Page No.
Date

Inside { } we can write anything

```
{
  id : "container"
  hellow : "world"
}
```

These are called "props"

props can be anything

To make an app production ready, we should :-

(1) minify our file. (removing our console log, bundle thing up)

(2) need a server to run things.

"Minify → optimization → clean code

↓

"Bundle".

Page No.
Date

In React, to get External functionalities, we can use "BUNDLERS" :-

a) webpack is a bundler
b) vite
c) parcel

} These are alternatives

Most bundlers do the same Job

Bundlers are packages. If we want to use a package in our code, we have to use a "package Manager"

we use a package manager know as "npm" or "yarn"

npm install -D parcel
- D → dev dependency - we want it in our developer machine
parcel → it is a one of a Dependency

[caret (^) & tidle sign (~)]

↳ our project will automatically update if we use caret sign (^) or tidle sign (~)

```
"devDependencies": {
    "parcel": "^2.8.2"
}
```
caret

Page No.
Date

package-lock.Json

→ will tell you which Exact version of the Library you are using.

→ The purpose of the 'package-lock.Json' file is to Ensure that your project uses the same dependencies.

COMMON ISSUE

Q = some things is working on my localhost / my machine but not work in production. why?.

→ package-lock files tells the Exact version of the 3rd Library we are using. suppose for the dev dependancy we're using 2.8.4 version & right written in package-lock.Json file.
But,
in in 'package.Json file' file it will be like "2.8.2".

package-lock files is an x file that Locks the version

→ Never put packet-lock.Json in gitignore

Page No.
Date

"node_modules"

→ which gets installed is like a database for the NPM.

