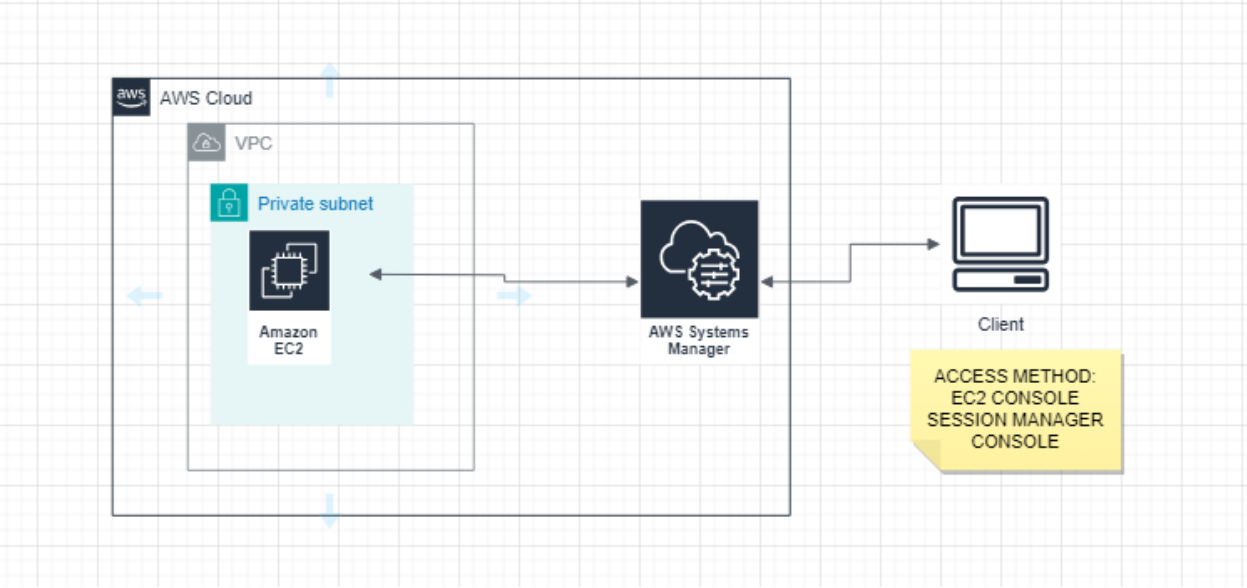
PROJECT 1

Remotely Run Commands on an EC2 Instance with AWS Systems Manager

Architecture diagram



In this project we will learn how to use AWS Systems Manager to remotely run commands on our Amazon EC2 instances. Systems Manager is a management tool that enables us to gain operational insights and take action on AWS resources safely and at scale. Using the run command, one of the automation features of Systems Manager, we can simplify management tasks by eliminating the need to use bastion hosts, SSH, or remote PowerShell.

as a System Administrator, we need to update the packages on our EC2 instances. To complicate this normally simple admin task, our security team does not allow you to direct access production servers via SSH or allow you to use bastion hosts. Fortunately, you can use Systems Manager to remotely run commands, like update packages, on your EC2 instances.

To solve this challenging scenario, you will create an Identity and Access Management (IAM) role, enable an agent on your instance that communicates with Systems Manager, then follow best practices by running the AWS-UpdateSSMAgent document to upgrade your Systems Manager Agent, and finally use Systems Manager to run a command on your instance.

AWS Systems Manager

AWS Systems Manager is a management service that helps you automatically collect software inventory, apply patches, create system images, and configure Windows and Linux operating systems. It provides a unified user interface so you can view operational data from multiple AWS services and automate operational tasks across your AWS resources. Systems Manager simplifies resource and application management, shortens the time to detect and resolve operational problems, and makes it easy to operate and manage your infrastructure securely at scale.

IAM roles

IAM roles in AWS are used to delegate access to AWS resources securely. Here are some common use cases for IAM roles.

EC2 intance

When you launch an EC2 instance, you can specify an IAM role to be associated with the instance. The IAM role determines the permissions that the EC2 instance has. You can change the IAM role associated with an instance while the instance is running, but you can't remove the IAM role from an instance once it has been assigned.

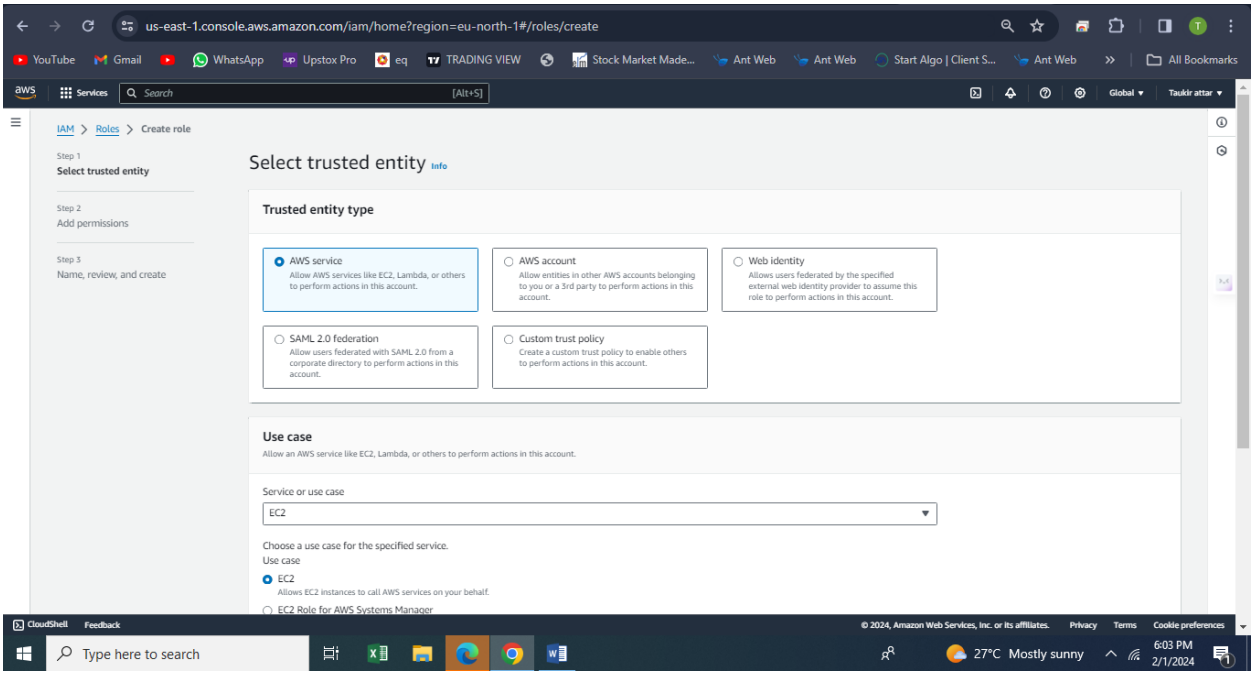
Let’s Start Creating Project

- Create an Identity and Access Management (IAM) role
- In this step, we will create an IAM role that will be used to give Systems Manager permission to perform actions on our instances.

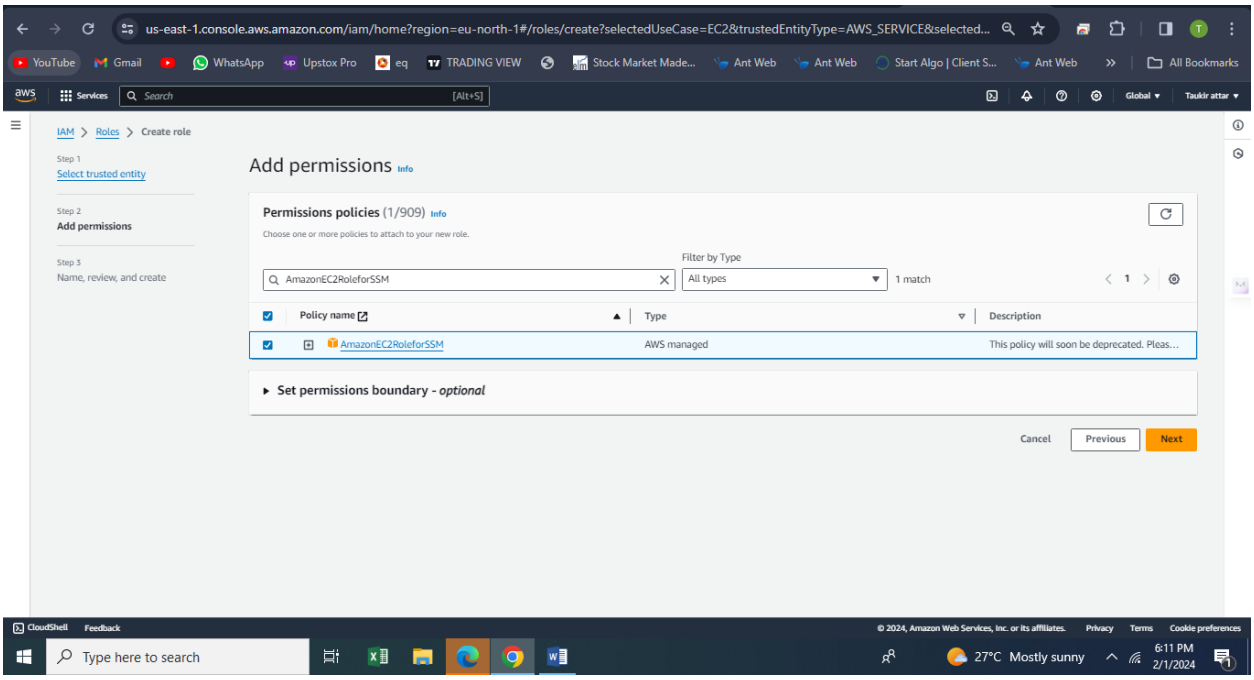
Open the IAM console

In the left navigation pane, choose Roles, and then choose Create role

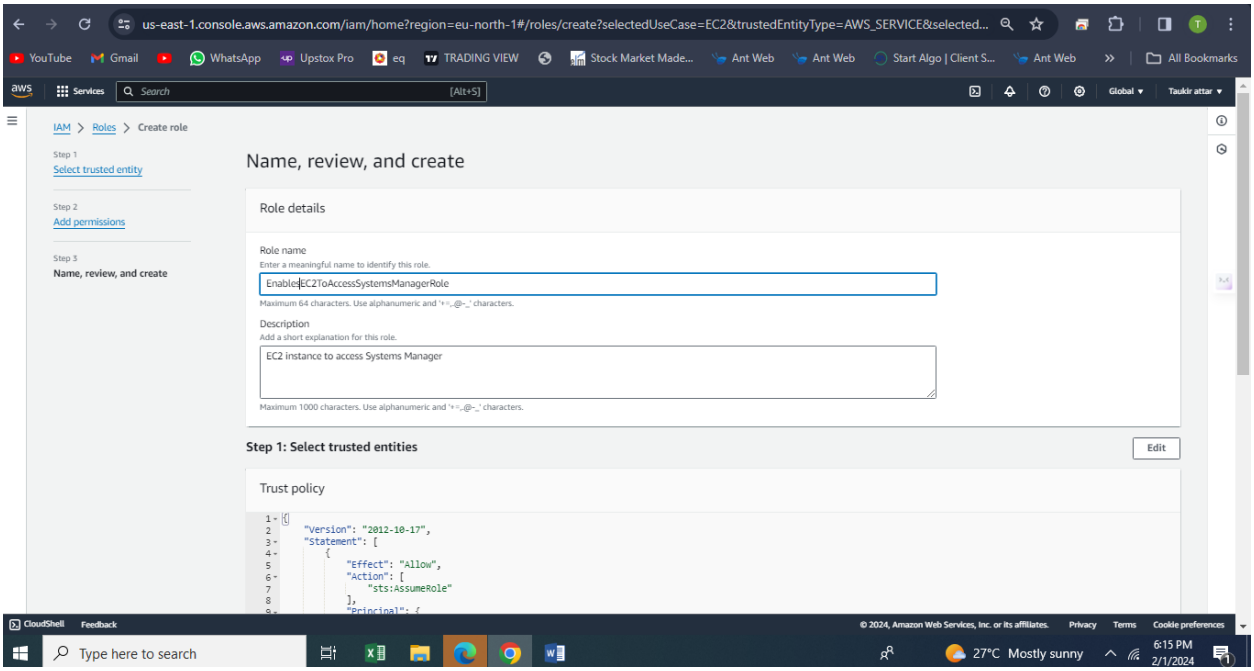
On the Select trusted entity page, under AWS Service, choose EC2, and then choose Next.



