**个人博客系统设计文档**

**作品名称： 个人博客**

**作 者： 晏司举**

目录

[摘 要 5](#_Toc524459055)

[1 引言 6](#_Toc524459056)

[1.1 编写目的 6](#_Toc524459057)

[1.2 背景 6](#_Toc524459058)

[1.3 运行环境 6](#_Toc524459059)

[1.3.1服务器端运行环境 6](#_Toc524459060)

[1.3.2 PC端运行环境 7](#_Toc524459061)

[2 需求分析 7](#_Toc524459062)

[2.1 系统功能列表 7](#_Toc524459063)

[2.2 需求描述 9](#_Toc524459064)

[2.2.1. 用户管理 9](#_Toc524459065)

[2.2.2.博客管理 10](#_Toc524459066)

[2.2.3.资源管理 11](#_Toc524459067)

[2.2.4.其它 11](#_Toc524459068)

[2.3 系统用例图 11](#_Toc524459069)

[2.4 用例规约 13](#_Toc524459070)

[表2.1 注册用户用例规约 13](#_Toc524459071)

[表2.2 登录用例规约 14](#_Toc524459072)

[表2.3 重置密码用例规约 15](#_Toc524459073)

[表2.4 忘记密码用例规约 17](#_Toc524459074)

[表2.5 找回用户名用例规约 19](#_Toc524459075)

[表2.6 发表博客用例规约 20](#_Toc524459076)

[表2.7 编辑博客用例规约 21](#_Toc524459077)

[表2.8 评论博客用例规约 22](#_Toc524459078)

[表2.9 删除博客用例规约 23](#_Toc524459079)

[表2.10 收藏博客用例规约 24](#_Toc524459080)

[表2.11 查看收藏用例规约 25](#_Toc524459081)

[表2.12 上传资源用例规约 26](#_Toc524459082)

[表2.13 下载资源用例规约 27](#_Toc524459083)

[表2.14 查看新闻用例规约 28](#_Toc524459084)

[表2.15 查看收藏用例规约 29](#_Toc524459085)

[表2.16 关注用例规约 30](#_Toc524459086)

[表2.17 查看我的粉丝用例规约 30](#_Toc524459087)

[表2.18 查看我的关注用例规约 31](#_Toc524459088)

[3 概要设计说明书 33](#_Toc524459089)

[3.1 功能模块图 33](#_Toc524459090)

[3.2 后台系统架构 33](#_Toc524459091)

[4 数据库设计说明书 34](#_Toc524459092)

[4.1 数据库分析 34](#_Toc524459093)

[4.2 数据库表设计 35](#_Toc524459094)

[4.2.1 表0:[用户]表 35](#_Toc524459095)

[4.2.2 表1:[我的收藏]表 36](#_Toc524459096)

[4.2.3 ----🡪 表2:[收藏标签]表 38](#_Toc524459097)

[4.2.4 表3:[博客]表 39](#_Toc524459098)

[4.2.5 表4:[博客评论]表 41](#_Toc524459099)

[4.2.6 表5:[博客标签]表 42](#_Toc524459100)

[4.2.7 表6:[博问]表 44](#_Toc524459101)

[4.2.8 表7:[博问评论]表 45](#_Toc524459102)

[4.2.9 表8:[博问标签]表 47](#_Toc524459103)

[4.2.10 表9:[资源]表 48](#_Toc524459104)

[4.2.11 表10:[资源标签]表 50](#_Toc524459105)

[5 详细设计说明书 52](#_Toc524459106)

[5.1开发工具和开发技术的选择 52](#_Toc524459107)

[5.1功能设计 52](#_Toc524459108)

[6 测试 61](#_Toc524459109)

[7用户操作手册 61](#_Toc524459110)

[7.1 61](#_Toc524459111)

[总 结 62](#_Toc524459112)

[参考文献 63](#_Toc524459113)

[附录 64](#_Toc524459114)

# 摘 要

本系统为华为手机商城前台设计系统，用户可以进行注册、登录，登录成功后跳转至首页，可以查看商城所有商品，亦或分类查询或按照价格销量高低查询商城商品，可以在右上角搜索框输入商品关键字进行模糊查询，可以点击商品图片进入单个商品信息展示界面，每个登录用户均可以在该页面下方评论一次。可选择商品颜色、配置及数量将商品添加至购物车,并在购物车页面进行结算，还可以点击导航栏的“我的订单”查看用户的所有订单详情。

关键词：首页，查询，评论，购物车，订单详情

# 1 引言

## 1.1 编写目的

本系统设计说明书编写的⽬目的是为了说明系统概要设计的技术方案，明确该项目的系统功能和设计模式，清楚系统的整体框架和需求，为软件开发人员有针对性地进行系统开发、测试提供理论基础支持。本设计说明书的预期读者为：软件开发人员、测试人员。

## 1.2 背景

项目的任务提出者：晏司举

项目开发者为：晏司举

## 1.3 运行环境

### 1.3.1服务器端运行环境

CPU：Inter Core i3 及以上

内存：2GB及以上

硬盘空间：500GB及以上

⽹络速度：10M及以上

操作系统： CentOS 6.8

应⽤服务器：Tomcat 8.0

### 1.3.2 PC端运行环境

CPU：800Mhz及以上

硬盘空间：512MB 及以上分辨率：320x480及以上

操作系统：windows xp及以上

# 2 需求分析

## 2.1 系统功能列表

|  |  |  |  |
| --- | --- | --- | --- |
| 模块 | 功能编号 | 功能名称 | 功能概述 |
| 用户管理 | 1 | 登录 | 用户登录系统，从而可以执行某些操作 |
| 2 | 注册 | 注册成为系统用户 |
| 3 | 重置密码 | 修改密码 |
| 4 | 忘记密码 | 通过手机号或者邮箱找回密码 |
| 5 | 找回用户名 | 通过手机号或者邮箱找回用户名 |
| 6 | 查看收藏 | 查看自己收藏的博客 |
| 7 | 我的关注 | 查看自己关注了哪些人 |
| 8 | 我的粉丝 | 查看自己有那些粉丝 |
| 博客管理 | 1 | 发表博客 | 用户发表新的博客 |
| 2 | 编辑博客 | 用户修改现有博客 |
| 3 | 删除博客 | 用户删除某篇博客 |
| 4 | 点赞 | 用户对喜欢的博客点赞 |
| 5 | 收藏博客 | 用户收藏自己喜欢的博客 |
| 6 | 评论 | 用户对博客发表评论 |
| 博问管理 | 1 | 提问 | 用户提出问题 |
| 2 | 评论 | 回答他人提出的问题 |
| 资源管理 | 1 | 上传资源 | 上传分享的资源 |
| 2 | 下载资源 | 下载他人分享的资源 |
| 其它 | 1 | 新闻 | 查看今日热点新闻 |

## 2.2 需求描述

### 2.2.1. 用户管理

#### 1.登录

用户可以选择已有帐号登录﹑QQ登录或微信登录。若帐号或密码输入错误，则提示用户。

#### 2.注册

用户输入基本信息，完成注册。注册时，需要用手机号进行验证。

#### 3.忘记密码

若用户忘记密码，可以通过手机号和用户名修改密码。

#### 4.找回用户名

若用户忘记用户名，可以通过手机号找回

#### 5.查看收藏

用户查看收藏的博客

#### 6.关注

用户关注某个感兴趣的用户

#### 7.我的粉丝

用户查看自己的粉丝，即关注自己的人

#### 8.我关注的人

用户查看关注的人

### 2.2.2.博客管理

#### 1.发表博客

用户可以发表新的博客，为博客添加分类和标签。

#### 2.编辑博客

用户可以修改已经发布的博客或草稿箱中的博客

#### 3.删除博客

删除已经发表的博客，其他用户将看不到此博客

#### 4.点赞

点赞自己喜欢的博客

#### 5.评论

对博客发表评论

### 2.2.3.资源管理

#### 1.上传资源

用户可以上传代码﹑工具﹑文档等资源，分享给其他用户

#### 2.下载资源

用户可以下载其他用户上传的资源

### 2.2.4.其它

#### 1.查看新闻

用户可以查看当日的热点新闻

## 2.3 系统用例图

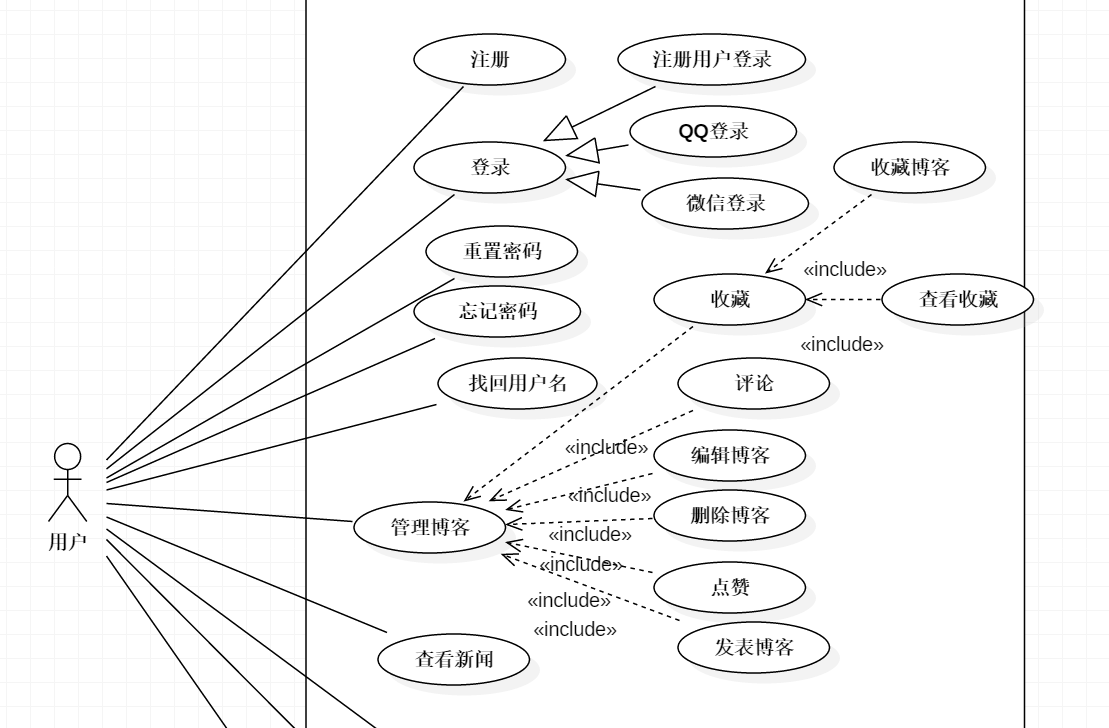


图 用例图(1)

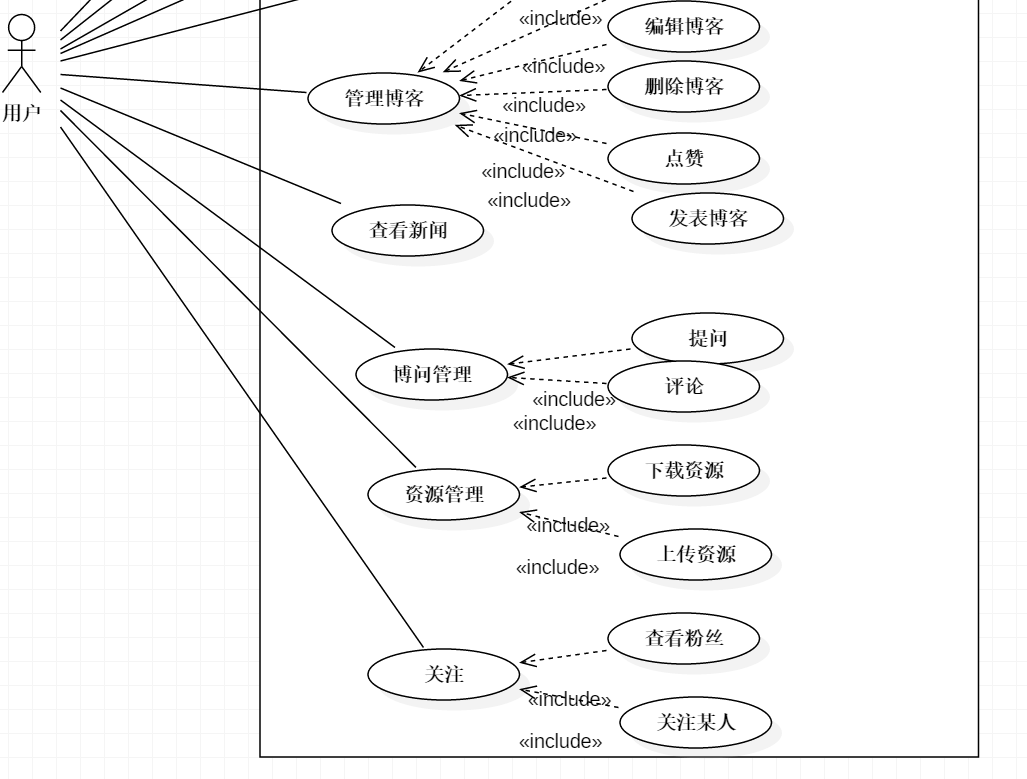


图 用例图(2)

## 2.4 用例规约

### 表2.1 注册用户用例规约

|  |  |
| --- | --- |
| 用例名称 | 注册用户 |
| 用例ID | 01 |
| 角色 | 游客 |
| 用例说明 | 游客注册用户 |
| 前置条件 | 无 |
| 基本事件流 | 1.游客进入注册用户页面  2.游客输入基本账户信息  3.游客点击“注册”按钮。  4.系统弹出“注册成功”信息页面。  5.5秒后页面跳转到登录页面，也可以点击“登录”链接直接进入登录页面 |
| 其他事件流 | 第2步：游客输入的用户名、昵称、密码、手机号为必选项，如果某一项没有填写，要给出提示信息。  第2步：游客输入的用户信息不符合格式要求，系统给出提示信息，要求重新输入。  第2步：如果有重名的用户名，则注册失败，给出提示信息：“此用户名已被注册”。  第2步：如果有重名的昵称，则注册失败，给出提示信息：“此昵称已存在”。 |
| 异常事件流 | 第3步，系统保存用户信息时出现系统故障，例如网络故障，数据库服务器故障，系统提示User：服务器或网络故障，注册失败。 |
| 后置条件 |  |

### 表2.2 登录用例规约

|  |  |
| --- | --- |
| 用例名称 | 登录 |
| 用例ID | 02 |
| 角色 | 游客 |
| 用例说明 | 用户登录系统 |
| 前置条件 | 注册（非必需） |
| 基本事件流 | 1.游客进入登录页面  2.游客可以使用已有帐号登录，也可以用QQ或微信登录  3.用户输入帐号或密码  3.游客点击“登录”按钮  4.登录成功，进入首页 |
| 其他事件流 | 第3步：用户输入的帐号或密码错误，停留在登录页面，并提示“帐号或密码错误”  第3步：用户输入的帐号或密码为空，登录失败，并提示“帐号或密码不能为空” |
| 异常事件流 |  |
| 后置条件 |  |

### 表2.3 重置密码用例规约

|  |  |
| --- | --- |
| 用例名称 | 重置密码 |
| 用例ID | 03 |
| 角色 | 登录用户 |
| 用例说明 | 修改密码 |
| 前置条件 | 登录系统 |
| 基本事件流 | 1.用户登录系统，进入首页  2.点击个人昵称，进入个人信息页面  3.点击“重置密码” ，进入重置密码页面  4.输入原密码﹑新密码和确认密码，点击确认，完成重置 |
| 其他事件流 | 第4步：两次输入的密码不一致，密码修改失败，并提示用户：两次输入的密码不一致  第4步：原密码输入错误，密码修改失败，并提示用户：密码错误,请输入正确密码  第4步：用户未输入确认密码，提示用户：请输入确认密码  第4步：用户未输入新密码，提示用户：请输入密码  第4步：密码格式不符合要求，密码修改失败，并提示用户：密码格式错误  第3步：该用户不是本系统注册用户，无法修改密码，并提示用户：您不是本系统注册用户，不能修改密 |
| 异常事件流 |  |
| 后置条件 |  |

### 表2.4 忘记密码用例规约

|  |  |
| --- | --- |
| 用例名称 | 忘记密码 |
| 用例ID | 04 |
| 角色 | 注册用户 |
| 用例说明 | 注册用户找回密码 |
| 前置条件 | 注册 |
| 基本事件流 | 1.用户在首页点击“登录”，进入登录页面  2.点击“忘记密码”按钮，进入修改密码页面  3.输入用户名﹑手机号，获取短信验证码  4.输入手机短信验证码，进入下一步  5.输入新的密码和确认密码，点击确认，密码修改成功 |
| 其他事件流 | 第3步：手机号格式错误，无法进入下一步，并提示用户：手机号格式错误  第3步：手机号与用户名不一致，无法进入下一步，并提示用户：手机号与用户名不一致  第3步：输入的用户名不存在，提示用户：用户名不存在  第4步：验证码输入错误，无法进入下一步，并提示用户：手机验证码错误  第5步：两次输入的密码不一致，密码修改失败，并提示用户：两次输入的密码不一致  第5步：密码格式不符合要求，密码修改失败，并提示用户：密码格式错误  第5步：用户未输入密码，提示用户：密码不能为空  第5步：用户未输入确认密码，提示用户：请输入确认密码 |
| 异常事件流 |  |
| 后置条件 |  |

### 表2.5 找回用户名用例规约

|  |  |
| --- | --- |
| 用例名称 | 找回用户名 |
| 用例ID | 05 |
| 角色 | 系统注册用户 |
| 用例说明 | 找回用户名 |
| 前置条件 | 注册 |
| 基本事件流 | 1.用户在首页点击“登录”，进入登录页面  2.点击“找回用户名”按钮，进入信息提交页面  3.输入手机号,获取验证码  4.输入手机短信验证码，点击“下一步”  5.用户在手机上查收用户名 |
| 其他事件流 | 第3步：手机号格式错误，无法进入下一步，并提示用户：手机号格式错误  第3步：验证码输入错误，无法获取用户名，并提示用户：验证码错误 |
| 异常事件流 |  |
| 后置条件 |  |

### 表2.6 发表博客用例规约

|  |  |
| --- | --- |
| 用例名称 | 发表博客 |
| 用例ID | 06 |
| 角色 | 登录用户 |
| 用例说明 | 用户发表新的博客 |
| 前置条件 | 登录系统 |
| 基本事件流： | 1.用户登录系统  2.用户点击“我的博客”,进入博客首页  3.点击“新随笔”，进入博客编辑页面  4.编辑博客，并给博客添加分类和标签  5.点击发布，成功发表博客 。用户也可以将博客存为草稿，下次继续编辑 |
| 其他事件流： | 第5步：用户未输入标题或内容即发表博客，提示用户“请输入标题或内容” |
| 异常事件流： |  |
| 后置条件： |  |

### 表2.7 编辑博客用例规约

|  |  |
| --- | --- |
| 用例名称 | 编辑博客 |
| 用例ID | 07 |
| 角色 | 登录用户 |
| 用例说明 | 修改已发表/草稿箱中的博客 |
| 前置条件 | 登录系统 |
| 基本事件流 | 1.用户登录系统  2.用户点击“我的博客”,进入博客首页  3.点击“管理”，进入博客管理页面  4.选中欲编辑的博客，点击“编辑”  5.进入编辑页面，编辑博客  6.编辑完成，点击“发表”或“存为草稿” |
| 其他事件流 | 第6步：用户未输入标题或内容即保存或存为草稿，提示用户“请输入标题或内容” |
| 异常事件流 |  |
| 后置条件 |  |

