广东工业大学本科生毕业设计（论文）任务书

题目名称	基于 React 的图片提供网站的设计与开发
学　　院	计算机学院
专业班级	2014 级软件工程 2 班
姓　　名	陈宏浩
学　　号	3114006205

一、毕业设计（论文）的内容与要求

学习和研究相关技术，基于 React 技术设计和实现一个图片提供网站。系统分登录注册模块、展示模块、编辑模块、个人空间模块四大模块。系统除了提供基本的注册、登录、信息修改外，还需为将图片分卷、展示图片描述、统计热度较高的图片，为用户提供图片浏览、图片收藏、图片编辑、图片上传等功能。要求系统能够正确运行，界面友好。

二、毕业设计（论文）应完成的工作

1、学习掌握开发需要的各种技术，包括 nodejs 和 thinkjs，react 相关技术。

2、使用 MySQL 作为后台数据库进行数据库设计。

3、兼容市场上主流的浏览器如 IE 浏览器、Chrome 浏览器、Firefox 浏览器等对应用的兼容性进行测试。

4、以 JavaScript 作为编程语言，运用 ThinkJS 框架设计并实现一个 MVC 框架的图片提供与编辑网站，并使用 react 技术完成 MVC 中 V 的部分。功能包括前台部分：注册、登录；主功能部分：图片查看、浏览、收藏、编辑、上传。

5、按要求撰写毕业论文。

三、毕业设计（论文）进程安排

序号	设计（论文）各阶段内容	起止日期
1	选定毕业设计题目，接受并理解任务书，查阅相关技术资料，准备开题报告	1.1～1.9
2	需求分析、安装配置系统开发和运行环境	1.10～1.31
3	分析和设计系统 E-R 图（设计数据结构）	2.1～2.10
4	分析和设计系统结构图、数据库外模式、模式和内模式（设计数据结构、设计关键算法）	2.21～3.20
5	创建数据库、应用程序编码和测试	3.21～4.20
6	系统试运行、整体调试和性能改进、收集论文资料	4.21～4.30
7	设计技术总结、撰写毕业论文	5.1～5.20
8	整理资料、准备和进行毕业答辩	5.20～5.30

四、应收集的资料及主要参考文献

[1] 刘波 .关于 B/S 和 C/S 架构的分析[J].人民检察,2004(09):77-78.

[2] 庄严.基于 B/S 结构的软件开发技术分析[J].电子制作,2016(24):44.

[3] 向雨濛. 中美图片社交产品对比研究[D].浙江传媒学院,2016.

[4] 百度百科 . react 词条 [EB/OL] . https://baike.baidu.com/item/react/18077599?fr=aladdin 2017-10-15/2018-5-10

[5] 朴灵．深入浅出 Node.js [M]．北京：人民邮电出版社 / 2014-08．19-29

[6] ThinkJS 官方文档．介绍 [EB/OL] . https://thinkjs.org/zh-cn/doc/2.2/index.html 2017-9-14/2018-5-11

[7] 郭彦辉.响应式 Web 设计的研究与实现[J].软件,2018,39(01):169-172.

[8] 田红玉.响应式网页设计与实现[J].电子测试,2016(03):80+79.

[9] 维基百科 . MySQL 词条 [EB/OL]. http://zh.wikipedia.org/zh/mysql 2018-5-10/2018-5-11

[10] 简书 . React 简单介绍 [EB/OL].https://www.jianshu.com/p/ae482813b791 2014-12-10/2018-5-11

[11]陈胤梁. jQuery 插件为 Web 应用带来的变化[J]. 计算机光盘软件与应用, 2013, (05):9-11.

[12] Tali Garsiel. How browsers work[OL].[2015]. http://taligarsiel.com/Projects/howbrowserswork1.htm 2015/2018-5-18

[13]周兴宇,卞佳丽. 基于 React 的前端组件化研究与设计[EB/OL].北京：中国科技论文在线.http://www.paper.edu.cn/releasepaper/content/201601-80. 2016-01-05/2018-5-12

[14]Responsive Design in 3 Steps [EB/OL]. http://webdesignerwall.com/tutorials/responsive-design-in-3-steps 2011-12-8/2018-5-12

发出任务书日期：　　年　月　日　　指导教师签名：

预计完成日期：　　年　月　日　　专业负责人签章：

主管教学院长签章：

廣東工業大學

本科毕业设计（论文）

基于 React 的图片提供网站的设计与实现

学　　院　计算机学院

专　　业　软件工程

年级班别　2014 级（2）班

学　　号　3114006205

学生姓名　陈宏浩

指导教师　何晓桃

2018 年 5 月

摘　　要

图片提供网站是以提供图片为核心的服务型网站，其主要目标用户群为设计师、摄影师、自媒体等，次要目标客户为以消遣、娱乐、猎奇、扩宽视野为目的的网民。本系统希望在满足主要目标用户群需求的同时，吸引次要目标客户，利用图片背后的故事、背景，向用户传达图片中的精神、人文关怀，从而提高网站的吸引力。

图片的价值在于其美感价值、实用价值、意义价值。而当下绝大多数图片提供网站都只有图片，没有文字，忽略了图片也是需要文字补充的。本系统针对以上问题，以“图片+故事/描述”为模式提供信息，旨在为用户提供带有故事性、叙述性的图片，阐述图片背后的故事，让图片更为厚重，吸引各用户群。

整个系统采用 B/S 结构，分为服务端和浏览器端,均采用 MVC 设计模型，服务端负责管理、提供数据，浏览器端需要完成页面的渲染、交互、网络请求。分为四大功能模块：登录注册模块、个人管理模块、图片展示模块、图片编辑模块。

本平台使用 NodeJs、React 等技术实现，采用了 ThinkJS 框架，运行在 NodeJS 上，是基于网络的第三方服务平台。

关键词：照片，图片故事，React，NodeJS，ThinkJS

Abstract

The image-providing website is a service-oriented website that provides images as its core. Its main target users are designers, photographers, and self-media. Secondary target customers are Internet users aiming at pastime, entertainment, novelty and widening their horizons. This website hopes to meet the needs of the major target user groups, while attracting secondary target customers. Using the stories and backgrounds behind the pictures, it will convey the spirit and humanistic care in the pictures to the users, thereby enhancing the attractiveness of the site.

The value of the picture lies in its aesthetic value, practical value and meaning value. The vast majority of image-providing websites at the moment only have pictures, no text, and ignoring the pictures also require text additions to show more information. This website aims to provide information based on the “image+story/description” model for the above issues. It aims to provide users with stories and narrative images, tap the stories behind the images, make the images more heavy, and attract users. .

The entire system adopts the B/S structure and is divided into the server and the browser. Both use the MVC design model. The server is responsible for managing and providing data. The browser needs to complete the rendering, interaction, and network requests of the page. Divided into four functional modules: login registration module, personal management module, picture display module, picture editing module.

The platform is implemented using technologies such as NodeJs and React. It adopts the ThinkJS framework and runs on NodeJS. It is a third-party service platform based on the Internet.

Keywords: Picture，Story in Picture，React，NodeJS，ThinkJS

目　　录

1 绪论 1

1.1 图片网站的前景 1

1.1.1 图片网站的市场分析 1

1.2 互联网发展 1

1.2.1 网络架构的变化 1

1.3 课题研究意义 2

1.4 本文结构 2

2 技术方案 3

2.1 Node.js 3

2.2 ThinkJS 3

2.3 响应式网页设计 3

2.4 MySQL 4

2.5 React 4

3 对 React 的研究与应用 5

3.1 React 5

3.1.1 React 原理研究 5

3.1.2 React 特性 6

3.2 React 的应用 6

3.2.1 内容类似的组件的复用 6

3.2.2 复合组件 7

3.3 对 React 的小结 9

4 系统需求分析 10

4.1 市场需求分析 10

4.2 业务需求分析 10

4.3 系统功能简介以及系统用例图 10

4.4 功能需求分析 11

4.4.1 登录注册模块 11
4.4.2 展示图片模块 11
4.4.3 编辑模块 11
4.4.4 个人空间模块 11
4.5 数据需求分析 12
4.5.1 数据流图 12
4.5.2 数据字典 12
5 系统设计 14
5.1 系统体系结构 14
5.1.1 前端 MVC 15
5.1.2 服务端 MVC 15
5.2 软件功能结构 16
5.3 数据库设计 17
5.3.1 概念结构设计 17
5.3.1 逻辑结构设计 17
6 系统实现 21
6.1 登录注册模块 21
6.2 个人空间模块实现 24
6.3 图片展示模块实现 28
6.4 编辑模块实现 32
7 系统测试 33
7.1 登陆注册模块 33
7.2 图片展示模块 35
7.3 图片编辑模块 36
7.4 个人空间模块 37
7.5 补充测试 39
8 系统安装使用说明 41

8.1 安装与环境配置.....41
8.2 使用说明.....42
9 心得体会.....47
参 考 文 献.....49
致 谢.....51

1 绪论

1.1 图片网站的前景

1.1.1 图片网站的市场分析

图片网站，可以分为图片社交网站和图片提供网站。

近年来，图片社交网站，比如最早的 Facebook、推特，到后来的微信朋友圈、Instagram、微博等，都吸引了大量的用户。他们以图片、文字作为主要媒介，让用户相互分享个人状态、想法。形象的图片可以弥补文字在表达上的不足，辅助文字的表达。同时，一图胜千言，图片可以让用户快速的浏览与理解，这一切决定了图片非常适合作为阅读内容被用户去消费，尤其是当下，现代人的阅读习惯趋于碎片化。因而，这类图片社交网站得以被用户快速地接纳并喜爱。

另一方面，在国内有许许多多图片提供网站。这些网站向摄影师、设计师购买作品，为需要的人提供图片，方便自媒体、创业公司收集素材以做下一步的工作。而这些图片素材最大的问题就是，很多时候，只是一张美轮美奂的图片，而没有对图片一丝一毫的描述，使图片只能停留在 2d 平面上。因此，这类网站往往只能在一个小圈子中流传，当用户有工作有任务有需要的时候才会去浏览。的确，在一些情况下，没有必要给图片附加多余的文字描述；但更多的时候，给图片加上文字的描述，能够让人身临其境，或与图片中的主角共情，这样能够让图片的价值成倍地提高。从而，提高用户分享图片的可能性，藉此提高网站的知名度。

因此，一个新的，将图片辅以文字描述的图片提供网站，相信能够吸引到很多用户。

1.2 互联网发展

1.2.1 网络架构的变化

C/S（Client/Server）结构，即客户机和服务器结构。用户需要先下载安装服务商提供的客户端，才能使用服务商提供的服务。其缺点有以下几点：一是客户端的开发和维护需要针对不同的版本进行，代价高、效率低，而 B/S（Browser/Server）结构，即浏览器和服务器结构，在这种结构下，系统界面是通过浏览器来展现，我们往往将少部分事

务逻辑交给前端浏览器，而将大部分的事务逻辑在服务器端完成[1]。这样就极大地降低了客户端的负担，同时，由于只需要维护浏览器端网页，因而能够极大地减轻系统维护与升级的成本和工作量，降低了用户的使用成本和服务提供商的维护成本，把压力转嫁给服务端，因此这种结构下服务器端的工作较重，对服务器的性能要求更高[2]。

1.3 课题研究意义

本次毕业设计所开发的系统是一个图片提供网站。目前，国内的许多图片提供网站只停留在提供图片给用户作为素材上，想要在与他们的竞争中取得优势，可以从当下如火如荼的图片社交网站上吸取经验，以“图片+故事/描述”的展现形态吸引用户，不仅可以吸引原有的需要找图片素材的用户，还可以吸引国内主流的“消遣型用户”。要知道，图片的价值从来不仅仅是图片表面的美感、实用价值，还在于其内容价值，即其纪实性的表达、实物与现场的描述[3]。由此，实现图片素材网站的升级，无疑能够提高整体竞争能力，吸引到更多的用户。

另一方面，本次毕业设计中大量使用的 React 是 Facebook 开发的一款 JS 库，相比以往的前端框架，React 被认为拥有革命性的创新，用于构建“可预期的”和“声明式的”Web 用户界面[4]。项目中对 react 的使用，的确使代码稍显繁琐，但思想上、组件关系上非常简单。开发时，代码的繁琐从来不是开发的痛点，代码间模块间的引用和关系才是，而 react 很好地解决了这个痛点，为网站的前端开发提供了很大的帮助。与此同时，react 在国外得到广泛的使用，被誉为一统江湖，在国内虽然还没有广泛的使用，但已经得到广泛的关注，作为一项新兴技术，react 已经是目前前端开发的热点。

1.4 本文结构

本文第 1 章简单对当前的图片网站前景，及互联网技术趋势以及本文结构做简单的概述。第 2 章对本系统用到的技术方案做一个介绍。第 3 章是对项目中使用的 React 技术做一个简介和分析。第 4 章是系统需求分析，主要分析了这个系统的市场前景和可行性，并分析各个模块。第 5 章主要对系统的前端后台数据库等方面进行设计。第 6 章是部分关键代码的实现截图。第 7 章是对系统的测试。第 8 章是系统安装使用说明。第 9 章是完成本系统的心得体会。最后则是参考文献和致谢部分。

2 技术方案

本次毕业设计采用 B/S 结构，即浏览器/服务端结构。其中，服务端使用的是基于 NodeJS 的 Thinkjs 框架，数据库选用了 MySQL 的社区版。浏览器端主要使用了 React 来构建用户界面，JQuery 负责交互，基于 Bootstarp 的 zui 框架完善界面外观及响应式设计。

2.1 Node.js

NodeJS 是一个事件驱动 I/O 服务端 Javascript 运行环境，实质是对 Chrome V8 引擎进行了封装，V8 引擎本身使用了一些最新的编译技术，这使得用 Javascript 这类脚本语言编写出来的代码运行速度获得了极大提升，又节省了开发成本，NodeJS 对一些特殊用例进行优化，提供替代的 API，使得 V8 在非浏览器环境下运行得更好[5]。

2.2 ThinkJS

Thinkjs 是一个快速、简单的基于 MVC 和面向对象的轻量级 Node.js 开发框架，遵循 MIT 协议发布，使用了 MVC 架构模式的思想，秉承简洁易用的设计原则，在保持出色的性能和至简的代码同时，注重开发体验和易用性，为 WEB 应用开发提供强有力的支持，Thinkjs 里面很多特性来源于 ThinkPHP，同时根据 NodeJS 的特点，使用了 Promise, WebSocket 等特性，让代码更简洁、优雅[6]。

2.3 响应式网页设计

以往很多网站，分别为手机和 PC 两种平台开发了两种不同的版本，用户使用不同的终端访问的时候其实看到的是不同的页面。而响应式网页设计的核心目标则是：无论用户正在使用笔记本还是手机，页面都应该根据不同的设备，自动切换分辨率、图片尺寸，甚至相关脚本功能等。这样，我们就可以不必为不断到来的新设备做专门的版本设计和开发[7]。毕竟，无论是当下各种各样的手机屏幕，还是不同的台式电脑、平板电脑，其页面尺寸都不可能完全一致。而同一个网页，在手机上看也许太多太挤，我们需要缩放来看；而电脑上看也许会觉得太少太空，很多空间没有利用。

而响应式 Web 设计不仅仅是关于屏幕分辨率自适应以及自动缩放的图片等等，它更像是一种对于设计的全新思维模式，即我们应当向下兼容、移动优先[8]。

2.4 MySQL

MySQL 是一个关系型数据库管理系统，由瑞典 MySQL AB 公司开发，目前属于 Oracle 公司。MySQL 分为社区版和商业版，本系统使用的是社区版，体积小、速度快、总体拥有成本低，而且开放源码，一般中小型网站的开发都选择 MySQL 作为网站数据库[9]。

2.5 React

React 是 Facebook 开发的一款 JS 库，主要用于构建 UI，相比以往的前端框架，被认为拥有革命性的创新。主要体现在：

1、Virtual DOM 虚拟 DOM

传统的 web 应用，操作 DOM 结构一般是对各个部分逐个进行更新操作的，但 DOM 更新通常比较昂贵，逐个部分进行更新会极大地影响性能。而 React 为了尽可能减少对 DOM 的操作，提供了一种强大的方式来更新 DOM，就是 Virtual DOM，React 抽象出来的一个轻量级的虚拟的 DOM 对象，描述 DOM 应该什么样子的，页面应该如何呈现，代码先根据需要，在这个 Virtual DOM 上进行修改，再根据这个修改后的 Virtual DOM 去更新真实的 DOM，这种多一层的虚拟 DOM 操作代替直接的 DOM 操作，不保证马上影响真实的 DOM，而 React 会等到事件循环结束，然后利用算法，通过当前新的虚拟 DOM 表述与之前的作比较，计算出最小的步骤更新真实的 DOM，从而提高性能[10]。

2、Components 组件

在 DOM 树上的节点被称为元素，在 React 的 Virtual DOM 上，节点则被称为 commponent。Virtual DOM 的节点就是一个完整抽象的组件，它是由 commponents 组成[10]。

3、声明式设计

React 采用声明范式，可以轻松描述应用[10]。

3 对 React 的研究与应用

在本系统中，用到的核心技术是 React，React 完成了本系统很多工作。下面将重点介绍 React。

在计算机软件设计中，组件化是常见的思想，但是在 Web 前端这个领域，并没有很通用的组件模式，因为缺少一个大家都能认同的实现方式，所以很多框架/库都实现了自己的组件化方式[11]。

3.1 React

React 是 Facebook 开发的一款 JS 库，主要用于构建 UI，为前端带来了与以往大不相同的开发方式和全新的设计理念，因而被认为拥有革命性的创新。

3.1.1 React 原理研究

1、虚拟 DOM

React 使用 JavaScript 在浏览器端实现了一套 DOM API，称之为 Virtual DOM 即虚拟 DOM，基于 React 的 DOM 构造都是通过虚拟 DOM 进行的，每当数据变化时，React 会刷新整个 DOM 树，并将当前的树和上一次的 DOM 树进行对比，得到 DOM 结构的区别，然后仅仅将需要变化的部分进行实际的浏览器 DOM 更新，并且，React 会批量处理虚拟 DOM 的刷新，在一次事件内的多次数据变化会被合并[13]。

2、有状态组件

React 把用户界面视为组件组成，每个组件可以由多个组件组成，形成一种树状结构。每个组件都是一个简单状态机，数据变化将会改变组件的状态，状态改变将会改变用户界面，React 中，每个组件拥有 state 属性， state 的变化将引起组件重新渲染，这个过程不需要手动操作 DOM， React 负责将变化高效地更新到实际的 DOM 当中[12]。

3.1.2 React 特性

1、虚拟 DOM 带来整体刷新

在原生的前端开发中，需要大量的 DOM 操作，而 DOM 操作往往是影响页面性能的最大因素；另一方面，在完成复杂的交互的时候，大量节点的大量 DOM 操作不仅可能带来性能的损失，还有可能带来一系列的问题。而 React 使用虚拟 DOM 树，先操作虚拟 DOM 树，再将更新反映到真实 DOM 中，尽可能地减少了程序中每次的 DOM 操作，解决了前端可能存在的层叠更新的问题，大大提高了性能，也提高了开发效率。

2、 有状态组件带来良好的复用性和可维护性

React 基于组件，组件的良好封装和组件间清晰的结构保证了无论是复合组件，还是单个组件，使用者都只需要关心组件的接口通信。在设计新组件的时候，复用已经封装好的旧组件，可以极大地减少工作量，提高开发效率，带来更强的健壮性，在维护、修改的时候，只需要替换和修改组件，将变更降低到最小。

3.2 React 的应用

3.2.1 内容类似的组件的复用

在本系统中，各个模块页面的头部结构类似，内容接近，都使用了同一个组件构建，页面间的不同之处，则交由各页面自主调整。效果如图 3.1、图 3.2、图 3.3，其公共组件代码如图 3.4。

图 3.1 头部样例一

图 3.2 头部样例二

图 3.3 头部样例三

图 3.4 头部代码实现

3.2.2 复合组件

复合组件是指组件内嵌套组件的组件。复合组件可以向子级组件传递值、读取值，但小组件不能向父级组件传递、读取值。具体实现如图 3.5 中的 render 方法，Container 组件里面嵌套了 Gallery 组件，并向该 Gallery 组件传递了若干值供使用。

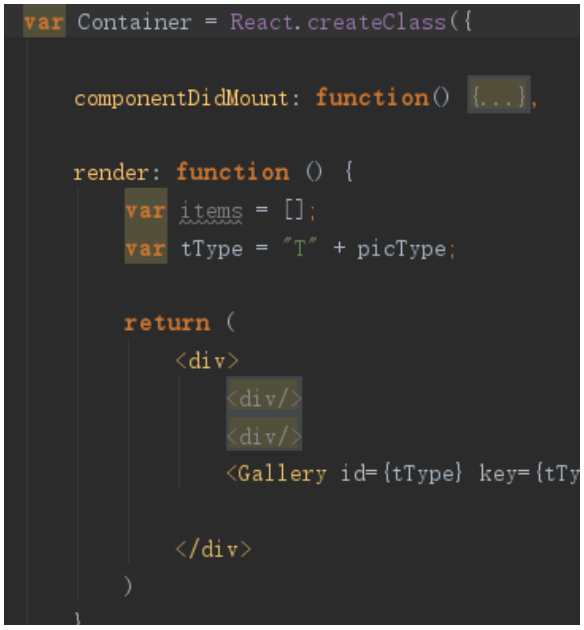

图 3.5 Container 组件

如图 6.6 所示，Gallery 组件内不仅含有一个 Ul 组件，还拥有若干个、根据后台数据动态生成的 Picture2 组件。

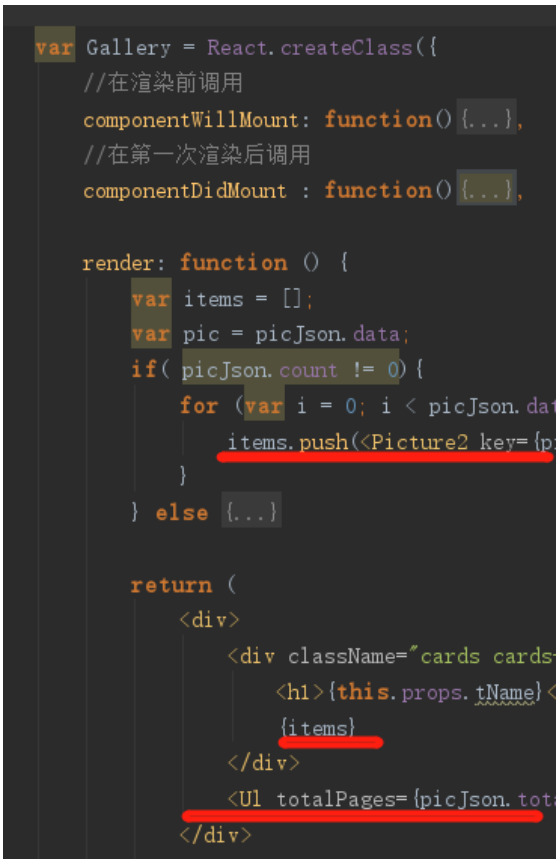

图 3.6 Gallery 组件

上述组件的效果如图 3.7，其结构为 Container 组件内嵌套 Gallery 组件，Gallery 组件内嵌套一个 Ul 组件和若干个 Picture2 组件。Pirture2 组件的数目是根据服务端传过来的 JSON 数据决定而动态加载的；而 Ul 组件展现了剩余页数，也是根据服务端数据动态加载的。

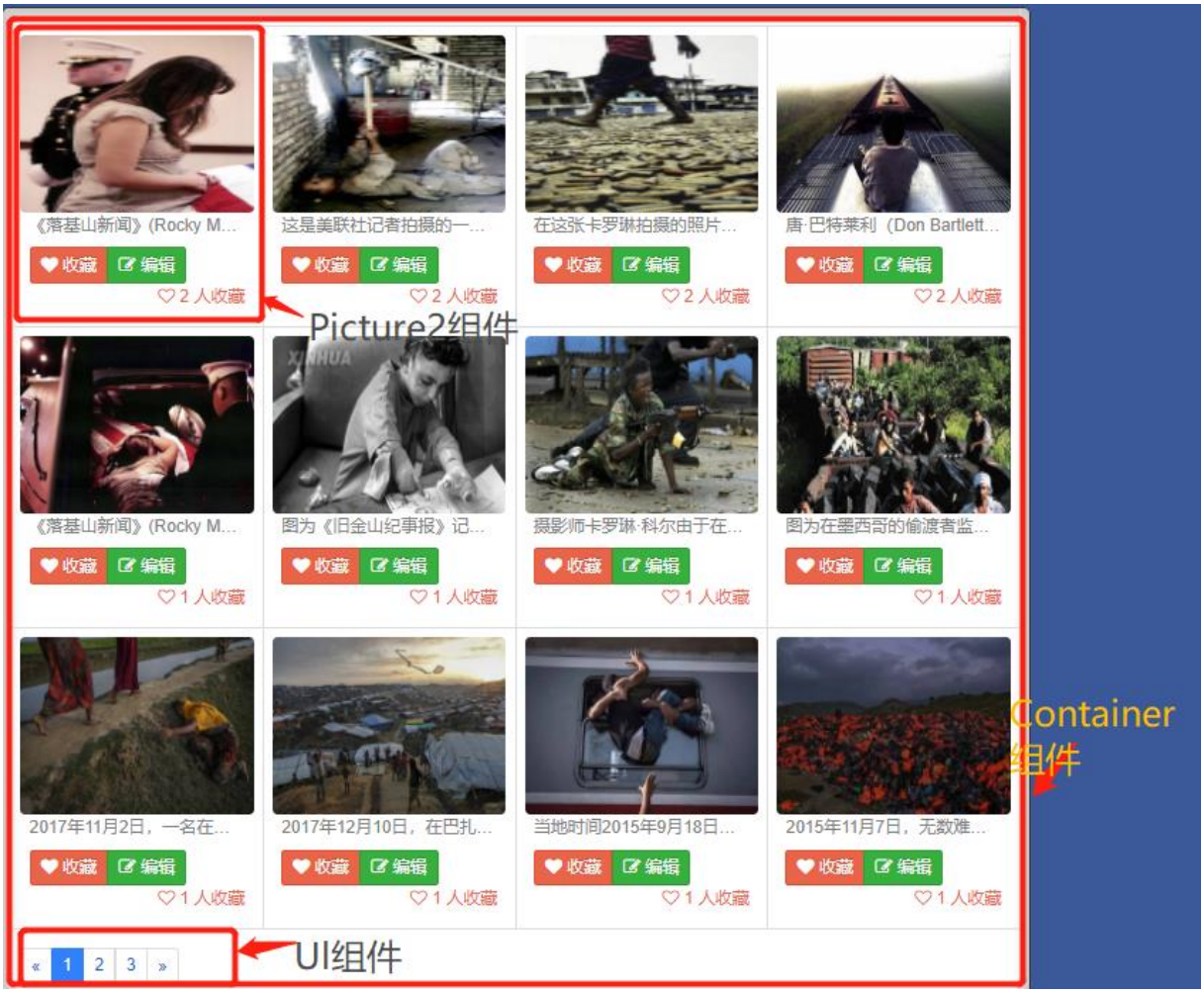

图 3.7 Gallery 组件 & Picture2 & Ul 组件效果图

3.3 对 React 的小结

在制作本系统的过程中，在面对 React 这个全新的前端技术时，有很多强大的功能、精妙的语法糖都还没有学明白，还存在很多瑕疵。但总的来说，React 为我的开发带来了很大的方便。一个个的组件像一块块积木，逐步地构建起整个页面。