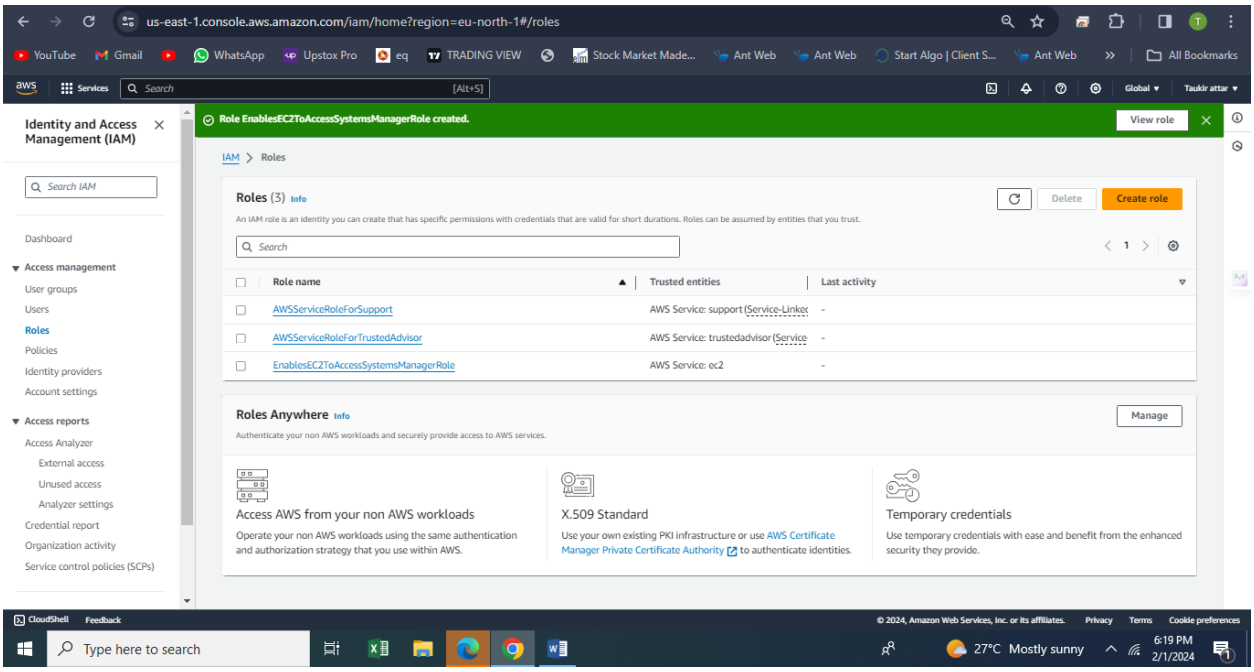
- On the Add permissions page, in the search bar type AmazonEC2RoleforSSM. From the policy list select AmazonEC2RoleforSSM and then choose Next.



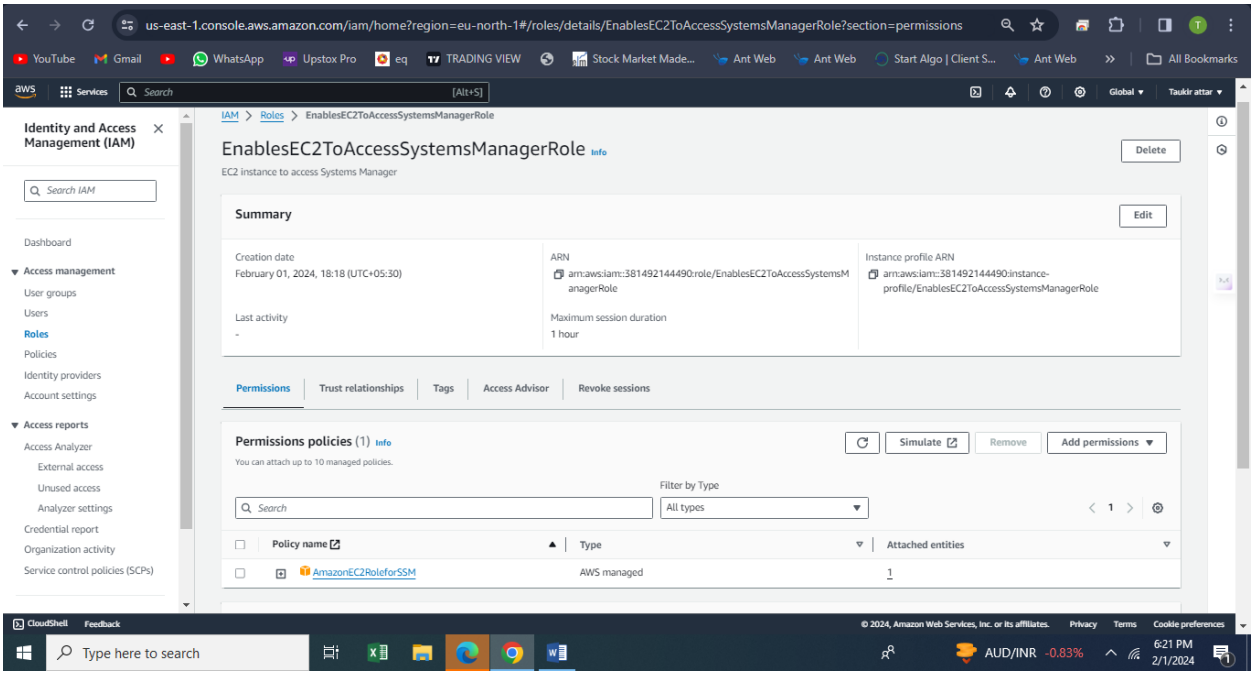
- On the Name, review, and create page, in the Role name box, type in EnablesEC2ToAccessSystemsManagerRole. In the Description box, type in Enables an EC2 instance to access Systems Manager. Choose Create role.



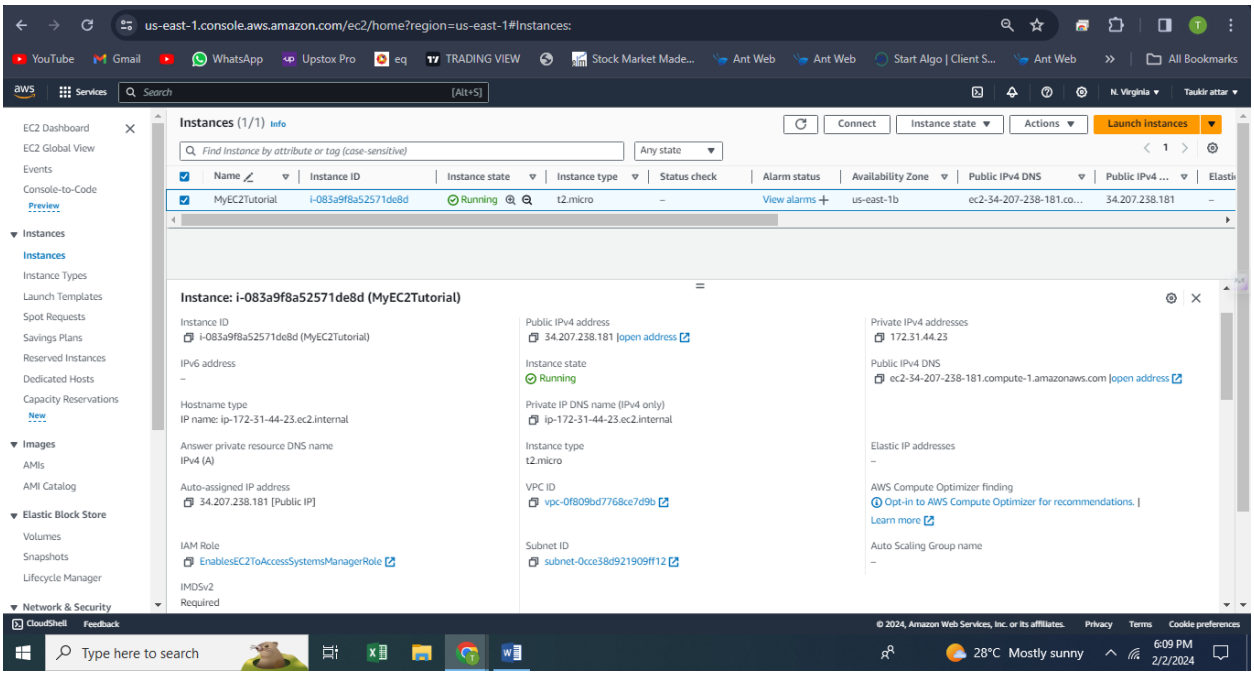
- Role is created in the below screenshot



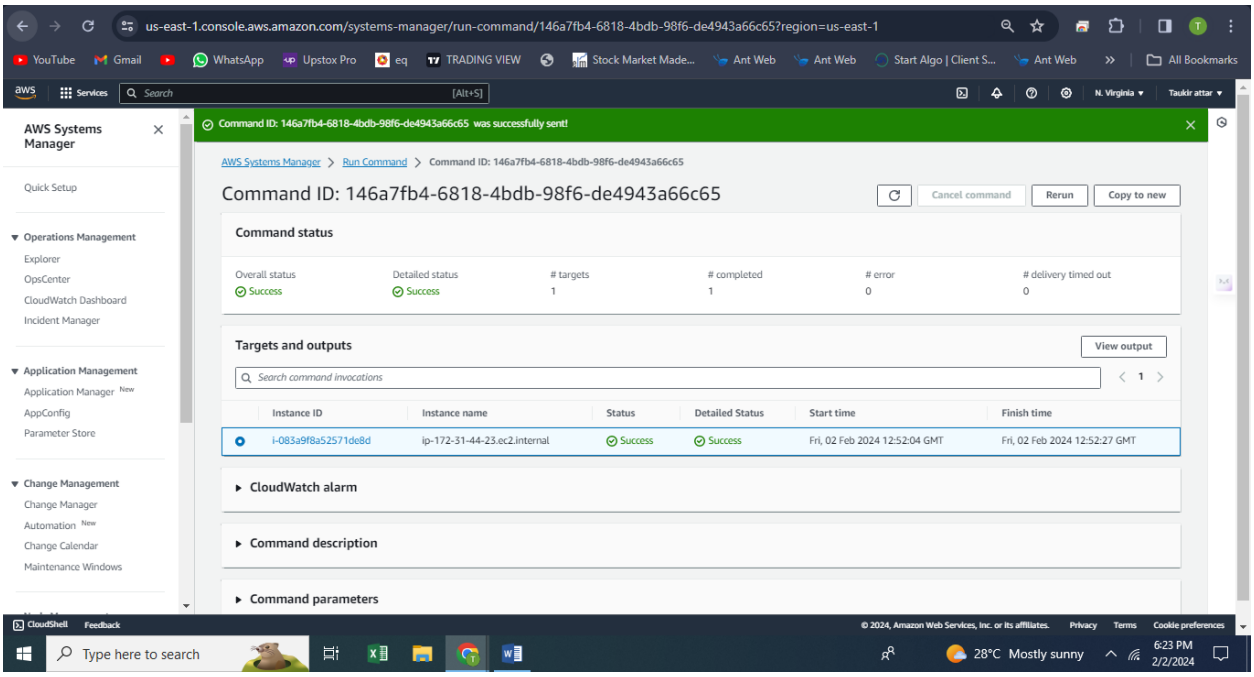
In details I had explained role in below screenshot



