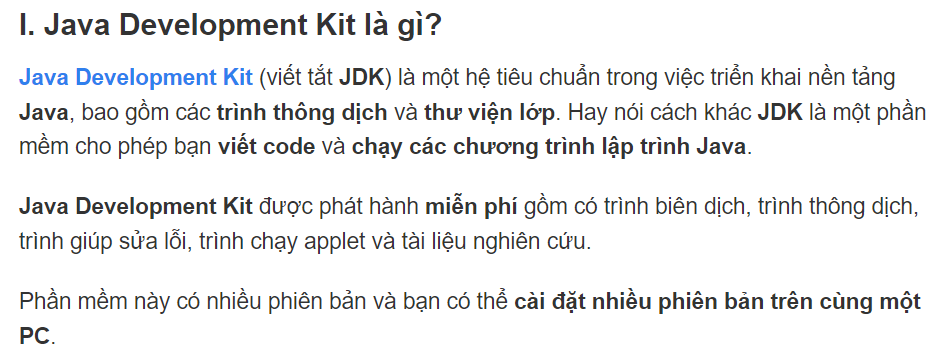
# Install Jenkins

Install trên window

Trước khi cài jenkins phải cài **JDK - Tải Java Development Kit** (java oracle)

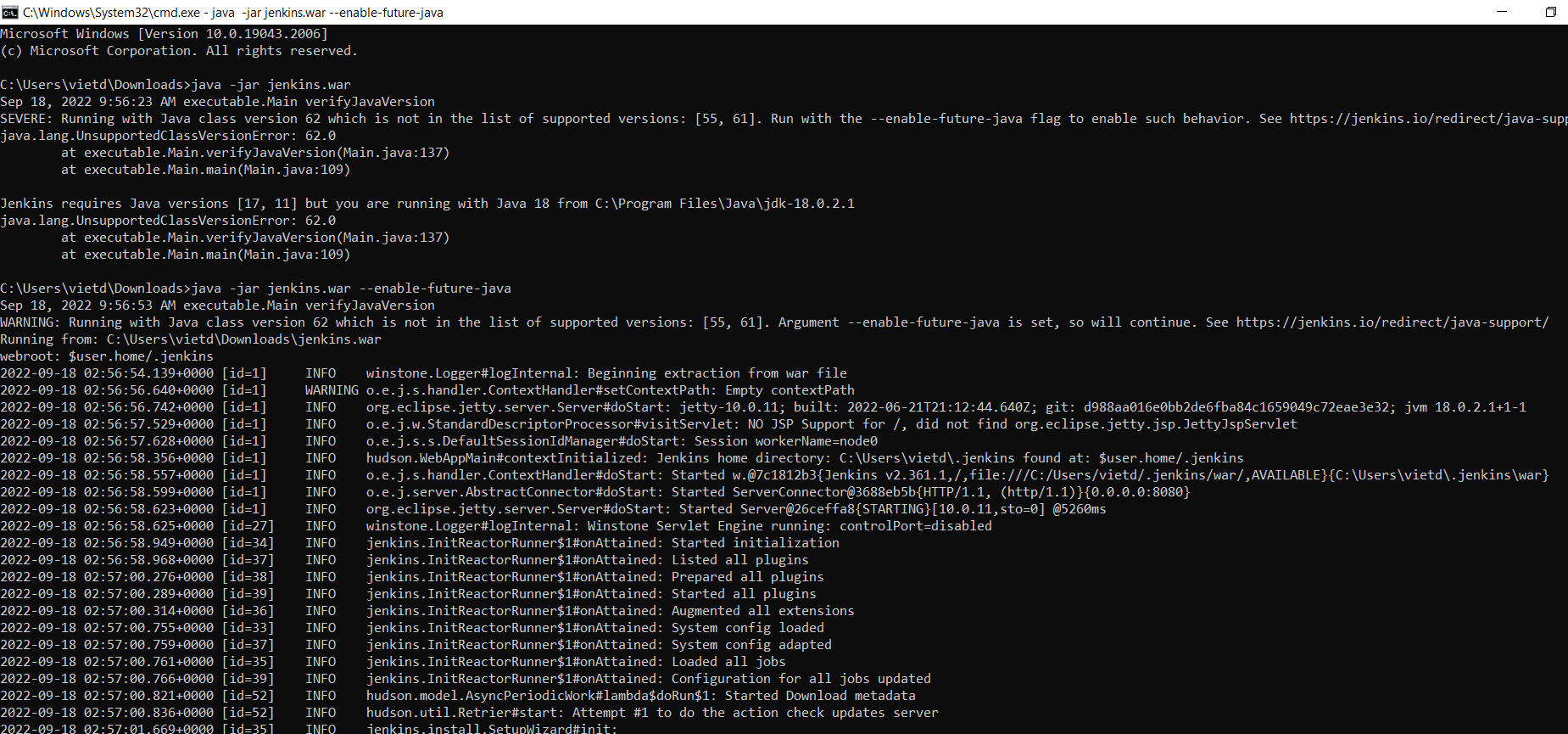
Java Development Kit (JDK) là một trong những phần mềm công nghệ cốt lõi được sử dụng trong lập trình, cho phép bạn **viết code** và **chạy các chương trình Java**



**Cả trên linux và window khi cài jenkins cũng cần phải có JDK**

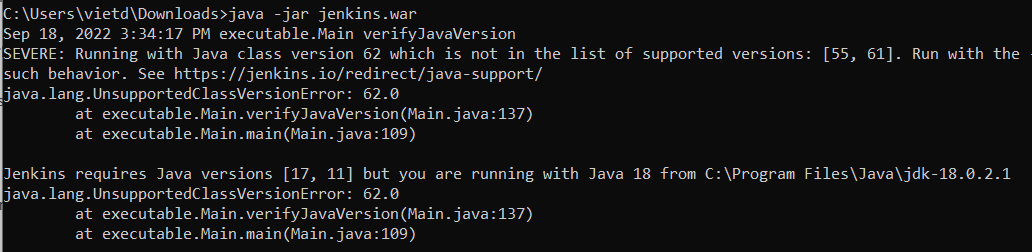
Install java xog, chạy jenkins bằng lệnh

java -jar jenkins.war --enable-future-java



<https://www.jenkins.io/doc/book/installing/war-file/> - install jenkins cumtoms port

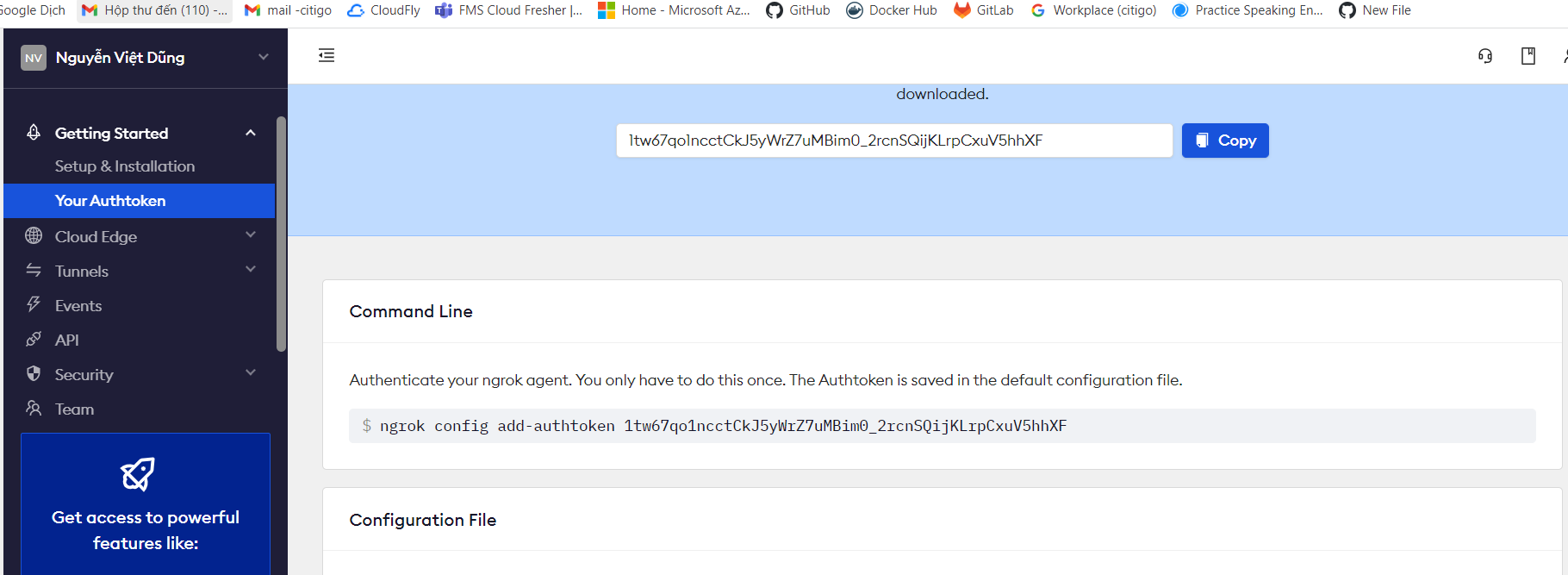
install jenkins



Lưu ý jenkins đang chạy lên cài java 17

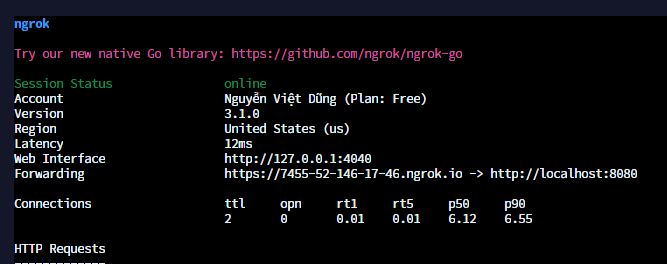
## Ngrok

<https://dashboard.ngrok.com/get-started/your-authtoken>

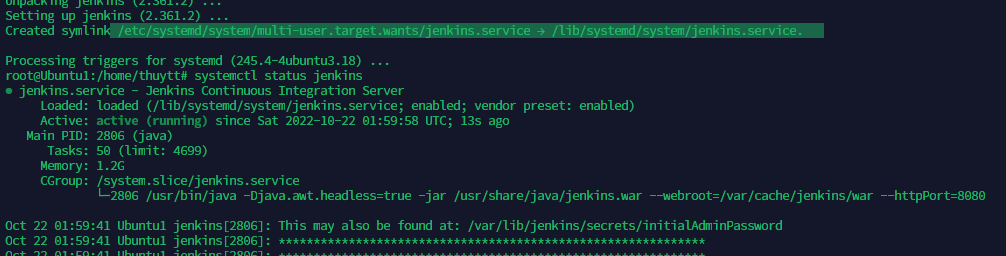


Chạy ngrok config

Ngrok http 8080 ->



### service



Install bằng docker

docker run -d --name jenkins -p 8080:8080 -p 50000:50000 -v /home/vietdung/:/var/jenkins jenkins/jenkins:lts



Docker ps

Docker logs [jenkins]

Để lấy pass



Cài ngrok để tạo public URL

* Tạo repo Github
* Tạo project trên Jenkins
* Thêm link Jenkins vào Github webhook
* Đẩy code lên Github

Install trên ubuntu

<https://www.jenkins.io/doc/book/installing/linux/>

## Install trực tiếp

2. Java installation

yum install -y java-1.8.0-openjdk-devel.x86\_64

alternatives --config java

java -version

3. Jenkins install

wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat/jenkins.repo

rpm --import <https://pkg.jenkins.io/redhat/jenkins.io.key>

yum install -y jenkins

4. Start Jenkins

systemctl start jenkins

5. Setting password (lấy pass authen)

$ less /var/lib/jenkins/secrets/initialAdminPassword

Less tương tự vi, nhưng less chỉ xem đk, ko sửa dk

## Deploy test jenkins build server

1. Web server installation

#!/bin/bash

yum update -y

yum install httpd -y

yum install git -y

amazon-linux-extras install epel

yum install epel-release

rpm -Uvh http://rpms.famillecollet.com/enterprise/remi-release-7.rpm

yum install -y php70 php70-php php70-php-fpm php70-php-pecl-memcached php70-php-mysqlnd php70-php-xml

ln -s /usr/bin/php70 /usr/bin/php

service httpd start

chkconfig httpd on

Tạo 1 user cho việc push code

2. Configure Apache

useradd www-user

cd /var/www/html/

mkdir tinhocthatladongian

chown -Rf www-user:apache tinhocthatladongian

vi /etc/httpd/conf/httpd.conf

Listen 81

NameVirtualHost \*:81

<VirtualHost \*:81>

DocumentRoot /var/www/html/tinhocthatladongian/

<Directory "/var/www/html/tinhocthatladongian">

Order deny,allow

Allow from all

AllowOverride All

Require all granted

</Directory>

</VirtualHost>

Ssh: user abc

Web-server

Jenkins

User/pass

Thằng jenkins sẽ ssh đến web-server để pull code

Vì vậy nó sẽ cần user để ssh đến web-server

* User sẽ cần quyền đọc thực thi trên folder của web-server

Vậy tạo 1 user có quyền cho folder web server pull code

3. Creating ssh key

$ sudo su - www-user

$ ssh-keygen -t rsa

$ cd /home/www-user/.ssh/

$ mv id\_rsa.pub authorized\_keys

4. Copy private key to Jenkin server

cd /home

mkdir jenkins

vi web-key.pem

chown -Rf jenkins:root

chmod 400 web-key.pem

Key này là key của user www-user

Mà thằng jenkins access đến web-server qua user www-user

* Jenkins phải có private key của user www-user
* Web server chỉ cần public key copy vào file authen ở user ww-user cho việc xác thực

