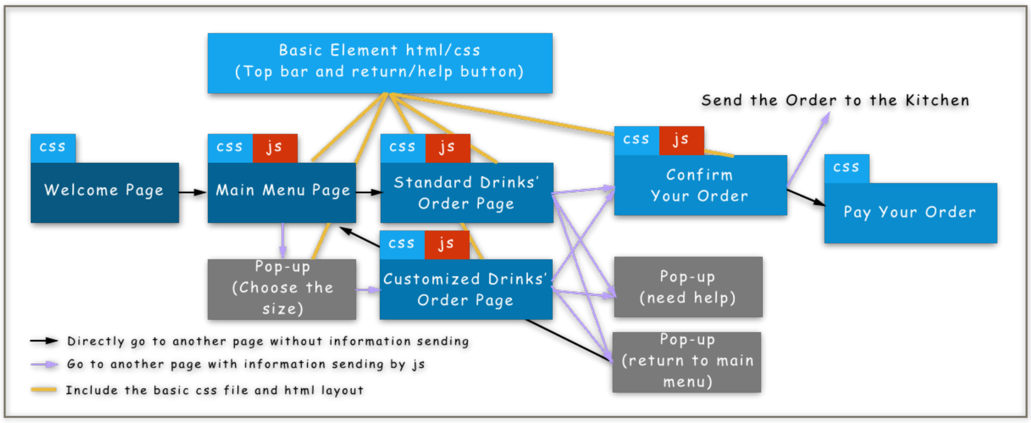
Shuyang Du

Reflection Report

Code:

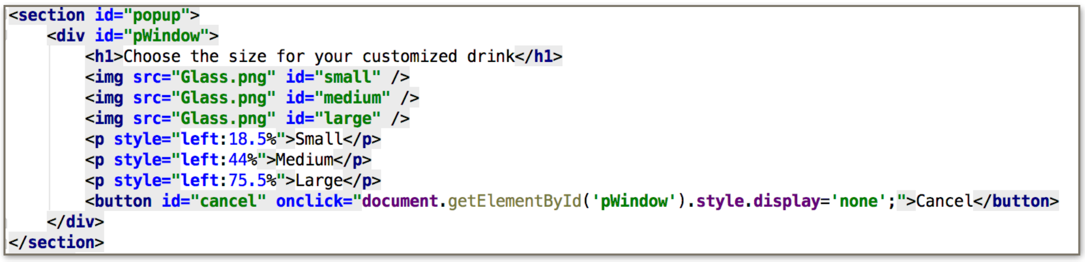
In our group project, we divided the coding part into 3 different parts. They are kiosk part, kitchen part, mobile part. I was responsible for the kiosk part, which are the interfaces showed on the screen of the order machine for the customers of the juice bar.

Here is the structure of the code in kiosk part:


Basic Element html/css
(Top bar and return/help button)
Send the Order to the Kitchen
css
Welcome Page
css js
Main Menu Page
css js
Standard Drinks' Order Page
css js
Confirm Your Order
css
Pay Your Order
Pop-up (Choose the size)
css js
Customized Drinks' Order Page
Pop-up (need help)
Pop-up (return to main menu)
Directly go to another page without information sending
Go to another page with information sending by js
Include the basic css file and html layout

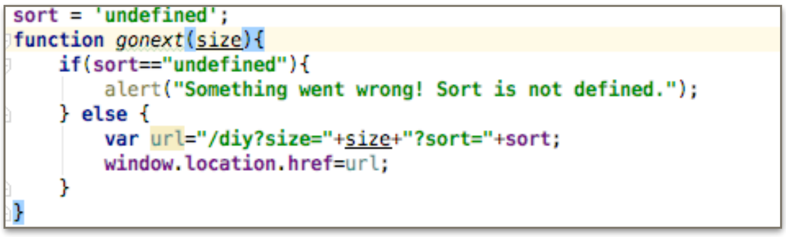
This diagram shows how we transform the standard usage of the system to the code. Next, I will introduce a few part of our code as an example to explain how the code works to finish the sequence.

A. This example shows how to write a popup and make it works. This is the part which shows the further information of a juice and lists the different size of it for the customer.



