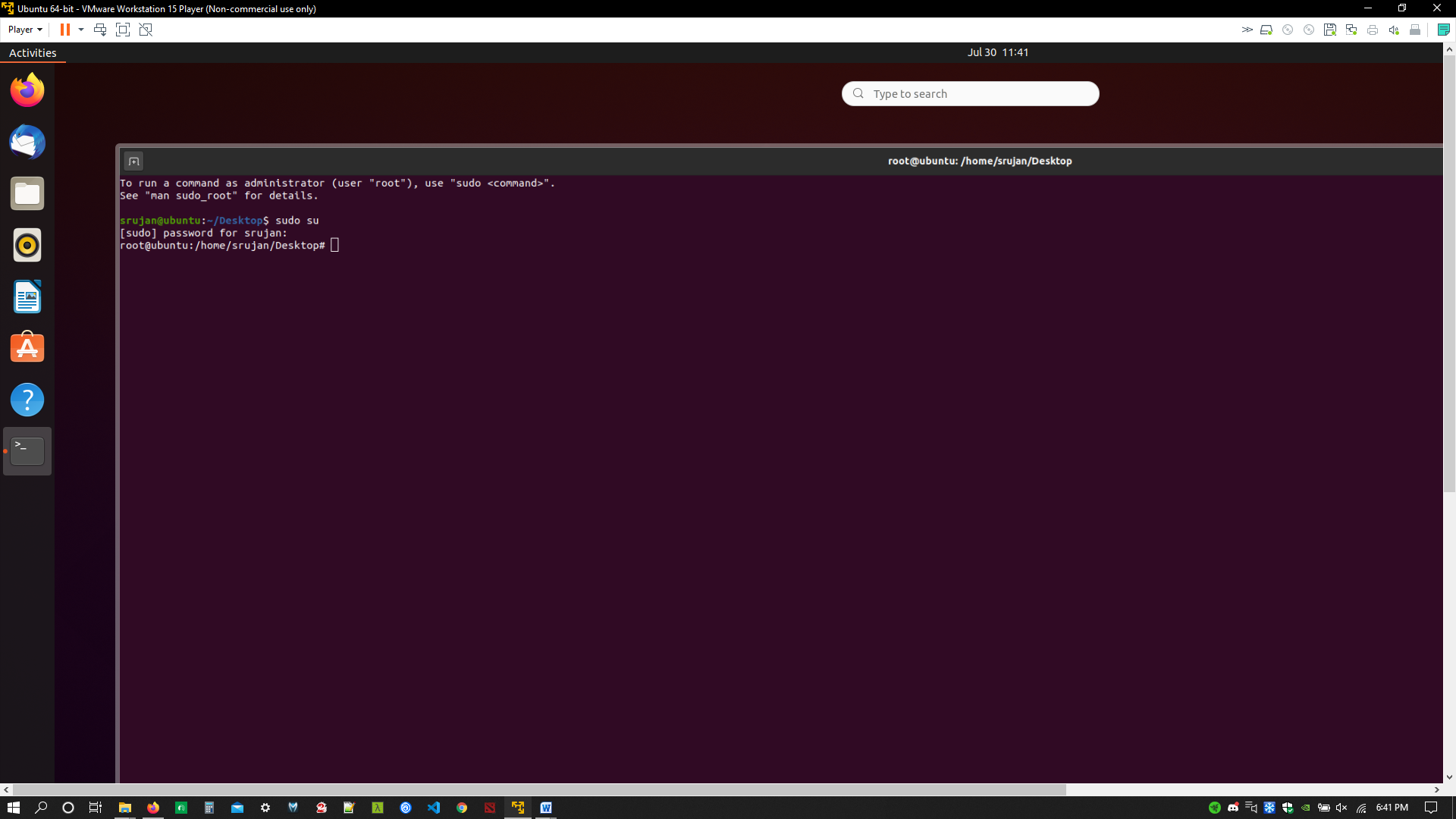
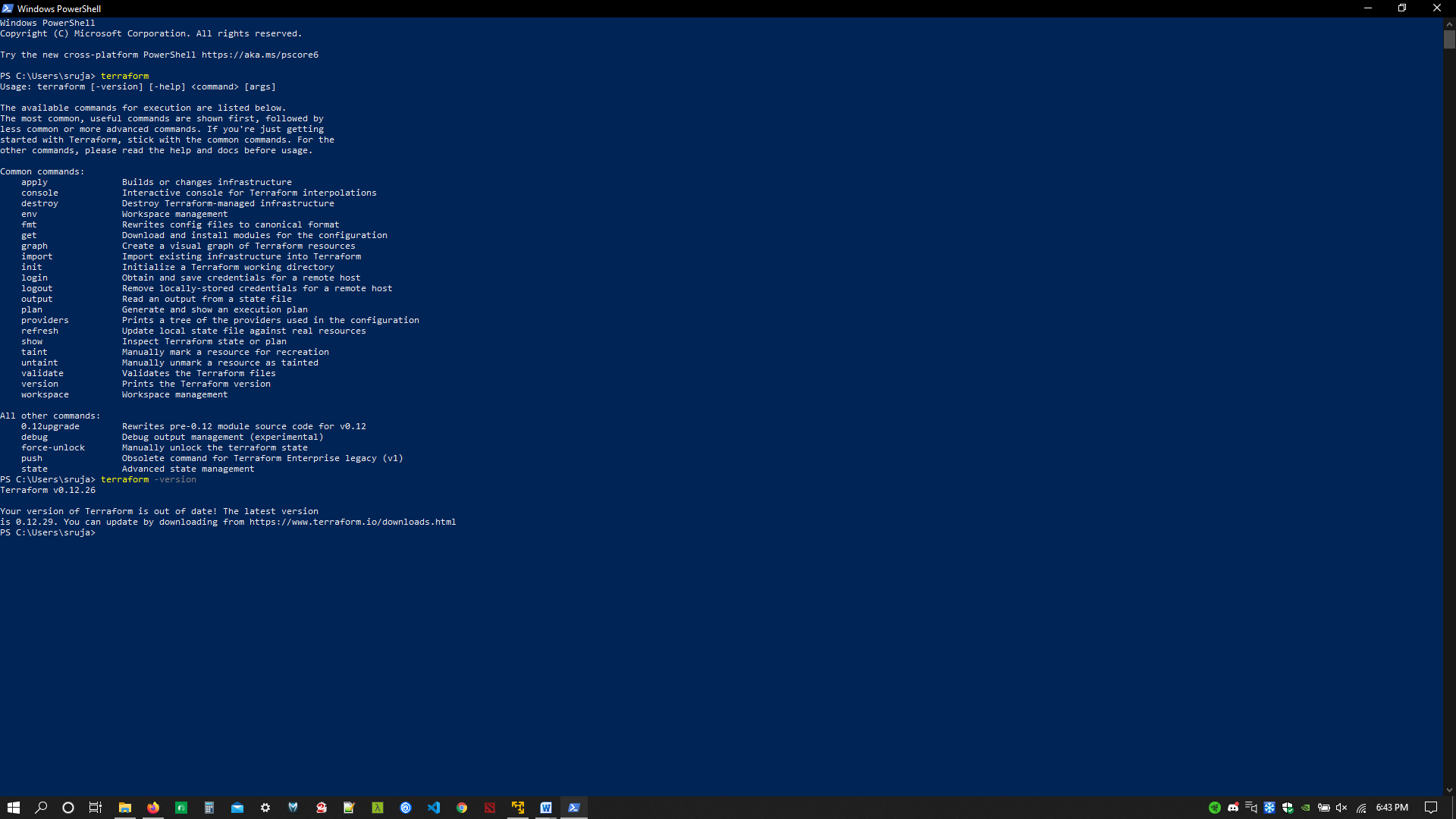
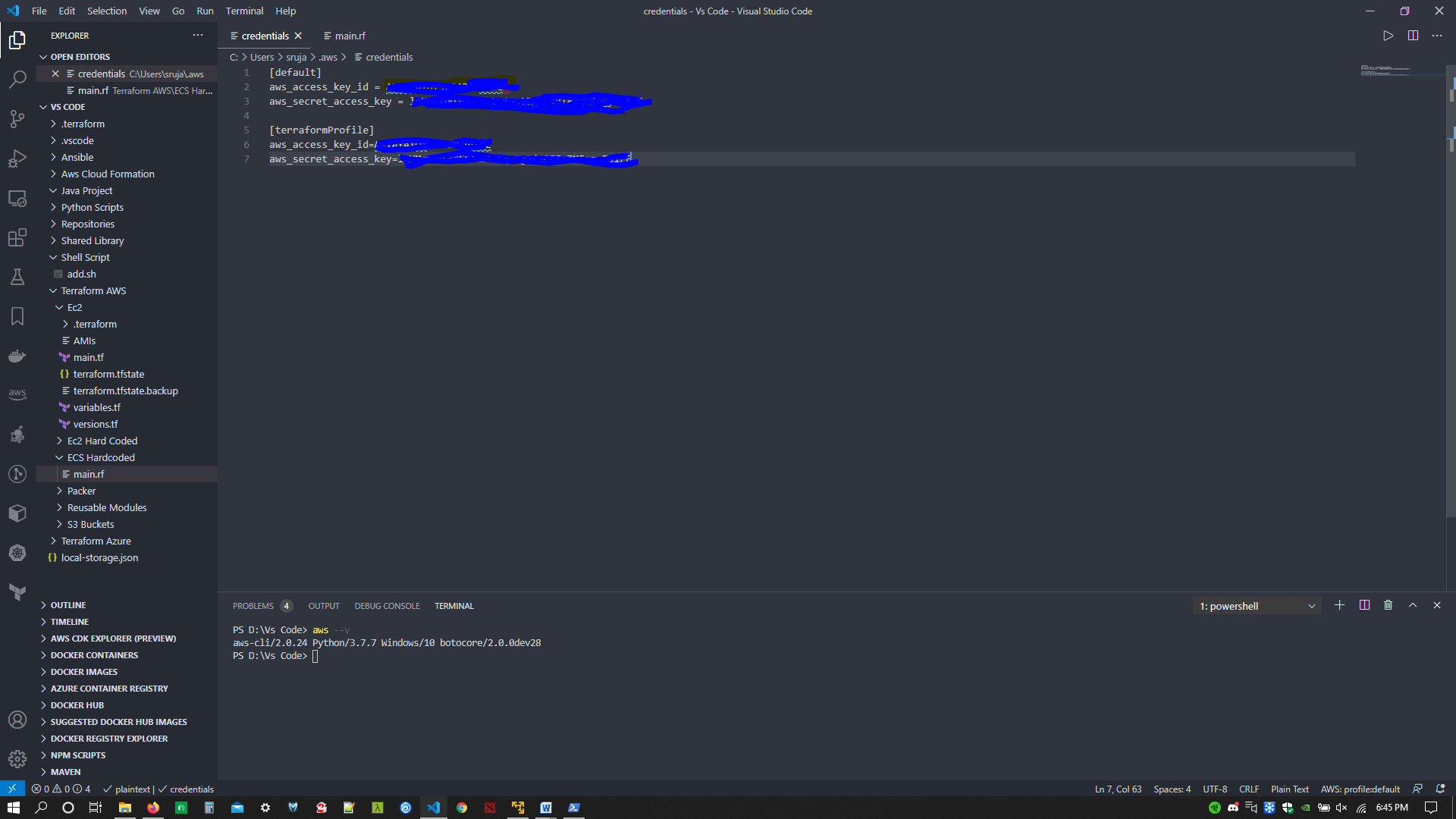
* Git hub Account : <https://github.com/srujantata>
* VM ware with Ubuntu 20.4 LTS Image running



* Terraform installed on windows with path and checked the version as well



* Installed AWS CLI in windows and integrated to VS code.
* Created two users with programmatic access in IAM with admin role and saved the access key and secret key under two different profiles named as default and terraformUser (blackened the key and secret)



* Code for ECS Cluster

provider "aws" {

  region  = "us-west-2"

  version = "~> 2.63"

  profile = "terraformProfile"

