新闻管理系统的设计与实现

袁一峰

摘要：本系统通过服务器端 SpringMVC、Spring、Mybatis 框架与前端 Bootstrap、angularjs 框架以及相关插件的结合，作为单页应用实现了对新闻发布、管理、查看等基本功能实现。在这些功能基础上，还增加了实时新闻推荐等功能。本论文主要阐述了系统结构设计以及相关功能的设计、开发工具环境相关信息、系统主要功能流程等。

1 产品概述

2 系统需求分析

2.1 系统功能需求

系统分三个模块实现功能，用户模块用于向注册及非注册用户提供相关功能，非注册用户提供查看新闻、查看评论的功能，注册用户除了拥有非注册用户的功能以外还拥有对文章的评论操作、收藏文章操作、实时内容推荐和基本信息维护。作者模块包括对文章的发布等维护，查看文章的评论，查看文章以及作者的相关数据统计。管理员模块可以审核作者发布的文章、管理员基本信息维护等相关功能。

2.2 系统运行环境要求

服务器端：

操作系统：Windows 2000/XP/Vista/Win7/Windows 2003/Windows 2008

安装软件：MySQL 数据库、tomcat7.0

客户端：

IE6.0 及以上版本，IE 内核浏览器以及火狐浏览器、谷歌浏览器等。

2.3 系统模块功能

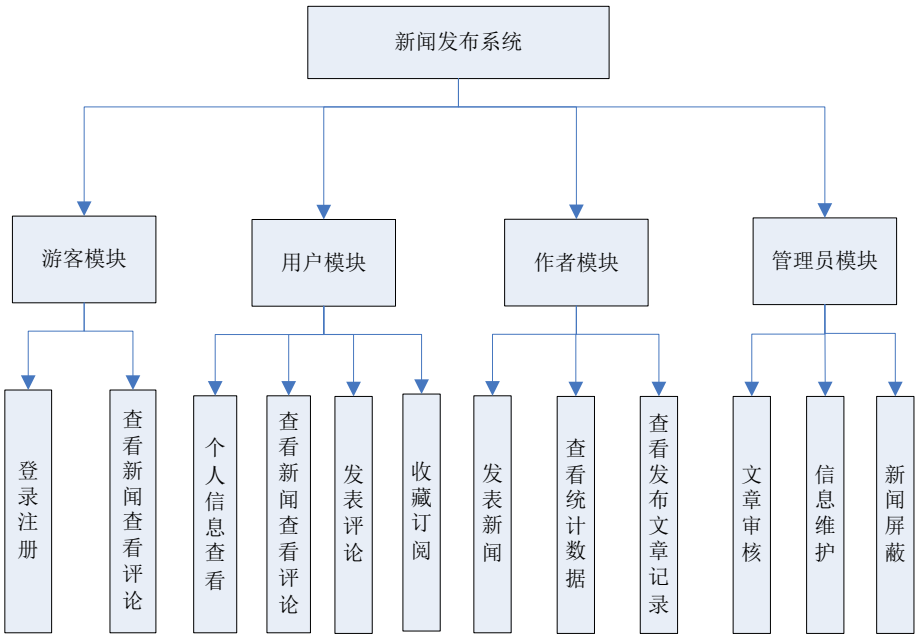

图 2-1 系统模块图

2.4 核心业务流程

2.4.1 用户评论功能

注册用户登录后可以对看过的文章进行评论。

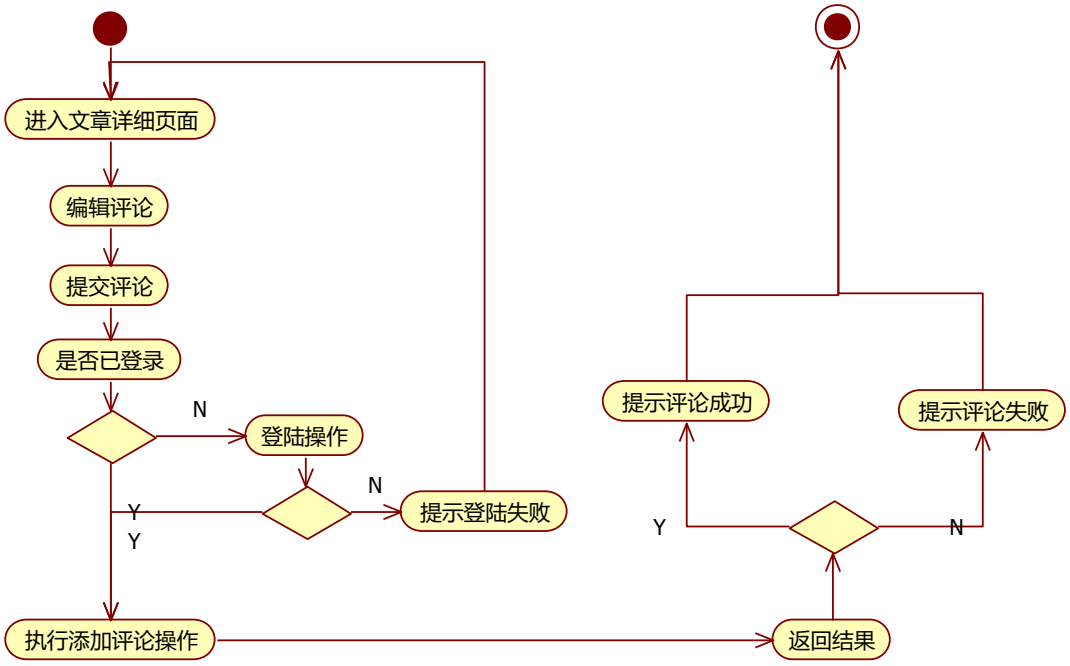

图 2-2 用户评论流程图

2.4.2 作者文章发布

注册为作者并登录后可以进行文章发布的操作。

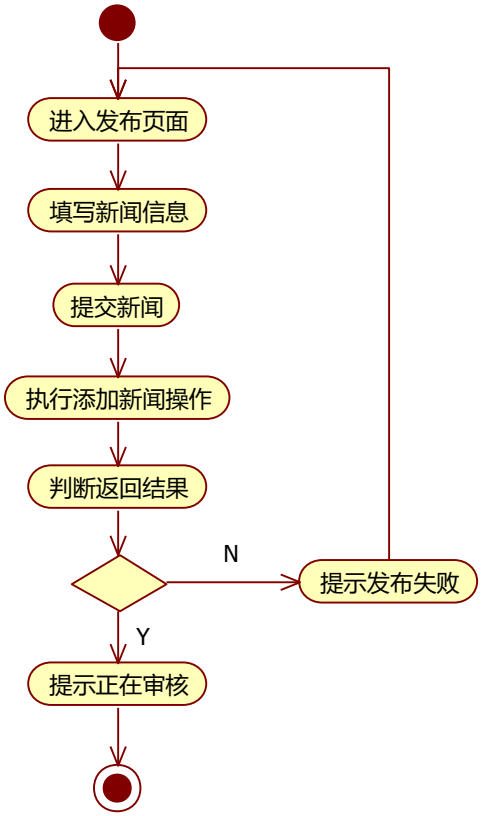

图 2-3 信息发布流程图

2.4.3 用户收藏订阅功能

注册用户登录后可以对文章进行收藏、对作者进行订阅。

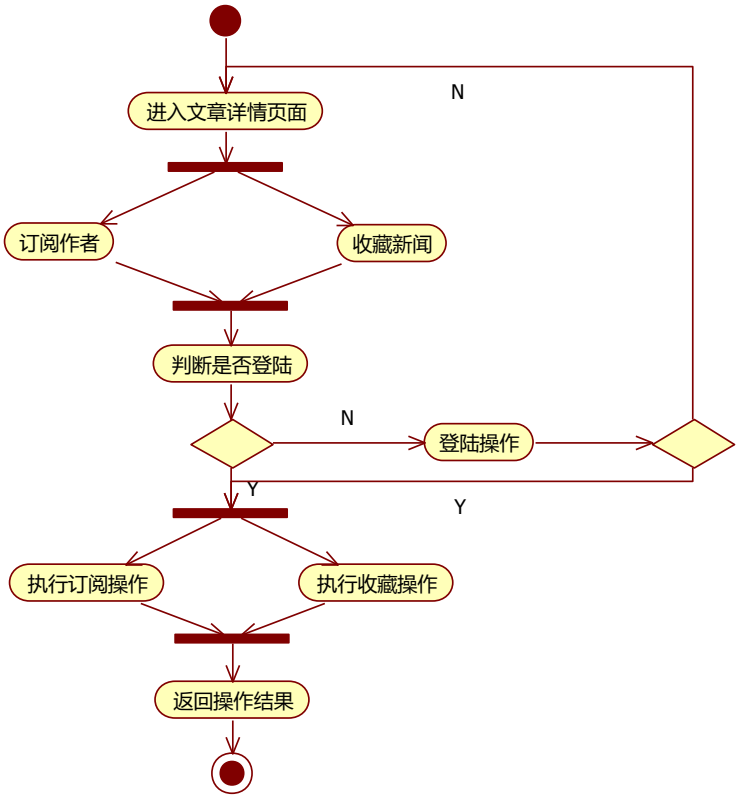

图 2-4 收藏订阅流程图

3 概要设计

3.1 系统功能模块划分

系统共分成三个模块：用户模块；作者模块；管理员模块。

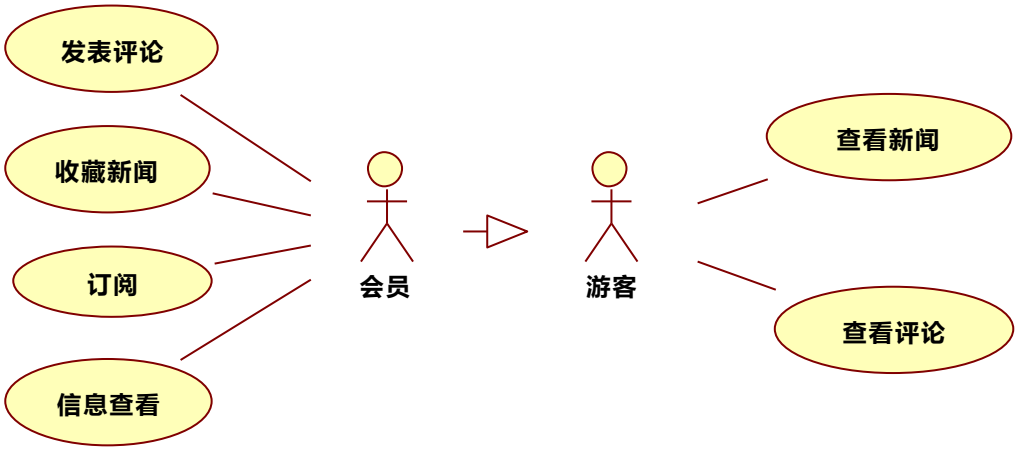

图 3-1 用户模块用例图

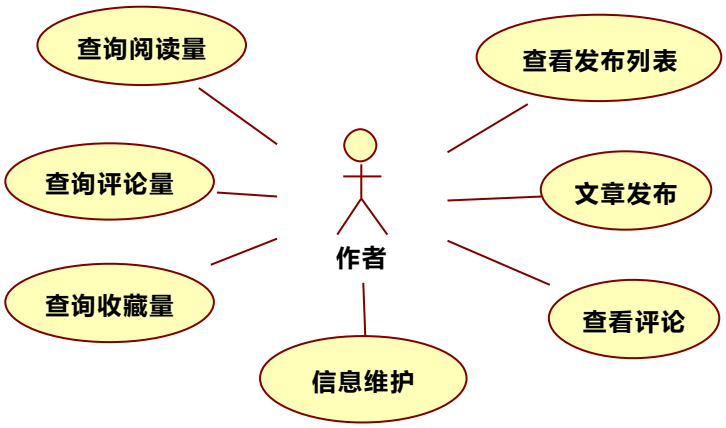

图 3-2 作者模块用例图

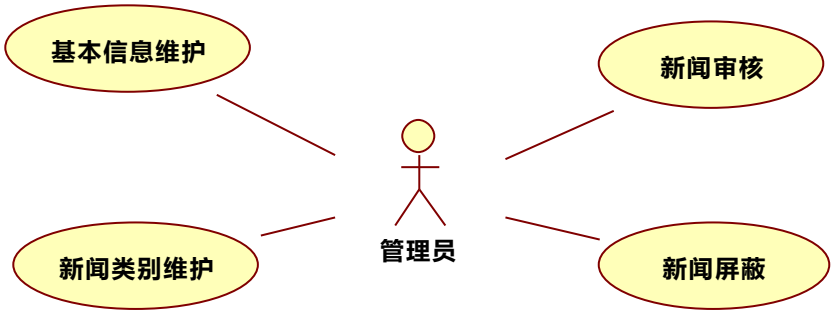

图 3-3 管理员模块用例图

3.2 数据库逻辑设计

