农产品销售信息平台

微信小程序开发

目录

小程序说明

需求背景

目前的农民收入中，经营性收入占30%多一点，务工收入占45%左右，家庭经营收入主要靠农产品销售，通过电商销售农产品成为农民增收的一个亮点。在实际电商销售过程中，对于农产品的规模、物流、品控都有较高的标准和要求，这对于一般农户、或者刚起步的创业者来说很困难。希望有一些结合当地旅游、美食相关的小程序，让游客查询到当地土特农产品，去到原产地边游玩边购买；也可以填写在当地临时住址，方便农户们送货上门。

应用场景

- 各地农产品销售

解决的实际问题

- 为农民提供统一的农产品销售平台
- 为客户提供统一的农产品浏览平台

产品设计

整体架构分为To C和To B两个部分，农民可以通过上传商品、管理商品等来实现农产品销售；客户可以通过浏览商品，下订单等功能来实现农产品的购买。

技术方法

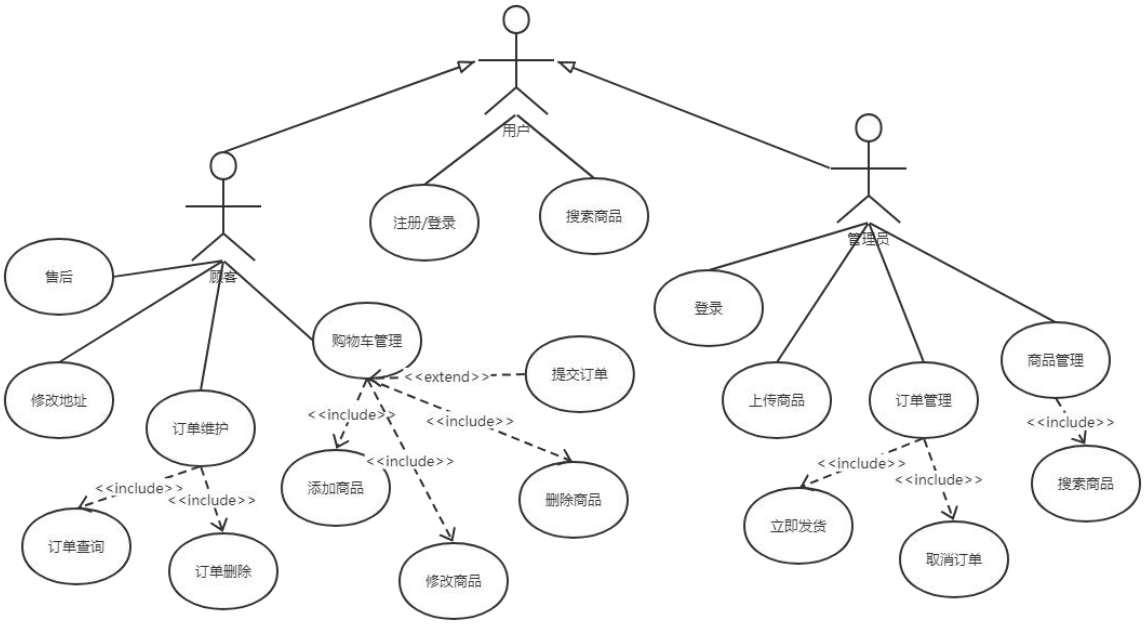
- 前端：js+css+html
- 后端：微信云开发
- 具体可在产品实现说明中查看

主要功能

- [x] 商品主页，商品列表可以浏览商品，点击列表中的商品可跳转至商品详情页，点击导航栏可以跳转购物车、我的等页面，搜索栏
- [x] 商品详情页展示商品图片、价格，可以选择规格、数量并添加至购物车，可以点击客服、购物车
- [x] 购物车功能，可以修改商品数量，删除商品，可以全选或者取消全选，每次更改都可以计算商品价格，可以提交订单，进入下单页面
- [x] 下单功能，在购物车提交订单跳转至下单页面，可以添加地址，查看购买商品列表，计算总价，添加备注，最后下单
- [x] 查看订单功能，可以分类别查看已付款、运输中、已完成订单
- [x] 登录功能，支持用户一键注册登录
- [x] 个人主页，可以查看各类别订单，修改地址等，可以登录后台
- [x] 简单管理端实现，可上传商品，搜索不同类型的商品，实现商品管理
- [x] 实现后台订单管理功能，可以查看所有类型订单，选择发货或者取消订单
- [x] 简单售后功能，用户可以选择是否售后，并联系客服
- [x] 简单搜索功能，在商品主页和后台管理均可实现商品的搜索

- [x] 支持商品分类查看

用例图



用例描述

用例名称：用户注册/登录
参与者：用户
简要说明： 用户一键注册登录
前置条件： 用户同意注册登录
基本事件流： 1．用户进入我的页面 2．弹窗提示是否允许注册登录 3．用户注册登录成功 4.用例终止
后置条件： 获得用户微信信息
注释：无

用例名称：添加商品
参与者：顾客
简要说明： 顾客选择想要的商品添加至购物车
前置条件： 顾客已注册登录
基本事件流： 1．顾客选择商品规格，数量 2．顾客点击加入购物车 3．顾客加入成功 4.用例终止
后置条件： 商品添加至购物车页面
注释：无

用例名称：删除商品
参与者：顾客
简要说明： 顾客删除购物车的商品
前置条件： 顾客已注册登录
基本事件流： 1．顾客长按商品 2．出现信息框，询问是否删除 3．点击删除，则商品从购物车中删除 4.用例终止
后置条件： 商品从购物车中删除
注释：无

用例名称：提交订单
参与者：顾客
简要说明： 顾客提交购物车或者直接购买的商品订单
前置条件： 顾客已注册登录
基本事件流： 1．顾客点击提交或者购买按钮 2．跳转至下单页面 3．顾客添加地址备注等 4.点击提交 4.用例终止
后置条件： 订单添加至订单页面
注释：无

用例名称：管理员注册/登录
参与者：管理员
简要说明： 管理员通过账号密码登录
前置条件： 进行用户登录
基本事件流： 1．管理员点击后台登录 2．弹窗提示输入账号密码 3．管理员登录成功 4.用例终止
后置条件： 无
注释：无

用例名称：管理员上传商品
参与者：管理员
简要说明： 管理员点击上传商品进行商品上传
前置条件： 进行管理员登录
基本事件流： 1．管理员点击上传商品 2.出现上传商品页面 3．管理员输入商品信息，点击提交 4.用例终止
后置条件： 商品添加至商品首页的商品列表

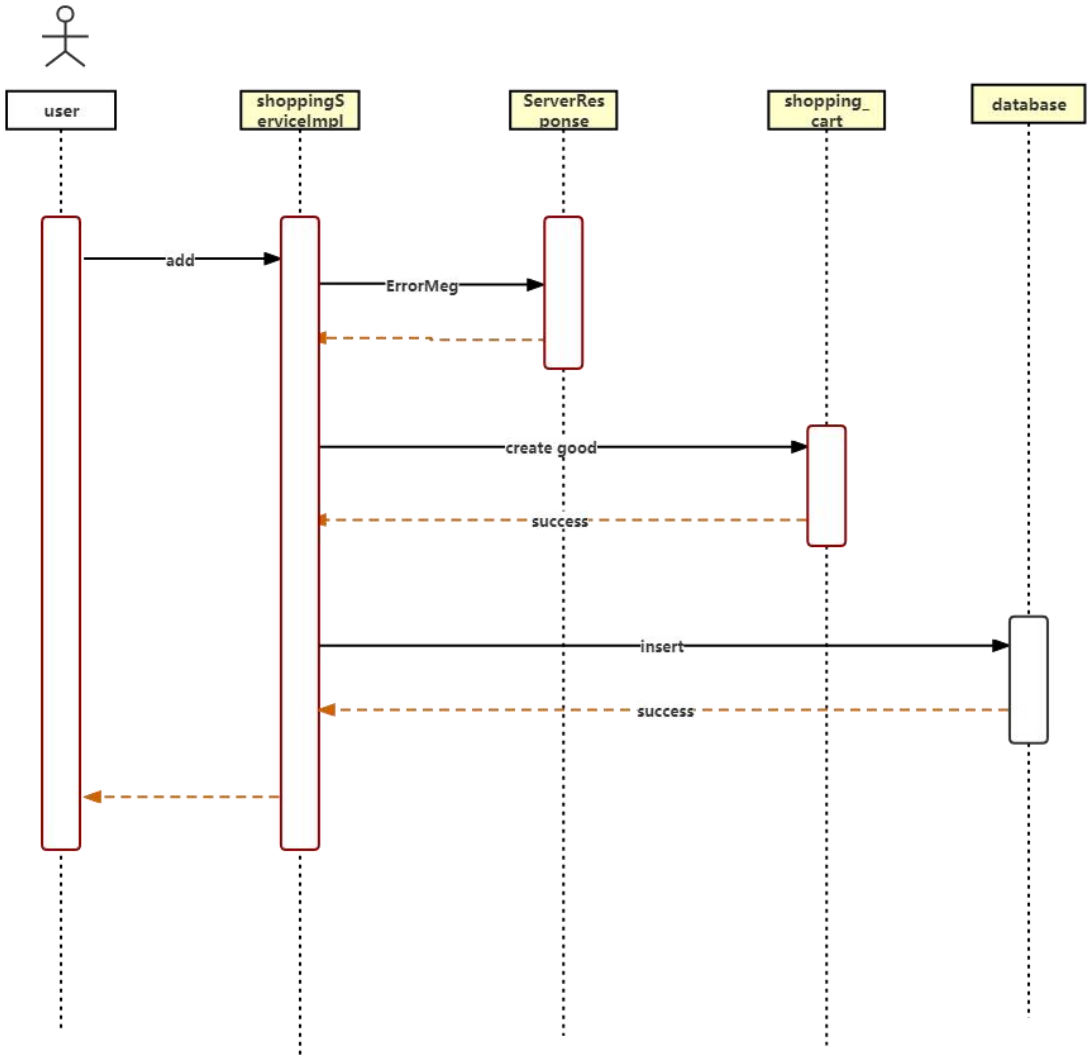
注释：无

用例名称：管理员发货
参与者：管理员
简要说明： 管理员点击立即发货
前置条件： 进行管理员登录
基本事件流： 1．管理员点击立即发货 2.出现弹窗 3．管理员输入物流信息，点击提交 4.用例终止
后置条件： 订单转入售后状态
注释：无

