

**湖南工程职业技术学院**

**毕业设计**

产品设计说明书

**题 目**  学智博客的设计与实现

**毕业设计类型** ☑ 产品设计 □ 工艺设计 □ 方案设计

**专 业 名 称**  软件技术

**所 属 班 级**  2018173801

**学 生 姓 名**  黄朝

**学 生 学 号**  201817380127

**指 导 老 师 （学校）**  胡瑶

**指 导 老 师 （企业）**  詹熊

**湖南工程职业技术学院教务处 制**

目录

[插图索引 III](#_Toc51316596)

[附表索引 IV](#_Toc51316597)

[第1章 项目概况 1](#_Toc51316598)

[1.1 设计背景 1](#_Toc51316599)

[1.2 设计意义 1](#_Toc51316600)

[1.3 设计思路 2](#_Toc51316601)

[第2章 需求分析 3](#_Toc51316602)

[2.1 需求描述 3](#_Toc51316603)

[2.2 用例建模 3](#_Toc51316604)

[2.3 用例描述 4](#_Toc51316605)

[2.3.1 博客文章管理用例描述 4](#_Toc51316606)

[2.3.2 用户信息管理用例描述 5](#_Toc51316607)

[2.3.3 评论信息管理用例描述 6](#_Toc51316608)

[2.3.4 举报记录管理用例描述 7](#_Toc51316609)

[第3章 系统设计 8](#_Toc51316610)

[3.1 数据库设计 8](#_Toc51316611)

[3.1.1 概念模型 10](#_Toc51316612)

[3.1.2 逻辑模型 11](#_Toc51316613)

[3.1.3 物理模型 12](#_Toc51316614)

[3.1.4 数据库脚本 13](#_Toc51316615)

[3.2 系统架构设计 26](#_Toc51316616)

[3.3 模块设计 27](#_Toc51316617)

[3.3.1 博客文章信息类图 27](#_Toc51316618)

[3.3.2 博客文章信息操作流程 28](#_Toc51316619)

[3.3.3 用户信息类图 29](#_Toc51316620)

[3.3.4 用户信息操作流程 30](#_Toc51316621)

[3.3.5 评论信息类图 31](#_Toc51316622)

[3.3.6 评论信息操作流程 32](#_Toc51316623)

[3.3.7 举报记录类图 33](#_Toc51316624)

[3.3.8 举报记录操作流程 34](#_Toc51316625)

[第4章 产品实现 35](#_Toc51316626)

[4.1 博客文章模块实现 35](#_Toc51316627)

[4.1.1 界面设计 35](#_Toc51316628)

[4.1.2 功能实现 36](#_Toc51316629)

[4.2 用户信息模块实现 37](#_Toc51316630)

[4.2.1 界面设计 37](#_Toc51316631)

[4.2.2 功能实现 39](#_Toc51316632)

[4.3 评论信息模块实现 40](#_Toc51316633)

[4.3.1 界面设计 40](#_Toc51316634)

[4.3.2 功能实现 41](#_Toc51316635)

[4.4 举报记录模块实现 42](#_Toc51316636)

[4.4.1 界面设计 42](#_Toc51316637)

[4.4.2 功能实现 43](#_Toc51316638)

[第5章 产品测试 44](#_Toc51316639)

[5.1 测试方法与策略 44](#_Toc51316640)

[1. 功能测试 44](#_Toc51316641)

[2. 数据项测试 44](#_Toc51316642)

[3. 业务测试 44](#_Toc51316643)

[5.2 博客文章信息模块测试 44](#_Toc51316644)

[5.3 用户信息模块测试 46](#_Toc51316645)

[5.4 评论信息模块测试 48](#_Toc51316646)

[5.5 举报记录模块测试 49](#_Toc51316647)

[第6章 结论 51](#_Toc51316648)

[6.1 毕业设计成果特点 51](#_Toc51316649)

[6.2 设计成果的实用价值或应用前景 51](#_Toc51316650)

[6.3 不足之处或遗留未予解决的问题 51](#_Toc51316651)

插图索引

[图 2-1 学智博客网用例模型图 4](#_Toc51254269)

[图 3-1 学智博客网概念模型图 11](#_Toc51254270)

[图 3-2 学智博客网逻辑模型图 12](#_Toc51254271)

[图 3-3 学智博客网物理模型图 13](#_Toc51254272)

[图 3-4 学智博客网架构设计图 27](#_Toc51254273)

[图 3-5 博客文章信息类图 28](#_Toc51254274)

[图 3-6 博客文章信息时序图 29](#_Toc51254275)

[图 3-7 用户信息类图 30](#_Toc51254276)

[图 3-8 用户信息时序图 31](#_Toc51254277)

[图 3-9 评论信息类图 32](#_Toc51254278)

[图 3-10 评论信息时序图 33](#_Toc51254279)

[图 3-11 举报记录类图 34](#_Toc51254280)

[图 3-12 举报记录时序图 34](#_Toc51254281)

[图 4-1 博客文章列表界面图 35](#_Toc51254282)

[图 4-2 博客详情界面图 35](#_Toc51254283)

[图 4-3 博客文章修改界面图 36](#_Toc51254284)

[图 4-4 博客文章主要代码图 37](#_Toc51254285)

[图 4-5 用户主页界面图 38](#_Toc51254286)

[图 4-6 用户基本设置界面图 38](#_Toc51254287)

[图 4-7 修改用户密码界面图 38](#_Toc51254288)

[图 4-8 用户信息模块主要代码图 39](#_Toc51254289)

[图 4-9 评论列表界面图 40](#_Toc51254290)

[图 4-10 发布评论界面图 40](#_Toc51254291)

[图 4-11 修改评论界面图 41](#_Toc51254292)

[图 4-12 评论信息模块主要代码图 42](#_Toc51254293)

[图 4-13 评论举报界面图 42](#_Toc51254294)

[图 4-14 举报原因界面图 43](#_Toc51254295)

[图 4-15 用户信息模块主要代码图 43](#_Toc51254296)

附表索引

[表 2-1 博客文章信息用例描述表 4](#_Toc51254250)

[表 2-2 用户信息用例描述表 5](#_Toc51254251)

[表 2-3 评论管理信息用例描述表 6](#_Toc51254252)

[表 2-4 举报记录管理用例描述表 7](#_Toc51254253)

[表 3-1 Users表（用户信息表） 8](#_Toc51254254)

[表 3-2 Admins表（管理员表） 8](#_Toc51254255)

[表 3-3 Articles（博客文章表） 8](#_Toc51254256)

[表 3-4 ArticleToCategories表（文章类别表） 9](#_Toc51254257)

[表 3-5 BlogCategories表（分类表） 9](#_Toc51254258)

[表 3-6 Comments表（文章评论表） 9](#_Toc51254259)

[表 3-7 ReplyComments表（评论回复表） 9](#_Toc51254260)

[表 3-8 Fans表（粉丝表） 9](#_Toc51254261)

[表 3-9 Links表（链接表） 10](#_Toc51254262)

[表 3-10 CommentReports表（评论举报表） 10](#_Toc51254263)

[表 3-11 ArticleCollects表（文章收藏表） 10](#_Toc51254264)

[表 5-1 博客文章信息模块测试用例表 44](#_Toc51254265)

[表 5-2 用户信息测试用例表 46](#_Toc51254266)

[表 5-3 用户信息测试用例表 48](#_Toc51254267)

[表 5-4 举报记录测试用例表 49](#_Toc51254268)

# 项目概况

## 设计背景

互联网的浪潮不断席卷人们的生活，人们对知识的获取要求越来越高，学智博客网站就是为IT专业技术人员提供良好的交流平台，本网站只限于交流与计算机软件技术相关行业的知识，不断更新新的版本，从用户的角度出发不断寻求新的发展空间，为用户带来更好的体验度。学智博客网站就是本着技术共享、精品原创、使得IT行业 的专家在此大展身手。用户可以对感兴趣的文章进行收藏、点赞、评论等，用户还可以通过搜索提取关键字在站内获取自己感兴趣的博主或者文章。有了随时记录思想火花的内存，有了随时了解IT行业最新动态的一体化网站，有了自己是知识库，个人公告等等。不断挖掘平台潜能，学智博客正朝着高品质的IT媒体与IT服务社区方向蓬勃发展。  
  计算机科学技术引发了全世界科技时代的大更新，知识更新速度也很快，快餐文化加速了人们对文化认知的新高度，如何在科技文化潮流背景下取之精华，弃之糟粕也是对人们思想高度和综合素质的一大考验。学智博客在此背景下以服务为核心，为广大技术人员提供最大最基础最高效“三最”知识分享网站,使得IT行业的高手能够在此大展身手，分享技术、分享生活。助力文化技术交流，过滤淤泥文化，逐步发展成为最大的IT技术交流平台，构建完整的基础体系IT行业高质量向前发展。

## 设计意义

随着科学技术的不断提高，国内外对IT行业的技术要求越来越高，许许多多的程序员需要有更高的竞争力才能避免被行业淘汰。改革开发以来，电子信息技术的变革席卷全国，人们对信息技术有了新的认知，各行各业都对电子信息技术有所涉及，在一定程度上推动了全国经济的发展。随着我国经济不断的发展，物质生活水平不断的提高，人们开始对知识领域不断扩展，寻求新的解决办法，人们不再是向以前一样到达目的要求即可，而是对技术的要求更快，更准、更安全。加上我国对高职院校的大力推关，对技术人员的要求和市场资源更高更多，职业培训也越来越多，IT行业也是其中之一。为此，学智博客也行动了起来，为整个IT行业提供良好的技术交流平台，无障碍沟通，不断更新知识库，每个人都可以享受互联网浪潮带来的激情。如何更好的让用户选择对自己技术有帮助的内容是学智博客今后要发展的方向，让用户体验度更高更好更便捷是学智博客发展的初衷。绝大部分的博客平台分类复杂、知识体系难以辨认，用户很难找到自己想要获取的信息，无法从中得到对用户有意义的数据。互联网交流难免会出现年龄段不一的用户，如何避免糟粕，取值精华，让互联网交流与浏览更加干净、绿色也是绝大部分博客网站需要考虑的问题。为适应企业信息化策略的要求，来实现学智博客网站的信息化管理。

## 设计思路

本系统目前的主要对象是IT行业的技术人员。

本系统分为九大模块，分别是登录注册模块、博客搜索模块、个人信息模块、用户管理模块、评论管理模块、举报记录模块、文章管理模块、管理员管理模块、后台系统模块。

登录注册模块：用于用户和管理员的登录。

博客搜索模块：查询不同的博客文章、搜索框搜索其他用户的功能。

个人信息模块：查看个人详细信息、个人公告信息。

用户管理模块：查询博客用户功能、修改个人基本信息。

评论管理模块：用户发表评论信息、针对评论的回复以及回复的回复、删除个人发布的评论。

举报记录模块：查询用户评论举报信息、反馈举报内容、审核举报记录、删除举报记录。

文章管理模块：查询个人发布的文章、编辑个人文章、查看文章详情、删除个人文章。

管理员管理：管理员查询、设置密码。

后台系统模块：对文章管理、用户管理、管理员管理、链接管理、评论管理的数据维护和多条件模糊查询。

设计开发流程如下：

1. 需求分析：包括需求调研，系统功能，性能分析，业务流程分析。

2. 概要设计：包括系统架构设计，功能模块设计，数据库设计。

3. 详细设计：包括方法设计、方法的封装、数据结构设计、属性使用设计。

4. 系统实现：包括编写代码，系统单元测试。

5. 系统测试：对项目功能进行系统测试。

# 需求分析

## 需求描述

系统是基于多用户的博客平台，每个用户都可单独添加文章类别；管理员能够对后台各功能模块管理进行数据维护。

管理员通过权限操作对系统的功能模块进行增删改查，对用户发布的文章、评论进行审核、可通过多条件模糊查询进行各功能模块的数据获取；实现了用户可设置编辑个人基本信息，如上传头像、修改个人说明、修改登录密码等基础操作。用户还可以对不当言论进行举报、管理员进行审核对此被举报账号进行冻结操作。除此之外，用户可自行设置是否对发布的文章开启或关闭评论区域，查看个人公告信息，加入学智博客的具体时间，不断提高对用户的体验度。

## 用例建模

本系统中主要包括两类用户：普通用户、管理员用户，其中可以对这几类用户说明下。

根据前面的分析，可以得到用户信息模块、博客文章、评论信息、举报记录模块的总体用例图如下图2-1所示:



图 2-1 学智博客网用例模型图

## 用例描述

### 博客文章管理用例描述

文章信息：管理员登录后可以查看全部博客文章信息，可以预览或删除博客文章或是删除选中的文章信息。其具体的用例描述如下表2-1所示：

表 2-1 博客文章信息用例描述表

|  |  |
| --- | --- |
| 用例编号 | **CASE01** |
| 用例名称 | 博客文章管理 |
| 用例描述 | 系统管理员对博客文章的信息进行CRUD操作 |
| 前置条件 | 系统管理员成功登录系统 |
| 后置条件 | 系统管理员在系统中浏览了文章信息或删除了系统中已经存在的文章或查看全部文章的详情 |
| 活动步骤 | 1-系统管理员登录系统  2-系统显示博客文章列表  3-系统管理员在文章信息列表，点击【删除】按钮，弹出提示框是否删除，删除文章信息，显示删除后的文章信息列表，点击【批量删除】按钮，弹出模态框提示是否删除所选文章。  4-系统管理员点击【预览文章】按钮，跳转至文章详情页面查看文章具体详细内容。  5-系统管理员在文本框内输入查询内容，点击【搜索】按钮，显示与查询内容对应的文章信息列表。 |
| 扩展点 | 无 |
| 数据格式 | 博客文章信息（博客ID） |

### 用户信息管理用例描述

用户基本信息：管理员登录后可以查看全部用户信息，可以修改用户信息或是删除选中的用户信息。其具体的用例描述如下表2-2所示：

表 2-2 用户信息用例描述表

|  |  |
| --- | --- |
| 用例编号 | **CASE02** |
| 用例名称 | 用户信息管理 |
| 用例描述 | 系统管理员对用户的信息进行CRUD操作 |
| 前置条件 | 系统管理员成功登录系统 |
| 后置条件 | 系统管理员在系统中修改用户信息或删除用户信息以及查看全部用户的信息 |
| 活动步骤 | 1-系统管理员登录系统  2-系统显示用户信息列表  3-系统管理员点击【修改】按钮，弹出新修改界面，系统管理员输入用户更改后的信息，点击【保存】按钮，系统对输入的信息进行验证，将合法的信息保存，然后显示修改后的用户列表。  4-系统管理员在用户列表中选择某个用户，点击【删除】，弹出是否删除此用户提示框，点击【确定删除】，系统会将当前用户信息删除，并返回到用户信息列表，显示删除后的用户信息列表。  5-系统管理员在用户列表中选择选中的用户，点击【批量删除】，弹出是否删除此用户提示框，点击【确定删除】，系统会将当前用户信息删除，并返回到用户信息列表，显示删除后的用户信息列表。  6-系统管理员在用户列表中，点击【冻结】，弹出是否冻结提示框，当点击确定冻结后，系统将当前用户账号冻结，该账号无法进行登录，并返回到用户信息列表，显示冻结成功。  6-系统管理员在用户列表文本框中输入内容，点击【搜索】，显示与文本框内容对应的用户信息。 |
| 扩展点 | 无 |
| 数据格式 | 用户信息（用户ID） |

### 评论信息管理用例描述

评论管理信息：管理员登录后可以查看全部评论信息，可以批量删除选中的评论信息或删除单个评论信息，查询评论信息。其具体的用例描述如下表2-3所示：

表 2-3 评论管理信息用例描述表

|  |  |
| --- | --- |
| 用例编号 | **CASE03** |
| 用例名称 | 评论信息管理 |
| 用例描述 | 系统管理员对评论的信息进行CRUD操作 |
| 前置条件 | 系统管理员成功登录系统 |
| 后置条件 | 系统管理员在系统中全部评论的信息 |
| 活动步骤 | 1-系统管理员登录系统  2-系统显示评论管理列表  3-系统管理员点击【违规】按钮，弹出是否移除评论提示框，点击【确定移除】按钮，系统会将当前违规的评论信息移除，并返回评论信息列表，显示移除后的评论信息列表。  4-系统管理员在评论管理列表中选择某个评论，点击【删除】，弹出是否删除此评论提示框，点击【确定删除】，系统会将当前评论信息删除，并返回到评论信息列表，显示删除后的评论信息列表。  5-系统管理员在评论列表中选择选中的评论，点击【批量删除】，弹出是否删除此评论提示框，点击【确定删除】，系统会将当前评论信息删除，并返回到评论信息列表，显示删除后的评论信息列表。  6-系统管理员在评论列表文本框中输入内容，点击【搜索】，显示与文本框内容对应的用户信息。 |
| 扩展点 | 无 |
| 数据格式 | 评论管理信息（评论信息ID） |

### 举报记录管理用例描述

举报记录信息：管理员登录后可以查看全部举报信息，可以批量删除选中的举报信息或删除单个举报信息，查询举报记录。其具体的用例描述如下表2-4所示：

表 2-4 举报记录管理用例描述表

|  |  |
| --- | --- |
| 用例编号 | **CASE04** |
| 用例名称 | 举报记录管理 |
| 用例描述 | 系统管理员对举报的记录进行CRUD操作 |
| 前置条件 | 系统管理员成功登录系统 |
| 后置条件 | 系统管理员在系统中全部举报的记录 |
| 活动步骤 | 1-系统管理员登录系统  2-系统显示举报记录管理列表  3-系统管理员点击【是否冻结此账号】按钮，弹出是否冻结提示框，点击【确定冻结】按钮，系统会将当前被举报用户进行举报，并返回评论举报记录管理列表，显示已被冻结的账号。  4-系统管理员在举报记录管理列表中选择某个举报信息，点击【删除】，弹出是否删除此举报记录提示框，点击【确定删除】，系统会将当前举报记录删除，并返回到举报记录管理列表，显示删除后的举报记录管理列表。  5-系统管理员在举报记录管理列表中选择选中的举报记录，点击【批量删除】，弹出是否删除此举报基本提示框，点击【确定删除】，系统会将当前举报记录删除，并返回到举报记录列表，显示删除后的举报记录列表。  6-系统管理员在举报记录管理列表文本框中输入内容，点击【搜索】，显示与文本框内容对应的举报记录。 |
| 扩展点 | 无 |
| 数据格式 | 评论管理信息（评论信息ID） |

# 系统设计

根据学智博客网站的需求分析结果，本章主要设计了学智博客网站的数据库设计，以数据字典的形式一一罗列出表自身的组成结构，并使用PowerDesigner工具设计了概念模型、逻辑模型和物理模型，客观清晰的展示了表与表之间的关系；本系统采用MVC架构、仓储设计模型、划分了系统的功能模块使用UML类图和UML时序图详细描述了各功能的设计思路。

## 数据库设计

学智博客网站的数据库共设计了十一张数据表，分别为用户信息表、管理员表、博客文章表、文章类别表、类别表、文章评论表、评论回复表、粉丝表、链接表、评论举报表、文章收藏表。各表的具体结构如下表3-1到3-11所示：

表 3-1 Users表（用户信息表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 中文名 | 类别 | 长度 | 允许为空 | 主键 |
| Id | 用户编号 | uniqueidentifier | 16 |  | Y |
| Eamil | 用户邮箱 | varchar | 40 |  |  |
| NickName | 用户昵称 | varchar | 20 |  |  |
| Password | 用户密码 | varchar | 30 |  |  |
| ImagePath | 用户头像 | varchar | 300 |  |  |
| PersonalDescription | 粉丝数 | nvarchar | -1 | Y |  |
| FansCount | 关注数 | int | 10 |  |  |
| FocusCount | 注册时间 | int | 10 |  |  |
| CreateTime | 是否删除 | datetime | 23 |  |  |
| IsRemoved | 是否冻结 | bit | 1 |  |  |

表 3-2 Admins表（管理员表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 中文名 | 类别 | 长度 | 允许为空 | 主键 |
| Id | 管理员编号 | uniqueidentifier | 16 |  | Y |
| Account | 账号 | varchar | 20 |  |  |
| Password | 密码 | varchar | 30 |  |  |
| CreateTime | 注册时间 | datetime | 23 |  |  |
| IsRemoved | 是否删除 | bit | 1 |  |  |

表 3-3 Articles（博客文章表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 中文名 | 类别 | 长度 | 允许为空 | 主键 |
| Id | 文章编号 | uniqueidentifier | 16 |  | Y |
| Title | 文章标题 | nvarchar | -1 |  |  |
| Content | 文章内容 | ntext | 1073741823 |  |  |
| UserId | 用户外键编号 | uniqueidentifier | 16 |  |  |
| GoodCounnt | 推荐数 | int | 10 |  |  |
| BadCount | 反对数 | int | 10 |  |  |
| State | 发布状态 | bit | 1 |  |  |
| CreateTime | 发布时间 | datetime | 23 |  |  |
| IsRemoved | 是否删除 | bit | 1 |  |  |
| IsClosingComments | 是否关闭评论 | bit | 1 |  |  |

表 3-4 ArticleToCategories表（文章类别表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 中文名 | 类别 | 长度 | 允许为空 | 主键 |
| Id | 文章类别编号 | uniqueidentifier | 16 |  | Y |
| BlogCategoryId | 分类外键编号 | uniqueidentifier | 16 |  |  |
| ArticleId | 文章外键编号 | uniqueidentifier | 16 |  |  |
| CreateTime | 添加时间 | datetime | 23 |  |  |
| IsRemoved | 是否删除 | bit | 1 |  |  |

表 3-5 BlogCategories表（分类表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 中文名 | 类别 | 长度 | 允许为空 | 主键 |
| Id | 分类编号 | uniqueidentifier | 16 |  | Y |
| CategoryName | 分类名称 | nvarchar | -1 | Y |  |
| UserId | 用户编号 | uniqueidentifier | 16 |  |  |
| CreateTime | 创建时间 | datetime | 23 |  |  |
| IsRemoved | 是否删除 | bit | 1 |  |  |

表 3-6 Comments表（文章评论表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 中文名 | 类别 | 长度 | 允许为空 | 主键 |
| Id | 评论编号 | uniqueidentifier | 16 |  | Y |
| Content | 评论内容 | nvarchar | 600 |  |  |
| UserId | 用户编号 | uniqueidentifier | 16 |  |  |
| ArticleId | 文章外键编号 | uniqueidentifier | 16 |  |  |
| CreateTime | 添加时间 | datetime | 23 |  |  |
| IsRemoved | 是否删除 | bit | 1 |  |  |

表 3-7 ReplyComments表（评论回复表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 中文名 | 类别 | 长度 | 允许为空 | 主键 |
| Id | 回复编号 | uniqueidentifier | 16 |  | Y |
| ReplierId | 回复用户编号 | uniqueidentifier | 16 |  |  |
| TargetToReplyId | 回复目标用户编号 | uniqueidentifier | 16 |  |  |
| ReplyType | 回复类型 | int | 10 |  |  |
| CommentParentId | 回复评论编号 | uniqueidentifier | 16 |  |  |
| ReplyToTargetCommentId | 回复目标评论编号 | uniqueidentifier | 16 |  |  |
| ReplyContent | 回复内容 | nvarchar | -1 |  |  |
| CreateTime | 添加时间 | datetime | 23 |  |  |
| IsRemoved | 是否删除 | bit | 1 |  |  |

表 3-8 Fans表（粉丝表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 中文名 | 类别 | 长度 | 允许为空 | 主键 |
| Id | 粉丝编号 | uniqueidentifier | 16 |  | Y |
| UserId | 关注用户编号 | uniqueidentifier | 16 |  |  |
| FocusUserId | 用户编号 | uniqueidentifier | 16 |  |  |
| CreateTime | 具体时间 | datetime | 23 |  |  |
| IsRemoved | 是否删除 | bit | 1 |  |  |

表 3-9 Links表（链接表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 中文名 | 类别 | 长度 | 允许为空 | 主键 |
| Id | 链接编号 | uniqueidentifier | 16 |  | Y |
| Title | 链接标题 | varchar | 30 |  |  |
| Url | 链接地址 | varchar | 50 |  |  |
| Describe | 链接描述 | varchar | 200 |  |  |
| CreateTime | 添加时间 | datetime | 23 |  |  |
| IsRemoved | 是否删除 | bit | 1 |  |  |

表 3-10 CommentReports表（评论举报表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 中文名 | 类别 | 长度 | 允许为空 | 主键 |
| Id | 举报编号 | uniqueidentifier | 16 |  | Y |
| UserId | 评论内容 | uniqueidentifier | 16 |  |  |
| CommentId | 用户编号 | uniqueidentifier | 16 |  |  |
| Content | 评论编号 | nvarchar | -1 | Y |  |
| CreateTime | 管理员是否审核 | datetime | 23 |  |  |
| IsRemoved | 是否确认 | bit | 1 |  |  |
| IsHandle | 用户是否确认 | bit | 1 |  |  |
| IsConfirm | 添加时间 | bit | 1 |  |  |
| IsBeReportUserConfirm | 是否删除 | bit | 1 |  |  |

表 3-11 ArticleCollects表（文章收藏表）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 中文名 | 类别 | 长度 | 允许为空 | 主键 |
| Id | 评论编号 | uniqueidentifier | 16 |  | Y |
| UserId | 用户编号 | uniqueidentifier | 16 |  |  |
| ArticleId | 文章外键编号 | uniqueidentifier | 16 |  |  |
| CreateTime | 添加时间 | datetime | 23 |  |  |
| IsRemoved | 是否删除 | bit | 1 |  |  |

