Git commands

git config --global user.name "akhil626626"

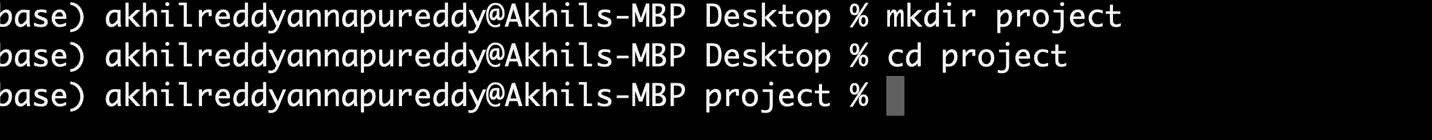
git config --global user.email [akhilreddy11100@gmail.com](mailto:akhilreddy11100@gmail.com)

then create the access key

Create the directory using below command.

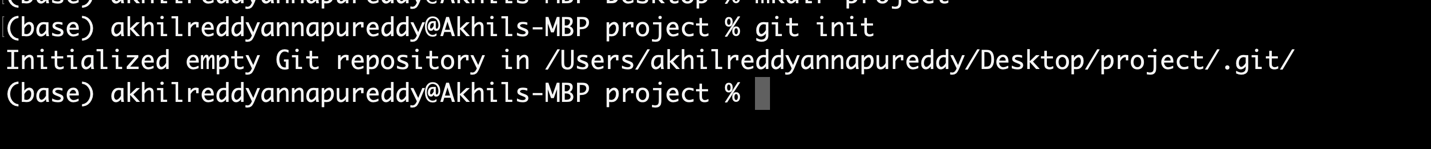
mkdir project

cd project



Then we have use below command to get the history, it will saved in .git folder. To gets the .git folder user below command

git init

.

A screen shot of a computer program

Description automatically generated

git status

Certainly, here's a short description of the **git status** command:

**git status** - A Git command that provides an overview of the current state of your Git repository. It displays information about changes to tracked files (modified, staged, or deleted), untracked files (new files not yet added to the repository), and branch information, helping you understand the status of your repository before performing further Git operations.

A black screen with white text

Description automatically generated

Git add names.txgt

Git commit -m “names files is created”

git remote add origin https://github.com/akhil626626/git\_01

**git push origin main**

creating the new branch.

**git branch master**

**now checking out to that branch.**

**(base) akhilreddyannapureddy@Akhils-MBP projectnew % git checkout master**

**Git log**

**git log** is a Git command used to display a chronological list of commits in a Git repository. It provides information such as commit hashes, authors, dates, and commit messages, allowing users to track the history of changes made to the repository. Users can customize the output and filter commits based on various criteria, making it a powerful tool for version control and collaboration in software development.

**git restore --staged 1**

**git reset <commit id> (it will remove all the commits above commit id>**

**git stash (it will help to use keep the files into back staging area)**

**git stash pop (it will use to come to staging area)**

**git stash clear (it used to delete the files in the back staging area)**

**git remote -v**

**git fetch --all –prune**

**git reset --hard upstream/main**

**git rebase -I <commit id>**

**git cherry-pick <commit id>**

**git merge (updated in up )**

**if remote origin already existes**

**A screen shot of a computer

Description automatically generated**