小米商城项目总结

**项目介绍**：

该项目主要是以PC端为主，通过Vue脚手架搭建小米商城，高仿小米商城的主要页面及其功能实现。

**主要功能：**

前台：登录、注册、首页、购物车、商品详情、商品列表、跳转支付、支付列表

后台：

项目运用技术：

vue脚手架、vue插件依赖、element-ui、axios实现页面交互及后台数据获取...

**框架搭建：**

1.跨域处理（vue.config.js）



1. 将各类页面在router目录中的index.js中路由引入，按主页面及子页面正确按序导入......
2. 在main.js中正确引入项目所需的依赖包，提前下载好（一定注意好版本）

**前端**

1. **首页**（index.vue/NavHeader.vue/NavFooter.vue）

首页中的头部和尾部在后续页面中是通用的，所以要单个在components中写成组件，方便后续页面引入组件即可。

Index.vue中轮播图实现：

引入依赖插件



Html文本部分



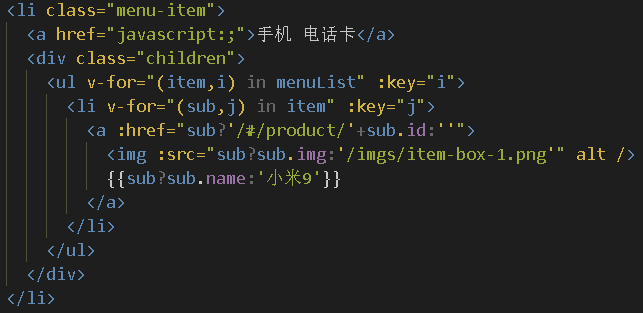
Js部分(下述代码要写在 data(){return{...}} 中)

|  |
| --- |
| swiperOption: {          autoplay: true,          loop: true,          effect: "cube",          cubeEffect: {            shadowOffset: 100,            shadowScale: 0.6,          },          pagination: {            el: ".swiper-pagination",            clickable: true,          },          navigation: {            nextEl: ".swiper-button-next",            prevEl: ".swiper-button-prev",          },        },        slideList: [          {            id: "42",            img: "/imgs/slider/slide-1.jpg",          },          {            id: "45",            img: "/imgs/slider/slide-2.jpg",          },          {            id: "46",            img: "/imgs/slider/slide-3.jpg",          },          {            id: "",            img: "/imgs/slider/slide-4.jpg",          },          {            id: "",            img: "/imgs/slider/slide-1.jpg",          },        ] |

Index.vue中侧边导航商品列表实现

Html部分

通过vue 中 v-for命令循环数据，然后渲染到页面中



Js中在data(){return{...}}中写入数组，让v-for遍历后并渲染到页面

|  |
| --- |
| menuList: [          [            {              id: 30,              img: "/imgs/item-box-1.png",              name: "小米CC9",            },            {              id: 31,              img: "/imgs/item-box-2.png",              name: "小米8青春版",            },            {              id: 32,              img: "/imgs/item-box-3.jpg",              name: "Redmi K20 Pro",            },            {              id: 33,              img: "/imgs/item-box-4.jpg",              name: "移动4G专区",            },          ],          [0, 0, 0, 0],          [0, 0, 0, 0],          [0, 0, 0, 0],          [0, 0, 0, 0],          [0, 0, 0, 0],        ], |

