**Install LAMP stack using Ansible playbook**

## **1- Project Description**

For testing purposes at work, we usually need to build a LAMP stack server, this is a manual effort (Here is the link for the manual [steps](https://docs.google.com/document/d/1N-m4ArXbGH2_C_WfZFgrWKcwXUsXlgMb8uYY5BNXJpI/edit?usp=sharing)) and we are tasked to create a playbook that will be used to automate the process.

Here, we will use only 3 servers for demonstration, as our computer cannot support 5 servers. One server will act as the master node, and the other two will function as worker nodes.

So, let's go ahead and write our playbook.

## **2- What will our playbook do?**

**Connect to the Remote host and execute the following tasks**

* Install all necessary packages like Apache (Httpd), MariaDB and PHP.
* Installing a firewall and enabling HTTP services
* Start the Apache HTTPd web server.
* Start the MariaDB server
* Download a Sample PHP page from the remote URL
* Access the website we have built by accessing the URL

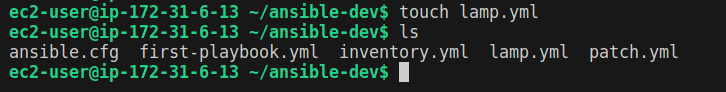


## **3- Writing the playbook**

Connect remotely to the master

Create a file **lamp.yml** inside **ansible-dev** folder and paste the code below inside.

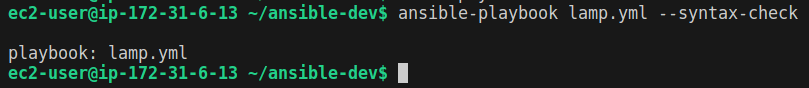
touch lamp.yml



| --- - name: Setting up LAMP Website  hosts: databaseservers  become: yes  vars:  ansible\_hostname: 0.0.0.0  tasks:  - name: Update package repositories for Debian/Ubuntu  apt:  update\_cache: yes  when: "ansible\_os\_family == 'Debian' or ansible\_os\_family == 'Ubuntu'"   - name: Update package repositories for Amazon Linux  yum:  name: '\*'  state: latest  when: "ansible\_os\_family == 'RedHat' or ansible\_os\_family == 'Amazon'"    - name: Determine package name for MariaDB on Debian/Ubuntu  set\_fact:  mariadb\_package\_name: "{{ 'mariadb-server' if ansible\_distribution == 'Debian' else 'mysql-server' }}"  when: "ansible\_os\_family == 'Debian' or ansible\_os\_family == 'Ubuntu'"  - name: Install required packages for Debian/Ubuntu  apt:  name:  - apache2  - "{{ mariadb\_package\_name }}"  - php  - libapache2-mod-php  - php-mysql  state: present  when: "ansible\_os\_family == 'Debian' or ansible\_os\_family == 'Ubuntu'"   - name: Install required packages for Amazon Linux  yum:  name:  - httpd  - mariadb-server  - php  - php-mysql  state: present  when: "ansible\_os\_family == 'RedHat' or ansible\_os\_family == 'Amazon'"   - name: Ensure firewalld is installed and enabled (for Amazon Linux)  package:  name: firewalld  state: present  when: "ansible\_os\_family == 'RedHat' or ansible\_os\_family == 'Amazon'"  - name: Start and enable Apache for Amazon Linux  service:  name: httpd  enabled: yes  state: started  when: "ansible\_os\_family == 'RedHat' or ansible\_os\_family == 'Amazon'"   - name: Start and enable MariaDB for Amazon Linux  service:  name: mariadb  enabled: yes  state: started  when: "ansible\_os\_family == 'RedHat' or ansible\_os\_family == 'Amazon'"   - name: Install unzip on Debian/Ubuntu  apt:  name: unzip  state: present  when: ansible\_os\_family == "Debian" or ansible\_os\_family == "Ubuntu"   - name: Install unzip on RedHat/Amazon Linux  yum:  name: unzip  state: present  when: ansible\_os\_family == "RedHat" or ansible\_os\_family == "Amazon"   - name: Download startboostrap template  get\_url:  url: "https://github.com/startbootstrap/startbootstrap-business-frontpage/archive/gh-pages.zip"  dest: /tmp/startbootstrap-business-frontpage-gh-pages.zip  mode: 0755   - name: Unzip the downloaded template  unarchive:  src: /tmp/startbootstrap-business-frontpage-gh-pages.zip  dest: /var/www/html/  remote\_src: yes   - name: Rename extracted folder  command:  cmd: mv /var/www/html/startbootstrap-business-frontpage-gh-pages /var/www/html/test  creates: /var/www/html/test |
| --- |

save and exit. Then do the syntax check

| ansible-playbook lamp.yml --syntax-check |
| --- |

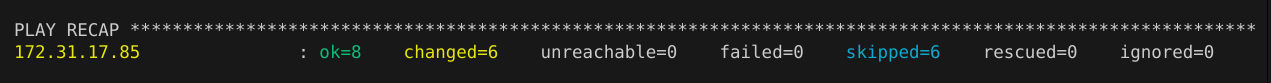


After validating the syntax, let's go ahead and run our playbook.

