**躬行网-在线答疑管理子系统的设计与实现**

|  |  |  |  |
| --- | --- | --- | --- |
| 学 院 | 计算机学院 | | |
| 专 业：  姓 名：  指导老师： | 软件工程 | | |
| 梁文柱 | 学 号：  职 称： | 130202061019 |
| 旷晓斌 | 讲师 |

中国·珠海

二○一七年五月

**诚信承诺书**

**本人郑重承诺：**本人郑重承诺呈交的毕业论文《躬行网-在线答疑管理子系统》是在指导教师旷晓斌老师的指导下，独立调研，独立开发的成果。文中如有引用他人观点，均会给出引用人或引用文献来源。

本人签名：

日期： 年 月 日

**躬行网-在线答疑管理子系统的设计与实现**

**摘 要**

直至2017年，随着互联网功能和网络信息的急剧变化，我国步入了一个崭新的互联网时代。在这个时代下，网络已经成为人们获取知识的一个异常重要的途径，于是，躬行网应运而生。

本系统是躬行网的子系统-在线答疑管理子系统。其核心功能是为在躬行网用户搭建一个形态为论坛（BBS）的在线答疑平台，目的是提供一个平台，让广大用户在躬行网学习的同时，可以更为主动的交流自己的观点，提出自己的疑问，让知识更频繁地互动，使学习更有效率。

在项目开发技术栈方面，躬行网-在线答疑管理子系统采用了最新的前端框架VUE.2.0，搭配PHP+Mysql，在MVVM的数据双向绑定的框架模式下进行敏捷迭代开发。系统的每一个功能都是一个独立模块，在最大程度上降低了各个功能相互之间的耦合度。

本文的阐述顺序将根据项目的开发顺序，项目立项、需求分析、环境搭建、代码编写、单元测试、系统测试进行阐述编写。

**关键词：** 在线答疑 BBS Vue2.0 MVVM

**Design and Implementation of Online Q & A Management Subsystem**

**Abstract**

Until 2017, with the rapid changes in Internet functions and network information, China has entered a new era of the Internet. In this era, the network has become a very important way to get people to obtain knowledge, so, bang line came into being.

The system is a sub-system - online Q & A subsystem. Its core function is to build a form for the forum (BBS) online Q & A platform, the purpose is to provide a platform for the majority of users in the study line at the same time, you can more actively exchange their views, put forward their own Of the question, so that knowledge more frequent interaction, so that learning more efficient.

In the project development technology stack, the online Q & A management subsystem uses the latest front-end framework VUE.2.0, with PHP + Mysql, in the MVVM data two-way binding framework for agile iterative development. Each function of the system is a separate module, to the greatest extent reduce the coupling between the various functions.

The order of this paper will be based on the project development order, project project, demand analysis, environmental construction, code preparation, unit testing, system testing to elaborate.

**Keywords:** Online Q & A BBS Vue2.0 MVVM

**目 录**

[1前言 1](#_Toc25984)

[1.1研究的现状及其意义 1](#_Toc24040)

[1.2研究目的、研究内容和拟解决的关键问题 1](#_Toc13083)

[1.2.1研究目的 1](#_Toc32366)

[1.2.2研究内容 1](#_Toc26911)

[1.2.3拟解决的关键问题 2](#_Toc31421)

[2开发平台、环境和工具以及相关技术支持 2](#_Toc27072)

[2.1开发环境 2](#_Toc9811)

[2.2技术栈选择 3](#_Toc13364)

[2.3架构模式 3](#_Toc24185)

[3需求分析 3](#_Toc18401)

[3.1系统参与者 3](#_Toc26419)

[3.1.1游客 3](#_Toc31245)

[3.1.2注册用户 3](#_Toc9404)

[3.1.3版主 3](#_Toc6637)

[3.1.4专家 4](#_Toc15821)

[3.1.5后台管理员 4](#_Toc24134)

[3.2总体需求分析 4](#_Toc21828)

[3.3详细需求分析 5](#_Toc12540)

[3.3.1需求用例 5](#_Toc15850)

[3.3.2业务流程需求设计 10](#_Toc20554)

[3.4非功能需求 16](#_Toc24929)

[4概要设计 16](#_Toc14958)

[4.1系统功能 16](#_Toc7038)

[4.2系统架构分析 17](#_Toc16406)

[4.2.1概要分析 17](#_Toc5544)

[4.2.2体系架构分析 17](#_Toc15055)

[4.2.2.1基于Vue2.0的MVVM架构开发环境搭建 18](#_Toc18927)

[4.2.2.2基于Vue2.0的MVVM架构开发过程 18](#_Toc4160)

[4.3数据库设计 23](#_Toc21484)

[4.3.1基础表设计 23](#_Toc12286)

[4.5开发难点分析 27](#_Toc9653)

[4.5.1 帖子分页展示 27](#_Toc26936)

[4.5.2 模态窗口 29](#_Toc10602)

[4.6界面设计 31](#_Toc7823)

[4.6.1主页 31](#_Toc23587)

[4.6.2各种操作页面 32](#_Toc20943)

[4.6.3子页 34](#_Toc5444)

[5系统测试 39](#_Toc9578)

[5.1测试概要 39](#_Toc7367)

[5.1.1测试用例设计 39](#_Toc15986)

[5.1.2测试环境配置 39](#_Toc31006)

[5.1.3测试方法 39](#_Toc1347)

[5.2主要测试用例 39](#_Toc6857)

[5.3测试结果与缺陷分析 40](#_Toc8456)

[6结论 41](#_Toc7116)

[参考文献 42](#_Toc7469)

[致 谢 43](#_Toc13679)

[附 录 44](#_Toc804)

# 1前言

## 1.1研究的现状及其意义

随着崭新的互联网阶段到来，越来越多的网民借助网络平台传递教学资源，对知识进行收集整合。如今，人们对网上教学的理解已经悄然发生变化，不再是传统地，被动地接受知识，而是变得更为主动。在学习的同时，他们还强调知识的交流和观点的碰撞。

躬行网-在线答疑管理子系统为这种知识的交流和观点的碰撞提供了一个平台，在以BBS论坛为形式的平台下，知识和观点变得更为直观，这将在很大程度上提高人们工作学习效率。本系统还将使躬行网用户更为紧密的联系起来，留住躬行网用户。

同时，本系统还开发了很多内容控制模块，如举报系统，管理员系统，允许用户对内容进行举报或者选举管理员，对不良信息进行清除。保证了论坛的秩序以及内容的学术纯净性。

## 1.2研究目的、研究内容和拟解决的关键问题

### 1.2.1研究目的

研究目的主要是研究在线答疑管理子系统对躬行网的意义以及开发框架模式的思考。.

躬行网是一个整合了很多教学资源如音频，视频等网站，而在线答疑管理子系统可以让躬行网用户在学习之余，有一个交流观点，解决疑问的平台。在这个平台上，每个用户都有发声以及解答的权利。同时，可以让用户流量留在躬行网内，这样，就会形成一个良性的循环。

在模式框架的选择上，我选择了时下最新颖的模式，MVVM。以JavaScript来驱动整个项目，所以在线答疑管理子系统是一个单页面（SPA）项目。同时，为了能灵活地应对需求变更等突发状况，本系统采用前端后端完全分离的办法。前端通过AJAX对后端发起请求，后台接收请求后，根据需求返回相应的数据，交由前端进行处理。这样，大大降低了前后端的耦合度，开发起来更方便，逻辑显的更清晰。

技术栈：Vue2.0、Jquery2.0、PHP、MySql

### 1.2.2研究内容

1. 注册登录系统：登录注册功能，用于躬行网的用户注册以及登录。
2. 发帖系统：在线提问功能，实现基本的论坛发帖功能。
3. 回复系统：动态生成回帖，实现基本的论坛回帖功能。
4. 控制系统：对内容进行控制处理，如顶置，删除，举报。
5. 个性化系统：允许用户进行个性化设置，如头像，个性签名。
6. 积分系统：用户激励系统，可以用于兑换技术文档。
7. 文档商城系统：该系统放置有许多技术文档，文档需要积分才能兑换，用于激励用户。
8. 搜索系统：根据关键字对帖子进行模糊搜索。
9. 用户权限：对不同类型的用户进行权限控制。
10. 数据表的设计以及字段设计。
11. 项目开发方向，开发流程，测试方法，集成方法。

### 1.2.3拟解决的关键问题

问题主要分两个方向：业务逻辑、技术实现：

（一）在业务逻辑上：

1. 界面前端设计，力求界面清晰，舒服同时符合广大用户使用习惯。
2. 操作错误时要给出相对应的提示，加强用户体验。
3. 用户权限的设定。
4. 如何防止不良信息大量充斥版面。

（二）在技术实现上：

1. 技术栈的选择。
2. 开发环境的选择。

# 2开发平台、环境和工具以及相关技术支持

## 2.1开发环境

操作系统:Windows8；

IDE：WebStorm 2016.3；

服务器：Apache本地服务器； Node.Js本地服务器。

浏览器：谷歌浏览器（Google Chrome）。

数据库：MySql。

项目构建工具：Webpack。

版本控制工具：Git。

## 2.2技术栈选择

前端：Vue2.0 + Vuex + Vue-router + Jquery2.0。

后端：PHP5.5 + MySql。

## 2.3架构模式

本系统采用B/S架构，在MVVM框架思想下进行开发 。

# 3需求分析

## 3.1系统参与者

本系统的参与者包括游客、注册用户、版主、专家、后台管理员。

### 3.1.1游客

游客参与本系统的功能有：

1. 注册功能：注册成为躬行网会员。
2. 帖子浏览：查阅别的用户发布的帖子。

### 3.1.2注册用户

注册用户是本系统的主要参与者。

1. 登录：根据已注册的账号密码信息进行登录。
2. 个性化：根据个人爱好，设置个性头像和个性签名。
3. 帖子浏览：查阅别的用户发布的帖子。
4. 发帖：根据兴趣发布自己的问题，让别的用户查阅答疑。
5. 回帖：对自己感兴趣的帖子进行回复。
6. 举报：对非法内容进行举报。
7. 赞：对自己赞赏的跟帖内容进行点赞。
8. 积分兑换：对系统所发的积分可以在商城系统内兑换技术文档。
9. 注册功能：注册成为躬行网会员。
10. 帖子浏览：查阅别的用户发布的帖子。

