分类号：TP311.5UDC：D10621-408-(2019)2388-0

密　级：公开编号：**2015081131**

成都信息工程大学

学位论文

ACM公众号的设计与实现

|  |  |
| --- | --- |
| 论文作者姓名： | 官怡婷 |
| 申请学位专业： | 软件工程 |
| 申请学位类别： | 工学学士 |
| 指导教师姓名（职称）： | 杜晓宇（讲师） |
| 论文提交日期： | 2019年5月20日 |

ACM公众号的设计与实现

**摘　要**

随着ACM竞赛的不断发展，各大高校逐渐重视ACM竞赛，纷纷在各大社交平台宣传学校内部的ACM校队。在这个新媒体时代，微信公众号是一种有效的传播手段，大部分高校已经拥有了自己的ACM公众号，而我校未能在这一大影响力的平台有自己的一席之地，因此设计实现ACM公众号去宣传我校ACM信息。

本系统使用IntelliJ IDEA开发工具进行开发，使用Java语言和SSM框架，前端页面使用React框架，采用前后端分离的开发模式，利用HTTP协议和JSON格式数据进行信息交互，服务器使用Apache Tomcat，数据库使用MySQL和Redis。本系统分为未授权用户、授权用户和管理员，根据不同的权限具有不同的功能。用户主要实现登录注册、论坛管理、发表评论、上传图片、讲座报名、发布问题及题解，管理员主要实现用户管理、相册及图片管理、论坛管理、问题管理、评论管理、讲座管理。经测试，该系统已经达到预计效果，让用户通过微信公众号就能实时的获取ACM相关信息。

**关键词：**Java；SSM；React；MySQL；ACM；微信公众号

The Design and Implementation of ACM WeChat Official Account

**Abstract**

With the continuous development of the ACM competition, major universities have gradually attached the importance to the ACM competition, and have promoted the ACM school team within the school on major social platforms. In this new media era, the WeChat official account is an effective means of communication. Most colleges and universities already have their own ACM official account, but our school has not had its own place on this platform of great influence, so I will design and implement the ACM official account to promote the ACM information of our school.

This system uses IntelliJ IDEA development tools for development, using Java language and SSM framework, front-end page uses React framework, adopts separation mode of front and back, uses HTTP protocol and JSON format data for information interaction, server uses Apache Tomcat, database uses MySQL and Redis. The system is divided into unauthorized users, ordinary users and administrators, with different functions according to different permissions. Users mainly implement login registration, forum management, commenting, uploading pictures, lecture registration, publishing questions and solutions. The administrator mainly implements user management, photo album and photo management, forum management, problem management, comment management, and lecture management. After testing, the system has achieved the expected results, allowing users to obtain ACM related information in real time through the WeChat official account.

**Key words:** Java; SSM; React; MySQL; ACM; WeChat Official account

**目 录**

论文总页数：34页

[1 引言 1](#_Toc10640315)

[1.1 课题背景 1](#_Toc10640316)

[1.2 国内外现状 1](#_Toc10640317)

[1.2.1 国内研究现状 1](#_Toc10640318)

[1.2.2 国外研究现状 1](#_Toc10640319)

[1.3 课题研究的意义 1](#_Toc10640320)

[1.4 本课题的研究方法 2](#_Toc10640321)

[2 需求分析及项目方案 2](#_Toc10640322)

[2.1 系统需求概述 2](#_Toc10640323)

[2.2 系统业务需求分析 2](#_Toc10640324)

[2.3 系统开发工具分析与选择 6](#_Toc10640325)

[2.3.1 系统开发工具及技术分析 6](#_Toc10640326)

[3 系统详细设计 7](#_Toc10640327)

[3.1 系统功能模块设计 7](#_Toc10640328)

[3.1.1 用户管理 7](#_Toc10640329)

[3.1.2 帖子管理 9](#_Toc10640330)

[3.1.3 相册管理 10](#_Toc10640331)

[3.1.4 讲座管理 11](#_Toc10640332)

[3.1.5 友链管理 12](#_Toc10640333)

[3.1.6 问题管理 12](#_Toc10640334)

[3.2 系统数据库设计 13](#_Toc10640335)

[3.2.1 数据库概念设计 13](#_Toc10640336)

[3.2.2 数据库逻辑设计 13](#_Toc10640337)

[3.3 数据库表结构详细说明 14](#_Toc10640338)

[4 系统模块功能实现 20](#_Toc10640339)

[4.1 用户管理 20](#_Toc10640340)

[4.1.1 登录 20](#_Toc10640341)

[4.1.2 用户注册 21](#_Toc10640342)

[4.1.3 关注用户 21](#_Toc10640343)

[4.1.4 用户管理 21](#_Toc10640344)

[4.2 帖子管理 21](#_Toc10640345)

[4.2.1 添加帖子 21](#_Toc10640346)

[4.2.2 管理帖子 22](#_Toc10640347)

[4.2.3 评论帖子 22](#_Toc10640348)

[4.3 相册管理 22](#_Toc10640349)

[4.3.1 相册管理 22](#_Toc10640350)

[4.3.2 上传相片 23](#_Toc10640351)

[4.3.3 删除图片 23](#_Toc10640352)

[4.4 讲座管理 23](#_Toc10640353)

[4.4.1 讲座管理 23](#_Toc10640354)

[4.4.2 讲座报名 24](#_Toc10640355)

[4.4.3 查看讲座报名人员 24](#_Toc10640356)

[4.4.4 讲座报名信息导出 24](#_Toc10640357)

[4.5 友链管理 24](#_Toc10640358)

[4.1 问题管理 26](#_Toc10640359)

[5 系统测试 26](#_Toc10640360)

[5.1 系统测试环境 26](#_Toc10640361)

[5.2 测试方法 26](#_Toc10640362)

[5.3 系统功能测试 26](#_Toc10640363)

[5.4 兼容性测试 30](#_Toc10640364)

[5.5 本章小结 30](#_Toc10640365)

[结 束 语 31](#_Toc10640366)

[参考文献 32](#_Toc10640367)

[致　　谢 33](#_Toc10640368)

[声　　明 34](#_Toc10640369)

# 引言

## 课题背景

随着科技的发展和移动设备的普及，移动互联网的发展十分迅猛，智能手机已经占领了手机市场的主要地位，同时，微信作为软件下载排行榜前列，已经成为手机的标配。据有效数据统计，微信在2018年用户超过了十亿，覆盖了90%以上的智能手机，成为中国社交软件的第一名。微信公众平台，作为微信最为重要的组成部分，以“再小的个体也有自己的品牌”为宣传口号，吸引着众多企业和个人开发者加入到微信开发阵营，争先在公众平台上创建并推广自己的品牌[1]。

随着我校ACM的不断发展，对ACM感兴趣的学生人数的增多，急需一个平台实时的让用户方便快捷的获取信息，展现我校ACM的风采，让用户参与到ACM中。现今，智能机已经成为大学生的标配，本系统利用微信公众号让用户通过微信就能查询想要的信息，降低了对设备的硬性需求。

## 国内外现状

### 国内研究现状

据最新数据显示，2018年微信的使用用户已经超多十亿人，微信公众号数量已经超过2000万，随着微信的不断普及和发展，微信公众号的不断完善，各行各业都开始利用微信公众号去宣传信息，为用户提供服务，人们的日常生活已经离不开它了。

据了解，我国高校大部分都拥有宣传本校ACM实验室的微信公众号，但是仅仅是为用户提供一些ACM信息，未能为用户提供一个讨论、交流的平台，用户能做的也仅仅是查阅信息。

### 国外研究现状

因国内外情况不同，适合国人的微信对于外国人而言并没有那么适用，而国外也有自己的社交平台比如Facebook、Twitter等软件，因此微信在国外的使用量并不大，国外用户也仅仅一亿左右。微信公众号的用户是基于微信的，因此微信公众号的使用量大不如国内[2]。国外高校对ACM的宣传更多是通过社团的Facebook等社交账号。

## 课题研究的意义

现实生活中，对ACM、数据结构感兴趣的学生没有一个方便快捷的途径去获取相关信息，仅仅只能通过QQ群、微信群去获取信息，此种方法未必能使用户获取需要的信息。通过微信公众号可以汇总信息，通过微信公众号即可获取想要的服务，同时也为算法爱好者提供了一个交流讨论的平台，让用户在接收信息的同时增强算法水平。

## 本课题的研究方法

本系统以IntelliJ IDEA开发工具进行开发，使用Apache Tomcat作为应用服务器，使用SSM框架进行后端开发，前端使用React、JavaScript和CSS进行开发，采取前后端分离的开发模式，利用HTTP协议及JSON格式数据进行数据交互，利用MySQL和Redis存储数据。利用Spring进行业务对象管理，SpringMVC进行请求转发，Mybatis实现数据对象持久化[3]。使用SSM框架利于项目管理、降低模块间代码耦合、减少了不必要的数据库连接操作[4]。该系统易维护、扩展，系统的开发满足网站开发、软件工程理念。通过对用户的需求进行系统性分析，然后对各个模块进行详细设计、功能实现，进过一系列测试、修复，最终完成该系统。

# 需求分析及项目方案

## 系统需求概述

系统业务的需求分析如下：

（1）在ACM公众号处理过程中，使用者主要分为未授权用户、授权用户和管理员。

（2）未授权用户主要针对不是本校学生，但是想了解更多我校ACM信息的用户，该种用户可以注册登录、查询信息。

（3）授权用户主要针对本校学生，通过学号大致确认用户是否为本校学生，该种用户可以登录注册、论坛管理、发表评论、上传图片、讲座报名、发布问题及题解。

（4）管理员能对该系统的信息进行管理，该种用户可以进行用户管理、相册及图片管理、论坛管理、问题管理、评论管理、讲座管理、报名用户信息查看。

## 系统业务需求分析

据上文所述，该系统用户按权限分为未授权用户、授权用户及管理员，现对三种用户分别进行阐述。

对未授权用户进行详细说明：

（1）注册登录：在第一次进入系统时，系统要求强制注册，注册后成为未授权用户。只有管理员审核用户信息审核通过后才能成为授权用户，只有注册后才能进入系统页面，进行操作。

（2）查看帖子：未授权用户进入相应页面可以查看用户发布的帖子。

（3）查看相册：未授权用户进入相应页面可以查看用户上传的图片。

（4）查看讲座：未授权用户进入相应页面可以查看管理员发布的讲座信息。

（5）查看问题：未授权用户进入相应页面可以查看用户发布的问题。

（6）查看友链：未授权用户进入相应页面可以查看管理员发布的友链信息。

未授权用户用例图如图 2‑1：



图 2‑1 未授权用户用例图

对授权用户进行详细说明；

（1）注册登录：在第一次进入系统，系统要求强制注册，注册后成为未授权用户。经管理员审核后成为授权用户。只有在登录状态下才能进入系统进行相关操作。

（2）论坛管理：用户可以发布、修改、查看相应的帖子信息，同时用户也可以评论帖子，为用户建立了一个交流的平台。

（3）相片管理：用户可以选择图片上传或删除本人上传的图片。根据相册的描述，用户可以在该相册上传符合描述的图片，同时用户也可以删除自己上传的相片。

（4）讲座管理：用户可以查看讲座信息，了解讲座内容，同时也可以报名讲座。

（5）每日一题管理：用户可以添加自己认为有意思的数据结构相关题，然后用户可以在题目下方发布自己认为正确的题解，在这可以使自己多加思考，增长对应的算法水平。

（6）查看友链：用户可以查看友链信息。

授权用户用例图如图 2‑2：



图 2‑2 授权用户用例图

对管理员进行详细说明：

（1）用户管理：管理员可以对用户信息进行管理。可以修改用户信息，查看用户注册信息，审核用户信息是否正确，正确即审核通过用户，管理员还可以删除用户。

（2）论坛管理：管理员可以管理对应讨论帖的信息。可以添加、修改、删除帖子，同时也能对帖子下方的评论进行管理，可以删除不适当的评论。

（3）相片管理：此模块分为相册管理和相片管理。首先管理员可以新增、删除相册，用户只能在管理员创建的相册内上传对应的图片，其次管理员本身也能上传相片，同时也能删除不适当的相片。

（4）讲座管理：管理员可以管理讲座信息，可以添加、修改、删除讲座信息，能截止讲座报名，同时也能查看对应的用户报名信息，根据报名信息可以开具对应的德育分加分表。

（5）每日一题管理：管理员可以管理用户发布的问题，可以添加、修改、删除用户发布的问题信息，同时也可以管理题解信息。

（6）友链管理：管理员可以管理对应的友链信息，可以添加、修改、删除友链信息。管理员用例图如图 2‑3：



图 2‑3 管理员用例图

## 系统开发工具分析与选择

本系统基于微信公众号开发者模式实现，使用IntelliJ IDEA开发工具进行开发，利用SSM框架进行后端开发，利用React进行前端开发，利用Mysql和Redis进行数据存储。

### 系统开发工具及技术分析

（1）微信公众号开发者模式

微信公众号开发模式是公众平台为开发者提供的一种自定义微信公众号的模式，当用户在微信客户端进行操作时，微信公众号服务器会将HTTP请求发送给第三方服务器，开发者可在开发者服务器上开发对应的功能模块[5]。三个服务器的交互关系如图 2‑4。



图 2‑4 微信客户端、微信公众号服务器、开发者服务器交互方式

（2）IntelliJ IDEA

IntelliJ IDEA是java编辑语言开发的集成环境，是业界公认的java语言第一开发工具，尤其在智能代码助手、代码自动提示、重构、J2EE支持、各类版本工具(git、svn等)、JUnit、CVS整合、代码分析、 创新的GUI设计等方面的功能可以说是超常的[6]。

（3）SSM框架

SSM框架是目前主流的java后端开发框架[7]。Spring方便解耦、简化开发，能有效地组织中间层，利用它进行业务对象管理[8][9]。SpringMVC能清晰地划分角色、配置功能灵活具有大量的控制器接口和实现类[10]。Mybatis是一种持久层框架，属于ORM框架，是一种半自动的框架，需要书写SQL语句、定义映射[11]。三个框架相结合，可以快速的搭建项目、迭代开发。

（4）React框架

React是现在常用的Facebook内部的JavaScript库，具有高效、灵活等特点，能够快速扩展页面应用，将页面组件化，使代码更加模块化，重用代码更简单，可维护性更高[12]。前端通过用户操作通过HTTP协议请求后端相应，后端返回JSON格式的数据，React将数据展现在页面[13][14]。

