**广州大学本科毕业设计**

知识分享网站的设计与实现

**Design and Implementation of Konwledge Sharing Website**

**刘耀惠**

**广州大学**

**2019年4月**

**本科毕业设计**

知识分享网站的设计与实现

|  |  |  |
| --- | --- | --- |
| 姓名 | ： | 刘耀惠 |
| 导 师 | ： | 赵波 |
| 专业 | ： | 电子商务 |
| 系别 | ： | 物流与电子商务系 |
| 学院 | ： | 工商管理学院 |

**Graduation Design for Undergraduates**

Design and Implementation of Knowledge Sharing Website

|  |  |
| --- | --- |
| Candidate： | Liu Yaohui |
| Supervisor： | Zhao Bo |
| Major： | E-Commerce |
| Speciality： | Logistics and E-commerce |
| Affiliation： | School of Business Administration |

摘 要

随着互联网的发展，凭借互联网本身具备的获取资源效率高和获取成本低的优势，以及应对激烈竞争的知识技能的需要，互联网知识分享平台，以及知识内容付费逐渐兴起。

本文设计与实现了一个前后端分离的知识分享网站。前端应用程序基于MVVM模型进行架构，以Vue作为前端应用程序MVVM框架的核心实现部分；服务器端程序基于MVC模型进行架构，以Koa作为服务器应用程序MVC框架的核心实现部分。

系统的设计过程如下：

（1）分析知识分享平台背景和现状

（2）系统开发环境配置

（3）系统功能需求分析和系统总体设计

（4）系统的设计与实现

网站实现的主要功能有获取推荐资源、分类资源、搜索资源、标签资源，个人用户账户管理，查看个人信息，查看具体资源，资源评论，资源点赞，资源收藏等功能。

关键词：知识分享；Vue；MVVM；Koa；MVC

Abstract

With the development of the Internet, with the advantages of high efficiency and low cost of acquiring resources, as well as the need to cope with the fierce competition of knowledge and skills, the Internet knowledge sharing platform and the payment of knowledge content are gradually rising.

This paper designs and implements a front-end and back-end knowledge sharing website. Front-end application is based on MVVM model, with Vue as the core implementation part of MVVM framework of front-end application; server-side application is based on MVC model, and Koa as the core implementation part of MVC framework of server application.

The design process of the system is as follows:

(1) Analyzing the background and current situation of knowledge sharing platform

(2) System Development Environment Configuration

(3) System Functional Requirement Analysis and System Overall Design

(4) Design and implementation of the system

The main functions of the website include obtaining recommendation resources, classifying resources, searching resources, label resources, personal account management, viewing personal information, viewing specific resources, resource reviews, resource comments, resource collection and other functions.

Keywords：Share Knowledge; Vue; MVVM; Node.js; MVC

目录

[第1章 绪论 - 1 -](#_Toc7168350)

[1.1 研究背景 - 1 -](#_Toc7168351)

[1.2 研究现状 - 1 -](#_Toc7168352)

[1.2.1 行业现状分析 - 1 -](#_Toc7168353)

[1.2.1 技术现状分析 - 1 -](#_Toc7168354)

[1.3 研究内容 - 2 -](#_Toc7168355)

[1.4 论文结构 - 3 -](#_Toc7168356)

[1.5 本章小结 - 3 -](#_Toc7168357)

[第2章 相关技术介绍 - 4 -](#_Toc7168358)

[2.1 ECMASript 新特性 - 4 -](#_Toc7168359)

[2.2 Vue技术栈 - 4 -](#_Toc7168360)

[2.3 Node 技术栈 - 5 -](#_Toc7168361)

[2.4 MongoDB数据库 - 5 -](#_Toc7168362)

[2.5 本章小结 - 6 -](#_Toc7168363)

[第3章 系统开发环境 - 7 -](#_Toc7168364)

[3.1 系统开发环境准备 - 7 -](#_Toc7168365)

[3.1.1 下载node - 7 -](#_Toc7168366)

[3.1.2 下载 yarn - 7 -](#_Toc7168367)

[3.1.3 前端应用程序的开发环境准备 - 7 -](#_Toc7168368)

[3.1.4 后端应用程序的开发环境准备 - 8 -](#_Toc7168369)

[3.2 系统开发辅助工具安装 - 10 -](#_Toc7168370)

[3.3 本章小结 - 10 -](#_Toc7168371)

[第4章 系统分析 - 11 -](#_Toc7168372)

[4.1 系统功能需求分析 - 11 -](#_Toc7168373)

[4.2 系统总体设计 - 20 -](#_Toc7168374)

[4.2.1 Node服务器应用程序的整体架构 - 21 -](#_Toc7168375)

[4.2.2 Vue前端应用程序的整体架构 - 22 -](#_Toc7168376)

[4.3 本章小结 - 23 -](#_Toc7168377)

[第5章 系统的设计与实现 - 24 -](#_Toc7168378)

[5.1 Node服务器应用程序的设计与实现 - 24 -](#_Toc7168379)

[5.1.1 目录结构及各文件功能设计 - 24 -](#_Toc7168380)

[5.1.2 数据库模型设计 - 26 -](#_Toc7168381)

[5.1.3 Node服务端路由设计 - 28 -](#_Toc7168382)

[5.1.4各路由功能的实现 - 31 -](#_Toc7168383)

[5.2 Vue前端应用程序的设计与实现 - 43 -](#_Toc7168384)

[5.2.1目录结构及各文件功能设计 - 43 -](#_Toc7168385)

[5.2.1前端路由与组件渲染关系设计 - 45 -](#_Toc7168386)

[5.2.3封装axios请求 - 52 -](#_Toc7168387)

[5.2.4 Vuex状态管理 - 53 -](#_Toc7168388)

[5.2.5网站主要界面展示及其功能测试 - 55 -](#_Toc7168389)

[5.2.1网站响应式设计 - 63 -](#_Toc7168390)

[5.3 本章小结 - 64 -](#_Toc7168391)

[结论 - 65 -](#_Toc7168392)

[参考文献 - 66 -](#_Toc7168393)

[致 谢 - 67 -](#_Toc7168394)

Contents

[Chater 1 Introduction - 1 -](#_Toc7168350)

[1.1 Research background - 1 -](#_Toc7168351)

[1.2 Research status - 1 -](#_Toc7168352)

[1.2.1 Analysis of the Current Situation of the Industry - 1 -](#_Toc7168353)

[1.2.1 Analysis of the Current Situation of Technology - 1 -](#_Toc7168354)

[1.3 Research contents - 2 -](#_Toc7168355)

[1.4 Thesis structure - 3 -](#_Toc7168356)

[1.5 Summary - 3 -](#_Toc7168357)

[Chater 2 Introduction of related technologies - 4 -](#_Toc7168358)

[2.1 New features of ECMASript - 4 -](#_Toc7168359)

[2.2 Technology stack of Vue - 4 -](#_Toc7168360)

[2.3 Technology stack of Node - 5 -](#_Toc7168361)

[2.4 MongoDB database - 5 -](#_Toc7168362)

[2.5 Summary - 6 -](#_Toc7168363)

[Chater 3 System Development Environment - 7 -](#_Toc7168364)

[3.1 System development environment preparation - 7 -](#_Toc7168365)

[3.1.1 Download node - 7 -](#_Toc7168366)

[3.1.2 Download yarn - 7 -](#_Toc7168367)

[3.1.3 Development environment preparation for front-end applications - 7 -](#_Toc7168368)

[3.1.4 Backend application development environment preparation - 8 -](#_Toc7168369)

[3.2 System Development Assistant Tool Installation - 10 -](#_Toc7168370)

[3.3 Summary - 10 -](#_Toc7168371)

[Chater 4 Systems analysis - 11 -](#_Toc7168372)

[4.1 Analysis of System Functional Requirements - 11 -](#_Toc7168373)

[4.2 System Overall Design - 20 -](#_Toc7168374)

[4.2.1 Overall Architecture of Node Server Applications - 21 -](#_Toc7168375)

[4.2.2 Overall architecture of Vue front-end applications - 22 -](#_Toc7168376)

[4.3 Summary - 23 -](#_Toc7168377)

[Chater 5 Design and Implementation of the System - 24 -](#_Toc7168378)

[5.1 Design and Implementation of Node Server Application Program - 24 -](#_Toc7168379)

[5.1.1 Directory structure and function design of each file - 24 -](#_Toc7168380)

[5.1.2 Design of database model - 26 -](#_Toc7168381)

[5.1.3 Node Server Routing Design - 28 -](#_Toc7168382)

[5.1.4 Implementation of Routing Functions - 31 -](#_Toc7168383)

[5.2 Design and Implementation of Vue Front-end Application Program - 43 -](#_Toc7168384)

[5.2.1 Design of Front-end Routing and Component Rendering - 43 -](#_Toc7168385)

[5.2.1 Design of Front-end Routing and Component Rendering - 45 -](#_Toc7168386)

[5.2.3 Package request of axios - 52 -](#_Toc7168387)

[5.2.4 Vuex state management - 53 -](#_Toc7168388)

[5.2.5 Main Interface Display and Functional Testing of Website - 55 -](#_Toc7168389)

[5.2.1 Responsive Design of Website - 63 -](#_Toc7168390)

[5.3 Summary - 64 -](#_Toc7168391)

[Conclusion - 65 -](#_Toc7168392)

[Reference - 66 -](#_Toc7168393)

[Acknowledgements - 67 -](#_Toc7168394)

# 第1章 绪论

## 1.1 研究背景

从1994年实现与互联网的第一次全功能连接开始[1]，到如今，中国的网民人数达到8.29亿，互联网普及率达到59.6%，中国互联网取得了举世瞩目的发展成就[2]，随着信息化的高速发展与网络的大众普及，越来越多的网络应用出现，如网络社交，网络游戏，网络购物，网络娱乐等等。

随着越来越多的人参与网络活动中，大规模的历史数据从中产生，大众逐渐发现，利用网络进行知识的检索，编辑与分享，与传统途径比较，具有高效率以及低获取成本的相对优势，兼具现代社会竞争日趋激烈，保持竞争优势，就必须不断补充学习知识技能，所以知识技能分享平台也应运而生。

近年来，共享经济取得了长足发展，截至2018年，知识技能的共享经济市场交易规模位居前三，年度增长率达到70.3%[3]，与众多共享经济发展领域比较，仍然保持强劲发展势头。

## 1.2 研究现状

### 1.2.1 行业现状分析

网络时代之前，我们获取知识的途径极其有限，如口传身授，生活实践，书本等方式。

在互联网初期，获取知识主要通过搜索引擎或者门户网站。随着网络历史数据量的增大，以及获取知识技能的焦虑感，人们开始对专业性的知识分享平台提出了需求。

知识分享平台是具有知识查询，管理，分享，即时通讯等于功能一体的网络平台[4]，其最突出的点在于知识的分享。2018年，根据统计，知识付费用户规模达到了2.9亿人次，增长速度为32.7%[5]，较上年有所回落，仍然保持强劲势头。

自2016年，知识分享平台进入井喷时期，喜玛拉雅，得到，知乎，分答，豆瓣等平台多足鼎立，其中还有众多的中小自媒体或者个人平台[6]。这些平台往往具有独特的平台定位，如喜玛拉雅的平台定位有声知识分享，得到的平台定位是知识服务，知乎的平台定位是网络问答，分答的平台定位是付费问答[7]，还有一些专业性更强的知识分享平台，如软件开发相关的StackOverflow，掘金，思否等平台，文章分享的简书平台，影评分享的豆瓣平台等等。

### 1.2.1 技术现状分析

构建网站的技术体系在知识分享平台大量出现之前就已经存在，目前知识分享平台的构建架构选择可能是：

（1）传统MVC架构。Web时代初期，没有前后端分离的概念。整个开发流程是根据需求进行系统原型设计，UI设计师根据设计原型得到设计图，前端使用JavaScript + Html + Css 编写页面，与此同时，后端负责使用Php或者Java等服务器端语言进行数据库操作，数据连接，业务处理，编写数据接口等工作；等待前端完成工作之后，后端还需要根据服务接口将Html页面改造为Jsp或者Php页面，以此实现后端接口的连接，至此一个网站才算基本完成。这种情况，很容易出现的问题是，如果网站出现了故障，因为模块的耦合度高，故障排查需要较多的时间。

（2）Ajax是前后端分离的重要技术，Ajax 通俗地讲是异步的无刷新的http请求[8]。Ajax 的出现极大提高了用户的体验，利用Ajax，用户发起异步请求，服务器根据请求响应数据，客户端根据响应数据重新渲染页面的过程是一个无刷新页面的过程。Ajax 术语中有XML的关键术语(XML是一种数据格式)，但现行网站大部分都是用JSON 作为Ajax返回响应数据的基本格式。由于Ajax的出现，后端终于可以摆脱对HTML文档进行Jsp页面改造或者Php页面改造，前端只需要在Html 文档中根据服务端应用程序的接口发起Ajax请求，然后接收服务器应用程序根据请求返回的响应数据并进行渲染，以此达到前端应用程序与服务器应用程序的初步分离。

（3）前端MVVM框架实现前后端完全分离，MVVM框架是将服务器端MVC的View层次分离出来，由前端框架实现页面的渲染。前端框架，如Vue，React，Angular ，可以将JavaScript的DOM操作交给框架来完成，开发者本身只需关注业务逻辑和数据操作，通过获取服务器应用程序响应的数据，将数据处理反馈到用户界面，或者用户界面操作产生数据，处理后请求服务器应用程序接口，即实现数据的双向绑定。

（4）node.js，它的出现对前后端协作的影响是巨大的，node.js 是一门服务器端语言，而且简易操作，这意味着，前端使用JavaScript，后端也能使用JavaScript，极大方便了开发者的沟通与交流。

## 1.3 研究内容

本论文设计一个知识分享网站，网站的宗旨是"分享更多，学习更多”，目的是建造一个知识分享社区，完成基本的知识资源的搜索，发布，修改，收藏，点赞和评论以及网站基本用户账户管理，个人信息查看以及修改，意在打造一个知识资源丰富，资源访问可靠，学习氛围浓厚的知识分享社区。本网站以Vue + Node.js 为基础进行构建，是一个前后端分离的项目，前端使用Vue技术栈进行开发前端应用程序，后端使用node的框架koa开发服务器应用程序，数据库使用mongoDB，使用mongoose在koa程序中管理操作mongoDB。

