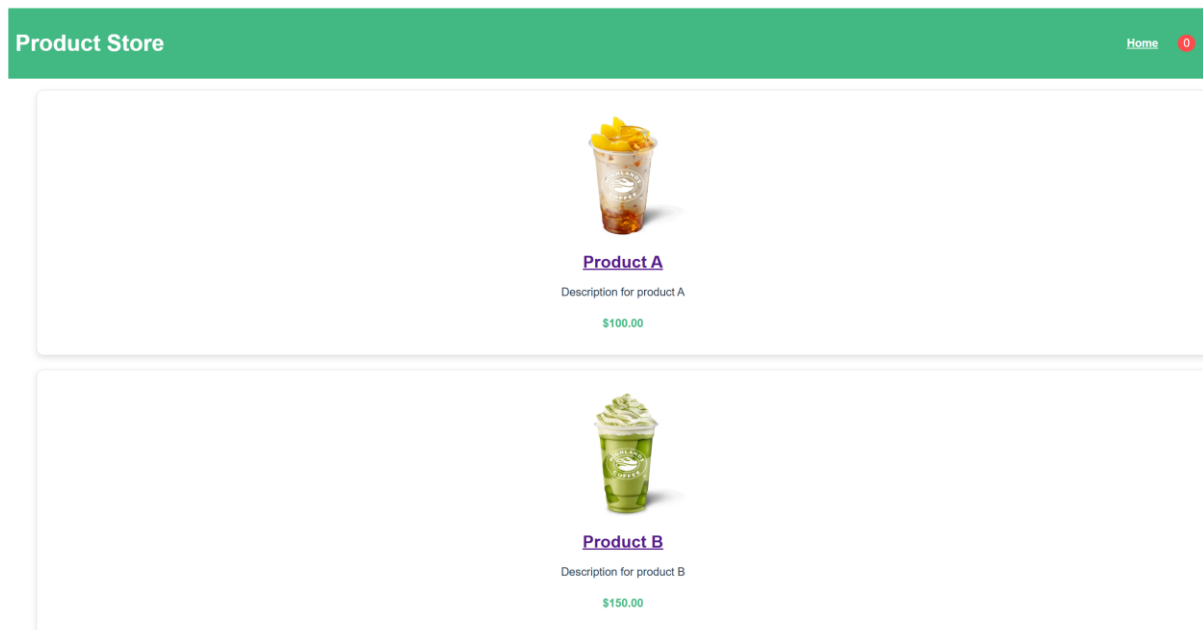
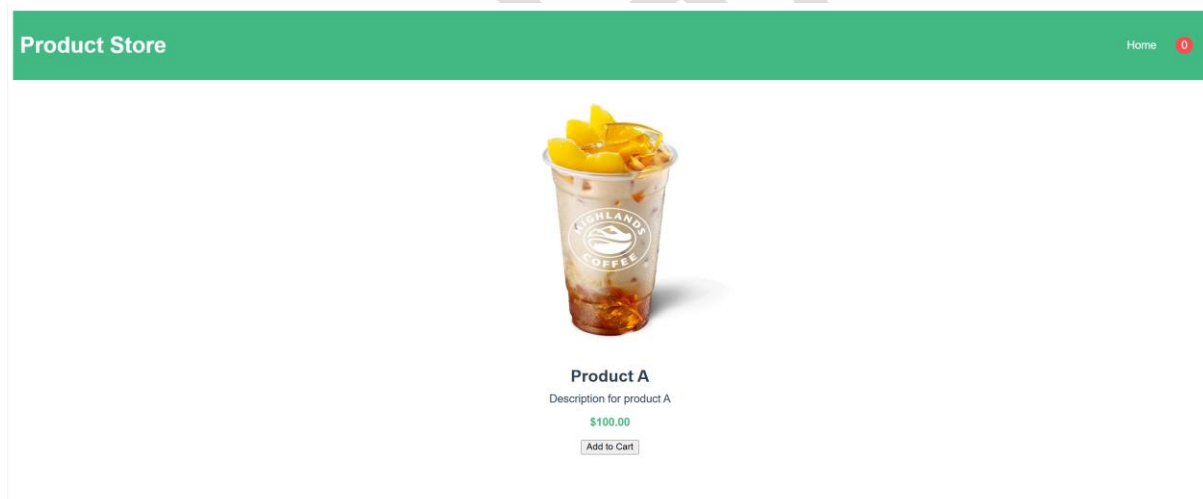