学智博客网站的博客文章模块和用户管理信息模块的设计与实现主要使用到了上表的3-1,3-2,3-3,3-8,3-11。其中3-10表主要用于举报记录信息模块,3-6、3-7表主要用于评论管理信息模块。

### 概念模型

根据前文的数据字典的描述，通过使用PowerDesigner工具进一步设计了数据库的概念模型结构，展现出各表的自身结构以及各表与表之间的关联。学智博客网站的概念模型如下图3-1所示:

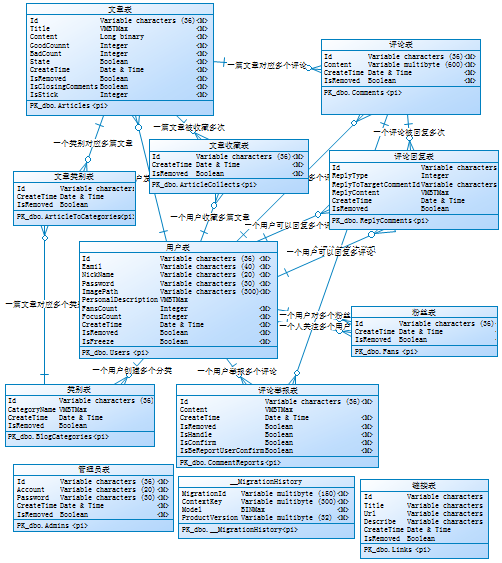


图 3-1 学智博客网概念模型图

### 逻辑模型

根据前文的数据字典的描述，通过使用PowerDesigner工具进一步设计了数据库的逻辑模型结构，展现出各表的自身结构以及各表与表之间的关联。

学智博客网站的逻辑模型如下图3-2所示:

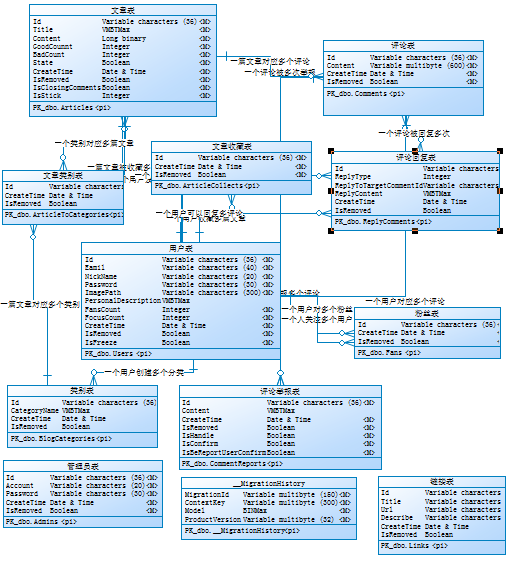


图 3-2 学智博客网逻辑模型图

### 物理模型

利用PowerDesigner工具，在上节的逻辑模型基础上考虑各种具体的技术实现因素，进行数据库体系结构设计，真正实现数据在数据库中的存放。学智博客网的物理模型如下图3-3所示:

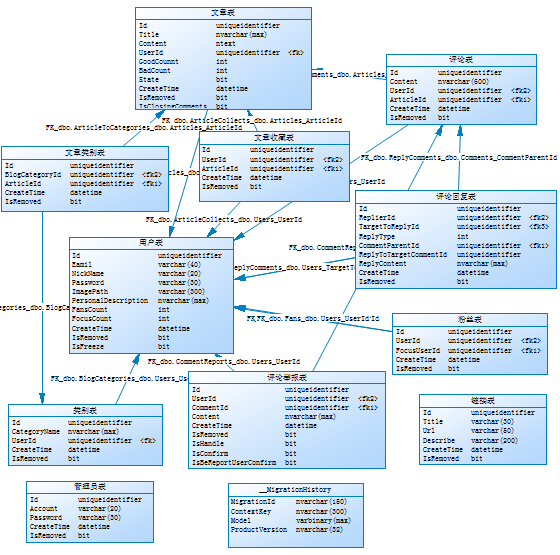


图 3-3 学智博客网物理模型图

### 数据库脚本

|  |
| --- |
| USE [master]  GO  CREATE DATABASE [BlogSystem]  CONTAINMENT = NONE  ON PRIMARY  ( NAME = N'BlogSystem', FILENAME = N'C:\Program Files (x86)\Microsoft SQL Server\MSSQL11.MSSQLSERVER\MSSQL\DATA\BlogSystem.mdf' , SIZE = 4160KB , MAXSIZE = UNLIMITED, FILEGROWTH = 1024KB )  LOG ON  ( NAME = N'BlogSystem\_log', FILENAME = N'C:\Program Files (x86)\Microsoft SQL Server\MSSQL11.MSSQLSERVER\MSSQL\DATA\BlogSystem\_log.ldf' , SIZE = 1040KB , MAXSIZE = 2048GB , FILEGROWTH = 10%)  GO  ALTER DATABASE [BlogSystem] SET COMPATIBILITY\_LEVEL = 110  GO  IF (1 = FULLTEXTSERVICEPROPERTY('IsFullTextInstalled'))  begin  EXEC [BlogSystem].[dbo].[sp\_fulltext\_database] @action = 'enable'  end  GO  ALTER DATABASE [BlogSystem] SET ANSI\_NULL\_DEFAULT OFF  GO  ALTER DATABASE [BlogSystem] SET ANSI\_NULLS OFF  GO  ALTER DATABASE [BlogSystem] SET ANSI\_PADDING OFF  GO  ALTER DATABASE [BlogSystem] SET ANSI\_WARNINGS OFF  GO  ALTER DATABASE [BlogSystem] SET ARITHABORT OFF  GO  ALTER DATABASE [BlogSystem] SET AUTO\_CLOSE OFF  GO  ALTER DATABASE [BlogSystem] SET AUTO\_CREATE\_STATISTICS ON  GO  ALTER DATABASE [BlogSystem] SET AUTO\_SHRINK OFF  GO  ALTER DATABASE [BlogSystem] SET AUTO\_UPDATE\_STATISTICS ON  GO  ALTER DATABASE [BlogSystem] SET CURSOR\_CLOSE\_ON\_COMMIT OFF  GO  ALTER DATABASE [BlogSystem] SET CURSOR\_DEFAULT GLOBAL  GO  ALTER DATABASE [BlogSystem] SET CONCAT\_NULL\_YIELDS\_NULL OFF  GO  ALTER DATABASE [BlogSystem] SET NUMERIC\_ROUNDABORT OFF  GO  ALTER DATABASE [BlogSystem] SET QUOTED\_IDENTIFIER OFF  GO  ALTER DATABASE [BlogSystem] SET RECURSIVE\_TRIGGERS OFF  GO  ALTER DATABASE [BlogSystem] SET DISABLE\_BROKER  GO  ALTER DATABASE [BlogSystem] SET AUTO\_UPDATE\_STATISTICS\_ASYNC OFF  GO  ALTER DATABASE [BlogSystem] SET DATE\_CORRELATION\_OPTIMIZATION OFF  GO  ALTER DATABASE [BlogSystem] SET TRUSTWORTHY OFF  GO  ALTER DATABASE [BlogSystem] SET ALLOW\_SNAPSHOT\_ISOLATION OFF  GO  ALTER DATABASE [BlogSystem] SET PARAMETERIZATION SIMPLE  GO  ALTER DATABASE [BlogSystem] SET READ\_COMMITTED\_SNAPSHOT ON  GO  ALTER DATABASE [BlogSystem] SET HONOR\_BROKER\_PRIORITY OFF  GO  ALTER DATABASE [BlogSystem] SET RECOVERY FULL  GO  ALTER DATABASE [BlogSystem] SET MULTI\_USER  GO  ALTER DATABASE [BlogSystem] SET PAGE\_VERIFY CHECKSUM  GO  ALTER DATABASE [BlogSystem] SET DB\_CHAINING OFF  GO  ALTER DATABASE [BlogSystem] SET FILESTREAM( NON\_TRANSACTED\_ACCESS = OFF )  GO  ALTER DATABASE [BlogSystem] SET TARGET\_RECOVERY\_TIME = 0 SECONDS  GO  EXEC sys.sp\_db\_vardecimal\_storage\_format N'BlogSystem', N'ON'  GO  USE [BlogSystem]  GO  /\*\*\*\*\*\* Object: Table [dbo].[\_\_MigrationHistory] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  SET ANSI\_NULLS ON  GO  SET QUOTED\_IDENTIFIER ON  GO  SET ANSI\_PADDING ON  GO  CREATE TABLE [dbo].[\_\_MigrationHistory](  [MigrationId] [nvarchar](150) NOT NULL,  [ContextKey] [nvarchar](300) NOT NULL,  [Model] [varbinary](max) NOT NULL,  [ProductVersion] [nvarchar](32) NOT NULL,  CONSTRAINT [PK\_dbo.\_\_MigrationHistory] PRIMARY KEY CLUSTERED  (  [MigrationId] ASC,  [ContextKey] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]  GO  SET ANSI\_PADDING OFF  GO  /\*\*\*\*\*\* Object: Table [dbo].[Admins] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  SET ANSI\_NULLS ON  GO  SET QUOTED\_IDENTIFIER ON  GO  SET ANSI\_PADDING ON  GO  CREATE TABLE [dbo].[Admins](  [Id] [uniqueidentifier] NOT NULL,  [Account] [varchar](20) NOT NULL,  [Password] [varchar](30) NOT NULL,  [CreateTime] [datetime] NOT NULL,  [IsRemoved] [bit] NOT NULL,  CONSTRAINT [PK\_dbo.Admins] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY]  GO  SET ANSI\_PADDING OFF  GO  /\*\*\*\*\*\* Object: Table [dbo].[ArticleCollects] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  SET ANSI\_NULLS ON  GO  SET QUOTED\_IDENTIFIER ON  GO  CREATE TABLE [dbo].[ArticleCollects](  [Id] [uniqueidentifier] NOT NULL,  [UserId] [uniqueidentifier] NOT NULL,  [ArticleId] [uniqueidentifier] NOT NULL,  [CreateTime] [datetime] NOT NULL,  [IsRemoved] [bit] NOT NULL,  CONSTRAINT [PK\_dbo.ArticleCollects] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Table [dbo].[Articles] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  SET ANSI\_NULLS ON  GO  SET QUOTED\_IDENTIFIER ON  GO  CREATE TABLE [dbo].[Articles](  [Id] [uniqueidentifier] NOT NULL,  [Title] [nvarchar](max) NOT NULL,  [Content] [ntext] NOT NULL,  [UserId] [uniqueidentifier] NOT NULL,  [GoodCounnt] [int] NOT NULL,  [BadCount] [int] NOT NULL,  [State] [bit] NOT NULL,  [CreateTime] [datetime] NOT NULL,  [IsRemoved] [bit] NOT NULL,  [IsClosingComments] [bit] NOT NULL,  [IsStick] [int] NOT NULL,  CONSTRAINT [PK\_dbo.Articles] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Table [dbo].[ArticleToCategories] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  SET ANSI\_NULLS ON  GO  SET QUOTED\_IDENTIFIER ON  GO  CREATE TABLE [dbo].[ArticleToCategories](  [Id] [uniqueidentifier] NOT NULL,  [BlogCategoryId] [uniqueidentifier] NOT NULL,  [ArticleId] [uniqueidentifier] NOT NULL,  [CreateTime] [datetime] NOT NULL,  [IsRemoved] [bit] NOT NULL,  CONSTRAINT [PK\_dbo.ArticleToCategories] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Table [dbo].[BlogCategories] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  SET ANSI\_NULLS ON  GO  SET QUOTED\_IDENTIFIER ON  GO  CREATE TABLE [dbo].[BlogCategories](  [Id] [uniqueidentifier] NOT NULL,  [CategoryName] [nvarchar](max) NULL,  [UserId] [uniqueidentifier] NOT NULL,  [CreateTime] [datetime] NOT NULL,  [IsRemoved] [bit] NOT NULL,  CONSTRAINT [PK\_dbo.BlogCategories] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Table [dbo].[CommentReports] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  SET ANSI\_NULLS ON  GO  SET QUOTED\_IDENTIFIER ON  GO  CREATE TABLE [dbo].[CommentReports](  [Id] [uniqueidentifier] NOT NULL,  [UserId] [uniqueidentifier] NOT NULL,  [CommentId] [uniqueidentifier] NOT NULL,  [Content] [nvarchar](max) NULL,  [CreateTime] [datetime] NOT NULL,  [IsRemoved] [bit] NOT NULL,  [IsHandle] [bit] NOT NULL,  [IsConfirm] [bit] NOT NULL,  [IsBeReportUserConfirm] [bit] NOT NULL,  CONSTRAINT [PK\_dbo.CommentReports] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Table [dbo].[Comments] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  SET ANSI\_NULLS ON  GO  SET QUOTED\_IDENTIFIER ON  GO  CREATE TABLE [dbo].[Comments](  [Id] [uniqueidentifier] NOT NULL,  [Content] [nvarchar](600) NOT NULL,  [UserId] [uniqueidentifier] NOT NULL,  [ArticleId] [uniqueidentifier] NOT NULL,  [CreateTime] [datetime] NOT NULL,  [IsRemoved] [bit] NOT NULL,  CONSTRAINT [PK\_dbo.Comments] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Table [dbo].[Fans] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  SET ANSI\_NULLS ON  GO  SET QUOTED\_IDENTIFIER ON  GO  CREATE TABLE [dbo].[Fans](  [Id] [uniqueidentifier] NOT NULL,  [UserId] [uniqueidentifier] NOT NULL,  [FocusUserId] [uniqueidentifier] NOT NULL,  [CreateTime] [datetime] NOT NULL,  [IsRemoved] [bit] NOT NULL,  CONSTRAINT [PK\_dbo.Fans] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Table [dbo].[Links] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  SET ANSI\_NULLS ON  GO  SET QUOTED\_IDENTIFIER ON  GO  SET ANSI\_PADDING ON  GO  CREATE TABLE [dbo].[Links](  [Id] [uniqueidentifier] NOT NULL,  [Title] [varchar](30) NOT NULL,  [Url] [varchar](50) NOT NULL,  [Describe] [varchar](200) NOT NULL,  [CreateTime] [datetime] NOT NULL,  [IsRemoved] [bit] NOT NULL,  CONSTRAINT [PK\_dbo.Links] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY]  GO  SET ANSI\_PADDING OFF  GO  /\*\*\*\*\*\* Object: Table [dbo].[ReplyComments] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  SET ANSI\_NULLS ON  GO  SET QUOTED\_IDENTIFIER ON  GO  CREATE TABLE [dbo].[ReplyComments](  [Id] [uniqueidentifier] NOT NULL,  [ReplierId] [uniqueidentifier] NOT NULL,  [TargetToReplyId] [uniqueidentifier] NOT NULL,  [ReplyType] [int] NOT NULL,  [CommentParentId] [uniqueidentifier] NOT NULL,  [ReplyToTargetCommentId] [uniqueidentifier] NOT NULL,  [ReplyContent] [nvarchar](max) NOT NULL,  [CreateTime] [datetime] NOT NULL,  [IsRemoved] [bit] NOT NULL,  CONSTRAINT [PK\_dbo.ReplyComments] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Table [dbo].[Users] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  SET ANSI\_NULLS ON  GO  SET QUOTED\_IDENTIFIER ON  GO  SET ANSI\_PADDING ON  GO  CREATE TABLE [dbo].[Users](  [Id] [uniqueidentifier] NOT NULL,  [Eamil] [varchar](40) NOT NULL,  [NickName] [varchar](20) NOT NULL,  [Password] [varchar](30) NOT NULL,  [ImagePath] [varchar](300) NOT NULL,  [PersonalDescription] [nvarchar](max) NULL,  [FansCount] [int] NOT NULL,  [FocusCount] [int] NOT NULL,  [CreateTime] [datetime] NOT NULL,  [IsRemoved] [bit] NOT NULL,  [IsFreeze] [bit] NOT NULL,  CONSTRAINT [PK\_dbo.Users] PRIMARY KEY CLUSTERED  (  [Id] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  ) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]  GO  SET ANSI\_PADDING OFF  GO  /\*\*\*\*\*\* Object: Index [IX\_ArticleId] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  CREATE NONCLUSTERED INDEX [IX\_ArticleId] ON [dbo].[ArticleCollects]  (  [ArticleId] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Index [IX\_UserId] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  CREATE NONCLUSTERED INDEX [IX\_UserId] ON [dbo].[ArticleCollects]  (  [UserId] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Index [IX\_UserId] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  CREATE NONCLUSTERED INDEX [IX\_UserId] ON [dbo].[Articles]  (  [UserId] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Index [IX\_ArticleId] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  CREATE NONCLUSTERED INDEX [IX\_ArticleId] ON [dbo].[ArticleToCategories]  (  [ArticleId] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Index [IX\_BlogCategoryId] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  CREATE NONCLUSTERED INDEX [IX\_BlogCategoryId] ON [dbo].[ArticleToCategories]  (  [BlogCategoryId] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Index [IX\_UserId] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  CREATE NONCLUSTERED INDEX [IX\_UserId] ON [dbo].[BlogCategories]  (  [UserId] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Index [IX\_CommentId] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  CREATE NONCLUSTERED INDEX [IX\_CommentId] ON [dbo].[CommentReports]  (  [CommentId] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Index [IX\_UserId] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  CREATE NONCLUSTERED INDEX [IX\_UserId] ON [dbo].[CommentReports]  (  [UserId] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Index [IX\_ArticleId] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  CREATE NONCLUSTERED INDEX [IX\_ArticleId] ON [dbo].[Comments]  (  [ArticleId] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Index [IX\_UserId] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  CREATE NONCLUSTERED INDEX [IX\_UserId] ON [dbo].[Comments]  (  [UserId] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Index [IX\_FocusUserId] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  CREATE NONCLUSTERED INDEX [IX\_FocusUserId] ON [dbo].[Fans]  (  [FocusUserId] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Index [IX\_UserId] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  CREATE NONCLUSTERED INDEX [IX\_UserId] ON [dbo].[Fans]  (  [UserId] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Index [IX\_CommentParentId] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  CREATE NONCLUSTERED INDEX [IX\_CommentParentId] ON [dbo].[ReplyComments]  (  [CommentParentId] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Index [IX\_ReplierId] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  CREATE NONCLUSTERED INDEX [IX\_ReplierId] ON [dbo].[ReplyComments]  (  [ReplierId] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  GO  /\*\*\*\*\*\* Object: Index [IX\_TargetToReplyId] Script Date: 2020/8/22 16:33:12 \*\*\*\*\*\*/  CREATE NONCLUSTERED INDEX [IX\_TargetToReplyId] ON [dbo].[ReplyComments]  (  [TargetToReplyId] ASC  )WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, DROP\_EXISTING = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]  GO  ALTER TABLE [dbo].[Articles] ADD DEFAULT ((0)) FOR [IsClosingComments]  GO  ALTER TABLE [dbo].[Articles] ADD DEFAULT ((0)) FOR [IsStick]  GO  ALTER TABLE [dbo].[CommentReports] ADD DEFAULT ((0)) FOR [IsHandle]  GO  ALTER TABLE [dbo].[CommentReports] ADD DEFAULT ((0)) FOR [IsConfirm]  GO  ALTER TABLE [dbo].[CommentReports] ADD DEFAULT ((0)) FOR [IsBeReportUserConfirm]  GO  ALTER TABLE [dbo].[Users] ADD DEFAULT ((0)) FOR [IsFreeze]  GO  ALTER TABLE [dbo].[ArticleCollects] WITH CHECK ADD CONSTRAINT [FK\_dbo.ArticleCollects\_dbo.Articles\_ArticleId] FOREIGN KEY([ArticleId])  REFERENCES [dbo].[Articles] ([Id])  GO  ALTER TABLE [dbo].[ArticleCollects] CHECK CONSTRAINT [FK\_dbo.ArticleCollects\_dbo.Articles\_ArticleId]  GO  ALTER TABLE [dbo].[ArticleCollects] WITH CHECK ADD CONSTRAINT [FK\_dbo.ArticleCollects\_dbo.Users\_UserId] FOREIGN KEY([UserId])  REFERENCES [dbo].[Users] ([Id])  GO  ALTER TABLE [dbo].[ArticleCollects] CHECK CONSTRAINT [FK\_dbo.ArticleCollects\_dbo.Users\_UserId]  GO  ALTER TABLE [dbo].[Articles] WITH CHECK ADD CONSTRAINT [FK\_dbo.Articles\_dbo.Users\_UserId] FOREIGN KEY([UserId])  REFERENCES [dbo].[Users] ([Id])  GO  ALTER TABLE [dbo].[Articles] CHECK CONSTRAINT [FK\_dbo.Articles\_dbo.Users\_UserId]  GO  ALTER TABLE [dbo].[ArticleToCategories] WITH CHECK ADD CONSTRAINT [FK\_dbo.ArticleToCategories\_dbo.Articles\_ArticleId] FOREIGN KEY([ArticleId])  REFERENCES [dbo].[Articles] ([Id])  GO  ALTER TABLE [dbo].[ArticleToCategories] CHECK CONSTRAINT [FK\_dbo.ArticleToCategories\_dbo.Articles\_ArticleId]  GO  ALTER TABLE [dbo].[ArticleToCategories] WITH CHECK ADD CONSTRAINT [FK\_dbo.ArticleToCategories\_dbo.BlogCategories\_BlogCategoryId] FOREIGN KEY([BlogCategoryId])  REFERENCES [dbo].[BlogCategories] ([Id])  GO  ALTER TABLE [dbo].[ArticleToCategories] CHECK CONSTRAINT [FK\_dbo.ArticleToCategories\_dbo.BlogCategories\_BlogCategoryId]  GO  ALTER TABLE [dbo].[BlogCategories] WITH CHECK ADD CONSTRAINT [FK\_dbo.BlogCategories\_dbo.Users\_UserId] FOREIGN KEY([UserId])  REFERENCES [dbo].[Users] ([Id])  GO  ALTER TABLE [dbo].[BlogCategories] CHECK CONSTRAINT [FK\_dbo.BlogCategories\_dbo.Users\_UserId]  GO  ALTER TABLE [dbo].[CommentReports] WITH CHECK ADD CONSTRAINT [FK\_dbo.CommentReports\_dbo.Comments\_CommentId] FOREIGN KEY([CommentId])  REFERENCES [dbo].[Comments] ([Id])  GO  ALTER TABLE [dbo].[CommentReports] CHECK CONSTRAINT [FK\_dbo.CommentReports\_dbo.Comments\_CommentId]  GO  ALTER TABLE [dbo].[CommentReports] WITH CHECK ADD CONSTRAINT [FK\_dbo.CommentReports\_dbo.Users\_UserId] FOREIGN KEY([UserId])  REFERENCES [dbo].[Users] ([Id])  GO  ALTER TABLE [dbo].[CommentReports] CHECK CONSTRAINT [FK\_dbo.CommentReports\_dbo.Users\_UserId]  GO  ALTER TABLE [dbo].[Comments] WITH CHECK ADD CONSTRAINT [FK\_dbo.Comments\_dbo.Articles\_ArticleId] FOREIGN KEY([ArticleId])  REFERENCES [dbo].[Articles] ([Id])  GO  ALTER TABLE [dbo].[Comments] CHECK CONSTRAINT [FK\_dbo.Comments\_dbo.Articles\_ArticleId]  GO  ALTER TABLE [dbo].[Comments] WITH CHECK ADD CONSTRAINT [FK\_dbo.Comments\_dbo.Users\_UserId] FOREIGN KEY([UserId])  REFERENCES [dbo].[Users] ([Id])  GO  ALTER TABLE [dbo].[Comments] CHECK CONSTRAINT [FK\_dbo.Comments\_dbo.Users\_UserId]  GO  ALTER TABLE [dbo].[Fans] WITH CHECK ADD CONSTRAINT [FK\_dbo.Fans\_dbo.Users\_FocusUserId] FOREIGN KEY([FocusUserId])  REFERENCES [dbo].[Users] ([Id])  GO  ALTER TABLE [dbo].[Fans] CHECK CONSTRAINT [FK\_dbo.Fans\_dbo.Users\_FocusUserId]  GO  ALTER TABLE [dbo].[Fans] WITH CHECK ADD CONSTRAINT [FK\_dbo.Fans\_dbo.Users\_UserId] FOREIGN KEY([UserId])  REFERENCES [dbo].[Users] ([Id])  GO  ALTER TABLE [dbo].[Fans] CHECK CONSTRAINT [FK\_dbo.Fans\_dbo.Users\_UserId]  GO  ALTER TABLE [dbo].[ReplyComments] WITH CHECK ADD CONSTRAINT [FK\_dbo.ReplyComments\_dbo.Comments\_CommentParentId] FOREIGN KEY([CommentParentId])  REFERENCES [dbo].[Comments] ([Id])  GO  ALTER TABLE [dbo].[ReplyComments] CHECK CONSTRAINT [FK\_dbo.ReplyComments\_dbo.Comments\_CommentParentId]  GO  ALTER TABLE [dbo].[ReplyComments] WITH CHECK ADD CONSTRAINT [FK\_dbo.ReplyComments\_dbo.Users\_ReplierId] FOREIGN KEY([ReplierId])  REFERENCES [dbo].[Users] ([Id])  GO  ALTER TABLE [dbo].[ReplyComments] CHECK CONSTRAINT [FK\_dbo.ReplyComments\_dbo.Users\_ReplierId]  GO  ALTER TABLE [dbo].[ReplyComments] WITH CHECK ADD CONSTRAINT [FK\_dbo.ReplyComments\_dbo.Users\_TargetToReplyId] FOREIGN KEY([TargetToReplyId])  REFERENCES [dbo].[Users] ([Id])  GO  ALTER TABLE [dbo].[ReplyComments] CHECK CONSTRAINT [FK\_dbo.ReplyComments\_dbo.Users\_TargetToReplyId]  GO  USE [master]  GO  ALTER DATABASE [BlogSystem] SET READ\_WRITE  GO |