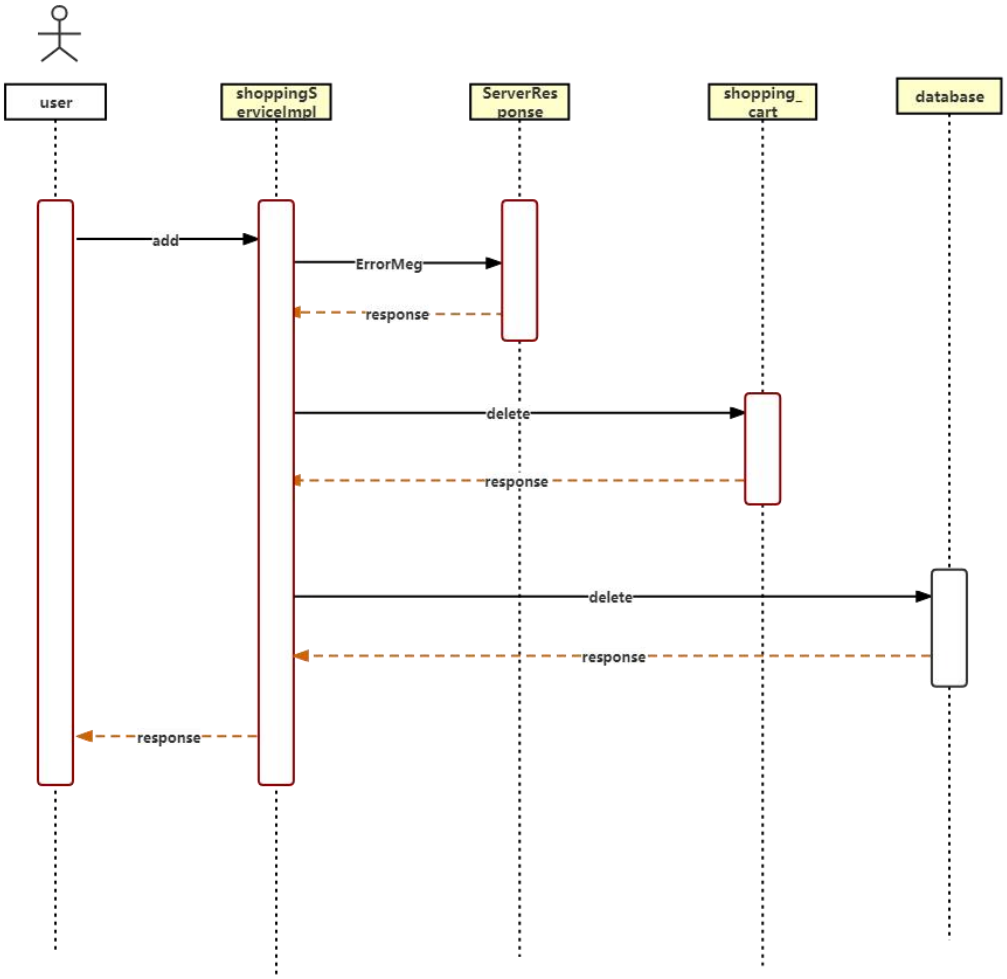
用例名称：售后
参与者：顾客
简要说明： 顾客点击联系客服或者同意售后
前置条件： 进行顾客登录
基本事件流： 1．顾客点击联系客服或者同意售后 2.顾客进入客服，物品转入售后状态 3.用例终止
后置条件：
注释：无

时序图

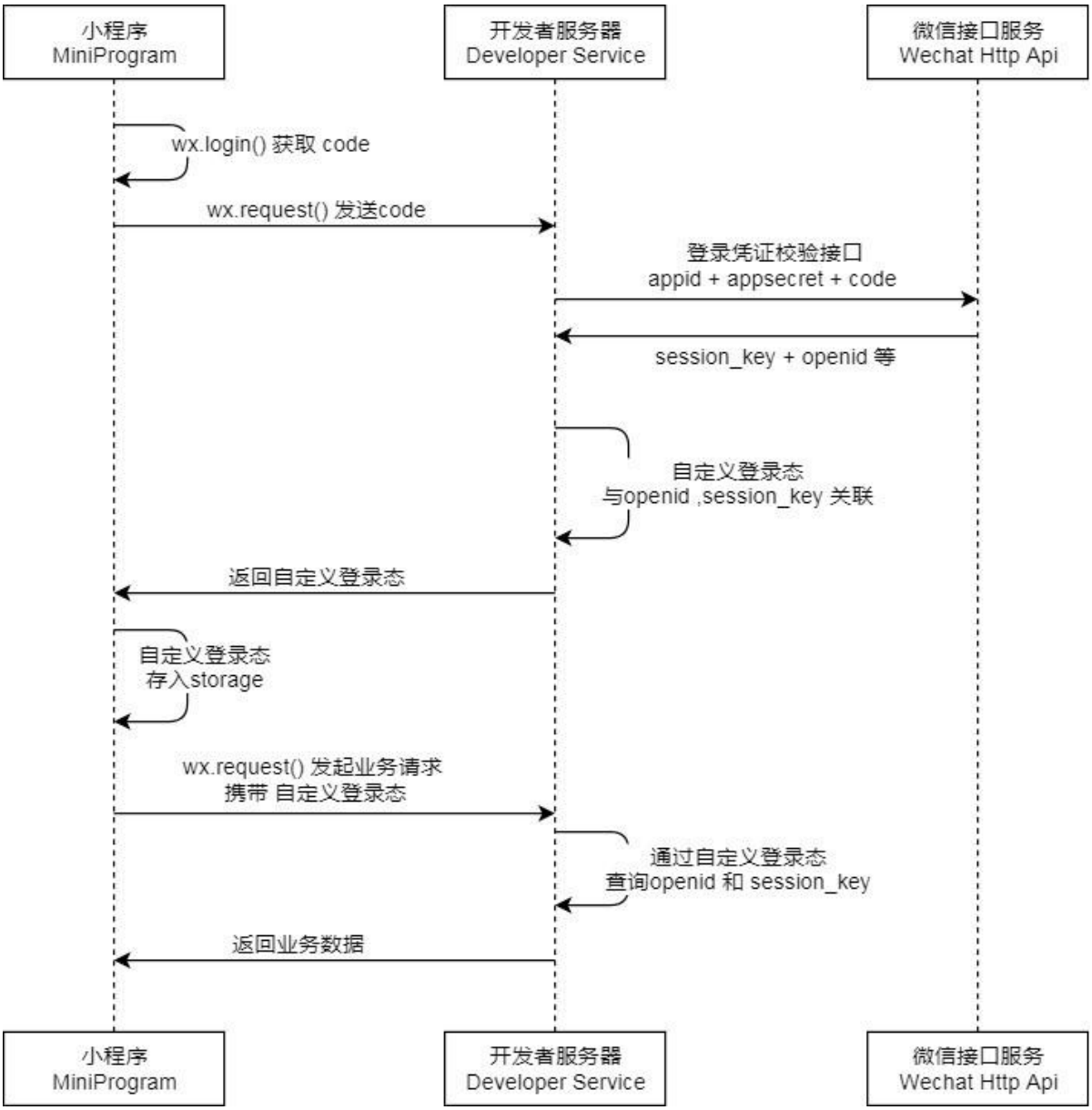
1.添加商品到购物车



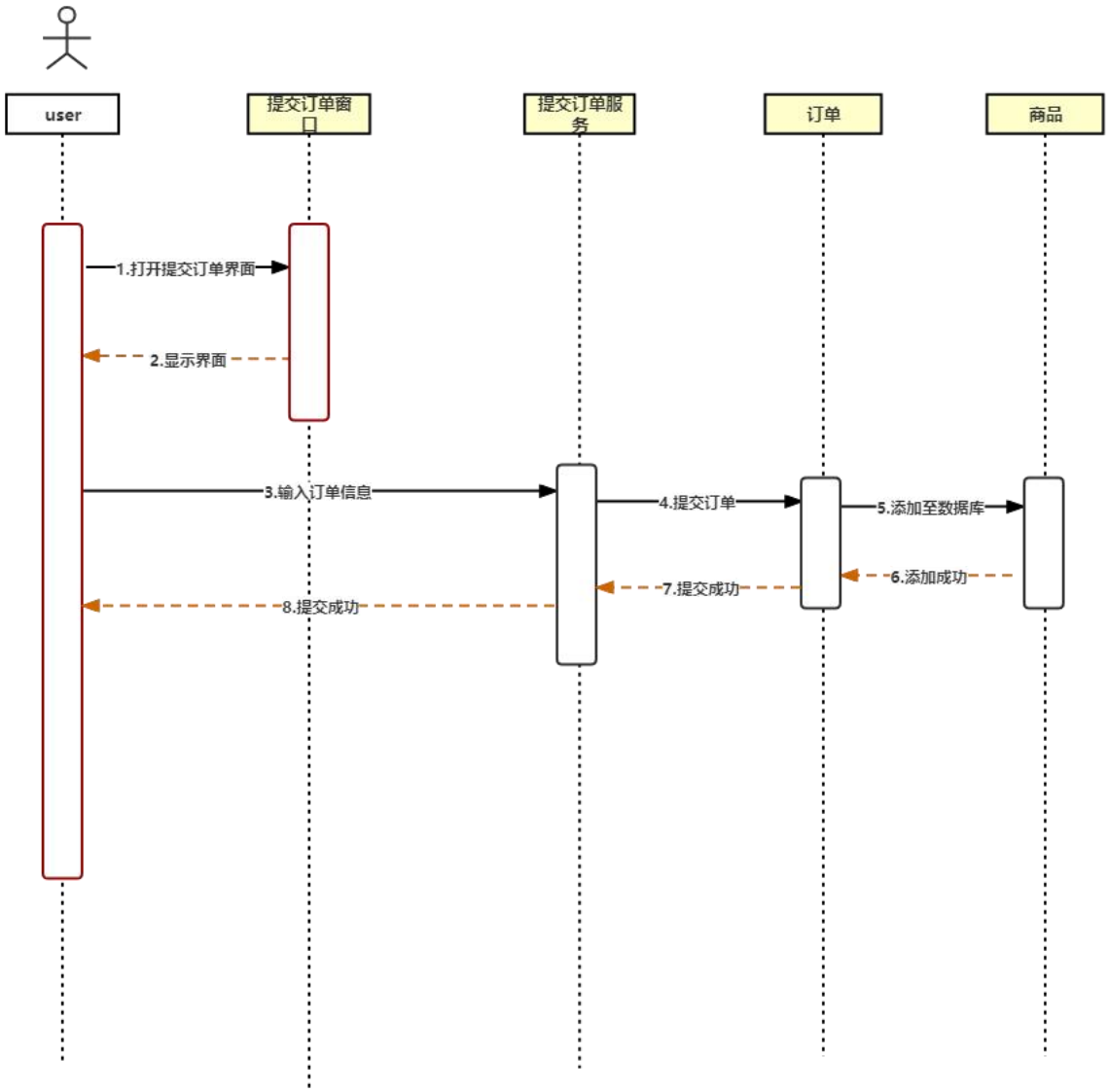
2.购物车删除商品


user
shoppingS erviceImpl
ServerRes ponse
shopping_ cart
database
add
ErrorMeg
response
delete
response
delete
response
response

