诚信承诺书

本人郑重承诺：本人承诺呈交的毕业设计《基于微信小程序的跨平台文章分享社区的设计与实现》是在指导教师的指导下，独立开展研究取得的结果，文中引用他人的概念和原料，均在文后按次列出其参考文献，设计使用的数据真实可靠。

本人签名：

日期：　　年　月　日

基于微信小程序的跨平台文章分享社区的设计与实现

摘要

本系统是以 PHP 编程语言为核心以及基于微信公众平台的小程序实现跨平台文章分享社区系统。用户不仅可以在 PC 端发表文章和进行相关评论，而且可以使用微信小程序使用相同的功能。在微信小程序端，用户可以无缝进行文章的转发，或者进行小程序码的生成，方便在于朋友圈扩散。用户不仅能进行分享，还可以关注相关用户，进行文章，评论的搜索等功能，补充人们的知识来源途径。系统还配置了相应的管理员后台，方便进行文章，专题，通知管理。除此之外，为了搭建系统的健壮性，系统使用了自动化部署的策略，并且使用了 ELKF 进行域名访问量，总体流量，返回状态码等不同方面的监控，尽可能保证程序的稳定安全可靠运行。整体系统搭建在腾讯云上，充分利用了云服务提供的便捷服务。

关键词：微信小程序 自动化部署 ELKF 腾讯云

Article Sharing Community Based On WeChat Public Platform

Abstract

The system is based on the PHP programming language and a small program based on the WeChat public platform to implement a cross-platform article sharing community system. Users can not only publish articles and make related comments on the PC, but also use WeChat applets to use the same features. In the WeChat applet, users can seamlessly forward articles or generate small program codes, which is convenient for the diffusion of friends. Users can not only share, but also can focus on relevant users, conduct articles, search for reviews and other functions, and supplement people's knowledge sources. The system also configures the corresponding administrator background to facilitate the management of articles, topics, and notifications. In addition, in order to build the robustness of the system, the system uses the automated deployment strategy and uses ELKF to perform monitoring of domain names, total traffic, return status codes, etc. to ensure that the programs are as stable, safe, and reliable as possible. . The overall system is built on Tencent Cloud, making full use of the convenient services provided by cloud services.

Keywords: WeChat Applets　Automated Deployment　ELKF　Tencent Cloud

目录

1 前言

1.1 课题的选题背景及研究意义

1.1.1 选题的背景

在信息时代的蓬勃发展的今日，人们越来越依赖网络进行知识的学习和传播。传统的知识来源已经满足不了人们越来越大的知识渴望需求，此时，基于微信小程序的文章分享社区能够直击痛点，补充人们的知识来源途径。通过认真的考研和分析，微信小程序满足了传播快捷的的需求，人们可以通过小程序的分享功能分享到群或者使用小程序码发布在朋友圈，进行知识传播，也扩大知识的影响力，进行更强烈的思维碰撞，结识更多有趣的小伙伴。

1.1.2 选题的研究意义

正如张小龙先生所说，在 PC 端互联网的入口是在搜索框（例如百度的搜索，谷歌的搜索），然而移动互联网的入口则出现在二维码。借助于微信的流量入口，用户可以方便地通过小程序的分享功能或者小程序码分享自己的独特见识。为了增加平台的趣味性，拥有专题板块，志趣相投的用户可以相聚一起讨论。系统属于跨平台且无需额外安装软件，只需要拥有微信便可立刻获取相应的知识内容。

1.2 课题的构建

1.系统服务架构基于 LNMP (Linux Nginx MySQL PHP)，Redis，ELKF (Elasticsearch Logstash Kibana Filebeta) 监控系统， Redis 缓存， Supervisor 进程管理工具。

2.PC 端和管理后台服务架构基于 Laravel5 框架，使用模板渲染。采用 mvc 的设计模式，将控制层，模型层，视图层进行分层，达到架构解耦的目的。 Laravel5 框架集合了 php7 比较新的特性，以及融入了各种各样的设计模式，依赖注入，Ioc 容器等。前端模块则通过 HTML ， JQuery ， Bootstrap 等工具架构，实现前端页面。利用 Bootstrap 可以提高开发效率，规范 CSS 的名称定义，便于维护，规范项目开发流程，CSS,HTML 代码更清晰，还有相关的栅格布局。

3.小程序后台服务基于 Thinkphp5 框架，接口采用 RESTful API 的设计方法。因为 Thinkphp5 作者是中国人，拥有丰富的中文文档，方便快速查找文档并且拥有强大

的社区。对于中小项目来说，不需要复杂的配置过程，创建骨架很简单，利于进行第三方扩展。

4.小程序基于 WePY 框架，该框架的作者在开发过程中融入了 Vue 等现有前端框架的部分语法风格和功能特性，对原生小程序的开发模式进行了再次优化，用 Vue 的开发思维进行小程序开发。例如：支持组件化开发，可以加载其他的 NPM 包，开发的目录结构清晰，因此开发思路更加清晰，默认使用 babel 编译，微信开发者工具不需要额外开启 es6 转 es5 编译，支持 JS 的部分新特性，针对原生 API 进行优化。

1.3 课题的亮点

1.在搭建 lnamp 环境的时候使用了自动化构建环境脚本，减轻了系统在搭建环境的复杂过程，只需要输入相应的选项即可顺利安装 PHP ， MySQL ， Nginx ， Redis ， Elasticsearch ， Logstash ， Kibana ， Filebeta。

2.在本地对 master 分支进行修改或者将 feature 分支代码合并到 master 分支，推送到远端的 GitHub 仓库之后，便可以坐着喝咖啡等待相应的监控脚本，自动将最新的远端 master 分支代码拉取到服务器进行代码发布和构建。

3.PC 端和管理后台服务架构基于 Laravel5 框架，采用模板渲染页面，还有相应的权限判断。利用了 Elasticsearch 搜索引擎检索相应的查询内容。在查询数据的时候，使用了相关的模型查询。

4.小程序后台服务基于 Thinkphp5 框架，引入自己的第三方类，例如：小程序加解密接口，七牛云上传图片接口， Html 转 Json 的接口等。在查询数据库的时候，采用了相关的模型查询，简化了相关的查询语句书写的复杂程度。

5.小程序基于 WePY 框架，创新地引入 redux 数据状态管理，封装了相关的小程序需要的类，例如：日志上报，原生的获取信息，原生的跳转，原生的网络，原生的提示等类。利用了框架的相关特性，组件化了 Loading 等组件。

6.如上所说，在自动化构建系统的时候搭建了 ELKF 的日志分析系统。不仅可以从海量日志文件中快速查找影响系统的事件，准确定位故障内容并提前预测安全威胁。还可以可视化监控业务数据，为运营人员提供准确的数据服务。

1.4 课题设计应达到的要求

该系统设计由 PHP 为核心开发语言，开源免费的 MySQL 关系型数据库为主导，能够快速存储，搜索，分析海量的数据的 Elasticsearch 企业级搜索引擎为辅助，并采用了敏捷 Web 应用开发的开源 PHP 框架 ThinkPHP5 框架和 Laravel5 框架。在微信小程序上采用了由腾讯大牛开源的类似于 Vue 的开发模式的 WePY 框架。

2 需求分析

2.1 功能清单

系统的实现功能包含以下部分

1.前台功能（PC + 小程序）

1.1 用户登录注册模块：主要负责用户登录，注册功能；

1.2 文章模块：主要负责用户管理文章功能；

1.3 评论模块：主要负责用户管理评论功能:

1.4 赞模块：主要负责用户取消点赞，点赞的功能；

1.5 搜索模块：主要负责管理用户搜索功能；

1.6 个人中心模块：主要负责管理用户个人信息的功能

1.7 专题模块：主要负责管理专题的功能；

1.8 分享模块：主要是小程序二维码分享，群分享的功能；

1.9 吐个槽模块：主要负责用户反馈的功能；

2.管理后台

2.1 后台管理人员模块：主要负责管理后台管理员的功能；

2.2 审核模块：主要负责文章的审核功能；

2.3 权限管理模块：主要负责管理后台权限的功能

2.4 专题管理模块：主要负责管理专题的功能；

2.5 系统通知模块：主要负责通知的发表；

2.2 用例图

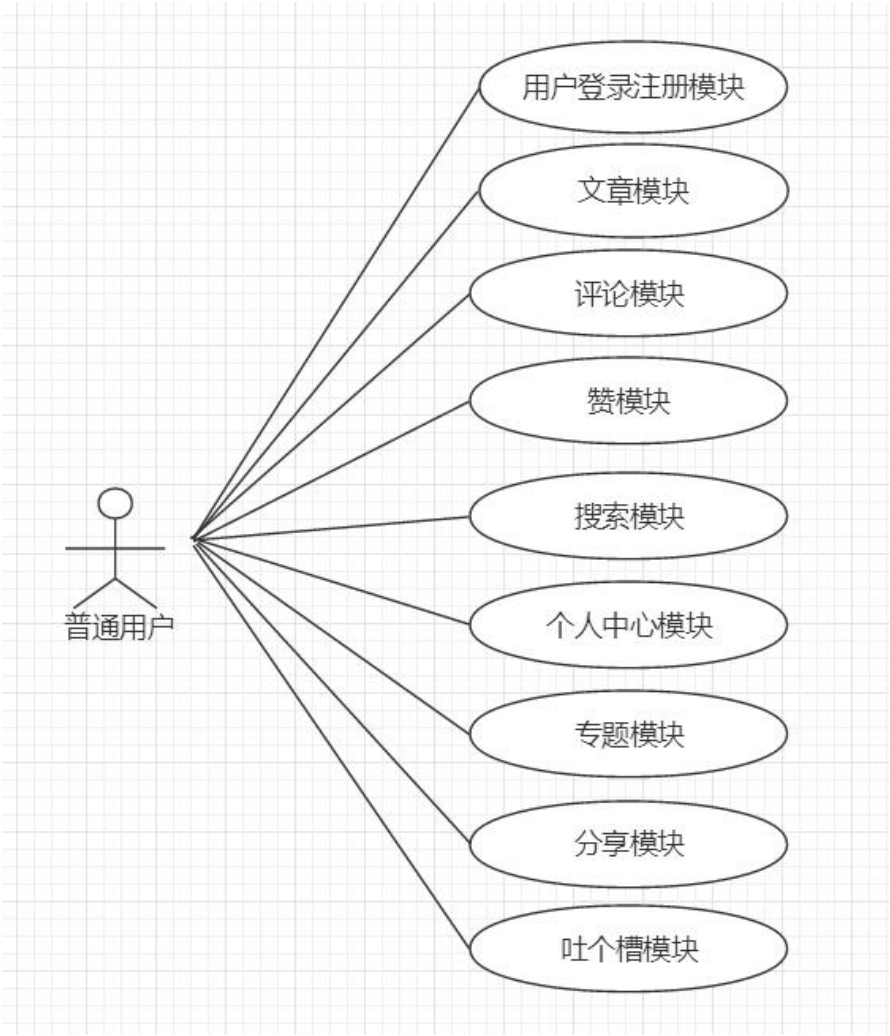

图 2.1 普通用户用例图

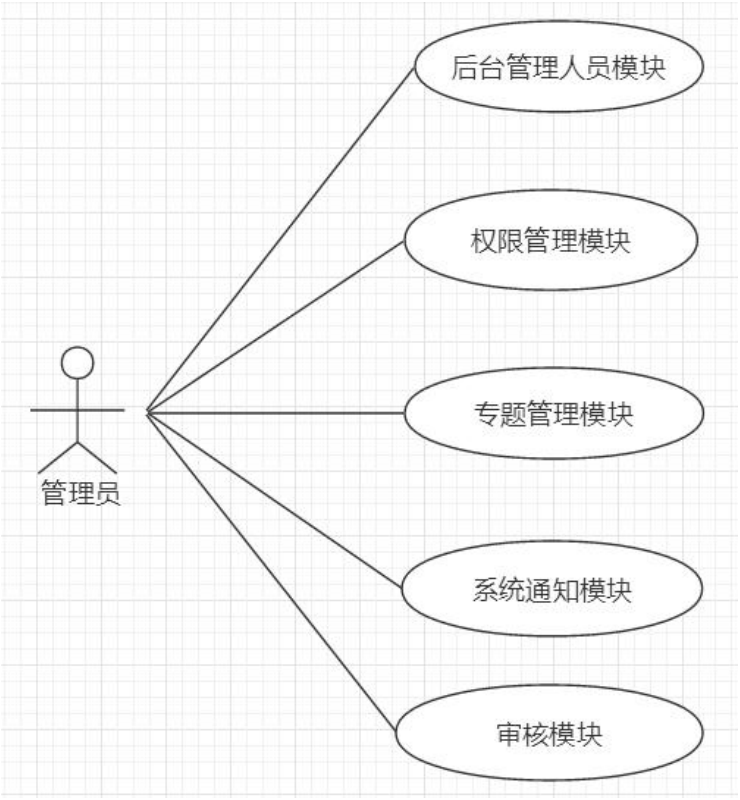

图 2.2 管理员用例图

2.3 用例详述

用例编号	1	用例名称	注册用户
1 描述	用户填写邮箱和密码进行注册		
2 范围	PC 端		
3 主要参与者	普通用户		
4 前置条件	用户已经打开平台的注册界面		
5 后置条件	用户注册成功后可登录		
6 触发事件	用户点击注册按钮		
7 最小保证	系统运行失败，将失败信息写入日志		