## 1.4 论文结构

本文共分为六个部分，具体是：

第一章是绪论部分，介绍本文所涉及系统的研究背景，知识行业的发展以及相关网站构建技术选择，并对本文需要研究开发的系统进行简要进行阐述。

第二章是相关技术部分，主要介绍ECMAScript 的新特性，Vue的技术栈，Node的技术栈和MongoDB数据库。

第三章是系统开发环境部分，主要介绍项目开发的环境准备。

第四章是系统分析部分，主要完成对项目需求的分析以及对项目的整体架构。

第五章是系统设计与实现部分，主要将项目分为前端和后端进行项目的设计。

第六部分是是总结与相关的致谢内容。

## 1.5 本章小结

本章介绍了项目开发的研究背景，知识行业的研究现状，以及网站构建技术的选择，在此基础上对本论文的研究内容以及论文结构进行介绍

# 第2章 相关技术介绍

## 2.1 ECMASript 新特性

本系统将分为前端和后端两部分进行设计。前端使用的是Vue，Vue是使用JavaScript进行数据双向绑定操作的；后端服务器使用Node.js, 也是使用JavaScript语言进行服务器应用程序的编写，所以有必要对JavaScript进行介绍。

JavaSript的官方标准版本为ECMAScript，2015年6月，ECMAScript 6.0 (以下简称ES6) 的发布，带来了更为丰富的语法和特性支持。本文所设计的系统使用的JavaScript 语法版本为ES6或以上版本。在系统设计过程中，会大量使用以下ES6新特性[9]：

(1) let 和const 的变量声明命名。let 为JavaScript 增加了块级作用域的操作；const 为JavaScript 提供了常量的操作。

(2) 对象的解构赋值。由于JavaScript 是面向对象的程序设计语言，对象可以在函数之间传递，为了方便可以直接获取到对象里面的属性，需要使用对象的解构语法。

(3) 模板字符串。字符串与变量的连接，使用模板字符串语法会更加方便。

(4) 箭头函数。箭头函数具有书写方便，保持外部this的引用的特点，所以本文的设计项目会大量使用箭头函数代替传统函数声明。

(5) 对象方法（函数）的简写。ES6的对象方法（函数）语法更加简洁。

(6) 对象的计算属性。通过对象的计算属性，可以动态取得对象的计算属性名。

(7) async 函数语法的使用。ES2017 引入了async 函数，async 函数语法是Promise, generator 的语法糖，利用async 函数，能够更简单地完成异步操作，避免传统回调地狱，或者繁琐的promise then 的链式调用操作。

(8) 模块语法。使用ES6 提供的模块导入和导出语法，更加方便对项目进行项目的模块化。

(9) 其它ES6新语法特性不常在奔系统设计中使用，故不一一在此阐述。

## 2.2 Vue技术栈

本文的项目将使用Vue的技术栈进行前端应用程序的构建，主要包括以下部分：

(1) Vue。Vue是一个用于构建用户界面的渐进式框架[10]。本文将使用Vue构建前端用户界面，借助Vue，可以不直接操作DOM（Document Object Model），而是对数据进行操作，Vue可以对数据进行响应式绑定，只要操作数据，数据的变化就能响应到页面中。反之，用户界面的操作也能触发数据的更改，Vue可以根据用户的操作，然后对更改，此过程有可能会向服务器应用程序发送异步请求以及接受响应数据。

(2) Vue router。Vue 不像Angular 提供前端路由的功能，需要借助Vue router 实现前端路由功能。Vue router 是Vue.js 的路由管理器，利用Vue router，可以实现更好的路由组件渲染，实现一个单页应用的构建。Vue router 为路由提供动态路由匹配，嵌套子路由，编程式的导航，重定向以及路由别名以及路由元信息的支持[11]。

(3) Vuex。Vue数据传递方式是通过动态属性向子组件props传值，或者子组件自定义事件传递数据给父组件，Vue缺乏一个全局的状态管理对象，而Vuex是专门用于Vue的应用程序的状态管理[12]。本项目使用Vuex 统一管理Ajax发送的异步请求返回的数据，使得数据可以被Vue应用程序的所有组件请求使用。

(4) axios，axios是基于promise的浏览器和node的http请求的库，用于发送Ajax请求。本项目不使用原生的XMLHttpRequest，而且需要对axios进行再次封装，利用封装后的axios发起异步请求。

(5) bootstrap.vue。 bootstrap.vue是基于bootstrap 样式的vue组件库，可以使用里面封装好的组件，如modal，input，navbar等

(6) Vue-cli。Vue-cli是开发Vue应用程序的脚手架，借助于Vue-cli，可以快速创建一个Vue的原型项目。

## 2.3 Node 技术栈

本文的项目使用Node 的技术栈进行服务器应用程序的开发，主要包括以下部分：

(1) Node.js。Node 是在Chrome浏览器的V8引擎上的JavaScript运行时[13]，使用Node.js，可以快速建立一个服务器应用程序，本文的系统设计不直接使用原生的Node.js, 但是Node.js 的path, crypto, HTTP以及fs模块也会使用到。

(2) Koa 框架。Koa 是基于Node.js 的web 开发框架，Koa 支持async 函数语法是本系统使用它作为Node服务器应用程序框架的重要原因。

## 2.4 MongoDB数据库

本文的项目主要使用MongoDB数据库及其框架Mongoose进行数据库操作：

(1) MongoDB 是一个分布式的数据库存储系统，它将数据存储为一个文档（document），数据结构由键值对（key/value）构成[14]，mongoDB中没有表（table）的概念，取而代之的是collection的概念，mongoDB也没有记录或行（row）的概念，取而代之的是document的概念。

(2) Mongoose 框架。在koa中进行MongoDB的操作，使用的是Mongoose框架，这是一个使用JavaScript语言操作的，用于异步操作MongoDB 的对象模型。

## 2.5 本章小结

本章主要对系统设计中需要用到的相关技术进行介绍。介绍了系统中使用的JavaScript官方版本ECMAScript 6.0 的语法新特性，Vue技术栈设计前端应用程序的主要技术，Node技术栈设计服务器应用程序的主要技术，最后介绍mongoDB数据库及其数据库框架mongoose。

# 第3章 系统开发环境

## 3.1 系统开发环境准备

由于本系统设计采用前后端分离的项目架构，所以，系统开发环境准备分为两个部分进行。

### 3.1.1 下载node

服务器应用程序使用的是node.js，node.js的运行需要Node，所以需要下载node。本项目是使用的node版本为v10.15.3。

### 3.1.2 下载 yarn

yarn 是由facebook 开源的一个快速，稳定，安全的包管理工具[15]。Node 中虽然自带了npm的node包管理工机具，本系统最终选择yarn进行node包的管理，因为经过与npm的对比，yarn具有更好的网络性能，能够进行扁平式安装（即避免重复安装依赖），可以重复引用依赖项，更高的包安装速率，所以本项目使用yarn作为包管理工具。

### 3.1.3 前端应用程序的开发环境准备

以下是本项目的Vue前端应用程序的环境依赖（dependence），见表3-1

表3-1 Vue前端应用程序的环境依赖

|  |  |  |
| --- | --- | --- |
| 程序依赖 | 描述 | 版本号 |
| vue | 构建用户界面 | v2.6.10及以上 |
| vue router | 前端路由处理 | v3.0.3及以上 |
| vuex | Vue应用程序状态管理 | v3.0.1及以上 |
| axios | Vue应用程序发起异步请求 | v0.18.0及以上 |
| node-sass | sass运行环境 | v4.9.0及以上 |
| sass-loader | 将sass转换为css | v7.1.0及以上 |
| bootstrap.vue | Vue组件库 | v2.0.0-rc.18及以上 |

本项目使用Vue-cli脚手架进行Vue前端项目的原型创建，脚手架版本为v3.6.2。使用脚手架，我们只需要进行简单的安装和配置，而不用像webpack一样需要手动配置loader和plugin，既可以进行项目的开发。本系统使用Vue-cli，需要对脚手架进行以下配置：

(1) 配置Babel，用于支持将ES6语法以及Common.js语法转换为浏览器能够识别的JavaScript代码

(2) 配置Router，默认会安装Vue-router模块

(3) 配置Vuex，默认会安装Vuex模块

(4) 安装axios，用于发起Ajax请求

(5) 安装 bootstrap-vue，用于使用基于bootstrap样式的组件库

(6) 安装 node-sass，用于为sass提供运行环境

(7) 安装 sass-loader，用于转换sass-loader

Vue-cli脚手架配置和系统所需依赖包安装完成之后，前端应用程序的开发环境配置即可完成。

### 3.1.4 后端应用程序的开发环境准备

以下是本项目的Node服务器应用程序的环境依赖（dependence），见表3-2

表3-2 Node服务器应用程序的环境依赖

|  |  |  |
| --- | --- | --- |
| 程序依赖 | 描述 | 版本号 |
| koa | node.js框架 | v2.7.0及以上 |
| mongoose | mongoDB数据库 | v5.4.22及以上 |
| nodemailer | 用于发送邮件 | v6.1.0及以上 |
| nodemon | 热加载，用于开发 | v1.18.10及以上 |
| node-sass | sass运行环境 | v4.9.0及以上 |
| validator | 用于验证 | v10.11.0及以上 |
| ejs | 模板引擎，用于渲染页面 | v2.6.1及以上 |
| bcrypt | 用于加密 | v3.0.5及以上 |
| jsonwebtoken | 用于生成与验证token | v8.5.1及以上 |
| gravatar | 用于获得全球用户头像 | v8.5.1及以上 |

Koa框架及其所需依赖包配置完成后，还需要安装和配置相应的koa中间件，见表3-3

表3-3 koa 中间件的配置

|  |  |  |
| --- | --- | --- |
| 程序依赖 | 描述 | 版本号 |
| koa-bodyparser | 用于获取请求body | v3.0.4及以上 |
| koa-body | 用于扩展koa-body功能，用于获取上传的文件 | v4.1.0及以上 |
| koa-jwt | 用于处理token验证 | v3.5.1及以上 |
| koa-router | 用于处理后端路由 | v7.4.0及以上 |
| koa-send | 用于处理文件 | v5.0.0及以上 |
| koa-static | 用于开发静态目录 | v5.0.0及以上 |
| koa-views | 用于设置模板引擎的渲染路径 | v6.2.0及以上 |
| koa2-cors | 用于实现跨域配置 | v2.0.6及以上 |

至此，node服务器应用程序的环境配置完成。

## 3.2 系统开发辅助工具安装

本项目开发过程需要使用两个辅助软件

(1) postman，版本为v7.0.9。借助postman，可以模拟向服务器应用程序发送请求，并在界面返回获取到响应数据，可以更好地分离前端和后端程序，独立测试服务器应用程序能否正常运行。

(2) robo3T，版本为1.2.1。robo3T是mongoDB的可视化管理工具，利用该工具免去了不断地在命令行界面对mongoDB进行命令式管理的重复操作，可以提供管理mongoDB数据库的效率

## 3.3 本章小结

本章主要介绍如何配置系统开发环境，介绍系统开发前的项目环境准备，介绍前端vue应用程序的具体环境配置以及node服务器应用程序的具体环境配置，最后介绍系统设计过程中使用到的用于模拟发送服务器请求的postman以及进行monogoDB的可视化管理robo3T

# 第4章 系统分析

## 4.1 系统功能需求分析

可以将系统的功能描述总结为如图4-1所示

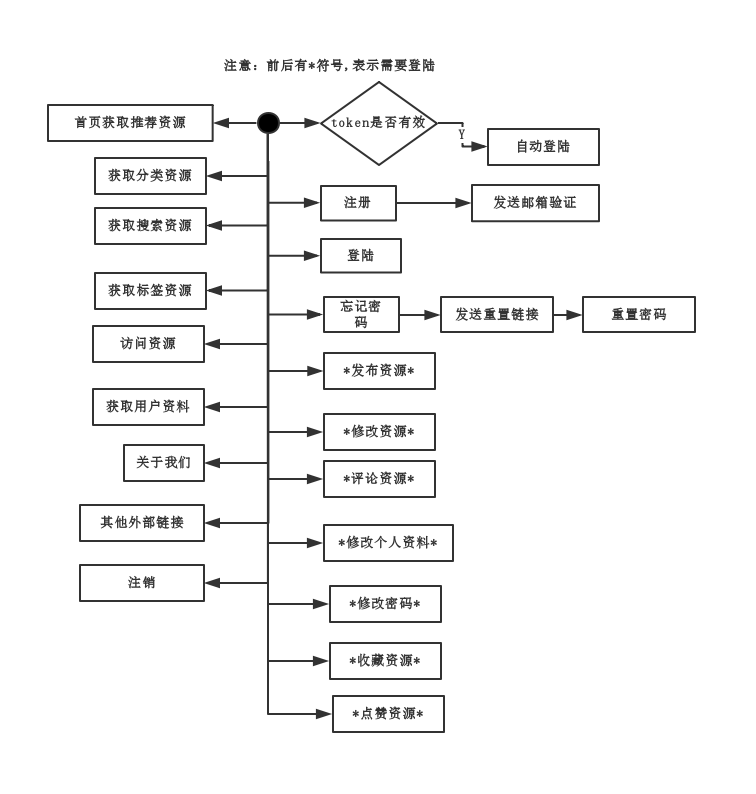


图4-1 系统功能需求分析图

对图4-1的系统功能需求分析可以描述为

（1）token 进行身份认证

本项目不使用传统的cookie/session 身份认证机制，使用token进行验证。相对于传统的身份认证机制，token认证支持跨域访问，而且token保存在客户端，意味着服务器端不需要保存session信息，降低了服务器的内存损耗，token不仅适用于网站认证，也适用于移动应用的认证[16]。所以本项目使用token进行身份认证。如果token验证成功，则自动登录

（2）首页获取推荐资源

每当用户访问首页，前端应用程序都会发送获取资源的请求到服务器应用程序，然后服务器返回响应数据，前端应用程序接受数据并渲染相应的组件或页面。首页推荐资源设置最热与最新的两个排序选择。默认是返回最热的资源，当用户点击最新资源，前端应用程序向服务器发起查找最新推荐资源的请求。过程如图4-2所示

