



QR Code generator using CI/CD Pipeline

TERM PROJECT

2023W CBD 2244 1 [B107] DevOps Fundamentals for Canadian Enterprises

INSTRUCTOR: Mohammad Salim

Course: DOCT

Prepared by:

Athul Sukumaran (C0880215)

ABSTRACT

Continuous Integration/Continuous Deployment (CI/CD) is a development approach that helps streamline the software development process by automating various stages of the software development lifecycle. More than that, the primary aim of this project is to create e a Continuous Integration/Continuous Deployment pipeline for a QR code generator and scanner application. The application will be developed using Python and deployed in Docker containers, managed using Docker Swarm. The project uses a variety of DevOps tools such as Azure and AWS for building virtual machines, Terraform for infrastructure as code, Ansible for automation, Nagios for monitoring, and Jenkins for continuous integration and delivery. The Terraform is used to create the required infrastructure in Azure, including the resources like virtual networks, subnets, security groups, and virtual machines

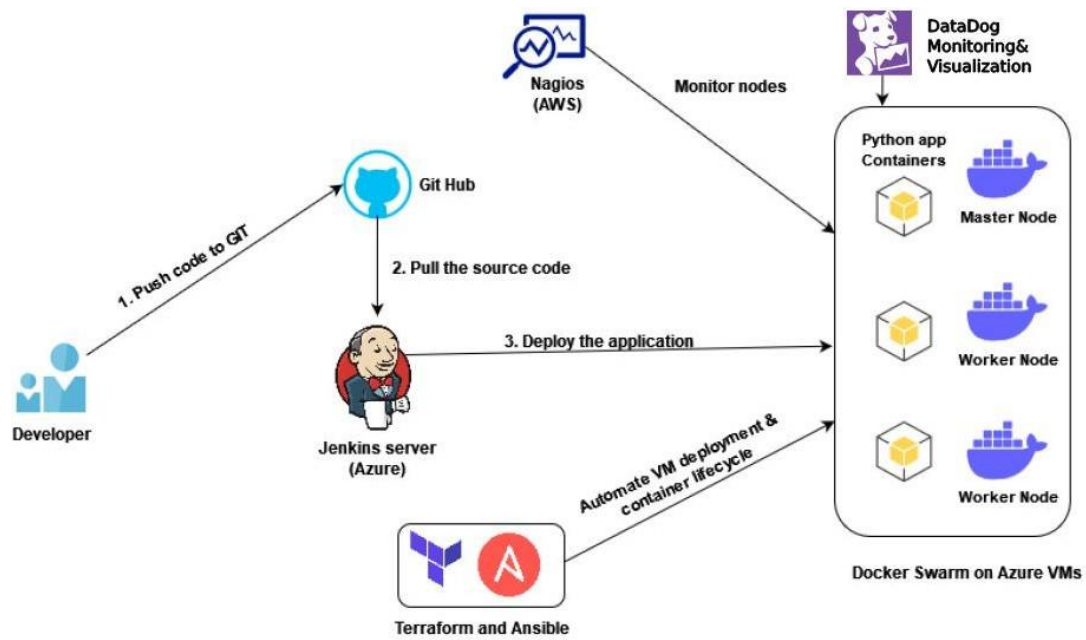
INTRODUCTION

This project focuses on implementing a Continuous Integration/Continuous Deployment (CI/CD) pipeline for a software development team. This project aims to automate the entire software development lifecycle, from code changes to deployment, while ensuring quality and reliability. The CI/CD pipeline consists of multiple stages, including building, testing, and deploying the application. Used modern tools and technologies such as Git, Jenkins, Docker, and Kubernetes to automate the pipeline in this project.

The CI/CD pipeline will automate the application's build and deployment processes, ensuring that the most recent changes are smoothly integrated, tested, and deployed. Jenkins will be used as the continuous integration tool, automatically initiating builds, testing, and deploying changes anytime they are uploaded to the code repository. The source code is committed to the version control system GIT, and updates are automatically grabbed by Jenkins before being tested and deployed as a Docker Swarm container. Terraform will be used to automate the provisioning of the cloud infrastructure that will host the docker swarm cluster, while Ansible will be used to manage the life cycle of the docker containers that will operate in the docker swarm cluster.

