

DEPLOYMENT OF WORDPRESS ENVIRONMENT

Steps to Perform:

- 1) Establish configuration management master connectivity with WordPress server
- 2) Validate connectivity from master to slave machine
- 3) Prepare IaC scripts to install WordPress and its dependent components
- 4) Execute scripts to perform installation of complete WordPress environment
- 5) Validate installation using the public IP of VM by accessing WordPress application

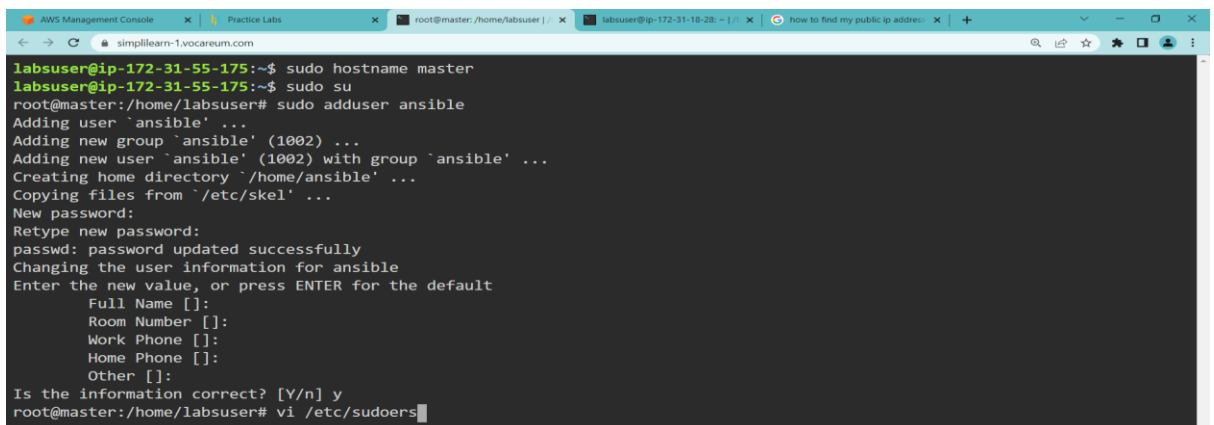
STEP-1: Establish configuration management master connectivity with WordPress server

- 1) Do these in the configuration management master server as well as node, i.e., wordpress server:

`sudo apt update` (updates the machine)

`sudo su` (switch to root user)

`sudo adduser ansible` (create an ansible user)
enter password and other credentials for ansible user



```
labsuser@ip-172-31-55-175:~$ sudo hostname master
labsuser@ip-172-31-55-175:~$ sudo su
root@master:/home/labsuser# sudo adduser ansible
Adding user `ansible' ...
Adding new group `ansible' (1002) ...
Adding new user `ansible' (1002) with group `ansible' ...
Creating home directory `/home/ansible' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for ansible
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n] y
root@master:/home/labsuser# vi /etc/sudoers
```

```
AWS Management Console x Practice Labs x root@master:/home/labsuser [ x root@wordpress-server:/home/ x +
simplilearn-3.vocareum.com
root@wordpress-server:/home/labsuser# sudo adduser ansible
Adding user `ansible' ...
Adding new group `ansible' (1002) ...
Adding new user `ansible' (1002) with group `ansible' ...
Creating home directory `/home/ansible' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for ansible
Enter the new value, or press ENTER for the default
  Full Name []:
    Room Number []:
    Work Phone []:
    Home Phone []:
    Other []:
Is the information correct? [Y/n] y
root@wordpress-server:/home/labsuser# vi /etc/sudoers
```

`vi /etc/sudoers`

(inside the sudoers file in line 45 under user privilege specification add this line to permit passwordless super user rights to ansible user)

ansible ALL=(ALL:ALL) NOPASSWD: ALL

```
# User privilege specification
root    ALL=(ALL:ALL) ALL
ansible ALL=(ALL:ALL) NOPASSWD: ALL
# Members of the admin group may gain root privileges
%admin   ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:

:wq
```

`vi /etc/ssh/sshd_config`

(inside the sshd_config file in line 57 change PasswordAuthentication from no to yes)

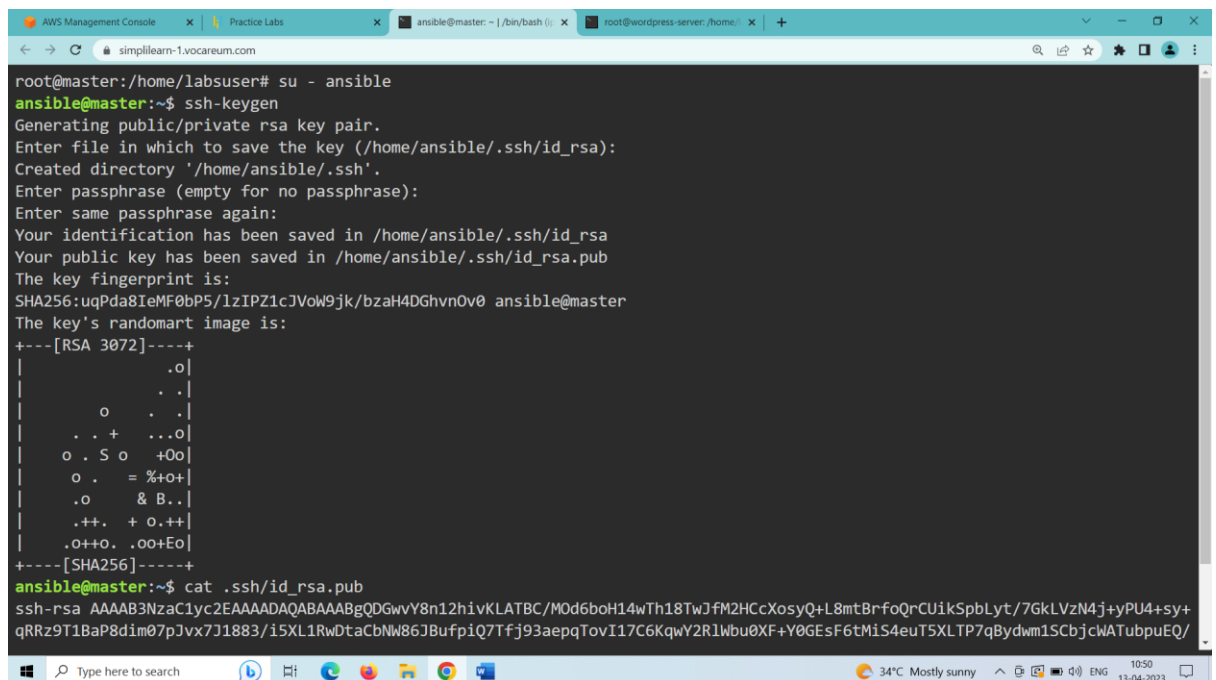
```
# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication yes
#PermitEmptyPasswords no
```