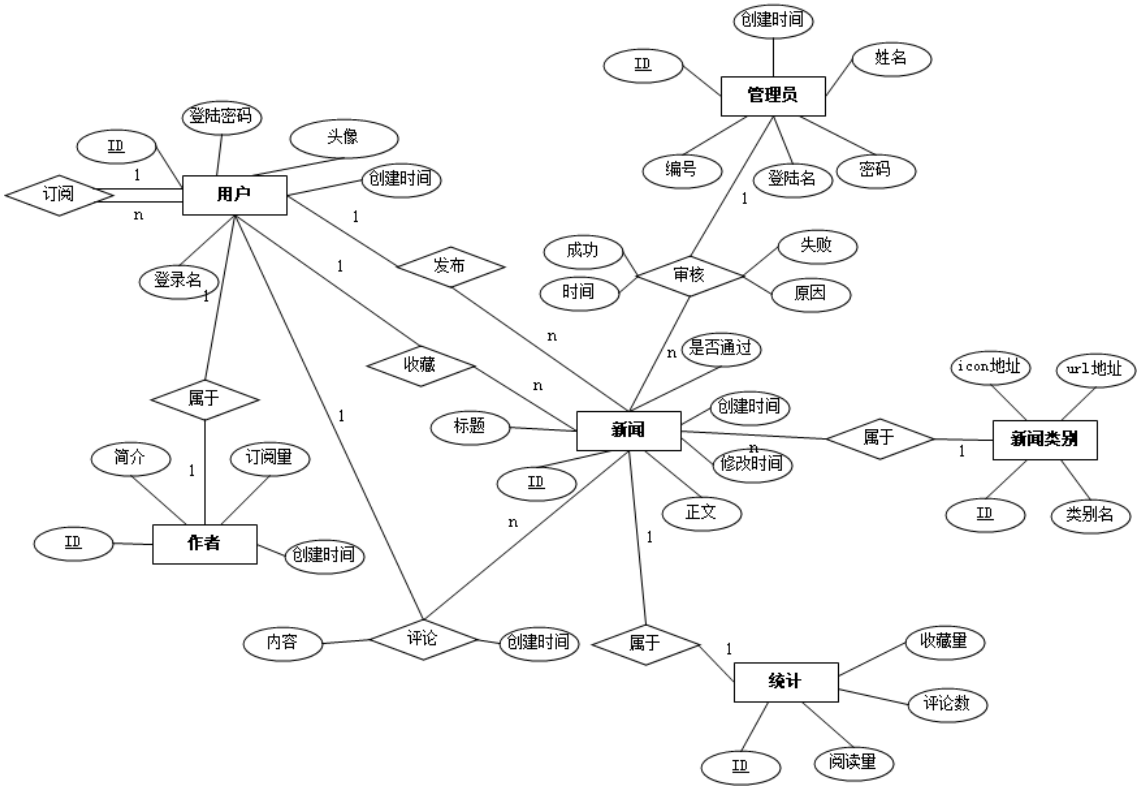

图 3-4 系统 er 图

3.3 系统开发环境

本系统的开发基于 win7 操作系统，后台数据库为 MySQL 数据库，数据库可视化工具为 SQLyog，后台代码编写软件为 Eclipse Luna，前台代码编写软件为 Sublime Text,服务器为 tomcat7.0。

4 详细设计

4.1 数据库物理设计

数据库表共 11 张，具体数据库表如下：

表 4-1：用户表,用于存储用户基本信息数据，包括主键 id、用户设置的昵称、登录名密码、创建时间以及用户上传的头像。登录名密码用于登录，当用户登录后，则显示用户的昵称，用户评论后，在评论列表显示用户昵称和设置的头像。

表 4-1 用户表(user)

序号	字段名	属性名称	属性类型	是否为空	约束	备注
A0001	user_id	ID	varchar(255)	不为空	主键	
A0002	user_name	用户昵称	varchar(255)	不为空		
A0003	user_loginname	登录名	varchar(10)	不为空		
A0004	user_password	登录密码	varchar(255)	不为空		
A0005	user_createtime	创建时间	datetime	不为空		
A0006	user_img_url	用户头像	varchar(255)			

表 4-2：作者表，当注册为用户时默认注册作者表，里面包含作者的相关信息如作者的简介以及该作者的订阅量，将订阅量写进此表是为了读取数据方便，节省查询数据时间。表中外键连接用户表，指向用户 id。只有创建用户时才会操作此表，以此保证用户表与作者表连接的唯一性。该表用于在作者主页中联合用户表显示作者相关信息。

表 4-2 作者表(author)

序号	字段名	属性名称	属性类型	是否为空	约束	备注
B0001	author_id	ID	varchar(255)	不为空	主键	
B0002	author_abstract	简介	varchar(255)			
B0003	author_createtime	创建时间	datetime	不为空		
B0004	author_subscribe	订阅量	int	不为空		
B0005	user_id	用户 ID	varchar(255)	不为空	fk(user_id)	

表 4-3：管理员表，管理员用于对作者发表的文章进行审核，审核通过才会显示给其他用户和游客，此表存储管理员的登录基本信息。

表 4-3 管理员表(manager)

序号	字段名	属性名称	属性类型	是否为空	约束	备注
