郑州轻工业学院

本科毕业设计（论文）

|  |  |  |
| --- | --- | --- |
| 题 目 | 基于Nodejs的社团 | |
|  | 集成工具式管理系统 | |
| 学生姓名 | 张超 | |
| 专业班级 | 电子信息科学与技术14-01 | |
| 学 号 | 541407020149 | |
| 院 （系） | 计算机与通信工程学院 | |
| 指导教师（职称） | | 陈冬冬（讲师） |
| 完成时间 | 2018年6月15日 | |
|  |  | |

郑州轻工业学院

毕业设计（论文）任务书

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 题目 | | 基于Nodejs的社团集成工具式管理系统 | | | | | | | | | | | | |
| 专业 | | 电子信息14-01 | | | | | 学号 | | 541407020149 | | | | 姓名 | 张超 |
| 主要内容、基本要求、主要参考资料等: | | | | | | | | | | | | | | |
| 一、主要内容  1. 了解社团管理过程中的缺陷；  2. 对基于Nodejs的社团管理系统进行需求分析，提出总体设计方案；  3. 设计实现基于Nodejs的社团管理系统的各功能模块。  二、基本要求  1. 拥有基本的功能，易于演示和上线部署；  2. 性能良好、易于维护、管理功能人性化，有较强的交互性。  三、主要参考资料   1. 张英. 微服务: 开创图书馆服务的“蓝海”[J]. 图书馆建设.2011(7):51-53. 2. WILSON J. Node. js 8 the Right Way: Practical, Server-side Javascript that Scales[M]. [S.l.]: Pragmatic Bookshelf,2018 3. MARDAN A. Express. js Guide: The Comprehensive Book on Express. js[M]. [S.l.]: AzatMardan, 2014. 4. 介绍—Vue.js[M/OL] [2018-04-06]. https://cn.vuejs.org/v2/guide/. | | | | | | | | | | | | | | |
|  | 完 成 期 限： | | | | 2018年6月15日 | | | | | | |  | | |
|  | 指导教师签名： | | | |  | | | | | | |  | | |
|  | 专业负责人签名： | | | |  | | | | | | |  | | |
|  |  | | 2018 | 年 | | 1 | 月 | 5 | | 日 |  |  | | |

|  |  |
| --- | --- |
| 基于Nodejs的社团集成工具式管理系统 | |
| 摘　　要 | |
| 随着科技越来越深入人们的生活，网络课程在网上越来越多，人们学习编程技术的门槛越来越低，越来越多的人走进了代码的世界。我们希望将来的生活能与科技结合起来，使我们感受到原来科技的进步可以帮助到我们这么多事情。作为大学生，我们发现随着社团的不断发展与人数的增加，其管理难度与人员的信息量也同样日益变大，传统的人工管理模式已经不能满足现在的社团需求。传统的管理方式存在许多缺陷，比如人员信息数据维护效率低下，不易保管储存，容易丢失一些数据，文档等等。为了弥补以上缺点，我分析了当今社团管理状况，开发出一套通用的，灵活的专门为社团管理使用的工具式软件，帮助他们更好里管理人员报名，考核，通知等一系列工作。本课题就是以这一目标创立，并总结了各社团的需求，详细讲解了社团集成工具式管理系统的设计，分析与实现方法。  本社团管理系统主要包含六个轻量级的工具，来协助社团管理员管理成员，分别是：报名管理，考核系统，短信通知平台，值班表制作工具，学习讨论平台，邮件通知平台。因为它们各自功能的不同，所以我们选用了3种不同的技术去开发这六大工具来加快开发效率。结合着优秀的开源框架与 GitHub 上众多的开源项目，在网站开发的基础之上，最终设计并实现了这套社团管理这套集成工具式系统。我相信这种工具化，多样化，网络化的管理平台是大部分大学社团所需要的一种管理技术，特别是在人群范围广，深受大学生欢迎的学生社团，这种技术能帮助他们的管理上层让社团走在规范化，科技化，高效率的道路上。真正从单一型管理向综合型转变。 | |
| 关键词 | 社团；管理；全栈 |

|  |  |
| --- | --- |
| Nodejs-based Community Integrated Tool Management System | |
| ABSTRACT | |
| As technology becomes more and more in-depth in people's lives, more and more online courses are on the Internet, and the threshold for people to learn programming techniques is getting lower and lower, and more and more people are entering the world of code. We hope that life in the future can be combined with science and technology so that we can feel that the advances in science and technology can help us with so many things. As college students, we find that with the continuous development of the community and the increase in the number of people, the difficulty of management and the amount of information of the people are also becoming larger and larger, and the traditional manual management mode can no longer meet the needs of the current society. There are many defects in traditional management methods, such as low maintenance of personnel information data, difficulty in storage and storage, loss of some data, documents, and so on. In order to remedy the above shortcomings, I analyzed the management status of today's societies and developed a set of general-purpose, flexible, tool-based software specifically designed for the management of societies to help them better manage employee registration, evaluation, and notification. This topic is created with this goal, and summarizes the needs of each community. It explains in detail the design, analysis and implementation of the community integrated tool management system.  The community management system mainly includes six lightweight tools to assist the community administrators in managing members. They are: registration management, assessment system, SMS notification platform, duty watch production tool, learning discussion platform, and email notification platform. Because of their different functions, we chose three different technologies to develop these six tools to speed up development. In combination with the excellent open source framework and numerous open source projects on GitHub, based on the website development, the final design and implementation of this set of community management of this integrated tool system. I believe that this instrumentalized, diversified, and networked management platform is a management technology that most university societies need, especially in student groups that are popular among undergraduates. This technology can help them The management upper level allows the associations to walk on the path of standardization, technology, and high efficiency. Really change from single-type management to comprehensive type. | |
| KEY WORDS | Community; Management; Full stack |

目　　录

[摘　　要 I](#_Toc515296382)

[ABSTRACT II](#_Toc515296383)

[1 绪论 6](#_Toc515296384)

[1.1 课题研究的背景 6](#_Toc515296385)

[1.2 课题研究的意义 6](#_Toc515296386)

[1.3 课题研究的目标 7](#_Toc515296387)

[1.4 网站开发的发展于现状 7](#_Toc515296388)

[1.4.1 全栈的概念 7](#_Toc515296389)

[1.4.2 前端与后端的融合 8](#_Toc515296390)

[1.4.3 前端的趋势 8](#_Toc515296391)

[1.4.4 后端的趋势 9](#_Toc515296392)

[1.4.5 Docker 集装箱模式的盛行 10](#_Toc515296393)

[2 系统分析 13](#_Toc515296394)

[2.1 系统目标 13](#_Toc515296395)

