Class 2: Assignment

▼ What is NPM?

- **NPM** is a package manager for the Javascript programming language and it is used to install packages(libraries, modules).
- npm helps in handling dependency and package management for the project.
- The command used to initialize npm in a project: `npm init`
- This creates a package.json file with all the scripts, dependencies and versions used in the project.

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

Parcel and webpack are bundlers.

A **bundler** is a development tool that combines many javascript code files into a single one that is production-ready loadable in the browser.

Bundlers also help with minifying the files and optimizing the code to improve performance.

▼ What is `.parcel-cache`?

- Parcel caches everything when it builds. So, if a dev server is restarted, Parcel will only rebuild files that have changed since the last time it ran.
- Parcel automatically tracks all of the files, configurations, plugins and dev dependencies that are involved in your build.
- The cache is stored in `.parcel-cache` folder by default and it should be added to the `.gitignore` file.

▼ What is `npx`?

npx is a command-line tool that is included with npm which is used to execute packages.

`npx parcel index.html`

Command to execute parcel with index.html as the entry point.

▼ What is the difference between `dependencies` vs `devDependencies`

- The package.json file keeps a track of dependencies and dev dependencies.
- **devDependencies** are the modules which are required only during development.
- `npm install -D parcel`
- `npm install —save-dev parcel`
- **dependencies** are modules which are also required at runtime.
- `npm install react`

▼ What is 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.
- Tree shaking means removing unwanted code.
- Example: If we are importing a library in our app and the library has too many helper functions. But I'm just using one function from the helper code. So, Parcel just ignores all the unused functions.

▼ What is Hot Module Replacement?

HMR updates modules in the browser at runtime without needing a whole page refresh. This means that the application state can be retained as you change small things in the code.

The **file watcher algorithm** written in C++ keeps a track of all the files in real-time and refreshes as soon as the code is saved.

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

- **Dev Server**: Parcel's built-in dev server is automatically started when you run the default parcel command. It starts a server at port 1234. If the port is already in use, it uses a fallback port.
- **Hot reloading**: As you make changes to the code, Parcel automatically rebuilds the changed files and updates the app in the browser.
- **HTTPS**: When we need to use HTTPS during development we can run `npx parcel index.html -https`
- **Minification** in prod: Parcel includes minifiers for JavaScript, HTML, and CSS files. It reduces the file size by removing white spaces, renaming variables to shorter names and other optimizations.

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

A `gitignore` file includes files and directories to ignore when a commit is made.

Anything which can be regenerated in the server can be added to `gitignore`.

Ex. node_modules, dist files, .parcel-cache files

Files like package.json and package-lock.json which track all the versions of the app should not be added to `gitignore`.

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

package.json is generated by `npm init`. It has the version of all the packages but if any automatic upgrades are made, it is not shown here.

package-lock.json is automatically generated when there is a change in either the `node_modules` or `package.json` file. It keeps track of the exact version numbers of each installed package.

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

The purpose of the package-lock.json file is to track the entire tree of dependencies and the exact version of each dependency.

It should be in the code repository so everyone can have the same configuration of the project. Editing this file might break the code since a different version might be in production.

▼ What is `node_modules`? Is it a good idea to push that on git?

npm is a package manager and it stores all the dependencies locally in the `node_modules` folder.

It can be reinstalled at any time using the package-lock.json file and `npm install` so this should be added in the gitignore file and not be pushed on git.

▼ What is the `dist` folder?

`dist` has the dynamically generated files that Parcel created for you. (Production files)This folder includes your HTML, CSS, and JavaScript files, but are all named `index` with random characters in between. It also includes `.map` files.

Parcel generates what is called a *source map* for you when you use the `npx parcel` command. The source map tells the browser how to locate the original source code from your bundled code. It is used to assist with debugging your code in development and in production

▼ What is `browserlists`?

For browser targets, the `browserslist` field in package.json can be used to specify which browsers you support.

```
{
  "browserslist": "last 2 versions"
}
```

▼ Read about: ^ - caret and ~ - tilda

2.1.`4`

This number refers to a patch release, which means it is a bug fix and is backward compatible.

2.`1`.4

This number refers to a minor release, which means new features have been added but it is still backward compatible.

`2`.1.4

This number refers to a major release, which means that it introduces major changes and may break backward compatibility.

Using a Tilde (~)

Using a tilde sign before our version number means that we can accept only a patch release when updating our package.

Using a Caret (^)

Using a caret (^) sign means that we can accept minor releases and patch releases, but not a major release when updating our package.

▼ Read about different bundlers: vite, webpack, parcel

▼ Read about script types in HTML (MDN Docs)