Computer Science and Application 计算机科学与应用, 2025, 15(8), 182-191
Published Online August 2025 in Hans. https://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2025.158208

Hans汉斯

基于Vue的电商数据可视化平台的设计与实现

张耕硕*，张玉成，李明格，王佳艺

西京学院计算机学院，陕西 西安

收稿日期：2025年7月12日；录用日期：2025年8月12日；发布日期：2025年8月21日

摘　要

信息技术的快速进步正推动电商领域创新不断。在这个过程中，数据可视化技术尤为关键，它不仅极大地促进了商业模式的演变，还加强了平台与用户之间的互动。在本研究中，开发了一款基于Web技术的电商数据可视化系统，旨在实现数据的实时呈现，从而优化用户体验和商业决策。该系统采用了MVVM设计模式，核心技术选用了Vue框架，结合HTML5、CSS3和JavaScript技术。Vue框架的响应式设计和组件化特性使得前端开发更为简洁，同时保证了用户界面的动态性和可维护性。为了全面展示电商数据，引入了Echarts图表库，这一图表库提供丰富的可视化图标，能够直观展示销售数据等关键信息，增强了数据的解读性和交互性。在后端部分，系统利用Node.js的Koa2框架搭建服务，采用了洋葱模型来处理HTTP请求，确保了数据处理的安全。前后端分离的架构策略进一步提升了系统的可扩展性和维护性。通过这一系统的实施，不仅为电商卖家提供了一个简洁的数据管理和销售策略分析工具，也优化了买家的购物体验，通过数据可视化技术，将复杂的数据信息转化为简洁直观的视觉展示，极大地提升了信息的透明度和商业决策的便捷性。总之，本项目示范了信息技术如何在电子商务领域内发挥创新的作用，为现代电商平台提供了价值。

关键词

Web技术，Koa2框架，Vue框架，电商数据可视化

Design and Implementation of an E-Commerce Data Visualization Platform Based on Vue

Gengshuo Zhang*, Yucheng Zhang, Mingge Li, Jiayi Wang

School of Computer Science, Xijing University, Xi'an Shaanxi

Received: Jul. 12th, 2025; accepted: Aug. 12th, 2025; published: Aug. 21st, 2025

*通讯作者。

文章引用: 张耕硕, 张玉成, 李明格, 王佳艺. 基于 Vue 的电商数据可视化平台的设计与实现[J]. 计算机科学与应用, 2025, 15(8): 182-191. DOI: 10.12677/csa.2025.158208

Abstract

Rapid advancements in information technology are propelling innovation within the e-commerce sector. In this process, data visualization technology plays a particularly crucial role, not only greatly facilitating the evolution of business models but also enhancing the interaction between the platform and users. This study has developed an e-commerce data visualization system based on Web technology, aiming to achieve real-time data presentation for optimizing user experience and business decision-making. The system adopts the MVVM design pattern, with its core technology utilizing the Vue framework combined with HTML5, CSS3, and JavaScript technologies. The responsive design and componentization of the Vue framework enhance front-end development efficiency while ensuring a dynamic and maintainable user interface. To fully display e-commerce data, the Echarts chart library was introduced to provide rich visual representations for key information such as sales data, thereby enhancing data interpretation and interactivity. In terms of back-end development, Node.js Koa2 framework is utilized to build services while adopting an onion model for processing HTTP requests to ensure efficient and secure data processing. The architecture strategy of front-end separation further improves system scalability and maintainability. Through implementing this system, it not only provides an efficient tool for e-commerce sellers in managing data and analyzing sales strategies but also optimizes buyers' shopping experiences. Data visualization technology transforms complex information into simple and intuitive visual displays that greatly improve information transparency and facilitate business decisions. In summary, this project demonstrates how information technology can play an innovative role in e-commerce by providing value to modern e-commerce platforms.

