Instructions for Test Console Setup (Windows):

*Pre-requisites:*

- ➢ Jenkins
- ➢ Python
- ➢ PIP
- ➢ GIT
- ➢ Virtual Environment

Jenkins:

- Download the installer from link https://jenkins.io/content/thank-you-downloading-windows-installer/.

    * Here we are downloading version 2.23
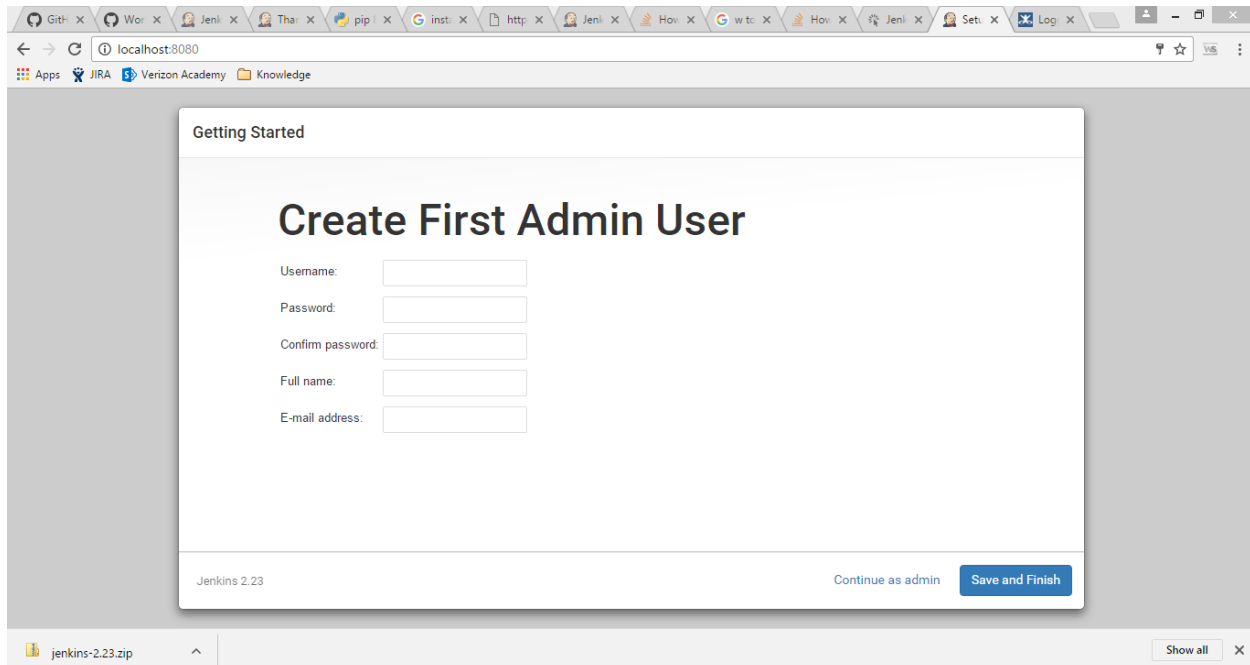
