AutoDeploy: Dockerized Node.js Todo App CI/CD Pipeline

Aman Jhamat

Overview

Developed a fully automated CI/CD pipeline for a Node.js Todo application hosted on an EC2 instance, utilizing Jenkins, CICD, Docker, Security group.

EC2 Instance Setup: Deployed and configured an EC2 instance to serve as the environment for the Todo application.

Jenkins Integration: Installed and configured Jenkins to automate build, test, and deployment processes. Integrated Jenkins with GitHub to enable automatic code fetching and execution.

CI/CD Pipeline: Created a Jenkins pipeline that automates the continuous integration and deployment of the Node.js-based Todo application, ensuring efficient and error-free updates.

Dockerization: Containerized the application using Docker, enabling consistent deployment across different environments and simplifying scalability.

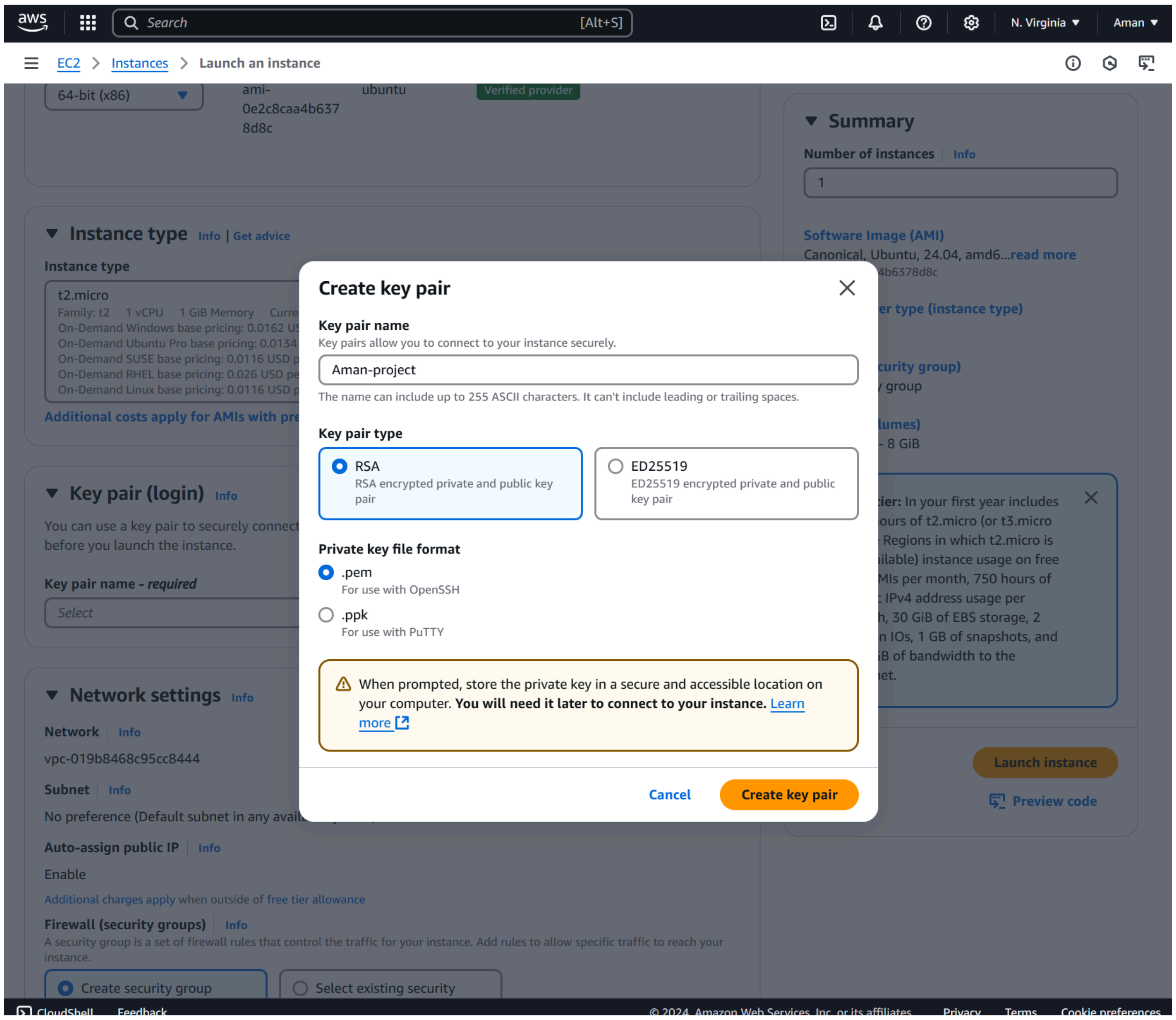
Automated Deployment: Established a seamless deployment process through Jenkins, automatically building Docker images and running containers, ensuring the application is always up-to-date.

GitHub Webhook Setup: Configured GitHub webhooks to trigger automatic deployments in Jenkins whenever changes are pushed to the repository, streamlining the update process.

Outcome: Successfully implemented a continuous deployment pipeline that allows for automatic application updates, with changes being deployed to the live application instantly upon code modification.

1) Creating EC2 Instance and Key Pair:

The first step in setting up the project was to launch an EC2 instance. After selecting the appropriate instance type, proceeded to create a key pair for secure login. Depending on the operating system, I chose either a .pem file for Linux or a .ppk file for Windows to enable SSH access.



64-bit (x86) | ami-0e2c8caa4b6378d8c | ubuntu | Verified provider

Instance type Info | Get advice

Instance type
t2.micro
Family: t2 1 vCPU 1 GiB Memory Curre
On-Demand Windows base pricing: 0.0162 US
On-Demand Ubuntu Pro base pricing: 0.0134
On-Demand SUSE base pricing: 0.0116 USD p
On-Demand RHEL base pricing: 0.026 USD pe
On-Demand Linux base pricing: 0.0116 USD p
Additional costs apply for AMIs with pre

Key pair (login) Info

You can use a key pair to securely connect
before you launch the instance.

Key pair name - *required*
Select

Network settings Info

Network Info
vpc-019b8468c95cc8444
Subnet Info
No preference (Default subnet in any avail
Auto-assign public IP Info
Enable
Additional charges apply when outside of free tier allowance
Firewall (security groups) Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.
● Create security group ○ Select existing security

Summary

Number of instances Info
1
Software Image (AMI)
Canonical, Ubuntu, 24.04, amd6...read more
4b6378d8c
er type (instance type)
curity group)
group
lumes)
- 8 GiB
ier: In your first year includes
ours of t2.micro (or t3.micro
Regions in which t2.micro is
ilable) instance usage on free
MIs per month, 750 hours of
IPv4 address usage per
h, 30 GiB of EBS storage, 2
n IOs, 1 GB of snapshots, and
B of bandwidth to the
et.

Launch instance
Preview code

Create key pair

Key pair name
Key pairs allow you to connect to your instance securely.
Aman-project
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

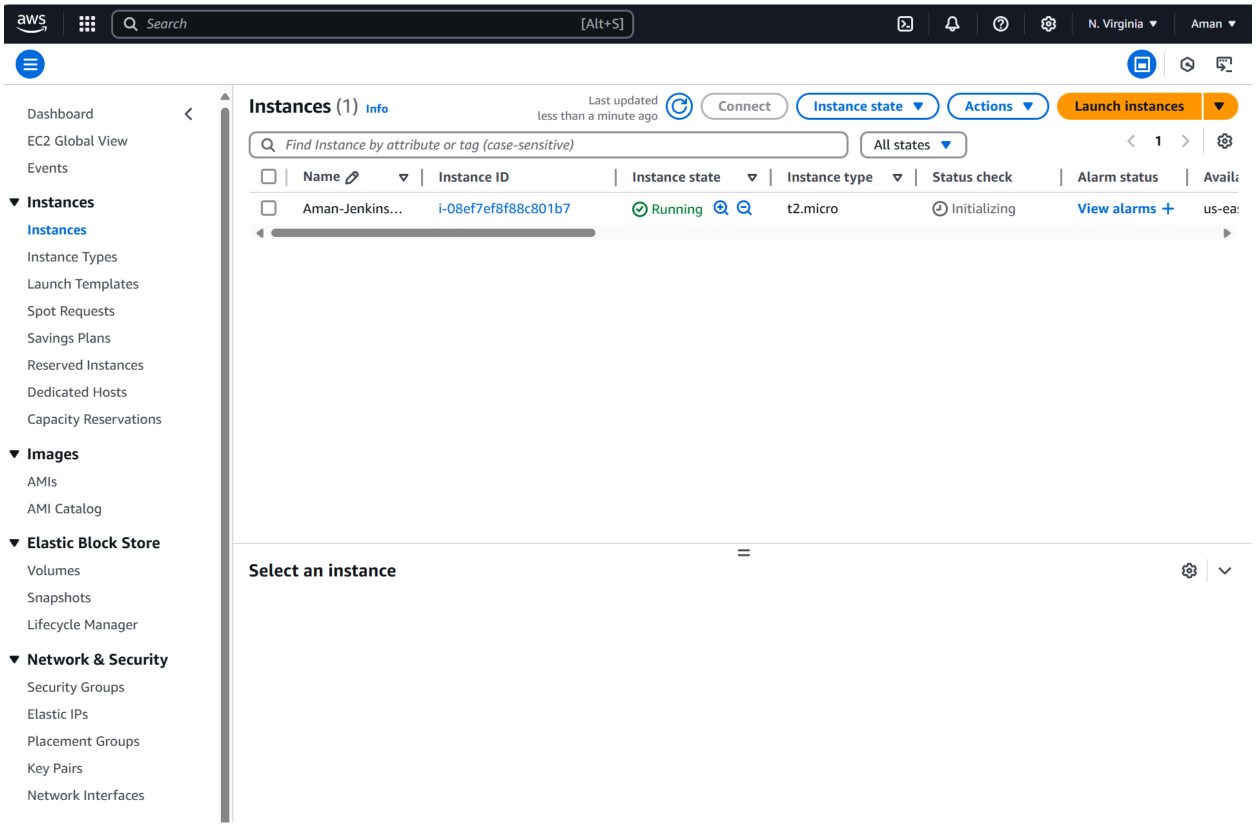
When prompted, store the private key in a secure and accessible location on your computer. **You will need it later to connect to your instance.** Learn more

Cancel | Create key pair

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

2) Successfully Installed EC2 Instance:

At this stage, I successfully launched the EC2 instance, which will serve as the host for deploying my application.



Instances (1) Info

Last updated
less than a minute ago

Find Instance by attribute or tag (case-sensitive)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availa
Aman-Jenkins…	i-08ef7ef8f88c801b7	Running	t2.micro	Initializing	View alarms +	us-ea:

Select an instance

3) Installing Jenkins on EC2:

To begin installing Jenkins, I first ensured that the Linux server was up to date by running the following command:

```
sudo apt update
```

- Since Jenkins is built using Java, I installed Java using the following command:

```
sudo apt install openjdk-17-jre
```

- Next, I added the Jenkins repository to the system's package list and installed Jenkins:

```
curl -fsSL https://pkg.jenkins.io/debian/jenkins.io.key | sudo tee
/usr/share/keyrings/jenkins-keyring.asc > /dev/null

echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] https://pkg.jenkins.io/debian
binary/ | sudo tee /etc/apt/sources.list.d/jenkins.list > /dev/null

sudo apt-get update

sudo apt-get install jenkins
```

