to install dev dependencies --save-dev

npm i -D parcel OR npm i --save-dev parcel

Day 2 - Igniting our App

We will build our own 'create-React-app'.

Code A :-

```
⇒ Const heading = React.createElement (
        "h1",
        {
            id : "title",
        },
        "Heading 1"
    );
```

⊗ This code is exactly similarly to :-

Code B :-

```
<h1 id = "title"> Heading 1 </h1>
```

ie, code A will produce code B in our DOM.

If I want to create div having 2 children h1 and h2, I'll do it like this :-

```
const container = React.createElement (
    "div", // createElement takes first argument as the tag
    { id : "container", // second arg is the HTML attributes.
    },
    [heading1, heading2] // third arg. is children
);
```

Date.....

The above code will be like this ↓ inside DOM

```
<div id="container">
    <h1 id="title">Heading 1 </h1>
    <h2 id="title">Heading 2 </h2>
</div>
```

Inside second argument, we can write anything

```
{
    id = "container"
    hello = "world"
}
```

These are called 'props'
Props can be anything.

To make an app production ready,

we should :-

1) minify our file (remove our console logs, bundle things up)
2) need a server to run things.

Even through we can load our "App.js" we can't get optimised version.

It is a development tool that combines many JavaScript code files into a single one that is production-ready loadable in the browser

In react, to get external functionalities we use "BUNDLERS" :-

a) Webpack is a bundlers
b) Vite
c) parcel

These are alternatives

dependencies → transitive dependencies
↳ k dependencies

Date.....

In create-react-app, the bundler used is "webpack".

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 'package mangers' such as
'npm' or 'yarn'

npm :- No Problem Man

'npm' doesn't mean node package manager but everything else.

npm init (create a package.json file)

We use npm because we want a lot of packages in our project/react app

[npm init -y] → will skip a lot of question nd put default values.

npm i -D parcel → devdependency → because we want it in our developers machine
parcel is one of the dependency

Then, we'll get package-lock.json.

[caret & tidle sign] → ^4.18.6 → 4.x....
~4.18.6 → 4.18....

Our project will automatically update if we use caret sign (^).

"devDependencies": {

"parcel": "^2.8.2"

}

Spiral

Teacher's Sign .....

Date.....

package-lock.json

will tell you which exact version of the library you are using.

"node modules"

→ It is like a database for the npm.

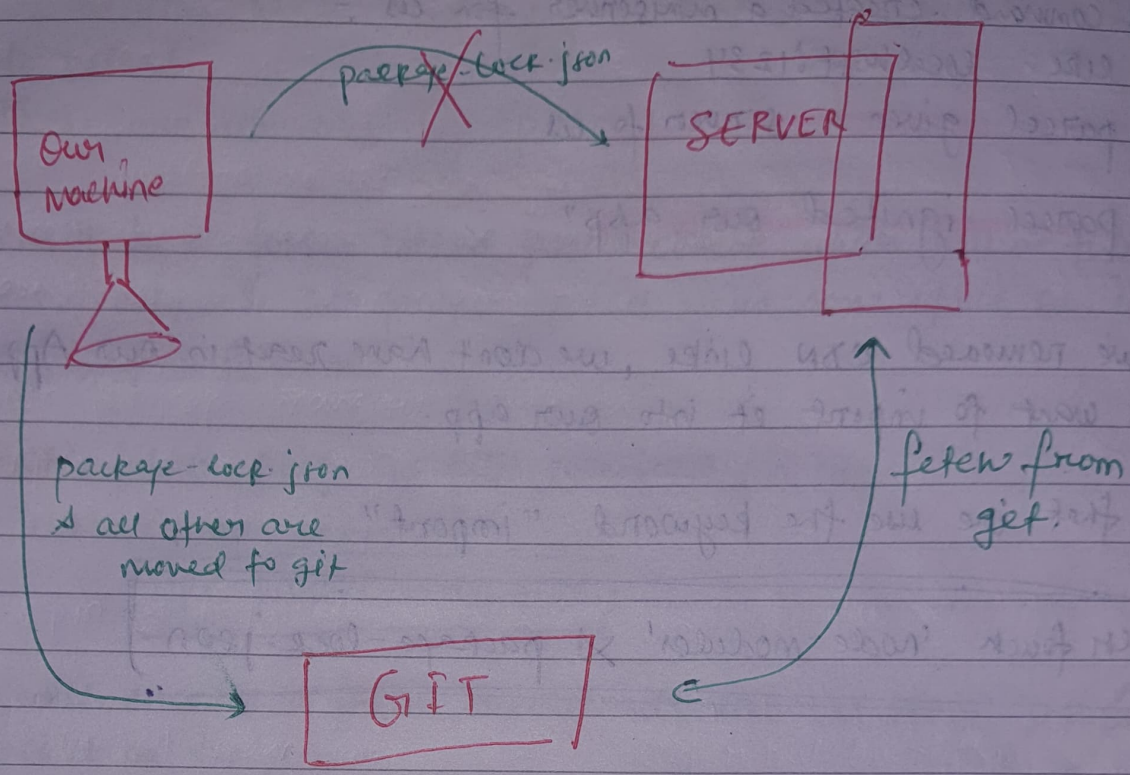
→ This is where the superpower comes from