### 表2.8 评论博客用例规约

|  |  |
| --- | --- |
| 用例名称 | 评论博客 |
| 用例ID | 08 |
| 角色 | 登录用户 |
| 用例说明 | 用户评论他人博客 |
| 前置条件 | 登录系统 |
| 基本事件流 | 1.用户登录成功后进入首页  2.选中欲评论的博客，点击进入  3.输入评论  4.点击“提交评论”按钮，提交评论 |
| 其他事件流： | 第4步：用户没有登录即提交评论，评论失败，提示用户：您还没有登录,请先登录  第4步：评论中含有敏感词，评论失败并提示用户：评论中含有敏感词 |
| 异常事件流： |  |
| 后置条件： |  |

### 表2.9 删除博客用例规约

|  |  |
| --- | --- |
| 用例名称 | 删除博客 |
| 用例ID | 09 |
| 角色 | 登录用户 |
| 用例说明 | 用户删除博客 |
| 前置条件 | 登录系统 |
| 基本事件流： | 1.登录系统，进入首页  2.用户点击“我的博客”,进入博客首页  3.点击“管理”，进入博客管理页面  4.选中欲删除的博客，点击“删除”，即可删除博客 |
| 其他事件流： |  |
| 异常事件流： |  |
| 后置条件： |  |

### 表2.10 收藏博客用例规约

|  |  |
| --- | --- |
| 用例名称 | 收藏博客 |
| 用例ID | 10 |
| 角 色 | 登录用户 |
| 用例说明 | 收藏博客 |
| 前置条件 | 登录 |
| 基本事件流 | 1.用户登录成功后进入首页  2.选中欲收藏的博客，点击进入  3.点击收藏，弹出收藏窗口  4.输入标题﹑标签和摘要  5.点击“收藏”按钮，完成收藏 |
| 其他事件流 | 第5步：用户未输入标题，收藏失败，并提示用户：标题不能为空  第3步：用户没有登录，不能收藏博客，提示用户：您还没有登录 |
| 异常事件流 |  |
| 后置条件 |  |

### 表2.11 查看收藏用例规约

|  |  |
| --- | --- |
| 用例名称 | 查看收藏 |
| 用例ID | 11 |
| 角色 | 登录用户 |
| 用例说明 | 查看收藏的博客 |
| 前置条件 | 登录系统 |
| 基本事件流 | 1.用户登录系统，进入首页  2.点击昵称，进入个人信息页面  3.点击“收藏”，即可查看收藏的博客  4.用户也可以通过选中标签查看同类型的博客 |
| 其他事件流 |  |
| 异常事件流 |  |
| 后置条件 |  |

### 表2.12 上传资源用例规约

|  |  |
| --- | --- |
| 用例名称 | 上传资源 |
| 用例ID | 12 |
| 角色 | 登录用户 |
| 用例说明 | 上传资源，分享给其他用户 |
| 前置条件 | 登录系统 |
| 基本事件流 | 1.用户登录系统，进入首页  2.点击资源，进入资源管理页面  3.点击“上传资源”，进入上传资源页面  4.上传资源并填写资源名称﹑选择类型及所属分类﹑填写描述等信息  5.点击“提交”，完成资源上传 |
| 其他事件流 | 第3步：用户没有登录，进入登录页面  第4步：用户没有填写资源名称，提示用户：请填写资源的名称 |
| 异常事件流 |  |
| 后置条件： |  |

### 表2.13 下载资源用例规约

|  |  |
| --- | --- |
| 用例名称 | 下载资源 |
| 用例ID | 13 |
| 角色 | 登录用户 |
| 用例说明 | 下载用户分享的资源 |
| 前置条件 | 登录 |
| 基本事件流 | 1.用户登录系统，进入首页  2.点击资源，进入资源管理页面  3.选中欲下载的资源，进入资源下载页面  4.点击“立即下载”，开始下载资源 |
| 其他事件流 | 第4步：用户没有登录，进入登录页面 |
| 异常事件流 |  |
| 后置条件 |  |

### 表2.14 查看新闻用例规约

|  |  |
| --- | --- |
| 用例名称 | 查看新闻 |
| 用例ID | 14 |
| 角色 | 游客或登录用户 |
| 用例说明 | 查看热点新闻 |
| 前置条件 | 无 |
| 基本事件流 | 1.用户登录系统，进入首页  2.点击新闻，进入新闻页面  3.点击新闻标题进入详情页面，即可查看新闻详情 |
| 其他事件流 |  |
| 异常事件流 |  |
| 后置条件 |  |

### 表2.15 查看收藏用例规约

|  |  |
| --- | --- |
| 用例名称 | 查看收藏 |
| 用例ID | 15 |
| 角色 | 登录用户 |
| 用例说明 | 查看收藏的博客 |
| 前置条件 | 登录 |
| 基本事件流 | 1.用户登录系统，进入首页  2.点击昵称，进入个人信息页面  3.点击收藏，即可查看收藏的博客  4.用户也可以通过选定标签，查看特定类型的博客 |
| 其他事件流 |  |
| 异常事件流 |  |
| 后置条件 |  |

### 表2.16 关注用例规约

|  |  |
| --- | --- |
| 用例名称 | 关注 |
| 用例ID | 16 |
| 角色 | 登录用户 |
| 用例说明 | 用户关注某人 |
| 前置条件 | 登录 |
| 基本事件流 | 1.用户登录系统，进入首页  2.用户浏览博客时，在博客下方点击“关注我”，即可关注博主 |
| 其他事件流 |  |
| 异常事件流 |  |
| 后置条件 |  |

### 表2.17 查看我的粉丝用例规约

|  |  |
| --- | --- |
| 用例名称 | 查看我的粉丝 |
| 用例ID | 17 |
| 角色 | 登录用户 |
| 用例说明 | 查看自己的粉丝 |
| 前置条件 | 登录 |
| 基本事件流 | 1.用户登录系统，进入首页  2.点击昵称，进入个人信息页面  3.点击“我的粉丝”，即可查看自己的粉丝 |
| 其他事件流 |  |
| 异常事件流 |  |
| 后置条件 |  |

### 表2.18 查看我的关注用例规约

|  |  |
| --- | --- |
| 用例名称 | 查看我的关注 |
| 用例ID | 18 |
| 角色 | 登录用户 |
| 用例说明 | 查看自己关注的人 |
| 前置条件 | 登录 |
| 基本事件流 | 1.用户登录系统，进入首页  2.点击昵称，进入个人信息页面  3.点击“我的关注”，即可查看自己关注的用户 |
| 其他事件流 |  |
| 异常事件流 |  |
| 后置条件 |  |

# 3 概要设计说明书

## 3.1 功能模块图

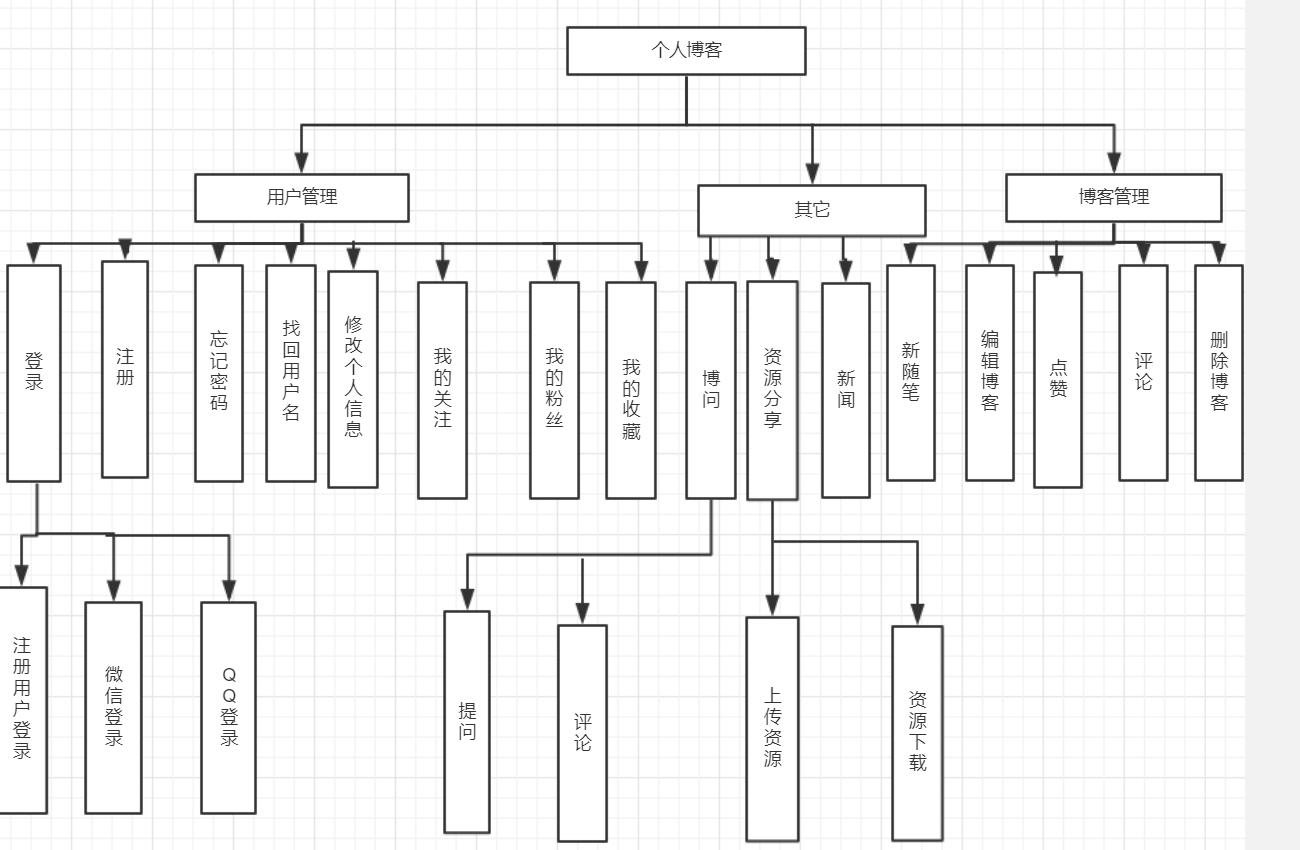


图3.1 系统功能模块图

## 3.2 后台系统架构

图3.2 后台系统架构

# 4 **数据库设计说明书**

## 4.1 数据库分析

概念模型⽤用于信息世界的建模，与具体的 DBMS ⽆无关。为了把现实世界中的具体事物抽象、组织为某一DBMS支持的数据模型。人们常首先将现实世界抽象为信息世界，然后再将信息世界转换为机器世界。也就是说，首先把现实世界中的客观对象抽象为某⼀种信息结构，这种信息结构并不依赖于具体的计算机系统和具体的DBMS，而是概念级的模型，然后再把模型转换为计算机上某一个 DBMS支持的数据模型。实际上，概念模型是现实世界到机器世界的⼀一个中间层次。

