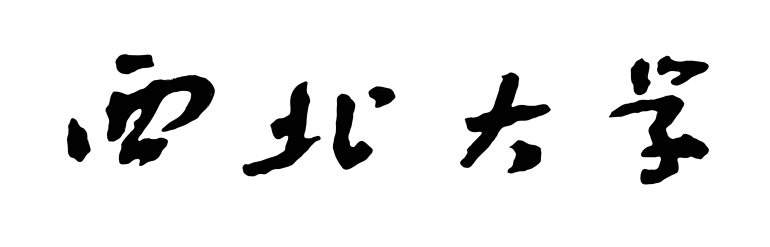
**** 

**本科毕业论文（设计）**

**题目： 书籍管理系统的设计与实现**

**学生姓名 刘佳宁**

**学 号 2013118111**

**指导教师 卢燕宁**

**院 系 信息学院**

**专 业 软件工程**

**年 级 2013级**

**教务处制**

**年 月**

诚信声明

本人郑重声明：本人所呈交的毕业论文（设计），是在导师的指导下独立进行研究所取得的成果。毕业论文（设计）中凡引用他人已经发表或未发表的成果、数据、观点等，均已明确注明出处。除文中已经注明引用的内容外，不包含任何其他个人或集体已经发表或在网上发表的论文。

特此声明。

论文作者签名：

日 期： 年 月 日

目 录

1.[摘要 1](#_Toc449629205)

2.[Abstract 2](#_Toc449629206)

3.[引言 3](#_Toc449629207)

3.[1 背景与意义 3](#_Toc449629208)

3.[2 技术栈 3](#_Toc449629209)

3.[3 研究内容 3](#_Toc449629210)

4.[技术选型说明 4](#_Toc449629213)

4.[1 代码版本控制 git 4](#_Toc449629214)

4.[2 包管理工具 npm 4](#_Toc449629215)

4.[3 前段模块化构建工具 webpack 4](#_Toc449629216)

4.[4 轻量高效的前端模块化解决方案 Vue.js 4](#_Toc449629217)

4.[5 基于 Vue-router 实现 SPA 4](#_Toc449629218)

4.[6 服务端语言 PHP 5](#_Toc449629219)

4.[7 数据库 Mysql 5](#_Toc449629220)

5.[系统需求分析 6](#_Toc449629223)

5.[1 用户分析 6](#_Toc449629224)

5.[2 用例分析 7](#_Toc449629225)

6.[系统设计 8](#_Toc449629226)

6.[1 系统流程设计 8](#_Toc449629227)

6.[2 数据库设计 9](#_Toc449629228)

6.[3 UI设计 1](#_Toc449629229)0

7.[代码实现 1](#_Toc449629230)2

7.[1 开发模式 1](#_Toc449629231)2

7.[2 开发环境 1](#_Toc449629235)2

[7.2.1 前端开发环境 1](#_Toc449629236)2

[7.2.1 后端开发环境 1](#_Toc449629236)3

1. [3功能模块的实现 1](#_Toc449629235)3

[7.2.1 单页路由的实现与管理 1](#_Toc449629236)3

[7.2.1 注册/登录模块 1](#_Toc449629236)4

[7.2.1 书籍信息管理模块 15](#_Toc449629236)

[7.2.1 信息管理模块 1](#_Toc449629236)6

8.[系统测试 1](#_Toc449629239)8

8.[1 测试环境 1](#_Toc449629240)8

8.[2 测试结果 1](#_Toc449629241)8

9.[总结与展望 2](#_Toc449629242)0

10.[参考文献 2](#_Toc449629243)1

# **1. 摘要**

随着近几年来互联网的高速发展，人与人的距离和沟通成本都大大减少，越来越多的新经济形式都应运而生，其中最值得一提的便是共享经济。比较成功的案例层出不穷，比如顺风车和共享单车的运营模式，不仅节省了用户的生活成本，同时也在传播一种节约共享的消费价值观。  
 以学生、家长为主体的用户人群，他们都有被教育或教育的需求，从而都会产生买书的需求。对于个人消费者而言，书是一种消耗品，大多数的书读过一次之后就闲置在了那里，造成了资源的浪费。  
 本次课题研究，秉承着“充分利用书籍资源”的理念，旨在实现一个共享书籍的 B/S 模式下的平台。  
 此系统采用前后端分离的开发模式。前端采用组件化的开发方案，使用 Vue.js 配合构建工具 webpack 实现由 vue 文件构成的单文件组件系统，使用 vue-router 实现单页应用（Simple Page Application）；后端使用 php ，数据库方面选用 mysql。  
 考虑到近几年移动互联的发展，为了让用户在移动端有更好的使用体验，系统采用媒介查询和 Bootstrap 等方案，实现了适配不同设备屏幕尺寸的响应式设计。