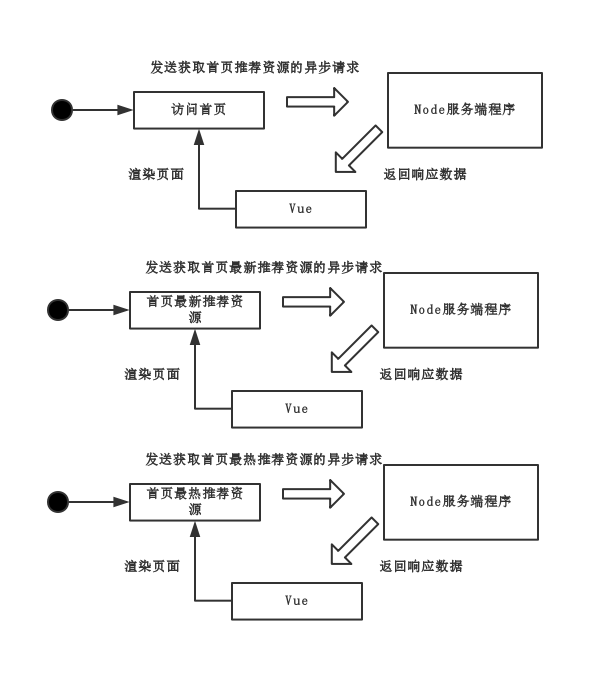


图4-2首页获取推荐资源图

(3)获取分类资源

每当用户点击分类，前端应用程序默认发送获取“计算机科学”分类的资源请求到服务器y应用程序，前端应用程序根据服务器返回的数据渲染页面。在分类页面中，如果用户点击对应的分类，前端应用程序将根据用户相应的点击发送获取对应分类的请求，然后根据返回的数据相应渲染页面。本项目设计的分类项目有：计算机科学、经济、社会科学、历史、人文艺术等，分类资源的获取。过程如下图4-3所示

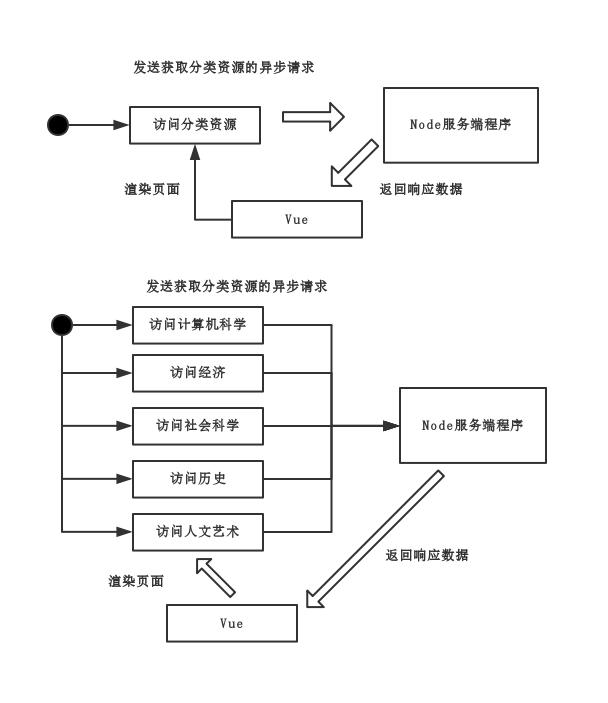


图4-3获取分类资源图

(4)获取搜索资源

每当用户在搜索框输入搜索关键字，用户按下键盘enter或者点击搜索按钮，客户端应用程序向服务器应用程序发起异步请求，服务器应用程序根据搜索关键字返会包含搜索关键字标题的资源，客户端程序根据返回的数据渲染页面。过程如图4-4所示

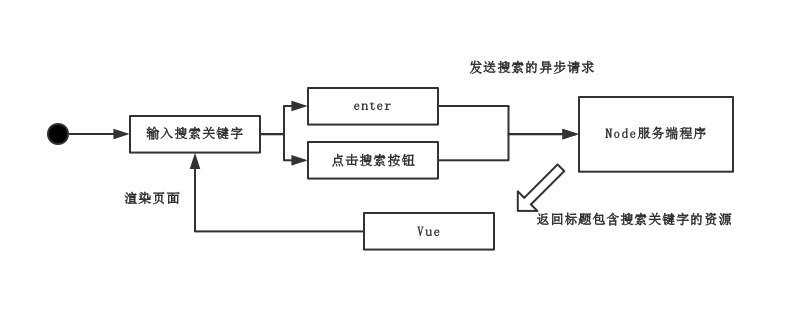


图4-4获取搜索资源图

(5) 获取标签资源

获取标签资源的实质是将比较流行的搜索关键字提取到页面，用户点击该标签，相当于用户以该标签名作为搜索关键字进行资源的搜索。

(6) 注册以及邮箱验证激活

用户点击注册，前端应用程序向服务器应用程序发送异步的注册请求，服务器程序返回响应的数据，如果注册成功，用户需要到注册邮箱进行激活才能正常登录，如果注册失败，则返回相应错误信息。注册应该能对以下错误信息进行处理：

用户名已经存在，邮箱格式不正确，密码长度不少于6位，两次密码不一致

过程如图4-5所示

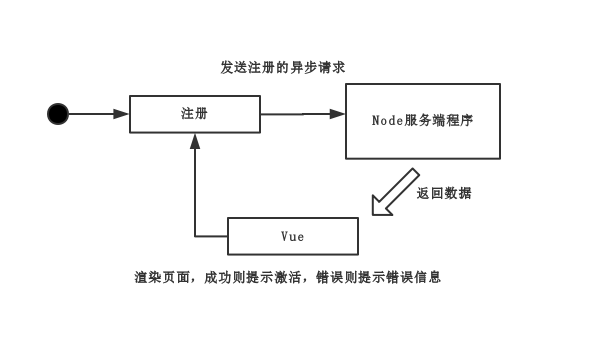


图4-5注册以及邮箱验证激活图

（7）登录

当用户点击登录，前端应用程序向服务器应用长吁发起异步的登录请求，服务端返回响应数据，前端应用程序根据响应数据结果进行处理，如果登录成功，则保存token，切换登录状态，如果错误，则输出错误信息，服务端应用程序应该能够处理以下错误信息：

邮箱格式不正确，密码长度不超过6位，密码不正确

过程如图4-6所示

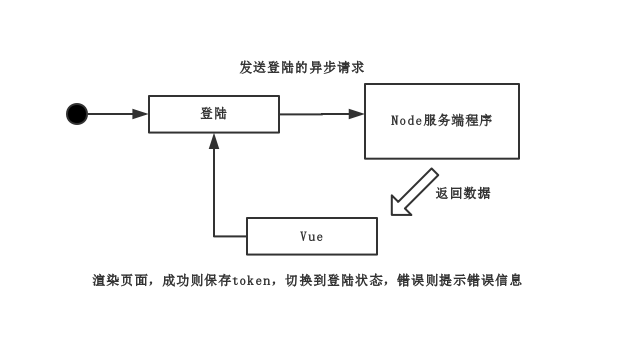


图4-6登录过程图

（8）忘记密码

当用户点击忘记密码，需要用户输入一个邮箱，然后用户点击发送重置密码链接到该邮箱，前端应用程序向服务器应用程序发送忘记密码的异步请求，如果邮箱不存在，服务器程序需要返回“该邮箱未绑定账户”的类似错误信息。如果邮箱存在，服务器应用程序发送一个临时token到指定邮箱上，并返回信息提示用户到相应邮箱去点击重置密码的链接。用户收到一个带有临时token的用于跳转到重置密码页面的链接。重置密码页面中，用户点击密码重置，向服务器应用程序发送重置密码的异步，服务器对临时token进行验证，如果成功，则返回重置密码成功的信息，如果失败，则返回非法权限的相关信息。具体过程如图4-7所示

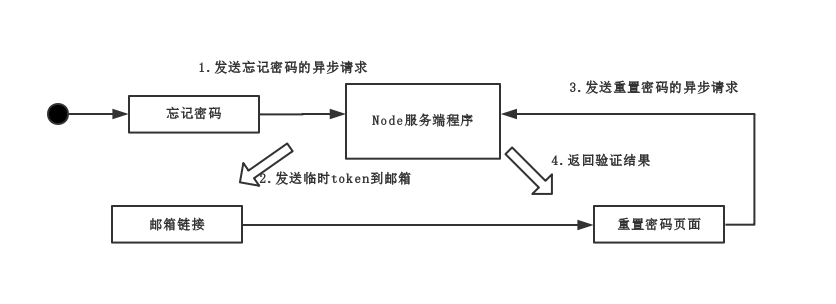


图4-7忘记密码过程图

（9）访问资源

用户点击相应的资源，进入具体的资源界面，向服务器应用程序发起访问资源的异步请求，服务器根据请求返回资源内容或错误信息，前端应用程序根据响应的数据渲染页面。过程如图4-8所示

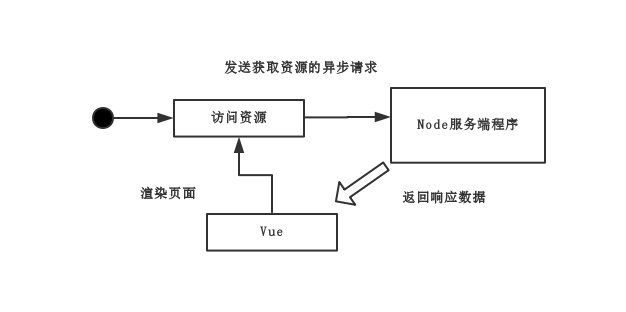


图4-8访问资源过程图

（10）获取用户资料

用户点击个人中心或者其他用户的链接，进入个人或者其他用户的个人中心，向服务器应用程序发起获取用户资料的异步请求，服务器返回响应数据，前端应用程序根据数据进行页面渲染。过程如图4-9所示

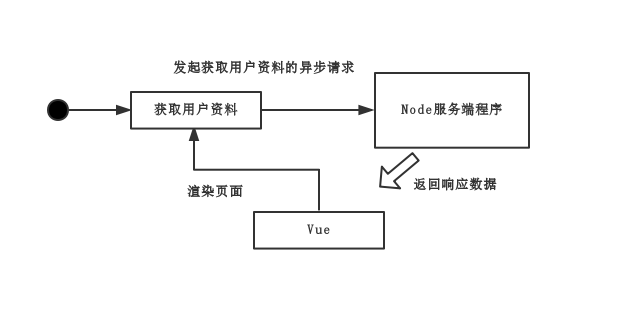


图4-9获取用户资料过程图

（11）关于我们以及其他静态页面

关于我们，404页面等页面属于静态渲染，非响应式的页面

（12）注销

当用户点击注销，不向服务器发送请求，需要改变客户端的登录状态，将localstorage中的token进行清除(本项目将token保存在浏览器的localstorage中)

（13）发布资源

发布资源需要登录作为前提。如果用户点击发布资源，前端应用程序向服务器应用程序发起发布资源的异步请求，服务端应用程序响应数据或者错误信息，前端应用程序根据数据渲染页面。其中资源标题和资源正文不能为空，在前端应用程序进行统一验证。过程如图4-10所示

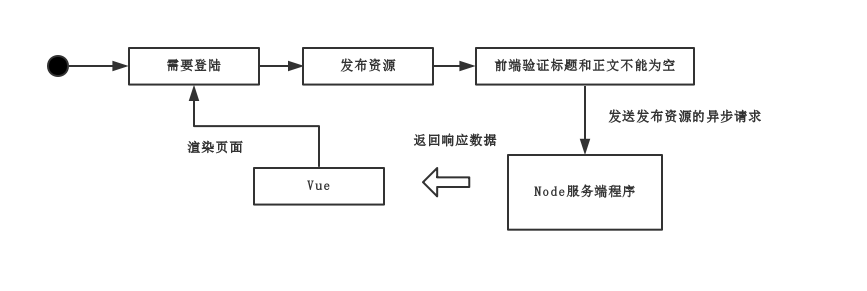


图4-10发布资源过程图

（14）修改资源

修改资源需要登录作为前提，只有作者本人才能看到资源的编辑链接。作者点击资源修改，向服务器应用程序发送修改资源的异步请求，前端应用程序根据数据渲染也main。服务器应用程序应该能够做出一些错误处理，即多种方式验证是否是作者本人进行文章修改。其中资源标题和资源正文不能为空，在前端应用程序进行统一验证。过程如图4-11所示

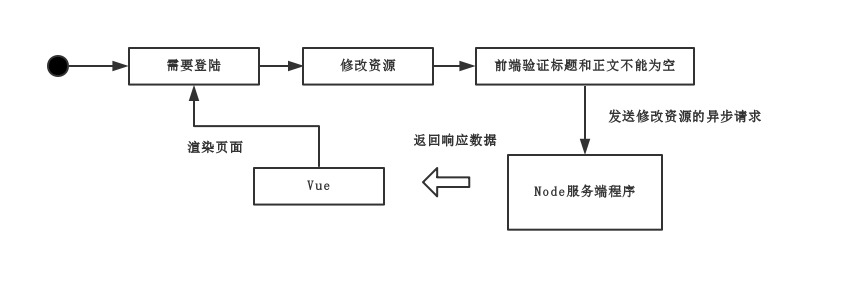
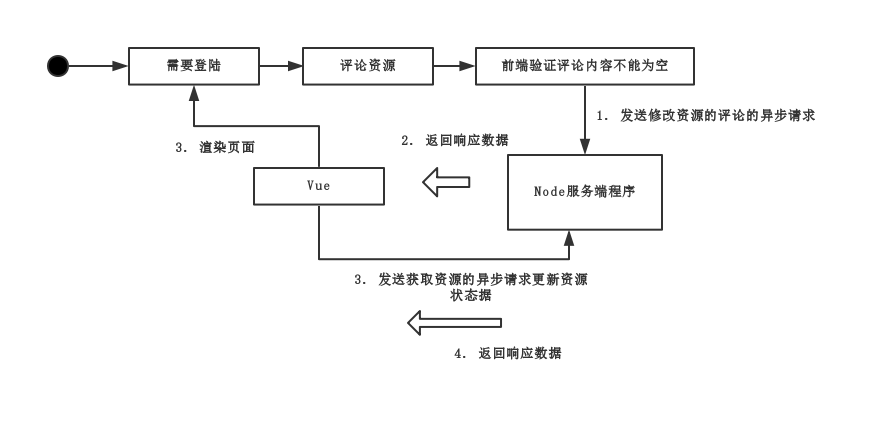


图4-11修改资源过程图

（15）评论资源

评论资源需要登录。用户进行资源评论，前端应用程序向服务器应用程序发起资源评论的异步请求，服务器应用程序响应数据，前端应用程序根据返回的数据渲染页面，如果成功评论资源，应该重新发起资源的请求以更新资源的数据状态。过程如图4-12所示

 图4-12评论资源过程图

（16）收藏资源

收藏资源需要登录。用户进行资源收藏，前端应用程序向服务器应用程序发起收藏资源的异步请求，服务器应用程序响应数据，前端应用程序根据返回的数据渲染页面，如果成功收藏，需要重新发起更新资源的异步请求来更新资源的状态。过程如图4-13所示

