**Những điểm chính**

* Cài đặt GitLab trên Ubuntu 20.04 cần: user sudo, firewall, tối thiểu 4 CPU/4GB RAM và một tên miền.
* Cài đặt GitLab trên Ubuntu 20.04 gồm 6 bước: cài đặt dependency, cài GitLab, thiết lập firewall, chỉnh cấu hình, cấu hình web và quản lý đăng ký.

**Điều kiện để cài đặt và cấu hình Gitlab trên Ubuntu 20.04**

Để làm theo với hướng dẫn này, bạn sẽ cần:

* [Ubuntu Server 20.04](https://vietnix.vn/ubuntu-20-04/) với non-root user có quyền sudo và [firewall](https://vietnix.vn/firewall-la-gi/).
* Cấu hình phần cứng cho máy chủ chạy [Gitlab](https://vietnix.vn/gitlab-la-gi/) được khuyến nghị sử dụng là: 4 core CPU và 4GB RAM (Có thể bạn dùng swap RAM, nhưng việc này không được khuyến khích).
* Một tên miền trỏ vào server của bạn. Hướng dẫn này sẽ sử dụng tên your\_domain làm ví dụ, nhưng bạn hãy nên thay thế bằng [tên miền](https://vietnix.vn/dang-ky-ten-mien/) của bạn. (hoặc fake hosts)

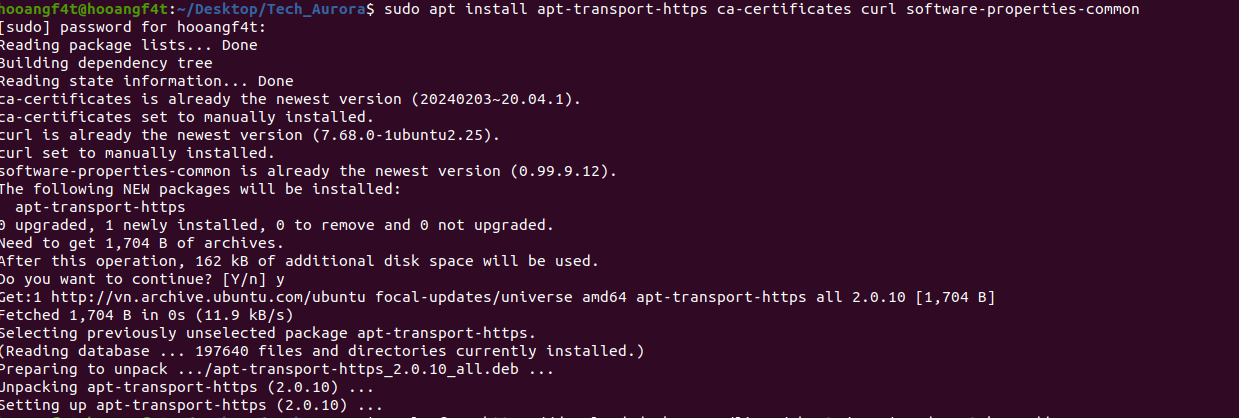
**Bước 1: Cài đặt Docker**

* Cập nhật package source của hệ thống:

sudo apt update

* Cài đặt một số gói cần thiết nhằm giúp apt có thể sử dụng package qua HTTPS:

sudo apt install apt-transport-https ca-certificates curl software-properties-common



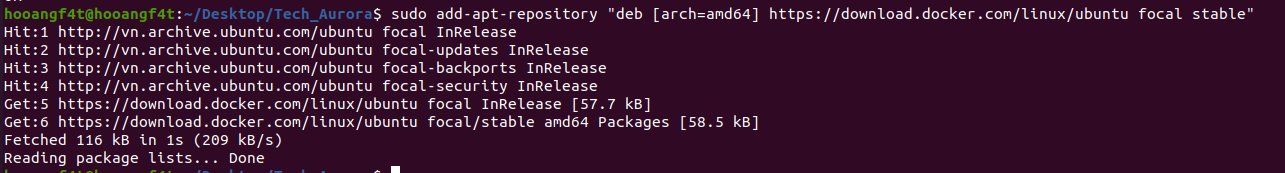
* Thêm mới GPG Key của Docker:

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -



* Thêm mới Docker Repository vào APT:

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu focal stable"

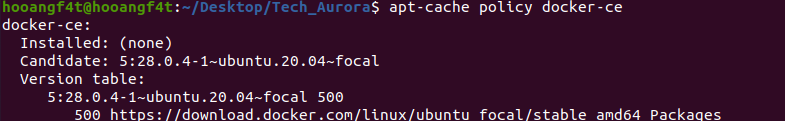


Bước này sẽ cũng thêm mới các Repo vào package database của hệ thống.

* Kiểm tra xem việc thay thế repo mới nhất với repo mặc định của hệ thông xem đã được thay thế chưa

apt-cache policy docker-ce

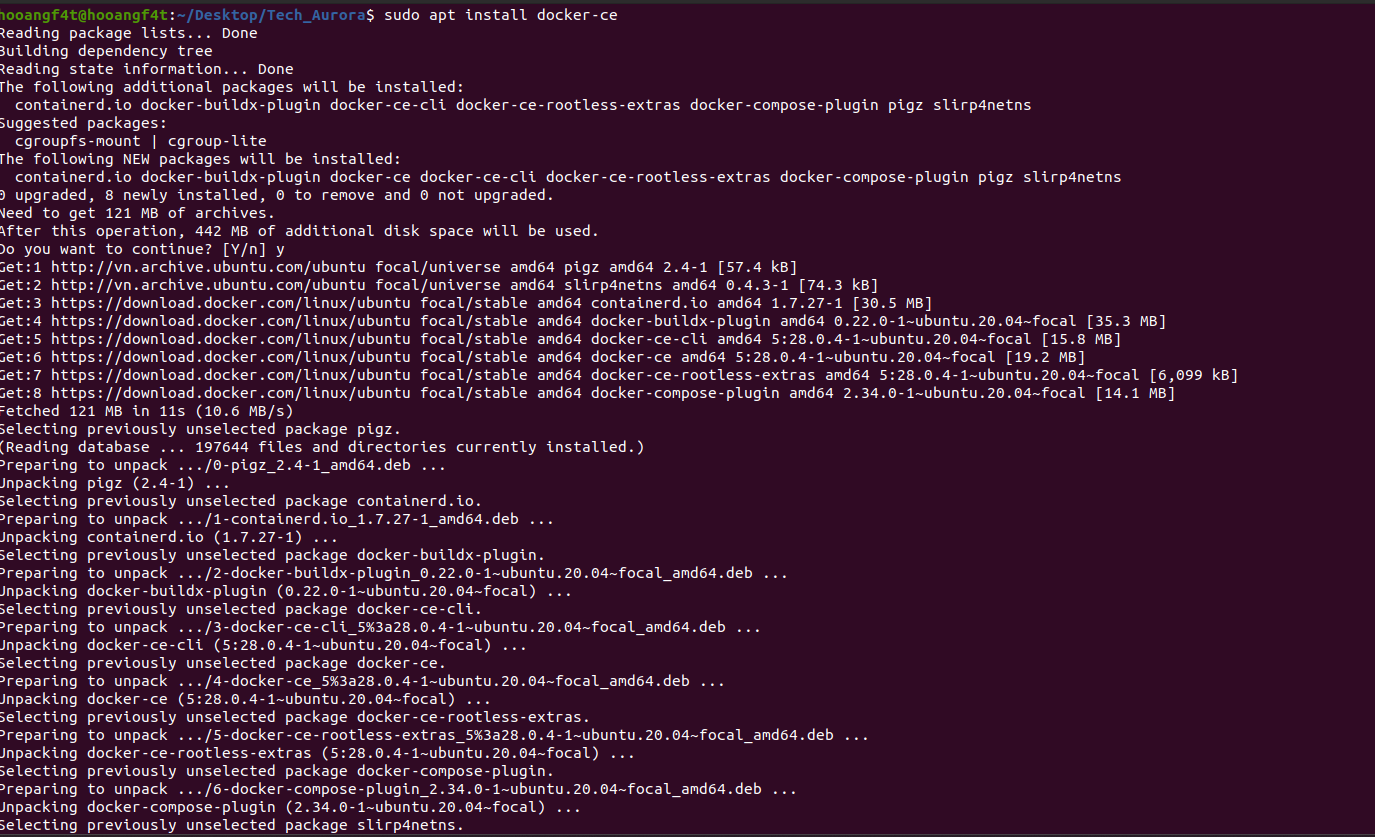
Kết quả trả về sẽ như sau, phiên bản Docker có thể thay đổi tùy vào thời điểm cài đặt:



Trong output trên, lưu ý rằng docker-ce chưa được cài đặt, nhưng đã có sẵn phiên bản 5:19.03.9~3-0~ubuntu-focal trong repo sẵn sàng để cài đặt.

* Cài đặt Docker:

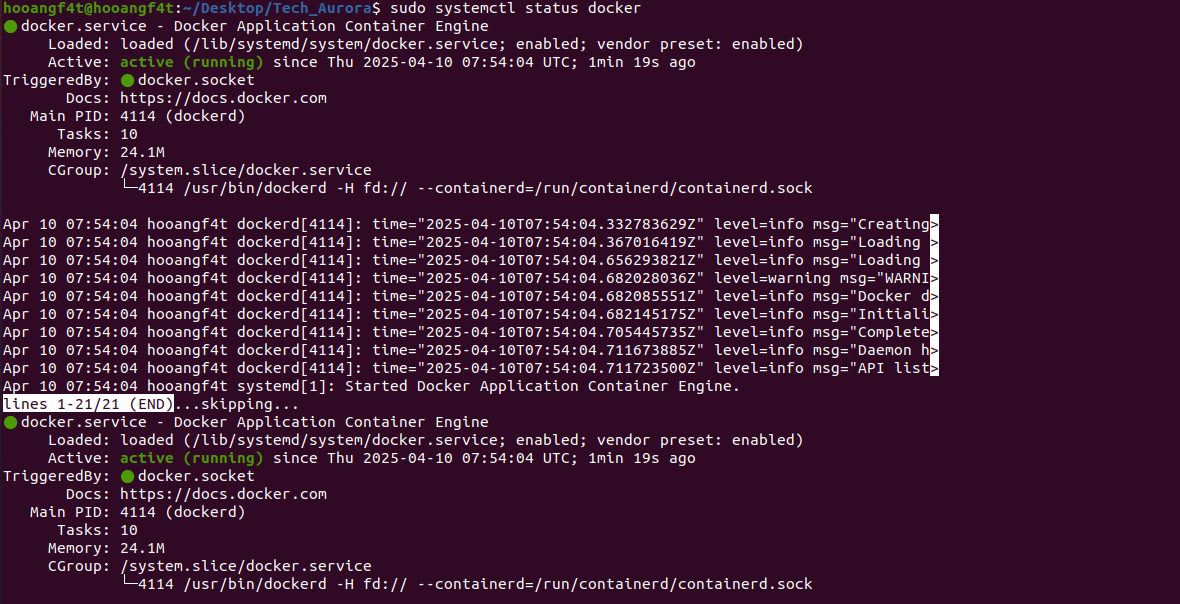
sudo apt install docker-ce



Sau bước này, Docker sẽ được cài đặt, deamon sẽ được khởi động (Docker Service chạy ngầm) và process sẽ được thêm vào boot (khởi động cùng hệ thống). Để kiểm tra xem Docker Deamon đã được khởi động hay chưa, chúng ta sử dụng lệnh sau:

sudo systemctl status docker

Service khi hoạt động bình thường sẽ trả về kết quả như sau:

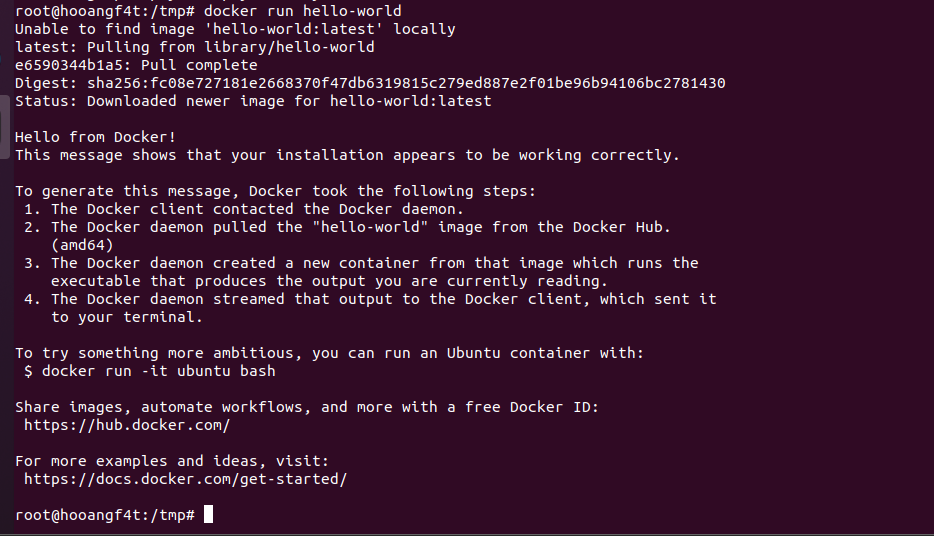


Trong trường hợp có lỗi xảy ra, giá trị Active sẽ là failed. Đối với các bản cài Docker hiện nay, gói cài đặt thường không chỉ bao gồm mỗi Docker Service (Deamon) mà sẽ bao gồm các tiện ích khác như Docker Command Line hoặc Docker Client để chúng ta có thể tương tác với Docker Service thông qua CLI. Trong các phần sau chúng ta sẽ tìm hiểu về cách sử dụng Docker Command Line để tương tác với Docker Service.

Để xác minh cài đặt docker, hãy chạy lệnh docker \hello-world\ bên dưới.

docker run hello-world

Lúc này bạn sẽ nhận được thông báo \hello-world\ từ docker như bên dưới.



Bây giờ đã sẵn sàng để cài đặt GitLab bằng docker container và docker-compose.

Trước khi cài đặt Gitlab, chúng tôi cần cài đặt một số gói nhất định cần thiết trong quá trình hướng dẫn.

sudo apt install ca-certificates curl openssh-server apt-transport-https gnupg lsb-release -y

Một số gói này có thể đã được cài đặt sẵn trên hệ thống của bạn.

