**React JS Industrial Notes**

**React Basics-**

React is an open-source JavaScript library for building user interfaces.

React is a project created and maintained by Facebook.

React has become increasingly popular among developers and is also one of the most sought out skill-sets by companies right now.

React has a component-based architecture. This lets you break down your application into small encapsulated parts which can then be composed to make more complex UI.

React will make it painless for you to create complex UIs by abstracting away the difficult parts.

React will handle efficiently updating and rendering just the right components in your application when your data changes.

DOM update which is one of the more expensive operations is handled gracefully in React.

**How does React work?**

React creates a VIRTUAL DOM in memory - Instead of manipulating the browser's DOM directly, React creates a virtual DOM in memory, where it does all the necessary manipulating, before making the changes in the browser DOM.

React only changes what needs to be changed - React finds out what changes have been made, and changes only what needs to be changed.

**React Features**

1. JSX − JSX is JavaScript syntax extension. It isn't necessary to use JSX in React development, but it is recommended.
2. Components − React is all about components. You need to think of everything as a component. This will help you maintain the code when working on larger scale projects.
3. Unidirectional data flow and Flux − React implements one-way data flow which makes it easy to reason about your app. Flux is a pattern that helps to keep your data unidirectional.
4. Simplicity- ReactJS uses JSX file which makes the application simple and to code as well as understand. We know that ReactJS is a component-based approach which makes the code reusable as your need. This makes it simple to use and learn.
5. 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. 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.

**React JS prerequisites-**

1. Basic familiarity with HTML & CSS.
2. Basic knowledge of JavaScript and programming.
3. Basic understanding of the DOM.
4. Familiarity with ES6 syntax and features.
5. Node.js and npm installed globally.

**Setup and Installation**

Let's begin by setting up our development environment to react. We need two things installed node js and a text editor of your choice.

For Node go to node.js org download and install the latest stable release if you already have it installed make sure to update it.

For text editor I recommend VS Code you can download and install it from https://code.visualstudio.com/download website.

**Create React App using Create React App CLI Tool**

This is a popular way of creating a single-page React application and we will use this method in all the react tutorials.

Create React App CLI tool is an officially supported way to create single-page React applications. It offers a modern build setup with no configuration.

Facebook has created Create React App, an environment that comes pre-configured with everything you need to build a React app. It will create a live development server, use Webpack to automatically compile React, JSX, and ES6, auto-prefix CSS files, and use ESLint to test and warn about mistakes in the code.

Step 1 - Create React App

To create a new app, you may choose one of the following methods:

npx

npx create-react-app firstapp

cd firstapp

**Files must exist with exact filenames**

For the project to build, these files must exist with exact filenames:

public/index.html is the page template;

src/index.js is the JavaScript entry point.

You can delete or rename the other files.

You may create subdirectories inside src folder. For faster rebuilds, only files inside src are processed by webpack. You need to put any JS and CSS files inside src, otherwise, webpack won’t see them.

Only files inside the public folder can be used from public/index.html.

package.json

The package.json file contains all the required dependencies for our React JS project. Most importantly, you can check the current version of the React that you are using. It has all the scripts to start, build, and eject our React app.

public folder

The public folder contains index.html. As react is used to build a single page application, we have this single HTML file to render all our components. Basically, it's an HTML template. It has a div element with id as root and all our components are rendered in this div with index.html as a single page for the complete react app.

src folder

In this folder, we have all the global javascript and CSS files. All the different components that we will be building

index.js

This is the top renderer of your react app. In the index.js file, we import React, ReactDOM, and the CSS file.

node\_modules

All the packages installed by NPM or Yarn will reside inside the node\_modules folder.

App.js

The App.js file contains the definition of our App component which actually gets rendered in the browser and this is the root component.

**Components**

In react, a component represents a part of the user interface. Components let you split the UI into independent, reusable pieces, and think about each piece in isolation.

**Component Types**

There are mainly two components in React:

Functional Components

Class Components

1. Functional Components

Functional components are basic JavaScript functions. These are typically arrow functions but can also be created with the regular function keyword.

Sometimes referred to as “dumb” or “stateless” components as they simply accept data and display them in some form; that is they are mainly responsible for rendering UI.

React lifecycle methods (for example, componentDidMount) cannot be used in functional components.

There is no render() method used in functional components.

These are mainly responsible for UI and are typically presentational only (For example, a Button component).

Functional components can accept and use props.

Functional components should be favored if you do not need to make use of the React state.

2. Class Components

Class components make use of ES6 class and extend the Component class in React.

Sometimes called “smart” or “stateful” components as they tend to implement logic and state.

React lifecycle methods can be used inside class components (for example, componentDidMount).

You pass props down to class components and access them with this.props.

**1. What is JSX?**

JSX stands for JavaScript XML.

JSX allows us to write HTML in React.

JSX makes it easier to write and add HTML in React.

JSX converts HTML tags into react elements.

**What are props?**

The props is short for properties and they are used to pass data between React components. React’s data flow between components is uni-directional (from parent to child only).

Props are immutable so we cannot modify the props from inside the component. We use props to pass data from parent component to child component and child component can't change it's value as it is immutable and owned by the parent component.

React props are like function arguments in JavaScript and attributes in HTML.

**React state**

The state is a built-in object in React components. In the state object, we store property values that belong to the component. When the state object changes, the component re-renders. The state object is modified with the setState() function.

The props serve a similar purpose to the state. Both are plain JavaScript objects. The difference between them is that props get passed to the component whereas the state is managed within the component.

**Handling events basics**

Handling events with React elements is very similar to handling events on DOM elements. There are few differences though.

React events are named using camelCase, rather than lowercase — onClick instead of onclick.

We write React event handlers inside curly braces.

In the case of React(/JSX), rather than passing a string, we pass a function as the event handler. onClick={buttonClicked} instead of onclick=”buttonClicked()”.

Full Stack Application-

Java Back End

React JS Front End

**Step 1:** Create React JS Application-

npx create-react-app hrmsoftware

**Step 2:** Adding Bootstrap in React-

npm bootstrap install –save

Open the src/index.js file and add the following code:

import 'bootstrap/dist/css/bootstrap.min.css';

src/index.js

**Step 3:** Consume CRUD REST API Call-

For our API calls, we will be using Axios. Below is the npm command to install Axios.

npm add axios

**Step 4:** package.json-

This file contains all the required dependencies for our React JS project. Most importantly, you can check the current version of React that you are using. It has all the scripts to start, build, and eject our React app

**Step 5:** React List Employee Component-

**Step 6:** Create Header and Footer-

**Step 7:** Configure Routing

npm install react-router-dom

**Step 8:** Add and Update Employee Component

**Step 9:** View Employee Component

**Step 9:** Run React App