Shopping App

App.js

import './App.css';

import Navbar from './components/Navbar';

import Footer from './components/Footer';

import Products from './components/Products';

function App() {

return (

<div className="App">

<Navbar/>

<Products/>

<Footer/>

</div>

);

}

export default App;

Components folder:Navbar.js,Footer.js,Products.js Navbar.js

Navbar.js

import React from 'react';

import '../App.css'

const Navbar = () => {

return (

<div className='Navbar'>

<h1>Shopping App</h1>

</div>

);

}

export default Navbar;

Products.js

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

import '../App.css';

import axios from 'axios';

const Products = () => {

const [loading, setLoading] = useState(false);

const [data, setData] = useState([]);

useEffect(() => {

setLoading(true);

axios({

method: "GET",

url: "https://fakestoreapi.com/products"

})

.then(res => {

console.log(res.data);

setData(res.data);

})

.catch(e => console.log(e))

.finally(() => setLoading(false));

}, []);

return (

<div className='products-container'>

{loading && (

<div>Loading...</div>

)}

{!loading && data.length > 0 && (

data.map(product => (

<div key={product.id} className="card">

<div>

<img src={product.image}

alt={product.title} />

</div>

<div className='card-description'>

<h6>{product.title}</h6>

<p>Category: {product.category}</p>

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

</div>

</div>

))

)}

</div>

);

}

export default Products;

Footer.js

import React from 'react';

import '../App.css'

const Footer = () => {

return (

<div className='Footer'>

<p>Shopping app 2024&copy;</p>

</div>

);

}

export default Footer;

App.css

body {

font-family: 'Helvetica Neue', Arial, sans-serif; margin: 0;

padding: 0;

background-color: #f8f9fa;

}

.Navbar {

background-color: #007bff;

color: white;

padding: 15px 20px;

text-align: center;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1); }

.Navbar h1 {

margin: 0;

font-size: 24px;

font-weight: 500;

}

.Footer {

background-color: #343a40;

color: white;

padding: 10px 20px;

text-align: center;

position: fixed;

width: 100%;

bottom: 0;

box-shadow: 0 -2px 4px rgba(0, 0, 0, 0.1); }

.Footer p {

margin: 0;

font-size: 14px;

}

.products-container {

padding: 20px;

display: grid;

grid-template-columns: repeat(3, 1fr);

gap: 20px;

background-color: #f8f9fa;

min-height: calc(100vh - 120px);

}

.card {

background-color: #fff;

border: 1px solid #ddd;

border-radius: 8px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

overflow: hidden;

text-align: left;

transition: transform 0.3s ease, box-shadow 0.3s ease; cursor: pointer;

}

.card:hover {

transform: translateY(-5px);

box-shadow: 0 8px 16px rgba(0, 0, 0, 0.2);

}

.card img {

width: 100%;

height: 200px;

object-fit: contain;

}

.card-description {

padding: 15px;

display: flex;

flex-direction: column; gap: 10px;

}

.card-description h6 { margin: 0;

font-size: 16px;

font-weight: 600;

color: #333;

}

.card-description p { margin: 0;

color: #555;

font-size: 14px;

}

.card-description .price { font-size: 18px;

font-weight: bold;

color: #007bff;

}

.products-container > div { font-size: 18px;

color: #333;

