# Results and Discussion

## Project Outcome

### **Customer: Product list page with recommendation, filtering, searching, and pagination**

When the customer opens Niubility, he will directly enter the product list page (home page as well). Before some customers stamped a variety of products as “like” or “dislike”, so the algorithm would use the previous data to calculate the result of which product this customer might like even if the customer is new to this application.

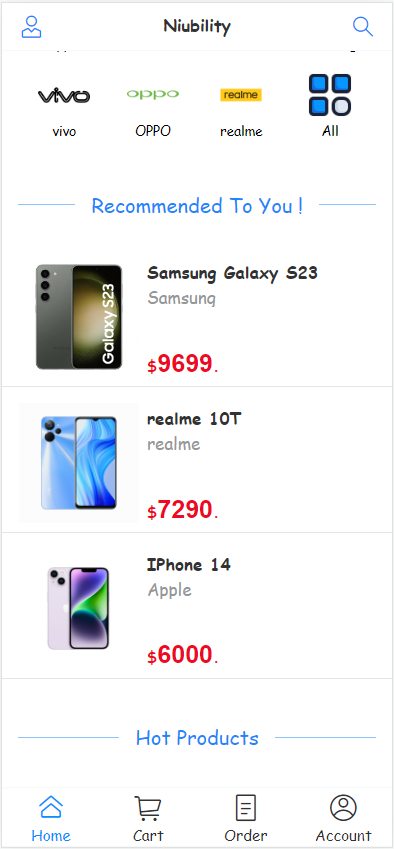


Figure: Recommended list for users

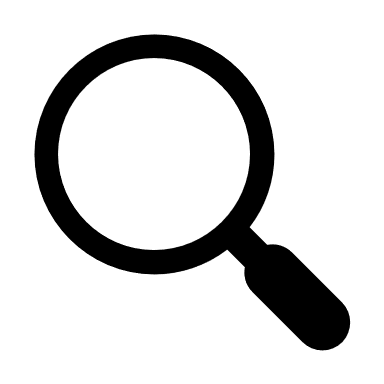
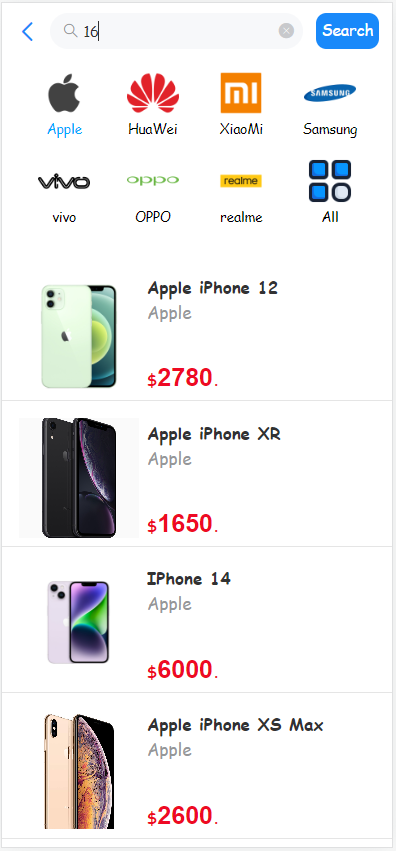
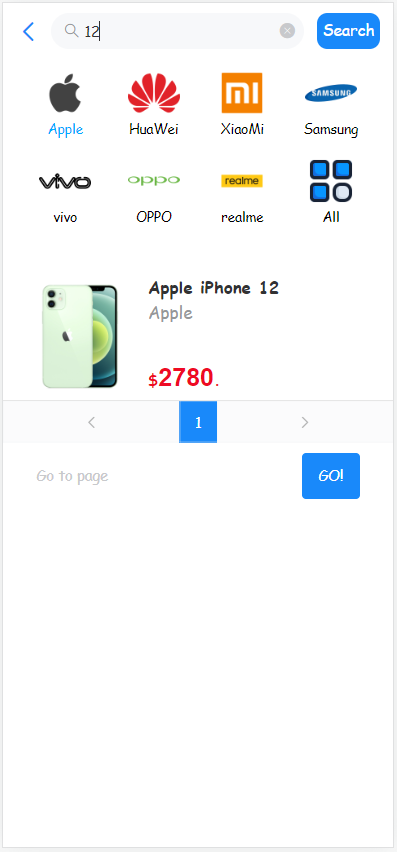
If the customer wants to search for a specific product, he could just tap the icon on the top of the product list page and then input the corresponding name in the search engine. If he wants to search for iPhone 12, he could first filter the brand “Apple” (the font color of the filtered brand will be blue) and simply type “12”. The result will show iPhone 12 exactly. But if he accidentally inputs “16” instead, the application will consider it as a fuzzy search because the system lacks the product whose name contains “16”. It will generate a list of products that it thinks might conform to the desire of the customer.

