THEORY

- What is `NPM`?

NPM doesn't have a specific full form, but normally we call it Node Package Manager. It is a command-line tool used for installing, managing, and sharing JavaScript packages or modules. It allows developers to easily include external libraries and dependencies in their projects.

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

Parcel and Webpack are both popular bundlers in the JavaScript ecosystem. They are used to bundle and optimise different assets such as JavaScript, CSS, and images for web applications. These tools help manage dependencies, optimise code, enable features like hot module replacement, and generate a build output that can be deployed to production.

- What is `.parcel-cache`?

The `.parcel-cache` directory is created by Parcel when building a project. It stores cached information about the project's dependencies, allowing Parcel to avoid re-parsing and re-analyzing everything from scratch on subsequent builds. It helps speed up the build process by reusing previously processed data.

- What is `npx`?

`npx` is a command-line tool that comes with NPM. It is commonly used to run scripts that are included in a package.
Ex - npx parcel build index.html

- What is the difference between `dependencies` vs `devDependencies`?

In the context of package.json, "dependencies" refers to the packages that are required for the application to run in both development and production environments

Dependencies that help in debugging and can show errors during development are known as devDependencies.

Dependencies that are helpful for both development and production are categorised as devDependencies.

- What is Hot Module Replacement?

It helps in automatically reloading our web page when we change something in our code, saving a lot of time.

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

- BrowserLists: Helps ensure compatibility of our website in different browsers.
- HMR: Allows for instant code updates during development, saving time by avoiding full page reloads.
- Minify: Removes whitespaces, renames variable names, etc., reducing the size of code files.

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

It is a file that consists of names of files or folders that we don't want to push into Git.

We should add `.parcel-cache`, `node_modules`, `.env`, etc., folders, and other files into it.

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

They are necessary when we have to generate the `node_modules` folder using `npm init`.

`package.json` doesn't provide detailed information about dependencies.

`package-lock.json` locks the version according to our needs using the (~ and ^ concept).

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

`package.json` is a configuration file used in Node.js projects. It contains metadata about the project, as well as information about its dependencies, scripts, and other project-specific configurations. On the other hand, `package-lock.json` is automatically generated by NPM and provides a detailed description of the exact versions of packages installed in the `node_modules` directory. It helps ensure consistent installations across different environments and guarantees the same dependencies are used when the project is built or deployed.

- What is `node_modules`? Is it a good idea to push that on Git?

`node_modules` is a directory where the dependencies for a Node.js project are installed. It contains all the packages and their dependencies required for the project to run. It is not necessary to push the `node_modules` directory to a Git repository because it can be generated again by running `npm install` based on the `package.json` and `package-lock.json` files. The `node_modules` directory can be quite large, and pushing it to Git can increase the repository size unnecessarily.

- What is the `dist` folder?

The `dist` folder is a directory in web development projects that holds the final optimised and bundled version of the project, ready for deployment to a production environment. It contains the minimised and concatenated

JavaScript files, compressed CSS stylesheets, optimised images, and other static assets necessary for the web application to run efficiently.

- What is Tree Shaking?
Tree shaking is a term commonly used within a JavaScript context to describe the removal of dead code.

- What is `browser lists`?

- "^" -> will update patch and minor version
- "~" -> will only update patch