`systemctl restart sshd` (to restart sshd)

- 2) Only in the master server:
switch to ansible user and generate keys for ssh login to wordpress server
`su - ansible` (switches to ansible user)

`ssh-keygen` (generates public key for ansible user)

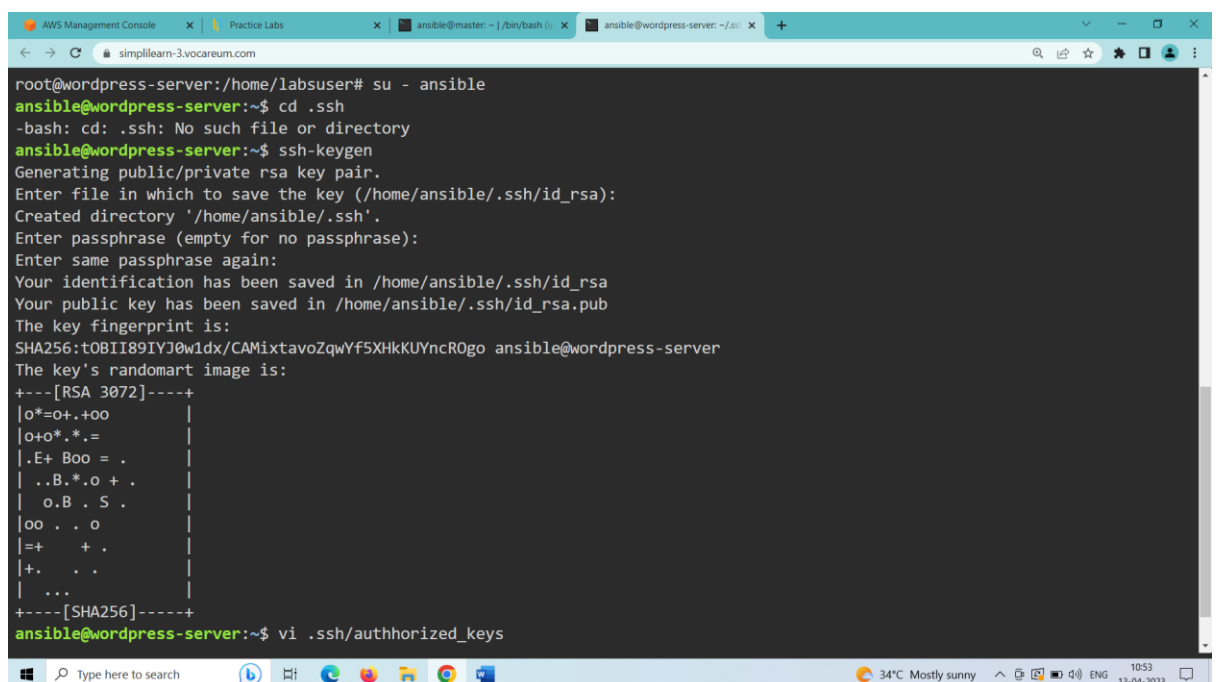
`cat .ssh/id_rsa.pub` (copy this key into `authorized_keys` folder inside the `.ssh` folder in wordpress server)



```
root@master:/home/labsuser# su - ansible
ansible@master:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ansible/.ssh/id_rsa):
Created directory '/home/ansible/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ansible/.ssh/id_rsa
Your public key has been saved in /home/ansible/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:uqPda8IeMF0bP5/1zIPZ1cJV0W9jk/bzaH4DGhvn0v0 ansible@master
The key's randomart image is:
+---[RSA 3072]-----+
|
| .o|
|  . .|
|   o . .|
|  . . + ..o|
| o . S o  +Oo|
| o . = %+o+|
| .o & B..|
| .++ . + O.++|
| .o++o. .oo+EO|
+---[SHA256]-----+
ansible@master:~$ cat .ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGDGwvY8n12hivKLATBC/Mod6boH14wTh18TwJfM2HCcXosyQ+L8mtBrfoQrCUikSpbLyt/7GkLVzN4j+yPU4+sy+qRRz9T1BaP8dim07pJvx7J1883/i5XL1RwDtaCbNW86JBuFpiQ7Tfj93aepqTovI17C6KqwY2R1Wbu0XF+Y0GESF6tMiS4euT5XLP7qBydwm1SCbjcWATubpuEQ/
```

3) Inside the wordpress server:
`su – ansible` (switch to ansible user)

`ssh-keygen` (to create a `.ssh` folder in ansible home directory)



```
root@wordpress-server:/home/labsuser# su - ansible
ansible@wordpress-server:~$ cd .ssh
-bash: cd: .ssh: No such file or directory
ansible@wordpress-server:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ansible/.ssh/id_rsa):
Created directory '/home/ansible/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ansible/.ssh/id_rsa
Your public key has been saved in /home/ansible/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:tOBII89IYJ0w1dx/CAMixtavoZqwYf5XHkKUYncR0go ansible@wordpress-server
The key's randomart image is:
+---[RSA 3072]-----+
|o*=O+.+oo|
|o+o*.*.=|
|.E+ Boo = .|
|..B.*.o + .|
|o.B . S .|
|oo . . o|
|=+ + .|
|+. . .|
|...|
+---[SHA256]-----+
ansible@wordpress-server:~$ vi .ssh/authorized_keys
```

