**基于Node.js的博客系统**

**摘要**

随着互联网不断发展，各种新的编程语言框架层出不穷。本文主要介绍了如何应用node.js技术开发博客系统以及其设计与实现。博客文章管理系统主要为博客注册用户以及网友提供了发表文章、浏览博客文章等多种功能,用户通过internet系统可以随时发表一些自己喜欢撰写的文章以和其他的网友进行互动和交流。博客文章管理系统主要是实现了对文章发表管理的系统数字化、信息化、智能化,是一个打破了传统的报刊、杂志等网站发表文章管理方式的新一次尝试。

本系统的开发设计实现采用Node.js技术，整个系统使用Node.js的Koa框架编写而成，主要使用了Node.js的中间件技术以及路由技术，系统前台界面使用Node的中间件koa-ejs编写，采用了服务端渲染技术，系统后台使用MySQL 数据库，并通过使用中间件mysql以及co-mysql技术访问。

本文首先在第一章介绍了本系统开发的背景以及开发本系统的意义，然后在第二章对在开发本系统的过程中使用的主要技术进行了简单介绍，第三章主要对系统进行了相关分析，在本文第四章对当下的数据库进行了简单介绍，并说明了选择MySQL的原因，本文的五六两章主要讲的是系统的设计，实现以及展示。

**关键字**： Node.js；中间件；服务端渲染；MySQL；博客系统

**Node.js based blog system**

**Abstract**

With the continuous development of the Internet, various new programming language frameworks emerge in an endless stream.This article mainly introduces how to apply node.js technology to develop blog system and its design and implementation.Blog article management system is mainly for blog registered users and netizens to provide a variety of functions such as publishing articles, reading blog articles, users can at any time through the Internet system to publish some of their favorite articles to interact and communicate with other netizens.Blog article management system is mainly to achieve the article published management system of digitization, information, intelligence, is a new attempt to break the traditional newspapers, magazines and other websites published articles management way.

The development and design of the system adopts Node. Js technology, the whole system using Node. Js Koa framework to write and become, mainly using the Node. Js middleware technology and routing technology, system at the front desk interface using the middleware Koa - ejs, adopted the service side rendering technology, use the MySQL database system background, and through the use of middleware access MySQL and co - MySQL technology.

In this paper, first of all in the first chapter of this system development background and the significance of developing this system, and then in the second chapter in the development of this system used in the process of the main technology has carried on the simple introduction, the third chapter mainly has carried on the correlation analysis to the system, the fourth chapter in this paper, the current database has carried on the simple introduction, and explains the reasons of choosing MySQL, the 562 chapter of this paper is primarily concerned with the design of the system, implementation and presentation.

**Keyword:** Node.js; The Middleware; Server Side Rendering; MySQL; The Blog System

目录

[**1** 系统概述 3](#_Toc35158449)

[1.1 项目背景 3](#_Toc35158450)

[1.2 系统开发的意义 3](#_Toc35158451)

[1.3 系统开发的目标 3](#_Toc35158452)

[**2** 博客系统相关技术 4](#_Toc35158453)

[2.1 前端基础Html5 4](#_Toc35158454)

[2.2 前端基础CSS3 5](#_Toc35158455)

[2.3 前端基础JavaScript 7](#_Toc35158456)

[2.4 Node.js技术 8](#_Toc35158457)

[**3** 系统分析 10](#_Toc35158458)

[3.1 需求分析 10](#_Toc35158459)

[3.2 业务分析及流程图 10](#_Toc35158460)

[3.3 数据流图分析 11](#_Toc35158461)

[**4** 数据库选择及设计 11](#_Toc35158462)

[4.1 数据库选择 11](#_Toc35158463)

[4.2 数据表设计 11](#_Toc35158464)

[**5** 系统功能模块设计 12](#_Toc35158465)

[5.1 主服务器模块 12](#_Toc35158466)

[5.2 管理员登录模块 12](#_Toc35158467)

[5.3 Banner管理模块 12](#_Toc35158468)

[5.4 文章管理模块 12](#_Toc35158469)

[5.5 类目管理模块 12](#_Toc35158470)

[5.6 用户管理模块 12](#_Toc35158471)

[5.7 主页显示模块 12](#_Toc35158472)

[5.8 用户注册登录模块 13](#_Toc35158473)

[5.9 列表页显示模块 13](#_Toc35158474)

[5.10 文章详情显示模块 13](#_Toc35158475)

[5.11 错误提醒模块 13](#_Toc35158476)

[**6** 系统实现及展示 13](#_Toc35158477)

[6.1 主服务器的实现及展示 13](#_Toc35158478)

[6.2 管理员登录实现及展示 14](#_Toc35158479)

[6.3 Banner管理实现及展示 14](#_Toc35158480)

[6.4 类目管理实现及展示 15](#_Toc35158481)

[6.5 文章管理实现及展示 15](#_Toc35158482)

[6.6 用户管理实现及展示 16](#_Toc35158483)

[6.7 主页显示实现及展示 16](#_Toc35158484)

[6.8 用户注册登录实现及展示 16](#_Toc35158485)

[6.9 列表页实现及展示 16](#_Toc35158486)

[6.10 列表页实现及展示 16](#_Toc35158487)

[6.11 错误提醒实现及展示 16](#_Toc35158488)

1. **系统概述**
   1. **项目背景**

blog的英文全称应该是名词weblog,。weblog,其实主要是由动词web和weblog两个词组成。web,当然也是用来指一个互连网了,log的原来的英文意思应该是"航海日志",现在则是指任何互联网类型的日常流水记录。通俗的意义上来讲,weblog就是在社交网络上的一种日常流水记录的形式,也能直接可以说明它是一种"网络日志"。那些在网络上习惯于日常流水记录并热衷于使用这种weblog社交网络工具的年轻人被统一称为weblogger或动词weblogger。虽然对此这个概念的中文译名五花八门,但现在已基本可以统一将其译名为"博客"一个名词。其实该词最早可能是著名的社交网络技术评论家王俊秀和方兴东一起在网上撰文评论时提出来的,时间大概应该是在2002年8月8日左右。不管我们是在网络上称其为社交博客还是称其为网志,它的英文本义还是动词weblog。不过,生活中我们所说的"博客",既能直接说明它是一个名词blogger或动词weblogeer.

