



Deployment Guide: Laravel web app on Debian 11 (Vagrant/VMware)

Date: February 5, 2026

- **Environment:** macOS (Apple Silicon) -> Vagrant (VMware Desktop) -> Debian 11
- **Stack:** Nginx, PHP 8.5, SQLite

Presented by : Hamza Lazaar

1. Infrastructure Setup (Vagrant)

Goal: Create a stable VM that doesn't hang on SSH connections.

Vagrantfile Configuration

Ensure the provider block includes `ssh_info_public` to fix macOS routing issues and allocates enough RAM for PHP 8.5.

```

config.vm.provider "vmware_desktop" do |v|
  v.gui = false
  v.ssh_info_public = true  # Fixes SSH hangs on macOS
  v.memory = "2048"        # 2GB RAM for Laravel/Composer
end

```

```

# backing providers for Vagrant. These expose provider-specific options.
# Example for VirtualBox:
#
config.vm.provider "vmware_desktop" do |vb|
  vb.gui = false

  vb.ssh_info_public = true
# Customize the amount of memory on the VM:
  vb.memory = "2048"
end
#
# View the documentation for the provider you are using for more
# information on available options.

# Enable provisioning with a shell script. Additional provisioners such as
# Ansible, Chef, Docker, Puppet and Salt are also available. Please see the
# documentation for more information about their specific syntax and use.
#config.vm.provision "shell", inline: <<-SHELL
# apt-get update
#apt-get install -y netcat
# sudo nohup nc -l -p 80 >/dev/null 2>&1 &
# sudo apt-get install -y apache2
# curl -o /var/www/html/index.html https://gist.githubusercontent.com/Bahlaouane-Hamza/11bcd425241bf4ab4fa0ed11904$
# curl -o /etc/apache2/sites-enabled/000-default.conf https://gist.githubusercontent.com/Bahlaouane-Hamza/fb27e854$
#SHELL
end

```

2. Server Provisioning

Goal: Install the latest PHP version (8.5) and the Web Server (Nginx).

Fix Debian Repositories

Debian 11 "Backports" often throws 404 errors. Disable it before updating.

```

# Comment out the backports line in sources.list
sudo sed -i '/bullseye-backports/s/^/#/' /etc/apt/sources.lis

```

```
t  
sudo apt update
```

Install PHP 8.5 (Ondřej Surý Repo)

Debian 11 defaults to old PHP versions. Use the maintainer's repo for the latest.

```
# 1. Add Repository  
sudo apt install -y lsb-release ca-certificates curl apt-tran  
sport-https software-properties-common gnupg2  
sudo curl -sSLo /usr/share/keyrings/deb.sury.org-php.gpg http  
s://packages.sury.org/php/apt.gpg  
echo "deb [signed-by=/usr/share/keyrings/deb.sury.org-php.gp  
g] https://packages.sury.org/php/ $(lsb_release -sc) main" |  
sudo tee /etc/apt/sources.list.d/php.list  
sudo apt update  
  
# 2. Install PHP + Extensions  
sudo apt install -y php8.5 php8.5-cli php8.5-common php8.5-fp  
m \  
php8.5-curl php8.5-mbstring php8.5-xml php8.5-zip php8.5-sqli  
te3
```

Note: The `php8.5-sqlite3` package is vital. Without it, migration commands will fail with "driver not found".

3. Application Setup (The Web App)

Goal: Deploy the official Laravel skeleton (`laravel/laravel`) to serve as our test web application.

We are using the standard framework repository because it provides a clean, guaranteed-to-work application structure, perfect for verifying that our Nginx and PHP-FPM stack is configured correctly.



N.B. Source Control & Authentication

Instead of cloning the public repository directly, I **forked** `laravel/laravel` into my own **private GitHub repository** to simulate a real-world private project. Because the repository is private, I used **SSH** for authentication.

Generated SSH Keys: Ran `ssh-keygen -t ed25519` inside the VM.

Authorized Key: Added the public key (`cat ~/.ssh/id_ed25519.pub`) to my GitHub account settings.

Cloned via SSH:

```
# Used the git@github.com format instead of https://
git clone git@github.com:your-username/your-private-l
aravel-fork.git .
```

Clone the Web App

```
# Navigate to the web root
cd /var/www/html

# Clear default index.html if present
rm -rf *

# Clone the official Laravel skeleton app
git clone https://github.com/laravel/laravel.git .

# Install PHP dependencies
composer install
```

Permissions & Environment

Since Nginx runs as `www-data`, we must ensure it owns the application files.

```
# Fix Ownership
sudo chown -R www-data:www-data /var/www/html
```

```
# Set Write Permissions for storage and cache
sudo chmod -R 775 /var/www/html/storage /var/www/html/bootstrap/cache

# Create Environment File
cp .env.example .env
php artisan key:generate
```

Database Configuration (SQLite)

For this test deployment, we switch from MySQL to SQLite to keep the environment lightweight.