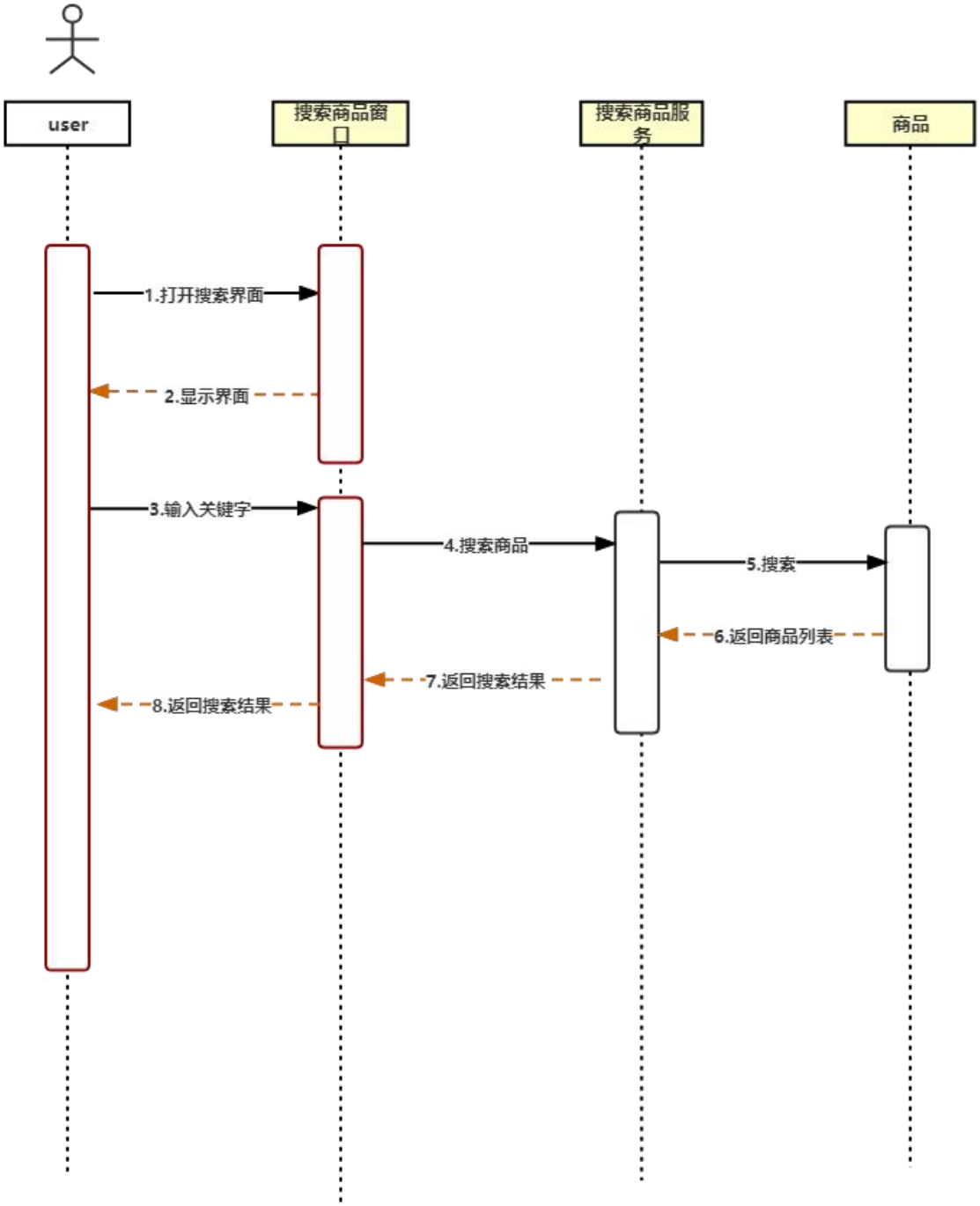
3.登录注册


小程序
MiniProgram
开发者服务器
Developer Service
微信接口服务
Wechat Http Api
wx.login() 获取 code
wx.request() 发送code
登录凭证校验接口
appid + appsecret + code
session_key + openid 等
自定义登录态
与openid ,session_key 关联
返回自定义登录态
自定义登录态
存入storage
wx.request() 发起业务请求
携带 自定义登录态
通过自定义登录态
查询openid 和 session_key
返回业务数据
小程序
MiniProgram
开发者服务器
Developer Service
微信接口服务
Wechat Http Api

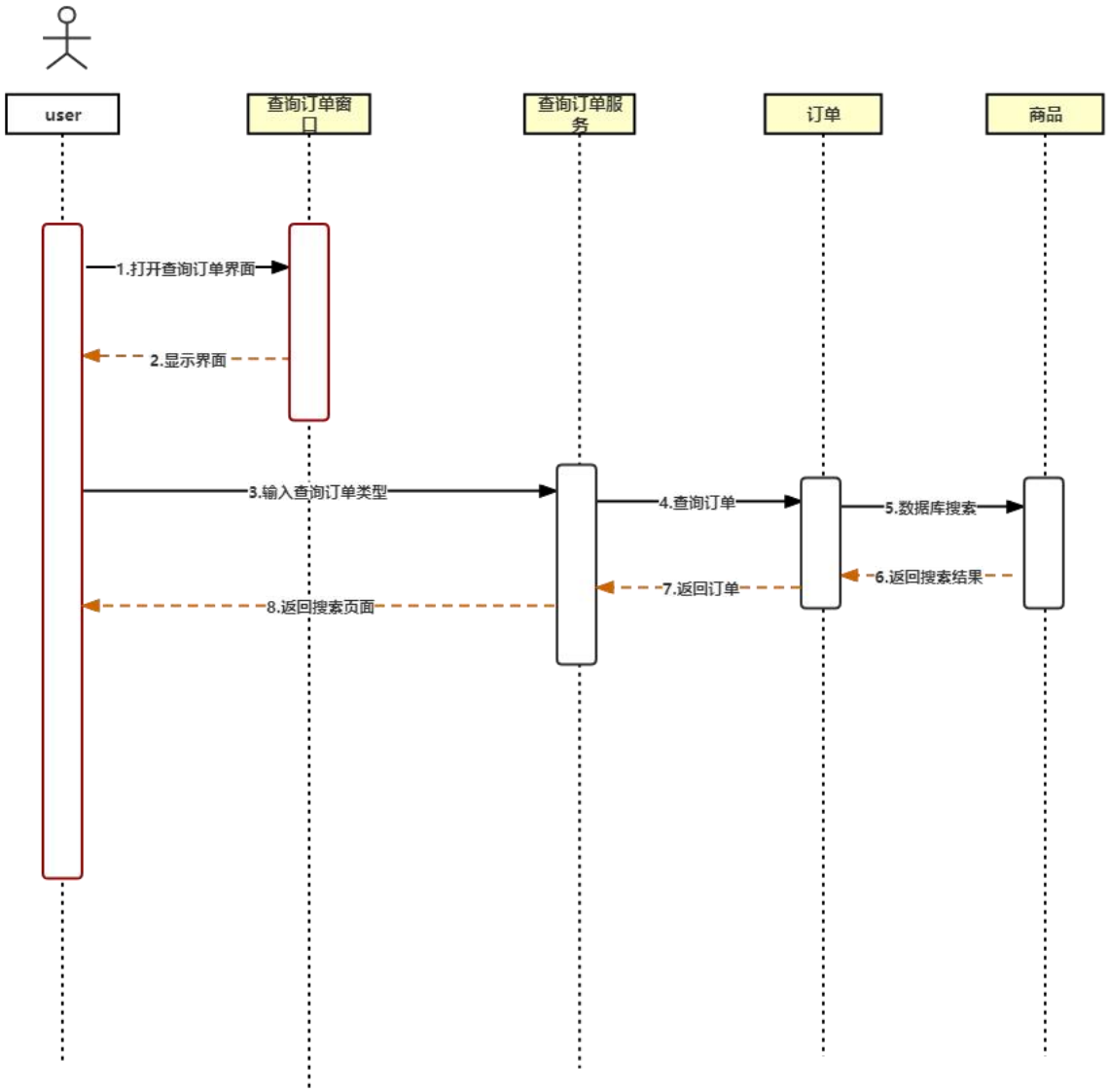
4.提交订单


user
提交订单窗口
提交订单服务
订单
商品
1.打开提交订单界面
2.显示界面
3.输入订单信息
4.提交订单
5.添加至数据库
6.添加成功
7.提交成功
8.提交成功

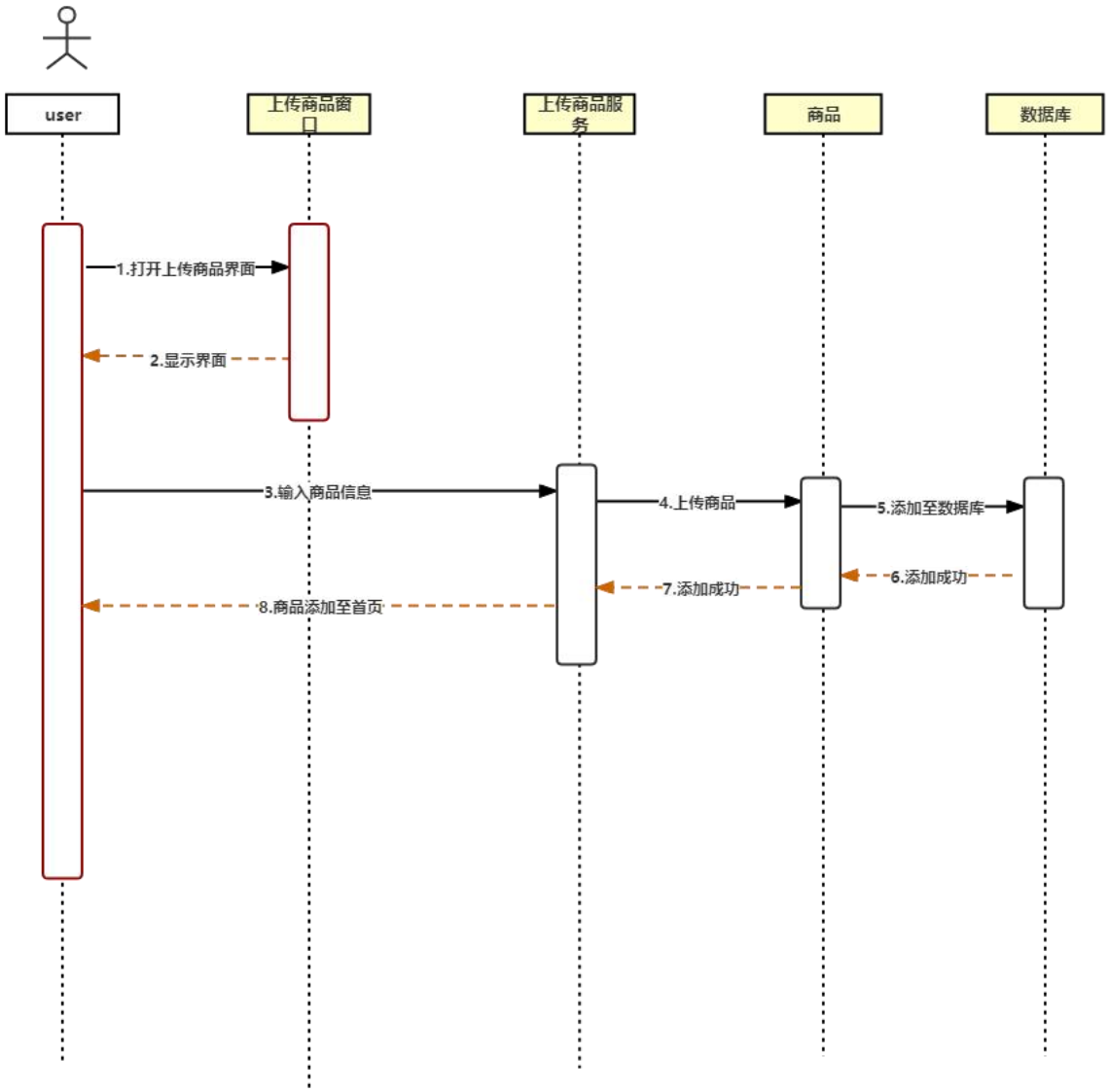
5.搜索商品


user
搜索商品窗口
搜索商品服务
商品
1.打开搜索界面
2.显示界面
3.输入关键字
4.搜索商品
5.搜索
6.返回商品列表
7.返回搜索结果
8.返回搜索结果

