ADVANCE DEVOPS CASE STUDY

CASE STUDY TOPIC:

Automated Deployment with Monitoring

- **Concepts Used**: Jenkins, EC2, Nagios.
- **Problem Statement**: "Set up a Jenkins CI/CD pipeline to deploy a simple web application on an EC2 instance. Configure Nagios to monitor the deployed application's availability."
- **Tasks**:
    - Create a Jenkins pipeline that builds and deploys a sample web app to an EC2 instance.
    - Install and configure Nagios to monitor the HTTP status of the deployed application.
    - Verify the pipeline by triggering a build and checking the monitoring status in Nagios.

INTODUCTION

Case Study Overview:

1. Title: Automated Deployment with Monitoring
2. Focus: This case study explores the integration of Continuous Integration and Continuous Deployment (CI/CD) practices through Jenkins for deploying a web application on AWS EC2.
3. Objective: The primary aim is to automate the deployment process, allowing for faster release cycles and reduced risk of human error, thereby enhancing the overall efficiency of the software development lifecycle.
4. Monitoring Aspect: After deployment, Nagios is configured to monitor the application's availability and health, providing real-time alerts and insights into the application's operational status.

Key Feature and Application:

1. Automation:
    - Jenkins Pipeline: The Jenkins CI/CD pipeline automates the entire build, test, and deployment process. This allows developers to focus on writing code rather than managing deployments.
    - Continuous Integration: Changes made by developers can be automatically tested and integrated into the main branch, ensuring that the application remains stable and functional.
2. Real-time Monitoring:

- Nagios Monitoring: Nagios continuously checks the HTTP status of the deployed application, enabling proactive identification of issues before they affect users.
- Alerting System: The monitoring setup includes an alerting mechanism that sends notifications to the development team whenever a critical issue arises, ensuring quick response times to mitigate downtime.

3. Benefits:
    - Enhanced Reliability: By automating deployment and monitoring, organizations can ensure that their applications are always available and performing optimally.
    - Improved User Trust: Reliable service availability fosters user confidence, encouraging them to engage more frequently with the application.
    - Reduced Downtime: Prompt detection and resolution of issues minimize downtime, leading to a better user experience.

Third-Year Project Integration:

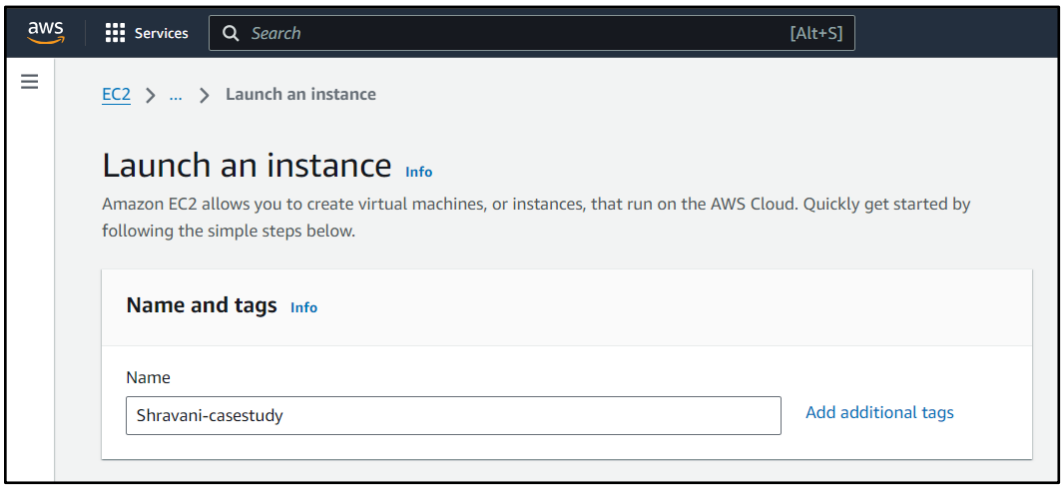
1. Project Title: Glamease – A salon appointment booking system.
2. Project Features:
    - User Capabilities: Customers can book appointments, purchase products, and compare various services based on pricing and ratings.
    - User Interface: The platform features a user-friendly interface that simplifies navigation and enhances user engagement.
3. Application of Case Study Principles:
    - Automated Deployment:
        - Integrating a Jenkins CI/CD pipeline for Glamease will facilitate the automation of deploying new features and updates. This allows for quicker rollouts of enhancements and bug fixes.
        - Ensures that every code change is validated and deployed seamlessly, maintaining high software quality.
    - Monitoring:
        - Implementing Nagios for monitoring critical services within the Glamease platform, such as appointment bookings and product transactions, ensures continuous availability and reliability.
        - With Nagios, I can set up alerts for key performance indicators, allowing for proactive management of system resources and services.
4. Outcome:
    - Enhanced Reliability: The combination of Jenkins and Nagios will provide a robust foundation for the Glamease platform, ensuring it can handle user demands effectively.
    - User Experience: By automating deployment and ensuring ongoing monitoring, users will benefit from a seamless experience when booking appointments and purchasing products.

IMPLEMENTATION

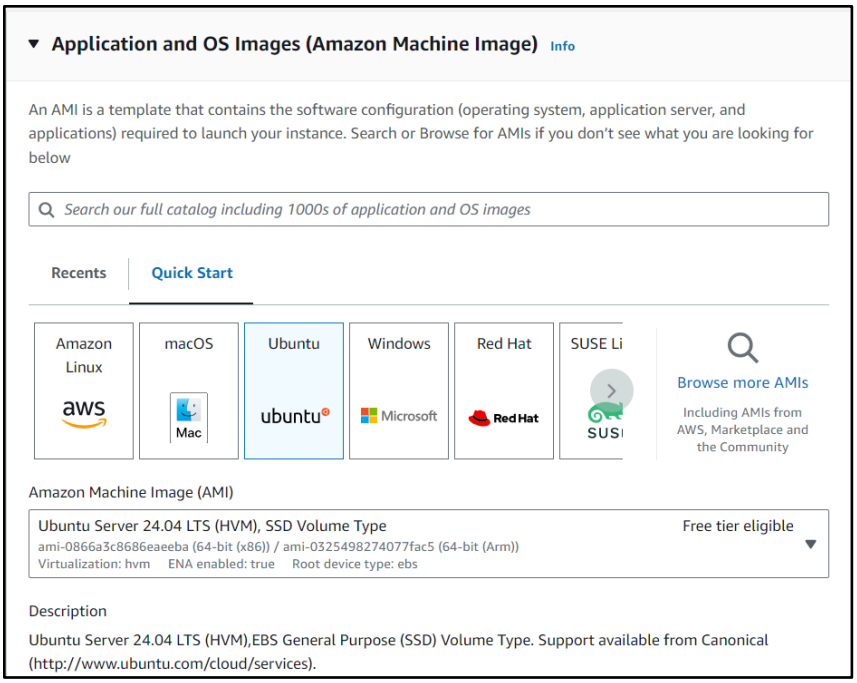
A] Create a Jenkins pipeline that builds and deploys a sample web app to an EC2 instance.

1. Launch EC2 instance:

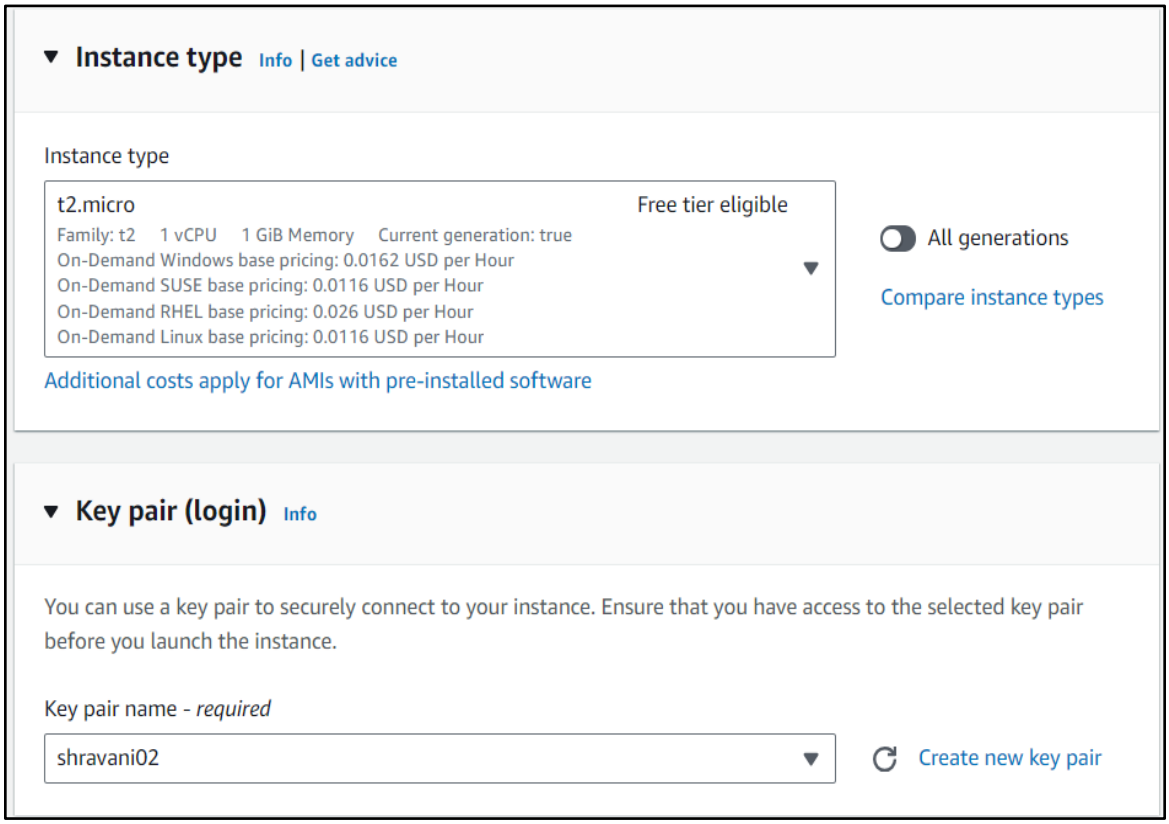
- Click on "Launch Instance" and configure:



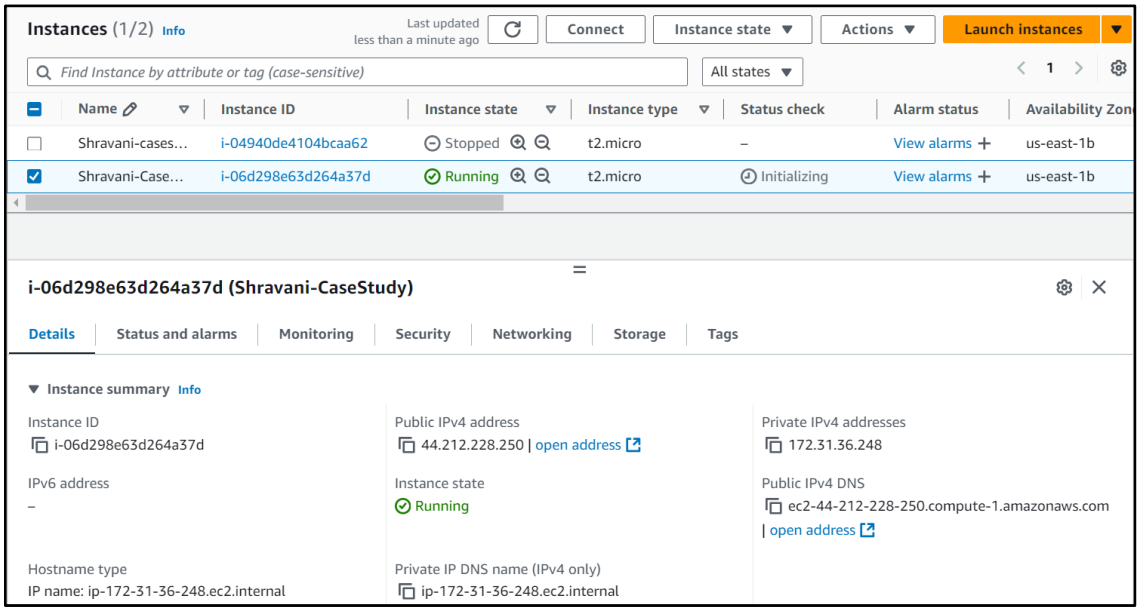
- **AMI**: Select Amazon Linux 2 or Ubuntu Server 22.04.



- **Instance type**: t2.micro (Free-tier eligible).
- **Key pair**: Create or use an existing key pair for SSH access.

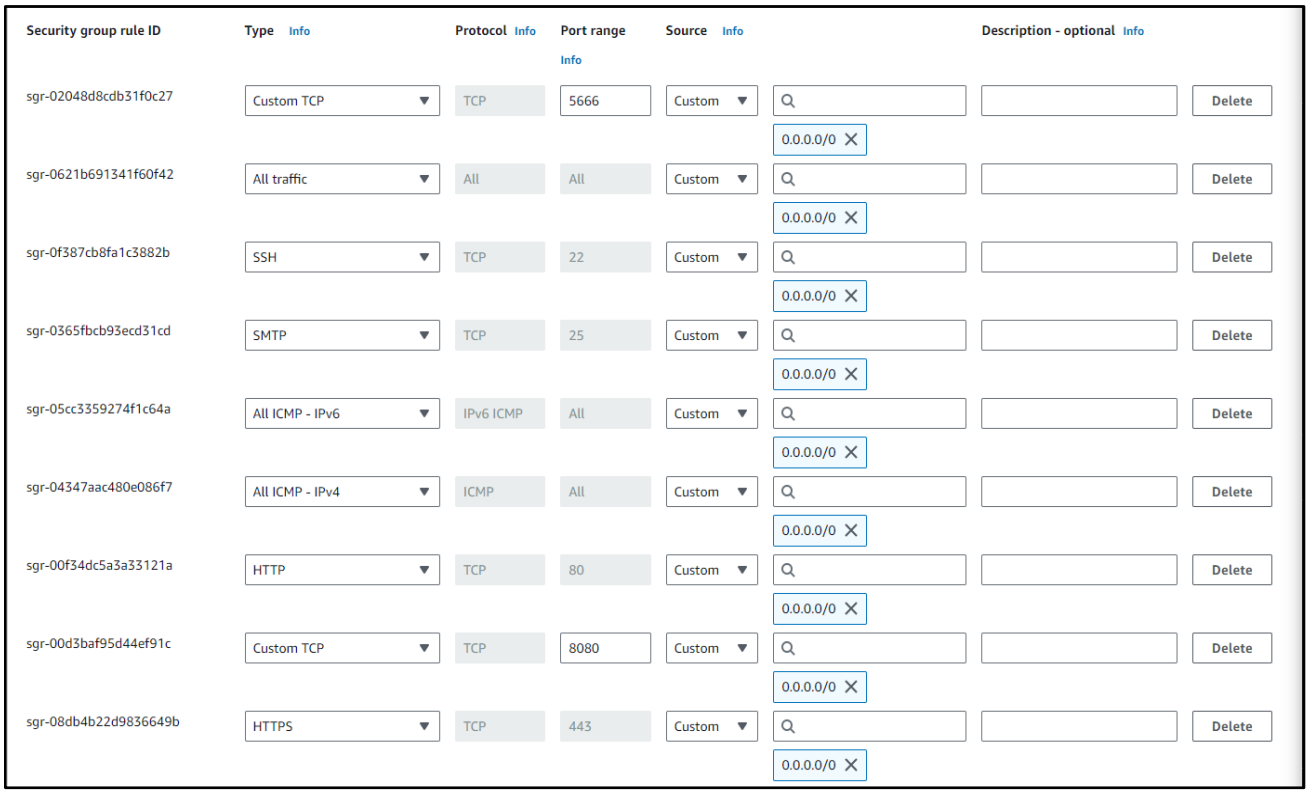


- **Launch** the instance.



2. Configure inbound rules

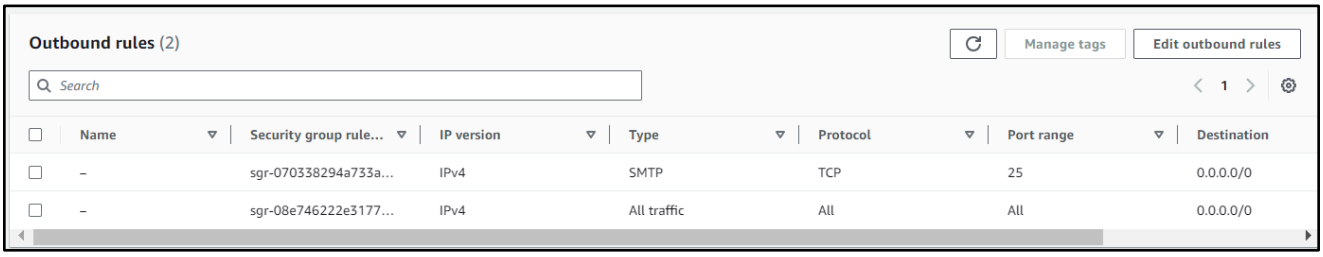
- SSH (TCP 22): Your IP.
- HTTP (TCP 80): Open to all (0.0.0.0/0).
- Nagios (TCP 5666): Open to all (0.0.0.0/0).
- All TCP (TCP 0-65535): Open to all (0.0.0.0/0).
- SMTP (TCP 587): Open to all (0.0.0.0/0)
- 8080 (Jenkins)



Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info		Description - optional Info	
sgr-02048d8cdb31f0c27	Custom TCP	TCP	5666	Custom	0.0.0.0/0		Delete
sgr-0621b691341f60f42	All traffic	All	All	Custom	0.0.0.0/0		Delete
sgr-0f387cb8fa1c3882b	SSH	TCP	22	Custom	0.0.0.0/0		Delete
sgr-0365fbcb93ecd31cd	SMTP	TCP	25	Custom	0.0.0.0/0		Delete
sgr-05cc3359274f1c64a	All ICMP - IPv6	IPv6 ICMP	All	Custom	0.0.0.0/0		Delete
sgr-04347aac480e086f7	All ICMP - IPv4	ICMP	All	Custom	0.0.0.0/0		Delete
sgr-00f34dc5a3a33121a	HTTP	TCP	80	Custom	0.0.0.0/0		Delete
sgr-00d3baf95d44ef91c	Custom TCP	TCP	8080	Custom	0.0.0.0/0		Delete
sgr-08db4b22d9836649b	HTTPS	TCP	443	Custom	0.0.0.0/0		Delete

3. Configure Outbound rule for email-notification

- SMTP (TCP 587): Open to all (0.0.0.0/0), for sending emails through Gmail's SMTP server.



Outbound rules (2)

Manage tags | Edit outbound rules

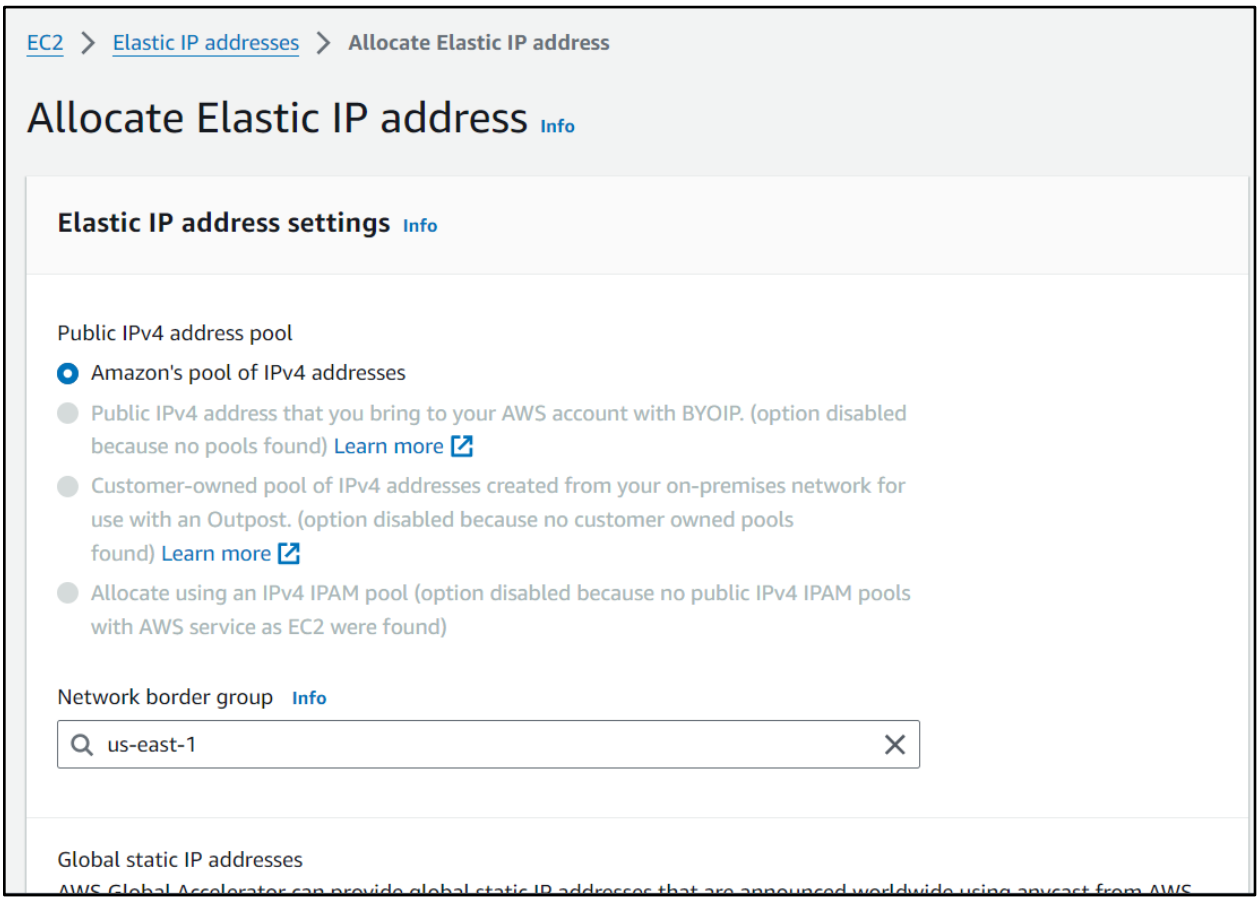
Search

1

Name	Security group rule...	IP version	Type	Protocol	Port range	Destination
–	sgr-070338294a733a...	IPv4	SMTP	TCP	25	0.0.0.0/0
–	sgr-08e746222e3177...	IPv4	All traffic	All	All	0.0.0.0/0

4. Allocate an Elastic IP Address

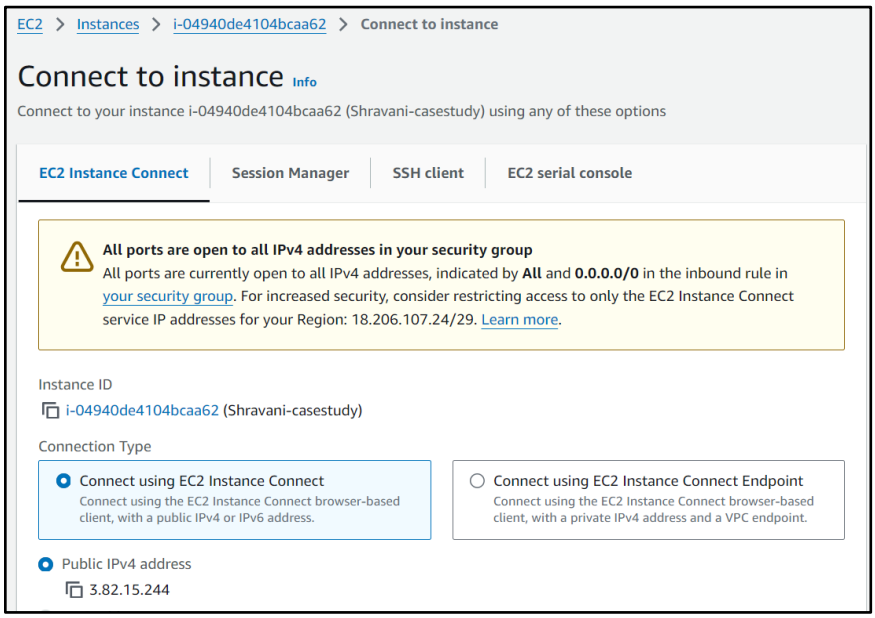
- Go to the AWS Management Console.
- Open the EC2 service.
- In the left-hand menu, under Network & Security, click on Elastic IPs.
- Click the Allocate Elastic IP address button.
- Choose the Amazon pool of IPv4 addresses, then click Allocate.



- **Associate Elastic IP with Your EC2 Instance**
  - Click Actions and choose Associate Elastic IP address.
  - In the Instance field, select the EC2 instance you want to associate with the Elastic IP.
  - Under Private IP address, choose the private IP associated with the instance (usually the default one).

5. Connect to the Instance:

- Click the Connect button at the top of the Instances page.
- In the Connect to instance window, choose the EC2 Instance Connect option.
- Ensure the username is set correctly (e.g., ec2-user for Amazon Linux, ubuntu for Ubuntu).
- Click the Connect button.



6. Install Apache (if not installed)

- sudo apt update
- sudo apt install apache2

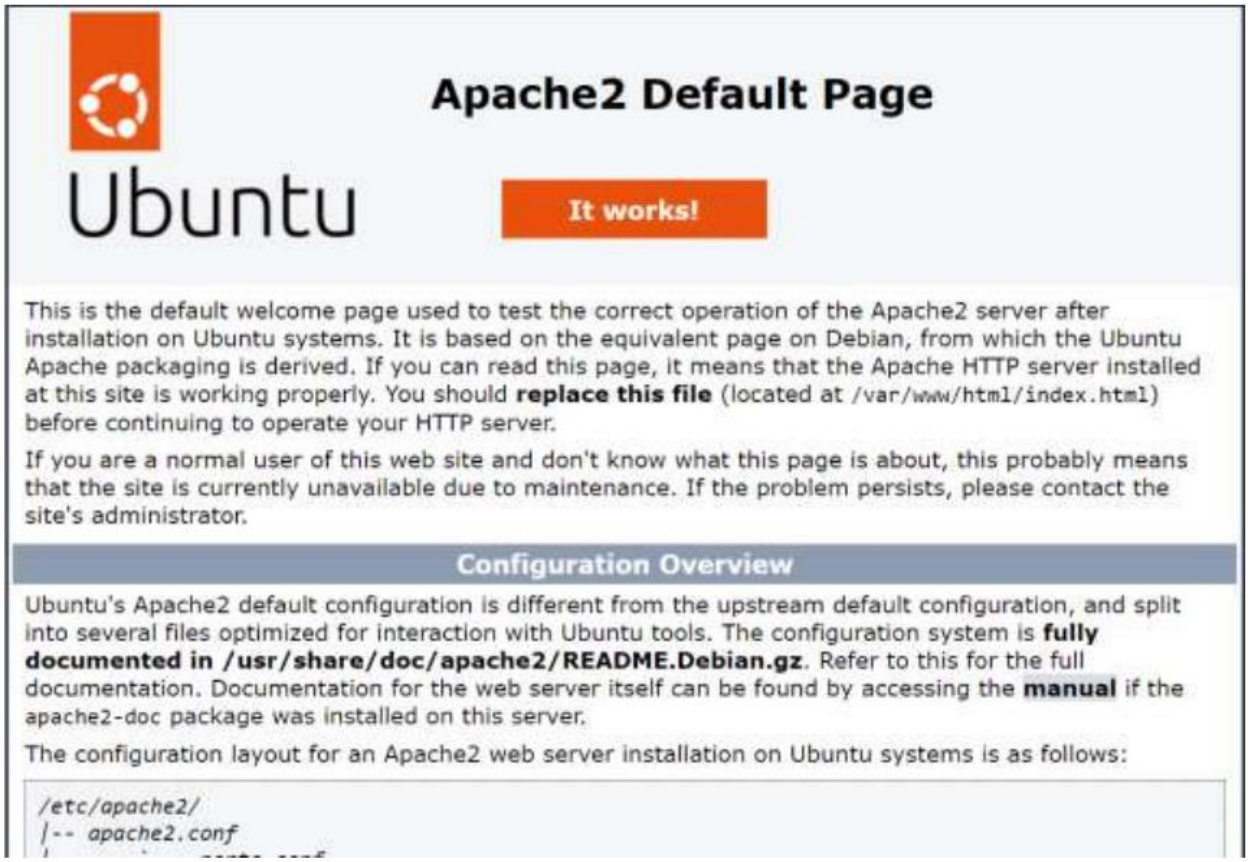
```
ubuntu@ip-172-31-33-21:~$ sudo apt update
sudo apt install apache2
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [431 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [597
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [146
```

```
ubuntu@ip-172-31-33-21:~$ sudo service apache2 start
ubuntu@ip-172-31-33-21:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
     Active: active (running) since Fri 2024-10-18 21:29:37 UTC; 1min 21s ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 2717 (apache2)
      Tasks: 55 (limit: 1130)
     Memory: 5.3M (peak: 5.6M)
        CPU: 36ms
     CGroup: /system.slice/apache2.service
             ├─2717 /usr/sbin/apache2 -k start
             ├─2720 /usr/sbin/apache2 -k start
             └─2721 /usr/sbin/apache2 -k start

Oct 18 21:29:37 ip-172-31-33-21 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Oct 18 21:29:37 ip-172-31-33-21 systemd[1]: Started apache2.service - The Apache HTTP Server.
ubuntu@ip-172-31-33-21:~$
```

