**Ex 6 GIT VERSION CONTROL SYSTEM**

**Date: 22.09.20**

**Aim:**

To study and creating a repository in GitHub.

**Description:**

**GitHub:**

GitHub is a Git repository hosting service, but it adds many of its own features. While Git is a command line tool, GitHub provides a Web-based graphical interface. It also provides access control and several collaboration features, such as a wikis and basic task management tools for every project.

GitHub is a website for developers and programmers to collaboratively work on code. The primary benefit of GitHub is its version control system, which allows for seamless collaboration without compromising the integrity of the original project.

GitHub offers its basic services free of charge. Its more advanced professional and enterprise services are commercial. Free GitHub accounts are commonly used to host [open-source](https://en.wikipedia.org/wiki/Open-source) projects. As of January 2019, GitHub offers unlimited private [repositories](https://en.wikipedia.org/wiki/Repository_(version_control)) to all plans, including free accounts, but allowed only up to three collaborators per repository for free. Starting from April 15, 2020, the free plan allows unlimited collaborators, but restricts private repositories to 2,000 actions minutes per month. As of January 2020, GitHub reports having over 40 million users and more than 190 million [repositories](https://en.wikipedia.org/wiki/Repository_(version_control))(including at least 28 million public repositories), making it the largest host of [source code](https://en.wikipedia.org/wiki/Source_code) in the world.

**Version Control:**

Version control is a system that records changes to a file or set of files over time so that you can recall specific versions later. For the examples in this book, you will use software source code as the files being version controlled, though in reality you can do this with nearly any type of file on a computer.

It is a very wise thing to use. It allows you to revert selected files back to a previous state, revert the entire project back to a previous state, compare changes over time, see who last modified something that might be causing a problem, who introduced an issue and when, and more. Using a VCS also generally means that if you screw things up or lose files, you can easily recover. In addition, you get all this for very little overhead.

**Commands:**

|  |  |  |  |
| --- | --- | --- | --- |
| Sl. No. | Command Name | Syntax | options |
| 1. | rpm | rpm {rpm-file} | **-a, --all**  Query all packages  **-f**  Query for packages owning given file |
| 2. | apt-get | apt-get [options] source pkg1 [pkg2 ...] | **-a**  It prints all the system information in the order  **-s**  It prints the kernel name.  **-n**  It prints the hostname of the network node  **-r**  It prints the kernel release date  **-v**  It prints the version of the current kernel |

**Exercise:**

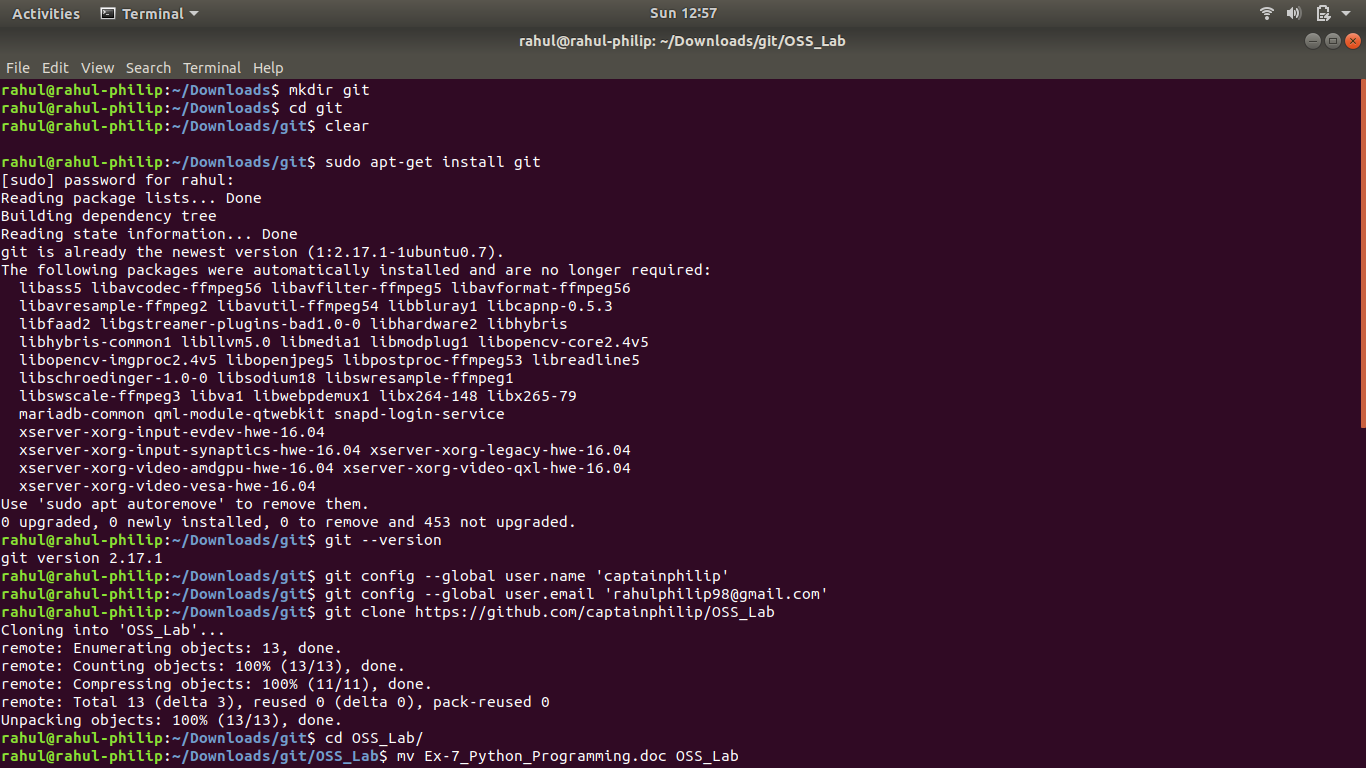
**1. Installing git and add three files to your existing repository**