## 系统架构设计

MVC模型学智博客网系统使用MVC设计模式进行设计，有利于把模型数据与视图层实现有效分离，设置较为灵活的用户页面，提高用户体验效果。

MVC设计模式包括业务模型、用户界面、控制器三层，MVC是一种使用MVC(Model View Controller模型-视图-控制器)设计创建Web应用程序的模式:Model(模型)表示应用程序核心(比如数据库记录列表)。View(视图)显示数据(数据库记录)。Controller(控制器)处理输入(写入数据库记录)。MVC模式同时提供了对HTML、CSS和JavaScript的完全控制。Model(模型)是应用程序中用于处理应用程序数据逻辑的部分。通常模型对象负责在数据库中存取数据。View(视图)是应用程序中处理数据显示的部分。通常视图是依据模型数据创建的。Controller(控制器)是应用程序中处理用户交互的部分。通常控制器负责从视图读取数据，控制用户输入，并向模型发送数据。MVC分层有助于管理复杂的应用程序，因为您可以在一个时间内专门关注一个方面。例如，您可以在不依赖业务逻辑的情况下专注于视图设计。同时也让应用程序的测试更加容易。MVC分层同时也简化了分组开发。不同的开发人员可同时开发视图、控制器逻辑和业务逻辑。MVC模型结构如图3-4所示:

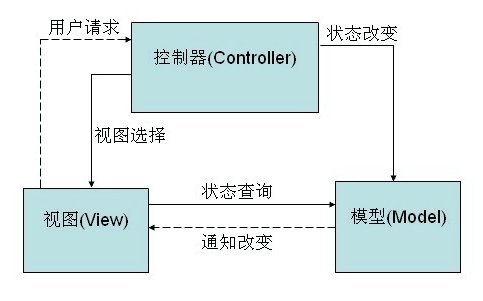


图 3-4 学智博客网架构设计图

## 模块设计

### 博客文章信息类图

博客文章信息的类图如下图3-5所示。各类具体说明如下：

(1) BaseEntity类：位于该项目目录Models/BaseEntity处。包括主键id、是否删除（IsRemoved：伪删除）、创建时间（CreateTime）。

(2) Article类：继承BaseEntity类，位于该项目目录Models/Article处。属性包括文章标题（Title）、文章内容 (Content)、发布状态(State)、关闭评论（IsClosingComment）、发布人（User）、推荐数（GoodCount）、反对数（BadCount）、发布人编号（UserId）。

(3) ArticleController控制器：ArticleController控制器完成对博客文章的增删查改，并将用户新发布的文章显示在首页，位于该项目目录BlogSystem.MvcUI/Controller处。方法包括文章的查询（InitData()）、文章发布（CreateArticle()）、文章修改（EditArticle()）、文章移除(RemoveArticle())等方法。

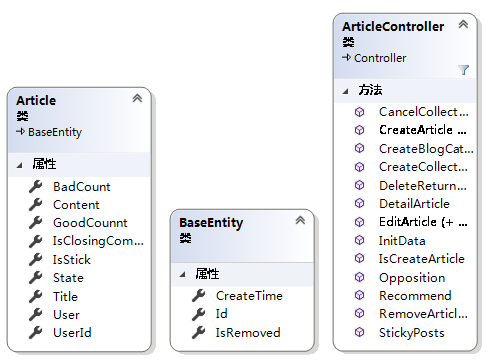


图 3-5 博客文章信息类图

### 博客文章信息操作流程

博客文章模块主要完成对博客文章基本信息的增删查改功能,其操作流程如图3-6博客文章时序图所示:

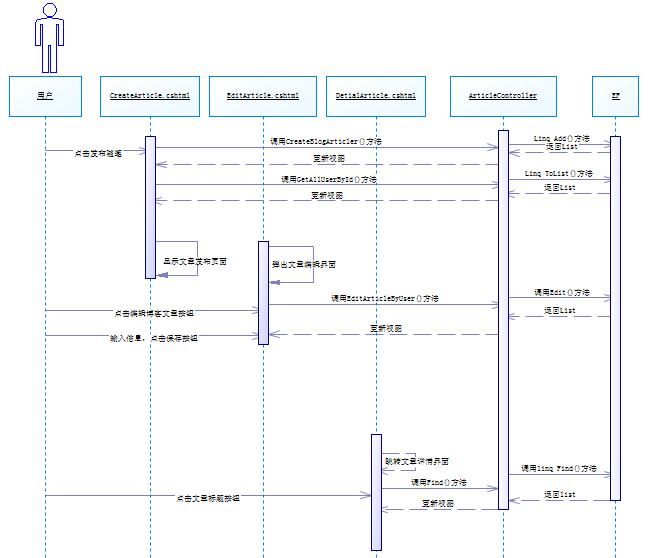


图 3-6 博客文章信息时序图

### 用户信息类图

用户信息的类图如下图3-7所示。各类具体说明如下：

(1) UserController类：UserController类完成对用户的增删查改操作，位于该项目目录BlogSystem.MvcUI/Controllers/UserController处。方法包括用户主页查询方法（UserHome ()）、用户注册方法（Register()）、用户基本修改方法(BasicSetting ())、忘记密码（ForgetPwd()）、用户登录（Login()）、ajax检测是否存在重复注册用户（AjaxCheckRegister()）、用户修改密码（SetPassword()）。

(2) User类：User类用户基本信息的类，位于该项目目录Models/User处。属性包括用户邮箱（Eamil）、用户昵称(NickName)、用户头像 (ImagePath)、用户密码（Password）、是否被冻结（IsFreeze）、粉丝数（FansCount）、关注数（FocusCount）、个人说明（PersonalDescrtion）。

(3) BaseEntity类：位于该项目目录Models/BaseEntity处。包括主键id、是否删除（IsRemoved：伪删除）、创建时间（CreateTime）。

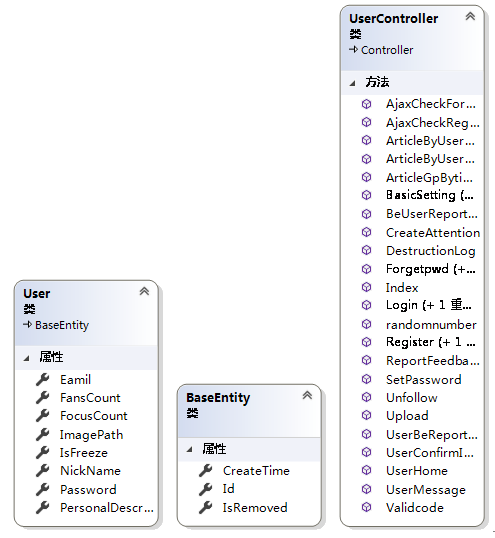


图 3-7 用户信息类图

### 用户信息操作流程

用户信息模块主要完成对用户信息的增删查改功能,其操作流程如图3-8用户信息时序图所示:

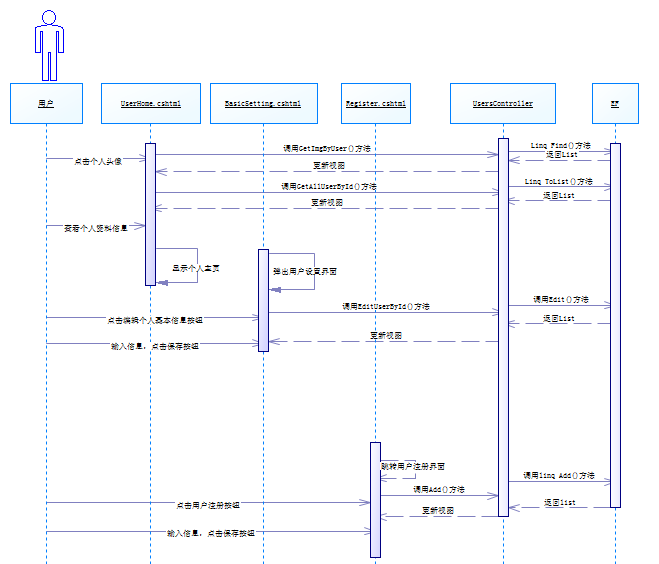


图 3-8 用户信息时序图

### 评论信息类图

评论信息的类图如下图3-7所示。各类具体说明如下：

(1) CommentController类：BackstageController类完成对评论的增删查改操作，位于该项目目录MvcApplication1/Controllers/CommentController处。方法包括评论查询方法（GetReplyComment ()）、评论添加方法（CreateComment()）、评论修改方法(EditComment ())、评论删除方法(DeleteReturnComment())。

(2) Comment类：Comment类评论基本信息的类，位于该项目目录Models/Comment处。属性包括评论内容（Content）、评论编号(ID)、文章编号 (ArticleId)、用户编号(UserId)。

(3) BaseEntity类：位于该项目目录Models/BaseEntity处。包括主键id、是否删除（IsRemoved：伪删除）、创建时间（CreateTime）。

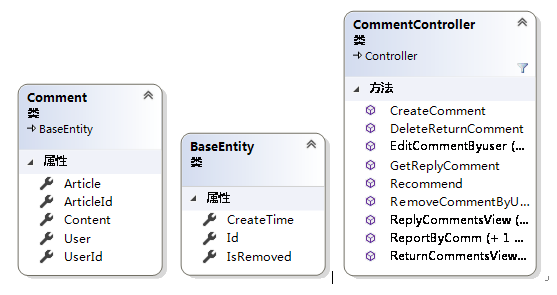


图 3-9 评论信息类图

### 评论信息操作流程

评论信息模块主要完成对评论基本信息的增删查改功能,其操作流程如图3-8评论信息时序图所示:

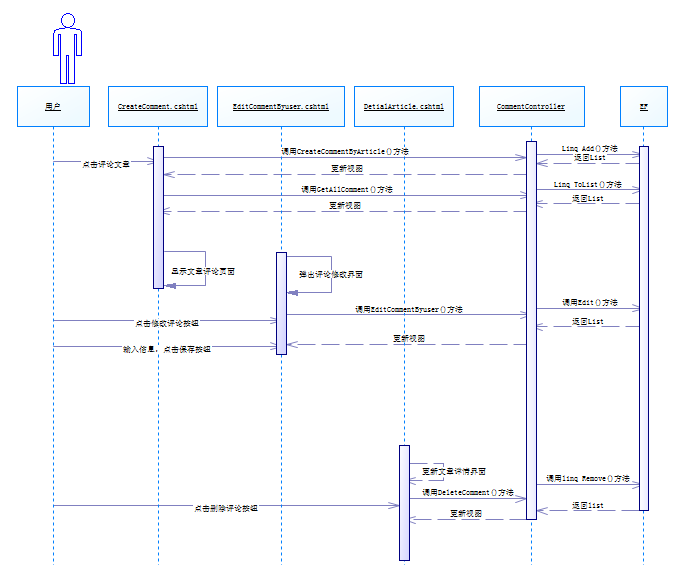


图 3-10 评论信息时序图

### 举报记录类图

举报记录的类图如下图3-7所示。各类具体说明如下：

(1) ReportController类：ReportController类完成对评论举报记录的增删查改操作，位于该项目目录BlogSystem.MvcUI/Controllers/ReportController处。方法包括初始化数据方法（InitData ()）、添加举报记录方法（ReportByComm()）、反馈举报信息方法(ReturnCommentsView ())、删除举报记录方法（RemoveCommentByUser()）。

(2) CommentReport类：CommentReport类举报记录的类，位于该项目目录Models/CommentReport处。属性包括举报原因（Content）、管理员是否审核(IsHandle)、举报人(IsBeReportUser)、用户是否确认（IsConfirm）、被举报人（User）、被举报人编号（UserId）、举报评论（Comment）、被举报评论编号（CommentId）。

(3) BaseEntity类：位于该项目目录Models/BaseEntity处。包括主键id、是否删除（IsRemoved：伪删除）、创建时间（CreateTime）。

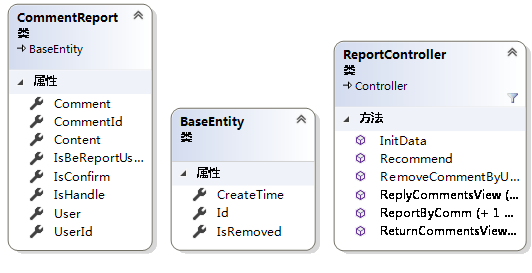


图 3-11 举报记录类图

### 举报记录操作流程

举报记录模块主要完成对评论举报记录基本信息的增删查改功能,其操作流程如图3-8举报记录时序图所示:

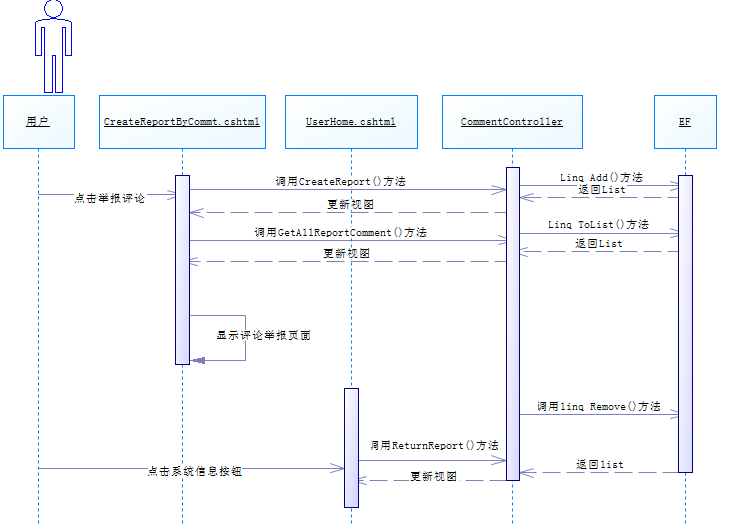


图 3-12 举报记录时序图

# 产品实现

## 博客文章模块实现

博客文章模块包括首页博客信息展示、新增博客、修改博客内容。

### 界面设计

博客文章界面如图4-1所示,点击博客标题链接跳转到博客详情页面，文章信息修改界面如图4-3所示，点击删除按钮删除对应的博客文章，文章添加界面如图4-2所示，添加成功后弹出添加成功弹窗：



图 4-1 博客文章列表界面图



图 4-2 博客详情界面图

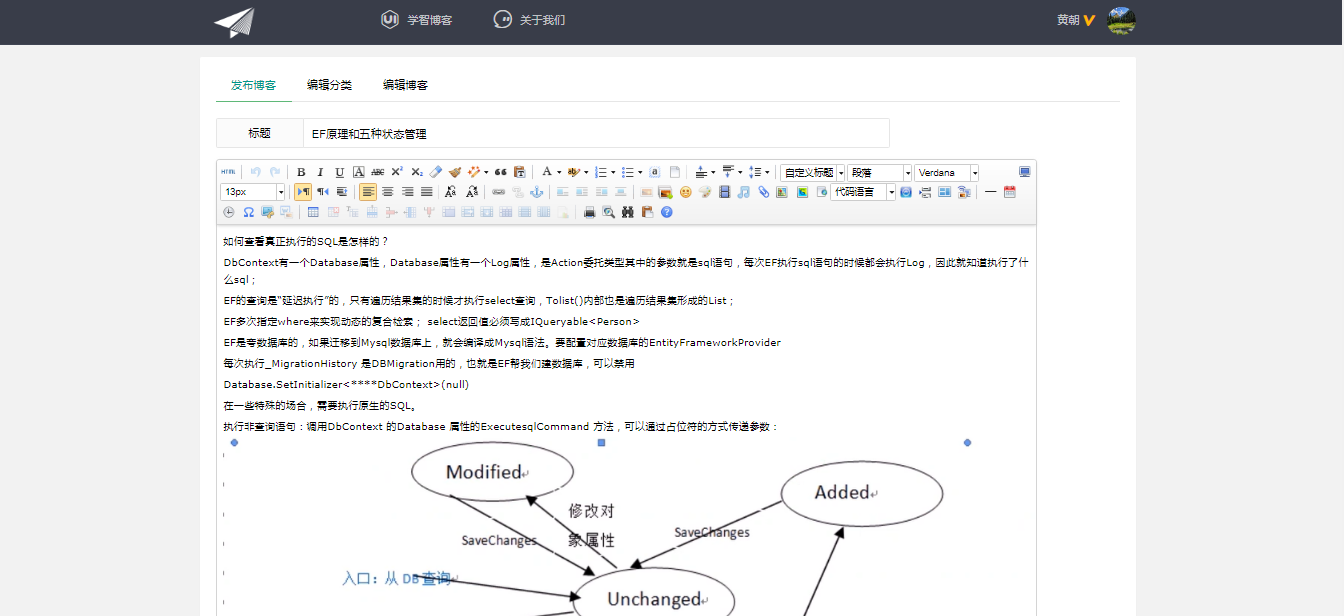


图 4-3 博客文章修改界面图

### 功能实现

博客文章模块增删查改的主要代码







图 4-4 博客文章主要代码图

## 用户信息模块实现

用户信息模块包括用户主页、用户基本设置、修改密码。

### 界面设计

用户主页界面如图4-5所示,点击用户头像跳转到我的主页页面，用户基本设置界面如图4-6所示，用户修改密码界面如图4-7所示，修改成功后跳转重新登录：



图 4-5 用户主页界面图

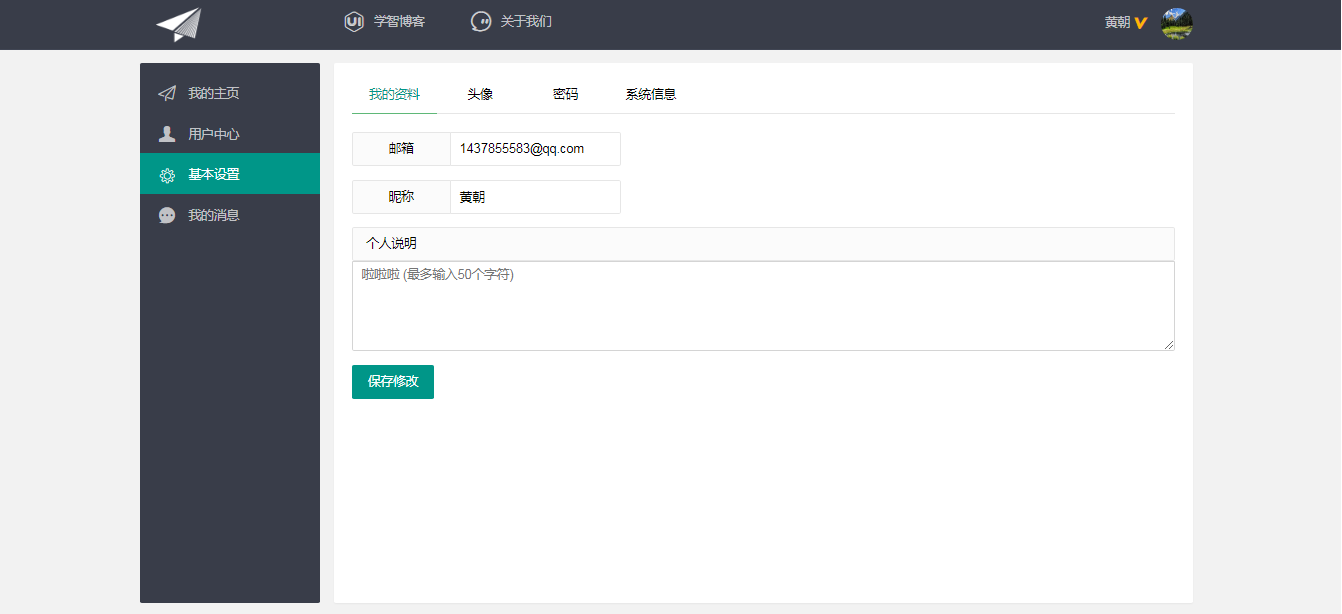


图 4-6 用户基本设置界面图

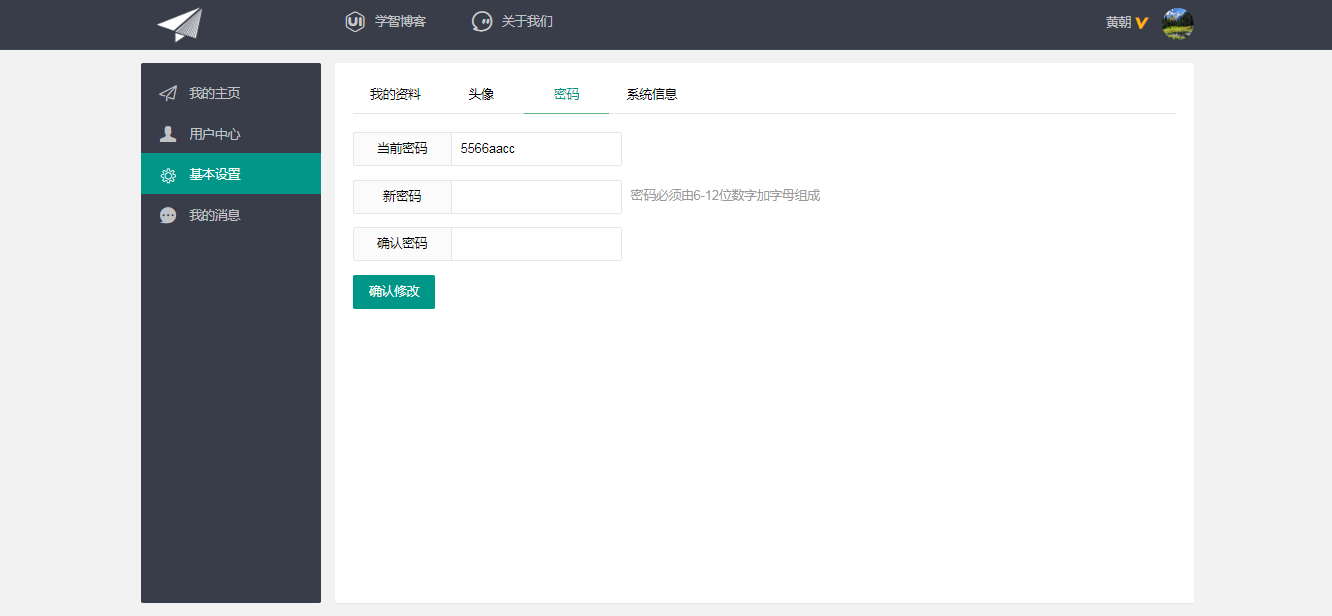
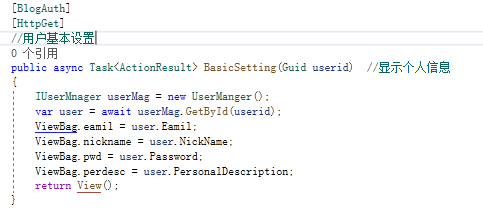


