摘 要

在互联网高速发展的时代,传统的学习技术难以满足客户远程学习和交流的需要,为网络用户提供简单高效的在线交流平台成为一项迫切任务。

本系统主要阐述了基于Web技术的博客系统的设计与实现。应用当前主流技术JavaScript的服务端运行平台Node.js；基于Node.js的应用开发框架Express；分布式文件存储的数据库MongoDB等技术设计一套支持网络用户发表文章与用户交流的在线博客系统。良好的实时交互能力在满足用户需求的同时，也使系统具有更好的响应速度。管理员在博客系统上发表的有价值的文章，供其他用户阅读，扩宽了用户的学习空间。同时也使系统更加轻量化，容易使用。Node.js社区拥有众多项目开发需要的资源，使开发更加快捷高效。

本系统前后端界面使用Bootstrap的响应式界面布局，其中前端使用Jade模板引擎来实现与后端数据的交互，后台使用Express的开发框架实现数据的调用，路由的控制。系统使用MongoDB来存储，管理博客中的数据。系统使用Yeoman脚手架工具来快速初始化项目，使用npm 包管理工具安装系统开发过程中需要的包。

**关键词：** Node.js； 博客系统； MongoDB

**Abstract**

This system mainly elaborates the design and realization of the blog system based on Web technology. In the era of rapid development of the Internet, the traditional learning technology is difficult to meet the needs of customers to learn and exchange for the network users to provide a simple and efficient online exchange platform has become an urgent task.

Application of the current technology mainstream JavaScript operating environment Node.js, based on Node.js application development framework Express, distributed file storage database MongoDB and other technologies to design a network of users to publish articles and user communication online blog system. Good real-time interactive ability to meet the needs of users at the same time, but also make the system more efficient. Users in the blog system published on the valuable articles for other users to read, widening the user's learning space. But also make the system more lightweight, easy to use. Node.js community has a lot of project development needs of the resources, is the development of more efficient and efficient.

The front and back interfaces of the system use the Bootstrap response interface layout, in which the front end uses the Jade template engine to realize the interaction with the backend data, and uses the Express development framework in the background to realize the data call and the route control. The system uses MongoDB to store and manage the data in the blog. The system uses the Yeoman scaffolding tool to initialize the project, using the npm package management tool to install the packages needed in the system development process.

**Keywords:** Node.js; blog system; MongoDB

目 录

[第1章 绪论 1](#_Toc20881)

[1.1系统开发背景 1](#_Toc13549)

[1.2系统研究目的和意义 1](#_Toc24176)

[1.3可行性分析 1](#_Toc9492)

[第2章 系统需求分析 3](#_Toc3539)

[2.1用户功能需求分析 3](#_Toc19201)

[2.2 系统性能要求 3](#_Toc25832)

[2.3业务流程分析 3](#_Toc15312)

[第3章 系统总体设计 5](#_Toc15989)

[3.1系统功能模块分析 5](#_Toc222)

[3.1.1 前台信息的展示 5](#_Toc31051)

[3.1.2 用户管理 5](#_Toc13562)

[3.1.3 后台信息管理 5](#_Toc10039)

[3.1.4 文章的浏览与查询 6](#_Toc25619)

[3.1.5 评论和点赞 6](#_Toc10361)

[3.2 实体联系图 6](#_Toc2373)

[3.3系统总体设计 8](#_Toc16405)

[第4章 系统详细设计 9](#_Toc9852)

[4.1系统开发环境 9](#_Toc20336)

[4.1.1 Node.js 9](#_Toc2566)

[4.1.2 MongoDB 9](#_Toc7196)

[4.1.3 Express开发框架 9](#_Toc14788)

[4.2 开发工具简介 9](#_Toc22720)

[4.2.1 Webstorm 9](#_Toc30942)

[4.2.2 Sublime 10](#_Toc12925)

[4.3 数据库设计 10](#_Toc17449)

[4.3.1用户信息 11](#_Toc28224)

[4.3.2分类信息 12](#_Toc17050)

[4.3.3文章信息 12](#_Toc15840)

[4.4用户界面设计 13](#_Toc21119)

[4.4.1系统主界面 13](#_Toc29438)

[4.4.2 评论文章页 14](#_Toc24919)

[4.4.3 登陆界面 16](#_Toc9722)

[4.4.4 系统管理界面 18](#_Toc27512)

[第5章 系统测试和调试 27](#_Toc22908)

[5.1白盒测试 27](#_Toc10730)

[5.2黑盒测试 29](#_Toc10082)

[第6章 工作总结和展望 31](#_Toc26658)

[6.1 工作总结 31](#_Toc10777)

[6.2 工作展望 31](#_Toc2175)

[参考文献 33](#_Toc11680)

[致 谢 34](#_Toc13745)

第1章 绪论

1.1系统开发背景

进入二十一世纪，计算机技术得到了飞速的发展特别是万维网的问世，打破了传统上人们对计算机网络的认识，各种网络交流互动的工具应运而生，其中以论坛，博客，社区等最受网民的欢迎，也是发展比较成熟的网络交流工具。随着网络技术的不断发展，互联网已经成为日常工作中不可缺少的工具，网络博客已经成为了各类网民展示自我，表达思想的强大工具，其强大的功能友好的互动性极大地满足了网民的需求。

1.2系统研究目的和意义

博客作为一种表达个人思想相互交流意见的平台，是网络世界的个人摘要，它重新定义了生活和工作方式，更重新定义了学习方式。它宣扬个人网站的自由精神，不断激发创造出新的交流模式，使其更具开放性和建设性，目的是要在网络世界体现个性，言论自由，传播文化，学习知识，拓展知识视野。当今越来越多的人注册成为各大博客网站的用户，在博客里自由发表个人经历，晒照片，展示技能，和更多人进行互动，相互学习。无形中，它已经成为我们生活不可或缺的交流工具。

如今博客在互连网世界中已经具有相当庞大的规模，在这里人们大胆想象未来的信息世界，在这里畅所欲言发表自己的观点与看法，同时这也是信息交流的集散地许多新的想法在这里产生。无疑博客这样一种影响力颇大的信息传播媒介将对好的思想，好的事物的传播具有强大的推动作用，有利于社会的进步，有利于大家学习新知识，探索新道路。所以我们要把这种影响力发挥到最大，努力提高开发博客的技术水平来获得更多人的认可。

1.3可行性分析

1.技术可行性。目前制作动态交互网站的Web应用技术主要有以下几种： Node.js的全栈开发以及Python的全栈网站开发。本系统是使用Node.js这一平台搭配相应的框架,后台数据库采用MongoDB Inc公司的Nosql型数据库MongoDB,开发的博客系统。

2.操作可行性。本博客系统采用B/S架构模式，用户可以通过浏览器来访问系统并且与服务器端进行交互。友好的界面设计对用户来说更容易使用，简单的操作方式可以轻松的满足用户的需求。

3.实现可行性。本博客系统使用Node.js的一系列开发技术。简约高效的前端界面设计对用户来说容易使用，搭配MongoDB的数据库服务模块对服务器端来说易于维护。由此可见本博客系统的开发切合实际是可行的。

第2章 系统需求分析

2.1用户功能需求分析

设计该博客系统时应该首先充分了解用户各个方面的需求，包括现有的以及将来可能增加的需求，本人通过对新浪博客、网易博客、博客园、天涯博客等站点进行测试与使用，了解了我国网民对于博客系统需求方面的一些现状以及网民、博客系统的管理人员对智能博客系统系统的各种要求，调研结果如下：

普通用户：不仅可以随时的浏览博客中发布的文章，发布时间，文章作者，分类，还能够根据用户的特殊需要，进行文章标题关键字的查询，查看某个分类下的所有文章，对自己感兴趣的文章点赞，添加评论等。

管理员：拥有系统的最高权限，可以对所有博客文章数据进行增加，删除和修改，添加评论，添加文章时对文章进行格式的调整，能对分类列表进行查看，编辑，删除和添加分类。

2.2 系统性能要求

1.安全性

该系统的安全性主要是防止他人非法进入系统，从而保证了数据的安全性，和数据的可靠性，同时设置加密技术形成双重保护。系统设置两种不同的身份进行操作，即管理员（注册用户）和普通用户（未注册）。