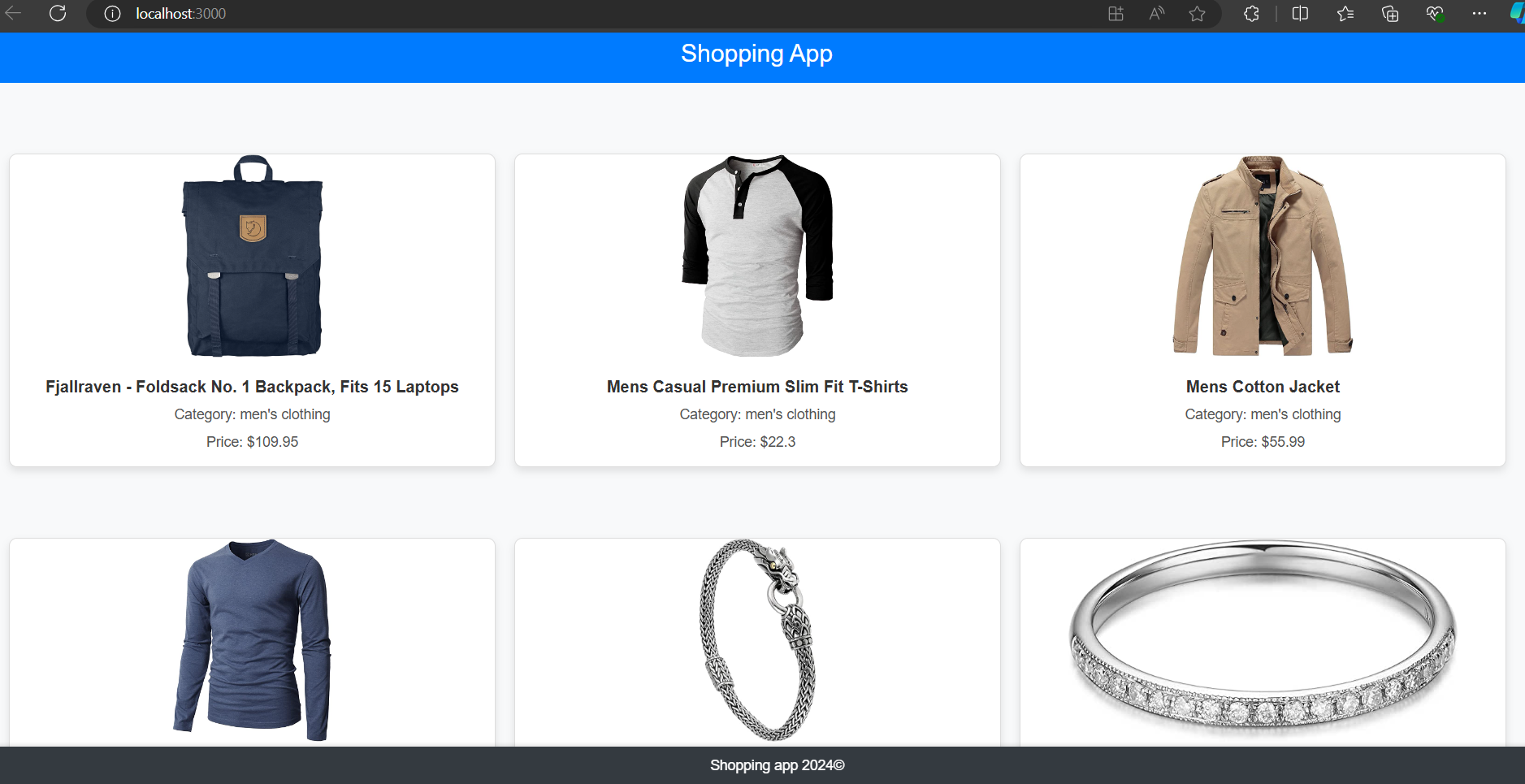
text-align: center;

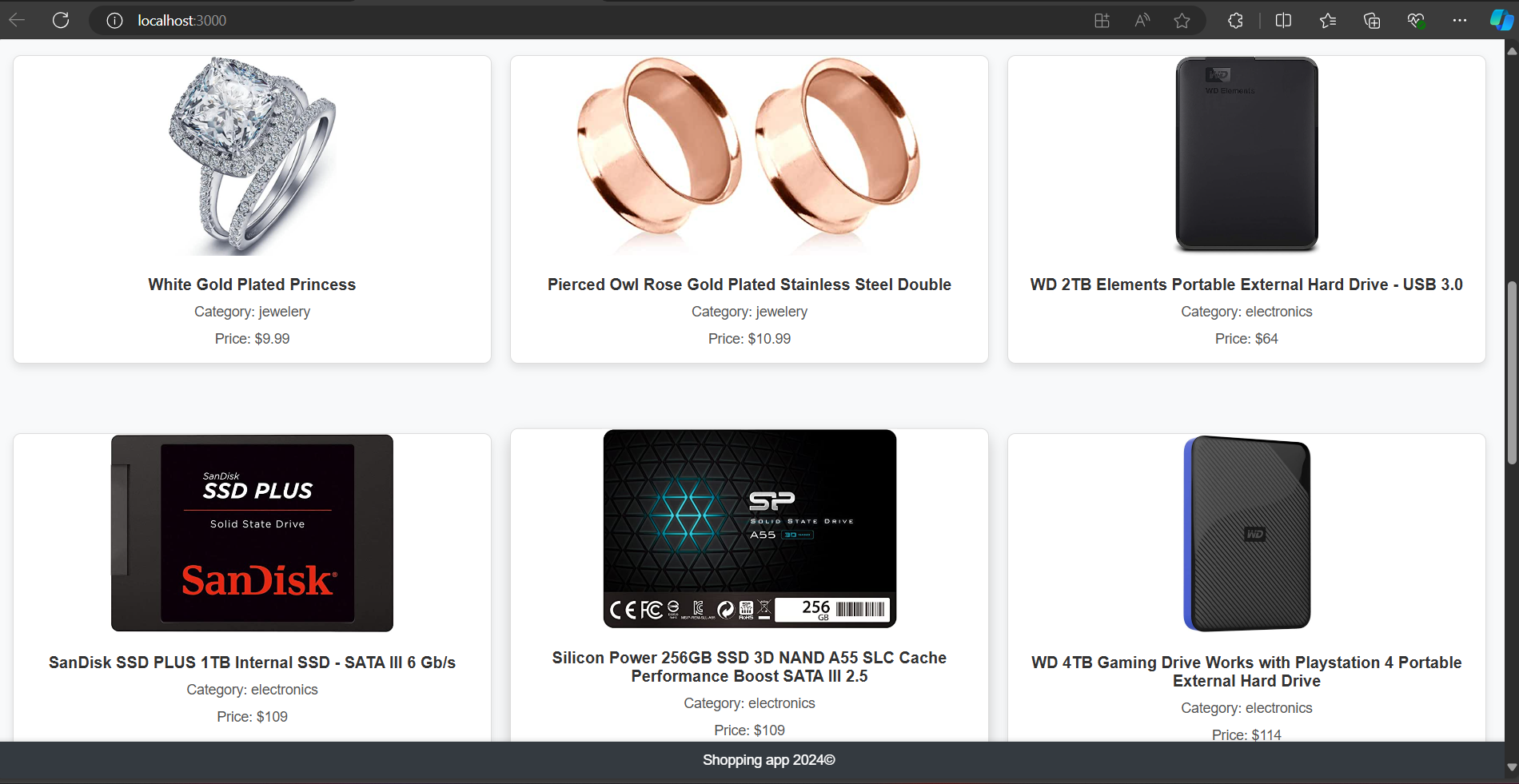
width: 100%;

margin-top: 50px;

}

Output:





Restaurant App

App.js

import React, { useState } from 'react';

import './App.css';

const App = () => {

  const [customerName, setCustomerName] = useState('');

  const [customerNumber, setCustomerNumber] = useState('');

  const [selectedItem, setSelectedItem] = useState('1');

  const [itemSize, setItemSize] = useState('Medium');

  const [quantity, setQuantity] = useState(1);

  const [specialInstructions, setSpecialInstructions] = useState('');

  const [toppings, setToppings] = useState([]);

  const [orderType, setOrderType] = useState('pickup');

  const [termsAccepted, setTermsAccepted] = useState(false);

  const [orderDetails, setOrderDetails] = useState(null);

  const [errors, setErrors] = useState({});

  const menuItems = [

    { id: '1', name: 'Pizza' },

    { id: '2', name: 'Burger' },

    { id: '3', name: 'Pasta' },

  ];

  const images = [

    '/images/pizza.jpg',

    '/images/burger.jpg',

    '/images/pasta.jpg',

  ];

  const handleSubmit = (e) => {

    e.preventDefault();

    const newErrors = {};

    if (customerNumber.length !== 10) {

      newErrors.customerNumber = 'Contact number must be exactly 10 digits.';

    }

    if (quantity < 1 || quantity > 3) {

      newErrors.quantity = 'Quantity must be between 1 and 3.';

    }

    if (!termsAccepted) {

      newErrors.terms = 'You must accept the terms and conditions.';

    }

    if (Object.keys(newErrors).length > 0) {

      setErrors(newErrors);

      return;

    }

    const order = {

      customerName,

      customerNumber,

      menuItem: menuItems.find(item => item.id === selectedItem).name,

      itemSize,

      quantity,

      toppings,

      specialInstructions,

      orderType,

      totalOrder: quantity,

    };

    setOrderDetails(order);

    setErrors({});

  };

  const handleReset = () => {

    setCustomerName('');

    setCustomerNumber('');

    setSelectedItem('1');

    setItemSize('Medium');

    setQuantity(1);

    setSpecialInstructions('');

    setToppings([]);

    setOrderType('pickup');

    setTermsAccepted(false);

    setOrderDetails(null);

    setErrors({});

