HOW TO GIT

NUMBER 1:

The list of things required:

A computer capable of installing GitHub

A web browser

Moderately strong internet connection

After obtaining these resources, it is time to create your very first GitHub account. ­­­­

A screenshot of a cell phone

Description automatically generated

1. Enter a unique username
2. Enter your personal or work email
3. Create a strong password
4. Verify you are not a robot by solving the puzzle

NUMBER 2

Congratulations on the new GitHub account! It is time to create a repository for you and your team.

A screenshot of a cell phone

Description automatically generated­­

1. The first step is to name the repository so you can have a way to identify it.
2. Second step is to add description and inform people what the aim and goal of the repository is.
3. You can choose to make it public, meaning anyone can view it. Or private where only added members can view it.
4. This is so the repository can be cloned. It will also add a readme text file to the repository.

NUMBER 3: The next step is to launch a terminal and start coding! Here are some basic commands to get you started.

**git config**

Usage: git config –global user.name “[name]”

Usage: git config –global user.email “[email address]”

This command sets the author name and email address respectively to be used with your commits.

Git Config Command - Git Commands - Edureka

**git init**

Usage: git init [repository name]

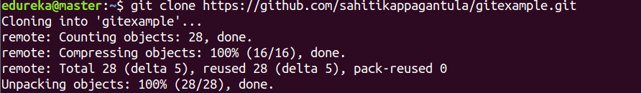
This command is used to start a new repository.

GitInit Command - Git Commands - Edureka

**git clone**

Usage: git clone [url]

This command is used to obtain a repository from an existing URL.



**git add**

Usage: git add [file]

This command adds a file to the staging area.

Usage: git add \*

This command adds one or more to the staging area.

Git Add Command - Git Commands - Edureka

**git commit**

Usage: git commit -m “[ Type in the commit message]”

This command records or snapshots the file permanently in the version history.



Usage: git commit -a

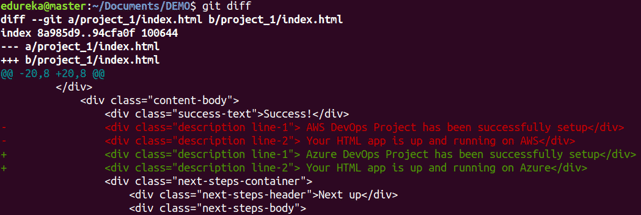
This command commits any files you’ve added with the git add command and also commits any files you’ve changed since then.

Git Commit Command - Git Commands - Edureka

**git diff**

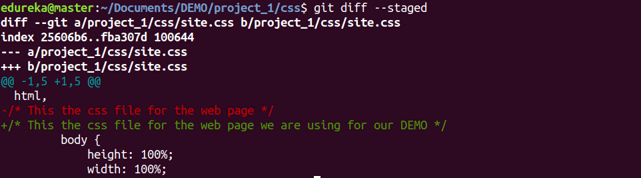
Usage: git diff

This command shows the file differences which are not yet staged.



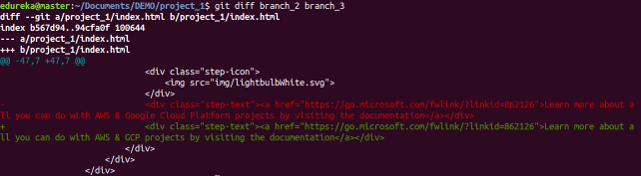
 Usage: git diff –staged

This command shows the differences between the files in the staging area and the latest version present.



Usage: git diff [first branch] [second branch]

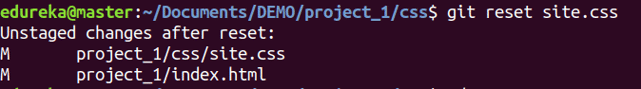
This command shows the differences between the two branches mentioned.



**git reset**

Usage: git reset [file]

This command unstages the file, but it preserves the file contents.



Usage: git reset [commit]

This command undoes all the commits after the specified commit and preserves the changes locally.

Git Reset Command - Git Commands - Edureka

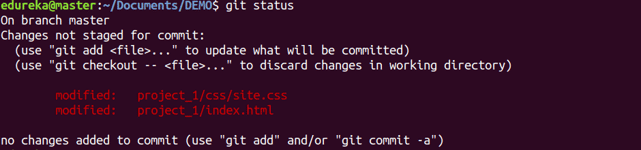
Usage: git reset –hard [commit]  This command discards all history and goes back to the specified commit.

Git Reset Command - Git Commands - Edureka

**git status**

Usage: git status

This command lists all the files that have to be committed.