Gitlab use default SSH port sẽ xung đột với port SSH của hệ thống. Để có kết quả tốt nhất, tốt hơn là thay đổi cổng mặc định của hệ thống.

Để thực hiện công việc này, hãy mở   /etc/ssh/sshd\_config tệp để chỉnh sửa.

sudo gedit /etc/ssh/sshd\_config

Tìm dòng sau, xóa dấu thăng (#) ở phía trước và thay đổi giá trị từ 22 thành bất kỳ giá trị nào bạn chọn. Đối với hướng dẫn của tôi, tôi đã chọn 2425 bằng cách thay đổi

Lưu tệp

Khởi động lại SSH service.

sudo systemctl restart sshd

# Gitlab

Tạo Docker ổ đĩa thư mục.

sudo mkdir /srv/gitlab -p

Tạo một thư mục cho Docker file editor.

mkdir ~/gitlab-docker

Chuyển sang thư mục.

cd ~/gitlab-docker

Tạo một môi trường biến tệp và mở nó để chỉnh sửa.

gedit .env

GITLAB\_HOME=/srv/gitlab

Bộ chứa Gitlab sử dụng các ổ đĩa được gắn trên máy chủ để lưu trữ liên tục dữ liệu. Bảng sau đây hiển thị bộ định vị cục bộ của các Gitlab thư mục đến vị trí của các bộ chứa và cách sử dụng ứng dụng tương thích của chúng.

|  |  |  |
| --- | --- | --- |
| $GITLAB\_HOME/dữ liệu | /var/opt/gitlab | Để lưu trữ ứng dụng dữ liệu. |
| $GITLAB\_HOME/nhật ký | /var/log/gitlab | Để lưu trữ nhật ký. |
| $GITLAB\_HOME/cấu hình | /etc/gitlab | Để lưu trữ cấu hình Gitlab. |

**Cài đặt Gitlab bằng Docker Compose**

Đảm bảo rằng bạn đang ở trong thư mục Docker soạn của Gitlab.

Tạo và mở Docker cấu hình tệp để chỉnh sửa chỉnh sửa.

sudo gedit docker-compose.yml

version: '3.6'

services:

web:

image: 'gitlab/gitlab-ee:latest'

container\_name: 'gitlab-congdonglinux'

restart: always

hostname: 'gitlab.example.com'

environment:

GITLAB\_OMNIBUS\_CONFIG: |

external\_url 'https://gitlab.example.com'

nginx['ssl\_certificate'] = "/etc/gitlab/ssl/gitlab.example.com.crt"

nginx['ssl\_certificate\_key'] = "/etc/gitlab/ssl/gitlab.example.com.key"

gitlab\_rails['smtp\_enable'] = true

gitlab\_rails['smtp\_address'] = "email-smtp.us-west-2.amazonaws.com"

gitlab\_rails['smtp\_user\_name'] = "SESUsername"

gitlab\_rails['smtp\_password'] = "SESKey"

gitlab\_rails['smtp\_domain'] = "example.com"

gitlab\_rails['smtp\_enable\_starttls\_auto'] = true

gitlab\_rails['smtp\_port'] = 587

gitlab\_rails['smtp\_authentication'] = "login"

gitlab\_rails['gitlab\_email\_from'] = 'gitlab@example.com'

gitlab\_rails['gitlab\_email\_reply\_to'] = 'noreply@example.com'

ports:

- '8080:80'

- '8443:443'

- '2222:22'

- '5878:587'

volumes:

- '${GITLAB\_HOME}/config:/etc/gitlab'

- '${GITLAB\_HOME}/logs:/var/log/gitlab'

- '${GITLAB\_HOME}/data:/var/opt/gitlab'

- '/root/gitlab/ssl:/etc/gitlab/ssl'

shm\_size: '256m'

Chúng ta hãy xem xét tất cả các tùy chọn được xác định trong tệp.

* **hình ảnh**   đề cập đến vị trí hình ảnh Docker của Gitlab trên Dockerhub.
* **container\_name**   cho phép bạn áp dụng nhãn cho vùng chứa docker của mình để sử dụng khi tham chiếu đến vùng chứa trong Docker mạng.
* **khởi động lại**   chính sách khởi động lại chỉ định cho vùng chứa. Đặt   thành vùng chứa **luôn**   có nghĩa nếu thoát sẽ tự động khởi động lại.
* **tên máy chủ**   xác định tên nội bộ máy chủ của vùng chứa hoặc trong trường hợp này là URL nơi Gitlab của bạn sẽ được cài đặt.
* **môi trường**   cung cấp biến   **GITLAB\_OMNIBUS\_CONFIG**   cho phép bạn nhập bất kỳ cấu hình Gitlab cài đặt nào.
* **external\_url**   là miền tên nơi Gitlab của bạn sẽ được cài đặt. Sử dụng   https giao thức SSL Let's Encrypt chỉ cài đặt tự động bảo mật.
* **Chi tiết SMTP**   – chúng tôi đã bao gồm chi tiết SMTP để phiên bản Gitlab có thể gửi email và thông báo quan trọng. Đối với hướng dẫn của chúng tôi, chúng tôi đang sử dụng dịch vụ SES của Amazon. Tuy nhiên, bạn có thể sử dụng bất kỳ dịch vụ nào bạn chọn. Kiểm tra   [tài liệu Gitlab dành riêng cho thư gửi SMTP](https://docs.gitlab.com/omnibus/settings/smtp.html)   để tìm hiểu cách cấu hình chúng.
* ports security container xuất bản các **cổng** hoặc một loạt cổng tới máy chủ. Vì Gitlab cần các cổng 22(SSH), 80(HTTP), 443(HTTPS) và 587(SMTP), nên chúng đã được đưa ra hệ thống. Nếu bạn muốn Gitlab sử dụng một cổng không chuẩn trên máy chủ của mình (có thể vì nó không khả dụng), trước tiên bạn sẽ cung cấp cổng máy chủ và mới đến cổng container. Ví dụ, vì máy chủ của bạn đã sử dụng cổng SSH(22), nên bạn có thể bảo đảm Gitlab sử dụng SSH qua một cổng khác, ví dụ như 3333. Sau đó, bạn sẽ thay đổi   **22:22**   trong tệp thành   **3333:22**  . Bạn cũng cần thêm dòng   gitlab\_rails['gitlab\_shell\_ssh\_port'] = 3333 bên dưới   **GITLAB\_OMNIBUS\_CONFIG**   ở trên.
* **Volume**   xác định các thư mục có trên máy chủ để lưu trữ liên tục dữ liệu. Như đã xác định ở bước 5,   $GITLAB\_HOME giờ đây có thể được sử dụng trong soạn thảo tệp để gắn các thư mục có liên quan vào vùng chứa.
* **shm\_size**   đề cập đến bộ nhớ dùng chung được sử dụng trong vùng chứa. Theo mặc định, Docker phân tích bổ sung 64MB cho chung bộ nhớ thư mục (có thể gắn tại   /dev/shm). Điều này có thể được chứng minh là không đủ cho Prometheus số liệu mà Gitlab tạo ra. Do đó, mức sử dụng bộ nhớ tối thiểu là 256MB docker run run chắc chắn. Bạn có thể tăng giá trị của nó tùy thuộc vào RAM mà hệ thống của bạn có. Ngoài ra, bạn có thể tắt Prometheus số liệu khỏi quản trị khu vực sau khi cài đặt. Chúng ta sẽ khám phá điều này trong bước tiếp theo.

Khởi động vùng chứa Gitlab Docker.

sudo docker compose up -d

Quá trình này sẽ mất vài phút để hoàn tất. Bạn có thể theo dõi tiến trình bằng cách sử dụng Docker cập nhật.

docker logs gitlab-congdonglinux -f

Nhấn   **Ctrl + C**   để thoát khỏi nhật ký theo dõi.

Bạn có thể kiểm tra trạng thái của container Gitlab bằng lệnh sau.

sudo docker ps

Bắt đầu từ Gitlab 14.0, nó tự động tạo mật khẩu gốc và lưu trữ trong   initiall\_root\_password tệp. Có thể tìm thấy tệp này trong   /srv/gitlab/config thư mục. Chạy lệnh sau để xem gốc mật khẩu.

sudo cat /srv/gitlab/config/initial\_root\_password

Bạn sẽ nhận được kết quả tương tự.

# WARNING: This value is valid only in the following conditions

# 1. If provided manually (either via `GITLAB\_ROOT\_PASSWORD` environment variable or via `gitlab\_rails['initial\_root\_password']` setting in `gitlab.rb`, it was provided before database was seeded for the first time (usually, the first reconfigure run).

# 2. Password hasn't been changed manually, either via UI or via command line.

#

# If the password shown here doesn't work, you must reset the admin password following https://docs.gitlab.com/ee/security/reset\_user\_password.html#reset-your-root-password.

Password: j/74gbo4dHdFcexs0k9ts14ME+yL9JCRFDiMp0QN0yQ=

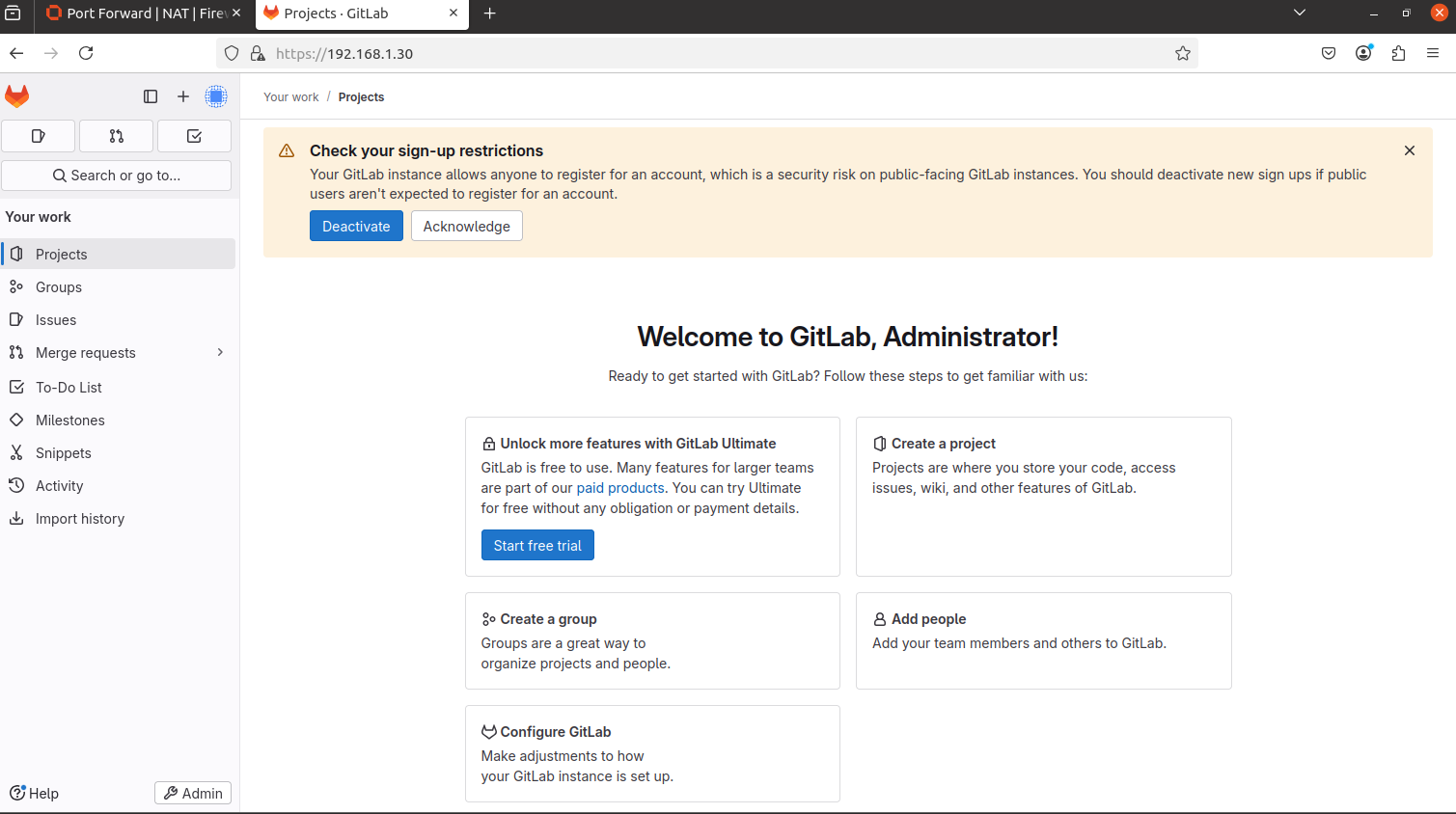
# NOTE: This file will be automatically deleted in the first reconfigure run after 24 hours.  
Sao chép mật khẩu và lưu lại để sử dụng sau. Bây giờ mọi thứ đã được thiết lập, chúng tôi có thể tiến hành cấu hình.

**Cấu hình Gitlab**

**Truy cập Gitlab**

Mở URL   https://gitlab.example.com trong trình duyệt của bạn và bạn sẽ nhận được màn hình đăng nhập sau đó.

Nhập   root tên người dùng và mật khẩu mà bạn đã lấy ở bước trước để đăng nhập vào bảng điều khiển Gitlab của bạn. Khi đăng nhập, bạn sẽ được đưa vào bảng điều khiển màn hình sau.

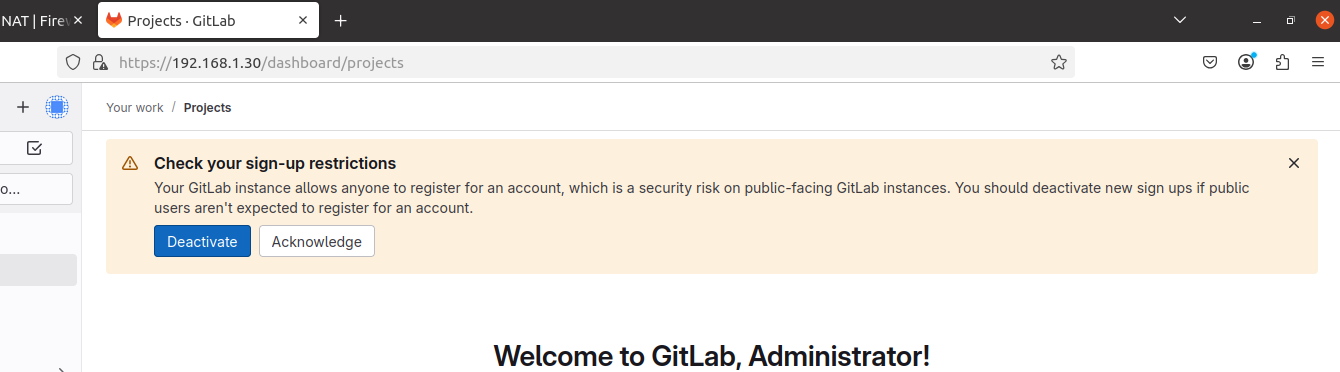
Giao diện sau khi đăng nhập  


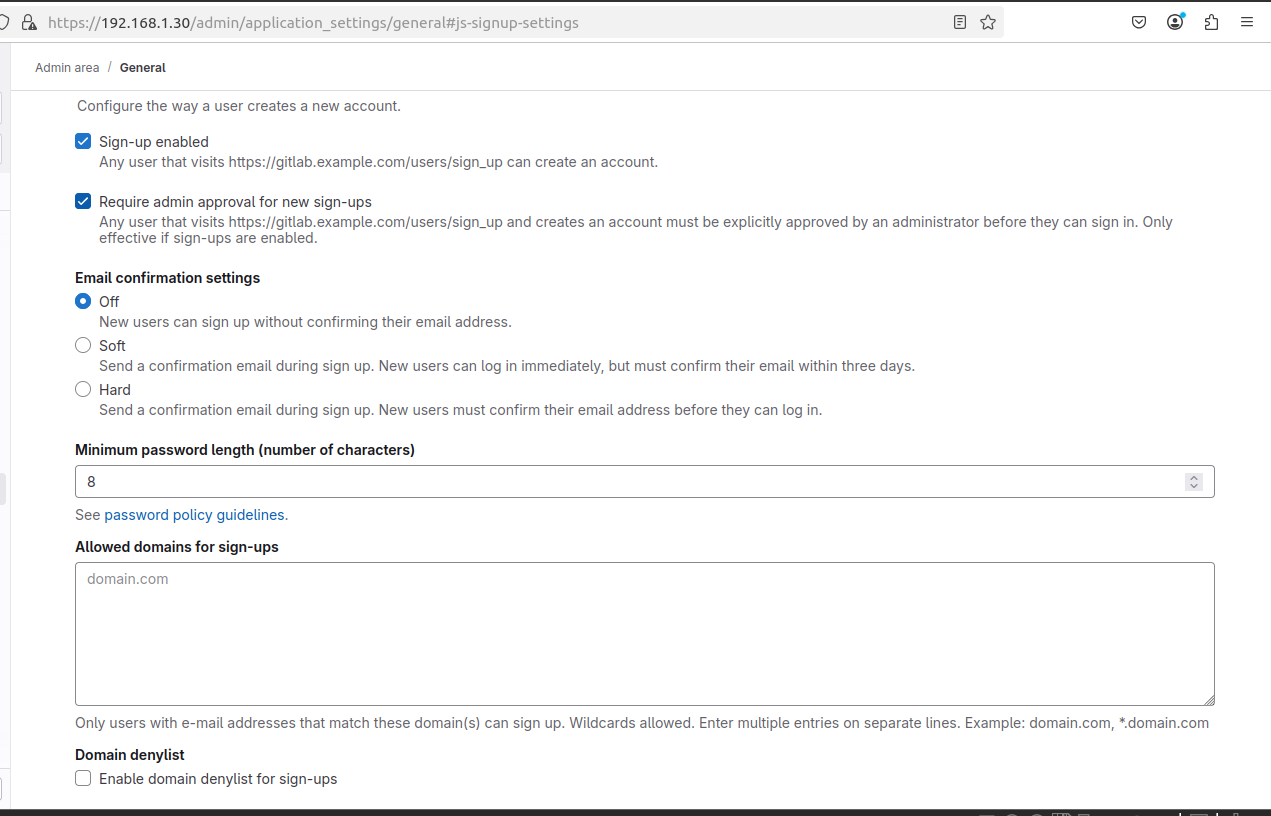
**Khai báo đăng ký chế độ**

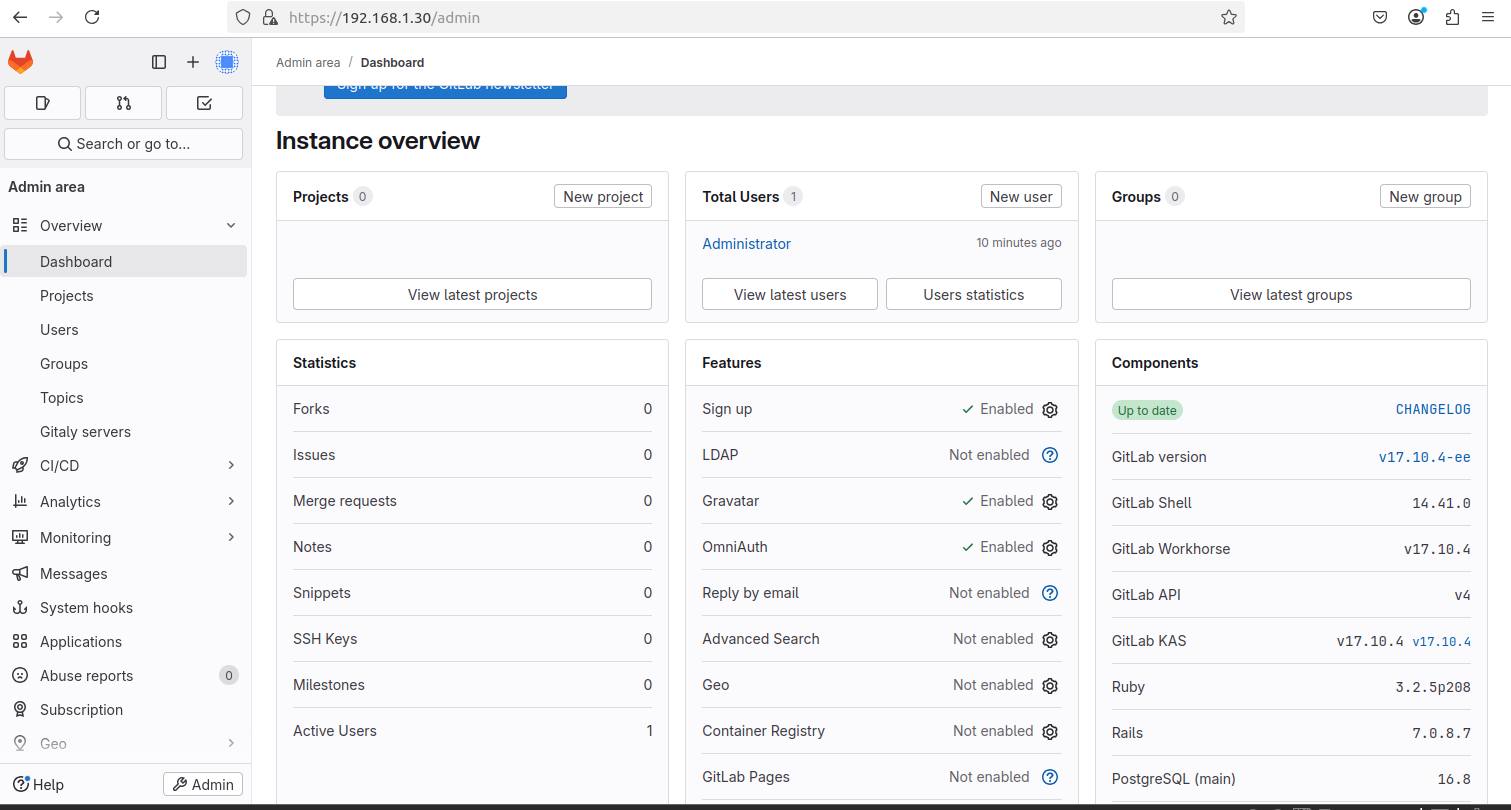
Theo mặc định, bất kỳ ai cũng có thể tạo tài khoản và có quyền truy cập. Nếu bạn không muốn, bạn có thể tắt nó đi. May mắn thay, cài đặt cho nó được hiển thị dưới dạng màn hình bật lên trên bảng điều khiển. Nhấn nút  **Tắt**   để hạn chế đăng ký công khai trên phiên bản Gitlab của bạn. Làm như vậy sẽ chuyển hướng bạn đến trang cài đặt sau.