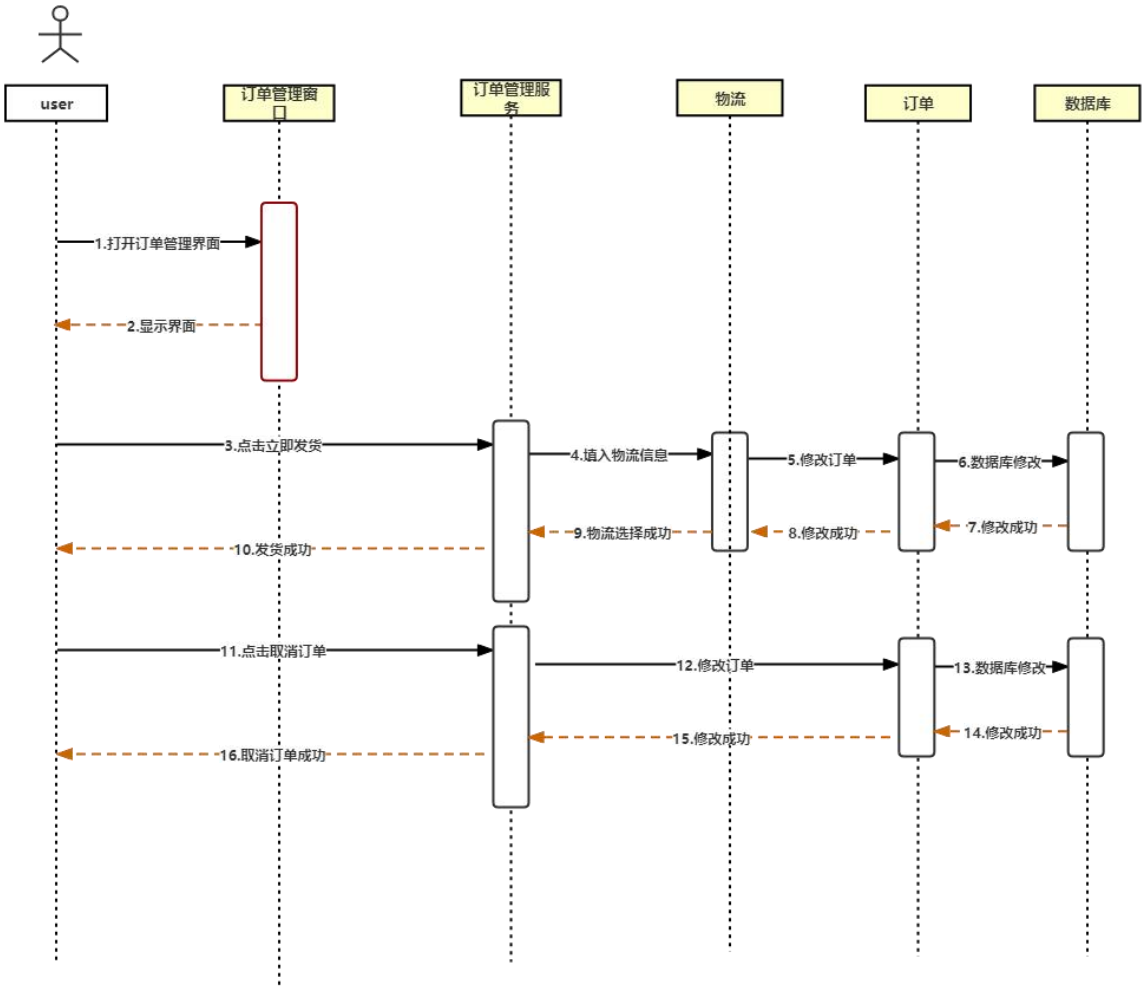
6.订单查询


user
查询订单窗口
查询订单服务
订单
商品
1.打开查询订单界面
2.显示界面
3.输入查询订单类型
4.查询订单
5.数据库搜索
6.返回搜索结果
7.返回订单
8.返回搜索页面

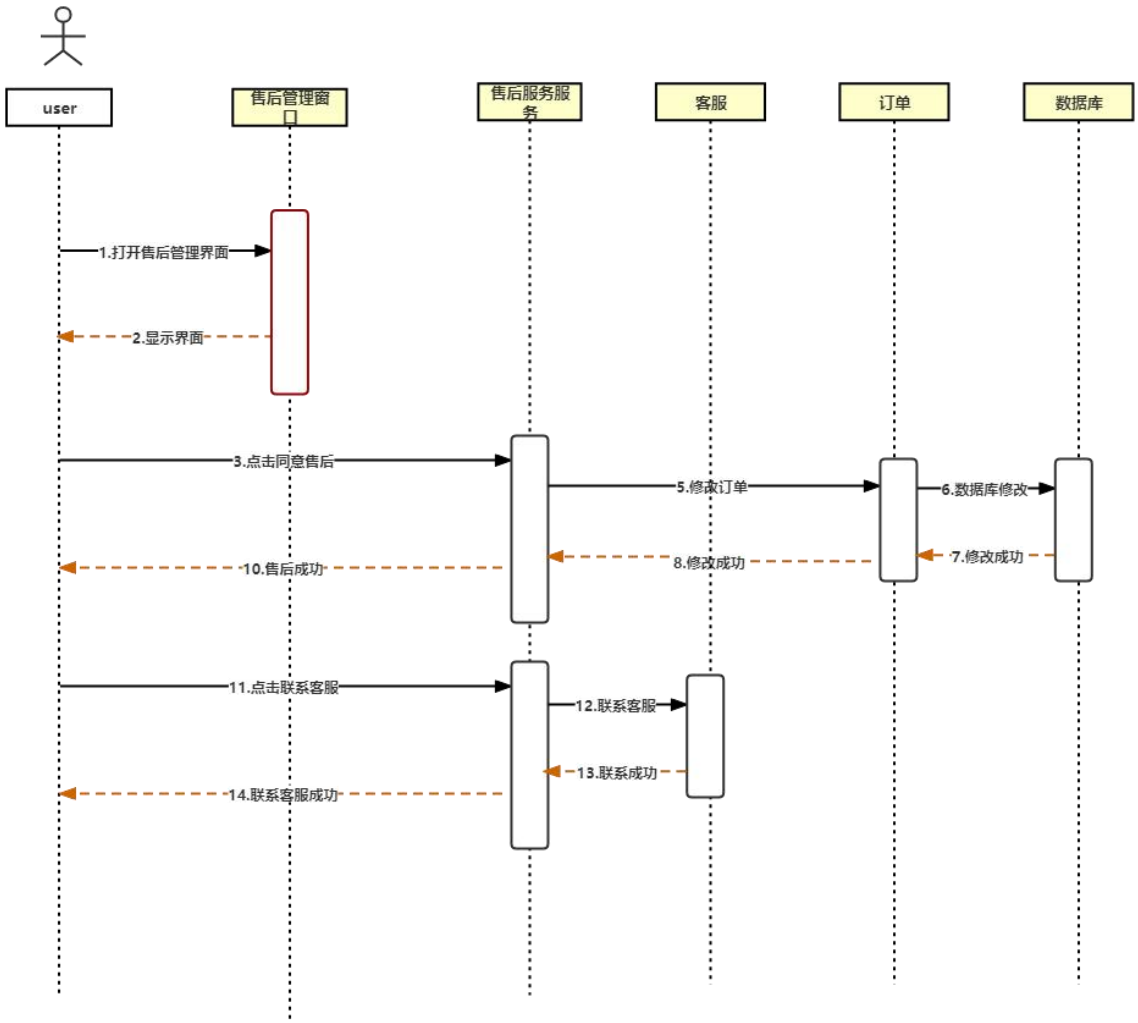
7.上传商品


user
上传商品窗口
上传商品服务
商品
数据库
1.打开上传商品界面
2.显示界面
3.输入商品信息
4.上传商品
5.添加至数据库
6.添加成功
7.添加成功
8.商品添加至首页

8.后台订单管理


user
订单管理窗口
订单管理服务
物流
订单
数据库
1.打开订单管理界面
2.显示界面
3.点击立即发货
4.填入物流信息
5.修改订单
6.数据库修改
7.修改成功
8.修改成功
9.物流选择成功
10.发货成功
11.点击取消订单
12.修改订单
13.数据库修改
14.修改成功
15.修改成功
16.取消订单成功

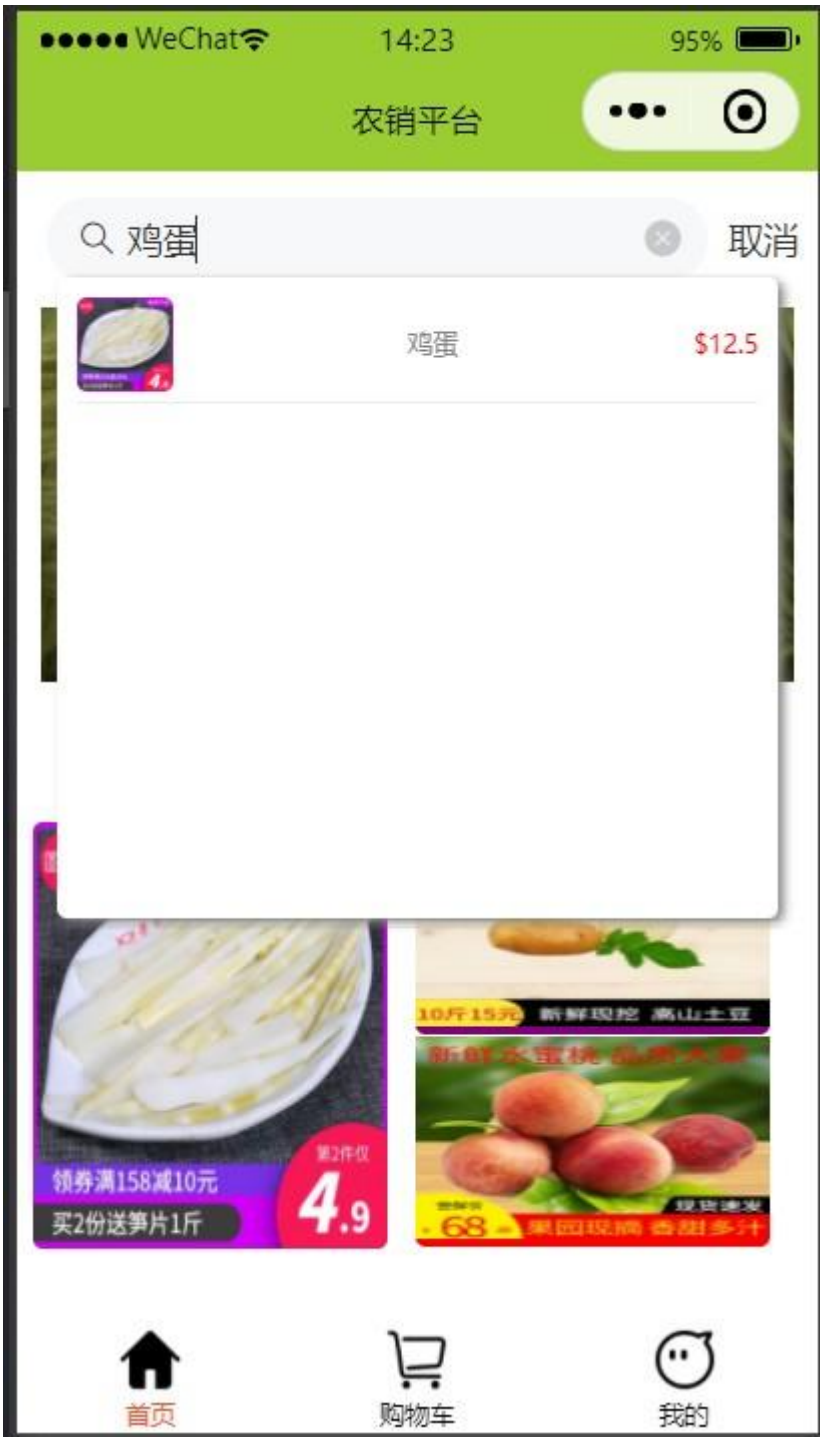
9.售后


user
售后管理窗口
售后服务服务
客服
订单
数据库
1.打开售后管理界面
2.显示界面
3.点击同意售后
5.修改订单
6.数据库修改
7.修改成功
8.修改成功
10.售后成功
11.点击联系客服
12.联系客服
13.联系成功
14.联系客服成功

产品实现说明

1.商城首页


WeChat
0:35
100%
农销平台
领券满158减10元
买2份送笋片1斤
4.9
商品
产品特点
77.5
源于大自然的礼物
农产品
$55.6
99.9
助农产品
领券满158减10元
买2份送笋片1斤
4.9
鸡蛋
$12.5
25.6
新鲜水蜜桃 品质大果
现货速发
68
果园现摘 香甜多汁
首页
购物车
我的



说明：

1.搜索框，引入Vant组件实现，点击后会出现搜索栏，通过API db.RegExp实现名称搜索并展示

```
<van-search
value="{{ value }}"
placeholder="请输入搜索关键词"
shape = "round"
show-action
bind:change="onSearch"
bind:cancel="onCancel"
bind:focus="search_case_show"
bind:blur="search_case_close"
style="background-color: #FF502F;"/>
<view class="lay_col_cen" wx:if="{{search_case}}">
    <view class="lay_col_sta case search_case">
        <scroll-view style="width: 100%;height: 100%;" scroll-y="true">

            <view class="lay_col_sta pad_20">
                <block wx:for="{{search_list}}" wx:key="index">
                    <view class="lay_row_spa">
                        <image src="{{item.img[0]}}" class="g_img"></image>
                        <text>{{item.name}}</text>
                        <text style="color:red;">${{item.price}}</text>
                    </view>
                    <van-divider style="width: 100%;" custom-style="margin-
top:10rpx;margin-bottom:10rpx;"/>
                </block>
            </view>
        </scroll-view>
    </view>
</view>
```

```
    onSearch(e){
        let that = this
        if(e.detail){
            wx.showLoading({ tit
                le: '搜索中',
            })
            db.collection('product').where({ nam
                e: db.RegExp({
                    regexp: e.detail,
                    options: 'i',
                })
            }).get().then(res=>{

                wx.hideLoading()
                console.log(res)
                that.setData({
                    search_list:res.data
                })
            })
        }else{
            that.setData({ search_li
                st:[],
            })
        }
    },
```

2.轮播图，调用API swiper 实现，在swiper内实现一个block块，通过for循环调用云数据库中的swiper

并展示

```
        <swiper class="swp_a" autoplay="true" circular="true" indicator-
dots="true">
            <block wx:for="{{swaper}}" wx:key="index">
                <swiper-item>
                    <image src="{{item.src}}"></image>
                </swiper-item>
            </block>
        </swiper>
```

3.今日特惠，使用image链接云存储中的图片并展示

4.商品列表，在block块使用for循环遍历云数据库中的product，并返回图片、商品名称、商品价格等信息，使用navigator链接到商品详情页

```
<block wx:for="{{pro_list}}" wx:key="index">
                <navigator class="pro_detail"
url="../product_detail/product_detail?id={{item._id}}">
                    <view style="width: 100%;">
                        <image src="{{item.img[0]}}" class="pro_img"></image>
                        <view>
                            <text>{{item.name}}</text>
                        </view>
                        <view>
                            <view>
                                <text style="color: red;">${{item.price}}</text>
                                <text style="color: #888;font-size: 25rpx;margin-
left: 15rpx;text-decoration: line-through;">{{item.h_price}}</text>
                            </view>
                            <van-icon name="cart-circle-o" size="50rpx"
color="#FF502F"/>
                        </view>
                    </view>
                </navigator>
            </block>
```

5.导航栏，在app.json中用API tobBar实现导航，并为每个图标设置选中与未选中两个状态，调用change选择

```
    onChange(event) {
        this.setData({ active: event.detail });
    },
```

2.商品详情页

1.轮播图，调用API swiper 实现，在swiper内实现一个block块，通过for循环调用云数据库中的swiper并展示

2.商品信息展示，调用云数据库中的product获取商品名称、价格、销量等信息

3.选择规格，通过wx:for获取云数据库中product的规格字段并展示，并设置选择事件，将数值传给后端函数

```
        <view class="lay_row_sta">
            <text style="font-size: 25rpx;">规格</text>
        </view>
        <van-divider style="width: 100%;" custom-style="margin-top:10rpx;margin-
bottom:10rpx;"/>
        <view class="lay_row_sta pro_specs">
            <block wx:for="{{product.specs}}" wx:key="index">
                <van-button wx:if="{{item != select_specs}}"  type="primary"
size="mini" style="margin: 10rpx;" bind:click="select_specs" data-specs="
{{item}}">{{item}}</van-button>
                <van-button wx:else type="danger" size="mini" style="margin:
10rpx;" bind:click="select_specs" data-specs="{{item}}">{{item}}</van-button>
            </block>
        </view>
```

4.选择数量，引用Vant组件van-stepper，默认为1，设置bind:change事件，将值传给后端

```
        <view class="lay_row_spa">
            <text>选择的数量：{{select_num}}</text>
            <van-stepper value="{{ select_num }}" bind:change="select_num"
style="margin-left: 200rpx;"/>
        </view>
```

5.底部购物栏，引用Vant组件van-goods-action，并设置点击事件

```
<van-goods-action custom-class="">
  <van-goods-action-icon icon="chat-o" text="客服" dot open-type="contact"/>
  <van-goods-action-icon icon="cart-o" text="购物车" bind:tap="to_shopping_car"/>
  <van-goods-action-icon icon="shop-o" text="店铺" />
  <van-goods-action-button text="加入购物车" type="warning"
bind:click="add_shopping_car"/>
  <van-goods-action-button text="立即购买" bind:click="add_order"
bind:click="add_order"/>
</van-goods-action>
```

- 点击客服可跳转至官方给出的客服
- 点击购物车链接到购物车页面
- 点击加入购物车，则调用add_shopping_car函数，将前面的选择信息生成一个购物车商品，加入云数据库中的shopping_car
- 点击立即购买，则调用add_order函数，将前面的选择信息生成订单，加入云数据库中的order

3.购物车页面

1.购物车列表，从数据库中获取product的信息，通过wx:for循环展示全部

2.每个商品前有一个Vant引入的单选框，绑定选择事件，将选择情况传给后端，并重新计算总价.同时，每一个商品有数量选择框，绑定一个bind:change事件将商品数量传给后端，并重新计算总价

```
<van-checkbox-group value="{{ result }}" bind:change="select_product"
style="width: 100%;">
        <block wx:for="{{product_list}}" wx:key="index">
            <view class="lay_row_spa">
                <van-checkbox name="{{index}}">
                </van-checkbox>
                <view class="lay_row_spa pad_20 product_case" style="margin-left:
20rpx;" data-id="{{item._id}}" bindlongpress="delete_product">
                    <view class="lay_row_sta" style="width: 60%;">
                        <image src="{{item.product_img}}" class="product_img">
</image>

                        <view class="lay_col_spa" style="width: 50%;height:
100rpx;font-size: 25rpx;">
                            <text>{{item.product_name}}</text>
                            <text>{{item.product_specs}}</text>
                            <text style="color: red;">￥:{{item.product_price}}
</text>
                        </view>
                    </view>
                    <van-stepper value="{{ item.product_name }}" data-index="
{{index}}" bind:change="select_product_num"/>
                </view>
            </view>
        </block>
    </van-checkbox-group>
```

3.提交栏，包括是否全选、计算总价、提交订单等，其中通过wx:if判断是否进行全选，并绑定全选事件，重新计算总价。

```
<van-submit-bar
  price="{{ all_price }}"
  button-text=" 提 交 订 单 "
  bind:submit="submit_order"
  tip="{{ true }}"
>
  <van-tag type="primary" wx:if="{{!is_all}}" data-name="全选"
bindtap="select_all" style="font-size: 30rpx;">全选</van-tag>
  <van-tag type="primary" wx:else data-name="取消全选" bindtap="select_all"
style="font-size: 30rpx;">取消全选</van-tag>
</van-submit-bar>
```

4.计算总价，利用当前页面选择的信息进行总价的计算

```
    // 计算总价
    get_all_price(pro){
        let that=this
        let all_price=0
        let product_list=that.data.product_list
        if(pro.length==0){
            that.setData({ all_price
                :0
            })
        }else{
            for(let
                i=0;i<pro.length;i++){ let
                index =parseInt(pro[i])
                all_price = all_price +
(product_list[index].product_num*product_list[index].product_price)
                if(i+1==pro.length){
                    that.setData({ all_price:parseFloat((all_price*100).toFi
                        xed(2))
                    })
                }
            }
        }
    }
```

5.提交订单，需要把购物车中的信息通过API　wx.setStorage缓存传给订单页面，通过wx.navigateTo导航至订单页面

```
    submit_order(){
        let that = this
        if(that.data.result.length != 0){
            wx.showLoading({ tit
              le: '提交中',
            })
            let goods = []
            for(let i=0;i <
                that.data.result.length;i++){ goods.push(that.data.product_list[th
                at.data.result[i]*i]) if(i+1==that.data.result.length){
                    wx.hideLoading()
                    wx.setStorage({
                        key:"goods",
                        data:goods
                    })
                    wx.navigateTo({
                        url: '../add_order/add_order',
                    })
                }
            }
        }else{
            wx.showToast({ title
              : '请选择商品',
              icon:"none"
            })
        }
    }
```

6.删除商品，长按商品进行删除，通过界面交互API　wx.showModal实现，在确定删除后，操作数据库

删除该商品

```
// 删除商品
delete_product(e){
    let that = this
    let id = e.currentTarget.dataset.id
    wx.showModal({
      title: '提示',
      content: '是否删除该商品',
      success(res){
          if(res.confirm){
              console.log('用户点击确定')
              wx.showLoading({
                title: '删除中',
              })
              db.collection('shopping_car').doc(id).remove().then(res=>{
                wx.hideLoading()
                wx.showToast({
                  title: '删除成功',
                })
                that.get_shopping_car()
            })
          }else if(res.cancel){
            console.log('用户点击取消')
          }
      }
    })
}
```

4.下单页面

1.添加地址，绑定bindtap事件，在后端通过API wx.chooseAddress获取，进入添加地址页面

```
// 我的地址
get_address(){
    let that = this
    wx.chooseAddress({
        success (res)
          { console.log('我的地址
          ',res) that.setData({
              address:res
          })
        }
      })
},
```

2.商品信息，通过API　wx.getStorage获取从购物车页面传来的商品信息，通过wx:for循环展示，在商品信息中可以增减数量并计算总价

```
    //获取商品
    get_goods(){
        let that = this
        wx.getStorage({
            key: 'goods',
            success (res) {
              console.log('获取到的商品',res.data)
              that.setData({ goods:r
                  es.data
              })
              that.get_all_price(res.data)
            }
        })
    },
```

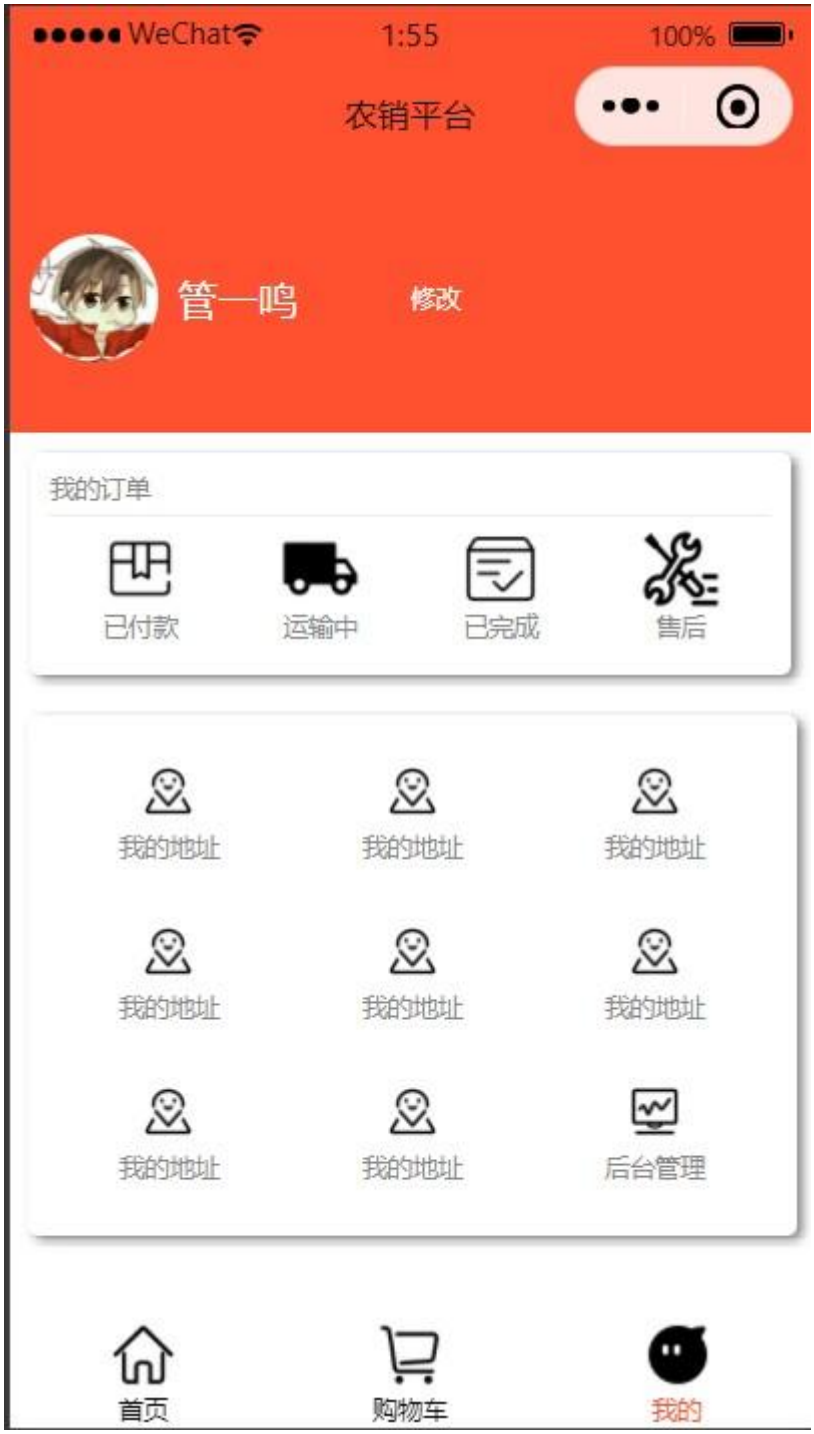
3.备注，通过textarea实现，绑定事件，将内容传送给后端

```
    <!-- 备注 -->
    <view class="lay_col_cen pad_20 address_case">
        <view class="lay_row_sta">
            <text>备注：</text>
        </view>
        <van-divider style="width: 100%;" custom-style="margin-top:10rpx;margin-
bottom:10rpx"/>
        <textarea name="" id="" cols="30" rows="10" style="width: 100%;height:
150rpx;" bindinput="input_remarks"></textarea>
    </view>
```

