GIT

As the project or application code keeps on improving and the developers release the new versions.

Example:

Code- version1

Code- version2

These scenarios we would likely to face in real world:

**Application upgrades**

* We might need to replace the existing code and deploy the new version(i.e upgraded version) to improve or add new features to the application

**Application rollback**

* We might also face some issues(bugs) when a new code is deployed. We need to avoid any issues in the application to the end users. In such cases we should revert back the changes(i.e rollback) to its previous version with a minimal downtime.

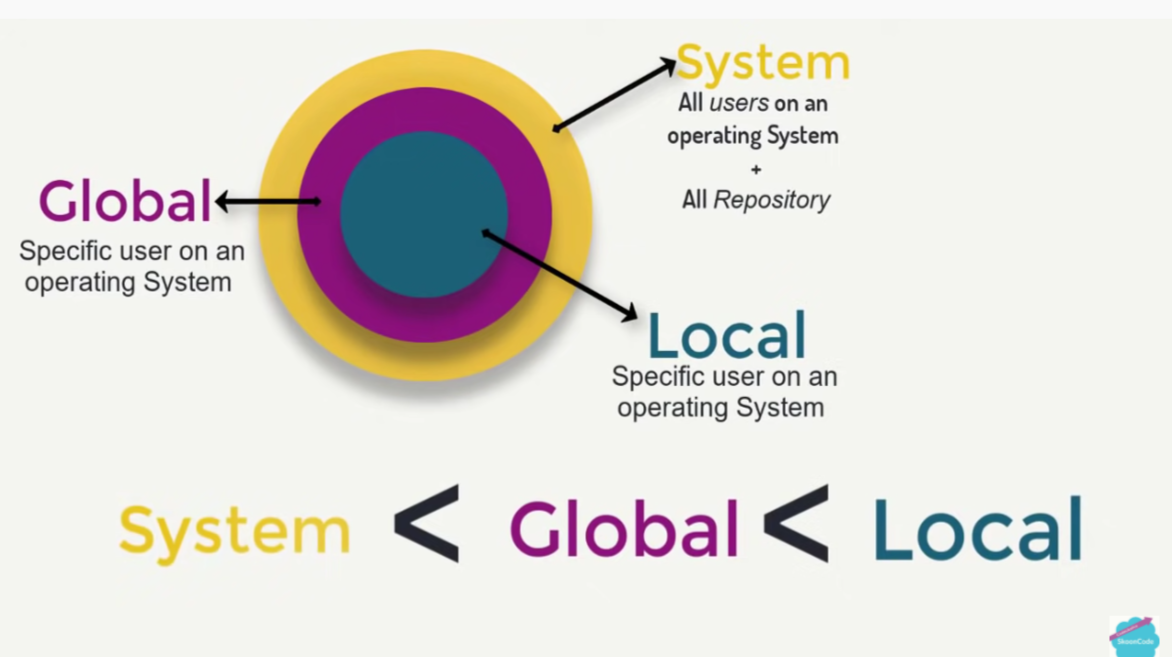
Every developer has their own laptop or workstation where they develop their own code.

**Git Architecture**

* **Working Copy:** The folder created in the developer local machine where the developer is keeping their code.
* **Local Repository:** The folder which is used to store the changes done on the working copy.
* **git init** is the command which creates the local repository.

Git is not the automated version control system. Whenever a developer makes changes to the working copy he should manually run the **git add** command to make the changes included in the Local Repo.

We have to set the configuration on the Git for any user to start working with it.



**System:**

git config --system user.name “name”

git config --system user.email “email address”

**Global:**

git config --global user.name “name”

git config --global user.email “email address”

We have to initialize a repository to set the Git configuration on the Local repository.

**Local:**

git config --local user.name “name”

git config --local user.email “email address”

git config --system --list

* **git status** is the command to check the status of the working copy

1. Checking the status of the files in the staging area(files which were added using git add but not committed yet.
2. Checking the status of untracked files which are not moved to the staging area.