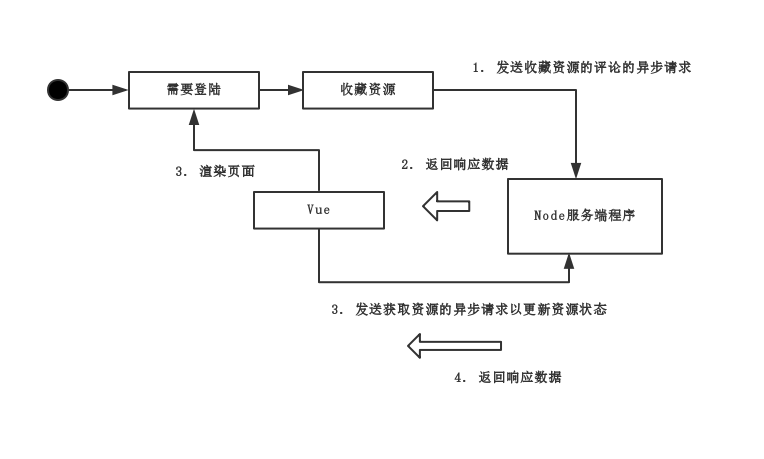


图4-13收藏资源过程图

（17）点赞资源

点赞资源需要登录。用户对资源进行点赞，前端应用程序向服务器应用程序发起点赞资源的异步请求，服务器应用程序响应数据，前端应用程序根据返回的数据渲染页面，如果成功点赞，需要重新发起更新资源的异步请求来更新资源的状态。过程如图4-14所示

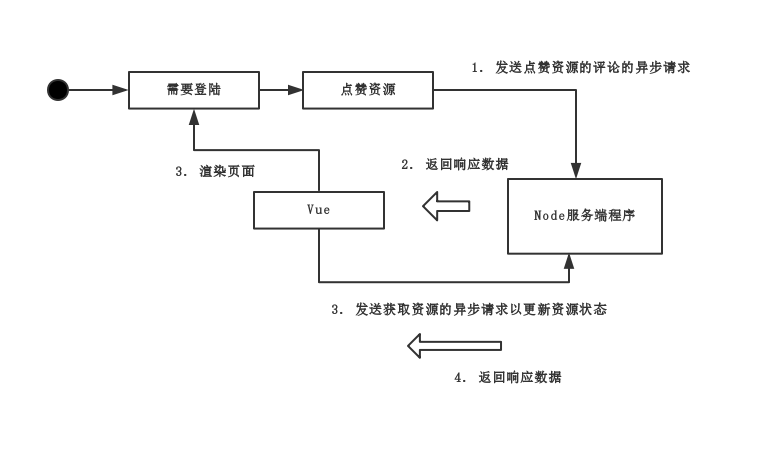


图4-14点赞资源过程图

（18）修改个人资料

修改个人资料需要用户登录。用户点击确认修改按钮，前端应用程序向服务器应用程序发起修改个人资料的异步请求，服务器端响应并返回数据，前端应用程序根据数据进行页面渲染，如果用户需要修改用户名，服务端应用程序应该具有判断用户名是否存在的能力，如果用户名存在，则前端应用程序应该给出提示。过程如图4-15所示

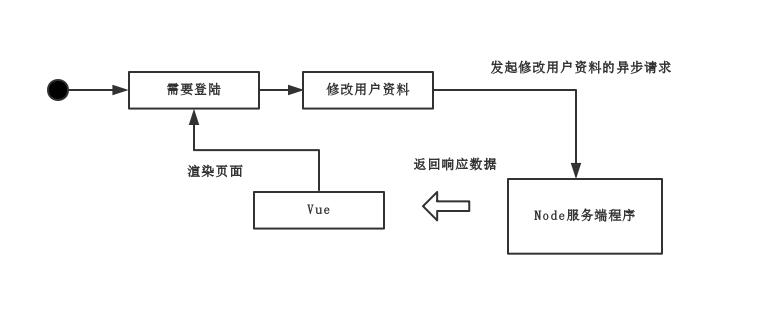


图4-15修改用户资料过程图

（19） 修改用户密码

修改用户密码需要用户登录。用户点击提交修改用户密码按钮，前端应用程序向服务器应用程序发送修改用户密码的异步请求，服务器响应并返回数据，前端应用程序根据数据进行页面渲染。服务器应用程序应该具备检测原密码的正确性，新密码长度是否大于等于6，两次密码是否不一致的错误检测。过程如图4-16所示

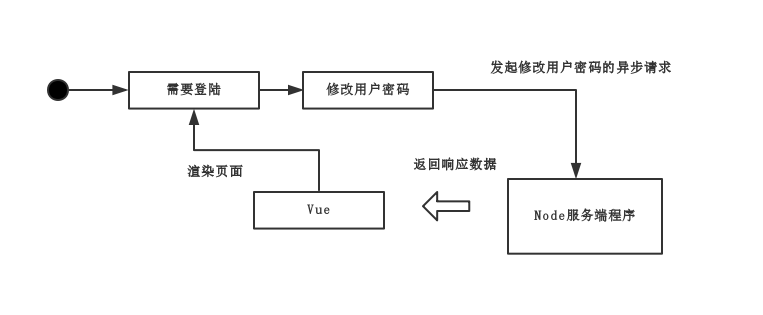
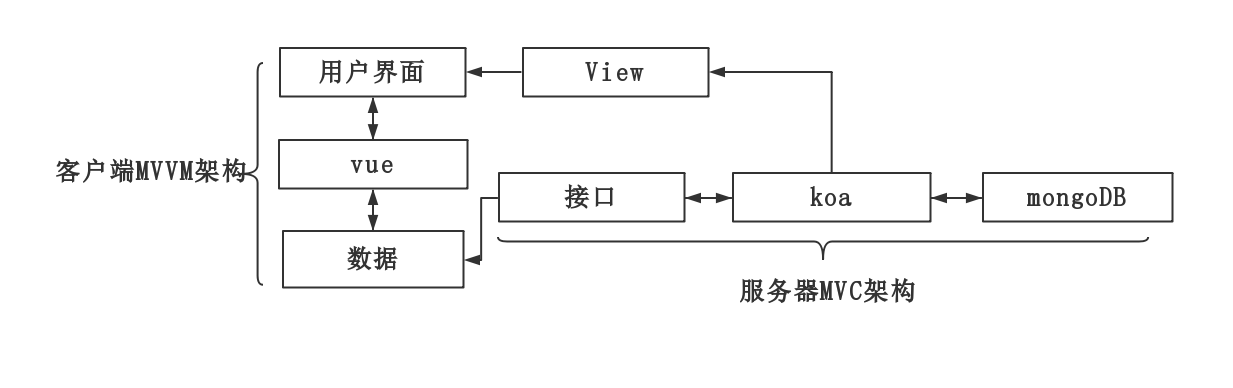


图4-16修改用户密码过程图

## 4.2 系统总体设计

根据系统需求分析的基础上，进行系统的总体设计。系统的总体设计如图4-17所示

图4-17系统架构图



考虑到系统的可维护性和可扩展性，系统设计分为前端应用程序设计和服务器应用程序设计两部分。前端应用程序由Vue负责用户界面的双向绑定，使用MVVM架构。服务器应用程序由koa框架实现服务器接口数据接收，业务处理，View渲染和操作数据库，使用MVC架构。

本系统采用前后端分离的设计方式，所以需要选择系统的设计顺序。因为本系统前端应用程序没有选择mock数据相关的技术，而服务器应用程序可以使用postman模拟发送异步请求，所以系统开发顺序上选择从设计服务器应用程序开始，最后进行前端应用程序的设计以及最终的测试。

### 4.2.1 Node服务器应用程序的整体架构

Node服务器应用程序采用传统的MVC框架进行开发，MVC框架中，往往会把应用程序划分为三个部分，模型层，视图层和控制层[17]：

模型层是业务规则和数据模型的设计[18]，它对数据操作进行了一定的规则限定；视图层负责页面渲染，如果模型层发生变化，视图层收到模型层数据变更的事件，然后根据数据状态渲染页面，视图层可以查询模型层的状态，但不能直接修改模型层，可以通过控制层完成对模型层状态的修改。控制层可以选择渲染的视图，以及接受视图层请求对模型层进行状态修改。传统的MVC架构如图4-18

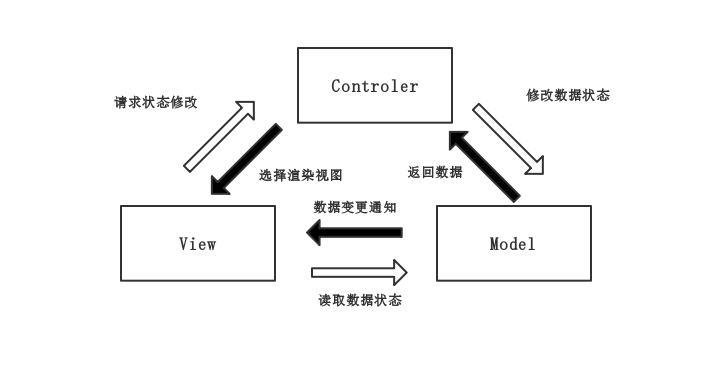


图4-18传统MVC架构图

Node服务器应用程序的架构会在传统的MVC架构上做出调整，由于视图的渲染绝大部分已经交给了前端应用程序去完成，MVC架构上的View层基本上已经分离出来了，所以，做出以下设计：

(1) model 存储业务规则和数据模型

(2) controler 负责为前端应用程序提供服务接口，负责修改和读取model的数据状态

(3)view 的业务绝大部分分离出来让前端应用程序处理

最终Node服务端的应用程序结构如下图4-19

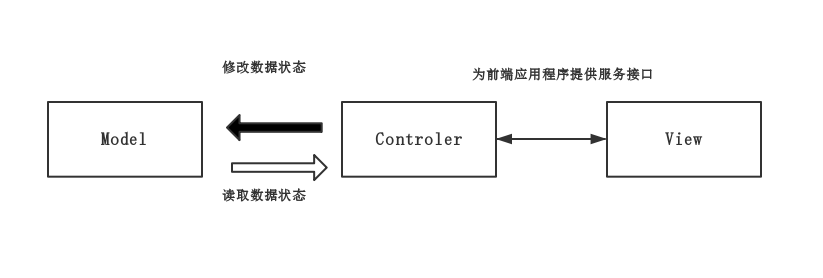


图4-19Node服务端的应用程序架构图

### 4.2.2 Vue前端应用程序的整体架构

node服务器应用程序的架构将传统MVC框架的View层分离出来由前端应用程序负责，本系统前端应用程序使用Vue完成页面渲染的功能。所以前端应用程序使用MVVM的架构完成,Vue本身是一个MVVM框架，所以不必要进行太多的架构设计。

MVVM框架由视图层，模型层和视图模型层组成[19]:

（1）Model层，即前端应用程序设计的数据部分，只负责保持数据的状态，不关注View层如何进行渲染。

（2）View层负责页面的渲染，不与模型层进行交互。

（3）ViewMode层负责用户行为的响，数据绑定和通知页面渲染。

最终Vue前端应用程序的架构设计如图4-20所示

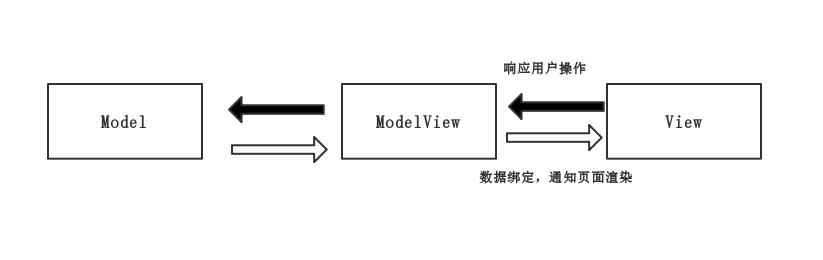


图4-20 Vue前端应用程序的整体架构图

## 4.3 本章小结

本章节围绕系统分析，首先从系统的功能需求出发，提出并设计了19个功能需求，然后从功能需求的基础上进一步对系统进行总体设计，总体设计上本系统分为Node服务器应用程序项目和Vue前端应用程序项目，其中Node服务器应用程序项目使用改造后的MVC架构，而Vue前端项目使用的是Vue框架所具备的MVVM架构。

# 第5章 系统的设计与实现

系统设计上，我们将系统划分为node服务器应用程序和Vue前端应用程序，设计顺序上，首先配合postman设计并实现node服务器应用程序，最后再完成Vue前端应用程序项目并进行整体功能的测试。

## 5.1 Node服务器应用程序的设计与实现

该部分的设计过程为：

（1）从Node服务器应用程序的架构出发，进行目录结构及各文件功能设计。

（2）从系统的需求分析出发，进行数据库模型设计

（3）进行服务端路由设计

（4）最后实现各路由功能

### 5.1.1 目录结构及各文件功能设计

基于node服务器的整体架构，进行目录结构以及各文件功能的设计，有利于分析和落实各文件完成的具体功能。具体目录设计及文件如图5-1所示：

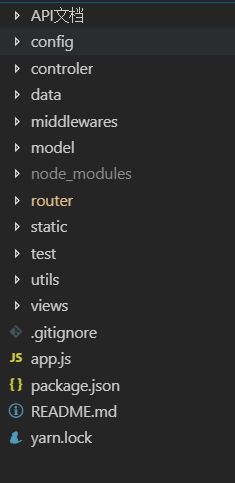


图5-1Node服务器应用程序的目录文件示意图

各目录及文件功能描述如下：

（1）API文档。

用于服务器应用程序接口的联调校对，以及为前端应用程序提供接口请求参考

（2）config

服务器应用程序的基本配置文件，如配置服务器的端口，数据库的连接，密码加密加盐的轮次（saltRounds），生产token的私钥以及过期时间，重置密码临时token的过期时间，基础路由（baseUrl），以及其他一些服务器的基本配置。

（3）controler

路由的处理控制器。对于本系统，控制器可以分为对用户相关路由的控制处理，对资源相关路由的相关处理，以及token认证的控制处理。

（4）Data

存放json数据。在数据库模型还未建立情况时，还可以作为接口的测试数据

（5）middlerwares

存放koa自定义的中间件函数。koa具有中间件的概念，可以理解为，在接受请求和返回响应对数据进行处理的函数

（6）model

数据模型。本系统设计主要有两个模型，User模型和Resource模型

（7）node\_modules

存放系统所需包依赖

（8）router

存放路由文件，简单可以分为用户路由，资源路由，token路由，以及一些测试路由

（9）static

存放静态资源文件

（10）test

进行测试的文件夹

（11）utils

存放自定义的工具函数。当需要重复使用某些功能，可以将他们封装为一个自定义的工具函数放在该文件夹下。本系统需要自定义邮件发送函数，token生成与验证函数，验证函数，密码加密函数等。