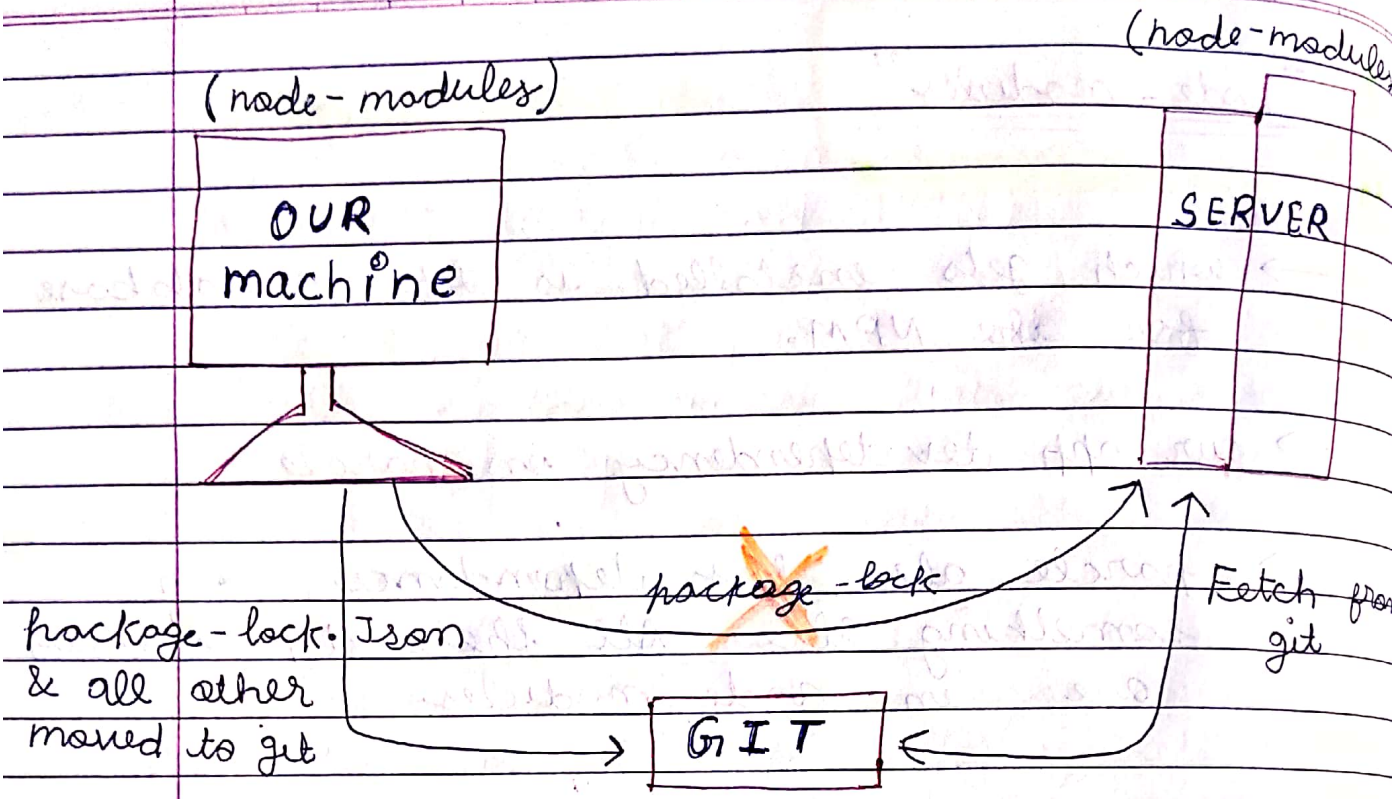
→ our app dependency on parcle

→ parcle also has dependancies on something else. All these dependancies are in node modules

Q we don't put 'node_modules' into git. why ?

→ Because our package-lock.json file have sufficient information to recreate a node-module.

→ package-lock.json files keep & maintain the version of Everything in node modules.



→ node-module in our machine can be regenerated in server using the package-lock.Json file.

```
npm install react
```

Never touch "node-modules" & package-lock.Json

Page No.
Date

npx parcel index.HTML → it is the entry point

→ means "execute using npm"

npx = node package execute

click Enter

- Then, a miniserver is created for us, like localhost : 1234.59.27
- parcle given a server to us

As we removed CND links, we don't have react in our app.

So, we import it into our app

For that we use the keyword "import"

In JS file

```
import React from "react";
import ReactDOM from "react-dom/client";
```

As we get got Error, we have to specify to the browser that "we are not using normal script tag, but as a module

```
<script type="module" src="App.JS"></script>
```

Page No.
Date

Note :-

* Hot Module ~~Reload~~ Replacement (HMR)
  → Means that parcel will keep track of all the files which you are updating

* How HMR works ?

→ There is File watcher algorithms (written in C++). It keep track of all the files which are changing realtime & it tells the server reload.

→ These all tt are done by parcel.

* parcel-cache

→ There are folder called parcel-cache which will be there automatically. In our project, parcel need some space so, it create parcel cache.

* dist folder

→ dist folder keeps the files minified for us. when we run command:

```
npx parcel index.html
```

→ This will create a faster development version of our project & serves it on

Q what takes a lot of time to load in a website?

A Media - images

Page No.
Date

when I tell parcel to make a production build:

```
npx parcel build index.html
```

↳ it will create a lots of things, minify your file and "parcel" will build all the production files to the dist folder.

* parcel also does image optimization
* parcel also take care of your older version of browser
* parcel also does caching while development
* parcel gives us a functionality that we can just build our app on HTTPS on dev machine

```
npx parcel index.html --https
```

