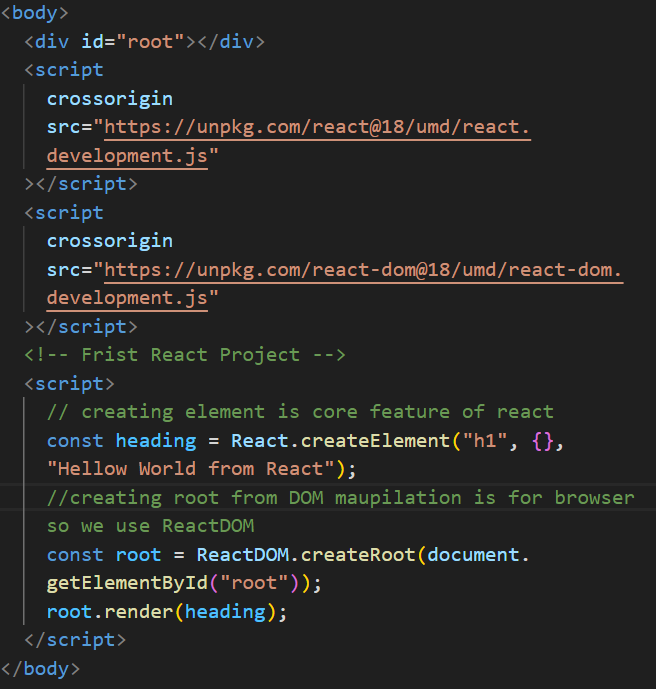
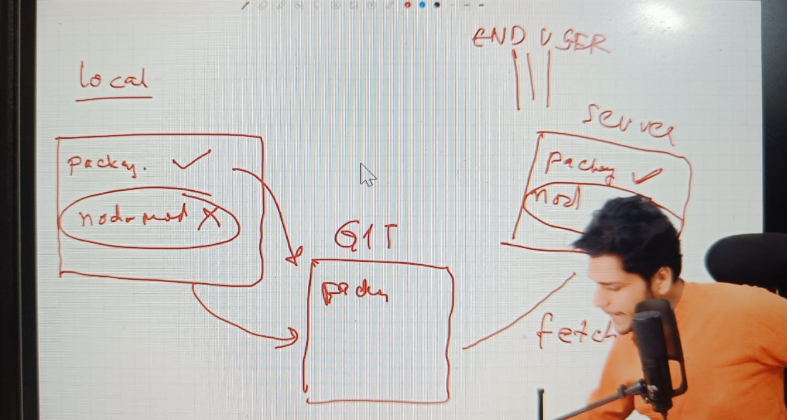
**Namaste - React**

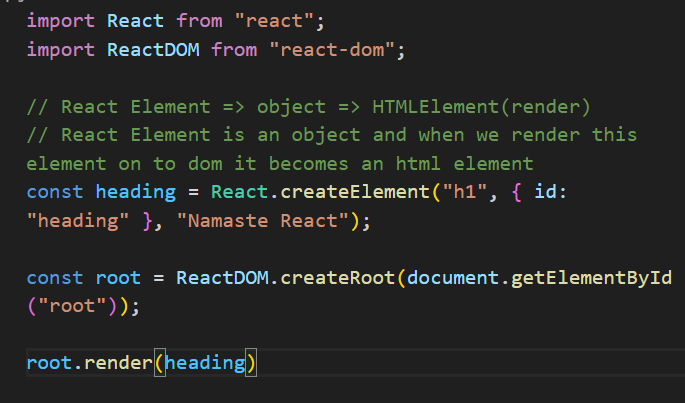
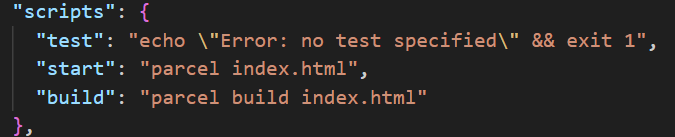
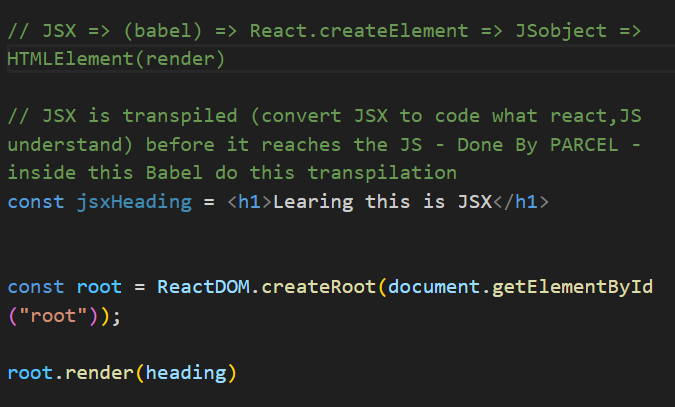
***Ep -01 || Inception***