### 3.1.3版主

版主是由普通用提出户申请后，再由后台管理人员选出。版主是对用户端内容进行管理的核心人员。它负责大部分的用户端版面秩序和内容管理工作。每个版主有其对应负责的板块。由于版主是由普通用户权限升级而来的系统参与者，因此其拥有所有注册用户所拥有的功能。同时还拥有以下功能。

1. 顶置帖子：对内容好的帖子或者版面公告进行顶置。
2. 取消顶置: 对已顶置的帖子或者过期公告进行取消顶置。
3. 删除: 对有不良内容的帖子或者跟帖进行删除操作。

### 3.1.4专家

专家是由普通用提出户申请后，再由后台管理人员确认而来。专家是特殊的用户，他们的资料将被展示在专家问答页。由于专家是由普通用户权限升级而来的系统参与者，因此其拥有所有注册用户所拥有的功能。同时还拥有以下功能。

1. 开设社区问答互动：开设社区问答互动贴，申请将被提交到后台进行处理，然后显示在专家问答页。专家开设的社区问答互动贴的专业性更强。因此版面资源更多。

### 3.1.5后台管理员

后台管理员主要在后台OPS端进行日常运维。对整个系统进行维护管理。

1. 用户管理：查询所有用户信息，对用户进行诸如封禁等操作。
2. 帖子管理：查询所有帖子信息，对帖子进行诸如顶置、删除等常规操作。
3. 版主管理：查询所有版主申请信息，并考虑是否给予版主权限。
4. 举报管理：查询用户举报的帖子和跟帖信息，对帖子或者发帖人回帖人进行相关常规操作。
5. 文章管理：对发布的文章进行常规管理，或者发布新的文章。发布的文章将在用户端滚动展示。
6. 资料管理：上传如PDF格式技术文档，文档将被展示在用户端的商城页面，供用户用积分进行兑换下载。可以对商城已有的技术文档进行诸如顶置，删除等常规操作。
7. 参数设置：对用户端的各种参数进行设置，如每页显示帖子数，每列显示的页数等等。

## 3.2总体需求分析

躬行网-在线答疑管理子系统主要研究的内容包括：

1. 前端界面：前端界面要求大方简洁，主题明确，符合用户使用习惯。
2. 各参与者职能：关注不同参与者所能使用的系统的不同功能，以及如何让他们联系起来。
3. 板块划分：不同技术专业对应不同板块，不同板块展示的内容不同。
4. 发帖回帖流程：发帖回帖作为本系统的最基本功能，以及数据来源的最基本入口，其流程必须明确起来，对输入的数据要经过处理，用户输入错误信息时能友好提示错误。
5. 内容控制流程：为了让论坛内容更为高质，必须对论坛内容进行控制。
6. 后台管理系统：系统必须有一个后台来对各种数据进行有效管理。

## 3.3详细需求分析

详细需求分析包需求用例分析以及业务流程需求分析。

### 3.3.1需求用例

用例图可以非常直观地显示系统参与者及其参与的功能之间的关系。

#### 3.3.1.1注册登录用例图

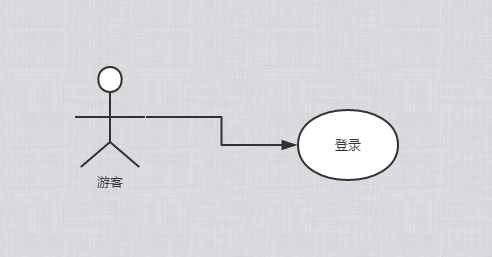


图3.1 注册登录用例图

用例图描述：

游客在不登录的情况下，可以随意浏览帖子，同时可以使用用户注册模块进行注册，成为躬行网的会员。用户注册后，注册信息将被系统录入到数据库的用户表内。在登录时系统将登录信息与用户表进行比对，匹配成功则登录成功，否则则登录失败。

#### 3.3.1.2发帖用例图

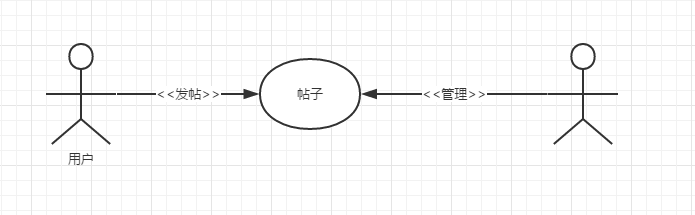


图3.2 发帖用例图

用例图描述：

在登录成功后，用户可以进行发帖操作。

点击页面的发帖按钮后，发帖界面随即弹出。用户输入相应的必填信息后点击发送，数据随即被传到后台进行处理，随后被录入到数据库的帖子表内。同时用户端的帖子列表进行刷新。刚刚发表的帖子即被实时显示在页面当中。

帖子发表后，该帖子的主人以及管理员可以对自己的帖子进行删除操作。

#### 3.3.1.3回帖用例图

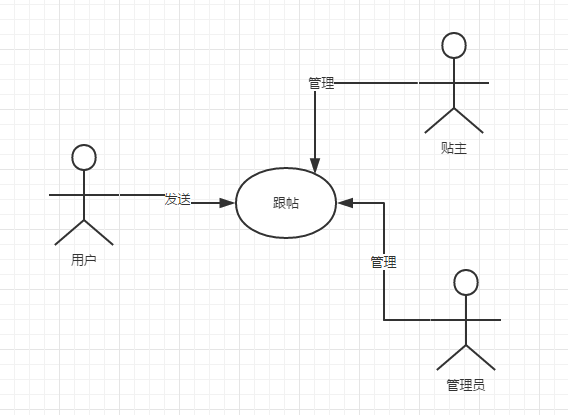


图3.3 回帖用例图

用例图描述：

在登录成功后，用户可以进行回帖操作。

用户浏览帖子时，可以对自己感兴趣的帖子或者感兴趣的楼层进行回帖操作。点击回帖按钮，回帖界面随即弹出。用户输入回帖内容即可点击发送。后台会对传入的数据进行处理后存到数据库的跟帖表中。同时对该楼层的跟帖进行刷新，刚刚发表的跟帖将会在该帖子的最后一楼显示。

跟帖发表后，跟帖发表人，帖子发表人（也就是俗称“贴主”）有删除该跟帖的权限。

#### 3.3.1.4个性化用例图

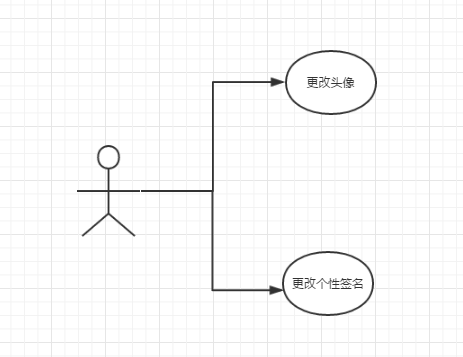


图3.4 个性化用例图

用例图描述：

注册用户登录后，可以对自己的账户进行个性化设置。

目前开放的个性化设置有两个：个性化头像、个性化签名。

用户找到用户详情页（右上角），点击系统默认头像下方的入口“更改头像”，在弹出的文件选择框内选择需要上传的图片，图片信息传给后台处理后，图片文件存到后台服务器文件目录下，图片文件名更新到数据库的用户表内。然后对用户头像进行实时更新。

用户找到用户详情页（右上角），点击一个icon样式为“笔”的按钮，在显示的输入框内输入签名，按下键盘的回车键后，个性签名数据随即上传到后台处理，信息无误后更新到数据库用户表内。

用户的个性化信息将在发帖和回帖时进行展示。

#### 3.3.1.5内容控制（删除、顶置、取消顶置、结贴）用例图

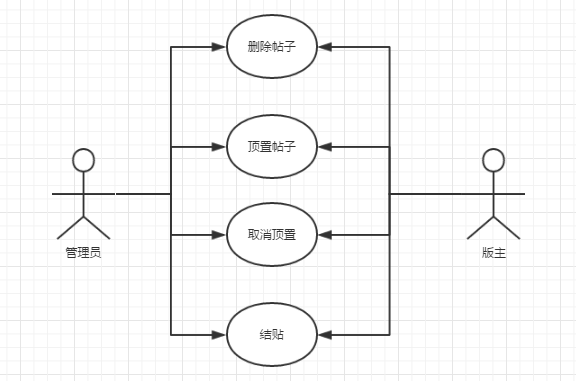


图3.5 处理系统数据维护用例图

用例图描述：

内容控制分帖子内容控制和回帖内容控制，是控制论坛秩序和内容的主要途径。

鼠标移到每个帖子页时，操作入口随即显示，点击相关操作按钮（删除、顶置、取消顶置），后台会将该帖子或者回帖的信息内容传给后台，处理后将数据从数据库的帖子表或者跟帖表中删除。

#### 3.3.1.6举报用例图

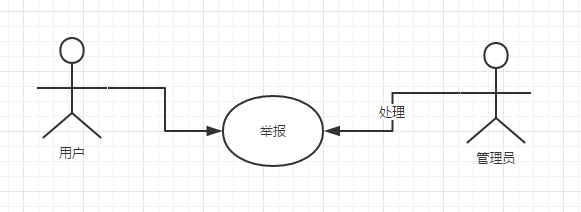


图3.6 举报用例图

用例图描述：

举报功能是每个注册用户都具有的功能，是控制论坛秩序和内容的次要途径。举报对象是帖子，跟帖，以及发表帖子或者跟帖的用户。

在每个帖子内容页的每一层楼的下方都有举报入口，点击后选择举报原因即可完成举报，后台会将该帖子或者回帖的信息内容传给后台，处理后将数据添加到数据库的举报表。后台管理员就可以从管理端查询到举报内容，并进行具体的操作（忽略、删除）。

#### 3.3.1.7专家开启社区问答用例图

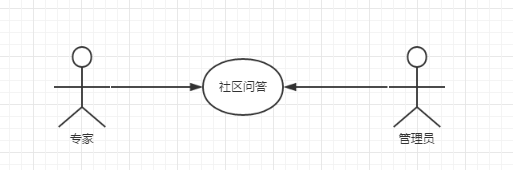


图3.7 专家开启社区问答用例图

用例图描述：

开启社区问答功能是只有专家才有的功能。

点击开启社区问答申请按钮后，输入相关信息，后台将对信息整理并让管理员进行审核。通过后，社区问答将以帖子的形式存在，但由于是专家发起的，其专业性和权威性更高，因此所占有的版面资源更多。