[2.2 系统功能需求 13](#_Toc515296396)

[2.3 系统功能模块分析 14](#_Toc515296397)

[3 系统设计 16](#_Toc515296398)

[3.1 系统布局 16](#_Toc515296399)

[3.2 技术选型 16](#_Toc515296400)

[3.2.1 ThinkPHP3.2 框架 16](#_Toc515296401)

[3.2.2 Express 与 Pug 框架 19](#_Toc515296402)

[3.2.3 Vue 与 Koa 前后端分离 23](#_Toc515296403)

[4 系统实现 24](#_Toc515296404)

[4.1 开发过程 24](#_Toc515296405)

[4.1.1 ThinkPHP3.2 框架 24](#_Toc515296406)

[4.1.2 Express 与 Pug 框架模块 28](#_Toc515296407)

[4.1.3 Vue 与 Koa 前后端分离 30](#_Toc515296408)

[4.2 部署过程 31](#_Toc515296409)

[4.2.1 Docker 安装 31](#_Toc515296410)

[4.2.2 Docker 配置 31](#_Toc515296411)

[结束语 33](#_Toc515296412)

[致　　谢 34](#_Toc515296413)

[参考文献 35](#_Toc515296414)

[附录1 成品 GitHub 仓库展示 36](#_Toc515296415)

[附录2 名词解析和相关资源 37](#_Toc515296416)

# 绪论

目前国内大学社团的相较于以前的设施与活动明显提升很多，越来越多的社团开始用技术创新，开创了一些新奇的创业领域。小到开店卖水果奶茶，大到创立技术公司，通过优秀的想法和兴趣来赚取了自己人生的第一桶金。不过很多组织忽略了社团中管理人员的问题，如果能够利用工具化的管理技术或者说工具，就能让社团的管理更加合理化，同时，整个社团也能花更多的精力在其在的活动中去。

## 课题研究的背景

从60年代，中国开始改革开放之后，再到79年人人可以通过考试进入大学。以至现如今，普遍的9年义务教育的时代。大学一直都是国内学生统一的接受知识的环境，这样的环境也同时让社团快速成长起来，以至于大学生参加社团活动成了其密不可分的组成。在这几十年里，大学生社团的管理也从分散逐渐走向严格，有序。管理好一个社团，成了每个社团的重中之重。

同时，有些不规范的管理方案，或者当前的管理方案没有很好的继承下去，下一年的社团必将经历一次重创。人员流失，人心涣散，整个社团死气沉沉，这可不是一个好社团继续发展的氛围。

良好的管理能带给社团活力，成员信息不会丢失，处理事务高效无误，将其他时间真正花在社团发展建设上去。新时期的高校大学生，价值观，世界观趋于多元化，如今的社团管理方式，在现社团上的管理效果甚微，如何去利用如今的信息化技术，科学化的管理社团人员，避免出现重复的劳力，脑力，让社团人员拥有更多的动力去开创新的领域实在是迫在眉睫的任务。

## 课题研究的意义

本课题先是通过文献研究，了解到在国内大学中，部分社团会偶尔会出现崩塌以至于难以管理的情况，个人认为在科学管理方面，可以通过自身学习的计算机知识去开创一种方便的，高效的系统，让简易的网站去管理人员制度。这种想法正好与其他社团的想法一致。比如如何加强社团人员培养，人员交流方面，促进发展等等结合起来，共同形成统一，自由化的制度体系。这样不仅让每个社团保留自己的个性，也同样在未知领域有其他方案可以参考。构成新时期社团优秀的管理模式体系。

相较于传统的社团管理，工具化管理社团带来了多方面的创新：

1. 人员信息管理的变革。从之前的手动填写到现在的线上线下填写，并以电子方式的保存；

2. 值班表制定的改变。利用计算机编程算法，自动计算每节课人员的安排情况；

3. 社团课下学习的改变。使用线上统计的方式，方便地查看人员学习状况，促进成员学习；

4. 通知活动人员的方式改变。从以前的每条短信人员编辑群发，到利用大公司的可靠 API 一键群发短信与邮件（有必要的话）。

## 课题研究的目标

利用自己在大学中所学的知识，完成对社团管理系统的开发，实现如下目标：

1. 整体系统简易，对于学生几分钟就能够学会；

2. 因为系统为分离式，所以每套系统都有各自的信息的导入导出功能。

大致系统包含如下：

1. 报名系统——用于招新活动或比赛活动的报名，包括报名者信息填写、管理员信息收集等；

2. 考核系统——用于人员选拔的水平初试、学习效果检测等；

3. 通知系统——用于短信通知，如会议、面试等的通知；

4. 学习系统——沟通与学习的平台，用于学习经验交流、生活心得体会、学习总结、学习笔记、学习进度等的记录，同时推送感兴趣或热门学习方向等；

5. 值班系统——用于安排成员监管活动的小工具；

6. 最后通过 Docker 工具进行快速的部署。

## 网站开发的发展于现状

### 全栈的概念

全栈大多指的是全栈工程师，英文 Full Stack。指的是掌握多种开发调试等技能，并能利用这些技术独立完成产品的人。他们大多以网站开发为主，不仅对前端页面的设计与开发，也会后端接口的实现，更深层次的就是将网站内容搬移到 iOS 与 Android 等平台的 APP 上，实现狭义上的全栈开发。如果说到广义，那就还要加上产品的运维，调试，测试等等，甚至产品的宣传。

可以说，一个全栈工程师在公司里可以凭借一己之力，有效减少公司内部的沟通成本，人员的招聘成本。可以扛起这个部门系统架构，当公司业务调整的时候，每个方向的人力都可以做到有效的补充。

### 前端与后端的融合

说到前后的融合，这就不得不说到前端的一些历史，早在二十年前，前端并不存在，那时候网站开发，无论是功能还是界面设计都是由后端人员独自包揽。到后来，FLASH 可以用来做动画，用 Firework 切图，总之 Web 1.0 时代的网站建设两者并没有很好分离，使得工作流程十分混乱。但是随着 Web 2.0 的到来，网站内容越来越多，前后端逐渐分离，伴随而来的就是 JavaScript 再次的爆发性的发展，前端专注于与用户的交互，而后端则是专注数据的传输，服务的稳定提供。通过Restful API 等一些新兴协议，接口的定义更加规范，HTTP 传输的内容不再冗余。从此，前端开始出现了一些基于 JavaScript 的框架，如React， Angular 和 Vue 等，后端则更加复杂，加入了 Node 中间层对大量 API 请求进行分发，真正的后端处理安全性，可靠性与逻辑性，确保数据上的绝对安全。

