# **Day 21 Task**

## **Project Overview:**

The goal of this capstone project is to combine shell scripting with system monitoring and log management practices. You will create a set of automated tools using shell scripts to manage logs, monitor system performance using Prometheus and Node Exporter, and generate insights using PromQL queries. The project will require a systematic approach, covering scripting fundamentals, log management, and monitoring setup.

## **Project Deliverables:**

### 1. Shell Scripts for Basic Operations:

**Task:** Write shell scripts to perform basic system operations, such as checking disk usage, memory usage, and CPU load.

**Commands:**

* metrics.sh

#!/bin/bash

LOGFILE="system\_metrics.log"

log\_message() {

echo "$1" | tee -a "$LOGFILE"

}

check\_disk\_usage() {

log\_message "Disk Usage:"

df -h | tee -a "$LOGFILE"

}

check\_memory\_usage() {

log\_message "Memory Usage:"

free -h | tee -a "$LOGFILE"

}

check\_cpu\_load() {

log\_message "CPU Load:"

top -bn1 | grep "Cpu(s)" | tee -a "$LOGFILE"

}

check\_disk\_usage

check\_memory\_usage

check\_cpu\_load

# Error handling

if [ $? -ne 0 ]; then

log\_message "Error occurred while collecting system metrics."

exit 1

fi

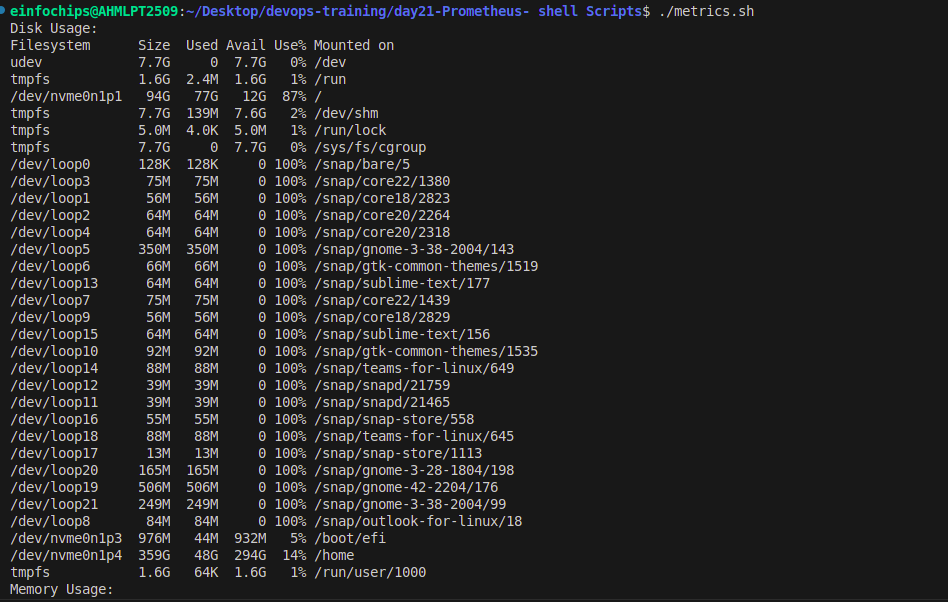
log\_message "System metrics collection completed successfully."

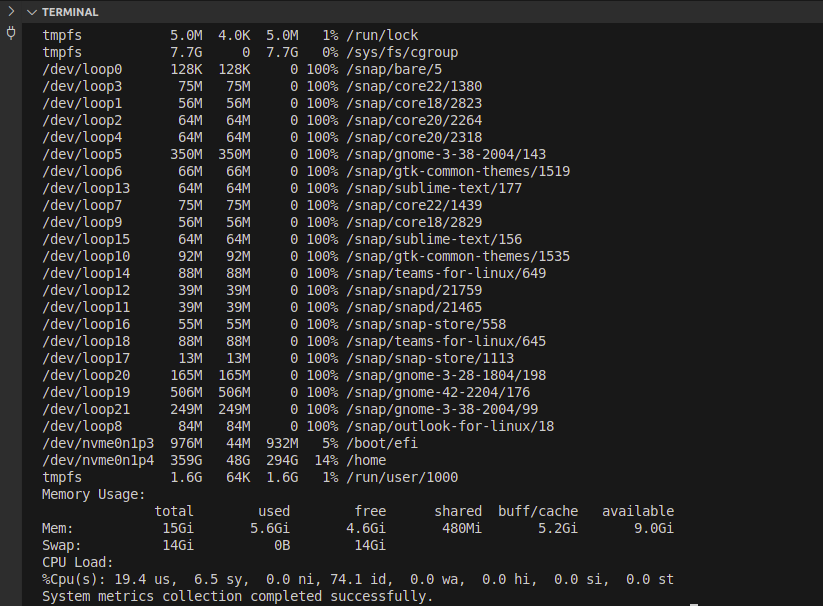
make script executable:

chmod +x metrics.sh

run script:

./metrics.sh





### **2. Log Management Script**

**Task:** Create a script to automate log management tasks.

**Script:**

* log\_management.sh: Automates log rotation and archiving.

