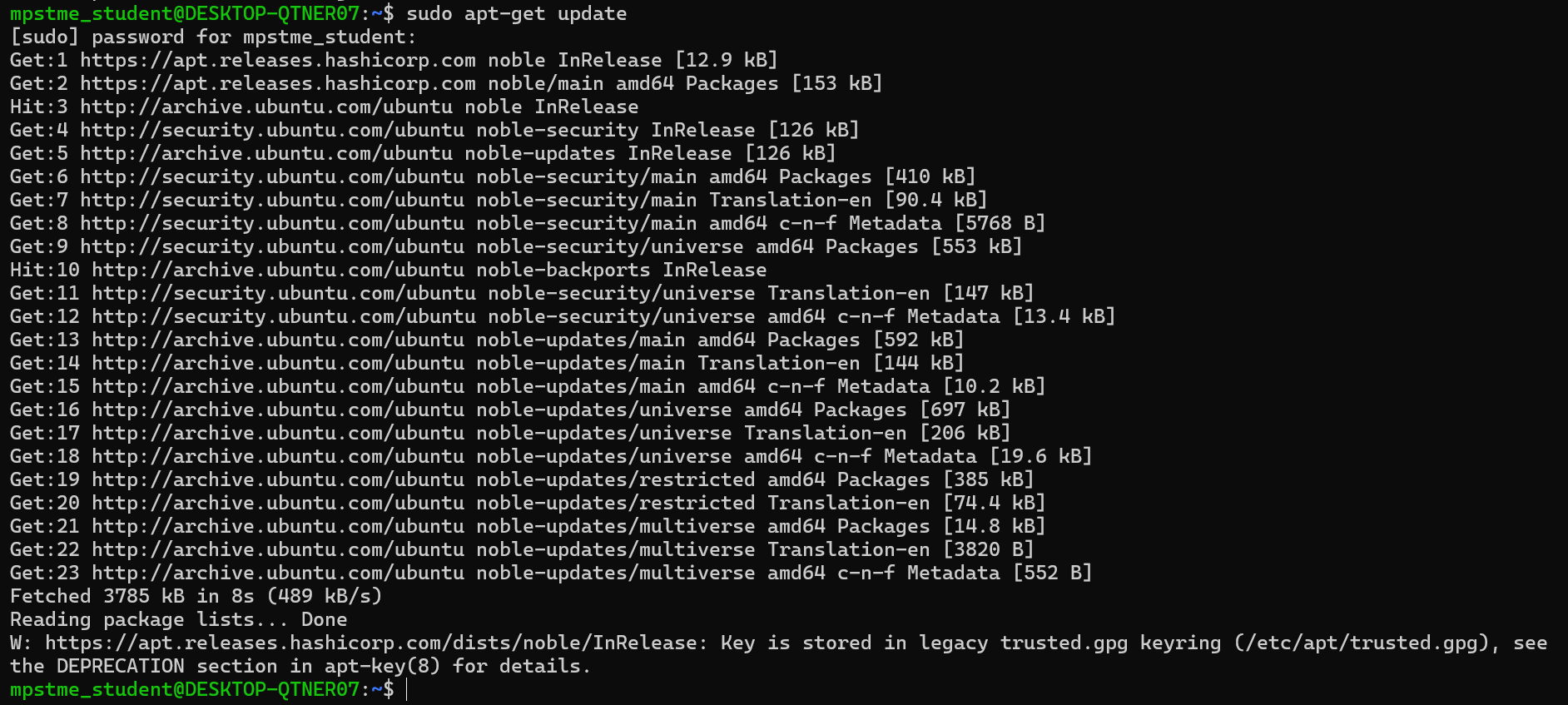
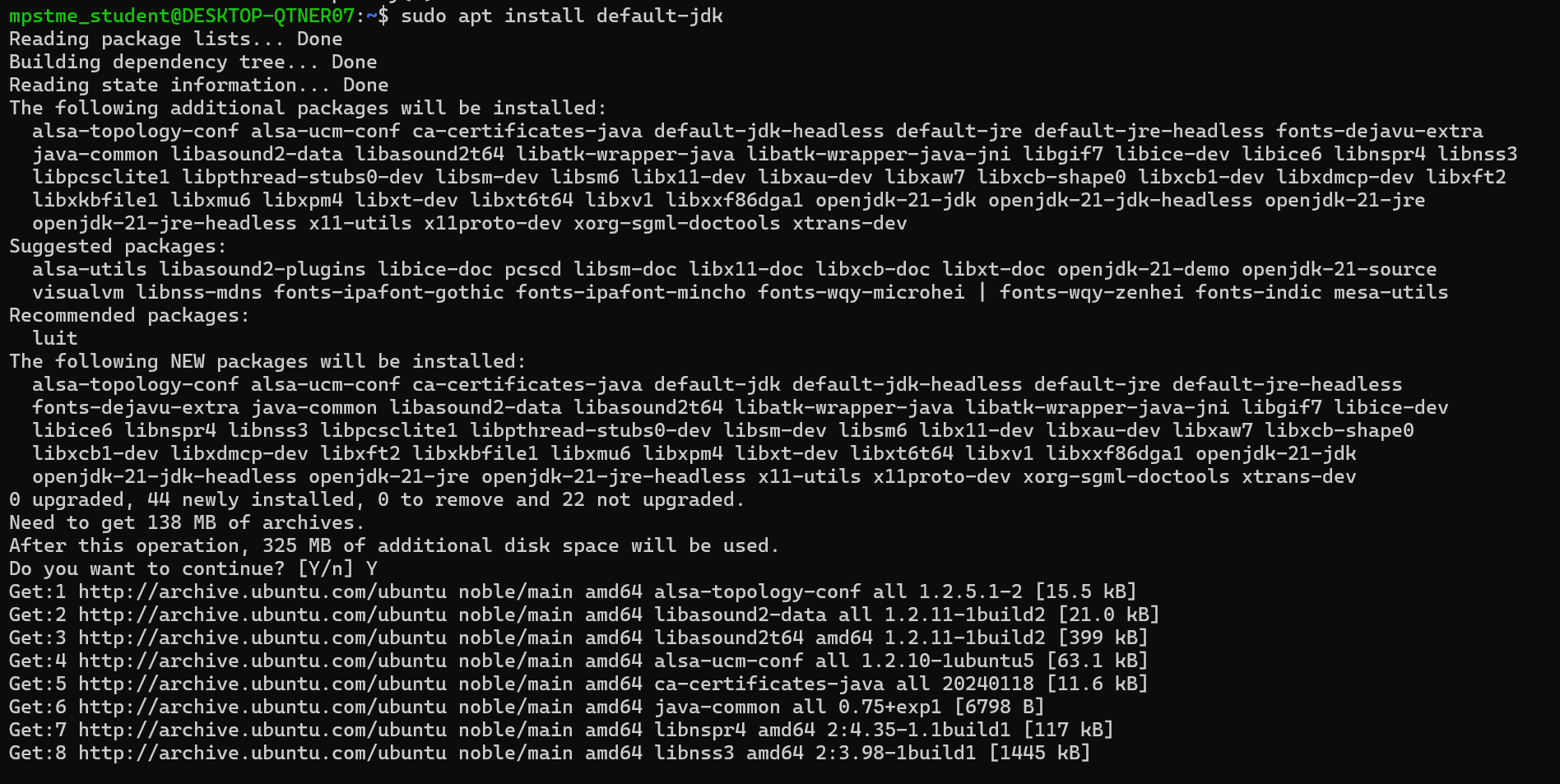
**Install Jenkins**

1. Update Ubuntu Repository: sudo apt-get update

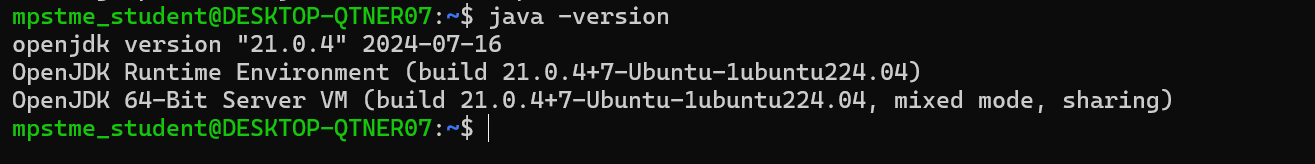


1. Jenkins needs Java17: Install Java Development Kit (JDK)

sudo apt install default-jdk



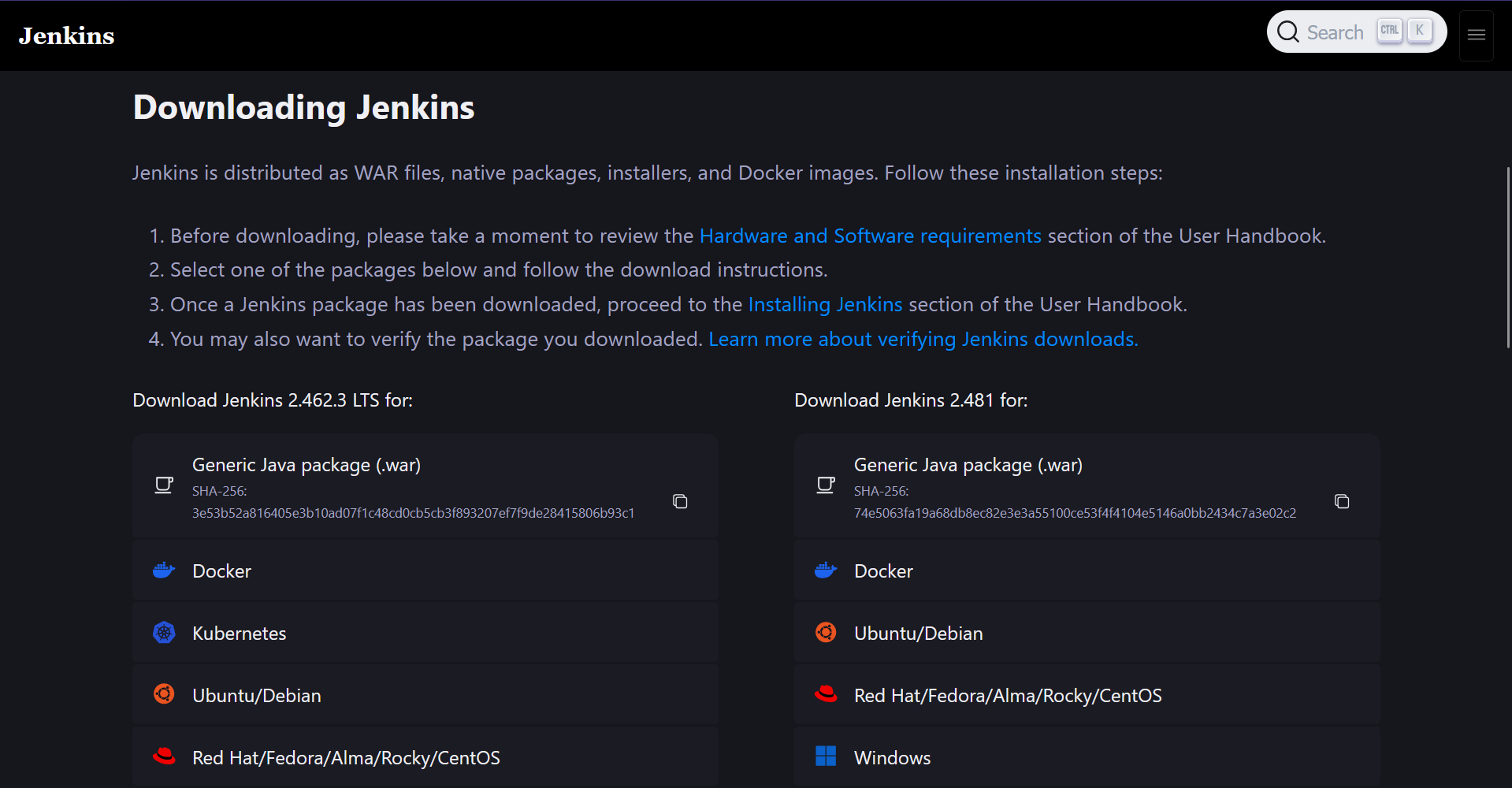
1. Check if java is installed properly: java -version



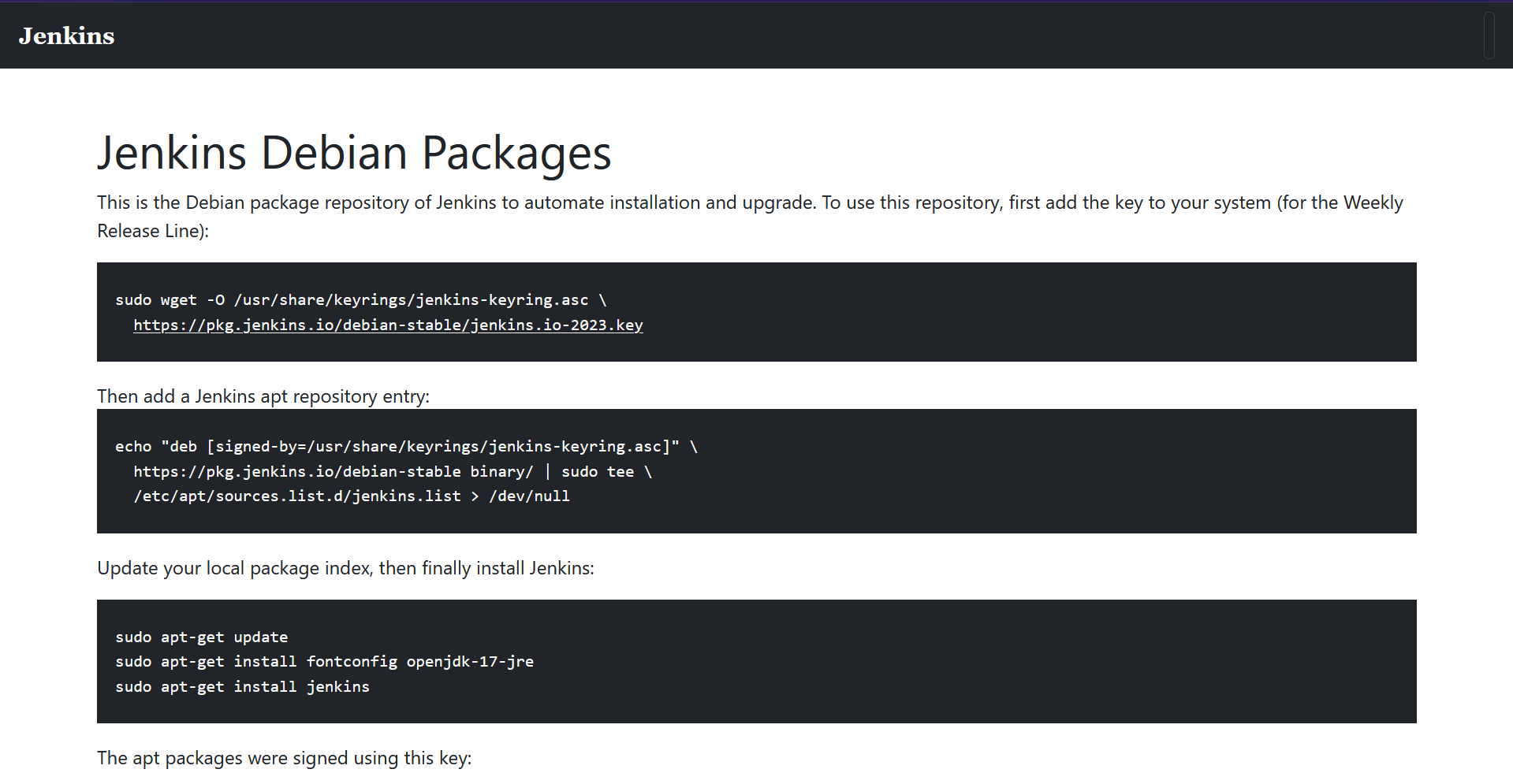
1. Now install Jenkins. Go to Jenkins.io



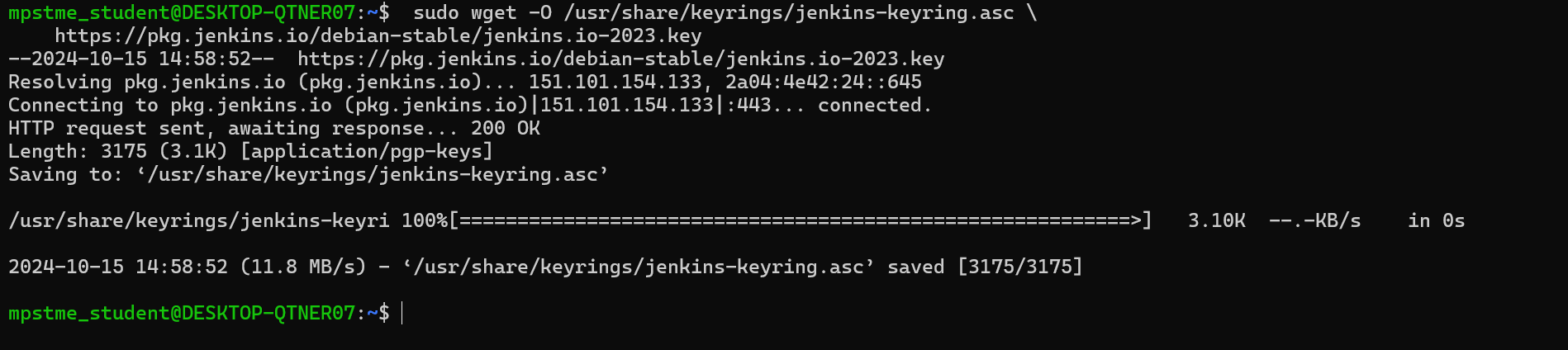
1. Click download. For LTS choose Ubuntu