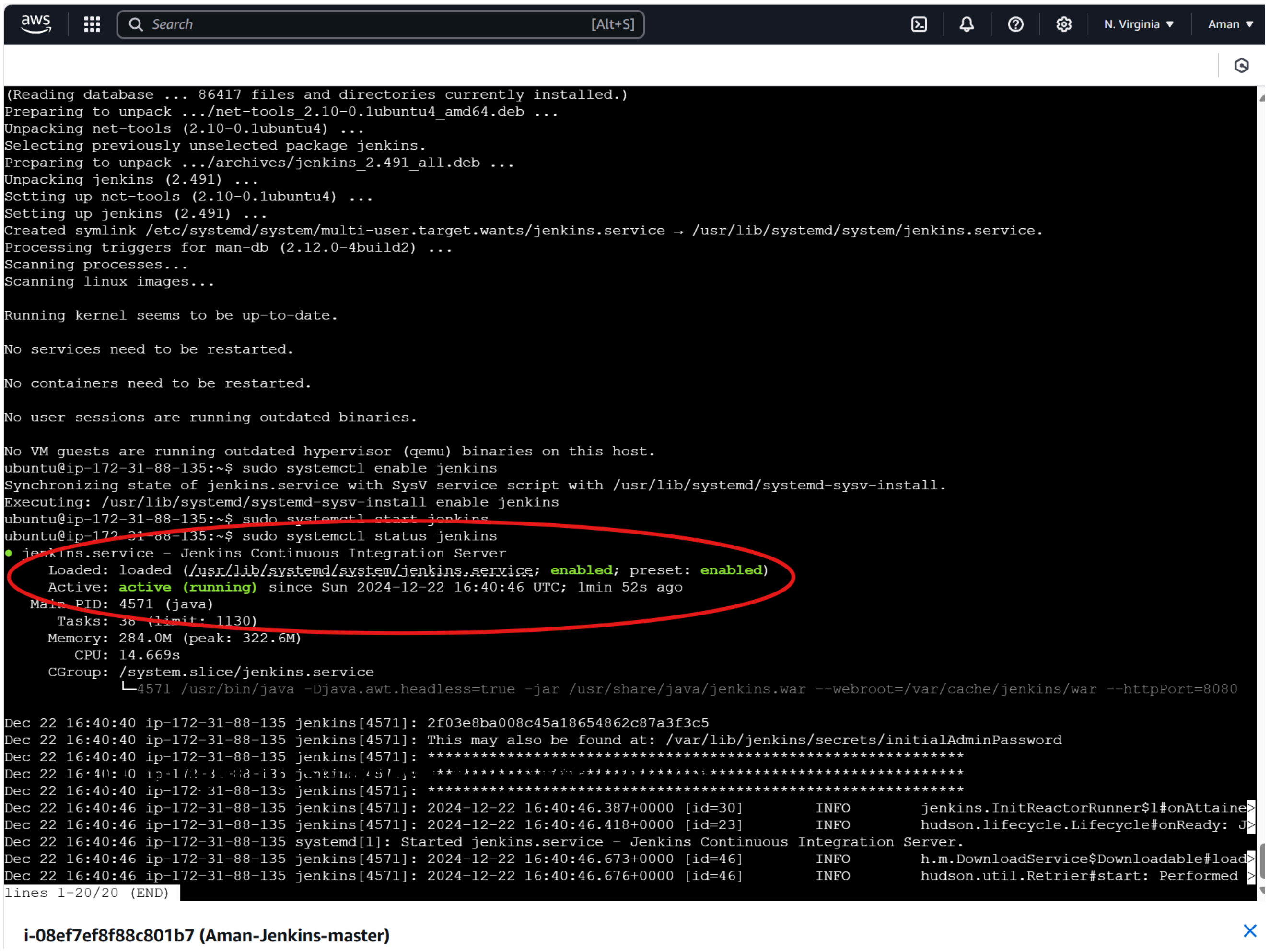
- I then enabled and started Jenkins to ensure it runs as a service:

```
sudo systemctl enable jenkins

sudo systemctl start jenkins
```

- Finally, I checked Jenkins' status to ensure it was running correctly. To retrieve the Jenkins initial admin password, I used the following command:

sudo systemctl status jenkins

sudo cat /var/lib/jenkins/secrets/initialAdminPassword

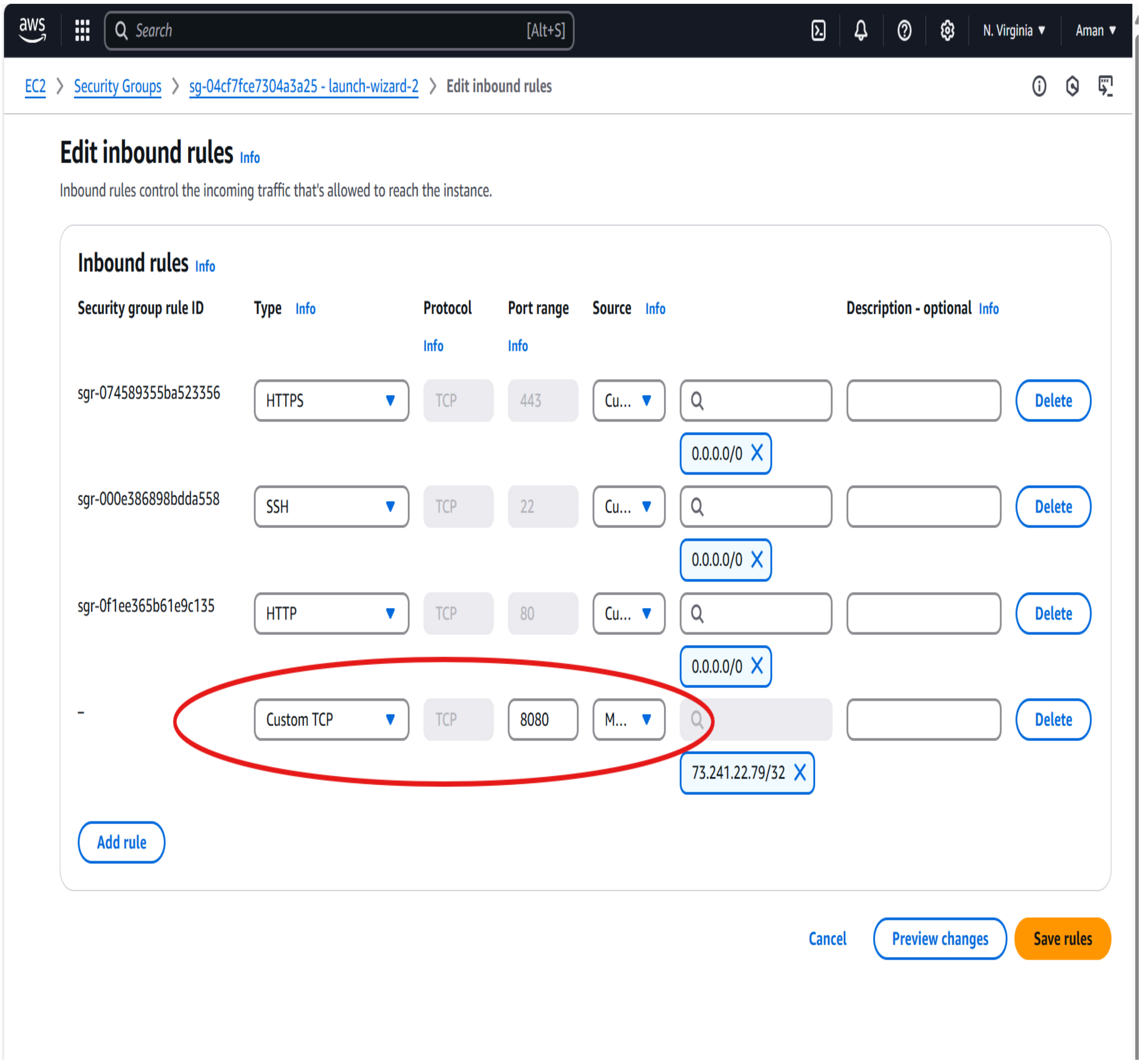


```
(Reading database ... 86417 files and directories currently installed.)
Preparing to unpack .../net-tools_2.10-0.1ubuntu4_amd64.deb ...
Unpacking net-tools (2.10-0.1ubuntu4) ...
Selecting previously unselected package jenkins.
Preparing to unpack .../archives/jenkins_2.491_all.deb ...
Unpacking jenkins (2.491) ...
Setting up net-tools (2.10-0.1ubuntu4) ...
Setting up jenkins (2.491) ...
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /usr/lib/systemd/system/jenkins.service.
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-172-31-88-135:~$ 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-172-31-88-135:~$ sudo systemctl start jenkins
ubuntu@ip-172-31-88-135:~$ 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 Sun 2024-12-22 16:40:46 UTC; 1min 52s ago
   Main PID: 4571 (java)
      Tasks: 38 (limit: 1130)
     Memory: 284.0M (peak: 322.6M)
        CPU: 14.669s
     CGroup: /system.slice/jenkins.service
             └─4571 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080

Dec 22 16:40:40 ip-172-31-88-135 jenkins[4571]: 2f03e8ba008c45a18654862c87a3f3c5
Dec 22 16:40:40 ip-172-31-88-135 jenkins[4571]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Dec 22 16:40:40 ip-172-31-88-135 jenkins[4571]: *************************************************************
Dec 22 16:40:40 ip-172-31-88-135 jenkins[4571]: *************************************************************
Dec 22 16:40:40 ip-172-31-88-135 jenkins[4571]: *************************************************************
Dec 22 16:40:46 ip-172-31-88-135 jenkins[4571]: 2024-12-22 16:40:46.387+0000 [id=30]        INFO        jenkins.InitReactorRunner$1#onAttaine>
Dec 22 16:40:46 ip-172-31-88-135 jenkins[4571]: 2024-12-22 16:40:46.418+0000 [id=23]        INFO        hudson.lifecycle.Lifecycle#onReady: J>
Dec 22 16:40:46 ip-172-31-88-135 systemd[1]: Started jenkins.service - Jenkins Continuous Integration Server.
Dec 22 16:40:46 ip-172-31-88-135 jenkins[4571]: 2024-12-22 16:40:46.673+0000 [id=46]        INFO        h.m.DownloadService$Downloadable#load>
Dec 22 16:40:46 ip-172-31-88-135 jenkins[4571]: 2024-12-22 16:40:46.676+0000 [id=46]        INFO        hudson.util.Retrier#start: Performed >
lines 1-20/20 (END)
```

i-08ef7ef8f88c801b7 (Aman-Jenkins-master)

4) **Configuring Security Groups for Jenkins:**

Jenkins runs on port 8080, I updated the EC2 security group to allow inbound traffic on port 8080 to ensure access to Jenkins from a browser.



Search [Alt+S]
N. Virginia
Aman
EC2 > Security Groups > sg-04cf7fce7304a3a25 - launch-wizard-2 > Edit inbound rules

Edit inbound rules Info

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

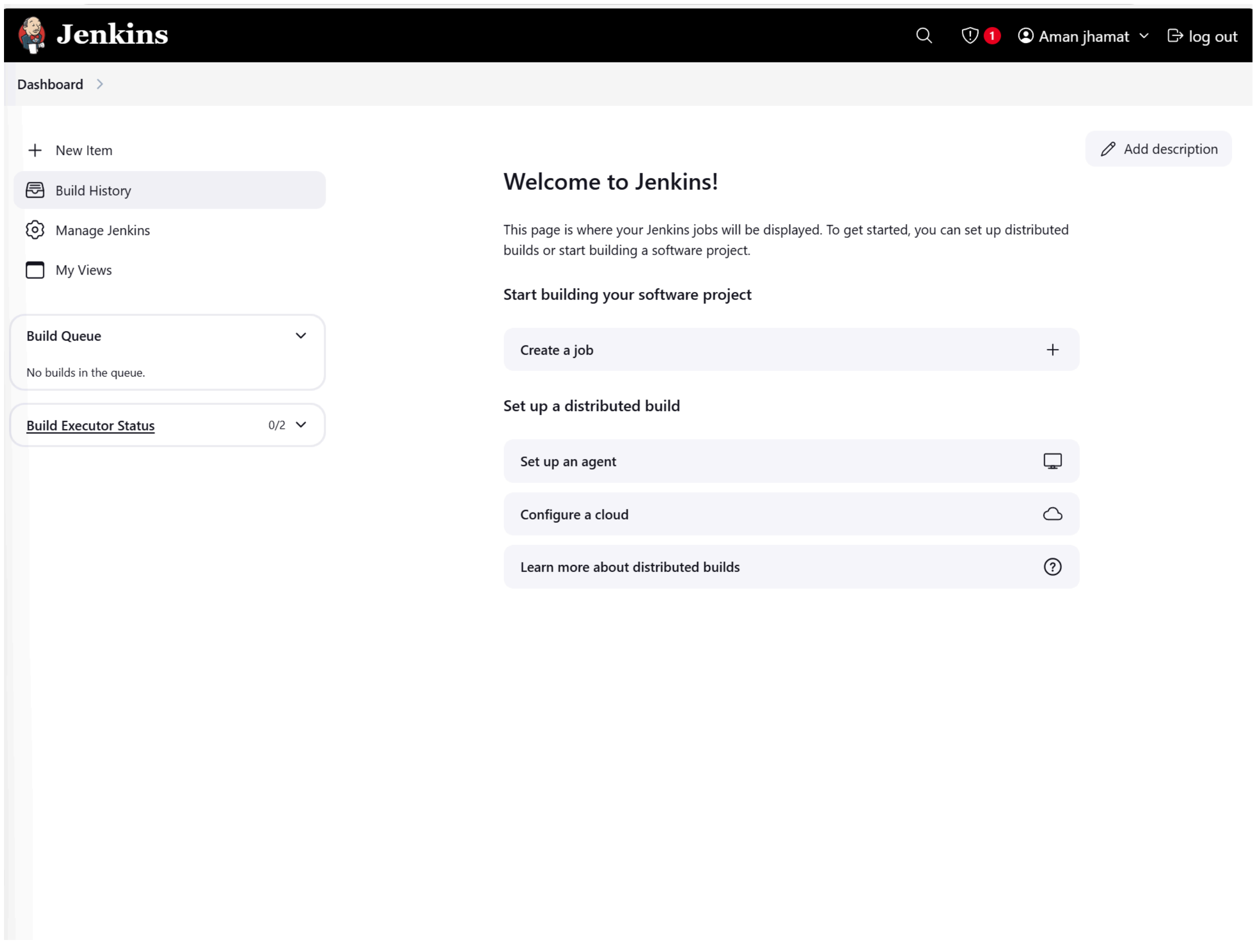
Inbound rules Info

Security group rule ID	Type	Protocol	Port range	Source		Description - optional	
sgr-074589355ba523356	HTTPS	TCP	443	Cu...	0.0.0.0/0		Delete
sgr-000e386898bdda558	SSH	TCP	22	Cu...	0.0.0.0/0		Delete
sgr-0f1ee365b61e9c135	HTTP	TCP	80	Cu...	0.0.0.0/0		Delete
-	Custom TCP	TCP	8080	M...	73.241.22.79/32		Delete

5) Accessing Jenkins with Password Key:

Upon accessing Jenkins at http://<your-ec2-ip>:8080, I was prompted to input the admin password. This key could be retrieved using the command:

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```



No builds in the queue.

