Using Vue to develop Mobile-friendly E-commerce Web-app Front-end Design

By

Jiahua You

Mar 2019

Table of Contents

1 Introduction
1.1 Motivations
1.2 Background

2 Project Design Framework
2.1 Design
2.2 Features
2.3 Implementations
2.4 Technology Stack

3 What is Vue.js
3.1 Programming using Vue.js

4 Background Overview of Program Design
4.1 Main File
4.2 Vuex
4.3 Vue-Router

5 Result, Prototype, and Implemented Function
5.1 Shopping Page Interface
5.2 Shopping Page Categories
5.2 Selecting detail features and functions
5.4 Login Page Authorization
5.5 Shopping Cart Interface
5.6 Checkout Interface
5.7 Run the program

6 Future Work
7 Conclusion

Project Name

Using Vue to develop Mobile-friendly E-commerce Web-app front-end Design

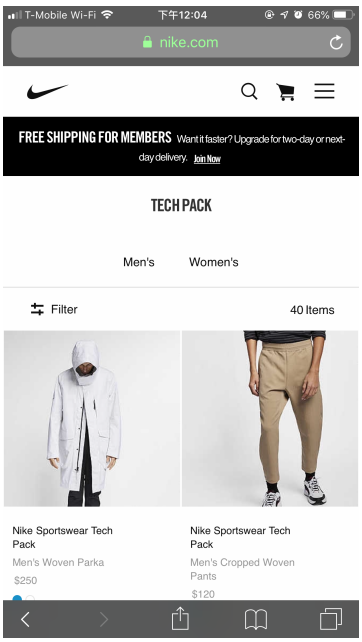
1 Introduction

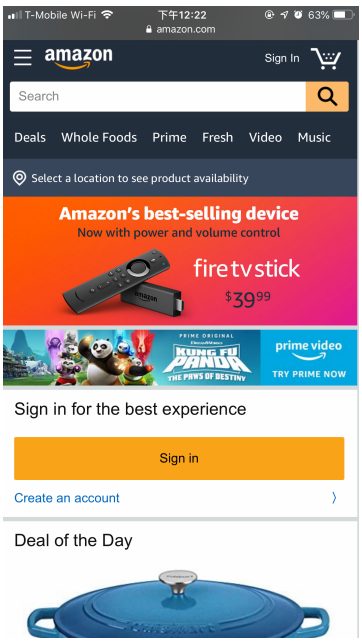
1.1 Motivation

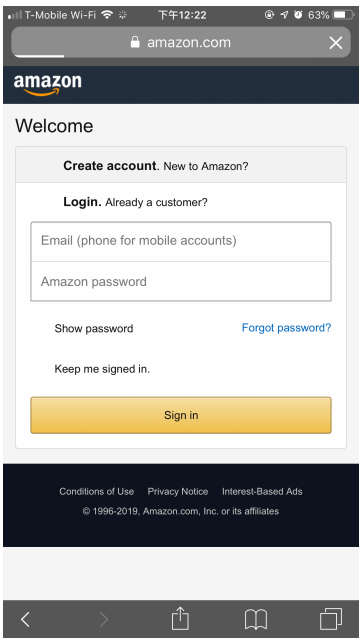
The principal purpose of this project is to design and implement a front-end for an online shopping website for mobile device. Customers can browse the list of commodities from the home page of the application and then place an order to purchase the items.

1.2 Background

Nowadays, smartphone and mobile device becomes the mainstream of online shopping. Take a look at the E-commerce industry, mobile shopping platform shows explosive growth, such as Amazon, Nike, Taobao, JD, etc.. Usually, an E-commerce website where anyone can buy or sell items online. Mobile-friendly E-commerce Web-app site configures nicely to both smartphones, mobile device and tablets.







It is provides a prominent search bar (base on type simple keywords) and they also have implemented a popular grid format with photos that many mobile sites use to ensure the display is clean and simple to use.