### 前端的趋势

在之前讲了前端的来源，相信前端以后的路也十分好走。这一切都归功于 JavaScript 十年以来快速的发展，和 HTML5 的标准发布。2014年，第五代 HTML 标准发布。H5 是由浏览器厂商主导，与 W3C 合作制定的一整套 Web 应用规范，至今仍在不断补充新的草案。我们可以清晰的感受到这一系列规范背后隐含的领导者的勃勃雄心：占领所有屏幕。

从2010年开始出现的 Backbone、Angular.js 等前端 JavaScript 框架的出现。前端开始火了起来。

充分发挥 JavaScript 的本身优势，减少页面的重复刷新，只通过少量数据的更新来更新交互界面的数据。以 MVC，甚至之后更加流行的 MVVC 架构的前端框架支撑起了相当可靠的 SPA （Single Page Application，单页应用）。

以后的趋势也显现出来，一方面 React Native 等一系列框架入侵 Android 与 iOS 等原生 APP 之中，使一个模子的代码可以用在多种客户端中。另一方面 Hybird APP 的诞生，使得想 Weex 阿里的一站式框架得以发展，让 APP 的更新不再依赖每次应用商店的审核，而是通过内置的应用浏览器，对页面进行定期更新。常见的如：淘宝，天猫，京东，QQ等活动页，直接采用的是一些 H5 小页面。

### 后端的趋势

相较于前端，后端的任务则变得更为简单了一些。



前后端理解

互联网发展的早期，前端代码只是后端代码的一部分，大致流程如下：

1. 后端收到浏览器请求
2. 生成静态页面
3. 发送到浏览器

那时开发网站，一般采用的都是后端 MVC 模式

* Model（模型层）：提供/保存数据
* Controller（控制层）：数据处理，实现业务逻辑
* View（视图层）：展示数据，提供用户界面

前端只是后端 MVC 的 V。以 PHP 框架 Laravel 为例。



后端 MVC 中的 View 前端视图

由于 Ajax 技术的广泛应用，前端的应用终于可以独立出来，它们通过异步的请求获取少量数据，这些技术一开始广泛的应用于网页地图上。再到后来，乔布斯发布智能手机开始，很多人都意识到，这种异步获取数据能应用于许多领域上，比如 APP 数据的获取等等。

这两个原因，导致前端开发方式发生了根本的变化，前端不再是后端 MVC 中的 V，而是单独的一层。

前后端分离以后，他们之间通过接口通信进行双向数据传输。后端暴露出接口，前端消费后端提供的数据。后端接口一般是 REST 形式，前后端的通信协议一般是 HTTP。

同时，Node 在2009年诞生，这也就意味着本来只能跑在浏览器的 JavaScript 语言可以同样运行在服务器上，其中最大的意义就是前端工程师可以编写后端程序了。于是，前端工程师正慢慢转变为全栈工程师，一个人负责开发前端与后端，从数据库到 UI 的所有开发。

### Docker 集装箱模式的盛行

软件开发最大的麻烦事之一就是环境配置。开发环境与部署环境的环境不同，你怎么知道自家的软件，能在哪些机器跑起来？所以开发者必须知道两件事，操作系统是如何设置的，各种第三方库和组件要如何安装。只有当他们都被正确的运行起来，你所开发的程序才能如你所望的跑起来。举个例子，安装一个 Node 应用，计算机必须有 Node 引擎，还必须有各种依赖，可能还要配置环境变量。如果某些老旧的模块与当前环境不兼容，那就麻烦了。开发者常常会说："它在我的机器可以跑了"（It works on my machine），言下之意就是，其他机器很可能跑不了。环境配置如此麻烦，换一台机器，就要重来一次，旷日费时。很多人想到，能不能从根本上解决问题，软件可以带环境安装？也就是说，安装的时候，把原始环境一模一样地复制过来。

**虚拟机**

虚拟机（virtual machine）是系统安装级的一种解决方案。它可以在操作系统里运行另一个操作系统，程序不会有问题，虚拟机提供的环境难以甄别，就如同在那个环境里运行一般。用户可以通过虚拟机还原程序所需的开发环境。但是，这个方案有几个缺点。

资源占用多。虚拟机的启动首先就需要大量的系统资源消耗。哪怕你开发的程序只需要 1mb 你也需要消耗大量的操作系统资源与虚拟机资源。

冗余步骤多。创建虚拟机操作系统的过程就是非常浪费时间。加上其它登录等其它冗余操作。

启动慢。启动操作系统需要多久，启动虚拟机就需要多久。可能要等几分钟，应用程序才能真正运行。

**Linux 容器**

由于虚拟机存在这些缺点，Linux 发展出了另一种虚拟化技术：Linux 容器（Linux Containers，缩写为 LXC）。与此同时，windows 也同样也有自己的容器组件，甚至在近几年，windows 系统可以开启 linux 的容器按钮。通过应用商店的下载，可以直接在 windows 上运行 linux 的子系统。容器模拟一个近似完整的系统，而是对进程进行隔离。由于容器是进程级别的，相比虚拟机有很多优势。

启动快。容器里面的应用，直接就是底层系统的一个进程，而不是虚拟机内部的进程。

资源占用少。容器只占用需要的资源，不占用那些没有用到的资源，虚拟机由于是完整的操作系统，不可避免要占用所有资源。

体积小。容器只要包含用到的组件即可，而虚拟机是整个操作系统的打包，所以容器文件比虚拟机文件要小很多。

但是自家的容器毕竟不能和别家的融合，所以，在系统跨度上一直是个问题，windows 开发的程序就很难在 linux 上运行。总之，容器有点像轻量级的虚拟机，能够提供虚拟化的环境，但是成本开销小得多。

**Docker**

Docker 属于对各家容器的一种封装，提供简单易用的容器使用接口。它是目前最流行的快速部署容器解决方案。Docker 将应用程序与该程序的依赖，打包在一个文件里面。运行这个文件，就会生成一个虚拟容器。程序在这个虚拟容器里运行，就好像在真实的物理机上运行一样。有了 Docker，就不用担心环境问题。不过 Docker 还是没有很好解决不同种类系统的公用问题。毕竟这也是一难题。