#### 3.3.1.8后台管理用例图

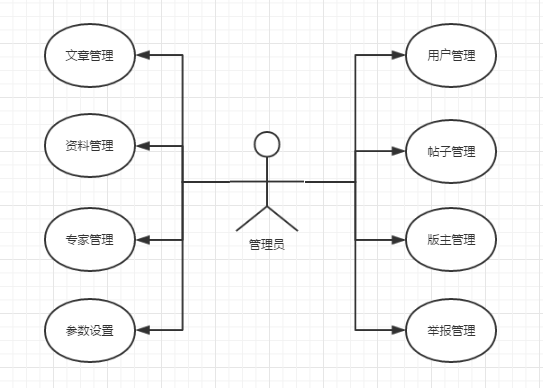


图3.7 专家开启社区问答用例图

用例图描述：

后台管理系统是独立于用户端的后台端，参与者只有后台管理人员。

后台管理项包括：用户管理、帖子管理、版主管理、举报管理、文章管理、资料管理、参数设置。

### 3.3.2业务流程需求设计

#### 3.3.2.1登录

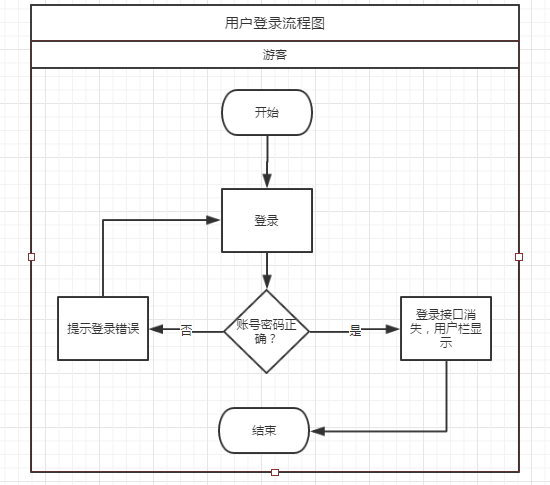


图3.8 用户登录流程图

1. 用户在未登录前的身份是游客，游客可在用户端的右上角进行登录操作。
2. 输入对应的登录信息，系统检测输入的字符是否合法。
3. 如果合法后则传给后端进行用户身份验证，如果存在非法字符则直接提示用户输入正确的用户信息。
4. 后台根据用户输入的登录信息进行验证，验证成功，则登录注册入口消失，用户的操作栏替换登录注册的位置。
5. 如果后台对用户输入的登录验证信息不成功，则提示用户重新输入正确登录信息。
6. 后台将可以对用户信息或者行为进行查询，并对其进行如封禁，发通知等常规操作。

#### 3.3.2.2发帖流程图

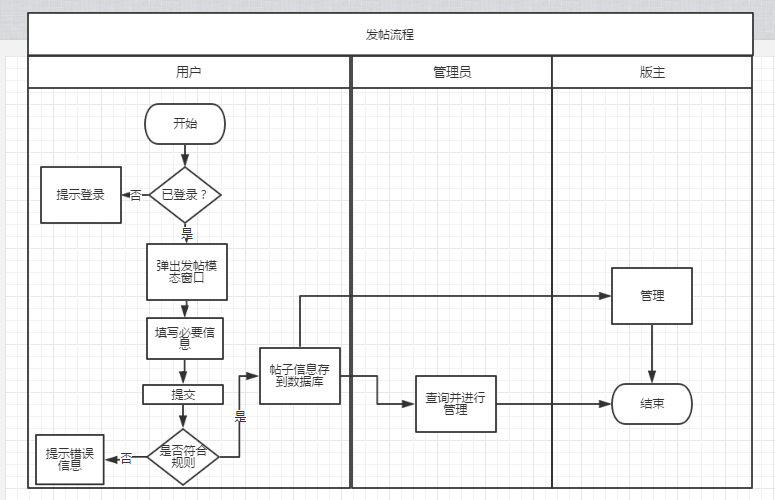


图3.9 发帖流程图

1. 用户找到发帖入口，点击发帖。
2. 系统会判断是否已经登录，如果没有登录，则弹出用户需要先进行登录操作信息。
3. 点击发帖且验证已登录成功后会弹出发帖界面模态窗口，用户在该模态窗口下进行数据录入。
4. 录入途中系统会随时监测输入的数据是否合法，如果非法，则在输入框旁进行提示。
5. 录入完成后点击发表帖子按钮，数据会经过后台处理，并存到数据库的帖子表里。
6. 发帖模态窗会在帖子发表后自动关闭，帖子列表局部刷新，刚刚发的帖子随即显示在列表的最顶端。
7. 后台管理人员可以在后台查询相对应的帖子信息，并进行删除、顶置等常规操作。

#### 3.3.2.3下载资料文档流程设计

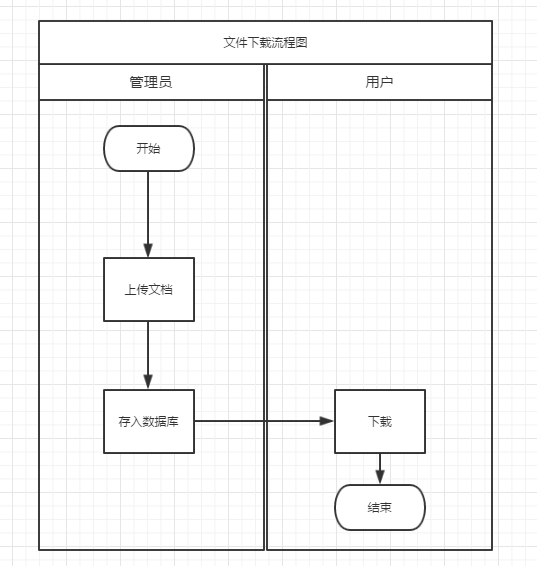


图3.10 下载资料文档流程图

1. 后台管理员上传文档资料，用户端根据后台管理员上传的资料信息将资料展示在用户端的商城页面。
2. 用户可以进入商城，浏览文档资料列表，并对感兴趣的文档资料进行下载操作。
3. 系统会审核是否已登录，未登录则提示用户必须先进行登录操作。
4. 审核通过后，系统会对用户积分余额进行对比，如果需要下载的文档所需积分高于用户剩余积分，则下载失败，提示用户积分余额不足，去跟帖或者发帖进行积分的补充。
5. 若用户积分充值，则文档会直接被下载到用户本地，系统积分将扣除用户的相应积分。

#### 3.3.2.4版主申请流程设计

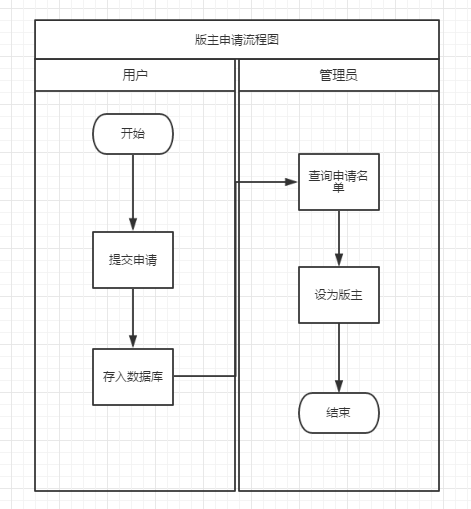


图3.11 版主申请流程设计

每个注册用户都有版主申请的权利，版主对每个特定的板块负责，拥有比普通注册用户更高的管理权限。

1. 在每个特定的板块的上方的板块信息中均有版主申请入口，点击版主申请按钮即可进行版主申请。
2. 点击版主申请时，系统会验证用户是否已登录，若未登录则提示应先进行登录提示。
3. 系统验证登录信息成功后，会弹出用户自我介绍摸态框，用户可在模态框内输入申请理由或者申请优势，系统会对输入的信息进行实时监测，验证是否符合输入规则。
4. 在输入完成且系统验证成功后，用户可以点击“提交申请”按钮来进行版主申请。
5. 后台首先会验证用户是否曾经进行过版主申请操作且申请还在后台审核中，若已申请过且正在审核，则返回相关提示，关闭模态框。
6. 后台验证通过后，用户信息和用户申请理由将被后台储存在数据库的用户申请表内。
7. 后台管理人员可以实时查询当前所有的版主申请条目，并对条目进行设为版主、忽略等常规操作。
8. 后台管理人员操作后，用户端的通知栏将会显示审核信息，提示用户是否申请成功。申请失败后还可以继续进行申请。

#### 3.3.2.5内容控制流程设计

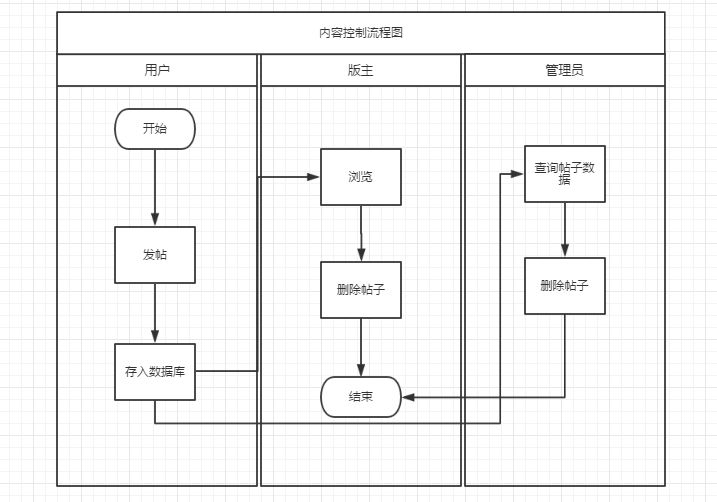


图3.12 内容控制流程（删除帖子）程设计

内容控制是维护论坛版面秩序和版面内容的重要途径，目前开放了四个控制入口，分别是：删除、结贴、顶置、取消顶置。这里以删除为例。

1. 版主或者后台管理员具有删除帖子或者删除跟帖的权限。在每个帖子的内容页的每一层楼均有删除按钮。
2. 点击删除按钮时，系统首先会判断是否已经登录，若未登录，则返回相关操作必须进行登录操作提示。
3. 系统验证登录成功后，会验证用户是否具有删除权限，若没有相关权限，则提示权限不足信息。
4. 系统验证权限成功后，为了防止误操作，会弹出一个摸态框，询问确认删除帖子后者跟帖。
5. 如果是误操作，版主或者管理员可以点击“取消”来对删除操作进行取消，模态框消失。
6. 版主或者管理员点击在弹出的模态框点击“删除”后，帖子或者跟帖将在数据库的帖子表或者跟帖表被删除，后台返回删除成功信号后，用户端实时刷新帖子列表或者跟帖列表，被删除的帖子或者跟帖不再显示。
7. 如果有游客在别的用户端进入了被删除的帖子或者操作了被删除的跟帖，系统会提示帖子或者跟帖被删除的相关信息。