Installing git using terminal

Check the version of git

Enter Your Credentials

Clone your git repository to the GitHub folder



**Part 2**

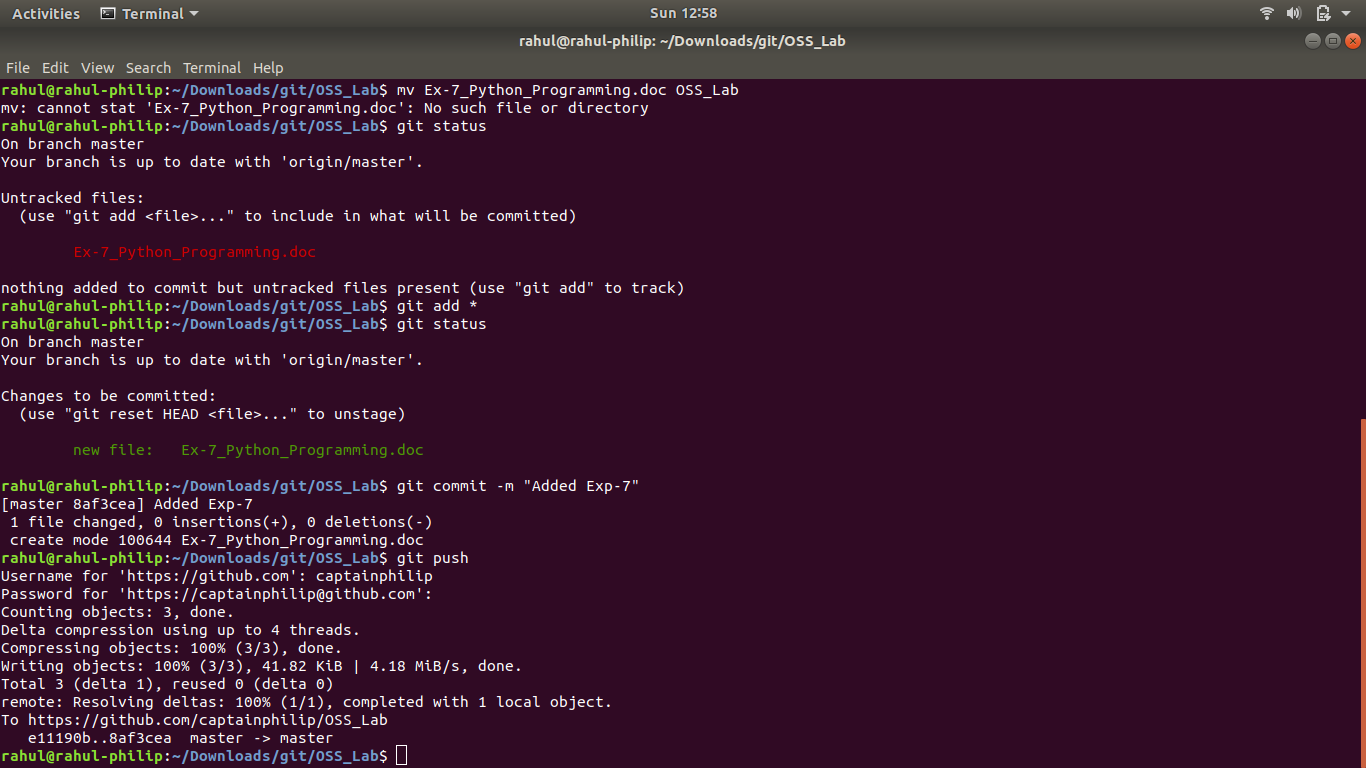
Go to the repo file and move the files which you need to add to your repository

Add the files and check the status

Give commit to save the work

Finally push the files to update your repository

Check the status of the repository



**Results:**

The GitHub version control is studied and executed.

**GitHub Link:**

<https://github.com/captainphilip/OSS_Lab>

**Video Link:** <https://youtu.be/AjWHQGnMeKY>