Welcome to Jenkins!

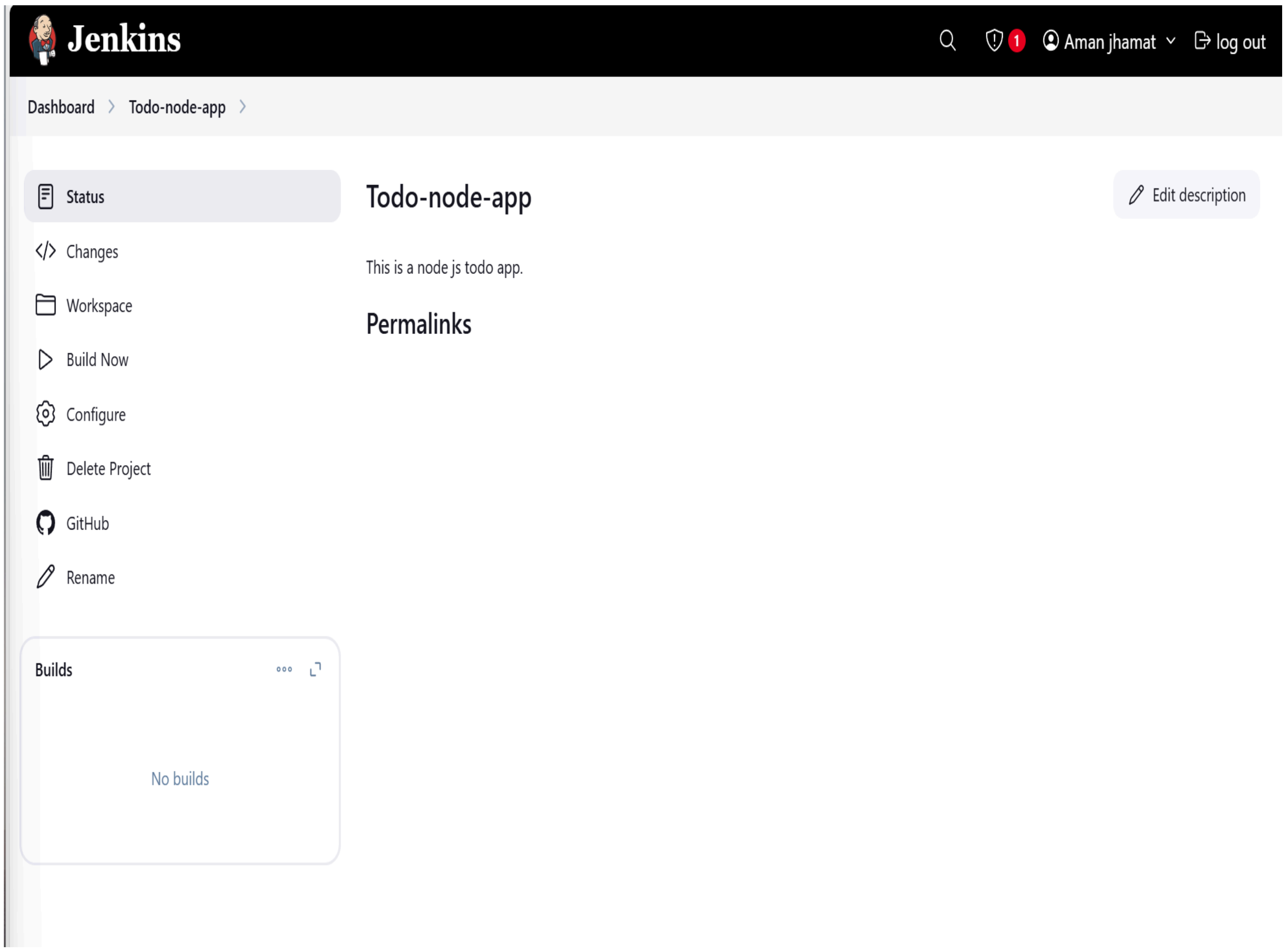
This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Set up a distributed build

6) Connecting GitHub to Jenkins:

I configured Jenkins to connect with the GitHub repository, enabling Jenkins to automatically pull the latest code from GitHub as needed for continuous integration.



Todo-node-app

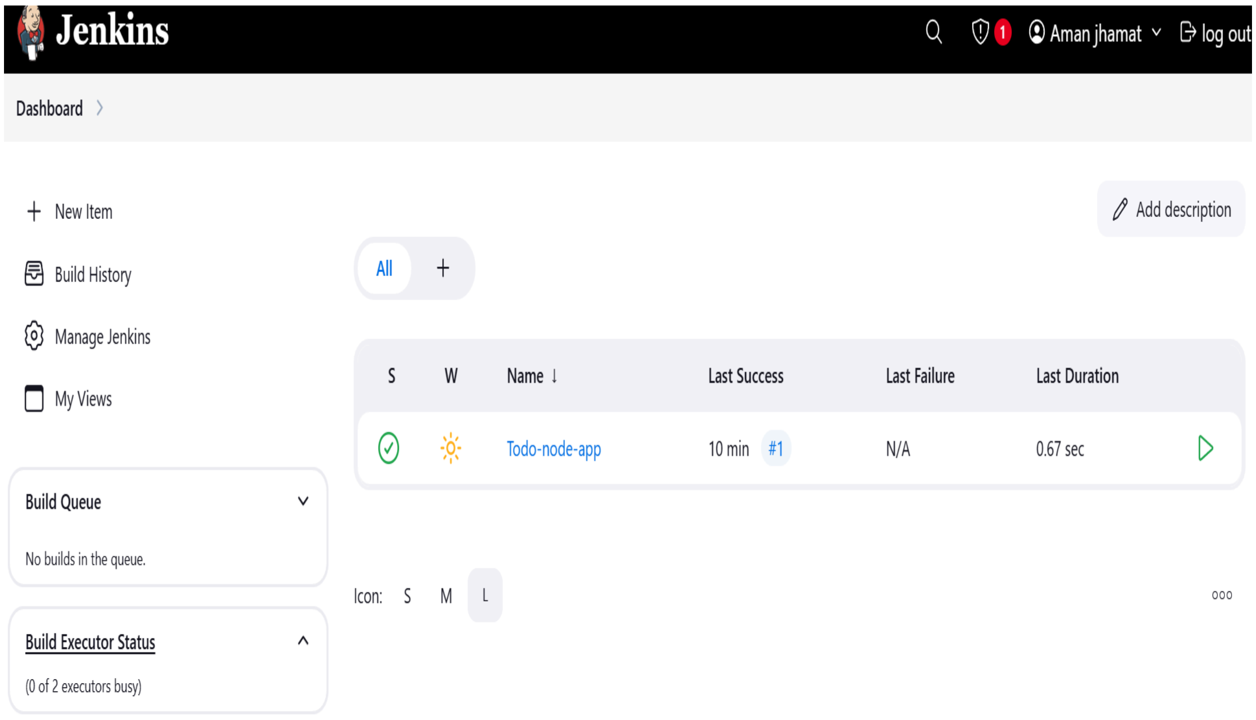
This is a node js todo app.

Permalinks

No builds

7) Creating a Jenkins CI/CD Pipeline:

Created a CI/CD pipeline in Jenkins, which is linked to the GitHub repository. By clicking the "Build" button, Jenkins initiated the process of fetching the latest code from GitHub and executing the necessary build, test, or deployment tasks as defined in the pipeline configuration.



No builds in the queue.

(0 of 2 executors busy)

S	W	Name ↓	Last Success	Last Failure	Last Duration
✓	☼	Todo-node-app	10 min #1	N/A	0.67 sec

8) Installing Node.js for Node.js Application:

Since the application is built with Node.js, I installed Node.js using the following command:

```
sudo apt install nodejs
```

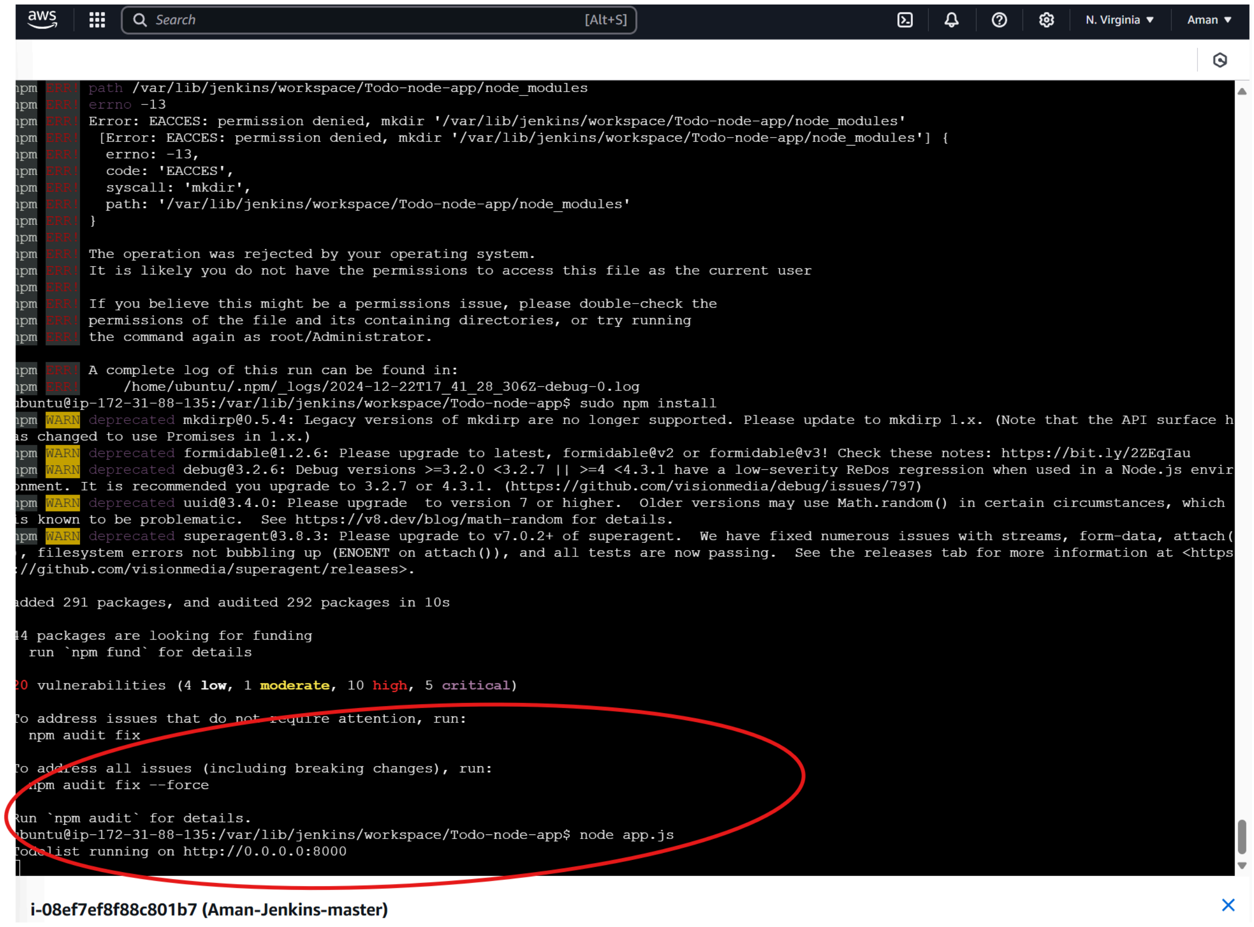
I then installed the Node Package Manager (npm) to manage the project dependencies:

```
sudo apt install npm
```

9) Running the Node.js Application:

I successfully ran the uploaded Node.js application using the command:

```
node app.js
```