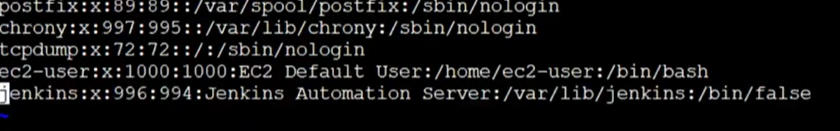
Thế cái file private key copy vào đâu ở jenkins

* Copy vào đâu cũng dk , nếu jenkins hđ user root thì copy vào /root/.ssh
* Nếu là user jenkins thì copy vào /home/jenkins/.ssh

Chmod 600 cho authen.key

Su sang user jenkins, nếu ko login được

Vi /etc/pass



Đổi cái false =bash

Test thử thằng jenkins ssh được ko

Ssh user jenkins

ssh www-user@[Web Server Address] -i /home/jenkins/web-key.pem

/home/jenkins/web-key.pem : là file chứa key

Nếu ko muốn thì đẩy vào /home/jenkins/.ssh

Tại sao phải su sang user jenkins

Vì để config vs webhook cho thằng jenkins tự chạy script thì n sẽ dùng user jenkins để chạy

## Webhook

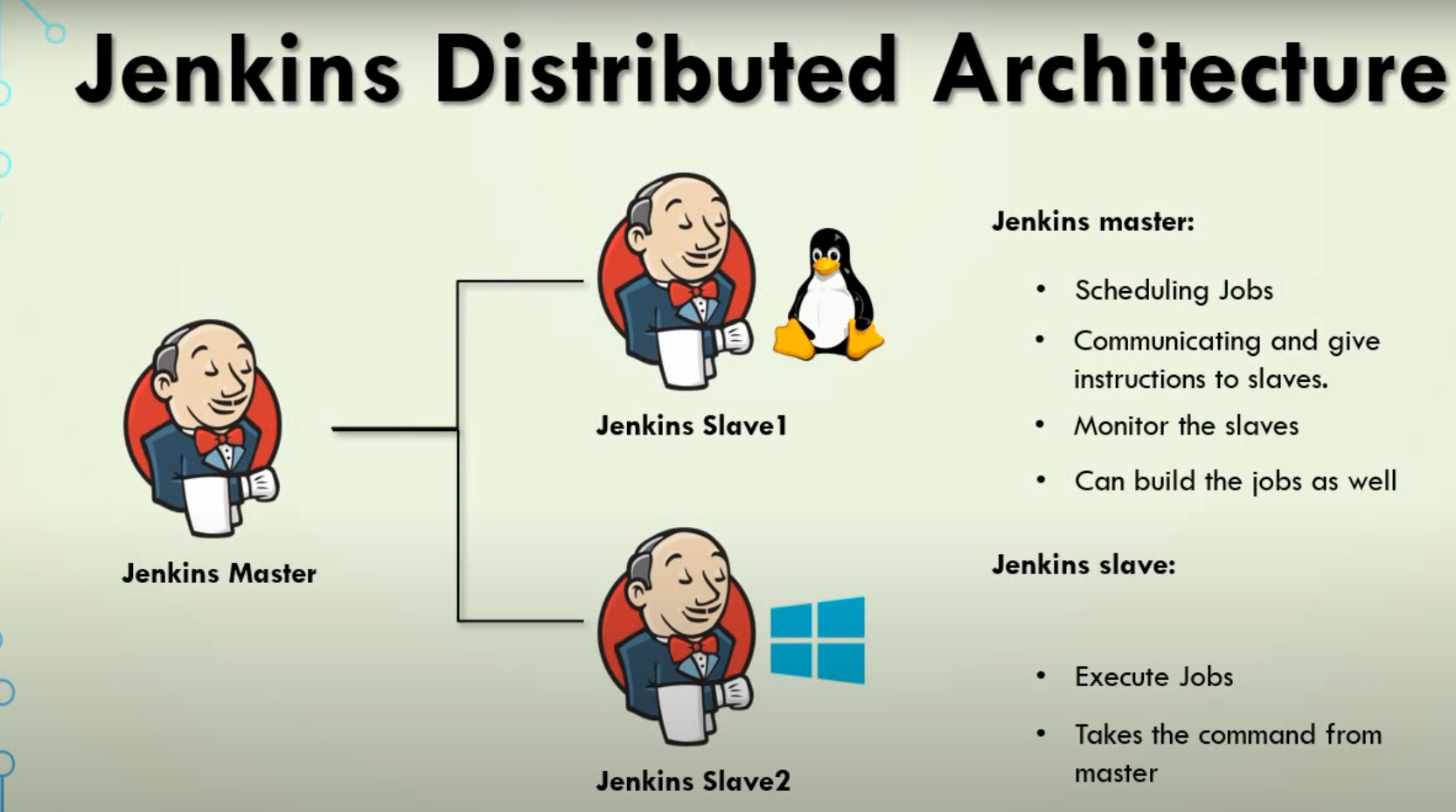
Ý nghĩ webhook: khi code đẩy lên git, thì thằng webhook sẽ thông báo cho jenkins là có sự thay đổi ở repo, m chạy đi

Lên github -> setting -> webhook -> add webhook -> copy URL jenkins vào

## Cấu hình Jenkins cho repo nào

Đảm bảo trên jenkins có git

# Jenkins architecture (the4)



## Jenkins slave là các Jenkins agent

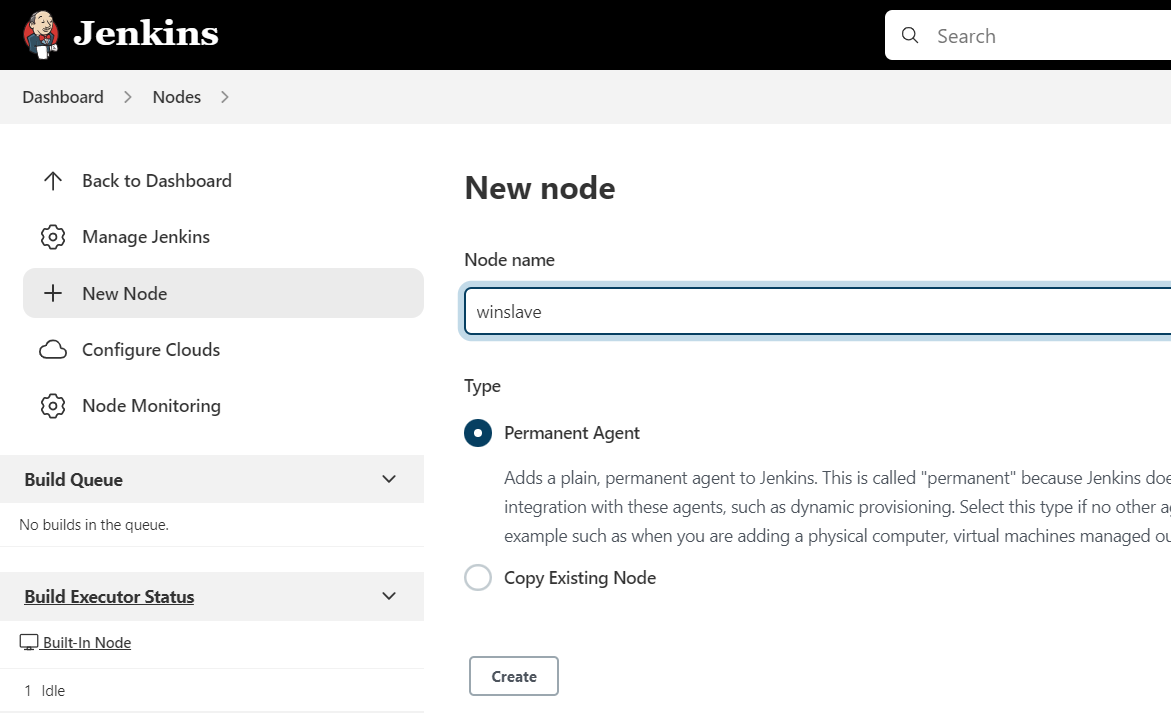
Vì 1 máy là ko thể chạy hết các job cùng 1 lúc và có 1 số job cần deploy trên window,

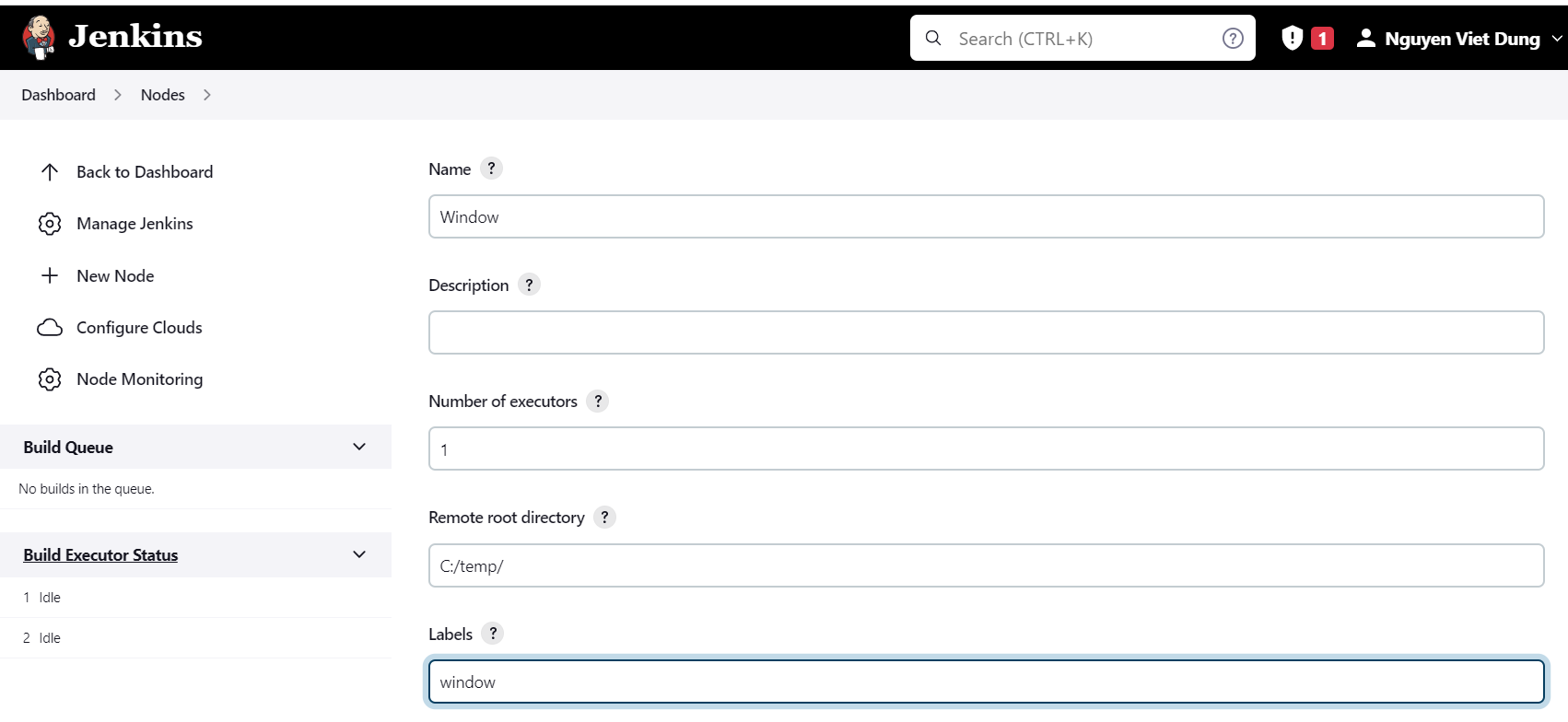
Ví dụ: khi tạo 1 job jenkins, bạn có 1 master jenkins, tạo job trên jenkins master, nhưng job phải dk run trên jenkins slave.