Figure: Search for iPhone 12 (left) and fuzzy search (right)

As we know all the products will be shown on the product list page. But the space is limited, which means we could not display all the products on one page. Therefore, we design pagination to tackle this problem. We could turn to the next page by simply tabbing “>” or return to the previous page by tabbing “<”. If we want to turn to a specific page, we just tab the identifier of that page or simply input the number in the box and tab “Go!”. If the customer inputs a number that is greater than the total number of pages, the system will just guide him to the last page.

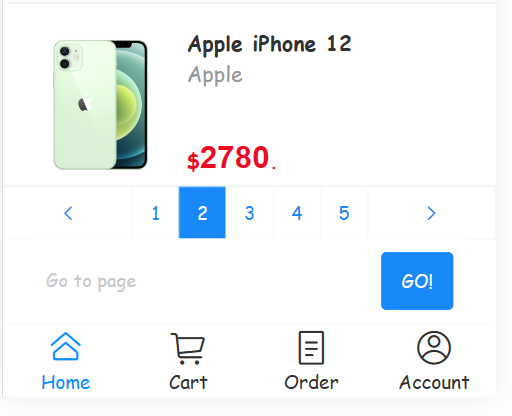


Fig x. Product list page pagination

### **Customer: Product detail page**

If the customer does not log in to the application, he will receive a reminder message which asks him to log in when he enters the product detail page. And if he just wants to directly buy this product or just simply add it to the shopping cart, the application will automatically lead him to the login page. Note that the biggest difference between the product detail page when the customer has logged in and the product detail page when the customer does not log in is the existence of an icon to determine whether the customer like this product.

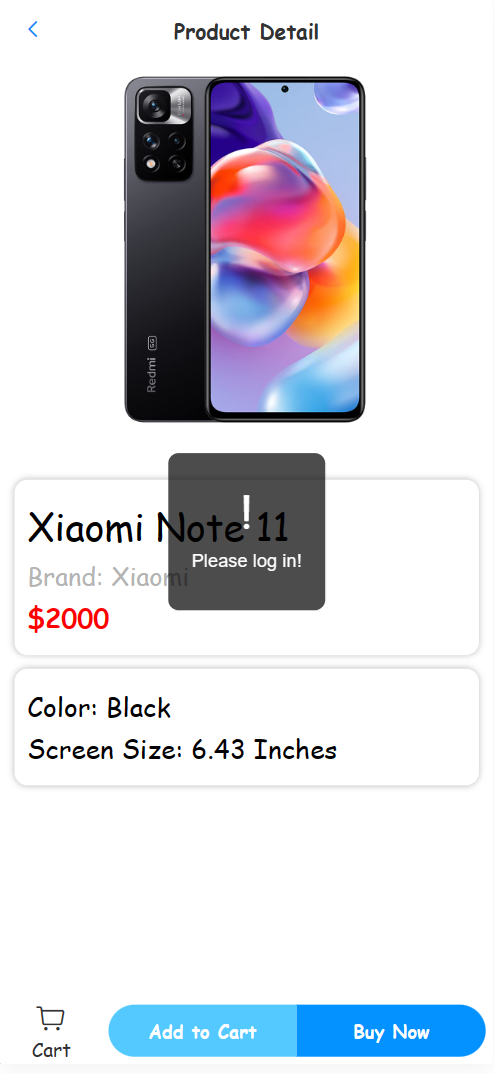
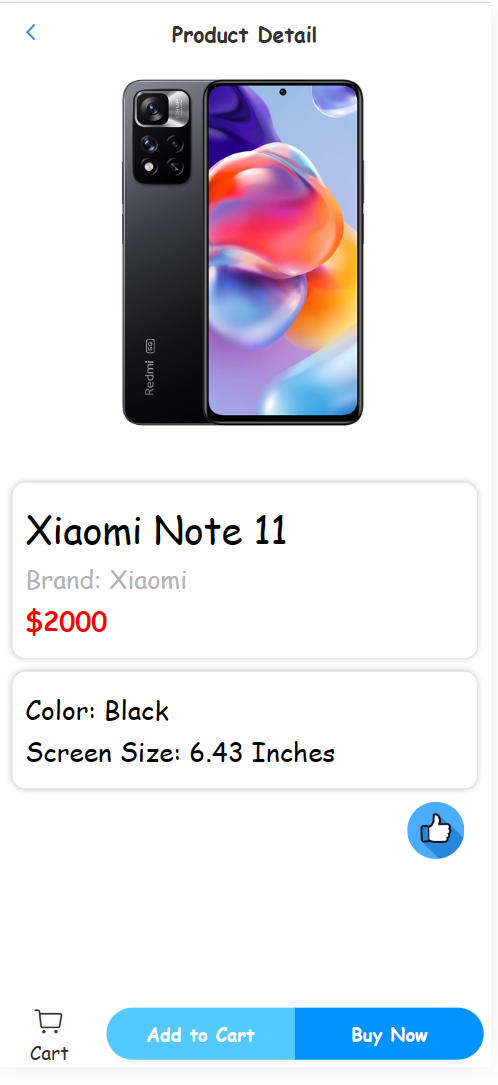


Figure: Product detail page where the user  
is logged in (left) or not logged in (right)

### **Customer: Shopping Cart Page**

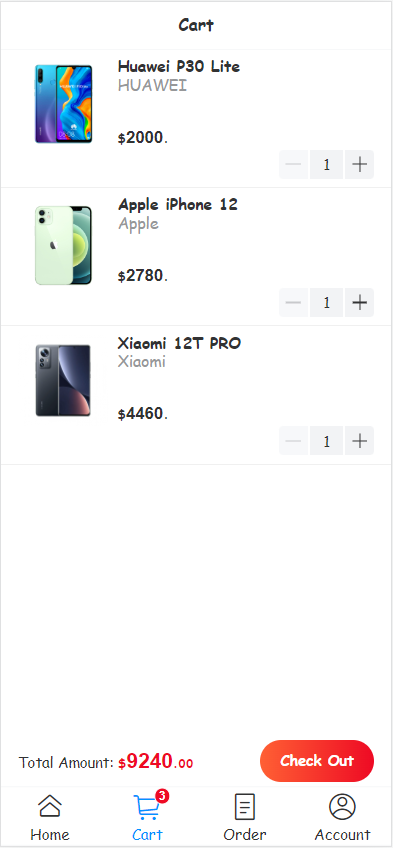
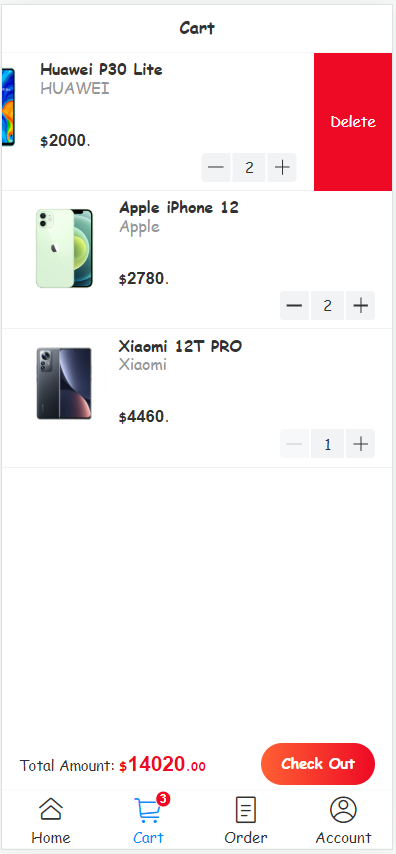
After the customer adds products to the shopping cart, this page will show the quantity and unit price of each product and the total amount of all the products in that shopping cart. For all products added to the shopping cart, their quantity defaults to 1. The customer can change the quantity of each product or remove them from the shopping cart. By the way, if there are products in the shopping cart, the upper right of the shopping cart icon in the footer will display how many products there are instead of the sum of the total number of each product. And the button “Check Out” allows the customer to turn into the final stage of shopping which will construct order in the end.

Figure: Shopping cart page

### **Customer: Order generation page**

After clicking the “Check Out” button, the customer will be navigated to the order generation page. On this page, customers can click the “edit” icon to change their shipping address. If nothing needs to be changed, they can just click the “Confirm” button to confirm the order. After confirming the order, it will automatically jump to the order details page.

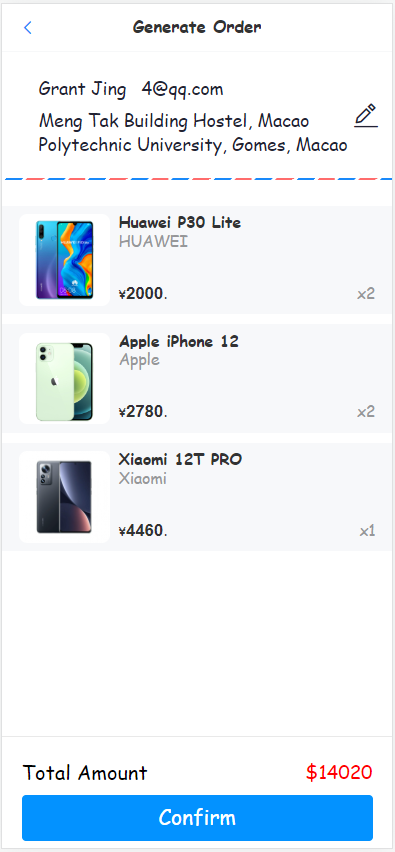


Figure: Confirm page

### **Customer: Order List Page with filtering**

On this page, it will present the list in chronological sequence. Customers could filter the orders between “All Orders”, “Current Purchases” and “Past Purchases”.

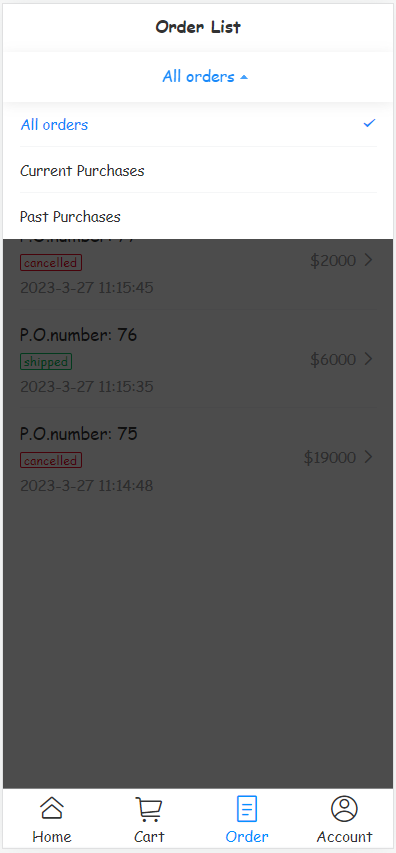


Fig x: Order list page and filtering

### **Customer: Order Detail Page**

On this page, three sections are displayed. They are order information, customer information, and product information respectively. The order status and purchase date will be displayed in the order information. In addition, after canceling the order, the order status will be changed and it will display the cancel date and who canceled it. Moreover, in the product information, the subtotal of each product will also be displayed.

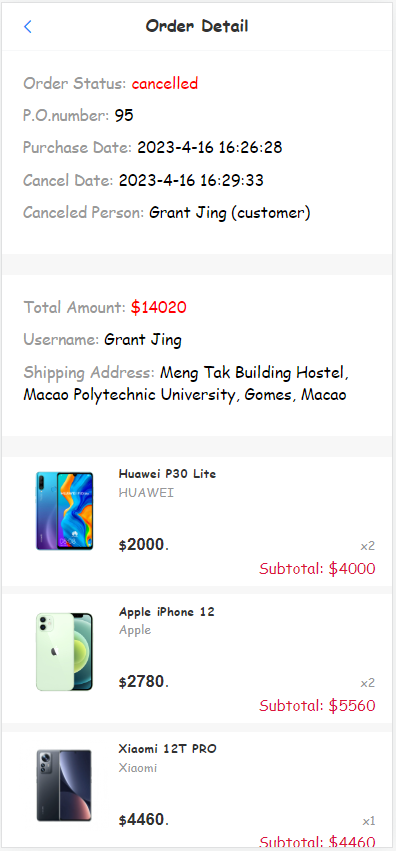
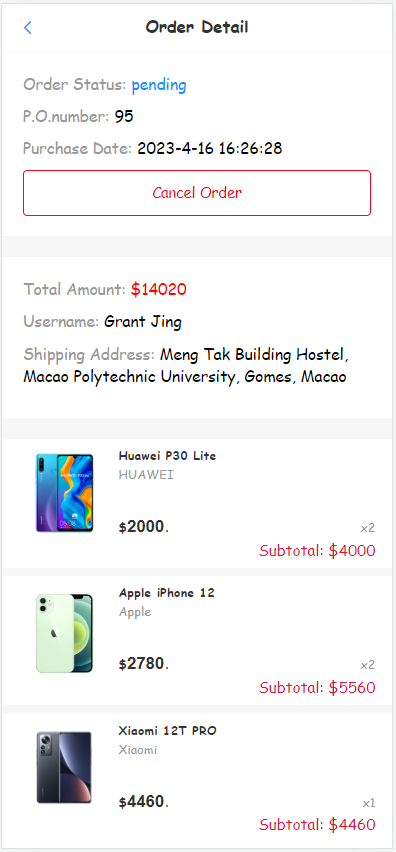


Figure: Order detail page

### **Customer: Account Page**

The username and e-mail of the current user will be displayed on the user account page. Users can perform corresponding operations by clicking the three options below the user card. For example, users can click "Account Management" to modify their username and password. And they can also click the eye icon to the right of the password input box to change the password's visibility.

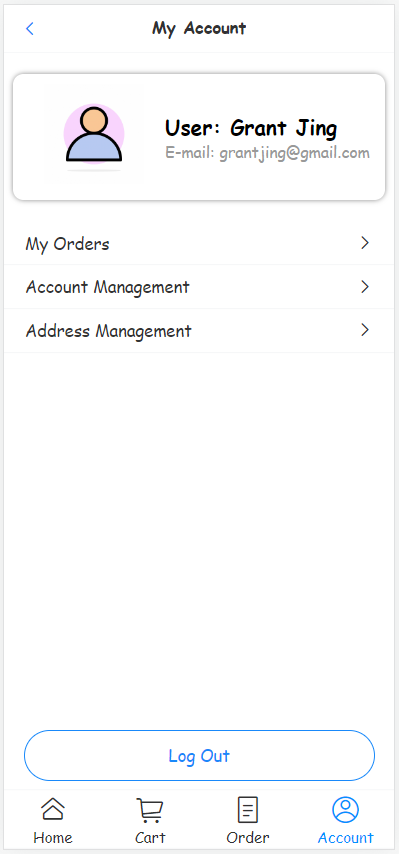
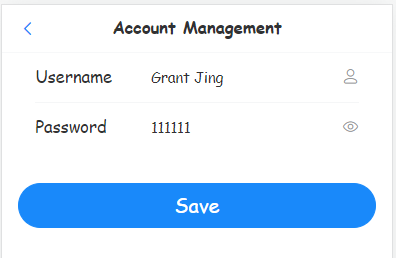
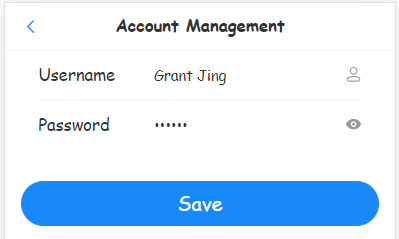


Figure: Account page and Account management page

### **Customer: Login Page and Signup Page**

On the Login page and signup page, if the user enters the wrong email format, an error message with the words "Please enter valid email" will appear at the bottom of the input box. If the user enters a password with less than 6 characters, an error message with the words “Password must be at least 6 characters” will appear. If the user does not enter anything, an error message "Please enter \*\*\*" will appear below the input box.

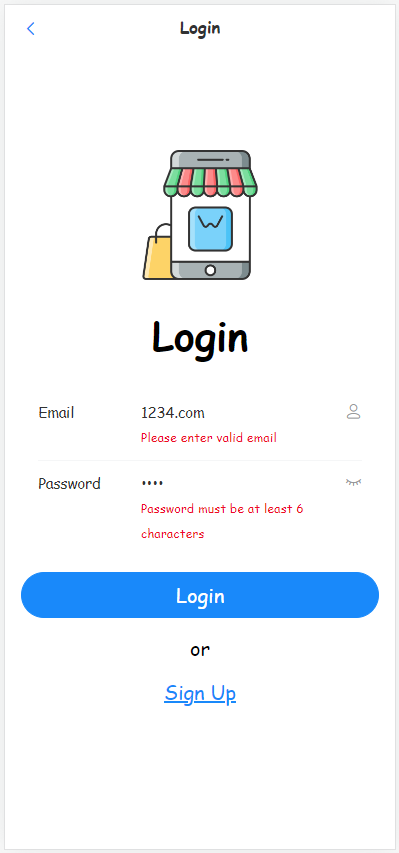
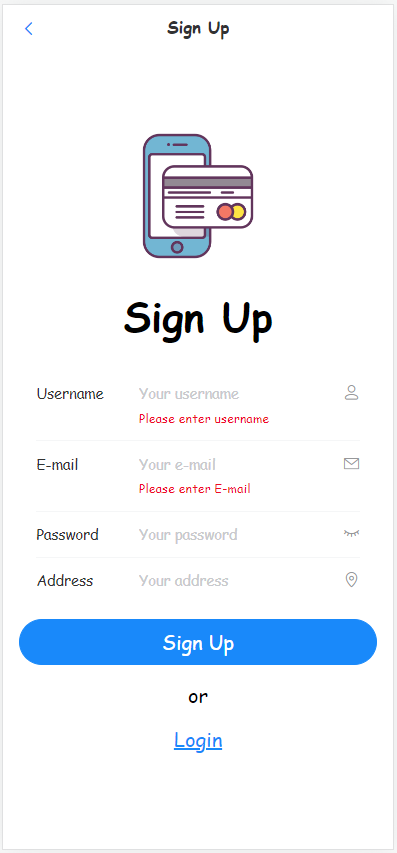


Figure: Login page (left) and Sign up page (right)

### **Vendor: Product List Page with searching and adding a new product**

Compared to the product list page for the customer, the biggest differences are a new way to search for a product and the port to add a new product into the system. For searching, the vendor could directly input the productId to search for the corresponding product (Just simply input “id:xx” in the search engine). And if the vendor wants to add a new product to the system, he could directly tap the button “Add a new product” and input all the related attributes of that product. After submitting, the system will automatically generate a productId for the product.

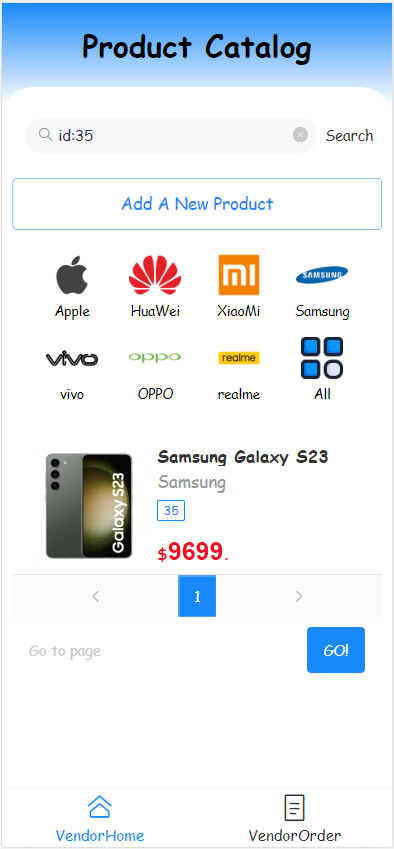
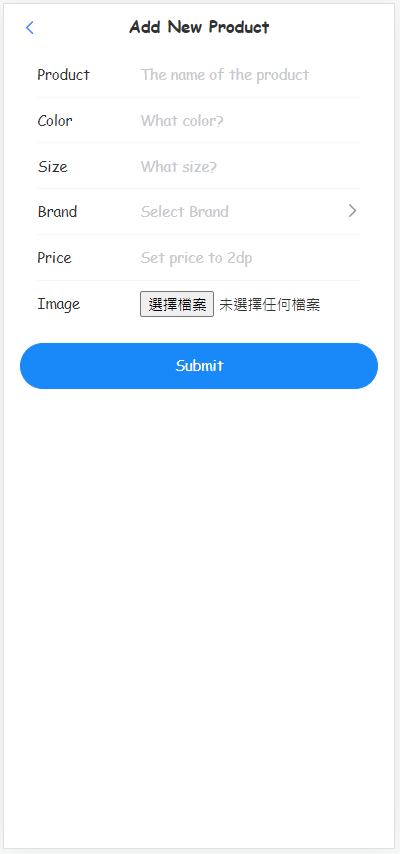


Figure: Vendor product list page (left) and add new product page (right)

### **Vendor: Order List Page**

On the order list page, the vendor could directly see which order belongs to the specific customer or input the specific purchase order number to get the corresponding order. Meanwhile, he can filter the system’s orders based on “All Orders”, “Pending Orders”, “Orders on Hold” and “Past Orders”.

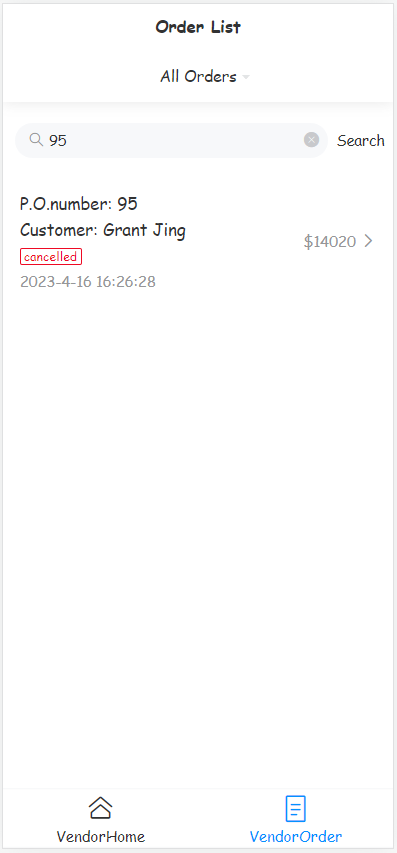
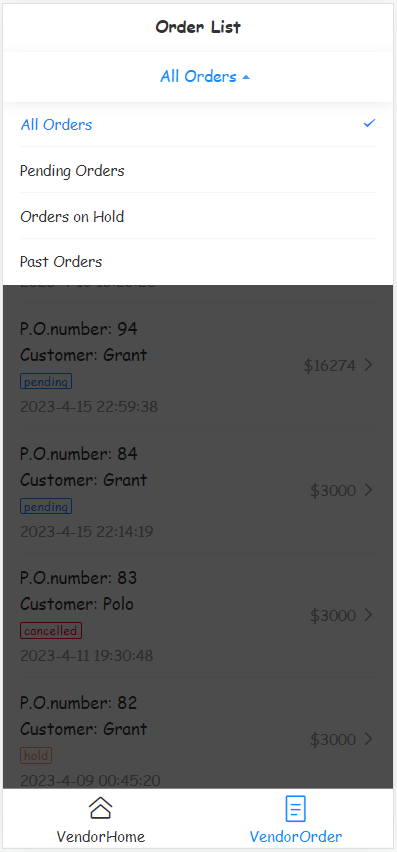


Figure: Vendor order list page

### **Vendor: Order Detail Page**

Compared to the order detail page for customers, the vendor has the right to hold, ship, and cancel the order. After choosing the specific status, the order detail will record the change time correspondingly.

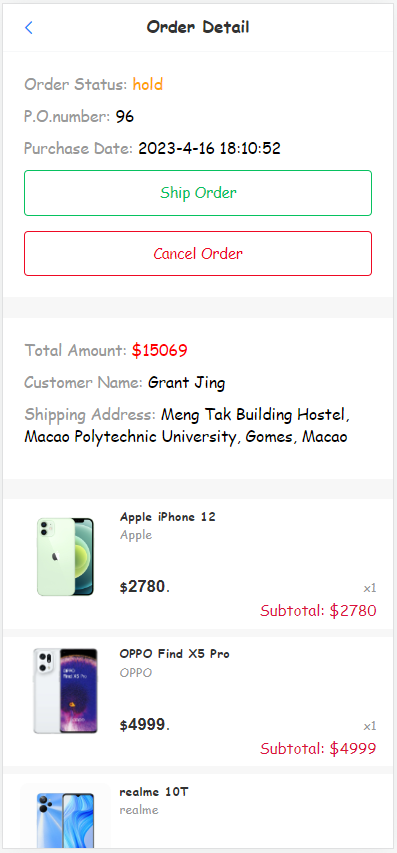
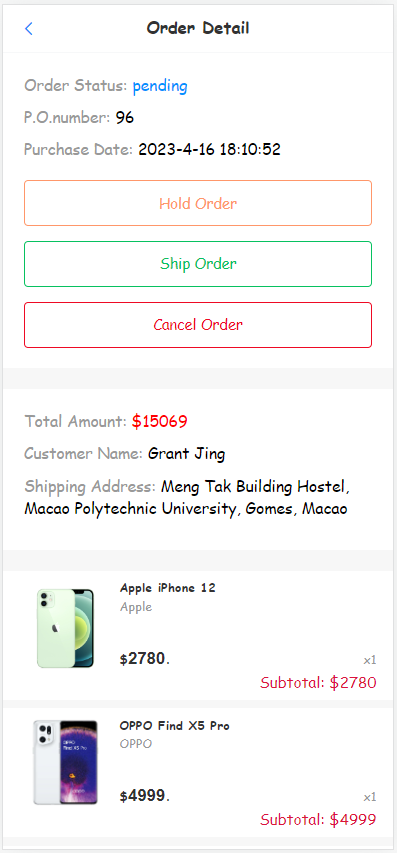


