Git & GitHub

Git is a powerful version control system that helps you manage changes in your codebase, collaborate with others, and keep track of the history of your projects. Here's a basic tutorial to get you started with Git.

### 1. Installing Git

Before you start using Git, you need to install it on your machine.

### 2. Setting Up Git

After installing Git, you should configure your name and email, which will be associated with your commits.

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

git config --global user.email "you@example.com"

### 3. Initializing a Repository

To start tracking a project with Git, you need to initialize a Git repository in your project folder.

cd path/to/your/project

git init

### 4. Checking Repository Status

To see the current state of your working directory and staging area, use:

git status --> This command shows which files are modified, added, or deleted.

### 5. Adding Files to the Staging Area

To track new or modified files, you need to add them to the staging area:

git add <file>

Or, to add all changes: git add .

### 6. Committing Changes

Once your changes are staged, you can commit them to the repository with a message describing the changes:

git commit -m "Your commit message"

### 7. Viewing Commit History

To view the commit history of your repository:

git log

### 8. Creating and Switching Branches

Branches allow you to work on different versions of a project simultaneously.

* Create a new branch: git branch <branch-name>
* Switch to a branch : git checkout <branch-name>
* Create and switch to a branch : git checkout -b <branch-name>

### 9. Merging Branches

When you're ready to merge changes from one branch into another:

git checkout <branch-to-merge-into>

git merge <branch-to-merge-from>

### 10. Pushing to a Remote Repository

To share your code with others or back it up on a service like GitHub, you'll push your changes to a remote repository.

* Add a remote repository : git remote add origin https://github.com/username/repository.git
* Push changes to the remote repository : git push origin <branch-name>

### 11. Pulling Changes from a Remote Repository

To update your local repository with changes from the remote repository:

git pull origin <branch-name>

### 12. Cloning a Repository

To copy a repository from GitHub to your local machine:

git clone https://github.com/username/repository.git

### 13. Resolving Merge Conflicts

When Git encounters conflicting changes during a merge, you'll need to resolve them manually.

* Open the conflicting files and decide which changes to keep.
* After resolving, add the resolved files and commit the merge:

git add <resolved-file>

git commit

### 14. Undoing Changes

Git allows you to undo changes at various stages:

* Unstage a file : git reset HEAD <file>
* Revert a commit : git revert <commit-hash>
* Discard local changes : git checkout -- <file>

### 15. Additional Resources

To deepen your understanding of Git, you can refer to these resources:

* [Pro Git Book](https://git-scm.com/book/en/v2)
* [GitHub Learning Lab](https://lab.github.com/)
* [Atlassian Git Tutorial](https://www.atlassian.com/git/tutorials)

**These steps should help you remove the submodule and properly commit and push your changes to the remote repository.**

**Remove the Submodule Entry**: You need to remove the submodule references correctly before you can commit and push your changes.

* Deinitialize the submodule: git submodule deinit -f Frontend
* Remove the submodule entry from .git/config: Open .git/config in a text editor and look for any references to the Frontend submodule. Remove any lines related to it.

**Remove the Submodule from the Index**:

* Run the following command to remove the submodule from the index:

git rm --cached Frontend

**Delete the Frontend Directory**: If the Frontend directory is still present in your working directory and you want to remove it, delete it manually through File Explorer, or use the command: rmdir /S /Q Frontend

**Re-add and Commit the Changes**: After removing the submodule and possibly deleting the directory, re-add your files: git add .

git commit -m "Removed submodule and re-added files"

**Push the Changes to the Remote Repository**: Finally, push your changes to the remote repository: git push origin main