  };

  const handleToppingsChange = (e) => {

    const { value, checked } = e.target;

    if (checked) {

      setToppings([...toppings, value]);

    } else {

      setToppings(toppings.filter(topping => topping !== value));

    }

  };

  return (

    <div className="App">

      <header className="App-header">

        <h1>Restaurant Page</h1>

      </header>

      <main className="container">

        <section id="order">

          <div className="form-container">

            <h1>Restaurant Order Form</h1>

            <form onSubmit={handleSubmit}>

              <label>

                Customer Name:

                <input

                  type="text"

                  value={customerName}

                  onChange={(e) => setCustomerName(e.target.value)}

                />

              </label>

              <label>

                Customer Number:

                <input

                  type="text"

                  value={customerNumber}

                  onChange={(e) => setCustomerNumber(e.target.value)}

                  maxLength="10"

                />

                {errors.customerNumber && (

                  <p className="error">{errors.customerNumber}</p>

                )}

              </label>

              <label>

                Menu Item:

                <select

                  value={selectedItem}

                  onChange={(e) => setSelectedItem(e.target.value)}

                >

                  {menuItems.map(item => (

                    <option key={item.id} value={item.id}>

                      {item.name}

                    </option>

                  ))}

                </select>

              </label>

              <label>

                Size:

                <select

                  value={itemSize}

                  onChange={(e) => setItemSize(e.target.value)}

                >

                  <option value="Small">Small</option>

                  <option value="Medium">Medium</option>

                  <option value="Large">Large</option>

                </select>

              </label>

              <label>

                Quantity:

                <input

                  type="number"

                  value={quantity}

                  min="1"

                  max="3"

                  onChange={(e) => setQuantity(Math.min(3, Math.max(1, e.target.value)))}

                />

                {errors.quantity && (

                  <p className="error">{errors.quantity}</p>

                )}

              </label>

              <label>

                Extra Toppings:

                <div className="toppings">

                  <label>

                    <input

                      type="checkbox"

                      value="Cheese"

                      checked={toppings.includes('Cheese')}

                      onChange={handleToppingsChange}

                    />

                    Cheese

                  </label>

                  <label>

                    <input

                      type="checkbox"

                      value="Pepperoni"

                      checked={toppings.includes('Pepperoni')}

                      onChange={handleToppingsChange}

                    />

                    Pepperoni

                  </label>

                  <label>

                    <input

                      type="checkbox"

                      value="Mushrooms"

                      checked={toppings.includes('Mushrooms')}

                      onChange={handleToppingsChange}

                    />

                    Mushrooms

                  </label>

                </div>

              </label>

              <label>

                Order Type:

                <div className="order-type">

                  <label>

                    <input

                      type="radio"

                      value="pickup"

                      checked={orderType === 'pickup'}

                      onChange={(e) => setOrderType(e.target.value)}

                    />

                    Pickup

                  </label>

                  <label>

                    <input

                      type="radio"

                      value="delivery"

                      checked={orderType === 'delivery'}

                      onChange={(e) => setOrderType(e.target.value)}

                    />

                    Delivery

                  </label>

                </div>

              </label>

              <label>

                Special Instructions:

                <textarea

                  value={specialInstructions}

                  onChange={(e) => setSpecialInstructions(e.target.value)}

                />

              </label>

              <label>

                <input

                  type="checkbox"

                  checked={termsAccepted}

                  onChange={(e) => setTermsAccepted(e.target.checked)}

                />

                I accept the terms and conditions

              </label>

              {errors.terms && (

                <p className="error">{errors.terms}</p>

              )}

              <button type="submit">Order Now</button>

              <button type="button" onClick={handleReset}>Reset</button>

            </form>

            {orderDetails && (

              <div className="order-summary-modal">

                <div className="modal-content">

                  <h2>Order Details</h2>

                  <p>Customer Name: {orderDetails.customerName}</p>

                  <p>Customer Number: {orderDetails.customerNumber}</p>

                  <p>Menu Item: {orderDetails.menuItem} - {orderDetails.itemSize}</p>

                  <p>Quantity: {orderDetails.quantity}</p>

                  <p>Toppings: {orderDetails.toppings.join(', ')}</p>

                  <p>Order Type: {orderDetails.orderType}</p>

                  <p>Special Instructions: {orderDetails.specialInstructions}</p>

                  <p>Total Order: {orderDetails.totalOrder}</p>

                </div>

              </div>

            )}

          </div>

        </section>

        <section id="gallery">

          <h2>Popular Dishes</h2>

          <div className="gallery">

            {images.map((src, index) => (

              <img key={index} src={src} alt={`Dish ${index + 1}`} />

            ))}

          </div>

        </section>

      </main>

      <footer>

        <p>&copy; 2024 Restaurant</p>

      </footer>

    </div>

  );