}

resource "aws\_ecs\_cluster" "web-cluster" {

  name               = "terraformEcsCluster"

  tags = {

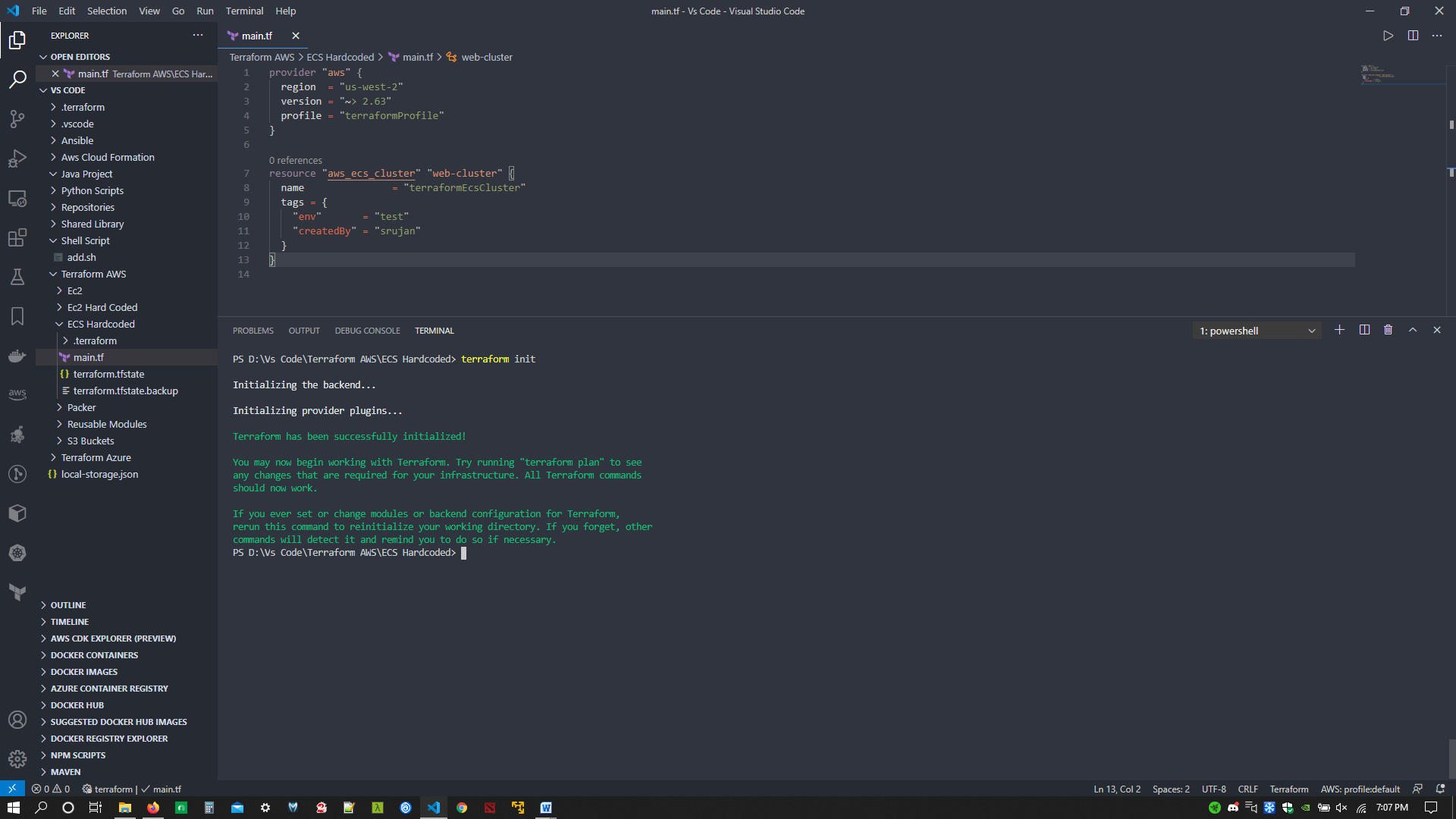
    "env"       = "test"

    "createdBy" = "srujan"

