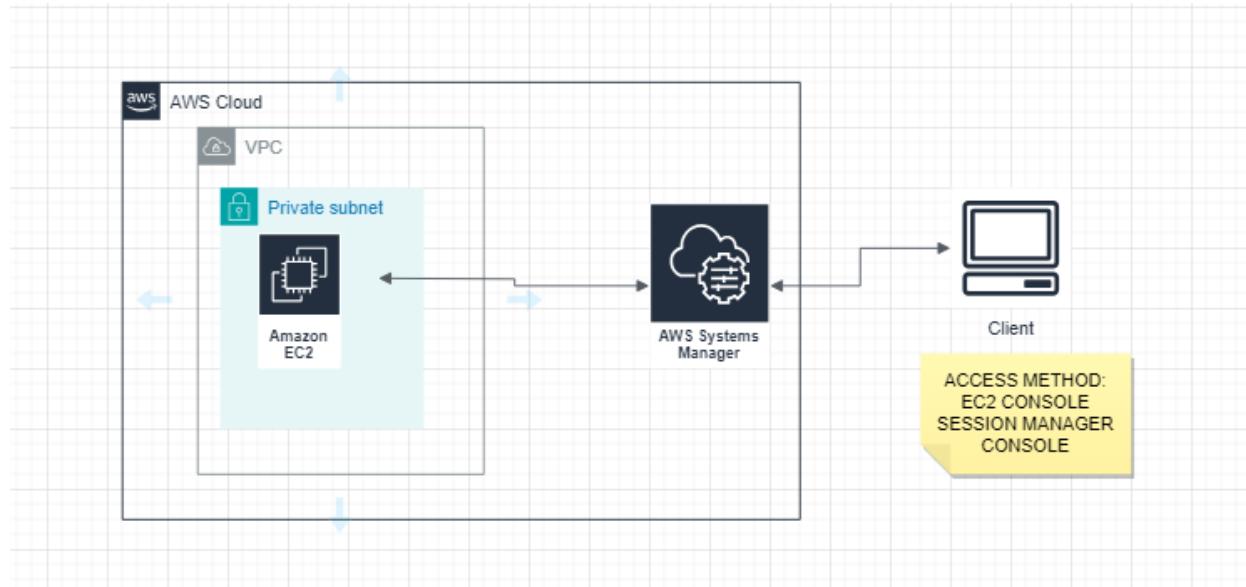


PROJECT 1

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 instance

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

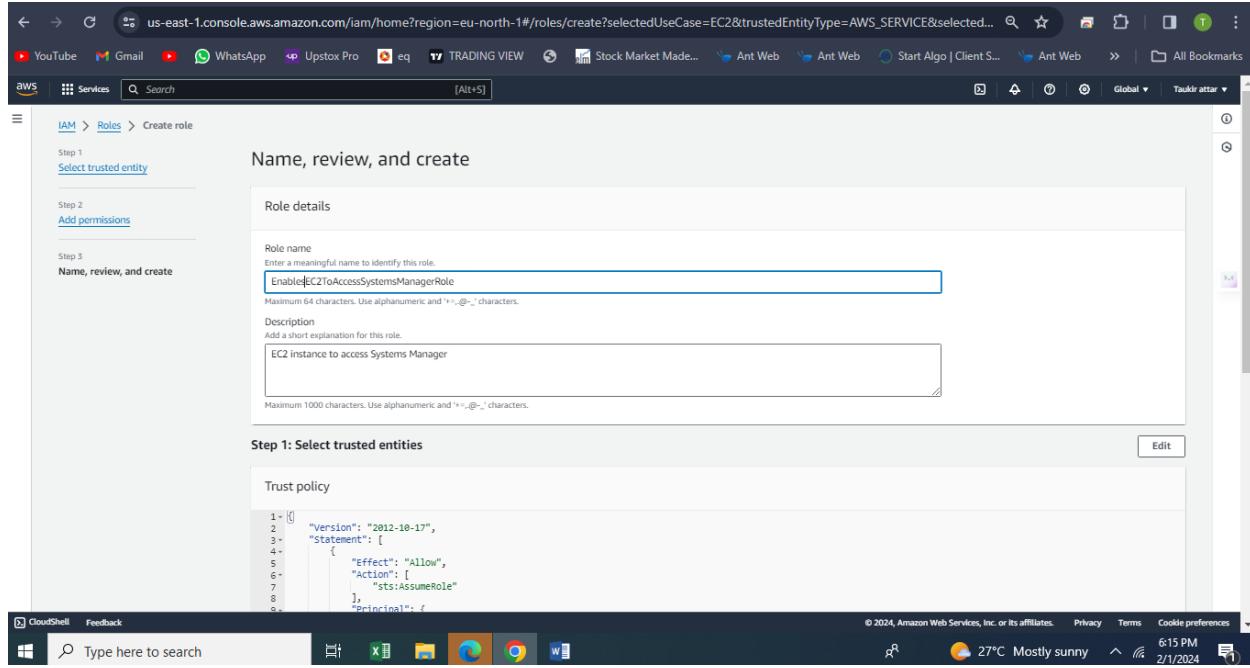
On the Select trusted entity page, under AWS Service, choose EC2, and then choose Next.

The screenshot shows the AWS IAM 'Create role' wizard at Step 1: Select trusted entity. In the 'Trusted entity type' section, 'AWS service' is selected. Other options like 'AWS account' and 'Web identity' are also listed. Below this is the 'Use case' section for EC2.