如果一定要说清楚一个Blog到底是什么，其实也很简单，日常生活中的一个简简单单的网页我们就能称其为一个Blog，它的组成也很简单，主要有一些帖子，这些帖子有的简短，有的生涩难懂，有的按照评论量排列，有的按照浏览量排列，当然也有的按照发表时间排列。Blog的内容丰富多彩，有对各种影视资源的分享，也有对各种学习资源的分享，还有一些是一些心得体会的分享文章，既有娱乐，又有能帮助人学习。Blog的目的也是多种多样，有的是为了商业化赚取钱财，有的只是为了记录自己的日常生活。发布这些Blog的人就叫做 Blogger，也可以称其为 Blog writer。

博客有自己的精神：比方说自由开放的精神，彰显博客的个性精神,包容的精神,还有博客的专业的精神。在自己的博客没有真正出现之前,大部分的人都会经常通过博客进出自己的论坛或者发表新的贴子,也有的一部分人通过即时语音通信的软件和朋友聊天,只是为了交流来通过博客表达自己的意见或者想法,可是这些都实际上是不成体系的。因为博客的横空出世，人们可以把自己的所思所想，一些突然出现的灵感都记录在自己的博客上。当人们在写文章的时候，有很大机率又会有一些新的想法产生。虽然有的想法只是思考一些简单的问题， 想的并不深远，但是对于当时所学的东西还是会有很大帮助，自己也会感觉到一种满足感，那种想法得以释放的满足感，思考得以延伸的满足感是在聊天的时候得不到的。生活中每天都会有形形色色的事情，有的时候只是一个小事都可能引起人们的思考，比方说只是一个和朋友的简单的聊天，就有可能会引起 人们的思考。博客可以督促自己努力、把自己想留住的东西记录下来，争取在平凡的生活里留下点有意义的东西，这是大部分人们写博客的主要目的。光阴似箭，日月如梭，写博客可以帮助我们记录我们生活的点点滴滴，也可以帮助我们去对一些问题进行深刻的思考。

* 1. **系统开发的意义**

博客给人们表达自己的心声提供了一个广阔的舞台。 在这里你可以收集到任何感兴趣的东西，也可以把自己搜集到的知识拿出来分享给大家 -- 可以是对一些时事政治的评论、 也可以是自己记录美好的一天的日记，甚至也可以是您感觉不错的一些网站的链接。大部分人使用博客只是为了表达自己的想法，而有一些人则是为了发表自己的看法， 还有的人只是为了在全球成千上万的浏览者中赢得粉丝，从而增加自己的浏览量，进而提高自己的威信。不管是普通老百姓，还是衣着光鲜的上班族，只要注册一个博客账号，就能发布自己想发布的任何东西。而对于那些有写日记习惯的人，在博客中宣示自己内心的想法则是他们最好的选择。而对于那些喜欢创作的人，在博客系统管理自己的作品成为了他们的第一选择，不管您要发表的意见是什么，博客都可以轻松地帮您表达出来。

博客作为当代越来越流行的一种表达人们想法的方式，它不但能传播作者的情绪，还把作者大量的智慧、看法和思想传递给来浏览的人们。从某种客观意义上来说，博客还是一种新的文化现象，随着博客的出现和繁荣，人们渐渐开始使用博客，网络的知识价值才通过博客真正凸显出来，互联网的发展也因为博客的流行而进入高速发展的阶段。

* 1. **系统开发的目标**

随着使用Blog的人越来越多，Blog开始以一种崭新的生活、工作和学习方式出现在的大众人民以及各行各业人们的视野，现在越来越多的人开始逐渐接受它，不仅如此，Blog还在悄悄地地改变传统的网络结构以及社会结构。也是因为有博客的存在，人们的交流和沟通变得越来越方便了；博客使得每个人都能发表自己的个人观点，因此博客的发布者可以把广大人民群众的智慧凝聚起来，从而解决一个凭自己一个人无法解决的问题；个人出版再也不是人们的梦想，人人都可以轻松地出版自己的“书籍”。用一句简单明了的话来说，Blog正在潜移默化地影响和改变着我们的生活。

如果一定要说清楚一个Blog到底是什么，其实也很简单，日常生活中的一个简简单单的网页我们就能称其为一个Blog，它的组成也很简单，主要有一些帖子，这些帖子有的简短，有的生涩难懂，有的按照评论量排列，有的按照浏览量排列，当然也有的按照发表时间排列。Blog的内容丰富多彩，有对各种影视资源的分享，也有对各种学习资源的分享，还有一些是一些心得体会的分享文章，既有娱乐，又有能帮助人学习。Blog的目的也是多种多样，有的是为了商业化赚取钱财，有的只是为了记录自己的日常生活。发布这些Blog的人就叫做 Blogger，也可以称其为 Blog writer。博客这种交流和表达方式之所以在生活中变得越来越流行，绝大部分是因为博客能收容这些不拘于个人的思想表达。博客除了具有个人网站的自由精神以为，而且更重要的是开拓了激发创造的新模式，这样就能让人们张扬自己的个性，从这个意义上说，博客将会变得越来越大众化，越来越多的人会慢慢地接受它。

本系统是用 Node.js 实现的一个基于浏览器的博客系统：主要由Node.js的Koa框架技术和Router路由技术以及Koa的各种中间件技术编写而成，这个系统的主要作用是用于资源分享，管理员的账号密码通过Md5加密之后放在服务器，登录管理员账号之后可以发布资源文章，更改前台banner的显示内容，banner就是前台的一个广告栏，可以更改资源的类目，也可以管理注册用户的信息。注册用户登录这个系统之后可以使用系统的检索功能，也可以对管理员发布的文章进行评论，当然也可以浏览文章，可以下载资源。游客用户只能浏览文章，不能搜索资源，也不能发表评论。