- Run the installer using admin access.
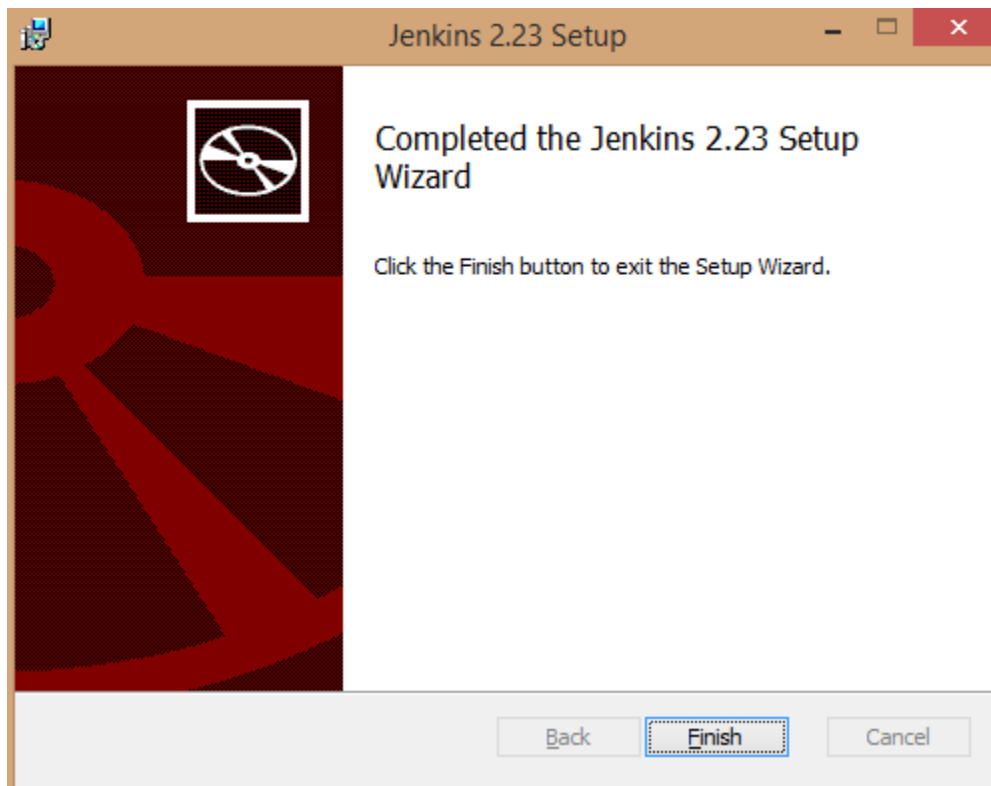
- Link on the finish button.
- User will be prompted for Admin password. Please note this is not the same password as computed password.
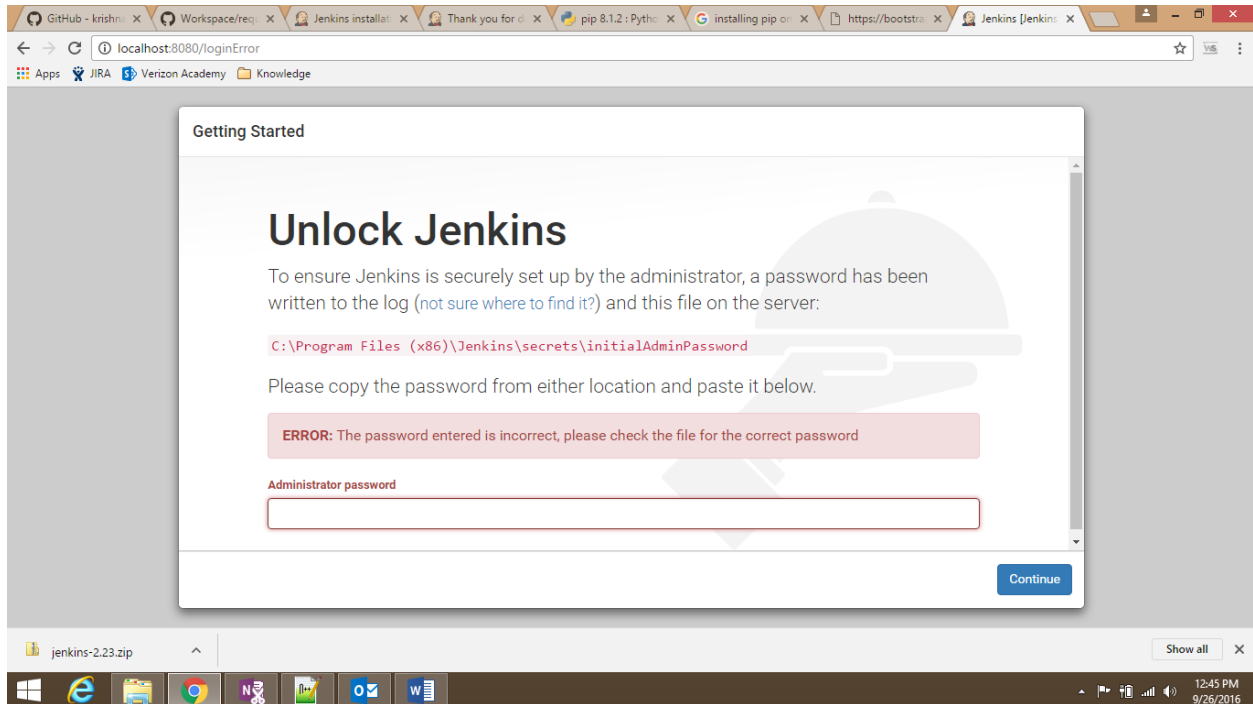
- Open with notepad below path-