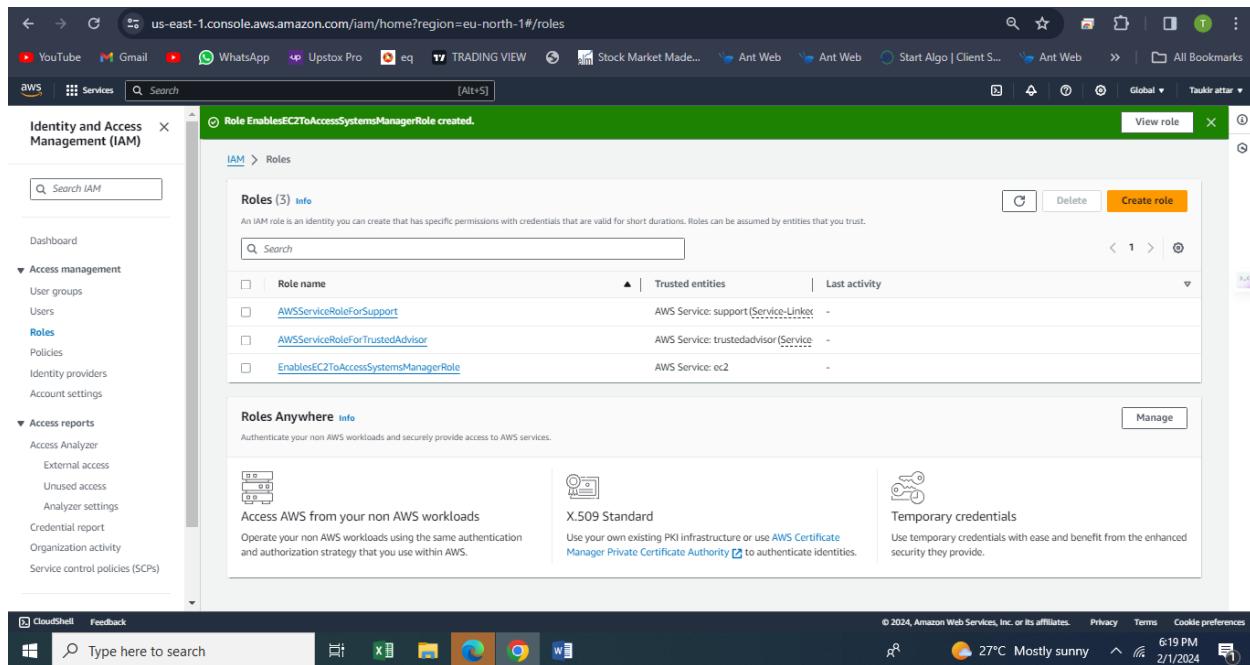
- On the Add permissions page, in the search bar type AmazonEC2RoleforSSM. From the policy list select AmazonEC2RoleforSSM and then choose Next.

The screenshot shows the AWS IAM 'Create role' wizard at Step 2: Add permissions. In the 'Permissions policies' section, 'AmazonEC2RoleforSSM' is selected from a list. The 'Next' button is highlighted.

- 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.



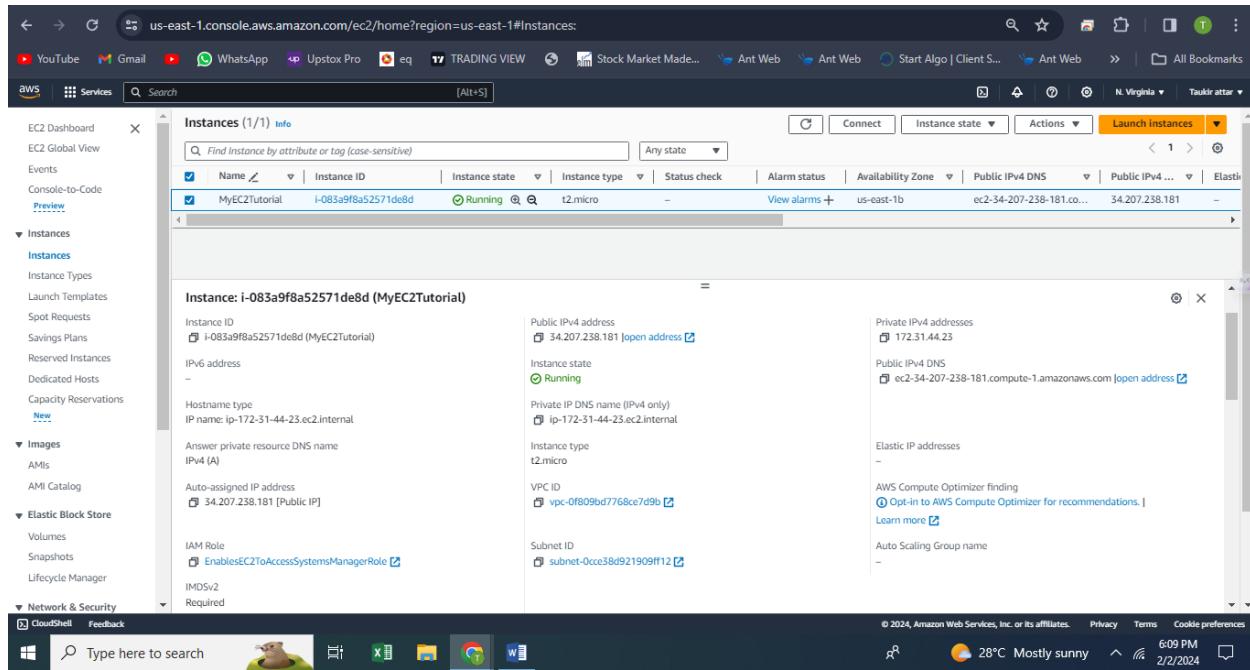
- Role is created in the below screenshot