管理员拥有以下功能：

( 1 ) 登录搏客系统的后台管理平台。

( 2 ) 用户信息管理

l

( 3 ) 博客资源管理

( 4 ) 广告管理

( 5 ) 评论管理

( 6 ) 资源分类管理

注册用户有以下功能：

( 1 ) 浏览博客系统中的文章

( 2 ) 发表评论

( 3 ) 搜索博客资源

( 4 ) 下载资源

游客用户有以下功能：

( 1 ) 浏览博客系统中的文章

1. **博客系统相关技术**
   1. **前端基础Html5**

HTML5指的是超文本标记语言,是一种用来描述网页的语言, 它不能算是一种编程语言，只是一种标记语言。那么什么是标记语言呢？其实很简单，标记语言就是一套标记标签。网页是由网页元素组成的 ，这些元素是利用html标签描述出来，然后通过浏览器解析，就可以显示给用户了。就跟我们日常生活的书信要遵循共同的约定一样， HTML 有自己的语言语法骨架格式：

```html

<html>

<head>

<title></title>

</head>

<body>

</body>

</html>

```

#### html骨架标签总结： <html></html> ，##简称HTML标签，也是页面中最大的标签##，我们经常称其为根标签 ； |<head></head> ，这个标签表示文档的头部 ，有一点需要注意的是在head标签中我们必须要设置title标签；

| <titile></title> ，这个标签表示文档的标题，主要作用就是让页面拥有一个属于自己的网页标题

<body></body> ，这个标签表示的是文档的主体，body里面放的是这个文档的所有内容，页面内容

<!DOCTYPE> 主要用于声明，位于文档中的最前面的位置，放在 <html> 标签之前。此标签用于告知浏览器文档使用哪种 HTML 或 XHTML 规范。

互联网 Web 标准构成主要包括结构（Structure）、表现（Presentation）和行为（Behavior）三个方面；结构：结构用于对\*\*网页元素\*\*进行整理和分类，咱们主要学的是HTML； 表现 ：表现用于设置网页元素的版式、颜色、大小等\*\*外观样式\*\*； 行为 ：行为是指网页模型的定义及\*\*交互\*\*的编写，而HTML5实现的就是结构这一层。

* 1. **前端基础CSS3**

说起CSS出现的原因，绝大部分是因为HTML的局限性。说起HTML，这其实是个非常单纯的家伙，他只关注内容的语义， 比如<h1>表明这是一个大标题，用<p> 表明这是一个段落，用<img> 表明这儿有一个图片， 用<a> 表示此处有链接。很早的时候，世界上的网站虽然很多，但是他们都有一个共同的特点： \*\*丑\*\*。有些人就忍受不了了，你就不能把自己打扮得漂亮一点吗？HTML说，我试试...。如果要改变下 高度或者变一个颜色，就需要大量重复操作：

<img src="media/css.png" width="400" />

HTML满足不了设计者的需求，操作html属性不方便， HTML里面添加样式带来的是无尽的臃肿和繁琐。所以才有了CSS 的横空出世，CSS就是网页的美容师，让我们的网页更加丰富多彩，布局更加灵活自如。CSS的最大贡献就是：让 HTML 从样式中脱离，实现了 HTML 专注去做结构呈现，样式交给css 。我们理想中的结果就是结构(html)与样式(css)相分离。而且CSS 做的很出色，如果JavaScript是网页的魔法师，那么CSS它是我们网页的美容师。CSS(Cascading Style Sheets) ，通常称为CSS样式表或层叠样式表（级联样式表）， 主要用于\*\*设置\*\* HTML页面中的文本内容（字体、大小、对齐方式等）、图片的外形（宽高、边框样式、边距等）以及\*\*版面的布局和外观显示样式。CSS以HTML为基础，提供了丰富的功能，如字体、颜色、背景的控制及整体排版等，而且还可以针对不同的浏览器设置不同的样式。

要是有CSS就要引入CSS样式表（书写位置），要书写css样式，那css样式书写的位置在哪呢？其实这个也很简单：

1. 行内式（内联样式）：也称行内样式、行间样式，是通过标签的style属性来设置元素的样式，其基本语法格式如下：

```html

<标签名 style="属性1:属性值1; 属性2:属性值2; 属性3:属性值3;"> 内容 </标签名>

```

实际上任何HTML标签都拥有style属性，用来设置行内式。案例：

~~~css

<div style="color: red; font-size: 12px;">青春不常在，抓紧谈恋爱</div>

使用时应当注意：

- style其实就是标签的属性

- 样式属性和值中间是`:`

- 多组属性值之间用`;`隔开。

- 只能控制当前的标签和以及嵌套在其中的字标签，造成代码冗余

当然这么做的话有一个明显的缺点：没有实现样式和结构相分离。

2． 内部样式表（内嵌样式表）：也称内嵌式，就是将CSS代码集中写在HTML文档的head头部标签中，并且用style标签定义

其基本语法格式如下：

```html

<head>

<style type="text/CSS">

选择器（选择的标签） {

属性1: 属性值1;

属性2: 属性值2;

属性3: 属性值3;

}

</style>

</head>

```

~~~css

<style>

div {

color: red;

font-size: 12px;

}

</style>

~~~

使用时应当注意：

- style标签一般位于head标签中，当然理论上他可以放在HTML文档的任何地方。

- type="text/css" 在html5中可以省略。

- 只能控制当前的页面

当然怎么做还是有一个缺点，还是没有彻底分离

3. 外部样式表（外链式）：又称链入式，是将所有的样式放在一个或多个以\*\*.CSS\*\*为扩展名的外部样式表文件中，​ 通过link标签将外部样式表文件链接到HTML文档中

- 其基本语法格式如下：

