BÁO CÁO CUỐI KỲ

Môn học: Chuyên đề Hệ thống thông tin

Đề tài: Tài liệu hướng dẫn thiết lập và triển khai sản phẩm

Họ Tên: Đinh Văn Ty

MSSV: N20DCCN070

Mã Lớp: D20CQCNHT01-N

# **Giới thiệu**

* **Mục đích tài liệu**: Tài liệu nhằm hướng đến cách triển khai bao gồm các công việc sau:
* Cài đặt website có hai thành phần là **CMS (Wordpress)** và **Ecommerce (WooCommerce)** mã nguồn mở.
* Dùng cơ chế **reverse proxy** (**NGINX**)
* Demo triển khai sản phẩm theo cách thức **CI/CD (Github Actions)**
* **Phạm vi**: Tài liệu này hướng dẫn triển khai ở môi trường phát triển (Development enviroment) và môi trường và môi trường kiểm thử (Testing enviroment)

# **Tổng quan**

**Các công nghệ sử dụng**:

* Programming Language: PHP
* Content Management System (CMS): Wordpress
* Database: MySQL
* Deployment: Docker Container
* Source Managment: Github
* CI/CD Tools: Github actions
* Cloud: Digital Ocean

1. **Yêu cầu hệ thống:**

## **Yêu cầu về phần cứng:**

* Hệ điều hành: Window: 10/11 64-bit: yêu cầu có WSL 2 (version 1.1.3.0+) hoặc HyperV. Linux, Mac OS
* CPU: vi xử lý 64-bit hỗ trợ ảo hóa (virtualization)
* Bộ nhớ RAM: tối thiểu 4GB
* Bộ nhớ: tối thiểu 128GB

## **Yêu cầu về phần mềm:**

* Docker và Docker Desktop
* IDE: Visual Studio, Visual Studio code ...
* Công cụ quản lý mã nguồn Git

# **Hướng dẫn sử dụng**

## **Pull sản phẩm:**

* Mở **shell** hoặc **command prompt**
* Điều tới thư mục của sản phẩm

cd /path/to/project\_directory

* Kéo mã nguồn từ **github**

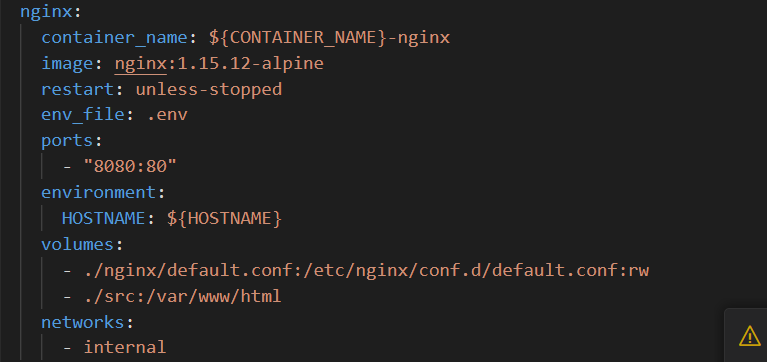
git clone https://github.com/Eren206/CICDFinalExam.git

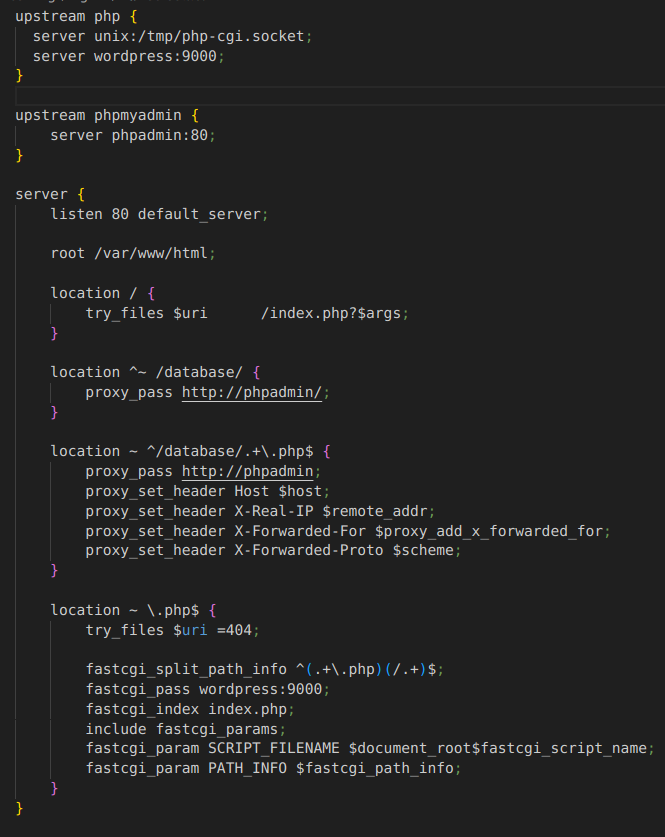
## **Triển khai phần mềm trên môi trường development:**

### **Thiết lập các cài đặt**

* Sản phẩm mặc định đã được thiết lập sẵn ở trên file **docker-compose.yml**, bao gồm các service như sau:
  + **nginx**: server đóng vai trò là một reverse proxy cho phép truy cập service **wordpress** và **phpadmin**
  + **database**: ở đây dùng database **mysql**
  + **phpadmin**: cho phép quản lý database của database service
  + **wordpress**: website

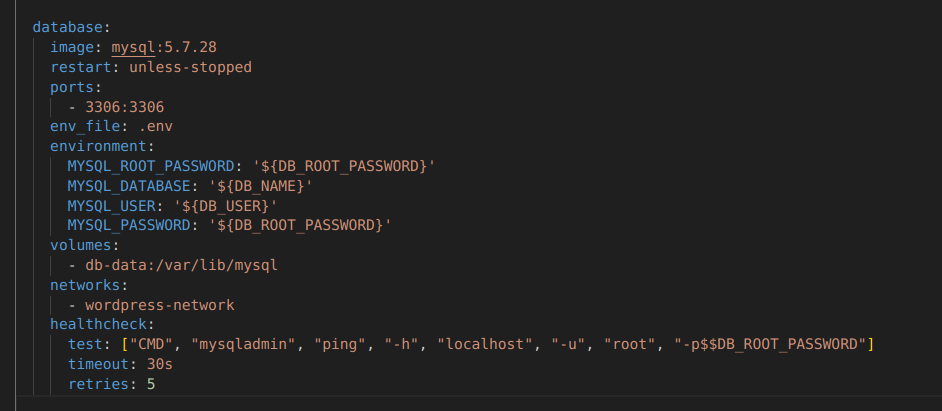
### **Service nginx**

* Service **nginx** được thiết lập trên **docker-compose** như sau:
* **port** mặc định ở đây sẽ là **8080** cho phép ta truy cập các service **wordpress** và **phpadmin** qua *http://localhost:8080*
* file config của **nginx** nằm trong thư mục **./nginx** được cấu hình như sau:



### **Service database**

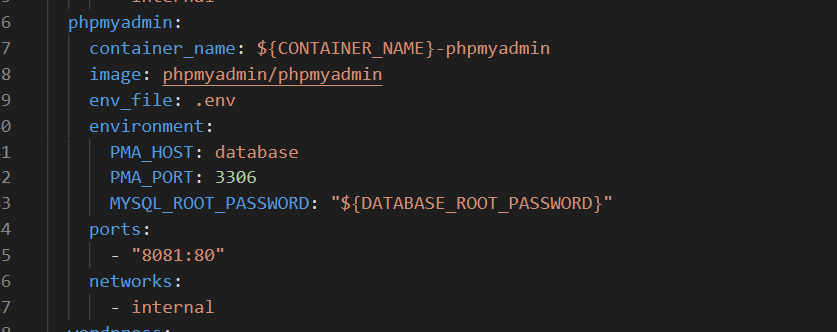
* Service **database** được cấu hình trong **docker-compose** như sau:



*(Note: các biến enviroment được lưu trữ trong file .env)*

### **Service phpmyadmin**

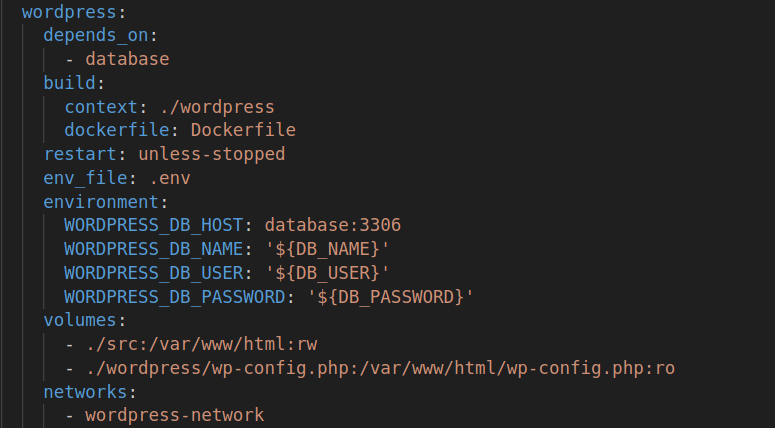
* Service **phpmyadmin** được thiết lập trong file **docker-compose** như sau:



*(Note:* ***PMA\_HORT*** *phải trùng với cổng được expose ở service* ***database*** *và tên của* ***PMA\_HOST*** *phải trùng với tên của service* ***database****)*

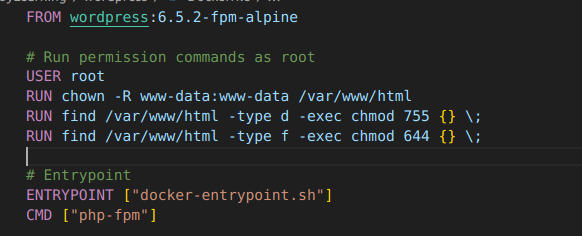
### **Service wordpress**

* Service **wordpress** được thiết lập trong file **docker-compose** như sau:



*(Note: giá trị của* ***WORDPRESS\_DB\_HOST*** *phải đi theo format {tên database* ***service****}:{****port*** *của service* ***database****})*

* File **Dockerfile** của service **wordpress** được cấu hình như sau:



*(Note: ở đây chỉ tập trung vào phần triển khai và cài đặt nên setup trong thư mục ./wordpress/wp-config.php có dùng* ***define('FS\_METHOD', 'direct')*** *để giúp việc cài đặt các plugins nhanh hơn))*

### **Khởi chạy dự án**

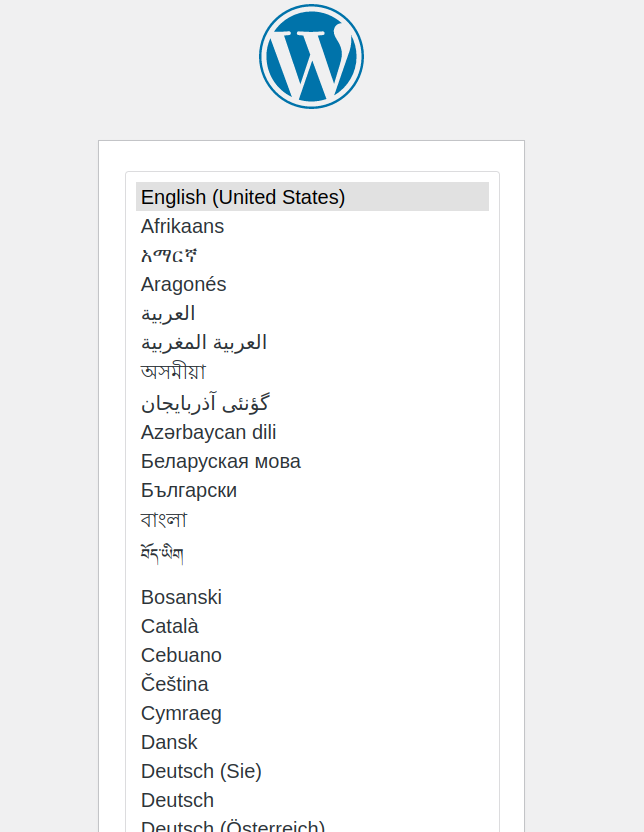
* Khởi động **Docker Desktop**
* Mở cmd và đứng ở thư mục chứa file **docker-compose.yaml**, chạy lệnh:

docker compose up -d

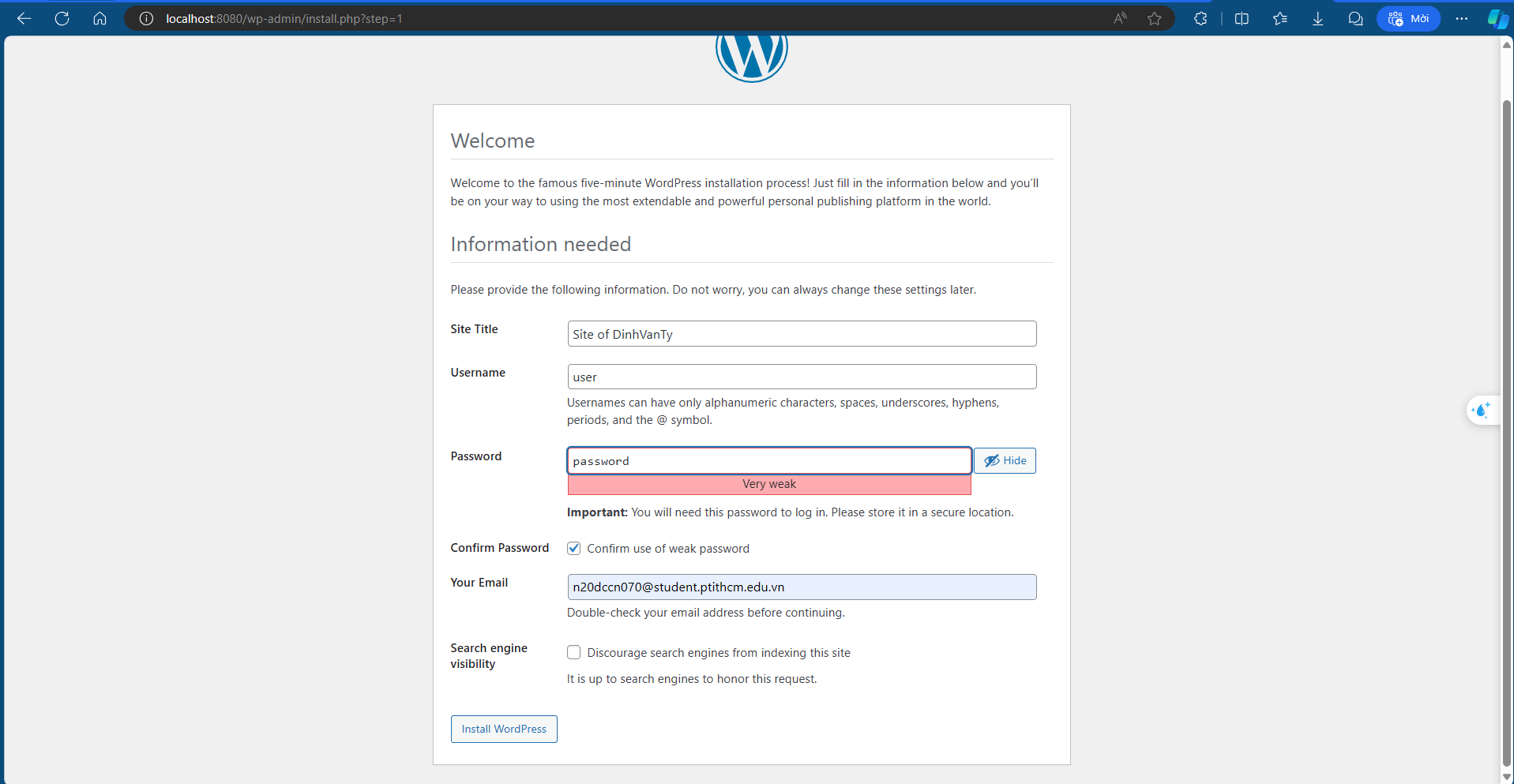
* Quá trình chạy này có thể mất 5-10 phút ở lần khởi động đầu tiên
* Trong trường hợp không thay đổi các tham số nào và để mặc định như trên
  + Truy cập vào **website**: [*http://localhost:8080/*](http://localhost:8082/)
  + Truy cập vào **phpadmin**: [*http://localhost:8080/*](http://localhost:8082/database/)

### **Thiết lập CMS Wordpress**

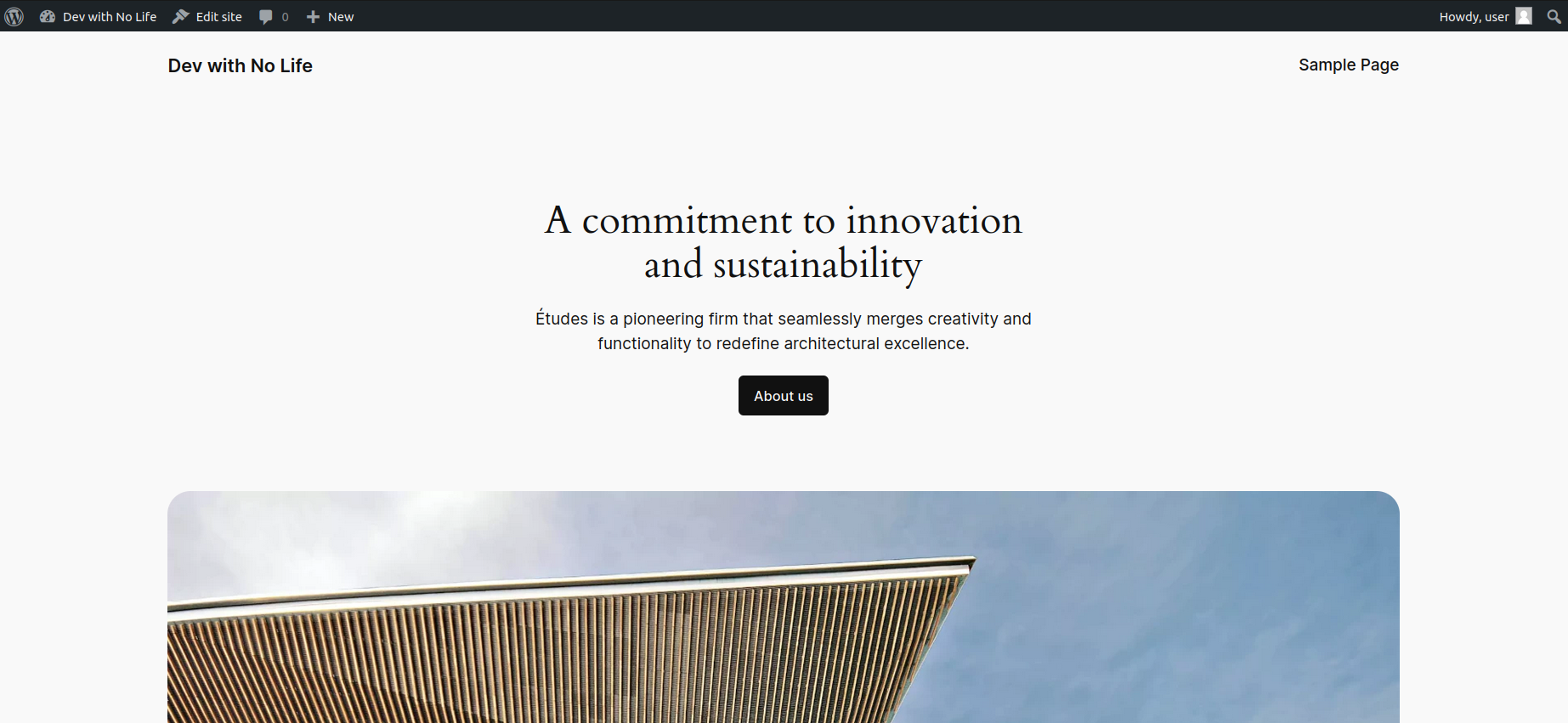
* + Truy cập vào [*website*](http://localhost:8082/) để tiến hành thiết lập
  + Thiết lập ngôn ngữ