2.高效性

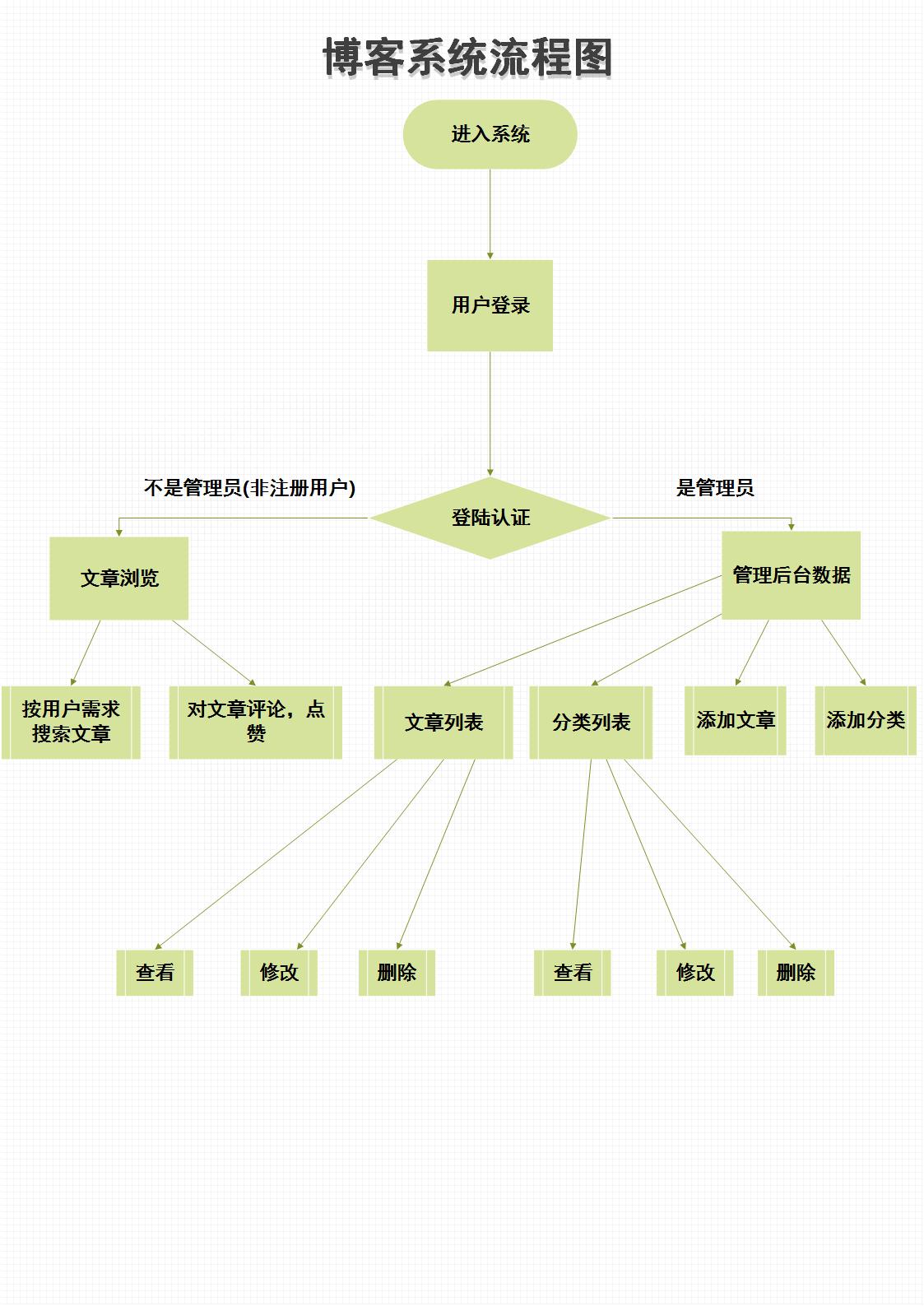
使用MongoDB数据库的优点，数据的查询功能更加快捷高效，能快速的根据查询条件对文章数据进行查询，排序。提高了响应速度。

3.便捷性

本系统的界面友好，使用Bootstrap前端库实现页面的响应式布局,使用font-awesome图标库添加个性化图标，操作方便，能够达到所见即所得的效果,同时采用B/S架构用户可以通过浏览器访问，方便用户的使用。

2.3业务流程分析

依据系统的需求分析和各功能之间的关系，得到系统的流程图如图2-1所示：



**图2-1 系统流程图**

第3章 系统总体设计

系统总体设计需完成系统的概念结构和逻辑结构设计，其中概念结构设计是将需求分析得到的用户需求抽象为概念模型的过程，它是整个数据库设计的关键。逻辑结构设计的任务是把概念结构设计阶段设计好的基本E-R图转化为与选用DBMS产品所支持的数据模型所符合的逻辑结构。

3.1系统功能模块分析

本系统主要分为前台文章信息的展示，管理员的注册，后台信息的管理，文章的浏览与查询，对文章进行点赞与评论等五个功能模块。

3.1.1 前台信息的展示

一般的浏览用户可以查看本站发布的文章的标题与摘要，对感兴趣的文章可以阅读全文，对喜爱的文章点赞，评论。在文章摘要也中可以清楚地看到每篇文章的点赞次数，评论次数，作者信息，发布时间等。

3.1.2 用户管理

一般访客不需注册就能对博客文章进行评论与点赞，注册成为管理员后拥有对文章数据进行增加，删除，编辑，查看的权限，可以发布新文章，对分类列表也拥有增加，删除，编辑，查看的权限，同时可以增加分类。

3.1.3 后台信息管理

文章列表：

1.查看所有文章信息；

2.博客文章的查看，修改，删除；

3.按照规则或关键字检索文章标题；

分类列表：

1.查看所有文章分类；

2.博客文章分类的查看，修改，删除。

添加文章：

1.添加文章标题，分类，内容；

2.对添加的内容进行格式调整。

添加分类：

1.为系统添加新的文章分类。

3.1.4 文章的浏览与查询

在前台页面中用户可以按照文章的分类查看相应分类下的所有文章，也可以通过分页器顺序查看所有文章。后台管理页面中管理员可以根据文章分类，作者进行单一的或是混合的筛选，也可以按分页器顺序查看。同时无论普通用户还是管理员都可以通过搜索框对文章标题进行关键字的检索，在后台管理页面中管理员还可以根据文章标题，作者，日期等对文章列表进行排序。

