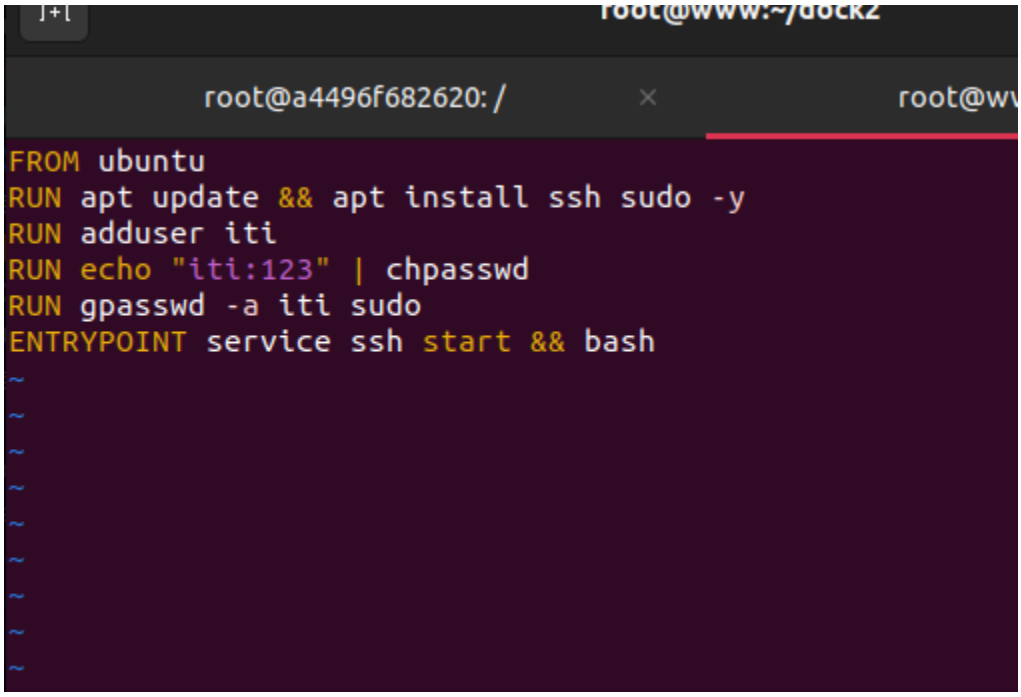
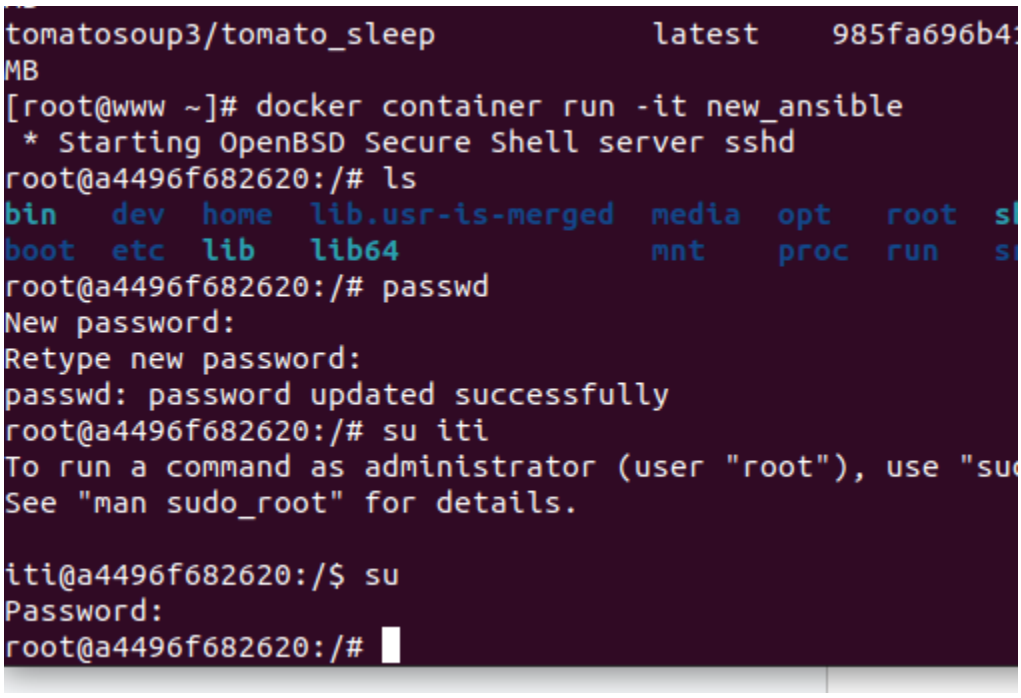
Q1:
Building a docker image from a docker file of ubuntu with the username iti and ocpying the public ssh key of the control node to the container using ssh-copy-id for it to be able to ssh remotely