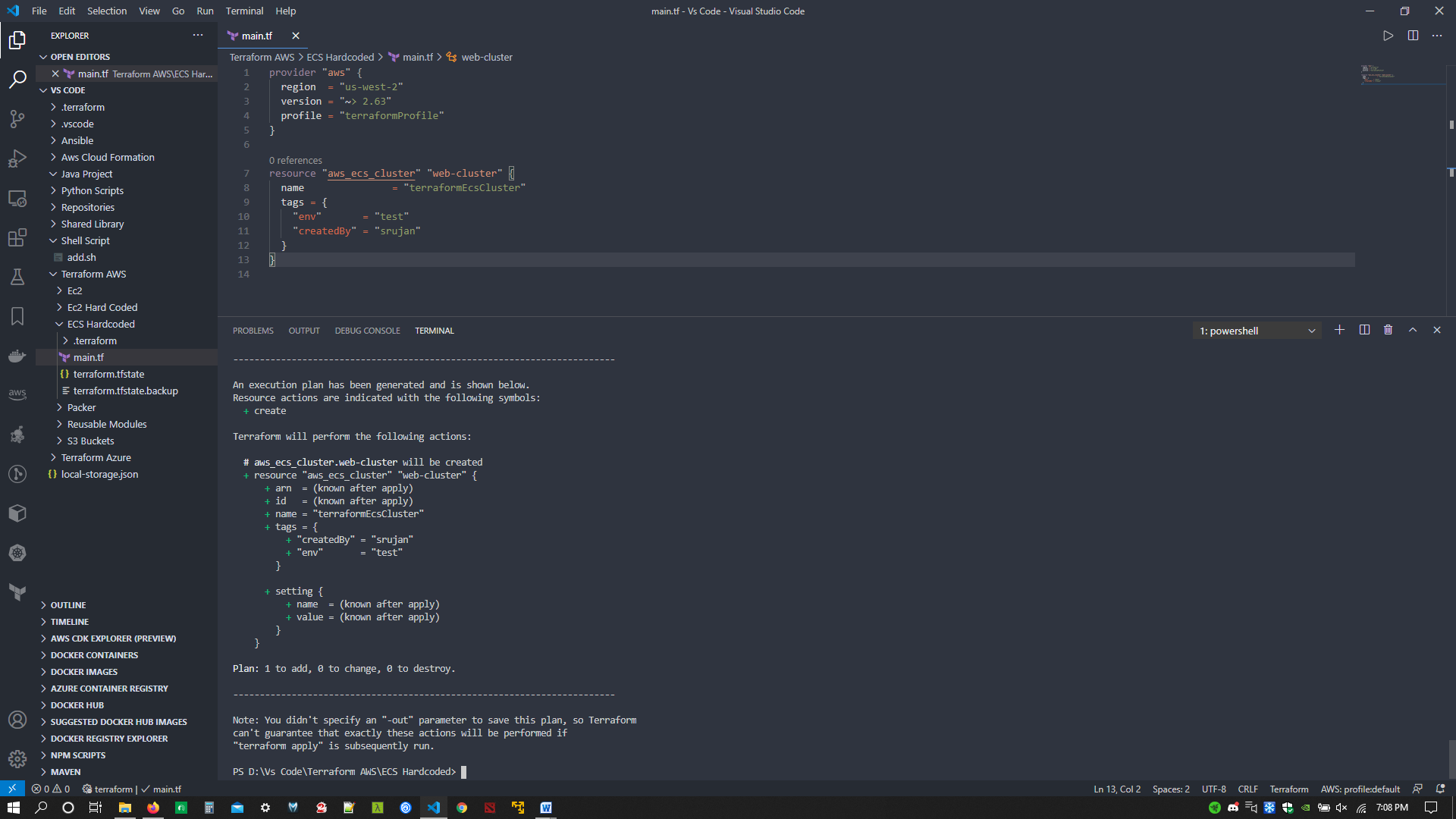
  }

}

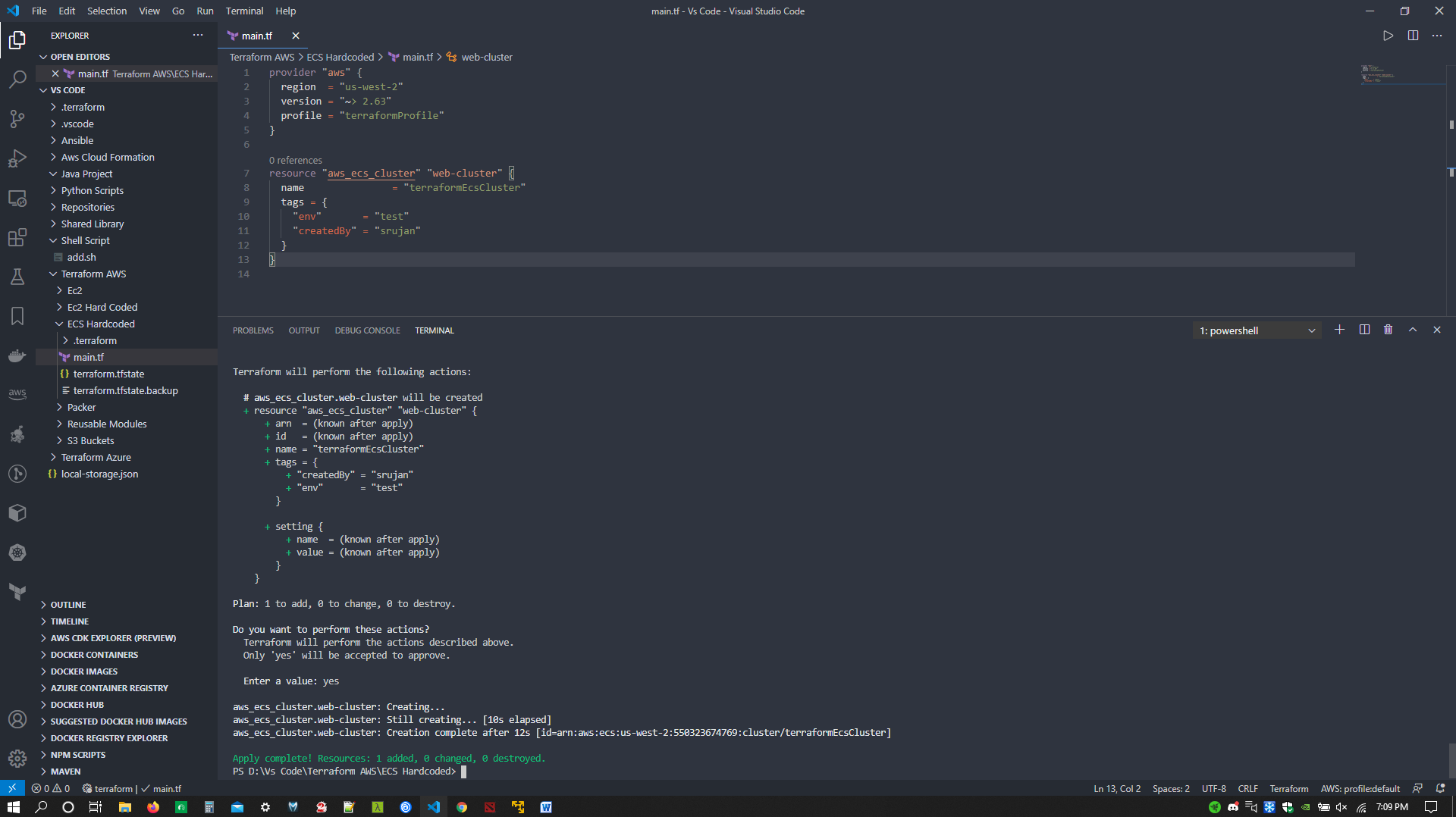
* Terraform initialized



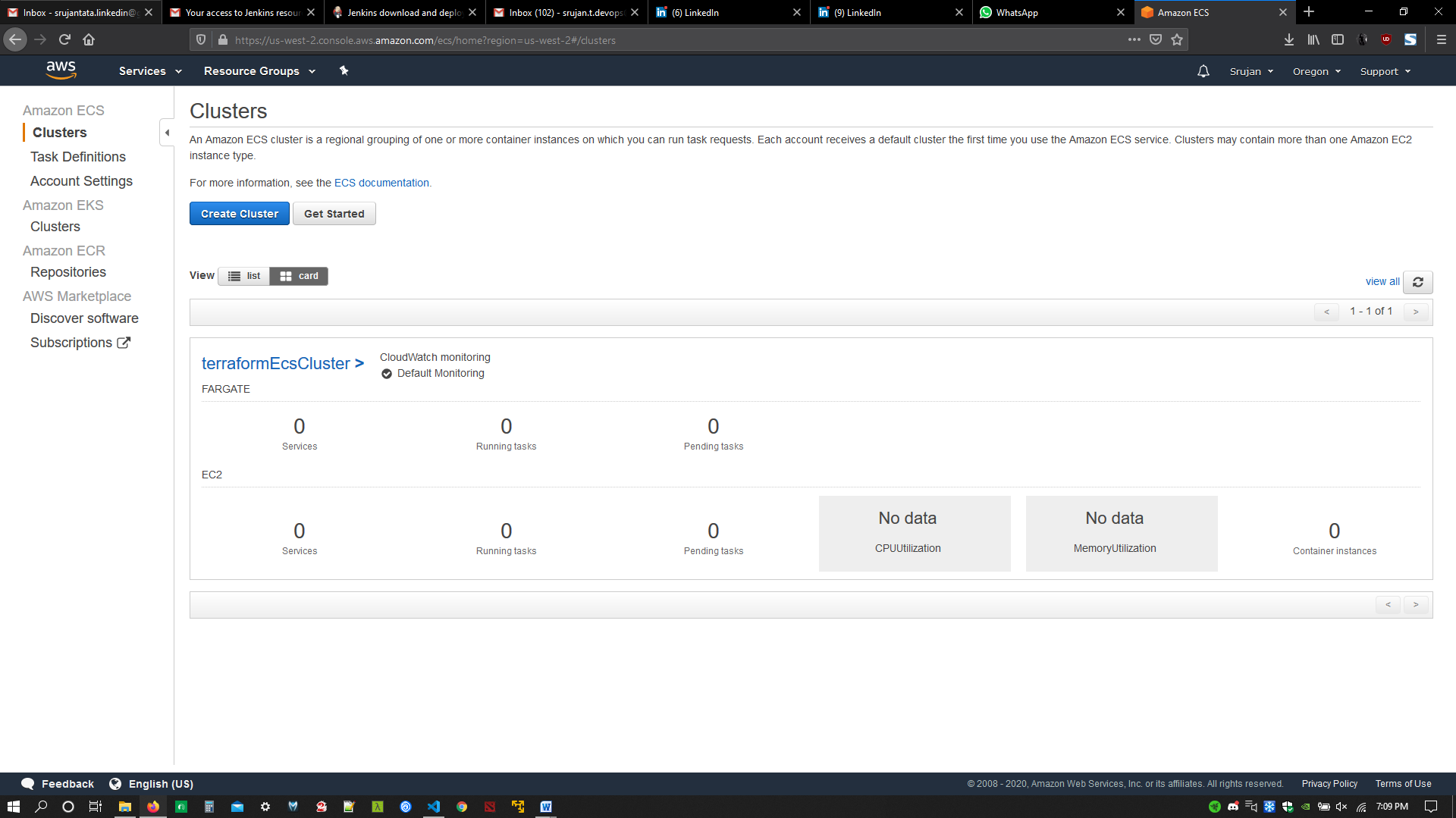
* Terraform plan



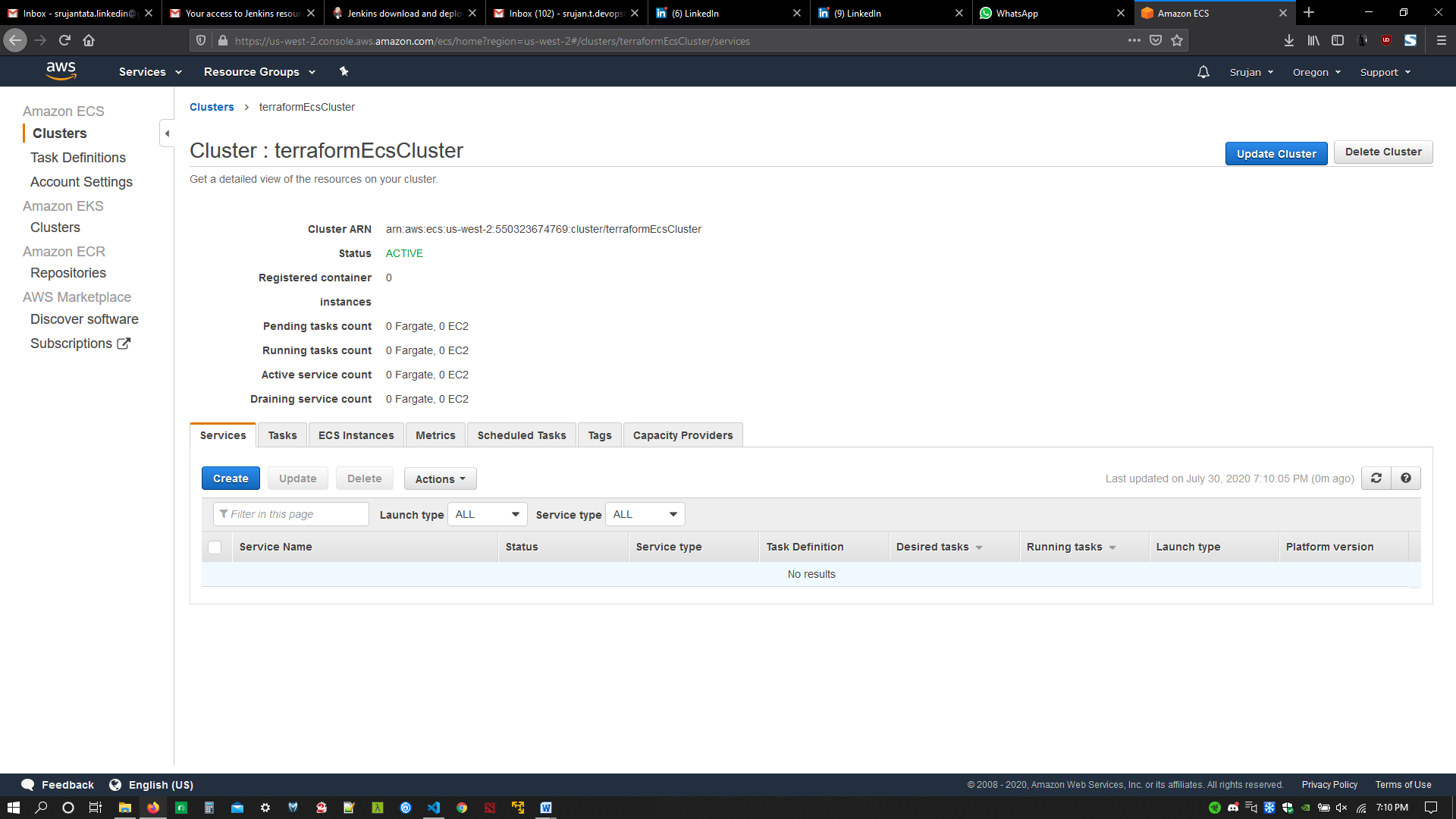
* Terraform apply



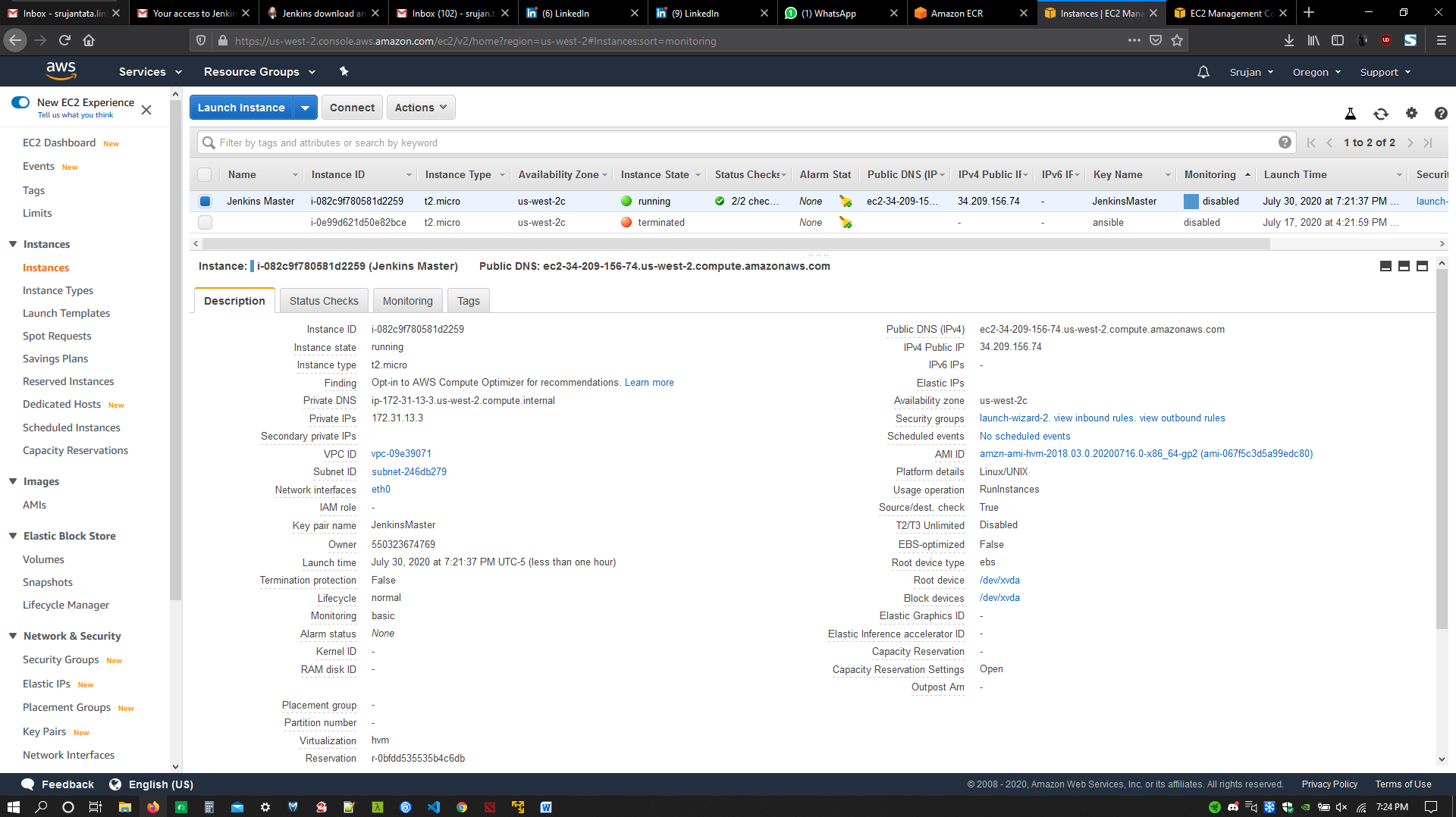
