

index:

```

1  <!DOCTYPE html>
2  <html lang="en">
3
4  <head>
5      <meta charset="UTF-8">
6      <meta http-equiv="X-UA-Compatible" content="IE=edge">
7      <meta name="viewport" content="width=device-width, initial-scale=1.0">
8      <link rel="stylesheet" href="./styles/style.css">
9      <link rel="stylesheet" href="./styles/modal-box.css">
10     <title>Product list</title>
11 </head>
12
13 <body>
14     <div class="toggle-switch">
15         <p class="dark-mode">&#x1f319;</p>
16         <span>
17             <label class="switch">
18                 <input id="theme-switch" type="checkbox">
19                 <span class="slider round"></span>
20             </label>
21         </span>
22         <p class="light-mode">&#x1f506;</p>
23     </div>
24     <div class="search">
25         <div class="icon">
26             <span>&#9906;</span>
27         </div>
28         <input type="text" placeholder="Search" id="search">
29     </div>
30     <div class="shopping-cart">
31         <div id="shopping-cart" class="icon" style="background-color: █ darkcyan; font-size: 1em;">
32             <span style="transform: rotate(0);">&#128722;</span>
33             <span id="total" style="transform: rotate(0);"><b>0</b></span>
34         </div>
35     </div>
36     <div id="modal-box" class="modal">
37         <div class="modal-content">
38             <div class="modal-header">
39                 <span class="close">&times;</span>
40                 <p><b>Net price: </b><span id="net-price">0</span></p>
41             </div>
42             <div class="modal-body"></div>
43         </div>
44     </div>
45     <script src="./modules/script.js" type="module"></script>
46     <script src="./modules/events.js" type="module"></script>
47     <script src="./modules/theme.js" type="module"></script>
48     <script src="./modules/shoppingCart/shoppingCart.js" type="module"></script>
49     <script src="./modules/loadStorage.js" type="module"></script>
50 </body>
51
52 </html>
53

```

monitor.js:

```
1  export default class Monitor {
2      constructor(code = 0, name = "unknown", size = 0, resolution = "1440x1080") {
3          this._code = code;
4          this._name = name;
5          this._sizeInInches = size;
6          this._stock = 0;
7          this._price = 0;
8          this._resolution = resolution;
9          this._imgURL = "path";
10     }
11
12     get name() {
13         return this._name;
14     }
15
16     get resolution() {
17         return this._resolution;
18     }
19
20     get code() {
21         return this._code;
22     }
23
24     get imgURL() {
25         return this._imgURL;
26     }
27
28     get size() {
29         return this._sizeInInches;
30     }
31
32     get price() {
33         return this._price;
34     }
35
36     get stock() {
37         return this._stock;
38     }
39
40     /**
41      * @param {string} path set a new path
42      */
43     set imgURL(path) {
44         this._imgURL = path;
45     }
46 }
```

```
47     /**
48      * @param {number} amount set a new amount
49      */
50     set stock(amount) {
51         this._stock += amount;
52     }
53
54     /**
55      * @param {number} size set a new size
56      */
57     set size(size) {
58         this._sizeInInches = size;
59     }
60
61     /**
62      * @param {string} name set a new name
63      */
64     set name(name) {
65         this._name = name;
66     }
67
68     /**
69      * @param {string} code set a new code
70      */
71     set code(code) {
72         this._code = code;
73     }
74
75     /**
76      * @param {string} resolution set a new resolution
77      */
78     set resolution(resolution) {
79         this._resolution = resolution;
80     }
81
82     /**
83      * @param {number} size set a new size
84      */
85     set size(size) {
86         this._sizeInInches = size;
87     }
88
89     /**
90      * @param {number} price set a new price
91      */
92     set price(price) {
93         this._price = price;
94     }
95 }
```