**git rm**

Usage: git rm [file]

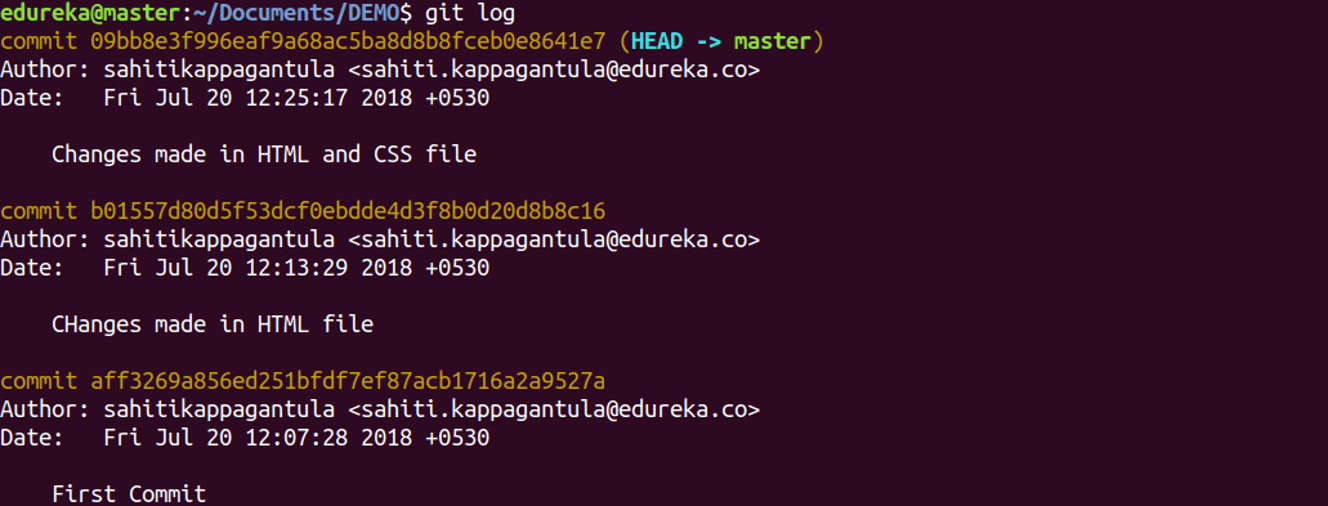
This command deletes the file from your working directory and stages the deletion.

Git Rm Command - Git Commands - Edureka

**git log**

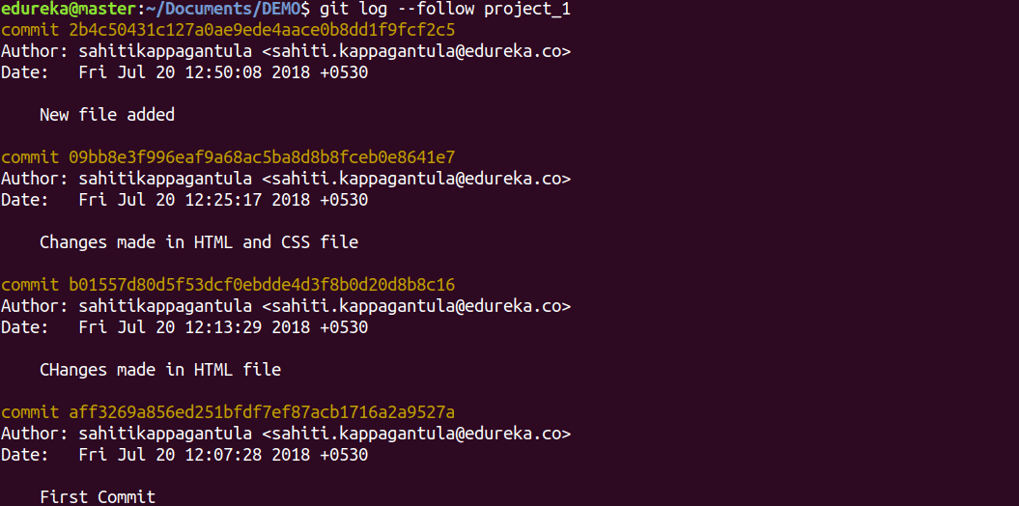
Usage: git log

This command is used to list the version history for the current branch.



Usage: git log –follow[file]

