Fragment là gì

Fragment là một common pattern được giới thiệu kể từ khi React 16 ra đời. Nó cho phép bạn return nhiều element từ một component mà không làm sinh ra những DOM element ko cần thiết

**Sự ra đời**

Dưới đây là một common pattern của một React component trả về một list các children:

class Table extends React.Component {

render() {

return (

<table>

<tr>

<Columns />

</tr>

</table>

);

}

}

<Columns /> component sẽ phải trả về nhiều phần tử <td> để đầu ra HTML hiển thị hợp lệ. Nếu một div cha được sử dụng bên trong render() của <Columns /> component, thì sẽ cho ra kết quả HTML không hợp lệ.

<Table /> component lúc này sẽ có dạng:

<table>

<tr>

<div>

<td>Hello</td>

<td>World</td>

</div>

</tr>

</table>

Chính vì thế, fragment được giới thiệu để giải quyết điều này.

**Cách sử dụng**

Đơn giản, các bạn chỉ cần wrap list các children bởi <React.Fragment>

class Columns extends React.Component {

render() {

return (

<React.Fragment>

<td>Hello</td>

<td>World</td>

</React.Fragment>

);

}

}

Sẽ cho ra kết quả là một <Table /> component chính xác.

<table>

<tr>

<td>Hello</td>

<td>World</td>

</tr>

</table>

**Short Syntax**

Có một cú pháp mới, ngắn hơn mà bạn có thể sử dụng để khai báo các fragment. Nó trông giống như các thể trống (empty tags)

class Columns extends React.Component {

render() {

return (

<>

<td>Hello</td>

<td>World</td>

</>

);

}

}

Bạn có thể sử dụng <> </> giống như cách bạn sẽ sử dụng bất kỳ phần tử nào khác ngoại trừ việc nó không hỗ trợ keys và attributes

Lưu ý rằng nhiều công cụ vẫn chưa hỗ trợ cú pháp n cày nên bạn được khuyên dùng <React.Fragment>.

Section 11: Practice Project: Building a Food Order App

Link All CSS to start

<https://github.com/academind/react-complete-guide-code/tree/11-practice-food-order-app/extra-files>

132. Staring Setup (Source structure in github)

Step 1: create new folder components/Cart, components/Layout, components/Meals, components/UI

133. Adding a ‘Header’ Component

Step 1: create new folder components/assets and paster image with name file meals.jpg



Step 2 create new file components/Layout/Header.js and Header.module.css (copy in github) in Layout

Then in Header.js

import { Fragment } from "react";

import mealsImage from "../assets/meals.jpg";

import classes from "./Header.module.css";

const Header = (props) => {

  return (

    <Fragment>

      <header className={classes.header}>

        <h1>ReactMeals</h1>

        <button>Cart</button>

      </header>

      <div className={classes["main-image"]}>

        <img src={mealsImage} alt="Atable full of delicious food!" />

      </div>

      <div></div>

    </Fragment>

  );

};

export default Header;

Step 3: in App.js change like this

import Header from "./components/Layout/Header";

function App() {

  return (

    <div>

      <Header />

    </div>

  );

}

export default App;

134. Adding the ‘Cart’ Button Component

Step 1: add HeaderCartButton.js in folder Layout

import CartIcon from "../Cart/CartIcon";

import React from "react";

import classes from './HeaderCartButton.module.css';

const HeaderCartButton = props => {

    return <button className={classes.button}>

        <span>

            <CartIcon />

        </span>

        <span>Your Cart</span>

        <span className={classes.badge}>

            3

        </span>

    </button>

};

export default HeaderCartButton;

Step 2: add HeaderCartButton.module.css (link in github)

Step 3: add CartIcon.js in folder Cart

const CartIcon = () => {

    return (

      <svg

        xmlns='http://www.w3.org/2000/svg'

        viewBox='0 0 20 20'

        fill='currentColor'

      >

        <path d='M3 1a1 1 0 000 2h1.22l.305 1.222a.997.997 0 00.01.042l1.358 5.43-.893.892C3.74 11.846 4.632 14 6.414 14H15a1 1 0 000-2H6.414l1-1H14a1 1 0 00.894-.553l3-6A1 1 0 0017 3H6.28l-.31-1.243A1 1 0 005 1H3zM16 16.5a1.5 1.5 0 11-3 0 1.5 1.5 0 013 0zM6.5 18a1.5 1.5 0 100-3 1.5 1.5 0 000 3z' />

      </svg>

    );

  };

  export default CartIcon;

Step 4: then in Header.js change button to HeaderCartButton

import { Fragment } from "react";

import mealsImage from "../assets/meals.jpg";

import classes from "./Header.module.css";

import HeaderCartButton from "./HeaderCartButton";

const Header = (props) => {

  return (

    <Fragment>

      <header className={classes.header}>

        <h1>ReactMeals</h1>

        <HeaderCartButton>Cart</HeaderCartButton>

      </header>

      <div className={classes["main-image"]}>

        <img src={mealsImage} alt="Atable full of delicious food!" />

      </div>

      <div></div>

    </Fragment>

  );

};

export default Header;

135. Adding a “Meals” Components

Step 1 Add MealsSummary.module.css in folder Meals (copy css in github)

Step 2: Add MealsSummary.js in folder Meals

import React from 'react';

import classes from './MealsSummary.module.css';

const MealsSummary = () => {

    return <section className={classes.summary}>

        <h2>Delicious food, Delivered To You</h2>

        <p>

            Choose your favorite meal from our boad selection of available meals

            and enjoy a delicious lunch or dinner at home

        </p>

        <p>

            All our meals are cooked with high-quality ingredients, just-in-time and of course by

            experienced chefs

        </p>

    </section>

}

export default MealsSummary;

Step 3: Add AvailableMeals.module.css in folder Meals (copy in github)

Step 4: Add AvailableMeals.js in folder Meals

import classes from "./AvailableMeals.module.css";

import React from "react";

const DUMMY\_MEALS = [

    {

      id: 'm1',

      name: 'Sushi',

      description: 'Finest fish and veggies',

      price: 22.99,

    },

    {

      id: 'm2',

      name: 'Schnitzel',

      description: 'A german specialty!',

      price: 16.5,

    },

    {

      id: 'm3',

      name: 'Barbecue Burger',

      description: 'American, raw, meaty',

      price: 12.99,

    },

    {

      id: 'm4',

      name: 'Green Bowl',

      description: 'Healthy...and green...',

      price: 18.99,

    },

  ];

const AvailableMeals = () => {

    const mealsList = DUMMY\_MEALS.map(meal => <li>{meal.name}</li>);

    return <section className={classes.meals}>

        <ul>

            {mealsList}

        </ul>

    </section>

};

export default AvailableMeals;

Step 5: change Meals.js like this

import MealsSummary from "./MealsSummary";

import AvailableMeals from "./AvailableMeals";

import React from "react";

import { Fragment } from "react";

const Meals = () => {

  return (

    <Fragment>

      <MealsSummary />

      <AvailableMeals />

    </Fragment>

  );