  C:\Program Files (x86)\Jenkins\secrets\initialAdminPassword

- Copy the text and enter in administrator password, then hit continue.
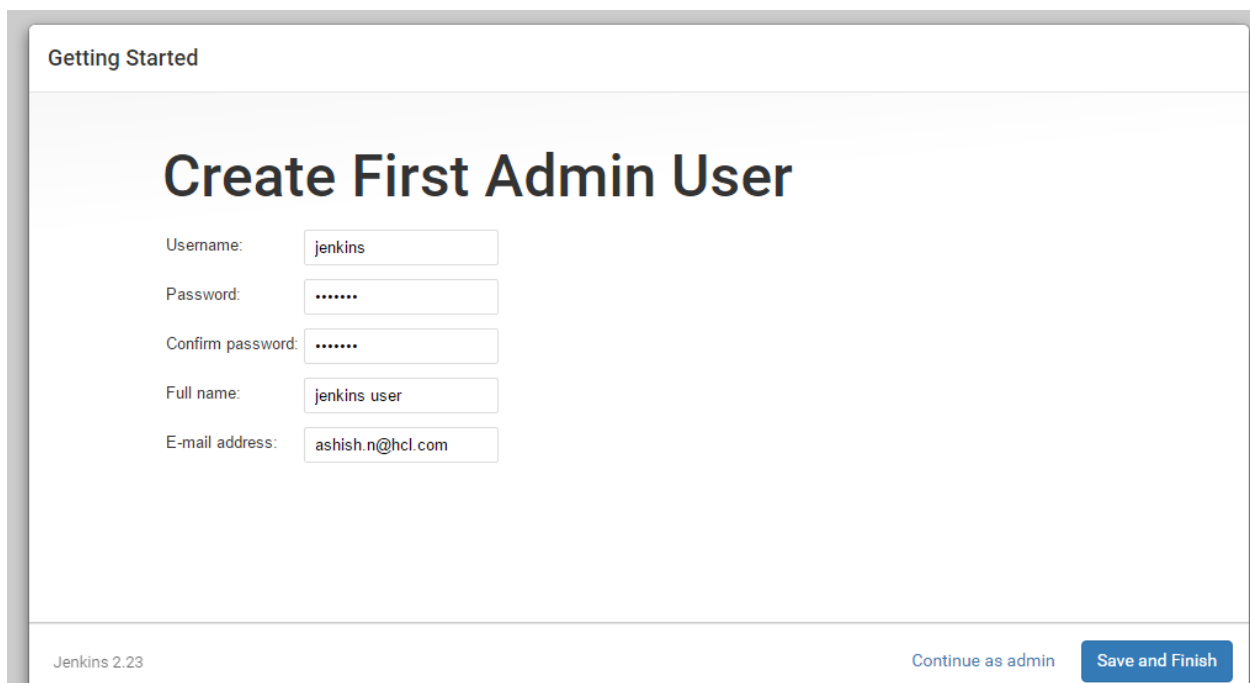- Select the first option in the window "Install Plugins". Then plugin will start downloading as below screen.
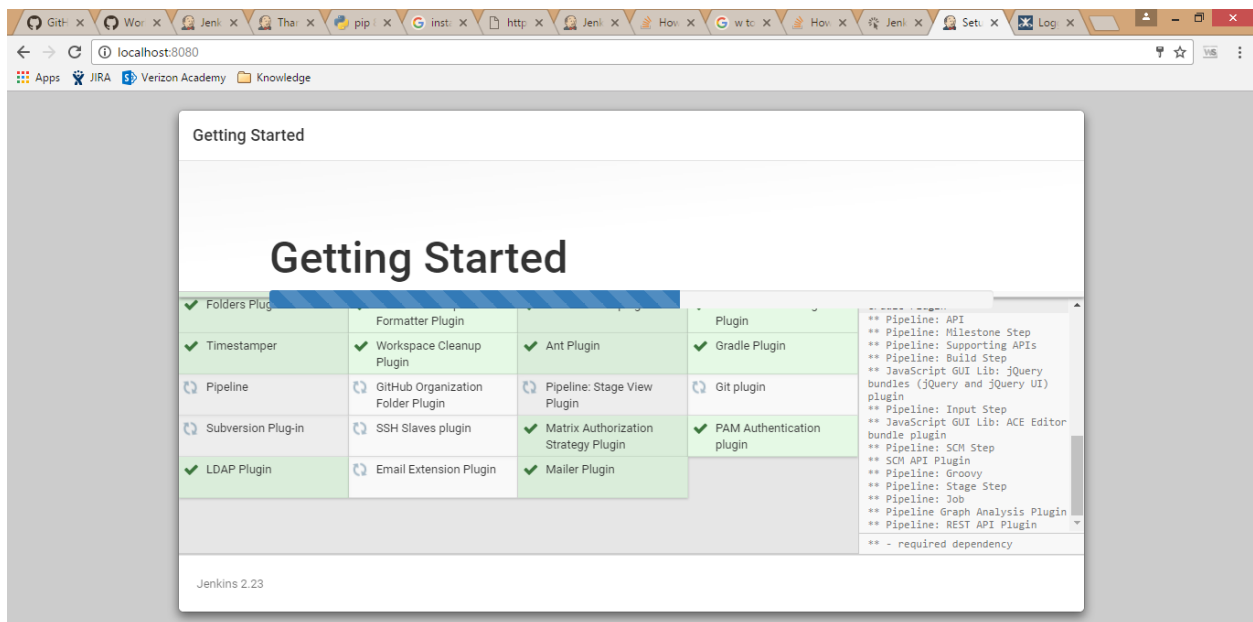
- Let the download finish.
- The next screen will prompt for Admin user creation.

- Enter the details and "continue as admin"

Getting Started

Jenkins is ready!

You've skipped creating an admin user. To log in, use the username: 'admin' and the administrator password you used to access the setup wizard.

Your Jenkins setup is complete.

Start using Jenkins

- The Jenkins is setup successfully. Click "Start using Jenkins" for confirmation. It will redirect to Jenkins page.

Note: We have to create a job manually, If jenkins is already installed. Name the job as sample, since it is the default job specified in our code. The following steps will help in creating a job.

- Create a new job by clicking on create new jobs

Welcome to Jenkins!

Please **create new jobs** to get started.

- Enter the item name as sample, and click on Freestyle project and then click ok.

Jenkins ›

Enter an item name

sample

» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Pipeline
Orchestrates long-running activities that can span multiple build slaves. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

External Job
This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed so that you can use Jenkins as a dashboard of your existing automation system.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

OK   tHub Organization
ane a GitHub organization (or user account) for all repositories matching some defined markers.

- It will direct to this page. Under 'Build', click on 'Add build step' and then click on 'Execute shell' and then click on 'Apply' and 'Save'.

- Now we can see the created job 'sample' on the Jenkins dashboard.
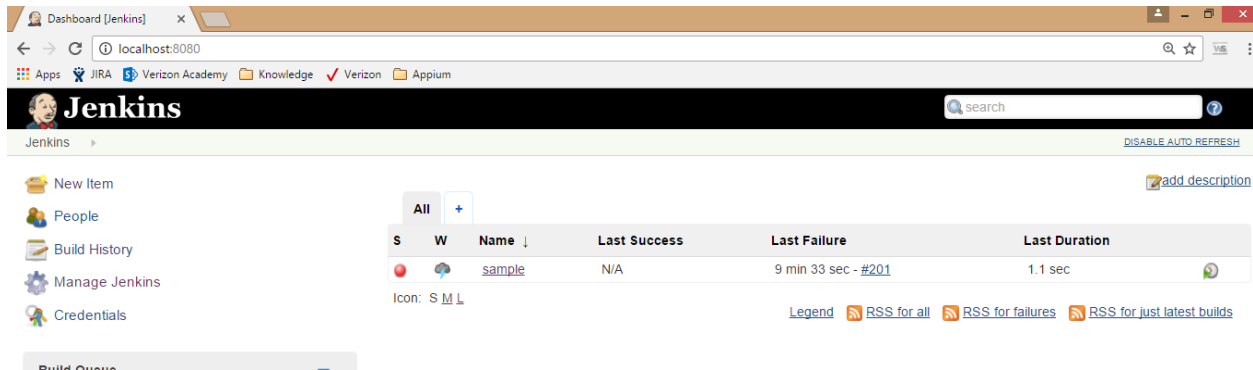
********Please note in case the User and Password is not working in Jenkins, please use below steps to bypass Username and Password. ********

