- # Module 9 - Introduction to React.js

Q-1 What is React.js? How is it different from other JavaScript frameworks and libraries?

Ans : React.js is an open-source JavaScript library developed by Meta (formerly Facebook) for building user interfaces, especially single-page applications (SPAs). It allows developers to create reusable UI components.

Feature	React.js	Others (e.g., Angular, Vue)
Type	Library	Angular is a full framework, Vue is similar
Approach	Component-based	Angular uses MVC/MVVM
DOM Handling	Virtual DOM	Angular uses real DOM or Shadow DOM
Learning Curve	Moderate	Angular is steeper, Vue is easier
Data Binding	One-way binding	Angular uses two-way binding

Q-2 Explain the core principles of React such as the Virtual DOM and Component-Based Architecture.

Ans : 1. Virtual DOM:

React creates a virtual copy of the real DOM in memory. When the state of an object changes, only the changed element is updated in the virtual DOM, and then the minimal difference is updated in the real DOM, increasing efficiency.

Benefit: Faster performance, fewer direct DOM manipulations.

- **2. Component-Based Architecture:**

React apps are made up of small, reusable components. Each component is a piece of UI (like a button, form, or navbar), which can manage its own state and logic.

Benefit: Easier maintenance, reusability, and code readability.

Question 3: What are the advantages of using React.js in web development?

Ans :

1. Fast Performance (thanks to Virtual DOM)
2. Reusable Components (modular and maintainable code)
3. One-Way Data Binding (predictable data flow)
4. Tooling & Ecosystem (React Developer Tools, CRA, Next.js)
5. Strong Community Support
6. Supports Mobile App Development via React Native
7. Easy Integration with other frameworks or libraries

- **JSX (JavaScript XML)**

Q-1 What is JSX in React.js? Why is it used?

Ans : Why JSX is used:

- It allows HTML-like syntax inside JavaScript.
- Makes code more readable and expressive.
- Helps React to create and render UI components more intuitively.
- Under the hood, JSX is converted into React.createElement() calls**.**

Example:

const element = <h1>Hello, Zero2Code!</h1>;

Q-2 How is JSX different from regular JavaScript? Can you write JavaScript inside JSX?

Ans : Differences:

JSX	Regular JavaScript
Looks like HTML	Pure JavaScript syntax
Not understood by browser	Understood by browser directly
Needs to be transpiled (via Babel)	No transpilation needed
Can embed JS expressions with {}	Doesn’t need embedding

Yes, you can write JavaScript expressions inside JSX using curly braces

Example:

jsx

CopyEdit

```
const name = "Raj";
const element = <h2>Hello, {name}!</h2>;
```

Q-3 Discuss the importance of using curly braces {} in JSX expressions.

Ans :

- Curly braces {} are used in JSX to embed JavaScript expressions inside the HTML-like syntax.
- This allows dynamic content, logic, or variable output in JSX.

Used for:

- Inserting variables
- Running simple expressions (e.g., 2 + 2)
- Function calls that return values

Example:

```
  jsx
CopyEdit
const age = 20;
const element = <p>Your age is {age}</p>;
Output: Your age is 20
```