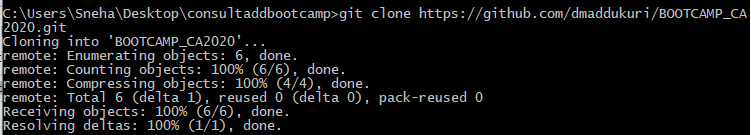
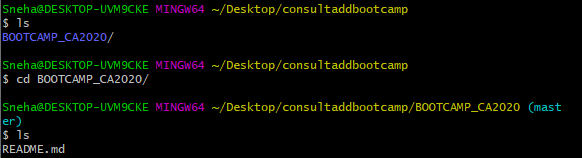
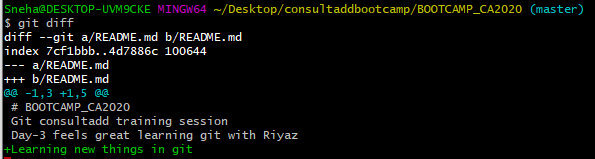
**TASK 01:**

Git Clone

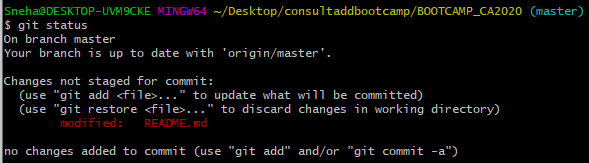




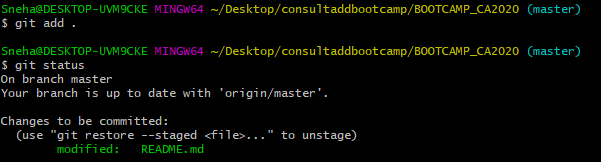
Git diff



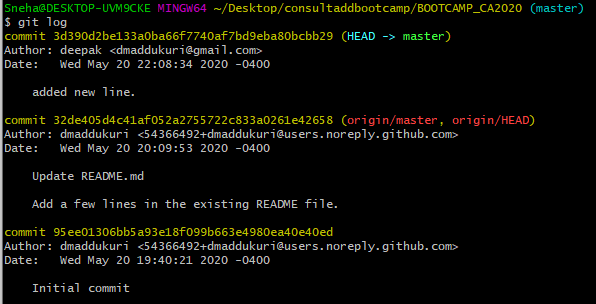
Git status



Git add .  & Git add filename and Git status



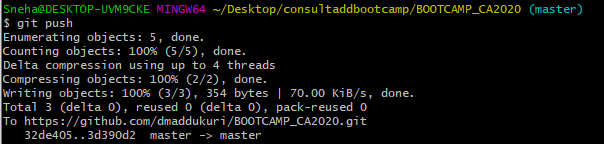
Git log



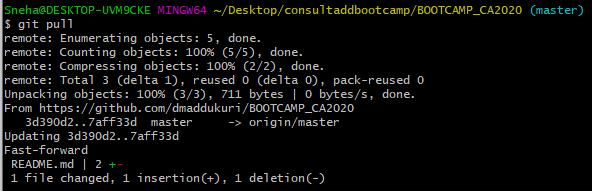
Git commit -m “message here”



Git push



Git pull



Git reset



Git config --global username “usernamehere”

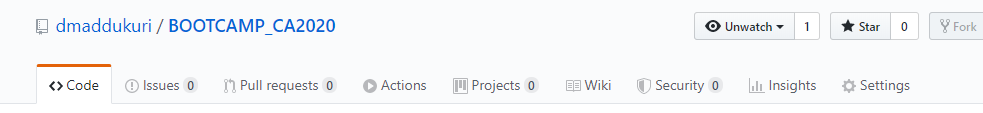


Git config --global user.email “emailhere”

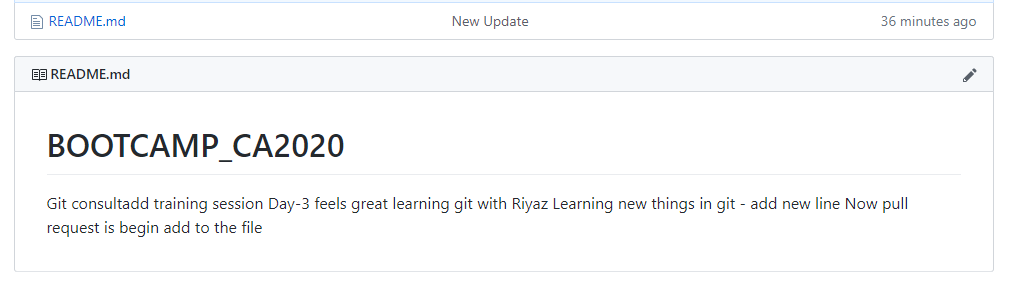


**TASK 02:**

* Create an account on GitHub.
* Create a repository with  the name “BOOTCAMP\_CA2020”.



* Make sure to create this repo with README.md file where you can write necessary information of what this repo is all about.



* Kindly, push both of your assignments as a separate file of each day.

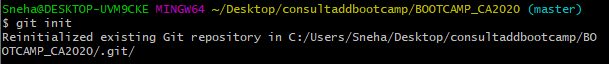
<https://github.com/dmaddukuri/BOOTCAMP_CA2020>

**NOTE:** Files that you have created on Google Doc please download that as doc file (Separate day1 and day2 task in individual file ) and push them on git.

Learn the concepts of Branching and Merging and create a doc on it. Push the same doc on Git as a separate file named it as day3\_task.txt.

