Chapter 02 - Assignment - Igniting our App

Ques 1 :-　What is `NPM`?

Solu 1 :-　NPM is the package manager for the Node JavaScript platform. It puts modules in place so that node can find them, and manages dependency conflicts intelligently. It is extremely configurable to support a wide variety of use cases. Most commonly, it is used to publish, discover, install, and develop node programs

Ques 2 :-　What is `Parcel/Webpack`? Why do we need it?

Solu 2 :-　Parcel/Webpack is a bundler that performs multiple actions on our code to make that code ready for deployment into the live server. Bundler helps to bundle up the entire javascript code into a single file. We need these because :-

- With the help of bundlers we easily minifiy and optimize our code.
- it is also used for the development environment to make our work faster through many algorithms they have like - HMR(Hot module replacement), Caching while development, Faster Algorithm, etc.
- It compresses and optimize media assets(images, video etc) for better performance.

Ques 3 :-　What is `.parcel-cache?

Solu 3 :-　It is basically a folder which is autogenerated whenever when start or restart our development server using : *npx parcel <entry point of code>* .

Pracel caches all the files either user generated or autogenerated. Parcel automatically tracks all of the files, configuration, plugins, and dev dependencies that are involved in your build. So, if we restart our server then parcel will rebuid only those files which have some changes since the last time run. So, parcel do all of these tracking using '.parcel-cache'.

Ques 4 :-　What is `npx` ?

Solu 4 :-　Npx is a tool that use to execute packages.

A package can be executable without installing the package. It is an npm package runner so if any packages aren't already installed it will install them automatically**.**

Ques 5 :-　What is difference between `dependencies` vs `devDependencies

Solu 5 :-	dependencies :- The libraries under dependencies are those that your project really needs to be able to work in production.

devDependencies :- The libraries under devDependencies are those that you need during development

**Ques 6 :-	What is Tree Shaking?**

Solu 6 :-	Tree shaking is a term commonly used within a JavaScript context to describe the removal of dead code.

In modern JavaScript applications, we use module bundlers (e.g., webpack or Rollup) to automatically remove dead code when bundling multiple JavaScript files into single files. This is important for preparing code that is production ready, for example with clean structures and minimal file size.

**Ques 7 :-	What is Hot Module Replacement?**

Solu 7 :-	If we change something in our code,	Parcel automatically rebuilds the changed files and updates your app in the browser. By default, Parcel fully reloads the page, but in some cases it may perform Hot Module Replacement (HMR)HMR improves the development experience by updating modules in the browser at runtime without needing a whole page refresh. This means that application state can be retained as you change small things in your code.

CSS changes are automatically applied via HMR with no page reload necessary. This is also true when using a framework with HMR support built in, like React (via Fast Refresh), and Vue.

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

Solu 8 :-	Dev Server, Hot reloading, Caching, File watcher, Minification, tree shaking, Zero Configration.

Dev Server :- Parcel’s builtin dev server is automatically started when you run the default parcel command, which is a shortcut for parcel serve. By default, it starts a server at http://localhost:1234.

Hot reloading :- When we change something on our code then parcel automatically rebuilds the changes files and reloads automatically.

Caching :- 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.

File watcher :- Using this watcher Parcel watches every file in your project root (including all node_modules). Based on events and metadata from these files, Parcel determines which files need to be rebuilt.

Minification :- Parcel includes minifiers for JavaScript, CSS, HTML, and SVG out of the box. Minification reduces the file size of your output bundles by removing whitespace, renaming variables to shorter names, and many other optimizations.

Tree shaking :- In production builds, Parcel statically analyzes the imports and exports of each module, and removes everything that isn't used. This is called "tree shaking" or "dead code elimination.

Image optimization :- Parcel supports resizing, converting, and optimizing images

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

Solu 9 :- .gitignore is a file that we create in our working directory. Which is used to ignore files/folders that we don't want to track in our git repo.

The files or folders we should put in .gitignore that can be autogenerated by your system or packages like node_moduleor that can be generated by the server.

The files or folders we should not add in .gitignore that are required in your production build and development builds like package.json and package-lock.json files.

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

Solu 10 :- package.json : It is a list of all the dependencies of a project (among other things) that need to be downloaded for the project. This includes version info for what needs to be downloaded.

package-lock.json : It automatically generated when you installed any dependencies in your project. It ensures that your project uses the exact version of its dependencies. It also having the record of the node_module folder. It's a good idea to commit both package.json and package-lock.json to version control so that other developers working on the project can easily reproduce your dependencies.

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

Solu 11 :- It is not a good idea to modify the package-lock.json file because it gets created when you have already installed the dependencies. So then It will conflict with the package.json dependencies that you have already installed. It will hamper your project.

Ques 12 :- What is `node_modules` ? Is it a good idea to push that on git?

Solu 12 :- It just a directory created by npm and a way of tracking each packages you install locally via package.json. The node_modules folder is used to save all downloaded packages from npm in your computer for the JavaScript project that you have.

No, the autogenerated files should not be push into git. It should always be in gitignore file.

Ques 12 :- What is the `dist` folder?

Solu 12 :- The /dist folder contains the minimized version of the source code. The code present in the /dist folder is actually the code which is used on production web applications. Along with the minified code, the /dist folder also comprises of all the compiled modules that may or may not be used with other systems.

Ques 12 :- What is browserlists ?

Solu 12 :- It helps to make our code compatible with browsers (including olders verison of browsers)