Clients will benefit from numerous aspects of the project, including faster and more reliable deployments, decreased human tasks, and increased application quality. As a result, implementing a CI/CD pipeline for the QR code generator and scanner application would increase the efficiency and reliability of the application development and deployment process dramatically. Another key feature that improves the end-user experience is high availability and fault tolerance.

Pipeline Architecture:



Methodology

Step 1: Create the application using Python

The application is created using Python, and the code is stored in the version control system (Github) repository. The source code is available in the below git URL.

- Application is a web app that uses the Pywebio framework
- It generates two types of codes, **QR codes and Bar codes**.
- It also allows users to download the generated codes

GitHub URL: [insightiq-devops/app.py at main · insightiq/insightiq-devops \(github.com\)](https://github.com/insightiq-devops/app.py)

Source code:

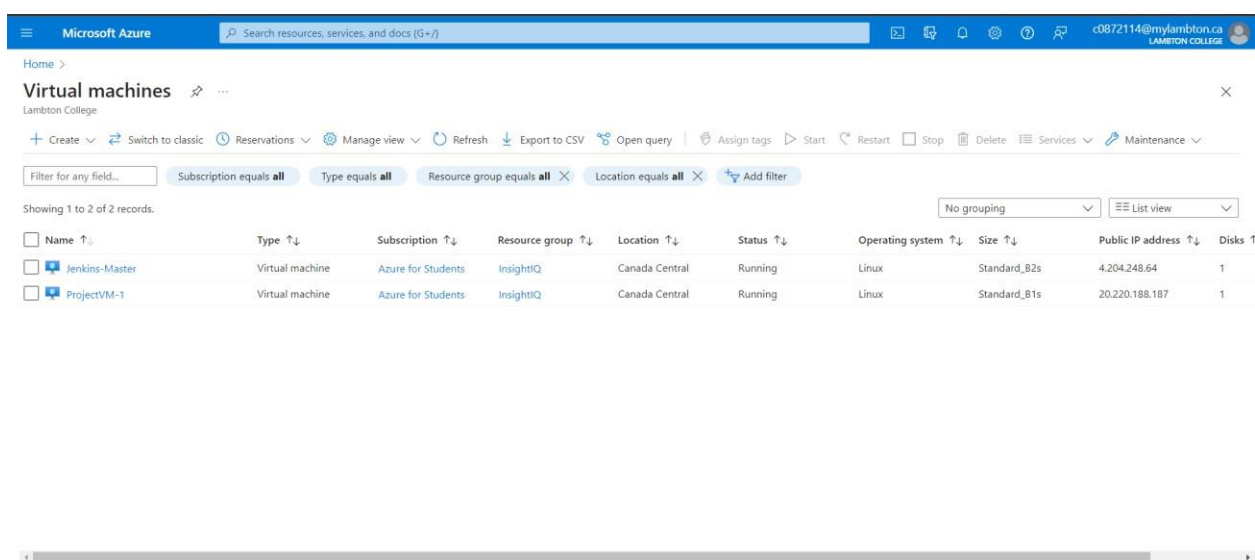
```

237 lines (154 sloc) | 7.23 KB
Raw Blame
1
2
3
4 #Importing Modules
5 import argparse
6 from pywebio.input import *
7 from pywebio.output import *
8 from flask import Flask
9 from pywebio.session import *
10 from pywebio.platform.flask import webio_view
11 from pywebio import STATIC_PATH
12 import qrcode
13 from PIL import Image
14 import re
15 import time
16 from pywebio.session import run_js
17 from pywebio import start_server
18 import barcode
19 #from barcode import EAN13
20 from barcode import Code39
21
22 #-----
23
24 #Creating a flask appls
25
26
27 app=Flask(__name__)
28
29
30
31 #Main function that creates the QR code
32
33 def QR():
34
35     put_html(r"""<h1 align="center"><strong> CODE GENERATOR:v2</strong></h1>""") # App Name in Main screen
36     # Drop-down selection

```

Step 2: Create the Virtual machines in Azure for the Jenkins server and Docker swarm.

Two virtual machines are created in the Azure cloud platform named Jenkins master server and Docker. The Jenkins server is used for automation purposes, and Docker is used for application deployment.



Name	Type	Subscription	Resource group	Location	Status	Operating system	Size	Public IP address	Disks
Jenkins-Master	Virtual machine	Azure for Students	InsightIQ	Canada Central	Running	Linux	Standard_B2s	4.204.248.64	1
ProjectVM-1	Virtual machine	Azure for Students	InsightIQ	Canada Central	Running	Linux	Standard_B1s	20.220.188.187	1

Step 3: Pull the Source code from the GitHub repository to the Jenkins server using pipelines.

Jenkins is an open-source automation server used for building, testing, and deploying software. It provides a wide range of plugins and integrations with other tools, allowing it to be customized to meet the needs. Used the Jenkins tool to automate the entire software development lifecycle, from building and testing code to deploying in Docker environments.

When any changes are made to the source code stored in the GitHub repository Jenkins will automatically pull the source code using the Git WebHook feature and initiate the Jenkins pipeline.

The Deployment stages in Jenkins:

- Stage-1: Declarative: Checkout SCM

If any new changes are pushed to the repository Jenkins triggers the job

- Stage-2: Clone GitHub Repository

In this stage cloning the Git Hub repository and updating the app.py file.

- Stage-3: Copy source code to Docker swarm

The updated source code is then pushed to Docker Swarm

- Stage-4: Build & Push the new Image to the Docker hub

Docker will build a new image using the updated code and push this image to Docker Hub

- Stage-5: Deploying new Service in Docker Swarm

A new service is deployed using the image and updated changes are reflected in the application.

Jenkins Search (CTRL+K) admin log out

Dashboard >

+ New Item

People All +

Build History

Manage Jenkins

My Views

Build Queue No builds in the queue.

Build Executor Status 1 Idle

S	W	Name	Last Success	Last Failure	Last Duration
✓	🔔	insightIQ	12 min #41	19 min #40	3 min 3 sec

Icon: S M L

Atom feed for all Atom feed for failures Atom feed for just latest builds

Jenkins Search (CTRL+K) admin log out

Dashboard > insightIQ >

Pipeline insightIQ DevOps Pipeline

Status

Changes

Build Now

Configure

Delete Pipeline

Full Stage View

Rename

Pipeline Syntax

GitHub Hook Log

Build History trend

Filter builds...

#42 21 Apr 2023, 01:14

#41 21 Apr 2023, 01:04

Stage View

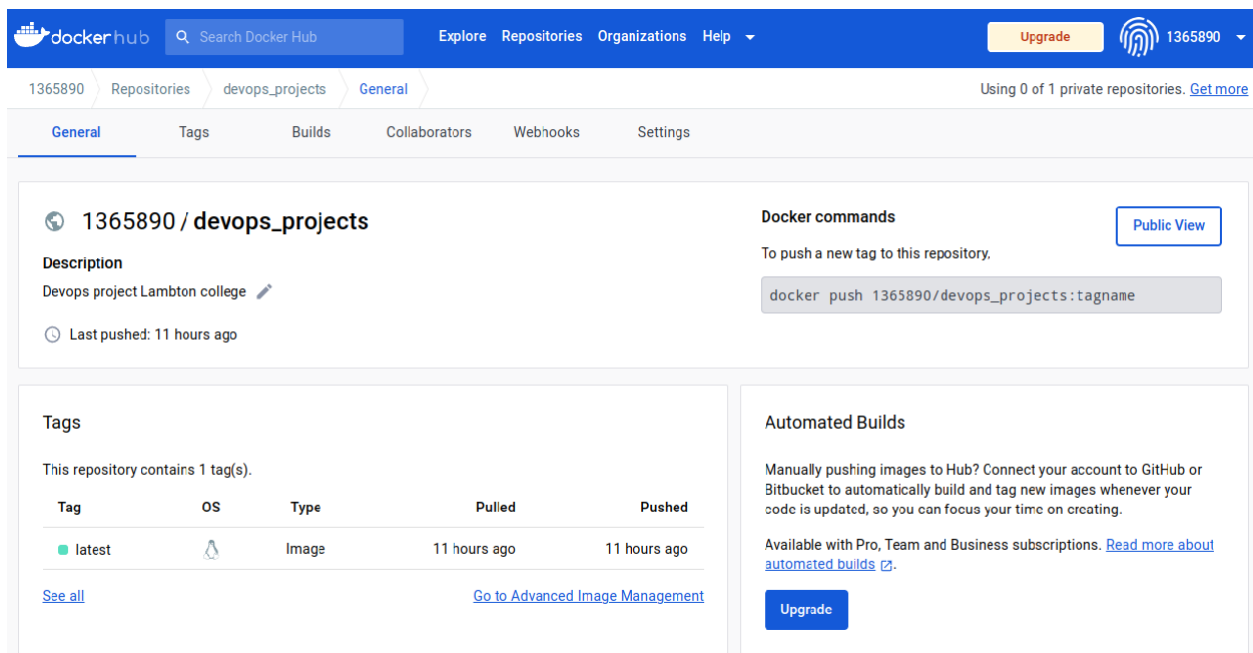
	Declarative: Checkout SCM	Clone GITHUB Repository	Copy source code to Docker swarm	Build & Push the new image to Dockerhub	Deploying new Service in Docker Swarm
Average stage times: (Average full run time: ~1min 51s)	2s	1s	21s	1min 45s	1min 31s
#42 Apr 20 21:14 1 commit	2s	1s	27s	59s	13s
#41 Apr 20 21:04 1 commit	2s	2s	43s	54s	1min 14s