## 3.4非功能需求

1. 用户端的首页渲染在8秒之内。
2. 界面前端设计，力求界面清晰，舒服同时符合广大用户使用习惯。
3. 每个系统功能点都模块化，并与其它模块都仅通过接口进行联系，最大限度地降低耦合度。
4. 用户对某些模块进行操作，如个性化，发帖回帖等，都局部刷新数据而不是全局刷新页面。
5. 浏览器兼容性，兼容包括IE9以上浏览器，IE9以下不予以兼容。

# 4概要设计

## 4.1系统功能

本系统的抽象结构图如下：

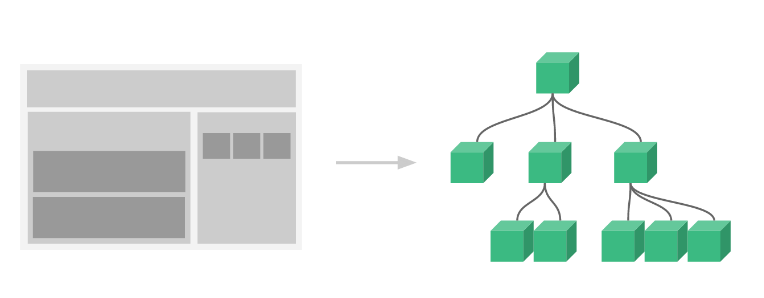


图4.1 系统抽象结构图

## 4.2系统架构分析

一个系统必须有清晰合理的架构对开发进行指导，否则无法有效率地进行开发和维护。为了对系统的前端和后端进行分离，系统架构上使用了MVC的衍生模式MVVM数据双向绑定架构模式。

组件化开发是Vue2.0的MVVM架构开发流程的关键。因为它是一种抽象，允许我们用小型，可复用的组件（图4.1的绿色方块）构建大型应用。通过组件的嵌套组成组件树，从而最终形成一个完整的项目。

### 4.2.1概要分析

MVVM。

VM-ViewModel层：ViewModel层是衍生于是MVC中的C-Controller层，原本是用来负责代码的逻辑，而在MVVM架构模式中其职能变得更为清晰。其代码主要组成部分是JavaScript,并将View层进行数据双向绑定。View层的变动将实时影响ViewModel层的变动。

M-Model层：负责数据的处理，其代码的主要组成部分是PHP，它仅仅负责数据的返回功能。

V-View层：负责前端页面的展示，其代码主要的组成部分是HTML。

### 4.2.2体系架构分析

MVVM图示如下：

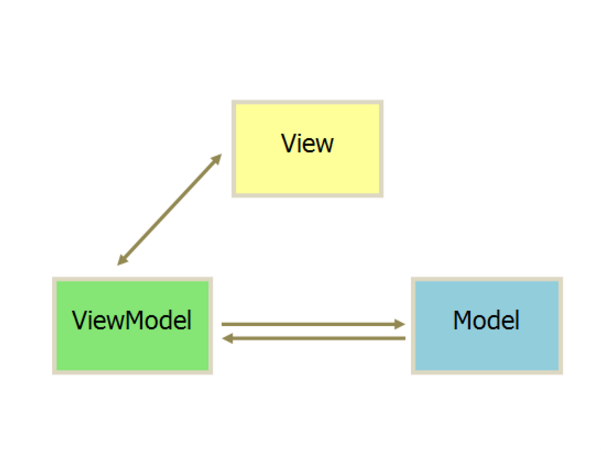


图4.2 MVVM图

### 4.2.2.1基于Vue2.0的MVVM架构开发环境搭建

Vue2.0是基于单文件组件进行模块单独开发的，所以到最后必须使用前端构建工具WebPack对各个模块进行整合。因此开发环境搭建流程如下：

1. 安装Node.js
2. 使用npm install webpack -g 指令全局安装webpack.
3. 使用npm install vue-cli -g 指令全局安装vue-cli 脚手架工具.
4. 使用 cd XXX 命令，进入一个具体的文件目录里面，开始搭建文件目录.
5. 使用 vue init webpack 对开发环境进行初始化，并在命令行内按提示输入项目名，项目作者等相关信息。
6. 此时目录下就会出现新的文件夹，使用cd XX 指令进入到刚刚创建的项目目录里。
7. 使用 npm install less --save 指令局部安装LESS预编译器。
8. 使用npm install vue-router --save 指令局部安装vue-router路由器。
9. 使用npm isntall vuex --save 指令局部安装vuex 数据状态管理器。
10. 在文件的src目录下找到main.js，在Vue () 实例的上方增加Vue.use(Vuex)以及Vue.use(router);两行代码，进行路由器和数据状态管理器的配置。
11. 使用npm run dev 让项目在node.js的本地服务器下运行。

由于有Vue-cli脚手架工具的存在，我现在可以进行模块化开发，每个功能模块都独立起来，模块之间的数据状态传递通过Vuex 的 store.js 文件来实现。这样就可以把功能模块之间的耦合度降到最低。同时由于npm的watch特性，开发文件的变动能即时引起浏览器自动刷新，大大提高了开发效率。同时又由于vue-cli有着良好的错误管理系统，每当代码出现时，它总可以及时提示错误位置和错误原因，这也很大程度上提高了代码调试和开发效率。

### 4.2.2.2基于Vue2.0的MVVM架构开发过程

这里以用户登录窗口弹出为例，我将详细地介绍在MVVM数据双向绑定架构开发过程：

1. 1.首先创建一个名为login.vue的单文件组件，该组件就是登录注册入口。
2. 2.单文件内容分为三个部分，分别是<template></template>、<script></script>以及<style></style>。这三个部分分别负责的是标记语言、逻辑以及样式。
3. 3.在login.vue文件内，<template>标签内的内容如下：

<ul class="login-in">

<li class="zhuce" v-on:click="register">注册</li>

<li class="denglu" v-on:click="login">登录</li>

<li>{{modelShow}}</li>

<li>{{modelActive}}</li>

</ul>

其中，v-on是vue定义的指令形式，而click则是触发事件的方式，这里是click。值则是在组件中<script>标签内定义的方法。整句代码的意思是点击这个li标签，则会触发名为register的方法。

1. 在login.vue文件内，<srcipt>标签内的内容如下：

export default {

data:function(){

return{

}

},

components:{

},

computed:{

modelShow(){

return this.$store.state.modelShow

},

modelActive(){

return this.$store.state.modelActive

},

},

methods:{

login:function(){

this.$store.commit('modelShow',true);

this.$store.commit('modelActive','login');

},

register:function(){

this.$store.commit('modelShow',true);

this.$store.commit('modelActive','register');

}

},

}

* export default是为了让别的组件可以调用该组件，组件之间是可以嵌套使用的。
* data对象是login.vue组件的局部数据，并且可以与<template>内的标签内容进行绑定。
* components对象是login.vue组件的组件声明，声明后可以在<template>调用被声明的组件。
* methods对象是login.vue组件的方法声明，声明后可以在<template>通过v-on:事件进行触发。
* 以登录的login.vue方法为例，点击登录按钮后，会执行两行代码。这两行代码的意思是改变vuex 数据状态管理器文件store.js 里的值。Store.js 的部分代码如下：

const store = new Vuex.Store({

state: {

/\*模态窗口开启\*/

modelShow: false,

/\*具体的模态窗口\*/

modelActive:'none',

}

mutations: {

modelShow(state,ifshow){

state.modelShow = ifshow;

},

modelActive(state,active){

state.modelActive = active;

},

}

})

export default store;

1. 打开另一个已经被我事先创建好的模态窗口Vue组件：modelWindow.vue。它<template>标签里的内容部分如下：

<div class="modal-window" v-show="modelShow">

<div class="modal-windows">

<!--登录-->

<login v-if="modelActive == 'login'"></login>

</div>

</div>

<script>标签的内容部分如下：

import login from '../three/modelLogin.vue';

export default {

components:{

login,

},

computed:{

modelShow(){

return this.$store.state.modelShow

},

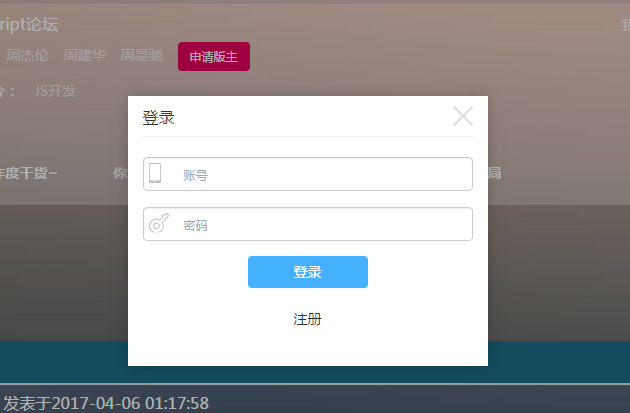
modelActive(){

return this.$store.state.modelActive

},

},

},

* Import 把登录模态窗口组件modelLogin.vue导入到该组件内，并通过components对象内进行声明。因此可以在<template>上面直接使用。
* v-show指令是vue的条件渲染指令之一，如果值为true，则渲染，否则不渲染。
* v-if指令是vue的条件渲染指令之一，如果判断成立，则渲染，否则不渲染。
* 无论是v-show还是v-if，他们的值都是从Vue的计算属性computed对象得到的。
* computed对象一共提供两个值，一个是modelShow(),一个是modelActive(),他们返回的是当前vux的store.js相对应的状态。可以看到，如果modelShow()返回的值为true，最外层的<div class="modal-window"></div>会被渲染，否则则不会被渲染。如果modelActive ()返回的值为'login'，则登录摸态框则会被渲染，否则则不会被渲染。
* 由于我们在login.vue点击了登录按钮。登录方法内，改变了store.js里modelShow和modelActive的值：分别为true和login。
* 最外层的<div class="modal-window"></div>和被导入的组件modelLogin.vue<login v-if="modelActive == 'login'"></login>同时被渲染出来了。效果图如下
* 