Bỏ chọn tùy chọn   **Đăng ký được bật**   để hạn chế chế độ của chúng. Nhấn nút   **Lưu thay đổi**   để áp dụng cài đặt.

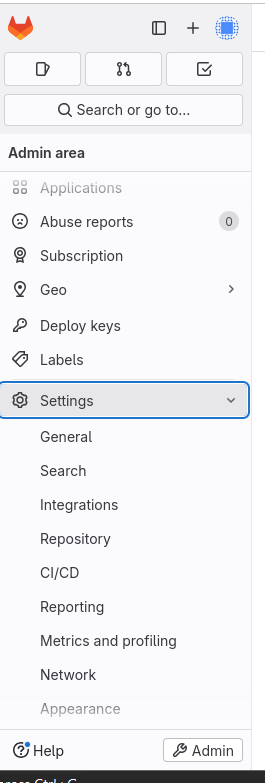
Trong trường hợp bạn không tìm thấy cửa sổ bật lên trên bảng điều khiển, bạn có thể truy cập trang cài đặt bằng cách nhấp vào nút  **Menu**   và truy cập bảng quản trị từ đó.



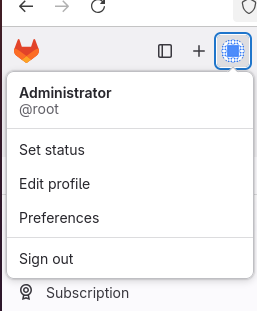




Khi đã vào bảng quản trị, hãy nhấp chuột qua tùy chọn  **Cài đặt**   ở thanh bên trái và nhấp vào menu  **phụ**   . Từ đó, bạn có thể truy cập vào bảng  **đăng ký**   .

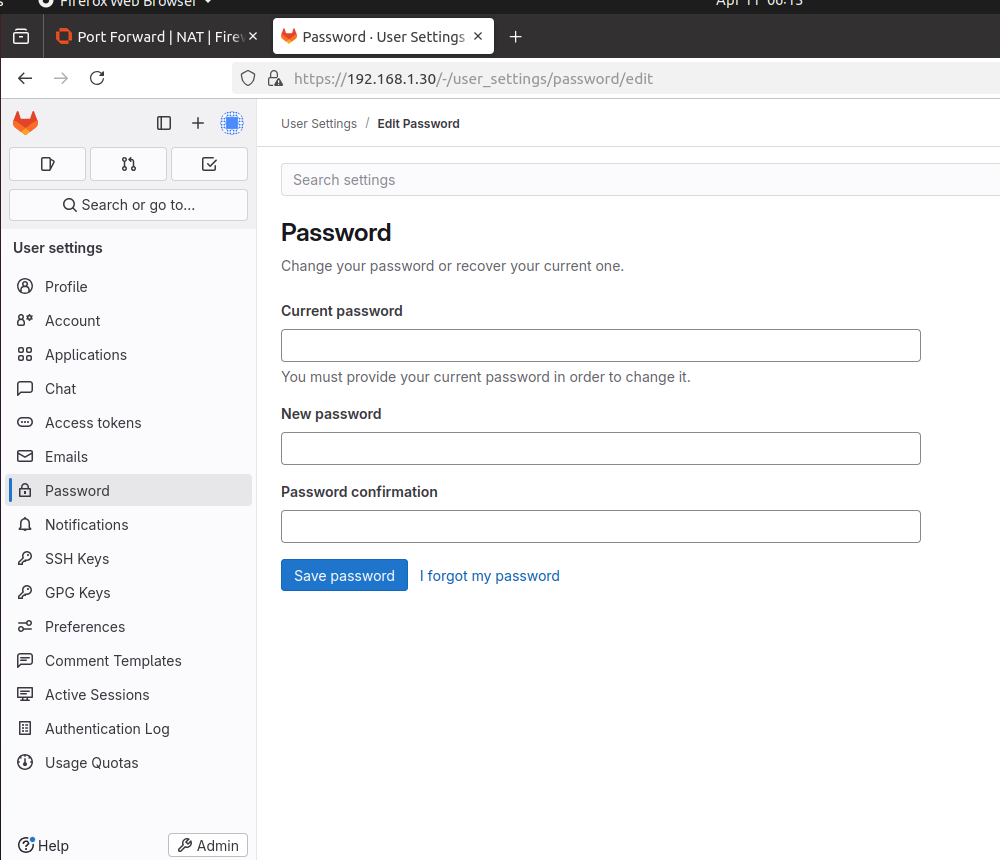


Bạn sẽ được đưa đến trang Cài đặt hồ sơ, tại đó bạn có thể thêm tên, email và các thông tin khác về bản thân. Bước vào   **Cập nhật sơ đồ cài đặt**   khi bạn hoàn tất. Đừng quay lại trang chủ vì chúng tôi còn một số thứ khác cần cấu hình ở đây.



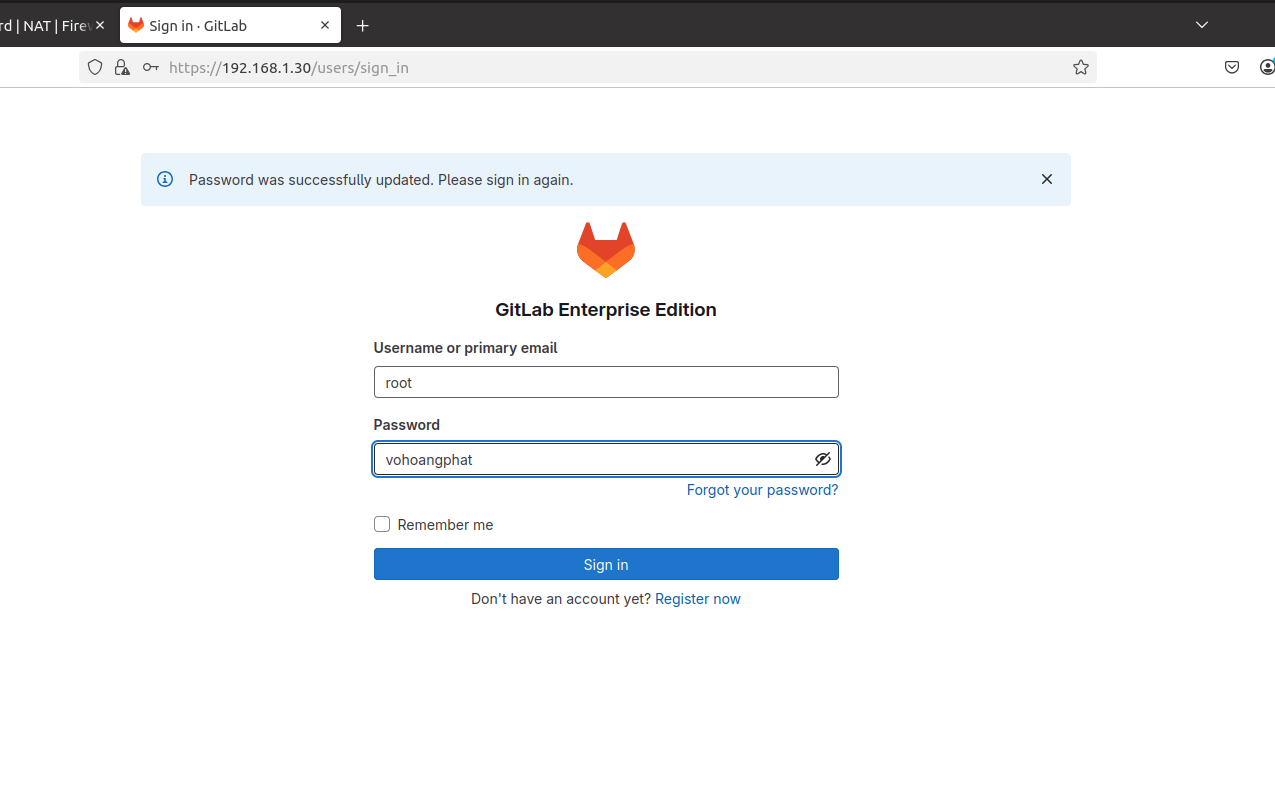
**Đổi mật khẩu gốc**

Đây là một trong những bước quan trọng nhất. Bạn nên thay đổi mặc định mật khẩu gốc ngay lập tức. Với các phiên bản trước, Gitlab yêu cầu bạn phải thay đổi mật khẩu như một phần của quá trình cài đặt nhưng giờ đây nó đã trở thành tùy chọn. Để thay đổi mật khẩu, hãy nhấp vào menu  **Mật khẩu**   từ thanh bên trái.



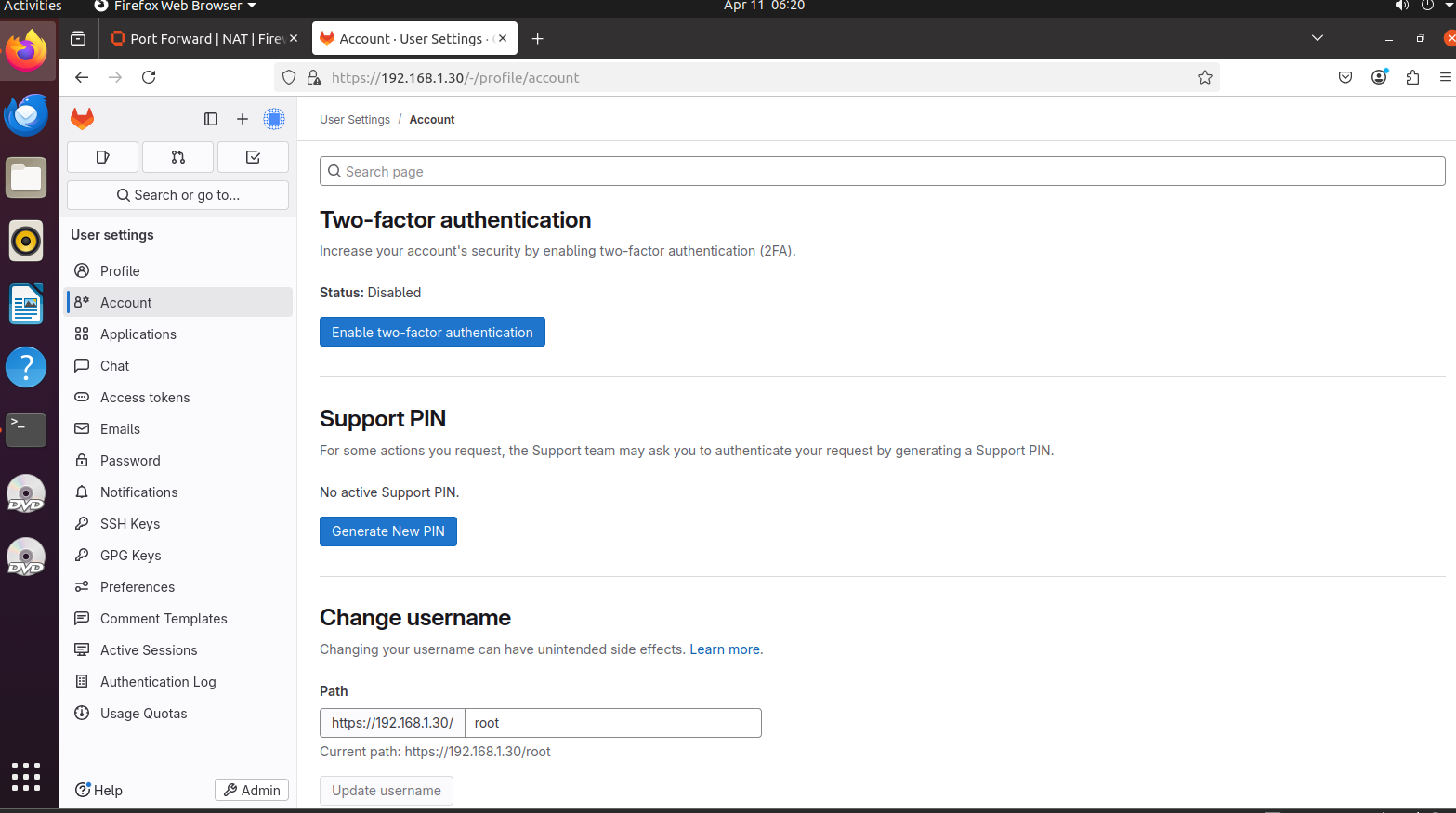
Nhập mật khẩu thông tin của bạn và nhấp vào   **Lưu mật khẩu**   để thực hiện thay đổi. Bạn sẽ được đăng nhập từ phiên bản của mình và phải đăng nhập lại.

