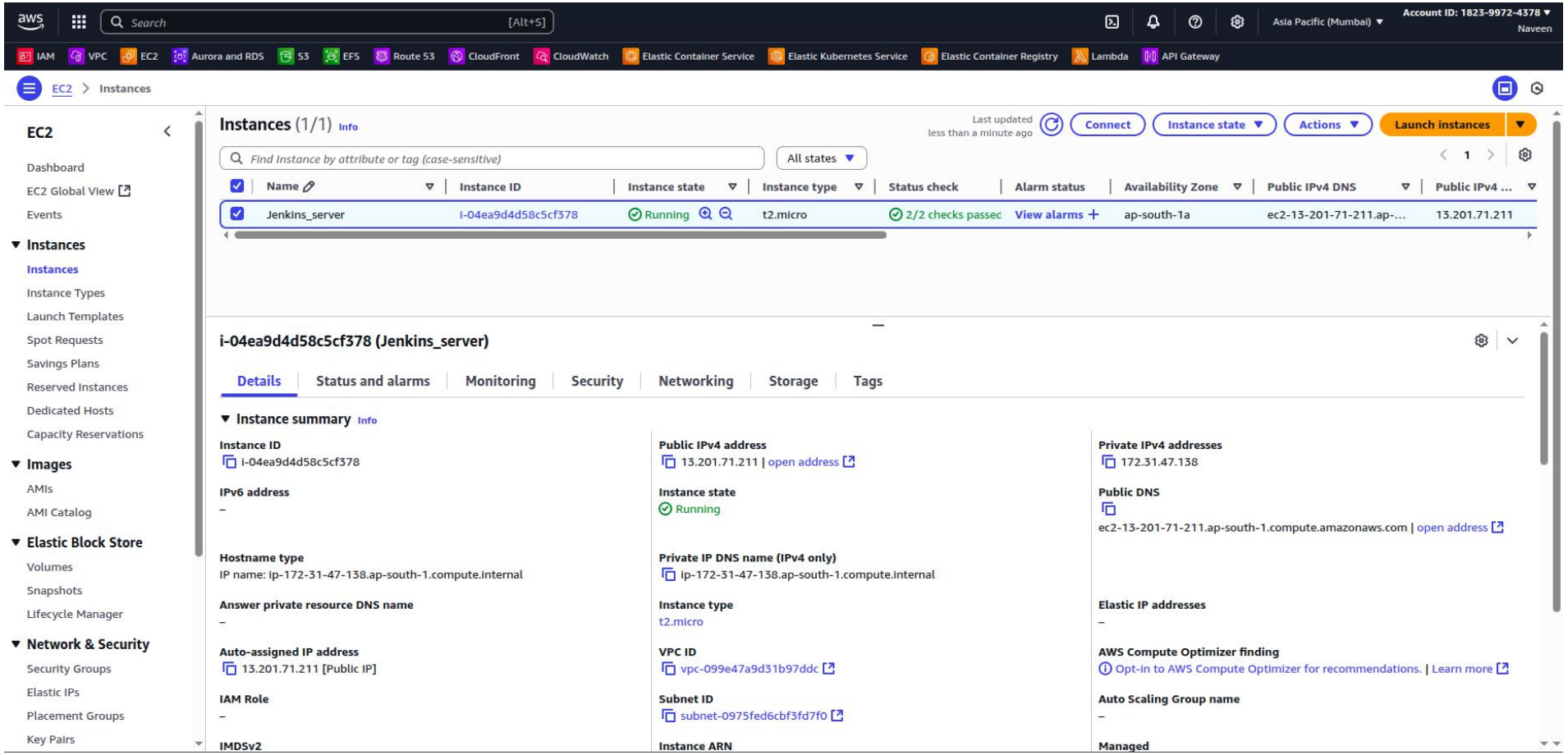
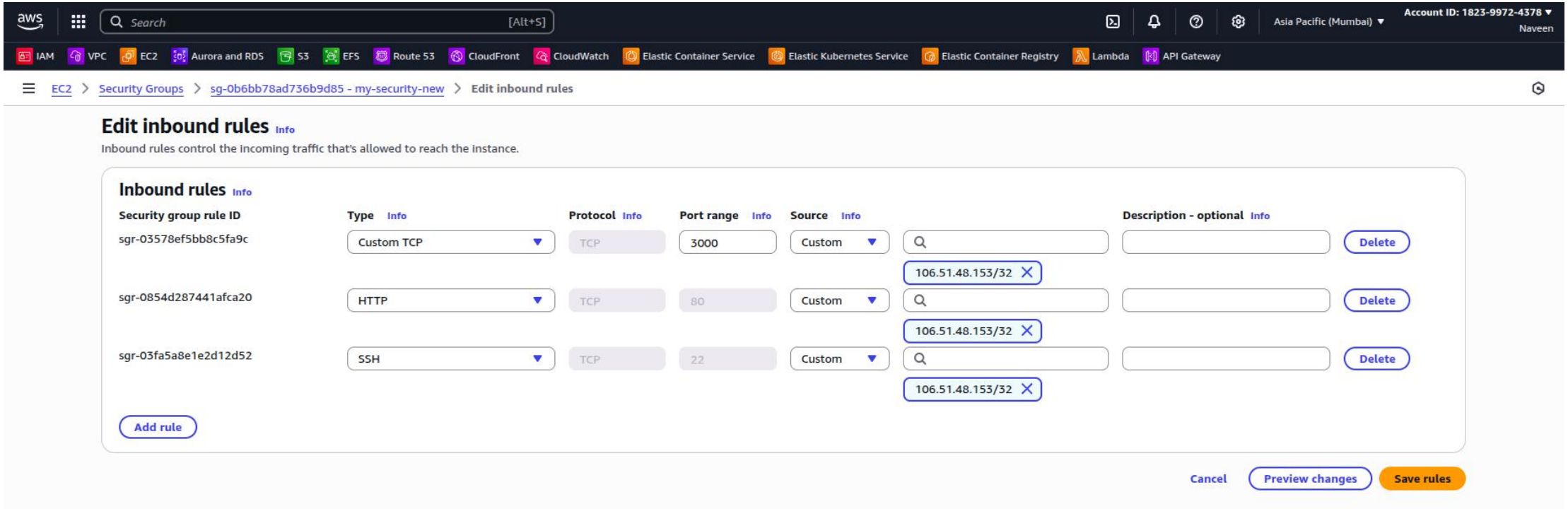
# Deploy a simple Node.js App to AWS EC2 using Jenkins CI/CD Pipeline

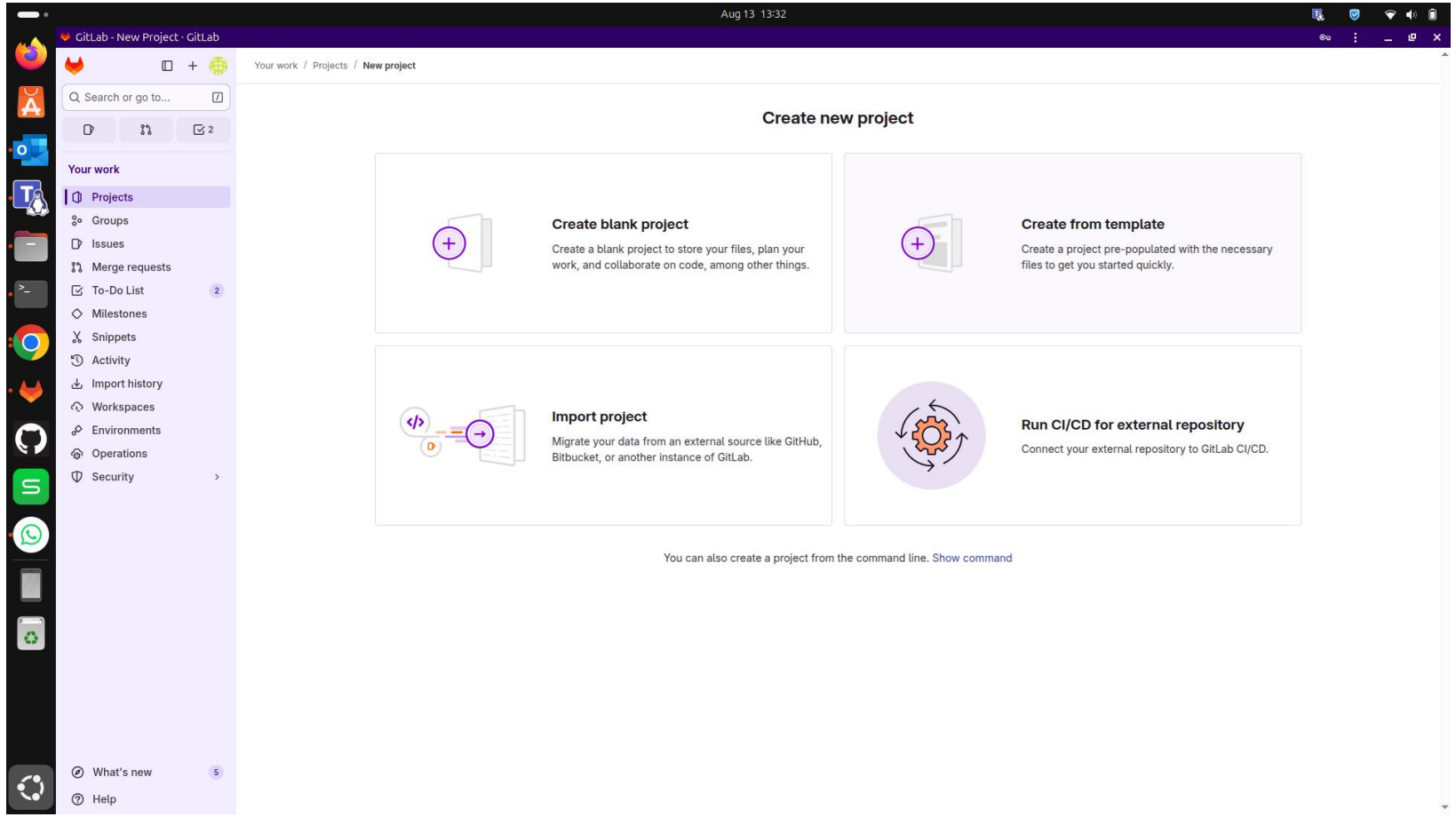
1. At first , Launch an Amazon Linux 2 instance in EC2 and name it as **jenkins_server**



