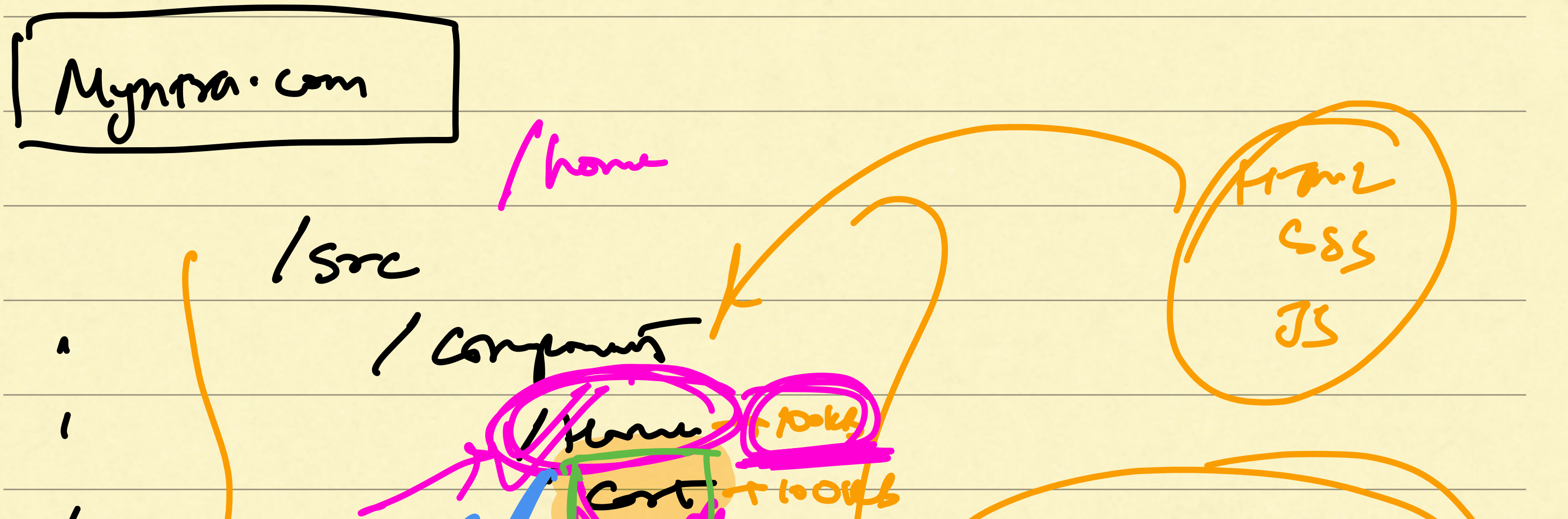
33. React Build Tools, List Forms, Props & State

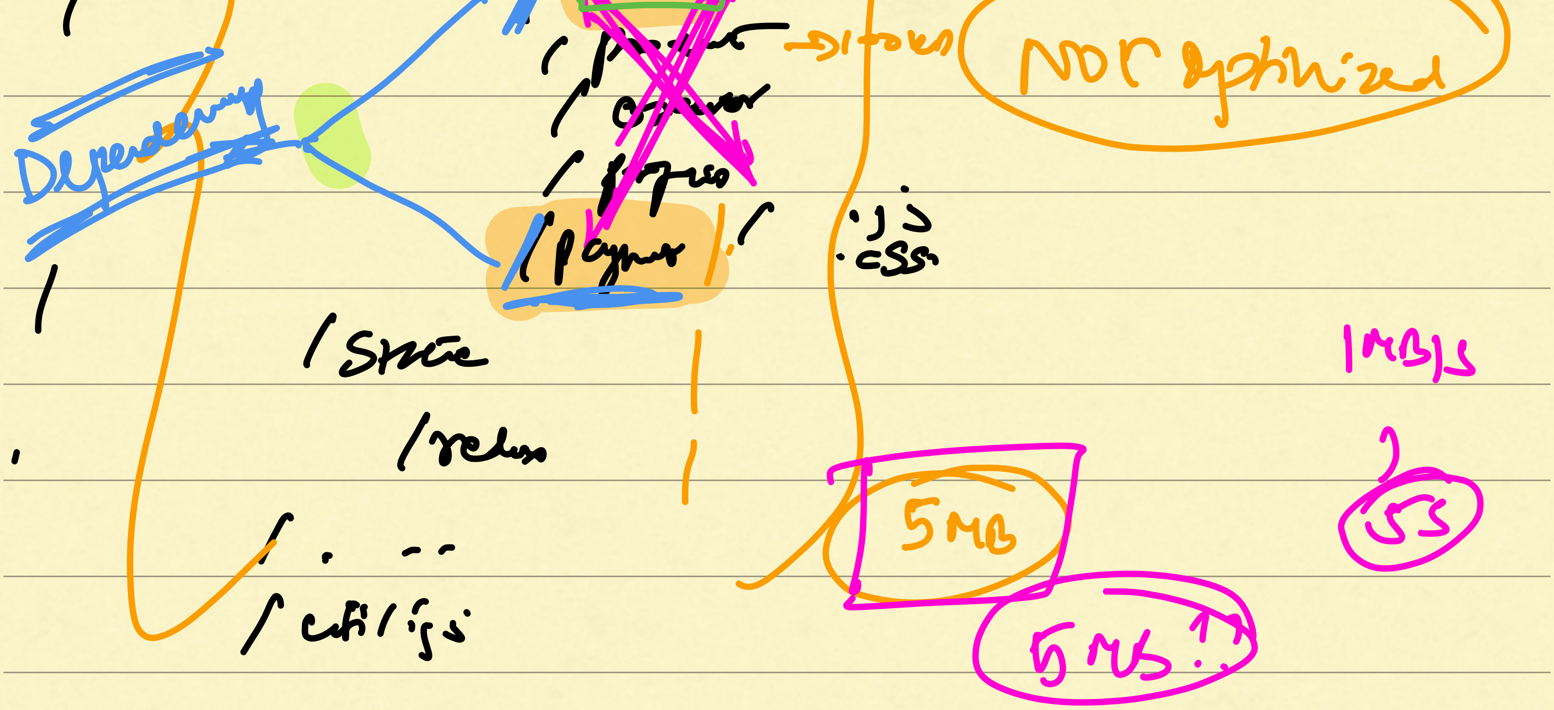
Agenda:

- Build tools
- Components & reuse
- Props
- State
- Event Handling in React

BUILD TOOLS:

Transforming, bundling & optimizing source code to make it ready for deployment.





→ Use cases of Build tools

1) Transpilation → Build tools like Babel transpile the code into backward compatible version that browsers support

2 Code Splitting → Different parts of application can load asynchronously.
→ They automatically analyze dependencies & split code accordingly

3 Module Bundling → Build tools like webpack, rollup, parcel bundle these modules into a single or multiple files. to improve load times,

Learn → multiple modules.

reduce HTTP req.

4) Optimization → Perf. & efficiency.

5) Deployment Server → Tools have builtin server.

HMR → Hot Module Replacement

Eg

- Webpack
- Vite
- Parcel
- Rollup

Short notes on differences

Package.json

dependencies :

{

react : 18.3.1,

react-Dom : 18.3.1

}

libA1

A2

A3

libA

libB

libC

devDependencies : {
 . .
}

Project Structure

1 package.json : Lists all dependencies, metadata to run code

2 package-lock.json : - Automatically gets created.
- Lists all dependencies & used packages

3 node_modules - Automatically created
- Houses 'code' of packages used.
- All installed packages & their dependencies.

4 src
└ index.html
App.js
App.css

Package.json :

dependencies : { . . - } ,

devDependen : { . - . } ,

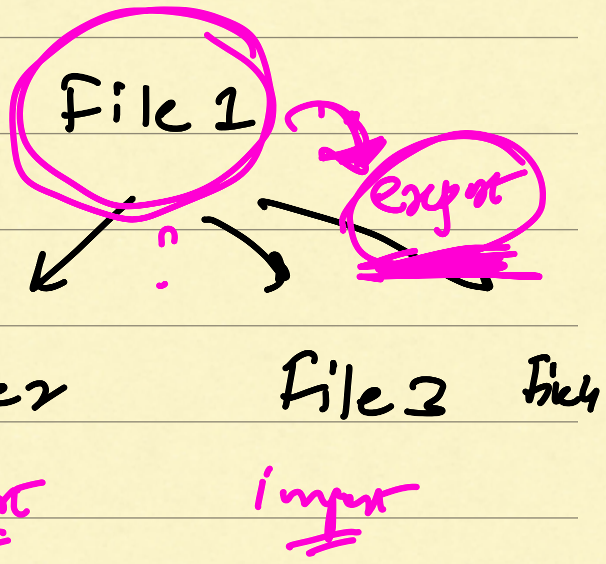
scripts : {

}

export

① default export

② named export



One file can have one default export

named export can be many.

Default

```
import MyComp from './MyComp'
```

```
import {a,b,c} from './myComp'
```

named

Why props?

- Extensibility
- Reuse the Component
- Separation of Concerns
- Efficiency

arr

```
[ 'Apple', 'Banana', 'Mango' ]
{
<ul>
arr.map((item, index) => {
        return (
                    <li > {item} </li>

                ,
        )
</ul>
```