Building a CI/CD Pipeline for Node.js with Jenkins on AWS

Project Overview

This project demonstrates a complete CI/CD pipeline for deploying a containerized Node.js application on an AWS EC2 instance using Jenkins, Docker, and GitHub.

🔹 Use Case: Automate the build and deployment process of a Node.js application when changes are pushed to GitHub.

🔹 Workflow:

1️⃣GitHub hosts the source code for the Node.js application.

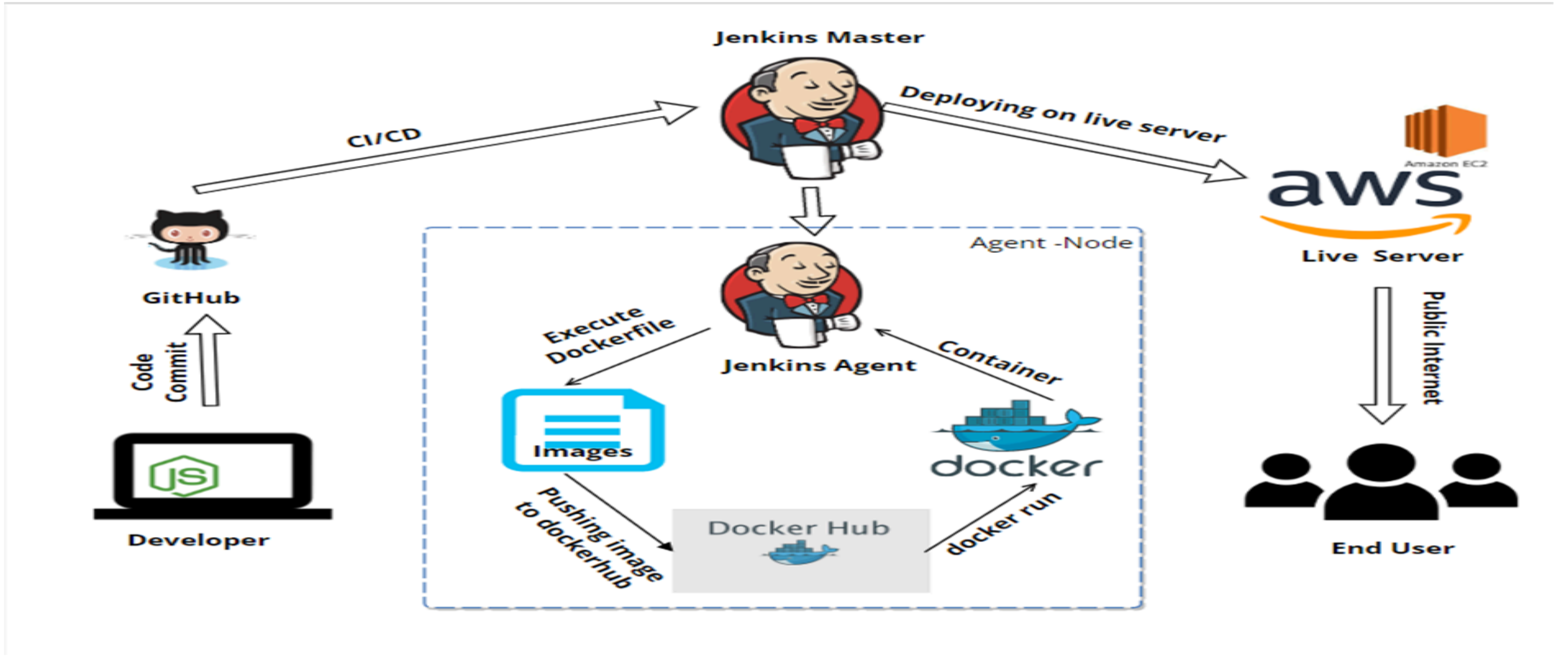
2️⃣Webhook notifies Jenkins whenever a new change is pushed to the repository.

3️⃣Jenkins pulls the latest code, builds a Docker image, and runs the application inside a Docker container.

4️⃣Docker ensures the application runs independently of the EC2 session and is exposed on port 8000.

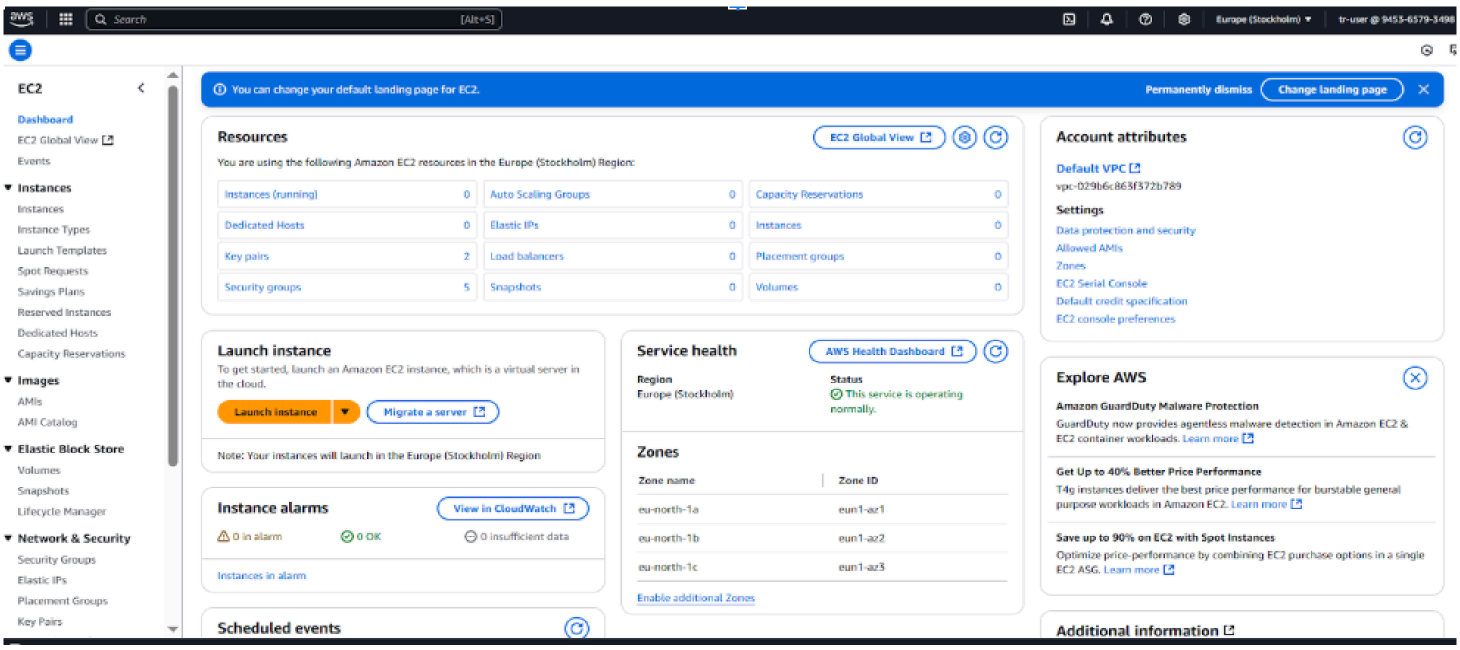
5️⃣AWS EC2 serves as the hosting environment, with appropriate inbound rules for Jenkins (8080) and the application (8000).

This solution provides a simple, scalable, and automated deployment workflow for Node.js applications with continuous integration and delivery (CI/CD) capabilities.



🏗️ Step 1: Launch EC2 Instance on AWS

- 🖥️ **Login** to your AWS Console: https://aws.amazon.com
- 🔍 Go to **EC2 > Launch Instance**



You can change your default landing page for EC2. Permanently dismiss | Change landing page

Resources

You are using the following Amazon EC2 resources in the Europe (Stockholm) Region:

Instances (running)	0	Auto Scaling Groups	0	Capacity Reservations	0
Dedicated Hosts	0	Elastic IPs	0	Instances	0
Key pairs	2	Load balancers	0	Placement groups	0
Security groups	5	Snapshots	0	Volumes	0

Launch instance

To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

Note: Your instances will launch in the Europe (Stockholm) Region

Service health

Region: Europe (Stockholm)
Status: This service is operating normally.

Zones

Zone name	Zone ID
eu-north-1a	eun1-az1
eu-north-1b	eun1-az2
eu-north-1c	eun1-az3

Instance alarms: 0 in alarm, 0 OK, 0 insufficient data

Account attributes

Default VPC: vpc-029b6c863f372b789

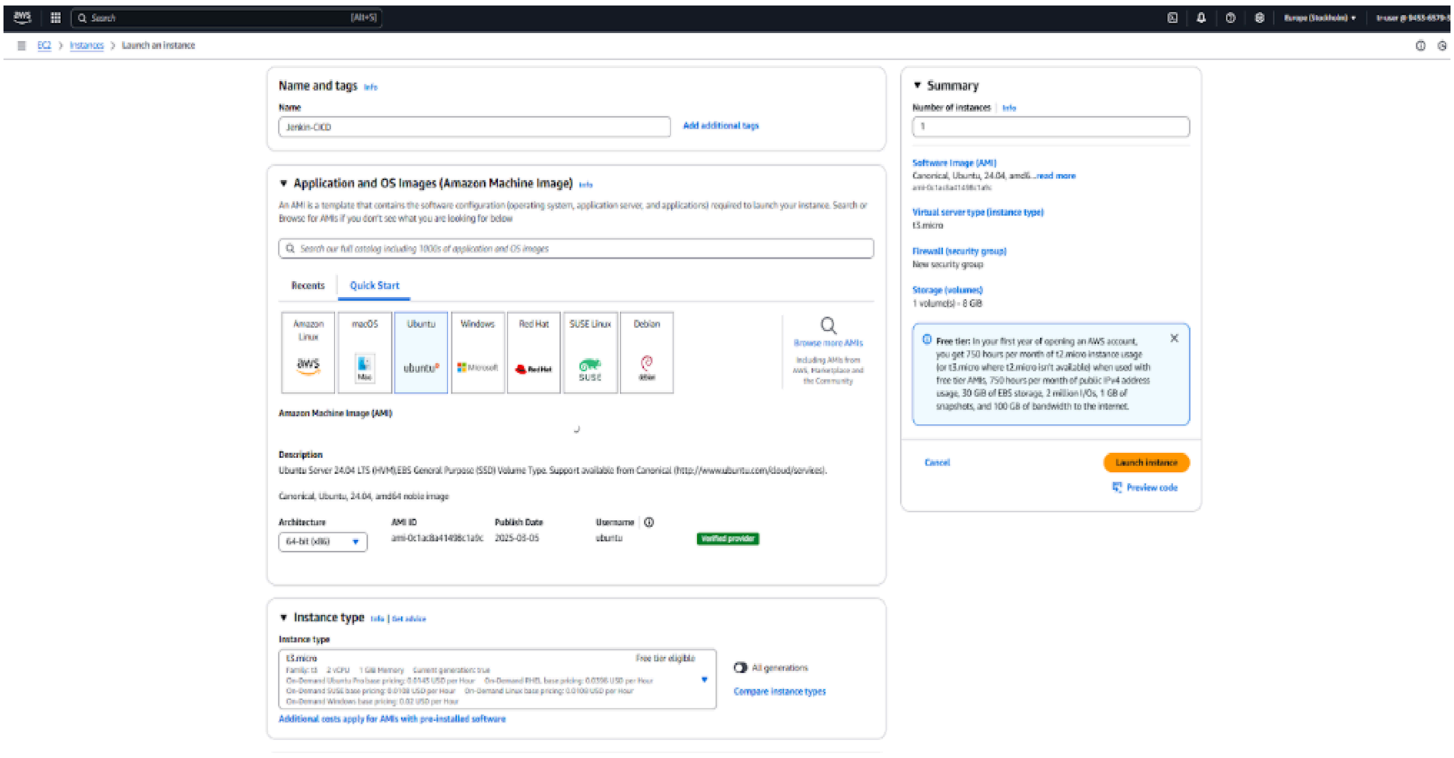
Explore AWS

Amazon GuardDuty Malware Protection
GuardDuty now provides agentless malware detection in Amazon EC2 & EC2 container workloads. Learn more

Get Up to 40% Better Price Performance
T4g instances deliver the best price performance for burstable general purpose workloads in Amazon EC2. Learn more

Save up to 90% on EC2 with Spot Instances
Optimize price-performance by combining EC2 purchase options in a single EC2 ASG. Learn more

- 🏷️ **Name** your instance (e.g.,Jenkins-CICD), 🐧 **Choose OS**: **Ubuntu 20.04 LTS (64-bit x86)**



EC2 > Instances > Launch an instance

Name and tags

Name: Jenkin-CICD

Application and OS Images (Amazon Machine Image)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Quick Start: Ubuntu

Amazon Machine Image (AMI)

Description
Ubuntu Server 24.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).