React 本身只是前端 MV* 中的一个 V，并不适合与其他的框架如 AngularJS、VueJS 作比较，但其组件化的开发方式，相信会能够提高开发效率的同时，保证乃至提高 web 应用的质量。

4 系统需求分析

4.1 市场需求分析

本系统的核心功能为找图+存图/收藏，与此网站最为相似的是国外的 pinterest，国内的市场只有花瓣比较接近，且花瓣更多的是类似 twitter 的图片社交网站，里面比较多用户的个人照片。而本系统希望以社会热点、国内外新闻图片为核心，阐述与分享图片背后的故事，通过赋予图片内容价值，增加图片的价值，让图片价值不仅仅只有其美感价值。

相比其他的图片提供网站，本系统能够拓展图片的内涵，让图片具有学习价值、使用/引用价值、人文价值，让系统受众不仅局限于设计师、摄影爱好者，更在广泛的国内群众。

4.2 业务需求分析

图片提供网站目的在于为各种用户提供带有叙述性的图片，为用户展示图片背后的故事，并在用户需要的情况下提供有限的图片编辑功能。系统设计以图片的展示为核心，以图片的文字为用户展示图片的内涵。

在系统提供上述服务的同时，提升代码健壮性、保证系统的可维护性和可拓展性，出现问题能够及时给予反馈。

4.3 系统功能简介以及系统用例图

本系统主要的功能如下：一、游客可以登录、注册为用户、浏览图片、编辑图片；二、用户拥有访客的所有功能，另外可以收藏自己喜爱的图片，修改个人信息、到个人空间查看收藏的图片、上传图片。本系统的用例图如图 4.1 所示。

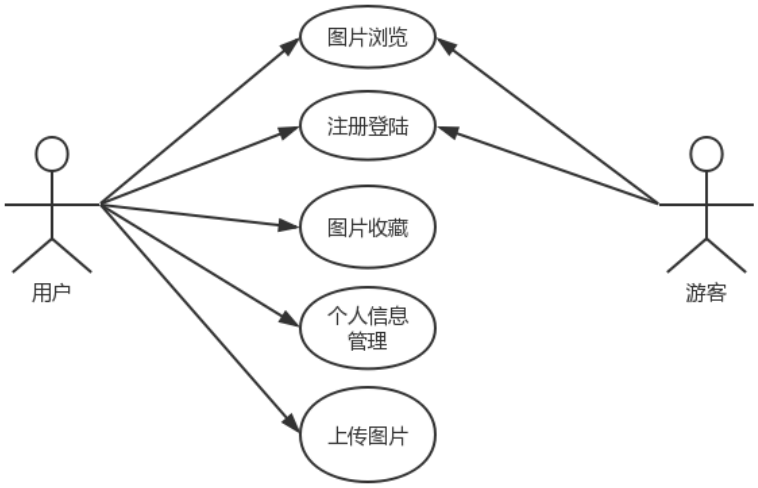

图 4.1 系统用例图

4.4 功能需求分析

本系统分为四大功能模块，分别为登录注册模块、展示模块、编辑模块和个人空间模块。各大功能模块下有相应的子功能模块。

4.4.1 登录注册模块

新用户用手机号作为账号进行注册，注册时只需要提供基本的个人信息。

4.4.2 展示图片模块

分为两部分，一部分是在首页展示各分类热度较高的图片的缩略图；另一部分是用户在首页选择类别进入对应分类后，查看图片，用户浏览图片的同时能够看到图片的描述、热度等，对自己喜爱的图片可收藏。

4.4.3 编辑模块

网站提供有限的图片编辑功能，用户可以对网站中的图片或自己的图片做有限的编辑，并提供下载功能。

4.4.4 个人空间模块

分为三部分，一部分是对自己的昵称、邮箱等信息的管理，另一部分是查看自己收藏的图片，还有一部分是上传图片。

4.5 数据需求分析

4.5.1 数据流图

本系统的数据流图如图 4.3、图 4.4 所示。

图 4.3 系统上下文图

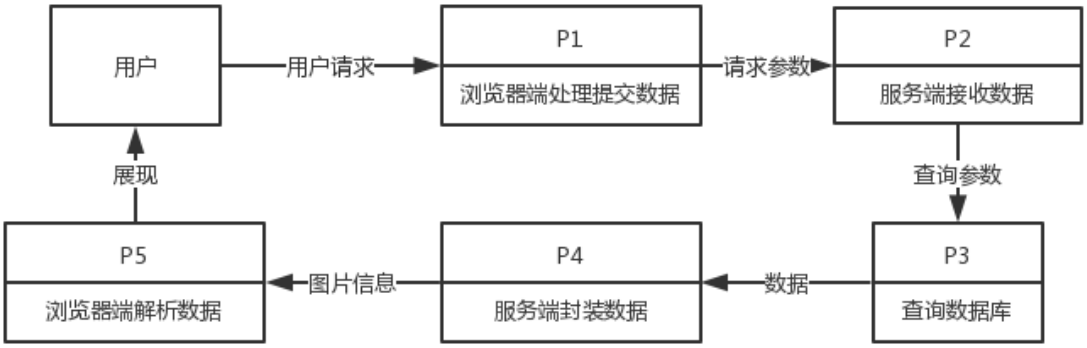

图 4.4 系统 1 层数据流图

4.5.2 数据字典

系统的所有数据结构如表 4.1、4.2、4.3、4.4 所示。

表 4.1 图片类别数据结构

数据项	含义说明	类型	长度	取值范围
编号	唯一标识图片类别	int	11	0—99999999
名称	组织名称	varchar	255	不大于 20 字节
被收藏总数		int	11	0—99999999
图片总数		int	11	0—99999999

表 4.2 用户数据结构

数据项	含义说明	类型	长度	取值范围
编号	唯一标识普通用户	int	11	0—99999999
昵称		varchar	20	不大于 20 字节
性别	男/女	char	2	
手机号		bigint	11	
密码		varchar	255	不大于 255 字节
邮箱		varchar	255	不大于 255 字节
已收藏图片数		int	2	0-99

表 4.3 图片信息数据结构

数据项	含义说明	类型	长度	取值范围
编号	唯一标识图片	int	11	0—99999999
图片作者		varchar	20	不大于 20 字符
上传用户编号	对应用户表中的编号	int	11	0—99999999
图片描述		Text		
图片所属类别	对应图片类别中的类别编号	int	11	0—99999999
图片地址	图片在服务器上的地址	text		

表 4.4 用户收藏数据结构

数据项	含义说明	类型	长度	取值范围
用户编号	对应用户表中用户编号	int	11	0—99999999
图片编号	对应图片表中图片编号	int	11	0—99999999

5 系统设计

5.1 系统体系结构

本系统采用的是 B/S 结构。客户端运行在浏览器上，服务端仅完成提供数据、组件、模板的功能，而将数据、组件、模版转化为网页的任务交到浏览器，除此以外，浏览器还需要完成与用户的交互任务，发送用户的请求给服务器，接收请求结果经分析后呈现给用户。在这种情况下，由于网站整体的前端 DOM 复杂度不高，对前端性能的影响不算明显；而服务端则变得更为轻松，负责数据的管理，只需要基于数据，完成与数据库的交互，并根据用户请求，将封装好的数据返回到客户端。

系统体系结构如图 5.1 所示。

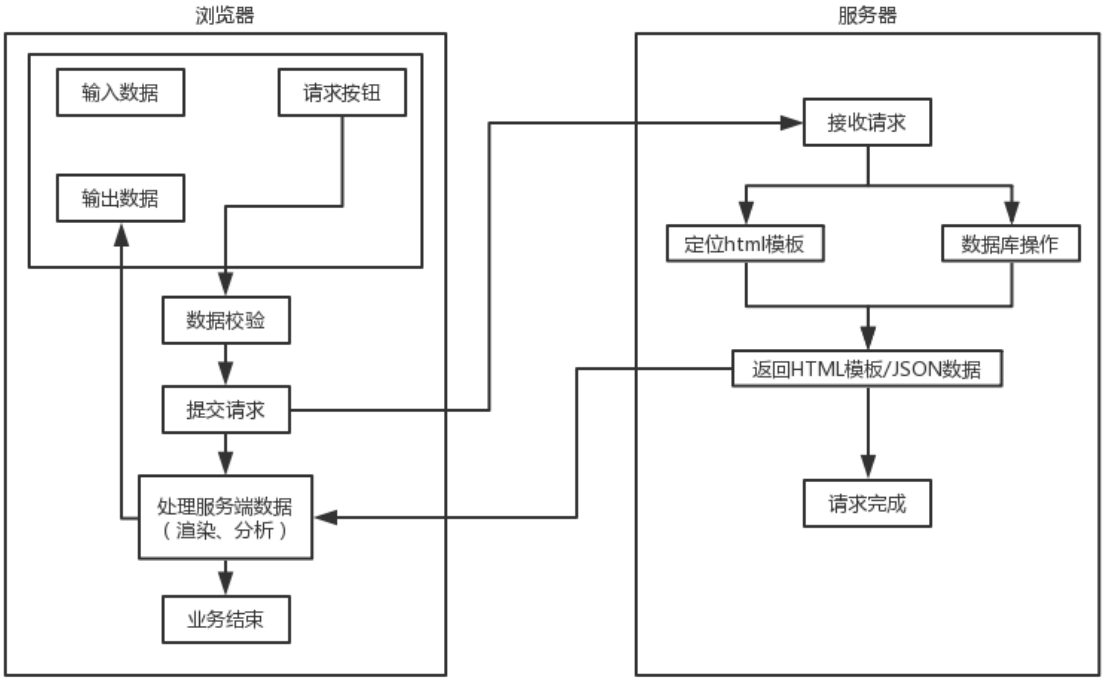

图 5.1 体系参考模型流程图

5.1.1 前端 MVC

如图 5.2，浏览器端采用 MVC 模型，Model 层负责整理数据、发送与接收请求；Controller 层负责逻辑处理，根据用户操作，校对数据、发起请求，以及根据 Model 层收到的数据，对 html 模版进行渲染、将数据绘制到 View 层；View 层负责页面展示、以及交互。

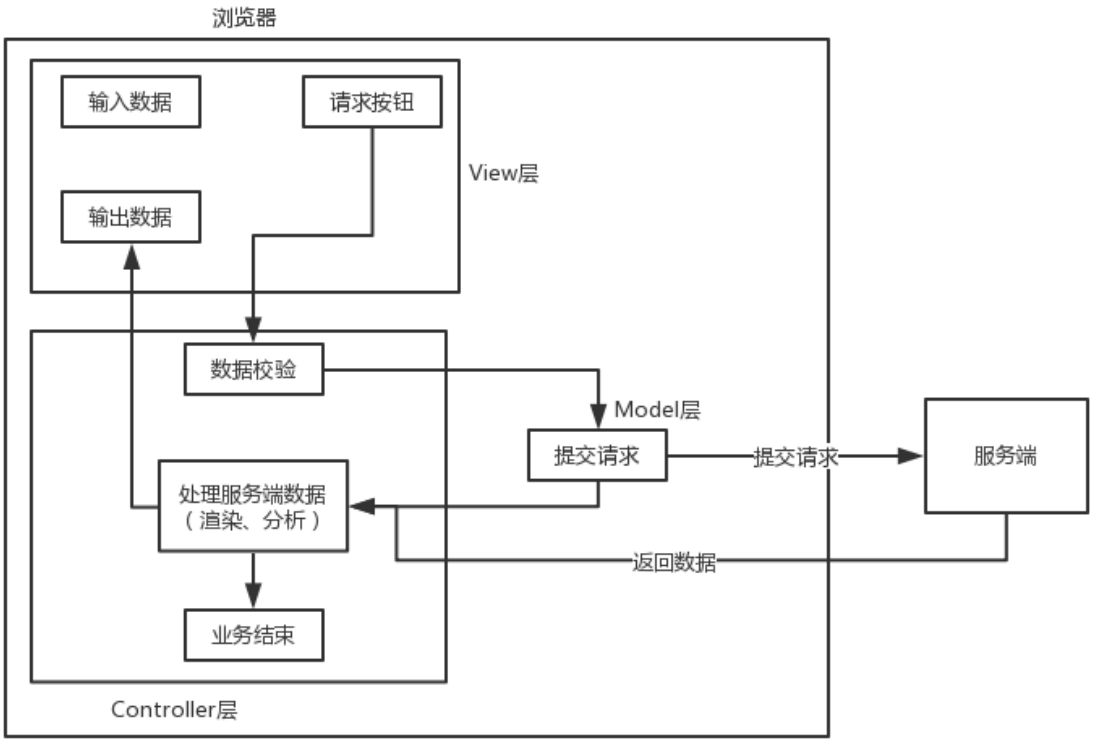

图 5.2 前端体系参考模型流程图

5.1.2 服务端 MVC

如图 5.3，服务端采用基于 NodeJS 的 ThinkJS 框架，是一个 MVC 模型，Model 层负责与数据库的交互；Controller 层负责逻辑处理，根据请求，调动 Model 层，分析数据，整理数据交到 View 层；View 层包括各种组件、html 文件、JSON 数据等。