- Go to C:\Program Files (x86)\Jenkins\config.xml
- Change the value

<useSecurity>true</useSecurity>
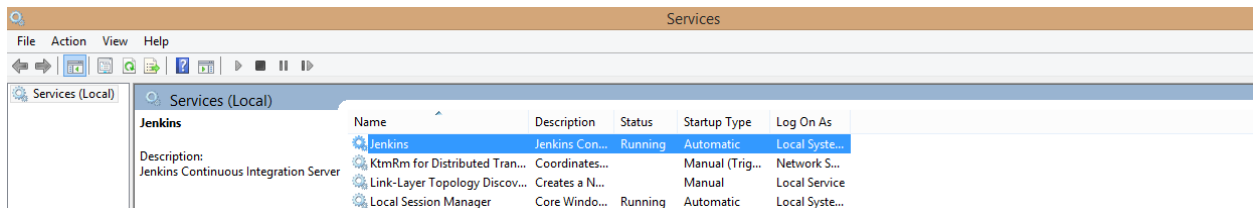
to <useSecurity>false</useSecurity>

and save.

- Go to services by entering "services.msc" in Run.

- Restart the Jenkin service.
- Open Jenkin in browser (http://localhost:8080/), and this time it should not ask for Username and Password anymore.

Please note in case the username and password are bypassed change the below line of code in view.py file (...\Workspace\app\view.py)

j = jenkins.Jenkins('http://localhost:8080', 'jenkins', 'jenkins123')

to j = jenkins.Jenkins('http://localhost:8080')

Install Python Plugin

- Go to Jenkin's menu
- Select Manage Plugin
- Search for "Python Plugin"

Python (2.7.12):

