Proof of Concept Technical Solution

for the *Marconi Law Firm, LLC*.

(WordPress Website)

Project Background: Assume that you are an entrepreneur and that you own your own Information Technology (IT) consulting firm. You have recently acquired a new client called “Marconi Law Firm”. As part of your client’s contract agreement, you are to deliver full documentation for their upcoming WordPress Website Hosting project implementation. This documentation includes a Proof-of-Concept Technical Solution which documents all software, hardware, and network configuration details. Assume that the finished document will be used in-house by the Marconi Law Firm and will be referenced by their in-house IT department---after the project has been successfully completed.

Humberto Gonzalez

IT & Cybersecurity Solutions

Preface

This document will serve as proof of concept to Mr. Marconi for creating his WordPress website for his law firm and as audit documentation.

The purpose of audit documentation is to provide a comprehensive record of the organization's information technology infrastructure and security controls and processes. It plays a crucial role in providing transparency, accountability, and QA/QC regarding an organization's cybersecurity controls and practices. It enables organizations to demonstrate compliance, identify areas for improvement, and make informed decisions to strengthen their overall organizational cybersecurity.

Audit documentation serves several important purposes:

- **Compliance**: Evidence that an organization has undergone a thorough examination of its systems. It helps validate that the organization has implemented appropriate controls to protect its information systems and sensitive data.

- **Validation**: Verification of the effectiveness and adequacy of cybersecurity controls. It provides detailed information about the design, implementation, and operation of these controls, enabling reviewers to assess their reliability and identify any gaps or weaknesses.

- **Records Maintenance**: Historical record of cybersecurity audits conducted over time. It enables organizations to track their progress, identify trends, and evaluate the effectiveness actions taken. It also serves as reference for future audits and allows auditors to understand the current cybersecurity implemented and facilitates a more targeted approach to future cybersecurity updates and audits.

- **Decision-making Support**: Valuable insights and information that can support decision-making processes. It allows management to make informed decisions about allocating resources, prioritizing cybersecurity investments, and addressing identified risks and vulnerabilities.

Table of Contents

Inventory

EQUIPMENT	OPERATIING SYSTEM	ADDITIONAL INFO	IP ADDRESS
Router/Custom Network	-	-	10.10.229.1
Docker	CentOS 7	Ghost Container	10.10.229.11
NginX Reverse Proxy	Rocky 8	Reverse Proxy	10.10.229.10
WordPress	Ubuntu	LAMP Stack running WordPress	10.10.229.12

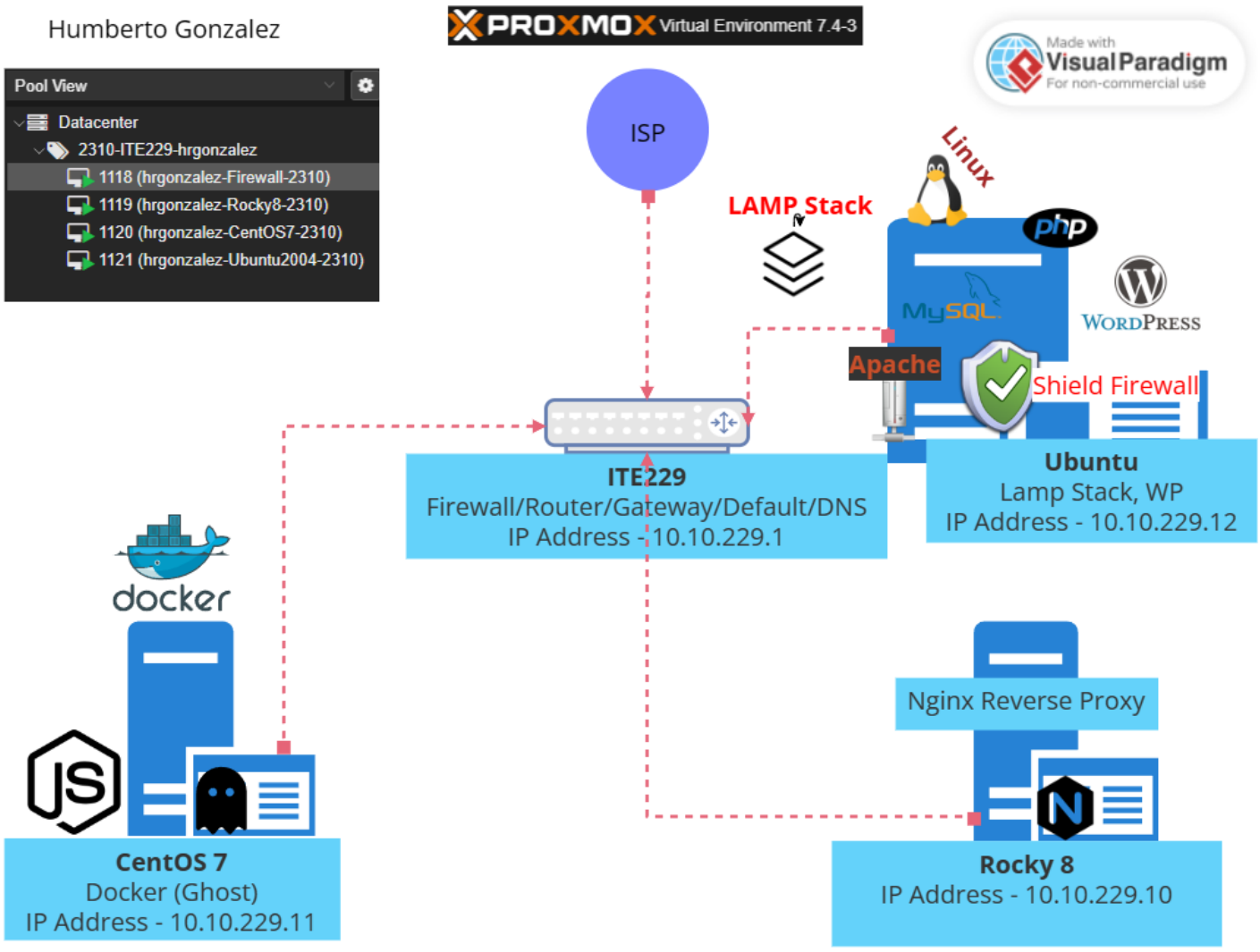
Custom Network

NETWORK NAME	SUBNET IP	SUBNET MASK	DNS	GATEWAY
ITE229	10.10.229.0	255.255.255.0	10.10.229.1	10.10.229.1

IDs and Passwords

ACCOUNT	USER ID	PASSWORD
CentOS 7 Root User	root	Fullsail1!
Rocky 8 Root User	root	Fullsail1!
MySQL Root User	root@localhost	[ugLm2vSIO]
MySQL WordPress User	WordPressUser	[NAc54KVa]
WordPress Admin	admin	[Naruto_Pain$&]
WordPress Admin 2FA	psycho	[Fullsail1!]

Marconi Law Network Topology Diagram



Summary for Topology

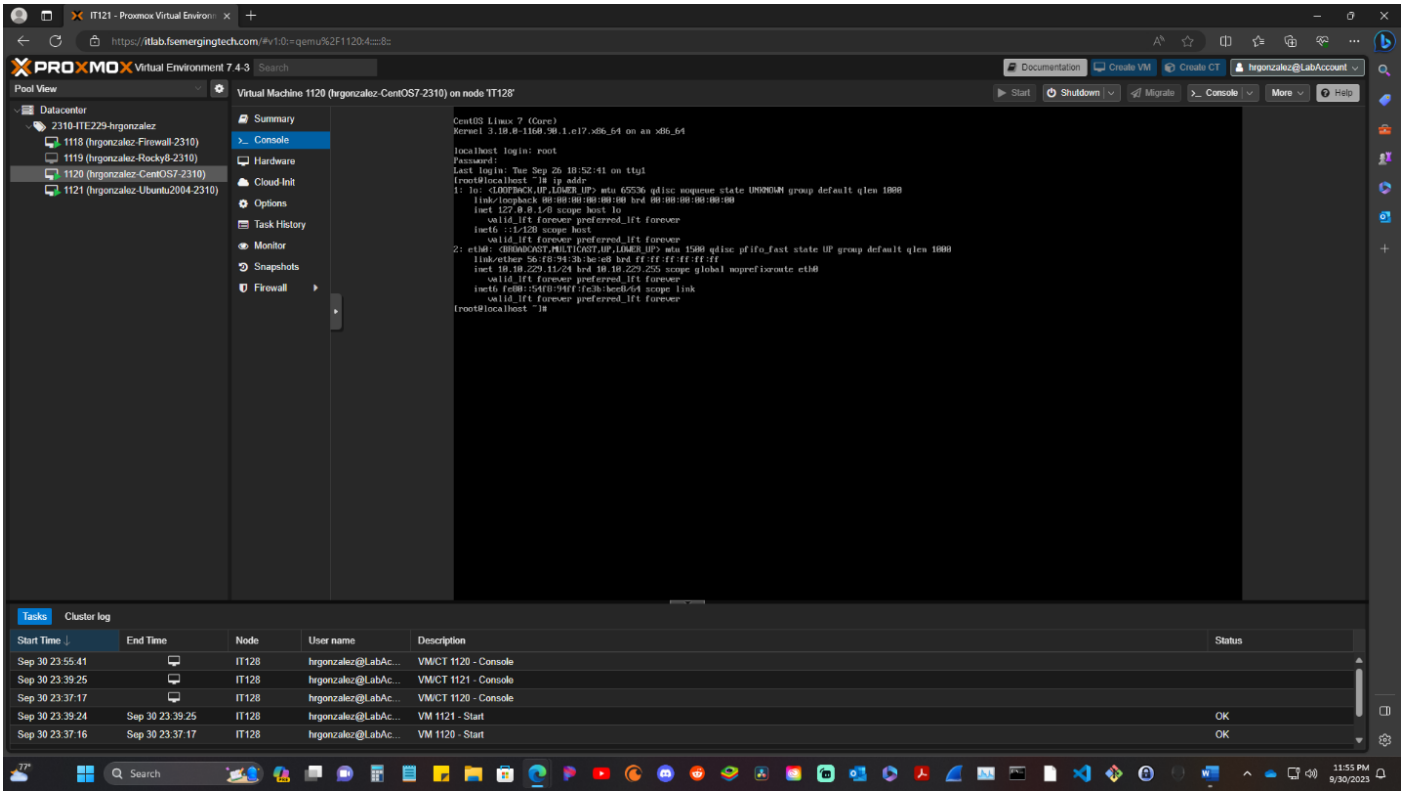
In summary, Nginx serves as a reverse proxy that directs incoming web traffic to the appropriate container (in this case, the Ghost container) based on the requested domain path. The Ghost container evaluates the request, collects data from a database if necessary, and prepares the HTML response, which is then transmitted back to the user's browser for rendering via Nginx. This configuration offers web application hosting flexibility, scalability, and security.

Adding LAMP Stack to the topology changes things a bit from previous setup. Nginx will operates as a central traffic director like usual, selecting which back end (WordPress or Ghost) should handle incoming requests based on the URL or domain. Docker is utilized to isolate and manage the Ghost containers, while the LAMP stack (Apache, PHP, and MySQL) handles WordPress site requests. This infastructure enables different websites and services to be hosted in the same environment, while maintaining it's flexibility, scalability, and security.

Node.js Application (Ghost) on Docker

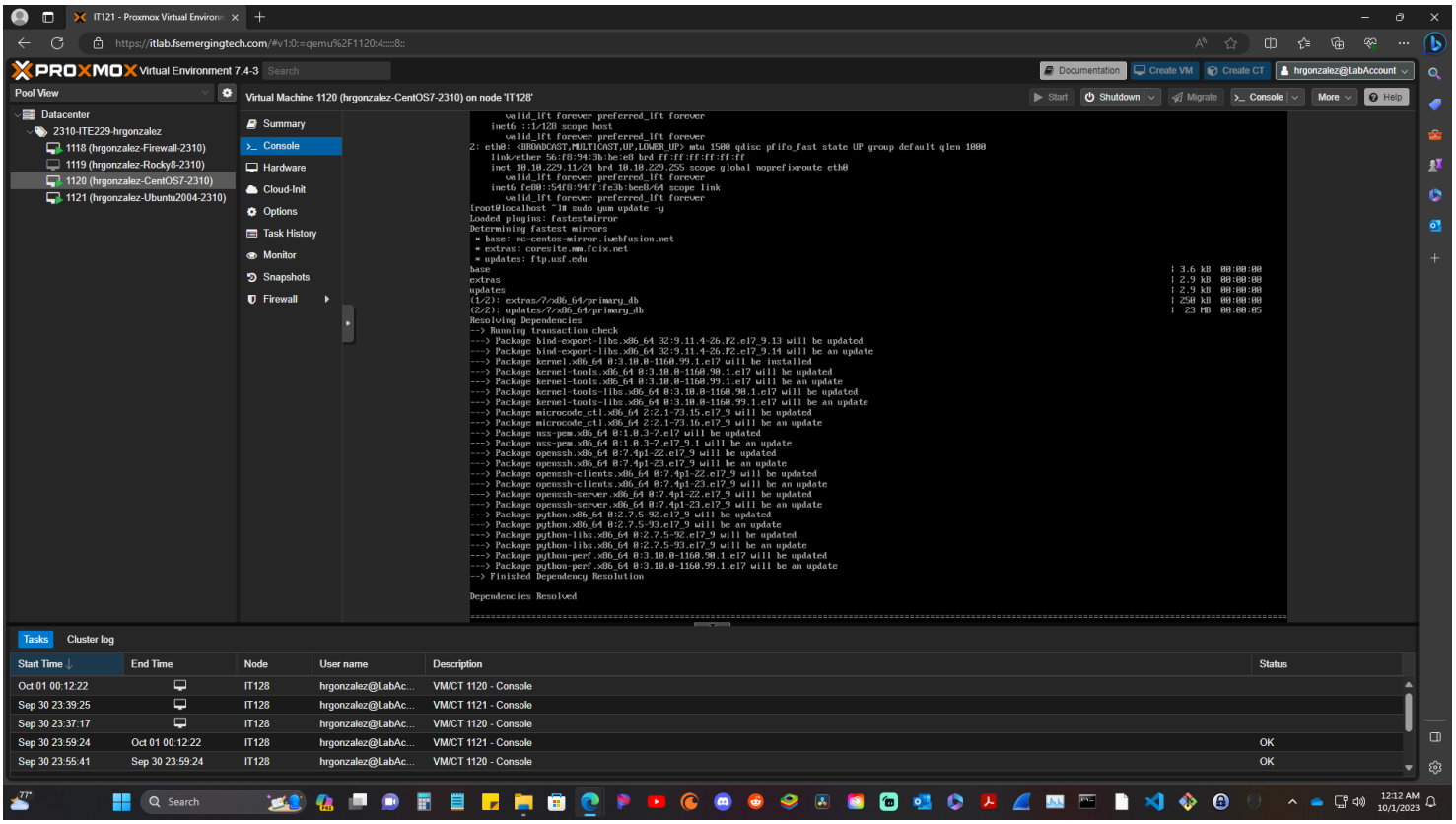
Show screenshot of your CentOS 7 Console in VE

Open CentOS 7 and log in as root w/password Fullsail1! In the terminal window verify IP Address by entering command prompt ip addr and make sure the ip address is 10.10.229.11/24



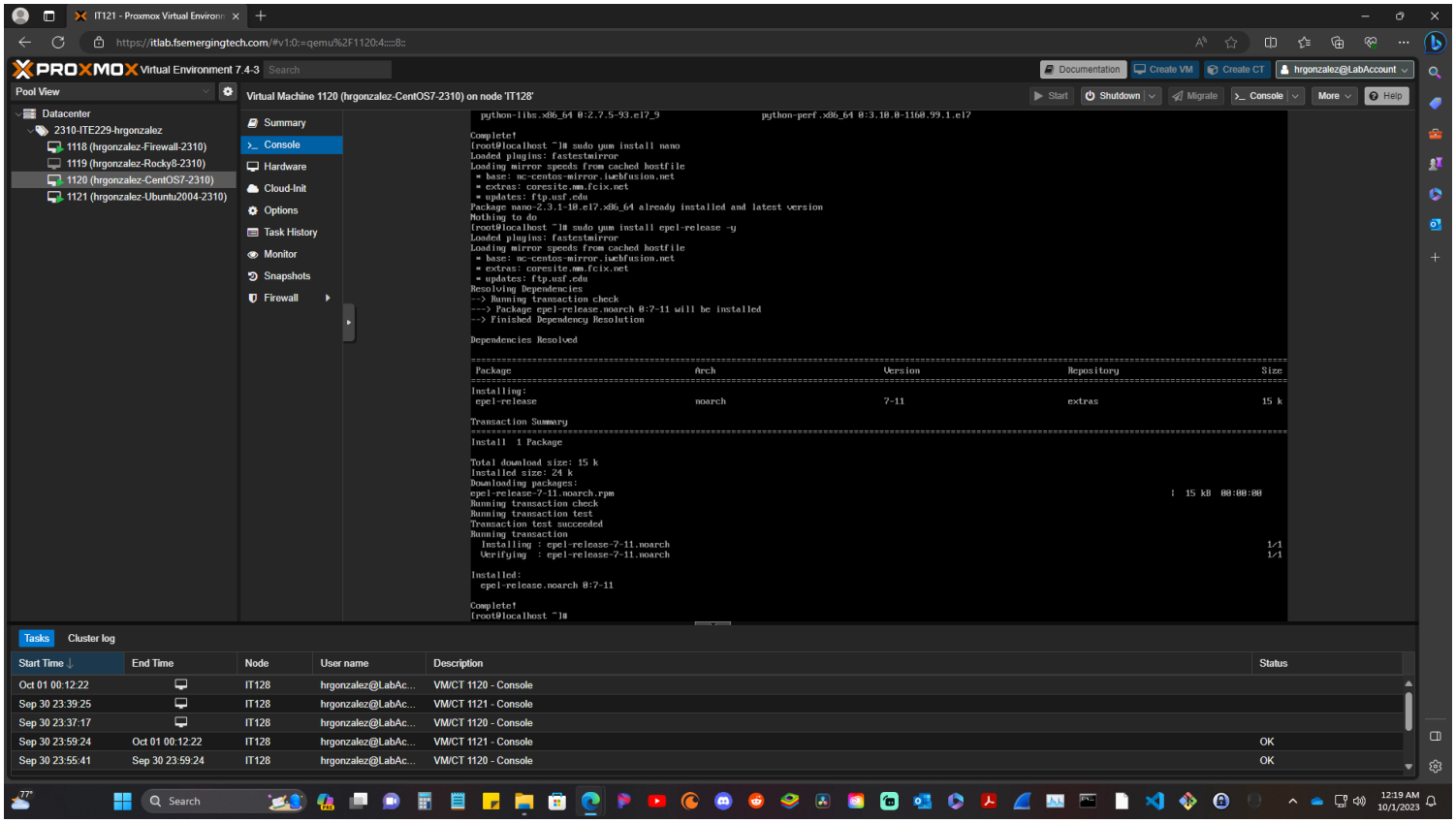
Update CentOS

Next make sure you are using all the current updates and files by entering the command prompt sudo yum update -y



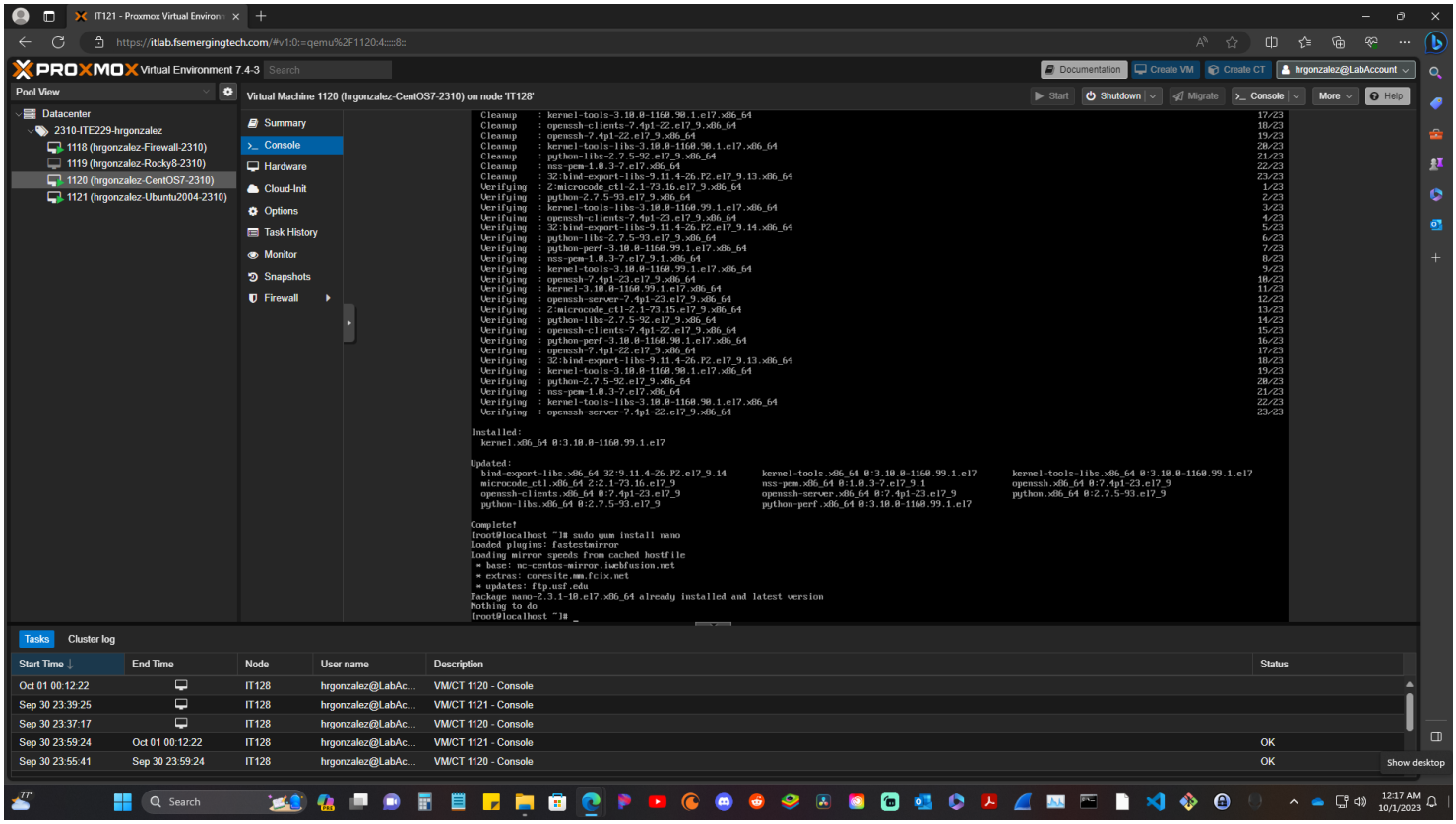
Install EPEL Packages

Install all the extra packages for enterprise Linux (EPEL) by inputting the command prompt sudo yum install epel-release -y



Install Nano Editor

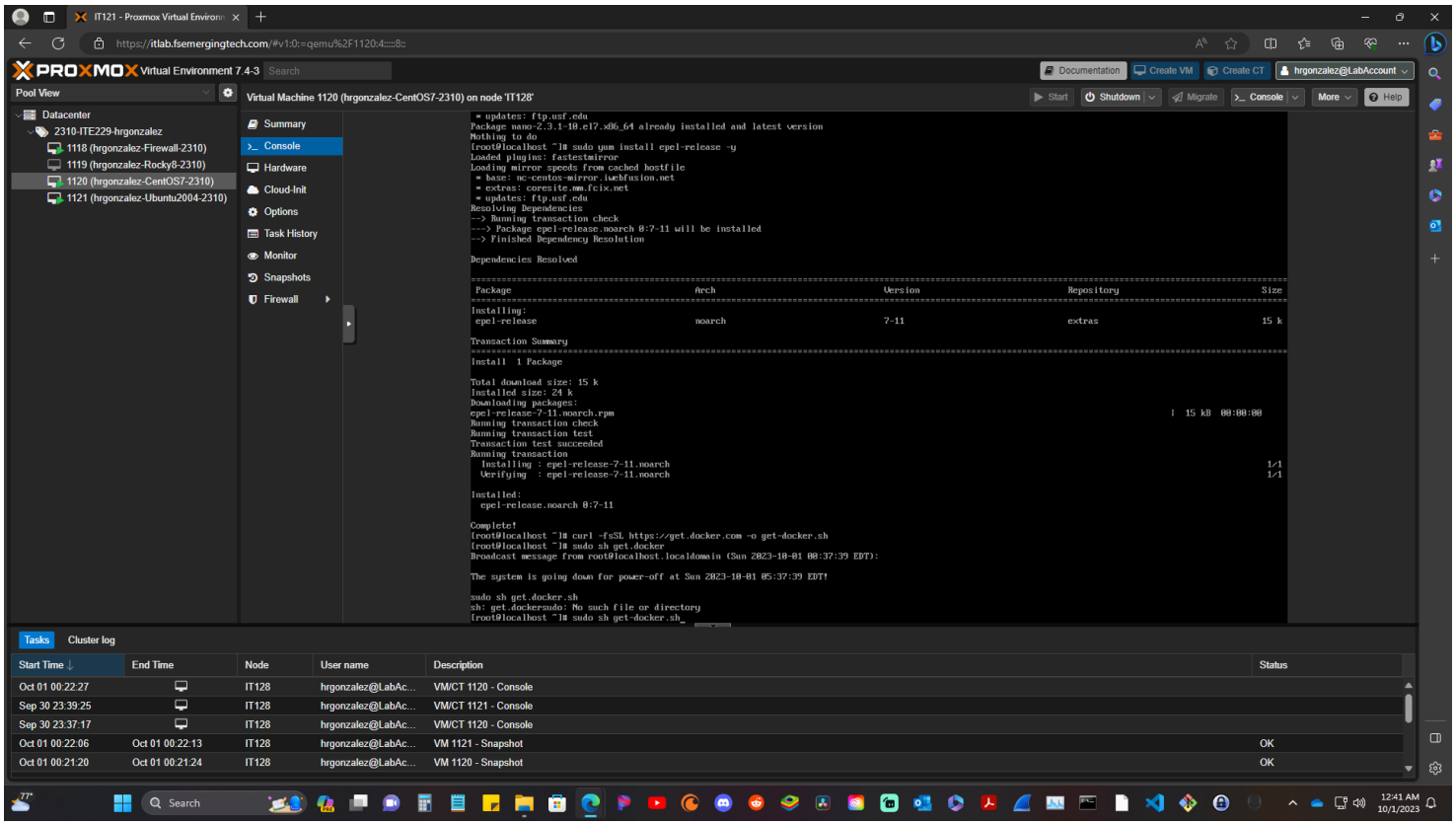
Next, we need to install nano editor. In the terminal window input command prompt sudo yum install nano



Docker CE

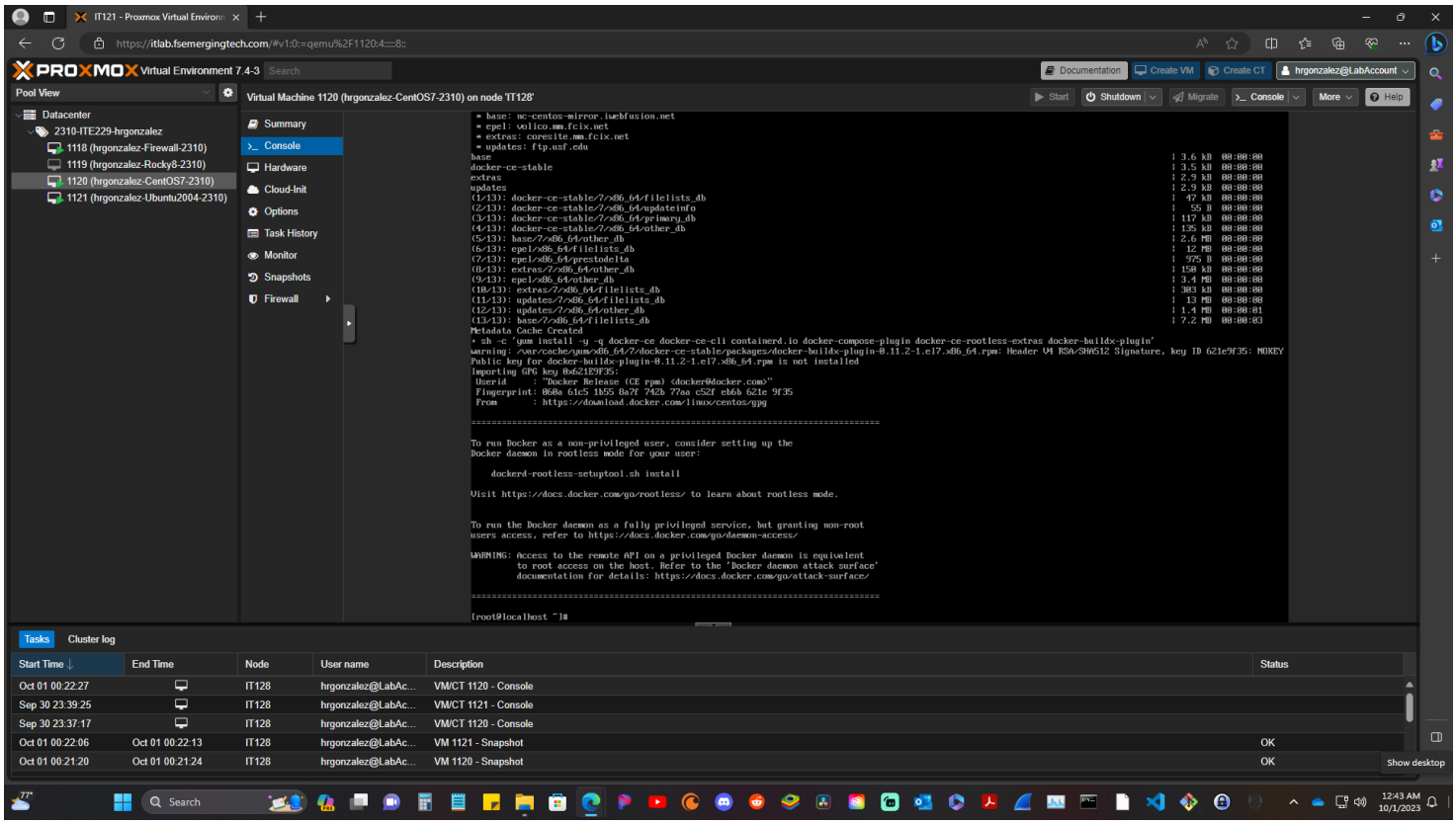
Install required packages

Now that we have all the prequesites is time to install docker and it's packages. In the terminal window input command prompt curl fsSL https://get.docker.com -o get-docker.sh exactly as typed. This will grab the file from is corresponding url without having to externally get it from the webpage itself. If prompted correctly nothing should happen besides returnig you back to your root name.



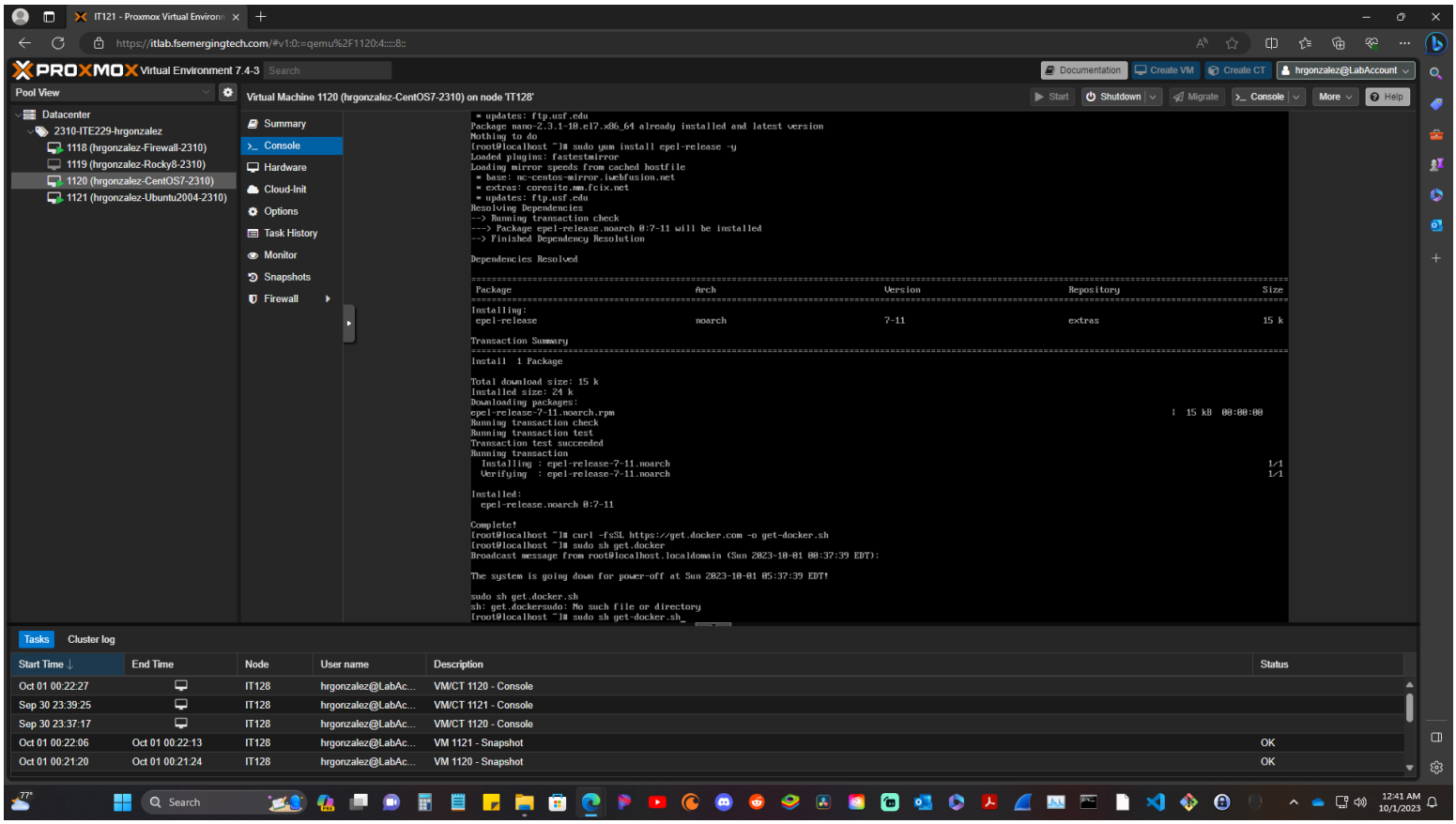
Set up stable repository

By installing Docker CE following the command prompt below it should download all stable repositories automatically.



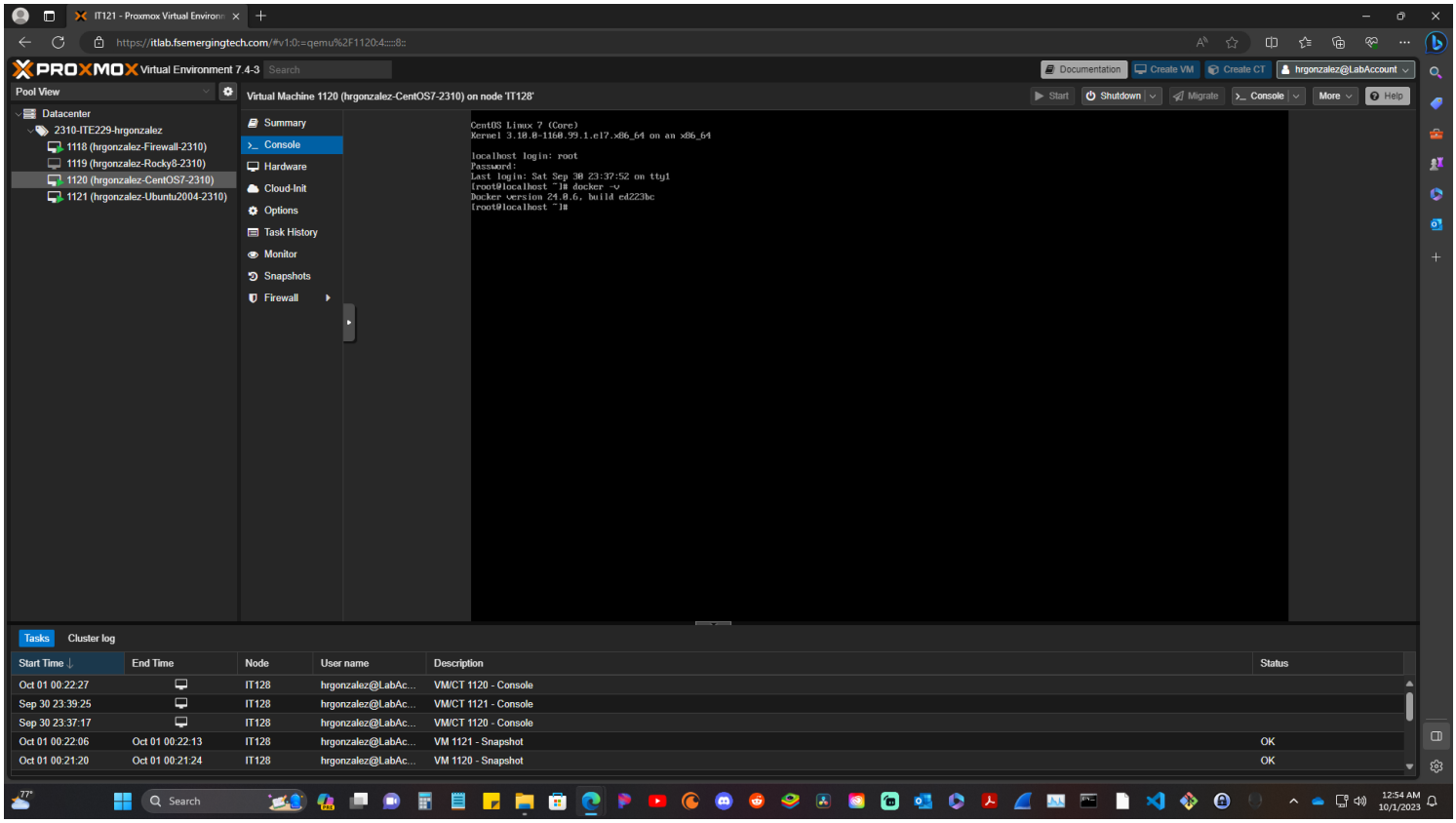
Install Docker CE

To install Docker CE input prompt sudo sh get-docker.sh



Verify docker version

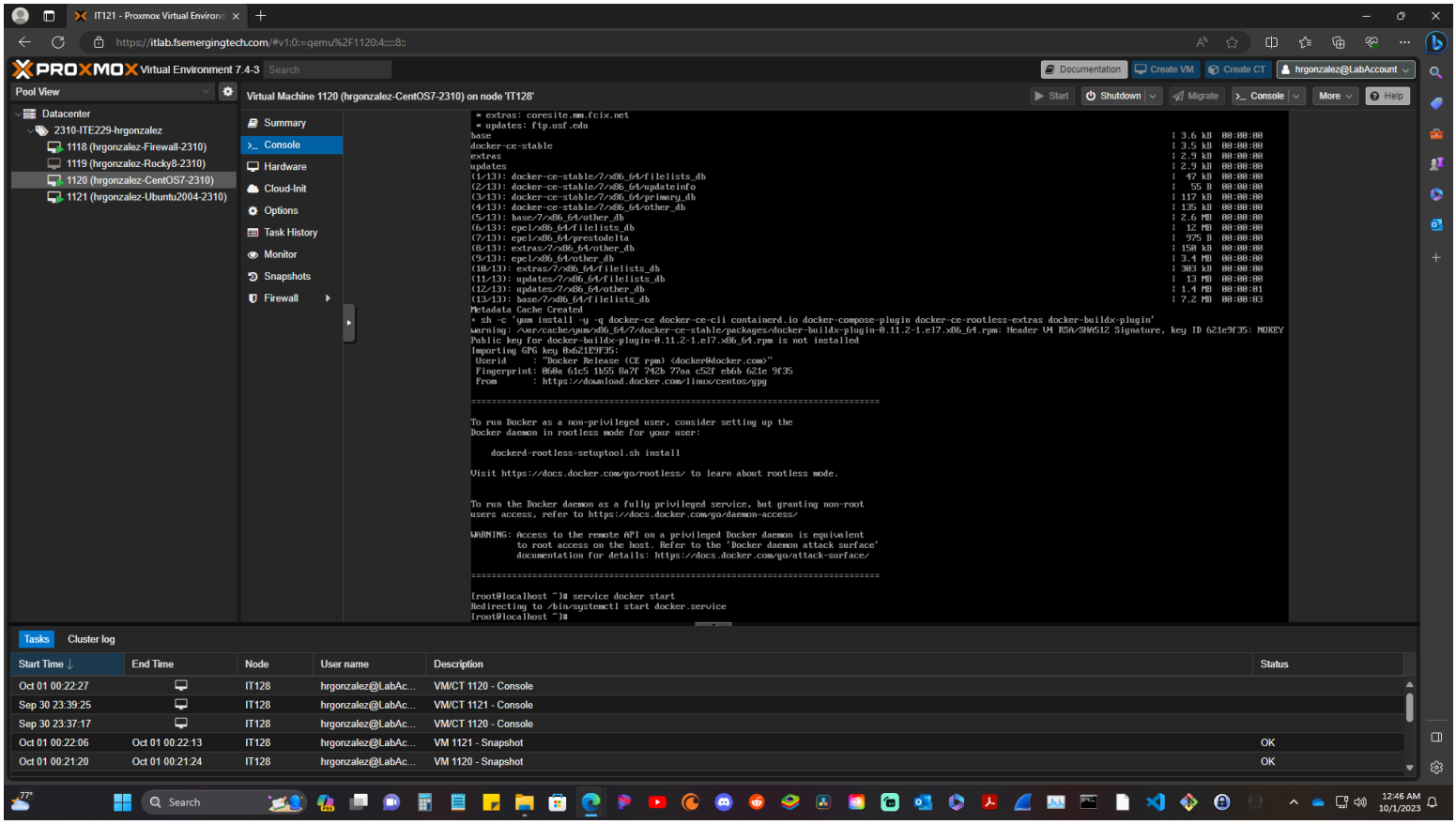
Next we have to verify Docker version by inputting command prompt docker -v



Initialize Docker

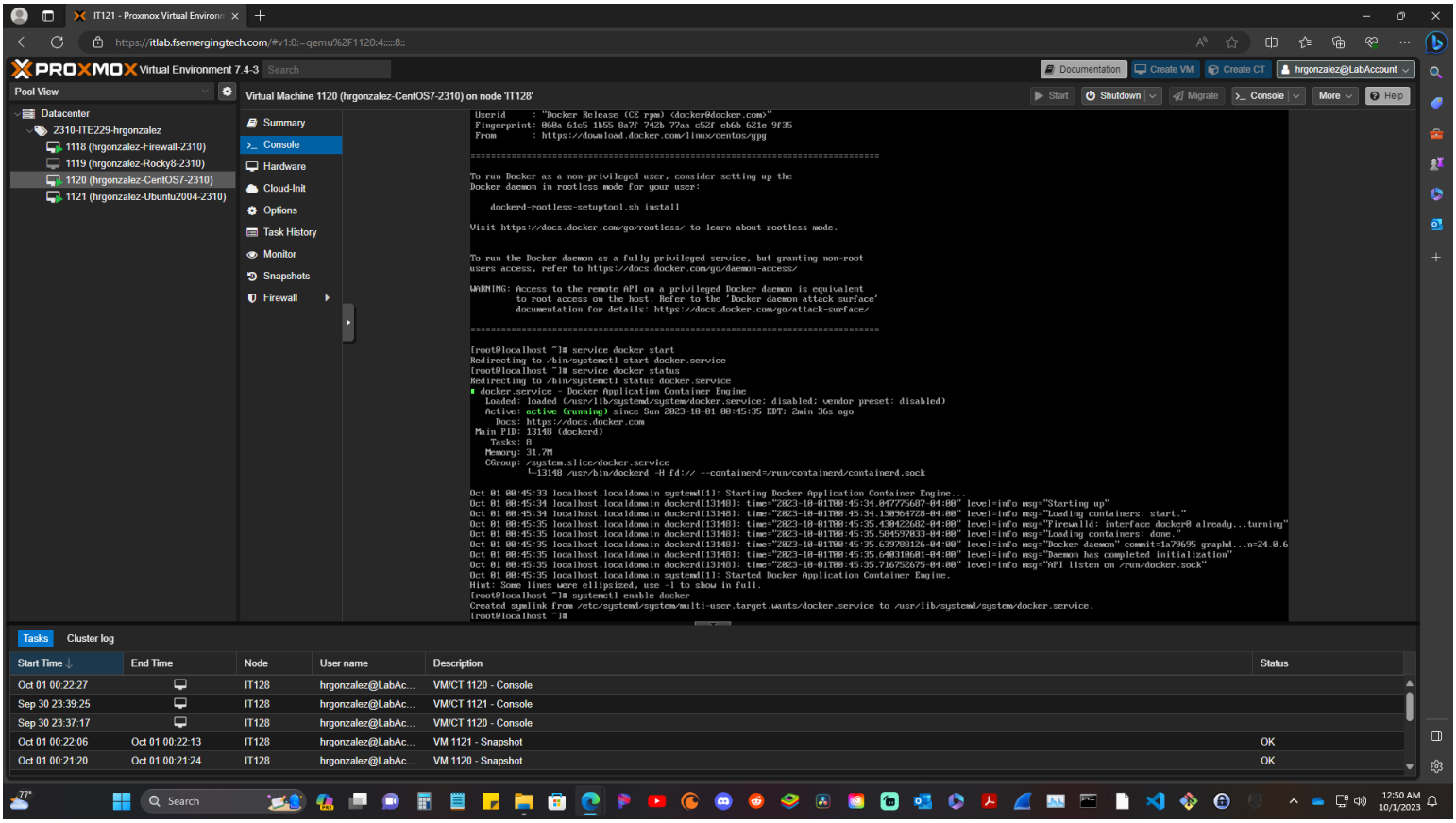
Start Docker

Our next step is to Start Docker service, by using command prompt service docker start if prompted correctly it should redirect you to bin/system like in the picture below.



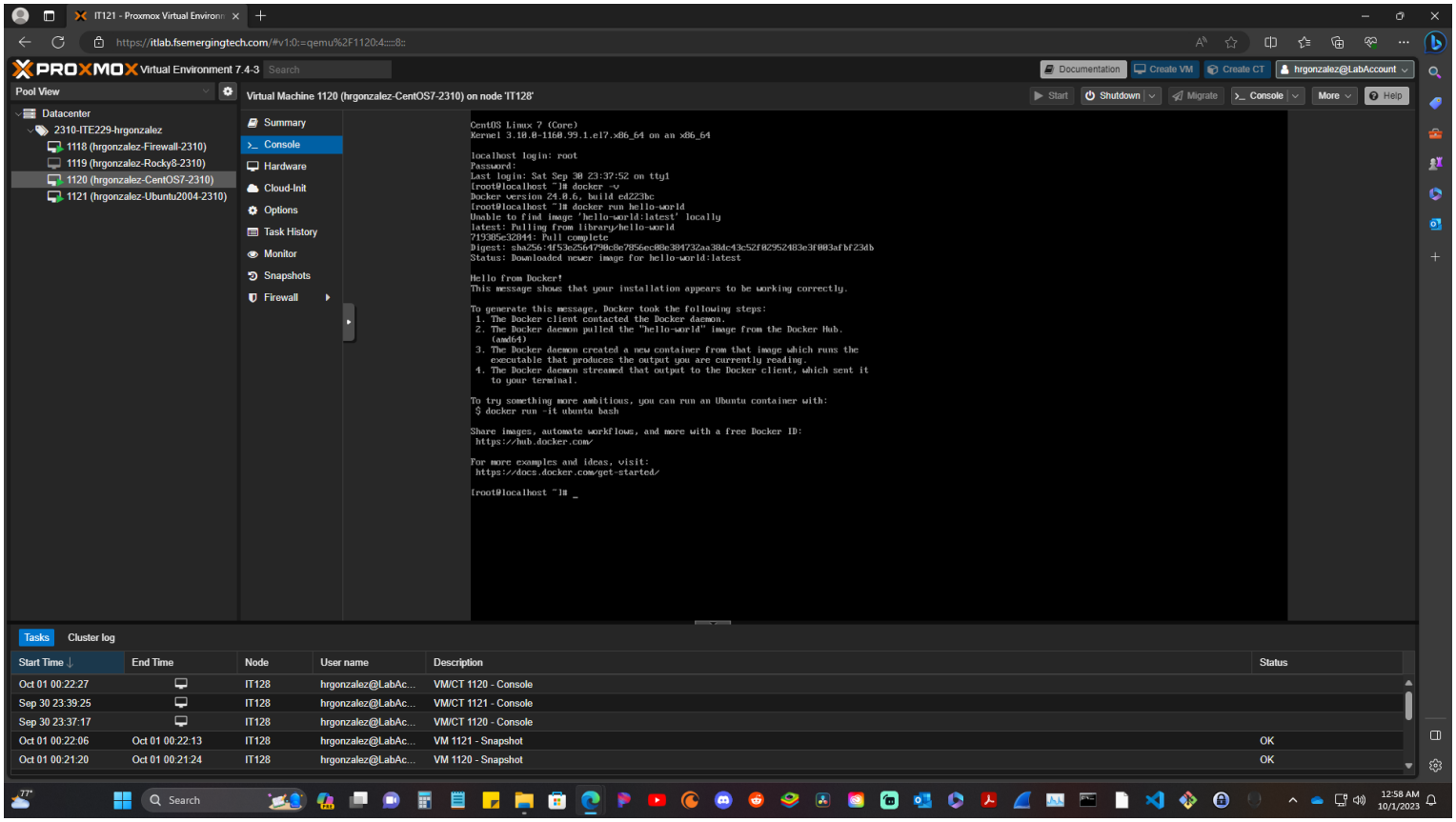
Enable Docker

Continuing we have to enable Docker by using command prompt systemctl enable docker if prompted correctly it should follow by a line which say created symlink like in the photo below.