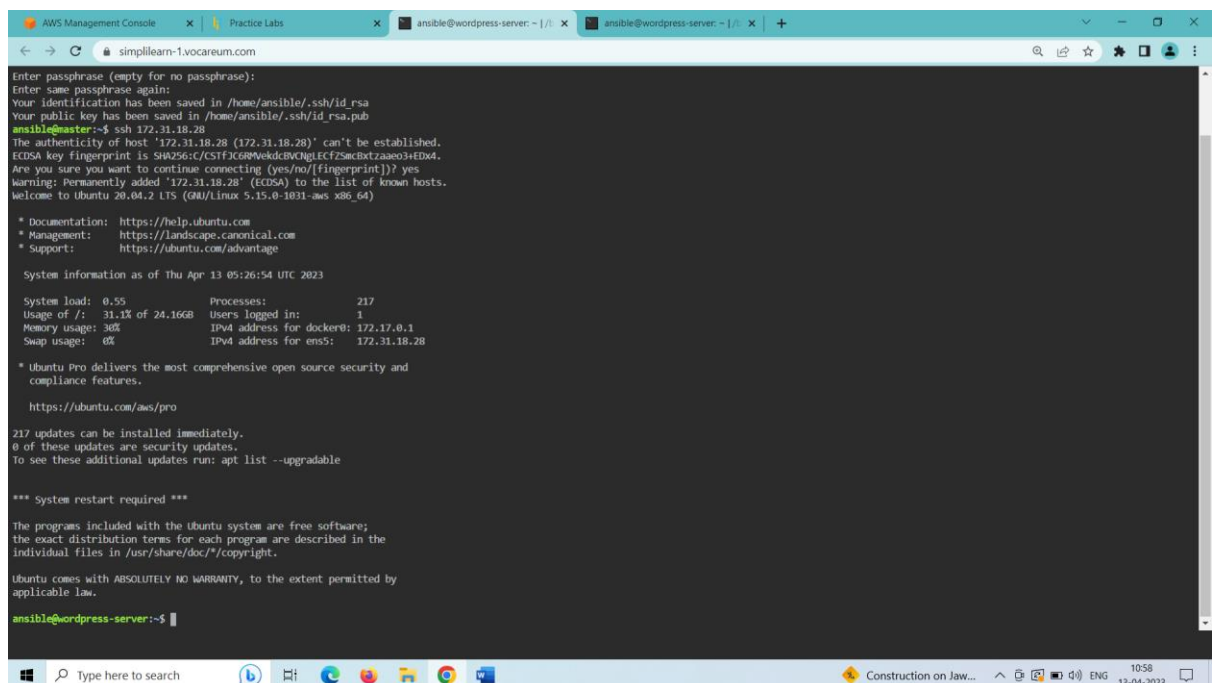
`vi .ssh/authorized_keys` (paste the key generated from master server to wordpress server in this folder)

STEP-2: Validate connectivity from master to slave machine

Inside the master server:

(check the connectivity from master to wordpress server)

`ssh <private-ip-of-wp-server>` (login to the wordpress server)



```
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ansible/.ssh/id_rsa
Your public key has been saved in /home/ansible/.ssh/id_rsa.pub
ansible@master:~$ ssh 172.31.18.28
The authenticity of host '172.31.18.28 (172.31.18.28)' can't be established.
ECDSA key fingerprint is SHA256:C/CSfjC6WwkdC8VcNgLECFZsacBxtzaao3+EDd4.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.31.18.28' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.15.0-1031-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

System information as of Thu Apr 13 05:26:54 UTC 2023

System load: 0.55      Processes:            217
Usage of /:  31.1% of 24.16GB   Users logged in:     1
Memory usage: 36%        IPv4 address for docker0: 172.17.0.1
Swap usage:  0%          IPv4 address for ens5:  172.31.18.28

 * Ubuntu Pro delivers the most comprehensive open source security and
   compliance features.
   https://ubuntu.com/aws/pro

217 updates can be installed immediately.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable

*** System restart required ***

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

ansible@wordpress-server:~$
```

`exit` (to exit the connection from wordpress server and login back to master server)



```
ansible@wordpress-server:~$ exit
logout
Connection to 172.31.18.28 closed.
ansible@master:~$
```

4) Install ansible and add ansible repositories:

`sudo apt-add-repository ppa:ansible/ansible` (installs ansible repositories)

`sudo apt update` (updates the machine)

`sudo apt install ansible -y` (installs ansible)

`sudo vi /etc/ansible/hosts` (inside the hosts file add the private ip of the wordpress server under wpserver host-group)

`[wpserver]`

`<private-ip-of-wp-server>`

(exit the file by using :wq)

```
# - A hostname/ip can be a member of multiple groups
[wpserver]
172.31.18.28
```

- 5) Check the ping from master to slave machine using ansible adhoc command:

`ansible -m ping wpserver`

```
ansible@master:~$ sudo vi /etc/ansible/hosts
ansible@master:~$ ansible -m ping wpserver
172.31.18.28 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
ansible@master:~$
```

Ping successful from master to wordpress server node.

STEP-3: Prepare IaC scripts to install WordPress and its dependent components

- 6) Create a wordpress working directory:

`mkdir wordpress` (creates a working directory named wordpress)

`cd wordpress` (enter into the wordpress directory)

```

ansible@master:~$ mkdir wordpress
ansible@master:~$ ls
wordpress
ansible@master:~$ cd wordpress
ansible@master:~/wordpress$

```

Create and write a playbook inside this folder to install WordPress and its dependent components.

