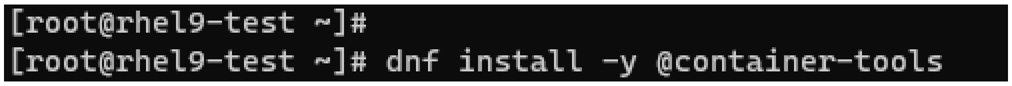
Container: Lecture 6

Persistent Container Lab (Even After Server Reboot):

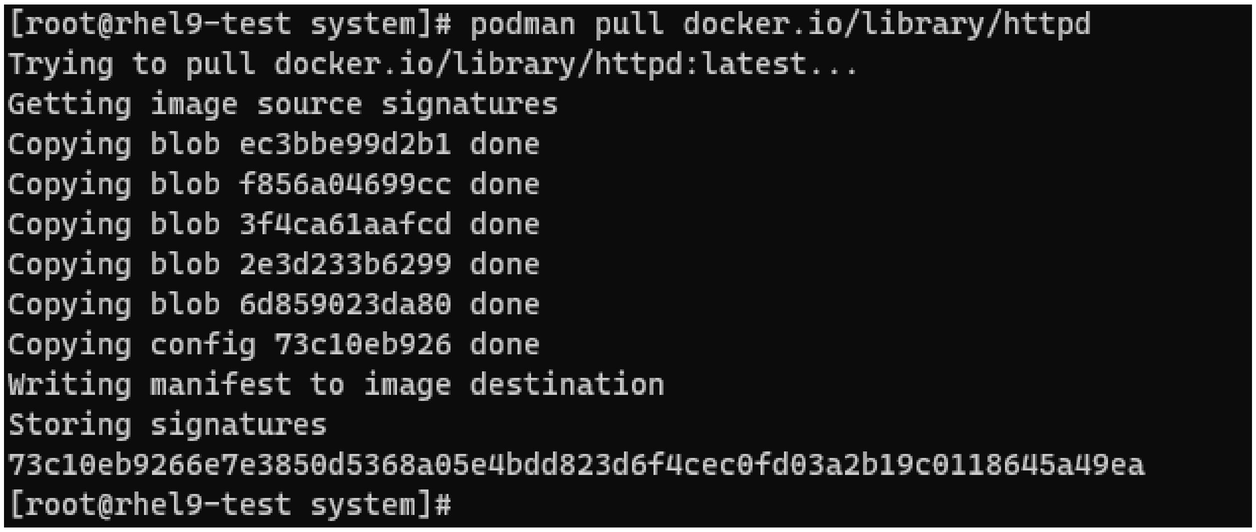
Root Full Mode:

1. Install Container packages-



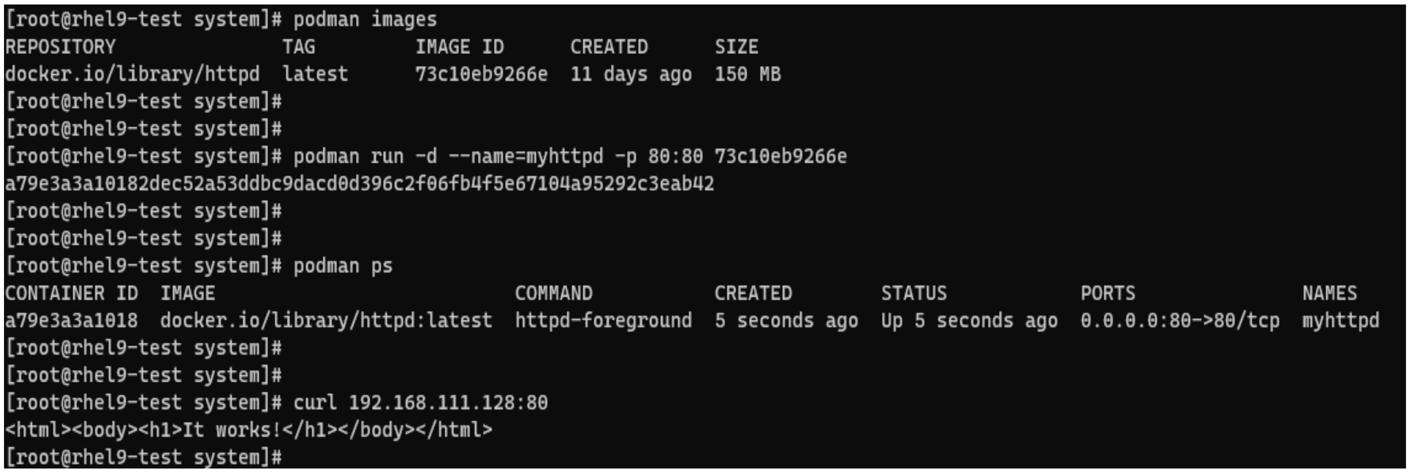
```
[root@rhel9-test ~]#
[root@rhel9-test ~]# dnf install -y @container-tools
```

2. Pull http container image-



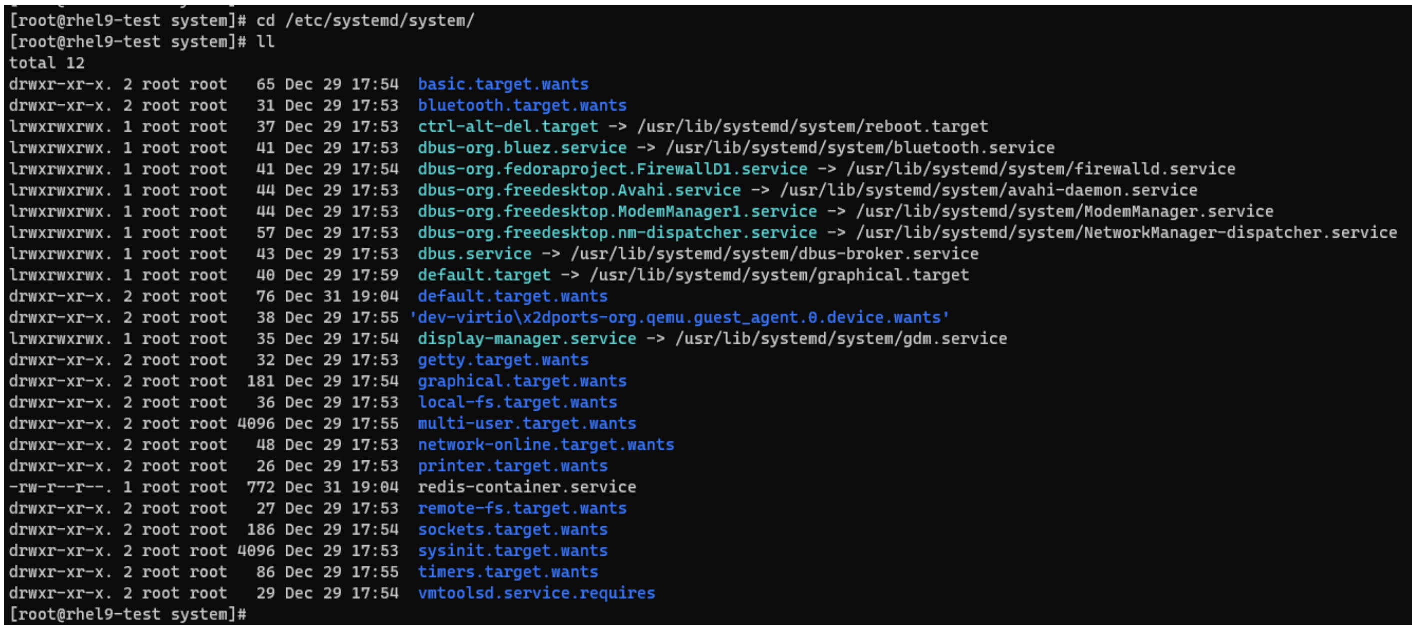
```
[root@rhel9-test system]# podman pull docker.io/library/httpd
Trying to pull docker.io/library/httpd:latest...
Getting image source signatures
Copying blob ec3bbe99d2b1 done
Copying blob f856a04699cc done
Copying blob 3f4ca61aafcd done
Copying blob 2e3d233b6299 done
Copying blob 6d859023da80 done
Copying config 73c10eb926 done
Writing manifest to image destination
Storing signatures
73c10eb9266e7e3850d5368a05e4bdd823d6f4cec0fd03a2b19c0118645a49ea
[root@rhel9-test system]#
```

3. Verify image. Run it in background & check web URL-



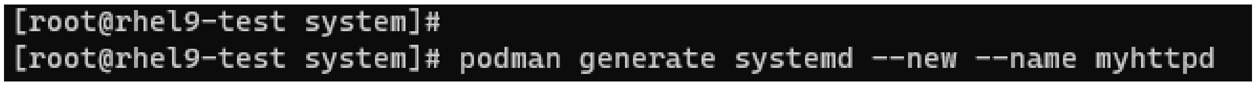
```
[root@rhel9-test system]# podman images
REPOSITORY               TAG         IMAGE ID      CREATED      SIZE
docker.io/library/httpd  latest      73c10eb9266e  11 days ago  150 MB
[root@rhel9-test system]#
[root@rhel9-test system]#
[root@rhel9-test system]# podman run -d --name=myhttpd -p 80:80 73c10eb9266e
a79e3a3a10182dec52a53ddbc9dacd0d396c2f06fb4f5e67104a95292c3eab42
[root@rhel9-test system]#
[root@rhel9-test system]#
[root@rhel9-test system]# podman ps
CONTAINER ID  IMAGE                           COMMAND           CREATED        STATUS            PORTS               NAMES
a79e3a3a1018  docker.io/library/httpd:latest  httpd-foreground  5 seconds ago  Up 5 seconds ago  0.0.0.0:80->80/tcp  myhttpd
[root@rhel9-test system]#
[root@rhel9-test system]#
[root@rhel9-test system]# curl 192.168.111.128:80
<html><body><h1>It works!</h1></body></html>
[root@rhel9-test system]#
```