8 成功保证	系统保存用户填写的信息
9 事件流	主要场景： 1. 用户进入首页，点击去注册按钮，跳转注册页面 2. 用户填写邮箱和密码，点击保存按钮 3. 系统验证邮箱和密码的合法性 4. 验证合法后，系统保存用户填写的信息 5. 系统提示注册成功，返回平台主页
	替代流程： 3a：系统验证邮箱和密码不合法，则提示用户注册失败，让用户重新注册

用例编号	2	用例名称	登录系统
1 描述	用户填写邮箱和密码进行登录		
2 范围	PC 端		
3 主要参与者	普通用户		
4 前置条件	用户进入首页界面		
5 后置条件	登录成功可进行其他操作		
6 触发事件	用户点击登录按钮		
7 最小保证	系统运行失败，将失败信息写入日志		
8 成功保证	系统保存用户的登录态		
9 事件流	主要场景： 1. 用户进入首页 2. 用户填写邮箱和密码，点击登录按钮 3. 系统验证邮箱和密码的正确性		

	4. 验证正确后，系统提示登录成功，返回帖子列表
	替代流程： 3a：系统验证邮箱和密码不匹配，则提示用户登录失败，让用户重新登录

用例编号	3	用例名称	绑定微信账号
1 描述	用户绑定微信账号		
2 范围	小程序		
3 主要参与者	普通用户		
4 前置条件	用户首次进入小程序		
5 后置条件	绑定成功后进行小程序的其他操作		
6 触发事件	用户点击绑定按钮		
7 最小保证	系统运行失败，将失败信息写入日志		
8 成功保证	系统保存用户填写的信息		
9 事件流	主要场景： 1. 用户首次进入小程序，跳转绑定页面 2. 用户填写邮箱和密码，点击绑定 3. 系统验证邮箱和密码的正确性 4. 验证正确后，系统提示绑定成功，返回首页		
	替代流程： 3a：系统验证密码不正确，则提示用户绑定失败，让用户重新绑定		

用例编号	4	用例名称	解绑微信账号
1 描述	用户解绑微信账号		
2 范围	小程序		
3 主要参与者	普通用户		
4 前置条件	用户进入更多界面		
5 后置条件	解绑成功后失去小程序的功能		
6 触发事件	用户点击解绑按钮		
7 最小保证	系统运行失败，将失败信息写入日志		
8 成功保证	系统保存用户清除的信息		
9 事件流	主要场景： 1. 用户进入更多页面，点击解绑按钮 2. 系统进行解绑 3. 系统提示绑定成功，返回首页		
	替代流程： 2a：系统解绑失败，则提示用户解绑失败，让用户重新解绑		

用例编号	5	用例名称	发表文章
1 描述	用户发表文章		
2 范围	PC 端，小程序		
3 主要参与者	普通用户		
4 前置条件	用户进入发表文章界面		
5 后置条件	无		
6 触发事件	用户点击发表按钮		

7 最小保证	系统运行失败，将失败信息写入日志
8 成功保证	系统保存用户填写的文章内容
9 事件流	主要场景： 1. 用户进入发表文章界面 2. 用户填写文章标题，文章内容，点击发表按钮 3. 系统验证文章内容 4. 系统验证成功后，系统提示发表成功，返回首页
	替代流程： 3a：系统验证失败，则提示用户发表文章失败的原因，让用户重新发表文章

用例编号	6	用例名称	发表评论
1 描述	用户发表文章		
2 范围	PC 端，小程序		
3 主要参与者	普通用户		
4 前置条件	用户进入发表评论界面		
5 后置条件	无		
6 触发事件	用户点击发表按钮		
7 最小保证	系统运行失败，将失败信息写入日志		
8 成功保证	系统保存用户填写的评论内容		
9 事件流	主要场景： 1. 用户进入发表评论界面 2. 用户填写评论内容，点击发表按钮		

	3. 系统验证评论内容 4. 系统验证成功后，系统提示发表成功，返回首页
	替代流程： 3a：系统验证失败，则提示用户发表评论失败的原因，让用户重新发表评论

用例编号	7	用例名称	搜索文章评论
1 描述	用户搜索相关的文章和评论		
2 范围	PC 端，小程序		
3 主要参与者	普通用户		
4 前置条件	用户进入搜索页面		
5 后置条件	无		
6 触发事件	用户点击搜索按钮		
7 最小保证	系统运行失败，将失败信息写入日志		
8 成功保证	系统搜索结果展示		
9 事件流	主要场景： 1. 用户进入搜索界面 2. 用户填写搜索的内容，点击搜索按钮 3. 系统进行相关搜索 4. 系统搜索成功后，系统展示相关内容		
	替代流程： 3a：系统搜索结果为空，则提示用户没有搜索到，让用户缩小搜索范围		

用例编号	8	用例名称	生成文章二维码
1 描述	用户生成文章二维码		
2 范围	小程序		
3 主要参与者	普通用户		
4 前置条件	用户进入文章页面		
5 后置条件	无		
6 触发事件	用户点击转发按钮		
7 最小保证	系统运行失败，将失败信息写入日志		
8 成功保证	系统显示文章的二维码		
9 事件流	主要场景： 1. 用户进入文章界面 2. 用户点击转发按钮，点击生成二维码 3. 系统显示文章的二维码		
	替代流程： 3a. 系统生成文章的二维码失败，则提示生成文章二维码失败的原因，让用户重新生成文章的二维码		

用例编号	9	用例名称	分享指定群聊
1 描述	用户分享文章到指定群聊		
2 范围	小程序		

3 主要参与者	普通用户
4 前置条件	用户进入文章页面
5 后置条件	无
6 触发事件	用户点击转发按钮
7 最小保证	系统运行失败，将失败信息写入日志
8 成功保证	系统转发文章到指定的群聊
9 事件流	主要场景： 1. 用户进入文章界面 2. 用户点击转发按钮，点击转发群，选择指定的群聊 3. 系统转发文章到指定的群聊
	替代流程： 3a. 系统转发文章失败，则提示转发文章失败的原因，让用户重新转发文章

用例编号	10	用例名称	编辑管理员
1 描述	管理员编辑管理员信息		
2 范围	PC 端		
3 主要参与者	管理员		
4 前置条件	用户进入编辑管理员页面		
5 后置条件	无		
6 触发事件	用户点击保存按钮		
7 最小保证	系统运行失败，将失败信息写入日志		
8 成功保证	系统保存管理员填写的新管理员信息		
9 事件流	主要场景： 1. 管理员进入编辑管理员信息界面		

	2. 管理员填写新的管理员信息，点击保存按钮 3. 系统验证管理员信息 4. 系统验证成功后，系统提示编辑管理员信息成功
	替代流程： 3a. 系统验证失败，则提示编辑管理员信息失败的原因，让管理员重新编辑管理员信息

2.4 系统可行性分析

2.4.1 技术可行性分析

依据提出的需求，本系统采用 WebStorm ， PhpStorm 和 微信开发工具 作为系统的开发工具。需要掌握的技术有 PHP 语言, Laravel 框架的模板渲染，权限管理，消息队列, Thinkphp 框架的路由接口,模型查询, 在 PC 端的页面需要掌握 HTML , CSS , JavaScript 技能，小程序开发则需要认真阅读相关的开发文档，ELKF 监控的搭建以及熟练执行 Ubuntu 系统下的相关命令。不仅需要相关技术,相应的设计模式,例如 1.MVC 软件设计典范，将控制层，模型层，视图层进行分层，达到架构解耦的目的。2.OOP 面向对象的思维，三大特性：封装，继承，多态。考虑开发人员的水平，大学四年累计的专业计算机知识，已经从一定程度上具备了开发小型网站和微信小程序的能力，因此课题具备开发的可行性。

2.4.2 经济可行性分析

开发的难度并不大，使用的开发软件也是开源免费的，因此开发成本比较低（整套系统开发需要 10K）。主要费用在于注册小程序的企业账号（一年 300 元），相关的服务器费用（约一个月 300 元）和定期缴纳一定的维护费用（约一个月 200 元）即可。但是一旦流量增大，可以在系统中某写特定位置植入广告等，利用流量资源创收一笔可观的收入。因此能做到前期需要亏损（人工费用，服务器费用和维护费用，暂时性没有广告收入），但是中后期能做到收支平衡，甚至创收的目标。

2.4.3 操作可行性分析

本课题的初衷是为了扩大知识的影响力，进行更强烈的思维碰撞，结识更多有趣的小伙伴。采用当前中国最大的流量入口微信平台，便于在用户群体中确定产品的定位以及获取用户相关的非隐私信息，只需要拥有微信账号即可。对于管理员，能够轻松针对管理员账号，文章，专题，通知的管理。对于普通用户，学习成本更低，只需要简单的几个操作，便能转发到群或者朋友圈。本系统不仅通过合理的 UI 界面，还有强化了用户体验，用户能够快速上手该系统。

2.4.4 法律可行性分析

本课题使用的软件都是开源免费的，遵循其开源协议。例如：MySQL 是开源免费使用的数据库，ELKF 使用其免费版本进行日志分析，本作者已经开源的自动化搭建工具和自动化构建工具等。且本课题在指导老师的指导下独立完成，故不存在侵权行为或者由于本课题带来的法律纠纷等问题。

2.5 角色分析

根据系统业务分析，确定了三大系统角色：“管理员”，“普通用户”。

1. 普通用户

   1.1 登录，注册功能；

   1.2 管理文章功能；

   1.3 管理评论功能:

   1.4 取消点赞，点赞的功能；

   1.5 搜索功能；

   1.6 管理用户信息的功能

   1.7 管理专题的功能；

   1.8 小程序二维码分享，群分享的功能；

   1.9 反馈的功能；

2. 管理员

   2.1 后台管理人员模块：主要负责管理后台管理员的功能；

   2.2 审核模块：主要负责文章的审核功能；

   2.3 权限管理模块：主要负责管理后台权限的功能

   2.4 专题管理模块：主要负责管理专题的功能；

2.5 系统通知模块：主要负责通知的发表；

3 系统开发工具及技术选型

3.1 系统开发环境

本次开发利用 Ubuntu 16.04 作为研发环境，安装 PHP 7.0，Mysql 5.7，使用 PhpStorm IDE WebStorm IDE 小程序开发工具作为开发工具，composer 作为 PHP 的包管理工具，npm 作为前端的包管理工具，以及 Redis 作为系统的消息队列临时存储。监控管理系统由 Elasticsearch ， Logstash ， Kibana ， Filebeta 搭建。

3.2 技术选型

PC 端和管理后台服务架构基于 Laravel5 框架，使用模板渲染。采用 mvc 的设计模式，将控制层，模型层，视图层进行分层，达到架构解耦的目的。前端模块则通过 HTML ，JQuery ， Bootstrap 等工具架构，实现前端页面。小程序后台服务基于 Thinkphp5 框架，接口采用 RESTful API 的设计方法。小程序基于 WePY 框架，该框架的作者在开发过程中利用了 Vue 等现有前端框架的特性，对原生小程序的开发模式进行了再次优化，更贴近于 Vue 的开发模式，让开发者更加容易上手。