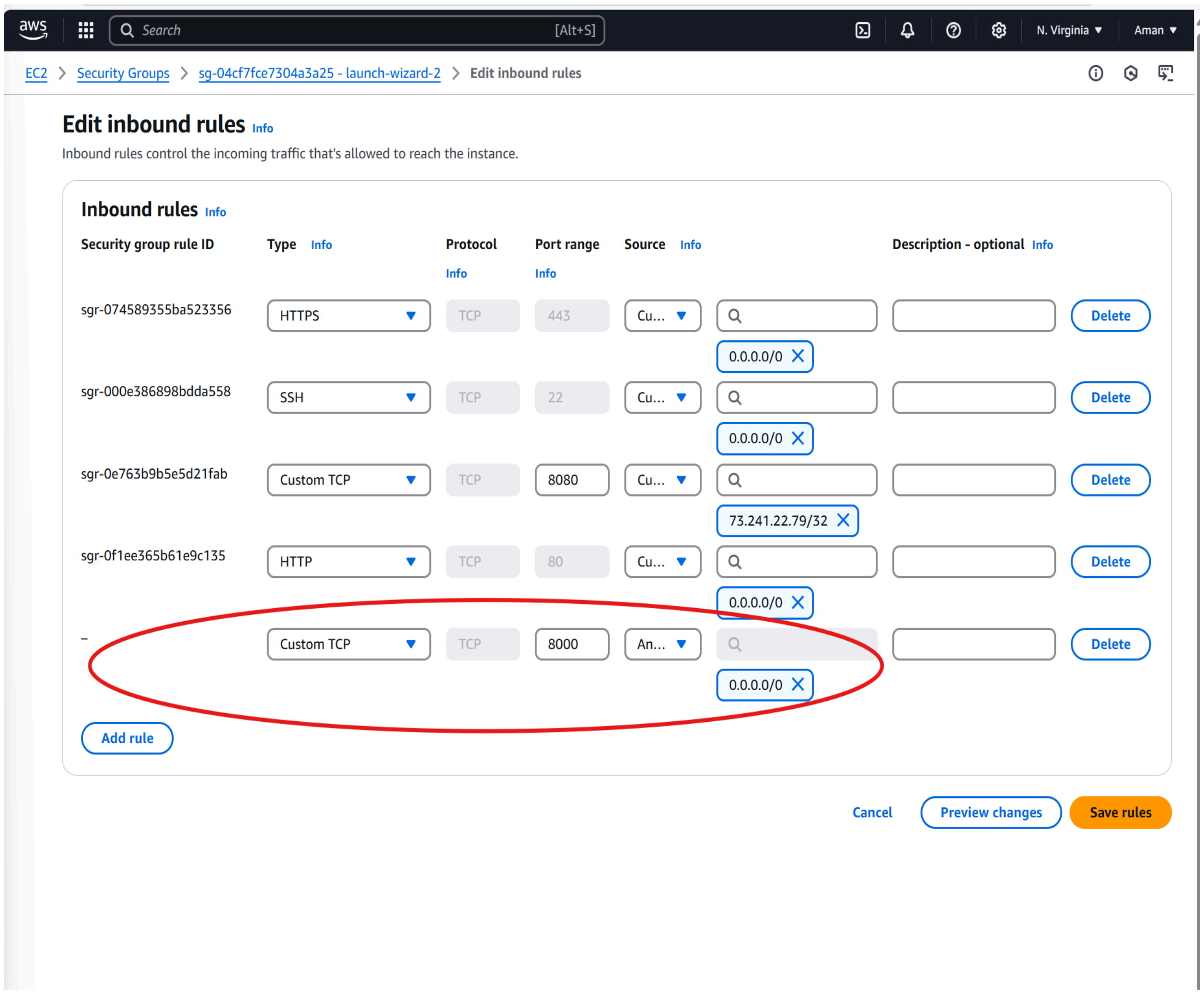


```
npm ERR! path /var/lib/jenkins/workspace/Todo-node-app/node_modules
npm ERR! errno -13
npm ERR! Error: EACCES: permission denied, mkdir '/var/lib/jenkins/workspace/Todo-node-app/node_modules'
npm ERR!  [Error: EACCES: permission denied, mkdir '/var/lib/jenkins/workspace/Todo-node-app/node_modules'] {
npm ERR!   errno: -13,
npm ERR!   code: 'EACCES',
npm ERR!   syscall: 'mkdir',
npm ERR!   path: '/var/lib/jenkins/workspace/Todo-node-app/node_modules'
npm ERR! }
npm ERR!
npm ERR! The operation was rejected by your operating system.
npm ERR! It is likely you do not have the permissions to access this file as the current user
npm ERR!
npm ERR! If you believe this might be a permissions issue, please double-check the
npm ERR! permissions of the file and its containing directories, or try running
npm ERR! the command again as root/Administrator.

npm ERR! A complete log of this run can be found in:
npm ERR!     /home/ubuntu/.npm/_logs/2024-12-22T17_41_28_306Z-debug-0.log
ubuntu@ip-172-31-88-135:/var/lib/jenkins/workspace/Todo-node-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 h
as 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 envir
onment. It is recommended you upgrade 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 https://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 10s

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-172-31-88-135:/var/lib/jenkins/workspace/Todo-node-app$ node app.js
Todolist running on http://0.0.0.0:8000
```

i-08ef7ef8f88c801b7 (Aman-Jenkins-master)

10) Configuring Security Group for Application:

To allow external traffic to reach the Node.js application running on port 8000, I updated the EC2 instance's security group settings to open port 8000 for inbound traffic.



EC2 > Security Groups > sg-04cf7fce7304a3a25 - launch-wizard-2 > Edit inbound rules

Edit inbound rules Info

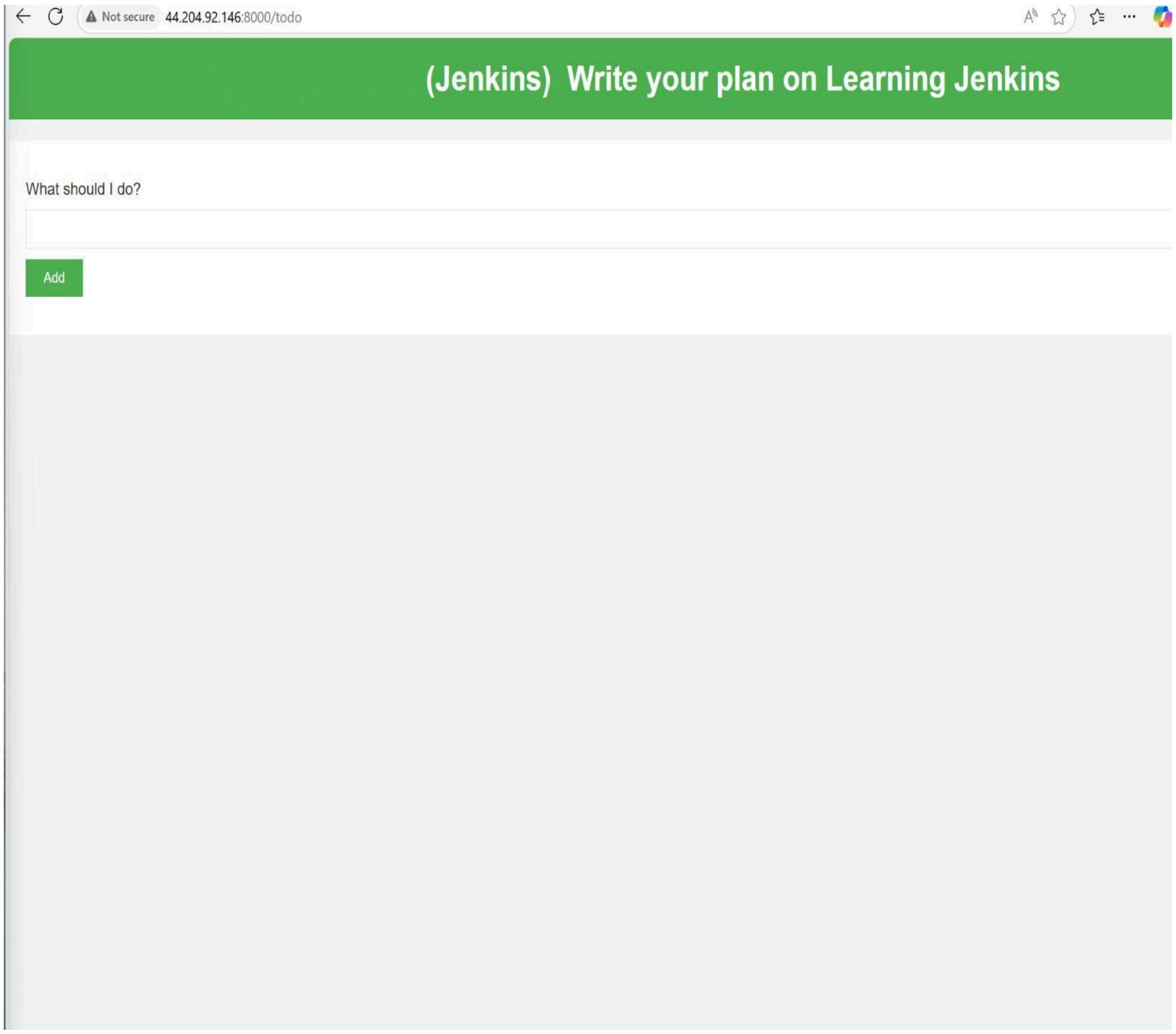
Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type	Protocol	Port range	Source		Description - optional
sgr-074589355ba523356	HTTPS	TCP	443	Cu...	0.0.0.0/0	
sgr-000e386898bdda558	SSH	TCP	22	Cu...	0.0.0.0/0	
sgr-0e763b9b5e5d21fab	Custom TCP	TCP	8080	Cu...	73.241.22.79/32	
sgr-0f1ee365b61e9c135	HTTP	TCP	80	Cu...	0.0.0.0/0	
–	Custom TCP	TCP	8000	An...	0.0.0.0/0	

11) Accessing Application in Browser:

I accessed the application in a web browser via the EC2 instance's public IP and port 8000, verifying that the application was running correctly.



12) Creating Dockerfile for Node.js Application:

To containerize the Node.js application, I created a Dockerfile that defines the setup for the application within a Docker container. The Dockerfile included the following:

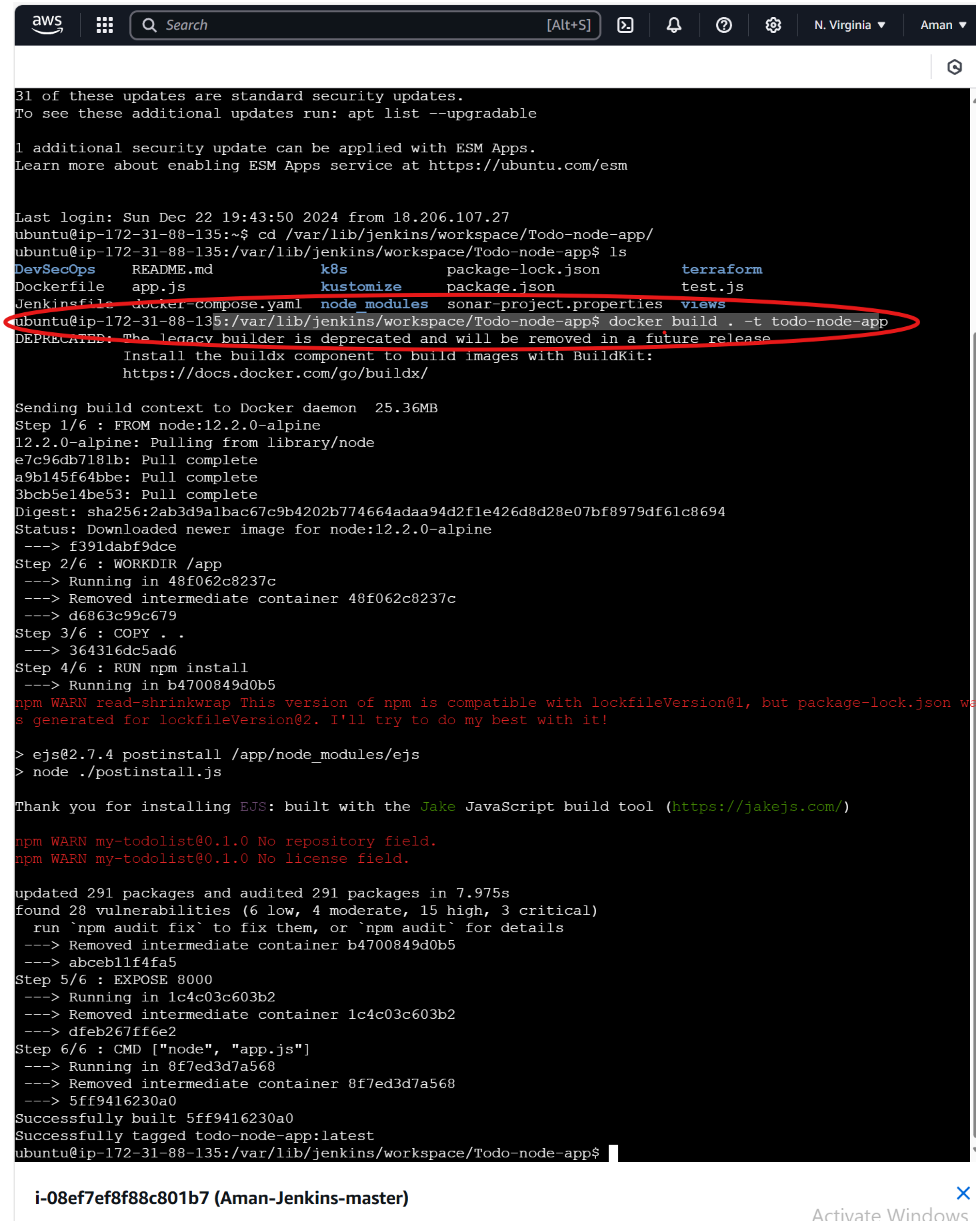
```
FROM node:12.2.0-alpine

WORKDIR /app

COPY . .

RUN npm install

EXPOSE 8000

CMD ["node", "app.js"]
```