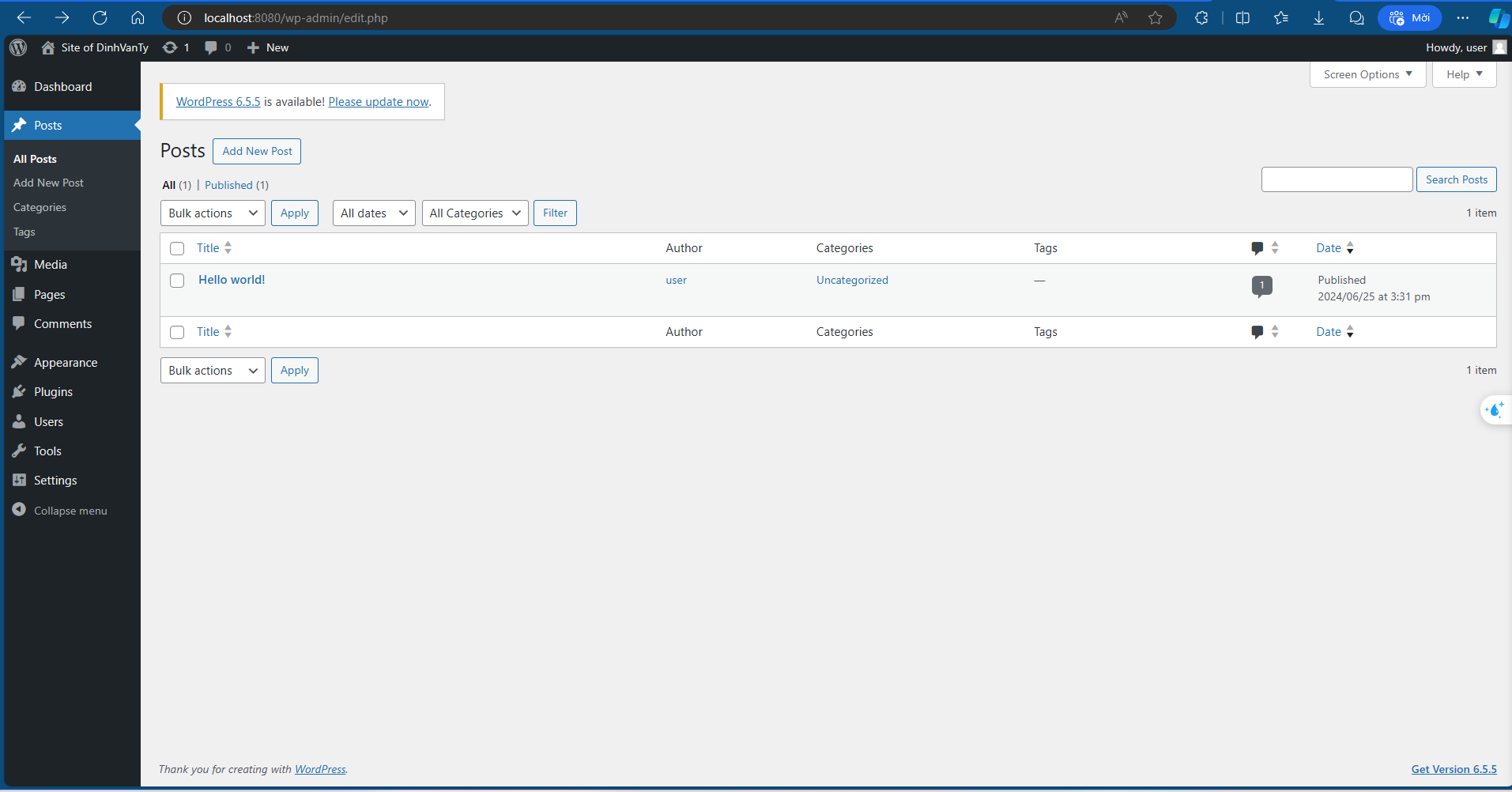
* + Thiết lập đăng nhập admin:



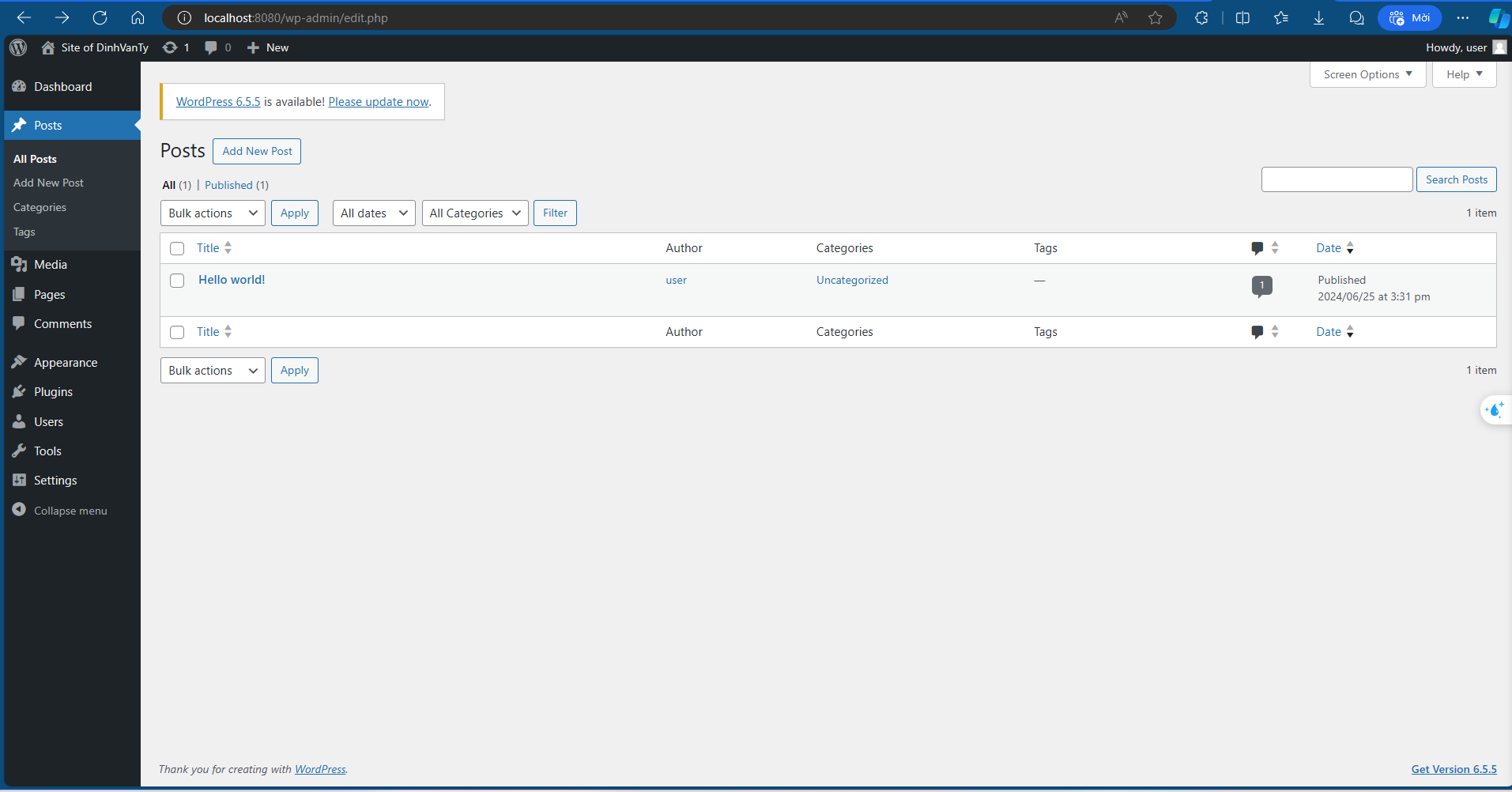
* + Sau khi cài đặt và [*đăng nhập*](http://localhost:8082/wp-login.php) vào tài khoản vừa được tạo ta được giao diện dưới



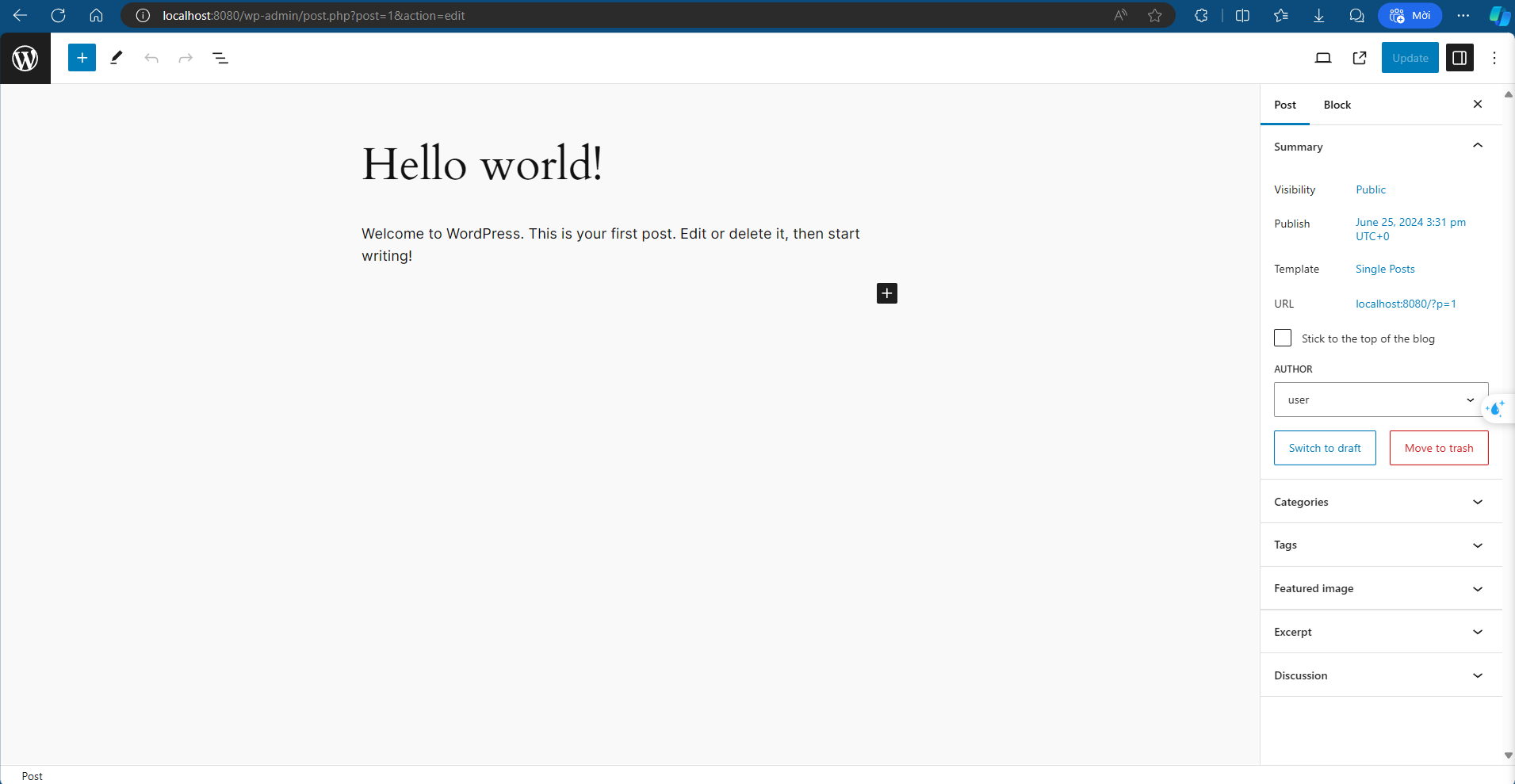
* + Giao diện của admin:



* + Giao diện quản lý blog trên trang web

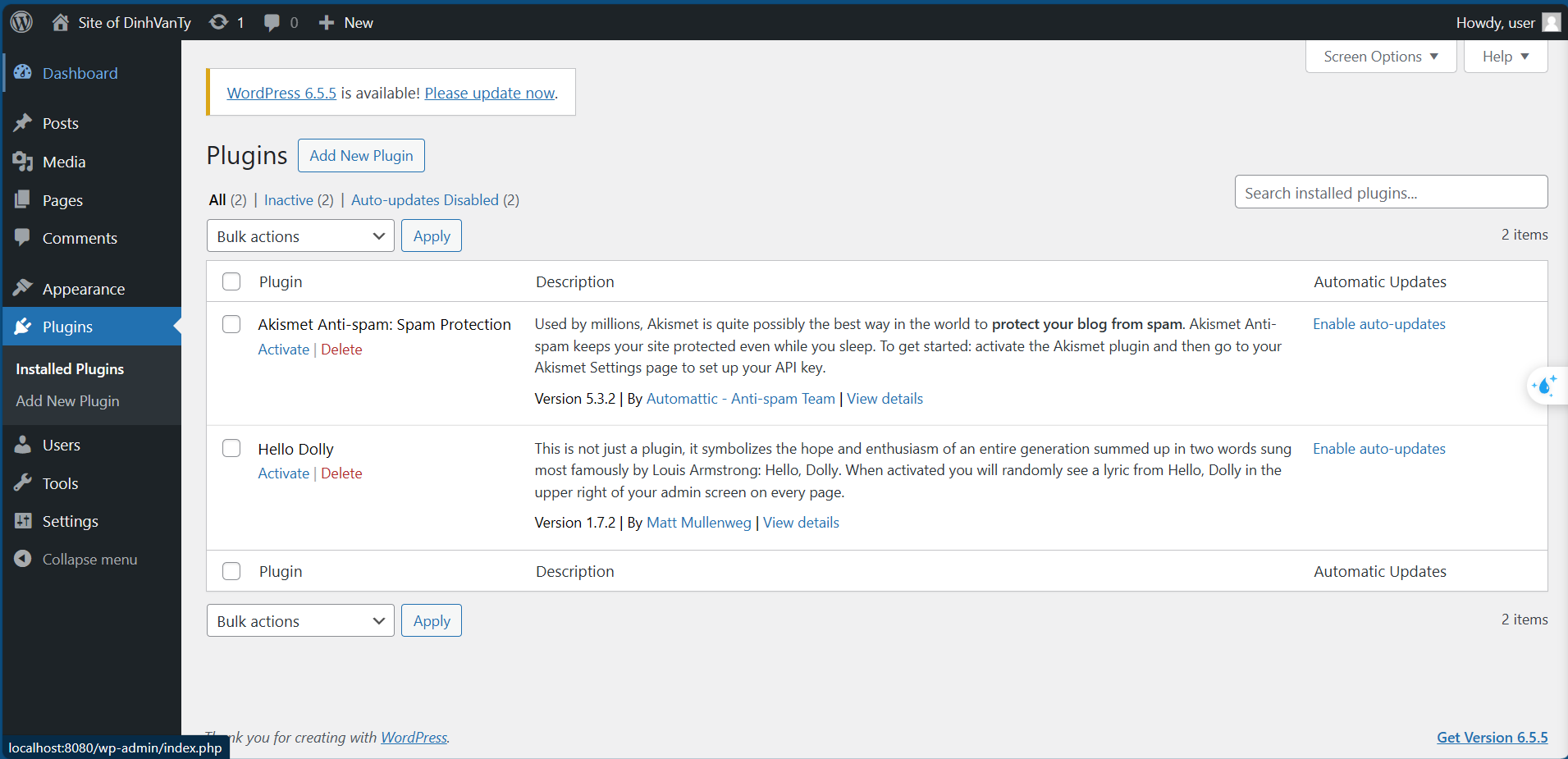


* + Giao diện blog

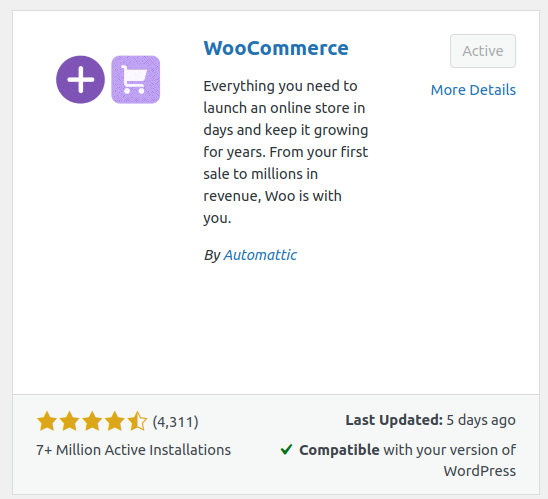


### **Thiết lập Ecommerce Plugin - WooCommerce:**

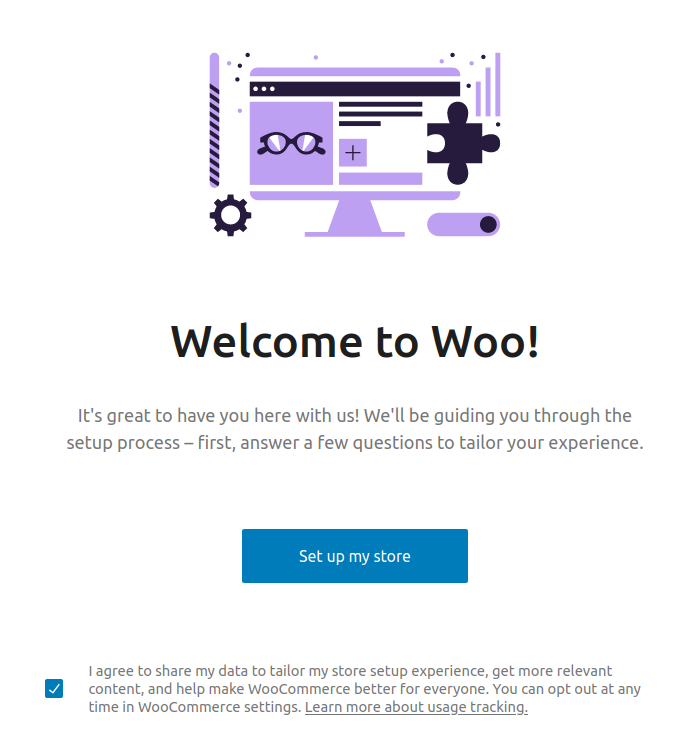
* + Ngay sau khi cài đặt hoàn tất **CMS - Wordpress** tiến hành vào plugins và tìm plugins **WooCommerce** (Plugins > Add New Plugins)