4.3 登录模态框效果图

6）虽然这是一个很基础的功能，但开发流程如上。可以看到，每一个功能模块对应一个单文件组件vue文件。每个组件有自己独立的标签、方法、样式。互不侵扰。模块与模块通过vuex 的store.js 进行数据交互联系，从而降低模块之间的耦合度。

可以看到，组件化开发是Vue2.0的MVVM架构开发流程的关键。因为它是一种抽象，允许我们用小型，可复用的组件（如上面的登录模态窗组件）构建大型应用。通过组件的嵌套组成组件树，从而最终形成一个完整的项目。

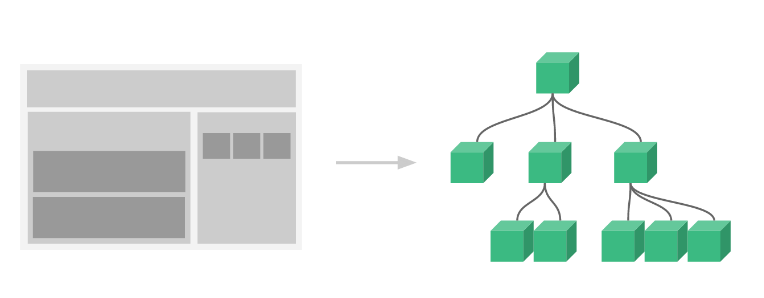


图4.4 系统抽象结构图

### 4.3数据库设计

### 4.3.1基础表设计

以下列出部分基础表设计。

#### 4.3.1 用户表

表4.1 用户表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 类型 | 摘要 | 说明 |
| 1 | userId | Int | 用户Id | 主键，不为空 |
| 2 | userName | varchar | 用户名 | 不为空 |
| 3 | passWord | varchar | 用户密码 | 不为空 |
| 4 | userSign | varchar | 用户签名 | 默认为“编辑自我介绍，让更多人了解你” |
| 5 | userHead | varchar | 用户头像 | 默认为默认头像名 |
| 6 | userScore | varchar | 用户积分 | 默认为200 |
| 7 | userFatieCount | varchar | 用户发帖数 | 默认为0 |
| 8 | userGentieCount | varchar | 用户跟帖数 | 默认为0 |
| 9 | userUpVoteCount | varchar | 用户被赞数 | 默认为0 |
| 10 | userDownVoteCount | varchar | 用户被踩数 | 默认为0 |
| 11 | userCreatedData | varchar | 用户创建日期 | 不为空 |
| 12 | userPowerLevel | varchar | 用户权限等级 | 默认为”0” |

#### 4.3.1.2 通知表

表4.2 通知表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 类型 | 摘要 | 说明 |
| 1 | tongzhiId | Int | 通知Id | 主键，不为空 |
| 2 | tieziId | varchar | 帖子Id | 外键，不为空 |
| 3 | gentieId | varchar | 跟帖Id | 外键，不为空 |
| 4 | userName | varchar | 用户名 | 不为空 |

#### 4.3.1.3 帖子表

表4.3 帖子表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 类型 | 摘要 | 说明 |
| 1 | tieziId | Int | 帖子Id | 主键，不为空 |
| 2 | tieziTitle | varchar | 帖子标题 | 不为空 |
| 3 | tieziContent | varchar | 帖子内容 | 不为空 |
| 4 | tieziTag | varchar | 帖子标签 | 不为空 |
| 5 | tieziSection | varchar | 帖子所属板块 | 不为空 |
| 6 | tieziScore | varchar | 帖子悬赏分数 | 不为空 |
| 7 | tieziCreater | varchar | 帖子创始人 | 不为空 |
| 8 | tieziCreaterData | varchar | 帖子创世时间 | 不为空 |
| 9 | tieziAnswer | varchar | 帖子回答数 | 默认为“0” |
| 10 | tieziLastAnswer | varchar | 帖子最后回答人 | 不为空 |
| 11 | tieziLastAnswerDate | varchar | 帖子最后回答日期 | 不为空 |
| 12 | tieziOverhead | tinyint | 是否顶置 | 默认为“0” |

#### 4.3.1.4 跟帖表

表4.4 跟帖表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 类型 | 摘要 | 说明 |
| 1 | gentieId | Int | 跟帖Id | 主键，不为空 |
| 2 | userName | varchar | 用户名 | 不为空 |
| 3 | tieziId | varchar | 跟帖所属帖子Id | 外键，不为空 |
| 4 | floorNum | varchar | 楼数 | 不为空 |
| 5 | gentieContent | varchar | 跟帖内容 | 索引，不为空 |
| 6 | gentieCreaterData | varchar | 跟帖创建日期 | 不为空 |
| 7 | beigentieFloorNum | varchar | 被跟帖楼层 | 默认为“1” |
| 8 | beigentieContent | varchar | 被跟帖内容 | 不为空 |
| 9 | beigentieUserName | varchar | 被跟帖用户 | 不为空 |
| 10 | dianzanCount | varchar | 跟帖被赞数 | 默认为“0” |
| 11 | caiCount | varchar | 跟帖被踩数 | 默认为“0” |

#### 4.3.1.5 举报表

表4.5 举报表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 类型 | 摘要 | 说明 |
| 1 | jubaoId | Int | 举报Id | 主键，不为空 |
| 2 | jubaoType | varchar | 举报类型 | 不为空 |
| 3 | tieziId | varchar | 被举报帖子Id | 外键，不为空 |
| 4 | gentieId | varchar | 被举报跟帖Id | 外键 |
| 5 | jubaoUserName | varchar | 举报人 | 不为空 |
| 6 | beijubaoUserName | varchar | 被举报人 | 不为空 |
| 7 | jubaoReason | varchar | 举报原因 | 不为空 |
| 8 | jubaoDate | varchar | 举报日期 | 不为空 |

#### 4.3.1.6 点赞表

表4.6 点赞表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 类型 | 摘要 | 说明 |
| 1 | dianzanId | int | 点赞ID | 主键，不为空 |
| 2 | tieziId | varchar | 帖子ID | 外键，不为空 |
| 3 | gentieId | varchar | 跟帖ID | 外键，不为空 |
|  | userName | varchar | 用户名 | 不为空 |
|  | beidianzanUserName | varchar | 被点赞用户名 | 不为空 |
|  | dianzaiDate | varchar | 点赞日期 | 不为空 |

#### 4.3.1.7 通知表

表4.7 通知表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 类型 | 摘要 | 说明 |
| 1 | tongzhiId | Int | 通知Id | 主键，不为空 |
| 2 | tieziId | varchar | 帖子Id | 外键，不为空 |
| 3 | gentieId | varchar | 跟帖Id | 外键，不为空 |
| 4 | userName | varchar | 用户名 | 不为空 |

#### 4.3.1.8 版主申请表

表4.8 版主申请表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 类型 | 摘要 | 说明 |
| 1 | requireId | Int | 申请Id | 主键，不为空 |
| 2 | userName | varchar | 申请人 | 外键，不为空 |
| 3 | advantage | varchar | 申请理由 | 不为空 |
| 4 | requireDate | varchar | 申请时间 | 不为空 |
| 序号 | 字段 | 类型 | 摘要 | 说明 |
| 1 | tongzhiId | Int | 通知Id | 主键，不为空 |
| 2 | tieziId | varchar | 帖子Id | 外键，不为空 |
| 3 | gentieId | varchar | 跟帖Id | 外键，不为空 |
| 4 | userName | varchar | 用户名 | 不为空 |

#### 4.3.1.9 后台管理员表

表4.9 后台管理员表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 类型 | 摘要 | 说明 |
| 1 | userId | Int | 用户Id | 主键，不为空 |
| 2 | userName | varchar | 用户名 | 不为空 |
| 3 | passWord | varchar | 用户密码 | 不为空 |
| 4 | userSign | varchar | 用户签名 | 默认为“编辑自我介绍，让更多人了解你” |
| 5 | userHead | varchar | 用户头像 | 默认为默认头像名 |
| 6 | userScore | varchar | 用户积分 | 默认为200 |
| 7 | userFatieCount | varchar | 用户发帖数 | 默认为”0” |
| 8 | userGentieCount | varchar | 用户跟帖数 | 默认为”0” |
| 9 | userUpVoteCount | varchar | 用户被赞数 | 默认为”0” |
| 10 | userDownVoteCount | varchar | 用户被踩数 | 默认为”0” |
| 11 | userCreatedData | varchar | 用户创建日期 | 不为空 |
| 12 | userPowerLevel | varchar | 用户权限等级 | 默认为”3” |
| 序号 | 字段 | 类型 | 摘要 | 说明 |
| 1 | requireId | Int | 申请Id | 主键，不为空 |
| 2 | userName | varchar | 申请人 | 外键，不为空 |
| 3 | advantage | varchar | 申请理由 | 不为空 |
| 4 | requireDate | varchar | 申请时间 | 不为空 |
| 序号 | 字段 | 类型 | 摘要 | 说明 |
| 1 | tongzhiId | Int | 通知Id | 主键，不为空 |
| 2 | tieziId | varchar | 帖子Id | 外键，不为空 |
| 3 | gentieId | varchar | 跟帖Id | 外键，不为空 |
| 4 | userName | varchar | 用户名 | 不为空 |