关键词： 共享书籍，前后端分离，Vue.js，单页应用，组件化

# **2. ABSTRACT**

With the rapid development of Internet in the recent years, the distance and cost of communication are both greatly reduced. In the meantime, a growing number of new economy form arose, one of the most worth mentioning is the Sharing Economy, such as DiDi and Ofo, under the help of their operation mode, this kind of company not only save the cost of living for users, but also spread the values which is shared and saving consumption.  
 In the case of students and parents, they all have the needs of being educated or educating, which creates the demand of books. Generally speaking, for individual consumers, books are consumables, and most books sit idle after being read, creating a waste of resources.  
 The research on this topic aims to build a book-shared platform under the B/S mode with the ideal of "making full use of the book resources".  
This project adopts the development pattern of separation between the front-end and back-end.  
 The front-end adopts the ideal of componentized development solutions, using Vue.js and building tools called webpack to implement a single vue-suffixed file component system; the back-end uses PHP and the database uses Mysql.  
Considering the development of mobile communications in recent years, in order to provide a better user experience in mobile terminal, the project uses media query and Bootstrap achieving a responsive design to fit different screen sizes.

****keywords:**** Sharing Economy, the separation of front-end and back-end, Vue.js, componentized, Simple Page Application

# **3. 引言**

## 3.1 背景与意义

现代社会，共享经济盛行，从顺风车到共享单车，越来越多的互联网产品从“充分利用已有资源”的角度出发，受到了越来越多人的信赖。而本次的设计课题，旨在设计并实现一个共享书籍的平台。  
 对于大多数人来说，读完的书就闲置在家里，成为“废品”。在这个共享书籍的平台，用户可以发布自己的闲置书籍信息或者想要交换/购买的书籍的信息，让有匹配需求的用户可以互相取得联系，从而实现书籍的共享。

## 3.2 技术栈

本项目采用 B/S 架构，以 Apache 作为 Web 服务器；使用 PHP 实现后端逻辑；使用 Mysql 关系型数据库存储用户信息、书籍信息等数据；前端使用 Vue.js 和 Vue-router，配合使用前端构建工具 Webpack 实现一个由 vue 单文件组件构成的 SPA（Simple Page Application） 组件系统。

## 3.3 研究内容

本次课题研究从用户的根本需求出发，目的是构建一个简单易用的共享书籍系统，从人机关系的角度有以下研究内容：

(1).分析用户的核心需求

(2).建立起系统与用户之间的友好关系

(3).设计一个简洁友好的用户界面

(4).在技术层面，提出以下研究内容：

(5).代码的版本控制

(6).数据库的设计，用最少的字段存储用户的相关信息

(7).前后端分离模式下开发环境的搭建

(8).利用构建工具实现前端开发的自动化

(9).使用 Vue.js 轻量高效组件化方案的实现

(10).使用 Vue-router 构建的单页应用，以及前端数据流的控制

# **4. 技术选型说明**

## 4.1 代码版本控制 Git

项目选用 Git + Github 的代码版本控制系统，通过其分布式的特点，可以快速并高效的管理一个项目。

## 4.2 包管理工具 npm

项目采用集成于 Node 环境下的包管理工具 npm (node package manager)，方便开发者可以仅仅用几条命令就能够便捷的管理项目所需要的第三方包模块。

## 4.3 前端模块化构建工具 Webpack

Webpack 是一款用于实现前端模块化的开源方案，通过安装不同的插件可以根据开发者的需要实现不同的构建功能，配合插件 webpack-dev-server 可以实现开发过程中的自动编译以及热重载，大大提高了开发效率。

## 4.4 轻量的前端模块化解决方案 Vue.js

Vue 是一款开源的 View 层框架，由于其友好的 API 设计以及更容易被接受的变成方式，近几年的发展势头十分迅猛。区别于传统的 DOM 编程，Vue 旨在提供一种指令时编程模式下的轻量高效的前端模块化解决方案。

## 4.5 基于 Vue-router 实现的 SPA

Vue-router 是一款 Vue 官方提供的前端路由解决方案，通过定义配置文件的路由设置，可以实现不同路径下 DOM 节点的刷新和替换。本项目使用 vue-router 实现一个单页应用（SPA: Simple Page Application），区别与传统的网页应用，单页应用不仅可以做到前后端的职责分离、架构清晰，还可以给用户提供一个良好的交互体验，前端只是进行局部渲染，避免了多余的重定向和渲染。

