BASIC GIT COMMANDS:-

A Git project consists of three major sections: **the working directory, the staging area,** and **the git directory.**

The working directory is where you add, delete, and edit files. Then, the changes are indexed in the staging area. After you commit your modifications, a snapshot of the changes will be saved in the directory.

The following git commands are:

*Git Configuration:This command sets the author name and email address respectively to be used with your commits.

```
git config –global user.name “lalitha”
```

```
git config –global user.email “lalitha45@gmail.com”
```

*Git Initializing and Cloning:This command is used to start a new repository.

While, Cloning is used to obtain a repository from an existing URL.

```
git init [repository name]
```

```
git clone [url]
```

*Git add: This command adds a file to the staging area.

```
git add [file]
```

This command adds one or more to the staging area.

```
git add *
```

*Git Commit: This command records or snapshots the file permanently in the version history.

```
git commit -m “message”
```

This command commits any files you’ve added with the git add command and also commits any files you’ve changed.

```
git commit -a
```

*Git Diff: This command shows the file differences which are not yet staged.

```
git diff
```

This command shows the differences between the files in the staging area and the latest version present.

```
git diff –staged
```

This command shows the differences between the two branches mentioned.

```
git diff [first branch] [second branch]
```

*Git Reset: This command unstages the file, but it preserves the file contents.

```
git reset [file]
```

This command undoes all the commits after the specified commit and preserves the changes locally.

```
git reset [commit]
```

This command discards all history and goes back to the specified commit.

```
git reset –hard [commit]
```

*Git Status:This command lists all the files that have to be committed.

```
git status
```

*Git Remove: This command deletes the file from your working directory and stages the deletion.

```
git rm [file]
```

*Git Log: This command is used to list the version history for the current branch.

```
git log
```

This command lists version history for a file, including the renaming of files also.

```
git log –follow[file]
```

*Git Show: This command shows the metadata and content changes of the specified commit.

```
git show [commit]
```

*Git Tag: This command is used to give tags to the specified commit.

```
git tag [commitID]
```

*Git Branch: This command lists all the local branches in the current repository.

```
git branch
```

This command creates a new branch.

```
git branch [branch name]
```

This command deletes the feature branch.

```
git branch -d [branch name]
```

*Git Checkout: This command is used to switch from one branch to another.

```
git checkout [branch name]
```

This command creates a new branch and also switches to it.

```
git checkout -b [branch name]
```

*Git Merge: This command merges the specified branch's history into the current branch.

```
git merge [branch name]
```

*Git Remote: This command is used to connect your local repository to the remote server.

Usage: `git remote add [variable name] [Remote Server Link]`

*Git Push: This command sends the committed changes of master branch to your remote repository.

```
git push [variable name] master
```

This command sends the branch commits to your remote repository.

```
git push [variable name] [branch]
```

This command pushes all branches to your remote repository.

```
git push –all [variable name]
```

This command deletes a branch on your remote repository.

```
git push [variable name] :[branch name]
```

*Git Pull: This command fetches and merges changes on the remote server to your working directory.

```
git pull [Repository Link]
```

*Git Stash: This command temporarily stores all the modified tracked files.

```
git stash save
```

This command restores the most recently stashed files.

```
git stash pop
```

This command lists all stashed changesets.

```
git stash list
```

This command discards the most recently stashed changeset.

```
git stash drop
```

Usage: `git init [repository name]`

```
git init [repository name]
```