Absolutely! Here’s an **updated Git & GitHub command guide**, now including **branch renaming commands**:

Git & GitHub Complete Commands Guide

1. Git Configuration

- `git config --global user.name "Your Name"` – Sets your Git username.
- `git config --global user.email "your.email@example.com"` – Sets your Git email.
- `git config --list` – Displays all configured Git settings.
- `git config --global core.editor "vim"` – Sets the default text editor for Git.
- `git config --global alias.st status` – Creates a shortcut alias for `git status`.

2. Git Basics

- `git init` – Initializes a new Git repository.
- `git clone <repo_URL>` – Creates a local copy of a remote repository.
- `git status` – Shows the current state of the working directory.
- `git add <file>` – Stages specific files for the next commit.
- `git add .` – Stages all changes.
- `git commit -m "message"` – Saves staged changes with a descriptive commit message.
- `git commit --amend -m "new message"` – Modifies the last commit message.
- `git rm <file>` – Removes a file from the repository and stages the deletion.

3. Branching & Merging

- `git branch` – Lists all branches.
- `git branch <branch_name>` – Creates a new branch.
- `git checkout <branch_name>` – Switches to an existing branch.
- `git checkout -b <branch_name>` – Creates and switches to a new branch.
- `git merge <branch_name>` – Merges another branch into the current branch.
- `git branch -d <branch_name>` – Deletes a local branch.

- `git branch -D <branch_name>` – Force deletes a branch.
- `git rebase <branch_name>` – Reapplies commits on top of another branch.

4. Renaming Branches

Rename the current branch locally

```
 git branch -m <new_branch_name>
```

-

Rename a different branch locally

```
 git branch -m <old_branch_name> <new_branch_name>
```

-

Push the renamed branch to remote

```
 git push origin <new_branch_name>
```

-

Delete the old branch from remote

```
 git push origin --delete <old_branch_name>
```

-

Update tracking for the renamed branch

```
 git branch --unset-upstream <new_branch_name>
git push -u origin <new_branch_name>
```

-

5. Working with Remotes

- `git remote -v` – Lists connected remote repositories.
- `git remote add origin <repo_URL>` – Links local repository to a remote.
- `git push -u origin <branch_name>` – Pushes local commits to a remote repository.
- `git pull origin <branch_name>` – Fetches and updates local repository with remote changes.
- `git fetch` – Retrieves updates from a remote repository without merging them.
- `git remote remove <remote_name>` – Removes a remote repository.

6. Undoing Changes

- `git reset --soft <commit_hash>` – Resets to a specific commit but keeps changes staged.
- `git reset --hard <commit_hash>` – Resets to a specific commit and deletes all changes.
- `git revert <commit_hash>` – Creates a new commit that undoes changes from a previous commit.
- `git checkout -- <file>` – Discards changes in a specific file.
- `git restore <file>` – Restores a file to its last committed state.

7. Stashing Work

- `git stash` – Saves uncommitted changes temporarily.
- `git stash apply` – Restores the latest stashed changes.
- `git stash drop` – Deletes the latest stash entry.
- `git stash list` – Displays all stashed changes.
- `git stash pop` – Applies and removes the latest stash.

8. Collaboration on GitHub

- `git push origin main` – Pushes the local repository to GitHub.
- `git pull origin main` – Pulls updates from GitHub.
- `git fork <repo_URL>` – Copies another repository into your GitHub account.
- `git clone <your_forked_repo_URL>` – Clones a forked repository locally.
- `git fetch upstream` – Retrieves changes from the original repo.
- `git merge upstream/main` – Merges latest changes from the original repo into your branch.
- `git pull --rebase origin main` – Pulls changes while keeping your commits on top.

9. Viewing History

- `git log` – Displays a list of commit history.
- `git log --oneline` – Shows a brief history of commits.
- `git log --graph --decorate --all` – Displays a visual representation of commit history.
- `git diff` – Compares changes between commits.
- `git blame <file>` – Shows who last modified each line of a file.

10. Tagging Releases

- `git tag <tag_name>` – Creates a new tag for a commit.
- `git tag -a <tag_name> -m "message"` – Creates an annotated tag.
- `git push origin <tag_name>` – Pushes a tag to the remote repository.
- `git tag -d <tag_name>` – Deletes a local tag.
- `git push origin --delete <tag_name>` – Deletes a remote tag.

11. Advanced Git Commands

- `git cherry-pick <commit_hash>` – Applies a specific commit to the current branch.
- `git reflog` – Shows a history of all changes, including deleted commits.
- `git bisect start` – Helps find a bug by performing a binary search through commits.
- `git shortlog -sn` – Displays commit statistics by author.
- `git archive --format=zip HEAD > archive.zip` – Creates a zip archive of the repository.

This guide now includes **branch renaming commands** along with all essential Git & GitHub commands. You can copy and paste this into a document and save it as a **PDF** using Microsoft Word or Google Docs. Let me know if you need any refinements! 🚀