- Now we are creating ec2 instance
- From the EC2 console, select our preferred Region.
- In the Name field, I have entered MyEC2Tutorial. Selected the Amazon Linux AMI. Retain the default selection that appears in the dropdown. We can also install the Systems Manager Agent on our own Windows or Linux system.
- Choose the t2.micro instance type.
- We will not need a keypair to use Systems Manager to remotely run commands. Scroll down to Key pair and under the Key pair name dropdown, choosed Proceed without a key pair.
- Under Advanced details, in the IAM instance profile dropdown choose the EnablesEC2ToAccessSystemsManagerRole role you created earlier. Leave everything else as default. Choose Launch instance.
- Ec2 is created in the below screen shot

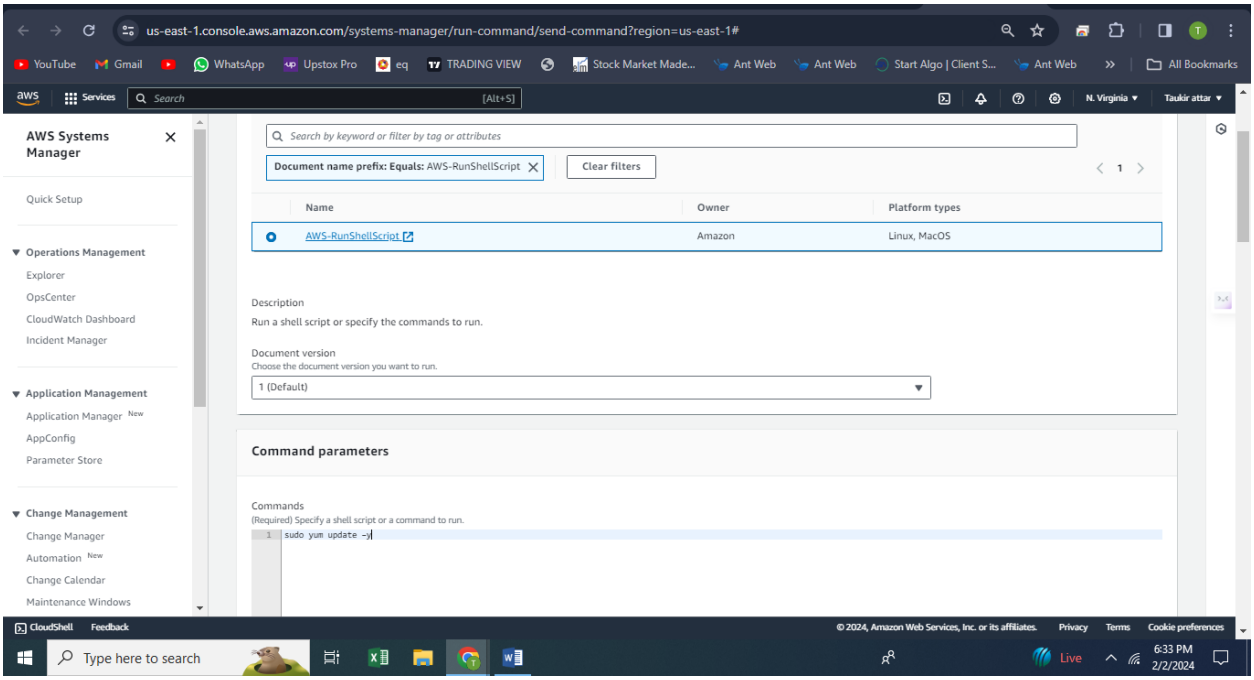


- Update the Systems Manager Agent
- Now that you have an EC2 instance running the Systems Manager agent, you can automate administration tasks and manage the instance. In this step, you run a pre-packaged command, called a document, that will upgrade the agent. It is best practice to update the Systems Manager Agent when you create a new instance.
- In the top navigation bar, search for Systems Manager and open the Systems Manager console.
- Under the Node Management section on the left navigation bar, choose Fleet Manager.
- Select the node ID created in step 2, MyEC2Tutorial, to open the node detail page.
- On the node detail page, in the Node actions dropdown, select Execute run command.
- On the Run a command page, click in the search bar and select, Document name prefix, then click on Equals, then type in AWS-UpdateSSMAgent.
- Scroll down to the Targets panel and select the check box next to your managed EC2 instance.
- Next you will see a page documenting your running command, and then overall success in green. Congrats, you have just run your first remote command using Systems Manager.



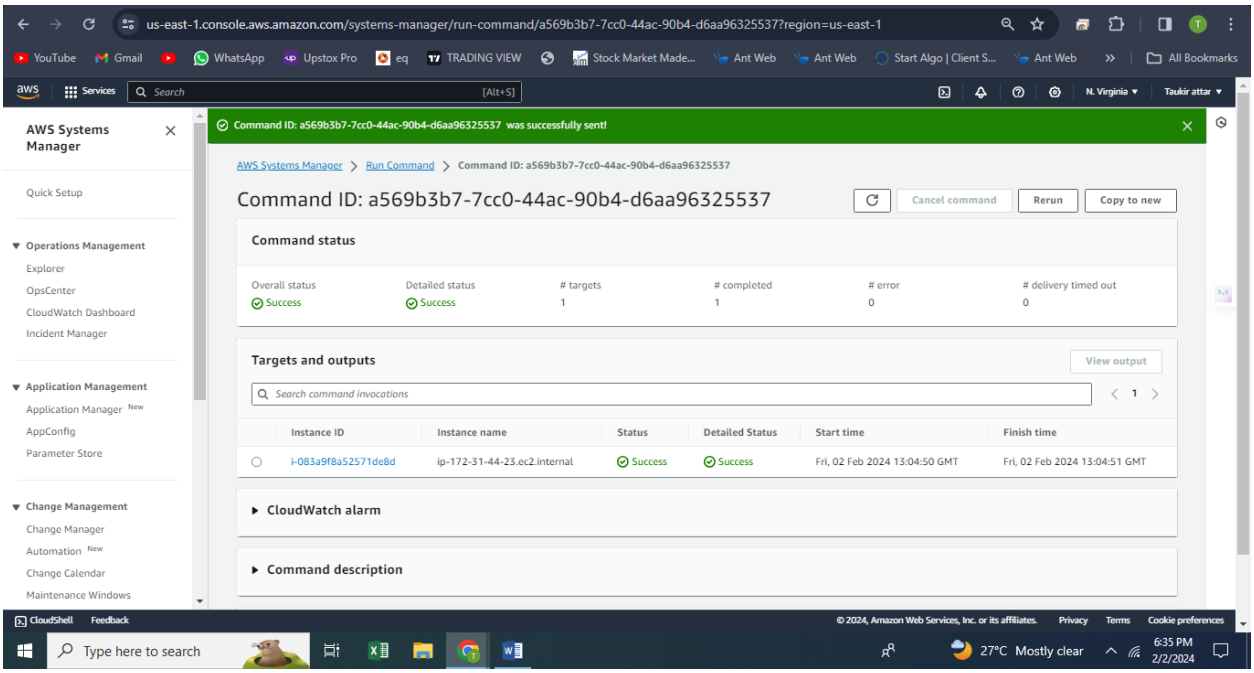
Run remote shell script

- Now that your EC2 instance has the latest Systems Manager Agent, you can upgrade the packages on the EC2 instance. In this step, you will run a shell script through Run Command.
- Under the Node Management section on the left navigation bar, choose Fleet Manager.
- Select the node ID created in step 2, MyEC2Tutorial, to open the node detail page.
- On the node detail page, in the Node actions dropdown, select Execute run command.
- On the Run a command page, click in the search bar and select, Document name prefix, then click on Equals, then type in *AWS-RunShellScript.*
- Now select the radio button on the left of AWS-RunShellScript.

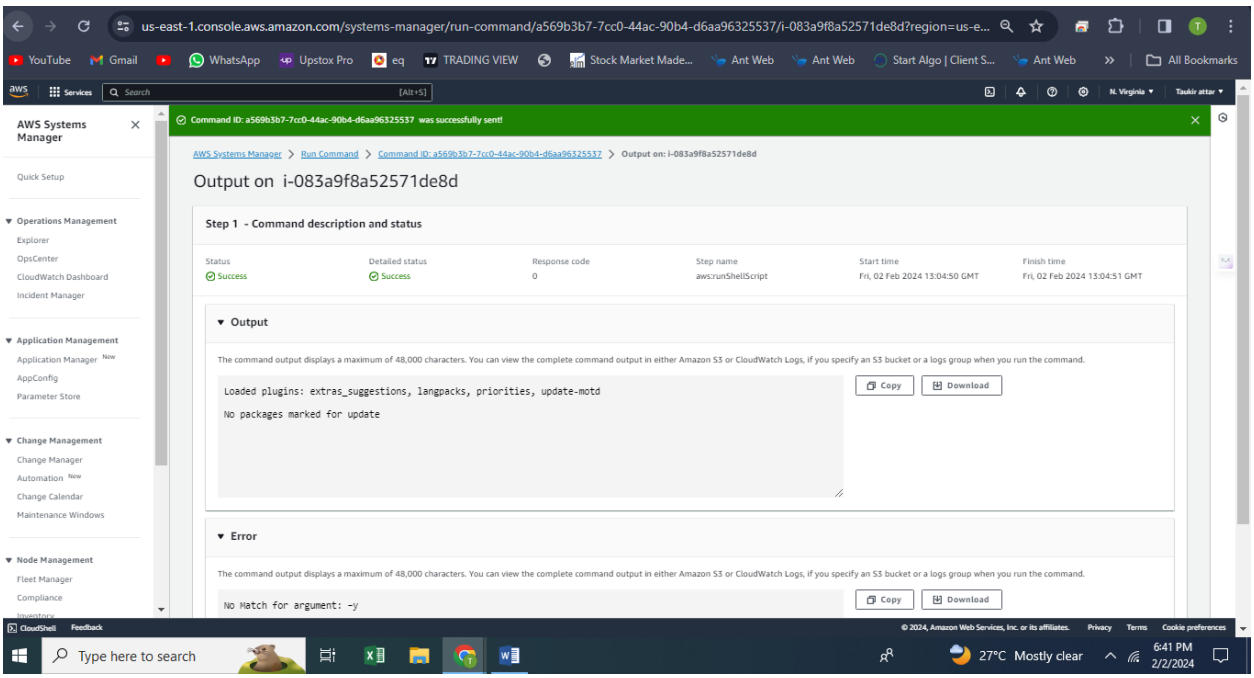


- Scroll down to the Command Parameters panel and insert the following command in the Commands text box:
- sudo yum update –y