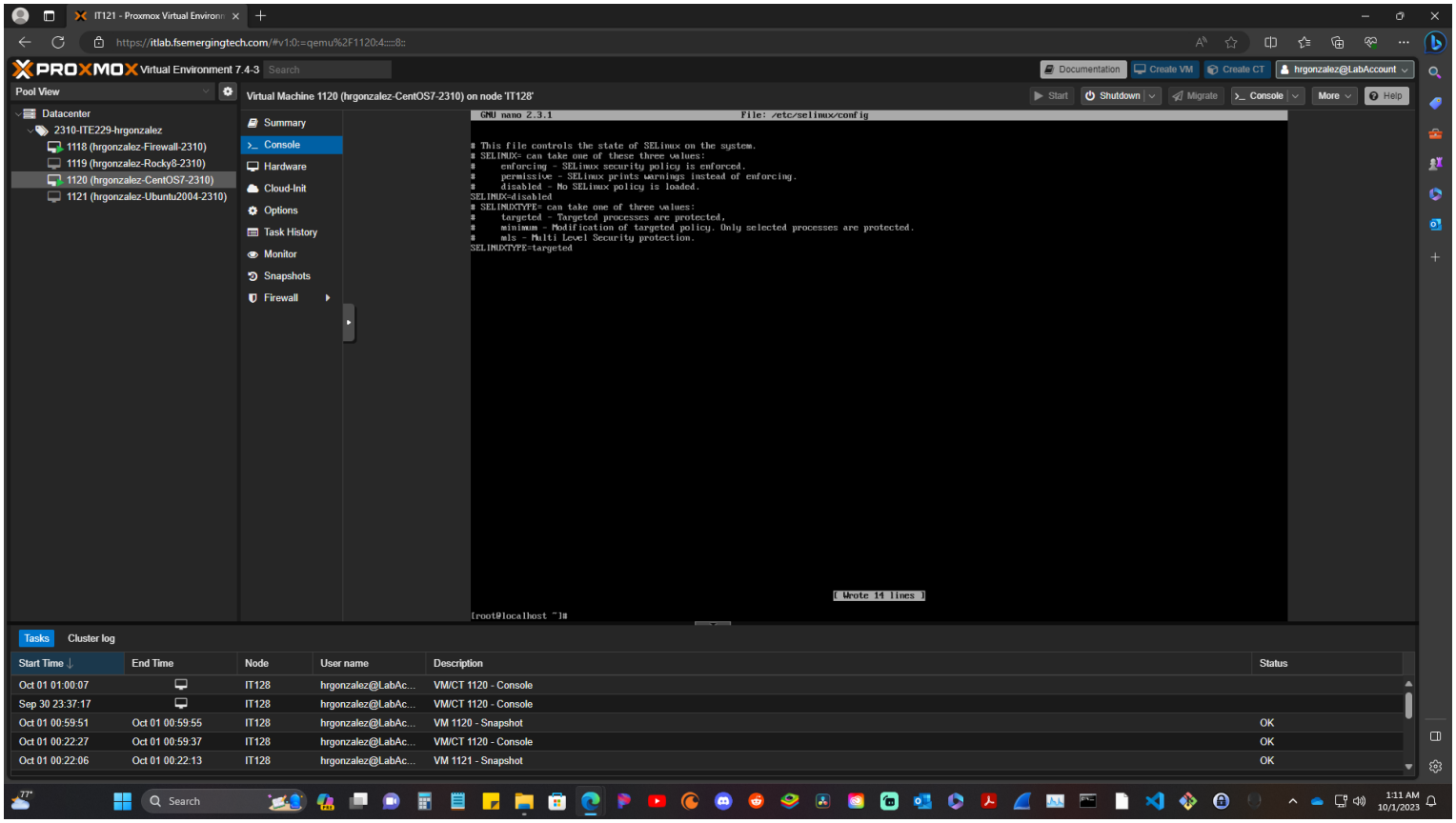
Test Docker (hello-world)

To make sure Docker is installed properly we need to test it, input command prompt docker run hello-world
Docker should respond back with “hello fromm docker!” as pictured below.



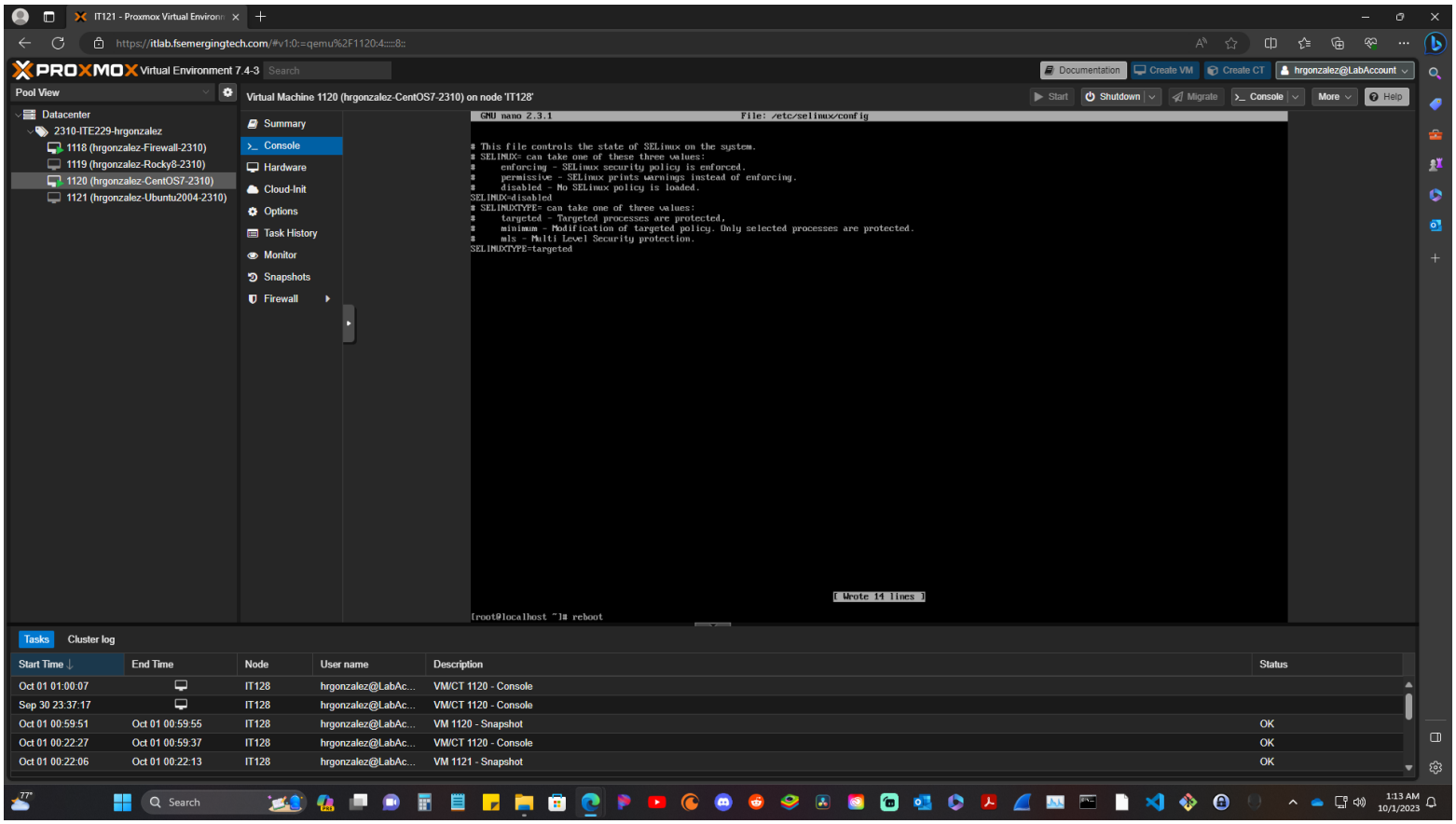
Disable SELinux

Now we need to disable SELinux in order for us to be able to download or get other files that currently we cannot get due to SELinux being enabled. In the terminal window input command prompt sudo nano /etc/selinux/config this will take you to the SELinux window where you need to changed enforcing to disabled by hitting the down arrow all the way to the SELINUX-enforcing and deleting enforcing and typing disabled. After hit control key and x to exit. Save the changes when prompted, then hit enter on the following prompt. You will be redirected to root name right after.



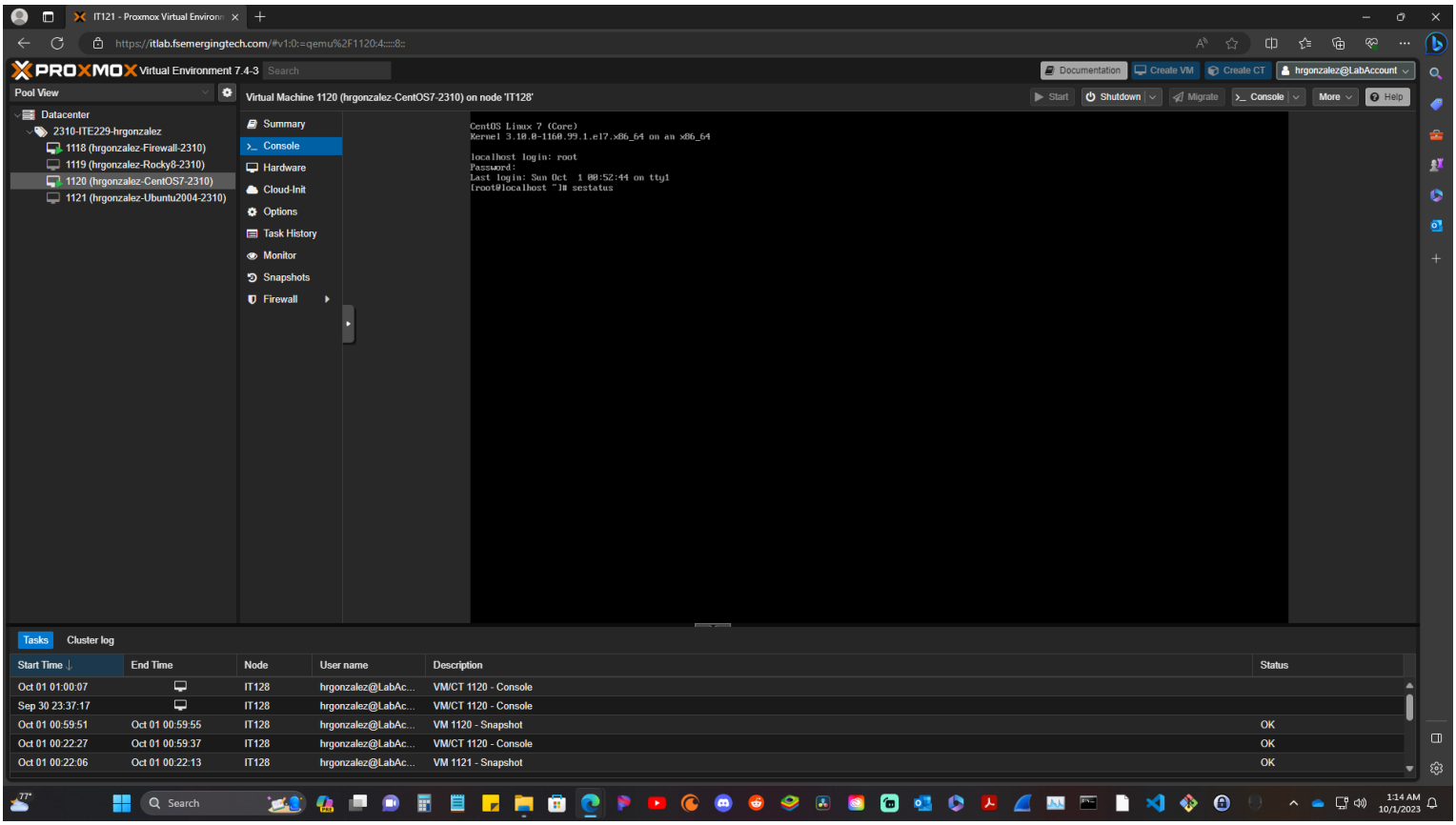
Reboot VM

Once back on your root username. Enter command prompt reboot to restart CentOS 7.



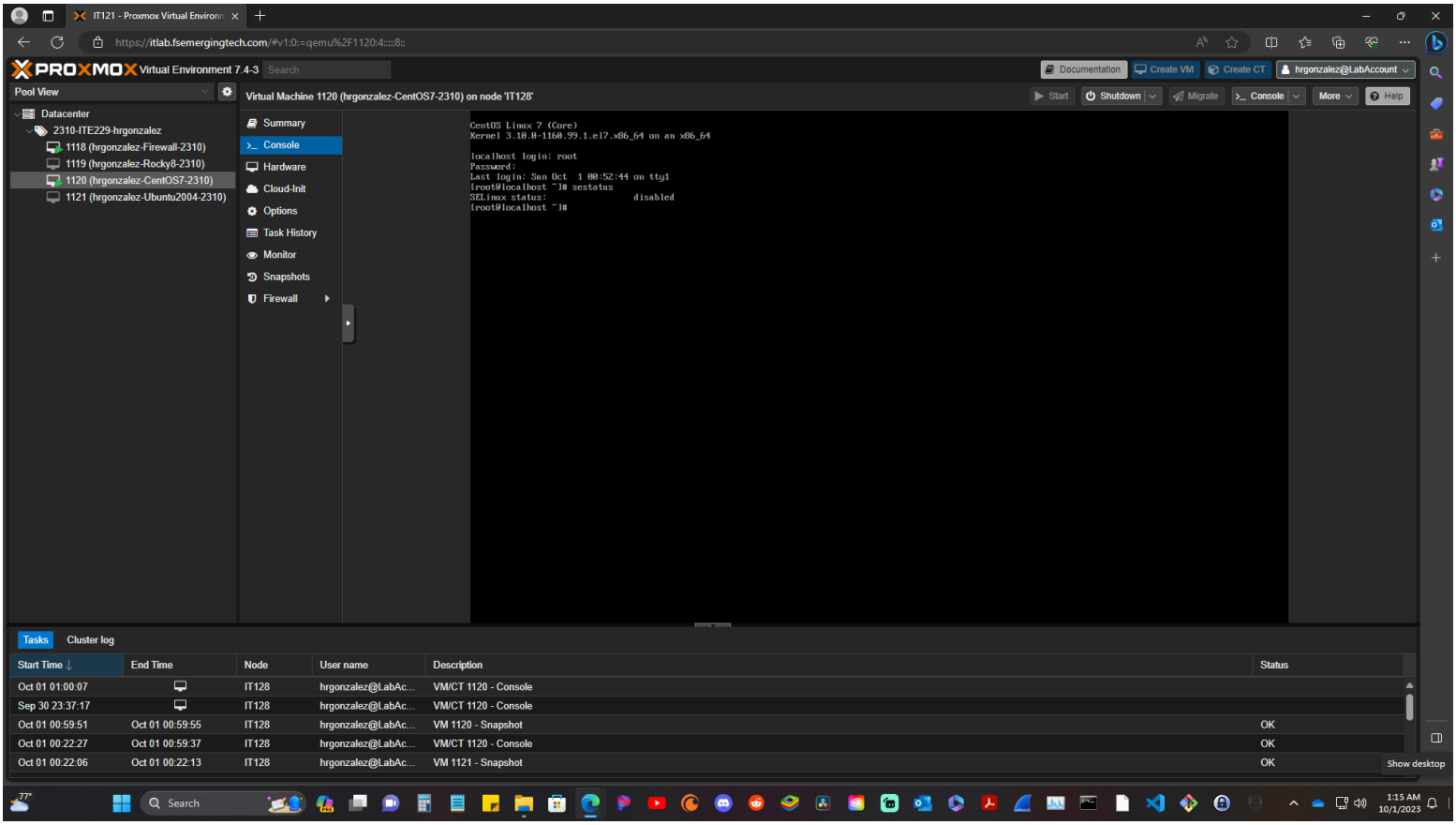
Test SELinux

Once back on the terminal enter command prompt sestatus. *This step and the one that follows are done at the same time, by inputting the sestatus command you are testing and at the same time seeing it's status.



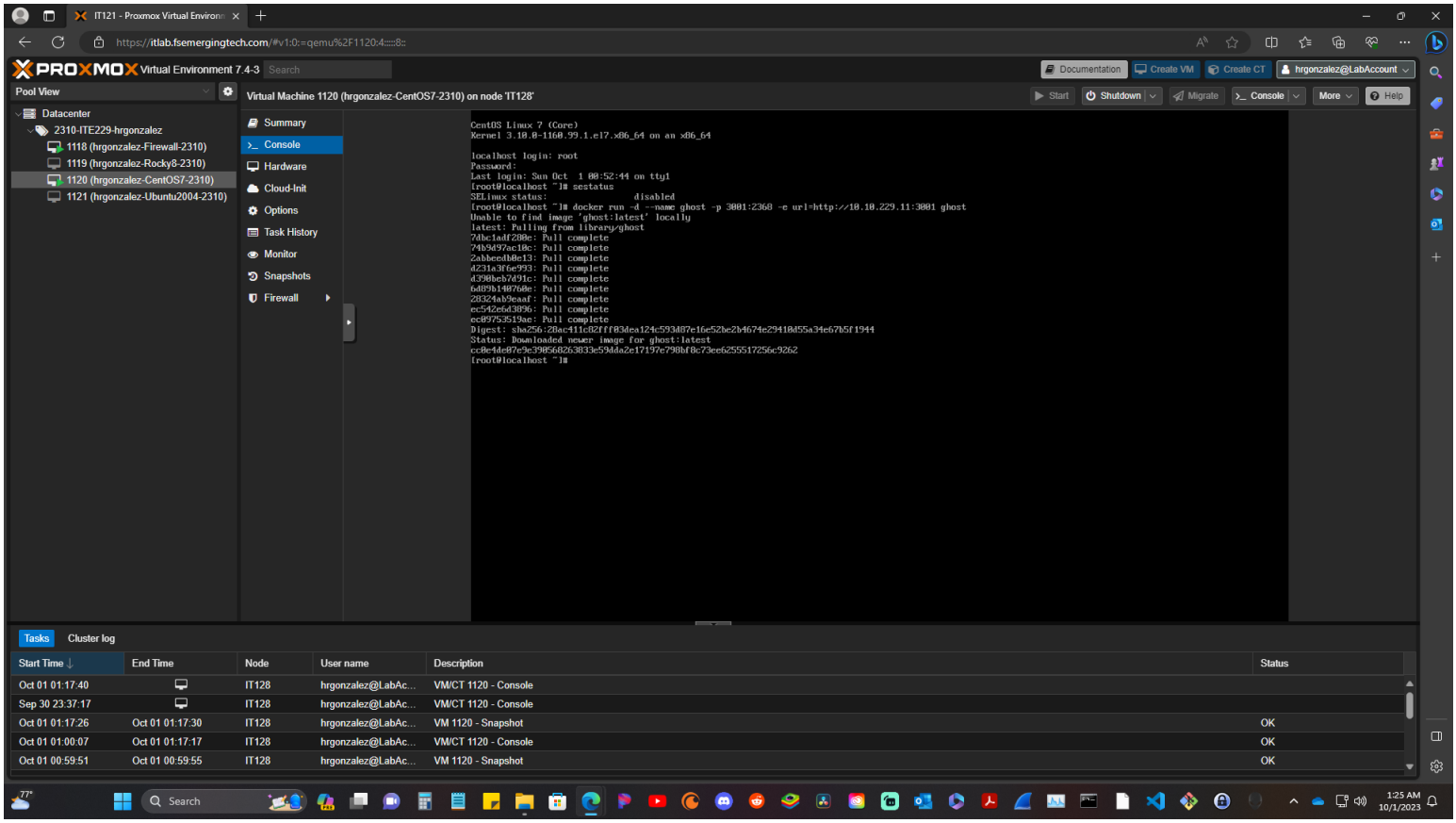
Confirm SELinux Status

Once rebooted log in with root username and your password. To confirm SELinux is disabled enter command prompt sestatus the SELinux status should say disabled.



Install Ghost Docker Container

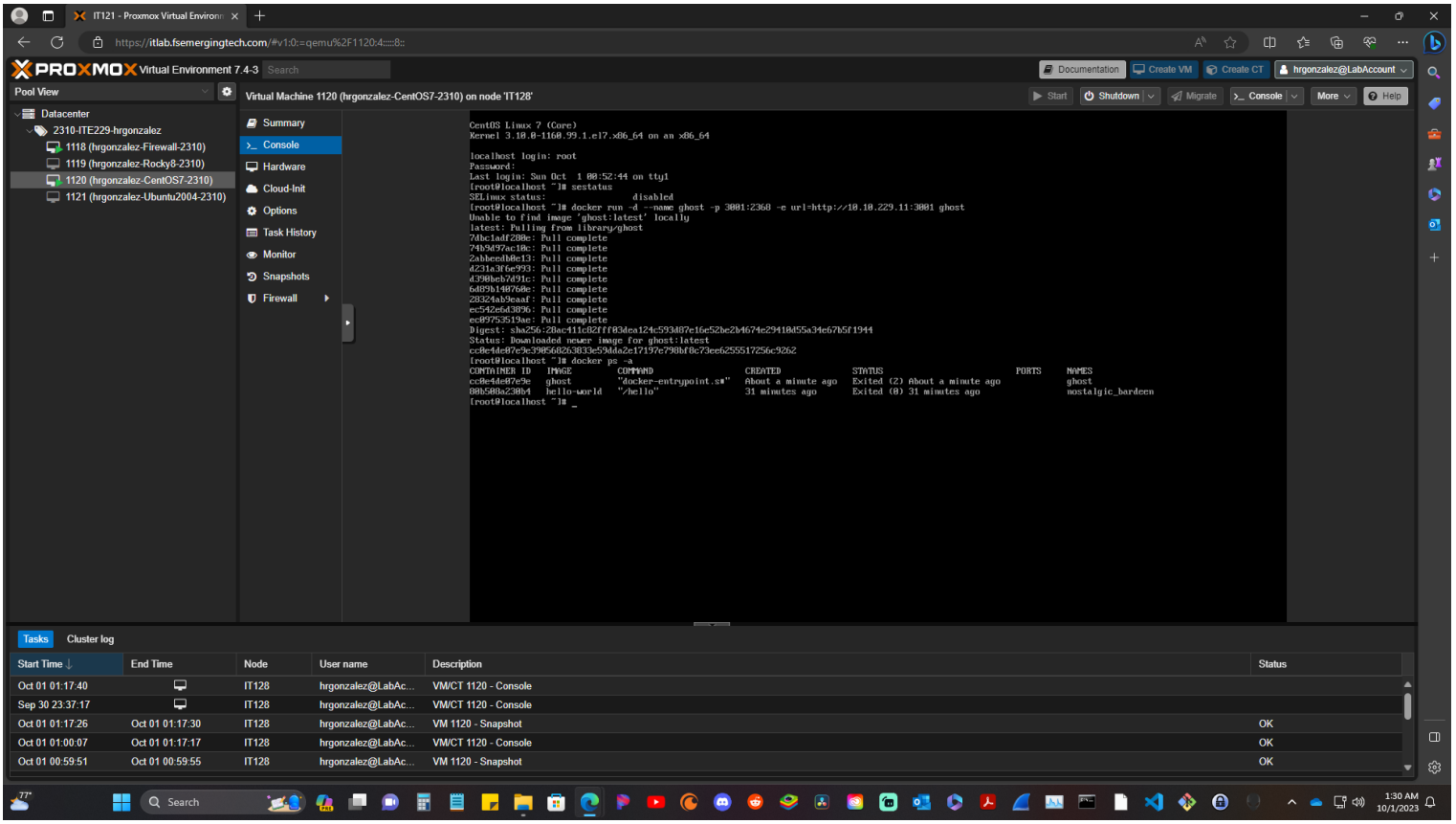
To install Ghost container run command prompt docker run -d –name ghost -p 3001:2368 -e url=http://10.10.229.11:3001 ghost if prompted correctly you will see the samething as the picture below.



Test Ghost

Ghost Container ID

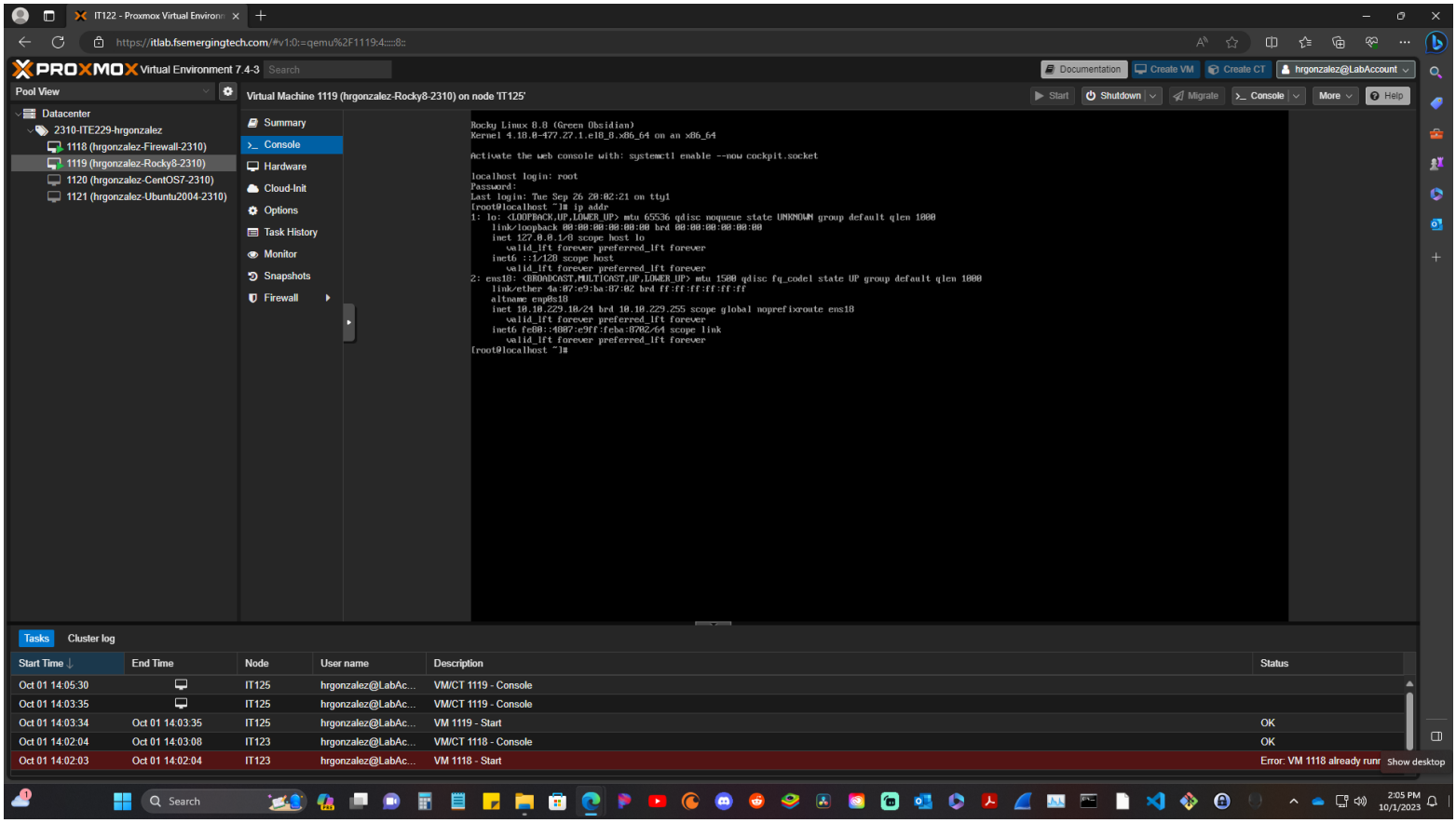
To test Ghost and that it was successfully installed enter command prompt docker ps -a this will show the ghost file and It's content including container ID. *Save the container ID with the file name Ghost, you will need it later.



NginX Reverse Proxy

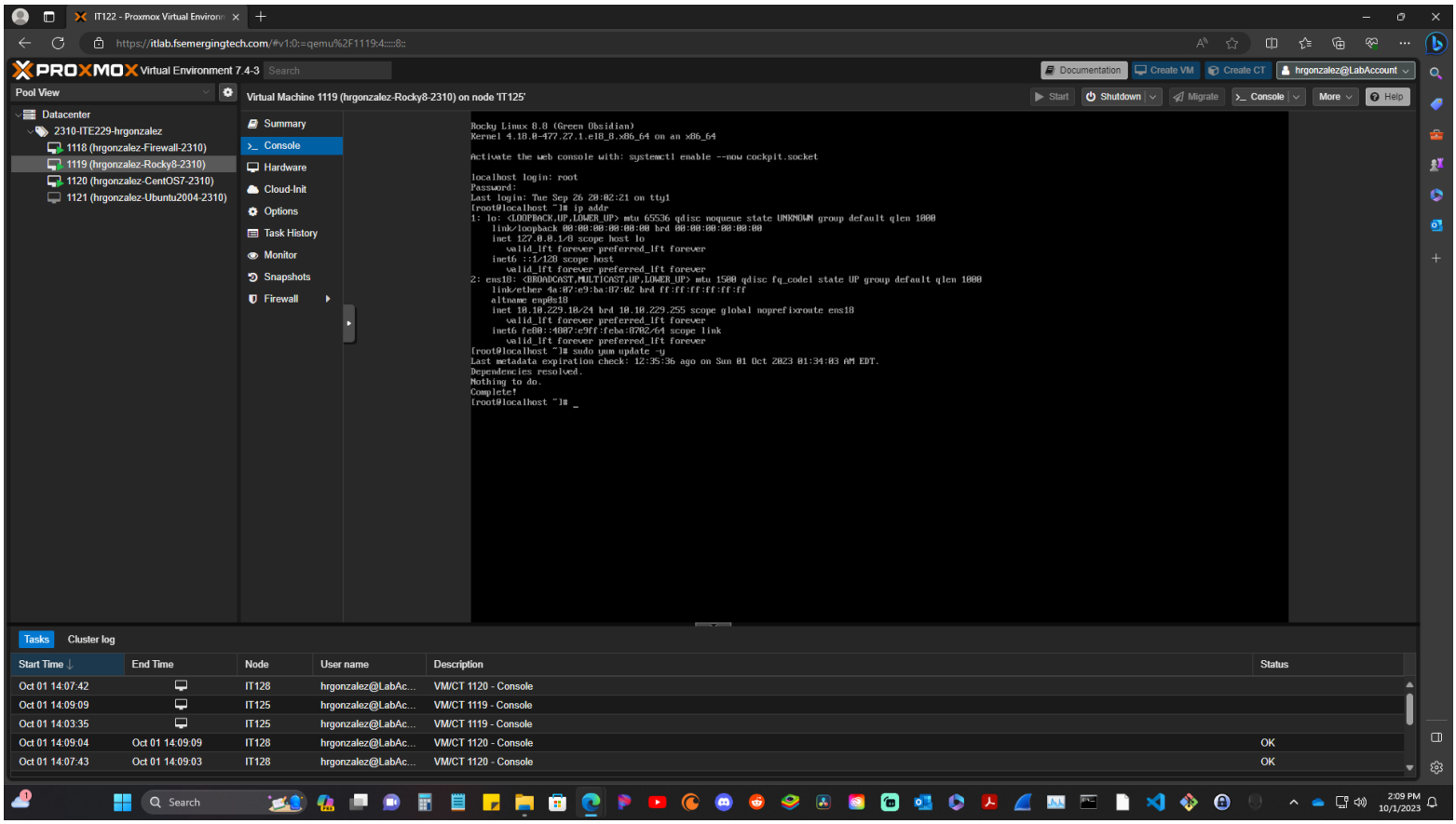
Show screenshot of your Rocky 8 Console in VE

Open Rocky 8 and log in as root w/password Fullsail1!. In the terminal window verify IP Address by entering command prompt ip addr and make sure the ip address is 10.10.229.10/24



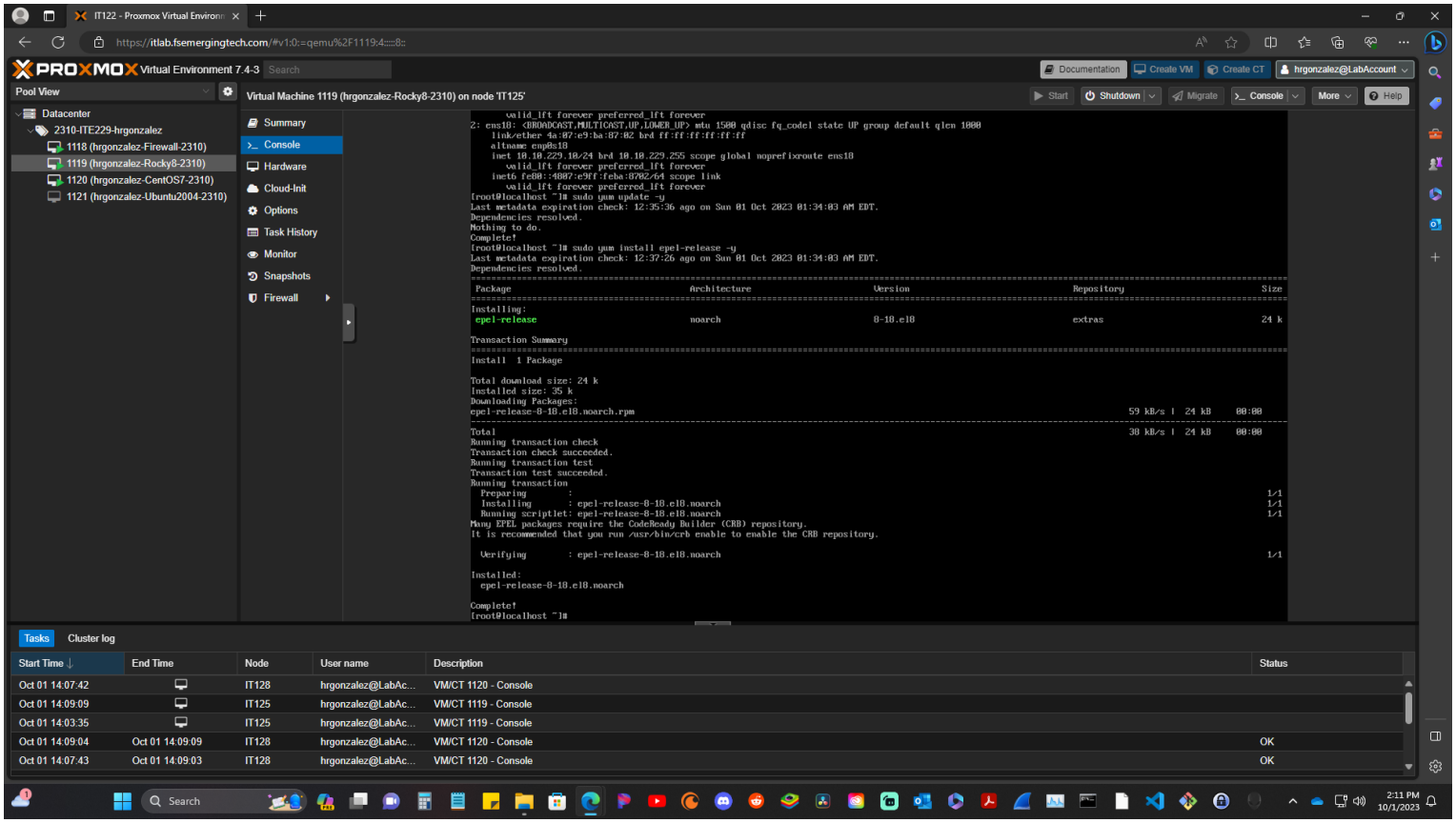
Update Rocky 8

Next make sure you are using all the current updates and files by entering the command prompt sudo yum update -y



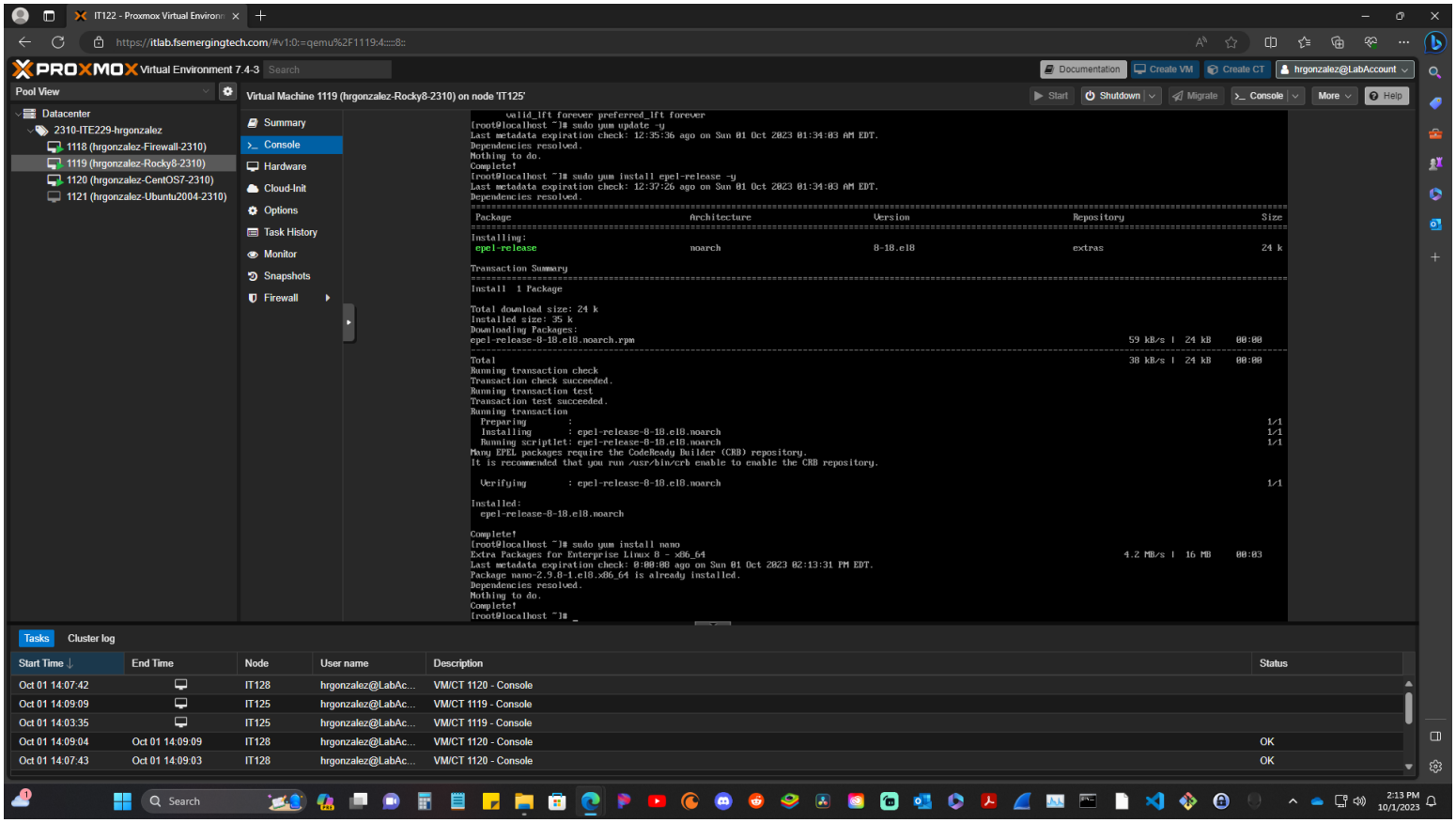
Install EPEL Packages

Install all the extra packages for enterprise Linux (EPEL) by inputting the command prompt sudo yum install epel-release -y



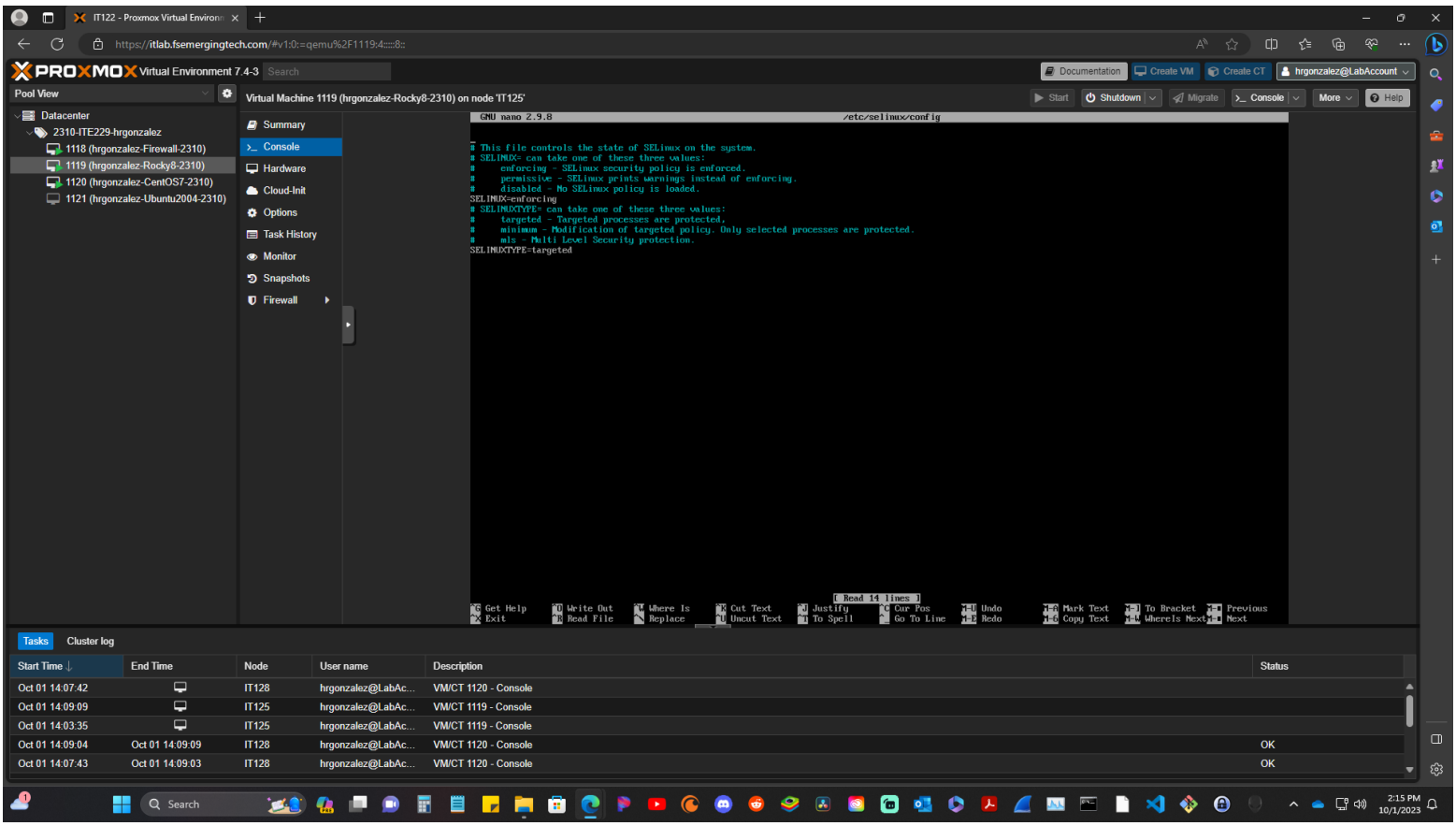
Install Nano Editor

Next, we need to install nano editor. In the terminal window input command prompt sudo yum install nano.



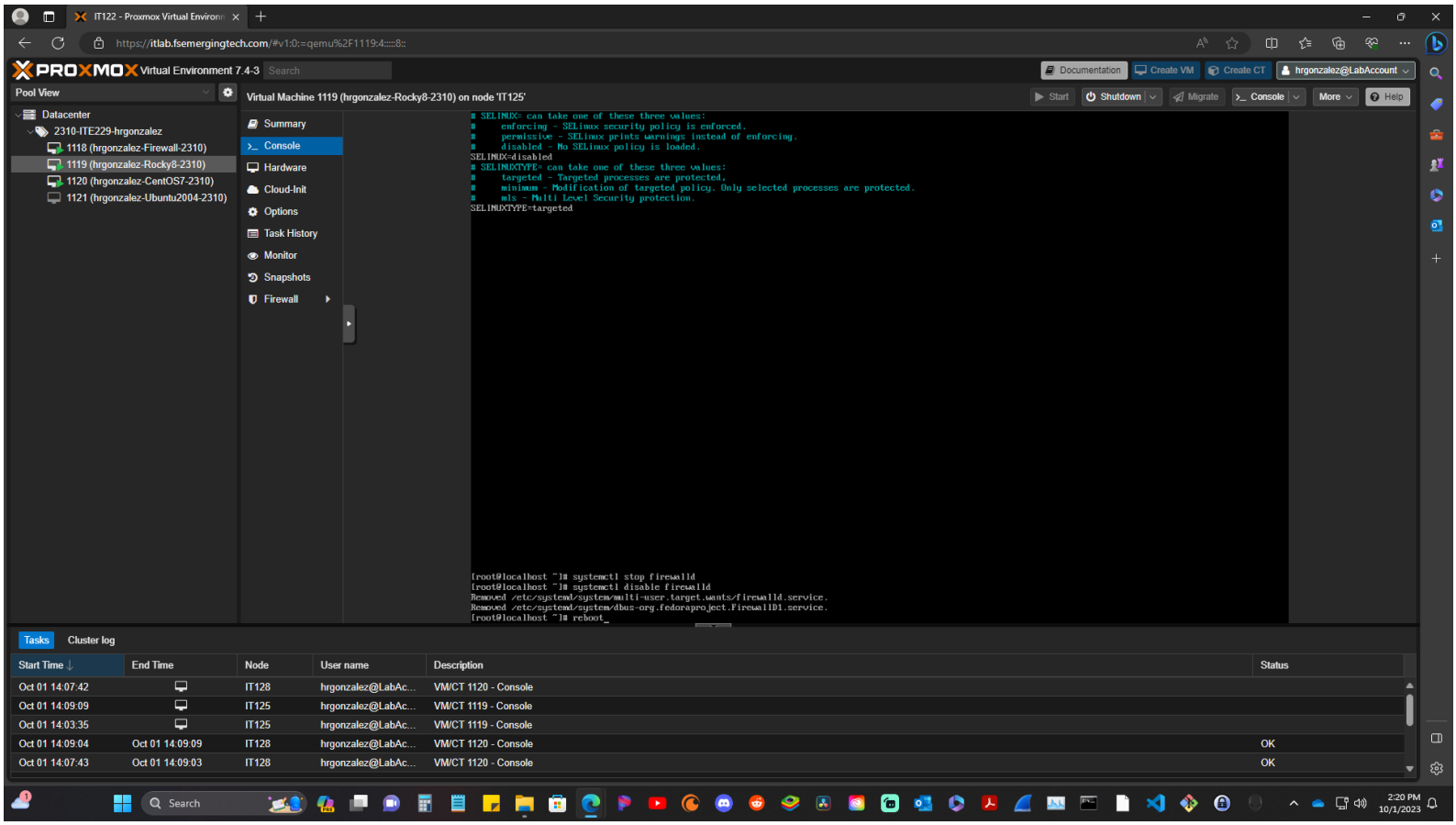
Disable SELinux

Now we need to disable SELinux in order for us to be able to download or get other files that currently we cannot get due to SELinux being enabled. In the terminal window input command prompt sudo nano /etc/selinux/config this will take you to the SELinux window where you need to changed enforcing to disabled by hitting the down arrow all the way to the SELINUX-enforcing and deleting enforcing and typing disabled. After hit control key and x to exit. Save the changes when prompted, then hit enter on the following prompt. You will be redirected to root name right after.



