in centos 9 install and wordpress and config

**ChatGPT said:**

Here's a **step-by-step guide to install and configure WordPress** on **CentOS Stream 9** using **Apache, MariaDB, and PHP (LAMP stack)**:

**✅ 1. Update Your System**

sudo dnf update -y

sudo dnf install epel-release -y

**✅ 2. Install Apache Web Server**

sudo dnf install httpd -y

sudo systemctl enable --now httpd

sudo systemctl status httpd

Allow HTTP & HTTPS traffic through the firewall:

sudo firewall-cmd --permanent --add-service=http

sudo firewall-cmd --permanent --add-service=https

sudo firewall-cmd --reload

**✅ 3. Install MariaDB (MySQL)**

sudo dnf install mariadb-server mariadb -y

sudo systemctl enable --now mariadb

sudo systemctl status mariadb

Secure the installation:

sudo mysql\_secure\_installation

During this, set a root password, remove anonymous users, disallow remote root login, remove test database, and reload privileges.

**✅ 4. Create a WordPress Database**

Log in to MariaDB:

sudo mysql -u root -p

Run the following (replace with your preferred dbname, user, and password):

CREATE DATABASE wordpress\_db;

GRANT ALL PRIVILEGES ON wordpress\_db.\* TO 'wp\_user'@'localhost' IDENTIFIED BY 'strong\_password';

FLUSH PRIVILEGES;

EXIT;

**✅ 5. Install PHP and Required Modules**

WordPress needs PHP 7.4+ (CentOS 9 includes PHP 8.x):

sudo dnf install php php-mysqlnd php-fpm php-opcache php-gd php-xml php-mbstring php-curl php-zip -y

sudo systemctl enable --now php-fpm

Restart Apache to load PHP:

sudo systemctl restart httpd

**✅ 6. Download and Set Up WordPress**

cd /tmp

curl -O https://wordpress.org/latest.tar.gz

tar -xvzf latest.tar.gz

sudo rsync -av wordpress/ /var/www/html/

Set permissions:

sudo chown -R apache:apache /var/www/html/

sudo find /var/www/html/ -type d -exec chmod 755 {} \;

sudo find /var/www/html/ -type f -exec chmod 644 {} \;

**✅ 7. Configure WordPress**

Copy the sample config:

cd /var/www/html/

cp wp-config-sample.php wp-config.php

Edit wp-config.php:

sudo nano wp-config.php

Set the DB settings:

define('DB\_NAME', 'wordpress\_db');

define('DB\_USER', 'wp\_user');

define('DB\_PASSWORD', 'strong\_password');

define('DB\_HOST', 'localhost');

Generate security keys here: https://api.wordpress.org/secret-key/1.1/salt/  
Replace the default placeholders with the generated values.

**✅ 8. Configure SELinux (Optional but recommended)**

If SELinux is enforcing:

sudo dnf install policycoreutils-python-utils -y

sudo chcon -R -t httpd\_sys\_rw\_content\_t /var/www/html/

sudo setsebool -P httpd\_can\_network\_connect\_db 1

**✅ 9. Restart Services**

sudo systemctl restart httpd

sudo systemctl restart php-fpm

**✅ 10. Access WordPress Setup**

Open your browser and go to:

http://YOUR\_SERVER\_IP/

You should see the WordPress installation page. Follow the steps to complete the setup.

**🛠️ Troubleshooting Tips:**

* **Apache blank page?** Check /var/log/httpd/error\_log
* **PHP not loading?** Ensure php module is installed and Apache restarted.
* **403 Forbidden?** Check permissions and SELinux context.
* **Firewall issues?** Make sure port 80/443 are allowed.

If you want, I can generate a bash script to automate this. Let me know.

**You said:**

i all ready install nginx and also test page in index i print hello world

**ChatGPT said:**

Great! Since you already have **Nginx installed and serving a test page**, you're on your way to setting up a **LEMP stack** (Linux, Nginx, MariaDB, PHP) for WordPress on **CentOS 9**.

Let’s continue from here and set up WordPress using Nginx + PHP + MariaDB.

**✅ Step-by-Step: Install and Configure WordPress with Nginx on CentOS 9**

**✅ 1. Install PHP and Required Modules**

sudo dnf install php php-fpm php-mysqlnd php-opcache php-gd php-xml php-mbstring php-curl php-zip php-intl -y

Enable and start PHP-FPM:

sudo systemctl enable --now php-fpm

Make sure PHP-FPM runs as **nginx** user (edit config if needed):

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

Set:

user = nginx

group = nginx

Restart PHP-FPM:

sudo systemctl restart php-fpm

**✅ 2. Install and Configure MariaDB**

If not installed:

sudo dnf install mariadb-server mariadb -y

sudo systemctl enable --now mariadb

Secure the installation:

sudo mysql\_secure\_installation

**✅ 3. Create WordPress Database and User**

sudo mysql -u root -p

Then run:

CREATE DATABASE wordpress\_db DEFAULT CHARACTER SET utf8 COLLATE utf8\_unicode\_ci;

GRANT ALL ON wordpress\_db.\* TO 'wp\_user'@'localhost' IDENTIFIED BY 'strong\_password';

FLUSH PRIVILEGES;

EXIT;

**✅ 4. Download and Configure WordPress**

cd /tmp

curl -O https://wordpress.org/latest.tar.gz

tar -xzf latest.tar.gz

sudo mkdir -p /var/www/wordpress

sudo rsync -av wordpress/ /var/www/wordpress/

Set permissions:

sudo chown -R nginx:nginx /var/www/wordpress

sudo find /var/www/wordpress -type d -exec chmod 755 {} \;

sudo find /var/www/wordpress -type f -exec chmod 644 {} \;

Copy and edit config:

cd /var/www/wordpress

cp wp-config-sample.php wp-config.php

sudo nano wp-config.php

Update the following:

define('DB\_NAME', 'wordpress\_db');

define('DB\_USER', 'wp\_user');

define('DB\_PASSWORD', 'strong\_password');

define('DB\_HOST', 'localhost');

Also, add secret keys from:  
https://api.wordpress.org/secret-key/1.1/salt/

**✅ 5. Configure Nginx for WordPress**

Create a new site config:

sudo nano /etc/nginx/conf.d/wordpress.conf

Paste this (adjust paths and domain as needed):

server {

listen 80;

server\_name your\_domain\_or\_ip;

root /var/www/wordpress;

index index.php index.html index.htm;

access\_log /var/log/nginx/wordpress\_access.log;

error\_log /var/log/nginx/wordpress\_error.log;

location / {

try\_files $uri $uri/ /index.php?$args;

}

location ~ \.php$ {

include fastcgi\_params;

fastcgi\_pass unix:/run/php-fpm/www.sock;

fastcgi\_index index.php;

fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;

}

location ~\* \.(js|css|png|jpg|jpeg|gif|ico|svg)$ {

expires max;

log\_not\_found off;

}

location = /favicon.ico {

log\_not\_found off;

access\_log off;

}

location = /robots.txt {

allow all;

log\_not\_found off;

access\_log off;

}

}

Test and reload:

sudo nginx -t

sudo systemctl reload nginx

**✅ 6. Set SELinux (if Enforcing)**

If SELinux is enabled:

sudo dnf install policycoreutils-python-utils -y

sudo chcon -R -t httpd\_sys\_rw\_content\_t /var/www/wordpress

sudo setsebool -P httpd\_can\_network\_connect\_db 1

**✅ 7. Complete WordPress Setup in Browser**

Open your browser and go to:

http://your\_server\_ip/

You should see the WordPress installation wizard. Choose your language, site name, admin credentials, and finish the setup.

**🎉 Done!**

Now you have a **working WordPress site on Nginx** with CentOS 9.

Let me know if you want:

* HTTPS with Let's Encrypt
* A shell script to automate this
* WordPress behind a domain name
* Performance tuning tips

Just ask.

**You said:**

By default, a MariaDB installation has an anonymous user, allowing anyone to log into MariaDB without having to have a user account created for them. This is intended only for testing, and to make the installation go a bit smoother. You should remove them before moving into a production environment. Remove anonymous users? [Y/n]

**ChatGPT said:**

Yes, you should **remove anonymous users** for security reasons — especially if you're deploying WordPress on a public server.

So, when prompted:

Remove anonymous users? [Y/n]

✅ **Type Y and press Enter**.

This will prevent unauthorized users from logging in without credentials, reducing the risk of exploitation or abuse.

If you’re curious, here’s what that step does:

* MariaDB often creates a user with no username (anonymous), which allows logins from localhost.
* Removing it ensures **only properly authenticated users** (like root, or your WordPress DB user) can access the database.

Let me know if you want help going through the rest of the mysql\_secure\_installation prompts.

**You said:**

Normally, root should only be allowed to connect from 'localhost'. This ensures that someone cannot guess at the root password from the network. Disallow root login remotely? [Y/n]

**ChatGPT said:**

✅ **You should answer Y (Yes)** to:

Disallow root login remotely? [Y/n]

**Why?**

* This **blocks remote root logins** to MariaDB, meaning the root user can only log in **locally** (via localhost or 127.0.0.1).
* It's a **recommended security best practice**, especially on production servers.
* It helps prevent **brute-force attacks** from the internet targeting the root user.

If you ever need remote database access (for backups, administration, etc.), it's better to:

* Create a **separate user** with **limited privileges**.
* Use **SSH tunneling** or **VPN**, not open ports.

Let me know if you want help setting up secure remote access properly later.