（12）views

存放视图文件，虽然视图层已经在本系统已经分离出来让Vue前端应用程序进行，对于一些操作，如邮箱的验证，仍然可以由koa利用模板引擎渲染该文件下的视图文件

（13）.gitignore

设置git提交时忽略的文件。本系统代码同步到github，有一些文件是没有同步的必要的，如node\_modules下的包依赖文件

（14）app.js

node服务器应用程序的入口文件

（15）package.json 以及 yarn.lock

包管理工具yarn生成的文件，用于记录项目下载的包信息，以及一些项目常用项目配置

（16）README.md

项目的说明文件

### 5.1.2 数据库模型设计

从系统的需求分析出发，将系统的模型划分为两个模型，User模型以及Resource模型。

User模型的设计如表5-1所示

表5-1 User模型设计

|  |  |  |  |
| --- | --- | --- | --- |
| 属性 | 属性类型 | 其他限制 | 描述 |
| username | String | 必须 | 用户名 |
| email | String | 必须 | 邮箱 |
| password | String | 必须 | 密码 |
| status | Number | 默认0 | 激活状态 |
| reset\_password\_token | String |  | 重置密码临时token |
| reset\_password\_expires | Date |  | 重置密码token过期时间 |
| avatar | String |  | 头像地址 |
| gender | Number | 默认为0 | 性别 |
| age | String |  | 年龄 |
| birthday | [String] | 默认[] | 生日 |
| location | String |  | 常住地 |
| mtto | String |  | 座右铭 |
| introduction | String |  | 简介 |
| private | Boolean | 默认0 | 是否私密 |
| collections | [String] |  | 收藏 |
| resources | [String] |  | 资源 |
| create\_time | Date | 默认Date.now | 创建时间 |
| bootstrap.vue | Vue组件库 |  | v2.0.0-rc.18及以上 |

Resource 模型的设计如表5-2所示

表5-2 Resource模型设计

|  |  |  |  |
| --- | --- | --- | --- |
| 属性 | 属性类型 | 其他限制 | 描述 |
| classification: | Number | 默认0 | 分类 |
| author\_id | String |  | 作者id |
| author\_name | String |  | 作者名 |
| last\_update\_time | Date | 默认Date.now | 最后更新时间 |
| title | String |  | 标题 |
| content | String |  | 正文内容 |
| like\_counts | Number | 默认0 | 点赞数量 |
| read\_counts | Number | 默认0 | 阅读数量 |
| collection\_counts | Number | 默认0 | 收藏数量 |
| birthday | [String] | 默认[] | 生日 |
| comments | Mix |  | 评论对象 |

### 5.1.3 Node服务端路由设计

从系统需求的各功能出发，对路由进行设计。路由设计使用REST架构。REST架构是一种软件的风格架构约束，本系统路由的规划遵循REST架构，并做出以下规定：

（1）URI使用小写字母，尽量不使用动词；用中杠线，不用下杠线；资源集合用复数表示

（2）使用统一规定的Http方法，用post新建资源，用put更新资源，get获取资源，delete删除资源

（3）响应数据返回的格式为JSON

（4）错误处理，本系统统一约束code为0表示错误，并将错误信息放在message中，code为1表示正确处理，正确信息也放在message中

（5）状态码。200表示成功返回，201表示创建资源成功。使用三个表示错误的状态码，404表示找不到资源，401表示token认证失败，500表示服务器内部错误

综合以上本项目要实现的REST设计原则，以及考虑到协助后续前端应用程序开发更容易理解服务器的接口请求，方便服务器端接口的维护工作，需要对所有接口请求进行一般性的设计。本项目的服务器接口返回数据一般性设计的伪代码如下所示：



具体地，根据接口的一般性设计，各路由的具体设计如表5-3所示：

表5-3 服务器路由具体设计表

|  |  |  |  |
| --- | --- | --- | --- |
| http请求方法 | 路由 | 描述 | data |
| GET | /token | token 身份认证 | user |
| GET | /recommend-resources/:sort | 获取推荐资源 | resources |
| GET | /classification-resources /:classification/:sort | 获取分类资源 | resources |
| GET | /search | 获取分类资源 | resources |
| POST | /user | 注册 | null |
| GET | /email-activation/:captcha | \*激活验证，特殊处理\* | null |
| POST | user/actions/login | 登陆 | user |
| POST | /forget-password | 发送重置密码邮件 | null |
| POST | /reset-password/:token | 重置密码 | null |
| GET | /resources/:\_id | 访问资源 | resource |
| GET | /users/:\_id | 获取用户信息 | user |
| POST | /users/resources | 发布资源 | null |
| PUT | /resources/:r\_id | 修改资源 | null |
| POST | /resources/:\_id/comments | 评论资源 | null |
| PUT | /resources/:\_id/collection-counts | 收藏资源 | null |
| PUT | /resources/:\_id/like-counts | 点赞资源 | null |
| PUT | users/profile | 修改个人资料 | null |
| PUT | /users/password | 修改用户密码 | null |

特殊地，为了用户界面更加友好，对于注册邮箱的验证处理请求，服务端不会像其他路由一样响应发送json数据，而是根据用户的激活状态利用模板引擎（本项目为ejs模板引擎）选择渲染模板，发送html页面响应给客户端。

### 5.1.4各路由功能的实现

客户端发起请求，Node服务端应用程序会根据路由的设定响应请求并做出处理或具体功能的实现，本项目的路由功能处理交由系统框架中的controler进行处理。以下为各路由功能的实现的逻辑。

