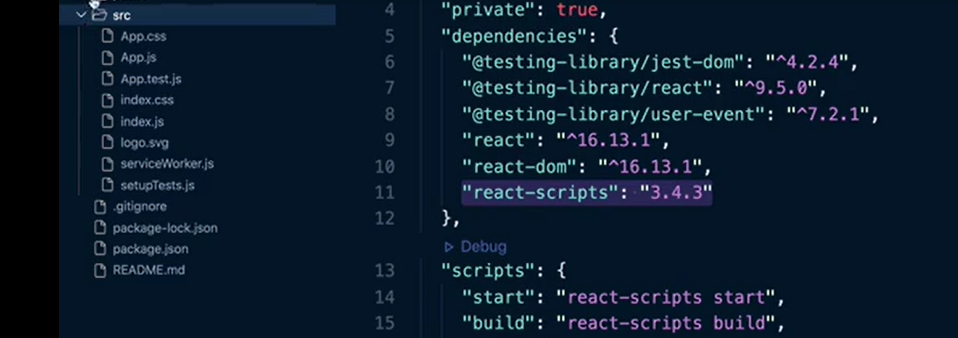
Source: <https://www.linkedin.com/learning/react-js-essential-training/>

* Install Node.js
* Create app using ***npx create-react-app My-Test-App-Name***
* react-scripts referenced in the package.json is the one that is used for different important activities
  + all sorts of tooling like babel and webpack they are configured using react-scripts
  + 
* **src** folder is where were write our application code, and **public** folder contains html files and other public assets, like icons and html files.
* **React.StrictMode**  throws warnings when our code is violating the best practices of react framework.
* **React.createElement –** first parameter is the tag to generate, second is the properties for the element, and third one is the content. In following screenshot style is being passed as one of the properties for the element
* To define a simple component, we use following code

function Header(){

return(

<h1> This is html of the component that we want to return</h1>

);

}

* We can receive props inside the components like this

function Main(props) {

return(

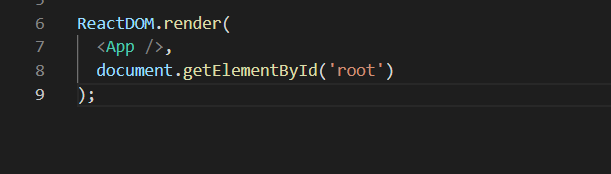
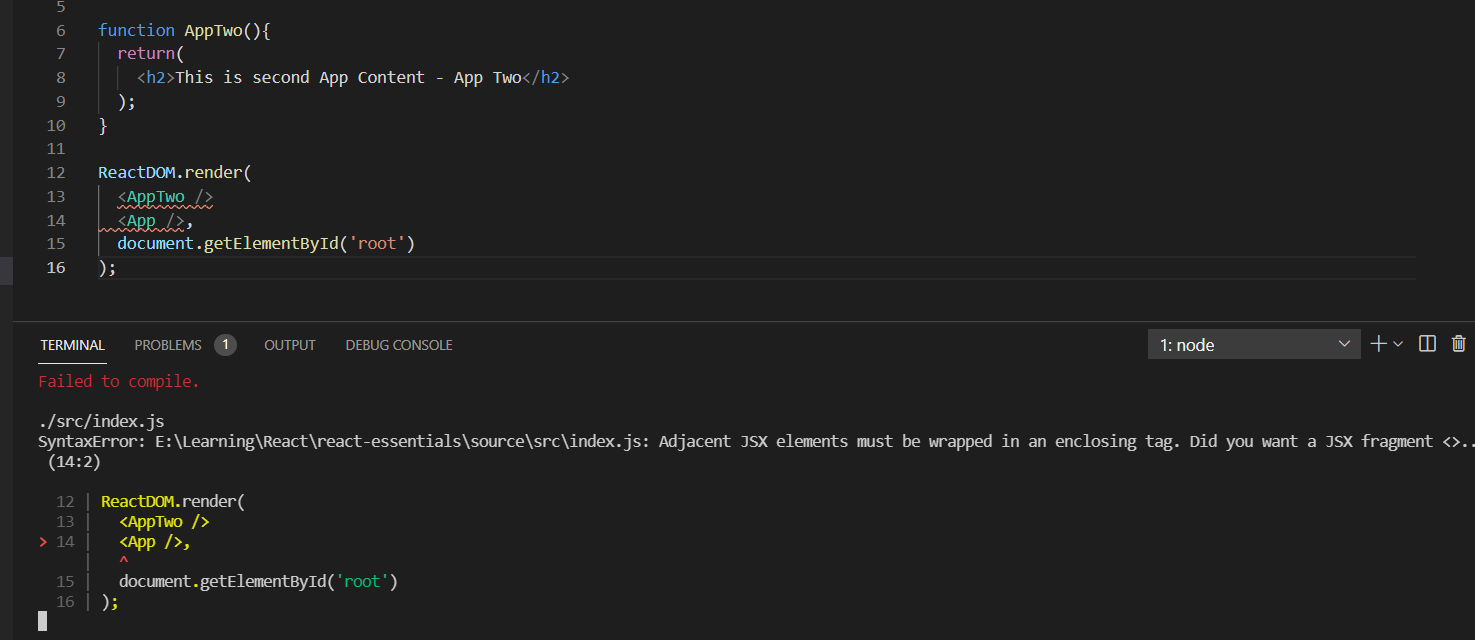
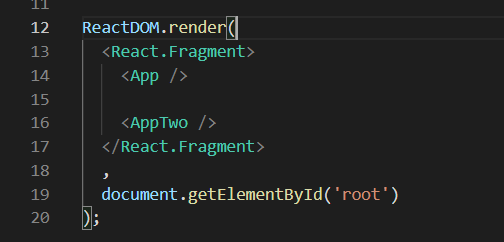
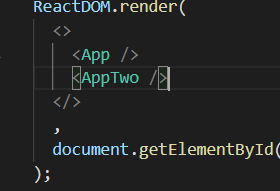
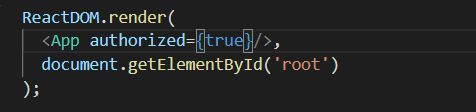
<p>Hello {props.name} </p>

);

}

* To set style property of a component we use JSX syntax with {} to pass in the objects like following, note first curly bracket is to start JSX object notation and brackets inside is Javascript object notation for passing in style object. Here we are not using css text-align, instead we use camel case because if use text-align this will break JSX

<ul style={{textAlign: “left”}}> </ul>

* 
* For lists, unique key is important. One way of doing that is assigning id of the map function inside the component to generate the unique key, and other way is to generate the unique id outside the component, part of the array being passed for lists, so this method is more reliable
* 
* We can use images in our components like we do text. First you have import the images from your src folder like import imageName from ‘./imageFileName.jpg’. Note imageName is something that you can use inside curly brackets inside your JSX syntax
* React.Fragment. When we have to render or return more than one elements from a React component, like for example inside ReactDOM.render method, usually we have an App object and then we have the target html element like this:
  + 
  + What if we want to add another AppTwo here…we get error: Adjacent JSX elements must be wrapped in an enclosing tag. Did you want a JSX Fragment
  + 
  + We can fix this error using React.Fragment
  + 
  + Or we can use simplified Fragment syntax like following
  + 
* Conditional Formatting: To use conditional formatting, we can use Fragment syntax <> {props.authorized ? <AuthorizedHeader /> : <DefaultHeader /> } </> like for example:
* 
* And 