## 4.6 服务端语言 PHP

PHP 是一个语法独特并且执行迅速的服务端编程语言，因为其功能的强大，所以它几乎可以实现所有的 CGI 功能。

## 4.7 数据库 Mysql

Mysql 是一个开源的关系型数据库，因为其稳定和高效的特点，Mysql 被广泛的用于中小型网站。

# **5. 系统需求分析**

## 5.1 用户分析

项目的目标用户是以学生和家长为代表的人群，他们有一个共同的特点：个人有闲置书籍的囤积同时有交换/交易这些闲置书籍的需求。

如何做到能够让用户便捷的发送并浏览共享书籍信息和让用户方便联系到对应的发布者从而达成共享书籍的交易或交换，这应该是系统应该做到和关心的事情。  
 从用户的角度出发，其核心诉求就是能与他人达成书籍的共享，因此根据这点，可以提炼出两个要点：

(1).用户应该有能力发布自己想要共享的书籍，又能力根据自己的实际需求 发送对应的信息类型，并可以被其他用户浏览或检索到。

(2).用户要具备可以浏览并检索他人书籍信息的能力，同时具备与其他用户 取得线上联系的能力。

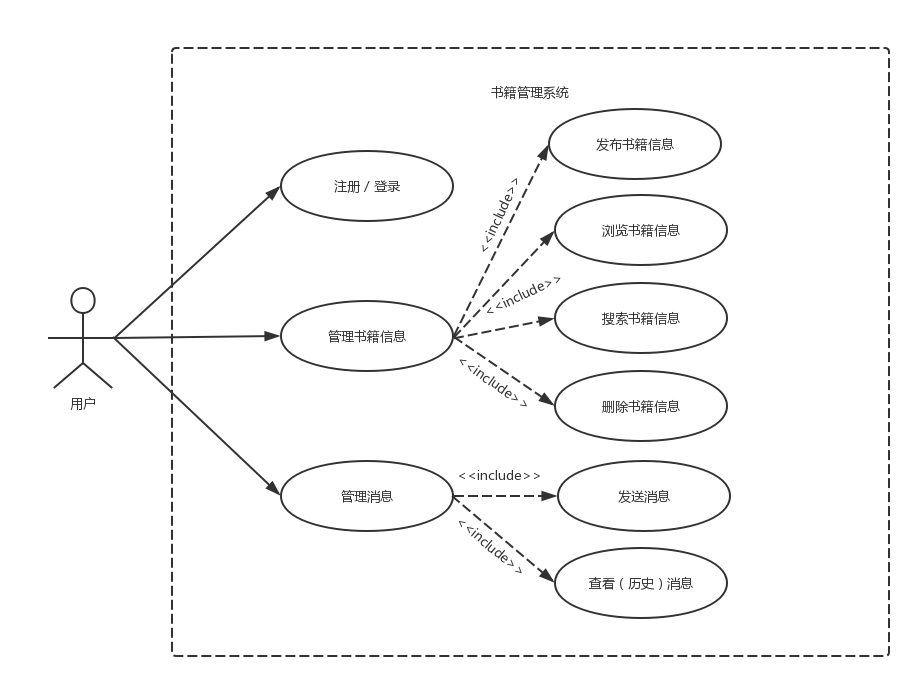
根据以上两点，系统为用户提供了发布书籍信息、浏览书籍信息、搜索以及联系这四个核心功能。  
 此外，考虑到一旦有书籍共享成功，该条信息就已经“过期”的情况，如何还存在于平台至上会给其他用户造成误解，所以系统还为用户提供了删除自己发布过信息的功能。

## 5.2 用例分析

项目（书籍管理系统）包括三个核心功能模块：注册/登录模块、书籍信息管理模块、消息管理模块。  
 其中，书籍信息管理模块由发布信息、浏览信息、搜索信息和删除信息四个子功能模块组成；管理消息模块则由发送消息和查看（历史）消息两个子功能模块组成。