Keywords

Web Technology, Koa2 Framework, Vue Framework, E-Commerce Data Visualization

Copyright © 2025 by author(s) and Hans Publishers Inc.
This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).
http://creativecommons.org/licenses/by/4.0/
Open Access

1. 引言

1.1. 背景介绍及意义

随着电商市场的不断扩大和竞争的加剧，电商平台产生了大量的数据，包括销售数据、用户行为数据、产品数据等。企业需要更加精细化地管理和分析海量数据，以便做出更加准确的业务决策。

而传统的数据报表往往呈现出来的是生硬的数据数字，难以直观地展示数据之间的关联和趋势。因此，利用数据可视化技术可以更加清晰地呈现数据，帮助企业决策者更好地理解数据、发现问题和机会。

基于 Vue 框架的电商数据可视化平台的设计与实现，具有提升数据可视化效率和质量的意义，Vue 框架具有响应式数据绑定和组件化开发的特性，能够快速构建交互式的数据可视化界面，并且具有良好的性能和稳定性，能够满足大规模数据可视化的需求，提升数据可视化的效率和质量；具有支持实时数据分析的意义，电商数据可视化平台可以实现对实时数据的动态监控和分析，帮助企业及时发现问题和机会，快速调整策略和方向。数据可视化平台可以帮助企业管理者直观地了解业务的运营状况、市场趋势和用户行为，从而基于数据进行科学化和精细化的业务决策，提升企业的竞争力和盈利能力。

1.2. 国内外研究现状

为了提高项目开发效率，国内对于前端框架的研究主要聚焦于实际应用场景，如电商、社交、媒体等行业的前端开发需求。同时，随着移动互联网和云计算的发展，前端框架在微服务、大数据可视化等方向也得到了广泛应用和研究。国外对前端框架的研究更加注重技术创新和框架设计原理。例如，新的数据绑定、状态管理、组件化、性能优化等方面的研究成果不断涌现[1]，推动前端技术的进一步发展。

1.2.1. 开发框架现状

随着前后端分离的趋势，前端框架的研究也越来越注重与后端技术的结合，以实现全栈开发。

随着移动端、桌面端和 Web 应用的融合，跨平台开发和性能优化成为前端框架研究的重要方向。用户体验和可访问性在前端框架的研究中也受到越来越多的关注，以提供更好的用户体验和满足不同用户群体的需求。随着技术的不断进步和应用场景的扩展，前端框架的研究和发展仍将是前端技术领域的重要方向。

1.2.2. 技术发展现状

前端框架技术在过去几年里经历了快速的发展，有三种框架易用性和应用率较高，这三种框架分别为 Angular 框架、React 框架、Vue 框架。

Angular 框架经历了从 AngularJS 到 Angular 的重大变革，现在的 Angular 框架更加现代化、模块化，支持 TypeScript，并且拥有强大的功能和工具集。尽管相对于 Vue.js 和 React，Angular 在新项目中的使用率有所下降，但在企业级应用和一些特定场景中仍然具有一定的市场份额[2]。

React 作为另一款流行的前端框架，在技术领域也保持着稳定的领先地位。其虚拟 DOM 和单向数据流的设计思想，以及庞大的社区支持[3]，使得 React 在大型项目和企业级应用中得到广泛应用。

Vue 框架作为一款轻量级、易学易用的前端框架，在过去几年中持续增长，成为了许多开发者的首选。其简洁的语法、响应式数据绑定以及丰富的生态系统受到了广泛的欢迎。

1.3. 论文研究内容

本项目采用了目前广泛使用的 Vue 框架进行设计与开发，通过利用 Vue 的组件化特性，可以显著提升代码的复用性。该框架使得电商数据可视化平台的开发更为简洁，能够轻松实现诸如商家分布图、地区销量趋势图等多个功能模块[4]。使用 Vue 框架的另一个优点是，它可以优化页面的渲染速度和数据更新过程，从而提升最终用户的体验。这种优化不仅提高了界面的响应性，还确保了数据显示的实时性，使平台用户能够获取到最新的商业洞察，以支持更好的决策制定。论文将通过电商可视化大屏案例，探讨电商可视化大屏的界面设计、数据可视化效果设计等方面内容。

2. 开发环境及相关技术介绍

2.1. 开发环境的选择和搭建

2.1.1. VS Code

本次项目开发选用的开发工具是 VS Code，VS Code 可以在 Windows、Mac 和 Linux 上运行，有向远程仓库提交代码的功能。启动速度快，占用系统资源少，适合于各种大小的项目，几乎支持所有流行的编程语言和开发框架[5]。可以通过安装相关的扩展来增强编辑器的功能，满足开发需求，VS Code 提供了强大的代码补全功能，能够根据上下文和已有的代码提示出可能的选项，大大提高了编码效率。

