**基于大数据平台的电商系统搭建**

**2020年5月**

# 目录

[1 项目概述 1](#_Toc39533059)

[2 建设目标及原则 1](#_Toc39533060)

[2.1 建设目标 1](#_Toc39533061)

[2.2 建设原则 1](#_Toc39533062)

[3 项目分工及安排 2](#_Toc39533063)

[4 需求分析 1](#_Toc39533064)

[4.1 概述 1](#_Toc39533065)

[4.2 功能需求 1](#_Toc39533066)

[4.2.1 GMV成交总额 1](#_Toc39533067)

[4.2.2 用户新鲜度 1](#_Toc39533068)

[4.2.3 品牌的复购率 1](#_Toc39533069)

[5 概要设计 1](#_Toc39533070)

[5.1 概述 1](#_Toc39533071)

[5.2 系统框架设计 1](#_Toc39533072)

[5.2.1 技术架构设计 1](#_Toc39533073)

[5.2.2 系统功能结构设计 2](#_Toc39533074)

[5.3 应用系统设计 3](#_Toc39533075)

[5.3.1 设计原则 3](#_Toc39533076)

[5.3.2 功能模块设计 4](#_Toc39533077)

[5.4 接口设计 4](#_Toc39533078)

[5.4.1 外部接口 4](#_Toc39533079)

[5.4.2 内部接口 4](#_Toc39533080)

[6 详细设计 4](#_Toc39533081)

[6.1 系统概述 4](#_Toc39533082)

[6.2 系统数据流程 5](#_Toc39533083)

[6.2.1 业务数据生成 5](#_Toc39533084)

[6.2.2 业务数据导入数仓 6](#_Toc39533085)

[6.3 功能设计 12](#_Toc39533086)

[6.3.1 GMV成交总额 12](#_Toc39533087)

[6.3.2 转化率之用户新鲜度及漏斗分析 12](#_Toc39533088)

[6.3.3 品牌复购率 14](#_Toc39533089)

[7 系统关键技术 18](#_Toc39533090)

[7.1 Vue 18](#_Toc39533091)

[7.2 Echarts 18](#_Toc39533092)

[7.3 ES6 18](#_Toc39533093)

1. **项目概述**

现如今，随着人们对互联网行业的深入研究，互联网技术在近几年得到了极大的普及，电商网站的产生更是让互联网逐步成为人们日常生活中必不可少的一部分，它便利了人们的生活，提高了人们的生活品质。

随着电商网站日益增长的用户体验，数据量增长势头也是只增不减，对这些海量数据进行加工处理使其成为有用的数据成为了一个巨大的挑战，但同时也带来了巨大的技术创新和商业机遇。正确的数据分析结果可以帮助网站能对自身得到更全新的认识，然后对网站的不足之处做出正确的判断，及时纠正，为网站创造更多的收益价值。

1. **建设目标及原则**
   1. **建设目标**

本系统是基于Hadoop框架，围绕Nginx服务器、MySQL数据库、Hive数据仓库、HighCharts图库表、JavaScript脚本语言、CSS样式表等众多前后端技术，实现电商网站用户日志数据的解析处理，最终生成一系列的数据指标，并对此进行可视化展示，为电商网站的改进提供一定的参考。

总之，该系统建设的总体目标是：

* 对电商日志数据进行处理；
* 分析和挖掘电商日志数据信息；
* 为电商网站带来巨大的商业价值；
* 提高电商网站的核心竞争力。
  1. **建设原则**

**开放性原则：**

运用开放的标准规范整合各类系统，为实现系统信息交换和共享提供标准接口。

**集约性原则：**

充分利用已有建设成果，辅以必要的补充完善，逐步提升系统智慧化水平。

**复用性原则：**

采用模块化和平台化研制方法构建应用服务体系，提高软件复用和重用能力，方便建设成果推广。

1. **项目分工及安排**

项目主要分为基础平台搭建、数据采集传输、数仓搭建以及数据可视化四个部分，具体的项目分工如下表：

表 1 项目分工表

|  |  |  |  |
| --- | --- | --- | --- |
| **任务划分** | | | |
| **序号** | **工作内容** | **负责人** | **备注** |
|  | Hadoop基础平台搭建 | 所有人 |  |
|  | 数据采集传输 | 王伟建 |  |
|  | 数仓搭建 | 薛晓玉、杨洁、刘国栋、周擂震、姚磊 |  |
|  | 数据可视化 | 宋希文、况金荣 |  |