2 Project Design Framework

2.1 Design

1. Platform Identification
2. Building consistent look and feel
3. API development
4. API integration
5. Testing
6. Deployment

2.2 Features

1. Authorization
2. Catalogue with categories
3. Different payment methods
4. Tracking purchase
5. Shopping cart

2.3 Implementation

1. Set up the project environment, including the directory structure design and the establishment of the git remote repository.
2. Design and develop the general function module for the application. The main content which including the package of the tool class, such as UI layout, style improvement, navigation development, and page development.
3. Develop and implemented the shopping cart module.
4. Develop order and payment module implementation.
5. Online environment configuration and project automation release.

2.4 Technology stack:

Front-end development:

- **Vue.js**
- **Vue-router**: is the official router for vue.js, it is deeply integrating with vue.js core to make building single page application with vue.js a breeze.
- **Vuex**: state management pattern + library for vue.js applications.
- **Axios**: promise based HTTP client for the browser and node.js.
- **Mock.js**: is a simulation data generator to help the front-end to develop and prototype separate from the back-end progress and reduce some monotony particularly writing automated tests.
- **ES6**: brings new syntax and new awesome features.
- **Less**: is a backward-compatible language extension for CSS.

Back-end development:

- **Node (express) Mock**: back-end data.

3 What is Vue.js

Over the last 10 years, the web page has become more dynamic and more powerful thanks to JavaScript. Programmer have moved a lot of code that was normally on the server side into the browsers and leaving us with thousands of lines of JavaScript code connecting the HTML and CSS files with no formal organization. This is why more and more developer are using JavaScript frameworks like, Angular, React, or **Vue**.

Vue is an approachable, versatile, and performant JavaScript framework that helps you create a more maintainable and testable code base. Vue is progressive JavaScript framework, which means, if there have an existing server-side application, and people usually can plug Vue into just one part of the application which that need a richer, more interactive experience.

Or if people want to develop more business logic into their front-end from the get go, Vue has the core libraries and the ecosystem which developer usually need to scale. Like other front-end frameworks, Vue allows people to take a webpage and split it up into reusable components each having its own HTML, CSS and JavaScript needed to render that piece of the page.

3.1 Programming using Vue.js

As many JavaScript applications, we start from the need to display the data on to our webpage. With Vue, it starts out really simple. For example, we can include the Vue library, create a Vue instance and example, and plug into the root element with the ID if app. And we can also move the data inside an object and change the expression to make the things happen simple.

And the magic of the Vue starts when data changes, Vue automatically updated the HTML. This is because Vue is reactive, meaning that when the data changes, Vue take care of updating all the places are using in the webpage. And this is works with any kinds of data, not just with strings.

4 Background Overview of Program Design

4.1 Main File

In main.js file, which is from the very beginning of the program design and programming, `import` initialize all the required object, which was include,

- Vue object
- Vuex object
- Vue-router object
- Less object

4.2 Vuex

Vuex, it stores the status of each component of this program. We also need to initialize first, in main.js file,

```
import store from '@/vuex/store.js'
```

`store` indicate the status, and it is also store the login permission and access authority.

4.3 Vue-router

As we mentioned above in the introduction, Vue-router, is the official router for vue.js, it is deeply integrating with vue.js core to make building single page application with vue.js a breeze. In index.js file, we need to build and establish the routing mapping relation. Then, in main.js file, we use hook function, `beforeEach()`, to make judgement of the routing, and it is also determines whether the login permission is required or not.

If the current page requires login permissions, and users do not have access authority, Vue-router will direct the page to the login page automatically. If the current page requires login permissions, and users have access authority/ or system doesn't require the user's access authority, Vue-router will jump directly to the page which we need.