#### 4.4 数据字典

表4.10 数据库总表

|  |  |  |
| --- | --- | --- |
| **表名** | **中文描述表名** | **功能说明** |
| Users | 普通用户表 | 存储普通用户角色 |
| Tongzhi | 通知表 | 存储通知信息 |
| Tiezi | 帖子表 | 存储帖子信息 |
| Gentie | 跟帖表 | 存储跟帖信息 |
| Report | 点赞表 | 存储点赞信息 |
| ManagerRequire | 版主申请表 | 存储版主申请信息 |
| Manager | 版主表 | 存储版主信息 |
| ExpertRequire | 专家申请表 | 存储专家申请信息 |
| Expert | 专家表 | 存储专家信息 |
| Jubao | 举报表 | 存储举报信息 |
| BackendUser | 后台管理员表 | 存储后台管理员信息 |

## 4.5开发难点分析

### 4.5.1 帖子分页展示

帖子分页是论坛最基本的功能需求，同时也是一个难点。一个好的分页组件应该有如下功能：

* 可以动态设置每页显示帖子列数。
* 可以动态设置页码显示数。多余的页码均用省略号（...）显示。
* 分页模块与其它模块解耦。
* 页数改变时，帖子内容实时改变。
* 有上一页、下一页、首页、刷新等常规分页按钮。
* 显示帖子总数，总页数。
* 不能一次性加载所有帖子数据以免加载速度过慢。

根据Vue2.0的数据驱动项目思想，解决步骤如下：

1. 先确定PHP查询mySql的方法，以及php文件所需要的参数。TieziFenye.php的核心查询代码如下：

$pagenum = $\_POST[pageNum];

$tiezimaxnum = $\_POST[tieziMaxNum];

$from = ($pagenum-1)\*$tiezimaxnum;

$sql = "select \* from tiezi

order by tieziOverhead desc,tieziId desc

limit $from,$tiezimaxnum";

$result =mysql\_query($sql);

由前端传入两个参数，分别是页码，以及每页显示最大帖子数。在sql语句的末尾加上查询条件limit即可。

1. 分页基本原理梳理。首先需要确定的是，帖子显示组件和分页组件是两个独立组件。两个组件的数据状态交流应该是通过状态管理器vuex来实现。分页组件的页码变化，引起store.js的属性变化，从而引起帖子组件显示变化。
2. 分页组件的构建：（详细代码见附录）。核心思路是改如下：
   1. 在data对象里定义一个名为page的空数组，用以存放页码。
   2. 在钩子函数created内进行ajax查询，得到所有帖子总数。
   3. 帖子总数 / 每页最大帖子显示数 = 页码总数。
   4. 使用FOR循环，构建页码对象，{page：具体页码}，并使用push()方法将所有的页码对象插入到在第一步已经被定义好的空数组内。
   5. 使用v-for="item in pages" 循环渲染页码组件bbs-pagination-li.vue。但这样是会将所有页码都会渲染出来，不符合需求（二），因此还必须在页码组件bbs-pagination-li.vue中增加相应的渲染控制。
   6. 在具体的页码组件bbs-pagination-li.vue 内，增加一个判定方法（具体算法代码见附录）并增加一个条件渲染指令v-if，值等于该判定方法。用以确定此时被被循环渲染的页码是否会被渲染到前端。
   7. 在具体的页码组件bbs-pagination-li.vue内添加点击方法，改变store.js里的属性pageNum属性，也就是此时被点击的页码。

以上就是分页组件的核心功能部分。分别是：页码的渲染、改变store.js的pageNum属性，目的是引起帖子显示组件bbs-table.vue的响应。

1. 构建帖子显示组件bbs-table.vue：核心思路如下：
   1. 在计算属性对象computed内访问store.js的pageNum属性，得到当前页码。
   2. 在计算属性对象computed内访问store.js的tieziMaxNum属性，得到每页显示帖子最大数。
   3. 在data对象里定义一个变量tieziContainObj。
   4. 使用钩子函数watch()，对计算属性computed对象的属性pageNum进行监控，如果pageNum的值变化，则执行分页查询方法。
   5. 分页查询方法核心思路是，使用AJAX对TieziFenye.php进行访问，并传入TieziFenye.php所需要的两个参数：页码，以及每页显示最大帖子数。并将返回的JSON数据赋给在（3）中定义的变量tieziContainObj。
   6. 使用v-for指令，对bbs-Table-Tr.vue组件进行循环渲染。并使用v-bind指令将数据传给子组bbs-Table-Tr.vue以获取具体的数据。

至此，帖子显示组件bbs-table.vue已经完成了他最重要的功能，监听pageNumd的值，并根据变化调用AJAX方法，同时把数据传给子组件bbs-Table-Tr.vue，并在子组件中渲染AJAX返回的数据里的具体的值。

以上就是帖子分页展示的核心思路和实现方法。这种思路的好处有：

* 模块解耦：分页模块和内容模块解耦。
* 数据驱动：分页模块和内容模块通过store.js进行联系，并通过属性pageNum的值来驱动页面的功能实现，这样的编码方式高效，并使得代码的维护和调试变得简单。
* 快速渲染：由于并没有一开始就渲染所有的帖子数据，因此网站的加载速度和整体性能会得到提升。
* 参数可控：由于所有的初始参数，如帖子最大显示页数，页码最多显示数都是存放于store.js里面的，因此我们只需要更改这些参数，就可以轻松应对相关的需求变更。甚至可以直接在管理端里设置这些参数，使用钩子函数created()对数据库进行查询来实时设置这些参数。这样会使整个系统变得非常灵活。
* 在帖子最大显示页数，页码最多显示数的值分别为6和3时，帖子分页展示的效果图如下：



图4.5 分页效果图

### 4.5.2 模态窗口

模态窗口是很多网站的基本功能之一，但也是开发难点之一。一个模态窗口应该有以下特性：

* 不同窗口之间解耦，互不影响。
* 界面简洁大方。
* 最好有出场退场动画，增强用户体验。
* 响应速度要快，不能有卡顿的情况。

根据Vue2.0的数据驱动项目思想，解决步骤如下：

1. 首先确定模态窗口核心思想：使用v-if或者v-show来对元素或者组件进行条件渲染。
2. 为了在模态窗口弹出时，其他功能均不能点击使用，因此设立两层DIV，第一层是position:fixed的遮罩层，大小为浏览器窗口大小，渲染在网站的最外层。这样就可以避免用户在弹出模态窗时还能点击别的按钮。第二层是在第一层的基础上垂直居中显示具体的模态窗口。
3. 创建一个名为modelWindow.vue的组件，这个组件是使用fixed定位的，也就是遮罩层，并这个组件里面根据条件渲染其他具体的模态窗口，也就是第二层。
4. （四）在modelWindow.vue的组件里的计算属性对象computed里获取store.js里的modelShow值，然后再使用v-show来对这个值进行条件判断，若为真。则渲染这个遮罩层组件。但此时并没有渲染具体的模态窗口。
5. 开始创建具体的模态窗口组件，以登录模态窗modelLogin.vue为例。创建完毕后，在modelWindow.vue使用如下声明方式<login v-if="modelActive == 'login'"></login>被调用。并使用v-if进行条件渲染。
6. 在modelWindow.vue下使用计算属性对象得到store.js里的属性modelActive 的值。若modelActive 的值为“login”，则登录模态窗口会被渲染出来。
7. 找到登录组件login.vue，在里面使用v-on指令进行点击事件声明，方法为，将store.js里的属性modelShow的值改为true，属性modelActive 的值改为：login。
8. modelWindow.vue的计算属性computed中的modelShow和modelActive 值因为store.js的值改变而发生了变化，遮罩层的渲染条件和登录模态窗口的渲染条件同时成立，遮罩层和登录模态窗口被同时渲染。

以上八步，是模态窗口的核心思路和实现方法。这种思路的好处有：

* 解耦：各种模态窗口之间解耦。
* 数据驱动：渲染与否都仅取决于store.js里面的数据。
* 首页渲染开销降低：由于使用了v-if的条件渲染进行不同模态窗口的渲染，因此会降低首页渲染开销，因为只有v-if的条件成立时，浏览器才会渲染该组件。
* 提高了开发效率和代码的可维护性。需要添加新的模态窗口时仅需要添加新的模态组件即可。



