DevOps engineers are like the bridge between development and operations teams, so their skillset reflects that. Here's a breakdown of the key areas to focus on:

**Technical Skills:**

* **Coding and Scripting:** Being able to automate tasks is a big part of DevOps. Knowing languages like Python, Ruby, or Java is essential for writing scripts and building automation tools. Scripting languages like Bash are great for system administration tasks.
* **Linux:** A lot of DevOps work is done on Linux systems, so getting comfortable with the command line and how Linux works is a big plus.
* **Cloud Computing:** Most companies have moved to the cloud, so familiarity with cloud providers like AWS, Azure, or Google Cloud Platform is essential. This includes knowing how to provision resources, manage services, and keep costs optimized.
* **Version Control Systems:** These tools track changes to code and configurations. Git is the most popular version control system out there, so being a Git pro is a must-have skill.
* **Configuration Management:** Keeping all your systems configured the same way is crucial. Configuration management tools like Ansible or Puppet help automate this process.
* **Monitoring and Alerting:** Keeping an eye on your systems for performance issues and potential problems is key. DevOps engineers need to be familiar with monitoring tools and how to set up alerts.

**Soft Skills:**

* **Communication:** Being able to clearly communicate with developers, operations staff, and other stakeholders is essential.
* **Collaboration:** DevOps is all about bringing teams together. You'll need to be able to work effectively with others to achieve common goals.
* **Problem-Solving:** Things will go wrong, and it's up to the DevOps engineer to troubleshoot and fix problems.
* **Customer Focus:** Ultimately, DevOps is about delivering value to the customer. Keeping the customer's needs in mind is important.

# Jenkins

## Installing Jenkins

### Prerequisites

Minimum hardware requirements:

* 256 MB of RAM
* 1 GB of drive space (although 10 GB is a recommended minimum if running Jenkins as a Docker container)

Recommended hardware configuration for a small team:

* 4 GB+ of RAM
* 50 GB+ of drive space

**Software requirements:**

* Java: see the [Java Requirements](https://www.jenkins.io/doc/book/platform-information/support-policy-java/) page
* Web browser: see the [Web Browser Compatibility](https://www.jenkins.io/doc/administration/requirements/web-browsers/) page
* For Windows operating system: [Windows Support Policy](https://www.jenkins.io/doc/administration/requirements/windows/)
* For Linux operating system: [Linux Support Policy](https://www.jenkins.io/doc/book/platform-information/support-policy-linux/)
* For servlet containers: [Servlet Container Support Policy](https://www.jenkins.io/doc/book/platform-information/support-policy-servlet-containers/)

