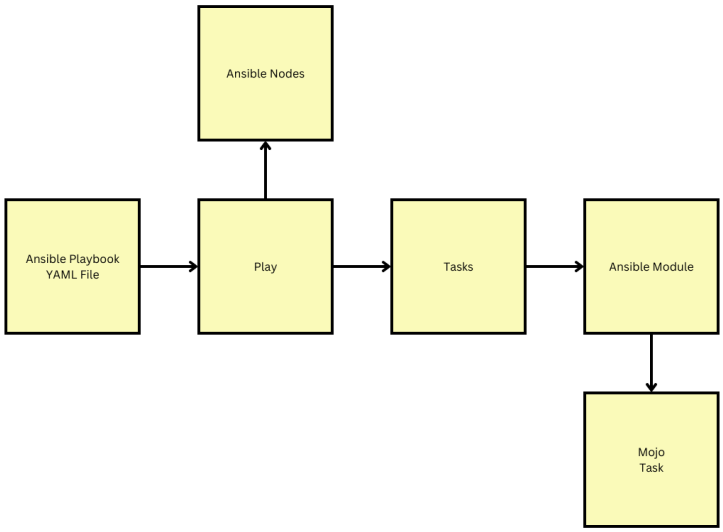
Day 4

Info - Ansible Idempotency property

```
- Idempotency is a feature of all Configuration Management tools including
Ansible
- Majority of the ansible modules support idempotency but not all of them
  - For example
    - copy, service, apt, yum modules supports idempotency
    - shell modules doesn't idempotency
- Ansible color status
  - Green - Success with no change
  - Yellow - Success with change
- Example
  - when we install nginx web server in an Ubuntu Virtual machine via
Ansible Playbook
  - ansible will check if already the Virtual machine has latest version of
nginx installed, in case it found that the machine already has latest
version then ansible do will nothing, it will simply report the task was
successful executed and it didn't change anything on the machine( it will
report in green color )
  - if suppose ansible found that the virtual machine has an older version
of nginx, then ansible will upgrade the nginx to latest
  - if suppose ansible found that the virtual machine has no nginx then, it
will install the latest version of nginx
  - once ansible has installed latest nginx when we execute the playbook it
will simply report the status in green i.e it won't repeat the installation
  - this property is called Idempotency, we don't need to anything for
this, ansible does this automatically
```

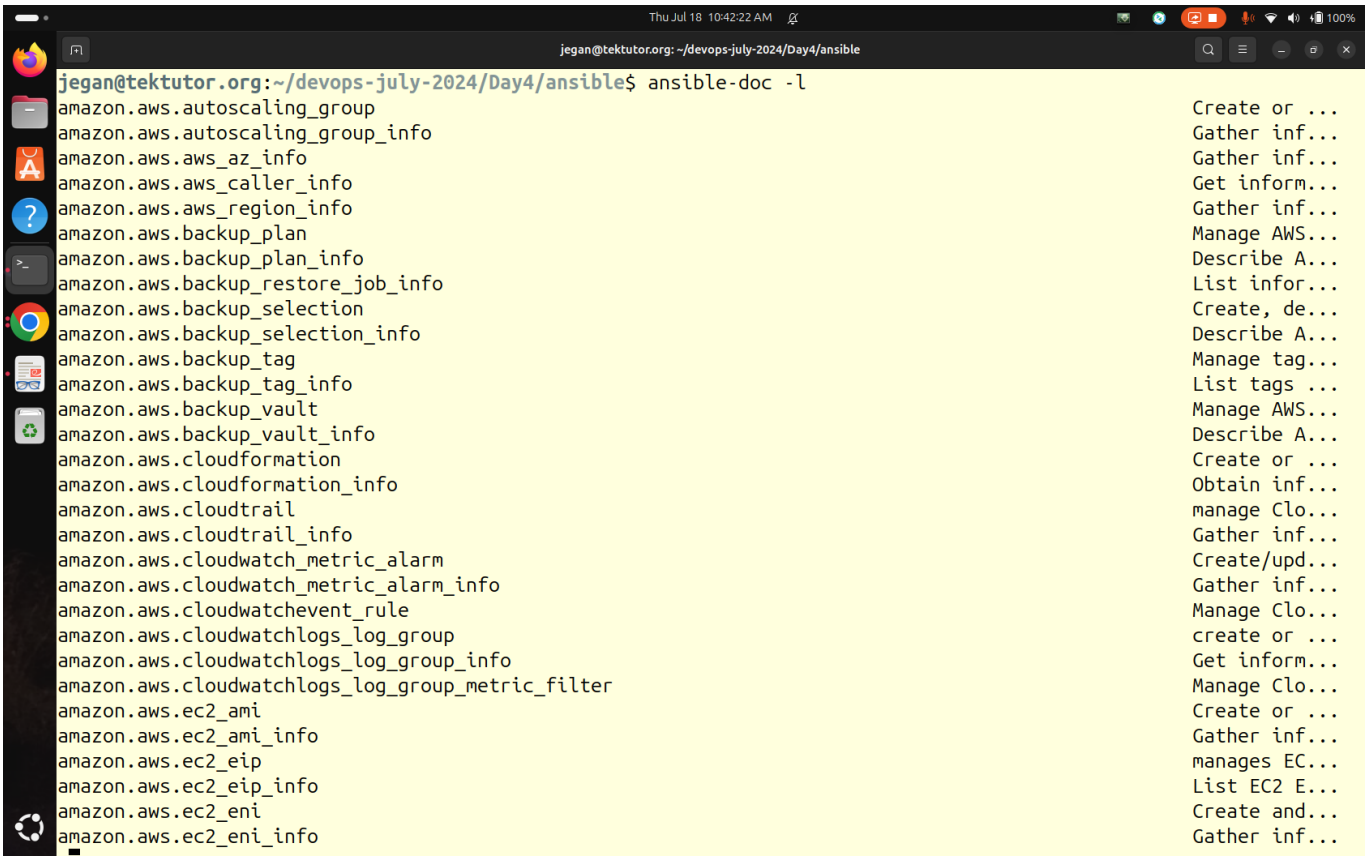
Info - Ansible Playbook Structure



Lab - Listing all the modules ansible supports

```
ansible-doc -l
```

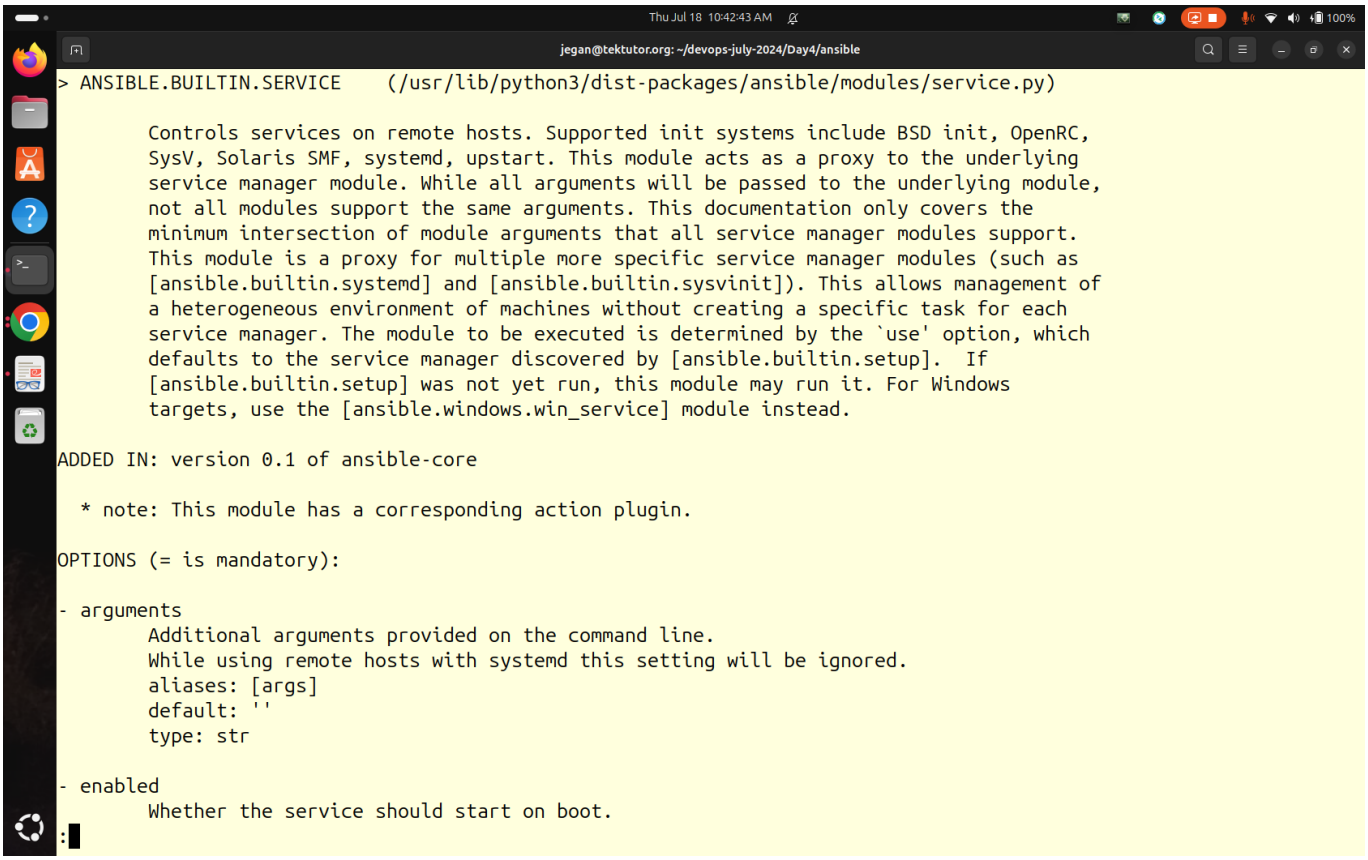
Expected output

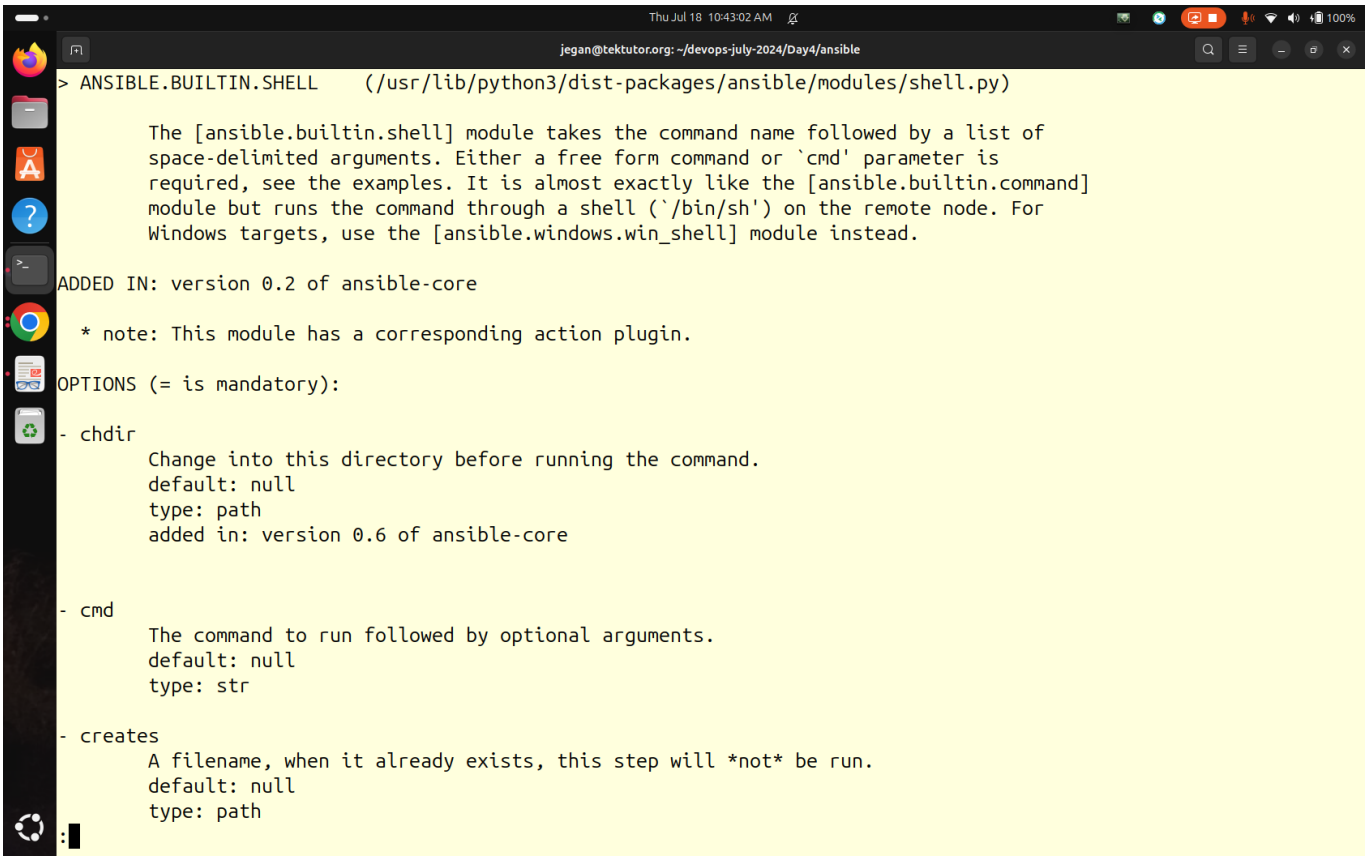


Lab - Finding syntax details of a particular ansible module

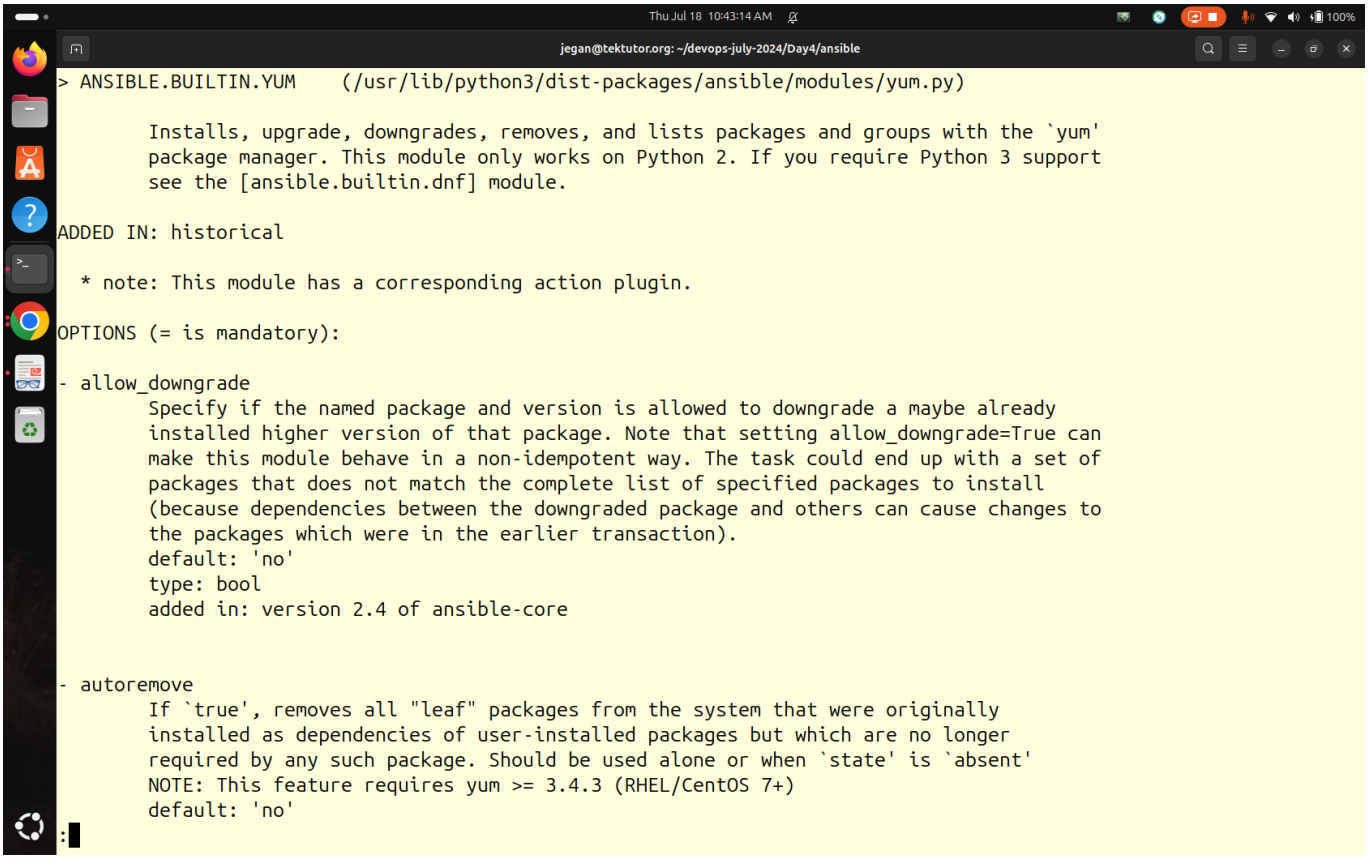
```
ansible-doc service
ansible-doc shell
ansible-doc yum
ansible-doc apt
ansible-doc file
ansible-doc template
ansible-doc copy
ansible-doc command
```

Exptected output

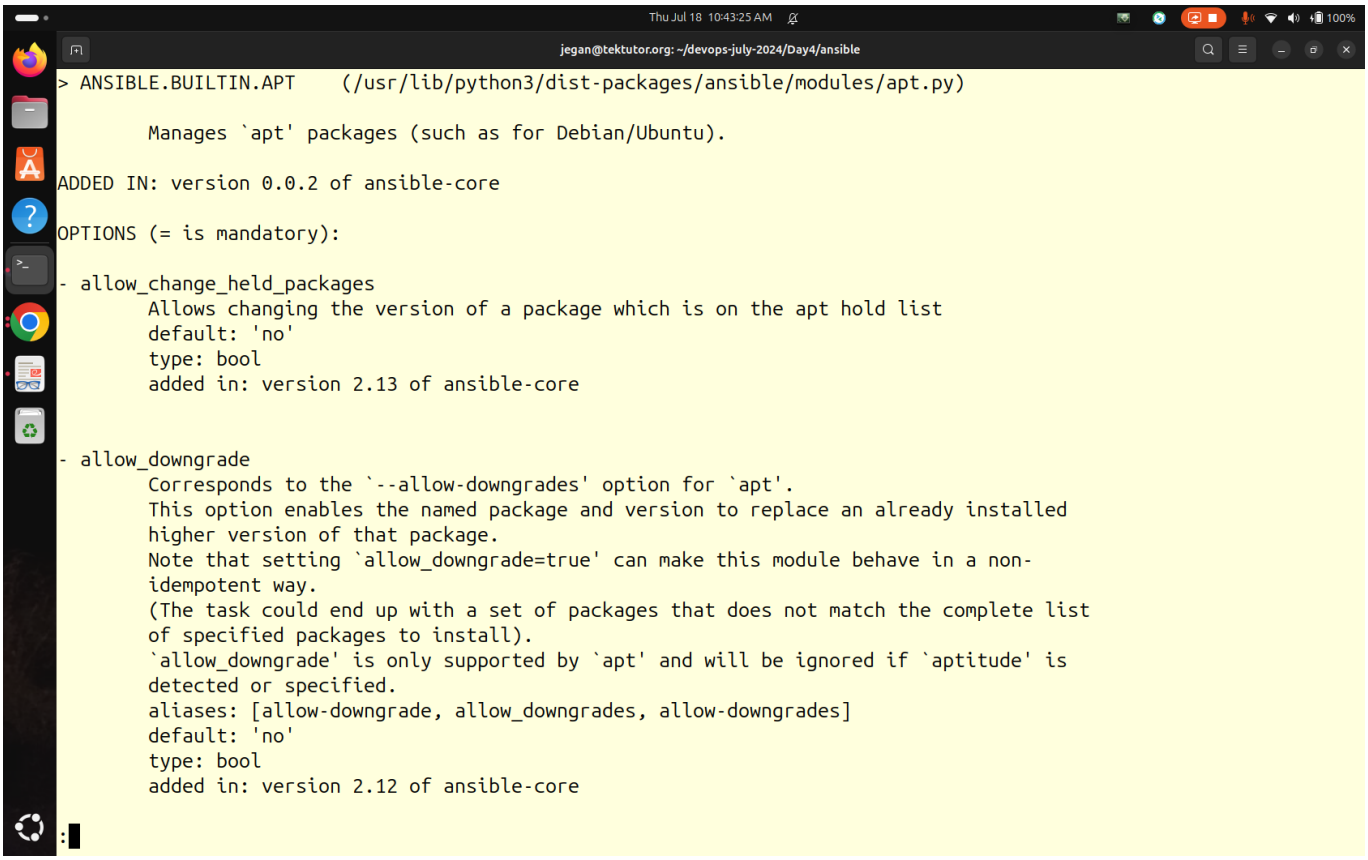




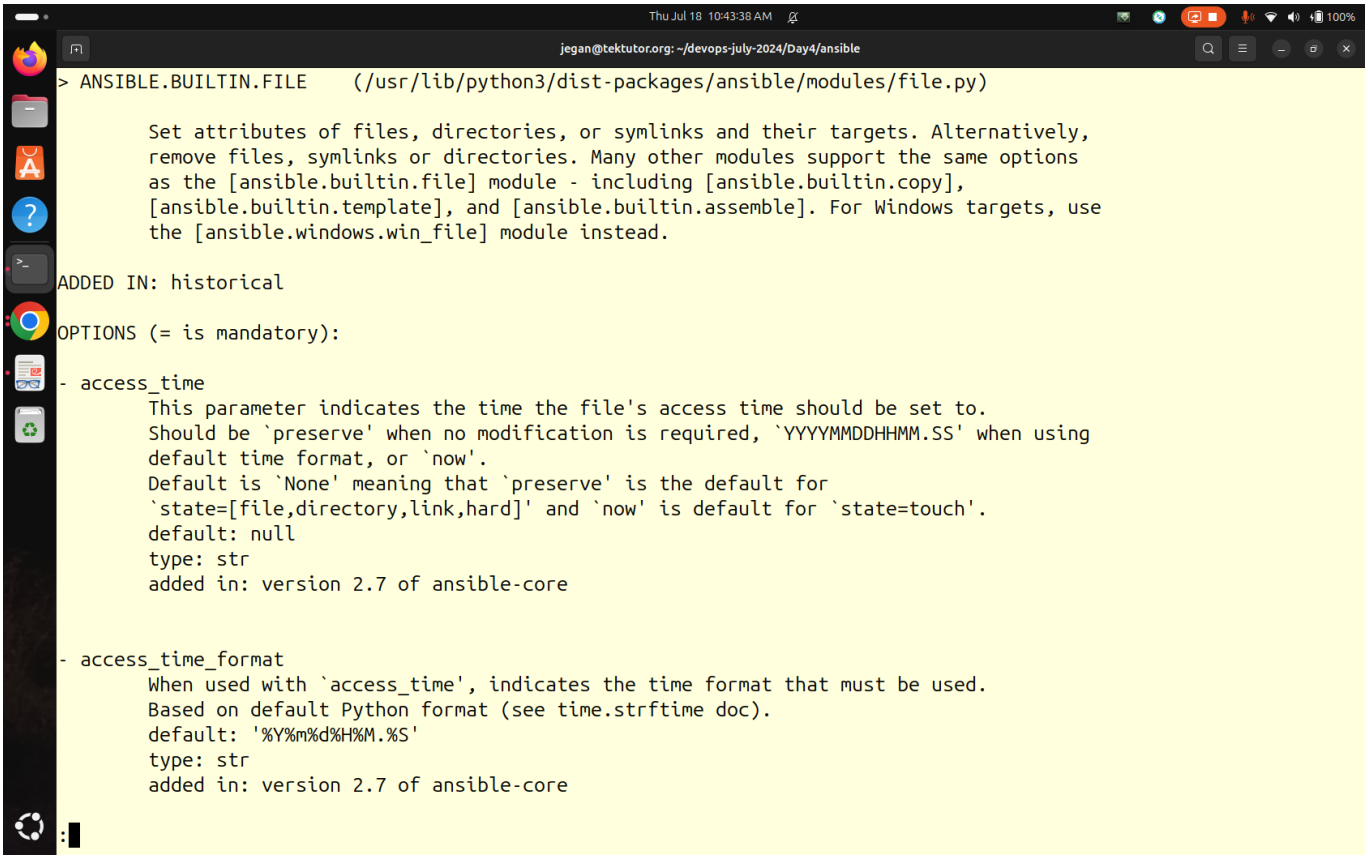

```
> ANSIBLE.BUILTIN.SHELL    (/usr/lib/python3/dist-packages/ansible/modules/shell.py)

       The [ansible.builtin.shell] module takes the command name followed by a list of
       space-delimited arguments. Either a free form command or `cmd' parameter is
       required, see the examples. It is almost exactly like the [ansible.builtin.command]
       module but runs the command through a shell (`/bin/sh') on the remote node. For
       Windows targets, use the [ansible.windows.win_shell] module instead.

ADDED IN: version 0.2 of ansible-core

  * note: This module has a corresponding action plugin.

OPTIONS (= is mandatory):

- chdir
       Change into this directory before running the command.
       default: null
       type: path
       added in: version 0.6 of ansible-core


- cmd
       The command to run followed by optional arguments.
       default: null
       type: str

- creates
       A filename, when it already exists, this step will *not* be run.
       default: null
       type: path
:
```