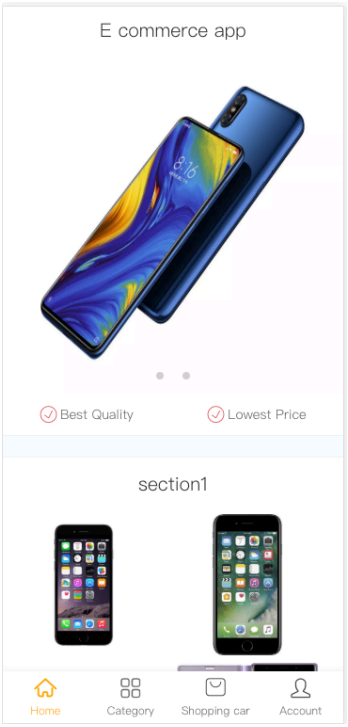
```
Vue.prototype.$api = api;
```

This line of code indicates the sets up for API interception. This is since this program do not have real back-end, and all the data comes from the back-end we used the source from files mock.js. Therefore, we need to intercept the API request. When we get API request, the system will jump to mock.js file to get the source.

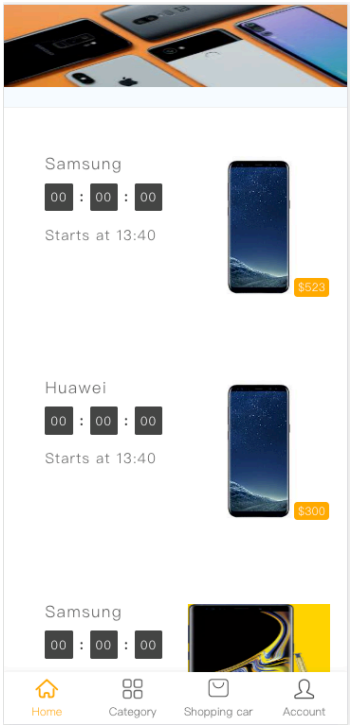
All of the above discussion is the process of loading judgement before the page is displayed, then we can start rendering the current open page for the program.

5 Result, Prototype, and Implemented Function

5.1 Shopping page interface



(Figure 1)



(Figure 2)

Figure 1 shows the interface of the shopping website from the mobile device. Users and on-line customers can browse the items they love for convince.

Feature implemented:

- **Picture Carousel**: Carousel Figure in the front page. Users and customer can see different items when item picture switch to another images.
- **Swiper demos**: Most modern mobile touch slider.
    - Default Setup
    - Navigation
    - Pagination
    - Dynamic Bullets
- **Tab bar Switcher**

Figure 2 implemented the feature which is the countdown clock for limit items for sell at the given time. Users and customer also can see different items and click the pictures to purchase the items.

For programming, in this section, we loading index.vue, first, which is in the directory of src/views/index.vue/

```
import Header from '@/components/index/header.vue'
import Swiper from '@/components/index/swiper.vue'
import Service from '@/components/index/service.vue'
import Section1 from '@/components/index/section1.vue'
import Section2 from '@/components/index/section2.vue'
import Section3 from '@/components/index/section3.vue'
import Section4 from '@/components/index/section4.vue'
import Baseline from '@/common/_baseline.vue'
import Footer from '@/common/_footer.vue'
import index from '@/http/mock.js' //simulation data
```

From the beginning, the program imports all the component which we need to display on the Shopping Page Interface. And mock.js file is the simulation data.

The current, index.vue , we can metaphor as the management center of this shopping page interface, for example, section1.vue, the data inside of the source is to assign it to here, then they can start rendering the current open page for the program, and finally display it.

```
import index from '@/http/mock.js'
```

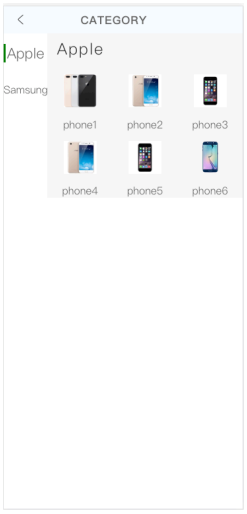
The `index` refer to the name of the data from the mock.js file. The `import` function can loading and import the data from the mock.js file, and then selected the data which we currently need via the method called `beforeCreate()`.