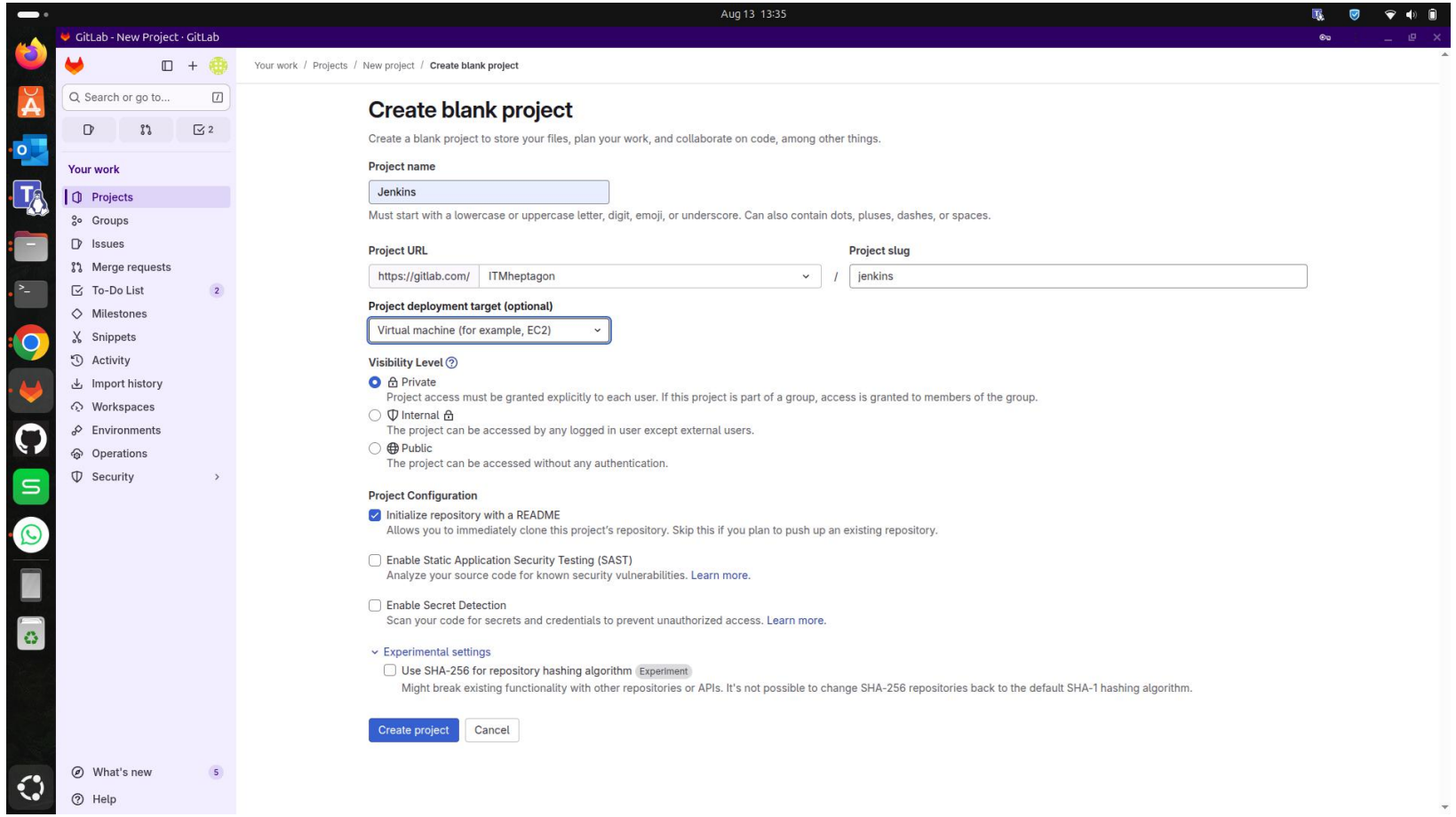
2. In the Security Group inbound rules, allow the following:
**SSH (22) HTTP (80) Custom TCP (3000) for Node.js**



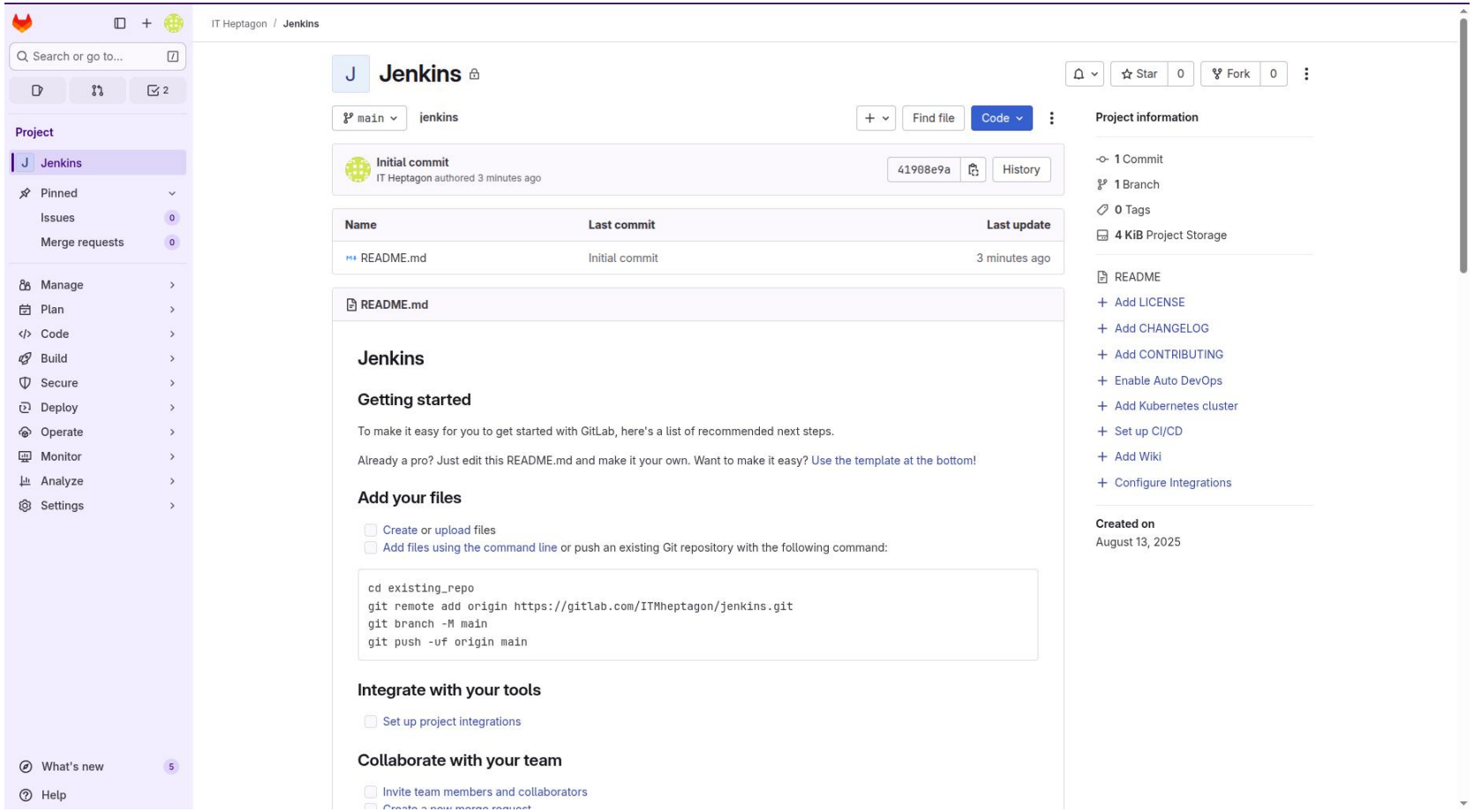
3. In GitLab, create a new project by selecting **Create Blank Project**



4. Name the project as **Jenkins**, select your **Gitlab URL** & select the project deployment target as Virtual machine (EC2). Make the repo as **private** and **create project**.



5. Once the repo is created, clone it to your local system.



6. Create a **Jenkinsfile** to trigger a build from GitLab and deploy the **Node.js** app to the EC2 server

Your changes have been committed successfully.

**Jenkinsfile** 1.88 KiB

```
pipeline {
    agent any

    tools {
        nodejs "Node 18" // Must match Manage Jenkins → Tools → NodeJS installations
    }

    environment {
        EC2_USER = "ec2-user"               // Change to 'ubuntu' if needed
        EC2_HOST = "13.201.71.211"          // Public IP or DNS
        EC2_PATH = "/home/ec2-user/jenkins" // Deployment directory on EC2
    }

    stages {
        stage('Checkout') {
            steps {
                git branch: 'main',
                    credentialsId: '7243f613-7608-4328-97ed-da54cc32ccb7',
                    url: 'https://gitlab.com/ITMheptagon/jenkins.git'
            }
        }

        stage('Build') {
            steps {
                sh 'npm install'
            }
        }

        stage('Test') {
            steps {
                sh 'npm test'
            }
        }

        stage('Deploy') {
    steps {
        withCredentials([sshUserPrivateKey(credentialsId: 'cc32cf60-9256-446b-b6a7-a7accb97c597', keyFileVariable: 'SSH_KEY')]) {
            sh """
                ssh -i $SSH_KEY -o StrictHostKeyChecking=no $EC2_USER@$EC2_HOST '
                    mkdir -p $EC2_PATH
                    if ! command -v npm &> /dev/null; then
                        curl -fsSL https://rpm.nodesource.com/setup_18.x | sudo bash -
                        sudo yum install -y nodejs
                    fi
                '
                scp -i $SSH_KEY -o StrictHostKeyChecking=no index.js package.json $EC2_USER@$EC2_HOST:$EC2_PATH
                ssh -i $SSH_KEY -o StrictHostKeyChecking=no $EC2_USER@$EC2_HOST "
                    cd $EC2_PATH && npm install && node index.js
                "
            """
        }
    }
}

    }

    post {
        always {
            echo 'Pipeline completed!'
        }
```

```
pipeline {
    agent any

    tools {
        nodejs "Node 18" // Must match Manage Jenkins → Tools → NodeJS installations
    }

    environment {
        EC2_USER = "ec2-user"               // Change to 'ubuntu' if needed
        EC2_HOST = "13.201.71.211"          // Public IP or DNS
        EC2_PATH = "/home/ec2-user/jenkins" // Deployment directory on EC2
    }

    stages {
        stage('Checkout') {
            steps {
                git branch: 'main',
                    credentialsId: '7243f613-7608-4328-97ed-da54cc32ccb7',
                    url: 'https://gitlab.com/ITMheptagon/jenkins.git'
            }
        }

        stage('Build') {
            steps {
                sh 'npm install'
            }
        }

        stage('Test') {
            steps {
                sh 'npm test'
            }
        }

        stage('Deploy') {
            steps {
                withCredentials([sshUserPrivateKey(credentialsId: 'cc32cf60-9256-446b-b6a7-a7accb97c597', keyFileVariable: 'SSH_KEY')]) {
                    sh """
                        # Create app directory on EC2
                        ssh -i $SSH_KEY -o StrictHostKeyChecking=no $EC2_USER@$EC2_HOST "mkdir -p $EC2_PATH"

                        # Copy latest code to EC2
                        scp -i $SSH_KEY -o StrictHostKeyChecking=no index.js package.json $EC2_USER@$EC2_HOST:$EC2_PATH

                        # Install dependencies & restart app with PM2
                        ssh -i $SSH_KEY -o StrictHostKeyChecking=no $EC2_USER@$EC2_HOST '
                            cd $EC2_PATH && npm install
                            if ! command -v pm2 &> /dev/null; then
                                sudo npm install -g pm2
                            fi
                            pm2 start ./index.js --name hello-jenkins --update-env || pm2 restart hello-jenkins
                        '
```

