**Viewing Commit History**

**Purpose:**

Understand what changes have been made over time.

**Key Commands:**

git log # Show commit history

$ git log

commit 6c73e137d09bf425bd5819e2913b6d730cb7f920 (**HEAD** -> **main**)

Author: test <abc@gmail.com>

Date: Thu Jul 31 04:49:07 2025 +0530

new file

git log --oneline --graph # Visual, concise view

$ git log --oneline --graph

\* 6c73e13 (**HEAD** -> **main**) new file

git show <commit-hash> # See details of a specific commit

$ git show 6c73e137d09bf425bd5819e2913b6d730cb7f920

commit 6c73e137d09bf425bd5819e2913b6d730cb7f920 (**HEAD** -> **main**)

Author: test <abc@gmail.com>

Date: Thu Jul 31 04:49:07 2025 +0530

new file

**diff --git a/new.txt b/new.txt**

**new file mode 100644**

**index 0000000..6bfa5cd**

**--- /dev/null**

**+++ b/new.txt**

@@ -0,0 +1 @@

+this is new files

**Undoing Changes**

**git reset**

* **Use Case**: Roll back commits (be careful—this can remove history).
* **Types**:
  + --soft: Keep changes in staging area
  + --mixed: Keep changes in working directory
  + --hard: Discard changes permanently

If you’re not ready to commit your current work:

Temporarily save work when switching branches.

git stash

git stash pop

**git revert**

* **Use Case**: Undo a commit safely by creating a new one.

$ git revert 6c73e137d09bf425bd5819e2913b6d730cb7f920

[main f3ce166] This reverts commit 6c73e137d09bf425bd5819e2913b6d730cb7f920.

1 file changed, 1 deletion(-)

delete mode 100644 new.txt

Connecting Local to GitHub

git init

add remote

git remote add origin <https://github.com/username/my-first-repo.git>

Pushing Changes

git add .

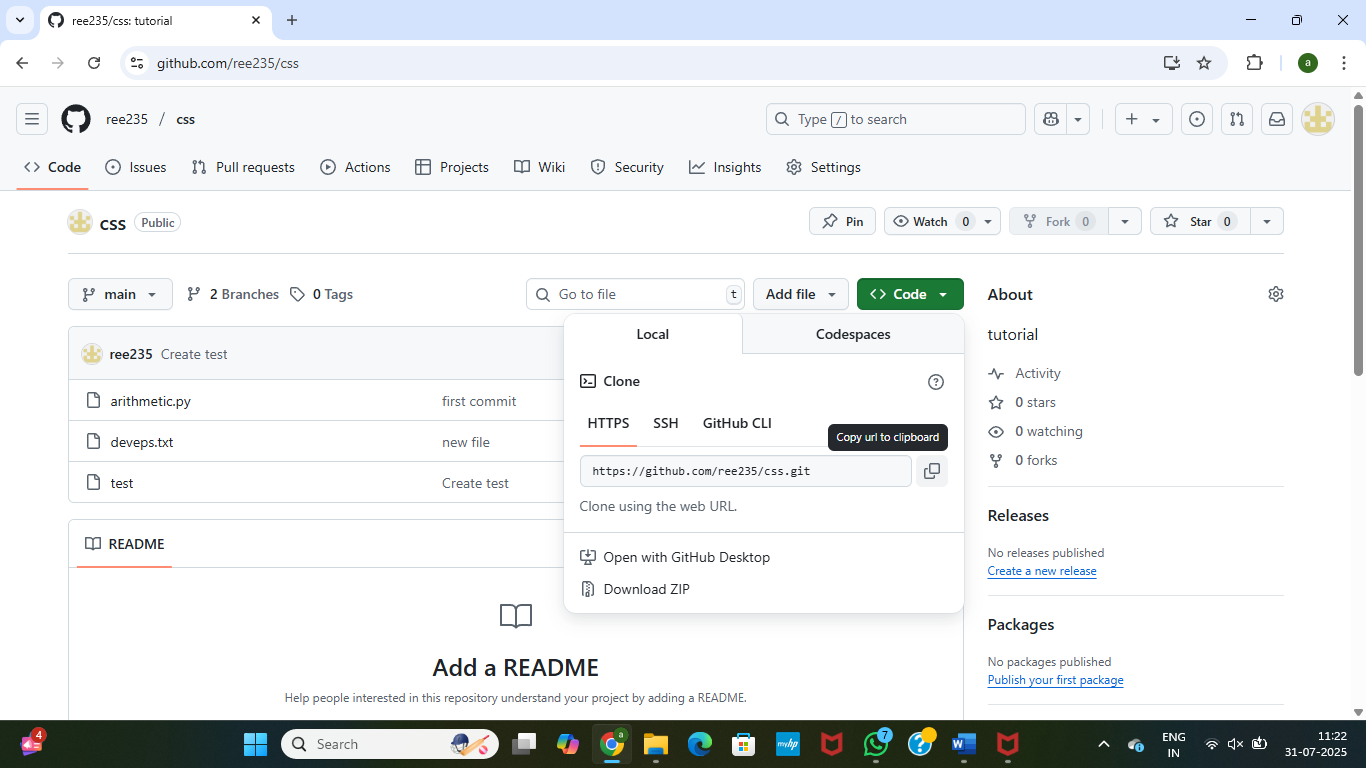
git commit -m "First commit"

