Monitoring Project

🚀 I completed Project of AWS Cloud On, A comprehensive monitoring solution using Prometheus and various exporters to ensure the reliability and performance of a web application hosted on AWS EC2 instances.

👨‍💻In the Project we basically hosted web application on AWS EC2 instances. This setup includes Node Exporter for hardware and OS metrics, Black box Exporter for probing endpoints, and Alert manager for handling alerts. Gmail integration was also configured to receive notifications for critical alerts.

✨Get ready to enhance your AWS skills and boost your confidence in cloud technology☁Special Thanks to @

Prerequisites -

Before we start, make sure you have:-

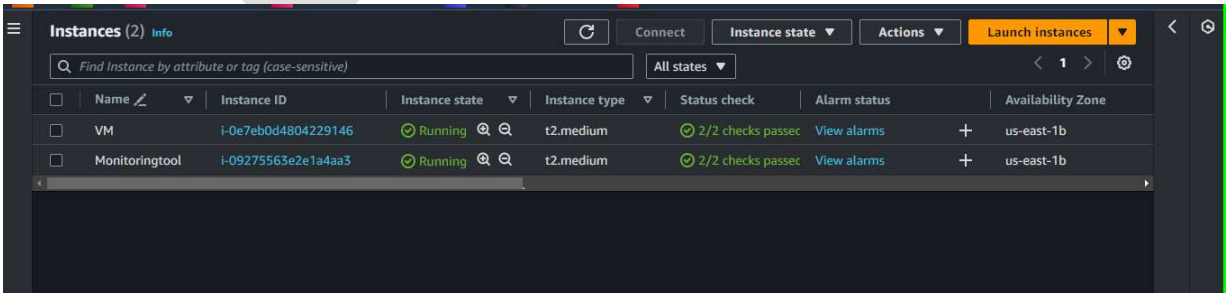
An AWS account

Basic knowledge of AWS services (EC2, Prometheus, Node Exporter, Black box Exporter, Alert manager)

AWS CLI or AWS Management Console access (Git / GitHub)

Step 1: Set Up AWS Environment -

- **Step 1.1** Provision EC2 Instances : Instance Type: t2.medium, VCPUs 2 , Memory 20 GB , Network Performance Moderate , Ubuntu Server 24.04 LTS , S.G - Prometheus 9090 , Alert manager 9093 , Black box Exporter 9115 , Node Exporter 9100 , Email transmissions 587.



[Fig 1: Successful Creation of EC2 [VM/ Monitoring Tool Instance]

Step 1.2 - Install and Configure Node Exporter and Deploy Web application on Instance 1...

1. Install Node Exporter: Need to download different package tool we used we can get it from this website **prometheus.io**,

   Wget https://github.com/prometheus/node_exporter/releases/download/v1.8.1/node_exporter-1.8.1.linux-amd64.tar.gz

2. **Install.Nginx**: sudo apt update / sudo apt install nginx
3. Deploy Web Application(Java based Application) :

Install Java using this command: sudo `apt install openjdk-17-jre-headless`

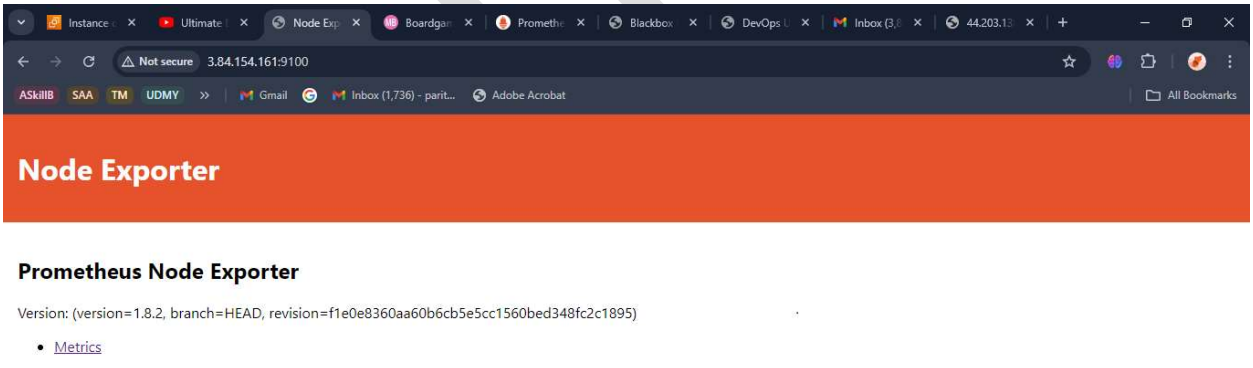
Install Maven: sudo apt install maven –y