products.js:

```
4  import Monitor from "./monitor.js";
5
6  const products = [];
7
8  products.push(new Monitor(1, "BENQ ZOWIE XL2731", 27, "1920x1080"));
9  products[0].price = 9900;
10 products[0].stock = 0;
11 products[0].imgURL = "./images/ZOWIE-XL2731.jpg";
12
13 products.push(new Monitor(2, "BENQ MOBIUZ EX3415R", 34, "3440x1440"));
14 products[1].price = 35900;
15 products[1].stock = 59;
16 products[1].imgURL = "./images/MOBIUZ-EX3415R.jpg";
17
18 products.push(new Monitor(3, "BENQ MOBIUZ EX2710R", 27, "2560x1440"));
19 products[2].price = 21900;
20 products[2].stock = 104;
21 products[2].imgURL = "./images/MOBIUZ-EX2710R.jpg";
22
23 products.push(new Monitor(4, "BENQ EW2780Q", 27, "2560x1440"));
24 products[3].price = 9500;
25 products[3].stock = 156;
26 products[3].imgURL = "./images/EW2780Q.jpg";
27
28 products.push(new Monitor(5, "BENQ EW2480", 24, "1920x1080"));
29 products[4].price = 5500;
30 products[4].stock = 222;
31 products[4].imgURL = "./images/EW2480.jpg";
32
33 products.push(new Monitor(6, "BENQ EX3203R", 31.5, "2560x1440"));
34 products[5].price = 16900;
35 products[5].stock = 25;
36 products[5].imgURL = "./images/EX3203R.jpg";
37
38 products.push(new Monitor(7, "BENQ ZOWIE XL2546", 24.5, "1920x1080"));
39 products[6].price = 15900;
40 products[6].stock = 79;
41 products[6].imgURL = "./images/ZOWIE-XL2546.jpg";
42
43 products.push(new Monitor(8, "BENQ GW2780", 27, "1920x1080"));
44 products[7].price = 5900;
45 products[7].stock = 0;
46 products[7].imgURL = "./images/GW2780.jpg";
47
48 export { products as default };
49
```

script.js:

```

3  import products from "../products/products.js";
4  import * as lib from "../shoppingCart/cart.js";
5
6  // code here
7  let mainBody = document.querySelector("body");
8  mainBody.id = "product-list";
9  mainBody.className = "p-6 justify-center";
10
11 // add h1 to mainBody
12 let title = document.createElement("h1");
13 title.className = "font-bold text-xl text-center";
14 title.textContent = "Monitor List";
15 mainBody.appendChild(title);
16
17 let divContainer = document.createElement("div");
18 divContainer.className = "product-container grid grid-cols-5";
19 divContainer.id = "store-product";
20
21 // add container to mainBody
22 let container = mainBody.appendChild(divContainer);
23
24 for (const product of products) {
25   // add div to container in mainBody
26   let divList = document.createElement("div");
27   divList.id = product.code;
28   divList.className =
29     "flex flex-col m-5 bg-white rounded-lg transition duration-200 ease-in-out hover:bg-purple-200 shadow-xl transform hover:scale-110 product";
30   divList.setAttribute("name", product.name + " " + product.size + "");
31   let divItem = container.appendChild(divList);
32
33   // add img to div in container
34   let img = document.createElement("img");
35   img.src = product.imgURL;
36   img.alt = product.name;
37   img.style = "width: 100%; height: 100%;";
38   img.className = "p-2";
39   divItem.appendChild(img);
40
41   // add details to div in container
42   let divDetails = document.createElement("div");
43   divDetails.className = "details p-2";
44
45   let name = document.createElement("div");
46   name.className = "p-1 bg-green-100 flex-1 m-1";
47   name.innerHTML = `<b>Product: </b>${product.name} ${product.size}`;
48   divDetails.appendChild(name);
49
50   let price = document.createElement("div");
51   price.className = "m-1 p-1";
52   price.innerHTML = `<b>Price: </b>${(new Intl.NumberFormat("th-TH", {
53     style: "currency",
54     currency: "THB",
55   })).format(product.price)}`;
56   divDetails.appendChild(price);
57 }

