基于 Vue 的 TaoKe 网站设计与实现

李钰玲 黄媛媛
四川大学锦城学院 计算机与软件学院 四川 成都 611731

【摘 要】本文是基于前端框架 Vue 设计并开发的一个在线购物商城类网站。在本网站中，用户可以在商品列表查看商品信息、将商品添加进购物车、查看购物车列表、购买商品、查看订单明细等 。TaoKe 网站以“更简洁，更便捷”的理念而生，是一款简单操作，简洁界面的网站。该网站基于 Vue.js 框架，后端使用 Node.js、数据库选中 MongoDB，前后端交互使用 Ajax 技术，是一个轻型在线购物商城类网站。

【关键词】前端；VUE Ajax； MongoDB

1 引言

在最近的这几年里，在大数据时代以及信息技术和科学技术的不断进步与发展的背景下，互联网行业的不断兴起和发展，伴随着电子商务的地不断完善，如今人们的日常生活都可以通过在线的形式得到一定的实现。在现代生活发展趋势中，网络购物已成为了人们生活的一部分。因此，本网站从某种意义上来说，让喜爱在线购物的群体得到更好的体验。各类潮流的服饰，先进的电子产品，多元化的商品整合在一起。本文则是通过对开发过程中技术选型介绍，可行性分析，需求分析以及功能设计等板块，简略地介绍此网站的设计和开发流程。

2 关键技术选型介绍

本次开发过程中，网站的开发和建设运用前后端分离的方法，关键技术选型如下：

Vue.js 是一套渐进式的自下而上的 JavaScript 框架，视图层是它唯一关注的对象，尽可能地通过便捷的方式实现数据绑定以及对组件的使用 [1]，开发简便，是一个轻量级的框架，对熟练掌握的要求也不高。

Element UI 是一套比较完备的组件库，种类繁多，在响应式布局的开发中与 Bootstrap 有一定的结合，增强了页面的自适应效果。在此次开发中，我们在商品的商品列表的显示方式上，使用了 Element UI 下面的 vue-infinite-scroll 插件，进行对商品无限加载的功能，达到一种用户简洁操作的效果。

Ajax（异步 JavaScript 和 XML），实现异步处理的数据更新，从而提高了响应的效率 [2] 。在优化信息的传输以及数据的交换等方面的速度上扮演着十分重要的角色。

MongoDB，是一种在分布式文件存储基础上的数据库，目的在于提供可扩展的高性能存储解决方案 [3]。在当前 Nosql 数据库中使用率比较高，具有很好的扩展性，很好的存储速度，以及存储可以是 JSON 格式。下图为此次开发设计的主要技术选型：

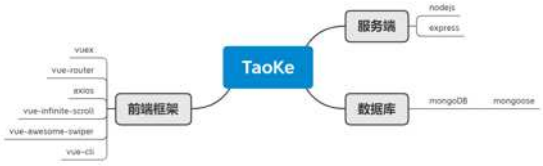

图 1 技术选型图

3 可行性分析

3.1 技术可行性

Vue.js 在前端的开发中，兼并了 React.JS、Angular.JS 的长处，也同时剔除了它们的不足，取其精华，去其糟粕，使其成为当下 Web 应用程序开发过程中的首选框架。适用于轻量级以及微型应用网站的开发，更是其开发的首要选择。

Ajax，适用于本网站对前后端交互的要求，通过 Ajax，前后端数据的相互传输变得更高速，也是一个很好上手的前后端交互的技术。

MongoDB 对数据的处理，以及数据存储格式的灵活，在对后台数据的存储的选择上，具有很好的优势，适用于本网站后端开发的技术选择。

综上，技术可行。

3.2 经济可行性

对于本次应用程序的开发，工具的使用是个人笔记本，选择终端测试语言进行开发，任意浏览器开发者工具，用于检查错误以及基本的测试。综上，经济上可行。

4 需求分析

立足于对市场用户的调查，网站的受众群体主要为青年人，由于他们忙碌的工作，促使他们没有过多的时间去逛街，所以会更可能地选择在线购物的方式来购买生活的一些必备品。对于在线商城网站的页面设计，用户往往更倾向于界面简洁，操作简单，可读性高的设计风格，并且能够满足在线商城的购物功能的需要。市场调查的分析结果，是我们进行设计与开发的依据。在本次的网站设计与开发过程中，主要是针对 PC 端的开发，参考了市面上同类网站的配色、页面布局等方面，需求主要分为：新用户注册，用户账号登录，用户退出登录，用户地址信息管理，商城的商品列表，用户订单信息管理，购物车详情等。

5 设计及实现

5.1 功能模块划分

下图为本网站的主要功能模块：

图 2 功能模块图

目的在于为用户提供一个操作简单，界面简洁的在线商城类网站。浏览网站时，当用户以游客的身份，只能查看和浏览商品列表的商品，但是不允许直接对商品进行加入购物车的操作，也不能直接进行对购物车的查看和浏览等后续的操作。功能模块只对游客选择性部分开放。如下为功能模块：