（5）MySQL和Redis

MySQL的性能卓越服务稳定，很少出现服务宕机的情况，软件体积较小，安装简单易于维护，同时支持多种操作系统，MySQL用户存储用户输入信息，将信息持久化[15][16]。Redis是基于内存的高性能key-value数据库，能够支持多种数据类型[17]。Redis利用队列技将数据库的访问变为单线程访问，消除了传统数据库串行控制的开销，同时也具有快速和持久化的特征[18]。Redis可以设置数据存活时长，可以将用户登录状态存入Redis，用Redis来记录用户登录状态。

# 系统详细设计

## 系统功能模块设计

该系统主要包含用户管理、论坛管理、相册管理、讲座模块、友链管理、每日一题六大模块。分析可得如图 3‑1功能结构图。



图 3‑1 功能结构图

### 用户管理

（1）用户注册

用户需以登录状态进入系统，如用户从未登录过会要求先注册，然后自动登录。访客根据页面的提示，填写注册要求数据，如若注册信息无误且手机号、用户名及学号唯一即注册成功，随后跳转至系统首页。如若用户填写的信息有误或格式不规范，系统会根据用户的不同错误提示不同的错误信息。流程图如图 3‑2。



图 3‑2 注册流程图

（2）用户登录

用户进入微信公众号首页前会根据本人的微信号自动登录，如若第一次登录会先强制要求注册，而后自动登录，如若已经是注册过的账号会直接进入首页。用户登录流程如图 3‑3。



图 3‑3 用户登录流程图

（3）用户管理

管理员对用户信息、用户权限进行管理，具体操作如表 3‑1。

表 3‑1 用户管理说明图

|  |  |
| --- | --- |
| 操作 | 说明 |
| 审核用户 | 对于注册用户，管理员查看用户对应信息是否正确，是则审核通过。 |
| 修改用户信息 | 管理员可以修改用户的部分信息，如用户头像等。 |
| 删除用户 | 管理员可以清除废弃账号。 |

如上所示，可得用户管理流程图如图 3‑4。



图 3‑4 用户管理流程图

（4）关注用户

用户关注其他用户，可以查看关注者发布的帖子和回复记录。

### 帖子管理

（1）添加帖子

用户可以发布讨论帖。用户输入帖子名称和内容之后，系统会对用户输入的信息进行验证，如若符合要求则发布成功，如若不符合会给予相应的提示。添加帖子流程图如图 3‑5。



图 3‑5 添加帖子流程图

（2）修改帖子

用户发表帖子后，若觉得自己发布的内容有误可进行修改，系统会对用户输入的信息进行验证，如若符合要求则发布成功，如若不符合会给予相应的提示。修改帖子的流程图如图 3‑6。



图 3‑6 修改帖子流程图

（3）评论帖子

表 3‑2 评论帖子

|  |  |
| --- | --- |
| 操作 | 说明 |
| 回复帖子 | 用户可以在评论区发表自己的看法。 |
| 回复已发表回复 | 用户可以回复讨论区他人发表的回复。 |

（4）删除帖子

用户可以删除无用或累赘的帖子。

### 相册管理

（1）相册管理

管理员登录后进入管理页面，可对相册模块进行管理，具体分为添加相册、删除相册、为相册设置封面。

具体功能说明如表 3‑3。

表 3‑3 相册管理功能说明表

|  |  |
| --- | --- |
| 操作 | 说明 |
| 添加相册 | 管理员添加相册，用户可在相册中上传图片 |
| 删除相册 | 管理员在管理页面选择一个相册信息，点击对应的删除按钮，即可删除相册。 |
| 为相册设置封面 | 管理员可为相册设置封面，方便区分相册，同时还可起汇总作用。 |

相册管理流程图如图 3‑7。



图 3‑7 相册管理流程图

（2）上传相片

用户在相册中可以上传自己认为符合相册描述的相片，提供他人观赏。系统会对用户上传的相片大小、格式进行检验，如若符合要求则上传成功，否则上传失败。上传图片流程图如图 3‑8。



图 3‑8 上传图片流程图

（3）删除相片

根据不同的权限可以删除的图片不一样，具体描述如表 3‑4。

表 3‑4 删除相片功能说明

|  |  |
| --- | --- |
| 操作 | 说明 |
| 用户删除相片 | 管理员添加相册，用户可在相册中上传图片 |
| 管理员删除相片· | 管理员在管理页面选择一个相册信息，点击对应的删除按钮，即可删除相册。 |

1. 用户可以删除与相册描述不符或上传错误的相片。
2. 管理员可删除与相册描述不符或某些不利于系统和谐的相片。

### 讲座管理

（1）讲座管理

管理员登录后进入管理页面，可对系统中讲座进行管理，具体分为添加讲座、修改讲座、删除讲座、截止讲座报名。具体功能说明如表 3‑5。

表 3‑5 讲座管理功能说明

|  |  |
| --- | --- |
| 操作 | 说明 |
| 添加讲座 | 管理员可以发布即将开展的讲座信息，用户在未截止报名前可以报名讲座。 |
| 修改讲座 | 管理员在管理页面选择一条需修改的讲座信息，点击修改按钮，进入修改页面，即可修改已经发布的讲座的信息。 |
| 截至讲座报名 | 管理员在管理页面选择一条即将截止的讲座信息，点击截止按钮，管理员可以关闭讲座的报名。 |
| 删除讲座 | 管理员在管理页面选择一条将要删除的讲座信息，点击删除按钮。管理员即可删除已发布的讲座。 |

讲座管理流程图如图 3‑9。



图 3‑9 讲座管理流程图

（2）讲座报名

用户根据讲座描述选择自己感兴趣的讲座，点击报名按钮，根据提示获取报名信息。

（3）查看讲座报名人员

管理员点击对应按钮，即可查看报名用户的信息。管理员可以利用对应的报名人员信息。

（4）讲座报名信息导出

管理员可以将报名讲座用户的姓名学号导出一个excel表，管理员可以通过excel表里的信息进行额外的操作。

### 友链管理

管理员进入管理页面可对友链信息进行管理。详细分为：添加友链、修改友链、删除友链。

1. 添加友链：管理员可以添加觉得有意义的友链。
2. 修改友链：管理员可以修改已发布的友链信息。
3. 删除友链：管理员可以删除不需要的友链。

### 问题管理

用户登录后可对问题模块进行管理，具体分为添加问题、修改问题、添加题解、删除问题。

1. 添加问题：用户可在问题模块分享、记录自己觉得有意义的数据结构题目。
2. 修改问题：用户可修改发布的问题。
3. 添加题解：用户可以分享自己对该题目的看法。
4. 删除问题：用户可以删除本人发布的问题。

流程图如图 3‑10。



图 3‑10 问题管理流程图

## 系统数据库设计

### 数据库概念设计

根据上文对系统所做的分析和设计，可以得到关系如图 3‑11。



图 3‑11 整体关系图

（1）1个用户可以管理多个相册。

（2）1个用户可以上传多个相片。

（3）1个相册包含多个相片。

（4）1个用户可以报名多个讲座。

（5）1个用户可以管理多个友链。

（6）1个用户可以管理多个帖子。

（7）1个用户可以发表多个评论。

（8）1个帖子可以拥有多个评论。

（9）1个用户可以发布多个问题。

（10）1个用户可以发布多个题解。

（11）1个问题可以拥有多个题解。

### 数据库逻辑设计

本系统中，数据表关系如图 3‑12。



图 3‑12 数据库关系图

如图 3‑12所示：user表是用户信息表，invitation是帖子信息表，comment是帖子评论信息表，album是相册信息表，album\_photo是相册相片关联表，photo是相片信息表，lecture是讲座信息表，applyLecture是用户参加讲座信息表，problem是问题信息表，replyProblem是题解信息表，friendUrl是友链信息表，follow是关注信息表。