- Click on the below link for Python installation -

Windows x86-64 MSI installer

- Run the downloaded python-2.7.12.amd64.exe

- Once the installation is ciompleted successfully update the environment variable as below.
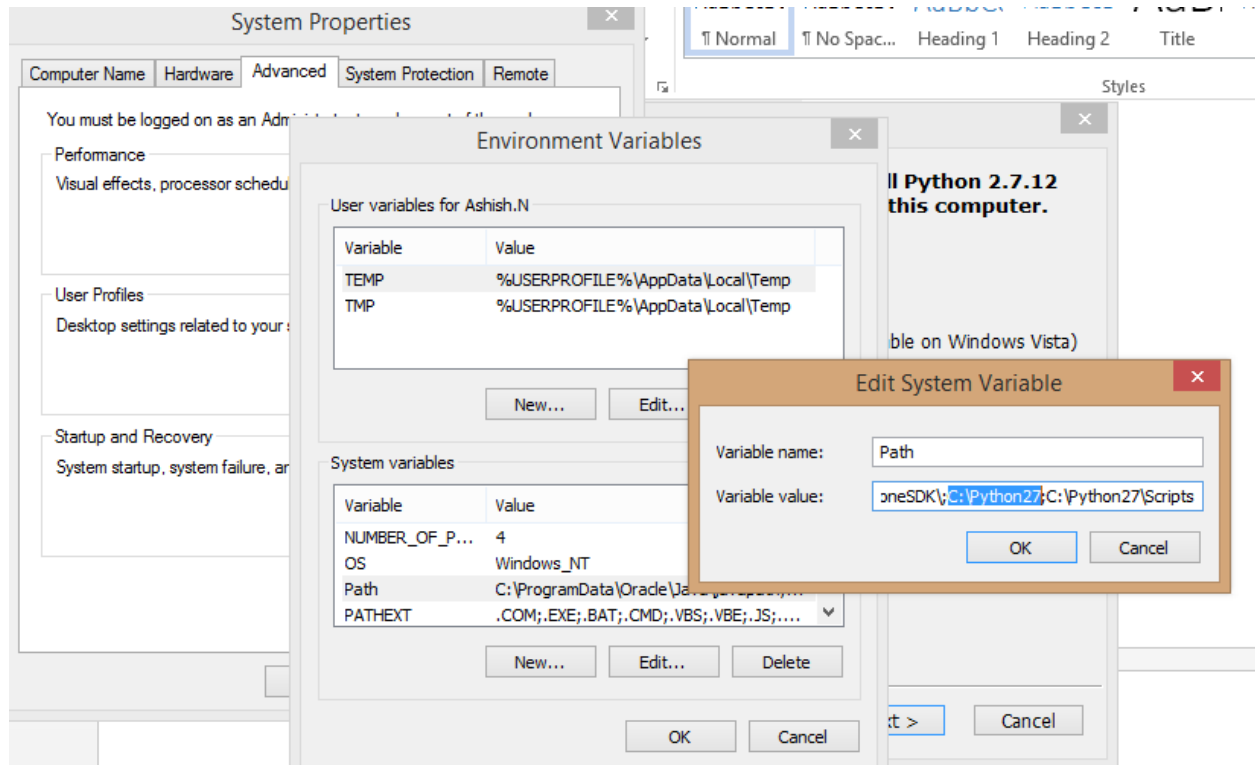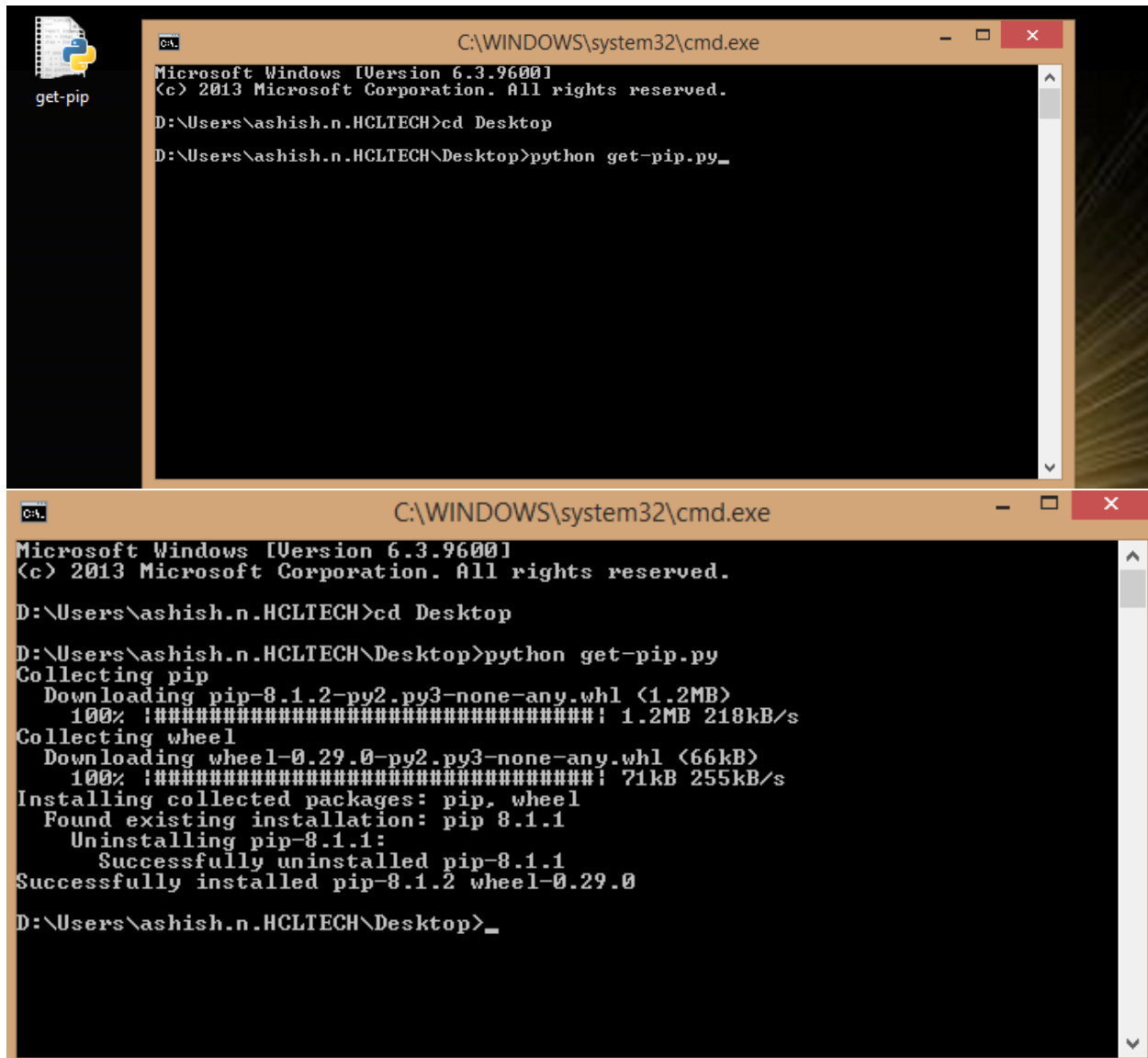
PIP: (The recommended tool for installing Python packages)

* The system should have Python already installed

- Go to link https://pip.pypa.io/en/stable/installing/ and download get-pip.py by right clicking and save link as.

* Please ensure it has to be a python file (*.py)

- Go to command prompt and run the python file.

- The PIP will be installed in the system.
- Update the system variable My Computer | <Right Click> Properties| Advanced System Settings.

- PIP installed successfully.

GIT

- **Open** https://git-scm.com/download/win
- GIT will start download

Git 2.10.0 Setup

Select Destination Location
Where should Git be installed?

Setup will install Git into the following folder.

To continue, click Next. If you would like to select a different folder, click Browse.

D:\Users\ashish.n.HCLTECH\AppData\Local\Programs\Git   Browse...

At least 192.9 MB of free disk space is required.

https://git-for-windows.github.io/

< Back   Next >   Cancel

Git 2.10.0 Setup

Adjusting your PATH environment
How would you like to use Git from the command line?

○ **Use Git from Git Bash only**

This is the safest choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.

