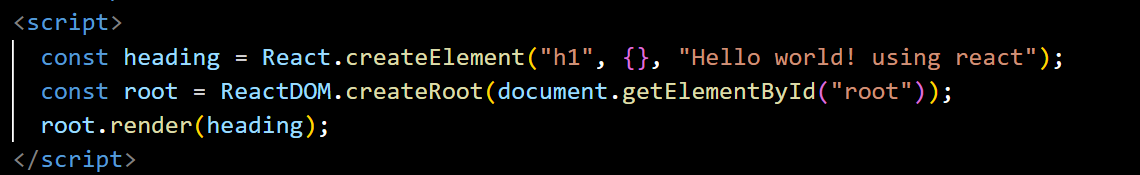
CDN: Content delivery networks helps to import the react code from the server and make our project configured to use react.

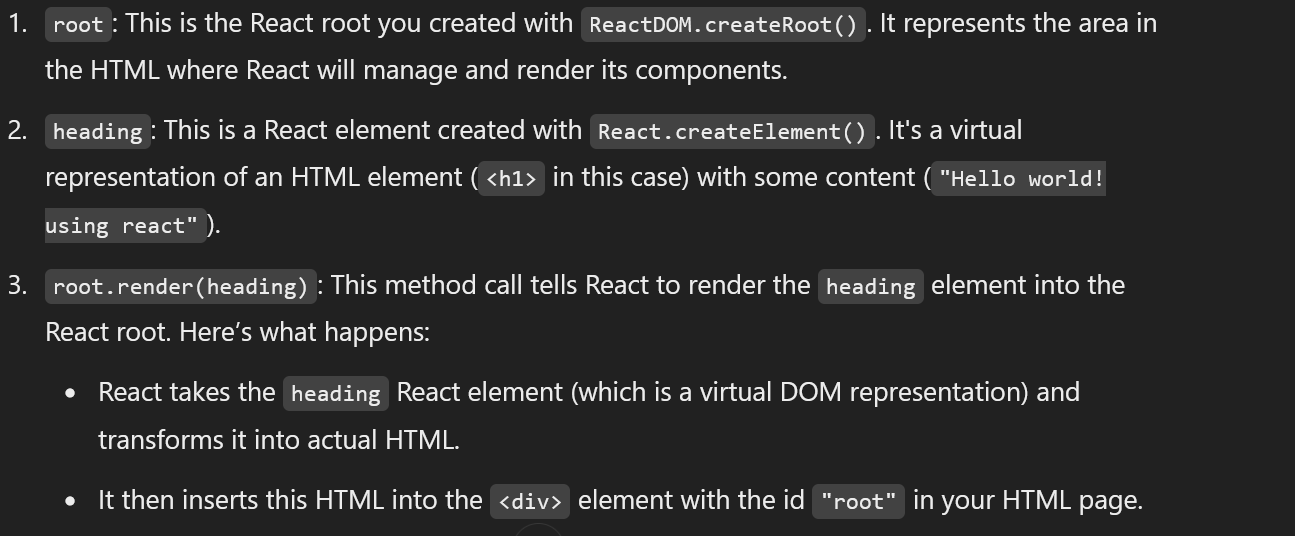
API: provides a way for 2 or more software applications to communicate with each other.

For example in JS there’s a creteElement API that lets you create an html element using JS.

It hides out the implementation details and provides a way to perform a certain task.



The {} is for giving attributes to our tags like an id to the h1 tag in the above example.



The 1 line is to create an element and put content inside it using react.

The 2 line is for creating a react root inside the html element with id=”root”( so to put all the react elements inside it ).

The 3 line is for actually placing the element created inside the html.

In summary, root.render(heading) is what places the virtual React element (heading) into the specified HTML element (<div id="root">), making it visible on the webpage.

See first what we create using the createElement is an react object. W

While rendering the the react object is converted to understandable html and puts it in the DOM.

DOM:

The costliest operation one can perform in a webpage is trying to manipulate the nodes of the DOM.

And every framework goal is essentially to optimize this.

The **DOM** is essentially a **tree-like structure** that the browser creates from the HTML (or XML) content. This structure makes it easier for developers to interact with, traverse, and manipulate the elements, attributes, and content of a webpage dynamically using **JavaScript**.

 The browser reads the static **HTML** file and turns it into a dynamic **DOM tree**.

 **JavaScript** allows developers to manipulate this tree, enabling them to:

* Change content (like text, images, etc.).
* Add or remove elements (like buttons, sections, etc.).
* Handle user interactions (like clicks, form submissions).

Modify styles (like colors, layouts).