4.下单，下单前需要计算总价，和购物车类似，不再赘述。下单后，将数据传给数据库的order，并添加类型为“已付款”方便以后查看，下单成功后用wx.removeStorage移除缓存

```
//下单事件
    add_order(){
        let that = this
        if(that.data.address=="" || that.data.goods.length == 0){
            wx.showToast({
              title: '请填写信息',
              icon:"none"
            })
        }else{
            wx.showLoading({
              title: '下单中',
            })

            db.collection('order').add({
                data:{
                    address:that.data.address,
                    goods:that.data.goods,
                    remarks:that.data.remarks,
                    all_price:that.data.all_price,
                    type:"已付款",
                    time:db.serverDate()
                }
            }).then(res=>{
                wx.hideLoading()
                wx.showToast({
                  title: '下单成功',
                })
                wx.removeStorage({
                  key: 'goods',
                  success(res){
                    wx.navigateBack({
                        delta: 1,
                    })
                  }
                })
                console.log('下单成功',res)
            }).catch(err=>{
                wx.showLoading({
                    title: '下单失败',
                    icon:"error"
                })
                console.log('下单失败',err)
            })
        }
    },
```

5.我的页面


WeChat
1:55
100%
农销平台
管一鸣
修改
我的订单
已付款
运输中
已完成
售后
我的地址
我的地址
我的地址
我的地址
我的地址
我的地址
我的地址
我的地址
后台管理
首页
购物车
我的

1.个人信息，从数据库中获取user里的用户信息并展示头像、昵称等信息

2.用户注册登录，利用API　wx.getUserProfile一键获取微信用户信息实现注册登录，每次进入该页面会加载注册登录函数

```
// 注册
    register(e){
        let that = this
        wx.showModal({
            title: '提示',
            content: '您还未注册，是否注册',
            success(res){
                if(res.confirm){ console.log
                    ('用户点击确定')
                    wx.showLoading({
                      title: '注册中',
                    })
                    wx.getUserProfile({
                        desc: '用于完善会员资料', // 声明获取用户个人信息后的用途，后续会
展示在弹窗中，请谨慎填写
                        success: (userInfo) =>
                            { db.collection('user').add({
                                data:{
                                    userInfo:userInfo.userInfo
                                }
                            }).then(user=>{ wx.hideLoadin
                                g() wx.showToast({
                                  title: '注册成功',
                                })
                                that.login()
                            })
                        }
                    })
                }else
                  if(res.cancel){ console.log('
                  用户点击取消')
                }
            }
        })
    },
```

3.订单查看，可以点击已付款、运输中进入订单页面

4.后台管理员登录，Vant引入van-popup弹窗功能，通过input输入账号密码，在数据库中查询，若成功则登入，进入后台页面

```
<!-- wxss页面 -->
<!-- 登录弹窗 -->
<van-popup show="{{ show_login }}" round closeable position="bottom" custom-
style="height: 60%"
bind:close="close_login_case">
    <view class="lay_col_sta pad_20">
        <view class="lay_row_cen" style="height: 100rpx;">
            <text>登录</text>
        </view>
        <view class="lay_row_sta" style="width: 70%;margin-top: 90rpx;">
            <van-icon name="friends-o" />
            <input type="text" placeholder="账号" style="margin-left: 20rpx;"
data-name="username" bindinput="input_msg"/>
        </view>
        <van-divider style="width: 70%;" custom-style="margin-top:10rpx;margin-
bottom:10rpx;"/>
        <view class="lay_row_sta" style="width: 70%;margin-top: 50rpx;">
            <van-icon name="friends-o" />
            <input type="password" placeholder="密码" style="margin-left: 20rpx;"
data-name="password" bindinput="input_msg"/>
        </view>
        <van-divider style="width: 70%;" custom-style="margin-top:10rpx;margin-
bottom:10rpx;"/>
        <button style="width: 70%;margin-top: 80rpx;" bindtap="login_admin"
disabled="{{is_login?'true':''}}">登录</button>
    </view>
</van-popup>
```

```
// js页面
// 后台登录
    login_admin(){
        let that = this
        wx.showLoading({
            title: '登录中',
          })
        if(that.data.username == '' || that.data.password ==
            ''){ wx.showToast({
              title: '请输入账号或密码',
              icon:"none"
            })
        }
        else{
            that.setData({ is_login:
                true
            })
            db.collection('admin').where({ use
                rname:that.data.username,
                password:that.data.password,
            }).get().then(res=>{ console.log
                ('登录',res) that.setData({
                    is_login:false
                })
                wx.hideLoading()
                if(res.data.length==0){
                    wx.showToast({
                      title: '账号或密码错误',
                    })
                }else{
                    app.globalData.admin=res.data[0]
                    wx.navigateTo({
                      url: '../admin_index/admin_index',
                    })
                }
            })
        }
    },
```

6.订单页面

1.头部导航，分为已付款、运输中、已完成，绑定事件进行订单类型的选择，每次点击传入订单类型的名称，并调取获取订单函数获得不同类型的订单

```
<!-- 头部 -->
<view class="lay_row_spa head">
    <view class="lay_row_cen {{title=='已付款'?'select_tile':''}}" data-name="已付
款" bindtap="select_title">
        <text style="margin-left: 80rpx;">已付款</text>
    </view>
    <view class="lay_row_cen {{title=='运输中'?'select_tile':''}}" data-name="运输
中" bindtap="select_title">
        <text style="margin-left: 80rpx;">运输中</text>
    </view>
    <view class="lay_row_cen {{title=='已完成'?'select_tile':''}}" data-name="已完
成" bindtap="select_title">
        <text style="margin-left: 80rpx;">已完成</text>
    </view>
</view>
```

```
    // 选择订单类型
    select_title(e){ le
        t that = this
        let name =e.currentTarget.dataset.name
        that.setData({
            title:name
        })
    },
```

2.获取不同类型订单并展示，get_order函数通过选择的订单类型在数据库中进行选择，并通过orderBy进行时间排序，时间戳通过utils中的time函数转化为“xxxx-xx-xx xx:xx:xx”格式，在wxss中通过wx:for遍历数据并展示

```
    // 获取订单
    get_order(type){ le
        t that = this
        wx.showLoading({ titl
          e: '获取订单中',
        })
        db.collection('order').where({ type:
            type
        }).orderBy('time','desc').get().then(res=>{ wx.hid
            eLoading()
            that.setData({ order:that.change_time(res
                .data)

            })
            console.log('获取订单成功',res.data)
        }).catch(err=>{ wx.hideLoadi
            ng()
            console.log('获取订单成功',err)
        })
    },
```

7.后台页面

1.上传商品、商品管理、订单管理分别通过navigator链接到相应页面

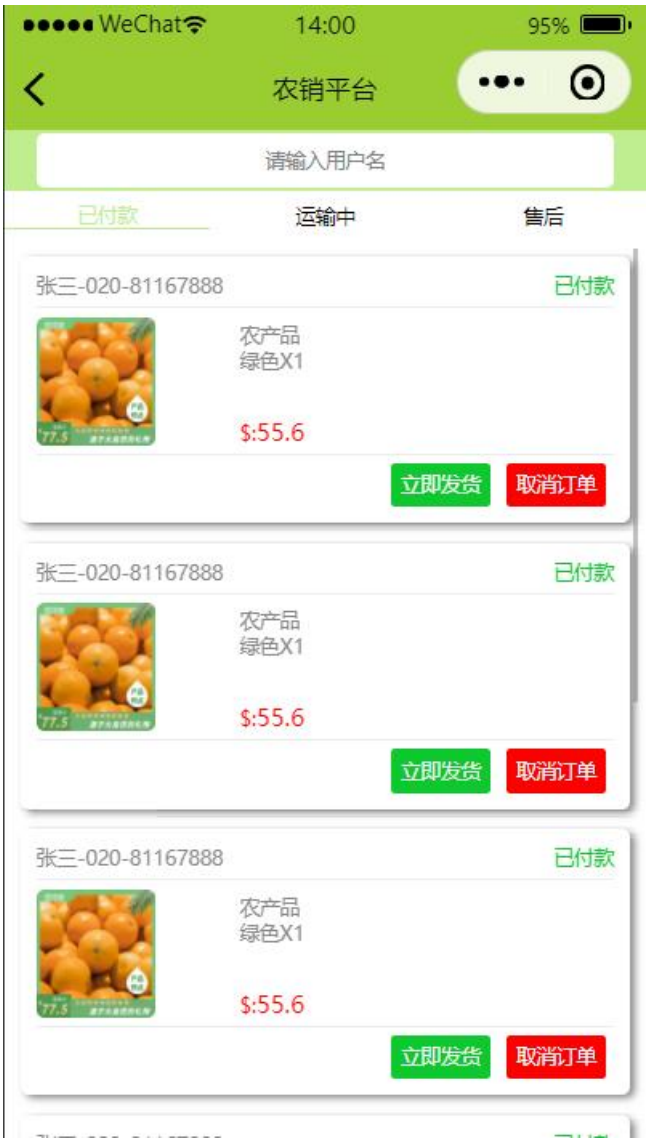
8.后台上传商品页面

上传商品，管理员可以输入商品信息，商品分类，商品规格等信息进行商品上传，上传后商品添加至商城首页的商品列表

所有的信息通过block，wx:for循环展示，每个按钮绑定事件

```
    <!-- 商品分类 -->
    <view class="lay_col_cen pad_20 case" style="margin-top: 40rpx;">
        <view class="lay_row_spa">
            <text style="font-size: 30rpx;">商品分类</text>
        </view>
        <van-divider style="width: 100%;" custom-style="margin-top:10rpx;margin-
bottom:10rpx;"/>
        <view class="lay_row_sta" style="flex-wrap: wrap;">
            <van-radio-group value="{{ classify }}" direction="horizontal"
bind:change="select_classify">
                <block wx:for="{{classify}}" wx:key="index">
                    <van-radio name="{{item.name}}" >{{item.name}}</van-radio>
                </block>
            </van-radio-group>
        </view>
    </view>
    <!-- 商品规格 -->
    <view class="lay_col_cen pad_20 case" style="margin-top: 40rpx;">
        <view class="lay_row_spa">
            <text style="font-size: 30rpx;">商品规格</text>
            <view class="lay_row_spa" style="width: 70%;margin-left: 80rpx;">
                <view class="lay_row_cen input_case" style="width: 70%;">
                    <input type="text" placeholder="请输入规格" value="
{{input_specs}}" bindinput="input_msg" data-name="input_specs"/>
                </view>
                <van-button type="primary" size="small" style="margin-left:
50rpx;" bind:click="add_specs">新增</van-button>
            </view>
        </view>
        <van-divider style="width: 100%;" custom-style="margin-top:10rpx;margin-
bottom:10rpx;"/>
        <view class="lay_row_sta" style="flex-wrap: wrap;">
            <block wx:for="{{specs}}" wx:key="index">
                <view class="lay_row_cen specs_case">
                    <text>{{item}}</text>
                </view>
            </block>
        </view>
    </view>
```

