1. What is NPM?

NPM is a software registry, that contains huge database of open source packages or javascript softwares, one can use and share.

NPM uses nodejs as its dependency and installs the needed package

NPM is used to create a package where project files will be stored along with the modules that our project is dependent on.

2. What is `Parcel/Webpack`? Why do we need it?

These are bundlers. Few to mention are Parcel, Webpack, VITE.

Purpose:

These bundlers will give us,

1. Production Ready Code.
2. Maintains Transitive Depenedancy
3. Performs Hot Module Replacement.
4. Caches.
5. Minifies the code
6. Performs Tree Shaking
7. Bundles several .js into 1/ several .CSS to one/ .html to one
8. Cleans code by removes console logs
9. Performa Image compression.
10. PolyFills – Makes code compatible with older browser versions using user customized browser list.
11. Enables https on local development Environment to test.
12. Maintains port number
13. Performs code splitting.

3. What is parcel-cache?

It is a separate folder that is created while parcel build our project.

Parcel caches everything it builds to disk. If you restart the dev server, Parcel will only rebuild files that have changed since the last time it ran. Parcel automatically tracks all of the files, configuration, plugins, and dev dependencies that are involved in your build, and granularly invalidates the cache when something changes. For example, if you change a configuration file, all of the source files that rely on that configuration will be rebuilt.

It will be in binary format.

4. What is npx?

NPX is the package executor or runner that is used to run a package that we installed locally or fetched from remote.

Similar to NPM run

5. What is difference between `dependencies` vs `devDependencies’?

It will be mentioned in package.json.

**Dependencies**: Packages required by our application in Production.

**devDependencies**: packages required that are need in local development environment.

6. What is TreeShaking?

**Treeshaking** is the process that is performed by bundlers like parcel or webpack to remove dead code based on import / export in Modular programming

It only imports the piece of code required in our application from the huge module that will be imported.

Eg. Lets say we need only add() function from calculator module, bundlers will only inject that one function from the module and removes all the other functions in it.

7. What is Hot Module Replacement?

Hot Module Replacement internally uses File Watcher Algorithm and watched each file changes with in our application and reload them in server.

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

1. Performs Hot Module Replacement – watches file changes and reload application in the server
2. Minifies the code – Removes console.log/ alerts/ warnings, performs Tree Shaking.
3. Bundles several .js into 1/ several .CSS to one/ .html to one – Merges all .js, .css, .html file into one .js, .css, .html file, also includes code from whatever Third Party modules are added.
4. Performs Image compression – Compress Images/ Media as huge media loading is expensive that effects the applications performance.
5. PolyFills – Makes code compatible with older browser versions using user customized browser list.

9. What is `.gitignore`? What should we add and not add into it?

**.gitignore** is a special file that git reads while commits and ignores to maintain versioning of those files and folders that are mentioned in it. To avoid those files to get inside the git repository.

**To add:** Any file that is not reacreated or regenerated. Package.json, package-lock.json

**Not to add:** Any file or folder that can be regenerated or recreated. Eg. Node\_modules, .parcel\_cache, dist folder

10. What is the difference between `package.json` and `package-lock.json`

**Package.json:** It is a json file that is generated during npm init and updated as and when we install new packages.

All the packages that we install are manintained in package.json, this will be used to rebuild our project in a different server or machines. It should always be added in the repo.

Package.json will have devdepedancy, dependency details along with the script details that we will used to start, test, build or application.

We can also use ^ & ~ in package.json so that the package gets updated automatically based on that rule.

**Package-lock.json**: It is separate file that contains the tree dependency of all the packages we install and specifies and locks the exact same version that is used in the project.

This helps in building applications will same package versions during building them on server or other machines.

Both package.json and package-lock.json will help to rebuild the node\_modules folder will all the actual dependant packages/ Modules and indirect ones as well.

11. Why should I not modify `package-lock.json`?

If we modify package-lock.json, while rebuilding the application in server or other machines, version mismatch on the packages will affect the running of the application.

12. What is `node\_modules` ? Is it a good idea to push that on git?

Node\_modules is a folder in which all project relates modules and dependant tree module/packages are installed.

Its not fair to push that to git, since it can be regenerated using npm install and its also use very huge. That’s why we make use of package.json and package-lock.json files.

This folder should be added in .gitignore.

13. What is the `dist` folder?

Dist folder is a separate folder that is created when parcel build an application.

Command: npx parcel build <entry\_point>

If the folder is deleted, during next build it will be recereated.

Keeps files minified and created one .html, .css, .js file

14. - What is `browserlists`?

Browserlist is a package that parcel or bundler uses so that our code will even work on older browsers.

It actually polyfills.

Step1: in package.json file add a property

Ex:

"browserslist": [

">0.2%",

"not dead",

"not ie <= 11",

"not op\_mini all"

]

15. Read about: ^ - caret and ~ - tilda

Caret or Tilda is used to mention approximate equivalent version(~) and compatible version (^) of packages to be installed. So that the bundler will automatically update the package.

Caret is the default notation as used by NPM

Eg :

1. **^8.4.1** – This will update to 8.**X.X –** Minor version and patch will be updated automatically.

2. **~8.4.1** – this will update to 8.4.**X –** patches will be updated automatically

16. read About script types in html

And: 1. **No attributes** – consider as default javascript file.

2. **module** – It treats the code as Javascript module