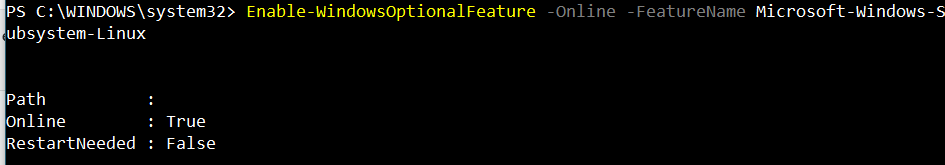
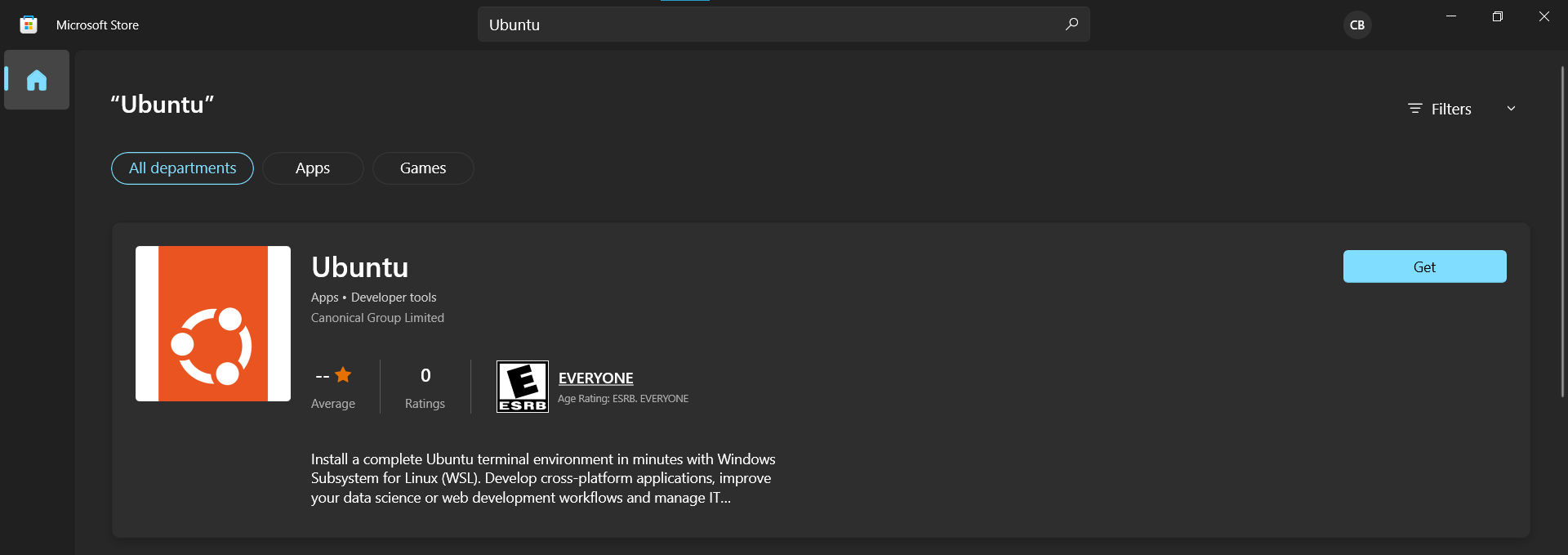
**Ansible Playbook**

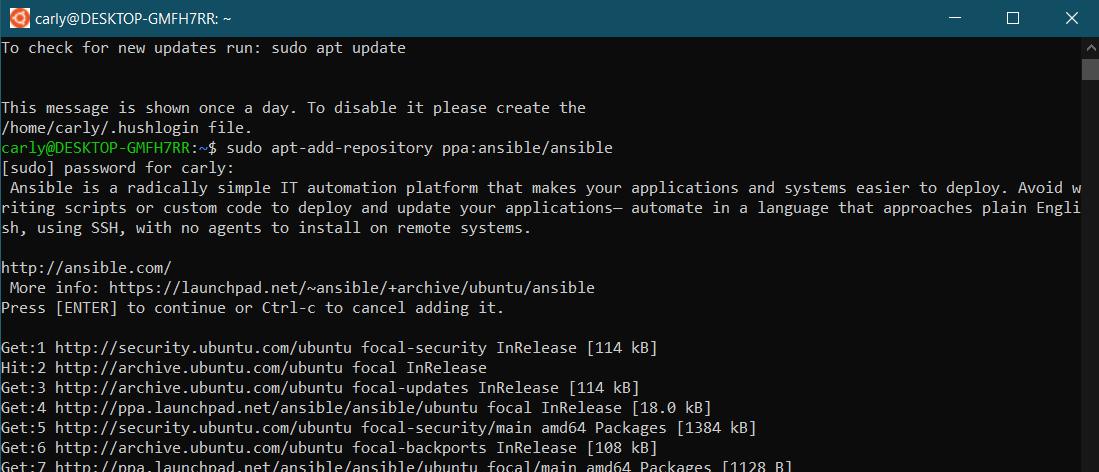
Fisrt we need to install Ansible in the machine that is going to be the host node. In this case, we installed ansible in Windows OS, to achieve this it was necessary to enable the Subsystem for Linux running the next command:

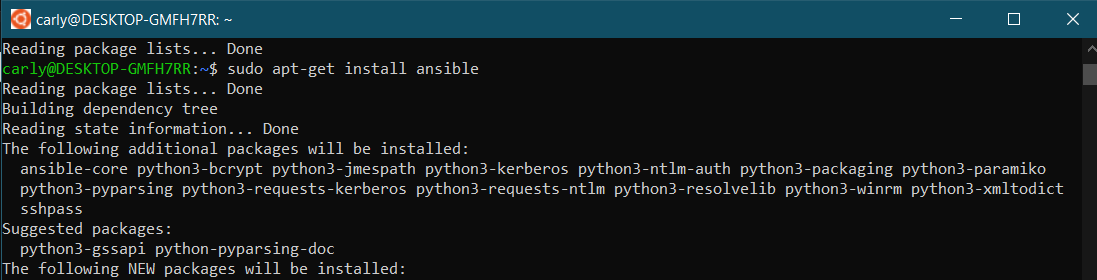


Then, we have to install Ubuntu distribution from Microsoft Store:

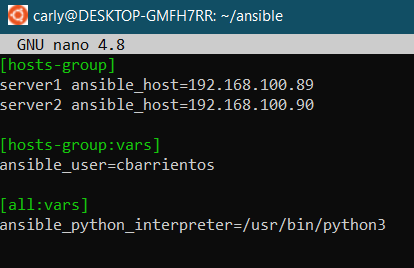


Once installed, we configure our user and password, and start downloading and installing Ansible in our Linux distribution:

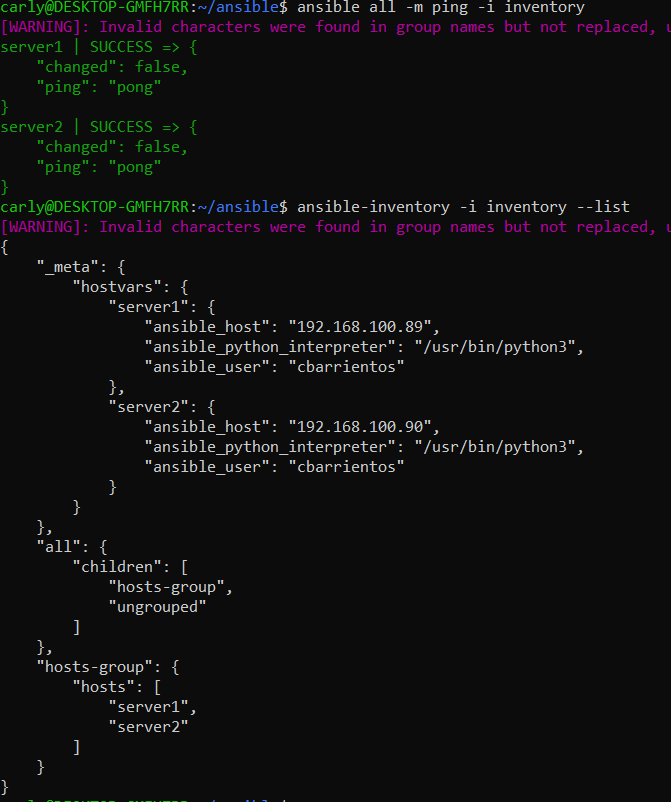




In this case, we used a custom inventory for our Ansible, in the next image we can see its content. The first host corresponds to an Ubuntu virtual machine and the second one to a Fedora virtual machine. We set the user for both machine using vars.

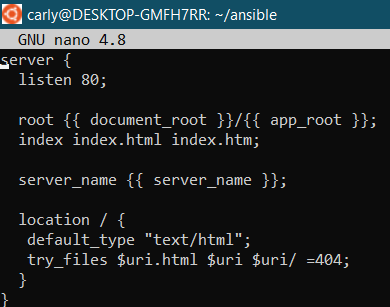


Ansible uses ssh to communicate to the guest’s machines, so it is necessary to configure it. In the next image we can see that it is already configured and working, the guest machines specified in the inventory are responding to the ping:



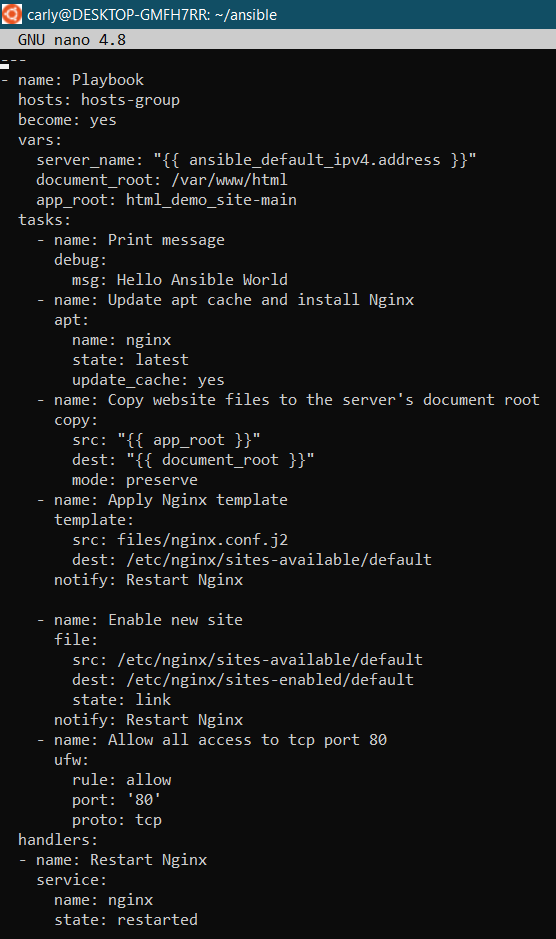
For the next exercise, we deploy a static HTML website using Nginx, to achieve this we created a playbook to automate this process. For demonstration we used a static HTML website already created, we download the project and unzip it.

Then we created a template for Nginx’s configuration. The content can be seen in the next image, it contains 3 variables that we are going to define when we create the playbook.

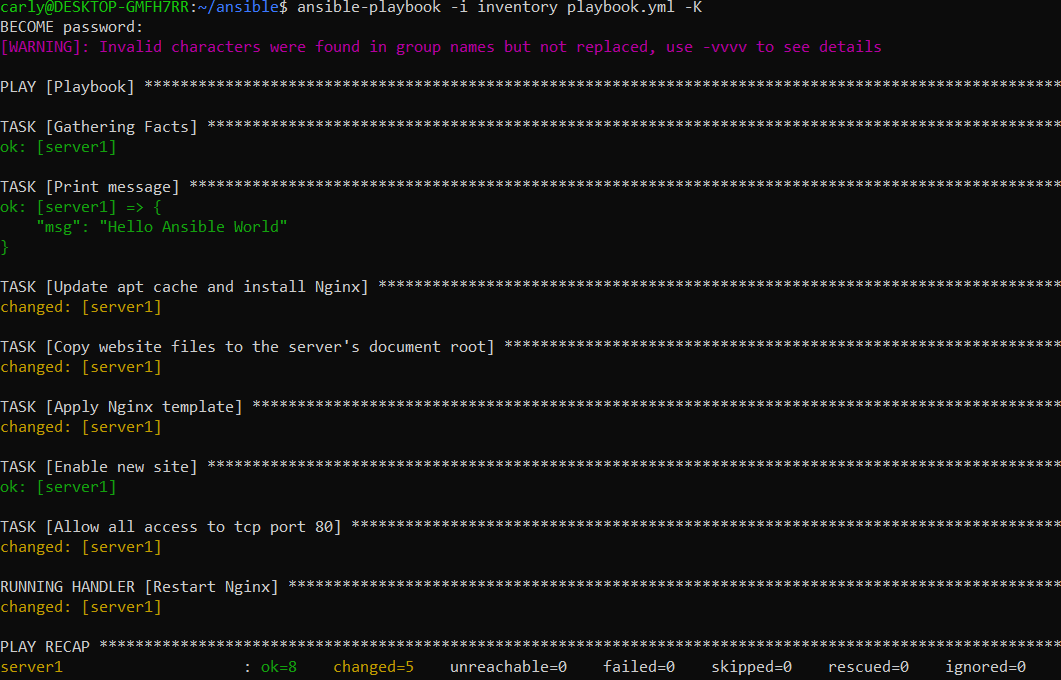


The content of the playbook can be seen in the next image. First, we define the group hosts that we want to deploy, the directive become that tells Ansible to run as root user. In the var section we have the values that we are going to use to configure Nginx, the server name is going to be the IP address of the host machine, the document root by default and in the app root we specify the folder where our project is located.

The first task is just a little test to check that everything is working, we only display a message. The next task updates the apt cache and installs nginx package on the remote node (in this case only ubuntu host). The third task copies the website files to the server’s document root specified in vars. Then we’ll apply the Nginx template that will configure the web server to host your static HTML file. After the configuration file is set at /etc/nginx/sites-available, we’ll create a symbolic link to that file inside /etc/nginx-sites-enabled and notify the Nginx service for a posterior restart. The last task enables tcp access on port 80. Finally we set up the restart Nginx handler that is going to run after all the previous tasks.



We run the next commando to execute the playbook on our Ubuntu host machine, including the -K argument provides the remote user sudo password when prompted by Ansible:



Once the playbook is finished we can access the static HTML webpage using our Ubuntu virtual machine IP and the port 80:

