# MSc DS - Web Technologies

Module No: 5

## Revision from last lecture

1. Module 4 focuses on web development and the connection between JavaScript and the Document Object Model (DOM).

2. The module covers modifying web page components using JavaScript and emphasises the importance of website responsiveness.

3. The module also explores sophisticated data storage techniques beyond cookies, including the Web Storage API and IndexedDB.

## Expected Learning Outcomes

1. Understanding more complex JavaScript ideas will help current web development frameworks take off.

2. Understand the fundamentals of React.js, including components, state, and props, to create dynamic user interfaces.

3. Understand the fundamentals of JSX and the Virtual DOM as you explore how they contribute to effective rendering in React apps.

4. Understand the fundamentals of Node.js and Express, master setting up a React development environment, and use server-side JavaScript.

## Basics of Advanced JavaScript

The Document Object Model (DOM) is a programming interface for web content.

It represents web pages as hierarchical structures composed of elements.

The DOM allows developers to retrieve, change, and interact with web page information, structure, and styles.

## React.js - Part I: Components, State, and Props

• React.js is a JavaScript library created by Facebook that revolutionises user interfaces in online apps by using components.

• Components are discrete code modules that encapsulate elements of the user interface, allowing for modular and scalable development.

• The concept of "state" in React allows for dynamic changes in the user interface based on user actions, while "props" facilitate the transmission of data between components.

## Introduction to JSX and Virtual DOM

React.js introduced JSX and the Virtual DOM, which have greatly influenced web development.

JSX allows developers to create user interface components using a syntax similar to HTML, making it more natural and expressive.

The Virtual DOM optimises performance by evaluating UI modifications and determining the most efficient way to update the actual DOM, reducing unnecessary computations and redraws.

## Setting Up a React Environment

• React.js is known for its component-based structure and rendering capabilities in online development.

• The Create React App (CRA) tool is recommended for initiating a React project, as it generates a scaffold with necessary files and settings.

• Using a specialised code editor, like Visual Studio Code with React extensions, enhances the coding experience by providing syntax highlighting and error detection.

## Node.js Fundamentals: Server-Side JavaScript and Express Basics

• Node.js enables the execution of JavaScript on the server-side, allowing for full-stack application development using a single programming language.

• Node.js's non-blocking, event-driven architecture ensures expedited response times and improved scalability by managing multiple requests concurrently.

• Express.js, a web application framework, simplifies server configuration, route management, and middleware handling, making it a preferred option for developers.

## Important Terminologies

• Components: Components are fundamental elements of React that are used to construct reusable and modular user interface portions.

• State: The internal data store inside React components is responsible for determining the rendering and behaviour of a component.

• Props: Abbreviated as "props," properties refer to the inputs provided to a React component, facilitating the transfer of data across such components.

• JSX: JSX is a syntactic extension of the JavaScript programming language that facilitates the seamless integration of JavaScript and HTML, hence enhancing the process of defining user interfaces (UI) in the React framework by providing a more natural approach.

• Virtual DOM: The Virtual DOM is a memory-based version of the actual DOM components in the React framework.

• Express.js: Express.js is a web application framework designed for Node.js, with the purpose of simplifying the process of setting up servers, managing middleware, and handling routing. It is known for its lightweight nature.

## Summary

❖ Node.js enables the execution of JavaScript on the server-side, allowing for full-stack application development using a single programming language.

❖ Node.js's non-blocking, event-driven architecture ensures expedited response times and improved scalability by managing multiple requests concurrently.

❖ Express.js, a web application framework, simplifies server configuration, route management, and middleware handling, making it a preferred option for developers.

**THANK YOU**