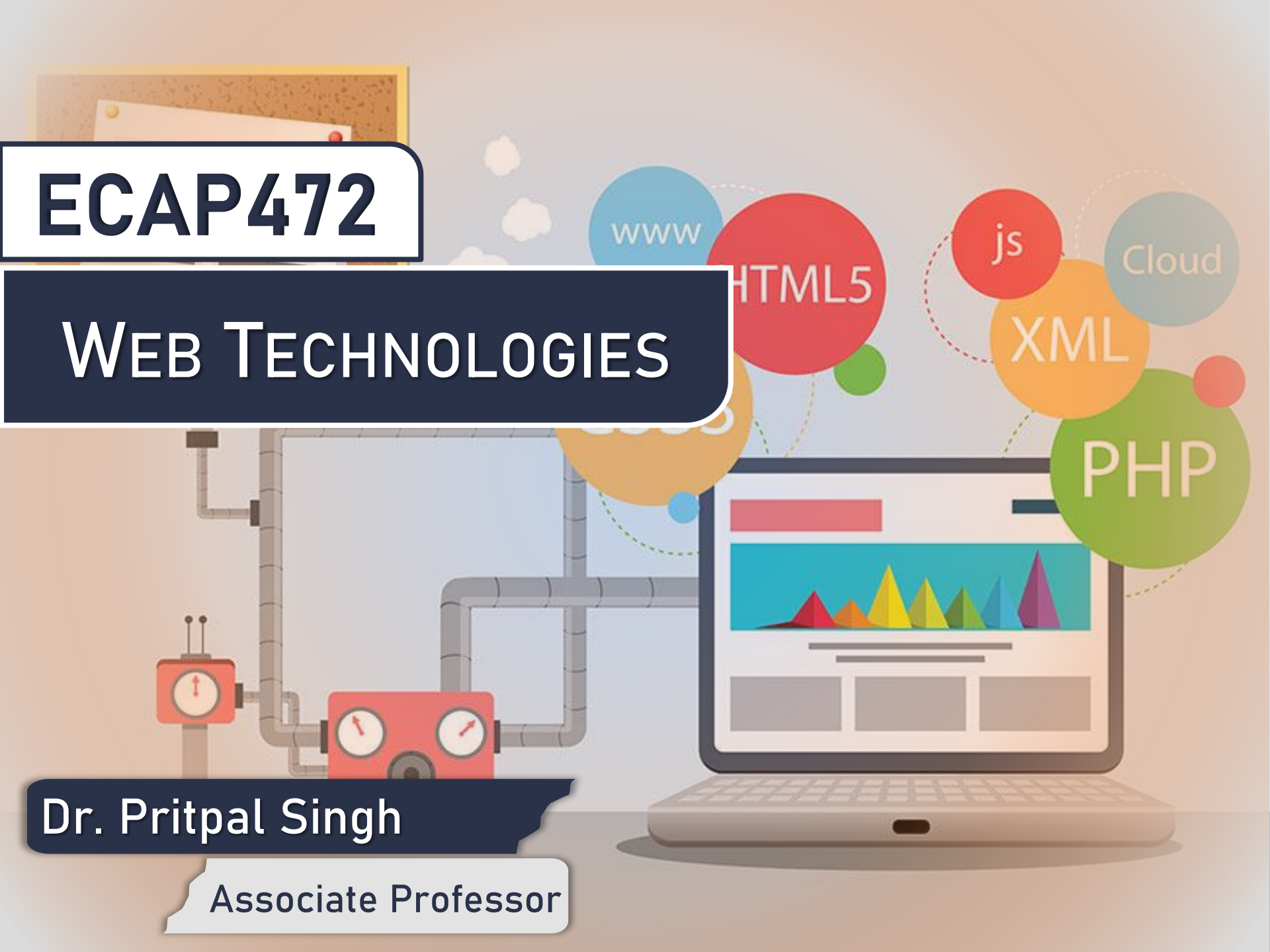
ECAP472

Web Technologies



Dr. Pritpal Singh

Associate Professor

Learning Outcomes

After this lecture, you will be able to

- understand concept and need of React Router .

React Router

- Routing is a process in which a user is directed to different pages based on their action or request.
- ReactJS Router is mainly used for developing Single Page Web Applications.
- React Router is used to define multiple routes in the application.

React Router

When a user types a specific URL into the browser and if this URL path matches any 'route' inside the router file, the user will be redirected to that particular route.

React Router

- React Router is a standard library system built on top of the React and used to create routing in the React application using React Router Package.
- It provides the synchronous URL on the browser with data that will be displayed on the web page. It maintains the standard structure and behavior of the application and mainly used for developing single page web applications.

Need of React Router

- React Router plays an important role to display multiple views in a single page application.
- Without React Router, it is not possible to display multiple views in React applications.
- Most of the social media websites like Facebook, Instagram uses React Router for rendering multiple views.