* AWS ECS Cluster created



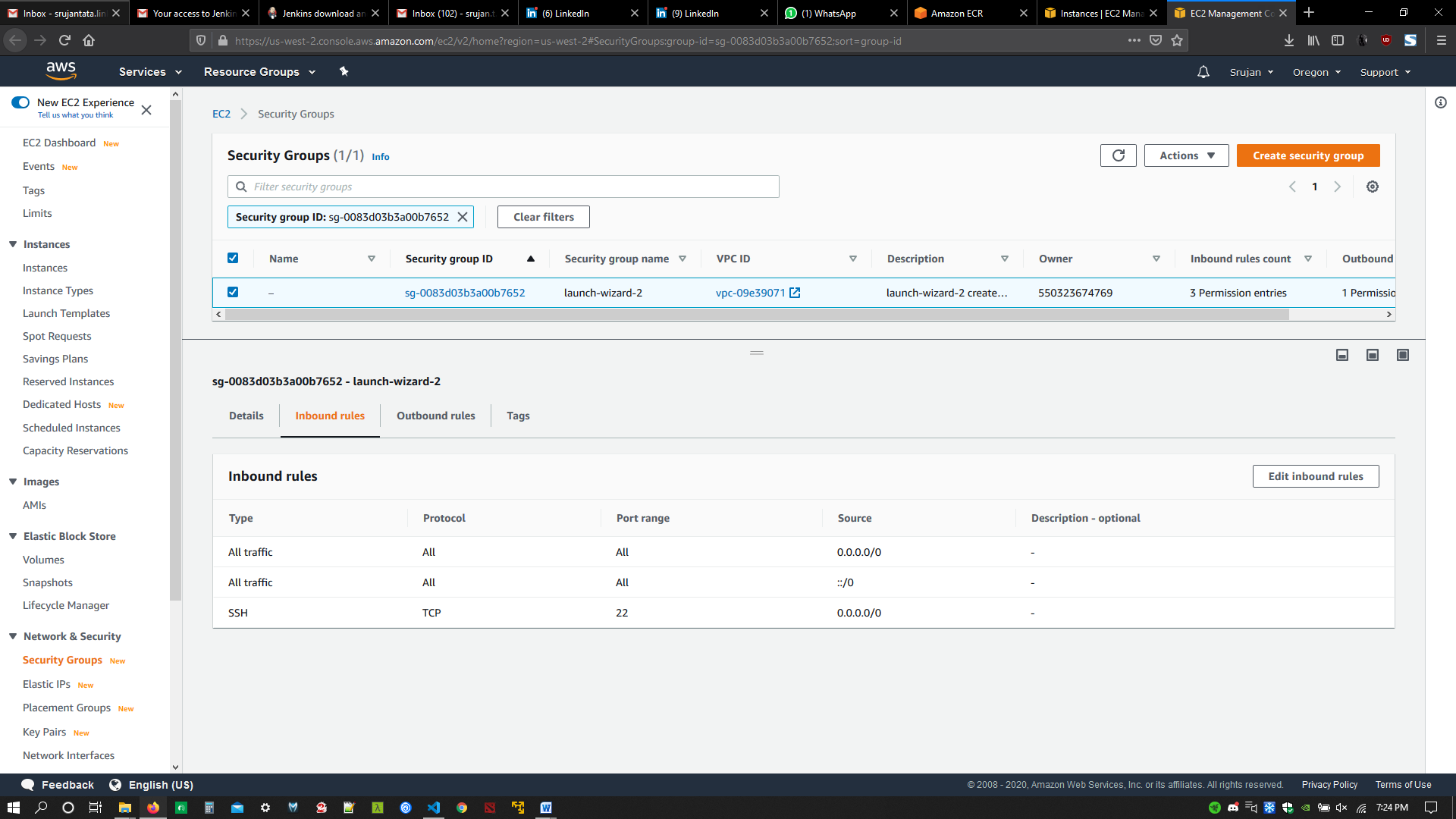
* AWS ECS Cluster configuration



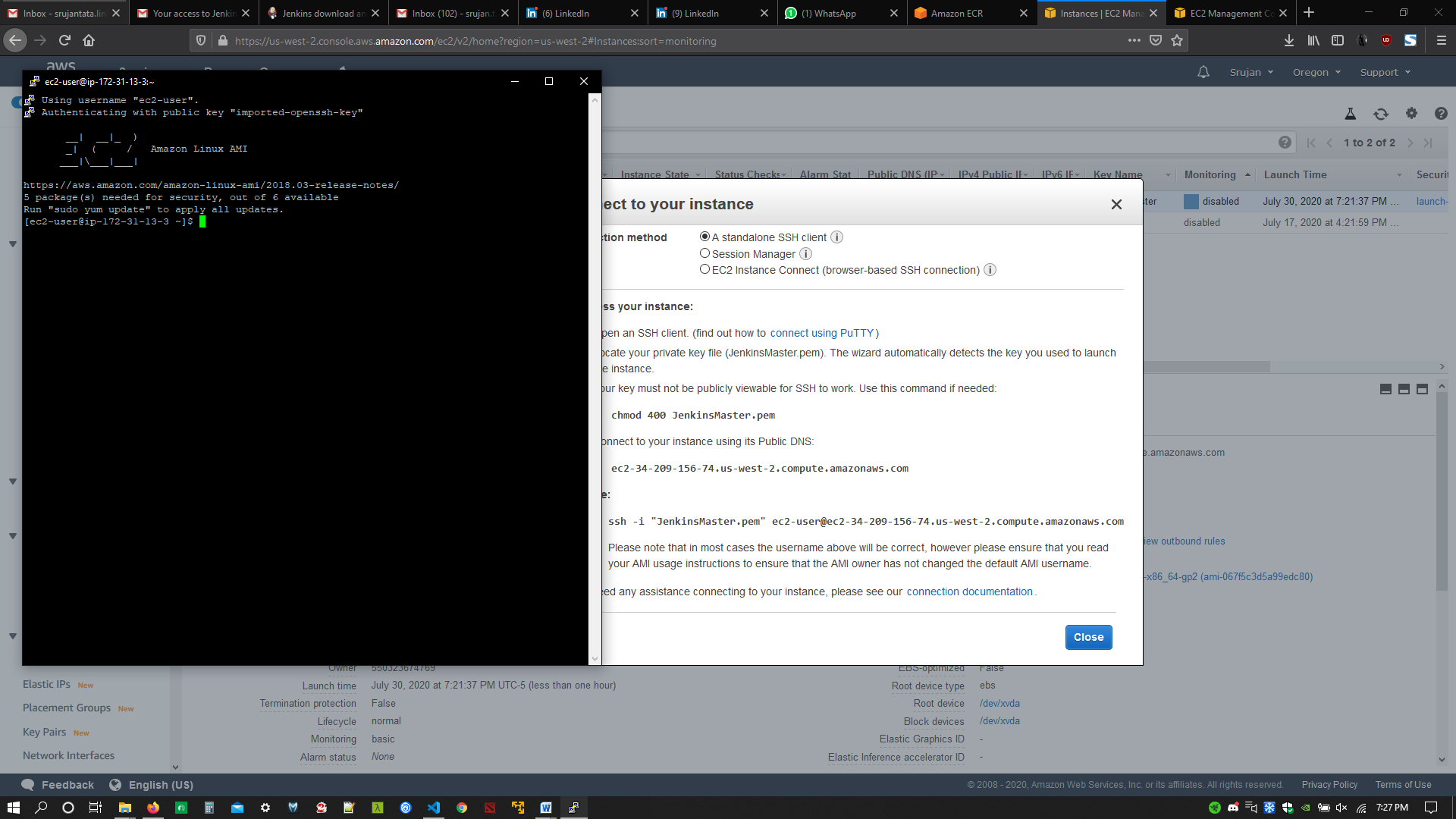
* Created Ec2 Instance with defaults



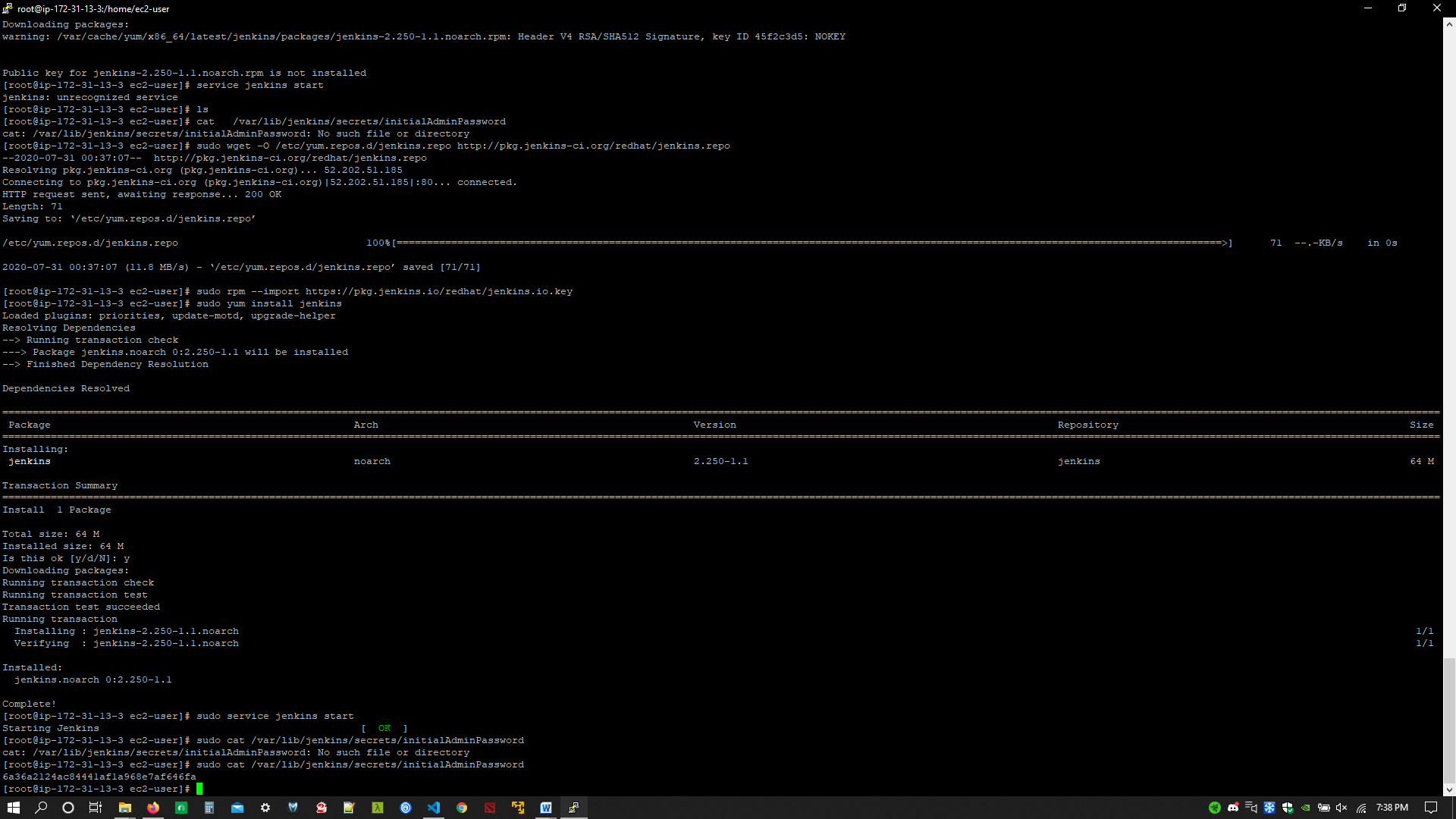
* Modified security groups to allow traffic