3.1.5 评论和点赞

在前台的文章详情页中，用户阅读完文章后可以对感兴趣的文章点赞，以及对文章进行评论，在后台的管理页面中可以看到每篇文章下的点赞次数，与评论内容。

3.2 实体联系图

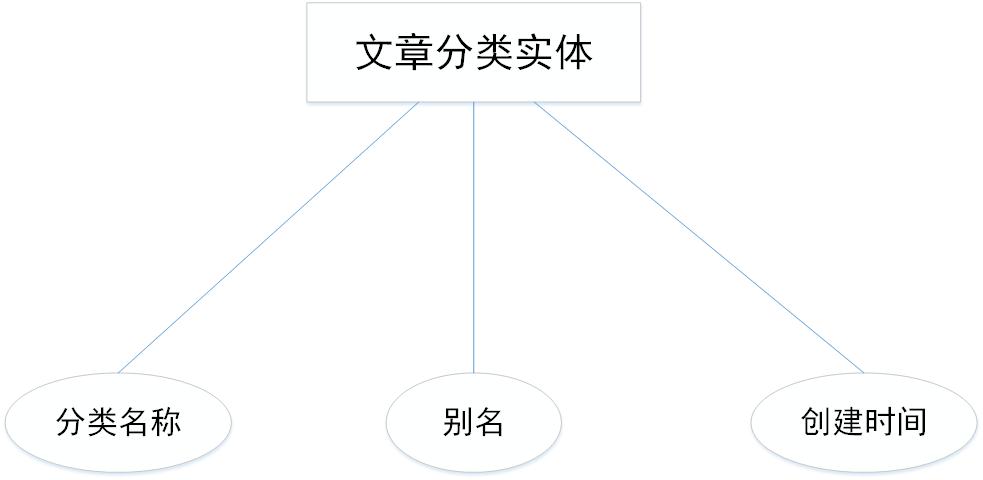
通过需求分析，现定义以下每个实体的属性。

文章分类表（分类名，创建时间）

文章内容表（文章标题，文章内容，文章分类，作者，是否发布，被赞，评论，创建时间）

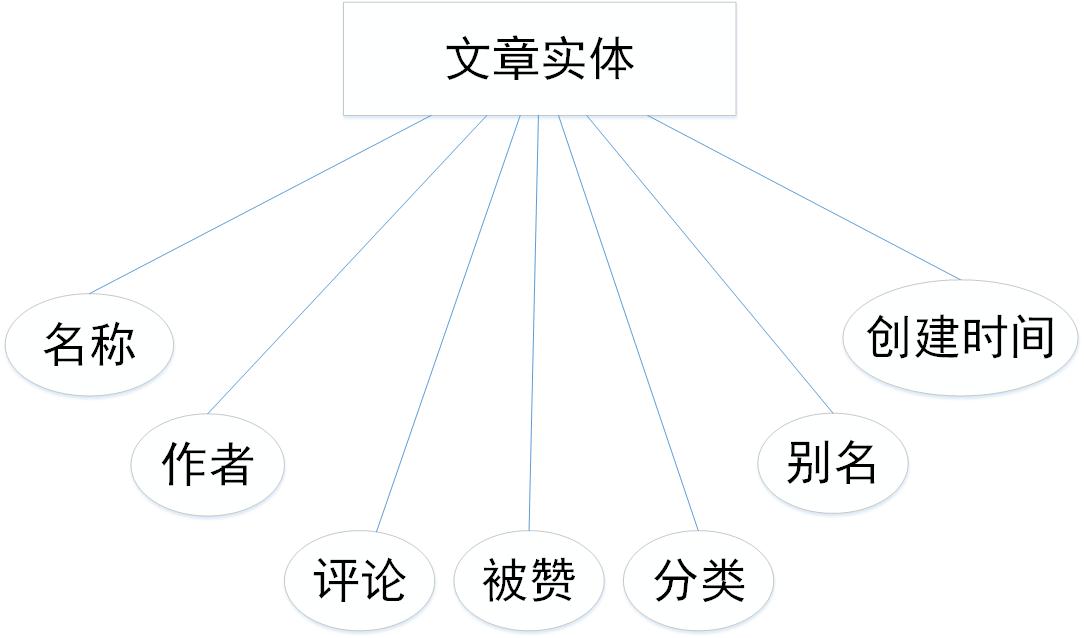
管理员信息表（用户名，邮箱，密码，注册时间）

1.文章分类实体属性如图3-1所示：



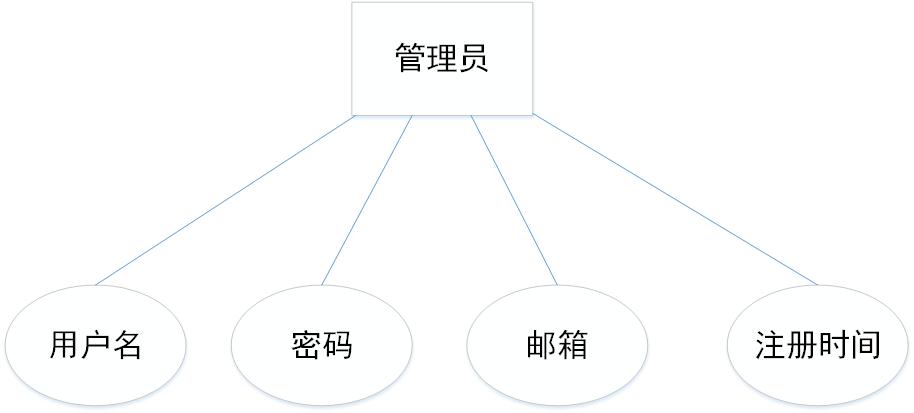
**图3-1 文章分类实体属性图**

2.文章内容实体属性如图3-2所示：



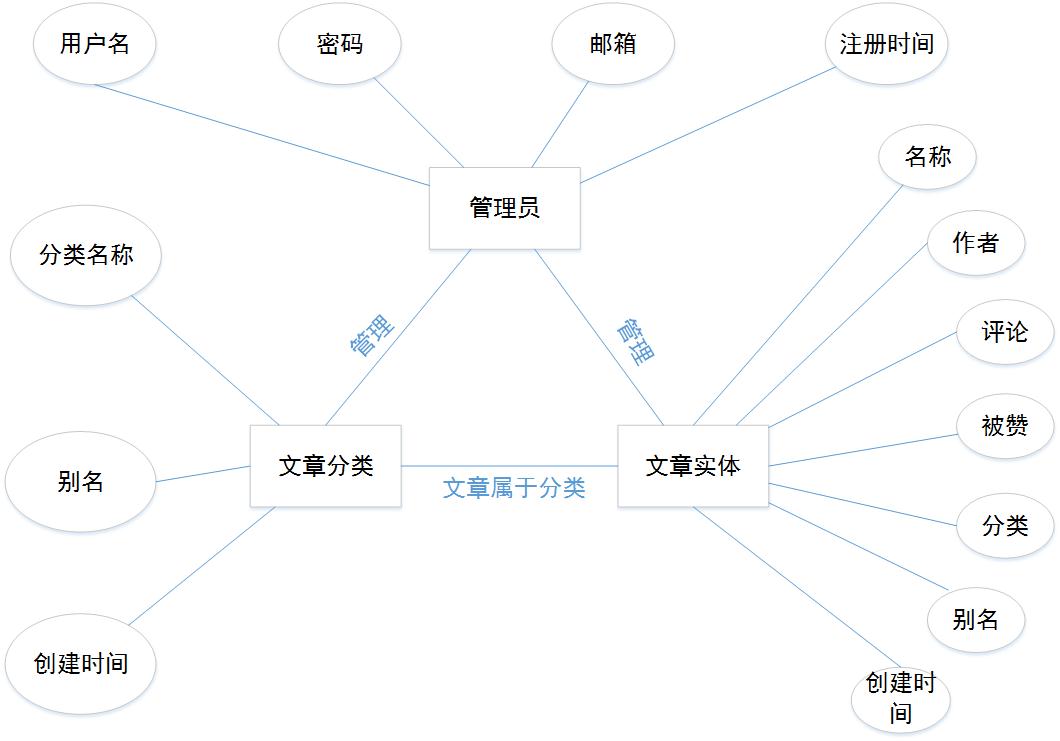
**图3-2 文章内容实体属性图**

3.管理员实体属性如图3-3所示：



**图3-3管理员实体属性图**

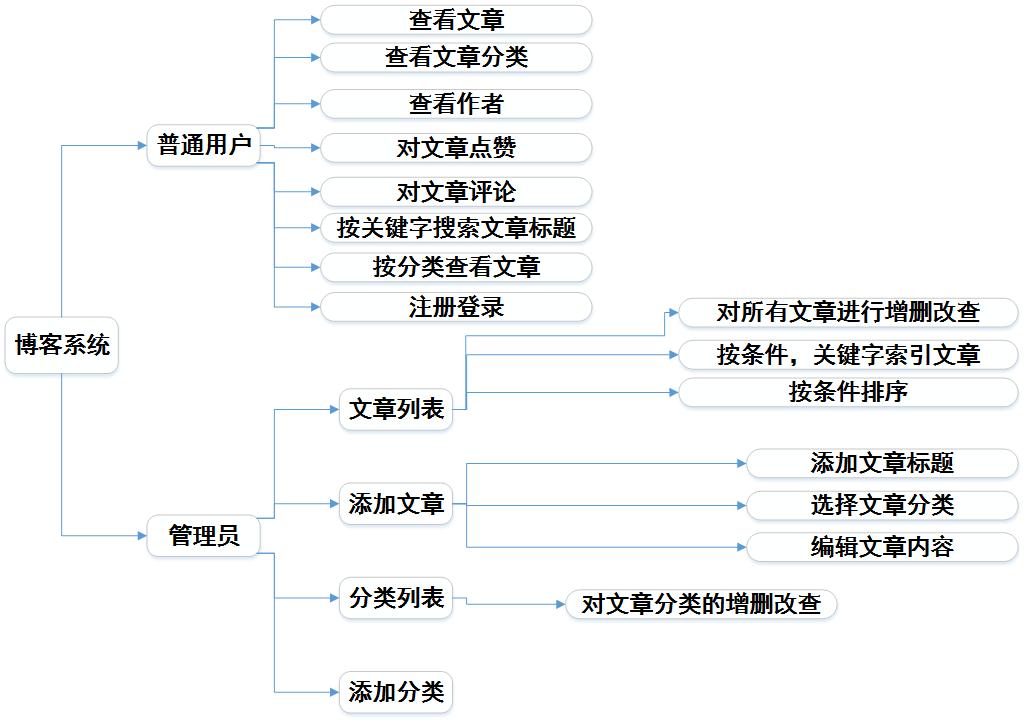
4.实体之间关系的E-R图如图3-4所示：

****

**图3-4 总体的实体属性图**

3.3系统总体设计

系统设置了两种身份的，包括普通用户、管理员。其中普通用户可以不需登录就能查看文章，查看文章分类，查看文章作者，对文章点赞，评论，按关键字搜索文章标题，按文章分类查看文章，注册登陆成为管理员。管理员则拥有最高的权限可以对文章和分类进行增删改查，按条件或关键字索引文章，按文章名称，发布时间，作者，分类，对文章列表进行升序或是降序排序，添加文章等操作，对分类进行增删改查。如图3－5就是系统的总体结构框图：