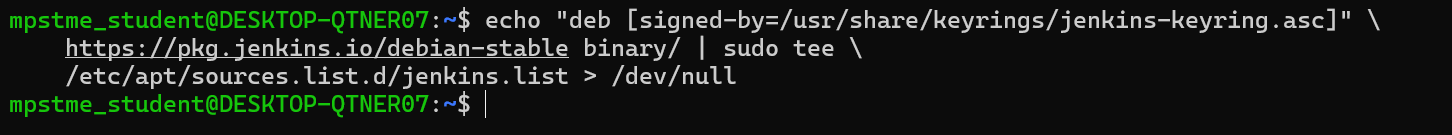
1. Run the commands given on WSL Ubuntu



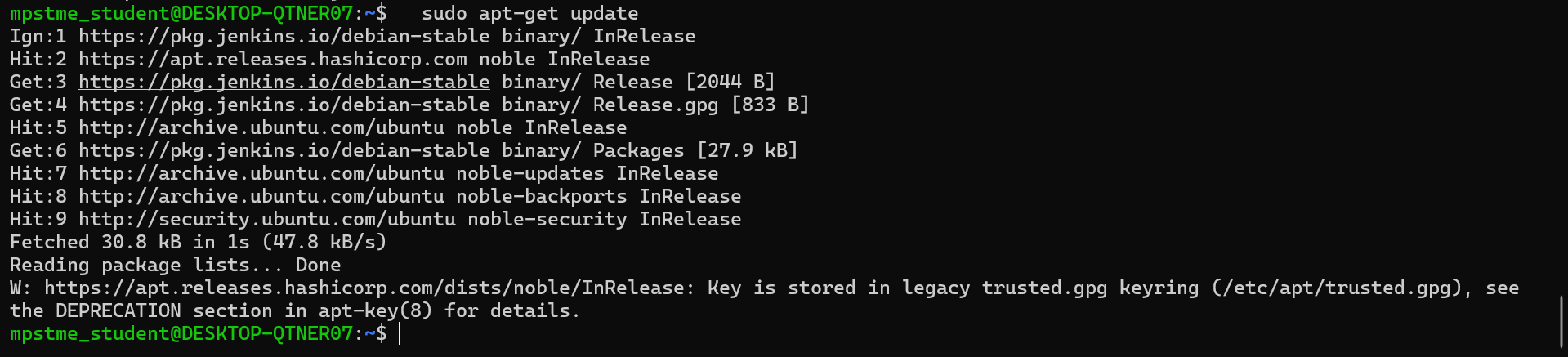
1. First add key to your system



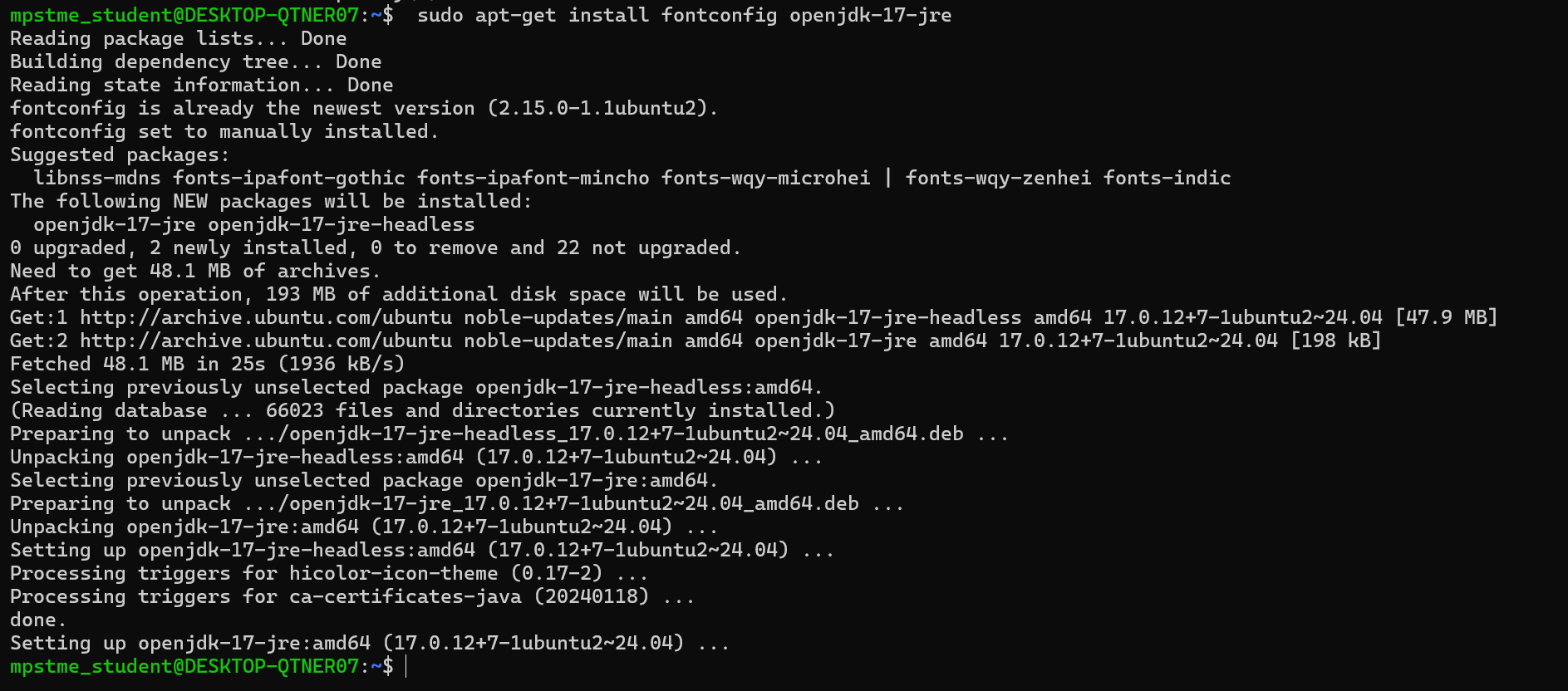
1. Add Jenkins apt repository entry



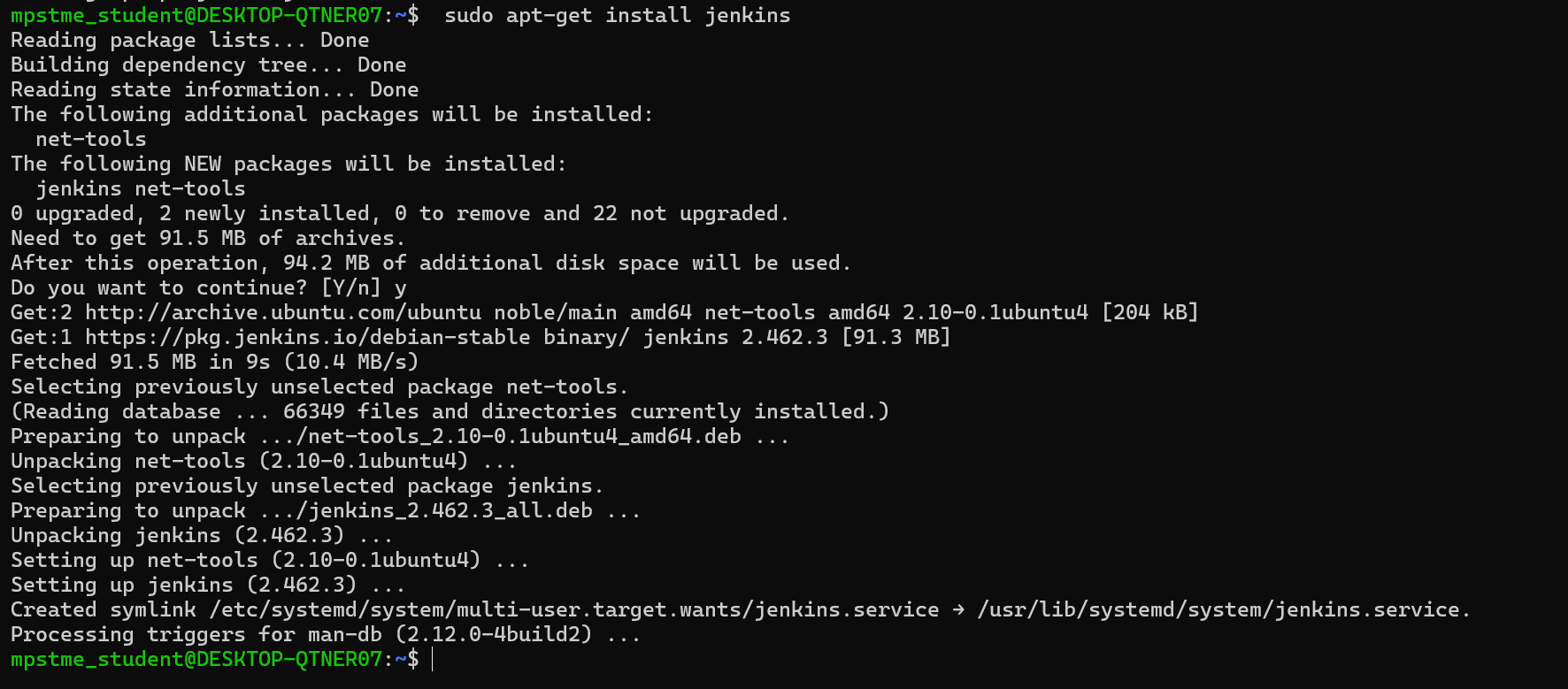
1. Update local package index



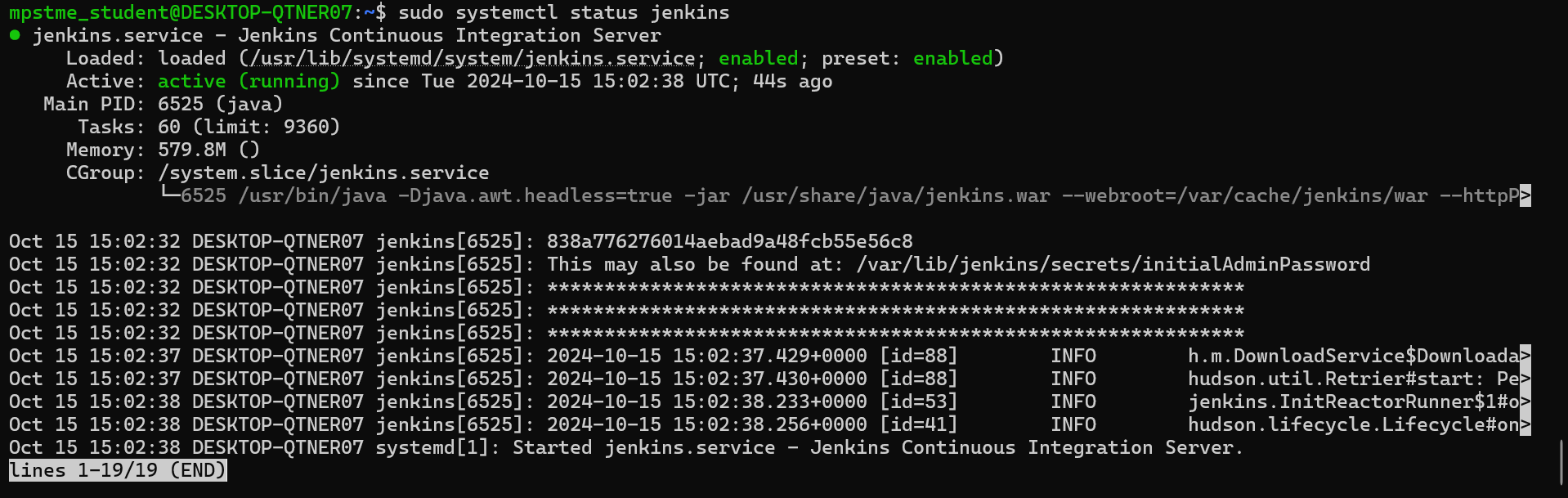
1. Finally install Jenkins



1. sudo apt-get install Jenkins



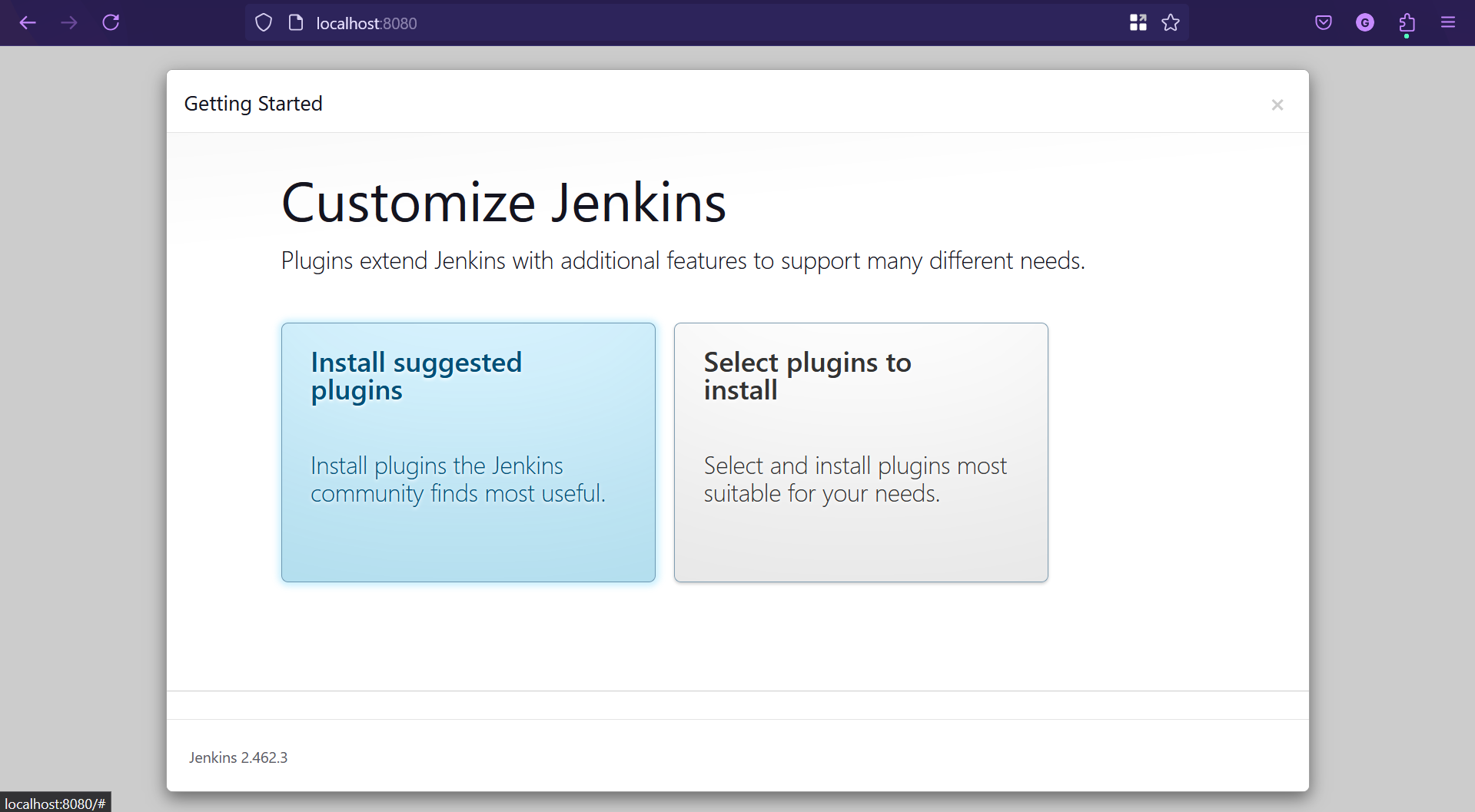
1. Check if Jenkins is installed



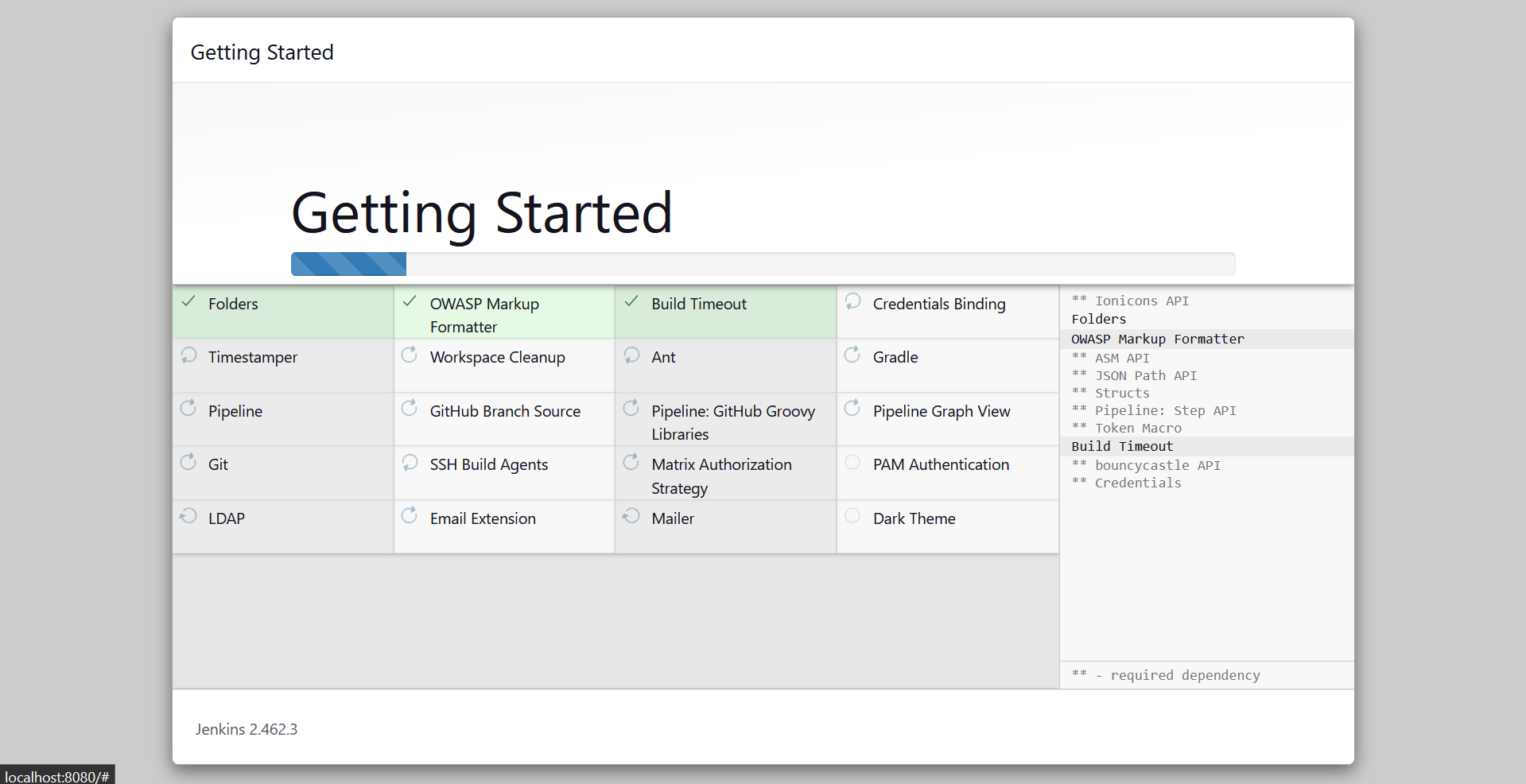
1. Copy password to unlock Jenkins: 838a776276014aebad9a48fcb55e56c8
2. Go to localhost:8080 and enter the copied password



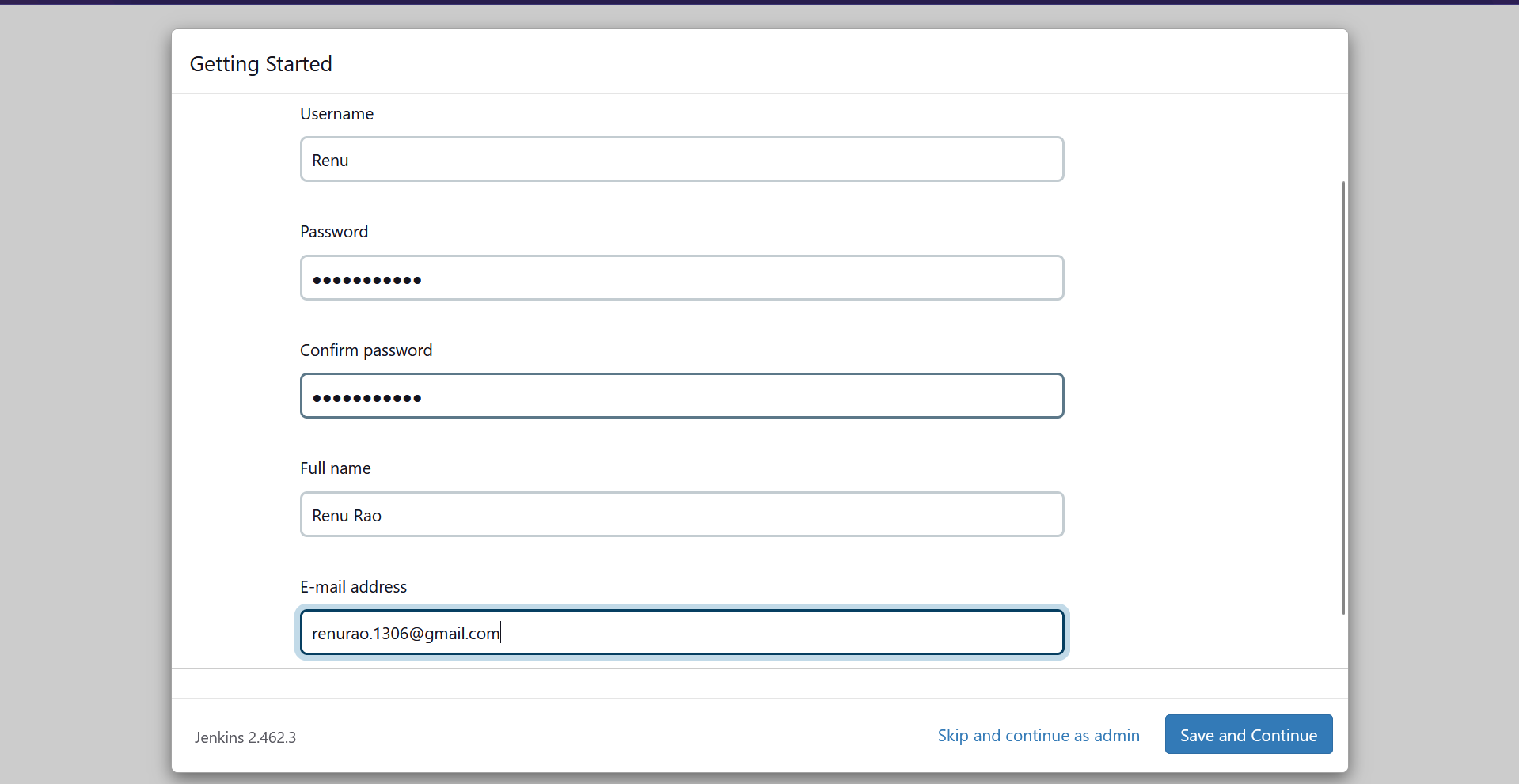
1. Click Install Suggested Plugins:

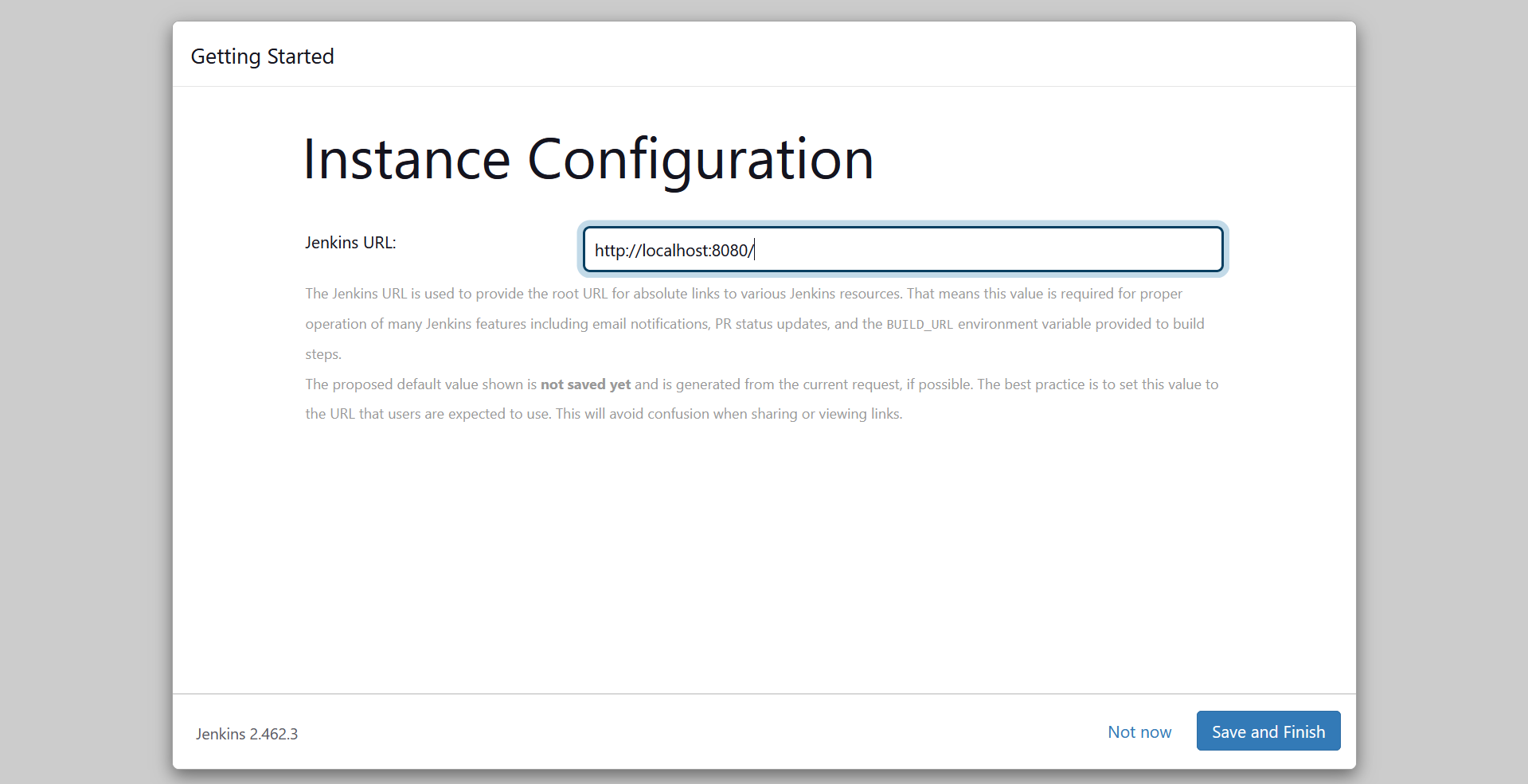


1. Installing:

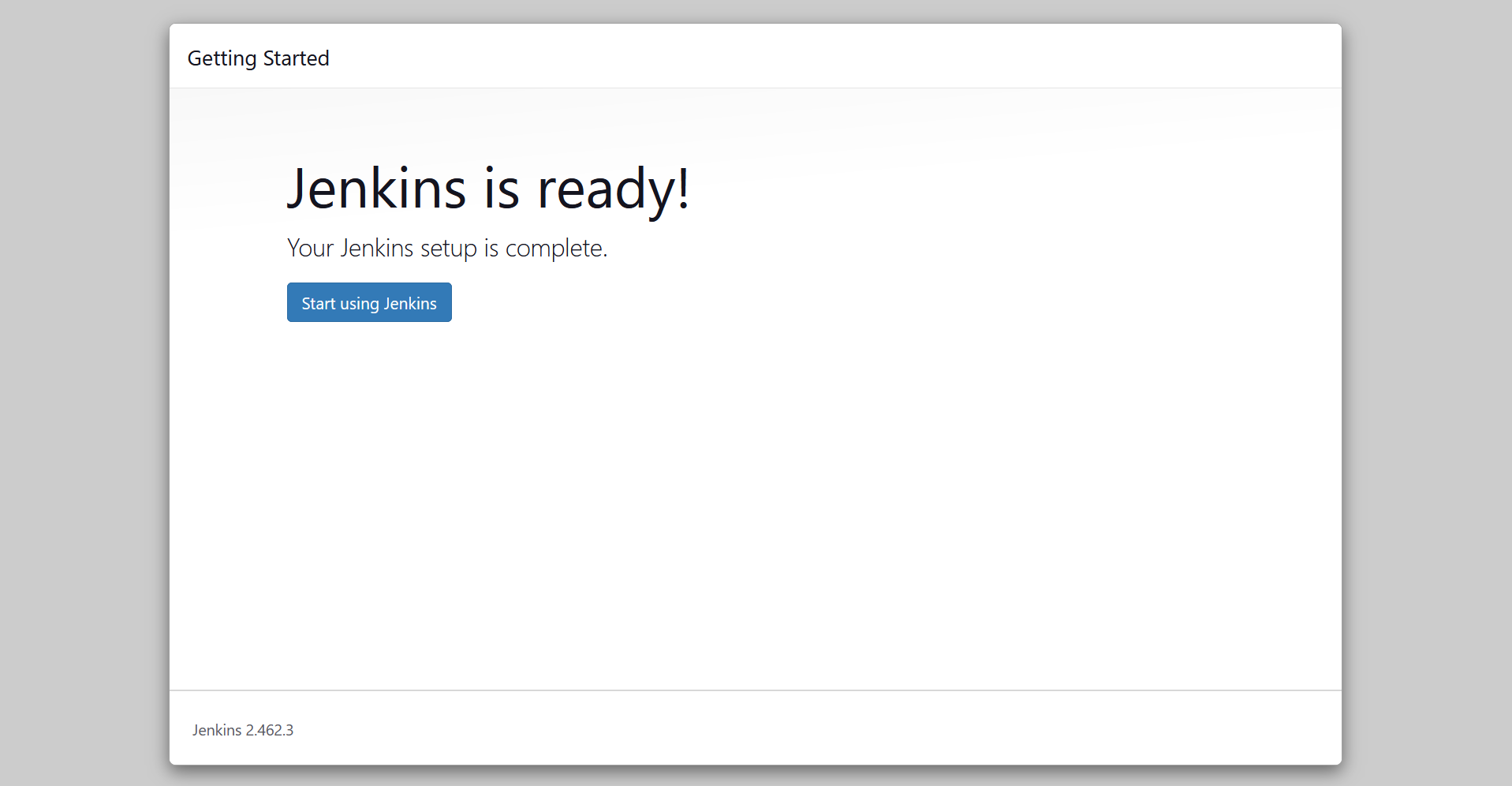


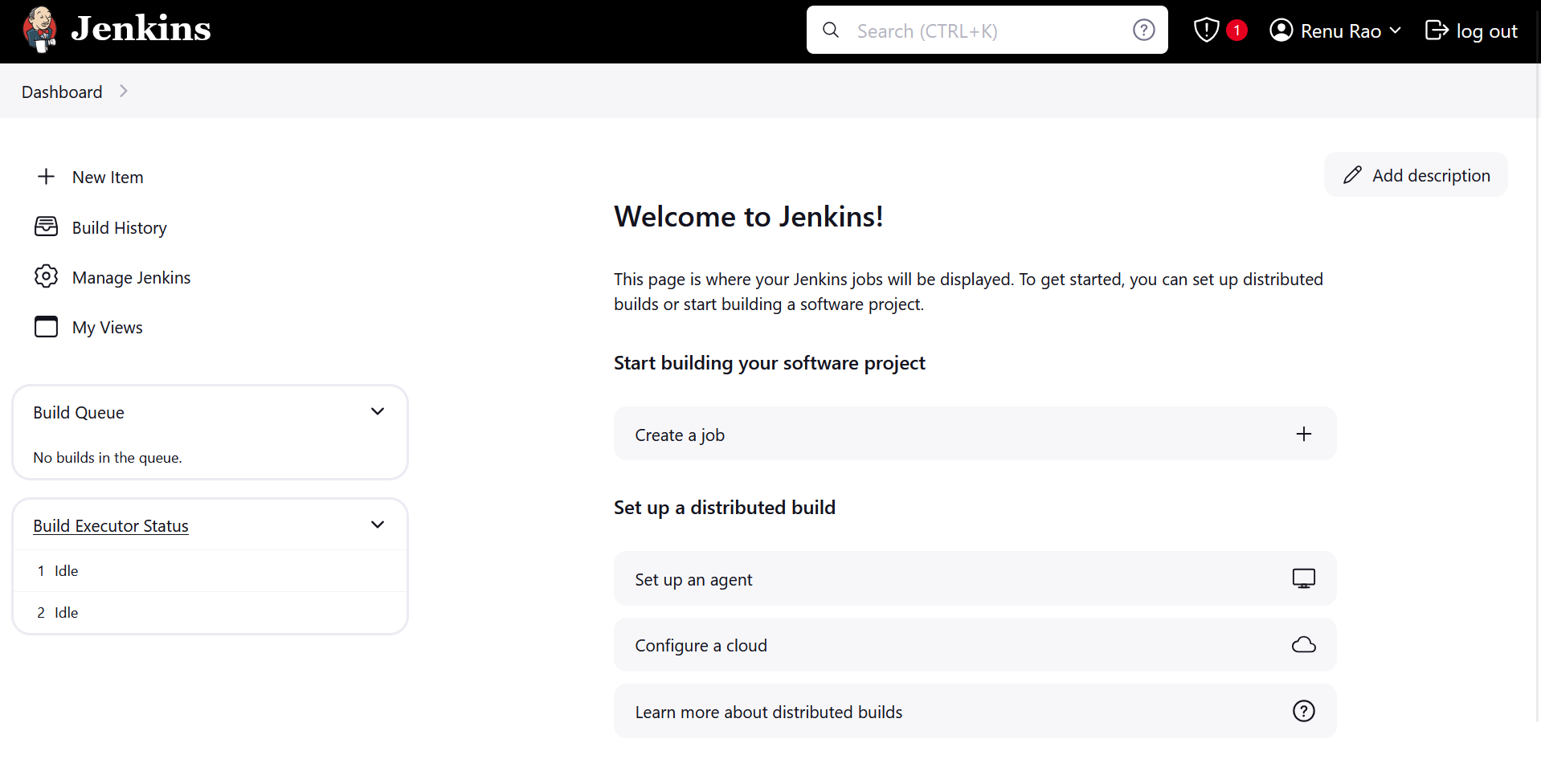
1. Create Account on Jenkins:



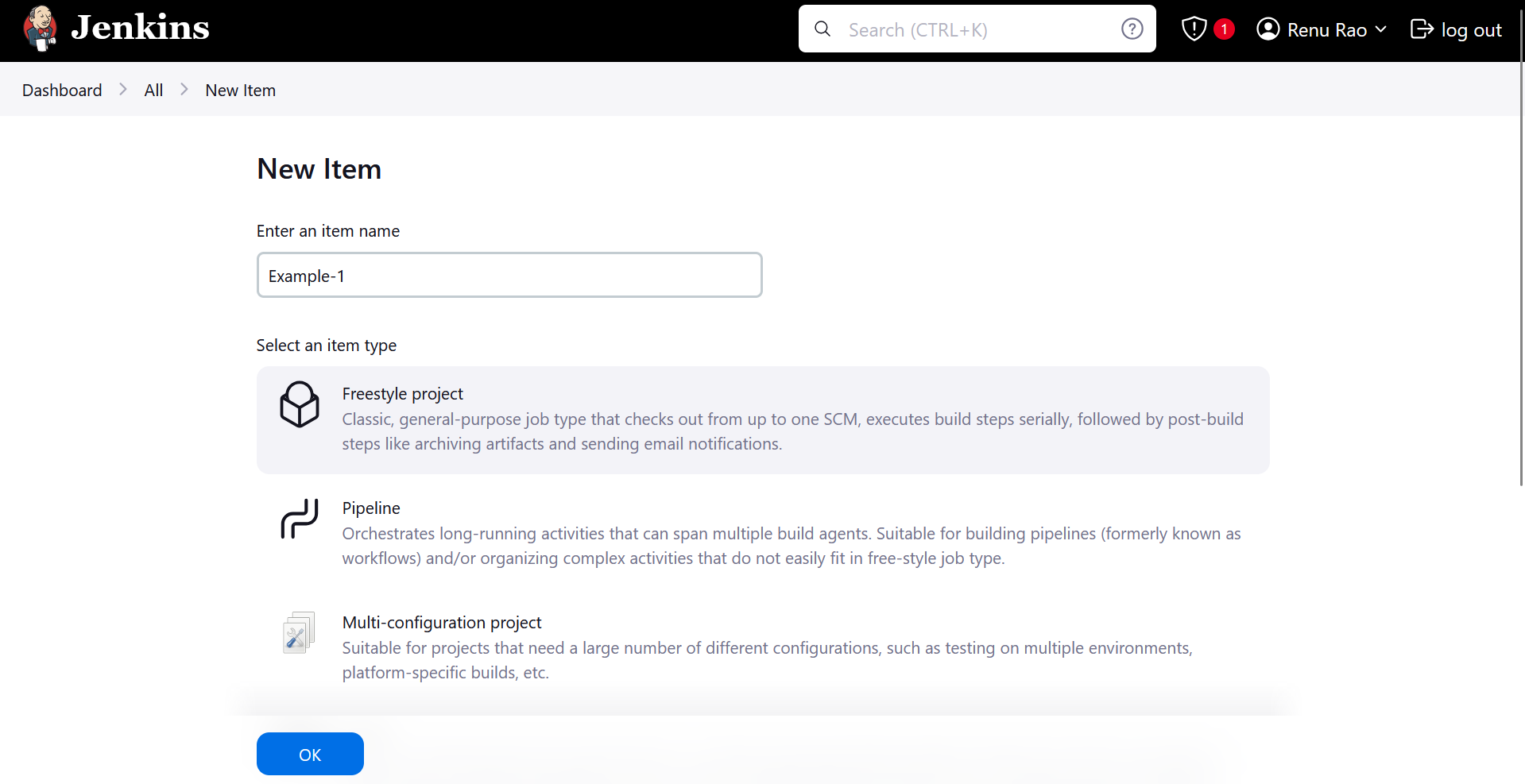


1. Jenkins is ready

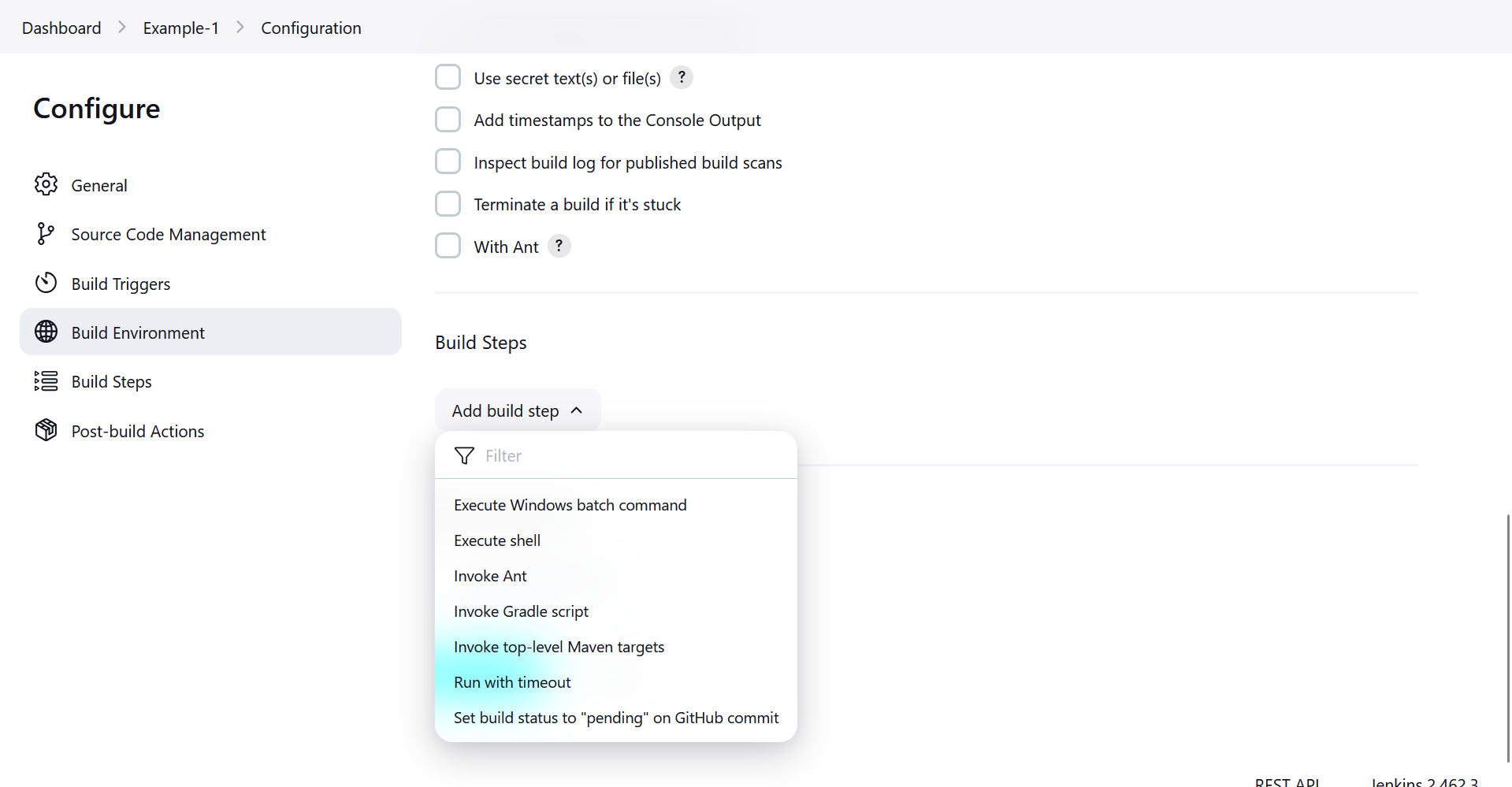




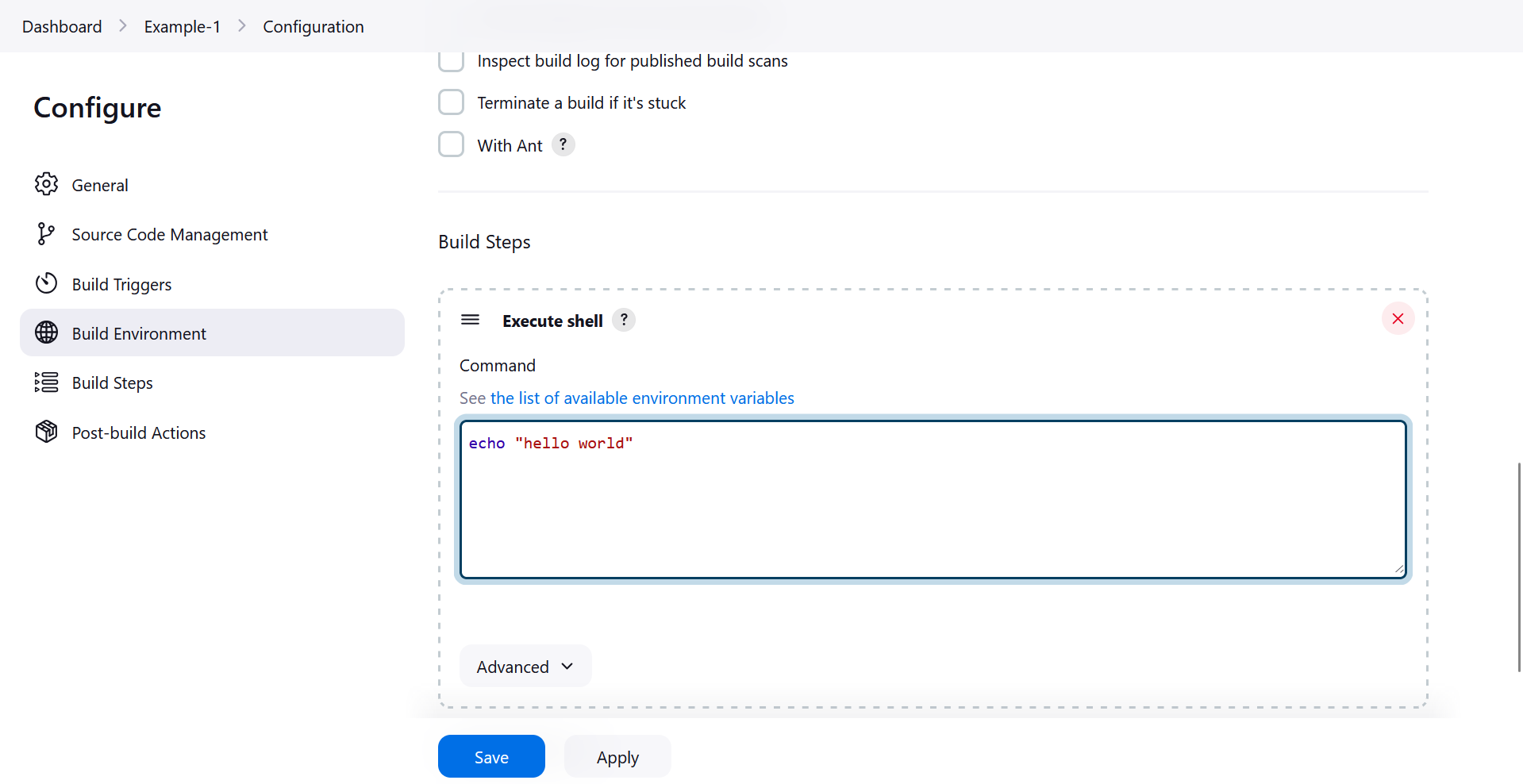
1. Click on New Item -> Freestyle Project



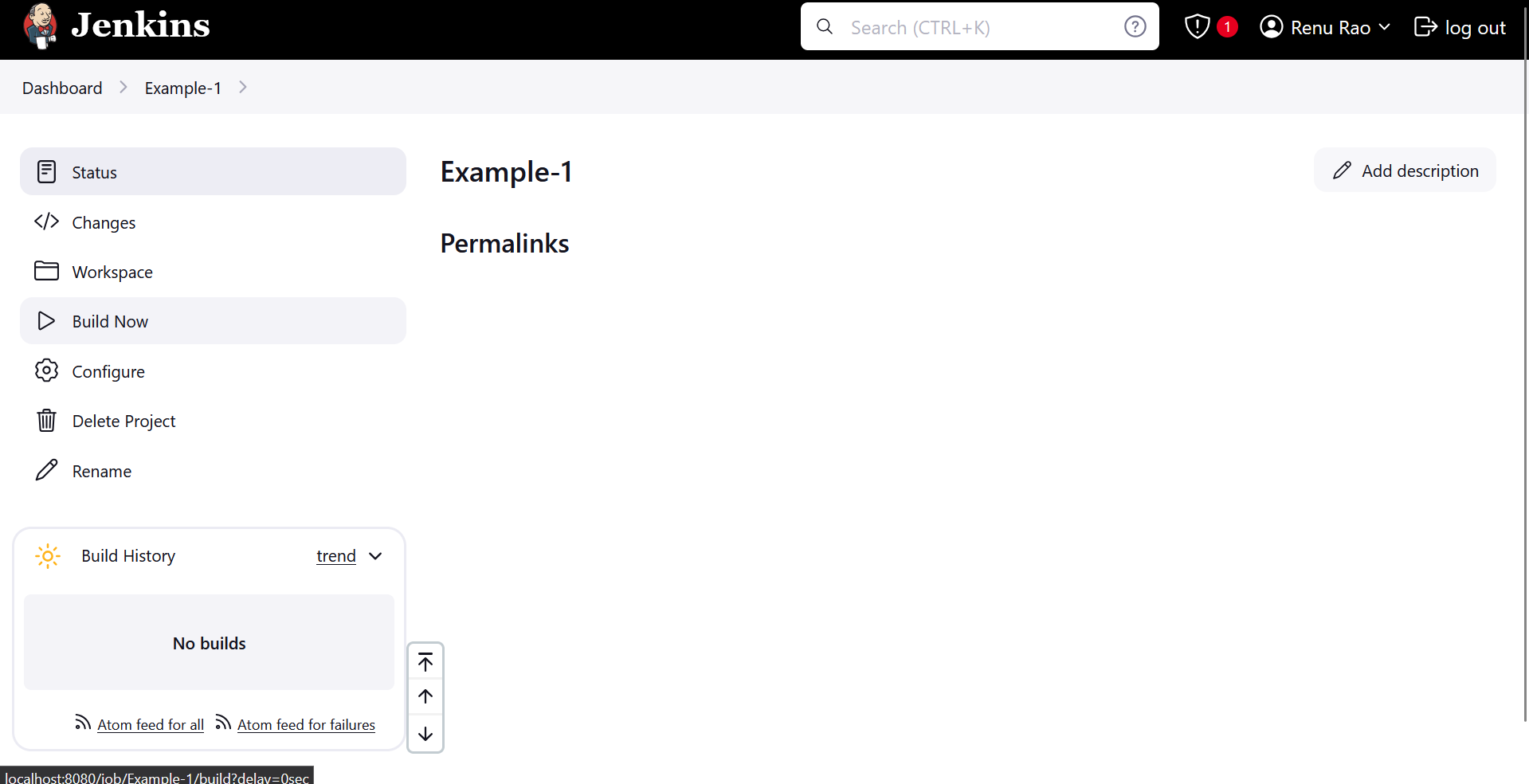
1. Go to Build steps-> add build step -> execute shell