1. **登录注册页面**（login.vue）

首先获得输入框中用户输入的用户名和密码，然后传到后台验证是否有注册，有就跳转到主页面，没有就注册后在登录。

Html部分



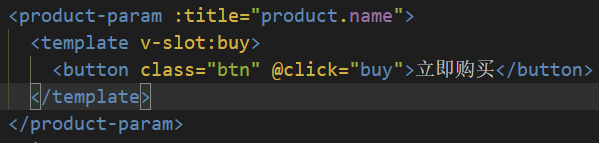
Js部分（在methods:{…}中写入html中标签调用的方法，当点击时执行此方法）

|  |
| --- |
| <script>  import {mapActions} from 'vuex'  export default {    name: 'login',    data() {      return {        username: '',        password: '',        userTd: ''      }    },    methods: {      //登录      login() {        let {username,password} = this;        this.axios.post('/user/login', {          username,          password        }).then((res) => {          this.$cookie.set('userId', res.id, {expires: 'Session'});          // this.$store.dispatch('saveUserName', res.username);          this.saveUserName(res.username);          this.$router.push({            name: 'index',            params: {              from: 'login'            }          })        })      },      ...mapActions(['saveUserName']),      //注册      register() {        this.axios.post('/user/register', {          username: 'kiven',          password: '8888',          email: 'kiven@163.com'        }).then(() => {          this.$message.success('注册成功');        })      }    }  }  </script> |

3.**产品页面**（product.vue）

（制作导航栏下拉滑动吸顶效果ProductParam.vue 因为此导航栏在后续子页面会用到，所以单个在components中写成组件来供调用）

Html部分（其中通过插槽方式使得根据不同的页面，动态更换副文本内容。例： product.vue中：



ProductParam.vue中：



ProductParam.vue中Js部分

|  |
| --- |
| <script>    export default{      name:'nav-bar',      props:{        title:String      },      data(){        return {          isFixed:false        }      },      //挂载      mounted(){        window.addEventListener('scroll',this.initHeight)      },      methods:{        initHeight(){          let scrollTop = window.pageYOffset || document.documentElement.scrollTop || document.body.scrollTop;          this.isFixed = scrollTop > 152? true:false;        }      },      //销毁      destroyed(){        window.removeEventListener('scroll',this.initHeight,false)      }    }  </script> |

给window添加scroll监听事件，监听滚动后调用initHeight()方法，当滑动距离大于152px时，使得<div :class=”{‘is\_fixed’:’isFixed’}”></div>中的is\_fixed的css属性激活，从而添加样式。

|  |
| --- |
| .is\_fixed{        position:fixed;        top:0;        width:100%;        box-shadow: 0 5px 5px #999;      } |

视频框弹出加载及关闭效果实现

Html部分

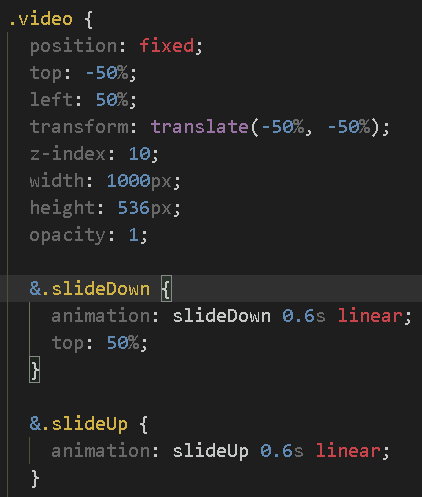


Js中写入html中@click绑定的事件方法供调用

在data(){return{...}}中定义好初始状态

|  |
| --- |
| data() {      return {        showSlide: "", //控制动画效果        product: {}, //商品信息      }    } |
| mounted() {      this.getProductInfo();    },    methods: {      getProductInfo() {        let id = this.$route.params.id;        this.axios.get(`/products/${id}`).then((res) => {          this.product = res;        });      },      buy() {        let id = this.$route.params.id;        this.$router.push(`/detail/${id}`);      },      closeVideo() {        this.showSlide = "slideUp";        setTimeout(() => {          this.showSlide = "";        }, 600);      },    },  }; |

其中html部分的视频打开关闭按钮实现功能通过给div盒子添加slideUp和slideDown的css属性来实现动画效果。



|  |
| --- |
| @keyframes slideDown {            from {              top: -50%;              opacity: 0;            }            to {              top: 50%;              opacity: 1;            }          }          @keyframes slideUp {            from {              top: 50%;              opacity: 1;            }            to {              top: -50%;              opacity: 0;            }          } |