```

```

58     let res = document.createElement("div");
59     res.className = "m-1 p-1";
60     res.innerHTML = `<b>Resolution: </b>${product.resolution}`;
61     divDetails.appendChild(res);
62
63     let stock = document.createElement("div");
64     // check if have in stock
65     stock.className = "m-1 p-1";
66     if (product.stock == 0) {
67         let outOfStock = document.createElement("b");
68         outOfStock.className = "text-red-500";
69         outOfStock.textContent = "Out of Stock";
70         stock.appendChild(outOfStock);
71     } else {
72         stock.innerHTML = `<b>In stock: </b>${product.stock}`;
73     }
74     divDetails.appendChild(stock);
75
76     // add all detail in item container
77     divItem.appendChild(divDetails);
78
79     // add "Add" button
80     let divBtn = document.createElement("div");
81     divBtn.className = "m-1 p-1";
82     let btn = document.createElement("button");
83     btn.className = `add-to-cart w-full p-2 rounded ${
84         product.stock == 0
85             ? "bg-gray-100 text-gray-300 pointer-events-none"
86             : "bg-purple-400 text-white transition duration-150 ease-in-out"
87     }`;
88     btn.innerHTML = "<b><i>Add</i></b>";
89
90     divBtn.appendChild(btn);
91     divDetails.appendChild(divBtn);
92 }
93
94 let totalQuantity = document.getElementById("total");
95 totalQuantity.innerHTML = `<b>${lib.getTotalQty()}</b>`;
96

```

events.js:

change toggle search bar to arrow function

```
1  import * as lib from "../shoppingCart/cart.js";
2  import products from "../products/products.js";
3
4  // toggle search bar
5  document.querySelector(".icon").addEventListener("click", () => {
6    document.querySelector("#search").classList.toggle("active");
7  });
8
9  // search function
10 const productStore = document.querySelectorAll(".product");
11 const search = document.getElementById("search");
12
13 search.addEventListener("keyup", (e) => {
14   const text = e.target.value.toLowerCase().trim();
15   for (let i = 0; i < productStore.length; i++) {
16     let productName = productStore[i].getAttribute("name");
17     if (productName.toLowerCase().trim().includes(text)) {
18       productStore[i].style.display = "";
19     } else {
20       productStore[i].style.display = "none";
21     }
22   }
23 });
24
25 // add to cart function
26 const addButton = document.querySelectorAll(".add-to-cart");
27
28 for (const [id, btn] of addButton.entries()) {
29   btn.addEventListener("click", () => {
30     lib.add(products[id].code);
31     let totalQuantity = document.getElementById("total");
32     totalQuantity.innerHTML = `<b>${lib.getTotalQty()}</b>`;
33   });
34 }
```

cart.js:

```

1  // store in Cookies
2  import products from "../products/products.js";
3  import CookieUtil from "../cookieUtil.js";
4
5  let shoppingCart = new Object();
6  export const cookieName = "cart";
7
8  /**
9   * @param {string} name name of cookie
10   */
11  export function getCookie(name) {
12      return CookieUtil.getCookie(name);
13  }
14
15  /**
16   * @param {string} name name of the cookie
17   * @param {string} value value of the cookie
18   * @param {number} expires duration of the cookie in days
19   */
20  function setCookie(name, value, expires) {
21      CookieUtil.setCookie(name, value, expires);
22  }
23
24  /**
25   * @returns {number} total quantity in cart
26   */
27  export function getTotalQty() {
28      if (getCookie(cookieName) === null) {
29          return 0;
30      } else {
31          shoppingCart = parseToObj(getCookie(cookieName));
32          return Object.values(shoppingCart).reduce(
33              (total, qty) => total + qty,
34              0
35          );
36      }
37  }
38
39  export function getNetPrice() {
40      if (getCookie(cookieName) === null) {
41          return 0;
42      } else {
43          shoppingCart = parseToMap(parseToObj(getCookie(cookieName)));
44          let total = 0;
45          shoppingCart.forEach((qty, code) => {
46              total +=
47                  products.find((product) => product.code == code).price * qty;
48          });
49          return total;
50      }
51  }