## LAB 4: KẾT NỐI API (JSON SERVER)

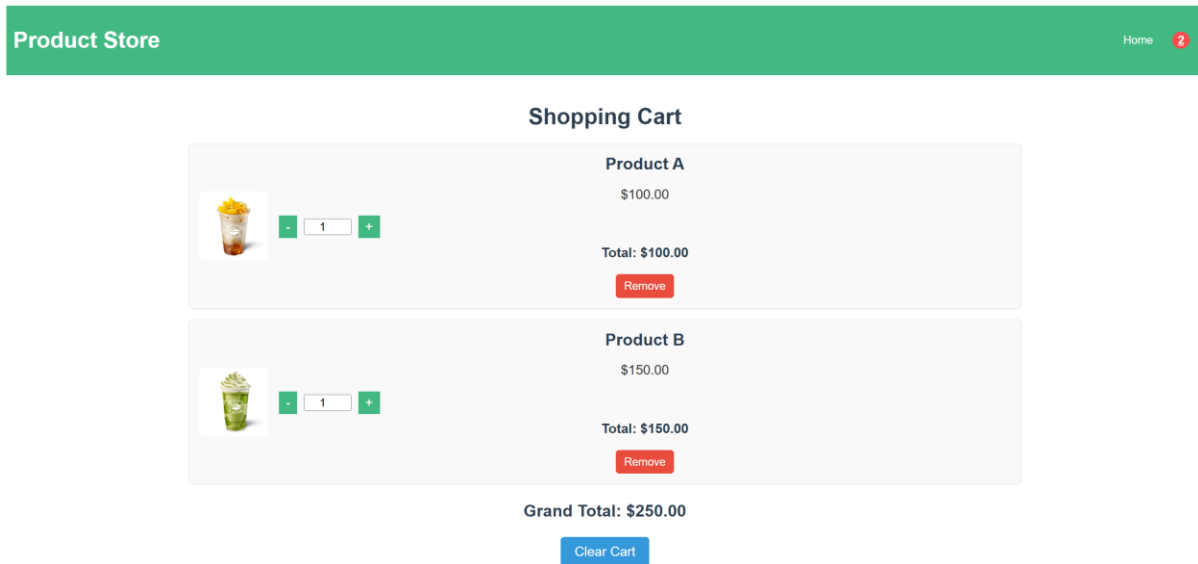
### Bài 1: Tạo một ứng dụng đơn giản kết nối API



Click vào tên sản phẩm ra chi tiết sản phẩm



Click Add to cart được thêm vào giỏ hàng

**Yêu cầu:**

- Thực hiện các chức năng như hình mẫu
- Trang giỏ hàng thực hiện tăng giảm sản phẩm, remove và clear Cart
- Trong giỏ hàng nếu số lượng giảm về 0 thì tự động xóa sản phẩm ra khỏi giỏ hàng

**Hướng dẫn:****Bước 1:** vue create bai1\_json\_serve**Bước 2:** npm install axios và cài vuex: npm install vuex@next --save**Bước 3:** Cấu Hình Vue Router

npm install vue-router@4

hoặc

npm install vue-router

**Bước 4:** Tạo json server như sau:

- B1: Cài npm i -g json-server ( cài toàn cục)
  - B2: Tạo file tên db.json trong bai1\_json\_serve với nội dung
- ```
{
  "products": [
    { "id": 1, "name": "Product A", "description": "Description for product A",
      "price": 100, "image": "trathachdao.jpg" },
    { "id": 2, "name": "Product B", "description": "Description for product B",
      "price": 150, "image": "traxanh.jpg" },
    { "id": 3, "name": "Product C", "description": "Description for product C",
      "price": 200, "image": "trathachdao.jpg" }
  ]
}
```

```
]
}
```