1. 实体 (entity)

客观存在并可相互区别的事物称为实体。实体可以是具体的⼈人、事、物，也可以是抽象的概念或联系。例例如，⼀一个⽤用户、⼀一⻔门课程、⼀一个试题、⼀一个视频等都是实体。

1. 联系 (relationship)

在现实世界中，事物内部以及事物之间是有联系的，这些联系在信息世界中反映为实体内部的联系

和实体之间的联系。实体内部的联系通常是组成实体的各属性之间的联系。两个实体型之间的联系可以 分为 3 类，⼀一对⼀一联系，(1:1)；⼀一对多联系(1:n)；多对多联系(m:n)。

概念模型是对信息世界建模，所以概念模型应该能够⽅方便便、准确地表示信息世界中的常⽤用概念。概 念模型的表示⽅方法很多，其中最为常⽤用的是P.P.S.Chen 于 1976年年提出的实体，联系⽅方法 (Entity-Relationship Approach)简记为 E-R 表示法)。该⽅方法⽤用 E-R 图来描述现实世界的概念模型，称为 实体-联系模型，简称E-R 模型。

## 4.2 数据库表设计

### 4.2.1 表0:[用户]表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 表名 | | user | | | | | |
| 数据库用户 | |  | | | | | |
| 主键 | | userId | | | | | |
| 其他排序字段 | |  | | | | | |
| 索引字段 | | userId | | | | | |
| 序号 | 字段名称 | 数据类型（精度范围） | 允许为空Y/N | 唯一Y/N | 区别度 | 默认值 | 约束条件/说明 |
| *0* | userId | int(11) | *N* | Y |  | 无 | *用户id（主键）* |
| 1 | userName | *varchar(6~12)* | *N* | *Y* |  | 无 | *用户名* |
| 2 | password | *varchar(255)* | *N* | *Y* |  | 无 | *密码（密文存储）* |
| 3 | tel | *varchar(15)* | *N* | *Y* |  | 无 | *手机号码* |
| 4 | nickName | *varchar(6~30)* | *N* | *Y* |  | 无 | *昵称* |
| sql脚本 | | DROP TABLE IF EXISTS `shoppingtrolley`;  CREATE TABLE `shoppingtrolley` (  `id` int(11) NOT NULL AUTO\_INCREMENT,  `userId` int(11) NOT NULL,  PRIMARY KEY (`id`),  KEY `fk\_shoppingTrolley\_userId` (`userId`),  CONSTRAINT `fk\_shoppingTrolley\_userId` FOREIGN KEY (`userId`) REFERENCES `user` (`id`) ON DELETE CASCADE ON UPDATE NO ACTION  ) ENGINE=InnoDB AUTO\_INCREMENT=2215 DEFAULT CHARSET=utf8; | | | | | |
| 备注 | | *无* | | | | | |
|  | |  | | | | | |

### 4.2.2 表1:[我的收藏]表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 表名 | | collection | | | | | |
| 数据库用户 | |  | | | | | |
| 主键 | | blogId﹑userId | | | | | |
| 其他排序字段 | |  | | | | | |
| 索引字段 | | blogId﹑userId | | | | | |
| 序号 | 字段名称 | 数据类型（精度范围） | 允许为空Y/N | 唯一Y/N | 区别度 | 默认值 | 约束条件/说明 |
| 0 | blogId | *int(11)* | *N* | *N* |  | 无 | *博客id(外键﹑联合主键)* |
| 1 | userId | *int(11)* | *N* | *N* |  |  | *用户id(外键﹑联合主键)* |
| 2 | title | *varchar(30)* | *N* | *N* |  | 无 | *标题* |
| 3 | abstract | *varchar(200)* | *Y* | *N* |  | 无 | *摘要* |
| sql脚本 | | DROP TABLE IF EXISTS `shoppingtrolley`;  CREATE TABLE `shoppingtrolley` (  `id` int(11) NOT NULL AUTO\_INCREMENT,  `userId` int(11) NOT NULL,  PRIMARY KEY (`id`),  KEY `fk\_shoppingTrolley\_userId` (`userId`),  CONSTRAINT `fk\_shoppingTrolley\_userId` FOREIGN KEY (`userId`) REFERENCES `user` (`id`) ON DELETE CASCADE ON UPDATE NO ACTION  ) ENGINE=InnoDB AUTO\_INCREMENT=2215 DEFAULT CHARSET=utf8; | | | | | |
| 备注 | | *无* | | | | | |
|  | |  | | | | | |

### 4.2.3 ----🡪 表2:[收藏标签]表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 表名 | | collectionTag | | | | | |
| 数据库用户 | |  | | | | | |
| 主键 | | tagId | | | | | |
| 其他排序字段 | |  | | | | | |
| 索引字段 | | tagId | | | | | |
| 序号 | 字段名称 | 数据类型（精度范围） | 允许为空Y/N | 唯一Y/N | 区别度 | 默认值 | 约束条件/说明 |
| *0* | tagId | int(11) | *N* | Y |  | 无 | *主键（标签id）* |
| 1 | tagName | *Varchar(0~10)* | *N* | *N* |  | 无 | 标签名 |
| 2 | collectionId | *int(11)* | *N* | *N* |  | 无 | *收藏id(*外键*)* |
| sql脚本 | | DROP TABLE IF EXISTS `shoppingtrolley`;  CREATE TABLE `shoppingtrolley` (  `id` int(11) NOT NULL AUTO\_INCREMENT,  `userId` int(11) NOT NULL,  PRIMARY KEY (`id`),  KEY `fk\_shoppingTrolley\_userId` (`userId`),  CONSTRAINT `fk\_shoppingTrolley\_userId` FOREIGN KEY (`userId`) REFERENCES `user` (`id`) ON DELETE CASCADE ON UPDATE NO ACTION  ) ENGINE=InnoDB AUTO\_INCREMENT=2215 DEFAULT CHARSET=utf8; | | | | | |
| 备注 | | *无* | | | | | |
|  | |  | | | | | |

### 4.2.4 表3:[博客]表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 表名 | | blog | | | | | |
| 数据库用户 | |  | | | | | |
| 主键 | | blogId | | | | | |
| 其他排序字段 | |  | | | | | |
| 索引字段 | | blogId | | | | | |
| 序号 | 字段名称 | 数据类型（精度范围） | 允许为空Y/N | 唯一Y/N | 区别度 | 默认值 | 约束条件/说明 |
| *0* | blogId | int(11) | *N* | Y |  | 无 | 博客id(主键) |
| 1 | title | Varchar(50) | *N* | *N* |  | 无 | *标题* |
| 2 | content |  | *N* | *N* |  | 无 | *博客内容* |
| 3 | publishTime | *datetime* | *N* | *N* |  | 无 | *发布时间* |
| 4 | authorId | *int(11)* | *N* | *N* |  | 无 | *作者id（外键）* |
| 5 | praiseCount | *int(11)* | *N* | *N* |  | 0 | *点赞数* |
| 6 | commentCount | *int(11)* | *N* | *N* |  | 0 | *评论数* |
| 7 | category | *varchar(15)* | *Y* | *N* |  |  | *类别* |
| 8 | state | *int* | *N* | *N* |  | 无 | *博客状态(0:已发布 1：已删除 -1：草稿)* |
| sql脚本 | |  | | | | | |
| 备注 | | *无* | | | | | |
|  | |  | | | | | |

### 4.2.5 表4:[博客评论]表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 表名 | | blogComment | | | | | |
| 数据库用户 | |  | | | | | |
| 主键 | | commentId | | | | | |
| 其他排序字段 | |  | | | | | |
| 索引字段 | | commentId | | | | | |
| 序号 | 字段名称 | 数据类型（精度范围） | 允许为空Y/N | 唯一Y/N | 区别度 | 默认值 | 约束条件/说明 |
| *0* | commentId | int(11) | *N* | Y |  | 无 | *评论id（主键）* |
| 1 | authorId | *int(11)* | *N* | *N* |  | 无 | *评论人id(*外键*)* |
| 2 | blogId | *int(11)* | *N* | *N* |  | 无 | *博客id(*外键*)* |
| 3 | comment | *varchar(255)* | *N* | *N* |  | 无 | *评论内容* |
| 4 | time | *datetime* | *N* | *N* |  | 无 | *评论时间* |
| 5 | type | *int* | *N* | *N* |  | 无 | *标识评论是否为回复(0:不是回复 1:是回复)* |
| 6 | userId | *int(11)* | *Y* | *N* |  | 无 | *被回复人id(外键)* |
| sql脚本 | | DROP TABLE IF EXISTS `shoppingtrolley`;  CREATE TABLE `shoppingtrolley` (  `id` int(11) NOT NULL AUTO\_INCREMENT,  `userId` int(11) NOT NULL,  PRIMARY KEY (`id`),  KEY `fk\_shoppingTrolley\_userId` (`userId`),  CONSTRAINT `fk\_shoppingTrolley\_userId` FOREIGN KEY (`userId`) REFERENCES `user` (`id`) ON DELETE CASCADE ON UPDATE NO ACTION  ) ENGINE=InnoDB AUTO\_INCREMENT=2215 DEFAULT CHARSET=utf8; | | | | | |
| 备注 | | *无* | | | | | |
|  | |  | | | | | |