- 7) Create a playbook to write the script to install Wordpress and it dependent components and configure each of the components:

`vi playbook.yml` (creates a playbook named `playbook.yml`)

#####inside the playbook:

```

---
- hosts: wpserver
  become: true
  vars_files:
    - vars/default.yml

  tasks:
    - name: Install prerequisites
      apt: name=aptitude update_cache=yes state=latest force_apt_get=yes
      tags: [ system ]

    - name: Install dependent packages
      apt: name={{ item }} update_cache=yes state=latest
      loop: [ 'apache2', 'mysql-server', 'python3-pymysql', 'php', 'php-mysql',
'libapache2-mod-php' ]
      tags: [ system ]

    - name: Install PHP Extensions
      apt: name={{ item }} update_cache=yes state=latest
      loop: "{{ php_modules }}"
      tags: [ system ]

# Apache Configuration
- name: Create document root
  file:

```

```
path: "/var/www/{{ http_host }}"
state: directory
owner: "www-data"
group: "www-data"
mode: '0755'
tags: [ apache ]
```

```
- name: Set up Apache VirtualHost
template:
  src: "files/apache.conf.j2"
  dest: "/etc/apache2/sites-available/{{ http_conf }}"
notify: Reload Apache
tags: [ apache ]
```

```
- name: Enable rewrite module
shell: /usr/sbin/a2enmod rewrite
notify: Reload Apache
tags: [ apache ]
```

```
- name: Enable new site
shell: /usr/sbin/a2ensite {{ http_conf }}
notify: Reload Apache
tags: [ apache ]
```

```
- name: Disable default Apache site
shell: /usr/sbin/a2dissite 000-default.conf
notify: Restart Apache
tags: [ apache ]
```

MySQL Configuration

```
- name: Set the root password
mysql_user:
  name: root
  password: "{{ mysql_root_password }}"
  login_unix_socket: /var/run/mysqld/mysqld.sock
tags: [ mysql, mysql-root ]
```

```
- name: Remove all anonymous user accounts
mysql_user:
  name: "
```

```
    host_all: yes
    state: absent
    login_user: root
    login_password: "{{ mysql_root_password }}"
    tags: [ mysql ]
```

```
- name: Remove the MySQL test database
  mysql_db:
    name: test
    state: absent
    login_user: root
    login_password: "{{ mysql_root_password }}"
    tags: [ mysql ]
```

```
- name: Creates database for WordPress
  mysql_db:
    name: "{{ mysql_db }}"
    state: present
    login_user: root
    login_password: "{{ mysql_root_password }}"
    tags: [ mysql ]
```

```
- name: Create MySQL user for WordPress
  mysql_user:
    name: "{{ mysql_user }}"
    password: "{{ mysql_password }}"
    priv: "{{ mysql_db }}.*:ALL"
    state: present
    login_user: root
    login_password: "{{ mysql_root_password }}"
    tags: [ mysql ]
```

UFW Configuration

```
- name: "UFW - Allow HTTP on port {{ http_port }}"
  ufw:
    rule: allow
    port: "{{ http_port }}"
    proto: tcp
    tags: [ system ]
```


WordPress Configuration

- name: Download and unpack latest WordPress

unarchive:

src: <https://wordpress.org/latest.tar.gz>

dest: "/var/www/{{ http_host }}"

remote_src: yes

creates: "/var/www/{{ http_host }}/wordpress"

tags: [wordpress]

- name: Set ownership

file:

path: "/var/www/{{ http_host }}"

state: directory

recurse: yes

owner: www-data

group: www-data

tags: [wordpress]

- name: Set permissions for directories

shell: "/usr/bin/find /var/www/{{ http_host }}/wordpress/ -type d -exec

chmod 750 {} \;"

tags: [wordpress]

- name: Set permissions for files

shell: "/usr/bin/find /var/www/{{ http_host }}/wordpress/ -type f -exec

chmod 640 {} \;"

tags: [wordpress]

- name: Set up wp-config

template:

src: "files/wp-config.php.j2"

dest: "/var/www/{{ http_host }}/wordpress/wp-config.php"

tags: [wordpress]

handlers:

- name: Reload Apache

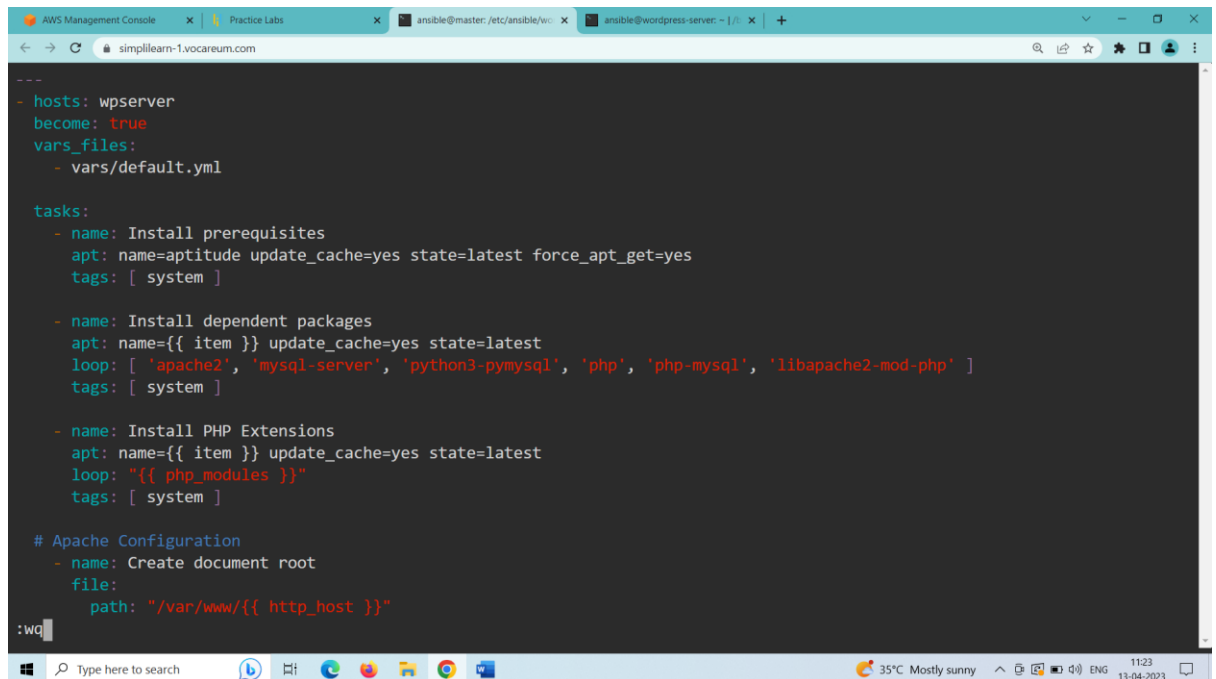
service:

name: apache2

state: reloaded

```
- name: Restart Apache
service:
  name: apache2
  state: restarted
```