};

export default App;

App.css

body {

font-family: 'Lato', sans-serif;

background-color: #1e1e2f;

color: #e4e4f5;

margin: 0;

padding: 0;

}

.App-header {

background-color: #26264f;

color: #f0f0ff;

padding: 20px;

text-align: center;

border-bottom: 5px solid #8e44ad;

}

.container {

padding: 40px;

max-width: 1100px;

margin: 0 auto;

background-color: #2a2a4d;

border-radius: 15px;

box-shadow: 0 12px 25px rgba(0, 0, 0, 0.3);

}

nav ul {

list-style: none;

padding: 0;

display: flex;

justify-content: center;

margin-bottom: 35px;

}

nav ul li {

margin: 0 30px;

}

nav ul li a {

text-decoration: none;

color: #9b59b6;

font-weight: bold;

font-size: 1.2rem;

transition: color 0.3s ease;

}

nav ul li a:hover {

color: #c39bd3;

}

.form-container {

background-color: #343457;

border-radius: 15px;

padding: 35px;

box-shadow: 0 8px 20px rgba(0, 0, 0, 0.2);

transition: transform 0.3s ease;

}

.form-container:hover {

transform: translateY(-6px);

}

.form-container h1 {

margin-top: 0;

margin-bottom: 30px;

color: #f0f0ff;

font-size: 2rem;

text-align: center;

}

label {

display: block;

margin-bottom: 15px;

font-weight: bold;

color: #dcdde1;

}

input[type="text"],

input[type="number"],

select,

textarea {

width: 100%;

padding: 15px;

margin-top: 6px;

margin-bottom: 25px;

border: 2px solid #6c5ce7;

background-color: #2c2c4e;

color: #f0f0ff;

border-radius: 6px;

box-sizing: border-box;

transition: border-color 0.3s ease;

}

input[type="text"]:focus,

input[type="number"]:focus,

select:focus,

textarea:focus {

border-color: #9b59b6;

}

input[type="checkbox"],

input[type="radio"] {

margin-right: 10px;

}

textarea {

height: 120px;

}

button[type="submit"],

button[type="button"] {

padding: 14px 28px;

background-color: #8e44ad;

color: #f0f0ff;

border: none;

border-radius: 8px;

cursor: pointer;

font-size: 1rem;

transition: background-color 0.3s ease, transform 0.3s ease;

margin-top: 20px;

}

button[type="button"] {

background-color: #575772;

}

button:hover {

background-color: #9b59b6;

transform: translateY(-4px);

}

button[type="button"]:hover {

background-color: #7f7f99;

}

.error {

color: #e74c3c;

font-size: 0.875rem;

margin-top: -10px;

margin-bottom: 20px;

}

.gallery {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(230px, 1fr));

gap: 25px;

margin-top: 25px;

}

.gallery img {

width: 100%;

height: auto;

border-radius: 12px;

box-shadow: 0 8px 15px rgba(0, 0, 0, 0.2);

transition: transform 0.3s ease;

}

.gallery img:hover {

transform: scale(1.08);

}

.order-summary-modal {

position: fixed;

top: 0;

left: 0;

width: 100%;

height: 100%;

background-color: rgba(0, 0, 0, 0.7);

display: flex;

align-items: center;

justify-content: center;

}

.modal-content {

background-color: #2a2a4d;

padding: 35px;

border-radius: 12px;

max-width: 550px;

width: 100%;

box-shadow: 0 12px 25px rgba(0, 0, 0, 0.4);

}

.modal-content h2 {

margin-top: 0;

color: #f0f0ff;

font-size: 1.7rem;

}

footer {

background-color: #26264f;

color: #ecf0f1;

text-align: center;

padding: 20px;

position: fixed;

bottom: 0;

width: 100%;

box-shadow: 0 -5px 12px rgba(0, 0, 0, 0.3);

font-size: 0.875rem;

}

Output:



