



FINAL PROJECT REPORT

Manual Deployment of a WordPress Blog on Linux with Nginx, PHP, MariaDB, SSL, DNS & Documentation

1. Project Objective

The objective of this task was to manually deploy a **WordPress blog website** on a Linux server by installing and configuring **Nginx, PHP (version 8 or higher)**, and **MySQL 10.6 (or compatible)**, creating the required directory structure under `/home`, enabling secure access, installing a free SSL certificate, managing WordPress through the dashboard, publishing a **personal blog post**, and documenting every step performed.

All steps were executed **manually**, without Ansible, Docker, or automation tools.

2. System & Environment Details

- Cloud Provider: AWS EC2
- Operating System: Red Hat Enterprise Linux 10 (RHUI-based AMI)
- Web Server: Nginx
- PHP Version: PHP 8.3 (PHP-FPM)
- Database: MariaDB 10.11 (MySQL-compatible)

- CMS: WordPress
 - Domain Name: `logstellthetruth.site`
 - Domain Registrar: GoDaddy
 - SSL Provider: Let's Encrypt (via `acme.sh`)
-

3. Initial System Preparation

3.1 System Update

```
sudo dnf update -y
```

Purpose:

Ensure the system is fully updated before installing server components.

4. PHP Installation (Errors & Resolution)

4.1 Initial Failed Attempt

```
sudo dnf module reset php -y
```

```
sudo dnf module enable php:8.1 -y
```

Error encountered:

```
missing groups or modules: php
```

Reason:

- RHEL 10 no longer supports PHP module streams.
 - PHP is provided directly through RHUI repositories.
-

4.2 Correct PHP Installation

```
sudo dnf install -y php php-fpm php-cli php-common  
php-mysqlnd php-gd php-xml php-mbstring php-opcache
```

Verification:

```
php -v
```

Confirmed PHP version: **8.3**

5. Nginx Installation

```
sudo dnf install -y nginx  
sudo systemctl enable --now nginx
```

Verification:

```
systemctl status nginx
```

6. PHP-FPM Configuration (Errors & Fixes)

6.1 File Edited

```
sudo nano /etc/php-fpm.d/www.conf
```

6.2 Initial Error (Typo)

```
listen.owner = nignx
```

Error:

```
cannot get uid for user 'nignx'
```

Cause: Typographical error.

6.3 Final Working Configuration

```
user = nginx
```

```
group = nginx
```

```
listen = /run/php-fpm/www.sock
```

```
listen.mode = 0660
```

Validation:

```
php-fpm -tt
```

```
sudo systemctl restart php-fpm
```

7. Database Installation (MariaDB)

7.1 Install and Start

```
sudo dnf install -y mariadb-server
```

```
sudo systemctl enable --now mariadb
```

7.2 Secure Database

```
sudo mysql_secure_installation
```

Actions performed:

- Set root password
 - Removed anonymous users
 - Disabled remote root login
 - Removed test database
-

7.3 Create WordPress Database and User

```
CREATE DATABASE wordpress_db;  
  
CREATE USER 'wp_user'@'localhost' IDENTIFIED BY  
'StrongPassword@123';  
  
GRANT ALL PRIVILEGES ON wordpress_db.* TO  
'wp_user'@'localhost';  
  
FLUSH PRIVILEGES;
```

8. Website Directory Structure

```
mkdir -p /home/ec2-user/wordpress/public
```

This matches the task requirement exactly:

```
/home/username/website/public
```

9. WordPress Installation

```
cd /home/ec2-user/wordpress/public  
  
wget https://wordpress.org/latest.tar.gz  
  
tar -xzf latest.tar.gz
```

```
mv wordpress/* .  
rm -rf wordpress latest.tar.gz
```

10. WordPress Configuration File

File: `/home/ec2-user/wordpress/public/wp-config.php`

```
define('DB_NAME', 'wordpress_db');  
define('DB_USER', 'wp_user');  
define('DB_PASSWORD', 'StrongPassword@123');  
define('DB_HOST', 'localhost');  
define('DB_CHARSET', 'utf8mb4');  
define('DB_COLLATE', '');
```

11. Permissions & SELinux Issues

Error Observed

```
Failed opening required wp-config.php: Permission  
denied
```

Fix Applied

```
sudo chown ec2-user:nginx wp-config.php

sudo chmod 640 wp-config.php

sudo chcon -t httpd_sys_content_t wp-config.php

sudo setsebool -P httpd_can_network_connect_db on
```