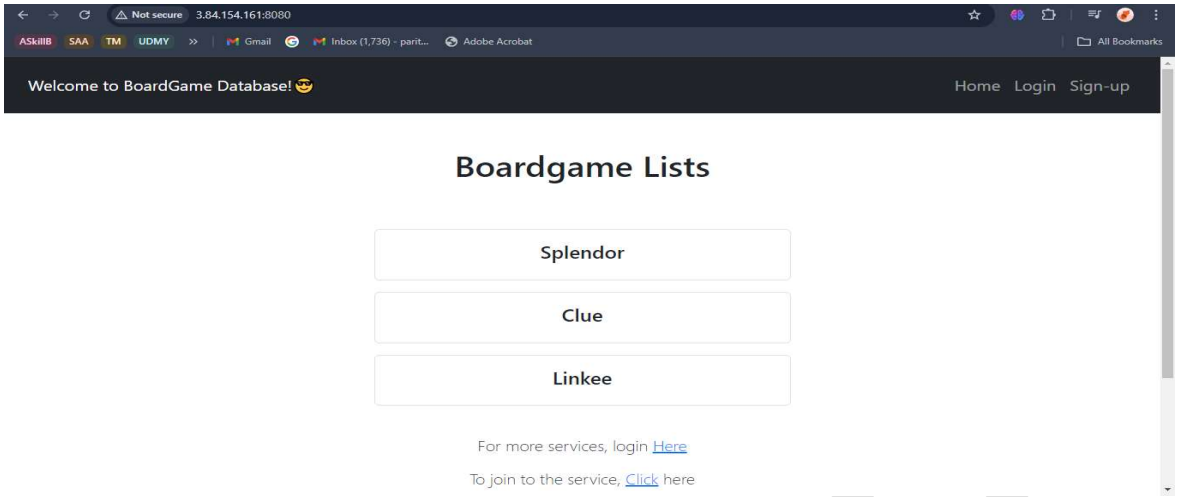
Build the Package: mvn package / go inside target folder / `cd target/`

Run the Application: . NODE EXPORTER

/ Java -jar database_service_project-0.0.2.jar & // Application runs on Port 9100 on browser <instance_ip>:9100

`Run command java -jar database_service_project-0.0.4.jar //` runs on Port 8080 on browser <instance_ip>:8080





[Fig 2: Successfully Install and Configure Node Exporter & Application]

Step 1.3 –: Install and Configure Prometheus, Black box Exporter, and Alert manager on Instance 2

1. Install Prometheus -:

Wget https://github.com/prometheus/prometheus/releases/download/v2.52.0/pro

tar -xvf prometheus-2.52.0.linux-amd64.tar.gz

mv prometheus-2.52.0.linux-amd64 / Prometheus

cd Prometheus

. /Prometheus **&**

2. Install Black box Exporter -:

Wget https://github.com/prometheus/blackbox_exporter/releases/download/v0.25.0/blackbox_exporter-0.25.0.linux-amd64.tar.gz

tar -xvf blackbox_exporter-0.25.0.linux-amd64.tar.gz

mv blackbox_exporter-0.25.0.linux-amd64/ Blackboxexporter

cd Blackboxexporter

./ Blackboxexporter &

3. Install Alertmanager -:

Wget https://github.com/prometheus/alertmanager/releases/download/v0.27.0/alertmanager-0.27.0.linux-amd64.tar.gz

tar -xvf alertmanager-0.27.0.linux-amd64.tar.gz

mv alertmanager

cd alertmanager

./ alertmanager **&**

Step 1.4 -: Configuration Files

Prometheus Configuration (prometheus.yml) Go inside the prometheus.yml file and add these configurations...

• **Global Configuration** -: Global: - scrape interval: - 15s / evaluation interval :- 15s

• **Alertmanager Configuration** -: alerting: / alertmanagers / static_configs / targets ['localhost:9093']

• **Scrape Configuration** -:

Prometheus -: scrape_configs / job_name "prometheus" static_configs / targets: ["localhost: 9090"]

Node Exporter -: job_name: "node_exporter" / static_configs: - targets: [":9100"]

Blackbox Exporter -:

- job_name: 'blackbox'

metrics_path: /probe

params:

module: [http_2xx]

static_configs: -

targets :http://prometheus.io / https://prometheus.io / http://:8080/

relabel_configs:

source_labels: [__address__]

target_label: __param_target

source_labels: [__param_target]

target_label: instance

target_label: __address__

replacement:{instance_IP}:9115

Note : You should restart your Prometheus or other services after completing all this configuration using this command before that need to stop / restart service which is running in background for that I'll run the command = **pgrep** Prometheus / You will get some service (ProcessID) id . Example - 5675 / Kill this service using command: **Kill** 5675 / u ll see Prometheus is stop running / run Prometheus - **./** Prometheus **&**

Alert Rules Configuration : (alert_rules.yml) - Alert Rules Group: Create a new file inside the prometheus directory named alert_rules.yml .

groups:

name: alert_rules

rules:

- alert: InstanceDown

expr: up == 0

for: 1m

labels:

severity: "critical"

annotations:

summary: "Endpoint {{ $labels.instance }} down"

description: "{{ $labels.instance }} of job {{ $labels.job }} has been down for more than 1 minute."

- alert: WebsiteDown

expr: probe_success == 0

for:1m

labels:

severity: critical

annotations:

description: The website at {{ $labels.instance }} is down.

```
summary: Website down

- alert: HostOutOfMemory

expr: node_memory_MemAvailable / node_memory_MemTotal * 100 < 25

for: 5m

labels:

severity: warning

annotations:

summary: "Host out of memory (instance {{ $labels.instance }})"

description: "Node memory is filling up (< 25% left)\n VALUE = {{ $value }}\n LABELS: {{ $labels }}"

- alert: HostOutOfDiskSpace

expr: (node_filesystem_avail{mountpoint="/"} * 100) / node_filesystem_size{mountpoint="/"} < 50

for: 1s

labels:

severity: warning

annotations:

summary: "Host out of disk space (instance {{ $labels.instance }})"

description: "Disk is almost full (< 50% left)\n VALUE = {{ $value }}\n LABELS: {{ $labels }}"

- alert: HostHighCpuLoad

expr: (sum by (instance) (irate(node_cpu{job="node_exporter_metrics",mode="idle"}[5m]))) > 80

for: 5m

labels:

severity: warning

annotations:

summary: "Host high CPU load (instance {{ $labels.instance }})"

description: "CPU load is > 80%\n VALUE = {{ $value }}\n LABELS: {{ $labels }}"
```

- alert: ServiceUnavailable

expr: up{job="node_exporter"} == 0

for: 2m

labels:

severity: critical

annotations:

summary: "Service Unavailable (instance {{ $labels.instance }})"

description: "The service {{ $labels.job }} is not available\n VALUE = {{ $value }}\n LABELS: {{ $labels }}" - alert: HighMemoryUsage

expr: (node_memory_Active / node_memory_MemTotal) * 100 > 90

for: 10m

labels:

severity: critical

annotations:

summary: "High Memory Usage (instance {{ $labels.instance }})"

description: "Memory usage is > 90%\n VALUE = {{ $value }}\n LABELS: {{ $labels }}"

- alert: FileSystemFull

expr: (node_filesystem_avail / node_filesystem_size) * 100 < 10

for: 5m

labels:

severity: critical

annotations:

summary: "File System Almost Full (instance {{ $labels.instance }})"

description: "File system has < 10% free space\n VALUE = {{ $value }}\n LABELS: {{ $labels }}"

Alertmanager Configuration -: alertmanager.yml Routing Configuration / add these in the alertmanager.yml file..

route:

roup_by: ['alertname']

group_wait: 30s

group_interval: 5m

repeat_interval: 1h

receiver: 'email-notifications'

receivers:

- name: 'email-notifications'

email_configs: - to: dc29912@gmail.com

from: test@gmail.com

smarthost: smtp.gmail.com:587

auth_username: your_email

auth_identity: your_email

auth_password: "bdcm derv kope abcd"

send_resolved: true

inhibit_rules:

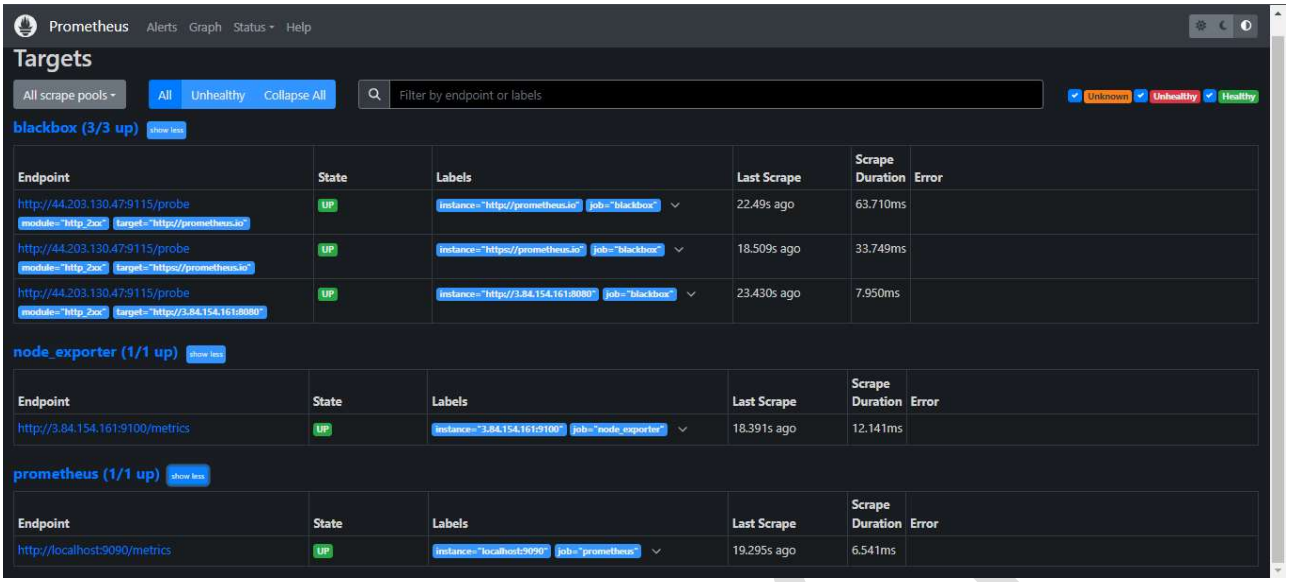
- source_match:

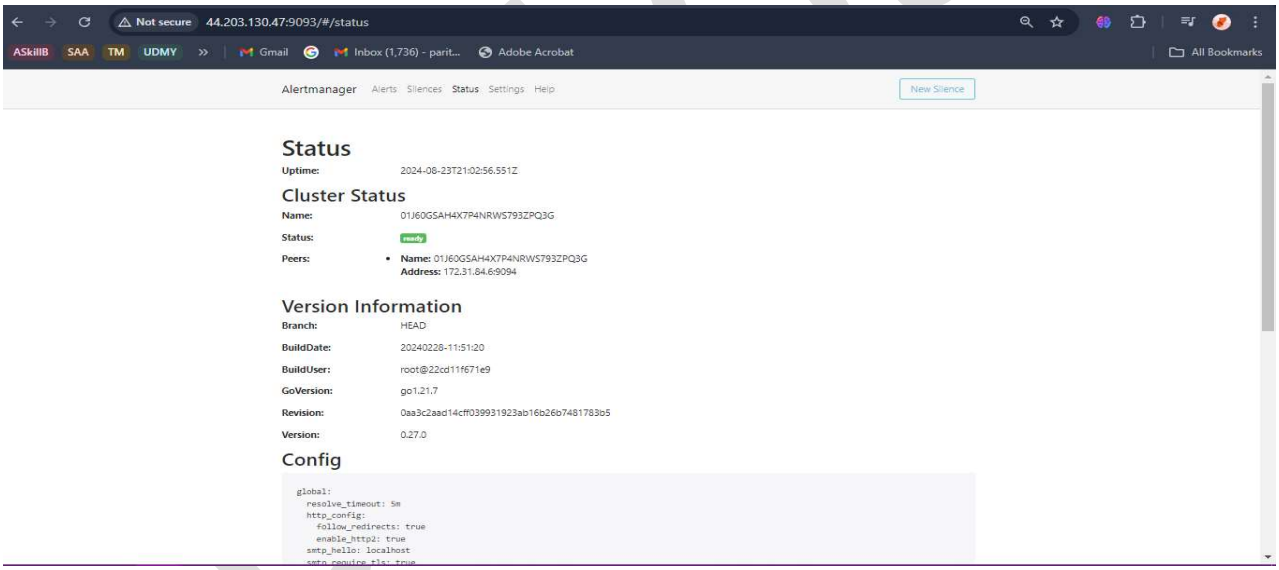
severity: 'critical'