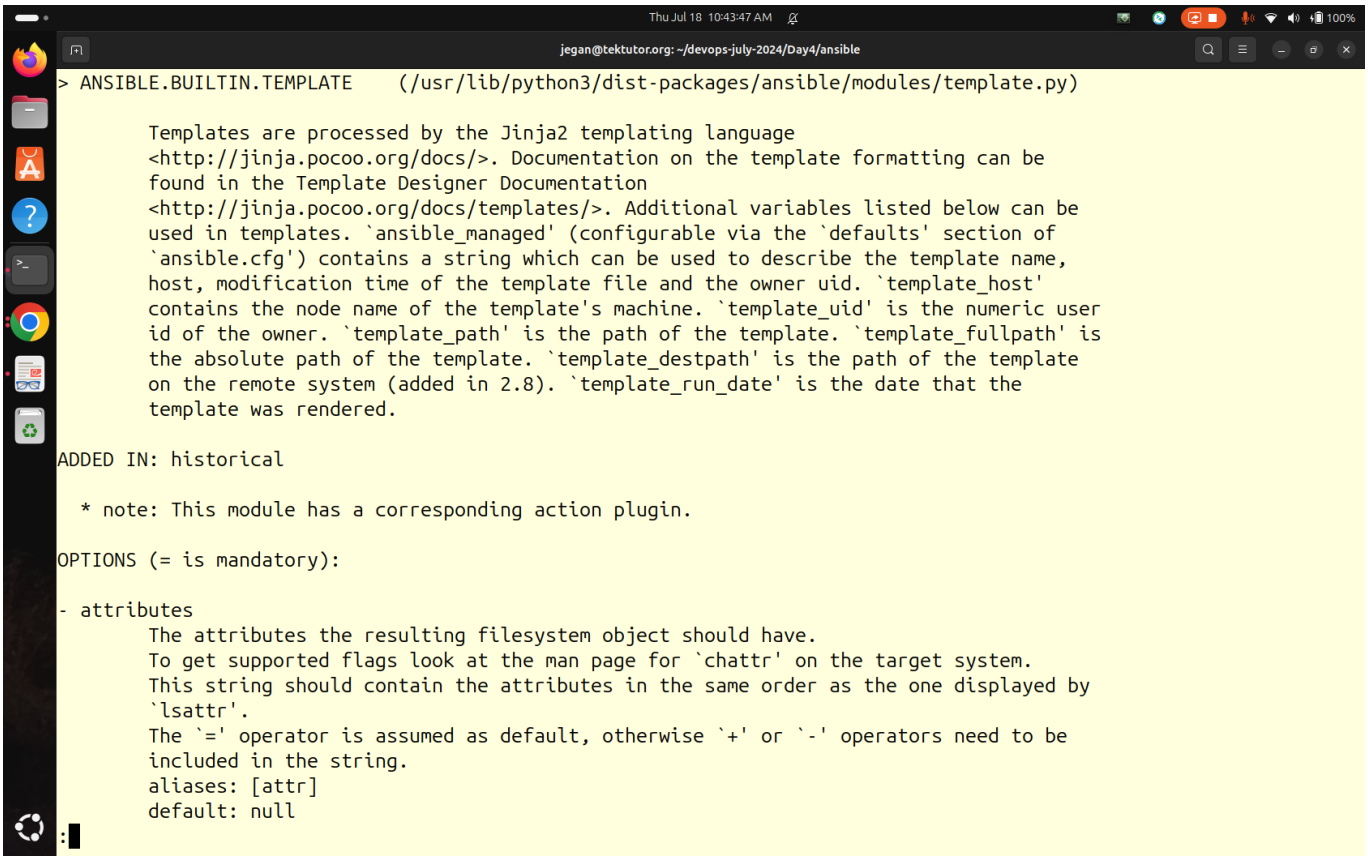

```
> ANSIBLE.BUILTIN.YUM    (/usr/lib/python3/dist-packages/ansible/modules/yum.py)

       Installs, upgrade, downgrades, removes, and lists packages and groups with the `yum'
       package manager. This module only works on Python 2. If you require Python 3 support
       see the [ansible.builtin.dnf] module.

ADDED IN: historical

  * note: This module has a corresponding action plugin.

OPTIONS (= is mandatory):

- allow_downgrade
       Specify if the named package and version is allowed to downgrade a maybe already
       installed higher version of that package. Note that setting allow_downgrade=True can
       make this module behave in a non-idempotent way. The task could end up with a set of
       packages that does not match the complete list of specified packages to install
       (because dependencies between the downgraded package and others can cause changes to
       the packages which were in the earlier transaction).
       default: 'no'
       type: bool
       added in: version 2.4 of ansible-core


- autoremove
       If `true', removes all "leaf" packages from the system that were originally
       installed as dependencies of user-installed packages but which are no longer
       required by any such package. Should be used alone or when `state' is `absent'
       NOTE: This feature requires yum >= 3.4.3 (RHEL/CentOS 7+)
       default: 'no'
:
```



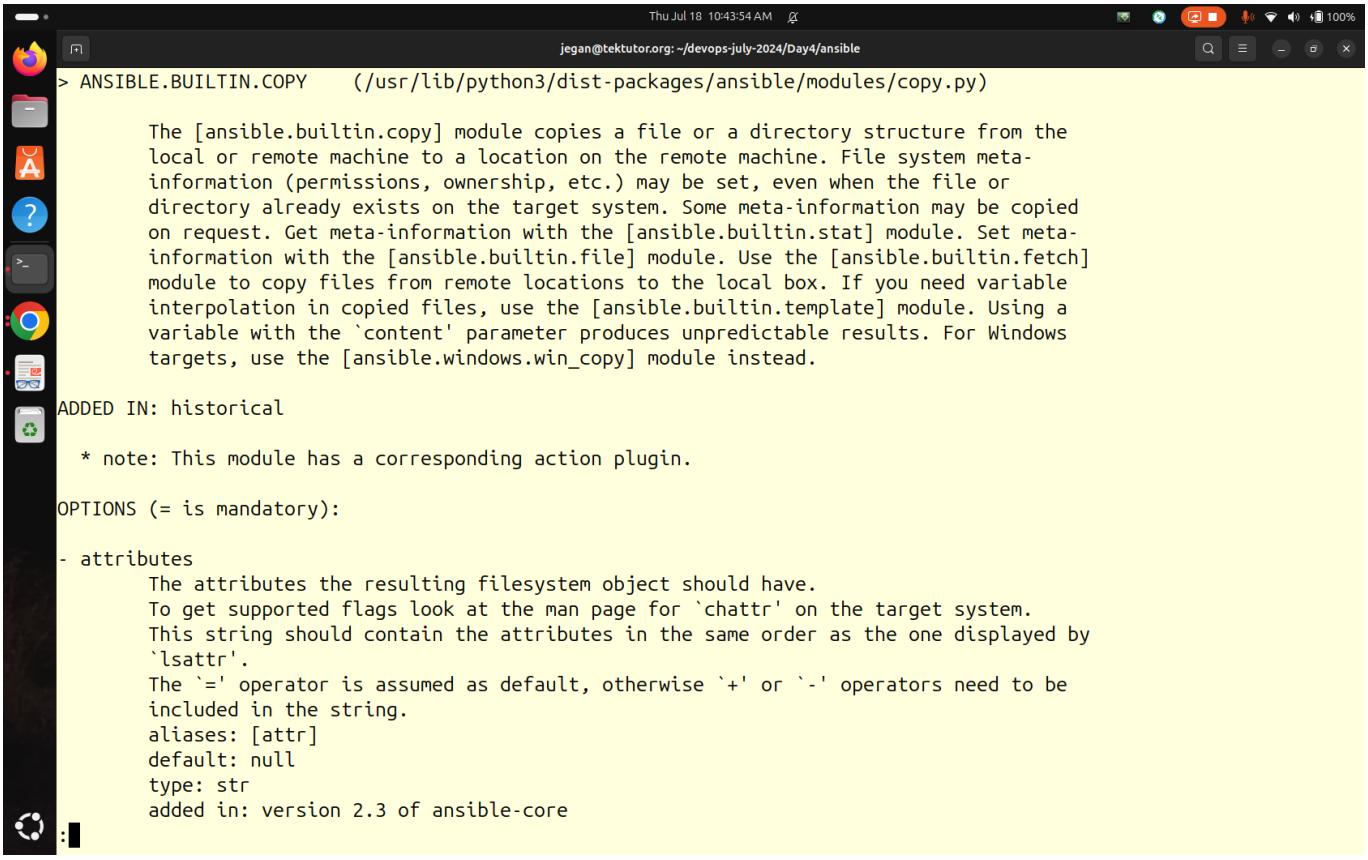

```
> ANSIBLE.BUILTIN.APT    (/usr/lib/python3/dist-packages/ansible/modules/apt.py)

       Manages `apt' packages (such as for Debian/Ubuntu).

ADDED IN: version 0.0.2 of ansible-core

OPTIONS (= is mandatory):

- allow_change_held_packages
       Allows changing the version of a package which is on the apt hold list
       default: 'no'
       type: bool
       added in: version 2.13 of ansible-core


- allow_downgrade
       Corresponds to the `--allow-downgrades' option for `apt'.
       This option enables the named package and version to replace an already installed
       higher version of that package.
       Note that setting `allow_downgrade=true' can make this module behave in a non-
       idempotent way.
       (The task could end up with a set of packages that does not match the complete list
       of specified packages to install).
       `allow_downgrade' is only supported by `apt' and will be ignored if `aptitude' is
       detected or specified.
       aliases: [allow-downgrade, allow_downgrades, allow-downgrades]
       default: 'no'
       type: bool
       added in: version 2.12 of ansible-core

:
```



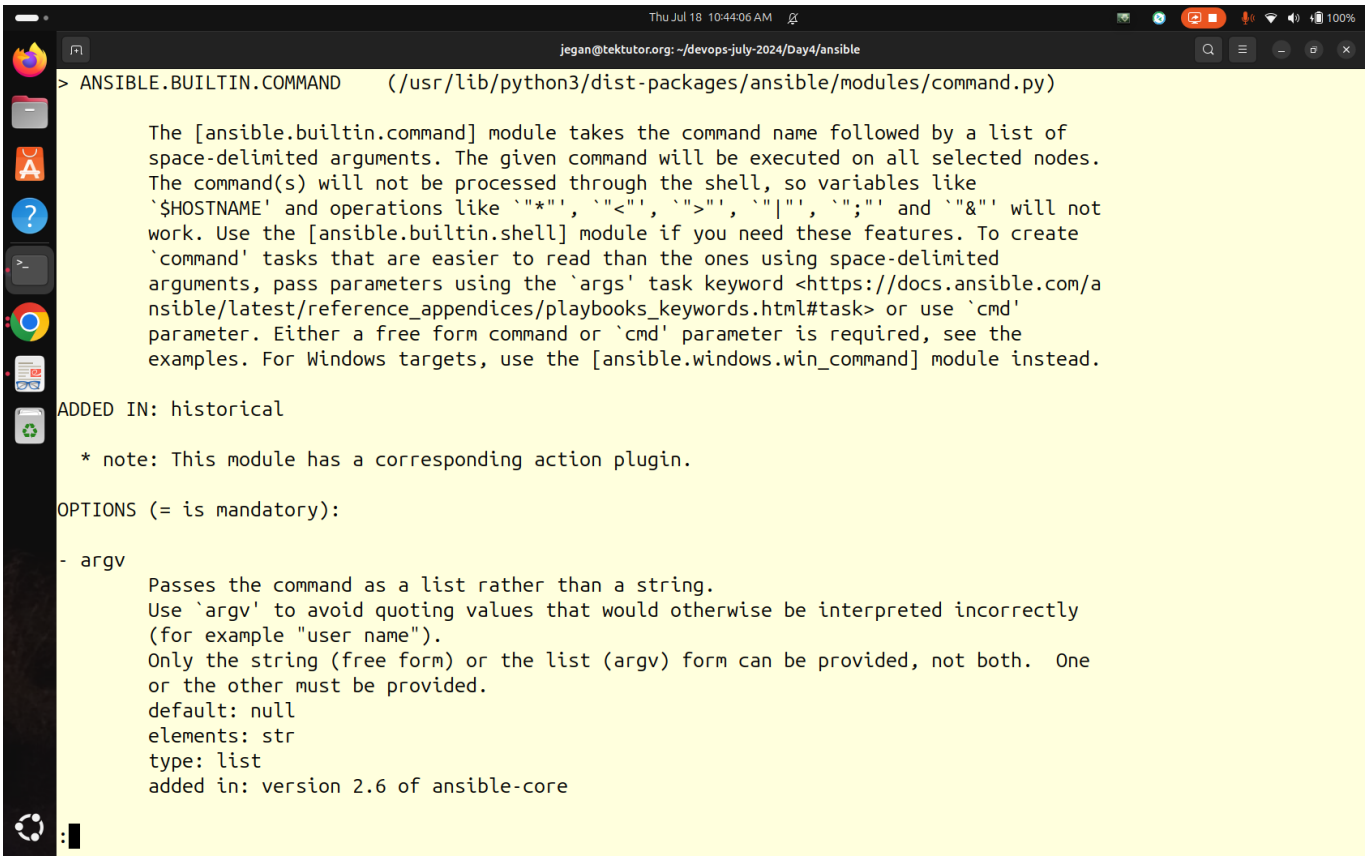

```
> ANSIBLE.BUILTIN.FILE    (/usr/lib/python3/dist-packages/ansible/modules/file.py)

       Set attributes of files, directories, or symlinks and their targets. Alternatively,
       remove files, symlinks or directories. Many other modules support the same options
       as the [ansible.builtin.file] module - including [ansible.builtin.copy],
       [ansible.builtin.template], and [ansible.builtin.assemble]. For Windows targets, use
       the [ansible.windows.win_file] module instead.

ADDED IN: historical

OPTIONS (= is mandatory):

- access_time
       This parameter indicates the time the file's access time should be set to.
       Should be `preserve' when no modification is required, `YYYYMMDDHHMM.SS' when using
       default time format, or `now'.
       Default is `None' meaning that `preserve' is the default for
       `state=[file,directory,link,hard]' and `now' is default for `state=touch'.
       default: null
       type: str
       added in: version 2.7 of ansible-core


- access_time_format
       When used with `access_time', indicates the time format that must be used.
       Based on default Python format (see time.strftime doc).
       default: '%Y%m%d%H%M.%S'
       type: str
       added in: version 2.7 of ansible-core

:
```




```
> ANSIBLE.BUILTIN.TEMPLATE    (/usr/lib/python3/dist-packages/ansible/modules/template.py)

        Templates are processed by the Jinja2 templating language
        <http://jinja.pocoo.org/docs/>. Documentation on the template formatting can be
        found in the Template Designer Documentation
        <http://jinja.pocoo.org/docs/templates/>. Additional variables listed below can be
        used in templates. `ansible_managed' (configurable via the `defaults' section of
        `ansible.cfg') contains a string which can be used to describe the template name,
        host, modification time of the template file and the owner uid. `template_host'
        contains the node name of the template's machine. `template_uid' is the numeric user
        id of the owner. `template_path' is the path of the template. `template_fullpath' is
        the absolute path of the template. `template_destpath' is the path of the template
        on the remote system (added in 2.8). `template_run_date' is the date that the
        template was rendered.

ADDED IN: historical

  * note: This module has a corresponding action plugin.

OPTIONS (= is mandatory):

- attributes
        The attributes the resulting filesystem object should have.
        To get supported flags look at the man page for `chattr' on the target system.
        This string should contain the attributes in the same order as the one displayed by
        `lsattr'.
        The `=' operator is assumed as default, otherwise `+' or `-' operators need to be
        included in the string.
        aliases: [attr]
        default: null
:
```



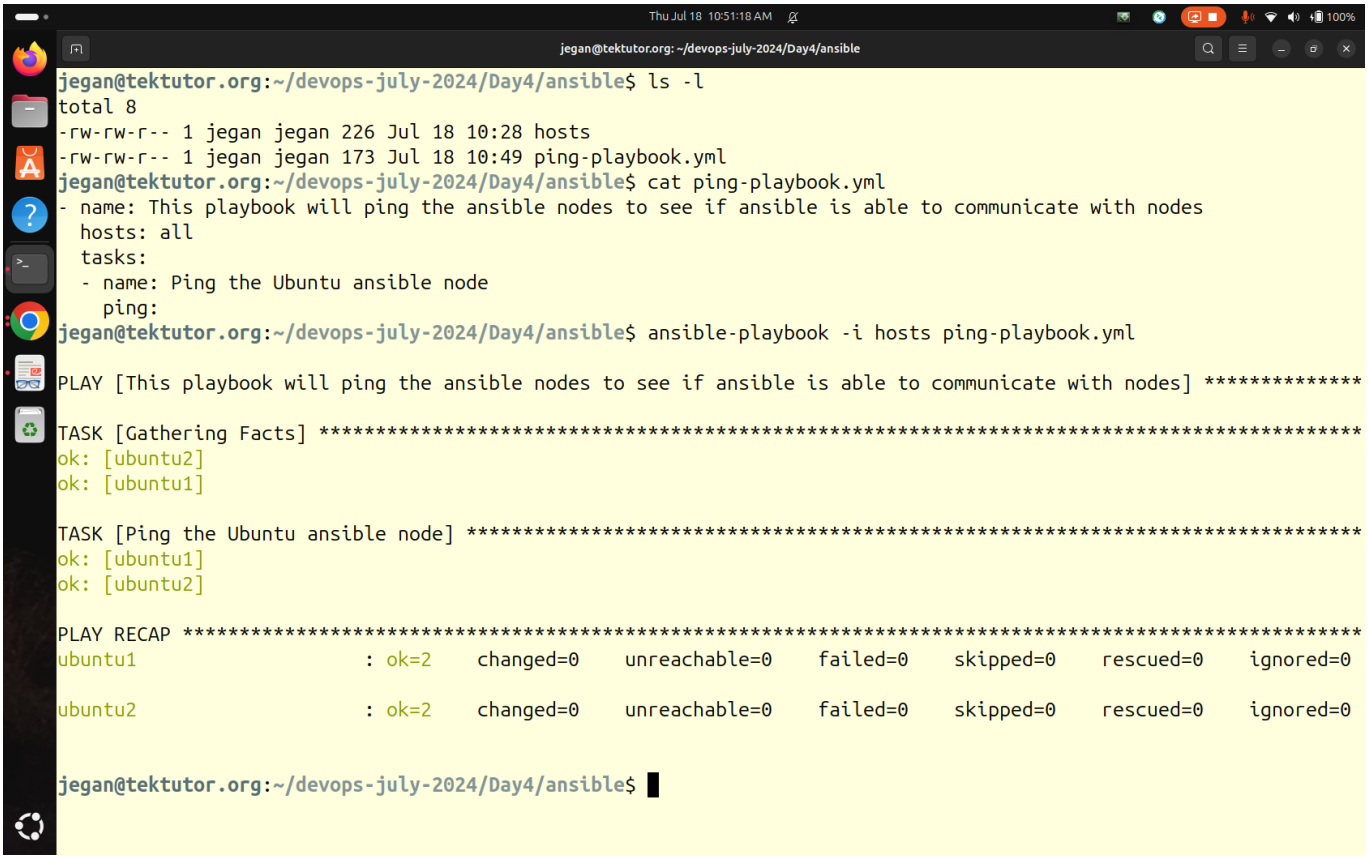

```
> ANSIBLE.BUILTIN.COPY    (/usr/lib/python3/dist-packages/ansible/modules/copy.py)

        The [ansible.builtin.copy] module copies a file or a directory structure from the
        local or remote machine to a location on the remote machine. File system meta-
        information (permissions, ownership, etc.) may be set, even when the file or
        directory already exists on the target system. Some meta-information may be copied
        on request. Get meta-information with the [ansible.builtin.stat] module. Set meta-
        information with the [ansible.builtin.file] module. Use the [ansible.builtin.fetch]
        module to copy files from remote locations to the local box. If you need variable
        interpolation in copied files, use the [ansible.builtin.template] module. Using a
        variable with the `content' parameter produces unpredictable results. For Windows
        targets, use the [ansible.windows.win_copy] module instead.

ADDED IN: historical

  * note: This module has a corresponding action plugin.

OPTIONS (= is mandatory):

- attributes
        The attributes the resulting filesystem object should have.
        To get supported flags look at the man page for `chattr' on the target system.
        This string should contain the attributes in the same order as the one displayed by
        `lsattr'.
        The `=' operator is assumed as default, otherwise `+' or `-' operators need to be
        included in the string.
        aliases: [attr]
        default: null
        type: str
        added in: version 2.3 of ansible-core
:
```



Lab - Running your first ansible playbook

```
cd ~/devops-july-2024
git pull
cd Day4/ansible
cat ping-playbook.yml
ansible-playbook -i hosts ping-playbook.yml
```

Expected output



Lab - Multiple Plays in a single Playbook

```
cd ~/devops-july-2024
git pull
cd Day4/ansible
cat multiple-play-in-a-playbook.yml
ansible-playbook -i hosts multiple-play-in-a-playbook.yml
```

Expected output

Lab - Understanding Ansible Setup module

```
cd ~/devops-july-2024
git pull
cd Day4/ansible
ansible -i hosts ubuntu1 -m setup
```

Expected output

Things to note

```
- ansible setup module will be the default module which will be executed in
every Playbook
- the setup module will retrieve many details about the ansible node
```

```
- just to give an idea
  - setup module collects details like
    - hostname of the ansible node
    - Operating System details
    - System Hardware details
    - Operating System Family
    - OS version
    - Python version installed on the node
    - Package Manager supported on the node
```

Lab - Install nginx in Ansible nodes using Ansible playbook

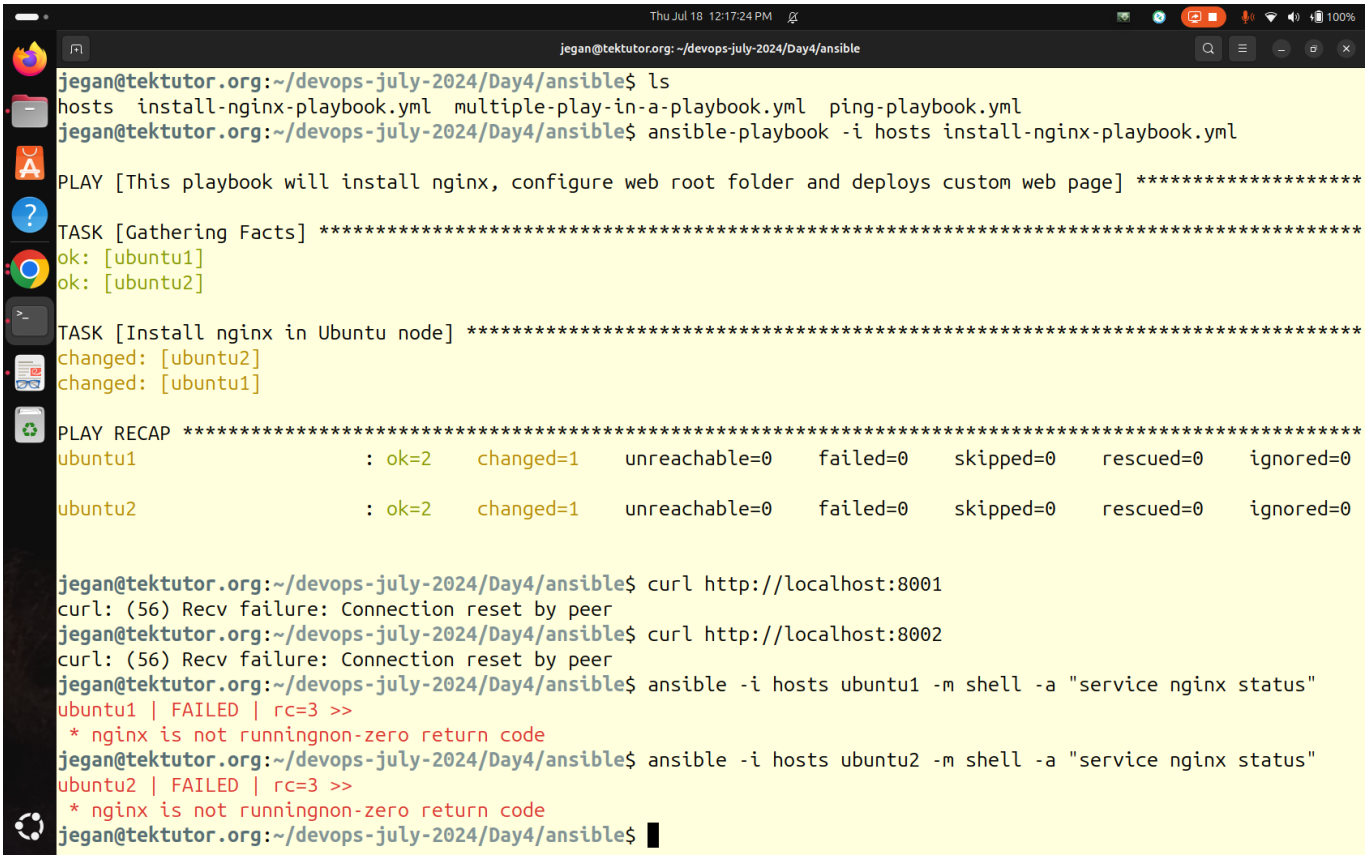
Install curl utility locally

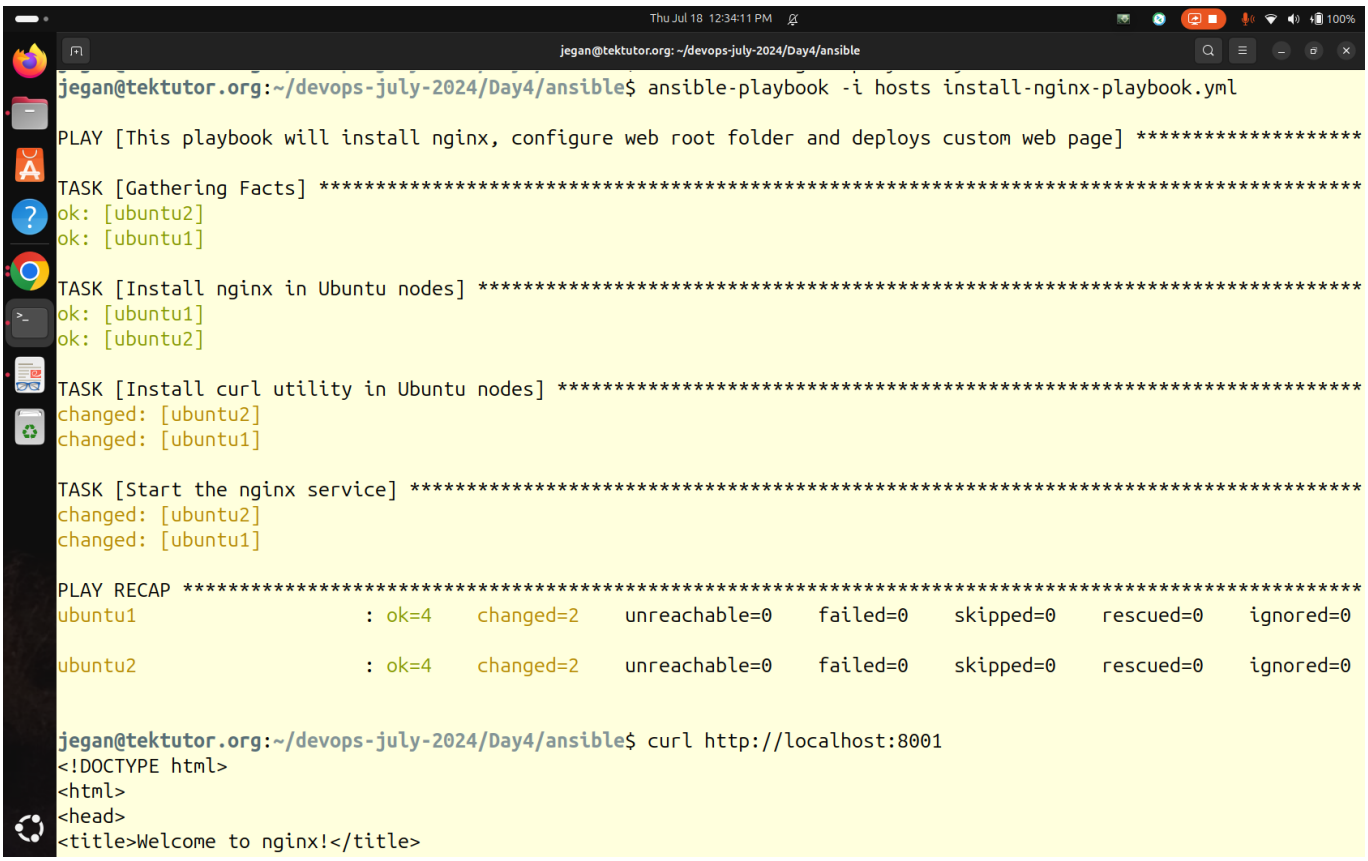
```
sudo apt install -y curl
```

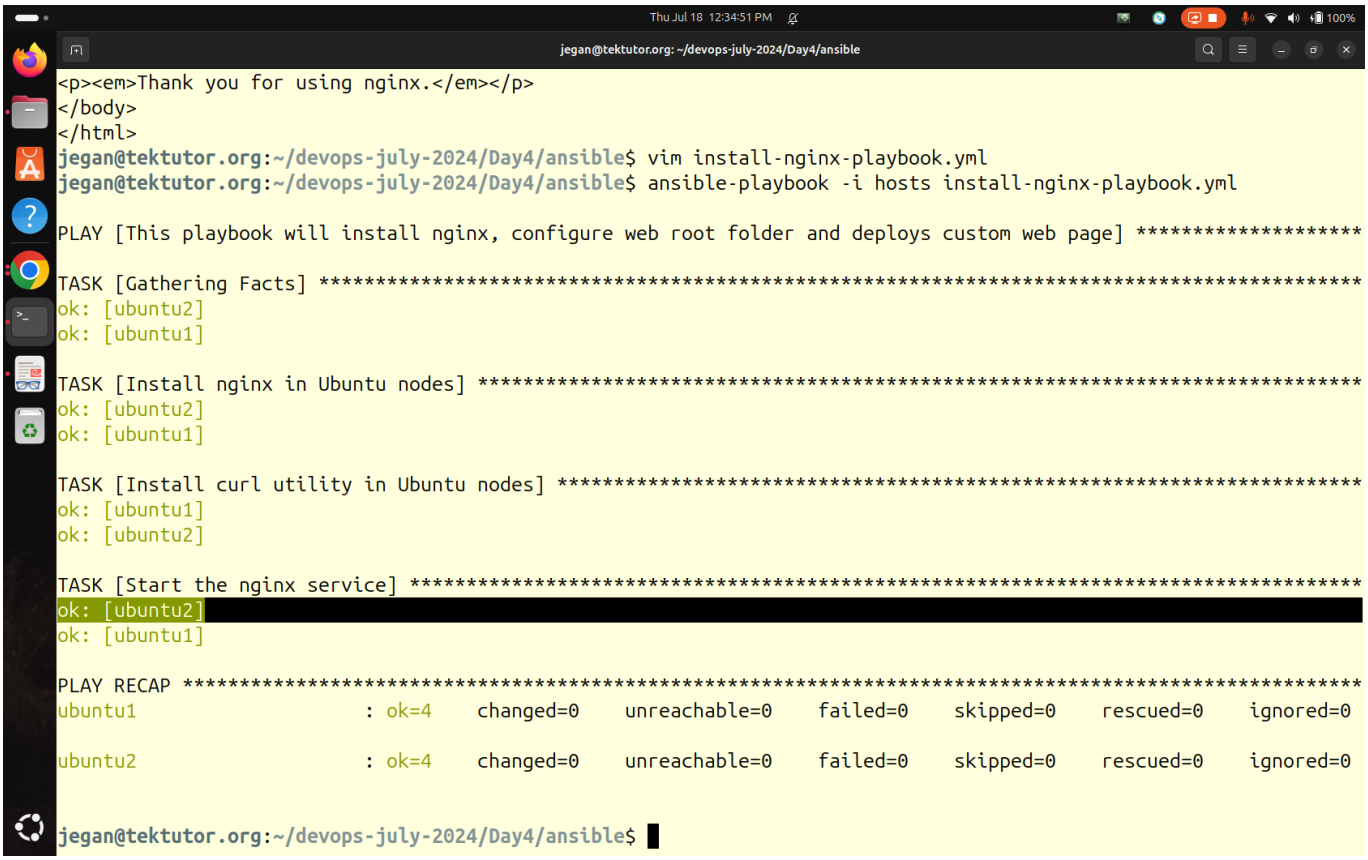
Now you may proceed with the below exercise

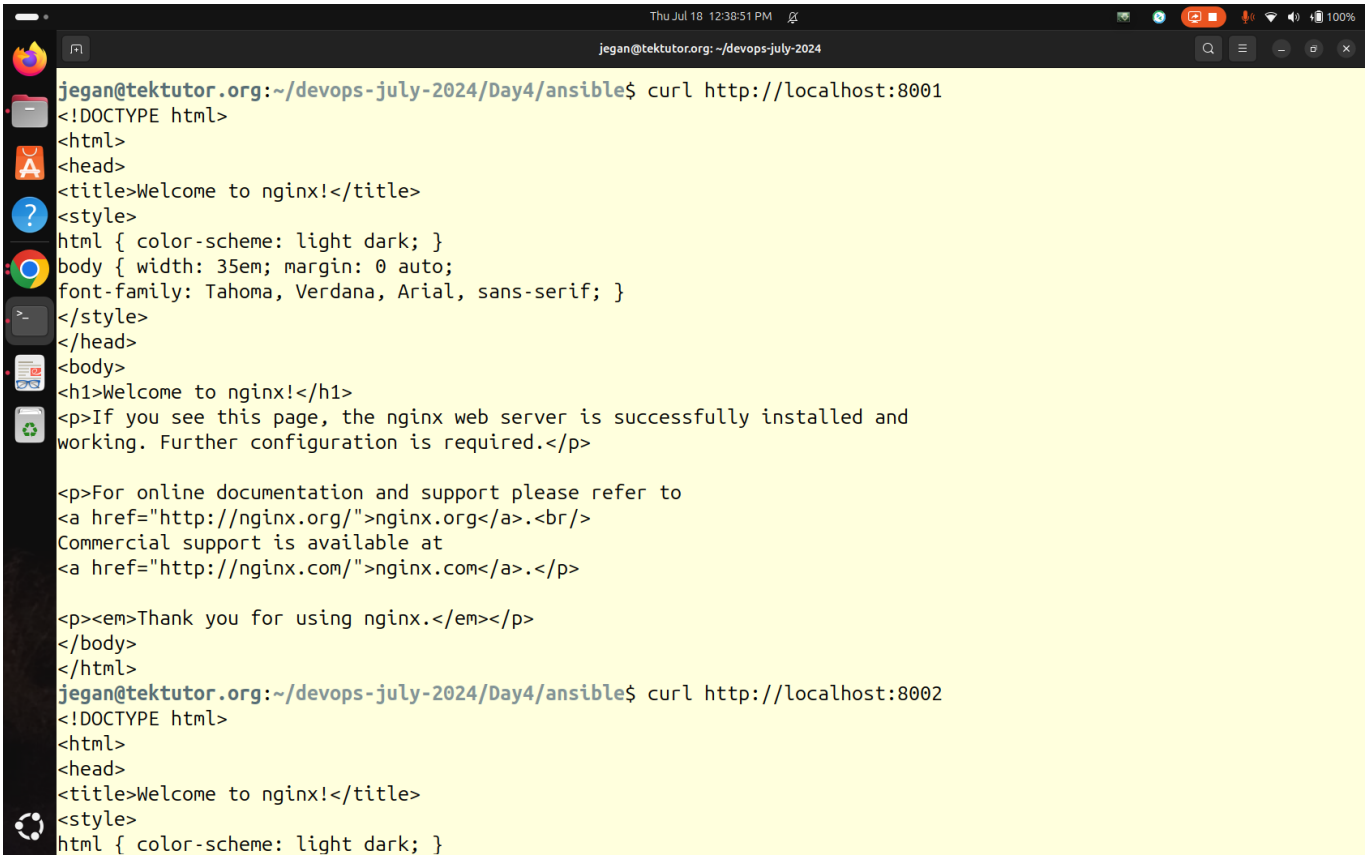
```
cd ~/devops-july-2024
git pull
cd Day4/ansible
cat install-nginx-playbook.yml
ansible-playbook -i hosts install-nginx-playbook.yml
curl http://localhost:8001
curl http://localhost:8002
```

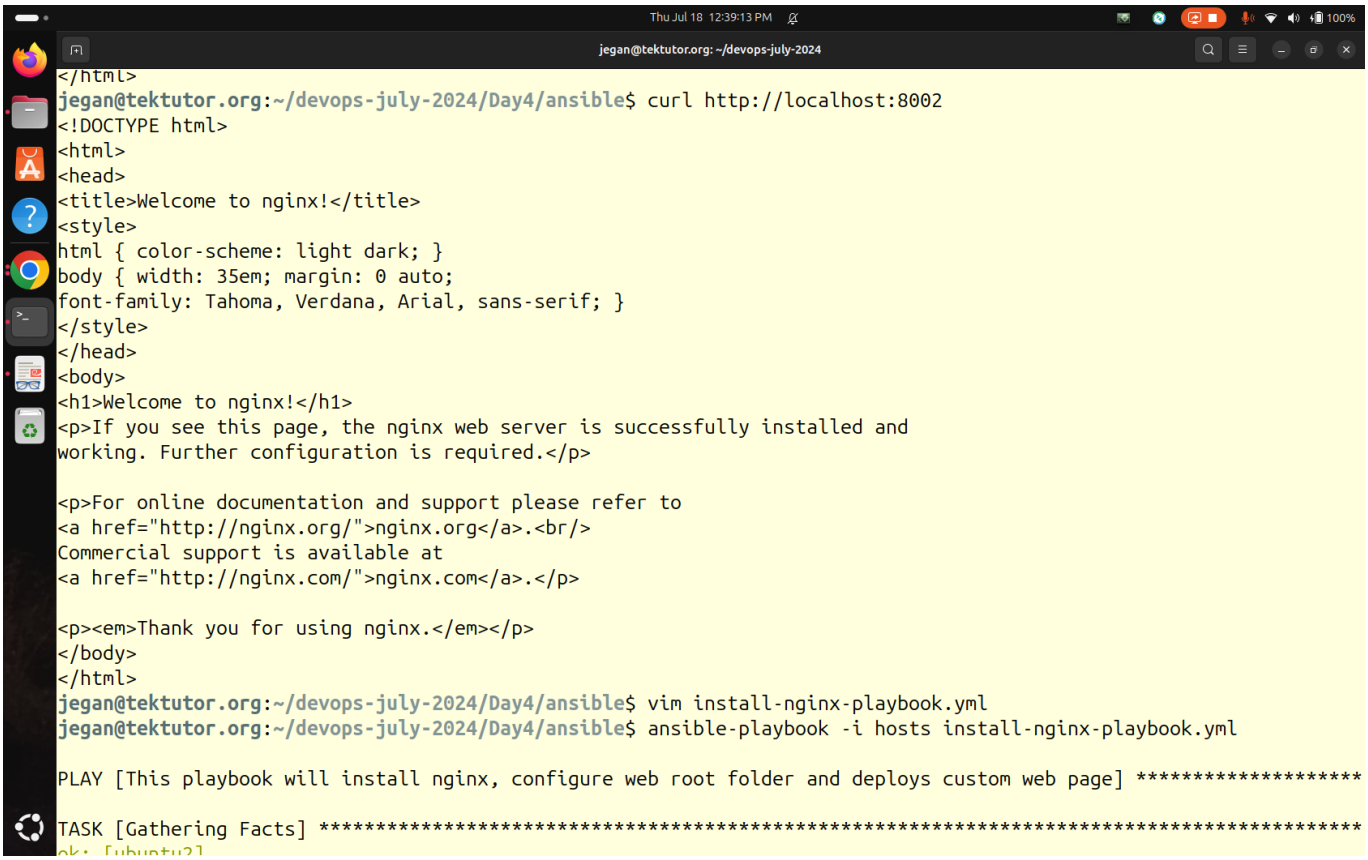
Expected output






Thu Jul 18 12:34:51 PM
jegan@tektutor.org: ~/devops-july-2024/Day4/ansible
Thank you for using nginx.
</body>
</html>
jegan@tektutor.org:~/devops-july-2024/Day4/ansible$ vim install-nginx-playbook.yml
jegan@tektutor.org:~/devops-july-2024/Day4/ansible$ ansible-playbook -i hosts install-nginx-playbook.yml
PLAY [This playbook will install nginx, configure web root folder and deploys custom web page] ********************
TASK [Gathering Facts] **
ok: [ubuntu2]
ok: [ubuntu1]
TASK [Install nginx in Ubuntu nodes] **
ok: [ubuntu2]
ok: [ubuntu1]
TASK [Install curl utility in Ubuntu nodes] ***
ok: [ubuntu1]
ok: [ubuntu2]
TASK [Start the nginx service] **
ok: [ubuntu2]
ok: [ubuntu1]
PLAY RECAP **
ubuntu1 : ok=4 changed=0 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0
ubuntu2 : ok=4 changed=0 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0
jegan@tektutor.org:~/devops-july-2024/Day4/ansible$


Thu Jul 18 12:38:51 PM
jegan@tektutor.org: ~/devops-july-2024
jegan@tektutor.org:~/devops-july-2024/Day4/ansible$ curl http://localhost:8001
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.
For online documentation and support please refer to
nginx.org.
Commercial support is available at
nginx.com.
Thank you for using nginx.
</body>
</html>
jegan@tektutor.org:~/devops-july-2024/Day4/ansible$ curl http://localhost:8002
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }



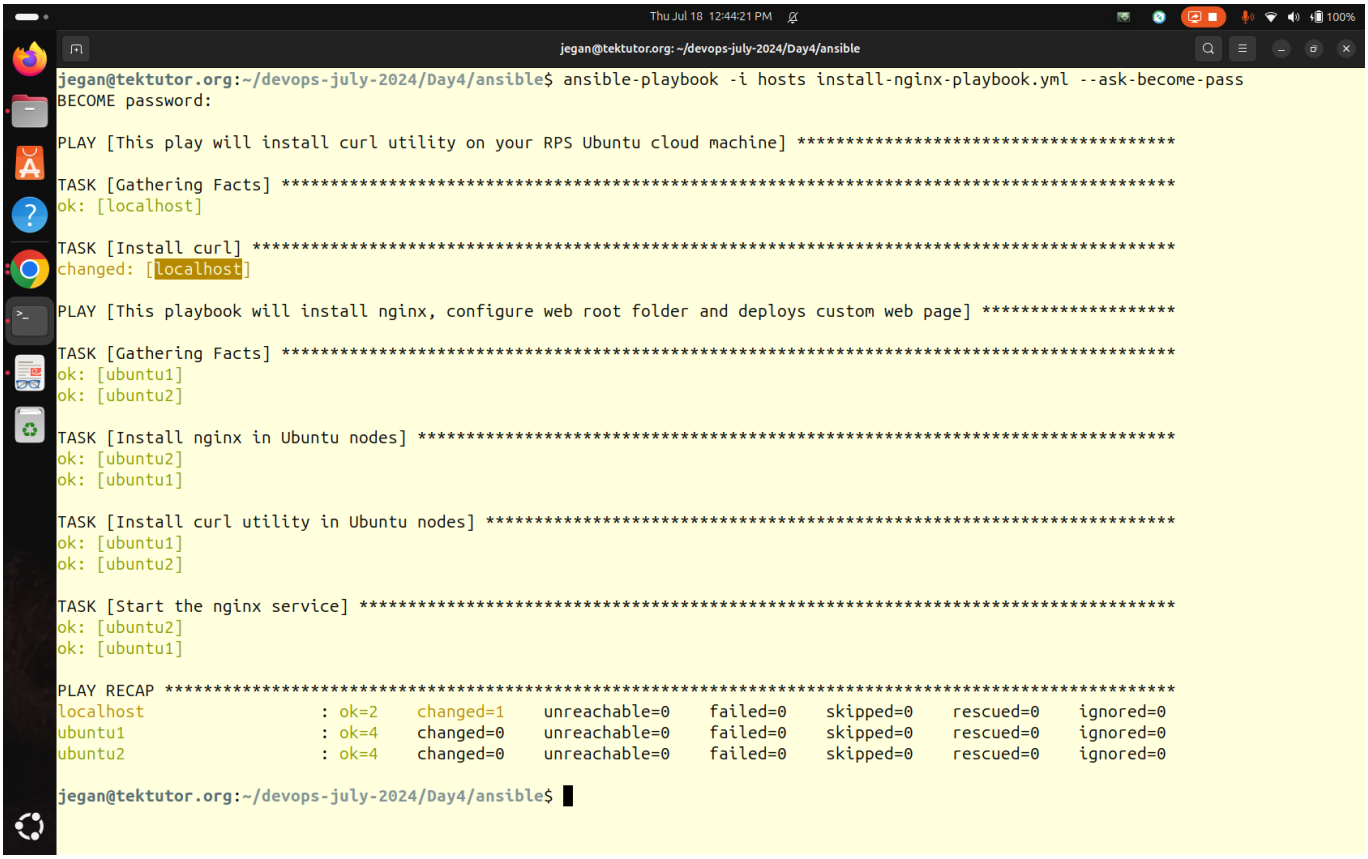
Things to note

```
- ansible apt module is idempotent, hence only the first time it will
install latest version of nginx, subsequent times we execute the playbook,
ansible will report in green color(meaning - it won't reinstall i.e ansible
will report - success with no change)
- shell module isn't idempotent, hence must be used only if no other option
is available/possible
```

Running the playbook with multiple play, the first play will install curl on your local machine

```
cd ~/devops-july-2024
git pull
cd Day4/ansible
cat install-nginx-playbook.yml
ansible-playbook -i hosts install-nginx-playbook.yml --ask-become-pass
```

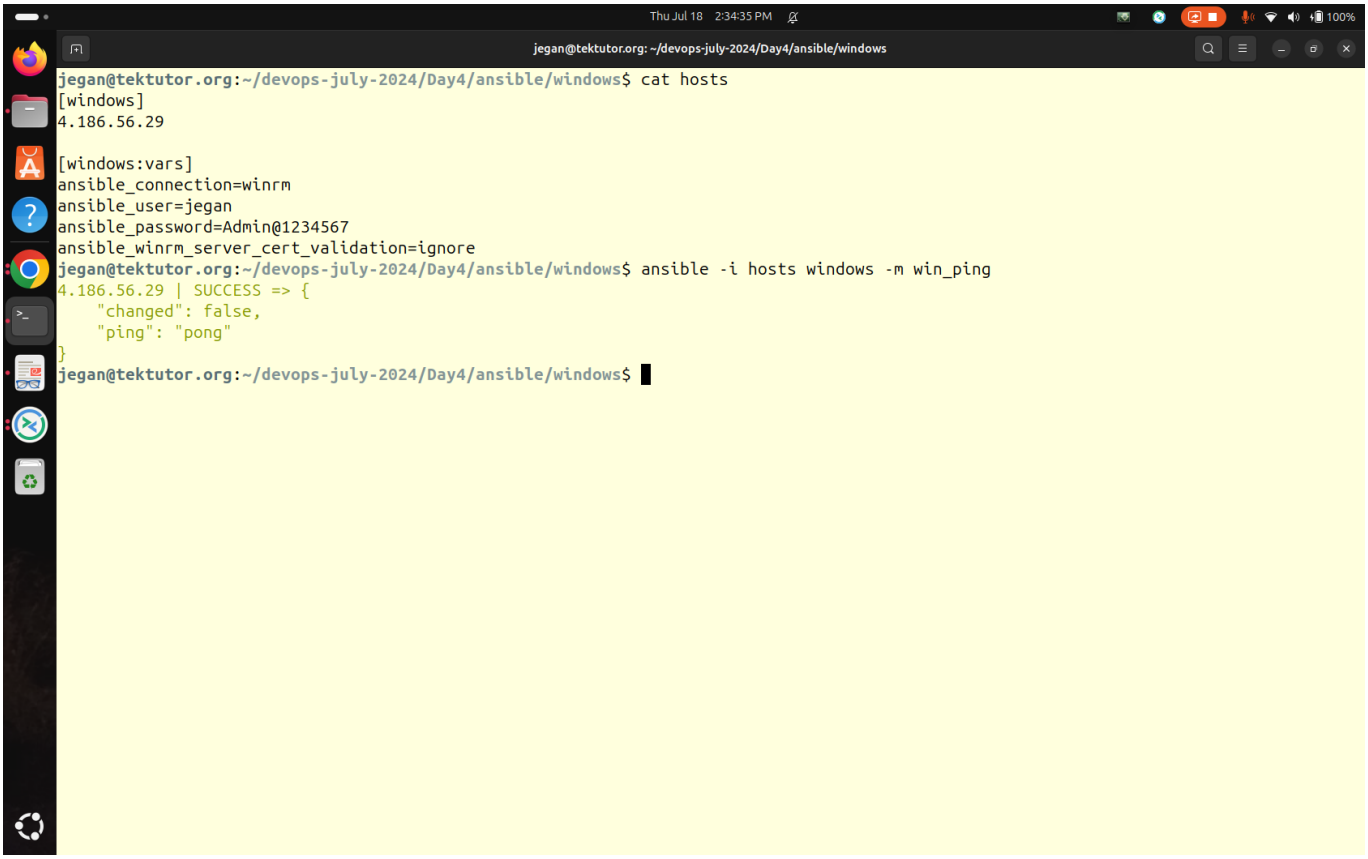
Expected output



Demo - Ansible ping a windows virtual machine from Azure portal

```
cd ~/devops-july-2024
git pull
cd Day4/ansible/windows
cat hosts
ansible -i hosts windows -m win_ping
```

Expected output

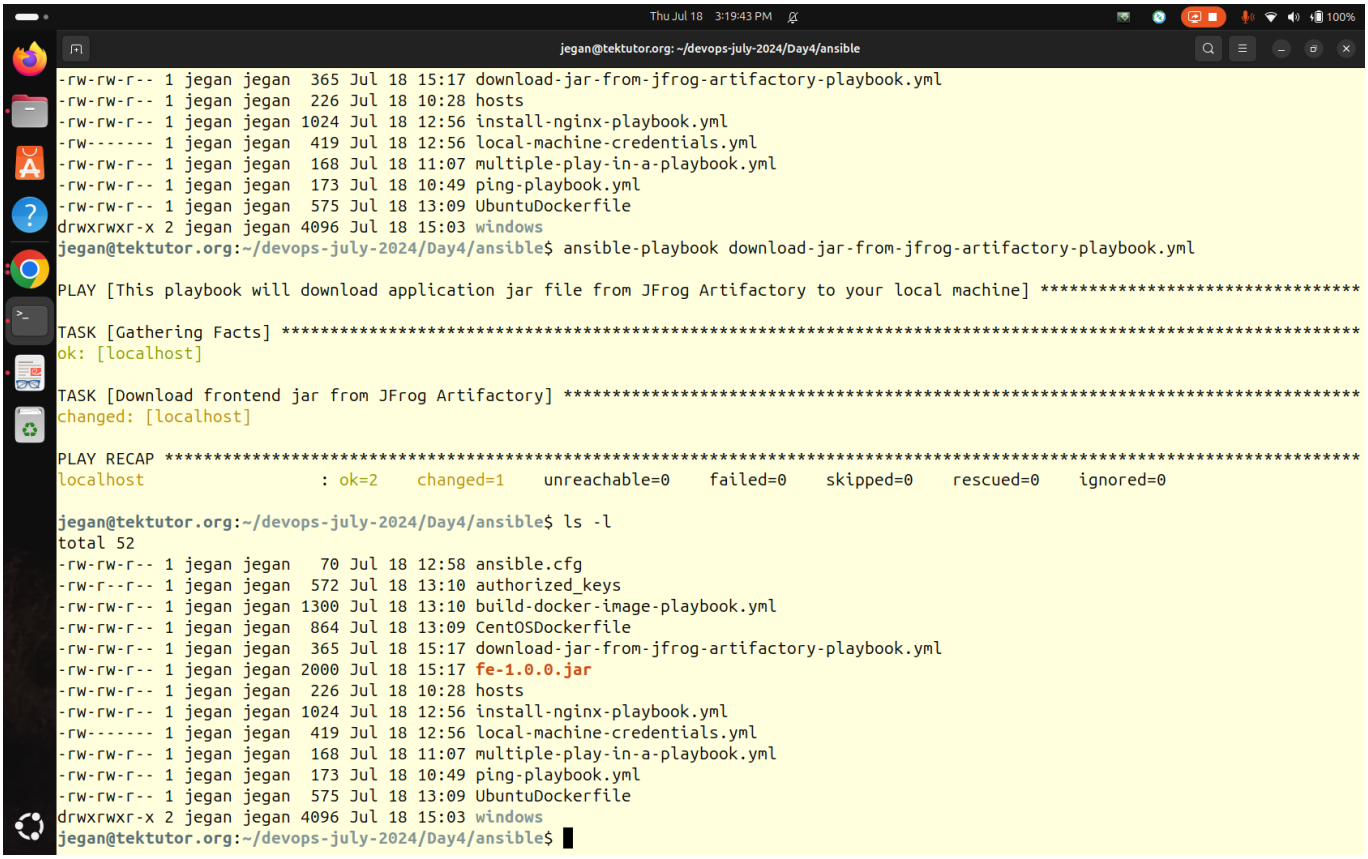


Lab - Downloading jar from JFrog Artifactory

Make sure your JFrog Artifactory container is running and it has the jar file we are downloading.

```
cd ~/devops-july-2024
git pull
cd Day4/ansible
cat download-jar-from-jfrog-artifactory-playbook.yml
ls -l
ansible-playbook download-jar-from-jfrog-artifactory-playbook.yml
ls -l
```