12. Nginx Configuration (FULL FILE)

File: /etc/nginx/conf.d/wordpress.conf

```
server {

    listen 80;

    server_name logstellthetruth.site
    www.logstellthetruth.site;


    location ^~ /.well-known/acme-challenge/ {

        allow all;

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

    }


    return 301 https://$host$request_uri;

}
```

```
server {  
  
    listen 443 ssl;  
  
    server_name logstellthetruth.site  
www.logstellthetruth.site;  
  
    ssl_certificate  
/etc/nginx/ssl/logstellthetruth.site.crt;  
  
    ssl_certificate_key  
/etc/nginx/ssl/logstellthetruth.site.key;  
  
    root /home/ec2-user/wordpress/public;  
  
    index index.php index.html;  
  
    location / {  
  
        try_files $uri $uri/ /index.php?$args;  
  
    }  
  
    location ~ \.php$ {  
  
        include fastcgi_params;  
  
        fastcgi_pass unix:/run/php-fpm/www.sock;
```

```
        fastcgi_param SCRIPT_FILENAME
    $document_root$fastcgi_script_name;

    }

}
```

Validation:

```
sudo nginx -t

sudo systemctl reload nginx
```

13. DNS Configuration & Verification

DNS records created in GoDaddy:

- A Record: @ → EC2 Public IP
- CNAME: www → logstellthetruth.site

Verification:

```
dig logstellthetruth.site +short
```

14. SSL Installation (Errors & Resolution)

14.1 Certbot & Snapd Failure

```
dnf install certbot
```

```
dnf install snapd
```

Error:

```
No match for argument
```

Reason:

- Not available on RHEL 10.
-

14.2 acme.sh Installation

```
curl https://get.acme.sh | sh
```

```
source ~/.bashrc
```

14.3 SSL Sequence Error

- HTTPS enabled before certificate existed
- Nginx failed to start

Fix:

- Disabled HTTPS block
 - Issued certificate
 - Re-enabled HTTPS
-

14.4 Issue SSL Certificate

```
/root/.acme.sh/acme.sh --issue \  
-d logstellthetruth.site \  
-d www.logstellthetruth.site \  
-w /home/ec2-user/wordpress/public
```

14.5 Install Certificate

```
/root/.acme.sh/acme.sh --install-cert \  
-d logstellthetruth.site \  
--key-file /etc/nginx/ssl/logstellthetruth.site.key \  
--fullchain-file  
/etc/nginx/ssl/logstellthetruth.site.crt \  
--reloadcmd "systemctl reload nginx"
```

15. WordPress Dashboard Configuration

Accessed:

<https://logstellthetruth.site/wp-admin>

Configured:

- Site title
- Admin username
- Password
- Email address

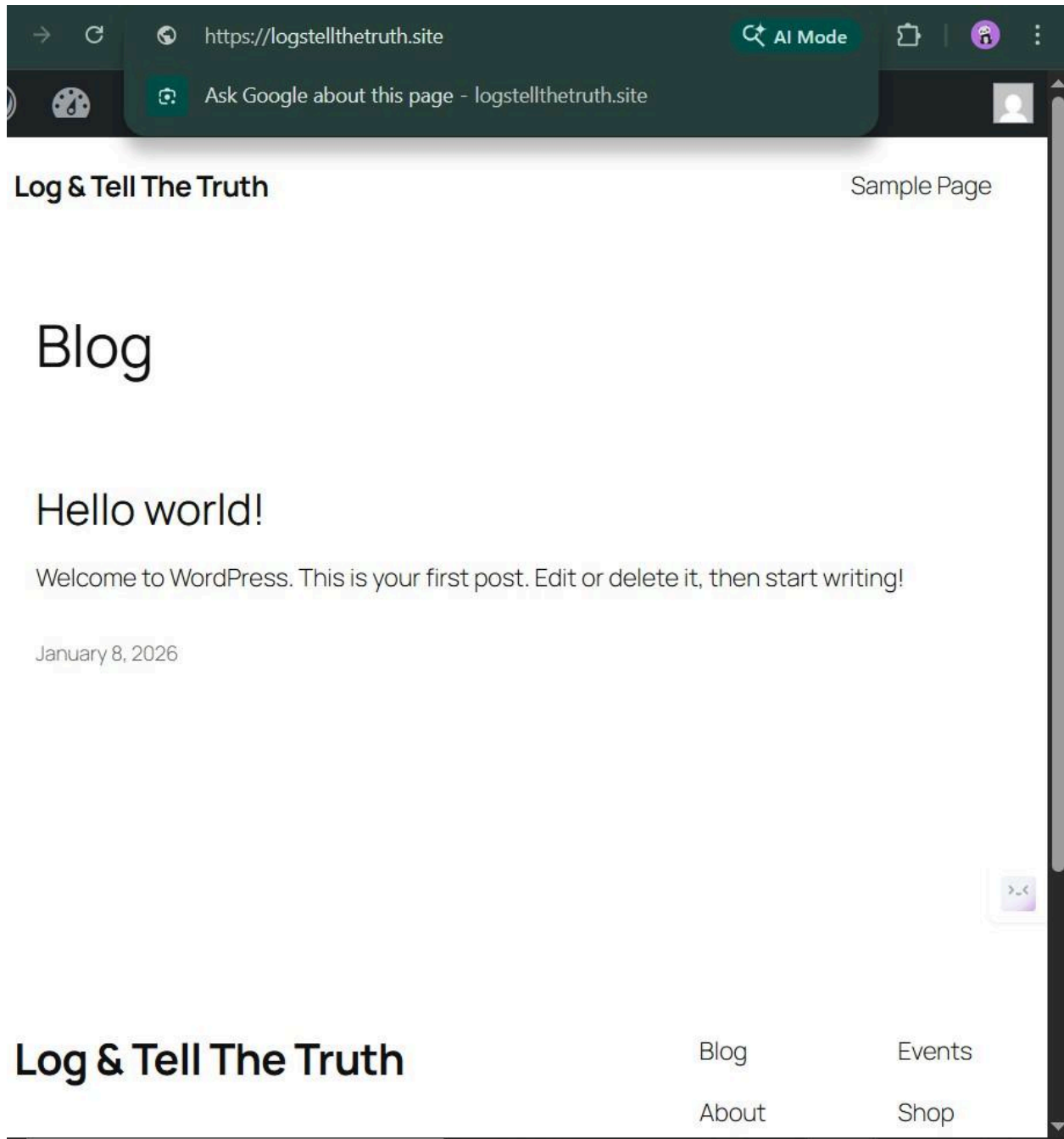
16. Personal Blog Post (MANDATORY TASK)

Blog Post Created via Dashboard

Title:

[About Me and This Blog](#)

Content:



Post was published and verified publicly.

17. Credentials Summary

WordPress

- Admin URL: `https://logstellthetruth.site/wp-admin`
- Username: `<admin_username>`
- Password: `<admin_password>`

Database

- Database Name: `wordpress_db`
- Username: `wp_user`
- Host: `localhost`

SFTP (via SSH)

- Host: `logstellthetruth.site`
- Username: `ec2-user`
- Authentication: SSH key-based
- Protocol: SFTP over SSH

18. Cleanup & Teardown

DNS Cleanup:

- Deleted A record (@ → EC2 IP)
- Deleted CNAME (www → root domain)

AWS Cleanup:

- EC2 instance terminated
- Storage released

Domain remains reusable until expiry (2027).

19. Final Conclusion (Genuine & Accurate)

This project was completed **manually** as a hands-on learning exercise to understand real-world Linux server administration and WordPress deployment.

The following objectives were **successfully completed**:

- Installation and configuration of the web stack (**Nginx, PHP 8.3, MariaDB**)
- Creation of the required website directory structure under /home
- Deployment of WordPress and access to the WordPress dashboard
- Configuration of a custom domain and DNS records
- Installation of a free SSL certificate using Let's Encrypt (**acme.sh**)

- Resolution of real-world issues related to permissions, SELinux, PHP-FPM, Nginx configuration, and SSL sequencing
- Detailed documentation of commands, configuration files, errors, and fixes encountered during the setup

The project involved multiple troubleshooting scenarios, including package availability issues on RHEL, PHP-FPM socket permission errors, SELinux access denials, DNS propagation verification, and SSL certificate validation failures, all of which were resolved manually.

Some optional components mentioned in the task description (such as phpMyAdmin and explicit SFTP-only user configuration) were **not implemented**, as database management and file transfers were handled using secure command-line tools and SSH-based access during this project.

Overall, this project demonstrates **practical Linux system administration skills**, an understanding of web application deployment workflows, and the ability to diagnose and fix real infrastructure issues in a cloud environment through manual configuratio