**图3-5 系统总体结构图**

第4章 系统详细设计

4.1系统开发环境

基于Node.js的全栈博客系统的开发需要Node.js的开发环境，MongoDB数据库，项目开发的框架Express，前端的开发引擎Jade，以及项目开发中需要的各种npm 包。

4.1.1 Node.js

Node.js不是一种独立的语言，也不是一个JavaScript框架，而是一个搭建在Chrome JavaScript运行时上的平台，Node.js采用的事件驱动，非阻塞I/O模型，使它既轻量又高效，并且成为构建运行在分布式设备上的数据密集型实时程序的完美选择。它实现了诸如文件系统统、模块、 包、操作系统 API、网络通信等。它在Github受关注项目排行榜上位列第二，并且社区中的用户们在npm包管理网站上发布了大量的模块，吸引了广大的程序员的关注。

4.1.2 MongoDB

MongoDB是一款强大，灵活，且易于扩展的通用性数据库，MongoDB是一个面向文档的数据库，而不是关系型数据库。不采用关系模型主要是为了获得更好的扩展型，与关系型数据库相比，面向文档的数据库不再有“行”的概念，取而代之的是更为灵活的“文档”模型，通过在文档中嵌入文档和数据，面向文档的方法能够仅使用一条数据来表现复杂的层次关系。

4.1.3 Express开发框架

精简的，灵活的Node.js Web程序框架，为构建单页，多页及混合的Web程序提供了一系列健壮的功能特性。Express强大的中间价机制搭配Node.js的平台是开发更加快捷。

4.2 开发工具简介

4.2.1 Webstorm

WebStorm：jetbrains公司开发一款智能的强大的JavaScript现代开发工具，被中国JS开发者誉为“Web前端开发神器”，HTML5的编辑器。WebStorm在支持JavaScript重构和单元测试的同时，还可以进行批量的代码分析。支持CSS、JavaScript，HTML5等多种语言的编辑。语言简洁地格式化输出，用户还可以根据自己的爱好定义规则，对于代码中出现的错误，WebStorm提供修复意见，是一个功能强大的软件。

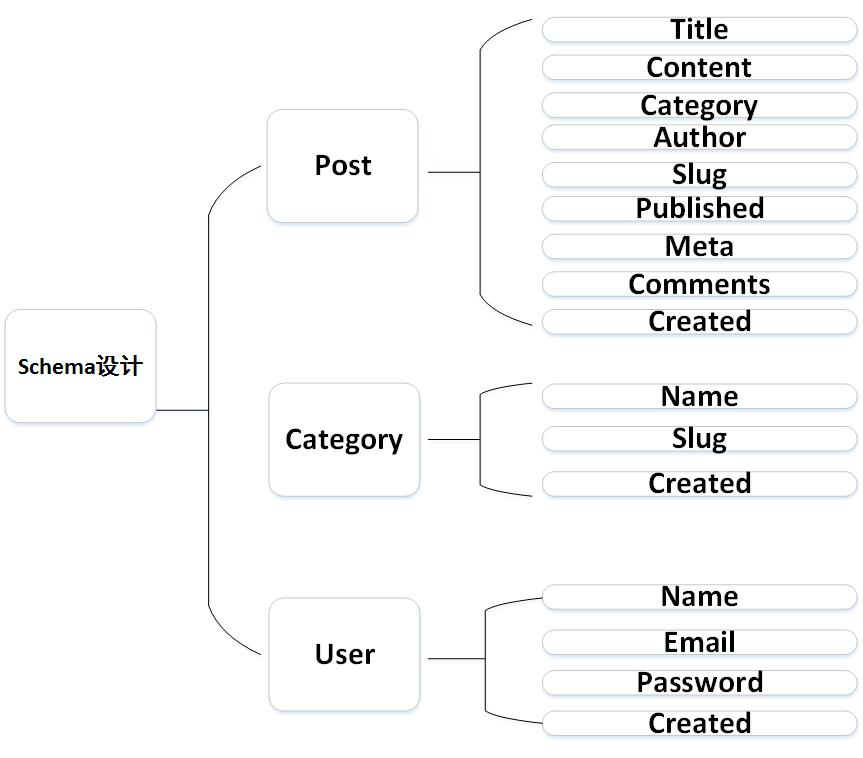
4.2.2 Sublime

Sublime Text：一款轻量级复杂的文本编辑器拥有非凡的功能和惊人的表现。它拥有代码缩略图，高级语言的插件，可操作的菜单栏和工具栏，它完美的操作界面和强大的功能让人称赞。支持在Windows,Linux等操作系统，进行多窗口切换，API查看等功能。众多的快捷键方便人们操作，加快了程序的开发速度。

4.3 数据库设计

对于一个信息管理系统来说数据库的设计占有非常重要的地位，数据库结构设计的好坏将直接影响应用系统处理数据的效率和方式。合理的数据库结构设计可以提高数据存储的效率，保证数据的完整和一致。

本系统数据库nodeblog的Schema设计：nodeblog数据库中存放三张表分别是Post文章信息表；Category文章分类表；User管理员信息表；相应的属性如图4-1所示



**图4-1 数据库的schema设计**

4.3.1用户信息

当前nodeblog数据库中注册的用户有：admin，123456，zhangyunlong，aaa；其中可以看出用户信息中的密码已经被Md5加密。如图4-2为系统中的管理员信息



**图4-2 用户信息表**

4.3.2分类信息

分类信息在创建数据时向categories表中添加四个分类信息（JavaScript，CSS，HTML，Tools），MongoDB为每条数据添加相应的“\_id”属性。如图4-3为文章的分类信息



**图4-3 分类信息表**

4.3.3文章信息

根据schema中的Post表的设计对Post表中的各个字段添加相应的数据，其中测试数据是通过Node.js的文件操作使用lorem-ipsum这个包按照分类批量生成的测试数据,每个分类下随机生成35篇文章，文章的标题，文章内容为随机数据。

4.4用户界面设计

4.4.1系统主界面



**图4-4 系统主界面**

/\*………………………………………………………………………………………

该页面为博客系统的首页用分页的形式展示了系统中所有的文章信息其中分页功能代码如下：

………………………………………………………………………………………\*/

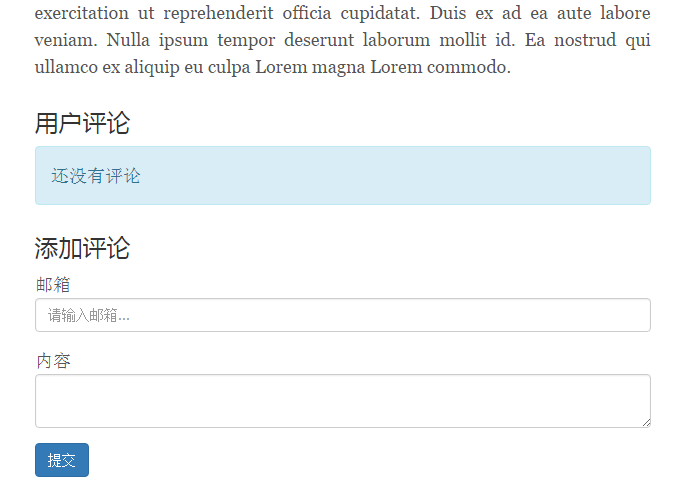
//实现分页功能的前端代码

if pageCount > 1  
 nav  
 ul.pagination  
 - var currentPage = 1  
 while currentPage < pageCount  
 if (currentPage === pageNum)  
 li.active  
 a(href="/posts?page=" + currentPage)= currentPage++  
 else  
 li  
 a(href="/posts?page=" + currentPage)= currentPage++

//实现分页功能的后端代码

var pageNum = Math.abs(parseInt(req.query.page || 1, 10));  
 var pageSize = 10;  
  
 var totalCount = posts.length;  
 var pageCount = Math.ceil(totalCount / pageSize);  
  
 if (pageNum > pageCount) {  
 pageNum = pageCount;  
 }  
  
 res.render('blog/index', {  
 title: 'Hello Nodeblog Home',  
 posts: posts.slice((pageNum - 1) \* pageSize,pageNum\*pageSize),  
 pageNum: pageNum,  
 pageCount: pageCount,  
 pretty:true  
});

4.4.2 评论文章页

****

**图4-5 评论文章界面**

/\*………………………………………………………………………………………

普通用户可以浏览博客中的所有文章并进行评论，用户在提交评论时需要填写邮箱和评论内容，评论内容会以提交的时间越晚内容就越靠前的方式显示，“提交”功能实现代码如下：

………………………………………………………………………………………\*/

//为文章添加评论

router.post('/comment/:id', function (req, res, next) {  
 if(!req.body.email){  
 return next(new Error('no email provided for commenter'))  
 }  
 if(!req.body.content){  
 return next(new Error('no content provided for commenter'))  
 }  
 //seo优化  
 var conditions={};  
 try{  
 conditions.\_id = mongoose.Types.ObjectId(req.params.id);  
 } catch(err){  
 conditions.slug = req.params.id;  
 })  
 //添加评论  
 Post.findOne(conditions).exec(function(err,post){  
 if (err){  
 return next(err);  
 }  
 var comment={  
 email: req.body.email,  
 content: req.body.content,  
 created: new Date(),  
 }  
 //unshift() 方法可向数组的开头添加一个或更多元素，并返回新的长度  
 post.comments.unshift(comment);  
 post.markModified('comments');  
 post.save(function(err,post){  
 req.flash('info','评论添加成功')  
 res.redirect('/posts/view/' + post.slug)  
 });  
 });  
});

4.4.3 登陆界面



**图4-6 登陆界面**

/\*………………………………………………………………………………………

成为管理员后拥有最高权限能修改文章数据，在注册阶段会对用户所输入的账户进行检查要符合邮箱格式要求，也不能为空，同时两次输入的密码也要相同。注册完成后会在用户表中生成相应的数据可以看到其密码部分是用md5加密过的。用户的登陆注册功能实现的代码如下：

………………………………………………………………………………………\*/

//用户登录

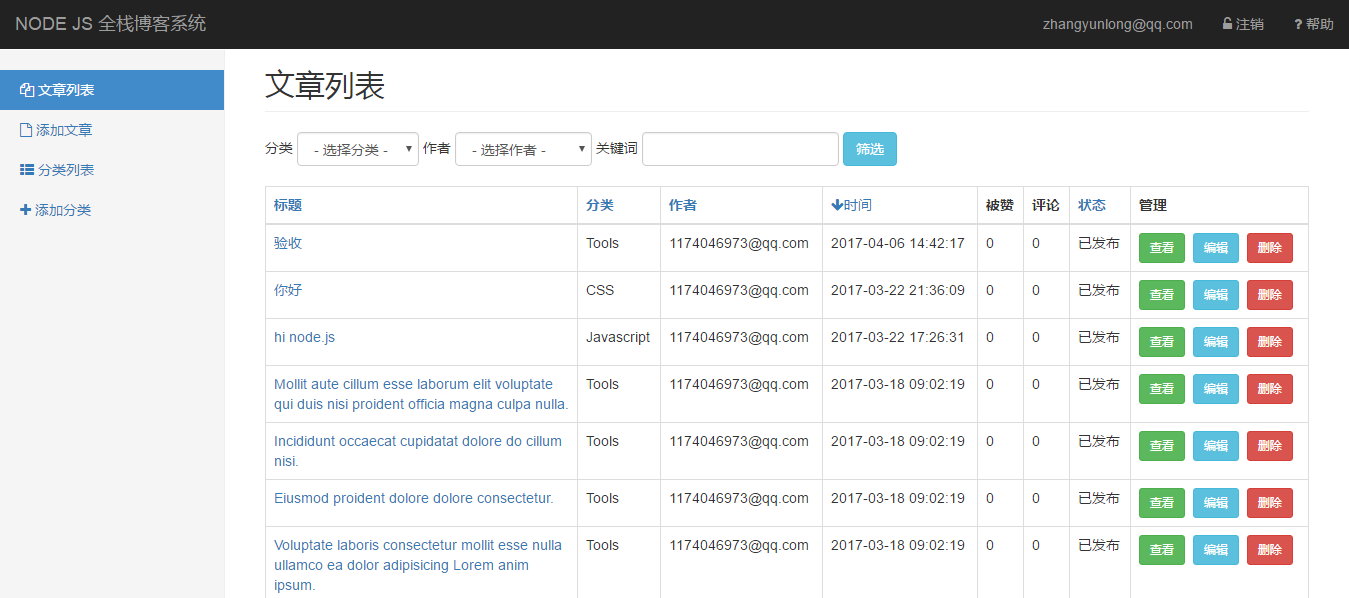
router.get('/login',function(req,res,next){  
 res.render('admin/user/login',{  
 pretty: true,  
 })  
});  
  
router.post('/login', passport.authenticate('local', {  
 failureRedirect: '/admin/users/login',  
 //passpost支持  
 failureFlash: '用户名或密码错误',  
 }),  
 function (req, res, next) {  
 console.log('user login success: ', req.body);  
 res.redirect('/admin/posts');  
 });

//用户注册

router.get('/register',function(req,res,next){  
 res.render('admin/user/register',{  
 pretty: true,  
 })  
});  
router.post('/register',function(req,res,next){  
 req.checkBody('email','邮箱不能为空').notEmpty().isEmail();  
 req.checkBody('password','密码不能为空').notEmpty();  
 req.checkBody('confirmPassord','两次密码不匹配').notEmpty().equals(req.body.password);  
 var errors= req.validationErrors();  
 if(errors){  
 console.log(errors);  
 return res.render('admin/user/register',req.body);  
 }  
 var user = new User({  
 name: req.body.email.split('@').shift(),  
 email: req.body.email,  
 password: md5(req.body.password),  
 created: new Date(),  
 });  
 user.save(function(err,user){  
 if(err){  
 console.log('/admin/user/register error:',err);  
 req.flash('error','用户注册失败');  
 res.render('admin/user/register');  
 }else{  
 req.flash('info','用户注册成功');  
 res.redirect('/admin/users/login');  
 }  
 })  
});

4.4.4 系统管理界面

1.文章列表



**图4-7 文章列表页**

/\*………………………………………………………………………………………

管理员登陆后能对文章进行查看，编辑，删除，可以按条件对文章进行筛选,排序。筛选文章和删除文章的功能代码如下：

………………………………………………………………………………………\*/

//按条件对文章进行筛选,排序

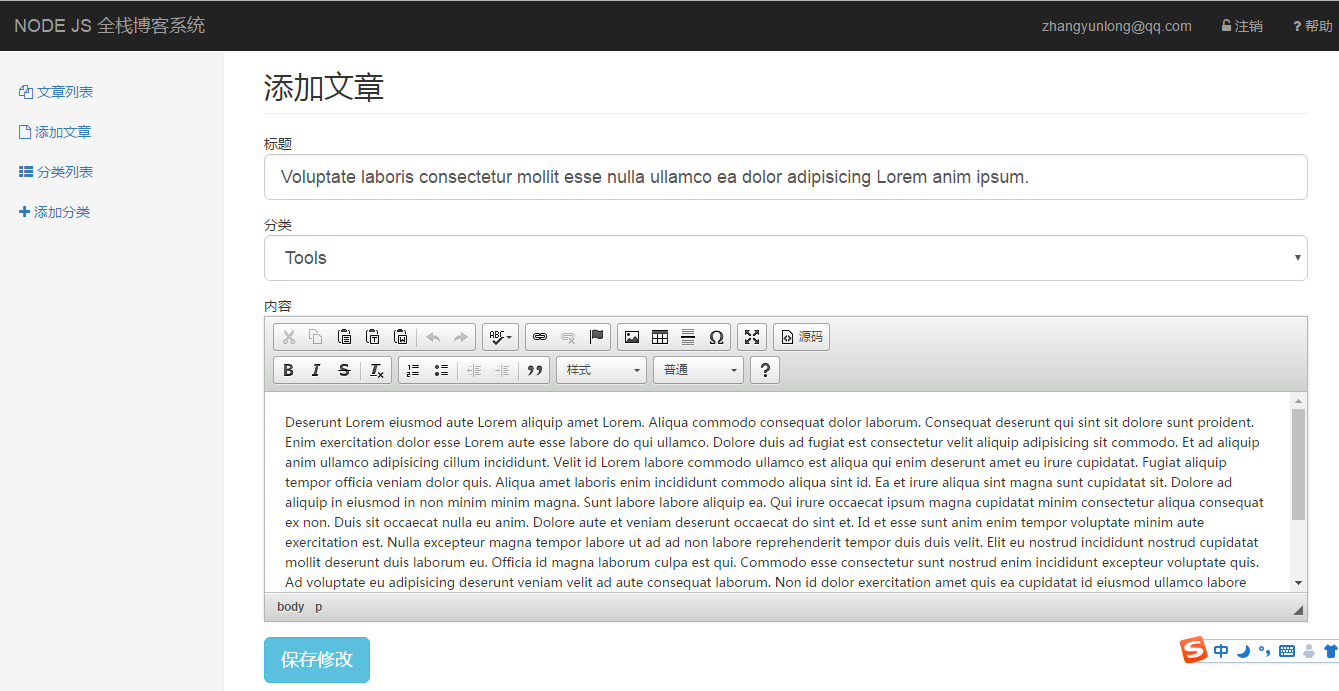
router.get('/', auth.requireLogin,function (req, res, next) {  
 //排序  
 var sortby = req.query.sortby ? req.query.sortby:'created';  
 var sortdir = req.query.sortdir ? req.query.sortdir:'desc';  
 if(['title','category','author','created','published']