(save and quit the file using :wq)

A screenshot of a web browser displaying an Ansible playbook. The browser has multiple tabs open, including 'AWS Management Console', 'Practice Labs', and 'ansible@master: /etc/ansible/vo...'. The address bar shows 'simplilearn-1.vocareum.com'. The playbook content is as follows:

```
---
- hosts: wpserver
  become: true
  vars_files:
    - vars/default.yml

  tasks:
    - name: Install prerequisites
      apt: name=aptitude update_cache=yes state=latest force_apt_get=yes
      tags: [ system ]

    - name: Install dependent packages
      apt: name={{ item }} update_cache=yes state=latest
      loop: [ 'apache2', 'mysql-server', 'python3-pymysql', 'php', 'php-mysql', 'libapache2-mod-php' ]
      tags: [ system ]

    - name: Install PHP Extensions
      apt: name={{ item }} update_cache=yes state=latest
      loop: "{{ php_modules }}"
      tags: [ system ]

  # Apache Configuration
  - name: Create document root
    file:
      path: "/var/www/{{ http_host }}"

:wq
```

The bottom of the screenshot shows a Windows taskbar with a search bar and various application icons. The system tray on the right indicates a temperature of 35°C, mostly sunny weather, and the date 13-04-2023.

PLAYBOOK EXPLANATION:

- The script in the playbook performs **installation** of prerequisites first and the **dependent packages** required for wordpress site.
- Performs installation of **Php Extensions**.
- Performs **Apache Configuration**.
- Performs **MySQL Configuration**.
- Performs **Firewall configuration** to allow connection to Wordpress site.
- Performs **WordPress Configuration**.
- Executes Handlers to reload and restart apache2.

- 8) Create a directory to add jinja templates for standard use whenever a new requirement or client comes in:

mkdir files (creates a directory named files inside the wordpress directory)

cd files (enter into files directory)

```
ansible@master:~/wordpress$ mkdir files
ansible@master:~/wordpress$ ls
files  playbook.yml
ansible@master:~/wordpress$ cd files
ansible@master:~/wordpress/files$ vi apache.conf.j2
```

- 9) `vi apache.conf.j2` (creates an apache jinja template file)

#####inside the apache.conf.j2 file:

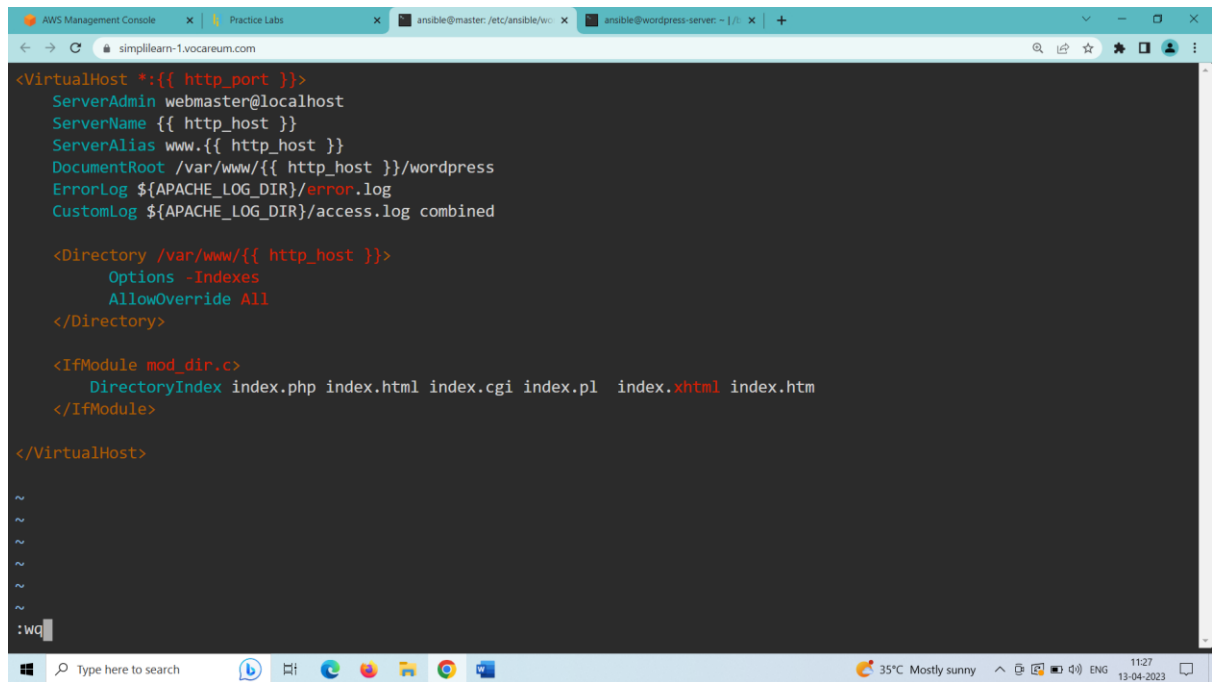
```
<VirtualHost *:{{ http_port }}>
    ServerAdmin webmaster@localhost
    ServerName {{ http_host }}
    ServerAlias www.{{ http_host }}
    DocumentRoot /var/www/{{ http_host }}/wordpress
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
```

```
    <Directory /var/www/{{ http_host }}>
        Options -Indexes
        AllowOverride All
    </Directory>
```

```
    <IfModule mod_dir.c>
        DirectoryIndex index.php index.html index.cgi index.pl index.xhtml
        index.htm
    </IfModule>
```

```
</VirtualHost>
```