git push -u origin main

git clone <https://github.com/username/my-first-repo.git>

git pull origin main # Pull changes and merge

git fetch origin # Fetch changes (no merge)



git commands:

git remote -v

use these url in

origin https://github.com/ree235/css.git (fetch)-- git fetch

origin https://github.com/ree235/css.git (push)-- git push

**Git Fork**

**A fork is a copy of a repository on GitHub (or another Git hosting platform) that belongs to your own account.**

**Making your own independent copy of someone else’s project so you can experiment, contribute, or modify it without affecting the original.**

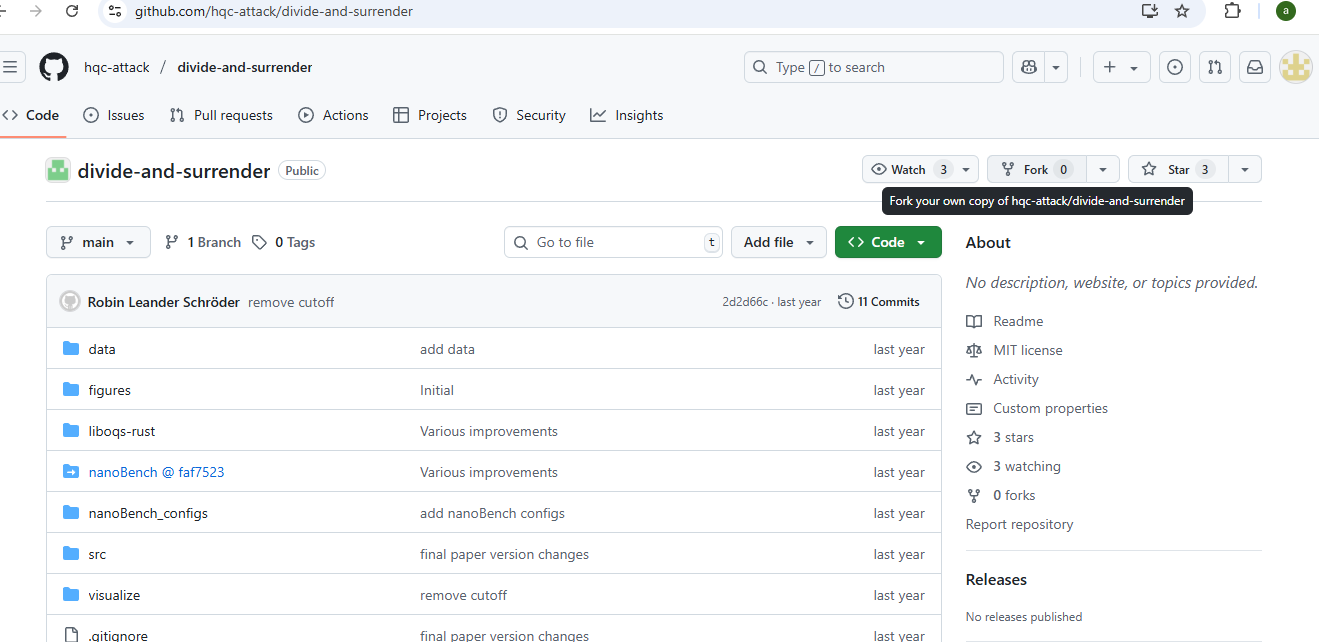
**Step 1: Fork the Repository**

1. **Go to the GitHub repo you want to fork.**
2. **Click the "Fork" button (top-right).**
3. **GitHub will create a copy in your account.**

**🔹 Step 2: Clone Your Fork Locally**

**git clone https://github.com/your-username/the-forked-repo.git**

**cd the-forked-repo**

****

**What is Cloning?**

**🔹 git clone downloads a copy of any repo to your local computer.**

You can view, edit, and experiment locally.

**You can't push changes** unless you're a collaborator on that repo.

**What is Forking?**

**🔹 Forking makes a copy of a GitHub repo under your own account.**

* You can **edit freely**, **push changes**, and create branches.
* You then **clone** your fork if you want to work locally.

You must **fork first**, then **clone** if you want to contribute to a project you don’t own.

**GitHub Issues and Project Boards**

**: Creating Issues**

1. Go to the **Issues** tab.
2. Click **"New issue"**.
3. Add title and description.
4. Assign to a team member.
5. Use labels (e.g., bug, enhancement).

**Creating Project Boards**

1. Go to the **Projects** tab.
2. Create a **Kanban-style board** with columns like:
   * To Do
   * In Progress
   * Done
3. Add issues as **cards** in the board.
4. Drag and drop to update progress.