#!/bin/bash

LOGFILE="log\_report.txt"

check\_log() {

echo "logs are shown below :-"

cat /var/log/syslog | tail -n 4

echo

}

if [ -n "$LOGFILE" ]; then

{

check\_log

} > "$LOGFILE"

echo "Report saved to $LOGFILE"

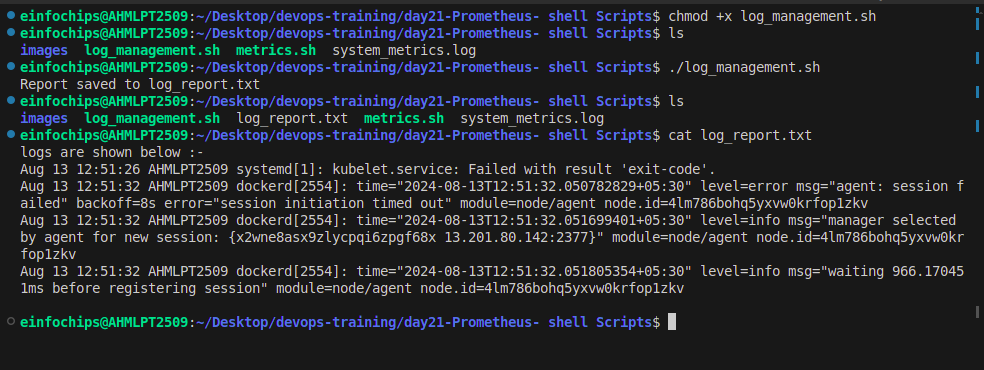
fi

make script executable:

chmod +x log\_management.sh

run script:

./ log\_management.sh



### **3. Advanced Shell Scripting - Loops, Conditions, Functions, and Error Handling**

**Scripts:**

* metrics\_advance.sh: Combines disk usage, memory usage, and CPU load checks with advanced scripting techniques.
* metrics\_advance.sh

#!/bin/bash

LOGFILE="metrics\_advance.log"

TMPFILE="/tmp/system\_metrics\_temp.log"

# Ensure the log file exists and is writable

if [ ! -w "$(dirname "$LOGFILE")" ]; then

echo "Log directory is not writable: $(dirname "$LOGFILE")" >&2

exit 1

fi

log\_message() {

local message="$1"

local timestamp

timestamp=$(date +"%Y-%m-%d %H:%M:%S")

echo "$timestamp - $message" | tee -a "$LOGFILE"

}

check\_disk\_usage() {

log\_message "Checking disk usage..."

df -h > "$TMPFILE" 2>> "$LOGFILE"

if [ $? -eq 0 ]; then

log\_message "Disk Usage:"

cat "$TMPFILE" | tee -a "$LOGFILE"

else

log\_message "Error checking disk usage."

fi

}

check\_memory\_usage() {

log\_message "Checking memory usage..."

free -h > "$TMPFILE" 2>> "$LOGFILE"

if [ $? -eq 0 ]; then

log\_message "Memory Usage:"

cat "$TMPFILE" | tee -a "$LOGFILE"

else

log\_message "Error checking memory usage."

fi

}

check\_cpu\_load() {

log\_message "Checking CPU load..."

top -bn1 | grep "Cpu(s)" > "$TMPFILE" 2>> "$LOGFILE"

if [ $? -eq 0 ]; then

log\_message "CPU Load:"

cat "$TMPFILE" | tee -a "$LOGFILE"

else

log\_message "Error checking CPU load."

fi

}

{

log\_message "Starting system metrics collection."

check\_disk\_usage

check\_memory\_usage

check\_cpu\_load

log\_message "System metrics collection completed successfully."

} || {

log\_message "An error occurred during the system metrics collection."

exit 1

}

# Clean up temporary file

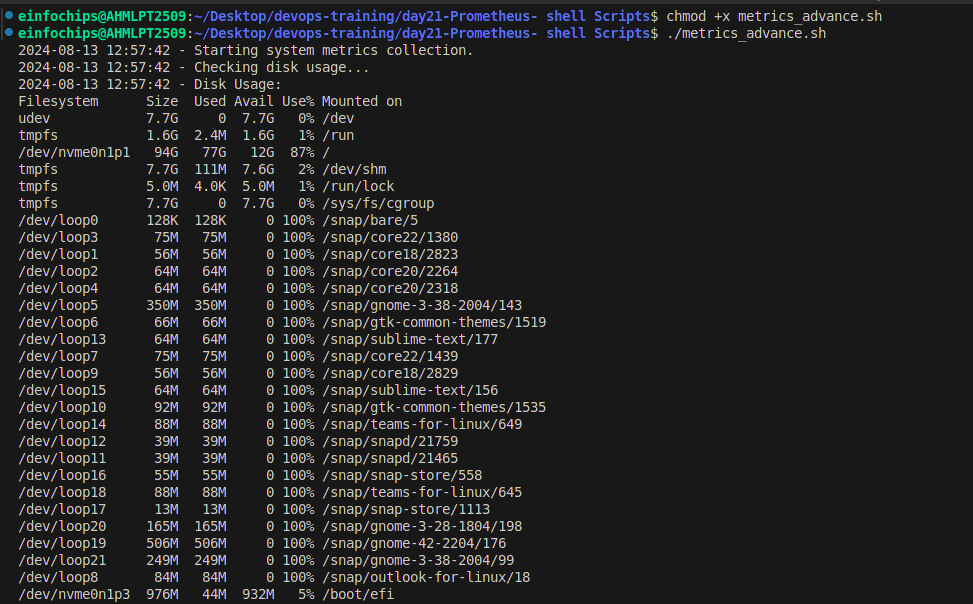
rm -f "$TMPFILE"