Initiating the root password

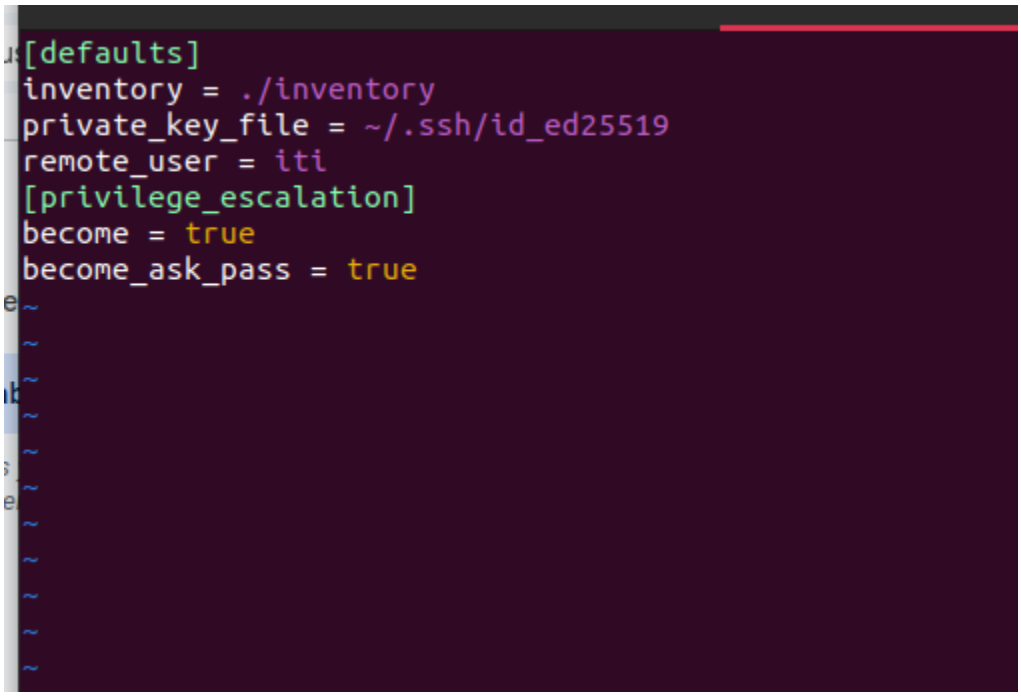


```
[root@www ~]# ssh-copy-id -i ~/.ssh/id_ed25519.pub iti@172.17.0.2
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_ed2
5519.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompt
ed now it is to install the new keys
iti@172.17.0.2's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'iti@172.17.0.2'"
```

Using the ansible.cfg file



Inventory file

```
[containers]
172.17.0.2
```

```yaml
- name: nginx_update
  hosts: containers
  tasks:
    - name: update_nginx
      tags: update
      apt:
        name: nginx
        only_upgrade: true

    - name: install nginx
      tags: install
      apt:
        name: nginx
        #latest upgrades the package such as the only_upg
        #but its used as the requirements of the lab i gu
        state: latest
    - name: ping_machine
      tags: always
      ping:
```

```
[root@www dock2]# ansible-playbook playbook.yml --tags update
BECOME password:

PLAY [nginx_update] ************************************************

TASK [Gathering Facts] *********************************************
ok: [172.17.0.2]

TASK [update_nginx] ************************************************
ok: [172.17.0.2]

TASK [ping_machine] ************************************************
ok: [172.17.0.2]

PLAY RECAP *********************************************************
172.17.0.2                 : ok=3    changed=0    unreachable=0    faile
kipped=0    rescued=0    ignored=0
```

Running with –tags update and the ping task runs anyways since it has the tag always

Q2:
First defining host_variables and group_variables:

```
[containers]
172.17.0.2 host_var=90

[containers:vars]
group_var=80
~
~
~
~
~
~
~
~
~
~
~
~
~
```

Second defining play specific variables:

```
- name: variables_play
  hosts: containers
  vars:
    package: vim
    version: latest
  tasks:
    - name: installing_package
      apt:
        name: "{{ package }}"
        state: "{{ version }}"
    - name: checking_variables
      debug:
        msg: "{{ host_var }} {{ group_var }} {{ runtime_var }} {{ packag
version }}"
~
~
~
~
```

And adding a debug message containing all variables even runtime variables which are defined by –extra_vars during the command and we can see all are present

```
[root@www dock2]# vim playbook2.yml
[root@www dock2]# ansible-playbook playbook2.yml --extra-var "runtime_var=300"
BECOME password:

PLAY [variables_play] ***************************************************

TASK [Gathering Facts] ***************************************************
ok: [172.17.0.2]

TASK [installing_package] ************************************************
ok: [172.17.0.2]

TASK [checking_variables] ************************************************
ok: [172.17.0.2] => {
    "msg": "90 80 300 vim latest"
}

PLAY RECAP ***************************************************************
172.17.0.2                 : ok=3    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0

[root@www dock2]#
```

Q3:

Creating a playbook file containing loops and loops with different states and packages

```
- name: looping_play
  hosts: containers
  tasks:
    - name: install packages
      apt:
        name: "{{ item }}"
        state: latest
      loop:
        - nginx
        - vim
        - apache2

    - name: install dict
      apt:
        name: "{{ item.package_name }}"
        state: "{{ item.state }}"
      loop:
        - {package_name: 'nginx',state: 'latest'}
        - {package_name: 'vim',state: 'present'}
        - {package_name: 'apache2',state: 'absent'}
~
```

```
[root@www dock2]# vim playbook3.yml
[root@www dock2]# ansible-playbook playbook3.yml
BECOME password:

PLAY [looping_play] ************************************************************************

TASK [Gathering Facts] *********************************************************************
ok: [172.17.0.2]

TASK [install packages] ********************************************************************
ok: [172.17.0.2] => (item=nginx)
^[[A^[[Aok: [172.17.0.2] => (item=vim)
ok: [172.17.0.2] => (item=apache2)

TASK [install dict] ************************************************************************
ok: [172.17.0.2] => (item={'package_name': 'nginx', 'state': 'latest'})
ok: [172.17.0.2] => (item={'package_name': 'vim', 'state': 'present'})
changed: [172.17.0.2] => (item={'package_name': 'apache2', 'state': 'absent'})

PLAY RECAP *********************************************************************************
172.17.0.2                 : ok=3    changed=1    unreachable=0    failed=0    skipped=0    res

[root@www dock2]# vim playbook3.yml
[root@www dock2]#
```

Q4:

Creating an identical dockerfile but including centos and not ubuntu

```
FROM centos
RUN sed -E -i 's/http:\/\/mirror(list)?/http:\/\/vault/g' /etc/yum.repos.d/*
RUN sed -E -i 's/^#baseurl=/baseurl=/g' /etc/yum.repos.d/*
RUN yum -y update && yum clean all
RUN yum install openssh-server openssh-clients -y
RUN /usr/bin/ssh-keygen -A
# dont know the consquences of such action
RUN rm /run/nologin
RUN adduser iti
RUN echo "iti:123" | chpasswd
RUN gpasswd -a iti wheel
ENTRYPOINT  /usr/sbin/sshd -DE /tmp/sshd.log
~
```

Needing to change several configurations and editing the mirror links yum is installed from to be able to install using yum and installing ssh and removing several files because of the absence of systemd

Setting up both ubuntu and centos containers with my ssh public key

```
[root@www .ssh]# ssh-copy-id -i ~/.ssh/id_rsa.pub iti@172.17.0.2
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_rsa.pub"
The authenticity of host '172.17.0.2 (172.17.0.2)' can't be established.
ED25519 key fingerprint is SHA256:jTpfZKlza2BlgevpcbPyfQqA26v4TYG0km5itmf3QbE.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
iti@172.17.0.2's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'iti@172.17.0.2'"
and check to make sure that only the key(s) you wanted were added.

[root@www .ssh]# ls
id_rsa  id_rsa.pub  known_hosts  known_hosts.old
[root@www .ssh]# ssh iti@172.17.0.2
[iti@6f05c5706842 ~]$ ls
[iti@6f05c5706842 ~]$ ls /
bin  boot  dev  etc  home  lib  lib64  lost+found  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
[iti@6f05c5706842 ~]$ exit
logout
Connection to 172.17.0.2 closed.
[root@www .ssh]# ls
id_rsa  id_rsa.pub  known_hosts  known_hosts.old
[root@www .ssh]# cd ..
[root@www ~]# ls
anaconda-ks.cfg  dock  dock2  web
[root@www ~]# ssh-copy-id -i ~/.ssh/id_rsa.pub iti@172.17.0.3
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_rsa.pub"
The authenticity of host '172.17.0.3 (172.17.0.3)' can't be established.
ED25519 key fingerprint is SHA256:c35iYODR8Y2DTamSoyu1fiLX6/ZzIbbB3XQlH99oSxg.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
iti@172.17.0.3's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'iti@172.17.0.3'"
and check to make sure that only the key(s) you wanted were added.
```

Writing the playbook with when skipping and installing httpd for centos and nginx for ubuntu

```yaml
- name: several distributions when playbook
  hosts: containers
  tasks:
    - name: download nginx on ubuntu
      apt:
        name: nginx
        state: present
        update_cache: yes
      when: ansible_facts['distribution'] == "Ubuntu"
    - name: download httpd on centos
      yum:
        name: httpd
        state: latest
        update_cache: yes
      when: ansible_facts['distribution'] == "CentOS"
    - name: restart nginx
      service:
        name: nginx
        state: restarted
        use: service
~
~
```

Q5:

```yaml
- name: practicing resgister
  hosts: 172.17.0.3
  tasks:
    - name: restart systemd service if it exists
      service:
        state: restarted
        name: sshd
        use: service
      when: false
      register: bla
    - name: install a service
      apt:
        name: nginx
        force: true
      register: output
    - name: printing output
      debug:
        var: output
    - name: restart if installed
      service:
        name: nginx
        state: restarted
        use: service
      when: not output.failed|bool or output.changed|bool
~
```

Writing a playbook for installing a service and restarting once installed