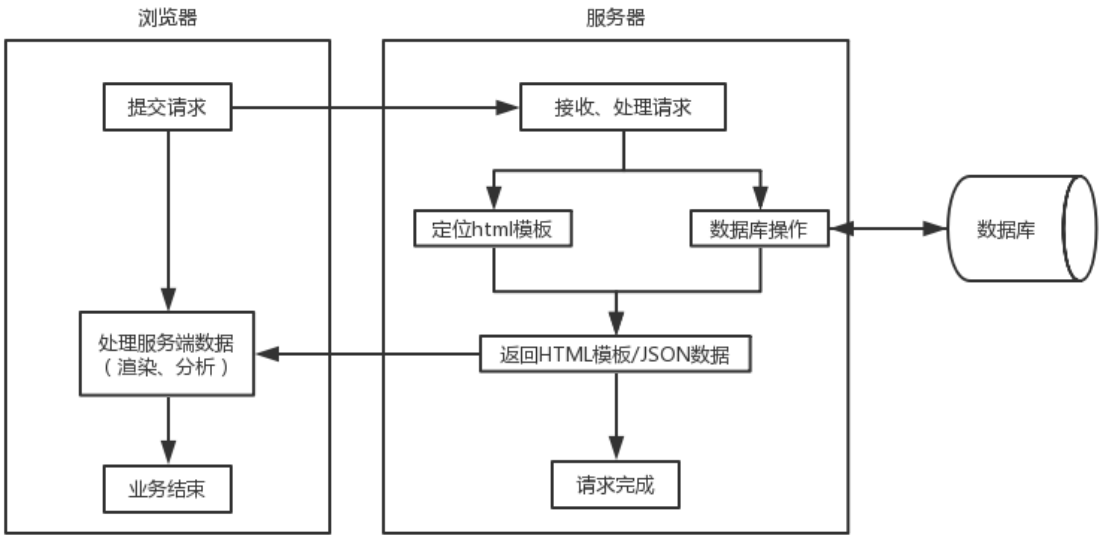

图 5.3 服务端体系参考模型流程图

5.2 软件功能结构

本系统的功能模块如图 5.4 所示，共四大模块。

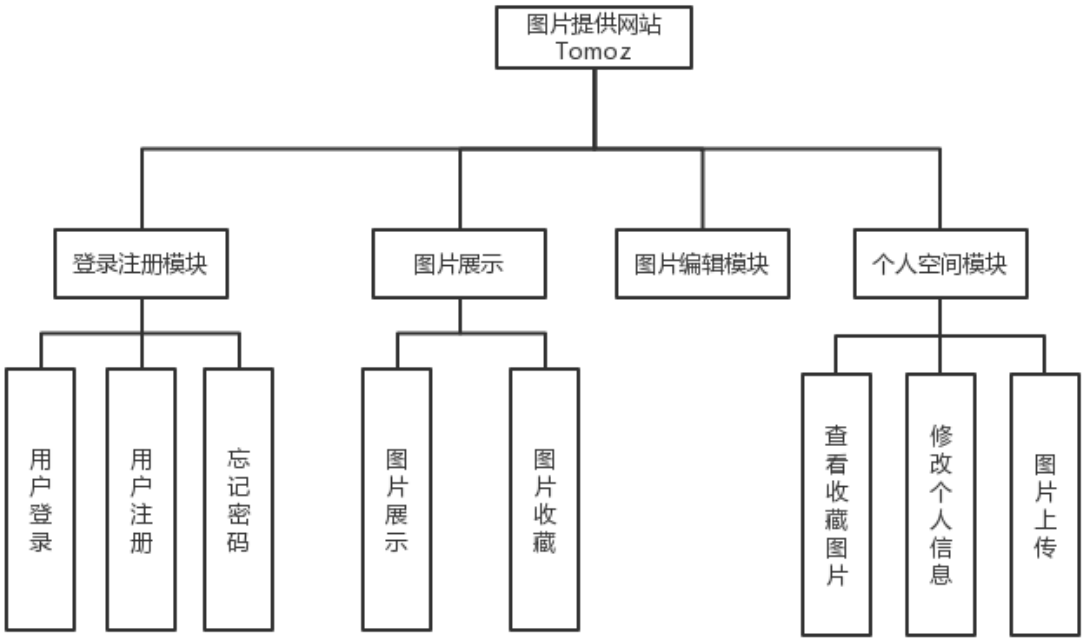

图 5.4 系统功能模块图

5.3 数据库设计

5.3.1 概念结构设计

如图 5.5 所示。其中，用户的图片数量是指用户收藏的图片数量；图片的热度等于收藏该图片的用户数量；类型下的数量是指该类型下的图片总数；类型下的热度是该类型下所有图片的总热度。

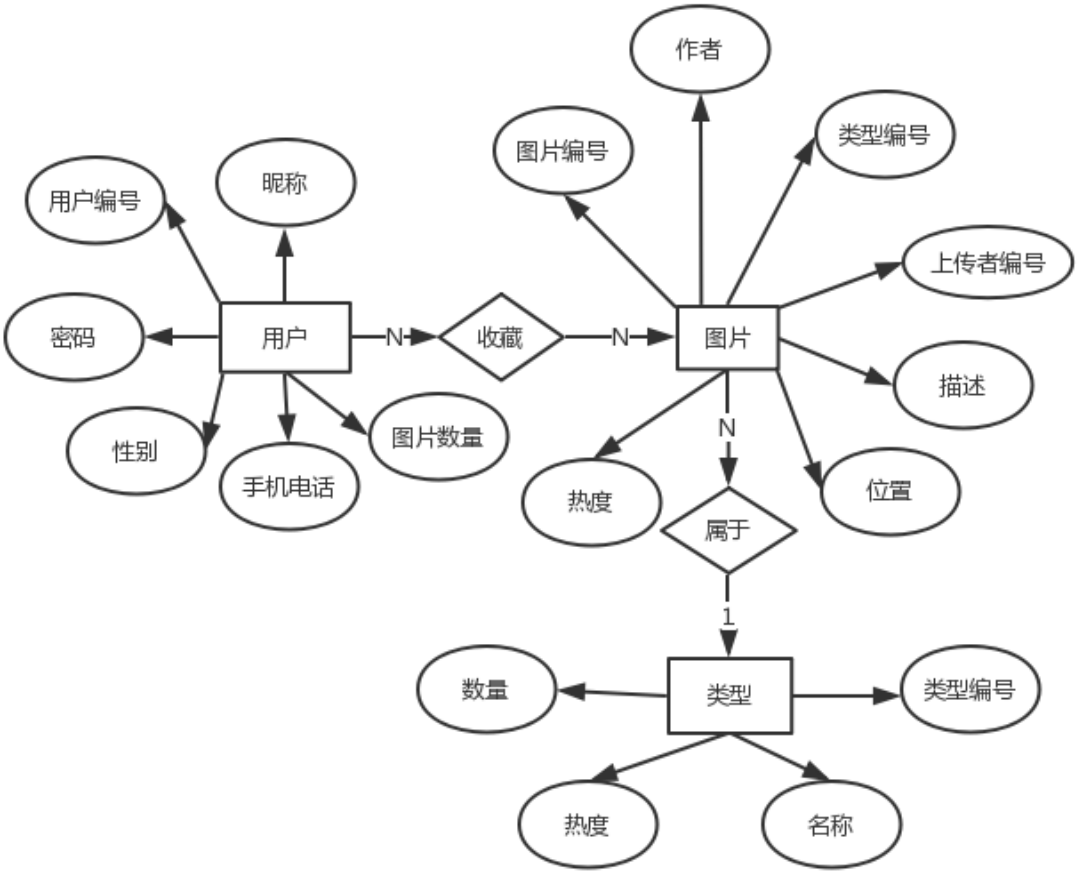

图 5.5 ER 图

5.3.1 逻辑结构设计

系统的逻辑结构设计如表 5.1、5.2、5.3、5.4

表 5.1 用户表

属性名	类型	约束条件	索引	备注
Id	Int	Primary key		唯一标识用户
Nickname	Varchar	Not null		昵称
Sex	Char	F/M		用户性别，F 表示女性，M 表示男性
Cellphone	Bigint	Not null		用户手机号码
Password	Varchar	Not null		用户密码
PictureNum	Int	Not null		收藏图片数

表 5.2 图片类别表

属性名	类型	约束条件	索引	备注
Id	Int	Primary key		唯一标识图片类别
Name	Varchar	Not null		类别名
Heat	Int	Nut null		类别下所有图片的总热度
Num	Int	Not null		类别下图片总数

表 5.3 图片信息表

属性名	类型	约束条件	索引	备注
Id	Int	Primary key		唯一标识图片
UploadId	Int			上传者 ID
Desc	Text	Not null		图片描述
TypeId	Int	Not null、Foreign key	Y	图片类型 ID
Address	Text	Not null		图片位置
Heat	Int	Not null		图片热度

表 5.4 用户收藏图片表

属性名	类型	约束条件	索引	备注
userId	Int	Primary key 、Foreign key	Y	用户 ID
picId	Int	Primary key 、Foreign key	Y	图片 ID