C0001	manager_id	ID	int	不为空	主键	自增
C0002	manager_loginname	登录名	varchar(255)	不为空		
C0003	manager_password	密码	varchar(255)	不为空		
C0004	manager_name	姓名	varchar(255)			
C0005	manager_createtime	创建时间	datetime	不为空		

表 4-4：新闻类别表，此表既作为类别区分文章，也是主页中的导航选项，用户在主页点击不同类别可预览不同类别的文章。表中存入类别名称、主页中显示类别的图标地址、以及点击这些类别的 url 地址。

表 4-4 新闻类别表(category)

序号	字段名	属性名称	属性类型	是否为空	约束	备注
D0001	category_id	ID	int	不为空	主键	自增
D0002	category_name	类别名	varchar(255)	不为空		
D0003	category_imgurl	icon 地址	varchar(500)	不为空		
D0004	category_url	url 地址	varchar(500)	不为空		

表 4-5：新闻表，此表为该系统中主要的表，作者发布新闻时将记录存入此表，其中作者需填写的为标题、正文、新闻类型。传入服务器后处理数据，过滤出新闻的简介、预览图等信息，最后在此表创建新纪录。该表也用于提供文章列表、文章详细信息、管理员审核等功能（管理员审核文章后修改该表的 new_pass 字段标识该新闻的状态，0 为未审核，1 为通过，-1 为未通过），外键作者 id 关联用户表中的用户 id。

表 4-5 新闻表(new)

序号	字段名	属性名称	属性类型	是否为空	约束	备注
E0001	new_id	ID	varchar(255)	不为空	主键	
E0002	new_title	标题	varchar(500)	不为空		
E0003	new_content	正文	text	不为空		
E0004	new_text	简介	varchar(255)	不为空		
E0005	new_img	新闻预览图	varchar(255)	不为空		
E0006	new_createtime	创建时间	datetime	不为空		
E0007	new_updatetime	修改时间	datetime	不为空		
E0008	new_pass	是否通过	int	不为空		0:待 1:Y-1:N
E0009	category_id	新闻类型	int	不为空	fk(category_id)	
E0010	user_id	作者	varchar(255)	不为空	fk(user_id)	