<https://github.com/dmaddukuri/BOOTCAMP_CA2020>

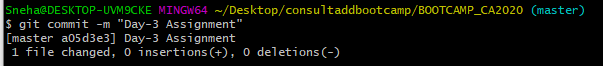
git init



git add



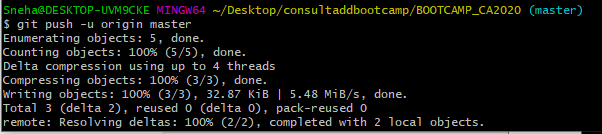
git commit -m



Git remote add origin url



Git push



**Braching**

Now you get the call that there is an issue with the website, and you need to fix it immediately. With Git, you don’t have to deploy your fix along with the changes you’ve made, and you don’t have to put a lot of effort into reverting those changes before you can work on applying your fix to what is in production. All you have to do is switch back to your master branch.

However, before you do that, note that you working directory or staging area has uncommitted changes that conflict with the branch you’re checking out, Git won’t let you switch branches. It’s best to have a clean working state when you switch branches. There are ways to get around this (namely, stashing and commit amending) that we’ll cover later on, in [Stashing and Cleaning](https://git-scm.com/book/en/v2/ch00/_git_stashing). For now, let’s assume you’ve committed all your changes, so you can switch back to your master branch

**Merging**

work is complete and ready to be merged into your master branch. In order to do that, you’ll merge your branch into master, much like you merged your hotfix branch earlier. All you have to do is check out the branch you wish to merge into and then run the git merge command

Instead of just moving the branch pointer forward, Git creates a new snapshot that results from this three-way merge and automatically creates a new commit that points to it. This is referred to as a merge commit, and is special in that it has more than one parent.

Occasionally, this process doesn’t go smoothly. If you changed the same part of the same file differently in the two branches you’re merging, Git won’t be able to merge them cleanly.

Anything that has merge conflicts and hasn’t been resolved is listed as unmerged. Git adds standard conflict-resolution markers to the files that have conflicts, so you can open them manually and resolve those conflicts.

Commands used in Braching git branch, git branch<brach\_name>, git checkout -b <branch\_name>, git commit -a, git branch -d <branch\_name>, git merge<brach\_name>.

**TASK 03:**

* What is an Elastic IP and how it is different from Dynamic IP?

**Elastic IP**

An Elastic IP address is a static IPv4 address designed for dynamic cloud computing. An Elastic IP address is associated with your AWS account. With an Elastic IP address, you can mask the failure of an instance or software by rapidly remapping the address to another instance in your account.

To use an Elastic IP address, you first allocate one to your account, and then associate it with your instance or a network interface.

Elastic IPs are totally free, as long as they are being used by an instance. However, Amazon will charge you $0.01/hr for each EIP that you reserve and do not use. You will be charged if you ever remap an EIP more than 100 times in a month.

when an instance is terminated the public IP attached to it gets released and further when you relaunch the same instance new IP address is assigned. Elastic IP do not change and they remain same even if you terminate the instance and later again restart the same instance.

**Dynamic IP**

A dynamic IP address is an IP address that changes from time to time unlike a static IP address. Most home networks are likely to have a dynamic IP address and the reason for this is because it is cost effective for Internet Service Providers (ISP's) to allocate dynamic IP addresses to their customers.

A dynamic Internet Protocol address (dynamic IP address) is a temporary IP address that is assigned to a computing device or node when it's connected to a network. A dynamic IP address is an automatically configured IP address assigned by a DHCP server to every new network node.

* What is the Client Server Model, Explain in detail?

Client-Server Model. The client-server model describes how a server provides resources and services to one or more clients. Examples of servers include web servers, mail servers, and file servers. Each of these servers provide resources to client devices, such as desktop computers, laptops, tablets, and smartphones.

Client–server model is a [distributed application](https://en.wikipedia.org/wiki/Distributed_application) structure that partitions tasks or workloads between the providers of a resource or service, called [servers](https://en.wikipedia.org/wiki/Server_(computing)), and service requesters, called [clients](https://en.wikipedia.org/wiki/Client_(computing)). Often clients and servers communicate over a [computer network](https://en.wikipedia.org/wiki/Computer_network) on separate hardware, but both client and server may reside in the same system. A server [host](https://en.wikipedia.org/wiki/Host_(network)) runs one or more server programs, which share their resources with clients. A client does not share any of its resources, but it requests content or service from a server. Clients, therefore, initiate communication sessions with servers, which await incoming requests. Examples of computer applications that use the client-server model are [Email](https://en.wikipedia.org/wiki/Email), network printing, and the [World Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web).

How the browser interacts with the servers   
There are few steps to follow to interacts with the servers a client.

* User enters the (Uniform Resource Locator) of the website or file. The Browser then requests the (DOMAIN NAME SYSTEM) Server.
* DNS Server lookup for the address of the WEB Server.
* DNS Server responds with the IP address of the WEB Server.
* Browser sends over an HTTP/HTTPS request to WEB Server’s IP (provided by DNS server).
* Server sends over the necessary files of the website.
* Browser then renders the files and the website is displayed. This rendering is done with the help of DOM (Document Object Model) interpreter, CSS interpreter and JS Engine collectively known as the JIT or (Just in Time) Compilers.

Advantages of Client-Server model:

* Centralized system with all data in a single place.
* Cost efficient requires less maintenance cost and Data recovery is possible.
* The capacity of the Client and Servers can be changed separately.

Disadvantages of Client-Server model:

* Clients are prone to viruses, Trojans and worms if present in the Server or uploaded into the Server.
* Server are prone to Denial of Service (DOS) attacks.
* Data packets may be spoofed or modified during transmission.
* Phishing or capturing login credentials or other useful information of the user are common and (Man in the Middle) attacks are common.