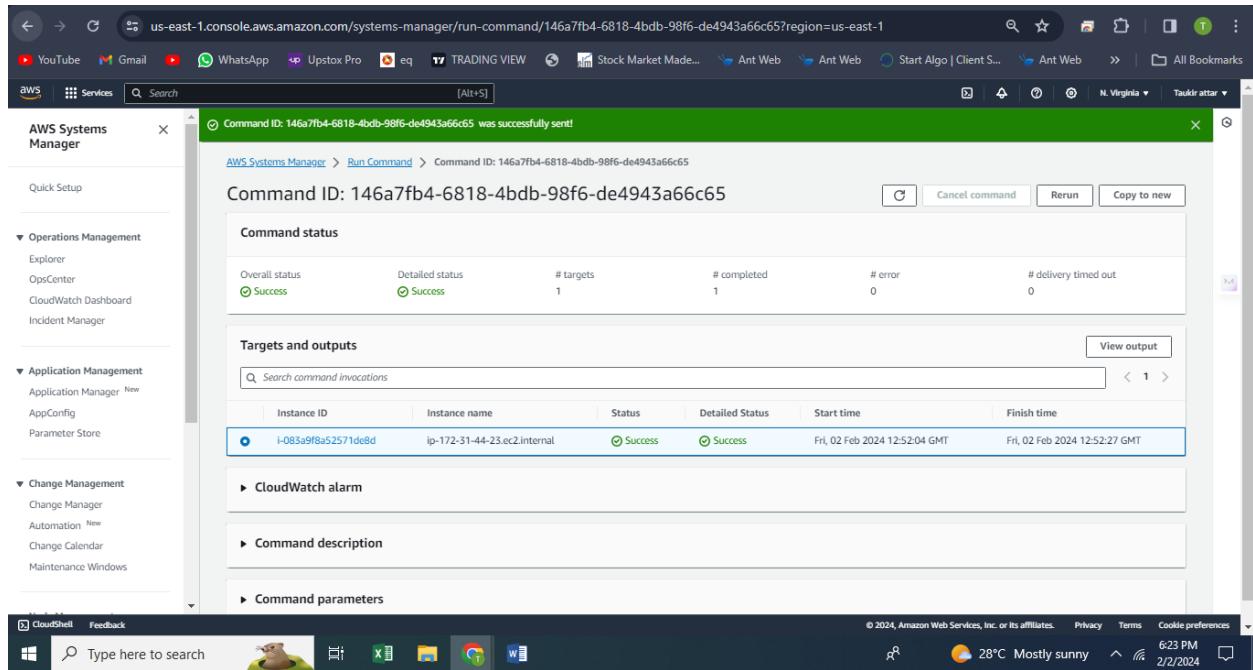
In details I had explained role in below screenshot

The screenshot shows the AWS IAM Roles page. The left sidebar navigation includes 'Identity and Access Management (IAM)', 'Dashboard', 'Access management' (with 'User groups', 'Users', and 'Roles' selected), 'Identity providers', 'Account settings', 'Access reports' (with 'Access Analyzer', 'External access', 'Unused access', 'Analyzer settings', 'Credential report', 'Organization activity', and 'Service control policies (SCPs)'), and 'CloudShell' and 'Feedback' buttons. The main content area displays the 'EnablesEC2ToAccessSystemsManagerRole' role. The 'Summary' tab shows the creation date (February 01, 2024, 18:18 (UTC+05:30)), ARN (arn:aws:iam::381492144490:role/EnablesEC2ToAccessSystemsManagerRole), and instance profile ARN (arn:aws:iam::381492144490:instance-profile/EnablesEC2ToAccessSystemsManagerRole). The 'Permissions' tab lists one policy: 'AmazonEC2RoleforSSM'. The bottom navigation bar includes links for CloudShell, Feedback, and various AWS services like Lambda, S3, and CloudWatch.

- 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*.

The screenshot shows the AWS Systems Manager Run Command interface. On the left, there's a navigation sidebar with sections like AWS Systems Manager, Quick Setup, Operations Management (Explorer, OpsCenter, CloudWatch Dashboard, Incident Manager), Application Management (Application Manager, AppConfig, Parameter Store), and Change Management (Change Manager, Automation, Change Calendar, Maintenance Windows). The main area has a search bar at the top with the query "Document name prefix: Equals: AWS-RunShellScript". Below it, a table lists a single document: "AWS-RunShellScript" by Amazon, which runs on Linux, MacOS. A "Description" section says "Run a shell script or specify the commands to run." A "Document version" dropdown is set to "1 (Default)". Under "Command parameters", there's a "Commands" section with a text box containing "1 sudo yum update -y". The bottom of the screen shows a Windows taskbar with icons for File Explorer, Task View, Google Chrome, and Word.

- 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.

The screenshot shows the AWS Systems Manager Run Command interface. A green banner at the top indicates that "Command ID: a569b3b7-7cc0-44ac-90b4-d6aa96325537 was successfully sent". Below this, the "Command ID: a569b3b7-7cc0-44ac-90b4-d6aa96325537" details are shown, including overall status (Success), detailed status (Success), number of targets (1), completed (1), errors (0), and delivery timed out (0). The "Targets and outputs" section lists one instance: i-083a9f8a52571de8d, which is also marked as Success. Below the table are sections for "CloudWatch alarm" and "Command description". The bottom right corner shows the AWS CloudShell icon and a feedback link.

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

The screenshot shows the "Output on i-083a9f8a52571de8d" panel. It starts with a "Step 1 - Command description and status" table with columns: Status (Success), Detailed status (Success), Response code (0), Step name (awsrunShellScript), Start time (Fri, 02 Feb 2024 13:04:50 GMT), and Finish time (Fri, 02 Feb 2024 13:04:51 GMT). Below this is the "Output" section, which displays command output: "Loaded plugins: extras_suggestions, langpacks, priorities, update-motd" and "No packages marked for update". There are "Copy" and "Download" buttons for this output. The "Error" section below it shows the message "No Match for argument: -y". The bottom right corner shows the AWS CloudShell icon and a feedback link.