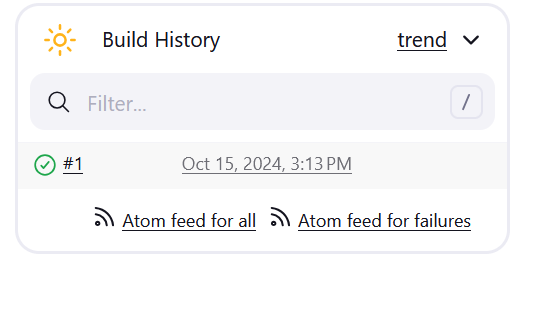
1. Write echo “hello world” -> Apply and Save



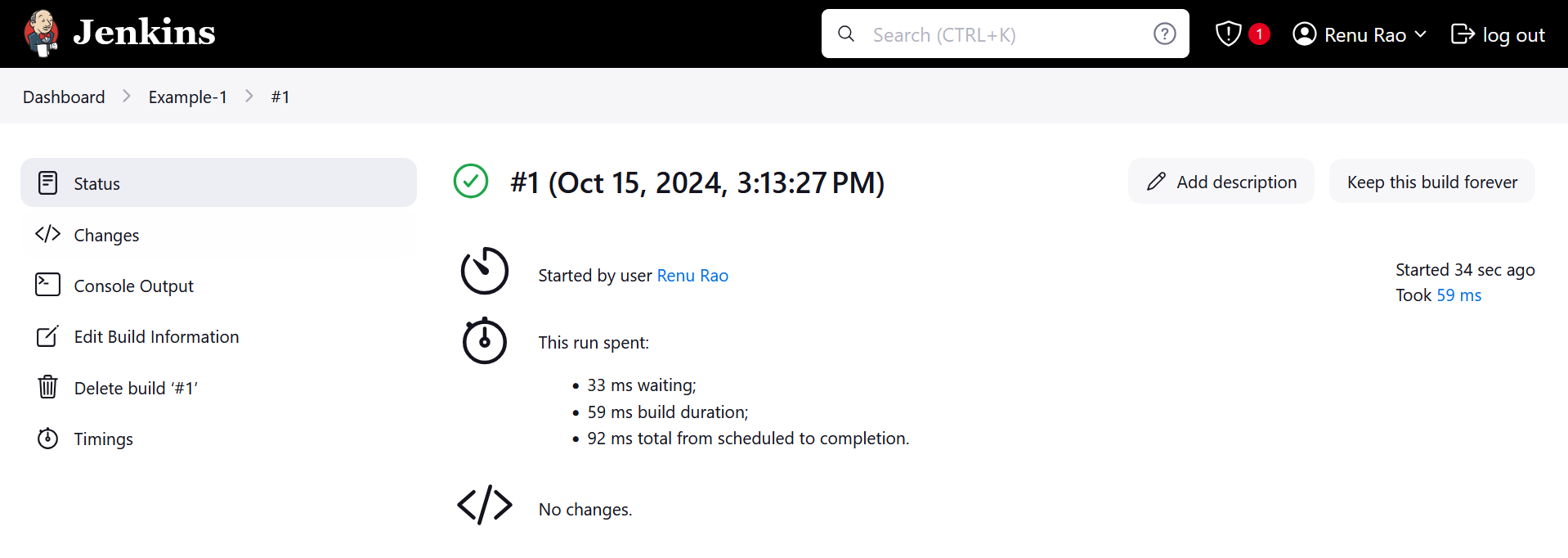
1. Click on Build Now



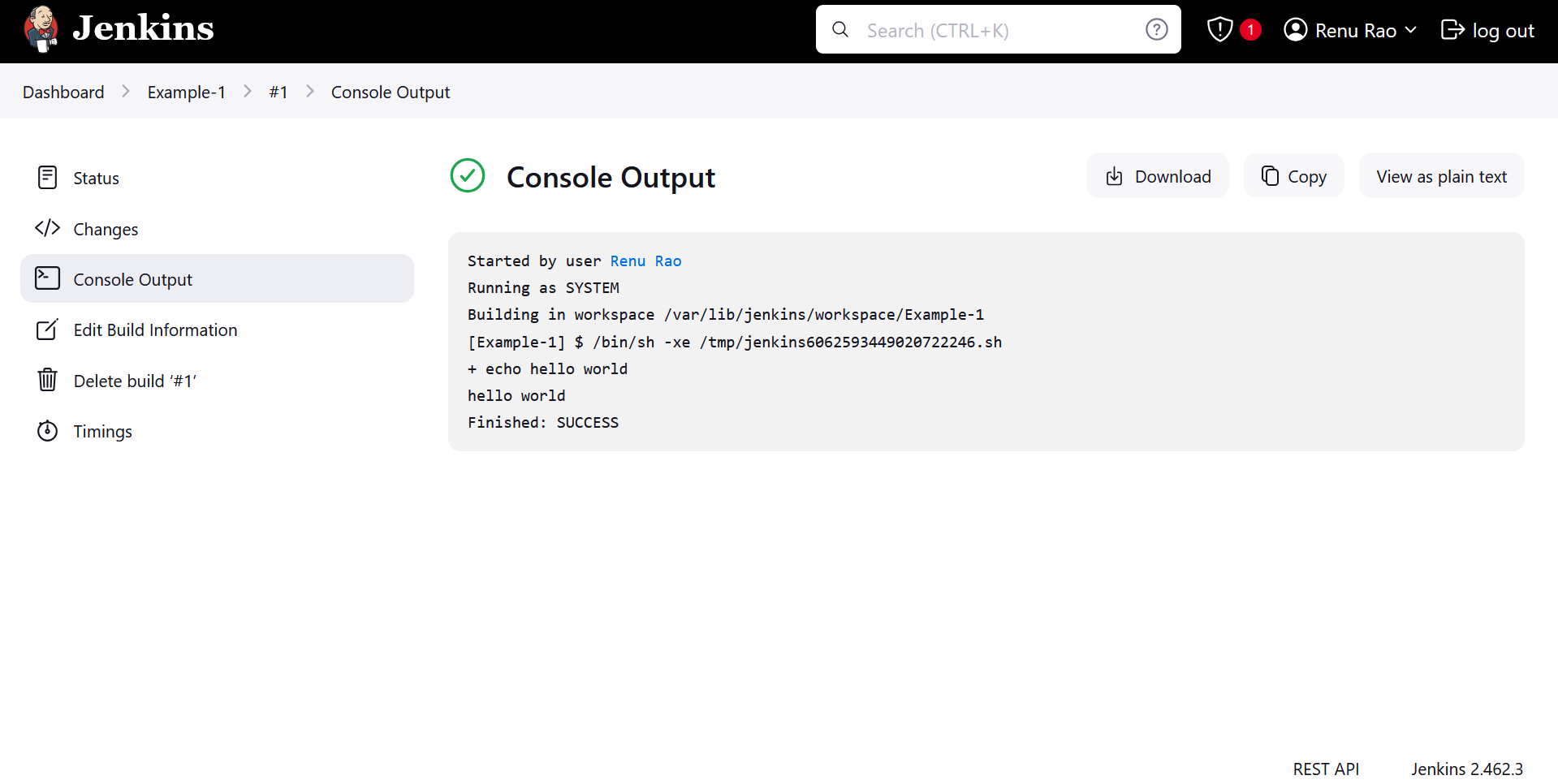
1. Click on #1



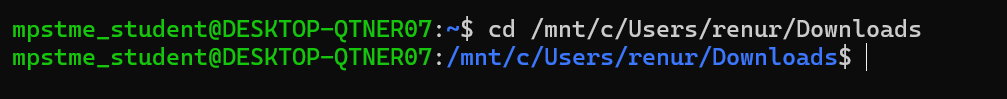
1. Click on Console Output



1. This is the output



1. **Parameterizing a Build in Jenkins**
2. Mount C:/Users/Downloads on WSL Linux



1. Create a bash file: nano example.sh and copy this code:

#!/bin/bash

if [ "$#" -eq 0 ]; then

echo "No parameters provided."

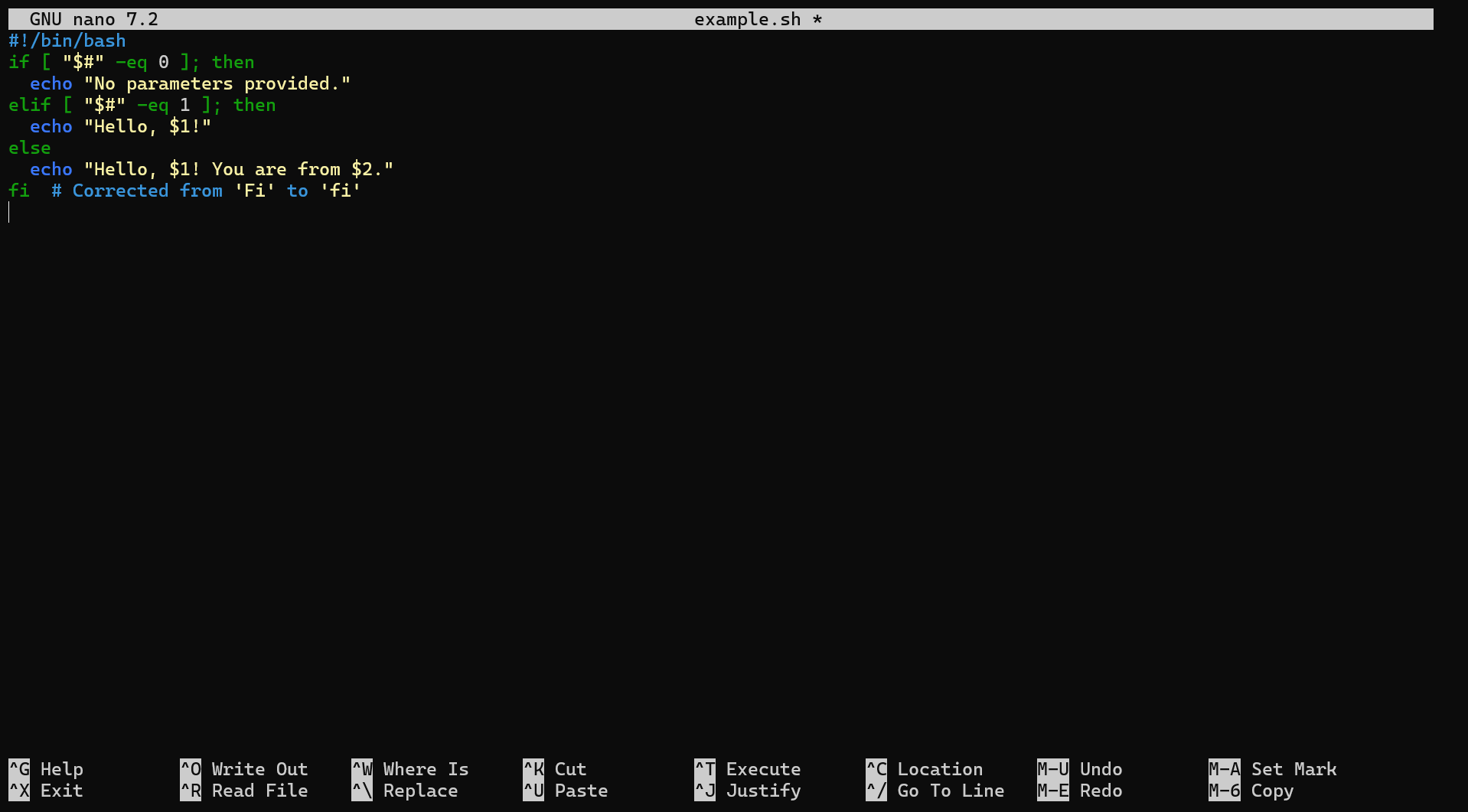
elif [ "$#" -eq 1 ]; then

echo "Hello, $1!"

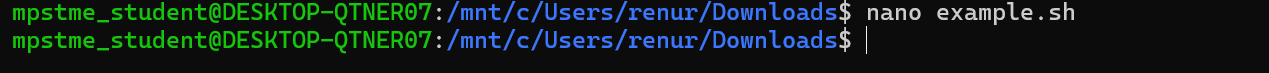
else

echo "Hello, $1! You are from $2."

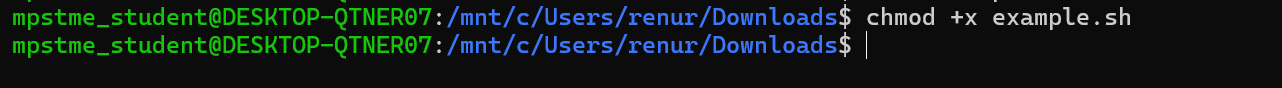
fi # Corrected from 'Fi' to 'fi'



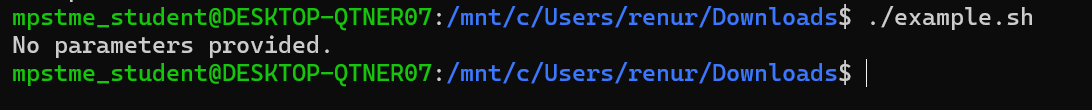
1. Ctrl-X, Y, Enter to save and exit



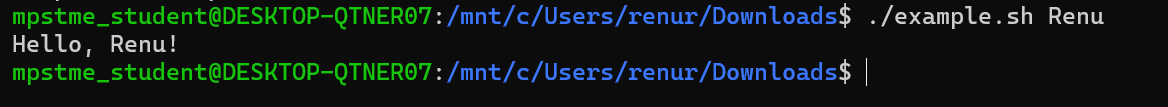
1. Give permissions to execute the file: chmod +x example.sh



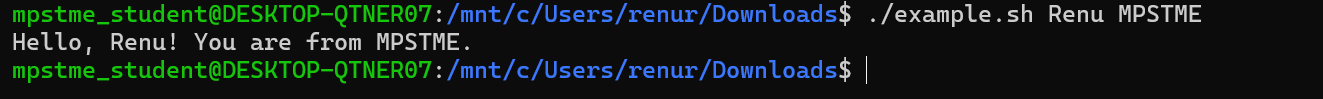
1. Run without any parameter:



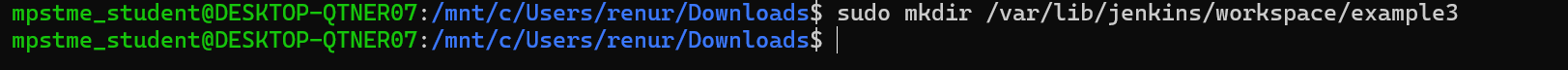
1. Run with 1 parameter:



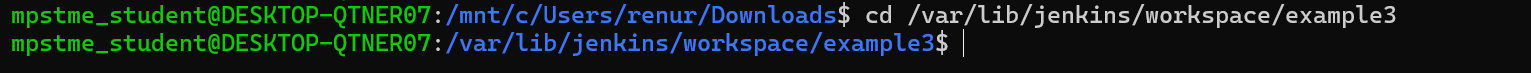
1. Run with 2 parameters:



1. The abv is just locally adding parameters. Now we use Jenkins to give parameters dynamically.
2. Create a folder



1. Change the directory

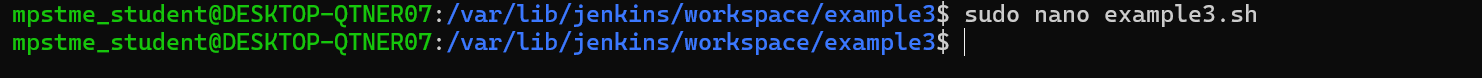


1. Create a file example3.sh

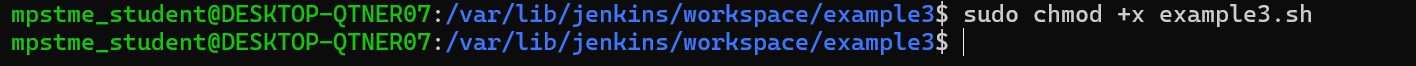
#!/bin/bash

echo "Hello, $1 from $2!"

