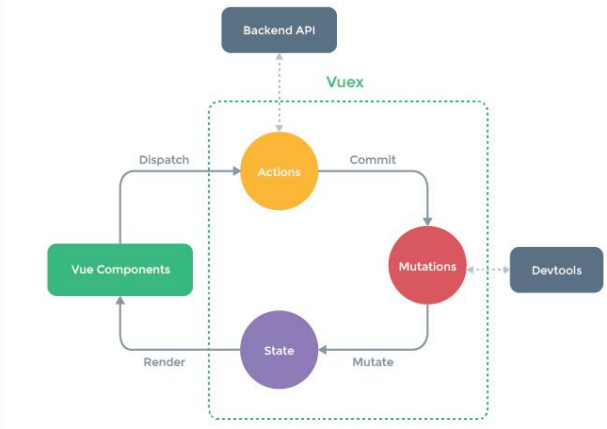

图 5-4　vuex 状态管理流程

5.6 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。

本系统未直接使用 axios 进行接口请求而是对 axios 进行了一层封装，封装为后的文件名为 request.js。axios 异步请求的本质实际上是返回了一个 Promise。它是对异步功能的封装，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。

第六章 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，让系统可以正常使用稳定运行。

(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：zhy123。

② 填写错误的用户名及密码,用户名：zhy，密码：zhy12345。

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：super，密码：123456a。

② 填写错误的用户名或密码,用户名：super，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：前台文章评论功能。

测试用例：进入博客首页点击第一篇文章进入文章详情页，在评论区输入“共同学习，共同进步”。

测试结果：文章详情评论功能无误。

(5) 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。

(6) 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。

(7) 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。

第七章 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,(22):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.

致谢

最后要感谢在整个论文写作过程中帮忙过我的每一个人。大学期间的课程知识在本次毕业设计中都有体现，本设计能够顺利的完成，也归功于各位任课老师的认真负责，使我能够很好的掌握和运用专业知识，是我可以在用到这门技术时可以用的出来。正是有了他们的悉心帮忙和支持，才使我的毕业论文工作顺利完成，在此向我的授课老师表示由衷的谢意。其中主要感谢的是我的指导老师，郑颖老师。在选题、系统设计与论文修改时都给了我很大帮助，帮助我确立了题目，在后续的时间里对于我的问题都给我了很多指导。最后也非常感谢在系统设计过程中给予我帮助的朋友与同学，你们是构成我大学生活的不可或缺的重要的部分。