**Restaurant**

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;

body {

  font-family: Arial, sans-serif;

  background-color: #ffffff;

  color: #333333;

  margin: 0;

  padding: 0;

}

.App-header {

  background-color: #b22222;

  color: #ffffff;

  padding: 20px;

  text-align: center;

  border-bottom: 4px solid #8b0000;

}

.container {

  padding: 20px;

  max-width: 800px;

  margin: 20px auto;

  background-color: #f8f8f8;

  border-radius: 8px;

  box-shadow: 0 5px 10px rgba(0, 0, 0, 0.1);

}

nav ul {

  list-style: none;

  padding: 0;

  display: flex;

  justify-content: space-around;

  margin-bottom: 20px;

  background-color: #e74c3c;

  border-radius: 8px;

  overflow: hidden;

}

nav ul li {

  flex: 1;

  text-align: center;

}

nav ul li a {

  display: block;

  padding: 15px;

  text-decoration: none;

  color: #ffffff;

  font-weight: bold;

  transition: background-color 0.3s ease;

}

nav ul li a:hover {

  background-color: #8b0000;

}

label {

  display: block;

  margin-bottom: 5px;

  font-weight: bold;

  color: #333333;

}

input[type="text"],

input[type="number"],

select,

textarea {

  width: 100%;

  padding: 10px;

  margin-top: 5px;

  margin-bottom: 15px;

  border: 1px solid #cccccc;

  border-radius: 4px;

  box-sizing: border-box;

}

input[type="text"]:focus,

input[type="number"]:focus,

select:focus,

textarea:focus {

  border-color: #b22222;

}

button[type="submit"],

button[type="button"] {

  padding: 10px 20px;

  background-color: #b22222;

  color: white;

  border: none;

  border-radius: 4px;

  cursor: pointer;

  transition: background-color 0.3s ease;

  margin-top: 10px;

}

button[type="button"] {

  background-color: #808080;

}

button:hover {

  background-color: #8b0000;

}

button[type="button"]:hover {

  background-color: #696969;

}

footer {

  background-color: #b22222;

  color: #ffffff;

  text-align: center;

  padding: 15px;

  position: fixed;

  bottom: 0;

  width: 100%;

  font-size: 0.8rem;

}