# Creating Ec2 instance

Name: Jenkins key

-----BEGIN RSA PRIVATE KEY-----

MIIEowIBAAKCAQEAqDu98abm7sS4RlO0AjEJNP2g93ZEdlvL5rgXvZiYsAG2M+bF

0B3zrmRhB7o6lW1Zhdfm3HHE/OAaYdfp0WItN8o4SbO/DNNsLd/bc6TIITtr6j7h

EW2UaC5P4Ap6XojONTg3nEAPCS8Dsx5k4V7dylpIntzz5DAZSAiJHc/s+HRJdzYT

QX+1M6UpCVjIpWlL5XfVnDXrbticpUF9SYn3DJhvebm9tgvJ/SFm6TV8zPEy1vKE

3GehmXKPSmmXMGOQpq2Drmo4UYOonbTHbc9ebhgcWV3h50ei7Odyporg2WgYklGp

qFHHP7USNTLbu9tqeIOsLdgipd4DO21eEOVAowIDAQABAoIBABLPm02J/Ah2tqQm

d9pG4WCre3QwJTeWzBXgP8dTaL9Fcbob88Jxlu3JC5BxoltS/s+YdJfxoLUl2c1W

9y4SbDvwwJp9Hr+wZEg836zun/IXUtm+biwktJNxmjzS7xvi8CvDPXSbNRMxzpgY

re8FTIdz0OryidLaDfmf+rT6quSc7a6vwaCJ2Xi/Htn8GBM1C04gaC1slwS1XTLa

P6Ww9OBvicRiJqXY2uEdg/5HbCUjS+PfQh5Llf4J57mRJCbBZZo682NMpCt217mm

CsM30ooOAQKV7lLkpM0IgTFTHKoQY653nHq4KJ2gDEc9LUAPHWwBLy8cte7DFUVF

mcRDvgECgYEA07kSvpuQXdcVZtbXo5JAuglgyLCkKuCOkBmLShT5wtKu8cwFqDdC

iO3lZugxb3UibM44tZLf3GCepELGem6cRlZ27DwJE2/lyZwthChxKdDgqKnf9/vx

rDYLlBEYanEP34YQPZl91mYfGmfw7xWcY35v4YFnNeB9sdHICfQ4q/sCgYEAy2pi

ezSgLeiO7yeAg4/nyIa4XgtDV/yHd5PV+ROQZM5wZp/fS/4+m9ID7SWCAtfTxKpu

s2zTFEHGTUi5jNSWyeg6lmPYf7UTI71qCw1h6sRTA1DJVypkyWFu75FiUru7PHEI

Bs5i6n/Z0GuraNVTPM6HGpfnPxhYWhBkhhKzNXkCgYBVadl1Cf8wzF928H9oHs17

/+KKtDBWl+cLnTKDPXkRLrujEoLJvRlbclXUsSGYbHHp4TwhjFJBbr0xWTsHE256

NdGeNg4XSFI0howaP9J/YyPvBzEysvJpXhhHsKnf7yLPMx+EKlMI1oP/vCUH3vkL

bmumVvUTWDB9NztzQJJIVwKBgQDKaCVua0Mn60yu9COQxdThWAyEp7hvCLzTY6t6

mNkJNdHqjBJpT6eE7Uky5CumIJR8EQruRs8STRDBPihD1hXk+suxcdCv4sIUwhmO

vdY9APWJSw3NSkXXjLgSchvgLyV0UxSlkSAkLuvR1BFP01VsICWXdKeJxPPFSqJ8

NodUIQKBgBltlfWxJr1pROui6m5U+80PEkdwkjglkswBMc5O1KXnD76QalsvGYxv

xhbjXXKQ0QiRQWUprQurII8KQ5JyFMD8lRWnvg4Mv5o/17Yy9hVguRtUR5esPxrd

gpn9Bkl4mLVYP+0qsP9X2c9SOzjua3/ZEl2r/4pGiP5EIv4UUvEk

-----END RSA PRIVATE KEY-----

## Steps

* 1. Install java
* sudo apt update
* sudo apt install fontconfig openjdk-17-jre
* java -version

2. install Jenkins

sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \

https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key

echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc]" \

https://pkg.jenkins.io/debian-stable binary/ | sudo tee \

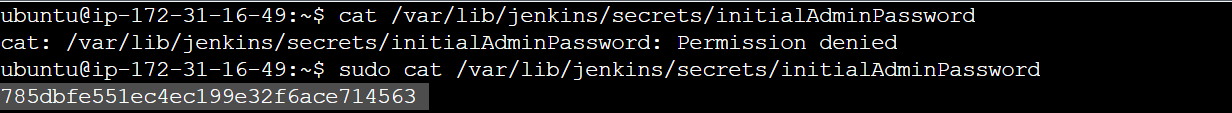
/etc/apt/sources.list.d/jenkins.list **>** /dev/null