4 系统概要设计说明

4.1 系统架构视图

4.1.1 逻辑视图

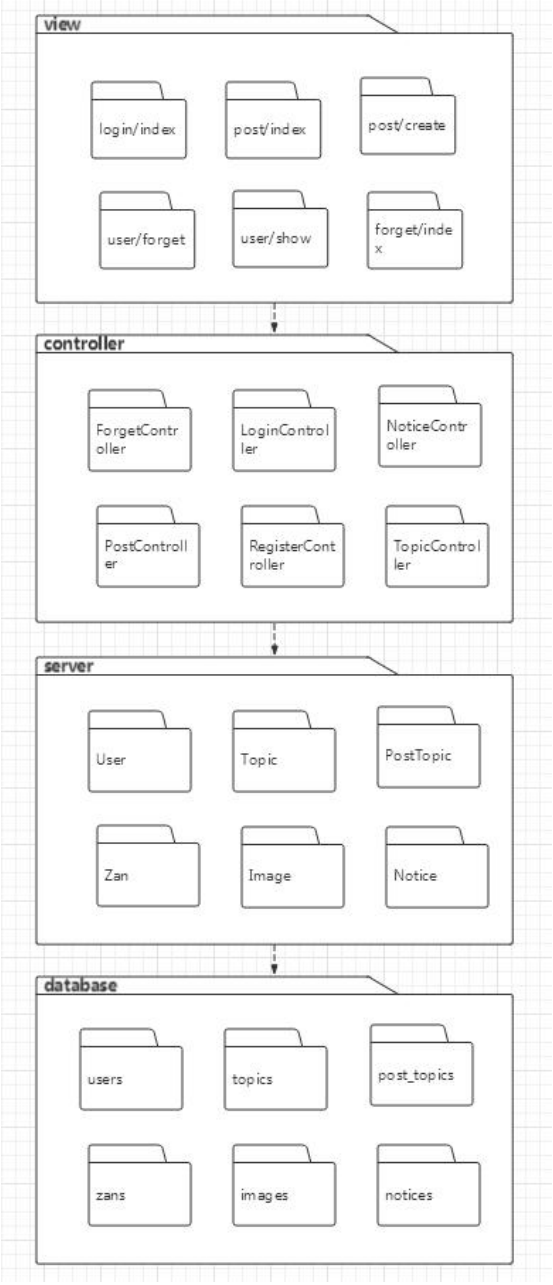

图 4.1 PC 端系统逻辑视图

在 PC 端采用的 MVC 模式，把应用系统划分为三个部分：Model ， View ，

Controller 。其中控制器负责转发请求，针对请求进行处理，视图层尽可能提升用户用户体验，模型层负责实现算法，数据库管理等。

4.1.2 开发视图

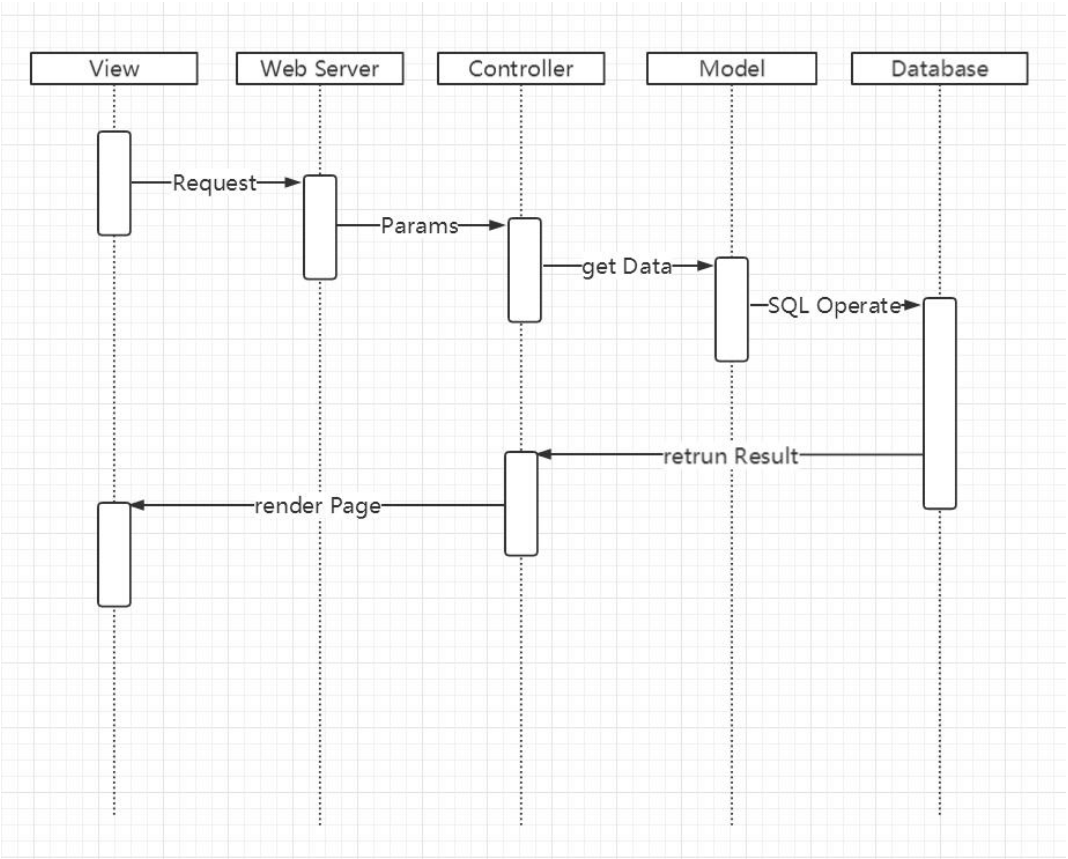

图 4.2 系统开发视图

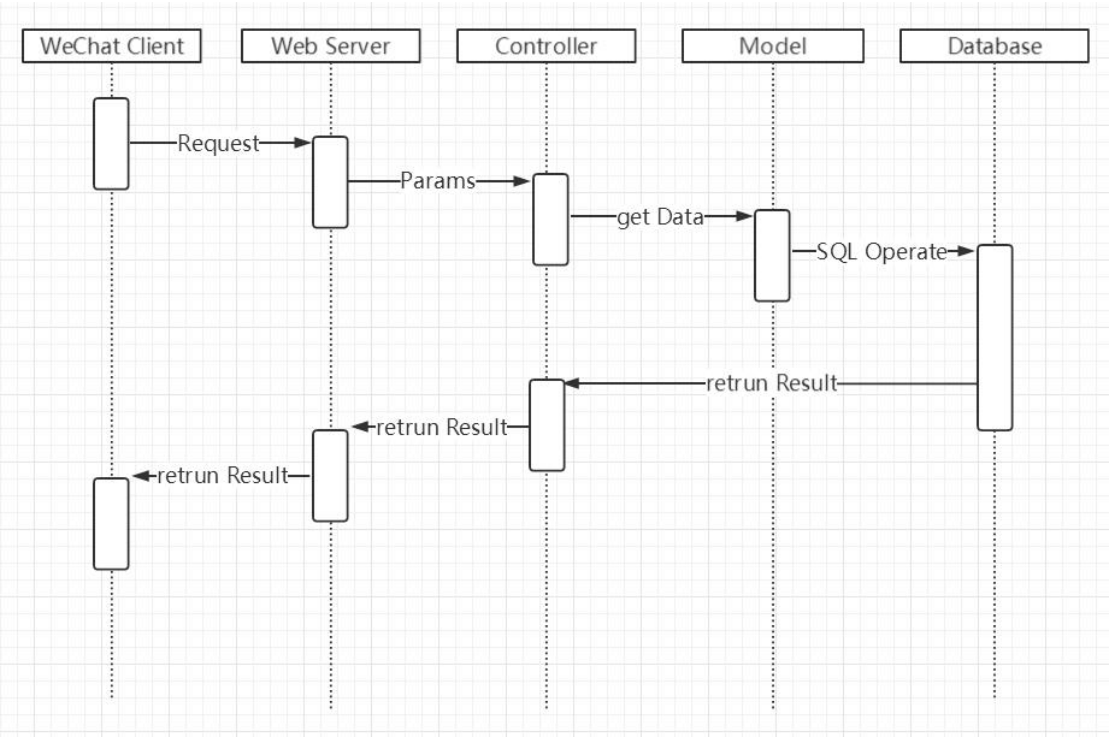

图 4.3 系统开发视图

在 PC 端（用户侧和管理员侧） 和小程序采用不同的开发视图。其中最大不同就是 PC 端采用模板渲染来渲染相关需要展示的数据（后端模板渲染），小程序则采用的是接口方式获取数据然后通过小程序内部的方式将数据展示出来（前端模板渲染）。

4.1.3 部署视图

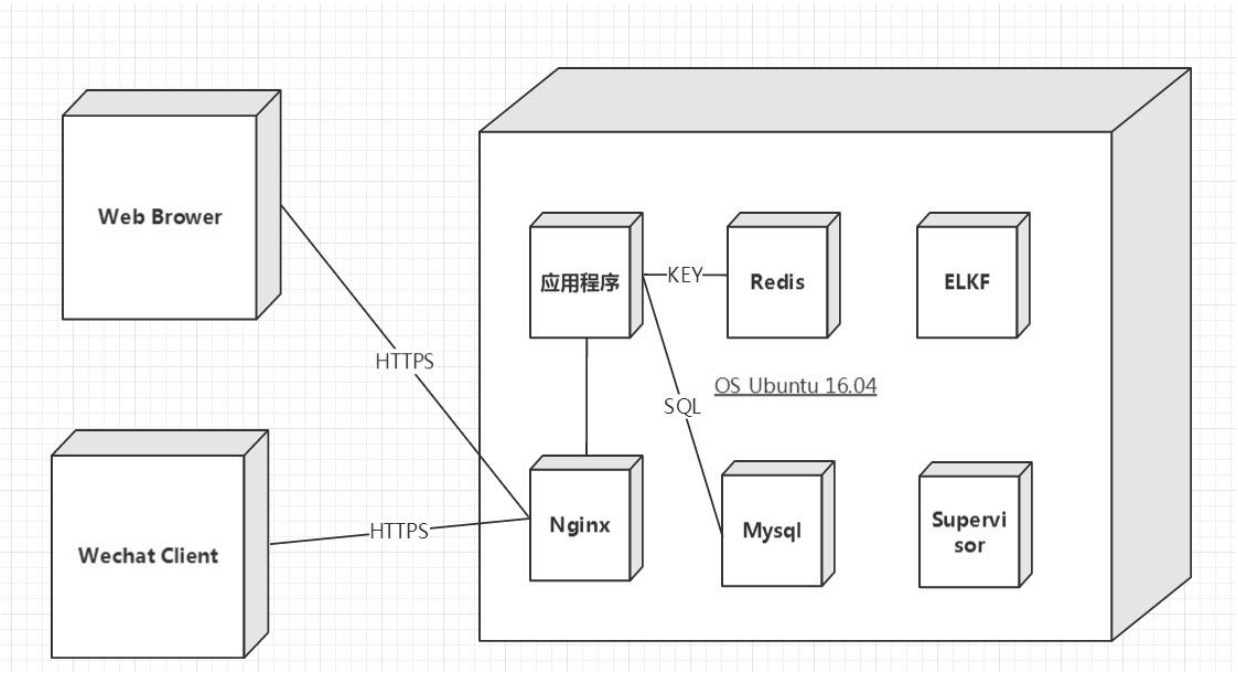

图 4.4 系统部署图

系统服务架构基于 LNMP (Linux Nginx MySQL PHP)，Redis，ELKF (Elasticsearch Logstash Kibana Filebeta) 监控系统，Redis 缓存，Supervisor 进程管理工具。客户端的请求通过 HTTPS，请求来源包含浏览器和微信小程序。

4.2 数据库设计

4.2.1 数据库环境说明

数据库名称：zd

数据库管理系统：MySQL 5.7

4.2.2 数据库命名规则

表 4-1 数据库术语表

缩写、术语	解 释
shop	知道网站数据库
admin_permission_role	权限和角色的关系表
admin_permissions	权限表
admin_role_user	角色和用户的关系表
admin_roles	角色表
admin_users	管理员用户表
browse_records	用户浏览记录
comments	评论表
fans	粉丝表
images	图片
jobs	队列表
migrations	Migration 表
notices	通知表

password_resets	重置密码
post_topics	文章表和专题表的联系
posts	发帖表
search_records	用户搜索记录
topics	专题表
user_notice	用户和通知的关系
users	用户表
zans	点赞表

