Git & GitHub – A Beginner-Friendly Guide with Practical Examples

What is Git?

Git is a **version control system** used by developers to keep track of changes in their code. It helps you **save snapshots** of your project at different stages, so if something breaks, you can easily go back to a previous working version.

With Git, you can:

- Work on different features or fixes without touching the main project
- Track who made what changes and when
- Avoid losing your work due to accidental mistakes

🧠 **Simple Example**:
Imagine you're writing an assignment. Every time you make progress, you save a copy with the date. If you make a mistake later, you can open the older saved version. That's how Git works — but for code.

🌐 What is GitHub?

GitHub is a **cloud-based platform** where you can **store and manage your Git repositories online**. It lets you **collaborate with others**, work on code together, review changes, and even contribute to open-source projects.

With GitHub, you can:

- Push your local Git projects to the cloud
- Invite team members to work on the same codebase
- Create branches, raise issues, and do pull requests for team collaboration
- Keep your code safe and accessible from anywhere

🧠 **Simple Example**:
Think of GitHub as **Google Drive for your code**. You upload your assignment (code) to GitHub so your teammates can view, comment, or help you improve it — all from anywhere..

Essential Git Commands with Examples

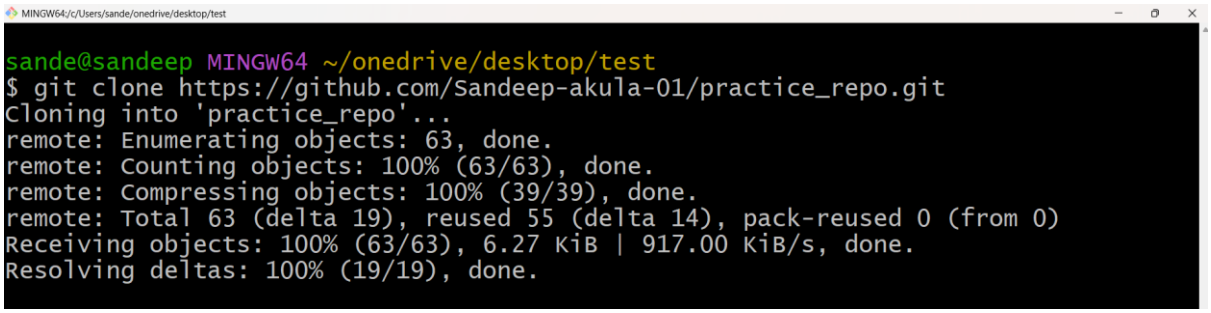
1. git config

Sets your username and email for your commits, so others know who made changes.
Use case: Before working on a project, you configure your name and email once to identify your commits.

```
sande@sandeep MINGW64 ~/OneDrive/desktop/test/practice_repo (main)
$ git config --global user.name "Sandeep"

sande@sandeep MINGW64 ~/OneDrive/desktop/test/practice_repo (main)
$ git config --global user.email "sandeepsandy958168@gmail.com"
```

2. git clone

Copies a remote repo to your computer so you can work locally.
Use case: When you join a new project, you clone it to start working on the code.



---

3. git status

Shows which files have been changed, added, or need to be committed.
Use case: Before committing, you check git status to see what's ready to save.

---

4. git add .

Stages all changed files to be included in the next commit.
Use case: After editing multiple files, you run git add . to prepare all changes for commit.

---

5. git commit -m "message"

Saves staged changes with a description message.
Use case: After finishing a feature or bug fix, you commit with a message describing your work.

---

6. git pull

Downloads changes from the remote repo and merges them into your local copy.

Use case: Before starting work, you pull to update your local repo with teammates' latest changes.

---

7. git push

Uploads your commits from local to the remote repository (e.g., GitHub).
Use case: Once your work is done, you push to share it with your team.

```
sande@sandeep MINGW64 ~/onedrive/desktop/test/practice_repo (aws)
$ git push origin aws
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 425 bytes | 425.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'aws' on GitHub by visiting:
remote:      https://github.com/Sandeep-akula-01/practice_repo/pull/new/aws
remote:
To https://github.com/Sandeep-akula-01/practice_repo.git
 * [new branch]      aws -> aws
```

---

8. git branch

Lists branches or creates a new one to work separately.
Use case: You create a branch to develop a new feature without affecting the main code

---

9. git checkout

Switches to a different branch or restores files.
Use case: You switch branches to work on a bug fix or feature.

---

10. git merge

Combines changes from one branch into another.
Use case: After finishing a feature branch, you merge it into the main branch.

```
sande@sandeep MINGW64 ~/onedrive/desktop/test/practice_repo (aws)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

sande@sandeep MINGW64 ~/onedrive/desktop/test/practice_repo (main)
$ git merge aws
Updating 419528e..578b3c2
Fast-forward
 aws | 9 +++++++++
 1 file changed, 9 insertions(+)
 create mode 100644 aws
```

11. Merge conflicts

Happen when two branches change the same code in different ways.
Use case: You manually fix conflicting lines before completing a merge.

```
sande@sandeep MINGW64 ~/onedrive/desktop/test/practice_repo (main)
$ git merge new_ft
Auto-merging arth.sh
CONFLICT (add/add): Merge conflict in arth.sh
Automatic merge failed; fix conflicts and then commit the result.

sande@sandeep MINGW64 ~/onedrive/desktop/test/practice_repo (main|MERGING)
$ vim arth.sh
```

Conflict Resolving

---

12. Fork

Makes your own copy of someone else's repo to experiment freely.
Use case: You fork a popular project to add your improvements without changing the original.