4.**商品详情页面**（detail.vue）

html中选择商品版本点击切换不同版本型号

|  |
| --- |
| data() {      return {        id: this.$route.params.id, //获取商品ID        version: 1, //商品版本切换        product: {}, //商品信息      }    } |



在data(){}中定义version=1初始默认选中型号1商品，点击改变version的值来激活checked属性从而选中商品。

Js部分（商品页面跳转会传过来商品的Id）,拿到商品Id去请求后台商品数据，然后保存到product:[]数组中去，html页面调用数组中的数据渲染到页面上。

添加购物车：通过@click=”addCart”绑定事件来调用addCart()方法，方法中运用vuex将商品添加到购物车中，具体看（store文件目录下的文件）



5.**商品购物车页面**（cart.vue）

实现点击加号减号来改变商品数量，从而改变商品价格以及选中所有商品的总价格。



通过@click=”updateCart(item,’-’/’+’)”绑定事件来调用updateCart()方法，并且在JS的方法中判断点击的按钮类型（加号/减号），以此对商品数量进行增加和减少，带动价格及总价格的增加和减少。

|  |
| --- |
| data() {      return {        list: [], //商品列表        allChecked: false, //是否全选        cartTotalPrice: 0, //商品总金额        checkedNum: 0 //选中商品数量      }    }  //更新购物车  updateCart(item,type) {        let quantity = item.quantity,            selected = item.productSelected;        if(type == '-'){          if(quantity == 1) {            this.$message.warning('商品至少保留一件');            return;          }          --quantity;        }else if(type == '+') {          if(quantity > item.productStock) {            this.$message.warning('购买数量不能超过库存数量');            return;          }          ++quantity;        }else {          selected = !item.productSelected;        }        this.axios.put(`/carts/${item.productId}`,{          quantity,          selected        }).then((res)=>{          this.renderData(res);        });      }, |
| //公共赋值      renderData(res) {        this.list = res.cartProductVoList || [];        this.allChecked = res.selectedAll;        this.cartTotalPrice = res.cartTotalPrice;        this.checkedNum = this.list.filter(item => item.productSelected).length;      }, |

选择好要购买的商品后，确定商品数量及价格，点击结算按钮跳转支付页面。



Js中跳转支付接口实现

|  |
| --- |
| //购物车下单      order() {        let isCheck = this.list.every(item=>!item.productSelected);        if(isCheck){          this.$message.warning('请选择一件商品');        }else {          this.$router.push('/order/confirm');        }      } |