The following program explain that the components assigned to the require data, all the components which need the data are in src/components/index/

```
<v-swiper :swiperData="datas.swiper"/>
<v-service/>
<v-section1 :list="datas.section1.list" :banner='datas.section1.banner'/>
<v-section2 :list="datas.section2.list" :banner='datas.section2.banner'/>
<v-section3/>
<v-section4 :list="datas.section4.list" :banner='datas.section4.banner'/>
<v-baseline/>
```

5.2 Shopping page categories

In this section, we implemented the categories of the shopping item (Figure 3) in the user interface, users and customers and select different categories can explore what type of items they want. And the click the “select pictures” to process the purchase.



(Figure 3)

From the beginning, the program imports all the component which we need to display on the Page of categories Interface. And mock.js file is the simulation data.The programming loading the source form the directory, src/views/Category.vue/

```
import Header from '@/common/_header.vue'
import Aside from '@/components/category/aside.vue'
import category from '@/http/mock.js' //simulations data
```

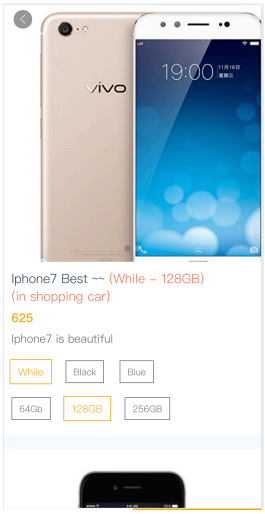
This is same as the implementation as the Main Shopping Page Interface. The current file, Category.vue , we can also metaphor as the management center of this page of categories interface. All the components are in src/components/category/, which we need to import all of the source into the program.

And then, we named the “category” as the simulations data file mock.js. The `import` function can loading and import the data from the mock.js file, and then selected the data which we currently need via the method called `Create()`. The following program explained, this is where we give the data, which we need, to the right components. And all of the sub-components are in the directory src/components/category/

```
<section class="view">
<v-aside :datas="allData.aside"/>
<router-view :datas="allData.aside" />
</section>
```

5.3 Selecting detail feature and function

When user click the “select pictures” to process the purchase. Figure 4 present can select different types or brands of the phone, different color of the phone, and different size of flash-memory of the phone for their demand. And the select item will directly save to the shopping cart and waiting customer for checkout.



(Figure 4)

In this section, we need to run the file, Detail.vue. This function’s source code is comes from src/views/Detail.vue/

This program imports all the component which we need to display on the this Detail Page Interface. And mock.js file is the simulation data. And then, we named the “detail” as the simulations data file mock.js. The `import` function can loading and import the data from the mock.js file, and then selected the data which we currently need via the method called `beforeCreate()`.

And in the directory of src/components/detail/, the following program function is apply to selecting the shopping items feature, which is include, chose.vue, content.vue, swiper.vue.

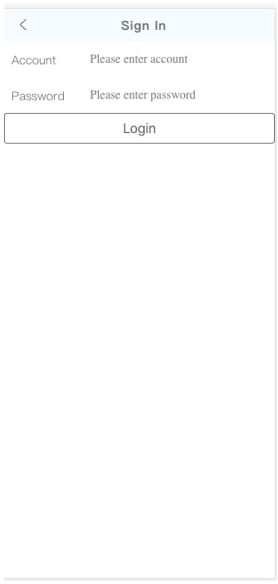
When the customer and user selecting the items (different brands of mobile phones i.e. Apple, Samsung) and selecting the items types (different size of flash-memory and different colors), we will apply “vuex” to modify the state, which of the “Vue” will storage.