◉ **Use Git from the Windows Command Prompt**

This option is considered safe as it only adds some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools. You will be able to use Git from both Git Bash and the Windows Command Prompt.
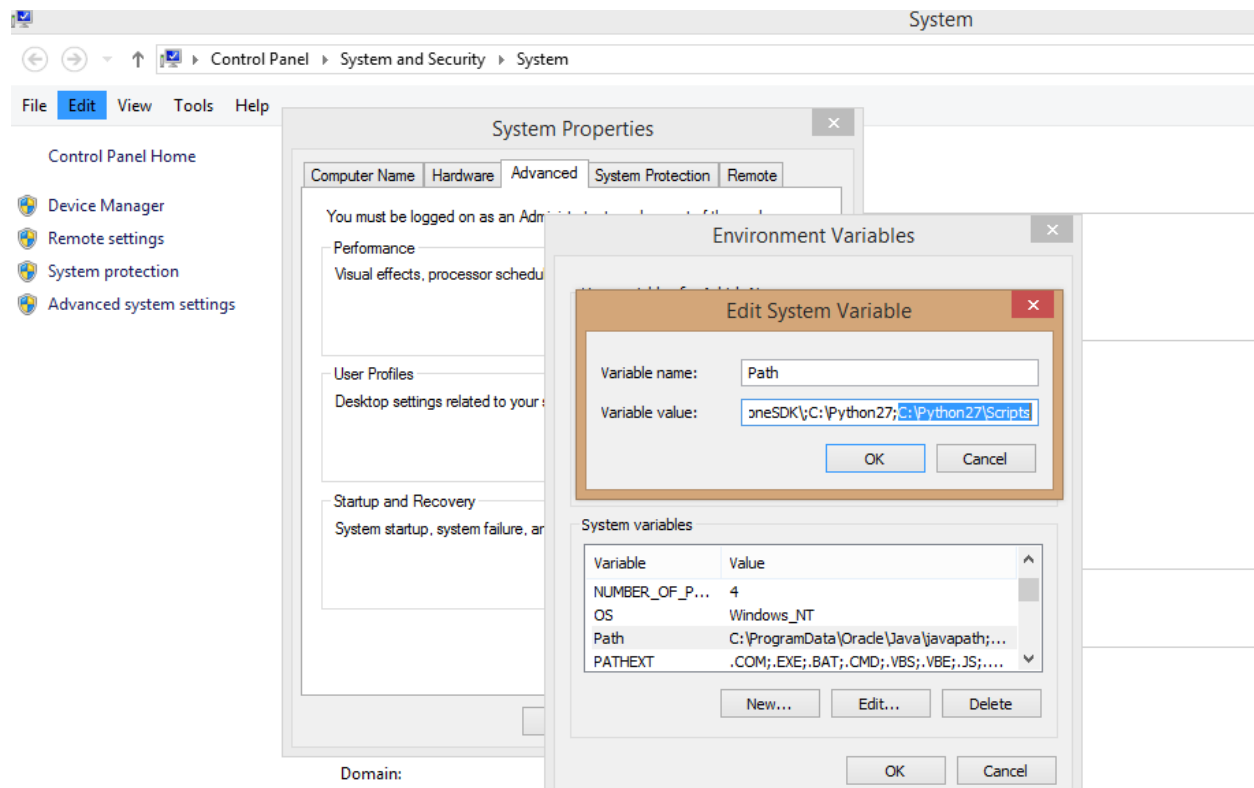
○ **Use Git and optional Unix tools from the Windows Command Prompt**

Both Git and the optional Unix tools will be added to your PATH.
Warning: This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.

https://git-for-windows.github.io/

< Back   Next >   Cancel

Git 2.10.0 Setup

Configuring the line ending conversions
How should Git treat line endings in text files?

⦿ **Checkout Windows-style, commit Unix-style line endings**

Git will convert LF to CRLF when checking out text files. When committing
text files, CRLF will be converted to LF. For cross-platform projects,
this is the recommended setting on Windows ("core.autocrlf" is set to "true").

◯ **Checkout as-is, commit Unix-style line endings**

Git will not perform any conversion when checking out text files. When
committing text files, CRLF will be converted to LF. For cross-platform projects,
this is the recommended setting on Unix ("core.autocrlf" is set to "input").

◯ **Checkout as-is, commit as-is**

Git will not perform any conversions when checking out or committing
text files. Choosing this option is not recommended for cross-platform
projects ("core.autocrlf" is set to "false").

https://git-for-windows.github.io/

[< Back] [Next >] [Cancel]

Git 2.10.0 Setup

Configuring the terminal emulator to use with Git Bash
Which terminal emulator do you want to use with your Git Bash?

⦿ **Use MinTTY (the default terminal of MSYS2)**

Git Bash will use MinTTY as terminal emulator, which sports a resizable window,
non-rectangular selections and a Unicode font. Windows console programs (such
as interactive Python) must be launched via `winpty` to work in MinTTY.

◯ **Use Windows' default console window**

Git will use the default console window of Windows ("cmd.exe"), which works well
with Win32 console programs such as interactive Python or node.js, but has a
very limited default scroll-back, needs to be configured to use a Unicode font in
order to display non-ASCII characters correctly, and prior to Windows 10 its
window was not freely resizable and it only allowed rectangular text selections.