sudo apt-get update

sudo apt-get install Jenkins

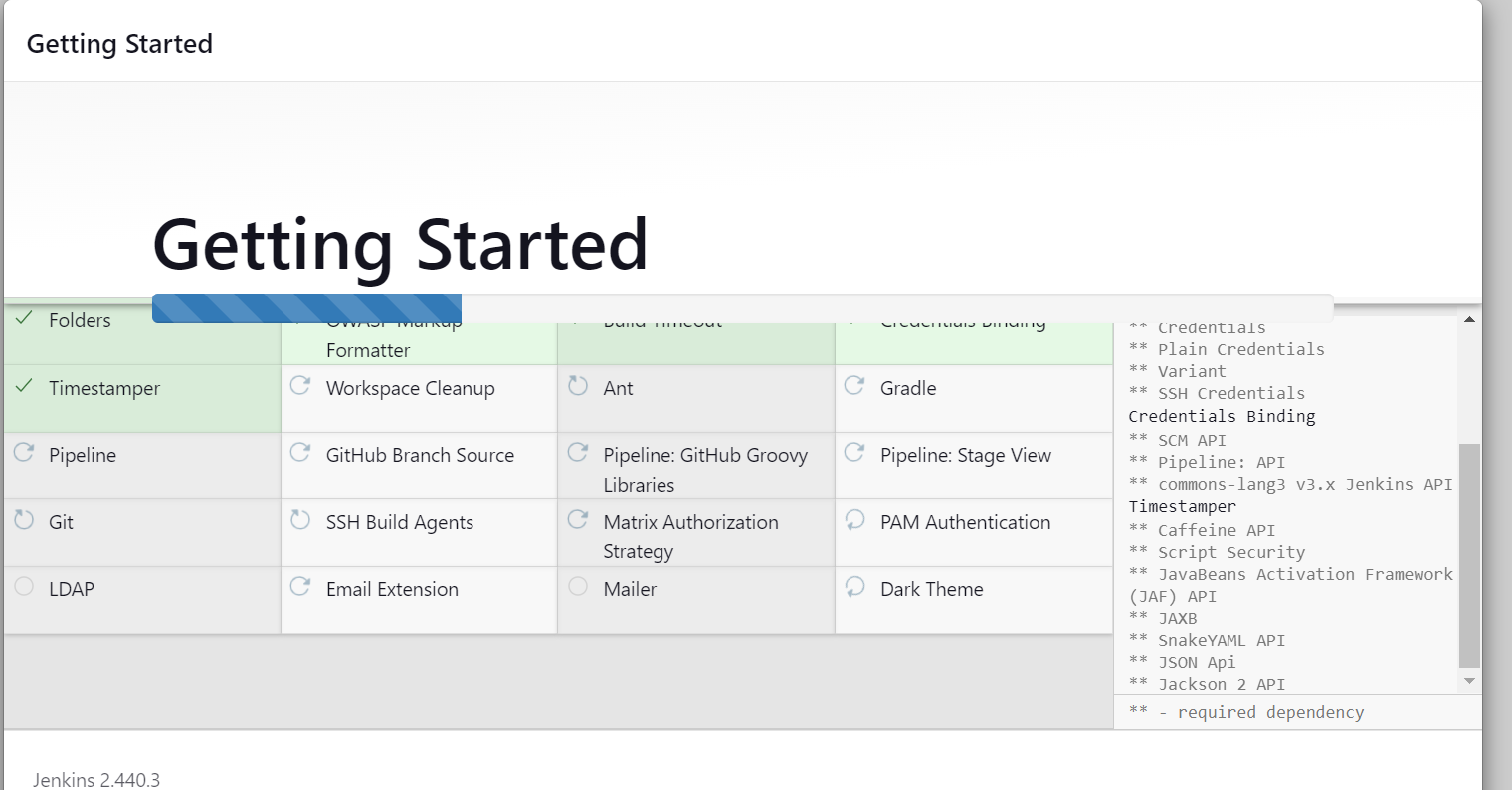
note: check if Jenkins is running or not

* 1. after running with publicIp it will ask you password:

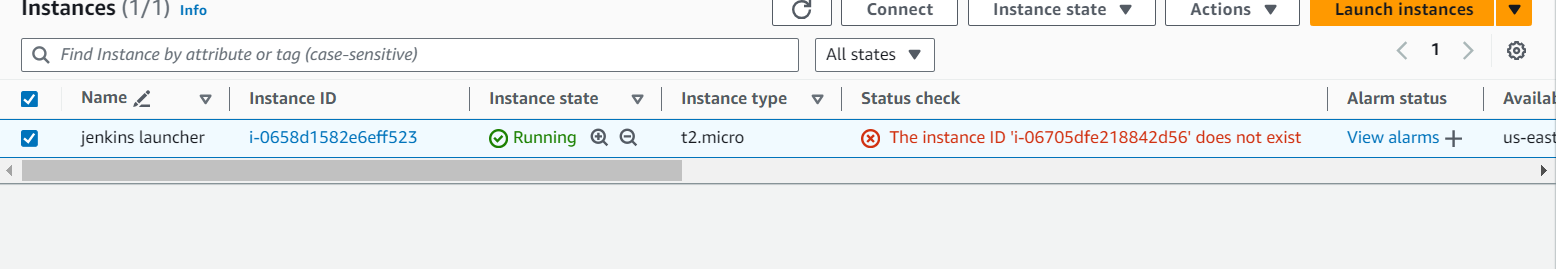


* 1. password: 785dbfe551ec4ec199e32f6ace714563

install suggested plugins:



Instance configuration: not now(this is used for using specific url for all Jenkins task.)



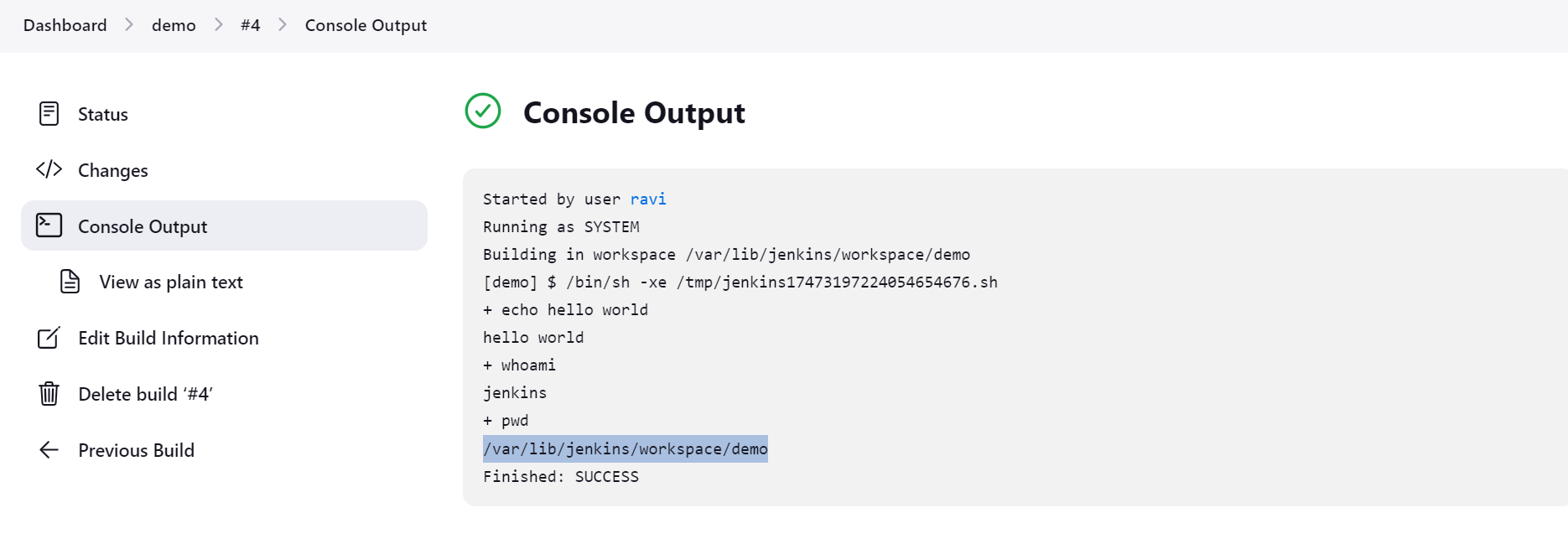
## Jenkins Dashboard

A screenshot of a computer

Description automatically generated

A white background with black text

Description automatically generated



## Search

* + it’s a navigation bar.
  + you can search your new item
  + if you search your new task name with 5 (that means 5th build)ex(demo 5)
  + go directly to console(demo 5 console)
  + last failed build(demo last failed build)

## configuration overview

configure system: any plugin configuration change.

