河北师范大学

本科生毕业论文（设计）

题目：基于 web 前端的书籍推荐网站的实现

学生姓名：李建辉

指导教师：王勇

学　　院：软件学院

专　　业：软件工程

年　　级：2016 级 4 班

完成日期：_____年_____月_____日

学位论文原创性声明

本人所提交的学位论文《基于 web 前端的书籍推荐网站的实现》，是在导师的指导下，独立进行研究工作所取得的原创性成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的研究成果。对本文的研究做出重要贡献的个人和集体，均已在文中标明。

本声明的法律后果由本人承担。

论文作者（签名）：　　指导教师确认（签名）：

年　月　日　　年　月　日

学位论文版权使用授权书

本学位论文作者完全了解河北师范大学有权保留并向国家有关部门或机构送交学位论文的复印件和磁盘，允许论文被查阅和借阅。本人授权河北师范大学可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或其它复制手段保存、汇编学位论文。

论文作者（签名）：　　指导教师（签名）：

年　月　日　　年　月　日

摘　要

随着 web 开发技术的不断更新与发展，我们如今的生活越来越方便。如何将 web 技术应用结合并应用于网站的开发与我们的日常生活中常见问题，是我们研究的问题。本网站采用前后端分离的原则，使用 MySQL 数据库，将数据通过数据库和后台结合，再呈现到前端页面，采用 http 通信协议在前后端传送数据；具体到网站的内容，就是结合自身兴趣搜索关键词网站会推荐出书籍，并将书籍的内容介绍与影视资料视频结合在一起，查询书籍的同时，如果有影视资料的匹配，那么也会有相关的影视资料的展示；在书籍中的经典语句或者对白都会通过这个网站展示出来。

这个网站的意义在于推荐高质量的书籍或者小说，避免打开书城的时候有很多没有营养的快速文学，不知道该选择哪本书。有这个网站之后，我们就能够直接到书城搜索自己想看的书，不用被其他书籍干扰。

关键词：前后端分离　　MySql　　书籍推荐与影视资料

Abstract

As the web technology advanced,we live in a more convenient world. The question that how to combine the technology and our life and then finish a website is the purpose of my research.My website use the model of the front and bank department, and use MySql database. Combined the back and database to get the data , then use HTTP to show the data in the website,which is my work. When it comes to the content, I will recommend some book you will be fond of according your keywords. We also will show you the video that connected with the book you search for .That's mean when you find a book,we will show you some video if the book have films version. Besides, if the book has classical words, it will show you.

The meaning of my website is that we can filter the useless information when we want to search our desired book. When we open our Book City, it often has many things we don't want to see and we don't how to choose . We can use the website find out which book we are preferred and avoid to be bothered by other books.

Key Words: Front-end separation　　MySql　　book recommendation and video

目 录

基于 web 前端的书籍推荐网站的实现

第 1 章 网站建立的目的及意义

1.1 书籍推荐网站的建立背景

在互联网技术十分发达的今天，一切行业和店铺都趋向于线上模式，我们很少去书店买纸质书籍，因此“快餐式”的线上阅读迎合了大部分人的需求。在这种大环境下，各种电子书籍网站和 app 层出不穷，但是这些网站或者 app 首页往往是一些内容质量并不高的书籍，这也导致我们的选择阅读书籍的时候往往受到误导，并非自己本意地去读一些书籍，浪费时间却没有收获。

1.2 书籍推荐网站的现实意义

时代和技术进步的同时，确实引发了其他的问题，让我们在享受生活的同时无法忽视其负面影响。具体问题有：第一，信息良莠不齐，是互联网时代发展的副作用产物，而我们怎样避免或者减轻这个问题，是我们需要进一步探索的意义；第二，我们总是在手机冲浪的时候就完全忘记了时间甚至忘记了打开智能设备的目的，等回头发现已经浪费了很多时间在无关的事上面；第三，在这个追求效率的时代里，花时间在无用事物上消耗人力物力，得不偿失。

正是由于以上原因，我结合自己的兴趣，想要做一个书籍推荐网站，通过这个网站，用户可以直接查到自己想要读的或者趋向去读的书籍，不用浪费时间在那些首页的书籍，提升效率。基于现在已经很成熟的 web 技术，完成这样一个网站需要经过需求分析、网站设计、代码实现以及网站测试这些环节。

第 2 章 网站的需求分析

2.1 业务需求

最典型的利益相关者是网站用户，用户直接使用网站并反复进行各种操作。其次，程序员需要对用户进行验证并实现用户需要的功能，并不断维护整个系统。网站以用户为核心，需要由用户完成的业务用例流程如下，用户注册：网站用户打开网站进行浏览，选择注册账号；用户登录：在已注册的前提下，完成登录；其他用例功能：已登录状态