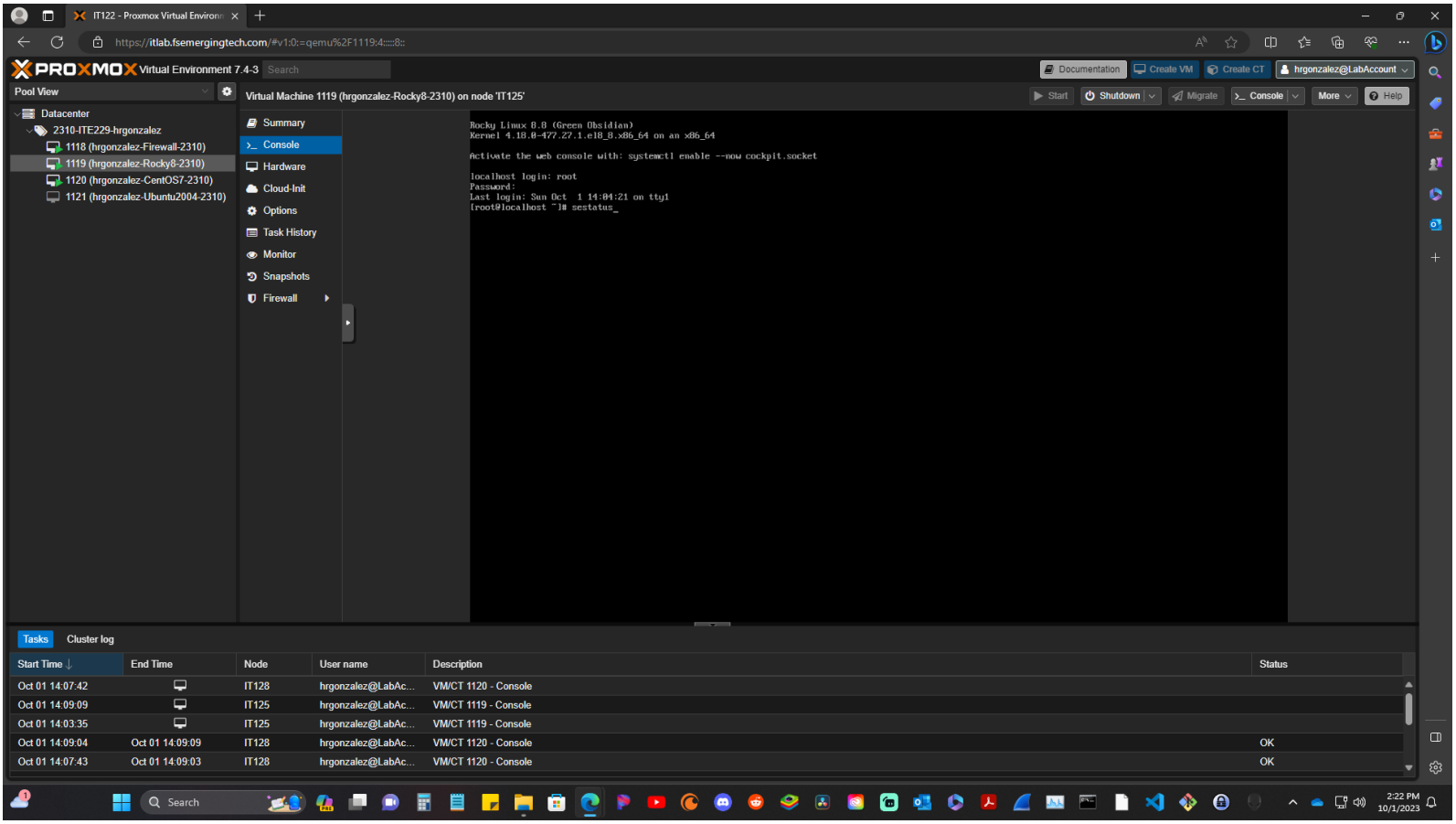
Reboot VM

Once back on your root username. Enter command prompt reboot to restart Rocky 8.



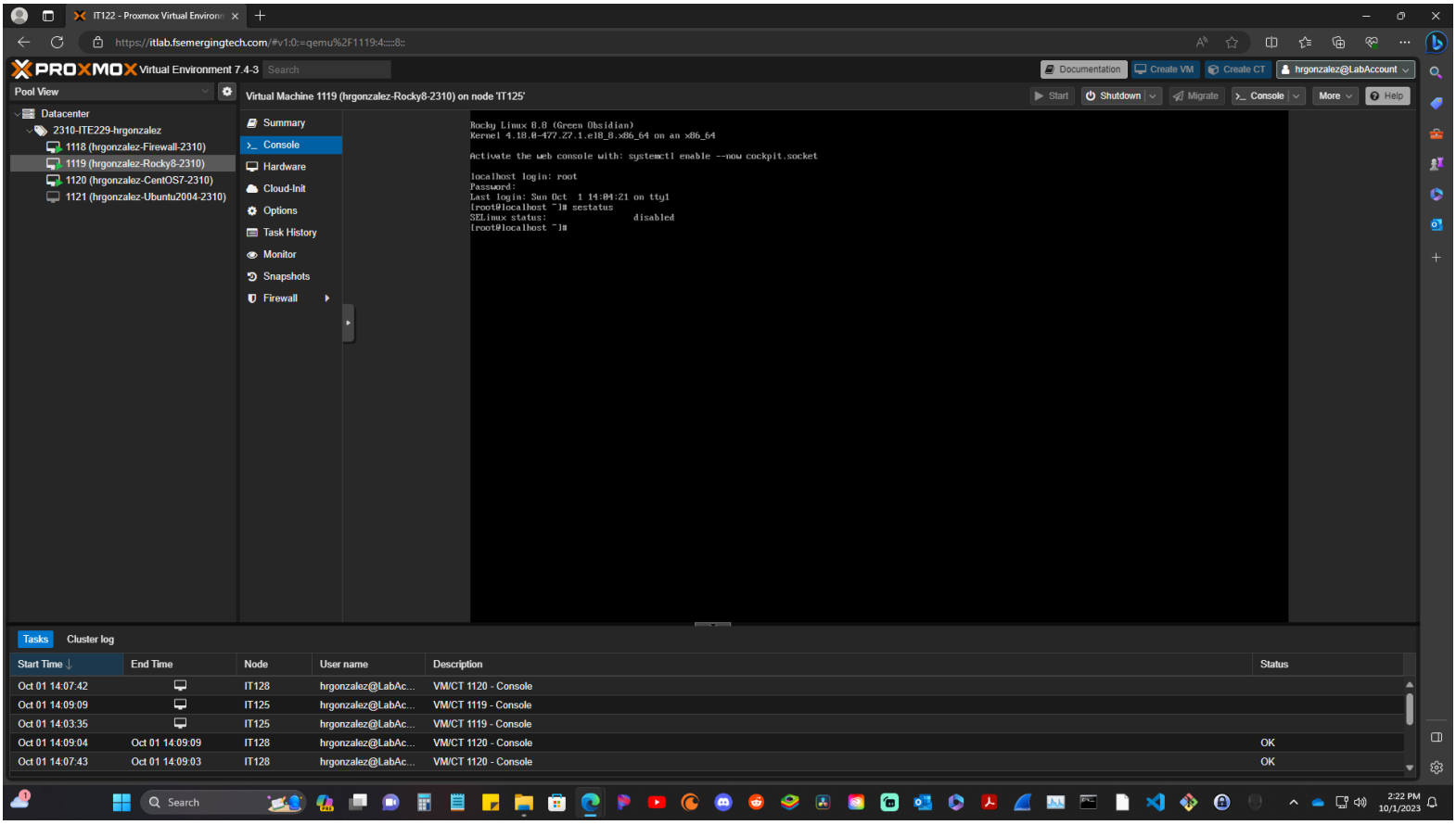
Test SELinux

Once back on the terminal enter command prompt sestatus. *This step and the one that follows are done at the same time, by inputting the sestatus command you are testing and at the same time seeing it's status.



Confirm SELinux Status

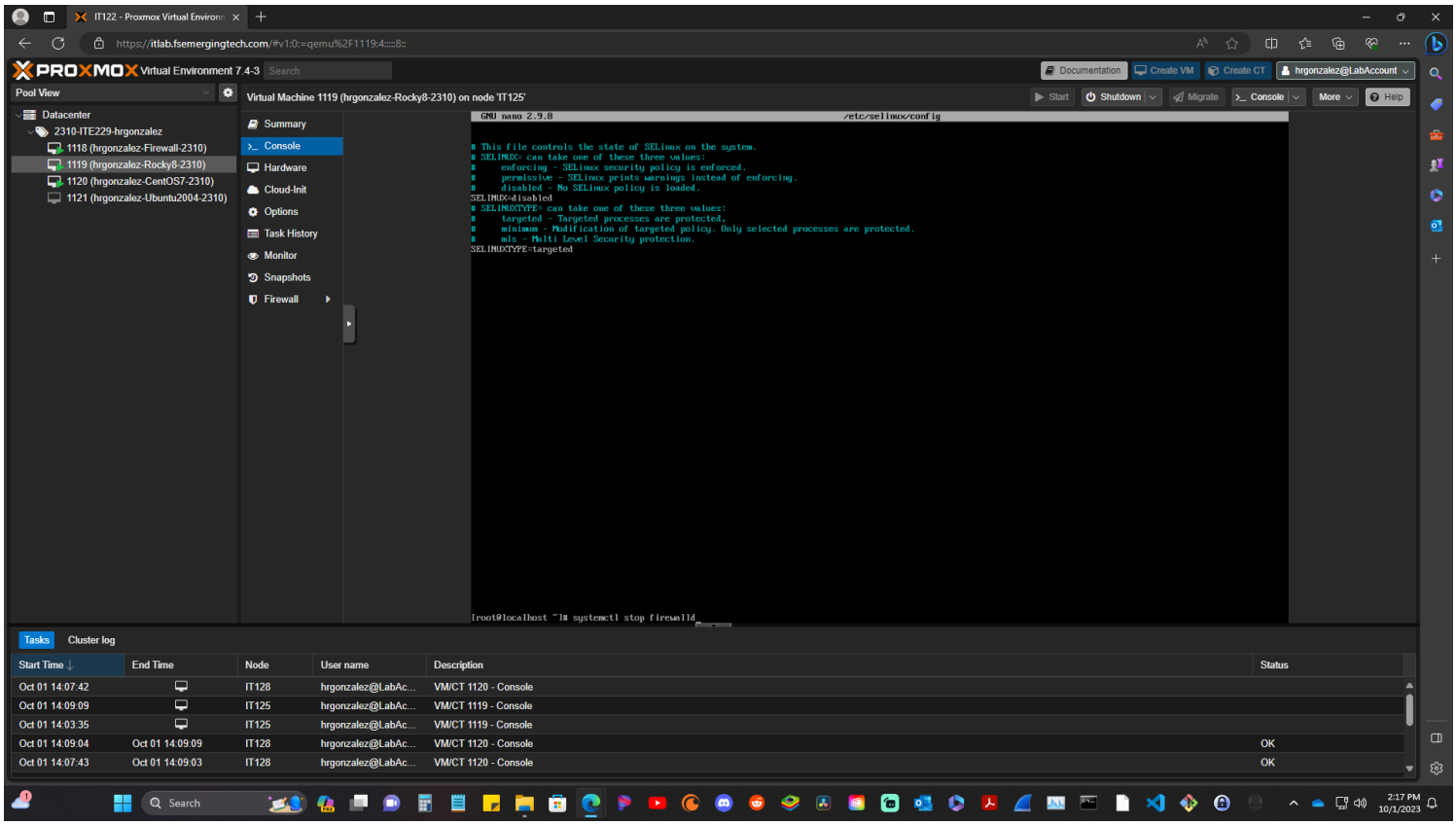
By inputting command prompt sestatus you are able to see if selinux is disabled.



Rocky Firewall

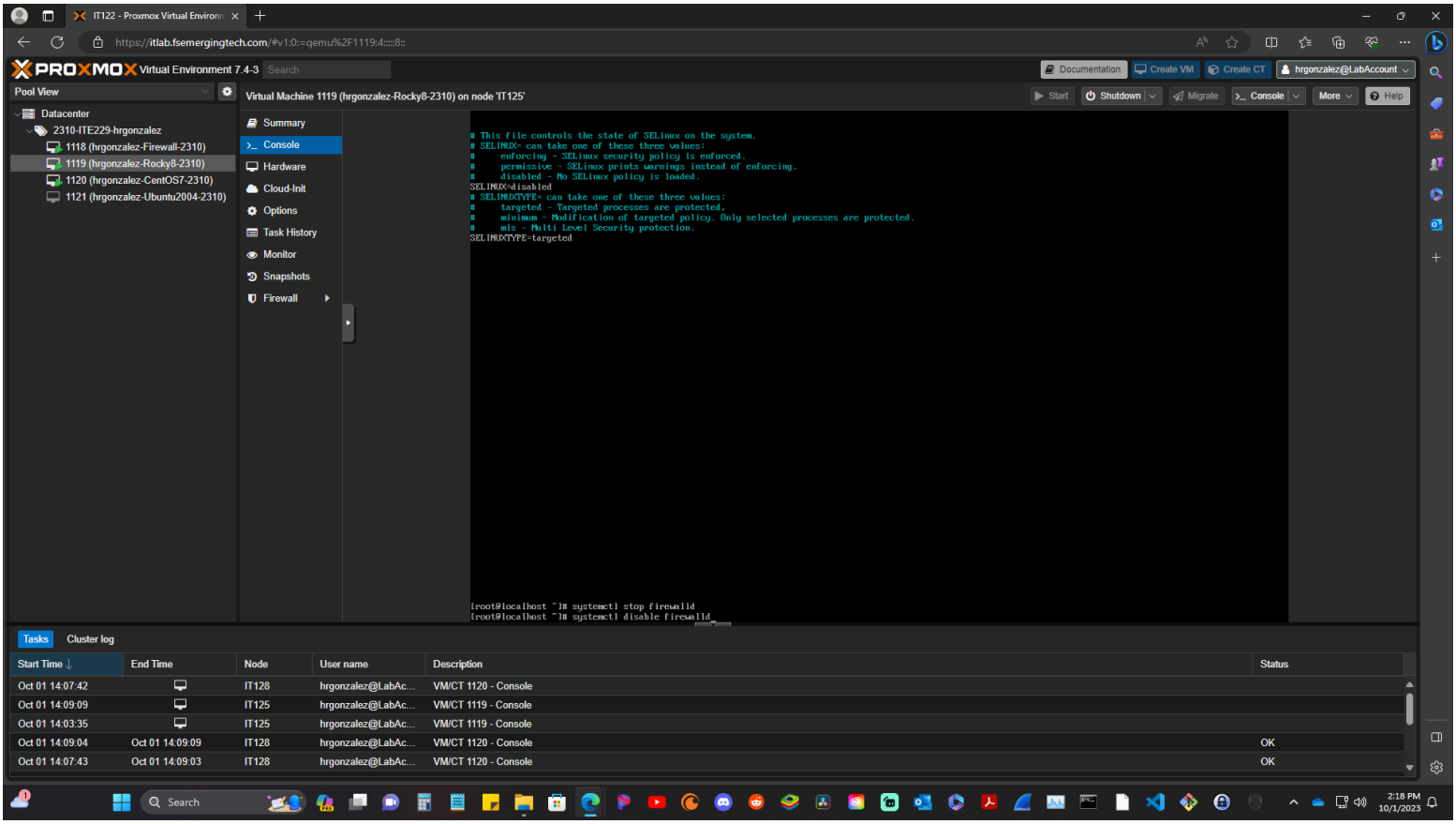
Stop Firewall

To stop firewall enter command prompt systemctl stop firewalld and hit enter. This will stop the firewall.



Disable Firewall

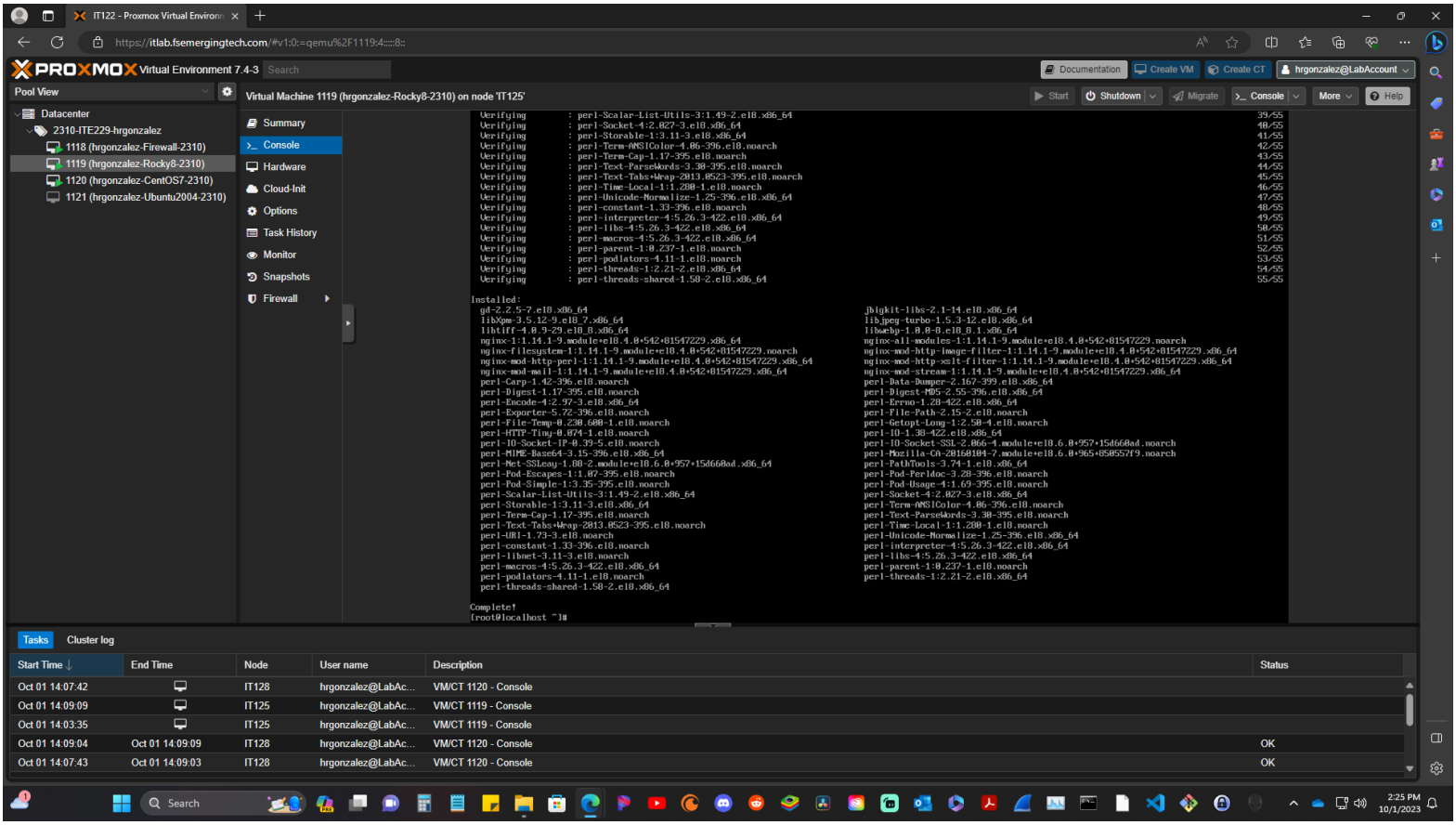
After enter command prompt systemctl disable firewalld and hit enter. This will disable the firewall.



NginX

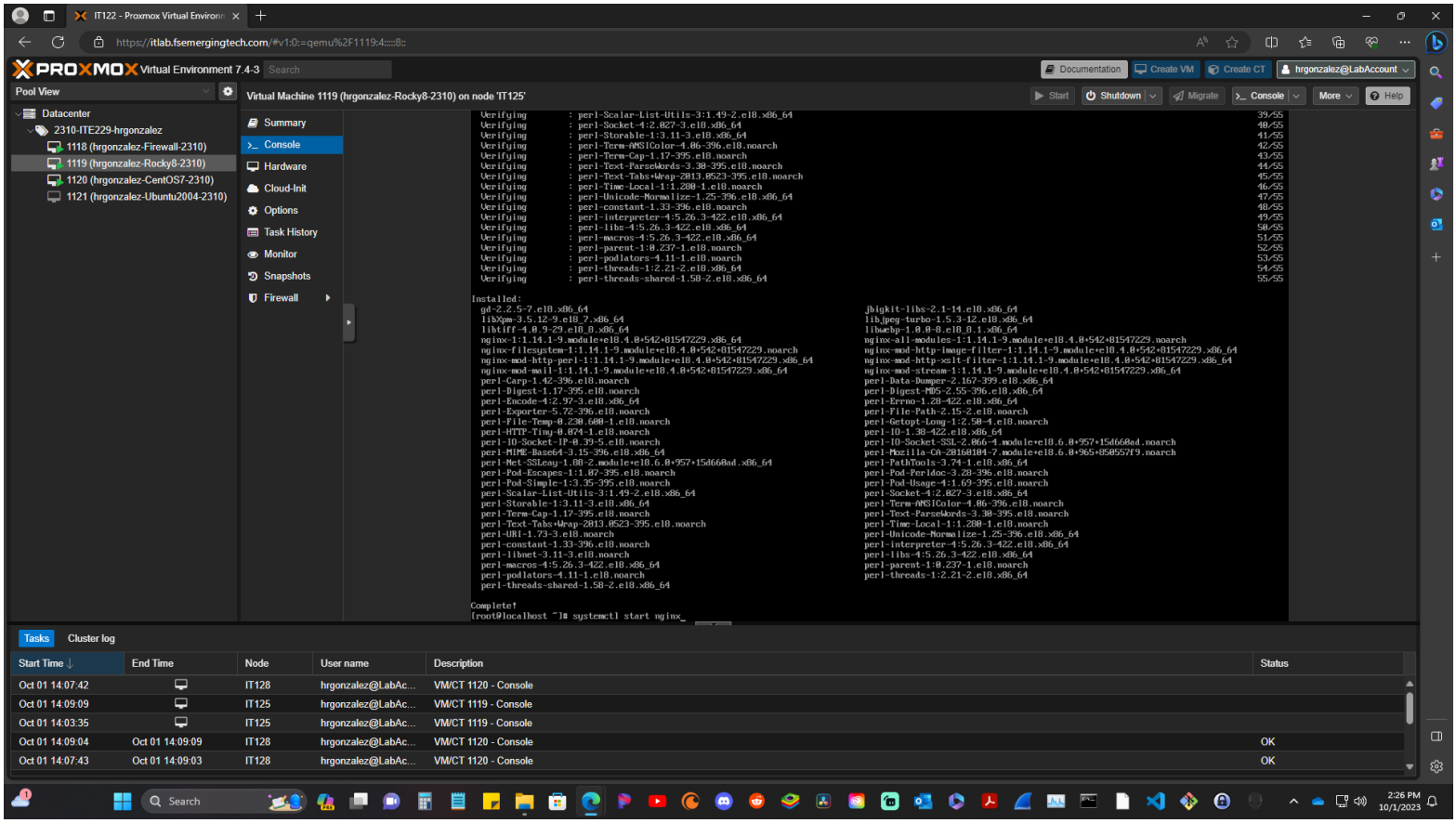
Install NginX

Next, we will be installing Nginx. In order to do so, we need to grab it from its location. Enter command prompt sudo yum install nginx -y. Let it run it's installation process, and after it finishes installing you should see a complete window like the one below.



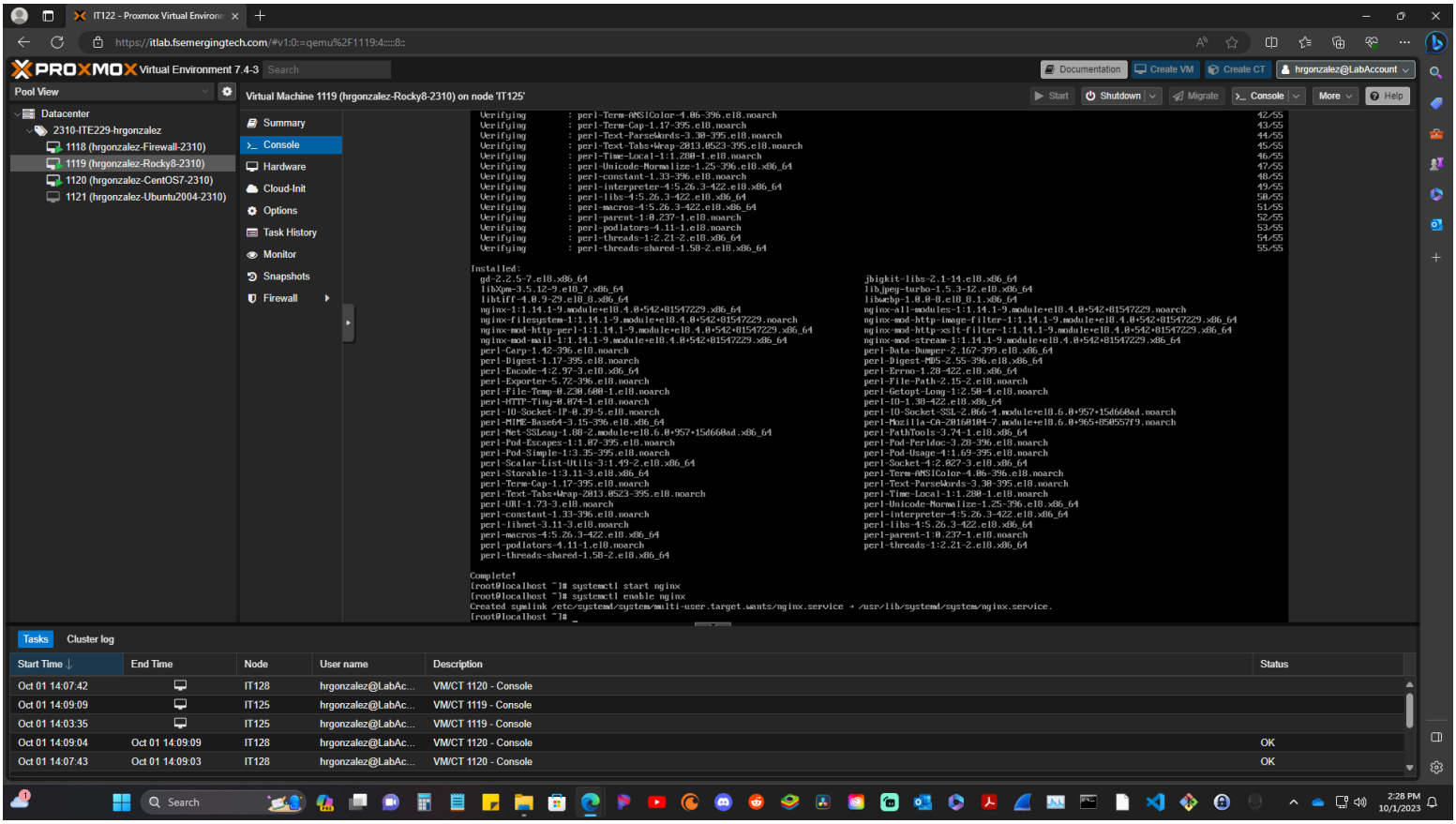
Start NginX

Next we need to start Nginx by inputting command prompt systemctl start nginx then hit enter.



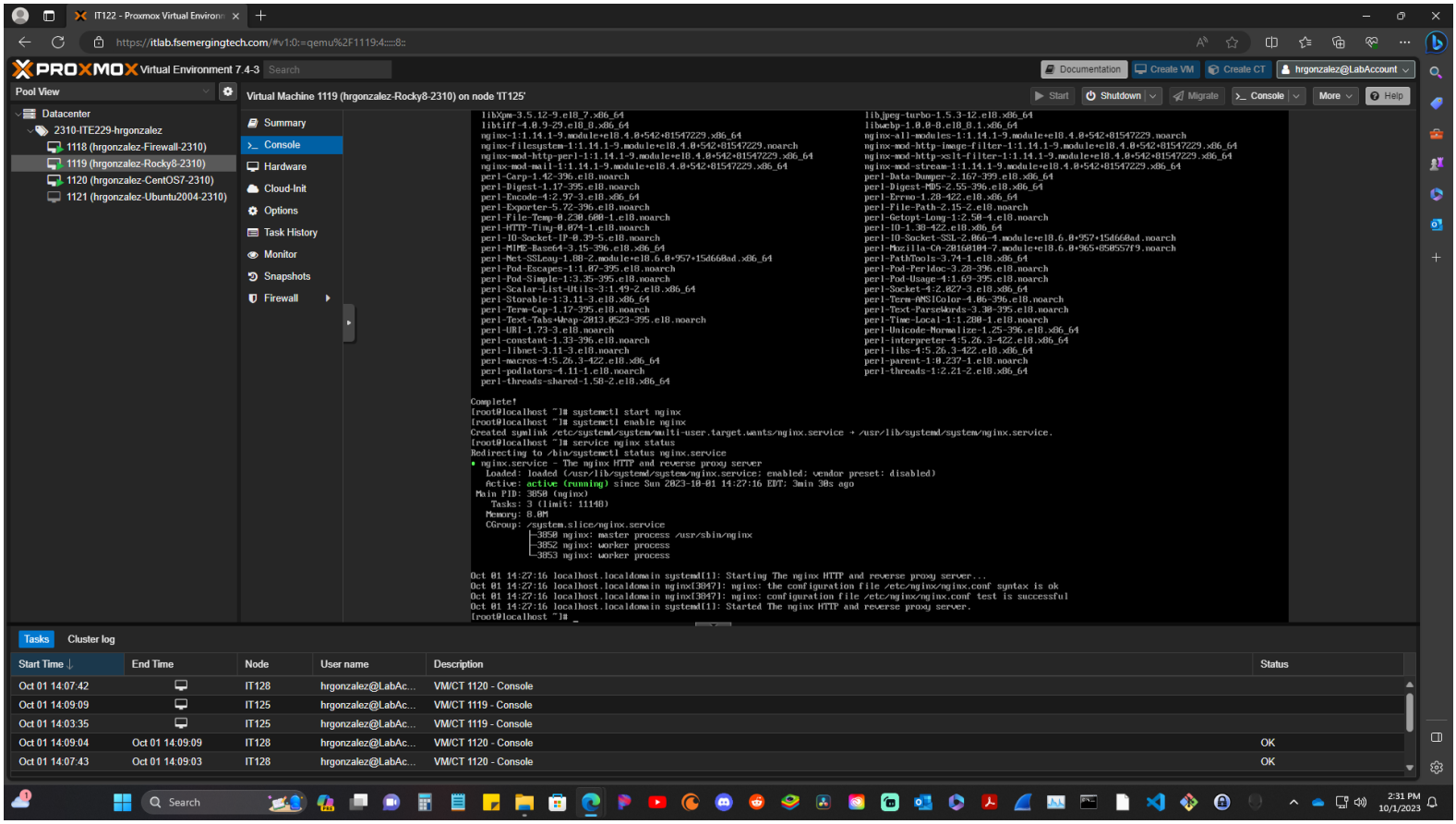
Enable NginX

Now we need to enable Nginx. In order to do this input command prompt systemctl enable nginx then hit enter. It should create a symlink and if everything is good to this point your terminal should look like the picture below.



Test NginX

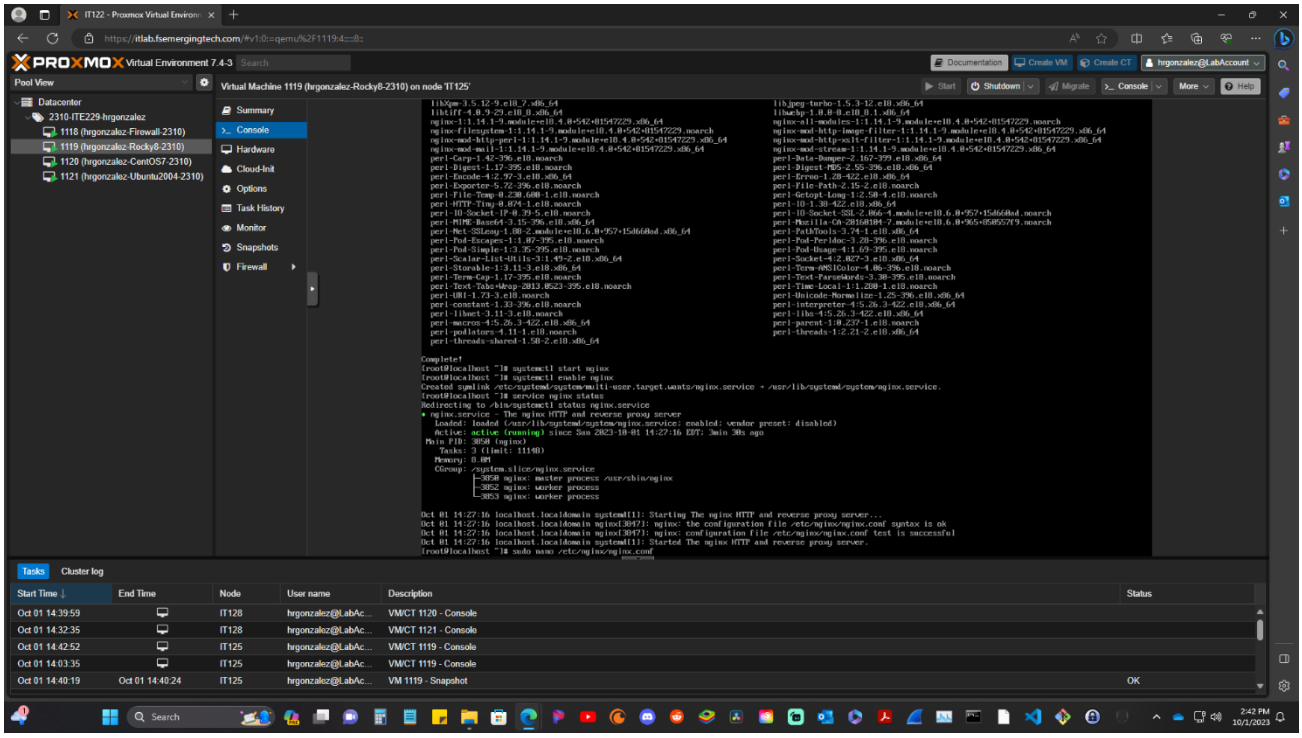
Time to test Nginx, by inputting command prompt service nginx status you should see that nginx is actively running.



Reverse Proxy for Ghost Site

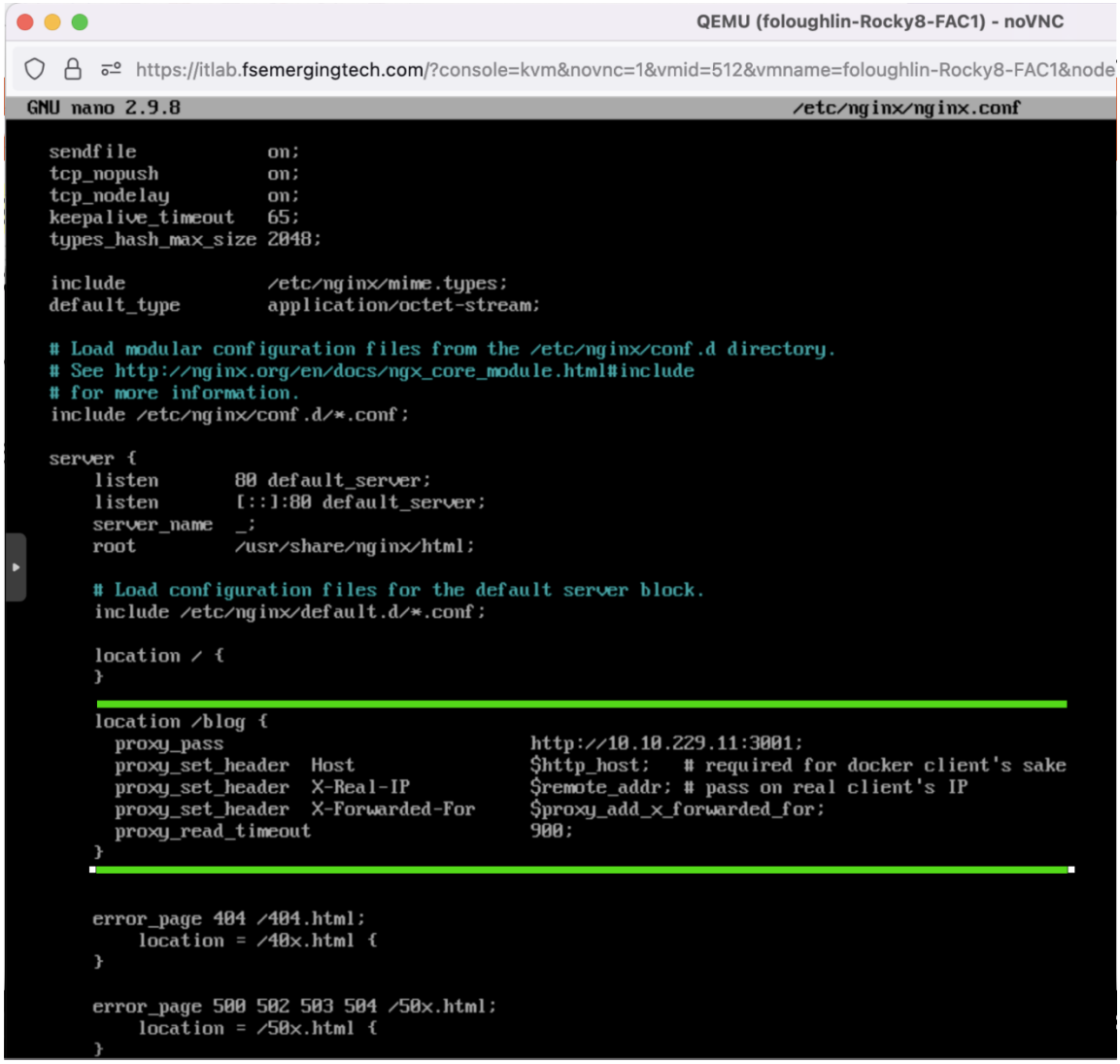
Edit NginX configuration file

This part involves a couple of steps. Input command prompt sudo nano /etc/nginx/nginx.conf this will take you to the nginx configuration file. In this file we are going to add some changes.



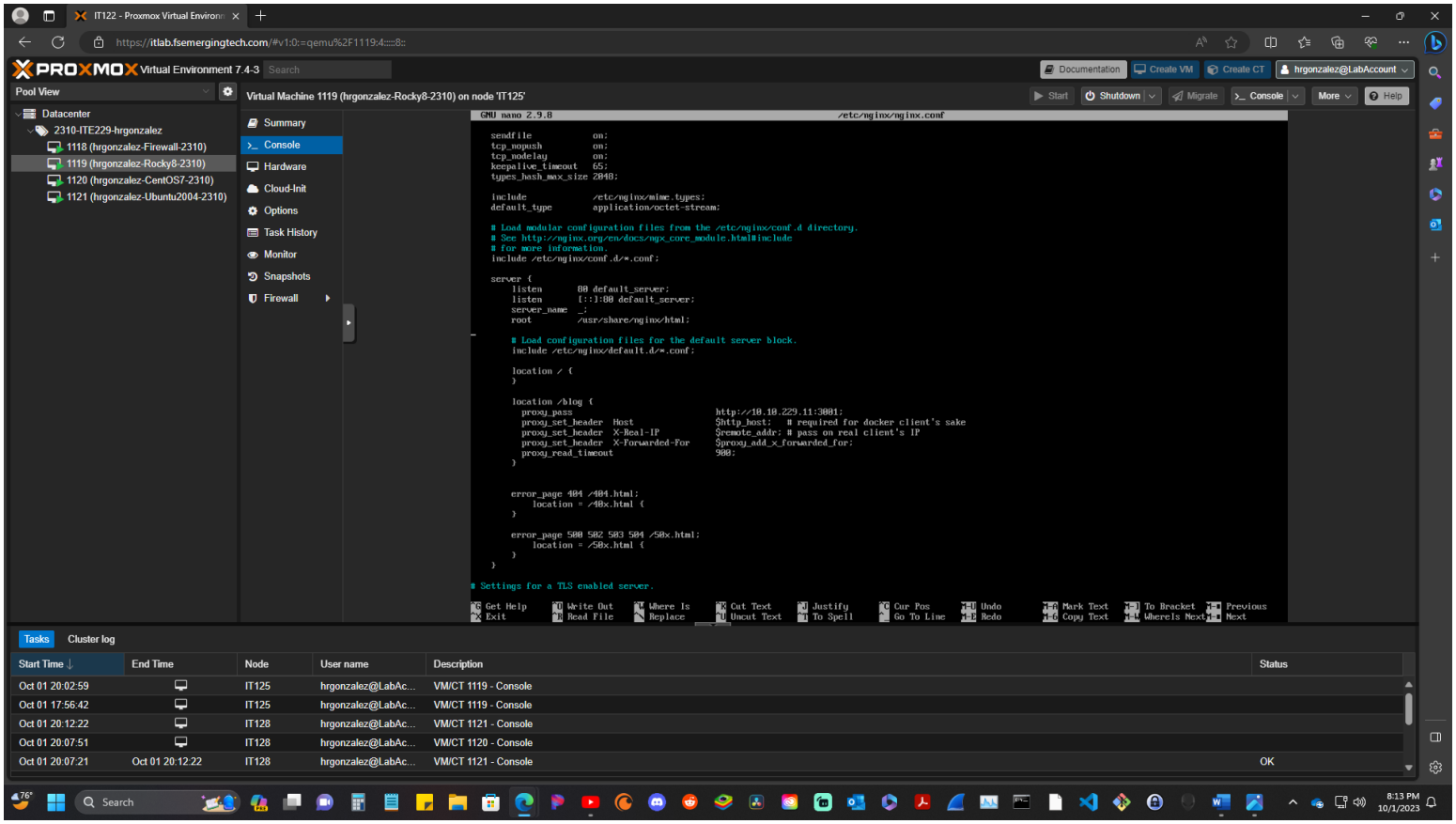
Once inside the configuration file you want to scroll down using the arrow key until you hit the section (*Don't pay attention to the text in between both green lines, this is what we are going to be adding*) that say # Load configuration in turquoise color and under the turquoise color it should say include /etc/nginx follow by location underneath. Use picture as reference. Once in this location you are going to type in the following code. Use the picture as reference because everything needs to be exactly the same.

```
location /blog {
  proxy_pass                          http://10.10.229.11:3001;
  proxy_set_header  Host              $http_host;   # required
for docker client's sake
  proxy_set_header  X-Real-IP         $remote_addr; # pass on
real client's IP
  proxy_set_header  X-Forwarded-For
$proxy_add_x_forwarded_for;
  proxy_read_timeout                  900;
}
```



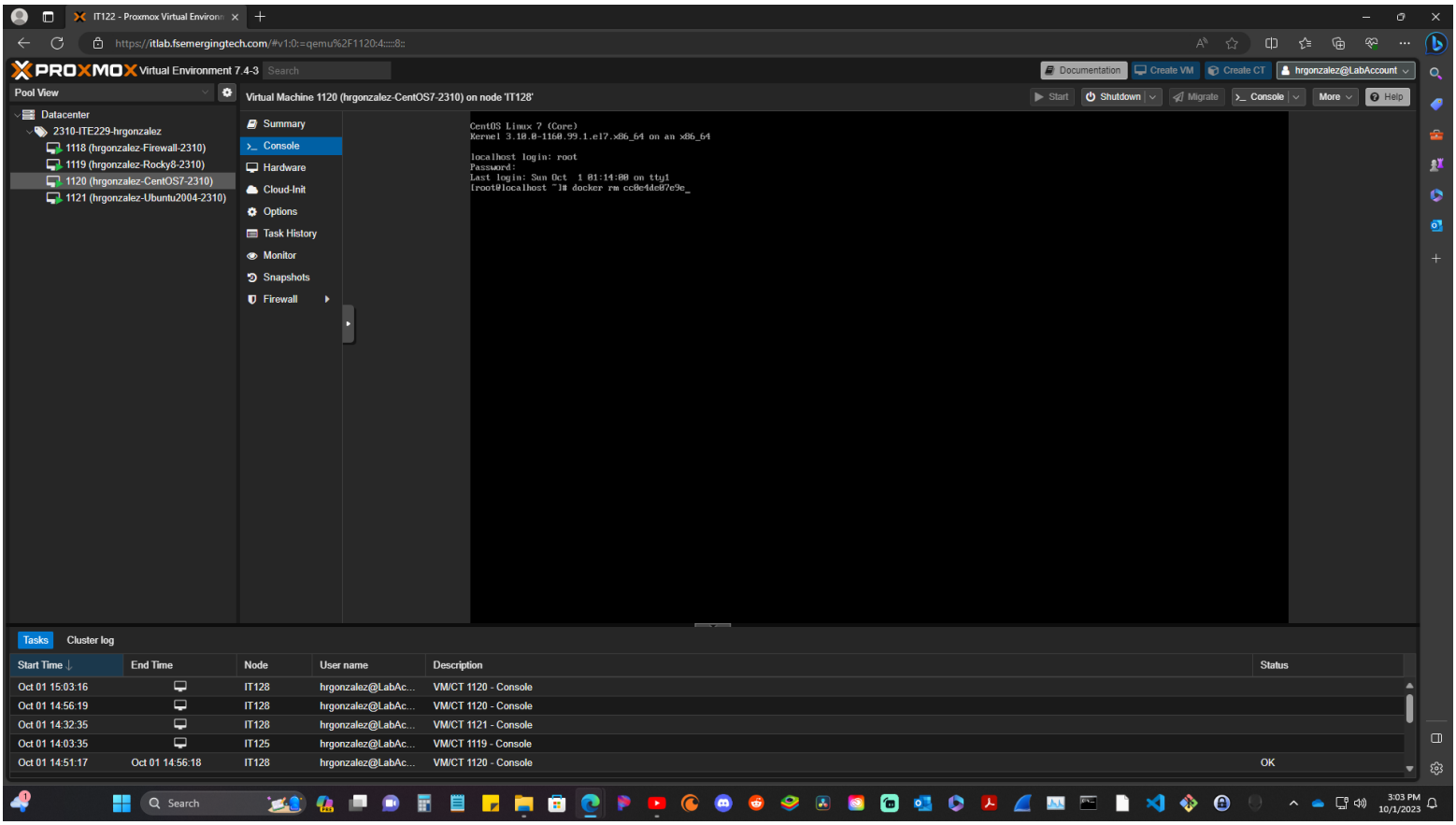
Reload NginX service

Once you finish typing everything as instructed hit control key and E to exit, it will prompt if you want to save changes make sure you hit Y for yes then hit enter. Once you hit enter it will take you back to the root user just like in the picture below saving all the changes you just made.



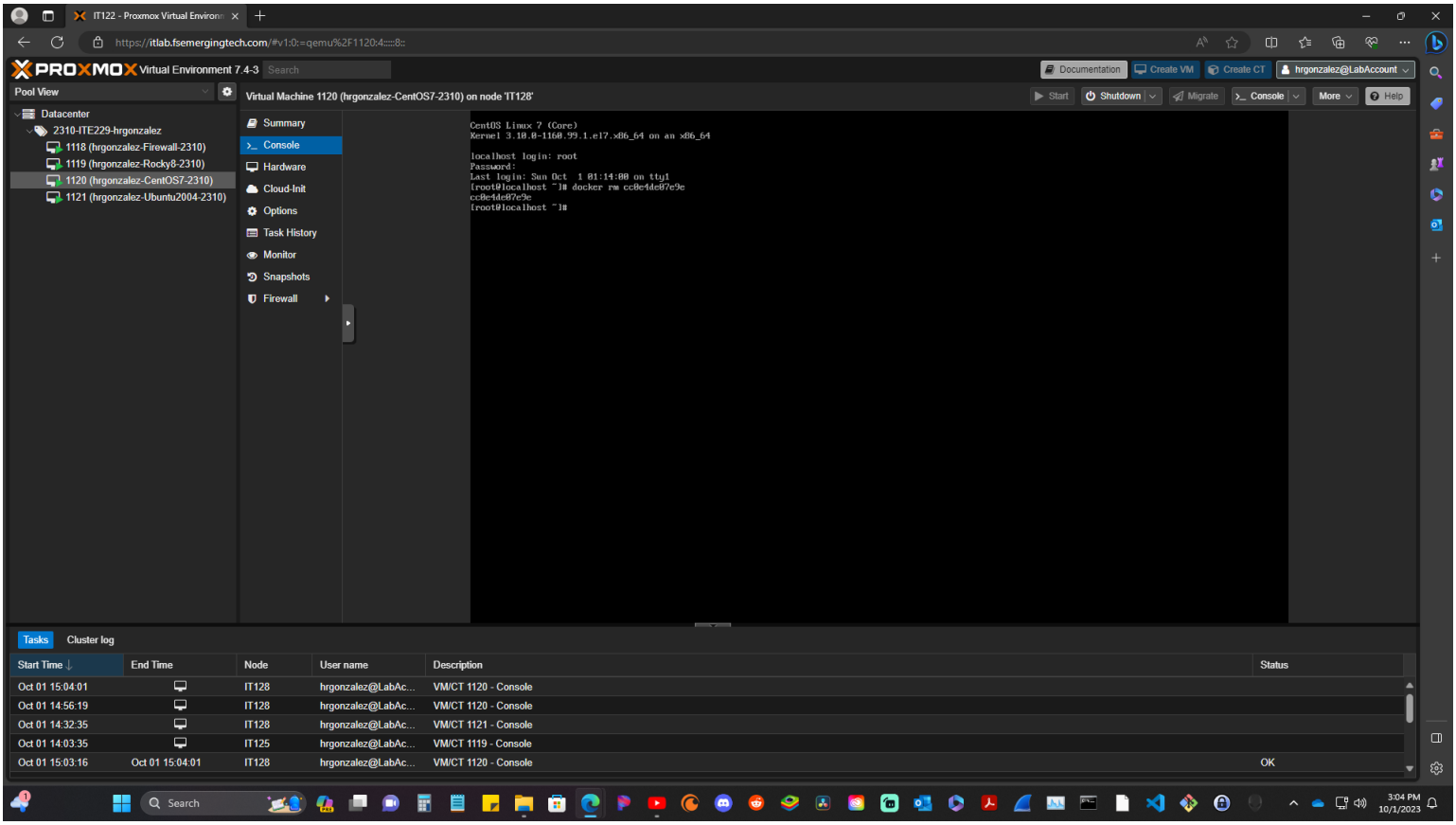
Terminate Docker

Now we need to terminate Docker. Input command prompt docker rm _____ in the blank you need to type in the container id you saved/wrote down from previous steps.