Enable Apache to Start on Boot

sudo systemctl enable apache2

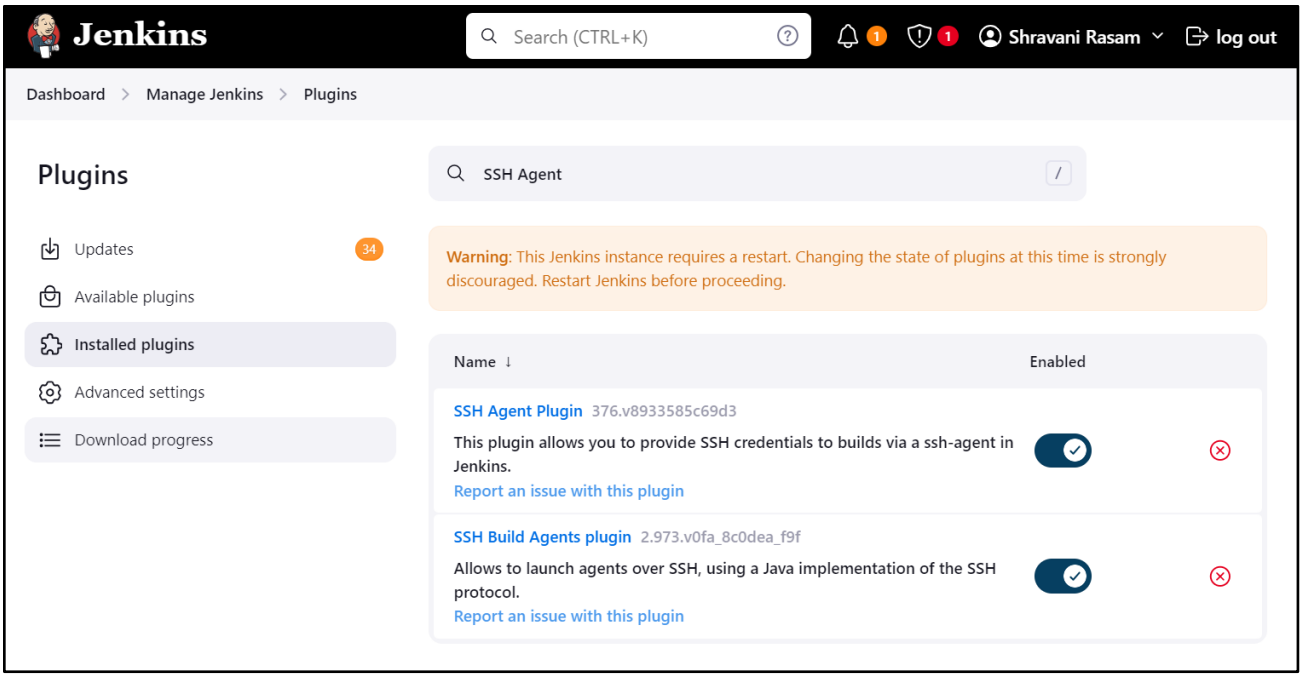
```
ubuntu@ip-172-31-33-21:~$ sudo systemctl enable apache2
Synchronizing state of apache2.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable apache2
ubuntu@ip-172-31-33-21:~$
```

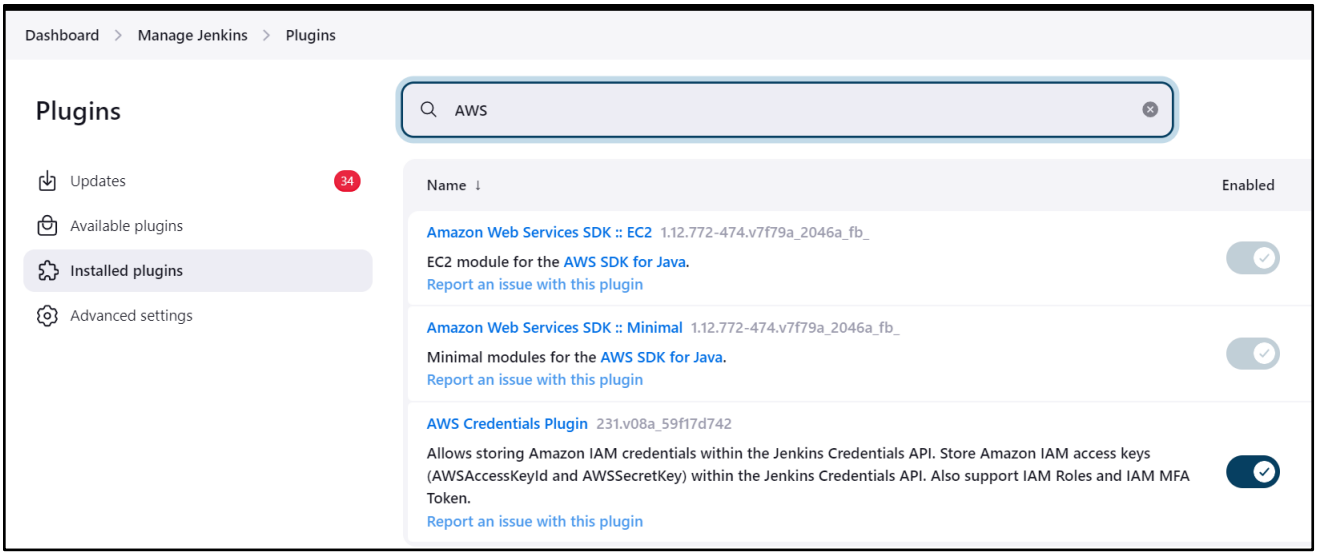


7. JENKINS Setup

Install Required Jenkins Plugins:

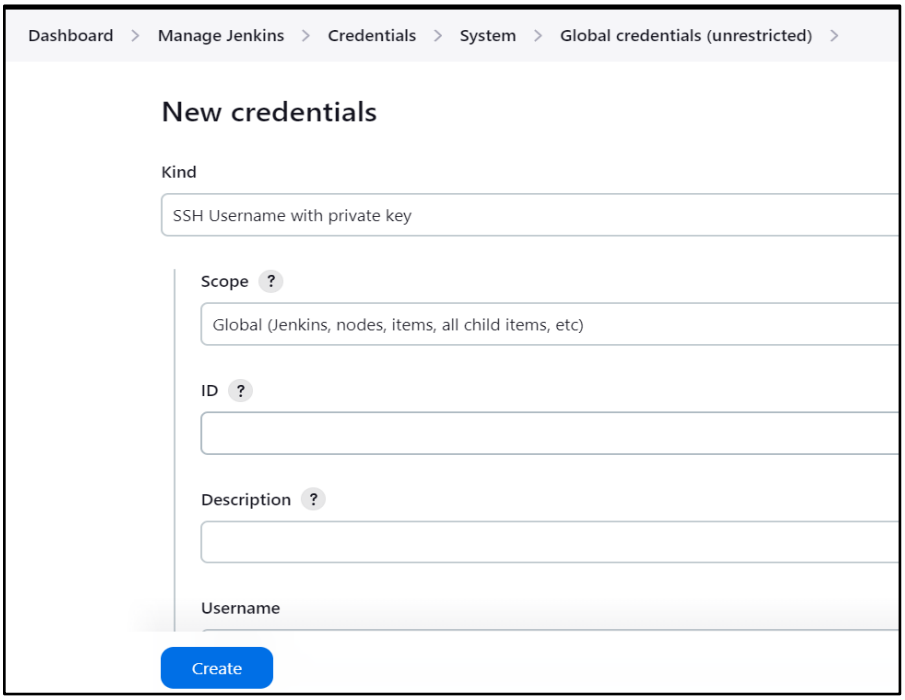
- Log in to Jenkins at `http://localhost:8080`.
- Go to `Manage Jenkins -> Manage Plugins`.
- Install the following plugins:
  - AWS EC2 Plugin (for managing EC2 instances).
  - SSH Agent Plugin (to SSH into your EC2 instance during deployment).

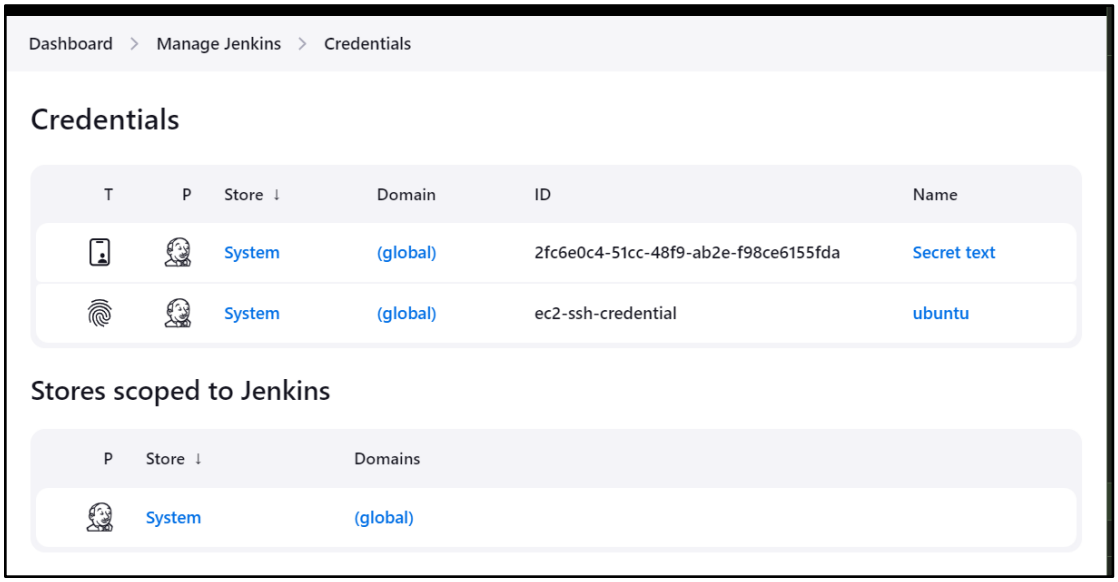




Steps to Handle the SSH Key in Jenkins:

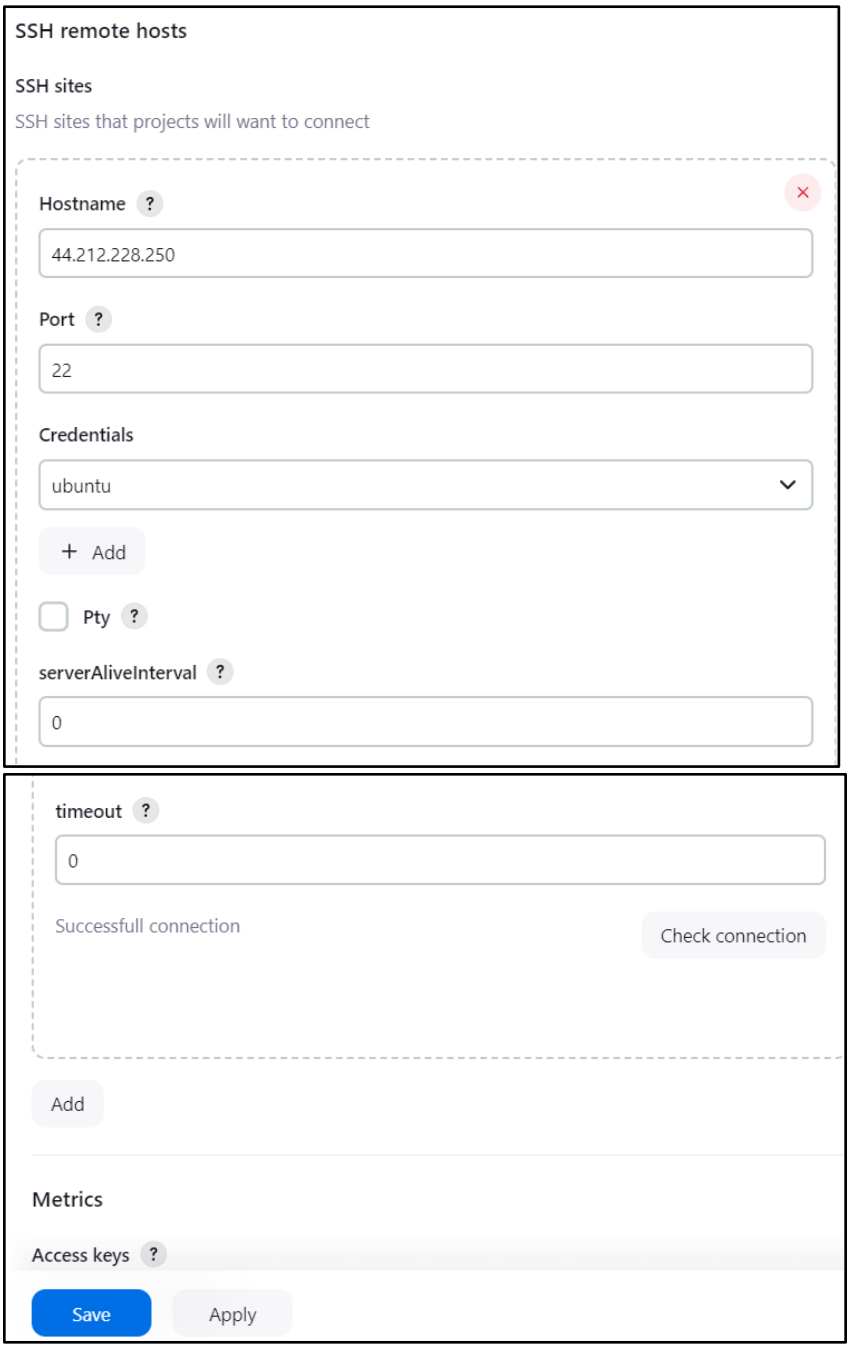
1. **Add SSH Credentials in Jenkins:**
    - Go to Manage Jenkins → Manage Credentials → (global) → Add Credentials.
    - Under Kind, select SSH Username with private key.
    - Fill in the Username (for example, `ubuntu` for Ubuntu instances).
    - For Private Key, select Enter directly, then paste the content of yourcasestudy`.pem` file.
    - Give it an ID, e.g., `ec2-ssh-credential`, for reference in the pipeline.