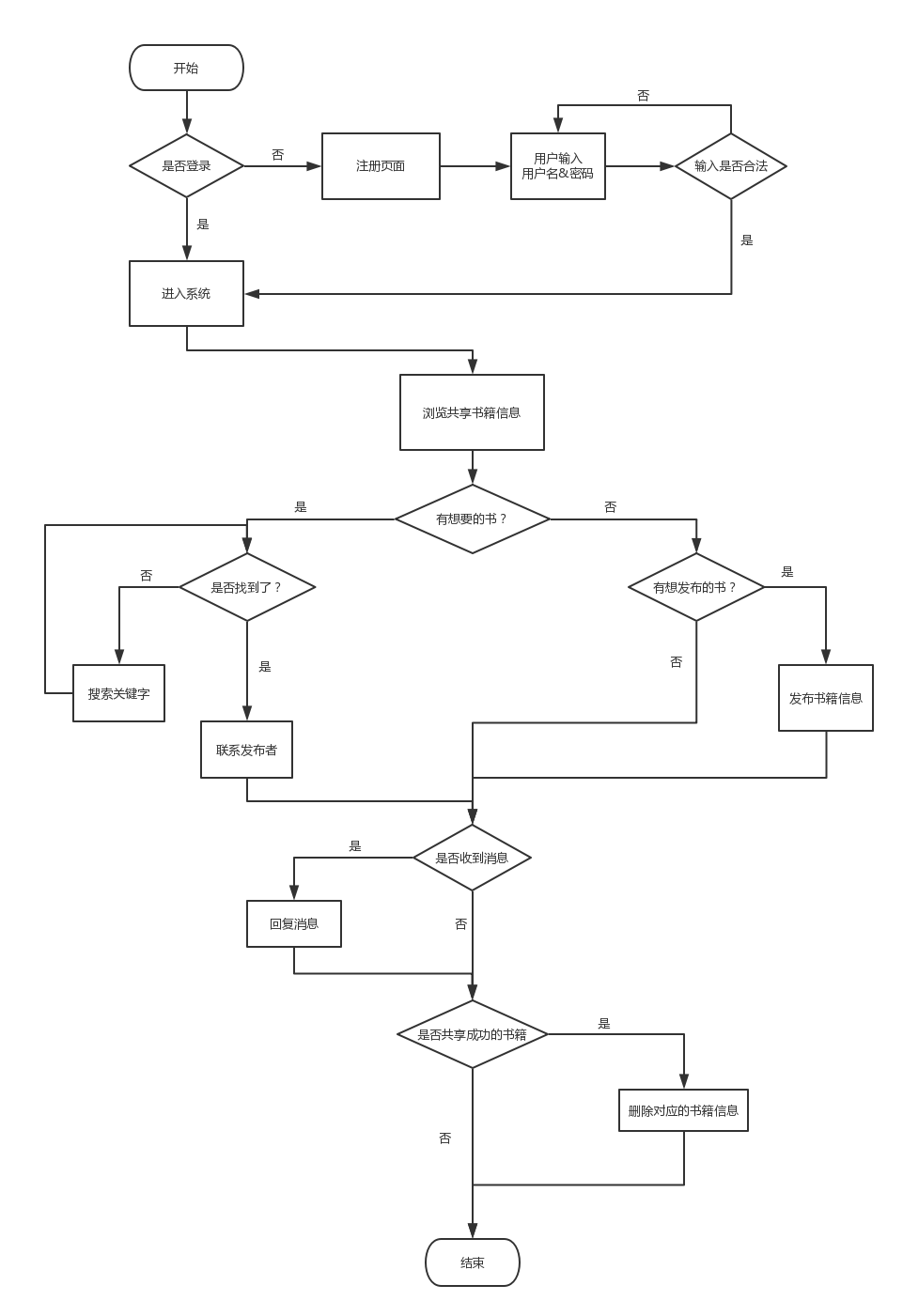
系统的参与者是用户，即有贡献或这获得共享书籍的人群。

系统的用例图如下所示。



# **6. 系统设计**

## 6.1 系统流程设计



## 6.2 数据库设计

数据库是保存一个系统中的原始数据和用户生产数据的地方，所以对于数据库的设计，要很大程度上依赖于系统功能的划分和人机关系的交互。  
 根据需求分析得出，系统所包含的三个主要功能模块分别为：注册/登录模块、书籍信息管理模块和消息管理模块。由此可以得出，我们的数据库中应当包含三张表，分别保存用户信息、书籍信息和用户消息信息。  
 下面逐个分析并设计。

## 6.2.1 用户表

用户表用于保存用户的登录信息，对于系统来说，能做到保存用户信息并实现区分，最终要的两个数据是用户名和密码，所以在该表中设置两个字段来保存这两个用户信息。



值得注意的是，从安全的角度出发，表中用户的密码不能直接用明文进行保存，所以在后端对其进行了加盐后的 md5 处理。

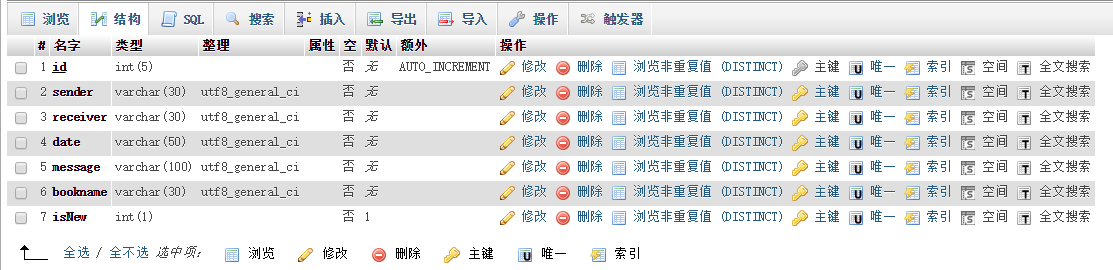
## 6.2.2 书籍信息表

顾名思义，这张表用来保存用户发布的书籍信息。向更深的层面考虑，关于一本书籍，最需要保存哪些信息？  
 首先，书名是必不可少的；另外，由发布者添加的有关书籍的描述也是很重要的一个信息，可以帮助其他用户了解更多关于这本书籍的信息；其次还要知道发布这本书信息的用户是一个什么样的角色：是书的拥有者或是想要这本书；接下来需要标注发布者期望的共享方式，分为交换和交易两种，若是交易，还应该增加一个记录期望价格的字段。

## t_books

## 6.2.3 消息表

## 一条消息需要记录的信息，保证消息可以正确的从消息的发送方传达到接收方。 因此，需要保存这些字段：消息的发送方、消息的接收方、消息的内容。 考虑到消息页的展示内容，还需要保存这些信息：消息发送的时间，特定的书名。



## 6.3 UI 设计

考虑到现在移动端的流行和发展趋势，对于用户界面的设计需要做到响应式设计，即根据用户访问设备屏幕的大小自动适配，不能出现由于访问设备不同而造成的样式错位，否则用户体验会大打折扣。  
 系统使用开源的移动端优先的样式库 Bootstrap的栅格系统 和 CSS3 的媒介查询来实现界面的响应式设计。

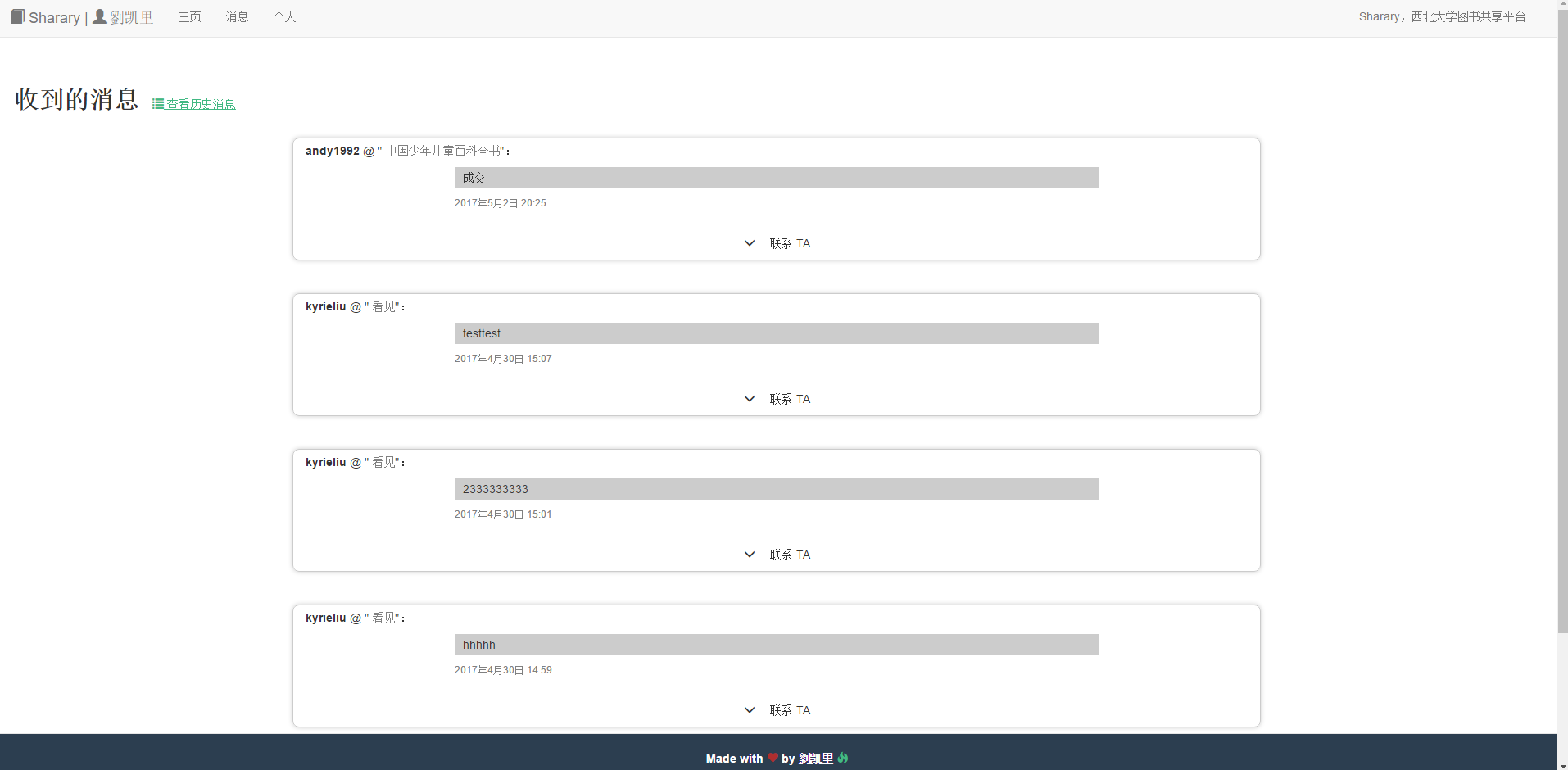
## 6.3.1 移动端

## 6.3.2 PC端

# **web_1**





# **7 代码实现**

## 7.1 开发模式

本系统采用业界较为前沿的开发模式：基于单页应用的前后端分离。  
单页应用，简称SPA（Simple Page Application）,区别于传统的 Web 系统，用户在页面内进行交互时，当点击某个按钮或链接时，不会跳转到其他页面，而是去改变 url 上的锚点，锚点的变更由路由进行监听，从而实现不同页面间 DOM 节点的切换。  
 页面的展示全部由前端的 JavaScript 控制，自然而然的形成了前后端职责的分离，从而避免了多余的跳转，前端部分全部采用局部更新渲染的方式，提供了更好的交互体验。另外，前后端通过 JSON 格式的协议进行通信，后端不再需要对页面模版进行拼装，能更好的专注于数据读取和逻辑运算，减轻了服务端的压力，同时也使得项目的代码更容易维护，便于应对后期变化的需求。

## 7.2 开发环境的搭建

### 7.2.1 前端开发环境

本项目的前端部分用到了 Vue.js，区别于传统的前端开发过程，用到了后缀名为.vue的项目文件，该文件中有三个主标签，分别是html、js和css。浏览器是不认识.vue文件的，所以需要对其进行预编译，这里用到了webpack。  
webpack是一款前端模块化的解决方案，配合多种多样的插件，也能做到前端构建的作用，比如在这个项目中，用到了vue-loader和babel-loader来处理vue特有的语法和ES6语法，用webpack-dev-server启动一个服务器，实现对项目文件变动的监听从而进行热刷新，大大的提高了开发效率。  
 每个用到webpack进行构建的项目都会有一个配置文件（webpack.config.js），由开发人员来指定webpack在这个项目中应该做什么事情，如下。



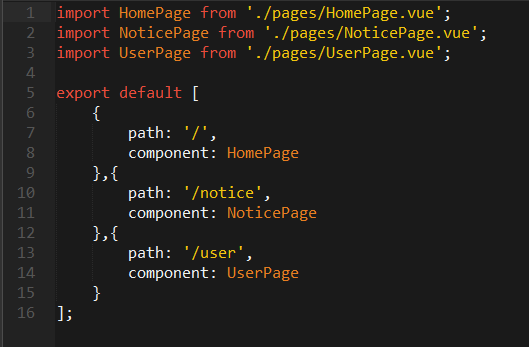
### 7.2.2 后端开发环境

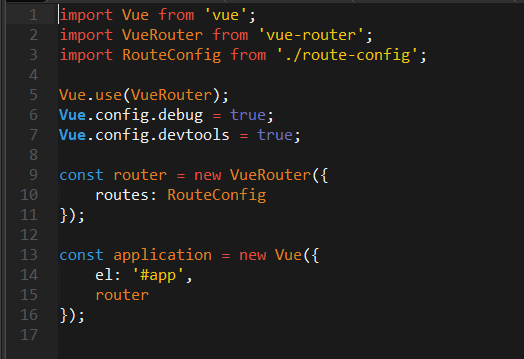
项目的后端语言是php，所以选用phpStudy集成环境，php + Apache + Mysql，将网站的根目录指向项目所在的文件夹即可。

## 7.3 功能模块的实现

### 7.3.1 单页路由的实现与管理

通过 vue-router 在配置文件中定义了路由的设置，接着在入口的 js 中将router实例化并通过 vue 绑定到整个项目中，这样就实现了一个单页应用的整体框架。