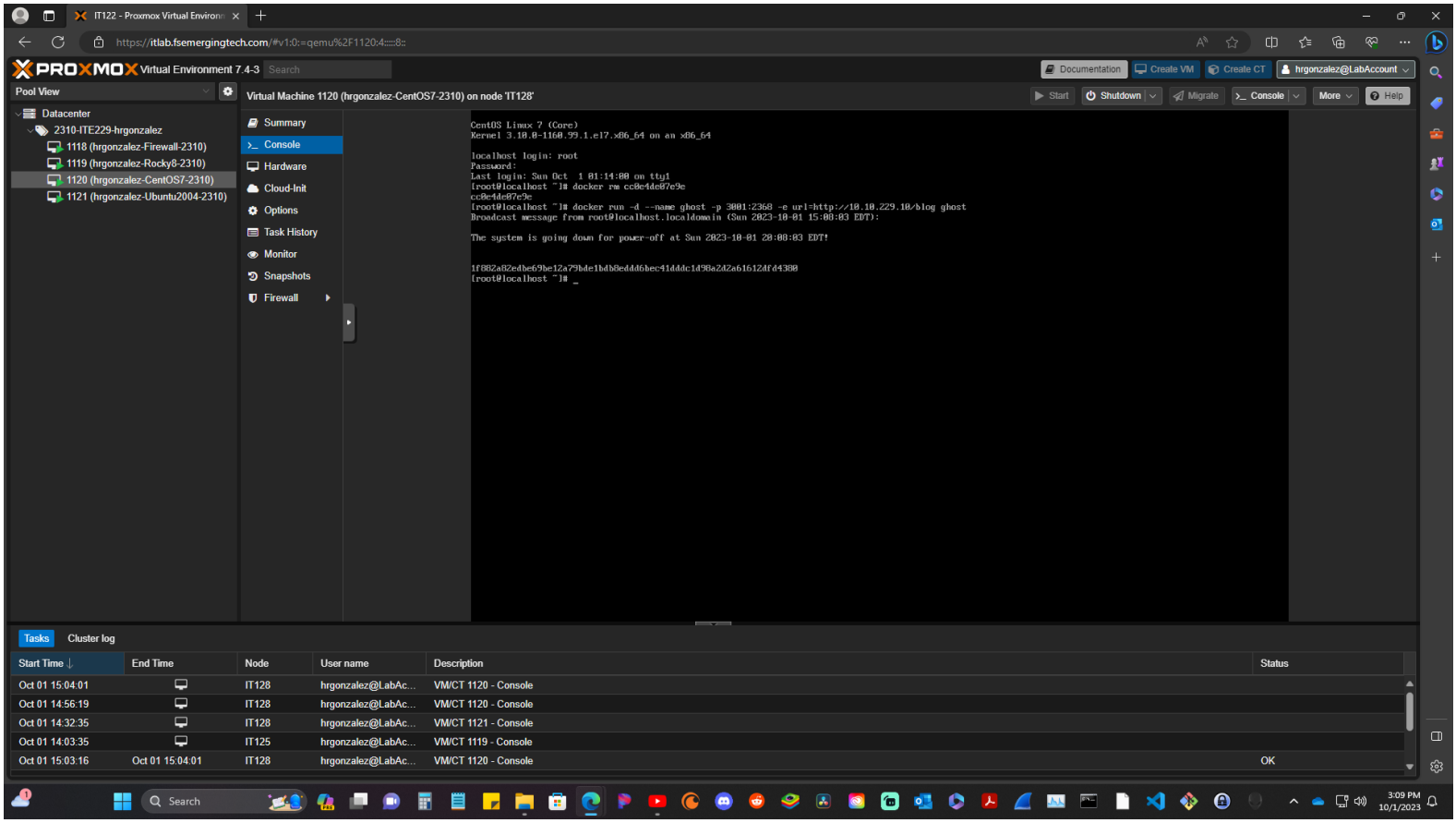
Delete Ghost Container

Once you input the command prompt from above and hit enter you will see the id will appear again and then redirect you to root username. This means it was succesfully deleted.



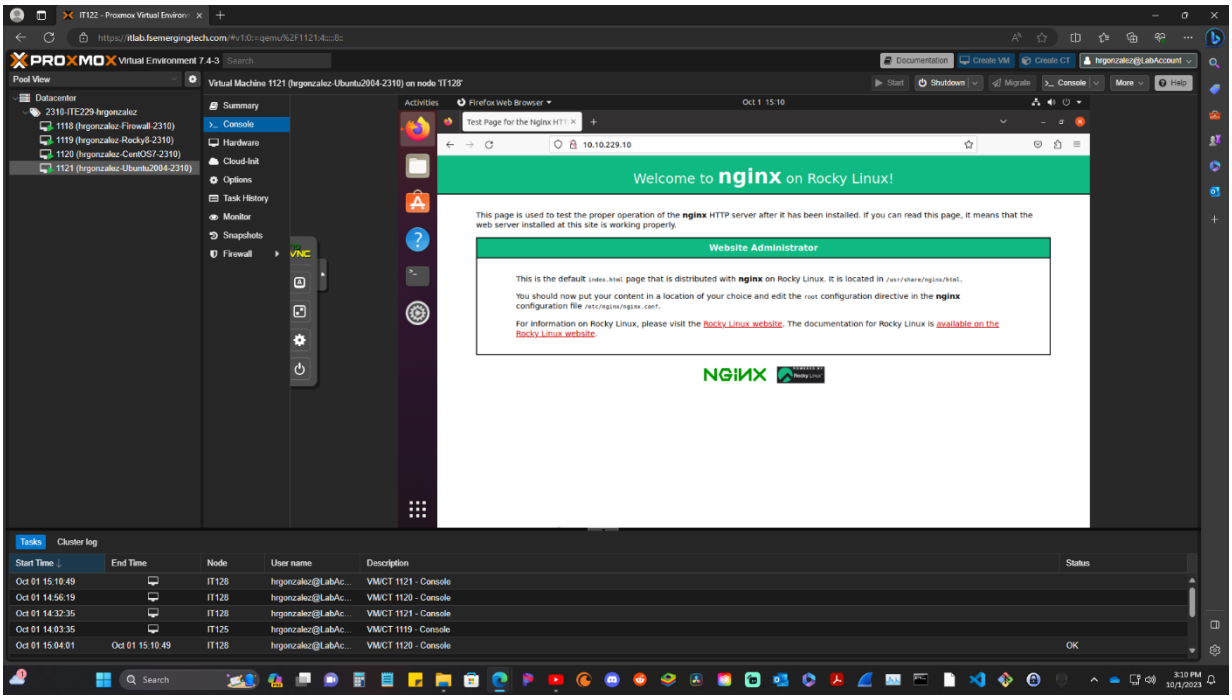
Create New Ghost Container

Now we need to create a new ghost container and in order to do this you need to type command prompt docker run -d –name ghost -p 3001:2368 -e url=http://10.10.229.10/blog ghost. This will create a new ghost container id. If succesful it should give you a long name with letters and numbers before returning you back to your root username.

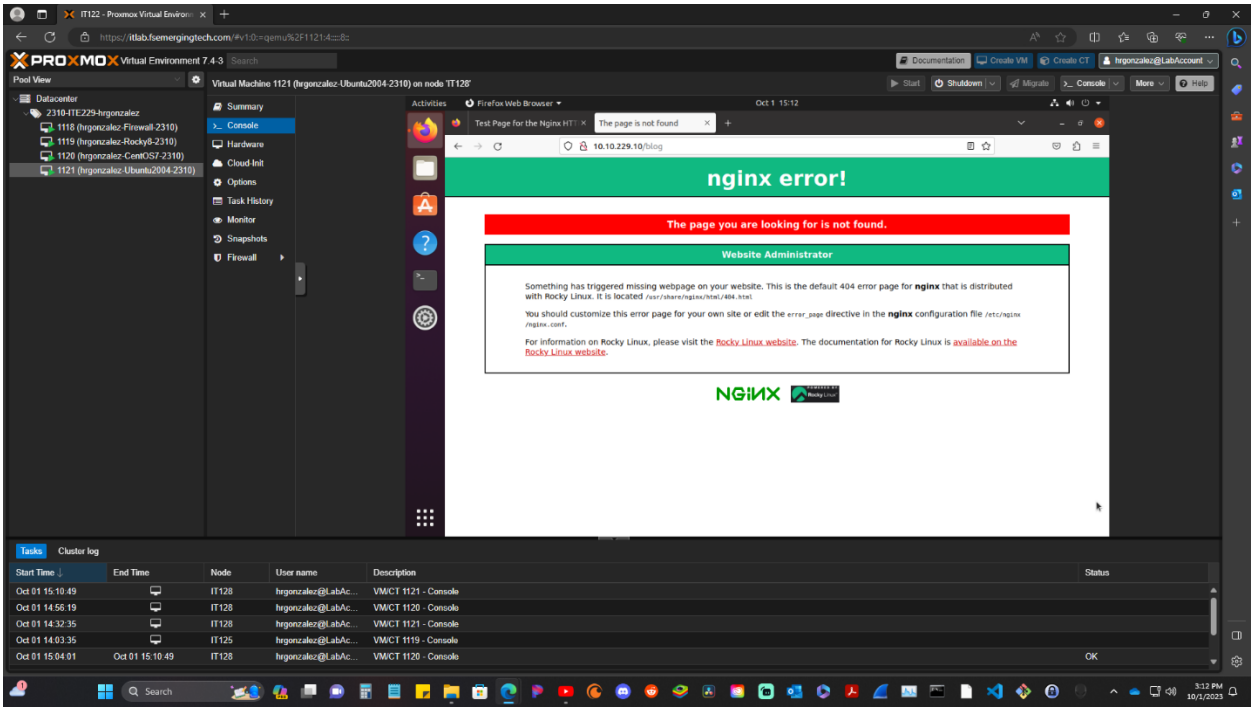


Browse to Ghost

Now we need to verify everyhting is working properly with Nginx and the blog ip address. Log in to your Ubuntu VM using the credentials user and Fullsail1! for the password. Once in Ubuntu VM go to firefox web browser and type http://10.10.229.10. in the url. You should get a window that looks llike the one below.

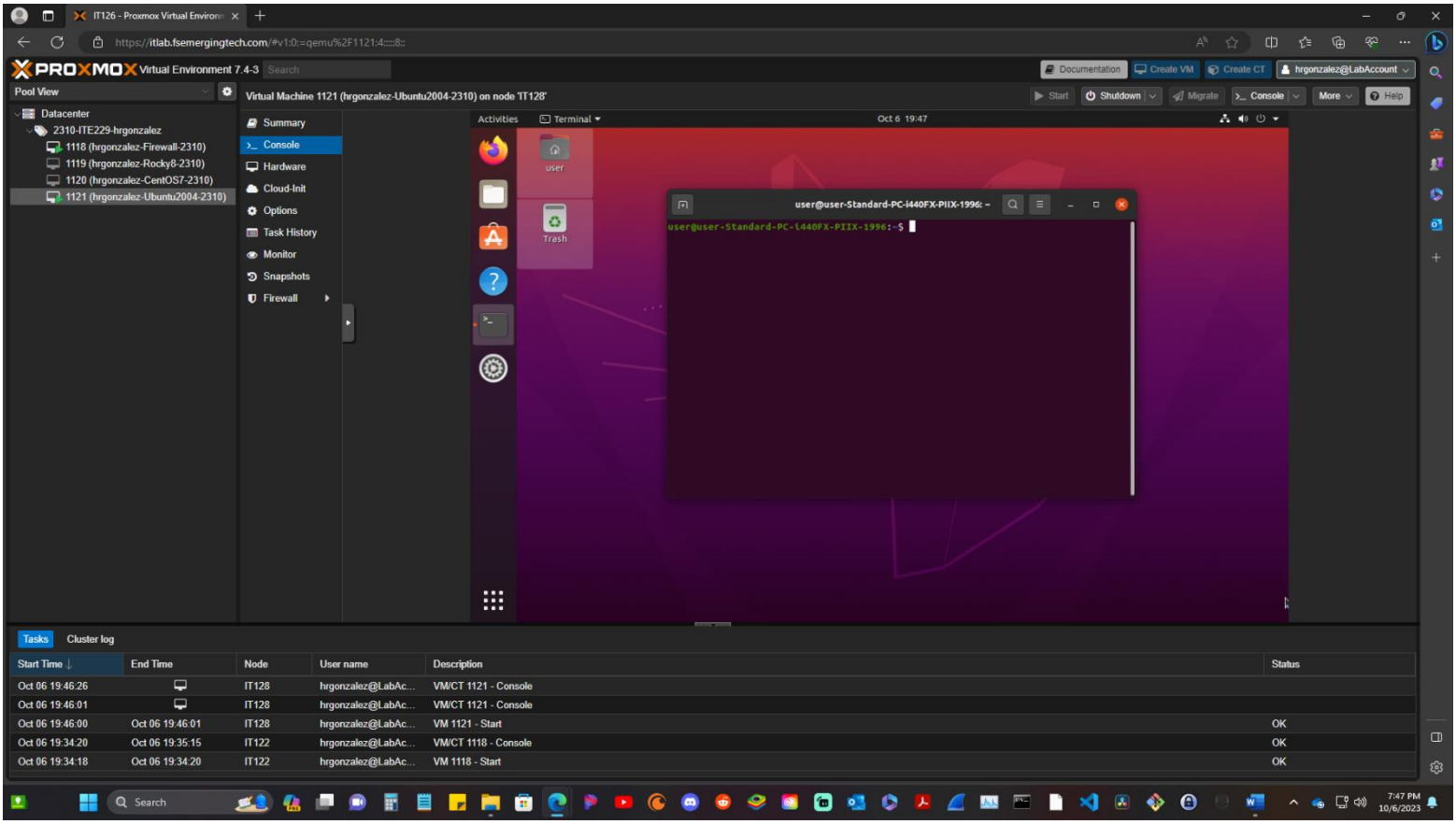


Now open a new window in firefox and type in http://10.10.229.10/blog in the url. You should get a window like the one below. Don't worry about the error, because we are still not finished setting up Word press, we are only verifying Nginx.



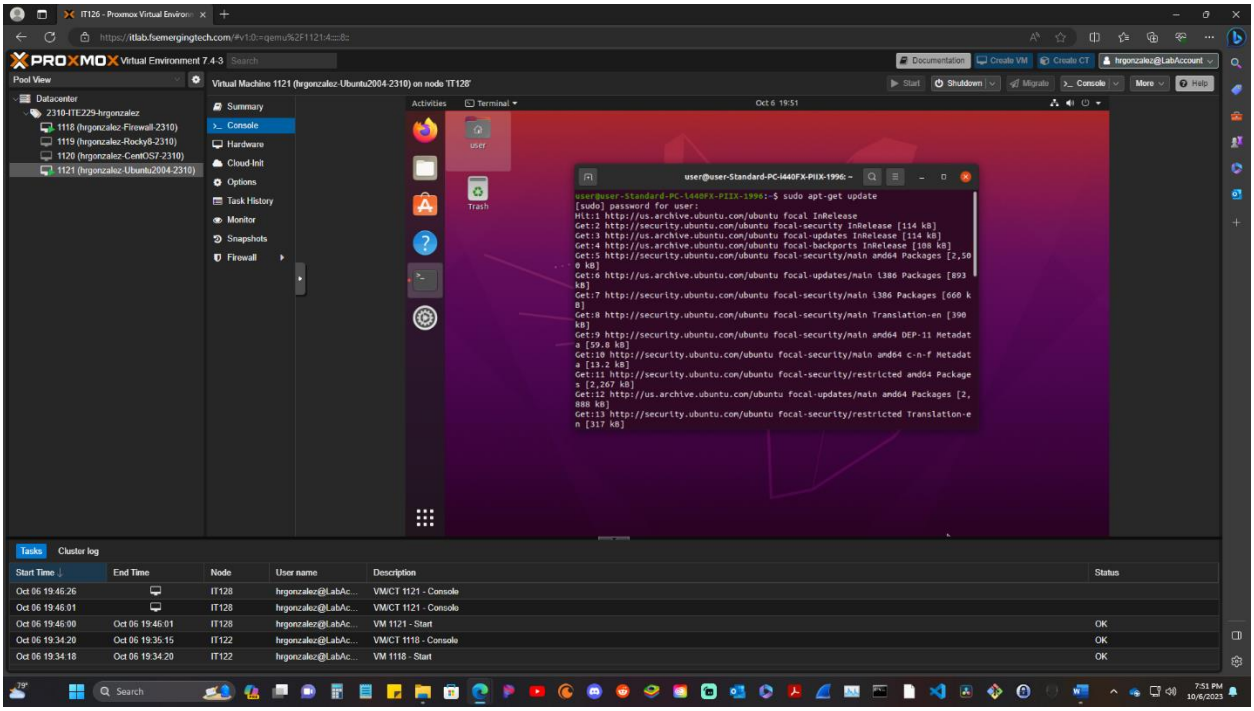
WordPress on Ubuntu - LAMP Stack

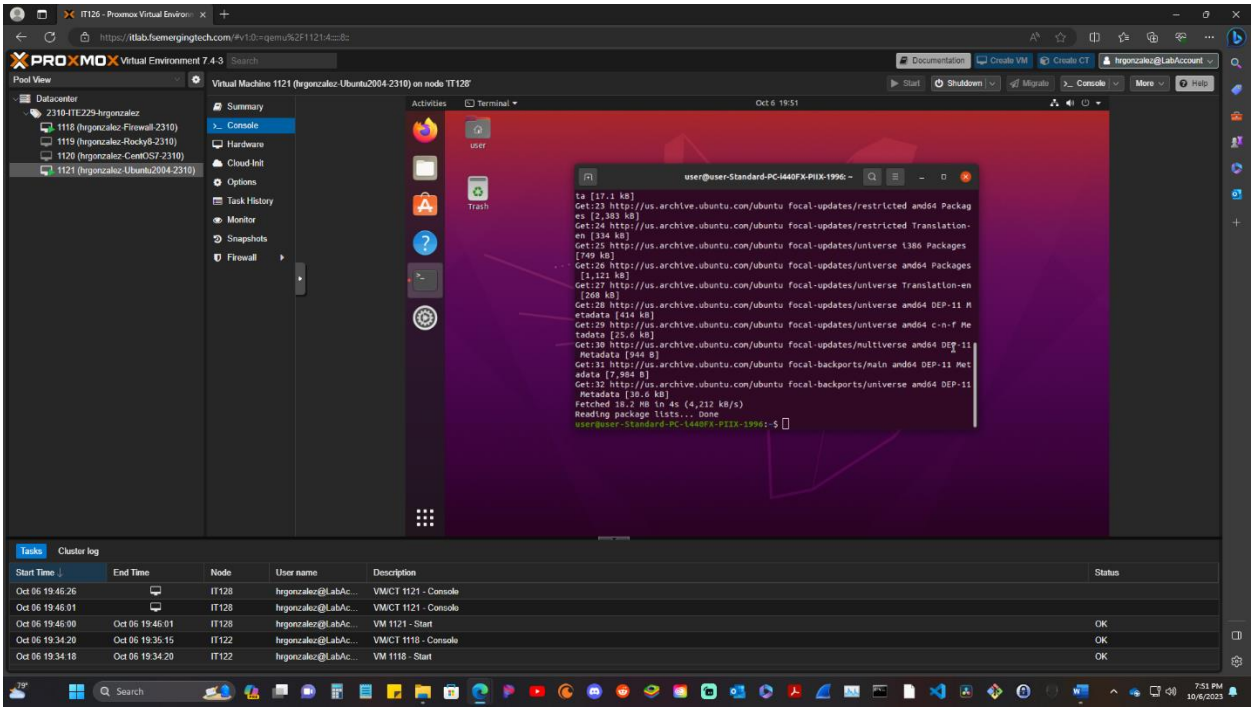
Show screenshot of your Ubuntu Console in VE



Update Ubuntu

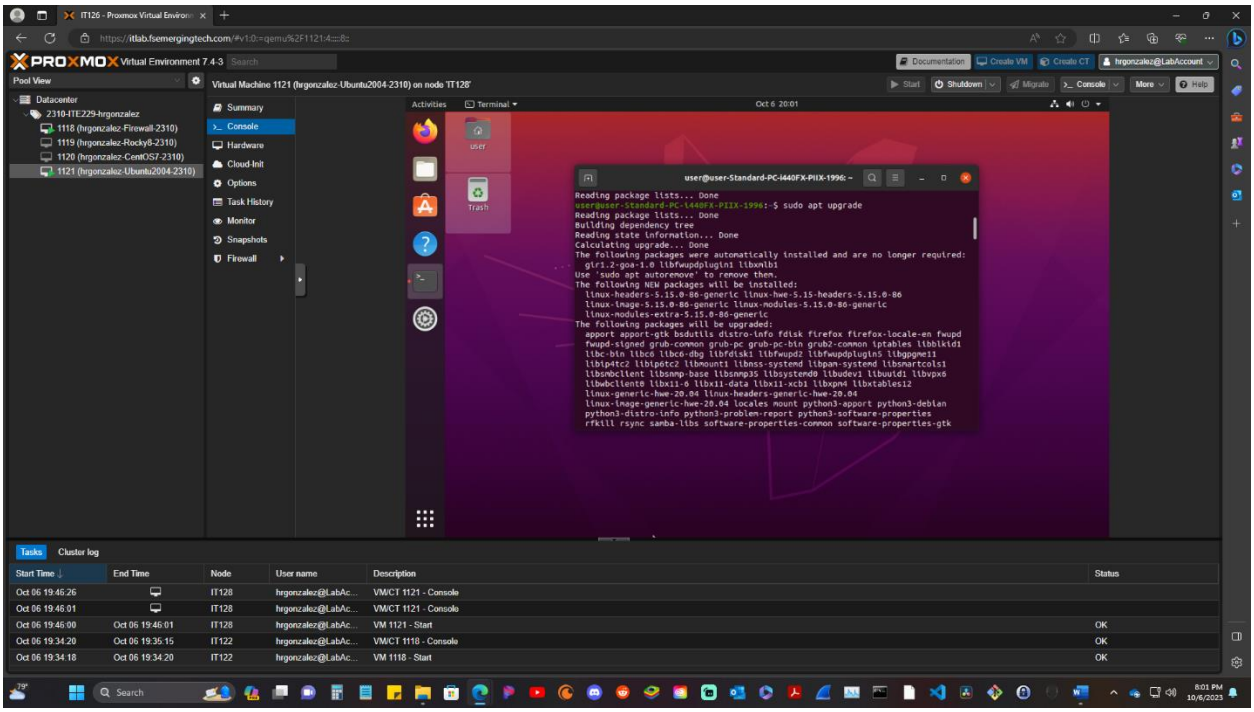
Open up Ubuntu Terminal and type command sudo apt-get update to update UBUNTU.

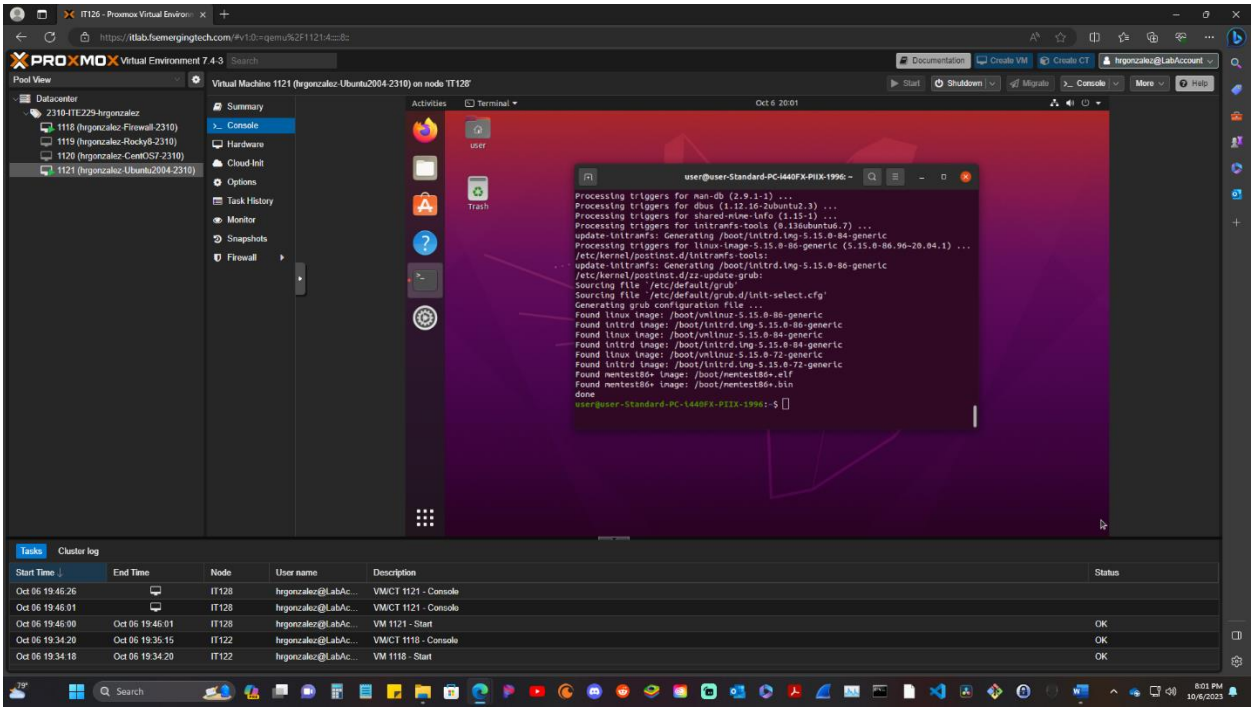




Upgrade Ubuntu

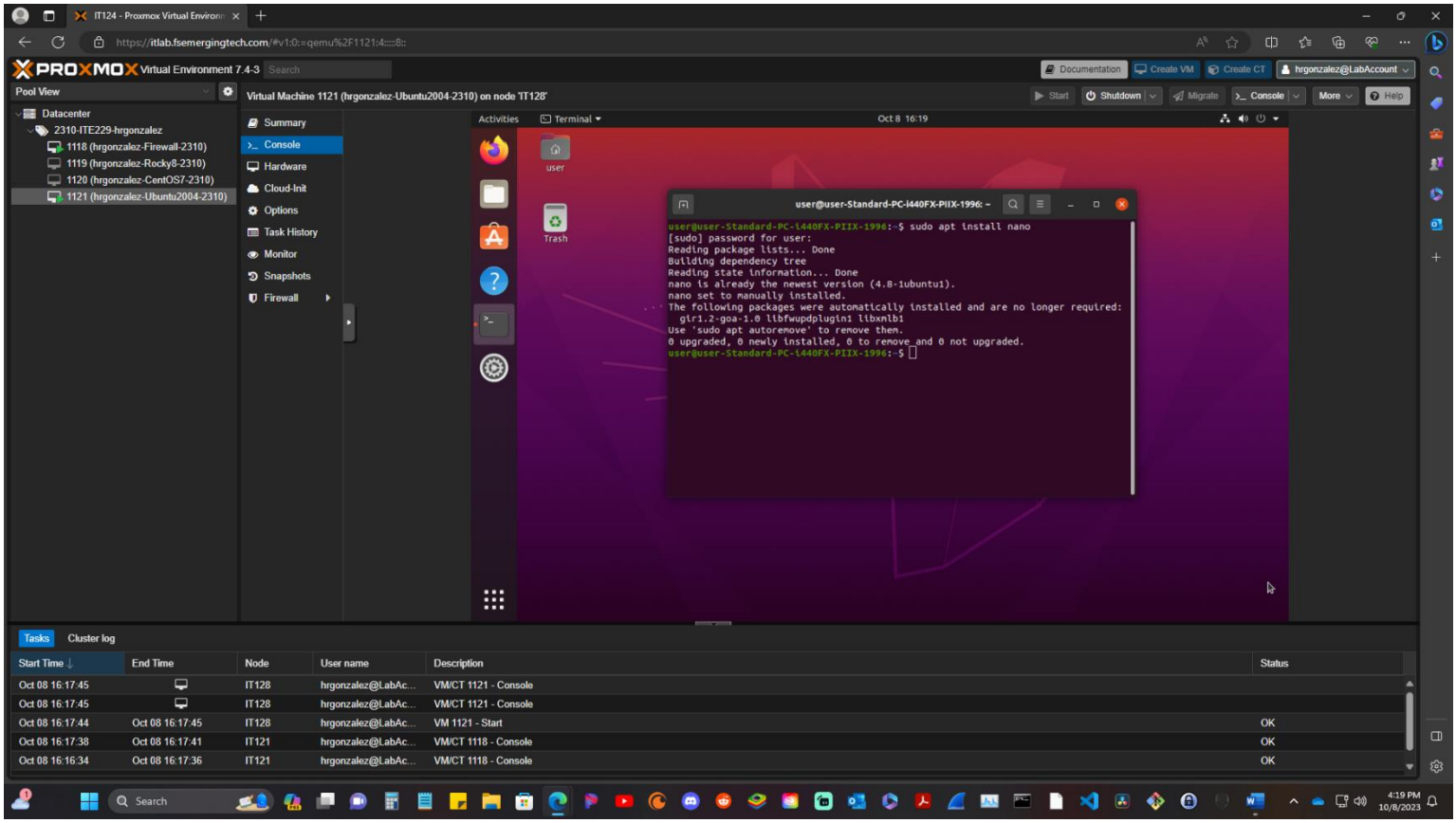
Next type command sudo apt upgrade to upgrade all packages to the latest versions. *When asked if you want to upgrade input y for yes, and it will upgrade all packages.





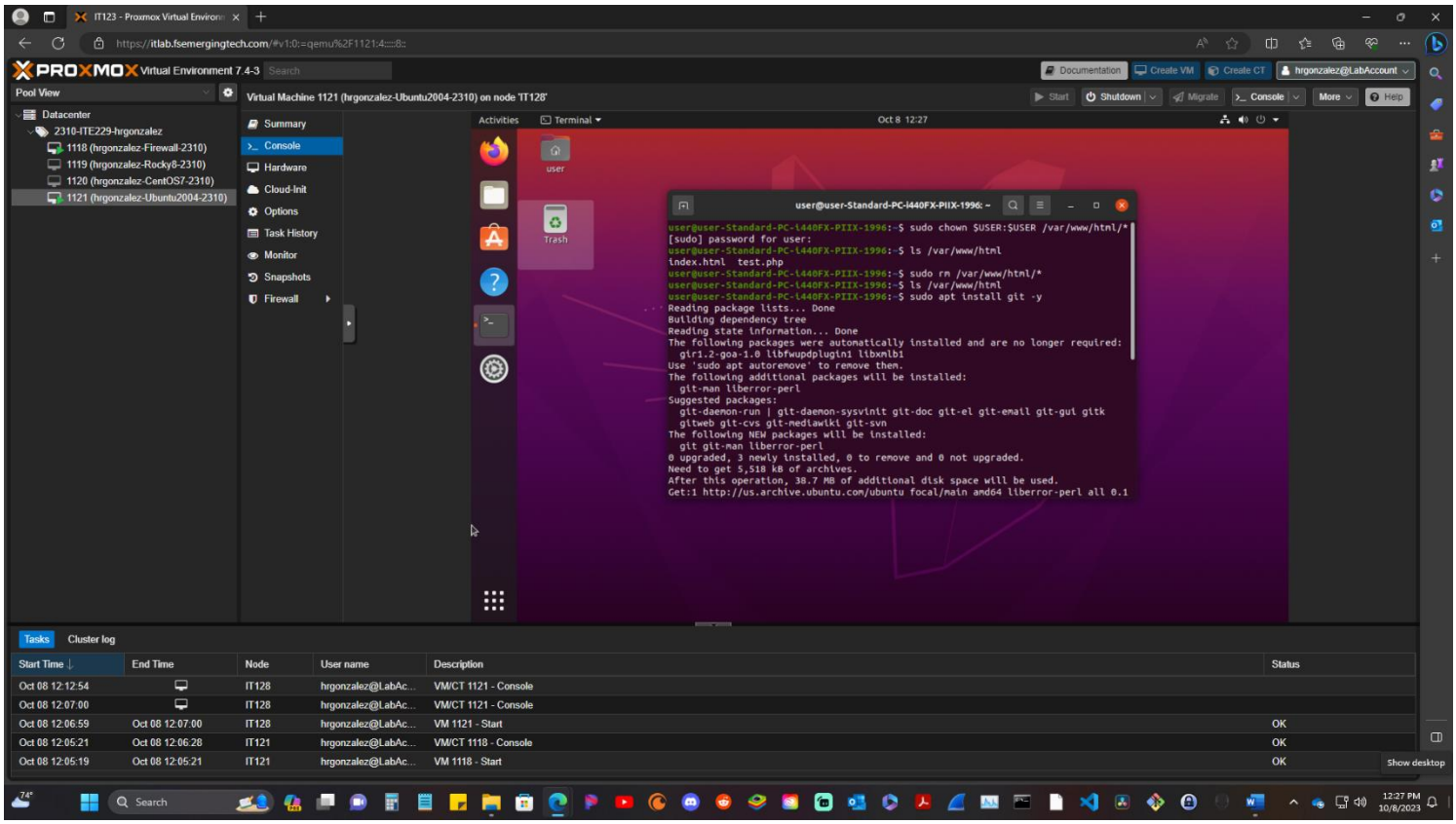
Install Nano Editor

To install nano you need to enter commmand line sudo apt install nano. If is not installed it will undergo the installation process. If it was previously installed they won't be any actions taken.



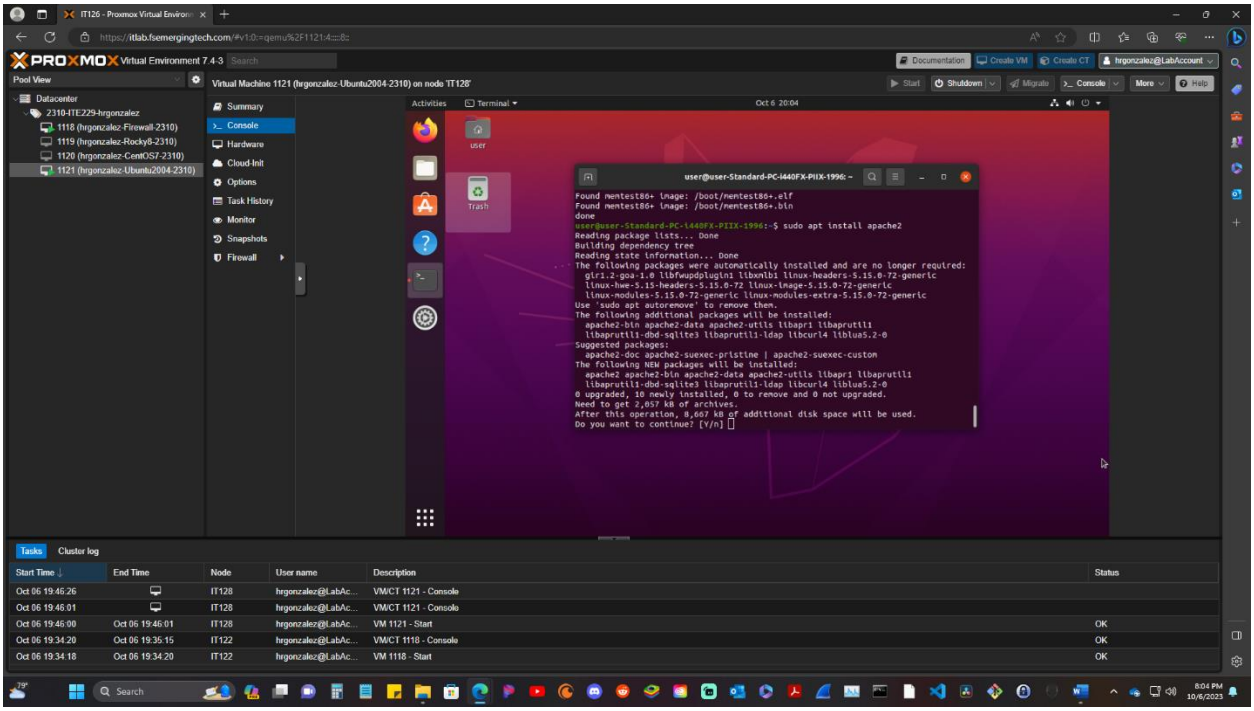
Install Git

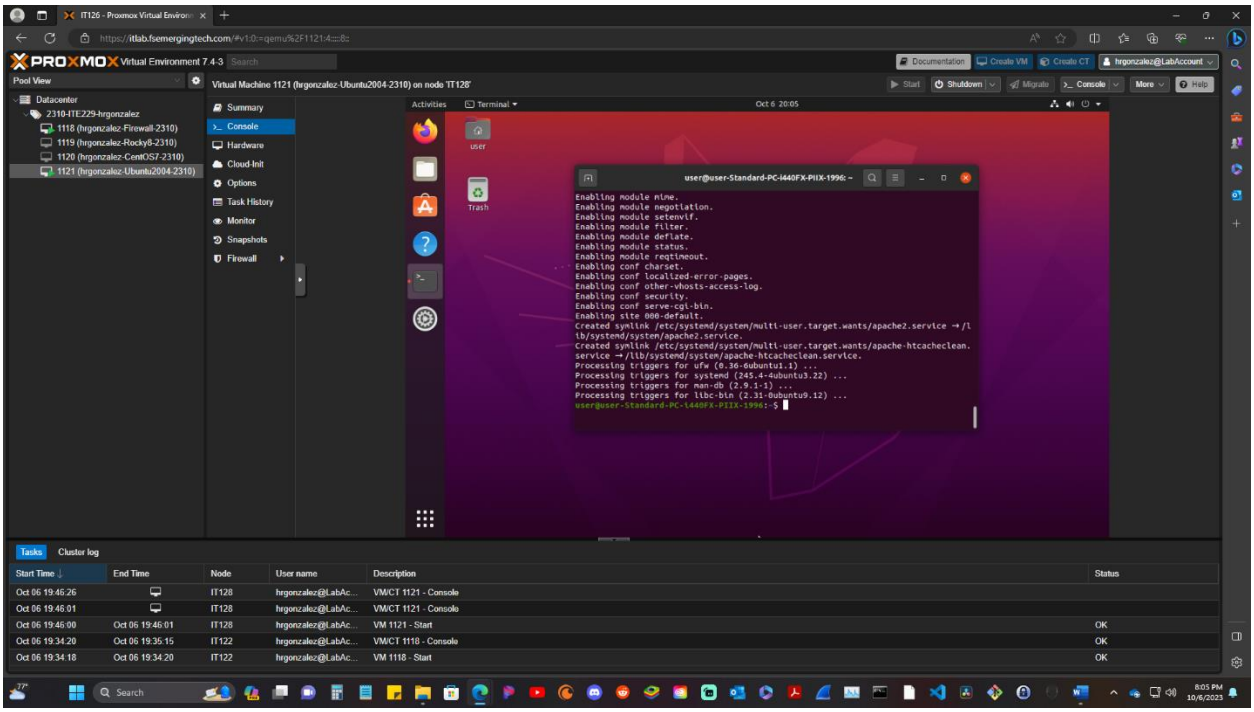
To install git you need to enter command line sudo apt install git -y. Wait until it completes installation before proceeding.



Install Apache2

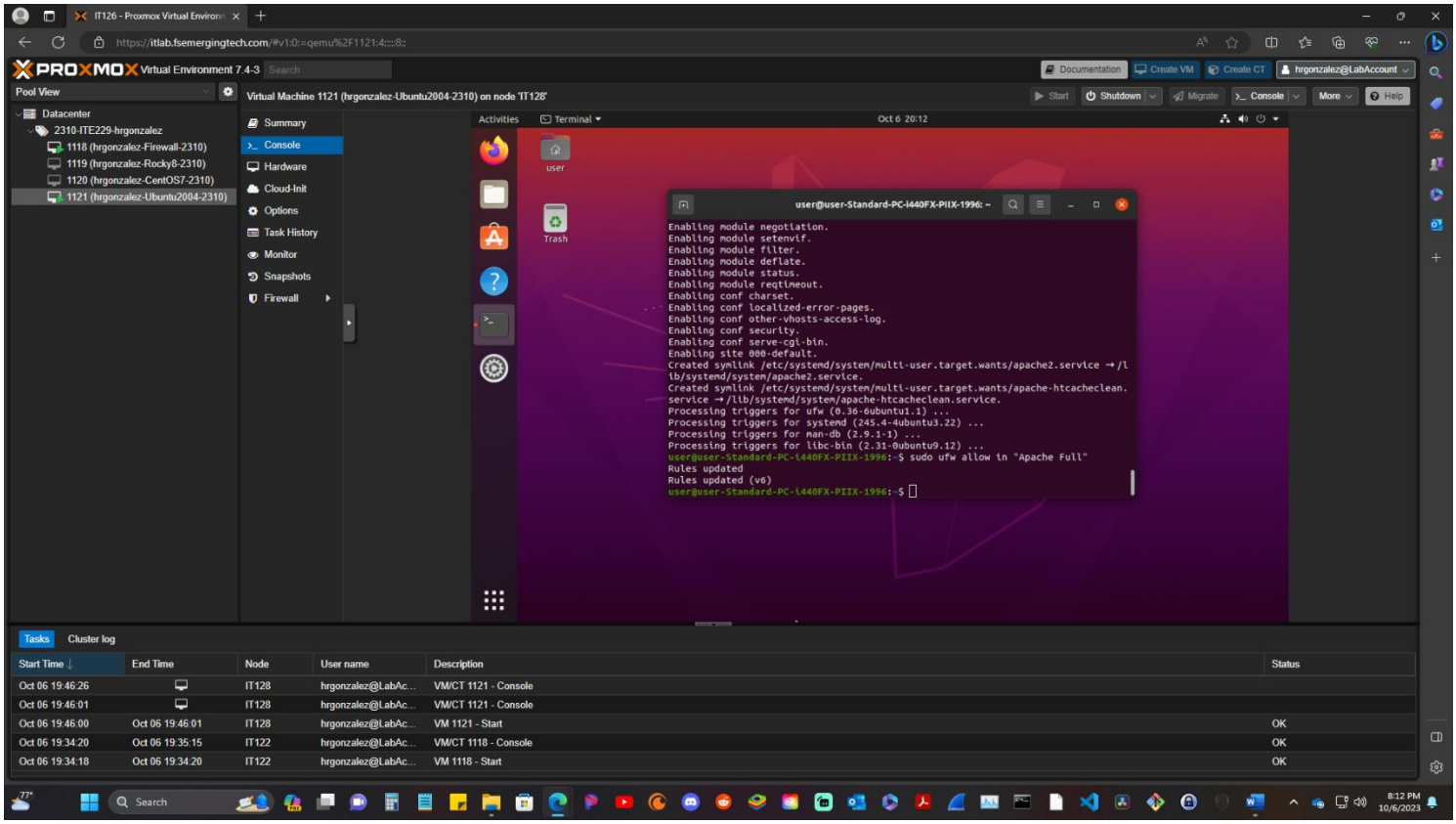
Enter command prompt sudo apt install apache2 to install Apache web server package. *When asked if you want to continue input y for yes, and it will continue with the install.





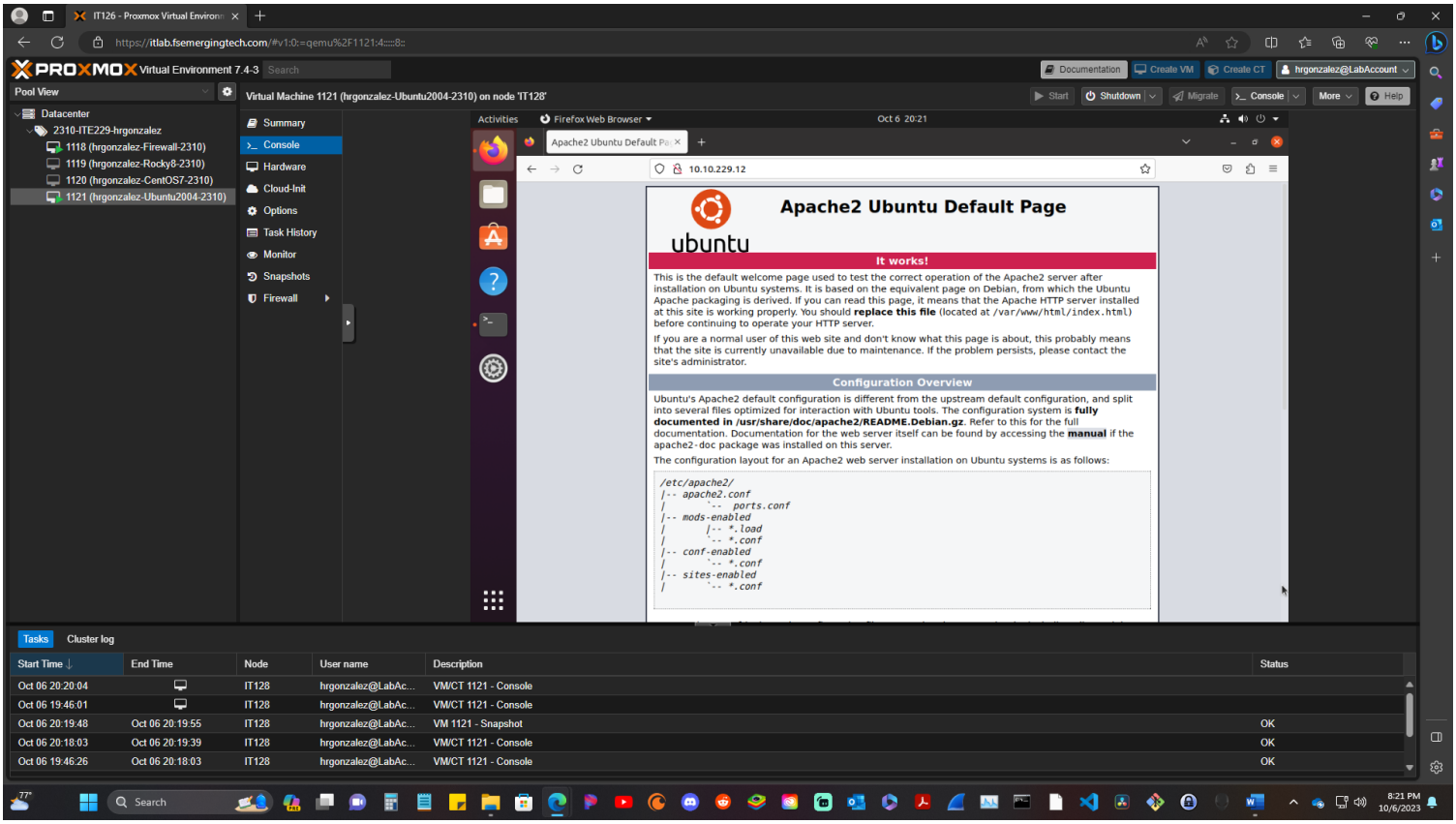
Open Firewall Ports 80 and 443

Enter command prompt sudo ufw allow in “Apache Full” to allow TCP/80 and TCP/443 in, which are communication ports needed. This command will apply a firewall profile for Apache.



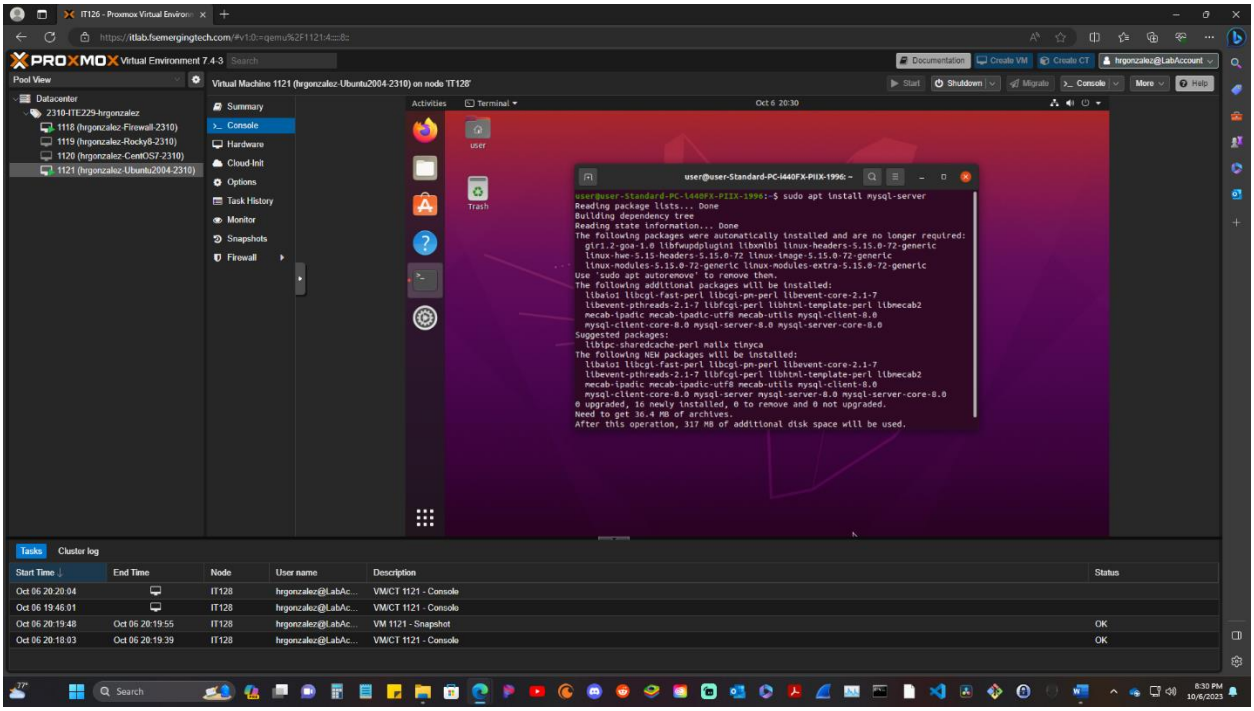
Browse to Apache2 Ubuntu Default Page

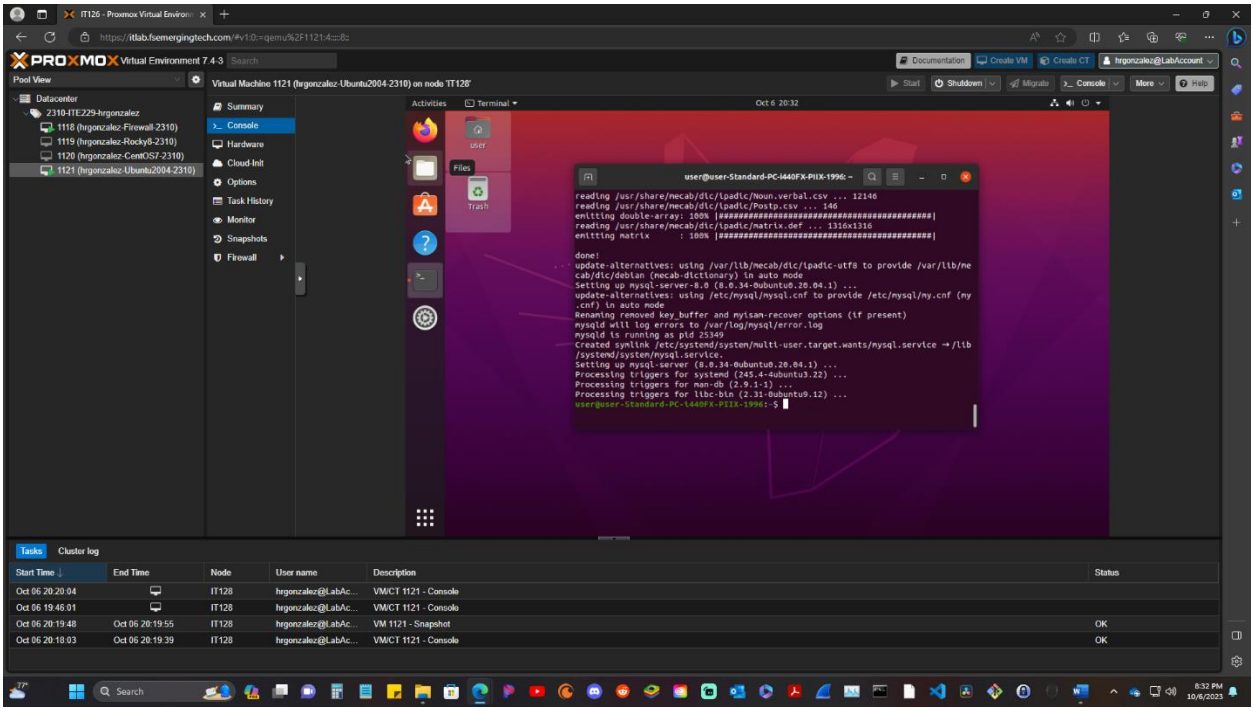
Next open up fire fox web browser in Ubuntu and browse to http://10.10.229.12 and make sure your installation was succesfull. You should see a page like the one below.



