Task Notification.js import React, { useState, useEffect } from "react"; import "./Notification.css"; function NotificationComponent() { const [notifications, setNotifications] = useState([]); const [prevLength, setPrevLength] = useState(0); const storedPrevLength = localStorage.getItem("prevLength"); useEffect(() => { fetch("https://databytess.com/api/adsapi/notifications?user=70") .then((response) => response.json()) .then((data) => { // Calculate the number of new notifications const newNotificationsCount = data.length - storedPrevLength; if (newNotificationsCount > 0) { // Get the latest notifications const latestNotifications = data.slice(data.length - newNotificationsCount); // Show popup for each new notification latestNotifications.forEach((notification, index) => { showPopup(notification.message, index); }); } // Update state with new notifications and store the current length setNotifications(data); // Store the current length in local storage localStorage.setItem("prevLength", data.length); }) .catch((error) => console.error("Error fetching notifications:", error)); }, []); // Fetch notifications only once on component mount const showPopup = (message, index) => { // Create a new popup for the given message const popup = document.createElement("div"); popup.className = "popup1234"; popup.style.bottom = `${index * 70 + 20}px`; // Adjust the spacing between popups popup.innerHTML = `×

${message}

`; document.body.appendChild(popup); // Close the popup after 5 seconds setTimeout(() => { popup.remove(); }, 50000); }; // return (//
// // // // // // {/* Render your notifications here */}
{notifications.map(notification => (

- {notification.message}

))}
//
//); } export default NotificationComponent; Notification.css .popup1234 { position: fixed; bottom: 20px; /* Adjust as needed */ right: 20px; background-color: rgba(0, 0, 0, 0.8); /* Black with transparency */ color: #fff; /* Text color */ padding: 10px; border-radius: 5px; box-shadow: 0 2px 5px rgba(0, 0, 0, 0.2); max-width: 300px; /* Adjust as needed */ z-index: 9; /* Ensure it's on top of other elements */ transition: opacity 0.3s ease; /* Add transition for smooth appearance */ } .mass{ color: aliceblue; font-size: 70%; } .popup1234 .close { float: right; cursor: pointer; color: #fff; /* Close button color */ } .popup1234 .close:hover { color: #ccc; /* Close button color on hover */ }

Feb batch

apply() method,bind() method,local storage,setitem(),getitem(),removeitem(),clear()

Assigment for day

Theoretical Questions: 1. Explain the purpose of the apply() method in JavaScript. How does it differ from call() method? 2. What is function binding in JavaScript? Describe the role of the bind() method in function binding. 3. What is local storage in web development? How does it differ from session storage? 4. Explain the role of the setItem(), getItem(), removeItem(), and clear() methods in working with local storage. Coding Questions: Using apply(): 1. Write a JavaScript function that finds the maximum element in an array using the apply() method. Using bind(): 1. Create a function named greet that takes a name parameter and prints a greeting message. Then use the bind() method to create a new function called greetJohn that always greets "John". Using Local Storage: 1. Write JavaScript code to store user preferences (e.g., theme color, font size) using local storage. Implement functions to set, get, remove, and clear these preferences. Using Local Storage in a ToDo List: 1. Create a simple ToDo list application using HTML, CSS, and JavaScript. Utilize local storage to persist the ToDo list items even after the page reloads. Implement functionality to add, delete, and mark items as completed. Error Handling with Local Storage: 1. Modify the previous ToDo list application to handle errors that might occur when accessing local storage. Implement try-catch blocks to gracefully handle exceptions.

March batch

Data Types,primitive datatype,non primitive data type(number,string,boolean,bigint,undefined,null,symbol,object)(Object,array,date object),Opertors:- assigment , arithmetic, logical , comparison

Data Types: a. Define primitive data types and provide examples. b. Explain non-primitive data types with examples. c. Discuss the difference between primitive and non-primitive data types. Operators: a. Explain arithmetic operators and provide examples of each. b. Discuss assignment operators and give examples. c. Define comparison operators and provide examples. d. Explain logical operators and provide examples. Coding Assignments: a. Write a program that takes two numbers, and performs addition, subtraction, multiplication, and division on them using arithmetic operators. b. Write a program to calculate the area of a rectangle using default values for length and width. Use appropriate assignment operators. c. Write a program that compares two numbers and prints whether the first number is greater than, less than, or equal to the second number. d. Write a program that takes two boolean values and performs logical AND, OR, and NOT operations on them.