图 4-7 修改用户密码界面图

### 功能实现

用户主页模块增删查改的主要代码





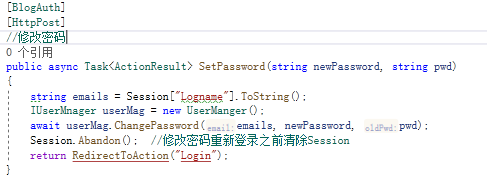


图 4-8 用户信息模块主要代码图

## 评论信息模块实现

评论信息模块评论列表、发布评论、修改评论。

### 界面设计

评论列表界面如图4-9所示,点击评论发布弹出评论界面如图4-10所示，用用户修改评论弹出模态框如图4-11所示：

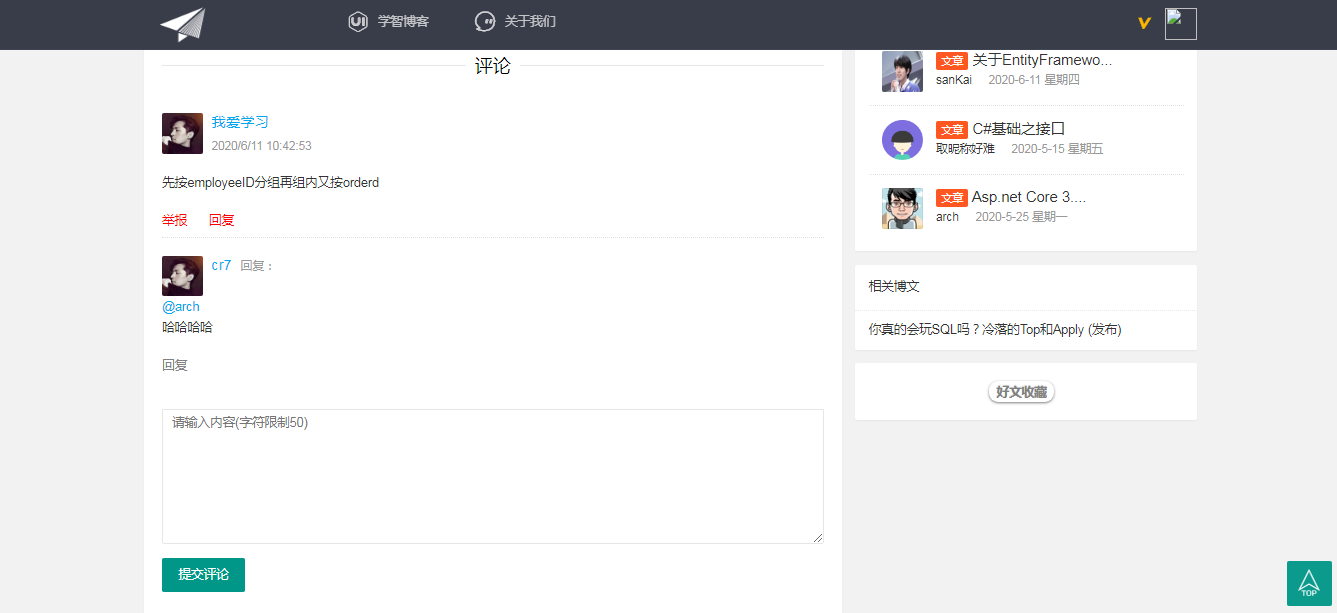


图 4-9 评论列表界面图

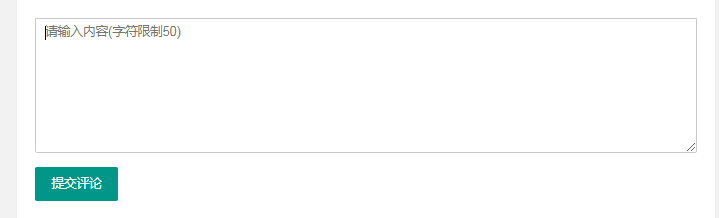


图 4-10 发布评论界面图

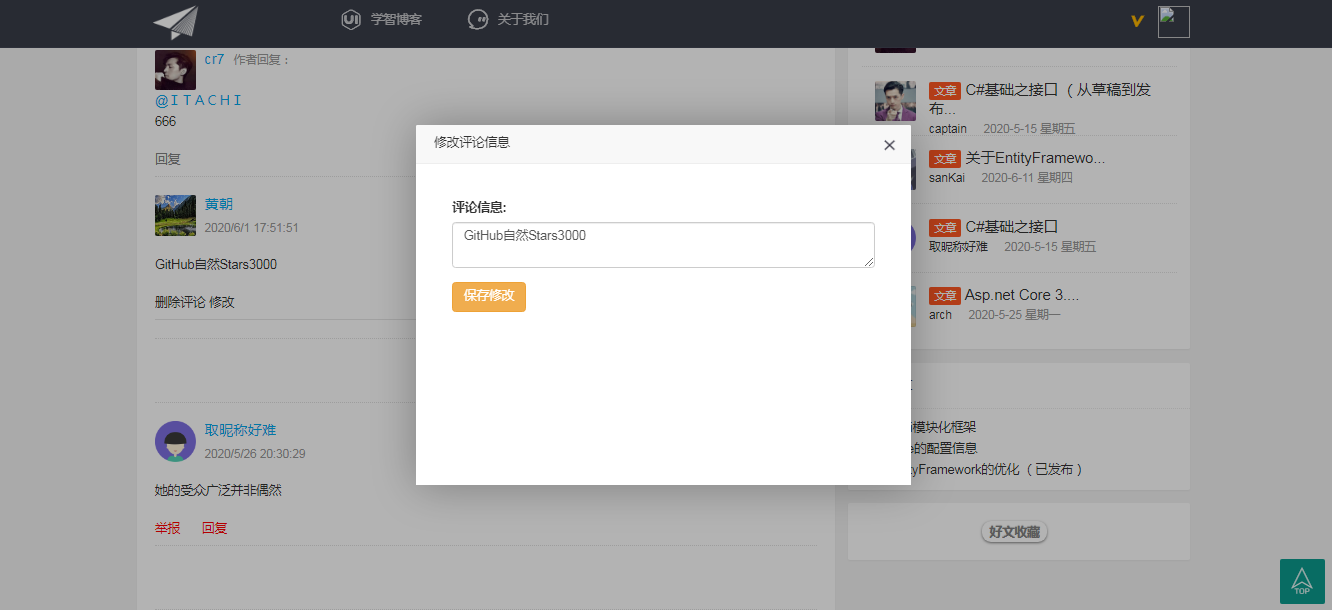


图 4-11 修改评论界面图

### 功能实现

评论信息模块增删查改的主要代码



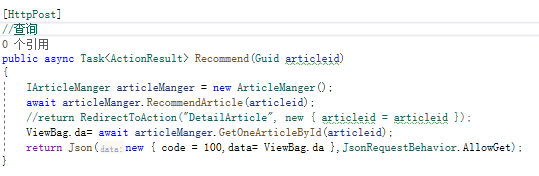




图 4-12 评论信息模块主要代码图

## 举报记录模块实现

举报记录模块包括举报评论、举报原因。

### 界面设计

评论举报界面如图4-5所示,点击举报连接弹出举报原因模态框，举报原因模态框界面如图4-6所示：

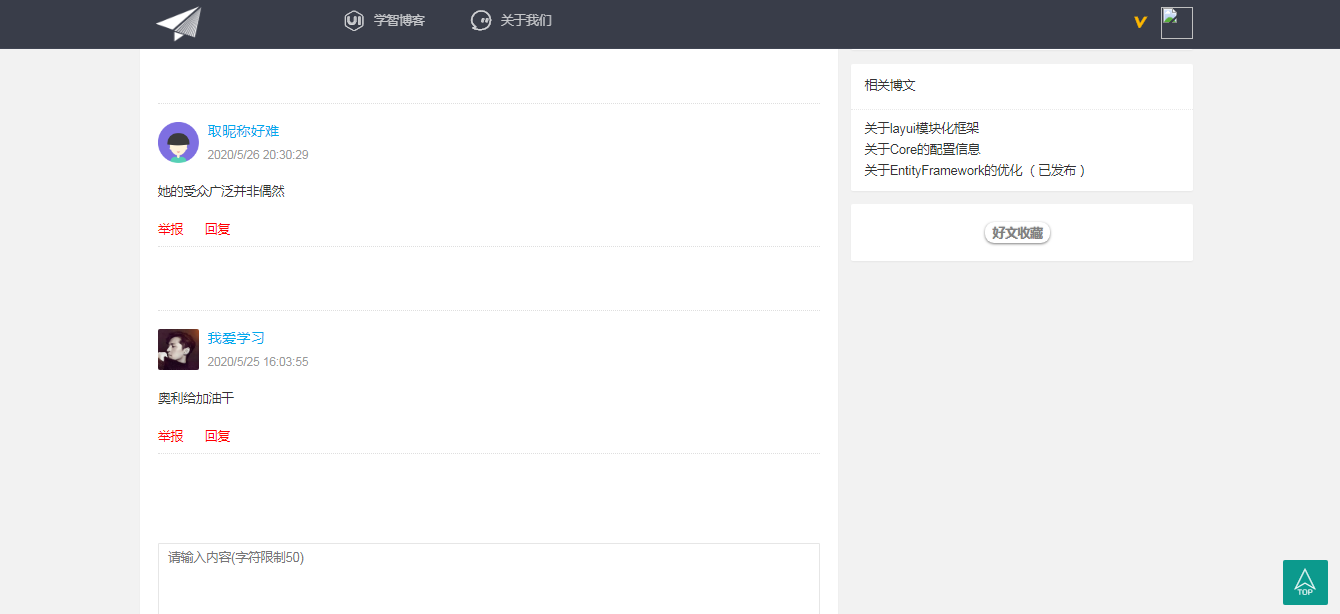


图 4-13 评论举报界面图

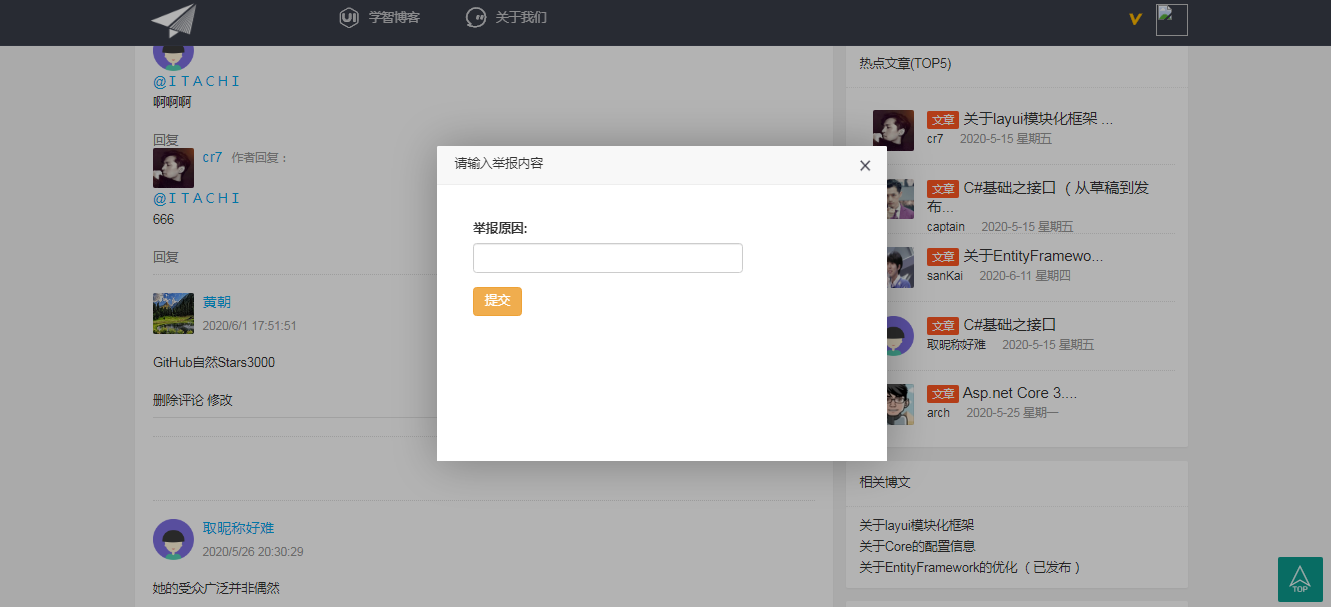


图 4-14 举报原因界面图

### 功能实现

举报模块的主要代码



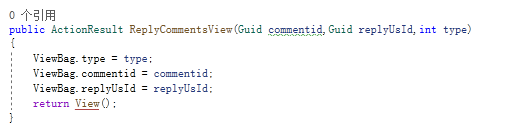


图 4-15 举报记录模块主要代码图

# 产品测试

## 测试方法与策略

本项目在所有功能编码完成且系统能完整运行的状态下进行系统测试，以确认各项需求的功能是否能正常运行，并给出测试结果。

本系统采用人工测试的方式进行黑盒测试对系统功能进行验证，使用到的测试用例设计方法包括：边界值分析、等价类划分、错误猜测法、因果图法等。

本项目的测试策略包括：

1. 功能测试

功能测试验证本系统能按照设计要求实现模块的各个功能，数据应完整、界面美观、操作方便。

1. 数据项测试

* 输入字母和数字等数据项是否能够正确显示，并添加到系统中；
* 鼠标点击、图形模式的数据项（如滑动条）是否正常工作；
* 是否能够识别非法数据；
* 提示消息是否可理解。

1. 业务测试

功能测试完成后进行业务测试，业务测试关注的要点是业务流程，及数据流从软件中的一个模块流到另一个模块的过程中的正确性。

## 博客文章信息模块测试

博客文章信息模块的测试主要验证用户发布文章标题是否为空，实现博客文章信息的增删改查功能，并对用户输入的数据进行合法性验证，对输入的非法数据给出提示，并验证提示信息的可理解性。该模块的测试用例及测试执行结果如下表5-1所示：