---

13. git revert

Creates a new commit to undo a past change safely.
Use case: You revert a commit that introduced a bug without removing later commits.

14. git log

Shows history of commits with details.
Use case: You look at the log to find when and who made specific changes.

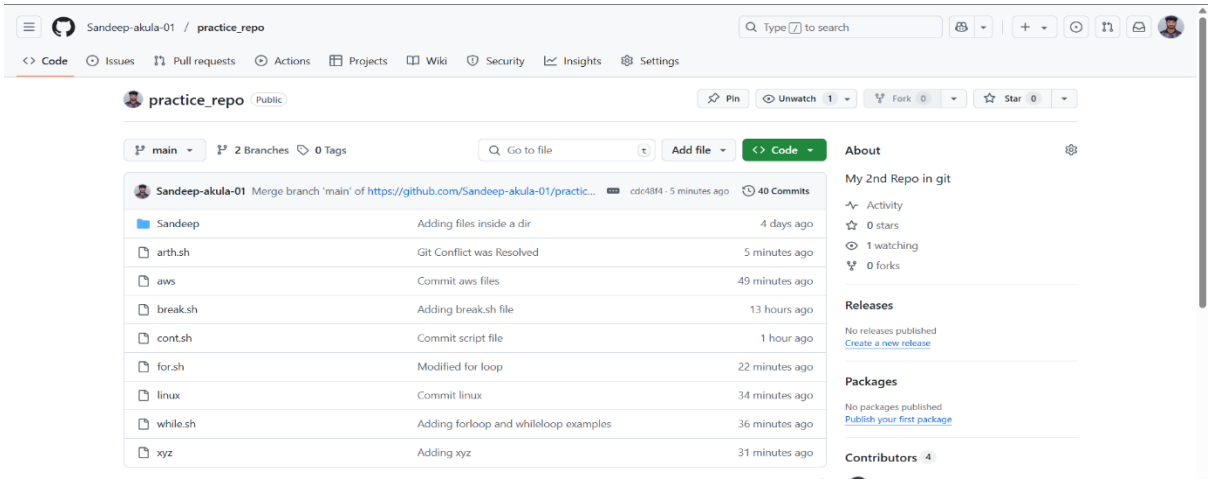
```
commit 252b1ae713f99614b48e840ef39a968b469f9dfa
Author: Sandeep Akula <sandeepsandy958168@gmail.com>
Date:   Tue Jun 3 10:45:22 2025 +0530

    Adding forloop and whileloop examples

commit 41d33ef37d8a6cd5b61f3254ee211a325a1c0b4d
Author: Sandeep Akula <sandeepsandy958168@gmail.com>
Date:   Tue Jun 3 10:39:24 2025 +0530

    Deleting empty files

commit 578b3c2bb7b46af2d397a8da3705cf0e89e4d542
Author: Sandeep Akula <sandeepsandy958168@gmail.com>
Date:   Tue Jun 3 10:32:40 2025 +0530

    Commit aws files

commit 419528e25bf2cd334b9c26795059ae22777cbe9b
Author: Sandeep Akula <sandeepsandy958168@gmail.com>
Date:   Tue Jun 3 10:25:43 2025 +0530

    Commit script file

commit 9c6cb94b17c5adfc5c5b0825bba801c7d3d45bb1
Author: Sandeep Akula <sandeepsandy958168@gmail.com>
Date:   Mon Jun 2 22:06:05 2025 +0530

    Adding break.sh file
```

---

15. git reset

Moves your branch to an earlier commit, undoing changes locally.
Use case: You reset to undo recent commits before pushing to fix mistakes.

My Practice Repo



📌 Conclusion & Personal Takeaways

Over the past few days, I've explored and practiced the core concepts of Git and GitHub as the first step in my DevOps journey.
Through hands-on exercises, I learned how version control works in real-world development environments and how teams collaborate effectively using Git.

- 🔹 I now understand how to:
  - Track code changes using Git.
  - Work with branches and merge them safely.
  - Resolve merge conflicts.
  - Clone, fork, and push code to GitHub.
  - Use important commands like git add, commit, pull, log, revert, and more.

In this document, I've chosen to highlight and explain some of the **core foundational commands and concepts** for clarity and simplicity. However, my learning journey covered a much broader set of topics and practical scenarios beyond what's shown here.

🔗 Let's Connect!

If you found this helpful or would like to collaborate or share suggestions, feel free to connect with me:

- 🌐 **LinkedIn:** https://www.linkedin.com/in/sandeep-akula-360716244/
- 💻 **GitHub:** https://github.com/Sandeep-akula-01