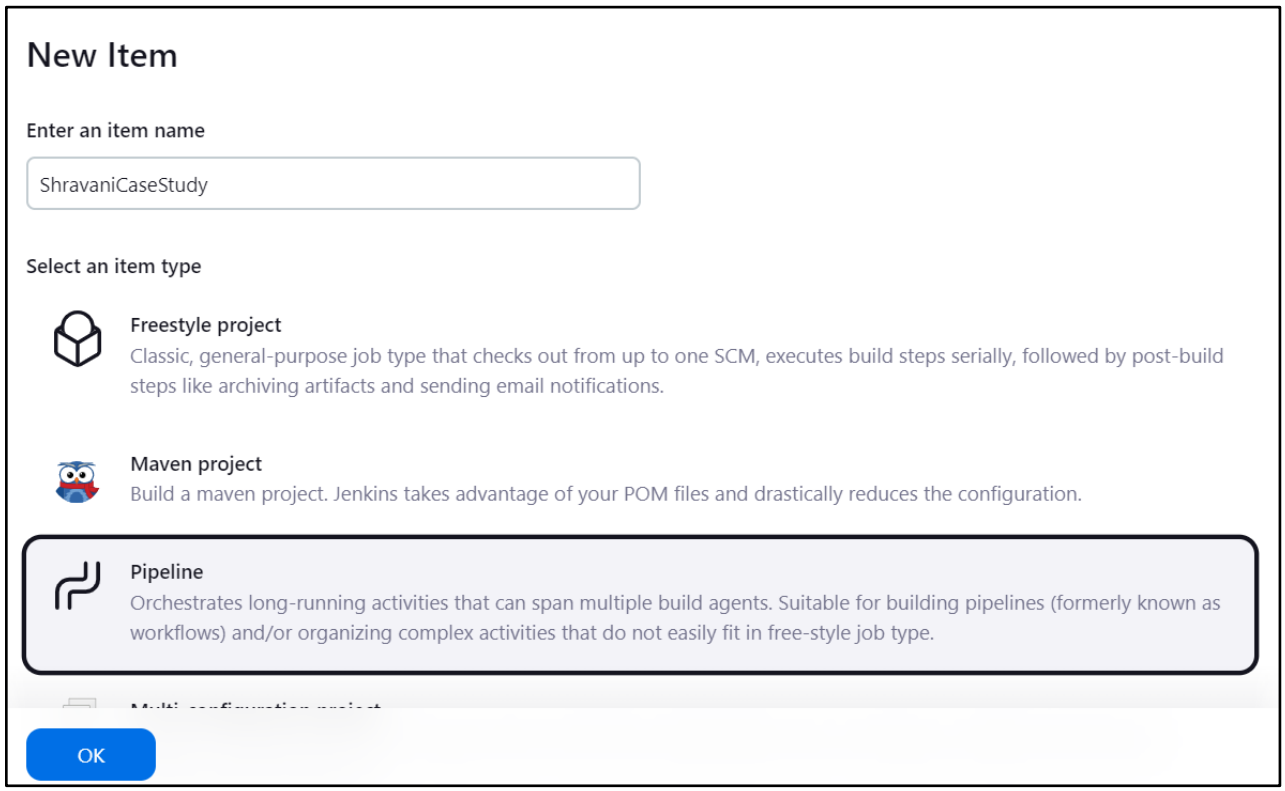


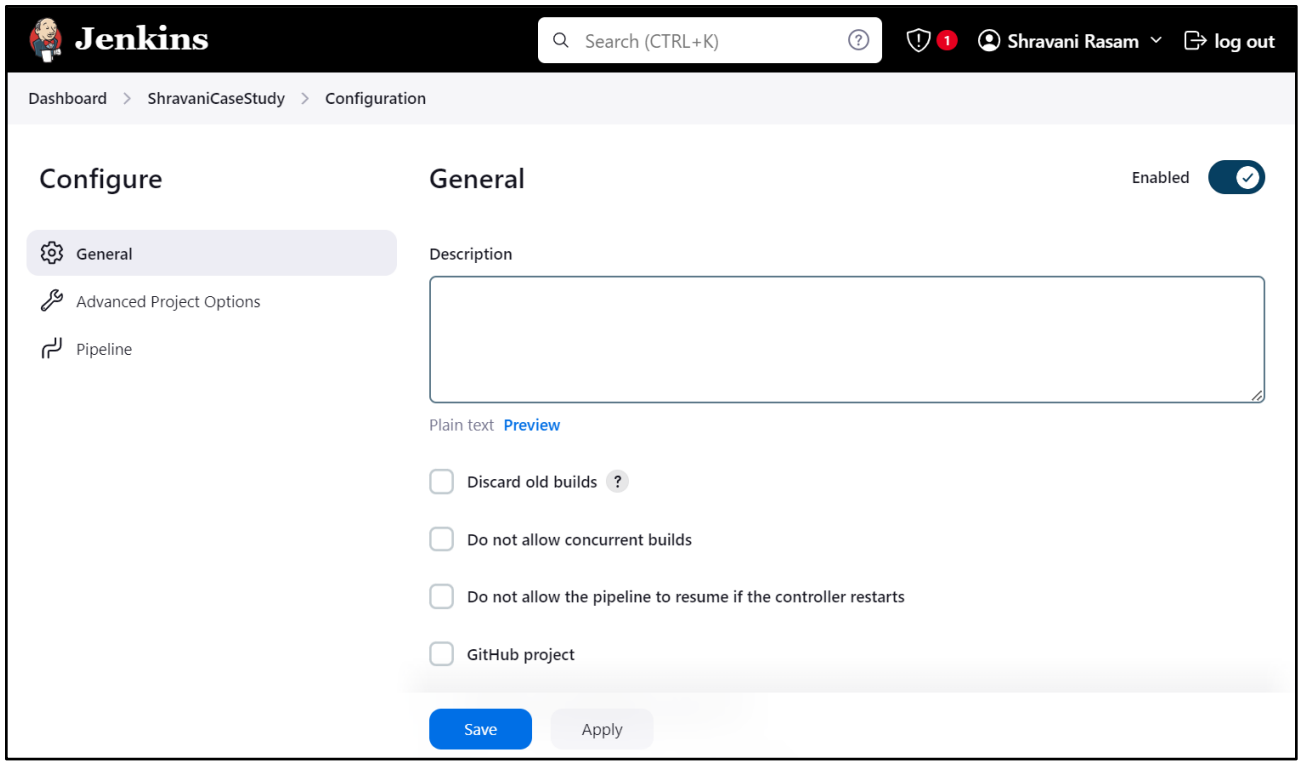
System Configuration of SSH site hosts

- Select **Configure System**.
- Scroll to **SSH Remote Hosts**.
- Click **Add SSH Host**.
  - **Hostname**: 44.212.228.250
  - **Username**: ec2-user
- Select credentials if configured.
- Click **Test Connection** to verify.



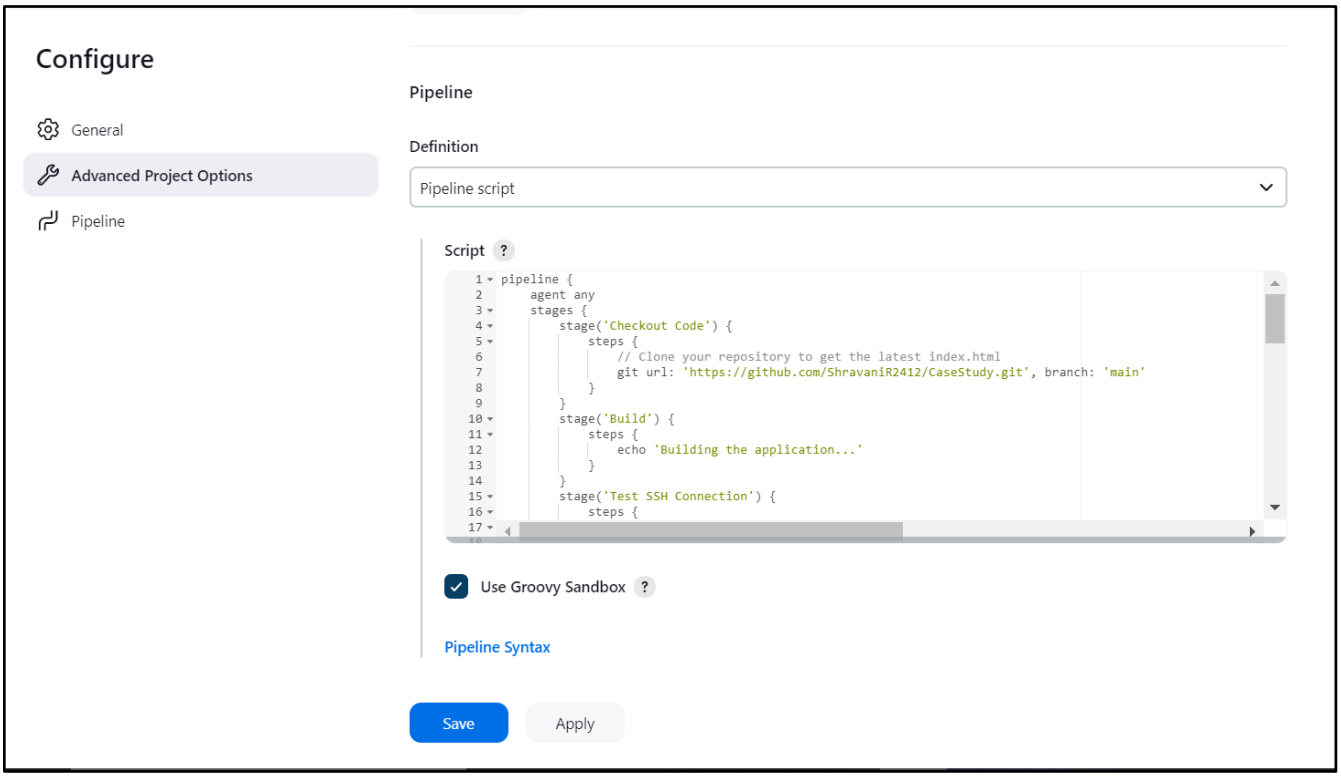
Build a Jenkins Pipeline:

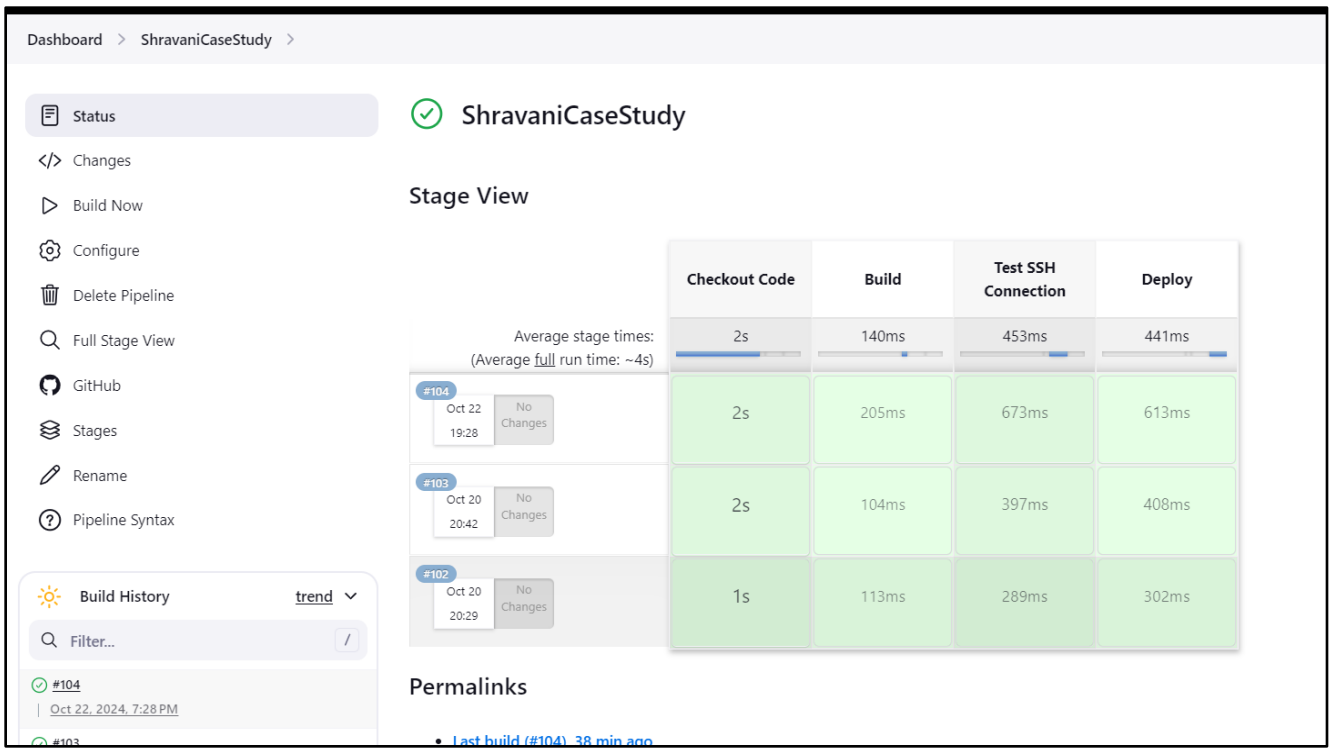


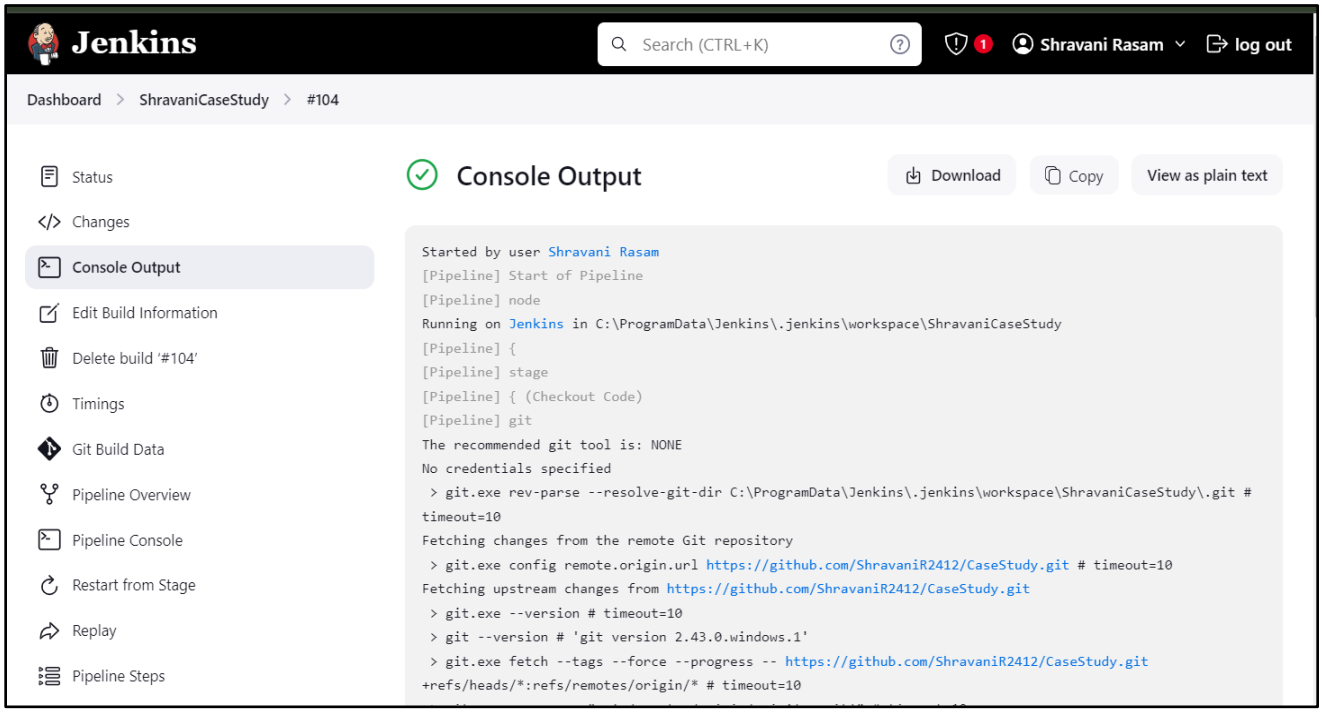


```
pipeline {
    agent any
    stages {
        stage('Checkout Code') {
            steps {
                // Clone your repository to get the latest index.html
                git url: 'https://github.com/ShravaniR2412/CaseStudy.git', branch: 'main'
            }
        }
        stage('Build') {
            steps {
                echo 'Building the application...'
            }
        }
        stage('Test SSH Connection') {
            steps {
                script {
                    echo "Testing SSH connection to EC2 instance..."
                    try {
                        sshagent(['ec2-ssh-credential']) {
                            // Attempt to run a simple command on the EC2 instance
                            sh """
                            ssh -o StrictHostKeyChecking=no -i
"C:/Users/Shravani/Downloads/casestudy.pem" ubuntu@ec2-44-212-228-250.compute-
1.amazonaws.com "echo 'SSH connection successful'"
                            """
                        }
                        echo "SSH connection test successful."
                    } catch (Exception e) {
                        echo "SSH connection test failed: ${e.getMessage()}"
                    }
                }
            }
        }
        stage('Deploy') {
            steps {
                script {
                    // Set the path to the index.html file in the cloned repository
                    def indexFilePath = "${WORKSPACE}/index.html"

                    echo "Workspace directory: ${WORKSPACE}"
                    echo "Index file path: ${indexFilePath}"

                    try {
```

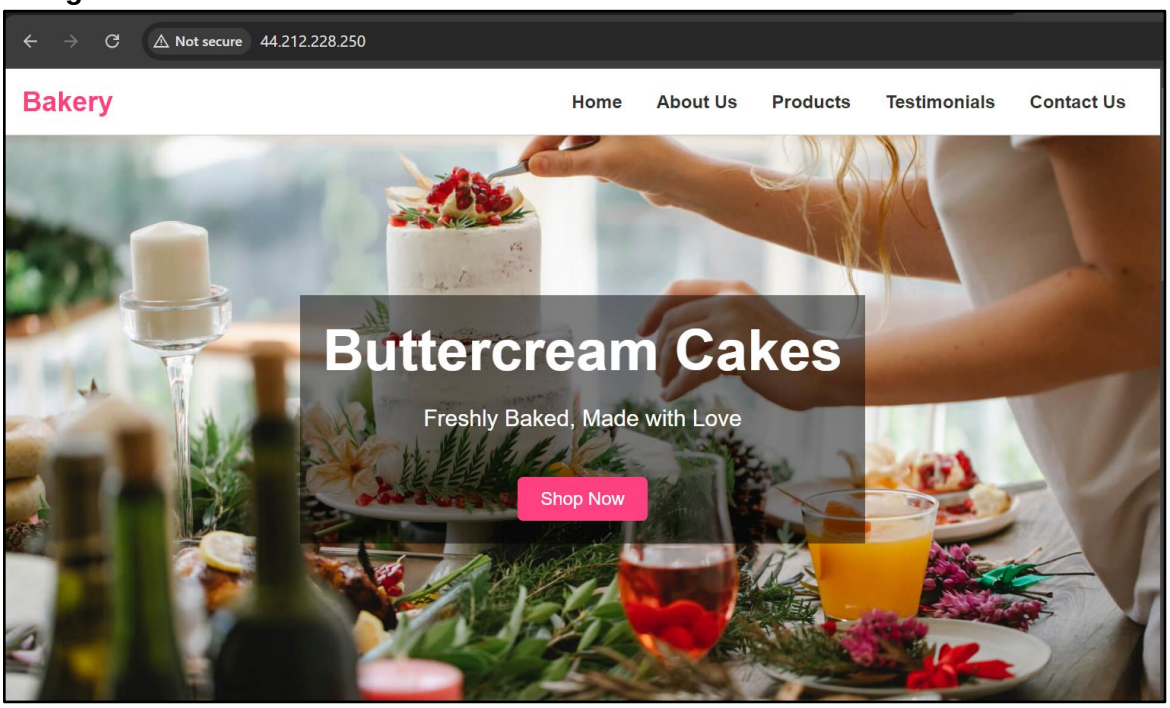
```
                echo "Copying index.html to EC2 instance..."
                sshagent(['ec2-ssh-credential']) {
                    sh """
                    scp -o StrictHostKeyChecking=no -i
"C:/Users/Shravani/Downloads/casestudy.pem" ${indexFilePath} ubuntu@ec2-44-212-228-
250.compute-1.amazonaws.com:/var/www/html/index.html
                    """
                }

                echo "Restarting Apache server..."
                sshagent(['ec2-ssh-credential']) {
                    sh """
                    ssh -o StrictHostKeyChecking=no -i
"C:/Users/Shravani/Downloads/casestudy.pem" ubuntu@ec2-44-212-228-250.compute-
1.amazonaws.com "sudo systemctl restart apache2"
                    """
                }

                        } catch (Exception e) {
                echo "Error during deployment: ${e.getMessage()}"
            }
        }}}}
```

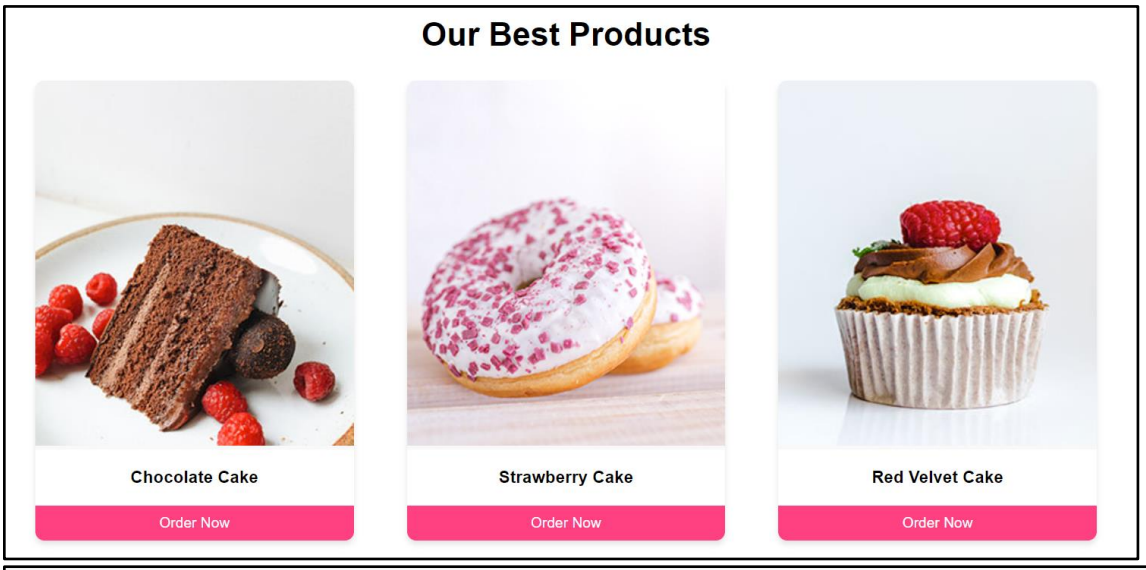



Dashboard > ShravaniCaseStudy >
Status
Changes
Build Now
Configure
Delete Pipeline
Full Stage View
GitHub
Stages
Rename
Pipeline Syntax
Build History
trend
Filter...
#104
Oct 22, 2024, 7:28 PM
ShravaniCaseStudy
Stage View
Checkout Code
Build
Test SSH Connection
Deploy
Average stage times:
(Average full run time: ~4s)
2s
140ms
453ms
441ms
#104
Oct 22
19:28
No Changes
2s
205ms
673ms
613ms
#103
Oct 20
20:42
No Changes
2s
104ms
397ms
408ms
#102
Oct 20
20:29
No Changes
1s
113ms
289ms
302ms
Permalinks
Last build (#104), 38 min ago


Jenkins
Search (CTRL+K)
Shravani Rasam
log out
Dashboard > ShravaniCaseStudy > #104
Status
Changes
Console Output
Edit Build Information
Delete build '#104'
Timings
Git Build Data
Pipeline Overview
Pipeline Console
Restart from Stage
Replay
Pipeline Steps
Console Output
Download
Copy
View as plain text
Started by user Shravani Rasam
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in C:\ProgramData\Jenkins\.jenkins\workspace\ShravaniCaseStudy
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Checkout Code)
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
> git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\ShravaniCaseStudy\.git # timeout=10
Fetching changes from the remote Git repository
> git.exe config remote.origin.url https://github.com/ShravaniR2412/CaseStudy.git # timeout=10
Fetching upstream changes from https://github.com/ShravaniR2412/CaseStudy.git
> git.exe --version # timeout=10
> git --version # 'git version 2.43.0.windows.1'
> git.exe fetch --tags --force --progress -- https://github.com/ShravaniR2412/CaseStudy.git +refs/heads/*:refs/remotes/origin/* # timeout=10

Navigate to the Elastic IP address in the browser





B] Install and configure Nagios to monitor the HTTP status of the deployed application.

Connect to Your EC2 Instance

- SSH into your EC2 instance or use EC2 Instance Connect from the browse
  sudo apt update
  sudo apt install nagios4 nagios-plugins-contrib nagios-nrpe-plugin -y

```
ubuntu@ip-172-31-36-248:~$ sudo apt update
sudo apt upgrade -y
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [114 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [306 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Components [212 B]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [940 B]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [208 B]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [21.2 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Get:13 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [7212 B]
Get:14 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [51.9 kB]
Get:15 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [208 B]
Get:16 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [208 B]
Fetched 881 kB in 1s (944 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
25 packages can be upgraded. Run 'apt list --upgradable' to see them.
ubuntu@ip-172-31-36-248:~$ sudo apt upgrade -y
Reading package lists... Done
Building dependency tree... Done
```

sudo apt install apache2 php libapache2-mod-php build-essential libgd-dev -y

```
ubuntu@ip-172-31-36-248:~$ sudo apt install -y apache2 libapache2-mod-php php gcc make autoconf libgd-dev libmcrypt-dev wget unzip
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
apache2 is already the newest version (2.4.58-1ubuntu8.4).
wget is already the newest version (1.21.4-1ubuntu4.1).
wget set to manually installed.
The following additional packages will be installed:
  automake autotools-dev binutils binutils-common binutils-x86-64-linux-gnu bzip2-doc cpp cpp-13 cpp-13-x86-64-linux-gnu
  cpp-x86-64-linux-gnu fontconfig-config fonts-dejavu-core fonts-dejavu-mono gcc-13 gcc-13-base gcc-13-x86-64-linux-gnu
  gcc-x86-64-linux-gnu libaom-dev libaom3 libapache2-mod-php8.3 libasan8 libatomic1 libbinutils libbrotli-dev libbz2-dev libc-dev-bin
  libc-devtools libc6-dev libcc1-0 libcrypt-dev libctf-nobfd0 libctf0 libdav1d-dev libdav1d7 libde265-0 libde265-dev libdeflate-dev
  libdeflate0 libexpat1-dev libfontconfig-dev libfontconfig1 libfreetype-dev libgcc-13-dev libgd3 libgomp1 libgprofng0 libheif-dev
  libheif-plugin-aomdec libheif-plugin-aomenc libheif-plugin-libde265 libheif1 libhwasan0 libisl23 libitm1 libjbig-dev libjbig0
  libjpeg-dev libjpeg-turbo8 libjpeg-turbo8-dev libjpeg8 libjpeg8-dev liblerc-dev liblerc4 liblsan0 liblzma-dev libmcrypt4 libmpc3
  libpkgconf3 libpng-dev libpng-tools libpthread-stubs0-dev libquadmath0 libsframe1 libsharpyuv-dev libsharpyuv0 libtiff-dev libtiff6
  libtiffxx6 libtsan2 libubsan1 libvpx-dev libvpx9 libwebp-dev libwebp7 libwebpdecoder3 libwebpdemux2 libwebpmux3 libx11-dev
  libx265-199 libx265-dev libxau-dev libxcb1-dev libxdmcp-dev libxpm-dev libxpm4 libzstd-dev linux-libc-dev m4 manpages-dev
```

Download and install Nagios:

- cd /tmp
- wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.4.6.tar.gz
- tar -zxvf nagios-4.4.6.tar.gz
- cd nagios-4.4.6/

```
ubuntu@ip-172-31-36-248:/tmp/nagios-4.4.11$ sudo apt-get install -y autoconf gcc libc6-dev make wget apache2 php libapache2-mod-php
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
autoconf is already the newest version (2.71-3).
gcc is already the newest version (4:13.2.0-7ubuntu1).
libc6-dev is already the newest version (2.39-0ubuntu8.3).
libc6-dev set to manually installed.
make is already the newest version (4.3-4.1build2).
wget is already the newest version (1.21.4-1ubuntu4.1).
apache2 is already the newest version (2.4.58-1ubuntu8.4).
php is already the newest version (2:8.3+93ubuntu2).
libapache2-mod-php is already the newest version (2:8.3+93ubuntu2).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

Compile and install Nagios:

- ./configure --with-command-group=nagioscmd
- sudo make all
- sudo make install-groups-users
- sudo make install
- sudo make install-daemoninit
- sudo make install-commandmode
- sudo make install-config
- sudo make install-webconf

```
ubuntu@ip-172-31-36-248:/tmp/nagios-4.4.11$ sudo ./configure --with-command-group=nagios
checking for a BSD-compatible install... /usr/bin/install -c
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking whether make sets $(MAKE)... yes
checking whether ln -s works... yes
checking for strip... /usr/bin/strip
checking how to run the C preprocessor... gcc -E
checking for grep that handles long lines and -e... /usr/bin/grep
checking for egrep... /usr/bin/grep -E
checking for ANSI C header files... yes
checking whether time.h and sys/time.h may both be included... yes
checking for sys/wait.h that is POSIX.1 compatible... yes
checking for sys/types.h... yes
checking for sys/stat.h... yes
checking for stdlib.h... yes
```