→ Our app has dependency on 'parcel'

→ 'parcel' also has dependencies on something else
All these dependencies / superpowers 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 node_modules.
→ package-lock.json file keep & maintain the version of everything in node_modules.

Date.....



→ 'node-modules in our machine can be regenerated in server using the package-lock.json file.

Previously, we used CDN links to get react into our app. This is not a good way. We need to keep react in our node-modules.

```
npm i react
```

To ignite our app

```
npx parcel index.html
```

npx → means execute using npm

index.html → is the entry point

Date.....

This command created a miniserver for us :-
like localhost:1234
So parcel given a server to us

"parcel ignited our app"

As we removed CDN links, we don't have react in our App.
So, we want to import it into our app.

For that we use the keyword "import"

> Never touch 'node_modules' & package-lock.json

Normal js browsers don't know "import"
so it shows error

```
//code in App.js
import React from "react";
import ReactDOM from "react-dom/client";
```

→ now add this in update & boom.

As we got an error, we have to specify to the browser that we are not using a normal JS file but this is module.

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

→ more details in assignment 3

~~We can't use import & export script~~

Note:-
* Hot Module Replacement (HMR)
→ Means that parcel will keep a track of all the files which you're updating.
* How HMR works?
→ There is File Watcher Algorithms (written in C++). It keeps track

Date.....

of all the files which are changing realtime & it tells the server to reload.

→ These are all done by "PARCEL"

* There'll be a folder called parcel-cache which will be there automatically. In our project, parcel needs some space. So, it creates parcel-cache.

* 'dist' folder keeps the files minified for us.
when we run command

`npx parcel index.html`

This will creates a faster development version of our project & serves it on the server.

When I tell parcel to make a production build

`npx parcel build index.html`

It creates a lot of things, minify your file. And parcel will build all the production files to the dist folder.

In dist file apart from the helper file we only have 3 important file

index.782659.html
index.9cfa.css
index.ab597.js
} main compressed file of our app.

index.cc789.css.map
index.f9abc.js.map
} helper files

Spiral

Teacher's Sign.....

Date.....

Q what takes a lot of time to load in a website?
⇒ Media — Images

"Parcel does image optimization also"

* Parcel also does "caching while development".
  ↳ isse bhi subsequent build time kam ho jaata hai

* Parcel also takes care of your older version of browsers.

Compatible with older version of browsers

Sometimes we need to test our app on https because something only works on https.

Parcel gives us a functionality that we can just build our app on https. on dev machine.

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

* should put the parcel-cache in gitignore
  ↳ because, anything which can be auto-generated should be put inside gitignore.

* Parcel uses :-

```
consistent Hashing Algorithms
```

* Parcel is 'Zero Config' → for other bundlers we need to config so many things before installing it but for parcel we need nothing

Date.....

Parcel features in a glance :

- → HMR - Hot Module Replacement
- → File watcher algorithm - C++
- → Bundling
- → minify code
- → cleaning our code
- → Dev and production build
- → Super fast build algorithm (caching se wajah)
- → Image optimization
- → Caching while development
- → Compression
- → Compatible with older browser version
- → HTTPs on dev
- → Port Number
- → Consistent Hashing algorithm
- → Zero config.
- → Tree shaking.