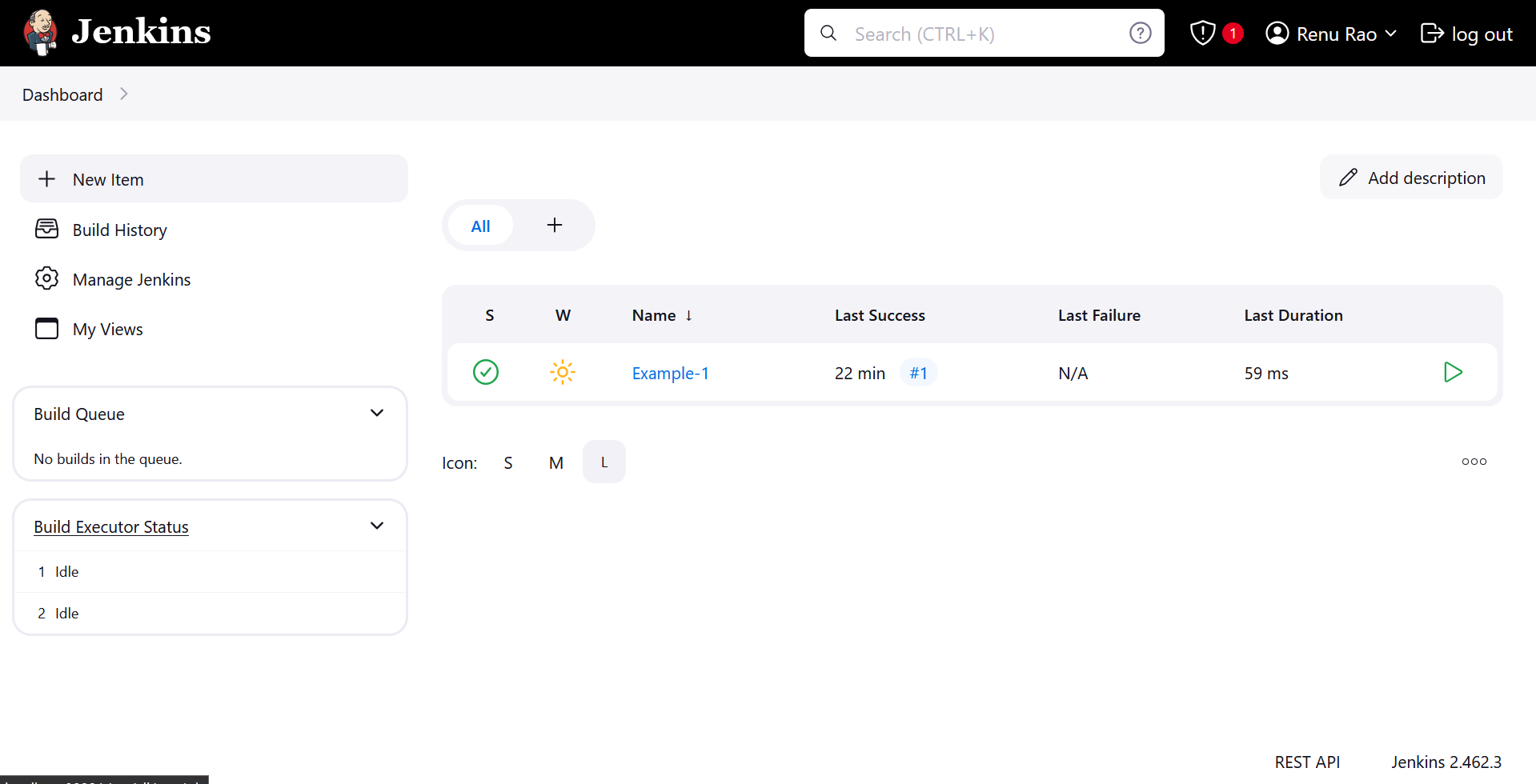


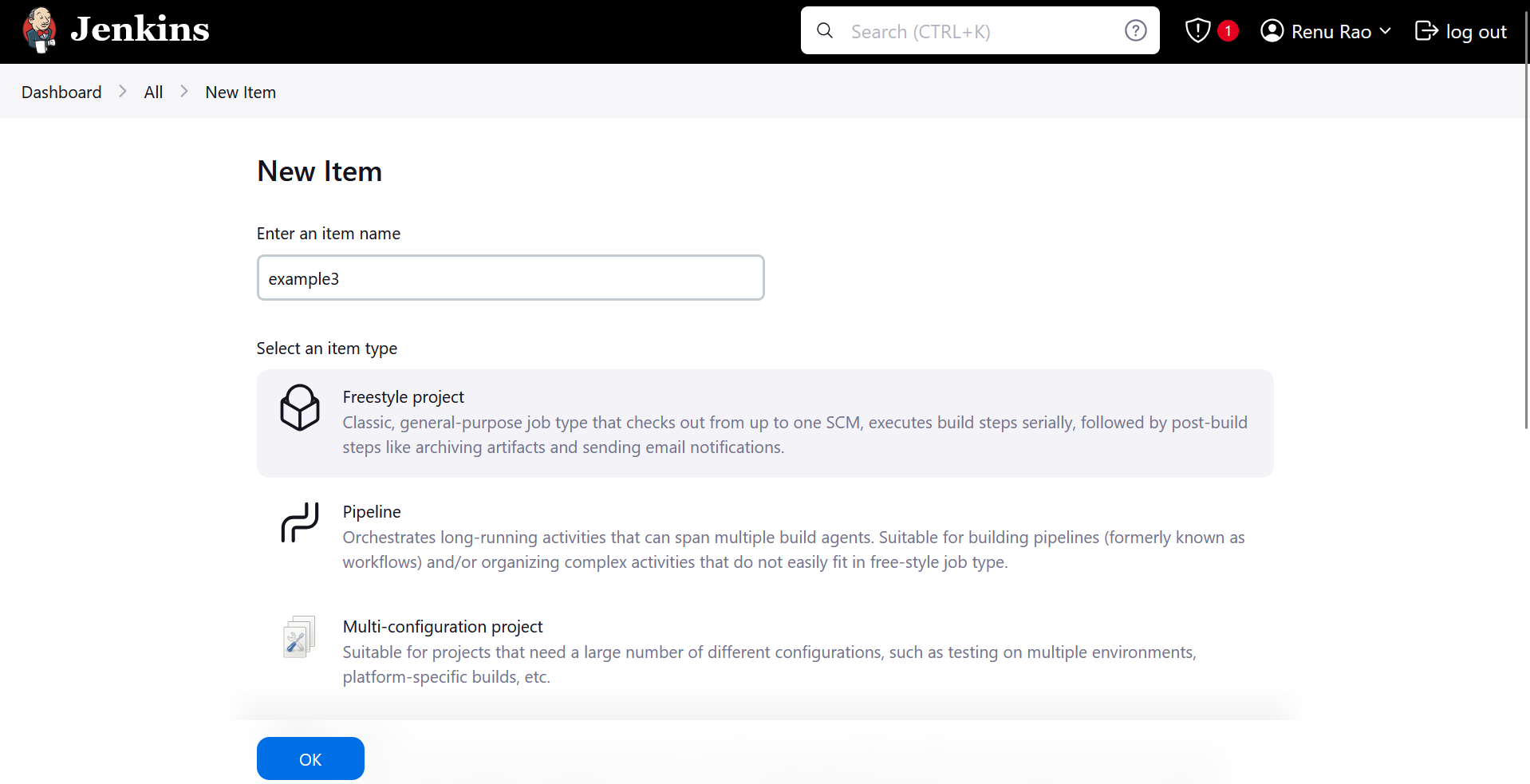


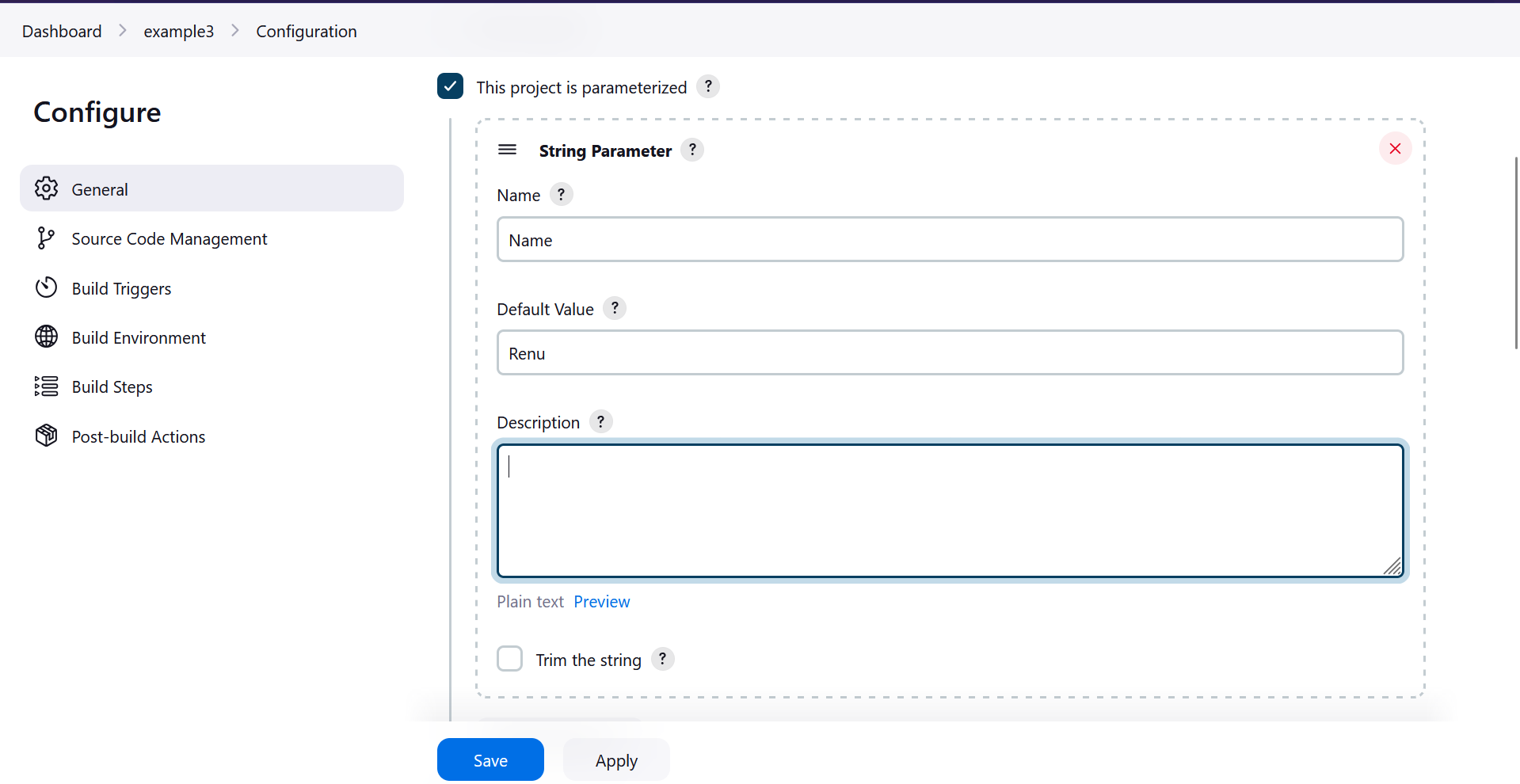
1. Give permissions to execute:

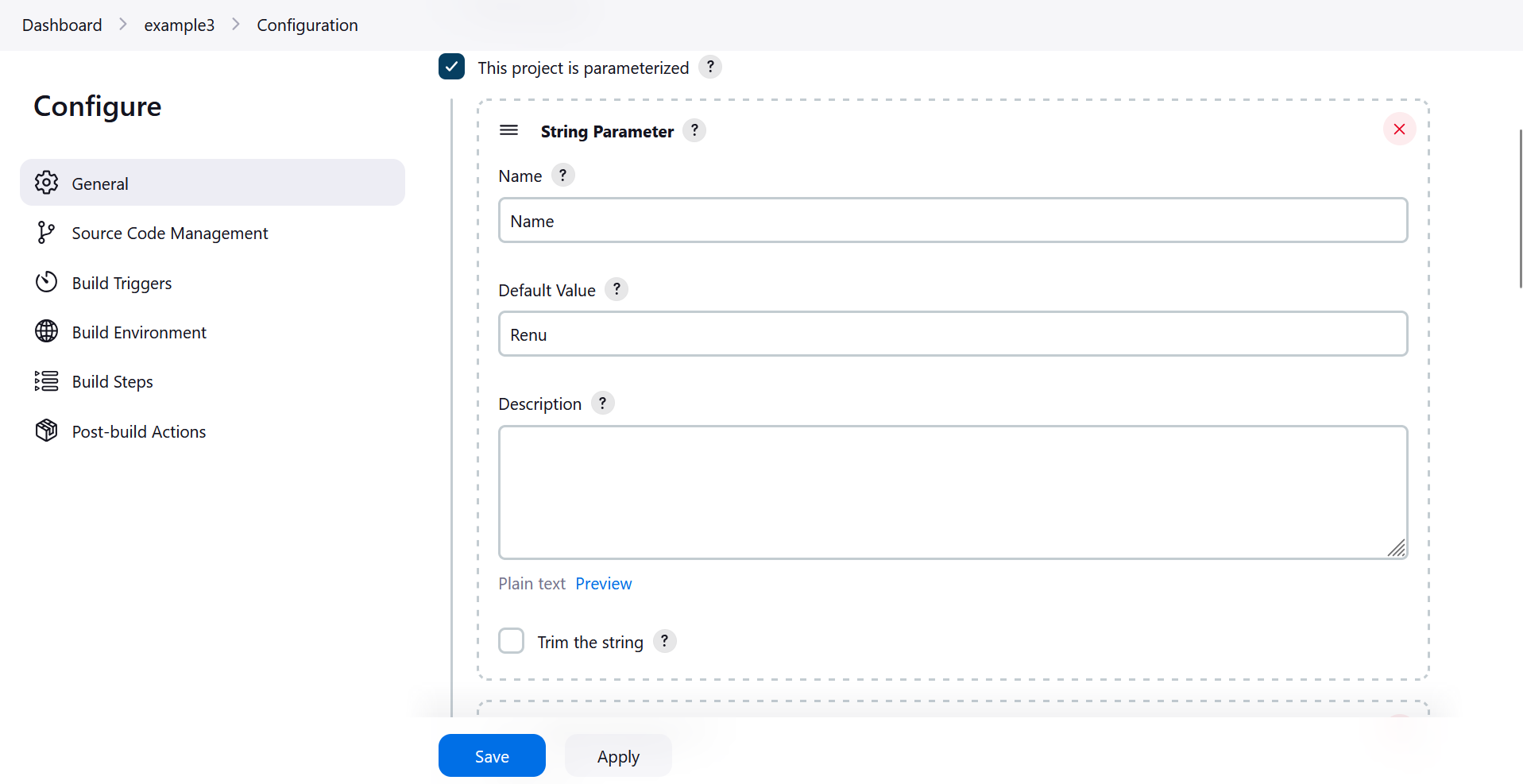


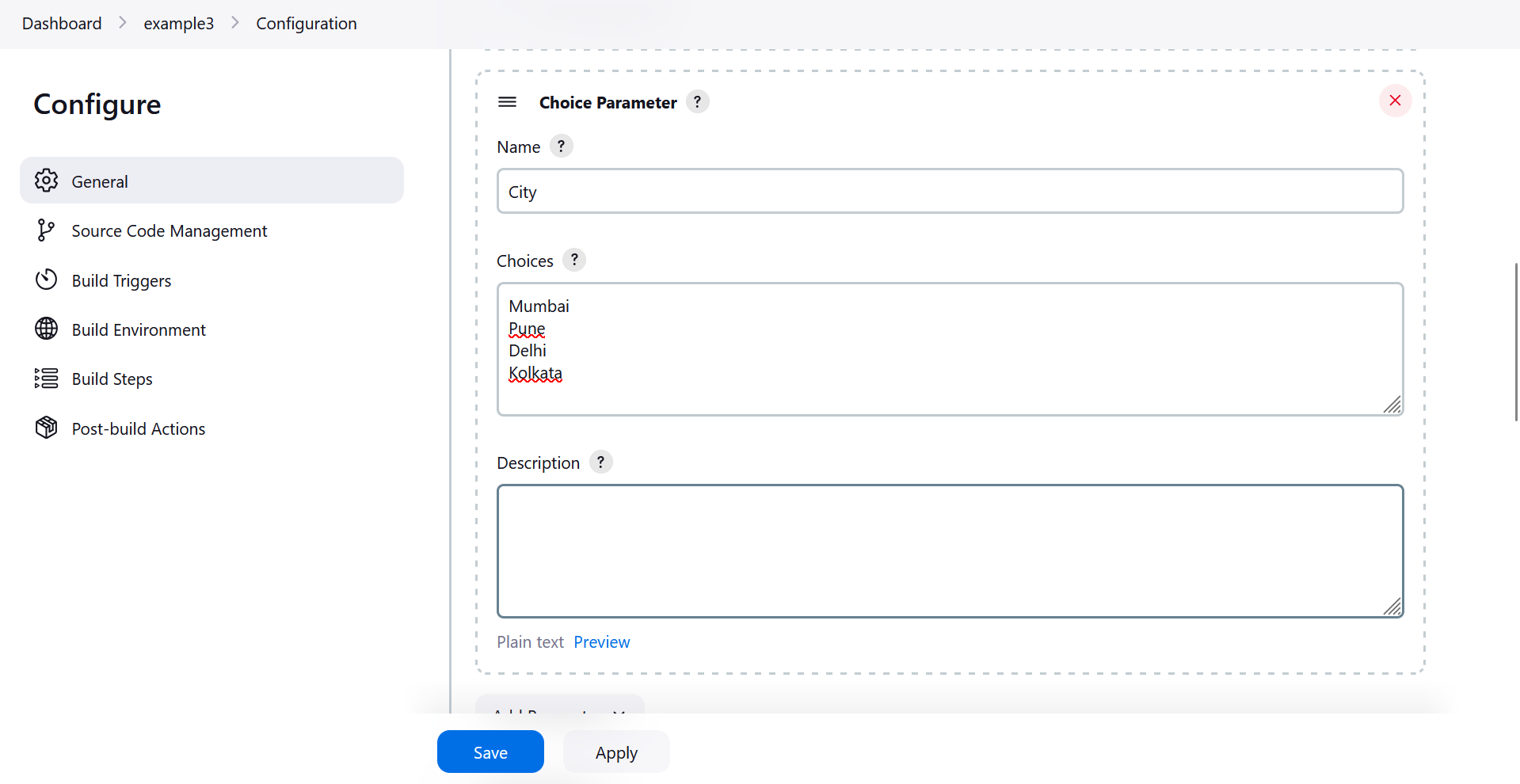
1. Create a example3.sh inside Downloads on Windows. Copy the same commands as abv
2. Create a new item in Jenkins