```
ubuntu@ip-172-31-36-248:/tmp/nagios-4.4.11$ sudo make all
cd ./base && make
make[1]: Entering directory '/tmp/nagios-4.4.11/base'
gcc -Wall -I.. -g -O2  -DHAVE_CONFIG_H -DNSCORE -c -o nagios.o nagios.c
nagios.c: In function ‘main’:
nagios.c:611:25: warning: ignoring return value of ‘asprintf’ declared wi
  611 |                         asprintf(&mac->x[MACRO_PROCESSSTARTTIME],
      |                         ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
nagios.c:841:25: warning: ignoring return value of ‘asprintf’ declared wi
  841 |                         asprintf(&mac->x[MACRO_EVENTSTARTTIME], "
      |                         ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
nagios.c: In function ‘nagios_core_worker’:
nagios.c:176:17: warning: ignoring return value of ‘read’ declared with a
  176 |                 read(sd, response + 3, sizeof(response) - 4);
      |                 ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
nagios.c: In function ‘test_path_access’:
nagios.c:122:17: warning: ignoring return value of ‘asprintf’ declared wi
  122 |                 asprintf(&path, "%s/%s", p, program);
      |                 ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
gcc -Wall -I.. -g -O2  -DHAVE_CONFIG_H -DNSCORE   -c -o broker.o broker.c
gcc -Wall -I.. -g -O2  -DHAVE_CONFIG_H -DNSCORE   -c -o nebmods.o nebmods
gcc -Wall -I.. -g -O2  -DHAVE_CONFIG_H -DNSCORE -c -o ../common/shared.o
```

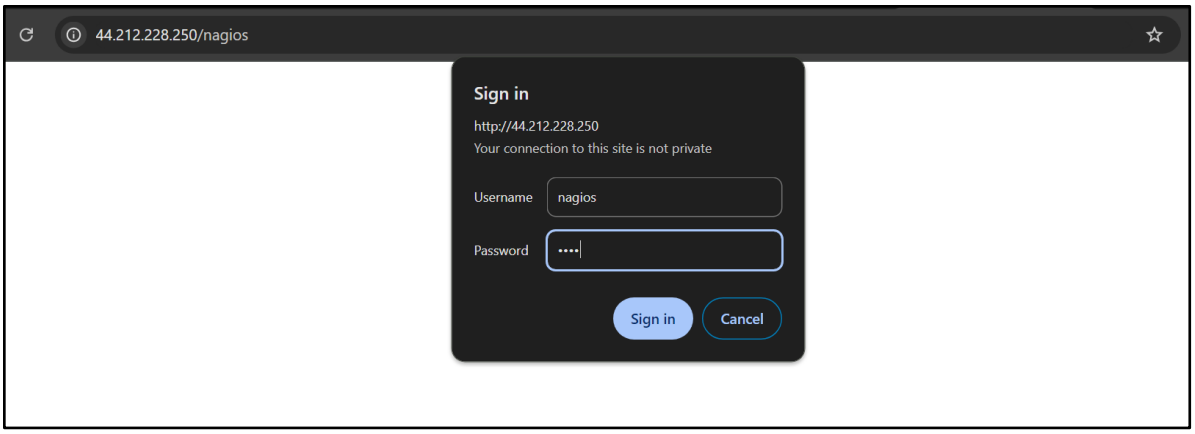
```
ubuntu@ip-172-31-36-248:/tmp/nagios-4.4.11$ sudo make install
cd ./base && make install
make[1]: Entering directory '/tmp/nagios-4.4.11/base'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/bin
/usr/bin/install -c -s -m 774 -o nagios -g nagios nagios /usr/local/nagios/bin
/usr/bin/install -c -s -m 774 -o nagios -g nagios nagiostats /usr/local/nagios/bin
make[1]: Leaving directory '/tmp/nagios-4.4.11/base'
cd ./cgi && make install
make[1]: Entering directory '/tmp/nagios-4.4.11/cgi'
make install-basic
make[2]: Entering directory '/tmp/nagios-4.4.11/cgi'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/sbin
for file in *.cgi; do \
        /usr/bin/install -c -s -m 775 -o nagios -g nagios $file /usr/local/nagios/
done
make[2]: Leaving directory '/tmp/nagios-4.4.11/cgi'
make[1]: Leaving directory '/tmp/nagios-4.4.11/cgi'
cd ./html && make install
make[1]: Entering directory '/tmp/nagios-4.4.11/html'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/media
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/styleshe
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/contexth
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/docs
```

```
ubuntu@ip-172-31-36-248:/tmp/nagios-4.4.11$ sudo make install-webconf
/usr/bin/install -c -m 644 sample-config/httpd.conf /etc/apache2/sites-available/nagios.conf
if [ 1 -eq 1 ]; then \
        ln -s /etc/apache2/sites-available/nagios.conf /etc/apache2/sites-enabled/nagios.conf; \
fi

*** Nagios/Apache conf file installed ***

ubuntu@ip-172-31-36-248:/tmp/nagios-4.4.11$ sudo make install-init
/usr/bin/install -c -m 755 -d -o root -g root /lib/systemd/system
/usr/bin/install -c -m 755 -o root -g root startup/default-service /lib/systemd/system/nagios.service
ubuntu@ip-172-31-36-248:/tmp/nagios-4.4.11$ sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
New password:
Re-type new password:
Adding password for user nagiosadmin
```

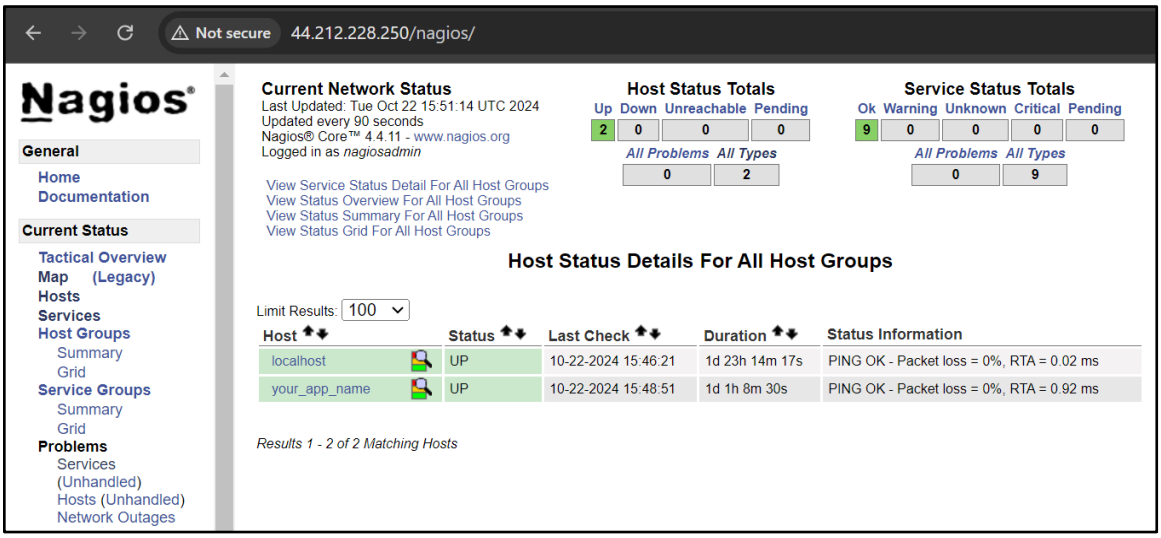
```
ubuntu@ip-172-31-36-248:~$ ubuntu@ip-172-31-36-248:~$ sudo systemctl restart nagios
Warning: The unit file, source configuration file or drop-ins of nagios.service changed on disk.
d units.
ubuntu@ip-172-31-36-248:~$ sudo systemctl daemon-reload
ubuntu@ip-172-31-36-248:~$ sudo systemctl restart nagios
ubuntu@ip-172-31-36-248:~$ sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

Nagios Core 4.4.11
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2023-04-14
License: GPL

Website: https://www.nagios.org
Reading configuration data...
   Read main config file okay...
   Read object config files okay...

Running pre-flight check on configuration data...

Checking objects...
        Checked 9 services.
        Checked 2 hosts.
        Checked 1 host groups.
        Checked 0 service groups.
        Checked 1 contacts.
        Checked 1 contact groups.
        Checked 24 commands.
        Checked 5 time periods.
        Checked 0 host escalations.
        Checked 0 service escalations.
```

Start Nagios and Apache:

- sudo systemctl enable apache2
- sudo systemctl enable nagios
- sudo systemctl start apache2
- sudo systemctl start nagios


44.212.228.250/nagios
Sign in
http://44.212.228.250
Your connection to this site is not private
Username
nagios
Password
Sign in
Cancel


Not secure
44.212.228.250/nagios/
Nagios
General
Home
Documentation
Current Status
Tactical Overview
Map (Legacy)
Hosts
Services
Host Groups
Summary
Grid
Service Groups
Summary
Grid
Problems
Services (Unhandled)
Hosts (Unhandled)
Network Outages
Quick Search:
Reports
Availability
Trends (Legacy)
Alerts
History
Nagios Core
Daemon running with PID 656
Nagios® Core™
Version 4.4.11
April 14, 2023
Check for updates
A new version of Nagios Core is available!
Visit nagios.org to download Nagios 4.5.6.
Get Started
Start monitoring your infrastructure
Change the look and feel of Nagios
Extend Nagios with hundreds of addons
Get support
Get training
Get certified
Quick Links
Nagios Library (tutorials and docs)
Nagios Labs (development blog)
Nagios Exchange (plugins and addons)
Nagios Support (tech support)
Nagios.com (company)
Nagios.org (project)
Tour


Not secure
44.212.228.250/nagios/
Nagios
General
Home
Documentation
Current Status
Tactical Overview
Map (Legacy)
Hosts
Services
Host Groups
Summary
Grid
Service Groups
Summary
Grid
Problems
Services (Unhandled)
Hosts (Unhandled)
Network Outages
Current Network Status
Last Updated: Tue Oct 22 15:51:14 UTC 2024
Updated every 90 seconds
Nagios® Core™ 4.4.11 - www.nagios.org
Logged in as nagiosadmin
View Service Status Detail For All Host Groups
View Status Overview For All Host Groups
View Status Summary For All Host Groups
View Status Grid For All Host Groups
Host Status Totals
Up Down Unreachable Pending
2 0 0 0
All Problems All Types
0 2
Service Status Totals
Ok Warning Unknown Critical Pending
9 0 0 0 0
All Problems All Types
0 9
Host Status Details For All Host Groups
Limit Results: 100
Host Status Last Check Duration Status Information
localhost UP 10-22-2024 15:46:21 1d 23h 14m 17s PING OK - Packet loss = 0%, RTA = 0.02 ms
your_app_name UP 10-22-2024 15:48:51 1d 1h 8m 30s PING OK - Packet loss = 0%, RTA = 0.92 ms
Results 1 - 2 of 2 Matching Hosts

C] Verify the pipeline by triggering a build and checking the monitoring status in Nagios.

Find or create a contact definition to specify who will receive notifications.

- sudo nano /usr/local/nagios/etc/objects/contacts.cfg

```
  GNU nano 7.2                                    /usr/local/nagios/etc/objects/contacts.cfg
# This contact definition inherits a lot of default values from the
# 'generic-contact' template which is defined elsewhere.

define contact {

    contact_name            nagiosadmin             ; Short name of user
    use                     generic-contact         ; Inherit default values from generic-contact template
    alias                   Nagios Admin
    service_notification_commands   notify-service-by-email           ; Full name of user
    email                   shravanistudy02@gmail.com  ; <<***** CHANGE THIS TO YOUR EMAIL ADDRESS ******
}



################################################################################
#
```

Edit Service and Host Configurations (Interval of monitoring after every 60 mins)

- sudo nano /usr/local/nagios/etc/objects/hosts.cfg

```
  GNU nano 7.2                                    /usr/local/nagios/etc/objects/hosts.cfg *
define host {
    use                     linux-server            ; Inherit default values from the linux-server template
    host_name               ec2-web-app-instance    ; The name of the host
    alias                   EC2 Web App Instance     ; A more descriptive name
    address                 44.212.228.250          ; Public IP address of your EC2 instance
    max_check_attempts      3                        ; Number of attempts before marking the host as down
    check_period            24x7                     ; Check period (you can set specific times if needed)
    notification_interval    60                      ; Time interval (in minutes) between notifications
    notification_period     24x7                     ; Period during which notifications can be sent
    notification_options    d,r                      ; Options for notifications: d = down, r = recovery
    contacts                nagiosadmin             ; Specify the contact for notifications
}
```

Add a New Service Definition)

- sudo nano /usr/local/nagios/etc/objects/services.cfg

```
# Check Swap Usage
define service {
    use                     generic-service
    host_name               your_server_name          ; Replace with your server name
    service_description     Swap Usage
    check_command           check_swap!20!10          ; Warning at 20% usage, Critical at 10% usage
    notifications_enabled    1                        ; Enable notifications
    contacts                nagiosadmin              ; Notify this contact on alerts
}

# Check HTTP
define service {
    use                     generic-service
    host_name               your_server_name          ; Replace with your server name
    service_description     HTTP
    check_command           check_http!http://your_web_app_url  ; Replace with your web app URL
    notifications_enabled    1                        ; Enable notifications
    contacts                nagiosadmin              ; Notify this contact on alerts
}

# Check SSH
```

(Services needed to be monitored Swap Uagae ,HTTP, SSH ,Ping)

```
# Check Swap Usage
define service {
    use                 generic-service
    host_name           your_server_name        ; Replace with your server name
    service_description     Swap Usage
    check_command           check_swap!20!10        ; Warning at 20% usage, Critical at 10%
usage
    notifications_enabled   1                   ; Enable notifications
    contacts            nagiosadmin             ; Notify this contact on alerts
}

# Check HTTP
define service {
    use                 generic-service
    host_name           your_server_name        ; Replace with your server name
    service_description     HTTP
    check_command           check_http!http://your_web_app_url  ; Replace with your web app URL
    notifications_enabled   1                   ; Enable notifications
    contacts            nagiosadmin             ; Notify this contact on alerts
}

# Check SSH
define service {
    use                 generic-service
    host_name           your_server_name        ; Replace with your server name
    service_description     SSH
    check_command           check_ssh               ; Default command to check SSH service
    notifications_enabled   1                   ; Enable notifications
    contacts            nagiosadmin             ; Notify this contact on alerts
}

# Check Ping
define service {
    use                 generic-service
    host_name           your_server_name        ; Replace with your server name
    service_description     PING
    check_command           check_ping!100.0%!500.0  ; Warning if > 100ms, Critical if > 500ms
    notifications_enabled   1                   ; Enable notifications
    contacts            nagiosadmin             ; Notify this contact on alerts
}
```

```
ubuntu@ip-172-31-36-248:~$ ls /usr/local/nagios/bin/
nagios  nagiostats
ubuntu@ip-172-31-36-248:~$ ^C
ubuntu@ip-172-31-36-248:~$ sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

Nagios Core 4.4.11
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2023-04-14
License: GPL

Website: https://www.nagios.org
Reading configuration data...
   Read main config file okay...
   Read object config files okay...

