Github Process Flow:-

* Go to **https://github.com/** and create a profile for working with Gitbash and Github
  + once logged in to Github -> create a new repository and name the same (Note:- when creating a repository if we are not checking the "init with readme" option, then it needs to be done manually by following the steps out in the repository page. since, unless it is not done it is not possible to work with Git and Github.
  + In the meantime it is to be ensured that Git software is installed properly, since with Gitbash only all commands can be executed to work with Github and the link for download is, **https://git-scm.com/downloads**

Let's start the work by starting with the steps below:-

* + Go to the working folder or to the folder where the files are to be uploaded into the Github repository
  + Right click and click **"Gitbash here"** command from the pop-down list
  + First type GIT INIT (to initialize the repository for tracking from that "particular" folder)
  + Then now from Gitbash, give **git status** command to check what are all the files to be tracked or in other words to be added to the repository and for this from Gitbash, we need to give **GIT ADD <FILENAME.TXT>** and this should be repeated for all files to be added individually or in short form we can give the command, **GIT ADD .** (to add all files in one shot.)
  + Now to establish remote connection with the Github, add the repository path for the same for example, if the repository name is **devopsdemo** then the remote path is, **git remote add origin https://github.com/jagdevops/devopsdemo.git** and here origin is equated with the remote destination hence in future origin alone is enough to be referred.
  + Once the remote connection with the repository is established then we need to move the files from the directory (if files are elsewhere which is not **PWD**), to the working directory and then the movement will take place with the help of PUSH command, for example, **GIT PUSH ORIGIN MASTER** (here origin is the remote path pointing to the Github repository and MASTER is the default branch)

In case, if not configured with the "username / email" of the Github, then the **GIT PUSH** command will thrown an exception and immediately we need to configure the **"Valid Credentials"** for the Github to make sure that the **GIT PUSH** command is successful and the commands for configuration is as follows,

- **GIT CONFIG --GLOBAL USER.EMAIL "devopsjag@gmail.com"**

- **GIT CONFIG --GLOBAL USER.NAME "jagdevops"**

once the configuration of credentials done, then once again **GIT PUSH** command to be executed and now since it is configured with credential to the repository, files will be added to the repository (**we need to refresh if auto refresh is not happening**)

* Likewise, if any new files created or to be added, once created or once identified to be added, then from Gitbash we need to give **GIT STATUS** command which will tell what are the files pending or new to be tracked in other words to be added and then, the following commands to be repeated to accomplish the task and they are,

- **GIT ADD .**

- **GIT COMMIT -M "<custom message>"**

- **GIT PUSH ORIGIN MASTER**

(once **GIT COMMIT** is done and **GIT PUSH** command executed, visit Github and refresh to see the new files that are added as part of last **COMMIT** and **PUSH** commands)

Now let us see how a **BRANCH** can be created apart from the **MASTER BRANCH** and for that we need to use either one of the 2 ways, and they are,

- from the Gitbash, we need to execute, **GIT BRANCH BRANCHNAME** or

- from the Github and via the **BRANCH** dropdown, we can give simply click and add the **BRANCH** and name the same.

Now, to find how many **BRANCH(s)** available we have to execute the command,

- **GIT BRANCH** (which will list all BRANCH(s) in the Github repository)

And like how different set of files can be created under **MASTER BRANCH** and here in individual **BRANCH** too, we can created files or identify files to be added and can be moved into the respective **BRANCH**, and for this we need to switch it to the respective BRANCH,

for example,

- **GIT CHECKOUT BRANCH BRANCHNAME** (also it can be given as **GIT CHECKOUT -B BRANCHNAME**) (AND after this from MASTER it would be switched to the respective BRANCH and all transactions will be tracked under BRANCH ONLY till we move out of BRANCH and do a **COMMIT**)

- **GIT ADD .**

- **GIT COMMIT -M "<custom message>"**

- **GIT PUSH ORIGIN BRANCHNAME**

(once **GIT COMMIT** is done and **GIT PUSH** command executed, visit Github and refresh to see the new files that are added as part of last **COMMIT** and **PUSH** commands in the respective **BRANCH**)

And also, **one catch here** in other words what is to be noted is, when we checkout to **MASTER** we will not see the files added when we added inside the BRANCH and for the files to getting reflected into **MASTER** as well we need to use the command **MERGE**

for example,

- **GIT MERGE MASTER BRANCH NAME**

- **GIT COMMIT -M "<custom message>"**

After the merge, to check the files if they are added to the **MASTER** then checkout to **MASTER** and then refresh, for example,

**- GIT CHECKOUT MASTER**

To **DELETE a BRANCH** it should be done in 2 ways,

- one is to **delete it locally and**

**- another one is remotely** (which is nothing but in the Github repository)

Now to **DELETE**, the command to be executed as follows,

**- git branch -d <branch-name>** (this will delete the branch locally)

And now, to delete it remotely, the command to be executed is,

**- git push origin :<branch-name>**

and for confirming the changes, we need to commit

- **GIT COMMIT -M "<custom message>"**