```html

<head>

<link rel="stylesheet" type="text/css" href="css文件路径">

</head>

```

- 注意：

- link 是个单标签

- link标签需要放在head头部标签中，并且指定link标签的三个属性

| 属性 | 作用 |

| ---- | :----------------------------------------------------------- |

| rel | 定义当前文档与被链接文档之间的关系，在这里需要指定为“stylesheet”，表示被链接的文档是一个样式表文件。 |

| type | 定义所链接文档的类型，在这里需要指定为“text/CSS”，表示链接的外部文件为CSS样式表。我们都可以省略 |

| href | 定义所链接外部样式表文件的URL，可以是相对路径，也可以是绝对路径。 |

<img src='./media/10css外链式.jpg'>

最后说一下三种样式表区别（位置）：

| 样式表 | 优点 | 缺点 | 使用情况 | 控制范围 |

| ----- | ------------ | ------------ | ------- | --------- |

| 行内样式表 | 书写方便，权重高 | 没有实现样式和结构相分离 | 较少 | 控制一个标签（少） |

| 内部样式表 | 部分结构和样式相分离 | 没有彻底分离 | 较多 | 控制一个页面（中） |

| 外部样式表 | 完全实现结构和样式相分离 | 需要引入 | 最多，强烈推荐 | 控制整个站点（多） |

* 1. **前端基础JavaScript**

首先我们要弄明白JavaScript是什么： JavaScript 是世界上最流行的语言之一，是一种运行在客户端的脚本语言 （Script就是是脚本的意思）：脚本语言：不需要编译，运行过程中由 js 解释器( js 引擎）逐行来进行解释并执行，现在也可以基于 Node.js 技术进行服务器端编程

JavaScript的作用也很简单：

（1） 表单动态校验（密码强度检测）（ JS 产生最初的目的 ）

（2）网页特效

（3）服务端开发(Node.js)

（4）桌面程序(Electron)

（5）App(Cordova)

（6）控制硬件-物联网(Ruff)

（7）游戏开发(cocos2d-js)

浏览器执行JS的执行过程也很简单，浏览器分成两部分：渲染引擎和 JS 引擎，浏览器本身并不会执行JS代码，而是通过内置 JavaScript 引擎(解释器) 来执行 JS 代码 。JS 引擎执行代码时逐行解释每一句源码（转换为机器语言），然后由计算机去执行，所以 JavaScript 语言归为脚本语言，会逐行解释执行。

JS 主要由三部分组成：

1. \*\*ECMAScript\*\*，ECMAScript 是由ECMA 国际（ 原欧洲计算机制造商协会）进行标准化的一门编程语言，这种语言在万维网上应用广泛，它往往被称为 JavaScript或 JScript，但实际上后两者是 ECMAScript 语言的实现和扩展，ECMAScript：规定了JS的编程语法和基础核心知识，是所有浏览器厂商共同遵守的一套JS语法工业标准。

2. \*\*DOM——文档对象模型\*\*

​ \*\*文档对象模型\*\*（DocumentObject Model，简称DOM），是W3C组织推荐的处理可扩展标记语言的标准编程接口。通过 DOM 提供的接口可以对页面上的各种元素进行操作（大小、位置、颜色等）

3. \*\*BOM——浏览器对象模型\*\*

​ \*\*浏览器对象模型\*\*(Browser Object Model，简称BOM) 是指浏览器对象模型，它提供了独立于内容的、可以与浏览器窗口进行互动的对象结构。通过BOM可以操作浏览器窗口，比如弹出框、控制浏览器跳转、获取分辨率等。

JS 有3种书写位置，分别为行内、内嵌和外部：

1. 行内式

```html

<input type="button" value="点我试试" onclick="alert('Hello World')" />

```

- 可以将单行或少量 JS 代码写在HTML标签的事件属性中（以 on 开头的属性），如：onclick

- 注意单双引号的使用：在HTML中我们推荐使用双引号, JS 中我们推荐使用单引号

- 可读性差， 在html中编写JS大量代码时，不方便阅读；

- 引号易错，引号多层嵌套匹配时，非常容易弄混；

- 特殊情况下使用

2. 内嵌式

```html

<script>

alert('Hello World~!');