# 7. Create an **index.js** file for the Node.js server

main / jenkins / index.js

## index.js

Find file | Blame | Edit

**Edit index.js**
IT Heptagon authored 40 seconds ago

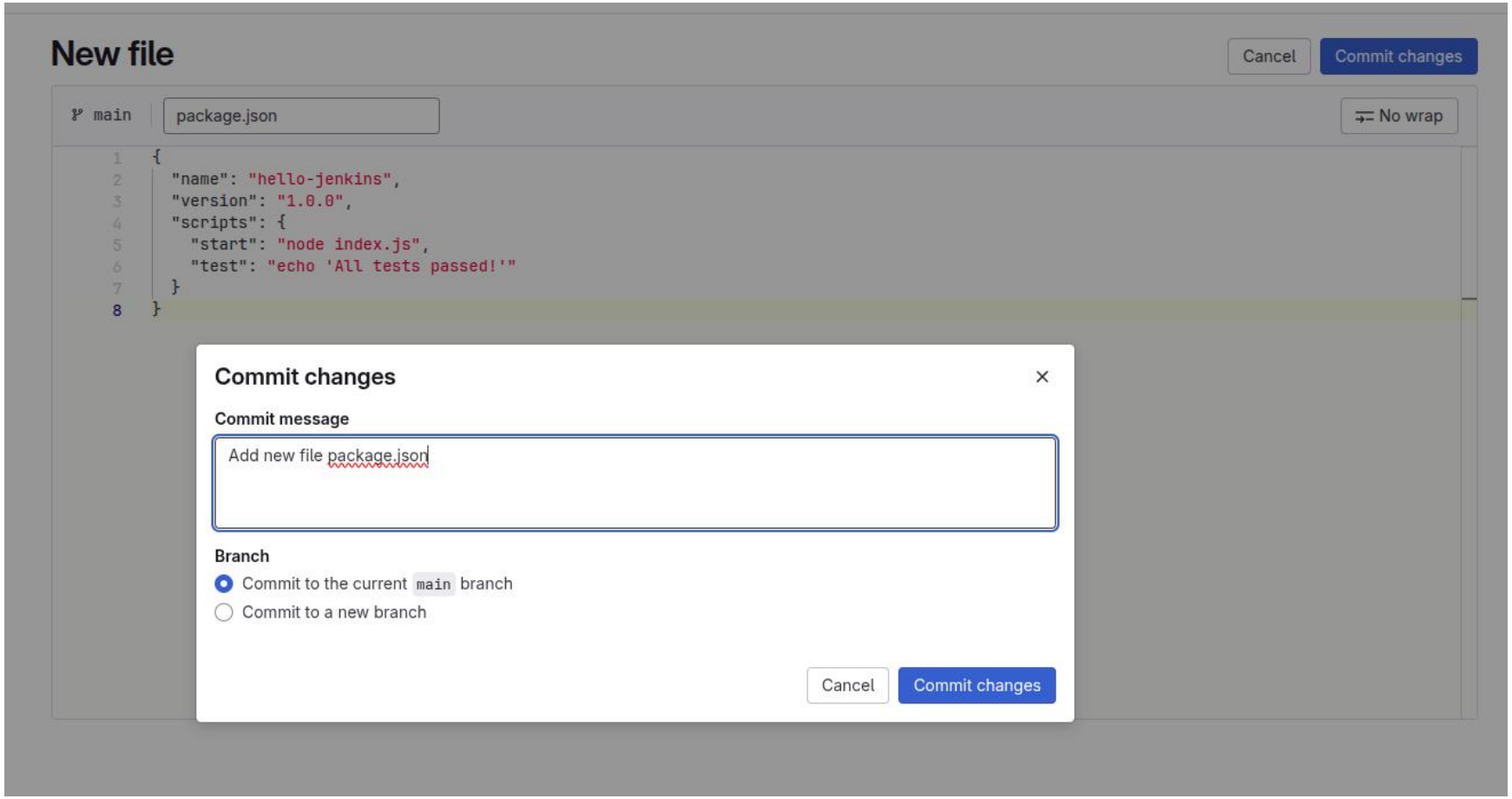
2858a583 | History

index.js 342 B

```js
const http = require('http');

const hostname = '0.0.0.0';
const port = 3000;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello, Jenkins!\n');
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});

```

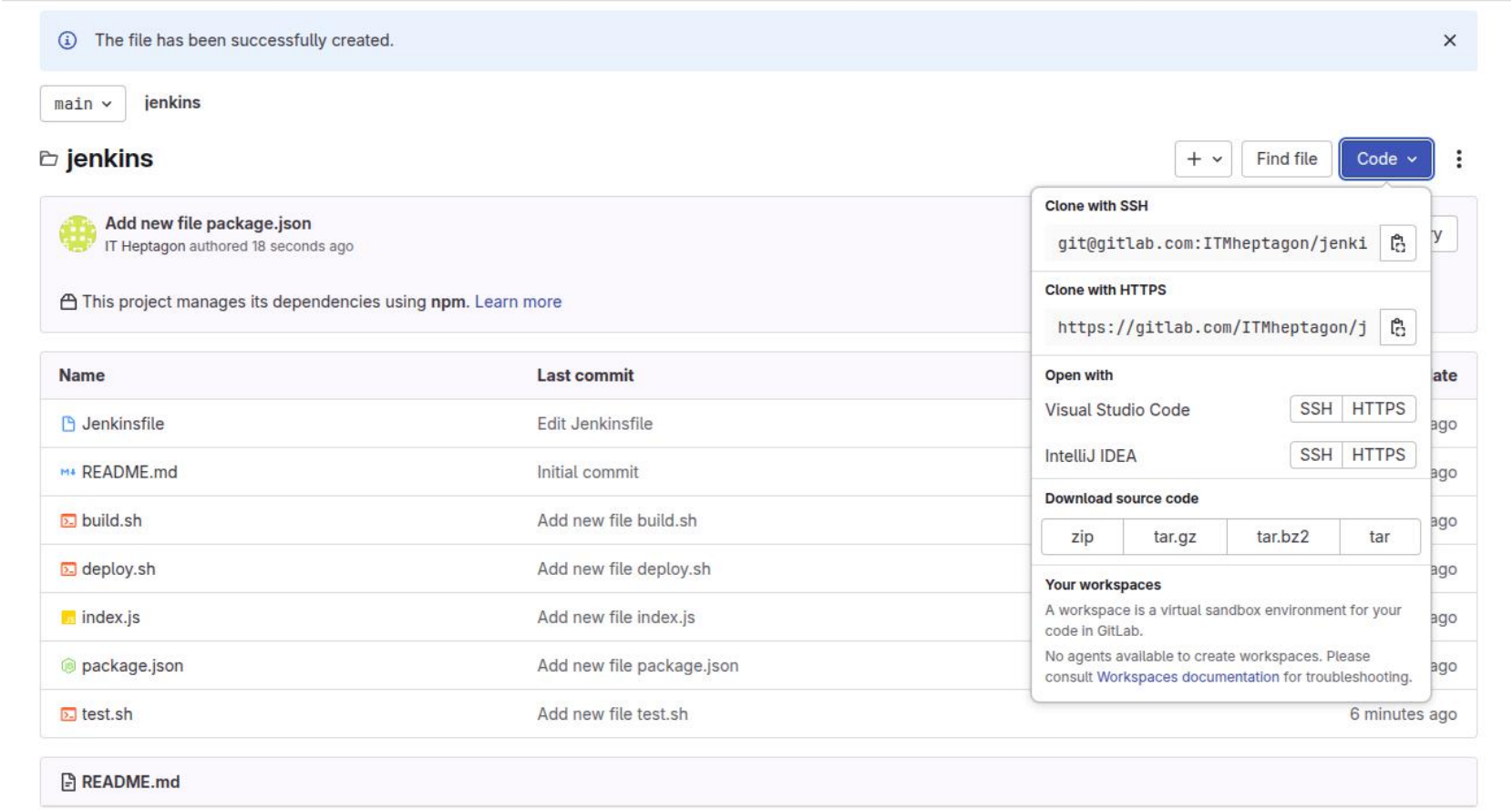
```
const http = require('http');
const hostname = '0.0.0.0';
const port = 3000;
const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello, Jenkins!\n');
});
server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

8. Also you need to create **package.json** file and commit the changes.



