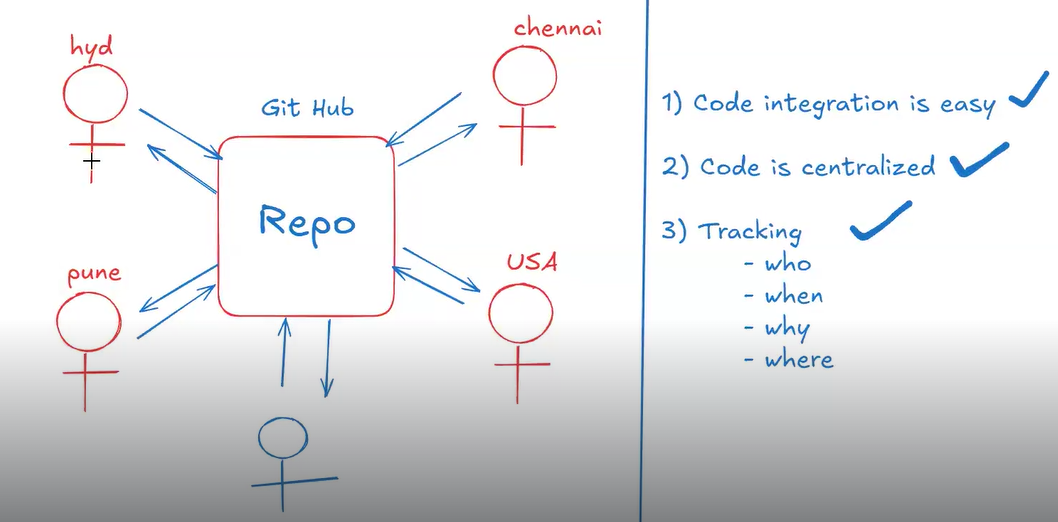
## 03-DevOps - 09-OCT-24

=========

**Git Hub**

=========



- It is version control software

-> Git Hub is a platform which is used to store project related files/code.

-> In git hub, we can create source code repository to store project code.

-> All the developers can connect to project repository to store all the source code

(Code Integration will become easy)

-> Code will be available at one central location (easy access)

-> Git Hub repository will monitor all code changes

- who modified

- when modified

- what modified

- why modified

**Environment Setup**

===================

1) Create account in www.github.com (free of cost)

2) Download & Install git client software

URL : https://git-scm.com/downloads

3) Open git bash tool and configure your name and email using below commands.

$ git config --global user.name "Ashok IT"

$ git config --global user.email "ashokitschool@gmail.com"

* Note: Configuring name and email is just one time process.

=====================================================================

Que - What is Git Hub Repository ?

- Repository is a place where we can store project source code / files.

- For every project one repository will be created

- We can create 2 types of repositories in git hub

1) Public Repo (anybody can see & you choose who can commit)

2) Private Repo (you choose who can see & commit)

## Project git repo url : https://github.com/ashokitschool/sbi\_mobile\_banking.git

- Project team members will connect with git repository using its URL.

**Que - What is the diff between git and git hub**

=====================================================================

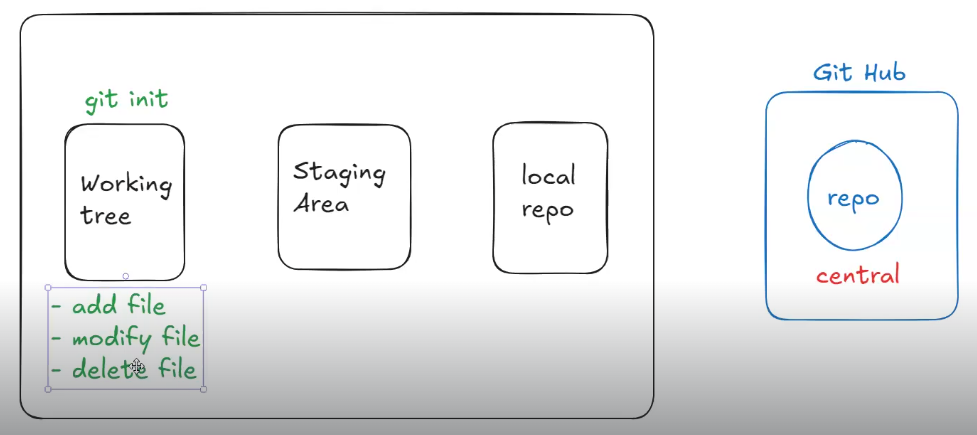
- git is a client s/w which is used to communicate with git hub repositories.

=====================================================================

**Git Architecture**

=====================================================================

1. Working tree



* Working tree represents the place from where we want to perform git operations. Generally project directory will be considerd as working tree.
* Note: To represent/initialize working tree we will execute 'git init' command
* Project folder is called as working tree.
* Directly sharing from working tree to central repository is not possible.

1. Staging area

* Ex. Out of 100 files which files we want to send to the central repository we will add to the staging area are eligible for commit.
* Only which files are added to the staging

3) Local Repo

4) Central Repo (remote)

- all the team members have access for central repository.

- Staging area represents which files are eligible for commit. To add files to staging area we will use 'git add' command.

git add <file-name>

git add \*.txt

git add \*.java

git add .

- Local Repo represents the commits we have done using 'git commit' command.

git commit -m 'commit-msg'

Note: With 'git commit' only staged files will be commited to local repo.

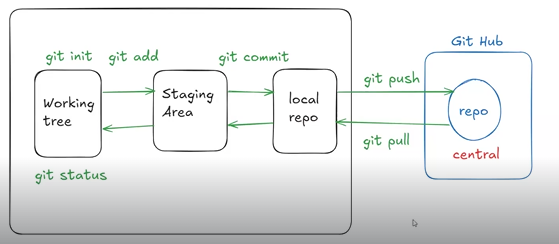
- Central Repo will be available in git hub. All the team members changes will be integrated in central repo.

Note: To send local commits to central repo we will use 'git push' command.

==============

**Git commands**

===============



git config : To configure name & email

git init : To initialize working tree

git status : To check working tree status (staged & un-staged)

git add : To add files to staging area

git commit : To commit files from staging to local repo

git push : To send files from local repo to remote repo

**\* After creating a public repository—**

**Step 1**

* Scroll down to the repository and you will able to see below command that is created by github team.

### …or create a new repository on the command line

echo "# sbi\_app" >> README.md

git init

git add README.md

git commit -m "first commit"

git branch -M main

git remote add origin https://github.com/sachin2460/sbi\_app.git

git push -u origin main

* After running above code github credentials will ask
* To check credentials search in windows – windows credentials

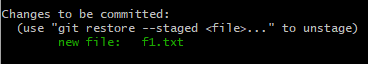
**Step 2**

* Create files using touch command



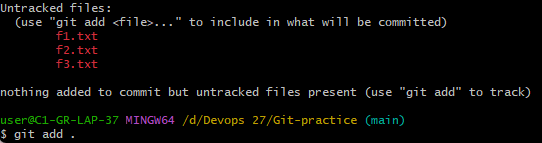
Note : - if you see file in red color that means it is not eligible for commit.

* Add this file into staging area
* git add f1.txt

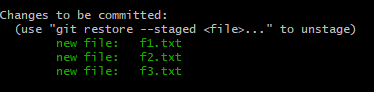


To add all the red files in the staging area

* Command - git add .



Execute Command – git add .

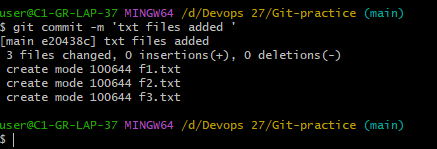


**Step 3**

Que – how to commit the files to the local repository ?

Ans –

Command – git commit –m ‘ txt files added ‘

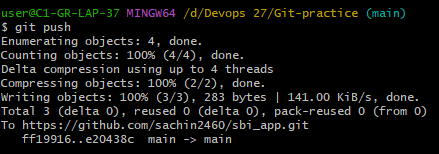


**Step 4**

After committing it will goes to the local repository only not to the central repository.

To send from the local repository to the central repository

**Command – git push**



Check commits – git log it will shows all the commits we have made.

* After git push command files are added to the central repository ( Github ).

Que – How to remove password from the credentials manager ?

Ans – credentials manager -🡪 windows credentials 🡪 remove github account

git restore :

1) To discard changes when the file is in unstaged state

2) To unstage the file when it is added to staging area

git log : To see commit history

**git pull :** To take latest changes from central repo to local repo

* From github to our local repository.

git clone : To clone remote repo to our local machine.

* From github to our machine

git clone <repo-url>

Que –when to go for clone ?

Ans – to download all the central repository

Que – why we use git pull command ?

Ans – to take latest changes from the central repo to the local repo

git rm : To remove files (rm + commit + push)

mvn package – it will create war files

## 04-DevOps - 10-OCT-24

**Que - What is .gitignore file ?**

=====================================================================

- .gitignore is used to exclude files & folders from our commits.