- The Docker image was built with the command:

```
docker build -t todo-node-app .
```

- Additionally, I granted the necessary permissions to run Docker commands:

```
sudo usermod -a -G docker $USER
```



```
31 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

1 additional security update can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm


Last login: Sun Dec 22 19:43:50 2024 from 18.206.107.27
ubuntu@ip-172-31-88-135:~$ cd /var/lib/jenkins/workspace/Todo-node-app/
ubuntu@ip-172-31-88-135:/var/lib/jenkins/workspace/Todo-node-app$ ls
DevSecOps     README.md             k8s           package-lock.json         terraform
Dockerfile    app.js                kustomize     package.json              test.js
Jenkinsfile   docker-compose.yaml   node_modules  sonar-project.properties  views
ubuntu@ip-172-31-88-135:/var/lib/jenkins/workspace/Todo-node-app$ docker build . -t todo-node-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/

Sending build context to Docker daemon  25.36MB
Step 1/6 : FROM node:12.2.0-alpine
12.2.0-alpine: Pulling from library/node
e7c96db7181b: Pull complete
a9b145f64bbe: Pull complete
3bcb5e14be53: Pull complete
Digest: sha256:2ab3d9a1bac67c9b4202b774664adaa94d2f1e426d8d28e07bf8979df61c8694
Status: Downloaded newer image for node:12.2.0-alpine
 ---> f391dabf9dce
Step 2/6 : WORKDIR /app
 ---> Running in 48f062c8237c
 ---> Removed intermediate container 48f062c8237c
 ---> d6863c99c679
Step 3/6 : COPY . .
 ---> 364316dc5ad6
Step 4/6 : RUN npm install
 ---> Running in b4700849d0b5
npm WARN read-shrinkwrap This version of npm is compatible with lockfileVersion@1, but package-lock.json wa
s generated for lockfileVersion@2. I'll try to do my best with it!

> ejs@2.7.4 postinstall /app/node_modules/ejs
> node ./postinstall.js

Thank you for installing EJS: built with the Jake JavaScript build tool (https://jakejs.com/)

npm WARN my-todolist@0.1.0 No repository field.
npm WARN my-todolist@0.1.0 No license field.

updated 291 packages and audited 291 packages in 7.975s
found 28 vulnerabilities (6 low, 4 moderate, 15 high, 3 critical)
  run `npm audit fix` to fix them, or `npm audit` for details
 ---> Removed intermediate container b4700849d0b5
 ---> abceb11f4fa5
Step 5/6 : EXPOSE 8000
 ---> Running in 1c4c03c603b2
 ---> Removed intermediate container 1c4c03c603b2
 ---> dfeb267ff6e2
Step 6/6 : CMD ["node", "app.js"]
 ---> Running in 8f7ed3d7a568
 ---> Removed intermediate container 8f7ed3d7a568
 ---> 5ff9416230a0
Successfully built 5ff9416230a0
Successfully tagged todo-node-app:latest
ubuntu@ip-172-31-88-135:/var/lib/jenkins/workspace/Todo-node-app$
```

i-08ef7ef8f88c801b7 (Aman-Jenkins-master)

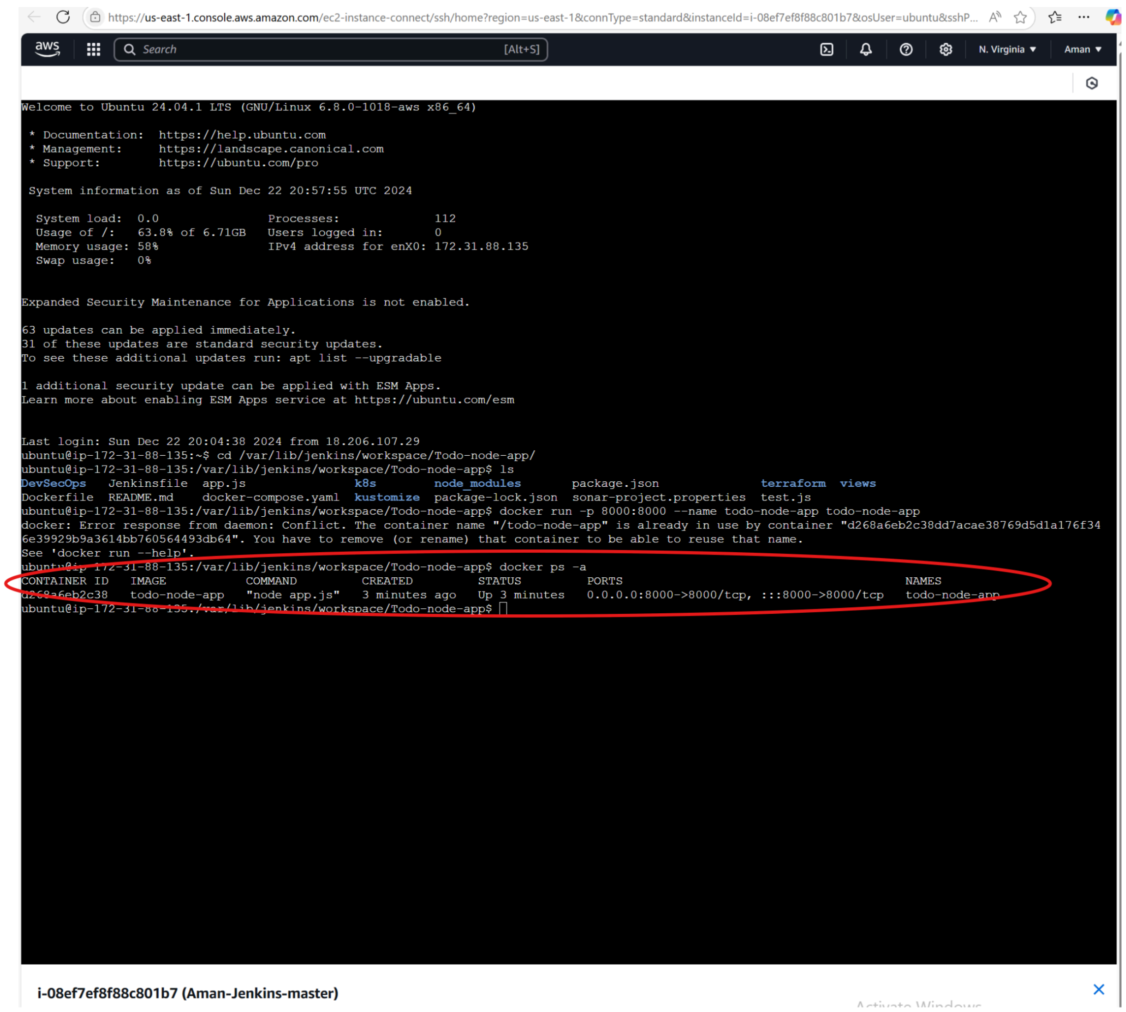
Activate Windows

13) Running the Docker Container:

After building the Docker image, I ran the container with the following command:

```
docker run -p 8000:8000 --name todo-node-app todo-node-app
```

This command maps port 8000 on the host machine to port 8000 inside the container, allowing the application to be accessed via http://localhost:8000. I used docker ps -a to verify that the container was running.


https://us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh/home?region=us-east-1&connType=standard&instanceId=i-08ef7ef8f88c801b7&osUser=ubuntu&sshP...
aws
Search [Alt+S]
N. Virginia
Aman
i-08ef7ef8f88c801b7 (Aman-Jenkins-master)

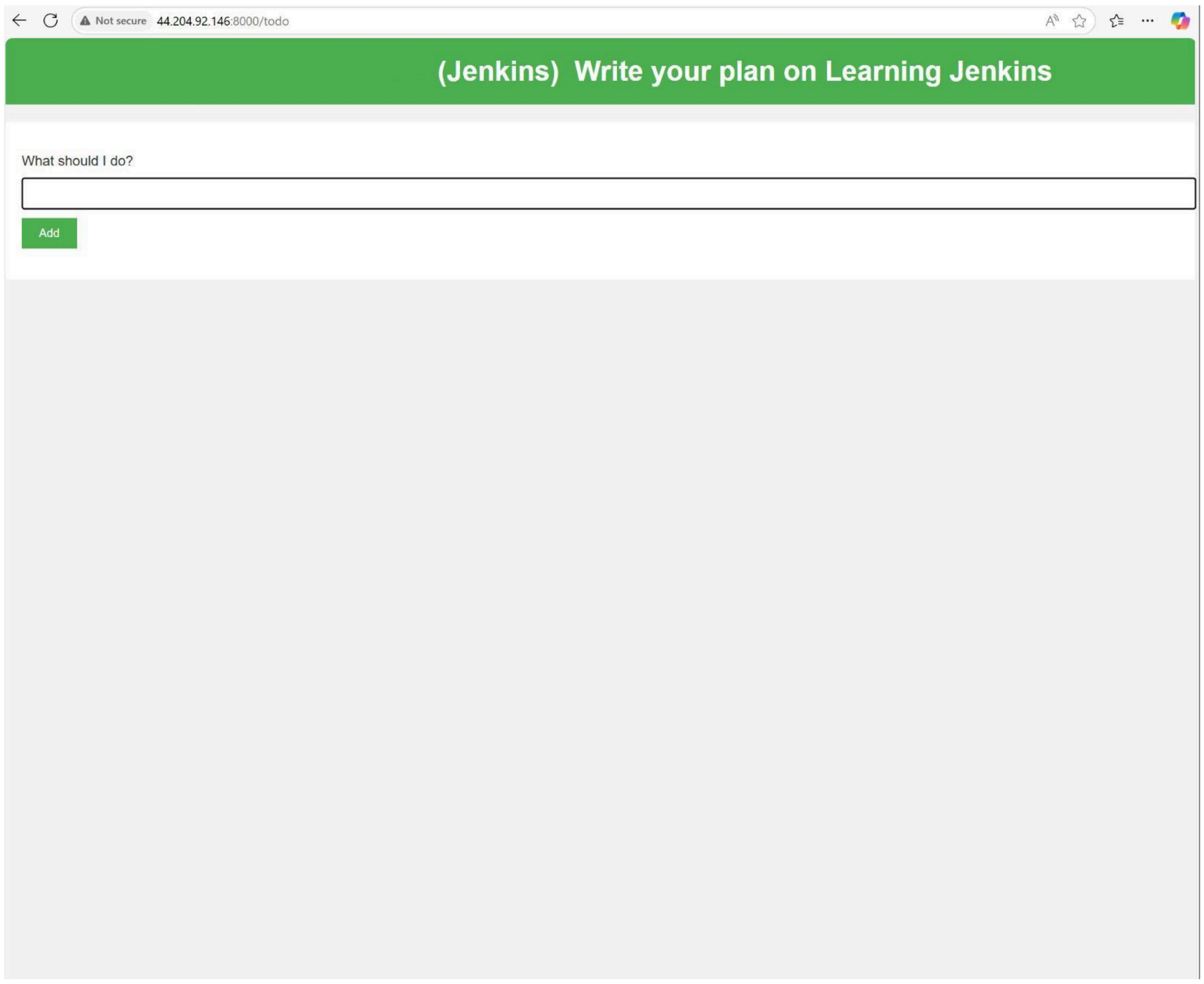
```
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1018-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Sun Dec 22 20:57:55 UTC 2024

  System load:  0.0               Processes:             112
  Usage of /:   63.8% of 6.71GB   Users logged in:       0
  Memory usage: 58%               IPv4 address for enX0: 172.31.88.135
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

63 updates can be applied immediately.
31 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

1 additional security update can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm


Last login: Sun Dec 22 20:04:38 2024 from 18.206.107.29
ubuntu@ip-172-31-88-135:~$ cd /var/lib/jenkins/workspace/Todo-node-app/
ubuntu@ip-172-31-88-135:/var/lib/jenkins/workspace/Todo-node-app$ ls
DevSecOps   Jenkinsfile  app.js               k8s        node_modules       package.json               terraform  views
Dockerfile  README.md    docker-compose.yaml  kustomize  package-lock.json  sonar-project.properties   test.js
ubuntu@ip-172-31-88-135:/var/lib/jenkins/workspace/Todo-node-app$ docker run -p 8000:8000 --name todo-node-app todo-node-app
docker: Error response from daemon: Conflict. The container name "/todo-node-app" is already in use by container "d268a6eb2c38dd7acae38769d5d1a176f34
6e39929b9a3614bb760564493db64". You have to remove (or rename) that container to be able to reuse that name.
See 'docker run --help'.
ubuntu@ip-172-31-88-135:/var/lib/jenkins/workspace/Todo-node-app$ docker ps -a
CONTAINER ID   IMAGE           COMMAND         CREATED         STATUS         PORTS                                       NAMES
d268a6eb2c38   todo-node-app   "node app.js"   3 minutes ago   Up 3 minutes   0.0.0.0:8000->8000/tcp, :::8000->8000/tcp   todo-node-app
ubuntu@ip-172-31-88-135:/var/lib/jenkins/workspace/Todo-node-app$
```

ng.