Install MySQL

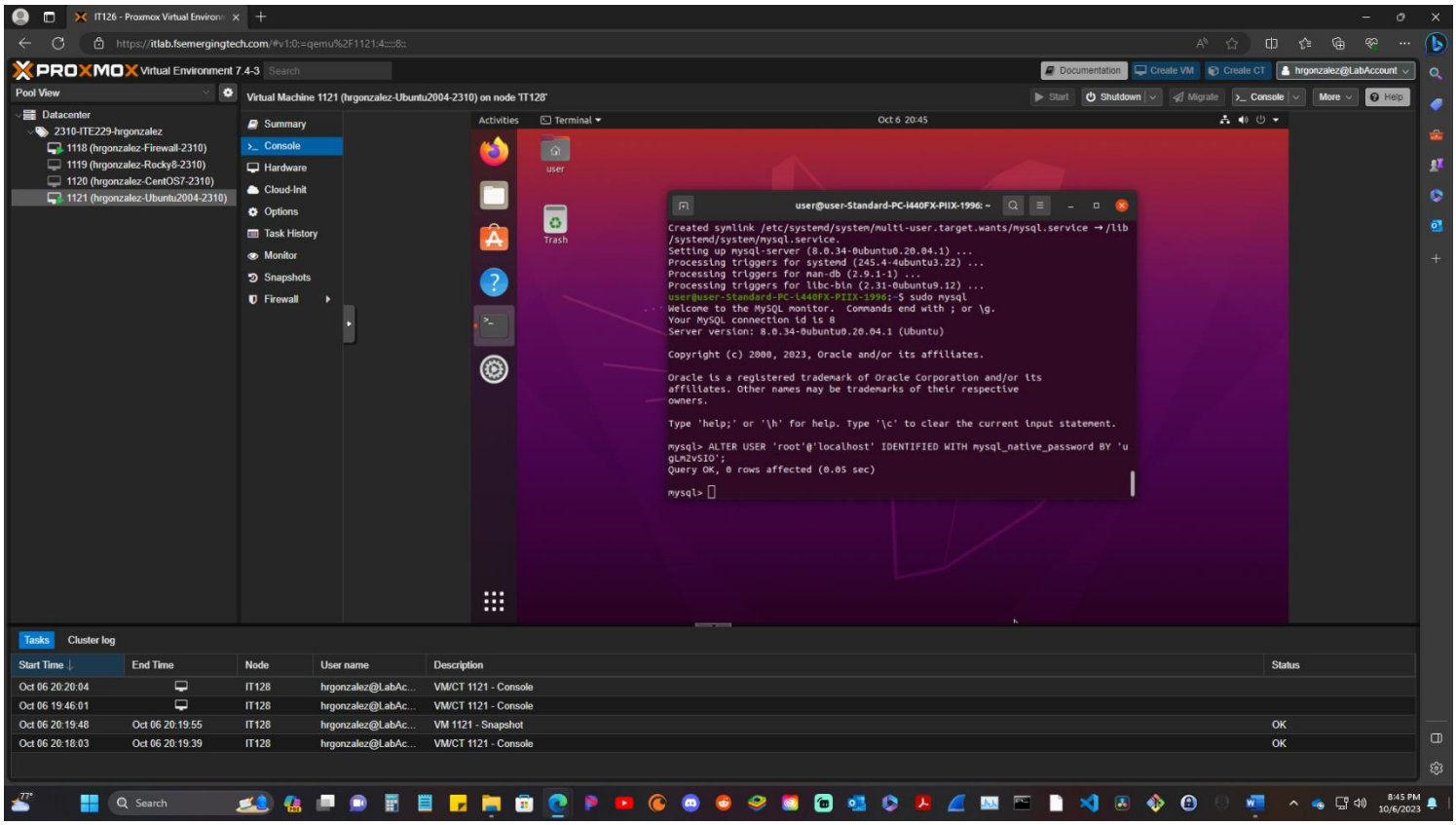
Now we need to install MySQL. Input command sudo apt install mysql-server. When asked to continue enter Y for yes, and then wait until install is completed.





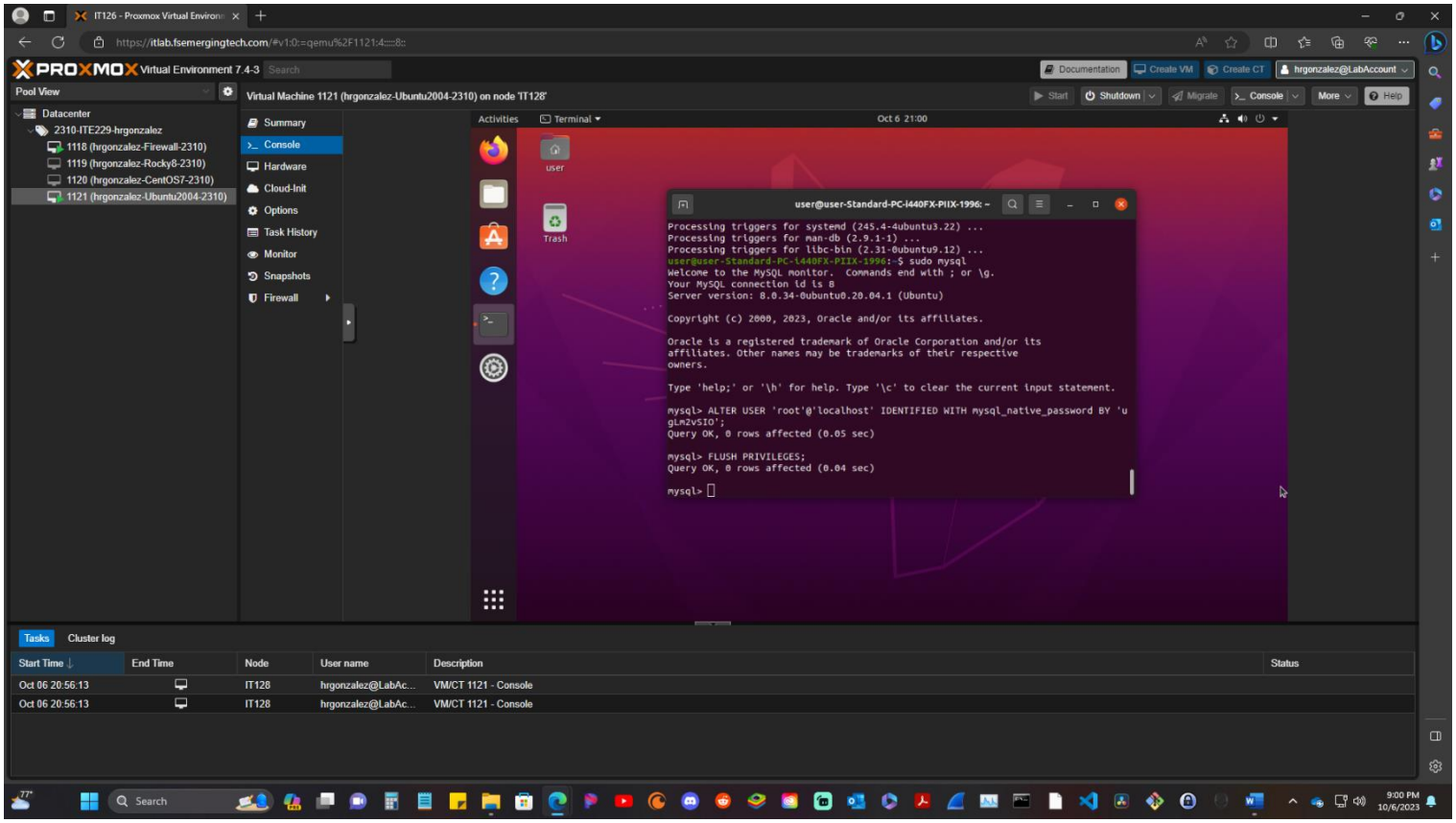
Alter root user password

Next we need to change our MySQL password, but before we do that you can use a password generator or make your own password (be sure to save this password somewhere safe this is the password we are going to use for MySQL). Enter command line sudo mysql. This will bring you to the MySQL command line. Here we are going to enter command line ALTER USER 'root'@localhost' IDENTIFIED WITH mysql_native_password BY 'newpasswordhere'; then hit enter. If input correctly you should see Query Ok.



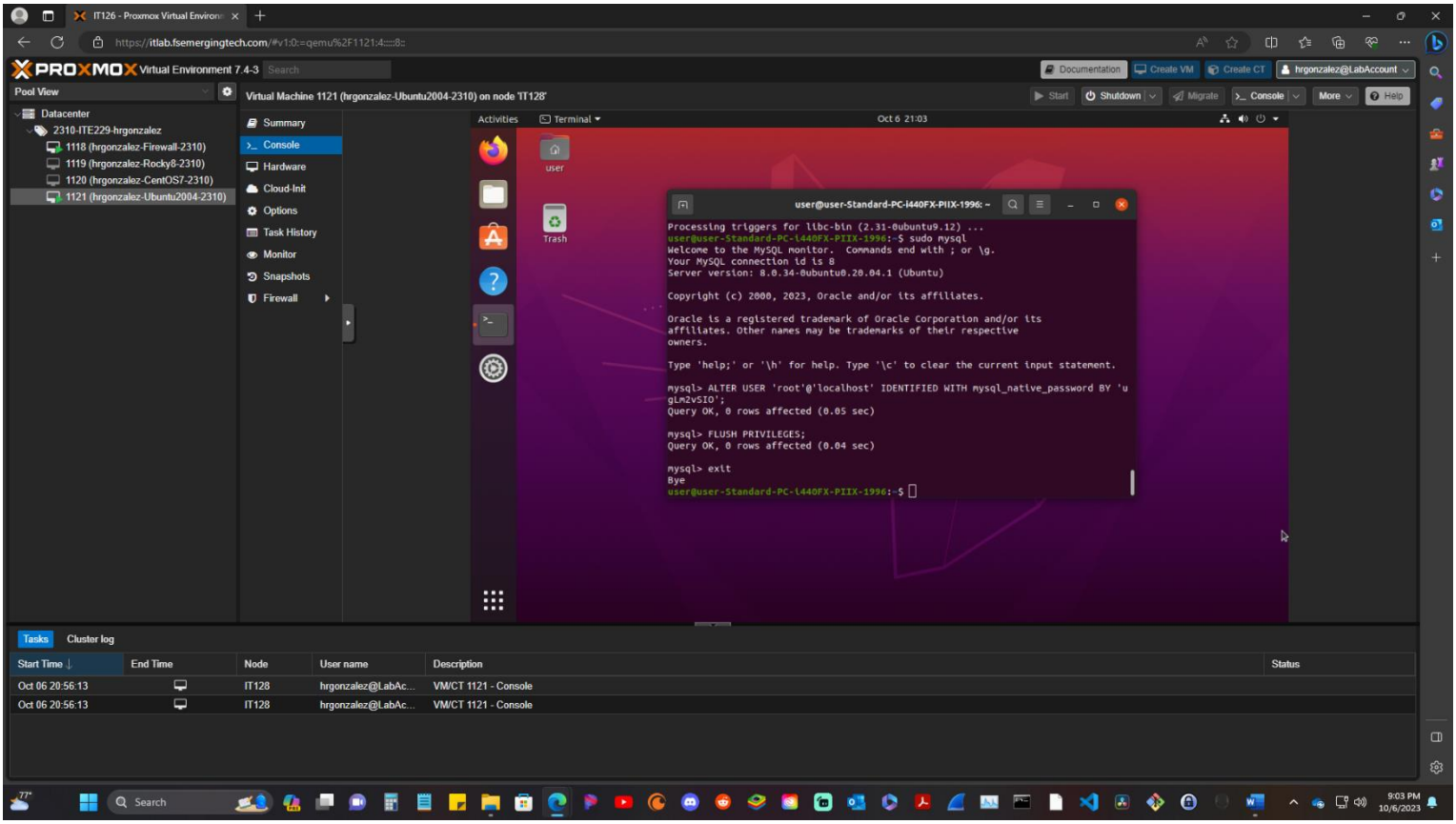
Flush Privileges

After we need to flush to reload the settings. Enter command line FLUSH PRIVILEGES; then hit enter. If prompt correctly you will see Query ok.



Quit MySQL

Now we need to exit out of My SQL. Enter command line exit. If correct it will redirect you to your username user@user. *Make sure to document password used for MySQL.

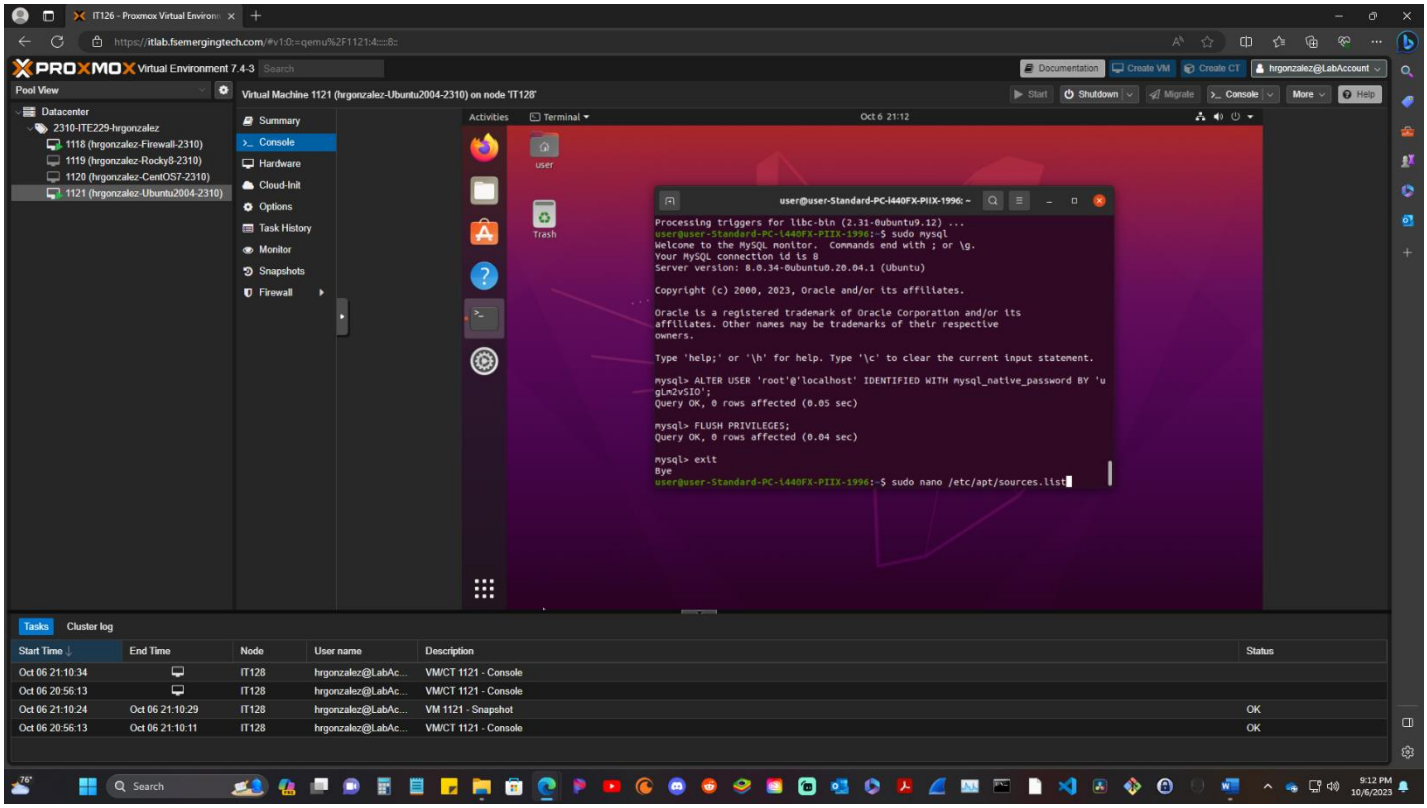


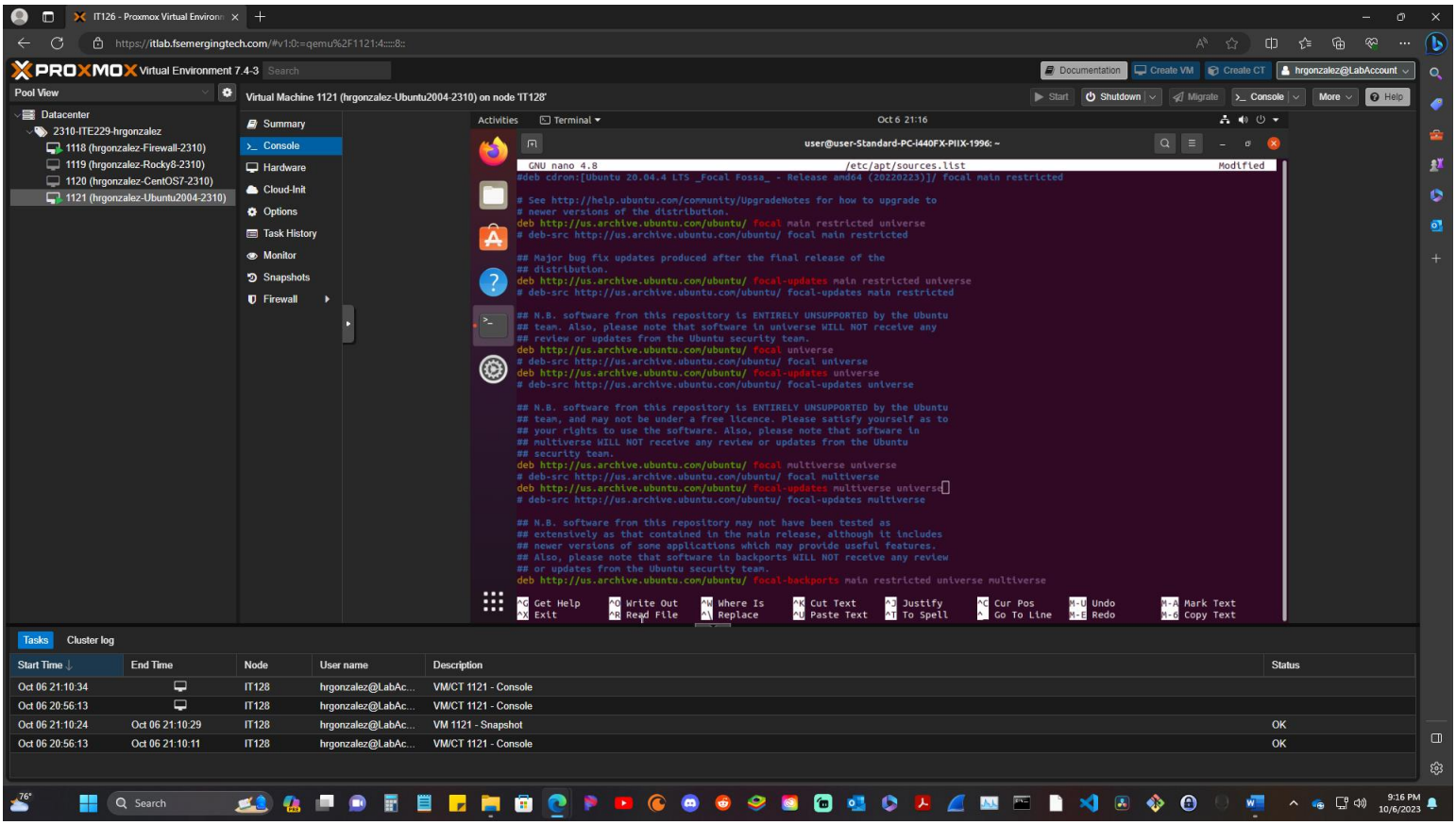
Install PHP

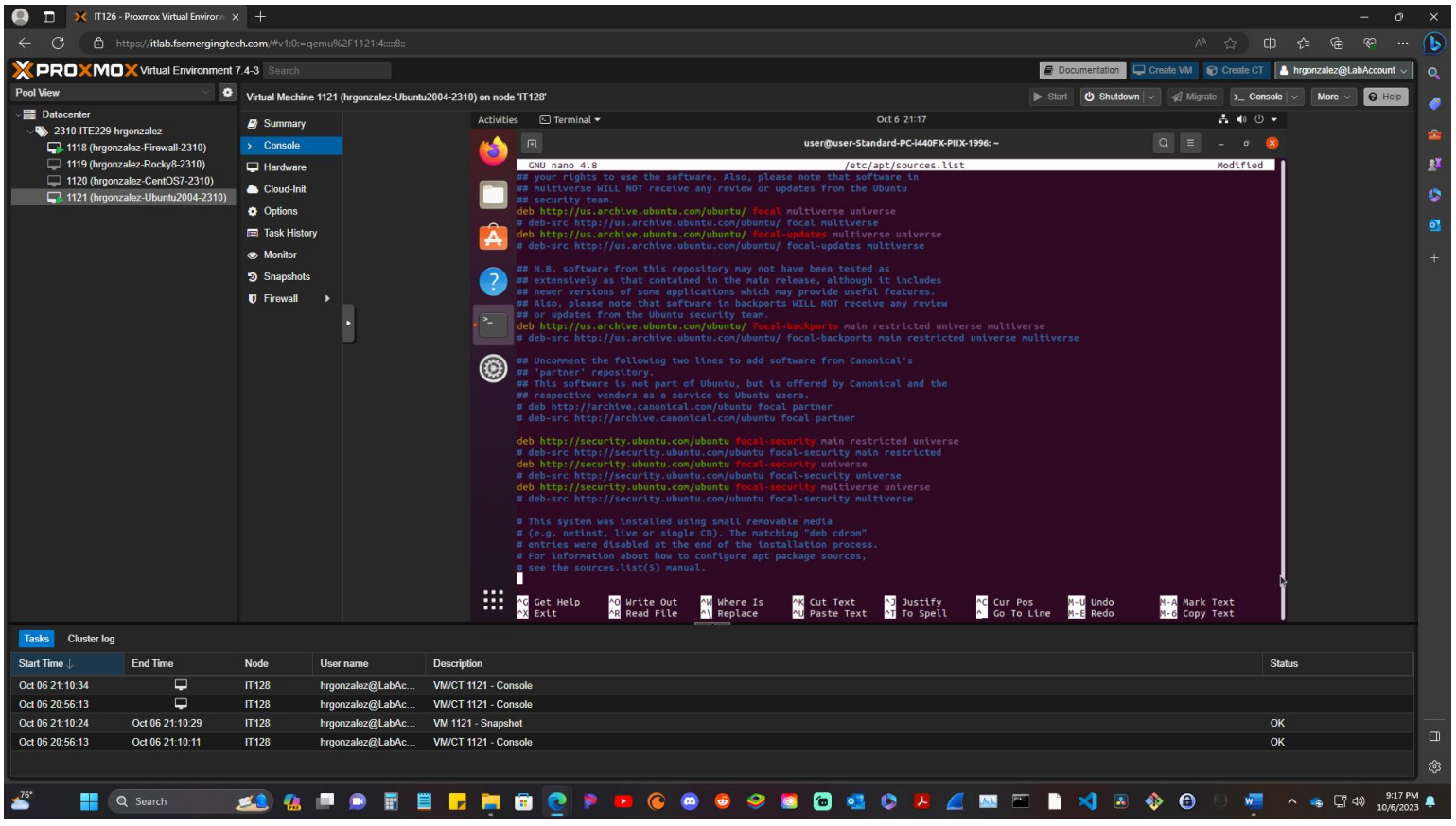
Edit Sources.list File

Add “universe” to all URLs

We need to enable Ubuntu universe repositories, because some of the PHP packages for WordPress require it. In the terminal enter command line sudo nano /etc/apt/sources.list. We need to add universe to all the URLs. Follow picture below and mimic it. After hit control key, save changes, and then hit enter. It should take you back to the username user@user if done correctly.

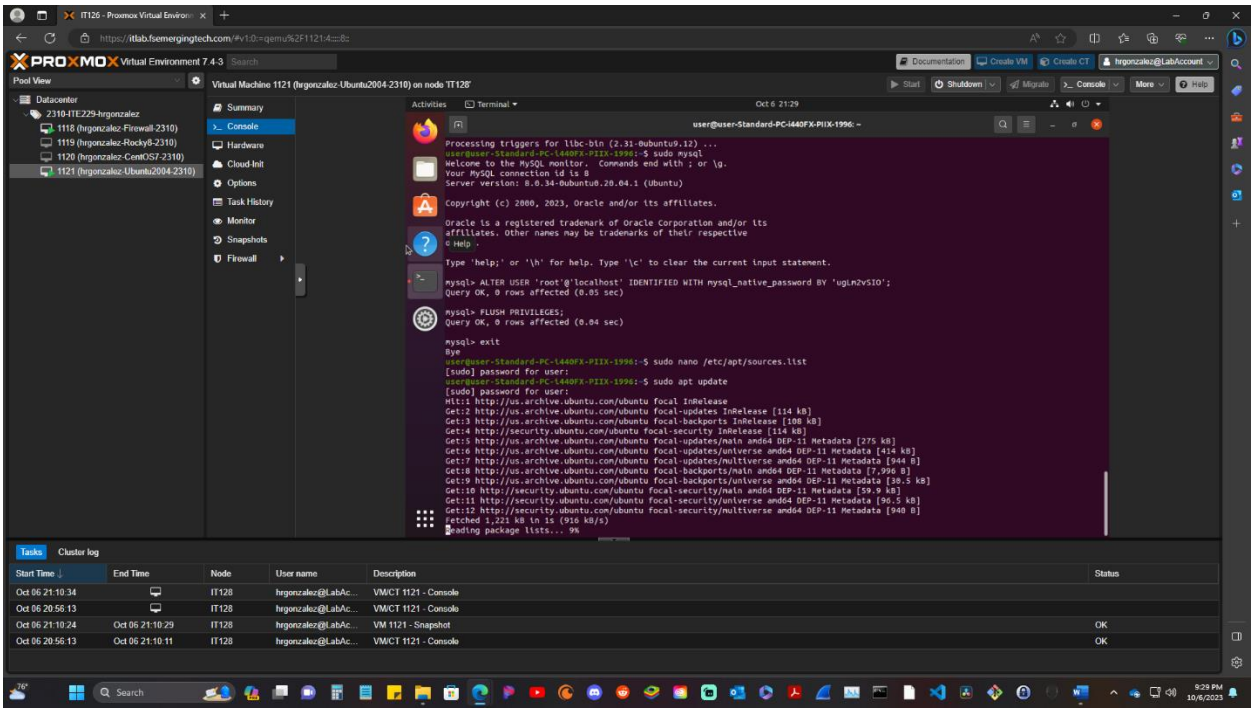


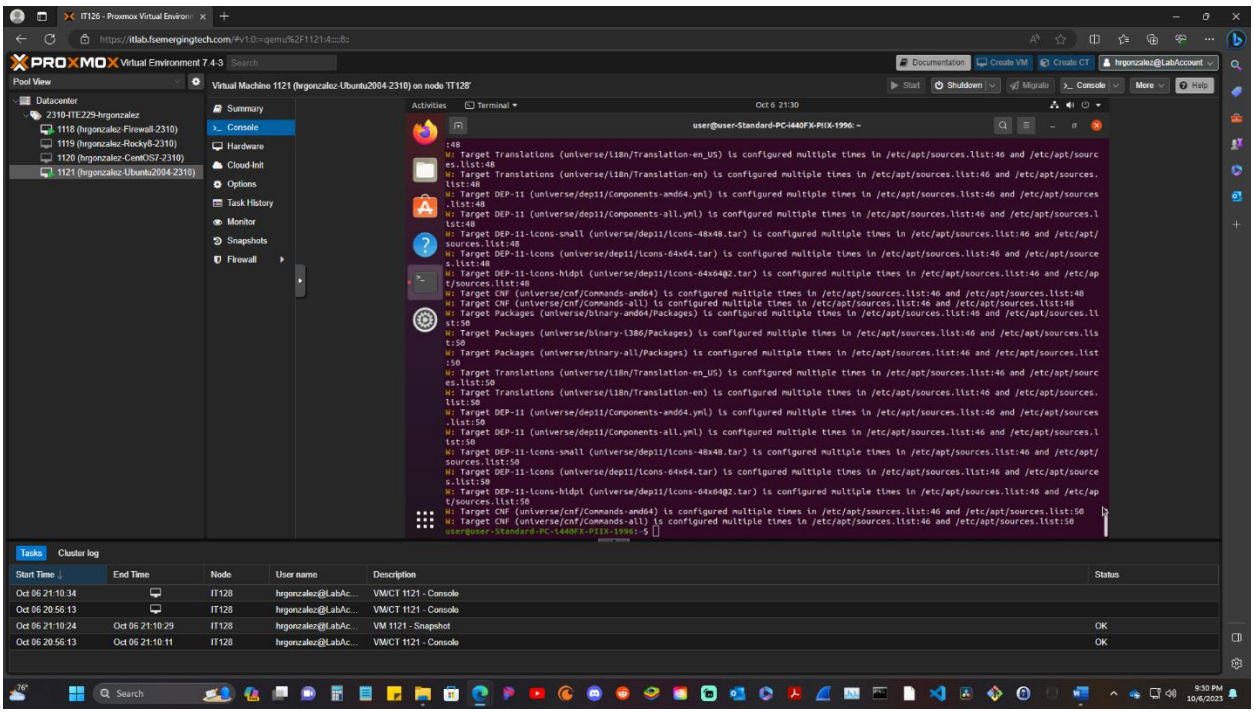

IT126 - Proxmox Virtual Environ
https://itlab.fsemergingtech.com/#v1:0:=qemu%2F1121:4::::8::
PROXMOX Virtual Environment 7.4-3
Search
Documentation
Create VM
Create CT
hrgonzalez@LabAccount
Pool View
Virtual Machine 1121 (hrgonzalez-Ubuntu2004-2310) on node 'IT128'
Start
Shutdown
Migrate
Console
More
Help
Datacenter
2310-ITE229-hrgonzalez
1118 (hrgonzalez-Firewall-2310)
1119 (hrgonzalez-Rocky8-2310)
1120 (hrgonzalez-CentOS7-2310)
1121 (hrgonzalez-Ubuntu2004-2310)
Summary
Console
Hardware
Cloud-Init
Options
Task History
Monitor
Snapshots
Firewall
Activities
Terminal
Oct 6 21:16
user@user-Standard-PC-i440FX-PIIX-1996: ~
GNU nano 4.8
/etc/apt/sources.list
Modified
#deb cdrom:[Ubuntu 20.04.4 LTS _Focal Fossa_ - Release amd64 (20220223)]/ focal main restricted
See http://help.ubuntu.com/community/UpgradeNotes for how to upgrade to
newer versions of the distribution.
deb http://us.archive.ubuntu.com/ubuntu/ focal main restricted universe
deb-src http://us.archive.ubuntu.com/ubuntu/ focal main restricted
Major bug fix updates produced after the final release of the
distribution.
deb http://us.archive.ubuntu.com/ubuntu/ focal-updates main restricted universe
deb-src http://us.archive.ubuntu.com/ubuntu/ focal-updates main restricted
N.B. software from this repository is ENTIRELY UNSUPPORTED by the Ubuntu
team. Also, please note that software in universe WILL NOT receive any
review or updates from the Ubuntu security team.
deb http://us.archive.ubuntu.com/ubuntu/ focal universe
deb-src http://us.archive.ubuntu.com/ubuntu/ focal universe
deb http://us.archive.ubuntu.com/ubuntu/ focal-updates universe
deb-src http://us.archive.ubuntu.com/ubuntu/ focal-updates universe
N.B. software from this repository is ENTIRELY UNSUPPORTED by the Ubuntu
team, and may not be under a free licence. Please satisfy yourself as to
your rights to use the software. Also, please note that software in
multiverse WILL NOT receive any review or updates from the Ubuntu
security team.
deb http://us.archive.ubuntu.com/ubuntu/ focal multiverse universe
deb-src http://us.archive.ubuntu.com/ubuntu/ focal multiverse
deb http://us.archive.ubuntu.com/ubuntu/ focal-updates multiverse universe
deb-src http://us.archive.ubuntu.com/ubuntu/ focal-updates multiverse
N.B. software from this repository may not have been tested as
extensively as that contained in the main release, although it includes
newer versions of some applications which may provide useful features.
Also, please note that software in backports WILL NOT receive any review
or updates from the Ubuntu security team.
deb http://us.archive.ubuntu.com/ubuntu/ focal-backports main restricted universe multiverse
^G Get Help
^O Write Out
^W Where Is
^K Cut Text
^J Justify
^C Cur Pos
M-U Undo
M-A Mark Text
^X Exit
^R Read File
^\ Replace
^U Paste Text
^T To Spell
^_ Go To Line
M-E Redo
M-6 Copy Text
Tasks
Cluster log
Start Time
End Time
Node
User name
Description
Status
Oct 06 21:10:34
IT128
hrgonzalez@LabAc...
VM/CT 1121 - Console
Oct 06 20:56:13
IT128
hrgonzalez@LabAc...
VM/CT 1121 - Console
Oct 06 21:10:24
Oct 06 21:10:29
IT128
hrgonzalez@LabAc...
VM 1121 - Snapshot
OK
Oct 06 20:56:13
Oct 06 21:10:11
IT128
hrgonzalez@LabAc...
VM/CT 1121 - Console
OK
Search
9:16 PM
10/6/2023


IT126 - Proxmox Virtual Environ
https://itlab.fsemergingtech.com/#v1:0:=qemu%2F1121:4::::8::
PROXMOX Virtual Environment 7.4-3
Search
Documentation
Create VM
Create CT
hrgonzalez@LabAccount
Pool View
Virtual Machine 1121 (hrgonzalez-Ubuntu2004-2310) on node 'IT128'
Start
Shutdown
Migrate
Console
More
Help
Datacenter
2310-ITE229-hrgonzalez
1118 (hrgonzalez-Firewall-2310)
1119 (hrgonzalez-Rocky8-2310)
1120 (hrgonzalez-CentOS7-2310)
1121 (hrgonzalez-Ubuntu2004-2310)
Summary
Console
Hardware
Cloud-Init
Options
Task History
Monitor
Snapshots
Firewall
Activities
Terminal
Oct 6 21:17
user@user-Standard-PC-i440FX-PIIX-1996: ~
GNU nano 4.8
/etc/apt/sources.list
Modified
your rights to use the software. Also, please note that software in
multiverse WILL NOT receive any review or updates from the Ubuntu
security team.
deb http://us.archive.ubuntu.com/ubuntu/ focal multiverse universe
deb-src http://us.archive.ubuntu.com/ubuntu/ focal multiverse
deb http://us.archive.ubuntu.com/ubuntu/ focal-updates multiverse universe
deb-src http://us.archive.ubuntu.com/ubuntu/ focal-updates multiverse
N.B. software from this repository may not have been tested as
extensively as that contained in the main release, although it includes
newer versions of some applications which may provide useful features.
Also, please note that software in backports WILL NOT receive any review
or updates from the Ubuntu security team.
deb http://us.archive.ubuntu.com/ubuntu/ focal-backports main restricted universe multiverse
deb-src http://us.archive.ubuntu.com/ubuntu/ focal-backports main restricted universe multiverse
Uncomment the following two lines to add software from Canonical's
'partner' repository.
This software is not part of Ubuntu, but is offered by Canonical and the
respective vendors as a service to Ubuntu users.
deb http://archive.canonical.com/ubuntu focal partner
deb-src http://archive.canonical.com/ubuntu focal partner
deb http://security.ubuntu.com/ubuntu focal-security main restricted universe
deb-src http://security.ubuntu.com/ubuntu focal-security main restricted
deb http://security.ubuntu.com/ubuntu focal-security universe
deb-src http://security.ubuntu.com/ubuntu focal-security universe
deb http://security.ubuntu.com/ubuntu focal-security multiverse universe
deb-src http://security.ubuntu.com/ubuntu focal-security multiverse
This system was installed using small removable media
(e.g. netinst, live or single CD). The matching "deb cdrom"
entries were disabled at the end of the installation process.
For information about how to configure apt package sources,
see the sources.list(5) manual.
^G Get Help
^O Write Out
^W Where Is
^K Cut Text
^J Justify
^C Cur Pos
M-U Undo
M-A Mark Text
^X Exit
^R Read File
^\ Replace
^U Paste Text
^T To Spell
^_ Go To Line
M-E Redo
M-6 Copy Text
Tasks
Cluster log
Start Time
End Time
Node
User name
Description
Status
Oct 06 21:10:34
IT128
hrgonzalez@LabAc...
VM/CT 1121 - Console
Oct 06 20:56:13
IT128
hrgonzalez@LabAc...
VM/CT 1121 - Console
Oct 06 21:10:24
Oct 06 21:10:29
IT128
hrgonzalez@LabAc...
VM 1121 - Snapshot
OK
Oct 06 20:56:13
Oct 06 21:10:11
IT128
hrgonzalez@LabAc...
VM/CT 1121 - Console
OK
Search
9:17 PM
10/6/2023

Update Ubuntu (refreshes the repolist)

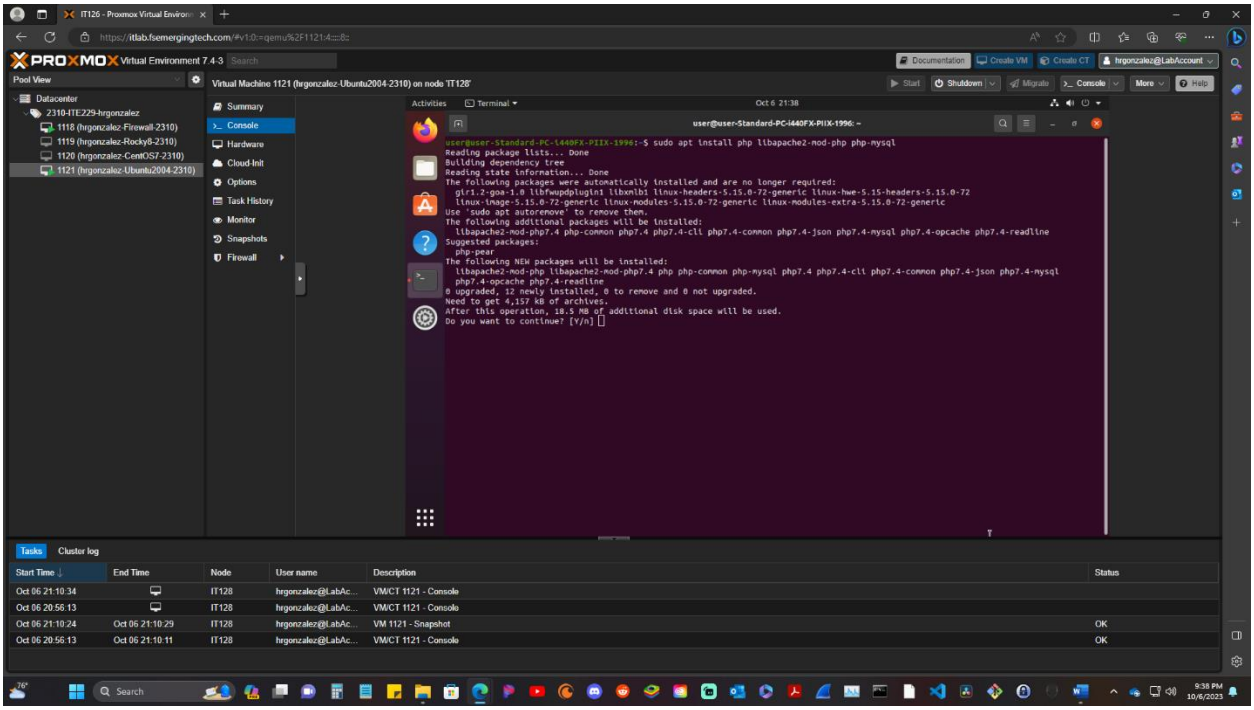
Next, we need to update/refresh/enable repolist that we just changed. To do this enter command line sudo apt update and wait until it completes.

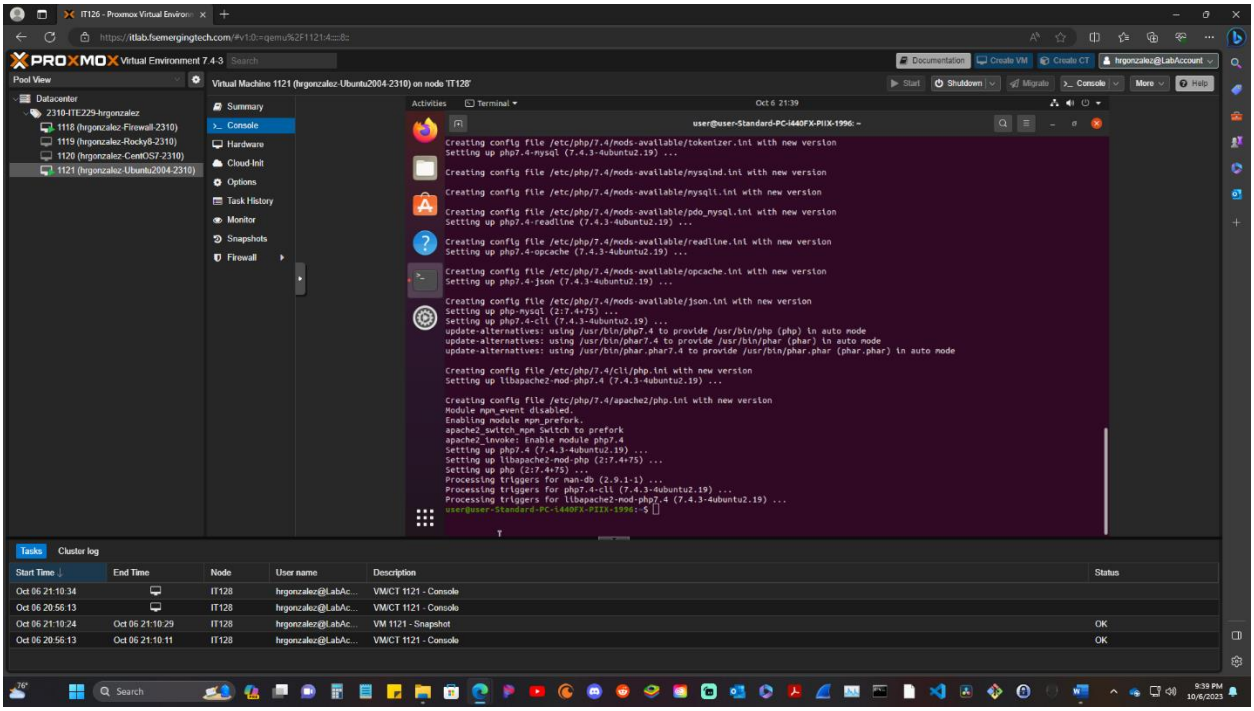




Install Required PHP Libraries

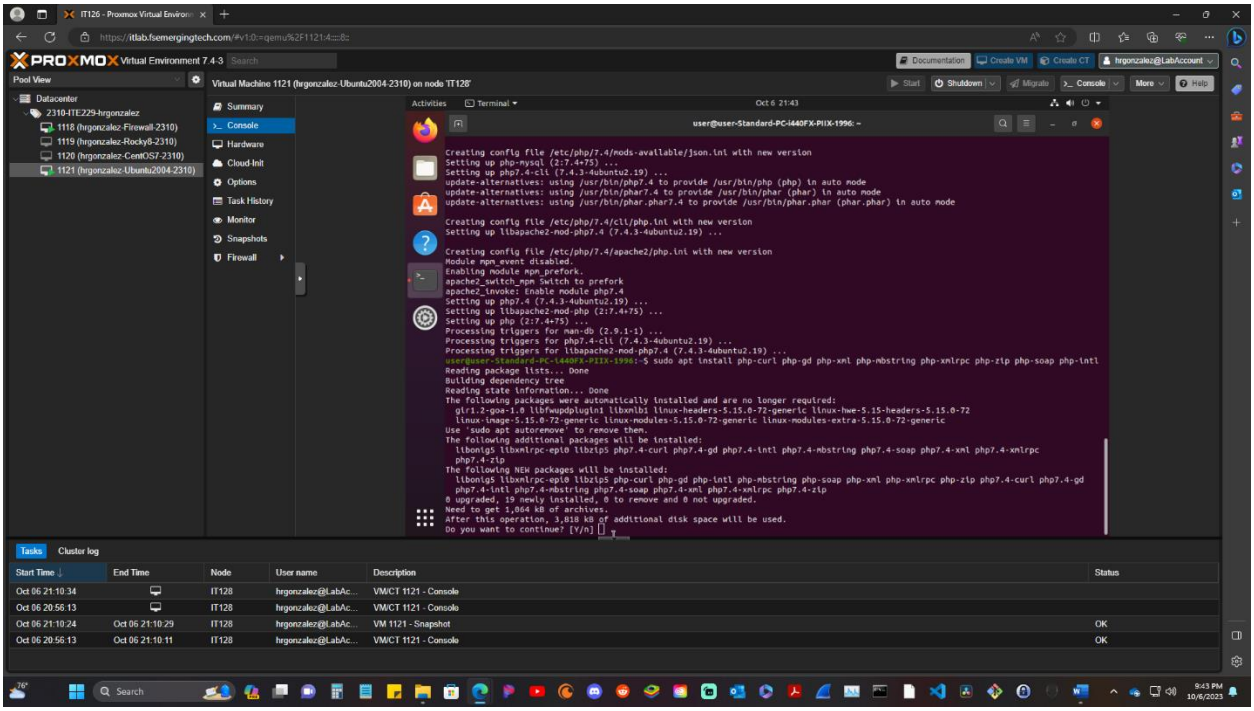
To do this enter command line sudo apt install php libapache2-mod-php php-mysql. Then input Y and hit enter to continue. Wait until it finish installs. This will install PHP base libraries.

