Git Vs Github

Git and GitHub are related but serve different purposes in the realm of version control and software development.

Git is a distributed version control system that allows developers to track changes in their code, collaborate with others, and manage different versions of their projects.

Git is used via command-line interface (CLI) or through graphical user interfaces (GUIs) like GitKraken, SourceTree, or integrated development environments (IDEs) like Visual Studio Code

GitHub is a web-based platform that provides hosting for Git repositories. It offers a user-friendly interface for managing Git repositories and adds additional features for collaboration and project management

GitHub is accessed through a web interface, and users can also interact with it using Git commands in the terminal or through GitHub's desktop application

| Feature | Git | GitHub |
| --- | --- | --- |
| Type | Version control system | Web-based platform for hosting Git repos |
| Purpose | Track changes in code | Collaborate and share code |
| Hosting | Local (on your machine) | Cloud-based (on GitHub servers) |
| Collaboration | Basic (via Git commands) | Advanced (pull requests, issues, etc.) |
| User Interface | Command-line or GUI | Web interface and desktop app |
| Community Features | None | Social features, project visibility |

What is NPM

Npm is not a node package manager (lots of people says npm stands for node package manager but its not correct) Npm has no fullform and its just npm. Npm manages packages

package,json is configuraion for NPM

we can inject npm with **npm init** command that will create package.json file for us and we will use parcel library and will install via **npm install -d parcel** command

Package.json vs package-lock.json

package,json is a file that contains metadata about your project, including its name, version, description, author, license, and most importantly, its dependencies.

Package-lock.json is an automatically generated file that locks the versions of the dependencies installed in your project. It ensures that the same versions of dependencies are installed across different environments. It Contains the exact version of each installed package, including nested dependencies. This ensures that everyone working on the project has the same setup

* package.json is essential for defining your project and its dependencies, while package-lock.json ensures that the exact versions of those dependencies are installed consistently across different environments.
* When you run npm install, npm will read package.json to determine which packages to install and will generate or update package-lock.json to reflect the exact versions that were installed.
* It is generally recommended to commit both files to version control (e.g., Git) to ensure that your project can be reliably built and run by others.

Package version with Caret (^) Symbol and with tilde Symbol (~)

In a package.json file, the caret (^) and tilde (~) symbols are used to specify version ranges for dependencies. They help manage how updates to those dependencies are handled.

* Use ^ when you want to allow for minor and patch updates, which are generally backward-compatible.
* Use ~ when you want to restrict updates to only patch-level changes, ensuring that the minor version remains the same.
* Understanding these symbols helps you manage your project's dependencies more effectively, ensuring that you can benefit from updates while maintaining stability

node\_modules is a collection of dependencies

The .gitignore file is a crucial part of using Git for version control. It specifies which files and directories should be ignored by Git when tracking changes in a repository. This is particularly useful for excluding files that are not necessary for the project, such as temporary files, build artifacts, and sensitive information.