提交时通过async实现提交，提交图片通过wx.cloud.uploadFile API，通过云函数wx.cloud.callFunction product将信息提交到数据库，并展示在商品首页




```
async submit(){
        let that = this
        let img = that.data.img
        let img_detail = that.data.img_detail
        wx.showLoading({
          title: '上传中',
        })
        for(let i = 0;i<img.length;i++){
            var timestamp = new Date().getTime()
            await wx.cloud.uploadFile({
                cloudPath: 'product/'+timestamp+''+i+''+'.png',
                filePath: img[i], // 文件路径
            }).then(async res => {
                // get resource ID
                console.log(res.fileID)
                img[i] = res.fileID
                if(i+1==img.length){
                    for(let j = 0;j<img_detail.length;j++){
                        var timestamp = new Date().getTime()
                        await wx.cloud.uploadFile({
                            cloudPath:
'product_detail/'+timestamp+''+j+''+'.png',
                            filePath: img_detail[j], // 文件路径
                        }).then(res_1 => {
```

```
                                // get resource ID
                                console.log(res_1.fileID)
                                img_detail[j] = res_1.fileID
                                if(j+1 == img_detail.length){
                                    console.log('1')
                                    wx.cloud.callFunction({
                                        name:"product",
                                        data:{
                                            name:that.data.name,
                                            h_price:that.data.h_price,
                                            price:that.data.price,
                                            specs:that.data.specs,
                                            img:img,
                                            img_detail:img_detail,
                                        }
                                    }).then(res_2=>{
                                        wx.hideLoading()
                                        console.log('2',res_2)
                                    })
                                }
                            }).catch(error => {
                                // handle error
                            })
                        }
                    }).catch(error => {
                        // handle error
                    })
                }
            },
```

9.后台订单管理页面


WeChat
14:00
95%
农销平台
请输入用户名
已付款
运输中
售后
张三-020-81167888
已付款
农产品
绿色X1
$:55.6
立即发货
取消订单
张三-020-81167888
已付款
农产品
绿色X1
$:55.6
立即发货
取消订单
张三-020-81167888
已付款
农产品
绿色X1
$:55.6
立即发货
取消订单


WeChat
14:00
95%
农销平台
请输入用户名
已付款
运输中
售后
张三-020-81167888
运输中
鸡蛋
绿色X4
$:12.5

订单管理，可以查看已付款、运输中、售后三个状态，每次选择在后端更新订单

```
<!-- 订单 -->
<scroll-view scroll-y="true" class="order">
<view class="lay_col_sta pad_20">
    <block wx:for="{{order}}" wx:key="index">
        <view class="lay_col_cen pad_20 case" style="font-size: 25rpx;margin-
bottom: 25rpx;">
            <view class="lay_row_spa">
                <text>{{item.address.userName}}-{{item.address.telNumber}}
</text>
                <text wx:if="{{item.type == '已付款'}}" style="color: rgb(15, 199,
46);margin-left: 300rpx;">{{item.type}}</text>
                <text wx:if="{{item.type == '运输中'}}" style="color: rgb(187,
134, 35);margin-left: 300rpx;">{{item.type}}</text>
                <text wx:if="{{item.type == '售后'}}" style="color: rgb(238, 32,
17);margin-left: 300rpx;">{{item.type}}</text>

            </view>
            <van-divider style="width: 100%;" custom-style="margin-
top:10rpx;margin-bottom:10rpx;"/>
            <view class="lay_row_spa">
                <image src="{{item.goods[0].product_img}}" class="goods_img">
</image>
                <view class="lay_col_spa" style="height: 140rpx;width: 70%;align-
items: flex-start;margin-left: 100rpx;">
                    <text>{{item.goods[0].product_name}}</text>
                    <text style="color: #888888;">
{{item.goods[0].product_specs}}X{{item.goods[0].product_num}}</text>
                    <text style="color: red;margin-top: 50rpx;">$:<text
style="font-size:  30rpx;">{{item.goods[0].product_price}}</text></text>
                </view>
            </view>
            <van-divider style="width: 100%;" custom-style="margin-
top:10rpx;margin-bottom:10rpx;"/>
            <view class="lay_row_end">
                <view class="lay_row_cen" style="width: 20%;" wx:if="
{{item.type=='已付款'}}">
                    <van-button type="primary" color="rgb(15, 199, 46)"
size="mini" data-id="{{item._id}}"
                    bind:click="deliver_case_show">立即发货</van-button>
                </view>
                <view class="lay_row_cen" style="width: 20%;" wx:if="
{{item.type=='已付款'}}">
                    <van-button type="primary" color="red" size="mini"data-id="
{{item._id}}"
                    bind:click="update_order_state">取消订单</van-button>
                </view>
            </view>
        </view>
    </block>
</view>
```

管理员可以点击立即发货，弹出物流选择，商品变为运输中

```
<!-- 物流弹出层 -->
<van-popup show="{{ deliver_case_show }}" round closeable position="bottom"
custom-style="height:40%;" bind:close="deliver_case_close">
<view class="lay_row_cen" style="height: 8vh;">
    <text>立即发货</text>
</view>
<scroll-view scroll-y="true" class="deliver_case">
    <view class="lay_col_sta pad_20">
        <view class="lay_row_cen">
            <picker mode="selector" style="width: 100%;" range="{{express}}"
bindchange="select_express">
                <view class="lay_row_spa" style="height: 70%;">
                    <text style="font-size: 25rpx;width: 30%;">快递公司</text>
                    <view class="lay_row_end" style="width: 70%;font-size:
25rpx;margin-left: 400rpx;">

                        <text>{{select_express}}</text>
                        <van-icon name="arrow"/>
                    </view>
                </view>
            </picker>
        </view>
        <van-divider style="width: 100%;" custom-style="margin-top:10rpx;margin-
bottom:10rpx;"/>
        <view class="lay_row_spa">
            <text>快递单号</text>
            <input type="text" placeholder="输入快递单号" style="width: 70%;text-
align: right;margin-left: 40rpx;"
             data-name="express_number" bindinput="input_msg"/>
        </view>
        <van-divider style="width: 100%;" custom-style="margin-top:10rpx;margin-
bottom:10rpx;"/>
        <view class="lay_col_cen" style="height:200rpx;">
            <van-button type="primary" disabled="{{is_submit?'true':''}}"
style="width: 100%;margin-top: 100rpx;" size="large"
            bindtap="deliver_goods">立即发货</van-button>
        </view>
    </view>
</scroll-view>
</van-popup>

//后端
    // 立刻发货
    deliver_goods(){ le
        t that = this
        if(that.data.select_express == '请选择快递' || !that.data.express_number){
            wx.showToast({ title:
              '请填入信息',
              icon:"none"
            })
        }else{
            wx.showLoading({
              title: '发货中',
            })
            that.setData({ is_submit:true})
```

```
            wx.cloud.callFunctio
                n({ name:"order
                ", data:{
                    method:"deliver_goods",
                    id:that.data.order_id,
                    logistics:{
                        select_express :
                        that.data.select_express,
                        express_number:
                        that.data.express_number,
                    }
                }
            }).then(res=>{ wx.h
                ideLoading()
                that.setData({
                    select_express:'请选择快递',
                    express_number:"",
                    deliver_case_show:false,
                    order_skip:0,
               })
               that.get_order('已付款',0)
           })
       }
   },
```

点击取消订单，商品变为售后

```
    // 更新订单状态
    update_order_state(e){ l
        et that = this
        let id =e.currentTarget.dataset.id
        wx.showModal({
            title: '提示',
            content: '是否取消订单',
            success (res) {
                if (res.confirm)
                    { console.log('用户点击确定
                    ') that.setData({
                        order_skip:0
                    })
                    wx.showLoading({ tit
                      le: '取消订单中',
                    })
                    wx.cloud.callFunction({
                         name:"order",
                        data:{
                            method:"cancel_order",
                            id:id,
                        }
                    }).then(order=>{ console.lo
                        g(order)
                        wx.hideLoading()

that.get_order(that.data.order_state,that.data.order_skip)
                    })
                } else if (res.cancel)
                { console.log('用户点击取消')
                }
            }
        })
    },
```

