📄 PROJECT REPORT

Automated WordPress Deployment on AWS EC2 (RHEL) using Ansible with Nginx, PHP-FPM, MariaDB & Let's Encrypt SSL

1️⃣ Project Overview

This project demonstrates the **end-to-end automation of a production-ready WordPress website** on an **AWS EC2 instance running Red Hat Enterprise Linux (RHEL)** using **Ansible**.

The deployment includes:

- Nginx web server
- PHP-FPM (PHP 8+)
- MariaDB database
- WordPress CMS
- SELinux-aware configuration
- Secure file permissions
- Let's Encrypt SSL via **acme.sh**
- Fully automated using **Ansible roles and playbooks**

The project was **first executed manually** to understand real-world Linux and SELinux behavior, then **fully automated using Ansible from scratch**.

2️⃣ Infrastructure Details

Component	Details
Cloud Provider	AWS EC2
Region	ap-south-1 (Mumbai)
OS	Red Hat Enterprise Linux (RHEL 9/10)
Web Server	Nginx
PHP	PHP-FPM (8.x)
Database	MariaDB
CMS	WordPress
Automation Tool	Ansible (control node on VMware RHEL)
SSL	Let's Encrypt (acme.sh)
Domain	logstellthetruth.site
DNS Provider	GoDaddy

3️⃣ Architecture Overview

```
Ansible Control Node (VMware RHEL)
        |
        | SSH (key-based)
        |
AWS EC2 (RHEL)
 ├── Nginx (80 → 443)
 ├── PHP-FPM (Unix socket)
 ├── MariaDB
 ├── WordPress (/home/ec2-user/wordpress/public)
 ├── SELinux (Enforcing)
 └── SSL Certs (/etc/nginx/ssl)
```

4️⃣ Ansible Project Structure

```
/home/anandu
├── ansible.cfg
```

```
├── inventory/
│   └── hosts.ini
├── group_vars/
│   └── web.yml
├── playbooks/
│   ├── phase5-mariadb.yml
│   ├── phase6-wordpress-files.yml
│   ├── phase7-nginx-vhost.yml
│   ├── phase8-wp-config.yml
│   └── phase9-ssl.yml
├── roles/
│   ├── nginx/
│   │   ├── tasks/
│   │   │   ├── main.yml
│   │   │   └── ssl.yml
│   │   ├── templates/
│   │   │   ├── wordpress.conf.j2
│   │   │   └── nginx-ssl.conf.j2
│   │   └── handlers/main.yml
│   ├── wordpress/
│   │   ├── tasks/main.yml
│   │   └── templates/wp-config.php.j2
│   └── ssl/
│       └── tasks/main.yml
```

5️⃣ Inventory Configuration

📄 inventory/hosts.ini

```
[web]
wordpress ansible_host=13.235.99.251 ansible_user=ec2-user
```

6️⃣ Ansible Configuration

📄 ansible.cfg

```
[defaults]
inventory = inventory/hosts.ini
host_key_checking = False
roles_path = roles
```

7️⃣ Database Variables

📄 group_vars/web.yml

```
wp_db_name: wordpress_db
wp_db_user: wp_user
wp_db_password: StrongPassword@123
wp_db_host: localhost
wp_table_prefix: wp_

domain_name: logstellthetruth.site
domain_www: www.logstellthetruth.site

ssl_cert_dir: /etc/nginx/ssl
web_root: /home/ec2-user/wordpress/public
```

8️⃣ Phase 5 – MariaDB Automation

📄 playbooks/phase5-mariadb.yml

```
- name: Phase 5 - Install and configure MariaDB
  hosts: web
  become: yes

  tasks:
    - name: Install MariaDB
      dnf:
        name: mariadb-server
        state: present

    - name: Start and enable MariaDB
```

```
    systemd:
      name: mariadb
      state: started
      enabled: yes
```

9️⃣ Phase 6 – WordPress Files & Permissions

📄 playbooks/phase6-wordpress-files.yml

```
- name: Phase 6 - Prepare WordPress files and directories
  hosts: web
  become: yes

  tasks:
    - name: Create WordPress base directory
      file:
        path: /home/ec2-user/wordpress
        state: directory

    - name: Create WordPress public directory
      file:
        path: /home/ec2-user/wordpress/public
        state: directory

    - name: Download WordPress using curl
      command: curl -o /tmp/wordpress.tar.gz
https://wordpress.org/latest.tar.gz

    - name: Extract WordPress into public directory
      unarchive:
        src: /tmp/wordpress.tar.gz
        dest: /home/ec2-user/wordpress/public
        remote_src: yes
        extra_opts: [--strip-components=1]

    - name: Set ownership
      file:
```

```
        path: /home/ec2-user/wordpress
        owner: ec2-user
        group: nginx
        recurse: yes

    - name: Set permissions
      command: chmod -R 755 /home/ec2-user/wordpress

    - name: Set SELinux context
      command: chcon -R -t httpd_sys_content_t
/home/ec2-user/wordpress
```

🔐 SELinux Critical Fixes (Manual)

```
sudo setsebool -P httpd_enable_homedirs on
sudo chmod o+x /home
sudo chmod o+x /home/ec2-user
```

