MODULE: 2 (JavaScript Essentials)

Q.1 What is JavaScript Output method?

Ans: The console.log () method is the most common way to write output in JavaScript.

Q.2 How to used JavaScript Output method?

Ans: The method to use JavaScript for output is console.log ("Hello, JS");

Q.3 How to used JavaScript Events to do all examples?

Ans: This JavaScript event is used to process and verify user input, actions, and browser activities.

**EXAMPLE: -** <button onclick ="alert('Hello')"> Click me </button>

MODULE: 3 (JavaScript Essentials)

Q.1 What is React Js?

Ans: ReactJS is a JavaScript library for building user interfaces, particularly for creating single-page applications where UI updates are dynamic and responsive.

Q.2 What is NPM in React Js?

Ans: NPM in React JS stands for Node Package Manager. It is used to manage and install packages in a React project.

Q.3 What is Role of Node Js in react Js?

Ans: Node.js in React JS serves as the runtime environment, enabling server-side rendering and facilitating the development of scalable and high-performance applications.

Q.4 What is CLI command In React Js?

Ans: CLI command in React JS is stands for Command Line Interface command. Common ones include creating a new React app with create-react-app and running the development server with npm start.

Q.5 What is Components in React Js?

Ans: Components in React JS are reusable, self-contained building blocks that represent parts of a user interface. They encapsulate logic and UI, making code modular and maintainable.

Q.6 What is Header and Content Components in React Js?

Ans: Header and Content Components in React JS are examples of components that structure the user interface. Header may contain navigation elements, while Content holds the main content of the application.

Q.7 How to install React Js on Windows, Linux Operating System? How to install NPM and How to check version of NPM?

Ans: To install React JS on Windows or Linux, use the command npx create-react-app my-app where "my-app" is the name of our project. To install NPM, it comes bundled with Node.js. Check its version with npm -v.

Q.8 How to check version of React Js?

Ans: Check the version of React JS in your project by navigating to the project directory in the command line and running npm list react.

Q.9 How to change in components of React Js?

Ans: To make changes in React JS components, edit the corresponding component files. Use state and props to manage data dynamically, and re-render components when necessary to reflect updates.

HTML-CSS Login Registration page

Jsx:

import React from 'react'

function Loginpage() {

  return (

    <>

<div class="container">

    <h2>Login</h2>

    <form action="login.php" method="post">

        <label for="username">Username:</label>

        <input type="text" id="username" name="username" required/>

        <label for="password">Password:</label>

        <input type="password" id="password" name="password" required/>

        <button type="submit">Login</button>

    </form>

    <div class="reg">

        <p>Don't have an account? <a href="#registration">Register here.</a></p>

    </div>

</div>

</>

  )

}

export default Loginpage

CSS:

/\* Login-Page start \*/

body{

  font-family: Arial, sans-serif;

  background-color: #f4f4f4;

  display: flex;

  justify-content: center;

  align-items: center;

}

.container {

  background-color: #fff;

  padding: 20px;

}

form {

  display: flex;

  flex-direction: column;

}

input {

  padding: 10px;

  margin-bottom: 15px;

  border: 1px solid #acacac;

  border-radius: 5px;

}

button {

  background-color: #449c47;

  color: white;

  padding: 10px;

  border: none;

  border-radius: 5px;

  cursor: pointer;

}

button:hover {

  background-color: #318235;

}

.reg a {

  color: #462bf4;

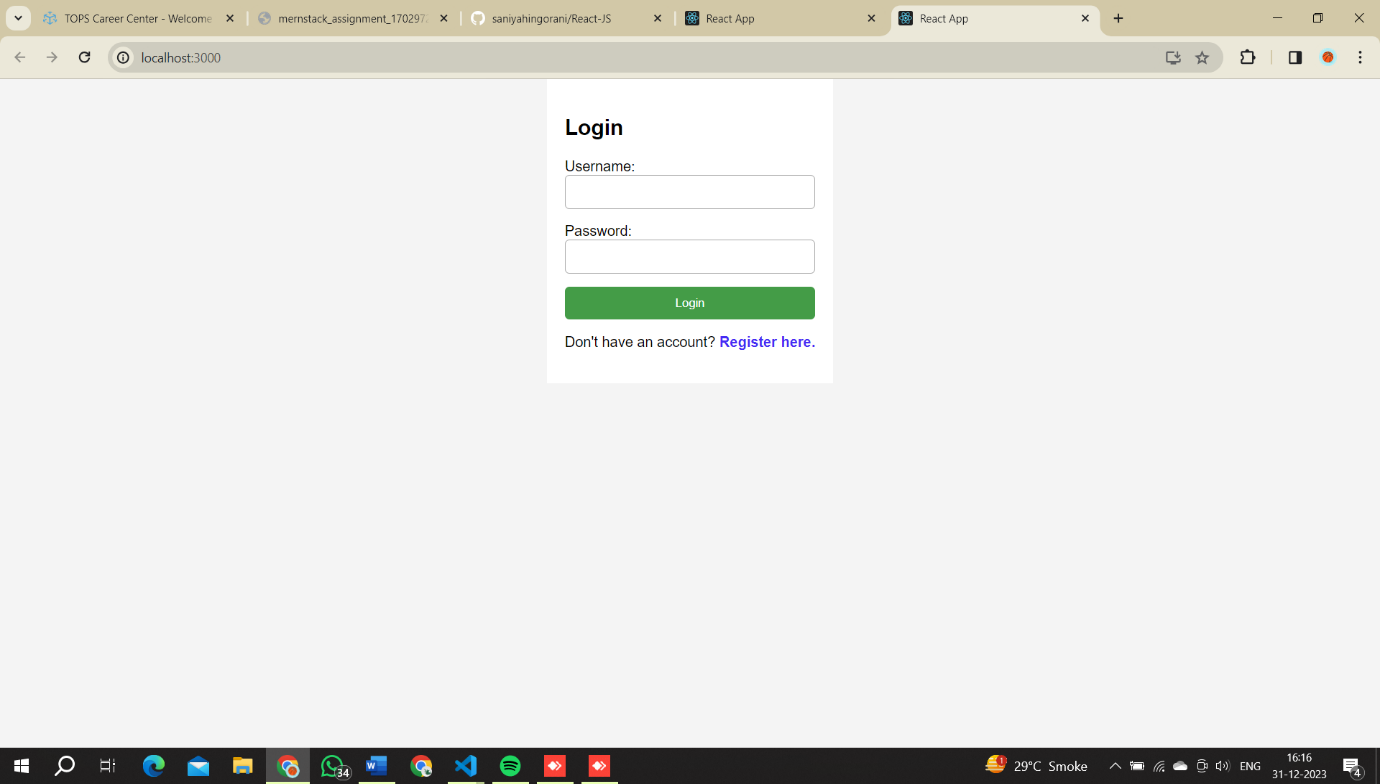
  text-decoration: none;

  font-weight: bold;

}

/\* Login-Page end \*/

Output:



How to Create a List View in React Js?

Jsx:

import React, { useState } from "react";

export default function Listview() {

  const [questions, setQuestions] = useState([

    // create state for list view

    {

      id: 1,

      question: "THE 'REACT WAY' TO RENDER LIST ",

      options: [

        "Use Array.map",

        "Not a for loop",

        "Give each item a unique key",

        "Avoid using array index as the key",

      ],

      // correctAnswer: ["Use Array.map"],

      selectedOptions: [], //For select Some options

    },

  ]);

  // For choose options

  const handleOptionSelect = (questionId, option) => {

    setQuestions((prevQuestions) =>

      prevQuestions.map((q) => {

        if (q.id === questionId) {

          const selectedOptions = q.selectedOptions.includes(option)

            ? q.selectedOptions.filter((selected) => selected !== option)

            : [...q.selectedOptions, option];

          return { ...q, selectedOptions };

        } else {

          return q;

        }

      })

    );

  };

  return (

    <>

      <div className="list-view">

        {questions.map((q) => (

          <div key={q.id}>

            <h1 className="Question">{q.question}</h1>

            <ul>

              {q.options.map((option) => (

                <li key={option}>

                  <label className="custom-checkbox-label">

                    {/\* For checkbox \*/}

                    <input

                      className="custom-checkbox-input"

                      type="checkbox"

                      name={`question\_${q.id}`}

                      value={option}

                      checked={q.selectedOptions.includes(option)}

                      onChange={() => handleOptionSelect(q.id, option)}

                    />

                    {/\* for options \*/}

                    <span className="custom-checkbox-text">{option}</span>

                  </label>

                </li>

              ))}

            </ul>

          </div>

        ))}

      </div>

    </>

  );

}

CSS:

/\* ------ Listview.css start ------ \*/

body {

  background-color: rgb(126, 233, 97);

}

h1,

div {

  text-align: center;

}

li {

  list-style: none;

}

.list-view {

  position: relative;

  top: 100px;

}

.custom-checkbox-label {

  display: flex;

  align-items: center;

}

.custom-checkbox-input {

  position: relative;

  left: 30%;

  margin: 5px;

  width: 30px;

  height: 30px;

  border-radius: 50%;

  vertical-align: middle;

  background: gainsboro;

  border: 1px solid gainsboro;

  appearance: none;

  -webkit-appearance: none;

  outline: none;

  cursor: pointer;

}

.custom-checkbox-input:checked {

  appearance: auto;

  clip-path: circle(50% at 50% 50%);

  background-color: blue;

}

.custom-checkbox-text {

  position: relative;

  left: 30%;

  height: 30px;

  width: 400px;

  text-align: left;

  font-size: 18px;

  border: 1px solid #fff;

  padding: 2px 5px;

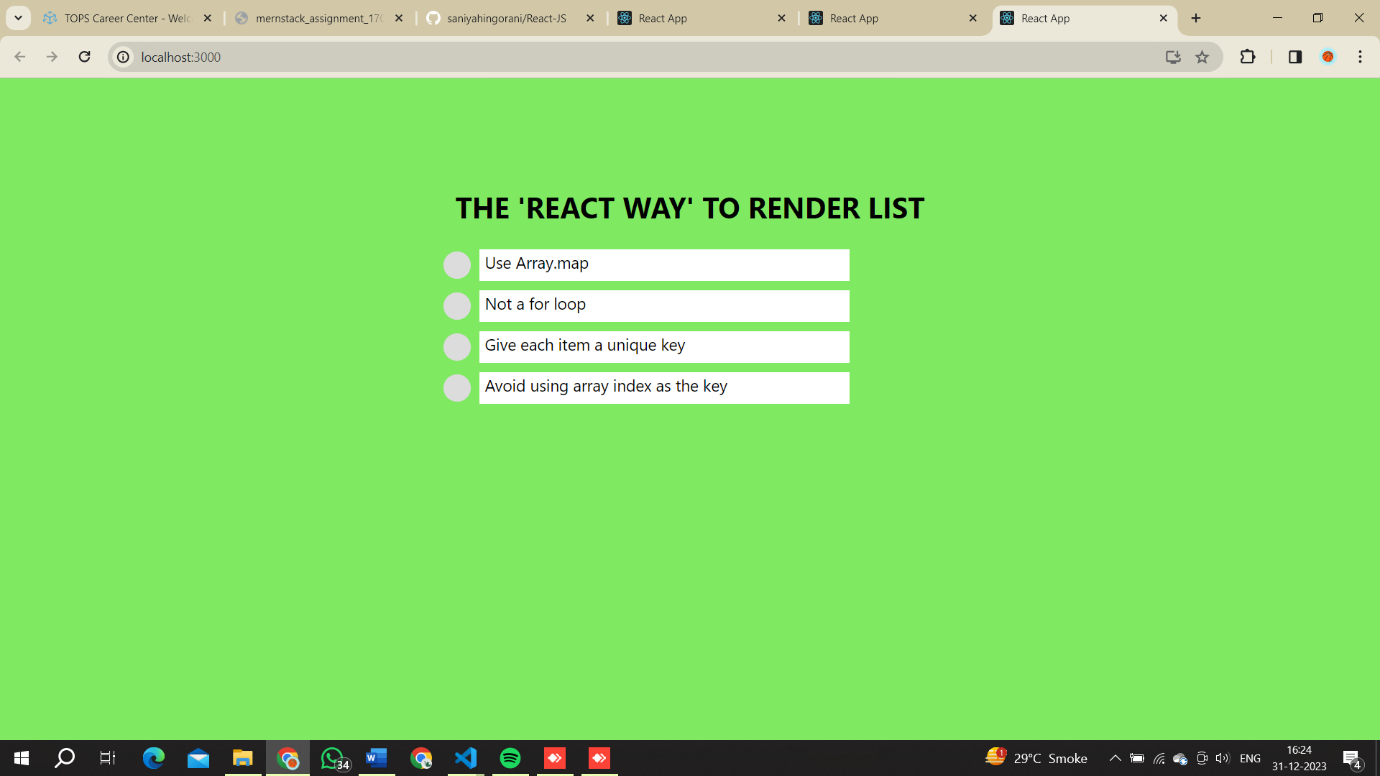
  margin: 5px;

  background: #fff;

}

/\* ------ Listview.css end ------ \*/

Output



Calculate subtotal price of quantity in JavaScript?

HTML/CSS:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Subtotal Price</title>

    <style>

        table {

          border-collapse: collapse;

          width: 100%;

        }

        th, td {

          border: 1px solid #ddd;

          padding: 8px;

          text-align: left;

        }

        th {

          background-color: #f2f2f2;

        }

        .remove-button {

          background-color: orange;

          color: white;

          border: none;

          padding: 5px 10px;

          cursor: pointer;

        }

      </style>

</head>

<body>

    <table>

        <thead>

          <tr>

            <th>Product Info</th>

            <th>Quantity</th>

            <th>Price per Unit</th>

            <th>Price Subtotal</th>

            <th></th>

          </tr>

        </thead>

        <tbody>

          <!-- First Row -->

          <tr>

            <td>Product 1</td>

            <td><input type="number" id="n1" onkeyup="a()"></td>

            <td>$20</td>

            <td id="r1">0</td>

            <td><button class="remove-button">Remove</button></td>

          </tr>

          <!-- Second Row -->

          <td>Product 2</td>

          <td><input type="number" id="n2" onkeyup="a()"></td>

          <td>$40</td>

          <td id="r2">0</td>

          <td><button class="remove-button">Remove</button></td>

        </tr>

          <!-- third Row -->

          <tr>

            <td></td>

            <td></td>

            <td></td>

            <td id="total">0</td>

            <td></td>

          </tr>

        </tbody>

      </table>

      <script>

     function a() {

        var n1=document.getElementById("n1").value

        var r1=document.getElementById("r1").innerHTML='$' + (20\*n1)

        var n2=document.getElementById("n2").value

        var r2=document.getElementById("r2").innerHTML= '$' + (40\*n2)

        var total=document.getElementById("total").innerHTML= '$' + (20\*n1 + 40\*n2)

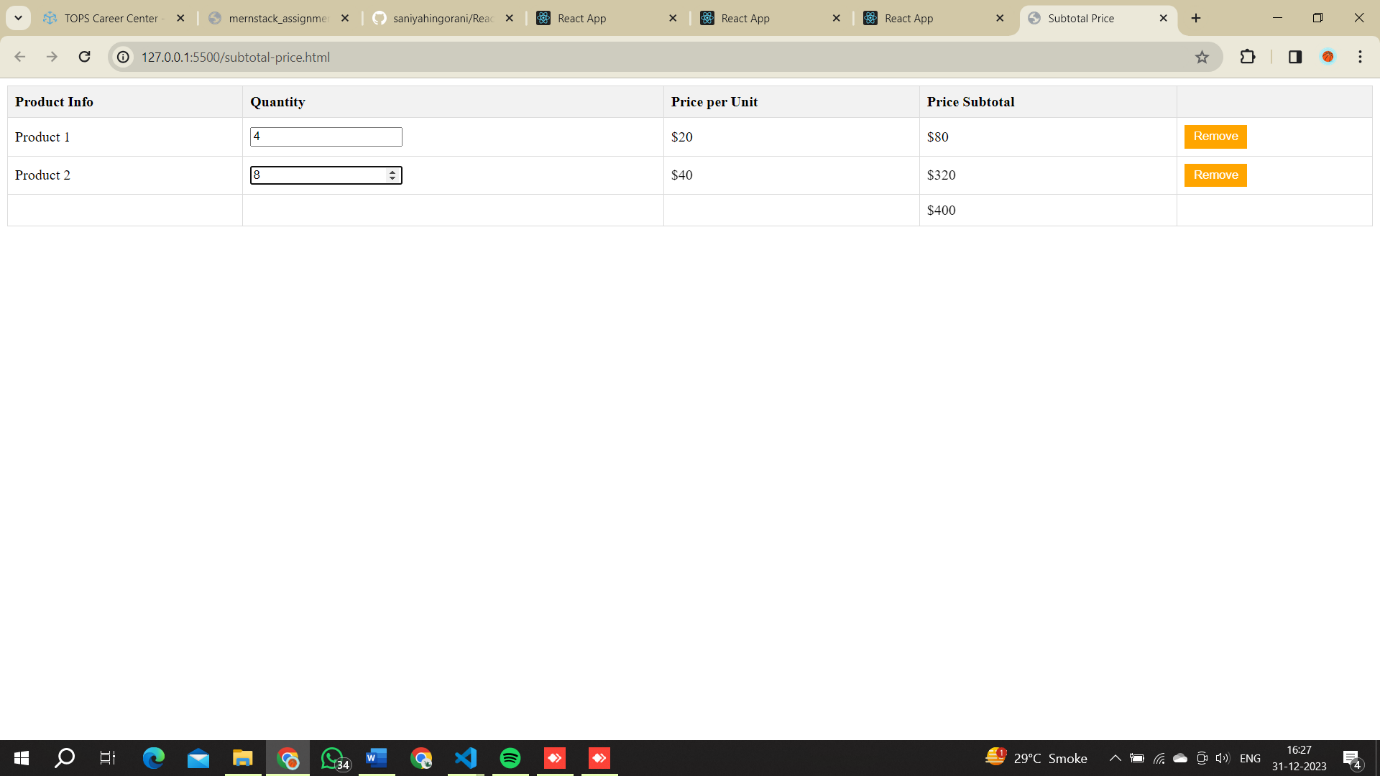
     }

      </script>

</body>

</html>

Output:



Create Increment decrement state change by button click?

Jsx:

import { useState } from "react";

import ReactDOM from "react-dom/client";

export default function IncrementDecrement() {

  const [count, setCount] = useState(0);

  // Function create for Decrement counting onClick button

  const Decrement = () => {

    if (count > 0) {

      setCount(count - 1);

    }

  };

  // Function to Reset the count to 0

  const handleReset = () => {

    setCount(0);

  };

  return (

    <>

    <div className='main container'>

        <div className="section">

        <h2>React Web</h2>

        <p>{count}</p>

         <button onClick={() => setCount((c) => c+1)}>Increment</button>

      <button className='btn2' onClick={() => setCount((c) => c-1)}>Decrement</button> <br />

      <button onClick={() => setCount(0)}>Reset</button>

    </div>

    </div>

    </>

  );

}

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

root.render(<IncrementDecrement />);

CSS:

/\* Counter css \*/

\*{

  background-color: rgb(81, 81, 235);

}

h2{

  color: white;

}

button{

  background-color: rgb(60, 60, 60);

  color: white;

  padding: 10px;

  margin-bottom: 20px;

}

.btn2{

  margin-left: 10px;

}

.main{

  display: flex;

  position: relative;

  align-items: center;

  justify-content: center;

}

.section{

 align-content: center;

 padding-top: 150px;

 text-align: center;

}

/\* Counter end \*/

Output