### 4.2.6 表5:[博客标签]表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 表名 | | blogTag | | | | | |
| 数据库用户 | |  | | | | | |
| 主键 | | tagId | | | | | |
| 其他排序字段 | |  | | | | | |
| 索引字段 | | tagId | | | | | |
| 序号 | 字段名称 | 数据类型（精度范围） | 允许为空Y/N | 唯一Y/N | 区别度 | 默认值 | 约束条件/说明 |
| *0* | tagId | int(11) | *N* | Y |  | 无 | *标签id（主键）* |
| 1 | tagName | *Varchar(0~10)* | *N* | *N* |  | 无 | 标签名 |
| 2 | blogId | *int(11)* | *N* | *N* |  | 无 | *博客id(*外键*)* |
| sql脚本 | | DROP TABLE IF EXISTS `shoppingtrolley`;  CREATE TABLE `shoppingtrolley` (  `id` int(11) NOT NULL AUTO\_INCREMENT,  `userId` int(11) NOT NULL,  PRIMARY KEY (`id`),  KEY `fk\_shoppingTrolley\_userId` (`userId`),  CONSTRAINT `fk\_shoppingTrolley\_userId` FOREIGN KEY (`userId`) REFERENCES `user` (`id`) ON DELETE CASCADE ON UPDATE NO ACTION  ) ENGINE=InnoDB AUTO\_INCREMENT=2215 DEFAULT CHARSET=utf8; | | | | | |
| 备注 | | *无* | | | | | |
|  | |  | | | | | |

### 4.2.7 表6:[博问]表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 表名 | | question | | | | | |
| 数据库用户 | |  | | | | | |
| 主键 | | questionId | | | | | |
| 其他排序字段 | |  | | | | | |
| 索引字段 | | questionId | | | | | |
| 序号 | 字段名称 | 数据类型（精度范围） | 允许为空Y/N | 唯一Y/N | 区别度 | 默认值 | 约束条件/说明 |
| *0* | questionId | int(11) | *N* | Y |  | 无 | 博问id(主键) |
| 1 | title | varchar(30) | *N* | *Y* |  | 无 | *标题* |
| 2 | content | varchar(255) | *N* | *N* |  | 无 | *博问内容* |
| 3 | publishTime | *datetime* | *N* | *N* |  | 无 | *提问时间* |
| 4 | authorId | *int(11)* | *N* | *N* |  | 无 | *作者id（外键）* |
| 5 | commentCount | *int(11)* | *N* | *N* |  | 0 | *评论数* |
| sql脚本 | |  | | | | | |
| 备注 | | *无* | | | | | |
|  | |  | | | | | |

### 4.2.8 表7:[博问评论]表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 表名 | | questionComment | | | | | |
| 数据库用户 | |  | | | | | |
| 主键 | | commentId | | | | | |
| 其他排序字段 | |  | | | | | |
| 索引字段 | | commentId | | | | | |
| 序号 | 字段名称 | 数据类型（精度范围） | 允许为空Y/N | 唯一Y/N | 区别度 | 默认值 | 约束条件/说明 |
| *0* | commentId | int(11) | *N* | Y |  | 无 | *评论id（主键）* |
| 1 | authorId | *int(11)* | *N* | *N* |  | 无 | *评论人id(*外键*)* |
| 2 | questionId | *int(11)* | *N* | *N* |  | 无 | *博问id(*外键*)* |
| 3 | comment | *varchar(255)* | *N* | *N* |  | 无 | *评论内容* |
| 4 | time | *datetime* | *N* | *N* |  |  | *评论时间* |
| 5 | type | *int* | *N* | *N* |  | 无 | *标识评论是否为回复(0:不是回复 1:是回复)* |
| 6 | userId | *int(11)* | *Y* | *N* |  | 无 | *被回复人id(外键)* |
| sql脚本 | | DROP TABLE IF EXISTS `shoppingtrolley`;  CREATE TABLE `shoppingtrolley` (  `id` int(11) NOT NULL AUTO\_INCREMENT,  `userId` int(11) NOT NULL,  PRIMARY KEY (`id`),  KEY `fk\_shoppingTrolley\_userId` (`userId`),  CONSTRAINT `fk\_shoppingTrolley\_userId` FOREIGN KEY (`userId`) REFERENCES `user` (`id`) ON DELETE CASCADE ON UPDATE NO ACTION  ) ENGINE=InnoDB AUTO\_INCREMENT=2215 DEFAULT CHARSET=utf8; | | | | | |
| 备注 | | *无* | | | | | |
|  | |  | | | | | |

### 4.2.9 表8:[博问标签]表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 表名 | | questionTag | | | | | |
| 数据库用户 | |  | | | | | |
| 主键 | | tagId | | | | | |
| 其他排序字段 | |  | | | | | |
| 索引字段 | | tagId | | | | | |
| 序号 | 字段名称 | 数据类型（精度范围） | 允许为空Y/N | 唯一Y/N | 区别度 | 默认值 | 约束条件/说明 |
| *0* | tagId | int(11) | *N* | Y |  | 无 | *标签id（主键）* |
| 1 | tagName | *Varchar(0~10)* | *N* | *N* |  | 无 | 标签名 |
| 2 | questionId | *int(11)* | *N* | *N* |  | 无 | *博问id(*外键*)* |
| sql脚本 | | DROP TABLE IF EXISTS `shoppingtrolley`;  CREATE TABLE `shoppingtrolley` (  `id` int(11) NOT NULL AUTO\_INCREMENT,  `userId` int(11) NOT NULL,  PRIMARY KEY (`id`),  KEY `fk\_shoppingTrolley\_userId` (`userId`),  CONSTRAINT `fk\_shoppingTrolley\_userId` FOREIGN KEY (`userId`) REFERENCES `user` (`id`) ON DELETE CASCADE ON UPDATE NO ACTION  ) ENGINE=InnoDB AUTO\_INCREMENT=2215 DEFAULT CHARSET=utf8; | | | | | |
| 备注 | | *无* | | | | | |
|  | |  | | | | | |

### 4.2.10 表9:[资源]表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 表名 | | source | | | | | |
| 数据库用户 | |  | | | | | |
| 主键 | | sourceId | | | | | |
| 其他排序字段 | |  | | | | | |
| 索引字段 | | sourceId | | | | | |
| 序号 | 字段名称 | 数据类型（精度范围） | 允许为空Y/N | 唯一Y/N | 区别度 | 默认值 | 约束条件/说明 |
| *0* | sourceId | int(11) | *N* | Y |  | 无 | *资源id（主键）* |
| 1 | name | *Varchar(25)* | *N* | *N* |  | 无 | *资源名* |
| 2 | type | *Varchar(4)* | *N* | *N* |  |  | *资源类型（“代码”﹑“文档”﹑“工具”﹑“其它”）* |
| 3 | category | *varchar(8)* | *N* | *N* |  | 无 | *类别* |
| 4 | descreption | *Varchar(0~50)* | *Y* | *N* |  | 无 | *描述* |
| 5 | authorId | *int(11)* | *N* | *N* |  | 无 | *上传人（外键）* |
| 6 | uploadTime | *datetime* | *N* | *N* |  | 无 | *上传时间* |
| 7 | path | *varchar(255)* | *N* | *Y* |  | 无 | *资源保存路径* |
| sql脚本 | | DROP TABLE IF EXISTS `shoppingtrolley`;  CREATE TABLE `shoppingtrolley` (  `id` int(11) NOT NULL AUTO\_INCREMENT,  `userId` int(11) NOT NULL,  PRIMARY KEY (`id`),  KEY `fk\_shoppingTrolley\_userId` (`userId`),  CONSTRAINT `fk\_shoppingTrolley\_userId` FOREIGN KEY (`userId`) REFERENCES `user` (`id`) ON DELETE CASCADE ON UPDATE NO ACTION  ) ENGINE=InnoDB AUTO\_INCREMENT=2215 DEFAULT CHARSET=utf8; | | | | | |
| 备注 | | *无* | | | | | |
|  | |  | | | | | |

### 4.2.11 表10:[资源标签]表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 表名 | | sourceTag | | | | | |
| 数据库用户 | |  | | | | | |
| 主键 | | tagId | | | | | |
| 其他排序字段 | |  | | | | | |
| 索引字段 | |  | | | | | |
| 序号 | 字段名称 | 数据类型（精度范围） | 允许为空Y/N | 唯一Y/N | 区别度 | 默认值 | 约束条件/说明 |
| *0* | tagId | int(11) | *N* | Y |  | 无 | *标签id（主键）* |
| 1 | tagName | *varchar(0~10)* | *N* | *N* |  | 无 | 标签名 |
| 2 | source Id | *int(11)* | *N* | *N* |  | 无 | *资源id(*外键*)* |
| sql脚本 | | DROP TABLE IF EXISTS `shoppingtrolley`;  CREATE TABLE `shoppingtrolley` (  `id` int(11) NOT NULL AUTO\_INCREMENT,  `userId` int(11) NOT NULL,  PRIMARY KEY (`id`),  KEY `fk\_shoppingTrolley\_userId` (`userId`),  CONSTRAINT `fk\_shoppingTrolley\_userId` FOREIGN KEY (`userId`) REFERENCES `user` (`id`) ON DELETE CASCADE ON UPDATE NO ACTION  ) ENGINE=InnoDB AUTO\_INCREMENT=2215 DEFAULT CHARSET=utf8; | | | | | |
| 备注 | | *无* | | | | | |
|  | |  | | | | | |

# **5** 详细设计说明书

## 5.1开发工具和开发技术的选择

前端：BootStrap + Angular Js + Ajax

后台：spring + spring mvc + mybatis

数据库：MySQL + Mongodb + redis

开发工具：Eclipse + HBuilder

版本管理工具:Git

## 5.1功能设计

#### 5.1.1 用户管理模块

##### 1 登录

具体实现:用户进入登录页面，输入用户名和密码并通过Ajax将数据传到服务器端。服务器端获取数据后，用shiro对用户身份进行验证：如果验证失败则返回“fail”，验证成功则返回“success”，同时将collectionedBlog\_userId列表和praisedBlog\_userId列表发送至客户端。

