Creating an online shopping system using Spring MVC for the backend and React for the frontend involves setting up a RESTful API with Spring Boot and developing a React application that interacts with this API. Here's a step-by-step guide to creating this project.

**Project Structure**

**Backend (Spring Boot)**

online-shop-backend/

│

├── src/main/java/com/onlineshop/

│ ├── OnlineShopApplication.java

│ ├── controller/

│ │ ├── ProductController.java

│ │ └── CartController.java

│ ├── model/

│ │ ├── Product.java

│ │ └── CartItem.java

│ ├── repository/

│ │ ├── ProductRepository.java

│ │ └── CartItemRepository.java

│ └── service/

│ ├── ProductService.java

│ └── CartService.java

│

└── src/main/resources/

├── application.properties

└── data.sql

**Frontend (React)**

online-shop-frontend/

│

├── public/

├── src/

│ ├── components/

│ │ ├── ProductList.js

│ │ ├── Product.js

│ │ ├── Cart.js

│ ├── App.js

│ ├── index.js

│ └── api/

│ ├── api.js

├── package.json

└── .env

**Backend: Spring Boot**

**1. Setup Spring Boot Application**

**OnlineShopApplication.java**

package com.onlineshop;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class OnlineShopApplication {

public static void main(String[] args) {

SpringApplication.run(OnlineShopApplication.class, args);

}

}

**2. Create Models**

**Product.java**

package com.onlineshop.model;

import javax.persistence.\*;

@Entity

public class Product {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

private String name;

private String description;

private double price;

private String imageUrl;

// Getters and setters

}

**CartItem.java**

package com.onlineshop.model;

import javax.persistence.\*;

@Entity

public class CartItem {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@ManyToOne

@JoinColumn(name = "product\_id")

private Product product;

private int quantity;

// Getters and setters

}

**3. Create Repositories**

**ProductRepository.java**

package com.onlineshop.repository;

import com.onlineshop.model.Product;

import org.springframework.data.jpa.repository.JpaRepository;

public interface ProductRepository extends JpaRepository<Product, Long> {

}

**CartItemRepository.java**

package com.onlineshop.repository;

import com.onlineshop.model.CartItem;

import org.springframework.data.jpa.repository.JpaRepository;

public interface CartItemRepository extends JpaRepository<CartItem, Long> {

}

**4. Create Services**

**ProductService.java**

package com.onlineshop.service;

import com.onlineshop.model.Product;

import com.onlineshop.repository.ProductRepository;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import java.util.List;

@Service

public class ProductService {

@Autowired

private ProductRepository productRepository;

public List<Product> getAllProducts() {

return productRepository.findAll();

}

public Product getProductById(Long id) {

return productRepository.findById(id).orElse(null);

}

}

**CartService.java**

package com.onlineshop.service;

import com.onlineshop.model.CartItem;

import com.onlineshop.repository.CartItemRepository;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import java.util.List;

@Service

public class CartService {

@Autowired

private CartItemRepository cartItemRepository;

public List<CartItem> getCartItems() {

return cartItemRepository.findAll();

}

public void addToCart(CartItem cartItem) {

cartItemRepository.save(cartItem);

}

public void removeFromCart(Long id) {

cartItemRepository.deleteById(id);

}

}

**5. Create Controllers**

**ProductController.java**

package com.onlineshop.controller;

import com.onlineshop.model.Product;

import com.onlineshop.service.ProductService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.\*;

import java.util.List;

@RestController

@RequestMapping("/api/products")

public class ProductController {

@Autowired

private ProductService productService;

@GetMapping

public List<Product> getAllProducts() {

return productService.getAllProducts();

}

@GetMapping("/{id}")

public Product getProductById(@PathVariable Long id) {

return productService.getProductById(id);

}

}

**CartController.java**

package com.onlineshop.controller;

import com.onlineshop.model.CartItem;

import com.onlineshop.service.CartService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.\*;

import java.util.List;

@RestController

@RequestMapping("/api/cart")

public class CartController {

@Autowired

private CartService cartService;

@GetMapping

public List<CartItem> getCartItems() {

return cartService.getCartItems();

}

@PostMapping

public void addToCart(@RequestBody CartItem cartItem) {

cartService.addToCart(cartItem);