4. Now, we will define this container as a service under systemd-



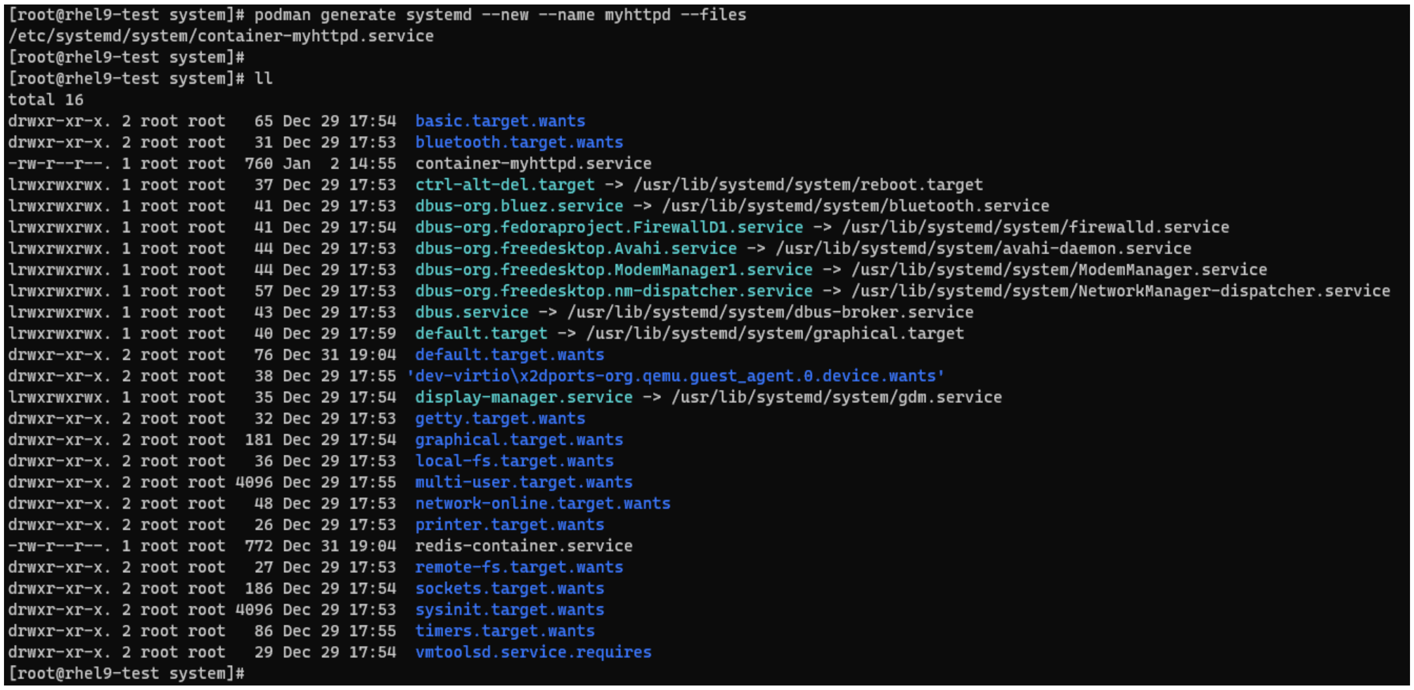
```
[root@rhel9-test system]# cd /etc/systemd/system/
[root@rhel9-test system]# ll
total 12
drwxr-xr-x. 2 root root   65 Dec 29 17:54  basic.target.wants
drwxr-xr-x. 2 root root   31 Dec 29 17:53  bluetooth.target.wants
lrwxrwxrwx. 1 root root   37 Dec 29 17:53  ctrl-alt-del.target -> /usr/lib/systemd/system/reboot.target
lrwxrwxrwx. 1 root root   41 Dec 29 17:53  dbus-org.bluez.service -> /usr/lib/systemd/system/bluetooth.service
lrwxrwxrwx. 1 root root   41 Dec 29 17:54  dbus-org.fedoraproject.FirewallD1.service -> /usr/lib/systemd/system/firewalld.service
lrwxrwxrwx. 1 root root   44 Dec 29 17:53  dbus-org.freedesktop.Avahi.service -> /usr/lib/systemd/system/avahi-daemon.service
lrwxrwxrwx. 1 root root   44 Dec 29 17:53  dbus-org.freedesktop.ModemManager1.service -> /usr/lib/systemd/system/ModemManager.service
lrwxrwxrwx. 1 root root   57 Dec 29 17:53  dbus-org.freedesktop.nm-dispatcher.service -> /usr/lib/systemd/system/NetworkManager-dispatcher.service
lrwxrwxrwx. 1 root root   43 Dec 29 17:53  dbus.service -> /usr/lib/systemd/system/dbus-broker.service
lrwxrwxrwx. 1 root root   40 Dec 29 17:59  default.target -> /usr/lib/systemd/system/graphical.target
drwxr-xr-x. 2 root root   76 Dec 31 19:04  default.target.wants
drwxr-xr-x. 2 root root   38 Dec 29 17:55 'dev-virtio\x2dports-org.qemu.guest_agent.0.device.wants'
lrwxrwxrwx. 1 root root   35 Dec 29 17:54  display-manager.service -> /usr/lib/systemd/system/gdm.service
drwxr-xr-x. 2 root root   32 Dec 29 17:53  getty.target.wants
drwxr-xr-x. 2 root root  181 Dec 29 17:54  graphical.target.wants
drwxr-xr-x. 2 root root   36 Dec 29 17:53  local-fs.target.wants
drwxr-xr-x. 2 root root 4096 Dec 29 17:55  multi-user.target.wants
drwxr-xr-x. 2 root root   48 Dec 29 17:53  network-online.target.wants
drwxr-xr-x. 2 root root   26 Dec 29 17:53  printer.target.wants
-rw-r--r--. 1 root root  772 Dec 31 19:04  redis-container.service
drwxr-xr-x. 2 root root   27 Dec 29 17:53  remote-fs.target.wants
drwxr-xr-x. 2 root root  186 Dec 29 17:54  sockets.target.wants
drwxr-xr-x. 2 root root 4096 Dec 29 17:53  sysinit.target.wants
drwxr-xr-x. 2 root root   86 Dec 29 17:55  timers.target.wants
drwxr-xr-x. 2 root root   29 Dec 29 17:54  vmtoolsd.service.requires
[root@rhel9-test system]#
```

5. If we run below command, it will show unit content, but won't create file.



```
[root@rhel9-test system]#
[root@rhel9-test system]# podman generate systemd --new --name myhttpd
```

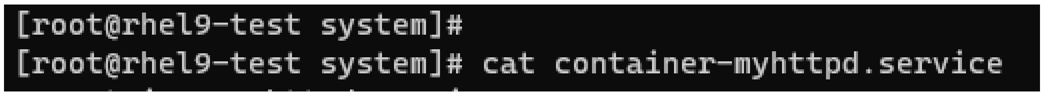
6. We will generate container service unit file under systemd & verify it (container-myhttpd.service)-



```
[root@rhel9-test system]# podman generate systemd --new --name myhttpd --files
/etc/systemd/system/container-myhttpd.service
[root@rhel9-test system]#
[root@rhel9-test system]# ll
total 16
drwxr-xr-x. 2 root root   65 Dec 29 17:54  basic.target.wants
drwxr-xr-x. 2 root root   31 Dec 29 17:53  bluetooth.target.wants
-rw-r--r--. 1 root root  760 Jan  2 14:55  container-myhttpd.service
lrwxrwxrwx. 1 root root   37 Dec 29 17:53  ctrl-alt-del.target -> /usr/lib/systemd/system/reboot.target
lrwxrwxrwx. 1 root root   41 Dec 29 17:53  dbus-org.bluez.service -> /usr/lib/systemd/system/bluetooth.service
lrwxrwxrwx. 1 root root   41 Dec 29 17:54  dbus-org.fedoraproject.FirewallD1.service -> /usr/lib/systemd/system/firewalld.service
lrwxrwxrwx. 1 root root   44 Dec 29 17:53  dbus-org.freedesktop.Avahi.service -> /usr/lib/systemd/system/avahi-daemon.service
lrwxrwxrwx. 1 root root   44 Dec 29 17:53  dbus-org.freedesktop.ModemManager1.service -> /usr/lib/systemd/system/ModemManager.service
lrwxrwxrwx. 1 root root   57 Dec 29 17:53  dbus-org.freedesktop.nm-dispatcher.service -> /usr/lib/systemd/system/NetworkManager-dispatcher.service
lrwxrwxrwx. 1 root root   43 Dec 29 17:53  dbus.service -> /usr/lib/systemd/system/dbus-broker.service
lrwxrwxrwx. 1 root root   40 Dec 29 17:59  default.target -> /usr/lib/systemd/system/graphical.target
drwxr-xr-x. 2 root root   76 Dec 31 19:04  default.target.wants
drwxr-xr-x. 2 root root   38 Dec 29 17:55 'dev-virtio\x2dports-org.qemu.guest_agent.0.device.wants'
lrwxrwxrwx. 1 root root   35 Dec 29 17:54  display-manager.service -> /usr/lib/systemd/system/gdm.service
drwxr-xr-x. 2 root root   32 Dec 29 17:53  getty.target.wants
drwxr-xr-x. 2 root root  181 Dec 29 17:54  graphical.target.wants
drwxr-xr-x. 2 root root   36 Dec 29 17:53  local-fs.target.wants
drwxr-xr-x. 2 root root 4096 Dec 29 17:55  multi-user.target.wants
drwxr-xr-x. 2 root root   48 Dec 29 17:53  network-online.target.wants
drwxr-xr-x. 2 root root   26 Dec 29 17:53  printer.target.wants
-rw-r--r--. 1 root root  772 Dec 31 19:04  redis-container.service
drwxr-xr-x. 2 root root   27 Dec 29 17:53  remote-fs.target.wants
drwxr-xr-x. 2 root root  186 Dec 29 17:54  sockets.target.wants
drwxr-xr-x. 2 root root 4096 Dec 29 17:53  sysinit.target.wants
drwxr-xr-x. 2 root root   86 Dec 29 17:55  timers.target.wants
drwxr-xr-x. 2 root root   29 Dec 29 17:54  vmtoolsd.service.requires
[root@rhel9-test system]#
```

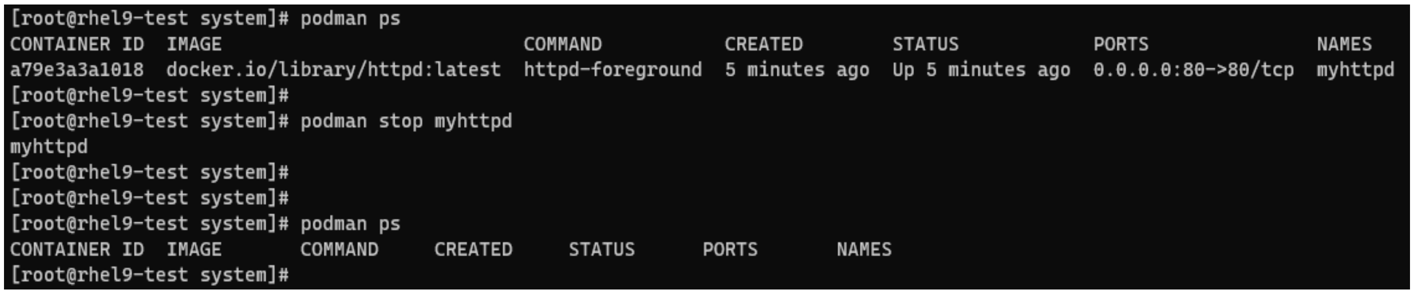
Note: Where ever we run this command, it create file in that location only.