14) Verifying Application is Running:

To verify the application was running correctly, I accessed it via a web browser at http://localhost:8000. Once confirmed, I stopped the container with the following command:

```
docker kill <container_id>
```

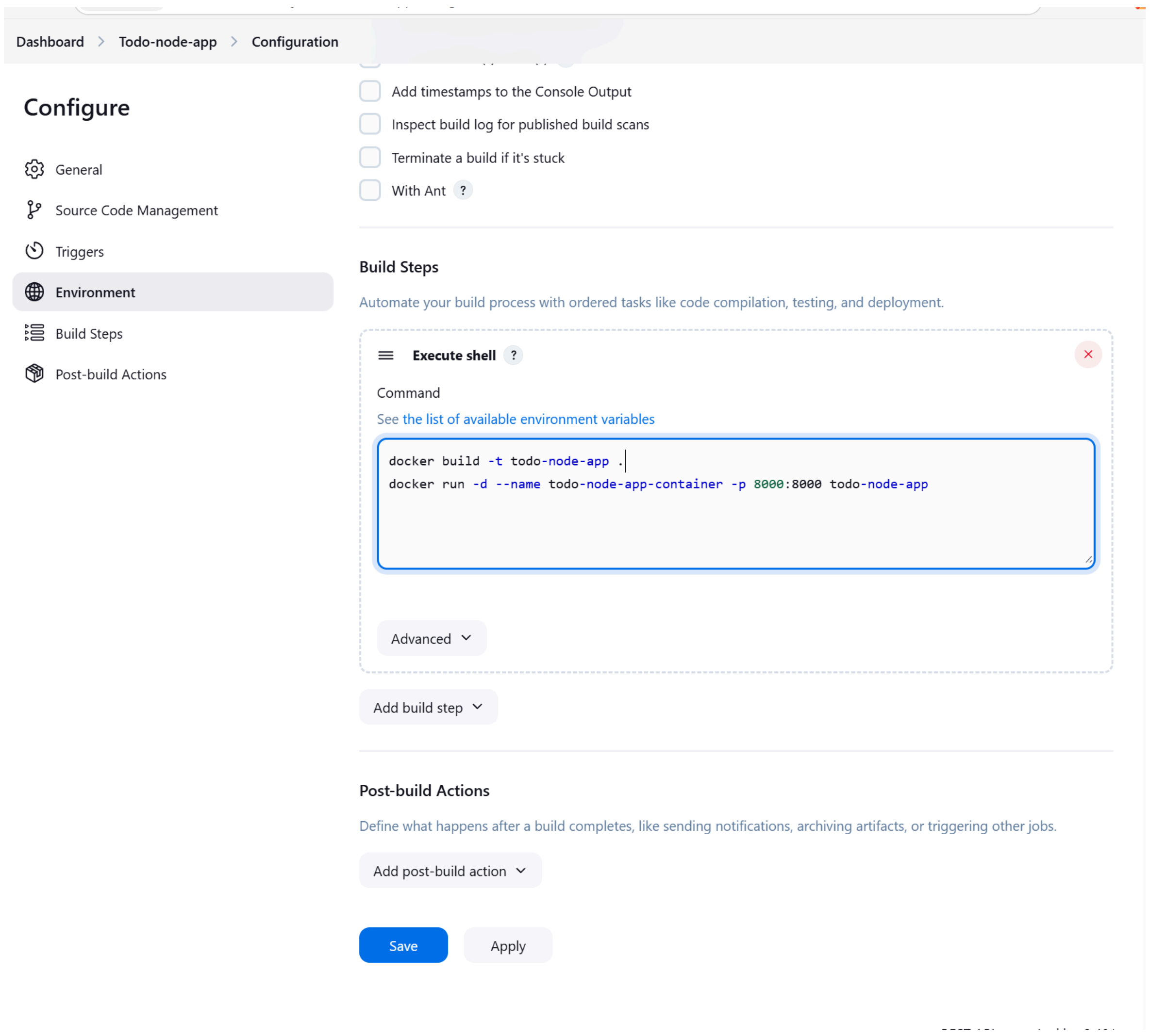

Not secure 44.204.92.146:8000/todo
(Jenkins) Write your plan on Learning Jenkins
What should I do?
Add

15) CI/CD Automation with Docker:

The Docker image was built and the container was deployed automatically as part of the CI/CD pipeline. The following commands were used to automate the process:

docker build -t todo-node-app .

docker run -d --name todo-node-app-container -p 8000:8000 todo-node-app



Add timestamps to the Console Output

Inspect build log for published build scans

Terminate a build if it's stuck

With Ant

Build Steps

Automate your build process with ordered tasks like code compilation, testing, and deployment.

Command

See the list of available environment variables

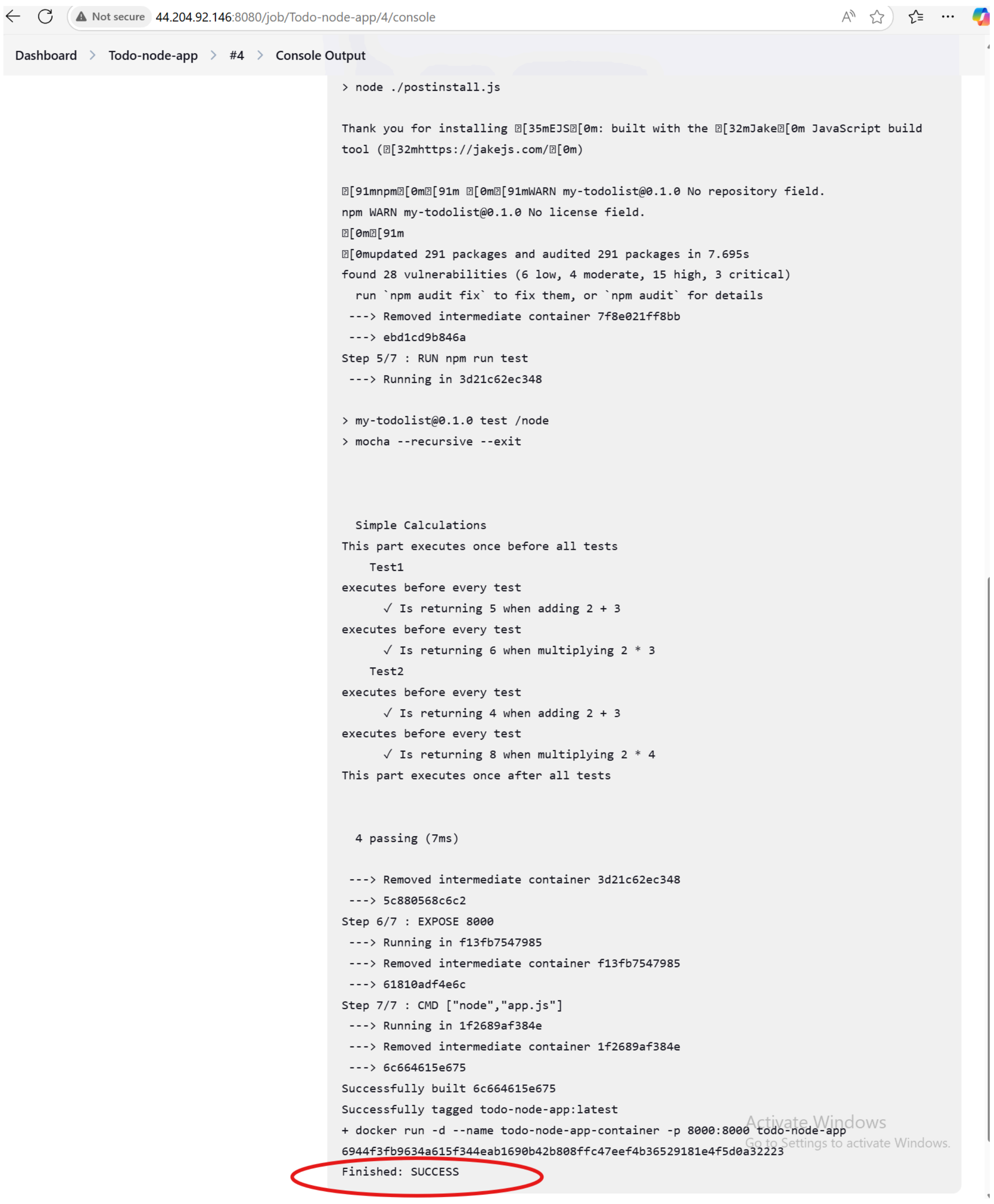
```
docker build -t todo-node-app .
docker run -d --name todo-node-app-container -p 8000:8000 todo-node-app
```

Post-build Actions

Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.

15.1) Automating the Process with Jenkins:

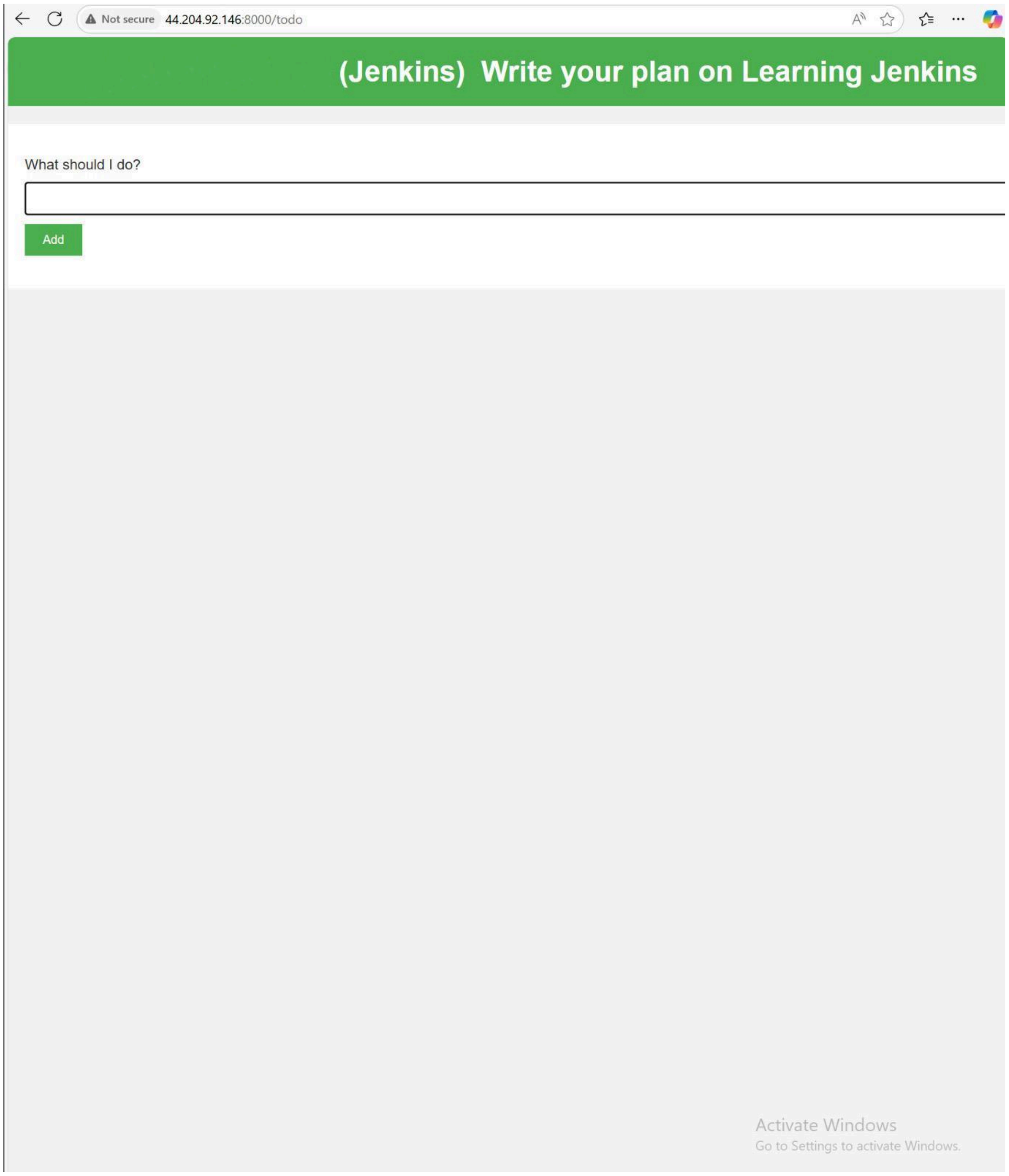
By clicking "Build Now" in Jenkins, the CI/CD pipeline was triggered, automating the entire process of building and deploying the application. The build and deployment tasks were executed automatically through the Jenkins Execute Shell build step.