## 数据库表结构详细说明

本系统涉及十张数据库表，分别是：用户信息表（user），帖子信息表（invitation），评论信息表（comment），相册信息表（album），相片信息表（photo），相册相片关联表（album\_photo），讲座信息表（lecture），用户参加讲座表（appleLecture），问题信息表（problem），题解表（answer），友链信息表（friendUrl）和关注表(follow)。

系统表说明如表 3‑6。

表 3‑6 系统表说明

|  |  |  |
| --- | --- | --- |
| 序号 | 表名称 | 说明 |
| 1 | user | 用户信息表，用于存储用户相关信息 |
| 2 | invitation | 帖子信息表，用户存储帖子基本信息 |
| 3 | comment | 评论信息表，用于存储用户评论帖子的主要内容 |
| 4 | album | 相册信息表，用于存储相册基本信息 |
| 5 | album\_photo | 相册相片信息关联表，用于存储相片和相册的归属信息 |
| 6 | photo | 相片信息表，用于存储用户上传图片后图片的基本信息 |
| 7 | lecture | 讲座信息表，用于存储讲座相关的基本信息表 |
| 8 | applyLecture | 用户参与讲座表，用于存储用户参加讲座的基本信息 |
| 9 | problem | 问题信息表，用于存储用户发表的问题的基本信息 |
| 10 | answer | 题解表，用于存储用户回复问题的题解信息 |
| 11 | friendUrl | 友链信息表，用于存储发布的友链信息 |
| 12 | follow | 关注信息表，用于存储用户关注信息 |

1. 用户信息表（user）：用于存储用户相关信息。除用户名、密码、真实姓名、学号、手机号等用户基本信息，还需存储用户的权限。用户在微信公众号登录时，会通过用户在微信的唯一编号，因此还需存储微信用户唯一编号。为了避免误操作等情况，再删除时并不会直接删除数据，而是将数据的某字段设置为对应的数值，标志不可见，防止误操作。

用户信息表表结构如表 3‑7。

表 3‑7 用户信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型 | 规则 | 注释 |
| userId | int | pk, not null | 用户编号，自增 |
| username | varchar(50) | not null | 用户昵称 |
| password | varchar(50) | not null | 用户密码,6-20位 |
| mobile | varchar(20) | not null | 手机号 |
| studentId | varchar(50) | not null | 学号，10位 |
| openId | varchar(50) | not null | 微信用户唯一编号 |
| createDay | datetime | not null | 创建时间 |
| image | varchar(50) | not null | 用户头像，用户头像的url地址 |
| auth | int | not null | 用户权限，0-未授权，1-授权，>3管理员 |
| isEffective | int | not null | 是否删除，1-存活，0-删除，默认值1 |

1. 帖子信息表（invitation）：用于存储讨论相关信息。讨论帖需知道讨论帖的标题、内容、创建人、创建时间这三种关键信息，同时因为在该系统下讨论帖能加精、置顶，因此还需记录讨论帖加精和置顶的两种状态。讨论帖发布后，还可能对讨论帖进行操作，因此需记录他的更新时间和更新人，以及当前的存活状态。

帖子信息表表结构如表 3‑8。

表 3‑8 帖子信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型 | 规则 | 注释 |
| invitationId | bigint | pk, not null | 帖子编号，随机生成数字 |
| invitationTitle | varchar(50) | not null | 帖子标题 |
| invitationBody | text | not null | 帖子内容 |
| createUser | int | not null | 创建人 |
| createDate | datetime | not null | 创建时间 |
| updateDate | datetime | not null | 更新时间 |
| updateUser | int | not null | 更新人 |
| isFrist | int | not null | 是否置顶，0-未置顶，1-置顶，默认值0 |
| isGreate | int | not null | 是否加精，0-未加精，1-加精，默认值0 |
| isEffective | int | not null | 是否删除，1-存活，0-删除，默认值1 |

1. 评论信息表（comment）：用于存储帖子的评论信息。用户可以评论帖子或回复他人评论，因此我们需知道评论的帖子编号和他人评论的评论编号，同时我们须知道评论人和评论时间，以及当前的存活状态。

评论信息表表结构如表 3‑9。

表 3‑9 评论信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型 | 规则 | 注释 |
| invitationId | bigint | pk, not null | 帖子编号，随机生成数字 |
| invitationTitle | varchar(50) | not null | 帖子标题 |
| invitationBody | text | not null | 帖子内容 |
| createUser | int | not null | 创建人 |
| createDate | datetime | not null | 创建时间 |
| isEffective | int | not null | 是否删除，1-存活，0-删除 |

1. 相册信息表（album）：存储系统相册相关信息。用户添加相册的标题、描述等信息，同时需记录相册创建人、创建时间以及当前的存活状态。

相册信息表结构如表 3‑10。

表 3‑10 相册信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型 | 规则 | 注释 |
| albumId | bigint | pk, not null | 相册编号,自增 |
| albumTitle | varchar(50) | not null | 相册标题 |
| createUser | int | not null | 创建人 |
| createDate | datetime | not null | 创建时间 |
| description | varchar(500) | not null | 描述 |
| isPublic | int | not null | 是否公，,0-私人，1-公开 |
| isEffective | int | not null | 是否删除，1-存活，0-删除 |

1. 相册相片关联表（album\_photo）：用于存储用户上传的每张图片与相册的对应信息。

相册相片关联表表结构如表 3‑11。

表 3‑11 相册相片信息关联表

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型 | 规则 | 注释 |
| apId | bigint | pk, not null | 关联编号，自增 |
| albumId | bigint | fk, not null | 相册编号，上传的相册编号 |
| photoId | bigint | fk, not null | 相片编号，上传的相片编号 |
| createUser | int | not null | 创建人 |
| createDate | datetime | not null | 创建时间 |
| isEffective | int | not null | 是否删除，1-存活，0-删除 |

1. 相片信息表（photo）：用于存储上传的图片信息。用户上传图片后系统会自动给相片设置相片名称，需记录上传人、上传时间、当前相片的公开状态以及当前的存活状态。相片信息表表结构如表 3‑12。

表 3‑12 相片信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型 | 规则 | 注释 |
| photoId | bigint | pk, not null | 相片编号 |
| photoName | varchar(50) | not null | 相片名称 |
| createUser | int | not null | 创建人 |
| createDate | datetime | not null | 创建时间 |
| isPublic | int | not null | 是否公开，0-未公开，1-公开 |
| photoUrl | varchar(100) | not null | 上传后相片的url |
| isEffective | int | not null | 是否删除，1-存活，0-删除 |

1. 讲座信息表（lecture）：用于存储讲座相关信息。用户添加讲座，输入对应的讲座标题、内容以及开始时间，管理员可以截止用户报名，因此还需记录用户创建状态、更新状态以及活动状态。