表 4-6：审核未通过表，当管理员审核文章未通过时，将新闻 id，不通过原因等信息存入此表，显示给作者看。

表 4-6 审核未通过表(checkfailed)

序号	字段名	属性名称	属性类型	是否为空	约束	备注
F0001	chf_id	ID	int	不为空	主键	自增
F0002	chf_reason	不通过原因	varchar(500)			
F0003	chf_createtime	审核时间	datetime	不为空		
F0004	new_id	新闻	varchar(255)	不为空	fk(new_id)	
F0005	manager	审核员	int	不为空	fk(manager_id)	

表 4-7：审核通过表，当管理员对文章的审核通过时，将审核的相关信息存入此表，此表内字段信息不显示在前台，只作为管理员操作的记录，并方便后期功能扩展和信息维护。

表 4-7 审核通过表(checkpass)

序号	字段名	属性名称	属性类型	是否为空	约束	备注
G0001	chp_id	ID	int	不为空	主键	自增
G0002	chp_createtime	创建时间	datetime	不为空		
G0003	new_id	新闻	varchar(255)	不为空	fk(new_id)	
G0004	manager_id	审核员	int	不为空	fk(manager_id)	

表 4-8：评论表，系统内所有文章的评论都存于此表，其中外键新闻 id 关联新闻表中的新闻主键，评论用户关联用户表中的用户主键。此表与用户表结合，用于在前台文章详情页中显示该文章的评论信息。

表 4-8 评论表(comment)

序号	字段名	属性名称	属性类型	是否为空	约束	备注
H0001	comment_id	ID	int	不为空	主键	
H0002	comment_content	内容	varchar(255)	不为空		
H0003	comment_createtime	创建时间	datetime	不为空		
H0004	new_id	新闻 id	varchar(255)	不为空	fk(new_id)	
H0006	user_id	评论用户	varchar(255)	不为空	fk(user_id)	

表 4-9：收藏表，用于注册用户对浏览过的文章进行收藏，为防止文章在其他用户收藏后删除出错，所以创建此记录时添加收藏文章的文章名作为用户对收藏新闻列表的预览，当进入某个收藏文章详细信息时在新闻表中读取该文章的实际信息。此表文章外键关联新闻表中的新闻主键，用户外键为点击收藏的用户，关联用户表主键。

表 4-9 收藏表(collection)

序号	字段名	属性名称	属性类型	是否为空	约束	备注
I0001	collection_id	ID	int	不为空	主键	自增
I0002	collection_title	收藏文章名	varchar(255)	不为空		
I0003	new	文章	varchar(255)	不为空	fk(new_id)	
I0004	user	用户	varchar(255)	不为空	fk(user_id)	

表 4-10：订阅表，用户查询作者信息时，可订阅该作者，该表关联用户表、作者表在用户管理中的订阅列表显示相关字段。外键用户对应用户表的主键，作者对应的也是该作者对应用户表的主键。

表 4-10 订阅表(subscribe)

序号	字段名	属性名称	属性类型	是否为空	约束	备注
J0001	subscribe_id	ID	int	不为空	主键	自增
J0002	user_id	用户	varchar(255)	不为空	fk(user_id)	
J0003	author_id	作者	varchar(255)	不为空	fk(user_id)	

表 4-11：统计表，存有每一篇文章的访问量、收藏量、等统计数据，用于在作者管理中展示不同的数据，每一篇发表的文章都会对应统计表的一条记录，当文章被访问、被收藏、被评论时都会修改该表的相关字段。

表 4-11：统计表(statistic)

序号	字段名	属性名称	属性类型	是否为空	约束	备注
M0001	statistic_id	ID	int	不为空	主键	自增
M0002	statistic_read	阅读量	int	不为空		
M0003	statistic_collection	收藏量	int	不为空		
M0004	statistic_comment	评论数	int	不为空		
M0005	new	文章	varchar(255)	不为空	fk(new_id)	