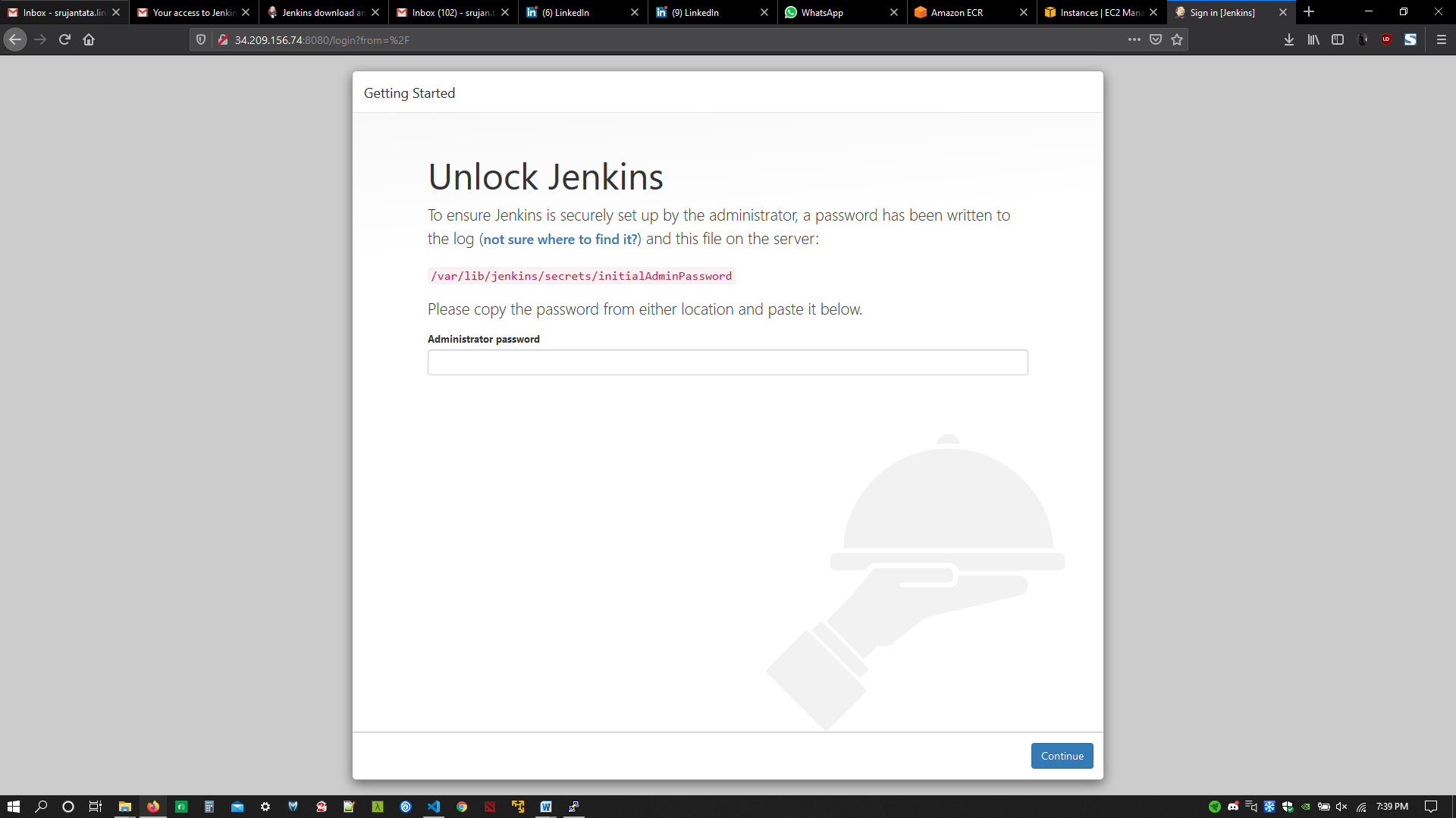
* Converted key and logged into machine using putty and puttygen



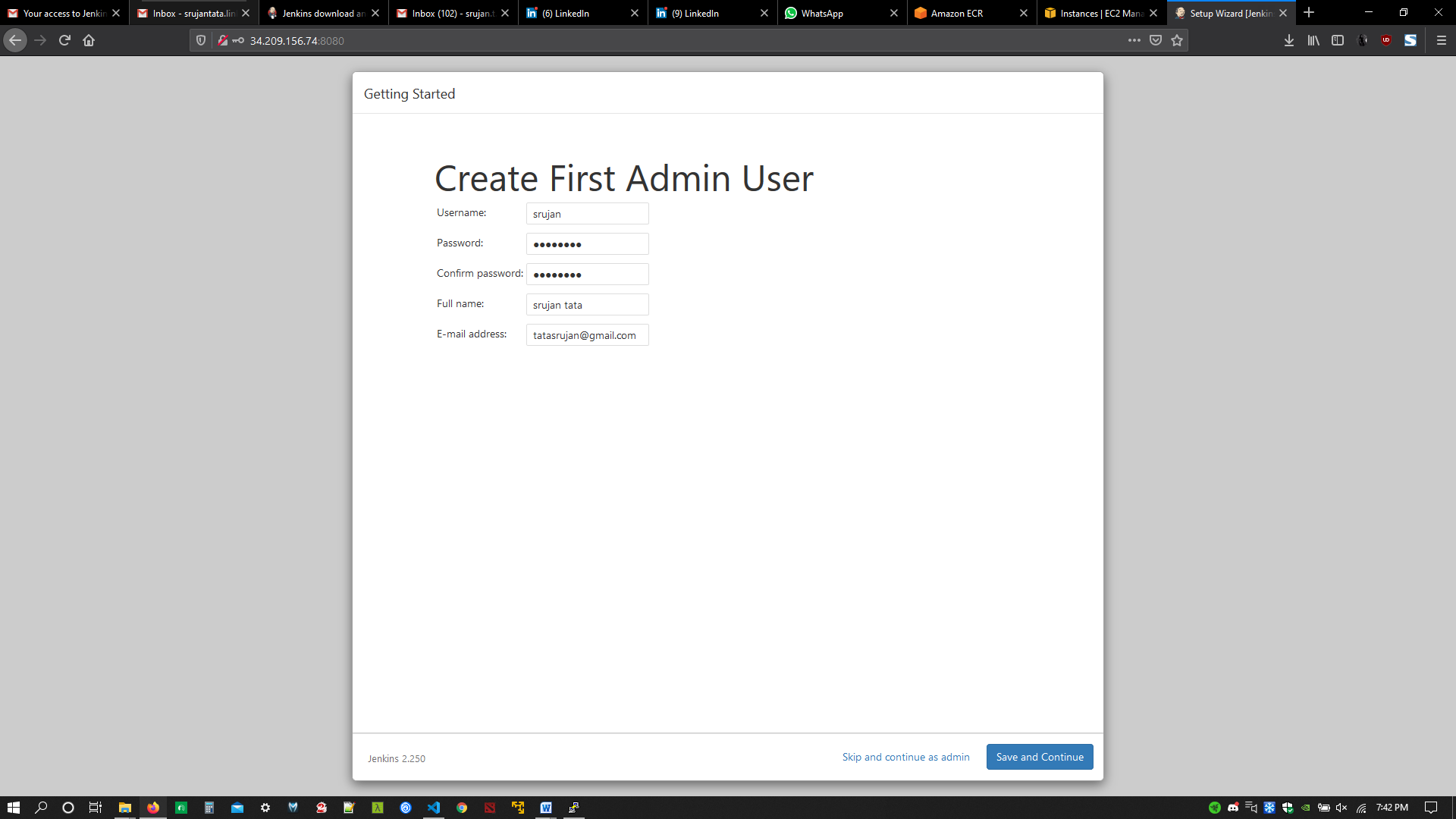
* Installed JDK 1.8 and Jenkins, Started Jenkins service



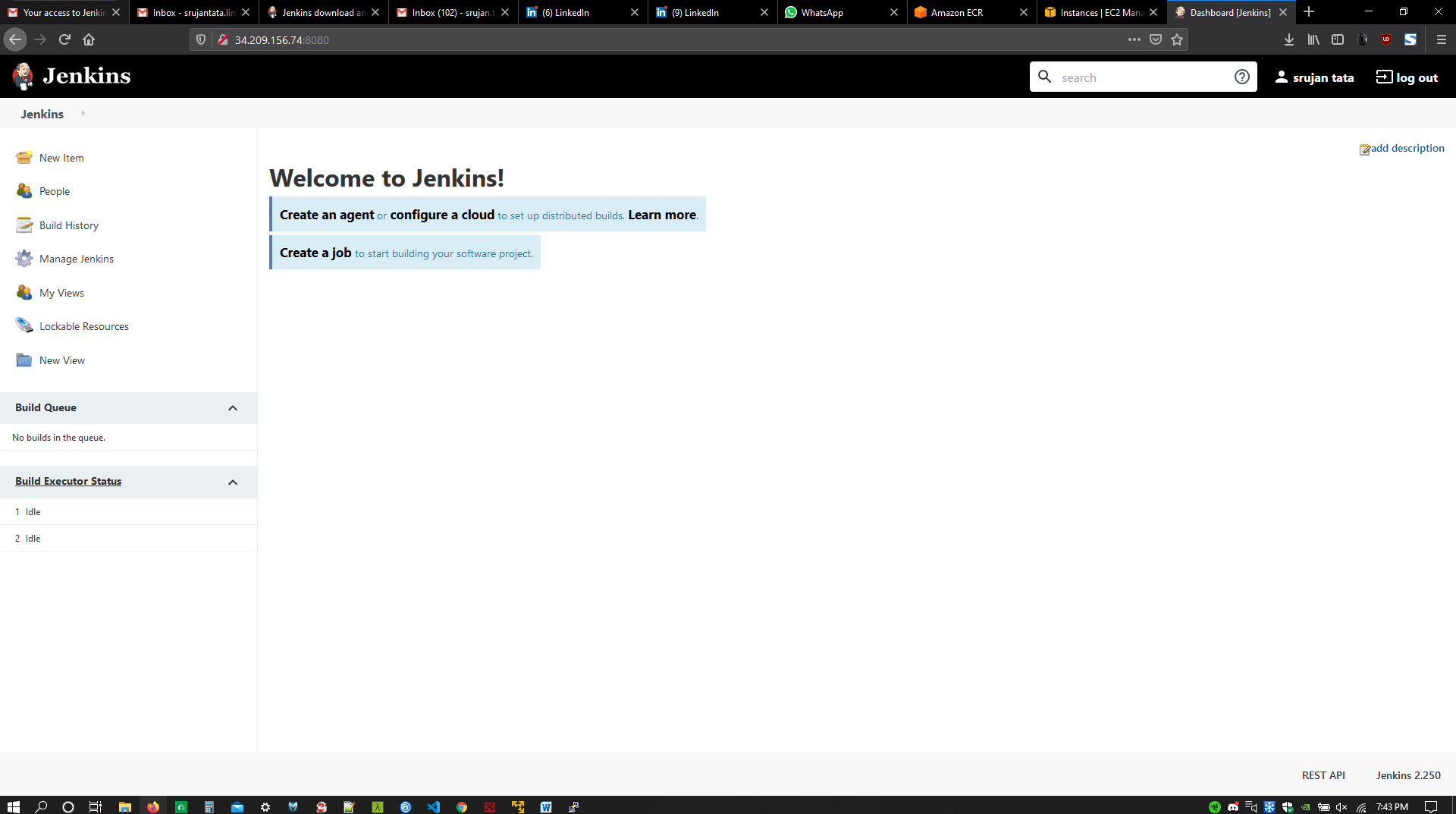
* Jenkins server started in port 8080



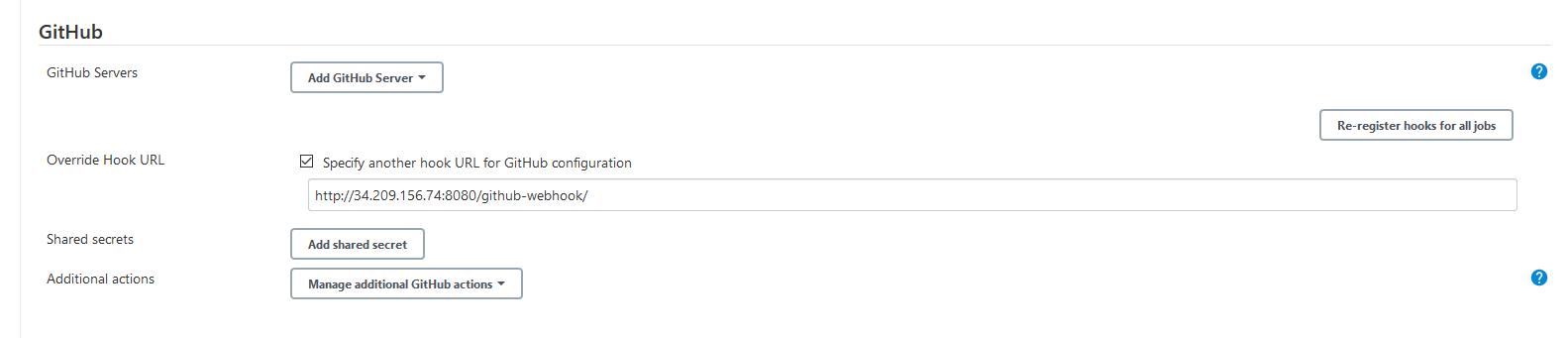
* Unlocked Jenkins, installed suggested plugins and created user