```



```

53  /**
54   * @param {string} id product code
55   */
56  export function add(id) {
57      if (getCookie(cookieName) === null) {
58          shoppingCart[id] = 1;
59      } else {
60          shoppingCart = parseToObj(getCookie(cookieName));
61          if (shoppingCart.hasOwnProperty(id)) {
62              shoppingCart[id] += 1;
63          } else {
64              shoppingCart[id] = 1;
65          }
66      }
67      setCookie(cookieName, JSON.stringify(shoppingCart), 1);
68      updateTotalQty();
69  }
70
71  /**
72   * @param {object} obj product code and quantity of each product
73   */
74  function parseToMap(obj) {
75      return new Map(Object.entries(obj));
76  }
77
78  /**
79   * @param {string} json JSON string format
80   * @returns {object} object from JSON
81   */
82  export function parseToObj(json) {
83      return JSON.parse(json);
84  }
85
86  export function clearAll() {
87      if (getCookie(cookieName) !== null) {
88          shoppingCart = parseToObj(getCookie(cookieName));
89          for (const key in shoppingCart) {
90              if (shoppingCart.hasOwnProperty(key)) {
91                  delete shoppingCart[key];
92              }
93          }
94          CookieUtil.unset(cookieName);
95      }
96  }

```

```

98  /**
99   * @param {string} id product code
100  */
101  export function removeItem(id) {
102      shoppingCart = parseToObj(getCookie(cookieName));
103      if (Object.keys(shoppingCart).length == 1) {
104          delete shoppingCart[id];
105          CookieUtil.unset(cookieName);
106      } else {
107          delete shoppingCart[id];
108          setCookie(cookieName, JSON.stringify(shoppingCart), 1);
109      }
110  }
111
112  /**
113   * @param {string} id product code
114  */
115  export function remove(id) {
116      shoppingCart = parseToObj(getCookie(cookieName));
117      shoppingCart[id] -= 1;
118      if (shoppingCart[id] == 0) {
119          removeItem(id);
120      } else {
121          setCookie(cookieName, JSON.stringify(shoppingCart), 1);
122      }
123      updateTotalQty();
124  }
125
126  /**
127   * @param {string} id product code
128  */
129  export function getTotalPrice(id) {
130      if (getCookie(cookieName) === null) {
131          return 0;
132      } else {
133          shoppingCart = parseToObj(getCookie(cookieName));
134          return (
135              products.find((product) => product.code == id).price *
136              shoppingCart[id]
137          );
138      }
139  }
140
141  function updateTotalQty() {
142      let totalQuantity = document.getElementById("total");
143      totalQuantity.innerHTML = `${getTotalQty()}`;
144  }

```

cookieUtil.js:

```

1  export default class CookieUtil {
2      /**
3       * @param {string} name name of the cookie
4       */
5      static getCookie(name) {
6          // console.log(`all cookies: ${document.cookie}`);
7          let cookieName = `${encodeURIComponent(name)}=`;
8          let cookieStart = document.cookie.indexOf(cookieName);
9          let cookieValue = null;
10         // console.log(`cookieName = ${cookieName}`);
11         // console.log(`cookieStart = ${cookieStart}`);
12
13         if (cookieStart > -1) {
14             let cookieEnd = document.cookie.indexOf(";", cookieStart);
15             // console.log(`cookieEnd = ${cookieEnd}`);
16             if (cookieEnd == -1) {
17                 cookieEnd = document.cookie.length;
18             }
19             cookieValue = decodeURIComponent(
20                 document.cookie.substring(
21                     cookieStart + cookieName.length,
22                     cookieEnd
23                 )
24             );
25             // console.log(`cookieValue = ${cookieValue}`);
26         }
27         return cookieValue;
28     }
29
30     /**
31      * @param {string} name name of the cookie
32      * @param {string} value value of the cookie
33      * @param {number} expires duration of the cookie in days
34      */
35     static setCookie(name, value, expires) {
36         let cookieText = `${encodeURIComponent(name)}=${encodeURIComponent(
37             value
38         )}`;
39
40         const expireDate = new Date();
41         expireDate.setTime(
42             expireDate.getTime() + expires * 24 * 60 * 60 * 1000
43         );
44
45         cookieText += `; expires=${expireDate.toUTCString()}`;
46         // cookieText += `; expires=${expires}`;
47         // console.log(`cookieText = ${cookieText}`);
48         document.cookie = cookieText;
49     }
50
51     /**
52      * @param {string} name name of the cookie
53      */
54     static unset(name) {
55         CookieUtil.setCookie(name, "", new Date(0));
56     }
57 }