4.2 系统类设计

用户模块类图，实现用户信息维护等相关操作：

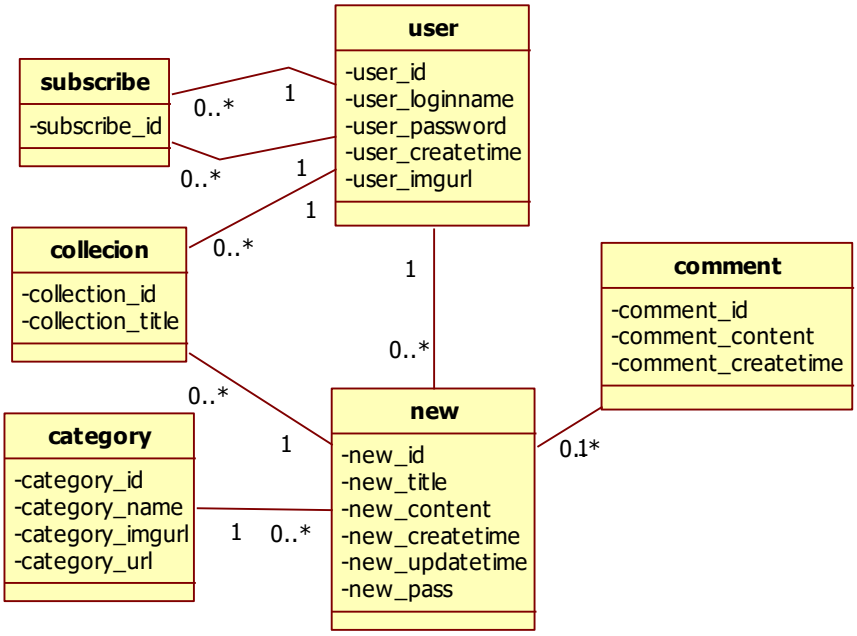

图 4-1 用户模块类图

作者模块类图，实现作者信息维护、文章信息维护、数据统计等操作。

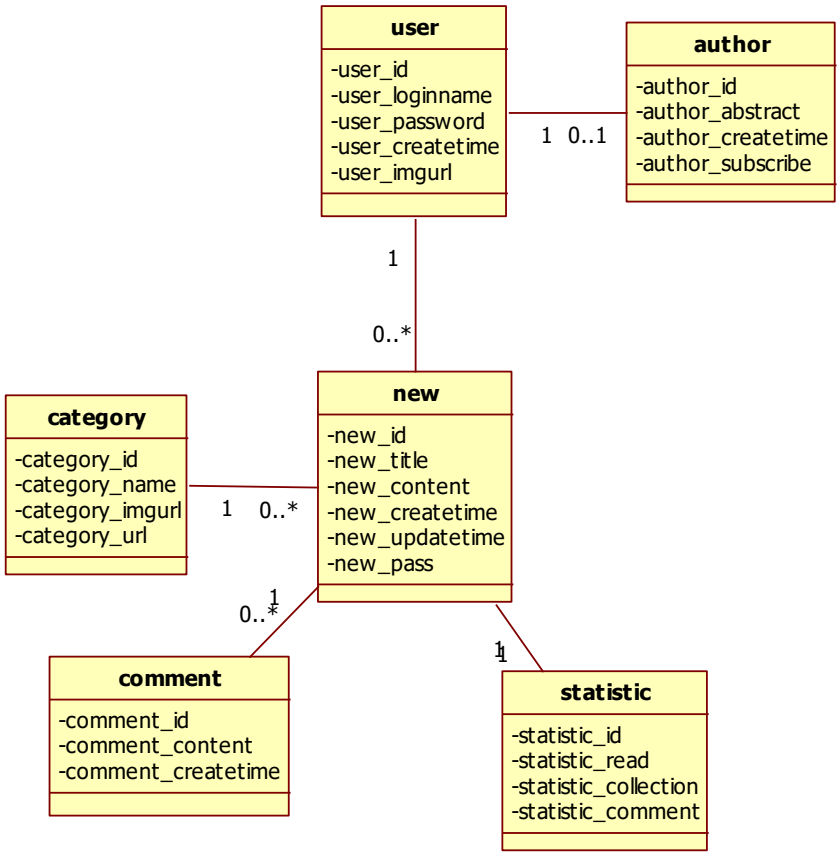

图 4-2 作者模块类图

管理员模块类图，实现维护管理员信息、审核文章、屏蔽文章等功能。

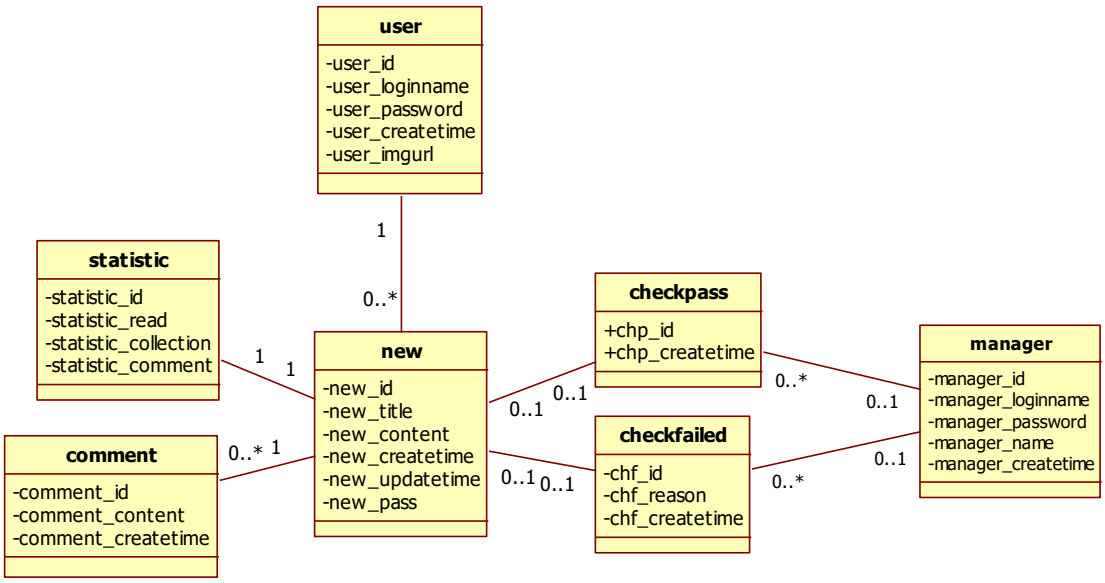

图 4-3 管理员模块类图

5 系统实现

5.1 系统主要功能说明

5.1.1 游客模块

游客登录功能描述：

ui 层：提交登录相关信息后，将信息封装传给 action 层。如果返回 sessionid，则将 sessionid、用户名、用户 id 等信息封装并存入前端的 session；如果获取错误信息，则提示用户名密码错误。

Action 层:获取登陆对象，将对象传给 service 层。获取返回值判断如果为空则用户名密码错误，如果对象不为空，则将对象传入 session，返回 sessionid；

Service 层:将传入的对象传给 Mapper 层查询，如果存在该对象则返回对象，不存在返回空。

图 5-1 游客登录界面

游客登录代码：

用户首次登录时，如果登录成功，则返回 sessionid 和登录用户信息，前端收到登录信息以后储存在本地 session(HTML5)中，每次刷新界面前端都会用 sessionid 和用户信息请求后端进行匹配，如果匹配成功则继续其他操作，如果不成功则前端删除本地的 session。

前端部分代码：

```
app.controller('login',function($scope,$http,loginService,$window,current
User){
    $scope.currentuser=[];
    $scope.logined=false;
    $scope.loginresule = true;
    var userString = $window.sessionStorage.getItem("user");
    if(userString == null || userString == ""){
        $scope.logined = false;
    }else{
        $scope.currentuser = JSON.parse(userString);
        var token =$scope.currentuser.token;
        currentUser.get(token).then(function(m){
            if(m.data.message == "1"){
                console.log("success");
                $scope.logined=true;
            }else{
                $scope.logined=false;
                $window.sessionStorage.setItem("user","");
            }

        },function(m){console.log("error");});
    }

    $scope.userlogin=function(user){
        var userLoginName = user.userLoginName;
        var userPassword = user.userPassword;

    loginService.userLogin(userLoginName,userPassword).then(function(messa
ge){
            var login = message.data;
            if(login.status == "1001"){
                var user =
{"token":login.token,"userId":login.userId,"userName":login.userName};

                $scope.currentuser=user;
                var userString  = JSON.stringify(user);
                $window.sessionStorage.setItem("user",userString );
                $scope.loginresule = true;
                $('#modal-login').modal('hide');
```

```
            $scope.logined = true;
        }else{
            //登录失败
            console.log("error");
            $scope.loginresule = false;
        }
    });
    };
```

前端 HTML：

```
<ul class="list-inline right">

    <li ng-show="logined"><a href="#/user/newsshow">发布</a></li>

    <li ng-hide="logined"><a href="#" data-toggle="modal"
            data-target="#modal-login">登录</a></li>

    <li ng-hide="logined"><a href="#"data-toggle="modal"
    data-target="#modal-register">注册</a></li>

    <li ng-show="logined"><a href="#/user/shoucang"
    ng-bind="currentuser.userName"></a></li>

    <li ng-show="logined"><a href="javascript:;"
    ng-click="logout()">注销</a></li>

</ul>
```