且将用户信息保存至session中。客户端根据返回的信息做出判断：如果是“fail”，则提示用户“用户名或密码错误”；如果“success”，则跳转到首页“index.hmtl”，并将用户信息传递过去。

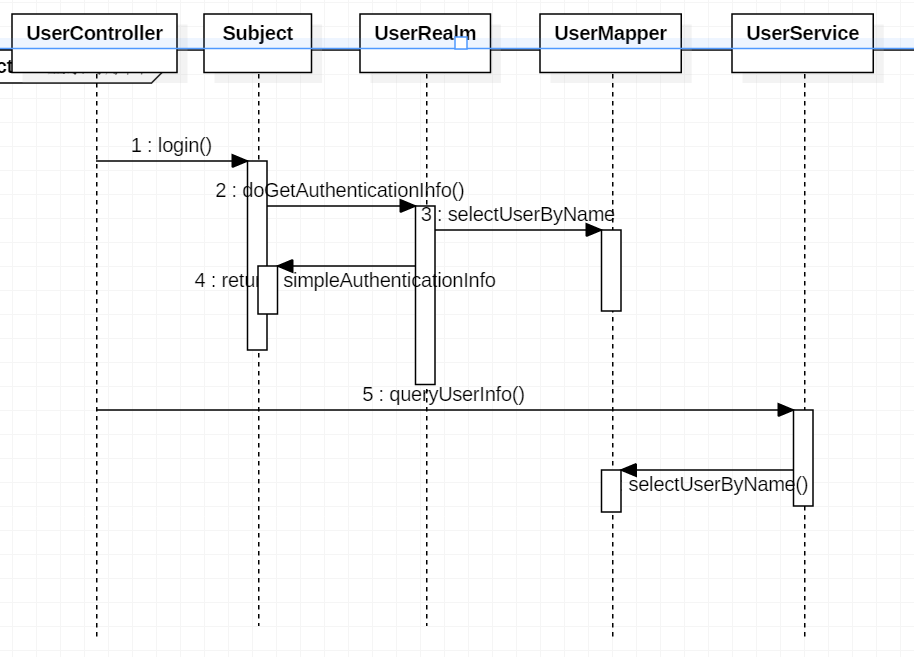


图 5.1 登录时序图

##### 2.注册

具体实现：用户输入用户名﹑昵称﹑密码以及手机号以获取手机验证码:将手机号发送至后台，后台调用SmsApi发送验证码，并将验证码传送至前台，与用户输入的验证码进行比对。若验证码未输入或者输入错误，则无法成功注册。用户输入的信息通过验证后，即使用Ajax将数据发送到服务器端。若用户名﹑手机号或者昵称重复，则注册失败，并将错误原因发送回客户端。若数据无误，则将用户信息添加至数据库,最后返回成功信息给前台。客户端根据服务器端传回的信息做出反应：若注册失败，则提示用户错误信息；若注册成功，则跳转至登录页面。

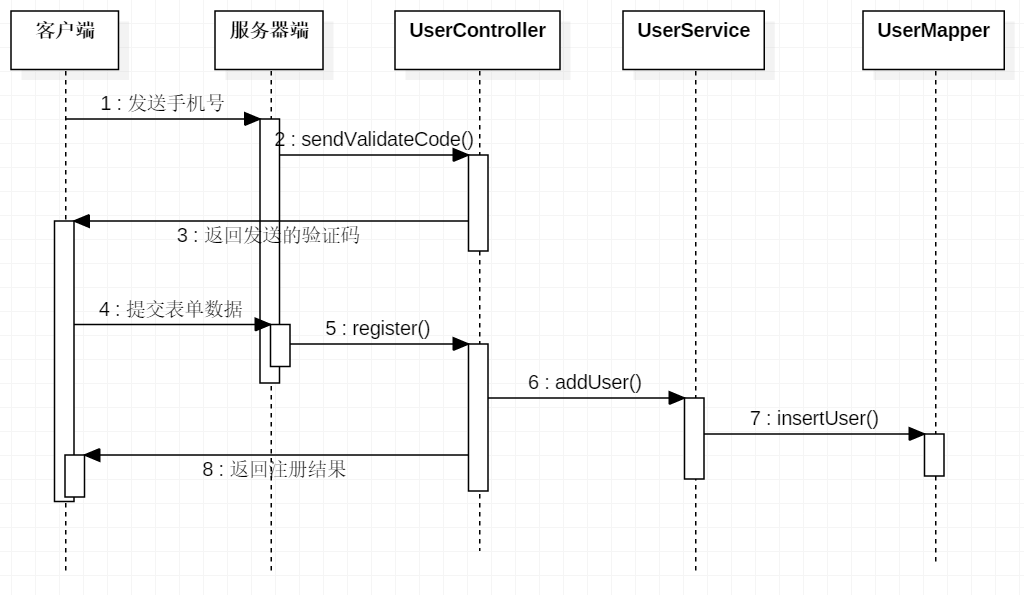


图 5.2 注册时序图

##### 3.忘记密码

##### 4.重置密码

##### 5. 忘记用户名

##### 6. 查看收藏

具体实现：用户进入个人信息页面后，点击收藏，即可查看收藏的博客。请求发送到后台，后台从数据库获取用户的收藏数据，并根据博客id获取博客内容，以json形式发送回前台。前台获取数据后，对页面进行渲染展示。

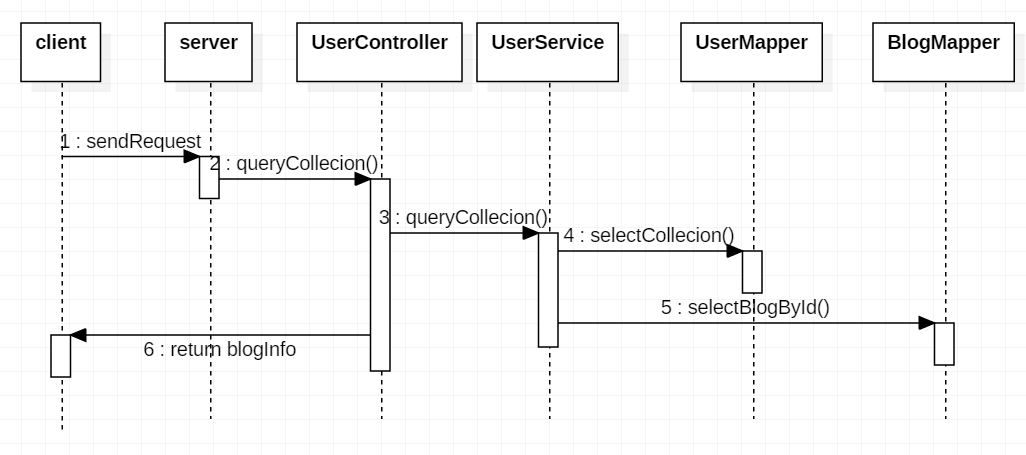


图5. 查看收藏时序图

##### 7.查看我的粉丝

具体实现：进入个人信息页面，点击“我的粉丝”，即可查看自己的粉丝。后台将MyFans\_userId列表发送到前台，前台遍历该列表，即可获取数据。

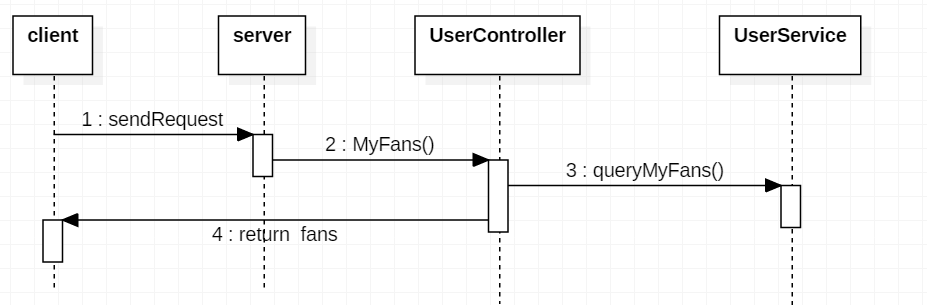


图5.7 我的粉丝时序图

##### 8.查看我的关注

具体实现：进入个人信息页面，点击“我关注的人”，即可查看自己关注的人。后台将MyAttention\_userId列表发送到前台，前台遍历该列表，即可获取数据。

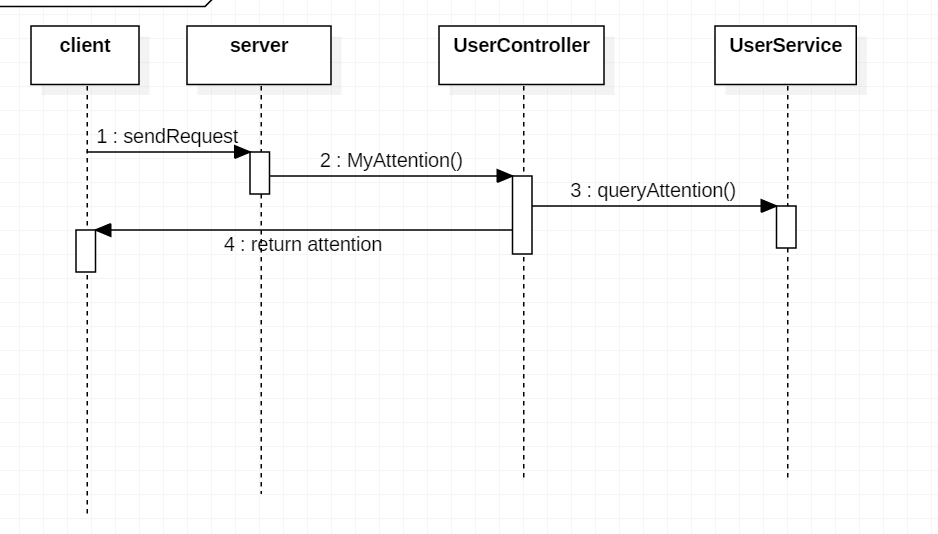


图5.8 我的关注时序图

##### 9.关注某人

具体实现：用户在浏览博客时，点击下方的“关注我”即可关注某人。使用Ajax将被关注人的id以及昵称发送到后台，同时从session中取出关注人的id以及昵称。将被关注人的id以及昵称存入MyAttention\_userId列表中,同时将关注人的id以及昵称存入被关注人的MyFans\_userId列表中。