- 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.
-

The screenshot shows the AWS EC2 Instances page with a single instance listed. The instance has been successfully terminated. The instance details are as follows:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 IP	Elastic IP	IPv6 IPs
MyEC2Tutorial	i-083a9f8a52571de8d	Shutting-down	t2.micro	2/2 checks passed	View alarms	us-east-1b	ec2-34-207-238-181.co...	34.207.238.181	-	-

Instance: i-083a9f8a52571de8d (MyEC2Tutorial)

Details | Status and alarms [New](#) | Monitoring | Security | Networking | Storage | Tags

Images

Instance ID: i-083a9f8a52571de8d (MyEC2Tutorial)
 Public IPv4 address: 34.207.238.181 [open address]
 Instance state: Shutting-down
 Private IP DNS name (IPv4 only): ip-172-51-44-25.ec2.internal
 Instance type: t2.micro
 VPC ID: vpc-0f809bd7768ce7d9b
 Subnet ID: subnet-0311000011111111

Network & Security

Auto-assigned IP address: 34.207.238.181 [Public IP]
 IAM role: AmazonSSMManagedInstanceCore

Elastic Block Store

Volumes:
 Snapshots:
 Lifecycle Manager

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

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

The screenshot shows the AWS Management Console with the URL us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#SecurityGroup:groupId=sg-06a008b670ae0f0f7. The left sidebar is collapsed, and the main content area displays a success message: "Security group (sg-06a008b670ae0f0f7 | WebServerSG) was created successfully". Below this, the security group details are shown:

Security group name	sg-06a008b670ae0f0f7	Security group ID	sg-06a008b670ae0f0f7	Description	VPC ID
Owner	38149214490	Inbound rules count	3	Inbound rules entries	vpc-0f809bd7768ce7d9b
		Outbound rules count	1	Outbound rules entries	1 Permission entry

The "Inbound rules" tab is selected, showing one rule:

Name	Security group rule...	IP version	Type	Protocol	Port range	Source
sgr-023468c0f52c681c1	IPv4	HTTP	TCP	80	0.0.0.0/0	

At the bottom right of the main content area, there is an "Actions" button.

Launching an Amazon EC2 instance

The screenshot shows the AWS Management Console with the URL us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#LaunchInstances. The left sidebar is collapsed, and the main content area displays the "Launch an instance" wizard:

Launch an instance info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags info

Name: project1

Application and OS Images (Amazon Machine Image) info

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

Search our full catalog including 1000s of application and OS images

Quick Start

Amazon Linux | macOS | Ubuntu | Windows | Red Hat | SUSE | Browse more AMIs

Summary

Number of instances: 1

Software Image (AMI): Amazon Linux 2 Kernel 5.10 AMI... read more

Virtual server type (instance type): t2.micro

Firewall (security group): WebServerSG

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

Free tier: In your first year X

includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million IOPS, 1 GB of bandwidth, and 100,000 API requests per month.

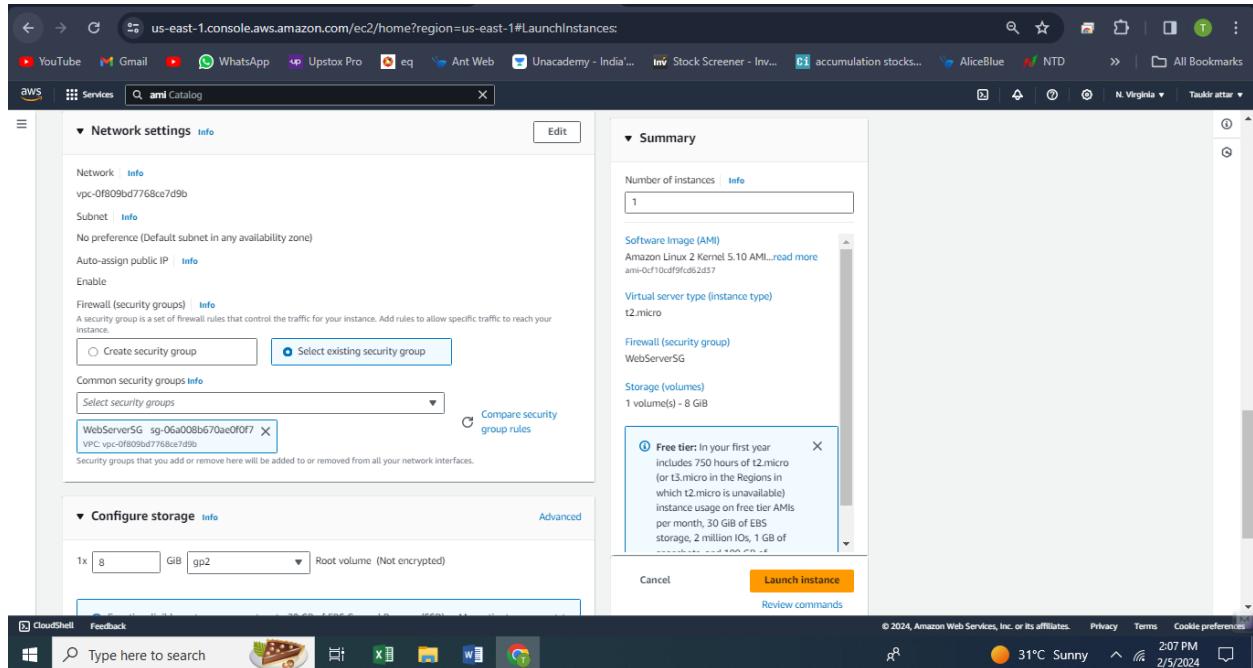
Cancel Launch instance Review commands

At the bottom right of the main content area, there is a "Review commands" button.