Not secure
44.204.92.146:8080/job/Todo-node-app/4/console
Dashboard > Todo-node-app > #4 > Console Output

```
> node ./postinstall.js

Thank you for installing [35mEJS[0m: built with the [32mJake[0m JavaScript build
tool ([32mhttps://jakejs.com/[0m)

[91mnpm[0m[91m [0m[91mWARN my-todolist@0.1.0 No repository field.
npm WARN my-todolist@0.1.0 No license field.
[0m[91m
[0mupdated 291 packages and audited 291 packages in 7.695s
found 28 vulnerabilities (6 low, 4 moderate, 15 high, 3 critical)
  run `npm audit fix` to fix them, or `npm audit` for details
 ---> Removed intermediate container 7f8e021ff8bb
 ---> ebd1cd9b846a
Step 5/7 : RUN npm run test
 ---> Running in 3d21c62ec348

> my-todolist@0.1.0 test /node
> mocha --recursive --exit



  Simple Calculations
This part executes once before all tests
    Test1
executes before every test
      ✓ Is returning 5 when adding 2 + 3
executes before every test
      ✓ Is returning 6 when multiplying 2 * 3
    Test2
executes before every test
      ✓ Is returning 4 when adding 2 + 3
executes before every test
      ✓ Is returning 8 when multiplying 2 * 4
This part executes once after all tests


  4 passing (7ms)

 ---> Removed intermediate container 3d21c62ec348
 ---> 5c880568c6c2
Step 6/7 : EXPOSE 8000
 ---> Running in f13fb7547985
 ---> Removed intermediate container f13fb7547985
 ---> 61810adf4e6c
Step 7/7 : CMD ["node","app.js"]
 ---> Running in 1f2689af384e
 ---> Removed intermediate container 1f2689af384e
 ---> 6c664615e675
Successfully built 6c664615e675
Successfully tagged todo-node-app:latest
+ docker run -d --name todo-node-app-container -p 8000:8000 todo-node-app
6944f3fb9634a615f344eab1690b42b808ffc47eef4b36529181e4f5d0a32223
Finished: SUCCESS
```

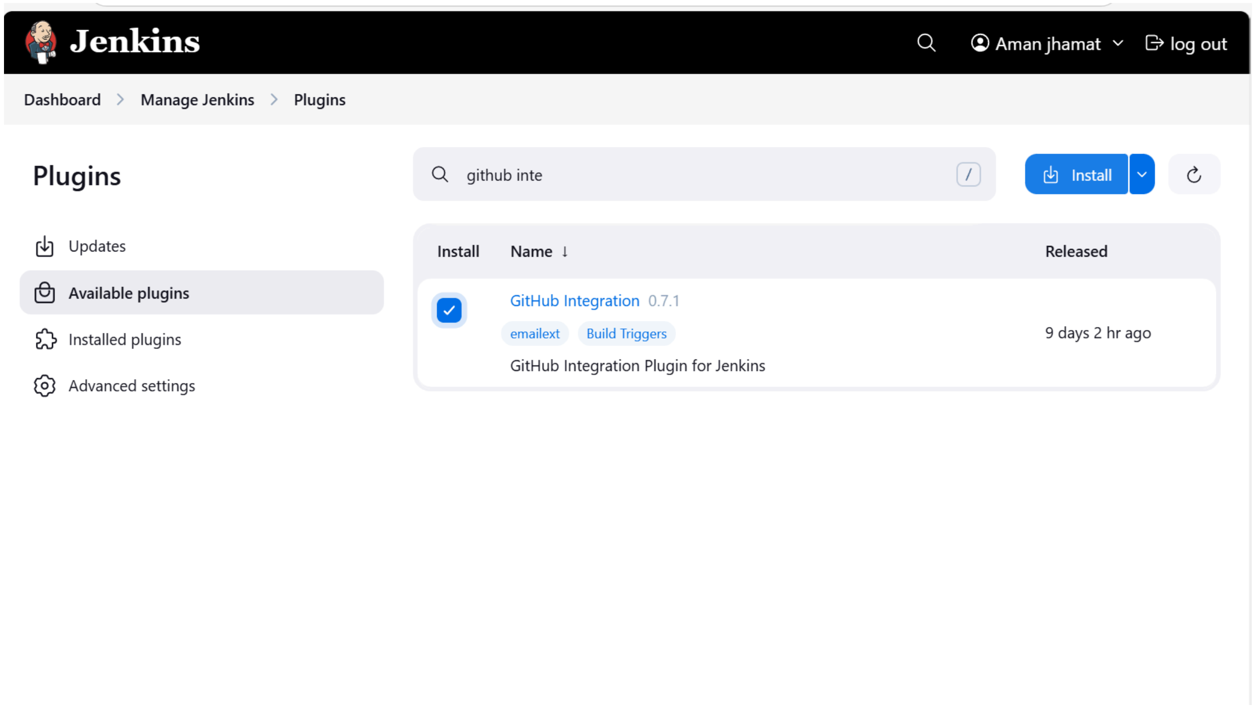
16) Confirming Successful Deployment:

After triggering the build and deployment through Jenkins, I verified that the application was accessible via http://<Jenkins_host_ip>:8000, confirming a successful deployment.


Not secure
44.204.92.146:8000/todo
(Jenkins) Write your plan on Learning Jenkins
What should I do?
Add
Activate Windows
Go to Settings to activate Windows.

17) Setting Up GitHub Webhooks for Auto Deployment:

To automate the deployment whenever code changes were pushed to the GitHub repository, I set up a GitHub webhook. First, I installed the necessary plugin in Jenkins to integrate it with GitHub.

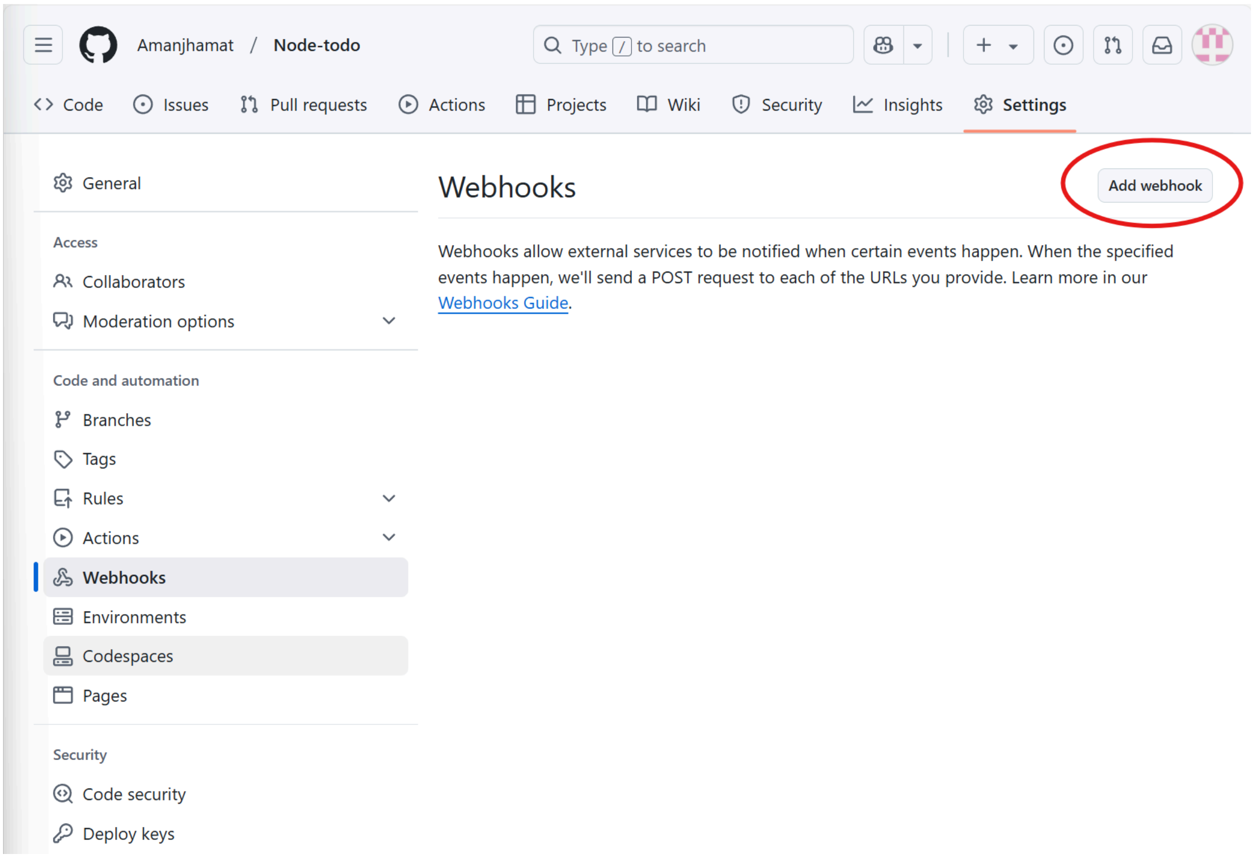


Install	Name ↓	Released
✓	GitHub Integration 0.7.1 emailext Build Triggers GitHub Integration Plugin for Jenkins	9 days 2 hr ago

17.1) Configuring GitHub Webhook:

I went to the GitHub repository settings and added a webhook, ensuring the correct port was accessible. If needed, I updated the EC2 security group to allow inbound traffic from GitHub.

GitHub.

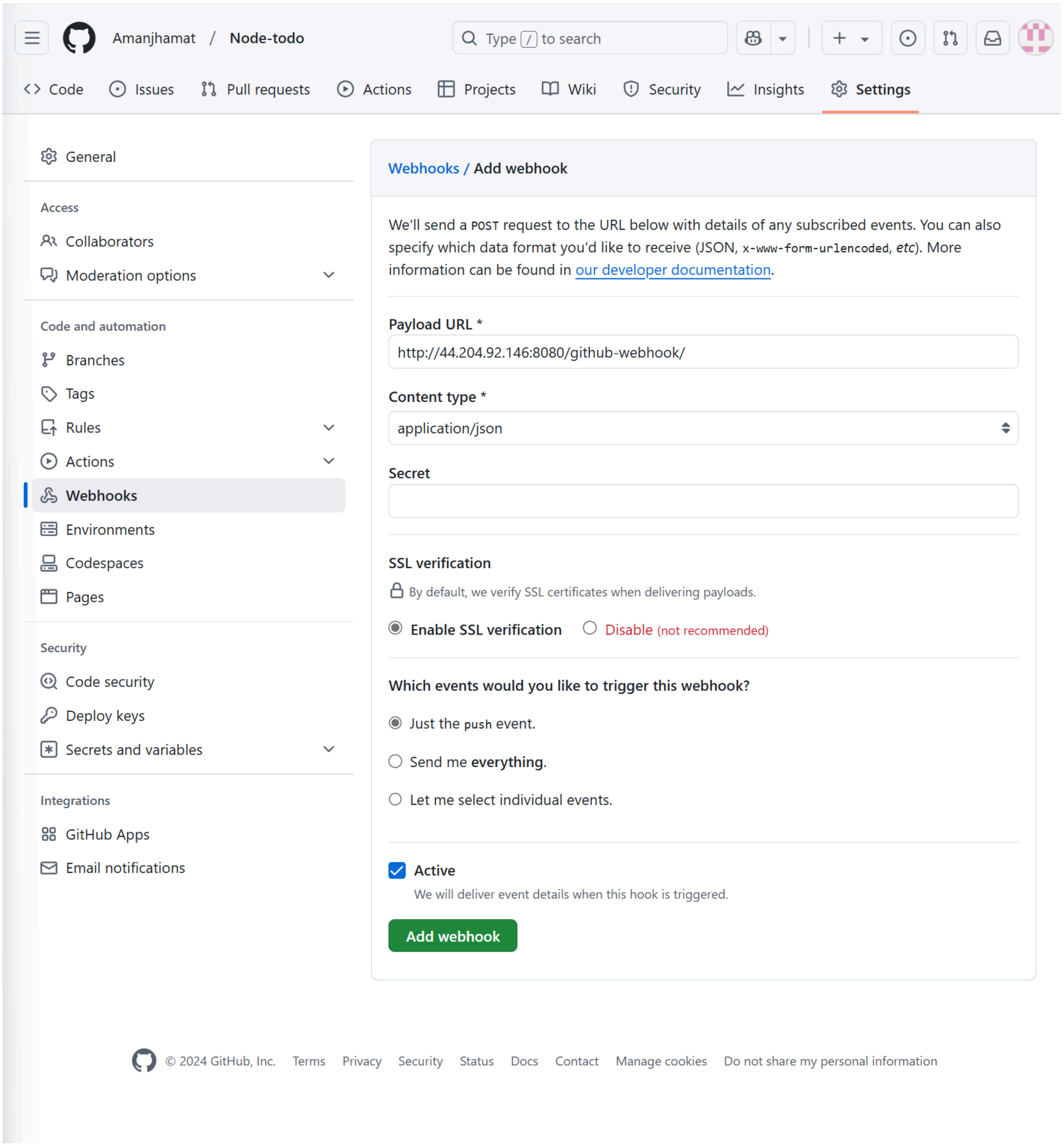


Webhooks

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our Webhooks Guide.