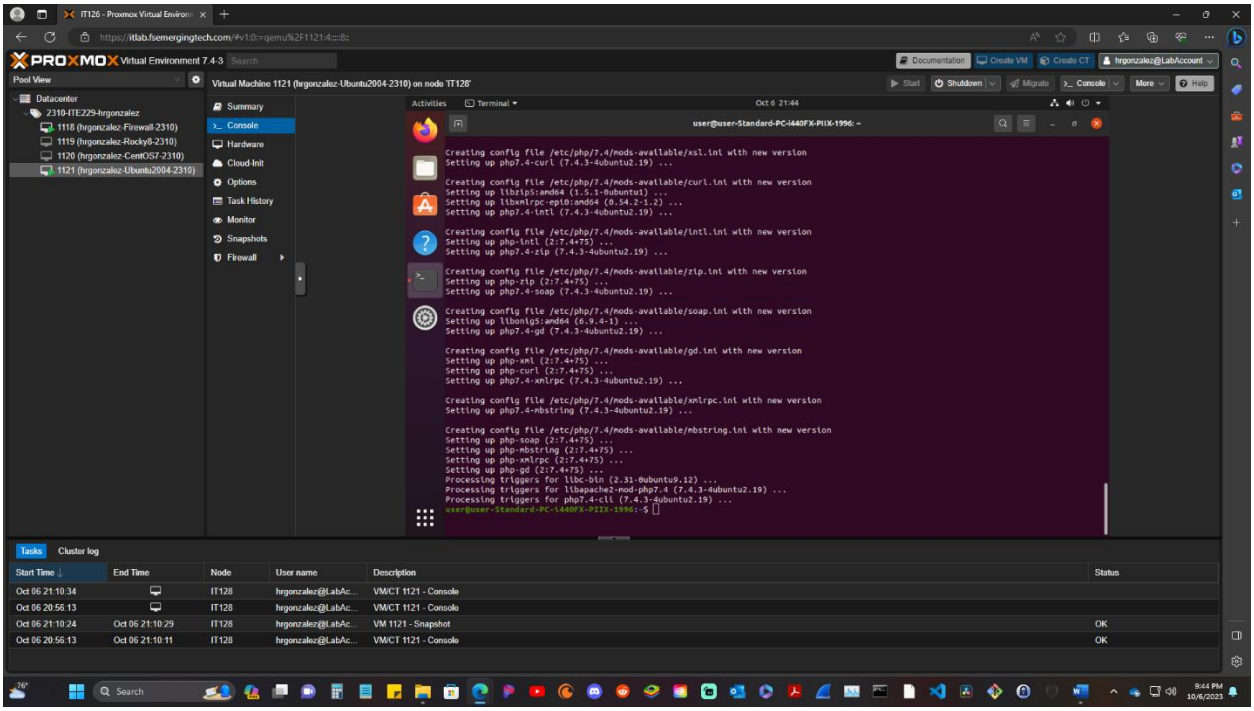




Install Required MySQL Libraries

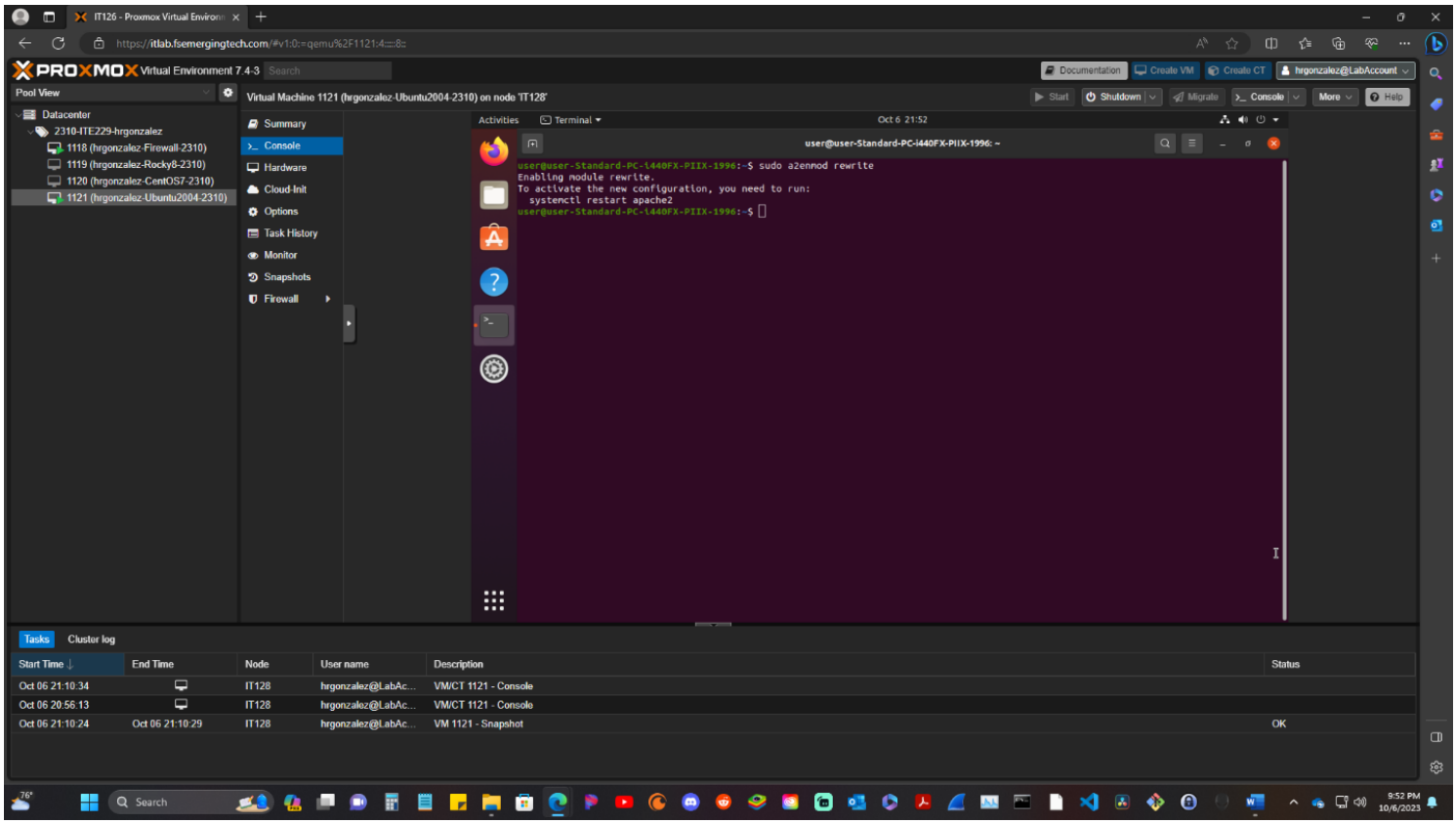
Now we need to install MySQL libraries required by Wordpress. Enter command line sudo apt install php-curl php-gd php-xml php-mbstring php-xmlrcp php-zip php-soap php-intl then hit enter. After input Y and hit enter to continue and wait for completion of installed.





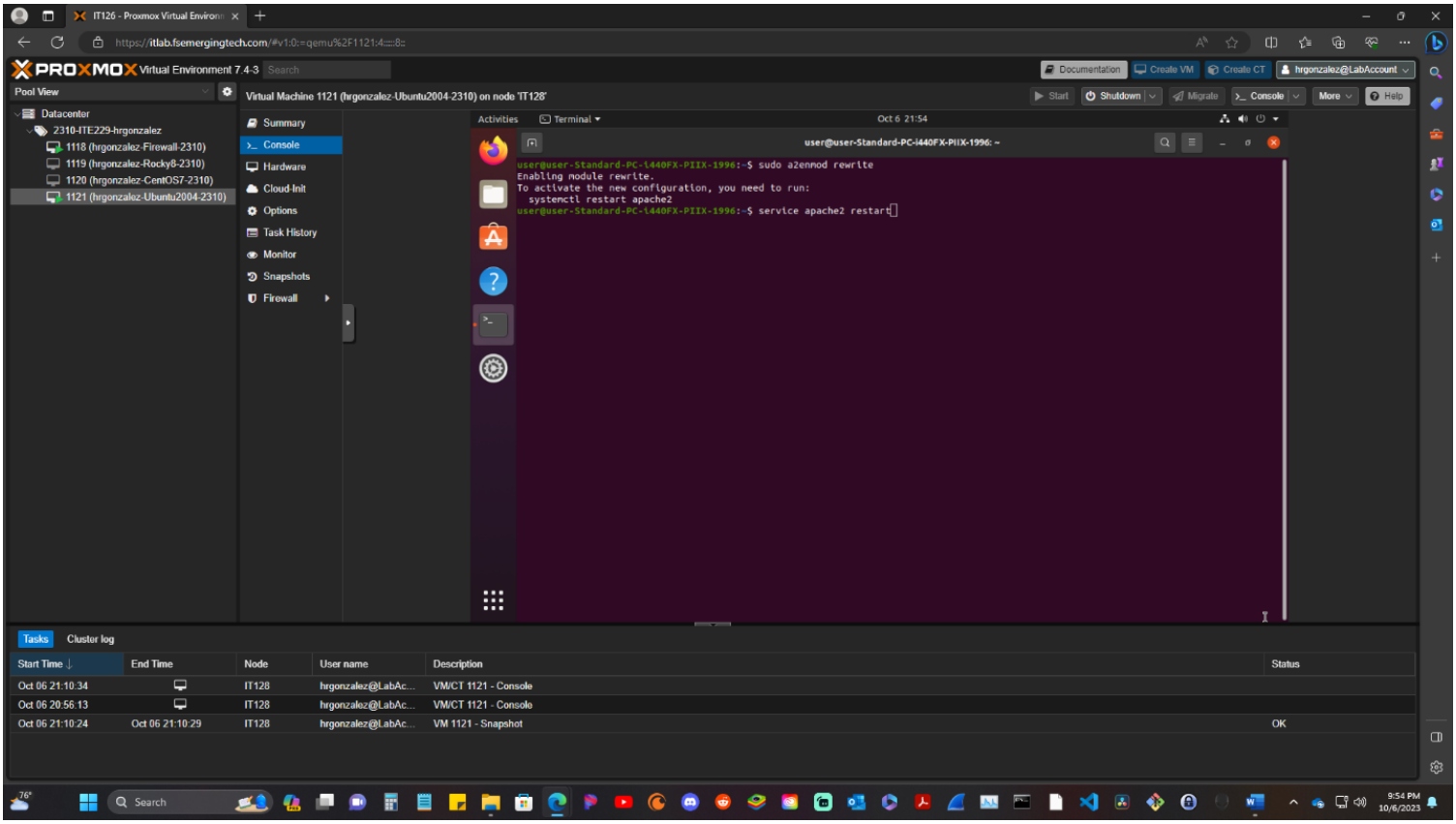
Enable URL Rewrites (clean URLs)

Then we need to enable URL rewrites which gives you the ability to enter clean URL without those long outputs that shows questions marks, numbers, etc. To do this we need to enter command line sudo a2enmod rewrite.



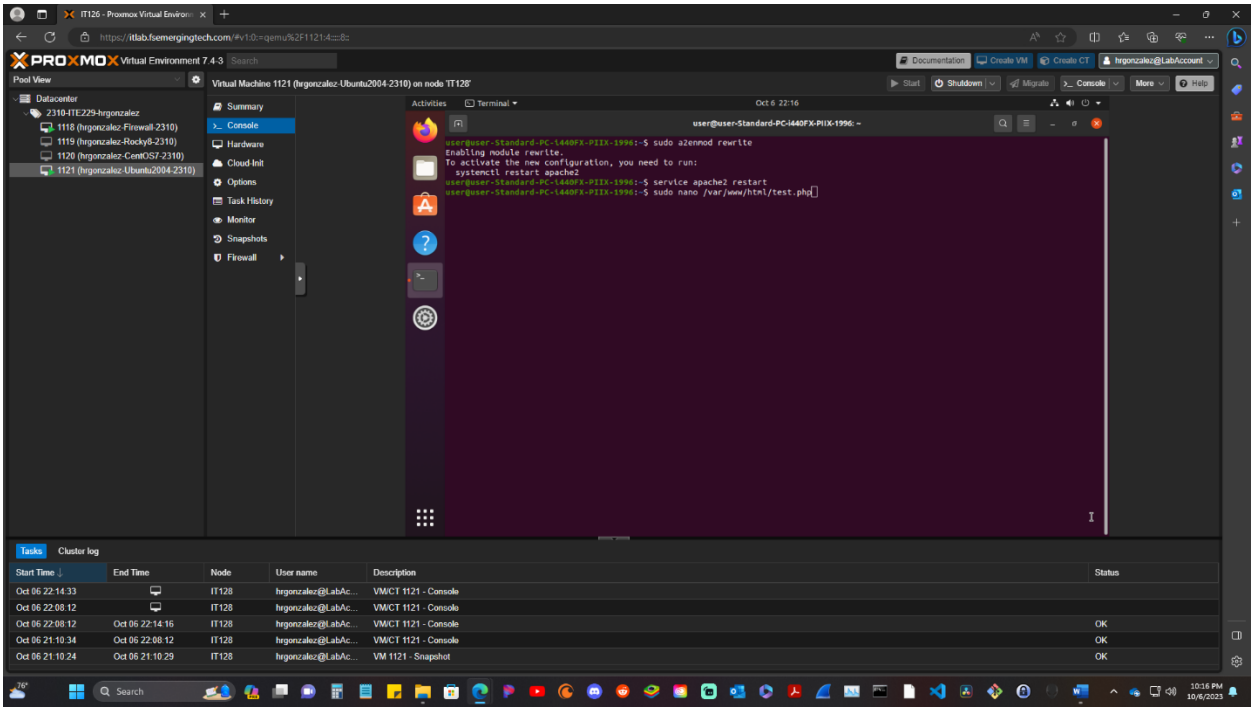
Restart Apache Service

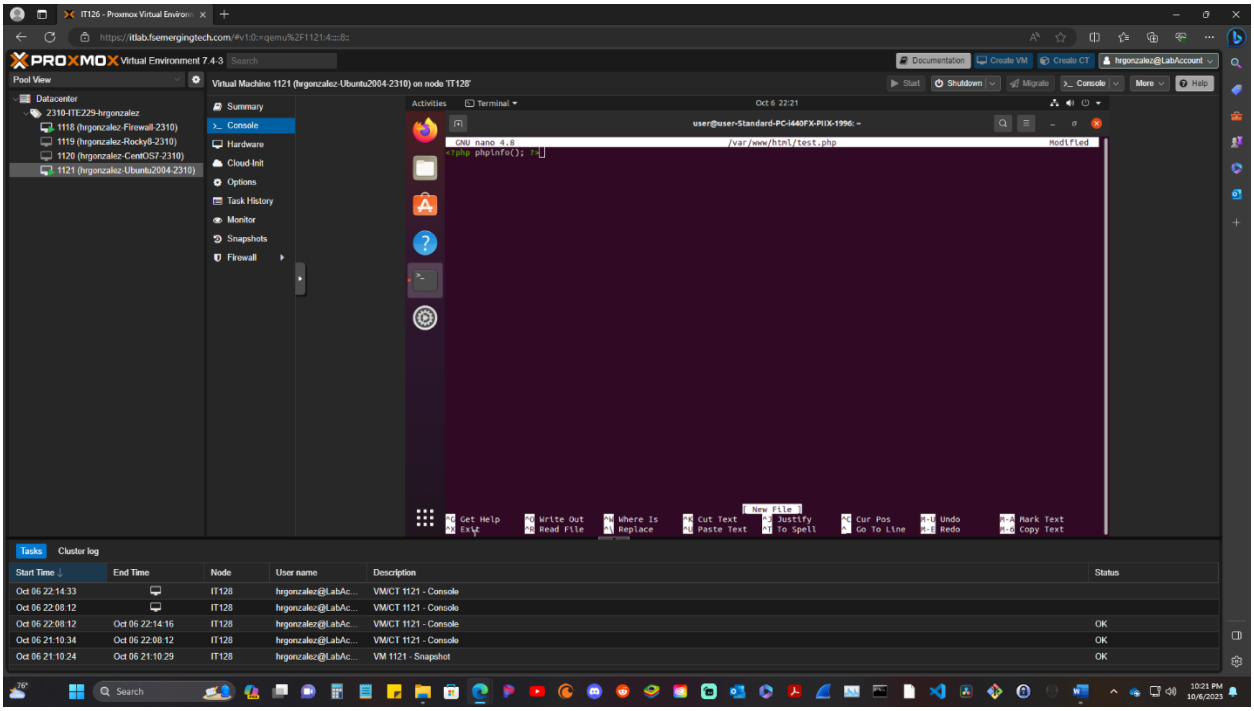
Now we need to restart the Aache service. We can do this 2 ways. One we can enter command line sudo systemctl restart apache2 or we can use command line service apache2 restart. We are going to use command line service apache2 restart. Enter command line service apache2 restart. In the pop up window enter your username password then hit authenticate. It should bring you back to the terminal window with a new command line ready to be entered.



Create a test.php Web Page

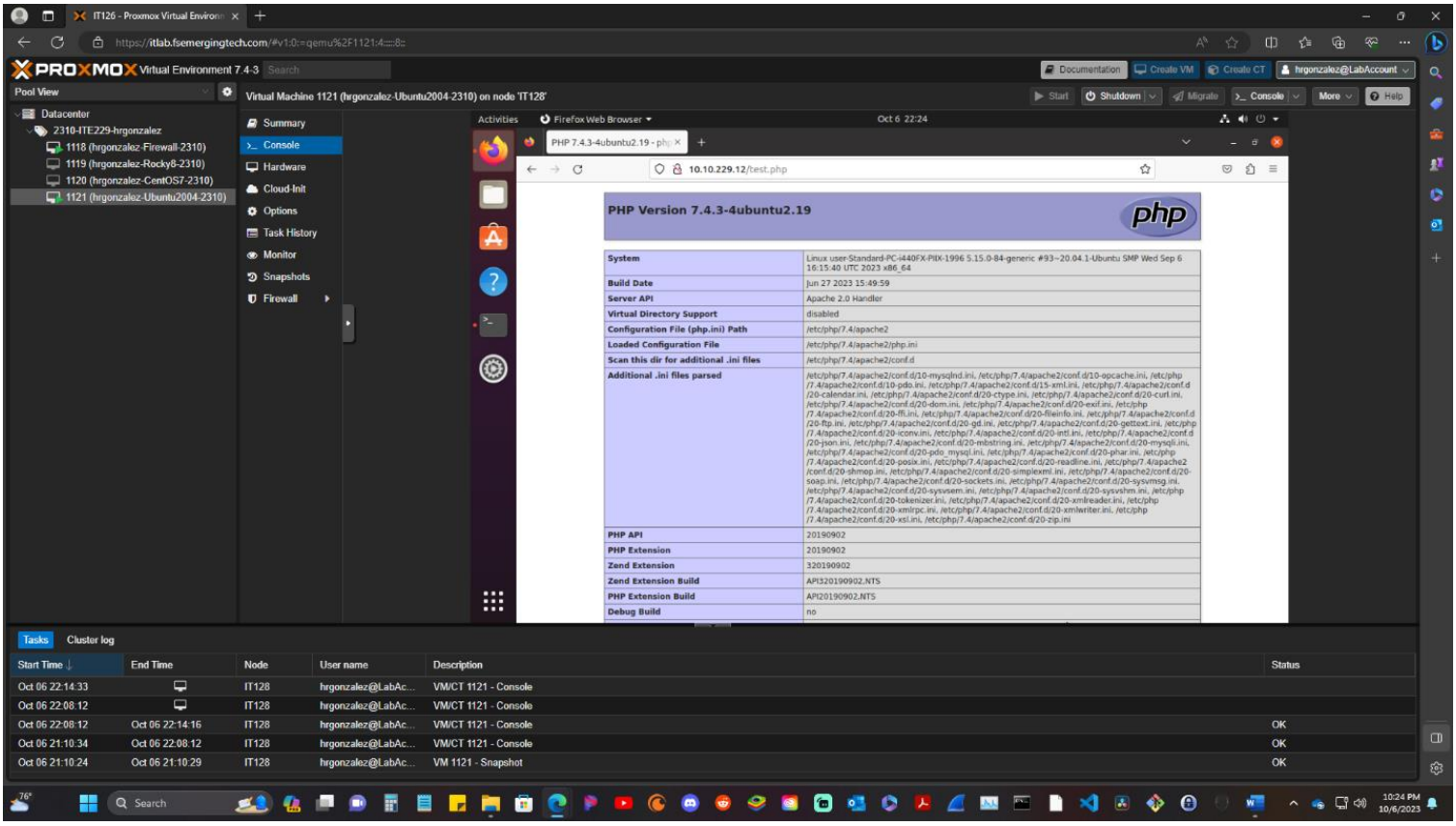
Now we need to create a test.php page. In order to do this you need to enter command line sudo nano /var/www/html/test.php. This will open a blank nano page. Here we are going to write <?php phpinfo(); ?>, which is basically saying to pull up php information page. After exit bvy hitting control x to exit, then save file and hit enter. This should take you back to your username line.





Test the test.php Web Page

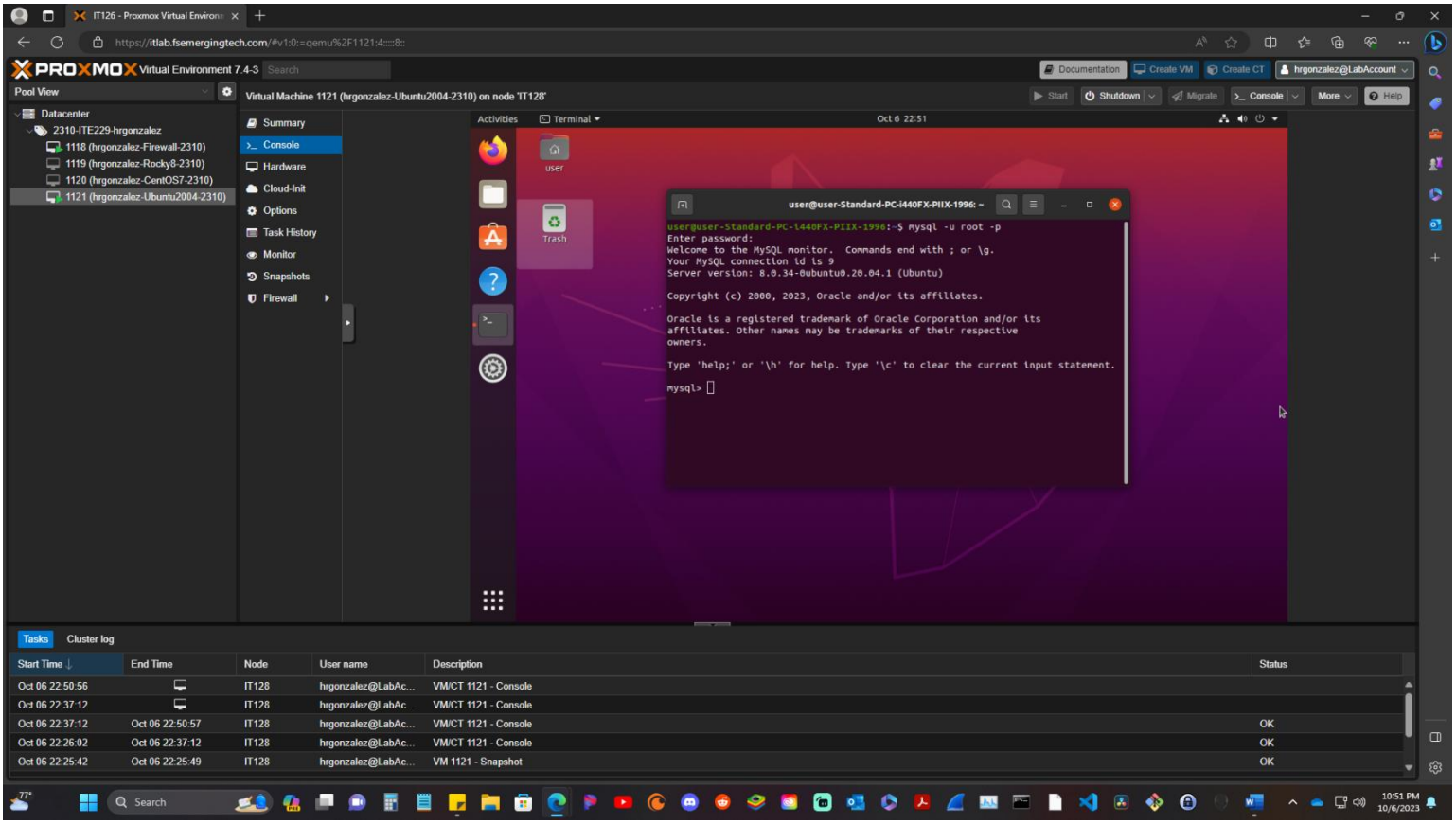
Now we need to test the page. Open up firefox web browser in your Ubuntu. Enter 10.10.229.12/test.php on the url. You should see a picture like the one below if installed and done correctly. This is the php information page.



Database Configuration in MySQL

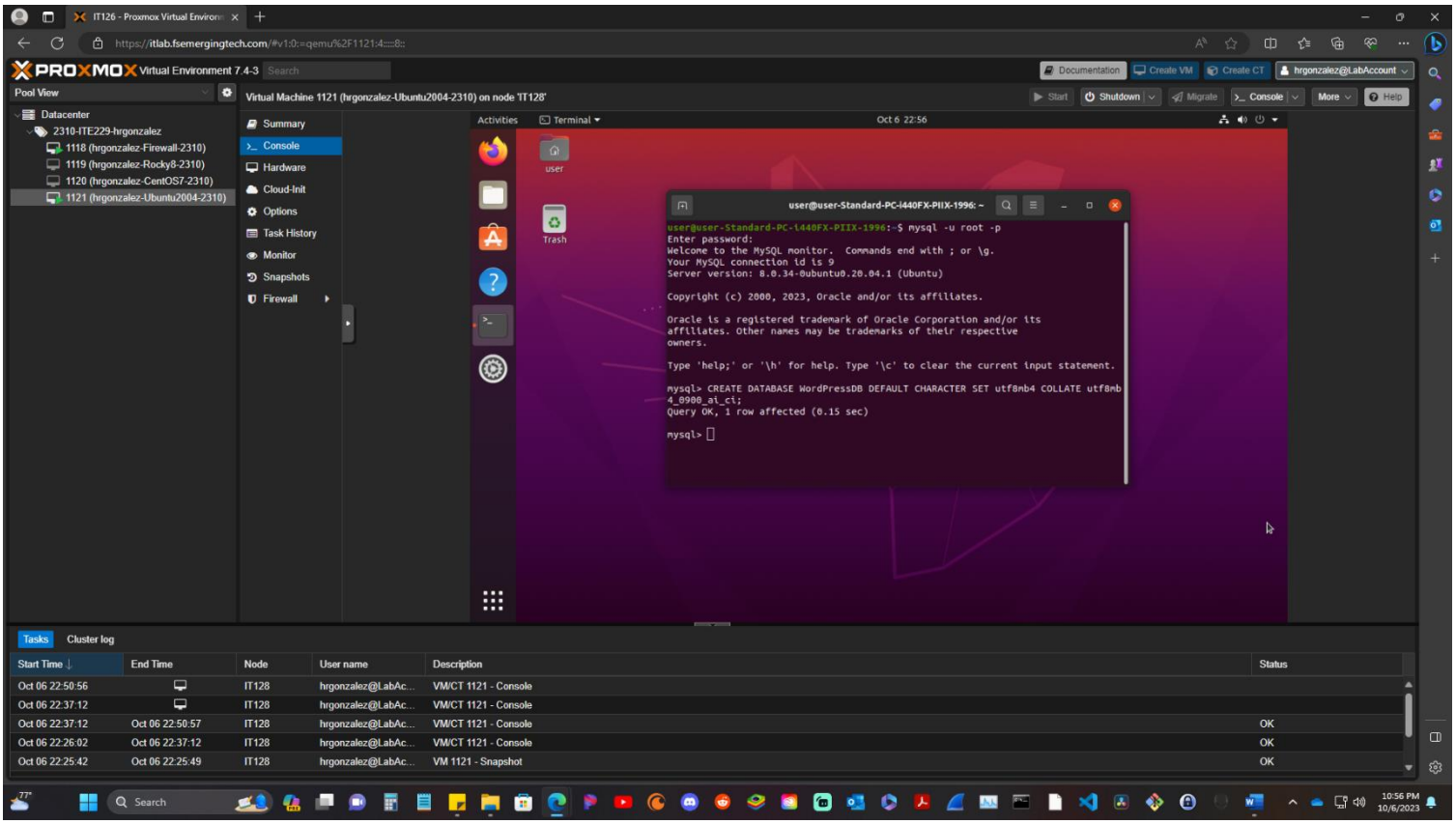
Log into MySQL Database

We need to log in to MySQL CLI. To do this enter command line mysql -u root -p. Then enter the password that was generated in previous step. This should take you to MySQL database.



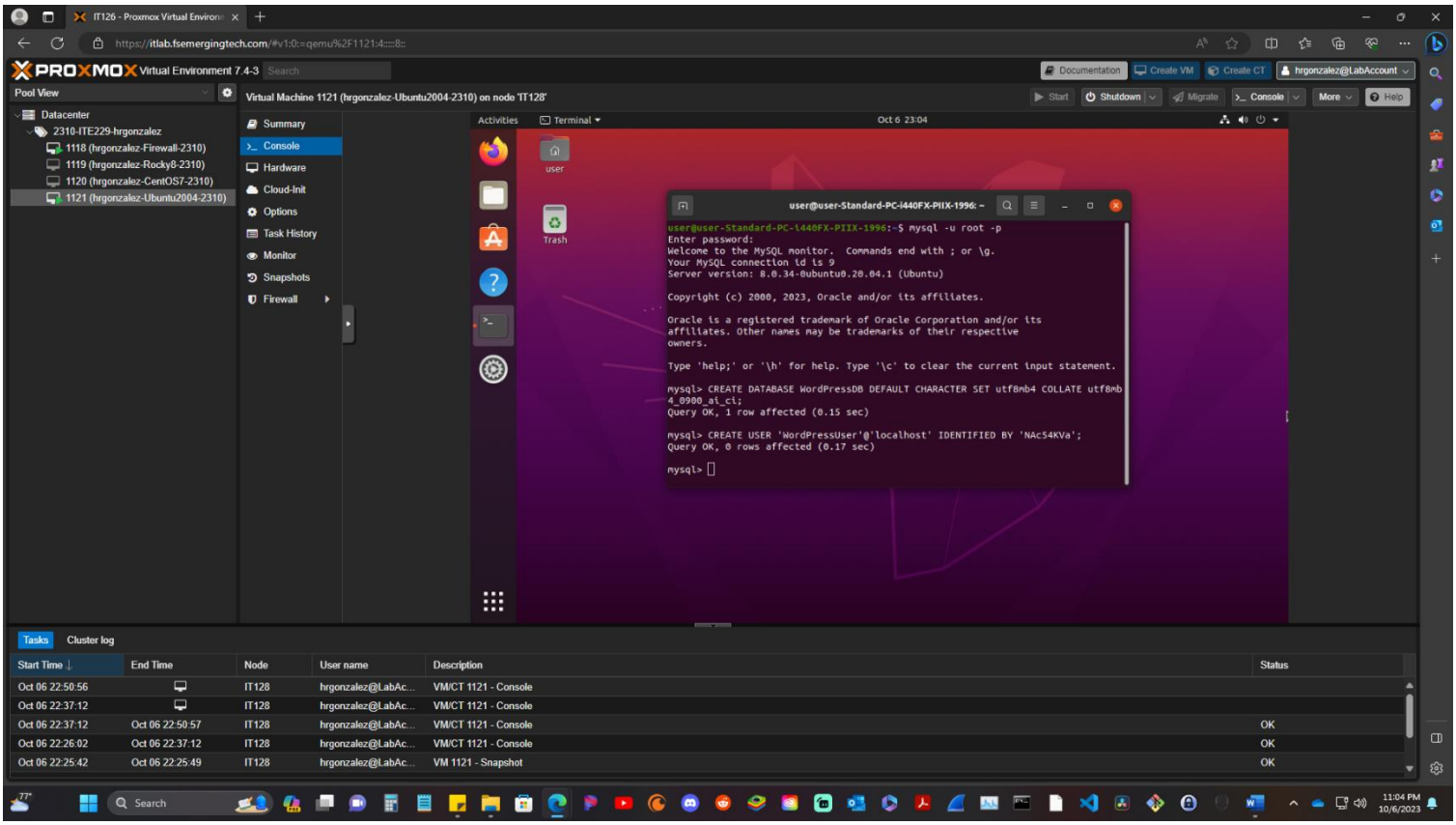
Create WordPress Database in MySQL

Inside MySQL Database we are going to create a WordPress database. In MySQL CLI enter CREATE DATABASE WordPressDB DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci; then hit enter.



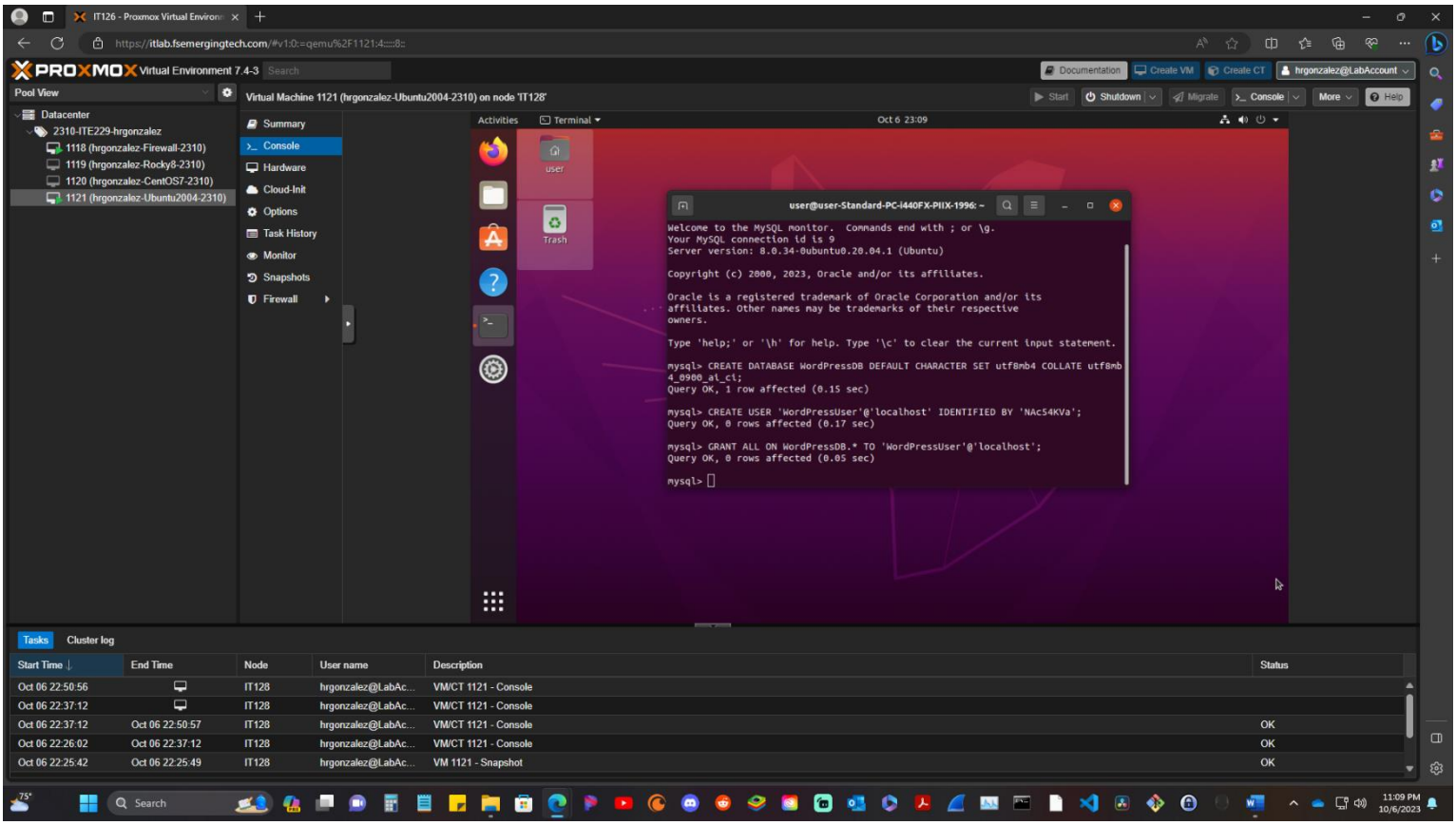
Create WordPress User for MySQL Database

Next step is to create a WordPress “User” for MySQL Database. We are going to create a new password for WordPress User. Create one and save it just like in the previous step (we are going to need it for this step). In the next MySQL line enter CREATE USER ‘WordPressUser’@’localhost’ IDENTIFIED BY ‘yournewpasswordhere’; then hit enter. This will create the User.



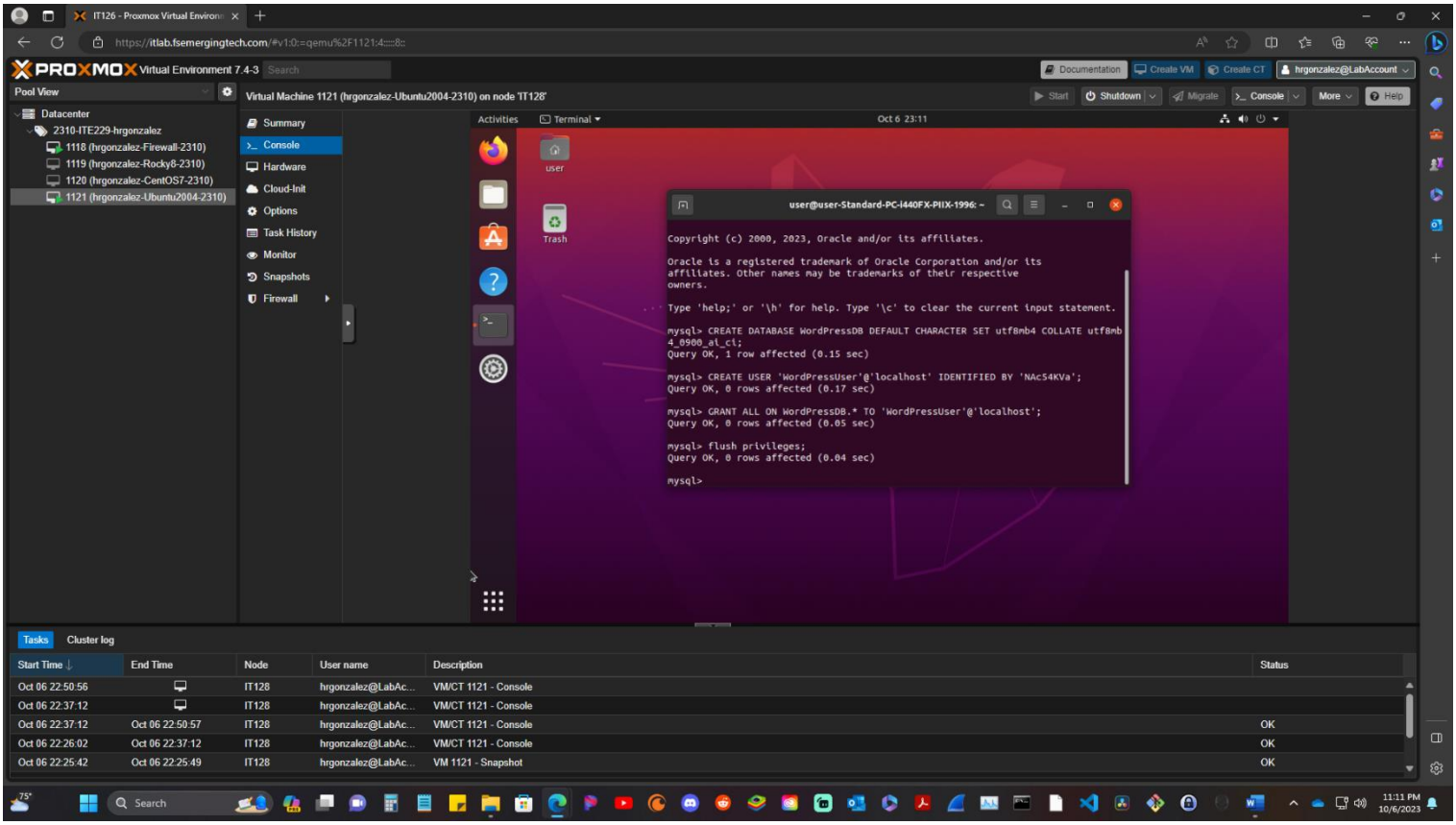
Grant Privileges to this New WordPress User

Now we need to grant permission to this new user. In the MySQL command line enter GRANT ALL ON WordPressDB.* TO ‘WordPressUser’@’localhost’;. This will enable permissions.



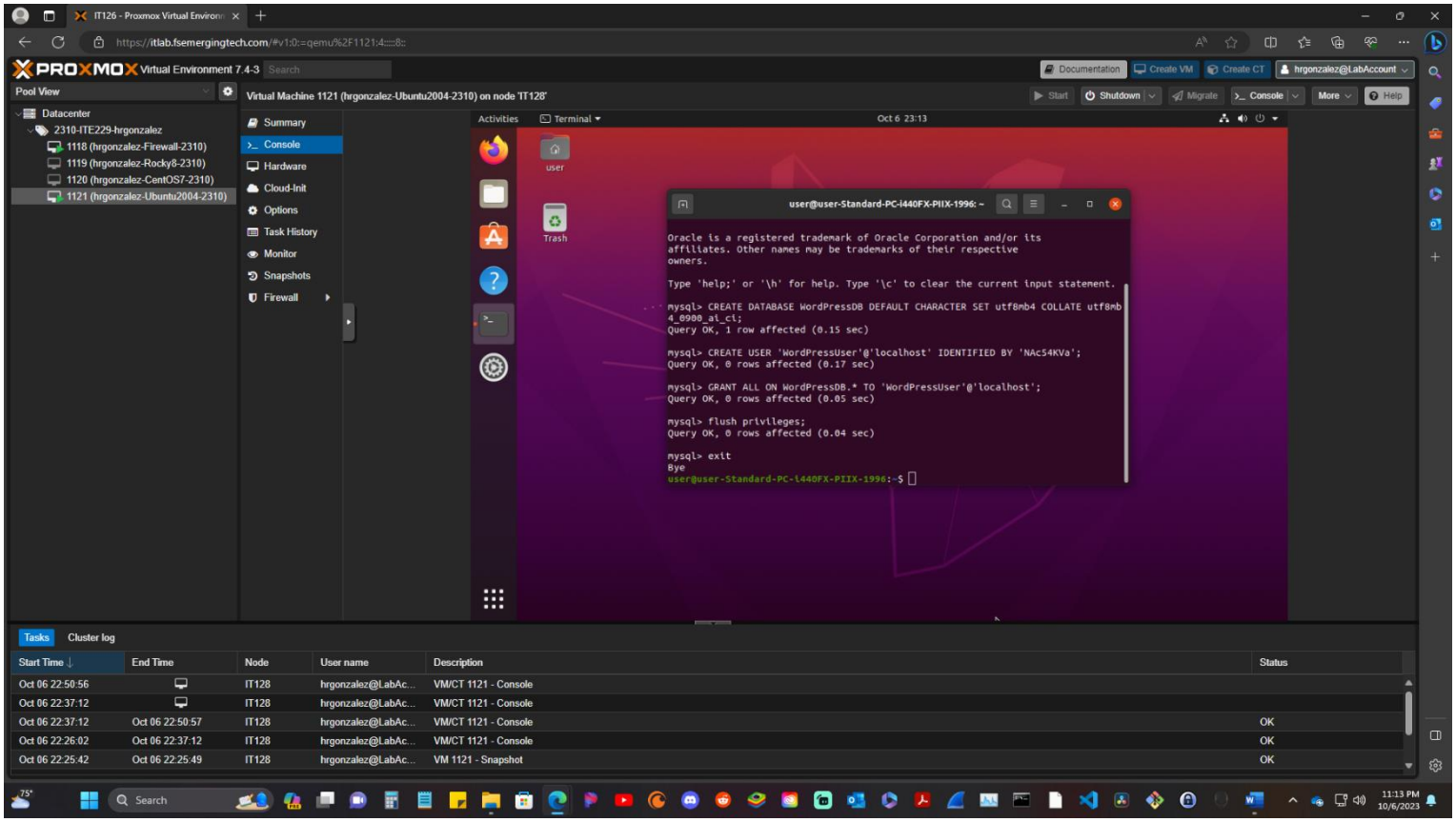
Flush Privileges

Now is time to flush the Privileges. In the MySQL command line enter flush privileges;



Quit MySQL

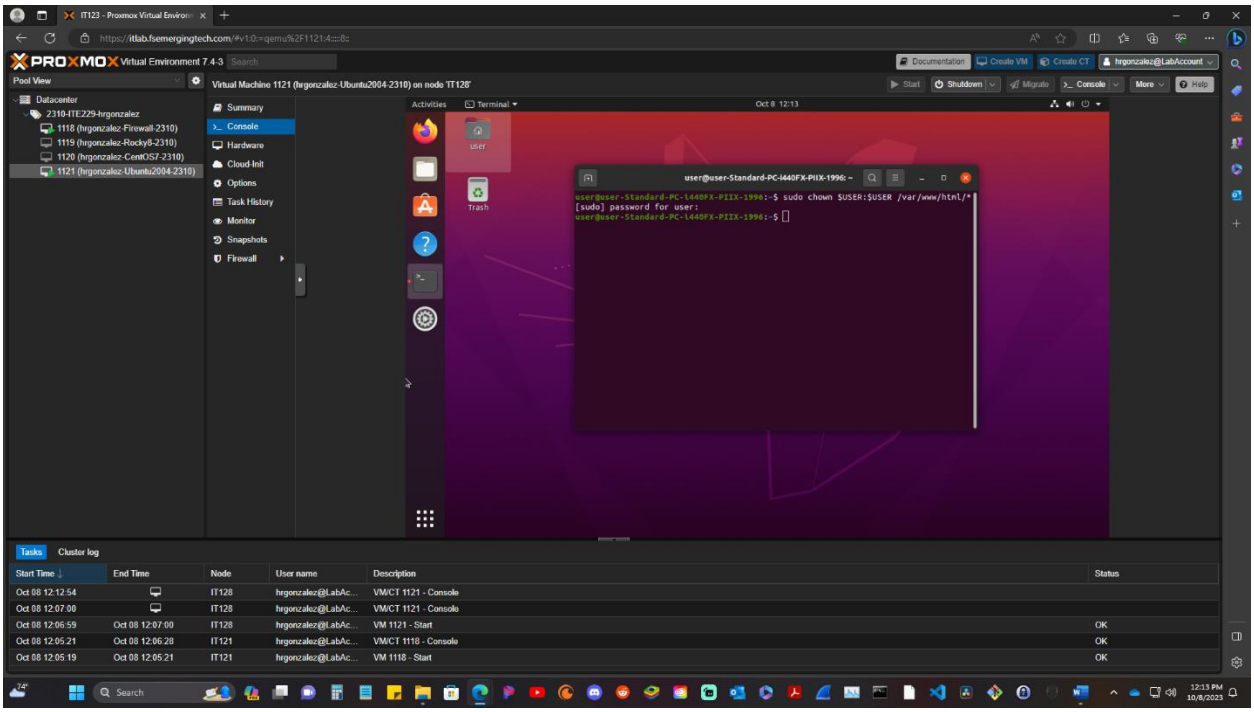
We are done with MySQL. To exit MySQL in the MySQL command line enter exit. You should return back to the username user.



Install WordPress

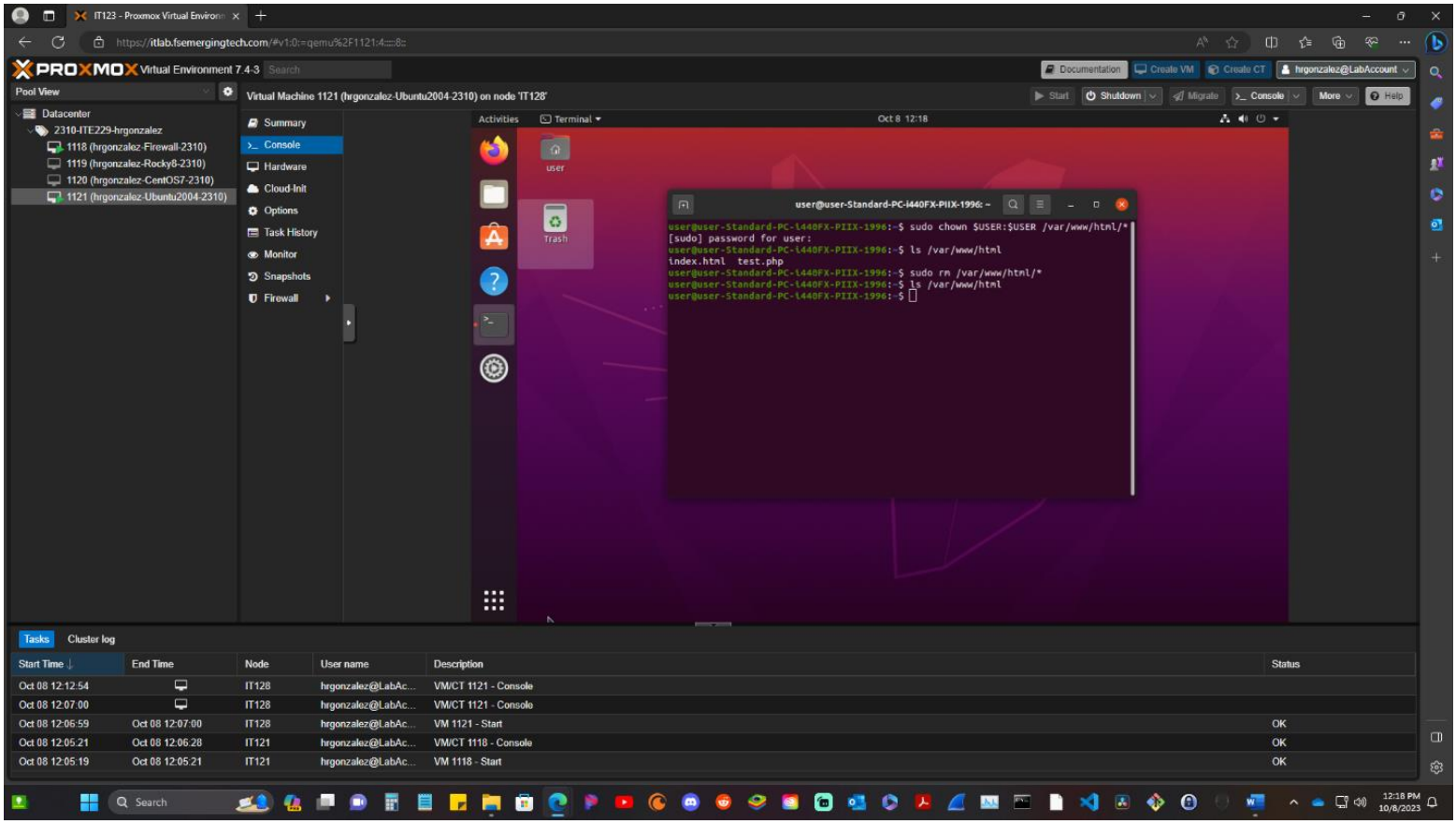
Grant Permission to html Directory to WordPress User

In the Ubuntu terminal enter commmand line sudo chown $USER:$USER /var/www/html/*. This will make sure ownership and group is changed granting permission to all files in that directory.