Running pre-flight check on configuration data...
```

Configure Email Notifications

Install Postfix

- sudo apt install postfix

```
ubuntu@ip-172-31-36-248:~$ sudo apt install postfix
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
postfix is already the newest version (3.8.6-1build2).
postfix set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
ubuntu@ip-172-31-36-248:~$
```

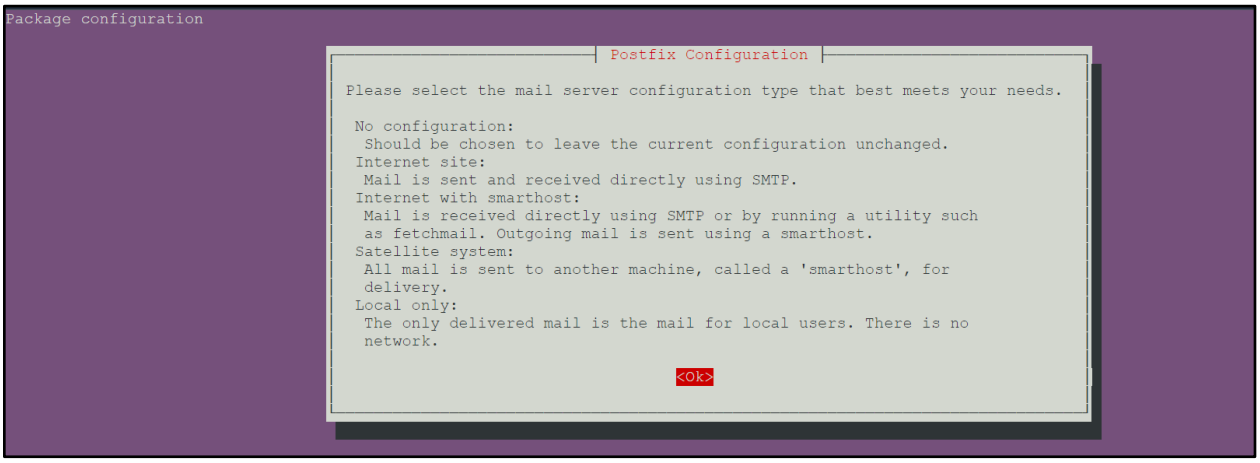
Set it to send mail directly using SMTP relay, then configure /etc/postfix/main.cf with the following:

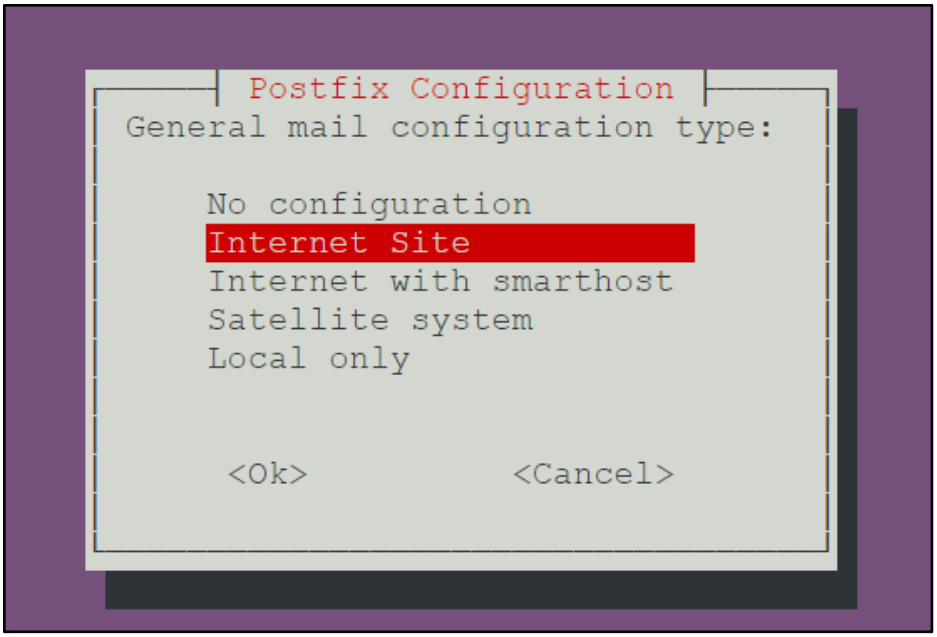
relayhost = [smtp.gmail.com]:587
smtp_sasl_auth_enable = yes
smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd
smtp_sasl_security_options = noanonymous
smtp_tls_CAfile = /etc/ssl/certs/ca-certificates.crt
smtp_use_tls = yes

```
ubuntu@ip-172-31-36-248:~$ sudo nano /usr/local/nagios/etc/contacts.cfg
ubuntu@ip-172-31-36-248:~$ sudo nano /usr/local/nagios/etc/objects/contacts.cfg
ubuntu@ip-172-31-36-248:~$ sudo systemctl restart nagios
ubuntu@ip-172-31-36-248:~$ sudo nano /etc/postfix/main.cf
ubuntu@ip-172-31-36-248:~$ sudo nano /etc/postfix/sasl_passwd
ubuntu@ip-172-31-36-248:~$ sudo chmod 600 /etc/postfix/sasl_passwd
ubuntu@ip-172-31-36-248:~$ sudo postmap /etc/postfix/sasl_passwd
ubuntu@ip-172-31-36-248:~$ sudo systemctl restart postfix
ubuntu@ip-172-31-36-248:~$ echo "Test email from Nagios" | mail -s "Test Subject" shravanistudy021@gmail.com
ubuntu@ip-172-31-36-248:~$ sudo tail -f /var/log/mail.log
2024-10-21T16:42:22.357496+00:00 ip-172-31-36-248 postfix/qmgr[6181]: 3528080522: from=<ubuntu@ip-172-31-36-24
2024-10-21T16:42:22.591843+00:00 ip-172-31-36-248 postfix/smtp[6186]: 3528080522: SASL authentication failed;
aid: 535-5.7.8 Username and Password not accepted. For more information, go to?535 5.7.8  https://support.goog
052e-460d3c62f4dsm19593181cf.28 - gsmtp
2024-10-21T16:42:22.592511+00:00 ip-172-31-36-248 postfix/smtp[6186]: connect to smtp.gmail.com[2607:f8b0:4004
```

sudo postmap /etc/postfix/sasl_passwd
sudo systemctl restart postfix

```
ubuntu@ip-172-31-36-248:~$ sudo postmap /etc/postfix/sasl_passwd
sudo systemctl restart postfix
ubuntu@ip-172-31-36-248:~$ sudo systemctl restart postfix
ubuntu@ip-172-31-36-248:~$ sudo postmap /etc/postfix/sasl_passwd
ubuntu@ip-172-31-36-248:~$ sudo nano /etc/postfix/sasl_passwd
ubuntu@ip-172-31-36-248:~$ sudo postmap /etc/postfix/sasl_passwd
ubuntu@ip-172-31-36-248:~$ sudo systemctl restart postfix
ubuntu@ip-172-31-36-248:~$ echo "Test email from Nagios-This is Shravani" | mail -s "Test Subject" shravanistudy02@gmail.com
ubuntu@ip-172-31-36-248:~$ sudo nano /usr/local/nagios/etc/objects/contacts.cfg
ubuntu@ip-172-31-36-248:~$ sudo systemctl restart nagios
sudo systemctl restart postfix
ubuntu@ip-172-31-36-248:~$ sudo systemctl restart nagios
ubuntu@ip-172-31-36-248:~$ sudo systemctl restart postfix
ubuntu@ip-172-31-36-248:~$ sudo nano /usr/local/nagios/etc/nagios.cfg
ubuntu@ip-172-31-36-248:~$ sudo nano /usr/local/nagios/etc/objects/hosts.cfg
ubuntu@ip-172-31-36-248:~$ sudo tail -f /var/log/mail.log
2024-10-21T17:44:18.371232+00:00 ip-172-31-36-248 postfix/qmgr[8227]: 586CA803F2: from=<nagios@ip-172-31-36-248>, size=707, nr
2024-10-21T17:44:18.982527+00:00 ip-172-31-36-248 postfix/smtp[8475]: 586CA803F2: to=<shravanistudy02@gmail.com>, relay=smtp.g
]:587, delay=0.63, delays=0.02/0.01/0.2/0.4, dsn=2.0.0, status=sent (250 2.0.0 OK  1729532658 6a1803df08f44-6ce008acf3bsm1973(
2024-10-21T17:44:18.982851+00:00 ip-172-31-36-248 postfix/qmgr[8227]: 586CA803F2: removed
```

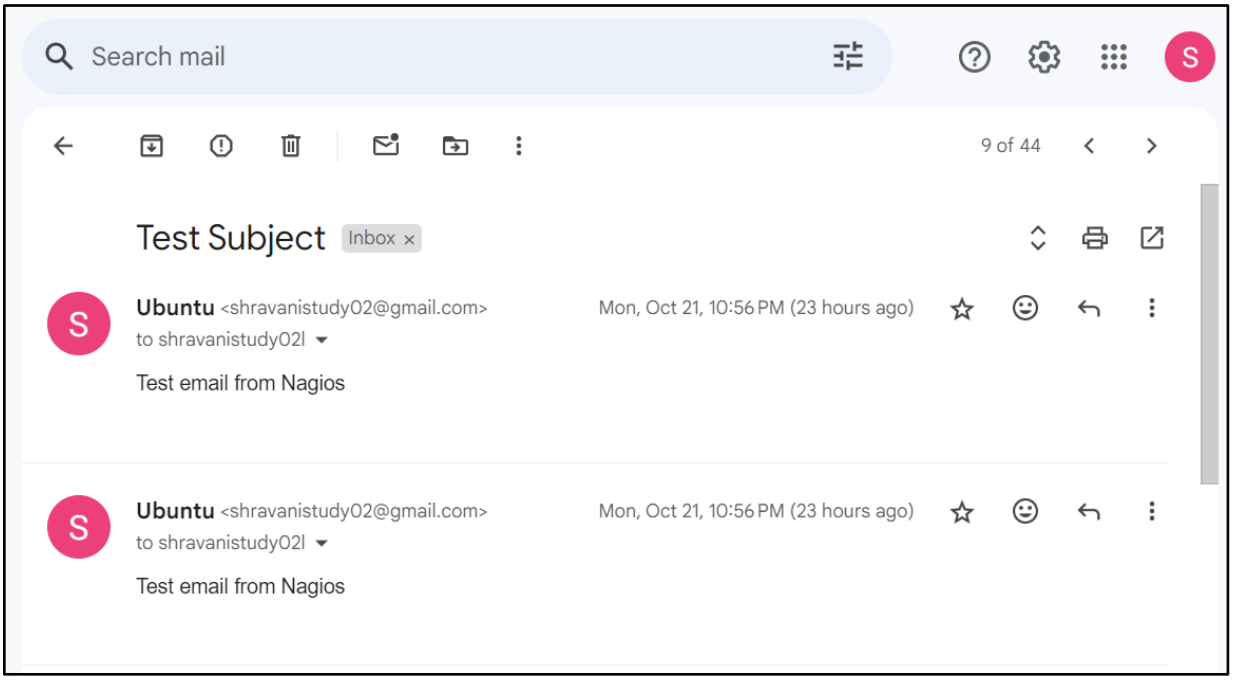




Test Email Notifications

- Trigger an alert and check the email sent to `shravani0212@gmail.com`.