17.2) Configuring Jenkins to Trigger Webhook:

I then configured Jenkins to respond to GitHub webhook triggers by navigating to the Jenkins job configuration page. Under the "Build Triggers" section, I selected the "GitHub hook trigger for GITScm polling" option.



Webhooks / Add 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://44.204.92.146:8080/github-webhook/

Content type *
application/json

Secret

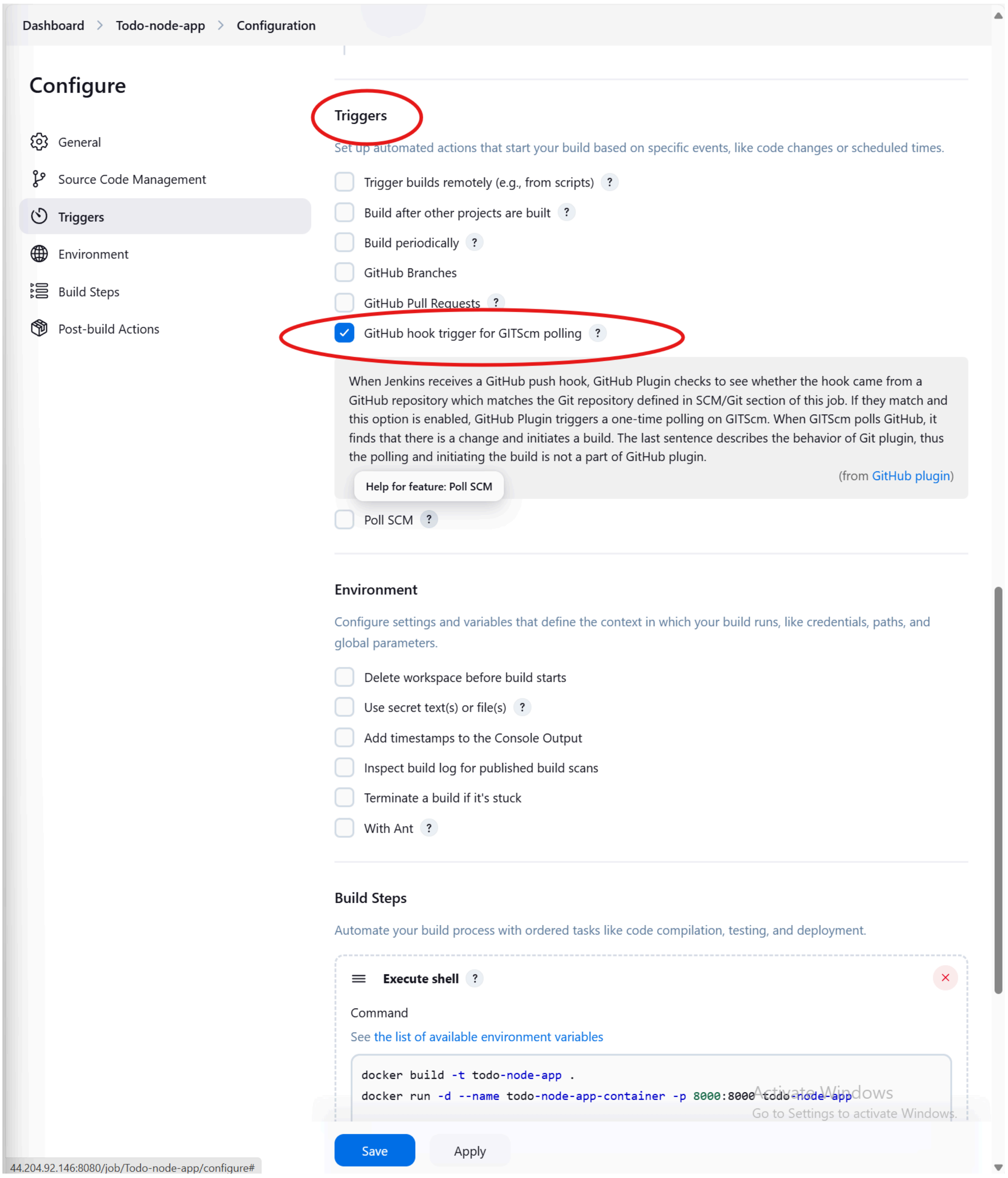
SSL verification
By default, we verify SSL certificates when delivering payloads.
[✓ Enable SSL verification] Disable (not recommended)

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.

Add webhook

© 2024 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information



Dashboard > Todo-node-app > Configuration

Configure

- General
- Source Code Management
- Triggers
- Environment
- Build Steps
- Post-build Actions

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 ?

When Jenkins receives a GitHub push hook, GitHub Plugin checks to see whether the hook came from a GitHub repository which matches the Git repository defined in SCM/Git section of this job. If they match and this option is enabled, GitHub Plugin triggers a one-time polling on GITScm. When GITScm polls GitHub, it finds that there is a change and initiates a build. The last sentence describes the behavior of Git plugin, thus the polling and initiating the build is not a part of GitHub plugin.

(from GitHub plugin)

Help for feature: Poll SCM

- [ ] 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) ?
- [ ] Add timestamps to the Console Output
- [ ] Inspect build log for published build scans
- [ ] Terminate a build if it's stuck
- [ ] With Ant ?

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

```
docker build -t todo-node-app .
docker run -d --name todo-node-app-container -p 8000:8000 todo-node-app
```

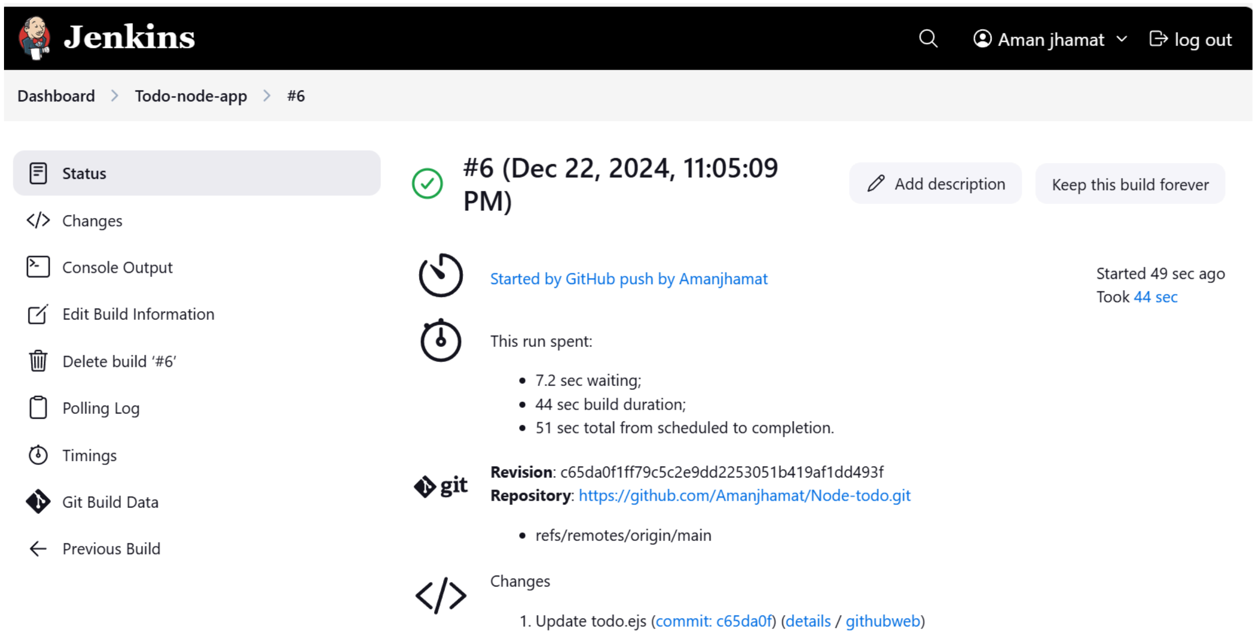
Activate Windows
Go to Settings to activate Windows.

Save Apply

44.204.92.146:8080/job/Todo-node-app/configure#

17.3) Testing the Webhook Integration:

After configuring the webhook, I made changes to the code, and Jenkins automatically detected the changes and triggered the deployment, as shown below.



#6 (Dec 22, 2024, 11:05:09 PM)

Started by GitHub push by Amanjhamat

Started 49 sec ago
Took 44 sec

This run spent:

- 7.2 sec waiting;
- 44 sec build duration;
- 51 sec total from scheduled to completion.

Revision: c65da0f1ff79c5c2e9dd2253051b419af1dd493f
Repository: https://github.com/Amanjhamat/Node-todo.git

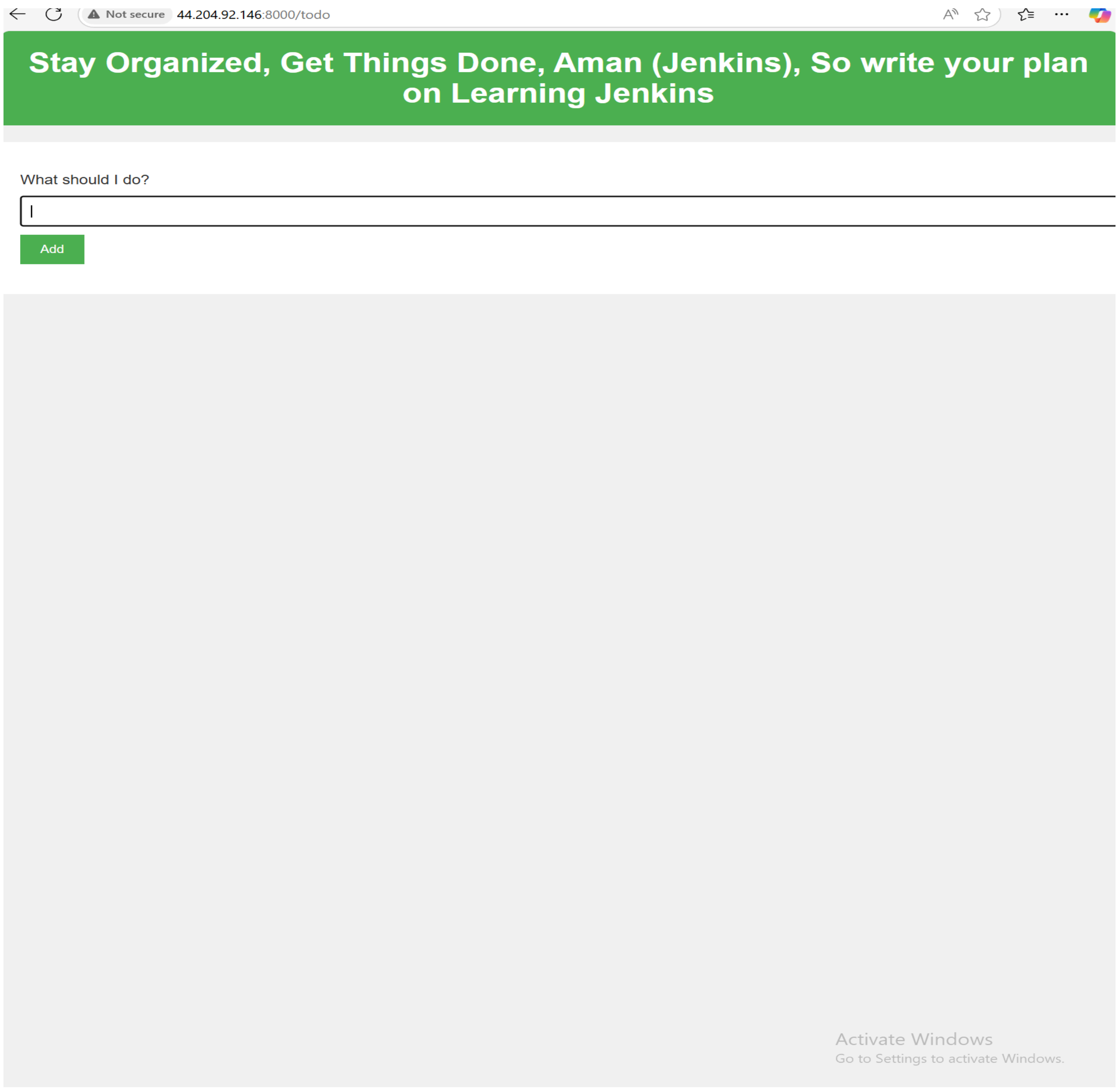
- refs/remotes/origin/main

Changes

1. Update todo.ejs (commit: c65da0f) (details / githubweb)

18) Successful Auto Deployment After Code Changes:

The webhook integration worked successfully, and any changes made to the GitHub repository were automatically deployed, with the application reflecting the updated headline: “Stay Organized, Get Things Done, Aman (Jenkins).



Stay Organized, Get Things Done, Aman (Jenkins), So write your plan on Learning Jenkins

What should I do?

Thank you,

Aman Jhamat