注：“Y”表示是，也即在该属性列上建立索引。

6 系统实现

6.1 登录注册模块

登录流程如图 6.1 所示。登录时，系统会验证用户提供的用户名密码，验证通过之后，返回用户的基本信息到客户端，直到用户退出登录。

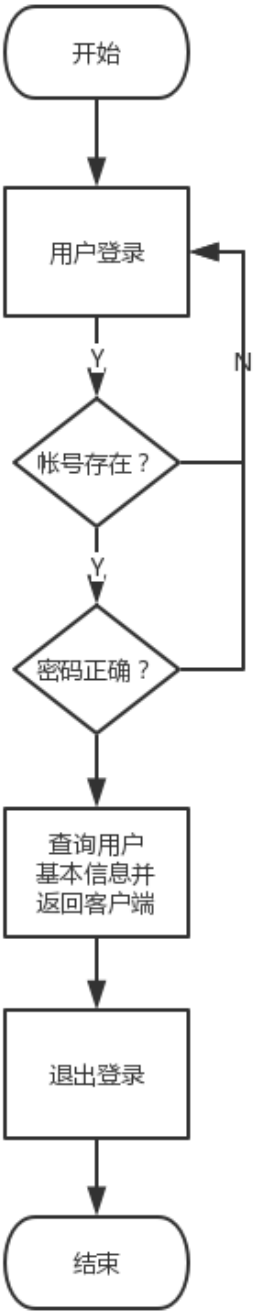

图 6.1 用户登录流程图

用户注册的时序图如图 6.2 所示。用户尚未登录时是游客身份，注册登录成功之后可以对收藏图片、上传图片、管理个人图片和信息。

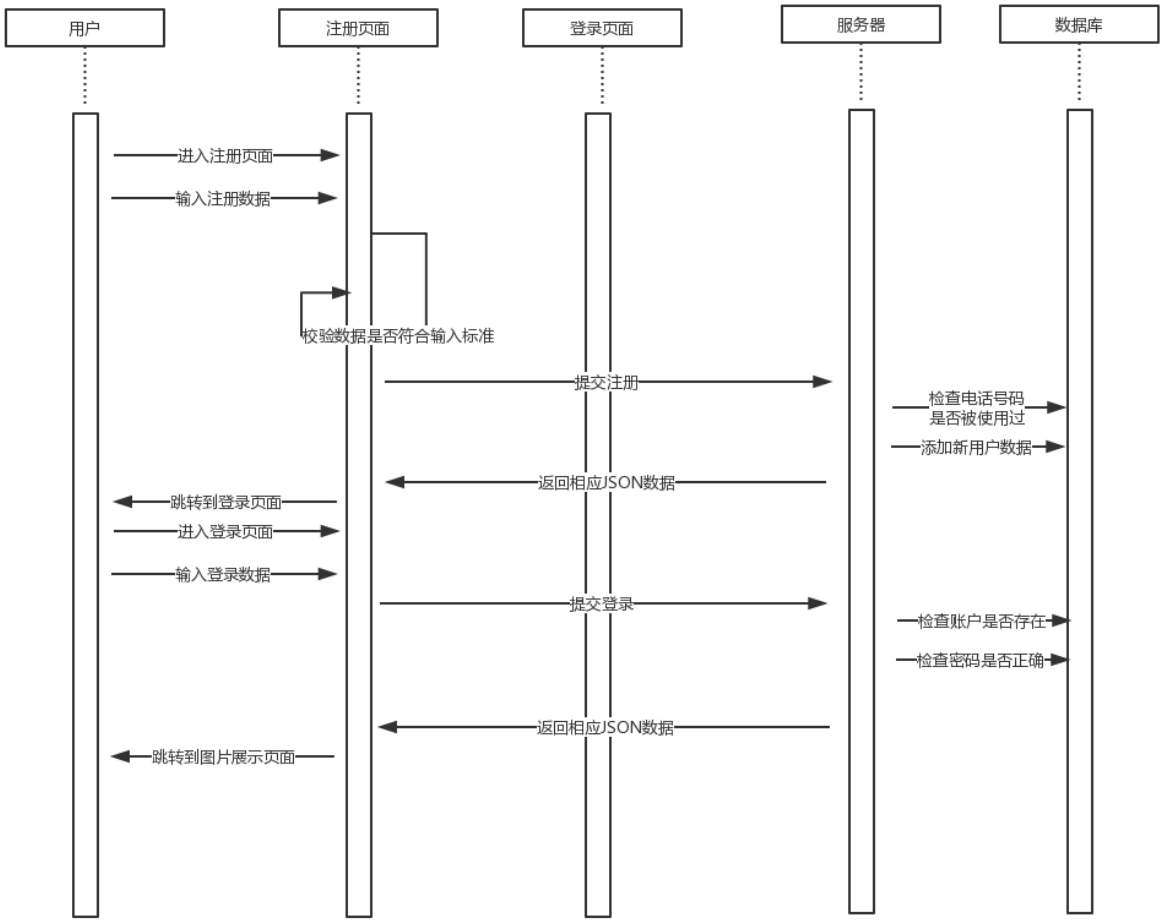

图 6.2 注册时序图

下面是该模块的前端和后端代码实现，如图 6.3、图 6.4、图 6.5 所示。将数据的校验交给前端，后台只检查请求是否合法、session 中 id 是否匹配，若无误则进行数据库操作，取用户密码与请求中的用户密码作对比，若无误则成功登陆，赋予用户新功能。。

```
$("#login").on("click", function () {

    var id = $("#id").val();
    var pwd = $("#pwd").val();
    var data = {
        id: id,
        password: pwd
    }
    debugger;
    $.ajax({
        type: "post",
        data: data,
        url: "/user/logincheck",
        async: true,
        success: function(data) {
```

图 6.3 登录前端代码实现

```
$("#register").on("click", function () {

    var id = $("#users").val();
    var nickn = $("#nickn").val();
    var pwd1 = $("#pwd1").val();
    var pwd2 = $("#pwd2").val();
    var sex = $("#sex").val();
    var data = {
        id: id,
        nickn: nickn,
        pwd1: pwd1,
        pwd2: pwd2,
        sex: sex
    }

    if( checkInfo(data)) {
        submitInfo(data);
    }
```

图 6.4 注册前端代码实现

```
async logincheckAction() {
    if(this.isPost()) {//判断是否以post发送消息给后台了
        let key = this.post();
        let data = await this.model('user').where({"Cellphone": key.id}).find();
        // 查询数据库
        var redata = {
            Id: data.Id,
            Nickname: data.Nickname,
            Sex: data.Sex,
            Cellphone: data.Cellphone,
            PictureNum: data.PictureNum
        }
        if( think.isEmpty(data)) {//查询不到用户，这里直接用think.isEmpty()
            this.json({"errmsg":"用户不存在", "errno":1});//USER_NOTEXIST
        }
        else{//查询到用户则检查密码
            if( key.password == data.Password) {//密码正确，转成功登陆
```

图 6.5 登录后台代码实现

6.2 个人空间模块实现

分为三部分，一部分是选取用户收藏的图片予以展示，另一部分则是提供用户信息修改功能，还有一部分则是图片上传。

展示用户图片收藏流程如图 6.6 所示。如果用户没有登录，是不能看到进入个人收藏的按钮的，用户登录后就可以查看个人收藏。

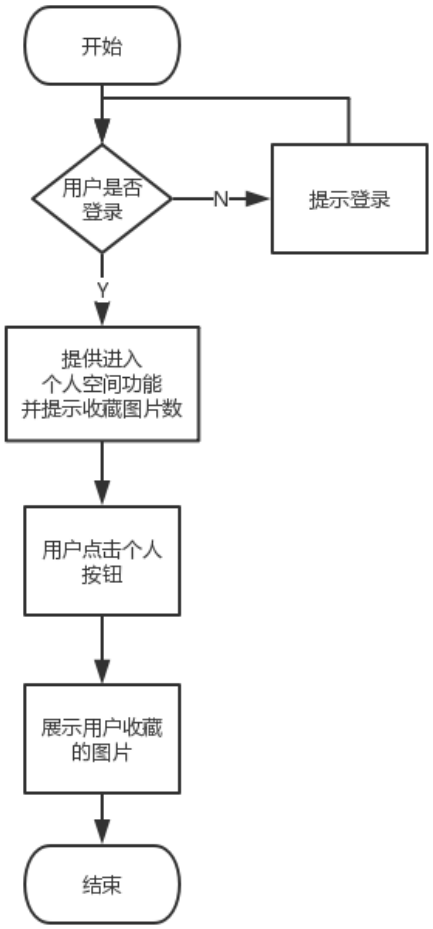

图 6.6 个人收藏查看流程

个人收藏查看的代码实现如图 6.7 所示，取用户 ID 提交到服务器。

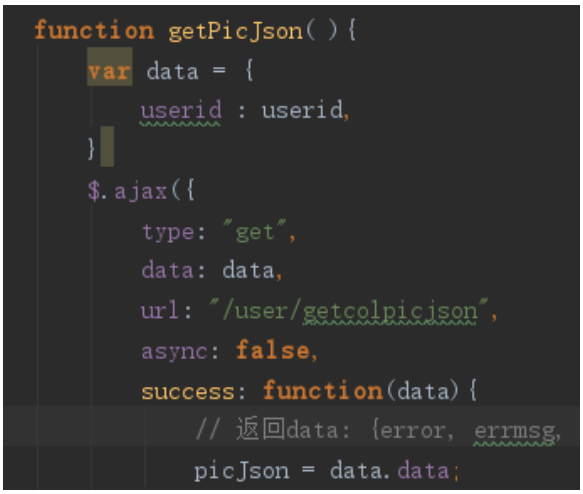

图 6.7 个人收藏查看前端代码实现

后台在收到发来的个人收藏查看请求后，现将该请求中的用户 ID 与 session 中的用户 ID 相比对，若一致后才会做进一步的操作。后端代码实现如图 6.8 所示。

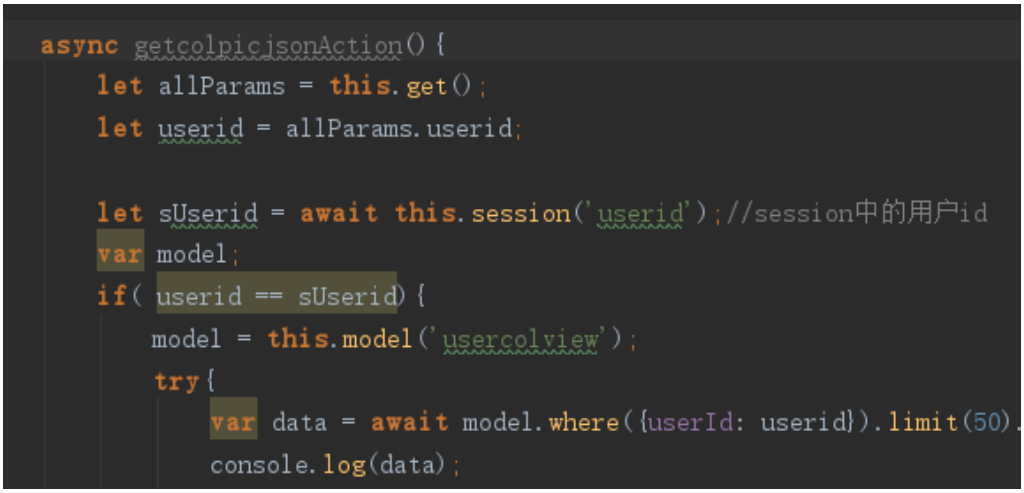

图 6.8 个人收藏查看后台代码实现

个人信息修改功能流程如图 6.9 所示，系统会先展示各个图片类型下收藏量最大的图片给用户看，用户再跟据自己需要，进入到类型下去看图片。

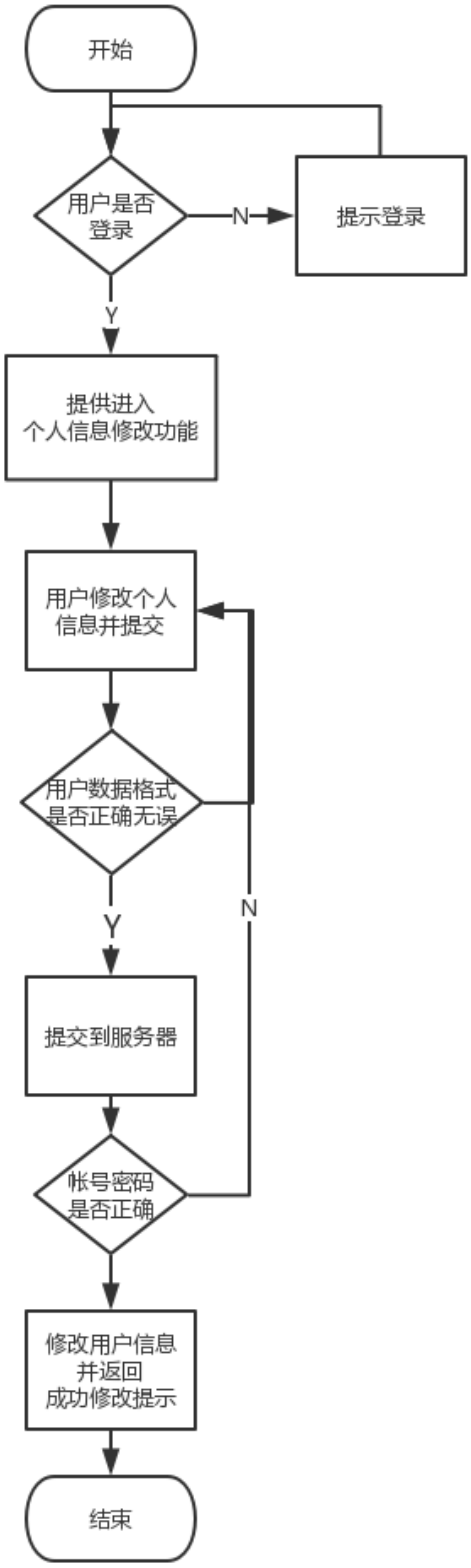

图 6.9 个人信息修改流程

图片展示功能主要代码实现如图 6.10、图 6.11 所示，浏览器会先获取用户提交的信息作检查，再提交到服务器。服务器获取到数据后，检查旧密码是否有误，再修改信息。

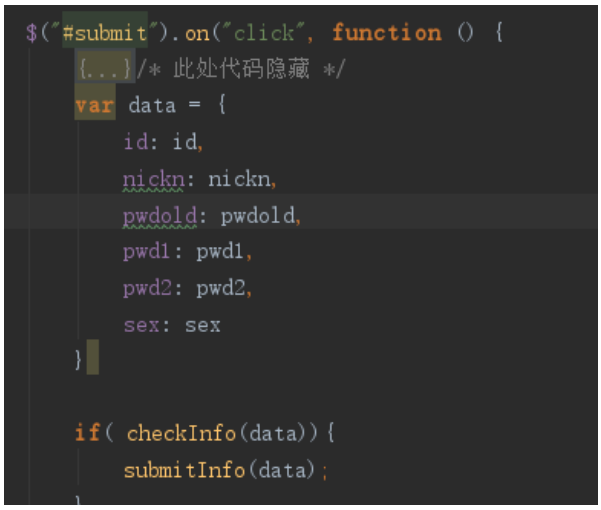

图 6.10 前端获取用户新数据

图 6.11 后端根据前端数据修改用户信息

图片上传的代码实现如图 6.12、图 6.13 所示。前端获取数据并检查，然后包装数据，再提交到服务器。而服务器先检查请求是否合法，再将获取到的用户提交的数据、文件，存入数据库、保存至本地。