1. **Edit** `.env`: Change `DB_CONNECTION=sqlite`.
2. **Initialize Database:**

```
# Create the database file
sudo touch /var/www/html/database/database.sqlite

# Ensure Nginx can write to it
sudo chown www-data:www-data /var/www/html/database/database.sqlite

# Create tables (Users, Sessions, etc.)
sudo php artisan migrate --force
```

4. Nginx Configuration

Goal: Serve the application correctly (avoiding "Insecure Download Blocked" errors).

File: `/etc/nginx/sites-available/default`

Nginx

```
server {
    listen 80;
    listen [::]:80;
```

```

# 1. Point to the public folder
root /var/www/html/public;

# 2. Add 'index' before the file names
index index.php index.html index.htm;

server_name _;

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

# 3. PHP Processing Block
location ~ \.php$ {
    include snippets/fastcgi-php.conf;
    # Ensure this matches your installed version
    fastcgi_pass unix:/run/php/php8.5-fpm.sock;
}
}

```

Restart Nginx:

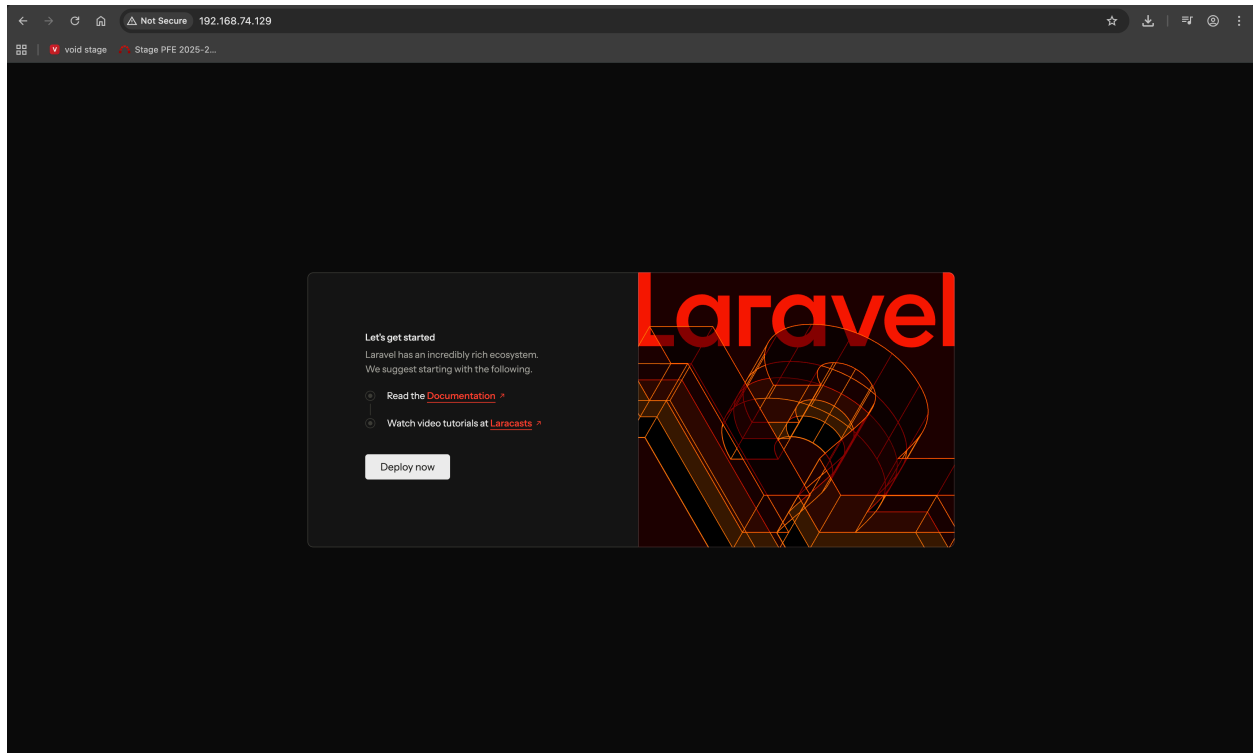
```

sudo nginx -t # Check for syntax errors
sudo systemctl restart nginx

```

5. Verification

1. **Get VM IP:** Run `hostname -I` inside the VM (e.g., `192.168.74.129`).
2. **Browser:** Visit `http://192.168.74.129`.
3. **Result:** You should see the Laravel Welcome page.



Troubleshooting Log

Error	Cause	Fix
"Directory not empty" (Vagrant up)	Stale VMware lock files (<code>.lck</code>)	Delete <code>.lck</code> files inside <code>.vagrant/machines/...</code>
"Insecure download blocked"	Nginx serving PHP as text	Fix Nginx config <code>fastcgi_pass</code> and <code>root</code> .
"Could not find driver"	Missing SQLite extension	Run <code>sudo apt install php8.5- sqlite3</code> .
Database file does not exist	SQLite file missing	Run <code>touch database.sqlite</code> and fix permissions.
SSH Timeout / Broken Pipe	VM out of RAM or DNS timeout	Increase RAM to 2048MB.