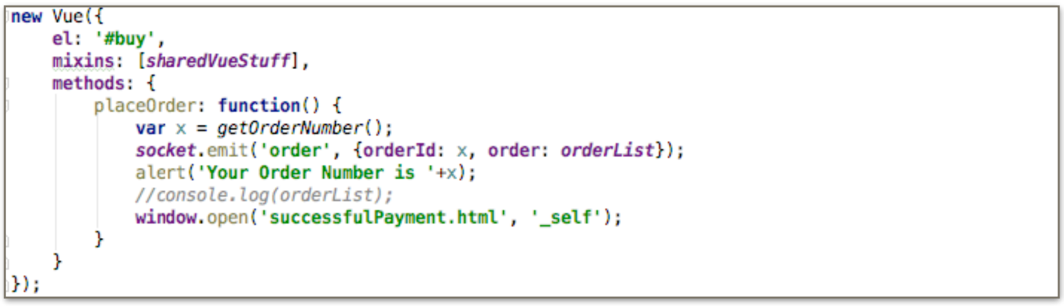
```
<section id="popup">
    <div id="pWindow">
        <h1>Choose the size for your customized drink</h1>
        <img src="Glass.png" id="small" />
        <img src="Glass.png" id="medium" />
        <img src="Glass.png" id="large" />
        <p style="left:18.5%">Small</p>
        <p style="left:44%">Medium</p>
        <p style="left:75.5%">Large</p>
        <button id="cancel" onclick="document.getElementById('pWindow').style.display='none';">Cancel</button>
    </div>
</section>
```

B. This example shows how to jump from the menu page to the DIY page. It includes a select part to judge which url should be used based on what the user have chosen(smoothie or juice).



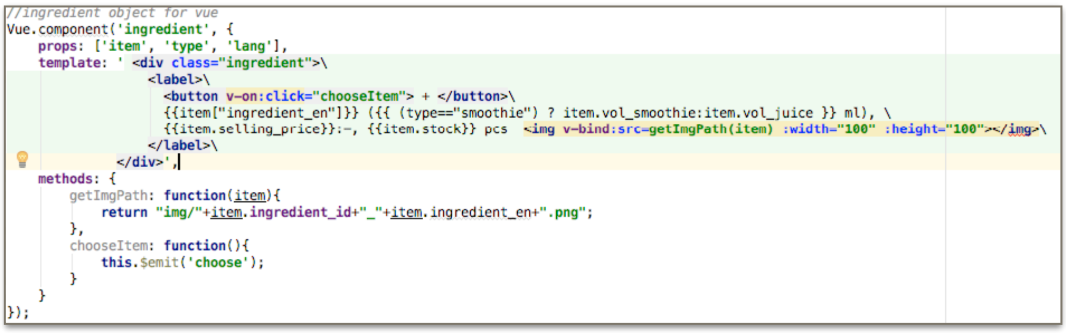
```
sort = 'undefined';
function gonext(size){
    if(sort=="undefined"){
        alert("Something went wrong! Sort is not defined.");
    } else {
        var url="/diy?size="+size+"?sort="+sort;
        window.location.href=url;
    }
}
```

C. This example shows how to send the order to the kitchen in both DIY order page and standard order page, this is actually an interface of Vue.



```
new Vue({
    el: '#buy',
    mixins: [sharedVueStuff],
    methods: {
        placeOrder: function() {
            var x = getOrderNumber();
            socket.emit('order', {orderId: x, order: orderList});
            alert('Your Order Number is '+x);
            //console.log(orderList);
            window.open('successfulPayment.html', '_self');
        }
    }
});
```

D. This example shows how to read the ingredients' information in the DIY order page and show it in the list. However, it is also where we find the bug when we do the integration, because we name the different variable with the same name at first. I think we need to use Github correctly next time to avoid this type of error.



```
//ingredient object for vue
Vue.component('ingredient', {
    props: ['item', 'type', 'lang'],
    template: ' <div class="ingredient">\
                    <label>\
                        <button v-on:click="chooseItem"> + </button>\
                        {{item["ingredient_en"]}} ({{ (type=="smoothie") ? item.vol_smoothie:item.vol_juice }} ml), \
                        {{item.selling_price}}:-, {{item.stock}} pcs  <img v-bind:src=getImgPath(item) :width="100" :height="100"></img>\
                    </label>\
                </div>',
    methods: {
        getImgPath: function(item){
            return "img/"+item.ingredient_id+"_"+item.ingredient_en+".png";
        },
        chooseItem: function(){
            this.$emit('choose');
        }
    }
});
```