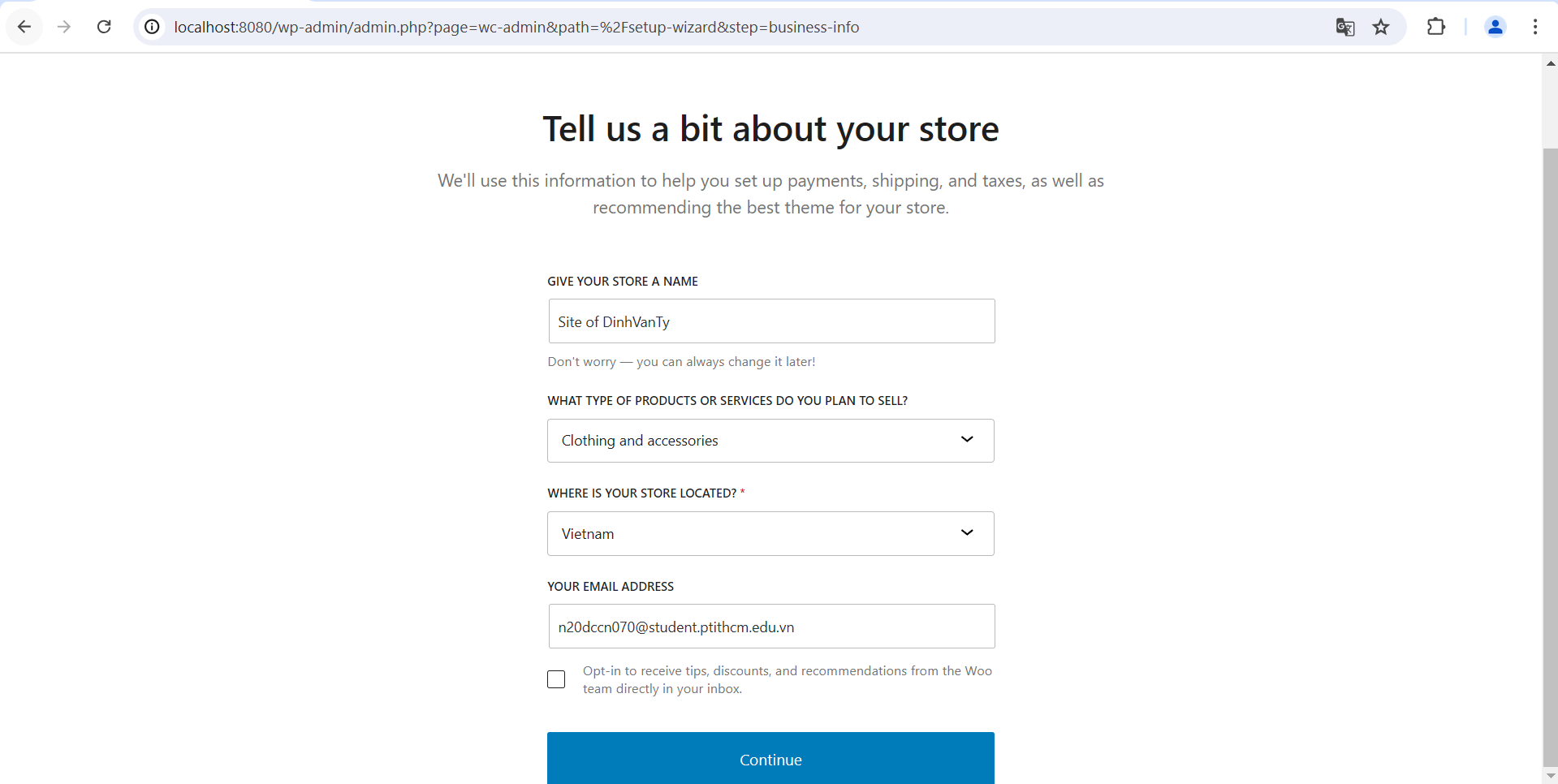
* + Cài đặt và active



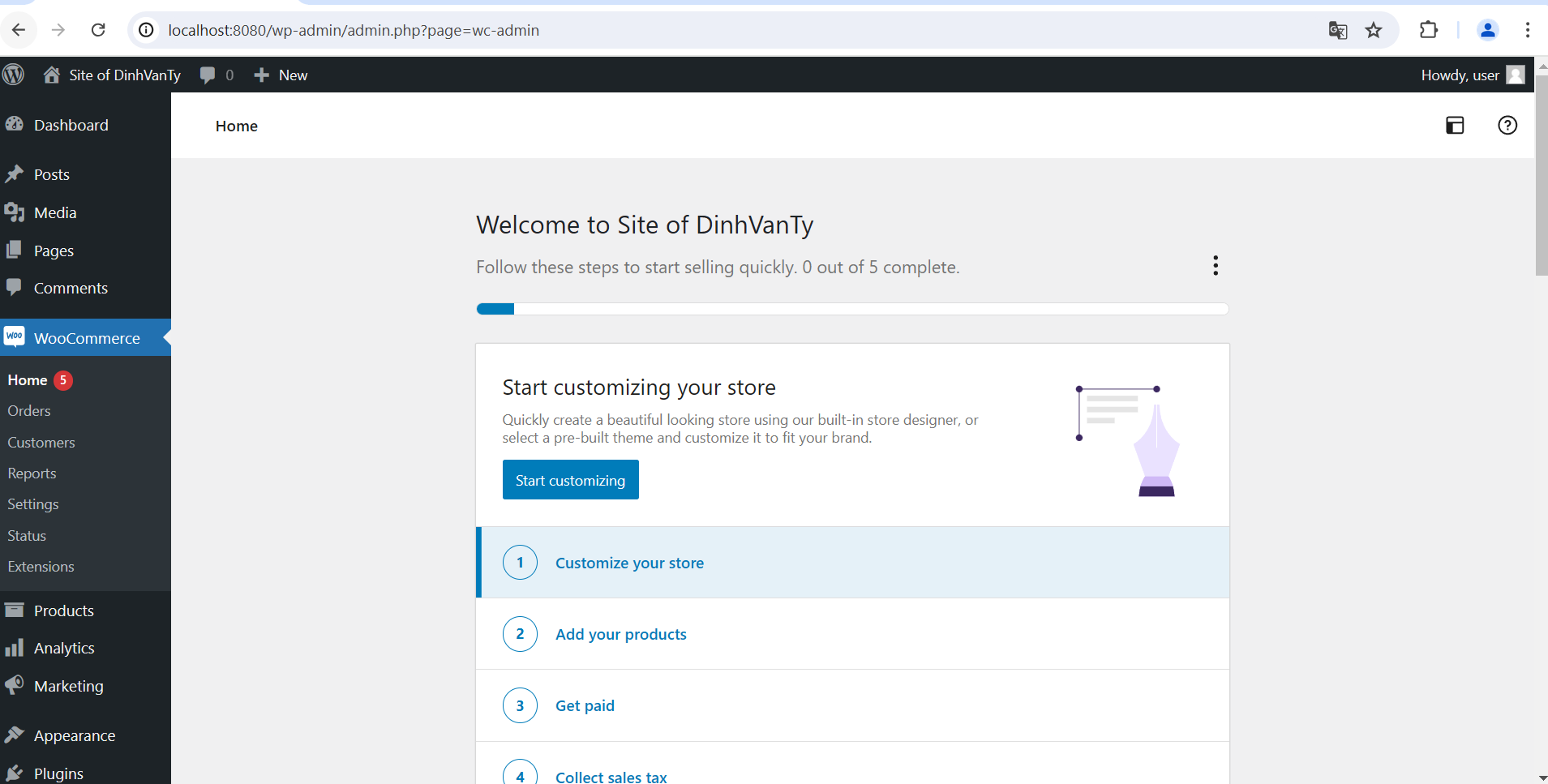
* + Ở phần menu bấm vào **WooCommerce** tiến hành setup



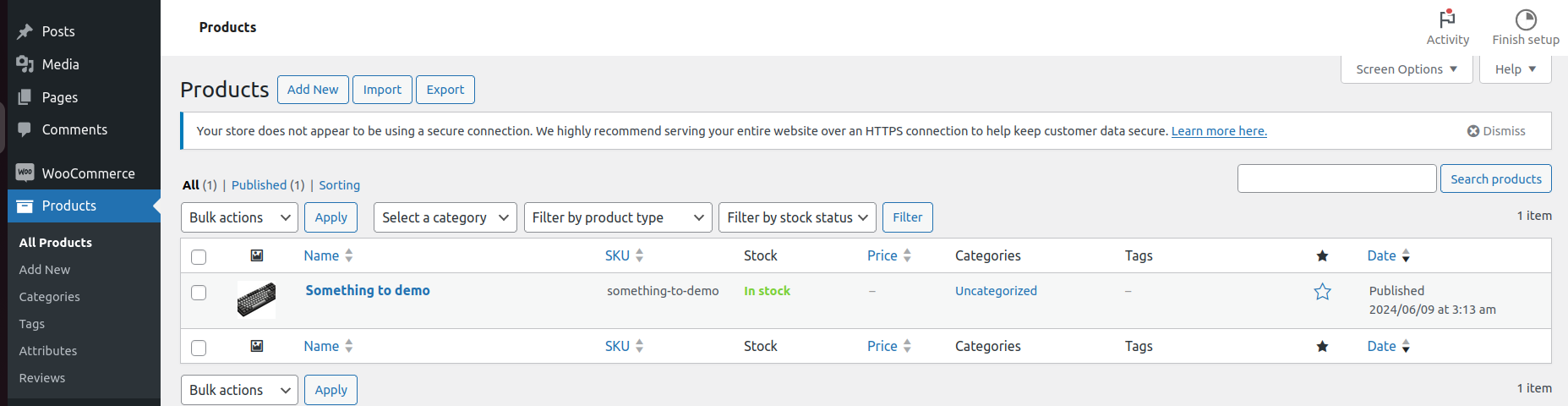
* + Setup các thông tin cần thiết để có thể dùng được **WooCommerce**