下的用户根据自己需要进行操作，完成自己的需求，如图 2-1。

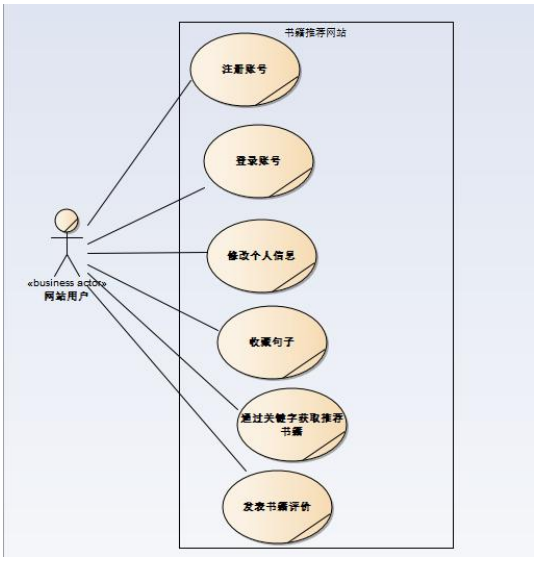

图 2-1 业务用例图

2.2 系统需求分析

首先应当明确，书籍推荐网站系统致力于推荐优质书籍，不会展示书籍的详细内容，只是使用户在信息繁杂的互联网上快速找到适合自己读的书籍。在内容简介和推荐理由上追求高质量，数据库中包含适用于不同年龄段的书籍，包含在文学史上具有深刻意义的书籍，拒接录入无意义、读完毫无收获的书籍。

网站分为未注册用户和已注册用户，这两种用户有着不同的权限。未注册用户可以浏览网页，查看经典句子；而已注册并登录用户可以浏览网页、查看经典句子、收藏句子、对书籍发表评论、根据关键字查看推荐书籍具体内容和影视资源、查看浏览记录，还可以查看和更改个人信息。系统同时要求页面 UI 指导性强，能够使用户看出功能点所在并且顺利操作。

其次，网站应具有可靠性，其数据库中的图片或者视频信息不应该会失效，以防用户有较差的体验。一旦出现稳定性问题，必须立即作出反应，在可接受时间之内找到问题所在，能够迅速维护网站。

第 3 章 网站设计

3.1 网站的 UI 设计

网页的整体页面设计，分为一级页面、二级页面和三级页面：其中一级页面有首页；

二级页面包括登录页面、注册页面、推荐理由页面、经典句子页面、句子收藏页面、个人信息页面；三级页面有书籍详情和发表评论页面；其中每个页面中有不同的图标/文字作为引导，js 点击事件切换并进入不同页面。

在网站的 UI 设计层面，我首先选了主题色。出于希望自己的网站比较轻快明亮一些，我选择了淡紫色（#EEAEEE）作为网站的主题颜色；其次，因为网站是一个推荐网站，我选择了圆圈和爱心作为网站的 logo，寓意是找自己比较喜欢的书籍。在整体视觉上，由于占满整个屏幕的网页不太好看，我选择了整个网页宽度占屏幕宽度的 90%，水平居中，css 表达为 margin:0 auto。

具体页面的介绍如下：首页是整个网站的门面，含有导航栏、轮播图以及热门书籍的呈现。导航栏中搜索框可以搜索关键字来找出自己可能喜欢的书籍，关键字可以是作者、书籍类型或者书名；其次，登录和注册按钮也设置在其中；轮播图采用 js 完成，每隔三秒触发一次切换图片的事件；轮播图右边有一个每日书单，用于推荐当日书籍，这个一般是根据个人搜索历史和类似书籍的品味设置的；热门书籍的呈现则是直接从数据库拿到的数据，从中选取搜索频率较高的书籍，在首页推荐给网站用户，如图 3-1。