```
name        "hello-jenkins"
version     "1.0.0"
scripts
start "node index.js"
test  "echo 'All tests passed!'"
```

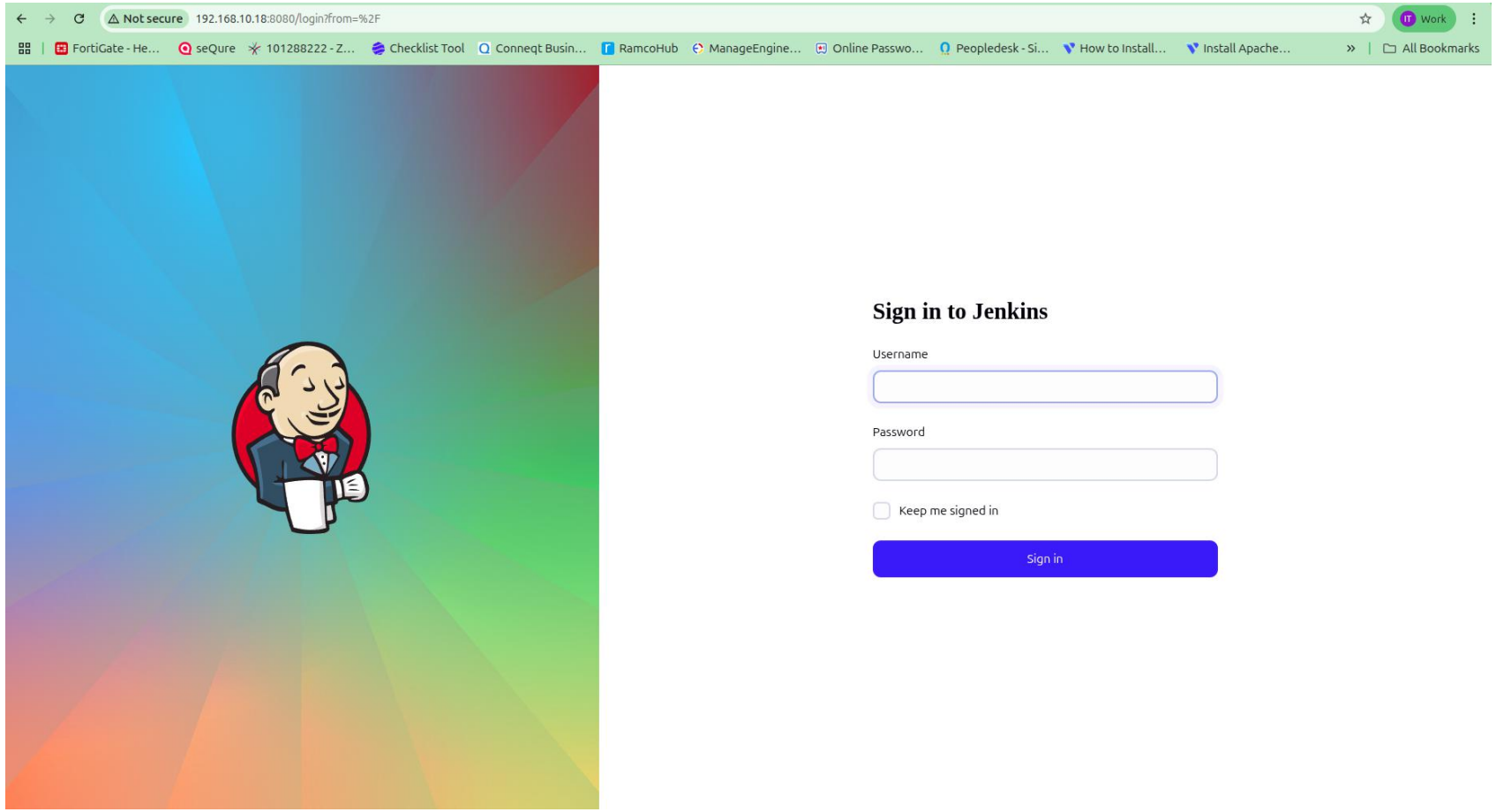
9. Now you can see all the files in the repo, copy the HTTPS url for cloning the repo.



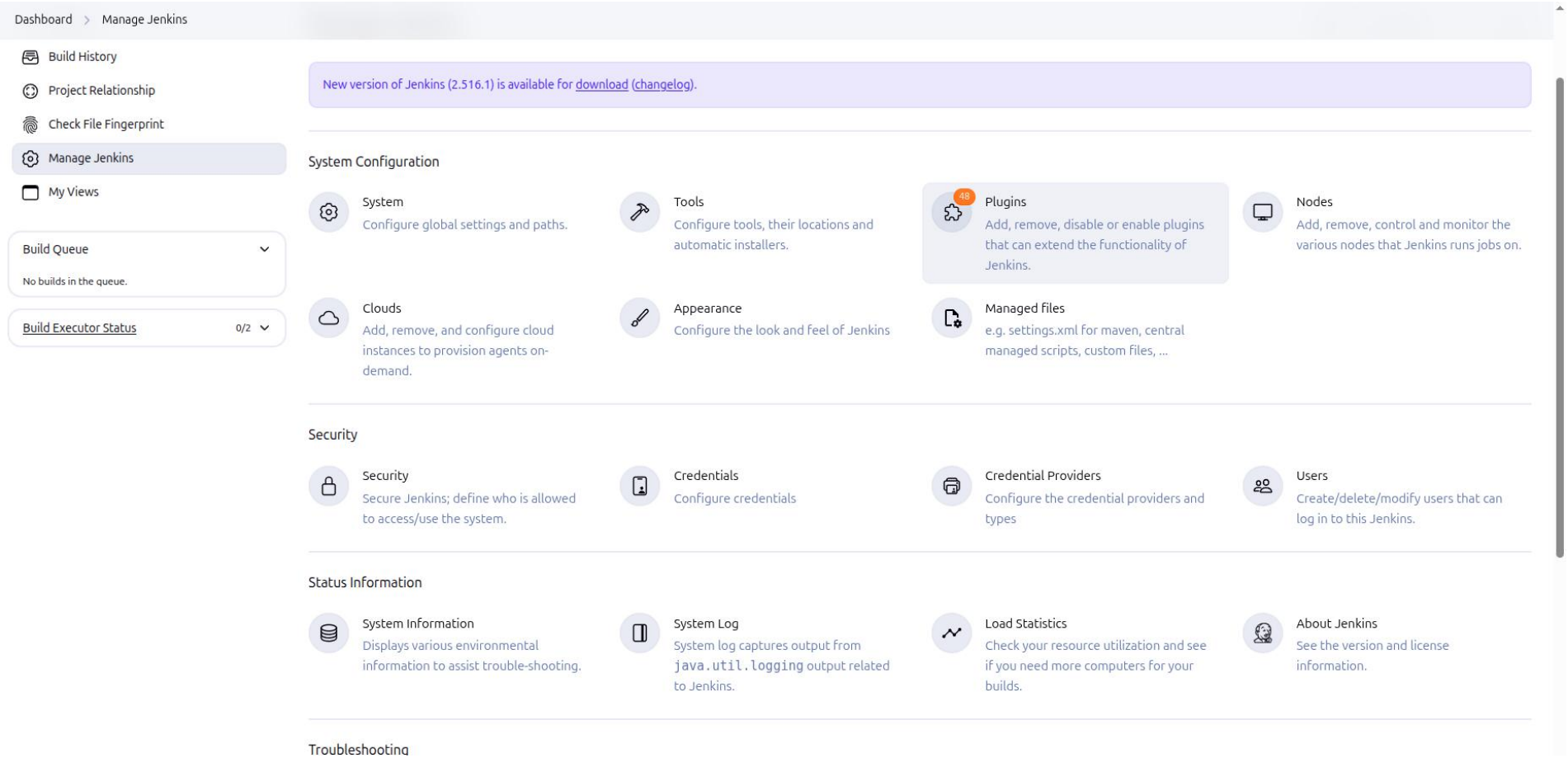
10. Make sure Git is installed in your local machine and clone the repo with the copied HTTPS url **git clone https://gitlab.com/ITMheptagon/jenkins.git**