Ex-1: In maven project, we should n't commit "target folder" to git repository hence we can give this info to git using .gitignore file.

Ex-2 : In Angular app, we should n't commit "node\_modules" folder.

=====================================================================

**Git Branches**

=====================================================================

* In a project multiple development teams will work paralelly

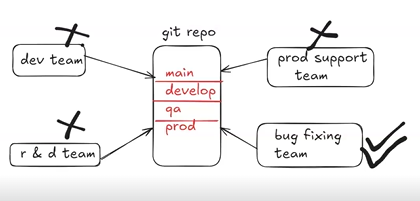
1) Bug fixing team

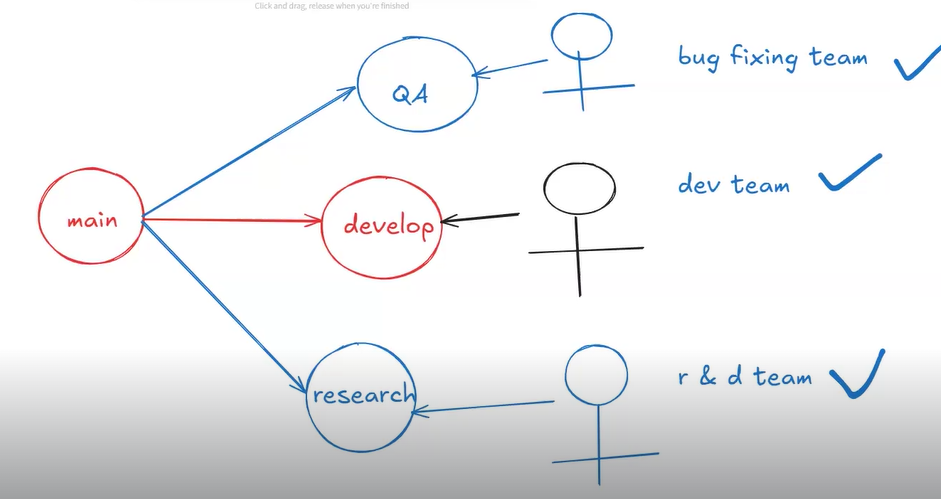
2) Enhancements team

3) change request (CR) team

4) Research & Development team

6) Prod support team





- When multiple teams work on single repository then it will become very difficult to manage the code.

Note: To overcome above problem, we will use "git branches" concept.

- Branches are used to maintain multiple code bases in the single repository.

Note: We can create any no.of branches in single repository.

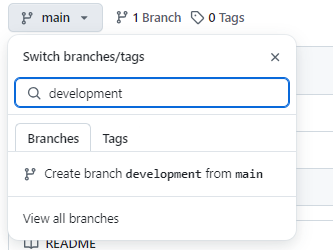
- If we have branches in git repo then multiple teams can work paralelly without effecting other teams code.

- if we made any mistake in project it will affected to the whole project

- we should not directly integrate our work to the main branch.

**Note – project will deliver to the client from main branch only.**

**How to create a branch**

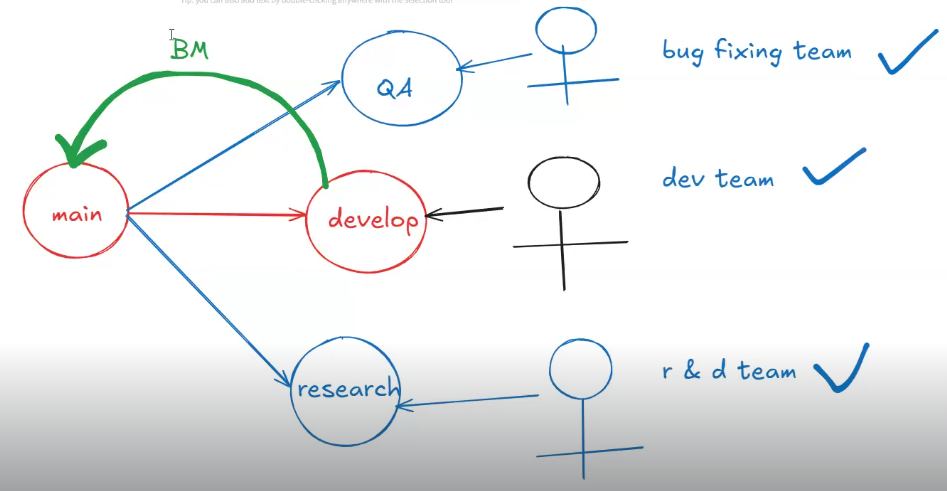
1. Go to the git repository
2. 