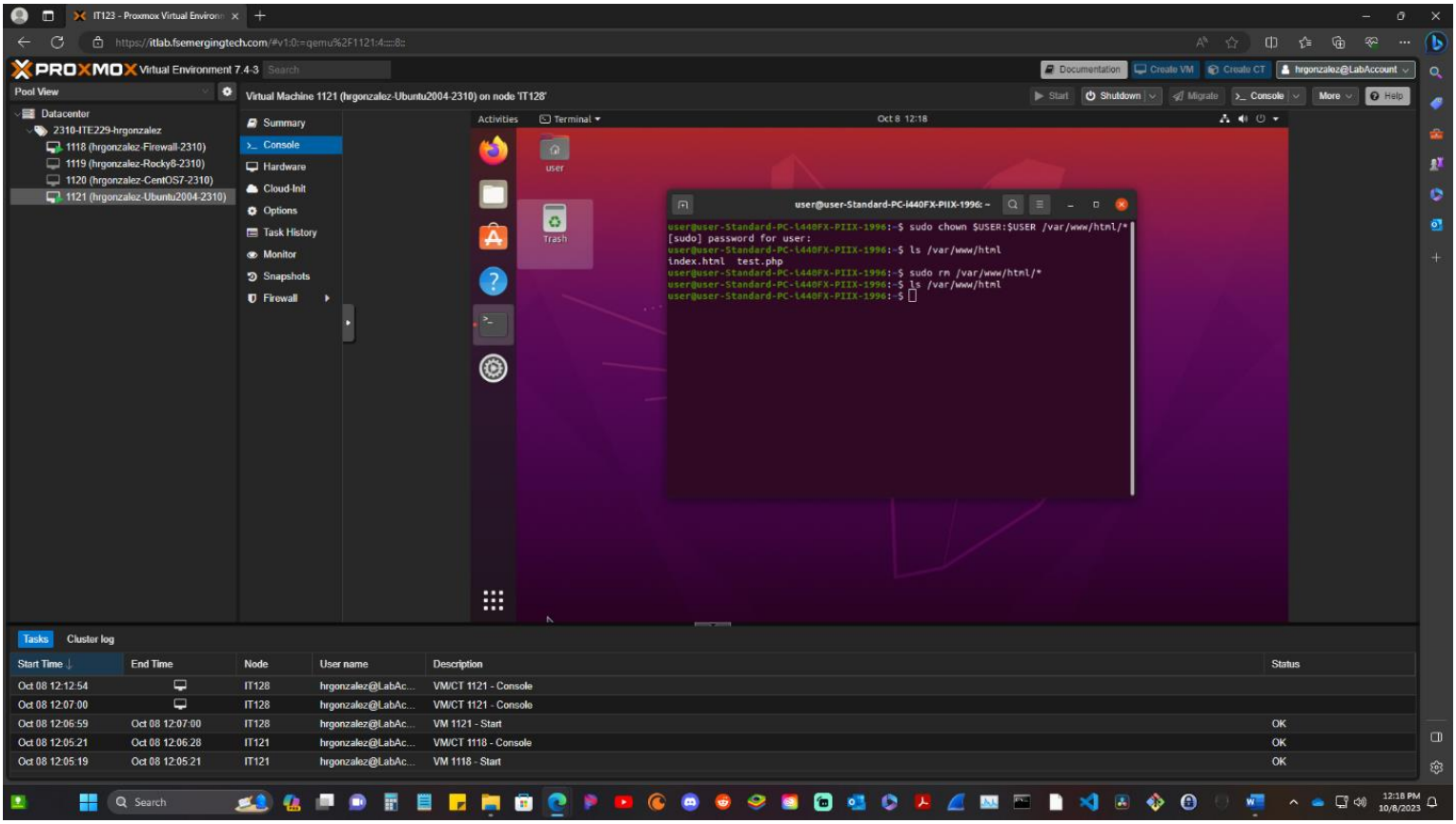
Delete Files from html Directory

Next we need to delete all files from the html directory. To do this enter command line sudo rm /var/www/www/html/*.



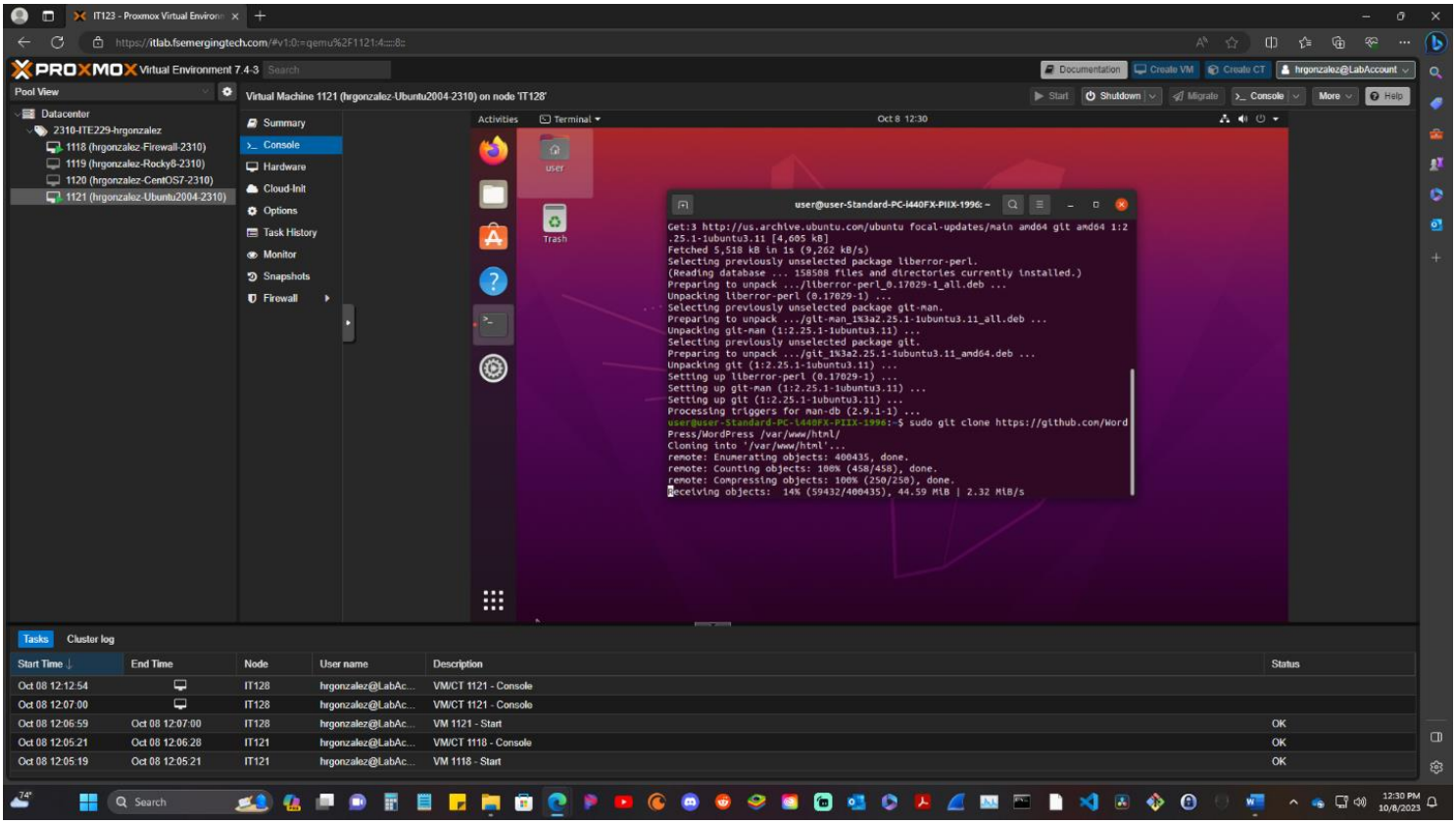
Verify html Directory is Empty

Now we need to verify html directory is empty by inputting command line sudo ls /var/www/html/. This will take us to the html directory and show us the files. If empty it will automatically take you to the username line.



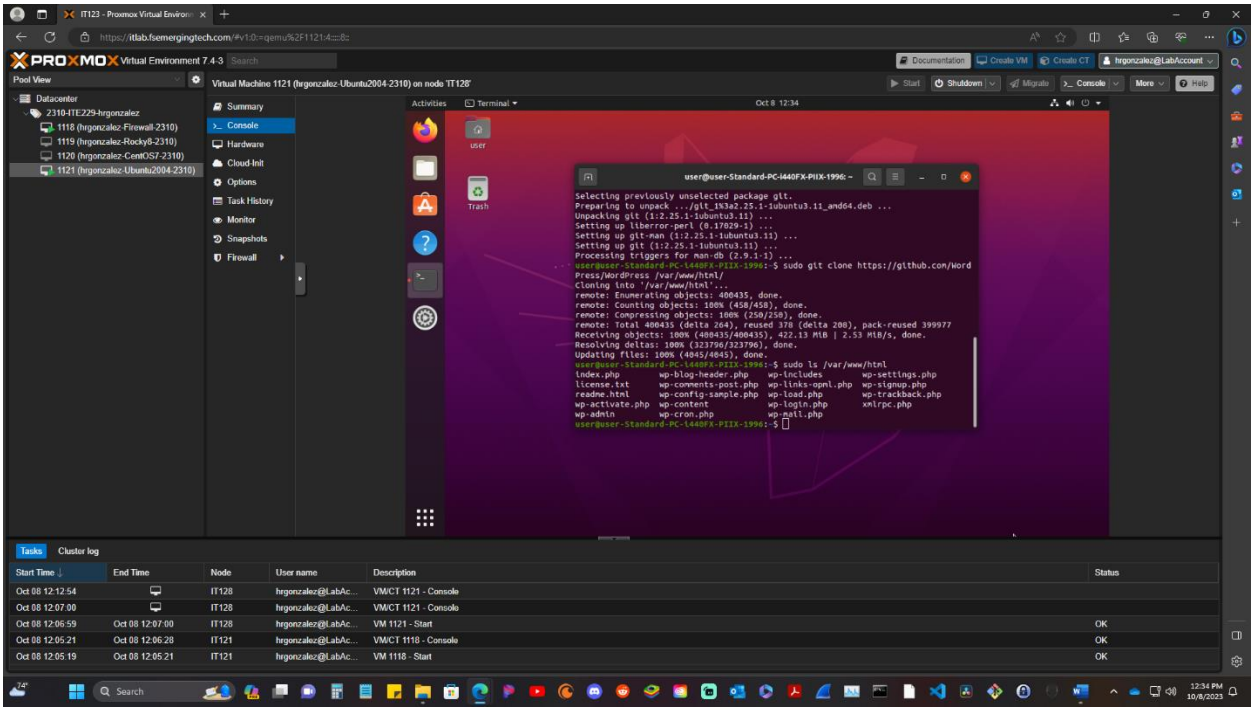
Clone WordPress to html Directory

Next you need to clone WordPress to html Directory, but first we need to make sure git is installed. Let's input command line sudo apt install git -y. If is completed, then it will let you know if not it will undergo the installation process (wait for it to finish). Now we are going to clone WordPress by inputting command line sudo git clone https:///github.com/WordPress/WordPress /var/www/html/.



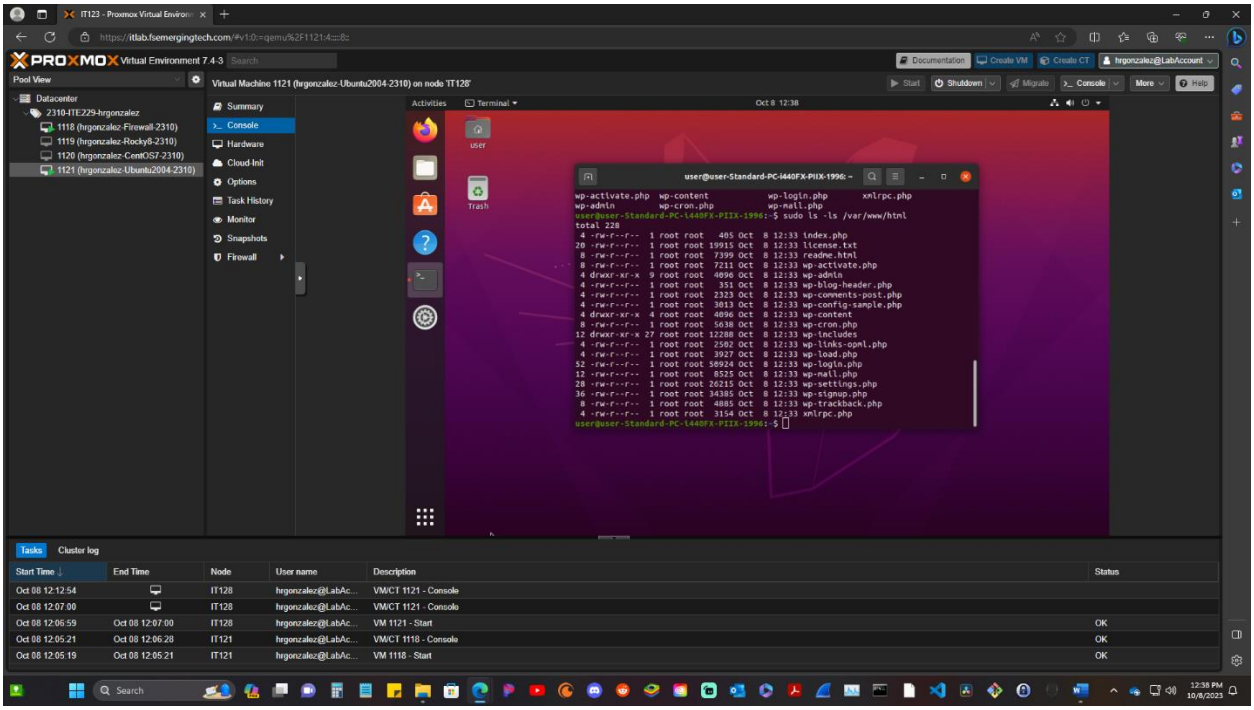
Verify html Directory Contains WordPress Files

Now we need to verify cloning was successful by entering command line sudo ls /var/www/html. You should see all the files in that directory.



Verify Permissions on html Directory

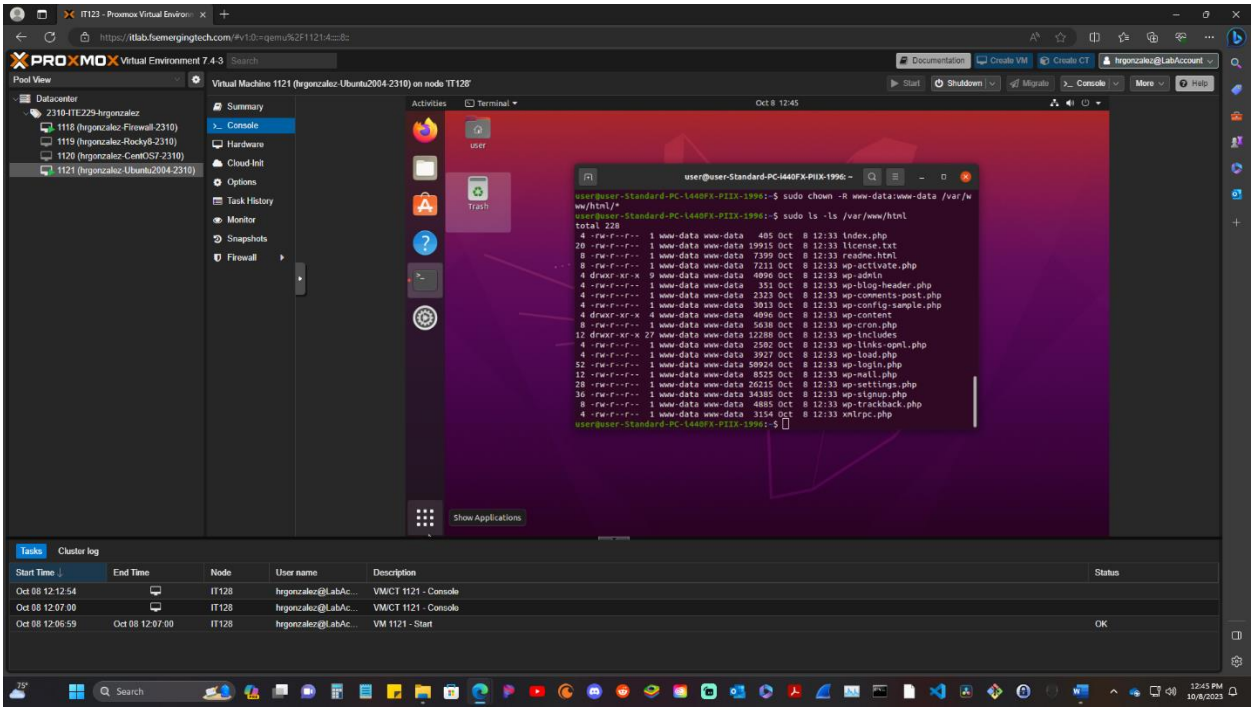
We need to verify our permission on the html Directory. To do this enter command line sudo ls -ls /var/www/html. It should look like the picture below.



Edit Ownership

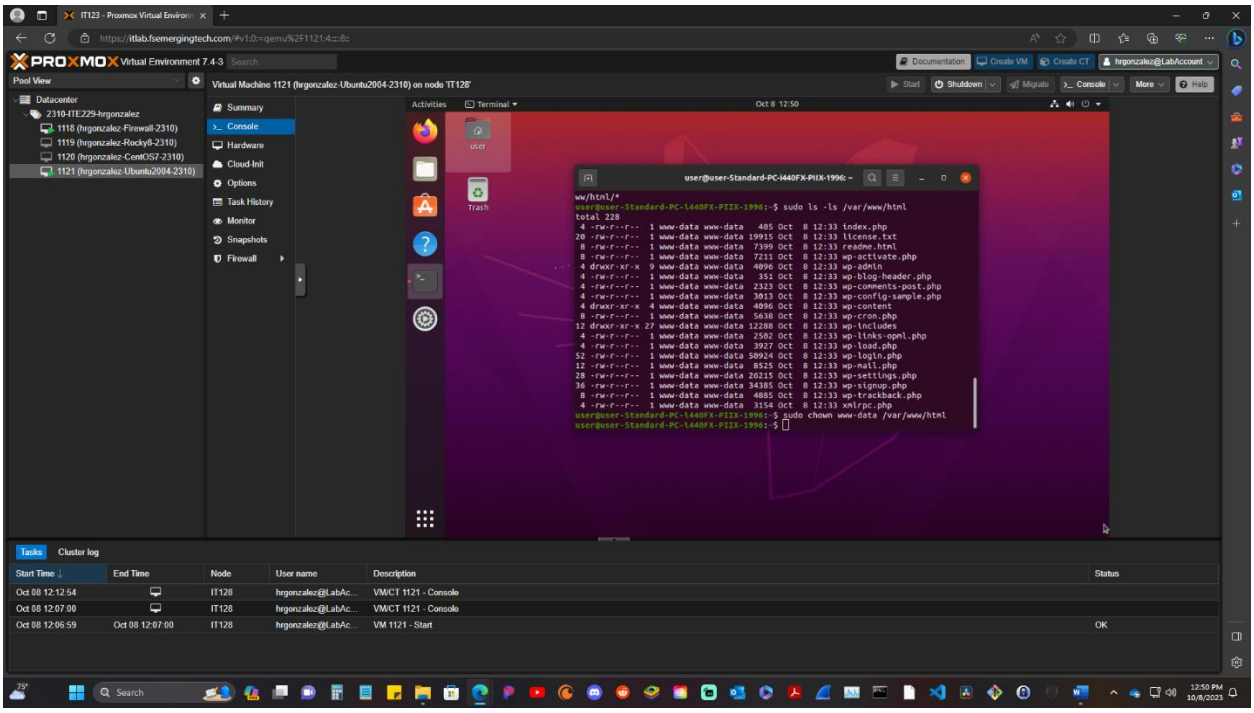
Edit Ownership of Contents of html Directory

You need to set ownership of the html Directory files. To do this you need to enter command line sudo chown -R www-data:www-data /var/www/html/*.



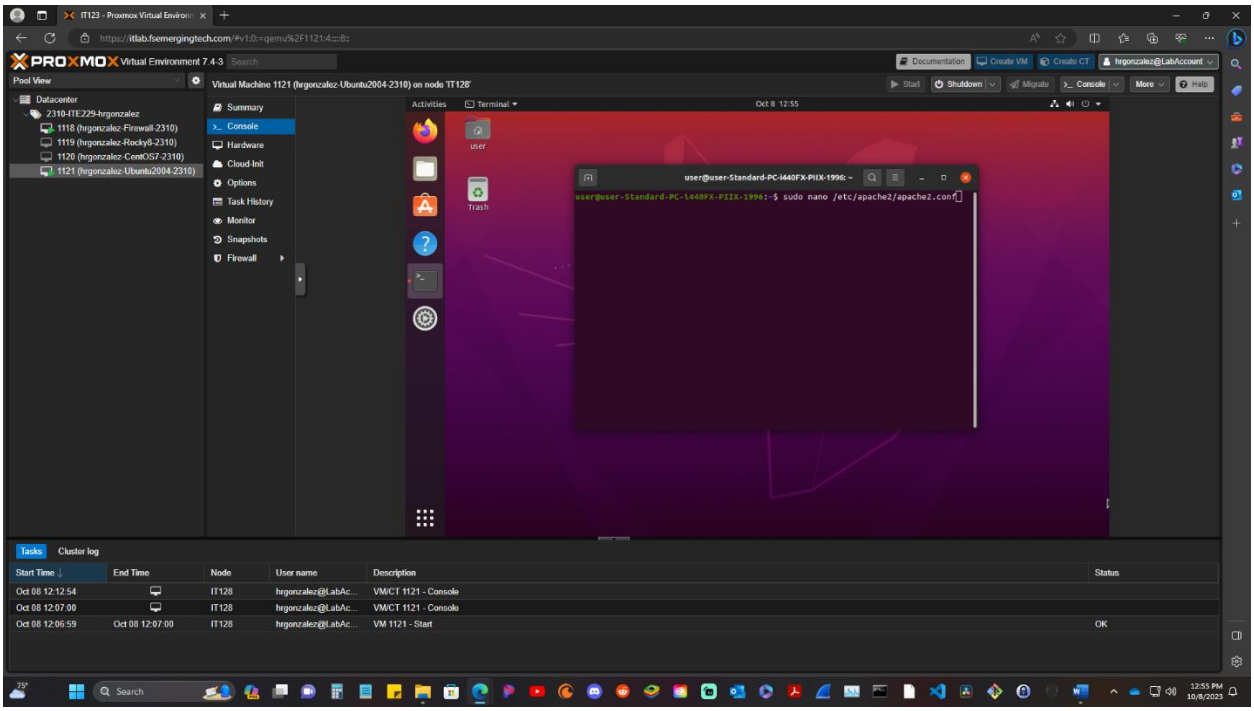
Edit Ownership of the html Directory Itself

Next you need to change ownership of the Directory itself. To do this enter command line sudo chown www-data:www-data /var/www/html it should take automatically do this, and you should be redirected back to your username in the next line.



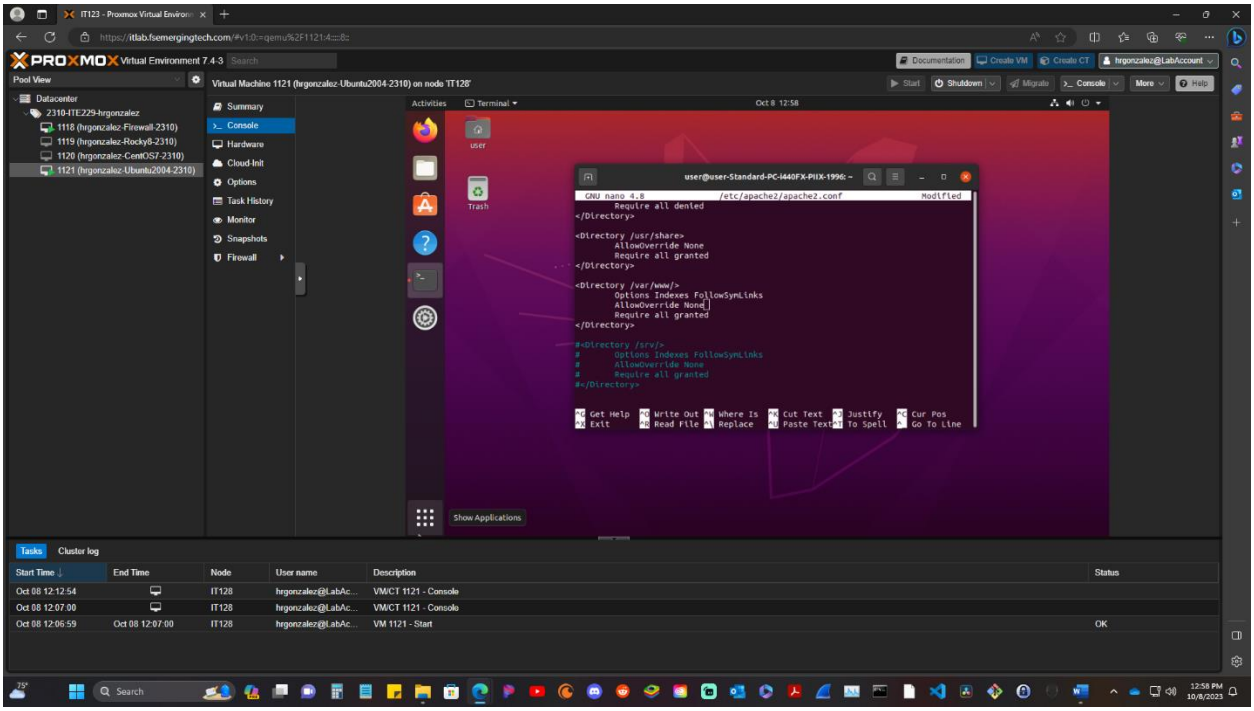
Edit the apache2.conf File

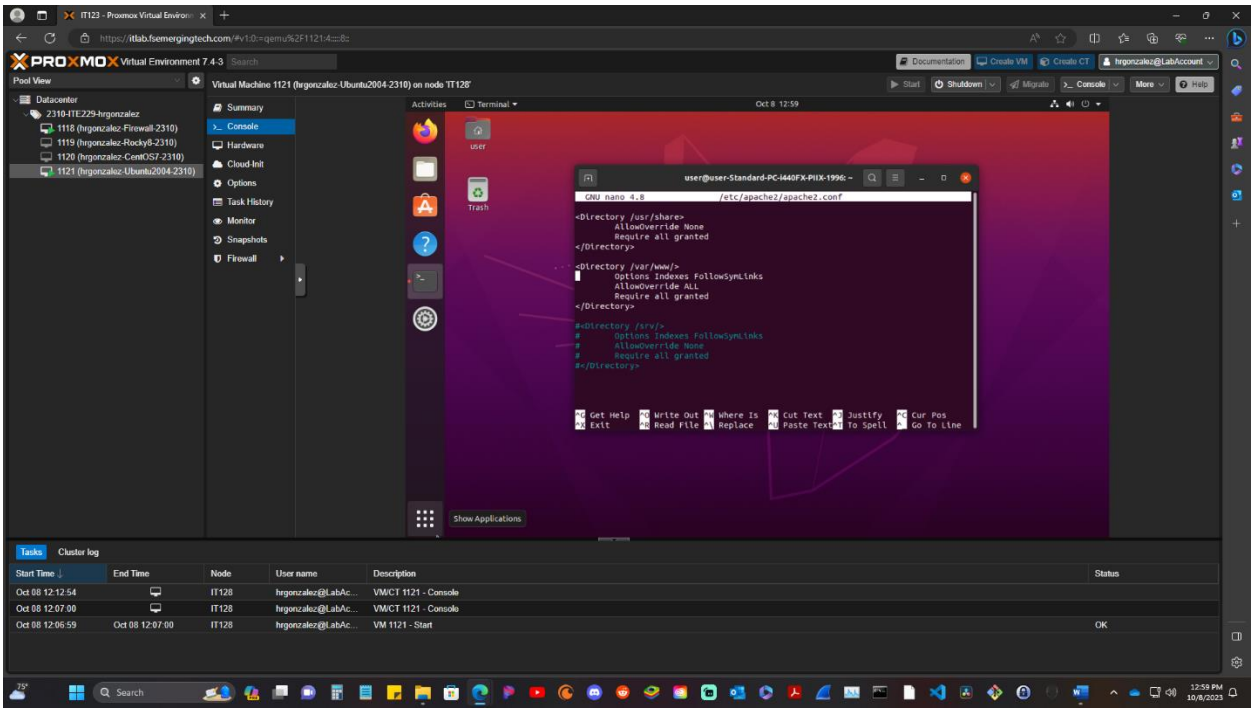
Next you will be enabling override for /var/www path to enhance security. We need to open the Apache config file. Enter command line sudo nano /etc/apache2/apache2.conf.



Override All Default Apache Directives

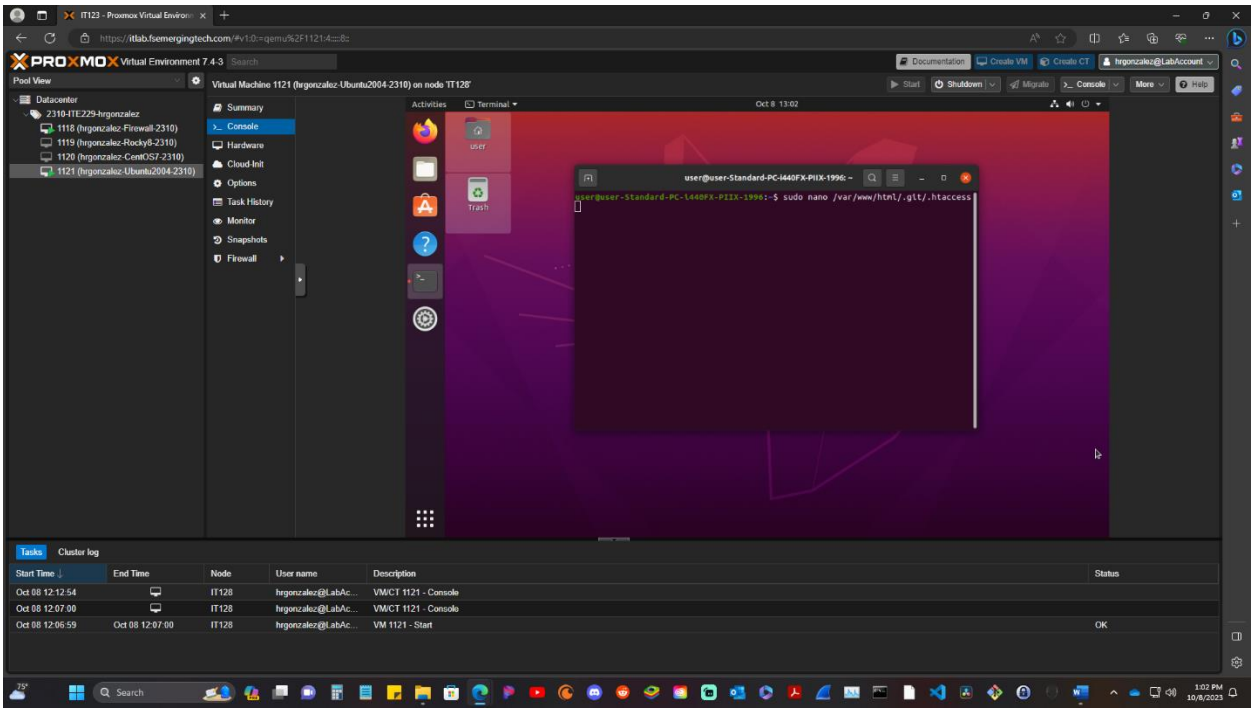
Here we need to find locate <Directory/var/www/> and change AllowOverride from None to All. Look at picture below for reference. After changes are made hit control X to exit, saved changes, then hit enter. You should be redirected back to your username.

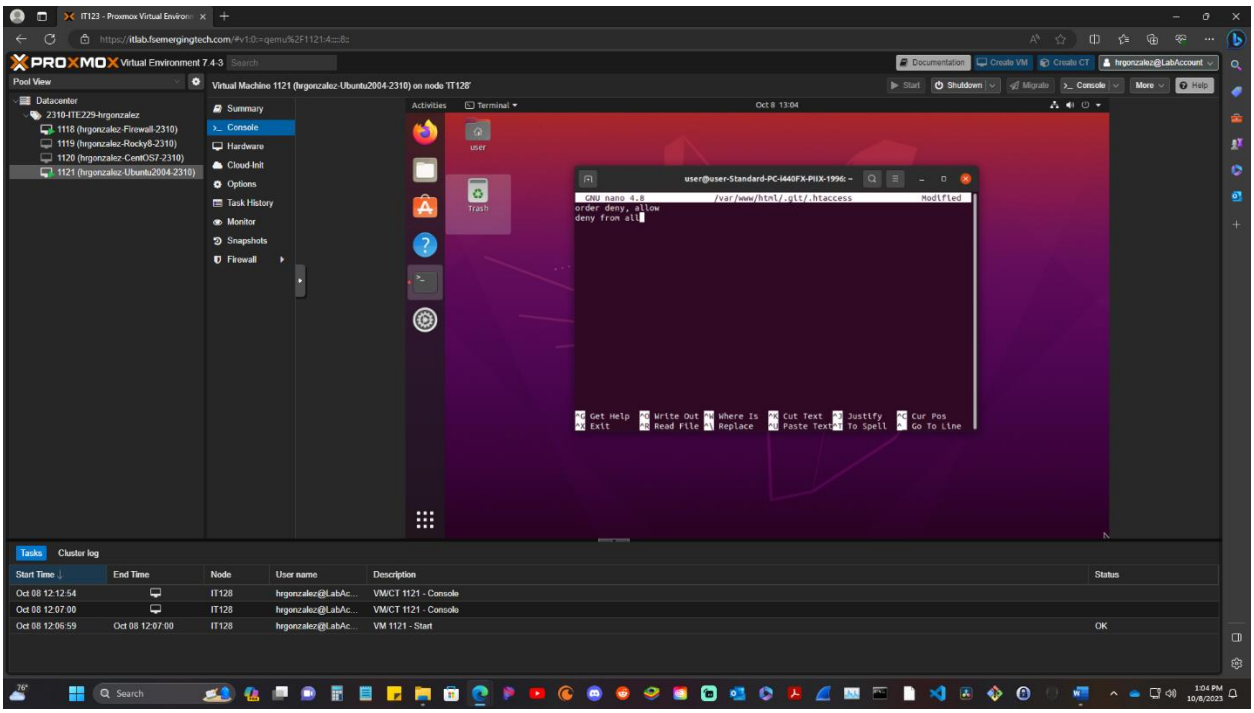




Create a .htaccess File in the /var/www/html/.git/ Directory

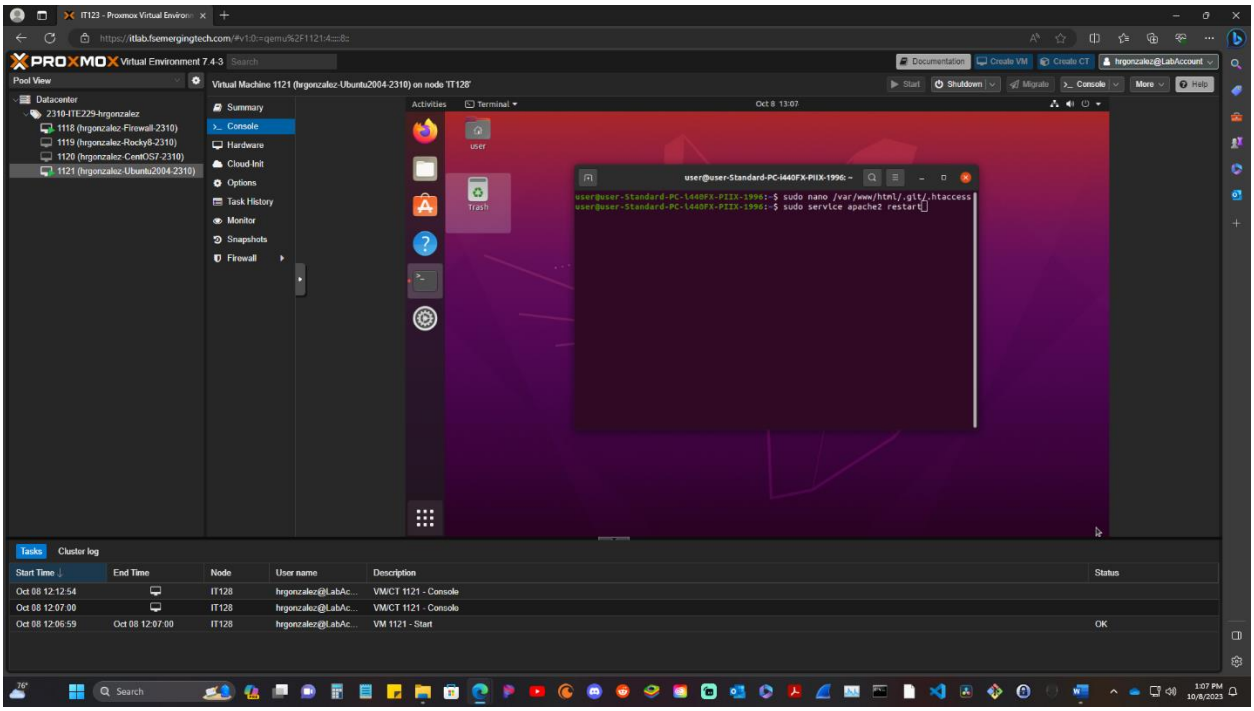
Now you need to secure the .htaccess file under the .git folder to prevent unauthorized users from accessing it. To do this let's enter command line sudo nano /var/www/html/.git/.htaccess. Inside nano type: (*Use picture below for reference)
order deny, allow
deny from all
After hit control X to exit, save changes, then hit enter. It should redirect you back to your username.





Restart the Apache Service

Now you need to restart Apache Service using command line sudo service apache2 restart. This will restart Apache with all the changes.



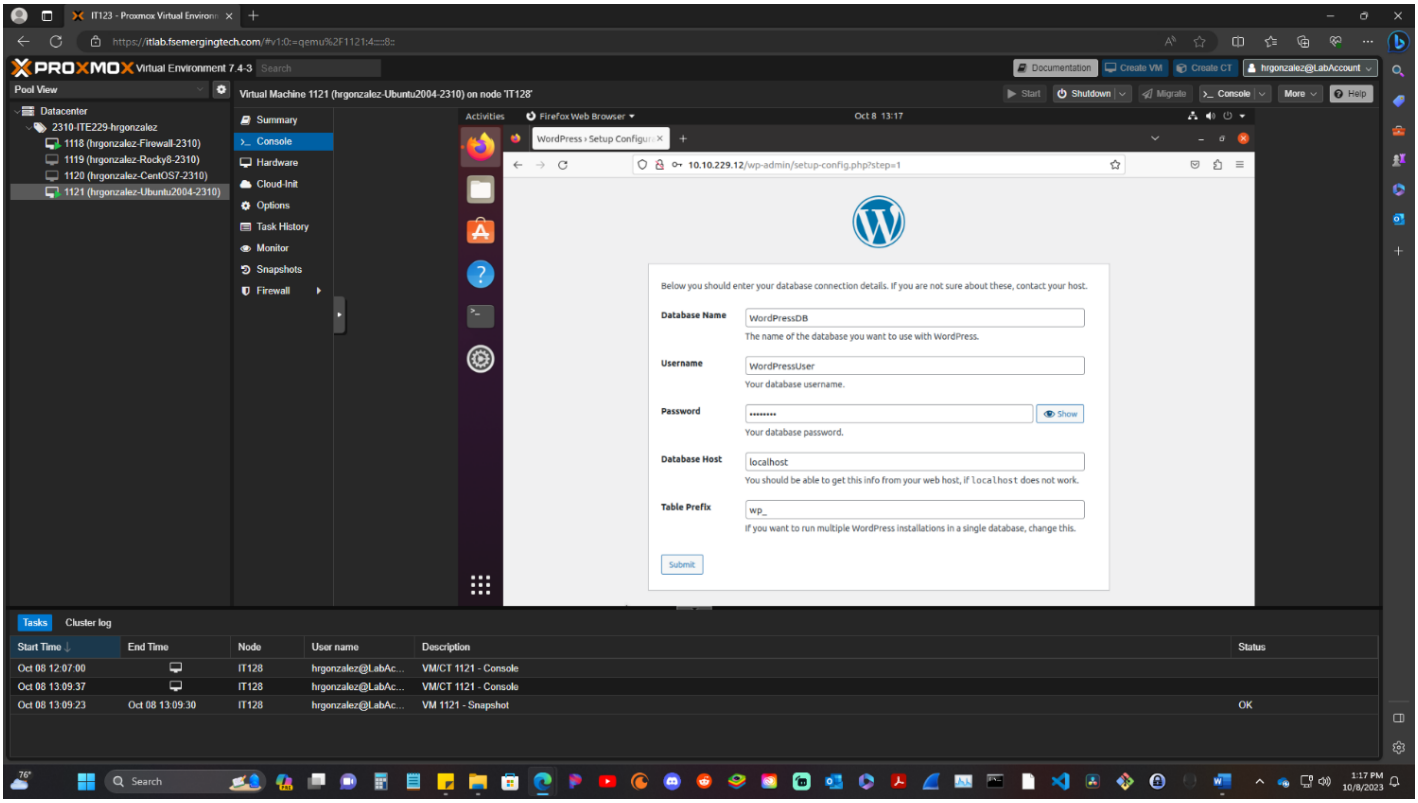
WordPress Configuration

Configure WordPress

WordPress Configuration Selections

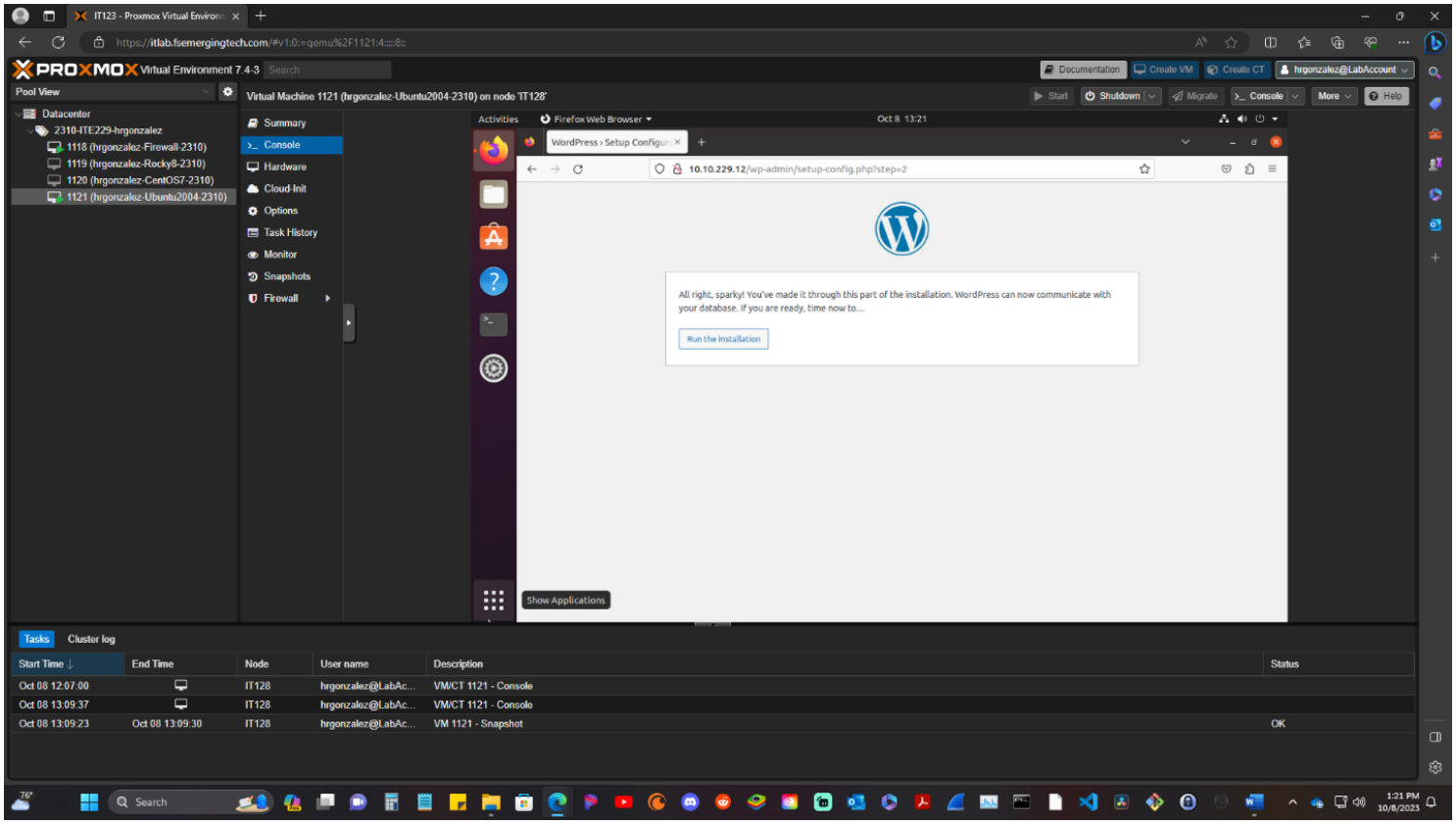
Is time to configure our WordPress site. Using Firefox in Ubuntu enter URL http://10.10.229.12. In this page we are going to enter the following information. Make sure all information is correct then hit the submit button.

- **Database Name:** WordPressDB
- **Username:** WordPressUser
- **Password:** *this is your WordPressUser password*
- **Database Host:** localhost
- **Table Prefix:** wp_



Run Installation

Next you need to hit run the installation button.

