## **Git**

## **What is Git?**

* Git is a popular version control system.
* It was created by Linus Torvalds in 2005 and has been maintained by Junio Hamano since then.

It is used for:

* Tracking code changes
* Tracking who made changes
* Coding collaboration

### **What does Git do?**

* Manage projects with **Repositories**
* **Clone** a project to work on a local copy
* Control and track changes with **Staging** and **Committing**
* **Branch** and **Merge** to allow for work on different parts and versions of a project
* **Pull** the latest version of the project to a local copy
* **Push** local updates to the main project

### **Working with Git**

* Initialize Git on a folder, making it a **Repository**
* Git now creates a hidden folder to keep track of changes in that folder
* When a file is changed, added or deleted, it is considered **modified**
* You select the modified files you want to **Stage**
* The **Staged** files are **Committed**, which prompts Git to store a **permanent** snapshot of the files
* Git allows you to see the full history of every commit.
* You can revert back to any previous commit.
* Git does not store a separate copy of every file in every commit but keeps track of changes made in each commit!

### **What is GitHub?**

* Git is not the same as GitHub.
* GitHub makes tools that use Git.
* GitHub is the largest host of source code in the world and has been owned by Microsoft since 2018.
* In this tutorial, we will focus on using Git with GitHub.

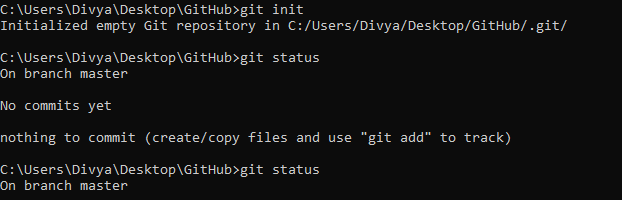
### **Why Git?**

* Over 70% of developers use Git!
* Developers can work together from anywhere in the world.
* Developers can see the full history of the project.
* Developers can revert to earlier versions of a project.

**GitHub**

**1. Git Workflow Summary for the Given Commands:**

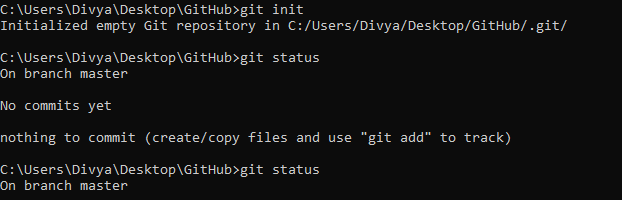
* **git init****:** Initializes a new Git repository in the current directory.
* The .git/ folder is created, which tracks all Git operations for the project.



**2.Checking Status:**

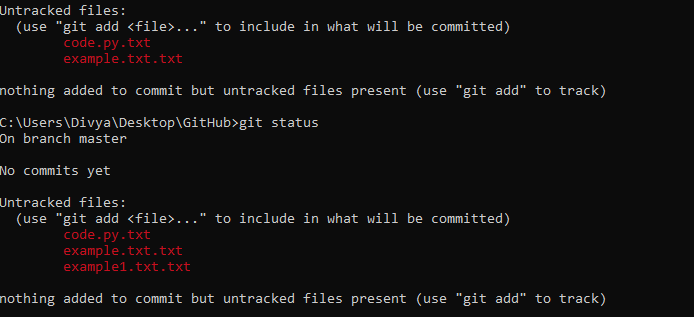
* **git status:** Displays the state of the working directory. Initially, after running git init, no commits are made, and files are untracked.
* Files listed under “untracked files” are those yet added to version control

(e.g., code.py.txt, example.txt.txt, example1.txt.txt).

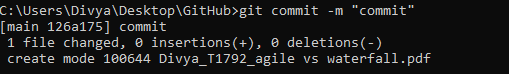


**3.Adding Files:**

* git add <file>: Adds a file to the staging area, marking it for inclusion in the next commit.
* Example: git add example1.txt.txt and git add. (adds all files in the directory).



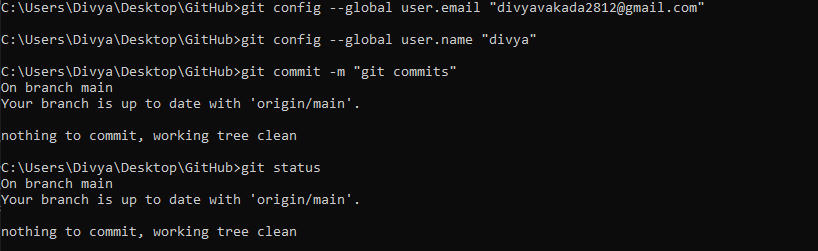
**4.Committing Changes:**

* git commit –m “<commit>”: Saves the staged changes to the repository with a descriptive message (e.g., “git commits”).
* First-time commit requires setting user details using git config --global user.email and git config --global user.name for identification.

**5.Adding Remote Repository:**

* git remote add origin <url>: Adds a remote repository (e.g., GitHub) to push changes to.
* Example: git remote add origin

**https://github.com/Divya-2812/Divya-2812.git**

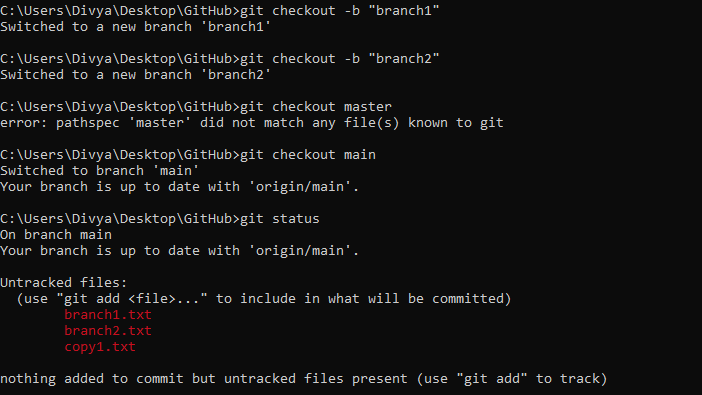


**6.Pushing Changes to Remote:**

* git push –u origin master: pushes the local changes to the remote repository on the master branch.
* Git push –set-upstream origin <branch>: Pushes changes to a newly created branch and sets up tracking for the remote branch.

**7.branch Operations:**

* **Git checkout –b <branch\_name>:** Creates a new branch and switches to it.
* **Git checkout <branch\_name>:** Switches to an existing branch.
* **Example:** git checkout –b “copy1” to create and switch to a new branch called copy1.

**8.Merging changes from Remote Branch:**

* git pull: Fetches changes from the remote repository and merges them with the local branch.
* Git merge <branch>: Merge the specified branch into the current branch. If there are no changes, it will show “already up to date”.

**9.Handling Untracked Files:**

* Git lists untracked files that haven’t been added to version control.
* Use git add <file> to track them, then commit.

**10.Push Operations for Branches:**

* After committing changes in a new branch (e.g., branch1), you must push it to the remote with git push --set-upstream origin branch1 to create and track the remote branch.

**11.Final Pushes:**

* When pushing a new branch, Git prompts to create a pull request on GitHub.