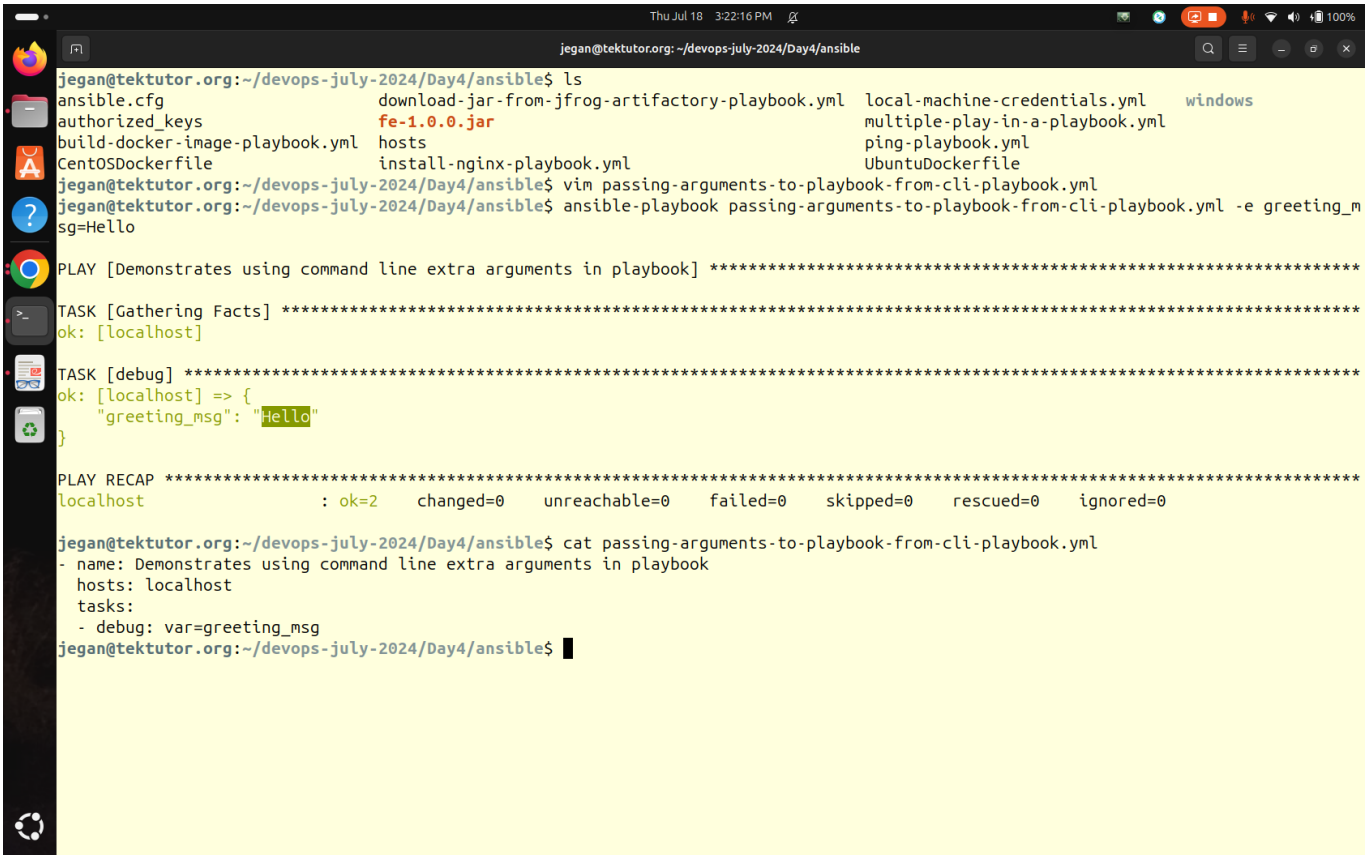
Expected output



Lab - Passing arguments to ansible-playbook from command line

```
ansible-playbook passing-arguments-to-playbook-from-cli-playbook.yml -e
greeting_msg=Hello

export GREETING_MSG="Hello!"
ansible-playbook passing-arguments-to-playbook-from-cli-playbook.yml -e
greeting_msg=$GREETING_MSG
```

Expected output

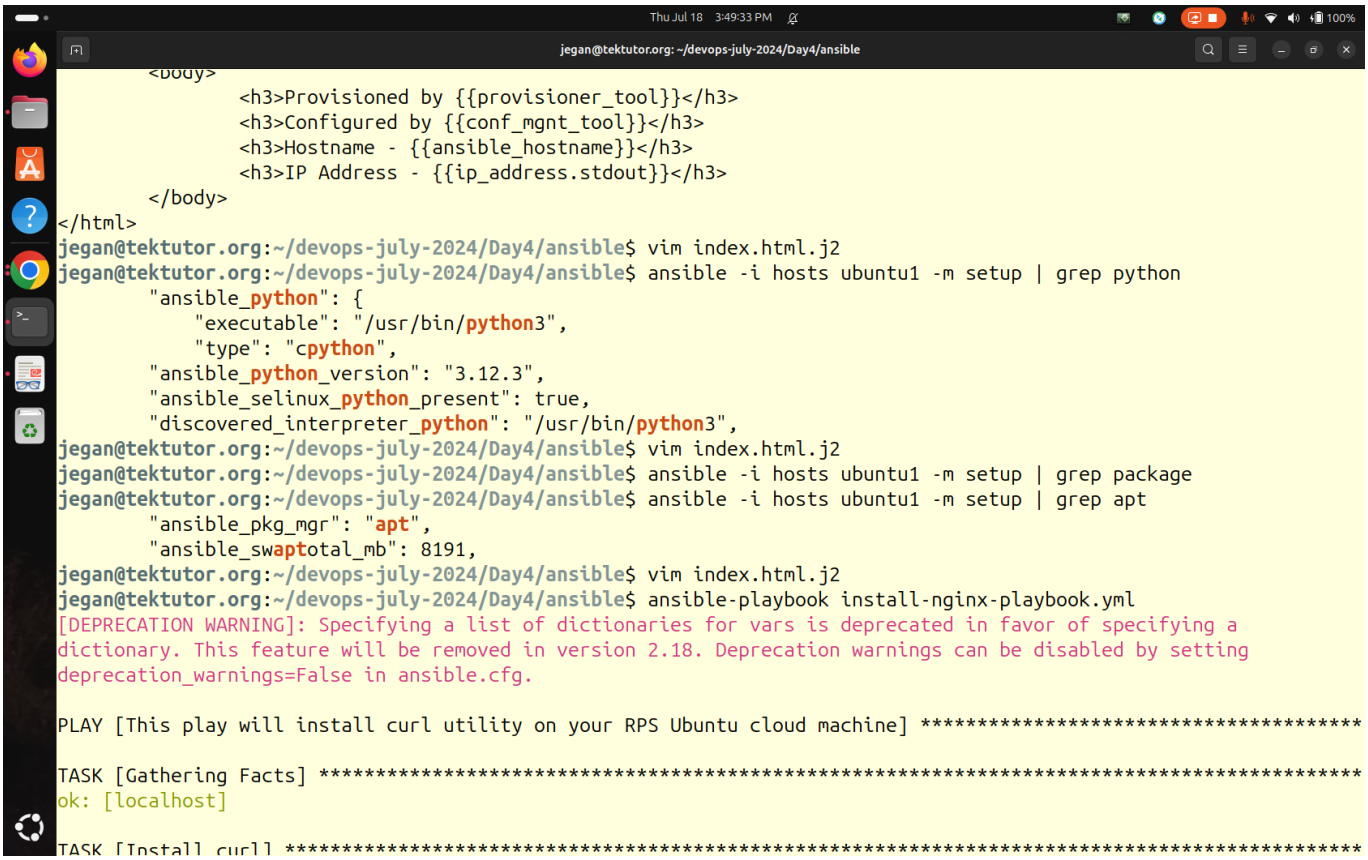


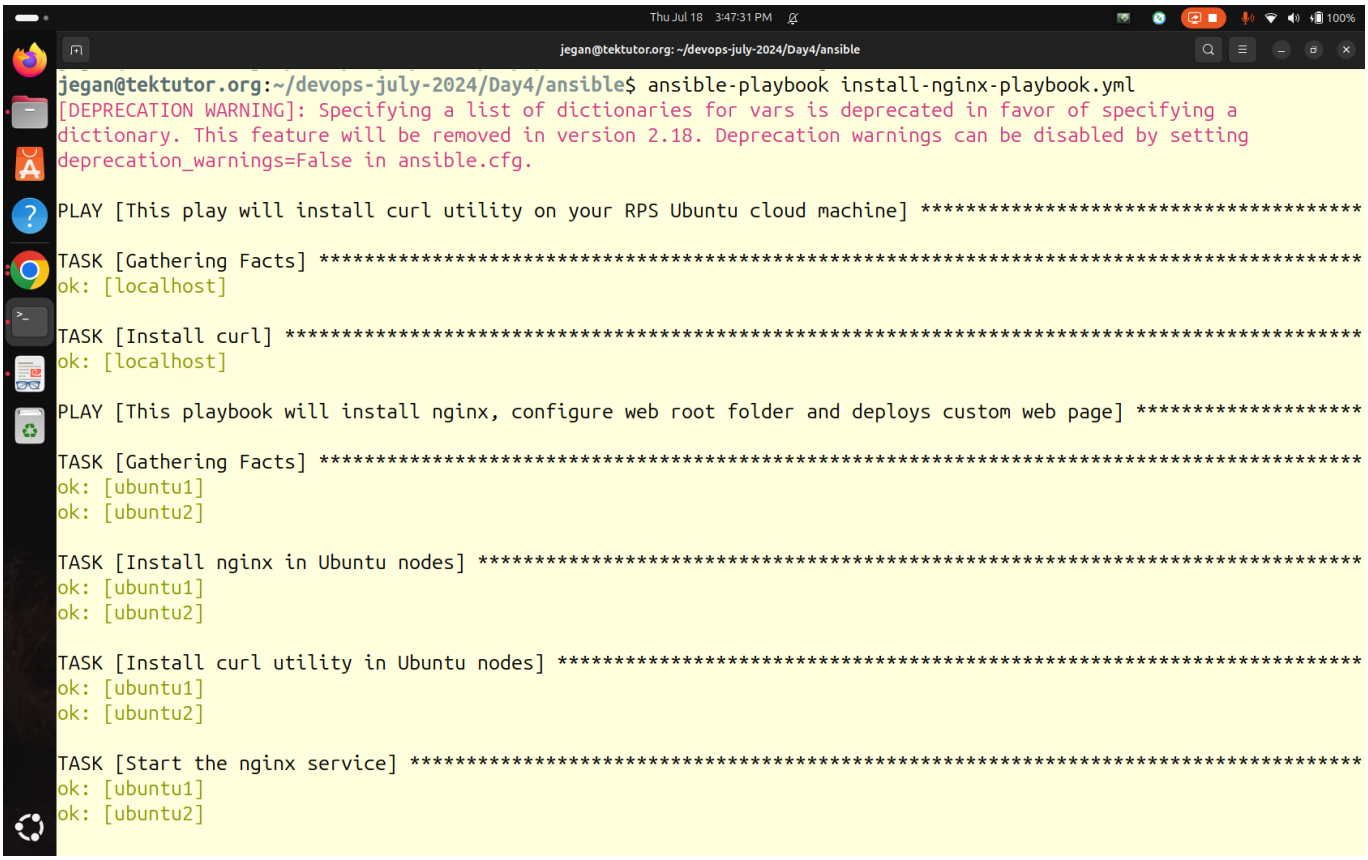
Lab - Using template module in playbooks

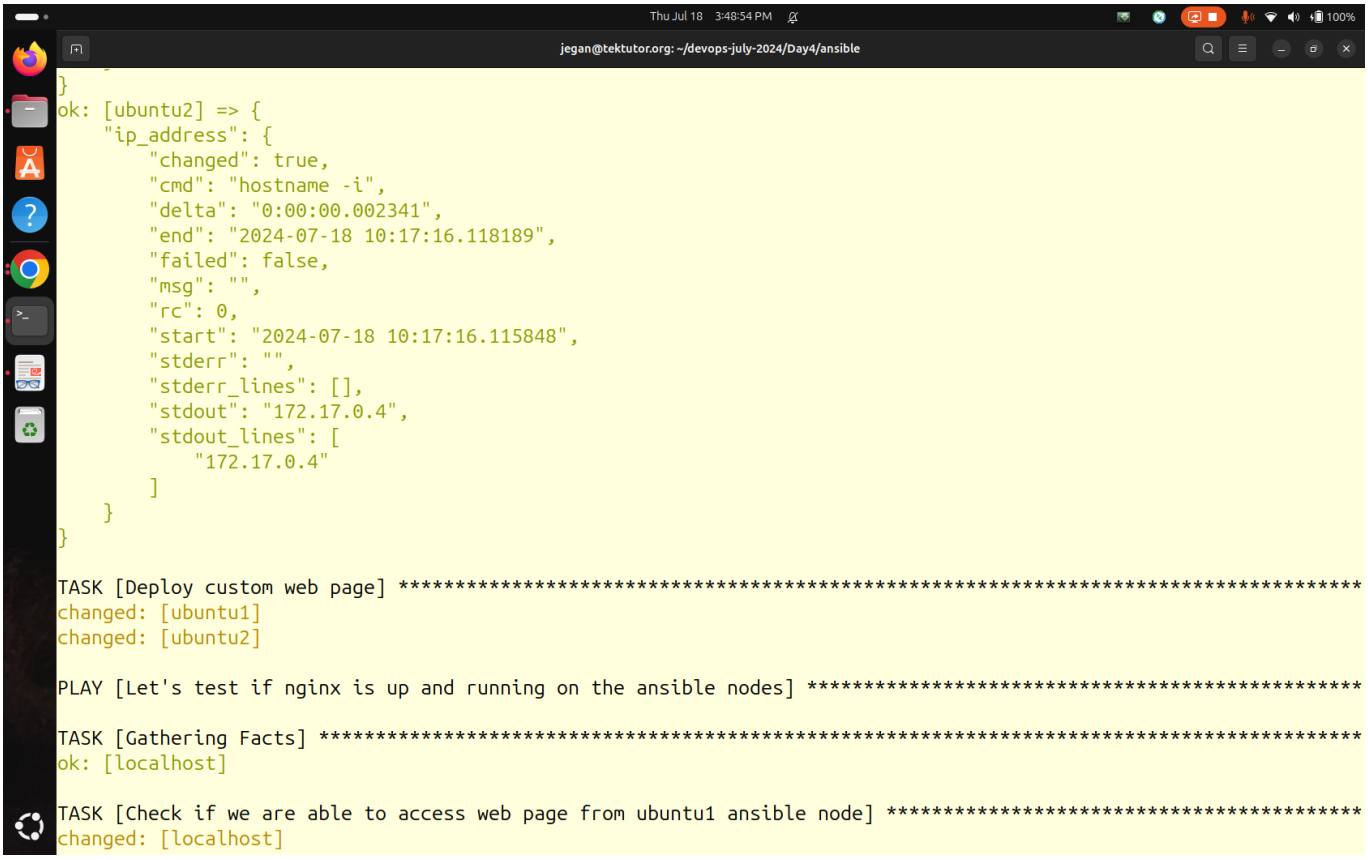
Create a file /home/rps/.my-vault-password with gedit and type 'root' without quotes and save the file.

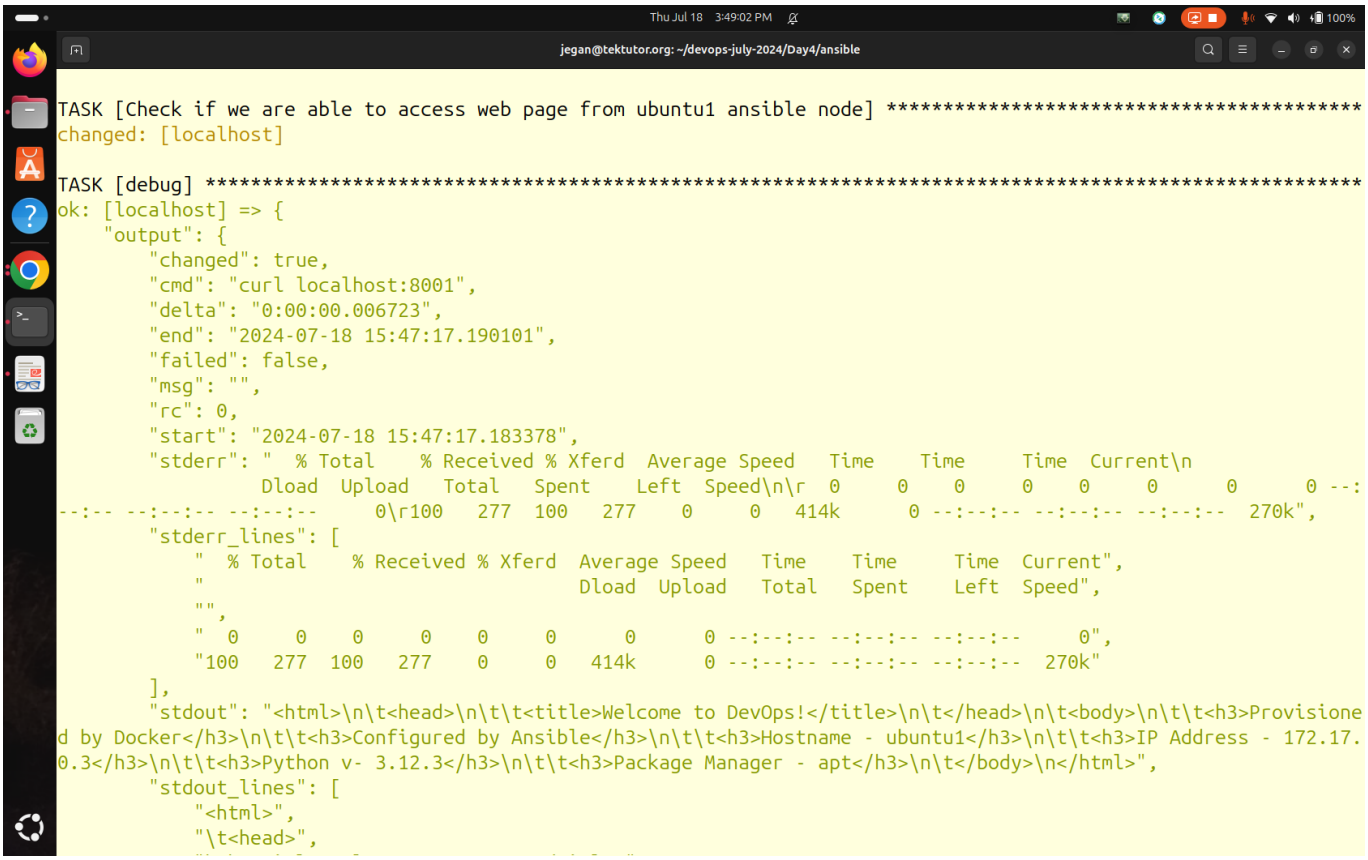
```
cd ~/devops-july-2024
git pull
cd Day4/ansible
ansible-playbook install-nginx-playbook.yml --ask-become-pass
```

Expected output

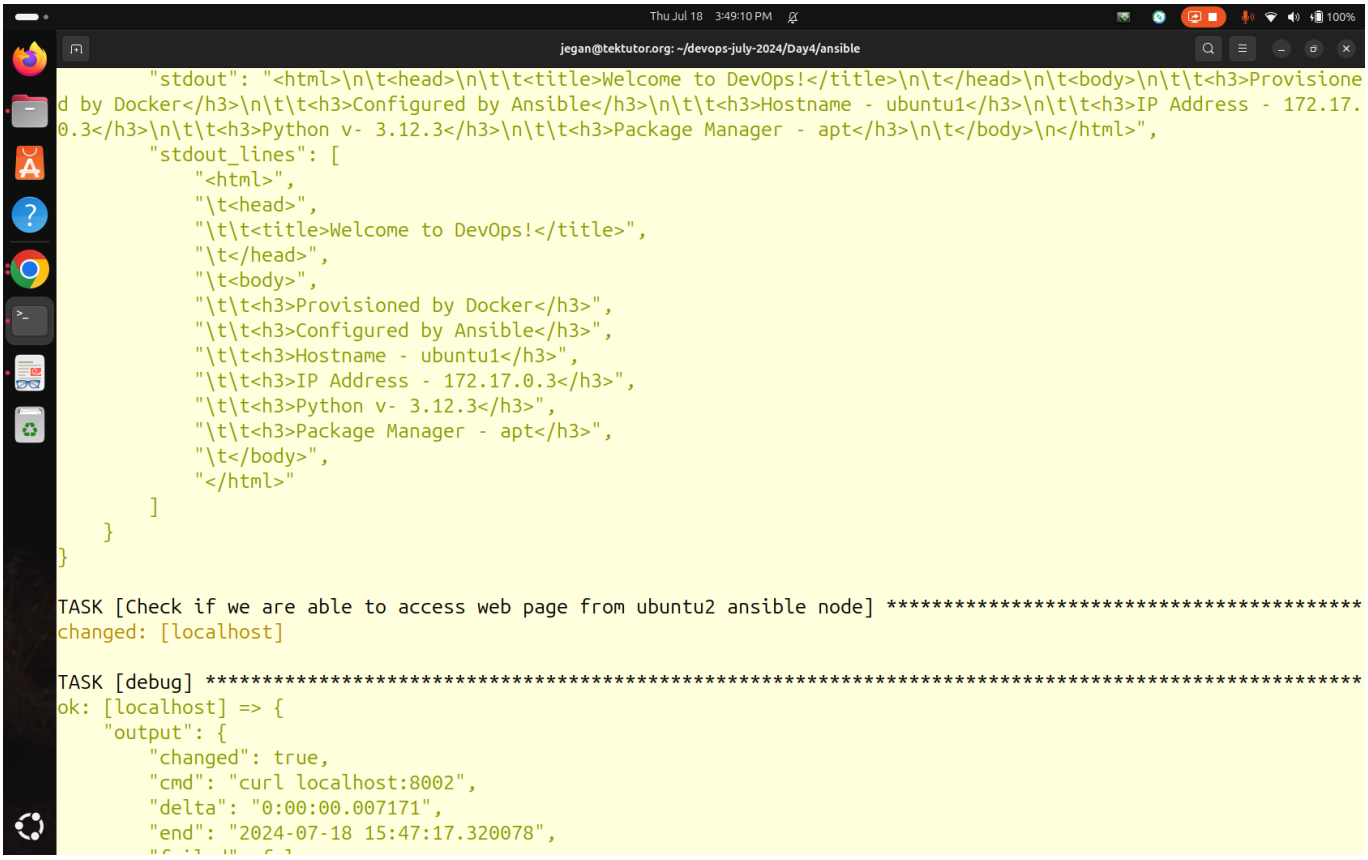



Thu Jul 18 3:47:31 PM
jegan@tektutor.org: ~/devops-july-2024/Day4/ansible
jegan@tektutor.org:~/devops-july-2024/Day4/ansible$ ansible-playbook install-nginx-playbook.yml
[DEPRECATION WARNING]: Specifying a list of dictionaries for vars is deprecated in favor of specifying a
dictionary. This feature will be removed in version 2.18. Deprecation warnings can be disabled by setting
deprecation_warnings=False in ansible.cfg.
PLAY [This play will install curl utility on your RPS Ubuntu cloud machine] ************************************
TASK [Gathering Facts] **
ok: [localhost]
TASK [Install curl] ***
ok: [localhost]
PLAY [This playbook will install nginx, configure web root folder and deploys custom web page] ******************
TASK [Gathering Facts] **
ok: [ubuntu1]
ok: [ubuntu2]
TASK [Install nginx in Ubuntu nodes] **
ok: [ubuntu1]
ok: [ubuntu2]
TASK [Install curl utility in Ubuntu nodes] ***
ok: [ubuntu1]
ok: [ubuntu2]
TASK [Start the nginx service] **
ok: [ubuntu1]
ok: [ubuntu2]