- B3: mở đường dẫn `bai1_json_serve` chạy lệnh `json-server --watch db.json`

```
C:\WINDOWS\system32\cmd. x
Microsoft Windows [Version 10.0.22631.4112]
(c) Microsoft Corporation. All rights reserved.

D:\OneDrive_Personal\OneDrive\Documents\NamHoc2024-2025\CongCuVaMoiTruongPhatTrieuUngDung\
Javascript\VueJS\BaiGiai\Lab4\Bai1\bai1-json>json-server --watch db.json

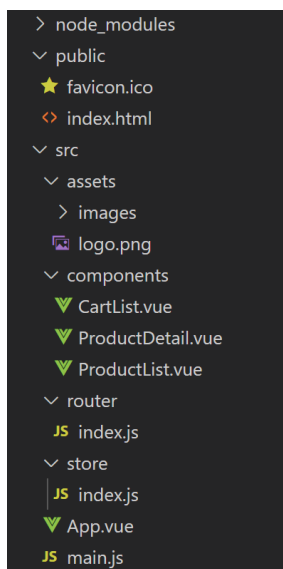
\{^_^}/ hi!

Loading db.json
Done

Resources
http://localhost:3000/products

Home
http://localhost:3000
```

Mở `bai1_json_server` tạo cấu trúc thư mục như sau:



## Bước 5: Tạo các components

### ProductList.vue

```
<template>
  <div>
    <div v-if="loading">Loading...</div>
    <ul v-if="!loading && products.length">
      <li v-for="product in products" :key="product.id" class="product-item">
        
        <router-link :to="{ name: 'ProductDetail', params: { id: product.id } }">
          <h2 class="product-name">{{ product.name }}</h2>
        </router-link>
        <p class="product-description">{{ product.description }}</p>
        <p class="product-price">${{ product.price.toFixed(2) }}</p>
      </li>
    </ul>
    <div v-if="!loading && !products.length">No products available</div>
  </div>
</template>
```

```

<script>
import axios from 'axios';

export default {
  name: 'ProductList',
  data() {
    return {
      products: [],
      loading: true,
    };
  },
  methods: {
    fetchProducts() {
      axios.get('http://localhost:3000/products')
        .then(response => {
          this.products = response.data;
          this.loading = false;
        })
        .catch(() => {
          this.loading = false;
          // Handle error without storing it in a variable
        });
    }
  },
  mounted() {
    this.fetchProducts();
  }
};
</script>

```

### ProductDetail.vue

```

<template>
  <div>
    <div v-if="loading">Loading...</div>
    <div v-if="!loading && product">
      
      <h2 class="product-name">{{ product.name }}</h2>
      <p class="product-description">{{ product.description }}</p>
      <p class="product-price">${{{ product.price.toFixed(2) }}}</p>
      <button @click="addToCart">Add to Cart</button>
    </div>
    <div v-if="!loading && !product">Product not found</div>
  </div>
</template>

```

```

<script>
import axios from 'axios';
import { mapMutations } from 'vuex';

export default {
  name: 'ProductDetail',
  props: {
    id: {
      type: String,
      required: true
    }
  },
  data() {
    return {
      product: null,
      loading: true,
    };
  },
  methods: {
    ...mapMutations(['addToCart']),
    fetchProduct() {
      axios.get('http://localhost:3000/products/${this.id}`)
        .then(response => {
          this.product = response.data;
          this.loading = false;
        })
        .catch(error => {
          console.error('Failed to fetch product:', error);
          this.loading = false;
        });
    },
    addToCart() {
      // Check if product is not null before calling mutation
      if (this.product) {
        this.$store.commit('addToCart', { ...this.product, quantity: 1 });
      }
    }
  },
  mounted() {
    this.fetchProduct();
  }
};
</script>