1. **需求分析**
   1. **概述**

数据可视化就是以更加有辨识度的图表或图形的格式显示已有的数据。它使用户能够在视觉上看到分析，从而掌握当前模式下的问题并加以改进。

* 1. **功能需求**

前端可视化部分主要是提供一些可视化的丰富展示，结合实际的业务需求，对不同场景下的业务数据可能的可视化的多样性进行探索。

在用户体验和交互设计方面。使用户快速定位到自己需要的内容，我们的图形化设计，让用户快速从数字中找到需要的信息。

在浏览器兼容性，我们的展示效果要兼容市场上常用的浏览器，另外我们可视化效果可能要兼容屏幕窗口，例如大屏展示等。

1. **概要设计**
   1. **概述**

前端系统主要实现数据图标的展示功能。项目整体基于Vue，路由采用Vue-router，全局状态管理采用Vuex，项目采用Vue官方支持的Vue-cli脚手架搭建，Vue-cli基于Webpack，开箱即用，基本不需要修改配置。

业务数据的展示主要采用Echarts。Echarts简单易用可配置性强。能够实现各种复杂的逻辑。项目展示可能存在大屏展示的需求，需要对适配性做较高的要求。

* 1. **系统框架设计**
     1. **技术架构设计**

项目前端展示部分基于Vue.js架构。Vue.js是一种渐进式框架，不同于以往的操作DOM的交互模式，Vue.js提供了更加优雅的双向数据绑定模式。其渐进式的设计理念决定了框架底层只依赖更少量的代码，其他功能可基于现有生态做更多的扩展。例如，路由组件采用的是Vue-router，全局状态管理采用的是Vuex等。

Vue简介。Vue是一套用于构建用户界面的渐进式框架。与其它大型框架不同的是，Vue 被设计为可以自底向上逐层应用。Vue 的核心库只关注视图层，不仅易于上手，还便于与第三方库或既有项目整合。另一方面，当与[现代化的工具链](https://cn.vuejs.org/v2/guide/single-file-components.html)以及各种[支持类库](https://github.com/vuejs/awesome-vue" \l "libraries--plugins" \t "_blank)结合使用时，Vue 也完全能够为复杂的单页应用提供驱动。

VueRouter简介。VueRouter是Vue.js官方支持的路由器组件。它和Vue.js的核心深度集成，让构建单页面应用变得易如反掌。

Webpack简介。本质上，webpack 是一个现代 JavaScript 应用程序的静态模块打包器。当 webpack 处理应用程序时，它会递归地构建一个依赖关系图，其中包含应用程序需要的每个模块，然后将所有这些模块打包成一个或多个bundle。

* + 1. **系统功能结构设计**
  1. **应用系统设计**
     1. **设计原则**

可视化的设计原则是要做到人们期望的理想效果，这个效果总体可概括为：

信：数据可信，表达力强，真实全面；

达：表达清晰，有效性强，用户容易理解；

雅：形式与内容，和谐统一，不能误导用户；

简：用更简的图形，传递更多信息。（数据墨水比越大越简洁）；

善：用意要符合正确价值观；

美：具有艺术美感

* + 1. **功能模块设计**

前端功能模块主要是数据图表的展示。

从功能上划分，包含一个导航菜单、诸多的图表组件。导航菜单通常是一个树形结构，因为本项目不涉及权限问题，导航菜单直接在前端写死。由于本项目图表种类较多，因此将类似的图表做成组件，组件化开发也是Vue技术栈的一大亮点。

从项目结构上划分，包括展示层、服务层。展示层即页面结构，页面结构只需要关注和页面渲染有关的任务。也就是说，从后端调取接口、接口数据的处理、包装都应该在服务层完成。