```
root@5CD20890KH: /home/naveen/jenkins
root@5CD20890KH:/home/naveen# git --version
git version 2.43.0
root@5CD20890KH:/home/naveen# ls
aws  Desktop  Documents  Downloads  Music  new-project  nginx-php  nginx-php.zip  nodejs  Pictu
root@5CD20890KH:/home/naveen# git clone https://gitlab.com/ITMheptagon/jenkins.git
Cloning into 'jenkins'...
Username for 'https://gitlab.com': ITMheptagon
Password for 'https://ITMheptagon@gitlab.com':
remote: Enumerating objects: 24, done.
remote: Counting objects: 100% (24/24), done.
remote: Compressing objects: 100% (22/22), done.
remote: Total 24 (delta 6), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (24/24), 5.39 KiB | 5.39 MiB/s, done.
Resolving deltas: 100% (6/6), done.
root@5CD20890KH:/home/naveen# ls
aws  Desktop  Documents  Downloads  jenkins  Music  new-project  nginx-php  nginx-php.zip  node
root@5CD20890KH:/home/naveen# cd jenkins/
root@5CD20890KH:/home/naveen/jenkins# ls
build.sh  deploy.sh  index.js  Jenkinsfile  package.json  README.md  test.sh
root@5CD20890KH:/home/naveen/jenkins#
```

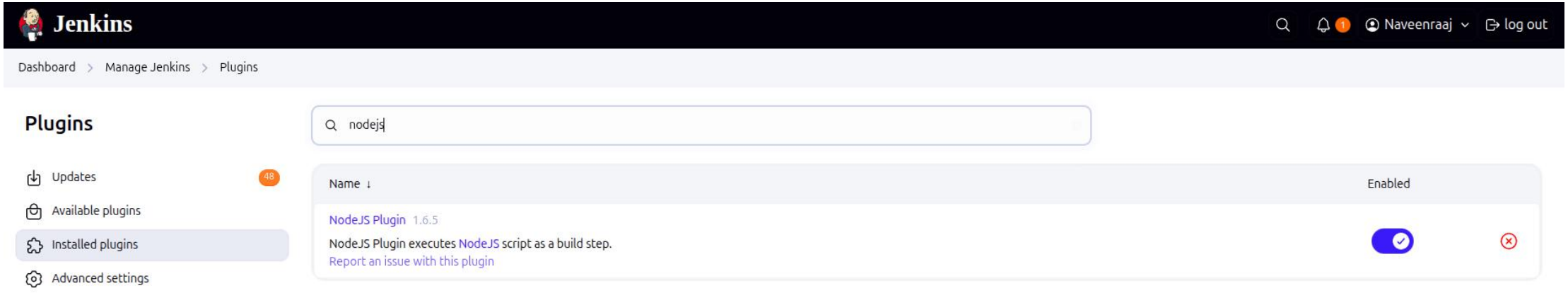
11. Install jenkins package in your local machine and login to the URL **http://localhost:8080** (Ensure port 8080 is open)



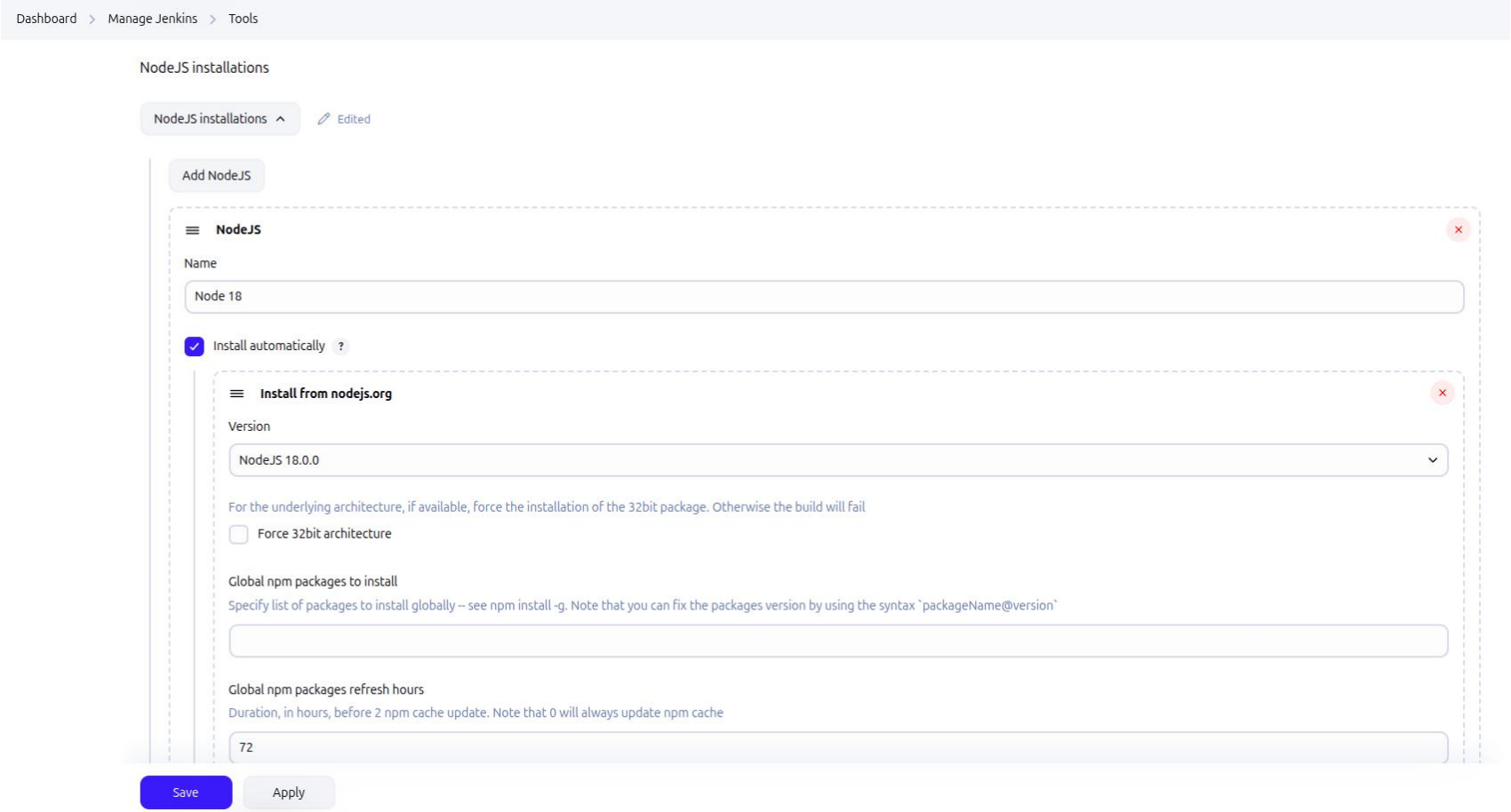
12. In the Dashboard, go to Manage Jenkins → Plugins and install the **NodeJS plugin**.



13. Install **NodeJS** plugin.

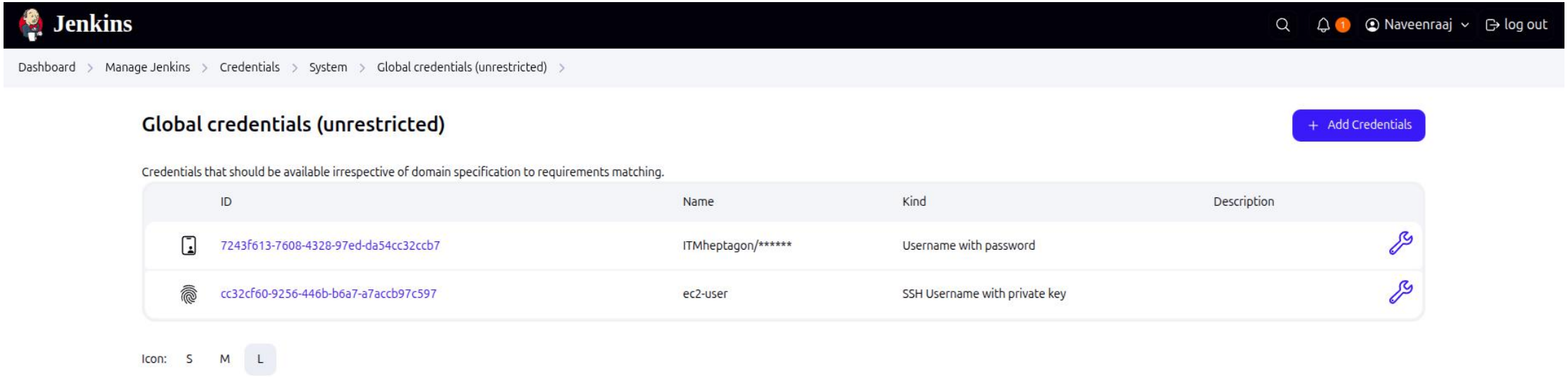


14. Configure **Node 18** in Manage Jenkins → Tools → NodeJS installations.