* Click on main and create a branch
* After that development branch will created.
* It will copy the main branch code to the development branch.
* I will push my changes to the development branch not to the main branch
* Latest code available to the developemet branch because we are working on the development branch since 10 days
* We need to test development branch not the main branch
* After testing if our code is ready then will merge that code to the main branch
* Final code is available to the main branch
* We can create a branch using git bash

Command - git branch branch-name

Branch Merging

* To merge the changes from one branch to another branch will use branch merging



# clone git repo (default main branch)

git clone https://github.com/ashokitschool/springboot\_register\_login\_security.git

# clone git repo develop branch only

Command -git clone -b develop <repo-url>

* If don’t use –b it will by default use main branch

\*git branch

- used to check branches

# swith branch

* To switch from one branch to another branch

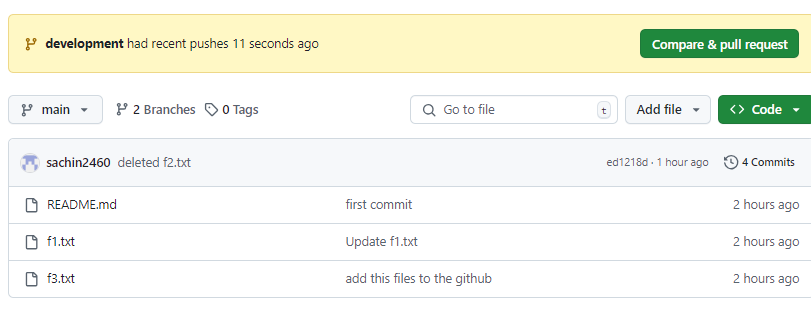
git checkout <branch-name> OR git switch <branch-name>

**git checkout:**

* Has more complex syntax because it handles various tasks. For example:
* To switch to an existing branch: git checkout <branch-name>
* To create and switch to a new branch: git checkout -b <new-branch-name>
* To restore files from another branch or commit:
* git checkout <commit-or-branch> -- <file>

**Git Switch**

* git switch: Has a simpler and more focused syntax for branch switching:
* To switch to an existing branch: git switch <branch-name>
* To create and switch to a new branch: git switch -c <new-branch-name>
* after pushing file from local repository to the central repo github shows



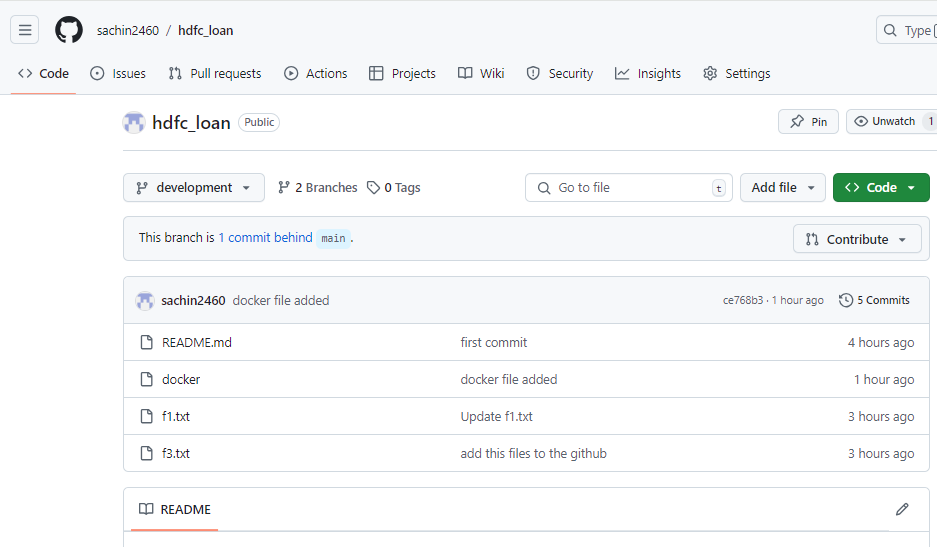
**Que- What is pull request (PR)**

=====================================================================

- It is used to merge changes from one branch to another branch.

Ex : develop -------------> main

Ex. Release PR ( Pull Request )



## 05-DevOps - 16-OCT-24

**Que - what is git stash ?**

=====================================================================

11:00 AM - Manager assigned task "AB-102"

2:00 PM - Completed half of the task (fully not completed)

3:00 PM - Manager called and informed "AB-103" is high priority. First complete 103 and then work on 102.