2.1.2. Node 环境配置

从官方网站下载合适的 node.js 的安装包，安装完成后，打开终端，输入相关命令查看 Node.js 和 npm

的版本，确保它们已正确安装，安装 Node.js 之后更新 npm 到最新版本。根据网络环境，可以配置 npm 镜像源来加速包的下载速度[6]。

2.1.3. Vue 项目搭建

安装 Vue 脚手架，进入想要创建项目的目录可以用 yarn 代替 npm 来解决建立项目速度慢的问题[7]。创建项目时，Vue CLI 会提示你选择项目的配置，可以选择手动配置或者使用预设配置，成功构建完项目后就可以正常启动项目。

3. 可视化平台的开发流程

项目采用 Vue3 + Koa2 + WebSocket 的极简架构：

前端用 Vue CLI 搭建单页应用，所有图表整合在 ScreenPage 内，通过 Vuex 统一管理主题与全屏状态；数据层前期用 Axios 轮询 RESTAPI，后期由 WebSocket 推送，保持实时更新[8]。

后端仅 Koa 服务：REST 接口负责初始数据，WebSocket 长连接把本地 JSON 文件内容实时推给前端。

静态资源全部放在/public/static，ECharts、主题、地图矢量数据和图片一次性加载。

整体就是“前端一屏、后端一服、数据一夹”的小而全方案。整体结构如图 1 所示。

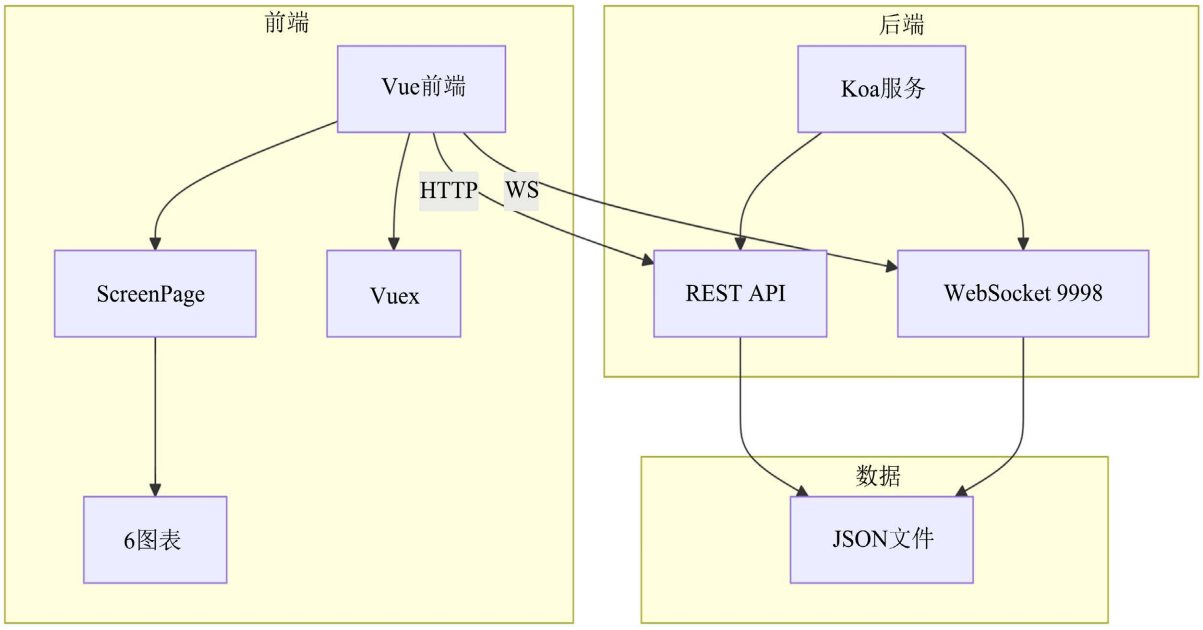

Figure 1. Overall architecture
图 1. 整体结构

3.1. 电商数据可视化平台后台开发流程

在 Koa2 项目中，只需完成三件事：先记录每次请求开始时间，待后续中间件执行完毕后将总耗时写入响应头；接着根据返回数据类型自动写入对应 MIME (如 JSON 输出 Content-Type: application/json)；最后把存放图表数据的指定目录映射为只读接口，直接读取文件内容返回，从而省掉数据库，实现轻量高效的后台，具体数据来源于电商平台的后台数据并稍作修改以验证一些极端数字，JSON 数据结构以热销产品数据为例如图 2 所示。