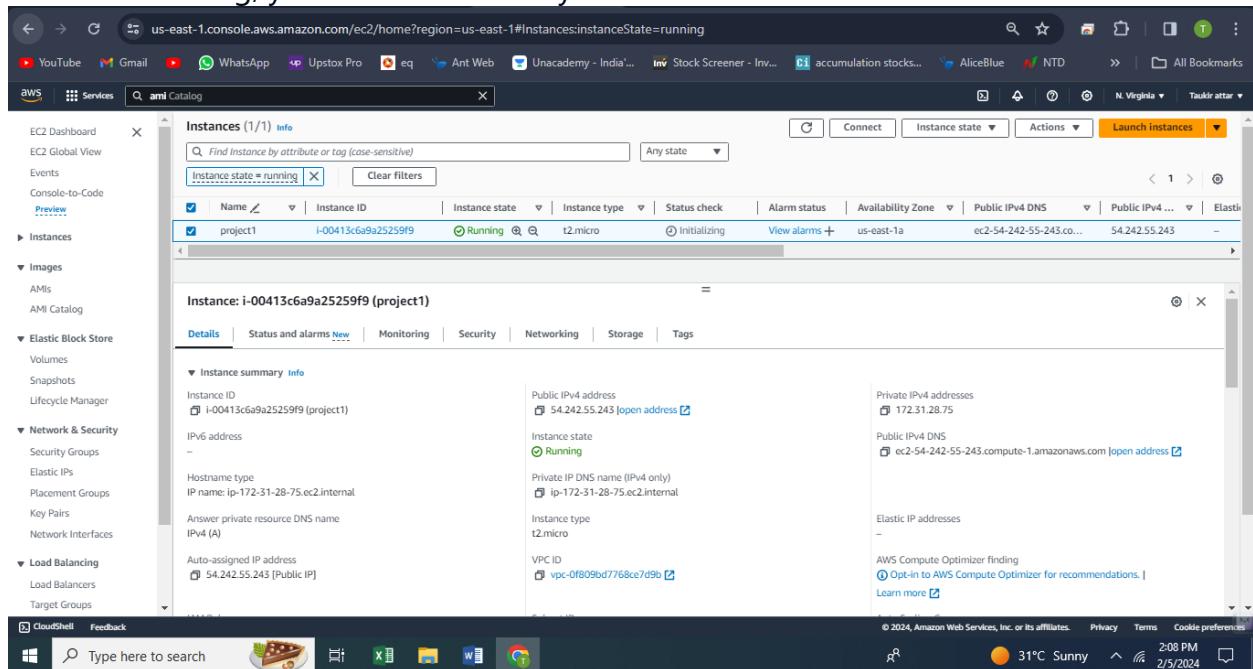
- The Choose an Amazon Machine Image (AMI) page displays a list of basic configurations called Amazon Machine Images (AMIs) that serve as templates for your instance. Select the HVM edition of the Amazon Linux AMI.

- 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.

- Select an existing security group.
- Select the WebServerSG security group that you created.
- 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.