Global tool configuration: to configure a tool configuration, location.

Manage plugins: add, remove pugins

Manage nodes & clouds: manage various node that Jenkins run.

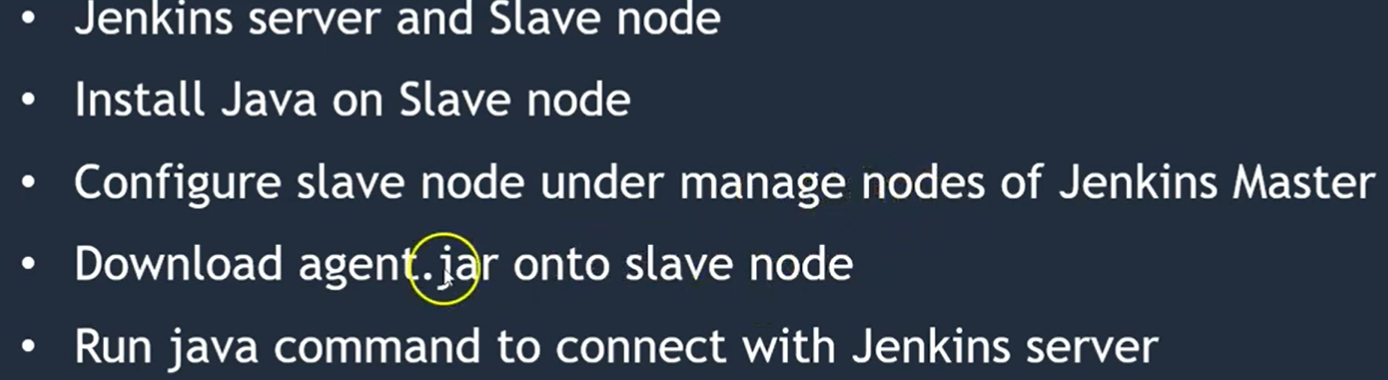
Configure global security:

* + User sign up permission.

Manage credentials: configure credentials

Manage user: create user, modify

## Master & Slave configuration:



* 1. Create 2 ec2 instance one for Jenkins server & other for build server where we can install java.
  2. Remember both ec2 instance have same version of java installed
  3. Go to **manage Jenkins** to configure slave.
  4. Go **to manage nodes & clouds**.(built in node= jenkin server.)
  5. Add a new node.(node name, select permanent agent)
  6. Define Remote root directory(build environment workspace location)it will create a directory. Next define label name
  7. Usage: only the lable name is mentioned will run in slave system.
  8. Launch method:
  + run the commands in slave system to connect to master system.
  + To run to a master node and connect to slave system.
  + Last one for window agent.
  1. Custom wordir: where we store the logs file(default use Remote root directory)
  2. Websocket(use tcp port)
  3. Avalability:
     + Always up & running
     + Cron jobs with scheduler.
     + When there is a task start the instance.
  + Right now java agent is not connected.

A screenshot of a computer

Description automatically generated

Dowenload the agent.jar file. Copy this file in slave system. Excute this command

A screen shot of a computer

Description automatically generated

Agent is connected

# Creating a pipeline in nursing app documentation

[Create a pipeline for healthcare compoent.docx](https://firminiqcom-my.sharepoint.com/:w:/g/personal/chand_ravi_firminiq_com/Ee8DpQOYWBFPvajcJodLrT8BdalD7W7gzR-naYsKPPmyAA?e=y9q6fy)

Check this document.

**How do I allow users to upload only certain file types to my Amazon S3 bucket?**

Check this link for documentation:

[How do I allow users to upload only certain file types to my Amazon S3 bucket.docx](https://firminiqcom-my.sharepoint.com/:w:/g/personal/chand_ravi_firminiq_com/ESQMJej-QONAqwvb0TB8nSUBr35Bn2ewR7PbQB-QHsjA1w?e=9rBXTp)

If you don't have permissions to do the following changes, contact your AWS Administrator.