7. We can check this file content & it is same as the output we receive from step 5-



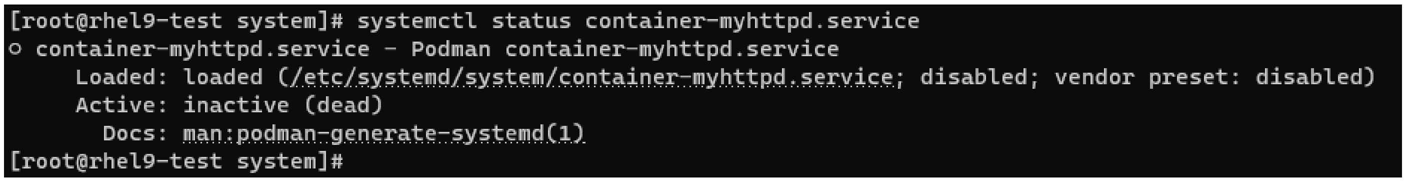
```
[root@rhel9-test system]#
[root@rhel9-test system]# cat container-myhttpd.service
```

8. Verify current status of httpd container image, stop it & verify again to go further-



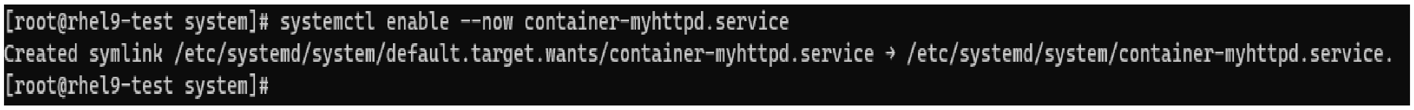
```
[root@rhel9-test system]# podman ps
CONTAINER ID  IMAGE                           COMMAND           CREATED        STATUS            PORTS                 NAMES
a79e3a3a1018  docker.io/library/httpd:latest  httpd-foreground  5 minutes ago  Up 5 minutes ago  0.0.0.0:80->80/tcp    myhttpd
[root@rhel9-test system]#
[root@rhel9-test system]# podman stop myhttpd
myhttpd
[root@rhel9-test system]#
[root@rhel9-test system]#
[root@rhel9-test system]# podman ps
CONTAINER ID  IMAGE       COMMAND     CREATED     STATUS      PORTS       NAMES
[root@rhel9-test system]#
```

9. Verify service status for newly created container service under systemd-



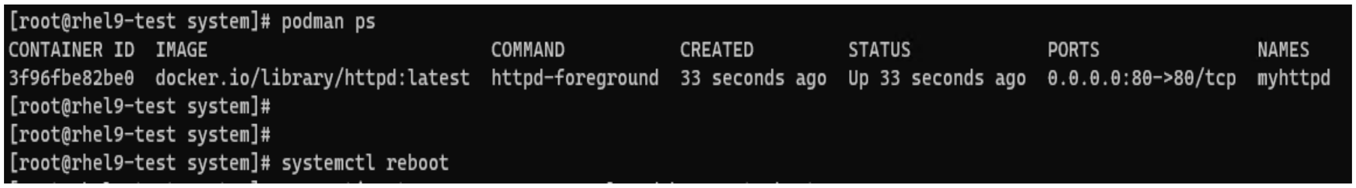
```
[root@rhel9-test system]# systemctl status container-myhttpd.service
○ container-myhttpd.service - Podman container-myhttpd.service
     Loaded: loaded (/etc/systemd/system/container-myhttpd.service; disabled; vendor preset: disabled)
     Active: inactive (dead)
       Docs: man:podman-generate-systemd(1)
[root@rhel9-test system]#
```

10. Start & enable this service-



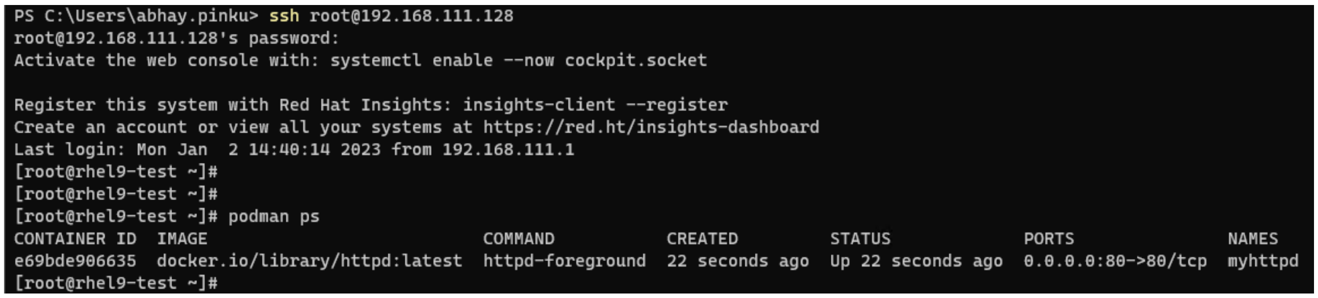
```
[root@rhel9-test system]# systemctl enable --now container-myhttpd.service
Created symlink /etc/systemd/system/default.target.wants/container-myhttpd.service → /etc/systemd/system/container-myhttpd.service.
[root@rhel9-test system]#
```

11. Now, check the container image status. We stopped it previously. Now it should be up after starting service. After that reboot server to check whether it withstand server reboot or not-



```
[root@rhel9-test system]# podman ps
CONTAINER ID  IMAGE                           COMMAND           CREATED         STATUS             PORTS               NAMES
3f96fbe82be0  docker.io/library/httpd:latest  httpd-foreground  33 seconds ago  Up 33 seconds ago  0.0.0.0:80->80/tcp  myhttpd
[root@rhel9-test system]#
[root@rhel9-test system]#
[root@rhel9-test system]# systemctl reboot
```

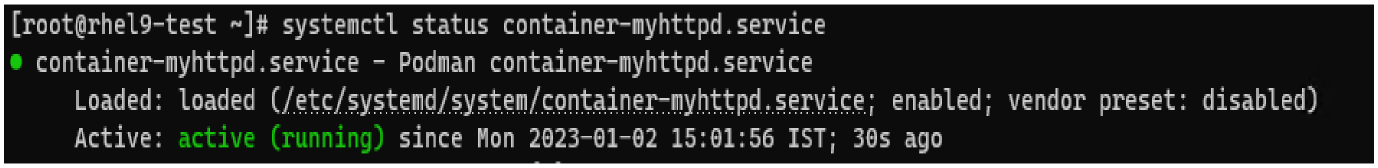
12. After rebooting, check the status of container-



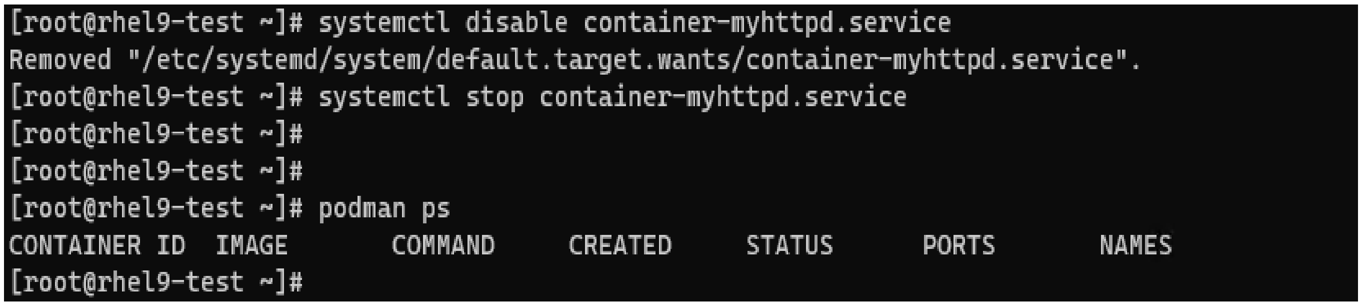
```
PS C:\Users\abhay.pinku> ssh root@192.168.111.128
root@192.168.111.128's password:
Activate the web console with: systemctl enable --now cockpit.socket

Register this system with Red Hat Insights: insights-client --register
Create an account or view all your systems at https://red.ht/insights-dashboard
Last login: Mon Jan  2 14:40:14 2023 from 192.168.111.1
[root@rhel9-test ~]#
[root@rhel9-test ~]#
[root@rhel9-test ~]# podman ps
CONTAINER ID  IMAGE                           COMMAND           CREATED         STATUS             PORTS               NAMES
e69bde906635  docker.io/library/httpd:latest  httpd-foreground  22 seconds ago  Up 22 seconds ago  0.0.0.0:80->80/tcp  myhttpd
[root@rhel9-test ~]#
```

It is started automatically.

13. Verify service status-

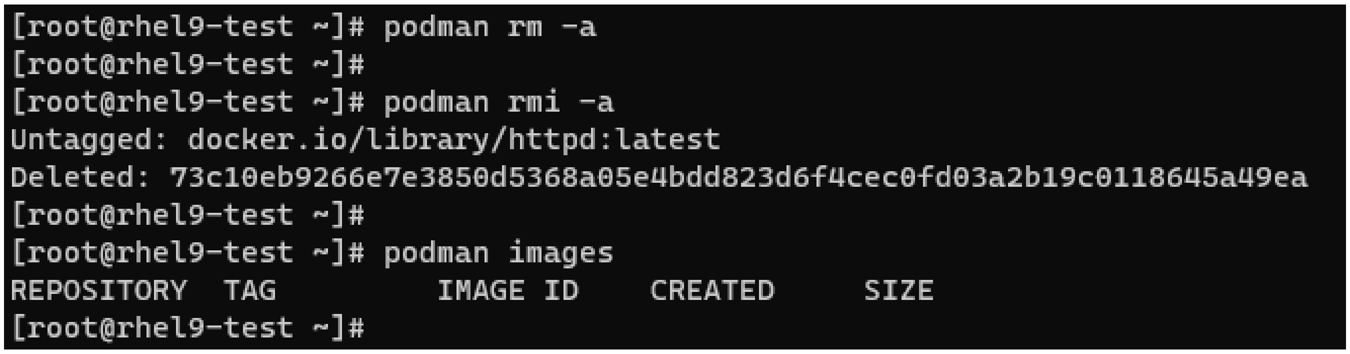

```
[root@rhel9-test ~]# systemctl status container-myhttpd.service
● container-myhttpd.service - Podman container-myhttpd.service
     Loaded: loaded (/etc/systemd/system/container-myhttpd.service; enabled; vendor preset: disabled)
     Active: active (running) since Mon 2023-01-02 15:01:56 IST; 30s ago
```

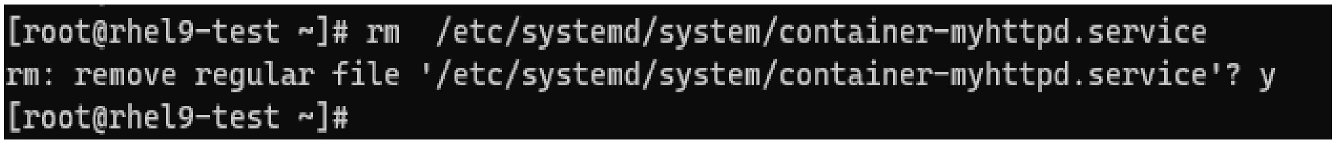
14. Our lab is completed. Now we will stop the service, verify container running status-


```
[root@rhel9-test ~]# systemctl disable container-myhttpd.service
Removed "/etc/systemd/system/default.target.wants/container-myhttpd.service".
[root@rhel9-test ~]# systemctl stop container-myhttpd.service
[root@rhel9-test ~]#
[root@rhel9-test ~]#
[root@rhel9-test ~]# podman ps
CONTAINER ID  IMAGE       COMMAND     CREATED     STATUS      PORTS       NAMES
[root@rhel9-test ~]#
```

Container is stopped too as we stopped service.