Làm thế nào: Set label trong configuration của master jenkins và slave jenkins

## Thêm slave

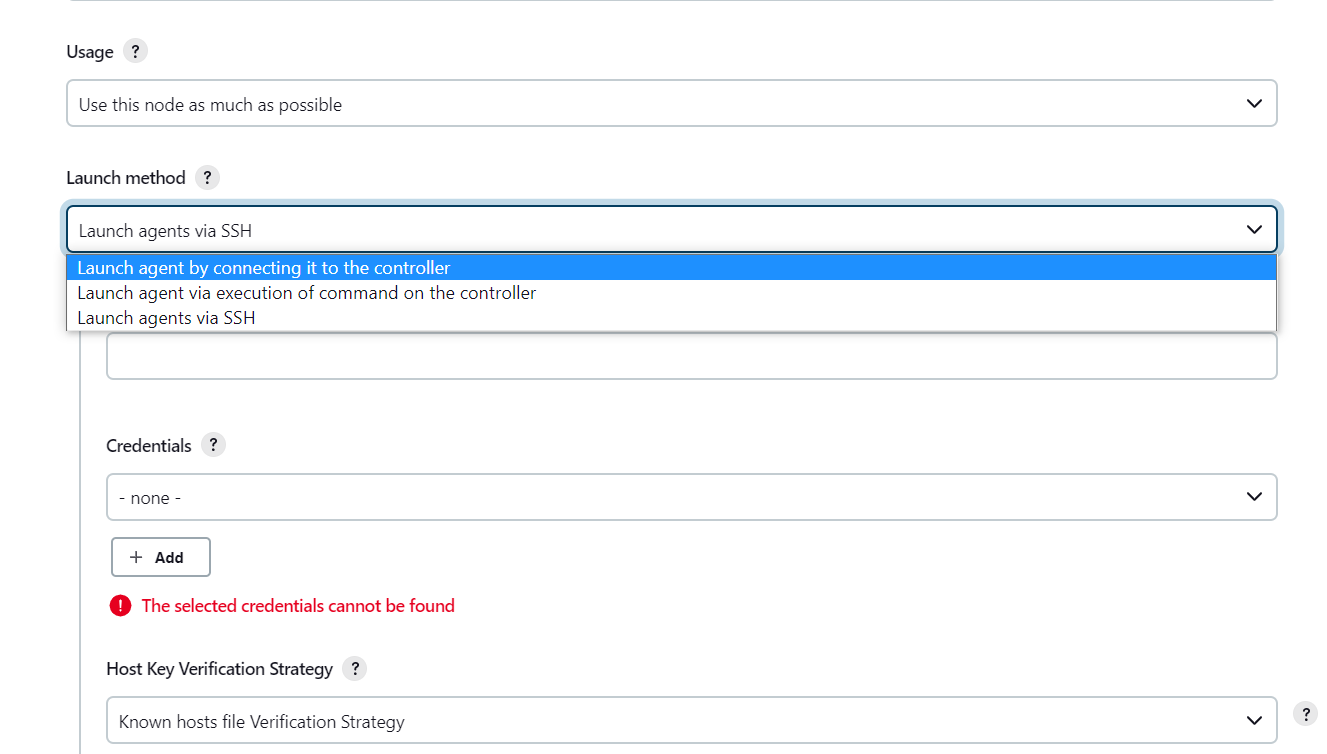
Vào manager -> Manage nodes and clouds -> new node





C:/temp : là thư mục root của jenkins agent, nơi jobs lưu file trong window slave

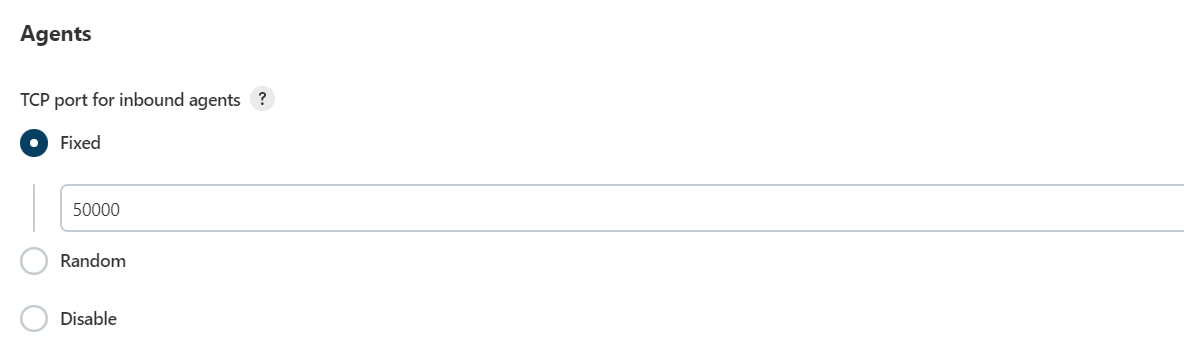
Label: xác định cái jenkins slave có label phù hợp để run job

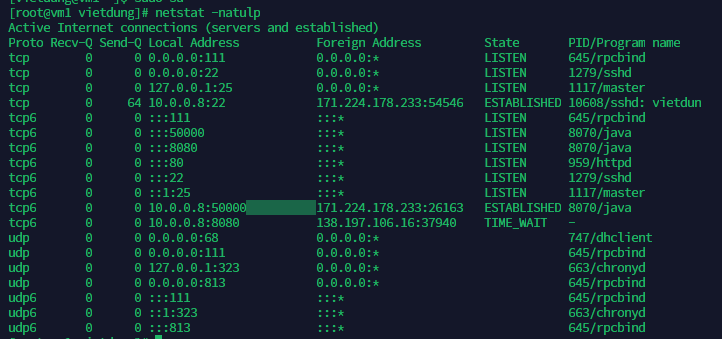


Usage: sử dụng deploy job cho chỉ …

B2: vào C**onfigure Global Security**

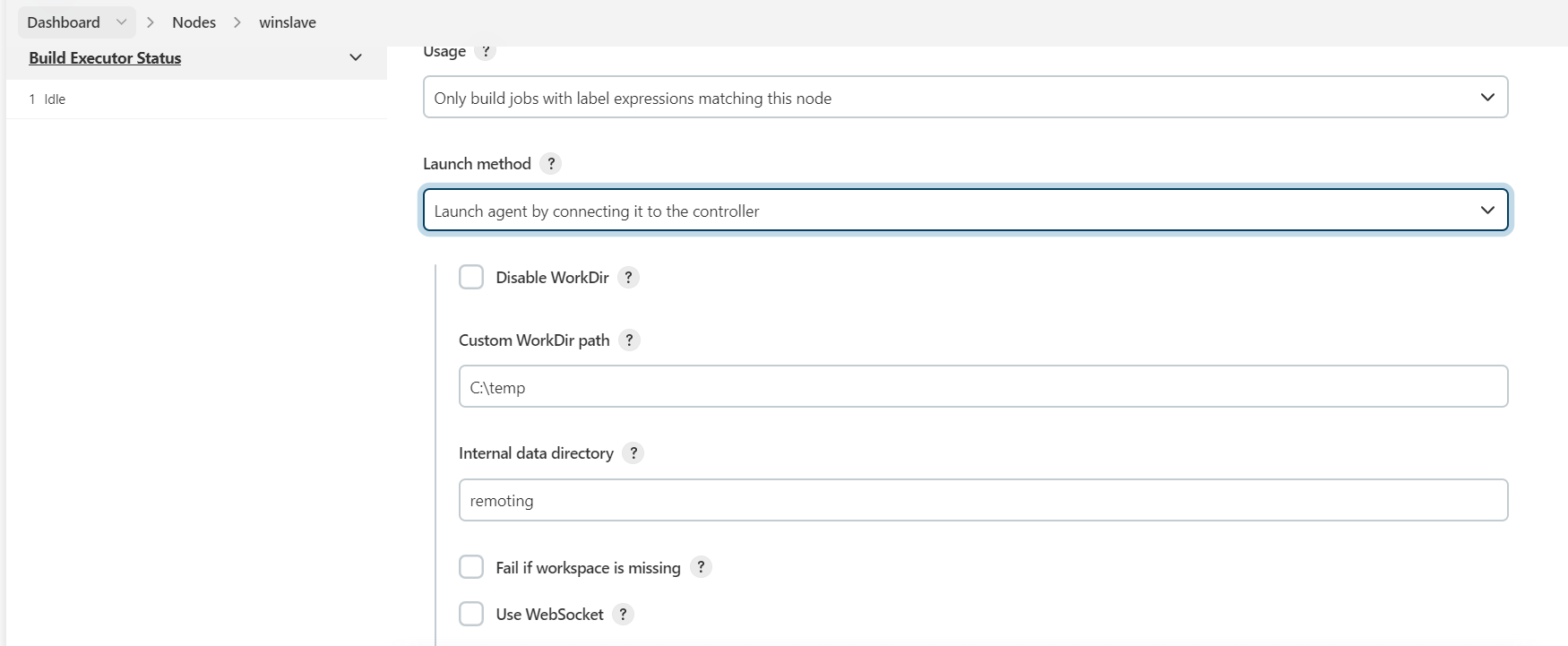
* Port 50000 hoặc random



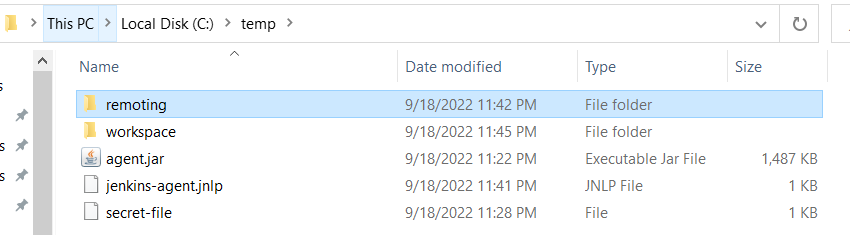


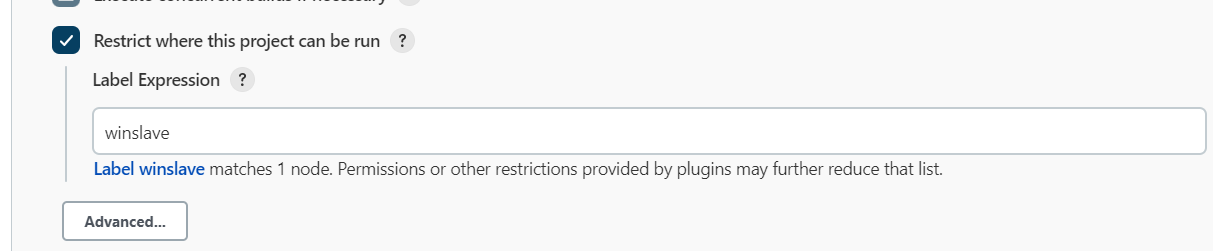
Port 50000 là của con chính (jenkins master) để thiết lập với con agent window