Canonical, Ubuntu, 24.04, amd64 noble image

Architecture	AMI ID	Publish Date	Username	
64-bit (x86)	ami-0c1ac8a41498c1a9c	2025-03-05	ubuntu	Verified provider

Instance type

Instance type: t3.micro — Free tier eligible
Family: t3 2 vCPU 1 GiB Memory Current generation: true
On-Demand Ubuntu Pro base pricing: 0.0145 USD per Hour On-Demand RHEL base pricing: 0.0396 USD per Hour
On-Demand SUSE base pricing: 0.0108 USD per Hour On-Demand Linux base pricing: 0.0108 USD per Hour
On-Demand Windows base pricing: 0.02 USD per Hour

Summary

Number of instances: 1

Software Image (AMI): Canonical, Ubuntu, 24.04, amd6...read more
ami-0c1ac8a41498c1a9c

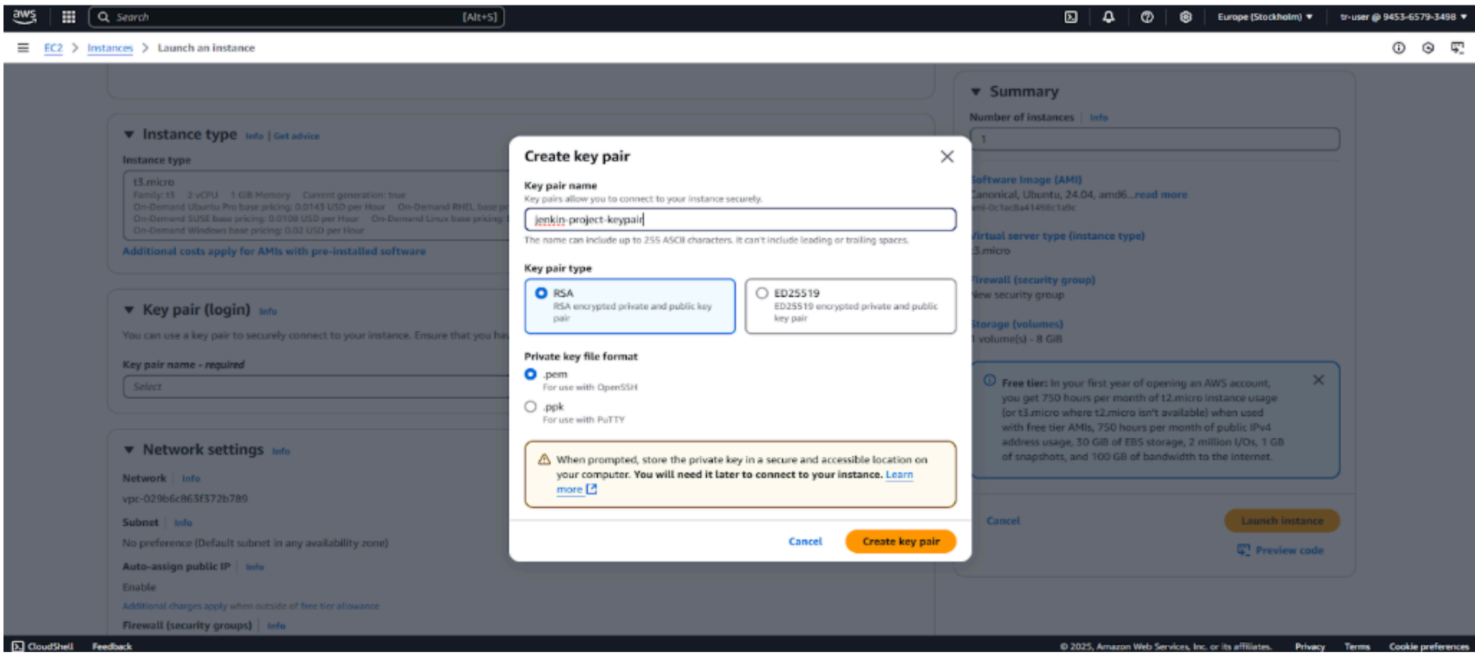
Virtual server type (instance type): t3.micro

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 8 GiB

Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

- 🔑 **Create or Select Key Pair** for SSH access



EC2 > Instances > Launch an instance

Create key pair

Key pair name
Key pairs allow you to connect to your instance securely.
jenkin-project-keypair
The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type
[✓ RSA] RSA encrypted private and public key pair
ED25519 ED25519 encrypted private and public key pair

Private key file format
[✓ .pem] For use with OpenSSH
.ppk For use with PuTTY

When prompted, store the private key in a secure and accessible location on your computer. **You will need it later to connect to your instance.** Learn more

Cancel | Create key pair

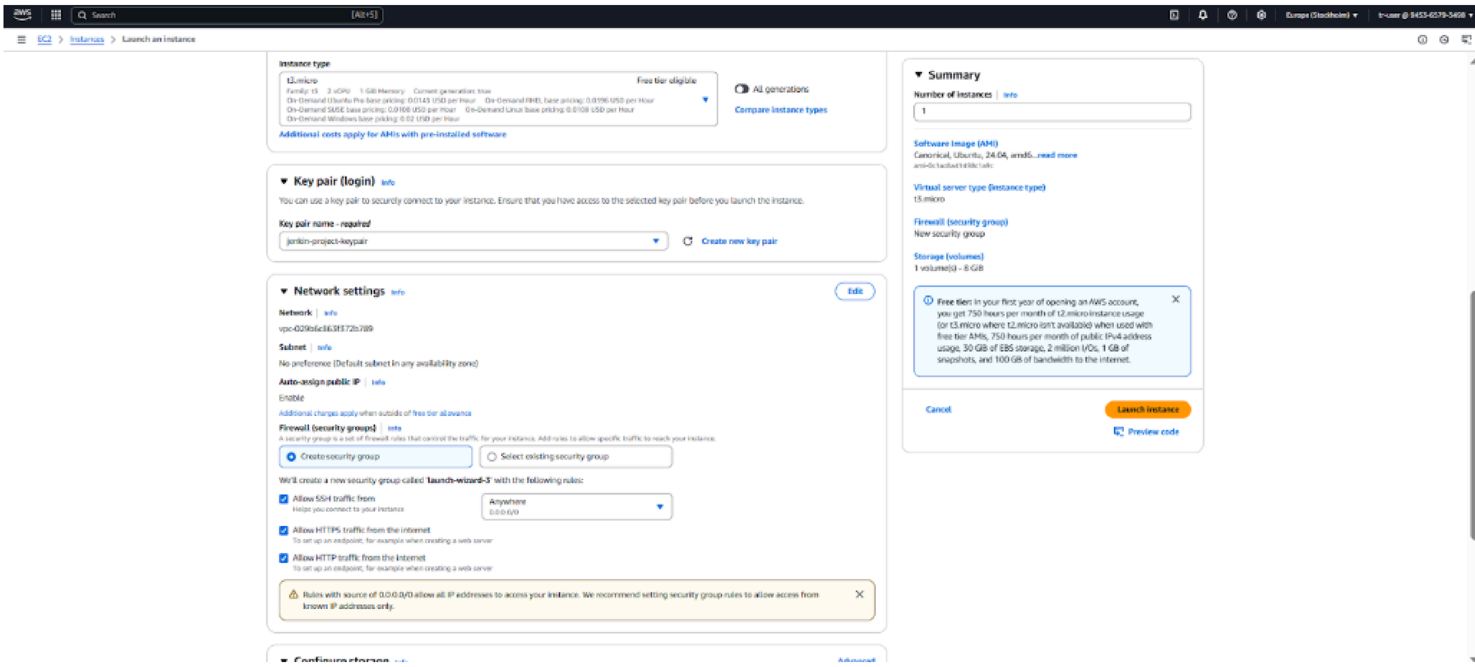
Instance type: t3.micro — Family: t3 2 vCPU 1 GiB Memory Current generation: true
On-Demand Ubuntu Pro base pricing: 0.0143 USD per Hour
On-Demand SUSE base pricing: 0.0108 USD per Hour
On-Demand Windows base pricing: 0.02 USD per Hour
Additional costs apply for AMIs with pre-installed software

Summary
Number of instances: 1
Software Image (AMI): Canonical, Ubuntu, 24.04, amd6...read more
Virtual server type (instance type): t3.micro
Firewall (security group): New security group
Storage (volumes): 1 volume(s) - 8 GiB

Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Network: vpc-029b6c863f372b789
Subnet: No preference (Default subnet in any availability zone)
Auto-assign public IP: Enable

- 🔓 **Configure Security Group**:
  - ✅ Allow **SSH (port 22)** from **your IP**
- 💾 **Storage**: Keep default (8 GiB)
- 🚀 Click **Launch Instance**



EC2 > Instances > Launch an instance

Instance type: t3.micro — Free tier eligible
Family: t3 2 vCPU 1 GiB Memory Current generation: true
On-Demand Ubuntu Pro base pricing: 0.0143 USD per Hour On-Demand RHEL base pricing: 0.0396 USD per Hour
On-Demand SUSE base pricing: 0.0108 USD per Hour On-Demand Linux base pricing: 0.0108 USD per Hour
On-Demand Windows base pricing: 0.02 USD per Hour
Additional costs apply for AMIs with pre-installed software

Key pair (login)
You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.
Key pair name - required: jenkin-project-keypair

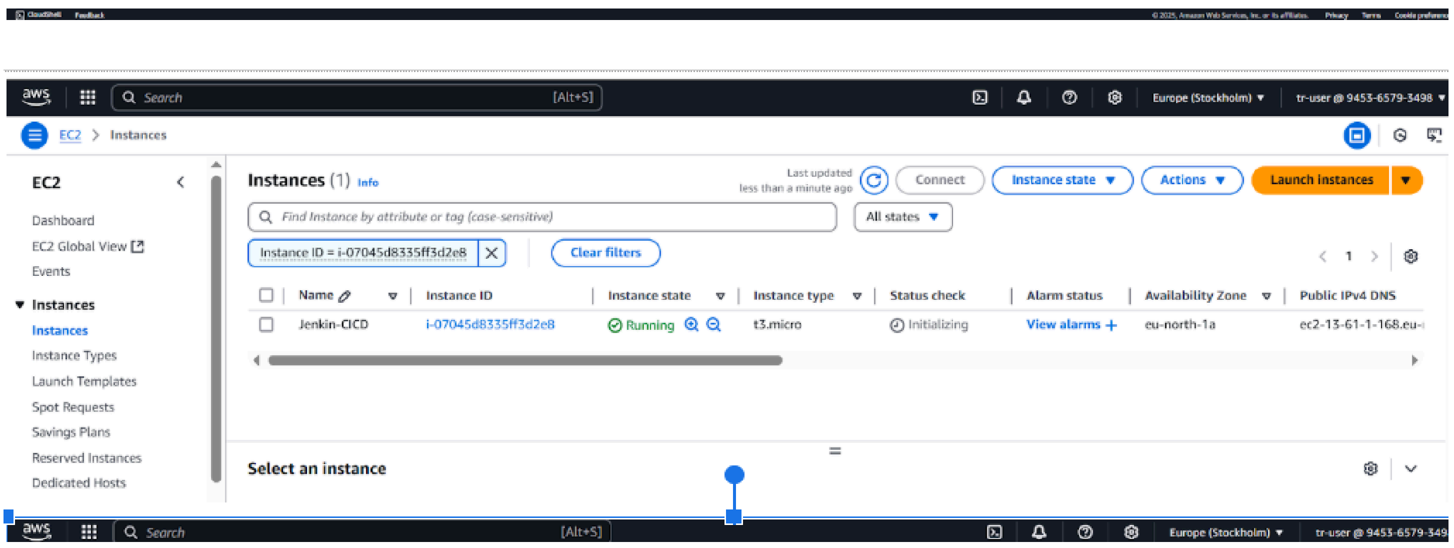
Network settings
Network: vpc-029b6c863f372b789
Subnet: No preference (Default subnet in any availability zone)
Auto-assign public IP: Enable
Firewall (security groups): [✓ Create security group] Select existing security group
We'll create a new security group called 'launch-wizard-3' with the following rules:
[✓] Allow SSH traffic from — Anywhere 0.0.0.0/0
[✓] Allow HTTPS traffic from the internet
[✓] Allow HTTP traffic from the internet
Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Summary
Number of instances: 1
Software Image (AMI): Canonical, Ubuntu, 24.04, amd6...read more
Virtual server type (instance type): t3.micro
Firewall (security group): New security group
Storage (volumes): 1 volume(s) - 8 GiB

Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

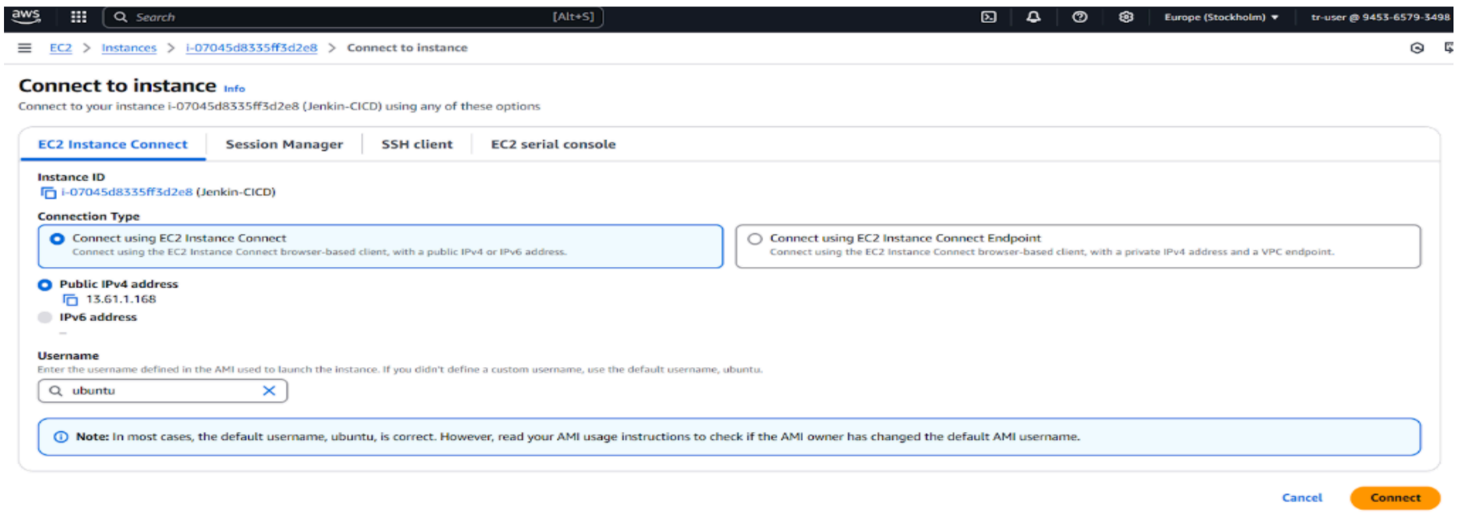
Cancel | Launch instance | Preview code

- 🟢 Wait for instance status to show **"Running"**


Instances (1) Info

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
☐	Jenkin-CICD	i-07045d8335ff3d2e8	Running	t3.micro	Initializing	View alarms +	eu-north-1a	ec2-13-61-1-168.eu-

- 🔗 **Connect via SSH:**


Connect to instance Info

Connect to your instance i-07045d8335ff3d2e8 (Jenkin-CICD) using any of these options

EC2 Instance Connect | Session Manager | SSH client | EC2 serial console

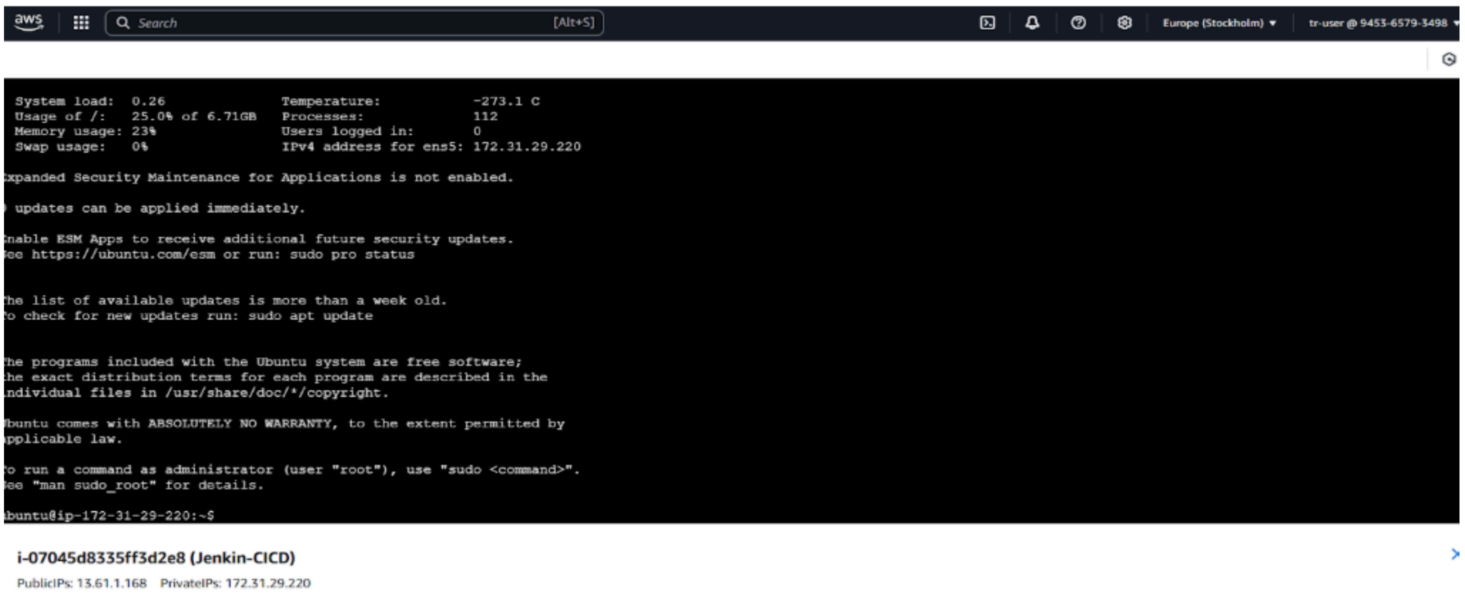
Instance ID
i-07045d8335ff3d2e8 (Jenkin-CICD)

Connection Type
[✓ Connect using EC2 Instance Connect] Connect using the EC2 Instance Connect browser-based client, with a public IPv4 or IPv6 address.
Connect using EC2 Instance Connect Endpoint — Connect using the EC2 Instance Connect browser-based client, with a private IPv4 address and a VPC endpoint.

[✓ Public IPv4 address] 13.61.1.168
IPv6 address

Username
Enter the username defined in the AMI used to launch the instance. If you didn't define a custom username, use the default username, ubuntu.
ubuntu

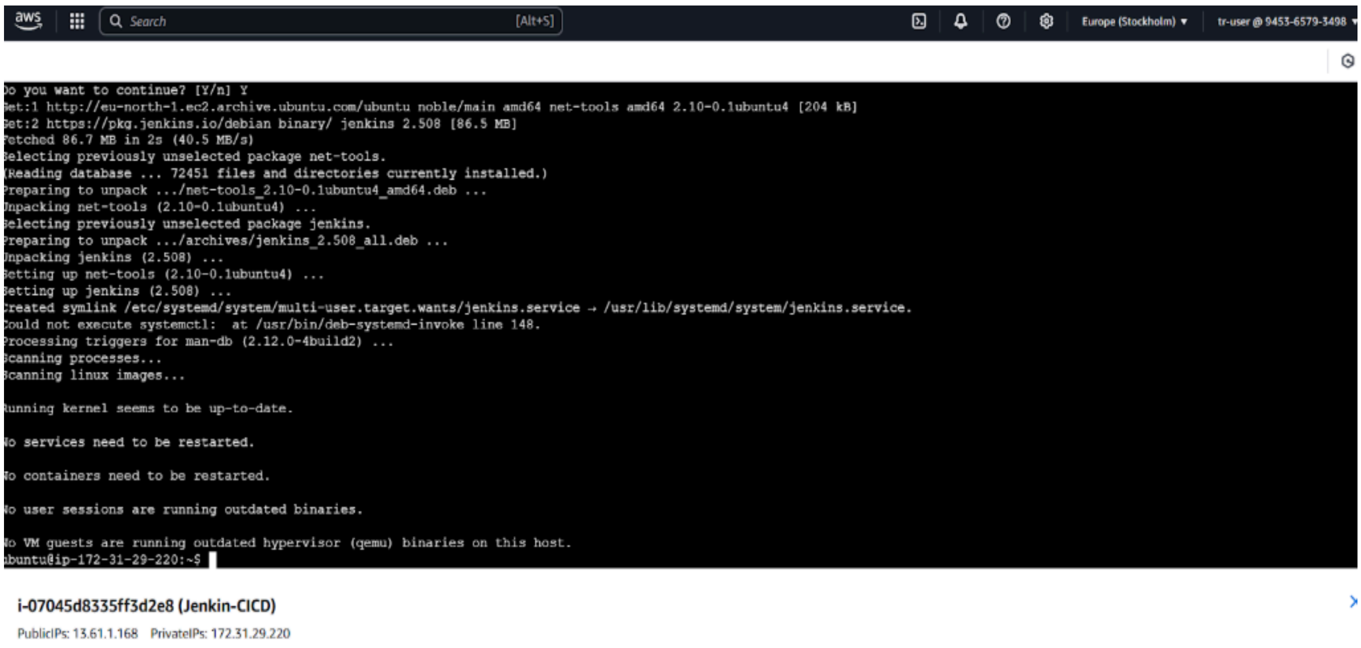
Note: In most cases, the default username, ubuntu, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.


```
 System load:  0.26              Temperature:           -273.1 C
 Usage of /:   25.0% of 6.71GB   Processes:             112
 Memory usage: 23%               Users logged in:       0
 Swap usage:   0%                IPv4 address for ens5: 172.31.29.220

xpanded Security Maintenance for Applications is not enabled.

 updates can be applied immediately.

nable ESM Apps to receive additional future security updates.
ee https://ubuntu.com/esm or run: sudo pro status


he list of available updates is more than a week old.
o check for new updates run: sudo apt update


he programs included with the Ubuntu system are free software;
he exact distribution terms for each program are described in the
ndividual files in /usr/share/doc/*/copyright.

buntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
pplicable law.

o run a command as administrator (user "root"), use "sudo <command>".
ee "man sudo_root" for details.

buntu@ip-172-31-29-220:~$
```

i-07045d8335ff3d2e8 (Jenkin-CICD)
PublicIPs: 13.61.1.168 PrivateIPs: 172.31.29.220

🧱 Step 2: Install Jenkins on Ubuntu EC2 Instance

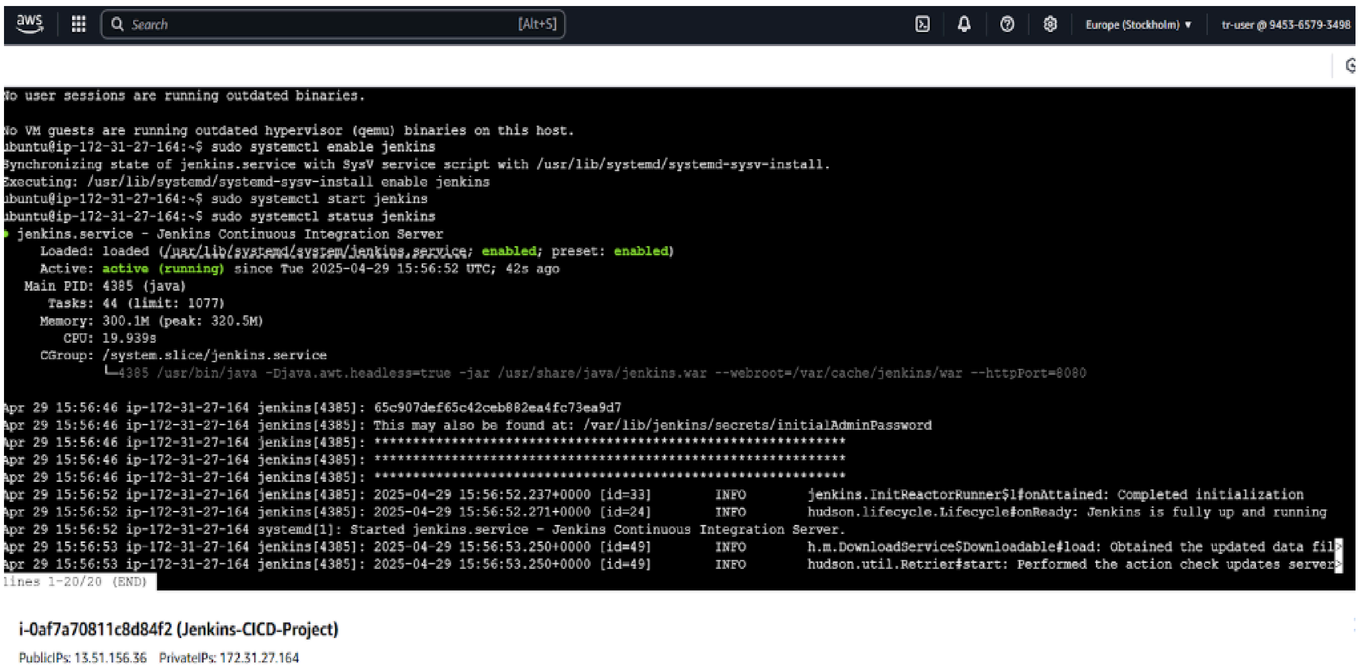
- 🔄 Update the Package List
- ☕ Install Java (OpenJDK 11)
- ✅ Verify Java Installation
- 🔐 Add Jenkins GPG Key
- 📦 Add Jenkins Package Repository
- ⚙️ Install Jenkins



```
Do you want to continue? [Y/n] Y
Get:1 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 net-tools amd64 2.10-0.1ubuntu4 [204 kB]
Get:2 https://pkg.jenkins.io/debian binary/ jenkins 2.508 [86.5 MB]
Fetched 86.7 MB in 2s (40.5 MB/s)
Selecting previously unselected package net-tools.
(Reading database ... 72451 files and directories currently installed.)
Preparing to unpack .../net-tools_2.10-0.1ubuntu4_amd64.deb ...
Unpacking net-tools (2.10-0.1ubuntu4) ...
Selecting previously unselected package jenkins.
Preparing to unpack .../archives/jenkins_2.508_all.deb ...
Unpacking jenkins (2.508) ...
Setting up net-tools (2.10-0.1ubuntu4) ...
Setting up jenkins (2.508) ...
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /usr/lib/systemd/system/jenkins.service.
Could not execute systemctl:  at /usr/bin/deb-systemd-invoke line 148.
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-29-220:~$
```

i-07045d8335ff3d2e8 (Jenkin-CICD)
PublicIPs: 13.61.1.168   PrivateIPs: 172.31.29.220

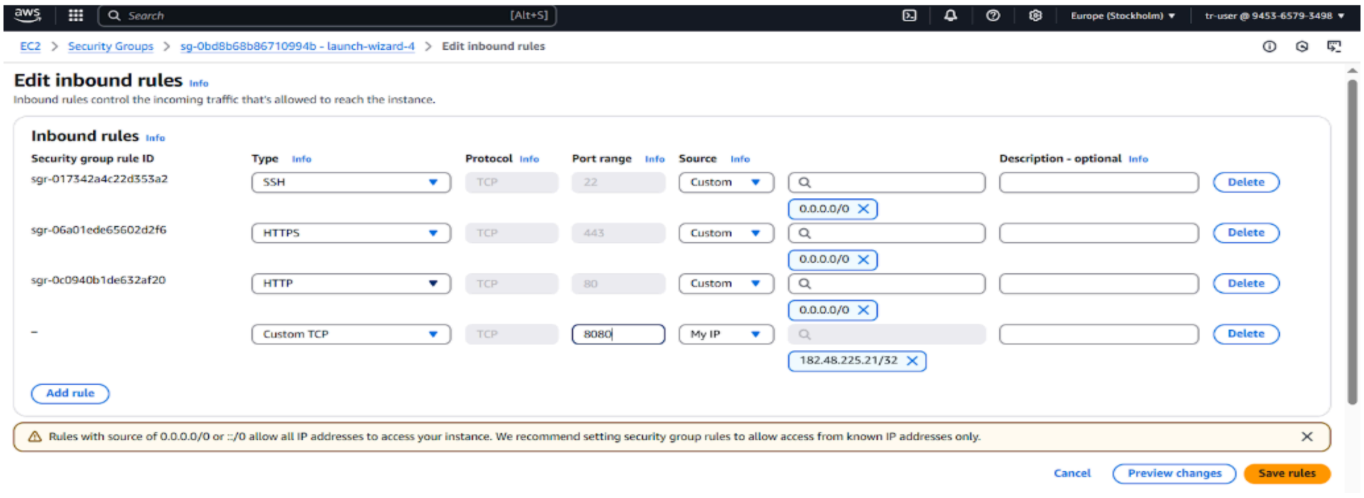
- 🚀 Start Jenkins



```
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-27-164:~$ sudo systemctl enable jenkins
Synchronizing state of jenkins.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable jenkins
ubuntu@ip-172-31-27-164:~$ sudo systemctl start jenkins
ubuntu@ip-172-31-27-164:~$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
     Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
     Active: active (running) since Tue 2025-04-29 15:56:52 UTC; 42s ago
   Main PID: 4385 (java)
      Tasks: 44 (limit: 1077)
     Memory: 300.1M (peak: 320.5M)
        CPU: 19.939s
     CGroup: /system.slice/jenkins.service
             └─4385 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080

Apr 29 15:56:46 ip-172-31-27-164 jenkins[4385]: 65c907def65c42ceb882ea4fc73ea9d7
Apr 29 15:56:46 ip-172-31-27-164 jenkins[4385]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Apr 29 15:56:46 ip-172-31-27-164 jenkins[4385]: *************************************************************
Apr 29 15:56:46 ip-172-31-27-164 jenkins[4385]: *************************************************************
Apr 29 15:56:46 ip-172-31-27-164 jenkins[4385]: *************************************************************
Apr 29 15:56:52 ip-172-31-27-164 jenkins[4385]: 2025-04-29 15:56:52.237+0000 [id=33]        INFO        jenkins.InitReactorRunner$1#onAttained: Completed initialization
Apr 29 15:56:52 ip-172-31-27-164 jenkins[4385]: 2025-04-29 15:56:52.271+0000 [id=24]        INFO        hudson.lifecycle.Lifecycle#onReady: Jenkins is fully up and running
Apr 29 15:56:52 ip-172-31-27-164 systemd[1]: Started jenkins.service - Jenkins Continuous Integration Server.
Apr 29 15:56:53 ip-172-31-27-164 jenkins[4385]: 2025-04-29 15:56:53.250+0000 [id=49]        INFO        h.m.DownloadService$Downloadable#load: Obtained the updated data fil>
Apr 29 15:56:53 ip-172-31-27-164 jenkins[4385]: 2025-04-29 15:56:53.250+0000 [id=49]        INFO        hudson.util.Retrier#start: Performed the action check updates server>
lines 1-20/20 (END)
```

i-0af7a70811c8d84f2 (Jenkins-CICD-Project)
PublicIPs: 13.51.156.36   PrivateIPs: 172.31.27.164

- 🛠️ **Enable Inbound Port 8080 in AWS**



EC2 > Security Groups > sg-0bd8b68b86710994b - launch-wizard-4 > Edit inbound rules

Edit inbound rules Info

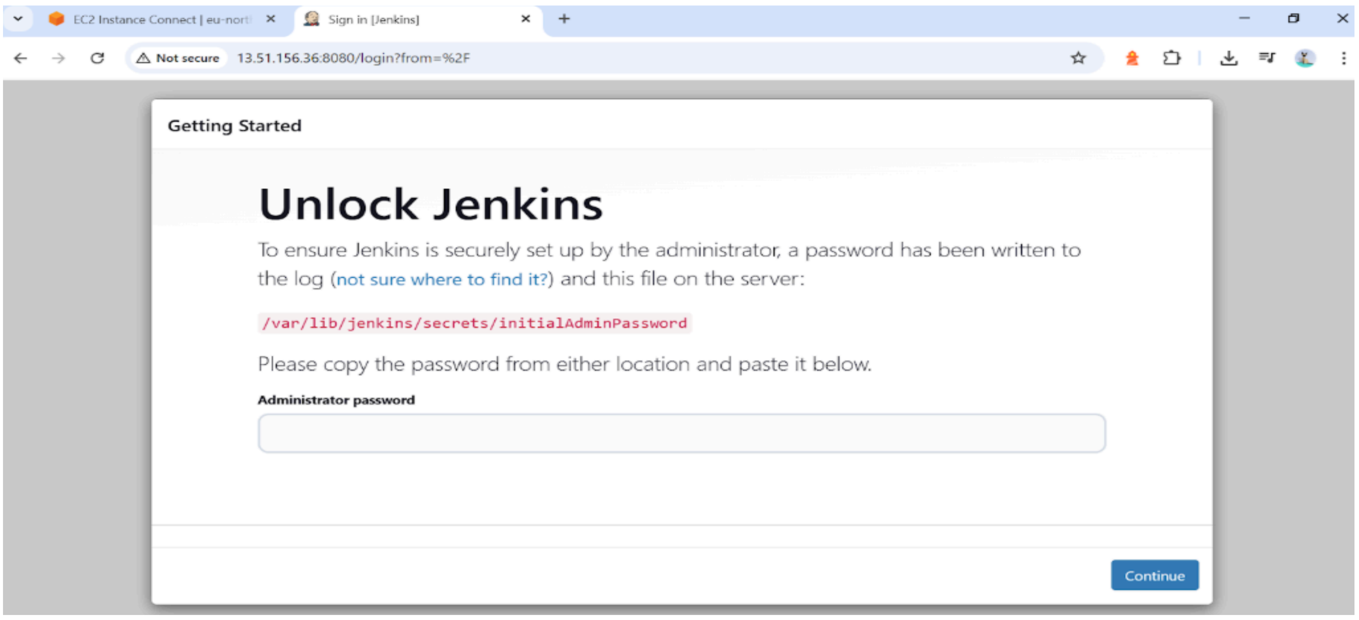
Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type	Protocol	Port range	Source		Description - optional
sgr-017342a4c22d353a2	SSH	TCP	22	Custom	0.0.0.0/0	
sgr-06a01ede65602d2f6	HTTPS	TCP	443	Custom	0.0.0.0/0	
sgr-0c0940b1de632af20	HTTP	TCP	80	Custom	0.0.0.0/0	
–	Custom TCP	TCP	8080	My IP	182.48.225.21/32	

⚠ Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

- 🌐 **Access Jenkins in Browser**



EC2 Instance Connect | eu-nort
Sign in [Jenkins]
Not secure 13.51.156.36:8080/login?from=%2F

Getting Started

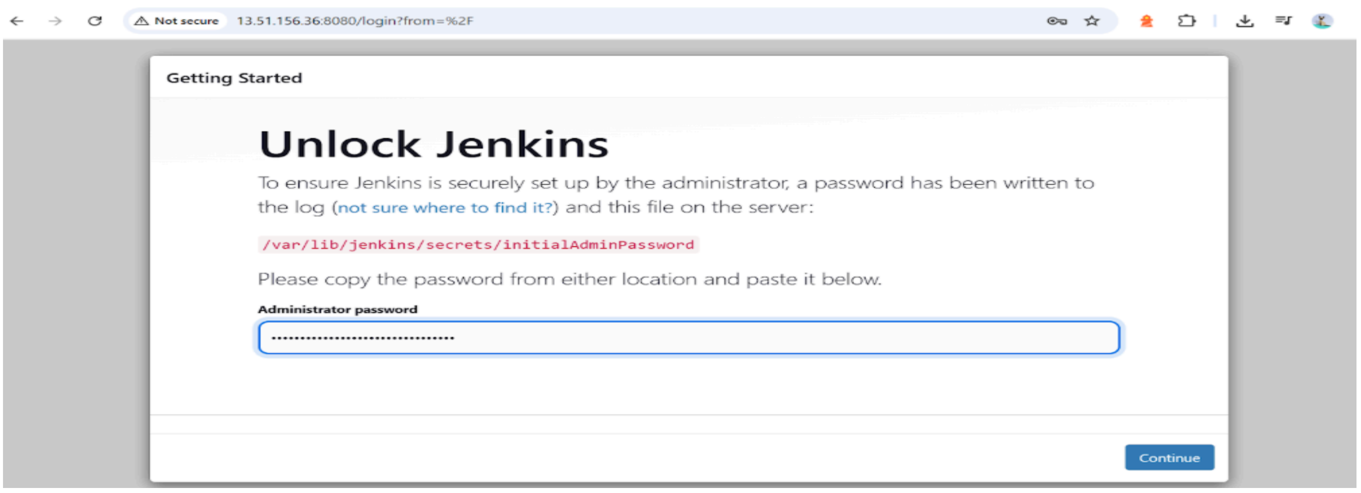
Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

`/var/lib/jenkins/secrets/initialAdminPassword`

Please copy the password from either location and paste it below.

Administrator password



Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

`/var/lib/jenkins/secrets/initialAdminPassword`

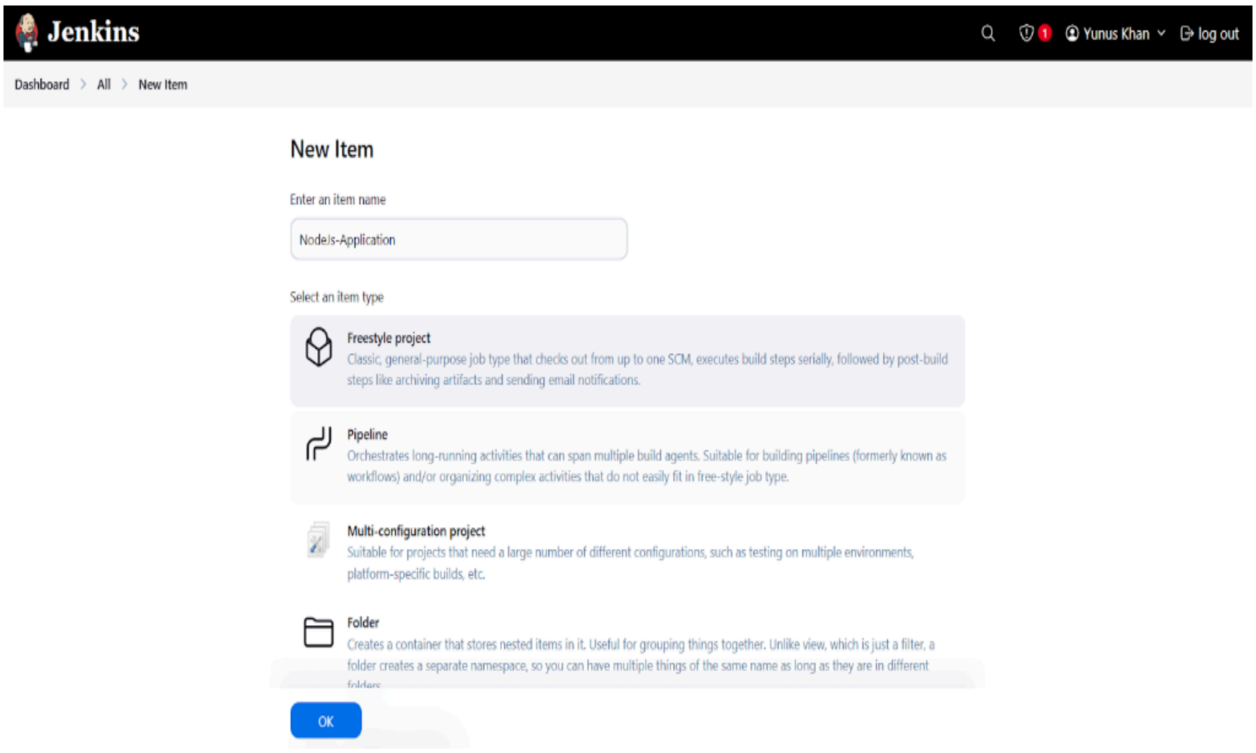
Please copy the password from either location and paste it below.

Administrator password

••••••••••••••••••••••••••••••••

🧪 Step3: Create Freestyle Jenkins Project

- 🎯 **Created a freestyle Jenkins project**



Dashboard > All > New Item

New Item

Enter an item name

NodeJs-Application

Select an item type

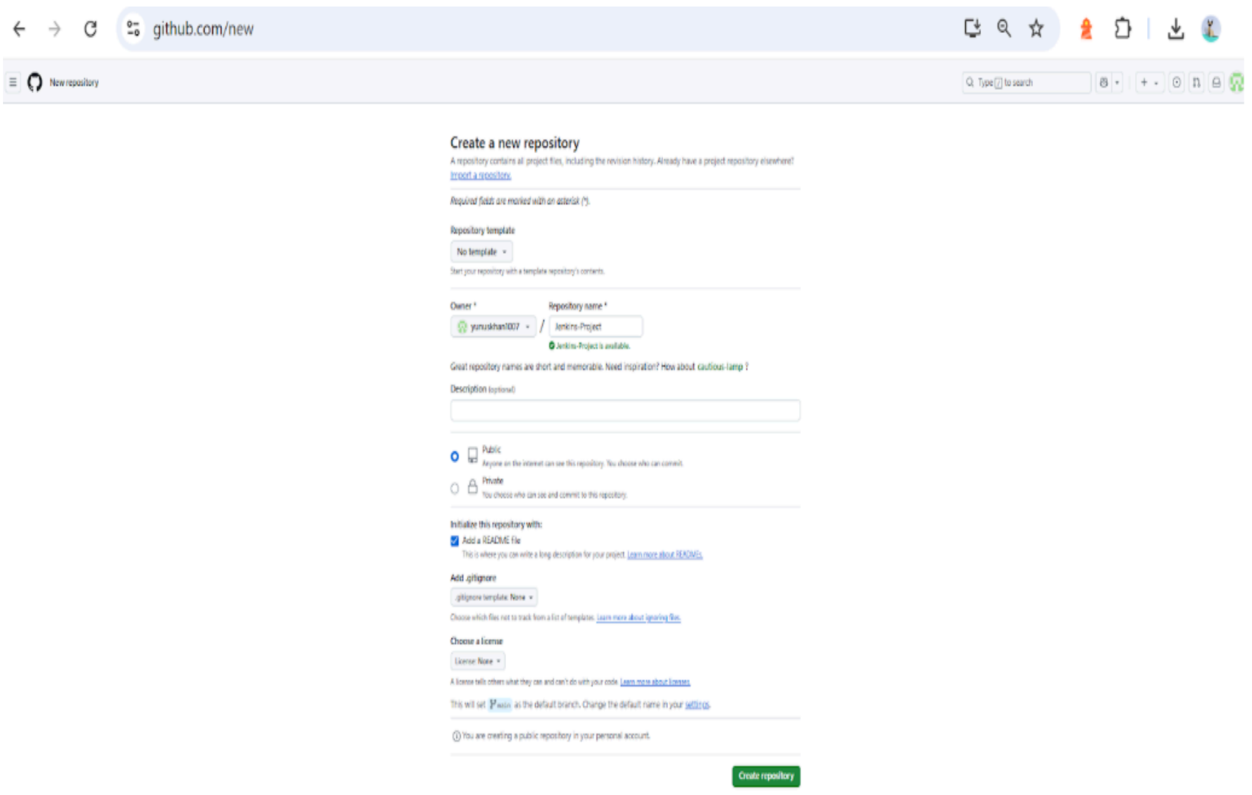
Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

- 🌐 **Create a Git Repository (e.g., on GitHub), Push Your Code to the Repo**



github.com/new

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Required fields are marked with an asterisk (*).

Repository template: No template

Start your repository with a template repository's contents.

Owner *: yunuskhan1007 / Repository name *: Jenkins-Project

Jenkins-Project is available.

Great repository names are short and memorable. Need inspiration? How about cautious-lamp ?

Description (optional)

[✓ Public] Anyone on the internet can see this repository. You choose who can commit.

Private — You choose who can see and commit to this repository.

Initialize this repository with:

☑ Add a README file
This is where you can write a long description for your project. Learn more about READMEs.

Add .gitignore: .gitignore template: None

Choose which files not to track from a list of templates. Learn more about ignoring files.

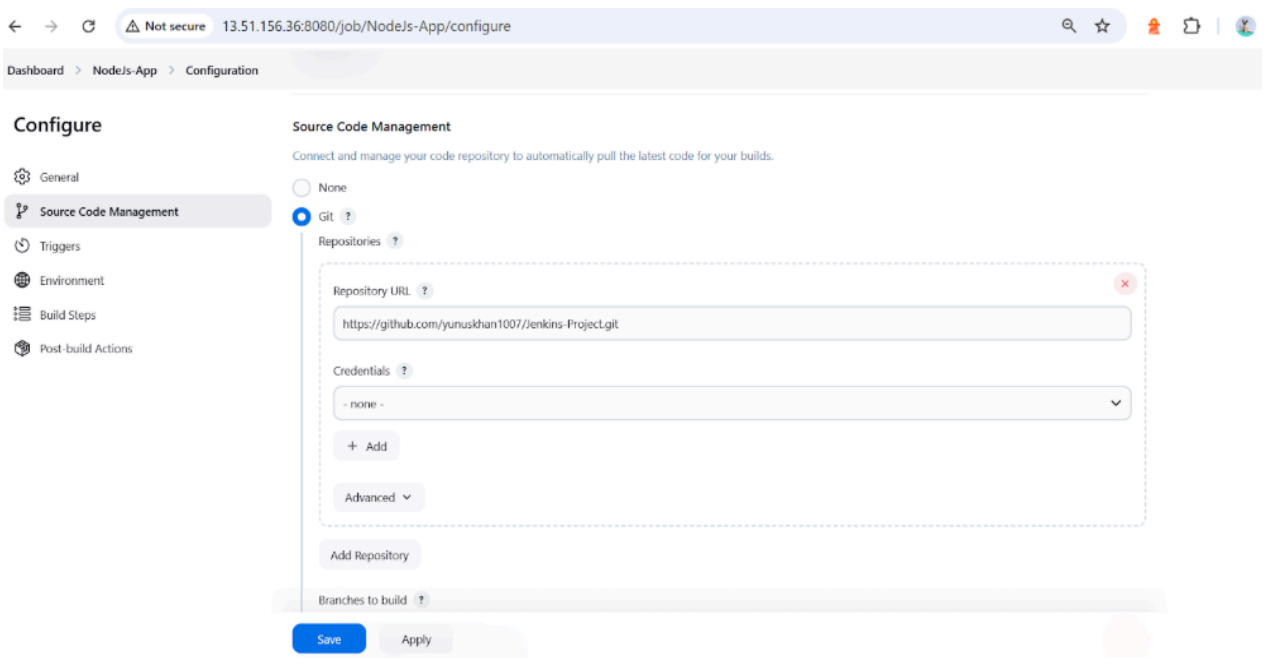
Choose a license: License: None

A license tells others what they can and can't do with your code. Learn more about licenses.

This will set main as the default branch. Change the default name in your settings.

ⓘ You are creating a public repository in your personal account.

- 🔗 **Connected it to your Git repo**



Source Code Management

Connect and manage your code repository to automatically pull the latest code for your builds.

None [✓ Git]

