**Git diff < commit id >**

Shows the difference and the modifications made.

**Git log -1**

Shows the latest commits, it will show top 1 commit based on the number we give next to the hyphen(-)

**Git log –oneline**

Shows the commits in compressed manner

**Git reflog**

This shows the complete logs of everything in compressed manner

**Git log -–author=”<author name>”**

Displays the logs with author name

**Git log –-since=1day**

Changes done in last one day, will be based on the number the result changes 1 day shows that 1 day record 2 days it will show 2 days records

**Git log –help**

Shows the commands we can execute

######################################################################

**Branches:**

**Master/main:**

Completed code or ready to go live code will be present no alteration or changes must be done, if we do it then it will impact the running code.

**Branch:**

All the editing and or checking of bugs or any other changes patch fixing will be done within the branches, we can give any number of branches and any name of the branch can be given

the code will be present in the main branch and once the editing or the bug fixing is done we can merge the branch to main branch

commands:

**git branch <branch\_name>**

this creates the new branch

**git branch**

displays all the branches present

**git checkout <branch\_name>**

to checkout of the branch

**git checkout -b <branch\_name>**

creates the branch name and moves to that branch

when a file or folder I created and committed in a particular branch then the file or folder will be present only in that branch, it will not be reflected in any other branch, if the commit is not happened then the file or folder will be visible.

To merge the command is : **git merge <branch\_name>**

If the merge has a confict then we need to abort the operation else we cannot go further

The command is : **git merge -–abort**

**Git log –-graph --decorate**

This shows the merging in a graph format

If there are any changes in the commit then the command is :

**Git reset –-soft <commit id first 5 letters>**

**Git restore –-staged <file name1> <file name2>**

A minute changes may occur when we do this which will effect the data

**Git stashing**

Cannot hold the project in the staged area

So, we can stash the project the command is

**Git stash**

To view the list present in stash then

**Git stash list**

To get back the file to working area then the command is

**Git stash apply <stash\_head>**

h/w what is git cherry pick ?

The **git cherry-pick** command in Git copies changes from one branch to another. It's a useful tool for applying specific commits without merging entire branches.