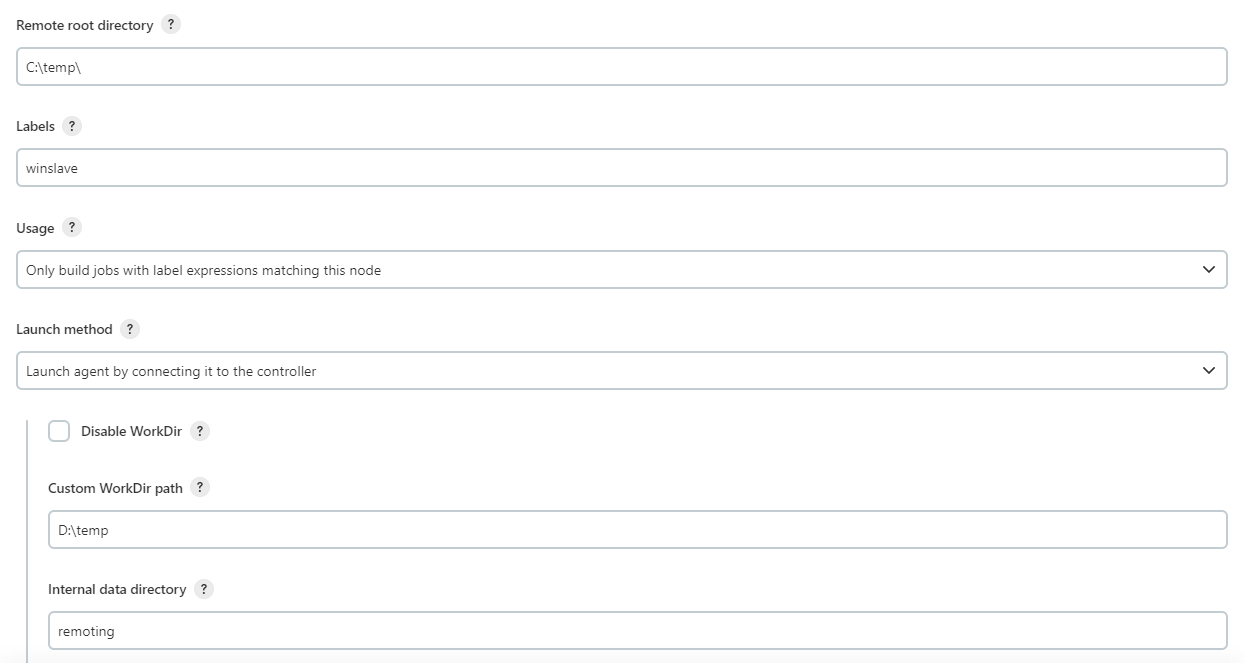
B3: Vào tiếp config của node window jenkins agent

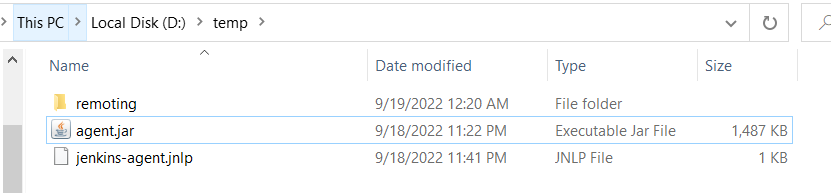


Thư mục để tạo ra cái workplace để chạy, build







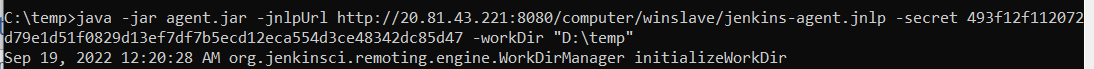
****

Cái launch method kia là cái thư mục để remoting với cái jenkins master, tức là ta sẽ cần 2 file **.jar** và **.jnlp** và sẽ chạy luôn cái để connect



Phải chạy đúng nơi khai báo cái remoting directory

Nếu ko sẽ bị lỗi như dưới đây cho dù ta có chạy cái lệnh

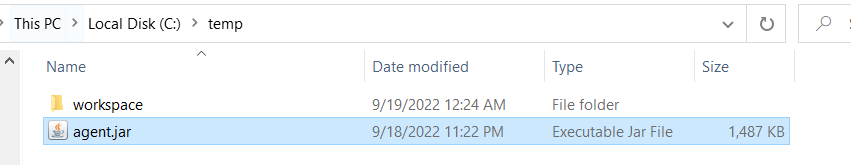


Và n cũng đã báo kết nối, nhưng do ta set thư mục remoting sai

* Cài 2 file agent.jar và .jnlp rồi vứt vào folder remoting của window và chạy connect

## Lỗi build



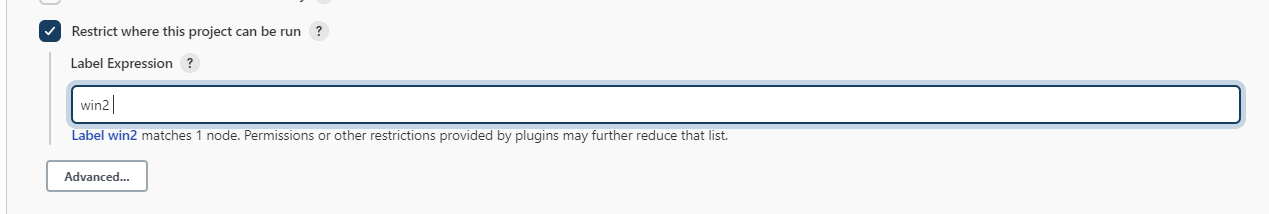


Thư mục root sẽ là nơi sinh ra **workspace**

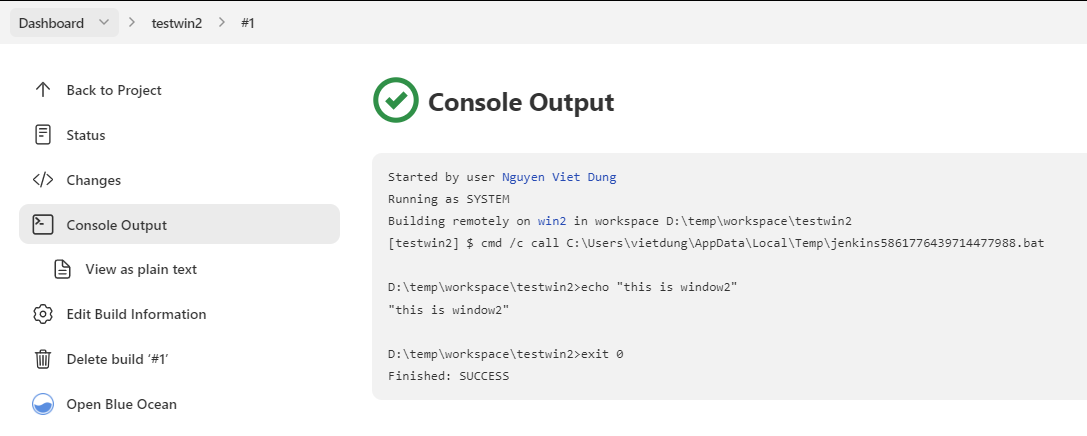
**Mà tóm lại cứ để 2 thư mục kia giống nhau**

**B3: tạo 1 job trên master để build, deploy trên agent**

New item “testwin2” -> freestyle Project

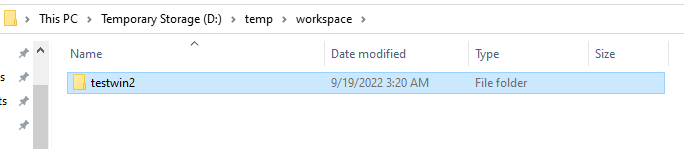


**Execute Windows batch command** - echo "this is window2"

****

Build now: done luôn

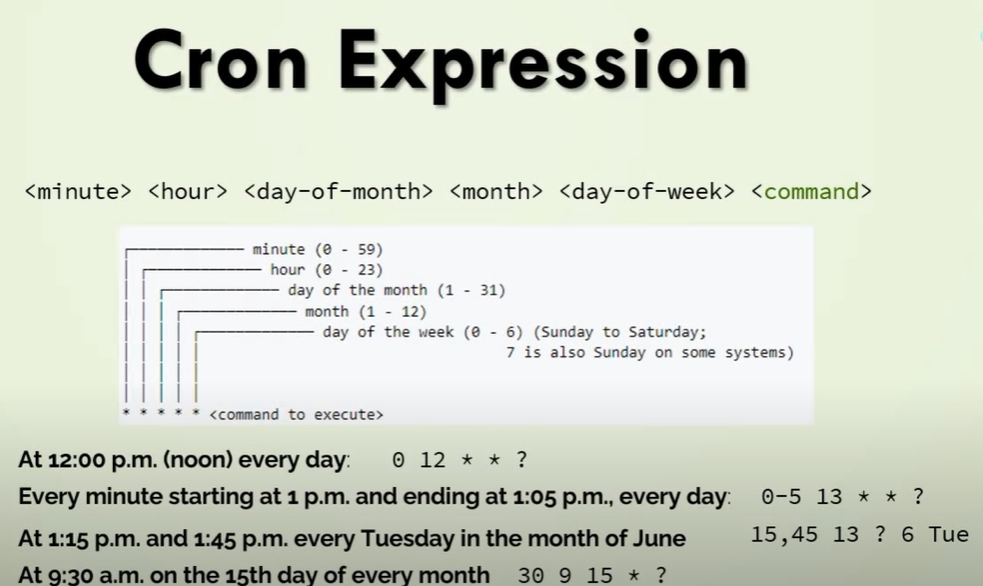
Building remotely on [win2](http://20.81.43.221:8080/computer/win2) in workspace D:\temp\workspace\testwin2

****

Trên window đã có

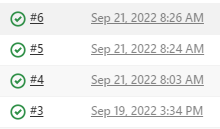


# Scheduling job in Jenkins (the4)





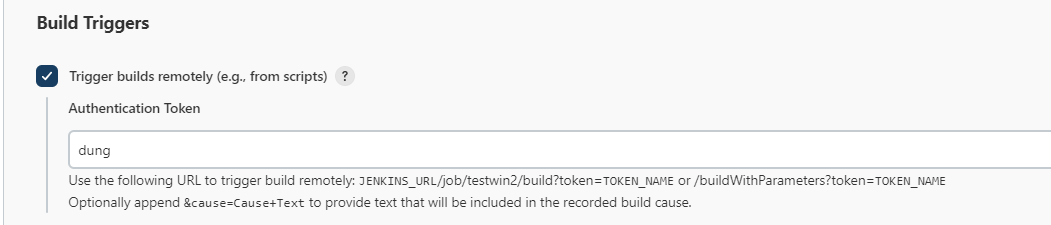
2 phút 1 lần chạy

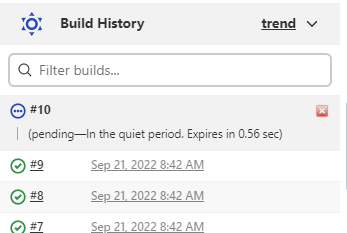


# Remote build trigger (the4)

Khi 1 máy mà có kết nối (remote) tới con jenkins thì chạy đường dẫn kia, job sẽ tự chạy mà ko cần click build

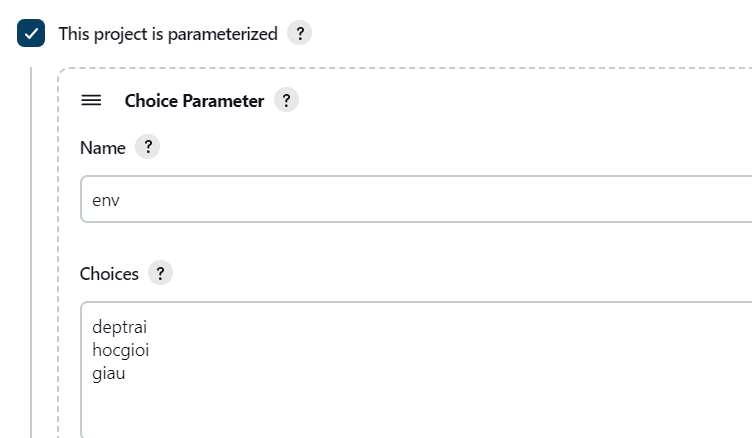
[20.81.43.221:8080/job/testwin2/build?token=dung](http://20.81.43.221:8080/job/testwin2/build?token=dung)

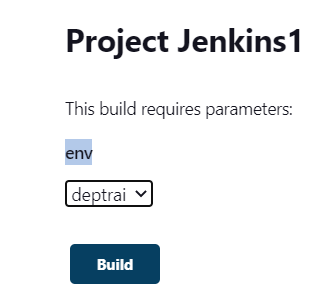




## This project is parameterized[?](http://20.6.73.22:8080/job/Jenkins1/configure)

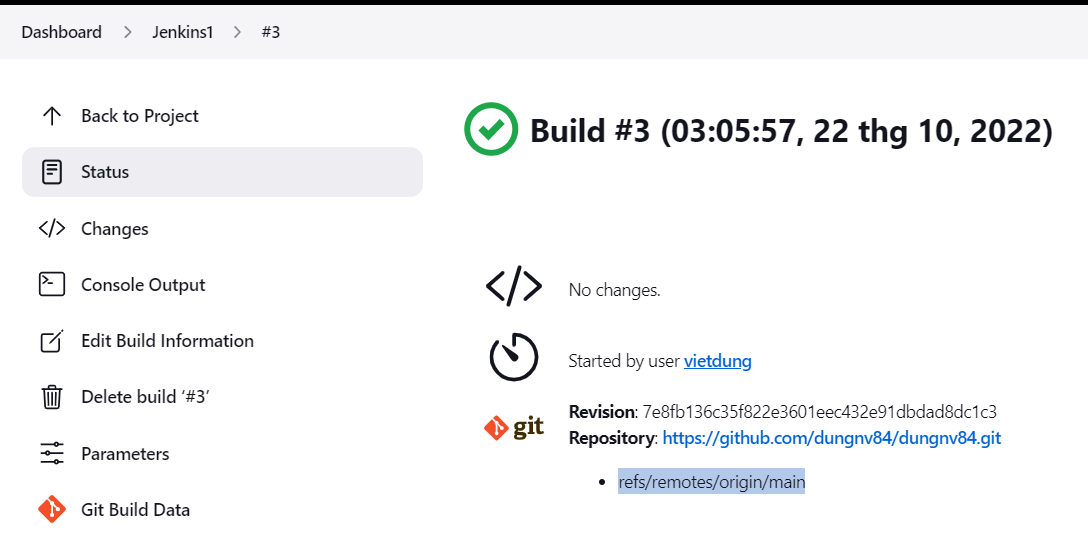
Truyền env khi build job





## Git pull

Nếu ta config là pull nhánh main cho n



Nó sẽ refer đến nhánh main để pull



Cái filenewmain là commit mới tôi vừa up lên git và t build lại trên jenkns

N sẽ checkout tới nhánh main (git checkout -f 9f1e396…..)