.indexOf(sortby) === -1){  
 sortby = created;  
 }  
 if(['desc','asc'].indexOf(sortdir) === -1){  
 sortdir = 'desc';  
 }  
 var sortObj = {};  
 sortObj[sortby]=sortdir;  
//按照分类，作者检索文章列表  
var conditions = {};  
if (req.query.category){  
 conditions.category = req.query.category.trim();  
}  
if (req.query.author){  
 conditions.author = req.query.author.trim();  
}  
//关键字搜索  
if (req.query.keyword){  
 conditions.title = new RegExp(req.query.keyword.trim(),'i');  
 conditions.content = new RegExp(req.query.keyword.trim(),'i');  
}  
User.find({},function(err,authors){  
 if(err) return next(err);  
 Post.find(conditions)  
 .sort(sortObj)  
 .populate('author')  
 .populate('category')  
 .exec(function (err, posts) {  
 if (err) return next(err);  
 //翻页  
 var pageNum = Math.abs(parseInt(req.query.page || 1, 10));  
 var pageSize = 10;  
 var totalCount = posts.length;  
 var pageCount = Math.ceil(totalCount / pageSize);  
 if (pageNum > pageCount) {  
 pageNum = pageCount;  
 }  
 res.render('admin/post/index', {  
 title: 'Admin',  
 posts: posts.slice((pageNum - 1) \* pageSize,pageNum\*pageSize),  
 pageNum: pageNum,  
 pageCount: pageCount,  
 sortdir: sortdir,  
 sortby: sortby,  
 authors: authors,  
 pretty:true,  
 filter: {  
 category: req.query.category || "",  
 author: req.query.author || "",  
 keyword: req.query.keyword || "",  
 }  
 });  
 });  
 });  
});

//删除文章

router.get('/delete/:id', auth.requireLogin,function (req, res, next) {  
 if(!req.params.id){  
 return next(new Error('no post id provided'));  
 }  
 Post.remove({\_id:req.params.id}).exec(function(err,rowsRemoned){  
 if(err){  
 return next(err);  
 }  
 if(rowsRemoned){  
 req.flash('sucess','文章删除成功');  
 }else{  
 req.flash('sucess','文章删除失败');  
 }  
 res.redirect('/admin/posts');  
 });  
});

2.编辑文章信息



**图4-8 修改文章信息**

/\*………………………………………………………………………………………

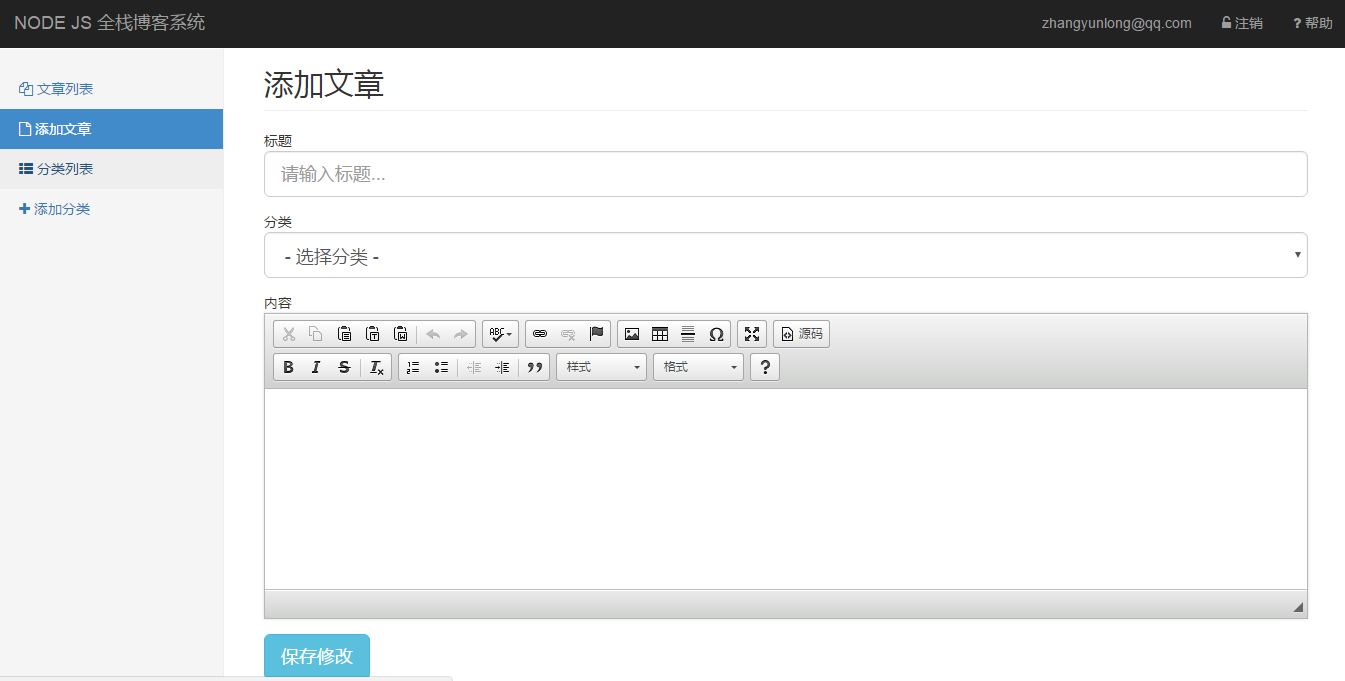
管理员登陆后能对文章标题，分类，内容进行修改，修改文章信息的功能代码如下：

………………………………………………………………………………………\*/

//编辑页  
router.get('/edit/:id',auth.requireLogin, getPostById,function (req, res, next) {  
 res.render('admin/post/add',{  
 action: "/admin/posts/edit/" + req.post.\_id,  
 post: req.post,  
 });  
});  
//编辑提交页  
router.post('/edit/:id',auth.requireLogin, getPostById, function (req, res, next) {  
 var post = req.post;  
 var title = req.body.title.trim();  
 var category = req.body.category.trim();  
 var content = req.body.content;  
 var py = pinyin(title,{  
 style: pinyin.STYLE\_NORMAL,  
 heteronym: false  
 }).map(function(item){  
 return item[0];  
 }).join(' ');  
 post.title = title;  
 post.catrgory = category;  
 post.content = content;  
 post.slug = slug(py);

//保存数据  
 post.save(function(err,post){  
 if (err){  
 req.flash('error','文章编辑失败');  
 res.redirect('/admin/posts/edit/'+ post.\_id);  
 }else{  
 req.flash('info','文章编辑成功');  
 res.redirect('/admin/posts');  
 }  
 });  
});