1. Go to the IAM console and select your user

2. Attach the `AmazonS3FullAccess` policy to your IAM user

3. Go back to the S3 console and try editing the bucket policy again

If the above steps do not resolve the issue, try the following:

1. Go to the S3 console and select your bucket

2. Go to the "Permissions" tab

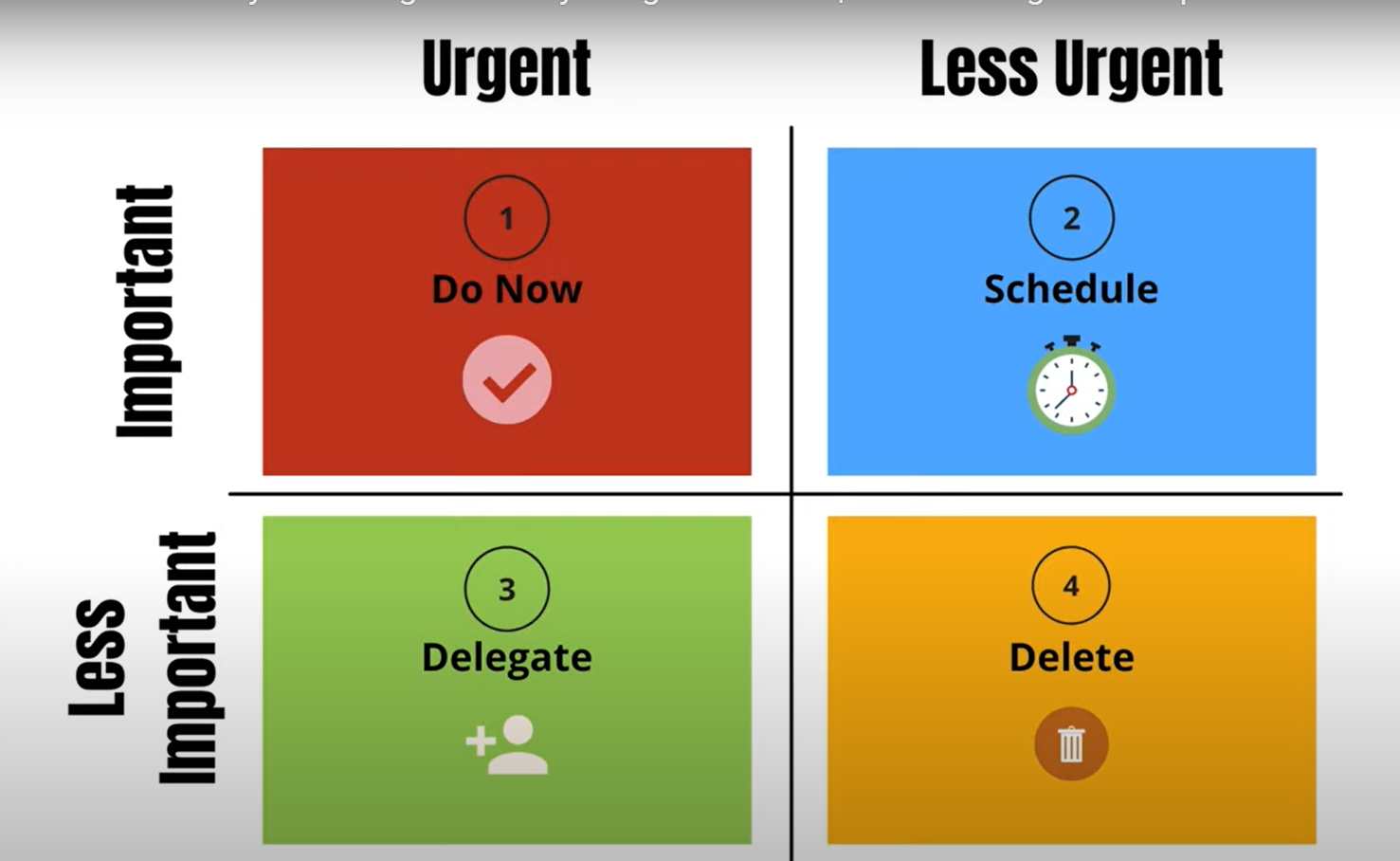
3. Scroll down to the "Block public access (bucket settings)" section

