**ASSIGNMENT-2 IGNITION OUR APP**

**1.What is NPM?**

NPM is not stand for Node Package Manager. But It’s a Package Manager. Basically, it’s a repository for all Packages. Its one of the biggest package manager. Any Package we need to include our project we use NPM. All the packages, all the library, all the utilities over there. These packages are also known as dependencies.

**2.What is ‘parcel/webpack’? Why do we need it?**

**Parcel:**

Parcel is a popular open-source web application bundler. A bundler is a tool that takes various sources files from your web application such as JS, CSS, HTML, and images and bundles them together into optimized files for deployment.

Parcel also integrates with React Fast Refresh & Use Hot Reloading API to automatically preserve your application state between updates. This gives you instant feedback as you make changes, without taking you out of context.

This includes tree shaking& minifying your JS, CSS & HTML resizing & Optimizing images, content hashing, automatic code splitting and much more.

Parcel is a zero-config setup. It requires minimal configuration and automatically handles many tasks, such as transforming & bundling your code, handling various file types, and optimizing assets for Production. The primary purpose of a bundler is to improve the performance & efficiency of web applications by reducing the number of HTTP requests & optimizing the size of the file sent to the browser.

**Webpack:**

Webpack is a static module bundler for JS applications, it takes all the code from your application and makes it usable in a web browser. Modules are reusable chunks of code built from your apps, JS, node\_modules, images & the CSS styles which are packaged to be easily used in our website.

Website applies automatic transformations on your working files to make them into production files that are most useful for your end users.

**3.What is ‘parcel-cache’?**

parcel-cache is a kind of binary file and caching everything you do.

The parcel-cache directory is a folder created by the parcel bundler during the build process. It’s used to store cached data and artifacts to improve the speed and efficiency of subsequent builds. This cache directory helps parcel avoid reprocessing & rebuilding files that haven't changed since the last build, which can significantly speed up development iterations.

**4.What is ‘NPX’?**

NPX is a Node Package execute. It is simply an NPM package runner. It allows developers to execute only JS package available on the NPM registry without even installing it.

NPX is installed automatically with NPM version 5.2.0 and above.

**Advantage of NPX:**

NPX allows you to run and use packages as without needing to locally or globally install them.

**5.What is the difference between ‘dependencies’ vs ‘devDependencies’?**

**dependencies:**

These packages are required for your application to work correctly and typically used for functionality that is part of the final product. Dependencies are specified in the “dependencies” section of the package. json.

**devDependencies:**

Development dependencies also known as “devDependencies”. devDependencies are the packages that are required for development & testing purpose only. These packages are not required for the application to run properly.

**peerDependencies:**

peerDependencies are the packages that your package expects to be installed in the user’s environment. These packages are not installed automatically when your package is installed, but the user is expected to install them manually.

**6.What is Tree shaking?**

Tree shaking is a process that bundlers like webpack, parcel. Tree shaking also known as dead code Elimination, it’s the practice of removing unused code in your production build.

**7.What is Hot Module Replacement?**

HMR is known as Hot Module Replacement. It’s a development feature that allows you to update parts of your application in real time without requiring a full page reload. Its simply known as if we make any change or need to update our code, it will get automatically refresh.

**8.List down the superPowers of Parcel?**

There are lot of superPowers contain parcel Few of them are discussed as follows:

* **HMR:**

HMR is known as Hot Module Replacement. It’s a development feature that allows you to update parts of your application in real time without requiring a full page reload. Its simply known as if we make any change or need to update our code, it will get automatically refresh.

* **File Watching Algorithm:**

File watching means it reads the file for automatically refresh by HMR. Its written in c++.

* **Differential Bundling:**

It support older browsers. That means our app can be open chrome, edge, firefox, internet explorer etc.

* **Diagnostic:**

It have a beautiful error display view. If we make an error in our code or configuration, parcel displays beautiful diagnostics in our terminal & in browser.

* **Tree shaking:**

Tree shaking is a process that bundlers like webpack, parcel. Tree shaking also known as dead code Elimination, it’s the practice of removing unused code in your production build.

* **Faster Builds-caching things:**

It shows console built in ms fastly.

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

gitignore file tells git which files to ignore when committing your project to the Github repository. Gitignore is located in the root directory of your repo. The .gitignore file itself is a plain text document.

You can create a .gitignore file in your repository’s root directory to tell git which files & directories to ignore when you make a commit.

Files containing passwords, API keys, private credentials, and other sensitive data that should not be exposed in the repo. Whatever files we are able to regenerate , it not need to add into github.

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

|  |  |
| --- | --- |
| **package.json** | **package.lock.json** |
| It is a metadata file that describes the project’s dependencies, scripts, configuration and other details. | It’s a lock file that provides an exact deterministics |
| It is typically created and modified manually by the developer to manage the project’s dependencies and configuration. | It is automatically generated by npm and updated whenever you install or update packages. |
| It lists the required dependencies and their version ranges, but not the exact versions to be installed. | It is used to ensure that the same dependencies are installed consistently across different environments & prevent conflicts due to different versions being installed. |
| It can be easily shared and committed to version control systems. | It is not meant to be manually modified and should be committed to the version control system to ensure consistency across all team members. |

**.**

**11.Why should I not modify “package-lock.json”?**

If you’re working on a team/sharing your project with others, modifying package-lock.json could lead to difficulties in recreating the exact environment needed to run the project.

Manually editing package-lock.json can result in conflicts when merging changes from collaborators, especially in team projects using version control system like git.

It working on local but not on production. It basically avoid the problem.

**12.What is node-modules? Is it a good idea to push that an git?**

node-modules directory is one of the crucial parts of any node or real project, but it shouldn’t be tracked by the version control system(git) due to its large size. The right approach is to track the package.json file and use the npm tool to regenerate node-modules.

Its not a good idea to push that an git. You should not commit the node -modules folder in git. Here the reasons: The node-modules folder has a massive size(up-to giga bytes) It is easy to recreate the node-modules folder via package.json

**13.What is the ‘dist’ folder?**

When you execute parcel in index.html, it generates development build of your project & hoisted on to 1234. It generates a development build it put into dist folder. So output coming from dist folder.

If we create a prod-build, In dist folder contains our whole code into compressed code. It contains minified & bundled JS files, minified & concatenated CSS files, optimized image files, HTML files that reference the minified assets, Any other asset required for the application to run.

**14.­BrowsersList:**

We have to tell browsersList that what are browser support our app. BrowsersList is a npm package and it needs some configuration.