表 5-1 博客文章信息模块测试用例表

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **用例编号** | **测试项目** | **测试标题** | **重要级别** | **预置条件** | **输入** | **执行步骤** | **预期输出** | **测试结果** |
| 1 | 添加博客文章信息 | 【发布随笔】按钮功能测试 | 高 | 用户登录成功，打开博客首页页面 | 无 | 点击【发布随笔】按钮 | 弹出“添加博客文章”页面 | 通过 |
| 2 | 添加博客文章信息 | 输入全部正确的新增数据，进行【发布】按钮功能测试 | 高 | 用户登录成功，打开“添加博客文章”窗口 | 1. 文章标题：C#基础教程系列2、文章内容：C#是一种面向对象的编程语言，语法简单，容易上手学习 | 用户输入标题，用户输入文章内容，点击【发布】按钮 | 保存草稿的文章将会出现在个人主页的草稿箱，发布的文章将会出现在博客首页列表中 | 通过 |
| 3 | 添加博客文章信息 | 不输入文章标题（非空验证）数据，进行保存 | 高 | 用户登录成功，打开“添加博客文章”窗口 | 1、文章标题：C#基础教程系列 | 输入以上数据，点击【发布】按钮 | 提示：请输入文章标题 | 通过 |
| 4 | 添加博客文章信息 | 不输入文章内容（非空验证）数据，进行保存 | 高 | 用户登录成功，打开“添加博客文章”窗口 | 1、文章内容：C#是一种面向对象的编程语言，语法简单，容易上手学习 | 输入以上数据，点击【发布】按钮 | 提示：请输入文章内容 | 通过 |
| 5 | 修改博客文章信息 | 【编辑博客】按钮功能测试 | 高 | 用户登录成功，打开修改博客页面 | 无 | 点击【保存修改】按钮 | 弹出“修改文章”页面，显示带入的博客文章信息 | 通过 |
| 6 | 修改博客文章信息 | “修改文章信息”窗口文字显示及UI排版正确性验证 | 中 | 用户登录成功，打开修改博客页面 | 无 | 查看页面显示 | 页面文字显示及UI排版正确 | 通过 |
| 7 | 修改博客文章信息 | 修改博客文章信息记录正确性验证 | 高 | 用户登录成功，打开修改博客页面 | 无 | 查看博客文章信息列表数据显示 | 首页文章内容随之更新 | 通过 |
| 8 | 修改博客文章信息 | 输入全部正确的修改数据，进行“取消”按钮功能测试 | 高 | 用户登录成功，打开修改博客页面 | 1、文章标题：修改C#基础教程系列2、文章内容：C#是一种面向对象的编程语言，语法简单，容易上手学习 | 输入以上数据，点击“取消”按钮 | 不保存当前编辑内容，关闭当前窗口，回到首页，列表页相应内容前后不变 | 通过 |
| 9 | 删除博客文章信息 | 【删除】按钮功能测试 | 高 | 用户登录成功，打开修改博客编辑页面 | 无 | 点击【删除】按钮 | 提示：确认删除该文章吗？ | 通过 |
| 10 | 删除博客文章信息 | 删除弹出框【确认】按钮测试 | 高 | 用户登录成功，点击【删除】按钮，弹出删除提示框 | 无 | 点击【确认】按钮 | 删除成功 | 通过 |
| 11 | 删除博客文章信息 | 删除弹出框【取消】按钮测试 | 高 | 用户登录成功，点击【删除】按钮，弹出删除提示框 | 无 | 点击【取消】按钮 | 取消删除 | 通过 |

## 5.3 用户信息模块测试

用户信息模块的测试主要验证管理员是否能正确的实现用户信息的增改查功能，并对管理员输入的数据进行合法性验证，对输入的非法数据给出提示，并验证提示信息的可理解性。该模块的测试用例及测试执行结果如下表5-2所示：

表 5-2 用户信息测试用例表

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **用例编号** | **测试项目** | **测试标题** | **重要级别** | **预置条件** | **输入** | **执行步骤** | **预期输出** | **测试结果** |
| 1 | 注册用户信息 | 【注册】按钮功能测试 | 高 | 无 | 1、用户邮箱：xx.com  2、用户昵称：游客  3、用户验证邮箱：xx | 点击【注册】按钮 | 弹出“用户登录”页面 | 通过 |
| 2 | 注册用户信息 | 输入全部正确的新增数据，进行【注册】按钮功能测试 | 高 | 无 | 1、用户邮箱：xx.com  2、用户昵称：游客  3、用户验证邮箱：xx | 输入以上数据，点击【取消】按钮 | 不保存当前新增内容，关闭当前窗口，回到登录页 | 通过 |
| 3 | 注册用户信息 | 不输入所属邮箱（非空验证）数据，进行保存 | 高 | 无 | 1、用户邮箱：xx.com  2、用户昵称：游客 | 输入以上数据，点击【注册】按钮 | 提示：请输入用户邮箱 | 通过 |
| 4 | 注册用户信息 | 不输入用户昵称（非空验证）数据，进行保存 | 高 | 无 | 1、用户邮箱：xx.com  2、用户昵称：游客 | 输入以上数据，点击【注册】按钮 | 提示：请输入用户昵称 | 通过 |
| 5 | 修改用户基本信息 | 【编辑】按钮功能测试 | 高 | 用户登录成功，打开用户编辑页面 | 无 | 点击【修改】按钮 | 弹出“修改用户”，显示带入的用户信息 | 通过 |
| 6 | 修改用户基本信息 | “修改用户信息”窗口文字显示及UI排版正确性验证 | 中 | 用户登录成功，打开“修改用户信息”窗口 | 无 | 查看页面显示 | 页面文字显示及UI排版正确 | 通过 |
| 7 | 修改用户基本信息 | 输入全部正确的修改数据，进行【保存】按钮功能测试 | 高 | 用户登录成功，打开“修改用户信息”窗口 | 1、用户邮箱：xx.com  2、用户昵称：游客  3、个人说明：我的信息 | 输入以上数据，点击【保存】按钮 | 保存当前修改内容，关闭当前窗口，回到个人主页 | 通过 |
| 8 | 修改用户基本信息 | 输入全部正确的修改数据，进行【取消】按钮功能测试 | 高 | 用户登录成功，打开“修改用户信息”窗口 | 1、用户邮箱：xx.com  2、用户昵称：游客  3、个人说明：我的信息 | 输入以上数据，点击【取消】按钮 | 不保存当前编辑内容，关闭当前窗口，回到个人主页，个人主页信息相应内容前后不变 | 通过 |

# 

## 5.4 评论信息模块测试

评论信息模块的测试主要验证用户是否能正确的实现文章评论的增删改查功能，并对评论内容的数据进行合法性验证，对输入的非法数据给出提示，并验证提示信息的可理解性。该模块的测试用例及测试执行结果如下表5-3所示：

表 5-3 用户信息测试用例表

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **用例编号** | **测试项目** | **测试标题** | **重要级别** | **预置条件** | **输入** | **执行步骤** | **预期输出** | **测试结果** |
| 1 | 用户发布评论信息 | 【发布】按钮功能测试 | 高 | 用户登录成功，打开博客文章详情页面 | 1、评论内容：评论测试内容 | 点击【发布】按钮 | 评论内容显示在对应博客文章下方 | 通过 |
| 2 | 用户发布评论信息 | 输入全部正确的新增数据，进行【发布】按钮功能测试 | 高 | 用户登录成功，打开博客文章详情页面 | 1、评论内容：评论测试内容 | 输入以上数据，点击【取消】按钮 | 不保存当前新增内容，关闭当前窗口，回到博客文章详情页 | 通过 |
| 3 | 用户发布评论信息 | 不输入所属评论内容（非空验证）数据，进行保存 | 高 | 用户登录成功，打开博客文章详情页面 | 1、用户邮箱：xx.com  2、用户昵称：游客 | 输入以上数据，点击【注册】按钮 | 提示：请输入用户邮箱 | 通过 |
| 5 | 修改用户评论信息 | 【修改】按钮功能测试 | 高 | 用户登录成功，打开博客文章详情页面 | 1、评论内容：修改评论测试内容 | 点击【修改】按钮 | 弹出“修改评论信息”，显示带入的评论信息 | 通过 |
| 6 | 修改用户评论信息 | “修改评论信息”窗口文字显示及UI排版正确性验证 | 中 | 用户登录成功，打开博客文章详情页面 | 无 | 查看页面显示 | 页面文字显示及UI排版正确 | 通过 |
| 7 | 修改用户评论信息 | 输入全部正确的修改数据，进行【保存】按钮功能测试 | 高 | 用户登录成功，打开博客文章详情页面 | 1、评论内容：修改评论测试内容 | 输入以上数据，点击【保存】按钮 | 保存当前修改内容，关闭当前窗口，回到博客文章详情页 | 通过 |
| 8 | 修改用户评论信息 | 输入全部正确的修改数据，进行【取消】按钮功能测试 | 高 | 用户登录成功，打开博客文章详情页面 | 1、评论内容：修改评论测试内容 | 输入以上数据，点击【取消】按钮 | 不保存当前编辑内容，关闭当前窗口，回到博客文章详情页，对应评论信息相应内容前后不变 | 通过 |
| 9 | 删除用户评论信息 | 【删除】按钮功能测试 | 高 | 用户登录成功，打开博客文章详情页面 | 无 | 点击【删除】按钮 | 提示：确认删除该评论吗？ | 通过 |
| 10 | 删除用户评论信息 | 删除弹出框【确认】按钮测试 | 高 | 用户登录成功，点击【删除】按钮，弹出删除提示框 | 无 | 点击【确认】按钮 | 删除成功 | 通过 |
| 11 | 删除用户评论信息 | 删除弹出框【取消】按钮测试 | 高 | 用户登录成功，点击【删除】按钮，弹出删除提示框 | 无 | 点击【取消】按钮 | 取消删除 | 通过 |

## 5.5 举报记录模块测试

举报记录模块的测试主要验证用户是否能正确的实现对某一评论进行举报的增改查功能，并对举报的评论内容后台进行审核，对输入的非法数据给出提示，并验证提示信息的可理解性。该模块的测试用例及测试执行结果如下表5-4所示：

表 5-4 举报记录测试用例表

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **用例编号** | **测试项目** | **测试标题** | **重要级别** | **预置条件** | **输入** | **执行步骤** | **预期输出** | **测试结果** |
| 1 | 用户举报评论信息 | 【举报】按钮功能测试 | 高 | 用户登录成功，打开博客文章详情页面 | 1、举报原因：评论违规处理 | 点击【确定举报】按钮 | 弹出“举报成功，后台将在24小时之进行处理” | 通过 |
| 2 | 用户举报评论信息 | 输入全部正确的举报原因数据，进行【举报】按钮功能测试 | 高 | 用户登录成功，打开博客文章详情页面 | 1、举报原因：评论违规处理 | 输入以上数据，点击【取消】按钮 | 不保存当前举报原因，关闭当前窗口，回到博客详情页面 | 通过 |
| 3 | 用户举报评论信息 | 不输入所属举报原因（非空验证）数据，进行保存 | 高 | 用户登录成功，打开博客文章详情页面 | 无 | 输入以上数据，点击【确定举报】按钮 | 提示：请输入举报原因 | 通过 |
| 5 | 修改举报评论信息 | 【修改】按钮功能测试 | 高 | 用户登录成功，打开博客文章详情页面 | 无 | 点击【修改】按钮 | 弹出“修改举报原因”，显示带入的用户信息 | 通过 |
| 6 | 修改举报评论信息 | “修改举报原因”窗口文字显示及UI排版正确性验证 | 中 | 用户登录成功，打开博客文章详情页面 | 无 | 查看页面显示 | 页面文字显示及UI排版正确 | 通过 |
| 7 | 修改举报评论信息 | 输入全部正确的修改数据，进行【确定修改】按钮功能测试 | 高 | 用户登录成功，打开博客文章详情页面 | 1、举报原因：测试修改举报内容 | 输入以上数据，点击【保存】按钮 | 保存当前修改内容，关闭当前窗口，回到博客详情页面 | 通过 |
| 8 | 修改举报评论信息 | 输入全部正确的修改数据，进行【确定修改】按钮功能测试 | 高 | 用户登录成功，打开博客文章详情页面 | 1、举报原因：测试修改举报内容 | 输入以上数据，点击【保存】按钮 | 不保存当前修改内容，关闭当前窗口，回到博客详情页面 | 通过 |

# 结论

## 毕业设计成果特点

学智博客网站管理员拥有用户管理信息、管理员信息的增删查改操作。通过此管理系统的开发，将软件技术专业授课阶段学习的前沿理论知识C#、MVC、面向对象、数据库等与实践联系，融会贯通，不但提高了理论水平，还提高了项目开发的能力。项目采用ASP.NET MVC进行以及结合C#仓储设计模式配合异步编程、ef分层搭建，Web前端采用Html+CSS、Ajax、Layui等前沿的技术，结合Javascript和jQuery实现动态特效。Web后端采用MVC+EF分层框架、异步编程、C#仓储设计模式实现业务逻辑的处理和数据库访问。

## 设计成果的实用价值或应用前景

计算机科学技术引发了全世界科技时代的大更新，知识更新速度也很快，快餐文化加速了人们对文化认知的新高度，如何在科技文化潮流背景下取之精华，弃之糟粕也是对人们思想高度和综合素质的一大考验。学智博客在此背景下以服务为核心，为广大技术人员提供最大最基础最高效“三最”知识分享网站,使得IT行业的高手能够在此大展身手，分享技术、分享生活。助力文化技术交流，过滤淤泥文化，逐步发展成为最大的IT技术交流平台，构建完整的基础体系IT行业高质量向前发展。

## 不足之处或遗留未予解决的问题

学智博客网站-博客文章管理信息、用户信息管理、评论信息管理模块、举报记录管理模块的设计与实现是基于B/S管理模式的结构平台和面向对象的设计思想，并利用一些与MVC相关的技术以及异步编程，对于学智博客网站的整体功能要求已全部完整的实现。尽管如此，本系统在面向众多用户的使用过程中，仍遇到了某些潜在的问题，但大体上还是完成了学智博客网站的功能需求。  
本系统的不足之处：  
1.在多用户同时登陆时可能会造成网站崩溃、没有处理数据高并发  
2.暂未实现用户单点登陆操作   
3.数据库的恢复和备份没有完全实现自动化。   
 以上这些问题，将在今后的工作中进行完善。同时我们会在使用该系统的过程中，不断总结反馈的信息，希望该系统能在未来广泛推广使用，服务更多的人群。