Transitive Dependencies

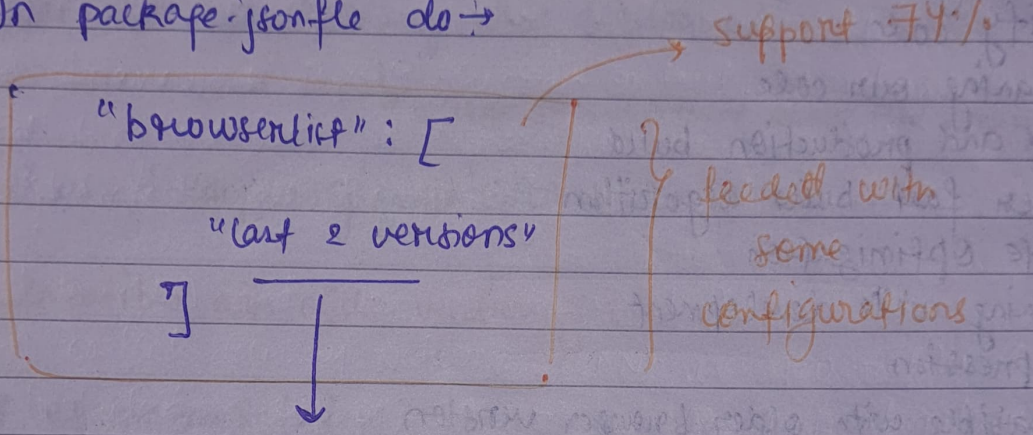
We have our package manager which handles and takes care of our transitive dependencies of our code.
If you've to build a production ready app. which uses all optimisation (like minify, cleaning, bundling, compression, consistent hashing etc) we need to do all these. But we can't do this alone. We need some dependencies on it. Those dependencies are also dependend on many other dependencies.

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

Sol: There is a package called 'browserlist' & parcel automatically gives it to us. Browserlist makes our code compatible for lots of browsers.

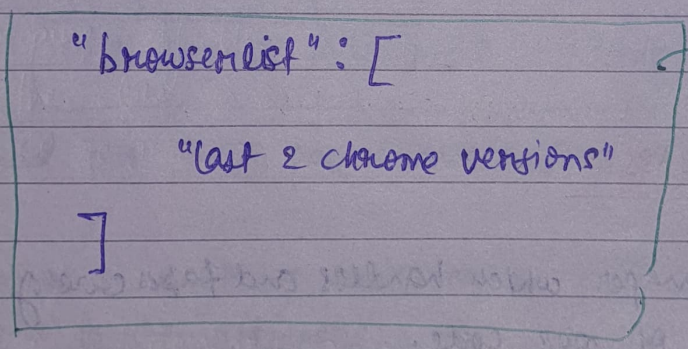
go to 'browserlist.dev'

In package.json file do →



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

If you don't care about other browsers, except chrome



FINALLY

"Parcel is a beast"--

Date.....

→ Tree Shaping → [removal of dead code] (elimination)

→ parcel has this superpower
→ means removing unwanted code.

Eg: Suppose your App is importing a library which has a lot of functions (say 20 helper func"). Then, all those 20 functions will come into your code. But in my App, I may want to use only 1 or 2 out of it.

Here, PARCEL will ignore all the unused code.

→ Create-react-app: uses 'webpack' along with 'babel'.

Q. How can you build a performant-web-scalable app? (Bundlers)
→ There are so many things that react optimizes for us and parcel gives us.
→ Our whole application is a combination of all these things.

Assignment - 1

Date.....

Q. what is Emmet ?

Soln:- Designed to speedup the process of writing and editing code by providing a set of shortcuts that can be quickly expandable to full code blocks

Q. Difference between a library and framework ?

Soln:- A framework inverts program control. It informs the developers of what they require. A library however, does not instead, a programmer calls the library when and where he needs it.

Q. what is CDN and why do we use it ?

Soln:- A content delivery Network (CDN) is geographical distributed group of servers which work together to provide fast delivery of internet content.
It helps to lower the server load also reduce the response time of request.

Q. Why React is Known as react ?

Soln:- because of its ability to react to changes in data.
when the data in a React component changes, React will automat-ically re-render the component so that it reflects the new data.

Q what is crossorigin in Script tag ?

Soln:- [CORS (Cross origin resource sharing) is an HTTP. header based mechanism that allows a server to indicate any cross origins (domain, scheme or port) other than it's own from which a browser should permit loading resources.]