What I learn from the coding part:

1. Github should be used correctly, since we didn't fully use its functions so we had a lot of problems when we started our integration and it significantly slow down our process.
2. We should write and share the document more frequently in order to know what have been done and what need to be done, so that everyone could concentrate on his own works.
3. The standard of the name rule should be discussed and we need to have an agreement before we started coding.
4. The whole structure should be drawn as a diagram and shown to everyone in the group to make sure that all can fully understand what we need to do, how much workload we have and how difficult it is.

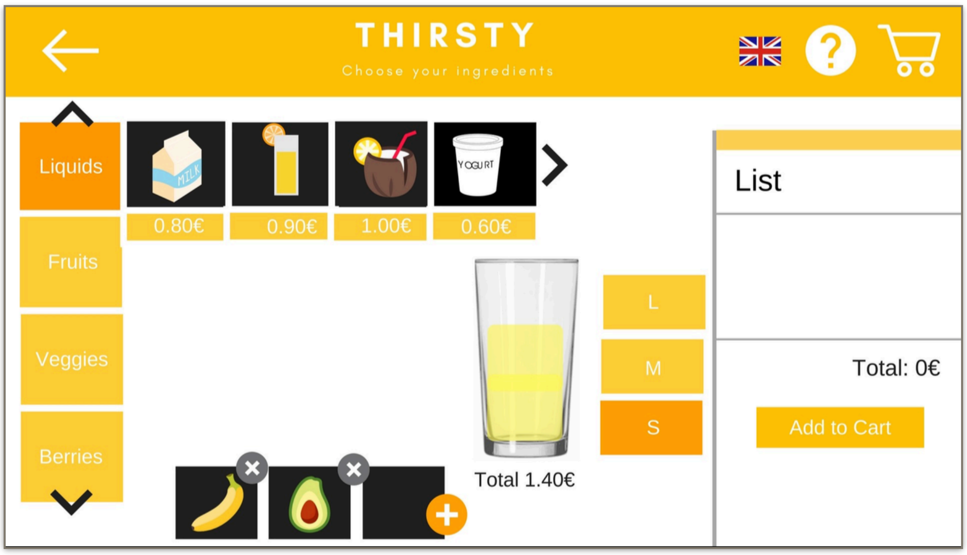
Interaction Design:

During the design of the user interface of our system, I introduced Jakob Nielsen's 10 general principles for interaction design, which are the golden rules used by the interaction designers all over the world for more than 20 years. Now I am going to use an example in our system to show you how these principles affect our project. (Because of the limit of the characters, I only choose 3 principles to exemplify)

1. Visibility of system status:

The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.

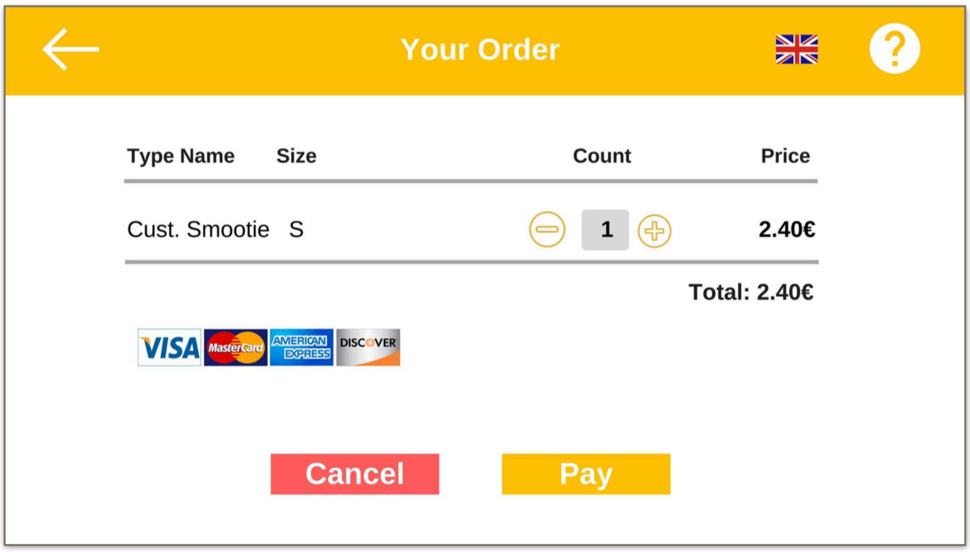
In our system, when you choose to customize your juice or smoothie, you can directly see what you just added in the drink and how many space left for you to add new ingredients.



