GIT AND GITHUB

Why we use Git?

Git is defined as Version control System and it used for

1. Tracking changes in computer files.
2. Code management in software development

Its main aim is for:

1. Tracking History
2. Working Together.

Version Controls like Git are classified into two main categories:

1. Centralized
   1. Subversion
   2. Team Foundation
2. Distributed
   1. Git
      1. Free
      2. Open Source
      3. Super-Fast
      4. Scalable

Git has Four Stages:

1. Working Directory
2. Staging Area
3. Commit History
4. Server Repository

We will use comment line interface (**GIT** WITH **GIT-BUSH**).

Git Configuration:

1. Name
2. Email
3. Editor

Level of access:

1. System – all users
2. Global- All repositories in current users
3. Local –current repositories.

By default windows Git bush is run of (vim editor) but you can run (Vs Code)

How Git works:

1. Work directory (work space)
2. Staging area(rest area)
3. Commit history( tracking files)
4. Server repository (remote repository )

The first three one are called = .Git repository or local repository.

Git is a repository service which provides Graphical UI for users and helps team to work together and collaborate easily.

Git configuration:

1. Name
2. Email
3. Editor

Git configuraton levels:

1. System- All users
2. Global-all repositories in current users
3. Local –current repository

Other software working like Git:

1. Bitbucket
2. AWS CodeCommit

**COMMENDS**

|  |  |  |
| --- | --- | --- |
| **No** | **COMMENDS** | **USED FOR** |
|  | LS or LS –a | Is use to show a list of things it is for Linux not for windows. Means it came from there. |
|  | Clear or ctrl +l | To clear the written lines of commends in the window. |
|  | Git config --global user.name “Himd wise” | This code allows you to make a name for your git configuration. |
|  | Git config --global user.email [himadwise@gamil.com](mailto:himadwise@gamil.com) | This code allows you to make an email for your git configuration. |
|  | Git config --global core.editor “code –wait” | This code allows you to make a code editor for your git configuration. And then link to Vs code instead of vim editor |
|  | Git config –global -e  Also {code .} | Allows to open the Vs code after configuration and fallowing the 3 steps above. |
|  | Git –help or Git | Which gives all code used in Git that can help. |
|  | Git config user.name | Code for detect whether you have a name or not |
|  | Git config user.email | Code for detect whether you have a email or not |
|  | Git config –h | Means get help in the Git-bush |
|  | Git config –l /--list | To get all name, email, and code editor |
|  | Cd d: or c: | change directory |
|  | Pwd | present working directory |
|  | Mkdir himadsoft –git | Make directory |
|  | Cd himadsoft-git | To get directory. |
|  | Ls –a | Used to know the hidden files that can’t be changed. |
|  | Git status | Show the situation of the Git |
|  | Git init | Starting initialization of Git |
|  | {Open .git/} {Start . .Git} for windows | To open the hidden .git file in explore |
|  | Git add / Git add . (name) | To add files or move to staging area. |
|  | Git rm – cached (name) | To re back from the staging after adding it on the staging area . |
|  | Git commit –m | To files now to Commit area while (m) stands for meaningful message. |
|  | Git log  Git log –oneline | To show files saved in commit area with hash code  To show in one line of abbreviated from the code above. |
|  | Git show (hash code) | To show particular files that have update into the commit or its self. |
|  | Git diff | To show the difference after update and project the new ones. |
|  | Cat (file name) | Is used to read what is inside your file. |
|  | Touch | Allows you to create a zero file with any extension |
|  | Vi (file name) | Is another code editor you can use to write files.  Press (i) to start coding  Press (esc) and then write (: wq/wq!) to save and quit. |
|  | Git cd .. | Return back to the former directory . |
|  | Git add –A | This flag allows to add more than one file into staging area. |
|  | git remote add origin (https://github.com/Himadwise/himadsoft-git.git  ) | To link your local repository to remote repository |
|  | Git push –u origin master (branch or other branches like main) | To upload files automatically into remoter repository. |
|  | Git fetch | Allows to show if changes occurred or not in GitH |
|  | Git pull | Read files form Rep after fetching. |
|  | Ctrl + ` | To open your folder after you inserted into vs code in the terminal. |
|  | Git checkout (filename)/ Git checkout HEAD (filename) | Removes last entrance of the file. |
|  | Git show HEAD | Shows last you did. |
|  | Git reset (filename) or all (.) | To remove files you put into staging area. |
|  | Git checkout HEAD . | To remove both working and staging area. |
|  | Git reset (HEAD=sha-id) | To remove files from commit to put into working directory (by coping the before commit and passing it) |
|  | Git reset –hard (sha-id) | Removes both commit and working area by passing the sha-id of the previous commit. |
|  | Git commit –am “massage” | Simple way to use both (git add and commit) |
|  | Git branch | Is used to know which branch you are in. |
|  | Git branch -v | To know the last commit of your branch |
|  | Git branch (branchName) | Allows you to create a new branch |
|  | Git checkout (branchName) | To switch the current branch to newly created branch |
|  | Git merge (branchName) | Is used to put files in another branch to the current branch. |
|  | Git branch –d (branchName) | Automatically deletes the branch passed. |
|  | Git checkout –b (branchName) | This line creates the branch and switches to the current branch. |
|  | Git clone (the Remote) .git (project Folder Name ) | Copy the project that has that remote . |
|  | Git config credential.helper cache | Save my access token |
|  | Git remote add (name) origin https | To always have update files without clicking fetch in GUI and go to your files and pull. |
|  | Git pull (name-like origin) (branch) | Allows you to make the files you cloned and changed to be up to date always by using this simple code. |
|  | SSh-keygen | Creates for you SSH key. |
|  | Eval “$(ssh-agent)” | Connects both the key in your laptop and that in the remote server. |
|  | Ssh-add | Allows to add the key private in your project. |
|  |  |  |
|  |  |  |