Figure 2. JSON data display
图 2. JSON 数据展示

3.1.1. 后台项目的实现

项目准备：

- 首先利用终端安装所需依赖 npm init -y npm install koa
- 建立文件并且构建相应的目录结构：

app.js 作为入口文件，data 目录存放模块相关的 JSON 文件数据，middleware 目录包含各种中间件代码，包括 koa_response_data.js 负责业务逻辑、koa_response_duration.js 计算服务器处理时长、以及 koa_response_header.js 设置响应头，具体文件结构如图 3 所示。

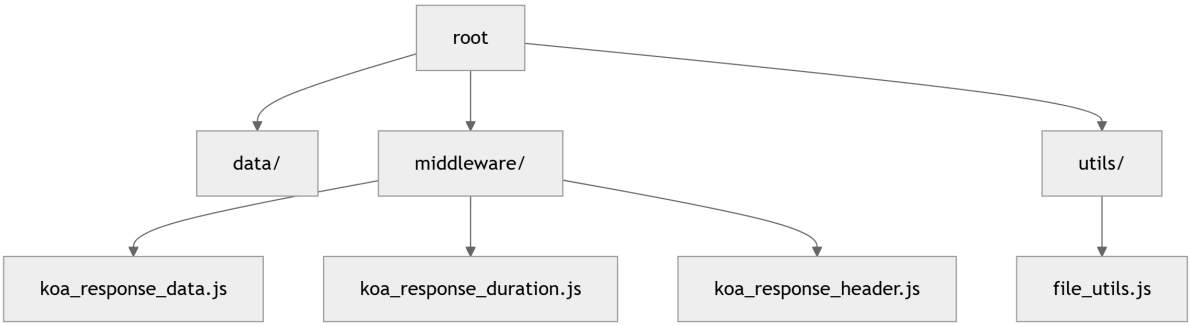

Figure 3. Backend file structure
图 3. 后台文件结构

3.1.2. 总耗时中间件

- 第 1 层中间件

总耗时中间件是一种常见的中间件，用于计算请求处理的总耗时，并将其记录下来，以便监控和性能优化[9]。还可以提供更详细的性能数据，比如每个请求处理阶段的耗时情况，以及可能的瓶颈或性能优化建议。

- 执行时间的计算

通过记录的结束时间减去记录的开始时间，计算出过程的以毫秒为单位的执行时间，总耗时中间体代码如图 4 所示。

```
// app.js
// 绑定第一层中间件
const respDurationMiddleware = require('./middleware/koa_response_duration')
app.use(responseIdonMiddleware)


// koa_response_duration.js
// 计算服务器消耗时长的中间件
module.exports = async (ctx, next) => {
    // 记录开始时间
    const start = Date.now()
    // 让内层中间件得到执行
    await next()
    // 记录结束的时间
    const end = Date.now()
    // 设置响应头 x-Response-Time
    const duration = end - start
    // ctx.set 设置响应头
    ctx.set('x-Response-Time', duration + 'ms')
}
```

Figure 4. Total time intermediate code
图 4. 总耗时中间体代码

3.1.3. 响应头中间件

➢ 获取 mime 类型

由于向前端浏览器返回的数据都是 JSON 格式的字符串，因此 MIME 类型可以统一设置为 application/json。

➢ 设置响应头

响应头的 key 是 Content-Type，其值为 application/json，响应头中间体代码如图 5 与图 6 所示。

```
// 绑定第二层中间件
const respHeaderMiddleware = require('./middleware/koa_response_header')
app.use(respHeaderMiddleware)
```

Figure 5. Response header intermediate code
图 5. 响应头中间体代码

```
// 设置响应头的中间件
module.exports = async (ctx, next) => {
  const contentType = 'application/json; charset=utf-8'
  ctx.set('Content-Type', contentType)
  await next()
}
```