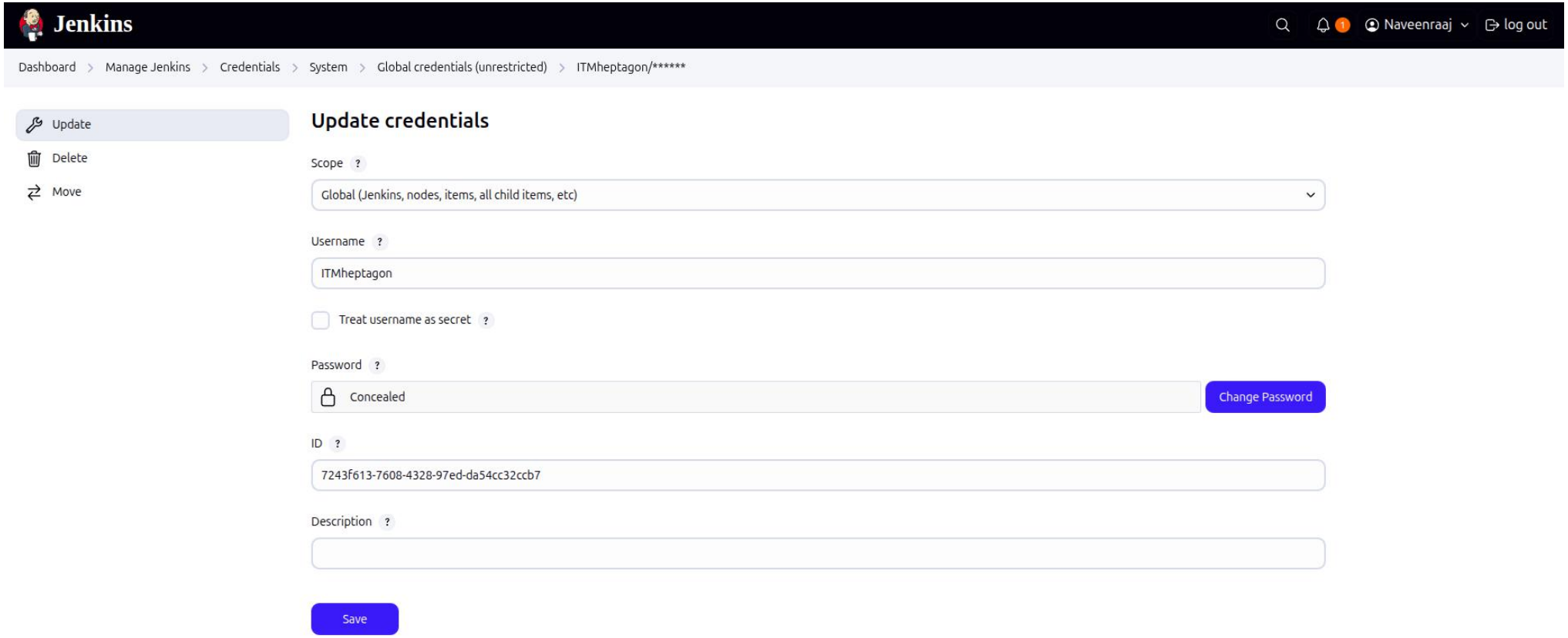


15. Configure credentials for GitLab and EC2:

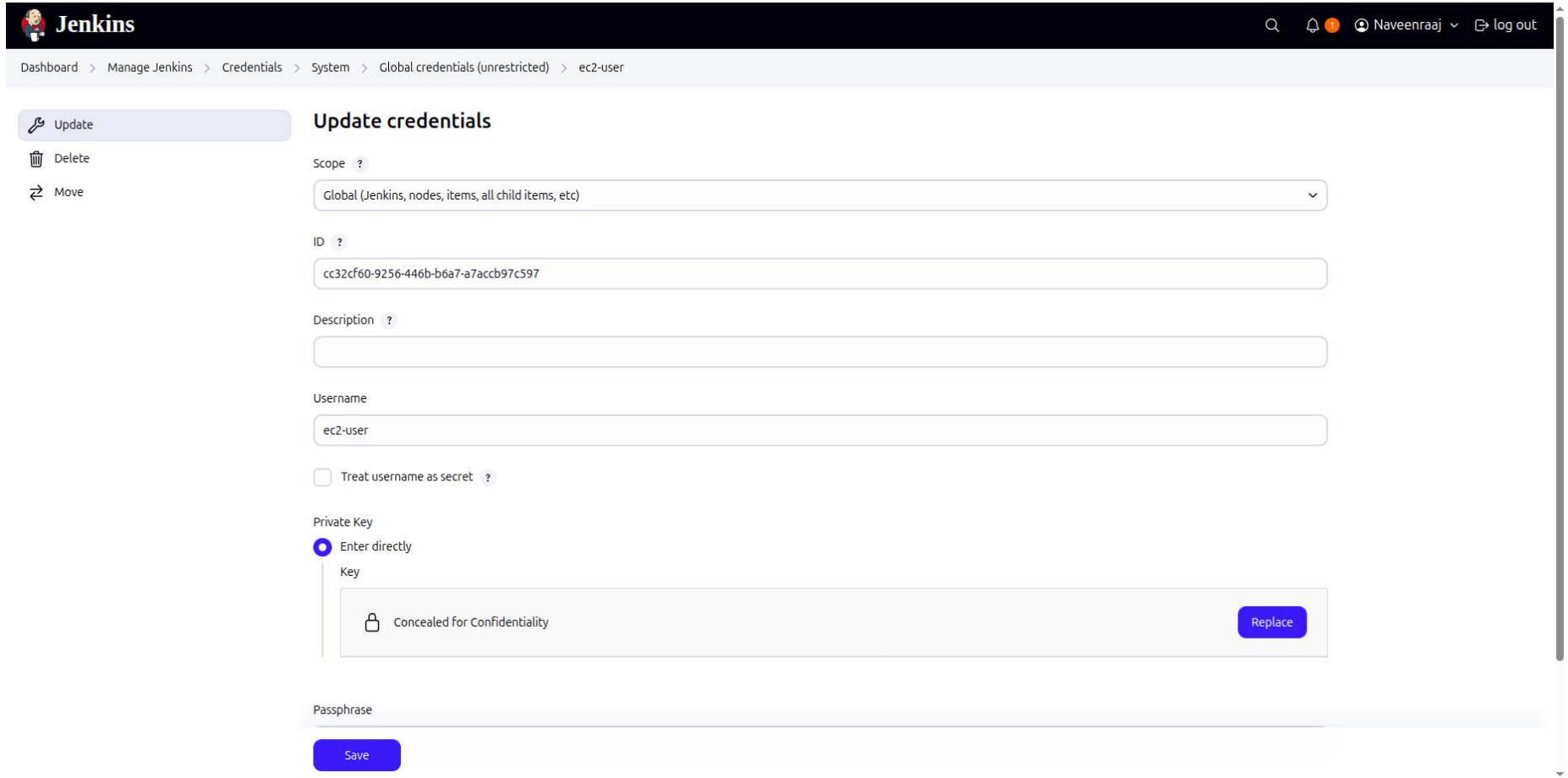
**GitLab: Username + Personal Access Token**
**EC2: SSH Username (ec2-user) + Private Key from key pair**



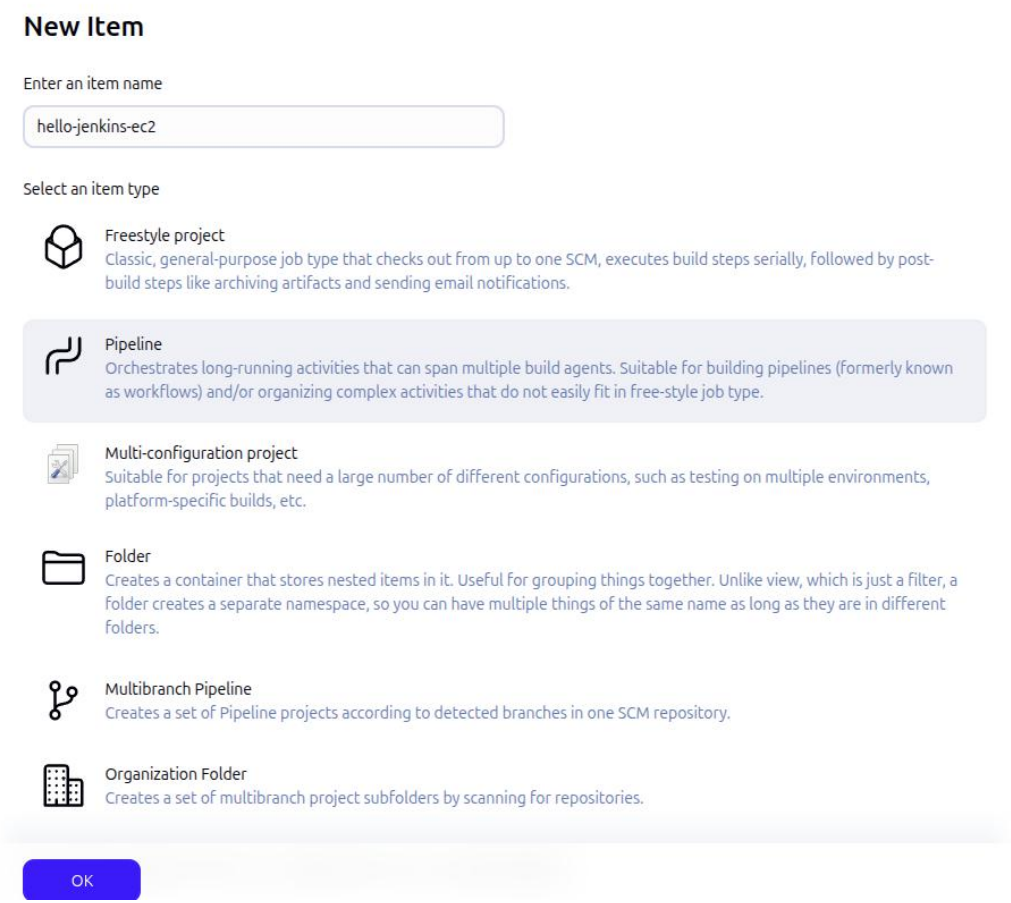
16. For GitLab, Kind **username & password,** type your username and copy the **Personal Access Token** from the GitLab and paste it in the password and save it.



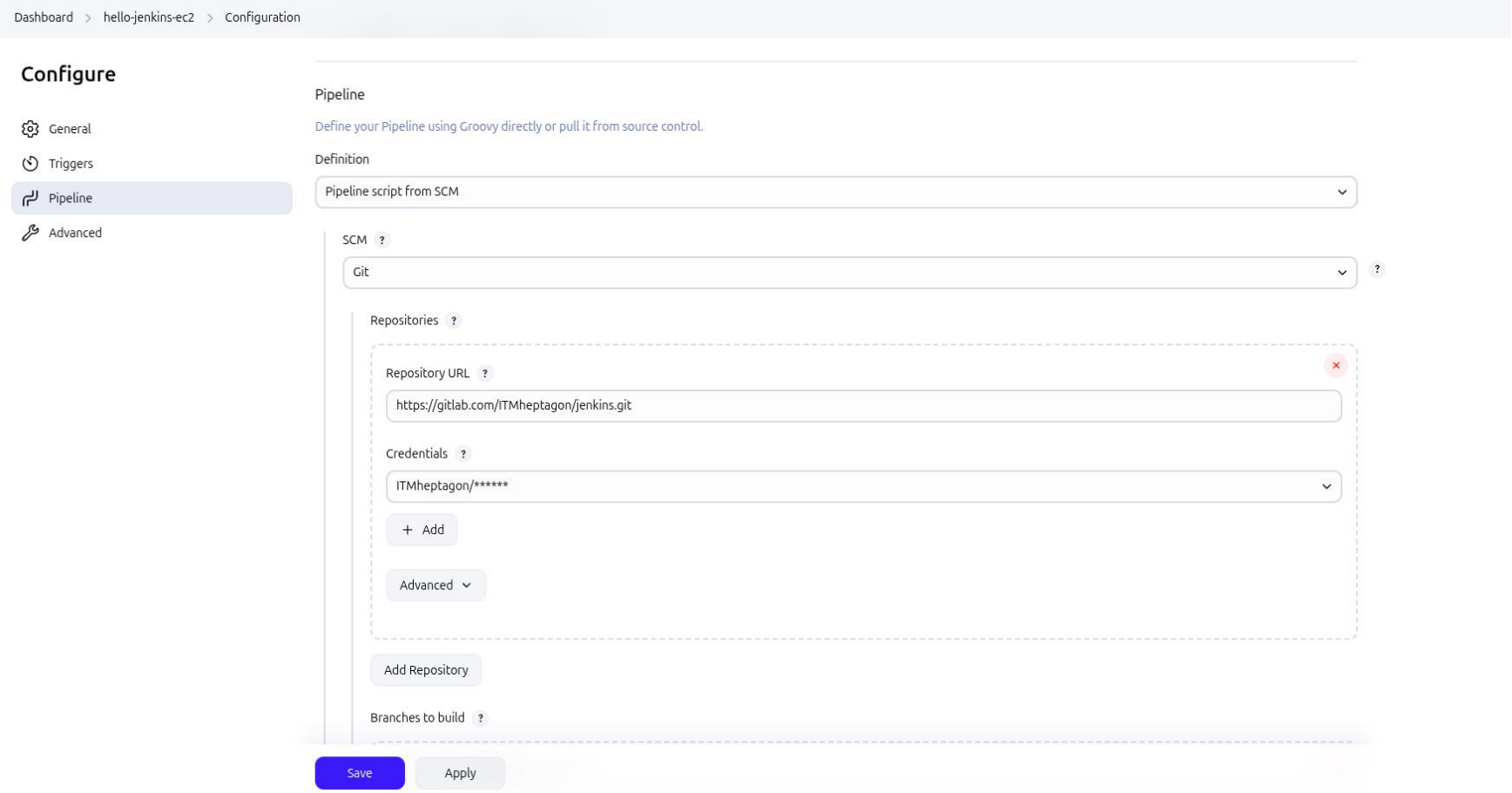
17. For EC2 server, select the kind as **SSH username with privatekey** and type the username as **ec2-user** then copy and paste the private key from the instance keypair.



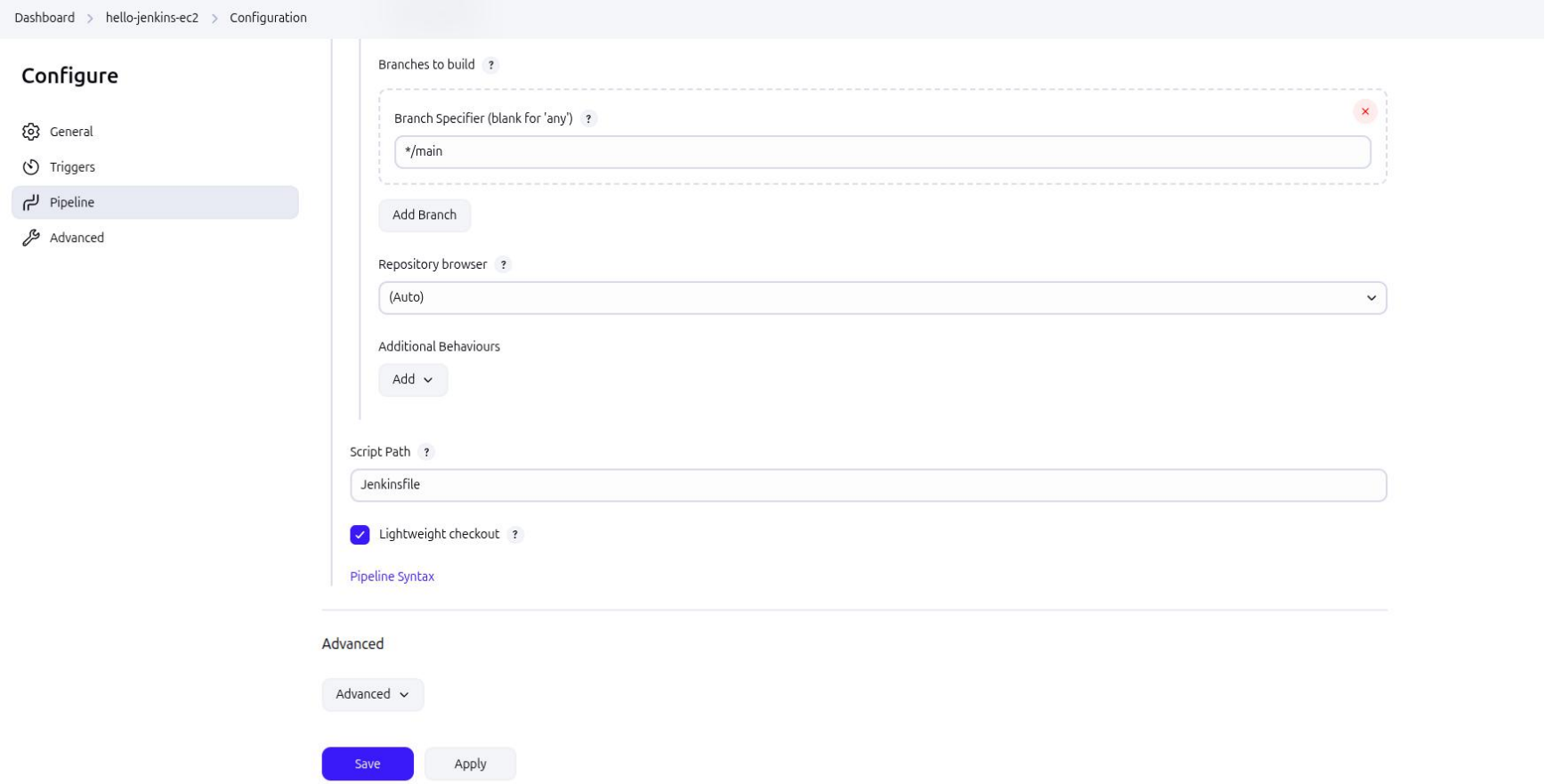
18. Create a pipeline named **hello-jenkins-ec2**.



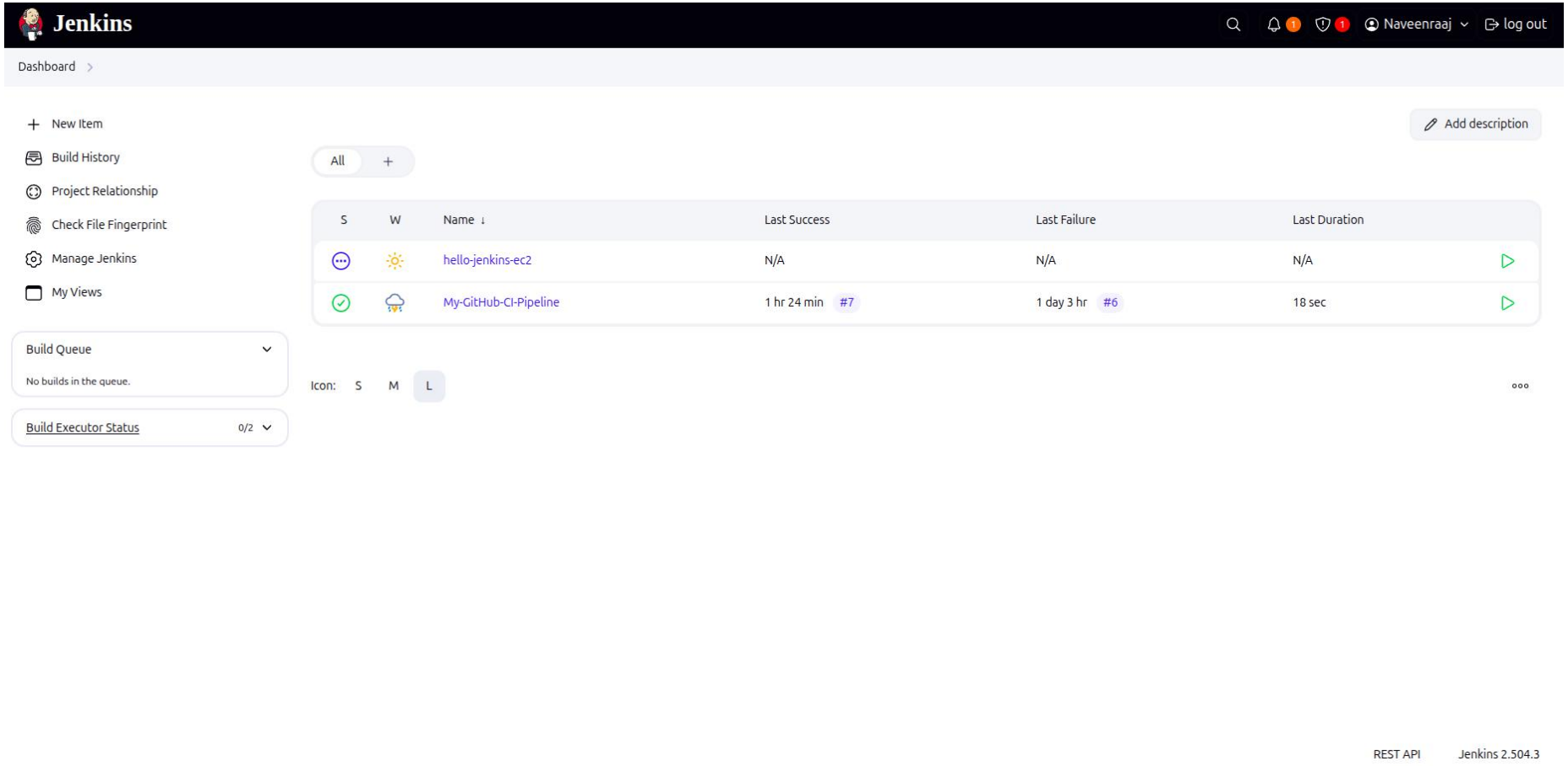
19. Configure the pipeline, choose **Pipeline script from SCM**, select **Git** as source code, paste the GitLab repo url **https://gitlab.com/ITMheptagon/jenkins.git** and select the GitLab credentials as saved in the global tool configuration.



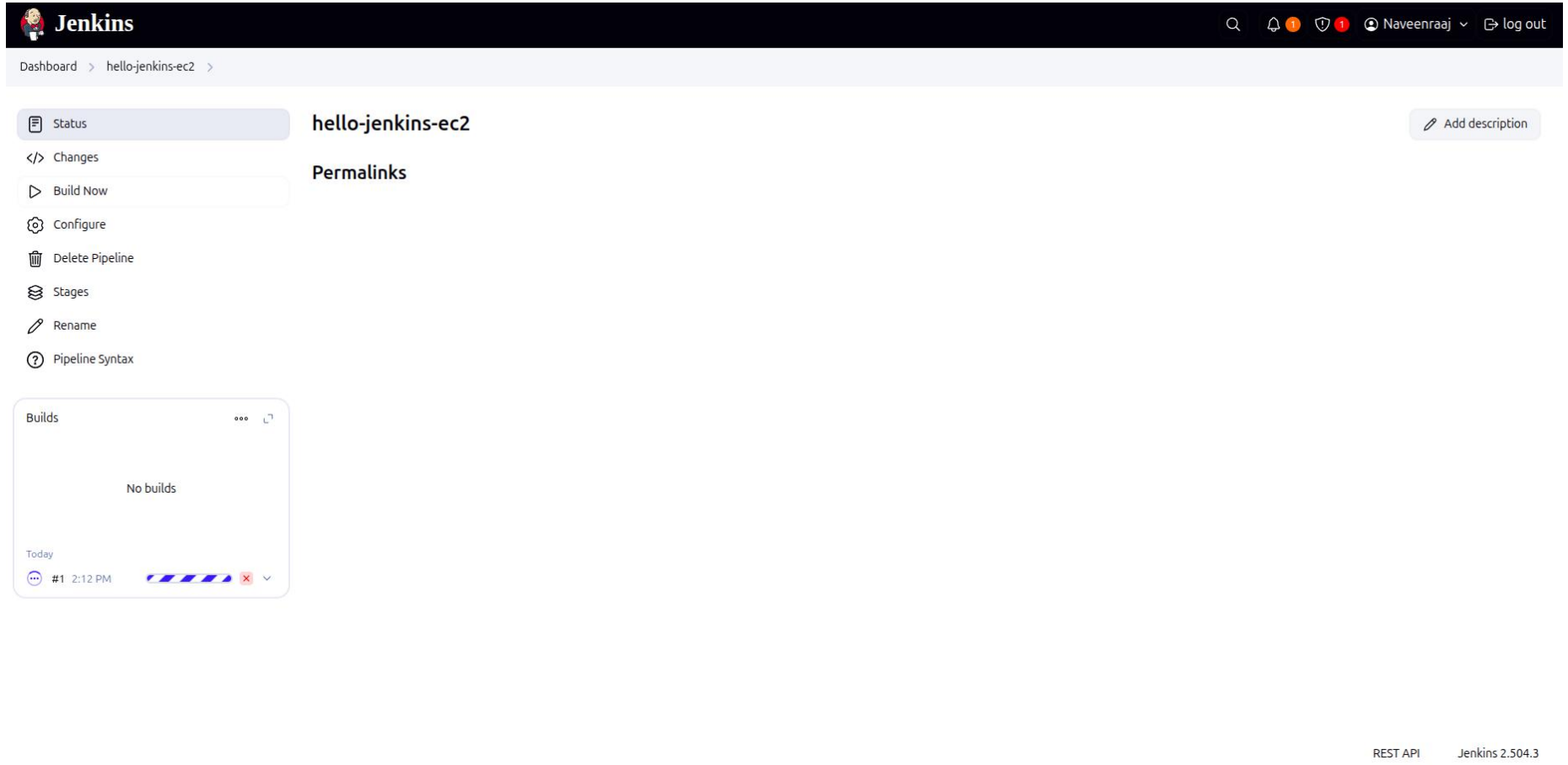
20. Specify the branch name as ***/Main** in the branch section. Click apply and save.



21. Now you can see the created pipeline project in the dashboard.



22. Build the pipeline and monitor the console output for errors.



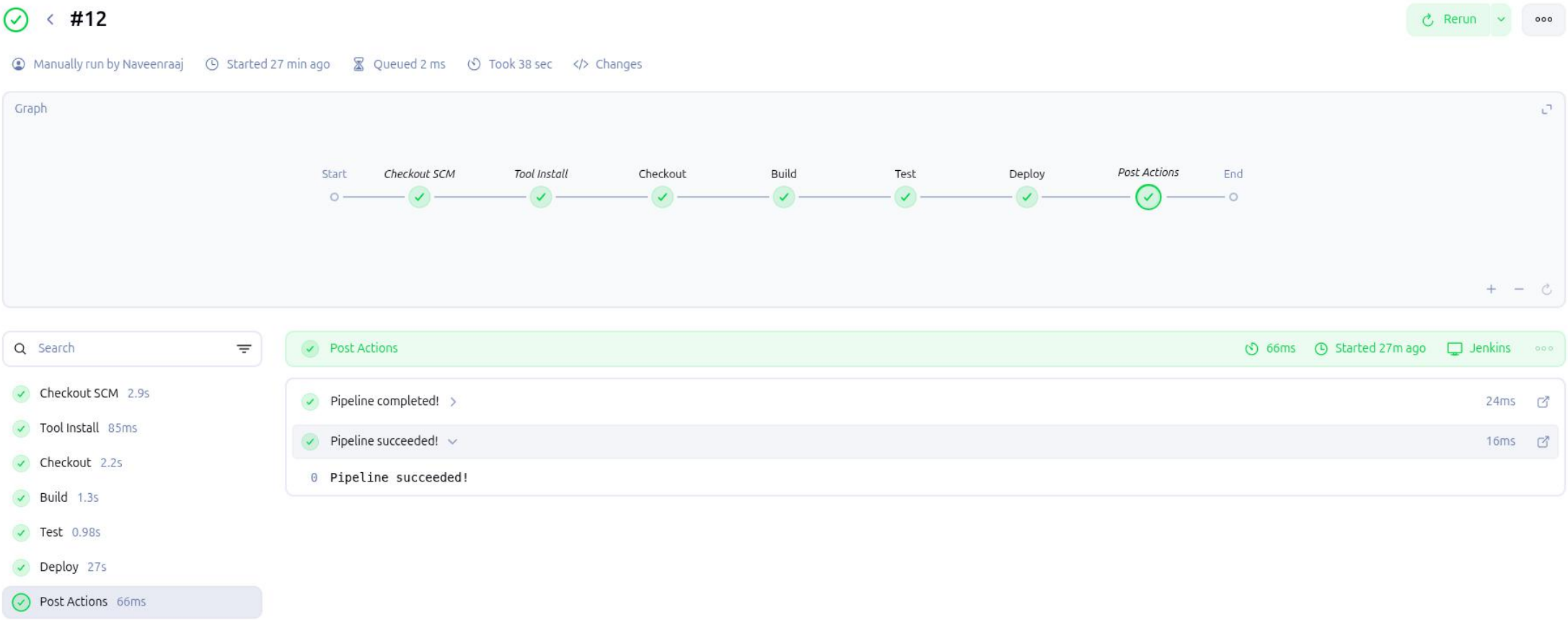
# 23. You can lookout for any errors in the console output

## Console Output

Download Copy View as plain text

