GIT / GITHUB

1. **git config --global user.name Muzamil**

   Running this command sets the global Git configuration for the user's name on your system to "Muzamil". This means that whenever you make a commit in any repository on your system, Git will associate that commit with the name "Muzamil"

2. **git config --global user.email m42@gmail.com**

   Running this command sets the global Git configuration for the user's email address on your system to "m42@gmail.com". This means that whenever you make a commit in any repository on your system, Git will associate that commit with the email address "m42@gmail.com".

3. **git init**

   When you run git init in a directory, Git creates a new repository in that directory. This repository will contain all the necessary files and directories that Git needs to track changes to your project's files.

4. **git status**

   The git status command is used to display the current state of the repository. When you run git status, Git will show you information about:
   - Which branch you are currently on.
   - Any changes you've made to files in your working directory.
   - Untracked files (files that Git is not currently managing).
   - Files that are staged (files that are ready to be committed).

5. **git add index.html**

   Running git add index.html will stage the file named index.html for the next commit in your Git repository.

6. **git commit**

   Running git commit without any additional options or arguments will typically open your default text editor (such as Vim or Nano) for you to enter a commit message.

7. **touch about.html**

   The touch about.html command creates a new file named about.html in the current directory. The touch command is commonly used in Unix-like operating systems to create empty files or update the timestamp of existing files.

8. **git add -A**

   The command git add -A is used to stage all changes in your working directory for the next commit, including modified, deleted, and new files.

9. **git commit -m**

   git commit -m "your commit message" is a command used to commit changes in Git, with the -m flag allowing you to provide a commit message directly from the command line.

GIT / GITHUB

10. git checkout contact.html

The command “git checkout contact.html is used to restore the state of the file named contact.html to the version that exists in the current branch or in the branch specified.

11. git checkout -f

`git checkout -f` forcefully switches branches, discarding local changes if necessary, ensuring the working directory matches the target branch's state.

12. git log

The git log command is used to display a history of commits in the current branch. When you run git log, Git will output a list of commits, showing information such as the commit hash, author, date, and commit message for each commit in reverse chronological order (from the most recent to the oldest).

13. git log -p -2

The command git log -p -2 displays the history of the last two commits in the repository along with the patch (or diff) showing the changes introduced by each commit.

14. git diff

`git diff` shows the difference between the changes in your working directory and the staging area. Adding a filename after `git diff` shows changes between the working directory and the last commit.

15. git diff --staged

`git diff --staged` shows the difference between the changes that are staged (added with `git add`) and the last commit.

16. git commit -a -m

The command `git commit -a -m` allows you to commit all changes, including modifications and deletions, without explicitly staging them using `git add`, while providing a commit message inline with the `-m` flag.

17. git rm waste.html

The command `git rm waste.html` removes the file named `waste.html` from both your working directory and the Git repository.

18. git rm --cached waste.html

The command `git rm --cached waste.html` removes the file named `waste.html` from the Git repository (i.e., from the index or staging area) but keeps it in your working directory.

19. git status -s

The command `git status -s` provides a short and concise summary of the current state of the repository. It displays a list of modified, untracked, and staged files in abbreviated form, with two characters representing the status of each file.

GIT / GITHUB

20. touch .gitignore

Running `touch .gitignore` creates a new file named `.gitignore` in the current directory. This file is typically used to specify intentionally untracked files that Git should ignore. It's commonly used to exclude files generated by the build process, logs, and other files that are not necessary to track in version control.

21. touch mylog.log

Running `touch mylog.log` creates a new file named `mylog.log` in the current directory. This command is often used to create an empty file, in this case, a log file named `mylog.log`.

22. git branch feature

The command `git branch feature` creates a new branch named "feature" in your Git repository. This branch will initially be identical to the branch you were on when you ran the command.

23. git checkout feature

The command `git checkout feature` switches to the branch named "feature" in your Git repository. This means you'll be working within the context of the "feature" branch, and any changes you make and commits you create will be on this branch.

24. git checkout feature

The command git remote set-url origin https sets the URL of the remote repository named "origin" to "https".

25. git merge feature

The command `git merge feature` merges the changes from the "feature" branch into the current branch. This integrates the changes made in the "feature" branch into the branch you are currently on.

26. git checkout -b ‘php’

The command `git checkout -b 'php'` creates a new branch named "php" and switches to it. This is often used when you want to create a new branch and immediately start working on it.

27. git remote

The command `git remote` is used to manage remote repositories associated with your local repository. It shows the names of the remote repositories you've configured for the current Git repository.

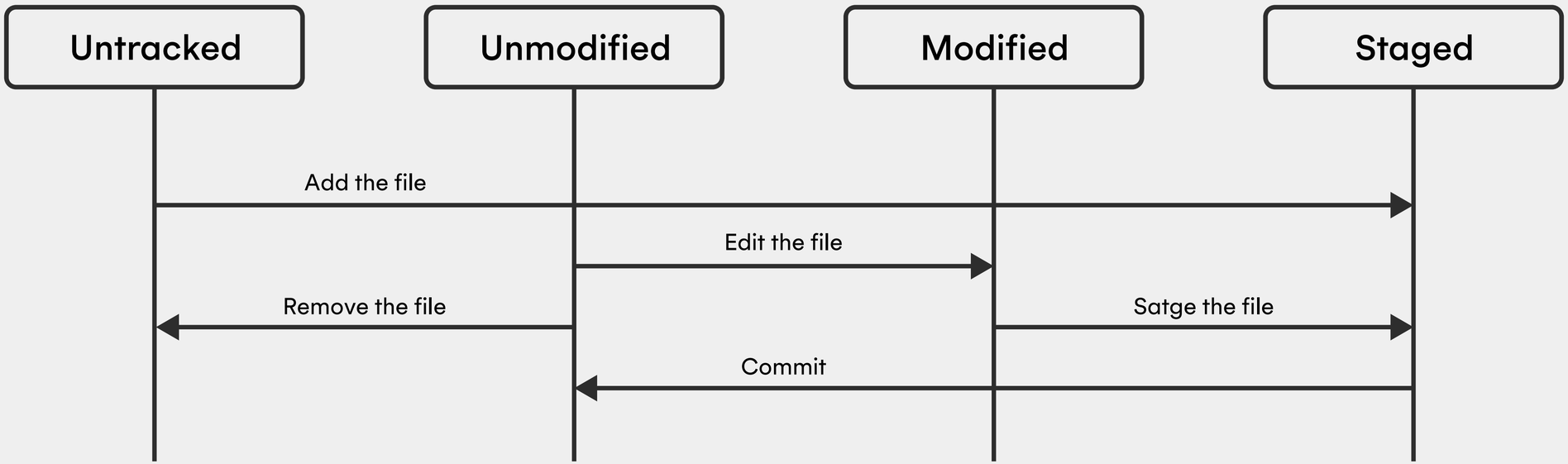
28. git remote -v

The command `git remote -v` lists the names and URLs of the remote repositories associated with your local repository. It shows both the fetch and push URLs for each remote repository.

29. git push origin master

The command `git push origin master` pushes the commits from your local "master" branch to the remote repository named "origin". This updates the "master" branch in the remote repository with the changes you've made locally.

GIT / GITHUB


Untracked
Unmodified
Modified
Staged
Add the file
Edit the file
Remove the file
Satge the file
Commit