3.添加文章



**图4-9 添加文章页**

/\*………………………………………………………………………………………

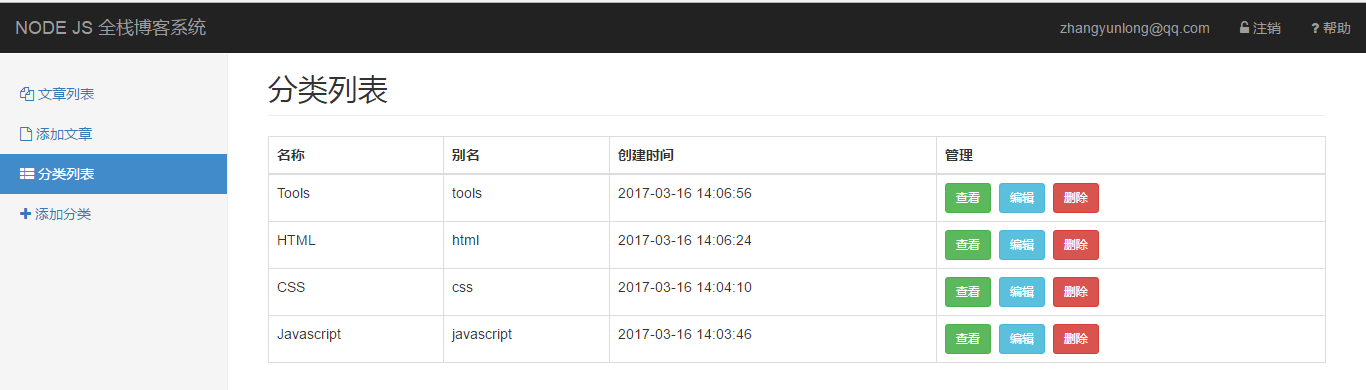
管理员可按照要求添加文章标题，选择文章分类，添加文章内容，具体代码如下：

………………………………………………………………………………………\*/

//添加文章

router.post('/add',auth.requireLogin, function (req, res, next) {  
 req.checkBody('title','文章标题不能为空').notEmpty();  
 req.checkBody('category','必须指定文章类型').notEmpty();  
 req.checkBody('content','文章内容至少写几句').notEmpty();  
 var errors = req.validationErrors();  
 if (errors){  
 return res.render('admin/post/add',{  
 errors: errors,  
 title:　req.body.title,  
 content: req.body.content,  
 })  
 }  
 var title = req.body.title.trim();  
 var category = req.body.category.trim();  
 var content = req.body.content;  
 User.findOne({},function(err, author){  
 if (err){  
 return next(err);  
 }  
 //处理汉字  
 var py = pinyin(title,{  
 style: pinyin.STYLE\_NORMAL,  
 heteronym: false  
 }).map(function(item){  
 return item[0];  
 }).join(' ');  
 console.log(py);  
 var post = new Post({  
 title: title,  
 slug: slug(py),  
 category: category,  
 content: content,  
 author: author,  
 published: true,  
 meta: {favorite: 0},  
 comments: [],  
 created: new Date(),  
 });  
 post.save(function(err,post){  
 if (err){  
 req.flash('error','文章保存失败');  
 res.redirect('/admin/posts/add');  
 }else{  
 req.flash('info','文章保存成功');  
 res.redirect('/admin/posts');  
 }  
 });  
 });  
});

4.分类列表



**图4-10 分类列表页**/\*………………………………………………………………………………………

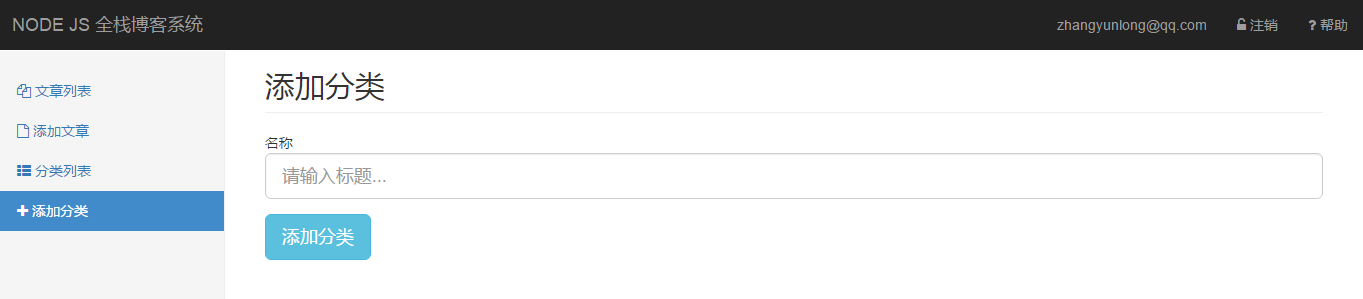
管理员对文章分类进行编辑,查看,删除其中修改文章分类功能的代码如下：

………………………………………………………………………………………\*/

//修改文章分类

router.post('/edit/:id',auth.requireLogin,getCategoryById,function (req, res, next) {  
 var category = req.category;  
 var name = req.body.name.trim();  
 var py = pinyin(name,{  
 style: pinyin.STYLE\_NORMAL,  
 heteronym: false  
 }).map(function(item){  
 return item[0];  
 }).join(' ');  
 category.name = name;  
 category.slug = slug(py);  
 category.save(function(err,category){  
 if (err){  
 console.log('category/edit error:', err);  
 req.flash('error','分类编辑失败');  
 res.redirect('/admin/categories/edit/'+ post.\_id);  
 }else{  
 req.flash('info','分类编辑成功');  
 res.redirect('/admin/categories');  
 }  
 });  
});

5.添加分类



**图4-11 添加分类页**

/\*………………………………………………………………………………………

管理员为博客系统添加文章分类,添加分类功能的代码如下：

………………………………………………………………………………………\*/

router.post('/add', auth.requireLogin,function (req, res, next) {  
 req.checkBody('name','分类标题不能为空').notEmpty();  
 var errors = req.validationErrors();  
 if (errors){  
 return res.render('admin/category/add',{  
 errors: errors,  
 name:　req.body.name,  
 });  
 }  
 var name = req.body.name.trim();  
 //处理汉字  
 var py = pinyin(name,{  
 style: pinyin.STYLE\_NORMAL,  
 heteronym: false  
 }).map(function(item){  
 return item[0];  
 }).join(' ');  
 console.log(py);  
 var category = new Category({  
 name: name,  
 slug: slug(py),  
 created: new Date(),});  
 category.save(function(err,category){  
 if (err){  
 console.log('category/add error:', err);  
 req.flash('error','分类保存失败');  
 res.redirect('/admin/categories/add');  
 }else{  
 req.flash('info','分类保存成功'); res.redirect('/admin/categories');} });  
})

第5章 系统测试和调试

在规定的条件下对程序进行操测试，找出程序中的错误，并及时对错误进行纠正以确保系统质量，并对其是否能满足设计要求和测试的出错情况对系统进行评估。目前软件测试仍然是保证软件系统可靠性的主要手段。系统测试的根本任务是发现并改正系统中的错误。测试用例的设计是测试阶段的关键技术，测试用例的好坏直接影响测试的效率，设计测试用例的基本目标是选用最少量的高效测试数据，做到尽可能完善的测试，从而尽可能多的发现软件中的问题。测试的对象是：程序，数据，文档。一般的测试过程可分为以下几步：

第一步：对要测试的系统进行分析，确定测试计划。

第二步：设计测试用例。

第三步：找出系统缺陷。

第四步：消除缺陷。

第五步：撰写测试报告。

白盒测试，黑盒测试是软件测试的两类基本方法。一般来说，白盒测试检查程序的逻辑正确性，测试的方法有代码检查法，逻辑覆盖法，基本路径测试法，路径覆盖等；黑盒测试检查程序是否符合设计要求，是否存在错误，注重测试系统的功能需求，一般有界面错误，输入输出错误，数据库访问错误等。

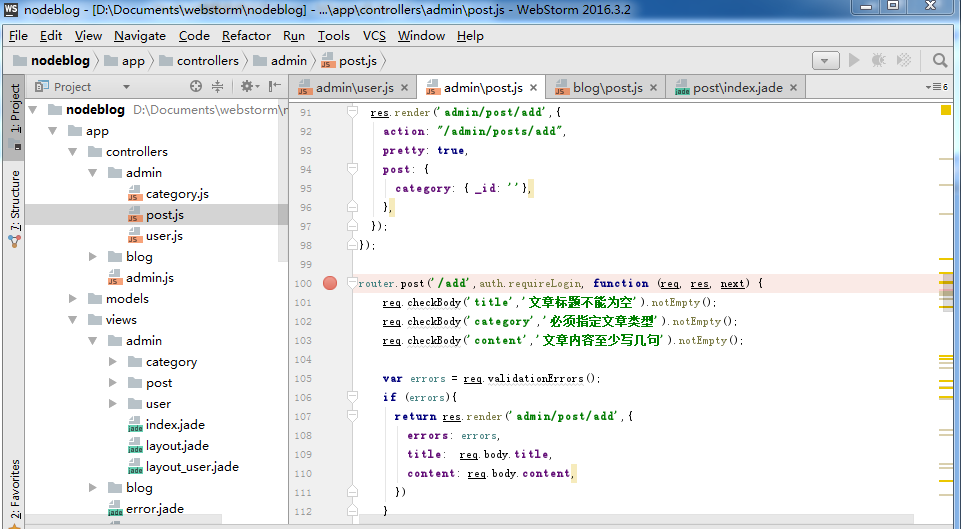
该系统运用白盒测试与黑盒测试相结合的方式以黑盒测试为主。下面将给出详细的测试过程：