```

shoppingCart.js:

add function showNetPrice & revise code

```

1  import * as lib from "../cart.js";
2  import products from "../products/products.js";
3
4  let modalBody = document.querySelector("div.modal-body");
5  let table = document.createElement("table");
6  table.style = "width: 100%;";
7  modalBody.appendChild(table);
8
9  let thead = document.createElement("thead");
10 thead.id = "thead";
11 let tbody = document.createElement("tbody");
12 tbody.id = "tbody";
13 table.appendChild(thead);
14 table.appendChild(tbody);
15
16 let tr = document.createElement("tr");
17 thead.appendChild(tr);
18
19 const heads = [
20     "#",
21     "Picture",
22     "Product code",
23     "Product name",
24     "Price",
25     "Quantity",
26     "Total",
27     "Delete",
28 ];
29 for (const head of heads) {
30     let th = document.createElement("th");
31     th.textContent = head;
32     tr.appendChild(th);
33 }
34
35 function showNetPrice() {
36     let netPrice = document.getElementById("net-price");
37     netPrice.textContent = new Intl.NumberFormat("th-TH", {
38         style: "currency",
39         currency: "THB",
40     }).format(lib.getNetPrice());
41 }
42

```

revise clear all and deleteRow

```

43 // Clear all products button
44 let clearAllButton = document.createElement("button");
45 clearAllButton.id = "clear-all-products";
46 clearAllButton.className =
47   "m-2 p-2 bg-red-500 text-white rounded-lg btn btn-outline-primary";
48 clearAllButton.innerHTML = "<b>Clear All</b>";
49 clearAllButton.addEventListener("click", () => {
50   let tableBody = document.getElementById("tbody");
51   while (tableBody.firstChild) {
52     tableBody.removeChild(tableBody.lastChild);
53   }
54   let totalQuantity = document.getElementById("total");
55   totalQuantity.innerHTML = "<b>0</b>";
56   lib.clearAll();
57   showNetPrice();
58   clearAllButton.style.visibility = "hidden";
59   // console.log("clear all!");
60 });
61 modalBody.appendChild(clearAllButton);
62
63 function deleteRow(id) {
64   let rw = document.querySelector(`#sc-row-${id + 1}`);
65   rw.remove();
66 }

```

```

68 // show shopping cart function
69 const shoppingCart = document.querySelector("#shopping-cart");
70 let modal = document.getElementById("modal-box");
71 shoppingCart.addEventListener("click", showModalBox);
72
73 function showModalBox() {
74     if (lib.getCookie(lib.cookieName) !== null) {
75         let sc = lib.parseToObj(lib.getCookie(lib.cookieName));
76         // console.log(JSON.stringify(cart, 0, 2));
77         let tableBody = document.querySelector("#tbody");
78         while (tableBody.firstChild) {
79             tableBody.removeChild(tableBody.lastChild);
80         }
81         for (let i = 0; i < Object.keys(sc).length; i++) {
82             let row = document.createElement("tr");
83             row.id = `sc-row-${i + 1}`;
84             tbody.appendChild(row);
85
86             let noCol = document.createElement("td");
87             noCol.className = "no-col";
88             noCol.textContent = i + 1;
89             noCol.style = "text-align: center;";
90             row.appendChild(noCol);
91
92             let imageCol = document.createElement("td");
93             let productImg = document.createElement("img");
94             imageCol.className = "image-col";
95             productImg.src = products.find(
96                 (product) => product.code == Object.keys(sc)[i]
97             ).imgURL;
98             imageCol.appendChild(productImg);
99             row.appendChild(imageCol);
100
101             let productCodeCol = document.createElement("td");
102             productCodeCol.className = "product-code-col";
103             productCodeCol.textContent = Object.keys(sc)[i];
104             productCodeCol.style = "text-align: center;";
105             row.appendChild(productCodeCol);
106
107             let productNameCol = document.createElement("td");
108             productNameCol.className = "product-name-col";
109             productNameCol.textContent = products.find(
110                 (product) => product.code == Object.keys(sc)[i]
111             ).name;
112             row.appendChild(productNameCol);
113
114             let productPriceCol = document.createElement("td");
115             productPriceCol.className = "product-price-col";
116             productPriceCol.textContent = new Intl.NumberFormat("th-TH", {
117                 style: "currency",
118                 currency: "THB",
119             }).format(
120                 products.find((product) => product.code == Object.keys(sc)[i])
121                     .price
122             );
123             productPriceCol.style = "text-align: center;";
124             row.appendChild(productPriceCol);