2. User control and freedom

Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

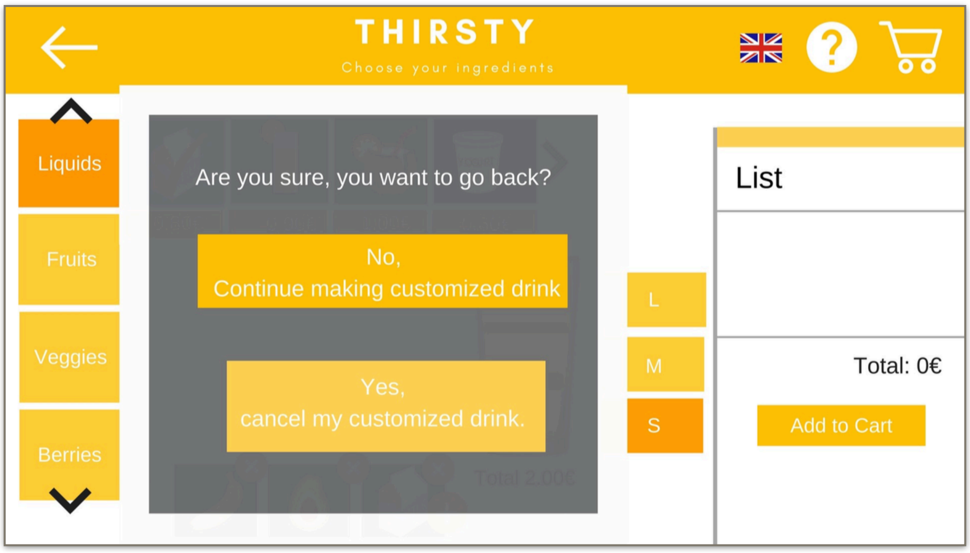
Consider of this principle, I added the return button in every pages, which could lead you to the last page you have just visited. it is locate on the left side of the top bar. Besides, there are also cancel buttons in some pages, which could help you undo or redo the order.



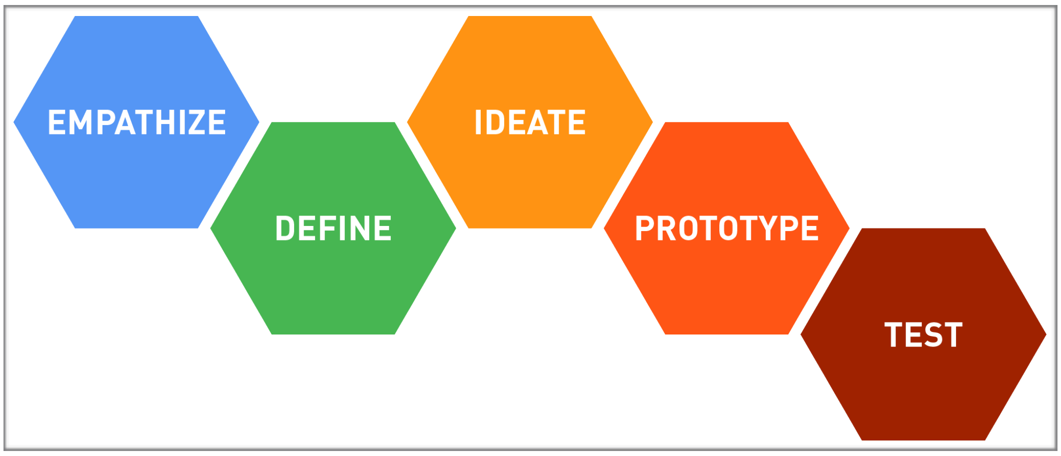
3. Help users recognize, diagnose, and recover from errors

Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

After the observation of the ordering system in MAX restaurant, I tried to introduce a pop-up to our own system in order to help the user recover from the accidentally wrong operation. Like this, when the user press the return button, he will lose all the ingredient he have chose for his drink, it is really necessary to ask him to think twice before he make the decision to redo the order.