```

## CartList.vue

```

<template>
  <div class="cart-container">
    <h1 class="cart-title">Shopping Cart</h1>
    <div v-if="cart.length === 0" class="empty-cart-message">Your cart is empty.</div>
    <ul v-if="cart.length > 0" class="cart-list">
      <li v-for="item in cart" :key="item.id" class="cart-item">
        
        <div class="cart-details">
          <h2 class="cart-name">{{ item.name }}</h2>
          <p class="cart-price">${{ item.price.toFixed(2) }}</p>
          <div class="quantity-container">
            <button class="quantity-button" @click="changeQuantity(item, -1)">-</button>
            <input
              type="number"
              v-model.number="item.quantity"
              @input="handleQuantityChange(item)"
              min="0"
              class="quantity-input"
            />
            <button class="quantity-button" @click="changeQuantity(item, 1)">+</button>
          </div>
          <p class="cart-total-price">Total: ${{ (item.price * item.quantity).toFixed(2) }}</p>
          <button class="remove-button" @click="removeFromCart(item.id)">Remove</button>
        </div>
      </li>
      <div class="cart-summary">
        <h3 class="cart-total">Grand Total: ${{ total.toFixed(2) }}</h3>
        <button class="clear-cart-button" @click="clearCart">Clear Cart</button>
      </div>
    </ul>
  </div>
</template>

```

```

<script>
import { mapState, mapMutations } from 'vuex';

export default {
  name: 'CartList',
  computed: {
    ...mapState(['cart']),
    total() {
      return this.cart.reduce((sum, item) => sum + item.price * item.quantity, 0);
    }
  },
  methods: {
    ...mapMutations(['removeFromCart', 'updateQuantity', 'clearCart']),
    handleQuantityChange(item) {
      if (item.quantity <= 0) {
        this.removeFromCart(item.id);
      } else {
        this.updateQuantity({ id: item.id, quantity: item.quantity });
      }
    },
    changeQuantity(item, delta) {
      const newQuantity = item.quantity + delta;
      if (newQuantity <= 0) {
        this.removeFromCart(item.id);
      } else {
        this.updateQuantity({ id: item.id, quantity: newQuantity });
      }
    }
  }
};
</script>

```

router/index.js

```
import { createRouter, createWebHistory } from 'vue-router';
import ProductList from '../components/ProductList.vue';
import ProductDetail from '../components/ProductDetail.vue';
import Cart from '../components/CartList.vue';

const routes = [
  {
    path: '/',
    name: 'ProductList',
    component: ProductList
  },
  {
    path: '/product/:id',
    name: 'ProductDetail',
    component: ProductDetail,
    props: true
  },
  {
    path: '/cart',
    name: 'Cart',
    component: Cart
  }
];

const router = createRouter({
  history: createWebHistory(process.env.BASE_URL),
  routes
});

export default router;
```

store/index.js

```
import { createStore } from 'vuex';

export default createStore({
  state() {
    return {
      cart: []
    };
  },
  mutations: {
    addToCart(state, product) {
      const item = state.cart.find(p => p.id === product.id);
      if (item) {
        item.quantity += product.quantity;
      } else {
        state.cart.push(product);
      }
    },
    removeFromCart(state, productId) {
      state.cart = state.cart.filter(item => item.id !== productId);
    },
    updateQuantity(state, { id, quantity }) {
      const item = state.cart.find(p => p.id === id);
      if (item) {
        item.quantity = quantity;
      }
    },
    clearCart(state) {
      state.cart = [];
    }
  }
});
```

Main.js

```
import { createApp } from 'vue';
import App from './App.vue';
import router from './router';
import store from './store';

const app = createApp(App);

app.use(router);
app.use(store);
app.mount('#app');
```

