**Name:** Rohit Harawade

**Roll Number:** A09

**Batch:** A1

**Id:** TU4F2122009

**Experiment - 3**

# Aim: To perform version control on websites/software using GIT.

**Requirement**: PC, GitHub account.

**Lab Objective:**

To be aware of different Version Control tools like GIT, CVS or Mercurial.

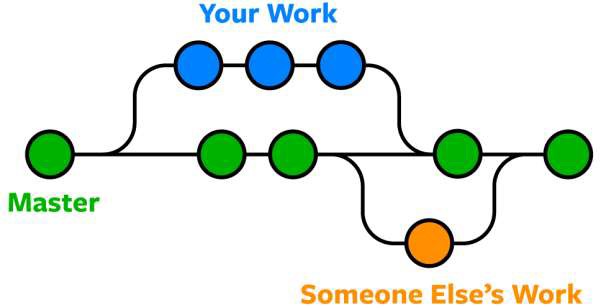
**Theory:**

Git is a distributed version control system. This means that every copy of a Git repository contains the entire history of the project. This makes it easy for multiple people to work on the same project simultaneously, and to collaborate on changes.

Git stores changes to files in a series of commits. Each commit contains a snapshot of the files at a particular point in time, along with a description of the changes that were made. This allows you to track the history of your project and to revert to previous versions if necessary.

Git also supports branches. A branch is a copy of the main line of development. This allows you to work on different features or bug fixes without affecting the main code. When you are finished working on a branch, you can merge it back into the main line of development.

Git is a powerful tool that can be used to manage the development of any project. It is easy to learn and use, and it is free and open source.



**History:**

Git development was started by Torvalds in April 2005 when the proprietary source-control management (SCM) system used for Linux kernel development since 2002, BitKeeper, revoked its free license for Linux development. Torvalds wanted a distributed system that he could use like BitKeeper, but none of the available free systems met his needs. He cited an example of a source-control management system needing 30 seconds to apply a patch and update all associated metadata, and noted that this would not scale to the needs of Linux kernel development, where synchronizing with fellow maintainers could require 250 such actions at once. For his design criterion, he specified that patching should take no more than three seconds, and added three more goals:

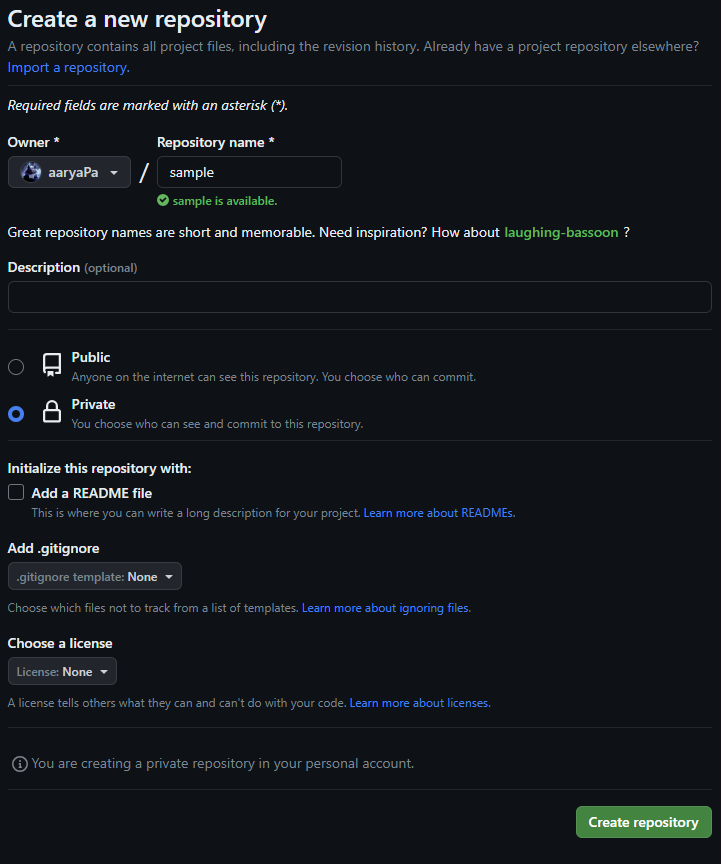
* Take the Concurrent Versions System (CVS) as an example of what not to do; if in doubt, make the exact opposite decision.
* Include very strong safeguards against corruption, either accidental or malicious.

