**Install GIT for windows**

**1. Configure Git (One-time setup)**

git config --global user.name "Your Name"

git config --global user.email [your\_email@example.com](mailto:your_email@example.com)

**Install gh cli (gh is GITHUB CLI) if not installed**

**will login to GIT User account**

**2. Login to GitHub (via CLI)**

gh auth login

Alternatively, use SSH authentication:

ssh-keygen -t rsa -b 4096 -C [your\_email@example.com](mailto:your_email@example.com)

This will ask to tick the scopes (Permission) and generate the SSK Key.

Then, add the SSH key to your GitHub account.

**3. Using Authentication Token for using HTTPS**

**Why Use an Authentication Token?**

* **More Secure**: Unlike passwords, tokens can be **revoked** if compromised.
* **Required for GitHub (if using HTTPS)**: GitHub disabled password authentication for Git over HTTPS. You must use a **personal access token (PAT)** instead.
* **Can Have Expiry & Scopes**: You can set an expiration date and define what the token can access.

**How to Generate an Authentication Token in GitHub**

1. **Go to GitHub**:
   * Open [GitHub Personal Access Tokens](https://github.com/settings/tokens).
2. **Click on** Generate new token (classic).
3. **Select Permissions** (Scopes):
   * For Git operations, select repo.
4. **Set Expiration** (Optional):
   * You can choose an expiration date or make it permanent.
5. **Generate & Copy the Token**:
   * **Important**: Copy it **immediately** because GitHub won’t show it again.

**4. Creating a new repository**

gh repo create my-repo --public --source=. --push

**5. Initialize a Local Repository**

git init

Creates a new Git repository in the current folder.

**5. Add a Remote Repository**

git remote add origin https://github.com/yourusername/my-repo.git

**6. Create and Switch to a New Branch (Optional)**

git checkout -b feature-branch

If working on the main branch, skip this.

**7. Add Files to Staging Area**

git add .

This stages all changes.

**8. Commit the Changes**

git commit -m "Initial commit"

**9. Push to GitHub**

If pushing to main:

git branch -M main # Ensures branch name is main

git push -u origin main

If pushing a feature branch:

git push -u origin feature-branch

**10. Merge Branch into Main (if working with branches)**

git checkout main

git merge feature-branch

git push origin main

Now, your code is successfully pushed to the main branch. Let me know if you need additional steps like handling pull requests or SSH key setup!

**1️⃣ Check Files in the Latest Commit**

git show --name-only

🔹 This shows the **list of files** modified in the latest commit.

If you want to see the changes **along with file names**, use:

git show –stat

**🛠️ Clear Git in Different Ways**

**1️⃣ Remove Git from the Folder (Start Fresh)**

If you want to **completely remove Git tracking** from your project and start over:

rm -rf .git # For Mac/Linux

rmdir /s /q .git # For Windows PowerShell

🔹 This **deletes** the Git repository from your folder.

👉 **After running this, you'll need to reinitialize Git:**

git init

**2️⃣ Unstage All Changes (Undo git add .)**

If you added files to staging and want to unstage them:

git reset

🔹 This **unstages all files**, but keeps them in the working directory.

**3️⃣ Undo the Last Commit**

**1. Undo the Commit but Keep Changes in the Working Directory**

If you want to undo the last commit but keep the changes (files and modifications) staged or in your working directory, use:

git reset --soft HEAD~1

🔹 HEAD~1: Refers to the previous commit (the one before the last).

🔹 --soft: Keeps the changes from the undone commit in the staging area.

**2. Undo the Commit and Discard Changes**

If you want to undo the last commit **and discard all the changes made in that commit**, use:

git reset --hard HEAD~1

🔹 --hard: Completely removes the changes made in the last commit. Be cautious, as this action is irreversible unless you have backups.

**3. Undo the Commit Without Changing the Working Directory**

If you want to undo the last commit but keep the changes in your working directory (unstaged), use:

git reset HEAD~1

This leaves the changes intact but unstaged (you’ll need to manually stage them if needed).

⚠️ **This will delete your changes permanently!**

**4️⃣ Remove Remote Repository (Unlink from GitHub)**

If you want to **remove the connection to GitHub**, use:

git remote remove origin

🔹 This removes the remote repository but keeps the local commits.

**5️⃣ Delete All Local Changes (Reset to the Last Commit)**

If you want to discard all uncommitted changes and reset your repo:

git restore .

or

git reset --hard

⚠️ **This will delete all local changes that are not committed!**

**6️⃣ Delete All Local Commits (Start Over)**

If you want to **delete all commits** and start fresh but keep the files:

git checkout --orphan new\_branch

git add .

git commit -m "Fresh start"

git branch -D main

git branch -m main

git push -f origin main

🔹 This **removes all commit history** but keeps the files.

**💡 Final Check**

Run git status to see if Git is still tracking your files.

Let me know what exactly you want to clear so I can guide you better! 🚀

To force remove go to powershell run as administrator and enter the below command

Remove-Item -Recurse -Force .git

**1️⃣ Using GitHub CLI (gh Command)**

If you have **GitHub CLI (gh) installed**, you can list all your repositories using:

gh repo list

✅ This will display all repositories under your GitHub account.

It is possible to pull a specific file from the remote GitHub repository (origin) into your local branch (main). However, Git doesn't allow pulling a single file directly; you need to use the following process:

**Steps to Pull a Specific File**

1. **Fetch the Changes**: Use the git fetch command to update your local repository with the latest changes from the remote repository without merging them.

git fetch origin

1. **Checkout the File**: Checkout the specific file from the remote branch (origin/main) into your local branch.

git checkout origin/main -- path/to/your/file

Replace path/to/your/file with the relative path of the file you want to pull (e.g., src/myfile.txt).

1. **Commit the File**: If necessary, you can stage and commit the file after checking it out:

git add path/to/your/file

git commit -m "Pulled specific file from origin/main"

This method retrieves only the desired file without merging all changes from the remote repository. Make sure you have the correct file path and that the file exists in the remote branch.

To remove one specific file from the remote repository (origin/main) without merging other changes, you can follow these steps:

**Steps to Remove a Specific File Without Merging Other Changes:**

1. **Fetch the Latest State of the Remote Repository**: Update your local repository with the latest data from the remote repository, but without merging any changes:

git fetch origin

1. **Checkout the Remote Branch**: Create or switch to a new temporary branch that tracks the remote main branch. This ensures you're working directly with the remote branch:

git checkout -b temp-main origin/main

1. **Remove the File Locally**: Delete the file you want to remove:

rm path/to/your/file

1. **Stage the Removal**: Use git add to stage the deletion:

git add path/to/your/file

1. **Commit the Changes**: Commit the deletion with a meaningful message:

git commit -m "Removed file: path/to/your/file"

1. **Push the Changes to the Remote Repository**: Push the committed changes directly to the origin/main branch:

git push origin main

**Final Step: Verify Your Changes**

Once the file is removed, you can delete the temporary branch if it's no longer needed:

git branch -d temp-main

This process allows you to make changes to a specific file on the remote branch without merging or introducing any other local changes.