Figure 6. Response header intermediate code
图 6. 响应头中间体代码

3.1.4. 业务逻辑中间件

它承载了服务端处理后的数据或信息，以供客户端进行解析和显示[10]。这通常包括了所请求资源的状态、内容、以及可能的其他相关信息。

➢ 读取相关文件内容

■ 首先，确定要读取的文件的路径，本地文件系统中的路径，取决于文件的存储位置。

■ 使用编程语言提供的文件操作函数，打开要读取的文件。

■ 根据需要，可以选择不同的方式读取文件内容

➢ 设置响应体

3.2. 电商数据可视化平台前端开发流程

前端项目的实现

- 全局环境中安装 vue-cli 脚手架
- 使用命令行执行创建项目 vision

1) 引入 Echarts 包

将名为 static 的文件夹从资料文件夹复制到 public 目录中，在 public/index.html 文件中添加 echarts.min.js 文件的引用，以便在网页中使用 echarts 进行数据可视化。代码如图 7 所示。

```
<body>
  <noscript>
    <strong>We're sorry but <%= htmlWebpackPlugin.options.title %> doesn't work properly without JavaScr
  </noscript>
  <div id="app"></div>
  <!-- built files will be auto injected -->
  <script src="static/lib/echarts.min.js"></script>
</body>
```

Figure 7. Importing the ECharts package
图 7. 引入 Echarts 包

2) 单独图表组件的开发(以商家销售量统计表为例)

在项目的初始阶段，首先将每种图表单独开发，再将它们合并到一个界面之中。在单独开发每个图表时，每个图表会使用一个独立的路径进行全屏展示，如图 8 所示为商家销售量统计的展示。

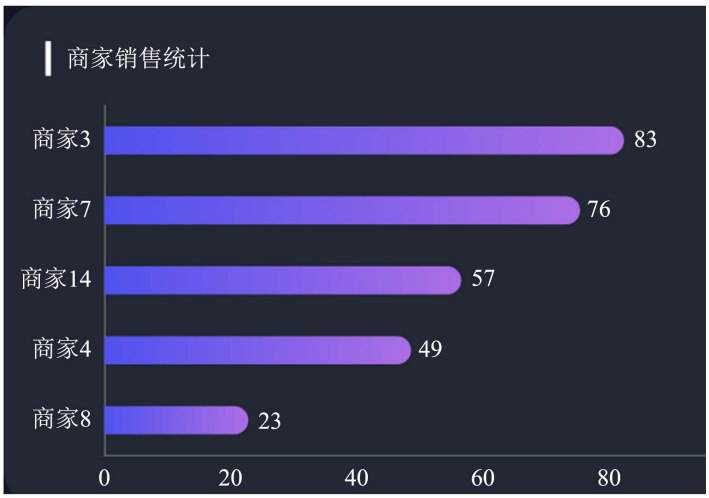

Figure 8. Expected sales volume statistics chart for merchants
图 8. 商家销售量统计预期图

- 组件结构设计

在 src/components/目录下创建 Seller.vue 组件，用于展示图表。为外层的 div 增加类样式为 com-container，随后建立一个新的显示图表的 div 标签元素，并且为新增的 div 增加类样式 com-chart，之后为外层 div 元素增加样式 com-page，在 SellerPage 中引入 Seller 组件，并注册和使用。

- 图表 Seller.vue 基本功能的实现

在 mounted 生命周期中初始化 echartsInstance 对象，确保图表库被正确加载。在 mounted 中获取服务器的数据，将获取到的数据设置到图表上，通过 echartsInstance 对象来渲染图表。

➢ 实现分页动画

首先进行数据处理，代码如图 9 所示。

```
data () {
  return {
    chartInstance: null,
    allData: null, // 服务器返回的数据
    currentPage: 1, // 当前显示的页数
    totalPage: 0, // 一共有多少页
  }
},
getData (ret) {
  // http://127.0.0.1:8888/api/seller
  const { data: ret } = await this.$http.get('seller')
  this.allData = ret
  // 对数据排序
  this.allData.sort((a, b) => {
    return a.value - b.value // 从小到大的排序
  })
  // 计算总页数
  this.totalPage = this.allData.length % 5 === 0 ? this.allData.length / 5 : this.allData.length / 5 + 1
  this.updateChart()
},
updateChart () {
  const start = (this.currentPage - 1) * 5
  const end = this.currentPage * 5
  const showData = this.allData.slice(start, end)
  const sellerNames = showData.map((item) => {
    return item.name
  })
  const sellerValues = showData.map((item) => {
    return item.value
  })
```

Figure 9. Data processing
图 9. 数据处理

鼠标事件的处理，代码如图 10 所示。

```
initChart () {
  this.chartInstance = this.$echarts.init(this.$refs.seller_ref) // 初始化echarts实例对象
  this.chartInstance.on('mouseover', () => {
    clearInterval(this.timerId)
  })
  this.chartInstance.on('mouseout', () => {
    this.startInterval()
  })
},
```