https://git-for-windows.github.io/

[< Back] [Next >] [Cancel]
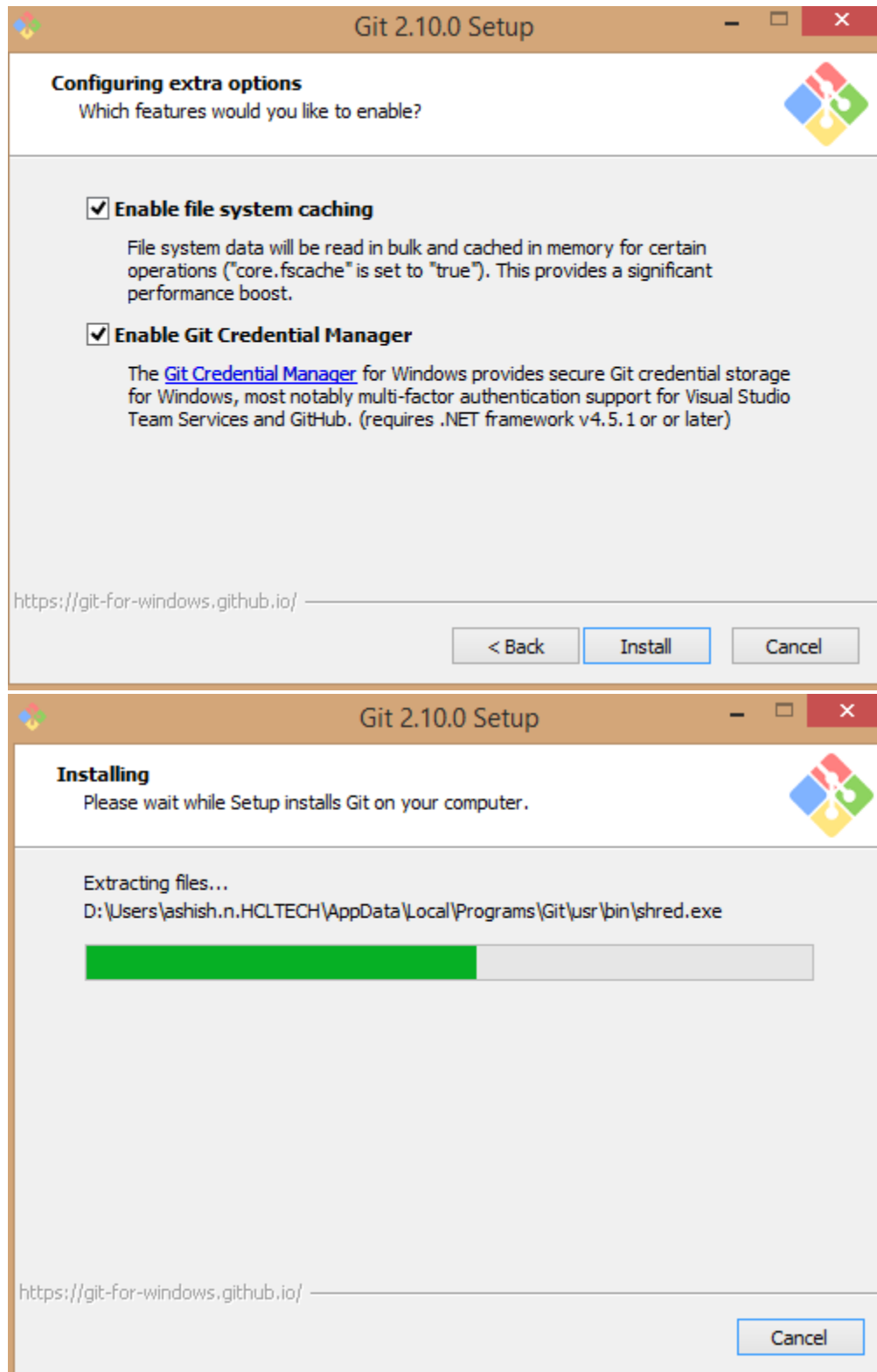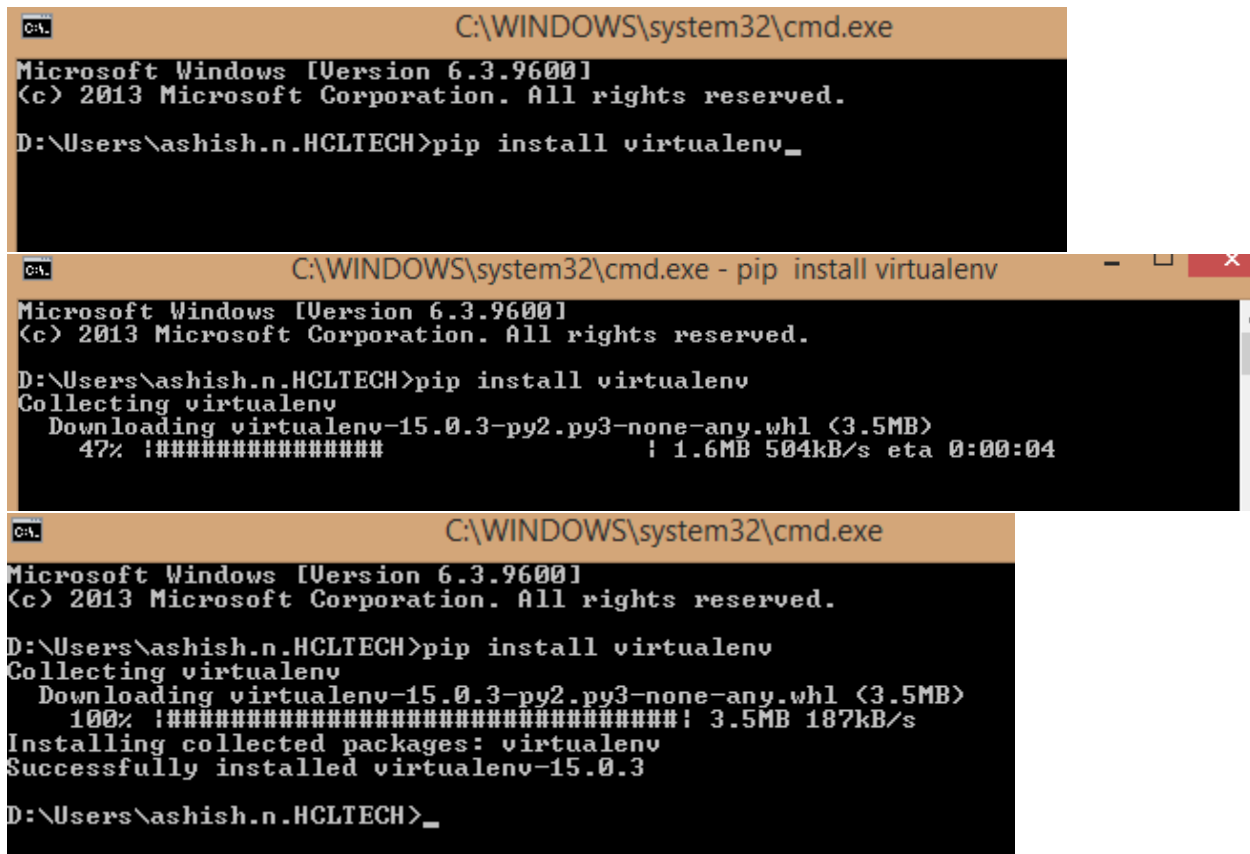
- Git created scuccessfully in Local Programs "D:\Users\ashish.n.HCLTECH\AppData\Local\Programs\Git".