图4.6 登录模态窗效果图

## 4.6界面设计

躬行网-在线答疑管理子系统的界面参考里诸如百度贴吧、知乎、CSDN等知名论坛或者社区的布局设计。力求大方优美，简介，主题明确。

### 4.6.1主页



图4.7 主页局部图

### 4.6.2各种操作页面

#### 4.6.2.1搜索栏

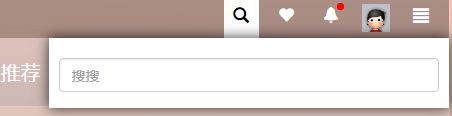


图4.8 搜索栏

#### 4.6.2.2通知栏（无通知情况下）



图4.9 通知栏（无通知下）

#### 4.6.2.3通知栏（有通知情况下）

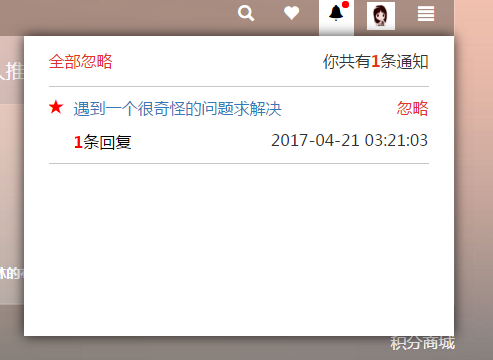


图4.10 通知栏（有通知下）

#### 4.6.2.4用户栏



图4.11 用户栏

#### 4.6.2.5发帖界面



图4.12 发帖界面

### 4.6.3子页

#### 4.6.3.1第一楼界面

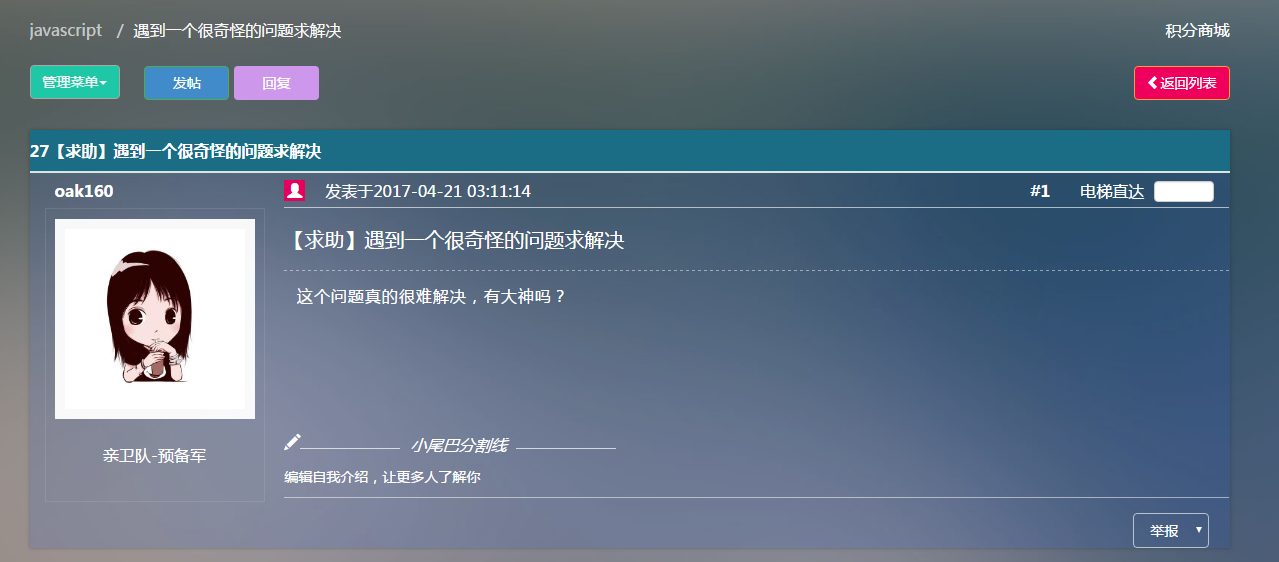


图4.13 第一楼界面局部

#### 4.6.3.2跟帖界面



图4.14 跟帖界面局部

**4.6.4后台管理**

#### 4.6.4.1**后台管理主页**

### 

图4.15 后台管理主页

#### 4.6.4.2**用户管理**

### 

图4.16 用户管理

#### 4.6.4.3**帖子管理**

### 

图4.17 帖子管理

#### 4.6.4.4**版主申请**

### 

图4.17 版主申请

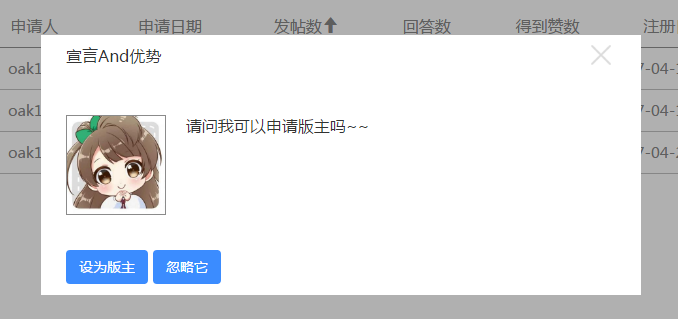


图4.18 版主申请理由窗口

#### 4.6.4.5**举报管理**

### 

图4.19 举报管理

图4.20 用户举报详情



图4.21 跟帖举报详情



图4.22 帖子举报详情

# 5系统测试

## 5.1测试概要

### 5.1.1测试用例设计

本系统的测试使用黑盒测试方法，对每个模块都进行测试，发现BUG修改后再进行回归测试。

### 5.1.2测试环境配置

服务器地址：node.js本地服务器

操作系统：Windows 8

数据库：mysql

测试对象：躬行网-在线答疑管理子系统

浏览器：Google Chrome

### 5.1.3测试方法

测试共分为两轮。第一轮为功能性测试，主要测试每个模块的工作是否正常。第二轮为：易用性测试，主要测试界面是否易于操作，错误提示是否到位等。

## 5.2主要测试用例

表5.1 主要测试用例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试用例 | 前置条件 | 输入 | 预期结果 | 实际结果 |
| 注册 | 无 | 符合规则账号以及符合规则密码 | 注册成功，注册模态框消失，弹出登录模态框 | 与预期结果一致 |
| 登录 | 1. 已注册，拥有账号密码。 | 正确账号以及正确密码 | 登录成功，登录模态框消失。注册登录入口换成用户操作栏。 | 与预期结果一致。 |
| 发帖 | 1、已登录。 | 帖子标题，内容，标签，板块，分数。 | 发帖成功后，显示发帖成功信息，帖子列表实时刷新。 | 与预期结果一致。 |
| 删除帖子 | 1、 版主或管理员账户已登录。 | 点击删除按钮即可。 | 删帖成功，显示删帖成功信息，帖子列表实时刷新 | 与预期结果一致。 |
| 个性化签名修改 | 1. 已登录。 | 输入新签名，并按回车键确认。 | 个性签名修改成功，并在人物详情栏实时显示。 | 与预期结果一致。 |

## 5.3测试结果与缺陷分析

经过本次的测试，我将测试的结果数据作出分析，以下为测试结果与缺陷分析：

图5.1 缺陷分布图

测试结论：

本次测试主要集中在两方面，功能测试和易用性测试。其中功能性缺陷较多，为34个，缺陷主要是由代码编写错误或者算法错误引起的。易用性缺陷较少，为16个。本次测试是伴随着开发的进行而进行的，发现即进行修复。缺陷修复率为95%。

# 6结论

经过2个月的调研、选题、选择技术栈、学习技术，2个月的代码开发，躬行网-在线答疑管理子系统终于按计划完成，并且能在各种测试条件下成功运行。

在技术层面上，躬行网-在线答疑管理子系统采用MVVM架构模式，实现前后端分离。在开发环境上，一共分成三个服务器，其中两个是node本地服务器，另一个是Apache服务器。两个本地服务器分别部署用户端和后台管理端代码。而Apache服务器则部署后台文件PHP，用户端和后台管理端只需要对Apache服务器进行跨域请求数据即可。逻辑处理交给JavaScript，从而实现了前端和后端分离，大大提高了开发的效率以及降低了前后端的耦合度。

同时，由于本系统采用基于Vue2.0的组件化应用构建思想，把整个系统分解成许多抽象的，可复用的组件，再进行开发。这样的好处是能最大限度的对功能模块进行解耦，同时也提高了代码的开发效率和调试效率。

参考文献

[1] 阮一峰：《MVC,MVP和MVVM的图示》。

http://www.ruanyifeng.com/blog/2015/02/mvcmvp\_mvvm.html

[2] 尤雨溪：《Vue.js》。

https://cn.vuejs.org/v2/guide

[3] 陆凌牛：《HTML5与CSS3权威指南》[M]，机械工业出版社 2011.4。

[4] （美）弗兰那根（David Flanagan）.著，淘宝前端团队译：《JavaScript权威指南(第6版)(中文版)》[M]，机械工业出版社2012.3。

致 谢

躬行网-在线答疑管理子系统从2016年11月开始调研选题，到2017年4月开发完成历时五个月，终于可以成功地在这里展现出来。编码的过程是痛苦的，也是快乐的。说它是痛苦的，是因为一开始的技术选型没选好，导致后面把整个系统架构重新推翻重做，做出这一决定是痛苦的，因为这意味着之前很大一部分的工作都将白费。但同时也是快乐的，特别是每解决一个难点，每调试完一个BUG时，都会有难以言喻的自豪感。也许这也是为什么这么多人喜欢当程序员的原因之一吧。

同时在这里，我特别感谢我的论文导师旷晓斌老师,因为我每当遇到困难或者需求不明确的点，他都能点出来，并耐心地向我解释。因此，他由始至终都在我的开发过程中扮演着非常重要的角色。在这里，我由衷地对旷晓斌老师表示感谢！

同时我也要感谢我的实习公司：珠海联创工场，在这里，我学到了很多在学校没有的知识，并将这些知识用在了躬行网-在线答疑管理子系统当中。

由于我的学术水平有限，因此在系统的开发和论文的撰写表达上会有欠缺，因此请老师们见谅。

附 录

附录1

**程序源代码**

分页组件代码：

（1）分页组件：bbs-pagination.vue

<template>

<div class="bbs-pagination">

<ul>

<li v-bind:class="{active : pageClickNum == 1}" v-on:click="firstPageClick" >首页</li>

<li v-bind:class="{active : pageClickNum == 1}" v-on:click="firstPageClick">1</li>

<li v-show="pageClickNum >= pageNumCon">...</li>

<li is="bbsPaginationLi" v-for="item in pages" v-bind:item="item"></li>

<li v-show="pageCount > pageMaxNum">...</li>

<li v-on:click="lastPage">上一页</li>

<li v-on:click="nextPage">下一页</li>

<li class="total\_tieziNum">总数：<span>{{tieziCount}}</span></li>

<li class="total\_tieziPage">共：<span>{{pageCount}}</span> 页</li>

</ul>

<div class="button-group">

{{pageClickNum}}

<button class="btn btn-success post" v-on:click="fatie">发帖</button>

<button class="btn btn-success postReload" v-on:click="firstPageClick">刷新</button>

</div>

</div>

</template>

<script>

import bbsPaginationLi from '../five/bbs-pagination-li.vue'

//import jqForm from '../../../lib/js/jqueryForm.js';

export default {

data:function(){

return{

pageCount:'',

// tieziCount:'',

pages: []

}

},

components:{

bbsPaginationLi,

},

computed:{

tieziCount(){

return this.$store.state.tieziCount;

},

pageMaxNum(){

return this.$store.state.pageMaxNum;

},

pageClickNum(){

return this.$store.state.pageClickNum;

},

pageIgnoreNum:function(){

return this.$store.state.pageIgnoreNum;

},

/\*常量，这个是不变的\*/

pageNumCon(){

return this.$store.state.pageNumCon;

},

tieziStateChange(){

return this.$store.state.tieziStateChange;

}

},

methods:{

fatie:function(){

this.$store.commit('modelShow',true);

this.$store.commit('modelActive','fatie');

},

firstPageClick(){

this.$store.commit({

type:'pageReflash',

pageNum:1,

});

},

lastPage(){

if(this.pageClickNum == 1){

alert("已经是第一页了")

}else{

this.$store.commit({

type:'lastPage',

});

if((this.pageClickNum + 1) % this.pageNumCon==0){

/\*触发规定页码事件\*/

this.$store.commit({

type:'tieziFenyeLast',

pageClickNum:this.pageClickNum,

});