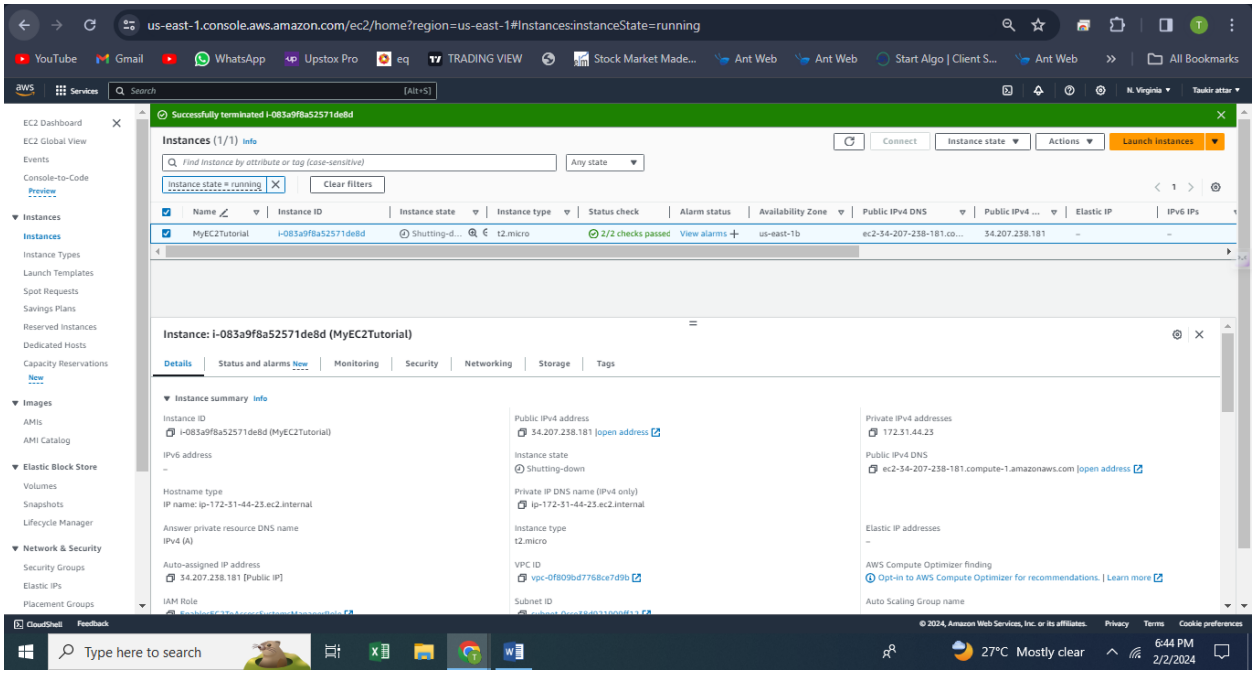
- Scroll down to the Targets panel and select the check box next to your managed EC2 instance.
- Finally, scroll down and select Run.
- While your script is running remotely on the managed EC2 instance, the Overall status will be In Progress. Soon the Overall status will turn to Success. When it does, scroll down to the Targets and outputs panel and select the Instance ID of your instance. Your Instance ID will be different than the one pictured.



From the Output on: i-083a9f8a52571d8d, select the header of the Output panel to view the output of the update command from the instance.



- In this step, you will terminate your Systems Manager and EC2 related resources. Important: Terminating resources that are not actively being used reduces costs and is a best practice. Not terminating your resources can result in a charge.
-



Cost Analysis

AWS Systems Manager: The pricing for

Systems Manager Run Command is $0.05 . Assuming each user triggers one command per day (30 commands per month per user):

Cost per user = $0.05 * 30 = $1.5

Total cost for 1000 users = $1.5 * 1000 = $1500

Amazon EC2: The pricing for a t2.micro instance in the US East (N. Virginia) region is $0.0116 per hour. Assuming the instance runs continuously for a month:

Cost per instance = $0.0052 * 24 * 30 = $3.744

Total cost for 1000 users = $3.744 * 1000 = $3744

Total Monthly Billing Estimate:

Systems Manager: $1500

EC2 instances: $3744

Total Estimate: $5244

lessons and observations

AWS Systems Manager to remotely run commands on Amazon EC2 instances, several key lessons and observations can be drawn:

- Cost Considerations:

Pay-as-You-Go Pricing: AWS Systems Manager follows a pay-as-you-go pricing model, where you pay only for the resources you use. This can be cost-effective, especially for small to medium-sized businesses or projects.

Optimization: To optimize costs, it's important to monitor and manage the usage of Systems Manager and EC2 instances. This includes reviewing usage patterns, selecting the right instance types, and leveraging cost-saving options like reserved instances or spot instances where applicable.

- Security and Compliance:

Secure Communication: Systems Manager uses secure communication channels to execute commands on EC2 instances, ensuring that sensitive information is protected.

Compliance Standards: Systems Manager helps in maintaining compliance with standards such as PCI DSS, HIPAA, and GDPR by providing audit logs, compliance reports, and encryption features.

- Operational Efficiency:

Automation: Systems Manager offers automation features that allow you to automate common administrative tasks, reducing manual intervention and improving efficiency.

Centralized Management: With Systems Manager, you can centrally manage EC2 instances across multiple AWS accounts and regions, streamlining operations and reducing complexity.

- Agent Management:

Agent Installation: Installing the SSM Agent on EC2 instances is essential for Systems Manager to communicate with the instances. You can automate this process using AWS Systems Manager State Manager or AWS CloudFormation.

Agent Updates: Regularly updating the SSM Agent ensures that you have access to the latest features, improvements, and security patches.

- Scalability:

Scaling Resources: Systems Manager is designed to scale with your infrastructure, allowing you to manage a large number of instances efficiently.

Resource Groups: Using resource groups in Systems Manager, you can organize and manage instances based on tags, simplifying management in large-scale environments.

By considering these aspects, you can effectively leverage AWS Systems Manager to remotely run commands on Amazon EC2 instances, ensuring cost-effectiveness, security, compliance, operational efficiency, and scalability in your AWS environment.

Project 2

Build a Jenkins build server for the development team

Jenkins on AWS

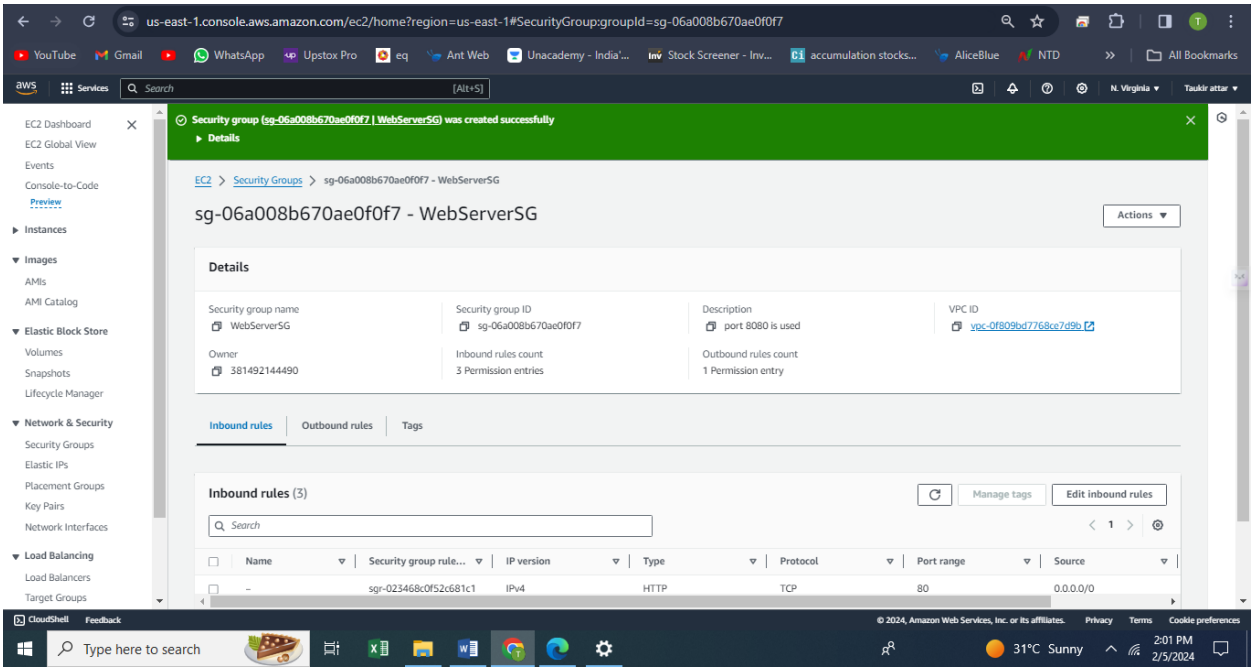
Jenkins is an open-source automation server that integrates with a number of AWS Services, including: AWS CodeCommit, AWS CodeDeploy, Amazon EC2 Spot, and Amazon EC2 Fleet. You can use Amazon Elastic Compute Cloud (Amazon EC2) to deploy a Jenkins application on AWS.

This tutorial walks you through the process of deploying a Jenkins application. You will launch an EC2 instance, install Jenkins on that instance, and configure Jenkins to automatically spin up Jenkins agents if build abilities need to be augmented on the instance.

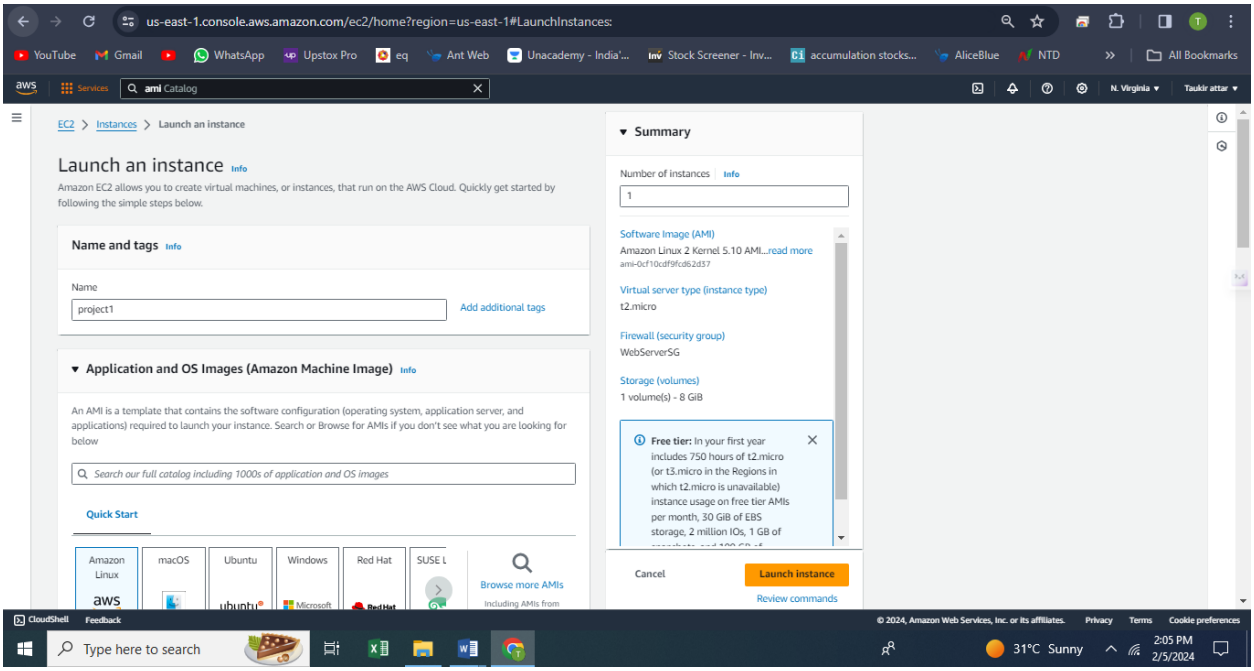
- Creating a security group

A security group acts as a firewall that controls the traffic allowed to reach one or more EC2 instances. When you launch an instance, you can assign it one or more security groups. You add rules that control the traffic allowed to reach the instances in each security group. You can modify a security group's rules any time, and the new rules take effect immediately.