Virtual Environment

- Go to cmd and enter **"pip install virtualenv"**

- Create project folders
  Commands to follow to create the project
  - D:\Users\ashish.n.HCLTECH>mkdir Project
  - D:\Users\ashish.n.HCLTECH>cd Project

- Download Git Hub Project. (https://github.com/krishnaprasad536/Workspace)
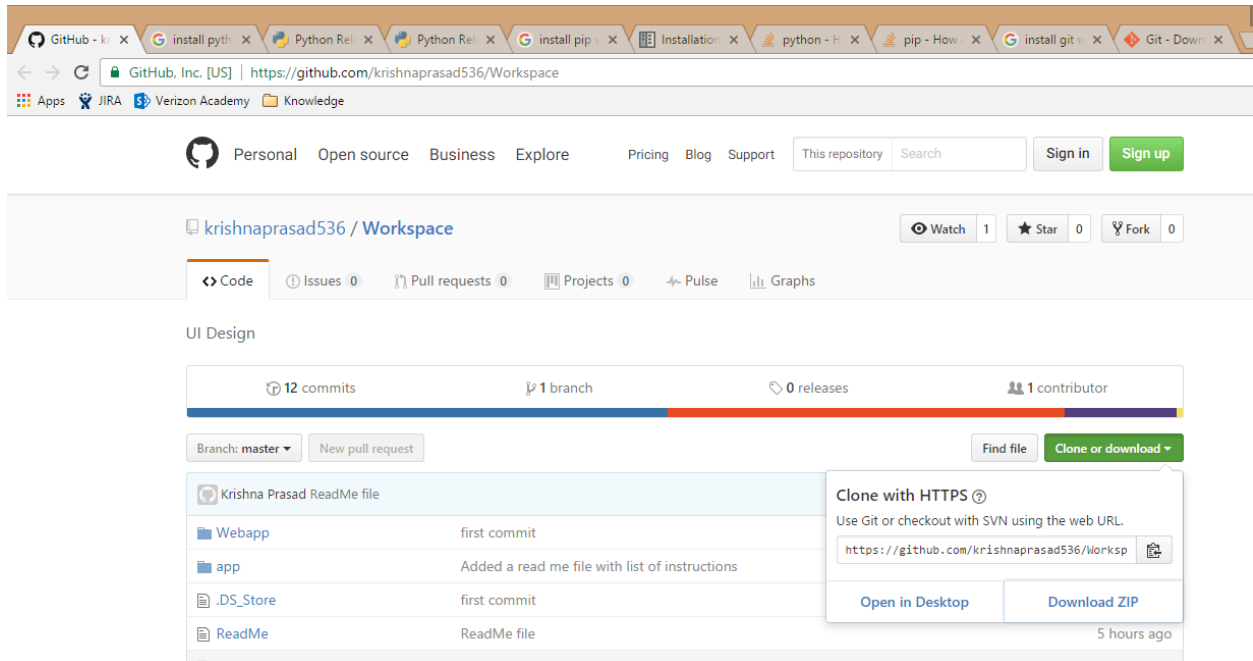
- Move the downloaded project into "Project" folder.

- Create virtual environment by running command on cmd
- Navigate to project folder and enter below commands
  (VE1) D:\Users\ashish.n.HCLTECH\Project>cd Workspace-master

  o D:\Users\ashish.n.HCLTECH\Project\Workspace-master>virtualenv VE1
- (Create virtual environment in home directory)
  o D:\Users\ashish.n.HCLTECH\Project\Workspace-master
    >VE1\Scripts\activate

* <VE1> name of virtual environment

Run below listed commands -

  o (VE1) D:\Users\ashish.n.HCLTECH\Project\Workspace-master>**pip install -r requirements.txt**
  o (VE1) D:\Users\ashish.n.HCLTECH\Project\Workspace-master>**pip freeze**
  o (VE1) D:\Users\ashish.n.HCLTECH\Project\Workspace-master>**python manage.py syncdb**

- o (VE1) D:\Users\ashish.n.HCLTECH\Project\Workspace-master>**python manage.py migrate**
- o (VE1) D:\Users\ashish.n.HCLTECH\Project\Workspace-master>**python manage.py runserver**