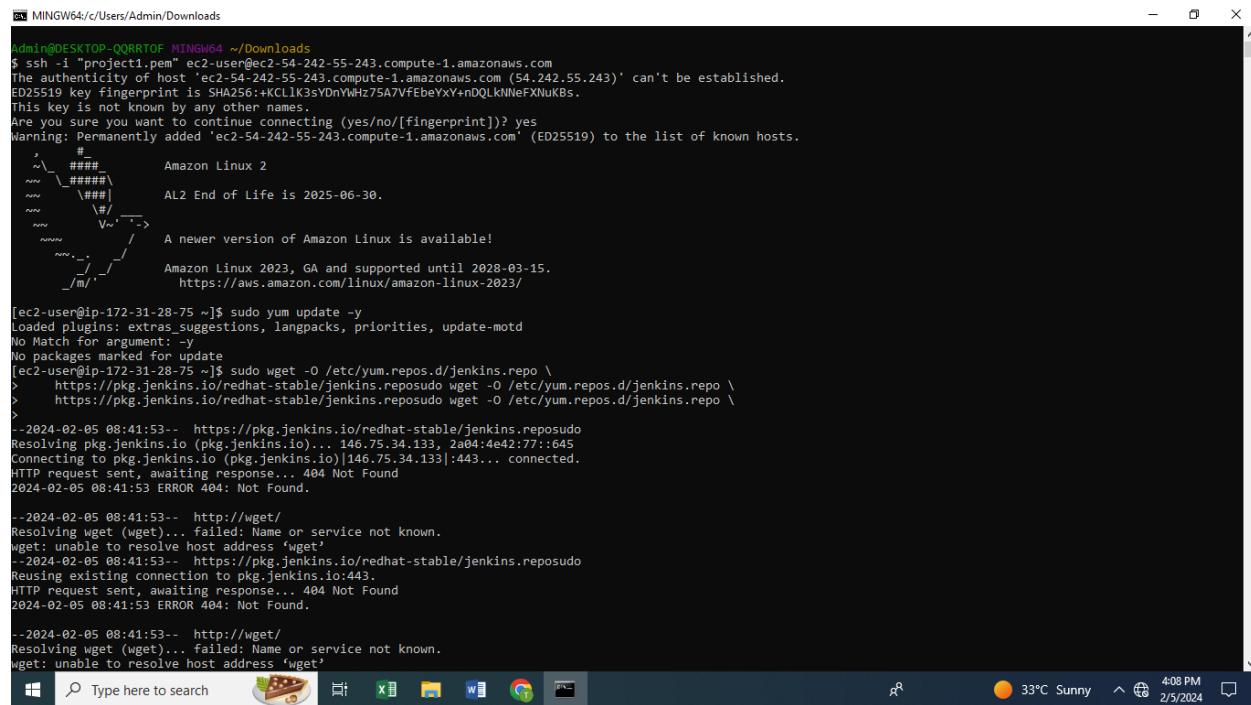
Installing and configuring Jenkins

- 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



```
admin@DESKTOP-QRRTOF MINGW64 ~\Downloads
$ ssh -i "project1.pem" ec2-user@ec2-54-242-55-243.compute-1.amazonaws.com
The authenticity of host 'ec2-54-242-55-243.compute-1.amazonaws.com (54.242.55.243)' can't be established.
ED25519 key fingerprint is SHA256:+KCLlK3syDnVHz75A7VFEBeyXXy+DQLKNefXNuKBs.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-242-55-243.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

.
~ \ ####          Amazon Linux 2
~ \###          AL2 End of Life is 2025-06-30.
~ \#/
~ \###          A newer version of Amazon Linux is available!
~ \###          Amazon Linux 2023, GA and supported until 2028-03-15.
~ \###          https://aws.amazon.com/linux/amazon-linux-2023/
~/m/ [ec2-user@ip-172-31-28-75 ~]$ sudo yum update -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
No Match for argument: -y
No packages marked for update
[ec2-user@ip-172-31-28-75 ~]$ sudo wget -O /etc/yum.repos.d/jenkins.repo \
> https://pkg.jenkins.io/redhat-stable/jenkins.reposudo wget -O /etc/yum.repos.d/jenkins.repo \
> https://pkg.jenkins.io/redhat-stable/jenkins.reposudo wget -O /etc/yum.repos.d/jenkins.repo \
>
--2024-02-05 08:41:53-- https://pkg.jenkins.io/redhat-stable/jenkins.reposudo
Resolving pkg.jenkins.io (pkg.jenkins.io)... 146.75.34.133, 2a04:4e42:77::645
Connecting to pkg.jenkins.io (pkg.jenkins.io)|146.75.34.133|:443... connected.
HTTP request sent, awaiting response... 404 Not Found
2024-02-05 08:41:53 ERROR 404: Not Found.

--2024-02-05 08:41:53-- http://wget/
Resolving wget (wget)... failed: Name or service not known.
wget: unable to resolve host address `wget'
--2024-02-05 08:41:53-- https://pkg.jenkins.io/redhat-stable/jenkins.reposudo
Reusing existing connection to pkg.jenkins.io:443.
HTTP request sent, awaiting response... 404 Not Found
2024-02-05 08:41:53 ERROR 404: Not Found.

--2024-02-05 08:41:53-- http://wget/
Resolving wget (wget)... failed: Name or service not known.
wget: unable to resolve host address `wget'
```

- [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
-

```
ea MINGW64/c/Users/Admin/Downloads
Saving to: '/etc/yum.repos.d/jenkins.repo'
100%[=====] 85      ---K/s  in 0s

