**Name of Title:** Learning Nginx

**Video Name:** Configuring a virtual host Part 1

**Estimated Length:**

**Author Name:** Michael Jenkins

**Chapter\_Section\_Video:**

**Video Objective:**

At the end of this video the learner will be able to create a basic server configuration file.

**Introductory Statement:**

Type your introductory statement here.

**Speaking Points:**

1. Create the conf file in conf.d
2. Enter the minimum viable configuration
3. Test the configuration
4. Create the directory for the site and set the permissions on the directory
5. Upload content to the directory and test access to the site.

**Script:**

Let’s create our first server configuration. My development VM is already up and running and I’m connected to it as the default user.

Since we’ll need elevated privileges for many of the commands we’ll be running, I’ll go ahead and use `sudo su -` to substitute my user for the root user.

sudo su -

First and foremost, let’s get rid of that default site. We know nginx is working fine so we don’t need this configuration anymore.

We’ll find the default configuration in /etc/nginx/sites-enabled.

ls -l /etc/nginx/site-enabled

Because the default configuration is linked to a file in sites-available, we can remove that link with the unlink command:

unlink /etc/nginx/sites-enabled/default

ls -l /etc/nginx/sites-enabled/

If you want to make sure the default file that was linked to is still there, you can check the contents of the sites-available directory:

ls -l /etc/nginx/sites-available/

Now we need to create the configuration file that will hold the configuration for our site. We’ll be storing our configurations in /etc/nginx/conf.d.

We’ll be setting up a demo site for a veterinary clinic called wisdom pet medicine. And our demo site will be called wisdompetmed.local.

I’ll use vim to edit a file named wisdompetmed.local.conf in the conf.d directory:

vim /etc/nginx/conf.d/wisdompetmed.local.conf

It's good practice to name the vhost file after the site to keep things organized. Since a single nginx server can run multiple sites, naming the configuration file the same as the site makes them easier to manage.

It's also important that we use the .conf extension so that the vhost gets picked up by nginx when we reload configuration. If there’s ever a vhost that we don’t want to use, we can either remove the file or just rename it so .conf isn’t the file extension.

Inside the file, i’ll enter insert mode and start the vhost with the server directive and and a set of opening and closing curly braces to define the server block.

server {

}

Now we can add a listen directive to tell nginx which port to listen on for requests to serve the site. The default port for web servers to listen on is 80 so we’ll use that here:

listen 80;

We also need to add the root directory where our site will be served from. This defines the path to the actual HTML, CSS, and Javascript that will make up our site.

It's good practice to give the root directory the same name as the site so I’ll enter:

root /var/www/wisdompetmed.local

Ok now we can save this file and check out the configuration. To save and exit a file in vim, you press escape to exit insert mode and then type colon w q to write the file and then quit.

cat /etc/nginx/conf.d/wisdompetmed.local.conf

There’s out vhost. Now let’s use the nginx command with the -t switch to test the configuration.

nginx -t

And if all went well, we should see output from nginx stating that the syntax is OK and the test was successful.

Now we need to reload the configuration into the running server. Do you remember the command for that? We’ll use the systemctl command and tell it to reload nginx:

systemctl reload nginx

If all goes well with this step we won’t see any output here but we can confirm with the systemctl status command:

systemctl status nginx

And at the very end of this output, we see that nginx was successfully reloaded.

Now let’s create the directory that we used in the server config.

mkdir -p /var/www/wisdompetmed.local

Just to make sure our configuration is correct, let’s put a test file in the directory. We can put a simple one in there just to confirm all is well.

echo ‘site coming soon’ > /var/www/wisdompetmed.local/index.html

Now let’s take a look at the site to see the response.

open browser and go to http://192.168.0.3/

Nice! Well, our site isn’t much to look at but seeing the test message gives some confidence that once we put content in the root directory, it’ll be served as expected.

In the next couple lessons, we’ll add a few more directives to our vhost config and then add the files for the website.

**Conclusion:**

Type your conclusion statement here.

**Script and Media:**

Break the script up into parts and align it with any media (slides, web, CLI, etc.)

| **Part** | **Script** | **Media** |
| --- | --- | --- |
|  |  |  |

**Exercise Files:**

wisdompetmed.local.conf

**Basement:**

or ssh vagrant@192.168.0.3

At this point, we have nginx installed, we know how to control the server with the command line interface, and we have a good handle on how to configure it.

Now l