Git & GitHub

Git is a version control system that lets you manage and keep track of your source code history. **GitHub** is a cloud-based hosting service that lets you manage **Git** repositories. If you have open-source projects that use **Git**, then **GitHub** is designed to help you better manage them.

Basic GIT Commands

Here are some basic GIT commands you need to know:

$ git init

- **git init will create a new local GIT repository. The following Git command will create a repository in the current directory:**

$ git init [project name]

- **Alternatively, you can create a repository within a new directory by specifying the project name:**

```
$ git clone username@host:/path/to/repository
```

- git clone is used to copy a repository. If the repository lies on a remote server, use:

  - Conversely, run the following basic command to copy a local repository:

```
$ git clone /path/to/repository
```

  - git add is used to add files to the staging area. For example, the basic Git following command will index the temp.txt file:

```
$ git add <temp.txt>
```

- git commit will create a snapshot of the changes and save it to the git directory.

```
$ git commit –m “Message to go with the commit here”
```

- Note that any committed changes won’t make their way to the remote repository.

- git config can be used to set user-specific configuration values like email, username, file format, and so on. To illustrate, the command for setting up an email will look like this:

```
$ git config --global user.email youremail@example.com
```

- The –global flag tells GIT that you’re going to use that email for all local repositories. If you want to use different emails for different repositories, use the command below:

```
$ git config --local user.email youremail@example.com
```

- git status displays the list of changed files together with the files that are yet to be staged or committed.

```
$ git status
```

- git push is used to send local commits to the master branch of the remote repository. Here's the basic code structure:

```
$ git push origin <master>
```

- Replace <master> with the branch where you want to push your changes when you're not intending to push to the master branch.

- git checkout creates branches and helps you to navigate between them. For example, the following basic command creates a new branch and automatically switches you to it:

```
$ git checkout -b <branch-name>
```

- To switch from one branch to another, simply use:

```
$ git checkout <branch-name>
```

- git remote lets you view all remote repositories. The following command will list all connections along with their URLs:

```
$ git remote –v
```

- To connect the local repository to a remote server, use the command below:

```
$ git remote add origin <host-or-remoteURL>
```

- Meanwhile, the following command will delete a connection to a specified remote repository:

```
$ git remote rm <name-of-the-repository>
```

- git branch will list, create, or delete branches. For instance, if you want to list all the branches present in the repository, the command should look like this:

```
$ git branch
```

- If you want to delete a branch, use:

```
$ git branch –d <branch-name>
```

- git pull merges all the changes present in the remote repository to the local working directory.

```
$ git pull
```

- git merge is used to merge a branch into the active one.

```
$ git merge <branch-name>
```