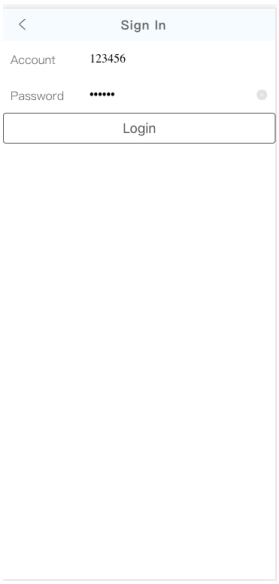
And in the directory of src/components/detail/, the following program function is applying to adding the shopping items into the shopping cart, which is including, footer.vue.

When customer and users click the button “add to shopping cart”, the function `addIntoCar()` will change the state for the shopping cart, and calculate the total price of the shopping items, the program state will re-rendering again for the different price.

5.4 Login page Authorization

Implemented the register/login page, the function will connect to the firebase authentication in the future development. For sign-up, sign-in, and sign-out function. When user sign-up, their account they can enter all the related personal information, and system will automatically save their account information to the back-end system.





(Figure 5) (Figure 6)

In this section, we need to apply login.vue. The source code is coming from the directory src/views/login.vue/

And here we apply the method called, `login()`, to set up and make sure the login status successfully.

```
this.toggle = false;
```

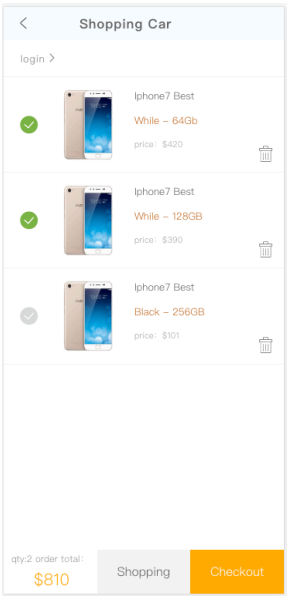
This code explain the status of the login permissions.If customers and user's login to the account successfully, it will modify the login status in Vuex. And then, the status of access of the login permission will shows passed in main.js file. Afterward, when we go to another page for shopping the status of login permission will always shows passed.

From previous discussion, if the current page requires login permissions, and users do not have access authority, Vue-router will direct the page to the login page automatically. If the current page requires login permissions, and users have access authority/ or system doesn't require the user's access authority, Vue-router will jump directly to the page which we need.

5.5 Shopping cart interface

The Shopping cart interface page shows users and customers and select several items they want to purchase at this time. Also, if customers and user notice they don't want to purchase the select items, they can click the button of "trash label" to delete the selected item and remove from the shopping cart.

In addition, the application system will automatically calculate the total price of selecting item, and give the total summary to the user and customers. Once they confirmed, they can click the button "checkout" to process the purchase. If they still want to continues shopping they can easily click "shopping" to go back to continues they shopping.



(Figure 7)

In this section, we need to apply Cart.vue. The source code of the Shopping cart interface is from, src/views/Car.vue/

```
import Header from '@/common/_header.vue'
import Nothing from '@/components/car/nothing.vue'
import Something from '@/components/car/something.vue'
import Footer from '@/components/car/footer.vue'
```

The current program, Cart.vue , we can metaphor as the Page management center of this shopping cart page interface, for example, in the directory of src/components/Cart/, the data inside of the source is to assign it to here, then they can start rendering the current open page for the program, and finally display it. The `import` function can loading and import the data from the mock.js file, and then selected the data which we currently need via the method called `beforeCreate()`.

`Nothing` refers to empty shopping cart.
`Something` refers to item/items in shopping cart.

From the source code directory, src/components/Car/something.vue/
`cut()` function will implementing the function of deleting items. Meanwhile, it is also delete and update the status of shopping cart in Vuex.