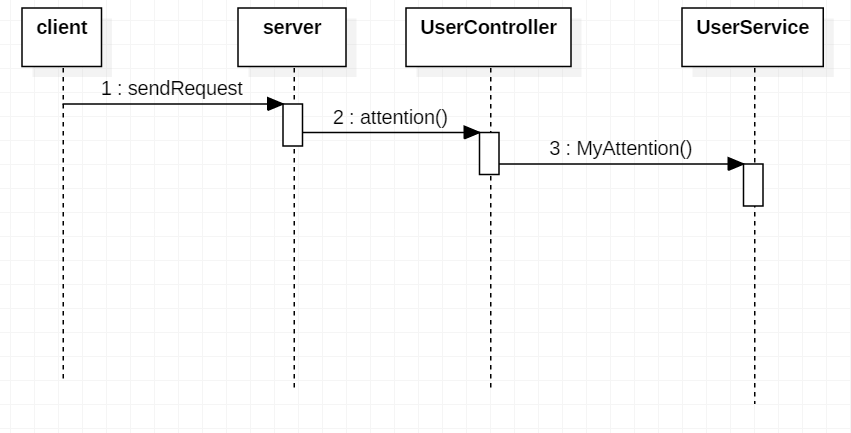


图5.9 关注时序图

#### 5.1.2 博客管理模块

##### 1.发表博客

具体实现：用户进入博客编辑页面编辑博客，编辑完成后，用户可以将博客保存为草稿，也可以立即发布。若博客标题或内容为空，则阻止提交，并提示用户“标题不能为空”或者“博客内容不能为空”。用户提交数据后，使用ajax将数据发送至服务器端，服务器端获取数据后，从session中取出用户id，将博客内容及与之关联的用户id存储至数据库。

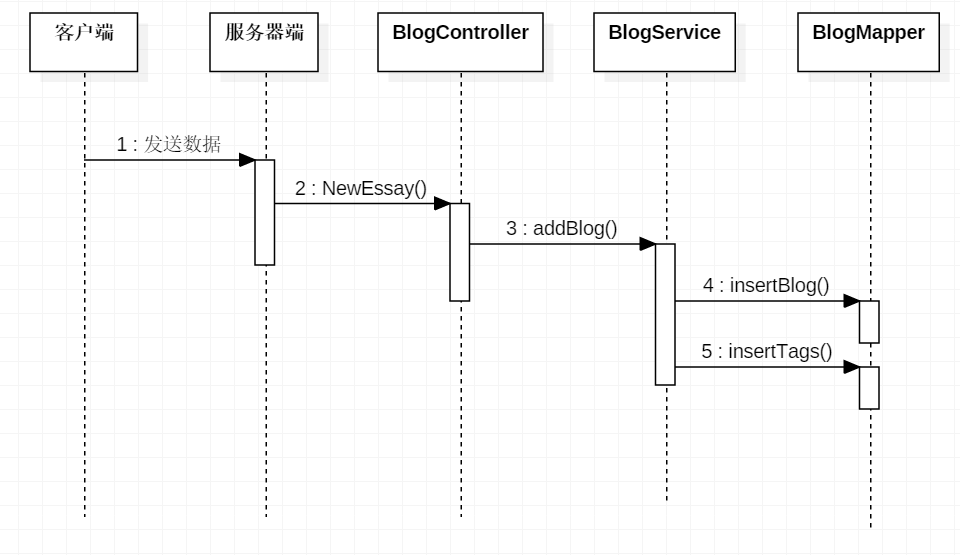


图5.10 发表博客时序图

##### 2.评论博客

具体实现：用户浏览博客时，进入博客详情页面即可评论。用户输入评论内容，点击提交即可。评论成功后，页面上即显示评论内容。同时使用Ajax将评论内容及博客id发送到后台。后台获取评论内容﹑时间﹑用户id及博客id后，构建评论对象，并将其保存至数据库，同时将相应博客的评论数+1。

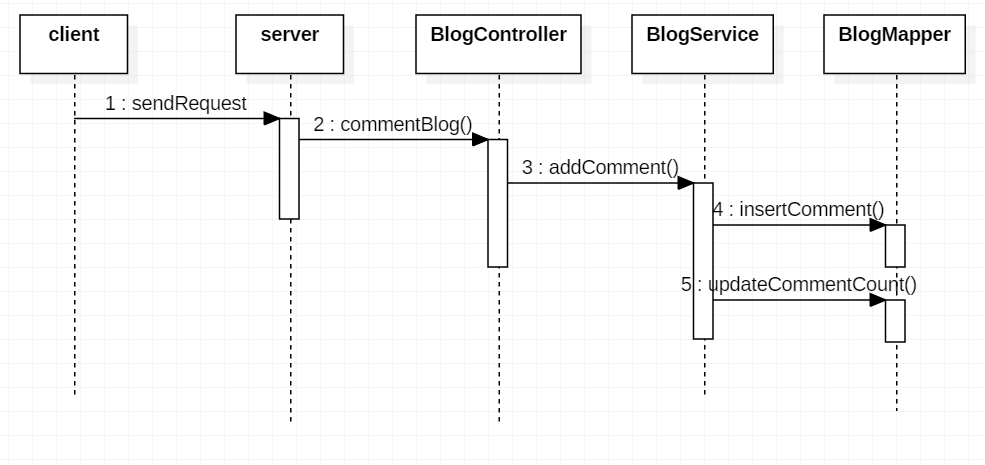


图5.11 评论博客时序图

##### 3.点赞

具体实现：用户点博客下的拇指按扭，即可对喜欢的博客点赞。用户点赞后，首先遍历praisedBlog\_userId列表判断其是否曾经点赞过该博客：若已经点赞过，则不做任何操作；否则使用Ajax将博客id发送至后台。接收到前台传来的数据后，将该博客id存入praisedBlog\_userId中，同时将其传回前台，更新前台数据，并将该博客的点赞数+1。

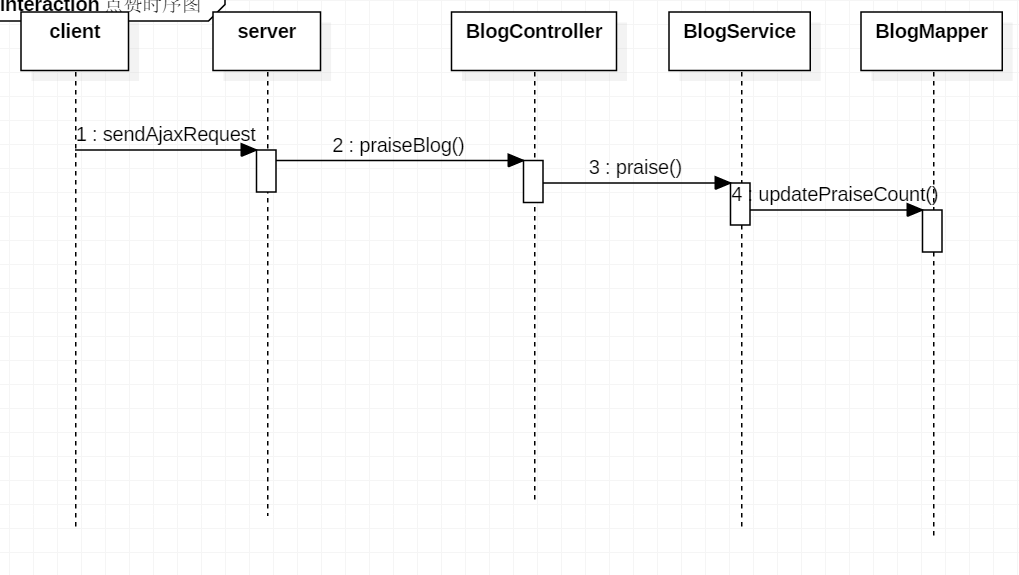


图5.12 点赞博客时序图

##### 4.删除博客

具体实现：用户登录后进入博客管理页面，选中欲删除的博客，点击后面的删除链接即可删除该博客。同时使用Ajax将博客id发送到后台，根据博客id将该博客的state置为1，意为该博客已删除。

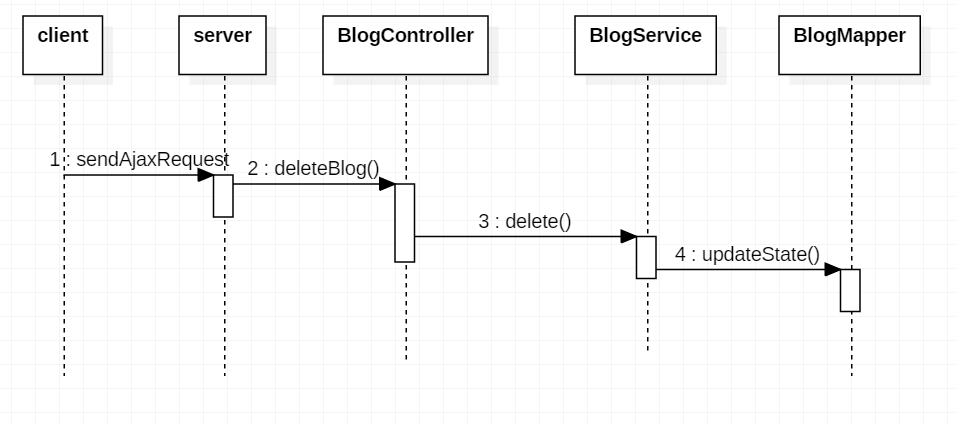


图5.13 删除博客时序图

##### 5.编辑博客

具体实现：用户登录后进入博客管理页面，点击管理，即可查看已有博客。选中欲编辑的博客，点击其后的“编辑”，即可进入博客编辑页面编辑博客，编辑完成后，用户可以将博客保存为草稿，也可以立即发布。若博客标题或内容为空，则阻止提交，并提示用户“标题不能为空”或者“博客内容不能为空”。用户提交数据后，使用ajax将数据发送至服务器端，服务器端获取数据后，从session中取出用户id，根据用户id及博客id更新数据库。