6.**支付页面**（order.vue）

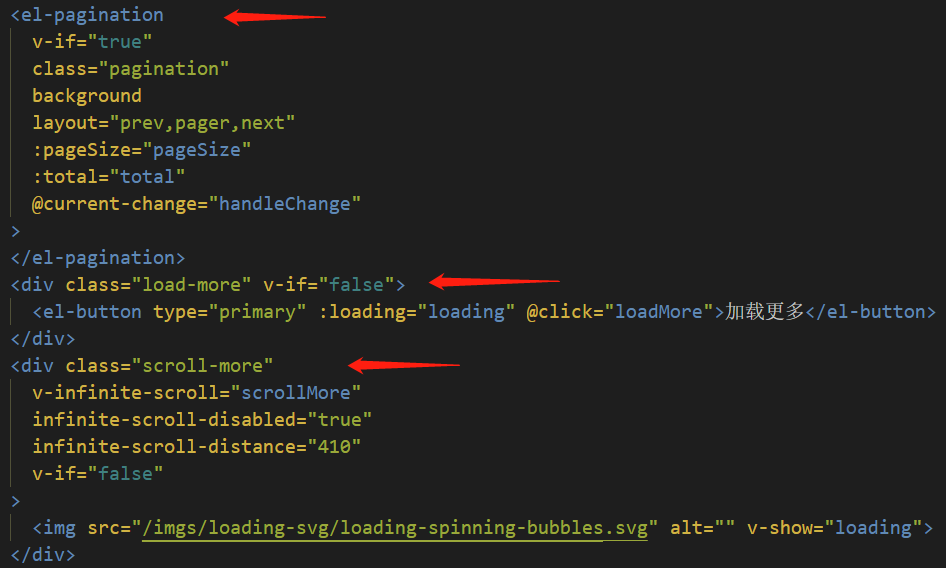
此页面为支付主页面，其他子页面都是基于主页面框架搭建。

7.**支付列表**（orderList.vue）

三种分页器效果讲解

Html页面部分，三种模块方式显示的效果不同，可根据要求选择不同的分页器方式。首先要引入所需的插件依赖：

|  |
| --- |
| import {Pagination,Button} from 'element-ui';  import infiniteScroll from 'vue-infinite-scroll'; |



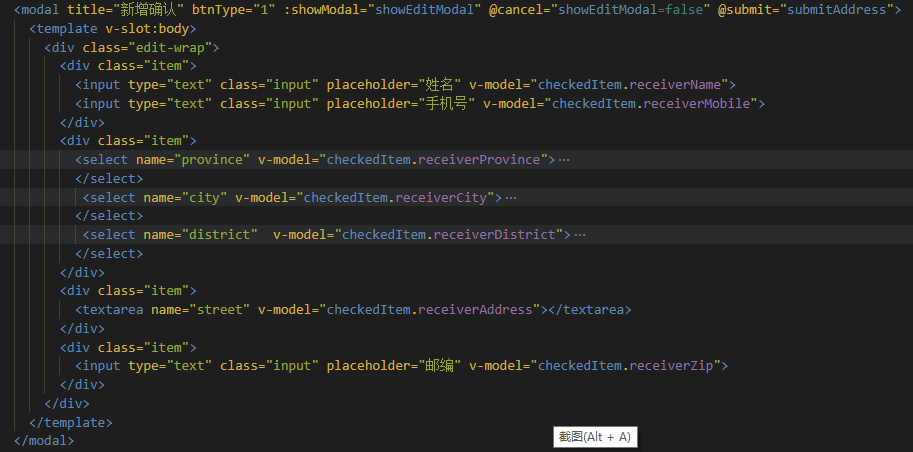
Js交互部分

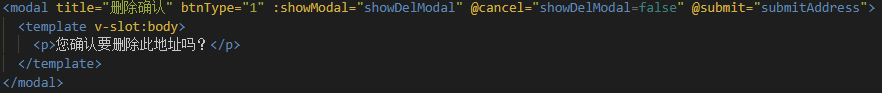
|  |
| --- |
| export default{      name:'order-list',      components:{        OrderHeader,        Loading,        NoData,        [Pagination.name]:Pagination,        [Button.name]:Button      },      directives: {        infiniteScroll      },      data(){        return {          loading: false,          list: [],          pageSize: 10,          pageNum: 1,          total: 0,          showNextPage: true, //加载更多：是否显示按钮          busy: false, //滚动加载，是否触发        }      },      mounted(){        this.getOrderList();      },      methods:{        getOrderList() {          this.loading = true;          this.busy = true;          this.axios.get('/orders',{            params: {              pageSize: 10,              pageNum:this.pageNum            }          }).then((res)=>{            this.loading = false;            this.list = this.list.concat(res.list);            this.total = res.total;            this.showNextPage = res.showNextPage;            this.busy = false;          }).catch(()=>{            this.loading = false;          })        },        goPay(orderNo) {          // 三种路由传参方式          // this.$router.push('/order/pay');          // this.$router.push({          //   name: '/order/pay',          //   query: {          //     orderNo          //   }          // })          this.$router.push({            path: '/order/pay',            query: {             orderNo            }          })        },        //第一种方法：分页器        handleChange(pageNum) {          this.pageNum = pageNum;          this.getOrderList();        },        //第二种方法：加载更多按钮        loadMore() {          this.pageNum++;          this.getOrderList();        },        //第三种方法：滚动加载，通过npm插件实现        scrollMore() {          this.busy = true;          setTimeout(()=>{            this.pageNum++;            this.getList();          },500)        },        //专门给scrollMore使用        getList() {          this.loading = true;          this.axios.get('/orders',{            params: {              pageSize: 10,              pageNum:this.pageNum            }          }).then((res)=>{            this.list = this.list.concat(res.list);            this.loading = false;            if(res.hasNextPage) {              this.busy = false;            }else {              this.busy = true;            }          });        },      }    } |