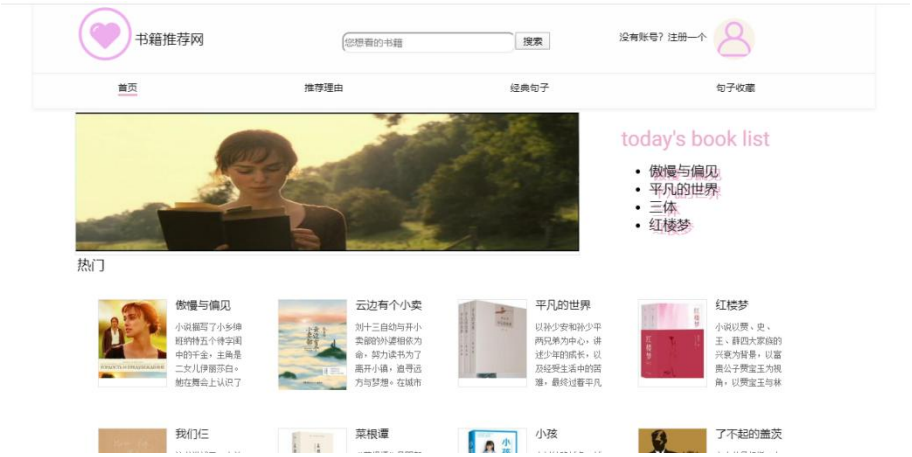

图 3-1 首页

注册页面简洁明了，页面中央是需要提交的表单项，包括用户名、密码、手机号和邮箱的填写项，最下方会有一个提交按钮，实际上注册完成之后也能够完善个人信息，比如性别，是否填写性别由用户决定。该页面设计成照片墙的形式，当鼠标悬停时书籍封面会放大；如图 3-2，图 3-3。

图 3-2 注册页面

图 3-3 选择性别页面

图 3-4 所示登录页面背景图与注册页面一致，含有登录选项的填写页面和提交按钮，通过点击按钮提交表单内容，按钮是用户与系统的接口。

图 3-4 登录页面

推荐书籍详情页是根据用户已经输入的关键词筛选符合词语的书籍，并且如果有相关影视资料，也会呈现在页面上；这个页面主要通过前后端交互实现，页面基本延续首页中热门书籍的样式，补充了内容简介和相关视频，如图 3-5 所示。

图 3-5 书籍详情页

推荐理由页面介绍了推荐该书籍的原因，并且可以点击“发表书评”按钮进入评论页面，如图 3-6，图 3-7。

图 3-6 推荐理由页

图 3-7 评论页

经典句子收藏页面（图 3-8）是我们将在页面上呈现优质书籍中广为流传的经典语录或者段落，而已登录用户可以根据自己的喜好来进行收藏，也可以看到自己收藏的句子。

图 3-8 句子收藏页

个人信息页面包含已登录用户的各种信息，图 3-9、图 3-10、图 3-11、图 3-12。

图 3-9 个人信息页

图 3-10 修改信息页

图 3-11 我的句子页

图 3-12 我的书架页

3.2 网站数据库的配置及表格设计

网站使用 mysql 数据库，基于 Ubuntu 的服务器环境，安装完 MySQL 并且设置好密码之后，进入数据库并授权，以便在客户端可以使数据库图形化；接下来完成本网站需要数据库的配置，下面所有操作皆由 SQL 语句完成，包括 create database 和 create

table 语句。首先创建自己这次要使用的 book 数据库，在 book 数据库下建立要用的表格，对应本次网站的页面设计，我首先建立了用户表，书籍表，书籍类型表，收藏表，搜索表，具体如下：

用户表，包含用户 id、用户名、密码、电话号码、邮箱、性别、头像字段，其中 user_id 是主键；

#	名称	数据类型	长度/设置
1	**user_id**	**INT**	**11**
2	username	VARCHAR	50
3	password	VARCHAR	50
4	tel	VARCHAR	50
5	email	VARCHAR	50
6	sex	CHAR	2
7	avatar	VARCHAR	100

图 3-13 用户表

书籍表，包含书籍 id、名字、内容简介、推荐理由、书籍类型、作者、封面图片、视频以及是否展示字段，其中 novel_id 作为主键；

#	名称	数据类型	长度/设置
1	**novel_id**	**INT**	**11**
2	name	VARCHAR	50
3	content	VARCHAR	1000
4	reson	VARCHAR	1000
5	type	VARCHAR	50
6	author	VARCHAR	50
7	img	VARCHAR	500
8	video	VARCHAR	50
9	isshow	INT	11

图 3-14 书籍表

句子表，包含句子 id，句子内容和书籍名称，句子 id 作为主键；

#	名称	数据类型
1	**juzi_id**	**INT**
2	content	VARCHAR
3	book	VARCHAR

图 3-15 句子表

句子收藏表，包含用户 id、句子的 id;

#	名称	数据类型	长度/设置
1	user_id	INT	11
2	juzi_id	INT	11

图 3-16 句子收藏表

搜索表，包含用户 id，搜索记录，浏览书籍名字；

#	名称	数据类型	长度/设置
1	user_id	INT	11
2	record	VARCHAR	500
3	book	VARCHAR	100

图 3-17 搜索表

评论表，包括用户 id，书籍 id 以及评论内容；

#	名称	数据类型	长度/设置
1	user_id	INT	11
2	novel_id	INT	11
3	plcontent	VARCHAR	500

图 3-18 评论表

3.3 网站语言、框架的安排

总体来说，在使用的语言和框架层面，网站采用前后端分离的原则，使用 HTML5，CSS3，JavaScript 基础知识结合 angular+ionic 框架作为前端，nodejs 中 express 框架作为后台服务，配合使用 MySQL 数据库。