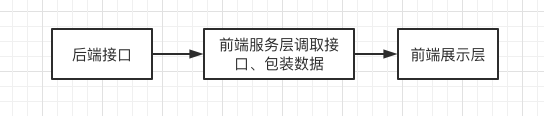
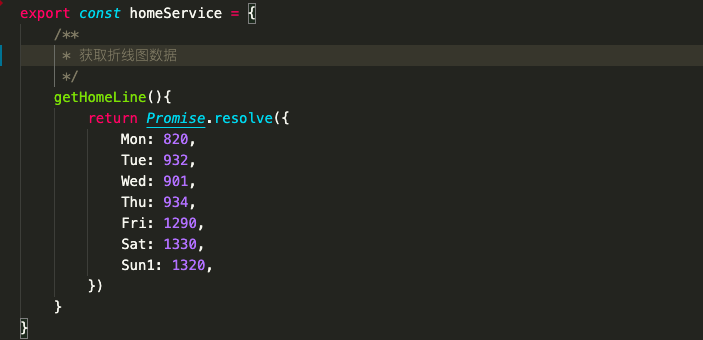


图5-1 项目结构

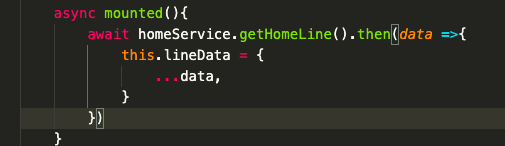
* 1. **接口设计**

前端主要调取java层的接口。前后分离的开发模式通常要求，前端和服务端通常约定好数据格式，然后各自开发。本项目将展示层和服务层分离，相当于在前端再次做了接口的封装，展示层只针对展示数据做交互，而服务层则做数据的请求和包装处理。这样做的好处是，有一些需要前端处理的复杂数据结构不再和页面耦合，页面只需要处理好展示层面的事即可，对于后续优化，甚至是增加单元测试测试逻辑层都非常有好处。

******

以上展示了服务层代码，采用Promise模拟后端数据请求的效果，将处理好的展示数据返回给展示层。Promise是ES6支持的原生异步操作接口，提供了更加方便的链式操作模式。

展示层代码调用如下：

******

在具体的页面中，引入homeService服务，然后通过promise链式操作取出数据，并赋值给页面具体操作。

1. **详细设计**
   1. **系统概述**
   2. **系统数据流程**
   3. **功能设计**
      1. **项目结构**

******

业务层项目结构如上图。

assets存储静态资源，如图片。

config记录项目中的一些自定义配置。

pages是具体的页面，这里采用的是单文件组件结构。数据展示组件也在这个目录下。

request是和接口请求路径相关。

service请求接口数据，包装数据提供给页面层。

store是全局状态管理。管理全局数据。

style封装了项目的通用样式。

utils封装了项目中经常使用的逻辑组件，例如时间日期转换等。

* + 1. **单文件组件**

Vue将模块的结构、逻辑、样式封装在一个组件之中，作为一个单文件组件。单文件组件可以限定某一块的逻辑，使得项目方便维护。下图展示了单文件组件的结构。

一个单文件组件包括template模板、script逻辑、style样式。

项目中的样式采用了scss框架，Scss支持嵌套的样式结构。scoped代表了样式仅在本组件中使用，不会污染外部样式。

template模板必须被一个div包裹，不然会报错。input标签中v-model就是双向绑定数据的例子。彻底摆脱了过去操作DOM的繁琐。只需要将定义好的数据绑定在模板层，就可以在数据改变时自动更新数据。

******

scripts是组件的逻辑部分。逻辑部分包括生命周期和组件状态管理。组件私有状态被存储在data中，为了组件可以复用，组件必须return一个对象。只有在组件中显示的定义的部分才能触发响应式更新。

props是父级组件向当前组件传入的数据。有的组件可能被做成函数式组件。所谓函数式组件，就是组件只接受外来参数，基本不保留自己的状态。props正是实现函数式组件的关键。props可以理解为函数的参数。