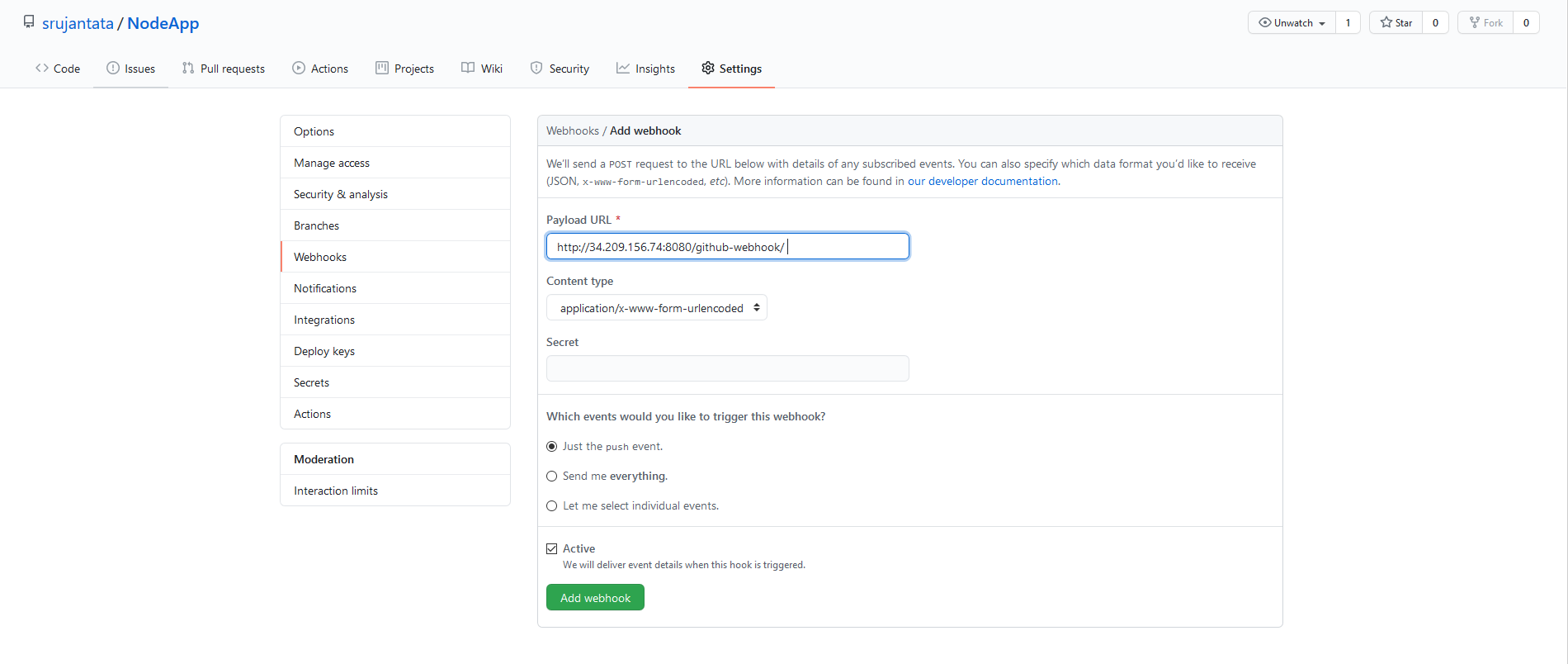
* Jenkins server up and running



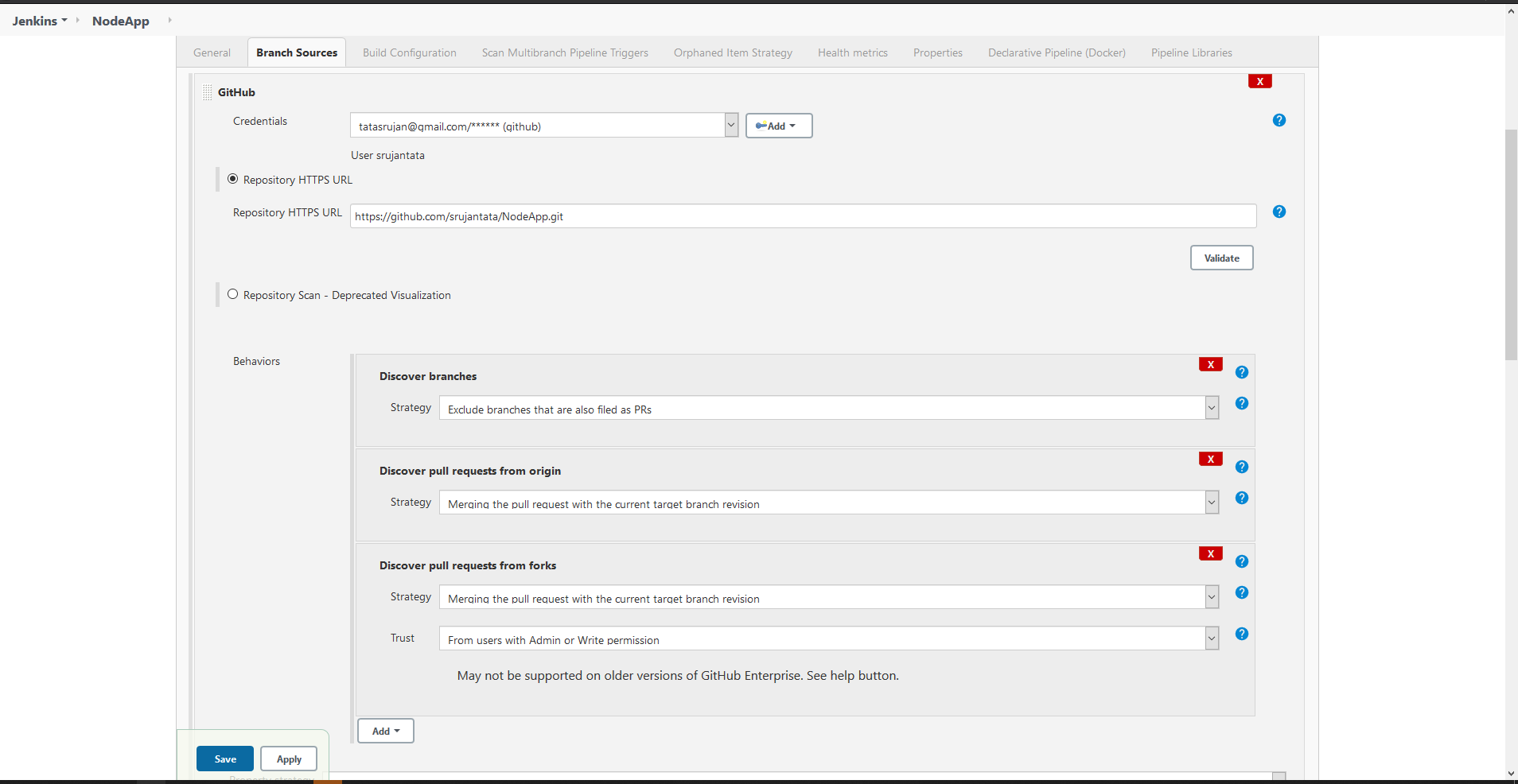
* Pre-requisites for the project, Install npm, docker in ec2 instance where Jenkins is installed and add Jenkins user to docker group to give the Jenkins user required permissions to run docker commands on Jenkins server.
* Clone a working repository to local : <https://github.com/rchidana/NodeApp.git> ( you can choose any working repositories)
* Updated all files by modifying user configuration.
* Updated repository in my GitHub account : <https://github.com/srujantata/NodeApp.git>
* Configured web hook for Jenkins. Go to manage Jenkins > GitHub



* Copy the hook url and paste it under your GitHub account > repository > settings > web hook and add the web hook.



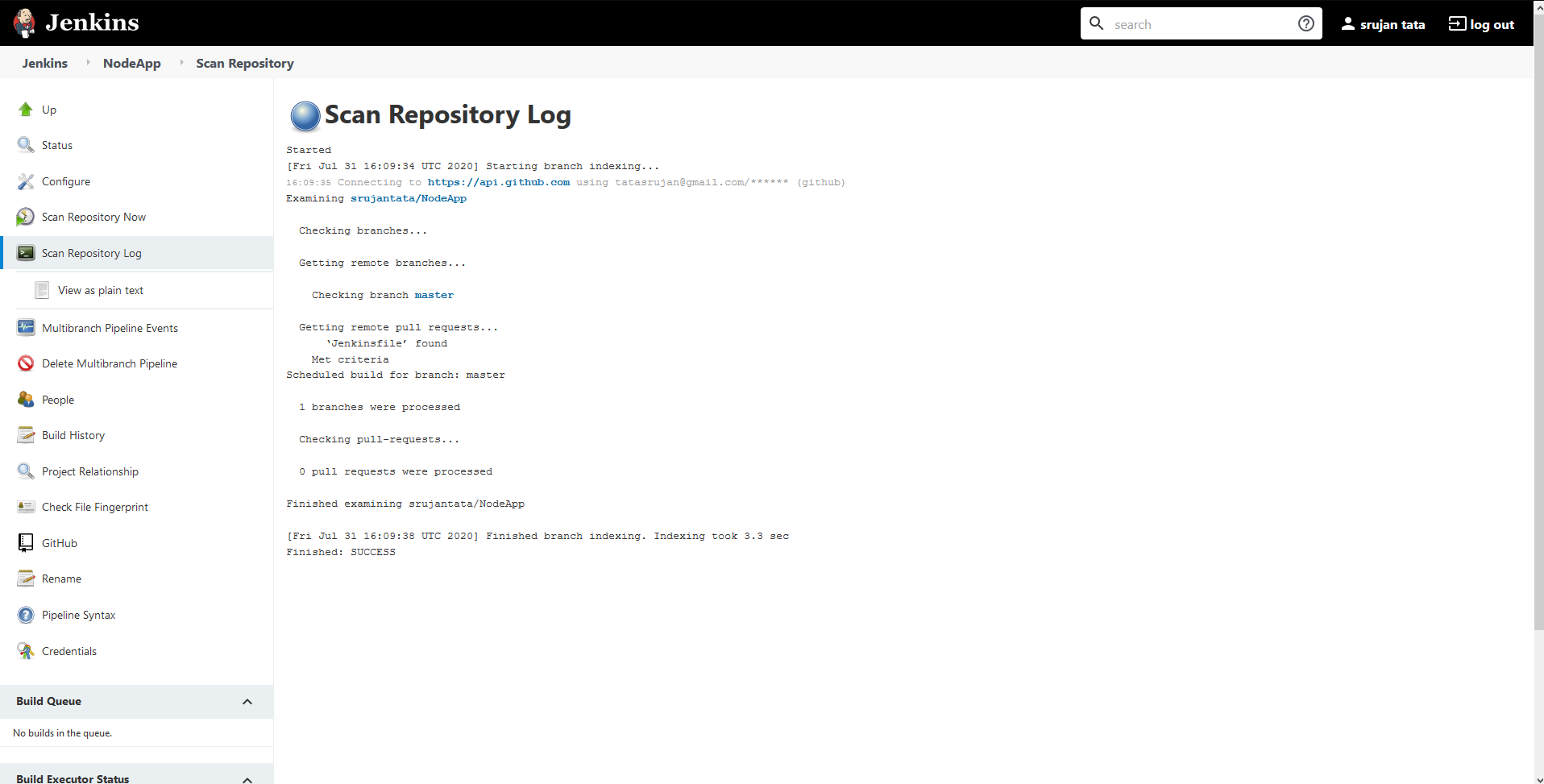
* I used a multibranch pipeline for more features and options.
* Multibranch pipeline required either a script ID or a Jenkinsfile for configuring the branch source and pipeline.
* I have composed my Jenkinsfile and pushed the changes to repository.
* It can be found in my GitHub repository which is public : <https://github.com/srujantata/NodeApp.git>
* pipeline {
* agent any
* stages {
* stage('Pre Cleanup Workspace') {
* steps {
* script {
* sh label: '', script: 'docker system prune -f > /dev/null 2>&1'
* cleanWs()
* }
* }
* }
* stage('Checkout SCM') {
* steps {
* script {
* git credentialsId: 'githubid', url: 'https://github.com/srujantata/NodeApp.git'
* }
* }
* }
* stage('NPM Install') {
* steps {
* script {
* sh label: '', script: 'npm install'
* }
* }
* }
* stage('Docker Build') {
* steps {
* script {
* sh label: '', script: 'docker build -t srujan .'
* }
* }
* }
* stage('Docker Push to ECR') {
* steps {
* script {
* docker.withRegistry('https://550323674769.dkr.ecr.us-west-2.amazonaws.com/srujan:latest', 'ecr:us-west-2:AWSCredentials') {
* docker.image('srujan').push('latest')
* }
* }
* }
* }
* stage('Post Cleanup Workspace') {
* steps {
* script {
* sh label: '', script: 'docker system prune -f > /dev/null 2>&1'
* cleanWs()
* }
* }
* }
* }
* }
* Branch Source configuration