讲座信息表表结构如表 3‑13。

表 3‑13 讲座信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型 | 规则 | 注释 |
| lectureId | bigint | pk, not null | 讲座编号，随机生成数字 |
| lectureTitle | varchar(50) | not null | 讲座名称 |
| lectureBody | text | not null | 讲座内容 |
| createUser | int | not null | 创建人 |
| createDate | datetime | not null | 创建时间 |
| updateUser | int | not null | 更新人 |
| updateDate | datetime | not null | 更新时间 |
| isDone | int | not null | 是否结束，0-未结束，1-结束 |
| isEffective | int | not null | 是否删除，1-存活，0-删除 |

1. 用户报名信息表（applyLecture）：用于存储用户对应的讲座报名情况。用户报名讲座之后需记录对应的报名关系，因此需记录讲座编号、参与者编号、创建信息以及存活状态。

用户报名表表结构如表 3‑14。

表 3‑14 讲座报名信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型 | 规则 | 注释 |
| applyLectureId | bigint | pk, not null | 关联编号，自增 |
| lectureId | bigint | fk. not null | 讲座编号 |
| joinUser | bigint | fk, not null | 参与者编号 |
| createDate | datetime | not null | 创建时间 |
| isEffective | int | not null | 是否删除，1-存活，0-删除 |

1. 问题信息表（problem）：用于存储用户发表、分享的数据结构题目信息。用户输入问题标题、内容以及创建者的题解，除创建信息外，还需知道被查看次数和现存活状态。

问题信息表表结构如表 3‑15。

表 3‑15 问题信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型 | 规则 | 注释 |
| problemId | bigint | pk, not null | 问题编号，自增 |
| problemTitle | varchar(50) | not null | 问题名称 |
| problemBody | text | not null | 问题内容 |
| createUser | int | not null | 创建人 |
| createDate | datetime | not null | 创建时间 |
| updateUser | int | not null | 更新人 |
| updateDate | datetime | not null | 更新时间 |
| readNum | int | not null | 被查看次数 |
| isEffective | int | not null | 是否删除，1-存活，0-删除 |

1. 题解信息表（replyProblem）：用于存储用户对某题目的观点和题解信息。题解信息表表结构如表 3‑16。

表 3‑16 题解信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型 | 规则 | 注释 |
| problemansId | bigint | pk, not null | 关联编号，自增 |
| problemId | bigint | not null | 问题编号，回答问题的编号 |
| joinUser | bigint | not null | 回答者编号 |
| ansBody | text | not null | 题解内容 |
| createDate | datetime | not null | 创建时间 |
| isEffective | int | not null | 是否删除，1-存活，0-删除 |

1. 友链信息表（friendUrl）：用于存储用户添加的友链的标题和网址等相关信息。友链信息表结构如表 3‑17。

表 3‑17 友链信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型 | 规则 | 注释 |
| friendurlId | bigint | pk, not null | 友链编号，自增 |
| friendTitle | varchar(50) | not null | 友链名称 |
| friendBody | text | not null | 友链内容 |
| createUser | int | not null | 创建人 |
| createDate | datetime | not null | 创建时间 |
| updateUser | int | not null | 更新人 |
| updateDate | datetime | not null | 更新时间 |
| isEffective | int | not null | 是否删除，1-存活，0-删除 |

（12）关注信息表（follow）：用户存储用户关注其他用户的相关信息。

关注信息表结构如表 3‑18。

表 3‑18 关注信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 列名 | 数据类型 | 规则 | 注释 |
| followId | bigint | pk, not null | 关注编号，自增 |
| beUserId | int | not null | 被关注用户编号 |
| createUser | int | not null | 关注用户编号 |
| createDate | datetime | not null | 创建时间 |
| isEffective | int | not null | 是否删除，1-存活，0-删除 |

# 系统模块功能实现

## 用户管理

### 登录

用户权限不同，用户登录登录后进入的界面不同

1. 普通用户登录：用户点击微信公众号下方的菜单栏，微信服务器传递对应的用户的code值，后端服务器会根据code值去微信服务器获取用户对应的openId，再去数据库中查询是否有对应的数据。
2. 管理员登录：管理员进入管理后台登录页面，输入用户密码之后，前端对输入数据进行格式校验，校验无误调用后端login接口，根据用户输入的username和password去查询数据库中是否有对应的数据，如若有，进入管理后台首页，反之提示错误信息。

用户登录后，将用户对应的登录信息存到Redis中，返回token值给前端，前端每次请求都在HTTP协议中的“Authority”存放token值，后端通过token值判断登录状态。

代码段如下：

/\*\*

\* 通过用户id生成对应的token值，存入redis当中

\* @param userId 指定用户的id

\* @return \*/

public TokenModel createToken(long userId) {

//使用uuid作为源token

//String token = UUID.randomUUID().toString().replace("-", "");

String token = UUIDUtil.get32UUID();//获取随机字符串

TokenModel model = new TokenModel(userId, token);

redis.set(token+"", userId+"", SysConst.TOKEN\_TIME);//将对应关系存入redis

return model;

}

### 用户注册

首次登录的用户会要求填写注册信息，填写完成后首先判断用户填写数据格式是否符合规范，此后调用Ajax请求访问后台接口，进行数据校验，避免某些信息重复。

### 关注用户

用户点击其他用户头像可以选择关注该用户，关注后可以查看关注者发布的帖子和回复记录。首先后端会再次校验被关注用户是否存在，再判断是否已经被关注过，如若符合要求，即关注成功。

### 用户管理

用户进入用户管理页面，能对用户的权限、信息进行对应的管理。

（1）审核用户：用户注册后，对应的用户信息会显示在用户管理界面，在注册时会对用户信息格式进行校验，而学号是否正确需管理员审核，如若信息都正确，即可审核通过。

（2）修改用户信息：点击修改按钮，管理员进入详情页面，可输入对应的用户信息，然后点击修改，执行updateUser方法，修改成功会弹出提示框。

（3）删除用户：管理员点击删除按钮，后端执行deleteUser接口，首先会判断该用户是否真实存在，然后将用户存活状态设为死亡。

效果图如图 4‑1。



图 4‑1 用户管理效果图

## 帖子管理

### 添加帖子

用户进入添加帖子界面后，输入帖子标题和内容，点击提交按钮会执行add函数，添加界面会对用户输入的信息进行长度校验，校验成功后调用后端接口将输入数据存入数据库中，后端会根据不同的运行结果返回不同的提示语给页面展示，添加成功后会回到帖子列表页面。

### 管理帖子

1. 删除帖子：点击删除按钮可以删除自己不想存留的帖子，执行deleteInvitation接口，后端接口会进行安全校验，首先判断该帖子是否存在，如何将该条数据设置为死亡状态，而后将运行结果返回给前端。
2. 修改帖子内容：点击修改按钮，跳转至修改页面，用户输入修改后的标题和内容，点击提交按钮，前端首先对数据进行校验，符合条件即调用后端接口，调用成功更新数据库数据。