4.2.3 数据库 ER 设计图

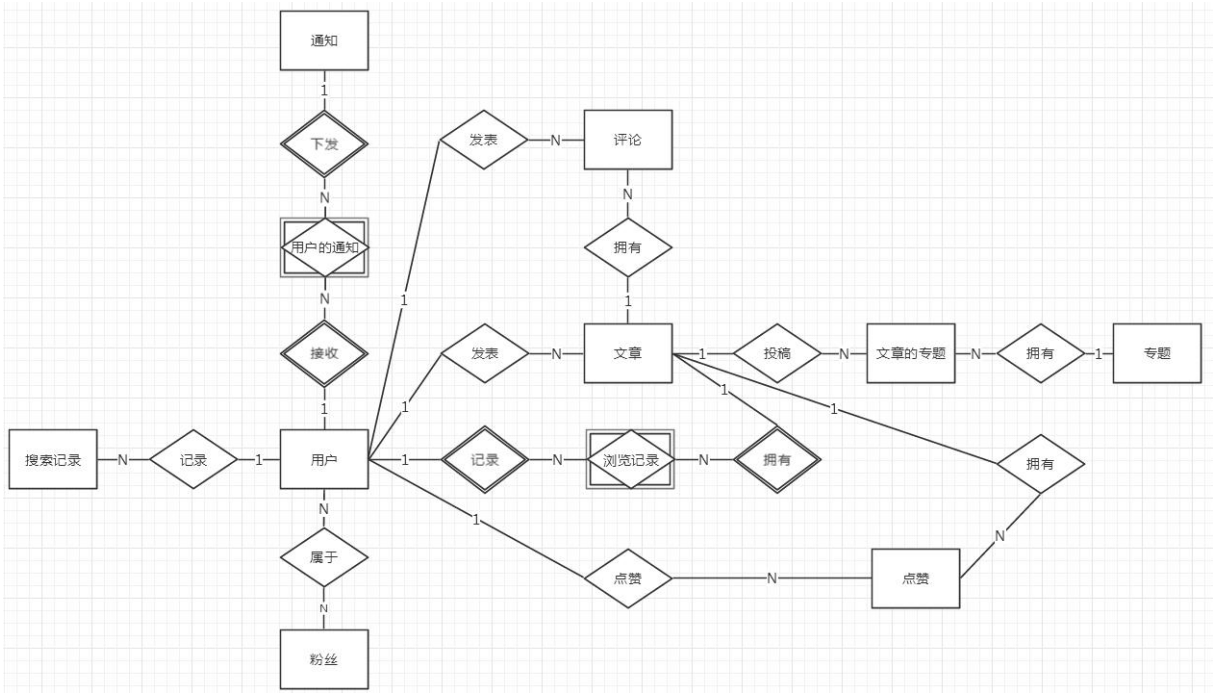

图 4.5 数据库 E-R 图 用户侧

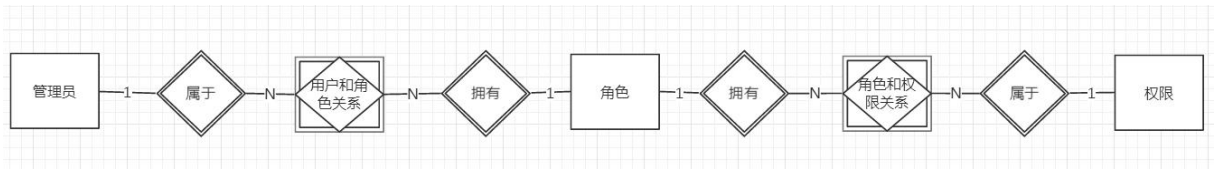

图 4.7 数据库 E-R 图 管理员侧

4.2.4 数据库逻辑图

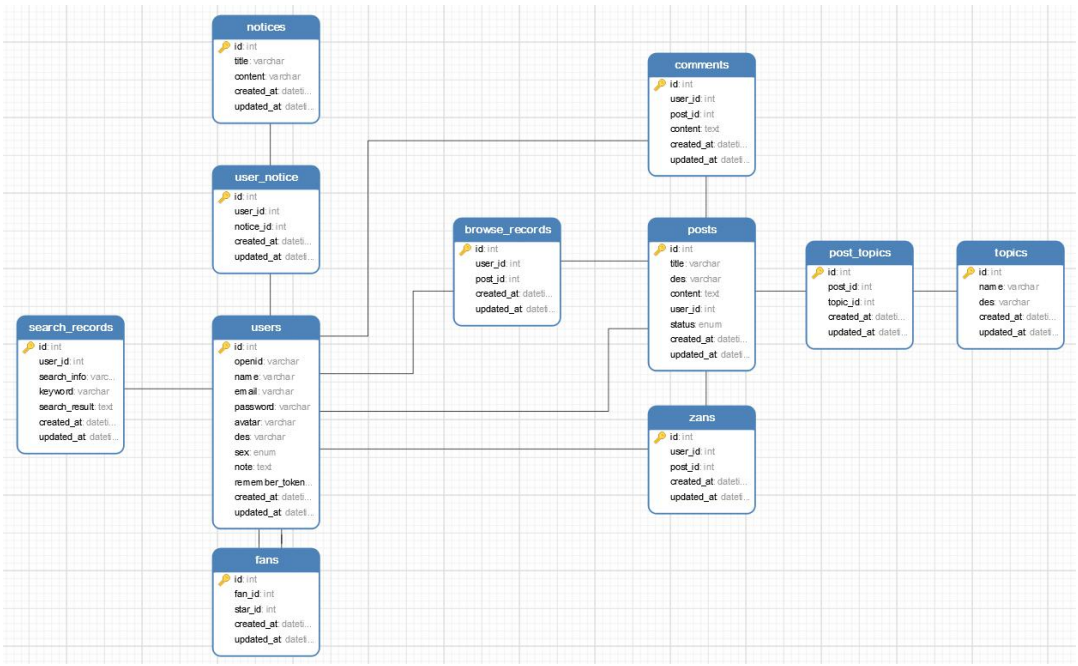

图 4.7 数据库逻辑图 用户侧

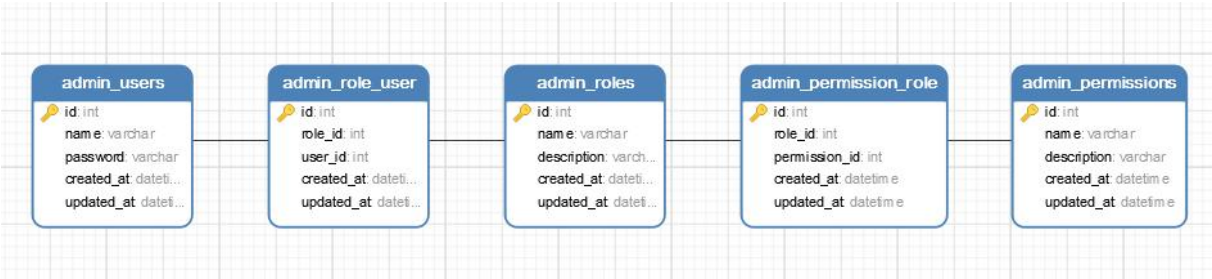

图 4.8 数据库逻辑图 管理员侧

4.2.5 数据字典

表 4-2 admin_permission_role 表

表名	admin_permission_role					
列名	数据类型	长度	空/非空	约束条件	中文描述	备注
id	int	int(10) unsigned	NO	PRI	序号	auto_increment
role_id	int	int(11) unsigned	NO	MUL	角色序号	

permissio n_id	int	int(11) u nsigned	NO	MUL	权限序号	
created_a t	datetime	datetime	NO		创建时间	
updated_a t	datetime	datetime	NO		更新时间	on update CURRENT_ TIMESTAMP

表 4-3 admin_permissions 表

表名	admin_permissions					
列名	数据类型	长度	空/非空	约束条件	中文描述	备注
id	int	int(10) u nsigned	NO	PRI	序号	auto_incr ement
name	varchar	varchar(3 0)	NO		名称	
descripti on	varchar	varchar(1 00)	NO		描述	
created_a t	datetime	datetime	NO		创建时间	
updated_a t	datetime	datetime	NO		更新时间	on update CURRENT_ TIMESTAMP

表 4-4 admin_role_user 表

表名	admin_role_user					
列名	数据类型	长度	空/非空	约束条件	中文描述	备注

id	int	int(10) unsigned	NO	PRI	序号	auto_increment
role_id	int	int(11) unsigned	NO	MUL	角色序号	
user_id	int	int(11) unsigned	NO	MUL	用户序号	
created_at	datetime	datetime	NO		创建时间	
updated_at	datetime	datetime	NO		更新时间	on update CURRENT_TIMESTAMP

表 4-5 admin_roles 表

表名	admin_roles					
列名	数据类型	长度	空/非空	约束条件	中文描述	备注
id	int	int(10) unsigned	NO	PRI	序号	auto_increment
name	varchar	varchar(30)	NO		名称	
description	varchar	varchar(100)	NO		描述	
created_at	datetime	datetime	NO		创建时间	
updated_at	datetime	datetime	NO		更新时间	on update CURRENT_TIMESTAMP

表 4-6 admin_users 表

表名	admin_users					
列名	数据类型	长度	空/非空	约束条件	中文描述	备注
id	int	int(10) unsigned	NO	PRI	序号	auto_increment
name	varchar	varchar(10)	NO	UNI	账号	
password	varchar	varchar(60)	NO		密码	
created_at	datetime	datetime	NO		创建时间	
updated_at	datetime	datetime	NO		更新时间	on update CURRENT_TIMESTAMP

表 4-7 browse_records 表

表名	browse_records					
列名	数据类型	长度	空/非空	约束条件	中文描述	备注
id	int	int(11) unsigned	NO	PRI	序号	auto_increment
user_id	int	int(11) unsigned	NO	MUL	用户序号	
post_id	int	int(11) unsigned	NO	MUL	发帖序号	
created_at	datetime	datetime	NO		创建时间	

updated_at	datetime	datetime	NO		更新时间	on update CURRENT_TIMESTAMP

表 4-8 comments 表

表名	comments					
列名	数据类型	长度	空/非空	约束条件	中文描述	备注
id	int	int(10) unsigned	NO	PRI	序号	auto_increment
user_id	int	int(11) unsigned	NO	MUL	用户序号	
post_id	int	int(11) unsigned	NO	MUL	文章序号	
content	text	text	NO		评论内容	
created_at	datetime	datetime	NO		创建时间	
updated_at	datetime	datetime	NO		更新时间	on update CURRENT_TIMESTAMP

表 4-9 fans 表

表名	fans					
列名	数据类型	长度	空/非空	约束条件	中文描述	备注
id	int	int(10) unsigned	NO	PRI	序号	auto_increment

fan_id	int	int(11) unsigned	NO	MUL	粉丝用户序号（1 关注 2　1）	
star_id	int	int(11) unsigned	NO	MUL	关注用户序号（1 关注 2　2）	
created_at	datetime	datetime	NO		创建时间	
updated_at	datetime	datetime	NO		更新时间	on update CURRENT_TIMESTAMP

表 4-10 images 表

表名	images					
列名	数据类型	长度	空/非空	约束条件	中文描述	备注
id	int	int(11)	NO	PRI	序号	auto_increment
user_id	int	int(11) unsigned	NO		用户序号	
type	varchar	varchar(50)	NO		类型	
type_id	int	int(11) unsigned	NO		类型序号	
url	varchar	varchar(500)	NO		图片链接	
extra_inf	text	text	NO		额外信息	

o						
status	enum	enum('0','1')	NO		状态 （0 不显示 1 显示)默认显示	
created_at	datetime	datetime	NO		创建时间	
updated_at	datetime	datetime	NO		更新时间	on update CURRENT_TIMESTAMP

表 4-11 jobs 表

表名	jobs					
列名	数据类型	长度	空/非空	约束条件	中文描述	备注
id	bigint	bigint(20) unsigned	NO	PRI	序号	auto_increment
queue	varchar	varchar(191)	NO	MUL		
payload	longtext	longtext	NO			
attempts	tinyint	tinyint(3) unsigned	NO			
reserved_at	int	int(10) unsigned	YES			
available	int	int(10) u	NO			

_at		nsigned				
created_at	int	int(10) unsigned	NO		创建时间	

表 4-12 migrations 表

表名	migrations					
列名	数据类型	长度	空/非空	约束条件	中文描述	备注
id	int	int(10) unsigned	NO	PRI		auto_increment
migration	varchar	varchar(255)	NO			
batch	int	int(11)	NO			