其中，我选择 angular+ionic 框架是因为我想在 angular 的基础之上运用 ionic 的部分组件，比如有好看又成熟的 UI 组件，除此之外，angular 封装了很多服务模块，包括 http 模块，通过它们可以很方便地请求后台数据和发送数据到后台，实现不同构件之间的通信。选择 nodejs 的 express 框架作为后台因为其 routes 的应用和模块化的思想，在 nodejs 中，每个 js 文件都是一个模块。不同模块对应不同功能，很适合作为网站的后台。在我看来，如果我单独把每个功能都放在一个独立的 js 模块中，那么如果在开发过程中出现问题或者代码未完成，我很容易根据自己设定的名称或代号找到相关代码，只要对指定的代码进行优化处理就可以了。

从软件过程层面来讲，我采用的是增量过程模型，根据优先级的不同，依次完成不同阶段的增量：第一个增量中完成了关键字查询书籍以及相关影视资料的功能，这也就整个网站的核心功能，也作为第一个版本；第二个增量是在第一版的基础上完成首页的页面布局以及各个页面跳转，自然也包括各个页面的设计实现；第三个增量版本完成了网站的登录、注册功能；第四个增量中完成了经典句子在前端页面的呈现以及收藏句子的功能；整体的增量过程之上，采用迭代的过程，在实现的同时将各个功能更加细化，去除不合理的环节，使用户使用环境更友好，功能更加完善。在整个过程中，我划分任务集的优先级方法就是对我想要实现的功能进行排序，明确在需求分析基础上各个功能

的重要程度，功能越重要，优先级越高。

3.4 功能模块设计

整个网站软件系统分为注册、登录、关键字查询推荐书籍、查看书籍介绍、收藏句子、查看以及修改个人信息的功能模块。内聚性和耦合性是评判系统独立性的重要指标，本着“高内聚、低耦合”的原则，模块内部紧密联系，而模块与模块之间依赖性较低。模块内部采用功能内聚，这是最高级别的内聚，如果有某个功能模块出现问题，可以最大程度地减少其他模块和整个项目的损失，能够有目的地解决问题。

关注点分离原则认为，任何复杂问题如果可以被分解为可以被独立解决的若干块，那么解决该问题就会变得容易得多。而模块化则是具体表现，而且模块化有利于构件的重复利用，相同功能的模块可以采用相同接口进行数据通信。例如，在本网站设计过程中，收藏句子对应的模块在查看个人信息中也有体现，我们只需要采用相同的接口进行数据请求，在不同页面就可以请求到相同数据信息并进行操作。

3.5 数据传送方式

采用前后端分离的架构，实现数据在前端和服务端的通信是必须解决的问题。事实上，我们解决的就是浏览器、服务器、数据库之间的通信。浏览器与服务器发送数据，而服务器与数据库设置连接。创建完了数据库应该具有的基础表格和字段之后，基于 nodejs 平台，我运用 mysql 的 createPool 方法创建连接池，完成了服务器后台与数据库的连接。这样通过 express 起后台服务，能够直接访问数据库，通过 SQL 语句对 book 数据库表格中的数据进行增删改查操作，也就可以更好的处理 http 请求。

以首页为例，我对我完成的数据库中的内容渲染到前端页面的整个过程进行论述，服务端代码如下：

```
var express = require('express');
var router = express.Router();
const db = require('../model/database');
router.get('/', function(req, res) {
      res.header('Access-Control-Allow-Origin','*');
      res.header('Content-Type','text/plain; charset="utf-8"')
      const sql = 'select * from novel';
```

```
        db.query(sql,(err,result)=>{
                        res.send(result);
                });
});
module.exports = router;
```

其间，采用 express 的路由模块，运用 sql 语句直接获取到了表格中所有条项，然后用回调函数将获取得到的结果发送到前台，result 整体是一个数组，数组中包含若干对象，实际上在浏览器访问服务器地址也能达到这个效果。第二步，在完成了跨域请求代理配置之后，按照后台逻辑，在前端设置同样的接口，请求到 result 中的数据并将数据赋值给已经设置好的变量，如下部分代码：

```
this.http.get('/').subscribe(data=>{  this.novel=data; });
```

第三步，运用变量和 angular 中的数组 ngFor，将所有数据呈现到页面上，配合使用 css 使样式美观。部分代码：

```
<ul *ngFor='let item of novel'>
    <li>
        <div class="book-img">
                <a href="#" ><img src="{{item.img}}" /></a>
        </div>
        <div class="book-info">
                <a><h4>{{item.name}}</h4> </a>
                <p>{{item.content}}</p>
        </div>
    </li>
</ul>
```