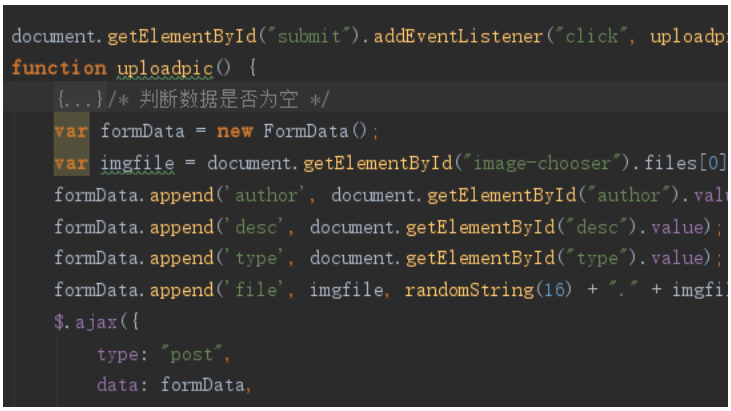

图 6.12 前端获取图片数据并提交

图 6.13 服务端获取图片数据并操作数据库

6.3 图片展示模块实现

分为两部分，一部分是选取所有类型热度排行前 10 的图片予以展示，另一部分则是因应用户选中的图片类别，按图片热度展示图片，还能让用户收藏图片。

图片收藏流程如图 6.14 所示。首先，用户选中自己喜爱的图片进行收藏后，系统首先判断用户是否登录，再判断用户此前是否已经收藏该图片，根据判断的结果给用户以适当的反馈。

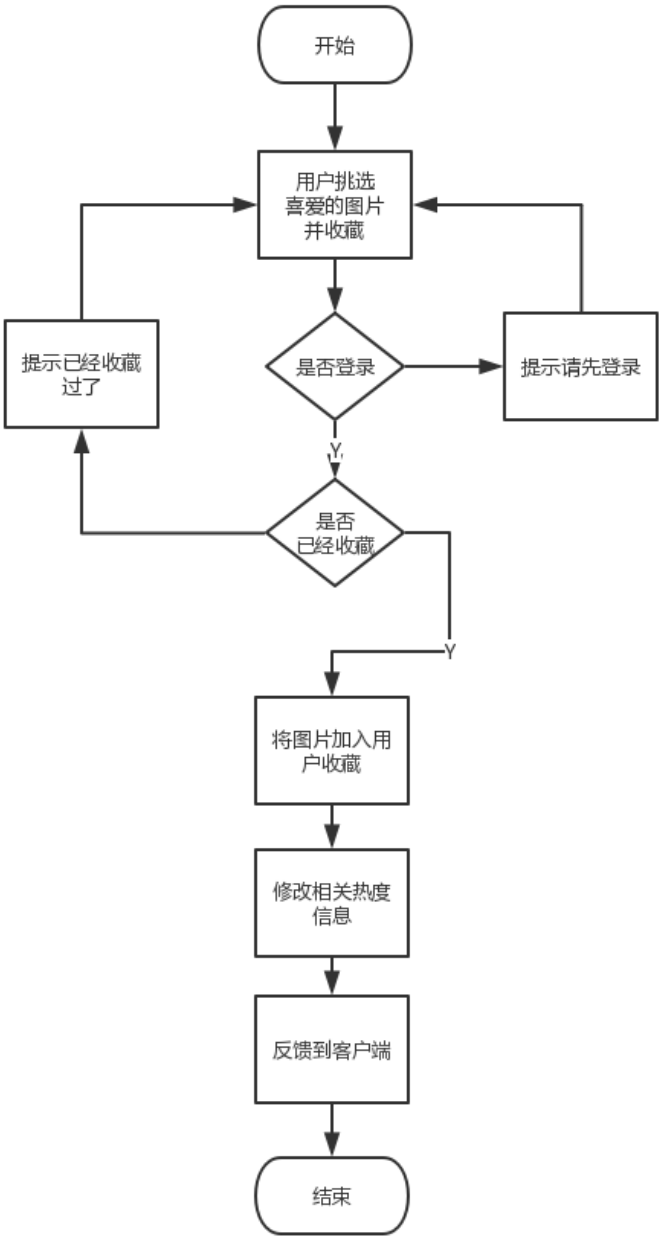

图 6.14 收藏图片流程

收藏功能的代码实现如图 6.15 所示，取图片 ID 和用户 ID 提交提交按到服务器。

```
$(".collectbtn").on("click", function (e) {
    debugger;
    if( cookieExist()){
        var cookieJson = cookieToJson(document.cookie);
        // 取图片id与用户id一并提交到服务器
        var picid = e.target.parentElement.parentElement.p
        var userid = cookieJson.id;

        $.ajax({
            type: "get",
            data: {
                userid: userid,
                picid: picid
            },
            url: "/index/usercollect",
            async: true,
            success: function(data){
```

图 6.15 收藏前端代码实现

后台在收到用户发来的收藏请求后，现将该请求中的用户 ID 与 session 中的用户 ID 相比对，若一致后才会做进一步的操作。后端代码实现如图 6.16 所示。

```
async usercollectAction(){
  let allParams = this.get();
  let userid = allParams.userid,
      picid = allParams.picid;

  let sUserid = await this.session('userid');//session中的用户id
  var model, num;
  if( userid == sUserid){
      model = this.model('usercollection');
      try{
          let insertId = await model.add({
              userId: userid,
              picId: picid
          })

          model = this.model("user");

          try{
              let usercol = await model.where({Id: userid}).find();
              console.log(usercol + "**********" + usercol.PictureNum);
              let upuser = await model.where({Id: userid}).update({PictureNum: usercol.PictureNum+1});//返回
              num = usercol.PictureNum+1;
```

图 6.16 收藏后台代码实现

图片展示功能流程如图 6.15 所示，系统会先展示各个图片类型下收藏量最大的图片给用户看，用户再跟据自己需要，进入到类型下去看图片。

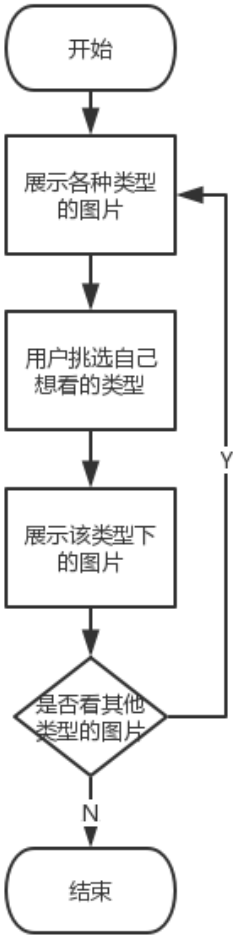

图 6.17 展示图片流程

图片展示功能主要代码实现如图 6.16、图 6.17 所示，浏览器会先获取图片类型 json 数据，再获取类型下的图片数据。

```
(function getPicTypeJson(){...})();

function getPicJson( picType, num){
    var data = {
        picType : picType,
        num : num
    }
    $.ajax({
        type: "get",
        data: {picType : picType},
        url: "/index/getPicJson",
```

图 6.18 前端获取 json 数据代码

```
async getpictypejsonAction(){
  let model = this.model('picturetype');
  let data = await model.page(this.get('page'), 25).order('He
  console.log(data);
  this.success(data)// Q: 对查询到的数据是否为空等检查应该在服务
}

async getpicjsonAction(){
  let allParams = this.get();
```

图 6.19 后端返回 json 数据代码

6.4 编辑模块实现

编辑模块是纯前端的一个模块，旨在为用户提供一个简便的修改网站图片、个人图片的功能。该模块调用了一个轻量级的 js 图片处理插件 chobi.js，能够满足用户基本需求。关键代码如图 6.18 所示

```
document.getElementById("image-chooser").addEventListener("change", function () {
    document.getElementById("resetbtn").removeEventListener("click", addressload);
    loadImage(this);
})

var imgObj = null; //global Chobi object
function loadImage(elem) {
    //you should probably check if file is image or not before passing it
    imgObj = new Chobi(elem);
    imgObj.ready(function() {
        this.canvas = document.getElementById("canvas");
        this.loadImageToCanvas();

        //show filters to users
        document.getElementById("filters").style.display = "block";
    });
}
```

图 6.20 图片编辑模块代码实现

7 系统测试

7.1 登陆注册模块

测试 1：

测试内容：当游客访问登陆页面时，是否能正常出现登陆验证页面。

操作：以游客身份访问登陆页面。

结果：出现了登陆页面。

结论：测试成功。

测试 2：

测试内容：当游客输入正确的账户和密码，能否登入。

操作：以游客身份输入正确的账户和密码。

结果：密码输入框出现一个表示正确的勾，并在短时间内跳转到图片展示页面。

结论：测试成功。

测试 3：

测试内容：当游客输入一个不存在的账户和密码，能否登陆。

操作：以游客身份输入不存在的账户和密码。

结果：密码输入框出现“无此账户”的提示。

结论：测试成功。

测试 4：

测试内容：当游客输入一个存在的账户和错误的密码，能否登陆。

操作：以游客身份输入存在的账户和错误的密码。

结果：密码输入框出现“密码错误”的提示。

结论：测试成功。

测试 5：

测试内容：访问注册页面时，是否能正常出现注册页面。

操作：访问注册页面。

结果：出现了注册页面。

结论：测试成功。

测试 6：

测试内容：当游客输入符合规范的相关信息，能否注册。

操作：以游客身份输入符合要求的相关信息。

结果：提示成功，并在短时间内跳转到登陆页面。

结论：测试成功。

测试 7：

测试内容：当游客输入一个此前用过的电话作为自己的电话，能否注册。

操作：输入一个此前用过的电话作为自己的电话。

结果：出现电话号码重复提示，并将页面焦点转到该地方。

结论：测试成功。

测试 8：

测试内容：当游客输入不符合输入规范的信息，包括低强度的密码、错误的手机号码等。

操作：以输入不符合输入规范的信息。

结果：若输入的是低强度的密码，将提示密码的强度要求；若输入的不受正确的手机号，将提示手机号的输入标准。

结论：测试成功。

7.2 图片展示模块

测试 1：

测试内容：分别以未登录的游客和已经登陆的用户身份访问图片展示首页面和次级页面时，是否能正常出现页面。

操作：访问图片展示页面。

结果：出现了图片展示页面。

结论：测试成功。

测试 2：

测试内容：在首页面，将鼠标移至图片上，能否出现图片描述预览。

操作：将鼠标移至图片上。

结果：出现图片描述。

结论：测试成功。

测试 3：

测试内容：在首页面，点击图片，能否正常跳转到次级页面的相应图片展示上。

操作：点击图片。

结果：跳转到次级页面，并浮现刚刚点击的图片和描述。

结论：测试成功。

测试 4：

测试内容：以游客身份点击收藏按钮，能否收藏

操作：以游客身份点击次级页面的收藏按钮。

结果：出现请先登陆的提示。

结论：测试成功。

测试 5：

测试内容：以用户身份点击收藏按钮，能否收藏。

操作：点击收藏按钮。

结果：会出现“此图片此前已经收藏”或“收藏成功”的提示。

结论：测试成功。

测试 6：

测试内容：以用户、游客身份分别点击图片下的编辑按钮，能否正常跳转到编辑按钮并加载该图片。

操作：分别以两种身份点击图片下的编辑按钮。

结果：均跳转到图片编辑按钮并加载刚刚那张图片。

结论：测试成功。

测试 7：

测试内容：分别点击下一张图片、下一页图片，能否正常跳转。

操作：分别点击下一张图片、下一页图片。

结果：加载下一张、下一页的图片。

结论：测试成功。

7.3 图片编辑模块

测试 1：

测试内容：访问图片编辑页面。

操作：访问图片编辑页面。

结果：出现了图片编辑页面。

结论：测试成功。

测试 2：

测试内容：上传图片。

操作：点击上传图片按钮。

结果：弹出上传图片选择框。

结论：测试成功。

测试 3：

测试内容：测试图片编辑的亮度、对比度修改功能。

操作：点击亮度+1/-1、对比度+1/-1 按钮。

结果：亮度修改工作正常，对比度修改不明显。

结论：测试成功。

测试 4：

测试内容：测试图片编辑的滤镜功能。

操作：点击各个滤镜。

结果：均正常工作

结论：测试成功。

7.4 个人空间模块

测试 1：

测试内容：以用户身份分别访问个人信息修改页面和个人图片收藏页面。

操作：登陆后，访问个人信息修改页面和个人图片收藏页面。

结果：出现了图片编辑页面。

结论：测试成功。

测试 2：

测试内容：输入符合标准的数据修改个人信息。

操作：输入符合标准的数据修改个人信息。

结果：提示修改成功，刷新页面后可见页面中用户昵称已经改变。

结论：测试成功。

测试 3：

测试内容：输入有误的数据修改个人信息，如性别不填写、错误的旧密码、两次输入不一致的新密码等。

操作：依照以上测试内容输入数据。

结果：弹出错误提示。

结论：测试成功。

测试 4：

测试内容：用户查看以前收藏的图片。

操作：点击个人按钮

结果：能够看到此前收藏的图片

结论：测试成功。

测试 5：

测试内容：用户上传图片。

操作：进入到上传图片界面，输入符合规范的数据并提交。

结果：弹出成功上传的提示，在其他图片展示模块能够找到刚上传的图片。

结论：测试成功。

测试 6：

测试内容：用户上传图片输入有误的数据，如上传非图片文件、输入为空等。

操作：依照上述要求输入数据并提交。

结果：当上传的文件不是图片时，会弹出请输入图片的提示；当输入为空时，也会弹出提示。

结论：测试成功。

7.5 补充测试

测试 1：

测试内容：登陆后，页面头部显示应与游客不同，拥有更多功能，展示更多的数据。。

操作：登陆后，查看页面头部。

结果：页面头部工作正常。

结论：测试成功。

测试 2：

测试内容：登陆后，页面头部显示用户此前收藏的图片。

操作：登陆后，查看页面头部。

结果：头部能够看到用户此前收藏的图片数量。

结论：测试成功。

测试 3：

测试内容：各页面跳转与链接。

操作：分别用游客、用户两个角色在页面中跳转，测试链接。

结果：各页面链接正常。

结论：测试成功。

8 系统安装使用说明

8.1 安装与环境配置

8.1.1 服务端安装配置

服务端需要在 NodeJS 环境下安装运行，请先到 nodejs.cn 下载 nodejs 安装。如果此前安装过 nodejs，请确保已经安装 npm 包管理工具，最新版的 nodejs 已经一起附带安装。