表 4-13 notices 表

表名	notices					
列名	数据类型	长度	空/非空	约束条件	中文描述	备注
id	int	int(10) unsigned	NO	PRI	序号	auto_increment
title	varchar	varchar(50)	NO		标题	
content	varchar	varchar(1000)	NO		内容	
created_at	datetime	datetime	NO		创建时间	
updated_a	datetime	datetime	NO		更新时间	on update

t						CURRENT_TIMESTAMP

表 4-14 password_resets 表

表名	password_resets					
列名	数据类型	长度	空/非空	约束条件	中文描述	备注
id	int	int(11)	NO	PRI	序号	auto_increment
name	varchar	varchar(30)	NO		名字	
email	varchar	varchar(100)	NO		序号	
token	varchar	varchar(500)	NO		token	
created_at	datetime	datetime	NO		创建时间	
updated_at	datetime	datetime	NO		更新时间	on update CURRENT_TIMESTAMP

表 4-15 post_topics 表

表名	Post_topics					
列名	数据类型	长度	空/非空	约束条件	中文描述	备注
id	int	int(10) unsigned	NO	PRI	序号	auto_increment

post_id	int	int(11) unsigned	NO	MUL	文章序号	
topic_id	int	int(11) unsigned	NO	MUL	专题序号	
created_at	datetime	datetime	NO		创建时间	
updated_at	datetime	datetime	NO		更新时间	on update CURRENT_TIMESTAMP

表 4-16 posts 表

表名	posts					
列名	数据类型	长度	空/非空	约束条件	中文描述	备注
id	int	int(10) unsigned	NO	PRI	文章序号	auto_increment
title	varchar	varchar(100)	NO		标题	
des	varchar	varchar(500)	NO		描述	
content	text	text	NO		内容	
user_id	int	int(11) unsigned	NO	MUL	用户序号	
status	enum	enum('0','1','-1')	NO		状态(0 1 -1)	
created_a	datetime	datetime	NO		创建时间	

t						
updated_at	datetime	datetime	NO		修改时间	on update CURRENT_TIMESTAMP

表 4-17 search_records 表

表名	search_records					
列名	数据类型	长度	空/非空	约束条件	中文描述	备注
id	int	int(11) unsigned	NO	PRI	序号	auto_increment
user_id	int	int(10) unsigned	NO	MUL	用户序号	
search_info	varchar	varchar(50)	NO		查询内容	
keyword	varchar	varchar(50)	NO		关键词	
search_result	text	text	NO		查询结果	
created_at	datetime	datetime	NO		创建时间	
updated_at	datetime	datetime	NO		更新时间	on update CURRENT_TIMESTAMP

表 4-18 topics 表

表名	topics

列名	数据类型	长度	空/非空	约束条件	中文描述	备注
id	int	int(10) unsigned	NO	PRI	序号	auto_increment
name	varchar	varchar(30)	NO		名称	
des	varchar	varchar(200)	NO		描述	
created_at	datetime	datetime	NO		创建时间	
updated_at	datetime	datetime	NO		更新时间	on update CURRENT_TIMESTAMP

表 4-19 user_notics 表

表名	user_notics					
列名	数据类型	长度	空/非空	约束条件	中文描述	备注
id	int	int(10) unsigned	NO	PRI	序号	auto_increment
user_id	int	int(11) unsigned	NO	MUL	用户序号	
notice_id	int	int(11) unsigned	NO	MUL	通知序号	
created_at	datetime	datetime	NO		创建时间	
updated_at	datetime	datetime	NO		更新时间	on update CURRENT_

						TIMESTAMP

表 4-20 users 表

表名	users					
列名	数据类型	长度	空/非空	约束条件	中文描述	备注
id	int	int(10) unsigned	NO	PRI	序号	auto_increment
openid	varchar	varchar(32)	YES		微信 openid	
name	varchar	varchar(30)	NO	UNI	名称	
email	varchar	varchar(30)	NO	UNI	邮箱	
password	varchar	varchar(60)	NO		密码	
avatar	varchar	varchar(500)	NO		头像	
des	varchar	varchar(200)	NO		描述	
sex	enum	enum('0','1')	NO		性别	
note	text	text	NO		备注	
remember_token	varchar	varchar(60)	YES		记住的 token	
created_at	datetime	datetime	NO		创建时间	

updated_at	datetime	datetime	NO		修改时间	on update CURRENT_TIMESTAMP

表 4-21 zans 表

表名	zans					
列名	数据类型	长度	空/非空	约束条件	中文描述	备注
id	int	int(10) unsigned	NO	PRI	序号	auto_increment
user_id	int	int(11) unsigned	NO	MUL	用户序号	
post_id	int	int(11) unsigned	NO	MUL	文章序号	
created_at	datetime	datetime	NO		创建时间	
updated_at	datetime	datetime	NO		更新时间	on update CURRENT_TIMESTAMP

5 系统详细设计与实现

5.1 邮箱注册 小程序注册 账号互通

需求：

在 PC 登录使用邮箱，在小程序登录直接使用微信账号，这样就需要将两个的账号体系打通。

如图所示：

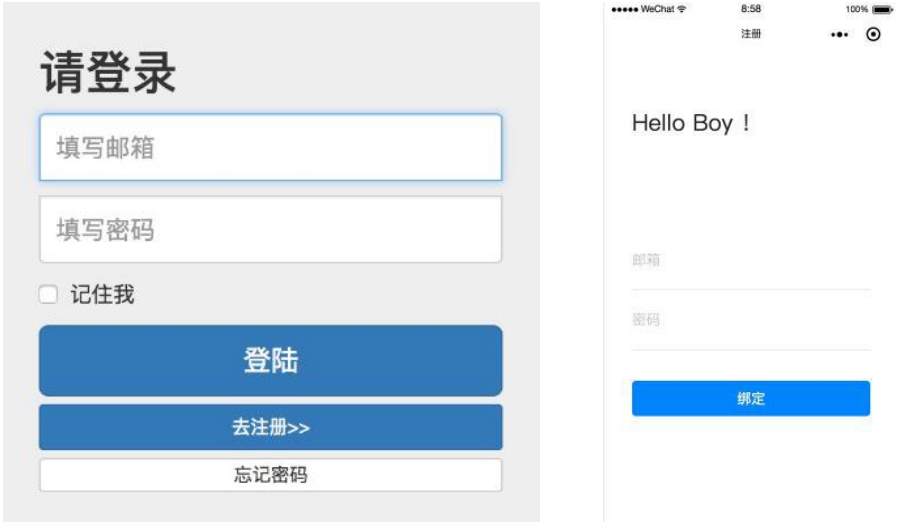

图 5.1 PC 登录 微信小程序登录

策略：

解决方法在数据库的设计上进行改动，有两种用户表的设计方案。

第一种方案：

id	int	int(10) unsigned	NO	PRI	序号	auto_increment
openid	varchar	varchar(32)	YES		微信 openid	
name	varchar	varchar(30)	NO	UNI	名称	
email	varchar	varchar(30)	NO	UNI	邮箱	

password	varchar	varchar(60)	NO		密码	

第二种方案：

id	int	int(10) unsigned	NO	PRI	序号	auto_increment
name	varchar	varchar(32)	NO	UNI	名字	
type	JSON_ARRAY		NO		注册类型	

目前采用的是第一种方案，能够满足目前的产品需求，openid 设置允许为空字段，email 和 password 字段设置为必填，在登录小程序的时候，需要先进行邮箱和密码的绑定。具体逻辑：判断是否存在这个邮箱，如果存在，且没有被其他微信账号绑定，则对比密码是否吻合，如果吻合，则将相关账号绑定 openid；如果不存在，则添加新的账号入数据表。如何应对需求变化（有新的接入方式）？可以添加新的字段进行相关的逻辑处理。

优化方法可以采取方案二。可以使用 MySQL 5.7 新加的特性：json 数据类型。一个账号允许有多个创建方式，例如：PC 端的邮箱注册，小程序的 openid 注册，QQ 的 uin 注册等。登录的时候查看是哪个登录方法，找到相应的注册方式进行账号的校验。需要注意 name 需要作为唯一验证依据，即 name 字段有唯一性，这样登录时候可以使用 name 作为检索。

关键代码

```php
// 检查邮箱是否 存在 已经绑定openid
$userModel = (new UserModel)->where(['email' => $param['email']])->find();
if ($userModel && $userModel->openid && $userModel->openid != $userInfo['openid']) {
    $e = new LogicException([
        'msg' => '该邮箱已经绑定其他微信账号，需要解绑',
    ]);
    throw $e;
}


if ($userModel && !$userModel->openid) {
    $checkPassword = sha1(config('params.salt') . sha1(config('params.salt') . $param['password']))
    == $userModel->password ? true : false;
    if ($checkPassword) {
        // 绑定邮箱
        $userModel->openid = $userInfo['openid'];
        $userModel->save();
    } else {
        $e = new LogicException([
            'msg' => '密码错误',
        ]);
        throw $e;
    }
} else if (!$userModel) {
    // 新建账号
    $userModel = new UserModel();
    $userModel->openid = $userInfo['openid'];
    $userModel->name = $userInfo['name'];
    $userModel->avatar = $userInfo['avatar'];
    $userModel->email = $param['email'];
    $userModel->note = '';
    $userModel->password = sha1(config('params.salt') . sha1(config('params.salt') . $param['password']));
    $userModel->save();
}
```

图 5.2 微信小程序账号注册 代码

```php
// 用户注册界面操作
public function register()
{
    // 验证
    $this->validate(request(), [
        'name' => 'required|min:3|max:20|unique:users,name',
        'email' => 'required|unique:users,email|email',
        'password' => 'required|min:5|max:16|confirmed'
    ], [
        'name.min' => '名字不能少于3个字',
        'name.max' => '名字不能超过20个字',
        'name.unique' => '名字出现重复',
        'name.required' => '必须填写名字',
        'email.email' => '必须填写邮箱格式',
        'email.required' => '必须填写邮箱',
        'email.unique' => '邮箱出现重复',
        'password.required' => '必须填写密码',
        'password.min' => '密码不能少于5个字',
        'password.max' => '密码不能超过16个字',
        'password.confirmed' => '两次密码不一致',
    ]);

    // 逻辑
    $name = request('name');
    $email = request('email');
    $password = sha1(config('myConfig.salt') . sha1(config('myConfig.salt') . request('password')));
    $avatar = 'https://images.charmingkamly.cn/system/default-header.png'; // 默认头像
    $note = '';

    $user = User::create(compact(['name', 'email', 'password', 'avatar', 'note']));
    // 返回
    return redirect('/login')->with('status', '注册成功');
}
```

图 5.3 PC 账号注册 代码

5.2 模板渲染

需求：

PC 端采用模板渲染来渲染相关需要展示的数据（后端模板渲染），小程序则采用的是接口方式获取数据然后通过小程序内部的方式将数据展示出来（前端模板渲染）。

策略：

对比两种渲染模式：

1. 后端模板渲染

   优势：

   a. 利于网站的 seo
   b. 数据安全性更高，不用担心 js 脚本被用户禁用
   c. 首屏渲染速度快

   缺点：

   a. 服务器的压力会增大
   b. 会降低用户体验感（页面数据需要改变，需要刷新页面）
   c. 可维护性差

2. 前端模板渲染

   优势

   a. 减少服务器压力
   b. 提高用户体验，灵活改变页面数据（无刷新页面）
   c. 可维护性高
   d. 提高页面性能（只改变 dom 结构）
   e. 跨平台（兼容不同后端技术）
   f. 完全实现前后端分离

   缺点

   a. 不利于 seo（搜索引擎无法抓取页面数据，因为只有模板，没有数据）
   b. 数据安全性比较低，有可能 js 脚本被用户禁用
   c. 首屏渲染比较慢

作者更加偏向前端模板渲染的方式，因为可以实现前后端分离。在接口定义上使用 Restful API 设计理论。例如：在协议（使用 https），域名（zdapp.charmingkamly.cn），版本（接口版本 v1），路径（user），HTTP 动词（get，post），过滤信息，状态码（200,404），错误处理，返回结果（json）等相关内容进行了相应的设计，尽量满足其设计原理。针