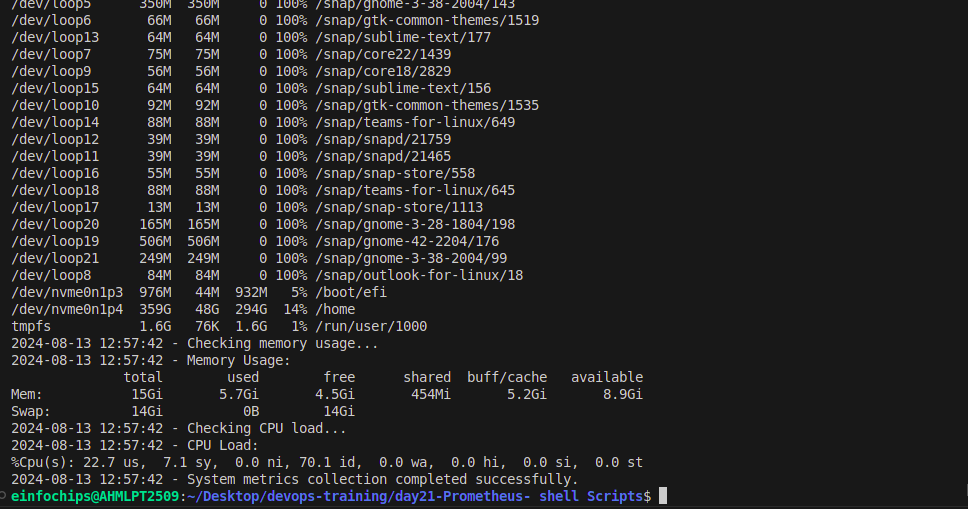
make script executable:

chmod +x metrics\_advance.sh

run script:

./metrics\_advance.sh





### **4. Log Checking and Troubleshooting**

**Task:** Develop a script to analyze system and application logs.

**Script:**

* log\_troubleshooting.sh: Analyzes logs for common issues and provides troubleshooting steps.
* log\_troubleshooting.sh

#!/bin/bash

LOGFILES=("/var/log/syslog" "/var/log/auth.log")

REPORT\_FILE="log\_troubleshooting\_report.log"

# Function to check for common issues

check\_logs() {

for logfile in "${LOGFILES[@]}"; do

if [ -f "$logfile" ]; then

echo "Checking $logfile" | tee -a "$REPORT\_FILE"

grep -i "out of memory\|failed\|error" "$logfile" | tee -a "$REPORT\_FILE"

else

echo "$logfile does not exist" | tee -a "$REPORT\_FILE"

fi

done

}

# Function to provide troubleshooting steps

troubleshoot() {

echo "Troubleshooting steps:" | tee -a "$REPORT\_FILE"

echo "1. Check system memory and processes using 'top' or 'free'." | tee -a "$REPORT\_FILE"

echo "2. Verify service status using 'systemctl status <service>'." | tee -a "$REPORT\_FILE"

echo "3. Inspect configurations or restart services as needed." | tee -a "$REPORT\_FILE"

}

check\_logs

troubleshoot

# Error handling

if [ $? -ne 0 ]; then

echo "Log checking or troubleshooting failed" | tee -a "$REPORT\_FILE"

exit 1

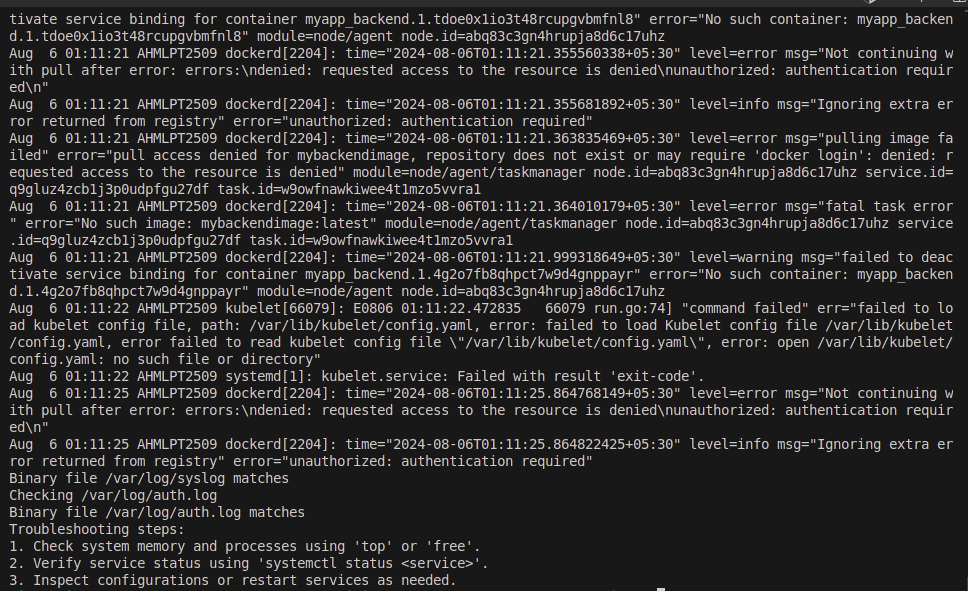
fi

make script executable:

chmod +x log\_troubleshooting.sh

run script:

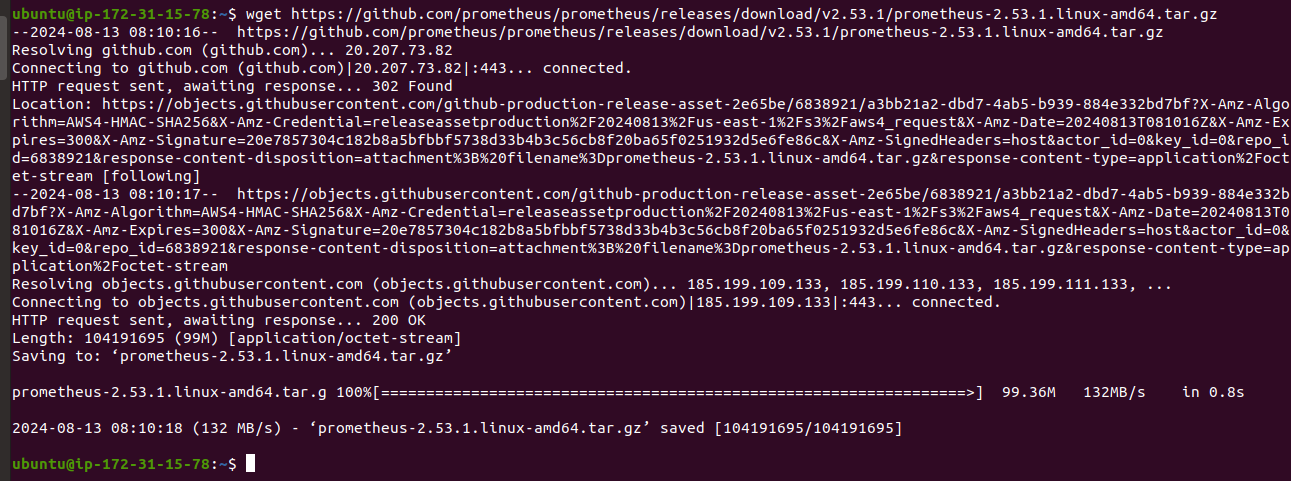
./log\_troubleshooting.sh



### **5. Installation and Setup of Prometheus and Node Exporter**

**Task:** Install and configure Prometheus and Node Exporter.

wget https://github.com/prometheus/prometheus/releases/download/v2.53.1/prometheus-2.53.1.linux-amd64.tar.gz



* unzip prometheus

tar -xvf prometheus-2.53.1.linux-amd64.tar.gz

cd prometheus-2.53.1.linux-amd64/

Configure Prometheus: Update the prometheus.yml configuration file to include Node Exporter as a scrape target.

scrape\_configs:

- job\_name: 'node\_exporter'

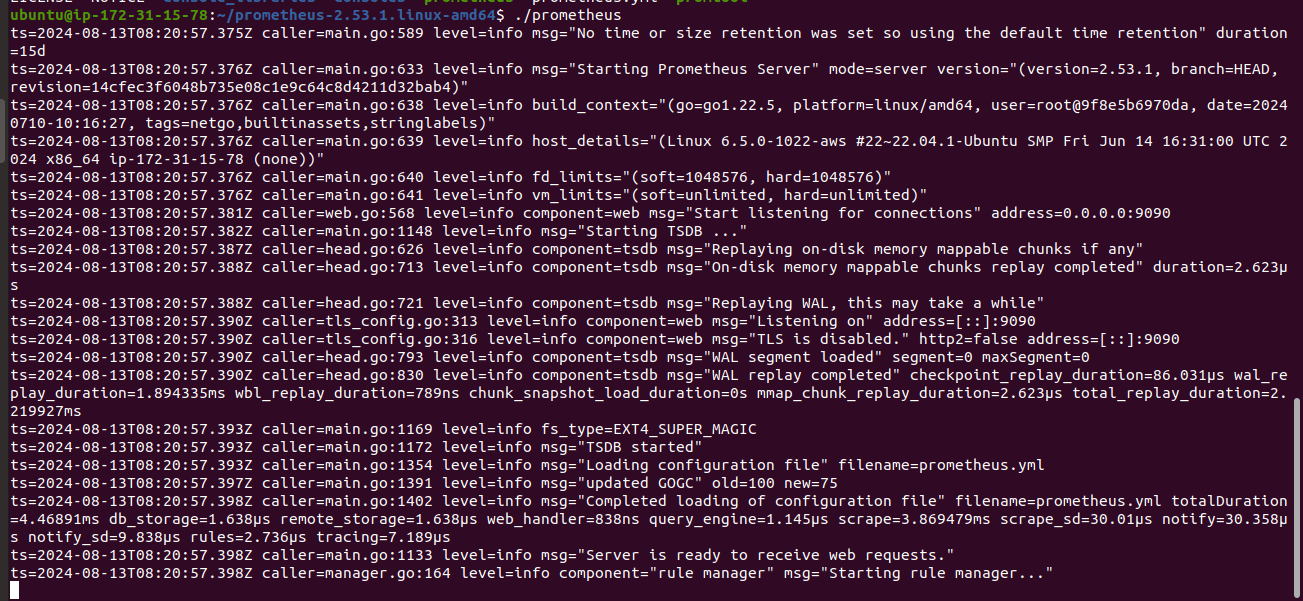
static\_configs:

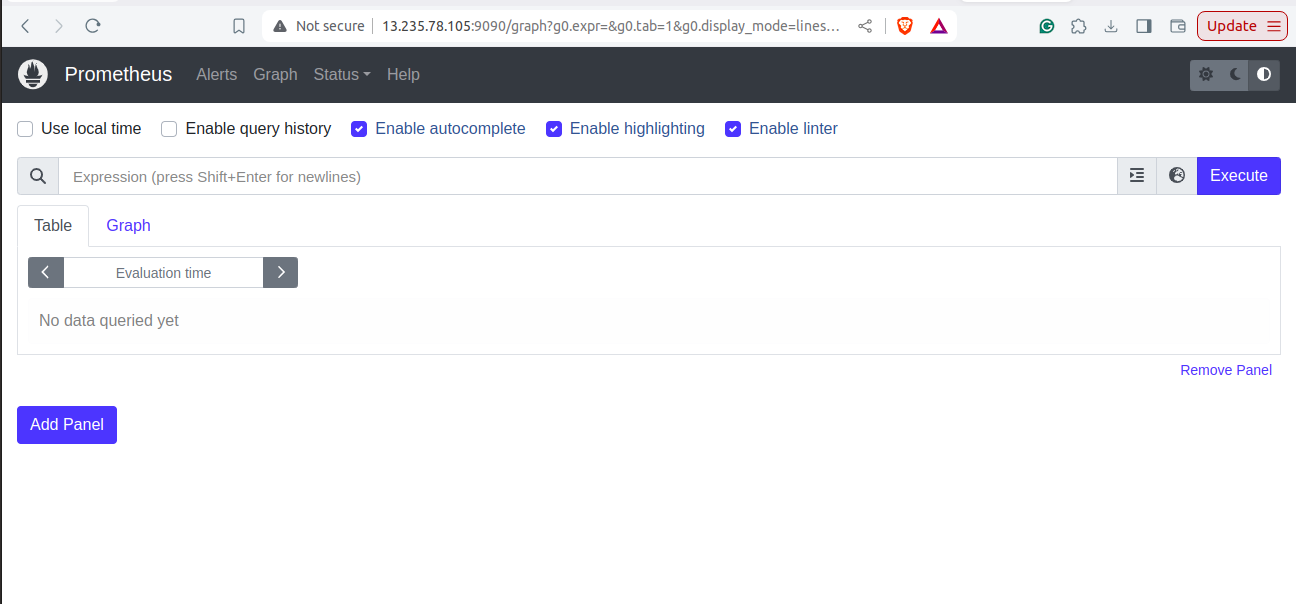
- targets: ['localhost:9100']



* start prometheus

./prometheus





Installing Node Exporter

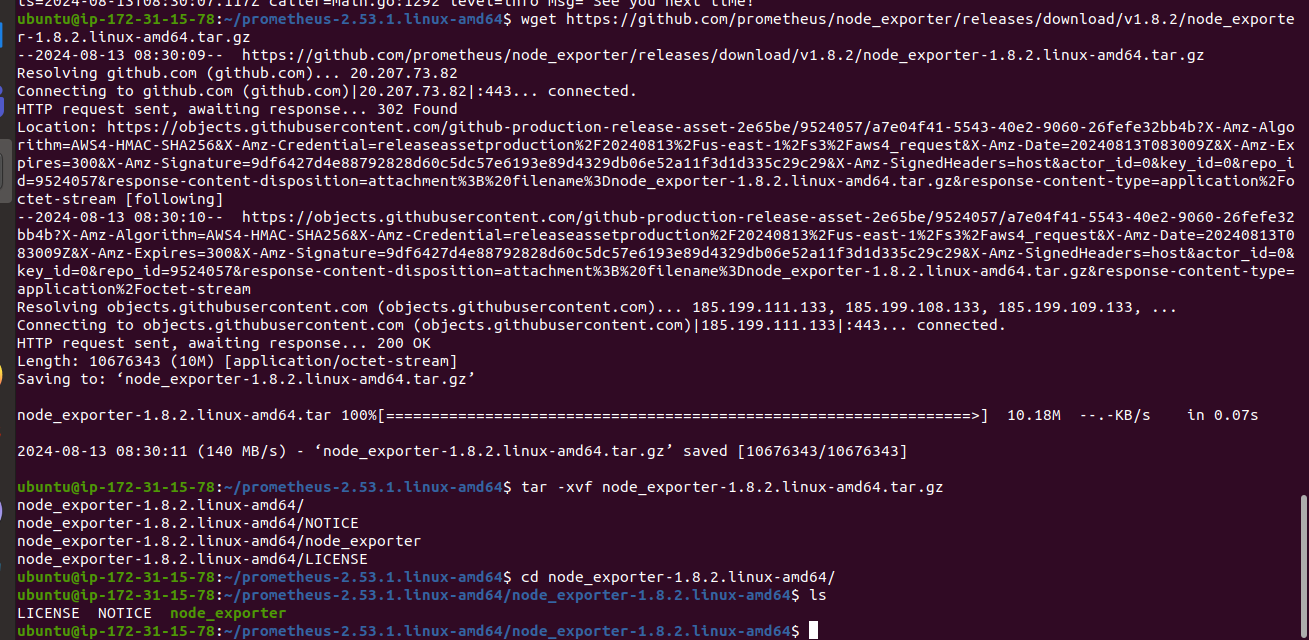
* download node\_exporter

wget https://github.com/prometheus/node\_exporter/releases/download/v1.8.2/node\_exporter-1.8.2.linux-amd64.tar.gz