Sau đó văng ra và nhập lại pass vừa đổi

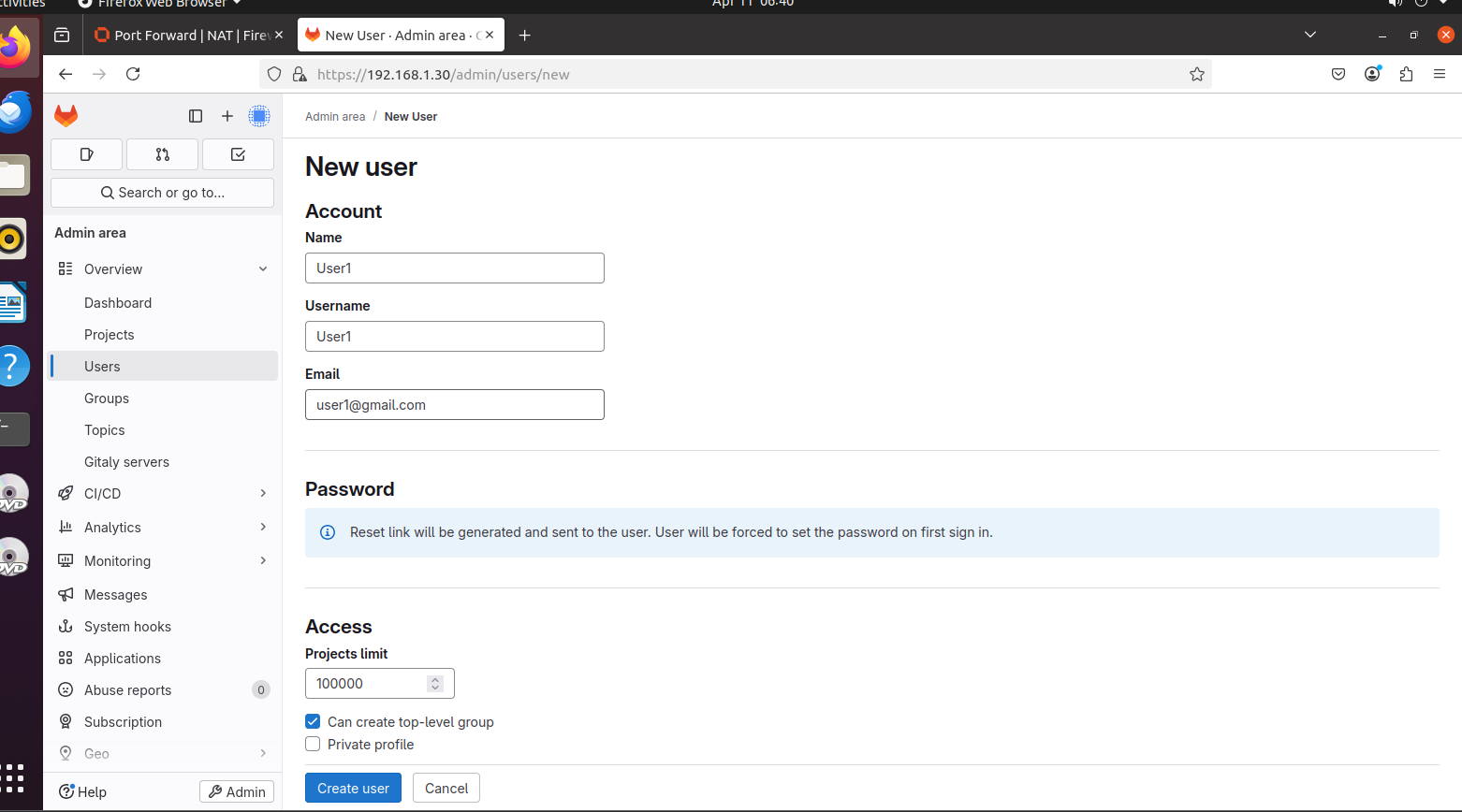


Khi vào trang, nhập tên người dùng mới của bạn và nhấp vào nút  **Cập nhật tên người dùng**   để thực hiện thay đổi. Bạn sẽ được xác nhận lần nữa. Nhấn nút  **Cập nhật tên người dùng**   một lần nữa để xác nhận các thay đổi.

Bạn cũng nên bật xác thực hai yếu tố tại đây để cải thiện tính bảo mật tài khoản của mình.



Tạo tài khoản user



✅ User này sẽ chỉ có quyền user bình thường, phải được mời vào Project mới làm việc được.

**🔥 Giải thích từng phần khi Tạo New User:**

**1. Account (Tài khoản cơ bản)**

* **Name**: Tên hiển thị (ví dụ: *Đẹp Trai*).
* **Username**: Tên đăng nhập (ví dụ: *deptrai*).
* **Email**: Email để kích hoạt, reset password.
* **Password**:
  + Nếu **nhập** mật khẩu: User dùng mật khẩu đó để login.
  + Nếu **bỏ trống**: GitLab sẽ tự tạo một **link reset password** gửi vào Email user → yêu cầu họ tự đặt mật khẩu lần đầu tiên.

**2. Access (Quyền truy cập)**

* **Projects limit**:  
  Giới hạn số lượng project người dùng này được tự tạo. (Mặc định: 10000).  
  → Bạn có thể giới hạn (ví dụ: chỉ cho tạo tối đa 5 project).
* **Can create top-level group**:
  + **Bật**: User này có quyền tạo Group cấp cao nhất (có thể quản lý nhiều Project trong Group).
  + **Tắt**: Chỉ có thể join Group có sẵn, không được tự tạo Group lớn.
* **Private profile**:
  + Bật cái này: User đó chỉ hiển thị thông tin với những người họ chia sẻ dự án chung → tránh lộ thông tin user.
* **Access level** (Rất quan trọng): 3 mức:

| **Access Level** | **Giải thích** | **Khi nào dùng** |
| --- | --- | --- |
| Regular | Người dùng bình thường. | Dùng cho nhân viên, sinh viên... làm việc bình thường. |
| Administrator | Admin toàn quyền | Bạn dùng cho root hoặc ai bạn muốn trao quyền admin cao nhất. |
| External | User bên ngoài | Dùng cho cộng tác viên, freelancer. Hạn chế quyền, chỉ vào project được mời. |

* **Validate user account**:
  + Nếu GitLab bạn bật **CI/CD Runner** Free → yêu cầu user phải xác thực thẻ tín dụng mới chạy được.
  + Nếu không quan tâm CI/CD thì bỏ qua.
  + (Bạn cũng có thể **Admin validate** cho user thủ công nếu cần.)

Nghiên cứu sau

**3. Profile (Thông tin cá nhân thêm)**

* **Avatar**: Ảnh đại diện.
* **Skype, Linkedin, X (Twitter), Website URL**:  
  Thông tin mạng xã hội, không bắt buộc.
* **Admin notes**:  
  Ghi chú nội bộ cho Admin (không ai khác thấy), ví dụ: "Đẹp Trai - Dev Backend".

# Nginx

1. Chuẩn bị thư mục và SSL

mkdir -p ~/gitlab/ssl

cd ~/gitlab/ssl

# Tạo chứng chỉ SSL tự ký cho GitLab

openssl req -new -newkey rsa:2048 -days 365 -nodes -x509 \

-subj "/CN=gitlab.example.com" \

-keyout gitlab.example.com.key \

-out gitlab.example.com.crt

3. Tạo Docker Network

docker network create backend

4. Khởi động GitLab Container

cd ~/gitlab-docker

docker compose up -d

5. Trỏ domain vào localhost (nếu chạy test máy local)

echo "127.0.0.1 gitlab.example.com" >> /etc/hosts

**💬 Ghi chú thêm:**

* Biến GITLAB\_HOME phải export hoặc set thủ công trong .env. Ví dụ:

export GITLAB\_HOME=/srv/gitlab

hoặc trong file .env:

GITLAB\_HOME=/srv/gitlab

Nếu dùng chứng chỉ Let's Encrypt thì phần SSL trong docker-compose.yml sẽ thay bằng cert thực.