总体来说，Docker 的接口相当简单，用户可以方便地创建和使用容器，把自己的应用放入容器。容器还可以进行版本管理、复制、分享、修改，就像管理普通的代码一样。

Docker 的主要用途，目前有三大类。

提供一次性的环境。比如，本地测试他人的软件、持续集成的时候提供单元测试和构建的环境。

提供弹性的云服务。因为 Docker 容器可以随开随关，很适合动态扩容和缩容。

组建微服务架构。通过多个容器，一台机器可以跑多个服务，因此在本机就可以模拟出微服务架构。

# 系统分析

本章主要探讨“基于Nodejs的大学生社团集成工具式管理系统”的需求分析，明确索要实现的各项功能，确定业务能达到的一定要求。

## 系统目标

“基于Nodejs的大学生社团集成工具式管理系统”旨在合理管理社团人力资源，强化人员管理意识，加强成员自主学习能力，为社团管理者提供一系列的全套服务。

本网站系统应实现以下主要业务目标：

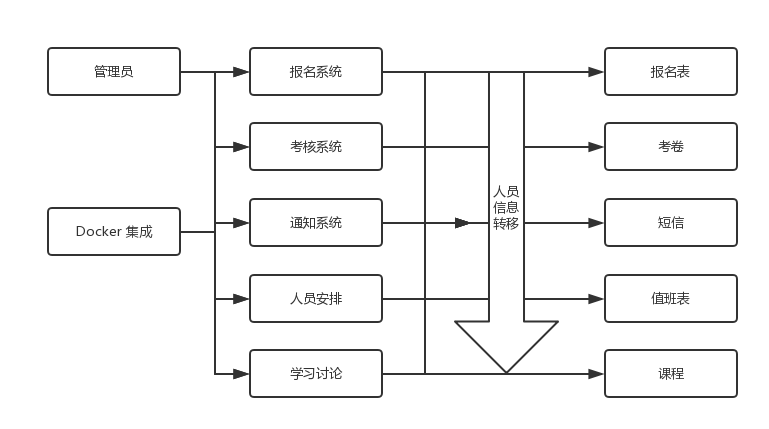
* 社团的管理者可以尽可能地通过这套系统管理整个社团，这种管理包括：人员的安置，人员的审核，人员的通知，人员的学习等等。
* 除了上述功能，我们还能提供简单高效的工具式分离，信息的导入导出方便，还有系统的部署卸载方便等等。

## 系统功能需求

当今的大学生社团，参加的人员越来越多，同时想法也越来越丰富。但与此同时，人员的增加也带来了一些问题。对于管理者来说，社团人数的增加无非增加了他们的工作量，他们需要花费更多的时间，人力，物力去管理社团人员的活动，组织上。分别在报名，考核，群发通知，让信息更准确的抵达到人员成为一种难点。缺少某种工具，可以快捷的统计人员，考核人员，通知人员，成为了社团管理员的急需。

面对以上问题，我开发了社团集成工具式系统，解决这些社团难题，开发六个网站型工具。致力于解决社团，特别是大社团人数多难于管理的难题。也同样希望今后有相似的技术，能真正地用技术解决每个社团管理上的问题。去除纸质化，让电子化更加方便，快速。

最后得出“大学生社团集成工具式管理系统”应该包括这些主要功能图



流程图

## 系统功能模块分析

经过以上分析，我总结出了六个系统模块，并打算利用 Docker 封装便于开发与部署：

报名系统：用于大学社团活动报名统计的项目。在整个大学生管理系统中，用于其新进成员的报名，已经数据的统计，另外可以方便地将其报名信息导入到其他系统中去，实现成员信息共享。

考核系统：用于大学社团成员考核统计分数的项目。简单的可以做到选择题的判题功能，填空题与简答题都可以利用人工进行评分。同样可以导入导出数据，做到信息的互通。管理员可以合理地分配老师，批卷人，学生的权限。并在留有排行用来分割成员成绩线。

通知平台：利用第三方收费 API 进行短信通知的基础。可以快速以表格人员名单的形式群发通知信息。同时加入多人管理的模式，可以让不同的账号也享有发送的权限。除了短信通知，文件等复杂内容的派发可以用邮件群发系统。

值班人员安排系统：利用常见的暴力算法，将收取的课表，按照学生有课无课对活动安排进行排序。采用尽可能多的安排人手，每个人都以相同的几率排到值班的可能。

学习讨论平台：网上课程有许多，通过对网上课程的学习，记录下课程进度，社团人员可以更好的比较自己与别人的差距。同时附加上论坛，可以方便人员讨论，学习。

Docker 部署：为了方便部署，采用 Docker 技术，可以使这个系统使用更加方便，持久。只需要几个命令就可以在任何一台服务器上运行。

# 系统设计

## 系统布局

考虑到本系统的复杂性，与基于每个社团的需求不一，所以将每个系统单独出来，各个完成。如有些社团仅需要报名与值班系统，我只要为你部署这两个系统即可，数据通过 CSV 表格形式进行人为传输。确保数据的可靠性与流动性。其次考虑到每个系统环境不一样，所以利用简单的 Docker 配置，可以一次性的安装多个系统。方便一些操作人员操作。

由于每个系统功能都有很大差别，所以系统设计每个模块都有不同的技术选型，一来，每个技术都有自己适合的领域，使开发方便又不容易出错。二来，尝试不同的技术，可以很好的察觉不同技术之间的差距，避免以后犯一些技术选型错误。三来，每个技术都可以展现自己不同时期学习的情况，也同样反应新技术更迭速度快，需要我们更快的理解新技术的诞生与发展趋势。

以下我会以技术选型的不同，分别介绍这六大系统模块。这三大技术，分别是：

* 以 PHP 和 MySQL 为基础的 ThinkPHP3.2 版本框架（报名系统与考核系统）
* 以 JavaScript 为基础的 Express 和 Pug 框架（群发短信平台，学习平台，值班系统）
* 以 JavaScript 为基础的，以前后端分离为思想的 Vue 前端框架与 Koa 后端接口框架（邮件群发平台）

## 技术选型

### ThinkPHP3.2 框架

ThinkPHP 是以 PHP 为底层的框架。相较于其他 Laravel，Yii，Zend 等大型框架，ThinkPHP 框架属于轻量型框架，没有什么特殊模块要求，底层运行的内容消耗也很低，不会出现空间和内存占用的瓶颈。并且它支持 Mysql、MsSQL、PgSQL、Sqlite、Oracle、Ibase、Mongo以及PDO等多种数据库和连接。对于这种一般型的项目足以。

1. 项目后端的搭建：

* 使用 PHP 语言的 ThinkPHP3.2 框架完成网站后端搭建；
* 使用 mysql 完成数据存储，通过 model 模块完成对 mysql 数据的构建使用 thinkphp 模板引擎完成页面创建渲染；
* 使用 ThinkPHP 的关联模型构建关系型模型。

1. 项目前端搭建：

* 使用 jQuery 和 Bootsrap 完成网站前端JS脚本和样式处理；
* 使用 jQuery.min.js 完成对账号以及选项的判断；
* 前后端的数据请求交互通过Ajax完成。