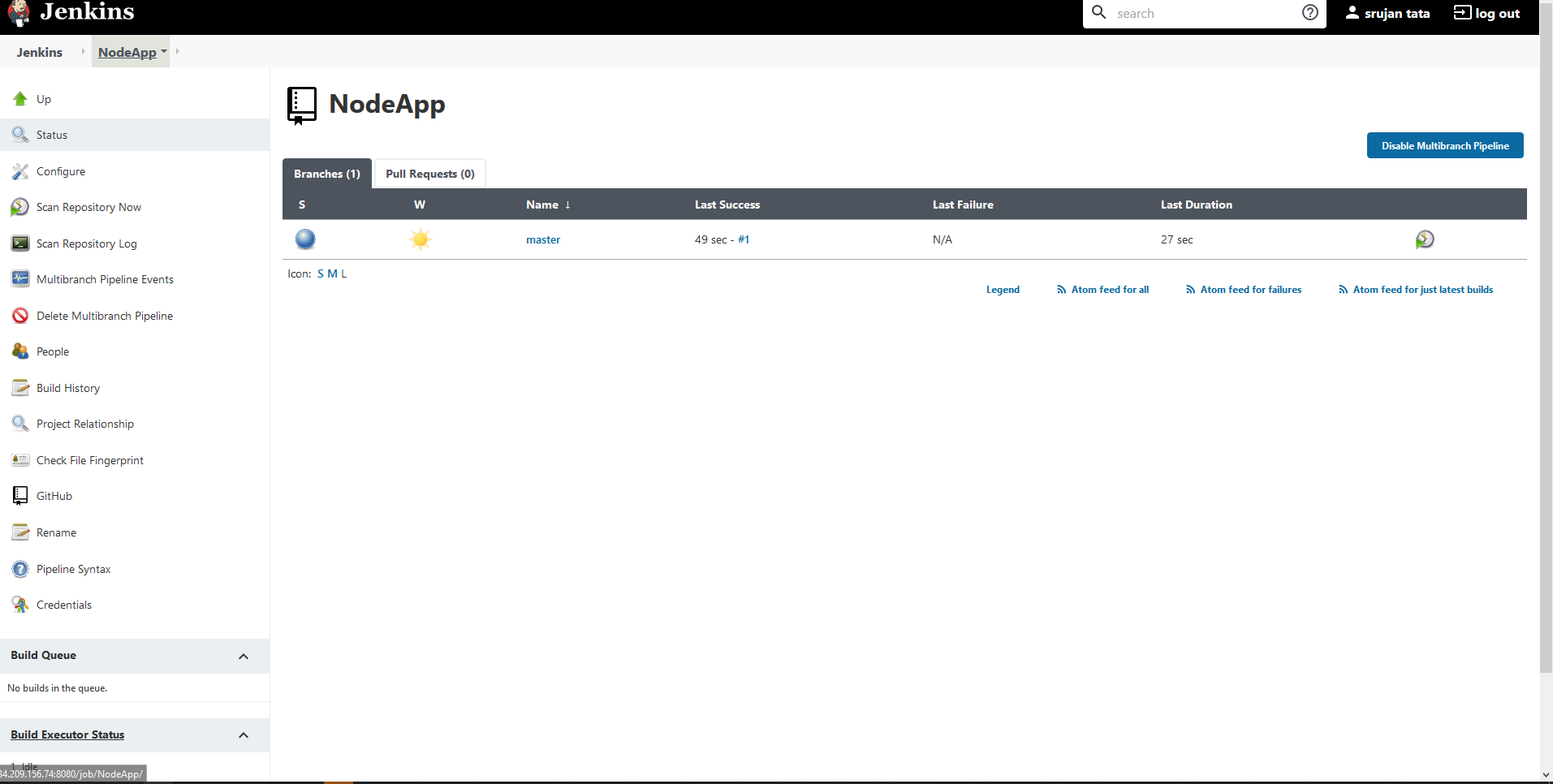
* Configured credentials for GitHub in credential manager.
* Build Configuration uses Jenkinsfile and saved the configuration.



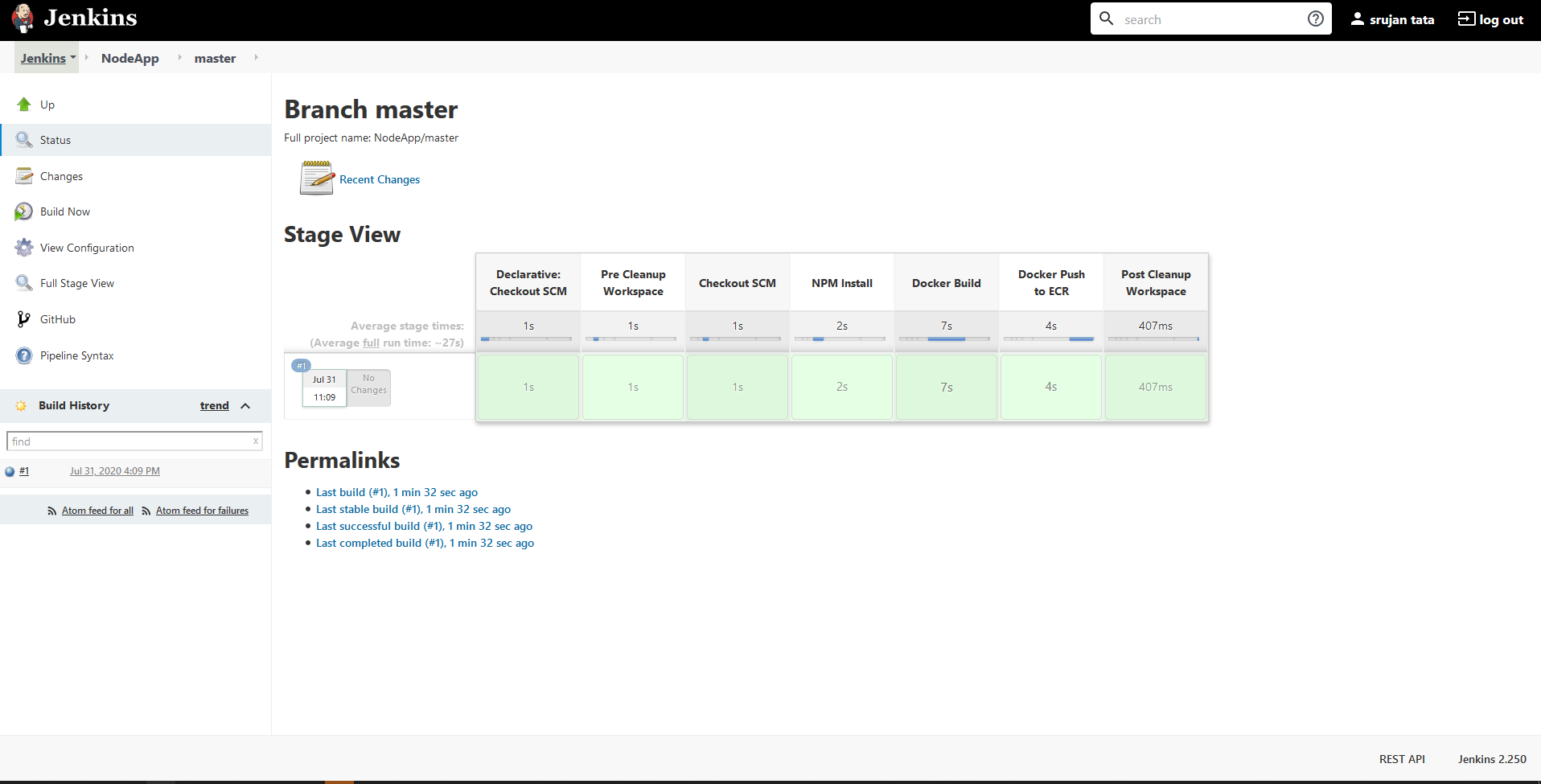
* Jenkins will automatically scan the repository and branch for Jenkinsfile



* Master branch detected with Jenkinsfile



* All the build steps are in the Jenkins pipeline. This is completely automated because I have configured webhook.