类似于首页，其他页面也是采用相同的方法来实现数据交互。涉及到表单填写并提交到后台时，具体问题具体分析，应当采用 post 请求。

第 4 章 基于 web 前端书籍推荐网站的实现

4.1 注册模块的实现

4.1.1 注册模块的流程

用户输入用户名、手机号、密码、以及密码确认；提交时系统检查是否符合填入规则，如果符合规则，那么将各个项写入数据库中；否则提示用户应该输入满足条件的数据。注册成功进入填写性别页面，用户填写性别完毕或选择跳过页面，也就是说，用户可选择是否填写性别选项，如图 4-1。

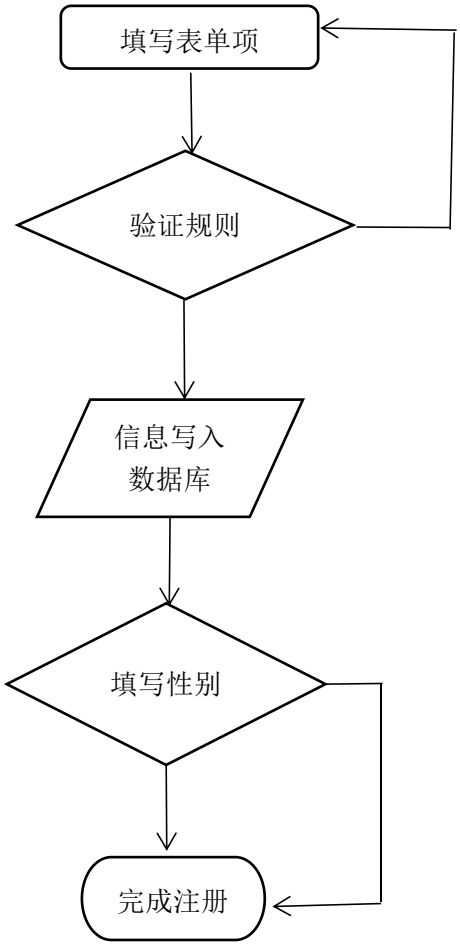

图 4-1 注册流程图

4.1.2 注册功能的实现

用户提交数据校验的规则如下：手机号必须为 11 位数字，且以 1 开头，利用正则表达式`^[1][3,4,5,7,8][0-9]{9}$` 验证；确认密码和密码必须填写一致；邮箱必须符合格式，验证代码如下：

```
myreg=/^[1][3,4,5,7,8][0-9]{9}$/;
emailreg=/\w+[@][a-zA-Z0-9_]+(\.[a-zA-Z0-9_]+)/;
go(username: HTMLInputElement, password:
HTMLInputElement,psw:HTMLInputElement,email:HTMLInputElement,tel:HTMLInput
Element){
        if (username.value == '') {
                alert("请输入账号");
        } else if (password.value == '') {
                alert("请输入密码");
        }else if(password.value!=psw.value){
                alert("请保证两次密码一致");
        }else if(!this.myreg.test(tel.value)){
                alert("清输入合法的手机号");
        }else if (email.value == '') {
                alert("请输入邮箱");
        }else if (!this.emailreg.test(email.value)) {
                alert("请输入正确邮箱");
        }
```

注册用户代码，填写性别部分 SQL 语句使用 update。

```
router.post('/', function(req, res) {
      res.header('Access-Control-Allow-Origin','*');
      res.header('Content-Type','text/plain; charset="utf-8"');
         var username=req.body.username;
        var password=req.body.password;
        var email = req.body.email;
        var tel=req.body.tel;
         const sql='insert into user(username,password,tel,email)
values(?,?,?,?)';
       if(username && password && email && tel){
```

```
        db.query(sql,[username,password,email,tel],(err,result)=>{
                if(err){
               console.error("Error:",err);
                        process.exit();
                 }
           console.log(result);
            res.send(result);

  });
}
```

4.2 登录模块的实现

4.2.1 登录模块的流程

图 4-2 为登录流程图，用户输入用户名/手机号、密码；后台代码根据用户输入信息查询数据库是否有该用户，如果有该用户且密码正确，那么成功登录，并且按照登录用户的数据重新渲染页面；如果密码不正确，则提示用户重新输入。