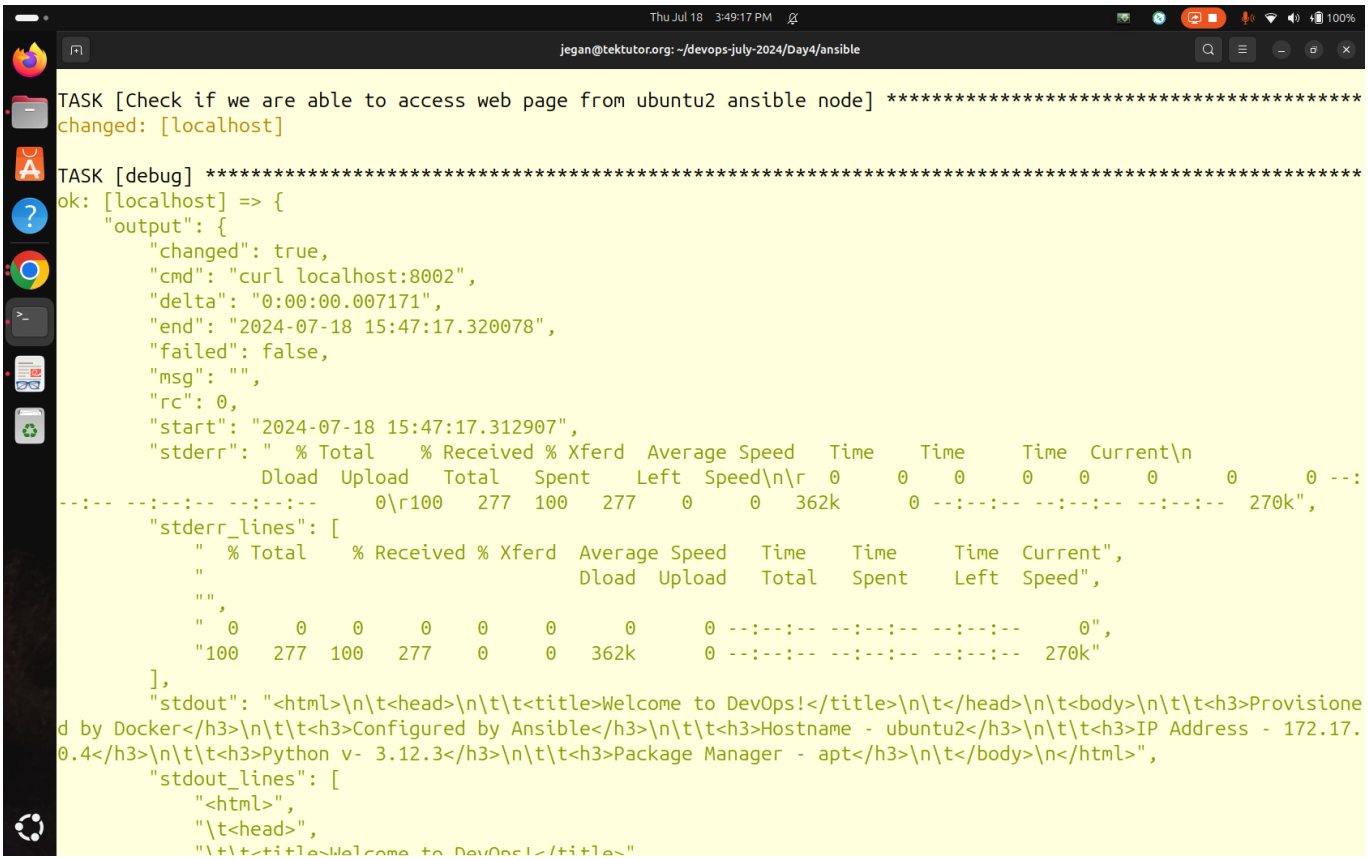

Thu Jul 18 3:48:54 PM
jegan@tektutor.org: ~/devops-july-2024/Day4/ansible
}
ok: [ubuntu2] => {
"ip_address": {
"changed": true,
"cmd": "hostname -i",
"delta": "0:00:00.002341",
"end": "2024-07-18 10:17:16.118189",
"failed": false,
"msg": "",
"rc": 0,
"start": "2024-07-18 10:17:16.115848",
"stderr": "",
"stderr_lines": [],
"stdout": "172.17.0.4",
"stdout_lines": [
"172.17.0.4"
]
}
}
TASK [Deploy custom web page] **
changed: [ubuntu1]
changed: [ubuntu2]
PLAY [Let's test if nginx is up and running on the ansible nodes] ***
TASK [Gathering Facts] **
ok: [localhost]
TASK [Check if we are able to access web page from ubuntu1 ansible node] ***********************************
changed: [localhost]




```
TASK [Check if we are able to access web page from ubuntu1 ansible node] ****************************************
changed: [localhost]

TASK [debug] ****************************************************************************************************
ok: [localhost] => {
    "output": {
        "changed": true,
        "cmd": "curl localhost:8001",
        "delta": "0:00:00.006723",
        "end": "2024-07-18 15:47:17.190101",
        "failed": false,
        "msg": "",
        "rc": 0,
        "start": "2024-07-18 15:47:17.183378",
        "stderr": "  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current\n                 Dload  Upload   Total   Spent    Left  Speed\n\r  0     0    0     0    0     0      0      0 --:
--:-- --:--:-- --:--:--     0\r100   277  100   277    0     0   414k      0 --:--:-- --:--:-- --:--:--  270k",
        "stderr_lines": [
            "  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current",
            "                                 Dload  Upload   Total   Spent    Left  Speed",
            "",
            "  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0",
            "100   277  100   277    0     0   414k      0 --:--:-- --:--:-- --:--:--  270k"
        ],
        "stdout": "<html>\n\t<head>\n\t\t<title>Welcome to DevOps!</title>\n\t</head>\n\t<body>\n\t\t<h3>Provisione
d by Docker</h3>\n\t\t<h3>Configured by Ansible</h3>\n\t\t<h3>Hostname - ubuntu1</h3>\n\t\t<h3>IP Address - 172.17.
0.3</h3>\n\t\t<h3>Python v- 3.12.3</h3>\n\t\t<h3>Package Manager - apt</h3>\n\t</body>\n</html>",
        "stdout_lines": [
            "<html>",
            "\t<head>",
```




```
        "stdout": "<html>\n\t<head>\n\t\t<title>Welcome to DevOps!</title>\n\t</head>\n\t<body>\n\t\t<h3>Provisione
d by Docker</h3>\n\t\t<h3>Configured by Ansible</h3>\n\t\t<h3>Hostname - ubuntu1</h3>\n\t\t<h3>IP Address - 172.17.
0.3</h3>\n\t\t<h3>Python v- 3.12.3</h3>\n\t\t<h3>Package Manager - apt</h3>\n\t</body>\n</html>",
        "stdout_lines": [
            "<html>",
            "\t<head>",
            "\t\t<title>Welcome to DevOps!</title>",
            "\t</head>",
            "\t<body>",
            "\t\t<h3>Provisioned by Docker</h3>",
            "\t\t<h3>Configured by Ansible</h3>",
            "\t\t<h3>Hostname - ubuntu1</h3>",
            "\t\t<h3>IP Address - 172.17.0.3</h3>",
            "\t\t<h3>Python v- 3.12.3</h3>",
            "\t\t<h3>Package Manager - apt</h3>",
            "\t</body>",
            "</html>"
        ]
    }
}

TASK [Check if we are able to access web page from ubuntu2 ansible node] ****************************************
changed: [localhost]

TASK [debug] ****************************************************************************************************
ok: [localhost] => {
    "output": {
        "changed": true,
        "cmd": "curl localhost:8002",
        "delta": "0:00:00.007171",
        "end": "2024-07-18 15:47:17.320078",
```



Info - SCRUM - Daily stand-up meeting

```
- is an inspect and adapt meeting
- in other words, it is fail-fast meeting
- the team inspects the yesterday's plan against the real status as on
yesterday
```

```
- if the team finds a deviation, if required the plan must be updated
```

Info - DevOps

```
- Developers
  - they need to automate unit/integrating testing using Test Frameworks
    - developers are expected to learn little bit of QA skills
  - Test Frameworks
    - Developers should follow TDD(Test Driven Development)
    - Java
      - JUnit/TestNg/Mockito/PowerMock/EasyMock/JMock, etc
    - C/C++
      - CUnit, CppUnit, Google Test/Mock
    - JavaScript ( Angular, NodeJS )
      - Jasmine, Karma
    - Python
      - PyTest
    - C#
      - NUnit/MSTest/Moq
- QA
  - they need to automate e2e functionality test, API Test, Stress/Load
Test, Regression Test, Sanity Check, Smoke Test
  - the test automation expects to convert the testing effort into source
code that can be pushed to Version control
  - QA folks are expected to learn little of development skills
  - Frameworks
    - Behaviour Driven Development Frameworks (BDD)
    - Cucumber, Specflow, Selenium, Raft
- System Administrators
  - they need to automate OS installation, provisioning, software
installations, user management, etc
  - Provisioning
    - System Adminstrators are expected to use Infrastructure as a Code
Tools like Cloudformation, Terraform, Docker, Vagrant, etc.,
    - System Administrators are expected to use Configuration Management
Tools like Ansible, Puppet/Chef, etc to install softwares on an existing
Virtual Machine or a OS on OnPrem environment(Private cloud), public cloud
and Hybrid Cloud
    - System Administrators also has to know coding
```

Info - Continuous Integration (CI)

```
- the code developed by each developer is continuously integrated in the
dev branch
- Jenkins or CI Build Server grabs the latest code and it is going run the
build along with unit and integration test cases
- If any test case fails, the build will fail, all the test cases are
passing then the build will pass
```

Info - Continuous Deployment (CD)

```
- the CI certified application binaries are automatically deployed into QA
environment for further QA automated test execution
```

Info - Continuous Delivery (CD)

```
- the QA certified release(application binaries) are deployed into
customer's environment which is close to production environment ( pre-prod
) environment
- the Customer will verify the functionality and if the customer found the
application is working as expected and found to be stable, they can decide
if they can make it live in production environment
```

Info - Jenkins Overview

```
- Jenkins is a Continuous Integration Build Server
- it was developed in Java by Kohsuke Kawaguchi while he was working for
Sun Microsystems
- the initial product developed was called Hudson
- it is an opensource tool
- Before leaving Oracle, Kohsuke Kawaguchi created a branch of Hudson
source code called Jenkins
- Kohsuke founded a company called Cloudbees
- Today Hudson and Jenkins shares lot of common code, but Jenkins has
become very popular than Hudson
- Jenkins has over 10000+ active opensource contributors
- there are 2 variants of Jenkins
  1. Jenkins open-source
  2. Cloudbees ( Enterprise edition of Jenkins )
- it supports creating CI build for any programming language
```

Info - Jenkins alternates

```
- Team City
- Bamboo
- Microsoft's Team Foundation Server (TFS)
- CircleCI
```

Info - Just in case you need to reset Jenkins Administrator password

```
https://www.jenkins.io/doc/book/system-administration/admin-password-reset-instructions/
```

Lab - Launching Jenkins from terminal

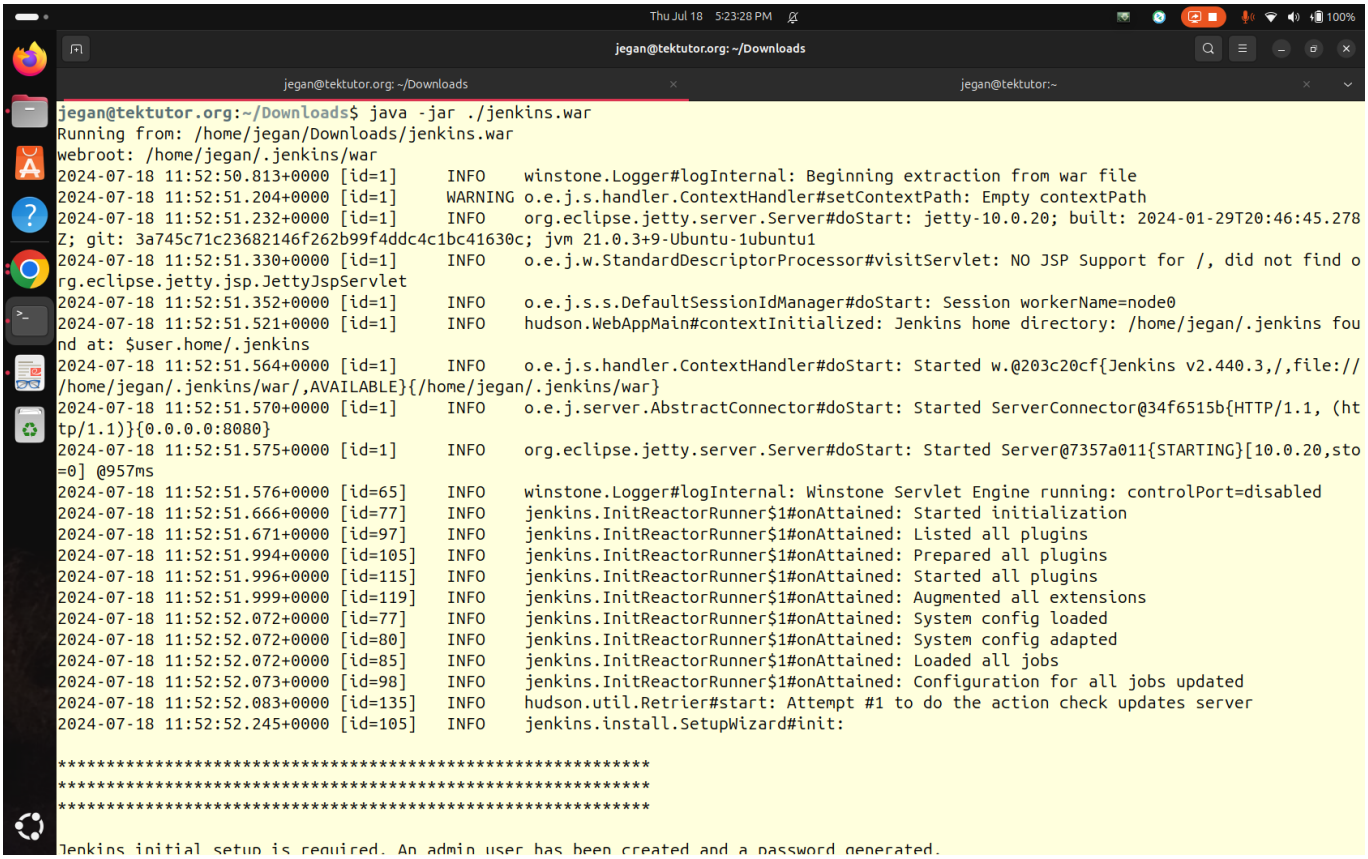
Verify if already any application is using the port 8080

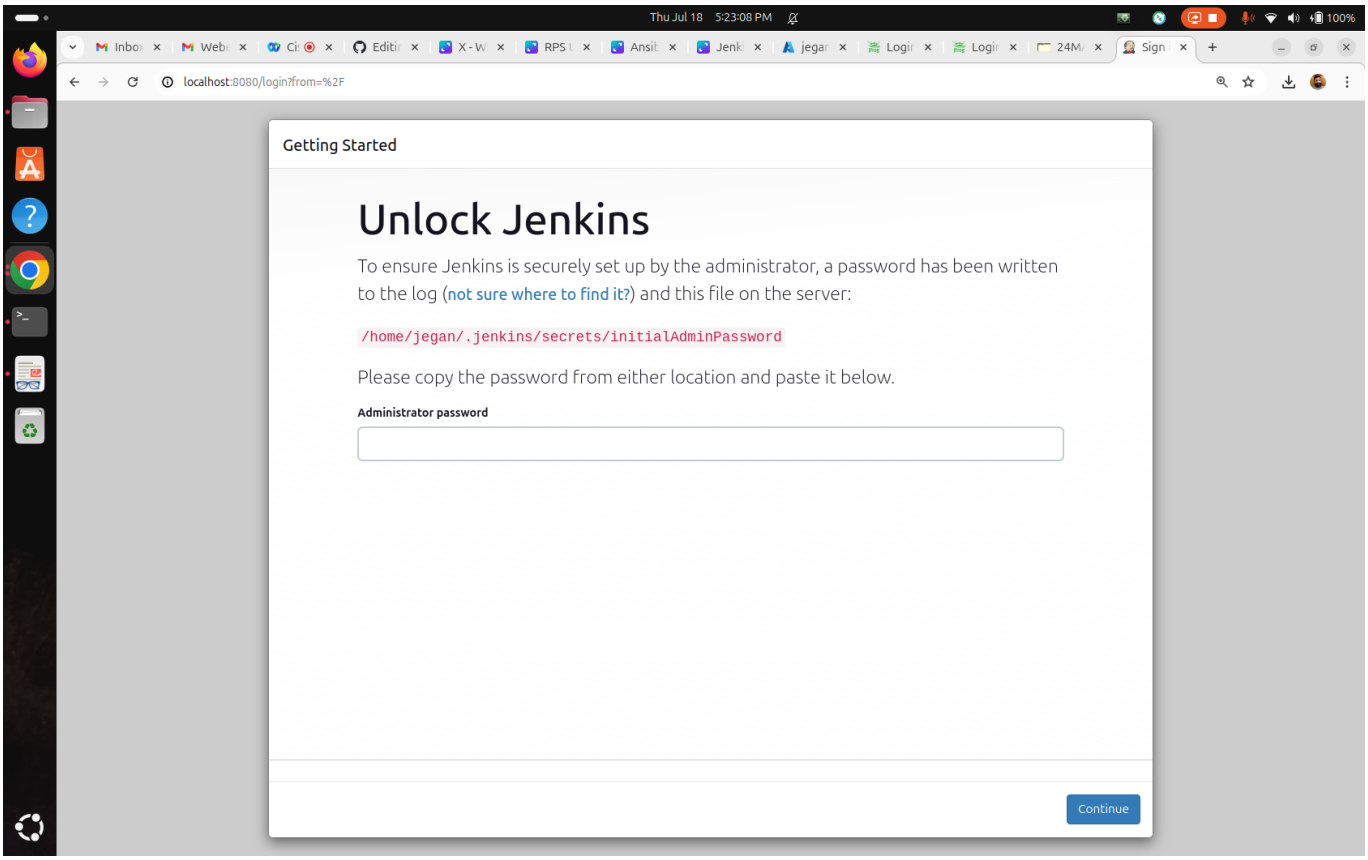
```

```

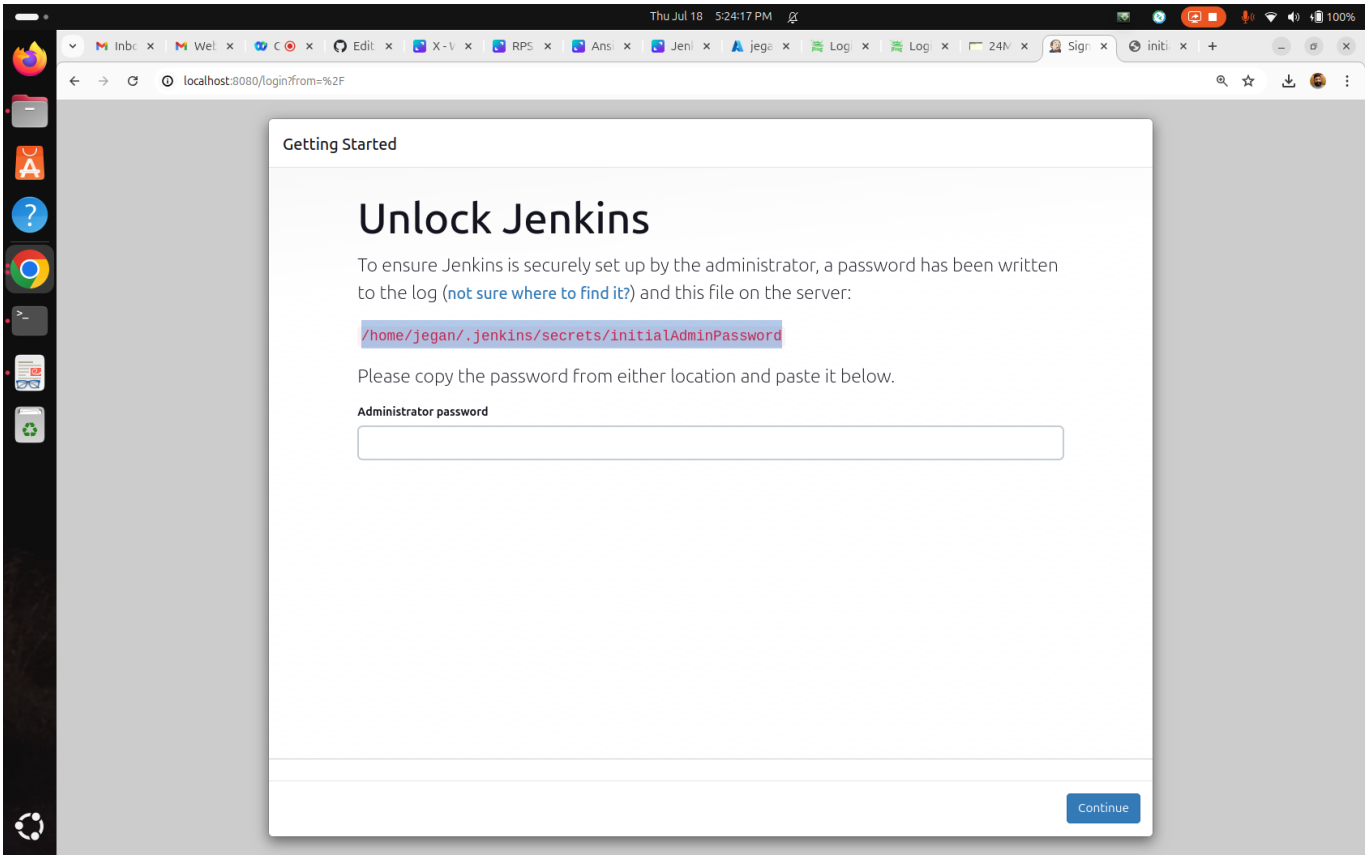
The below command is a blocking command, hence don't press Ctrl+C, you may open new terminal tab to run other commands

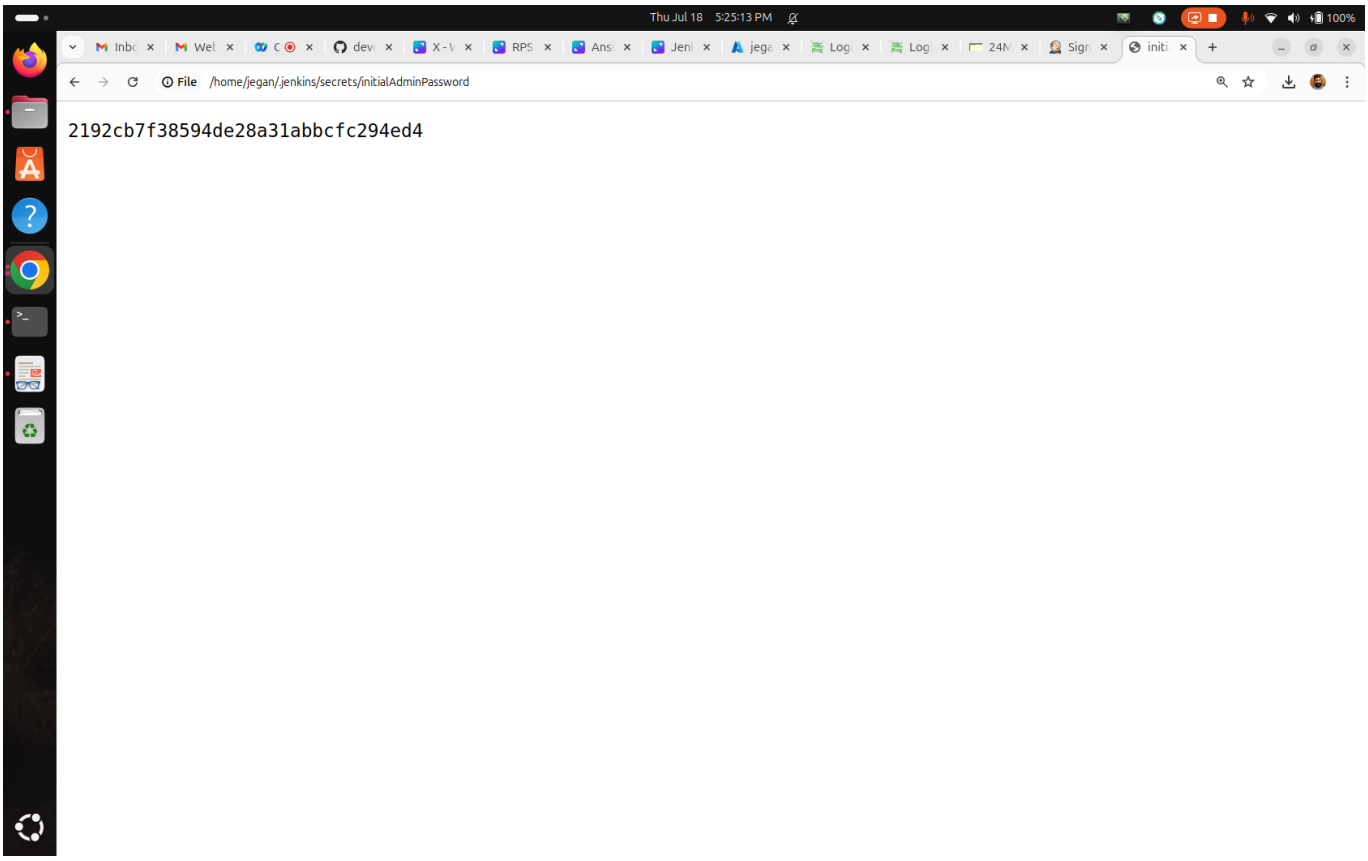
```
cd ~/Downloads
java -jar ./jenkins.jar
```

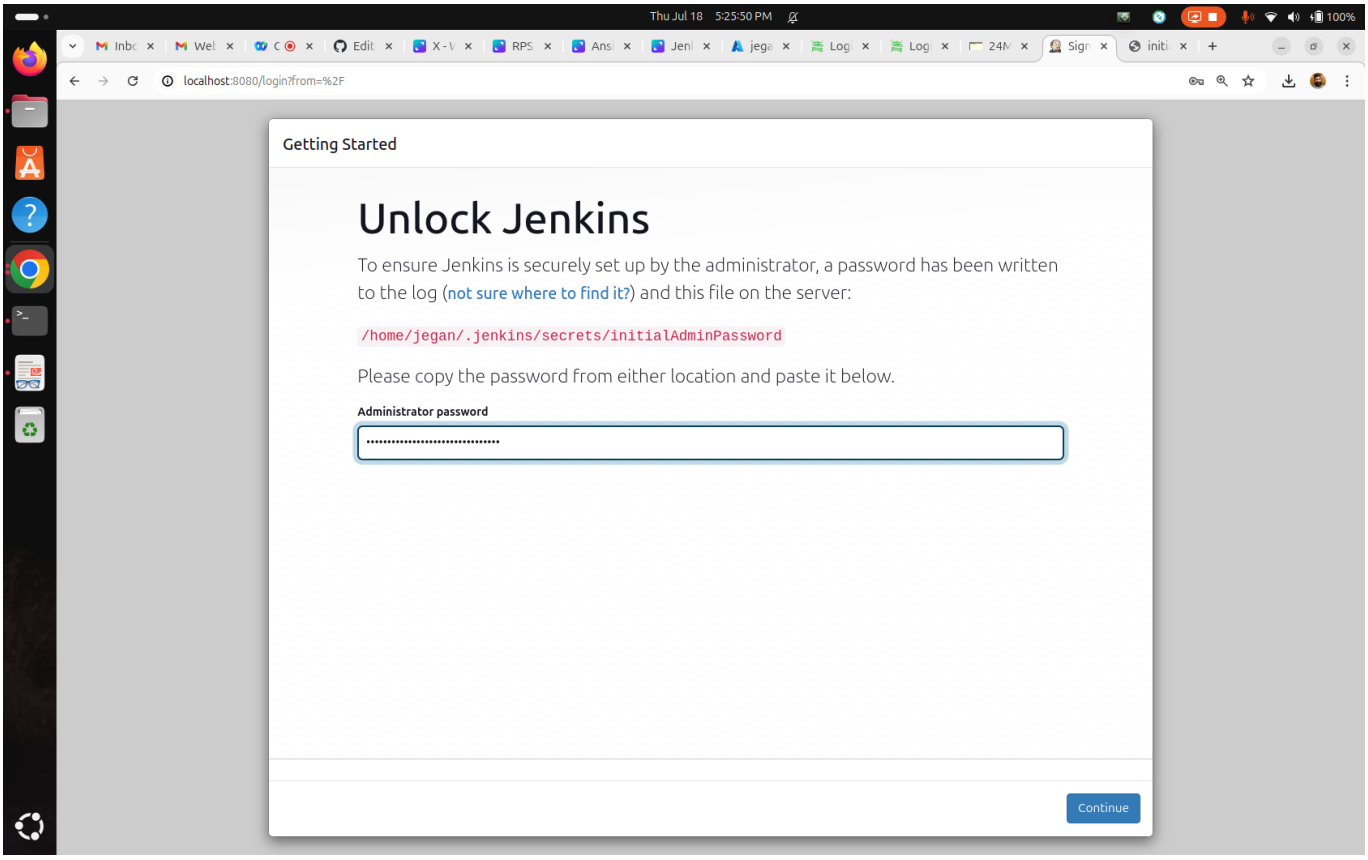
Expected output

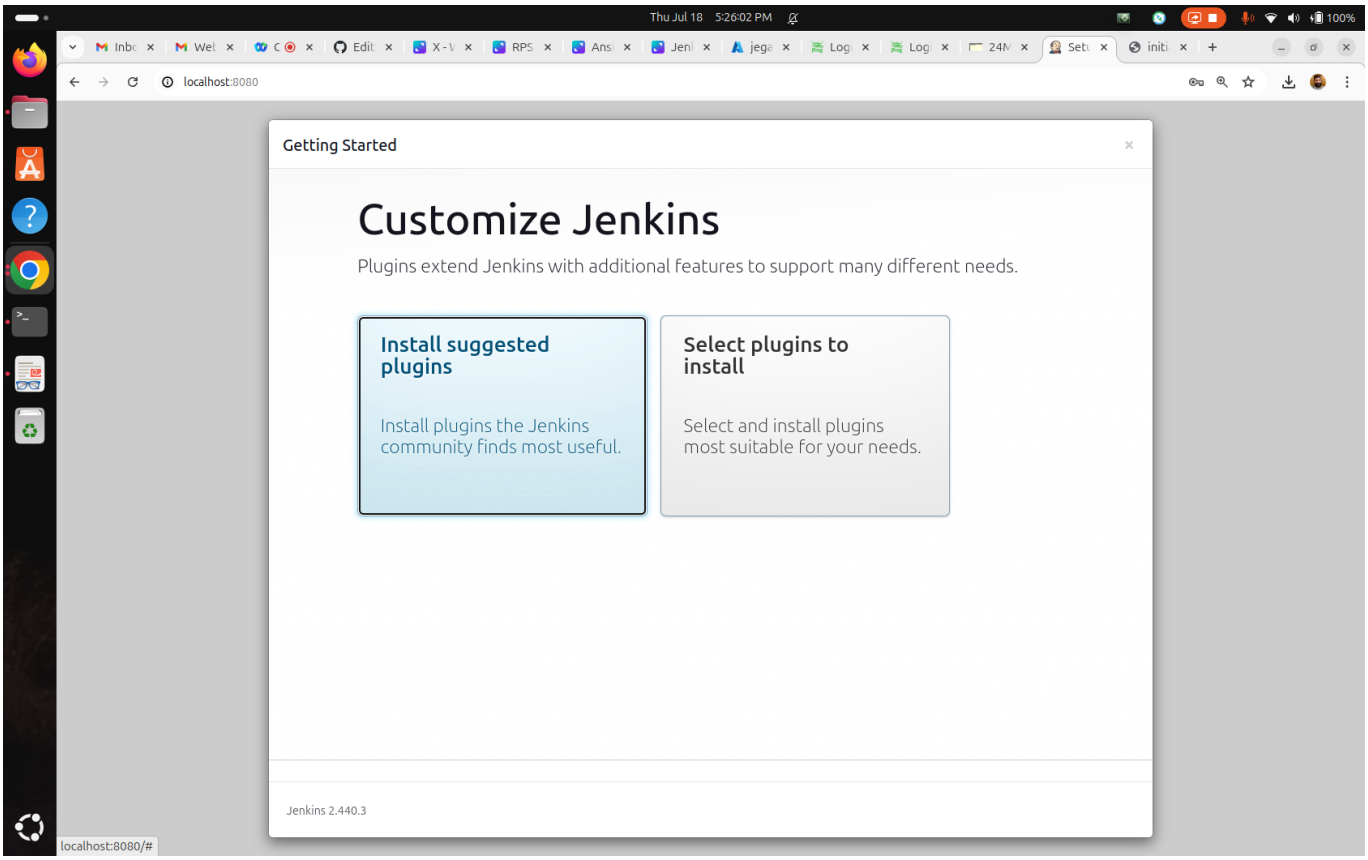



Thu Jul 18 5:23:08 PM
localhost:8080/login?from=%2F
Getting Started
Unlock Jenkins
To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:
/home/jegan/.jenkins/secrets/initialAdminPassword
Please copy the password from either location and paste it below.
Administrator password
Continue