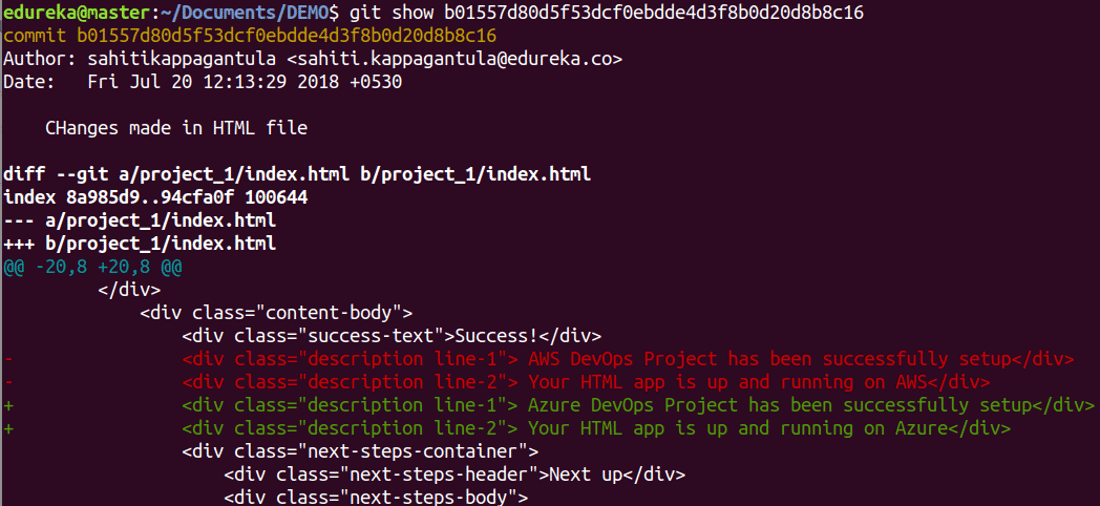
This command lists version history for a file, including the renaming of files also.



**git show**

Usage: git show [commit]

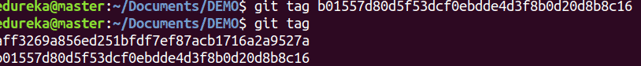
This command shows the metadata and content changes of the specified commit.



**git tag**

Usage: git tag [commitID]

This command is used to give tags to the specified commit.



**git branch**

Usage: git branch

This command lists all the local branches in the current repository.

Git Branch Command - Git Commands - Edureka

Usage: git branch [branch name]

This command creates a new branch.

Git Branch Command - Git Commands - Edureka

Usage: git branch -d [branch name]

This command deletes the feature branch.

Git Branch Command - Git Commands - Edureka

**git checkout**

Usage: git checkout [branch name]

This command is used to switch from one branch to another.

Git Checkout Command - Git Commands - Edureka

Usage: git checkout -b [branch name]

This command creates a new branch and also switches to it.

Git Checkout Command - Git Commands - Edureka

**git merge**

Usage: git merge [branch name]

This command merges the specified branch’s history into the current branch.

Git Merge Command - Git Commands - Edureka

**git remote**

Usage: git remote add [variable name] [Remote Server Link]

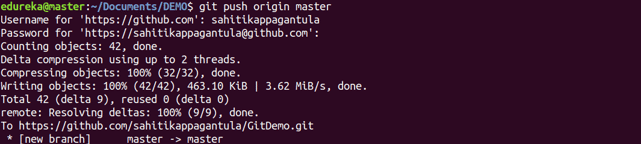
This command is used to connect your local repository to the remote server.

Git Remote Command - Git Commands - Edureka

**git push**

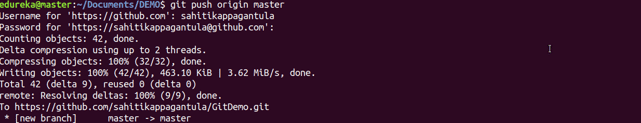
Usage: git push [variable name] master

This command sends the committed changes of master branch to your remote repository.



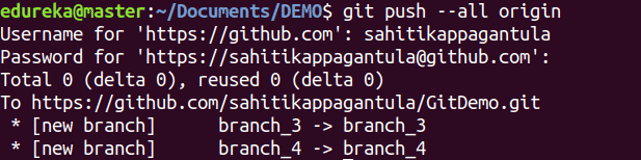
Usage: git push [variable name] [branch]

This command sends the branch commits to your remote repository.



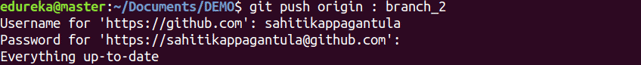
Usage: git push –all [variable name]

This command pushes all branches to your remote repository.



Usage: git push [variable name] :[branch name]

This command deletes a branch on your remote repository.



**git pull**

Usage: git pull [Repository Link]

This command fetches and merges changes on the remote server to your working directory.



**git stash**

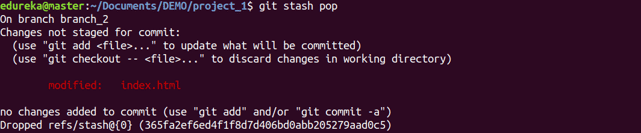
Usage: git stash save

This command temporarily stores all the modified tracked files.

Git Stash Command - Git Commands - Edureka

Usage: git stash pop

This command restores the most recently stashed files.



Usage: git stash list

This command lists all stashed changesets.

Git Stash Command - Git Commands - Edureka

Usage: git stash drop

   command discards the most recently stashed changeset.

Git Stash Command - Git Commands - Edureka