THIRSTY
Choose your ingredients
Liquids
Fruits
Veggies
Berries
Are you sure, you want to go back?
No,
Continue making customized drink
Yes,
cancel my customized drink.
L
M
S
List
Total: 0€
Add to Cart

Usability:



In order to make sure the usability in our system, I introduce the process diagram of design thinking to our project. It has 5 steps, which are empathize, define, ideate, prototype and test.

Empathize: Before we do the design of the system, I put myself as a user not a designer and go to use the order machine in Mc Donald and MAX, tried to figure out what matters to the user? What are they really care about?

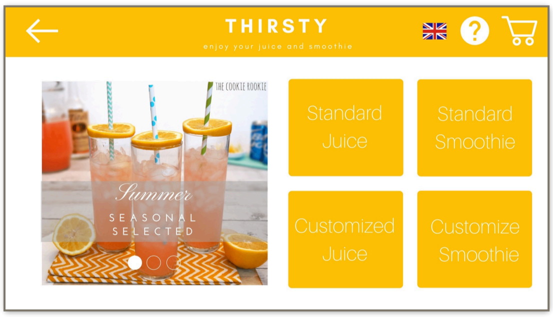
Define: In this step, we need to combine the points collected in the former step(empathize) and the requirements of our own system(juice bar) together, in order to define what function does our system really need?

Ideate: Now we are going to brainstorm how to make our system come true. We need to consider not only the functional requirements concluded in the last step, but also some non-functional requirements like user friendly, error prevention and flexibility.

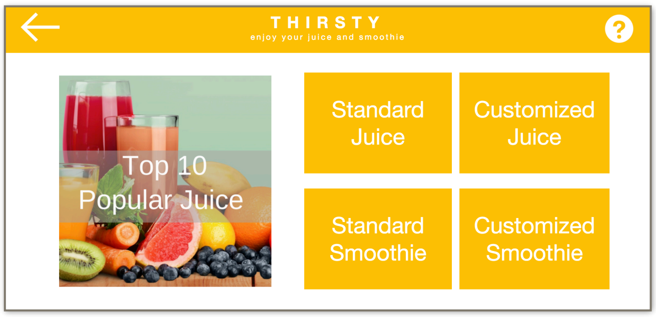
Prototype: It's time for us to start our work with the mock up, which is the prototype of our system, the slides that show the sequence of the standard usage of the system.

Test: After finished the mock up prototype, we need to use it to do the usability test, which is an important part of the whole design process. We invited our teachers, our friends in other groups and some volunteers to join the test. Through this test we can collect the feedback and find out what works well? What didn't? Then we could make some improvement in the coding part later.

Here is an example that shows the change after collecting then feedback in the usability test.



Before



After

According to the feedback, we increase the font weight to make the user recognize the name of the button more easily. We also remove the cart button on the right of the top bar, because our tester think it is a redundancy. So we just use a shopping list to replace it in the ordering page, in order to simplify the sequence of actions and make the appearance of the interface better.

However, there is still a limitation of our usability test. All of our testers are adult and the older person is younger than 50 years old. So what will happen when a child or an old man used our system? We can not predict because of the lack of the feedbacks of these people in the test. To solve this problem, I think I can only do some improvement after collect some feedback from those people when the system is already used.

Reference:

1. *"The Design of Everyday Things".* By Norman, Donald A

2. *"About Face: The Essentials of Interaction Design, 4th Edition".* By Christopher Noessel, Alan Cooper, Robert Reimann, David Cronin

3. *10 Usability Heuristics for User Interface Design* By Jakob Nelson on January 1st, 1995

4. The theory of design thinking is introduced from d.school in Stanford University.