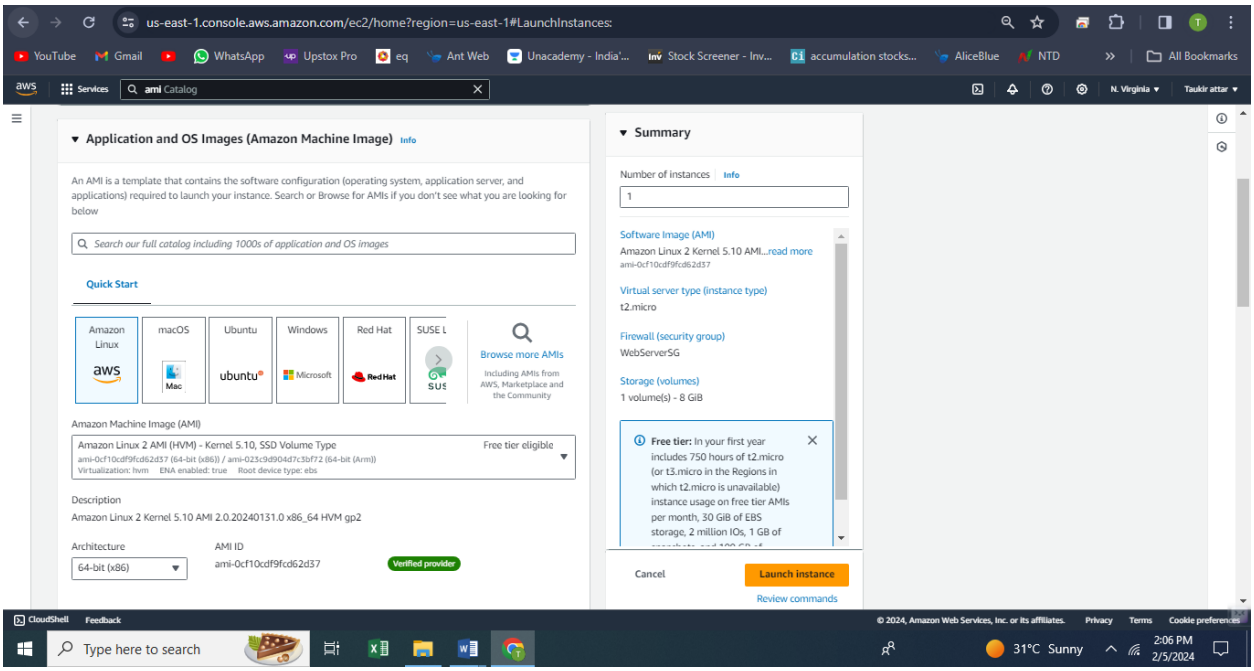
- We will create a security group and add the following rules:
- Allow inbound HTTP access from anywhere.
- Allow inbound SSH traffic from your computer's public IP address so you can connect to your instance.
- In Security group name, enter WebServerSG or any preferred name of your choice, and provide a description.
- Select your VPC from the list. You can use the default VPC.
- On the Inbound tab, add the rules as follows:
- Select Add Rule, and then select SSH from the Type list.
- Under Source, select Custom, and in the text box, enter the IP address from step 1, followed by /32 indicating a single IP Address. For example, 104.34.241.123/32 is a single IP address, while 198.51.100.2/24 results in a range of 256 IP addresses.
- Select Add Rule, and then select HTTP from the Type list.
- Select Add Rule, and then select Custom TCP Rule from the Type list.
- Under Port Range, enter 8080.
- Select Create



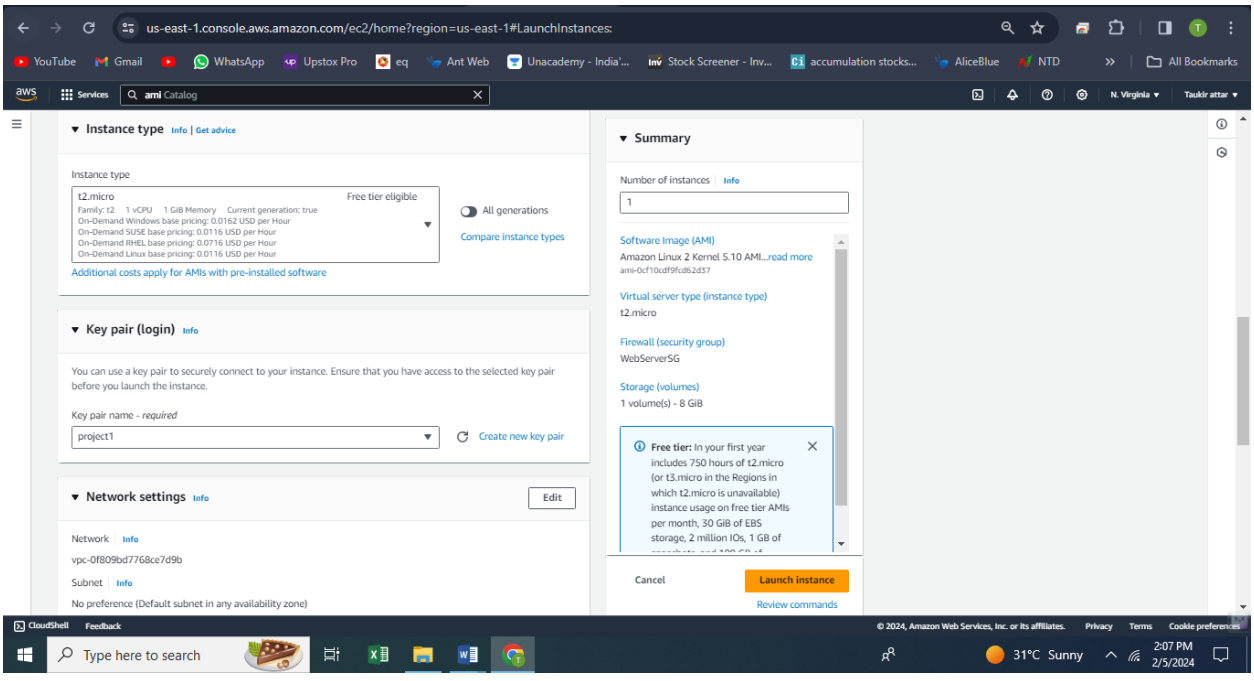
Launching an Amazon EC2 instance



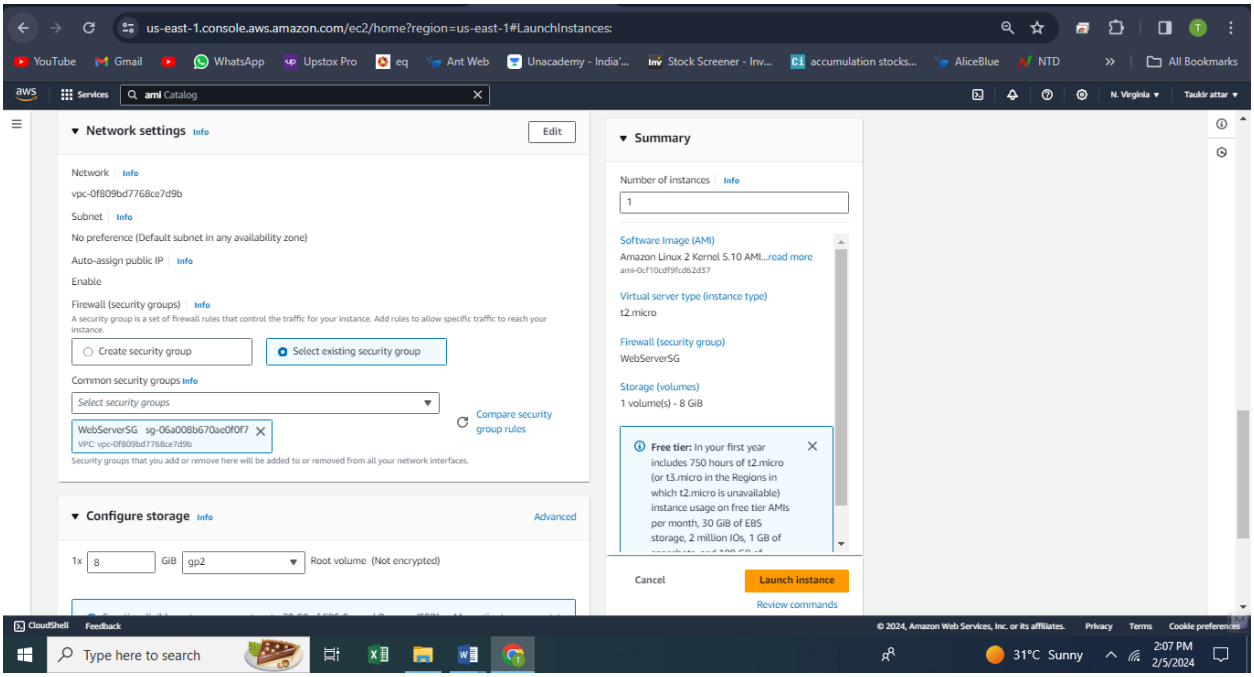
- The Choose an Amazon Machine Image (AMI) page displays a list of basic configurations called Amazon Machine Images (AMIs) that serve as templates for your instance. Select the HVM edition of the Amazon Linux AMI.



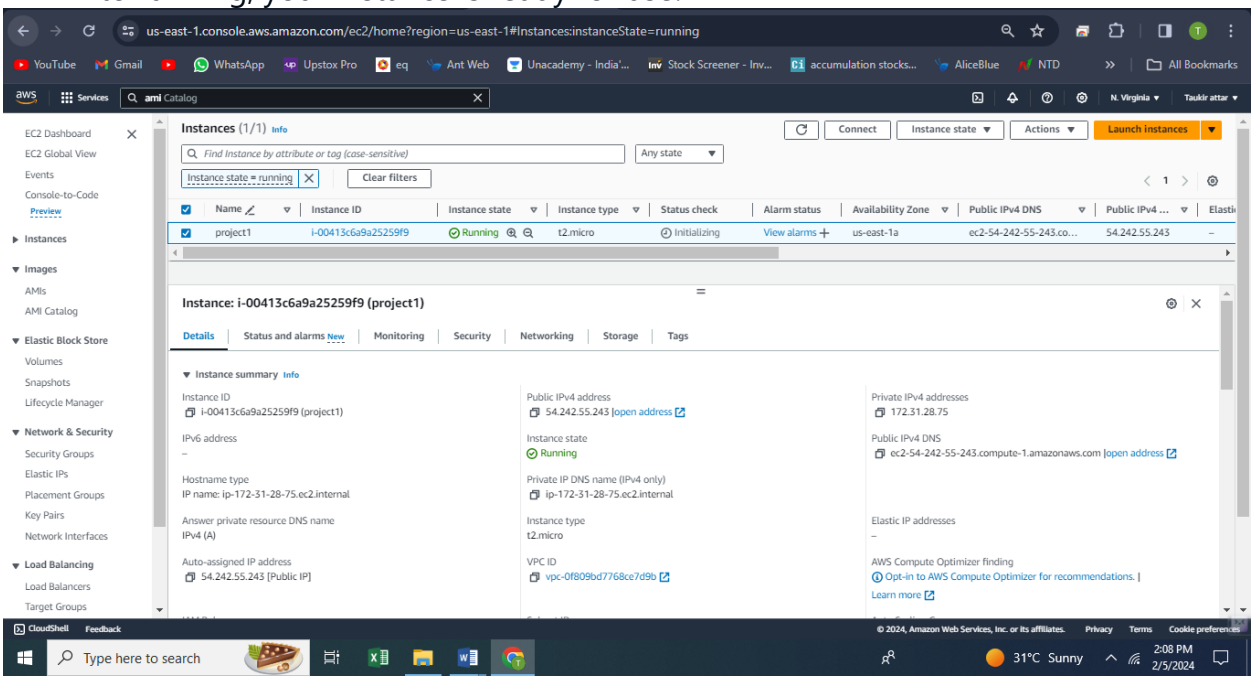
- Scroll down and select the key pair you created in the creating a key pair section above or any existing key pair you intend to use.



a. Select Select an existing security group.
b. Select the WebServerSG security group that you created.
c. Select Launch Instance



- In the left-hand navigation bar, choose Instances to view the status of your instance. Initially, the status of your instance is pending. After the status changes to running, your instance is ready for use.