Figure: Vendor order detail page

## Testing and System Evaluations

To evaluate the effectiveness of the recommendation algorithm, we have designed and conducted a simple test case based on preferences on brands.

Firstly, we create 2 new accounts as User A and User B. We assume that User A likes Huawei. Oppositely, User B dislikes Huawei and like Apple. They both like Xiaomi.

Figure 1, 2 show the top 3 products recommended to Tester 1, 2. Tester 1 disliked some Huawei, and in the recommendation generated next time has become all iPhones. Meanwhile, Tester 2 disliked some iPhones, and the brand of 3 recommendation products are consistent with the desired brand: Huawei. According to the brands of the products, we summarize the outcomes in Table 1.

In conclusion, after conducting the test case, we noticed that our recommendation algorithm is sufficiently effective to find out the similar users of a new user and recommend the preferences of them to the new user.

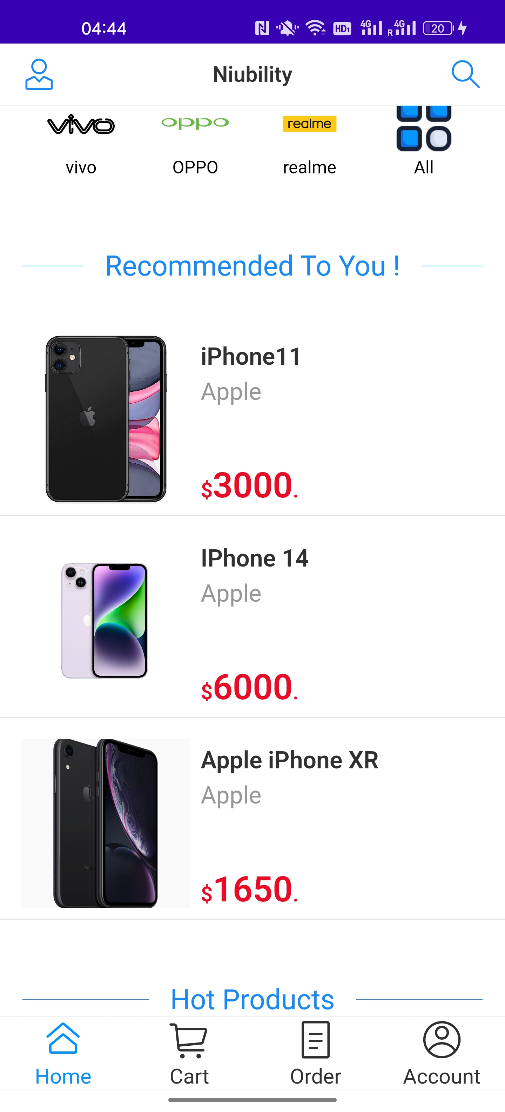
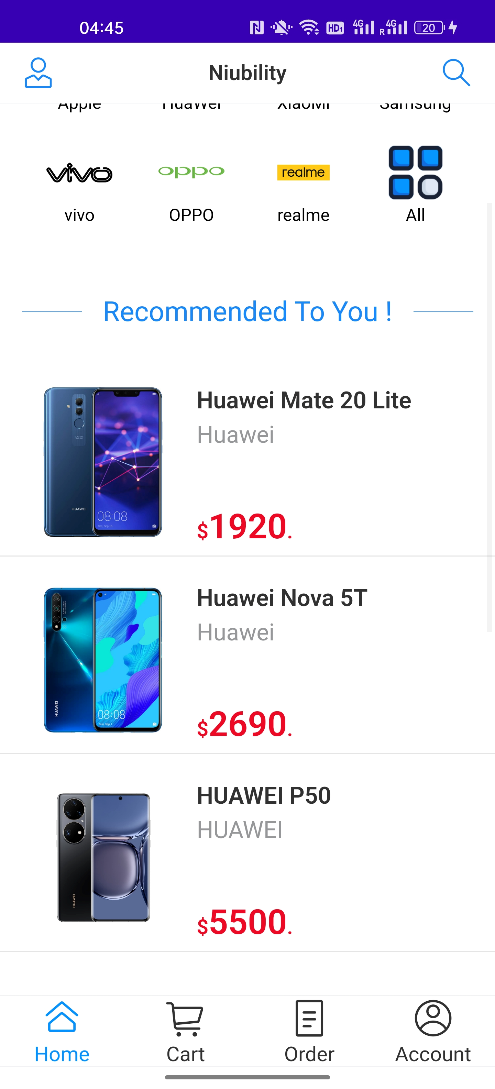