Note: In this situation we can't delete 102 changes from working tree because 3 hours we spent and we can't commit also because changes are not tested.

git stash : save working tree changes in temp area and make working tree clean.

git stash apply : Get stashed changes back to working tree.

### **Key Commands Recap:**

* git stash – Stashes the current changes.
* git stash list – Lists the stashes.
* git stash apply – Applies the stashed changes.
* git stash drop – Deletes the stash.
* git stash pop – Applies and removes the stash.

=====================================================================

**Que - What is git fetch ?**

=====================================================================

git pull : it will download latest changes from central repo to local repo and merge with working tree also.

Note: with pull command there is a chance of getting conflicts.

git fetch : It will download latest changes from central repo to local repo and it will not merge with working tree.

Note: To merge changes from local repo to working tree we need to execute 'git merge' command.

git pull = git fetch + git merge

What is git revert ( Undo Changes)

=====================================================================

- Revert is used to remove our changes from central repository based on commit id.

git revert <commit-id>

Note: When we execute 'revert' command it will open editor for msg. Save and close it using "esc + :wq".

- After revert command we need to execute "git push" to publish revert operation to central repo.

=====================================================================

Que - What is git conflict ?

=====================================================================

- When we execute "git pull" command there is a chance of getting git conflicts.

Note: If two team members working on same file and same line then we will get conflicts.

* Pull rewquest will give you the conflicts.

- When we merge one branch changes with another branch then there is a chance of getting git conflicts.

- there is no automated solution for conflict we need to resolve manually.

- remove conflict problem from the file

- go to the git status

- after that add file to the stage

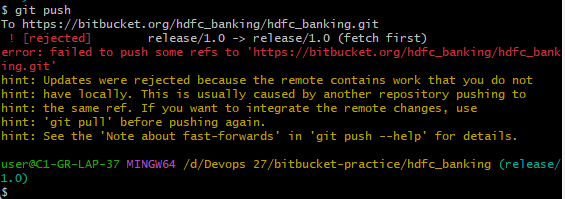
-git commit ‘ resolved issue‘

- git push

Note: When we get conflicts then it is our responsibility to resolve those conflicts and commit to github without conflicts.

git clone <https://ashokitclasses@bitbucket.org/ashokit-ops/hdfc_mobile_banking.git>

Example

* Conflict 1
* When you are working on a file and also your team-mate is also is working on same file if your team-mate has push chenges on same file and also your are pushing changes on same line of file then you get below conflict
* 

Solution

* Run git pull command
* Also oen file and remove Junk code
* Discuss with the team-mate and resolve issue
* Then run git status command
* Add + commit + push

Example 2

* If your main branch is up to date and you just updated the release branch in bitbucket and when you try to create pull request you get an error

Solution

* Check commits
* Get pull and remove junk code

=====================================================================

## 06-DevOps - 17-OCT-24

=====================================================================

What is git conflict ?

=====================================================================

- When we execute "git pull" command there is a chance of getting git conflicts.

Note: If two team members working on same file and same line then we will get conflicts.

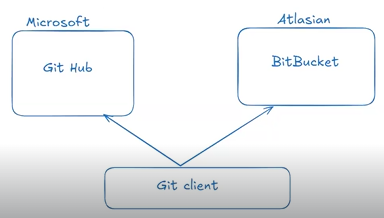
- When we merge one branch changes with another branch then there is a chance of getting git conflicts.

Note: When we get conflicts then it is our responsibility to resolve those conflicts and commit to github without conflicts.

Collaborator

* Is used to commit changes for two person in same repository.
* If you are not a collaborator then you can clone repository

## 05-DevOps - 15-OCT-24



=====================================================================

What is git fork

=====================================================================

- It allows you to create a personal copy of someone else's repository

Note: In general, we will use this forking concept for experiments on open source projects.

* Go to the required person repository and click on fork
* Then it will copy that repository to your account.
* Go to your repositories there your forked repository available
* Copy repository url and go to git bash and copy to the working tree.

=====================================================================

How to remove git local commits

=====================================================================

Que – how to remove commits from local repository ?

Ans -

# remove latest local repo commit and keep changes in working tree

git reset --soft HEAD~1

# remove latest local repo commit and discard changes from working tree

git reset --hard HEAD~1

# remove last 3 local commits

git reset --soft HEAD~3

=====================================================================

## 06-DevOps - 16-OCT-24

=====================================================================

**Que-What is cherry-pick ?**

=====================================================================

In Git, cherry-pick is a command used to apply the changes from a specific commit from one branch to another.