methods部分是组件的方法，可以在这里定义函数。

computed是计算属性。计算属性也具有函数式特性，计算属性的值返回一个函数，这个函数基于另一个值，当另一个值变化时，计算属性会随着变化。

watch是监听器。当我们需要监听一个值时需要用到。

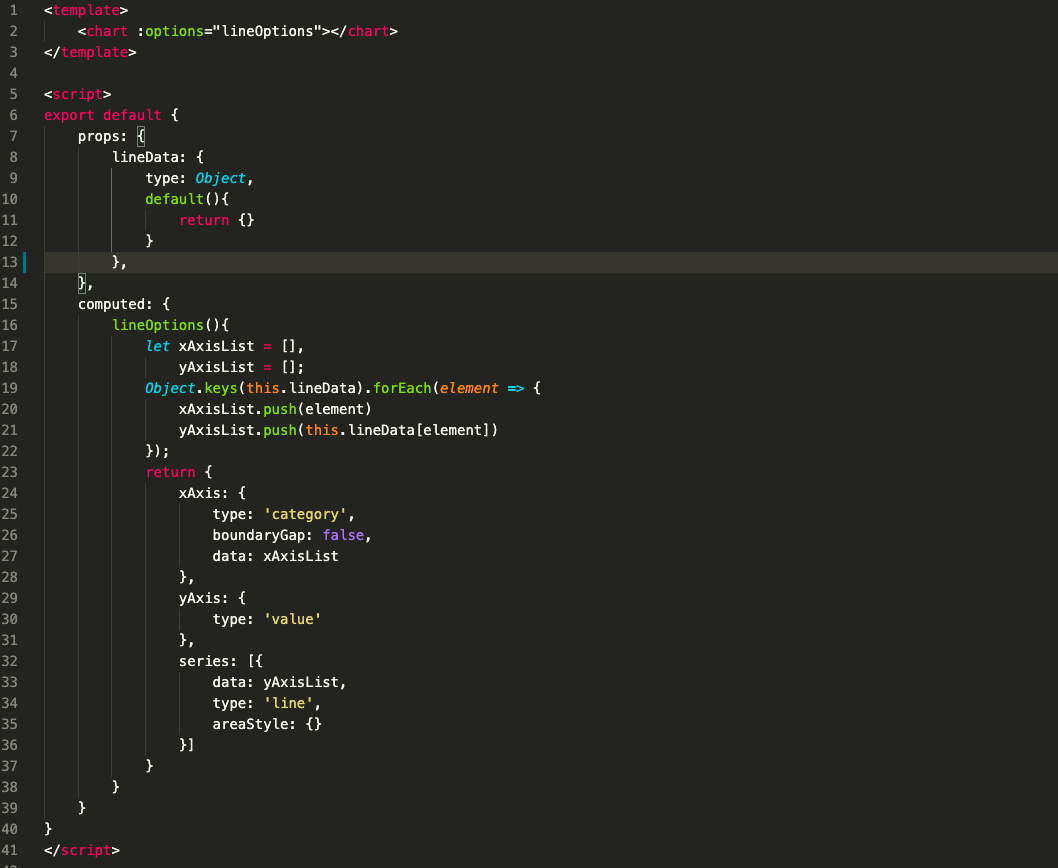
created和mounted都是生命周期钩子函数。通常在mounted阶段去做异步请求。

以上部分构成了单文件组件的基本构件。

* + 1. **组件化开发**

组件在服务端通常只和逻辑相关，但是在前端，组件的概念包含视图结构、样式和逻辑。组件化开发主要是为了代码的复用。本项目中，涉及的图表诸多，用以展现不同的逻辑。如果每个图表都重写一次逻辑，这将是灾难性的代码重复。因此，封装组件是非常有必要的。

由于图表组件通常只聚焦展示层，因此通常可封装为函数式组件。封装组件的第一步是定义props，即定义函数要接受哪些参数，以及参数的类型。然后根据参数的状态做封装和处理。组价封装应该符合持续集成的原则。对于更大型的组件甚至需要为组件编写测试用例。所为持续集成，是指组件的更新不得影响原有props的应用，除非是万不得已或者有更大版本的更新。

******

上图就是项目中封装的一个函数式组件。props传入了lineData，也就是经服务层包装后的数据。当前组件不存储自己的状态，也就是，没有data。只对传入的参数做更改。computed计算属性就是在对props传入的数据做更改。computed取出数据，然后返回封装好的，适合于echarts渲染的数据。