In Simple word CORS refers to the method that allows you to make requests to the server deployed at a different domain.

Teacher's Sign .....

Spiral

Date.....

Q. what is ReactDOM?

Sol:- ReactDOM is a package that provides DOM specific methods that can be used at the top level of a web app to enable an efficient way of managing DOM elements of the web page.

Q. why use async and defer?

Sol:- Defer is used for scripts that need the whole DOM and for their relative execution order is important. And async is used for independent scripts like counters or ads.

Assignment-2

Q what is 'NPM'?

Sol:- It is a tool used for package management and default package manager for Node packages. NPM is installed when NodeJs is installed in the machine.

Q what is Parcel/webpack? why do we need it?

Sol:- Parcel/webpack is type of a web application bundler used for development and productions purposes on power our application with different type functionalities and features.

Parcel features:

- HMR (Hot module replacement)
- File watcher algorithm (made with C++)
- Minification
- cleaning our code
- DEV and production Build
- Super fast building algorithm
- image optimization.
- caching while development

Spiral

Teacher's Sign .....

Date.....

- compress
- compatible ~~with~~ with older version of browser. (polyfills)
- HTTPs in dev
- Port Number
- Consistent hashing algorithm
- Zero configuration
- Automatic code splitting

Q. what is .parcel-cache ?

Soln:- It is used by parcel (bundler) to reduce the building time. It stores information about your project when parcel builds it, so that when it rebuilds, it doesn't have to re-parse and re-analyze from scratch. It's a key reason why parcel can be so fast in development mode.

Q. what is npx ?

Soln:- npx is a tool that is used to execute the packages. It comes with the npm, when you installed npm above 5.2.0 version then automatically npx will installed. It is a npm package runner that can execute any package that you want from npm registry without even installing that package.

Q. what is difference between 'dependencies' vs 'devDependencies' ?

Soln:- A dependency is a library that a project needs to function effectively. DevDependies are that packages a developer needs during development.

Q. what is Tree shaking ?

Soln:- Tree shaking is process of removing the unwanted code that we do not use while developing the application. In computing, tree shaking is a dead code elimination technique that is applied when optimizing code.

Spiral

Teacher's Sign.....

Date.....

Q. What is Hot module Replacement?

Sol:- Hot module Replacement (HMR) exchanges, adds, or removes modules while an application is running, without a full reload. This can significantly speed up development in a few ways. Retain application state which is lost during a full reload.

Q. List down your favourite 5 superpowers of Parcel and describe any 3 of them in your own words.

Sol:- 5 superpower of parcel are

- HMR (Hot module Replacement) :- It exchanges, adds or remove the modules while an application is running, without a full reload.
- File watcher algorithm :- File watchers monitor directories on the file system and perform specific actions when desired files appear.
- Minification :-
- Image optimization
- Caching while development :- It cache files used by parcel to fasten the development and production built time. It stores information about your project when parcel builds so that when it rebuild it doesn't have to re-parse & re-analyse from scratch.

Q. what is .gitignore? what should we add not add into it.?

Sol:- The .gitignore file is a text file that tells Git which files or folders to ignore in a project during commit to the repository. you should ignore the files that do not need to get committed.

Q. what is the difference between package.json and package-lock.json.

Sol:- package.json file contain the dependencies with their required version in the project, where as pack-lock.json contain current installed version of dependencies and lock their version.

^4.2.1 ⇒ 4.X.X
~4.2.1 ⇒ 4.2.X

Date.....

Q. Why should I not modify package-lock.json?

Solu: file contains the information about the dependencies and their versions used in the project. Changing it or deleting it would cause dependencies issue in the production environment. So don't modify it, it's being handled automatically by NPM.

Q. what is node-modules? Is it a good idea to push that on git.

Solu: The node_modules folder is used to save all download packages from npm. It take care of the dependencies of your project as well as their transitive dependencies.
No it's not a good idea to push your node module on git. as you can create it using package.json and package-lock.json

Q. what is 'dist' folder?

Solu: The /dist folder contains the minimized version of the source code. The code present in the /dist folder is acctually the code which is used on production web application.

Q. what is browserslist?

Solu: Browserslist is a tool that allows specifying which browsers should be supported in your frontend app by specifying "queries" in a config file. It's used by frameworks/libraries such a react, Angular and Vue, but it's not limited to them.