* unzip node\_exporter

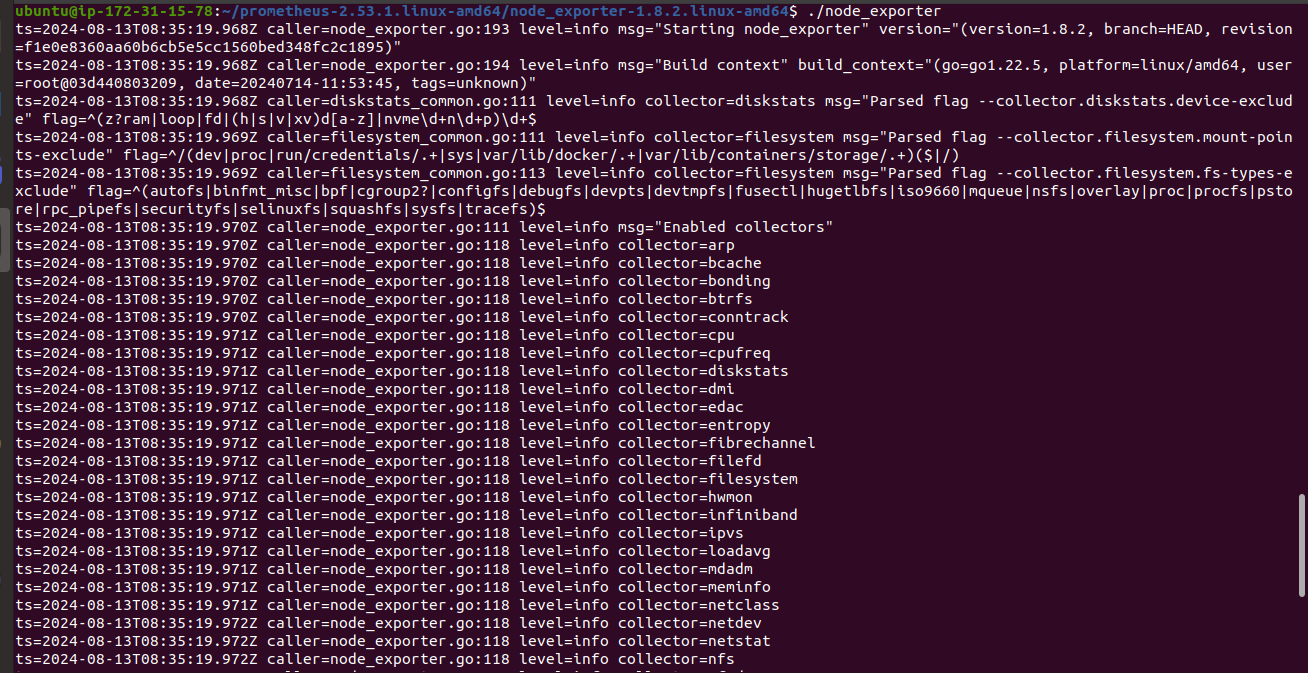
tar -xvf node\_exporter-1.8.2.linux-amd64.tar.gz