### 评论帖子

（1）评论帖子：用户可在评论区对帖子进行评论。

（2）评论回复：用户可以回复他人的评论。

以上两种情况调用同一个接口，但是传递的参数不同，利用p\_commentId代表是否是评论的他人回复，而后将数据存入数据库。

## 相册管理

### 相册管理

1. 添加相册：管理员输入对应的相册信息，前端页面对数据进行格式校验，而后Ajax调用后端接口，将数据存入数据库中。
2. 删除相册：管理员对遗弃的相册可以采取删除操作，后端首先会对相册是否存在进行校验，而后改变相册的存活状态。
3. 将图片设为相册封面：新增的相册是有一个默认的封面，管理员可以将某张相片设为该相册的封面。效果图如图 4‑2。

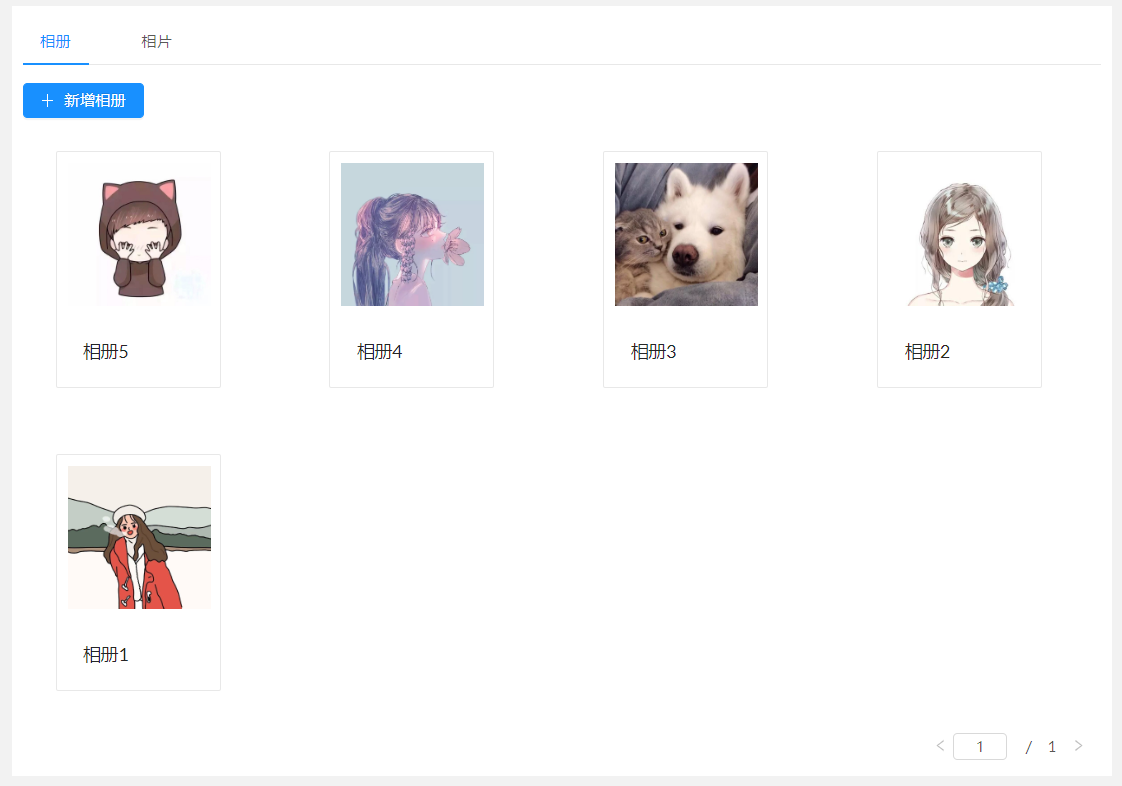


图 4‑2 相册管理效果图

### 上传相片

用户点击上传按钮，弹出文件选择框，选择图片，后端会将图片存执对应的文件夹下。

代码段如下：

/\*\*

\* file是上传的文件

\*/

@RequestMapping(value = "/uploadImg", method = RequestMethod.POST)

@ResponseBody

public Object uploadImg(@RequestParam(value = "myFileName",required = false)

MultipartFile file) throws Exception{

if (file.isEmpty()) {

LOG.error("文件为空空");

return null;

}

String fileName = file.getOriginalFilename(); // 文件名

String suffixName = fileName.substring(fileName.lastIndexOf(".")); // 后缀名

String filePath = "D://pic/"; // 上传后的路径

fileName = UUID.randomUUID() + suffixName; // 新文件名

File dest = new File(filePath + fileName);

if (!dest.getParentFile().exists()) { //判断上级文件是否存在

dest.getParentFile().mkdirs();

}

file.transferTo(dest);

Map<String, Object> map = new HashMap<>(); //以map形式返回

map.put("errno", 0);

String url = "http://"+ SysConst.localhost+"/image/"; //显示路径

map.put("data", url+fileName);

return map;

}

### 删除图片

用户对于不想要的相片可以采取删除操作。普通用户只能删除自己提交的图片，而管理员可以删除所有图片。

## 讲座管理

### 讲座管理

（1）添加讲座：管理员输入相应的讲座的讲座信息，前端会对输入的讲座信息的长度进行校验，而后调用后端的addLecture接口，将讲座信息存入数据库中。

（2）修改讲座信息：进入修改页面，调用detailLecture接口会将对应的讲座信息展示在输入框中，用户修改完成后点击修改按钮会调用updateLecture接口，修改对应的讲座信息，将数据存入数据库。

（3）截止讲座报名：管理员进入管理页面，点击截止按钮，会调用后端的doneLecture接口，修改对应讲座的状态值，将更新后的数据修改至数据库。

（4）删除讲座：管理员对遗弃的讲座可以采取删除操作，后端首先会对讲座是否存在进行校验，而后改变讲座的存活状态。

### 讲座报名

用户进入讲座信息详情页面，调用detailLecture接口将显示详细信息，点击信息下方的报名按钮，调用applyLecture接口，将对应的信息存入数据库中。

### 查看讲座报名人员

管理员可以查看对应讲座的报名人员，点击按钮后会调用personLecture接口，将对应页面的人员信息展示在界面上。

### 讲座报名信息导出

管理员查看讲座的对应报名信息，可以将某个讲座的所有报名用户的姓名和学号放入excel里面导出。利用java中集成的操作excel的库对excel进行操作，先读取数据库中该场讲座的报名用户信息，遍历存入excel表中，并通过浏览器下载到本地。效果图如图 4‑3。



图 4‑3 报名信息效果图

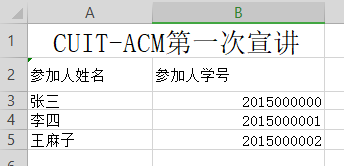


图 4‑4 excel表效果图

## 友链管理

1. 添加友链：管理员输入对应的友链标题和网址，前端对输入框的内容进行格式校验，而后调用addFriendurl接口，会校验输入的网址是否正确，而后将数据存入数据库中。
2. 修改友链：管理员进入修改页面，首先会调用detailFriendurl接口获取友链的具体信息，将信息展示在页面上，而后修改输入框的数据，调用updateFriendurl接口，会校验输入的网址是否正确，而后更新数据。
3. 删除友链：管理员可以对遗弃的友链采取删除操作，后端首先会对友链的存在状态进行判断，而后改变存活状态。

代码段如下：

/\*\*

\* 检验网址是否有效 \*/

public synchronized bool isConnect(String urlStr) {

int counts = 0;

if (urlStr == null || urlStr.length() <= 0) {

return null;