App.vue

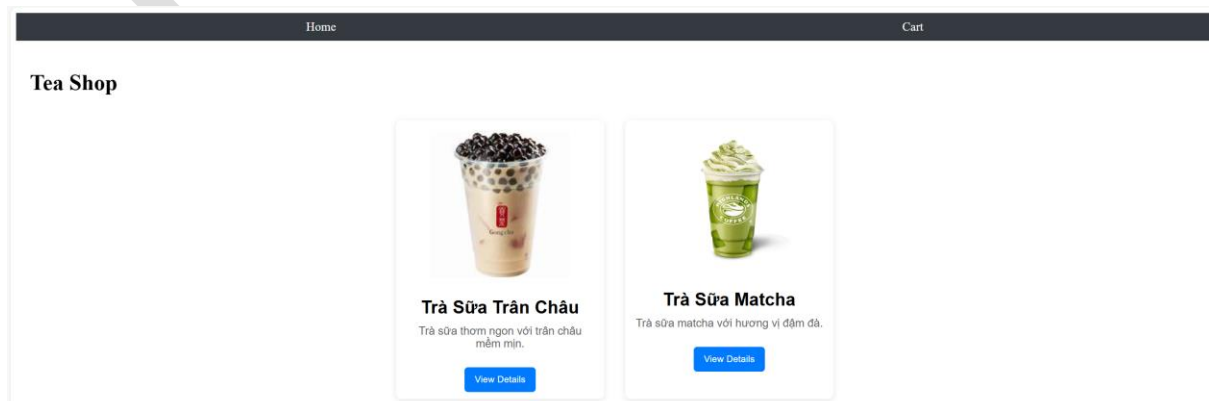
```
<template>
  <div id="app">
    <header>
      <h1>Product Store</h1>
      <nav>
        <router-link to="/">Home</router-link>
        <router-link to="/cart" class="cart-link">
          <i class="fa fa-shopping-cart"></i>
          <span class="cart-count">{{ cartCount }}</span>
        </router-link>
      </nav>
    </header>

    <router-view></router-view>
  </div>
</template>

<script>
import { mapState } from 'vuex';

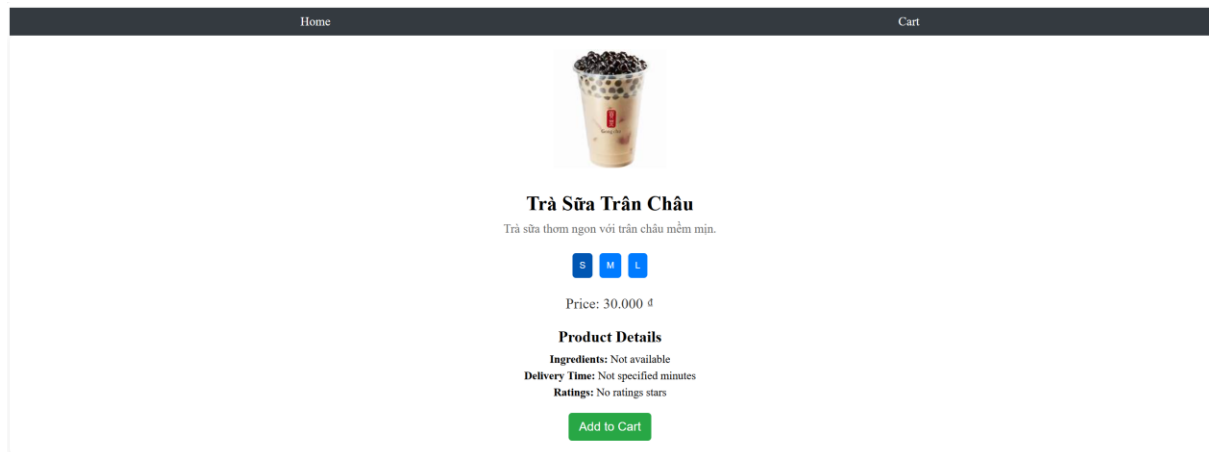
export default {
  name: 'App',
  computed: {
    ...mapState(['cart']),
    cartCount() {
      return this.cart.reduce((total, item) => total + item.quantity, 0);
    }
  }
};
</script>
```