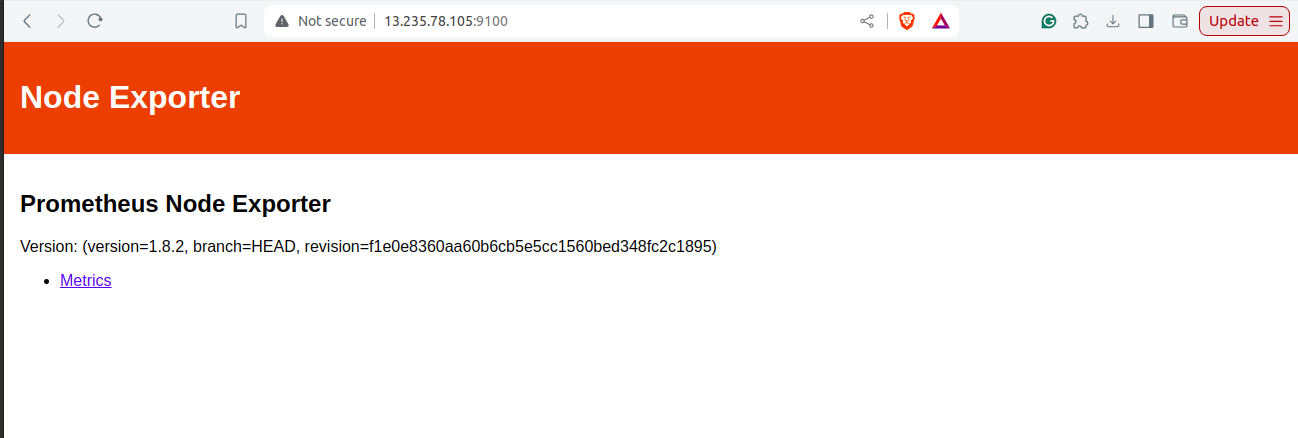
cd node\_exporter-1.8.2.linux-amd64/



* start node\_exporter

./node\_exporter





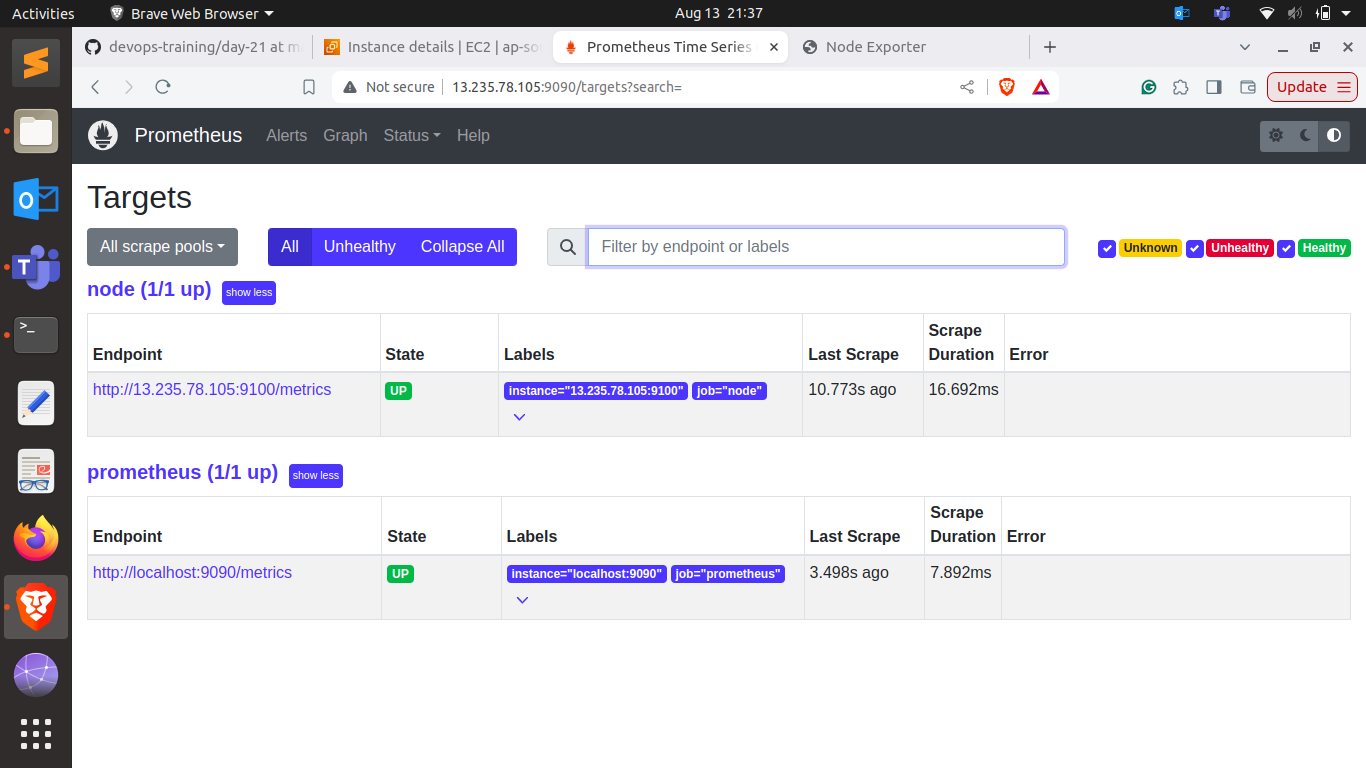
Updating config file and adding new job in scrape\_config

- job\_name: "node"

static\_configs:

- targets: ["<ip\_addr>:9100"]

Prometheus scraping metrics from Node Exporter



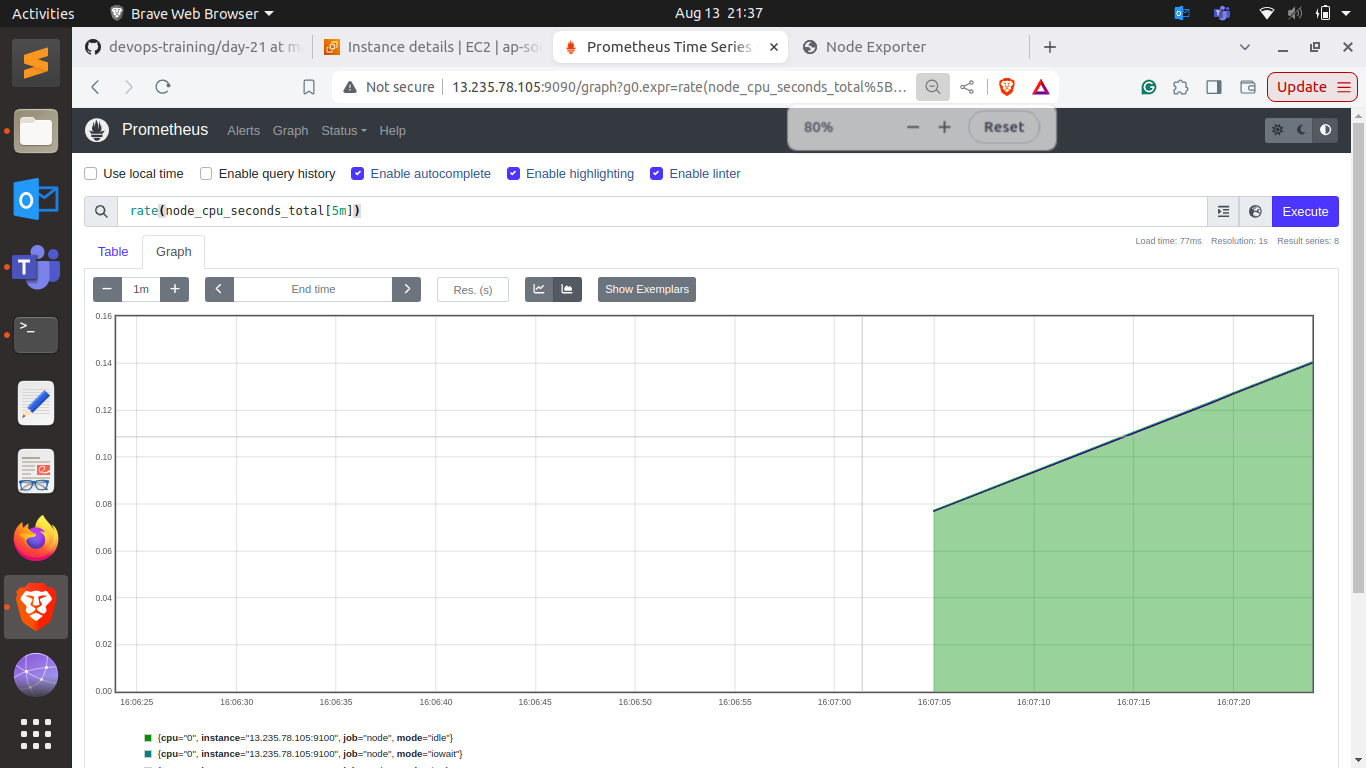
### **6. Prometheus Query Language (PromQL) Basic Queries:**