5.1白盒测试

白盒测试是指按照程序的执行逻辑进行路径检测，即在程序实际运行过程中所经历的每一路径都必须测试到，检查是否存在引起程序出现错误的语句。通过检查软件内部的逻辑结构以确定实际运行状态与预期状态是否一致。在进行白盒测试时，使用调试工具进行流程跟踪，确保操作经过每个流程，揭示隐含在代码中的错误。

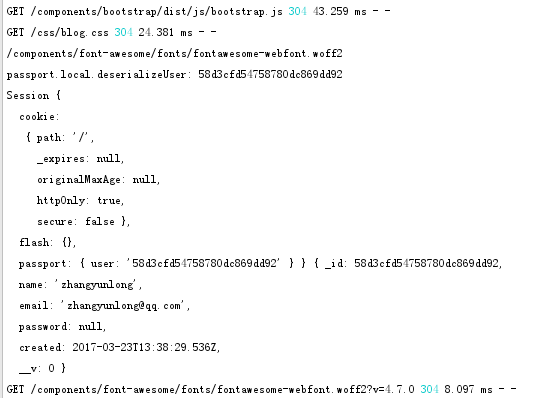
调试是指通过测试来检测系统中的错误并做出纠正。为了改正错误，首先必须准确判定错误的位置，同时尽量避免在调试过程中引进新的错误，保证一个模块中的独立路径至少被使用一次。

1.在程序中设置短点；如题5-1所示



**图5-1设置断点**

2.在跟踪模式下运行程序；如图5-2所示



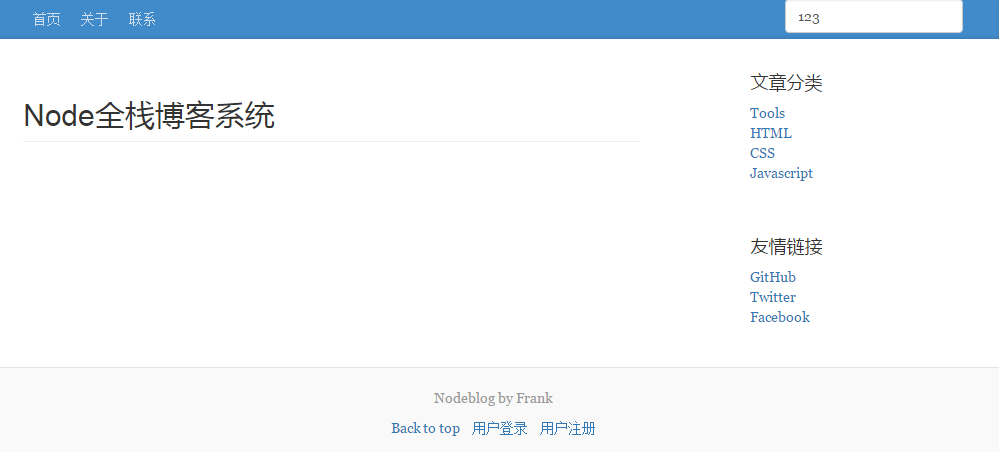
**图5-2调试应用程序**

5.2黑盒测试

黑盒测试是指考虑程序内部的逻辑结构及代码的实现，只检查其输入和输出的内容。对某段程序输入各类参数值，观察其结果。输入的数据可以根据设计内容而确定。理论上讲，黑盒测试只采用穷举输入测试，把所有可能的输入都作为测试情况考虑，实际上测试情况是又穷多个，既要测试合法输入，又要测试非法输入，从而保证系统的质量与健壮性。

设计黑盒测试方案时应考虑的问题如下：

1.测试数据的有效性；如系统中没有文章标题为‘123’的文章，如图5-3



**图5-3测试数据有效**

2.测试数据的合法性；如登录时密码错误如图5-4，注册时信息不合法如图5-5



**图5-4 登录密码错误**

****

**图5-5注册信息不合法**

3.测试功能的实用性；如对文章的列表按章分类和作者进行筛选，来方便用户查看文章。如图5-6



**图5-6 按条件筛选文章**

第6章 工作总结和展望

6.1 工作总结

该博客系统以前端文章为例，把文章分为 JavaScript，HTML，CSS，tools，四类，最基本的功能是向用户提供一个信息交流的平台，例如用户可以根据自己的需求查找或是搜索自己想要看到的文章，点击查看全文就可以对文章进行查看同时可以为其点个赞，也可以与发布者进行留言，交流，对于一般的浏览者可以通过分类查询，关键字搜索的方法快速找到想要看到的文章，关键的技术就是数据库表结构的设计，与各个模块之间的调用与配合，以及对Express中间件机制的掌握。对于后台管理页面。

本系统在研究的过程中，遇到了一些问题，例如函数模块的使用问题，由于本系统使用的模块较多，其功能有些没有熟练地掌握，消耗了一些时间，但我通过上网查阅资料，询问老师，自己理解与应用，使问题得到了解决；对一些算法的理解与掌握，使我对编程产生了更大的兴趣，扩宽了自己的知识面，收益颇深，提高了自己动手能力和解决问题的能力，增加了自信。

该系统满足了网络用户和博主的的需求。系统以其简洁，操作简单，的界面给用户带来方面，系统不仅具有添加、删除、查看，对文章进行分类等功能，还使用密码登录确保了管理员的隐私性，有效地解决了传统的学习方式所不具有的远程学习的问题，实现了资源共享，丰富了网络用户的业余生活。

6.2 工作展望

本次任务只是基于网络用户发表日志,但其适用性还是广泛的。在教育领域，教学部门把博客作为教学的新领域。博客在教育界带来的不仅有新的学习方式和教学方式，在教学改革方面注入了新的力量，适应了时代的需要。开放的课外教学课堂，让老师和学生发表自己独特性的见解，把课内外知识带到“博客课堂”，讨论和研究；把博客作为一个档案管理工具，学生对教师的教学活动做评价，教师之间的教学评价同样可以促进教师教学质量的提高，因此其应用前景还是广泛的。

尽管系统展现的功能和界面，验证了想法的合理性和有效性，但由于时间有限，关于课题还有一些功能需要进一步研究：

1.实现一个检测文章质量的功能，博客使用者具有一定的信息素养，对文章进行严格的筛选，以免形成网络垃圾。

2.代码的实现还不够完善，相信随着自我技术的提高与磨练，系统会出现质的飞跃。

3.实现一个能根据用户浏览最多文章的类型来为用户推送一些他么可能感兴趣的文章。

4.用户评论是可以在评论内容中输入表情。

5.实现用户的个人信息功能，用户可以添加昵称，修改密码，添加头像等。

# 参考文献

[1] Ethan Brow.Node与Express开发[M] .北京: 人民邮电出版社，2015:20-45.

[2] Cbodorow.MongoDB权威指南[M].北京: 人民邮电出版社 ,2014:21-56

[3] 赵坤.Node.js实战[M].北京: 电子工业出版社，2014:10-34

[4] Guillermo Rauch.了比起的Node.js[M].北京: 电子工业出版社 ,2014:44-61

[5] 朴灵.深入浅出的Node.js[M].北京: 人民邮电出版社, 2013:177-231

[6] BYVoid.Node.js开发指南[M].北京: 人民邮电出版社, 2012:79-129

[7] Cantelon.Node.js实战[M].北京: 人民邮电出版社, 2014:64-86

[8] Hughes-Croucher.Node即学即用[M].北京: 人民邮电出版社, 2013:109-125

[9] Benedetti.Head First jquery[M].北京: 中国电力出版社, 2012:76-122

[10] Flanagan.JavaScript权威指南[M].北京: 机械工业出版社, 2012:251-315

致 谢

在此毕业设计完成之日,谨向导师蔡晓龙老师致以诚挚的感谢!本论文在指导老师的指导下,从论文的选题,细节和成稿过程都得到了导师耐心的指导,使我的论文按时完成.蔡老师的严于律己，一丝不苟的工作作风，使我受益终生。

感谢洛阳师范学院的老师们的默默无闻的奉献和淳淳教诲！在传授给我们知识的同时，教会我们如何做一个对社会有用的人，赐予了我宝贵的财富！

感谢我的同学,是他们的支持和帮助，让我解决了一些问题!