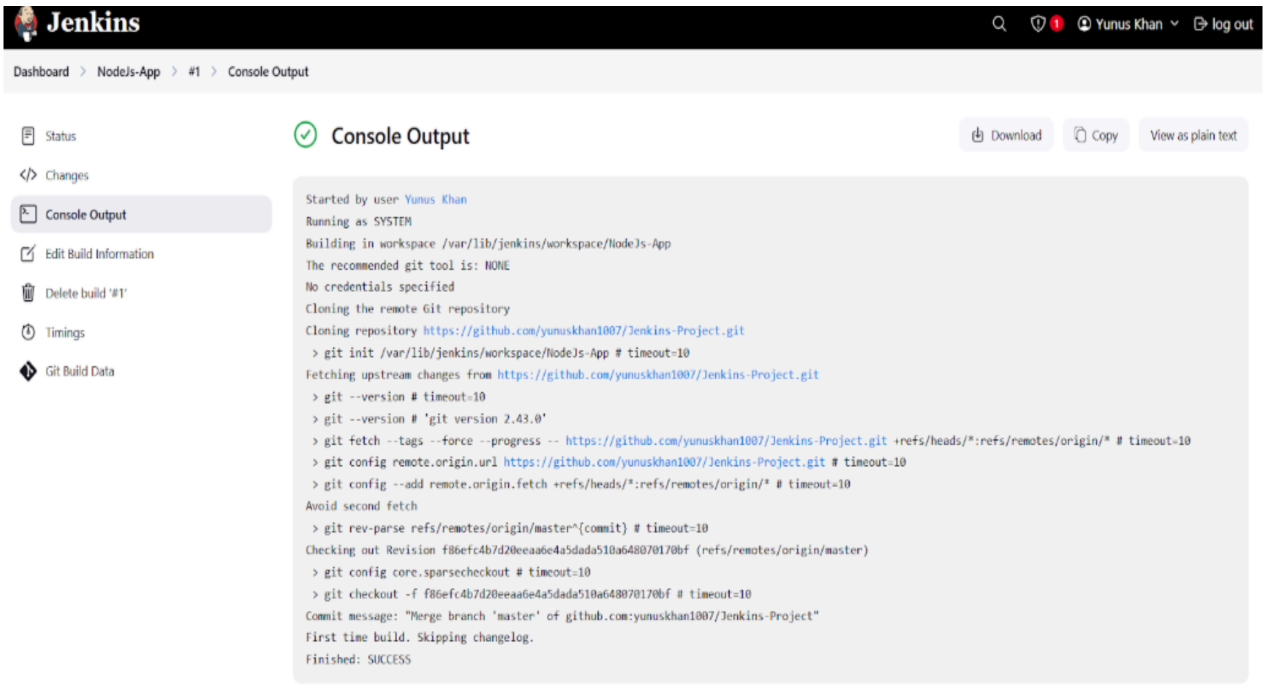
Repositories

Repository URL: https://github.com/yunuskhan1007/Jenkins-Project.git

Credentials: - none -

Branches to build

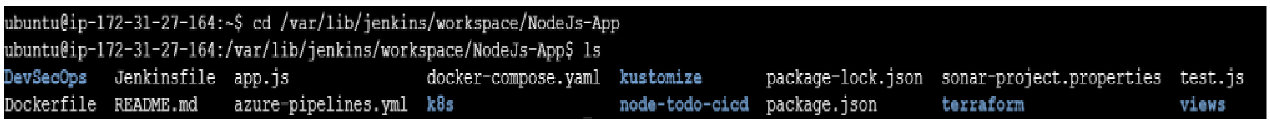
- 🚀 **Successfully built the job**



Console Output

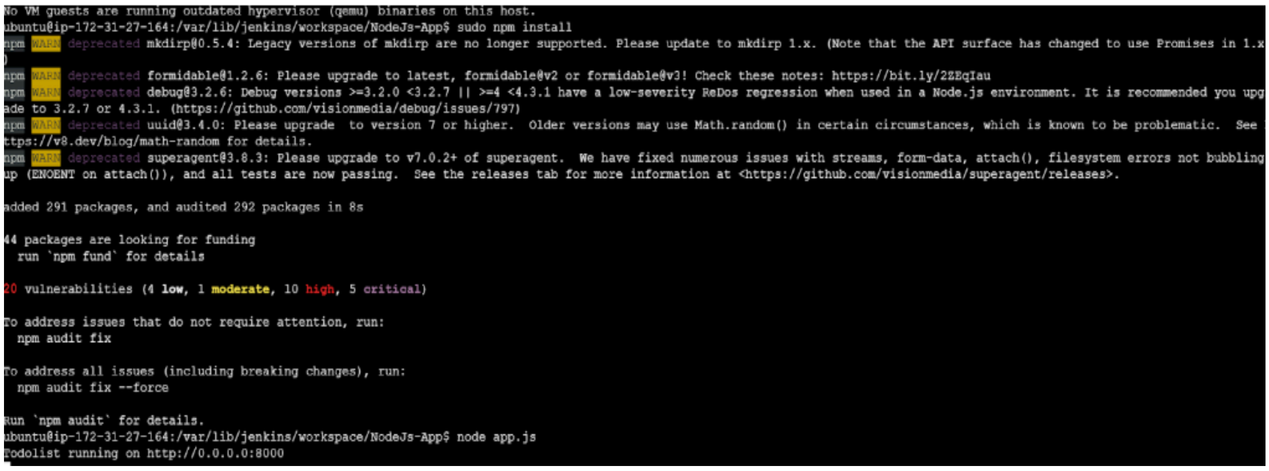
```
Started by user Yunus Khan
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/NodeJs-App
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/yunuskhan1007/Jenkins-Project.git
 > git init /var/lib/jenkins/workspace/NodeJs-App # timeout=10
Fetching upstream changes from https://github.com/yunuskhan1007/Jenkins-Project.git
 > git --version # timeout=10
 > git --version # 'git version 2.43.0'
 > git fetch --tags --force --progress -- https://github.com/yunuskhan1007/Jenkins-Project.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/yunuskhan1007/Jenkins-Project.git # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision f86efc4b7d20eeaa6e4a5dada510a648070170bf (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f f86efc4b7d20eeaa6e4a5dada510a648070170bf # timeout=10
Commit message: "Merge branch 'master' of github.com:yunuskhan1007/Jenkins-Project"
First time build. Skipping changelog.
Finished: SUCCESS
```

- 📂 **Verified all files are present in the project workspace**



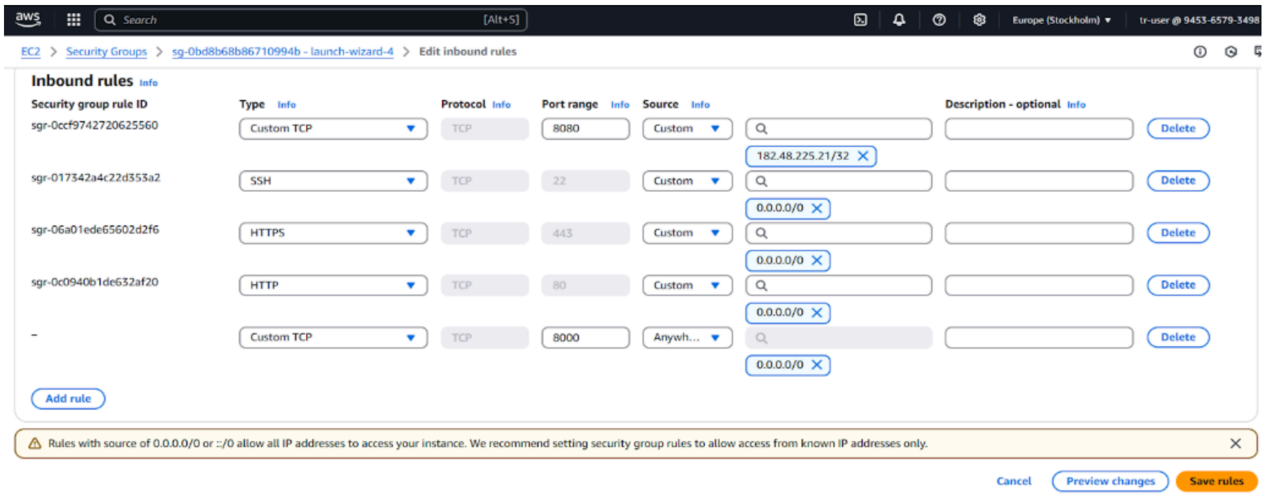
```
ubuntu@ip-172-31-27-164:~$ cd /var/lib/jenkins/workspace/NodeJs-App
ubuntu@ip-172-31-27-164:/var/lib/jenkins/workspace/NodeJs-App$ ls
DevSecOps   Jenkinsfile  app.js              docker-compose.yaml  kustomize      package-lock.json  sonar-project.properties  test.js
Dockerfile  README.md    azure-pipelines.yml  k8s                 node-todo-cicd  package.json       terraform                 views
```

- ✅ **Node.js app is running with node app.js**



```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-27-164:/var/lib/jenkins/workspace/NodeJs-App$ sudo npm install
npm WARN deprecated mkdirp@0.5.4: Legacy versions of mkdirp are no longer supported. Please update to mkdirp 1.x. (Note that the API surface has changed to use Promises in 1.x
)
npm WARN deprecated formidable@1.2.6: Please upgrade to latest, formidable@v2 or formidable@v3! Check these notes: https://bit.ly/2ZEqIau
npm WARN deprecated debug@3.2.6: Debug versions >=3.2.0 <3.2.7 || >=4 <4.3.1 have a low-severity ReDos regression when used in a Node.js environment. It is recommended you upg
ade to 3.2.7 or 4.3.1. (https://github.com/visionmedia/debug/issues/797)
npm WARN deprecated uuid@3.4.0: Please upgrade  to version 7 or higher.  Older versions may use Math.random() in certain circumstances, which is known to be problematic.  See
ttps://v8.dev/blog/math-random for details.
npm WARN deprecated superagent@3.8.3: Please upgrade to v7.0.2+ of superagent.  We have fixed numerous issues with streams, form-data, attach(), filesystem errors not bubbling
up (ENOENT on attach()), and all tests are now passing.  See the releases tab for more information at <https://github.com/visionmedia/superagent/releases>.

added 291 packages, and audited 292 packages in 8s

44 packages are looking for funding
  run `npm fund` for details

20 vulnerabilities (4 low, 1 moderate, 10 high, 5 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
ubuntu@ip-172-31-27-164:/var/lib/jenkins/workspace/NodeJs-App$ node app.js
Todolist running on http://0.0.0.0:8000
```

- 🛠️ **Enable Inbound Port 8080 in AWS**



EC2 > Security Groups > sg-0bd8b68b86710994b - launch-wizard-4 > Edit inbound rules

Inbound rules Info

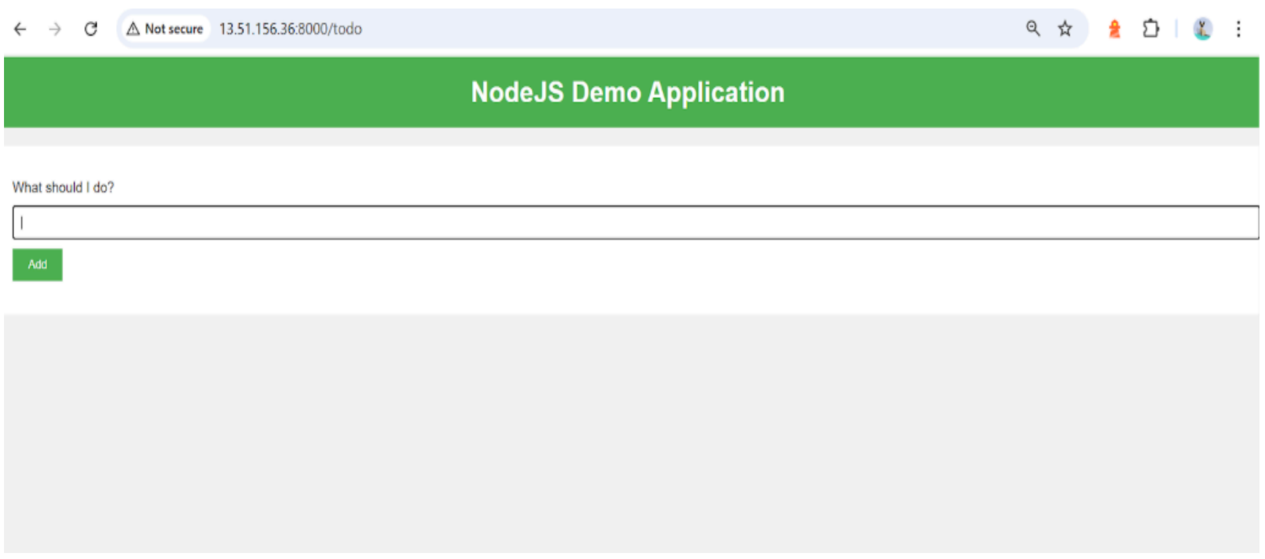
Security group rule ID	Type	Protocol	Port range	Source		Description - optional
sgr-0ccf9742720625560	Custom TCP	TCP	8080	Custom	182.48.225.21/32	
sgr-017342a4c22d353a2	SSH	TCP	22	Custom	0.0.0.0/0	
sgr-06a01ede65602d2f6	HTTPS	TCP	443	Custom	0.0.0.0/0	
sgr-0c0940b1de632af20	HTTP	TCP	80	Custom	0.0.0.0/0	
–	Custom TCP	TCP	8000	Anywh...	0.0.0.0/0	

Add rule

⚠ Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Cancel Preview changes Save rules

- 🧪 **Accessing via browser**



Not secure 13.51.156.36:8000/todo

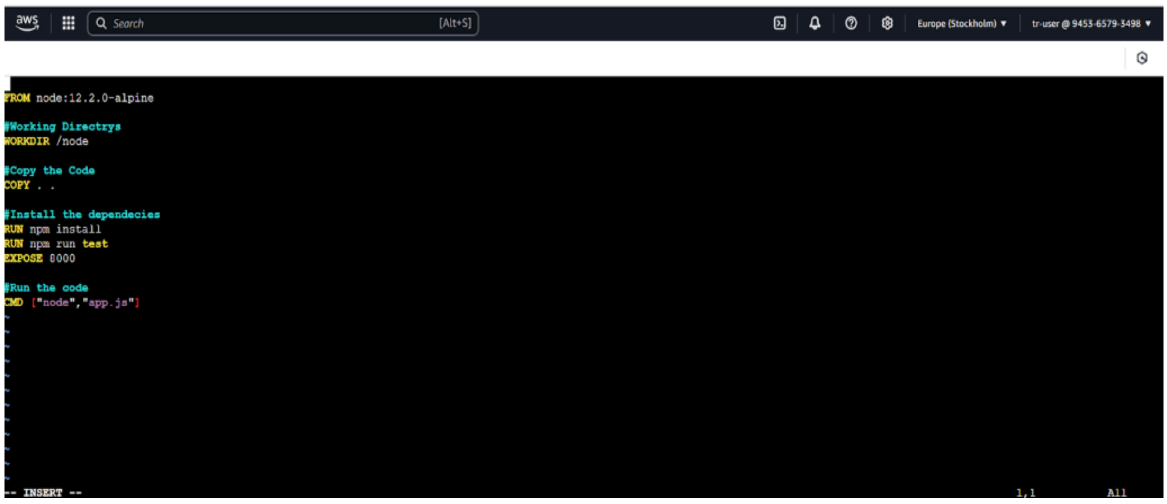
NodeJS Demo Application

What should I do?

Add

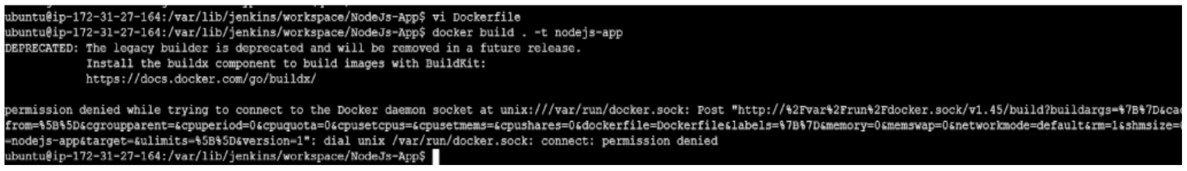
🐳 Step4: Add Dockerfile to Your GitHub Repo

- 🔧 **Create a Dockerfile**



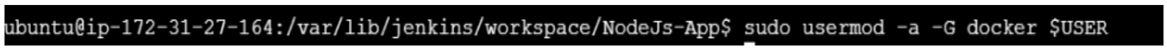
```
FROM node:12.2.0-alpine

#Working Directrys
WORKDIR /node

#Copy the Code
COPY . .

#Install the dependecies
RUN npm install
RUN npm run test
EXPOSE 8000

#Run the code
CMD ["node","app.js"]
~
~
~
~
~
~
~
~
~
~
~
-- INSERT --                                                                  1,1           All
```

- 🐳 **Build the Docker Image**



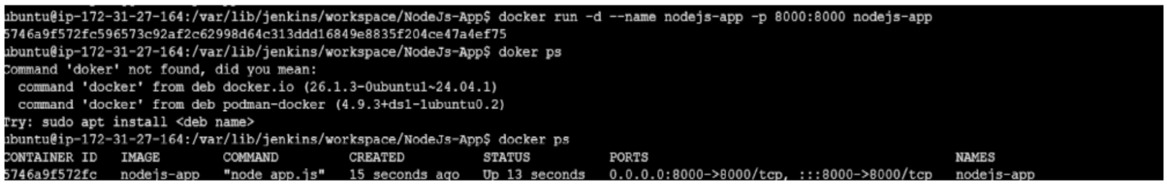
```
ubuntu@ip-172-31-27-164:/var/lib/jenkins/workspace/NodeJs-App$ vi Dockerfile
ubuntu@ip-172-31-27-164:/var/lib/jenkins/workspace/NodeJs-App$ docker build . -t nodejs-app
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post "http://%2Fvar%2Frun%2Fdocker.sock/v1.45/build?buildargs=%7B%7D&ca
from=%5B%5D&cgroupparent=&cpuperiod=0&cpuquota=0&cpusetcpus=&cpusetmems=&cpushares=0&dockerfile=Dockerfile&labels=%7B%7D&memory=0&memswap=0&networkmode=default&rm=1&shmsize=
=nodejs-app&target=&ulimits=%5B%5D&version=1": dial unix /var/run/docker.sock: connect: permission denied
ubuntu@ip-172-31-27-164:/var/lib/jenkins/workspace/NodeJs-App$
```

- 🔐 **Fix Docker Permissions (if needed)**



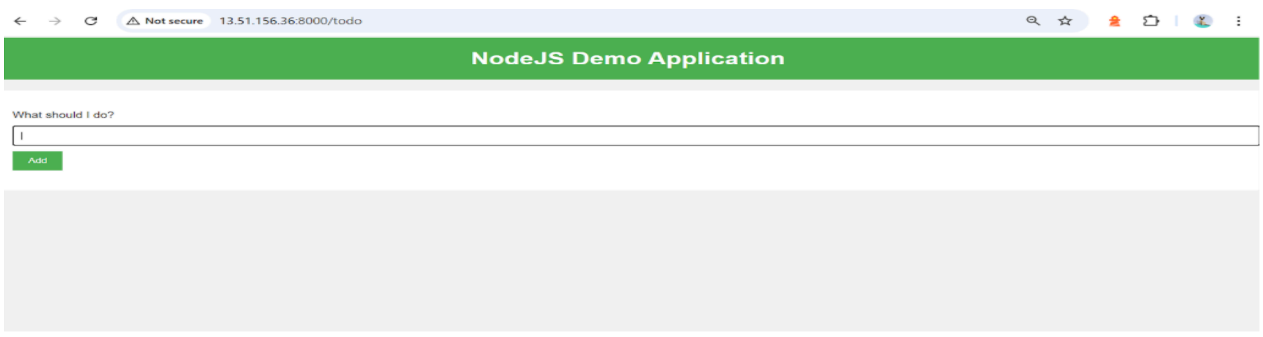
```
ubuntu@ip-172-31-27-164:/var/lib/jenkins/workspace/NodeJs-App$ sudo usermod -a -G docker $USER
```

- 🚀 **Run the App in Detached Mode**



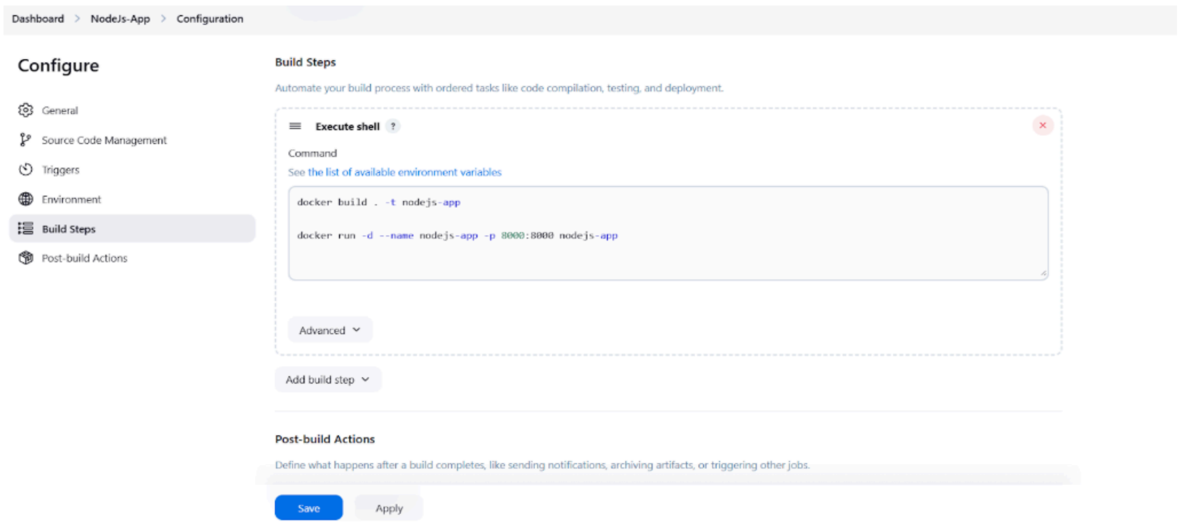
```
ubuntu@ip-172-31-27-164:/var/lib/jenkins/workspace/NodeJs-App$ docker run -d --name nodejs-app -p 8000:8000 nodejs-app
5746a9f572fc596573c92af2c62998d64c313ddd16849e8835f204ce47a4ef75
ubuntu@ip-172-31-27-164:/var/lib/jenkins/workspace/NodeJs-App$ doker ps
Command 'doker' not found, did you mean:
  command 'docker' from deb docker.io (26.1.3-0ubuntu1~24.04.1)
  command 'docker' from deb podman-docker (4.9.3+ds1-1ubuntu0.2)
Try: sudo apt install <deb name>
ubuntu@ip-172-31-27-164:/var/lib/jenkins/workspace/NodeJs-App$ docker ps
CONTAINER ID   IMAGE        COMMAND         CREATED          STATUS          PORTS                                       NAMES
5746a9f572fc   nodejs-app   "node app.js"   15 seconds ago   Up 13 seconds   0.0.0.0:8000->8000/tcp, :::8000->8000/tcp   nodejs-app
```

- 🌐 **Access Your Application**



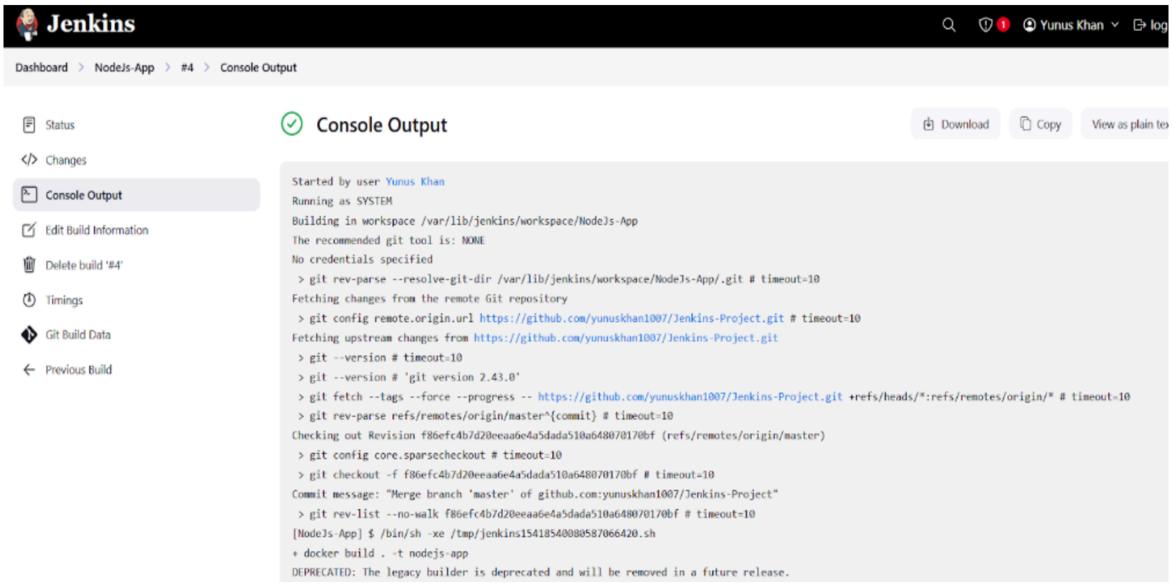
⚙️ Step5. Add Shell Commands to Jenkins Job

- 🛠️ **Scroll to Build → Add build step → Execute shell**


Dashboard > NodeJs-App > Configuration
Configure
General
Source Code Management
Triggers
Environment
Build Steps
Post-build Actions
Build Steps
Automate your build process with ordered tasks like code compilation, testing, and deployment.
Execute shell
Command
See the list of available environment variables
Advanced
Add build step
Post-build Actions
Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.
Save
Apply

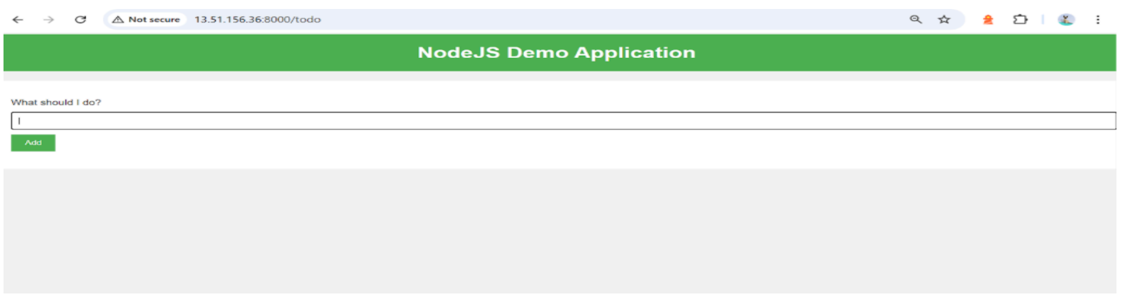
```
docker build . -t nodejs-app

docker run -d --name nodejs-app -p 8000:8000 nodejs-app
```

- 📊 **Jenkins Build Result**



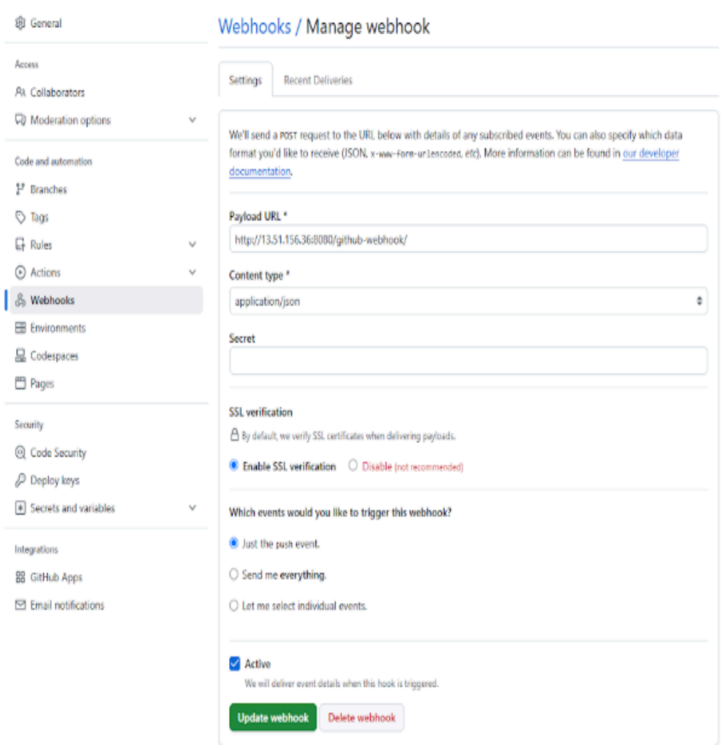
```
Started by user Yunus Khan
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/NodeJs-App
The recommended git tool is: NONE
No credentials specified
 > git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/NodeJs-App/.git # timeout=10
Fetching changes from the remote Git repository
 > git config remote.origin.url https://github.com/yunuskhan1007/Jenkins-Project.git # timeout=10
Fetching upstream changes from https://github.com/yunuskhan1007/Jenkins-Project.git
 > git --version # timeout=10
 > git --version # 'git version 2.43.0'
 > git fetch --tags --force --progress -- https://github.com/yunuskhan1007/Jenkins-Project.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision f86efc4b7d20eeaa6e4a5dada510a648070170bf (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f f86efc4b7d20eeaa6e4a5dada510a648070170bf # timeout=10
Commit message: "Merge branch 'master' of github.com:yunuskhan1007/Jenkins-Project"
 > git rev-list --no-walk f86efc4b7d20eeaa6e4a5dada510a648070170bf # timeout=10
[NodeJs-App] $ /bin/sh -xe /tmp/jenkins15418540080587066420.sh
+ docker build . -t nodejs-app
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
```

- 🌐 **Access Your Application**



🔗 Step6: Setup GitHub Webhook for Automation

- 🌐 **Create Webhook in GitHub**



Webhooks / Manage webhook

We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, x-www-form-urlencoded, etc). More information can be found in our developer documentation.