8.**支付确认页面**（orderConfirm.vue）

Html部分



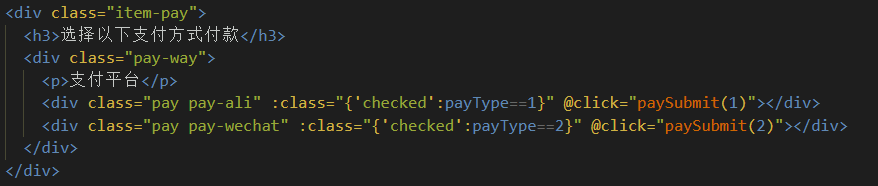




Js交互部分

|  |
| --- |
| export default{    name:'order-confirm',    data(){      return {       list: [], //收货地址列表       cartList: [], //购物车中需要结算的商品列表       cartTotalPrice: 0, //商品总金额       count: 0, //商品结算数量       checkedItem: {}, //选中的商品对象       userAction: '', //用户行为 0：新增 1：编辑 2：删除       showDelModal: false, //是否显示删除弹框       showEditModal: false, //是否显示新增或者编辑弹框       checkIndex: 0 //当前收货地址选中的索引      }    },    components:{      OrderHeader,      Modal    },    mounted(){      this.getAddressList();      this.getCartList();    },    methods:{      getAddressList(){        this.axios.get('/shippings').then((res)=>{          this.list = res.list;        })      },      //打开新增地址弹框      openAddressModal() {        this.checkedItem = {};        this.userAction = 0;        this.showEditModal = true;      },      //编辑新增地址弹框      editAddressModal(item) {        this.checkedItem = item;        this.userAction = 1;        this.showEditModal = true;      },      //删除地址      delAddress(item) {        this.checkedItem = item;        this.userAction = 2;        this.showDelModal = true;      },      //地址新增、编辑、删除功能      submitAddress() {        let {checkedItem,userAction} = this;        let method,url,params={};        if(userAction == 0) {          method = 'post',url = '/shippings';        }else if(userAction == 1) {          method = 'put',url = `/shippings/${checkedItem.id}`;        }else {          method = 'delete',url = `/shippings/${checkedItem.id}`;        }        if(userAction == 0 || userAction == 1) {          let {receiverName,receiverMobile,receiverProvince,receiverCity,receiverDistrict,receiverAddress,receiverZip} = checkedItem;          let errMsg='';          if(!receiverName){            errMsg = '请输入收货人名称';          }else if(!receiverMobile || !/^1[3-9]\d{9}$/.test(receiverMobile)) {            errMsg = '请输入正确格式的手机号';          }else if(!receiverProvince) {            errMsg = '请选择省份';          }else if(!receiverCity) {            errMsg = '请选择对应的城市';          }else if(!receiverDistrict || !receiverAddress) {            errMsg = '请输入收货地址';          }else if(!receiverZip || !/\d{6}/.test(receiverZip)) {            errMsg = '请输入六位邮编';          }          if(errMsg) {            this.$message.error(errMsg);            return;          }          params = {            receiverName,            receiverMobile,            receiverProvince,            receiverCity,            receiverDistrict,            receiverAddress,            receiverZip          }        }        this.axios[method](url,params).then(()=>{          this.closeModal();          this.getAddressList();          this.$message.success('操作成功');        });      },      closeModal() {        this.checkedItem = {};        this.userAction = '';        this.showDelModal = false;        this.showEditModal = false;      },      getCartList() {         this.axios.get('/carts').then((res)=>{          let list = res.cartProductVoList; //获取购物车中所有商品数据          this.cartTotalPrice = res.cartTotalPrice; //商品总金额          this.cartList = list.filter(item=>item.productSelected);          this.cartList.map((item)=>{            this.count += item.quantity;          })        })      },      //订单提交      orderSubmit() {        let item = this.list[this.checkIndex];        if(!item) {          this.$message.error('请选择一个收货地址');          return;        }        this.axios.post('/orders',{          shippingId:item.id        }).then((res)=>{          this.$router.push({            path: '/order/pay',            query: {              orderNo:res.orderNo            }          })        })      }    }  } |

9.**支付订单**（orderpay.vue）

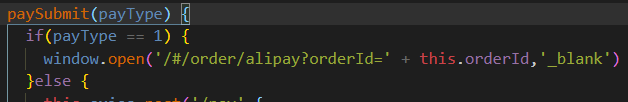


主要实现两种支付方式（支付宝支付payType==1，微信支付payType==2）

在JS部分中首先要定义好各类属性初始值

|  |
| --- |
| data() {      return {        orderId: this.$route.query.orderNo,        addressInfo: "", //收货人地址        orderDetail: [], //订单详情，包含商品列表        showDetail: false, //是否显示订单详情        payType: '', //支付类型        showPay: false, //是否显示微信支付弹框        payImg: '', //微信支付的二维码地址        showPayModal: false, //是否显示二次支付确认弹框        payment: 0, //订单总金额        T: '' //定时器ID      }    } |

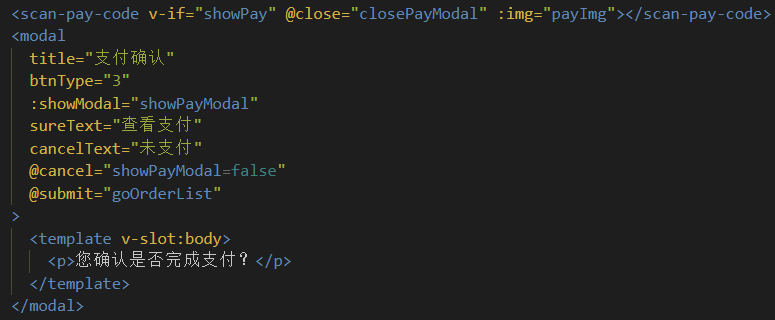
方式一：支付宝支付



Alipay.vue页面

|  |
| --- |
| methods: {        paySubmit() {          this.axios.post('/pay',{            orderId:this.orderId,            orderName: 'Vue高仿小米商城', //扫码支付时订单名称            amount: 0.01, //单位元            payType: 1 //1支付宝 2.微信          }).then((res)=>{            this.content = res.content;            setTimeout(()=>{              document.forms[0].submit();            },100)          })        }      } |

方式二：微信支付



Js交互部分

|  |
| --- |
| paySubmit(payType) {        if(payType == 1) {          window.open('/#/order/alipay?orderId=' + this.orderId,'\_blank')        }else {          this.axios.post('/pay',{            orderId:this.orderId,            orderName: 'Vue高仿小米商城', //扫码支付时订单名称            amount: 0.01, //单位元            payType: 2 //1支付宝 2.微信          }).then((res)=>{            QRCode.toDataURL(res.content)            .then(url=>{              this.showPay = true;              this.payImg = url;              this.loopOrderState();            })            .catch(()=>{              this.$message.error('微信二维码生成失败，请稍后重试');            })          })        }      },      //关闭微信弹框      closePayModal() {        this.showPay = false;        this.showPayModal = true;        clearInterval(this.T);      } |