#### 报名管理系统

**设计**

项目相关设计 UI 图如下所示



报名首页



后台管理页



报名页

**详细功能**

本项目主要有活动活动的创建与 excel 导出功能，其次有对活动时间的控制，可以对活动进行修改，富文本的编辑与操作。

#### 考核系统

**设计**

项目相关设计 UI 图如下所示



项目首页



选择考试页

**详细功能**

本项目主要由试卷 exam 和文章 article 两大模型：

* 其中具有重要特色的功能是对试卷的添加与编辑和批改等功能；
* 其次在克服试卷的模型上我们做了很多尝试，最后用了稳定而不易出错的- thinkphp 自带关联模型；
* 对用户的考试成绩进行排序 rank (可以比较出学员的优异性)；
* 对考试时间的设定与修改；
* 还有对大量用户数据的批量处理；
* 对用户的权限处理。

### Express 与 Pug 框架

1. 项目后端的搭建：

* 使用 NodeJs 的 express 框架完成网站后端搭建；
* 使用 mongodb 完成数据存储，通过 mongoose 模块完成对 mongodb 数据的构建使用 pug 模板引擎完成页面创建渲染；
* 使用 Moment.js 格式化电影存储时间；
* 利用 alibaba.aliqin.fc.sms.num.send（短信发送）收费API作为发短信支持。

1. 项目前端搭建：

* 使用 jQuery 和 Bootsrap 完成网站前端JS脚本和样式处理；
* 使用 jQuery.min.js 完成对账号以及选项的判断；
* 前后端的数据请求交互通过 Ajax 完成；
* 前端的页面渲染通过 PUG 最新插件完成；
* 跨域的数据请求交互通过 Ajax 中的 jsonp 完成。

#### 短信群发平台

**设计**

项目相关设计 UI 图如下所示



项目首页



模板信息填写页



信息群发页

**详细功能**

本项目主要由 CSV 名单导入和短信群 smsMass 发两大功能：

* 其中具有重要特色的功能是对权限的控制上附加了对申请 key 的操作；
* 其次在短信模板上可以自己相应的信息,进行合理的增删改查与默认的功能；
* 对短信群发的信息有日志的记录log；
* 对用户的权限处理。

#### 排值班系统

**设计**

项目相关设计 UI 图如下所示



首页

**详细功能**

排值班系统是一套集成C/C++暴力算法排序的，集成 xlsx 的导入与导出制作的一套系统。通过人编写的算法，对成员的值班表进行进行智能的分析，以不重复安排一个人的前提下，尽量将每个成员分配上去，以保证公平，公正。但是最主要的是减少人工排课的工作量，方便管理员的导入导出与使用。

#### 学习平台

**设计**

项目相关设计 UI 图如下所示



课程页



手记页

**详细功能**

本项目由学习进度 course 和文章发表 article 两大功能：

* 其中具有重要特色的功能是慕课网信息的爬取与使用 spider，利用superagent插件；
* 其次具有简单的用户登录和注册 user ，用户的头像上传；
* 对用户的学习进度进行排序 rank (可以比较出学员的积极性)；
* 对课程 course 的搜索 search；
* 还有对每个列表页面进行分页 page 处理；
* 访客次数统计 pv。

### Vue 与 Koa 前后端分离

#### 邮件发送平台

**设计**

项目相关设计 UI 图如下所示



发送页

**详细功能**

本项目利用 nodemailer 和 QQ 邮箱的 STMP 利用，加上独特的域名配置，可以实现简单的 HTML 格式邮件发送。

# 系统实现

系统实现上，我会同样以三大框架大分，每个模块小分的形式，介绍各个模块在实现上的难点与代码解析。并且以开发和部署两大部分粗略讲解。在讲解开发的过程中，我也会逐个分析每个语言和框架的有点与缺点，为什么时代的前进，后面的框架会越来越吸引人去开发。

## 开发过程

### ThinkPHP3.2 框架

首先使用 ThinkPHP 框架之前，你需要安装配置好 PHP 5.6 之上的环境。PHP 环境的配置相较于 Java 和 Nodejs 与略显麻烦的，但是和 C# 相比就是速度会明显快很多。PHP 的安装配置与其他语言都不太一样，可以直接傻瓜式安装，不论在 Windows 还是 Linux 都需要一个简单的步骤，添加到环境变量。不过如果你是使用 LAMP 等一种开发包去安装，那会十分简单。

一套开发环境 PHP Apache 和 Mysql 配置完毕后，你就可以开始用一些框架开发网站了。这里我使用的是 ThinkPHP，从官方下载框架并解压，得到一下框架目录结构：

├── index.php 项目入口文件  
├── Application ThinkPHP 后端 MVC 文件目录  
│ ├── Common 公共函数目录  
│ ├── Home Home目录  
│ ├── Manager 后台目录  
│ ├── Manager\_Detail 后台beta目录  
│ ├── README.md 框架README文件  
│ └── index   
├── db 供参考的数据库数据  
├── ThinkPHP 框架系统目录（可以部署在非 web 目录下面）  
├── public 静态文件目录  
│ ├── assets 后台样式  
│ ├── css 样式目录  
│ ├── fonts 字体目录  
│ ├── images 静态图片目录  
│ ├── js JS 脚本目录  
│ └favicon.png favicon  
├── README.md  
└── package.json

Application 目录下除了 Common 目录是用来存放一些公共配置文件以后，其他目录都是以如下目录结构排列，以代表 ThinkPHP 框架中一定的规范

├─Common 公共文件  
├─Conf 这个模块的单独配置  
├─Controller 控制器模块  
├─Model 模型模块  
└─View 视图模块

Common 与 Conf 就不讲了，这类框架与许多框架都十分类似，最重要的一点就是都是 MVC 框架，也就是目录结构中的 Model，Controller，View 三个目录。

先解释下 Model 文件夹，ThinkPHP 是如何与 MySQL 等这类数据库链接数据的呢，Model 的中的 Model.class.php 文件中的写法就很好的解释原因。

├─MemberModel.class.php 报名人模块  
├─ProjectModel.class.php 报名项目模块  
└─index.html 空白文件

// MemberModel.class.php File Content  
namespace Home\Model;  
use Think\Model;  
class MemberModel extends Model{  
 protected $tableName = 'member'; // 对象的数据表  
 protected $\_validate = array( /\* 省略... \*/ );  
 public function insertM($pid, /\*省略...\*/ )  
 { /\* 省略... \*/ }  
}

在 ThinkPHP 中，如果表的形式很复杂，可以使用这种模型文件定义，通过定义其$tableName 表名，可以很方便的操控其表中的数据。当然如果你的模型或者说表结构很简单，你完全可以在控制器直接写：