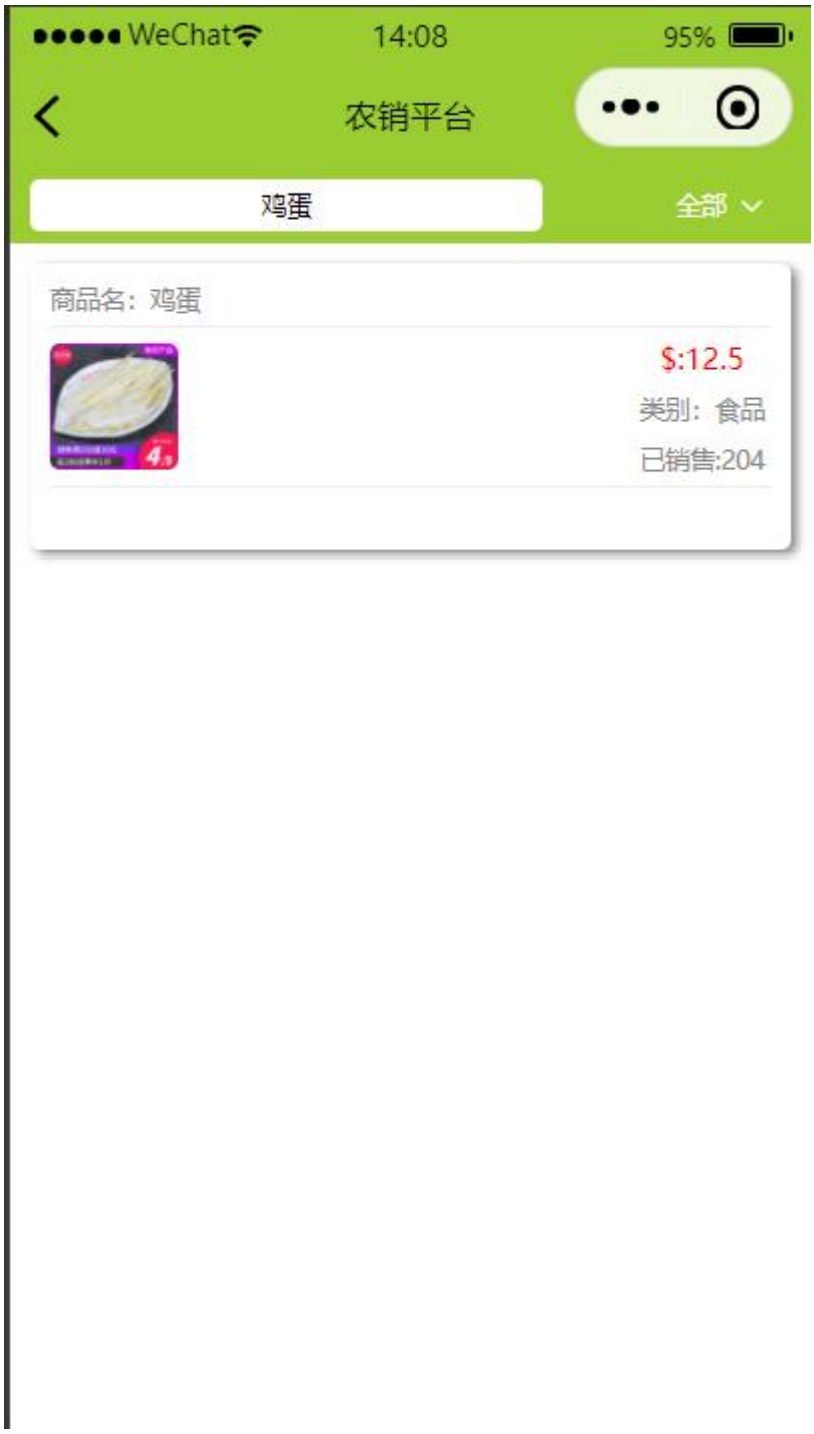
10.后台商品管理页面

可以查看所有商品，按类型、名称搜索商品，通过scroll-view实现，每个商品通过navigator链接到商品详情页，通过block和wx：for循环展示商品信息

```
<!-- 内容 -->
<scroll-view style="width: 100%;height: 94vh;" scroll-y="true">
    <view class="lay_col_sta pad_20">
        <block wx:for="{{product}}" wx:key="index">
            <navigator url="../product_detail/product_detail?id={{item._id}}"
class="lay_col_cen pad_20 case" style="margin-bottom: 20rpx;font-size: 25rpx;">
                    <view class="lay_row_sta">
                        <text>商品名：{{item.name}}</text>
                    </view>
                    <van-divider style="width: 100%;" custom-style="margin-
top:10rpx;margin-bottom:10rpx;"/>
                    <view class="lay_row_spa">
                        <image src="{{item.img[0]}}" class="pro_img"></image>


                        <view class="lay_col_spa" style="height:130rpx;width:
130rpx;margin-left: 400rpx;">
                            <text style="color: red;font-size: 30rpx;">$:
{{item.price}}</text>
                            <text>类别：{{item.select_classify}}</text>
                            <text style="color: #888888;">已销售:
{{item.sales_volume}}</text>
                        </view>
                    </view>
                    <van-divider style="width: 100%;" custom-style="margin-
top:10rpx;margin-bottom:10rpx;"/>
                    <view class="lay_row_end" style="margin-left:700rpx">
                        <text>{{item.time}}</text>
                    </view>
            </navigator>
        </block>
    </view>
</scroll-view>
```

支持按类型搜索，通过picker组件实现，在后端调用相应云函数wx.cloud.callFunction product_manage

```
    <view class="lay_row_cen" style="width: 25%;font-size: 25rpx;color:
#ffffff;margin-left: 80rpx;">
        <picker range="{{classify_list}}" range-key="name"
bindchange="select_classify">
            <text style="margin-right: 12rpx;">{{classify}}</text>
            <van-icon name="arrow-down" />
        </picker>
    </view>

    // 选择分类
    select_classify(e){
        let that = this
        let classify = that.data.classify_list[e.detail.value*1].name
        that.setData({

            classify:classify
        })
        if(classify=='全部
            '){ that.get_product()
        }else{
            wx.showLoading({ tit
                le: '搜索中',
            })
            wx.cloud.callFunction({ name
                :"product_manage", data:{
                    method:"to_classify",
                    classify:classify
                }
            }).then(res=>{ wx.hideLo
                ading()
                console.log('获取商品',res.result.data)
                that.setData({ product:res.resu
                    lt.data
                })
            })
        }
    },
```



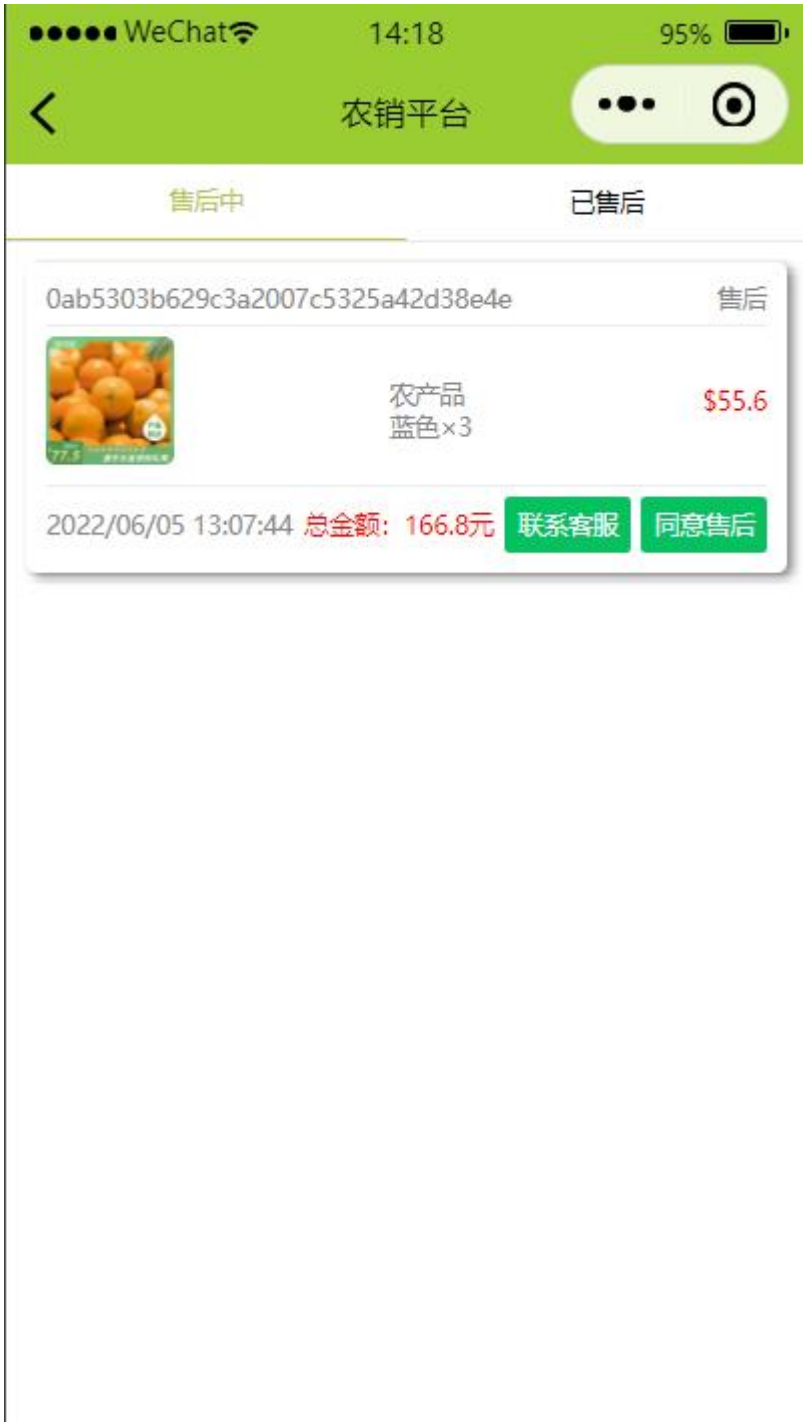
支持按名称搜索，将名称传给后端，调用云函数wx.cloud.callFunction product_manage

```
    //搜索商品
    search(e){
        let that = this
        if(e.detail.value){
            wx.showLoading({ tit
                le: '搜索中',
            })
            wx.cloud.callFunction({ name
                :"product_manage", data:{
                    method:"search",
                    name:e.detail.value
                }
            }).then(res=>{ 、
                wx.hideLoading()

                console.log('获取商品',res.result.data)
                that.setData({
                    product:res.result.data
                })
            })
        }else{
            that.get_product()
        }
    },
```

云函数如下，通过API db.RegExp实现正则化搜索

```
// 云函数入口函数
exports.main = async (event, context) =>
    { const wxContext =
    cloud.getWXContext()

    if(event.method == 'get_product'){
        return await db.collection('product').orderBy("time","desc").get()
    }else if(event.method == 'search'){
        return await
                db.collection('product').where({ nam
                e: db.RegExp({
                regexp: event.name,
                options: 'i',
            })
        }).orderBy("time","desc").get()
    }else if(event.method == 'to_classify'){
        return await
            db.collection('product').where({ select_
            classify:event.classify
        }).orderBy("time","desc").get()
    }else if(event.method == 'inc_sales_volume'){
        return await
            db.collection('product').doc(event.id).update({ data:{
                sales_volume:_.inc(event.num)
            }
        })
    }
}
```

11.售后页面


WeChat
14:18
95%
农销平台
售后中
已售后
0ab5303b629c3a2007c5325a42d38e4e
售后
农产品
蓝色×3
$55.6
2022/06/05 13:07:44 总金额: 166.8元
联系客服
同意售后

用户可以查看售后中和已售后两个页面，用户点击联系客服可通过微信联系客服，通过官方open-type=“contact”实现，点击同意售后，产品状态变为已售后

```
<!-- 内容 -->
<scroll-view style="width: 100%;height: 94vh;" scroll-y="true">
    <view class="lay_col_sta pad_20">
        <block wx:for="{{order}}" wx:key="index">
            <view class="lay_col_cen pad_20 case order">
                <view class="lay_row_spa">
                    <text style="color: #888888;">{{item._id}}</text>
                    <text style="margin-left: 130rpx;">{{item.type}}</text>
                </view>
                <van-divider style="width: 100%;" custom-style="margin-
top:10rpx;margin-bottom:10rpx;"/>
                <view class="lay_col_cen">
                    <block wx:for="{{item.goods}}" wx:for-item="goods"
wx:key="index">
```

```
                        <view class="lay_row_spa" style="margin-bottom: 10rpx;">
                            <image src="{{goods.product_img}}" class="goods_img">
</image>
                            <view class="lay_col_cen" style="height:
120rpx;width: 40%;align-items: flex-start;margin-left: 180rpx;">
                                <text style="padding-top: 20rpx;">
{{goods.product_name}}</text>
                                <text>{{goods.product_specs}}
x{{goods.product_num}}</text>
                            </view>
                            <text style="color: red;">${{goods.product_price}}
</text>
                        </view>
                    </block>
                </view>
                <van-divider style="width: 100%;" custom-style="margin-
top:10rpx;margin-bottom:10rpx;"/>
                <view class="lay_row_spa">
                    <text style="color: #888888;">{{item.time}}</text>
                    <text style="color: red;">总金额：{{item.all_price}}元</text>
                    <van-button type="primary" size="mini" open-type="contact">联
系客服</van-button>
                    <van-button type="primary" size="mini" wx:if="
{{item.aftermarket_state=='售后中'}}"
                    data-id="{{item._id}}" bindtap="agree_aftermarket">同意售后
</van-button>
                </view>
            </view>
        </block>
    </view>
</scroll-view>
```

```
    // 同意售后
    agree_aftermarket(e){
        let that = this
        let id =e.currentTarget.dataset.id
        wx.showModal({
            title: '提示',
            content: '是否同意售后',
            success (res) {
                if (res.confirm)
                    { console.log('用户点击确定
                    ') wx.showLoading({
                      title: '确认售后中',
                    })
                    db.collection('order').doc(id).update({ data
                        :{
                            aftermarket_state:"已售后"
                        }
                    }).then(order=>{ console.
                        log(order)
                        wx.hideLoading()
                        that.get_order(that.data.title)
                    })
                } else if (res.cancel)
                    { console.log('用户点击取消
                    ')}}})},
```

创新和优势

- 集To B和To C端为一体，商家和客户均可使用
- 设计简洁，功能完整清晰

总结与收获

从5月18日到6月15日，将近一个月的时间，完成了这款农产品商城小程序的开发。虽然完成版本比较粗糙，还有很多功能和界面设计等待完善，但在开发的过程中，我学习到了很多前端、后端以及交互的知识，提高了我的开发能力，为今后不被饿死又添加了一项技能。另外一点深有感触，我坚持每周都花费几天去写代码、写文档，所以才可以使项目的进程有序推进，不至于在ddl前不知所措。而每次的迭代过程中，都会看到自己的项目宛如一个进度条慢慢加载满，庞大的项目也是如此慢慢完成的，伟大的成就起始于每一天、每一点的进步。

非常感谢师老师的帮助，两位助教也十分辛苦。总之，完结撒花！