};

export default Meals;

Step 6: Change App.js

import { Fragment } from "react";

import Header from "./components/Layout/Header";

import Meals from "./components/Meals/Meals";

function App() {

  return (

    <Fragment>

      <Header />

      <main>

        <Meals />

      </main>

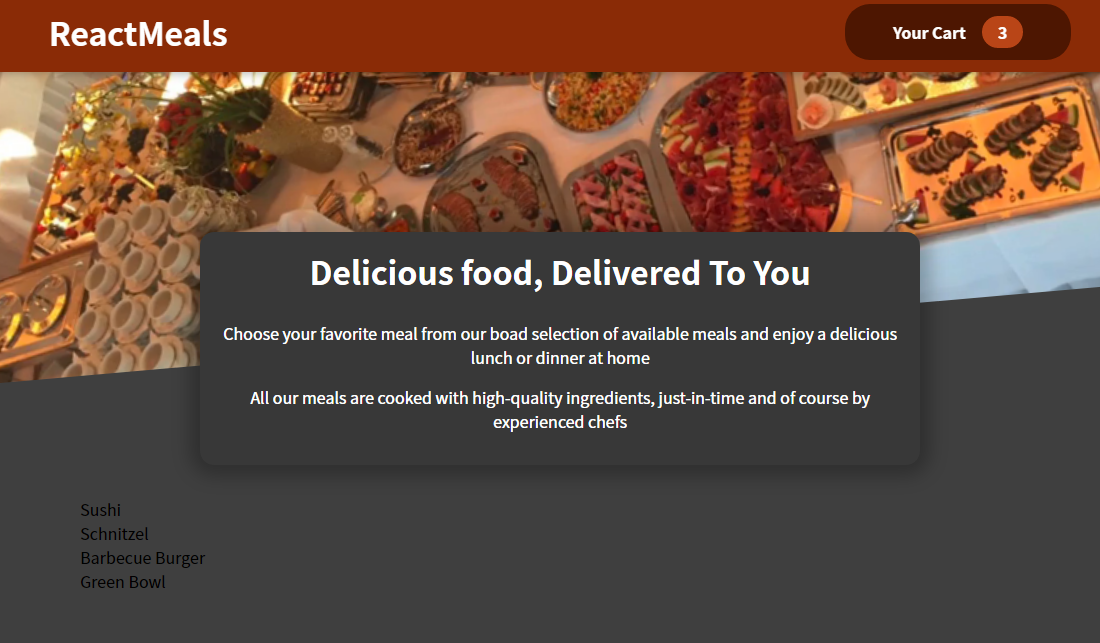
    </Fragment>

  );

}

export default App;

Step 7: Check result



136. Adding Individual Meal Item & Displaying Them

Step 0: add new file Card.module.css in folder UI (copy in github)

Step 1: add new file Card.js in folder UI

import classes from './Card.module.css';

const Card = props => {

    return <div className={classes.card}>

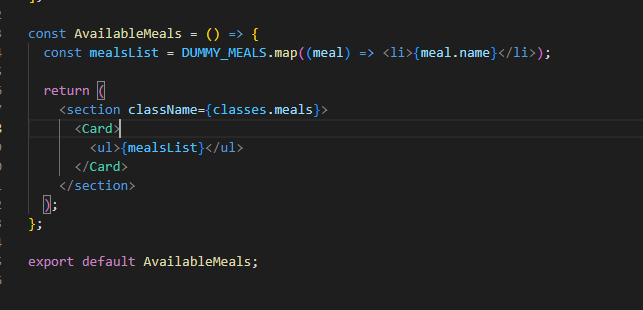
        {props.children}

    </div>

};

export default Card;

Step 2: then in AvailableMeals.js add like this (remember import Card)



Step 3: add subfolder MealItem in folder Meals

import React from "react";

import classes from './MealItem.module.css';

import MealItemForm from "./MealItemForm";

const MealItem = (props) => {

    const price = `$${props.price.toFixed(2)}`;

  return (

    <li className={classes.meal}>

      <div>

        <h3>{props.name}</h3>

        <div className={classes.description}>{props.description}</div>

        <div className={classes.price}>{price}</div>

      </div>

      <div>

            <MealItemForm />

      </div>

    </li>

  );

};

export default MealItem;

Step 4: add MealItem.module.css

.meal {

    display: flex;

    justify-content: space-between;

    margin: 1rem;

    padding-bottom: 1rem;

    border-bottom: 1px solid #ccc;

  }

  .meal h3 {

    margin: 0 0 0.25rem 0;

  }

  .description {

    font-style: italic;

  }

  .price {

    margin-top: 0.25rem;

    font-weight: bold;

    color: #ad5502;

    font-size: 1.25rem;

  }

Step 5 add MealItemForm.js

import React from "react";

const MealItemForm = () => {

    return <></>;

}

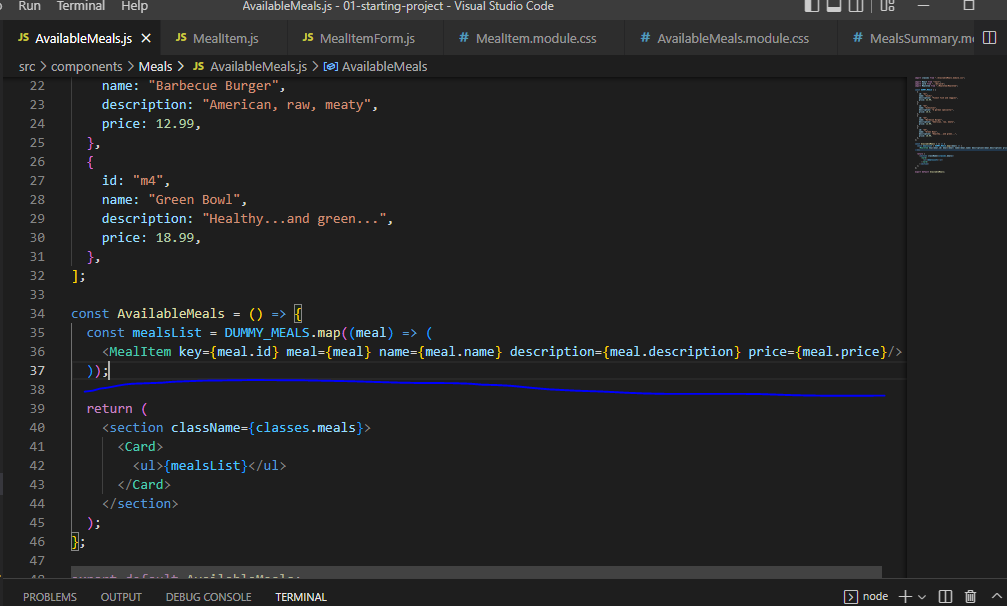
export default MealItemForm;

Step 6: in AvailableMeals.js add like this

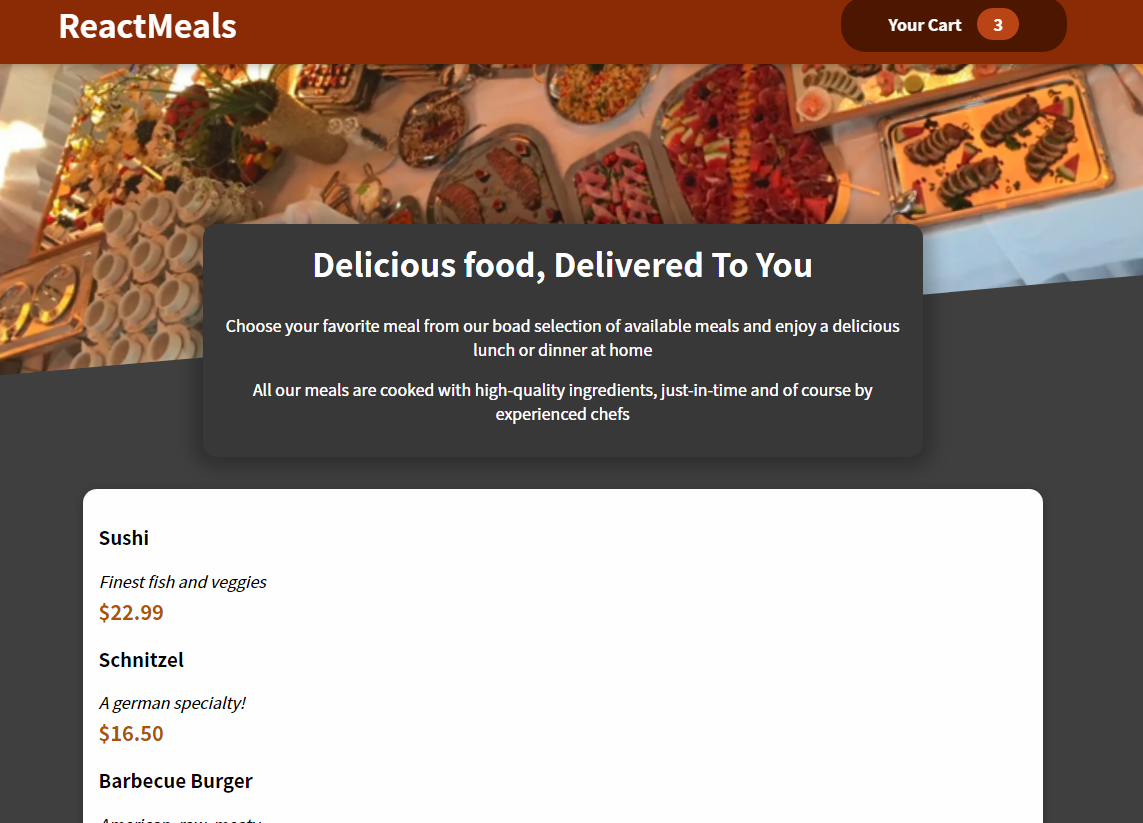
  const mealsList = DUMMY\_MEALS.map((meal) => (

    <MealItem key={meal.id} meal={meal} name={meal.name} description={meal.description} price={meal.price}/>

  ));



Step 7: result



137. Adding a Form

Step 1: create Input.js in folder UI

import classes from './Input.module.css';

const Input = (props) => {

  return (

    <div className={classes.input}>

      <label htmlFor={props.input.id}>{props.label}</label>

      <input {...props.input} />

    </div>

  );

};

export default Input;

Step 2: create Input.module.css in folder UI

.input {

    display: flex;

    align-items: center;

    margin-bottom: 0.5rem;

  }

  .input label {

    font-weight: bold;

    margin-right: 1rem;

  }

  .input input {

    width: 3rem;

    border-radius: 5px;

    border: 1px solid #ccc;

    font: inherit;

    padding-left: 0.5rem;

  }

Step3: then create new MealItemForm.js in folder MealItem

import Input from '../../UI/Input';

import classes from './MealItemForm.module.css';

const MealItemForm = (props) => {

  return (

    <form className={classes.form}>

      <Input

        label='Amount'

        input={{

          id: 'amount\_' + props.id,

          type: 'number',

          min: '1',

          max: '5',

          step: '1',

          defaultValue: '1',

        }}

      />

      <button>+ Add</button>

    </form>

  );

};

export default MealItemForm;

Step 4: create MealItemForm.module.css

.form {

    text-align: right;

  }

  .form button {

    font: inherit;

    cursor: pointer;

    background-color: #8a2b06;

    border: 1px solid #8a2b06;

    color: white;

    padding: 0.25rem 2rem;

    border-radius: 20px;

    font-weight: bold;

  }

  .form button:hover,

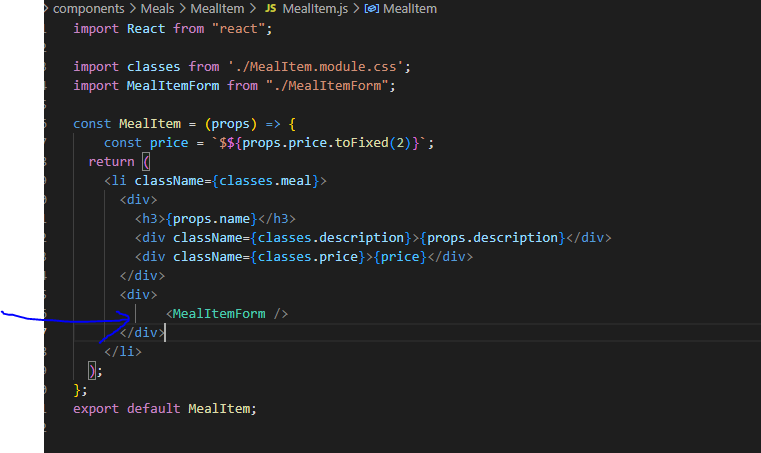
  .form button:active {

    background-color: #641e03;

    border-color: #641e03;

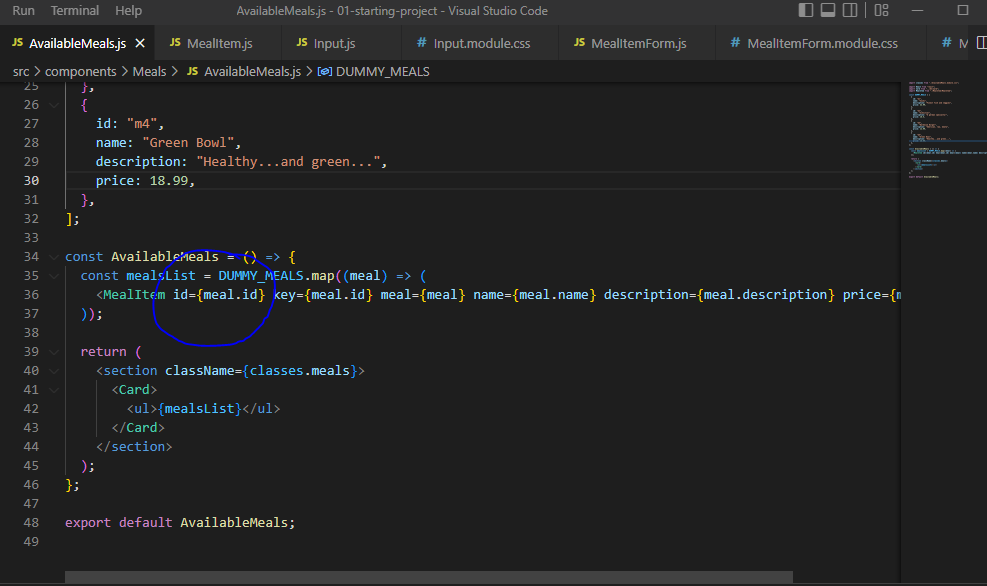
  }

Step 5: then in MealItem.js



138 Fixing Form Input IDs

Go to file AvailableMeals.js add like this



139. Working on the ‘Shoppping Cart’ Component

Step 1: add Cart.module.css in Cart (link in git)

Step 2: add Cart.js in folder Cart

import classes from "./Cart.module.css";

const Cart = (props) => {

  const cartItems = (

    <ul className={classes["cart-item"]}>

      {[{ id: "c1", name: "Sushi", amount: 2, price: 12.99 }].map((item) => (

        <li>{item.name}</li>

      ))}

    </ul>

  );

  return (

    <div>

      {cartItems}

      <div className={classes.total}>

        <span>Total Amount</span>

        <span>35.62</span>

      </div>

      <div className={classes.actions}>

            <button className={classes['button--alt']}>Close</button>

            <button className={classes.button}>Order</button>

      </div>

    </div>

  );

};

export default Cart;

140. Adding a Modal via a React Portal

Step 1: add Modal.module.css in folder UI (link in github)

Step 2: Add Modal.js

import React, { Fragment } from 'react';

import classes from './Modal.module.css';

import ReactDOM from 'react-dom';

const Backdrop = props => {

    return <div className={classes.backdrop}/>

};

const ModalOverlay = (props) => {

    return (

        <div className={classes.modal}>

            <div className={classes.content}>{props.children}</div>

        </div>

    )

}

const portalElement = document.getElementById('overlays');

const Modal = props => {

    return <Fragment>

        {ReactDOM.createPortal(<Backdrop /> , portalElement)}

        {ReactDOM.createPortal(

            <ModalOverlay>{props.children}</ModalOverlay>,

            portalElement

        )}

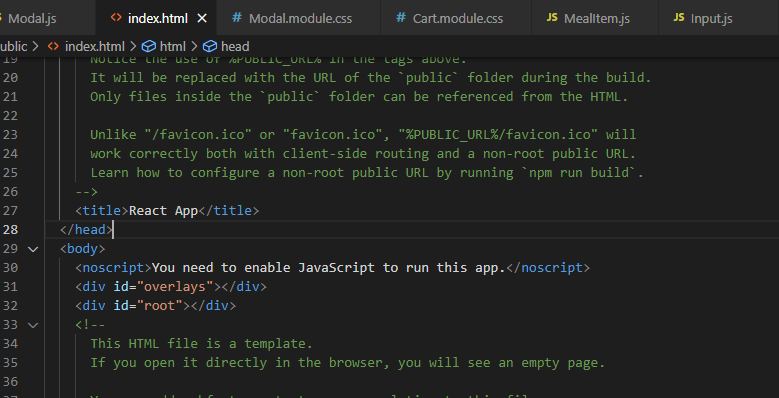
        <Backdrop />

    </Fragment>

};

export default Modal;

Step 3: then in index.html



Step 4: change Modal.js like this

import React, { Fragment } from "react";

import classes from "./Modal.module.css";

import ReactDOM from "react-dom";

const Backdrop = (props) => {

  return <div className={classes.backdrop} />;

};

const ModalOverlay = (props) => {

  return (

    <div className={classes.modal}>

      <div className={classes.content}>{props.children}</div>

    </div>

  );

};

const portalElement = document.getElementById("overlays");

const Modal = (props) => {

  return (

    <Fragment>

      {ReactDOM.createPortal(<Backdrop />, portalElement)}

      {ReactDOM.createPortal(

        <ModalOverlay>{props.children}</ModalOverlay>,

        portalElement

      )}

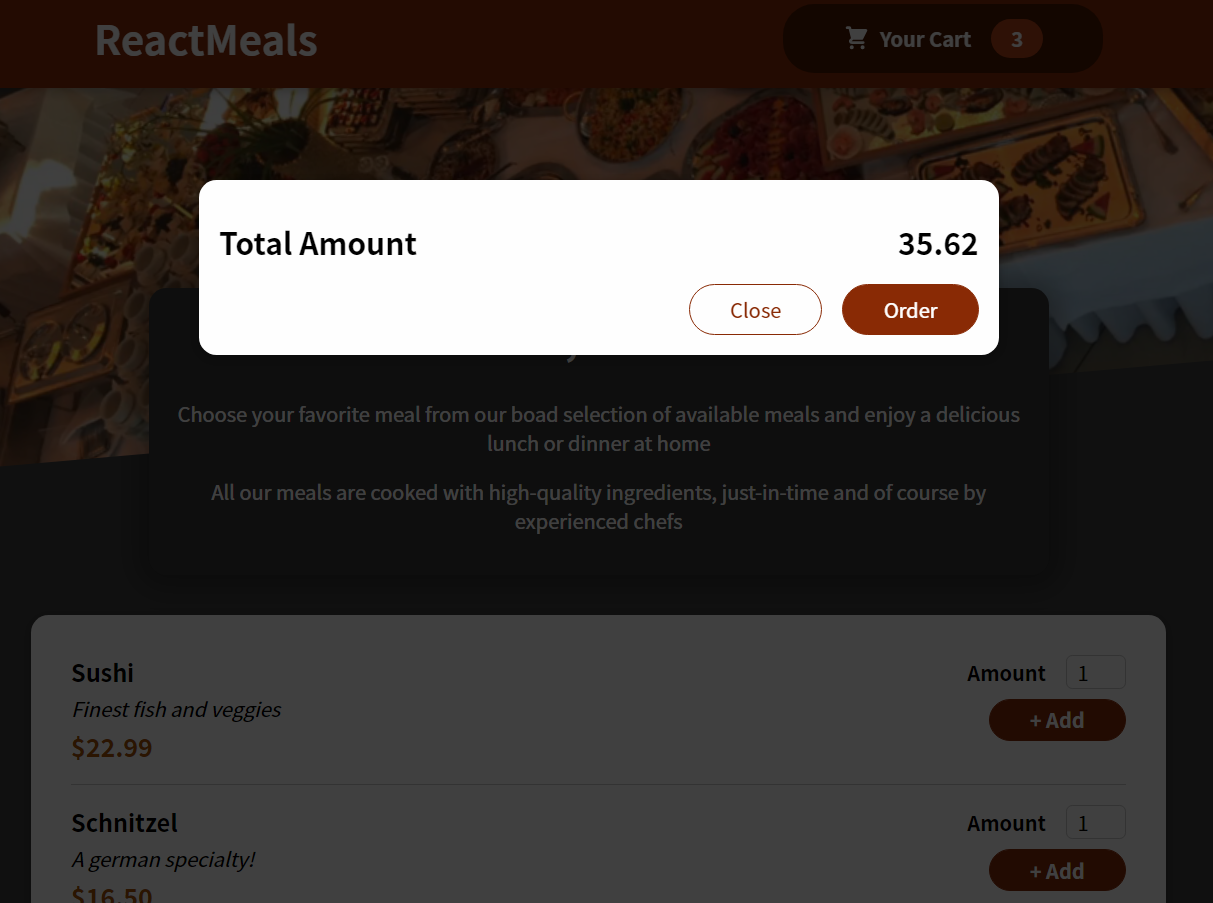
    </Fragment>

  );

};

export default Modal;

step 5: check result



141. Managing Cart & Modal State

Step 1: Change app.js like this

import { Fragment, useState } from 'react';

import Cart from './components/Cart/Cart';

import Header from './components/Layout/Header';

import Meals from './components/Meals/Meals';

function App() {

  const [cartIsShown, setCartIsShown] = useState(false);

  const showCartHandler = () => {

    setCartIsShown(true);

  }

  const hideCartHandler = () => {

    setCartIsShown(false);

  }

  return (

    <Fragment>

      {cartIsShown && <Cart onClose={hideCartHandler}/>}

      <Header onShowCart={showCartHandler}/>

      <main>

        <Meals />

      </main>

    </Fragment>

  );

}

export default App;

Step 2: in Header.js

import { Fragment } from 'react';

import HeaderCartButton from './HeaderCartButton';

import mealsImage from '../assets/meals.jpg';

import classes from './Header.module.css';

const Header = (props) => {

  return (

    <Fragment>

      <header className={classes.header}>

        <h1>ReactMeals</h1>

        <HeaderCartButton onClick={props.onShowCart}/>

      </header>

      <div className={classes['main-image']}>

        <img src={mealsImage} alt='A table full of delicious food!' />

      </div>

    </Fragment>

  );

};

export default Header;

Step 3: in HeaderCartButton

import CartIcon from '../Cart/CartIcon';

import classes from './HeaderCartButton.module.css';

const HeaderCartButton = (props) => {

  return (

    <button className={classes.button} onClick={props.onClick}>

      <span className={classes.icon}>

        <CartIcon />

      </span>

      <span>Your Cart</span>

      <span className={classes.badge}>3</span>

    </button>

  );

};

export default HeaderCartButton;

Step 4 in Cart.js

import Modal from '../UI/Modal';

import classes from './Cart.module.css';

const Cart = (props) => {

  const cartItems = (

    <ul className={classes['cart-items']}>

      {/\* {[{ id: 'c1', name: 'Sushi', amount: 2, price: 12.99 }].map((item) => (

        <li>{item.name}</li>

      ))} \*/}

    </ul>

  );

  return (

    <Modal onClose={props.onClose}>

      {cartItems}

      <div className={classes.total}>

        <span>Total Amount</span>

        <span>35.62</span>

      </div>

      <div className={classes.actions}>

        <button className={classes['button--alt']} onClick={props.onClose}>Close</button>

        <button className={classes.button}>Order</button>

      </div>

    </Modal>

  );

};

export default Cart;

Step 5 Then in Modal

import React, { Fragment } from "react";

import classes from "./Modal.module.css";

import ReactDOM from "react-dom";

const Backdrop = (props) => {

  return <div className={classes.backdrop} onClick={props.onClose}/>;

};

const ModalOverlay = (props) => {

  return (

    <div className={classes.modal}>

      <div className={classes.content}>{props.children}</div>

    </div>

  );

};

const portalElement = document.getElementById("overlays");

const Modal = (props) => {

  return (

    <Fragment>

      {ReactDOM.createPortal(<Backdrop onCLose={props.onClose}/>, portalElement)}

      {ReactDOM.createPortal(

        <ModalOverlay>{props.children}</ModalOverlay>,

        portalElement

      )}

    </Fragment>

  );

};

export default Modal;

Step 6 check popup is close or not

142. Adding a Cart Context

Step 1: create new folder components/store

Step 2: create new cart-context.js

import React from "react"

const CartContext = React.createContext({

    item: [],

    totalAmount: 0,

    addItem: (item) => {},

    removeItem: (id) => {}

})

export default CartContext;

step 3: create CartProvider.js in folder store

import CartContext from "./cart-context";

const CartProvider = props => {

    const addItemToCartHandler = item => {};

    const removeItemFromCartHandler = id => {};

    const cartContext = {

        items: [],

        totalAmount: 0,

        addItem: addItemToCartHandler,

        removeItem: removeItemFromCartHandler

    }

    return <CartContext.Provider value={cartContext}>

        {props.children}

    </CartContext.Provider>

};

export default CartProvider;

step 4: change App.js like this

import { useState } from 'react';

import Cart from './components/Cart/Cart';

import Header from './components/Layout/Header';

import Meals from './components/Meals/Meals';

import CartProvider from './components/store/CartProvider';

function App() {

  const [cartIsShown, setCartIsShown] = useState(false);

  const showCartHandler = () => {

    setCartIsShown(true);

  }

  const hideCartHandler = () => {

    setCartIsShown(false);

  }

  return (

    <CartProvider>

      {cartIsShown && <Cart onClose={hideCartHandler}/>}

      <Header onShowCart={showCartHandler}/>

      <main>

        <Meals />

      </main>

    </CartProvider>

  );

}

export default App;

143. Using the Context

Step 1: in HeaderCartButton.js, add new code

import { useContext } from "react";

import CartIcon from "../Cart/CartIcon";

import CartContext from "../store/cart-context";

import classes from "./HeaderCartButton.module.css";

const HeaderCartButton = (props) => {

  const cartCtx = useContext(CartContext);

  //reduce allow tranform data

  const numberOfCartItems = cartCtx.items.reduce((curNumber, item) => {

    return curNumber + item.amount;

  }, 0);

  return (

    <button className={classes.button} onClick={props.onClick}>

      <span className={classes.icon}>

        <CartIcon />

      </span>

      <span>Your Cart</span>

      <span className={classes.badge}>{numberOfCartItems}</span>

    </button>

  );

};

export default HeaderCartButton;

144. Adding a Cart Reducer

Step 1: in CartProvider.js add reducer

import { useReducer } from "react";

import CartContext from "./cart-context";

const defaultCartState = {

  items: [],

  totalAmount: 0,

};

const cartReducer = (state, action) => {

  if (action.type === "ADD") {

    const updatedItems = state.items.concat(action.item);

    const updatedTotalAmount = state.totalAmount + action.item.price \* action.item.amount;

    return {

        items: updatedItems,

        totalAmount: updatedTotalAmount

    };

  }

  return defaultCartState;

};

const CartProvider = (props) => {

  const [cartState, dispatchCartAction] = useReducer(

    cartReducer,

    defaultCartState

  );

  const addItemToCartHandler = (item) => {

    dispatchCartAction({

      type: "ADD",

      item: item,

    });

  };

  const removeItemFromCartHandler = (id) => {

    dispatchCartAction({ type: "REMOVE", id: id });

  };

  const cartContext = {

    items: cartState.items,

    totalAmount: cartState.totalAmount,

    addItem: addItemToCartHandler,

    removeItem: removeItemFromCartHandler,

  };

  return (

    <CartContext.Provider value={cartContext}>

      {props.children}

    </CartContext.Provider>

  );

};

export default CartProvider;

145. Working with Refs & Forward Refs

Step 1: add useRef in MealItemForm.js

import { useState } from "react";

import { useRef } from "react";

import Input from "../../UI/Input";

import classes from "./MealItemForm.module.css";

const MealItemForm = (props) => {

   const [amountIsValid, setAmountIsValid] =  useState(true);

  const amountInputRef = useRef();

  const submitHandler = (event) => {

    // prevent page fresher

    event.preventDefault();

    const enterAmount = amountInputRef.current.value;

    // convert number

    const enteredAmountNumber = +enterAmount;

    if (

      enterAmount.trim().length === 0 ||

      enteredAmountNumber < 1 ||

      enteredAmountNumber > 5

    ) {

        setAmountIsValid(false);

      return;

    }

    props.onAddToCart(enteredAmountNumber);

  };

  return (

    <form className={classes.form} onSubmit={submitHandler}>

      <Input

        ref={amountInputRef}

        label="Amount"

        input={{

          id: "amount\_" + props.id, // this changed!

          type: "number",

          min: "1",

          max: "5",

          step: "1",

          defaultValue: "1",

        }}

      />

      <button>+ Add</button>

      {!amountIsValid && <p>Please enter a valid amount (1-5).</p>}

    </form>

  );

};

export default MealItemForm;

Step 2: then in Input.js

import React from 'react';

import classes from './Input.module.css';

const Input = React.forwardRef((props, ref) => {

  return (

    <div className={classes.input}>

      <label htmlFor={props.input.id}>{props.label}</label>

      <input ref={ref} {...props.input} />

    </div>

  );

});

export default Input;

Step 3: in MealItem.js

import React, { useContext } from "react";

import CartContext from "../../store/cart-context";

import classes from './MealItem.module.css';

import MealItemForm from "./MealItemForm";

const MealItem = (props) => {

   const cartCtx =  useContext(CartContext);

    const price = `$${props.price.toFixed(2)}`;

    const addToCartHandler = amount => {

        cartCtx.addItem({

            id: props.id,

            name: props.name,

            amount: amount,

            price: props.price

        });

    };

  return (

    <li className={classes.meal}>

      <div>

        <h3>{props.name}</h3>

        <div className={classes.description}>{props.description}</div>

        <div className={classes.price}>{price}</div>

      </div>

      <div>

            <MealItemForm onAddToCart={addToCartHandler} />

      </div>

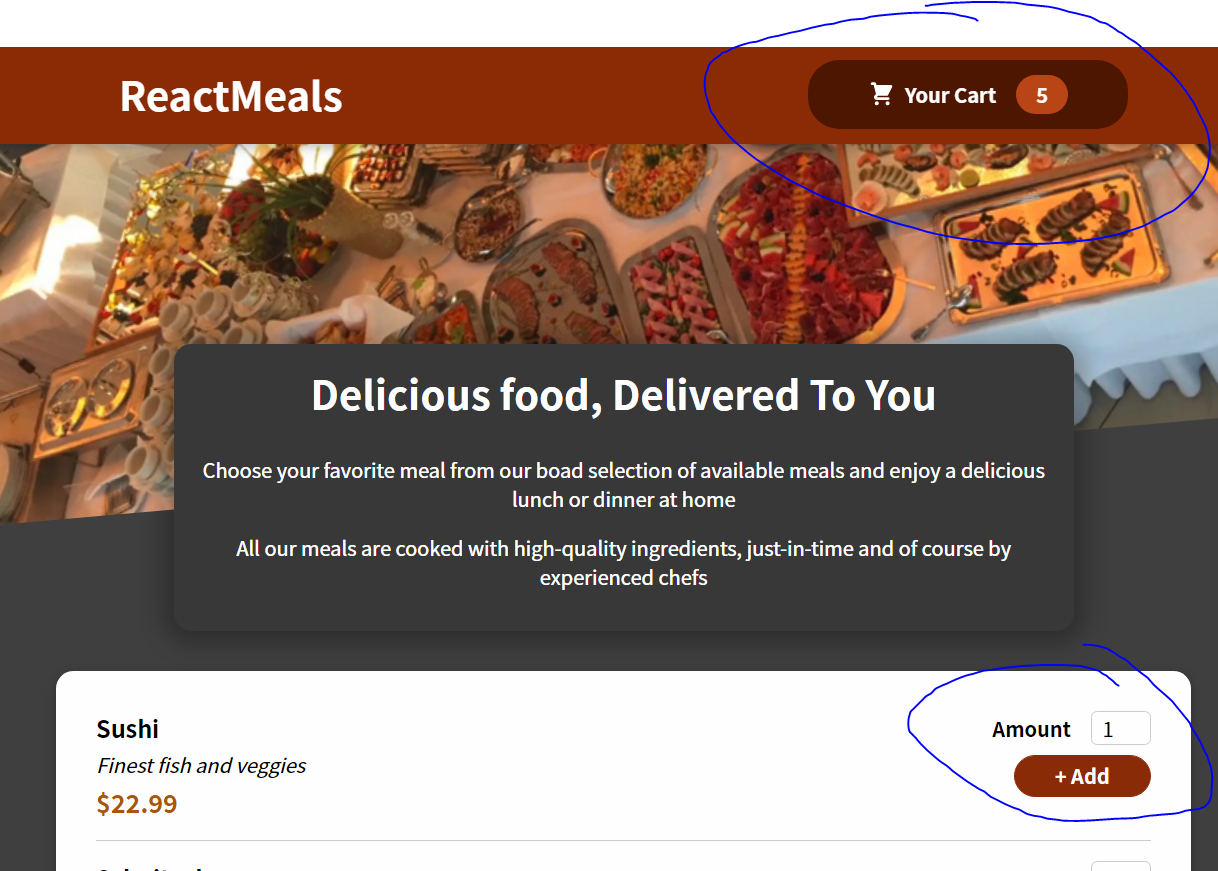
    </li>

  );

};

export default MealItem;

Step 4: check result



146. Outputting Cart Items

Step 0 add CartItem.js in fodler Cart

import classes from './CartItem.module.css';

const CartItem = (props) => {

  const price = `$${props.price.toFixed(2)}`;

  return (

    <li className={classes['cart-item']}>

      <div>

        <h2>{props.name}</h2>

        <div className={classes.summary}>

          <span className={classes.price}>{price}</span>

          <span className={classes.amount}>x {props.amount}</span>

        </div>

      </div>

      <div className={classes.actions}>

        <button onClick={props.onRemove}>−</button>

        <button onClick={props.onAdd}>+</button>

      </div>

    </li>

  );

};

export default CartItem;

Step 2 copy CartItem.module.css in git

.cart-item {

    display: flex;

    justify-content: space-between;

    align-items: center;

    border-bottom: 2px solid #8a2b06;

    padding: 1rem 0;

    margin: 1rem 0;

  }

  .cart-item h2 {

    margin: 0 0 0.5rem 0;

    color: #363636;

  }

  .summary {

    width: 10rem;

    display: flex;

    justify-content: space-between;

    align-items: center;

  }

  .price {

    font-weight: bold;

    color: #8a2b06;

  }

  .amount {

    font-weight: bold;

    border: 1px solid #ccc;

    padding: 0.25rem 0.75rem;

    border-radius: 6px;

    color: #363636;

  }

  .actions {

    display: flex;

    flex-direction: column;

  }

  @media (min-width: 768px) {

    .actions {

      flex-direction: row;

    }

  }

  .cart-item button {

    font: inherit;

    font-weight: bold;

    font-size: 1.25rem;

    color: #8a2b06;

    border: 1px solid #8a2b06;

    width: 3rem;

    text-align: center;

    border-radius: 6px;

    background-color: transparent;

    cursor: pointer;

    margin-left: 1rem;

    margin: 0.25rem;

  }

  .cart-item button:hover,

  .cart-item button:active {

    background-color: #8a2b06;

    color: white;

  }

Step 3 in Cart.js

import { useContext } from "react";

import CartContext from "../store/cart-context";

import Modal from "../UI/Modal";

import classes from "./Cart.module.css";

import CartItem from "./CartItem";

const Cart = (props) => {

  const cartCtx = useContext(CartContext);

  const totalAmount = `$${cartCtx.totalAmount.toFixed(2)}`;

  const hasItems = cartCtx.items.length > 2;

  const cartItemRemoveHandler = (id) => {};

  const cartItemAddhandler = (item) => {};

  const cartItems = (

    <ul className={classes["cart-items"]}>

      {cartCtx.items.map((item) => (

        <CartItem

          key={item.id}

          name={item.name}

          amount={item.amount}

          price={item.price}

          onRemove={cartItemRemoveHandler.bind(null, item.id)}

            onAdd={cartItemAddhandler.bind(null, item)}

        />

      ))}

    </ul>

  );

  return (

    <Modal onClose={props.onClose}>

      {cartItems}

      <div className={classes.total}>

        <span>Total Amount</span>

        <span>{totalAmount}</span>

      </div>

      <div className={classes.actions}>

        <button className={classes["button--alt"]} onClick={props.onClose}>

          Close

        </button>

        {hasItems && <button className={classes.button}>Order</button>}

      </div>

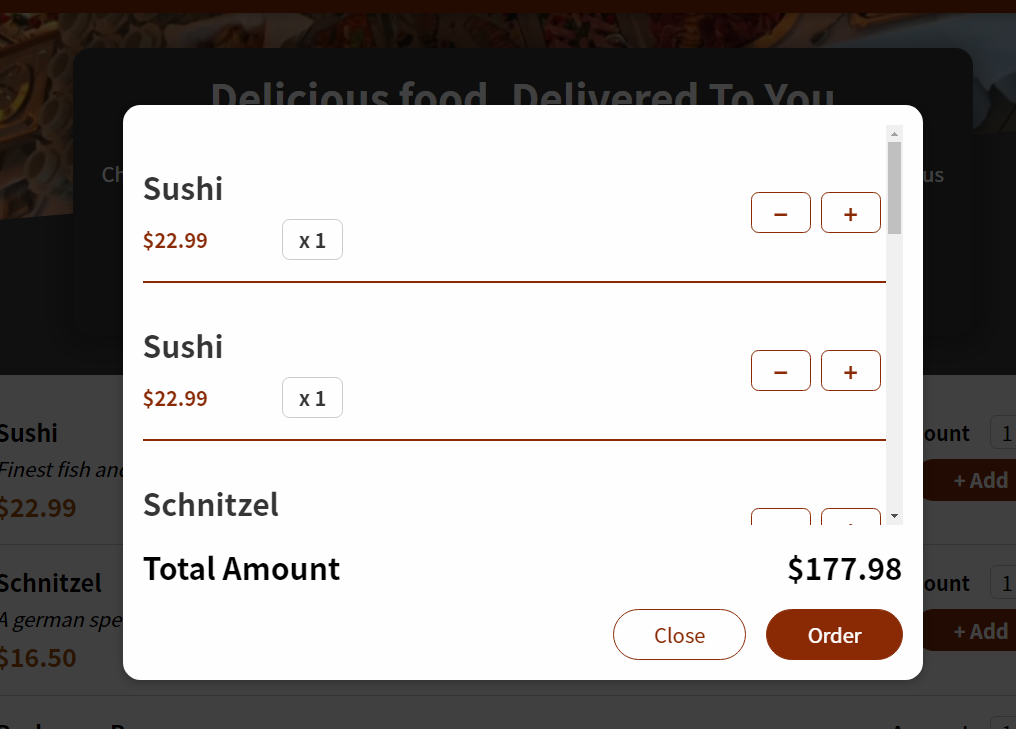
    </Modal>

  );

};

export default Cart;

Step 4 Check result



147. Working on a more complex reducer logic

Change in CartProvider.js

import { useReducer } from 'react';

import CartContext from './cart-context';

const defaultCartState = {

  items: [],

  totalAmount: 0,

};

const cartReducer = (state, action) => {

  if (action.type === 'ADD') {

    const updatedTotalAmount =

      state.totalAmount + action.item.price \* action.item.amount;

    const existingCartItemIndex = state.items.findIndex(

      (item) => item.id === action.item.id

    );

    const existingCartItem = state.items[existingCartItemIndex];

    let updatedItems;

    if (existingCartItem) {

      const updatedItem = {

        ...existingCartItem,

        amount: existingCartItem.amount + action.item.amount,

      };

      updatedItems = [...state.items];

      updatedItems[existingCartItemIndex] = updatedItem;

    } else {

      updatedItems = state.items.concat(action.item);

    }

    return {

      items: updatedItems,

      totalAmount: updatedTotalAmount,

    };

  }

  if (action.type === 'REMOVE') {

    const existingCartItemIndex = state.items.findIndex(

      (item) => item.id === action.id

    );

    const existingItem = state.items[existingCartItemIndex];

    const updatedTotalAmount = state.totalAmount - existingItem.price;

    let updatedItems;

    if (existingItem.amount === 1) {

      updatedItems = state.items.filter(item => item.id !== action.id);

    } else {

      const updatedItem = { ...existingItem, amount: existingItem.amount - 1 };

      updatedItems = [...state.items];

      updatedItems[existingCartItemIndex] = updatedItem;

    }

    return {

      items: updatedItems,

      totalAmount: updatedTotalAmount

    };

  }

  return defaultCartState;

};

const CartProvider = (props) => {

  const [cartState, dispatchCartAction] = useReducer(

    cartReducer,

    defaultCartState

  );

  const addItemToCartHandler = (item) => {

    dispatchCartAction({ type: 'ADD', item: item });

  };

  const removeItemFromCartHandler = (id) => {

    dispatchCartAction({ type: 'REMOVE', id: id });

  };

  const cartContext = {

    items: cartState.items,

    totalAmount: cartState.totalAmount,

    addItem: addItemToCartHandler,

    removeItem: removeItemFromCartHandler,

  };

  return (

    <CartContext.Provider value={cartContext}>

      {props.children}

    </CartContext.Provider>

  );

};

export default CartProvider;

148. Making Items Removable

Step 1 change in CardProvider first

import { useReducer } from 'react';

import CartContext from './cart-context';

const defaultCartState = {

  items: [],

  totalAmount: 0,

};

const cartReducer = (state, action) => {

  if (action.type === 'ADD') {

    const updatedTotalAmount =

      state.totalAmount + action.item.price \* action.item.amount;

    const existingCartItemIndex = state.items.findIndex(

      (item) => item.id === action.item.id

    );

    const existingCartItem = state.items[existingCartItemIndex];

    let updatedItems;

    if (existingCartItem) {

      const updatedItem = {

        ...existingCartItem,

        amount: existingCartItem.amount + action.item.amount,

      };

      updatedItems = [...state.items];

      updatedItems[existingCartItemIndex] = updatedItem;

    } else {

      updatedItems = state.items.concat(action.item);

    }

    return {

      items: updatedItems,

      totalAmount: updatedTotalAmount,

    };

  }

  if (action.type === 'REMOVE') {

    const existingCartItemIndex = state.items.findIndex(

      (item) => item.id === action.id

    );

    const existingItem = state.items[existingCartItemIndex];

    const updatedTotalAmount = state.totalAmount - existingItem.price;

    let updatedItems;

    if (existingItem.amount === 1) {

      updatedItems = state.items.filter(item => item.id !== action.id);

    } else {

      const updatedItem = { ...existingItem, amount: existingItem.amount - 1 };

      updatedItems = [...state.items];

      updatedItems[existingCartItemIndex] = updatedItem;

    }

    return {

      items: updatedItems,

      totalAmount: updatedTotalAmount

    };

  }

  return defaultCartState;

};

const CartProvider = (props) => {

  const [cartState, dispatchCartAction] = useReducer(

    cartReducer,

    defaultCartState

  );

  const addItemToCartHandler = (item) => {

    dispatchCartAction({ type: 'ADD', item: item });

  };

  const removeItemFromCartHandler = (id) => {

    dispatchCartAction({ type: 'REMOVE', id: id });

  };

  const cartContext = {

    items: cartState.items,

    totalAmount: cartState.totalAmount,

    addItem: addItemToCartHandler,

    removeItem: removeItemFromCartHandler,

  };

  return (

    <CartContext.Provider value={cartContext}>

      {props.children}

    </CartContext.Provider>

  );

};

export default CartProvider;

Step 2 change in Cart.js

import { useContext } from "react";

import CartContext from "../store/cart-context";

import Modal from "../UI/Modal";

import classes from "./Cart.module.css";

import CartItem from "./CartItem";

const Cart = (props) => {

  const cartCtx = useContext(CartContext);

  const totalAmount = `$${cartCtx.totalAmount.toFixed(2)}`;

  const hasItems = cartCtx.items.length > 2;

  const cartItemRemoveHandler = (id) => {

    cartCtx.removeItem(id);

  };

  const cartItemAddhandler = (item) => {

    cartCtx.addItem({ ...item, amount: 1 });

  };

  const cartItems = (

    <ul className={classes["cart-items"]}>

      {cartCtx.items.map((item) => (

        <CartItem

          key={item.id}

          name={item.name}

          amount={item.amount}

          price={item.price}

          onRemove={cartItemRemoveHandler.bind(null, item.id)}

            onAdd={cartItemAddhandler.bind(null, item)}

        />

      ))}

    </ul>

  );

  return (

    <Modal onClose={props.onClose}>

      {cartItems}

      <div className={classes.total}>

        <span>Total Amount</span>

        <span>{totalAmount}</span>

      </div>

      <div className={classes.actions}>

        <button className={classes["button--alt"]} onClick={props.onClose}>

          Close

        </button>

        {hasItems && <button className={classes.button}>Order</button>}

      </div>

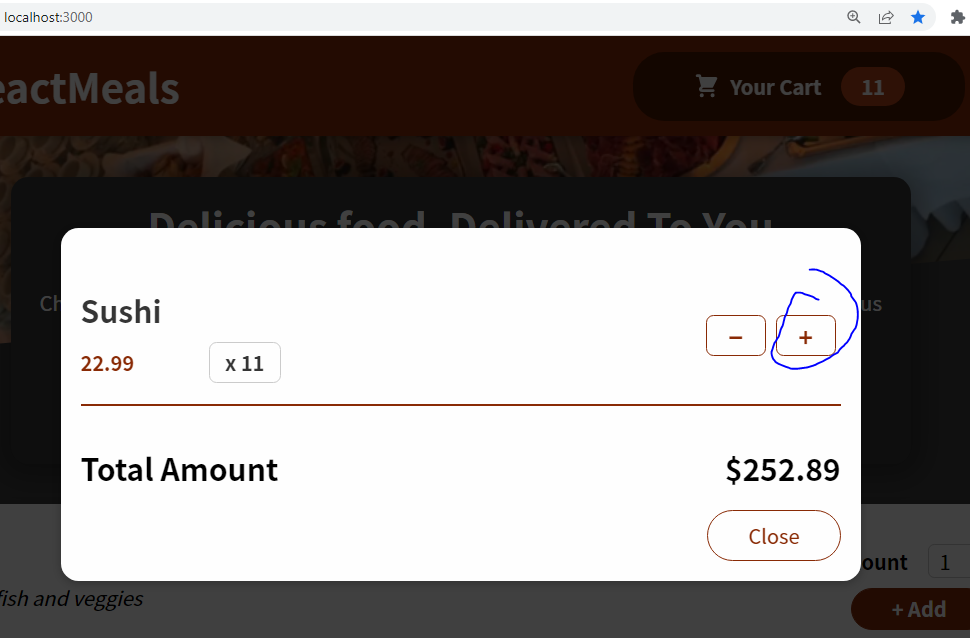
    </Modal>

  );

};

export default Cart;

step 3: check result



149. Using the useEffect Hook

Step 1 In HeaderCartButton add like this



And check animation when add new item

150. Module Resources

You may want to **compare your code to mine** (e.g. to find + fix errors).

For that, you find **multiple code snapshots** for this module here in this Github repository: <https://github.com/academind/react-complete-guide-code/tree/11-practice-food-order-app>