**GIT Assignment 3**

1. What is git stockpile, and how does it work? How do you use it?

* There are times when you are working on some say branch(like development) or items which are of lower priority and suppose due to some other work comes in say with high priority like production issue then you need to stop you current low priority development work and jump to work with a few other branch(let's say production issue) as a top priority. The half-done job cannot be commited as it is not in that total phase, you wish to return as well as work with it in a brief time. To solve this concern, git lower deal assistance.
* It waits in a queue of incomplete modifications of branch(development)which you can finish as well as dedicate later on.
* To return to this item of job, all you need to do is call the git stockpile to use command. It brings you your job where you left it to your functioning directory site.
* Stockpile is a protocol of caching system which has been designed to temporarily save data from the some API in order to prevent unnecessary change requests and simplify tasks such as the retrieval of name associations.

1. Describe the various branching strategies.

* Let us first understand branching.
* A branching strategy is mostly used by software development team uses when interacting with a version control system for writing and managing code.
* As per the name it's much clear the branching strategy focuses on how branches are used in the software development process.
* Brach in git means that we can get a space where we do not want to make any changes into master or main branch. instead we get a sub branch and within that sub branch we develop/test our code and once it is successfully we can merge that into main/master branch. It's as simple as that there can be n number of team members using that same source code in master/main. so to avoid conflicts also we use branching so that each team member can develop, build, test within that area and once test is successful can merger into master/main branch.
* A branching strategy helps define how the delivery team functions and how each feature, improvement, or bug fix is handled.
* It provides a roadmap or path from development to production state.
* it help to do parallel development with some planned releases
* It allows to create workflows that lead to structured releases
* Now let us see what all **branches strategies** are there and can be used as below:
* **Git Flow, Github Flow, Trunk base development, Gitlab flow**
* **Git Flow**
* Git Flow is the mostly used branching strategy.
* In this it uses multi-branch approach to manage the source code. This approach has two main branches that live throughout the development lifecycle.
* **master.** The primary main branch where all the production code is stored.
* After the code is in the develop branch it is ready and tested to be released then all the changes are merged to the master branch and used in the deployment.
* **develop** . This is place where software development is done. All the pre-production code is stored here, and the completed code of all the supporting branches is merged directly to the develop branch.
* **Support Branches**
* During the development, developers create different branches for specific conditions or use cases using the develop branch as the base. The following are some branches created like that:
* **feature-** feature branches are used to develop new features and branches off exclusively from the develop branch.
* **hotfix-** This is to deal with production issues where quick fixes are required. They can branch off from the master itself, but need to be merged to both master and develop branches.
* **release-**This branch is used to combine the fixes and improvements and now the is ready for the production release. It will be branched from the develop branch and merged to both develop and master.
* **GitHub Flow**
* This is the simplest approach to manage the development.
* **master.** The primary branch is where code is branched off from and merged to main/master.
* Any new change (like new feature/bug fix) is made in a new branch derived from the master with a descriptive branch name describing the development.
* Commit to the development branch locally and regularly push to the branch.
* Users to create a pull request once the development is done so that the code can be reviewed.
* Once the code is reviewed and approved, it must be tested in the branch before merging to the master branch.
* From here users can immediately deploy the master branch with the new changes.
* **Trunk Based Development**
* The Trunk Based Development strategy involves developers to integrate changes directly into a shared trunk (master) at least once a day.
* This shared trunk is always in a releasable state.
* Developers can pull from this trunk, create a local repository, and then push the code to the shared trunk.
* **GitLab Flow**
* The GitLab strategy combines feature driven development and feature branches with issue tracking. This strategy includes environmental branches such as development, pre-production, and production.
* this provides an isolation of all 3 dev/pre-prod/prod env.
* there are few more like Attribute branching. Launch branching, Job branching etc