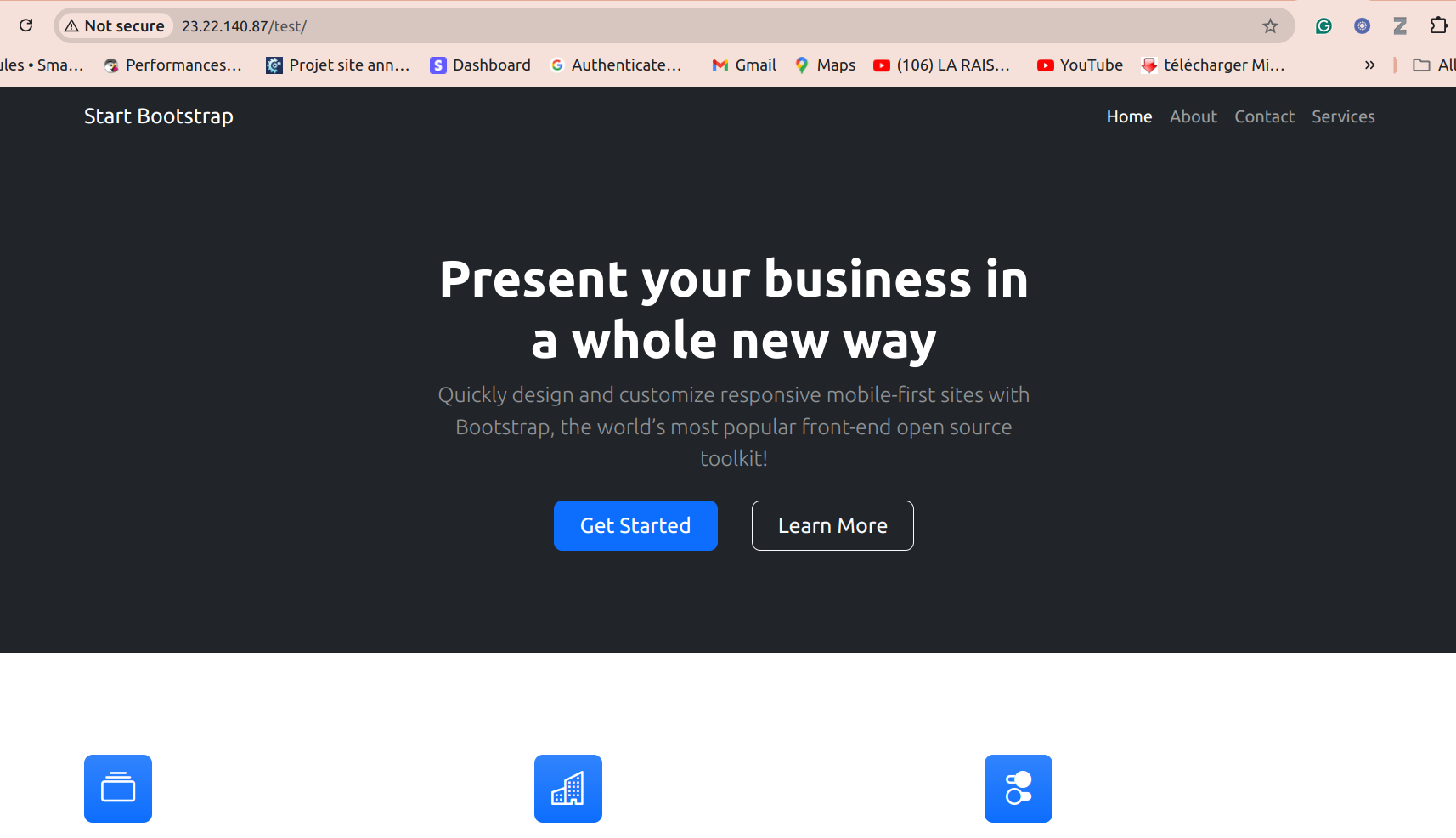
To run the playbook, run the code below:

| ansible-playbook lamp.yml |
| --- |

When the playbook is done running, you will have the following recap.



Visiting the IP addresses of node2 **(node2-public-ip/test),** you should see the following page.



Now that the LAMP stack has been installed using the Ansible playbook, let's go ahead and break down the Ansible playbook for better understanding.

## **5- Explaining the playbook.**

| --- - name: Setting up LAMP Website  user: root  hosts: databaseservers  become: yes |
| --- |

Above is the first segment of our playbook code.

* Line 2 describes the procedure.
* On line 3 we are specifying the user.
* Line 4 sets the host's severs. Note that the host servers are defined in our inventory file
* line 5 configures sudo access.

The rest of the code is self-explanatory, as each code block has a clear explanation of what it's doing.

## **6- Conclusion.**

As you can see, using Ansible to facilitate a task like this one reduces the stress of logging into multiple servers and doing the installation manually.

## **Read more**

<https://docs.ansible.com/ansible/latest/cli/ansible-playbook.html>