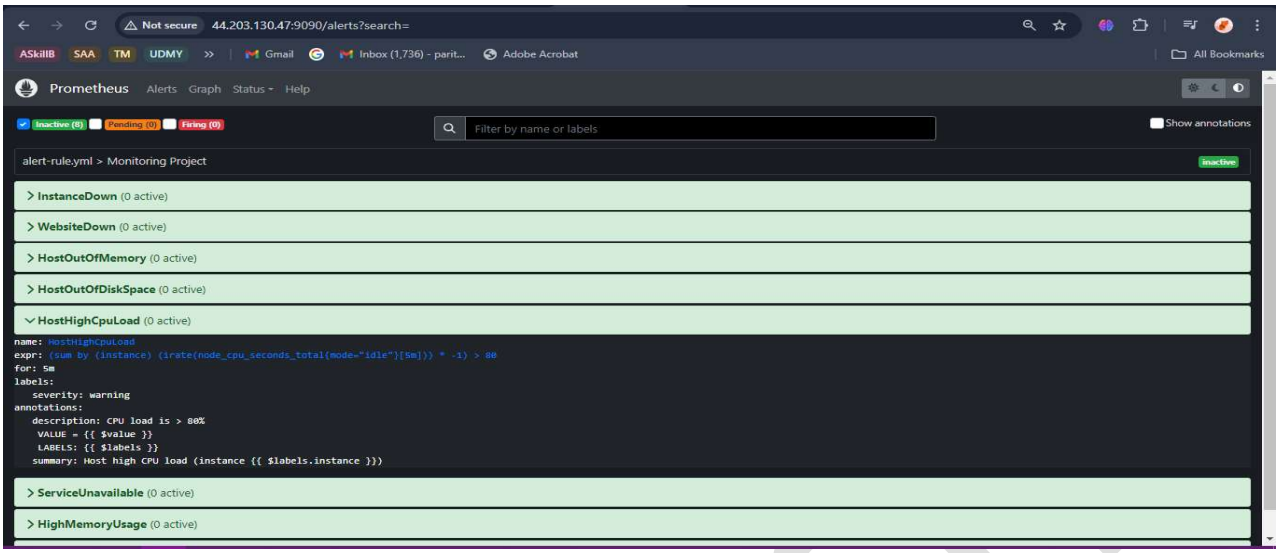
target_match:

severity: 'warning'

equal: ['alertname', 'dev', 'instance']


Prometheus
Alerts
Graph
Status
Help
Targets
All scrape pools
All
Unhealthy
Collapse All
Filter by endpoint or labels
Unknown
Unhealthy
Healthy
blackbox (3/3 up)
show less
Endpoint
State
Labels
Last Scrape
Scrape Duration
Error
http://44.203.130.47:9115/probe
module="http_2xx"
target="http://prometheus.io"
UP
instance="http://prometheus.io"
job="blackbox"
22.49s ago
63.710ms
http://44.203.130.47:9115/probe
module="http_2xx"
target="https://prometheus.io"
UP
instance="https://prometheus.io"
job="blackbox"
18.509s ago
33.749ms
http://44.203.130.47:9115/probe
module="http_2xx"
target="http://3.84.154.161:8080"
UP
instance="http://3.84.154.161:8080"
job="blackbox"
23.430s ago
7.950ms
node_exporter (1/1 up)
show less
Endpoint
State
Labels
Last Scrape
Scrape Duration
Error
http://3.84.154.161:9100/metrics
UP
instance="3.84.154.161:9100"
job="node_exporter"
18.391s ago
12.141ms
prometheus (1/1 up)
show less
Endpoint
State
Labels
Last Scrape
Scrape Duration
Error
http://localhost:9090/metrics
UP
instance="localhost:9090"
job="prometheus"
19.295s ago
6.541ms


Not secure
44.203.130.47:9093/#/status
ASkillB
SAA
TM
UDMY
Gmail
Inbox (1,736) - parit...
Adobe Acrobat
All Bookmarks
Alertmanager
Alerts
Silences
Status
Settings
Help
New Silence
Status
Uptime:
2024-08-23T21:02:56.551Z
Cluster Status
Name:
01J60GSAH4X7P4NRWS793ZPQ3G
Status:
ready
Peers:
Name: 01J60GSAH4X7P4NRWS793ZPQ3G
Address: 172.31.84.6:9094
Version Information
Branch:
HEAD
BuildDate:
20240228-11:51:20
BuildUser:
root@22cd11f671e9
GoVersion:
go1.21.7
Revision:
0aa3c2aad14cff039931923ab16b26b7481783b5
Version:
0.27.0
Config
global:
resolve_timeout: 5m
http_config:
follow_redirects: true
enable_http2: true
smtp_hello: localhost
smtp_require_tls: true



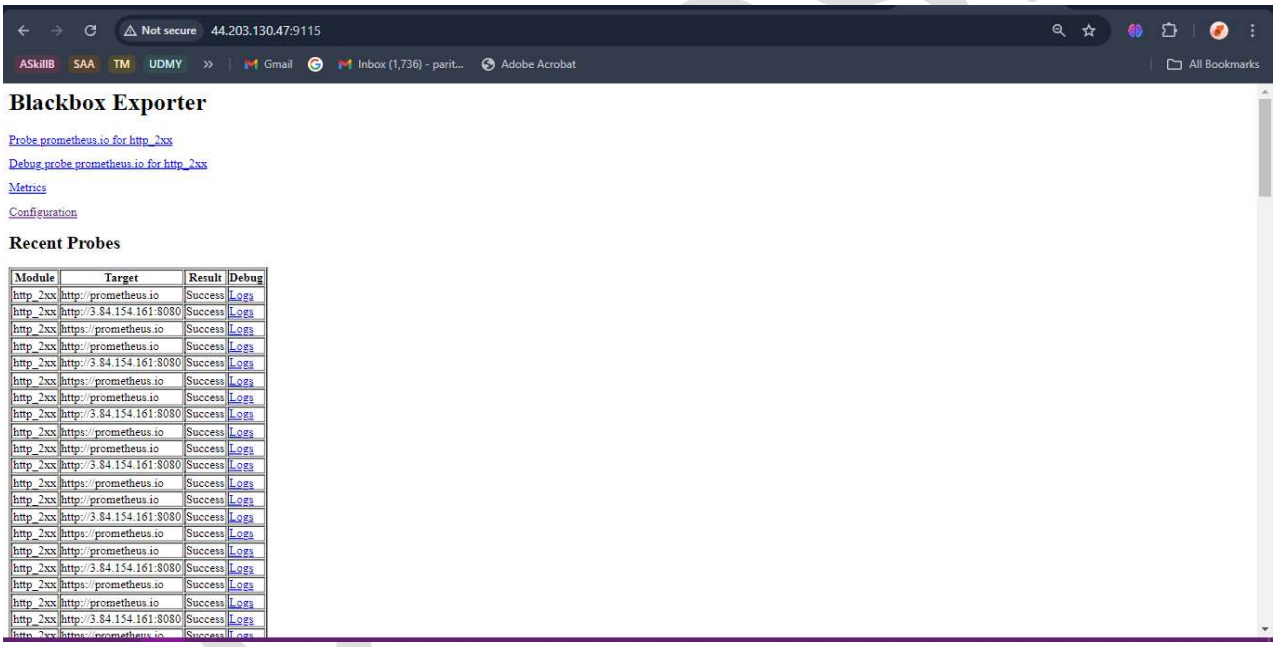


Fig 3: Successful Creation of Node Exporter, Blackbox Exporter, Prometheus, Alert manager).

Conclusion –

In this project, we created a strong monitoring system using Prometheus and its various tools to keep track of the reliability and performance of a web application running on AWS EC2 instances. We used several components like Node Exporter to gather detailed metrics about the server, Blackbox Exporter to check if the different parts of the application are accessible, and Alertmanager to handle alerts whenever something goes wrong. This setup allows us to monitor everything closely and respond quickly to any issues that arise.

DELL