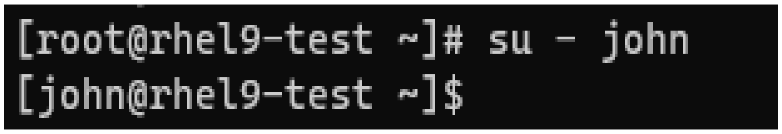
15. Now remove all images & created service for the container. Verify the same-


```
[root@rhel9-test ~]# podman rm -a
[root@rhel9-test ~]#
[root@rhel9-test ~]# podman rmi -a
Untagged: docker.io/library/httpd:latest
Deleted: 73c10eb9266e7e3850d5368a05e4bdd823d6f4cec0fd03a2b19c0118645a49ea
[root@rhel9-test ~]#
[root@rhel9-test ~]# podman images
REPOSITORY  TAG         IMAGE ID    CREATED     SIZE
[root@rhel9-test ~]#
```

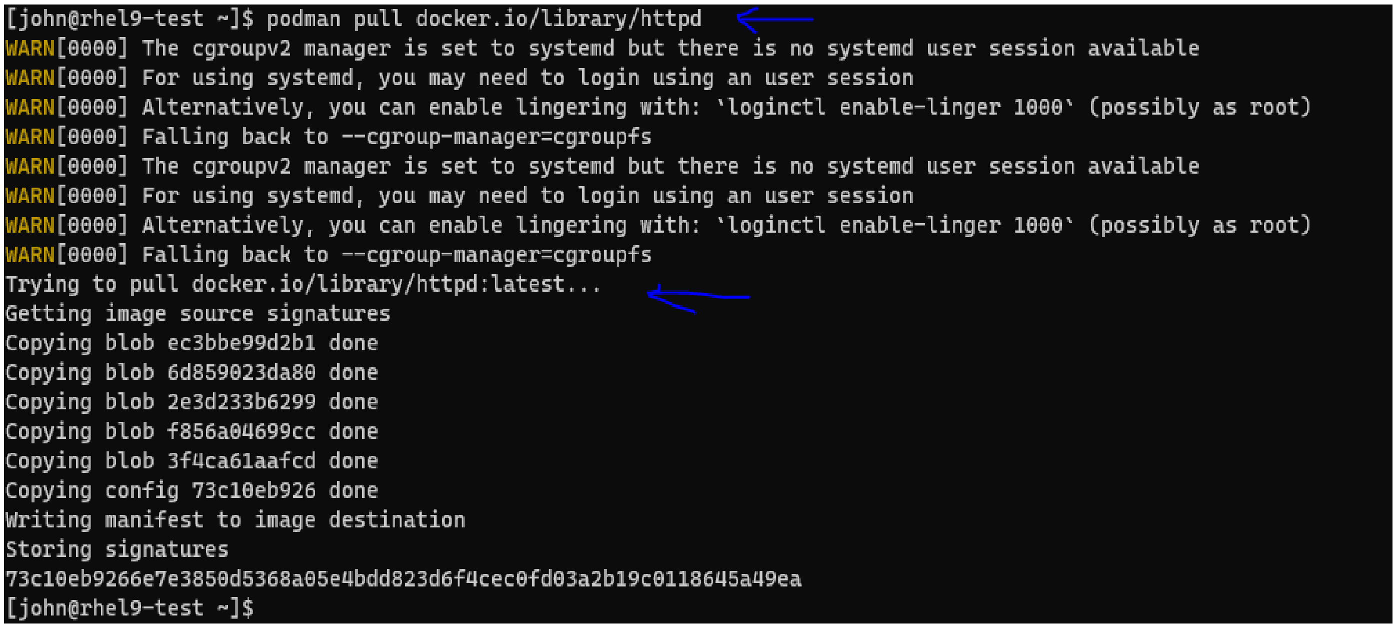

```
[root@rhel9-test ~]# rm  /etc/systemd/system/container-myhttpd.service
rm: remove regular file '/etc/systemd/system/container-myhttpd.service'? y
[root@rhel9-test ~]#
```

Root Less Mode:

16. Login with a standard user-

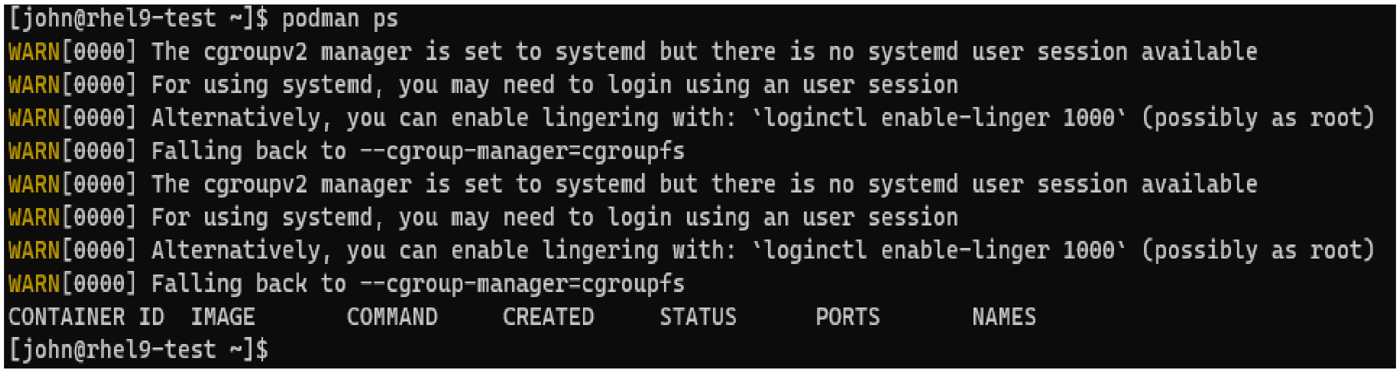

```
[root@rhel9-test ~]# su - john
[john@rhel9-test ~]$
```

17. Pull httpd container image-



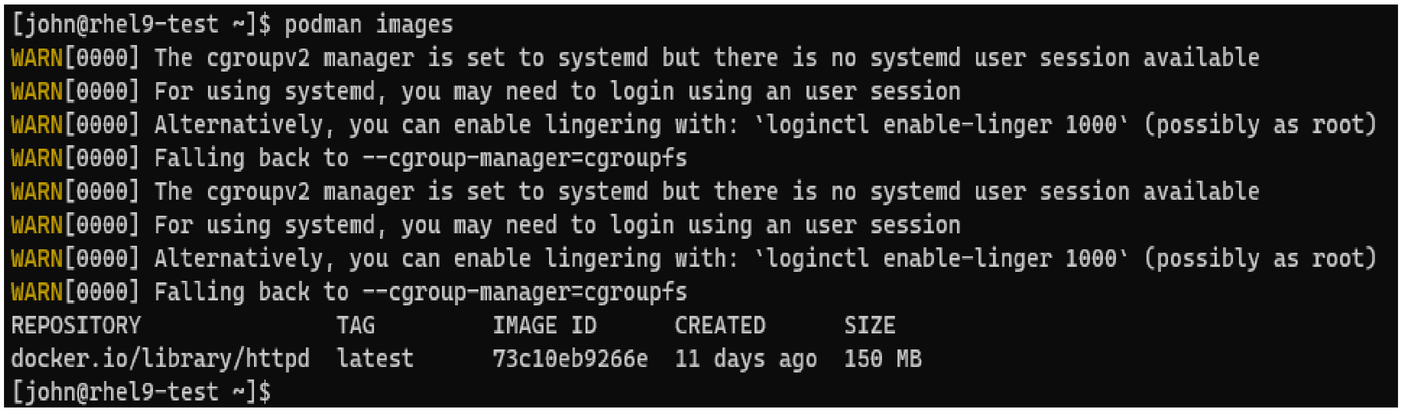
```
[john@rhel9-test ~]$ podman pull docker.io/library/httpd
WARN[0000] The cgroupv2 manager is set to systemd but there is no systemd user session available
WARN[0000] For using systemd, you may need to login using an user session
WARN[0000] Alternatively, you can enable lingering with: `loginctl enable-linger 1000` (possibly as root)
WARN[0000] Falling back to --cgroup-manager=cgroupfs
WARN[0000] The cgroupv2 manager is set to systemd but there is no systemd user session available
WARN[0000] For using systemd, you may need to login using an user session
WARN[0000] Alternatively, you can enable lingering with: `loginctl enable-linger 1000` (possibly as root)
WARN[0000] Falling back to --cgroup-manager=cgroupfs
Trying to pull docker.io/library/httpd:latest...
Getting image source signatures
Copying blob ec3bbe99d2b1 done
Copying blob 6d859023da80 done
Copying blob 2e3d233b6299 done
Copying blob f856a04699cc done
Copying blob 3f4ca61aafcd done
Copying config 73c10eb926 done
Writing manifest to image destination
Storing signatures
73c10eb9266e7e3850d5368a05e4bdd823d6f4cec0fd03a2b19c0118645a49ea
[john@rhel9-test ~]$
```

18. Check is any container is running previously-



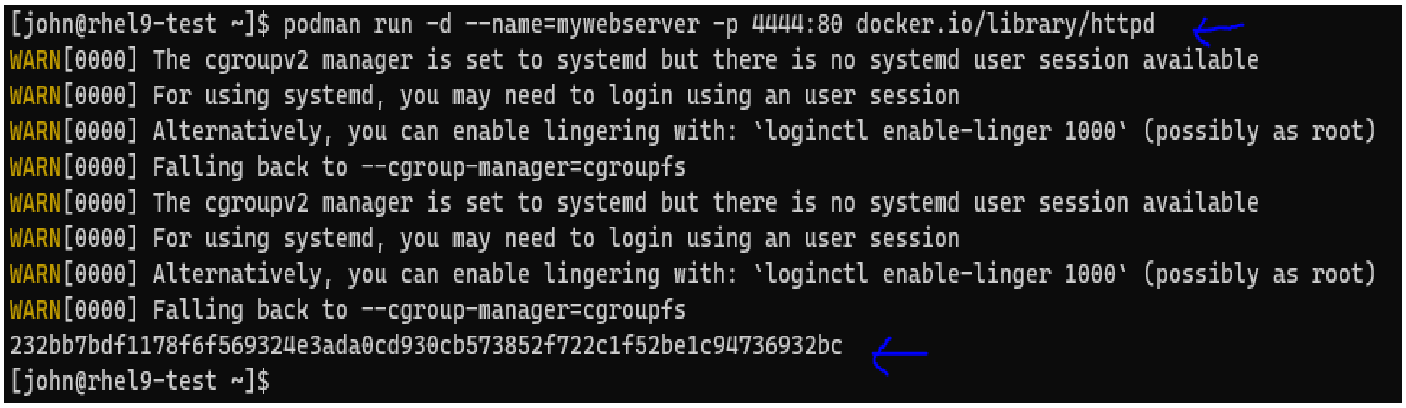
```
[john@rhel9-test ~]$ podman ps
WARN[0000] The cgroupv2 manager is set to systemd but there is no systemd user session available
WARN[0000] For using systemd, you may need to login using an user session
WARN[0000] Alternatively, you can enable lingering with: `loginctl enable-linger 1000` (possibly as root)
WARN[0000] Falling back to --cgroup-manager=cgroupfs
WARN[0000] The cgroupv2 manager is set to systemd but there is no systemd user session available
WARN[0000] For using systemd, you may need to login using an user session
WARN[0000] Alternatively, you can enable lingering with: `loginctl enable-linger 1000` (possibly as root)
WARN[0000] Falling back to --cgroup-manager=cgroupfs
CONTAINER ID  IMAGE       COMMAND     CREATED     STATUS      PORTS       NAMES
[john@rhel9-test ~]$
```

