**Theory Chapter 02**

**Q: What is `NPM`?**

**A:** NPM is a package manager. Its full form is not node package manager. Even on the official site one can check there is nowhere mention that it is node package manager**.** It helps to manage all the packages which are required in our React app. It not only helps to download a package but it can also be used to manage versions and package dependency (transitive dependency).

**Q: What is `Parcel/Webpack`? Why do we need it?**

**A:** Parcel/webpack are module bundlers. They are not only needed to bundle our javascript files but also plays an important role in optimizing our app along with react or any other framework/library. Below are the few points which Parcel does

1. **Hot Module Replacement:** when any file is changed in the code , it automatically creates a new build for us and refresh the page for us. But in some cases it replaces some modules in the browser without refreshing the whole page and our app state is also maintained . This is called hot module replacement. It replace a modules and then starts replacing all its parent module. CSS changes are done in HMR fashion.
2. **File Watcher: T**o perform optimal use of caching and development process , parcel watch every file in your project including nod modules. These algo are written in C++ language. Based on events and meta data parcel knows which file is changed and rebuild that file
3. **Cleaning our code**
4. It does not remove console by default. Once has to configure in .bable.rc to do
5. No initial config needed to setup
6. **Tree Shaking:** Removal of dead code/unused code from a module while bundling of multiple javascript files into while .Pracel does this automatically .It depends upon the import and export statement to detect if the code is used in other files or not
7. **Image Optimization**
8. **Bundling**
9. **Code splitting: P**arcel supports zero config code splitting . This means we can form multiple bundles on demand of particular pages so that our initially bundle while be smaller in size which results in faster load of our app . Dynamic import helps us to split our code , dynamic import gives a promise that a module can be loaded async.
10. **Dev Server:** when one runs the default cmd of parcel , it automatically starts a dev server for us. By default it is 1234 but on can easily override it using –port in the cmd. One can also add –open in the cmd so the dev server is open in which browser .for eg to open the server is safari one can add --open in cmd. Even on can also override the host name using --host in the cmd
11. **Minification :** Pracel includes minfiser for jS , html ,css , SVG . It reduces the size of the files so that size of the output bundle is small . It removes white space , renaming of variable and many more things for minification .It uses terser for js , htmlnano for HTML , svgo and lightining css
12. **Development Target:** When using dev server dev target is off. One can easily configure it by using --target and if he wants a legacy target once can use --target legacy and if one has already mentioned the browser list in package.json then one has to pass --target default as the cmd. It is basically transpiled of newer version of js code to old version so that browser can support our functionality
13. **Lazy mode:** If you are working on a large project so it might happen that build is time consuming and you don’t want to wait for the other features to build , so you can add --lasy so that now you don’t have to wait for other features build , they ll be build when the user navigates to that particular feature . One can also use dynamic import to access this feature.once the user navigate to a different features all the dependency are eagrly downloaded.
14. **Caching/ what is .parcel-cache:** Parcel cache every build it made to the disk . Every time a file/configuration is changed all the files which are depend on that will be rebuild when you start the dev server. Parcel keep the track of all the dev dependency ,plugins , configuration .and if any of then changes from the previous build it ll then rebuild those. By default it ll keep the cache files in the .parcel-cache folder . one can also change the folder location and also disable the reading from cache . disabling does not mean it ll not build that folder.
15. **HTTPS:** it can also provide the https protocol . by simply using the --https flag in the CLI . this is a self signed certi . can also use other certi for that
16. **AUTO INSTALL :** Parcel auto install a dependency . for eg if you user .scss file and the scss transformer is not installed in the project , it will auto install that and it will show a message in the terminal . this auto install only helps in the dev environment , it production the build will fail when the required package is not installed in the project . on can also disable this functionality. Using --no--autoinstall in the CLI.
17. **SHARED BUNDLES:** Commonly used modules in our app are not duplicated everywhere . Instead it split out in the shared bundle . so that it can be parallel downloaded with our app code . and it is caches separated from other bundles in the browser.

**Q: What is `npx` ?**

**A:** Node package execute . It helps you to execute any package without downloading it.

**Q: What is difference between `dependencies` vs `devDependencies`**

**A: dependencies –** Packages which are required for your app in production

**devDependencies –** Packages which are required for development and testing

**Q: What is `.gitignore`? What should we add and not add into it?**

**A: gitignore-** contains all the files/ folder which are not pushed into the repo , even when the code inside them changes .

One should add only those files which cannot be auto generated with the help of other files using any cmd or anything . For eg – package.json , your code files , package-lock.json

One should not add node\_modules , cache folder like .parcel-cache which are build when we build our app

**Q: What is the difference between `package.json` and `package-lock.json`**

**A:Package Json** keeps the track of all the installed packages in our application

**package-lock.json – keeps**  the track of all the installed packages in our application along with the exact versions of each packages

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

**A:** Because it contains the exact version of the packages which one has installed using any package manager .It might happen that the new version of the package might break our code .

**Q: What is `node\_modules` ? Is it a good idea to push that on git?**

**A: node\_modules** – they bacially contain the packages code which are installed by using a package manager . no one should not push into git as they can easily be generated with the help of package.json and pakckage-lock.json with the help of npm i cmd

**Q: What is the `dist` folder?**

**A : dist** folder contains the build files after one has build its app using any bundler like parcel/webpack

**Q: What is `browserlists` Read about dif bundlers: vite, webpack, parcel**

**A: browserlists –** its a package which make sure that our code will run into different browser . like if we have specify that it should run into last 2 versions of chrome , so it will make sure that no feature is broken down into these 2 versions

**Q: Read about: ^ - caret and ~ - tilda**

**A:** Package installed via manager has version , and the exact version is maintained in package.lock file.

Package-version: 1.2.3 Major.Minor.Patch

Patch – when there is a bug fix

Minor – new changes that wont break the previous code

Major – Which break the previous code

**Tilde** in the package file signifies that this package will automatically updated where there is change in the Patch

**caret** in the package file signifies that this package will automatically updated where there is change in the Patch or Major

**Q: Read about Script types in html (MDN Docs)**

**A:**  1.Ifone does not specify any type in script tag or a blank or JS MIME (multipurpose Internet Mail extension) text/javascript type **.**This means that the related file is a normal JS files.It is good to specify nothing instead of specify JS MIME type for a classic javascript file.

2. Module type-if one specify module that means the corresponding file is a javaScript module . Defer , async tags are not required as they don’t have any effect.

3. ImportMap –it states that the corresponding file contains a JS JSON file . It shows that how browser resolves importing

If one specify any other type that will be ignored by the browser.