**git status**

* **git add** will not add the changes directly to the working copy instead they will be added to the *staging area*first.

**git add “filename”**

To add all the files in the directory

**git add .**

* **git stage** will move the modified files to the staging area

**git stage “filename”**

* **git commit** command is used to commit the changes from staging area to Local Repo

**git commit -m “message”**

* **git log** will give all the details of the commits
* **git restore** will allow you to unstage a file

**git restore --staged “filename”**

* **git reset** will allow you to move the head and the branch reference to the specific commit or hash.

**git reset -- soft “commit hash”**

* **git stash** is a temporary place or shell where we can move the files from the working directory or staging area. Used in scenarios when you cannot commit the code now as you are working on a new requirement

**git status save “stash-name”** to save the staged files to stash

**git stash list** is used to check the all the available or created stash

**git stash apply “stash index”** to bring the stash back

**git stash drop “stash index”** to drop the stash

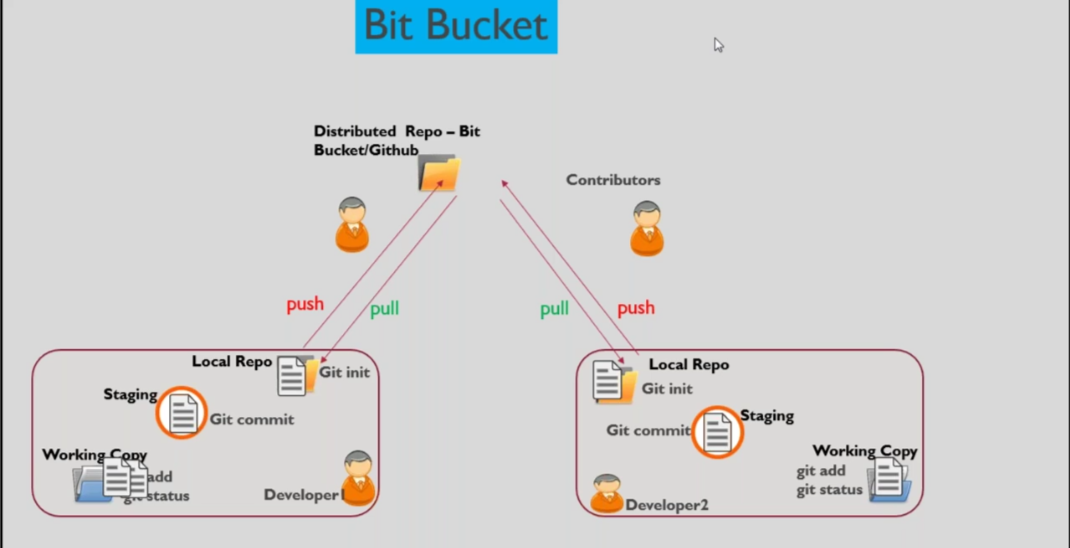
**git stash pop “stash index”**  will apply the stash and drop the stash as well

* **git branch** to check the list of the branches available

**git branch “new branch”** to create a new branch

**git checkout “new branch”** to switch to the new branch

There should be a collaboration between the developers to avoid any code conflicts. To solve this problem **Bitbucket/Github** is used which is a version control system. It is the distributed/remote version control system.



* **git remote**

To add the github or bitbucket remote configuration to the local repository

**git remote add bitbucket “url”**

**git remote show bitbucket**

* **git push** is used to push the committed files from local repository to configured remote repository.

**git push bitbucket master**

Here bitbucket is just the name which is mapped to the remote repo URL, this was added to local previously(git remote add bitbucket “url”)

* **git pull is used to pull the code from remote repository to local repository**

**git pull bitbucket master**

* **git clone** is used to clone the remote repository to local repository.

**git clone -b master “remote repo URL”**

* **git rebase** is a process in which we combine a sequence of commits into a clean rebase, also maintains the clean commit history.