These criteria eliminated every version-control system in use at the time, so immediately after the 2.6.12-rc2 Linux kernel development release, Torvalds set out to write his own.

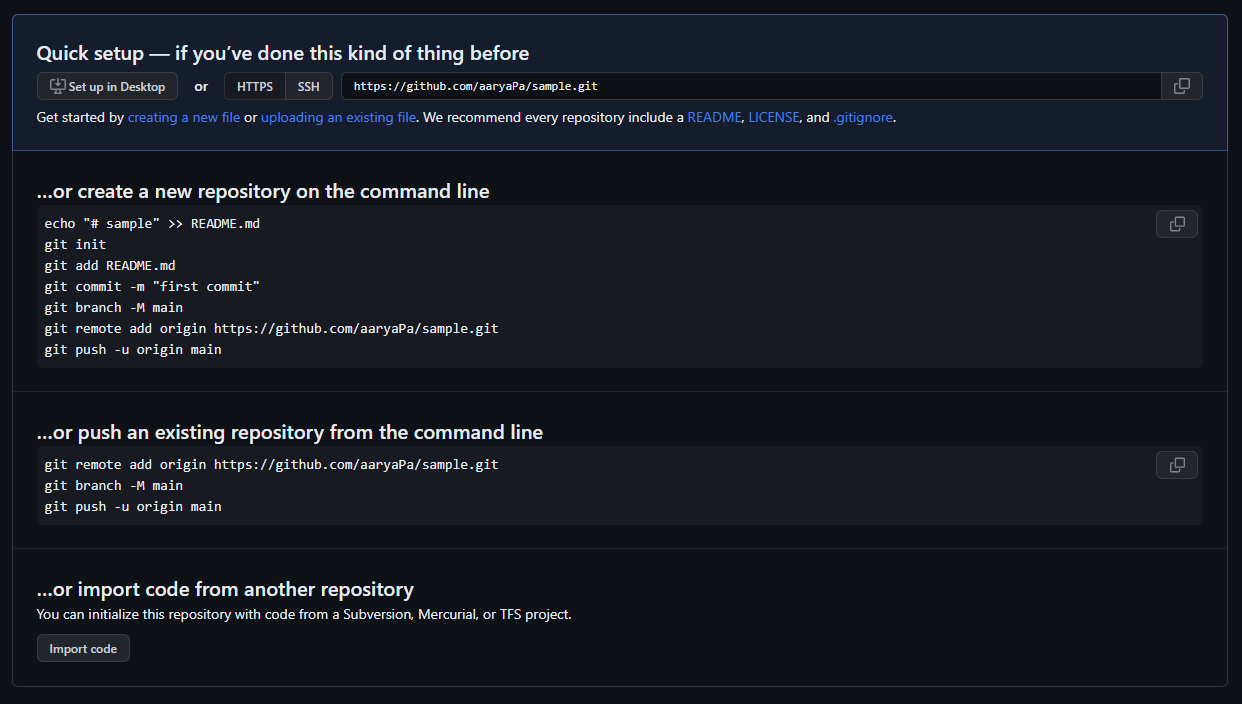
The development of Git began on 3 April 2005. Torvalds announced the project on 6 April and became self-hosting the next day. The first merge of multiple branches took place on 18 April. Torvalds achieved his performance goals; on 29 April, the nascent Git was benchmarked recording patches to the Linux kernel tree at a rate of 6.7 patches per second. On 16 June, Git managed the kernel 2.6.12 release.

**Steps:**

1. First create a repository through Github. Decide the name, the access and the license for the repository.



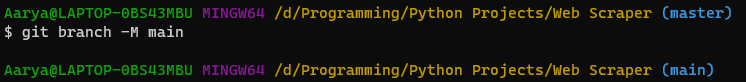
1. A setup window for the file will be opened.



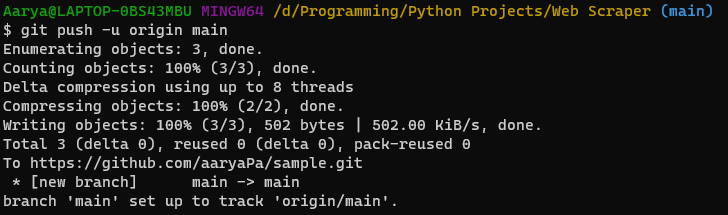
1. As we already have an existing repository, we will push this repository to Github from the command line.
2. We will first add a remote origin to the existing repository.



