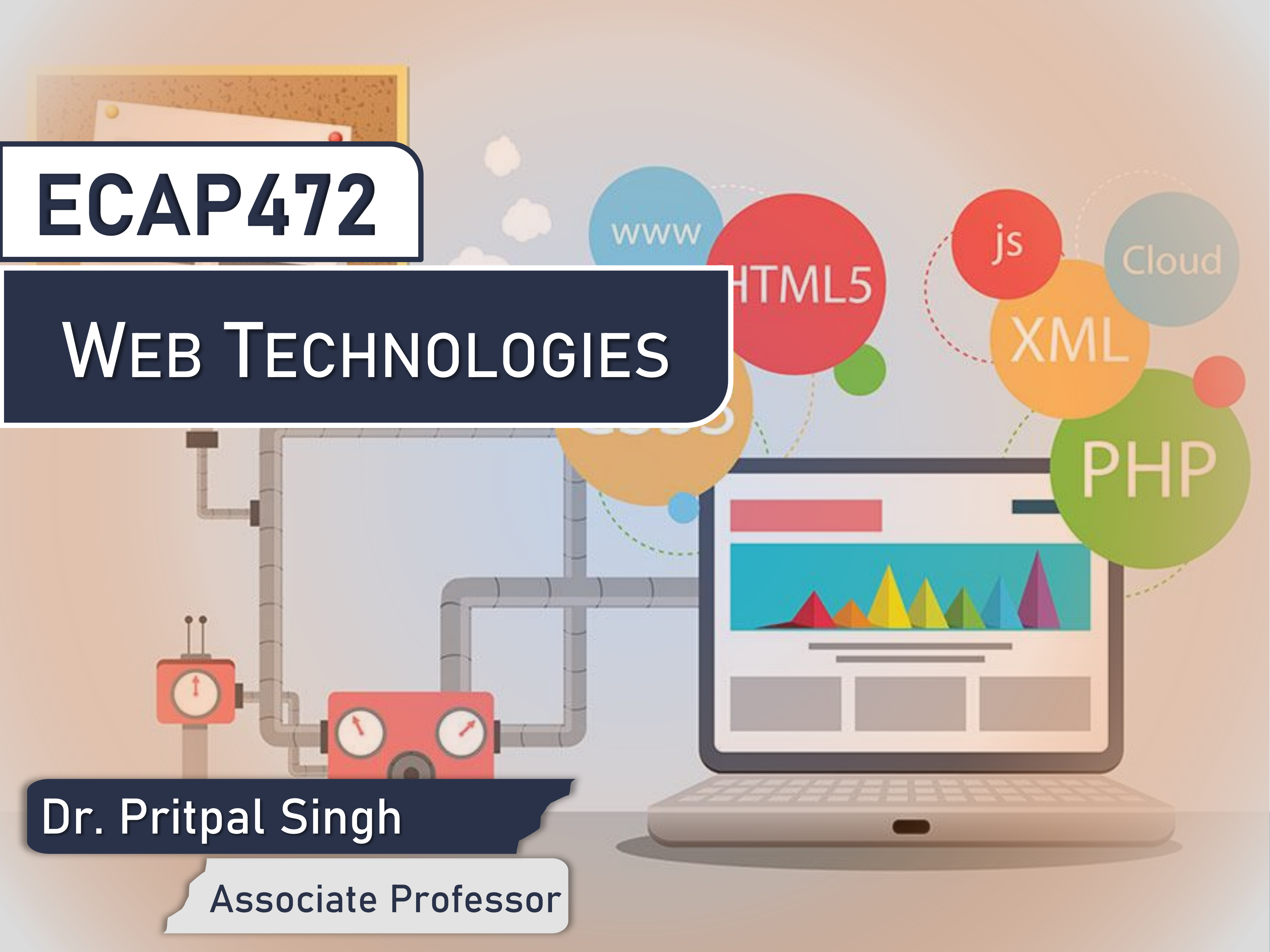
ECAP472

Web Technologies

Dr. Pritpal Singh

Associate Professor



Learning Outcomes

After this lecture, you will be able to

- understand concept of React Lists and Forms.

React Lists

Lists are used to display data in an ordered format and mainly used to display menus on websites. In React, Lists can be created in a similar way as we create lists in JavaScript.

React Lists

- In React, you will render lists with some type of loop.
- The JavaScript map() array method is generally the preferred method.

Example:

```
function Car(props) {
 return <li>I am a { props.brand }</li>;}
function Garage() { const cars = ['Ford',
'BMW', 'Audi'];
 return (
  <>  <h1>Who lives in my garage?</h1>
    <ul>
     {cars.map((car) => <Car brand={car}
/>)}
    </ul>
  </>
 );
}
```

React Forms

Just like in HTML, React uses forms to allow users to interact with the web page.

React Forms

Forms are an integral part of any modern web application. It allows the users to interact with the application as well as gather information from the users.

React Forms

Forms can perform many tasks that depend on the nature of your business requirements and logic such as authentication of the user, adding user, searching, filtering, booking, ordering, etc. A form can contain text fields, buttons, checkbox, radio button, etc.

Creating Form

React offers a stateful, reactive approach to build a form. The component rather than the DOM usually handles the React form. In React, the form is usually implemented by using controlled components.

Creating Form

There are mainly two types of form input in React.

Uncontrolled Component

Controlled Component

Uncontrolled Component

- The uncontrolled input is similar to the traditional HTML form inputs.
- The DOM itself handles the form data.
- The HTML elements maintain their own state that will be updated when the input value changes.

Uncontrolled Component

- To write an uncontrolled component, you need to use a ref to get form values from the DOM.
- In other words, there is no need to write an event handler for every state update.
- You can use a ref to access the input field value of the form from the DOM.

Controlled Component

- In HTML, form elements typically maintain their own state and update it according to the user input.
- In the controlled component, the input form element is handled by the component rather than the DOM.
- The mutable state is kept in the state property and will be updated only with setState() method.

Controlled Component

- Controlled components have functions that govern the data passing into them on every onChange event, rather than grabbing the data only once, e.g., when you click a submit button.
- This data is then saved to state and updated with setState() method. This makes component have better control over the form elements and data.

Controlled Component

- A controlled component takes its current value through props and notifies the changes through callbacks like an onChange event.
- A parent component "controls" this changes by handling the callback and managing its own state and then passing the new values as props to the controlled component.
- It is also called as a "dumb component."

Adding Forms in React

```
function MyForm() {
  return (
    <form>
      <label>Enter your name:
        <input type="text" />
      </label>
    </form>
  )
}
ReactDOM.render(<MyForm />,
document.getElementById('root'));
```

Handling Forms

- Handling forms is about how you handle the data when it changes value or gets submitted.
- In HTML, form data is usually handled by the DOM.
- In React, form data is usually handled by the components.

Handling Forms

- When the data is handled by the components, all the data is stored in the component state.
- You can control changes by adding event handlers in the onChange attribute.
- We can use the useState Hook to keep track of each inputs value and provide a "single source of truth" for the entire application.

Multiple Input Fields

- You can control the values of more than one input field by adding a name attribute to each element.
- We will initialize our state with an empty object.
- To access the fields in the event handler use the event.target.name and event.target.value syntax.
- To update the state, use square brackets [bracket notation] around the property name.

React is a JavaScript Library for Building User Interfaces.

Declarative:

- React makes it painless to create interactive UIs. Design simple views for each state in your application, and React will efficiently update & render just the right components when data changes.
- Declarative views make code more predictable, simpler to understand, and easier to debug.

Component-Based

Component-Based

- Build encapsulated components that manage their state, then compose them to make complex UIs.
- Since component logic is written in JavaScript instead of templates, you can easily pass rich data through your app and keep the state out of the DOM.

Learn Once, Write Anywhere

Learn Once, Write Anywhere

- We don't make assumptions about the rest of your technology stack, so you can develop new features in React without rewriting existing code.
- React can also render on the server using Node and power mobile apps using

Known to be SEO Friendly

- Traditional JavaScript frameworks have an issue in dealing with SEO. The search engines generally having trouble in reading JavaScript-heavy applications.
- Many web developers have often complained about this problem. ReactJS overcomes this problem that helps developers to be easily navigated on various search engines.

Known to be SEO Friendly

- It is because React.js applications can run on the server, and the virtual DOM will be rendering and returning to the browser as a regular web page.

Virtual DOM

- A virtual DOM object is a representation of the original DOM object.
- It works like a one-way data binding. Whenever any modifications happen in the web application, the entire UI is re-rendered in virtual DOM representation.

Virtual DOM

- Then it checks the difference between the previous DOM representation and new DOM. Once it has done, the real DOM will update only the things that have actually changed.
- This makes the application faster, and there is no wastage of memory.

Performance

- ReactJS is known to be a great performer. This feature makes it much better than other frameworks out there today.
- The reason behind this is that it manages a virtual DOM.
- The DOM is a cross-platform and programming API which deals with HTML, XML or XHTML.
- The DOM exists entirely in memory.

Performance

- Due to this, when we create a component, we did not write directly to the DOM. Instead, we are writing virtual components that will turn into the DOM leading to smoother and faster performance.

That's all for now…