**Task:** Create a series of PromQL queries to monitor system performance, such as CPU usage, memory usage, and disk I/O.

**Commands:**

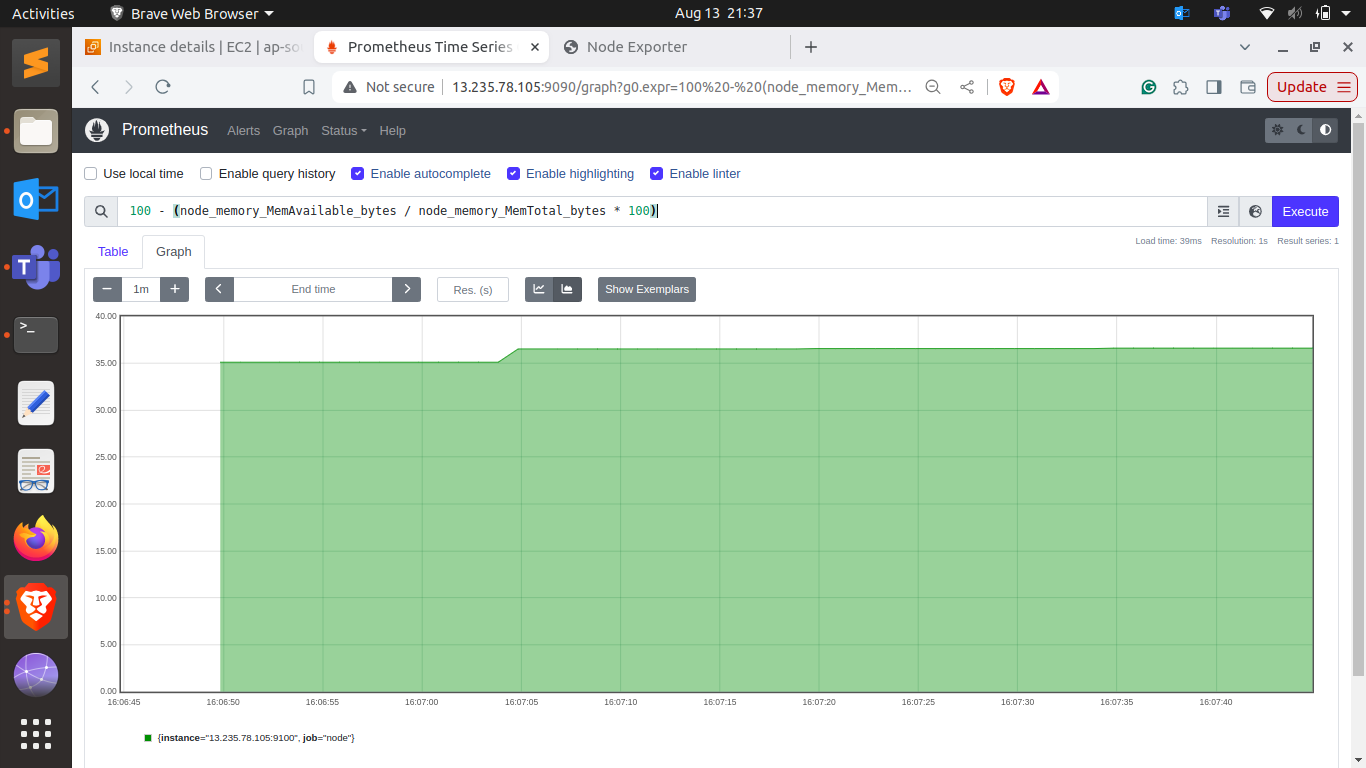
* monitor CPU usage

rate(node\_cpu\_seconds\_total[5m])



* monitor memory usage

100 - (node\_memory\_MemAvailable\_bytes / node\_memory\_MemTotal\_bytes \* 100)



* monitor disk I/O