(save and quit the file using :wq)



```
<VirtualHost *:{{ http_port }}>
    ServerAdmin webmaster@localhost
    ServerName {{ http_host }}
    ServerAlias www.{{ http_host }}
    DocumentRoot /var/www/{{ http_host }}/wordpress
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    <Directory /var/www/{{ http_host }}>
        Options -Indexes
        AllowOverride All
    </Directory>

    <IfModule mod_dir.c>
        DirectoryIndex index.php index.html index.cgi index.pl index.xhtml index.htm
    </IfModule>
</VirtualHost>
```

10) `vi wp-config.php.j2` (creates a wordpress configuration jinja template file)

#####inside the wp-config.php.j2 file:

```
<?php
/**
 * The base configuration for WordPress
 *
 * The wp-config.php creation script uses this file during the
 * installation. You don't have to use the web site, you can
 * copy this file to "wp-config.php" and fill in the values.
 *
 * This file contains the following configurations:
 *
 * * MySQL settings
 * * Secret keys
 * * Database table prefix
 * * ABSPATH
 *
 * @link https://codex.wordpress.org/Editing_wp-config.php
 *
 * @package WordPress
 */
```

```

// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define( 'DB_NAME', '{{ mysql_db }}' );

/** MySQL database username */
define( 'DB_USER', '{{ mysql_user }}' );

/** MySQL database password */
define( 'DB_PASSWORD', '{{ mysql_password }}' );

/** MySQL hostname */
define( 'DB_HOST', 'localhost' );

/** Database Charset to use in creating database tables. */
define( 'DB_CHARSET', 'utf8' );

/** The Database Collate type. Don't change this if in doubt. */
define( 'DB_COLLATE', '' );

/** Filesystem access */
define( 'FS_METHOD', 'direct' );

/**#@+
 * Authentication Unique Keys and Salts.
 *
 * Change these to different unique phrases!
 * You can generate these using the {@link
https://api.wordpress.org/secret-key/1.1/salt/ WordPress.org secret-key
service}
 * You can change these at any point in time to invalidate all existing
cookies. This will force all users to have to log in again.
 *
 * @since 2.6.0
 */
define( 'AUTH_KEY', '{{ lookup('password', '/dev/null
chars=ascii_letters length=64') }}' );
define( 'SECURE_AUTH_KEY', '{{ lookup('password', '/dev/null
chars=ascii_letters length=64') }}' );
define( 'LOGGED_IN_KEY', '{{ lookup('password', '/dev/null
chars=ascii_letters length=64') }}' );

```

```

define( 'NONCE_KEY',          '{{ lookup('password', '/dev/null
chars=ascii_letters length=64') }}' );
define( 'AUTH_SALT',          '{{ lookup('password', '/dev/null
chars=ascii_letters length=64') }}' );
define( 'SECURE_AUTH_SALT',  '{{ lookup('password', '/dev/null
chars=ascii_letters length=64') }}' );
define( 'LOGGED_IN_SALT',     '{{ lookup('password', '/dev/null
chars=ascii_letters length=64') }}' );
define( 'NONCE_SALT',         '{{ lookup('password', '/dev/null
chars=ascii_letters length=64') }}' );

```

```

/**#@-*/

```

```

/**

```

```

 * WordPress Database Table prefix.

```

```

 *

```

```

 * You can have multiple installations in one database if you give each

```

```

 * a unique prefix. Only numbers, letters, and underscores please!

```

```

 */

```

```

$table_prefix = 'wp_';

```

```

/**

```

```

 * For developers: WordPress debugging mode.

```

```

 *

```

```

 * Change this to true to enable the display of notices during development.

```

```

 * It is strongly recommended that plugin and theme developers use

```

```

WP_DEBUG

```

```

 * in their development environments.

```

```

 *

```

```

 * For information on other constants that can be used for debugging,

```

```

 * visit the Codex.

```

```

 *

```

```

 * @link https://codex.wordpress.org/Debugging_in_WordPress

```

```

 */

```

```

define( 'WP_DEBUG', false );