### 7.3.2 注册/登录模块

本系统在用户的注册和登录机制上做了区别于传统流程的创新，能让用户更便捷的使用系统。  
 流程图如下。

插入注册流程图

用户在输入用户名和密码并后，系统会检测用户的输入是否合法（是否为空、是否超出长度限制、是否含有非法字符），若通过检验则会显示“进入系统”的按钮。  
 通过类型为POST的Ajax请求将用户的输入传递到后端进行处理。首先判定用户名是否已经存在，若存在则说明是老用户登录，否则为新用户注册。

#### 7.3.2.1 注册

判断当前用户为新用户后（用户表中检索不到用户名），走注册逻辑。  
将用户名和加密后的密码写入用户表中，写入成功则返回相应的返回码给前端，前端在本地的localStorage中保存用户名和由用户密码经过md5加密后的token，以便于用户在第二次进入系统时实现自动登录；若写入成功，返回对应的返回码和错误给前端，前端根据错误信息给用户提示和建议。

#### 7.3.2.2 登录

当后端判断用户为老用户后（用户表中能检索到用户名），走登录逻辑。  
后端查询到对应用户名的密码后，记录为 \_password；获取到由前端传递的用户密码，并做盐值相同的md5加密算法处理后，记作password；对比 password 和 \_password 的值是否相同，若相同，则登录成功，将对应的返回码返回给前端，由前端将用户信息写入本地localStorage后（便于下次自动登录）进入系统；若不相同，登录失败，返回对应的返回码，前端提示用户“用户名或密码错误”。

#### 7.3.2.3 自动登录

用户在登录过一次之后，前端将通过后端验证的用户信息存储在浏览器的缓存当中，当用户再次进入系统时，前端会自动从缓存中获取之前保存好的用户信息，自动通过同步请求发送给后端进行校验，如果校验通过，则跳过登录页面直接进入系统。

### 7.3.3 书籍信息管理模块

书籍信息管理模块是整个系统中最重要的功能模块，用于帮助用户实现书籍的信息共享。

### 7.3.3.1 浏览书籍信息

浏览书籍信息的模块方便用户浏览所有已发布的书籍信息。当用户切换到主页或者个人页面时，前端通过类型为GET为Ajax请求调用后端的接口，后端接受到请求后，从数据库中读取书籍信息，组装成json后返回给前端，前端再将这些数据以模版的形式显示在页面上。  
 为了提升用户体验并节省流量，当用户首次进入页面时，只拉取数据库中的前五条书籍信息，同时监听滚动条，当滚动条触底时，再去拉取五条书籍信息并显示在页面上。

### 7.3.3.2 搜索书籍信息

首页顶部的搜索框，可以让用户输入关键字并进行快速检索功能。  
用户输入关键词后，点击放大镜按钮进行搜索。系统会首先检测输入是否合法，如果不合法则终止这次查询操作；如果输入合法，前端通过类型为GET的Ajax请求，向后端发送用户搜索的关键词，后端在数据库中，根据模糊匹配原则，检索书名字段和描述字段，将符合条件的信息组装成json字符串，返回给前端，前端解析由后端返回的数据，并以模版的形式渲染在页面上。

### 7.3.3.3 发布书籍信息

用户点击位于页面右侧的添加按钮，弹出发布书籍信息的浮层，进行共享书籍信息的发布。  
 当点击发布按钮时，系统先对用户输入的合法性进行检验，若不和法，弹出提示信息并终止发布操作；若输入合法，前端发送一次类型为POST的Ajax请求，向后端传递用户输入的数据，后端将用户输入的书籍插入书籍表中，根据操作的成功与否返回响应的信息给前端，前端拿到响应数据后再反馈给用户。

### 7.3.3.4 删除书籍信息

用户在个人页，点击删除按钮后，向后端发送一个异步请求，传递的数据为可以唯一标识相应书籍信息的时间戳信息和用户名，后端根据这两个信息在数据库中检索到对应的项目后进行删除，将删除结构返回给前端，前端再根据删除结果对用户进行反馈。

### 7.3.4 信息管理模块

信息管理模块保证用户可以向感兴趣书籍的发布者发送信息取得联系，且能接收到来自别人的信息。

### 7.3.4.1 发送消息

当用户点击发送按钮，系统会一如既往的对用户的输入进行合法性检验，如果不合法，直接终止发送请求；若合法，前端发起一次POST请求，将当前用户的名称、发送的信息内容、对应的书籍信息、以及发送时间等数据传递给后端，后端将这些数据插入消息表中，并将结果返回，前端再将结果信息反馈给用户。