从 github、光盘或其他渠道获得的项目文件通常没有为项目运行安装依赖文件，请在安装好 npm 后，在项目根文件夹目录下打开命令提示符 CMD，输入 npm install，npm 将为您自动安装好项目运行所需的依赖文件，此过程可能需要较长时间，有需要的话建议百度 npm 淘宝镜像进行修改。

安装好上述文件和环境后，在项目根文件夹目录下打开命令提示符 CMD，输入 npm start，项目开始运行，服务器启动完成界面如图 8.1 所示：

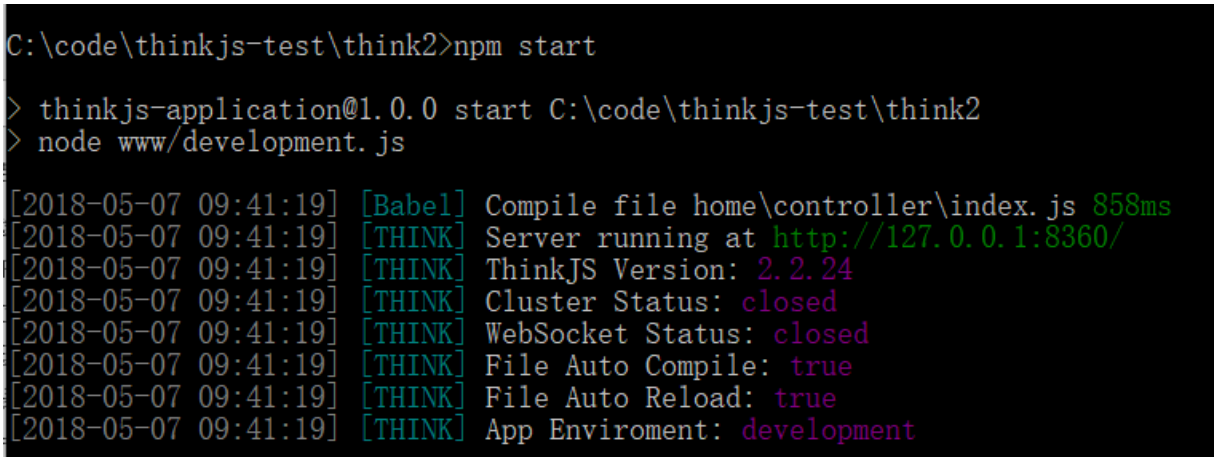

图 8.1 运行项目页面

8.1.2 客户端安装配置

本系统是 B/S 结构的系统，客户端只需要拥有浏览器即可。但为了系统能够提供更好的用户体验，建议客户端安装最新的 Chrome 浏览器。

8.2 使用说明

点击“现在就进入 Tomoz”进入系统，打开网站的初始界面如图 8.2。

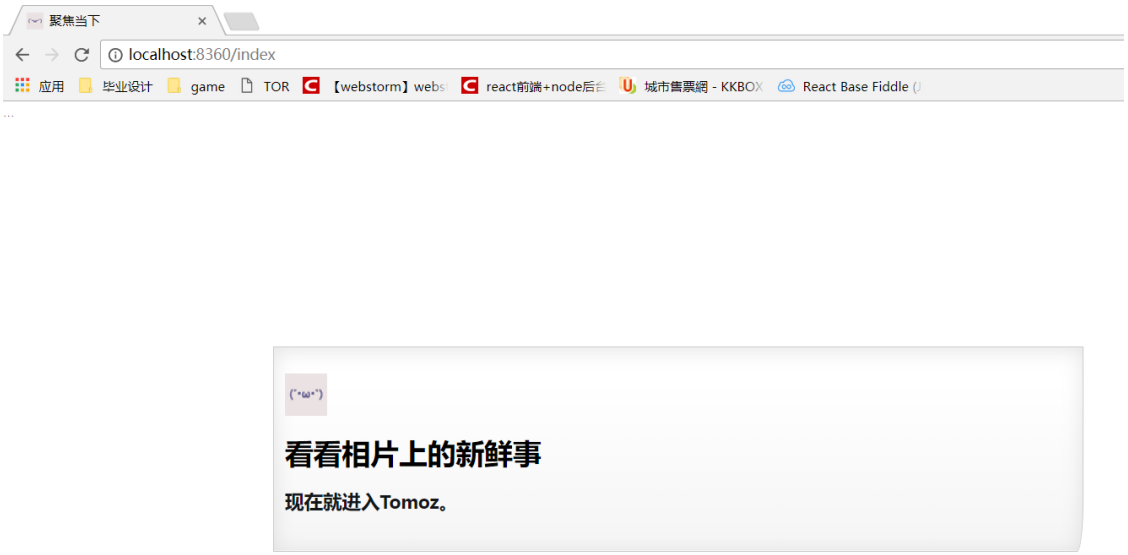

图 8.2 运行项目页面

接下来可以自由地浏览图片，将鼠标移到图片上可以看到图片的一点介绍，想了解更多或想看大图的话点击图片即可，如图 8.3 所示。

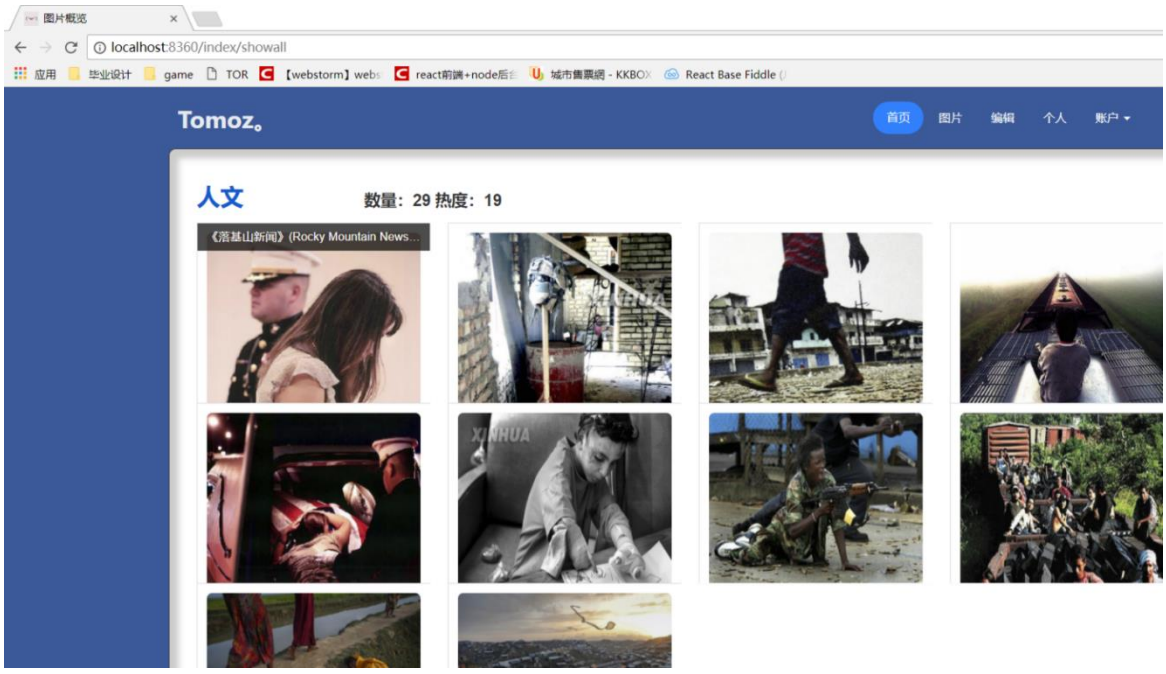

图 8.3 图片展示模块页面

点击图片后会发生跳转，从而看大图和看详细的图片介绍，如图 8.4 所示。

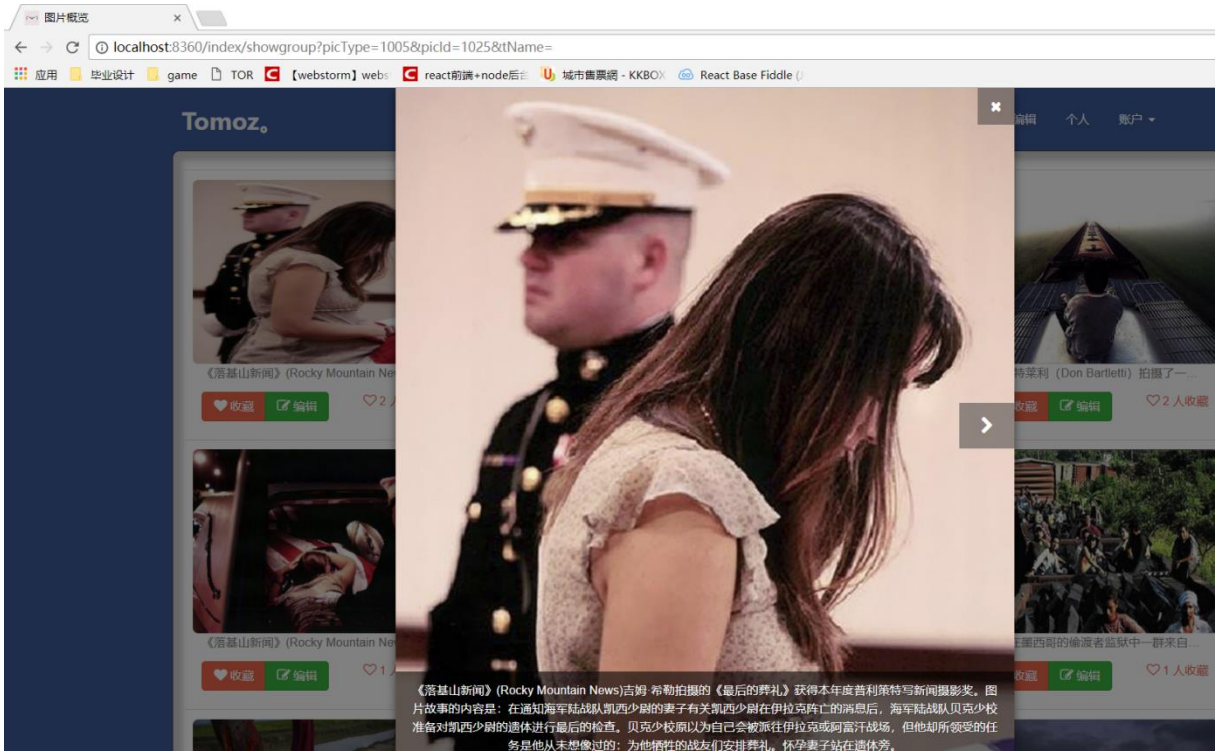

图 8.4 图片展示模块页面

关闭掉图片之后可以选择自己喜欢的图片进行收藏，如果没有登录，将弹出提示要求登录，效果如图 8.5 所示。

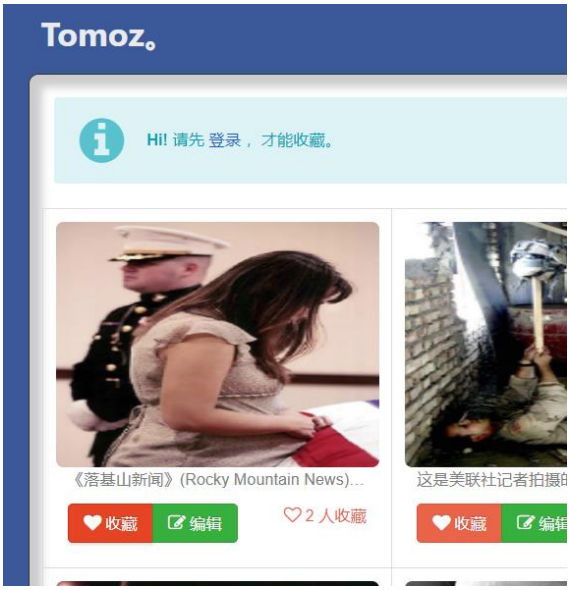

图 8.5 登陆提示

而图片编辑功能则无论是否登录都可以使用，进入到图片编辑页面后，用户可以根据自己需要，对图片做简单的修改，修改后点击下载图片可以下载；也可以选择上传自己的图片进行编辑，界面如图 8.6、图 8.7 所示。

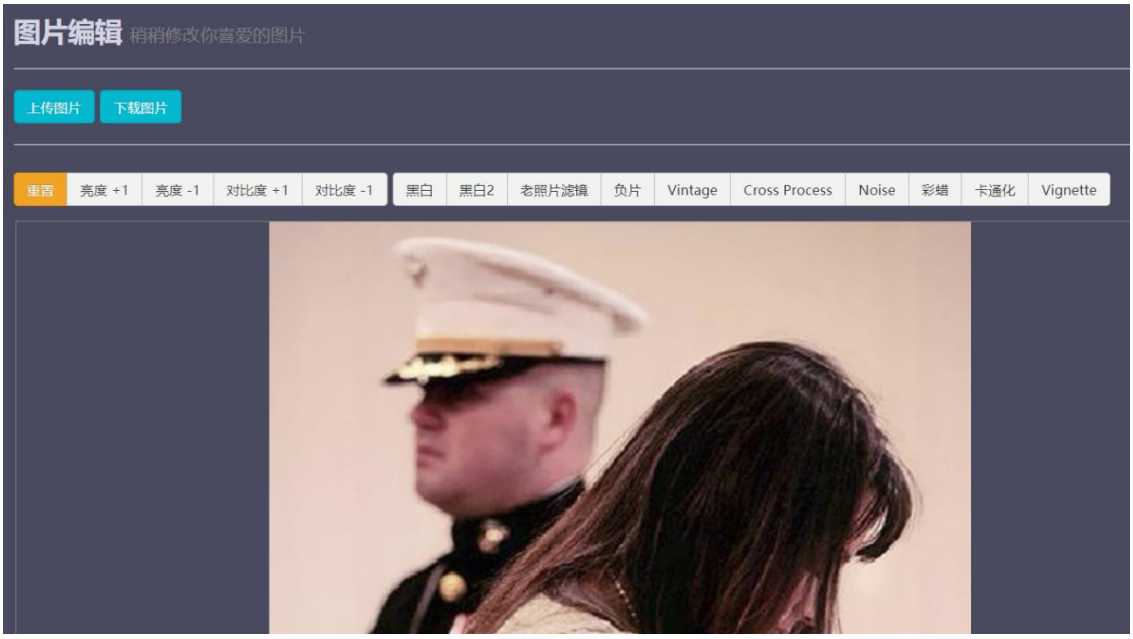

图 8.6 图片编辑页面

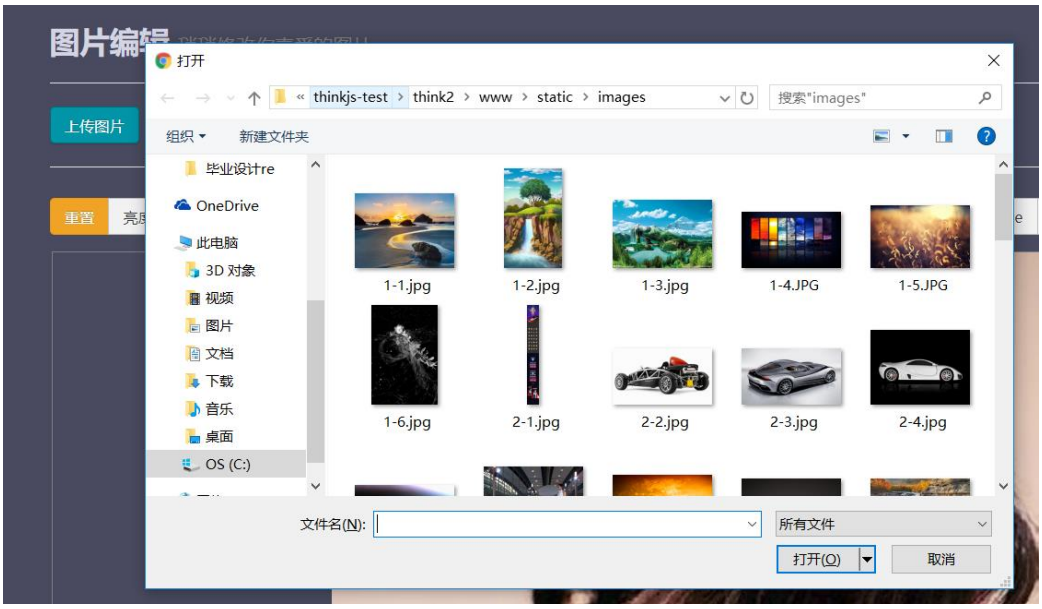

图 8.7 上传个人图片进行编辑

登录后，才能收藏自己喜爱的图片，并上传图片，登陆与注册页面如图 8.8、图 8.9 所示。

图 8.8 注册页面

图 8.9 登录页面

登陆后，我们可以看到，在页面头部有自己收藏的图片数的提示，点击进入个人收藏页面，如图 8.10、图 8.11 所示。

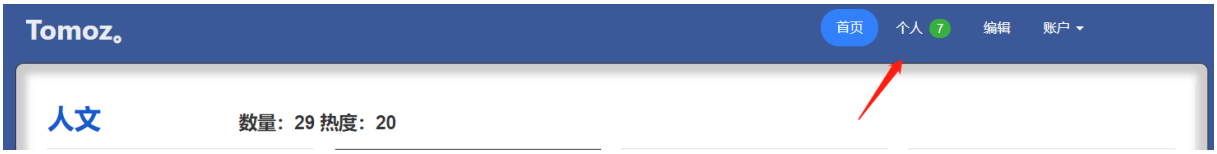

图 8.10 个人收藏页面入口

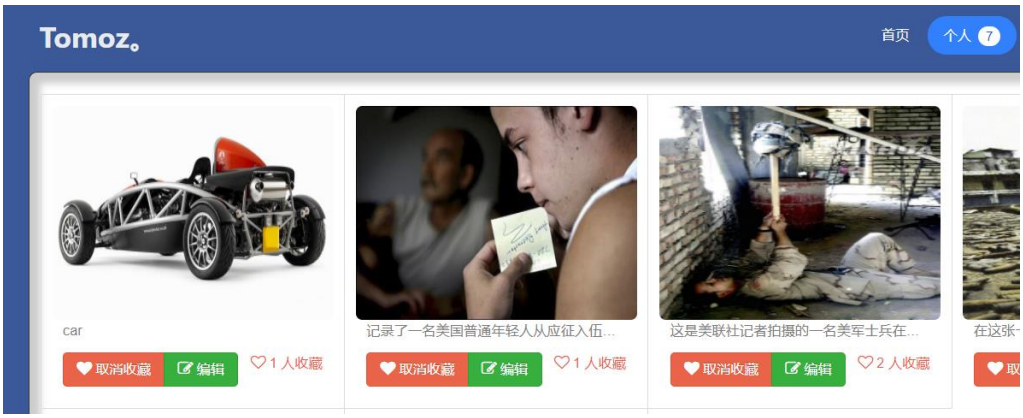

图 8.11 个人收藏页面

登陆后还可以上传图片，如图 8.12、图 8.13 所示。

图 8.12 个人收藏页面

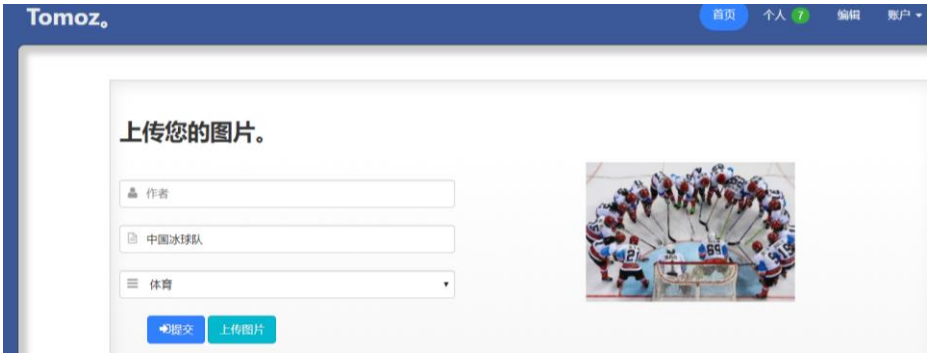

图 8.13 图片上传页面

9 心得体会

本论文主要论述了基于 React 的一个图片提供网站的设计与实现。整个系统前端建设以目前前端的新兴技术 React 作为核心，配合 zui 框架、jQuery 库制作，将网页渲染的任务全部交给浏览器；网站后台采用基于 NodeJS 的开源框架 ThinkJS，负责沟通数据库、提供数据。于前端，相比于 AngularJS、backboneJS，React 非常的年轻；于服务端，相比于 Ruby on Rails、SSH 等技术，ThinkJS 和 NodeJS 方兴未艾。他们两者，文档、社区都还不是非常齐全，我使用他们更多的是抱着学习、探索的想法。在编写和设计的过程中，我感受到了 React 在思想上的简洁，使用上的方便；感受到 thinkjs 的强大，对开发效率的提高。通过参考花瓣网、twitter、tumblr 的设计，我了解到很多关于图片网站的知识：如图片作为信息媒介的价值与能力、图片价值如何挖掘、图片在社交中的作用；通过实现网站及编写论文，我对 React、thinkjs 有了更深的理解：如何分模块、如何设计数据库表、如何划分前端模块、如何提高代码重用性与可维护性，等等。

本系统的特点如下：一、采用了基于 NodeJS 的 ThinkJS 框架，是一个能够使用 ES6/7 全新特性开发的 MVC 框架。同时，NodeJS 的完全异步 IO 模型能够极大地提高 web 服务的并发性；二、采用 React 作为前端核心，组件化地构成前端页面，采用虚拟 DOM 提高了页面性能。三、系统按照 MVC 架构设计，包括 View 层、Controller 层、Model 层，便于系统的维护和变更。四、系统将所有 html 文件的渲染交给浏览器完成，极大地减轻了服务器的负担。

本系统目前发现的不足：一、部分的页面有些单调，不够美观。二、将所有的 html 渲染交给浏览器完成固然减轻了服务器的负担，但浏览器的负担过重。一开始设计的时候只考虑到系统 DOM 结构不算复杂，页面延迟不会高，但实际上，还要考虑到文件和依赖需要请求和下载所需要花费的时间，导致了系统首加载的延迟比预想要高；三、采用的前端渲染方式对搜索引擎不友好；四、系统目前仅仅是把基本的功能实现，还需要继续完善。

改进方向：一、减少客户端渲染，增加服务端渲染的比重；二、压缩 js、jsx、css 文件的大小，利用 webpack 等工具使之按需加载。

非常感谢能有这样的一次机会，让我大胆地尝试和实践，相信这次的毕业设计能为我带来巨大的进步。

参 考 文 献

[1] 刘波 .关于 B/S 和 C/S 架构的分析[J].人民检察,2004(09):77-78.

[2] 庄严.基于 B/S 结构的软件开发技术分析[J].电子制作,2016(24):44.

[3] 向雨濛. 中美图片社交产品对比研究[D].浙江传媒学院,2016.

[4] 百度百科 . react 词条 [EB/OL] . https://baike.baidu.com/item/react/18077599?fr=aladdin 2017-10-15/2018-5-10

[5] 朴灵．深入浅出 Node.js [M]．北京：人民邮电出版社 / 2014-08．19-29

[6] ThinkJS 官方文档．介绍 [EB/OL] . https://thinkjs.org/zh-cn/doc/2.2/index.html 2017-9-14/2018-5-11

[7] 郭彦辉.响应式 Web 设计的研究与实现[J].软件,2018,39(01):169-172.

[8] 田红玉.响应式网页设计与实现[J].电子测试,2016(03):80+79.

[9] 维基百科 . MySQL 词条 [EB/OL]. http://zh.wikipedia.org/zh/mysql 2018-5-10/2018-5-11

[10] 简书 . React 简单介绍 [EB/OL].https://www.jianshu.com/p/ae482813b791 2014-12-10/2018-5-11

[11]陈胤梁. jQuery 插件为 Web 应用带来的变化[J]. 计算机光盘软件与应用, 2013, (05):9-11.

[12]周兴宇,卞佳丽. 基于 React 的前端组件化研究与设计[EB/OL].北京：中国科技论文在线.http://www.paper.edu.cn/releasepaper/content/201601-80 2016-01-05/2018-5-12

[13] Tali Garsiel. How browsers work[OL].[2015]. http://taligarsiel.com/Projects/howbrowserswork1.htm 2015/2018-5-18

[14]Responsive Design in 3 Steps [EB/OL]. http://webdesignerwall.com/tutorials/responsive-design-in-3-steps 2011-12-8/2018-5-12

致 谢

衷心感谢导师何晓桃老师对我的精心指导，为我的毕业设计的顺利完成打下了良好的基础，在此对何老师表示由衷的感谢。

感谢在大学四年来的各位老师、辅导员，你们不仅给了我知识，还教会了我很多方法和做人的道理，谢谢！

感谢我的家人和朋友，感谢你们的支持和帮助，谢谢！

感谢我的母校广工，让我能够在大学四年遇到这么多同学和朋友，增长了才干和学识，广工团结、勤奋、求是、创新的校训我将永记于心。