Verification:

```
namei -l /home/ec2-user/wordpress/public
```

🔟 Phase 7 – Nginx Configuration

📄 roles/nginx/templates/wordpress.conf.j2

```
server {
    listen 80;
    server_name _;

    root /home/ec2-user/wordpress/public;
    index index.php index.html;

    location ^~ /.well-known/acme-challenge/ {
        allow all;
```

```
        root /home/ec2-user/wordpress/public;
    }

    location / {
        try_files $uri $uri/ /index.php?$args;
    }

    location ~ \.php$ {
        include fastcgi.conf;
        fastcgi_pass unix:/run/php-fpm/www.sock;
    }

    location ~ /\. {
        deny all;
    }
}
```

📄 roles/nginx/tasks/main.yml

```
- name: Remove default nginx config
  file:
    path: /etc/nginx/conf.d/default.conf
    state: absent

- name: Deploy WordPress nginx vhost
  template:
    src: wordpress.conf.j2
    dest: /etc/nginx/conf.d/wordpress.conf
  notify:
    - Reload nginx

- name: Deploy SSL nginx vhost
  template:
    src: nginx-ssl.conf.j2
    dest: /etc/nginx/conf.d/wordpress-ssl.conf
  notify:
    - Reload nginx
```

📄 roles/nginx/handlers/main.yml

```
- name: Reload nginx
  service:
    name: nginx
    state: reloaded

- name: Restart nginx
  service:
    name: nginx
    state: restarted
```

1️⃣1️⃣ Phase 8 – wp-config.php Automation

📄 roles/wordpress/templates/wp-config.php.j2

```
<?php
define('DB_NAME', '{{ wp_db_name }}');
define('DB_USER', '{{ wp_db_user }}');
define('DB_PASSWORD', '{{ wp_db_password }}');
define('DB_HOST', '{{ wp_db_host }}');

define('DB_CHARSET', 'utf8mb4');
define('DB_COLLATE', '');

{{ wp_salts }}

$table_prefix = '{{ wp_table_prefix }}';

define('WP_DEBUG', false);

if (!defined('ABSPATH')) {
    define('ABSPATH', __DIR__ . '/');
}

require_once ABSPATH . 'wp-settings.php';
```

📄 roles/wordpress/tasks/main.yml

```
- name: Generate WordPress salts
  command: curl -s https://api.wordpress.org/secret-key/1.1/salt/
  register: wp_salt_output
  changed_when: false

- name: Set WordPress salts
  set_fact:
    wp_salts: "{{ wp_salt_output.stdout }}"

- name: Create wp-config.php
  template:
    src: wp-config.php.j2
    dest: /home/ec2-user/wordpress/public/wp-config.php
    owner: ec2-user
    group: nginx
    mode: "0640"

- name: Set SELinux context
  command: chcon -t httpd_sys_content_t
/home/ec2-user/wordpress/public/wp-config.php
```

1️⃣2️⃣ Phase 9A – SSL Automation

📄 roles/ssl/tasks/main.yml

```
- name: Install curl
  dnf:
    name: curl
    state: present

- name: Install acme.sh
  shell: curl https://get.acme.sh | sh
  args:
    creates: /root/.acme.sh/acme.sh

- name: Set Let's Encrypt CA
```

```
  command: /root/.acme.sh/acme.sh --set-default-ca --server
letsencrypt

- name: Create SSL directory
  file:
    path: /etc/nginx/ssl
    state: directory

- name: Issue SSL certificate
  command: >
    /root/.acme.sh/acme.sh --issue
    -d {{ domain_name }}
    -d {{ domain_www }}
    -w {{ web_root }}

- name: Install certificate
  command: >
    /root/.acme.sh/acme.sh --install-cert
    -d {{ domain_name }}
    --key-file /etc/nginx/ssl/{{ domain_name }}.key
    --fullchain-file /etc/nginx/ssl/{{ domain_name }}.crt
    --reloadcmd "systemctl reload nginx"
```

📄 roles/nginx/templates/nginx-ssl.conf.j2

```
server {
    listen 80;
    server_name {{ domain_name }} {{ domain_www }};
    return 301 https://$host$request_uri;
}

server {
    listen 443 ssl;
    server_name {{ domain_name }} {{ domain_www }};

    ssl_certificate {{ ssl_cert_dir }}/{{ domain_name }}.crt;
    ssl_certificate_key {{ ssl_cert_dir }}/{{ domain_name }}.key;
```

```
    root {{ web_root }};
    index index.php index.html;

    location / {
        try_files $uri $uri/ /index.php?$args;
    }

    location ~ \.php$ {
        include fastcgi.conf;
        fastcgi_pass unix:/run/php-fpm/www.sock;
    }

    location ~ /\. {
        deny all;
    }
}
```

1️⃣3️⃣ Final Verification Commands

```
sudo nginx -t
sudo ss -tulpn | grep nginx
ls -l /etc/nginx/ssl
dig logstellthetruth.site
```

1️⃣4️⃣ Final Outcome

- ✅ WordPress running over HTTPS
- ✅ SELinux enforcing
- ✅ SSL auto-renew enabled
- ✅ Fully automated via Ansible

- ✅ Reusable and idempotent setup

1️⃣5️⃣ Conclusion

This project demonstrates **real-world DevOps automation** on RHEL with:

- Secure Linux practices
- SELinux-aware configurations
- Proper Ansible role separation
- Production-grade SSL automation

It reflects **hands-on troubleshooting, system administration expertise, and infrastructure automation skills** suitable for **DevOps / Linux Administrator roles**.