5.1.1 用户模块

用户评论功能描述：

ui 层：用户填写评论信息并提交后，首先判断 session 中是否有用户登录，如果没有则提示登录，如果已经登录，则获取该篇文章的 id 以及评论的内容，封装传入 Action 层中。在 action 层中获取的返回值状态如果为成功，则提示发表评论成功，并刷新评论列表，如果失败则返回失败提示。

Action 层:获取的评论相关数据封装成评论对象传给 service 层，如果未捕获异常则将成功的状态码传给 ui 层。

Service 层：将获取的对象设置未定义字段，如创建时间。赋值后将评论对象传给 Mapper 层进行持久化。

图 5-2 用户评论功能页面

5.1.2 作者模块

作者发布文章描述：

ui 层:用户输入文章标题、内容、类别等相关信息后，将文章信息传入 action 层。获取 action 层的返回值，如果为成功，则提示发表成功正在审核，并跳到作者管理主页中。

Action 层:通过 session 获取用户 id，并将 ui 层传入的文章等相关信息封装成对象传给 service 层。如没捕获异常则返回成功状态码，并进行相关操作。如果捕获异常则提示添加失败，返回失败状态码。

Service 层：获取文章后处理文章相关字段，然后添加默认字段以及注册时间等信息，然后将文章对象传给 Mapper 层执行添加操作。

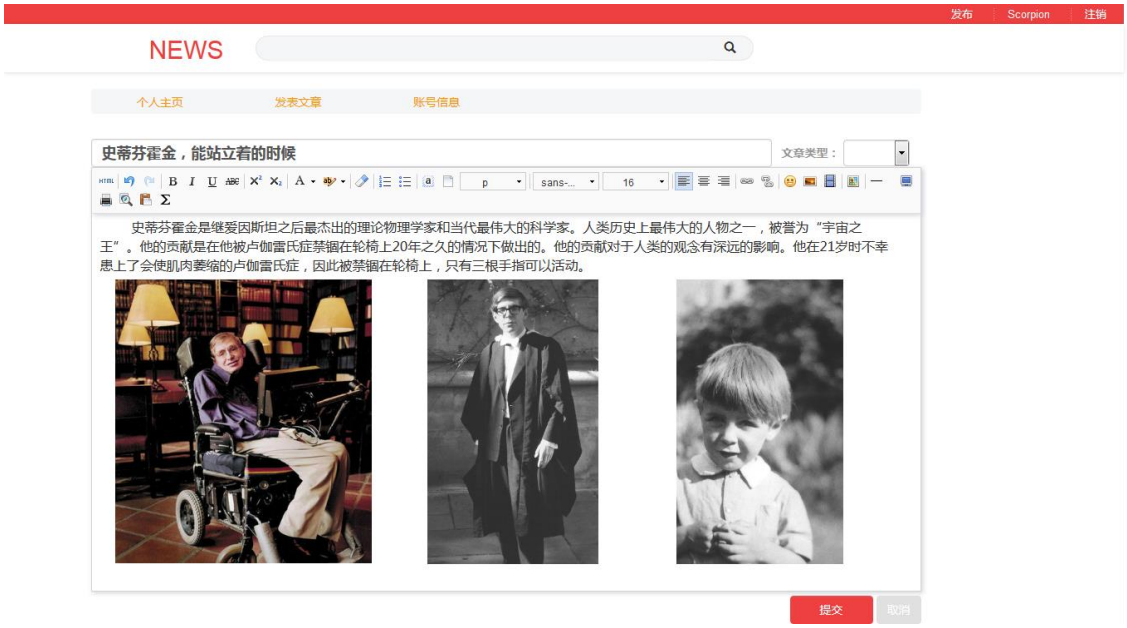

图 5-3 添加文章功能页面

作者发布文章代码：

进行添加新闻操作时，当用户在编辑器中插入图片时，将图片通过异步操作上传至服务器的临时文件夹中，上传成功后返回图片地址。用户点击上传后，获取内容并传至后端，在后端将内容中的图片复制到项目外，并更改图片的项目路径。其中将第一张图片设置成新闻的预览图（过滤 gif 表情动画），最后传至数据库中。将图片从数据库中分离出来，减缓数据库存储压力。

前端部分代码：

```
app.controller("addnewsCtrl",function($scope,$http){
    $scope.content = "";
    $scope.title = "";
    $scope.categroy = "";
    $scope.text = "";
    $scope.submit = function(){
        $scope.content = UM.getEditor('myEditor').getContent();
        text = UM.getEditor('myEditor').getContentTxt();
        $scope.text = text.replace(/\ +/g,"").substring(0,78);
        var news =
{"newTitle":$scope.title,"newContent":$scope.content,"categoryId":$scope.
categroy,"newText":$scope.text};
        $http({
            method:'post',
            url:'addNews',
            data:news
    }).success(function(m){
        console.log(m.status);
        window.location.href="#/user/newsshow"
    });
```

```
    }

});
```

后端图片处理代码：

```
public Map<String, String> changeFilePath(String source,String
              picPath,String contextPath) throws IOException{
    List<String> picPathList = new ArrayList<String>();
    Pattern p=Pattern.compile("(/\\w*){2}/[0-9]{1,}[.](jpg|png)");
    source = source.replaceAll("_src=\\s*['\"]([^'\"]+)['\"]", "");
    Matcher m=p.matcher(source);
    Map<String,String> map = new HashMap<String, String>();
    String newImg = "";
    if(m.find()){
        newImg = m.group().substring(1);
        do{
        picPathList.add(m.group().substring(1).replaceAll("/","\\\\"));
           }while(m.find());
     String tempPath = newImg.substring(0,newImg.lastIndexOf("/"));
     String newPath = ("G:\\newsfile\\newspic\\"+ tempPath +
        "\\").replaceAll("/","\\\\").replaceAll("\\\\","\\\\\\\\");
     File file = new File(newPath);
     if(!file.exists() || !file.isDirectory()){
           file.mkdirs();
        }
     FileInputStream[] fis = new FileInputStream[picPathList.size()];
     FileOutputStream[] fos = new FileOutputStream[picPathList.size()];
     for(int i = 0; i<picPathList.size(); i++){
        fis[i] = new FileInputStream((picPath+picPathList.get(i))
            .replaceAll("\\\\", "\\\\\\\\"));
        fos[i] = new FileOutputStream(("G:\\newsfile\\newspic\\"+
            picPathList.get(i)).replaceAll("\\\\", "\\\\\\\\"));
        FileChannel fci = fis[i].getChannel();
        FileChannel fco = fos[i].getChannel();
        ByteBuffer bb = ByteBuffer.allocate(1024);
        int n = fci.read(bb);
        while(n != -1){
            bb.flip();
            fco.write(bb);
            bb.clear();
            n = fci.read(bb);
         }
            fci.close();
            fco.close();
            fis[i].close();
```

```
            fos[i].close();
        }
        String content = source.replaceAll(contextPath+"/"+tempPath,
                        "/newspic/"+tempPath);
        map.put("content", content);
        map.put("newsImg", "/newspic/"+newImg);
        }else{
            map.put("content", source);
            map.put("newsImg", "");
         }
      return map;
    }
```