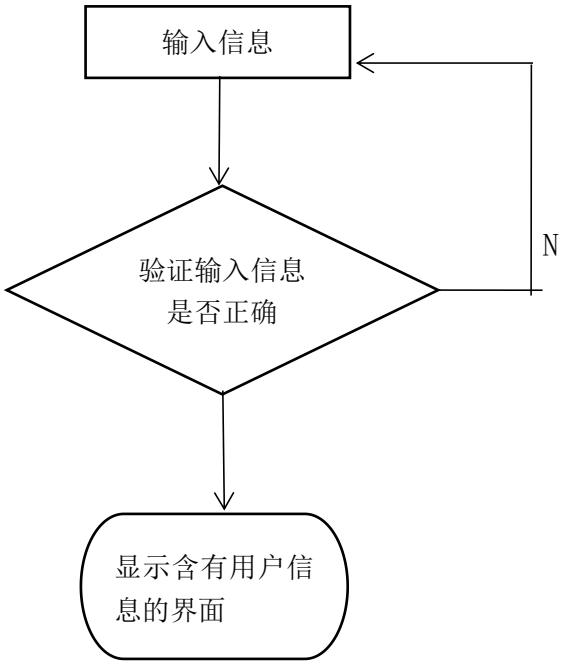

图 4-2 登录流程图

4.2.2 登录功能的实现

用户登录的另一个关注点在于一旦登录某账号，所有与该账号相关的内容都可以显示在页面。在此涉及到本地存储的 localstorage，在登录时将用户 id 存储，之后其他页面可以使用此用户 id，发送请求时连同用户 id 一起发送，也就能使用户获取自身的相关信息。

服务端代码 login.js 如下：

```
router.post('/', function(req, res) {
      res.header('Access-Control-Allow-Origin','*');
      res.header('Content-Type','text/plain; charset="utf-8"');
         var username=req.body.username;
         var password=req.body.password;
         const  sql =  "select * from user where username = '"+ username +"'and
password = '"+password+"' ";
         db.query(sql,(err,result)=>{
                 if(err){
                 res.send('查询失败: '+err);
         }else{
                 res.send(result);

          }

})
```

4.3 用户收藏模块的实现

4.3.1 经典句子收藏的流程

用户观看界面的经典句子/语录时，如果喜欢可以点击收藏，下次登录时会自动显示在用户页面中。点击收藏时，若本身是收藏状态，那么再点一下可以取消收藏，系统也会随着用户本身的状态来调整页面图标的颜色和数据库的插入与删除。

4.3.2 经典句子收藏功能的实现

在点击到经典句子页面时，为了显示正确的颜色，我为从后台获取的数组增加了flag属性，以便于标记句子的状态（用户此时是否收藏）。随着点击事件的触发，网站要立即做出颜色变化反应，切换句子是否收藏的状态。整体的解决方案为嵌套两层请求，第一次get请求获取到数据库中所有句子信息，第二次内层发送post请求获取到数据库中用户收藏的句子信息，最后将两层获取的数据进行循环判断，将重合部分添加flag属性为true。若flag属性为true，则图标显示紫色，否则为白色。

为句子添加flag属性：

```
this.http.get('/juzi').subscribe(data=>{
        this.juzi=data;
        this.user_id = localStorage.getItem('user_id');
        let a={user_id:this.user_id};
        this.http.post('/shoucang/content',a,{
                headers : this.headers,
                observe : 'body',
                responseType : 'json'
        }).subscribe(data1=>{
                this.scjz=data1;
                for(var i=0;i<this.juzi.length;i++){
                        for(var j=0;j<this.scjz.length;j++){
                        if(this.juzi[i].juzi_id==this.scjz[j].juzi_id){
                                this.juzi[i].flag=true;}
                        }
                }
        });
});
```

点击收藏模块部分JavaScript代码：

```
shoucang(index){
        this.flag=this.juzi[index].flag;
        if(!this.flag){         this.flag=false;        }
```

```
        let a={username:this.username,juzi_id:index};
//每点击一次，应该重新获取状态，使 flag 正确标识图标
        if(!this.flag){
                this.http.post('/shoucang/insert',a,{
                        headers : this.headers,
                        observe : 'body',
                        responseType : 'json'
                }).subscribe(data=>{
                console.log(data);
                });
                this.juzi[index].flag=!this.flag;
        }else{
                this.http.post('/shoucang/delete',a,{
                headers : this.headers,
                observe : 'body',
                responseType : 'json'
                }).subscribe(data=>{});
                this.juzi[index].flag=!this.flag;
        }
};
//服务端代码中收藏句子和取消收藏
router.post('/insert', function(req, res) {
      res.header('Access-Control-Allow-Origin','*');
      res.header('Content-Type','text/plain; charset="utf-8"');
      var username=req.body.username;
      const sql = 'insert into shoucang(username,juzi_id) values(?,?) ';
      db.query(sql,[username,juzi_id],(err,result)=>{
                      if(err){ console.log(err);   }
                      res.send(result);
```

```
            });


});
router.post('/delete',function(req,res){
      res.header('Access-Control-Allow-Origin','*');
      res.header('Content-Type','text/plain; charset="utf-8"');
      var username=req.body.username;
      var juzi_id=req.body.juzi_id;
       const sql = 'delete from shoucang where username=? and juzi_id=?';
              db.query(sql,[username,juzi_id],(err,result)=>{
              if(err){ console.log(err); }
              res.send(result);
        });
});
```