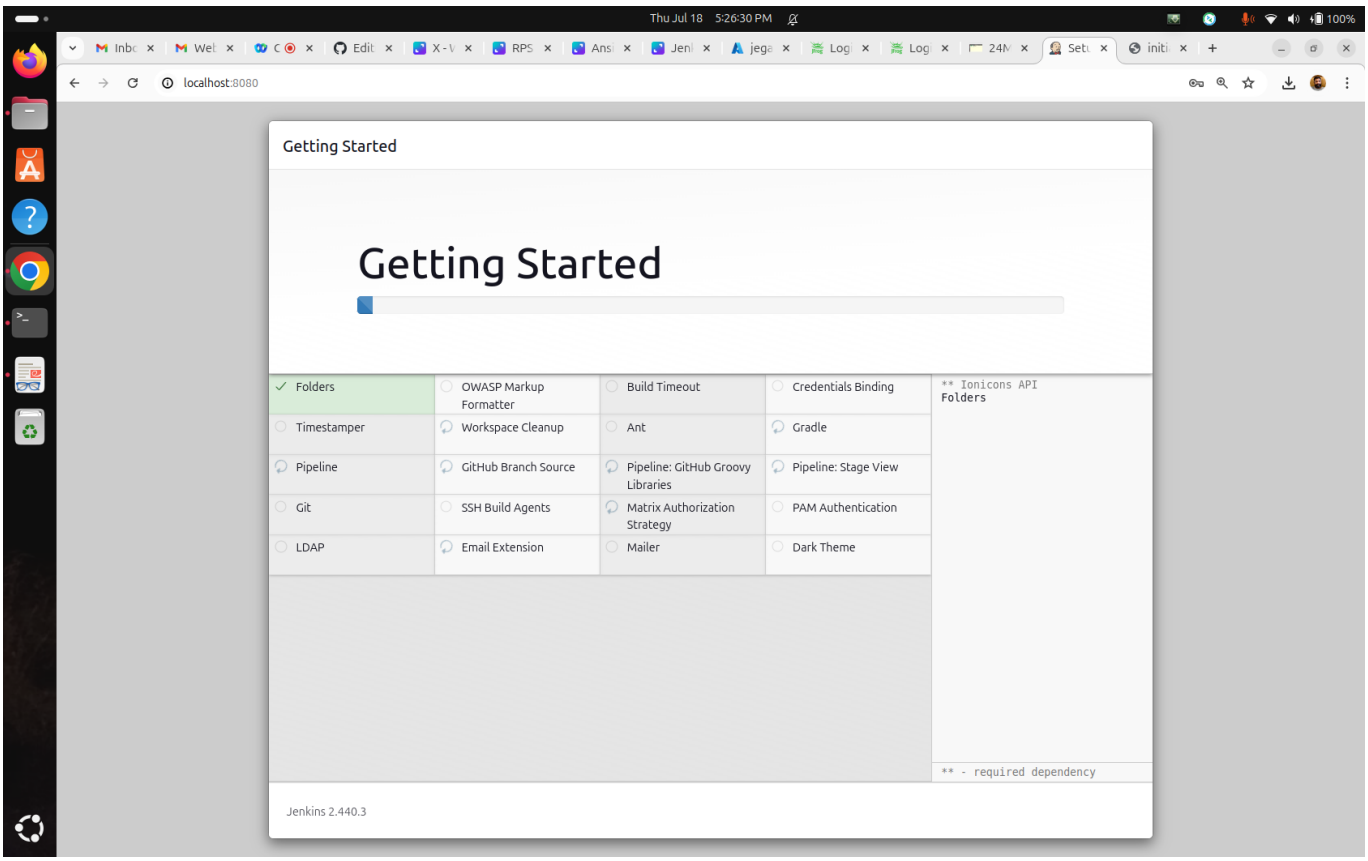

Thu Jul 18 5:24:17 PM
localhost:8080/login?from=%2F
Getting Started
Unlock Jenkins
To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:
/home/jegan/.jenkins/secrets/initialAdminPassword
Please copy the password from either location and paste it below.
Administrator password
Continue

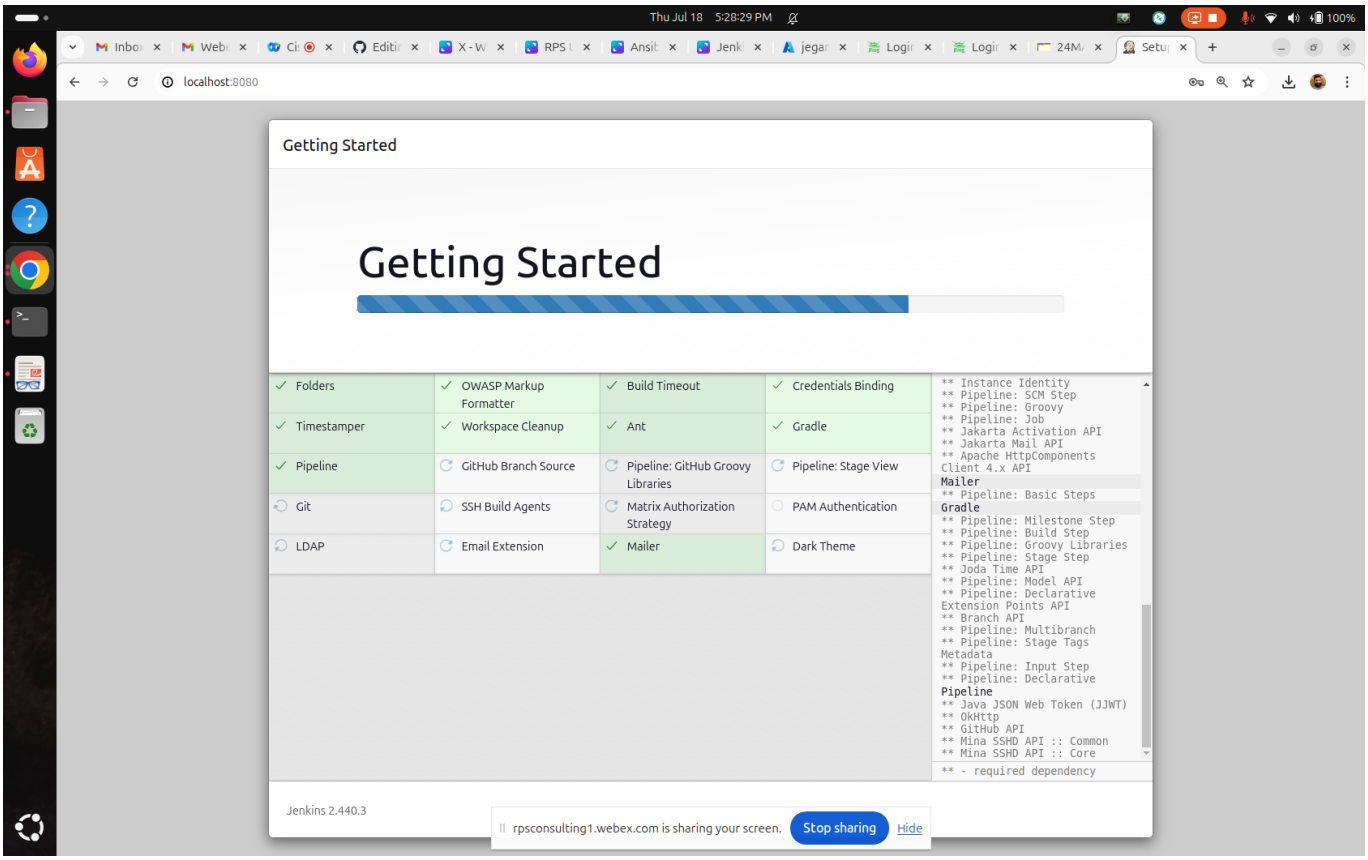

Thu Jul 18 5:25:13 PM
File /home/jegan/jenkins/secrets/initialAdminPassword
2192cb7f38594de28a31abbcfc294ed4

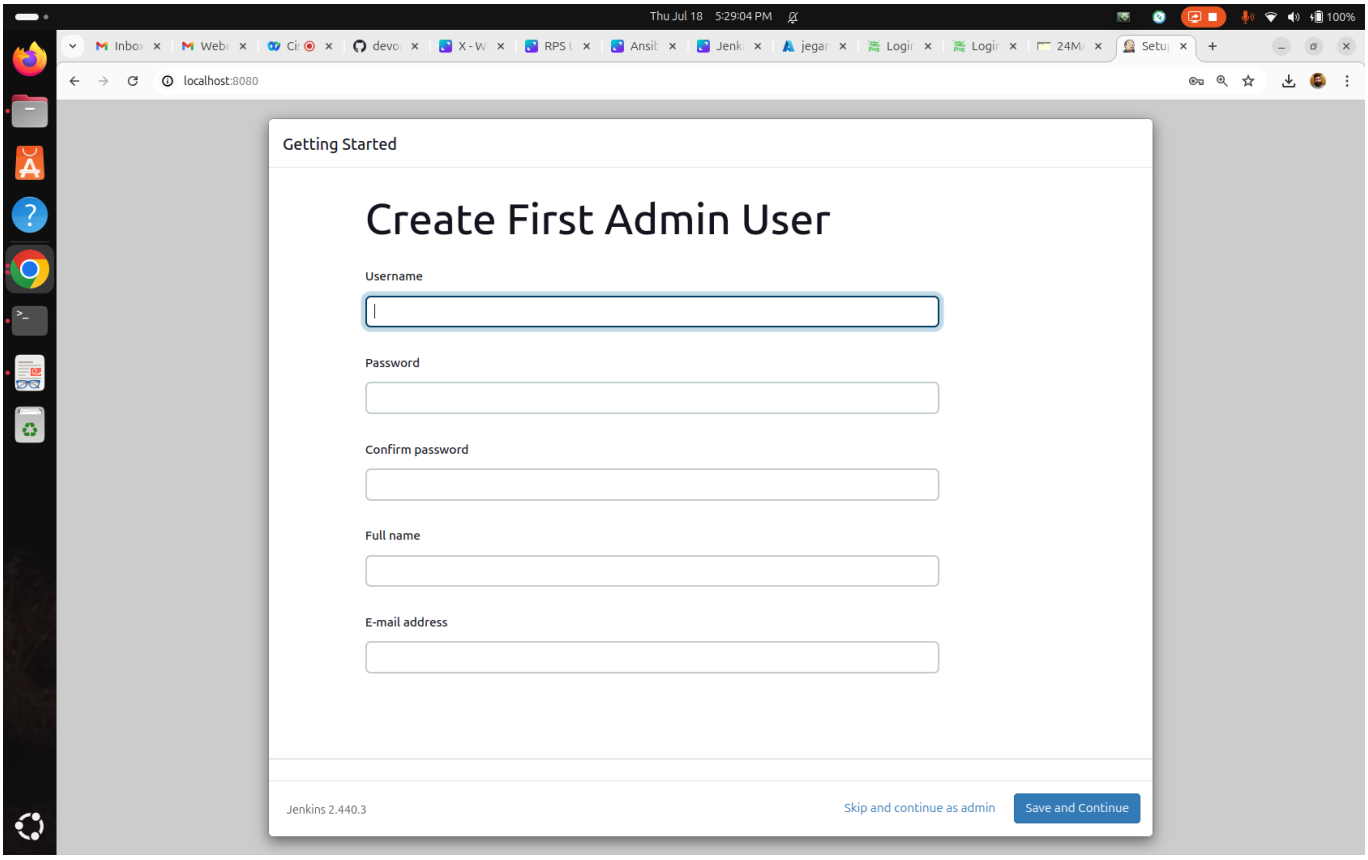

Thu Jul 18 5:25:50 PM
localhost:8080/login?from=%2F
Getting Started
Unlock Jenkins
To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:
/home/jegan/.jenkins/secrets/initialAdminPassword
Please copy the password from either location and paste it below.
Administrator password
Continue

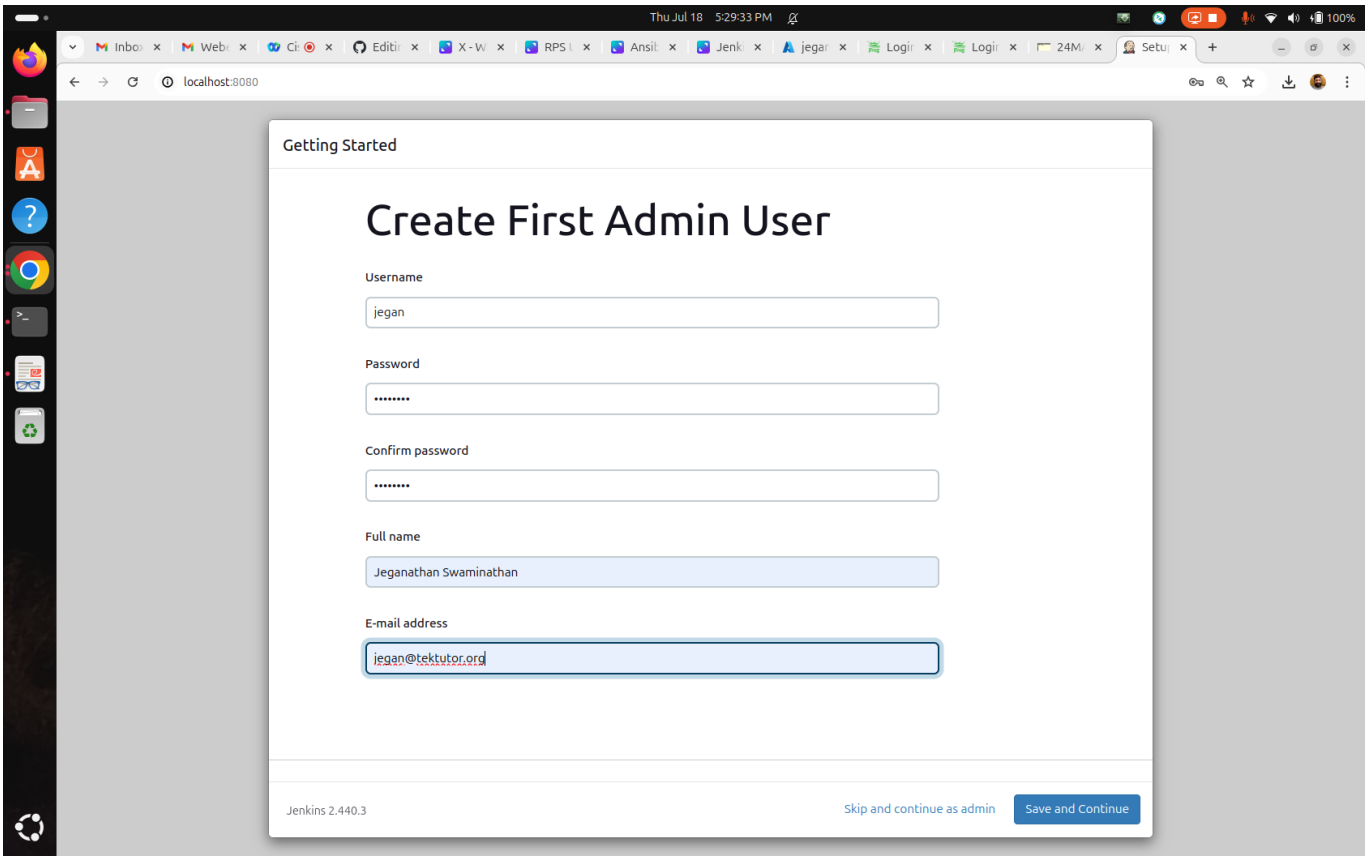


Select "Install Suggested Plugins"

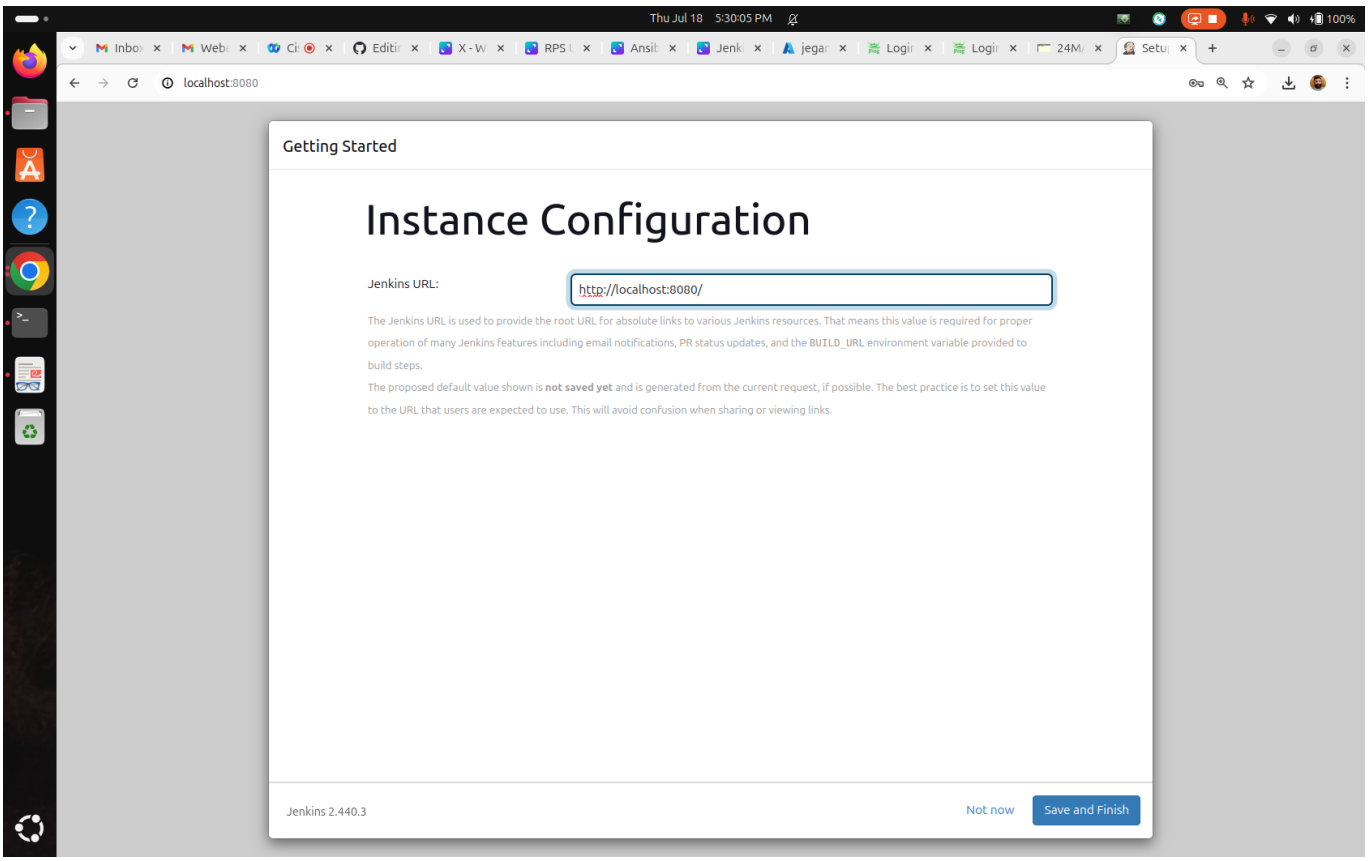



Thu Jul 18 5:28:29 PM
localhost:8080
Getting Started
Getting Started
Folders
OWASP Markup Formatter
Build Timeout
Credentials Binding
Timestamper
Workspace Cleanup
Ant
Gradle
Pipeline
GitHub Branch Source
Pipeline: GitHub Groovy Libraries
Pipeline: Stage View
Git
SSH Build Agents
Matrix Authorization Strategy
PAM Authentication
LDAP
Email Extension
Mailer
Dark Theme
** Instance Identity
** Pipeline: SCM Step
** Pipeline: Groovy
** Pipeline: Job
** Jakarta Activation API
** Jakarta Mail API
** Apache HttpComponents Client 4.x API
Mailer
** Pipeline: Basic Steps
Gradle
** Pipeline: Milestone Step
** Pipeline: Build Step
** Pipeline: Groovy Libraries
** Pipeline: Stage Step
** Joda Time API
** Pipeline: Model API
** Pipeline: Declarative Extension Points API
** Branch API
** Pipeline: Multibranch
** Pipeline: Stage Tags Metadata
** Pipeline: Input Step
** Pipeline: Declarative Pipeline
** Java JSON Web Token (JJWT)
** OkHttp
** GitHub API
** Mina SSHD API :: Common
** Mina SSHD API :: Core
** - required dependency
Jenkins 2.440.3
rpsconsulting1.webex.com is sharing your screen.
Stop sharing
Hide

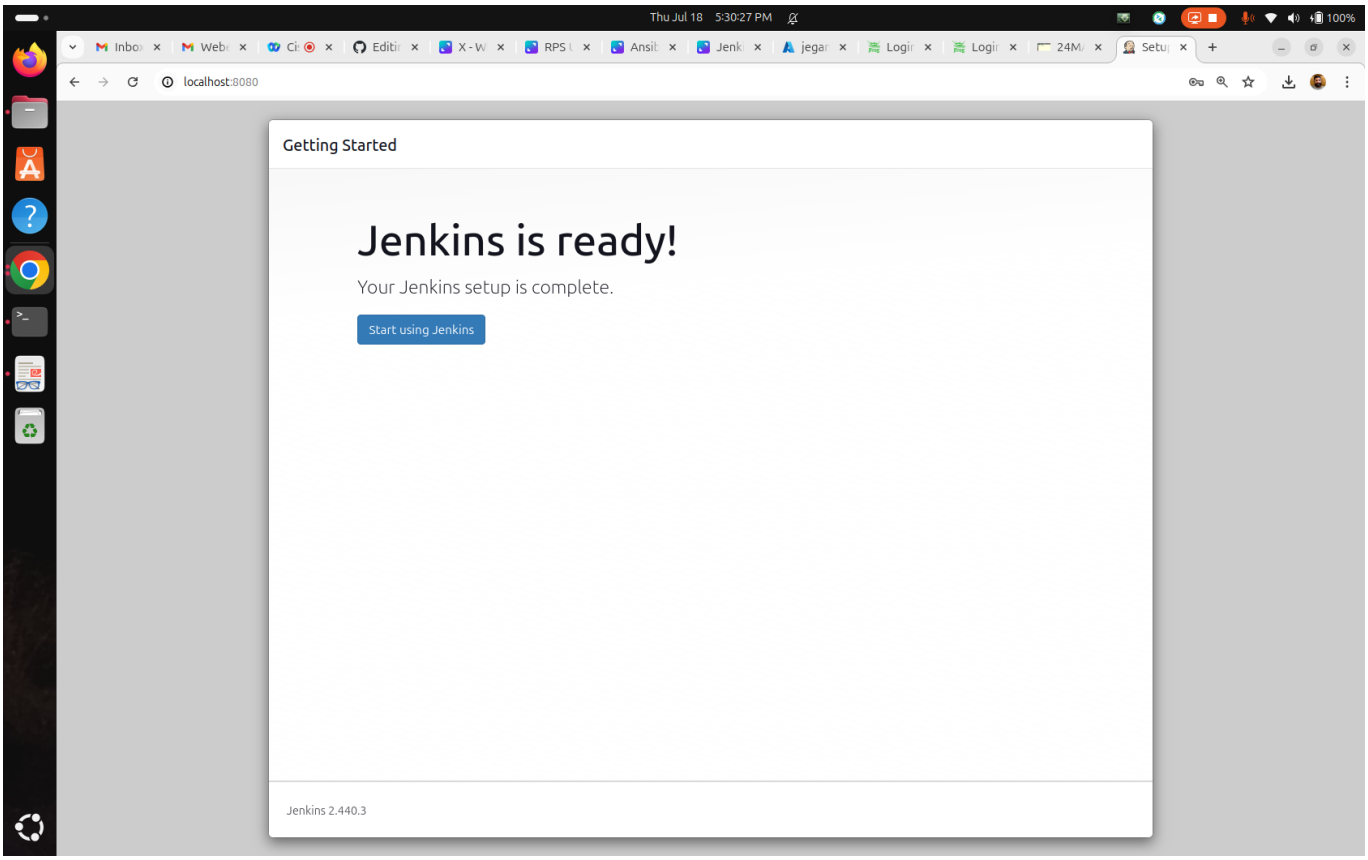

Thu Jul 18 5:29:04 PM
localhost:8080
Getting Started
Create First Admin User
Username
Password
Confirm password
Full name
E-mail address
Jenkins 2.440.3
Skip and continue as admin
Save and Continue