```
Started by user Naveenraaj
Obtained Jenkinsfile from git https://gitlab.com/ITMheptagon/jenkins.git
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/hello-jenkins-ec2
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
using credential 7243f613-7608-4328-97ed-da54cc32ccb7
 > git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/hello-jenkins-ec2/.git # timeout=10
Fetching changes from the remote Git repository
 > git config remote.origin.url https://gitlab.com/ITMheptagon/jenkins.git # timeout=10
Fetching upstream changes from https://gitlab.com/ITMheptagon/jenkins.git
 > git --version # timeout=10
 > git --version # 'git version 2.43.0'
using GIT_ASKPASS to set credentials
 > git fetch --tags --force --progress -- https://gitlab.com/ITMheptagon/jenkins.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 81f4ded6a5472cd8a57951da4841189afa35a9b0 (refs/remotes/origin/main)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 81f4ded6a5472cd8a57951da4841189afa35a9b0 # timeout=10
Commit message: "Edit Jenkinsfile"
 > git rev-list --no-walk 3c30edf0366df9743ed95f321a659279a8855357 # timeout=10
[Pipeline] }
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] {
[Pipeline] withEnv
```

# 24. This is the pipeline overview

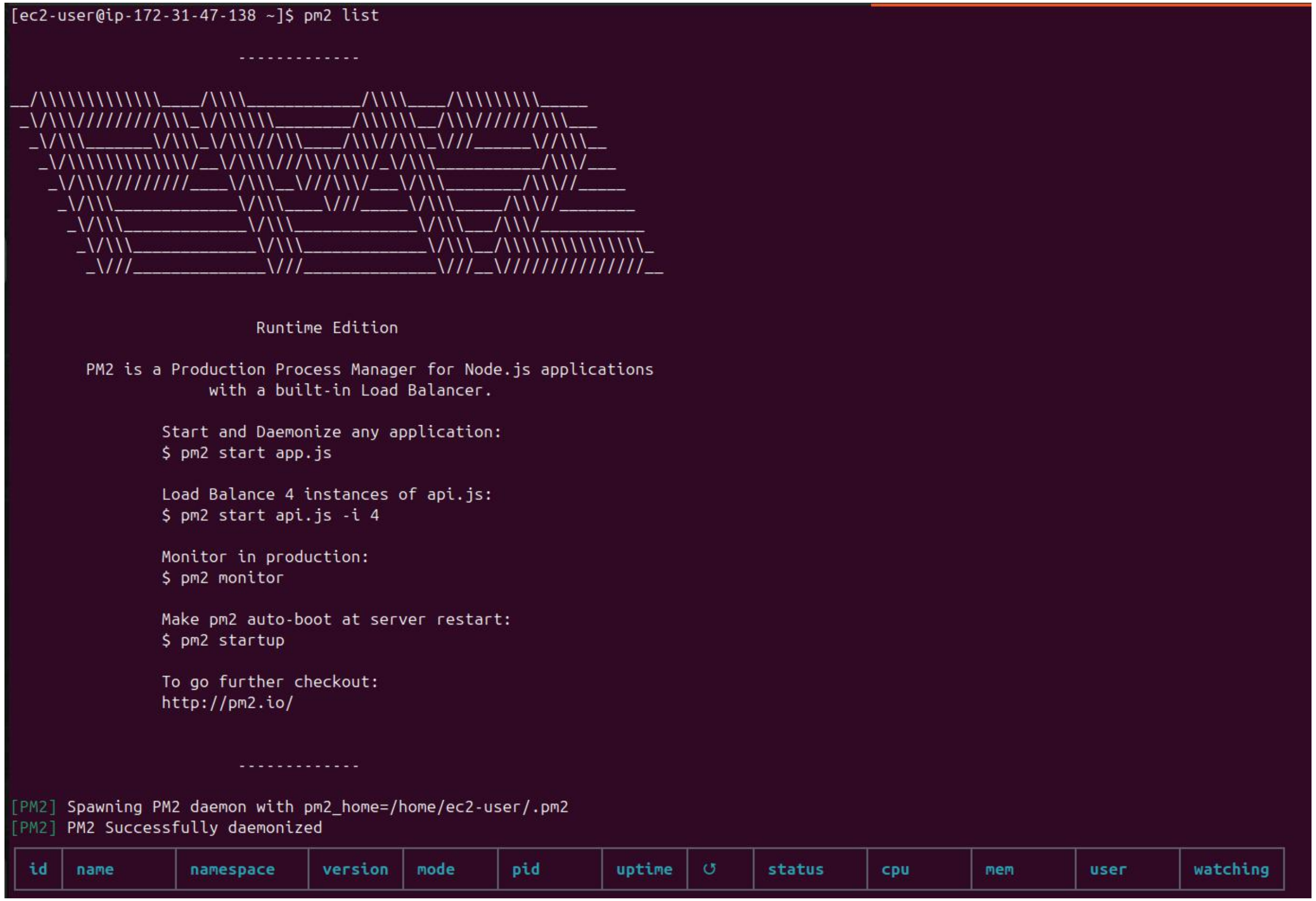


25. Log in to the EC2 server and verify the files are deployed.