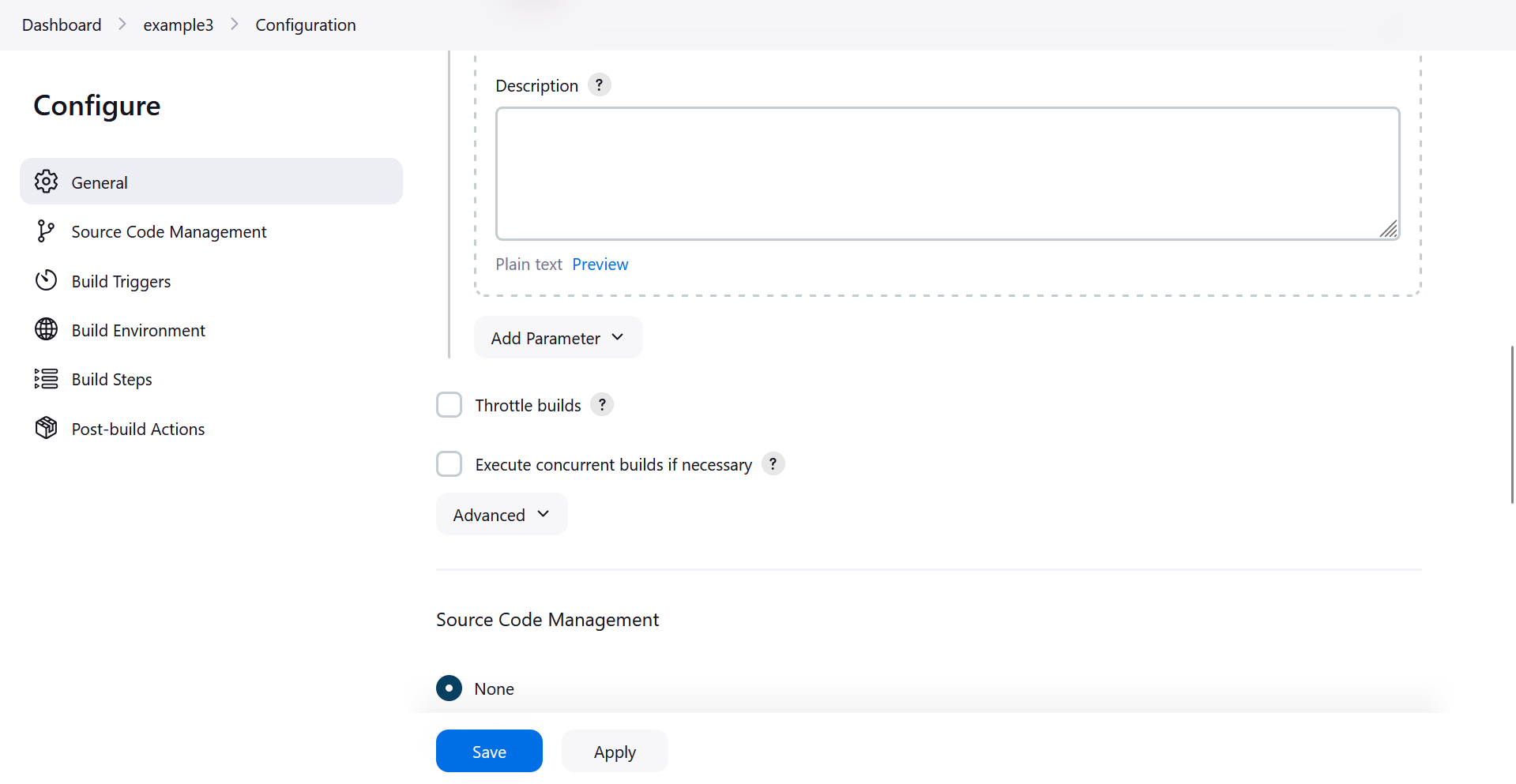


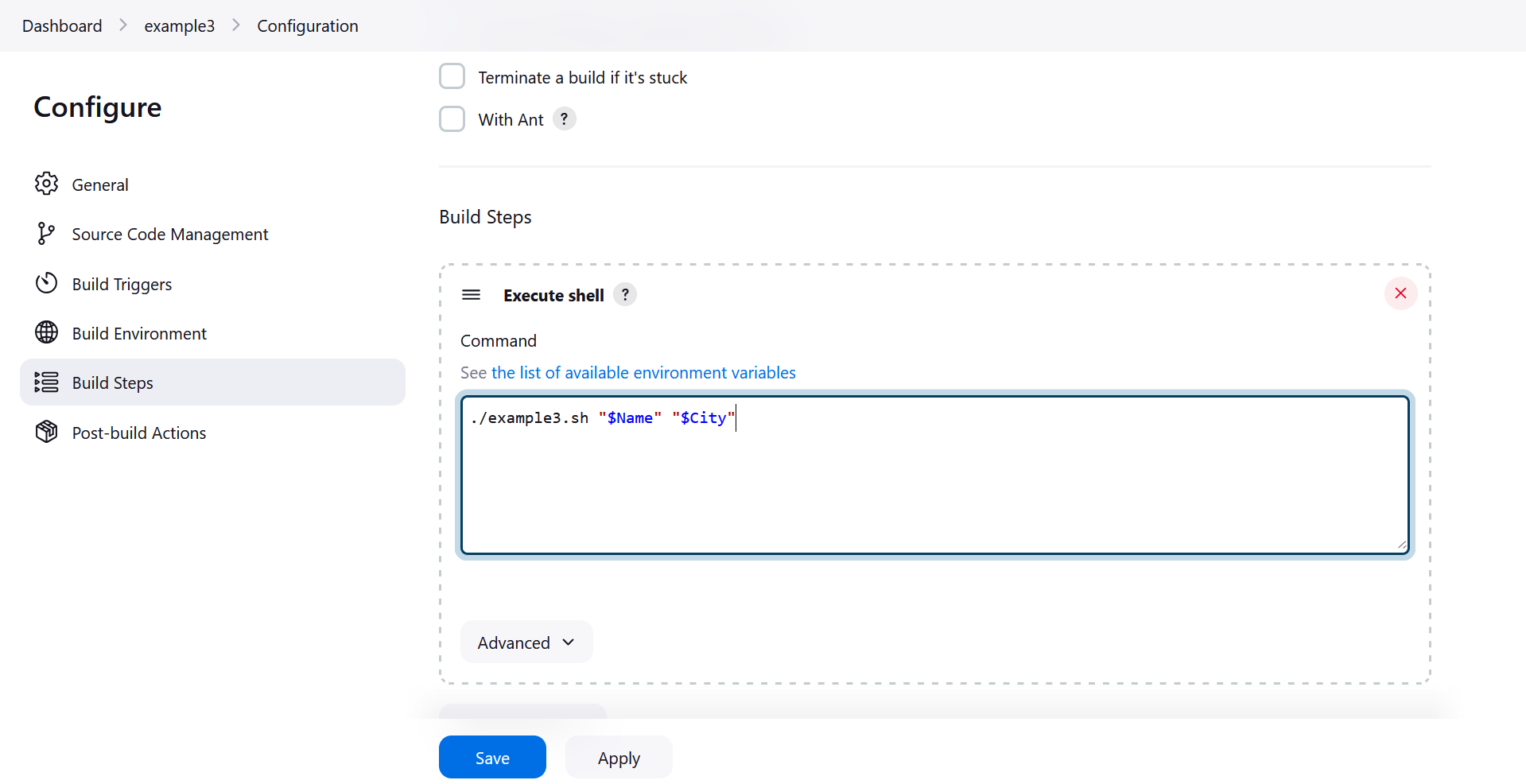




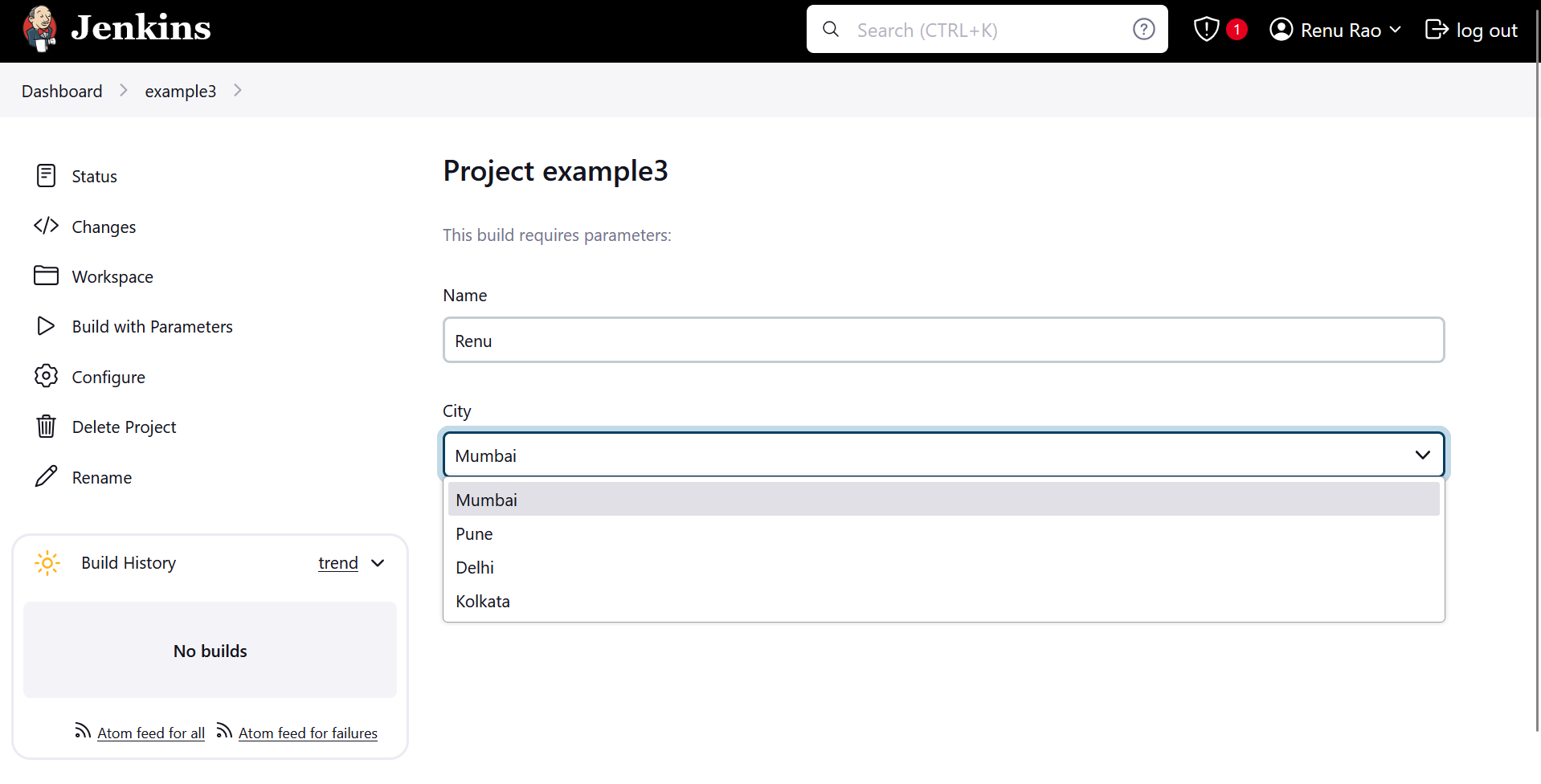




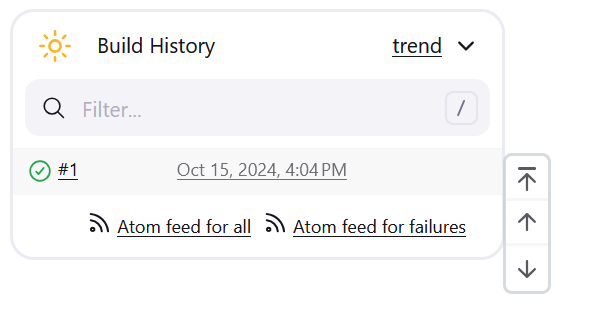




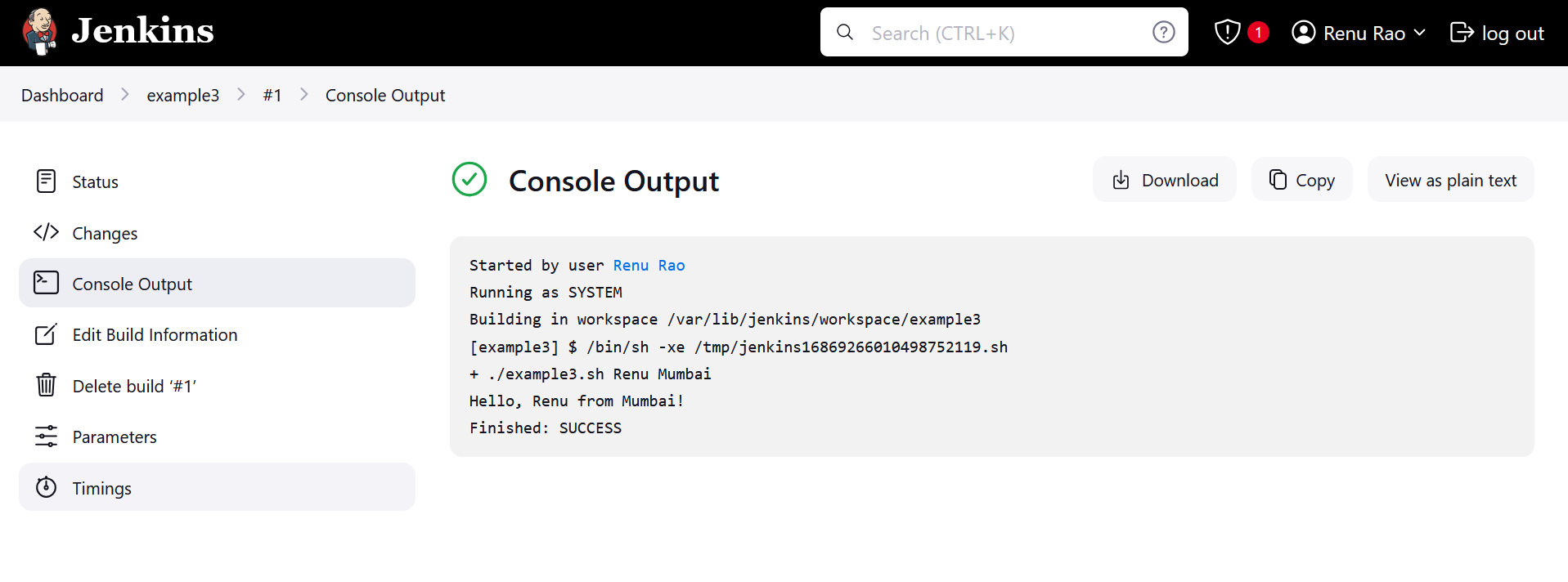
1. Apply and Save. Go to Build with Parameters



1. Click Build. Click on #1

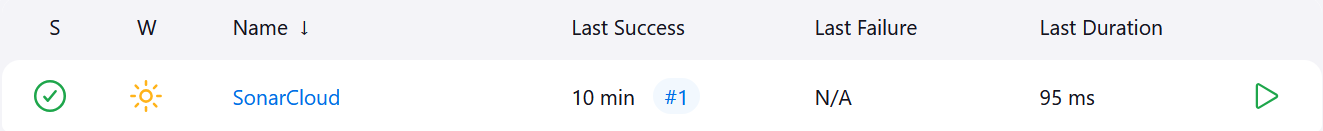


1. Click on Console Output



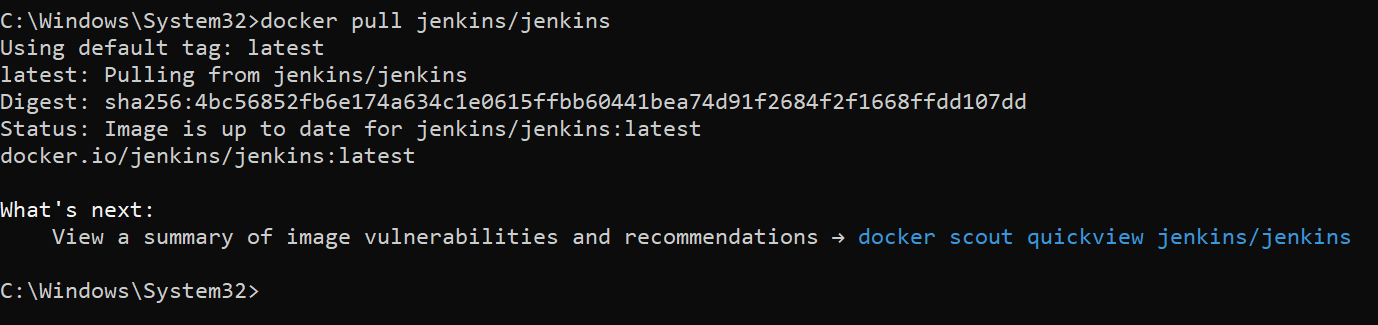
Temp ss





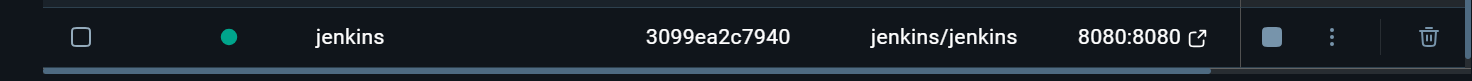
Configure Jenkins to deploy a Docker container running Nginx to a local Docker host. The pipeline should build the Docker image and start the container.

First pull Jenkins docker image: docker pull jenkins/jenkins



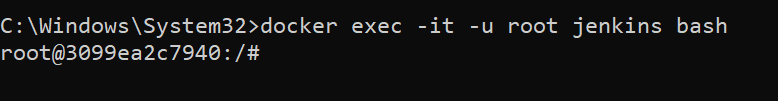
Then we start the Jenkins container:

docker run -d --name jenkins -v jenkins\_home:/var/jenkins\_home -p 8080:8080 jenkins/Jenkins



Enter the Jenkins container:

docker exec -it -u root jenkins bash

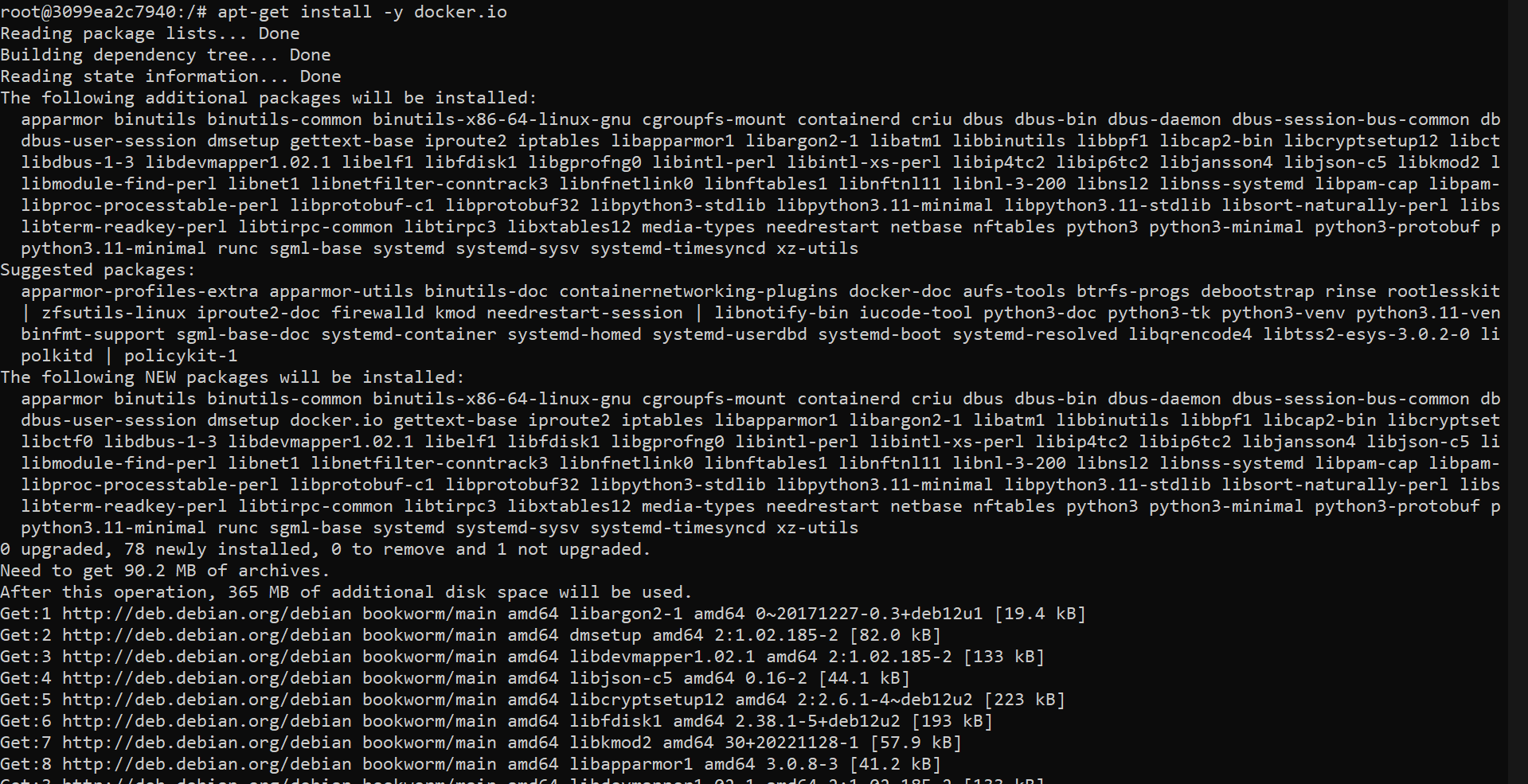


Then install Docker inside the Jenkins container:

apt-get update

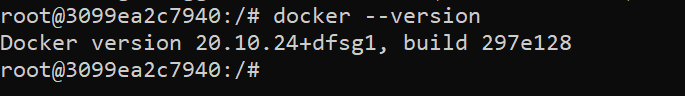


apt-get install -y docker.io



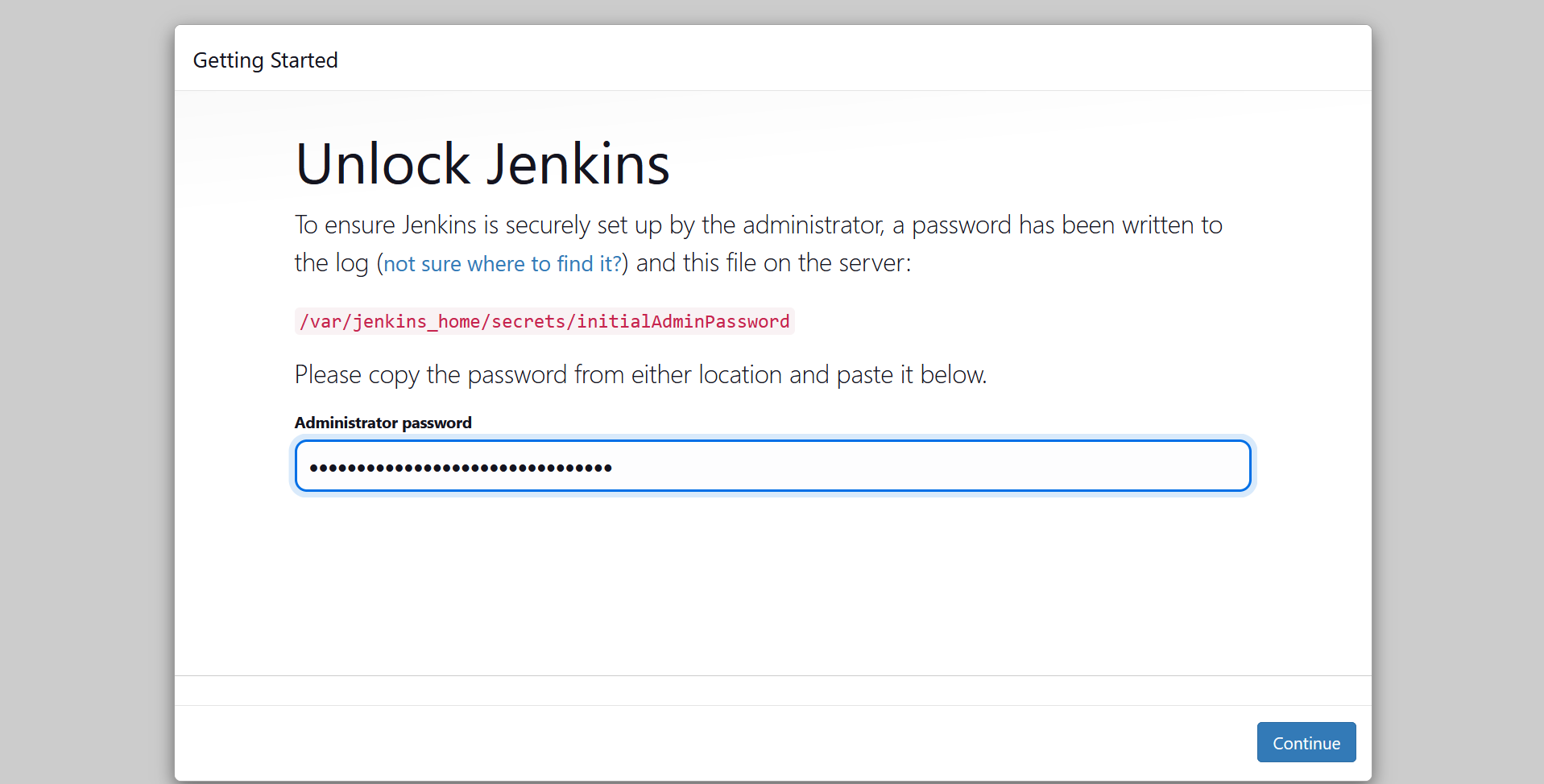
Check if docker was successfully installed in the Jenkins container:

docker –version

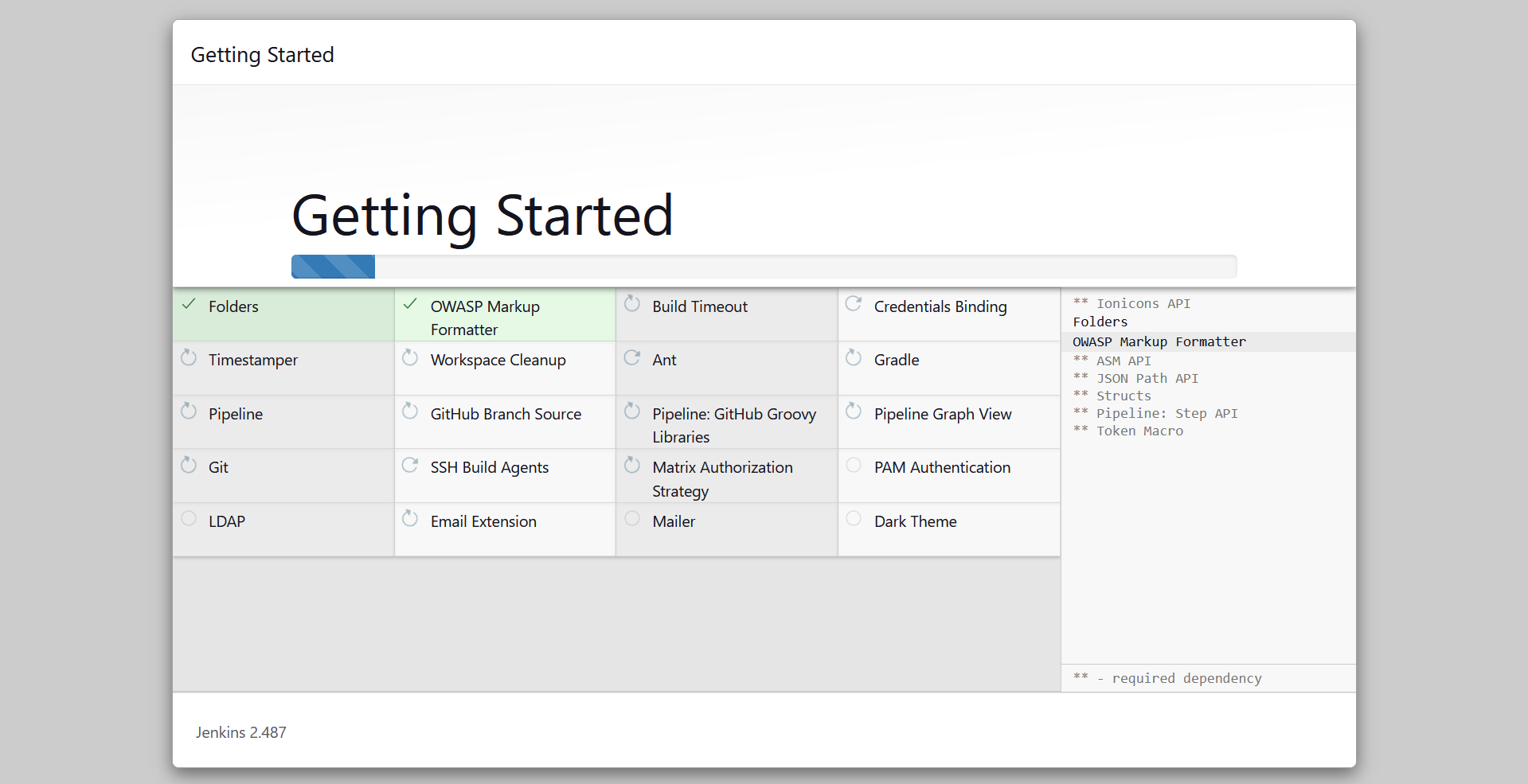


Then open localhost:8080. You can get the password by check the logs by clicking on the container in Docker Desktop