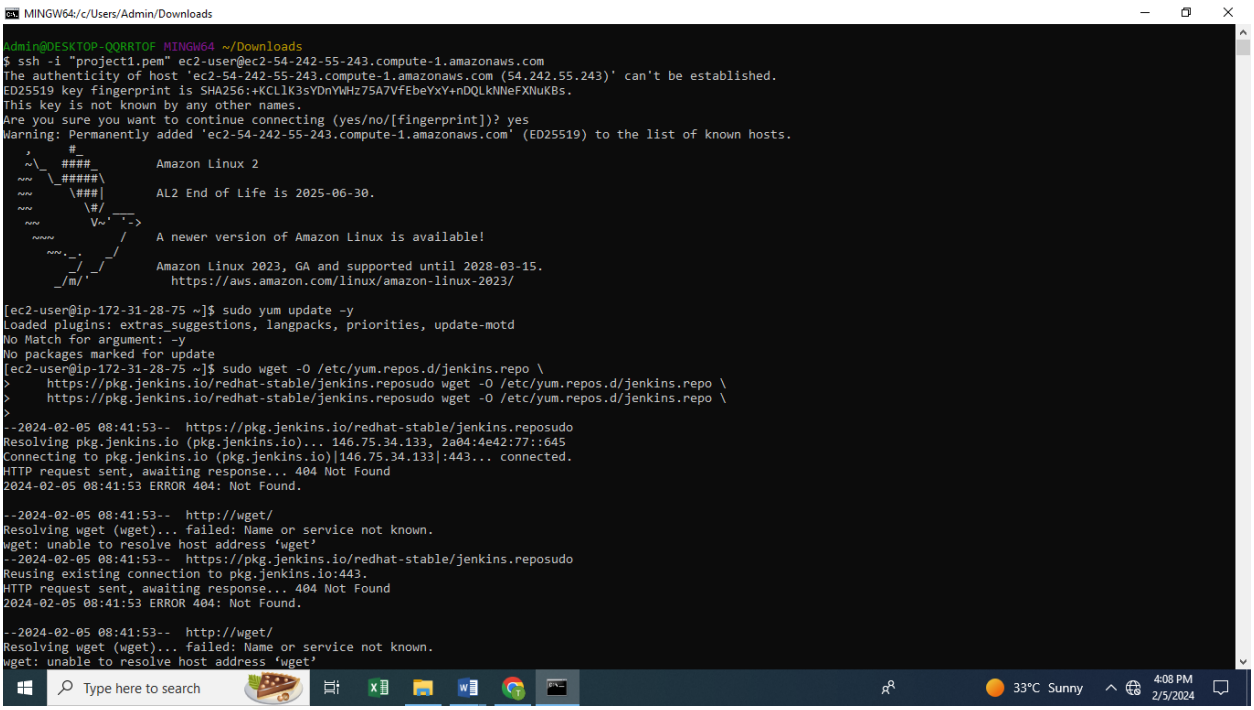


Installing and configuring Jenkins

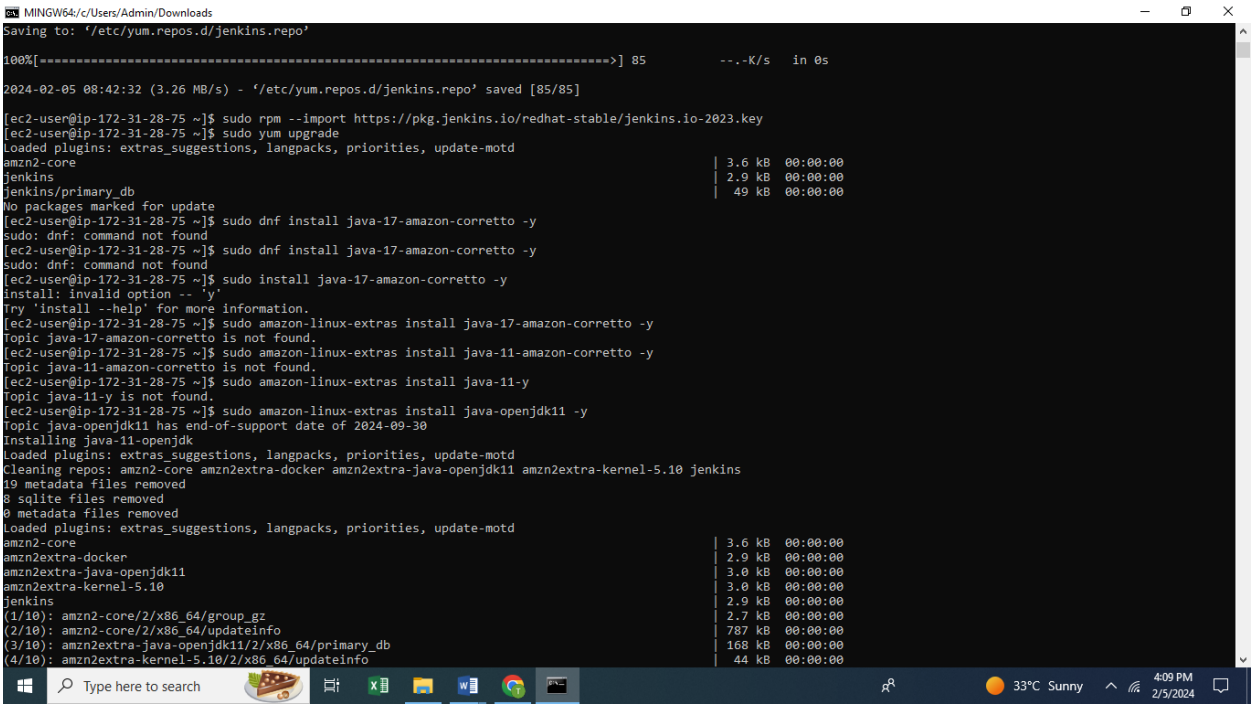
- Now that the Amazon EC2 instance has been launched, Jenkins can be installed properly.

Using SSH to connect to your instance
Use the ssh command to connect to the instance. You will specify the private key (.pem) file and ec2-user@public_dns_name.

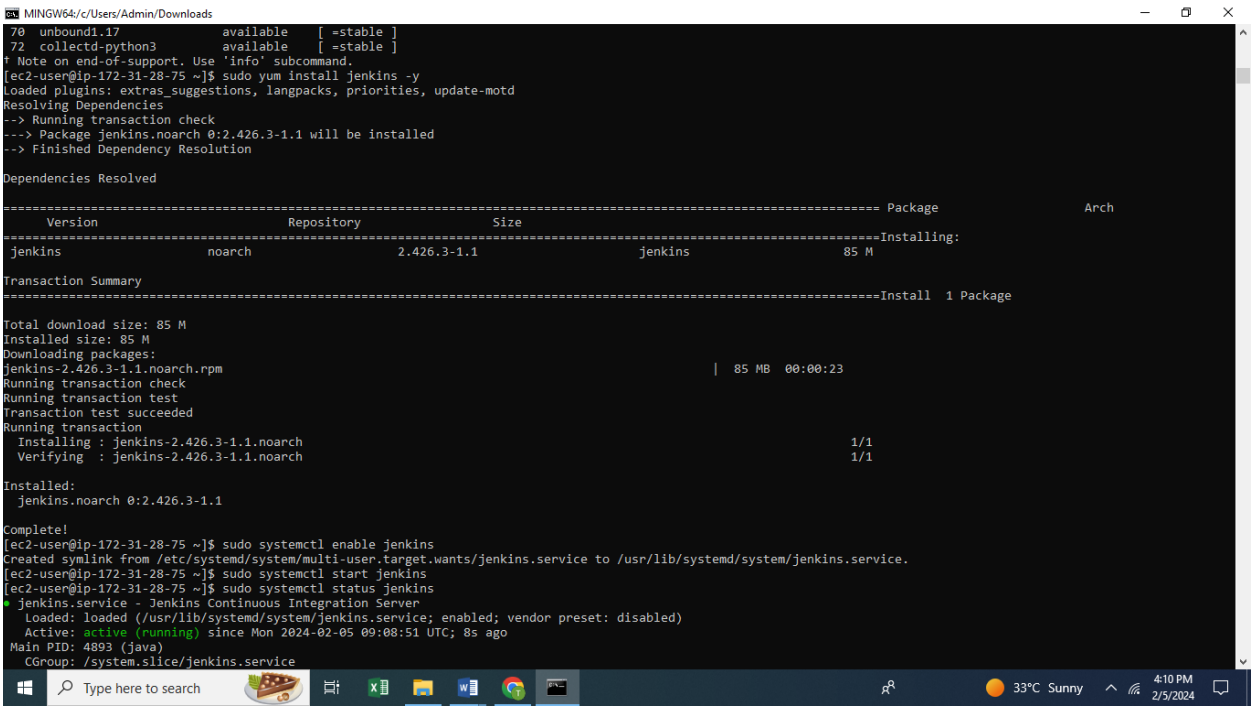
- [ec2-user ~]$ sudo yum update –y



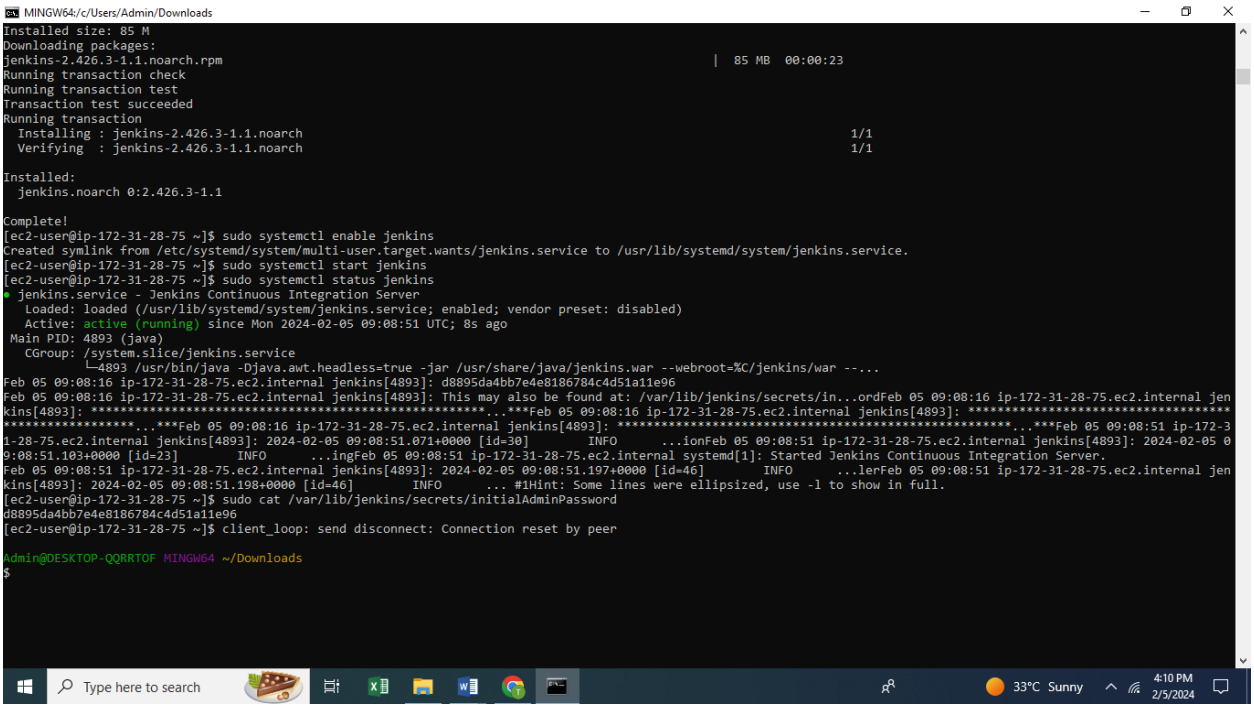
- [ec2-user ~]$ sudo wget -O /etc/yum.repos.d/jenkins.repo \
  https://pkg.jenkins.io/redhat-stable/jenkins.repo
- [ec2-user ~]$ sudo yum upgrade
- [ec2-user ~]$ sudo dnf install java-17-amazon-corretto -y
-