}

@DeleteMapping("/{id}")

public void removeFromCart(@PathVariable Long id) {

cartService.removeFromCart(id);

}

}

**6. Application Properties**

**application.properties**

spring.datasource.url=jdbc:h2:mem:testdb

spring.datasource.driverClassName=org.h2.Driver

spring.datasource.username=sa

spring.datasource.password=password

spring.jpa.database-platform=org.hibernate.dialect.H2Dialect

spring.h2.console.enabled=true

**7. Sample Data**

**data.sql**

INSERT INTO PRODUCT (name, description, price, image\_url) VALUES ('Product 1', 'Description 1', 10.0, 'url1');

INSERT INTO PRODUCT (name, description, price, image\_url) VALUES ('Product 2', 'Description 2', 20.0, 'url2');

**Frontend: React**

**1. Setup React Application**

Create a new React app using Create React App.

npx create-react-app online-shop-frontend

cd online-shop-frontend

npm install axios

**2. API Configuration**

**src/api/api.js**

import axios from 'axios';

const api = axios.create({

baseURL: 'http://localhost:8080/api',

});

export default api;

**3. Components**

**src/components/ProductList.js**

import React, { useState, useEffect } from 'react';

import api from '../api/api';

import Product from './Product';

function ProductList() {

const [products, setProducts] = useState([]);

useEffect(() => {

api.get('/products')

.then(response => {

setProducts(response.data);

})

.catch(error => {

console.error('There was an error fetching the products!', error);

});

}, []);

return (

<div>

{products.map(product => (

<Product key={product.id} product={product} />

))}

</div>

);

}

export default ProductList;

**src/components/Product.js**

import React from 'react';

import api from '../api/api';

function Product({ product }) {

const addToCart = () => {

api.post('/cart', { productId: product.id, quantity: 1 })

.then(response => {

console.log('Product added to cart');

})

.catch(error => {

console.error('There was an error adding the product to the cart!', error);

});

};

return (

<div>

<h3>{product.name}</h3>

<p>{product.description}</p>

<p>${product.price}</p>

<button onClick={addToCart}>Add to Cart</button>

</div>

);

}

export default Product;

**src/components/Cart.js**

import React, { useState, useEffect } from 'react';

import api from '../api/api';

function Cart() {

const [cartItems, setCartItems] = useState([]);

const [totalPrice, setTotalPrice] = useState(0);

useEffect(() => {

api.get('/cart')

.then(response => {

setCartItems(response.data);

calculateTotal(response.data);

})

.catch(error => {

console.error('There was an error fetching the cart items!', error);

});

}, []);

const calculateTotal = (items) => {

let total = 0;

items.forEach(item => {

total += item.product.price \* item.quantity;

});

setTotalPrice(total);

};

return (

<div>

<h2>Cart</h2>

<ul>

{cartItems.map(item => (

<li key={item.id}>

{item.product.name} - {item.quantity} x ${item.product.price}

</li>

))}

</ul>

<h3>Total: ${totalPrice}</h3>

</div>

);

}

export default Cart;

**4. Main Application**

**src/App.js**

import React from 'react';

import { BrowserRouter as Router, Route, Switch } from 'react-router-dom';

import ProductList from './components/ProductList';

import Cart from './components/Cart';

function App() {

return (

<Router>

<div>

<Switch>

<Route path="/" exact component={ProductList} />

<Route path="/cart" component={Cart} />

</Switch>

</div>

</Router>

);

}

export default App;

**5. Entry Point**

**src/index.js**

import React from 'react';

import ReactDOM from 'react-dom';

import './index.css';

import App from './App';

import { BrowserRouter } from 'react-router-dom';

ReactDOM.render(

<BrowserRouter>

<App />

</BrowserRouter>,

document.getElementById('root')

);

**Running the Project**

1. **Backend: Spring Boot**
   * Start the Spring Boot application by running the OnlineShopApplication class.
   * Ensure the H2 console is accessible at http://localhost:8080/h2-console and check the data.
2. **Frontend: React**
   * Navigate to the online-shop-frontend directory.
   * Start the React application.

npm start

Your online shopping system should now be running, with a Spring Boot backend serving a RESTful API and a React frontend interacting with this API. The frontend displays products, allows adding them to a cart, and shows the cart's contents.