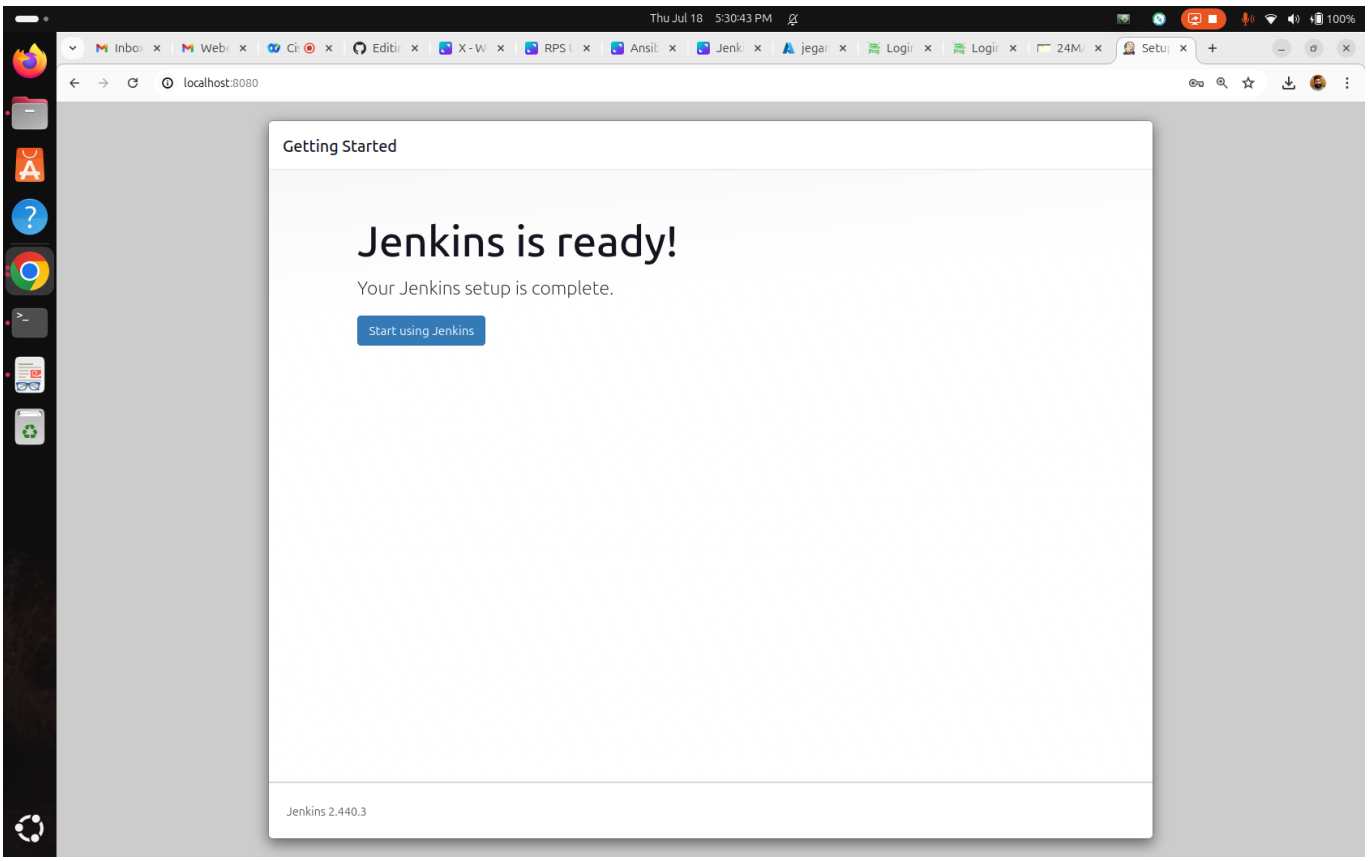
Click on "Save and Continue"

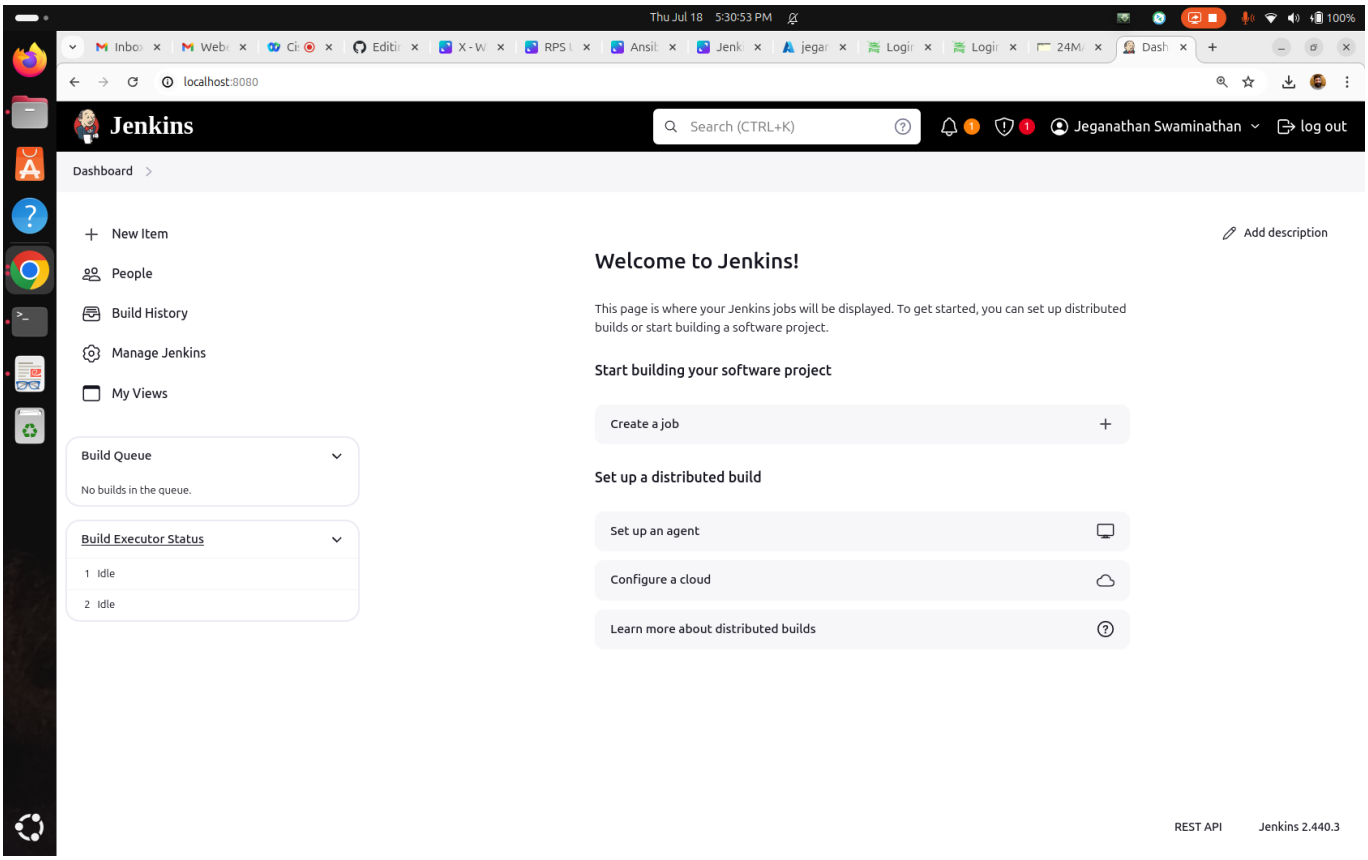


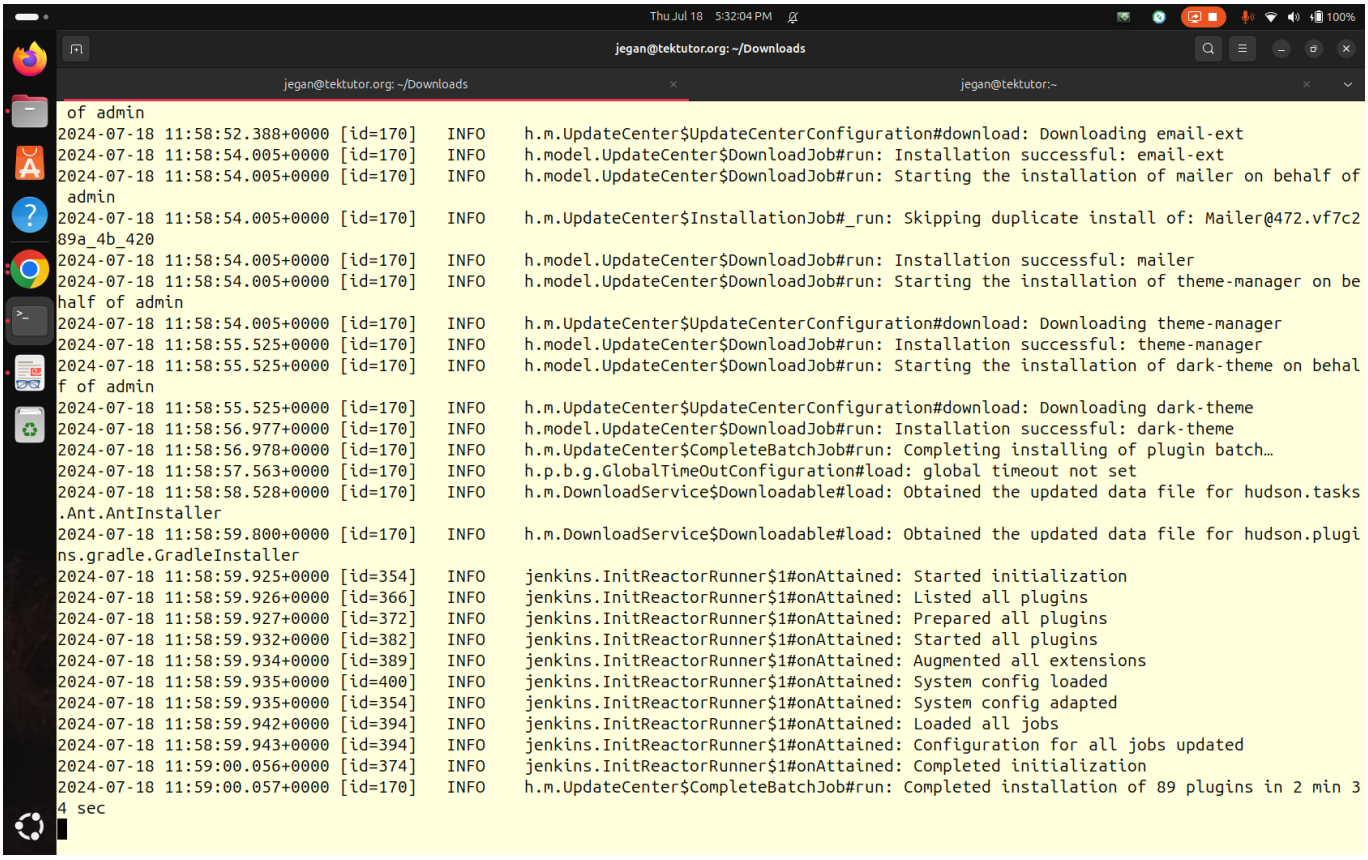
Click on "Save and Finish"



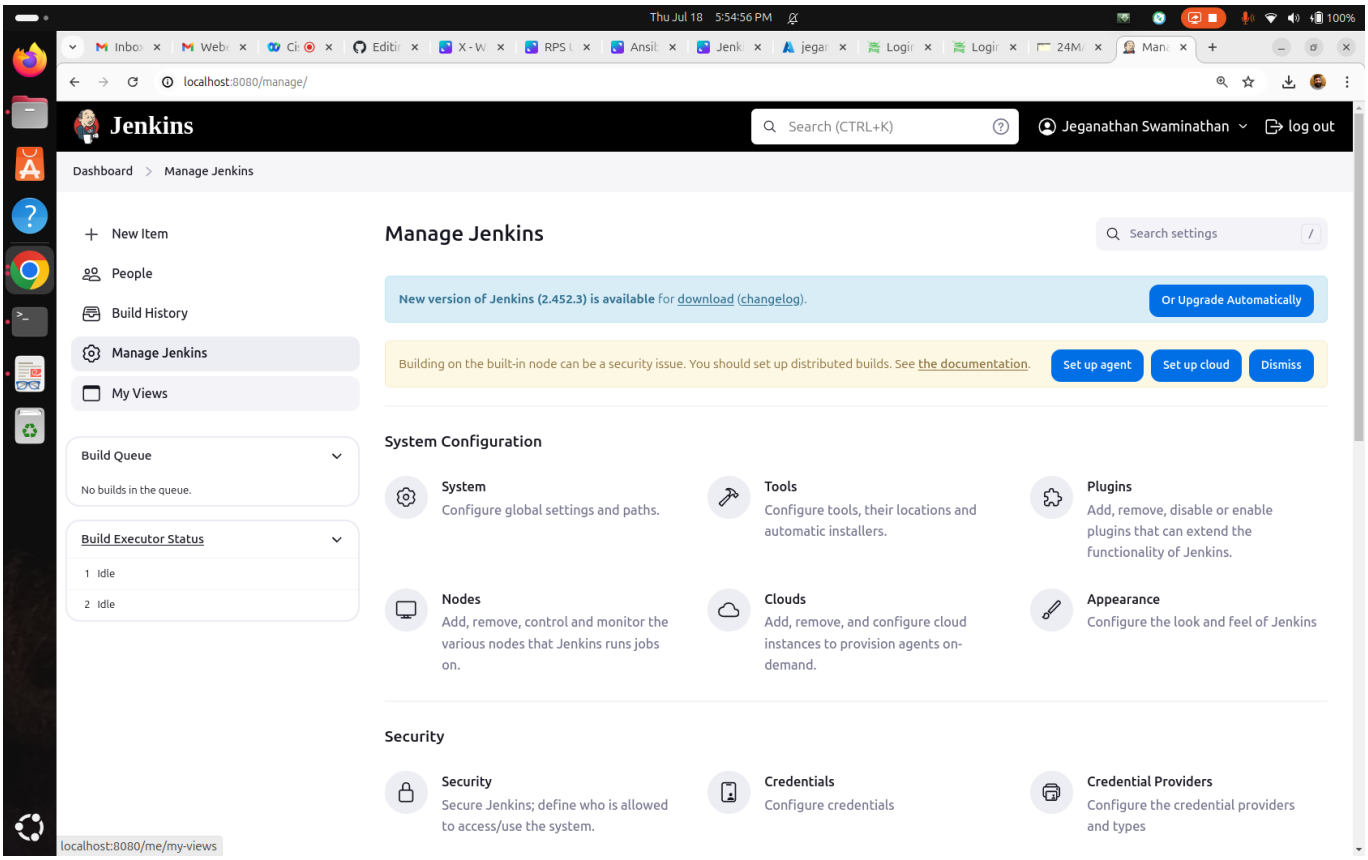
Click on "Start using Jenkins"



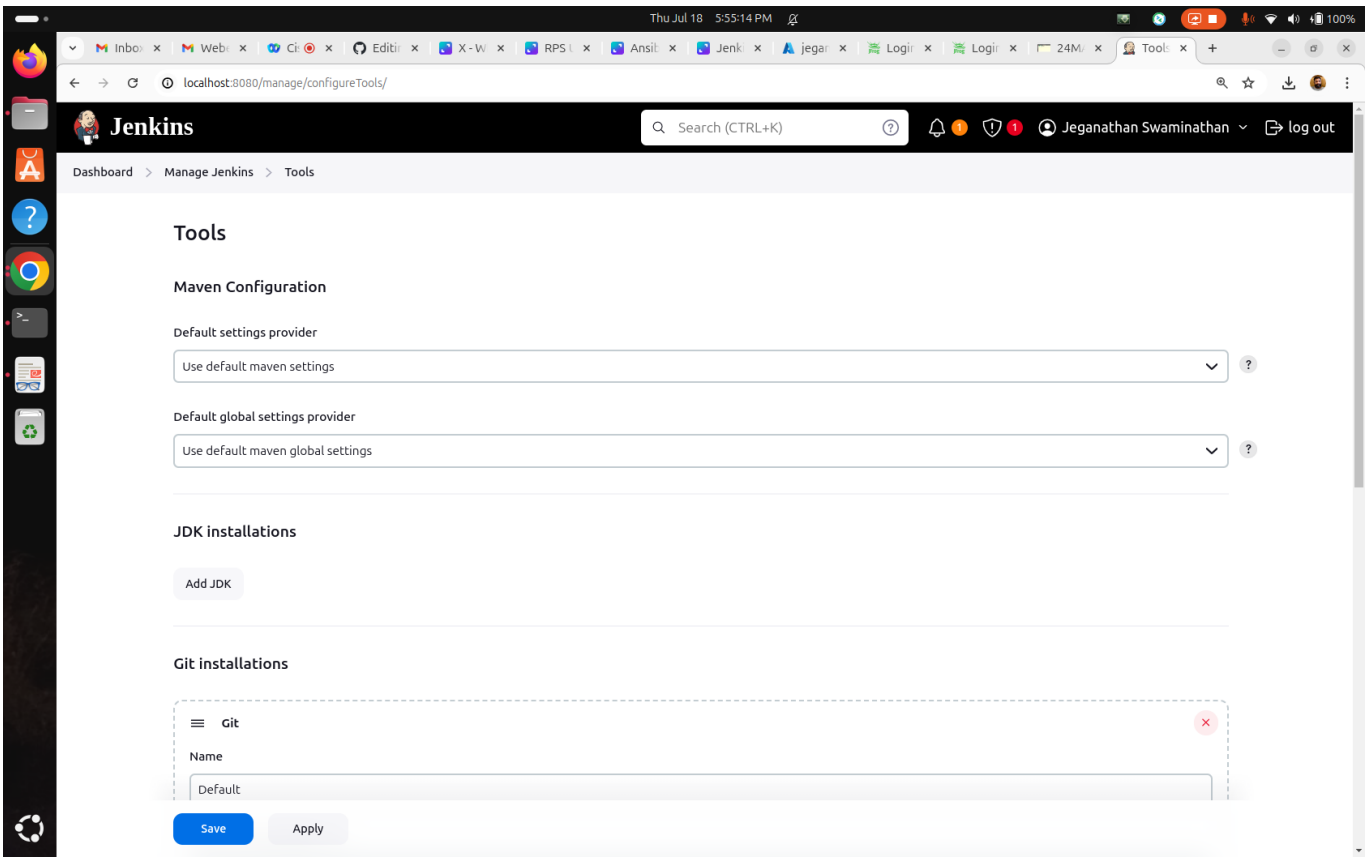




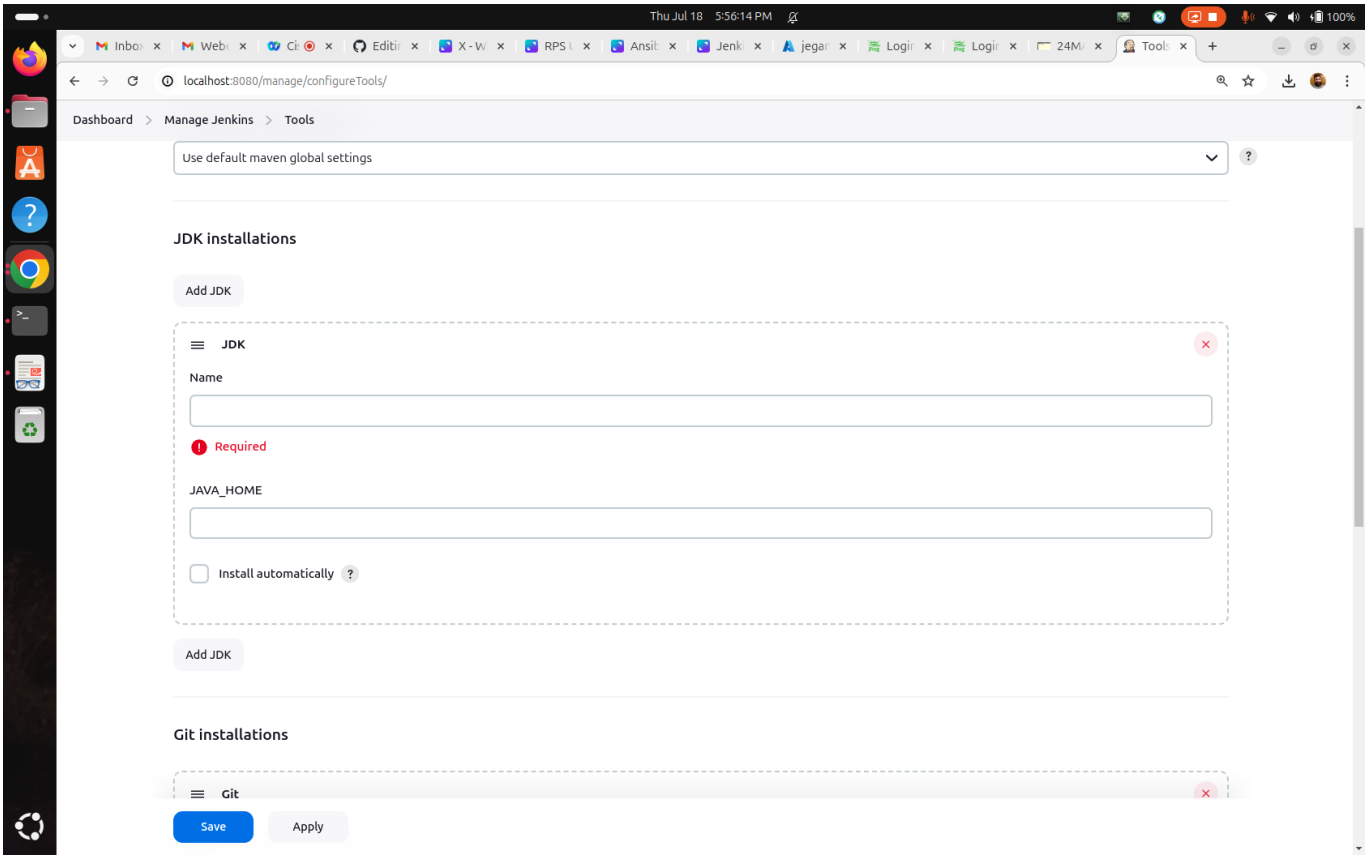
Click on "Manage Jenkins"



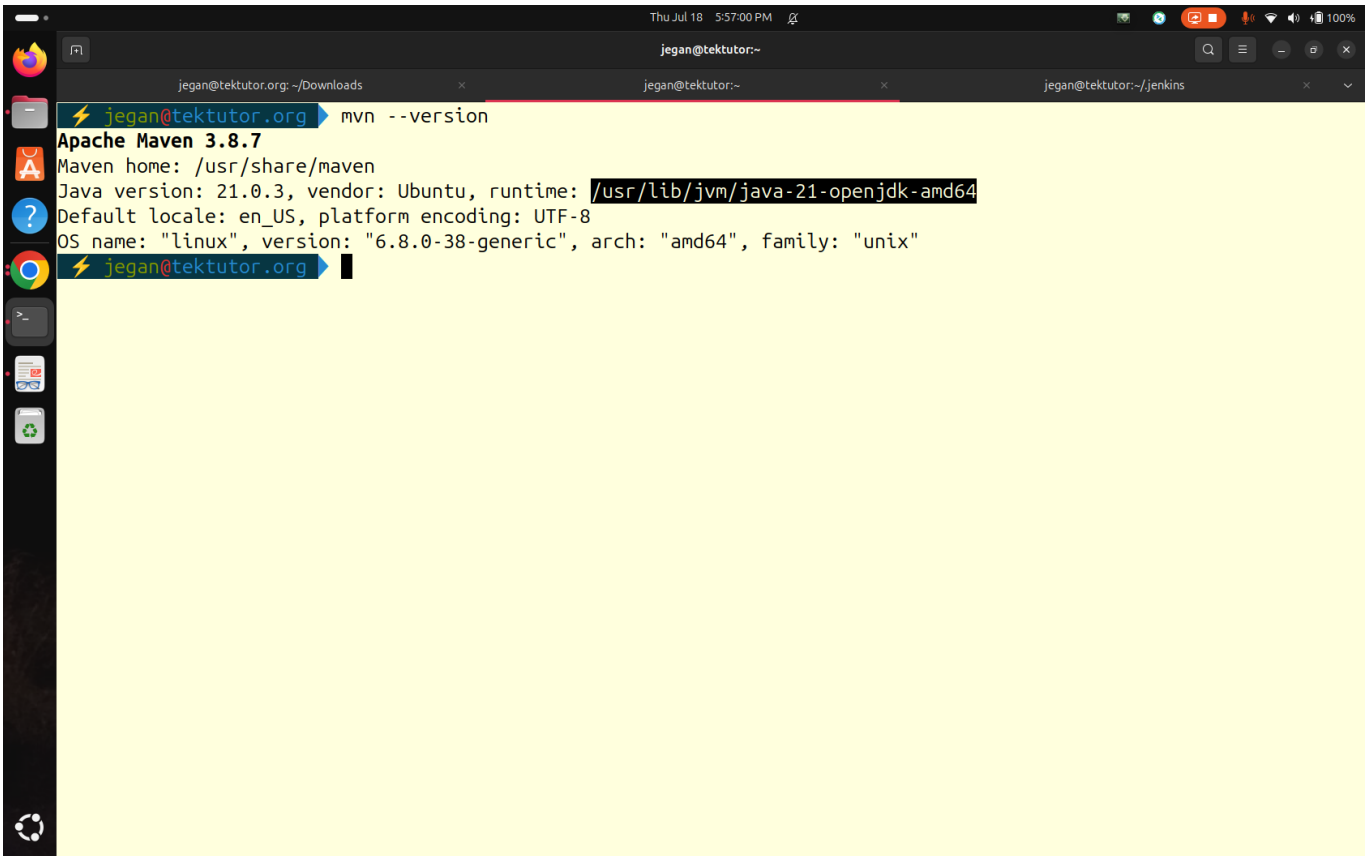
Click on "Tools"