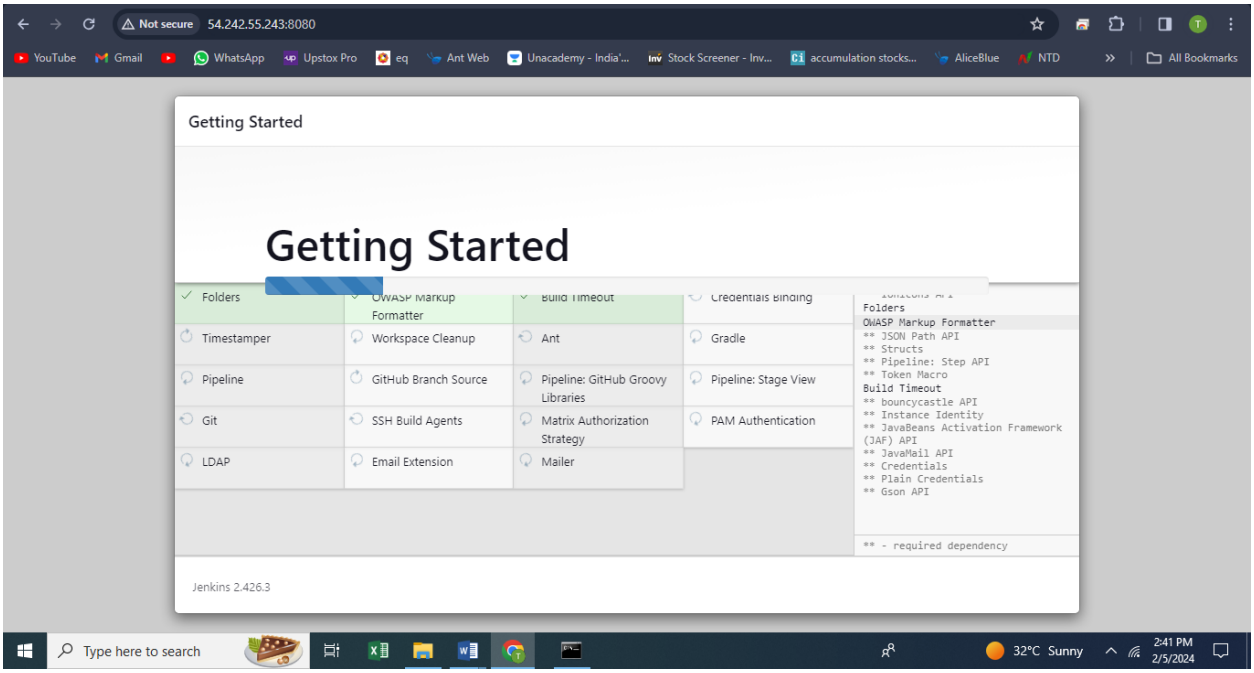
- [ec2-user ~]$ sudo yum install jenkins –y
- [ec2-user ~]$ sudo systemctl enable jenkins
- [ec2-user ~]$ sudo systemctl start jenkins
- [ec2-user ~]$ sudo systemctl status jenkins



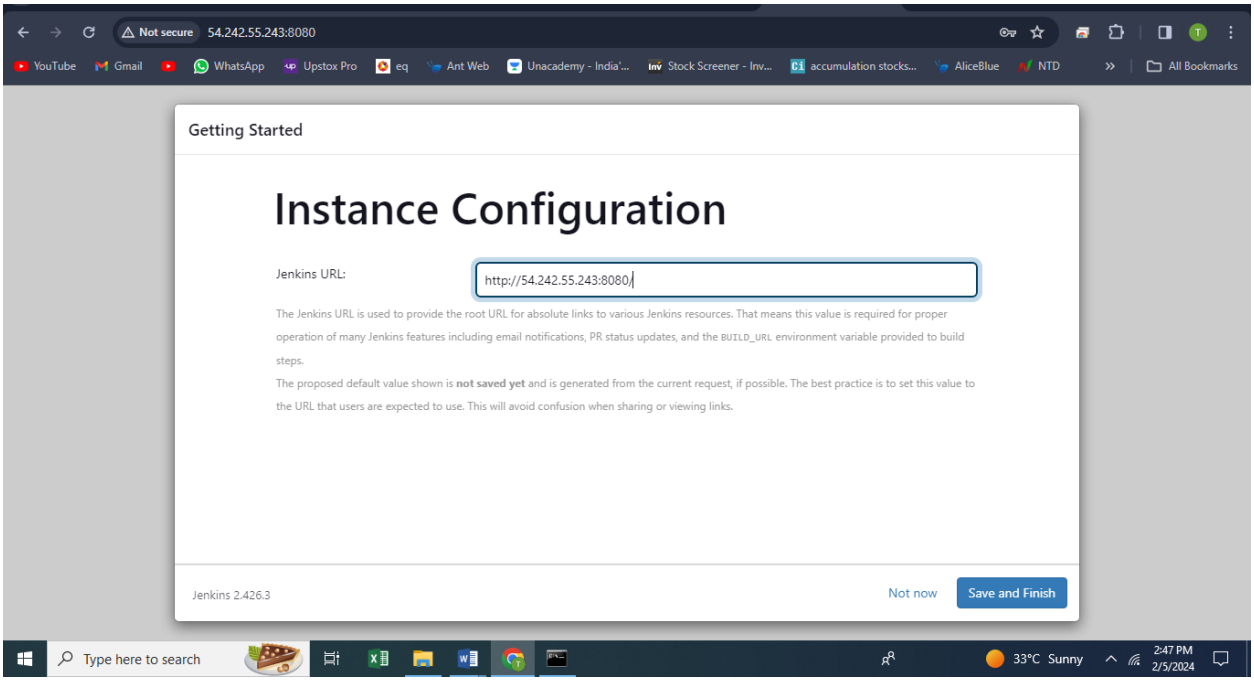
[ec2-user ~]$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword

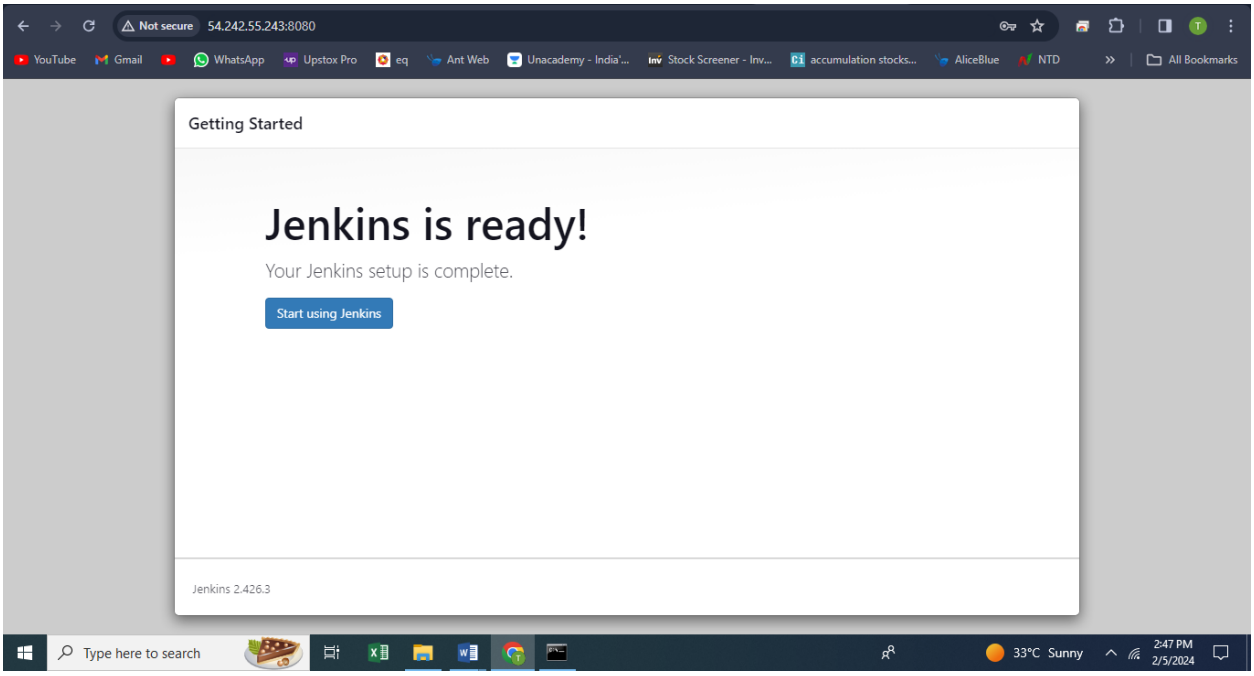


- Connect to http://<your_server_public_DNS>:8080 from your browser. You will be able to access Jenkins through its management interface:

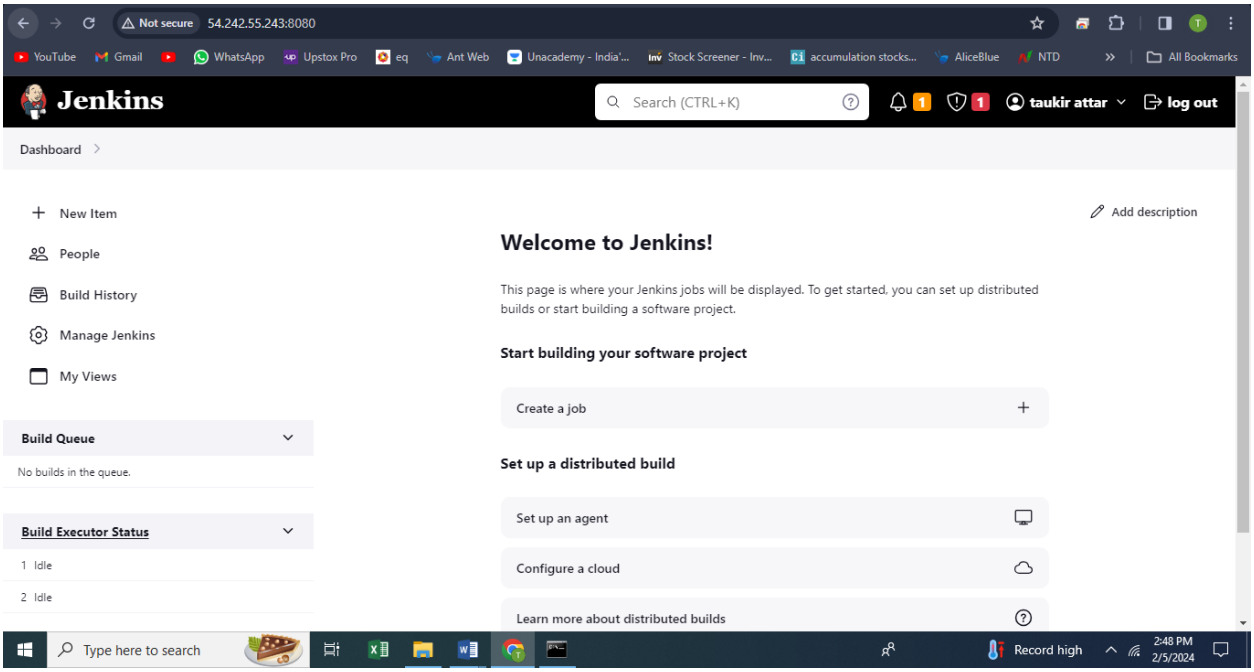


Once the installation is complete, the **Create First Admin User** will open. Enter your information, and then select **Save and Continue**.