1. How do you remove data from Git without being removed from your system?

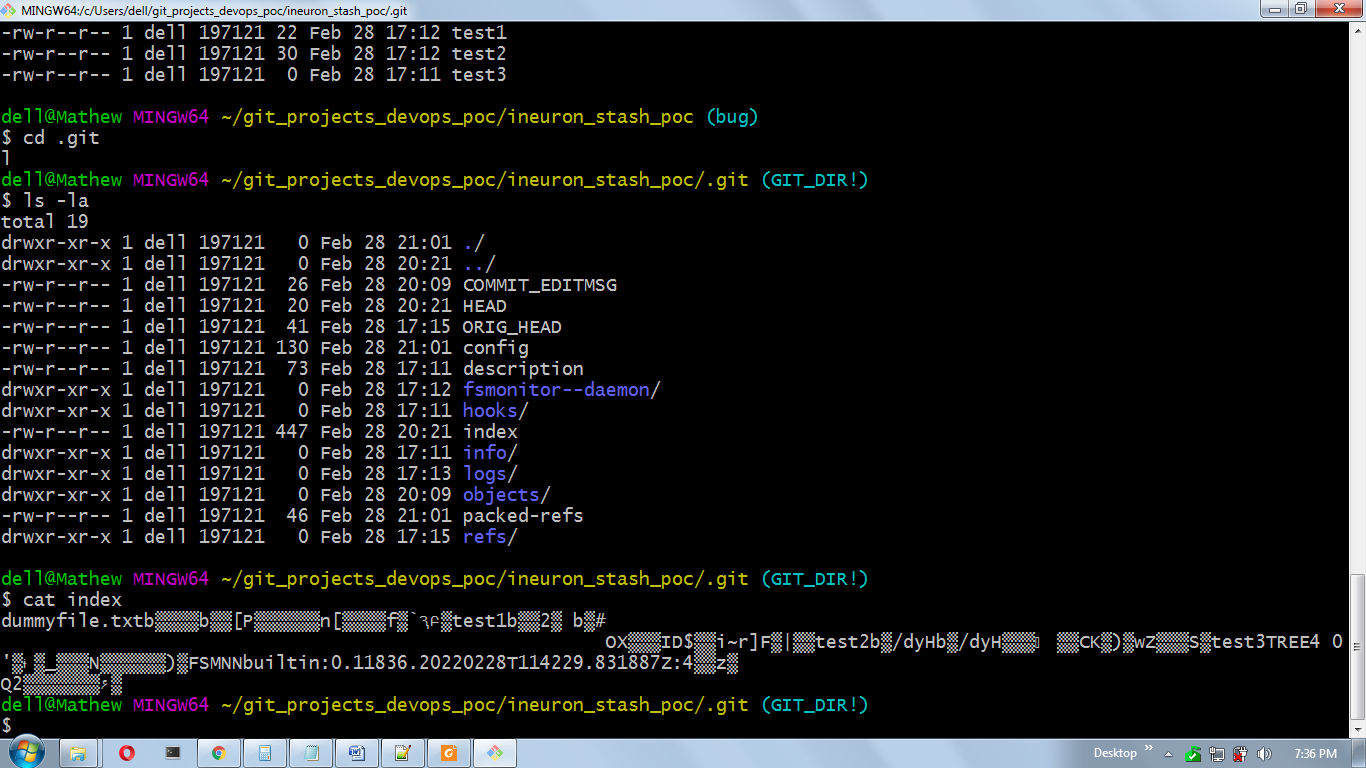
* To remove a file from your Git repository but want to keep that on local system then you can use the –cached flag.
* command **git rm --cached filename** eg. **git rm --cached README.log**
* just fyi for knowledge purpose let's see if we want to ignore some file or folders in git project then we can do that by adding those files/folder in **.gitignore**
* Also some alternatives **are git reset filename or resemble filename >>> >. gitingore& nbsp;**

1. In Git, what does 'index' or 'hosting location' mean?

* The staging area is a file contained in your Git directory, that stores information about

what will go into your next commit. it's technical name in Git is known as the “index”

* Index is a view of your working directory that is ready for commit
* In other words index is a single, large, binary file in repo/.git/index, which lists all files in the current branch, their sha1 checksums, time stamps and the file name
* In simple words the intermediate area where modifications can be formatted or checked before commiting
* Let's see below poc/e.g.



1. What is the difference between 'git remote' and 'git duplicate'?

* remote repository are version of your project that are hosted on internet or network somewhere
* git remote add just creates an entry in your git config that specifies a name for a particular URL
* with git remote we can do three type of operations
* List all remote : **git remote -v**
* adding remote : **git remote add origin\_name https\_link/ssh**
* delete remotes : **git remote remove origin\_name**
* git clone is a git command line utility which is used to target an existing repository and create a clone of the target repository.
* In other words git clone /duplicate creates a brand new git repository by copying an existing one located at the URI you specify.
* **git clone https\_url** (connection-string started with https end with .git)
* **git clone ssh\_link** (starting with git@ end with .git)

1. What is Git Remote's purpose?

* Remote repositories are versions of your project that are hosted on internet/network for. eg github, gitlab
* let us explain with e.g. If you are working in your organization so there will be many team members so all will have one common remote centralized server where they will save their repository, suppose user 1 commits and saves in central repository, now after some days new team member join the team so he will connect to these central remote repository and with pull request will get the code from remote centralized repo to his location machine and will add/change the code or new features and will push that to remote repos(centralized server).. same things repeat when again new user joins. so this is git remote used as centralized server location where all members can pull/push their code as per requirements. Also there can be multiple remote repos(centralized server)

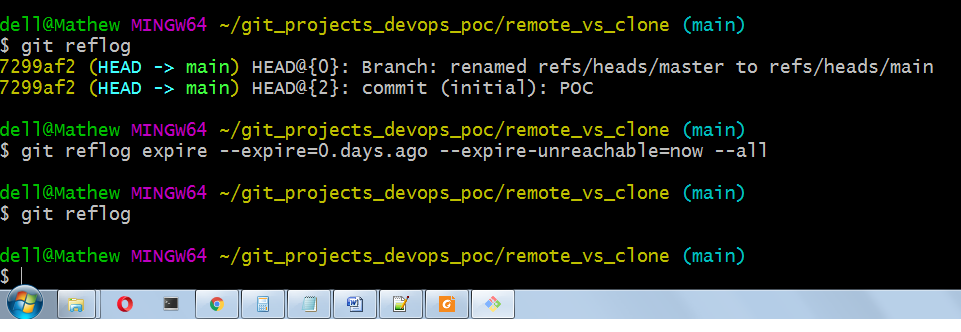
1. How can I clear up a git reflog?

* In git refog means reference log as name suggest it keeps a reference.
* Git keeps track of updates to the tip of branches using a mechanism called reference logs

one way to clear the reflog is we can replcae XYZ with days as below

**git reflog expire --expire=*XYZ*.days.ago --expire-unreachable=now --all**

**git reflog delete** along with parameter can be used for deleting passed reflog entries.



1. How can you distinguish between the git pull and git bring commands?

* The git fetch command communicates with a remote repository and fetches down all the

information that is in that repository that is not in your current one and stores it in your local database.

* The git pull command is basically a combination of the git fetch and git merge commands, where Git will fetch from the remote you specify and then immediately try to merge it into the branch you are on.
* Git pull = git bring + git combine.