Và commit message

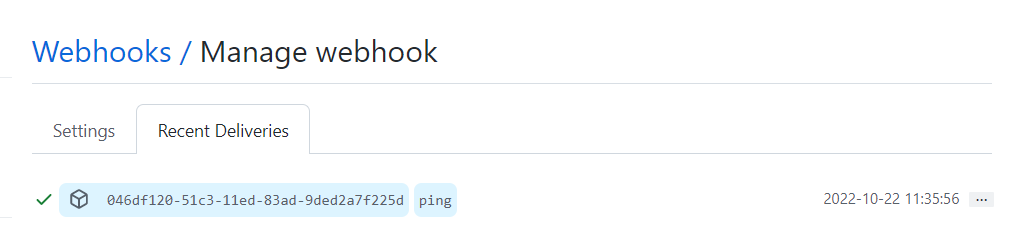
## Webhook

Install plugin: github, webhook

Webhook nhận event của 1 pipeline

* Set webhook sẽ ở trong từng pipeline riêng của gitlab, github

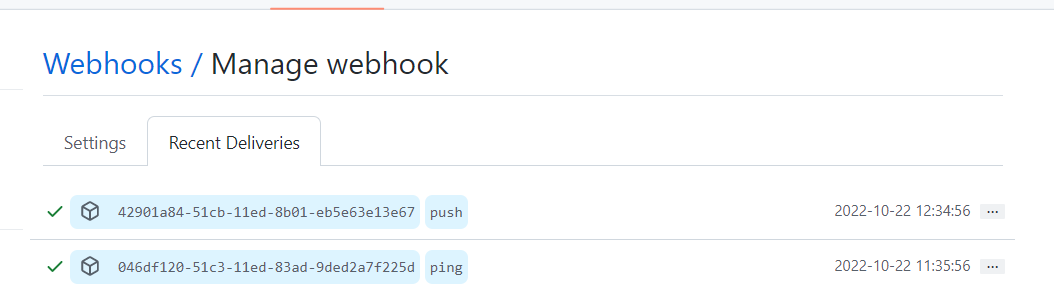
Vào project -> setting -> webhook -> add URL: <http://20.6.73.22:8080/github-webhook/>



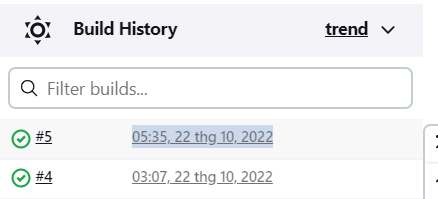
Hiện như này là oke,

Lưu ý là chỉ cho 1 job pipeline của github

Bgio thử commit 1 event cho pipeline này, jenkins sẽ tự build



Đây là commit vừa push, và commit này webhook n đã nhận biết

 và jenkins đã tự build

Có thể check thay đổi ở /var/lib/jenkins/workspace

## Jenkins Maven Intergration

Đối với project của java, ta sẽ cần Maven,…

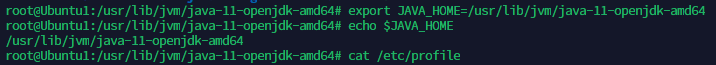
Khi mà dự án java đã làm xog, để triển khai cả dự án lên server thì ta sẽ build cả dự án ra file war giống đóng image cho docker file

Maven có nhiệm vụ build cả dự án ra file war

Or .Net project, ta sẽ cần MSBuild tool cho CI/CD

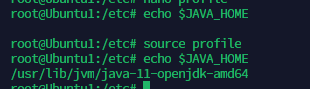
Tạo Enviroment để check file, nhưng sau khi restart thì n sẽ mất

Ta lên lưu cái này trong /etc/profile



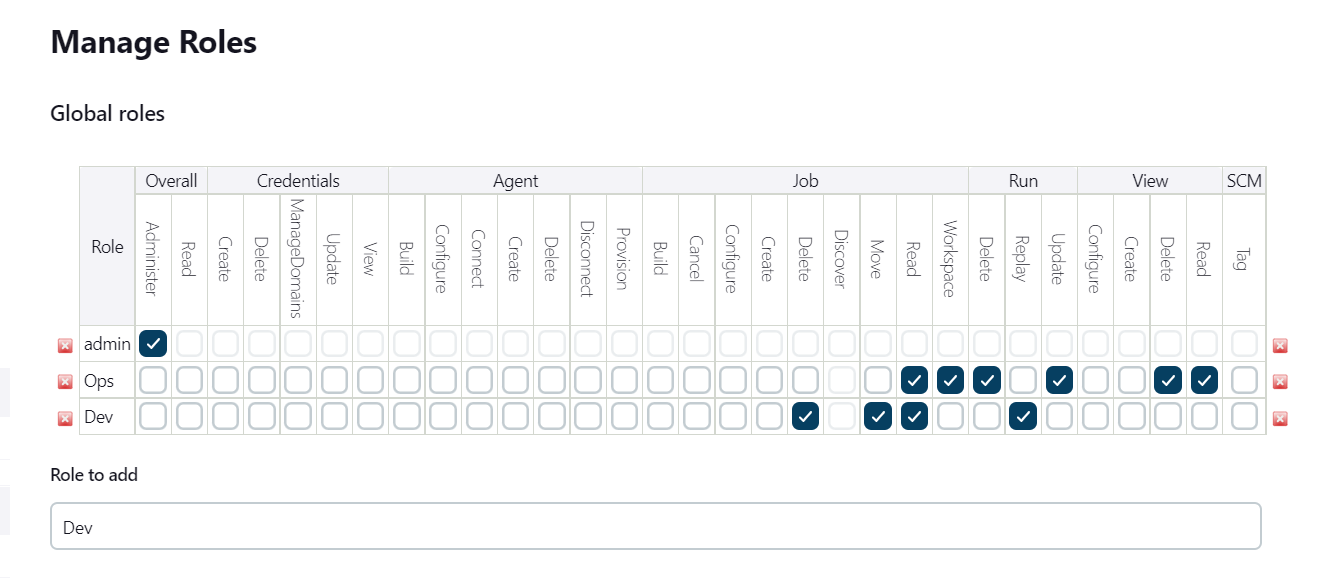
Nano /etc/profile

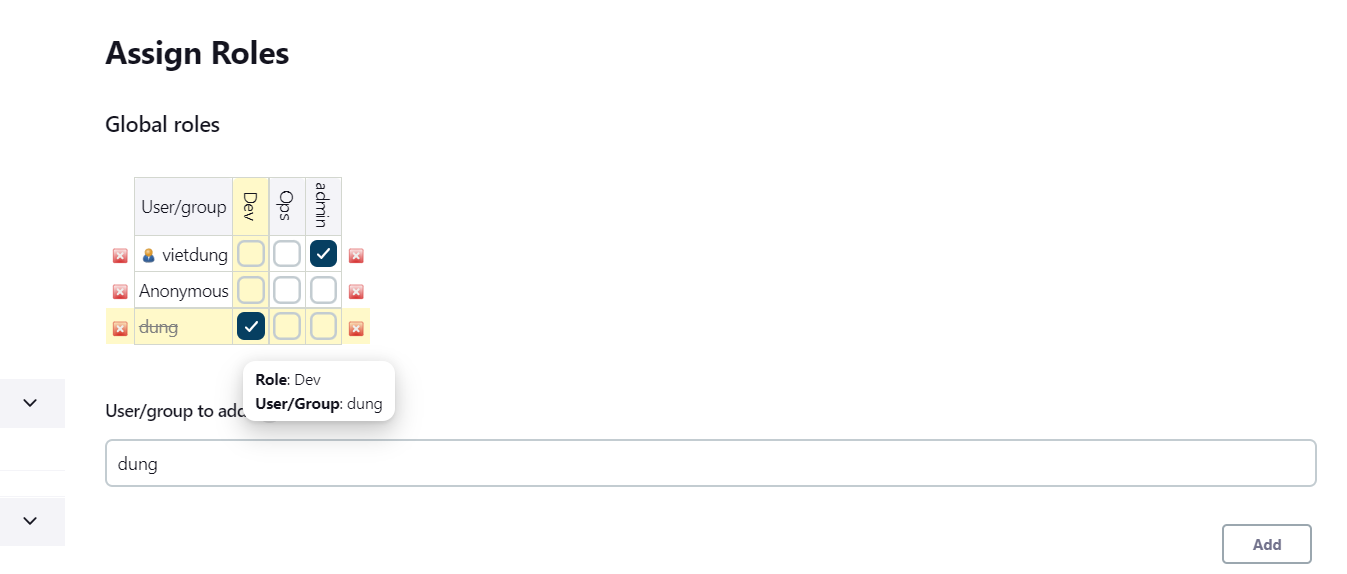




Role base

Install role base cho phân quyền group





Add user cho role Dev hay Ops

Cái này chắc cũng có thể dùng cho user email

Làm sau nhé

## Jenkins pipeline job

Pipeline {

agent anyname

stages{

stage(‘stage name1’){

steps{}

}

stages{

stage(‘stage name2’){

steps{}

}

}

}

# Tích hợp github – Jenkins

Đẩy code lên github

Cấu hình Jenkins tự động nhận biết sự kiện đó và kéo code về



Mỗi lần github có thay đổi, thì webhook sẽ call đến github bằng cái URL của Jenkins (service của Jenkins)

Webhook nhận biết thay đổi của github thì n sẽ call đến jenkins

Jenkins thực hiện trên repository nào của github



Bất kỳ thay đổi nào của nhánh master trên repository này thay đổi thì jenkin n sẽ tự động build

Còn nhánh khác có thay đổi thì Jenkins sẽ ko care

# Jenkins + ansible

Jenkins – ansible trên cùng 1 con

Cài plugin ansible

Install git trên con jenkins – ansible

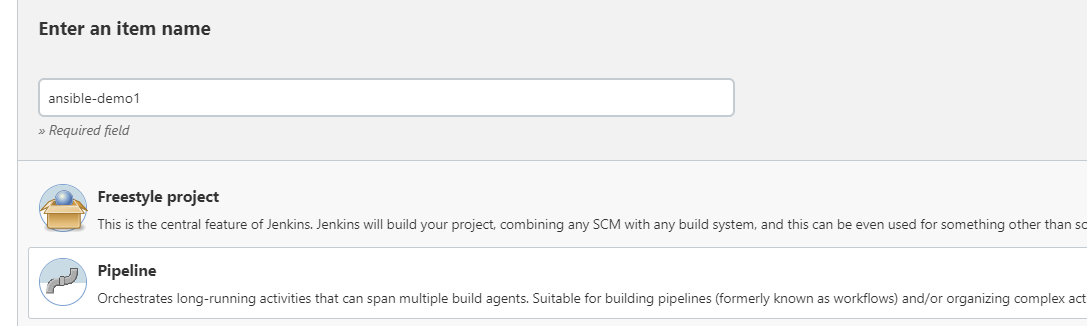
Manager Jenkins -> [Global Tool Configuration](http://192.168.226.128:8080/configureTools/)



