To create react app - npx create-react-app@5 pizza-menu

To start project – npm start or npm run start

**Section 5**

**33. Rendering the root component and strict mode.**

# Delete all from srcc folder

# Create a file in src – index.js

#

import React from "react";

import reactDOM from "react-dom/client";

// creating app component. its name will start with an uppercase (function)

function App() {

  return <h1>Hello React!</h1>;

}

//to render it to the dom

// bolow is the way we render the app in dom in react version 18 - new

const root = reactDOM.createRoot(document.getElementById("root"));

root.render(<App />);

// before react 18 - old versions

// ReactDOM.render(<App />);

// another change is - the second line is

// import reactDOM from "react-dom";

#

// root.render(<App />);

// strict mode for above one

root.render(

  <React.StrictMode>

    <App />

  </React.StrictMode>

);

Strict mode – what it does is during development it will render all components twice in order to find certan bugs and also react will check if we are using outdated parts in the react api. It is always a good idea to activate it when we develop a react applicaton.

34. before we start coding – debuggng

@ check the terminal if the app has started – eg npm start

@ if still not working- stop the app and restart

To stop – ctrl + c again- npm start

@reload browser

@ check the console (of browser) for error – keep it open

@ errors also shown on browser – copy error and goolge

@ always work with eslint. Underline shown on file, hover and see. Also check problem tab in vs code terminal

@ compare with jonas code, as downloaded from github

35. Components as building blocks

@refer notes

@ component can be nested inside other component

@ let there is three question list in a question list, we can create one question component and then use it three times,

Data for each of them will be different, but we can easily pass different data in different question components by using sthg called props.

When we need some duplication in our UI, we create a new component and reuse it many times.

36. Creating and using a component

import React, { StrictMode } from "react";

import reactDOM from "react-dom/client";

const pizzaData = [

  {

    name: "Focaccia",

    ingredients: "Bread with italian olive oil and rosemary",

    price: 6,

    photoName: "pizzas/focaccia.jpg",

    soldOut: false,

  },

  {

    name: "Pizza Margherita",

    ingredients: "Tomato and mozarella",

    price: 10,

    photoName: "pizzas/margherita.jpg",

    soldOut: false,

  },

  {

    name: "Pizza Spinaci",

    ingredients: "Tomato, mozarella, spinach, and ricotta cheese",

    price: 12,

    photoName: "pizzas/spinaci.jpg",

    soldOut: false,

  },

  {

    name: "Pizza Funghi",

    ingredients: "Tomato, mozarella, mushrooms, and onion",

    price: 12,

    photoName: "pizzas/funghi.jpg",

    soldOut: false,

  },

  {

    name: "Pizza Salamino",

    ingredients: "Tomato, mozarella, and pepperoni",

    price: 15,

    photoName: "pizzas/salamino.jpg",

    soldOut: true,

  },

  {

    name: "Pizza Prosciutto",

    ingredients: "Tomato, mozarella, ham, aragula, and burrata cheese",

    price: 18,

    photoName: "pizzas/prosciutto.jpg",

    soldOut: false,

  },

];

function App() {

  return (

    <div>

      <h1>Hello React!</h1>

      <Pizza />

      <Pizza />

      <Pizza />

    </div>

  );

}

// creating a new component - we use function to create a component in react

// function name start with capital and it needs to retutn some markup in jsx form

function Pizza() {

  return (

    <div>

      <img src="pizzas/spinaci.jpg" alt="pizza spinaci" />

      <h2>Pizza Spinaci</h2>

      <p>Tomato, mozarella, spinach, and ricotta cheese</p>

    </div>

  );

}

// to make the above work, put it in in app component in div (coz two are not allowed) eg <Pizza /> - then we get it in UI

// eg we have nested the Pizza component in App component eg we have called it in App

//after it open data.js from public folder(it was copied earlier)

// copy and paste it. eg pizza data  let after import 2nd line and delete that data.js file

// after that copy paste - Pizza Spinaci in h2 of Pizza() component from pizzaData

// then copy paste its ingredients in p (without quotes)

// can add images too

// now reusing this Pizza component. let add two more times in App component.Thus pizza component will be included three times.

//check browser. data is same. will be solved in props

// then

//sfsdf

const root = reactDOM.createRoot(document.getElementById("root"));

root.render(

  <React.StrictMode>

    <App />

  </React.StrictMode>

);

// not going to use github as of now - 9:10

// setting - setting - diff decorations - diff decorations then - All – to - none

37. What is jsx

@ component is a piece of UI.

@ babel converts jsx code to javaScript.

38. Creating more components.

function App() {

  return (

    <div>

      <Header />

      <Menu />

      <Footer />

    </div>

  );

}

// creating components - header - menu - footer

function Header() {

  return <h1>Fast React Pizza Co.</h1>; // using it in app component

}

// can combine small components in bigger one eg <Pizza /> componts in Menu compts

function Menu() {

  return (

    <div>

      <h2>Our Menu</h2>

      <Pizza />

      <Pizza />

      <Pizza />

    </div>

  );

}

//lets insted of immediately retuning to jsx, return a create element call

// writing component without jsx. in ()

// 'footer' - returning a Html node or element, named as footer

// 2nd argument - null -it is for props

// and 3rd - the child element -let here we want some text eg - "We're currently open"

// then add it in compnent app to render.

// also check chrome f12

// function Footer() {

//   return React.createElement("footer", null, "We're currently open");

// }

//creating the above again

// entering some js for time - {new Date().toLocaleTimeString()} - this is js in html

function Footer() {

  return (

    <footer>{new Date().toLocaleTimeString()} We're currently open</footer>

  );

}

//we can also write this as function expression and arrow function

// const Test = function () {};  //like this or even simpler like

// const Test = () =>   // for now use regular function

39. js logic in components

// writing logic inside react components.

// componts are just js functions we can do any js in them

function Footer() {

  const hour = new Date().getHours();

  const openHour = 12;

  const closeHour = 22;

  const isOpen = hour >= openHour && hour <= closeHour;

  console.log(isOpen);

  // if (hour >= openHour && hour <= closeHour) alert("we are currently open");

  // else alert("Sorry we're closed");

  return (

    <footer>{new Date().toLocaleTimeString()} We're currently open</footer>

  );

}

40. Separation of Concerns

41. Styling React Applications