</script>

```

- 可以将多行JS代码写到 script 标签中

- 内嵌 JS 是学习时常用的方式

3. 外部JS文件

```html

<script src="my.js"></script>

```

- 利于HTML页面代码结构化，把大段 JS代码独立到 HTML 页面之外，既美观，也方便文件级别的复用

- 引用外部 JS文件的 script 标签中间不可以写代码

- 适合于JS 代码量比较大的情况

* 1. **Node.js技术**

学习Nodejs服务器端开发基础能够和后端程序员更加紧密的配合,网站业务逻辑前置，学习前端技术需要后 端技术支撑（Ajax）,还能扩宽知识视野，能够站在更高的角度审视整个项目。服务器端开发要做的事情主要是实现网站的业务逻辑和数据的增删改查。至于为什么选择Node，原因也很简单：

（1）可以使用JavaScript语法开发后端应用

一些公司要求前端工程师掌握Node开发生态系统活跃，

（3） 有大量开源库可以使用

（4）前端开发工具大多基于Node开发

Node.js的使用基本有如下几个步骤：

（1）电脑需要有Node环境，需要到Node官网下载Node.js,并安装到电脑

（2）初始化项目，在cmd 命令窗口下运行npm init -y 命令，会自动生成一个package.json文件，则表示初始化成功。

（3）使用npm命令安装各种第三方中间件，例如

npm i koa -D

其中-D表示将安装的中间件记录到package.json文件,当需要改变运行环境时，只需要在新环境的cmd窗口下运行npm I -y 就可以把本机安装的所有中间件都安装到新机器上。

在开发过程中主要使用了Node.js的Koa框架，Router路由技术，以及mysql，co-mysql，koa-session，koa-better-body，koa-static等第三方中间件技术，详细介绍如下：

（1） Koa框架：koa是目前Node.js的主流框架，主要优点就是支持独立路由，支持路由的嵌套，同时还支持大量第三方中间件的使用，极大地简化了开发地代码量。框架的使用很简单：

const Koa=require(‘koa’)

let server=new Koa();

server.listen(8080); let

（2） Koa-router：koa-router是koa 框架的一个独立的路由，因为koa-router独立于整个koa框架之外，所以很方便地实现了路由的嵌套。使用时需要引入该模块：

const Router=require(‘koa-router’);

路由的使用也很简单，主要有三个方法：

Let router=new Router();

router.get主要处理浏览器的get请求

router.post主要处理浏览器的post请求

router.all可以处理浏览器所有类别请求

（3）mysql中间件，主要处理与数据库有关的一些事情，使用方法如下：

const conn=require(‘mysql’);

conn.createPool();//创建数据库连接池

（4）co-mysql中间件，主要是包装mysql,实现数据库的异步操作：

const co=require(‘co-mysql’);

let con=co(conn);//包装mysql

(5)koa-session中间件，主要用于在服务端存储一些信息，比如用户密码等:

const session=require(‘session’)

server.keys=[];//存放一些字符串，用于验证session

server.use(session({ },server));

(6)koa-better-body中间件，主要用于文件上传：

Const body=require(‘koa-better-body’)

Server.use(body({

//在这指定上传路径

}));

(7)koa-static中间件，主要用于指定css,html等静态文件的存放路径

Const static=require(‘koa-static’)

Server.use(static());

1. **系统分析**
   1. **需求分析**

需求分析主要包括功能性需求分析和非功能性需求分析

功能性需求分析主要从两方面入手，第一个方面就是用户需要实现的功能，第二个方面就是管理员需要实现的功能。

用户需求分析：

（1）首先需要实现用户注册，登录功能。

（2）用户能发表文章，管理自己发表的文章，评论别人的文章，使用检索功能

管理员需求分析：

（1）管理员不能注册，管理员的账号密码存储在服务器

（2）管理员能管理注册用户的信息，管理资源分类，能修改删除所有的文章，能发表文章，能管理Banner

非功能性需求分析主要包括以下几个方面：

安全性：系统的管理员账户存储在服务端，采用md5加密技术，一般用户根本拿不到管理员的账户密码，无法管理后台，数据库采用Mysql,这是一种十分安全的数据库，所以总体来说，系统的安全性是没有什么问题的

健壮性：系统主要基于Node.js，这是一种十分成熟的系统，Node的第三方中间件十分强大，Node的开发者一直都在维护，并且第三方中间件的更新频率也十分高，在开发过程中完全遵照中间件开发者提供的使用说明进行一系列操作，所以我开发出来的这个系统能够十分稳定地运行。

易使用性：这个系统是一个web系统，用户主要在浏览器使用相关功能，可视化的界面给用户和管理员都带来了方便，使用起来就像我们平常生活中用电脑手机等电子设备浏览网页一样简单，所以系统的使用绝对不是问题。

可扩充性：系统主要采用Node开发的，其中主要用了Node的中间件技术，而在所有中间件中，路由的使用一直没减少过，所有功能都是基于路由技术实现的，如果以后想要添加什么新功能，或者想要删掉哪些功能，又或者想要修改哪些功能，只要修改相应的路由就可以了，所以说整个系统的可扩展性还是十分符合要求的，修改起来也十分的简单。

可重用性：整个系统采用Node开发，而模块化就是node一个十分显著的特征，代码都是在各个模块内部编写的，通过模块间的相互调用完成整个项目，所以说代码的重用率还是很高的，而且各个模块都有自己独特的功能，以后如果想要实现什么功能，只需要调用该模块即能完成该功能，所以说代码的可重用性还是十分高的。

* 1. **业务分析及流程图**

首先所有用户都能到达主界面，之后管理员可以登录，登录之后可以对系统进行管理操作，包括增加文章，管理注册用户信息，管理资源分类等，注册过的用户也可以登录，但是没有权限管理系统，只能评论管理员发布的文章，使用系统的搜索功能，下载管理员发布的资源；游客用户只能浏览文章，不能评论管理员发布的文章，也不能下载管理员发布的文章，也不能使用系统的检索功能。业务流程图如下图所示：

* 1. **数据流图分析**

在整个系统的运行过程当中，大概分为两个方向的数据流动，一个是管理员在后台界面实现管理功能时对数据库进行一系列操作而产生的数据流动，另一个是注册用户在前台界面使用系统时向数据库发出的数据请求以及数据库返回请求内容的数据流动。数据流图如下图所示：

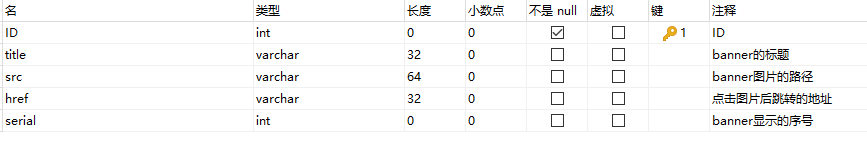
1. **数据库选择及设计**
   1. **数据库选择**

数据库的主要作用就是用来存储信息的，数据库的选择标准对于这个系统来说最重要的是安全性，其次就是操作简单。 数据库大概分为四种类型，第一种是文件型：这种数据库的优点是简单，缺点也是太过于简单，没有什么高级功能，适合用来存储手机短信之类的少量信息，不能存储大量信息，常见的有access, sqlite 等；第二种是关系型数据库：这种数据库是最常见的数据库，所谓的关系型数据库，与其名字一样，数据之间是有关系的，它的最突出的优点就是功能强大，可以满足日常所有稀奇古怪的需求，但是关系型数据库有一个小小的缺点，那就是它的性能不是最高的，MySQL,Oracle都是关系型数据库；第三种就是分布式数据库：分布式数据库可以把数据库安装在很多很多机器上，它的优点就是可以把各种请求，各种查询等乱七八糟的东西分散开 ，以此充分调动各个机器的资源，此外数据的安全性是比较高的，一般来讲数据都会做备份，如果有一台机器不能正常工作了，这时可以再开一个新机器，自动把数据同步过去，这样的话可以提供数据的安全性，mongoDB是我们见的比较多的一种分布式数据库；第四种数据库是NoSQL：这种数据库砍掉了许多复杂的查询，突出的优点就是性能比较高，常见的有memcache,redis等。

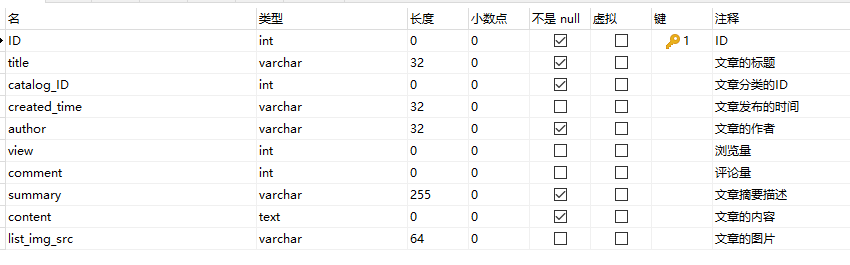
为什么会选择MySQL数据库？原因有两个，第一：它的应用是最广的，因为它是开源免费的，商业免费可以随便使用，所以很多公司会直接使用MySQL;第二：如果MySQL学会了，其实关系型数据库都差不多，会用一个其它也没什么区别。

* 1. **数据表设计**

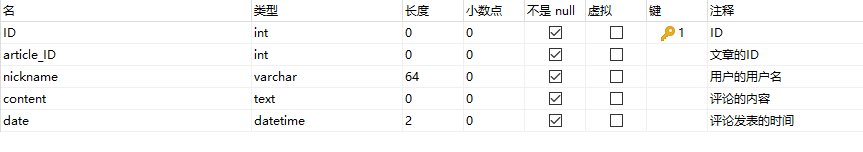
（1）Banner表，此表主要用于保存Banner相关的信息，具体结构如下图所示:



(2)文章信息表，此表主要用于保存文章的相关信息，具体结构如下图所示：



(3)评论信息表，此表主要用于保存评论的相关信息，具体结构如下图所示：



(4)用户信息表，此表主要用于保存用户的相关信息，具体结构如下图所示：



(5)类目信息表，此表主要用于保存文章分类的相关信息，具体结构如下图所示：



1. **系统功能模块设计**
   1. **主服务器模块**

该模块主要实现前台界面与后台数据库的数据传递，整个程序需要启动服务器才能运行，主要负责从数据库读取数据，获取前台表单数据，向数据库添加数据等一系列操作，是整个系统的核心，所有模块都在服务器运行的情况下才有意义。

* 1. **管理员登录模块**

该模块主要实现管理员的登录功能，管理员的账户密码都存储在服务端，密码采用md5加密技术，管理员没有注册功能，只有管理员才能登录后台管理系统。

* 1. **Banner管理模块**

该模块主要实现了对前台Banner的管理，管理员进入后台管理界面点击Banner管理，可以看到Banner相关的信息，界面主要显示Banner的ID，标题，图片，跳转链接，序号。图片用于在前台显示，当用户点击图片时，会自动跳转到跳转链接所指向的网页。管理员可以在这里添加，修改，删除Banner。

* 1. **文章管理模块**

该模块主要实现了管理员对文章信息的管理，当用户在管理界面点击文章管理按钮时，可以看到文章的相关信息，界面主要显示文章的标题，分类，图片，浏览量，评论量，作者，发表时间，描述，内容，下载链接。管理员可以添加新文章，也可以删除文章，还可以通过修改文章的评论量来改变文章在前台界面的排序，也可以修改资源的下载链接。

* 1. **类目管理模块**

该模块主要实现了管理员对文章类目信息的管理，当管理员在管理界面点击类目管理按钮时，可以看到类目的相关信息，界面主要显示类目的ID与类目名称。管理员可以添加新分类，也可以删除分类，还可以通过修改类目的名称。

* 1. **用户管理模块**

该模块主要实现了管理员对用户信息的管理，当管理员在管理界面点击用户管理按钮时，可以看到用户的相关信息，界面主要显示用户的用户名与密码。管理员可以添加新用户，也可以删除用户，还可以通过修改用户的用户名和密码。

* 1. **主页显示模块**

该模块主要用于显示系统的主页，主页的顶部显示用户的登录状态，主要用于给用户以及管理员提供登录入口；左侧部分主要由以下几部分组成：首先是一个Banner,用于发布一些广告之类的；Banner下面有一个公告栏，用于显示管理员发布的通知提醒；再下面是资源的分类列表，显示资源的类别；再往下是一个按照最新发布时间排序的文章列表，文章列表的每个文章需要在左侧显示文章图片，右侧显示文章的标题，浏览量，评论量，发布时间，文章概述。右侧主要是一个显示网站总资源数量以及网站运行总时间和站长联系方式的统计模块，下面是一个搜索框，主要用于实现搜索功能，再往下是一个文章显示列表，显示的文章按照评论量从高到低排序，每个文章显示图片，文章标题，发布时间，文章评论量；最后是站长提供的一个友情链接。

* 1. **用户注册登录模块**

该模块提供登录及注册功能，用户需要输入自己的账户密码，如果账户不存在，则提示用户账户不存在，如果用户输入的密码不对，则提醒用户密码不对，如果用户需要注册，则需要点击没有账号，请注册的超链接跳转到注册界面，在注册界面用户需要输入自己想要注册的用户名，还需要输入两次密码，如果用户的用户名已经存在，则提示用户账号已存在，如果用户输入的两次密码不一致，则提示用户两次输入的密码不一致。如果用户注册成功，则提醒用户注册成功，并给出请登录的超链接。

* 1. **列表页显示模块**

当在首页点击某个分类时跳转到该模块，该模块顶部提供一个返回首页的人口，左侧显示类别名称，下面是一个文章显示列表 ，文章按照发布时间排序，主要显示文章的图片，标题，发布时间，浏览量，评论量；右侧部分是一个搜索框，主要提供搜索功能，搜索框下面是一个统计信息显示，主要用于显示当前类别的文章总数，以及网站运行总时间；统计信息显示往下一个文章显示列表，显示文章的标题以及评论量和文章图片，文章按照评论量由高到低排序。

* 1. **文章详情显示模块**

该模块主要用于显示将文章展示出来，最上面提供文章的信息，包括标题，作者，发布时间，文章类别，浏览量，评论量，文章内容；往下提供一个跳转链接，主要用于跳转到其它分类列表；再往下是评论区，主要用于让用户发表评论，用户发表的评论会在下面按照时间排序显示；右侧主要是一个显示网站总资源数量以及网站运行总时间和站长联系方式的统计模块，下面是一个搜索框，主要用于实现搜索功能，再往下是一个文章显示列表，显示的文章按照评论量从高到低排序，每个文章显示图片，文章标题，发布时间，文章评论量。

* 1. **错误提醒模块**

该模块主要用于出错提醒，如果用户访问的页面不存在，或者访问的资源不错，就会提醒用户404错误，并提供一个返回首页的入口，防止服务器崩溃停止运行。

该模块主要用于出错提醒，如果用户访问的页面不存在，或者访问的资源不错，就会提醒用户404错误，并提供一个返回首页的入口，防止服务器崩溃停止运行。

1. **系统实现及展示**
   1. **主服务器的实现及展示**

服务器是前台与后台数据库沟通的桥梁，打开server文件夹，在Windows地址栏输入cmd进入命令行窗口，输入node server.js 命令启动服务器如图6.1所示；主要实现代码如图6.2所示；

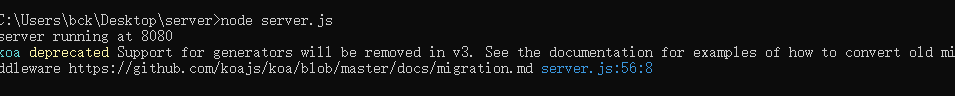


图6.1

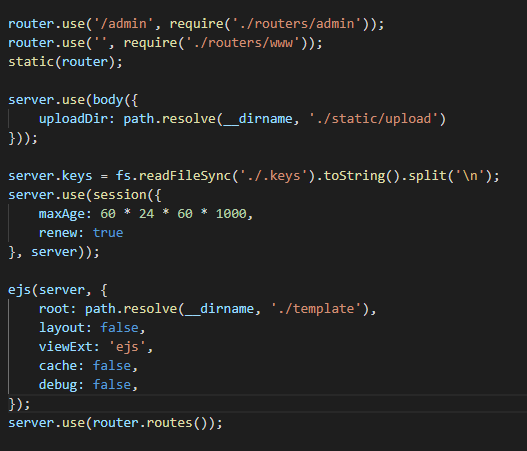


图6.2

* 1. **管理员登录实现及展示**

在浏览器输入<http://localhost:8080/admin/login>会出现登录界面如下图6.3所示； 当输入的用户名不存在时，会出现提示，如下图6.4所示；当输入的密码不正确时，如下图6.5所示；当用户名与密码都正确时会进入后台管理界面，如图6.6所示；实现的主要思路是管理员密码存放在服务端，密码经过md5加密；用户名密码正确时，会重定向到后台管理界面，用户名或密码存在一个不正确时会重定向到当前界面，并传递过来一个错误参数，在前台显示，主要由两个路由实现：

router.get('/login', async ctx => {…});//这个路由主要显示登录界面；

router.post('/login', async ctx => {…})这个路由处理用户提交过来的表单，检验用户名及密码

|  |  |  |
| --- | --- | --- |
|  |  |  |

图6.3 图6.4 图6.5

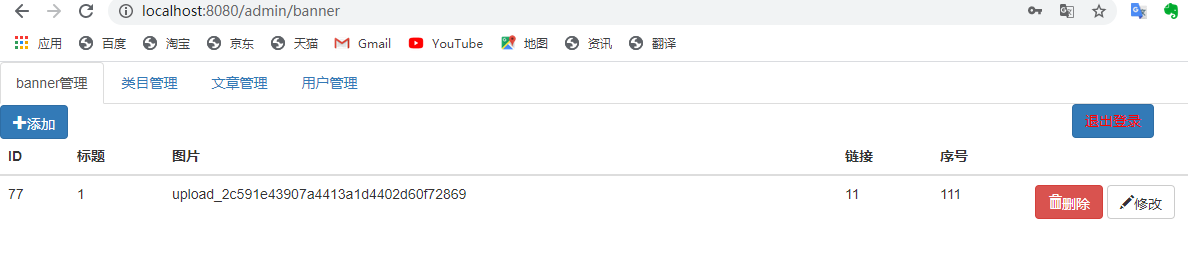


图6.6

* 1. **Banner管理实现及展示**

在banner管理界面可以管理banner相关信息，点击添加按钮时如图6.7；点击修改按钮时如图6.8，表单显示原来的数

据；点击删除按钮时删除该条信息；该模块主要通过以下个五个路由实现：

router.get('/banner', async(ctx, next) => {…})//这个路由主要用于显示Banner页面；

router.post('/banner', async ctx => {…})//这个路由主要处理添加信息时前台提交过来的数据；

router.get('/banner/delete/:id', async ctx => {…})//这个路由主要实现了删除功能；

router.get('/banner/modify/:id', async ctx => {…})//这个路由主要把修改界面呈现在前台；

router.post('/banner/modify/:id', async ctx => {…})//这个路由主要处理前台修改过的表单，实现修改功能；



图6.7

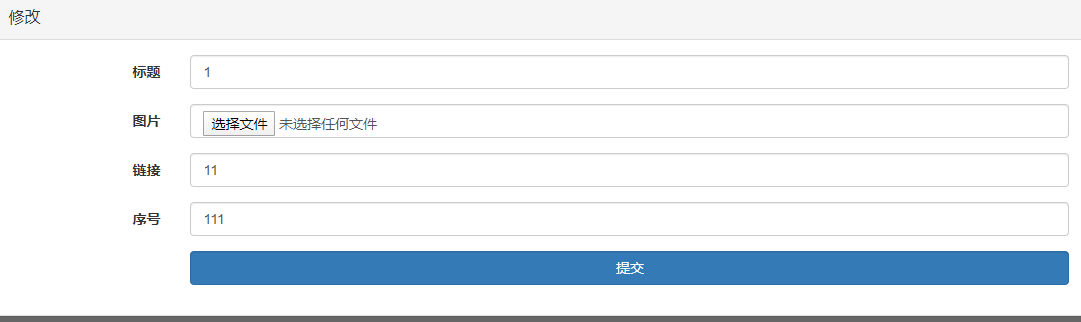


图6.8

* 1. **类目管理实现及展示**

在类目管理界面可以管理类目相关信息，点击添加按钮时如图6.9；点击修改按钮时如图6.10，表单显示原来的数据；点

击删除按钮时删除该条信息；该模块主要通过以下五个路由实现：

router.get('/catalog', async(ctx, next) => {…})//这个路由主要用于显示类目页面；

router.post('/catalog', async ctx => {…})//这个路由主要处理添加信息时前台提交过来的数据；

router.get('catalog/delete/:id', async ctx => {…})//这个路由主要实现了删除功能；

router.get('/catalog/modify/:id', async ctx => {…})//这个路由主要把修改界面呈现在前台；

router.post('/catalog/modify/:id', async ctx => {…})//这个路由主要处理前台修改过的表单，实现修改功能；

* 1. 文章管理实现及展示

在文章管理界面可以管理文章相关信息，点击添加按钮时如图6.11；点击修改按钮时如图6.12，表单显示原来的数据；点

击删除按钮时删除该条信息；该模块主要通过以下五个路由实现：

router.get('/article, async(ctx, next) => {…})//这个路由主要用于显示类目页面；