(1) token 进行身份认证。token认证的具体逻辑如下图5-2所示

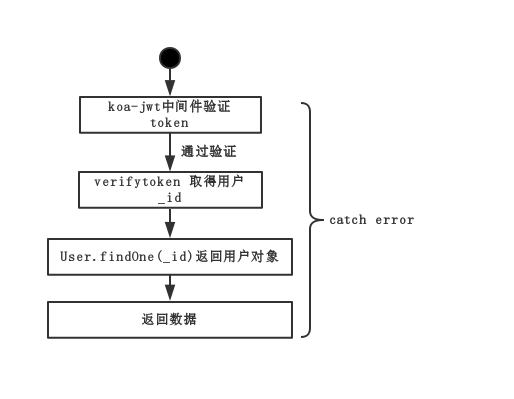


图5-2token认证逻辑图

(2)首页获取推荐资源。首页推荐资源的具体逻辑如图5-3所示

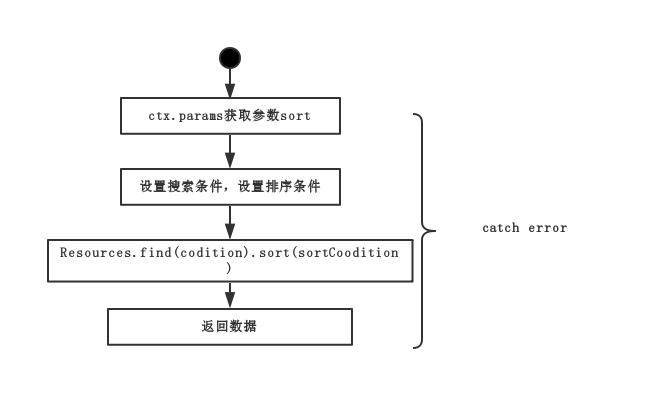


图5-3首页获取推荐资源图

(3) 获取分类资源。获取分类资源的具体逻辑如图5-4所示



图5-4获取分类资源图

(4)获取搜索资源。获取搜索资源的具体逻辑如图5-5所示



图5-5获取搜索资源图

(5) 获取标签资源。逻辑与获取搜索资源一样

(6) 注册以及邮箱验证激活。注册以及邮箱激活的具体逻辑如图5-6和图5-7所示

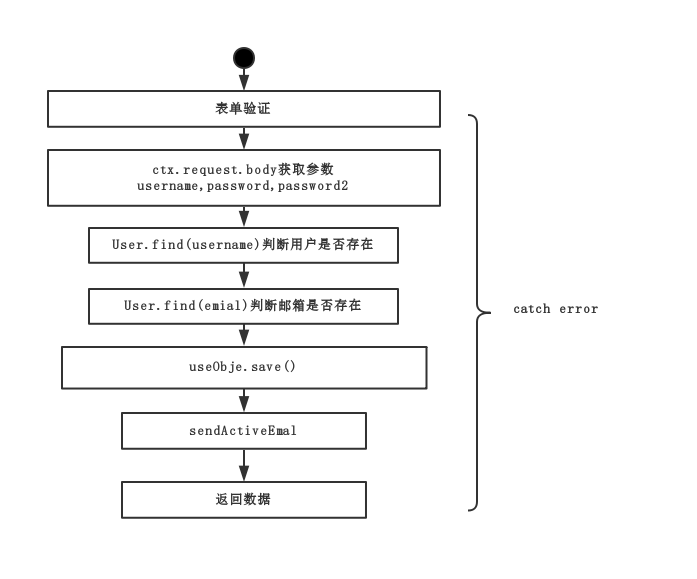


图5-6注册逻辑图

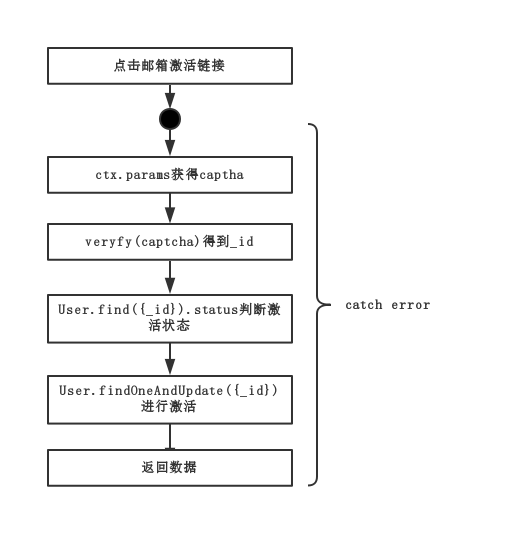


图5-7邮箱验证逻辑图

（7）登录。登录的具体逻辑如图5-8所示

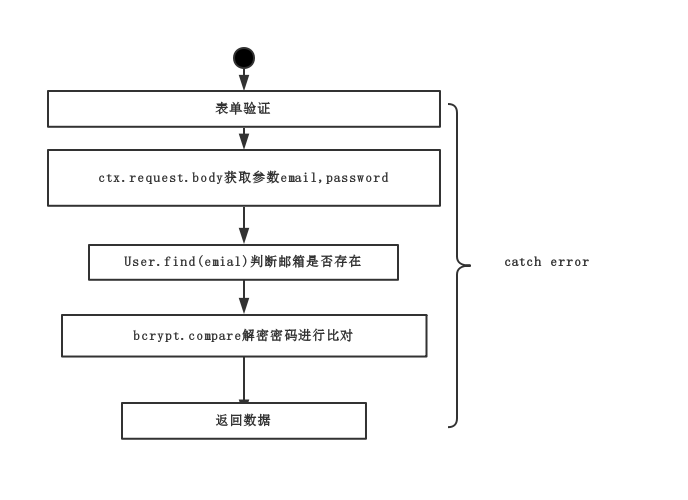


图5-8登录逻辑图

（8）忘记密码。忘记密码和重置密码的具体逻辑如图5-9和图5-10所示

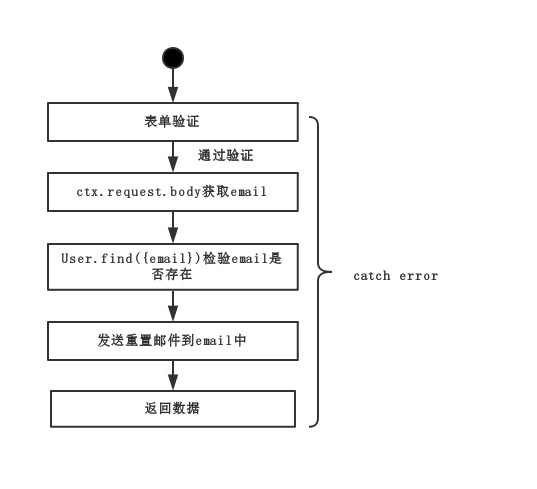


图5-9登录逻辑图

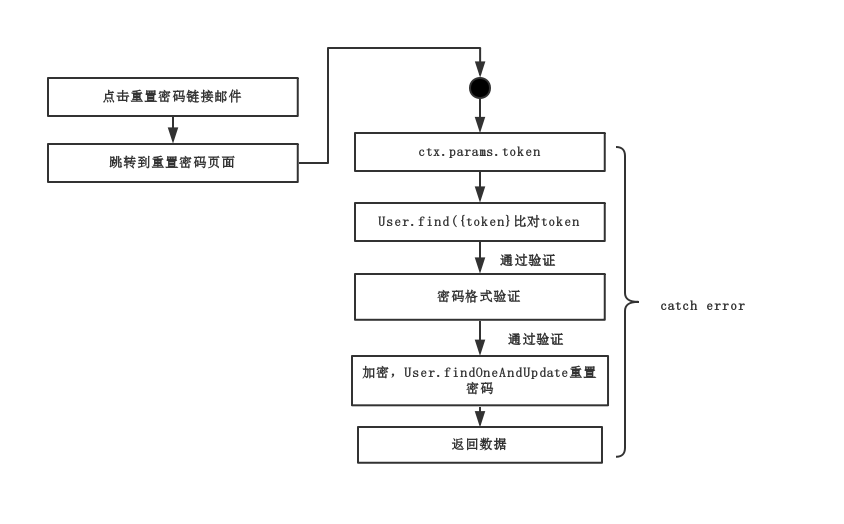


图5-10重置密码逻辑图

（9）访问资源。访问资源的具体路基如图5-11所示

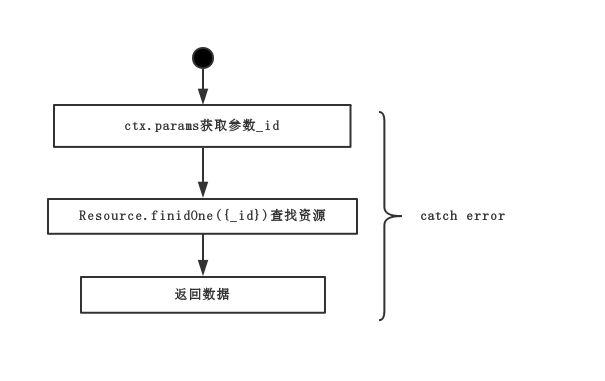


图5-11访问资源逻辑图

（10）获取用户资料。获取用户资料的逻辑图如图5-12所示

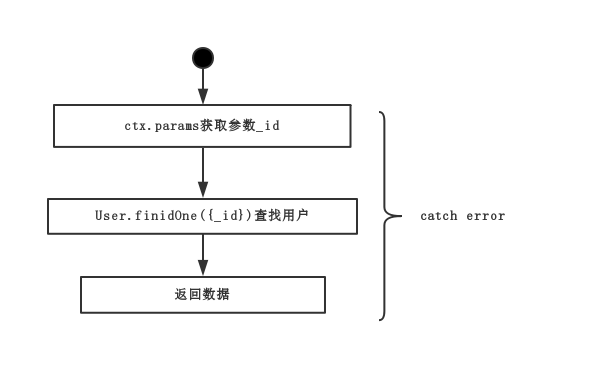


图5-12获取用户资料逻辑图

（11）关于我们以及其他静态页面

不进行请求，不需要控制器（controler）进行处理

（12）注销

不发起请求，不需要控制器（controler）进行处理

（13）发布资源。发布资源的具体逻辑如图5-13所示

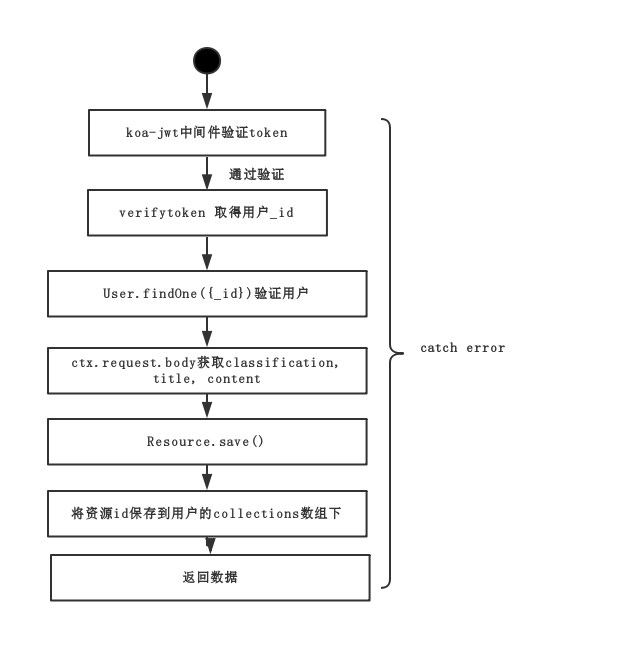


图5-13发布资源逻辑图

（14）修改资源。修改资源具体逻辑如图5-14所示

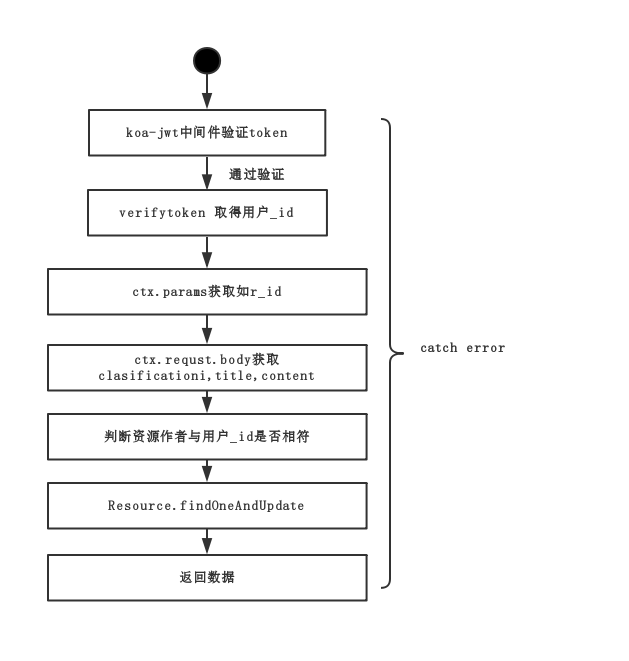


图5-14修改资源逻辑图

（15）评论资源。评论资源的具体的逻辑如下图5-15所示

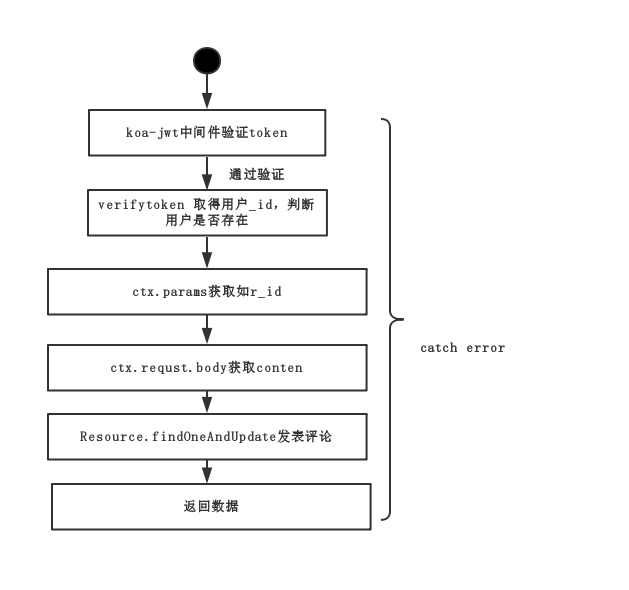


图5-15评论资源逻辑图

（16）收藏资源。收藏资源的具体逻辑如图5-16所示

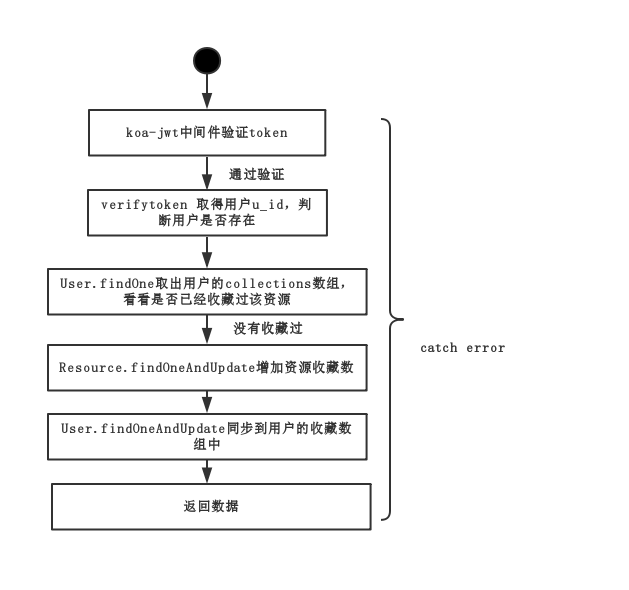


图5-16收藏资源逻辑图

（17）点赞资源。点赞资源的具体逻辑如图5-17所示

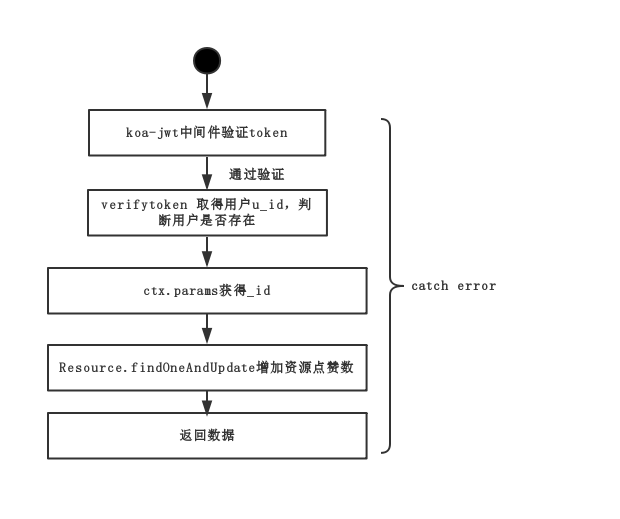


图5-17点赞资源逻辑图

（18）修改个人资料。修改个人资料的逻辑如图5-18如下

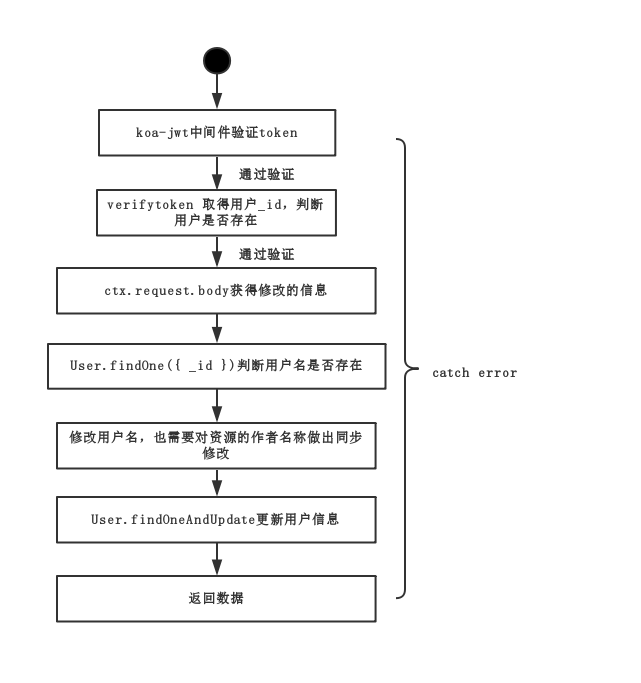


图5-18修改用户资料逻辑图

（19）修改密码。修改用户密码的具体路基如图5-19所示

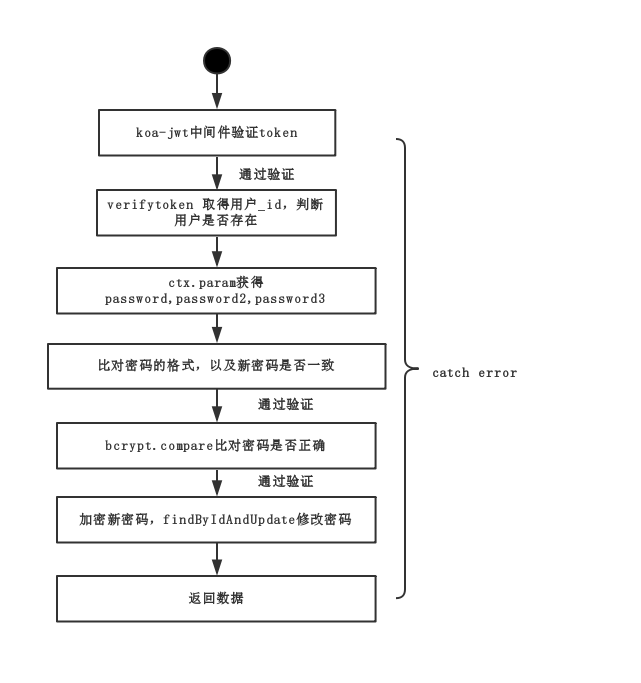


图5-19修改密码逻辑图

## 5.2 Vue前端应用程序的设计与实现

Vue是一个MVVM框架，借助Vue-cli脚手架，很迅速就可以完成项目及其环境的配置。Vue前端应用程序的设计与实现过程如下：

（1）在Vue-cli脚手架的工程配置支持下，进行目录结构和文件功能分析

（2）根据node服务器提供的接口，系统功能分析的基础上进行前端路由和组件渲染关系设计

（3）封装axios请求

（4）Vuex状态管理

（5）网站主要界面展示及其功能测试

（6）网站响应式设计

### 5.2.1目录结构及各文件功能设计

在Vue-cli的基础上，进行前端应用程序的目录结构和文件功能设计，有利于应用程序的功能模块划分和增强可维护性。具体目录和文件如图5-20所示：

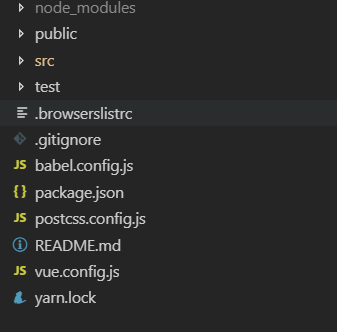


图5-20Vue前端应用程序目录及文件图

各目录和文件的具体功能描述如下

（1）node\_modules

包依赖文件

（2）public

公共目录，存放logo图片，渲染模板等

（3）src

源码目录

（4）test

测试文件夹

（5）.browserslistrc

浏览器的兼容性配置

（6）babel.config.js

babel的配置。babel用于转换es6的语法

（7）package.json 与 yarn.lock

包管理工具yarn产生的文件，package.json还可以用于基本环境的配置

（8）postcss.config.js

postcss的配置

（9）README.md

项目说明文档

（10）vue.config.js

vue-cli的基本配置。可以根据系统需要自定义配置loadder及plugin

对于src源码目录下的文件设置如图5-21所示

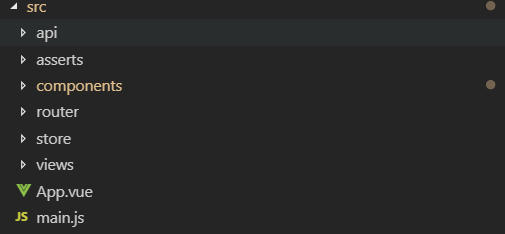


图5-21 src源码目录文件示意图

其中源码目录下的文件具体功能描述如下:

（1）api

存放axios封装的库

（2）asserts

存放项目中的资源，如图片文件，字体文件，css文件，scss文件等

（3）components

存放组件

（4）router

存放前端路由文件

（5）store

存放vuex相关文件

（6）views

存放vue视图文件

（7）App.vue

默认的视图入口文件

（8）main.js

默认项目入口文件

### 5.2.1前端路由与组件渲染关系设计

根据系统总体设计的功能需求，路由与组件的渲染关系如下

（1）首页，默认渲染推荐资源。

路由：/recommend-resource

路由别名：/

首页组件的UI设计如图5-22所示

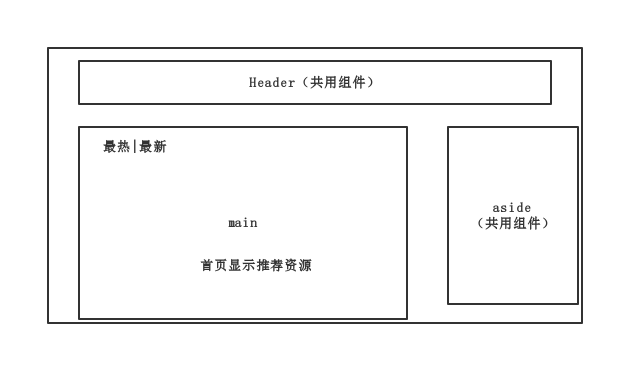


图5-22首页UI设计图

（2）分类资源页面

路由：/classification

路由别名：/classification/:classification

分类资源组件的UI设计如图5-23所示

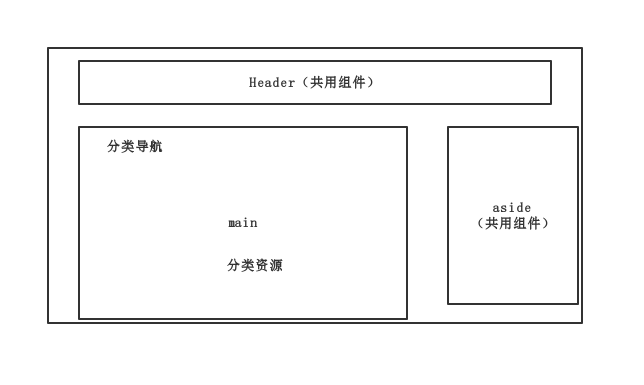


图5-23分类资源UI设计图

（3）搜索资源页面

路由：/search/:title

搜索资源组件的UI设计如图5-24所示

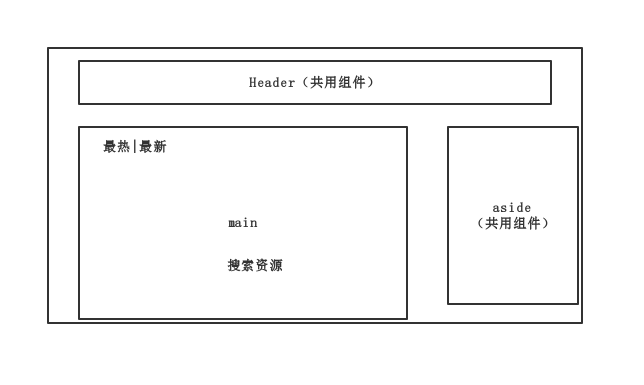


图5-24搜索资源UI设计图

（4）注册页面

路由：不需路由渲染

触发途径：点击注册按钮，打开注册绑定的modal

注册modal的UI设计如图5-25所示

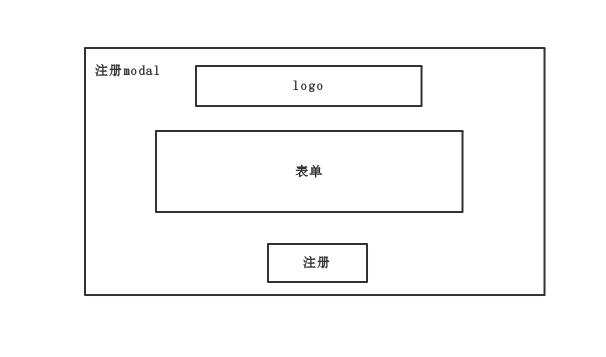


图5-25注册modal UI设计图

（5）登录页面

路由：不需路由渲染

触发途径：点击登录按钮，打开登录绑定的modal

注册modal的UI设计如图5-26所示

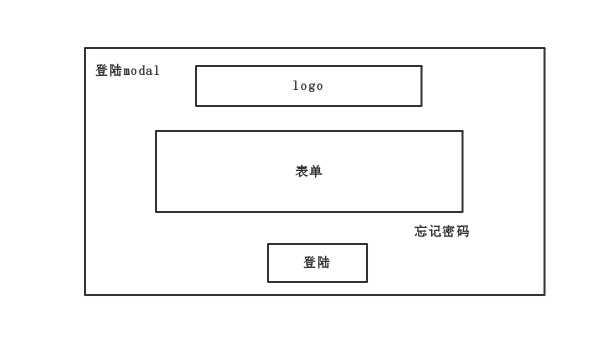


图5-26登录modal UI设计图

（6）忘记密码页面

路由：不需路由渲染

触发途径：在登录modal内部，点击忘记密码按钮，打开忘记密码绑定的modal

注册modal的UI设计如图5-27所示

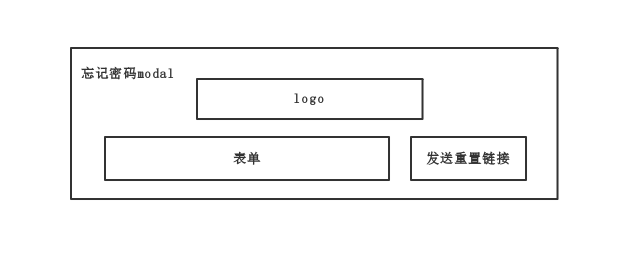


图5-27忘记密码modal UI设计图

（7）关于我们

路由：/about-us

关于我们页面的UI设计如图5-28所示

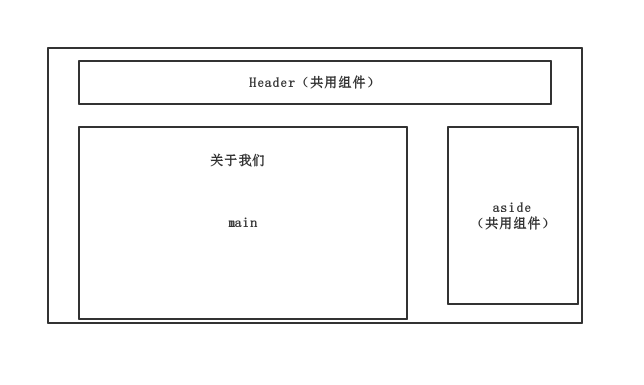


图5-28关于我们UI设计图

（8）发布资源

路由：/write

发布资源页面的UI设计如图5-29所示



图5-29发布资源UI设计图

（9）修改资源

路由：/resources/:\_id/edit

修改资源页面的UI设计如图5-30所示



图5-30修改资源UI设计图

（10）访问资源

路由：/resources/:\_id

访问资源页面的UI设计如图5-31所示

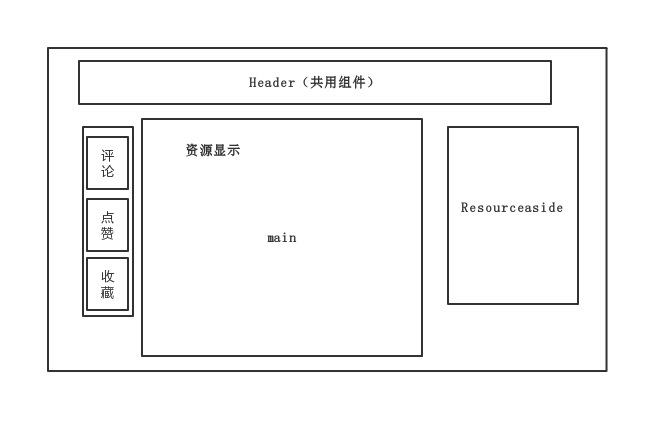


图5-31访问资源UI设计图

（10）评论资源

路由：无

触发途径：通过点击评论按钮，显示评论modal

评论modal的UI设计如图5-32所示

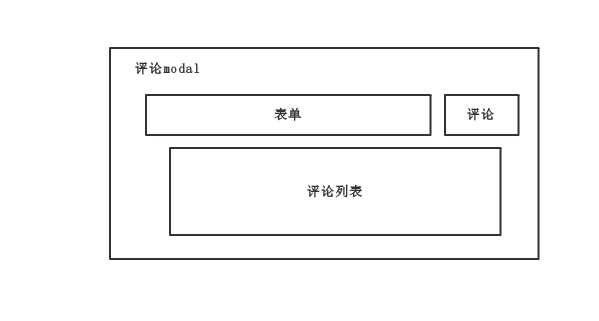


图5-32评论modal UI设计图

（10）获取用户资料

路由：/users/:\_id

子路由：basic-info 渲染子组件基本信息

子路由：resources 渲染子组件个人资源

子路由：collections 渲染子组件个人收藏

获取用户资料页面的UI设计如图5-33所示

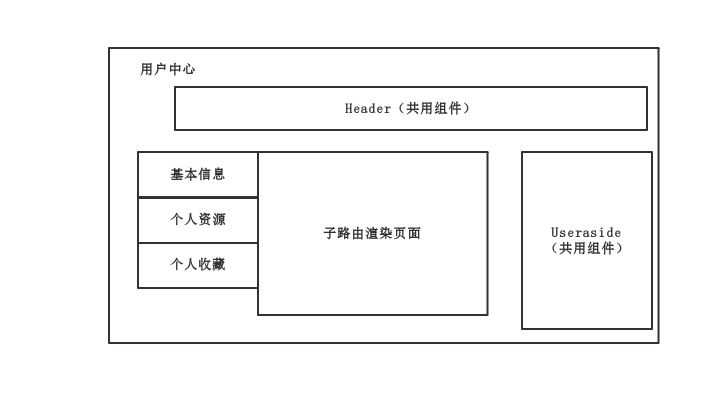


图5-33获取用户信息UI设计图

（10）设置页面

路由：/users/:\_id/settings

子路由：profile

子路由：password

设置页面的UI设计如图5-34所示

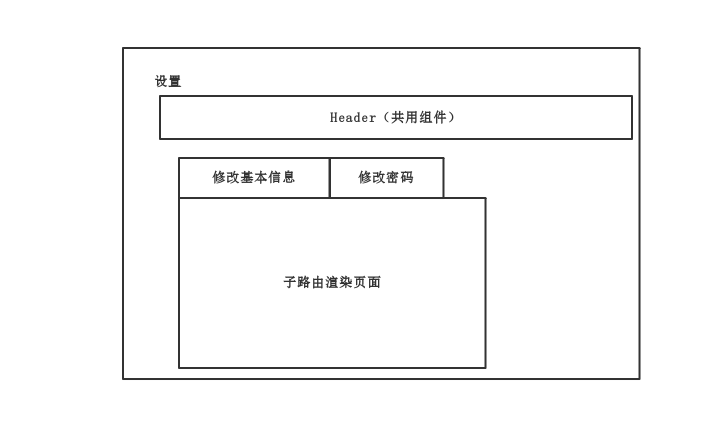


图5-34设置UI设计图

（10）404页面

对于没有配置路由规则的路由，渲染404页面

404页面UI设计如图5-35所示

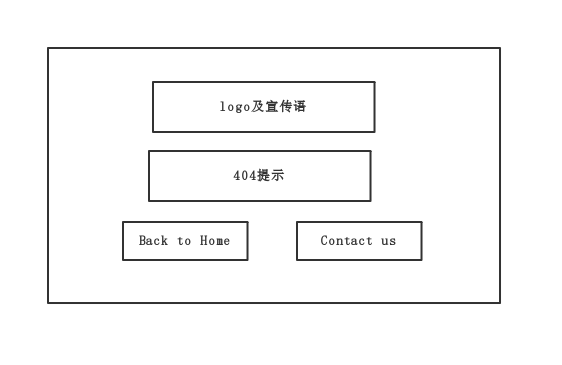


图5-35 404页面UI设计图

### 5.2.3封装axios请求

本系统的Vue前端应用程序使用Ajax向服务器应用程序发送异步请求，原生的Ajax容易出现回调嵌套的问题，因为axios基于promise进行封装，所以使用axios发起异步请求能够避免回调地狱的出现。

axios集成了一系列异步请求的方法，为了能够使用axios的请求拦截器以及在promise语法基础上使用async函数语法，需要进一步封装axios。

首先需要配置请求拦截器，请求拦截器可以为在发送异步请求前对请求进行配置，这里需要为每个请求带上token，配置拦截器的具体逻辑如图5-48所示



图5-36 拦截器逻辑图

最后，为了可以使用async的语法直接取到数据，我们还需要封装get和post，put方法，封装的一个一般性例子如下所示



至此，axios的基本封装完成

### 5.2.4 Vuex状态管理

Vue之间的数据传递是上下级的数据传递，Vue没有提供直接关联所有组件的状态的管理方法，所以需要使用Vuex进行统一的状态管理。

使用Vuex需要建立一个store，并将它挂载到Vue实例上。store中需要配置好state，mutations和actions。

（1）state，存储store的响应式的数据。我们需要设置的state如下表5-4

表5-4 state属性

|  |  |  |
| --- | --- | --- |
| 属性 | 描述 | 默认值 |
| isLogin | 是否登录 | false |
| user | 用户对象 | {} |
| profile | 个人中心对象 | {} |
| resource | 资源对象 | {} |
| recommendSourcesList | 推荐资源列表 | [] |
| classificationList | 分类资源列表 | [] |
| searchList | 搜索资源列表 | [] |

（2）mutation，mutation用于改变state，改变state的唯一方法，就是对mutation进行commit。mutations设置如下表5-5

表5-5 mutations

|  |  |
| --- | --- |
| 属性 | 行为描述 |
| LOG\_IN | isLogin设为ture，赋值user对象 |
| LOG\_OUT | isLogin设为false，赋值user对象为空 |
| VALIDATE\_TOKEN | isLogin设为ture，赋值user对象 |
| GET\_RECOMMEND\_RESOURCES | 赋值recommendSourcesList |
| GET\_CLASSIFICATIOIN\_RESOURCES | 赋值classificationList |
| SEARCH | 赋值searchList |
| GET\_RESOURCE | 赋值resource |
| GET\_USER\_INFO | 赋值profile对象 |

（3）action, action用于发送异步请求获取数据，将数据处理后作为commit参数调用相关的mutation，从而完成state的属性值的改变。actions设置如下表5-6

表5-6 actions

|  |  |
| --- | --- |
| 属性 | 行为描述 |
| login | 发起登录请求，commit LOGIN |
| logout | commit LOG\_OUT |
| register | 发起注册请求 |
| validateToken | commit VALIDATE\_TOKEN |
| forgetPassword | 发起忘记密码请求 |
| resetPassword | 发起重置密码请求 |
| getRecommendResources | 发起请求推荐资源请求，commit GET\_RECOMMEND\_RESOURCES |
| getClassificationResources | 发起请求分类资源请求，commit GET\_CLASSIFICATIOIN\_RESOURCES |
| write | 发起新建资源请求 |
| getResource | 发起访问资源请求，commit GET\_RESOURCE |
| comment | 发起评论请求 |
| addLikeCounts | 发起点赞请求 |
| addCollectionCounts | 发起收藏请求 |
| edit | 发起编辑资源请求 |
| getUserInfo | 发起获取用户资料请求，commit GET\_USER\_INFO |
| setUserInfo | 发起修改用户资料请求 |
| changePassword | 发起修改密码 |