5.1.3 管理员模块

管理员文章审核：

ui 层:查看文章后，将审核结果和原因以及文章 id 传到 action 层中。

Action 层:将获取信息传给 service 层。

Service 层：如果审核结果通过，将文章 id、审核员等信息封装传入 Mapper 层添加到审核通过表，再通过 Mapper 层修改文章审核字段。如果审核未过则将文章 id、审核员等信息封装传入 Mapper 层添加到审核不通过表。

图 5-4 添加文章功能页面

管理员获取文章审核列表代码：

此段代码在获取新闻列表的基础上，增加了防止不同管理员审核同一篇文章的情况。实现方式为在数据库获取指定长度的新闻列表添加到请求新闻队列中，管理员每次请求新闻列表都会在此队列中获取，如果小于请求的新闻队列长度（系统中为 10 条），则会在数据库获取新闻列表补充至最大长度（系统中为 20 条）。管理员未完成的新闻列表将在 session 注销时返回到队列。从而减轻数据库压力。

获取新闻属性：

```
public class NewsQueue {
```

```
    private String newsId;//新闻 id
    private String newsTitle;//新闻标题
        private double newsTimeWeight;//新闻时间权重
    }
```

获取队列类：

```
public class ManageNewsQueue {
    private List<NewsQueue> queue;//新闻队列
    private double timeWeight = -1;//时间权重
    private boolean lock;//锁标记

    @Autowired
    private  NewsQueueMapper newsQueueMapper;

    public ManageNewsQueue() {
        if(getTimeWeight() == -1){
            queue = new ArrayList<NewsQueue>();
            timeWeight = 0;
            lock =false;
        }
    }

    public  List<NewsQueue> getNewsQueue() throws InterruptedException{
        while(true){
            if(!isLock()){
                setLock();//加锁
                break;
            }
            else Thread.sleep(500);
        }
        Integer length = getQueue().size();
        if(length<10){
            Map<String, Object> map = new HashMap<String, Object>();
            map.put("quantity", 20-length);
            map.put("weight", getTimeWeight());
            List<NewsQueue> list = newsQueueMapper.getNewstoQueue(map);
            if(list.size()>0){
                setTimeWeight(list.get(list.size()-1).getNewsTimeWeight());
                getQueue().addAll(list);
            }
        }
        List<NewsQueue> q = new ArrayList<NewsQueue>();
            int count = 0;
            while(count < getQueue().size()){
                q.add(getQueue().get(0));
```

```
                getQueue().remove(0);
                if(++count >= 9)break;
            }
        setLock();//解锁
        return q;
                    }
    public boolean isLock() {
        return lock;
    }

    public void setLock() {
        this.lock = !this.lock;
    }
}
```

获取新闻 sql：

```
<select id="getNewstoQueue"  parameterType="java.util.Map" resultType="com.news.entity.NewsQueue">
  SELECT new_id newsId,new_title newsTitle,new_createtime+new_updatetime newsTimeWeight
  FROM `new`
  WHERE  new_pass = 0 AND new_createtime+new_updatetime > #{weight}
  ORDER BY new_createtime+new_updatetime
  LIMIT 0,#{quantity}
</select>
```