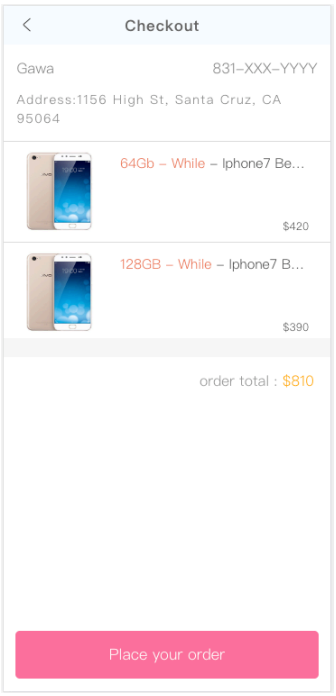
`toggle()` function is the selected items ready to be checking-out. If selected, the item/items will be updated and removed from the shopping cart status.

From the source code directory, src/components/Car/footer.vue/

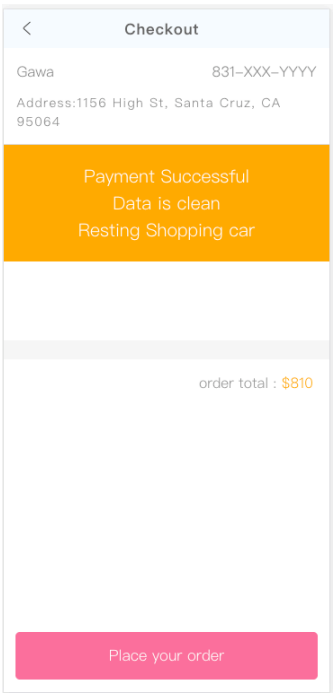
`count ()` function is applied for calculate the items number of the selected items.
`allpay ()` function is applied for calculate the sum of the price of the selected items.
`goPay ()` function is used for judge if cumsters and users want to go to the payment page interface to make a payment, when they click the button called “checkout”. If shopping cart do not have any items, the user can not click the button of “checkout”

5.6 Check out interface

When user and customers click “checkout” button, they system will moving to the next page, check out page, this page require users account information, such as name, cell phone, email, and shipping address. And feedback the total summary of all the user information and shipping information to let user confirmed. Once user and customers confirmed the information, they can click the button for “Place your order” to finish the shopping.



(Figure 8)



(Figure 9)

The final page (Figure 9) shows the users or customer finished the shopping, and the application return the shopping confirmation to let users know the order has been place, and payment is successful process. And, system also provide the shipping address to let the customer know where the item will ship to the destination address.

From the source code directory, [src/components/Car/pay/pay.vue/]

`payConfirm ()` function is applies for when user and customer click the “place your order”, and then the system will refresh the status of Vuex, and will re-set all the status of shopping cart, and update the status of the shopping cart.

Finally, Vue will modify the CSS file, then hide the checkout interface page, and finaaly display the Payment successful page.

5.7 Run the program

This is the link for the source code:

https://github.com/Gawaa/Mobile-friendly-E-commerce-Web-app-Front-end-design-master

Install independencies

Npm install

Serve with hot reload at localhost server

Npm run dev

Build for production with minification

Npm run build

Build for production and view the bundle analyzer port

Npm run build - - report

6 Future work

In the future work for this application development, I will be setting, push notification function, to let users and customers track their progress of shopping, and easily know their shopping and shipping status via notification.

Add “select all” and “deleted all” function in the shopping cart interface. In order to let user shopping experience more and more comfortable in the shopping cart interface.

7 Conclusion

Smartphones are easier to carry indoors and outdoors compared to computer and large desktop device. This project build and develop a simple front-end online shopping application using Vue.js which is called mobile-friendly E-commerce Web-app front-end, that allows users to easily browse between items from various categories and process purchase.

For Programming, Vue is a progressive framework for building user interface. It is designed from the ground up to be incrementally adoptable and can easily scale between a library and a framework depending on different use cases. If we were building a larger application we might want to split thing up at this point into their own components and files. Vue even has a command line interface, which makes it simple to start real project quickly. We might even use single file Vue Components, which contain their own HTML, JavaScript, and Scoped CSS, or SCSS.