### 5.2.5网站主要界面展示及其功能测试

（1）logo以通用的分享图标加上SKE组成。SKE是“Share Knowledge EveryWhere” 的缩写，logo如图5-37所示



图5-37 logo示意图

（2）Header 是一个很多页面共用的组件，也是关键的路由导航所在，最中header的设计如图5-38所示

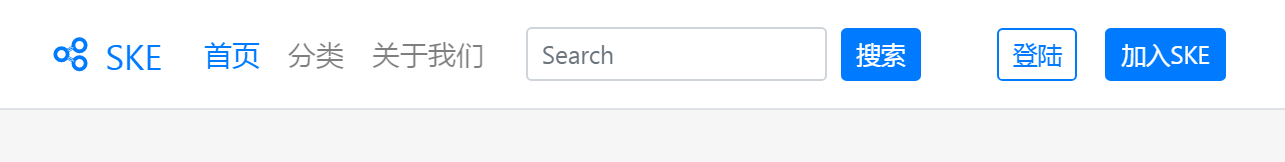
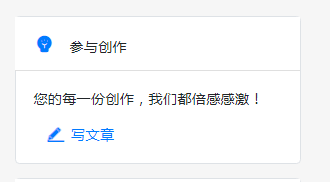


图5-38 放大后的header 示意图

（3）Aside组件也被很多页面所共用，它主要展示了一些网站相关信息，以下为Aside组件最终设计，由参与创作，参与我们，热门标签，网站信息构成，最终设计如图5-39所示



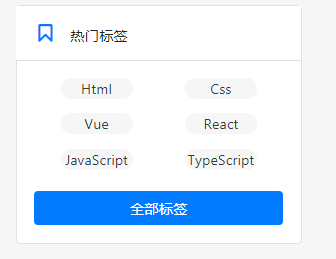
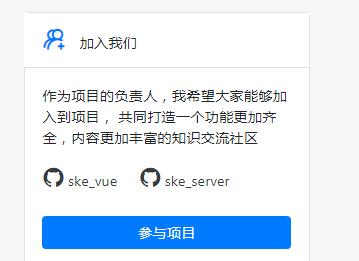
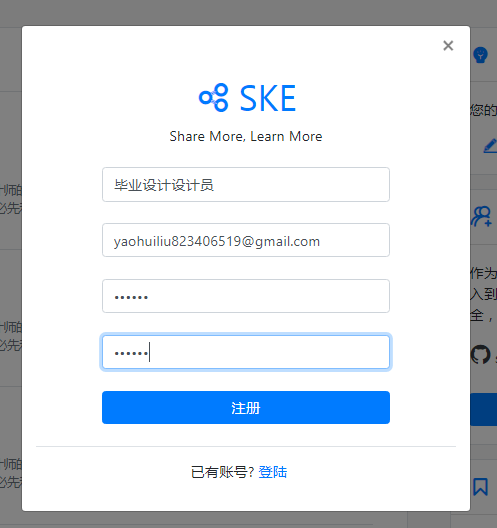
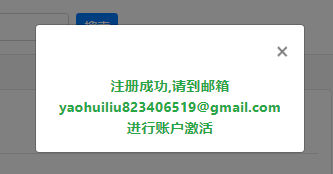


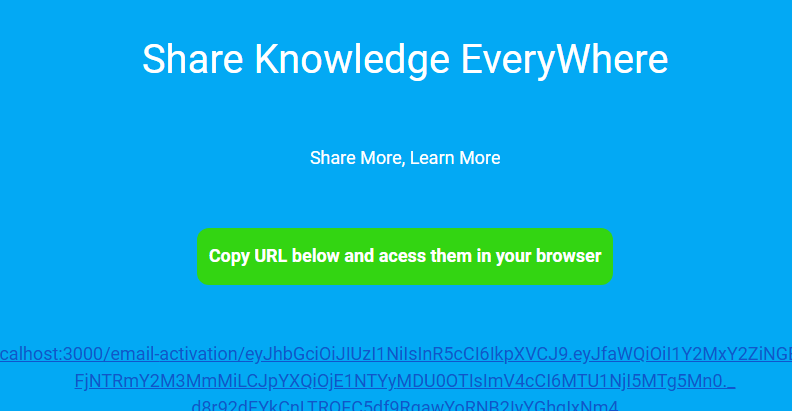


图5-39 分模块截图的Aside示意图

（4）注册最终表现如下。如图5-40所示







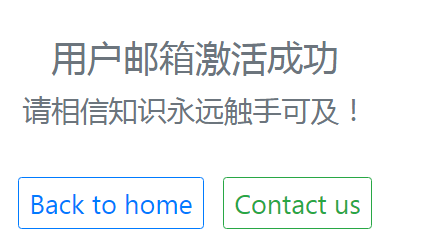


图5-40 注册过程示意图

（5）登录过程如图5-41所示



图5-41 登录示意图

（6）忘记密码的过程如图5-42所示









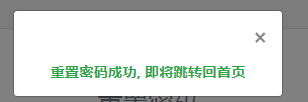


图5-42忘记密码过程示意图

（6）发布资源的过程如图5-43所示



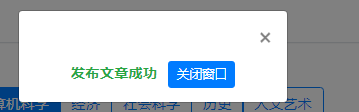


图5-43发布资源过程示意图

（6）打开个人中心的过程如图5-44所示



图5-44打开个人中心过程示意图

（6）首页推荐资源的示意图，如图5-45所示  


图5-45 首页推荐资源示意图

（6）分类资源的示意图，如图5-46所示



图5-46 打开个人中心过程示意图

（7）搜索资源的示意图，如图5-47所示

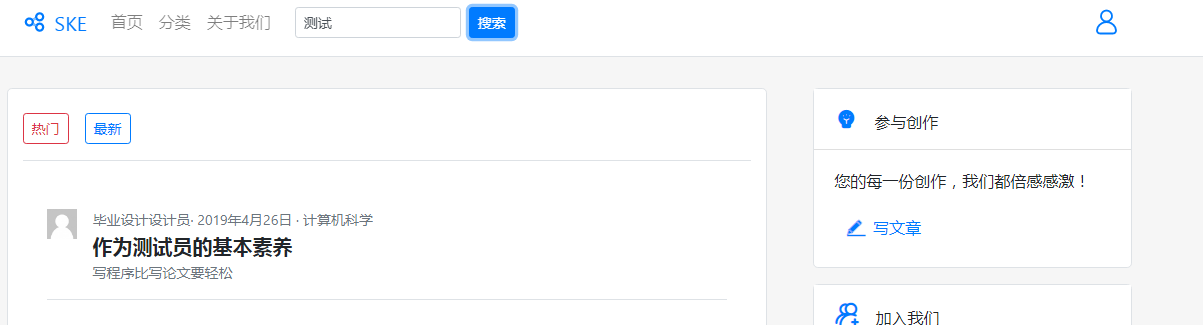


图5-47 搜索资源示意图

（8）访问资源示意图如图5-48所示



图5-48 访问资源资源示意图

（9）编辑资源如图5-49所示



图5-49编辑资源示意图

（10）评论资源如图5-50所示

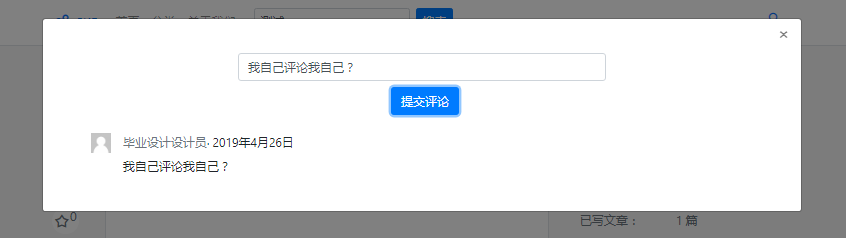


图5-50评论资源示意图

### 5.2.1网站响应式设计

响应式网页设计即将媒介查询(Media Queries)、流式布局(Fluid Grids)、自适应图片(Scalable Images)结合起来的设计理念[20]。本系统使用bootstrap的相关类对网站进行相关响应式的设计，以下是本系统设计过程为了实现响应式设计使用的bootstrap类，见表5-7

表5-7 bootstrap相关响应类

|  |  |
| --- | --- |
| 属性 | 行为描述 |
| col-[breakpoint]-[0-12] | 根据视窗宽度达到的哪个breakpoint设置col宽度 |
| m-[brekpoint]-[0-5] | 根据视窗宽度达到的哪个breakpoint设置margin |
| p-[breakpoint]-[0-5] | 根据视窗宽度达到的哪个breakpoint设置padding |
| my-[brekpoint]-[0-5] | 根据视窗宽度达到的哪个breakpoint设置上下margin |
| mx-[brekpoint]-[0-5] | 根据视窗宽度达到的哪个breakpoint设置左右margin |
| m[t,b,r,l]-[breakpoint]-[0-5] | 根据视窗宽度达到的哪个breakpoint设置指定margin |
| py-[breakpoint]-[0-5] | 根据视窗宽度达到的哪个breakpoint设置上下padding |
| px-[breakpoint]-[0-5] | 根据视窗宽度达到的哪个breakpoint设置左右padding |
| p[t,b,r,l]-[breakpoint]-[0-5] | 根据视窗宽度达到的哪个breakpoint设置指定padding |
| d-[breakpoint]-block | 根据视窗宽度达到的哪个breakpoint设置显示 |
| d-[breakpoint]-none | 根据视窗宽度达到的哪个breakpoint设置隐藏 |
| d-[breakpoint]-none | 根据视窗宽度达到的哪个breakpoint设置隐藏 |

## 5.3 本章小结

本章实现系统的设计与实现，为了便于扩展与维护，采用了前后端分离的方式。对于Node服务端应用程序，基于特定的MVC架构，设计程序目录及文件功能，设计数据模型，进行服务端路由的设计，以及各路由的控制器的逻辑实现，服务端程序的是否正常运行，使用postman软件进行辅助。对于Vue前端程序，设计其目录结构和功能，设计路由与组件，封装axios，设计vuex store，进行界面的展示与功能测试，最后对Vue前端项目利用bootstrap类进行响应式改造。

# 结论

本系统项目是一个前后端分离的项目。这是一个主题鲜明的网站，致力于构造一个知识分享，交流的网站，建设网站的目的始终都像本网站名称一样，“Share Knowledge EveryWhere”，为更多需要搜索知识，分享知识的人们提供一个平台。

本系统采用前后端分离的设计方式，意味着，我们在往后的维护工作中，更方便进行项目维护和扩展。对于本项目，node服务器应用程序使用的是Koa和mongoose框架，前端项目使用的是Vue技术栈。

本项目从分析行业和技术现状，分析系统需求到系统设计以及实现，始终围绕实现以下功能为中心进行：

（1）实现token认证

（2）基本的用户账户管理。如实现用户登录，用户注册，用户找回密码，用户修改密码，用户编辑个人信息等

（3）基本信息获取。在不危害系统以涉及到其他账户安全的操作，本系统具有很大的开放性，对于基本的资源获取，用户中心访问，用户资源访问，用户收藏访问都是不作账户登录限制。

（4）提供用户发表资源，修改资源的能力

（5）增强社区功能，如资源评论功能，资源点赞功能，资源收藏功能

本项目还具有以下特色：

（1）单页应用的用户体验。单页应用在首屏渲染上消耗时间比多页应用要长，但是在加载后的能给予用户更快地资源获取效率以及更强的无刷新体验

（2）简洁的页面布局。本项目布局css代码使用flex样式布局，flex样式布局能够更加快速完成页面的简单布局

（3）良好的动画效果。借助于bootstrap-vue，以及vue框架的transition过渡。

（4）响应式布局。本项目使用bootstrap类进行响应式布局开发，增强了项目在不同设备上的表现能力。

项目虽然基本构建完成，仍然具有以下问题

（1）由于时间仓促，虽然系统构建过程中，始终注意保持项目的模块化和工程化，最终并没有完美实现。

（2）很遗憾，项目中仍然遗留一些垃圾代码。虽然在项目中已经大量使用了ES6的新语法，看起来代码构建仍具有很大改进空间。

（3）对于系统没有进行大量测试，这是比较遗憾的，代码中存在的bug没能被及时发现。

（4）项目功能还不够完善，未来应该着眼于个性化推荐，完善用户评论体系，完善社区建设，更友善的用户界面操作，数据库模型以及controler对数据库的处理都是有待完善的

# 参考文献

1. 中国互联网络信息中心.1994年～1996年互联网大事记[EB/OL]. http://www.cnnic.net.cn/hlwfzyj/hlwdsj/201206/t20120612\_27415.htm
2. 中国互联网络信息中心.第43次中国互联网络发展状况统计报告[R]， 2019.
3. 中国互联网络信息中心.中国分享经济发展报告[R]，2019.
4. 罗敏,涂科.知识分享平台的商业模式探析[J].管理现代化,2018,38(06):111-113.
5. 艾媒报告.中国知识付费行业研究与商业投资决策分析报告[R]，2019.
6. 沈嘉熠.知识付费发展现状与未来展望[J].中国编辑,2018(11):35-39.
7. 邹伯涵,罗浩.知识付费——以开放、共享、付费为核心的知识传播模式[J].新媒体研究,2017,3(11):110-112+132.
8. 靖伟.Ajax技术的研究与应用[J].中国传媒大学学报(自然科学版),2015,22(06):50-55.
9. 阮一峰. ES6标准入门(第3版) [M]. 北京：电子工业出版社，2017.
10. Vue.js官网 . Vue.js 是什么[EB/OL].https://cn.vuejs.org/v2/guide/#Vue-js-是什么，2019
11. Vue router官网 . Vue route介绍[EB/OL]. https://router.vuejs.org/zh/，2019
12. Vuex官网 . Vuex 是什么[EB/OL].https://vuex.vuejs.org/zh/，2019
13. 朴灵. 深入浅出Node.js [M]. 北京：人民邮电出版社，2013.
14. 霍多罗夫. MongoDB权威指南（第2版） [M]. 北京：人民邮电出版社，2014.
15. yarn官网. yarn起步[EB/OL].https://yarnpkg.com/en/docs/getting-started，2019
16. 韩方圆. 基于REST和单点登录的财务管理系统设计与实现[D].北京工业大学,2016.
17. 霍福华.Web前端MVC框架的发展方向以及意义[J].软件工程,2019,22(04):44-46.
18. 张军芳. 基于J2EE平台和MVC模式的Web研究与应用[D].武汉理工大学,2008.
19. 刘春晓. 基于MVVM的软件架构模式MVVD的设计与实现[D].天津大学,2018.
20. 王愉,潘明明.响应式网页设计初探[J].北京印刷学院学报,2014,22(03):13-15.

# 致 谢

首先感谢我的指导老师赵波老师，赵波老师始终督促我们的论文进程，给予我我们多次的修改意见，他也是大学期间我们专业的多个技术课程的任课老师，在课程学习上也给予了我很大的帮助。

感谢Vue框架的创始人尤雨溪以及广大参与Vue社区建设的开发者。对于项目配置中使用到的相关plugin 和 loader 提供者同样表示感谢

感谢Node创始人Ryan Dahl及Node社区的开发者。

感谢Koa框架创始人T.J. Holowaychuk及其团队以及广大社区开发者。对于Koa中使用开源中间件贡献者同样表示感谢