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">

{/\* <nav>

<ul>

<li><a href="#order">Order Now</a></li>

<li><a href="#gallery">Gallery</a></li>

</ul>

</nav> \*/}

<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} 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

/\* App.css \*/

body {

font-family: Arial, sans-serif;

background-color: #f8f9fa;

margin: 0;

padding: 0;

}

.App-header {

background-color: #343a40;

color: white;

padding: 20px;

text-align: center;

}

.container {

padding: 20px;

max-width: 1200px;

margin: 0 auto;

}

nav ul {

list-style: none;

padding: 0;

display: flex;

justify-content: center;

margin-bottom: 20px;

}

nav ul li {

margin: 0 15px;

}

nav ul li a {

text-decoration: none;

color: #007bff;

font-weight: bold;

}

nav ul li a:hover {

text-decoration: underline;

}

.form-container {

background-color: white;

border: 1px solid #ced4da;

border-radius: 5px;

padding: 20px;

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

}

.form-container h1 {

margin-top: 0;

margin-bottom: 20px;

color: #343a40;

}

label {

display: block;

margin-bottom: 10px;

font-weight: bold;

}

input[type="text"],

input[type="number"],

select,

textarea {

width: 100%;

padding: 10px;

margin-top: 5px;

margin-bottom: 15px;

border: 1px solid #ced4da;

border-radius: 4px;

box-sizing: border-box;

}

input[type="checkbox"],

input[type="radio"] {

margin-right: 10px;

}

input[type="checkbox"]:not(:last-child),

input[type="radio"]:not(:last-child) {

margin-bottom: 10px;

}

textarea {

height: 80px;

}

button[type="submit"],

button[type="button"] {

padding: 10px 20px;

background-color: #007bff;

color: white;

border: none;

border-radius: 5px;

cursor: pointer;

margin-right: 10px;

margin-top: 15px;

}

button[type="button"] {

background-color: #6c757d;

}

button:hover {

background-color: #0056b3;

}

button[type="button"]:hover {

background-color: #5a6268;

}

.error {

color: red;

font-size: 0.875rem;

margin-top: -10px;

margin-bottom: 10px;

}

.gallery {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));

gap: 15px;

}

.gallery img {

width: 100%;

height: auto;

border-radius: 5px;

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

}

.order-summary-modal {

position: fixed;

top: 0;

left: 0;

width: 100%;

height: 100%;

background-color: rgba(0, 0, 0, 0.5);

display: flex;

align-items: center;

justify-content: center;

}

.modal-content {

background-color: white;

padding: 20px;

border-radius: 5px;

max-width: 500px;

width: 100%;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);

}

.modal-content h2 {

margin-top: 0;

color: #343a40;

}

footer {

background-color: #343a40;

color: white;

text-align: center;

padding: 15px;

position: fixed;

bottom: 0;

width: 100%;

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

}

Output:

