**REACT**

**NAME: SUMAN SOURABH**

**SUPERSET ID: 6363866**

HANDS-ON 9:

**Objectives**

* List the features of ES6

ES6 or ECMAScript 2015 brings several powerful features such as 'let' and 'const' for block-scoped variables, arrow functions for concise syntax, classes and inheritance, template literals, destructuring, default parameters, spread and rest operators, Promises for asynchronous programming, modules (import/export), and new data structures like 'Map' and 'Set'.

* Explain JavaScript let

'let' is a keyword that declares block-scoped variables in JavaScript. It allows the variable to be reassigned but does not allow redeclaration within the same scope. It helps in avoiding bugs arising from variable scoping.

* Identify the differences between var and let

'var' is function scope and allows the same variable to be re-declared within that same scope. On the other hand, 'let' is block scoped and disallows the same variable to be re-declared. Apart from this, variables declared by 'var' are hoisted and initialized with a default value of 'undefined', whereas variables declared by 'let' are hoisted differently because they are not initialized.

* Explain JavaScript const

'const' creates variables that are block-scoped constants. Variables with a 'const' declaration can never be reassigned; however, they can be mutated in the form of changing the object's inner property values or array element values.

* Explain ES6 class fundamentals

Essentially, ES6 classes are meant to give a very clean and concise syntax for objects and inheritance. A class can have a constructor method and other methods, and a class can be instantiated using 'new'.

* Explain ES6 class inheritance

It inherits using the 'extends' keyword. 'Super()' needs to be called inside the constructor of the child class in order to call the constructor of the parent class and inherit properties and methods.

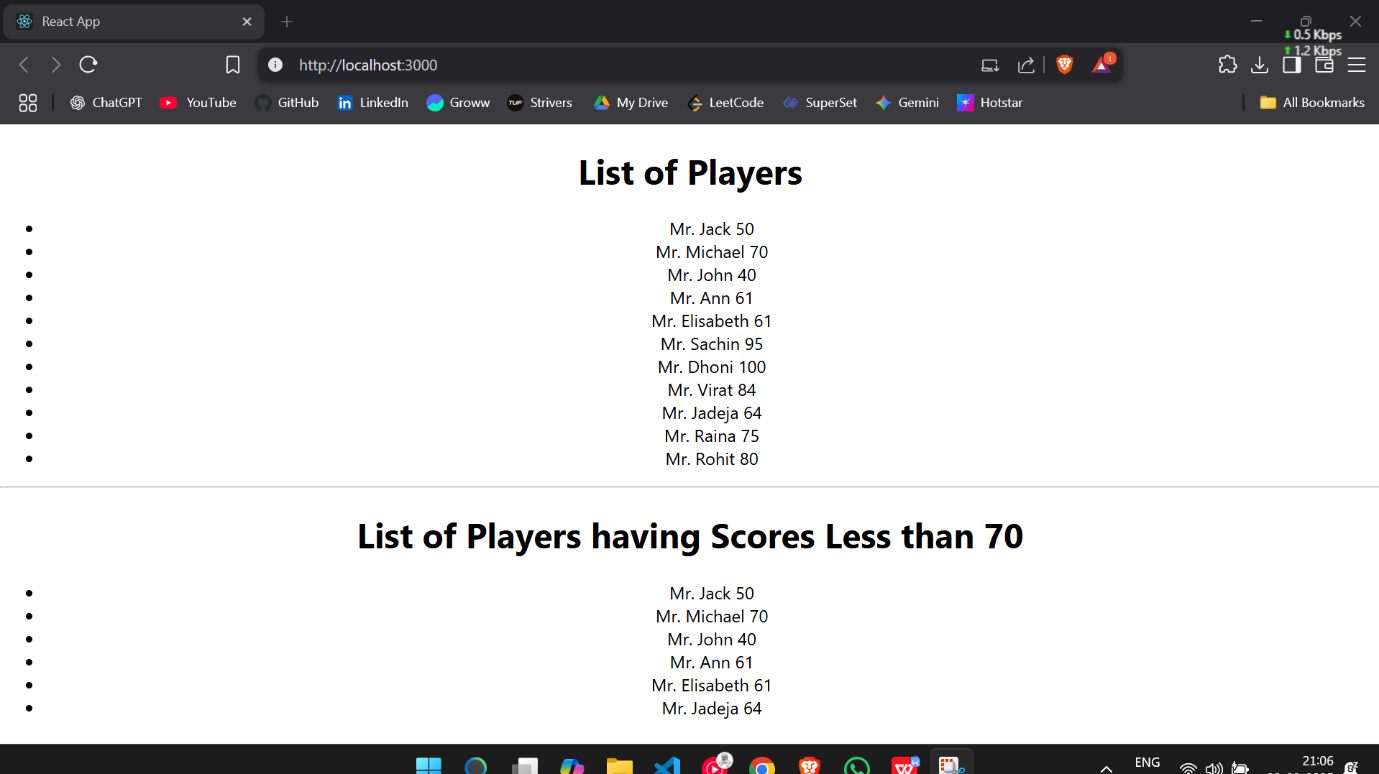
* Define ES6 arrow functions

Arrow functions provide a quick way of writing functions, using the '=>' symbol. They do not bind their own 'this' value, meaning that they inherit 'this' from the parent scope, hence serving well for the callbacks and closures.

* Identify set(), map()

A 'Set' is a collection of unique values where duplicates will be removed automatically. A 'Map' is a collection of key-value pairs where the keys can be of any data type, and it preserves the order of insertion.

Flag= True



Flag= False

A screenshot of a computer

AI-generated content may be incorrect.

HANDS-ON 10:

**Objectives**

* Define JSX

JSX stands for JavaScript XML. It is a special syntax that allows writing HTML-like tags within a JavaScript file. The main use of JSX is in React to design UI structures in a more human-readable way.

* Explain about ECMA Script

ECMAScript is the specification standard upon which JavaScript is based. ES6 is the sixth set of specifications, adding modern capabilities to the language.

* Explain React.createElement()

`React.createElement()` is the method by which JSX creates virtual DOM elements internally. It can be called directly to manually create React elements without resorting to JSX.

* Explain how to create React nodes with JSX

React nodes can be created using the JSX syntax. For instance, `<h1>Hello</h1>` creates React nodes that represent an HTML heading element containing the text "Hello."

* Define how to render JSX to DOM

JSX can be rendered to the DOM using `ReactDOM.render()`. For example, `ReactDOM.render(<App />, document.getElementById('root'))` would render the App component inside the DOM node with the ID of `'root'`.

* Explain how to use JavaScript expressions in JSX

JavaScript expressions can be included inside JSX by wrapping them in curly braces `{}`. For instance, `<h1>Hello, {name}</h1>` will render an H1 tag containing "Hello, " followed by whatever string the variable `name` represents.

* Explain how to use inline CSS in JSX

Inline CSS in JSX is done with inline styles expressed as a JavaScript object. For instance:

<h1 style={{ color: 'red', fontSize: '20px' }}>Styled</h1>.

A screenshot of a computer

AI-generated content may be incorrect.

HANDS-ON 11:

**Objectives**

* Explain React events

React events are similar to DOM events and are triggered by user interactions like clicks or inputs. They are written in camelCase (e.g., onClick) and are handled using functions.

* Explain about event handlers

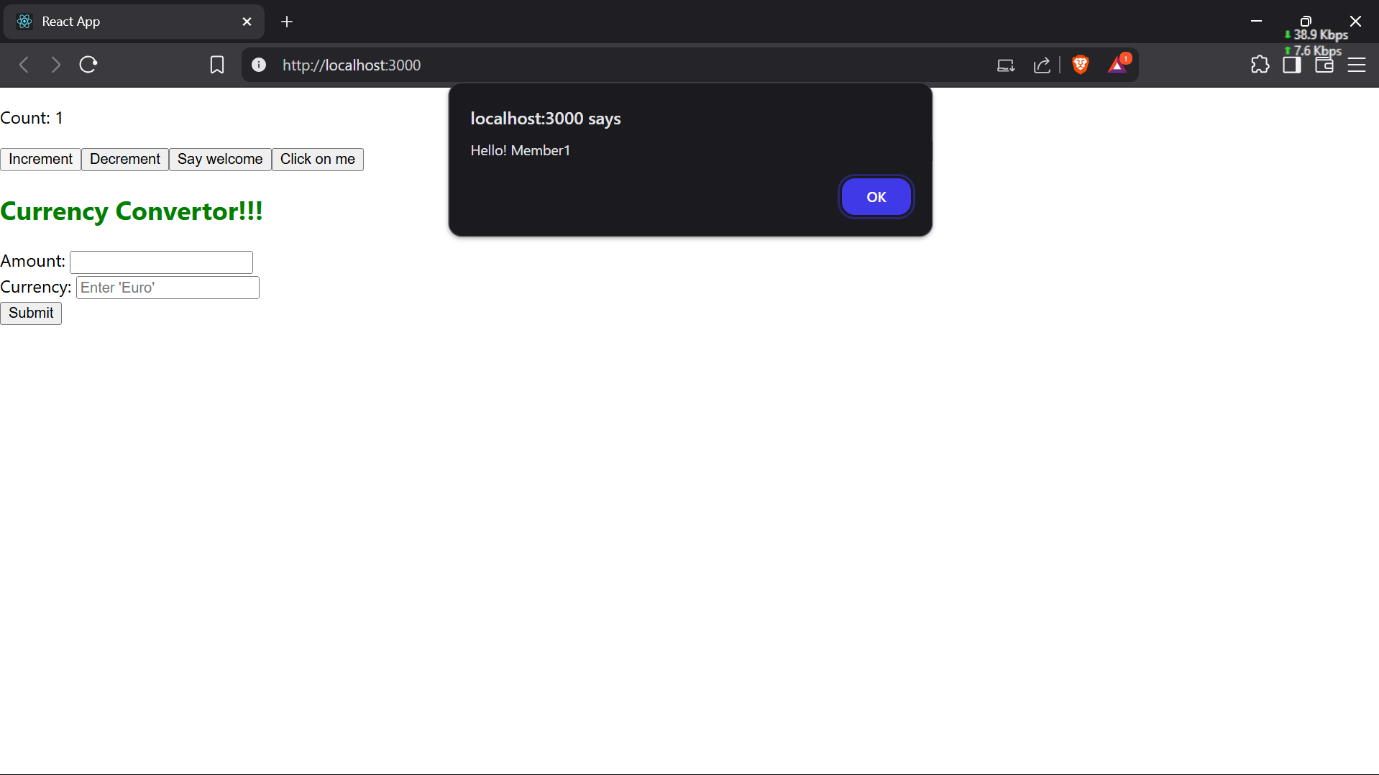
Event handlers are functions that are triggered when a specific event occurs. For example, a button can use onClick={handleClick} to respond to a click event.

* Define Synthetic event

A Synthetic Event is a wrapper around the browser’s native event system in React. It ensures consistent behavior across different browsers by normalizing the event object.

* Identify React event naming convention

React uses camelCase for event names, such as onClick, onChange, onMouseEnter, etc., unlike the lowercase naming used in HTML.



A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

HANDS-ON 12:

**Objectives**

* Explain about conditional rendering in React

Conditional rendering in React essentially permits components to display different outputs depending on certain conditions. This is frequently accomplished through the use of if statements, ternary operators, or logical &&. The approach allows for dynamic control over what the user sees based on the state or props.

* Define element variables

Element variables in React refer to variables that are assigned JSX elements. These are particularly useful for conditional rendering; for instance, one might assign either a <LoginButton /> or a <LogoutButton /> to a variable, depending on the current application state.

* Explain how to prevent components from rendering

To prevent a component from rendering in React, simply return null from the component’s render method or function. Doing so instructs React to omit rendering that component in the DOM.

A screen shot of a computer

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

HANDS-ON 13:

**Objectives**

* Explain various ways of conditional rendering

There are several techniques available for conditional rendering in React, including if/else statements, ternary operators (condition ? true : false), logical AND (&&), or by defining functions that return different components according to specific conditions.

* Explain how to render multiple components

Rendering multiple components in React is straightforward. Multiple child components can be grouped within a parent container, such as a <div>, a React.Fragment, or the shorthand <>...</> syntax.

* Define list component

A list component in React is designed to render multiple items dynamically. This is commonly achieved using the map() function to iterate over an array and return a list of components to be displayed.

* Explain about keys in React applications

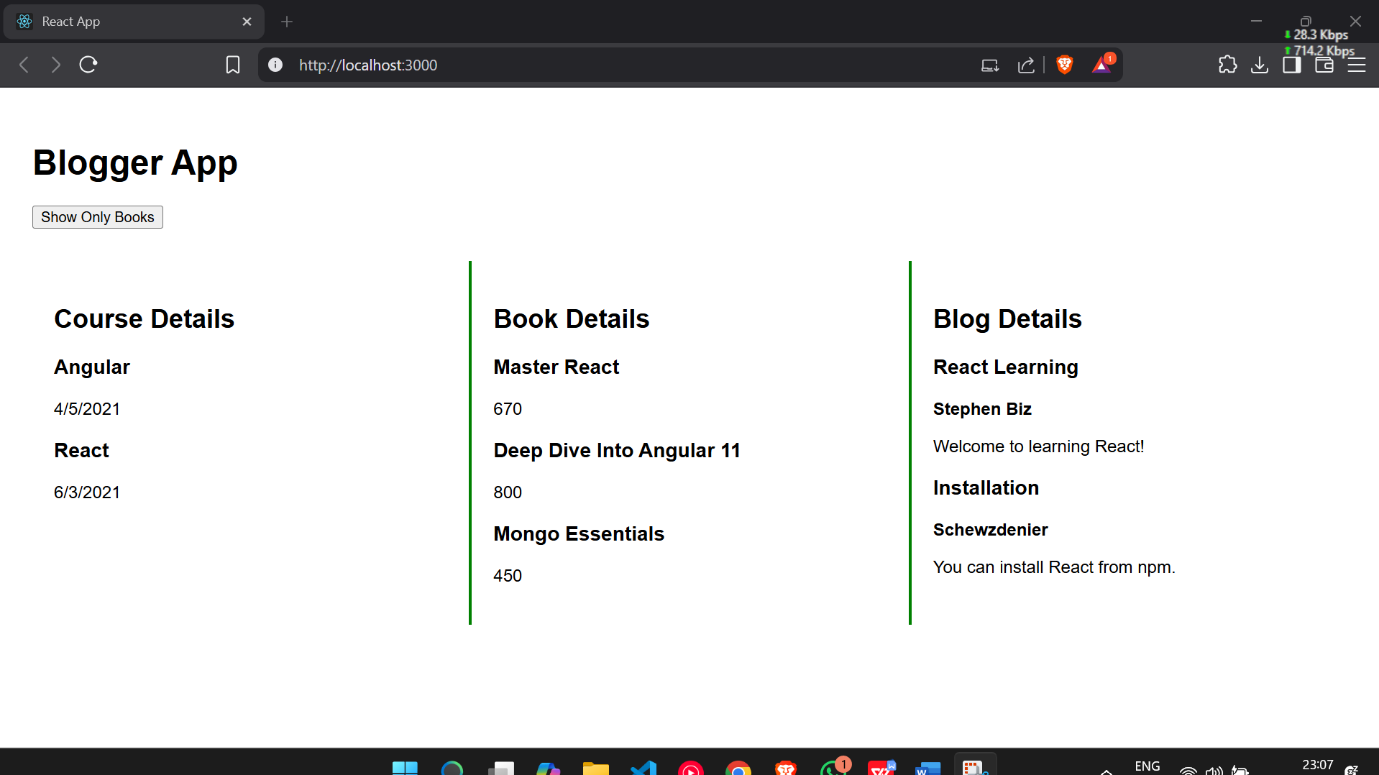
Keys in React are unique identifiers assigned to elements within a list. Their primary function is to help React distinguish between individual items when updating the UI, thereby improving rendering performance and accuracy.

* Explain how to extract components with keys

Extracting components with keys involves creating a separate component for each item in a list and passing the key as a prop. This practice enhances code structure and maintainability, especially in larger applications.

* Explain React Map, map() function

The map() function in React is used to iterate over arrays and generate a new array of JSX elements. It is an indispensable method for rendering lists efficiently within React applications.



A screenshot of a computer

AI-generated content may be incorrect.