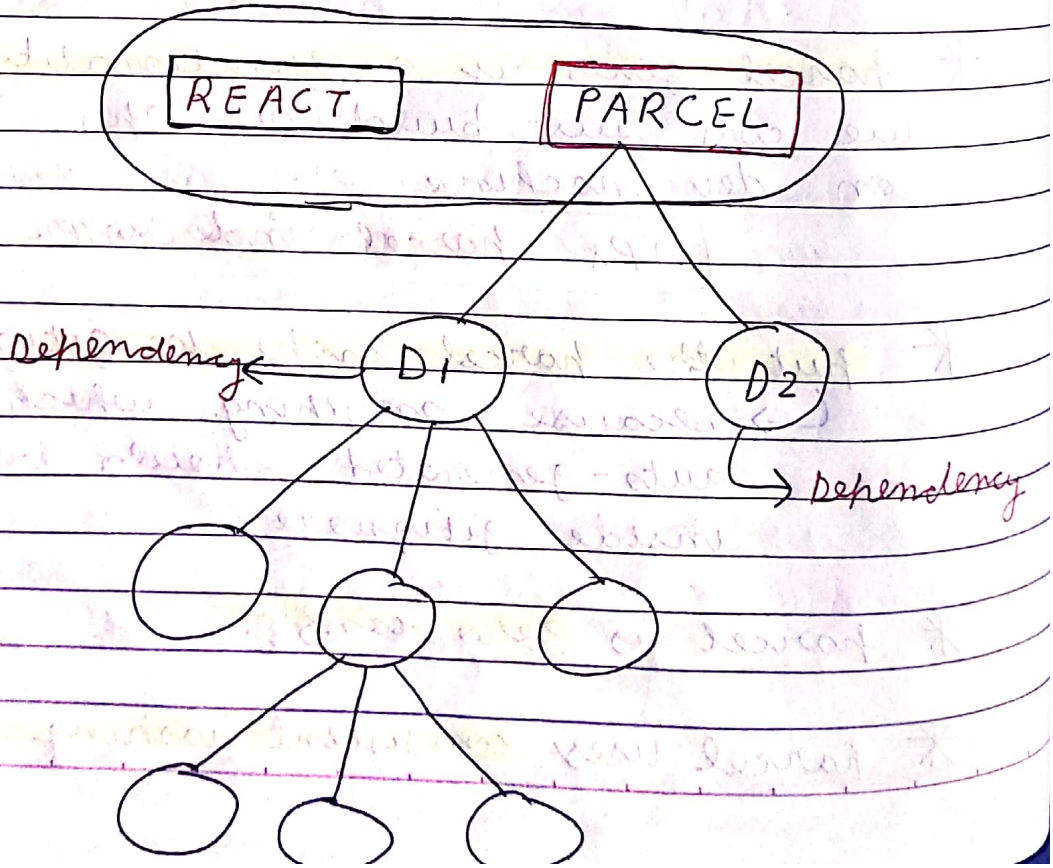
* put the parcel-cache in gitignore
  ↳ because, anything which can be auto-generated should be put inside gitignore
* parcel is zero 'config'
* parcel uses consistent hashing algorithms

TRANSITIVE DEPENDECIES

we have own package manager which handles and take care of our transitive Dependencies of our code.

If you have to build a production ready app which use all optimization (like minify, cleaning bundling, compression, consistent hashing, etc) we need to do all these,

But we can't do this alone we need some dependencies on it. These dependencies are also dependend on many other dependeneis.



Page No.
Date / /

Q How do I make our app compactible with older browser.

→ There is a package called "browserlist" & parcel automatically give it to us.

→ Browserlist makes our code compactible for a lots of browser.

go to "browserlist.dev"

In package.Json file, do :-

```
"browserlist": [
    "last 2 versions"
]
```

→ support 74%

Loaded with some configurations

means my parcel will make sure that my app work in last 2 versions of all the browser available.

If you don't care about other browser except ~~or~~ chrome

```
"browserlist": [
    "last 2 chrome version"
]
```

→ support 16%

→ Tree shaking

☆ parcel has this superpower

☆ means removing unwanted code.

Example :- suppose your app is importing a library which has a lots of function (say, 20 helper function). Then all of those 20 function will come into code. But in our app I may want to use only 1 or 2 out of it.
Here, PARCEL will ignor all the unused code.

create-react-app : uses "webpack" along with babel.

Q How can you build a performant - web scalable app?

→ There are so many things that react optimizes for us and parcel (bundler) given us.

→ our whole application is a combination of all these things.