}

}

},

nextPage(){

if(this.pageClickNum >= this.pageCount){

alert("已经最后一页了");

}else{

this.$store.commit({

type:'nextPage',

});

if(this.pageClickNum % this.pageNumCon==0){

/\*点中规定页码事件\*/

this.$store.commit({

type:'tieziFenye',

pageClickNum:this.pageClickNum,

});

}

}

},

},

watch:{

/\*检测帖子状态，改变页码\*/

tieziStateChange:function(){

var that = this;

that.pages.length = 0;

var tieziMaxNum = that.$store.state.tieziMaxNum;

var pageNum = that.$store.state.pageClickNum;

/\*分页对象构建，同时初始帖子总数和页码总数\*/

$.ajax({

url:"php/tieziSelectCount.php",

data:{

tieziMaxNum:tieziMaxNum,

},

type:"post",

dataType:"text",

success:function(data){

var pageNum = Math.ceil(data/tieziMaxNum);

that.pageCount = pageNum;

//that.tieziCount = data;

that.$store.commit('tieziCount',data);

for(var i=2;i < that.pageCount+1;i++){

that.pages.push({page:i})

}

}

});

},

/\*\*\*根据页码查询帖子列表，检测当前点击页码数\*\*\*/

pageClickNum:function(){

var that = this;

var tieziMaxNum = that.$store.state.tieziMaxNum;

var pageNum = that.$store.state.pageClickNum;

/\*分页数据构建\*/

$.ajax({

url:"php/tieziFenyeSelect.php",

data:{

pageNum:pageNum,

tieziMaxNum:tieziMaxNum,

},

type:"post",

dataType:"JSON",

success:function(data){

that.$store.commit('tieziObj',data);

},

error:function(){

alert("失败");

}

});

}

},

created:function(){

this.$store.commit('tieziStateChange');

/\*分页组件被复用了，因此created在这里会执行两次。\*/

this.$store.commit('pageinit');

},

}

</script>

<style lang="less" rel="stylesheet/less" type="text/css" scoped>

@import '../../../lib/css/selfSet.less';

.bbs-pagination{

font-size: 12px;

overflow: hidden;

margin-top:10px;

margin-bottom:10px;

ul{

padding: 0;

li{

cursor: pointer;

padding-left: 12px;

padding-right: 12px;

min-width:25px;

height:25px;

line-height: 25px;

// .border(1px,solid, #dfdfdf);

background-color: #536781;

margin-left: 10px;

float:left;

text-align: center;

color:white;

.transition(0.4s);

&:nth-child(1){

margin: 0px;

}

&:nth-last-child(1){

cursor: default;

background-color: rgba(255, 255, 255, 0);

}

&:nth-last-child(2){

cursor: default;

background-color: rgba(255, 255, 255, 0);

}

&:hover{

background-color: #394759;

.transition(0.4s);

}

}

.active{

color:white;

font-weight: 800;

background-color: #3276b1;

&:hover{

background-color: #285e8e;

}

}

}

.button-group{

float: right;

margin-top:-10px;

.post{

width:65px;

color: white;

background-color: #3276b1;

border: none;

.transition(0.4s);

&:active{

background-color: #285e8e;

outline: none;

}

&:hover{

background-color: #285e8e;

.transition(0.4s);

}

}

.postReload{

width:65px;

//height:25px;

color:white;

background-color: #14937a;

border: none;

.transition(0.4s);

&:hover{

background-color: #398439;

.transition(0.4s);

}

&:active{

background-color: #398439;

outline: none;

}

}

}

}

</style>

（2）页码组件：bbs-pagination-li.vue

<template>

<li v-bind:class="{active : item.page == pageClickNum}"

v-if="ifPageShow=='show'"

v-on:click="pageClick" >

{{item.page}}

</li>

</template>

<script>

//import aaa from '../seven/inform-bottom.vue'

export default {

props:['item'],

data:function(){

return{

}

},

components:{

//aaa,

},

computed:{

pageMaxNum:function(){

return this.$store.state.pageMaxNum

},

pageIgnoreNum:function(){

return this.$store.state.pageIgnoreNum

},

ifPageShow:function(){

if(this.item.page > this.pageMaxNum || this.item.page < this.pageIgnoreNum){

return 'notShow'

}else{

return 'show'

}

},

pageClickNum(){

return this.$store.state.pageClickNum

},

},

methods:{

pageClick:function(){

// if(this.item.page%this.pageMaxNum==0){

if(this.item.page%this.pageMaxNum==0){

/\*点中规定页码事件\*/

this.$store.commit({

type:'tieziFenye',

pageClickNum:this.item.page,

});

}

/\*正常事件\*/

this.$store.commit({

type:'pageClickNum',

pageClickNumForNormal:this.item.page,

});

}

},

// props:['message']

}

</script>

<style lang="less" rel="stylesheet/less" type="text/css" scope>

@import '../../../lib/css/selfSet.less';

</style>

（3）帖子显示表格组件：bbs-table.vue

<template>

<table class="table bbs-main ">

<thead is="bbsThead"></thead>

<tbody class="tiezibody">

<tr is="tbodyTr" v-for="item in tieziContainObj" :key="item.tieziId" v-bind:item="item"></tr>

</tbody>

<thead is="bbsThead"></thead>

</table>

</template>

<script>

import bbsThead from '../five/bbs-Table-thead.vue'

import tbodyTr from '../five/bbs-Table-Tr.vue'

export default {

data:function(){

return{

}

},

components:{

bbsThead,

tbodyTr

},

computed:{

tieziContain(){

return this.$store.state.tieziContain

},

tieziContainObj(){

return this.$store.state.tieziContainObj

}

},

methods:{

},

// props:['message']

}

</script>

<style lang="less" rel="stylesheet/less" type="text/css" scope>

@import '../../../lib/css/selfSet.less';

.bbs-main{

background-color: rgba(0, 0, 0, 0.3);

font-size: 13px;

color:white;

tr{

.transition(0.1s);

&:hover{

background-color: rgba(255, 255, 255, 0.1);

.transition(0.1s);

}

td{

vertical-align: middle;

text-align: center;

&:nth-child(1){

min-width:450px;

text-align: left;

}

&:nth-child(2){

width:49px;

height:51px;

}

&:nth-child(3){

width:160px;

height:51px;

}

&:nth-child(4){

width:65px;

height:51px;

}

&:nth-child(5){

width:160px;

height:51px;

}

&:nth-child(6){

width:51px;

height:51px;

}

border-color:rgba(255, 255, 255, 0.1);

p{

margin-top:10px;

color: rgba(255, 255, 255, 0.8);

a{

color: rgb(255, 255, 255);

}

}

.overhead{

color:#e6005c;

}

.recommend{

color:deeppink;

}

.question-data{

margin: 0;

font-size: 12px;

}

.username{

margin: 0;

font-size: 12px;

}

.control{

position: relative;

cursor: pointer;

color:white;

span{

line-height:38px;

padding-top:20px;

padding-bottom:20px;

}

ul{

width:150px;

height:55px;

background-color: rgba(0, 208, 217, 0.67);

position: absolute;

padding: 0;

margin-left:-145px;

margin-top:-47.5px;

text-align: center;

-webkit-box-shadow:0 0 20px rgba(0, 0, 0, 0.78);

-moz-box-shadow:0 0 20px rgba(0, 0, 0, 0.78);

box-shadow:0 0 20px rgba(0, 0, 0, 0.78);

li{

width:60px;

height:55px;

line-height: 55px;

float:left;

margin-left:00px;

color:white;

&:nth-child(1){

margin-left:15px;

}

&:hover{

}

button{

margin-bottom:5px;

}

}

}

}

}

}

}

</style>

（4）帖子表格头部组件：bbs-Table-thead.vue

<template>

<thead>

<tr>

<th>标题</th>

<th>分数</th>

<th>提问人</th>

<th>回复数</th>

<th>最后更新时间</th>

<th>功能</th>

</tr>

</thead>

</template>

<script>

//import aaa from '../seven/inform-bottom.vue'

export default {

data:function(){

return{

}

},

components:{

//aaa,

},

computed:{

navActive(){

return this.$store.state.navActive

},

},

methods:{

},

// props:['message']

}

</script>

<style lang="less" rel="stylesheet/less" type="text/css" scope>

@import '../../../lib/css/selfSet.less';

thead{

color: white;

background-color: rgba(26, 74, 196, 0.15);

tr{

border-color:rgba(255, 255, 255, 0.1);

th{

text-align: center;

&:nth-child(1){

text-align: left;

}

}

}

}

</style>

（5）帖子tbody的tr组件bbs-Table-Tr.vue

<template>

<tr class="overHeadTr" v-bind:key="item.tieziId">

<td>

<p><span class="overhead" v-show="ifOverHead.tieziOverhead == 1">【置顶】</span><router-link v-bind:to="tieziUrl">{{item.tieziTitle}}</router-link><b class="hidden1">{{item.tieziId}}</b></p>

<td>

<p>{{item.tieziScore}}</p>

</td>

<td>

<p class="username"><a href="#">{{item.tieziCreater}}~</a></p>

<p class="question-data">{{item.tieziCreaterData}}</p>

</td>

<td>

<p>{{item.tieziAnswer}}</p>

</td>

<td>

<p class="username"><a href="#">{{item.tieziLastAnswer}}</a></p>

<p class="question-data">{{item.tieziLastAnswerDate}}</p>

</td>

<td is="optionTd" v-bind:overState="ifOverHead"></td>

</tr>

</template>

<script>

import optionTd from '../six/bbs-table-optionTd.vue'

export default {

props:['item'],

data:function(){

return{

hovering:false,

}

},

components:{

optionTd,

},

computed:{

tieziUrl(){

return "/content/"+ this.item.tieziId

},

ifOverHead:function(){

return {

tieziOverhead :this.item.tieziOverhead,

tieziId : this.item.tieziId

};

}

},

methods:{

mouseMove:function(){

//alert("123");

this.hovering = true;

},

mouseOut:function(){

this.hovering = false;

}

},

// props:['message']

}

</script>

<style lang="less" rel="stylesheet/less" type="text/css" scope>

@import '../../../lib/css/selfSet.less';

</style>