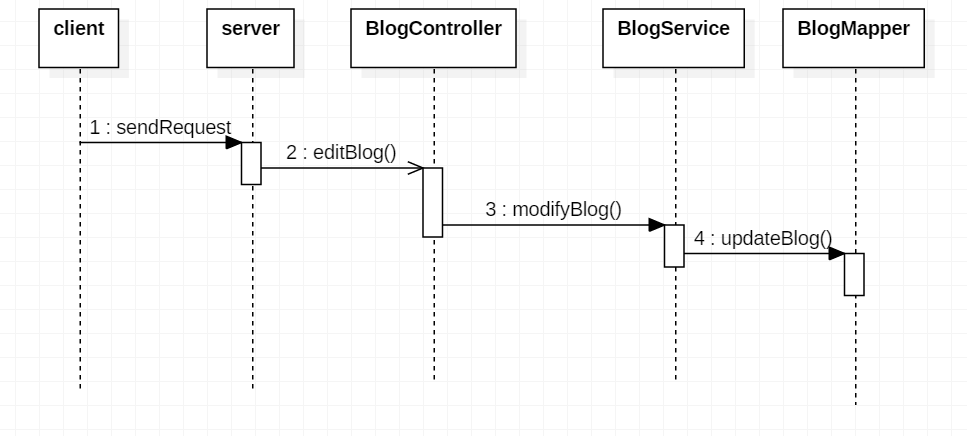


图5. 编辑博客时序图

##### 6.收藏博客

具体实现：用户进入博客页面后，点击页面下的收藏，会弹出收藏按钮，用户输入标题﹑标签﹑描述，点击下方的收藏按钮，即可收藏该博客。用户点击收藏后，首先通过collectionedBlog\_userId列表判断其是否曾收藏过该博客：若已经收藏过，则提示用户；否则使用Ajax将数据（博客id﹑标题﹑标签﹑描述）发送至后台。后台从session中获取用户id，与前台发来的数据一起构建收藏对象，并将记录存入数据库，最后将该博客id存入collectionedBlog\_userId列表,同时将其传回前台，更新前台数据。

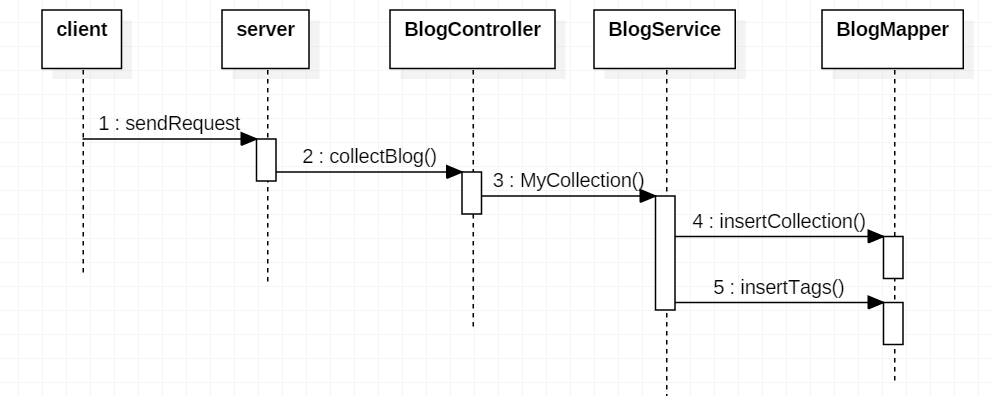


图5.15 收藏博客时序图

#### 5.1.3 博问管理模块

##### 1.提问

具体实现：用户进入提问页面编辑问题。若问题标题或内容为空，则阻止提交，并提示用户“标题不能为空”或者“内容不能为空”。用户提交数据后，使用ajax将数据发送至服务器端，服务器端获取数据后，从session中取出用户id，将问题内容及与之关联的用户id存储至数据库。

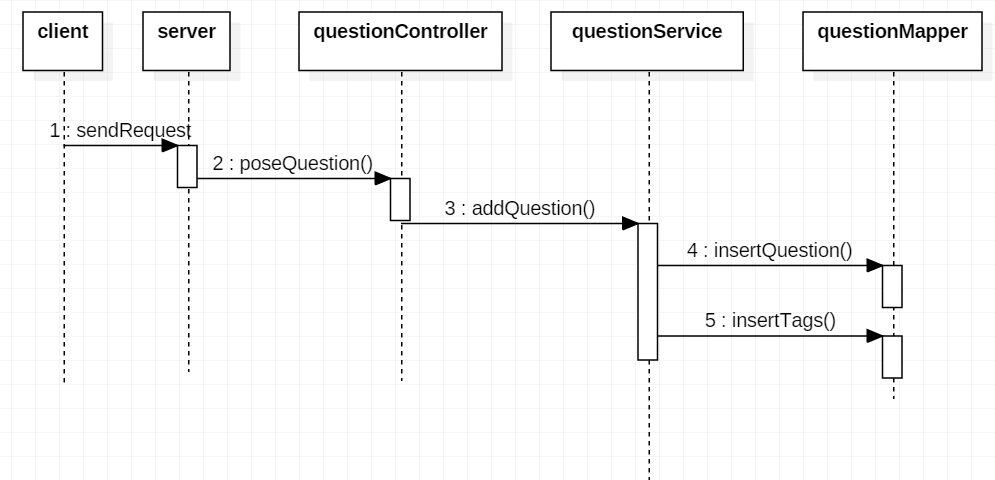


图5.16 发表博问时序图

##### 2.评论

具体实现：用户进入博问详情页面即可评论。用户输入评论内容，点击提交即可。评论成功后，页面上即显示评论内容。同时使用Ajax将评论内容及博问id发送到后台。后台获取评论内容﹑时间﹑用户id及博问id后将其保存至数据库，同时将相应博问的评论数+1。

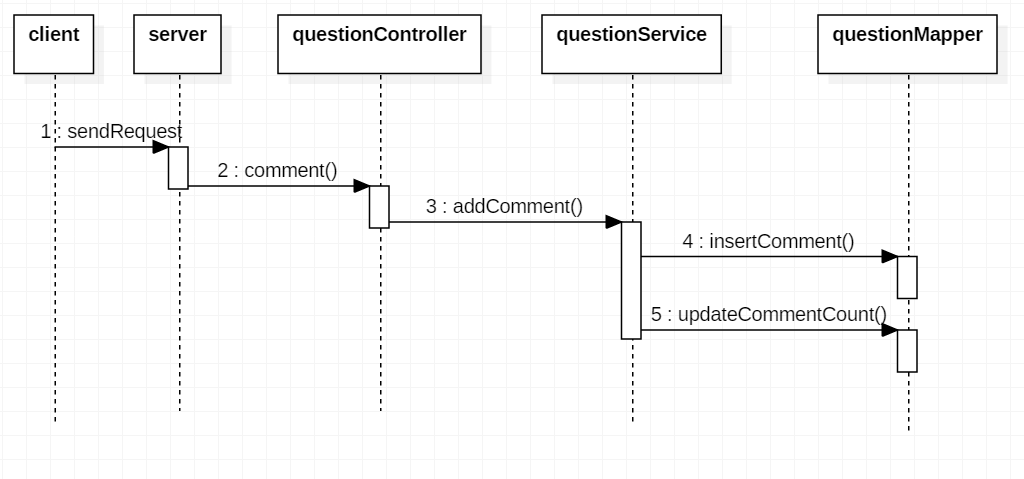


图5.17 评论博问时序图

#### 5.1.4 资源模块

##### 1.上传资源

具体实现：用户在首页点击资源即可进入资源管理页面，点击上传资源可进入资源下载页面。用户上传资源后，输入资源名称﹑选择所属分类﹑资源类型，并添加标签和描述，即可提交，完成资源上传。数据通过Ajax发送到后台，后台使用前台发来的数据构建资源对象，并保存至数据库，同时保存资源的标签至数据库。

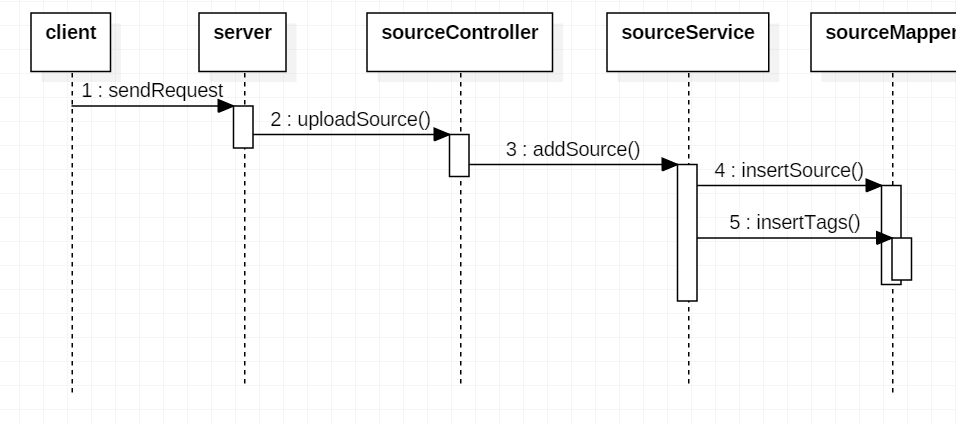


图5.18 上传资源时序图

##### 2.下载资源

具体实现：用户进入资源管理页面后，可以根据分类搜寻资源，也可以输入关键字搜索资源。点击欲下载的资源，即可进入下载。下载完成后，将资源id发送到后台，根据id将该资源的下载数+1。

# 6 测试

# 7用户操作手册

## 7.1

# 总 结

# 参考文献

[1] 温谦编著. CSS网页设计标准教程[M]. 北京：人民邮电出版社，2009.

[2]

[3]

[4]

# 附录