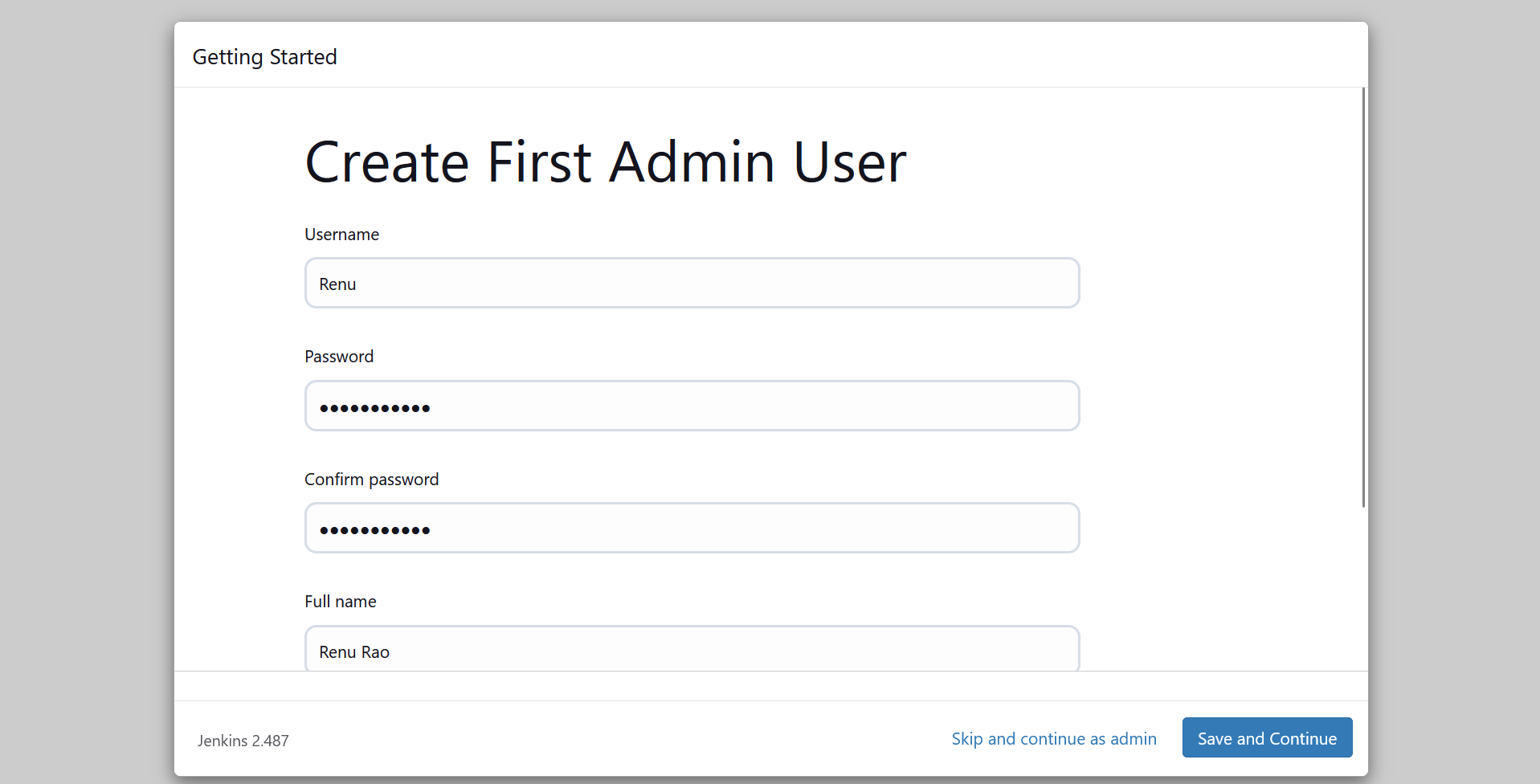




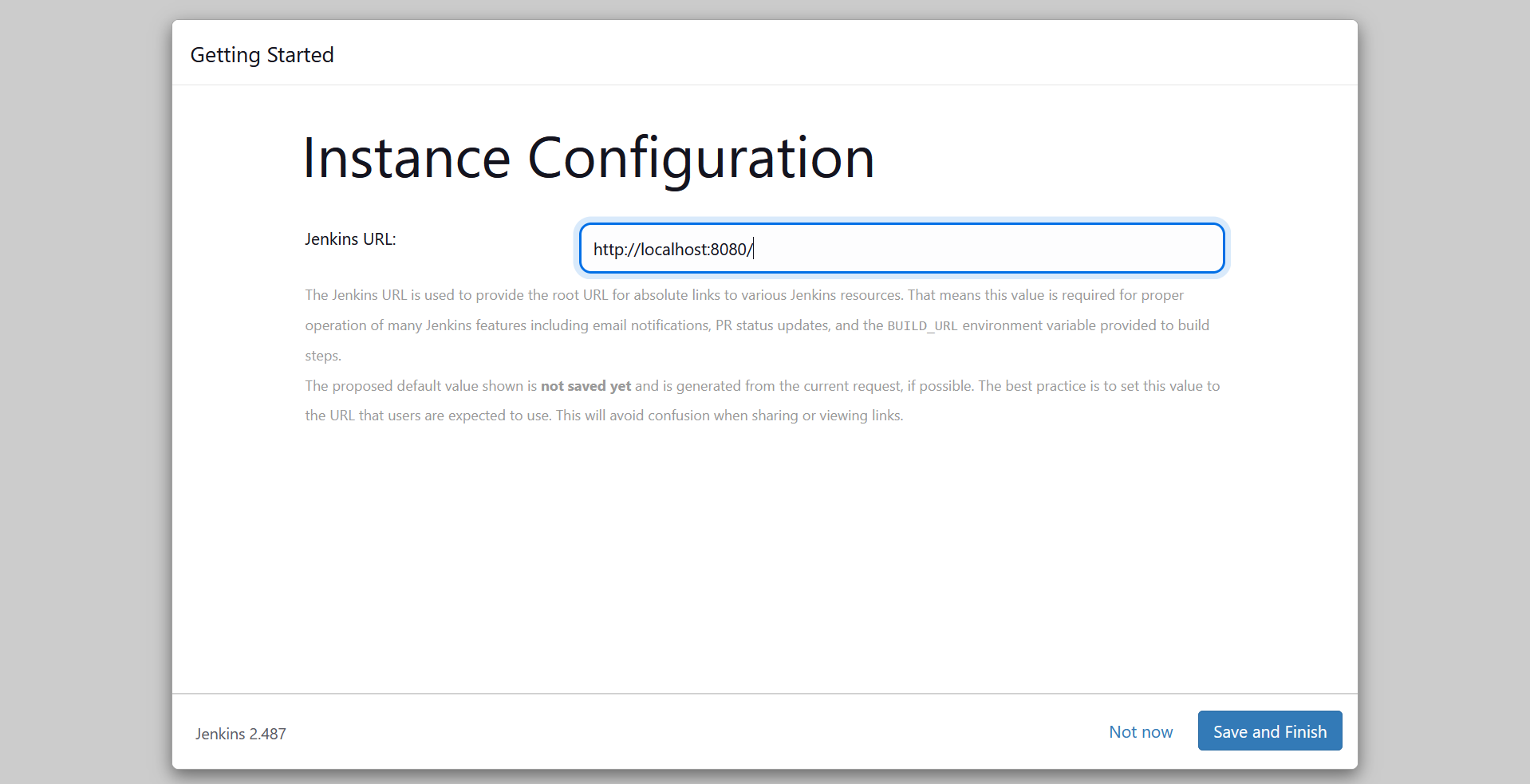
It might ask you to install plugins. Select install suggested plugins



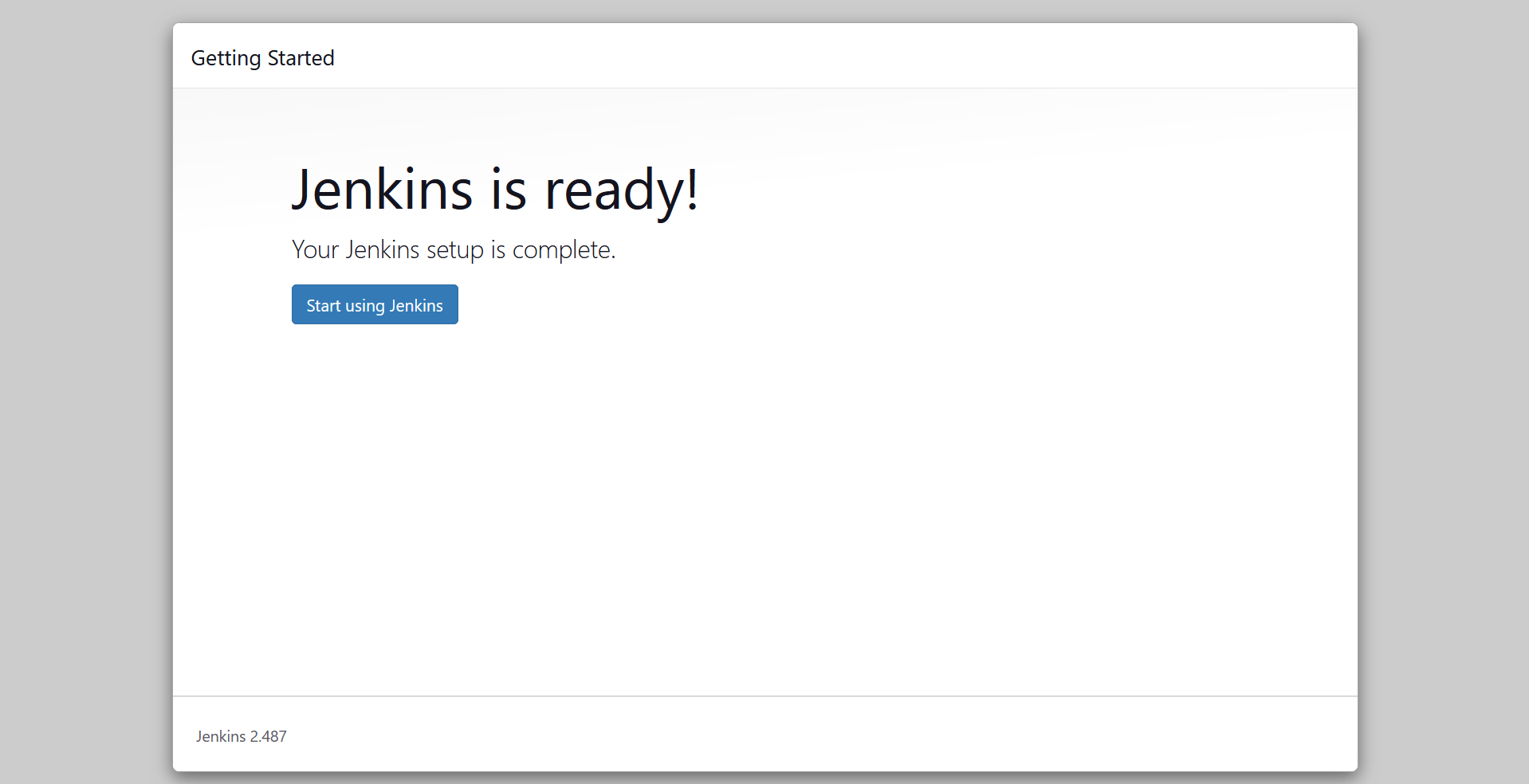
Once that is done, it might ask you to create a new user. Enter your credentials



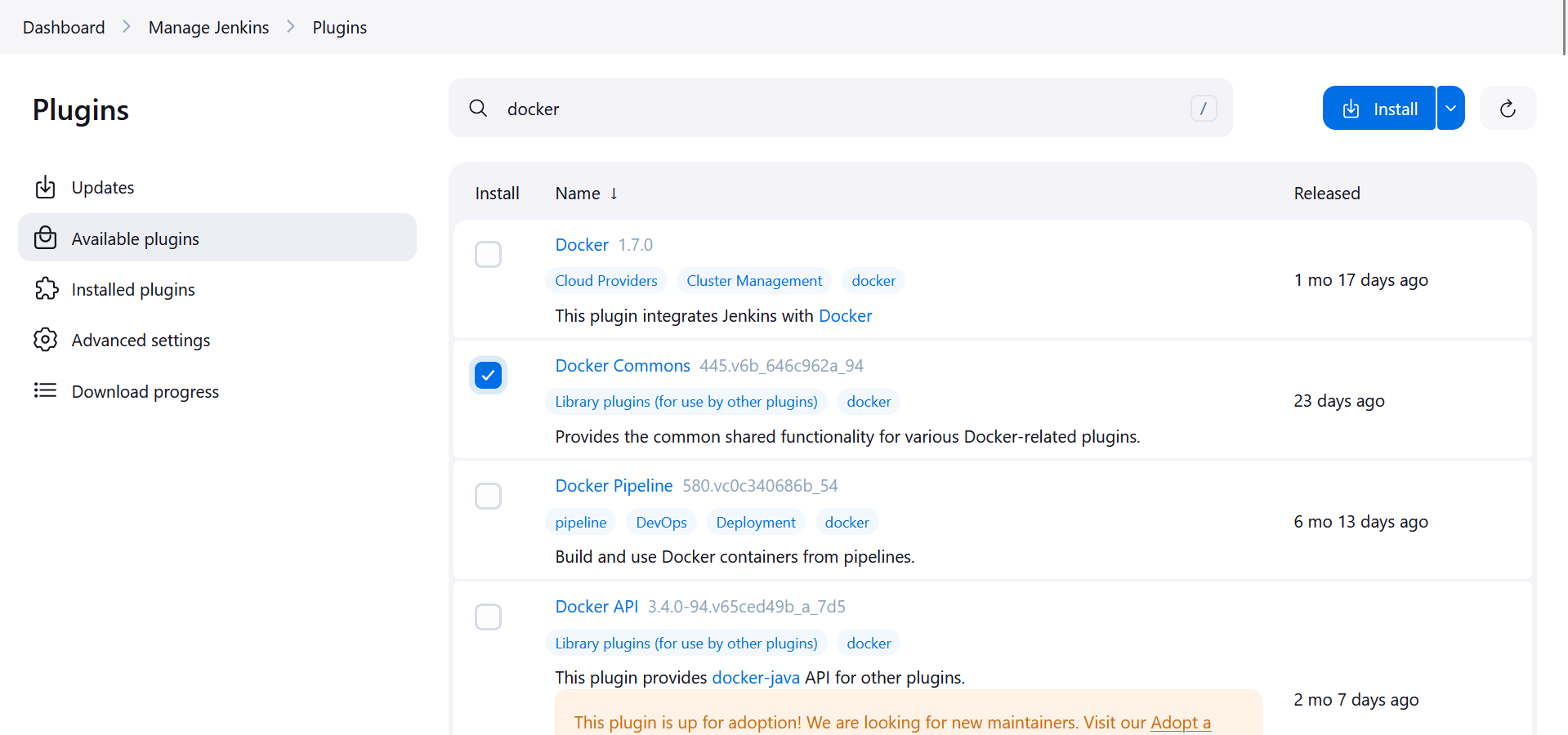
Click Save and finsh

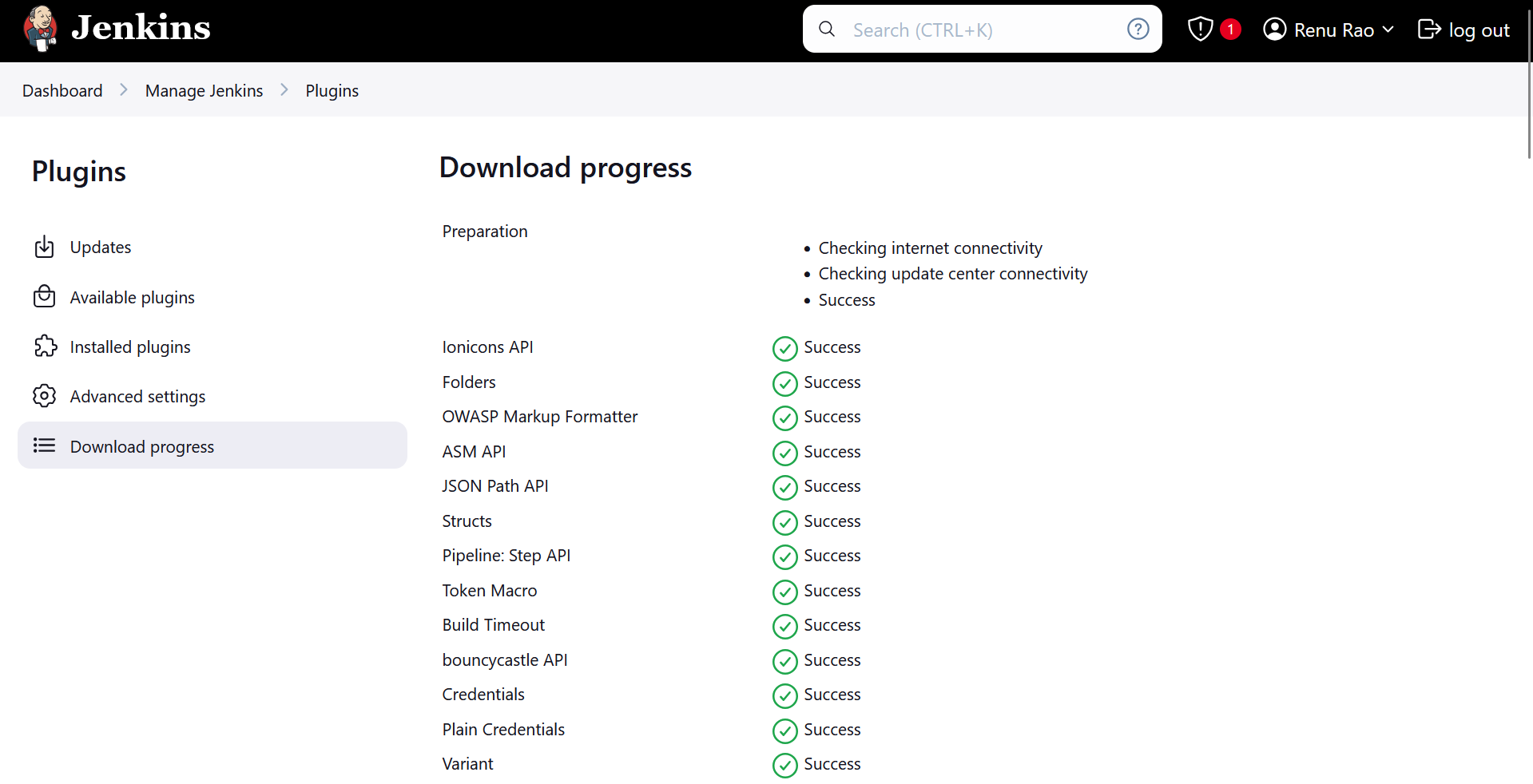


Your Jenkins setup is ready

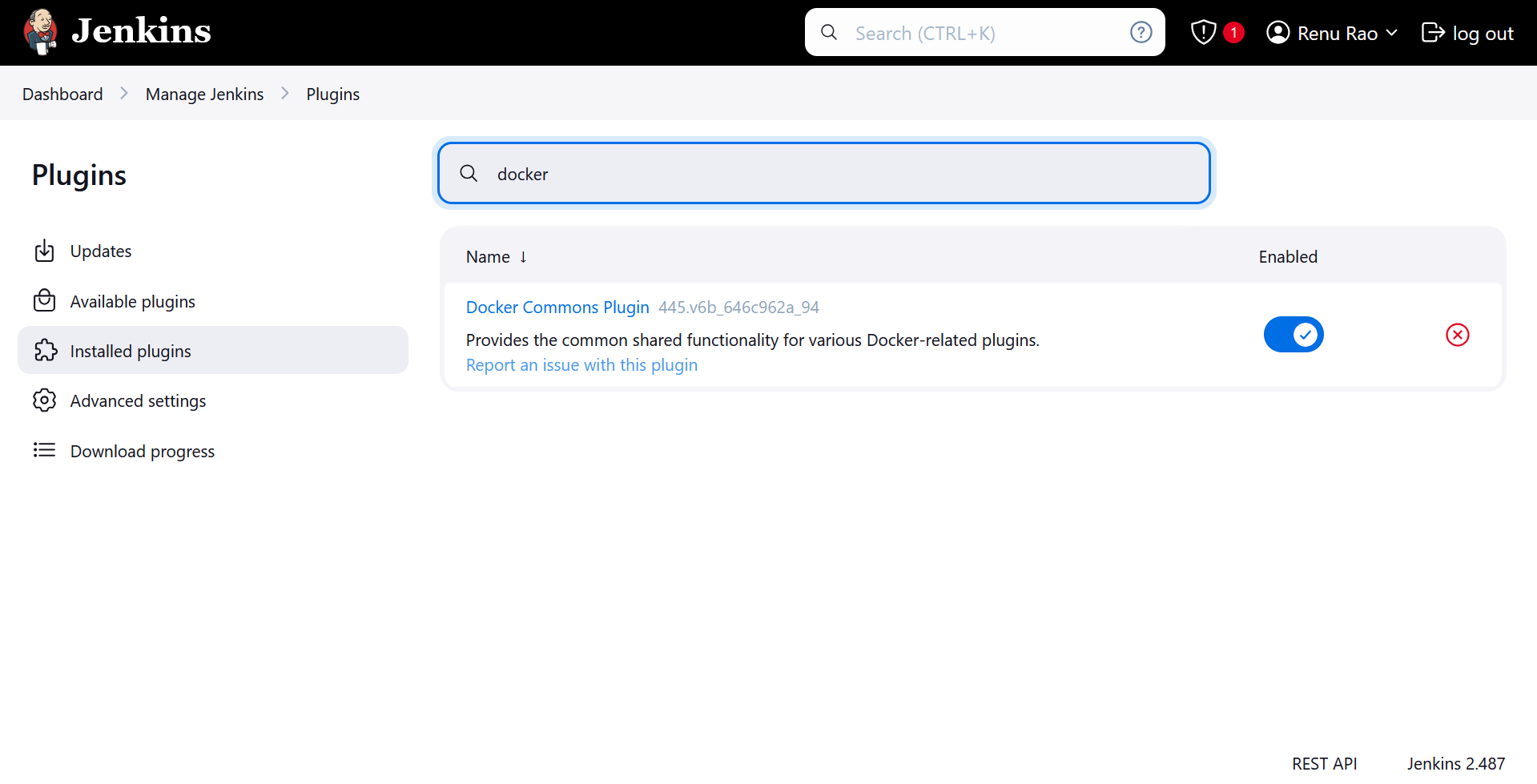


Once you see you Jenkins dashboard -> go to manage plugins -> available plugins -> Search for docker -> Install the Docker Commons Plugin





After the installation of the plugin -> Refresh and sign-in again -> Go to Manage Jenkins -> Installed plugins -> Search of docker and Check if installed successfully

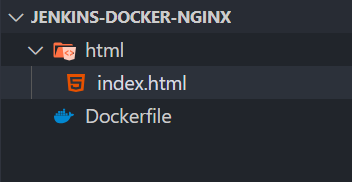


Create a New Folder named Jenkins-Docker-Nginx



Open this in VS code->Dockerfile should be in Jenkins-Docker-Ngnix

Inside The Jenkins-Docker-Ngnix->Create a html folder-> Create your index.html inside the html folder:



Inside the Dockerfile add this content:

*# Use the official Nginx image from the Docker Hub*

FROM nginx:latest

*# Copy custom Nginx configuration file (optional)*

*# COPY nginx.conf /etc/nginx/nginx.conf*

*# Copy HTML files or your web app into the container*

*# Make sure you have an 'index.html' or other web content to serve*

COPY ./html /usr/share/nginx/html

*# Expose port 80 for web traffic*

EXPOSE 80

*# Start Nginx in the foreground*

CMD ["nginx", "-g", "daemon off;"]

Add this content inside the index.html file:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

</head>

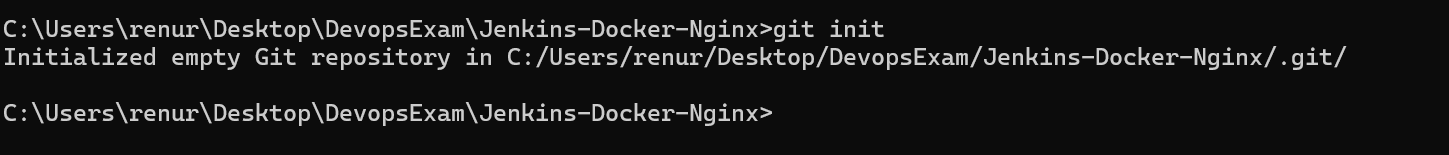
<body>

<h1>Hello World</h1>

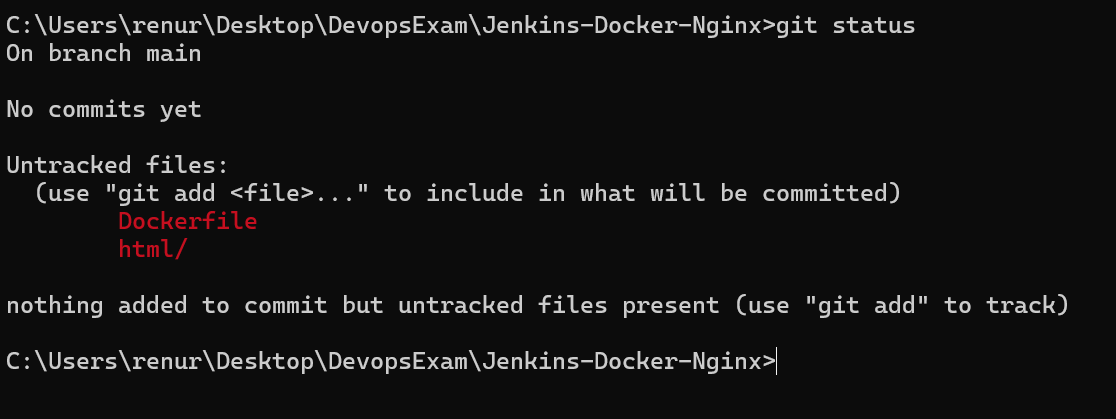
</body>

</html>

Now we need to push this project to github. For that first initialize the repo. Navigate to Jenkins-Docker-Nginx folder in the CMD and run: git init

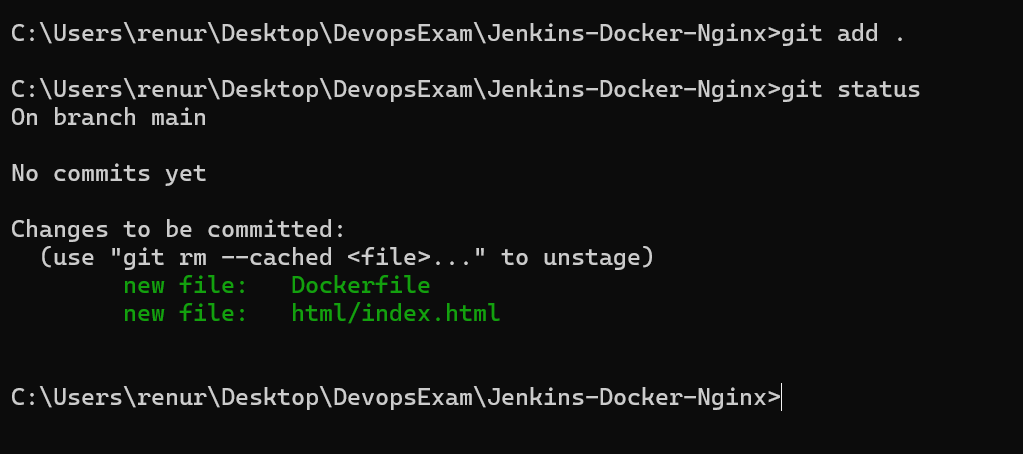


Check if git can track the uncommitted file by running: git status

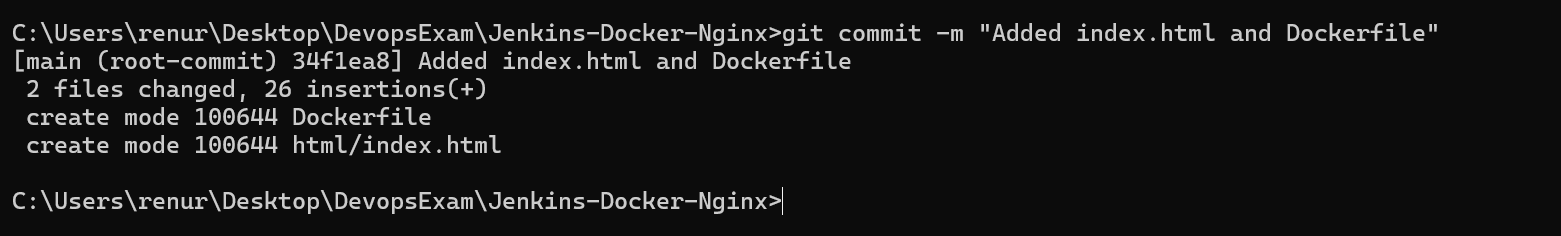


Add the files by using: git add .