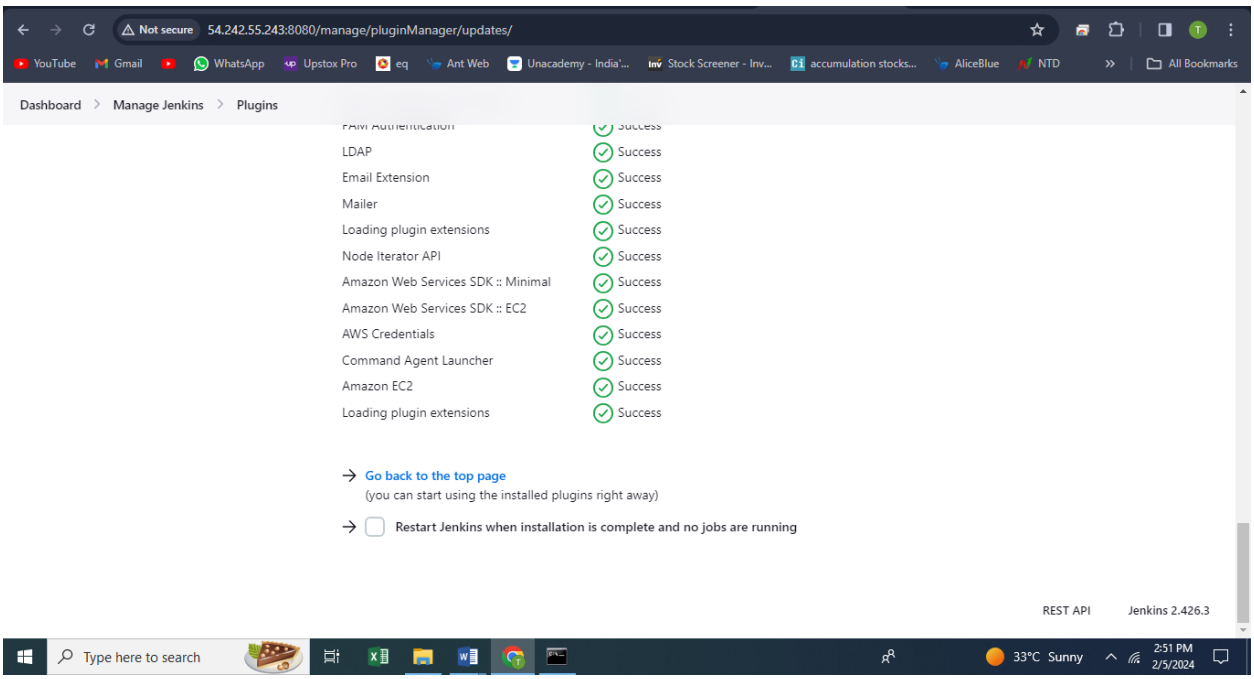




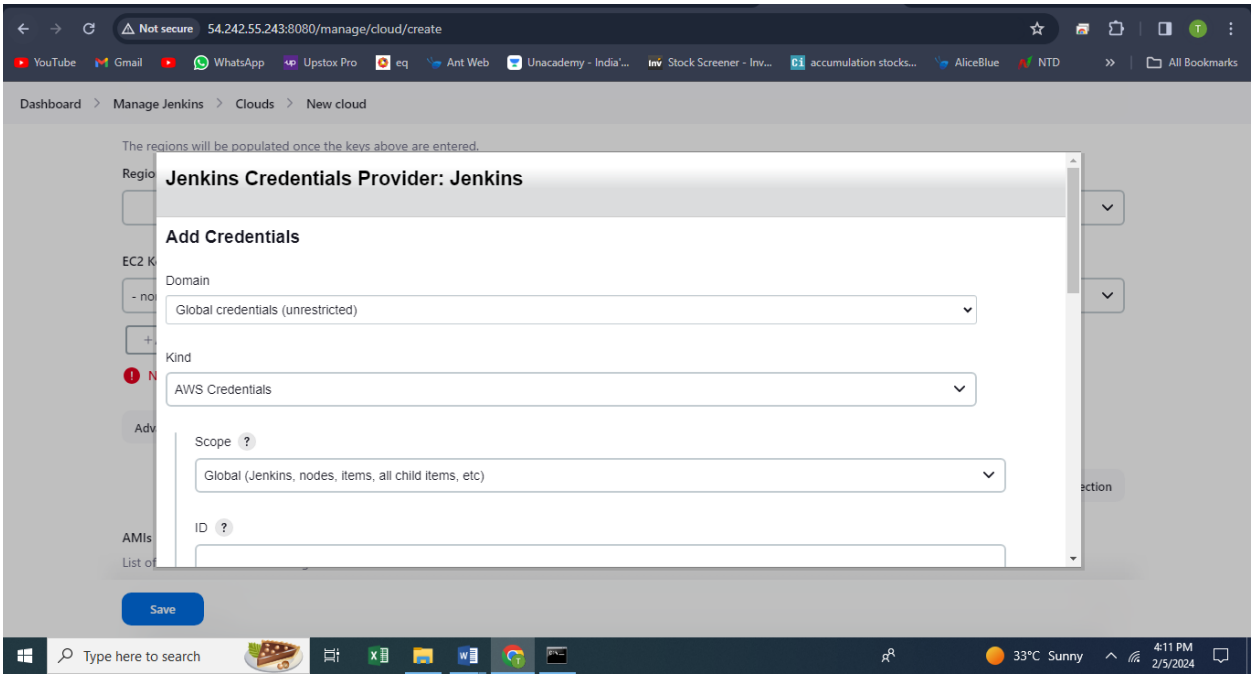
- If you already have other nodes or clouds set up, select **Manage Jenkins**.
- After navigating to **Manage Jenkins**, select **Configure Nodes and Clouds** from the left hand side of the page.
- Select **Add a new cloud**, and select **Amazon EC2**. A collection of new fields appears.

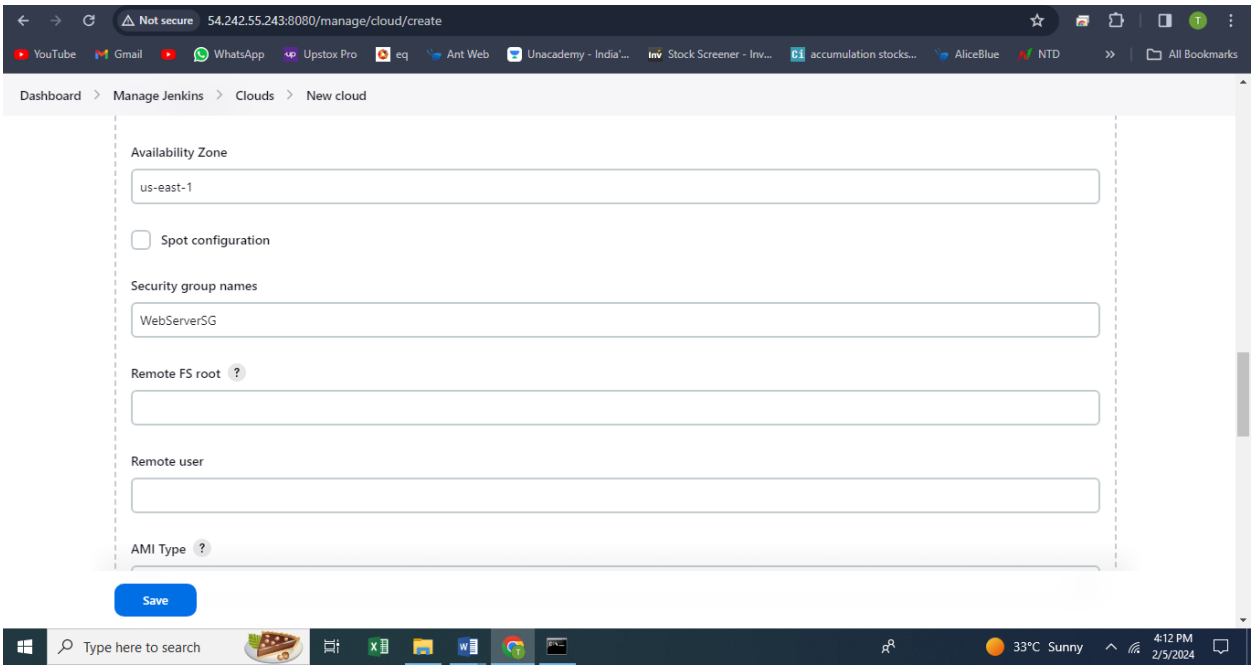


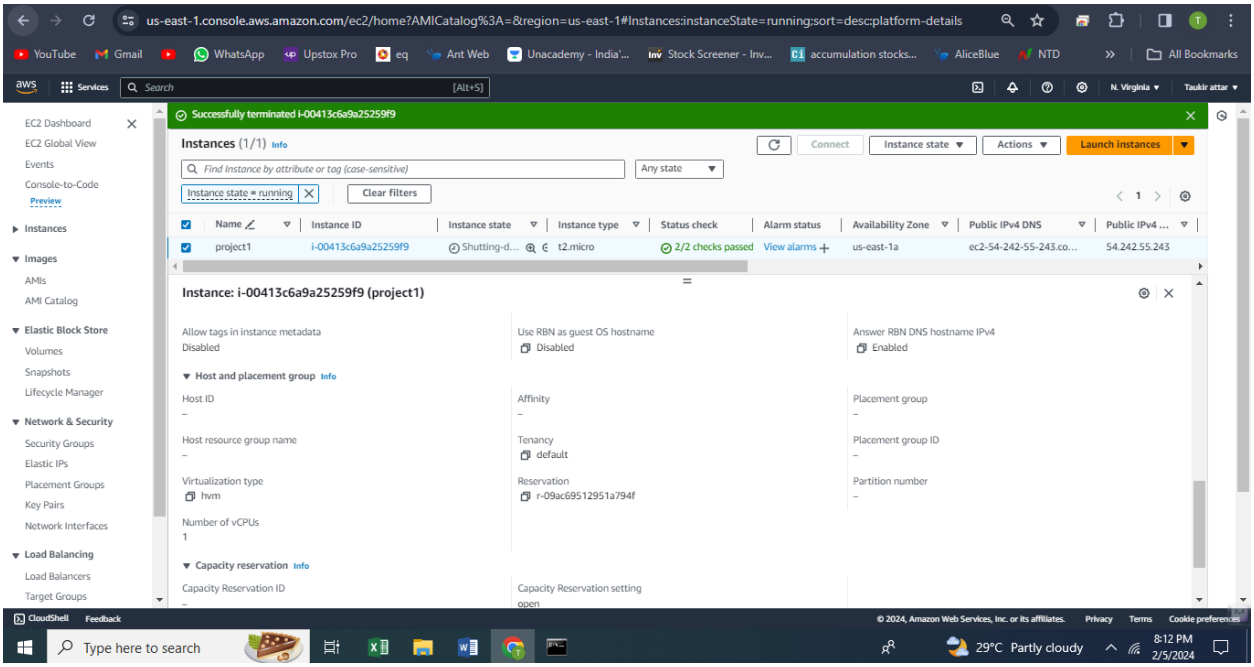
Amazon ec2 install



- Click **Add** under Amazon EC2 Credentials


Not secure 54.242.55.243:8080/manage/cloud/create
Dashboard > Manage Jenkins > Clouds > New cloud
Jenkins Credentials Provider: Jenkins
Add Credentials
Domain
Global credentials (unrestricted)
Kind
AWS Credentials
Scope
Global (Jenkins, nodes, items, all child items, etc)
ID
Save
Type here to search
33°C Sunny
4:11 PM
2/5/2024


Not secure 54.242.55.243:8080/manage/cloud/create
Dashboard > Manage Jenkins > Clouds > New cloud
Availability Zone
us-east-1
Spot configuration
Security group names
WebServerSG
Remote FS root
Remote user
AMI Type
Save
Type here to search
33°C Sunny
4:12 PM
2/5/2024



Cleaning up

After completing this tutorial, be sure to delete the AWS resources that you created so you do not continue to accrue charges.

Deleting your EC2 instance

1. In the left-hand navigation bar of the Amazon EC2 console, select **Instances**.
2. Right-click on the instance you created earlier, and select **Terminate**.