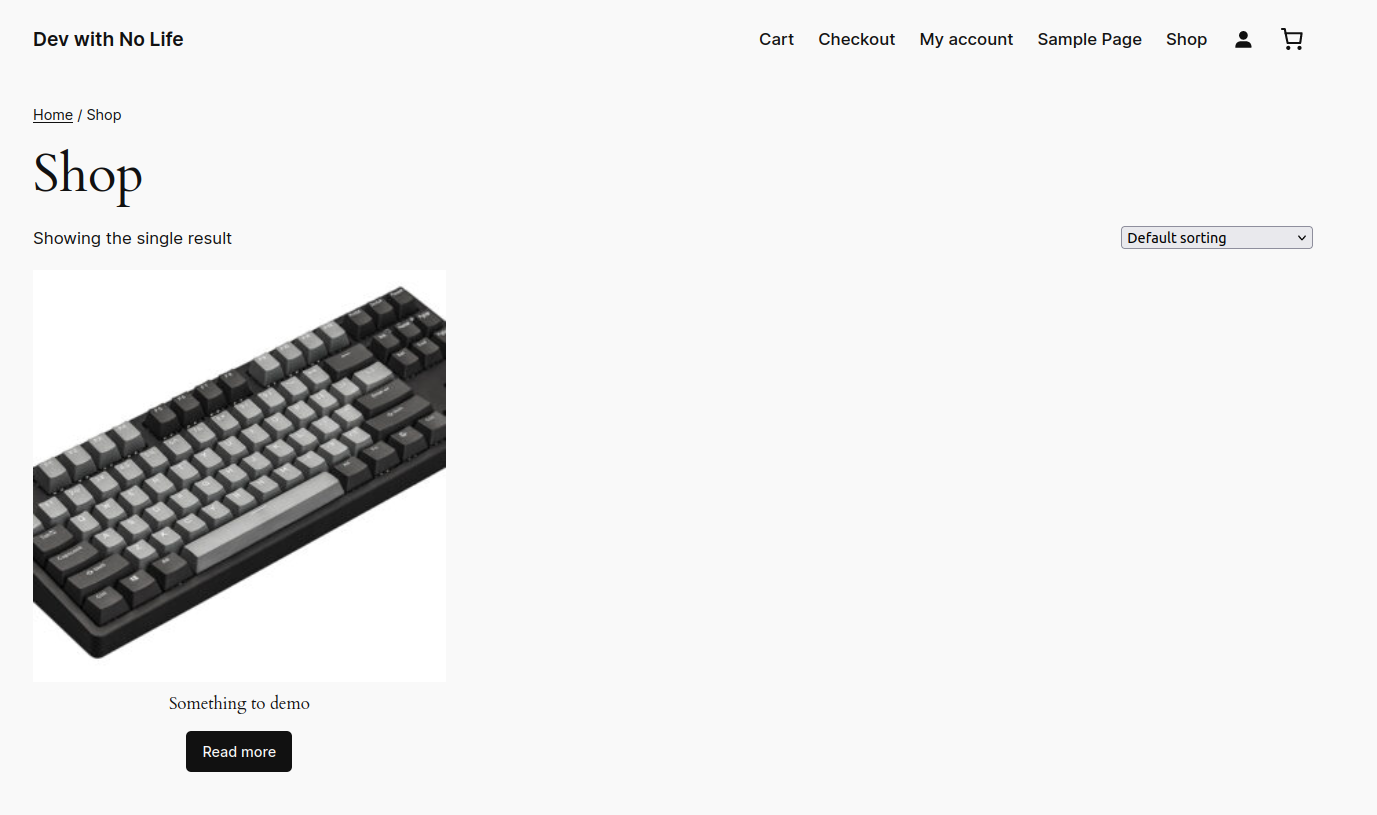
* + Customize các thông tin về store



* + Giao diện quản lý sản phẩm



* + Giao diện sản phẩm trên website

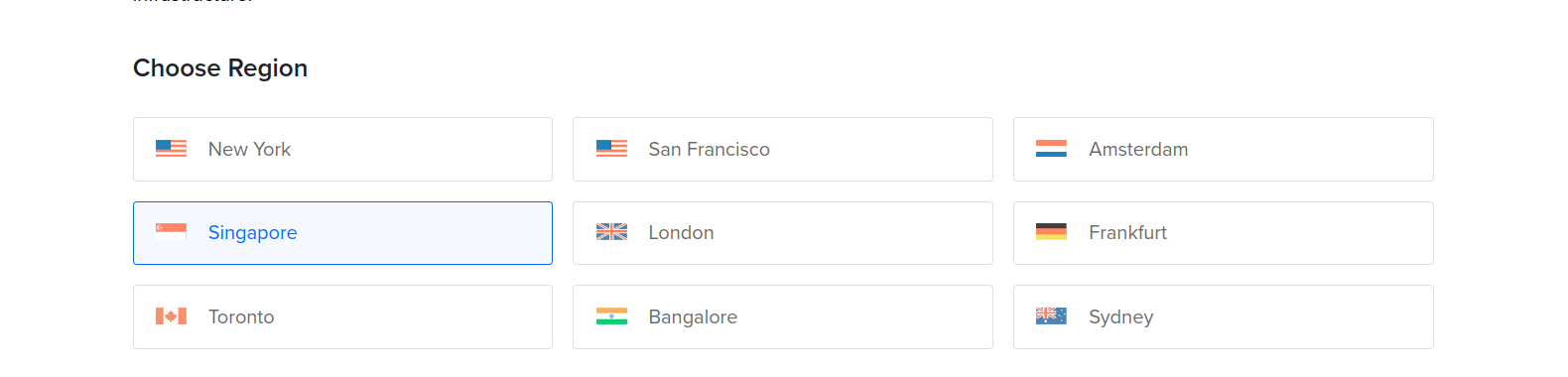


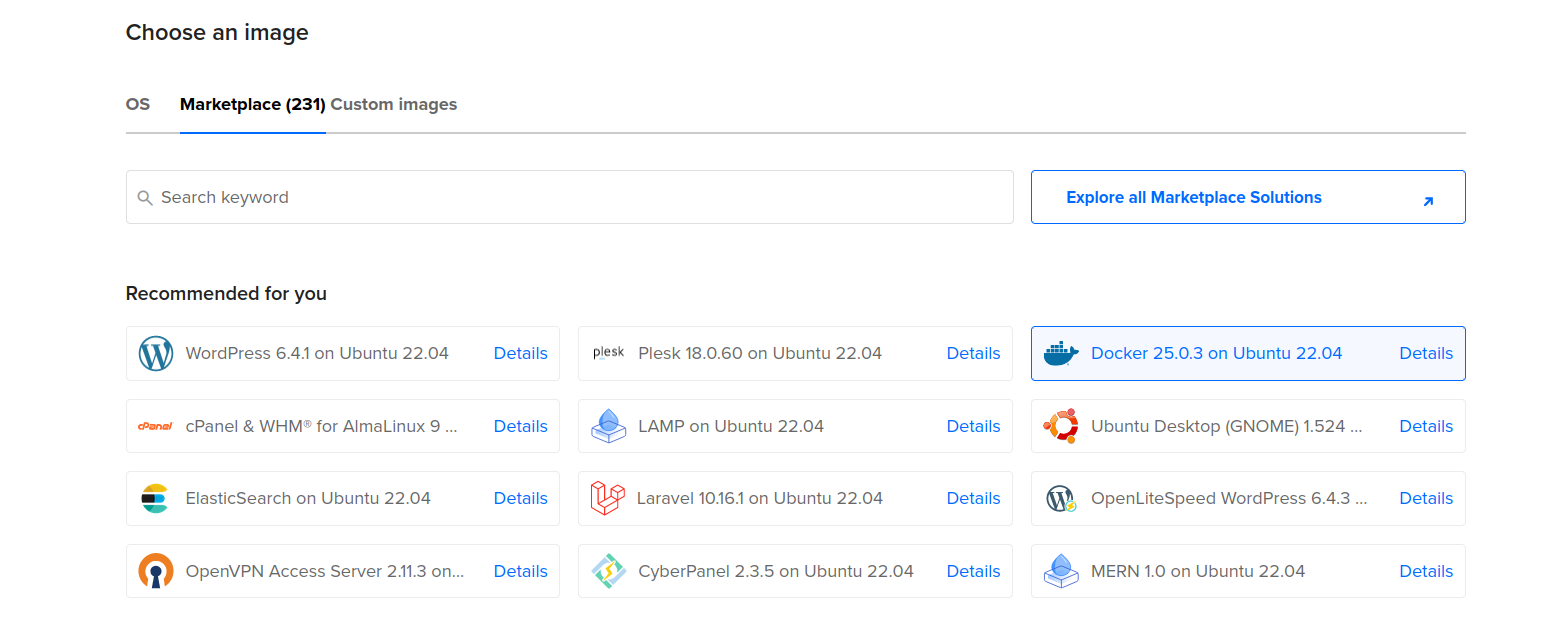
## **Triển khai sản phẩm trên môi trường Testing:**

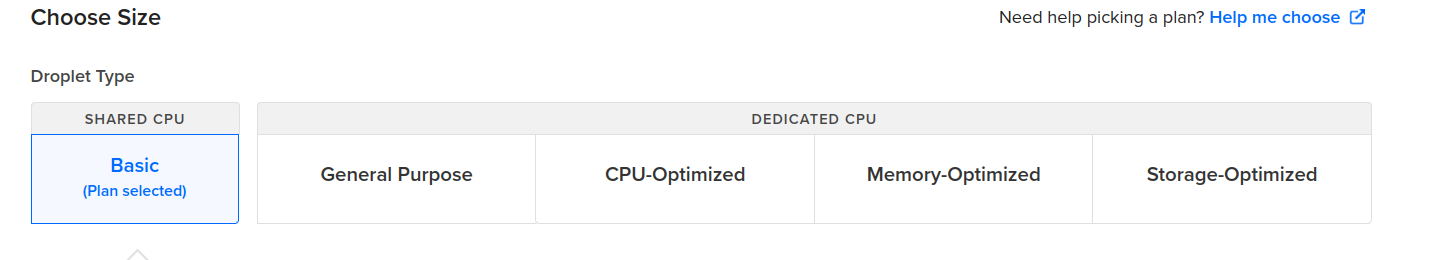
* Việc triển khai trên môi trường sử dụng cloud **Digital Ocean** với mục đích triển khai sản phẩm bằng **Github actions** CI/CD

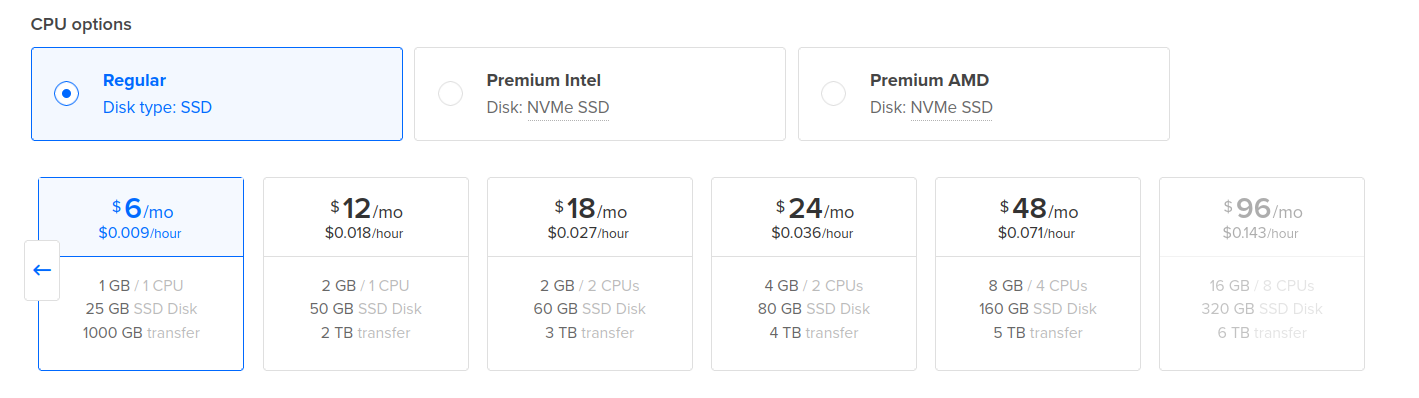
### **Triển khai trên môi trường cloud Digital Ocean**