对于 RESTful API 还有另外一种解决设计思路是 GraphQL，它是一种用于 API 的查询语言，让前端更加方便访问数据。

除了前端渲染，后端渲染之外，还有同构策略，利用 JavaScript 编写的应用可以同时运行在客户端和服务器。

关键代码

```php
/*
 * 动态，文章创建时间排序
 */
public function indexPost(Request $request)
{
    $this->is_allow_access();
    $this->check_token();

    $param = $request->param();
    $page = empty($param['page']) ? 1 : $param['page'];
    $size = empty($param['size']) ? 5 : $param['size'];

    $posts = (new PostModel)->scope('avaiable')->order('created_at DESC')->page($page, $size)->select();

    $result = [];
    foreach ($posts as $post) {
        $result[] = $post->to_public_array($this->options);
    }

    return json_info($result);
}
```

图 5.4 后端模板渲染 代码

```php
// 文章详情页
public function show(Post $post)
{
    $post->load('comments'); // 预加载，这样view层不会再做查询操作
    $post->content = Markdown::defaultTransform($post->content);
    return view('post/show', compact('post'));
}
```

图 5.5 前端模板渲染 代码

5.3 数据同步写入 elasticsearch

需求：

用户发帖回帖的内容需要写入 elasticsearch 索引中，但是应用程序先将数据存储在 MySQL 的数据表中，因此会有一定的时间差。

策略：

针对于这个需求有两种解决方案。

第一种：重写框架中的函数，在操作完数据库后，立刻执行 Elasticsearch 的相关操作。第二种：使用相关的同步插件，将 MySQL 的操作同步到 Elasticsearch，比如：elasticsearch-jdbc，elasticsearch-river-mysql 等工具。

考虑到实际需求，本课题不需要严格的数据同步（即不需要全部数据都同步到 Elasticsearch 的索引中），因此采用的是第一种解决方案，重写框架中的 save,delete 函数，成功操作完数据库后，马上执行相应 Elasticsearch 的操作，实现数据同步策略。注意：这里马上执行相应的 Elasticsearch 的操作是扔进消息队列中进行队列消耗，避免过久的操作影响用户体验，对于用户来说是无感知的同步策略。还需要考虑 Elasticsearch 操作失败的情况。如果操作失败，需要重新扔进消息队列。

关键代码

```php
public function save(array $options = [])
{
    $post = parent::save($options);

    // 写入ela
    $jobData = [
        'type' => 'post',
        'action' => 'save',
        'id' => $this->id,
    ];
    // 分发任务
    dispatch(new \App\Jobs\ElasticsearchTodoJob($jobData));

    return $post; // TODO: Change the autogenerated stub
}

public function delete()
{
    $post = parent::delete();

    // 写入ela
    $jobData = [
        'type' => 'post',
        'action' => 'delete',
        'id' => $this->id,
    ];
    dispatch(new \App\Jobs\ElasticsearchTodoJob($jobData));

    return $post; // TODO: Change the autogenerated stub
}
```

图 5.6 PC 端同步 代码

```php
// 保存
public function save($data = [], $where = [], $sequence = null)
{
    $post = parent::save($data, $where, $sequence);

    // 写入ela
    $jobData = [
        'type' => 'post',
        'action' => 'save',
        'id' => $this->id,
    ];
    toDoJob('app\api\job\elasticsearchTodoJob', 'elasticsearchTodoJobQueue', $jobData);

    return $post;
}

// 删除
public function delete()
{
    $post = parent::delete();

    // 写入ela
    $jobData = [
        'type' => 'post',
        'action' => 'delete',
        'id' => $this->id,
    ];
    toDoJob('app\api\job\elasticsearchTodoJob', 'elasticsearchTodoJobQueue', $jobData);

    return $post;
}
```

图 5.7 小程序同步 代码

5.4 通知分发

需求：

管理员在发布通知的时候，需要遍历整个用户表。

策略：

正如需求所说，需要遍历整个用户表（数据量增大），有可能会造成锁表的结果（其他用户操作用户表），因此需要将这个操作放进消息队列中进行批量发布操作。

关键代码

```php
// 通知创建动作
public function store()
{
    $this->validate(request(), [
        'title' => 'required|min:4|max:50',
        'content' => 'required'
    ], [
        'title.min' => '名字不能少于4个字',
        'title.max' => '名字不能超过50个字',
        'title.required' => '必须填写名字',
        'content.required' => '必须填写内容',
    ]);

    $notice = Notice::create(request(['title', 'content']));

    // 分发任务
    dispatch(new \App\Jobs\SendNoticeMessage($notice));

    return redirect('admin/notices');
}
```

图 5.8 管理后台通知分发 代码

5.5 上传图片

需求：

本课题有多处需要上传图片的功能。

策略：

正常来说都是将用户上传的图片存储在本地云服务器中，直接通过相关链接请求。但是直接存储在云服务器，会增加系统本身的流量和带宽压力，且由于本身云服务器的带宽资源仅有 1mbps，因此本课题考虑到这三个原因，所以将图片资源存储在七牛云服务器上，将部分静态资源请求的压力转移到其他云服务上。

选择七牛云的原因有多个：1.针对于对象存储有相应较高的技术，稳定可靠，提 cdn 的加速传输；2.存储空间，流量和带宽收费在合理的接受范围；3.可以定制自己的二级域；4.提供免费的证书，即可以使用 https 的请求。

由于课题有多处上传图片的操作，所以在使用七牛云的 sdk 中，额外封装了一层自己写的第三方类库，方便自己调用上传，获取图片 https 链接等的功能。

关键代码

```php
if ($request->file('avatar')) {

    $disk = QiniuStorage::disk('qiniu');
    $filename = $disk->put('header', $request->file('avatar'));
    $img_url = urldecode($disk->downloadUrl($filename, 'https')); //获取下载链接

    // 插入Image
    $image = new Image();
    $image->user_id = \Auth::id();
    $image->type = 'header';
    $image->url = $img_url;
    $image->status = '1';
    $image->extra_info = '';
    $image->save();

    $user->avatar = $img_url;
}
```

图 5.9 PC 端上传图片 代码

```php
// 二进制
public function put($key, $stream)
{
    $uploadMgr = new UploadManager();  // 构建 UploadManager 对象
    return $uploadMgr->put($this->token, $key, $stream);
}

// 文件
public function putFile($key, $filePath)
{
    $uploadMgr = new UploadManager();  // 构建 UploadManager 对象
    return $uploadMgr->putFile($this->token, $key, $filePath);
}

// extra=?imageView2/1/h/500
public function downloadUrl($key, $extra = null)
{
    return $this->domains . $key . $extra;
}
```

图 5.10 小程序上传图片 代码

5.6 小程序登录

需求：

小程序通过微信官方提供的登录接口能够方便获取用户身份标示，建立小程序内的账号体系。

策略：

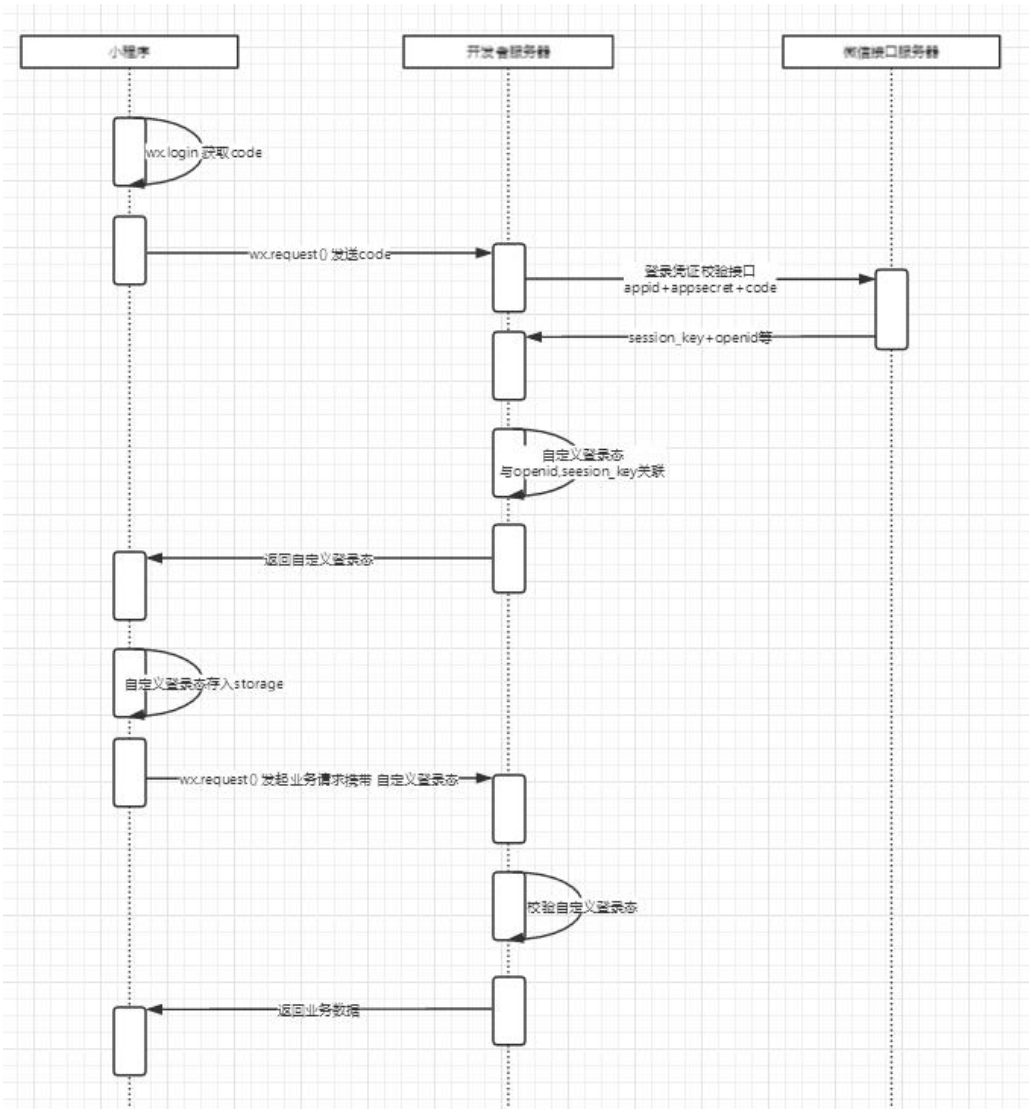

图 5.11 小程序登录 时序图

小程序调用 wx.login()的 api 获取有时效 code（登录凭证），使用微信小程序提供 request 的 api 传送其内容到指定的开发者服务器。开发者服务器利用接收到的登录凭证请求微信服务器，换取用户唯一标识 openid 和会话密钥 session_key。开发者服务器在获取得到 openid 和 session_key 之后，需要进行相关解密操作（具体操作查看微信文档）。解密成功之后，开发者服务器可以根据实际的业务需求来自定义用户的登录态（例如进行自己写的加密算法生成相应的加密数据），用于后续业务逻辑中前端后端交互时检测用户的相应的状态，识别用户信息。

对于本课题的自定义登录态携带的信息有用户序号，用户 openid 和当前时间，并且进行 des 和 aes 的字符串加密。在需要识别用户身份的时候，需要通过携带_token 头请求开发者服务器，开发者服务器进行相应的解密和数据校验。其中当前时间是和请求时间进行对比，如果超过某个阀值则判断自定义登录态失效，需要用户重新登录。

关键代码

```php
// 获取 code rawData signature encryptedData  iv
$param = $request->param();

// 通过code换取相关信息
$MyWxApi = new \MyWxApi();
$webapp = $MyWxApi->wx_get_webapp_openid($param['code']);
if (!$webapp) {
    $e = new LogicException([
        'msg' => '通过code换取相关信息为空值',
    ]);
    throw $e;
}

// 这里有坑，校验的时候会有失败的情况
// 验证 signature = sha1(rawData+session_key)
if ($param['signature'] != sha1($param['rawData'] . $webapp['session_key'])) {
    $message = [
        'functionName' => 'application/api/controller/v1/Webapp.php/token',
        'params' => $param,
        'type' => 'error',
        'message' => '验证 signature = sha1(rawData+session_key) 失败',
        'result' => $webapp,
    ];
    log_function($message);
}

// 解密数据
$err_code = $MyWxApi->aes128_cbc_decrypt($webapp['session_key'], $param['encryptedData'], $param['iv']);
if (!$err_code) {
    $e = new LogicException([
        'msg' => '解密数据为空值',
    ]);
    throw $e;
}
```

图 5.12 小程序获取 openid 代码

5.7 生成小程序码

需求：

用户可以通过小程序码快速进入小程序的相应界面，方便用户进行知识传播，也扩大知识的影响力，进行更强烈的思维碰撞，结识更多有趣的小伙伴。

策略：

当用户请求生成小程序码的时候，需要两个步骤，第一个步骤请求开发者服务器，开发者服务器请求微信服务器获取小程序码，第二个步骤获取用户头像，将小程序码的 logo 换成用户头像。

针对于第二个步骤，有两种方案。第一种方案将合成的功能放置到开发者服务器，如果请求量增大，这样就会造成服务器的压力过大，但是也有好处合成速度更快。第二种方案使用小程序内的 canvas 函数合成图片，这样可以减轻服务器压力，但是合成速度根据不同机型有不同的效果。本课题采用的是第二种方案，由于尽量避免对开发者服务器造成过大压力，将压力转移到每个客户端上。

关键代码

```
// 请求服务器生成二维码 scene page
async createQrcodeDownloadAvatar (scene) {
  // 请求网络生成二维码，并且下载本地
  const that = this
  const res = await store.dispatch(apiFetchCreateQrcodebRedux({
    token: wepy.getStorageSync('token'),
    scene: scene,
    page: `/pages/postShare`,
    width: 430
  }))

  const isSuccess = await util.common.isSuccessPromise(res.data)
  if (isSuccess) {
    // 将二维码下载本地
    const qrcodeTempFile = await wxNet.downloadFilePromise(res.data.message.url)
    // 将头像下载本地
    const avatarTempFile = await wxNet.downloadFilePromise(that.showPost.user.avatar)

    return [qrcodeTempFile.tempFilePath, avatarTempFile.tempFilePath]
  }
}
```

图 5.13 小程序请求生成二维码 代码

```
$MyWxApi = new \MyWxApi();
$accessToken = $MyWxApi->wx_get_access_token('webapp'); // 获取access_token
$url = "https://api.weixin.qq.com/wxa/getwxacodeunlimit?access_token={$accessToken}";

$result = curlPost($url, json_encode($postData)); // 请求链接，获取图片的二进制
$checkResult = json_decode($result, true);
if ($checkResult['errcode'] != 0) {
    $e = new LogicException([
        'msg' => '生成二维码失败',
    ]);
    throw $e;
}
```

图 5.14 小程序后台生成二维码 代码

5.8 抽离小程序组件

需求：

针对于 loading 效果的代码优化，可以将页面内重复使用的功能模块抽离成自定义组件，便于开发者在不同的 page 中重复利用，有助于代码维护。

如图所示：

图 5.15 loading 效果

策略：

WePY 框架中的拥有组件特性，实现组件化的解耦与复用的效果。本课题就是利用 WePY 的组件特性，只需要 import 指定的组件，引用到页面中，即可组装成为页面的元素。

引入 loading 组件，可以提高用户体验，增加用户视觉感，让用户操作更加清晰。

关键代码

```
<template>
  <view>
    <block wx:if="{{loadingContainer === 0}}">
      <view class="loading-container-0">
        <view class="rect1" style="background-color:{{loadingColor}}"></view>
        <view class="rect2" style="background-color:{{loadingColor}}"></view>
        <view class="rect3" style="background-color:{{loadingColor}}"></view>
        <view class="rect4" style="background-color:{{loadingColor}}"></view>
        <view class="rect5" style="background-color:{{loadingColor}}"></view>
      </view>
    </block>

    <block wx:if="{{loadingContainer === 1}}">
      <div class="loading-container-1">
        <div class="cube1" style="background-color:{{loadingColor}}"></div>
        <div class="cube2" style="background-color:{{loadingColor}}"></div>
      </div>
    </block>

    <block wx:if="{{loadingContainer === 2}}">
      <div class="loading-container-2" style="background-color:{{loadingColor}}"></div>
    </block>

    <block wx:if="{{loadingContainer === 3}}">
      <div class="loading-container-3">
        <div class="dot1" style="background-color:{{loadingColor}}"></div>
        <div class="dot2" style="background-color:{{loadingColor}}"></div>
      </div>
    </block>

  </view>
</template>
```

图 5.16 loading 组件 代码

5.9 小程序和 Thinkphp 日志

需求：

通过日志文件记录 nginx 服务器接收不同类型的请求和程序自身运行时各种日志信息，并且以.log 结尾。经过 ELKF 日志分析工具针对日志文件进行相应统计、分析等操作就能快速地把握网站运行状况，快速地发现和排除错误原因，更好地强化系统的维护和管理。

如图所示：

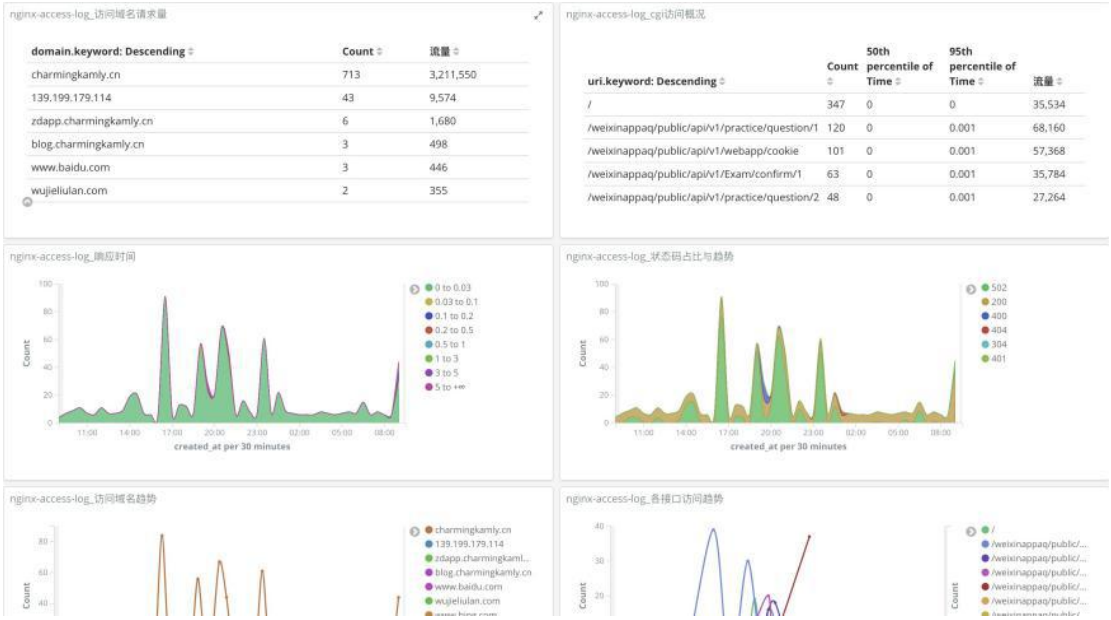

图 5.17 Kibana 可视化

策略：

小程序自身也有丰富的日志记录功能，管理员可以在后台或者微信小程序管理平台中轻松了解得到。但是这还不满足于针对性的日志记录，因此本课题封装了针对于 WePY 的自动上报日志的第三方类库。只需要简单的引入，就能将程序中发生的错误或者需要收集指定的点击日志收集起来，更加丰富了小程序的日志体系，可以从多个方面，多个维度进行小程序的产品分析，制定更好的运营策略。

除了小程序的自动上报，在小程序的后台也采用相似的功能，将每一次的输出，每次的报错都记录到日志中，其中 cgi.log 负责输出日志记录，包括错误输出，function.log 负责应用程序中特定的函数执行情况，且使用统一的日志格式记录，方便分析和及时排除错误。

本课题使用的是 ELKF 日志分析工具，其中 Elasticsearch 当成日志数据搜索引擎，Logstash 作为日志数据上报系统，日志文件收集系统使用 Filebeat，Kibana 为日志分析工具提供可视化的 Web 操作页面。这样不仅可以集中化管理日志，还可以实现查询，排序，统计等功能，从不同的报表，不同的折线图等图来监控业务性能，业务的指标等指标，即将日志可视化展示出来。

关键代码

```
class WxLog {
  /* eslint-disable no-console, no-bitwise*/
  constructor(param) { // 初始配置

    let config = param;
    if (typeof config === 'undefined') {
      throw new Error('lajax初始化错误 - 构造函数的参数不能为空! ');
    }
    if (typeof config === 'object') {
      if (typeof param.url !== 'string') {
        throw new Error('lajax初始化错误 - 构造函数的参数 url 必须是一个字符串! ');
      } else if (param.app === null) {
        throw new Error('lajax初始化错误 - 构造函数的参数 app 不能为空! ');
      }
    } else {
      throw new Error('lajax初始化错误 - 构造函数的参数格式不正确! ');
    }
```

图 5.18 小程序日志上报 代码

```
/*
 * 写日志-丰富message内容
 */
function log_message($message, $filename)
{
    $request = \think\Request::instance();
    $data = [
        'date' => date('c'), // 时间
        'ip_address' => $request->ip(), // IP地址
        'action' => "{$request->module()}.{$request->controller()}.{$request->action()}", // 动作
        'response_time' => intval((microtime(true) - $_SERVER['REQUEST_TIME_FLOAT']) * 1000), // 回应时间
        'url' => $request->url(), //URL地址
        'user_agent' => stripslashes(@$_SERVER['HTTP_USER_AGENT']), // 浏览器信息
        'message' => json_encode($message), // 信息
        'errorCode' => $message['errorCode'] // errorCode
    ];
    write_log(json_encode($data), $filename);
}
```

图 5.19 小程序后台日志记录 代码

6 系统界面

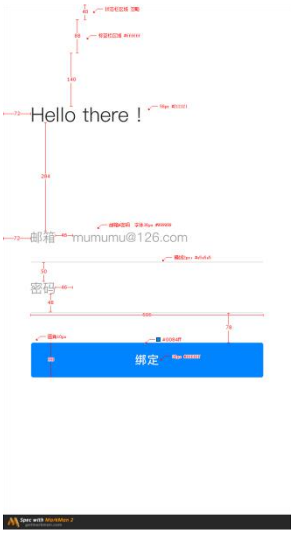

图 6.1 小程序绑定

图 6.2 搜索结果

图 6.3 搜索界面

图 6.4 更多界面

图 6.5 动态页面

图 6.6 专题页面

7 系统测试

7.1 测试概述

本课题采用黑盒测试。针对不同的浏览器（IE，火狐，谷歌，QQ 浏览器）和不同的手机（苹果和安卓），检测是否存在界面的兼容性问题；针对于输入框的，检测用户输入的内容的临界条件以及各种不符合输入规范的内容。

7.2 测试目的

世界上没有完美的系统，可以通过规范的测试，发现软件的各种不同的缺陷。系统存在的缺陷带来的损失是不可估量的，一旦发现，应该立即进行相关的修复，避免有意之人利用缺陷给用户或者系统造成无法挽回的损失。

7.3 测试环境

服务器	
硬件	腾讯云服务器 4 核 8G
软件	操作系统：Ubuntu16.04，数据库：MySQL 5.7
网络环境	以太网 100.0MBps

客户端	
硬件	内存:48.00GB、硬盘：SSD 5T、CPU: Internet I7
软件	浏览器：谷歌、操作系统：Win7(64bit)
网络环境	以太网 100.0MBps

7.4 测试执行步骤

1. 购买相应配置的腾讯云服务器，安装 ubuntu16.04 系统
2. 利用本作者开源的自动化构建脚本搭建相应的测试环境。LNMP，Redis，ELKF 日志监控系统，Redis 缓存，Supervisor 进程管理工具
3. 利用本作者开源的自动化部署脚本搭建相应的应用程序。

4. 应用程序部署成功，执行测试用例
5. 根据测试用例详细执行步骤逐一执行
6. 在执行测试用例过程发现的缺陷，将其逐一记录到缺陷文档中
7. 重复步骤 6，直至全部测试用例执行完

7.5 测试需求

需求编号	需求简述	优先级	备注说明
001	注册用户	高	1.校验邮箱合法性 2.校验密码长度
002	登录系统	高	1.校验邮箱合法性 2.校验密码长度 3.校验邮箱和密码是否匹配
003	申请重置密码	高	校验合法性
004	重置密码	高	校验密码长度
005	绑定微信账号	高	1.校验邮箱合法性 2.校验密码长度
006	解绑微信账号	高	清除相关信息
007	发表文章	高	1.校验标题长度 2.校验文章内容长度
008	编辑文章	高	1.校验标题长度 2.校验密码长度
009	删除文章	高	清除相关信息
010	发表评论	高	1.校验评论长度
011	编辑评论	高	1.校验评论长度
012	删除评论	高	清除相关信息