```

```

/* That's all, stop editing! Happy publishing. */

```

```

/** Absolute path to the WordPress directory. */

```

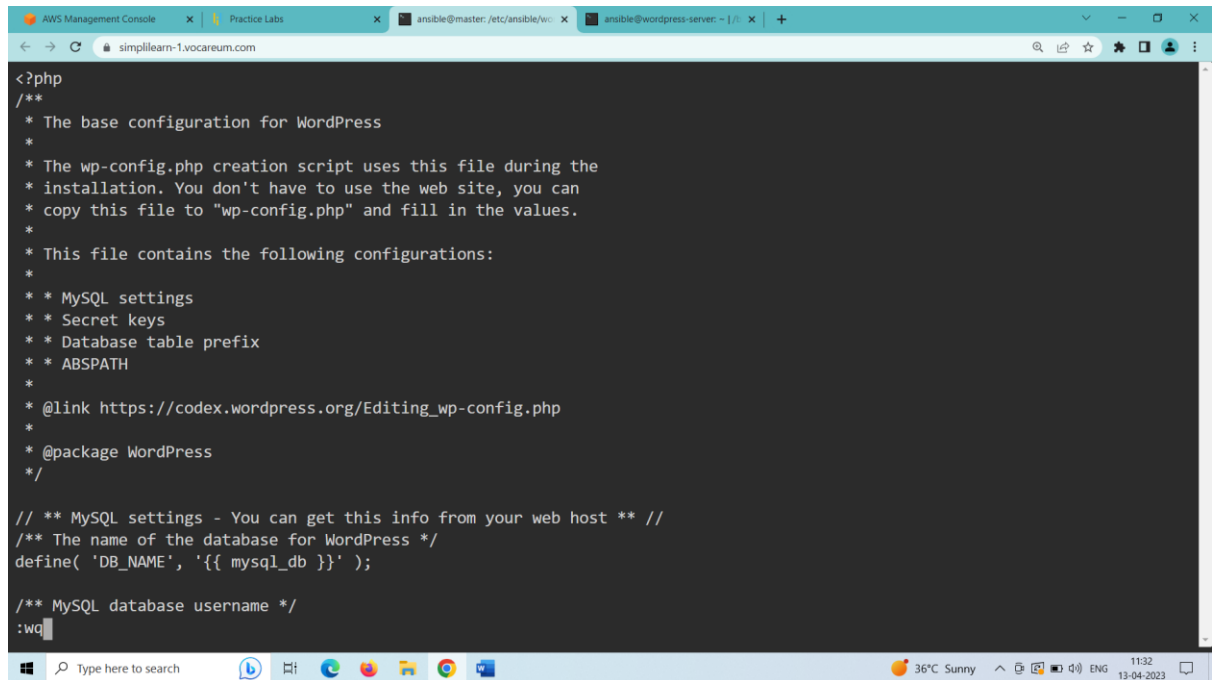
```

if ( ! defined( 'ABSPATH' ) ) {

```

```
define( 'ABSPATH', dirname( __FILE__ ) . '/' );  
}
```

```
/** Sets up WordPress vars and included files. */  
require_once( ABSPATH . 'wp-settings.php' );
```



The screenshot shows a web browser window with the URL `simplilearn-1.vocareum.com`. The page content is the WordPress `wp-config.php` file, which includes comments about the base configuration, MySQL settings, and database name. The code is displayed in a dark-themed editor.

```
<?php  
/**  
 * The base configuration for WordPress  
 *  
 * The wp-config.php creation script uses this file during the  
 * installation. You don't have to use the web site, you can  
 * copy this file to "wp-config.php" and fill in the values.  
 *  
 * This file contains the following configurations:  
 *  
 * * MySQL settings  
 * * Secret keys  
 * * Database table prefix  
 * * ABSPATH  
 *  
 * @link https://codex.wordpress.org/Editing_wp-config.php  
 *  
 * @package WordPress  
 */  
  
// ** MySQL settings - You can get this info from your web host ** //  
/** The name of the database for WordPress */  
define( 'DB_NAME', '{ mysql_db }' );  
  
/** MySQL database username */  
:wq
```

(save and quit the file using :wq)

`cd ..` (to enter into the wordpress directory)



The screenshot shows a terminal window with the following commands and output:

```
ansible@master:~/wordpress/files$ ls  
apache.conf.j2 wp-config.php.j2  
ansible@master:~/wordpress/files$ cd ..  
ansible@master:~/wordpress$ ls  
files playbook.yml  
ansible@master:~/wordpress$
```

- 11) Create a directory named vars in the wordpress working directory to store all the variables for the playbook:

`mkdir vars` (creates a directory named vars)

`cd vars` (enter into the vars directory)



The screenshot shows a terminal window with the following commands and output:

```
ansible@master:~/wordpress$ mkdir vars  
ansible@master:~/wordpress$ cd vars  
ansible@master:~/wordpress/vars$ vi default.yml
```

12) `vi default.yml` (creates a file named default.yml)

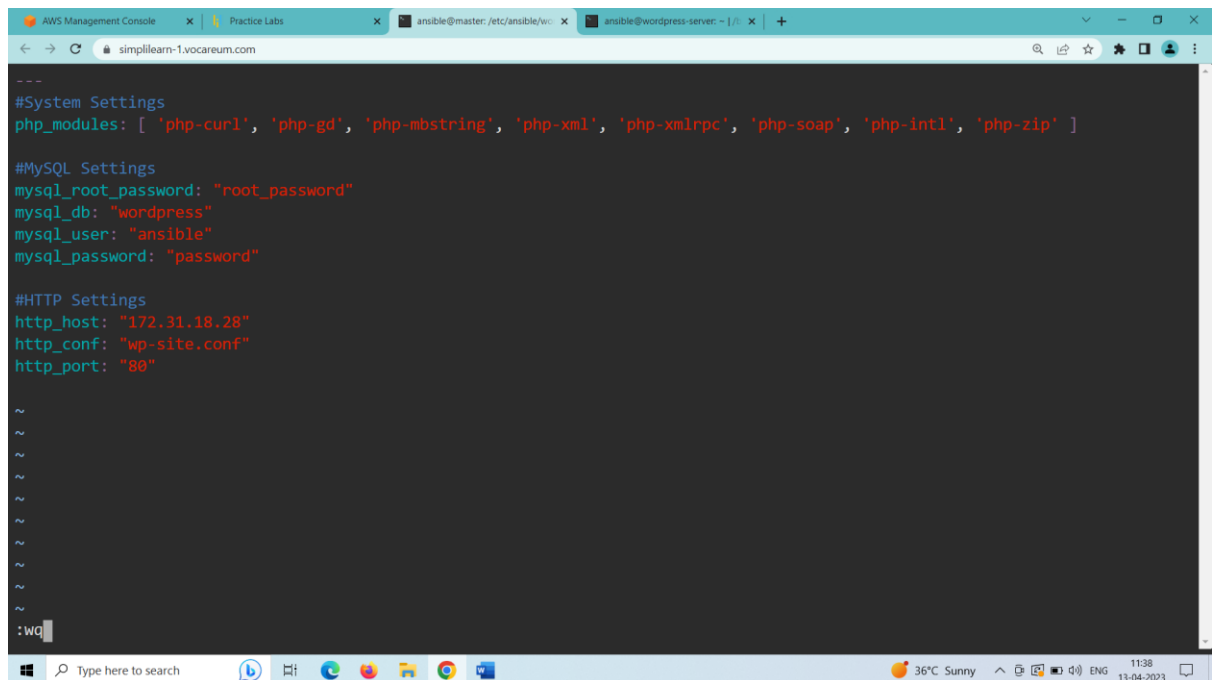
#####inside the default.yml file:

```
---
#System Settings
php_modules: [ 'php-curl', 'php-gd', 'php-mbstring', 'php-xml', 'php-xmlrpc', 'php-soap', 'php-intl', 'php-zip' ]

#MySQL Settings
mysql_root_password: "root_password"
mysql_db: "wordpress"
mysql_user: "ansible"
mysql_password: "password"

#HTTP Settings
http_host: "172.31.18.28"
http_conf: "wp-site.conf"
http_port: "80"
```

(save and quit the file using `:wq`)

A screenshot of a terminal window with a dark background. The terminal shows the contents of the default.yml file, which includes system settings for PHP modules, MySQL database settings, and HTTP settings. The file is edited in the vi editor, and the prompt at the bottom shows the user is in the 'wq' (write and quit) command. The terminal window is titled 'ansible@wordpress-server: ~ | /' and has several tabs open at the top, including 'AWS Management Console', 'Practice Labs', and 'ansible@master: /etc/ansible/wo...'. The bottom of the screen shows a Windows taskbar with various icons and system information like '36°C Sunny' and '11:38 13-04-2023'.