19. Check available container images-



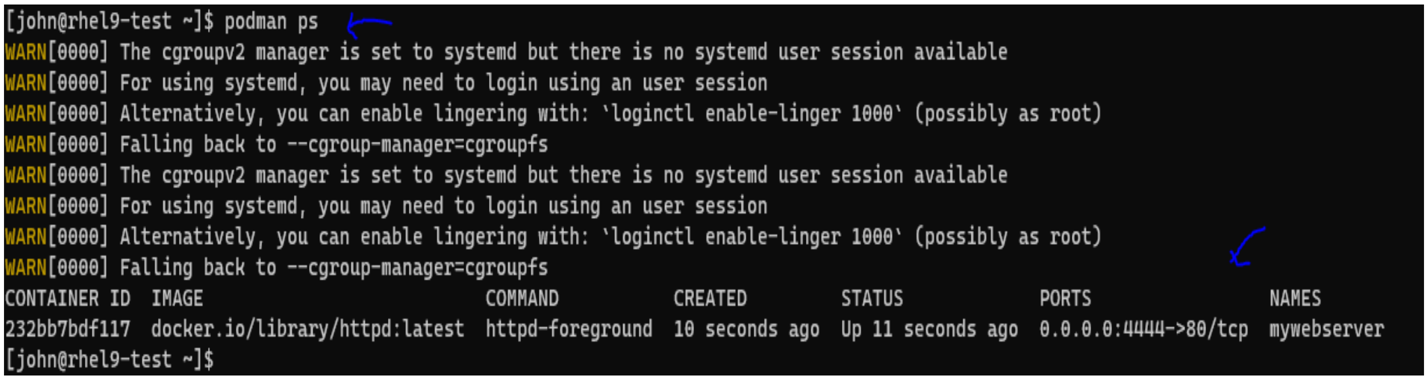
```
[john@rhel9-test ~]$ podman images
WARN[0000] The cgroupv2 manager is set to systemd but there is no systemd user session available
WARN[0000] For using systemd, you may need to login using an user session
WARN[0000] Alternatively, you can enable lingering with: `loginctl enable-linger 1000` (possibly as root)
WARN[0000] Falling back to --cgroup-manager=cgroupfs
WARN[0000] The cgroupv2 manager is set to systemd but there is no systemd user session available
WARN[0000] For using systemd, you may need to login using an user session
WARN[0000] Alternatively, you can enable lingering with: `loginctl enable-linger 1000` (possibly as root)
WARN[0000] Falling back to --cgroup-manager=cgroupfs
REPOSITORY               TAG         IMAGE ID      CREATED      SIZE
docker.io/library/httpd  latest      73c10eb9266e  11 days ago  150 MB
[john@rhel9-test ~]$
```

20. Run this container image in background at port greater than 1024-



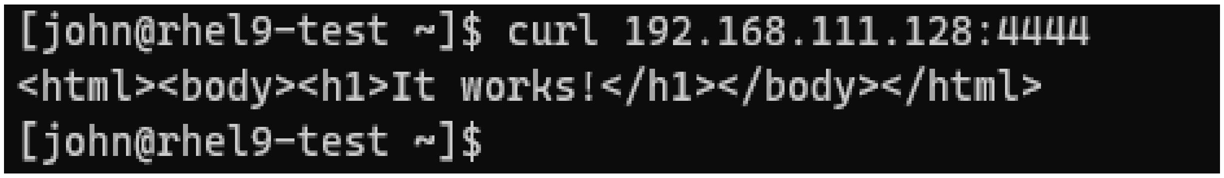
```
[john@rhel9-test ~]$ podman run -d --name=mywebserver -p 4444:80 docker.io/library/httpd
WARN[0000] The cgroupv2 manager is set to systemd but there is no systemd user session available
WARN[0000] For using systemd, you may need to login using an user session
WARN[0000] Alternatively, you can enable lingering with: `loginctl enable-linger 1000` (possibly as root)
WARN[0000] Falling back to --cgroup-manager=cgroupfs
WARN[0000] The cgroupv2 manager is set to systemd but there is no systemd user session available
WARN[0000] For using systemd, you may need to login using an user session
WARN[0000] Alternatively, you can enable lingering with: `loginctl enable-linger 1000` (possibly as root)
WARN[0000] Falling back to --cgroup-manager=cgroupfs
232bb7bdf1178f6f569324e3ada0cd930cb573852f722c1f52be1c94736932bc
[john@rhel9-test ~]$
```

21. Verify the container running status-



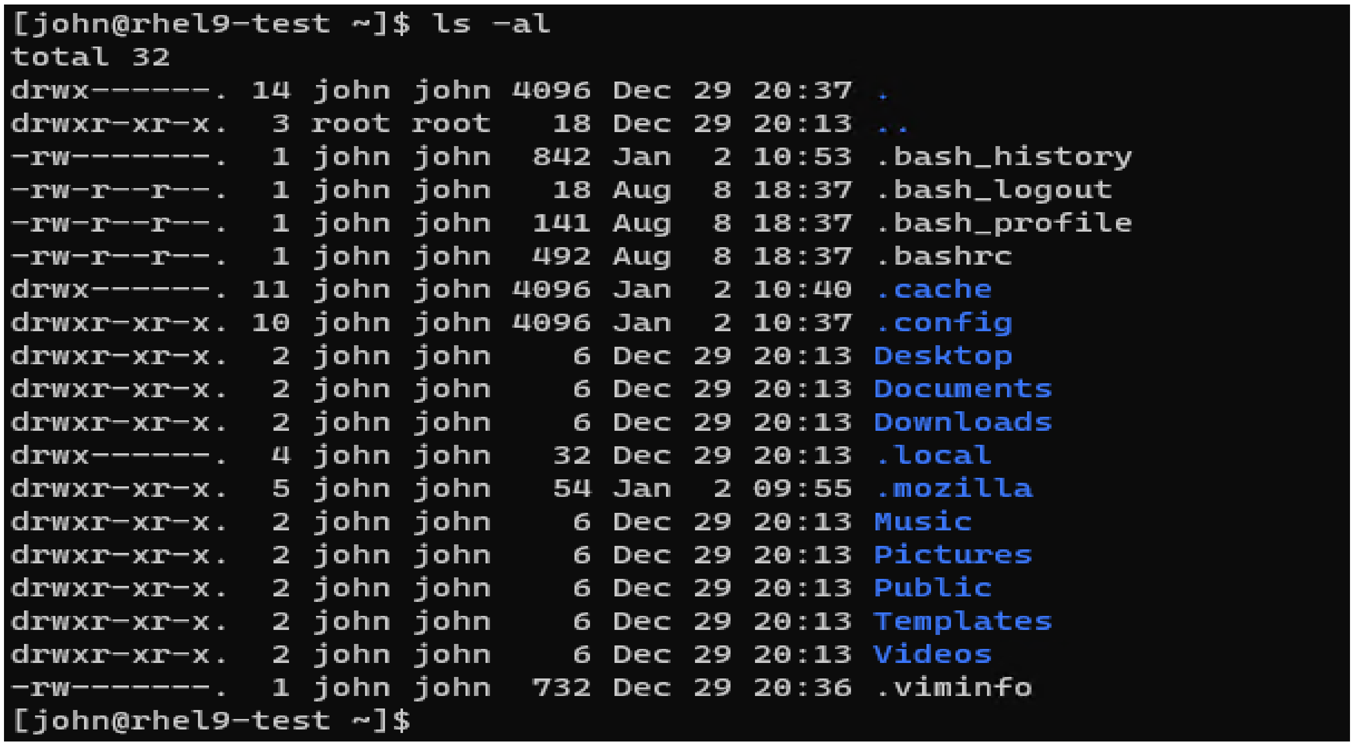
```
[john@rhel9-test ~]$ podman ps
WARN[0000] The cgroupv2 manager is set to systemd but there is no systemd user session available
WARN[0000] For using systemd, you may need to login using an user session
WARN[0000] Alternatively, you can enable lingering with: `loginctl enable-linger 1000` (possibly as root)
WARN[0000] Falling back to --cgroup-manager=cgroupfs
WARN[0000] The cgroupv2 manager is set to systemd but there is no systemd user session available
WARN[0000] For using systemd, you may need to login using an user session
WARN[0000] Alternatively, you can enable lingering with: `loginctl enable-linger 1000` (possibly as root)
WARN[0000] Falling back to --cgroup-manager=cgroupfs
CONTAINER ID  IMAGE                           COMMAND           CREATED         STATUS             PORTS                 NAMES
232bb7bdf117  docker.io/library/httpd:latest  httpd-foreground  10 seconds ago  Up 11 seconds ago  0.0.0.0:4444->80/tcp  mywebserver
[john@rhel9-test ~]$
```

22. Verify web URL-



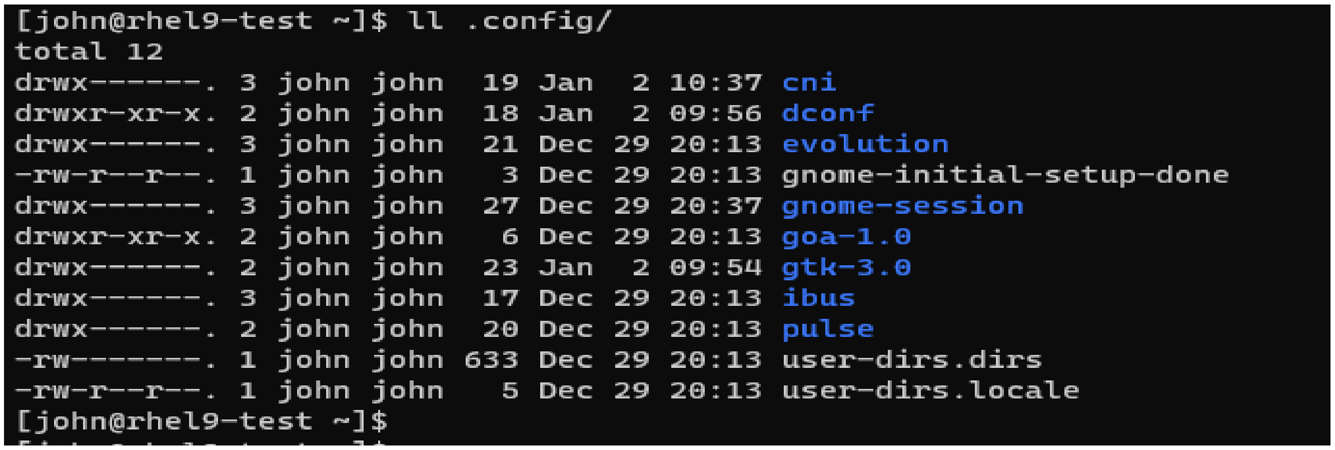
```
[john@rhel9-test ~]$ curl 192.168.111.128:4444
<html><body><h1>It works!</h1></body></html>
[john@rhel9-test ~]$
```

23. This lab is for keep container image running even after server reboot in root less mode. First look for .config directory under user's home directory-