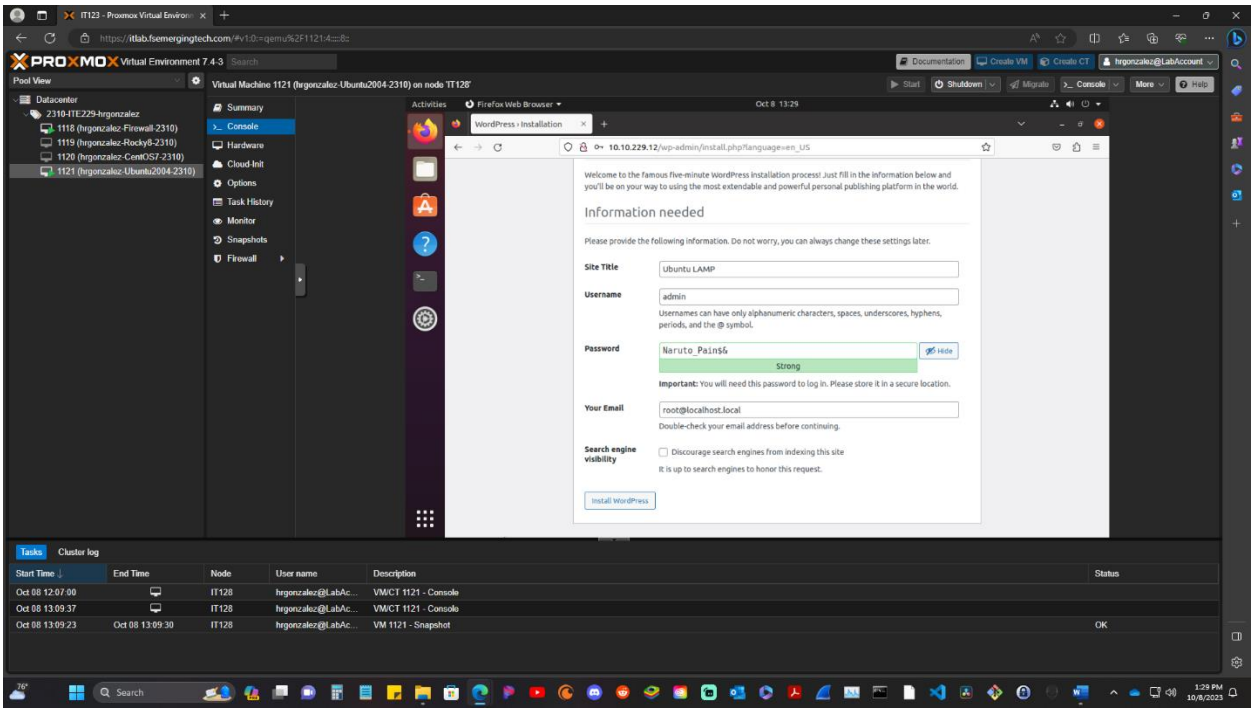


Create an Admin WordPress User

WordPress Site Selections

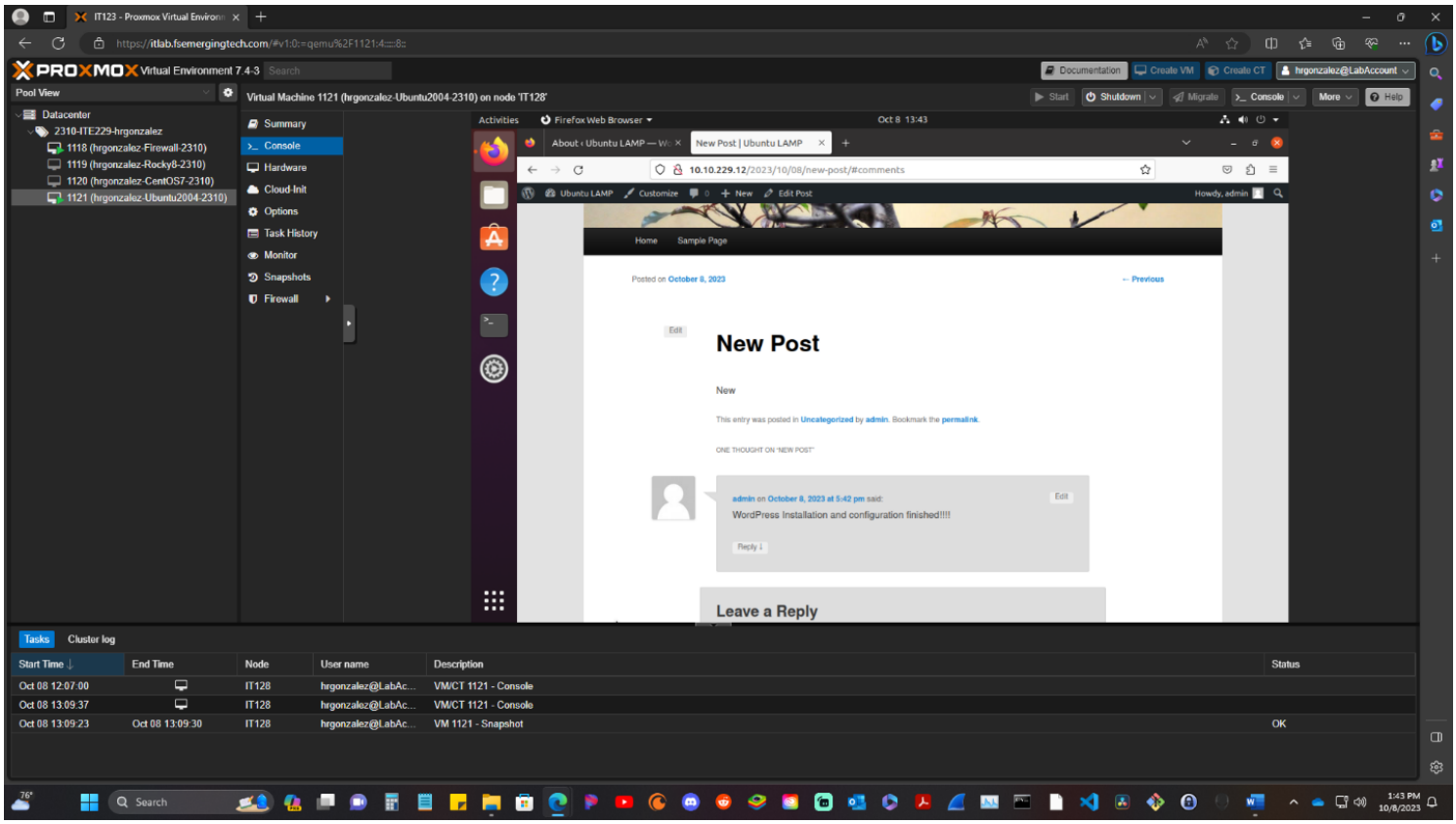
Site Title: Ubuntu LAMP
Username: admin
Password: *You will create this*
Your email: root@localhost.local
Search Engine Visibility: leave unchecked

Let's create an admin WordPress User. In this page we will enter the information provided above. You need to create your own password or use the one that generate it for you (*be sure to save this password). Use the picture for reference. After information is filled in hit install WordPress button. It will bring a success page next, here hit Log in. It will redirect you to the WordPress log in page.



Test WordPress

To test WordPress you need to log in to the WordPress page by using your credentials. Once logged in, it will take you to the WP dashboard. Here you can make a theme change by hovering over appearance and choose a theme and activate it. Next you need to create a post by hovering over to post and click all post > Click on Add New Post in the upper left corner. Type a title and then New below Title. Hit Publish on the right hand side and hit publish again. Click on the WordPress W on the left side upper corner this will take you back to the dashboard. Time to test our changes. Open a new web browser and in the URL type your blog address 10.10.229.12 and hit enter. You should see your picked theme, title and under it the word New. If you see this congratulations test is successful.



WordPress Security Settings and Configurations

WordPress Security Summary

WordPress directory (folder) and file permissions is critical for data security and integrity. WordPress relies on a variety of files and directories to work and ensuring that only allowed users and processes can access and modify them is critical. You may prevent unauthorized access, data breaches, and harmful behaviors like code injection by properly defining permissions. Handling permissions inappropriately can reveal critical data, jeopardize the site's integrity, and expose it to attack. Successfully managing permissions is a crucial component of keeping a secure and dependable WordPress website, which greatly helps in the protection of your site's content, user information, and the general performance and reputation of your online presence.

I implemented changes to WordPress site by appointing the right permissions for folders and files within the WordPress installation folder. For the WordPress admin file I added an extra layer of authentication. This means that when you log in to WordPress admin website it will redirect you to input a separate username and password redirecting you to the normal username and password then granting you access to the WordPress Dashboard. For the WordPress content folder I disable Indexing by creating a security file Options -Indexes. This deny a user from looking into the folder of uploads on our website. As for WordPress content folder I went into the upload subfile and created a security .htaccess which denies any php file from being uploaded to that folder.

The wp-config.php file in WordPress must be secured since it contains sensitive information such as database credentials, site configuration, and encryption keys. If this file is compromised, an attacker can obtain unauthorized access to your database, change site settings, or introduce malicious code, potentially resulting in data breaches, site downtime, and user distrust. I secured this file by disabling edit themes, plugins, and editing files by implementing a code within the files writable content. I then proceeded and changed the permissions on the file to read only to user, group, and others who have access to the file.

Defense-in-depth

Securing your WordPress site is absolutely critical, and it all starts with keeping your core WordPress software and all plugins up to date. This step is like ensuring that your home's front door is always locked and secured. Just as you wouldn't leave your door wide open, you shouldn't leave your website vulnerable to outdated software. Moreover, it's essential to maintain regular backups of your site and take screenshots when making changes. This is your insurance policy, safeguarding your digital assets.

In our approach to security at IT & Cybersecurity Solutions, we went beyond simple updates and backups. We understood that strong and unique passwords are a must, but we didn't stop there. We fortified our defenses by implementing two-factor authentication for WordPress. This extra layer of security is like installing a deadbolt lock on your home's front door, adding a formidable barrier to unauthorized access.

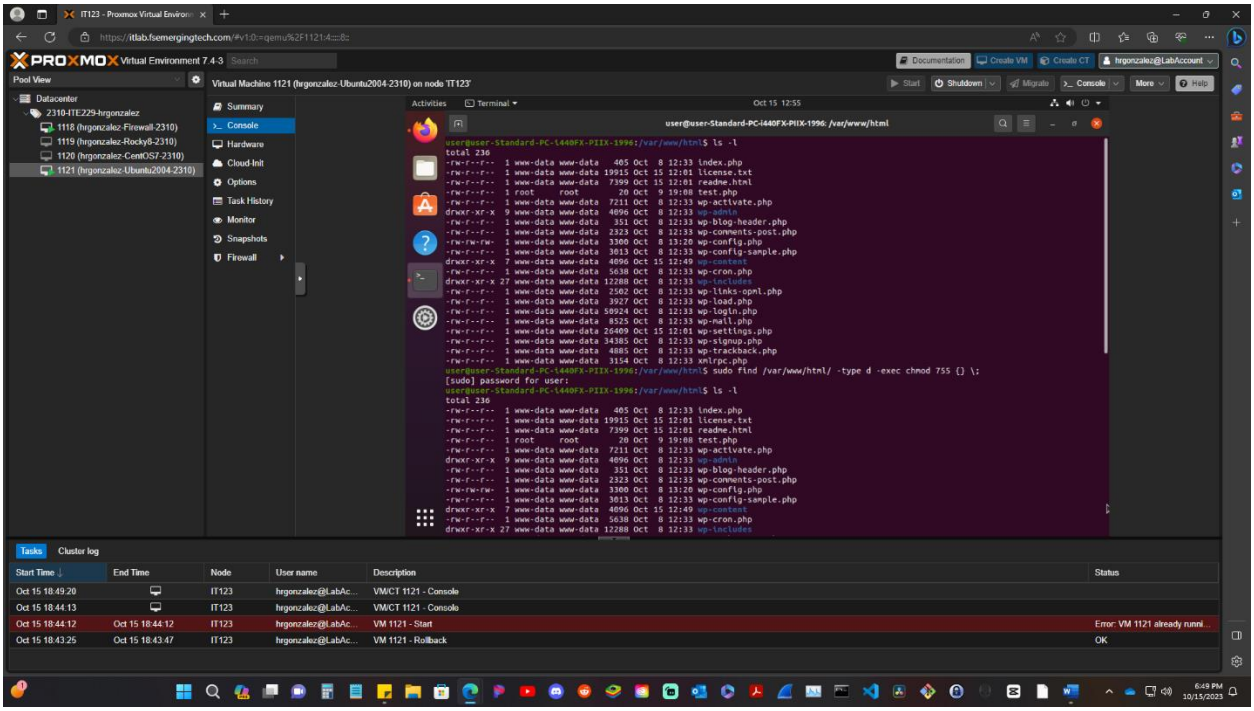
File permissions were another critical aspect of our security strategy. We meticulously ensured that no one could simply walk in and access sensitive files. Installing and enabling the Shield Security plugin was akin to adding a concrete wall between malicious actors and our WordPress site. This plugin not only acts as a defense against digital invaders but also monitors incoming and outgoing traffic, maintaining a comprehensive log of all changes and the overall security level of our WordPress site.

By employing these robust security measures, we effectively constructed a fortress around our WordPress site, making it exceptionally challenging for anyone with malicious intent to breach our defenses. Shield Security serves as our vigilant guardian, ensuring that our fortifications remain solid and reliable.

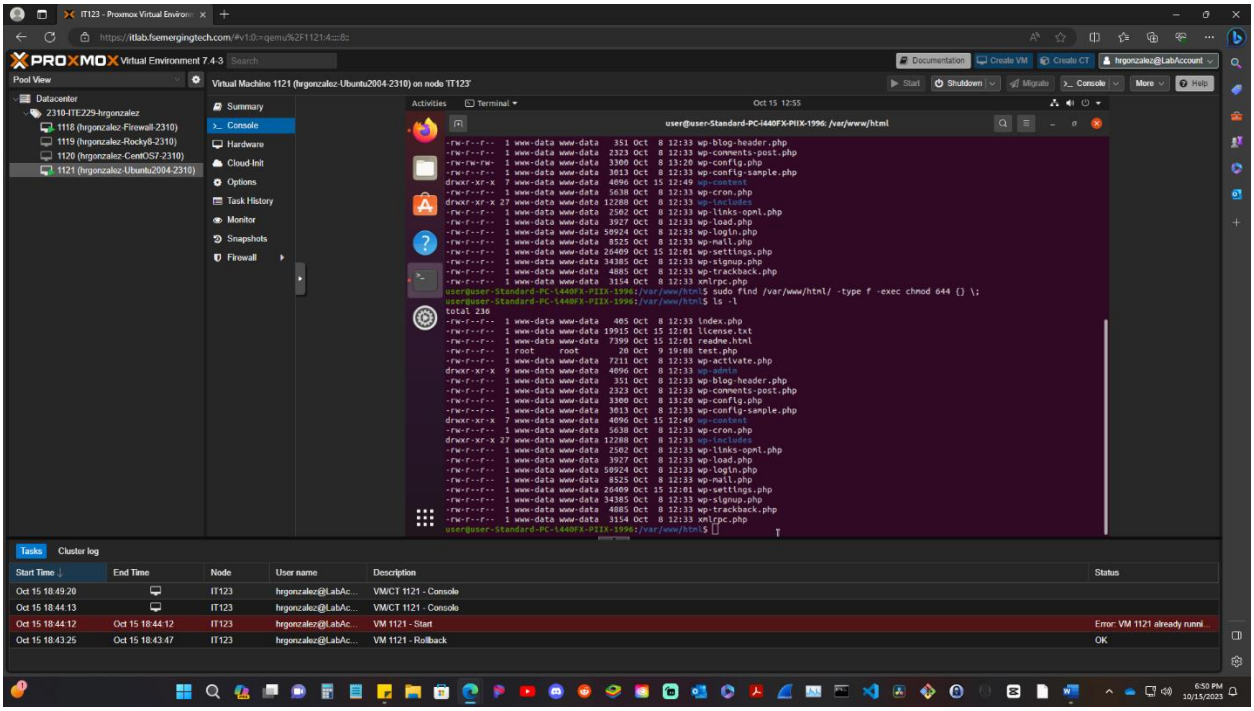
In summary, our defense-in-depth strategy involved the implementation of multiple layers of protection, consistent updates, and continuous monitoring for intrusion alerts. Just as you would take every precaution to safeguard your home, we've applied these principles to protect our digital presence, making our WordPress site as secure as possible.

File Permissions

Before Changes



After Changes



Vulnerability

File permissions are an important part of computer security, but they can also pose risks if not maintained properly. Overly permissive permissions, which provide excessive access to files and directories, are a severe vulnerability. Malicious actors can easily edit, remove, or inject malicious code into files that are set to be world-writable or have careless permissions in directories. On the other hand, overly tight permissions can prevent genuine users and apps from functioning properly, resulting in compatibility concerns. Incorrect user authorization management can expose sensitive data to unauthorized users.

Configuration

In order to configure files to meet the proper file permissions for WordPress we went and follow WordPress hardening (hardening- a word for securing further) configuration process. I used the following command:

sudo find /var/www/html/ -type d -exec chmod 755 {} \; this changed the WordPress directory(folder) permissions.

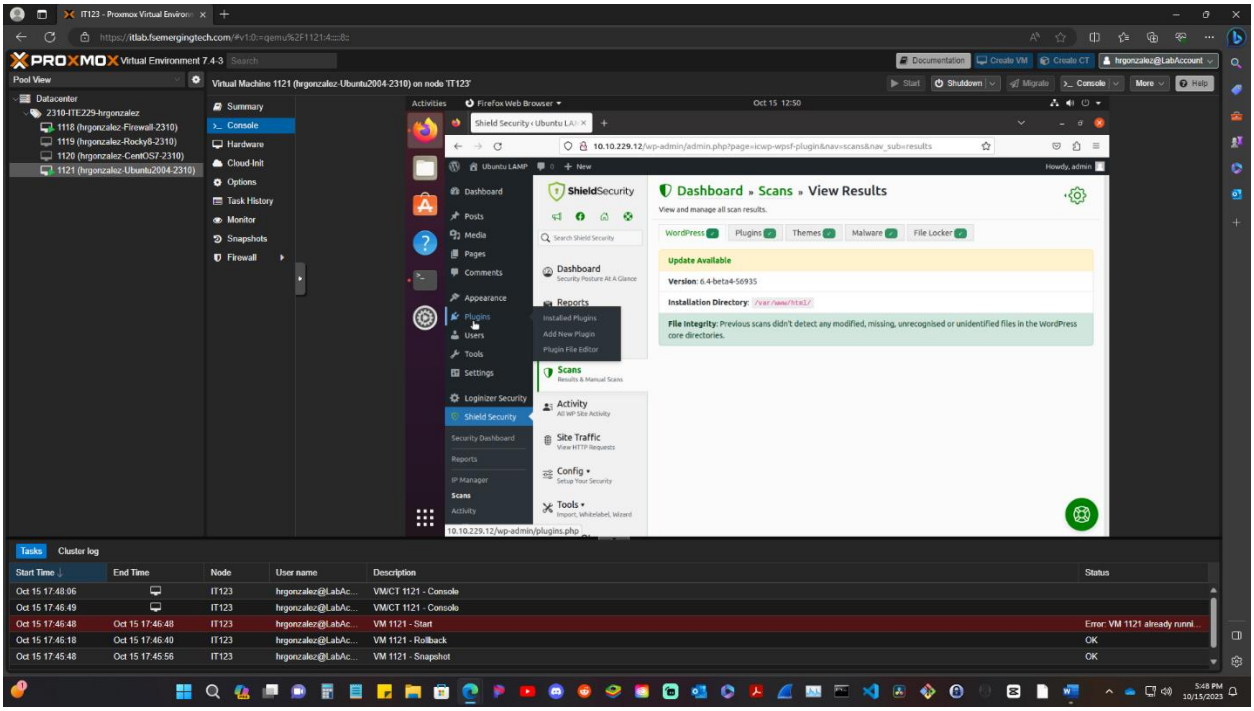
Sudo find /var/www/html/ -type f -exec chmod 644 {} \; this changed permission for files within WordPress directory.

Validation

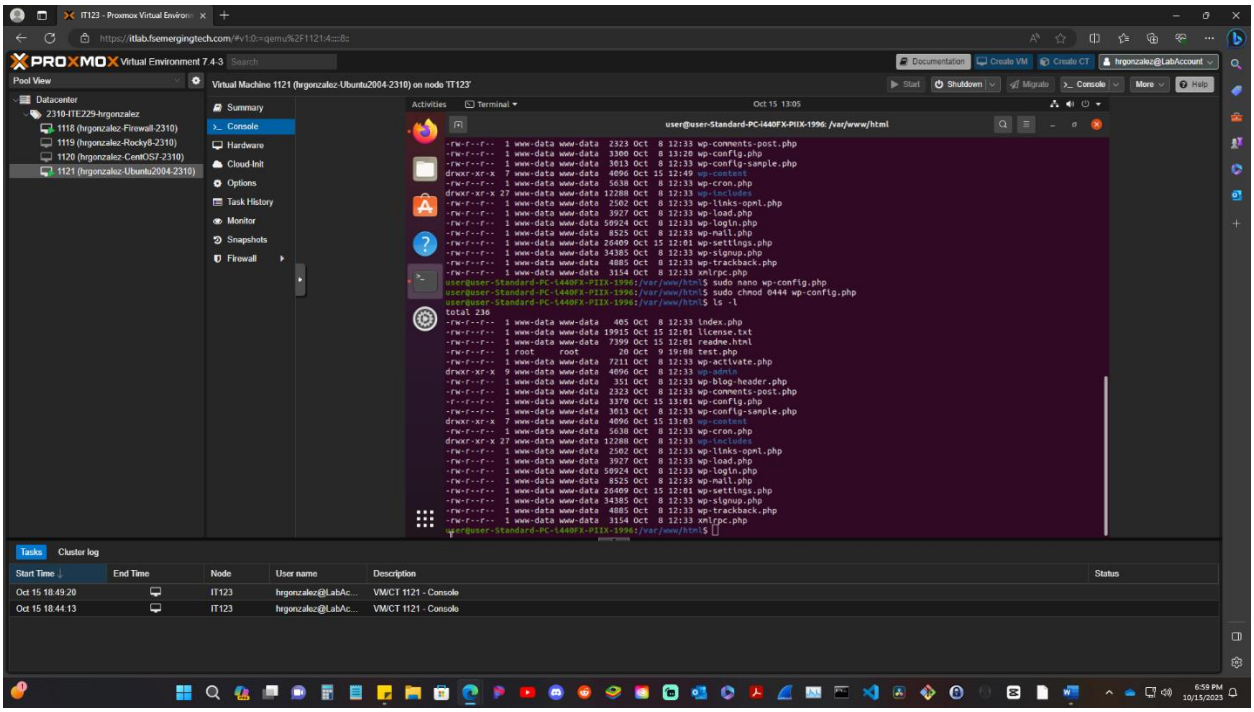
By performing command ls -l after the permission changes you can verify permissions and compare from before and after, along with WordPress site permissions to make sure all are correctly configured.

Securing wp-config.php

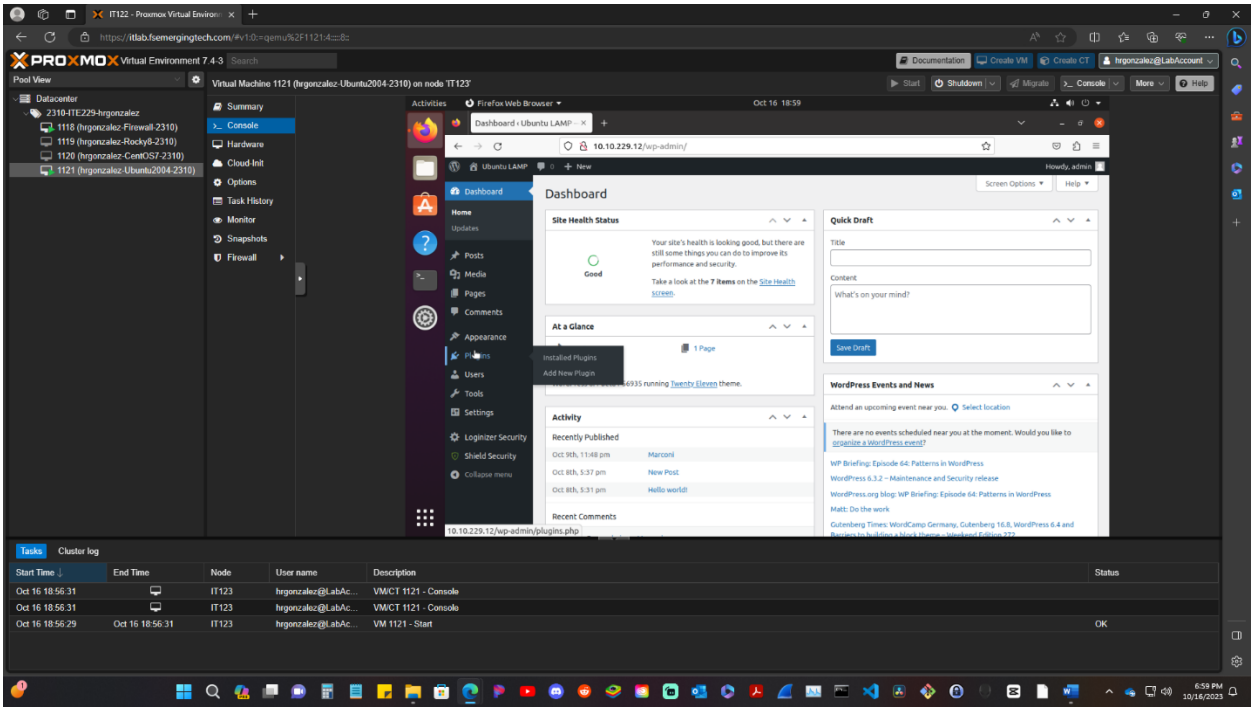
Before Changes



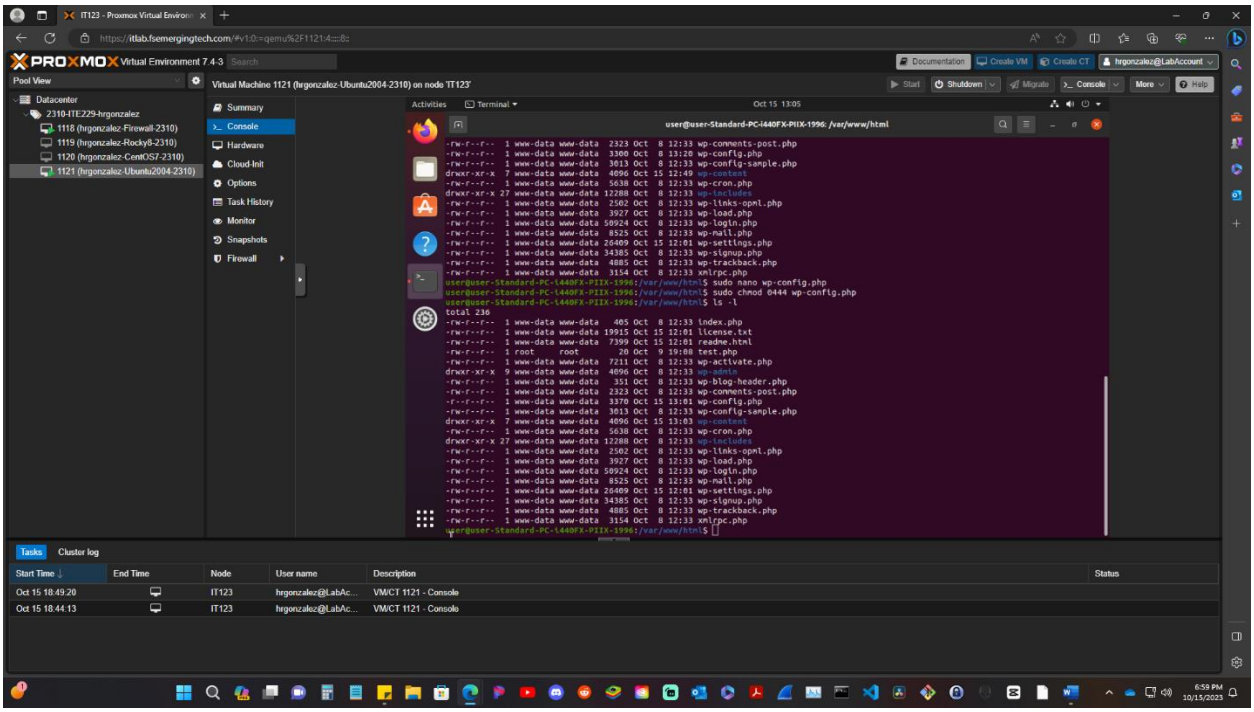
wp-config.php permission displayed on top (above command lines).



After Changes



wp-config.php permssion changed displayed on the bottom after command lines.



Vulnerability

Unauthorized access is one of the most serious vulnerabilities related with the wp-config.php file. If an attacker acquires access to this file, they will be able to obtain critical information such as database credentials, encryption keys, and authentication salts, potentially leading to a complete compromise of the website. This is especially dangerous because it allows direct access to the site's database, which contains all content and user data.

Configuration

To meet the file security level of security we made 2 changes to the file. We disable File Editing, and we changed the file permission settings to only read for owner, user, and others. Allowing for two levels of security. Changes were made using command:

sudo nano wp-config.php and in the following window scroll all the way down and add context define('DISALLOW_FILE_EDIT', true); this will disable edit themes, edit plugins, and edit files in the WordPress admin dashboard.

sudo chmod 0444 wp-config.php changes permission of file to only readable for everyone.

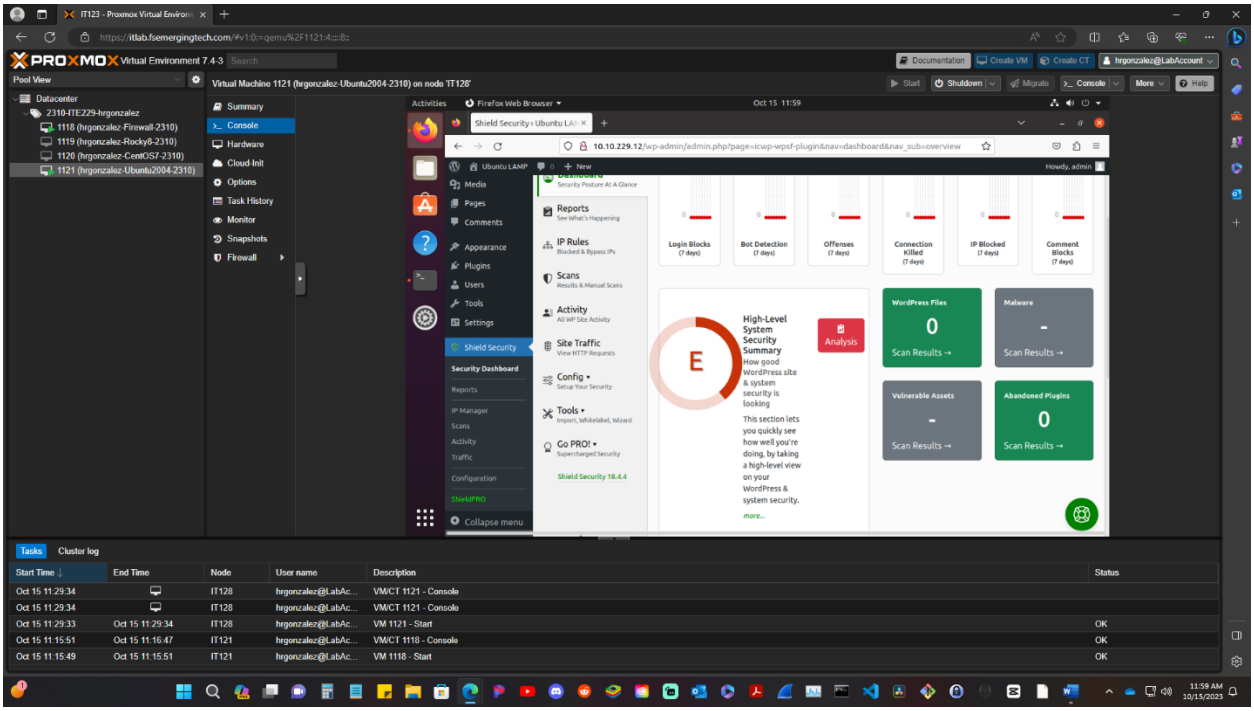
*PS you can also move the file location for a third layer of security.

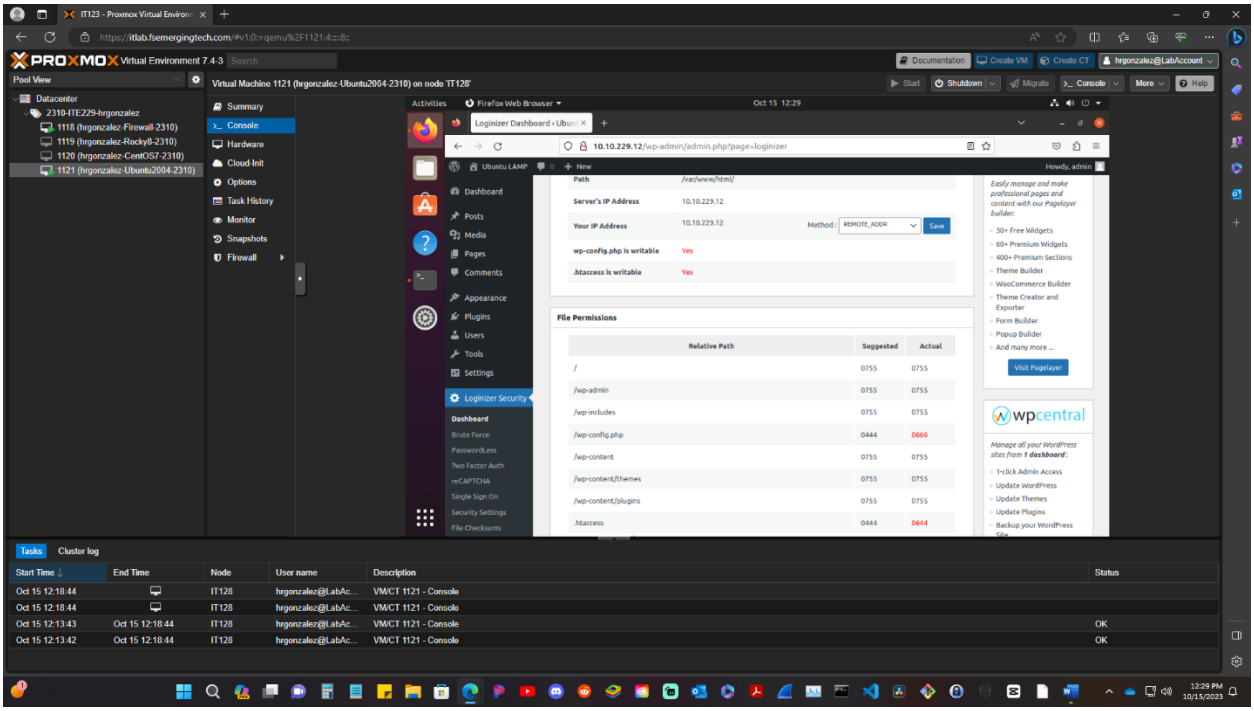
Validation

I was able to validate changes by going to WordPress dashboard and making sure edits weren't available. For the permission setting I verified by checking permissions were changed to only read on the file by using command ls -l in the directory folder /var/www/html and looking at wp-config.php file for r-r-r on permission.

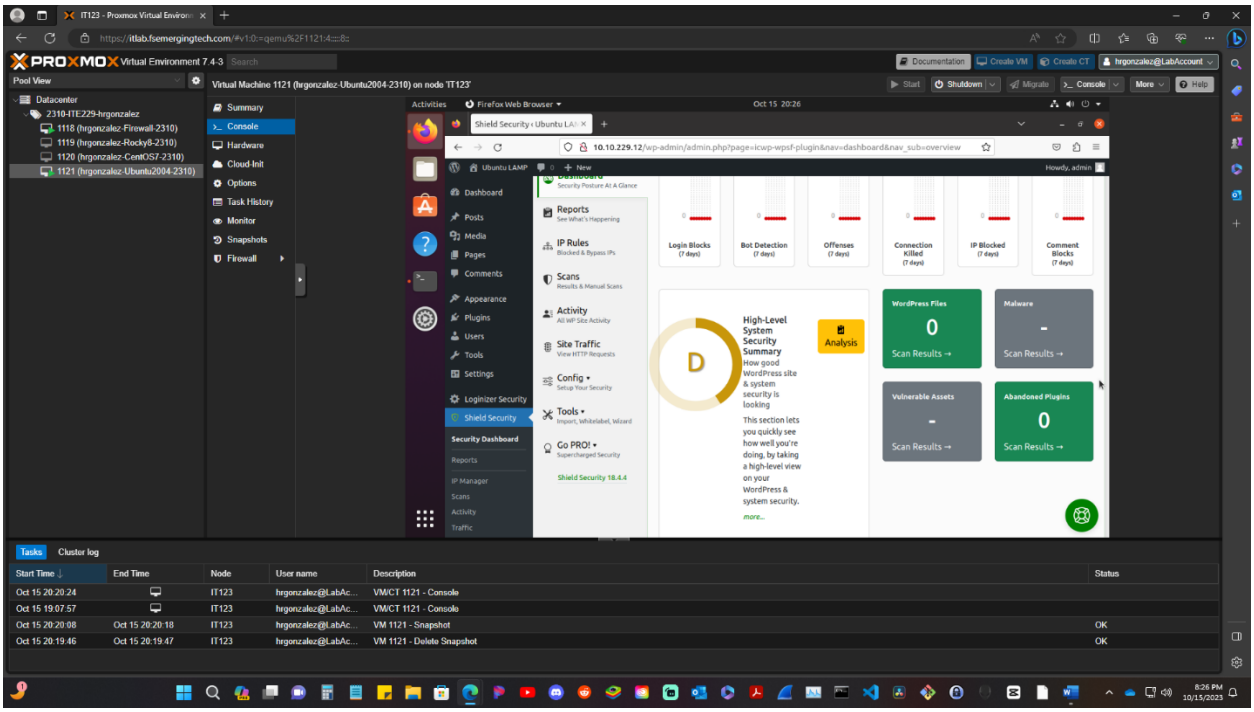
Firewall (Shield)

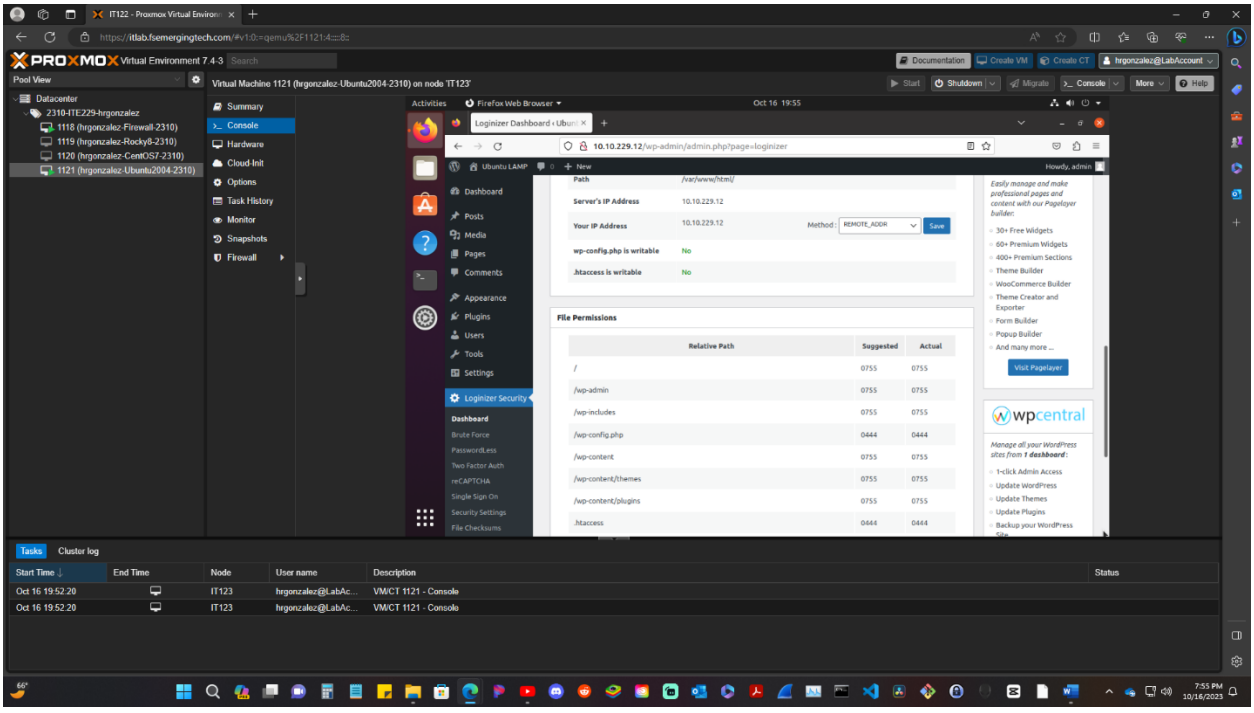
Before Changes





After Changes





Vulnerability

A WordPress website without a firewall is vulnerable to unauthorized access, brute force assaults, SQL injection, cross-site scripting, malware injections, DDoS attacks, and the exploitation of known or zero-day vulnerabilities. Data theft and file inclusion assaults are also major issues. To mitigate these dangers, it is critical to use a web application firewall (WAF), a security plugin, and to keep WordPress, themes, and plugins up to date. These safeguards are crucial in defending the site from a wide range of security threats.

Configuration

WordPress hardening website recommends a couple of options for firewalls. We went with Shield Security because is a great plugin firewall for WordPress and it has a great reputation, it's also user friendly. To add this firewall plugin to WordPress all we had to do was head to plugins and select add a new plugin in WordPress dashboard. After, we searched for Shield on the search bar and select Shield Security. We activate it and then followed the installation wizard. We enabled logging to be able to see changes made on WordPress. The great thing is that Shield will automatically scan security level for the WordPress site and give it a letter rating. When we install Shield our letter rating was E and after we harden the system it went up to D. Now other changes can be made in the future to secure the WordPress site even more and bring the rating higher.

We went and install a plugin call Loginizer. We got Loginizer by doing the same steps as above but with Loginizer on search bar. The beauty of this plugin is that it basically shows you on the Dashboard the files that need hardening for WordPress and their recommended permissions. To be able to get the end results we got we just needed to harden the system normally with the exception of the .htaccess file secured. To get this file secured we needed to enter command:

sudo chmod 0444 .htaccess this will change .htaccess security file to read only for everyone.

Validation

By Shield Security changing from E to D it gave me confirmation that Shield Security was working properly and monitoring changes. As far as Loginizer with us making the changes needed during hardening process and securing .htaccess it confirmed everyhting matched with it's recommendation (letter went fromm red to green or white).

Conclusion

The security methodology used to reinforce our WordPress website through a defense-in-depth strategy has shown to be a solid and versatile approach to protecting our digital assets. This technique is based on the idea of deploying many layers of security mechanisms, similar to how we secure our homes. This starts with the fundamental yet critical practice of continuously updating both the WordPress core and all plugins, which serves as the digital equivalent of guarding one's front door.

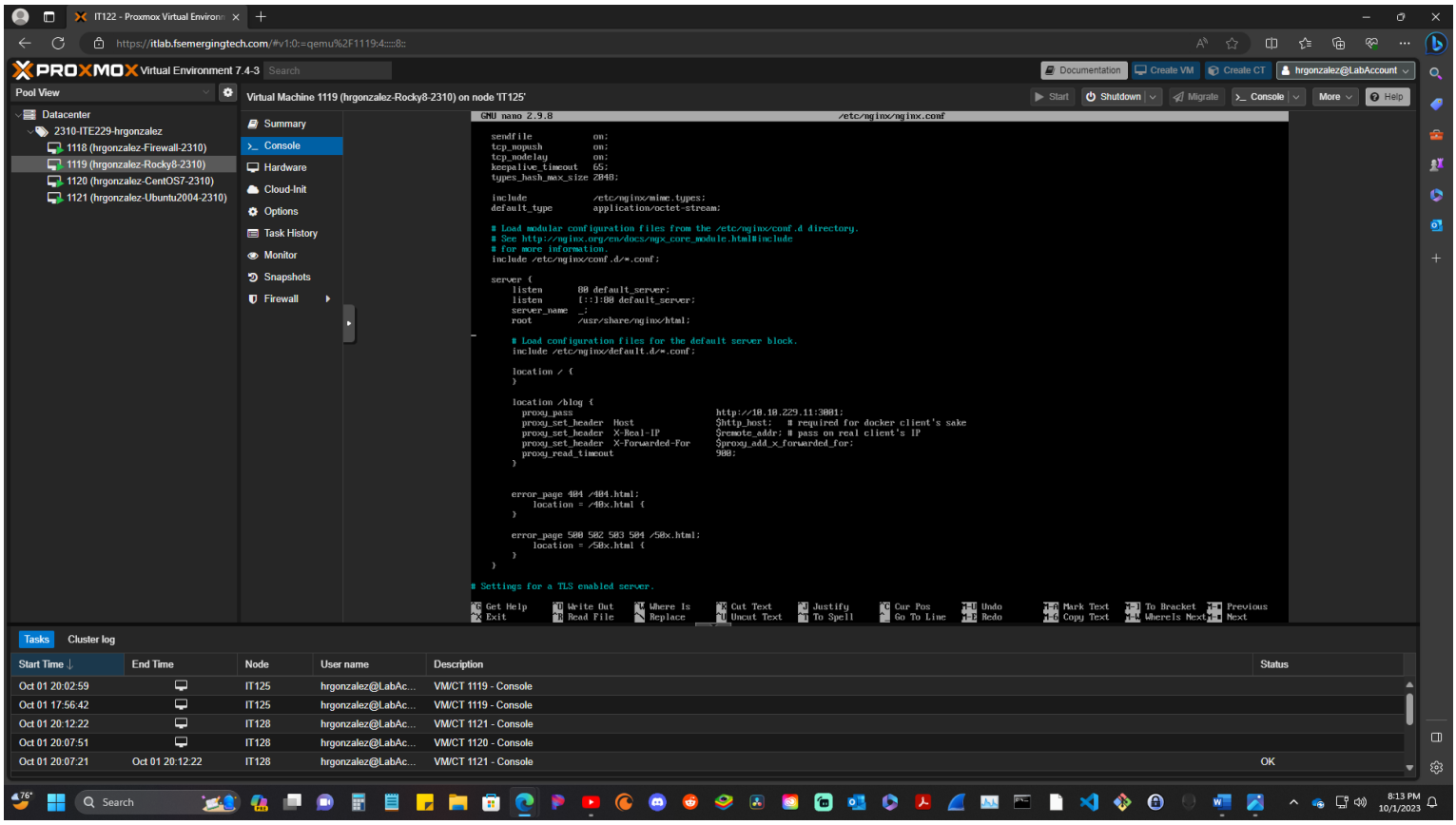
We acknowledged the crucial need of keeping regular backups, which act as our digital insurance policy, protecting our data from potential catastrophes or cyber threats. The use of strong, unique passwords, in conjunction with the adoption of two-factor authentication, adds an extra layer of security, similar to installing a powerful deadbolt lock on our front door. This strategy dramatically improves our security against illegal access. The thorough monitoring of file permissions strengthens our security stance even further, ensuring that illegal access is effectively prevented. The Shield Security plugin, functioning as our digital sentinel, diligently monitors incoming and outgoing traffic, logs any changes, and keeps an overall record of the security state of our WordPress site.

This security methodology is not limited to WordPress, its adaptability may be expanded to various systems and digital environments with ease. The essential concepts of this method remain same whether securing a different content management system, an e-commerce platform, a business infrastructure, or any other digital infrastructure. A solid defense is formed by layering security measures, guaranteeing constant software and system updates, and adopting effective authentication techniques. Furthermore, file permission management and the integration of monitoring systems provide additional levels of protection, guaranteeing that unwanted access is rigorously guarded against.

The defense-in-depth concept emerges as a guiding light for security best practices in our interconnected and ever-changing technological world. By applying this methodology to additional systems, we not only protect our data but also adhere to the fundamental principles of cybersecurity, consisting of Confidentiality, Integrity, and Availability.

Appendix A

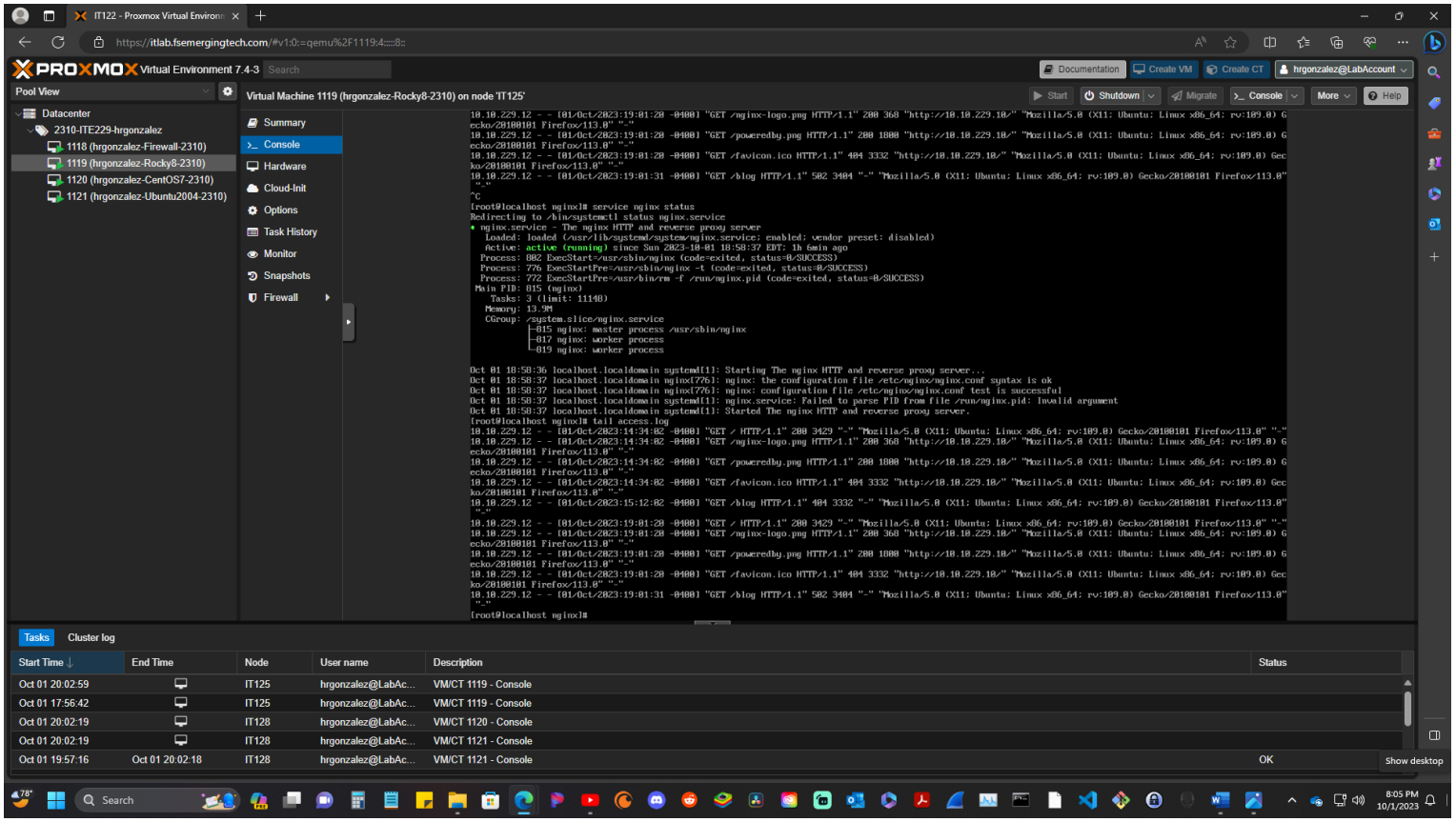
NginX Config File



Appendix B

NginX Access Log File

Tail the NginX log file on Rocky8 machine. 1st run your service nginx status command. Next, run command prompt cd /var/log/nginx. Then use command prompt tail access.log.



NginX Error Log File

Tail the NginX log file on Rocky8 machine. 1st run your service nginx status command. Next, run command prompt cd /var/log/nginx. Then use command prompt tail error.log