```
ubuntu@ip-172-31-36-248:~$ ^C
ubuntu@ip-172-31-36-248:~$ sudo postmap /etc/postfix/sasl_passwd
sudo systemctl restart postfix
ubuntu@ip-172-31-36-248:~$ sudo systemctl restart postfix
ubuntu@ip-172-31-36-248:~$ sudo postmap /etc/postfix/sasl_passwd
ubuntu@ip-172-31-36-248:~$ sudo nano /etc/postfix/sasl_passwd
ubuntu@ip-172-31-36-248:~$ sudo postmap /etc/postfix/sasl_passwd
ubuntu@ip-172-31-36-248:~$ sudo systemctl restart postfix
ubuntu@ip-172-31-36-248:~$ echo "Test email from Nagios-This is Shravani" | mail -s "Test Subject" shravanistudy02@gmail.com
ubuntu@ip-172-31-36-248:~$
```



Website Monitoring Interval:

- Nagios will monitor the website every 60 minutes to check its status.

Critical Service Notification:

- If any service becomes critical during the monitoring, Nagios will immediately send a notification email to the configured contact group.

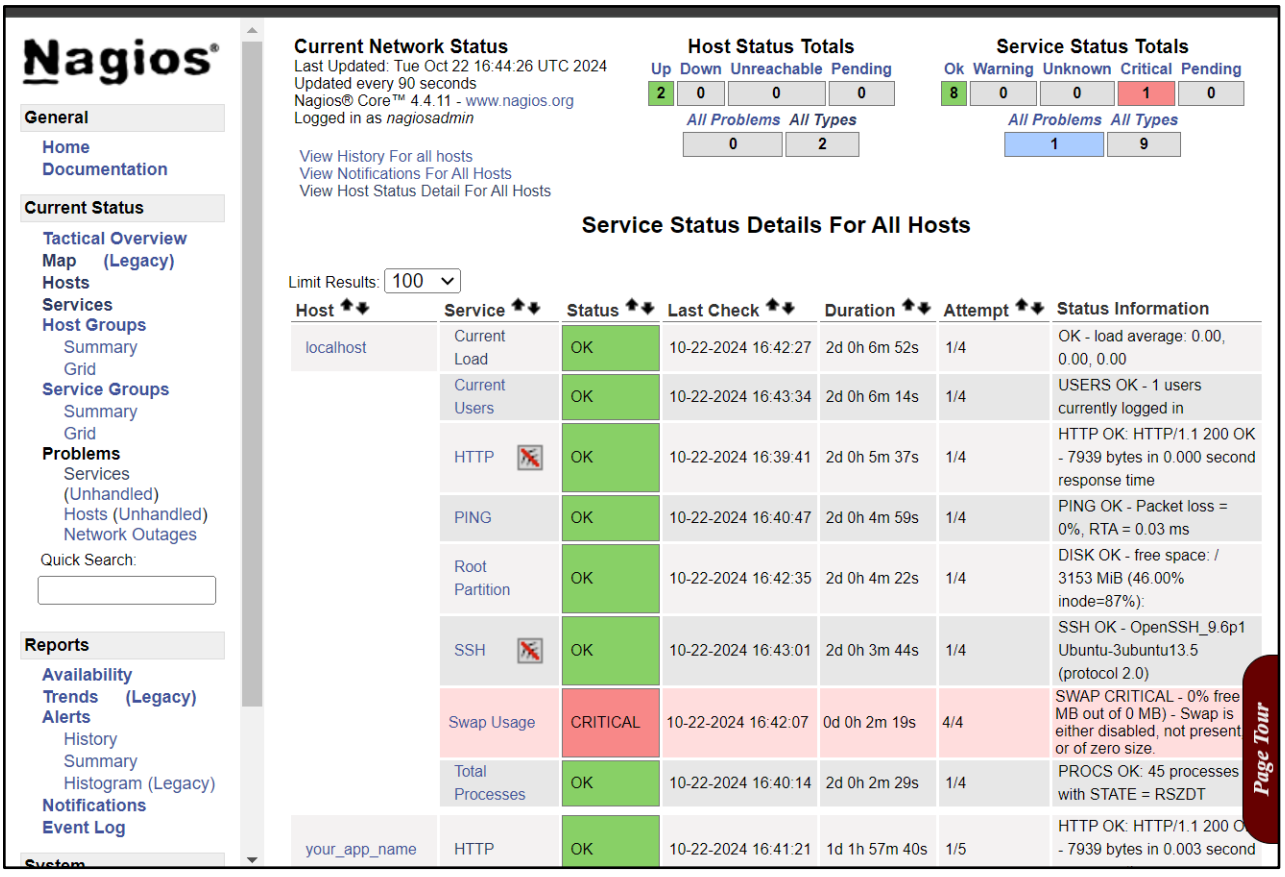
Recovery Notification:

- When the critical service is restored, Nagios will automatically send a recovery email to notify that the service is back to normal.

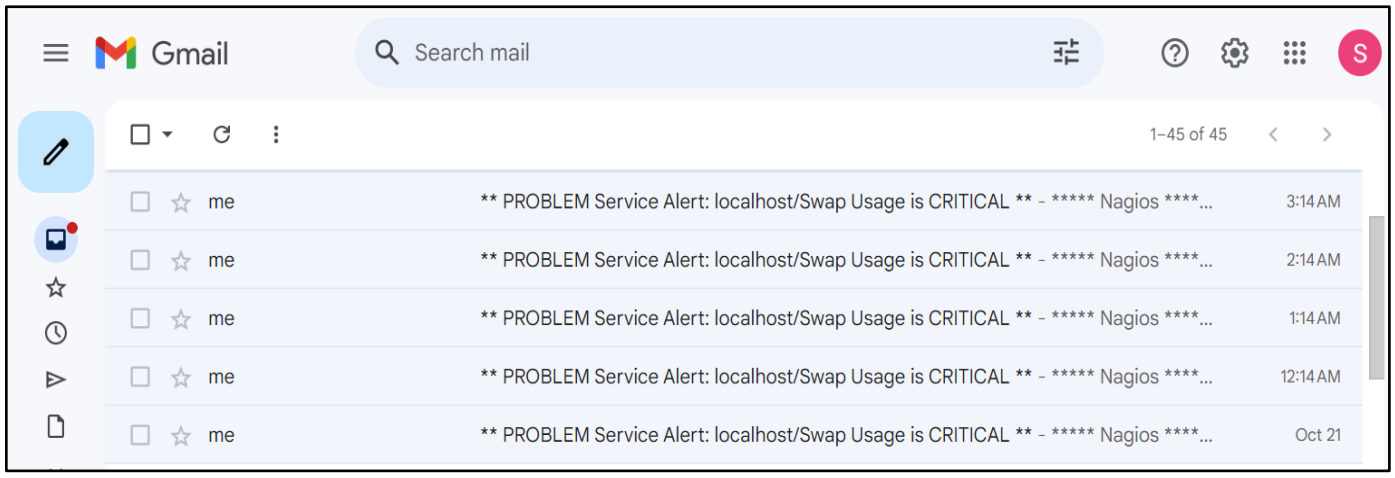
Service Details:

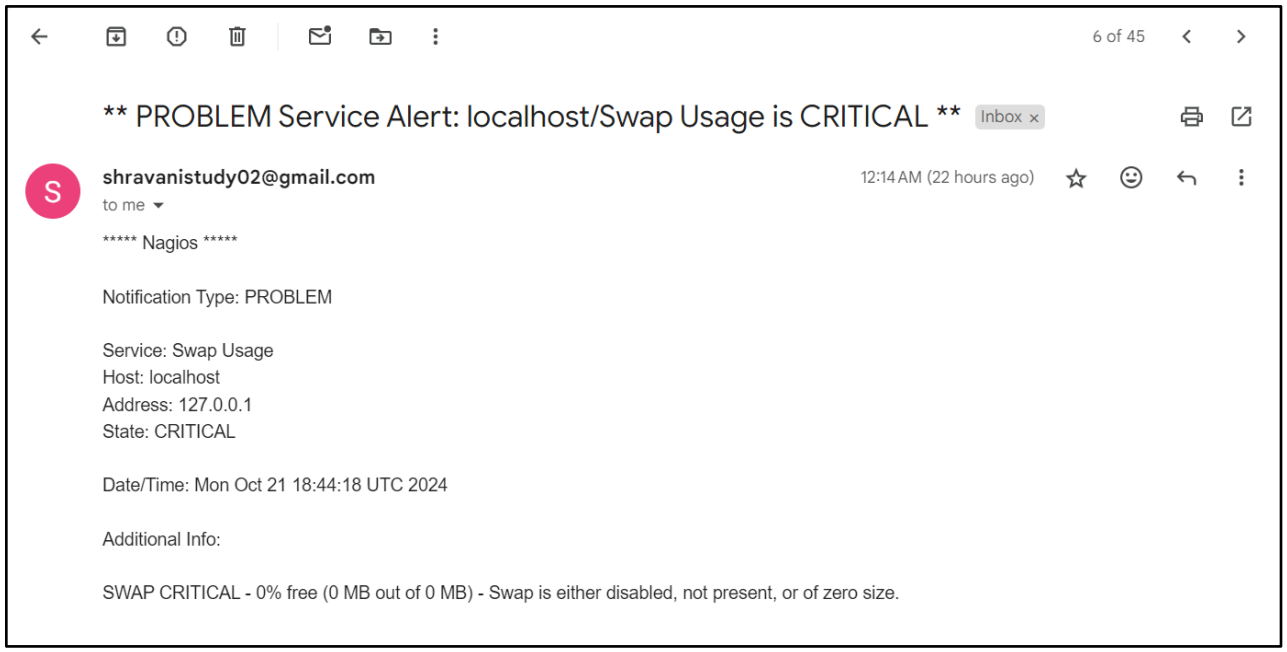
- The service being monitored is the website, with the check performed via the HTTP protocol.

Swap Usage is Critical



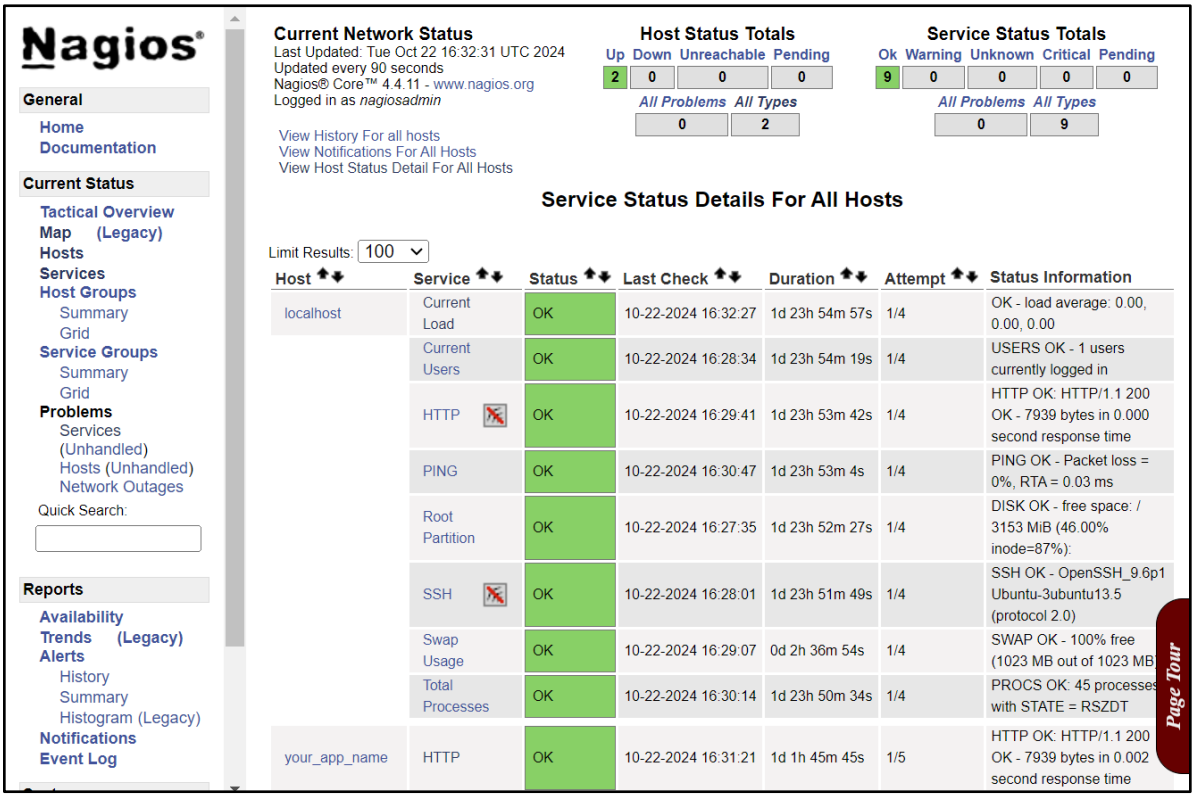
Nagios will monitor the service after every 60 minutes and notify via mail if service is still critical



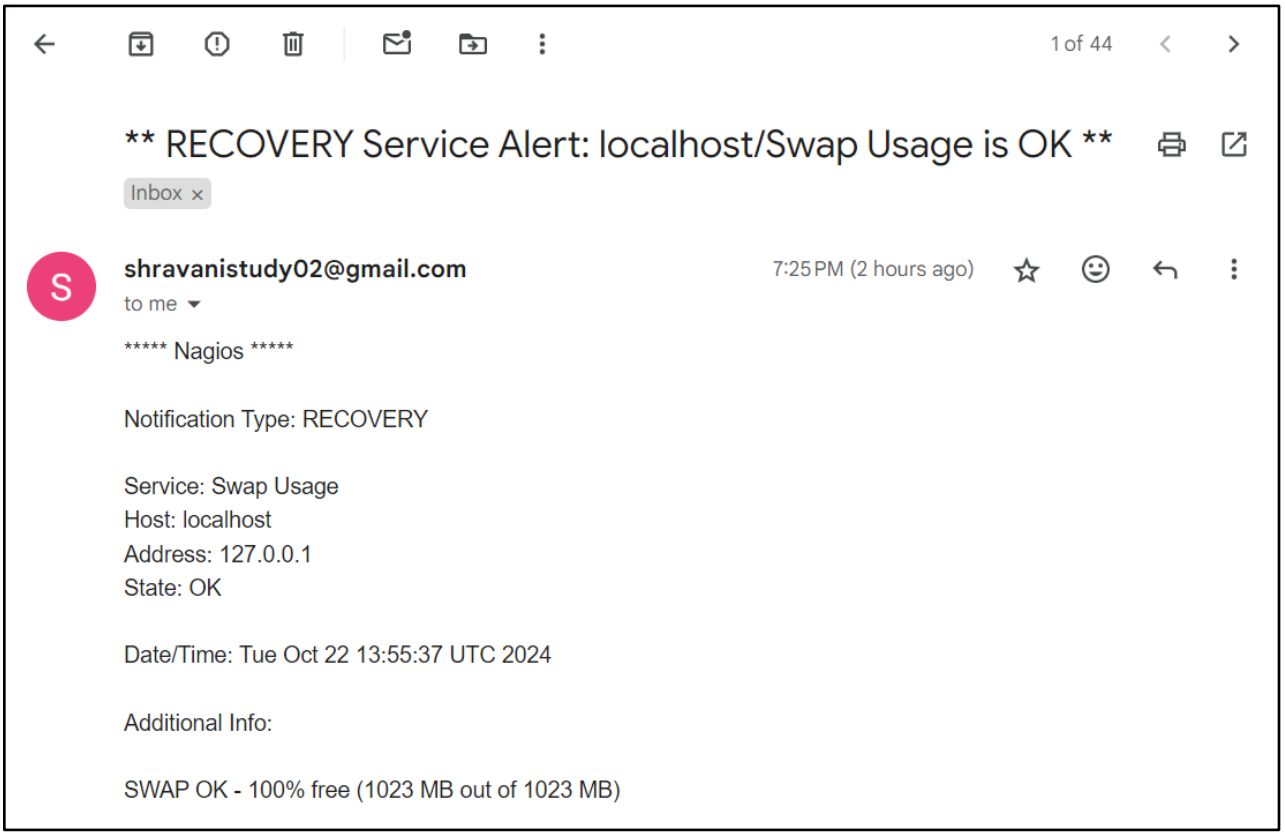


Fixing the Swap Usage Service

```
ubuntu@ip-172-31-36-248:~$ sudo swapon --show
ubuntu@ip-172-31-36-248:~$ sudo fallocate -l 1G /swapfile
ubuntu@ip-172-31-36-248:~$ sudo chmod 600 /swapfile
ubuntu@ip-172-31-36-248:~$ sudo mkswap /swapfile
Setting up swapspace version 1, size = 1024 MiB (1073737728 bytes)
no label, UUID=08e2608c-00d2-4ae5-99cd-2a79115486c7
ubuntu@ip-172-31-36-248:~$ echo '/swapfile none swap sw 0 0' | sudo tee -a /etc/fstab
/swapfile none swap sw 0 0
ubuntu@ip-172-31-36-248:~$ free -h
               total        used        free      shared  buff/cache   available
Mem:           957Mi       375Mi       286Mi       3.4Mi       451Mi       581Mi
Swap:             0B          0B          0B
ubuntu@ip-172-31-36-248:~$ sudo swapon --show
ubuntu@ip-172-31-36-248:~$
```



Recovery Mail - Swap Usage OK



CONCLUSION

In conclusion, the **Automated Deployment with Monitoring** case study has effectively demonstrated the integration of CI/CD practices using Jenkins and real-time monitoring through Nagios for a simple web application. This implementation has significantly streamlined the deployment process, reducing manual intervention and minimizing the risk of errors. By automating deployments and leveraging Nagios for proactive monitoring, the system ensures high availability and reliability, promptly alerting the development team to any critical issues. This approach not only enhances the overall efficiency of application management but also sets a strong foundation for future enhancements, illustrating the vital role of automation and monitoring in modern software development practices.