Git:

1.version control system

2.keep track of code changes

3.used to collaborate on code

Git install:

You can download Git for free from the following website: https://www.git-scm.com/

Configuring git for the first time:

Git config –global user.name “vallapuarjun”

Git config –global user.email “vallapuarjun0011@gmail.com”

```
VALLAPU ARJUN@ARJUN-PC MINGW64 ~ (main)
$ git config --global user.name "VallapuArjun"

VALLAPU ARJUN@ARJUN-PC MINGW64 ~ (main)
$ git config --global user.email "vallapuarjun0011@gmail.com"

VALLAPU ARJUN@ARJUN-PC MINGW64 ~ (main)
$ git config --global --list
user.email=vallapuarjun0011@gmail.com
user.name=VallapuArjun
```

Steps to follow:

Creating git folder

```
VALLAPU ARJUN@ARJUN-PC MINGW64 ~ (main)
$ mkdir gitproject

VALLAPU ARJUN@ARJUN-PC MINGW64 ~ (main)
$ cd gitproject
```

mkdir **make**s a **new directory**.

cd **changes** the **current working directory**.

Initializing the git:

1.git init-command:

```
VALLAPU ARJUN@ARJUN-PC MINGW64 ~/gitproject (main)
$ git init
Initialized empty Git repository in C:/Users/VALLAPU ARJUN/gitproject/.git/
```

Creates a hidden folder to keep track of changes

2.staging the files and adding the files to git repo:

ls will **list** the files in the directory. We can see that index.html is there.

```
VALLAPU ARJUN@ARJUN-PC MINGW64 ~/gitproject (main)
$ vi index.html

VALLAPU ARJUN@ARJUN-PC MINGW64 ~/gitproject (main)
$ ls
index.html

VALLAPU ARJUN@ARJUN-PC MINGW64 ~/gitproject (main)
$ git status
On branch main

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        index.html

nothing added to commit but untracked files present (use "git add" to track)
```

Staged files are files that are ready to be committed to the repository you are working on. When you first add files to an empty repository, they are all untracked. To get Git to track them, you need to stage them, or add them to the staging environment

git add<filename>

3.staging all file in a folder:

Lets add an another style.css file and update the index.html file with stylesheet

git add –all

Or

git add -A

```
VALLAPU ARJUN@ARJUN-PC MINGW64 ~/gitproject (main)
$ git add -A
warning: in the working copy of 'index.html', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'style.css', LF will be replaced by CRLF the next time Git touches it

VALLAPU ARJUN@ARJUN-PC MINGW64 ~/gitproject (main)
$ git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   index.html
        new file:   style.css
```

4.making a commit for staged files:

Adding commits keep track of our progress and changes as we work. Git considers each commit change point or "save point". It is a point in the project you can go back to if you find a bug, or want to make a change. When we commit, we should always include a message

git commit -m “message”

```
VALLAPU ARJUN@ARJUN-PC MINGW64 ~/gitproject (main)
$ git commit -m "my first helloworld program"
[main (root-commit) ced4364] my first helloworld program
 2 files changed, 20 insertions(+)
 create mode 100644 index.html
 create mode 100644 style.css
```

5.git commit without stage:

we will use the --short option to see the changes in a more compact way:

```
VALLAPU ARJUN@ARJUN-PC MINGW64 ~/gitproject (main)
$ git status --short
 M index.html
```

Note: Short status flags are:

- ?? - Untracked files
- A - Files added to stage
- M - Modified files
- D - Deleted files

Sometimes, when you make small changes, using the staging environment seems like a waste of time. It is possible to commit changes directly, skipping the staging environment. The -a option will automatically stage every changed, already tracked file.

git commit -a -m “message”

```
VALLAPU ARJUN@ARJUN-PC MINGW64 ~/gitproject (main)
$ git commit -a -m "updated index.html with a new line"
warning: in the working copy of 'index.html', LF will be replaced by CRLF the next time Git touches it
[main 037b1bc] updated index.html with a new line
 1 file changed, 1 insertion(+), 1 deletion(-)
```

6.status of files:

```
VALLAPU ARJUN@ARJUN-PC MINGW64 ~/gitproject (main)
$ git status
On branch main
nothing to commit, working tree clean
```

7.log of a file:

To view the history of commits for a repository, you can use the log command:

```
VALLAPU ARJUN@ARJUN-PC MINGW64 ~/gitproject (main)
$ git log
commit 037b1bc4b2854ca63be248633392f3e2307b5853 (HEAD -> main)
Author: VallapuArjun <vallapuarjun0011@gmail.com>
Date:   Tue Oct 8 19:18:51 2024 +0530

    updated index.html with a new line

commit ced4364f152fd3ce97e7eebdf79304ff64d21e61
Author: VallapuArjun <vallapuarjun0011@gmail.com>
Date:   Tue Oct 8 19:11:08 2024 +0530

    my first helloworld program
```

Git branching:

1.making a new branch

Git branch <name of branch>

```
VALLAPU ARJUN@ARJUN-PC MINGW64 ~/gitproject (main)
$ git branch hello-world
```

2.checking all branches:

```
VALLAPU ARJUN@ARJUN-PC MINGW64 ~/gitproject (main)
$ git branch
  hello-world
* main
```

3.switching to other branches:

Git checkout <branch name>

```
VALLAPU ARJUN@ARJUN-PC MINGW64 ~/gitproject (main)
$ git checkout hello-world
Switched to branch 'hello-world'
```

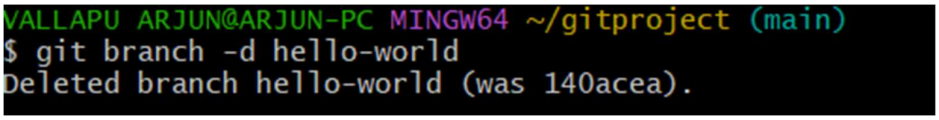
4.merging two branches:

Git merge <branch name>

```
VALLAPU ARJUN@ARJUN-PC MINGW64 ~/gitproject (hello-world)
$ git checkout main
Switched to branch 'main'

VALLAPU ARJUN@ARJUN-PC MINGW64 ~/gitproject (main)
$ git merge hello-world
Updating 037b1bc..140acea
Fast-forward
 index.html | 6 ++++++
 1 file changed, 6 insertions(+)
```

5.deleting a branch:



git branch -d <branch name>