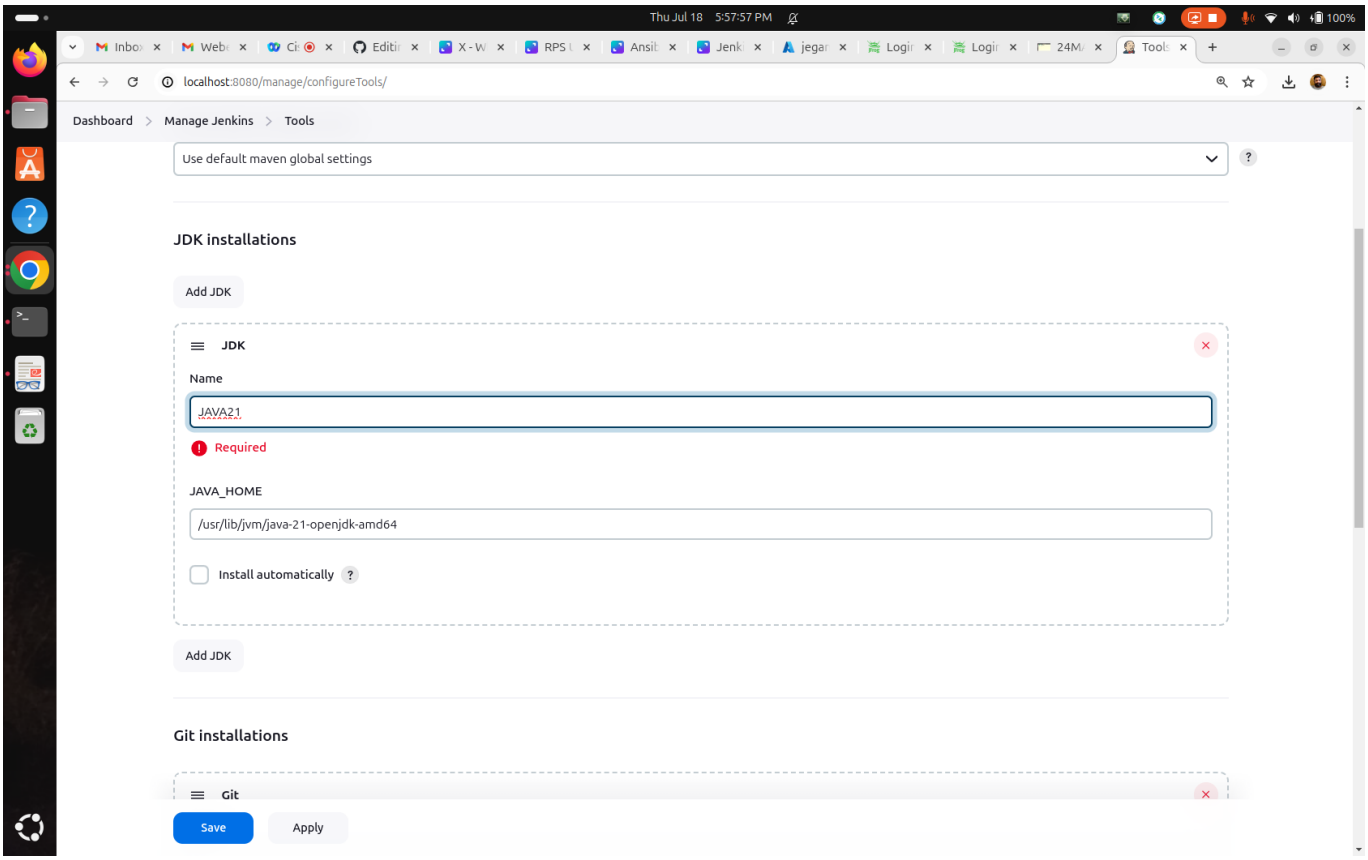
Under "JDK Installation", select "Add JDK"

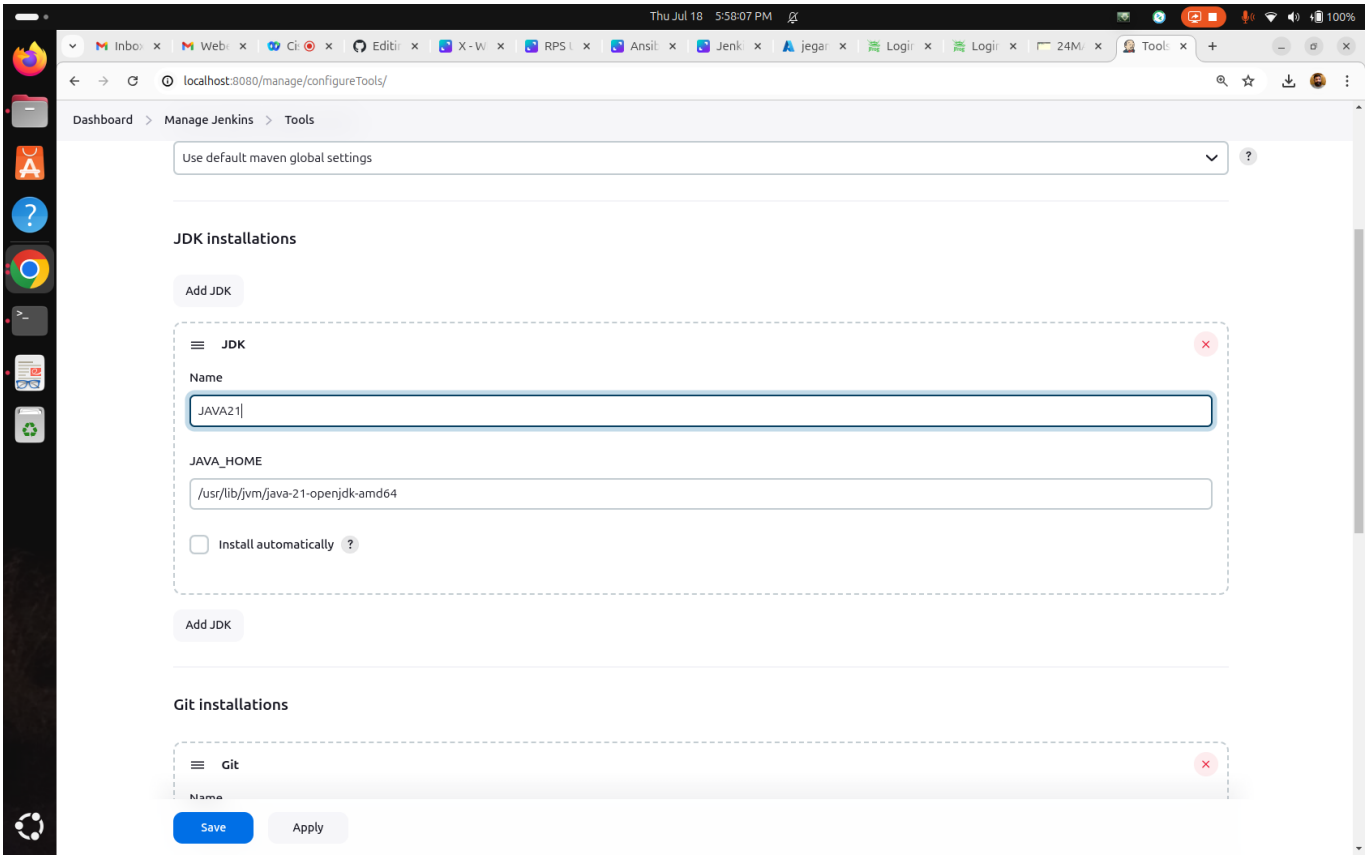


Go to your Linux Terminal and type "mvn --version", copy the Java home and return back to Jenkins browser tab

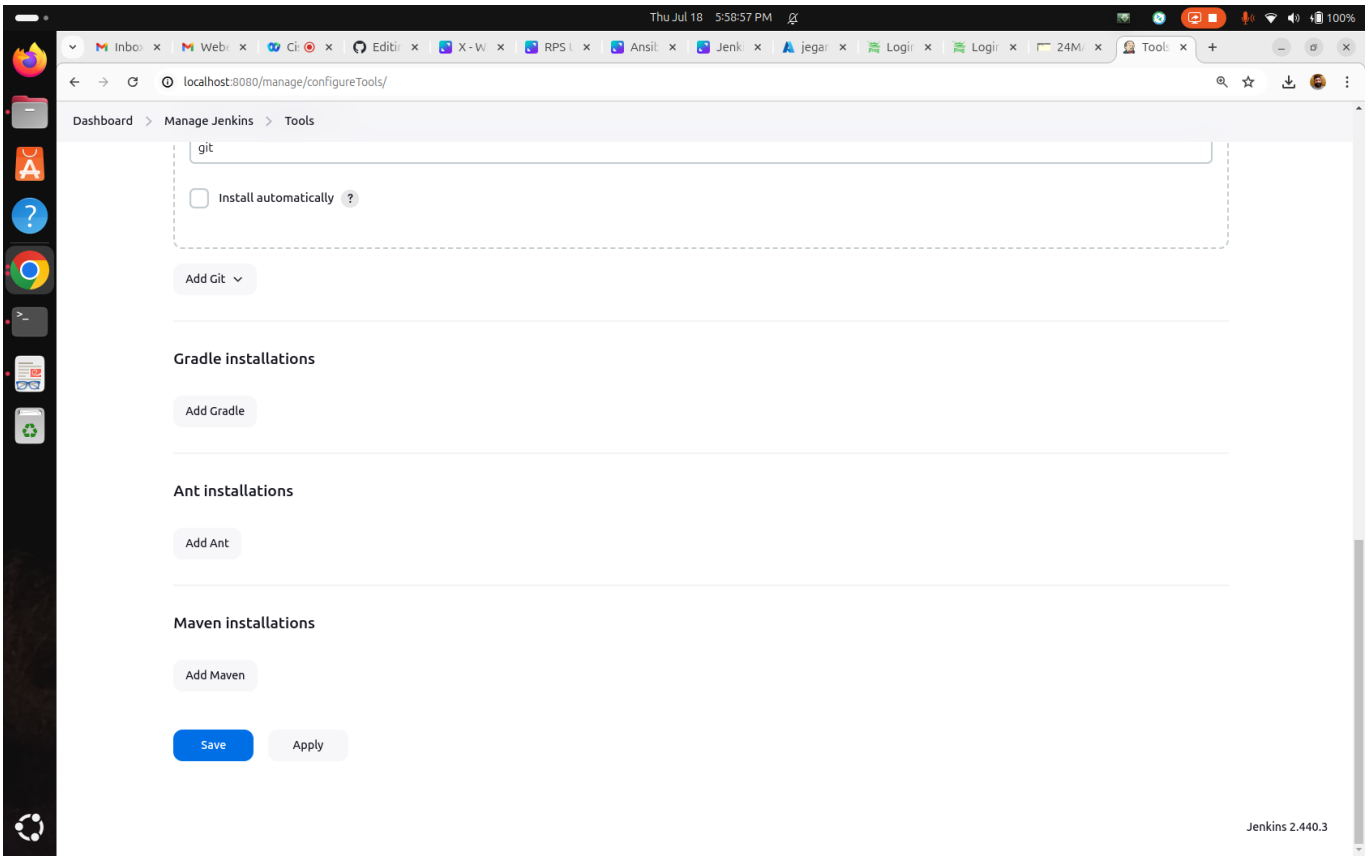


Paste the JDK path your copied from mvn terminal into the JDK_PATH in the Jenkins page

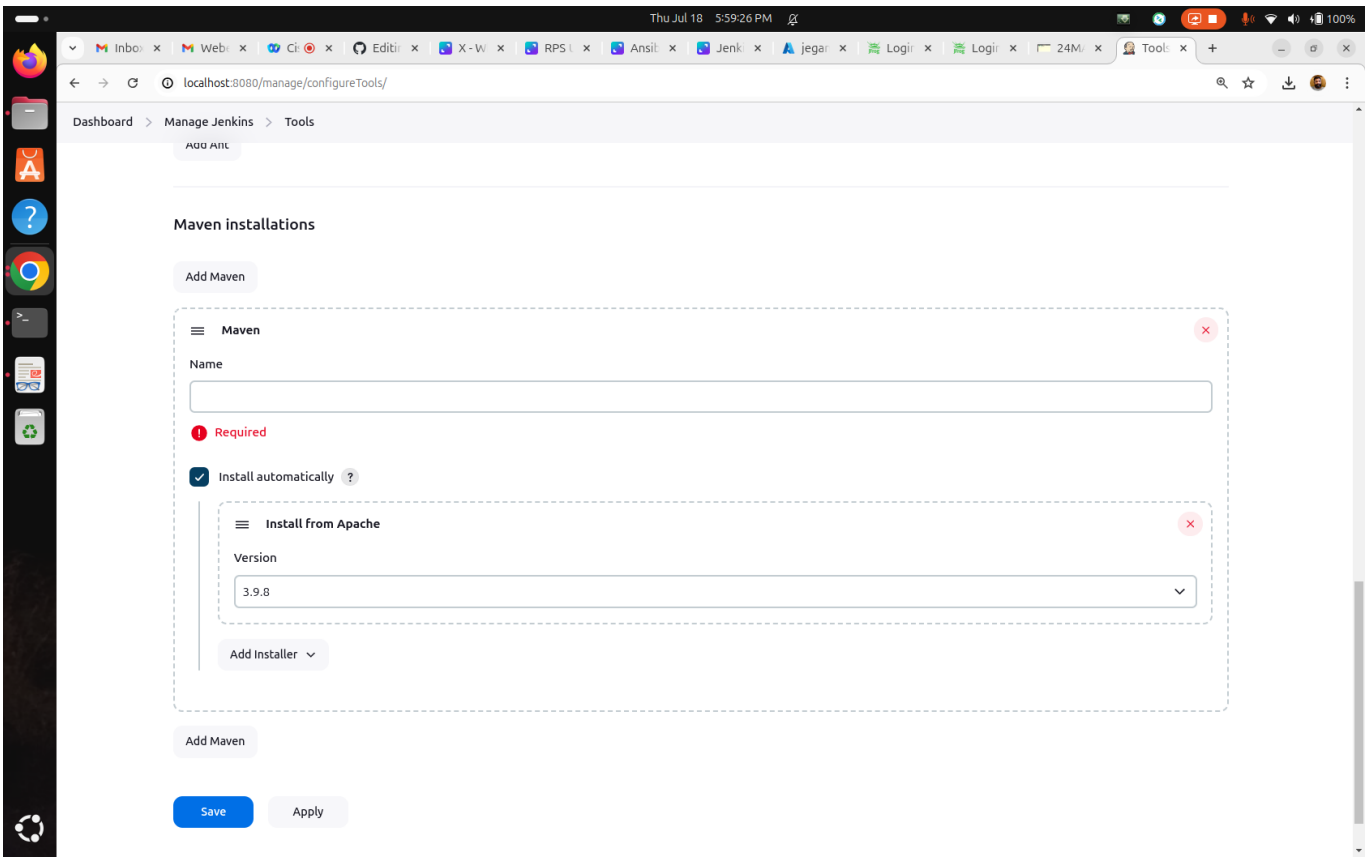




Scroll to the bottom and you will see "Maven Installations"

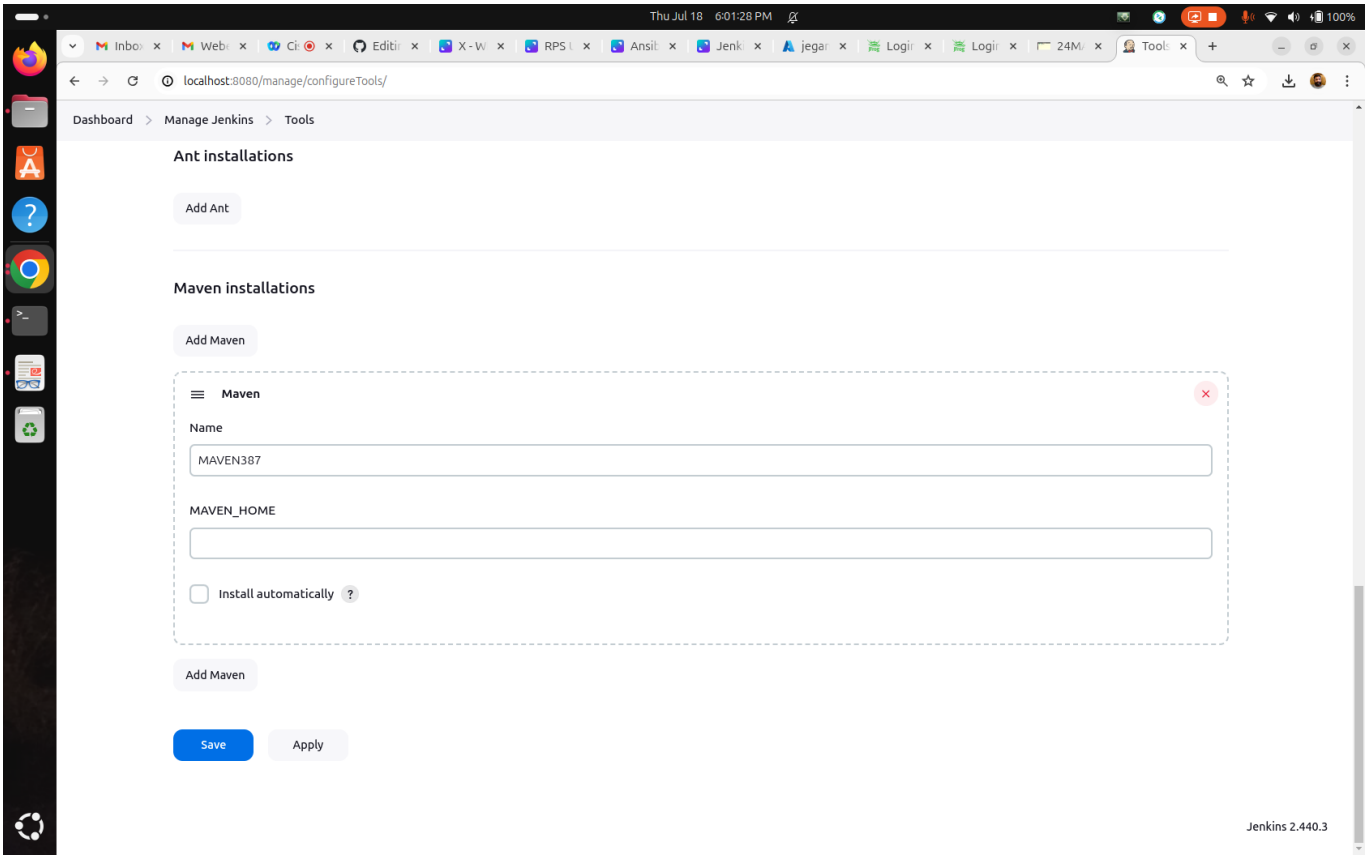


Click on "Add Maven"

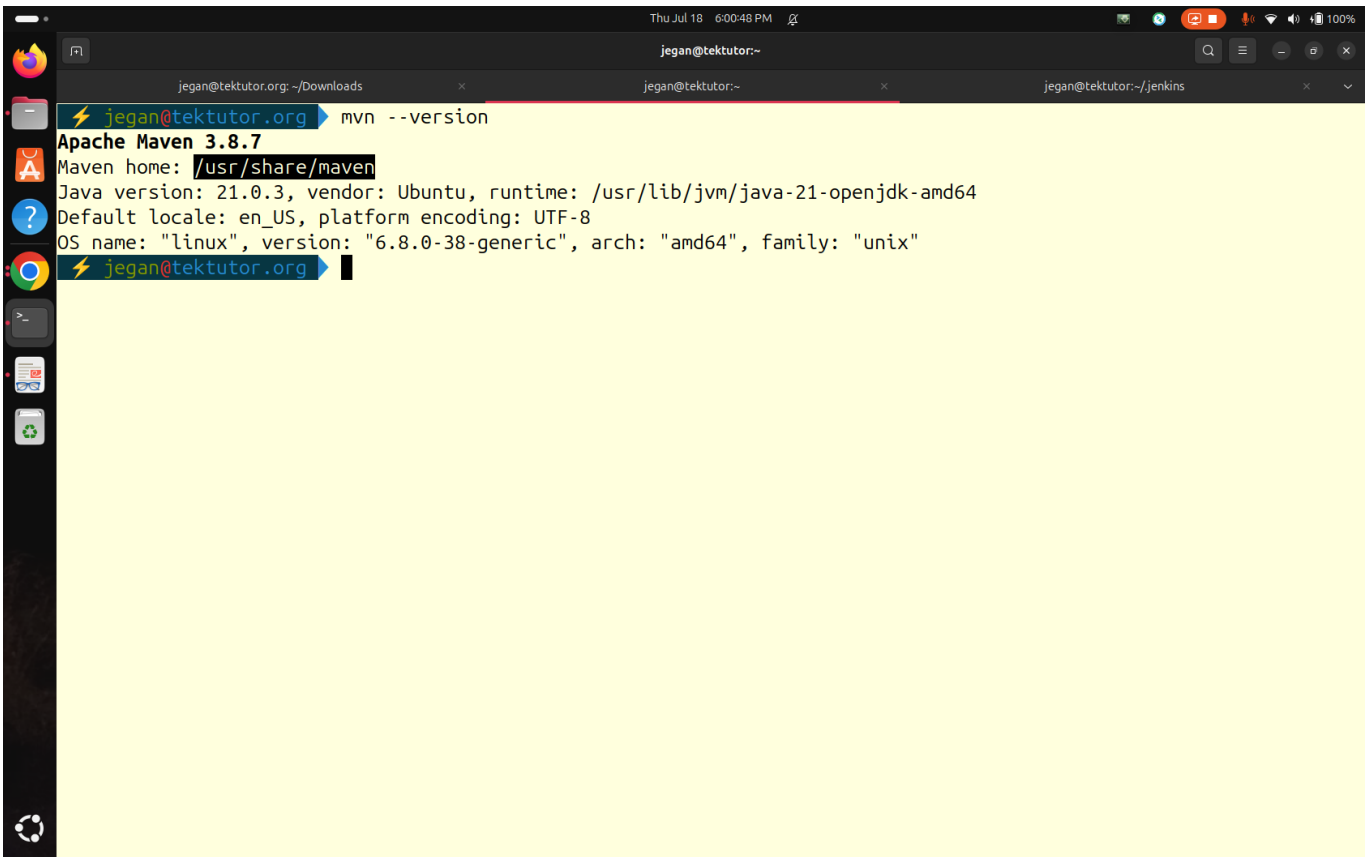


Under Name, you can type MAVEN387 to indicate maven 3.8.7 or given an appropriate string as per your

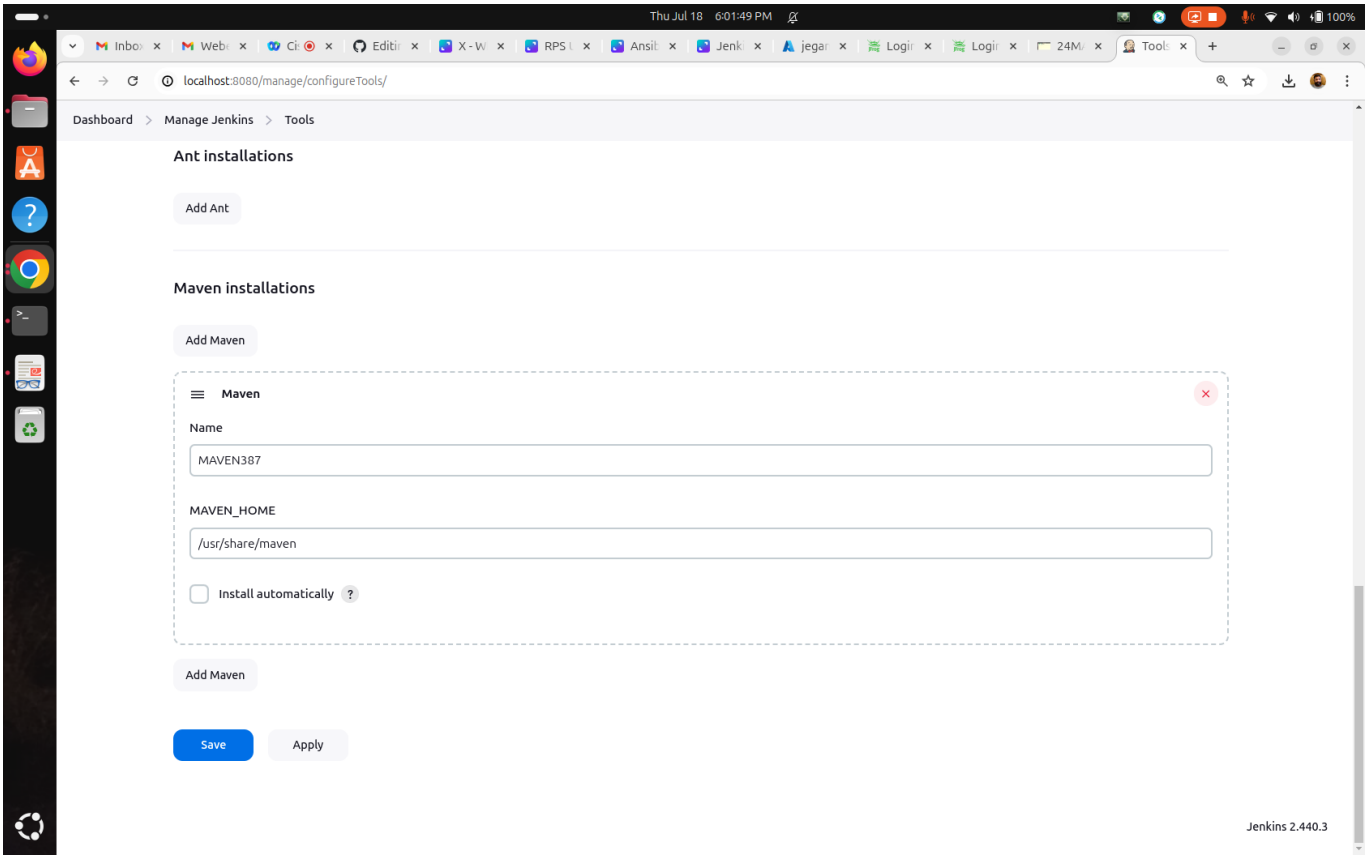
maven version. Uncheck, "Install automatically"



Go to terminal, copy the Maven home



Return back to your Jenkins configuration page and paste the Maven Path as shown below



Click "Save" button