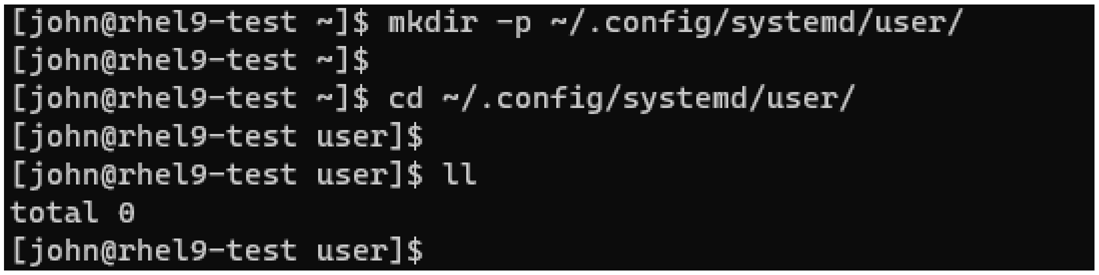
```
[john@rhel9-test ~]$ ls -al
total 32
drwx------. 14 john john 4096 Dec 29 20:37 .
drwxr-xr-x.  3 root root   18 Dec 29 20:13 ..
-rw-------.  1 john john  842 Jan  2 10:53 .bash_history
-rw-r--r--.  1 john john   18 Aug  8 18:37 .bash_logout
-rw-r--r--.  1 john john  141 Aug  8 18:37 .bash_profile
-rw-r--r--.  1 john john  492 Aug  8 18:37 .bashrc
drwx------. 11 john john 4096 Jan  2 10:40 .cache
drwxr-xr-x. 10 john john 4096 Jan  2 10:37 .config
drwxr-xr-x.  2 john john    6 Dec 29 20:13 Desktop
drwxr-xr-x.  2 john john    6 Dec 29 20:13 Documents
drwxr-xr-x.  2 john john    6 Dec 29 20:13 Downloads
drwx------.  4 john john   32 Dec 29 20:13 .local
drwxr-xr-x.  5 john john   54 Jan  2 09:55 .mozilla
drwxr-xr-x.  2 john john    6 Dec 29 20:13 Music
drwxr-xr-x.  2 john john    6 Dec 29 20:13 Pictures
drwxr-xr-x.  2 john john    6 Dec 29 20:13 Public
drwxr-xr-x.  2 john john    6 Dec 29 20:13 Templates
drwxr-xr-x.  2 john john    6 Dec 29 20:13 Videos
-rw-------.  1 john john  732 Dec 29 20:36 .viminfo
[john@rhel9-test ~]$
```

24. We will list its content-



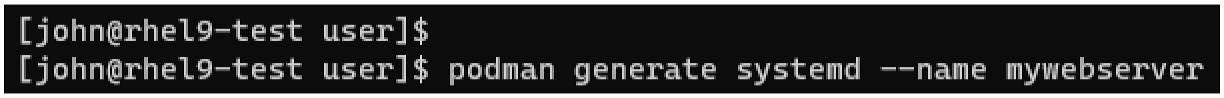
```
[john@rhel9-test ~]$ ll .config/
total 12
drwx------. 3 john john  19 Jan  2 10:37 cni
drwxr-xr-x. 2 john john  18 Jan  2 09:56 dconf
drwx------. 3 john john  21 Dec 29 20:13 evolution
-rw-r--r--. 1 john john   3 Dec 29 20:13 gnome-initial-setup-done
drwx------. 3 john john  27 Dec 29 20:37 gnome-session
drwxr-xr-x. 2 john john   6 Dec 29 20:13 goa-1.0
drwx------. 2 john john  23 Jan  2 09:54 gtk-3.0
drwx------. 3 john john  17 Dec 29 20:13 ibus
drwx------. 2 john john  20 Dec 29 20:13 pulse
-rw-------. 1 john john 633 Dec 29 20:13 user-dirs.dirs
-rw-r--r--. 1 john john   5 Dec 29 20:13 user-dirs.locale
[john@rhel9-test ~]$
```

25. Now create one directory in parent-child form-



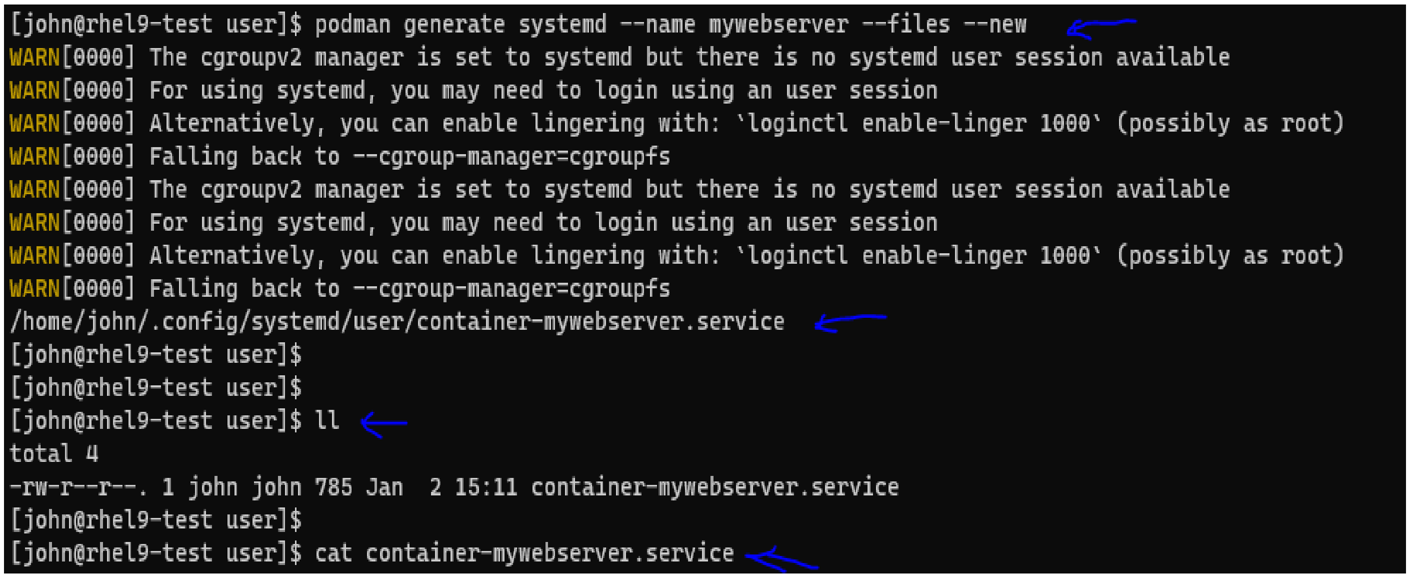
```
[john@rhel9-test ~]$ mkdir -p ~/.config/systemd/user/
[john@rhel9-test ~]$
[john@rhel9-test ~]$ cd ~/.config/systemd/user/
[john@rhel9-test user]$
[john@rhel9-test user]$ ll
total 0
[john@rhel9-test user]$
```

26. We need to generate unit file as we did in root full mode. To just see content of systemd unit file, run below command-



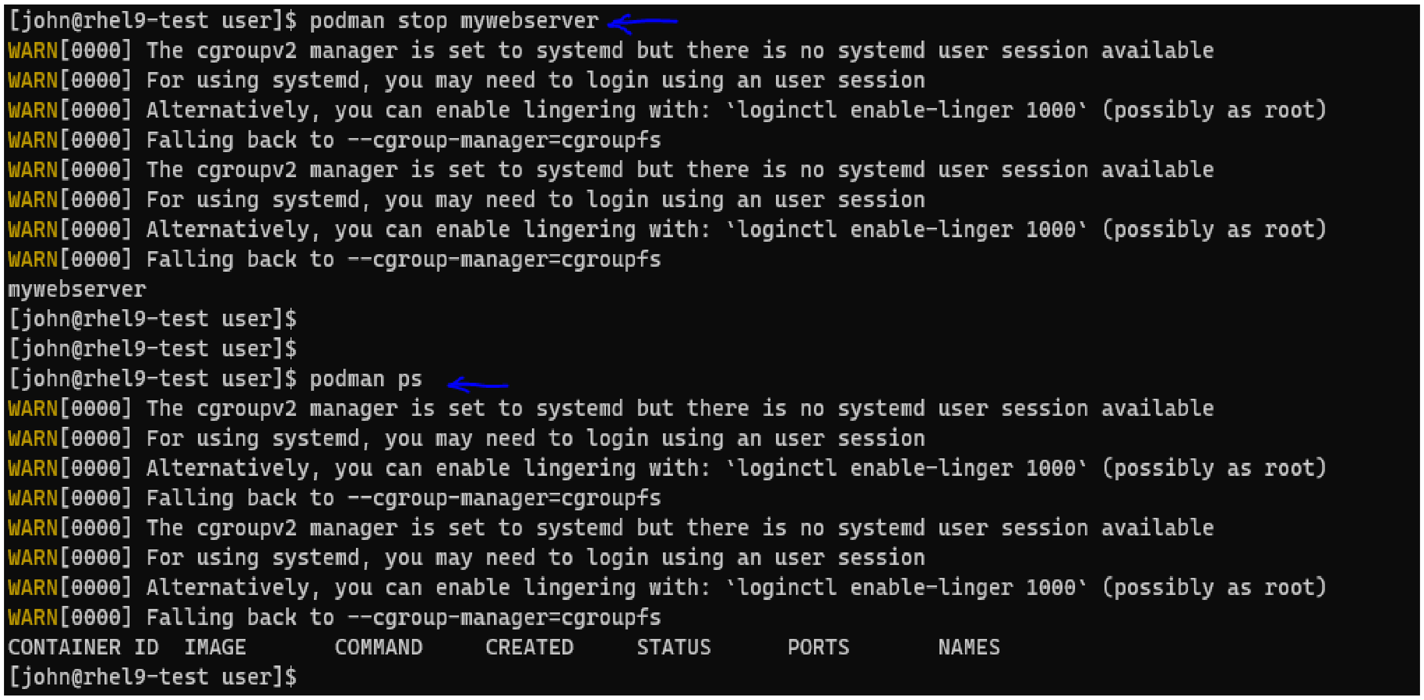
```
[john@rhel9-test user]$
[john@rhel9-test user]$ podman generate systemd --name mywebserver
```

27. Now generate that unit file in newly created child directory & we can check its content as well-



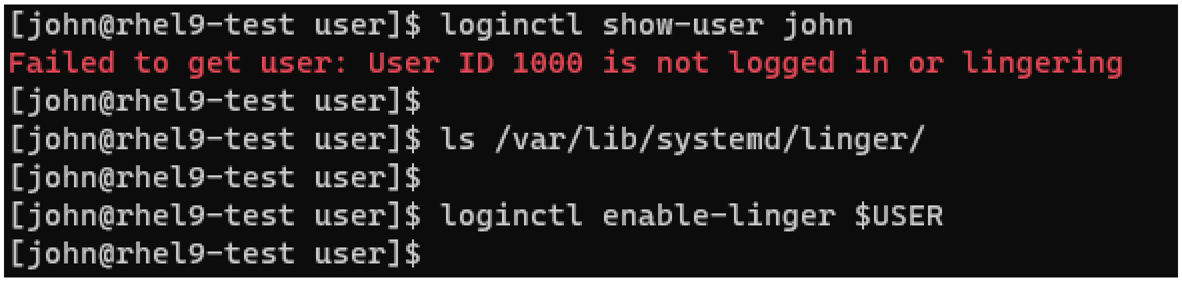
```
[john@rhel9-test user]$ podman generate systemd --name mywebserver --files --new
WARN[0000] The cgroupv2 manager is set to systemd but there is no systemd user session available
WARN[0000] For using systemd, you may need to login using an user session
WARN[0000] Alternatively, you can enable lingering with: `loginctl enable-linger 1000` (possibly as root)
WARN[0000] Falling back to --cgroup-manager=cgroupfs
WARN[0000] The cgroupv2 manager is set to systemd but there is no systemd user session available
WARN[0000] For using systemd, you may need to login using an user session
WARN[0000] Alternatively, you can enable lingering with: `loginctl enable-linger 1000` (possibly as root)
WARN[0000] Falling back to --cgroup-manager=cgroupfs
/home/john/.config/systemd/user/container-mywebserver.service
[john@rhel9-test user]$
[john@rhel9-test user]$
[john@rhel9-test user]$ ll
total 4
-rw-r--r--. 1 john john 785 Jan  2 15:11 container-mywebserver.service
[john@rhel9-test user]$
[john@rhel9-test user]$ cat container-mywebserver.service
```