```

Add plus button and minus button

```

126     let productQuantityCol = document.createElement("td");
127     productQuantityCol.className = "product-quantity-col ";
128
129     let plusBtn = document.createElement("button");
130     plusBtn.className =
131       "plus transition duration-300 ease-in-out inline-block m-1 px-2 text-white rounded";
132     plusBtn.innerHTML = "+";
133     productQuantityCol.appendChild(plusBtn);
134     plusBtn.addEventListener("click", () => {
135       lib.add(Object.keys(sc)[i]);
136       showModalBox();
137     });
138
139     let divQty = document.createElement("div");
140     divQty.className = "px-5 py-1 rounded-lg bg-gray-200 inline-block";
141
142     let qty = document.createElement("p");
143     qty.className = "text-xl font-bold";
144     qty.innerHTML = sc[Object.keys(sc)[i]];
145     divQty.appendChild(qty);
146     productQuantityCol.appendChild(divQty);
147
148     let minusBtn = document.createElement("button");
149     minusBtn.className =
150       "minus transition duration-300 ease-in-out inline-block m-1 px-2.5 text-white rounded";
151     minusBtn.innerHTML = "-";
152     minusBtn.addEventListener("click", () => {
153       lib.remove(Object.keys(sc)[i]);
154       if (lib.getCookie(lib.cookieName) === null) {
155         deleteRow(i);
156       }
157       showModalBox();
158     });
159     productQuantityCol.appendChild(minusBtn);
160     row.appendChild(productQuantityCol);

```

```

162     let totalPriceCol = document.createElement("td");
163     totalPriceCol.className = "total-price-col";
164     totalPriceCol.textContent = new Intl.NumberFormat("th-TH", {
165         style: "currency",
166         currency: "THB",
167     }).format(lib.getTotalPrice(Object.keys(sc)[i]));
168     totalPriceCol.style = "text-align: center;";
169     row.appendChild(totalPriceCol);
170
171     let deleteCol = document.createElement("td");
172     let deleteButton = document.createElement("button");
173     deleteButton.type = "button";
174     deleteButton.innerHTML = "&#x1F5D1;";
175     deleteButton.className = "delete-button";
176     deleteButton.addEventListener("click", () => {
177         let totalQuantity = document.getElementById("total");
178         lib.removeItem(Object.keys(sc)[i]);
179         totalQuantity.innerHTML = `<b>${lib.getTotalQty()}</b>`;
180         deleteRow(i);
181         showModalBox();
182         showNetPrice();
183     });
184     deleteCol.style = "text-align: center;";
185     deleteCol.appendChild(deleteButton);
186     row.appendChild(deleteCol);
187 }
188 clearAllButton.style.visibility = "visible";
189 } else {
190     clearAllButton.style.visibility = "hidden";
191 }
192 modal.style.display = "block";
193 showNetPrice();
194 // console.log(lib.getNetPrice());
195 }
196
197 let span = document.getElementsByClassName("close")[0];
198 span.addEventListener("click", () => {
199     modal.style.display = "none";
200 });
201
202 // When the user clicks anywhere outside of the modal, close it
203 window.onclick = (event) => {
204     if (event.target == modal) {
205         modal.style.display = "none";
206     }
207 };
208