* Complete log of Jenkins pipeline:
* Branch indexing
* 16:09:38 Connecting to <https://api.github.com> using tatasrujan@gmail.com/\*\*\*\*\*\* (github)
* Obtained Jenkinsfile from 04717d6a0ab10063e46667df4c3ef7e06a28f17a
* Running in Durability level: MAX\_SURVIVABILITY
* [Pipeline] Start of Pipeline
* [Pipeline] node
* Running on [Jenkins](http://34.209.156.74:8080/computer/(master)/) in /var/lib/jenkins/workspace/NodeApp\_master
* [Pipeline] {
* [Pipeline] stage
* [Pipeline] { (Declarative: Checkout SCM)
* [Pipeline] checkout
* using credential githubid
* Cloning the remote Git repository
* Cloning with configured refspecs honoured and without tags
* Cloning repository <https://github.com/srujantata/NodeApp.git>
* > /usr/bin/git init /var/lib/jenkins/workspace/NodeApp\_master # timeout=10
* Fetching upstream changes from <https://github.com/srujantata/NodeApp.git>
* > /usr/bin/git --version # timeout=10
* using GIT\_ASKPASS to set credentials github
* > /usr/bin/git fetch --no-tags --progress -- <https://github.com/srujantata/NodeApp.git> +refs/heads/master:refs/remotes/origin/master # timeout=10
* > /usr/bin/git config remote.origin.url <https://github.com/srujantata/NodeApp.git> # timeout=10
* > /usr/bin/git config --add remote.origin.fetch +refs/heads/master:refs/remotes/origin/master # timeout=10
* > /usr/bin/git config remote.origin.url <https://github.com/srujantata/NodeApp.git> # timeout=10
* Fetching without tags
* Fetching upstream changes from <https://github.com/srujantata/NodeApp.git>
* using GIT\_ASKPASS to set credentials github
* > /usr/bin/git fetch --no-tags --progress -- <https://github.com/srujantata/NodeApp.git> +refs/heads/master:refs/remotes/origin/master # timeout=10
* Checking out Revision 04717d6a0ab10063e46667df4c3ef7e06a28f17a (master)
* > /usr/bin/git config core.sparsecheckout # timeout=10
* > /usr/bin/git checkout -f 04717d6a0ab10063e46667df4c3ef7e06a28f17a # timeout=10
* Commit message: " update repository"
* First time build. Skipping changelog.
* [Pipeline] }
* [Pipeline] // stage
* [Pipeline] withEnv
* [Pipeline] {
* [Pipeline] stage
* [Pipeline] { (Pre Cleanup Workspace)
* [Pipeline] script
* [Pipeline] {
* [Pipeline] sh
* + docker system prune -f
* [Pipeline] cleanWs
* [WS-CLEANUP] Deleting project workspace...
* [WS-CLEANUP] Deferred wipeout is used...
* [WS-CLEANUP] done
* [Pipeline] }
* [Pipeline] // script
* [Pipeline] }
* [Pipeline] // stage
* [Pipeline] stage
* [Pipeline] { (Checkout SCM)
* [Pipeline] script
* [Pipeline] {
* [Pipeline] git
* using credential githubid
* Cloning the remote Git repository
* Cloning repository <https://github.com/srujantata/NodeApp.git>
* > /usr/bin/git init /var/lib/jenkins/workspace/NodeApp\_master # timeout=10
* Fetching upstream changes from <https://github.com/srujantata/NodeApp.git>
* > /usr/bin/git --version # timeout=10
* using GIT\_ASKPASS to set credentials github
* > /usr/bin/git fetch --tags --progress -- <https://github.com/srujantata/NodeApp.git> +refs/heads/\*:refs/remotes/origin/\* # timeout=10
* > /usr/bin/git config remote.origin.url <https://github.com/srujantata/NodeApp.git> # timeout=10
* > /usr/bin/git config --add remote.origin.fetch +refs/heads/\*:refs/remotes/origin/\* # timeout=10
* > /usr/bin/git config remote.origin.url <https://github.com/srujantata/NodeApp.git> # timeout=10
* Fetching upstream changes from <https://github.com/srujantata/NodeApp.git>
* using GIT\_ASKPASS to set credentials github
* > /usr/bin/git fetch --tags --progress -- <https://github.com/srujantata/NodeApp.git> +refs/heads/\*:refs/remotes/origin/\* # timeout=10
* > /usr/bin/git rev-parse refs/remotes/origin/master^{commit} # timeout=10
* > /usr/bin/git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10
* Checking out Revision 04717d6a0ab10063e46667df4c3ef7e06a28f17a (refs/remotes/origin/master)
* > /usr/bin/git config core.sparsecheckout # timeout=10
* > /usr/bin/git checkout -f 04717d6a0ab10063e46667df4c3ef7e06a28f17a # timeout=10
* > /usr/bin/git branch -a -v --no-abbrev # timeout=10
* > /usr/bin/git checkout -b master 04717d6a0ab10063e46667df4c3ef7e06a28f17a # timeout=10
* Commit message: " update repository"
* [Pipeline] }
* [Pipeline] // script
* [Pipeline] }
* [Pipeline] // stage
* [Pipeline] stage
* [Pipeline] { (NPM Install)
* [Pipeline] script
* [Pipeline] {
* [Pipeline] sh
* + npm install
* npm notice created a lockfile as package-lock.json. You should commit this file.
* added 2 packages from 7 contributors and audited 2 packages in 0.328s
* found 0 vulnerabilities
* [Pipeline] }
* [Pipeline] // script
* [Pipeline] }
* [Pipeline] // stage
* [Pipeline] stage
* [Pipeline] { (Docker Build)
* [Pipeline] script
* [Pipeline] {
* [Pipeline] sh
* + docker build -t srujan .
* Sending build context to Docker daemon 1.844MB
* Step 1/4 : FROM node:7-onbuild
* # Executing 5 build triggers
* ---> Using cache
* ---> Using cache
* ---> Running in 29e9fe7b4412
* [91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mit worked if it ends with[0m[91m ok
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91musing[0m[91m npm@4.2.0
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91musing[0m[91m node@v7.10.1
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mattempt[0m[91m registry request try #1 at 4:09:56 PM
* [0m[91mnpm[0m[91m [0m[91mhttp[0m[91m [0m[91mrequest[0m[91m GET <https://registry.npmjs.org/test>
* [0m[91mnpm[0m[91m [0m[91mhttp[0m[91m [0m[91m200[0m[91m <https://registry.npmjs.org/test>
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91maddNameTag[0m[91m [ 'test', 'latest' ]
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mretry[0m[91m fetch attempt 1 at 4:09:56 PM
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mattempt[0m[91m registry request try #1 at 4:09:56 PM
* [0m[91mnpm[0m[91m [0m[91mhttp[0m[91m [0m[91mfetch[0m[91m GET <https://registry.npmjs.org/test/-/test-0.6.0.tgz>
* [0m[91mnpm[0m[91m [0m[91mhttp[0m[91m [0m[91mfetch[0m[91m 200 <https://registry.npmjs.org/test/-/test-0.6.0.tgz>
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mattempt[0m[91m registry request try #1 at 4:09:56 PM
* [0m[91mnpm[0m[91m [0m[91mhttp[0m[91m [0m[91mrequest[0m[91m GET <https://registry.npmjs.org/ansi-font>
* [0m[91mnpm[0m[91m [0m[91mhttp[0m[91m [0m[91m200[0m[91m <https://registry.npmjs.org/ansi-font>
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mretry[0m[91m fetch attempt 1 at 4:09:56 PM
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mattempt[0m[91m registry request try #1 at 4:09:56 PM
* [0m[91mnpm[0m[91m [0m[91mhttp[0m[91m [0m[91mfetch[0m[91m GET <https://registry.npmjs.org/ansi-font/-/ansi-font-0.0.2.tgz>
* [0m[91mnpm[0m[91m [0m[91mhttp[0m[91m [0m[91mfetch[0m[91m 200 <https://registry.npmjs.org/ansi-font/-/ansi-font-0.0.2.tgz>
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mlifecycle[0m[91m crudsinfotech-nodeapp@1.0.0~preinstall: crudsinfotech-nodeapp@1.0.0
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mlifecycle[0m[91m ansi-font@0.0.2~preinstall: ansi-font@0.0.2
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mlifecycle[0m[91m test@0.6.0~preinstall: test@0.6.0
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mlinkStuff[0m[91m ansi-font@0.0.2
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mlinkStuff[0m[91m test@0.6.0
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mlifecycle[0m[91m ansi-font@0.0.2~install: ansi-font@0.0.2
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mlifecycle[0m[91m test@0.6.0~install: test@0.6.0
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mlifecycle[0m[91m ansi-font@0.0.2~postinstall: ansi-font@0.0.2
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mlifecycle[0m[91m test@0.6.0~postinstall: test@0.6.0
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mlinkStuff[0m[91m crudsinfotech-nodeapp@1.0.0
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mlifecycle[0m[91m crudsinfotech-nodeapp@1.0.0~install: crudsinfotech-nodeapp@1.0.0
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mlifecycle[0m[91m crudsinfotech-nodeapp@1.0.0~postinstall: crudsinfotech-nodeapp@1.0.0
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mlifecycle[0m[91m crudsinfotech-nodeapp@1.0.0~prepublish: crudsinfotech-nodeapp@1.0.0
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mlifecycle[0m[91m crudsinfotech-nodeapp@1.0.0~prepare: crudsinfotech-nodeapp@1.0.0
* [0mcrudsinfotech-nodeapp@1.0.0 /usr/src/app
* `-- test@0.6.0
* `-- ansi-font@0.0.2
* [91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mok[0m[91m
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mit worked if it ends with[0m[91m ok
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91musing[0m[91m npm@4.2.0
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91musing[0m[91m node@v7.10.1
* [0m[91mnpm[0m[91m [0m[91mWARN[0m[91m [0m[91musing --force[0m[91m I sure hope you know what you are doing.
* [0m[91mnpm[0m[91m [0m[91minfo[0m[91m [0m[91mok[0m[91m
* [0mRemoving intermediate container 29e9fe7b4412
* ---> 93fadab3a4a2
* Step 2/4 : LABEL maintainer "tatasrujan@gmail.com"
* ---> Running in 2f1f5264c777
* Removing intermediate container 2f1f5264c777
* ---> 6c93aa4d51d8
* Step 3/4 : HEALTHCHECK --interval=5s --timeout=5s CMD curl -f <http://127.0.0.1:8000> || exit 1
* ---> Running in 52461d9bdf6f
* Removing intermediate container 52461d9bdf6f
* ---> 1ddc14134f49
* Step 4/4 : EXPOSE 8000
* ---> Running in 7a644c0cce3b
* Removing intermediate container 7a644c0cce3b
* ---> cb9f72701aef
* Successfully built cb9f72701aef
* Successfully tagged srujan:latest
* [Pipeline] }
* [Pipeline] // script
* [Pipeline] }
* [Pipeline] // stage
* [Pipeline] stage
* [Pipeline] { (Docker Push to ECR)
* [Pipeline] script
* [Pipeline] {
* [Pipeline] withEnv
* [Pipeline] {
* [Pipeline] withDockerRegistry
* $ docker login -u AWS -p \*\*\*\*\*\*\*\* <https://550323674769.dkr.ecr.us-west-2.amazonaws.com/srujan:latest>
* WARNING! Using --password via the CLI is insecure. Use --password-stdin.
* WARNING! Your password will be stored unencrypted in /var/lib/jenkins/workspace/NodeApp\_master@tmp/214e7f64-c396-4766-9bd0-d78ad93ff42e/config.json.
* Configure a credential helper to remove this warning. See
* <https://docs.docker.com/engine/reference/commandline/login/#credentials-store>
* Login Succeeded
* [Pipeline] {
* [Pipeline] isUnix
* [Pipeline] sh
* + docker tag srujan 550323674769.dkr.ecr.us-west-2.amazonaws.com/srujan:latest
* [Pipeline] isUnix
* [Pipeline] sh
* + docker push 550323674769.dkr.ecr.us-west-2.amazonaws.com/srujan:latest
* The push refers to repository [550323674769.dkr.ecr.us-west-2.amazonaws.com/srujan]
* 693b499f2487: Preparing
* c1b9a7a2f613: Preparing
* 10ee67290052: Preparing
* 2895be281ac1: Preparing
* ab90d83fa34a: Preparing
* 8ee318e54723: Preparing
* e6695624484e: Preparing
* da59b99bbd3b: Preparing
* 5616a6292c16: Preparing
* f3ed6cb59ab0: Preparing
* 654f45ecb7e3: Preparing
* 2c40c66f7667: Preparing
* 8ee318e54723: Waiting
* e6695624484e: Waiting
* da59b99bbd3b: Waiting
* 5616a6292c16: Waiting
* f3ed6cb59ab0: Waiting
* 654f45ecb7e3: Waiting
* 2c40c66f7667: Waiting
* 2895be281ac1: Layer already exists
* ab90d83fa34a: Layer already exists
* 8ee318e54723: Layer already exists
* e6695624484e: Layer already exists
* da59b99bbd3b: Layer already exists
* 5616a6292c16: Layer already exists
* f3ed6cb59ab0: Layer already exists
* 654f45ecb7e3: Layer already exists
* 2c40c66f7667: Layer already exists
* 10ee67290052: Pushed
* c1b9a7a2f613: Pushed
* 693b499f2487: Pushed
* latest: digest: sha256:0ab0509763ddd98345fc158a27c5e3dbbc63c1d112085eb551ec2202ace92338 size: 2841
* [Pipeline] }
* [Pipeline] // withDockerRegistry
* [Pipeline] }
* [Pipeline] // withEnv
* [Pipeline] }
* [Pipeline] // script
* [Pipeline] }
* [Pipeline] // stage
* [Pipeline] stage
* [Pipeline] { (Post Cleanup Workspace)
* [Pipeline] script
* [Pipeline] {
* [Pipeline] sh
* + docker system prune -f
* [Pipeline] cleanWs
* [WS-CLEANUP] Deleting project workspace...
* [WS-CLEANUP] Deferred wipeout is used...
* [WS-CLEANUP] done
* [Pipeline] }
* [Pipeline] // script
* [Pipeline] }
* [Pipeline] // stage
* [Pipeline] }
* [Pipeline] // withEnv
* [Pipeline] }
* [Pipeline] // node
* [Pipeline] End of Pipeline
* GitHub has been notified of this commit’s build result
* Finished: SUCCESS
* The Image is pushed to my ECR repository