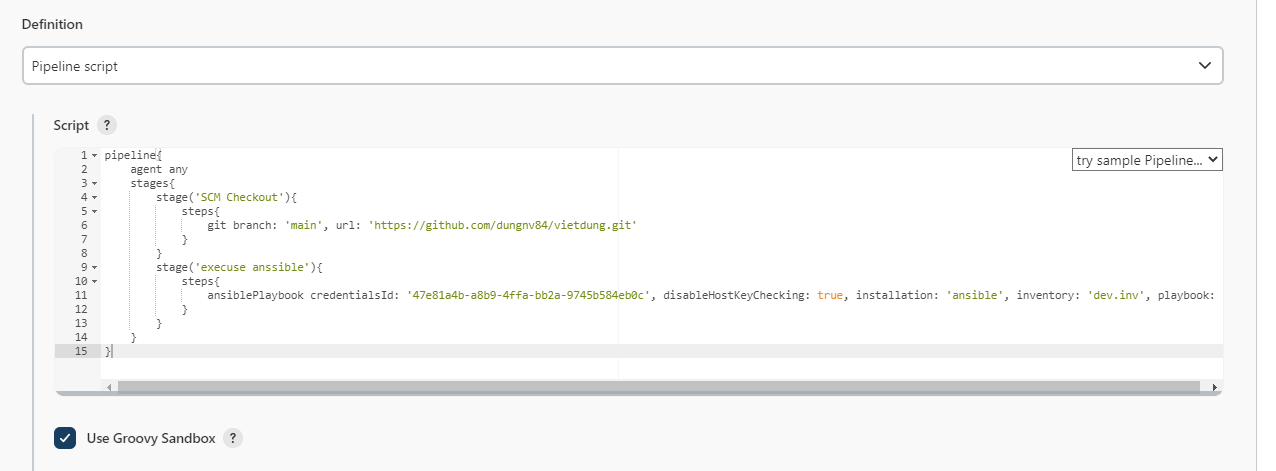


Tên là bất kỳ, tool này để tý dùng

Tạo mới 1 pipeline, dùng pipeline để có thể refer đến git, ansible, …



Add script



Step là được clone từ pipeline Syntax

# Step: là các bước thực hiện

Step là những cái được gen trong pipeline syntax

pipeline{

    agent any

    stages{

        stage('SCM Checkout'){

            steps{

                git branch: 'main', url: 'https://github.com/dungnv84/vietdung.git'

            }

        }

        stage('execuse anssible'){

            steps{

                ansiblePlaybook credentialsId: '47e81a4b-a8b9-4ffa-bb2a-9745b584eb0c', disableHostKeyChecking: true, installation: 'ansible', inventory: 'dev.inv', playbook: 'apache.yml'

            }

        }

    }

}

Step1: Clone git repo, trong git repo có

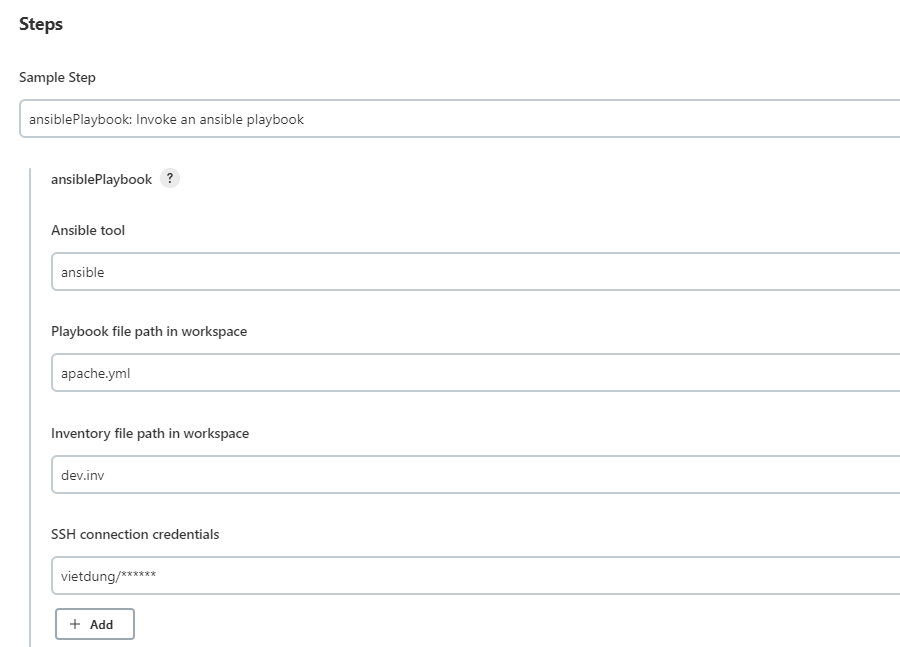
* file playbook để install web server và đẩy source code từ repo về server web
* inventory – IP server
* file HTML để chạy



Gen ra rồi copy vào step 1: coi như là đã pull được git repo về,

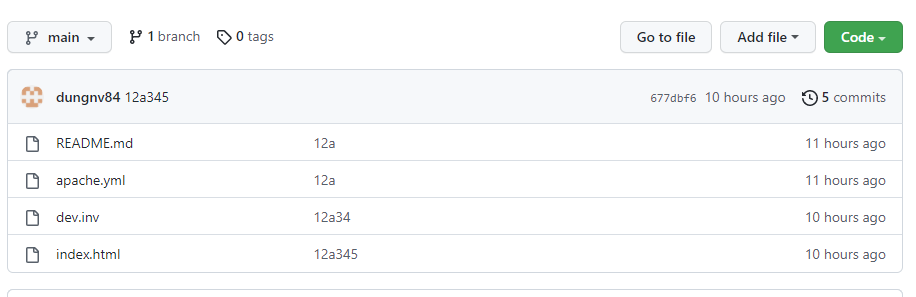
Gọi là task check out

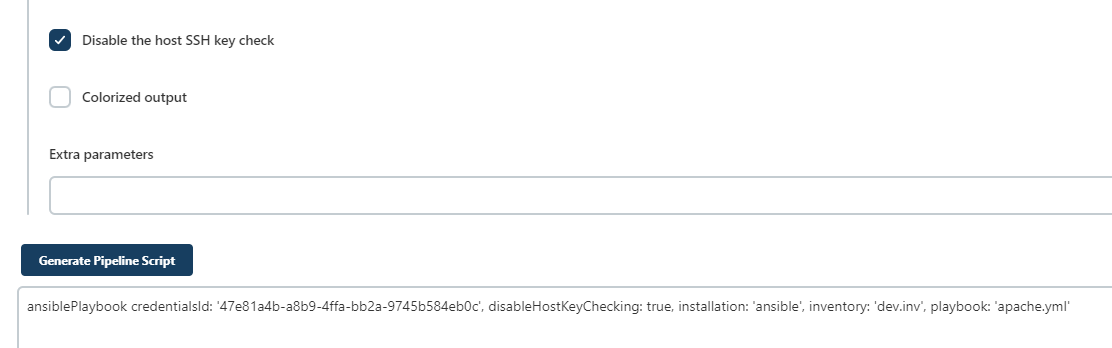
Step2: chạy các file ansible



Asible tool: tức là dùng ansible trên chính con jenkins đang chạy, tên ansible là tên vừa được tạo ra ở **global tool**

Còn các file playbook, inventory được tính từ vị trí đầu repo





Bỏ cái disable check SSH key đi, Case này gặp r, issue viết trong file ansible

Copy cái gen vào step 2

File playbook.yml (apache.yml)

---

- hosts: webservers

  become: True

  tasks:

    - name: Install packages

      yum:

        name: "httpd"

        state: "present"

    - name: Start Apache server

      service:

        name: httpd

        state: started

        enabled: True

    - name: Deploy static website

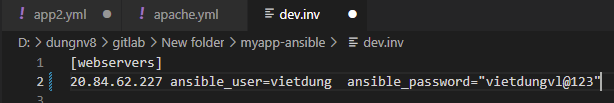
      copy:

        src: index.html

        dest: /var/www/html/

...

File inventory (dev.inv)