图 5-5 获取新闻列表 sql

5.2 系统其它功能截图

主页：用户最先看到的页面，进行的操作有获取类别导航列表、获取文章列表、获取实时新闻列表、判断是否登陆等功能。

图 5-6 系统主页

用户注册：验证用户输入的数据，如果不符合则不能注册，相关验证有：手机格式、密码位数及不能有空格、密码与确认密码一致，如果通过则可以注册。

图 5-7 注册界面

添加新闻图片：通过前端插件可在文章中插入多张图片，也可复制图片粘贴进去。

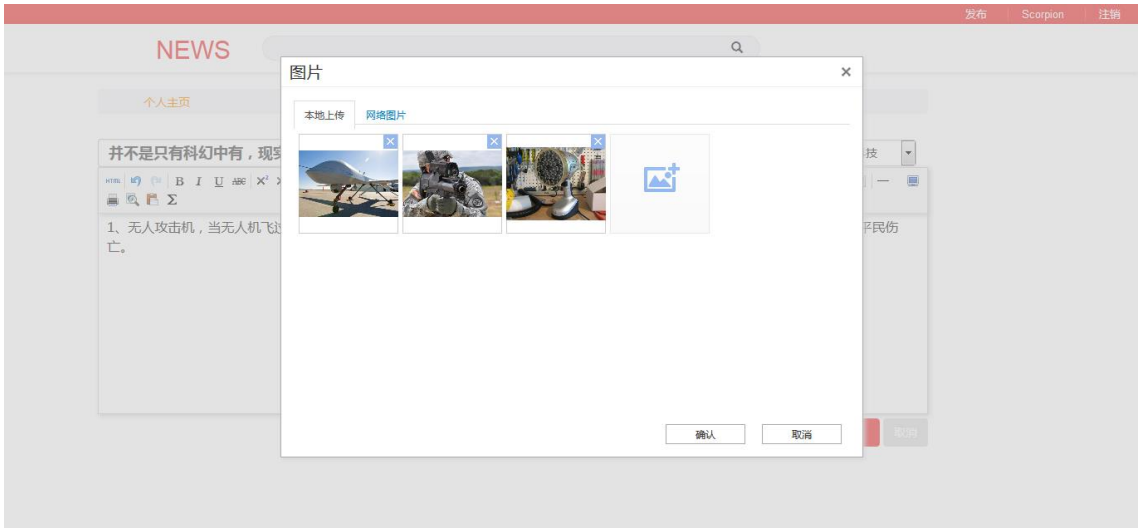

图 5-8 添加新闻图片界面

新闻详情页面：通过传入的新闻 id 获取该新闻信息以及该新闻的评论列表、作者等信息。点击作者名可以跳入作者信息页面。

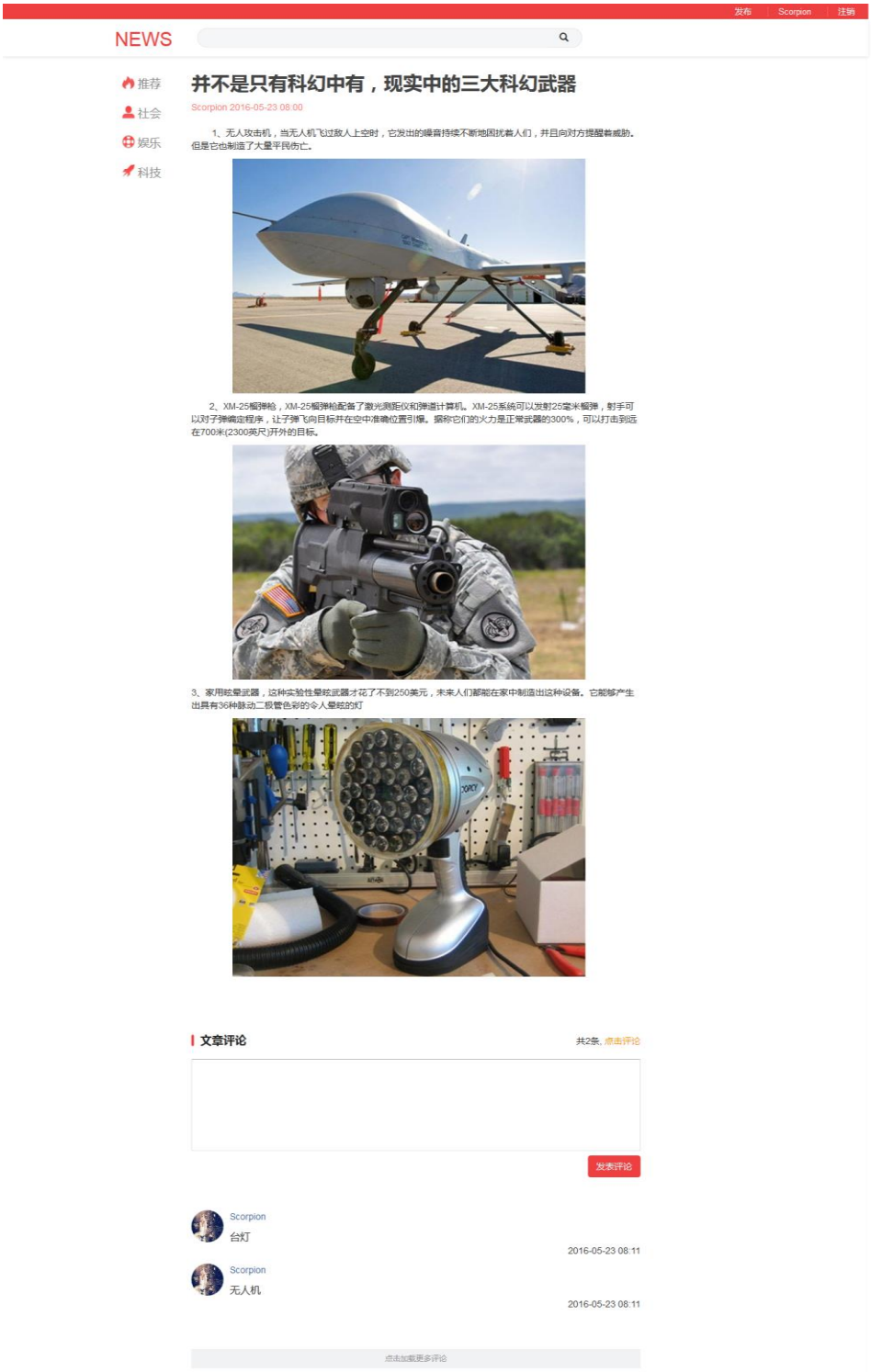

图 5-9 新闻详情界面

作者信息页面：通过作者 id 获取该作者的基本信息、发布过的文章列表以及发布量订阅量。

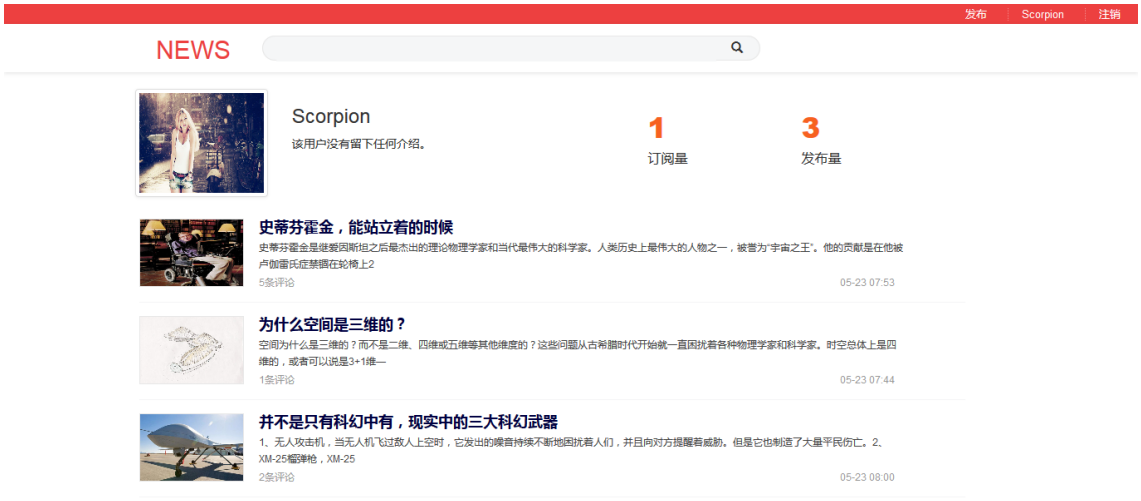

图 5-10 作者信息界面

用户收藏界面：显示登陆用户所有收藏过的文章，其中显示文章预览图、标题、简介、作者名称以及创建时间。

图 5-11 用户收藏界面

用户订阅界面：注册用户可浏览已经订阅过的作者，点击作者头像可进入该作者的详情页面。

图 5-12 用户订阅界面

用户信息页面：显示登录用户的基本信息及头像。

图 5-13 用户信息界面

作者管理主页：显示该作者曾经发布过的新闻列表，并显示该文章的评论数。

图 5-14 作者管理主页

6 设计总结

在本次项目中，使用的数据库为 MySQL，后台逻辑实现主要依照 MVC 思想，将模型、视图和控制层分离出来，用 springMVC、mybatis、spring 框架对代码进行具体实现。提高了代码的重用率和便于维护的能力。用这些框架主要是因为整合方便、更轻量级，易上手，并且可以缩短开发时间，而 springMVC 拥有更快的响应速度，可以获得更好的用户体验。通过 mybatis

对 sql 语法控制更加精准。

前台使用框架为 bootstrap 和 angularjs。通过模型的双向绑定等功能，大大减少了代码量，提高效率。使用了一些算法比如输出评论，防止不同管理员审核同一篇文章等。系统的不足为无法进行文章的查重，尽管写出了一些查重的方法但是效果不理想于是将此功能删除。这次设计的最大收获就是对自己自学的能力有了很大的提升，没有依靠老师教授来学会新的东西，找到了很多学习的方法并可以运用到以后得项目中去。