013	投稿专题	高	投稿文章内容
014	点赞文章	高	无
015	取消点赞文章	高	无
016	关注其他用户	高	无
017	取消关注其他用户	高	无
018	搜索文章评论	高	模糊搜索相关文章和评论
019	编辑个人信息	高	校验个人信息
020	通知信息	高	无
021	生成文章二维码	高	二维码能否跳转
022	分享指定群聊	高	指定的群聊
023	编辑权限	高	校验权限名称
024	添加管理员	高	1.校验管理员名字 2.校验管理员备注
025	跳转发表文章页面	高	链接测试
026	跳转具体文章	高	链接测试
027	跳转个人设置	高	链接测试
028	跳转重置密码	高	链接测试
029	跳转通知页面	高	链接测试

7.6 测试用例

测试用例编号	测试用例名称	步骤	用例状态	预期结果	测试结果
001	注册用户	1.用户进入首页，点击去注册按钮，跳转注册页面	执行	系统提示注册成功，返回平台主页	与预期结果一致

		2.用户填写邮箱和密码，点击保存按钮 3.系统验证邮箱和密码的合法性			
002	登录系统	1.用户进入首页 2.用户填写邮箱和密码，点击登录按钮 3.系统验证邮箱和密码的正确性	执行	系统验证正确后，系统提示登录成功，返回帖子列表	与预期结果一致
003	申请重置密码	1.用户进入首页 2.点击忘记密码按钮，进入重置密码页面 3.用户填写邮箱，点击发送邮件按钮 4.系统验证邮箱合法性	执行	系统验证合法后，系统提示邮件发送成功	与预期结果一致
004	重置密码	1.用户进入收到的邮件，点击重置密码的链接，进入重置密码页面 2.用户填写新密码，点击确认密码按钮 3.系统验证密码的合法性	执行	系统验证合法后，系统提示重置密码成功，返回首页	与预期结果一致
005	绑定微信账号	1.用户首次进入小程序，跳转绑定页面	执行	系统验证正确后，系统提示绑定成功，返回首页	与预期结果一致

		2.用户填写邮箱和密码，点击绑定 3.系统验证邮箱和密码的正确性			
006	解绑微信账号	1.用户进入更多页面，点击解绑按钮 2.系统进行解绑	执行	系统提示绑定成功，返回首页	与预期结果一致
007	发表文章	1.用户进入发表文章界面 2.用户填写文章标题，文章内容，点击发表按钮 3.系统验证文章内容	执行	系统验证成功后，系统提示发表成功，返回首页	与预期结果一致
008	编辑文章	1.用户进入文章编辑界面 2.用户填写文章标题，文章内容，点击保存按钮 3.系统验证文章内容	执行	系统验证成功后，系统提示发表成功，返回首页	与预期结果一致
009	删除文章	1.用户进入文章界面 2.用户点击删除按钮 3.系统验证删除内容	执行	系统删除成功后，系统提示删除成功，返回首页	与预期结果一致

010	发表评论	1.用户进入发表评论界面 2.用户填写评论内容，点击发表按钮 3.系统验证评论内容	执行	系统验证成功后，系统提示发表成功，返回首页	与预期结果一致
011	编辑评论	1.用户进入评论编辑界面 2.用户填写评论内容，点击保存按钮 3.系统验证评论内容	执行	系统验证成功后，系统提示发表成功，返回首页	与预期结果一致
012	删除评论	1.用户进入评论界面 2.用户点击删除按钮 3.系统验证删除内容	执行	系统删除成功后，系统提示删除成功，返回首页	与预期结果一致
013	投稿专题	1.用户进入专题界面 3.用户点击投稿按钮,选择需要投稿的文章，点击确定 3.系统验证投稿内容	执行	系统验证成功后，系统提示投稿成功	与预期结果一致
014	点赞文章	1.用户进入文章界面	执行	系统验证成功后，系统提示点赞成功	与预期结果一致

		2.用户点击点赞按钮 3.系统验证点赞文章			
015	取消点赞文章	1.用户进入文章界面 2.用户点击取消点赞按钮 3.系统验证取消点赞文章	执行	系统验证成功后，系统提示点赞成功	与预期结果一致
016	关注其他用户	1.用户进入用户介绍界面 2.用户点击关注按钮 3.系统验证关注用户	执行	系统验证成功后，系统提示关注成功	与预期结果一致
017	取消关注其他用户	1.用户进入用户介绍界面 2.用户点击取消关注按钮 3.系统验证取消关注用户	执行	系统验证成功后，系统提示关注成功	与预期结果一致
018	搜索文章评论	1.用户进入搜索界面 2.用户填写搜索的内容，点击搜索按钮 3.系统进行相关搜	执行	系统搜索成功后，系统展示相关内容	与预期结果一致

		索			
019	编辑个人信息	1.用户进入个人信息编辑界面 2.用户填写个人信息内容，点击保存按钮 3.系统验证个人信息内容	执行	系统验证成功后，系统提示编辑成功，返回首页	与预期结果一致
020	通知信息	用户进入通知界面	执行	系统显示通知	与预期结果一致
021	生成文章二维码	1.用户进入文章界面 2.用户点击转发按钮，点击生成二维码	执行	4. 用户进入文章界面 5. 用户点击转发按钮，点击生成二维码	与预期结果一致
022	分享指定群聊	1.用户进入文章界面 2.用户点击转发按钮，点击转发群，选择指定的群聊	执行	系统转发文章到指定的群聊	与预期结果一致
023	编辑权限	1.管理员进入权限界面 2.管理员选择需要编辑的权限信息 3.管理员编辑权限信息，点击保存按钮 4.系统验证权限信息	执行	系统验证成功后，系统提示编辑权限信息成功	与预期结果一致
024	添加管理	1.管理员进入添加	执行	系统验证成功后，系统提示添加新管理员成功	与预期结果一致

	员	管理员界面 2.管理员填写新的管理员信息,点击保存按钮 3.系统验证管理员信息			
025	跳转发表文章页面	点击发表按钮	执行	跳转成功	与预期结果一致
026	跳转具体文章	点击文章页面	执行	跳转成功	与预期结果一致
027	跳转个人设置	点击个人设置页面	执行	跳转成功	与预期结果一致
028	跳转重置密码	点击重置密码按钮	执行	跳转成功	与预期结果一致
029	跳转通知页面	点击通知页面	执行	跳转成功	与预期结果一致

总结

本课题设计从 17 年 10 月已经开始筹划，经历多次代码迭代，在 18 年的 4 月推出了知道网站的第一个版本，相信未来还会继续进行迭代。过程虽然艰辛且漫长，但是能够学习到能多东西，弥补了很多自己不足之处，还能扩大自己的视野，向未知领域探索。

大学的四年生活中，遇到不少坑，其中搭建环境这个坑实在是太大太大了。在很多时候，漫出第一步是最最最困难的，相当于后台开发的搭建环境。在学习初期有一键安装的软件包，但是当学习深入后，发现环境不知道咋配置了。因此在环境配置上留下了很多血与泪，可谓说处处挖坑，处处碰壁，但服务搭建成功后那种喜悦是说不出来的。所以在本次课题设计中，我开源了和 PHP 环境相关的意见配置脚本，还有拥有 ELKF 一键安装的脚本。该脚本开源在 Github 上，欢迎不同的开发者提出 issue。利用此脚本，可以在 20 分钟之内构建好拥有 lnmp 的服务器环境。尽可能做到应用开发者只需要关注业务层面，将服务器运维的业务抽离出来。

为了实现部署应用程序自动化，本课题设计还开源了以 PHP 为核心的自动化部署脚本。利用了 GitHub 的 webhooks 的钩子，一旦有 git push 的提交，GitHub 的 webhooks 就会发送相关内容到指定的服务器。服务器接收到相关内容之后，进行相应的脚本化部署。这个开源项目已经经历过 2 个版本迭代，最新的一个版本使用了消息队列处理，并且还有相应的日志记录。这样就能做到提交代码后，喝着咖啡，等待应用程序的部署。将开发人员从繁杂和重复的部署任务抽离出来。

在本次课题设计中，采用了后端模板渲染和前端模板渲染两种不同的渲染模式，对此有更加深刻的了解。我偏向于前端模板渲染的方式，这样后端能更专注于接口的处理，前端则只需要关注接口的使用。在接口设计方面，还需要进一步优化，虽然在实践上使用上了 RestfulAPI 的设计理论，但是还有很多坑没有填，由于不熟悉，导致接口设计方面越写越乱，最终需要全部重新进行调整。

随着业务的发展，网站更容易被黑产盯上，因此更需要修复系统中的漏洞，及时进行热修复，避免有意之人的有意行为。例如可以排除 XSS 攻击，CSRF 攻击，点击劫持，传送安全，密码安全等。针对于传送安全，本课题的网站全部使用 HTTPS，采用腾讯云提供的免费证书，保证传送内容的安全性和可靠性；在小程序中，派发的 token 也是经过相应的加密处理，即使有意之人获取也没法将其内容解析。

本课题的系统仍有很多不足之处。例如：随着业务量的提升，在数据库设计层面，应该考虑将数据库中的表进行分表等；在架构设计方面，可以使用相应的分布式集群处理，将请求分发到不同的机器上，减轻同一台机子的压力；还能将数据库的读写进行分离，进一步提升机器的性能；在数据库中，还需要注意相应的数据备份策略，使用热备

份还是冷备份；在应用层面，对于突然来袭的大流量，可以考虑使用漏洞算法和令牌桶算法，更加平滑地限制流量。

给我印象深刻的是，在遇到逻辑上的坑时候，需要停停下来思考，自己从多个维度进行思考，给出不同的解决方案，最后择优使用。这样不仅能够避免拍脑子思考，还能扩大自己的思维能力，让自己思维更加敏捷。

参考文献

[1]王珊.萨师煊.《数据库系统概论》第 4 版 2013 年，高等教育出版社；
[2]谢希仁.《计算机网络》第 5 版 2008 年 1 月，电子工业出版社；
[3]朱林.《Elasticsearch 技术解析与实战》第 1 版 2017 年 1 月，机械工业出版社；
[3]鸟哥.《鸟哥的 Linux 私房菜 基础学习篇》第 3 版 2010 年 7 月，人民邮电出版社；
[4]鸟哥.《鸟哥的 Linux 私房菜 服务器架设篇》第 3 版 2012 年 7 月，机械工业出版社；
[5]腾讯 CDC.《在你身边，为你设计 2 腾讯的移动用户体验设计之道（全彩）》第 1 版 2016 年 1 月，电子工业出版社；
[6]Andrew S. Tanenbaum，Herbert Bos.《现代操作系统（原书第 4 版）》第 4 版 2017 年 7 月，机械工业出版社；
[7]BaronSchwartz，PeterZaitsev，VadimTkachenko，宁海元.《高性能 MySQL（第 3 版）》第 3 版 2013 年 4 月，电子工业出版社；
[8]尼古拉斯·泽卡斯.《JavaScript 高级程序设计（第 3 版）》第 3 版 2012 年 3 月，人民邮电出版社；
[9]苗泽.《Nginx 高性能 Web 服务器详解》第 1 版 2013 年 10 月，电子工业出版社；
[10]秦朋.《PHP7 内核剖析》第 2 版 2017 年 10 月，电子工业出版社；

谢辞

岁月如歌，大学的四年光阴瞬间即逝，回想当年的懵懂入学，到现在准备带上四方帽。在大学四年里，感谢学校的每一位老师无私的传授知识，指引我们走上正确道路上，让我们知道接下来的日子该怎么走，在这里对他们表示由衷的感激。感谢他们提供了不同的平台让我学习到更多和计算机相关的知识，更有信心在软件开发这条路上勇往直前，撸起袖子干！

在这几个月中的课题设计中，不断地在优化方案，尽可能将自己学习到的知识全部应用到毕设，更加深了相关知识的应用。从毕设选题到实践，再到论文的书写，邵珂老师一直耐心地指导我整个系统，指出了系统中的重点和难点需要优先突破。每当遇到思维短路的时候，老师总能把我从坑里挖出来，重新找到一条新的路子。在实践中，严格要求我的代码风格，尽可能做到看代码即懂其逻辑，教会我相应项目的组织架构，让我有更清晰的代码逻辑。从心感谢邵珂老师的专业指导，让我顺利完成毕业设计。