`cd ..` (to enter into the wordpress directory)


```
ansible@master:~/wordpress/vars$ cd ..
ansible@master:~/wordpress$ ls
files  playbook.yml  vars
ansible@master:~/wordpress$
```

Change the template files and variable files as per the requirement of the client.

STEP-4: Execute scripts to perform installation of complete WordPress environment

- 13) `ansible-playbook playbook.yml` (to run the created playbook named `playbook.yml`)

```
ansible@master:~/wordpress$ ls
files  playbook.yml  vars
ansible@master:~/wordpress$ ansible-playbook playbook.yml

PLAY [wpserver] *****

TASK [Gathering Facts] *****
```

```
TASK [Download and unpack latest WordPress] *****
changed: [172.31.18.28]

TASK [Set ownership] *****
changed: [172.31.18.28]

TASK [Set permissions for directories] *****
changed: [172.31.18.28]

TASK [Set permissions for files] *****
changed: [172.31.18.28]

TASK [Set up wp-config] *****
changed: [172.31.18.28]

RUNNING HANDLER [Reload Apache] *****
changed: [172.31.18.28]

RUNNING HANDLER [Restart Apache] *****
changed: [172.31.18.28]

PLAY RECAP *****
172.31.18.28      : ok=22  changed=19  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
```

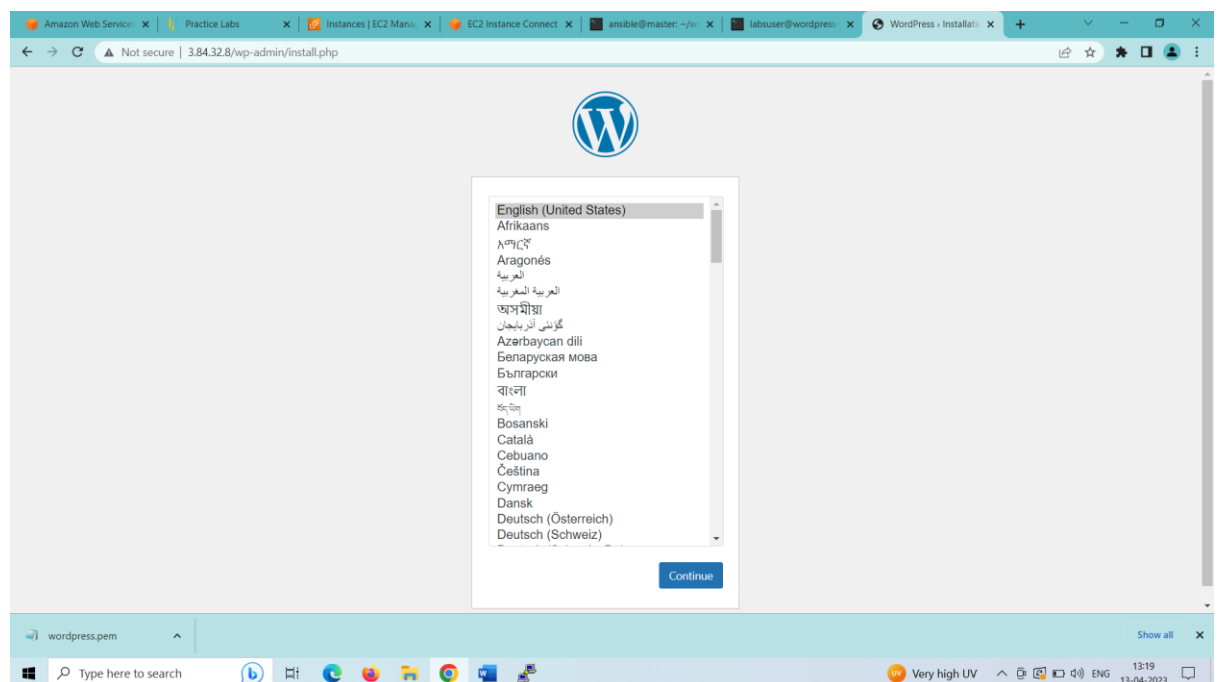
(playbook is executed successfully)

14) To know the public ip of your wordpress server go to terminal of wordpress server and type the following command:

```
ansible@wordpress-server:~$ curl ifconfig.me
54.245.47.35ansible@wordpress-server:~$
```

← → ↻ <http://54.245.47.35:80>

When I have done the same steps using aws lab it worked. Here is the result I got:



Wordpress language selection and login page is opened: enter your credentials and click on install wordpress.

Welcome

Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.

Information needed

Please provide the following information. Do not worry, you can always change these settings later.

Site Title: Akshay's Wordpress

Username: akshay
Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.

Password: *****
Weak
Important: You will need this password to log in. Please store it in a secure location.

Confirm Password: ☒ Confirm use of weak password

Your Email: akshay@gmail.com
Double-check your email address before continuing.

Search engine visibility: ☐ Discourage search engines from indexing this site
It is up to search engines to honor this request.

[Install WordPress](#)

Now login to your account with your credentials and you will reach the dashboard of the wordpress site.

Username or Email Address: akshay

Password: *****

☐ Remember Me [Log In](#)

[Lost your password?](#)

[Go to Akshay's Wordpress](#)