router.post('/article', async ctx => {…})//这个路由主要处理添加信息时前台提交过来的数据；

router.get('/article/delete/:id', async ctx => {…})//这个路由主要实现了删除功能；

router.get(' /article/modify/:id', async ctx => {…})//这个路由主要把修改界面呈现在前台；

router.post('/article/modify/:id', async ctx => {…})//这个路由主要处理前台修改过的表单，实现修改功能；

* 1. **用户管理实现及展示**

在用户管理界面可以管理文章相关信息，点击添加按钮时如图6.13；点击修改按钮时如图6.14，表单显示原来的数据；点

击删除按钮时删除该条信息；该模块主要通过以下五个路由实现：

router.get('/users async(ctx, next) => {…})//这个路由主要用于显示类目页面；

router.post('/users ', async ctx => {…})//这个路由主要处理添加信息时前台提交过来的数据；

router.get('/ users/delete/:id', async ctx => {…})//这个路由主要实现了删除功能；

router.get('/users/modify/:id', async ctx => {…})//这个路由主要把修改界面呈现在前台；

router.post('/users/modify/:id', async ctx => {…})//这个路由主要处理前台修改过的表单，实现修改功能；

* 1. **主页显示实现及展示**

在浏览器输入[http://localhost:8080](http://localhost:8080/admin/login)会出现主页界面如下图6.15所示，主页的显示主要有一个路由实现router.get('/', async ctx =>

{…});当用户在主页使用搜索功能时，会将请求发送到处理搜索功能的路由router.post('/search', async ctx => {…}),在这个路由会判断用户是

否有权限使用搜索功能，并将搜索的结果发送到前台。

* 1. **用户注册登录实现及展示**

这个模块主要实现了用户的登录注册功能，用户在首页点击登录时如图6.16，没有账号时点击注册超链接如图6.17，主要有四个路由

router.get('/login', async ctx => {…})//这个路由主要将登录界面展现到前台；

router.post('/login', async ctx => {…})//这个路由获取前台表单数据，判断用户名密码是否正确；

router.get('/register', async ctx => {…})这个路由主要将注册界面展现到前台

router.post('/register', async ctx => {…})这个路由主要获取注册表单的信息，判断数据是否合法，将数据增加到数据库

* 1. **列表页实现及展示**

当在首页点击某一个分类时会跳转到列表页，该页面如图6.18所示，当用户在该页面使用搜索功能时，会将请求发送到处理搜索功能的路由router.post('/search', async ctx => {…}),在这个路由会判断用户是否有权限使用搜索功能，并将搜索的结果发送到前台。

* 1. **列表页实现及展示**

当用户点击某一个列表的文章时会跳转到文章详情页面，如图6.19所示，当用户在该页面使用搜索功能时，会将请求发送到处理搜索功能的路由router.post('/search', async ctx => {…}),在这个路由会判断用户是否有权限使用搜索功能，并将搜索的结果发送到前台；当用户在该页面使用评论功能时如图6.20，会发送请求到处理评论功能的路由router.post ('/comment', async ctx => {…}),在这个路由会判断用户是否有权限使用评论功能，并将评论的结果按照评论时间展现在文章详情下的列表中，如图6.21.

* 1. **错误提醒实现及展示**

该模块主要用于出错提醒，如果用户访问的页面不存在，或者访问的资源不错，就会提醒用户404错误，并提供一个返回首页的入口如图6.22，防止服务器崩溃停止运行。错误提醒功能的实现关键就是在所有路由执行之前执行一个异步操作，主要代码如下所示

server.use(async(ctx, next) => {

const HTTP\_ROOT = ctx.config.HTTP\_ROOT;

try {

await next();//等待后面代码执行

//后面执行没错误

if (!ctx.body) {

//要找的页面不存在

await ctx.render('/www/404', {

HTTP\_ROOT,

})

}

} catch (e) {

//如果后面的路由执行有错误，就会抛出异常404

await ctx.render('/www/404', {

HTTP\_ROOT,

})

}

})

致谢

在杜莹老师的精心指导下完成了本次毕业设计， 她丰富的专业知识、博爱的胸怀、谦逊的品质和严谨细致、一丝不苟的作风是我工作、学习中的榜样；她循循善诱的教导和不拘一格的思路给予我无尽的启迪。通过这短短几个月的毕业设计环节， 我无论在基础理论知识、软件操作技能、还是独立思考问题能力方面都有了长足的进步，从开始的在做网站方面没有一点能力的我，到现在能做出一点成绩，多亏了老师的悉心教导。与此同时，杜莹老师还经常教育我一些为人处事的道理，使我在增长知识的同时也收获了许多人生哲理， 在此谨对杜莹老师致以衷心的感谢！ 本论文从选题、编程设计到毕业论文整个过程都倾注了杜莹老师的心血， 再一次向她表示衷心的感谢和诚挚的敬意！ 同时要感谢我的学校， 让我愉快而有意义的度过这四年的大学时光；感谢我的父母，他们给予了我极大的支持和鼓励；另外，还要感谢陪伴我共同走过大学时代并给予我帮助的老师、同学和朋友们，感谢你们，祝福你们在以后的日子里，开心、快乐！