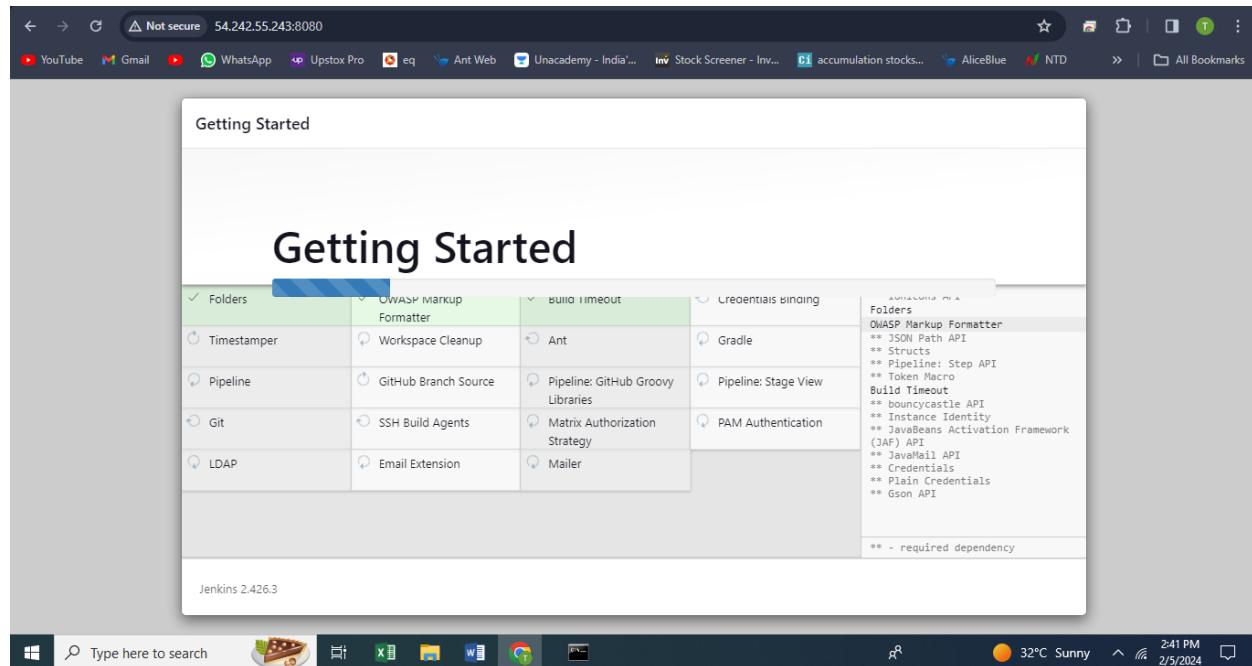
2024-02-05 08:42:32 (3.26 MB/s) - '/etc/yum.repos.d/jenkins.repo' saved [85/85]

[ec2-user@ip-172-31-28-75 ~]$ sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
[ec2-user@ip-172-31-28-75 ~]$ sudo yum upgrade
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core                                         | 3.6 kB  00:00:00
jenkins                                           | 2.9 kB  00:00:00
Jenkins/primary_db                                | 49 kB   00:00:00
No packages marked for update
[ec2-user@ip-172-31-28-75 ~]$ sudo dnf install java-17-amazon-corretto -y
sudo: dnf: command not found
[ec2-user@ip-172-31-28-75 ~]$ sudo dnf install java-17-amazon-corretto -y
sudo: dnf: command not found
[ec2-user@ip-172-31-28-75 ~]$ sudo install java-17-amazon-corretto -y
install: invalid option -- '-y'
Try 'install -help' for more information.
[ec2-user@ip-172-31-28-75 ~]$ sudo amazon-linux-extras install java-17-amazon-corretto -y
Topic java-17-amazon-corretto is not found.
[ec2-user@ip-172-31-28-75 ~]$ sudo amazon-linux-extras install java-11-amazon-corretto -y
Topic java-11-amazon-corretto is not found.
[ec2-user@ip-172-31-28-75 ~]$ sudo amazon-linux-extras install java-11-y
Topic java-11-y is not found.
[ec2-user@ip-172-31-28-75 ~]$ sudo amazon-linux-extras install java-openjdk11 -y
Topic java-openjdk11 has end-of-support date of 2024-09-30
Installing java-11-openjdk
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Cleaning repos: amzn2-core amzn2extra-docker amzn2extra-java-openjdk11 amzn2extra-kernel-5.10 jenkins
19 metadata files removed
8 sqlite files removed
0 metadata files removed
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core                                         | 3.6 kB  00:00:00
amzn2extra-docker                                | 2.9 kB  00:00:00
amzn2extra-java-openjdk11                         | 3.0 kB  00:00:00
amzn2extra-kernel-5.10                            | 3.0 kB  00:00:00
jenkins                                           | 2.9 kB  00:00:00
(1/10): amzn2-core/2/x86_64/group_gz            | 2.7 kB  00:00:00
(2/10): amzn2-core/2/x86_64/updateinfo          | 787 kB  00:00:00
(3/10): amzn2extra-java-openjdk11/2/x86_64/primary_db | 168 kB  00:00:00
(4/10): amzn2extra-kernel-5.10/2/x86_64/updateinfo | 44 kB   00:00:00
```