```


theme.js:

revise code

```
1  //add local storage theme
2  const themeSwitcher = document.getElementById("theme-switch");
3
4  themeSwitcher.checked = false;
5  function clickHandler() {
6      if (this.checked) {
7          document.body.classList.remove("light");
8          document.body.classList.add("dark");
9          localStorage.setItem("theme", "dark");
10     } else {
11         document.body.classList.add("light");
12         document.body.classList.remove("dark");
13         localStorage.setItem("theme", "light");
14     }
15 }
16 themeSwitcher.addEventListener("click", clickHandler);
17
18 function checkTheme() {
19     const localStorageTheme = localStorage.getItem("theme");
20
21     if (localStorageTheme === "dark") {
22         document.body.classList.add(localStorageTheme);
23         themeSwitcher.checked = true;
24     } else {
25         document.body.classList.add(localStorageTheme);
26         themeSwitcher.checked = false;
27     }
28 }
29
30 export { checkTheme as default };
```


loadStorage.js:

```
1  import checkTheme from "../theme.js";
2
3  window.onload = checkTheme();
```

Output:

W 6

Monitor List




Product: BENQ ZOWIE XL2731 27"

Price: \$9,900.00

Resolution: 1920x1080

Out of Stock

Add




Product: BENQ MOBIUZ EX3415R 34"

Price: \$35,900.00

Resolution: 3440x1440

In stock: 59

Add




Product: BENQ MOBIUZ EX2710R 27"

Price: \$21,900.00

Resolution: 2560x1440

In stock: 104

Add




Product: BENQ EW2780Q 27"

Price: \$9,500.00

Resolution: 2560x1440

In stock: 156

Add






Product: BENQ EW2480 24"

Price: \$5,500.00

Resolution: 1920x1080


In stock: 222

Add



W 0

Monitor List




Product: BENQ ZOWIE XL2731 27"

Price: \$9,900.00

Resolution: 1920x1080

Out of Stock

Add




Product: BENQ MOBIUZ EX3415R 34"

Price: \$35,900.00

Resolution: 3440x1440

In stock: 59

Add




Product: BENQ MOBIUZ EX2710R 27"

Price: \$21,900.00

Resolution: 2560x1440

In stock: 104

Add




Product: BENQ EW2780Q 27"

Price: \$9,500.00

Resolution: 2560x1440

In stock: 156

Add






Product: BENQ EW2480 24"