Payload URL *
http://13.51.156.36:8080/github-webhook/

Content type *
application/json

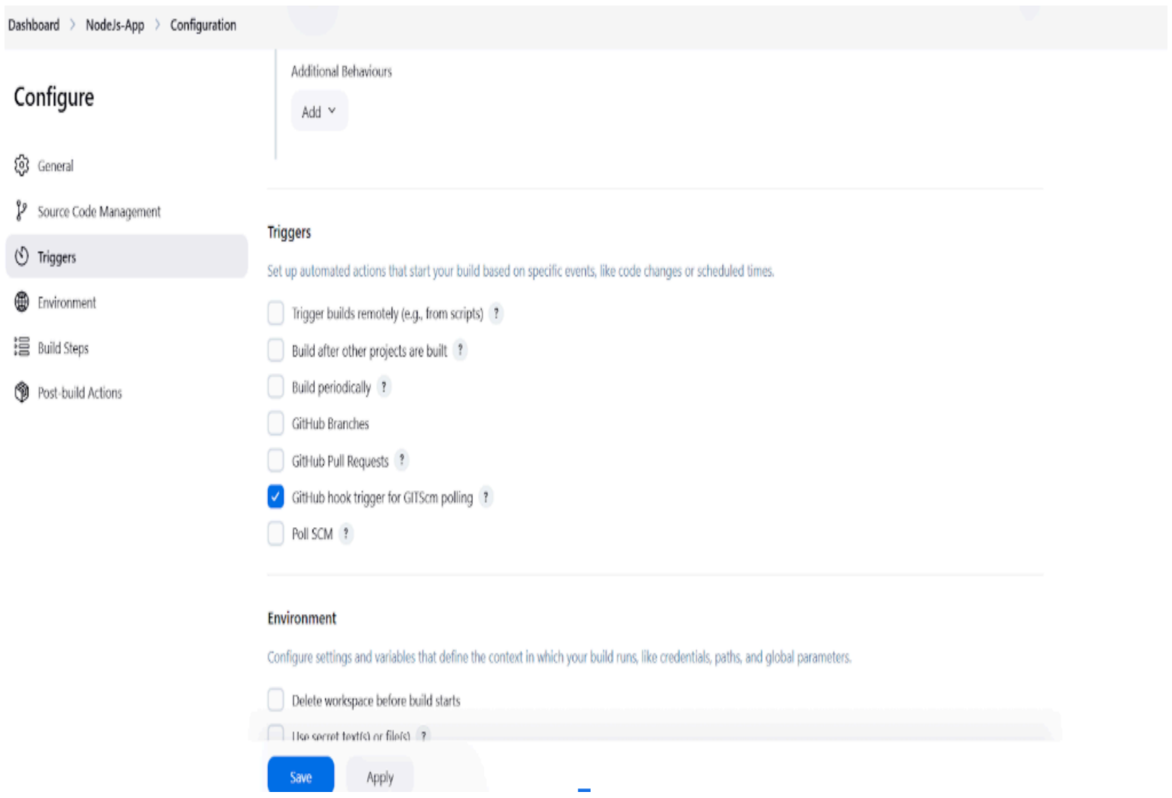
Secret

SSL verification
By default, we verify SSL certificates when delivering payloads.
[✓ Enable SSL verification] Disable (not recommended)

Which events would you like to trigger this webhook?
[✓ Just the push event.]
Send me everything.
Let me select individual events.

[✓ Active]
We will deliver event details when this hook is triggered.

- ✅ **Configure Jenkins Project**



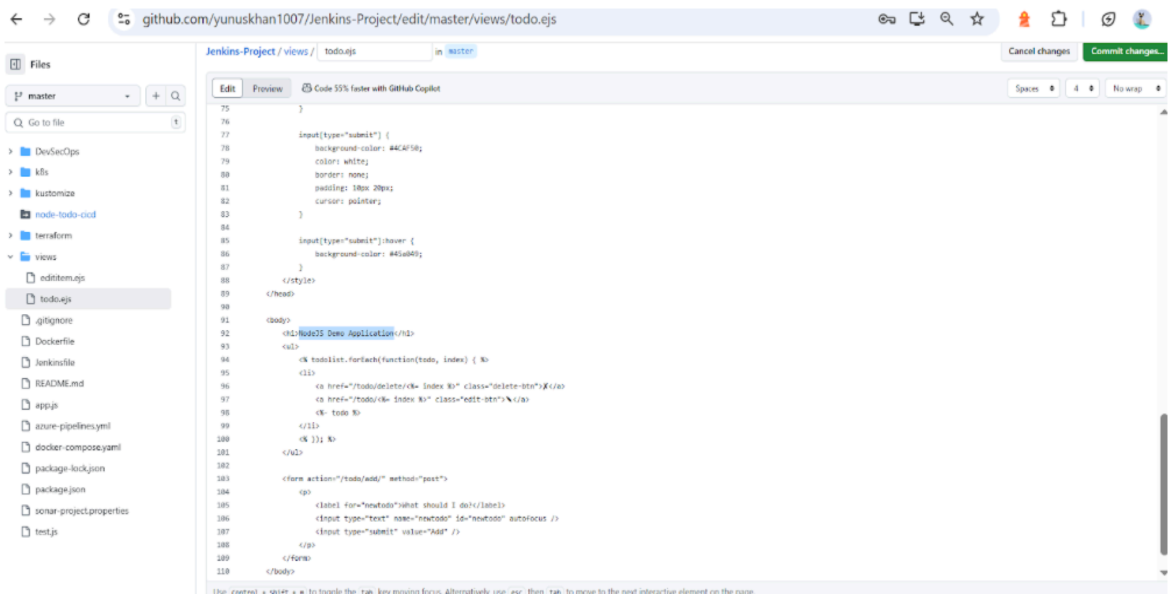
Triggers
Set up automated actions that start your build based on specific events, like code changes or scheduled times.

Trigger builds remotely (e.g., from scripts)
Build after other projects are built
Build periodically
GitHub Branches
GitHub Pull Requests
[✓ GitHub hook trigger for GITScm polling]
Poll SCM

Environment
Configure settings and variables that define the context in which your build runs, like credentials, paths, and global parameters.

Delete workspace before build starts
Use secret text(s) or file(s)

- 🧪 **Test the Webhook**

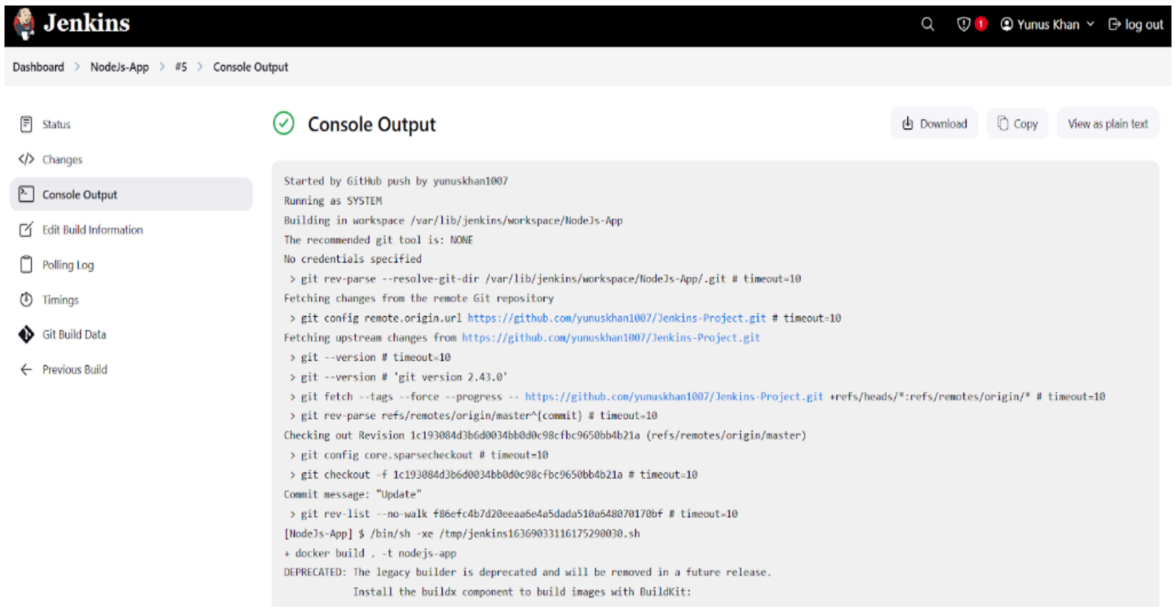


github.com/yunuskhan1007/Jenkins-Project/edit/master/views/todo.ejs

Jenkins-Project / views / todo.ejs in master

```
            }

            input[type="submit"] {
                background-color: #4CAF50;
                color: white;
                border: none;
                padding: 10px 20px;
                cursor: pointer;
            }

            input[type="submit"]:hover {
                background-color: #45a049;
            }
        </style>
    </head>

    <body>
        <h1>NodeJS Demo Application</h1>
        <ul>
            <% todolist.forEach(function(todo, index) { %>
            <li>
                <a href="/todo/delete/<%= index %>" class="delete-btn">✗</a>
                <a href="/todo/<%= index %>" class="edit-btn">✎</a>
                <%- todo %>
            </li>
            <% }); %>
        </ul>

        <form action="/todo/add/" method="post">
            <p>
                <label for="newtodo">What should I do?</label>
                <input type="text" name="newtodo" id="newtodo" autofocus />
                <input type="submit" value="Add" />
            </p>
        </form>
    </body>
```

- 📦 **Jenkins Build Output**

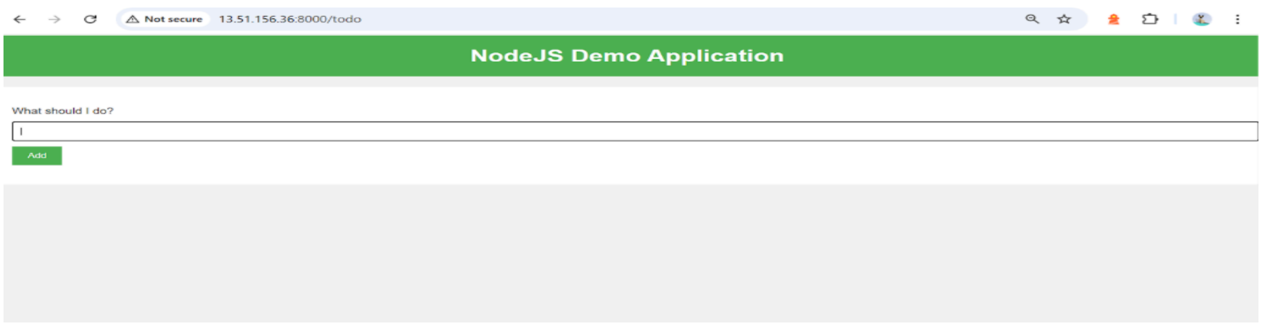


Dashboard > NodeJs-App > #5 > Console Output

Console Output

```
Started by GitHub push by yunuskhan1007
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/NodeJs-App
The recommended git tool is: NONE
No credentials specified
 > git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/NodeJs-App/.git # timeout=10
Fetching changes from the remote Git repository
 > git config remote.origin.url https://github.com/yunuskhan1007/Jenkins-Project.git # timeout=10
Fetching upstream changes from https://github.com/yunuskhan1007/Jenkins-Project.git
 > git --version # timeout=10
 > git --version # 'git version 2.43.0'
 > git fetch --tags --force --progress -- https://github.com/yunuskhan1007/Jenkins-Project.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 1c193084d3b6d0034bb0d0c98cfbc9650bb4b21a (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 1c193084d3b6d0034bb0d0c98cfbc9650bb4b21a # timeout=10
Commit message: "Update"
 > git rev-list --no-walk f86efc4b7d20eeaa6e4a5dada510a648070170bf # timeout=10
[NodeJs-App] $ /bin/sh -xe /tmp/jenkins16369033116175290030.sh
+ docker build . -t nodejs-app
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
```

- 🌍 **Access Your Application**



Not secure 13.51.156.36:8000/todo

NodeJS Demo Application

What should I do?

📘 Conclusion

You've built a complete and automated CI/CD pipeline using industry-standard tools:

- 💻 Jenkins for automation
- 🐳 Docker for deployment
- ☁️ AWS EC2 for hosting
- 🌐 GitHub Webhook for continuous integration