$User = new \Home\Model\UserModel();  
$Info = new \Admin\Model\InfoModel();  
// 带参数实例化  
$New = new \Home\Model\NewModel('blog','think\_',$connection);

在 ThinkPHP 中，可以无需进行任何模型定义。只有在需要封装单独的业务逻辑的时候，模型类才是必须被定义的，因此 ThinkPHP 在模型上有很多的灵活和方便性，让你无需因为表太多而烦恼。

在 MVC 中，除了需要很好的操控数据库那一环，还需要一环去操控视图，也就是这里的 View 视图层。为了不出现代码冗余，通常会利用 HTML 模板引擎去渲染。

├─<Directory> 对应控制器的模板文件  
├─layout.html 模板布局  
└─index.html 空白文件

ThinkPHP的模板引擎内置了布局模板功能支持，可以方便的实现模板布局以及布局嵌套功能。layout.html 文件，默认开启 layout 的情况下，是一个父容器。

读取 layout 模板之后，会再解析 /.html 模板文件，并把解析后的内容替换到 layout 布局模板文件的 {CONTENT} 特定字符串。

最后就是控制器，这个 MVC 中，最重要的一环，用来控制模板渲染与数据交互的逻辑。在 ThinkPHP 中，它几乎可以与路由相结合，做到很简单的访问。一般来说，ThinkPHP的控制器是一个类，而操作则是控制器类的一个公共方法。

通过 display() 等方法，可以让视图与控制器联系起来，再加上 Model 层的操作，ThinkPHP 就可以做到一般网站开发能做的所有事情。

#### 报名管理系统模块

报名管理系统是一个相较于简单的系统，涉及到数据库中，人员 Member 表与活动 Project 表的增删改查。

其中主要的难点还是整个后台的管理上，比如说只有管理员才能进入后台，否则出现跳转，让他以管理员形式登录。

还有就是活动的创建中，需要添加图片到后台，这里利用的是 ThinkPHP 自带的图片文件上传 API 来实现对文件上传保存的功能。

这个报名模块只有一个管理者。管理者可以通过后台 Manager\_Detail.php 去管理创建活动，并且对报名者的信息进行增删改查。

#### 考核系统模块

考核系统相较于报名系统就复杂许多，首先，它有多种数据需要存储，这也就意味着，你需要建立多张表，同时，每张表之间有着一对多或者一对一的关系，所以，你需要将某些表进行关联。

难点二，你不仅需要做到提交试卷的功能，制作试卷你也需要考虑到出卷人是怎么设计试卷类型的。经过我细致的思考，我总结出一下一套方案去在网页端设计提交出一份试卷。也是处于简单的逻辑考虑。

首先你需要定义一个考试名称吧，还有制定考试的开始时间与结束时间，再加上考试的题型组别，如果可以的话，你可以为你的考试添加上一张个性化的图片，来吸引顾客的目光。一张华丽的图片在设计中是有必要放在某些主题上，充当修饰效果的。

然后，已经进入了这场考试中的组别题型设计，你可以为你的选择题添加一些题目，然后再为你的题目添加几个选项，每个选项的添加都是利用 Ajax 技术去无刷新添加的，可以避免一些无用的数据获取。

在你添加的同时，你可以随时返回题组，添加新的题组，或者在讨论题上新增几个讨论，在选择题中，添加默认的正确答案。最重要的是，当你完成整组试卷的制作，如果你发现问题，你可以在第一时间重新返回题库，进行多次修改。

与此同时，除了选择题，有机器可以帮你判题打分，但是填空题，讨论题是没有这么智能的，所以你需要安排批卷人去审核每套试卷。指定的批卷人可以给定的权限，对每道题进行批改，而且是可以同时批改，可以节约大量时间的。

最后你的试卷的结果分会出现在排行中，你的最终分数将与别人一比高低。

其中试卷的一对多是一个很棘手的问题，当你要获取试卷时，你只能获取一张二维的表格，但是试卷的这个数据结构明显是个树结构，你当然可以想过通过循环去连续获取，不过你根本不知道这张试卷会有多少题型，会有多少题组，更不知道会有多少选择题，每个选择题有多少个选项。所以我们这里利用的是ThinkPHP 提供的关联模型，只需要在 Model 层定义一个简单的 Model 继承自 RelationModel，并在其属性中写上相应关联的表名即可。

总的来说，利用 MySql 建立 user 表，用来存储导入用户数据，exam 表用来建立一张试卷，group 表用来组织一张试卷中的题型，question 表用来表示试卷中的问题，answer 表用来表示选择题中的选项，record 表用来记录答题者的答题内容情况，finish 表用来记录答题者完成这整张试卷的时间与计算分数。

#### 总结

php 与 一些 php 框架搭配的网站开发模式虽然在易用性上大大胜出，学习成本也很低，同时 php 也是基于 C++ 开发的语言，其速度，效率也毋庸置疑，是初学者开始学习编程的良好工具。但是它也有很多不足，比如在构建上你需要同时拥有 Apache 软件或者 Nginx 才能去调试网站，局限性也是突出的一点，php 只能开发一些网站，而手机app，应用，还是前端都是 JavaScript ，所以你必须学会一些前端知识，才能真正地去开发一些网站，最后，也是最关键的一点，虽然有 Composer 包管理工具，但 php 本身并不包含此工具，在代码迁移，或者代码构建时会浪费很多时间和空间。

### Express 与 Pug 框架模块

相较于 PHP 语言，Express 与 Pug 都是基于 Nodejs 的，所以在安装 Nodejs 上，会方便许多。其次，nodejs 本身自带了 npm 包管理器，在安装第三方库的时候会十分方便。服务器软件的选择上，不用担心，nodejs本身就自带服务器 API，所以你无需安装一些 Apache 或者 Nginx 软件。数据库方面，你完成可以不用数据库，采用 JSON 文本也可以实现简单的数据库的操作。如果你的项目比较大型，Mysql 等关系型数据库和 Mongodb 非关系型数据库都是非常好的选择。

如果你是一名前端开发者，学习 Nodejs 的成本不是很高，你可以在一天内能搭建出一套网站。这也引申除了全栈工程师为什么会在 JavaScript 程序员中特别多的原因。

Express.js 框架是一个基于 Nodejs 的 Web 应用程序框架，发布于 2010年11月16日，它被称为 nodejs 中标准的服务器框架。由 TJ 大神开发，并迅速成为流行的 Node 服务器框架。

Pug.js 是个高性能的模板引擎，受 Haml 的影响很大，前身是 Jade.js 由于商标问题改为 Pug.js。由于它们共同都是 npm 包，也是网站建设的的贡献者，所以两者都很好安装与互用。