4. Ensure the "Block public access" setting is disabled

5. Save the bucket settings

6. Go back and edit the bucket policy

Time management



1. proper scedule

2. early morning

3. weekend fun

4. avoid destraction

5. app my to do list

6. app magical

priorities are:

# Giving developer/devops an access of ssh in ec2 instance

Word file: [Giving developer an access of ssh in ec2 instance.docx](https://firminiqcom-my.sharepoint.com/:w:/g/personal/chand_ravi_firminiq_com/ER7cTbTNFaxLpOU4wntQ4SMBfgnr0luqKBM2BZzaoWRsxA?e=2TrszG)

Developer: will provide you instance ID.

I am running a script which first find out the security id & then will give access to the specific ip address.

#!/bin/bash

# Get the EC2 inst$2

INSTANCE\_ID="$1"

CIDR="$2"

# Get the security group ID associated with the EC2 instance

SECURITY\_GROUP\_ID=$(aws ec2 describe-instances --instance-ids $INSTANCE\_ID --query 'Reservations[].Instances[].SecurityGroups[].GroupId' --output text)

# Update the security group inbound rules to allow access from a specific IP address

aws ec2 authorize-security-group-ingress \

--group-id $SECURITY\_GROUP\_ID \

--protocol tcp \

--port 22 \

--cidr 203.5.113.0/32

# Overwrite the values if provided as arguments

if [ $# -ge 2 ]; then

INSTANCE\_ID="$1"

CIDR="$2"

fi

A blue screen with text

Description automatically generated



A screenshot of a computer

Description automatically generated

using script:- working fine time consuming. #!/bin/bash # Get the EC2 inst$2 INSTANCE\_ID="$1" CIDR="$2" # Get the security group ID associated with the EC2 instance SECURITY\_GROUP\_ID=$(aws ec2 describe-instances --instance-ids $INSTANCE\_ID --query 'Reservations[].Instances[].SecurityGroups[].GroupId' --output text) # Update the security group inbound rules to allow access from a specific IP address aws ec2 authorize-security-group-ingress \ --group-id $SECURITY\_GROUP\_ID \ --protocol tcp \ --port 22 \ --cidr $CIDR # Overwrite the values if provided as arguments if [ $# -ge 2 ]; then INSTANCE\_ID="$1" CIDR="$2" fi using script and MFA is enabled so it will take a lot of time to configure mine aws CLI & run it. 2. \*\*Lambda trigger\*\*: as searching for the lambda is also time-consuming in my project. Note: The manual method is faster than this method. 3. Giving Aws CLI access to a user(developer) but restricting 0.0.0.0 inbound access. note can I do it i am not sure because can`t find the specific condition in iam policy.

# Task WAF lock\_token & WAF Bot

[WAF automation of lock.docx](https://firminiqcom-my.sharepoint.com/:w:/g/personal/chand_ravi_firminiq_com/ERvfuuLA2UBInTnpbBl0XC4By0KU0kpzfPpmyKDOwK7r7w?e=JAynzz)

# Task has to be done:

# Update the security group ssh task

* 1. Add a description in a security group.
  2. If the description is same then overwrite it.

1. Capacity reservation kasa work karta h.

<https://firminiqcom-my.sharepoint.com/:w:/g/personal/chand_ravi_firminiq_com/ER7cTbTNFaxLpOU4wntQ4SMBfgnr0luqKBM2BZzaoWRsxA?e=25G2oy>

# Task: 7

## Auto-scaling with Graceful Termination

[Task Auto scaling.docx](https://firminiqcom-my.sharepoint.com/:w:/g/personal/chand_ravi_firminiq_com/EQDKVmd4cUxHpgwPcs9d9JAB7Mn20KdDyfr9Q6kW2tbcoQ?e=sJirR7)

# Task 8

Update excel for all resources.