Price: \$5,500.00

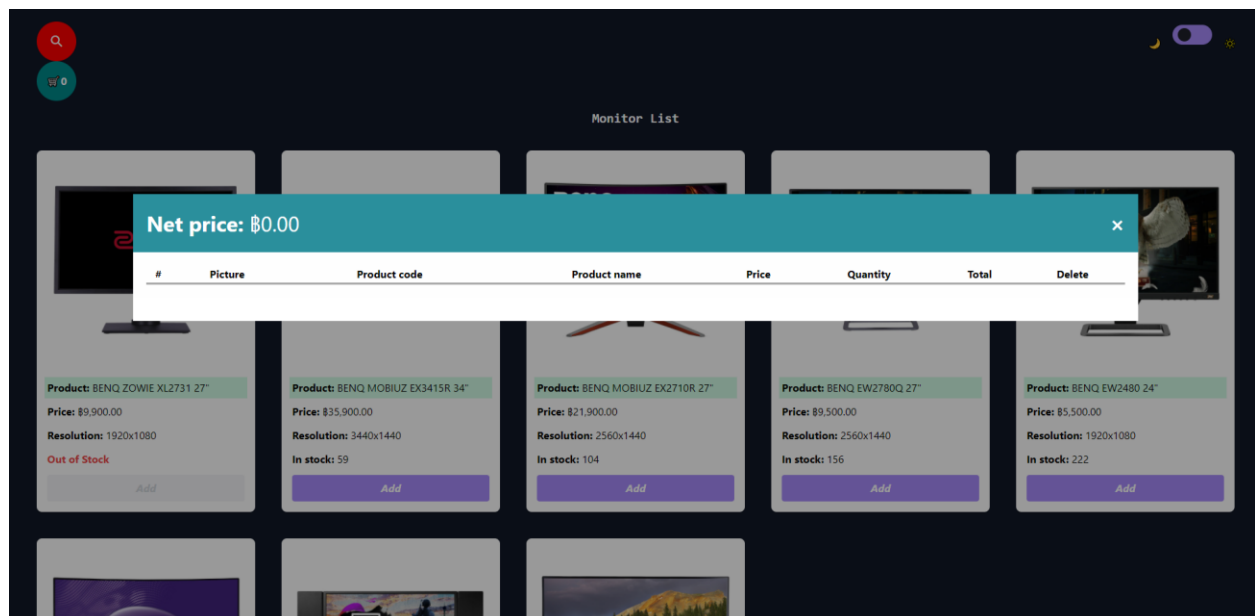
Resolution: 1920x1080

In stock: 222

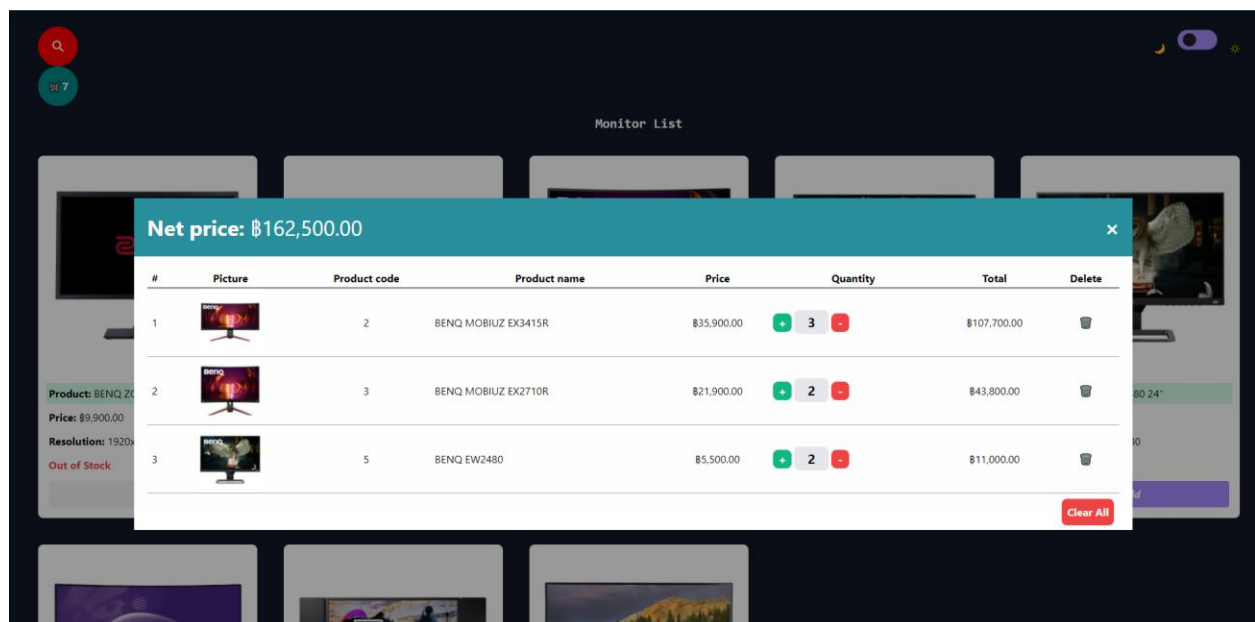
Add



ถ้าไม่มีสินค้าจะไม่มีปุ่ม clear all



สามารถเพิ่มลดสินค้าได้โดยกดที่ปุ่มบวกและลบ



GIT LINK (This Assignment): [Click here](#)

GIT LINK (All Assignment): [Click here](#)

Work ratio

| No. | Student ID | Name | Part | Percentage |
|-----|-------------|----------------------|---|------------|
| 1 | 63130500006 | Kittiphum Uamthon | add "show net price function" and documents | 20% |
| 2 | 63130500026 | Chotiwit Souyan | add "plus & minus button" in modal box | 20% |
| 3 | 63130500032 | Nutchanon Assawachin | minor fix theme.js & clear all button and documents | 15% |
| 4 | 63130500034 | Nuttida Meeboon | add "plus & minus button" in modal box | 15% |
| 5 | 63130500042 | Songglod Petchamras | revise & improve cart.js (rewrite the whole file) | 30% |