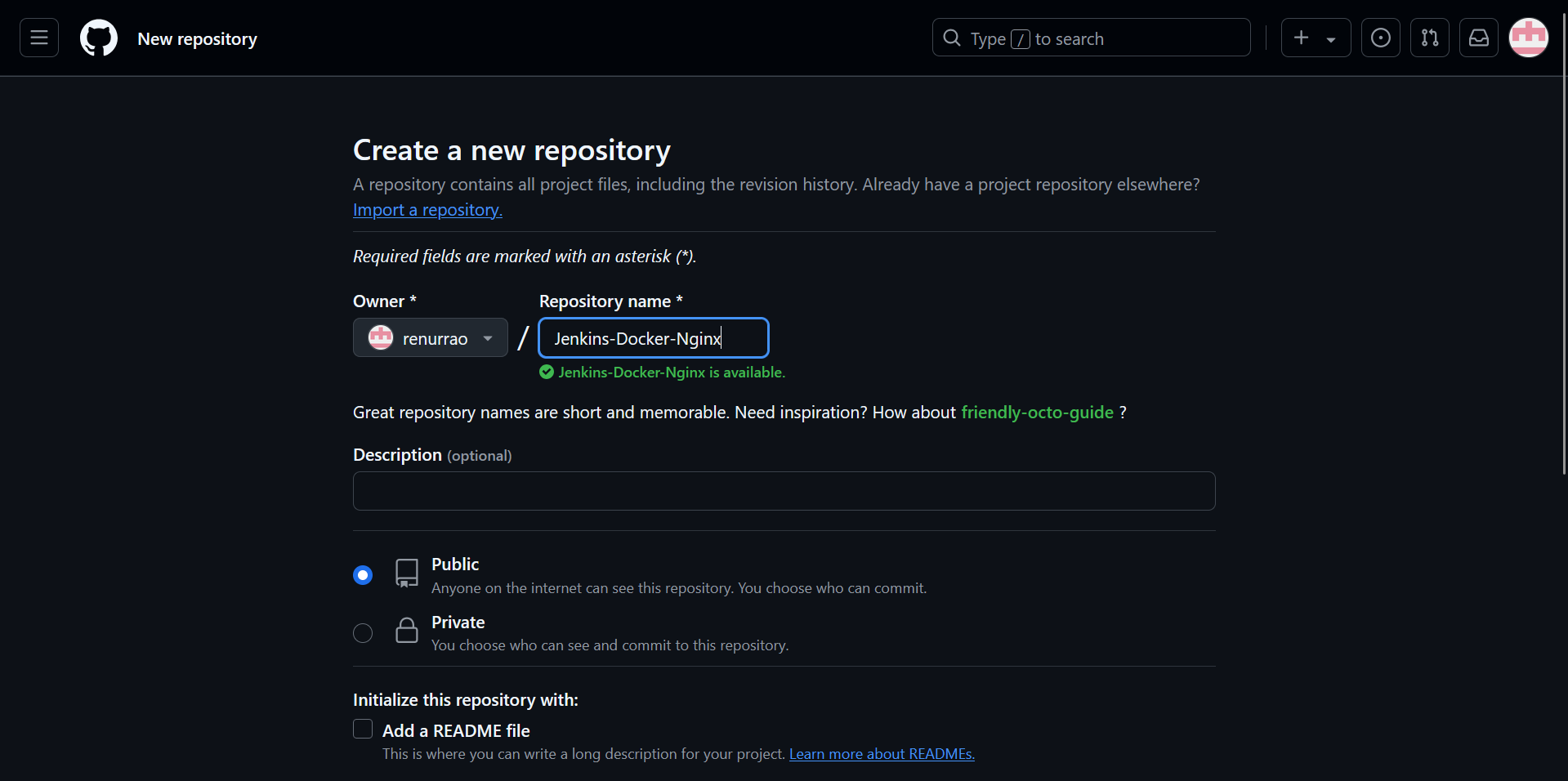
Run git status to check if the files have been added



Then make a commit: git commit -m “Added index.html and Dockerfile”



Now on Github -> Create a new repo named Jenkins-Docker-Nginx

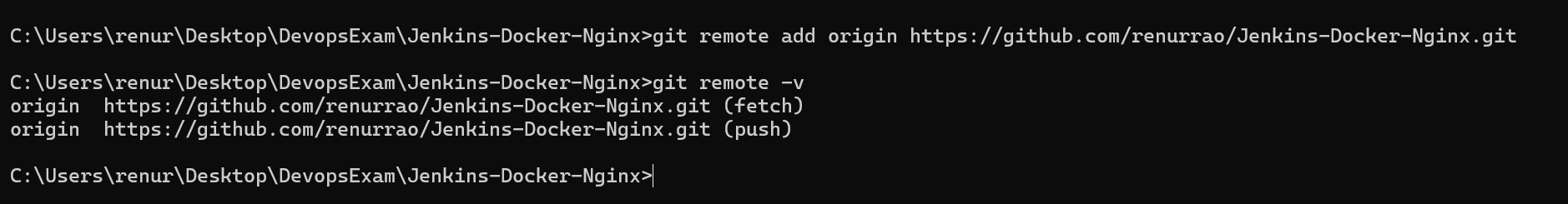


We need to add the origin of this repo to our local git:

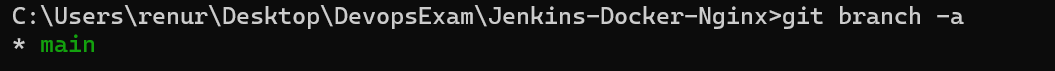
git remote add origin <https://github.com/renurrao/Jenkins-Docker-Nginx.git>

Check if has been added successfully by running:

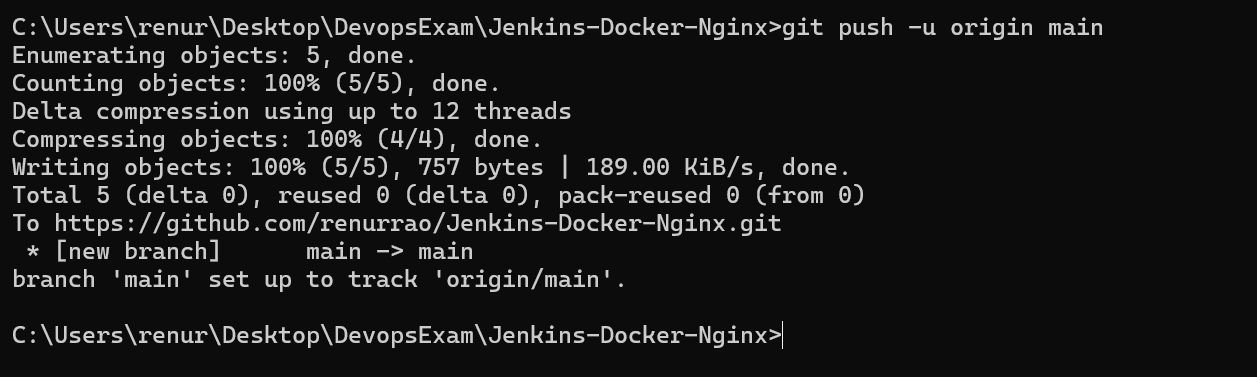
git remote -v



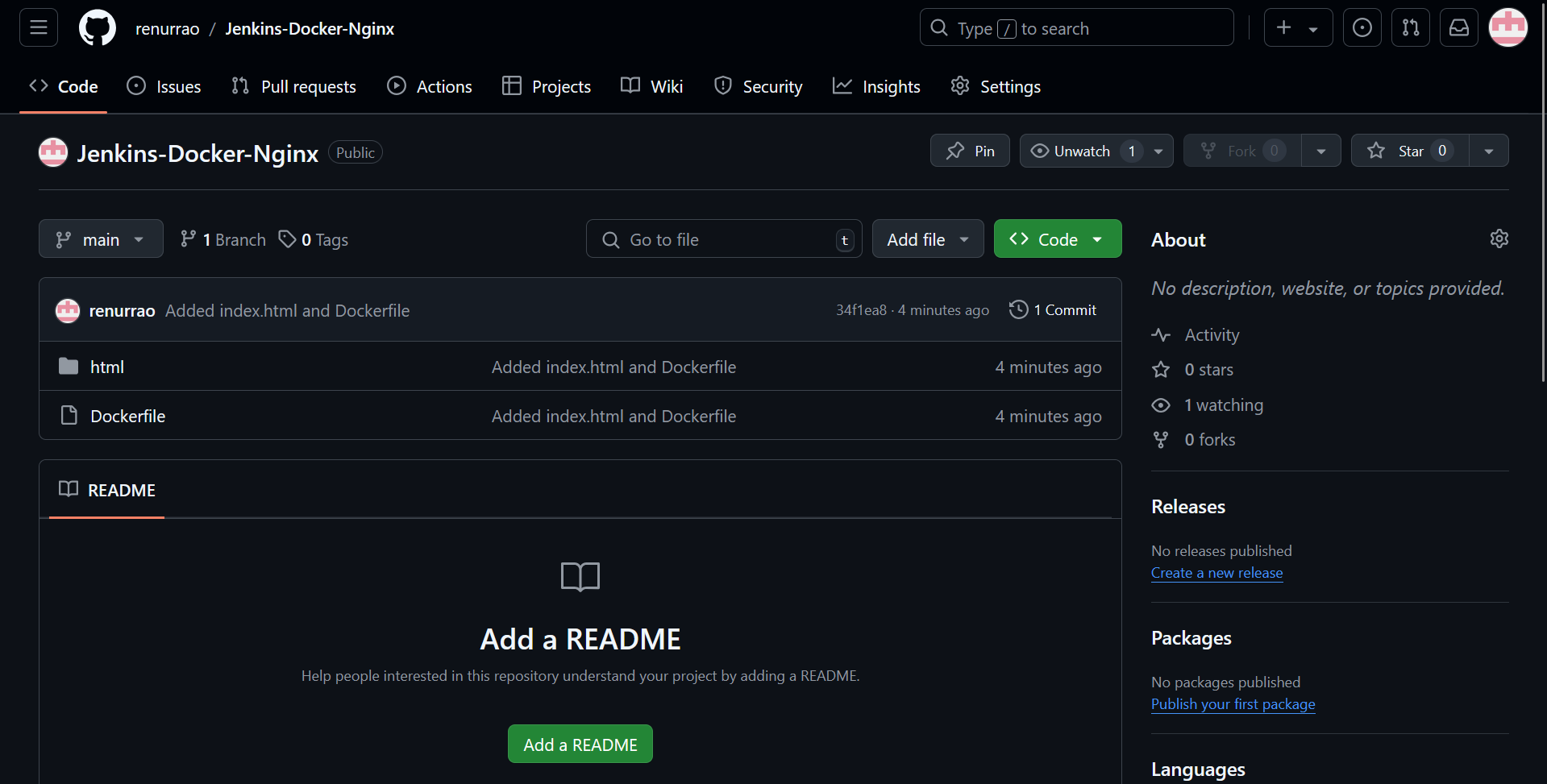
Check the branch: git branch -a



Push the commit to the remote repo on github: git push -u origin main



We can verify this by check if our files have been added to the repo on github



Now we will create a new pipeline in Jenkins.

Go to Jenkins Dashboard -> New Item -> Name it as ‘Jenkins-Docker-Nginx’ -> Pipeline Project -> Add the following code to Pipeline Script -> Click Apply and Save

pipeline {

agent any

stages {

stage('Clone Repository') {

steps {

// Example: clone your repository that contains your Dockerfile

git branch: 'main', url: 'https://github.com/renurrao/Jenkins-Docker-Nginx.git'

}

}

stage('Build Docker Image') {

steps {

script {

// Build the Docker image using the Dockerfile in your repository

sh 'docker build -t my-nginx-image .'

}

}

}

stage('Run Docker Container') {

steps {

script {

// Stop and remove any previous containers

sh 'docker ps -aq --filter "name=my-nginx-container" | xargs -r docker stop | xargs -r docker rm'

// Run the new container

sh 'docker run -d --name my-nginx-container -p 8081:80 my-nginx-image'

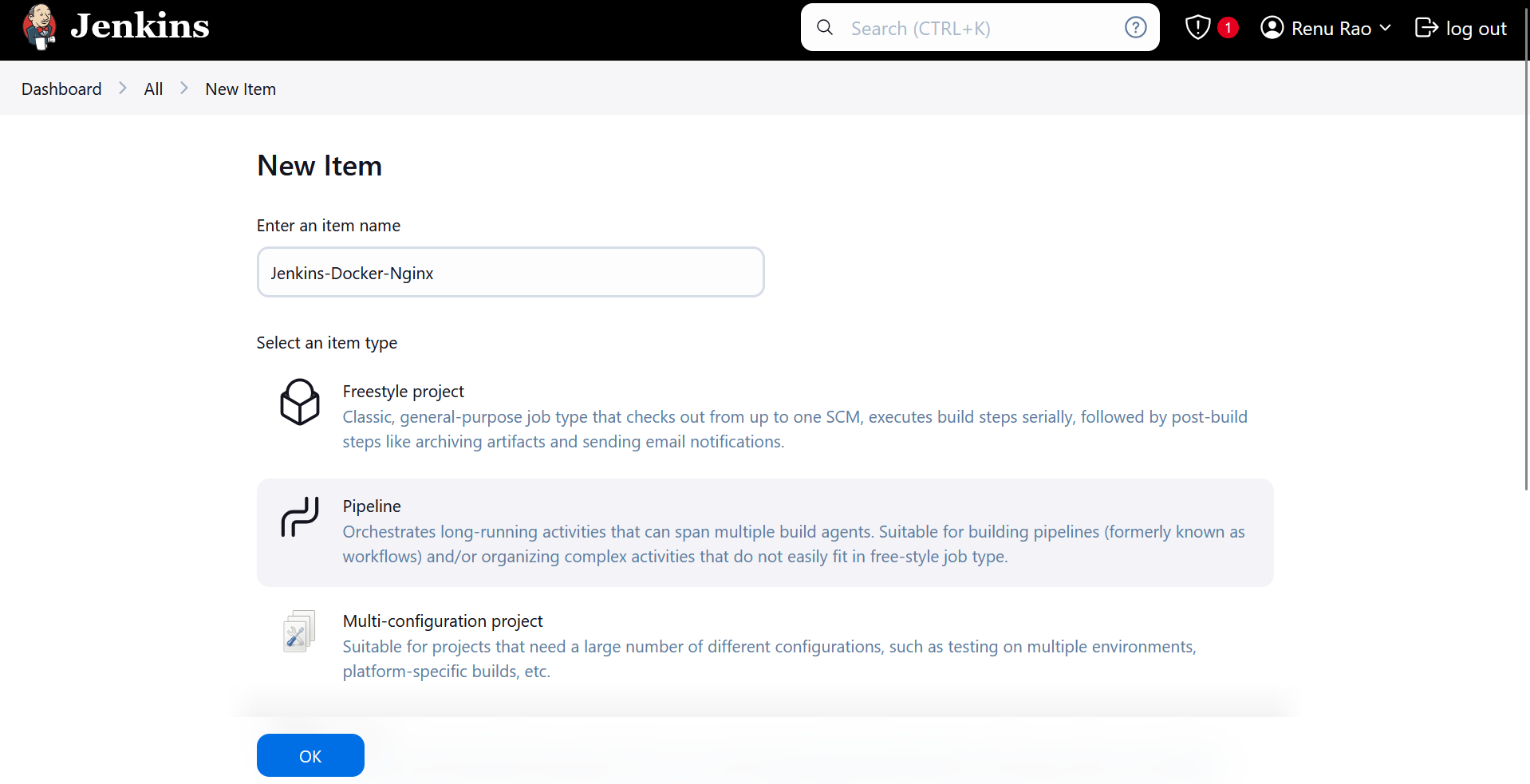
}

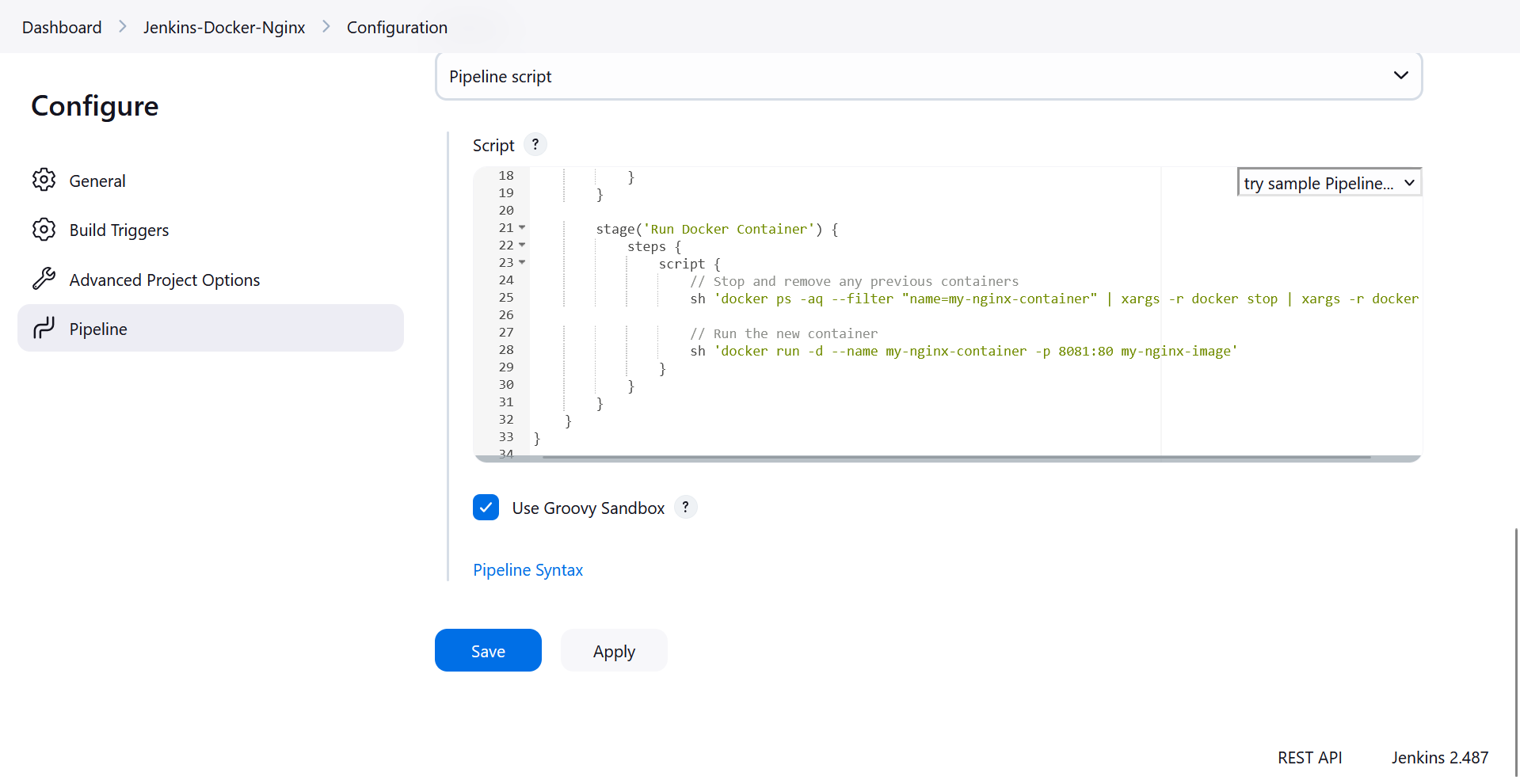
}

}

}

}





After that Click on Build Now to start the build process