IP của con server web, con des để chạy

Oke build now



Oke web server đã chạy với nội dung từ file index.html của repo

# JenkinsFile

Tạo 1 thư mục code trên local

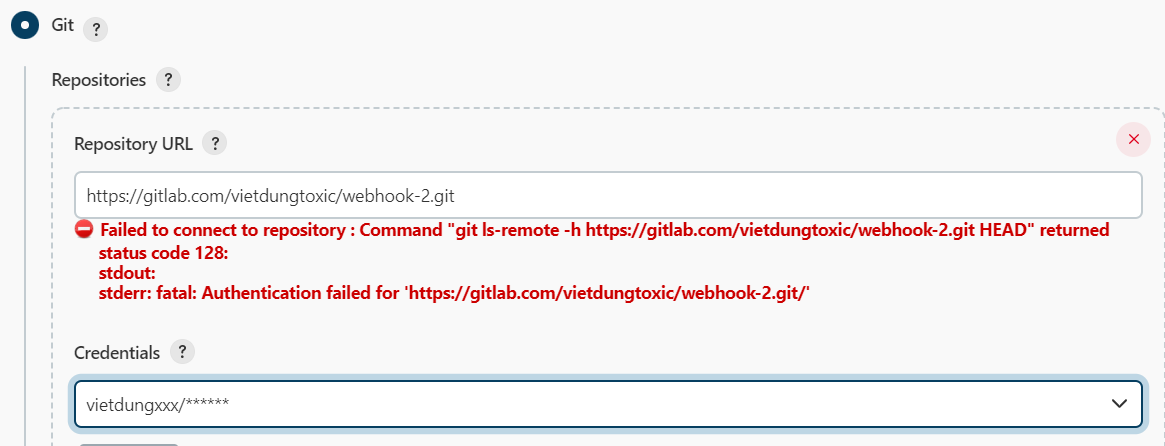
# Tích hợp Gitlab – jenkins

Install Flugin

* GitLab
* Gitlab Hook

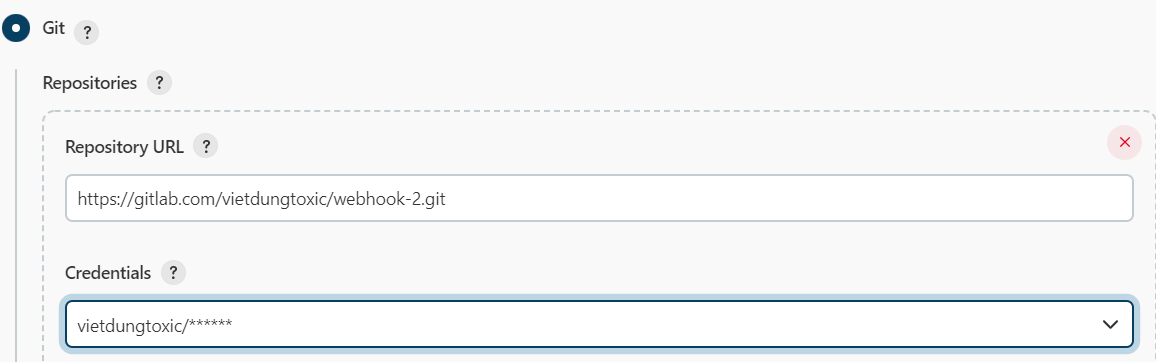


Sau khi install xog thì phần Build Triggers sẽ có GitLab webhook



Lưu ý rằng việc credential này tới gitlab phải điền đúng user/password

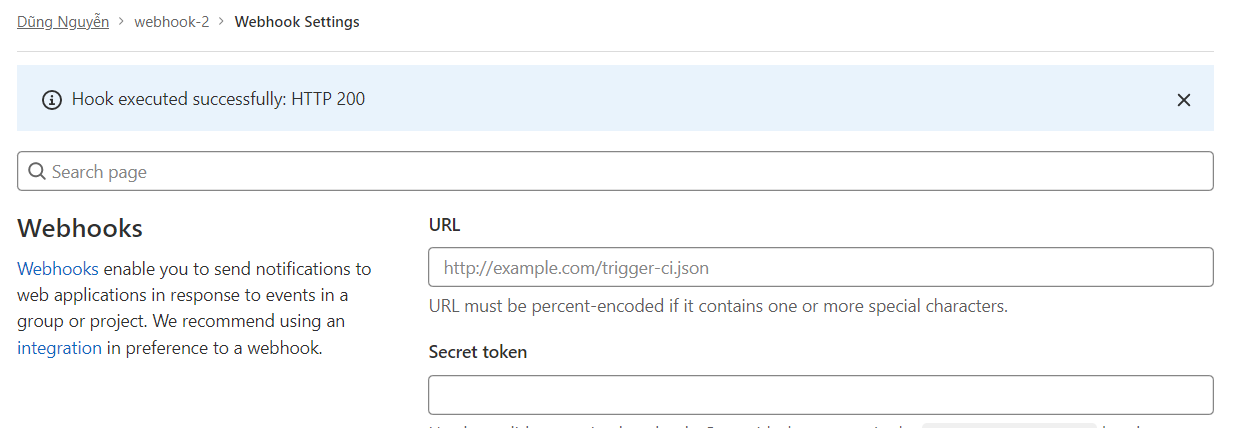
* Kqua đúng



Check username của git repo thì vào user-setting -> account

Xog vào pipeline -> setting-> webhook, copy đoạn URL trên jenkins vào

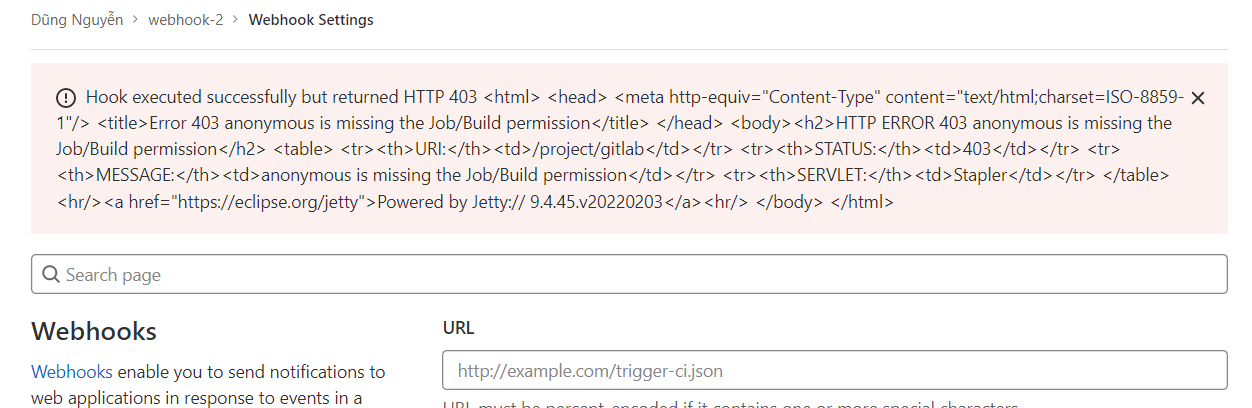
Xog test lên hiện ra 200 là oke rồi



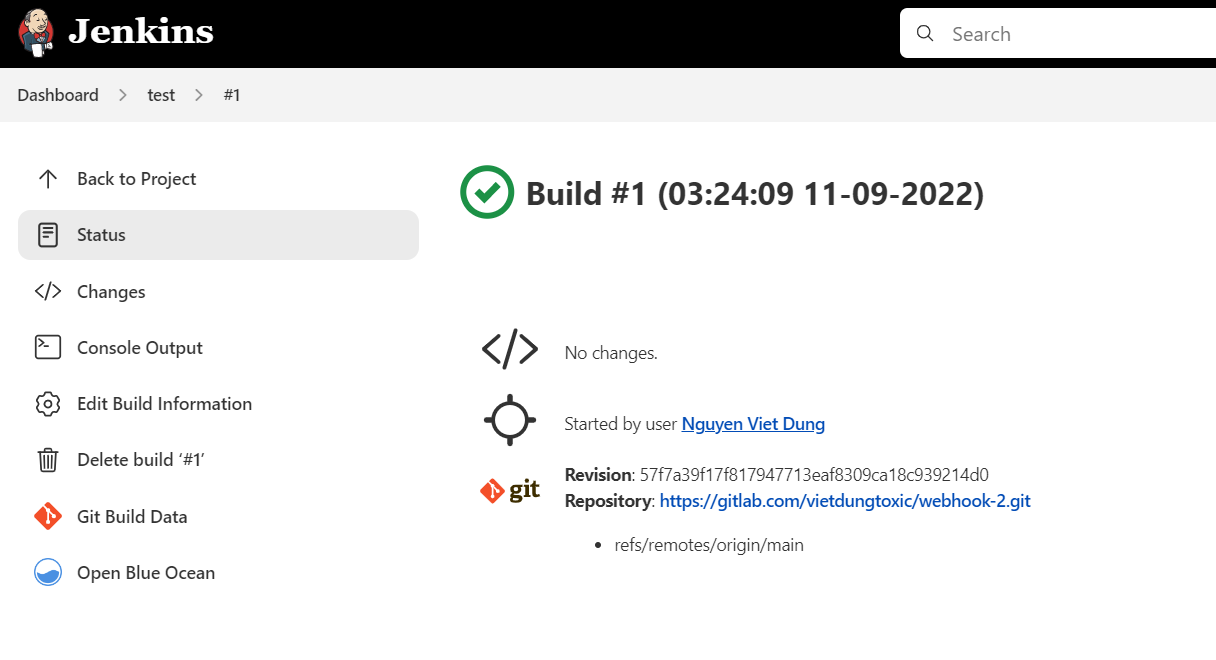
Lỗi này là jenkins chưa ssh key

Nhầm, vào manager jenkins -> config system -> bỏ tich cái authen này đi





Vào jenkins build thử



Oke jenkins đã gọi dk đến gitlab

# Hướng Dẫn Publish Lên IIS Và Cài Đặt CI/CD Với Jenkin (Publish TO IIS , Setup CI/CD With Jenkin)

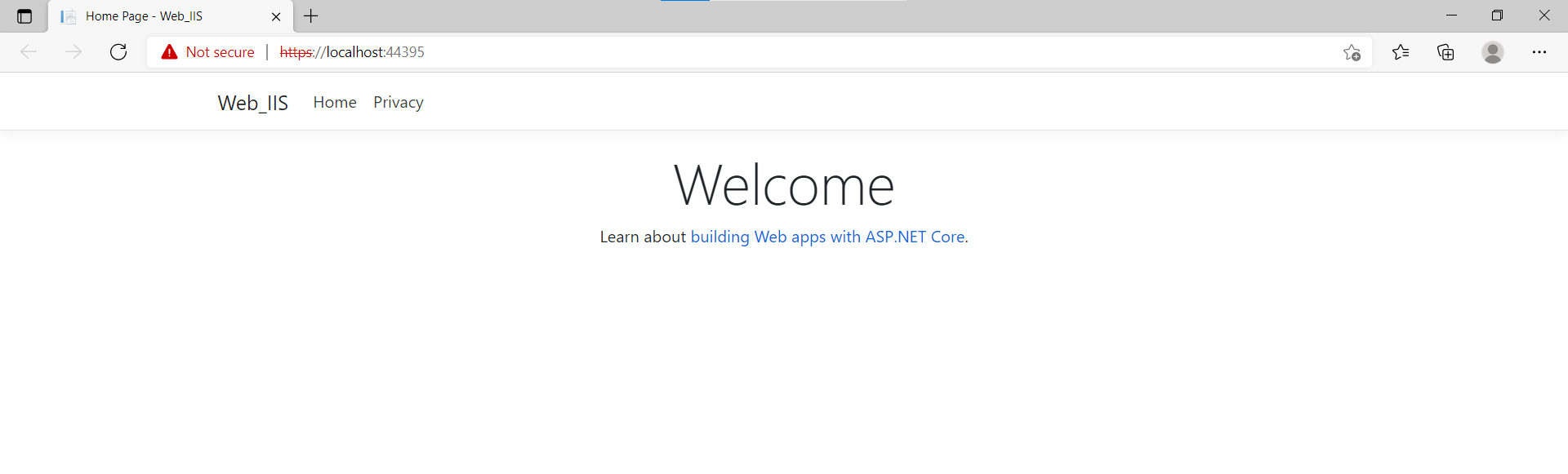
1. Install .netcore sdk => phục vụ câu lệnh dotnet restore, donet build
2. Install git scm => pull code
3. install JDK => mã nguồn mở thì cài
4. install jenkins.msi
5. install ngrok

deploy web .Net bằng tay trước, sau đó CICD sau, để hiểu bản chất

* tạo tay

Vào visual studio, tạo 1 project ASP .Net core web

Vào dự án ấn express, n chạy như này là oke

****

Đây là mặc định giống web html

[**https://www.w3schools.com/bootstrap/bootstrap\_tables.asp**](https://www.w3schools.com/bootstrap/bootstrap_tables.asp)

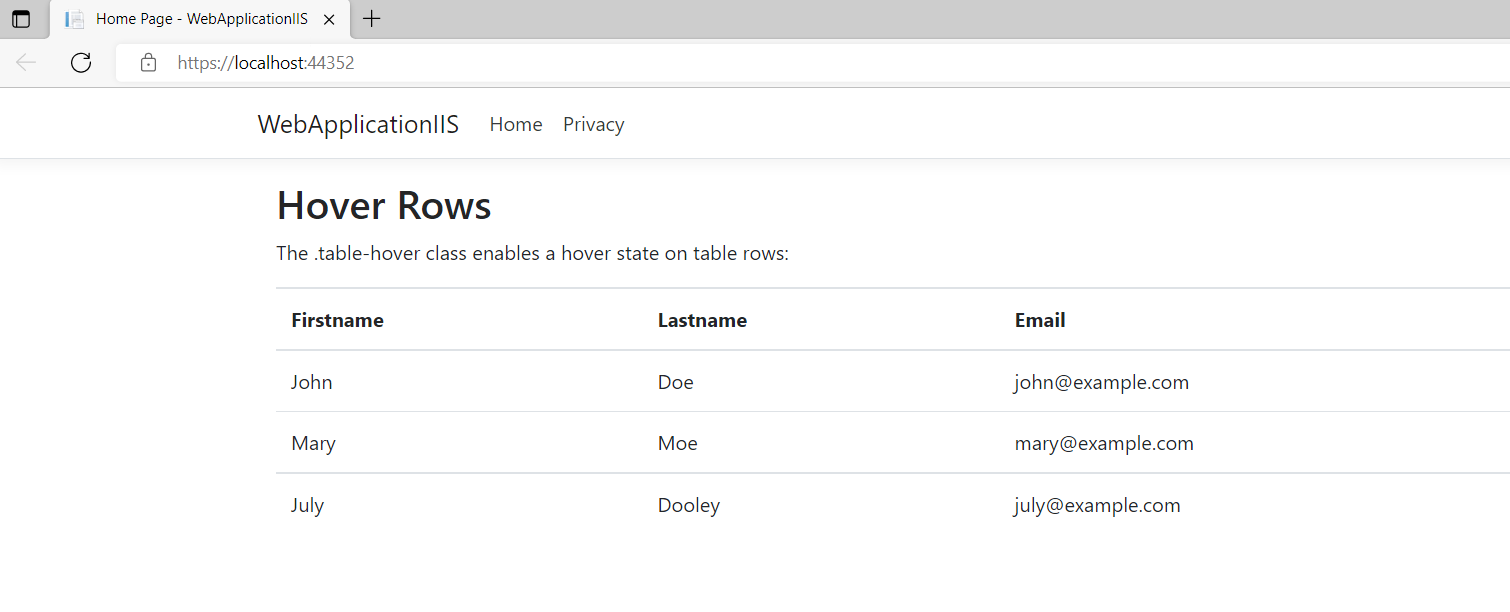
copy 1 đoạn text trên w3school copy về file index.cshtml

sau đó lại chạy tiếp

Express



Kết quả là:

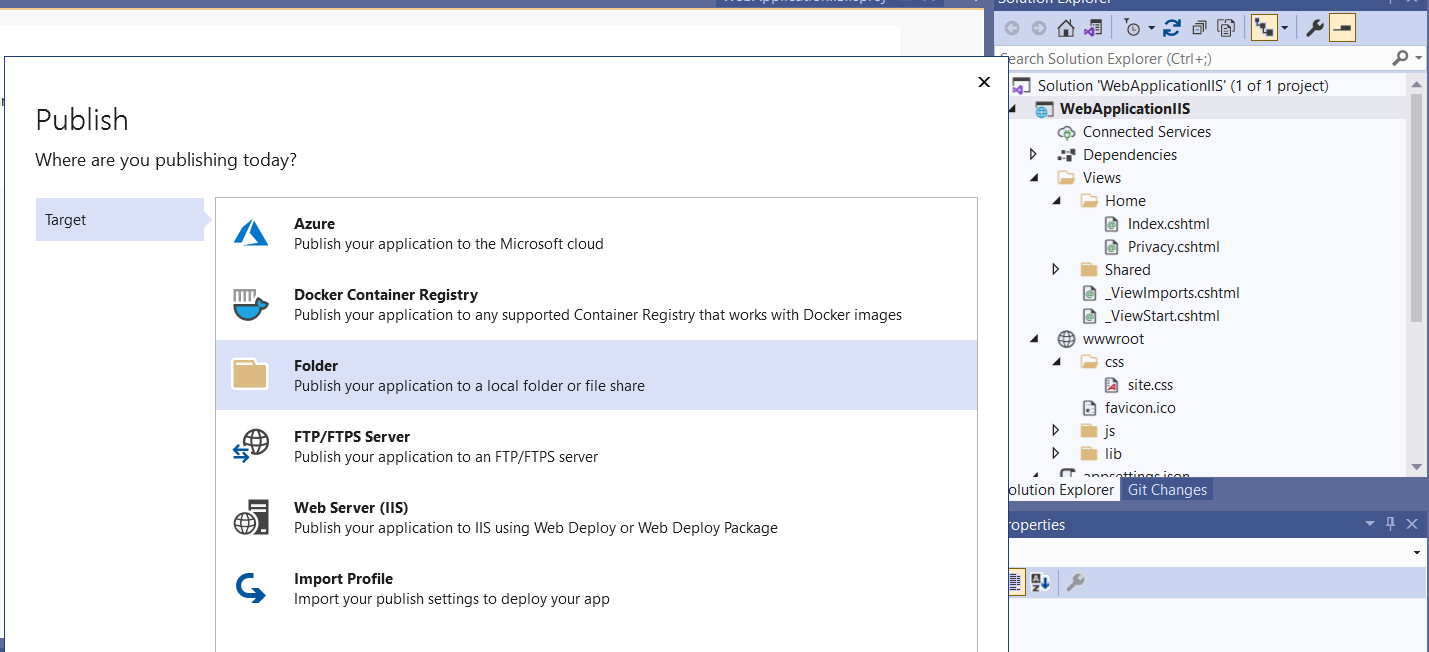


Oke đã lên

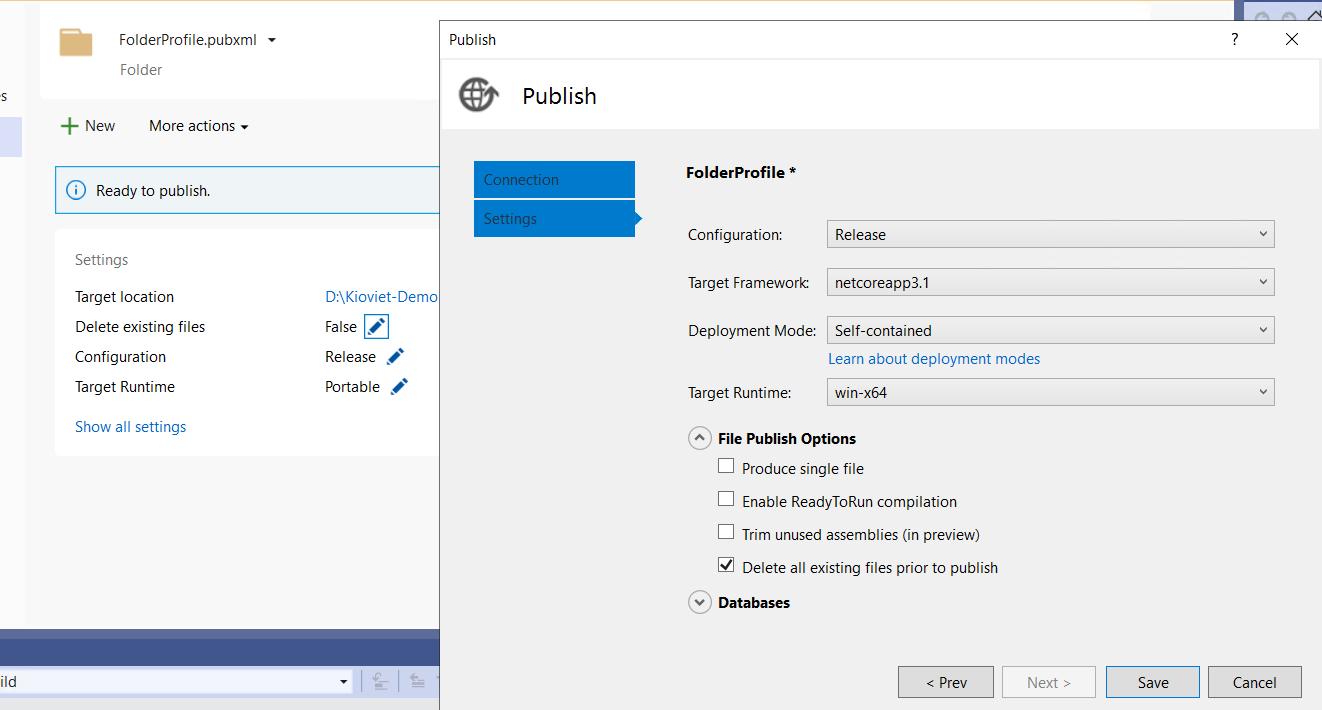
B2: Publish

Tại sao cần phải publish, bthuong để chạy ta phải ấn **IIS Express**

Nhưng để chạy trong web, ta cần publish n vào 1 folder nào đó rồi dùng IIS web để trỏ đến folder vừa được publish



Publich cái này vào folder **kiotviet-demo**



Chỉnh 1 tý, r save lại, publish,

Sau đó lại copy đống file vừa được publish sang 1 folder khác

Để làm j, folder đấy chỉ để publish thôi

Ta sẽ copy file trong folder publish sang folder của site web

Folder site web để chỉnh sửa thêm 1 số file

B3: tạo 1 git pipeline chưa source code, đây là 1 source code chính, chưa publish

## Jenkins dot Net

Tạo 1 pipeline

pipeline{

    agent any

    stages{                             // định nghĩa các giai đoạn của stage

        stage('clone code')             // tên của từng stage

        {                               // trong stage này sẽ chạy những steps nào

            steps{

                git credentialsId: '2def9ee2-4ce6-4265-a4d8-013ff29519c3', url: 'https://github.com/dungnv84/w3\_band.git'

            }

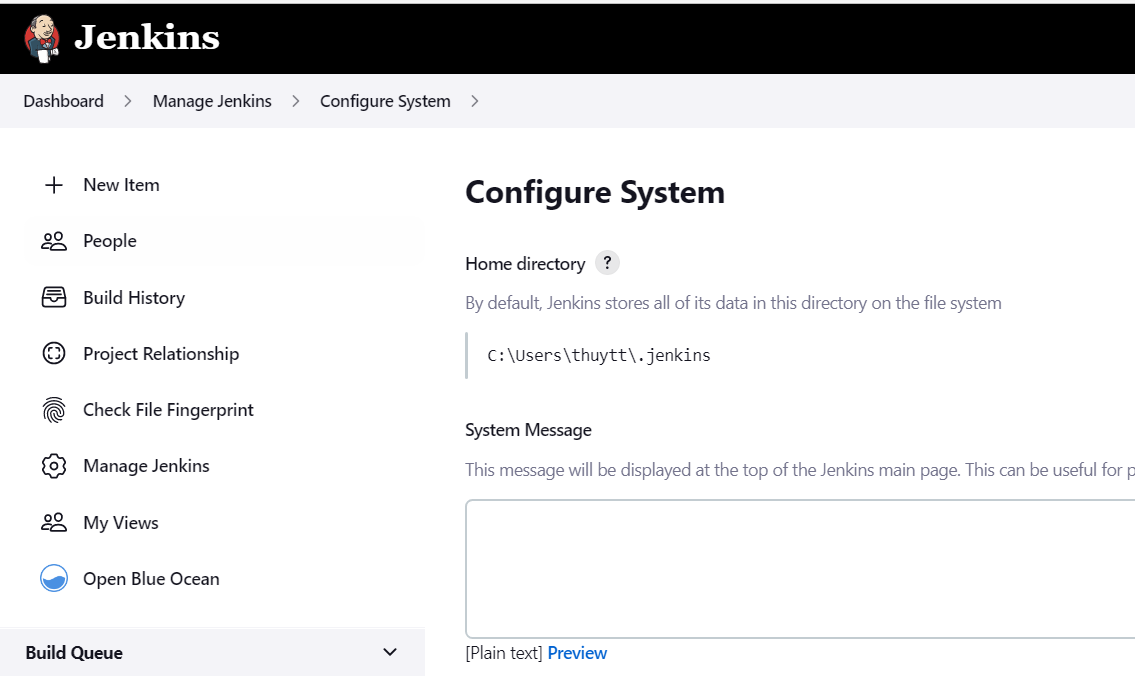
        }

    }

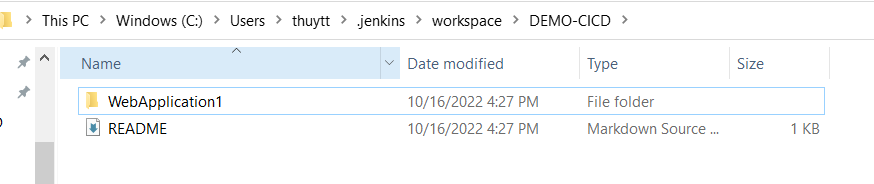
}

Lưu ý: Nếu ko credential đến link git được thì check ENV: ../git/bin

Oke build nè



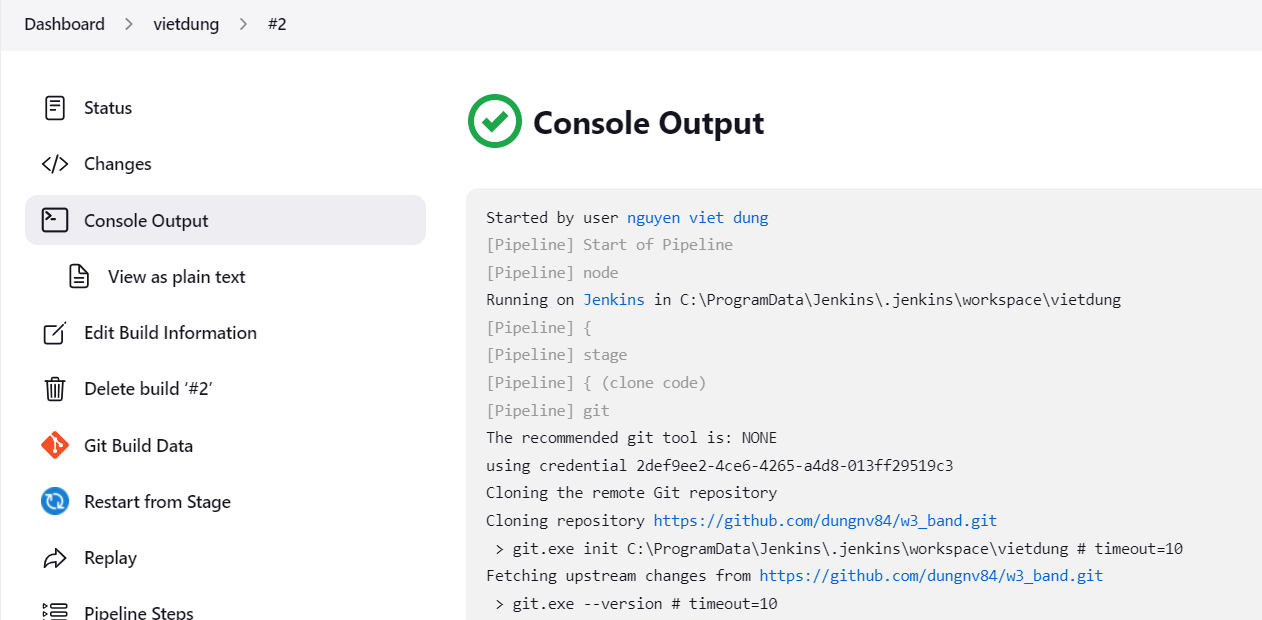
Mặc định khi build về n lưu ở đây



Build về trước

Khi clone code về thì cần restore

Tạm thời thế đã



Trên con jenkins

Nó đã clone được source code về và lưu ở folder C:\ProgramData\Jenkins\.jenkins\workspace\vietdung

# Jenkins Udemy

Lỗi Centos bị Appstream ko install, update được

cd /etc/yum.repos.d/

sed -i 's/mirrorlist/#mirrorlist/g' /etc/yum.repos.d/CentOS-\*

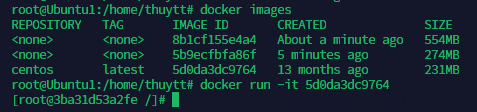
sed -i 's|#baseurl=http://mirror.centos.org|baseurl=http://vault.centos.org|g' /etc/yum.repos.d/CentOS-\*

yum update -y

* test image centos

docker pull centos

docker run –it [image-Name]



docker build -t image:v1 -f dockerfile .