}

while (counts < 5) {  //5次连接不上认为无法连接，认为不是有效友链

try {

url = new URL(urlStr);

con = (HttpURLConnection) url.openConnection();

state = con.getResponseCode();

if (state == 200) {  //200即url可访问

return true;

}

break;

}catch (Exception ex) {

counts++;

urlStr = null; continue;

}

}

return false;

}

效果图如图 4‑5。

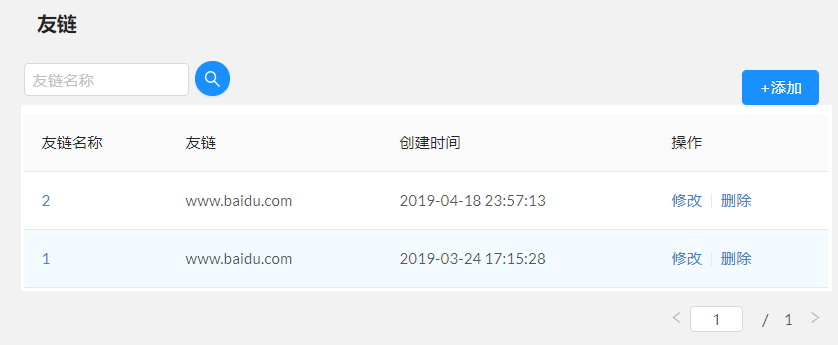


图 4‑5 友链管理效果图

## 问题管理

（1）添加问题：管理员输入对应的问题信息，前端页面对问题信息进行校验，如若有误弹出提示信息，随后调用后端addProblem接口，将数据存入数据库中。

（2）修改问题：管理员进入修改页面，首先调用detailProblem接口将问题详细信息展示在页面上，随后用户在输入框中输入修改后的信息，点击修改按钮，将数据更新至数据库。

（3）添加题解：用户添加问题完成后有显示成功页面，会让你选择是否添加自己的题解，点击后可在输入框中输入自己的题解，调用addAns接口将数据存入数据库中。

（4）删除问题：管理员对遗弃的问题可以采取删除操作，后端首先会对问题是否存在进行校验，而后改变问题的存活状态。

效果图如图 4‑6。



图 4‑6 问题管理效果图

# 系统测试

## 系统测试环境

为了保证该系统能够正常运行，在代码完成后对系统进行了一系列测试，其中主要包括系统测试和兼容性测试。该系统的测试环境是java、maven、Mysql、redis和nodeJ。

## 测试方法

本系统主要使用黑盒测试和单元测试两种测试方法。

黑盒测试又称为功能测试、数据驱动测试或基于规格说明书的测试，是一种从用户观点出发的测试，本系统利用边界值分析法、场景法等方法检验系统功能是否正确[19]。

## 系统功能测试

该测试主要是对模块的功能进行测试，确保所有功能正常运行、满足用户需求。测试主要针对用户管理、论坛、相片管理、讲座管理、每日一题等模块进行功能测试。

（1）用户管理模块测试如表 5‑1。

表 5‑1 用户管理测试模块

|  |  |  |  |
| --- | --- | --- | --- |
| 系统 | ACM微信公众号——用户管理 | 版本 | 1.0 |
| 功能模块名 | 用户管理 | 用例编号 | WX-01 |
| 测试人员 | 本人 | 测试日期 | 2019.05.12 |
| 功能特性 | 用户的登录、注册、关注 | | |
| 测试目的 | 验证系统用户的正确性 | | |
| 前置条件 | 管理员登录。 | | |
| 测试步骤 | 1.注册用户，查看用户是否注册成功  2.用户通过微信公众号或者密码登录是否登陆成功  3.用户关注另一用户查看是否关注成功 | | |
| 期望结果 | 1.用户注册成功  2.用户登录成功  3.用户关注成功 | | |
| 实际结果 | 与期望结果一致。 | | |
| 结论 | 用户管理功能正确。 | | |

1. 论坛模块测试如表 5‑2。

表 5‑2 论坛管理测试模块

|  |  |  |  |
| --- | --- | --- | --- |
| 系统 | ACM微信公众号——论坛管理 | 版本 | 1.0 |
| 功能模块名 | 论坛管理 | 用例编号 | WX-02 |
| 测试人员 | 本人 | 测试日期 | 2019.05.12 |
| 功能特性 | 讨论帖的添加、修改、回复、删除、显示、删除回复 | | |
| 测试目的 | 验证讨论帖操作的正确性。 | | |
| 前置条件 | 管理员登录。 | | |
| 测试步骤 | 1.添加讨论帖，查看讨论帖添加是否成功  2.修改讨论帖，查看讨论帖显示及修改是否成功  3.回复讨论帖，校验用户回复帖子和回复他人评论是否正确  4.删除讨论帖，查看讨论帖删除是否成功  5.查看讨论帖，查看讨论帖对应信息是否正确。  6.删除回复，查看对应用户回复是否删除 | | |
| 期望结果 | 1.添加讨论帖成功  2.修改讨论帖成功  3.回复讨论帖成功  4.删除讨论帖成功  5.讨论帖显示正确  6.删除回复成功 | | |
| 实际结果 | 与期望结果一致。 | | |
| 结论 | 讨论帖管理功能正确。 | | |

1. 相册管理测试如表 5‑3。

表 5‑3 相册管理测试

|  |  |  |  |
| --- | --- | --- | --- |
| 系统 | ACM微信公众号——相册管理 | 版本 | 1.0 |
| 功能模块名 | 相册管理 | 用例编号 | WX-03 |
| 测试人员 | 本人 | 测试日期 | 2019.05.12 |
| 功能特性 | 相册的添加、删除，相片的上传、查看、删除 | | |
| 测试目的 | 验证相册操作的正确性。 | | |
| 前置条件 | 管理员登录。 | | |
| 测试步骤 | 1.添加相册，查看相册是否添加成功  2.删除相册，查看相册是否删除成功  3.上传相片，查看相册是否上传成功  4.查看相册下图片，相册和相片对应关系一致  5.删除相片，查看相片是否删除成功 | | |
| 期望结果 | 1.添加相册成功  2.删除相册成功  3.上传相片成功  4.查看相册下图片成功  5.删除相片成功 | | |
| 实际结果 | 与期望结果一致。 | | |
| 结论 | 相册管理功能正确。 | | |

（4）讲座模块测试如表 5‑4。

表 5‑4 讲座管理测试

|  |  |  |  |
| --- | --- | --- | --- |
| 系统 | ACM微信公众号——讲座管理 | 版本 | 1.0 |
| 功能模块名 | 讲座管理 | 用例编号 | WX-04 |
| 测试人员 | 本人 | 测试日期 | 2019.05.12 |
| 功能特性 | 讲座的添加、修改、删除、报名、信息表查看 | | |
| 测试目的 | 验证讲座管理的正确性。 | | |
| 前置条件 | 管理员登录。 | | |
| 测试步骤 | 1.添加讲座，查看讲座是否添加成功  3.删除讲座，查看讲座是否删除成功  4.报名讲座，查看报名讲座是否成功  5.查看讲座报名表，查看报名表名单是否正确 | | |
| 期望结果 | 1.添加讲座成功  3.删除讲座成功  4.报名讲座成功  5.查看讲座报名表正确 | | |
| 实际结果 | 与期望结果一致。 | | |
| 结论 | 讲座管理功能正确。 | | |