组件封装好后，通过es6模块化，引入组件。通过import语句引入组件，然后在components中注册组件，之后组件就可以使用了。在父级组件中，定义lineData数据。在父级组件mounted时，异步获取组件的数据，然后赋值给lineData。这样，一个函数式组件的渲染就成功了。

mounted钩子函数中的“...”是扩展运算符，这是es6语法的新特性，有兴趣的可以参考阮一峰老师的es6语法教程。

******

* + 1. **路由信息**

VueRouter是Vue官方推荐的路由管理工具。Vue构建的属于单页应用，不同于以往的多页应用。例如，localhost:8080/#/index.html这就是典型的单页应用。#后面的是页面的锚，也就是片段标识符。一旦标识符变化，即触发hashchange事件。而以往都得多页应用路径是localhost:8080/index.html。

******

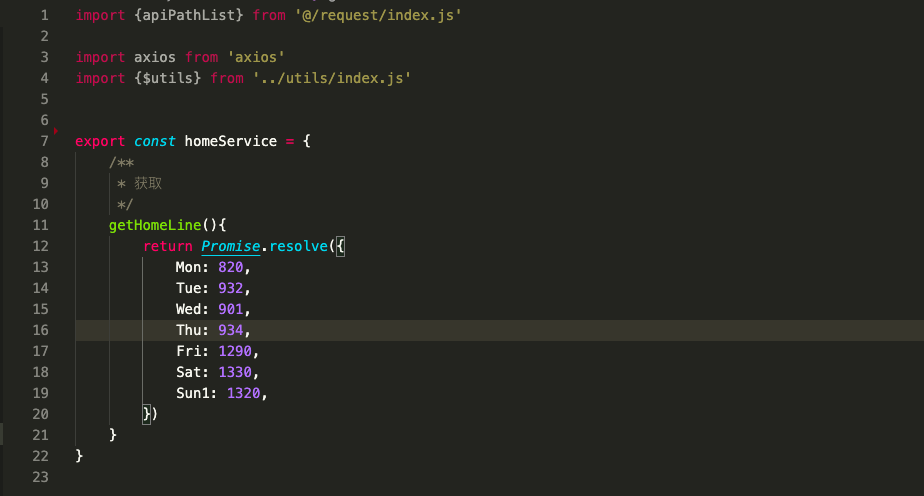
对于多页应用，通常认为更适合于页面的SEO优化，Vue也针对此做了设计。Vue的服务端渲染(SSR)，可以保证SEO优化。单页应用，开发起来更加方便，其逻辑性更适合于大型的B端应用。现在越来越多的逻辑密集型管理系统平台采用单页应用的模式。

******

VueRouter引入后，直接new一个Router实例即可。然后将路由以数组的形式引入。每个组件模块维护自己的路由数组，引入组件，即可加载。

* + 1. **前端服务层**

通常，为了方便开发，将数据请求和数据处理也会分开。有的项目会在逻辑层采用Typescript做强类型检查，减少逻辑出错的可能性。这里，只将接口请求的地址单独列出,写在resuest目录下。然后在服务层引入apiPathList。数据的请求采用axios组件。这里只用Promise模拟，后续将更新至真正的接口。

******

1. **系统关键技术**
   1. **Vue**

Vue是一套构建用户界面的渐进式框架，只关注视图层，采用自底向上增量开发的设计， 它的目标是通过尽可能简单的API实现响应的数据绑定和组合的视图组件。

Vue框架采用MVVM模式，能够将数据的变化显示并反映出来，这种效果是建立在对数据的双向绑定上，最终将其变化映射到虚拟DOM上。

Vue主框架主要具有以下特点：

(1)编码简洁，体积小，运行效率高。

(2)只关注视图层，易于与第三方库整合。

* 1. **Echarts**

Echarts是一个免费开源的可视化图表组件，是一个基于javaScript的图表库。Echarts底层依赖轻量级的Canvas类库ZRender, 提供直观、生动、可交互、可高度个性化定制的可视化图表。提供了常规的折线图、柱状图、散点图、饼图、K线图, 用于地理数据可视化的地图、热力图、线图, 用于关系数据可视化的关系图、treemap, 还有用于BI的漏斗图。

* 1. **ES6**

ES6 (ECMAScript 6)是一种新的Java Script语言标准, 为适应更复杂的应用，其新增了许多语法特性，并简化实现。