1. What is ‘NPM’?
   * It is a tool for package management and the default package manager for Node projects. NPM is installed when NodeJS is installed on a machine. It comes with a CLI used to interact with online database of NPM. This database is called NPM Registry, and it hosts public and private packages. To add or update packages, we use the NPM CLI to interact with this database.
   * To initialize npm => npm init
2. What is ‘Parcel/Webpack’? Why do we need it?
   * Parcel/Webpack is a type of web application bundler used for development and production purposes or power-up our applications with different type of functionalities and features. It requires zero configuration and it can take any type of file as an entry point, but an HTML or JS is a good place to start.
   * Features of Parcel:
     + Dev and Production Build
     + HMR (Hot Module Replacement) – Parcel keeps track of file changes via file watcher algorithm and renders the changes in files
     + File watcher algorithm – made with C++
     + Minification
     + Cleaning our code
     + Super fast build because of Caching
     + Image Optimization
     + Compressing
     + Consistent Hashing
     + Differential Bundling – can be open on any older version of browser
     + Diagnostic – Better Error Suggestion
     + HTTPs – gives you a way to host it on HTTPs server also
     + Creates local server – provide port number
     + Zero configuration
     + Automatic code splitting

* Installation commands:
  + npm install -D parcel
    - -D is used for development and as a development dependency.
      * Parcel Commands:
        + For development build:

npx parcel <entry\_point>

* For production build:
  + npx parcel build <entry\_point>

1. What is .parcel-cache?
   * It is used by parcel(bundler) to reduce the building time. It stores information about your project when parcel builds it so that when it rebuilds, it doesn’t have to re-parse and re-analyze everything from scratch. It’s a key reason why parcel can be so fast in development mode.
2. What is ‘npx’?
   * npx is a tool that is used to execute the packages. It comes with the npm, when you install npm above 5.2.0 version then automatically npx will be installed. It is an npm package runner that can execute any package that you want from npm registry without even installing that package.
3. What is difference between ‘dependencies’ vs ‘devDependencies’?
   * **Dependencies** are the packages required for your application to function in a production environment or testing environment. This includes packages that your code imports and uses to function effectively such as React, Angular, Vue, Express, JQuery etc.
   * **DevDependencies** should contain modules/packages a developer needs during development process unnecessary for the application to run. This includes tools like parcel, webpack, vite, mocha, Jest. These packages are used during development phase and before deployment but are not required by the end-users of your application.
   * To save a dependency as a devDependency on installation we need to do,
     + npm install –save-dev
     + Instead of just, npm install –save
4. What is Tree Shaking?
   * **Tree Shaking** is a process of removing the unwanted code that we do not use while developing the application. In computing, tree shaking is a dead code elimination technique that is applied when optimizing code.
   * Suppose you have 20 function in your file and you’re using just 5 of them. So, it’ll remove unused code from file.
5. What is Hot Module Replacement?
   * HMR exchanges, adds, or removes modules while an application is running without a full reload. This can significantly speed-up development in a few ways:
     + Retain application state which is lost during full reload
     + Save valuable development time by only updating what’s changed
     + Instantly update the browser when modifications are made to CSS/JS in the source code.
6. List down your favourite 5 superpowers of Parcel and describe any 3 of them in your own words.
   * 5 superpowers of Parcel are:
     + HMR – adds, or removes modules while an application is running without a full reload.
     + File Watcher Algorithm – File Watcher is monitoring directories on the file system and perform specific actions when desired files appear.
     + Caching while development – It’ll give you a faster build because of Caching
     + Minification – It is the process of minimizing code and markup in your web pages and script files.
     + Image Optimization
7. What is ‘.gitignore’? What should we add and not add into it?
   * The .gitignore file is a text file that tells Git which files or folders to ignore in a project during commit to the repository. The types of files you should consider adding to a .gitignore file are any files that do not get committed.
   * For example; for security, the security key files and API keys should get added to the gitignore.

package-lock.json should not get added into your .gitignore file.

1. What is the difference between ‘package.json’ and ‘package-lock.json’?
   * package.json:
     + This file gets generated after initialising npm => npm init.
     + This is a configuration for npm and this file is mandatory for every project.
     + It keeps a track of version of all the dependencies, all the packages that is installed into our system.
     + It keeps a approx version of packages.
   * Package-lock.json
     + This file is automatically generated for hose operations where npm modifies either the node\_modules tree or package.json.
     + It is generated after an npm install.
     + It locks the exact version of package/dependency that we installed in our project and keeps a track of it.
     + It allows future devs & automated systems to download the same dependencies as the project.
     + It also allows to go back to the past version of the dependencies without actual ‘committing the node\_modules folder’.
2. Why should I not modify ‘package-lock.json’?
   * This file contains the information about the dependencies and their versions used in the project. Deleting it would cause dependencies issue in the production environment. So don’t modify it. It’s being handled automatically by NPM.
3. What is node\_modules? Is it a good idea to push that on git?
   * node\_modules folder is like a cache for the external modules that your project depends upon. It contains actual data of dependencies/packages that our project need.
   * When you npm install them, they are downloaded from the web and copied to node\_modules folder and Nodejs is trained to look for them there when you import them (without a specific path).
   * Don’t push node\_modules on github because it contains lot of files (more than 100 MB), & it will cost you memory space. Anyway this node\_modules can be re-generated again and again after doing npm install.
4. What is the ‘dist’ folder?
   * 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.
5. What is ‘browserlists’?
   * It is a tool that allows specifying which browsers should be supported in your frontend app by specifying “queries” in a config file. It’s used by frameworks/libraries such as React, Angular and Vue, but it’s not limited to them.