1. 友链模块测试如表 5‑5。

表 5‑5 友链管理测试

|  |  |  |  |
| --- | --- | --- | --- |
| 系统 | ACM微信公众号——友链管理 | 版本 | 1.0 |
| 功能模块名 | 友链管理 | 用例编号 | WX-05 |
| 测试人员 | 本人 | 测试日期 | 2019.05.12 |
| 功能特性 | 友链的添加、修改、删除 | | |
| 测试目的 | 验证友链管理的正确性。 | | |
| 前置条件 | 管理员登录。 | | |
| 测试步骤 | 1.添加友链，查看友链是否添加成功  2.修改友链，查看友链是否修改成功  3.删除友链，查看友链是否删除成功 | | |
| 期望结果 | 1.添加友链成功  2.修改友链成功  3.删除友链成功 | | |
| 实际结果 | 与期望结果一致。 | | |
| 结论 | 友链管理功能正确。 | | |

（6）每日一题模块测试如表 5‑6。

表 5‑6 每日一题测试模块

|  |  |  |  |
| --- | --- | --- | --- |
| 系统 | ACM微信公众号——问题管理 | 版本 | 1.0 |
| 功能模块名 | 问题管理 | 用例编号 | WX-06 |
| 测试人员 | 本人 | 测试日期 | 2019.05.12 |
| 功能特性 | 问题的添加、修改、删除，题解的添加、查看、修改 | | |
| 测试目的 | 验证问题管理的正确性。 | | |
| 前置条件 | 管理员登录。 | | |
| 测试步骤 | 1.添加问题，查看问题是否添加成功  2.修改问题，查看问题是否修改成功  3.删除问题，查看问题是否删除成功  4.添加题解，查看题解是否添加成功  5.查看题解，题解与问题对应  6.修改题解，修改创建者的题解 | | |
| 期望结果 | 1.添加问题成功  2.修改问题成功  3.删除问题成功  4.添加题解成功  5.查看题解与问题对应  6.修改题解成功 | | |
| 实际结果 | 与期望结果一致。 | | |
| 结论 | 题解管理功能正确。 | | |

## 兼容性测试

同一个系统在不同的操作系统、不同的环境、不同的浏览器下会有不用的显示效果，因此需对该系统进行兼容性测试，以防出现显示或效果偏差[20]。

此处的兼容性测试主要测试在安卓系统下的微信公众号和IOS系统下的微信公众号显示效果是否存在差异，同时也测试了不同浏览器下管理员管理界面的展示效果是否相同。测试结果：通过。

## 本章小结

经测试，各个模块的功能正常运行，前端页面在不同的系统、不同浏览器下的展示效果基本一致，符合用户基本需求。综上所述，本系统符合要求。

结 束 语

本系统根据我校ACM校队的实际需求实现了ACM微信公众号，基本实现了对用户、帖子、评论、讲座、相册、问题的管理过程。为对ACM感兴趣的学生提供了一个分享、讨论平台。关注公众号后，通过微信就能获取自己想要的信息，报名想参加的讲座，同时也能通过手机上传自己觉得有意义的图片。

经测试，该系统基本满足了用户的业务需求和功能设计，具有较高的稳定性和容错性，系统界面友好，能对用户操作起指导性作用。现系统稳定运行，已达到预期期望。

在开发过程中，使用的是模块化开发，降低了模块与模块间的代码耦合，同时也提取了某些方法的公共特征，避免代码重复。同时也使用了当前主流的SSM框架及React框架。

通过本次毕业设计的设计开发，增强了代码能力和学习能力，对软件开发的过程有了较为清晰的认知，对现在主流的java框架和前端框架更为熟悉，能够更好地适应未来工作。

参考文献

1. 李单, 李梦瑶, 沈月娥. 微信公众平台的品牌传播模式研究[J]。 新闻研究导刊, 2016, 7(13):330.
2. 黄宠. 公共图书馆公众号的用户使用意愿影响因素分析[D]. 2017.
3. 刘昊, 李民. 基于SSM框架的客户管理系统设计与实现[J]. 软件导刊, 2017,16(07):87-89.
4. 战锴. 软件工程技术发展趋势研究[J]. 电子技术与软件工程,2013,13(14):51.
5. 兰欣. 微信公众平台CMS的设计与实现[D]. 南昌大学,2015.
6. 陈立兵. IntelliJ IDEA——开发人员利器[J]. Intelli JIDE中文爱好者站点, 2009(01):118-120..
7. 邢东旭,王峰. 基于Spring框架的Web系统的研究与应用[J]. 内蒙古科技大学学报. 2012.
8. 汪云飞. Java EE开发的颠覆者Spring Boot实战[M]. 电子工业出版社, 2016.
9. 梁弼,王光琼,邓小清.基于Spring框架的Web应用轻量级3S解决方案[J].西华大学学报(自然科学版),2018,37(03):78-82.
10. Paul Deck. Spring MVC学习指南[M]. 人民邮电出版社, 2017.
11. 刘增辉. MyBatis从入门到精通[M]. 电子工业出版社, 2017.
12. Azat Mardan. 快速上手React编程[M]. 清华大学出版社, 2018.
13. 许兆伟,王颖.HTTP/2新特性与Web性能关系的研究与实验[J].计算机技术与发展,2017,27(11):192-195.
14. 仇小花,秦栓栓,邱果.基于WEB开发中的XML与JSON数据传输格式研究[J].信息技术与信息化,2017(04):123-125.
15. 李仕伟, 周坤, 刘新蕊, 李宝林. MySQL数据库优化技术[J]. 信息与电脑(理论版),2016(12):173-174.
16. 杨凯 .浅谈基于MySQL数据库结构设计[J]. 科技展望,2015,25(36):104-105.
17. 邱书洋. Redis缓存技术研究及应用[D]. 2016.
18. 姚经纬, 冷文浩. 基于Redis的主模型管理系统设计与实现[J].软件导刊,2017,16(05):75-79.
19. 杨金丽. 软件测试方法研究及其在DM平台中的应用[D]. 2006.
20. 路海英. Web测试技术研究与应用[D]. 北京邮电大学,2011.

致　　谢

首先诚挚的感谢我的指导老师杜晓宇，感谢他在百忙之中对我的毕业设计进行指导、论文进行修正。同时也感谢教过我的老师，感谢他们为我打下了牢靠的基础，让我能独立完成该毕业设计。

最后向在百忙之中评审本文的各位专家、老师表示衷心的感谢！

声　　明

本论文的工作是2018年10月至2019年5月在成都信息工程大学软件工程学院完成的。文中除了特别加以标注地方外，不包含他人已经发表或撰写过的研究成果，也不包含为获得成都信息工程大学或其他教学机构的学位或证书而使用过的材料。

关于学位论文使用权和研究成果知识产权的说明：

本人完全了解成都信息工程大学有关保管使用学位论文的规定，其中包括：

（1）学校有权保管并向有关部门递交学位论文的原件与复印件。

（2）学校可以采用影印、缩印或其他复制方式保存学位论文。

（3）学校可以学术交流为目的复制、赠送和交换学位论文。

（4）学校可允许学位论文被查阅或借阅。

（5）学校可以公布学位论文的全部或部分内容（保密学位论文在解密后遵守此规定）。

除非另有科研合同和其他法律文书的制约，本论文的科研成果属于成都信息工程大学。

特此声明！

作者签名：

年 月 日