Similar pull request is used to merge changes from one branch to another branch but pull request will pull all commits

But

In cherry-pick only partial commits will be merged.

Ex:

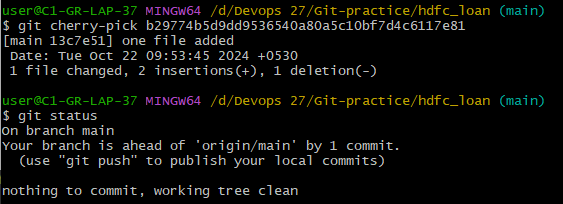
- In develop branch we have done 2 commits. If we create pull request to merge develop branch changes to main branch then it will merge all commits at once (full merge)

- If we don't to perform full merge, we want to merge only particular commit from develop branch to main then use 'git cherry-pick' option

Ex : git cherry-pick <commit-id>

For executing cherry-pick

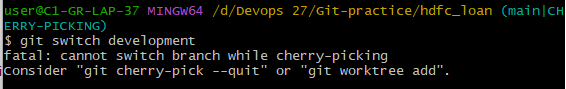
* Execute git log and copy commit id
*  Represent latest commmit
* Then , switch branch where you want to merge the required commit
* Execute cherry-pick + commit id ( Always in hashed code)
* After executing cherry-pick command



* Use git push command for merge the changes to the central repository.

Note - You cannot switch branch while doing cherry-pick

* If you try to switch you will get an error



* To resolve this use below command
* Command – git cherry-pick --quit
* **Abort Cherry-picking**: Command - git cherry-pick --abort

If you want to cancel the cherry-pick operation, you can abort it:

* **Purpose**: Cancels the ongoing cherry-pick operation entirely and resets everything back to the state before the cherry-pick started.
* **Effect**:
* It **removes any changes** that were applied by the cherry-pick.
* Resolves any conflicts by discarding the partially applied cherry-pick.

The working directory is **restored** to the state before you ran git cherry-pick.

- **Use case**: You want to **completely abandon** the cherry-pick, either because of conflicts or you changed your mind about applying the commit.

After running this command, Git will act as if the cherry-pick operation never occurred.

* Command -git cherry-pick –quit
* After running this command, you are no longer in a cherry-pick operation, but the changes will still be in your working directory, ready to be committed or further modified.

**What is git tag ?**

=====================================================================

- Tags in Git provide a convenient way to mark specific commits, usually for releases or deploy important milestone.

- **Lightweight Tag**: Simply creates a pointer to a specific commit.

- command – git tag tag-name

- **Annotated Tag**: Creates a separate Git object that contains the commit hash, along with additional information like a tag message, the name of the person creating the tag, and the date.

- Command – git tag –a tag-name –m ‘message’

# show all tags

git tag

# create a tag

git tag -a <tag-name> -m '<tag-msg>'

# push tags to central repo

git push origin <tag-name>

# push all tags to central repo

git push --tags

# delete tag

git tag -d <tag-name>

# We can checkout particular tag using its name

git checkout <tag-name>

Note: To comeout from tag stat we can use below command

git switch –

**Process of how tag works ?**

* First git pull
* Create one file and push
* After create one tag – git tag –a tagname –m ‘ commit’
* Create another file and push
* Then git push origin tag name
* Git checkout tagname
* Ll

=========================

git merge vs git rebase

=========================

- These commands are used to merge one branch commits to another branch

Ex :

- We have committed changes to develop branch now we want to merge develop branch changes to main branch.

git switch main

git merge develop ( or ) git rebase develop

merge : Creates new commit and combines and preserves commit history.

rebase : Reapplies commits on top of our branch. It will not preserve commit history.

=====================

Real-Time work flow

=====================

- As a devops engineer we are responsible to manage source code repositories in the project.

- Development team will Create JIRA story for git repo creation and will assign to devops team with development manager approval.

- Once request got approved, we should create git repo and we need to share git repo url with developmen team.

- Development team will integrate code in git repo and they will create multiple branches.

Note: DevOps team will decide branching strategy. That means development code should be there in which branch and which branch code will be deployed to production.

- DevOps team will use user permissions for git repo (RBAC)

- Read Permissions

- Read & Write Permissions

Note: Before production deployment we need to go for Code Freeze

(15 days before or 1 month before)