Figure 10. Mouse event handling
图 10. 鼠标事件处理

➢ 分辨率的适配

- 对窗口大小变化的事件进行监听
- 将字体大小的值设置给图表的某些区域

3) WebSocket 的引入

WebSocket 协议通过在客户端和服务器之间建立的单个 TCP 连接上提供全双工通信，允许服务器实时地向客户端推送数据，而无需等待客户端发起请求[11]，涉及到前端代码和后端代码的改造，前端需要进行 websocket 包的安装以及创建实例对象和监听事件等任务，后端需要约定好与客户端数据交换的格式和含义等任务，此处不再详细描述后端改造过程，而只描述部分前端代码改造过程。

➢ 定义注册函数

➢ 记录当得到数据时，应该调用的函数回调

➢ 连接服务端

- 在 main.js 中连接服务器端
- 将 SocketService 实例对象挂载到 Vue 的原型对象上

  Vue.prototype.$socket = SocketService.Instance

➢ 发送数据给服务端

- 在 socket_service.js 中定义发送数据的方法
- created 中注册回调函数
- destroyed 中取消注册
- 修改在 socket_service.js 中 send 方法具体代码如图 11 所示。

```
// 发送数据的方法
send (data) {
  // 判断此时此刻有没有连接成功
  if (this.connected) {
    this.sendRetryCount = 0
    this.ws.send(JSON.stringify(data))
  } else {
    this.sendRetryCount++
    setTimeout(() => {
      this.send(data)
    }, this.sendRetryCount * 500)
  }
}
```

Figure 11. Code processing
图 11. 代码处理

3.3. 本章小结

在本章节中，详细介绍了电商数据可视化平台的功能实现情况，包括主要功能模块和设计步骤。实现的功能基本符合需求分析的要求。通过对电商数据可视化平台的前端页面进行设计分析，发现其优点在于数据显示清晰、准确，并且图形展示效果良好。然而，仍有一些地方需要优化，比如页面略显单一，以及地图显示时容易出现不稳定的情况。

参考文献

[1] 项露芬, 孙佳怡, 李梦婷. 基于 Vue 和 Node.js 的音乐门票管理系统的设计与实现[J]. 现代信息科技, 2025, 9(11): 96-101.

[2] 吕志强. 基于 Vue 前端开发框架的管理系统设计[J]. 电脑知识与技术, 2024, 20(36): 56-59.

[3] 黄苗苗. 基于 Vue 和 Node.js 的作业管理系统的设计与实现[J]. 现代信息科技, 2024, 8(22): 102-105+110.

[4] 高迎. 基于 Vue 技术框架的短视频广告交易系统开发[J]. 科技与创新, 2024(12): 26-29+33.

[5] 朱嘉涵, 何治斌, 陆昊, 等. 基于 NodeJS+VueJS 的船舶机务管理系统的设计与实现[J]. 计算机应用与软件, 2024, 41(11): 39-45+77.

[6] 刘泉晶, 霍鹏举. JSON 和 Websocket 在显控软件前后端数据交互中的应用[J]. 电脑知识与技术, 2025, 21(8): 48-50.

[7] 宋文凯, 王利萍, 荆巍巍. 基于 Vue 框架的多源 WEB 前端页面数据可视化方法[J]. 电子设计工程, 2025, 33(6): 63-66+71.

[8] 张植才, 张福峰. 基于 Vue 前后端分离技术的海产品饲料信息追溯系统设计[J]. 办公自动化, 2025, 30(4): 19-21+124.

[9] 赵海国. Ajax 技术支持下的 ECharts 动态数据实时刷新技术的实现[J]. 电子技术, 2018, 47(3): 25-27+57.

[10] Song, J., Zhang, M. and Xie, H. (2019) Design and Implementation of a Vue.js-Based College Teaching System. *International Journal of Emerging Technologies in Learning* (*iJET*), **14**, Article No. 59. https://doi.org/10.3991/ijet.v14i13.10709

[11] Li, N. and Zhang, B. (2021) The Research on Single Page Application Front-End Development Based on Vue. *Journal of Physics*: *Conference Series*, **1883**, Article ID: 012030. https://doi.org/10.1088/1742-6596/1883/1/012030