### 7.3.4.2 查看消息

用户进入系统时，首先在主页前端会发送一个Ajax请求到后端去查询消息表中 is\_new 字段为1的未读消息的个数，输入大于零，则在页面顶部显示一个通知告诉用户接受到的新消息的个数，并提供一个超链接让用户直接跳转到消息页面。  
 在消息列表中，页面打开时前端发起一个GET请求，后端会检索消息表中 username 与当前用户匹配并且 is\_new 字段为1的消息（说明这条消息是用户未读的）并返回给前端进行模版渲染，返回后将这些消息的是否已读字段置为0。  
当用户点击页面上的“查看历史消息按钮”时，前端发起一个GET请求，后端将查询 username 与当前用户匹配的所有消息，并将查询结果组装成json字符串返回给前端，前端收到返回数据后进行模版渲染，将结果反馈给用户。

# **8.系统测试**

## 8.1 测试环境

### 8.1.1 PC端

Windows 10 64位操作系统，浏览器采用 Chrome、Firefox、Microsoft Edge、Opera、UC等主流浏览器。

### 8.1.2 移动端

IOS9 操作系统，浏览器采用Chrome、Firefox、UC和微信的webview。

## 8.2 测试结果

### 8.2.1 注册 / 登录模块

用户第一次进入系统，进入登录页面，输入用户名和密码后，点击按钮，进入系统；用户再次进入页面时，自动登录生效，直接进入系统。

### 8.2.2 书籍信息管理模块

### 8.2.2.1 浏览书籍信息

用户进入系统，经过短暂的加载时间后，页面显示了已发布的共享书籍信息；进入个人页面，经过短暂的加载时间后，页面显示了当前用户发布的所有共享书籍信息。

### 8.2.2.2 搜索功能

在搜索框输入存在的关键字后，成功搜索出相关的共享书籍信息。

### 8.2.2.3 发布功能

点击添加按钮，弹出发布浮层后，输入合法的书籍信息后，点击发布，系统提示“发布成功”，数据库中成功添加一条对应的数据。

### 8.2.2.4 删除功能

进入用户页面，点击删除按钮，成功删除当前的书籍信息。

### 8.2.3 消息管理模块

### 8.2.3.1 发送消息

在任意一条共享书籍信息底下输入合法的文字信息后，点击发送按钮，经过短暂的等待时间后，系统提示“消息发送成功”，数据库中的消息表中新增一条对应的数据。

### 8.2.3.2 接受消息

进入系统，提示有新的消息，点击进入消息页面，看到最新一条接受到的消息；点击页面上的“查看历史消息”按钮，成功展示当前用户接受到的所有消息。

# **9.总结与展望**

这是一个需要前后端同时把控的项目，这对我来说有一定的挑战性。在完成这个项目的过程中，我掌握了一下知识：

(1).分析并研究了一个小型的具有分享信息功能的社交网络的核心需求，同时站在用户和开发者的角度对整个项目进行了概念模型的建设。

(2).向传统的用户登录机制做出挑战，用浏览器的本地 localStorage 来保存加密后的用户信息，同时实现用户再访问时的自动登录。

(3).结合 Vue 和 Vue-router 实现了前后端分离模式下的单页应用，不仅用局部渲染策略提升了用户体验，还从代码层面增强了项目的可维护性。

(4).结合 Mysql 的增删改查，为整个项目的后端数据管理提供稳定的支持。

同时由于个人能力的局限性已经一些客观原因，系统仍旧存在一些需要改进的地方：

(1).消息管理功能不够完善，比如若是结合 WebSocket 为用户提供一个在线交流的窗口，会更大程度上提高用户体验和整个产品的用户粘性。

(2).用户登录机制的安全性还可以增强，比如结合本地 localStorage 信息的读取，为每次请求增加一个用户身份的校验过程。

(3).对于书籍信息模块来说，虽然增加书籍封面图片功能可能会造成更大的开发和运营成本，但是也能让用户更直观的浏览书籍信息。

# **10.参考文献**

[1][英]Jeremy Keith, [加]Jeffrey Sambells / 杨涛, 王建桥, 杨晓云等. JavaScript DOM编程艺术[M], 北京: 人民邮电出版社, 2011:66-112.  
[2][美]Eric A.Meyer / 侯妍, 尹志忠. CSS权威指南(第三版)[M], 北京: 中国电力出版社, 2007:12-66.  
[3][美]Nicholas C. Zakas / 李松峰, 曹力．JavaScript高级程序设计[M], 北京: 人民邮电出版社, 2012:1-123．

[4]刘佳宁.git-workflow[OL].http://kyrieliu.cn/huno/2017/03/11/my-git-workflow/, 2017.  
[5]尤雨溪. Vue渐进式JavaScript框架[OL]. <https://cn.vuejs.org/>, 2017.  
[6]尤雨溪. vue-router2[OL]. <http://router.vuejs.org/zh-cn/>, 2017.  
[7][美]Aaron Jensen等. webpack[OL]. <http://webpack.github.io/docs/>. 2017.