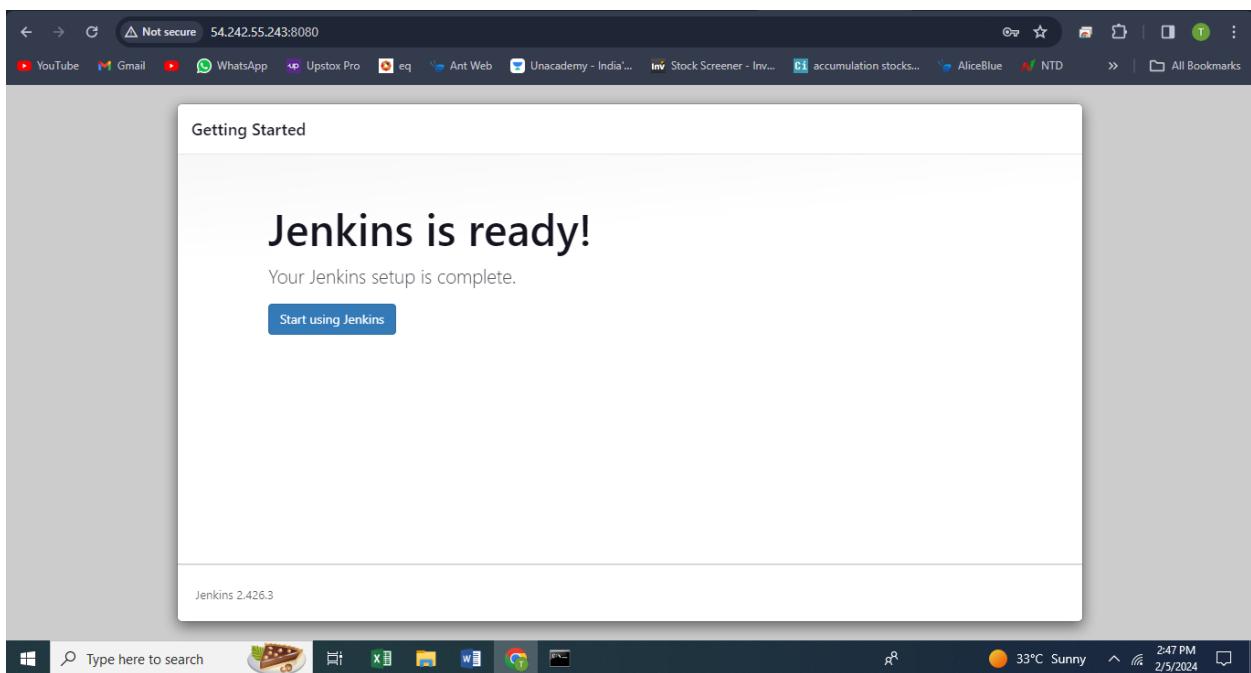
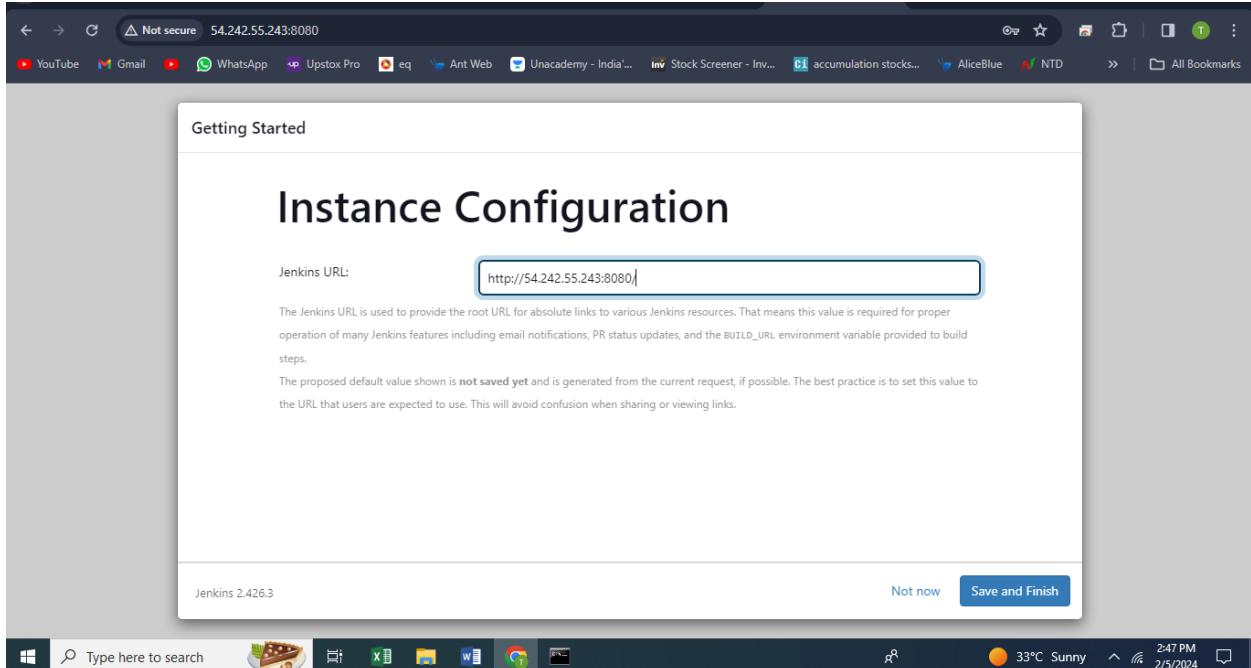
- [ec2-user ~]\$ sudo yum install jenkins -y
 - [ec2-user ~]\$ sudo systemctl enable jenkins
 - [ec2-user ~]\$ sudo systemctl start jenkins
 - [ec2-user ~]\$ sudo systemctl status jenkins

```
[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.

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

Create a job +

Build Queue
No builds in the queue.

Build Executor Status
1 Idle
2 Idle

Set up a distributed build

Set up an agent

Configure a cloud

Learn more about distributed builds ?

Amazon ec2 install

API Authorization

Plugin	Status
LDAP	Success
Email Extension	Success
Mailer	Success
Loading plugin extensions	Success
Node Iterator API	Success
Amazon Web Services SDK :: Minimal	Success
Amazon Web Services SDK :: EC2	Success
AWS Credentials	Success
Command Agent Launcher	Success
Amazon EC2	Success
Loading plugin extensions	Success

→ Go back to the top page
(you can start using the installed plugins right away)

→ Restart Jenkins when installation is complete and no jobs are running

REST API Jenkins 2.426.3

- Click **Add** under Amazon EC2 Credentials

Not secure 54.242.55.243:8080/manage/cloud/create

YouTube Gmail WhatsApp Upstox Pro eq Ant Web Unacademy - India... Stock Screener - Inv... accumulation stocks... AliceBlue NTD All Bookmarks

Dashboard > Manage Jenkins > Clouds > New cloud

The regions will be populated once the keys above are entered.

Jenkins Credentials Provider: Jenkins

Add Credentials

EC2 Keys

Domain: Global credentials (unrestricted)

Kind: AWS Credentials

Scope: Global (Jenkins, nodes, items, all child items, etc)

ID: (empty)

AMIs

Save

This screenshot shows the 'Add Credentials' dialog for Jenkins. It is titled 'Jenkins Credentials Provider: Jenkins'. Under the 'Add Credentials' section, there are dropdown menus for 'Domain' (set to 'Global credentials (unrestricted)'), 'Kind' (set to 'AWS Credentials'), and 'Scope' (set to 'Global (Jenkins, nodes, items, all child items, etc)'). Below these are input fields for 'ID' and 'AMIs', both of which are currently empty. A 'Save' button is at the bottom.

Not secure 54.242.55.243:8080/manage/cloud/create

YouTube Gmail WhatsApp Upstox Pro eq Ant Web Unacademy - India... Stock Screener - Inv... accumulation stocks... AliceBlue NTD All Bookmarks

Dashboard > Manage Jenkins > Clouds > New cloud

Availability Zone: us-east-1

Spot configuration

Security group names: WebServerSG

Remote FS root: (empty)

Remote user: (empty)

AMI Type: (empty)

Save

This screenshot shows the 'New cloud' configuration dialog for Jenkins. It includes fields for 'Availability Zone' (set to 'us-east-1'), a checkbox for 'Spot configuration' which is unchecked, and fields for 'Security group names' (containing 'WebServerSG'), 'Remote FS root' (empty), 'Remote user' (empty), and 'AMI Type' (empty). A 'Save' button is at the bottom. The dialog is divided into sections by dashed vertical lines.

The screenshot shows the AWS EC2 Instances page. A modal window titled "Successfully terminated i-00413c6a9a25259f9" is open, indicating that the instance has been terminated. The main table lists one instance:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...
project1	i-00413c6a9a25259f9	Shutting-d...	t2.micro	2/2 checks passed	View alarms +	us-east-1a	ec2-54-242-55-243.co...	54.242.55.243

The left sidebar shows the navigation menu for the EC2 service, including options like EC2 Dashboard, Instances, Images, AMIs, AMI Catalog, Elastic Block Store, Network & Security, Load Balancing, and CloudShell.

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**.