React Router Installation

React contains three different packages for routing.

React Router Installation

- react-router: It provides the core routing components and functions for the React Router applications.
- react-router-native: It is used for mobile applications.
- react-router-dom: It is used for web applications design.

React Router Installation

- It is not possible to install react-router directly in your application.
- To use react routing, first, you need to install react-router-dom modules in your application.
- The below command is used to install react router dom.
- $ npm install react-router-dom --save

Components in React Router

- The BrowserRouter, Route, Switch and Link are all components of the React-Router.
- These components are divided into three categories.
- The first category is routers, for example <BrowserRouter>. The second category is route matchers, such as <Route> and <Switch> and the last category is navigation, such as <Link>, and <Redirect>

<BrowserRouter>:

- BrowerRouter is a router implementation that has the ability to incorporate routing in react.
- It uses the HTML5 History API which include pushState, replaceState and the popState event to keep our UI in sync with the URL.
- It is the parent component that is used to store all other components and it uses regular URL paths.

<Route>:

- Route is the conditional component that renders a component based on the URL defined or the URL it is pointing to.
- In other words, it is a component that renders some UI when its path matches the current URL.

<Link>:

- Link component is used to create links to different routes and implements navigation around the application.
- Links accepts the to prop, which signifies where we want the link to navigate our user to.

<Switch>:

The switch component is used to render only the first route that matches the location rather than rendering all matching routes.

React Router and Client-Side Routing

- React Router is an API for React applications. Most current code is written with React Router 3, although version 4 has been released. React Router uses dynamic routing.
- When we say dynamic routing, we mean routing that takes place as your app is rendering, not in a configuration or convention outside of a running app. That means almost everything is a component in React Router.

Why use React Router?

- React Router, and dynamic, client-side routing, allows us to build a single-page web application with navigation without the page refreshing as the user navigates.
- React Router uses component structure to call components, which display the appropriate information.

Why use React Router?

- By preventing a page refresh, and using Router or Link, which is explained in more depth below, the flash of a white screen or blank page is prevented.
- This is one increasingly common way of having a more seamless user experience.
- React router also allows the user to utilize browser functionality like the back button and the refresh page while maintaining the correct view of the application.

What Happens When You Need To Navigate TWO Routing System?

An API is any place where a piece of code talks to another piece of code, but we often use it to mean somebody's external resource that gives me values, or our own internal database resource(s).

What Happens When You Need To Navigate TWO Routing System?

- If you are using a frontend and a backend, and you are potentially writing in multiple languages that don't necessarily have the same routing conventions, don't worry!
- The backend functions just as an API, and the user really doesn't interact with it at all.
- The routes that used to manage the user experience and the routes that used to manage queries to the database are not the same.

Add React Router

To add React Router in your application, run this in the terminal from the root directory of the application.

```
npm i -D react-router-dom
```

Folder Structure

Within the src folder, we'll create a folder named pages with several files:

- src\pages\:
  - Layout.js
  - Home.js
  - Blogs.js
  - Contact.js
  - NoPage.js
  - Each file will contain a very basic React component.

Basic Usage

Now use Router in our index.js file.

Example

- Use React Router to route to pages based on URL:
- import ReactDOM from "react-dom";
- import { BrowserRouter, Routes, Route } from "react-router-dom";

Basic Usage

Now use Router in our index.js file.

Example

- import Layout from "./pages/Layout";
- import Home from "./pages/Home";
- import Blogs from "./pages/Blogs";
- import Contact from "./pages/Contact";
- import NoPage from "./pages/NoPage";

Pros

- Routing between components is fast as the amount of data that renders is less.
- The rest of the data is rendered by the DOM, and even when there's tons of HTML and CSS to render, the DOM handles that part in the blink of an eye.
- Using lazy loading, any delay in rendering HTML is compensated for

Pros

For better user experience, animations and transitions can be easily implemented when switching between different components.

Pros

It gives a real sense of a single-page application in action. No separate pages are rendered, and the current page doesn't refresh to load a new view

Cons

The initial loading time is considerably large as all the routes, components, and HTML have to be loaded at once when the application first mounts . The whole website or web app needs to be loaded on the first request.

Cons

There is unnecessary data download time for unusable views that cannot be anticipated on the first render of the application

Cons

- It generally requires an external library, which means more code and more dependency on external packages, unlike routing on the server-side.
- Client-side routing and rendering convert JavaScript to HTML, making search engine crawling less optimized.

What is react-router-dom ?

- React Router DOM is an npm package that enables you to implement dynamic routing in a web app.
- It allows you to display pages and allow users to navigate them.
- It is a fully-featured client and server-side routing library for React.

What is react-router-dom ?

- React Router Dom is used to build single-page applications i.e.
- Applications that have many pages or components but the page is never refreshed instead the content is dynamically fetched based on the URL.
- This process is called Routing and it is made possible with the help of React Router Dom.

Practical

That’s all for now…