* We can use react in our project using CDN - react script tag in html file , there are two file , one is react , 2nd is react-dom (used mainly for dom manipulation )
* React is just JS code , written by facebook eng
* 
* Here, heading is react element (not h1 tag) which is JS object and it have some props and other stuff. And root.render is responsible to convert it to h1 tag and put it to dom.
* This will became more complex and complex later on, if dom tree is big , that’s why JSX is introduced (JavaScript XML)
* React is a library , it can work on small portions of app.
* Ques - what Is CDN , what is crossorigin

***Ep -02 || Igniting our App***

* Npm full is not a Node Package Manager. (NPM don’t have full form) , NPM manages packages but it does not stand for node package manager , all the libraries and utility we need comes from npm and NPM manages that in our project.
* Package.json is configuration for npm. Why we need it - NPM manages all package that we install , sometimes this packages will also known as dependency and npm take care of these dependency like what its version and all in package.json.
* Most important package is bundler (bundle our app , compress) , vite , parcel , webpack are some example.
* There are two types of dependency – dev dependency , normal dependency(use in production also) [npm install -D parcel] -D is dev dependency.
* Package-lock.json – keeps track of exact version, which is installed.
* Node-moudles – fetches all the code of dependency and put it inside nodemodules
* I question – if we just install parcel – node modules should have only parcel in it why other many dependencies present -- this is because parcel has its own dependencies , its dependencies has its own dependency and so on.., that’s why node modules become huge and this is known as transitive dependency.
* If we want to install a package we write npm , if we want to execute a package we write npx.
* What is import React from ‘react’ , we are importing react from node modules.
* What is bundler do (ex parcel) – DevBuild , Local server , HMR – hot module replacement , file watching algorithm , Image optimization , Minification , compress , bundling , compress , consistent hashing , code spilting , differential bundling , diagnostics , tree shaking , different dev and prod bundles
* 

***Ep -03 || Laying the foundation***

* If we don’t want to write command again and again to start project in dev and prod , we make scripts for it for fast process inside package.json
* 
* When we do react.render it takes heading object and convert it to html element
* This way of creating element is tough so facebook developers created JSX – JavaScript Syntax helps to create react element
* JSX is not html in JS , it is HTML like syntax
* 

***Npm vs Npx --***

* Npm –
* npm is primarily a package manager for JavaScript. It is used to **install**, **uninstall**, and **manage dependencies** in your project.
* You use npm to execute scripts defined in the package.json file of your project (e.g., npm run start).
* When you run npm run start, npm looks for a start script defined in your package.json file and executes it.
* NPX – (Node Package eXecute)
* npx is a tool that comes with npm (version 5.2 and above) and is used to **execute binaries from the node\_modules/.bin folder** or temporarily download and execute packages **without installing them globally**.
* Unlike npm run, npx doesn’t look at your package.json scripts by default. Instead, it directly executes a binary or package.
* When you use npx <package-name>, npx checks if the package is available either **locally in your project’s node\_modules folder** or **globally**.
* If the package isn’t found in either location, npx temporarily downloads it to a cache and executes it, then deletes it afterward. This is helpful if you just need to run a tool once and don’t want to clutter your system with a global install or add it to your project dependencies.
* Let’s say you want to use the create-react-app tool:
* **With npm install -g create-react-app**: create-react-app is globally installed, so you can run create-react-app from anywhere on your system.
* **With npx create-react-app my-app**: npx will download and execute create-react-app (if not already installed globally), allowing you to use it without permanently installing it.