5.2 功能设计

5.2.1 用户注册 / 登录 / 退出

本功能面向第一次进入商城或者没有商城账号的用户（已经注册过的用户可以忽略此操作，直接进行登录操作）。

用户进入商城，单击顶部“注册”字样的超链接进行注册（注册需要填写的内容包括：用户名，密码，确认密码）。

用户注册成功，此过程向后台发送请求消息，服务器接收信息，将用户信息存储进数据库，对数据库的用户表进行添加一条用户数据的操作，并且实时更新用户表信息，注册成功后，可以通过注册成功的用户名和密码进行登录操作，登录实则是一个对数据库的查询操作，查询数据库是否存在此用户相关的信息。若数据库中存储了该用户的相关信息，通过判断用户的登录信息是否正确，并返回相应的消息体；若不存在该用户，则返回相应的错误提示。

用户登录成功以后，可以随时点击顶部“退出”字样，从而退出当前登录。

用户退出，此过程向后台发送消息，后台服务器接收消息，清空用户的 cookie 信息，并且将路由定位到首页，即退出成功。

用户退出登录功能的实现，关键代码如下：

前端发送请求：

```
logout() {
    axios.post(“/users/logout”).then(res => {
      if (res.data.code === 0 && res.status === 200) {
        this.$store.commit(“updateinfo”, “”);
        this.loginstatus = false;
        if (this.nopublicpath.includes(this.$route.path)) {
          this.$router.push(“/”);
        }
      }
    });
},
```

后端响应请求：

```
router.post(‘/logout’, (req, res) => {
  res.cookie(‘usersid’, ‘’, {
    path: ‘/’,
    maxAge: -1
  })
  res.json({
    code: 0,
    msg: ‘退出成功’,
    data: []
  });
});
```

5.2.2 商品列表管理

（1）用户进入商城，首页为用户展示商城的商品。

（2）用户可以通过滑动鼠标来对商品列表的商品进行无限加载，避免用户进行翻页操作。

（3）用户通过单击商品页面各商品“加入购物车”字样的按钮，可对当前选中的商品进行加入购物车的操作。

（4）用户以游客的身份进行加入购物车操作，会给后台发送消息，执行一个检查登录的操作，并且通过检查返回消息体给用户界面，提示用户此时未进行登录，

请进行登录或者注册，表示加入购物车失败，反之，则用户加入购物车成功。

（5）用户将商品成功加入购物车，此操作前台会给后台服务器发送请求消息，服务器接收消息后，对数据库进行相应的操纵，对用户表下进行新增信息的操作，通过服务器返回相应的消息体给用户，并在浏览器中渲染出来。

5.2.3 购物车管理

（1）用户单击顶部“购物车”图标，进入购物车列表界面。

（2）用户进入购物车列表界面，单击“删除”图标，以从购物车列表中移除现有的不需要的商品，将其进行删除操作。

（3）用户单击商品数量旁边的“+”（“-”）可对当前选中的商品现有的总共数量进行新增（减少）操作，每点击一次，则新增（减少）数量 1。

（4）用户对购物车的增加、删除操作，前台给后台服务器发送请求消息体，后台服务器接收到请求信息后，对数据库进行相应的操作，并且返回相应的消息体给前端，最后通过渲染，呈现在浏览器。

（5）获取用户购物车列表的实现，关键代码如下：

前端发送请求：

```
methods: {
  // 调用商品数据
  getCart() {
    axios.get("/users/shoppingList").then(res => {
      console.log(res.data);
      if (res.data.code === 0&& res.status === 200) {
        if (res.data.result.cartList.length > 0) {
          this.cartList = res.data.result.cartList;
        } else {
          this.emptycart = true;
        }
      }
    });
  },
```

后端响应请求：

```
router.get('/shoppingList', (req, res) => {
  let usersid = req.cookies.usersid;
  Users.findOne({ _id: usersid }, { cartList: 1000, _id: 0 },
(err, doc) => {
    if (err) {
      res.json({
        code: 1,
        msg: "系统错误",
        result: []
      });
    } else {
      res.json({
        code: 0,
        msg: '获取购物车数据成功',
        result: doc
      });
    }
  })
});
```

5.2.4 地址管理

（1）用户单击购物车列表界面 “立即支付”，进入用户地址信息管理界面。

（2）通过用户的 id 来进行该用户地址信息的查询和获取。

（3）用户可以根据自己的需要，单击“设置默认地址”字样，用户即可以将所选地址设置为默认收货地址，也可以对现有地址信息进行新增或者删除操作。

（4）用户可以对地址进行获取、新增、删除以及设置默认地址等操作，并且会相应的给后台发送消息，后台服务器接收信息后，对数据库进行相应的查询、新增、删除操作，并返回对应的消息体给浏览器。

6 结束语

目前市面上此类网站繁多，因此，本网站致力于界面简洁，操作简单的需求进行设计与开发。

本文给出了基于 Vue 的在线商城设计与开发过程，其中 Vue 的响应式特性，使得数据的变化能够在界面中得到快速呈现；Vue 框架的双向绑定，使开发人员避免掉大部分的 DOM 操作，减少了开发人员的繁琐开发工作，从而大大提高开发效率；对于网站的使用者而言，本网站界面美观、操作高效，让用户拥有更好的体验。

【参考文献】

[1] 卢奇荣 . 基于 Vue2+Koa2+MongoDB 平台的网站技术分析 [J]. 广播电视信息 ,2020(02):103-105.

[2] 刘志洋 .ajax 技术在 web 程序开发中的运用探讨 [J]. 轻纺工业与技术 ,2020,49(02):169-170.

[3] 黄承明 . 基于 MongoDB 文档模型的教学资源数据的建模研究 [J]. 软件工程 ,2020,23(05):46-49.