**GIT HUB**

**Source Code Repository Tools (or) Version Control Software’s:**

* **Multiple developers will work for project development**
* **Developers will be working from multiple locations**
* **All developers code should be stored at one place (Code Integration Should Happen)**
* **To integrate all the developers source code at one place we will use Source Code Repository Software’s**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Advantages with Source code repository software:**

**1) All the developers can connect to repository server and can integrate the code**

**2) Code Integration will become easy**

**3) Repository server will provide monitored access**

**- Who**

**- When**

**- Why**

**- What**

**Repository Tools:**

* **SVN (outdated)**
* **Git Hub**
* **Bitbucket**

**Note: GitHub , Bitbucket is more demand.**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Git hub:**

* **It is a cloud platform**
* **It is providing version control facility using git**
* **We can create repositories to store the project source code**
* **Git hub repositories will provide monitored access . we can easily track the code(who, when, what, why)**
* **Repositories mean where we store the project source code.**

**Environment Setup to work with Git Hub:**

**1) Create GitHub account ( www.github.com )**

**2) Download and install Git Client software ( https://git-scm.com/downloads )**

**3) Once installation completed, right click on the mouse, and verify git options display (If git options displaying our git client installation completed successfully)**

**Note: Git client is used to communicate with git hub repositories**

**Version Control system:**

**Whatever commit have done by yesterday and what commit will have done by today , that all commits will maintained in git hub.**

**Whenever we want to go back the previous version, that changes back up also maintained in repositories.**

**Working with GitHub:**

**-> Create Repository in GitHub**

**Note: Repository is used to store project source code. Every Project will have one repository**

**-> When we create a repository, unique URL will be generated with Repository Name (i.e Repo URL)**

**Ex: https://github.com/arun/hdfc\_bank\_app.git**

**-> All the developers will connect to repository using Repository URL**

**-> We can create 2 types of Repositories in Git Hub**

**1) public repository**

**2) private repository**

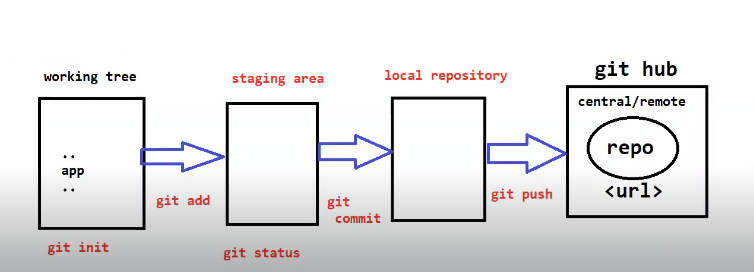
**-> Public Repository means everybody can access but we can choose who can modify our repository**

**-> Private Repository means we will choose who can access and who can modify**

**Repo URL : https://github.com/arun/01-devops-app.git**

**sudo yum install git**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***



**Working Tree: Where we are going to do performing the git operations**

**Staging area: The files which are eligibility for committees**

**Git init: To initialize the Folder as a working tree**

**$ git add: It is used to add the staging area**

**$ git status: It is used to check which files are adding to staging area, which files are not added to staging area.**

**$ git commit : To Commit the changes from Staging to Local Repositories.**

**$ git clone: When you want to download the project for first time from central repo to your machine.**

**$ git pull : It is used to latest updated version from central repo to local.**

**$ git restore : It is used to discard the change in the working directory**

**$ git restore –staged <filename> : it is used to restore the changes from staging area to Working directory**

**$ git rm : It is used to delete file in local repo, central repo.**

**If u delete the file by using gitrm , it will automatically stage to staging area without using git add command**

**$ git remote -v : it used to check the connectivity of which git hub url link from local repo to central repo in**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Git Branches:**

**Branches : It is used to maintained separate code bases for our project.**

**In Git Repositories we can create multiple branches**

**Main**

**Develop**

**QA**

**Release**

**What is Branch in Git Hub?**

**Branches : It is used to maintained separate code bases for our project.**

**When we create Git Repositories by default it will provide the Master Branch**

**In Git Repositories we can create multiple branches**

**Generally, in git repositories we will create branches like below**

**Master (default) , along with that several branches will create**

**Develop**

**Feature**

**QA**

**UAT**

**Release**

**Workflow:**

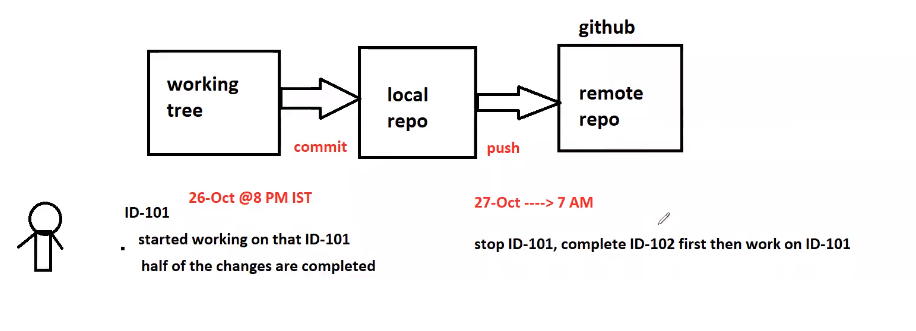
* **Log into Git Repo**
* **Create develop branch form master branch**
* **Clone Develop branch code**

**Note: If we execute the git clone <Repo URL> it clones master branch code by default.**

* **If we want to clone specific branch code , then we should execute below command**

**$ git clone -b <branch name> <repo-url>**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***



**Git stash:**

**Save Working tree changes in temp area and make working tree clean**

**For an example:**

**I am working on the task ID:01**

**Suddenly manager told me that task ID:02**

**But first task partially completed. I will store the first task changes in temp area , I will make my working tree clean then I work on the task2 , once task2 is completed. I will restore the task1 code back to my working tree by using git stash apply**

**Git merge and rebase:**

* **Merge and rebase commands are used to merge changes from one branch to another branch through CLI**

**Git merge:**

* **If you maintain the commit history when we are merging the changes from one branch to another branch.**
* **If you want to preserve the commit history , when merge the changes form one branch to another branch**

**Git rebase:**

**When one branch code to released to the production then we don’t bother about**

**the commit history. In this case we are going to use git rebase**