**You said:**

By default, MariaDB comes with a database named 'test' that anyone can access. This is also intended only for testing, and should be removed before moving into a production environment. Remove test database and access to it? [Y/n]

**ChatGPT said:**

You should choose **Y** (Yes) here:

Remove test database and access to it? [Y/n]

**Why?**

* The test database is meant for testing and is **accessible by all users**, which can be a security risk.
* Removing it helps **tighten database security** on your production server.
* You won’t need the test database for your WordPress setup anyway.

So go ahead and type Y to remove it, then press Enter.

If you want, I can help you with the next steps!

**You said:**

server { listen 80; server\_name your\_domain\_or\_ip; root /var/www/wordpress; index index.php index.html index.htm; access\_log /var/log/nginx/wordpress\_access.log; error\_log /var/log/nginx/wordpress\_error.log; location / { try\_files $uri $uri/ /index.php?$args; } location ~ \.php$ { include fastcgi\_params; fastcgi\_pass unix:/run/php-fpm/www.sock; fastcgi\_index index.php; fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name; } location ~\* \.(js|css|png|jpg|jpeg|gif|ico|svg)$ { expires max; log\_not\_found off; } location = /favicon.ico { log\_not\_found off; access\_log off; } location = /robots.txt { allow all; log\_not\_found off; access\_log off; } } ## in this copy past is coorect for the or i need to change anything

**ChatGPT said:**

Your current Nginx config block is mostly correct for running **WordPress on CentOS 9 with Nginx + PHP-FPM**. However, a few **important things need to be checked or improved** before using it in production.

**✅ Review of Your Config Block**

Here’s your block again with annotations and recommendations:

server {

listen 80;

server\_name your\_domain\_or\_ip; # ✅ Replace this with your actual domain or server IP

root /var/www/wordpress; # ✅ Make sure this path is correct

index index.php index.html index.htm;

access\_log /var/log/nginx/wordpress\_access.log;

error\_log /var/log/nginx/wordpress\_error.log;

location / {

try\_files $uri $uri/ /index.php?$args; # ✅ WordPress-compatible URL rewriting

}

location ~ \.php$ {

include fastcgi\_params;

fastcgi\_pass unix:/run/php-fpm/www.sock; # ✅ Make sure PHP-FPM is using this socket

fastcgi\_index index.php;

fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;

}

location ~\* \.(js|css|png|jpg|jpeg|gif|ico|svg)$ {

expires max;

log\_not\_found off;

}

location = /favicon.ico {

log\_not\_found off;

access\_log off;

}

location = /robots.txt {

allow all;

log\_not\_found off;

access\_log off;

}

}

**🔧 What You Must Update:**

**1. ✅ server\_name**

Replace:

server\_name your\_domain\_or\_ip;

With your actual domain or server IP address. For example:

server\_name example.com;

Or, if you're testing locally:

server\_name localhost;

Or your IP:

server\_name 192.168.1.100;

**2. ✅ root Path**

Ensure this path:

root /var/www/wordpress;

…matches where you extracted WordPress.

You can verify:

ls /var/www/wordpress/index.php

If it exists, you’re good.

**3. ✅ PHP-FPM Socket Path**

Make sure the socket file exists:

ls /run/php-fpm/www.sock

If it doesn't exist, or if you use TCP instead of a socket, change this line:

fastcgi\_pass unix:/run/php-fpm/www.sock;

To:

fastcgi\_pass 127.0.0.1:9000;

(Only if you're running PHP-FPM over TCP instead of Unix socket.)

**🛡 Optional Improvements**

**✅ Add client\_max\_body\_size for media uploads:**

If you plan to upload media files through WordPress admin:

client\_max\_body\_size 64M;

Put it inside the server {} block.

**✅ Add index.php fallback inside location ~ \.php$ (already correct in your case).**

**✅ Recommended full Nginx config (after tweaks):**

server {

listen 80;

server\_name your\_domain\_or\_ip; # Replace with your domain or IP

root /var/www/wordpress;

index index.php index.html index.htm;

access\_log /var/log/nginx/wordpress\_access.log;

error\_log /var/log/nginx/wordpress\_error.log;

client\_max\_body\_size 64M;

location / {

try\_files $uri $uri/ /index.php?$args;

}

location ~ \.php$ {

include fastcgi\_params;

fastcgi\_pass unix:/run/php-fpm/www.sock;

fastcgi\_index index.php;

fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;

}

location ~\* \.(js|css|png|jpg|jpeg|gif|ico|svg|woff|woff2|ttf|eot)$ {

expires max;

log\_not\_found off;

access\_log off;

}

location = /favicon.ico {

log\_not\_found off;

access\_log off;

}

location = /robots.txt {

allow all;

log\_not\_found off;

access\_log off;

}

}

**✅ After Updating**

1. Test the config:

sudo nginx -t

1. Reload Nginx:

sudo systemctl reload nginx

1. Open in browser:

http://your\_domain\_or\_ip/

You should see the WordPress installation screen.