Chạy npm run serve

**Bài 2: Bán trà sữa**

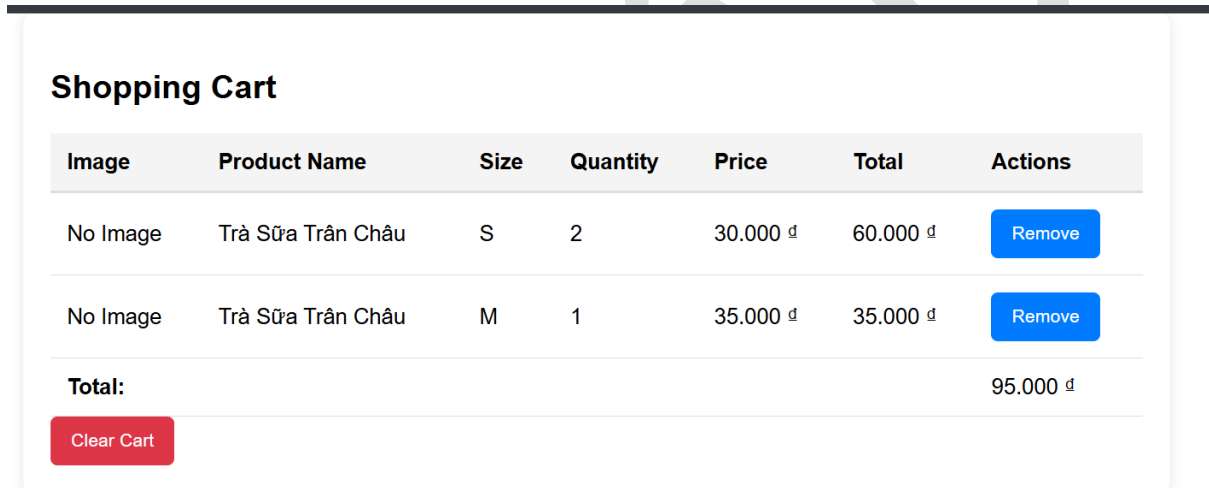
Trang chi tiết sản phẩm

- Người dùng click vào View Details ra trang chi tiết sản phẩm

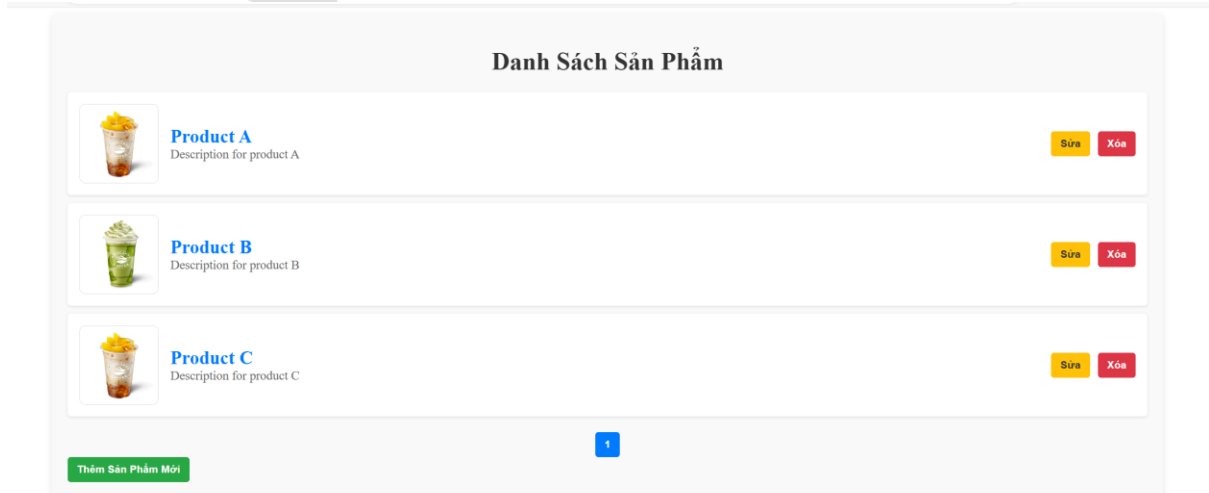


### Trang giỏ hàng

- Người dùng chọn size, mỗi size sẽ có giá tiền tương ứng, nếu cùng sản phẩm cùng size thì số lượng tăng lên, khác sản phẩm thì khác size thì tạo ra dòng mới