Figure1: Recommendation of Tester1 and Tester2



## 

Table1: Outcomes

## Testing and System Evaluations

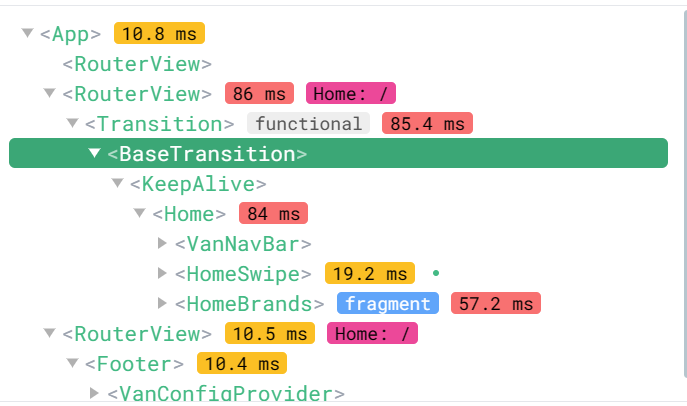
Because our e-commerce platform is a web-mobile application. So, in order to guarantee the performance on mobile phones, we first test its performance on the computer based on its load speed.

First, I want to talk about the reason why we choose load speed as testing metric. It is normal that the longer a webpage takes to load, the more its bounce rate will skyrocket. A high bounce rate will inform the search engine that this website has useless content. Therefore, the ranking will correspondingly decline. If e-commerce website loads check out page slowly, the customer might give up paying money. This website may fall behind the competitors inevitably.

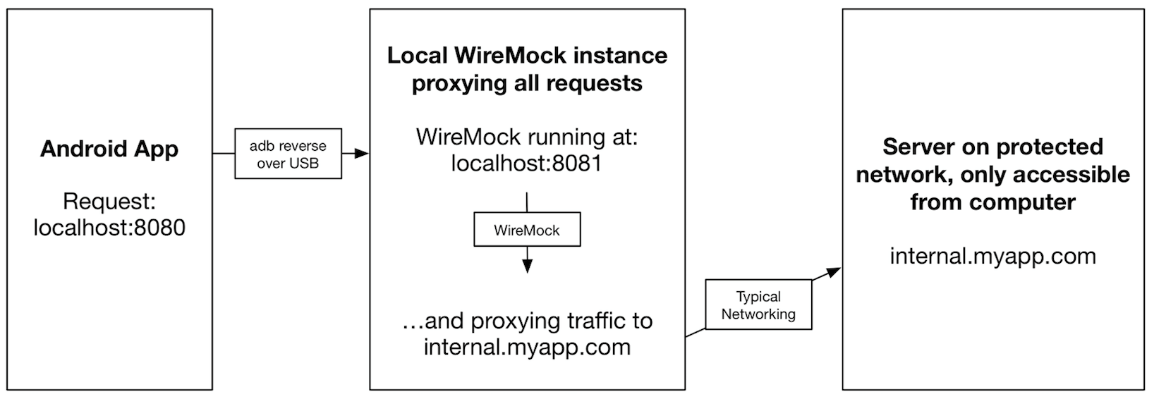
So how fast should a website load? Based on the information we gained: “Ideally, you’ll want your website to load within three seconds, or two seconds if it’s an ecommerce site. The two-to-three second mark is the turning point where bounce rates skyrocket – in fact, 40% of consumers will wait no more than three seconds before abandoning a site.” We use two seconds as a criterion to judge whether our mobile application is user-friendly or not.

After testing, we found out the time all the pages of our application take to load is far less than one seconds. This indicates in the metric of load speed, our mobile application is quite brilliant.

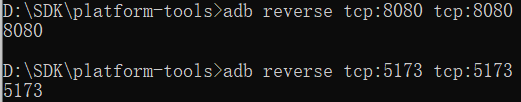
[Website Load Time Statistics: Why Speed Matters in 2023 (websitebuilderexpert.com)](https://www.websitebuilderexpert.com/building-websites/website-load-time-statistics/)



**Debug On a Real Android Phone**

Android Debug Bridge (adb) is a versatile command-line tool that enables communications with devices.

*Figure 1. AndroidAndroid Debug Bridge (ADB)*



*Figure 2. ADB Commands*

By using the adb command, it allows the mobile phone to access host 127.0.0.1 (loopback address), in the same way that we debug in the PC browser. The API server running on port 8080 should also be bridged to allow data communication between the mobile app and the API server. Moreover, we are able to test our mobile App by simply accessing 127.0.0.1/5173 without any deployment

*Figure 3. The vue.js accesses API server running on port 8080.*