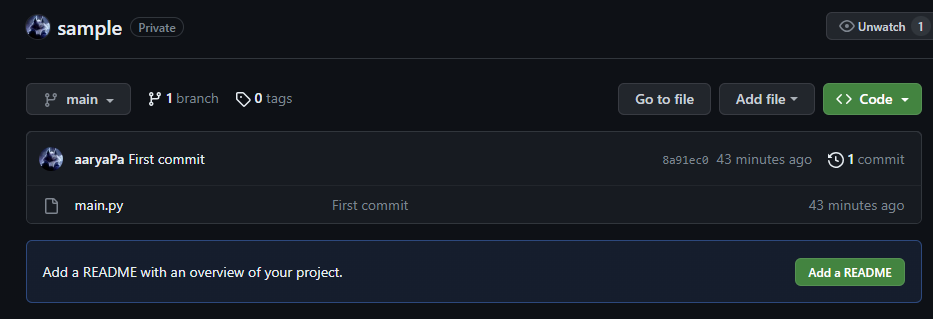
1. Now, we will add the existing repository to the main branch of our newly created repository’s staging environment.



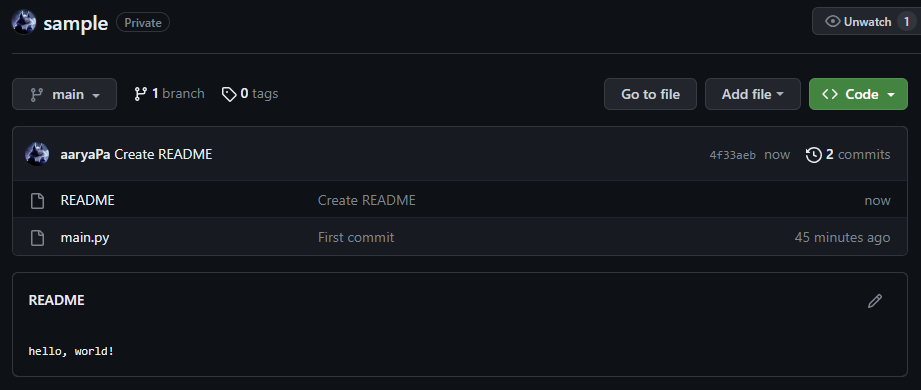
1. Now, we will push the repository.



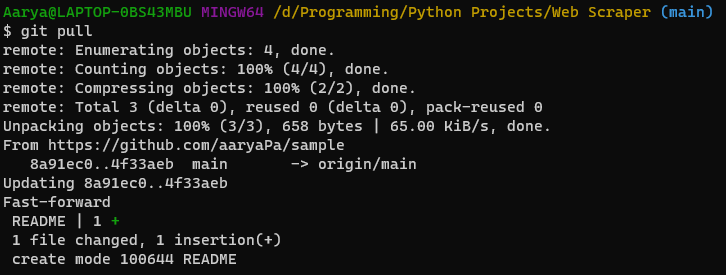
1. Now, the changes will be reflected on Github.



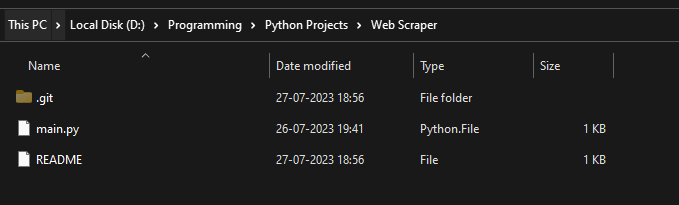
1. Now, any changes can be made to the code using Github itself.



1. Now, we can make these changes available in the local machine using the “git pull command”,



(10) The new “README” file is now also available in our local machine.



**Lab Outcome:**

To obtain complete knowledge of the “version control system” to effectively track changes augmented with Git and GitHub

**Conclusion:**

Through this experiment, we have learned about the usage of Git and GitHub and how, we can use both of them in sync with each other.