### Bài 3: Quản lý thông tin trà sữa sử dụng JSON SERVE



Yêu cầu:

Thêm sản phẩm mới



## Thêm Sản Phẩm Mới


Tên sản phẩm	Mô tả sản phẩm
Tên file ảnh	<input type="button" value="Thêm"/> <input type="button" value="Hủy"/>

Nhập thông tin và nhấn nút thêm

## Thêm Sản Phẩm Mới

Trà đào cam sả	Vị đào thơm ngon, vị sả tươi
trathachdao.jpg	<input type="button" value="Thêm"/> <input type="button" value="Hủy"/>

### Danh Sách Sản Phẩm

	<b>Trà đào cam sả</b> Vị đào thơm ngon, vị sả tươi mát	<input type="button" value="Sửa"/> <input type="button" value="Xóa"/>
-------------------------------------------------------------------------------------	-----------------------------------------------------------	-----------------------------------------------------------------------

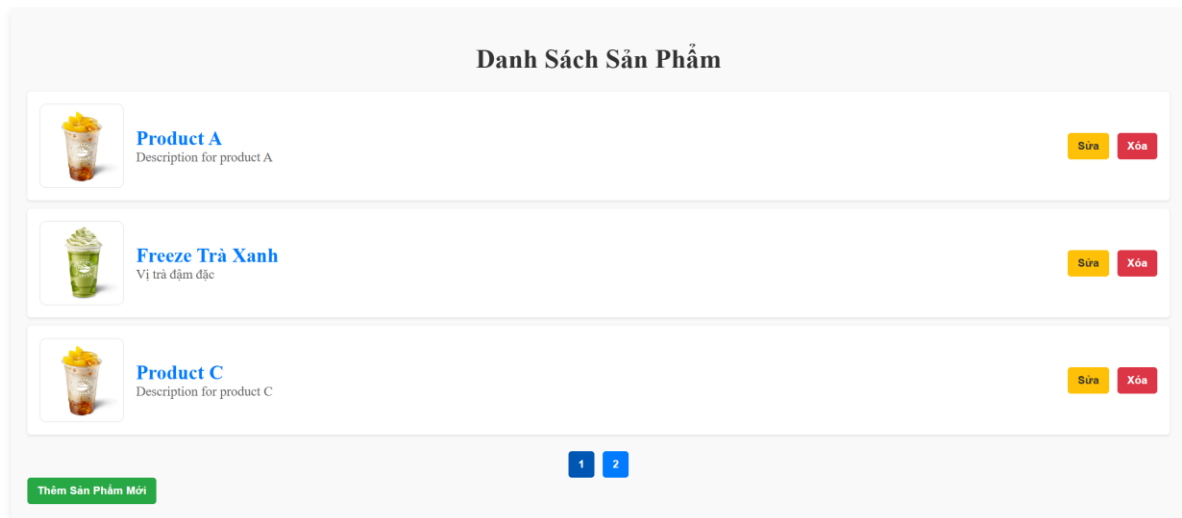
1 2

Chỉnh sửa sản phẩm

## Chỉnh sửa Sản Phẩm

Freeze Trà Xanh	Vị trà đậm đặc
traxanh.jpg	<input type="button" value="Lưu"/> <input type="button" value="Hủy"/>

Sản phẩm sau khi được chỉnh sửa



Thực hiện chức năng xóa sản phẩm.

Thực hiện chức năng phân trang, 3 sản phẩm là 1 trang

Hướng dẫn:

Tạo thư mục như hình mẫu