只需要简单的几句可以方便地使用 pug 语法来编写 Html 模板。

├─controllers 控制器  
├─models 模型  
├─schemas 表结构  
└─views 视图层  
 ├─includes 视图中的小插件  
 └─pages 视图中的页面

用 Expressjs 搭建的网站目录结构也类似 PHP，采用 MVC 的框架，不过不同的是，我这里用的是非关系型数据库 Mongo，利用 schema 就可以在应用初始化时帮你创建好一些列表结构，这相当于你的数据库与应用也绑定在了一块，十分方便。

#### 短信群发平台模块

短信群发平台主要是以发送短信为核心。不过发送一个短信并没有你用手机发送这么简单，你需要一个类似通信服务商发送短信的 API，只要得知对方的手机号，和发送内容就可以发送。这需要代码去完成，所以我利用了阿里大于的短信 API 去完成这一部分内容。他们提供了一整套的 API 类似于通信服务商一样的内容，你可以语音，短信，验证码等等一系列内容。

下载阿里大于关于 Nodejs 的 SDK，稍加改装就可以实现简单的发送短信的功能。由于要写的群发短信的功能，这里就要考虑到一些语言会碰到的阻塞性问题，发送一条短信，你需要等阿里大于的服务器返回的结果，然后再去发送，这样会慢许多，通常用异步去解决这类问题。不过 Nodejs 几乎不需要考虑这类情况，Nodejs 本身就是单线程异步的，利用 JavaScript 的回调函数或者 Promise 写法，或者更高级的 Async/await 写法，就可以流畅地去使用异步与同步。

用户权限的分配上，这个系统大致有三类角色，分别是接受短信用户，发送信息者，管理员。

接受短信用户：主要从报名系统模块中抽取需要发送短信的人群，作为接收者。

发送信息者：可以向管理员索要发送短信权限，并发送短信。同时可以向容器内导入数据。

管理员：权力最大，可以同意或否决发送信息者的请求。自已同时可以添加人员信息与发送短信的功能。

所以，我在 mongodb 数据库中建立了以下表：人员表，用户表，申请key表，短信发送模板表，会话表。

#### 排值班系统模块

排值班系统主要是用学生的课表，根据有课无课，或者说有空与没空去安排人员。举一个比较简单的例子，一个社团需要招新了，他们已经有很多成员，成员们也愿意将他们的没课的时间去参加招新的时间上去。一般招新的时间点无非是上午第一大节，第二大节，中午，下午的第三大节，第四大节。所以具体算法就是，先将大家的无课表，记录在一个统一的课表上，然后根据配置，每个值班时间点需要多少人手。按照多劳少排的规则去安排每个人。

#### 学习平台模块

这里的学习平台主要是为了督促学习用，我在各个平台抓取每个网站的课程信息，写在自己的 Mongo 数据库中，人员信息从报名系统中提取，两者结合就可以实现学习进度情况。除了有爬虫的功能，同时也加入了 NodeBB 论坛，用于学员之类的讨论。社团的通知等等。

除了 NodeBB 论坛是使用第三方的论文之外，其余都是由自己开发实现，因此建了如下几个表：article 表用来存放文章数据，courses 表用来存放网络课程数据， sessions 表用来存放会话，userapis 表用来引入用户信息，user 表用来存放用户信息。

### Vue 与 Koa 前后端分离

之前讲的 MVC 不管是 PHP 实现还是 Nodejs 实现，都是 Web1.0 的思想，而真正需要发展的，而且目前大公司都在用的都是前后端分离的。后端不用操心前端的事，前端不用管数据安全，服务是如何运行。他们各司其职，它们之间唯一的联系就是 API，后端提供 API 的使用方法，前端用这套 API 去调用即可。一切就变得简单许多，你甚至可以将这套 API 用在各个地方，比如部分开放给公众，用于公开数据，利用大数据的分析，部分可以与手机APP，微信小程序共用。

其中 Vue.js 是国人开发的一款足以与 FaceBook 开发的 React 媲美的前端框架，它简单易于上手，也同样具有庞大的第三方库的支持，甚至有 Gitlab 等一些著名的项目都是基于 Vue 创建的。

Koa.js 是 Tj 大神的另一开山大作，从 2.0 开始 Koa 放弃了老旧的 co.js 依赖，转而支持新技术来保持它异步的功能。你可以看到 Koa2.0 代码十分简单，短小，精悍，它只保留属于服务器的功能，其他功能都通过第三方库来扩建。使开发变得简单高效。

├─client 前端  
│ ├─dist 生产文件  
│ └─src 核心文件  
└─server 后端  
 ├─dist 生产文件  
 └─src 核心文件

可以看到在目录构建上，已经将其分别前端与后端两部分，这样能很好的专注于某一部分的开发。

#### 邮件发送平台模块

邮件发送平台主要用于发送邮件，只靠前端是不能完成的，前端只是提供友好的界面，将用户的数据包装发送给后端，后端接收之后，将数据匹配至相应的 API，我这里利用的是 nodemailer 第三方插件，通过邮箱的配置，就可以轻松的发送邮件。

## 部署过程

### Docker 安装

在 Linux 中安装 Docker 很方便，只要下载对应脚本就可以自动进行安装，同时为了方便使用，在 docker 命令之前不用加上 sudo，你可以将 docker 加入相应的用户组。

此时，Docker 已安装好，如果需要暂停或者启动只需用 systemctl 来启用它即可。当然为了用于 docker-compose 你还需要安装 docker-compose 来实现一键化部署。

### Docker 配置

Docker 把应用程序以及其依赖打包在 image 文件里面。只有通过这个文件才能生成 Docker 容器。image 文件可以看作是容器的模板，Docker 根据 image 文件生成容器的实例，而且同一个 image 文件可以生成多个运行实例。

image 是一个二进制文件。在实际开发中，一个 image 文件往往通过集成另一个 image 文件，加上一些个性化的设置而生成。比如在这个项目中，就是继承了 php:5.6-apache 的 image，往里面加上我的项目文件，就可以无数据库的运行我的 php 单个项目了。

然而有时候你需要和多个容器协作，才能正常运行一个实例。所以，这里我为了完整运行一个网站，我引用了官方的 MySQL 镜像，利用 Docker 的链接，将两者容器用 link 关键字进行数据链接，这样一个完成的需要数据库的网站的部署好了。

接下来就是 docker 配置，具体的配置如下：

# 基于 php 这个版本的容器  
FROM php:5.6-apache  
# 工作目录设定  
WORKDIR /var/www/html  
# 安装 php 额外的扩展  
RUN docker-php-ext-install pdo pdo\_mysql  
# 开放 80 端口  
EXPOSE 80  
# 最后执行 apache  
CMD apache2-foreground