```
naveen@5CD20890KH:~/Downloads$ sudo ssh -i "new_keypair.pem" ec2-user@13.201.71.211
[sudo] password for naveen:
       #_
       ####_          Amazon Linux 2023
      _#####\
      \###|
        \#/ ___      https://aws.amazon.com/linux/amazon-linux-2023
         V~' '->
          /
        _/
       _/ _/
     _/m/'
Last login: Wed Aug 13 08:18:49 2025 from 106.51.48.153
[ec2-user@ip-172-31-47-138 ~]$ ls
jenkins
```

```
[ec2-user@ip-172-31-47-138 ~]$ cd jenkins/
[ec2-user@ip-172-31-47-138 jenkins]$ ls
index.js  package.json
[ec2-user@ip-172-31-47-138 jenkins]$ nano index.js
[ec2-user@ip-172-31-47-138 jenkins]$ nano package.json
[ec2-user@ip-172-31-47-138 jenkins]$ ls
index.js  package.json
```

# 26. Check if the Node.js process is running **pm2 list**



```
[ec2-user@ip-172-31-47-138 ~]$ pm2 list

                        -------------

__/\\\\\\\\\\\\\____/\\\\____________/\\\\____/\\\\\\\\\_____
 _\/\\\/////////\\\_\/\\\\\\________/\\\\\\__/\\\///////\\\___
  _\/\\\_______\/\\\_\/\\\//\\\____/\\\//\\\_\///______\//\\\__
   _\/\\\\\\\\\\\\\/__\/\\\\///\\\/\\\/_\/\\\___________/\\\/___
    _\/\\\/////////____\/\\\__\///\\\/___\/\\\________/\\\//_____
     _\/\\\_____________\/\\\____\///_____\/\\\_____/\\\//________
      _\/\\\_____________\/\\\_____________\/\\\___/\\\/___________
       _\/\\\_____________\/\\\_____________\/\\\__/\\\\\\\\\\\\\\\_
        _\///______________\///______________\///__\///////////////__


                          Runtime Edition

        PM2 is a Production Process Manager for Node.js applications
                     with a built-in Load Balancer.

                Start and Daemonize any application:
                $ pm2 start app.js

                Load Balance 4 instances of api.js:
                $ pm2 start api.js -i 4

                Monitor in production:
                $ pm2 monitor

                Make pm2 auto-boot at server restart:
                $ pm2 startup

                To go further checkout:
                http://pm2.io/


                        -------------

[PM2] Spawning PM2 daemon with pm2_home=/home/ec2-user/.pm2
[PM2] PM2 Successfully daemonized
│ id │ name       │ namespace   │ version │ mode    │ pid      │ uptime │ ↺    │ status    │ cpu      │ mem      │ user     │ watching │
```

27. Start the process manually if needed

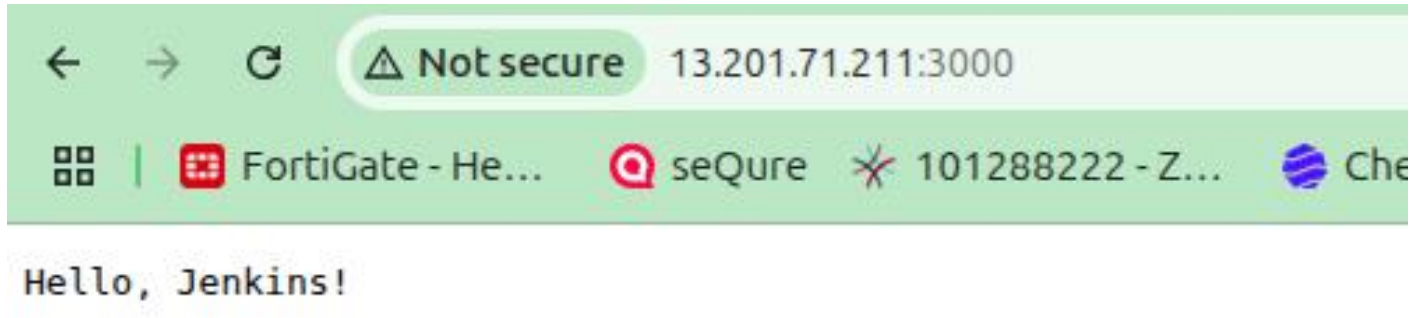
**pm2 start /home/ec2-user/jenkins/index.js --name hello-jenkins**

```
[ec2-user@ip-172-31-47-138 ~]$ pm2 start /home/ec2-user/jenkins/index.js --name hello-jenkins
[PM2] Starting /home/ec2-user/jenkins/index.js in fork_mode (1 instance)
[PM2] Done.
```

| id | name | namespace | version | mode | pid | uptime | ↺ | status | cpu | mem | user | watching |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | hello-jenkins | default | 1.0.0 | fork | 32757 | 0s | 0 | online | 0% | 32.6mb | ec2-user | disabled |

```
[ec2-user@ip-172-31-47-138 ~]$
```

28. Access the Node.js app **http://13.201.71.211:3000**



# Thank You