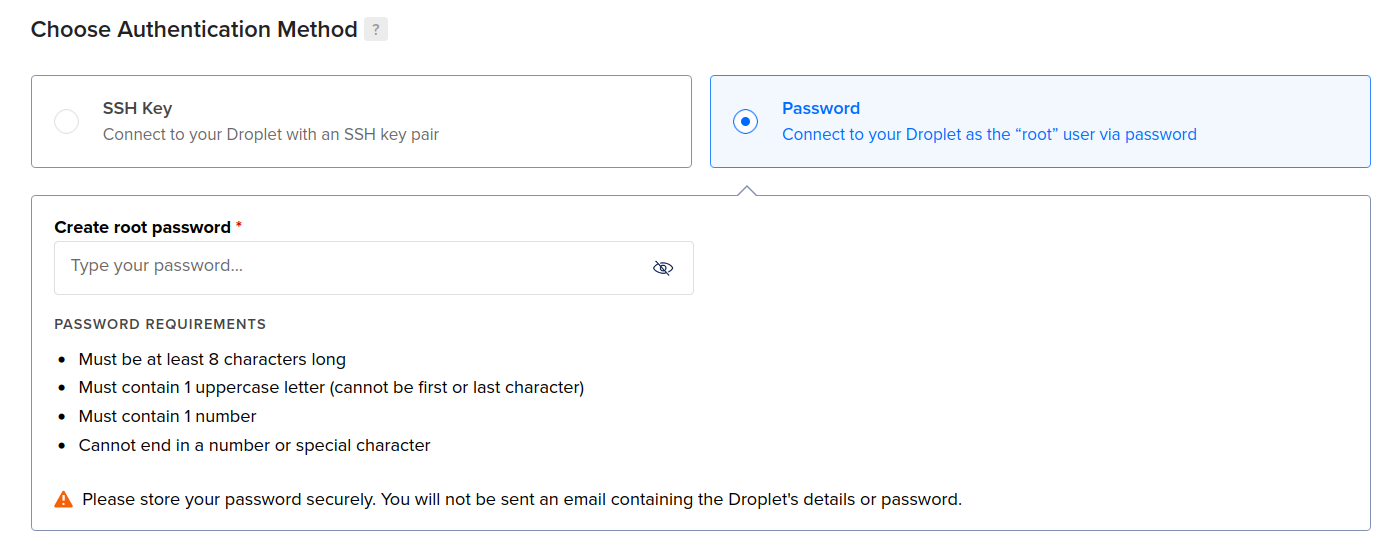
* + Tạo tài khoản / Đăng nhập **Digital Ocean**
  + Tạo một **Droplet** và chọn các config như sau:





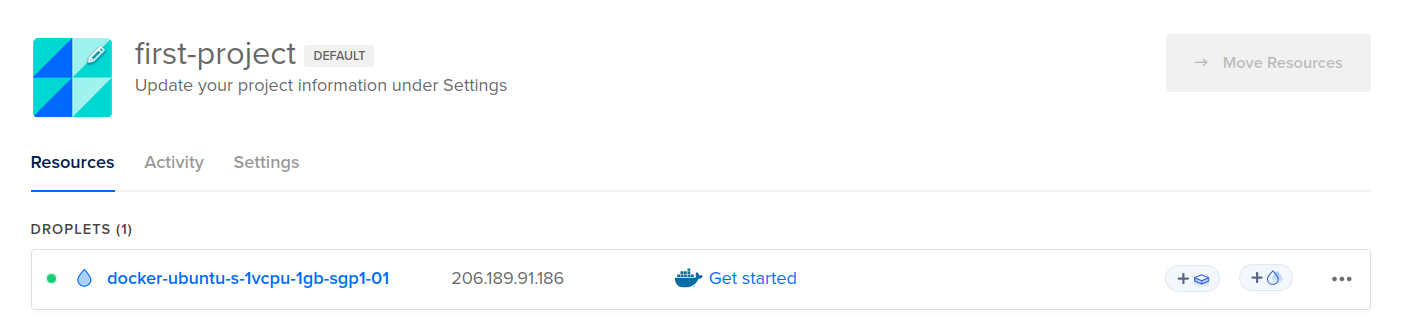






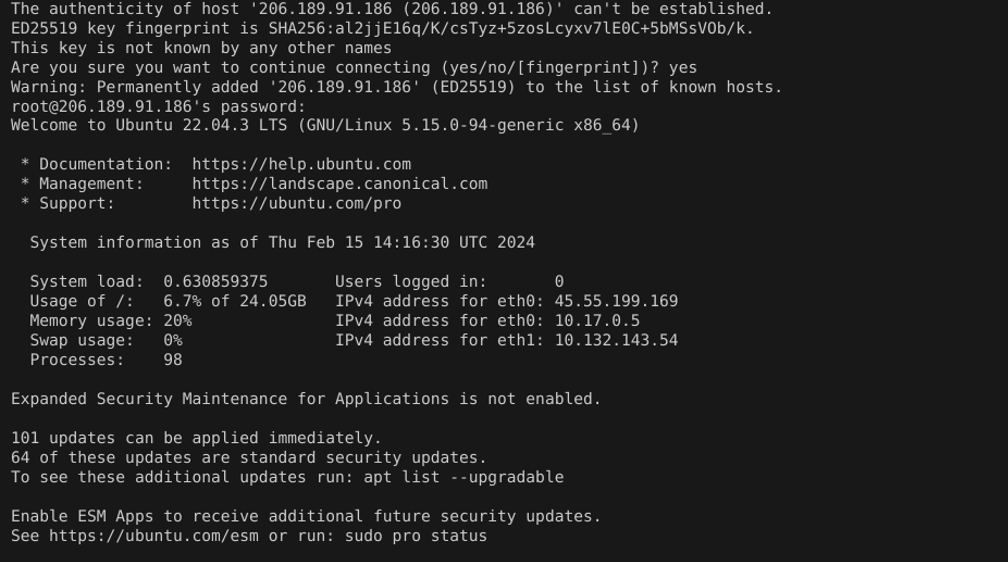


* + Lấy public IP vừa mới tạo và tiến hành đăng nhập vào **droplet**



* + Đăng nhập vào **Droplet** bằng lệnh dưới:

sudo ssh 206.189.91.186

* + Nhập mật khẩu đã tạo ở droplet và có kết quả như hình dưới:
  + Cấp quyền cho người dùng

Vì **Github** không khuyến khích trong việc triển khai CI/CD khi đang ở root nên ta cần tạo ra người dùng.

* + Tạo **user** (username ở phần hướng dẫn này là **tydinh**)

adduser tydinh

* + Cấp quyền cho người dùng
  + usermod –aG sudo tydinh
  + Cấp phép cho người dùng này tạo **docker**
    - Nếu đứng ở root có thể nhận thấy user không thuộc group với **docker** nên sẽ không thể dùng được vì vậy cần nhóm với group docker cho **user** này (ở đây là **tydinh**)
    - (có thể kiểm chứng bằng lệnh cat /etc/group)

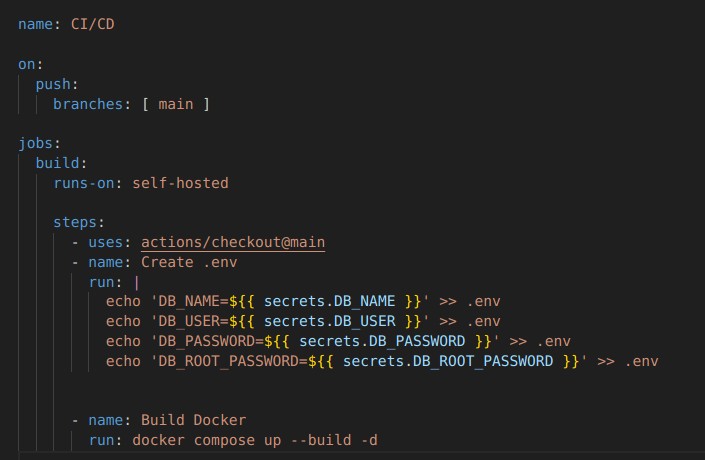
sudo usermod –aG docker tydinh

* + Đăng nhập vào **user** mới tạo (ở hướng dẫn này là tài khoản **tydinh**)

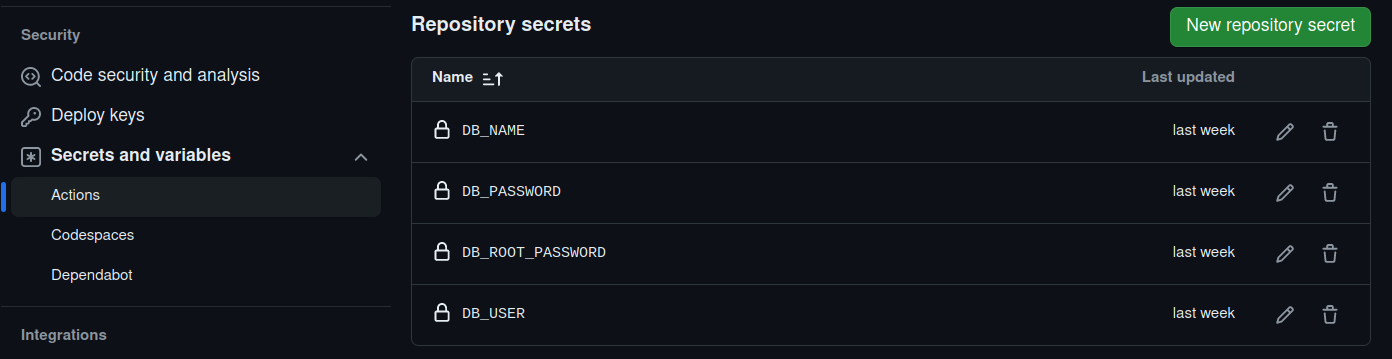
sudo su -tydinh

### **Thiết lập CI/CD**

* + Trong thư mục *./.github/workflows/ main.yml*. Project github định dạng workflow như sau:

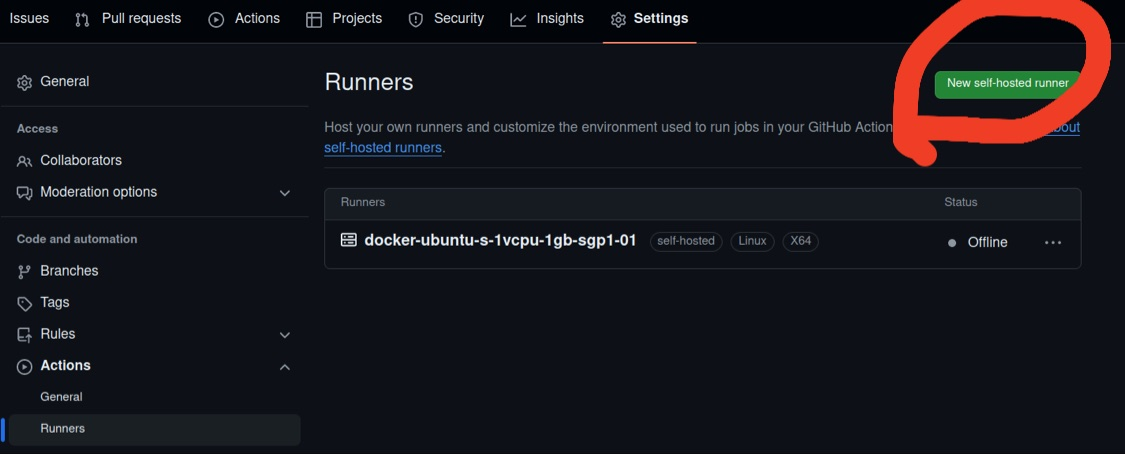


* + các giá trị enviroment được lưu trữ trong **Github Security** với các giá trị như sau:

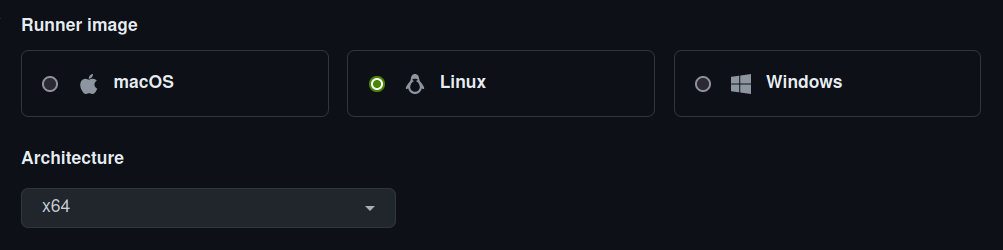


*(Note: với mục đích demo CI/CD nên các giá trị ở đây hiện giờ đang cùng giá trị với file* ***.env****)*

* + Tạo **runner** trên **github**

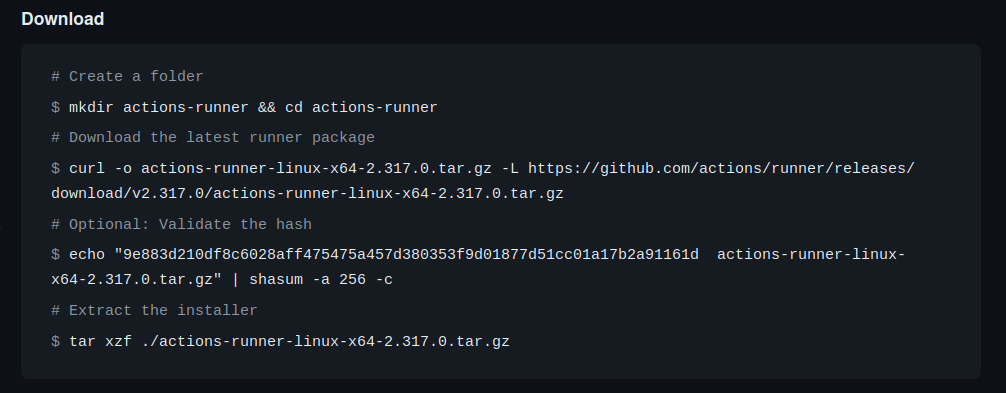


* + Tạo **self-hosted runner** với các config như sau:



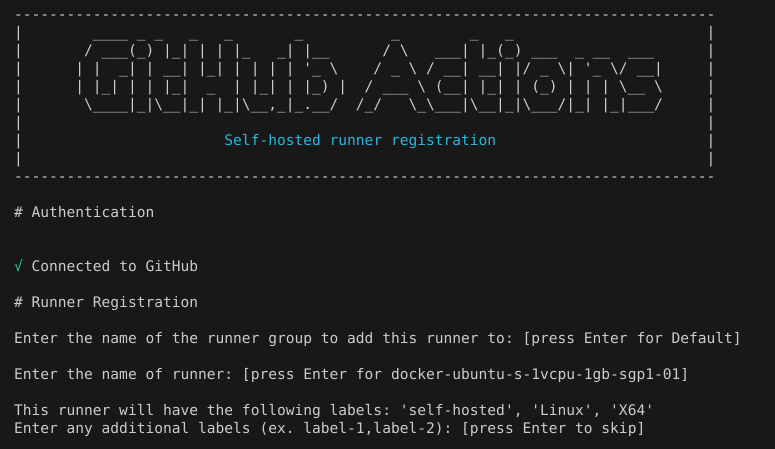
*(Note: phiên bản cloud được dùng là hệ điều hành Linux - Ubuntu 22.04)*

* + Thực hiện các bước được chỉ dẫn của **github**



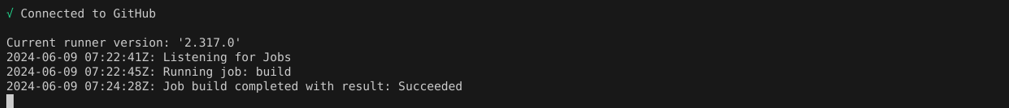
* + Ở phần đăng nhập vào **user** (**tydinh**) trong phần **cmd**, chạy các lệnh do github cung cấp để chạy **runner**



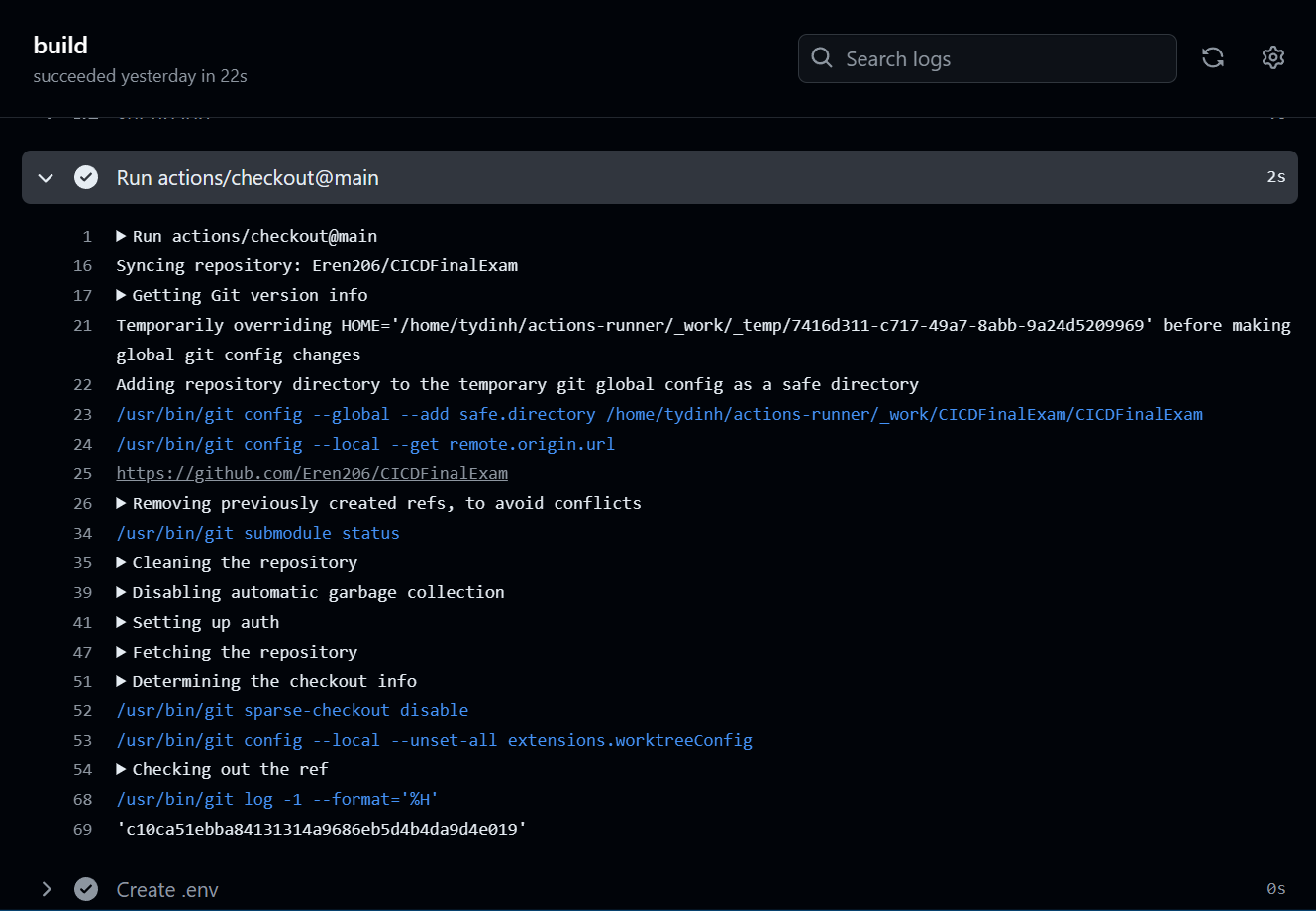


### **Kết quả demo**

* + **Runner** đã chạy và thông báo mỗi khi có sự thay đổi lên **main**



* + Kết quả:



# **Tài liệu đính kèm**

* File **.env** sample:

DB\_NAME=wordpress

DB\_USER=user

DB\_PASSWORD=password

DB\_ROOT\_PASSWORD=password

* Link sản phẩm: https://github.com/Eren206/CICDFinalExam