28. Stop the webserver & verify the container status-



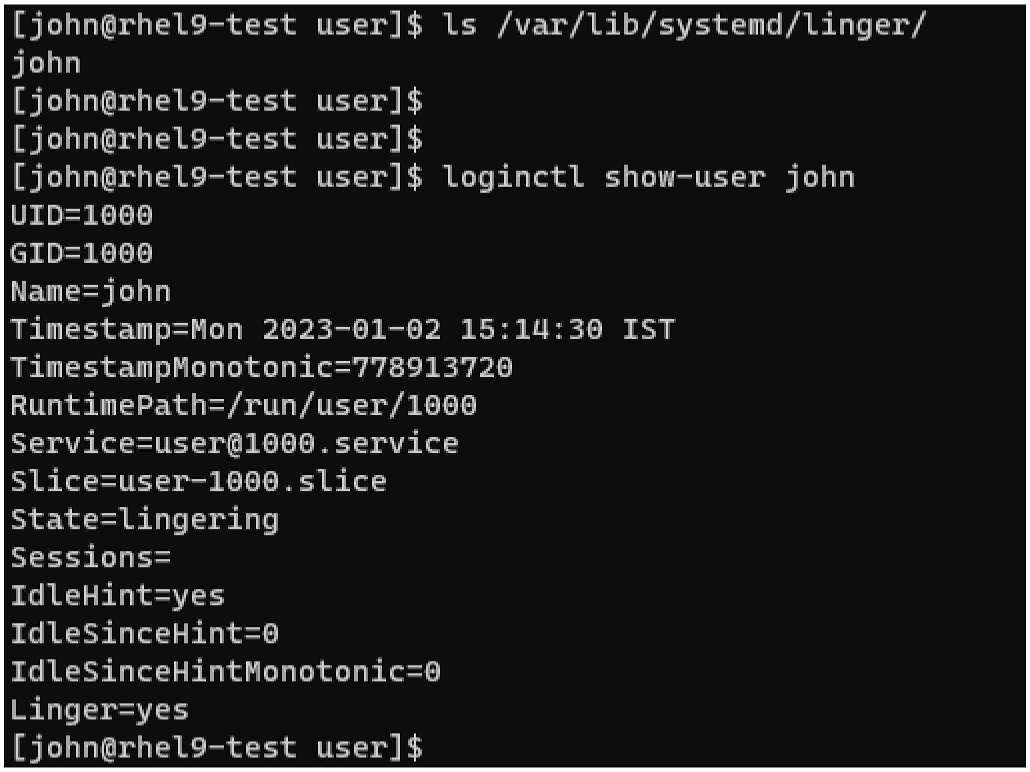
```
[john@rhel9-test user]$ podman stop mywebserver
WARN[0000] The cgroupv2 manager is set to systemd but there is no systemd user session available
WARN[0000] For using systemd, you may need to login using an user session
WARN[0000] Alternatively, you can enable lingering with: `loginctl enable-linger 1000` (possibly as root)
WARN[0000] Falling back to --cgroup-manager=cgroupfs
WARN[0000] The cgroupv2 manager is set to systemd but there is no systemd user session available
WARN[0000] For using systemd, you may need to login using an user session
WARN[0000] Alternatively, you can enable lingering with: `loginctl enable-linger 1000` (possibly as root)
WARN[0000] Falling back to --cgroup-manager=cgroupfs
mywebserver
[john@rhel9-test user]$
[john@rhel9-test user]$
[john@rhel9-test user]$ podman ps
WARN[0000] The cgroupv2 manager is set to systemd but there is no systemd user session available
WARN[0000] For using systemd, you may need to login using an user session
WARN[0000] Alternatively, you can enable lingering with: `loginctl enable-linger 1000` (possibly as root)
WARN[0000] Falling back to --cgroup-manager=cgroupfs
WARN[0000] The cgroupv2 manager is set to systemd but there is no systemd user session available
WARN[0000] For using systemd, you may need to login using an user session
WARN[0000] Alternatively, you can enable lingering with: `loginctl enable-linger 1000` (possibly as root)
WARN[0000] Falling back to --cgroup-manager=cgroupfs
CONTAINER ID  IMAGE       COMMAND     CREATED     STATUS      PORTS       NAMES
[john@rhel9-test user]$
```

29. To check whether this user is allowed for lingering or not. If not, he will not be able to define systemd unit. We can list linger file content, but it will show empty if no user allowed to define systemd unit. To fix it, we need to enable user for lingering as shown-



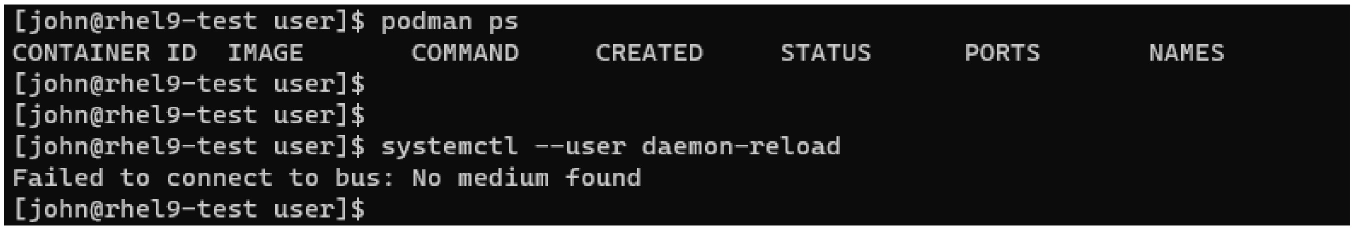
```
[john@rhel9-test user]$ loginctl show-user john
Failed to get user: User ID 1000 is not logged in or lingering
[john@rhel9-test user]$
[john@rhel9-test user]$ ls /var/lib/systemd/linger/
[john@rhel9-test user]$
[john@rhel9-test user]$ loginctl enable-linger $USER
[john@rhel9-test user]$
```

30. Now verify linger file content. It will show this user name i.e this user has permission to define systemd unit. We can also check this user detail which was not showing last time-



```
[john@rhel9-test user]$ ls /var/lib/systemd/linger/
john
[john@rhel9-test user]$
[john@rhel9-test user]$
[john@rhel9-test user]$ loginctl show-user john
UID=1000
GID=1000
Name=john
Timestamp=Mon 2023-01-02 15:14:30 IST
TimestampMonotonic=778913720
RuntimePath=/run/user/1000
Service=user@1000.service
Slice=user-1000.slice
State=lingering
Sessions=
IdleHint=yes
IdleSinceHint=0
IdleSinceHintMonotonic=0
Linger=yes
[john@rhel9-test user]$
```

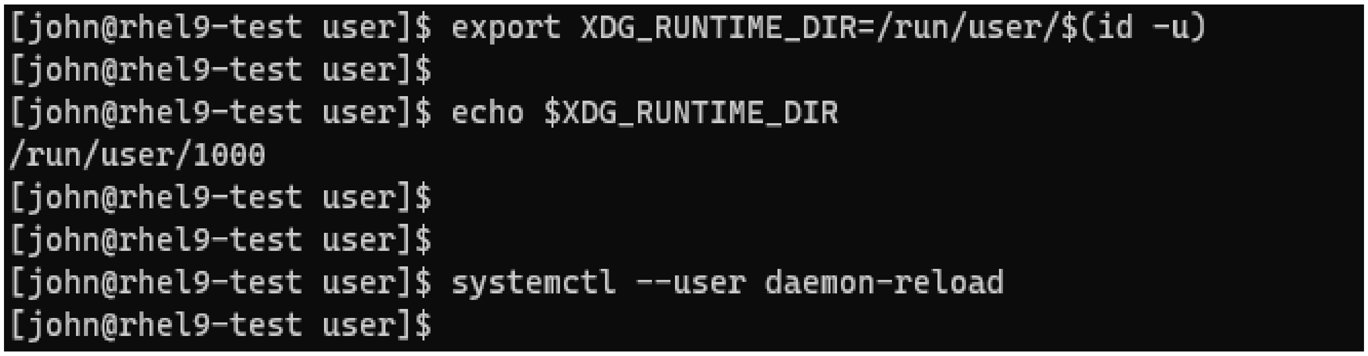
31. Check for any running container. Now we will reload daemon for user-



```
[john@rhel9-test user]$ podman ps
CONTAINER ID  IMAGE       COMMAND     CREATED     STATUS      PORTS       NAMES
[john@rhel9-test user]$
[john@rhel9-test user]$
[john@rhel9-test user]$ systemctl --user daemon-reload
Failed to connect to bus: No medium found
[john@rhel9-test user]$
```

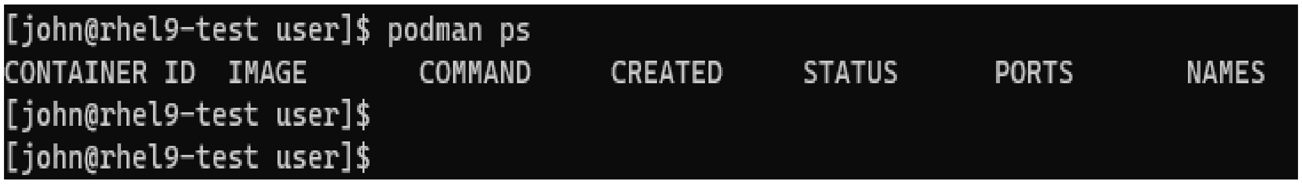
It is failing. This is the method use to define systemd unit in root less mode. This issue arises after RHEL 8.5 version.

32. To solve this, we will define a variable & export it so that it can be available in other shells as well-

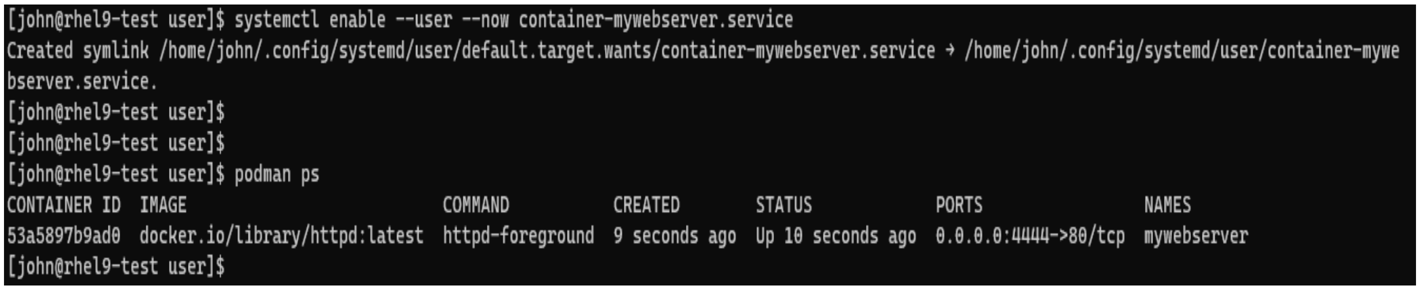

```
[john@rhel9-test user]$ export XDG_RUNTIME_DIR=/run/user/$(id -u)
[john@rhel9-test user]$
[john@rhel9-test user]$ echo $XDG_RUNTIME_DIR
/run/user/1000
[john@rhel9-test user]$
[john@rhel9-test user]$
[john@rhel9-test user]$ systemctl --user daemon-reload
[john@rhel9-test user]$
```

We can see this variable content. It shows user-id for current user i.e john here. Now we will again reload daemon for this user & this time it succeeds.

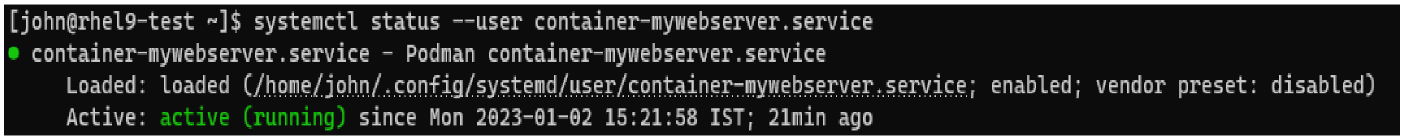
33. Again check for any running container-


```
[john@rhel9-test user]$ podman ps
CONTAINER ID  IMAGE       COMMAND     CREATED     STATUS      PORTS       NAMES
[john@rhel9-test user]$
[john@rhel9-test user]$
```

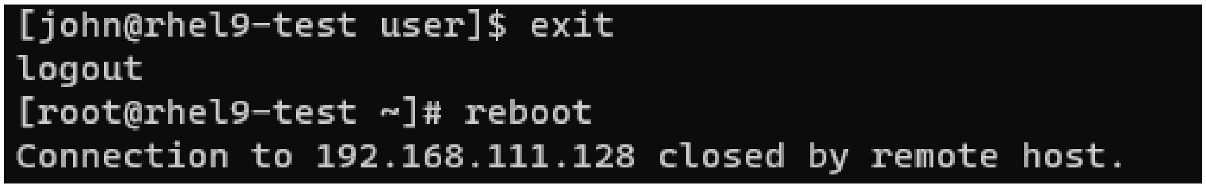
34. Now we will start & enable created systemd unit & check for the container status-


```
[john@rhel9-test user]$ systemctl enable --user --now container-mywebserver.service
Created symlink /home/john/.config/systemd/user/default.target.wants/container-mywebserver.service → /home/john/.config/systemd/user/container-mywe
bserver.service.
[john@rhel9-test user]$
[john@rhel9-test user]$
[john@rhel9-test user]$ podman ps
CONTAINER ID  IMAGE                           COMMAND           CREATED        STATUS             PORTS                 NAMES
53a5897b9ad0  docker.io/library/httpd:latest  httpd-foreground  9 seconds ago  Up 10 seconds ago  0.0.0.0:4444->80/tcp  mywebserver
[john@rhel9-test user]$
```

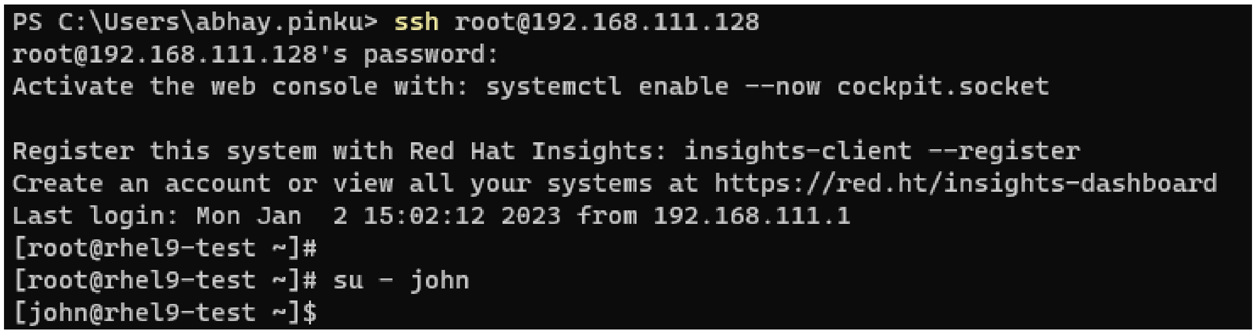
35. Check its service status-


```
[john@rhel9-test ~]$ systemctl status --user container-mywebserver.service
● container-mywebserver.service - Podman container-mywebserver.service
     Loaded: loaded (/home/john/.config/systemd/user/container-mywebserver.service; enabled; vendor preset: disabled)
     Active: active (running) since Mon 2023-01-02 15:21:58 IST; 21min ago
```

36. Next, we will exit from this user & reboot the server-

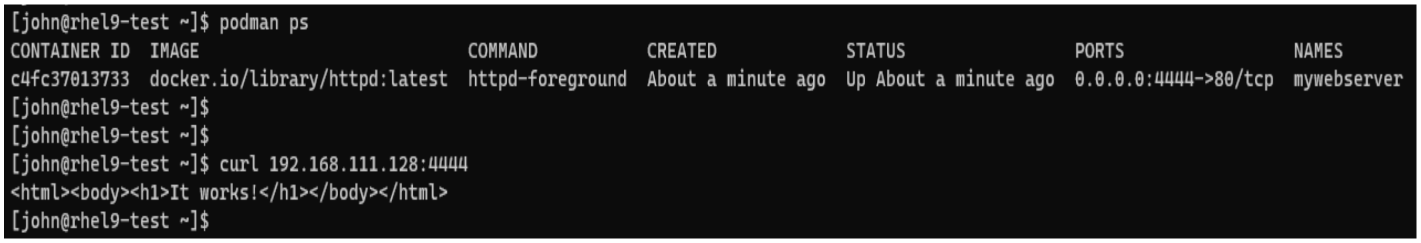

```
[john@rhel9-test user]$ exit
logout
[root@rhel9-test ~]# reboot
Connection to 192.168.111.128 closed by remote host.
```

37. We will login back with john user after server reboot-



```
PS C:\Users\abhay.pinku> ssh root@192.168.111.128
root@192.168.111.128's password:
Activate the web console with: systemctl enable --now cockpit.socket

Register this system with Red Hat Insights: insights-client --register
Create an account or view all your systems at https://red.ht/insights-dashboard
Last login: Mon Jan  2 15:02:12 2023 from 192.168.111.1
[root@rhel9-test ~]#
[root@rhel9-test ~]# su - john
[john@rhel9-test ~]$
```

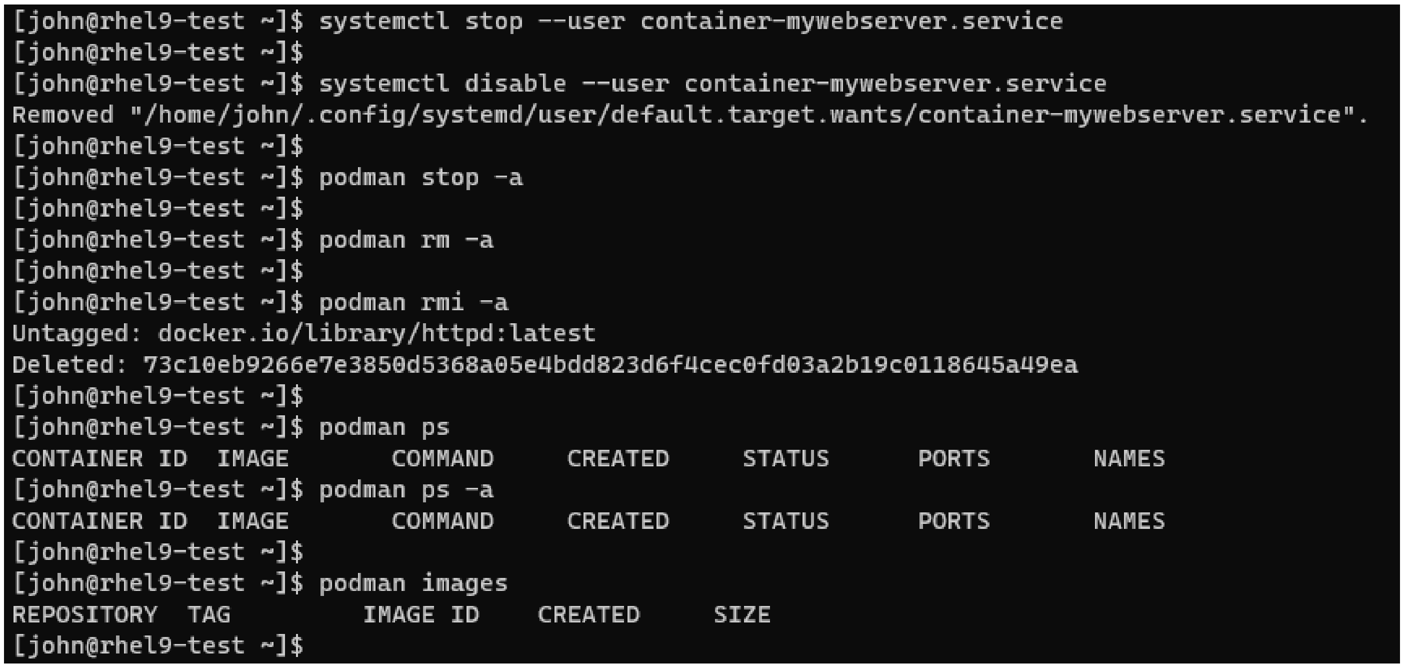
38. Verify the container running status & check the web URL-



```
[john@rhel9-test ~]$ podman ps
CONTAINER ID  IMAGE                           COMMAND           CREATED           STATUS               PORTS                 NAMES
c4fc37013733  docker.io/library/httpd:latest  httpd-foreground  About a minute ago  Up About a minute ago  0.0.0.0:4444->80/tcp  mywebserver
[john@rhel9-test ~]$
[john@rhel9-test ~]$
[john@rhel9-test ~]$ curl 192.168.111.128:4444
<html><body><h1>It works!</h1></body></html>
[john@rhel9-test ~]$
```

It is up & running even after server reboot. Thus we succeed in setting up root less container which withstand server reboot.

39. Now we will stop running container & remove it as well as remove container images-



```
[john@rhel9-test ~]$ systemctl stop --user container-mywebserver.service
[john@rhel9-test ~]$
[john@rhel9-test ~]$ systemctl disable --user container-mywebserver.service
Removed "/home/john/.config/systemd/user/default.target.wants/container-mywebserver.service".
[john@rhel9-test ~]$
[john@rhel9-test ~]$ podman stop -a
[john@rhel9-test ~]$
[john@rhel9-test ~]$ podman rm -a
[john@rhel9-test ~]$
[john@rhel9-test ~]$ podman rmi -a
Untagged: docker.io/library/httpd:latest
Deleted: 73c10eb9266e7e3850d5368a05e4bdd823d6f4cec0fd03a2b19c0118645a49ea
[john@rhel9-test ~]$
[john@rhel9-test ~]$ podman ps
CONTAINER ID  IMAGE       COMMAND     CREATED     STATUS      PORTS       NAMES
[john@rhel9-test ~]$ podman ps -a
CONTAINER ID  IMAGE       COMMAND     CREATED     STATUS      PORTS       NAMES
[john@rhel9-test ~]$
[john@rhel9-test ~]$ podman images
REPOSITORY  TAG         IMAGE ID    CREATED     SIZE
[john@rhel9-test ~]$
```

This is it for Lecture 6!!!