4.4 发表评论模块的实现

4.4.1 发表评论功能的流程

用户在文本域中输入评价内容，点击按钮将输入内容提交，若评论内容非空，则将评价内容顺利发送，用户完成评价，否则，内容系统提示用户应输入内容。

4.4.2 发表评论功能的实现

系统从浏览器将数据发送至服务器，服务器接收数据并进行数据库插入操作；值得注意的是，在完成新数据插入时，前端应再次请求数据，以保证页面显示最新内容。服务端部分代码如下：

```
router.post('/content', function(req, res) {
      res.header('Access-Control-Allow-Origin','*');
      res.header('Content-Type','text/plain; charset="utf-8"');
      var novel_id=req.body.index;
      const sql = 'select * from (pinglun,novel,user) where
novel.novel_id=pinglun.novel_id and pinglun.user_id=user.user_id and
```

```
novel.novel_id=? ';
        db.query(sql,[novel_id],(err,result)=>{
                        if(err){ console.log(err);   }
                          res.send(result);
                });
});
router.post('/write',function(req,res){
         res.header('Access-Control-Allow-Origin','*');
        res.header('Content-Type','text/plain; charset="utf-8"');
         var user_id=req.body.user_id;
         var novel_id=req.body.novel_id;
        var pinglun=req.body.pinglun;
         const sql='insert into pinglun(user_id,novel_id,plcontent)
values(?,?,?)';
         db.query(sql,[user_id,novel_id,pinglun],(err,result)=>{
                 if(err){console.log("ERROR"+err)}
                 res.send(result);
         });
})
```

4.5 关键字查询书籍模块的实现

4.5.1 发表评论功能的流程

用户输入作者/书籍类型，网站会推荐出最匹配的书籍，并且显示在用户界面上；在查询到书籍的同时，系统也会将推荐书籍与相关影视资料呈现在页面上。

4.5.2 发表评论功能的实现

用户点击查询图标时，触发了点击事件，js 部分点击事件的作用是将输入信息传送到后台 nodejs 服务；后台中使用 post 方法得到前端传送的数据 A，根据 A 的内容，对数据库进行 select 筛选查询，选出符合 A 的书籍，并将结果（是一个数组，数组里面包含对象）发送至前端；前端对数组中对象进行解析，同时运用框架的双向数据绑定完成前端页面的渲染，包括具体信息和影视资料。

将查找内容作为参数传递到 novel 页面：

```
search(type:HTMLInputElement){
        this.navCtrl.push(NovelPage,{type:type.value});
}
```

NovelPage 获取参数并查询：

```
this.type=this.navParams.get('type');
        let a={type:this.type};
this.http.post('/novel/type',a,{
        headers : this.headers,
        observe : 'body',
        responseType : 'json'
        }
).subscribe(data=>{
console.log(data);
this.novel=data;
});
```

这个功能实际上是我自己结合自身兴趣，用户中也有一部分人群可能有着类似需求。现如今，越来越多的文学作品被当作素材拍成了影视作品，我们也会希望在看到推荐书籍简介的同时也有关联影视作品片段的呈现。

在实现过程中，如何获取这些视频也成了我的难题，解决这个问题我花费了一些时间，按照原来计划我直接利用网络资源，使用直接从网络上请求图片/视频的链接地址。但是在写项目过程中，我发现随着时间推移这些链接部分会失效，这也就导致了我的网站软件系统的可用性和生存性并不高。在进行了深入探索之后，最终的解决方案是先将图片和视频放在我自己的云服务器的指定文件夹里，利用 nginx 配置中的 location 进行网站路由的划分，这样通过自己域名对应的网址就可以直接获取到文件资源（包括图片和视频），在数据库对应 video/img 字段直接输入对应网址链接就可以完成任务。例如在我的数据库中有代表书籍封面的 img 字段，想要把下面图片作为封面，直接将 http://lijianhui.site/image/yun.jpg 作为 img 的值填入表格即可，如下图：

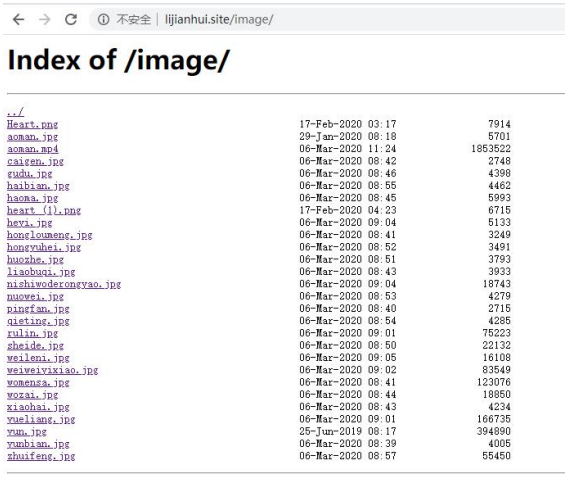

图 4-3 静态资源

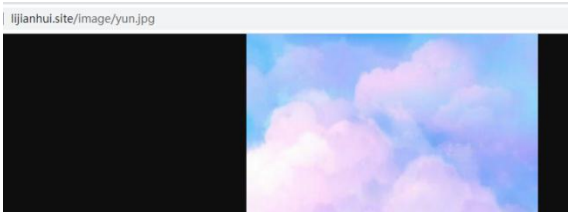

图 4-4 图片链接图

服务端代码：

```
router.post('/type',function(req,res){
        res.header('Access-Control-Allow-Origin','*');
       res.header('Content-Type','text/plain;charset="utf-8"');
       var type=req.body.type;
       const sql='select * from novel where type=?';
       db.query(sql,[type],(err,result)=>{
                    if(err){
                    console.log(err);
                    process.exit();
                    }
                    res.send(result);
              })
})
```

结　论

本次毕业设计的创新点和收获点就在于使用 nginx 反向代理完成了自己域名的部署，学会了通过链接形式访问自己服务器上的静态资源，并运用到了项目开发过程中，能够找到问题所在、思考问题并且解决问题，在很大程度上锻炼了我的逻辑思维。

在这次实践中我更加遵循敏捷开发过程，遵循可运行的软件胜过文档的原则。在实现功能时我并没有提前完成许多文档，但是当最后完成整个项目时，所有的东西都在过程中有了体现，比如系统需求分析、基于场景的需求模型即用例以及软件质量管理和整体项目管理等。比如，我在完成编码过程的同时也在不断测试数据。当然在整个过程中，我不断调整，由于客观原因，我只能逐渐地对原有计划进行变更，在变更中尽力靠近目标愿景。

总体来说，这次毕业设计对于我而言是一次很好的锻炼，既增加了我对各个框架内容的理解，又增加了我的实战经验，同时我发现了自己的不足。在宏观层面上，我最大的收获就是加深了总体对于整个开发流程的了解，具体到项目内部细节，发现自己对于知识的探索之前并没有做得很好，但是这次的毕业设计让我从之前的一知半解到开始抠各种细节，不论是简单的 UI 设计，还是复杂的后端代码完善。在此过程中补齐短板、努力达到预想实现的功能，是我一直在追求的。

参考文献

[1]强琳,林世平.Ionic 与.NET WebApi 实现简单数据交互[J].福建广播电视大学学报,2018(01):28-31.

[2]王伶俐,张传国.基于NodeJS+Express框架的轻应用定制平台的设计与实现[J].计算机科学,2017,44(S2):596-599.

[3]罗杰 S.普莱斯曼（Roger S.Pressman）,布鲁斯 R.马克西姆（Bruce R.Maxim）著; 郑人杰等译. 软件工程：实践者的研究方法[M]. 北京:机械工业出版社.2016.9

[4]《数据库系统概论》（第 4 版），王珊　萨师煊　编著，高等教育出版社，2006.

[5]龚兰兰,凌兴宏.基于敏捷开发的 SSM Web 应用开发实践[J].实验技术与管理,2020,37(02):160-163+167.

[6]Martin Kropp,Andreas Meier,Craig Anslow,Robert Biddle. Satisfaction and its correlates in agile software development[J]. The Journal of Systems & Software,2020,164.

[7]马志强,刘利民,赵俊生.“软件过程与 UML 建模”课程增量式案例教学法的研究与实践[J].内蒙古农业大学学报(社会科学版),2010,12(02):169-170.

致　谢

一眨眼，我们已经在河北师范大学转眼间度过了四个春夏秋冬。曾经我们在去学院的路上不断抱怨路程太长，可现在终于明白，从宿舍去学院的路仅仅是四年那么长。在这四年里，我收获了经久不变的知识，也学习了随着时代前行而不断进步的互联网技术，最重要的是，我感谢在这四年里遇到的同学和老师。感谢老师在这四年中对我的授课与教诲，同样谢谢您对于解答我的问题的热情。本论文在王勇老师的指导下完成，从开始对论文内容无从下手，到后来各种技术问题，老师一直帮我解答疑惑，在此感谢您不厌其烦的教诲。

“工欲善其事，必先利其器”，我会不断提升自己，带着自己的梦想不断前进。毕业意味着我步入人生的另一阶段，我希望带着这四年的积累不断前行，找好自己的方向，不断完成一个个目标。