DockerFile 文件是用来部署生成镜像 image 的文件，用来可以生成运行相应的容器 container。为了方便容器之间的互相调用，一般采用数据库容器与服务器分离的方式来写 DockerFile 文件，然后再用 Compose 这个定义和运行多个 Docker 应用程序的工具，来使整个网站运行起来。这其中比较重要的就是 link 关键字，web 页容器必须找到 mysql 的容器名字进行链接。具体的配置文件如下：

version: '2'  
services:  
 mysql:  
 image: mysql:5.7  
 container\_name: mysql\_es  
 environment:  
 MYSQL\_ROOT\_PASSWORD: 123456  
 MYSQL\_DATABASE: es  
 volumes:  
 - ./db/es.sql:/docker-entrypoint-initdb.d/1.es.sql  
 web:  
 build: .  
 container\_name: web\_es  
 links:  
 - mysql  
 ports:  
 - "8080:80"  
 volumes:  
 - ./:/var/www/html

结束语

本课题利用VsCode，Vim等一系列IDE工具完成了这套前端网站的设计，完成了以下工作。

（1）研究了如何用VsCode的第三方插件，快速编辑开发程序，比如Sublime Text Keyboard插件拥有优秀的快捷键习惯方式，Vuter的Vue语法高亮，还有基于Ruby的windows下linux命令行工具，可以帮我更好的使用命令行；

（2）对于koa框架的jsonp格式的返回进行研究，利用相关配置即可返回规范的jsonp格式的数据形式；

（3）对vue的脚手架工具vue-cli进行阅读，同时写出了适用于自己的一套vue脚手架模板；

（4）对网络，及linux的命令与Docker进行对比，还是有许多相通之处，所以，利用linux运行docker容器还是很方便的。

本课题在虽然完成了社团管理的工具开发，也同样对其大规模的应用进行有关测试，不过还是因为时间与精力问题，留有以下问题：

（1）邮件与短信通知平台没有很好的 结合，邮件通知还是简单的利用API界面。

（2）用户数据的转移还是很混乱，没有做很好的统一，希望以后可以通过方便的流处理与传递数据。

（3）Docker 的配置可以再简化，做到一个命令就可以按需部署，而非一个一个的部署。

致　　谢

感谢指导老师

本系统虽然有多部分构成，但是有些系统，在老师的指导下，我才得以完成。老师提示的建议以及给与学生的帮助，更好的帮助我在系统的测试中没有困扰。老师的每个建议和意见都对我启发很大。在完成毕业设计和毕业论文这几个月的时间里，从老师的身上，我学到了很多做事的方法，受益匪浅。

感谢成员

因为整个系统很庞大，报名系统，考核系统都需要大量的人员测试，所以成员们都成了测试员，帮助我去测试系统的 BUG 并且修复它。还有就是我的室友，帮助我完成了用 C 语言完成了排值班的暴力排序算法，我得以将它用在了我的应用上。最后还是感谢我加入的社团，让我的系统得以在社团中有小部分应用。

感谢开源社区

这套系统最终能够完成，还是归功于开源社区的发展。这里首先也要感谢 Nodejs 与 PHP，他们本身的开源，也同样助力与其的第三方优秀插件的开发。其次就是要感谢 NPM 这个 Nodejs 库，里面有上百万的第三方库能免费开源的让我们开发者使用，使我们在平日开发中，减少大量的劳动量。最后还是要感谢 GitHub 一直以来让开源免费，私有收费的策略。这使得许多新手得以使用和接触开源社区宝贵资源，为他们入门铺平的道路。

感谢互联网

现在日益发展的互联网，是我5年前都不敢想象的，它获取知识如此之快，使得一些没有上过学的人，也能从互联网上汲取知识，加上不断地努力一样可以成为优秀的人才。这里我特别的感谢 Google 在搜索上给予我的帮助，Google 的搜索给我的都是一些新鲜又专一的搜索结果，其次，Stackoverflow 也是我是要感谢的，它帮助开发者把编程的难点的提问都汇集起来，而且与国外的人交流，你似乎获得的更多。

参考文献

1. 介绍—Vue.js[M/OL] [2018-04-06]. https://cn.vuejs.org/v2/guide/.
2. 阮一峰-前端开发的历史和趋势[M/OL] [2017-05-26]. https://github.com/ruanyf/jstraining/blob/master/docs/history.md.
3. Docker: a Software as a Service, Operating System-Level Virtualization Framework[M/OL].2014-07-21. http://journal.code4lib.org/articles/9669?utm\_source=feedburner&utm\_medium=feed&utm\_campaign=Feed%3A+c4lj+.
4. 阮一峰-Docker 入门教程[M/OL]. 2018-02-09. http://www.ruanyifeng.com/blog/2018/02/docker-tutorial.html.
5. 张英. 微服务: 开创图书馆服务的“蓝海”[J]. 图书馆建设. 2011 (7): 51–53.
6. ThinkPHP3.2 完全开发手册[M/OL]. 2013-01-01. http://document.thinkphp.cn/manual\_3\_2.html#environment.
7. WILSON J. Node. js 8 the Right Way: Practical, Server-side Javascript that Scales[M]. [S.l.]:Pragmatic Bookshelf, 2018.
8. MARDAN A. Express. js Guide: The Comprehensive Book on Express. js[M]. [S.l.]: AzatMardan, 2014.

附录1 成品 GitHub 仓库展示

• 报名管理系统：https://github.com/Lanseria/signinsys-php

• 考核系统：https://github.com/Lanseria/exam-sys

• 短信群发平台：https://github.com/Lanseria/sms-mass-platform

• 社团排值班工具及系统：https://github.com/Lanseria/arrageclasssys

• 邮件群发系统：https://github.com/Lanseria/mail-system

• 学习进度平台：https://github.com/Lanseria/learn-site

附录2 名词解析和相关资源

**概念**

MVVM: http://baike.baidu.com/view/3507915.htm

**Docker**

Docker Documentation: https://docs.docker.com/

What is Docker: https://www.docker.com/what-docker

**JavaScript框架**

Angular.js: https://angular.io/

React: https://reactjs.org/

Vue.js: https://cn.vuejs.org/

Node.js: https://nodejs.org/en/

Express.js: http://expressjs.com/

**JavaScript规范**

ECMAScript: https://zh.wikipedia.org/zh-cn/ECMAScript

ESLint: https://eslint.org/

**工具**

Gulp: https://www.gulpjs.com.cn/

Webpack: https://webpack.js.org/

**平台**

阿里大于：https://dayu.aliyun.com/