Step 5: Deployment of the application using Docker Swarm

Docker is a software platform that allows developers to create, deploy, and run applications in containers easily. Containers are lightweight, portable, and self-contained environments that can run an application and all its dependencies. The Updated source code is pulled to docker from the Jenkins server.

- Docker swarm is a container orchestration platform consisting of a cluster of Docker Nodes (Master and Worker)
- It helps in the high availability and fault tolerance of containers
- Have 2 nodes, one Master and another Worker
- Master node controls the cluster operations.
- Cluster is built on top of Azure Virtual machines

Once the image is built by the docker, it will be pushed and stored in Docker Hub.



The screenshot shows the Docker Hub interface for the repository **1365890 / devops_projects**. The page includes a search bar, navigation links (Explore, Repositories, Organizations, Help), and an 'Upgrade' button. The repository details section shows the description 'Devops project Lambton college' and 'Last pushed: 11 hours ago'. A 'Docker commands' section provides the command `docker push 1365890/devops_projects:tagname`. The 'Tags' section shows a table with one tag, 'latest', which was pulled and pushed 11 hours ago. The 'Automated Builds' section explains how to connect GitHub or Bitbucket for automated builds.

Tag	OS	Type	Pulled	Pushed
latest	linux	Image	11 hours ago	11 hours ago

This image is pulled by the docker swarm. Using this image, created a docker swarm service with two replicas running on two different nodes for high availability and fault tolerance.

The service is listening on port 80, and have deployed an Azure load balancer that will distribute the traffic between the two docker servers.

```
1 insightiq@ProjectVM-1: ~
insightiq@ProjectVM-1:~$ sudo docker service ls
ID            NAME          MODE          REPLICAS  IMAGE                                     PORTS
6l8gmmdobw22 insightiq      replicated    2/2        1365890/devops_projects:latest        *:80->8083/tcp
insightiq@ProjectVM-1:~$
```

Docker swarm running service (insightiq) with 2 replicas.

```
1 insightiq@ProjectVM-1: ~
insightiq@ProjectVM-1:~$ sudo docker service ps insightiq
ID            NAME          IMAGE                                     NODE          DESIRED STATE  CURRENT STATE
rb0vqi4hacn7 insightiq.1    1365890/devops_projects:latest        ProjectVM-1   Running        Running 11 hours ago
xcgu474gkeec insightiq.2    1365890/devops_projects:latest        Jenkins-Master Running        Running 2 minutes ago
xpp3pq5q5t96 \_ insightiq.2 1365890/devops_projects:latest        Jenkins-Master Shutdown        Shutdown 2 minutes ago
insightiq@ProjectVM-1:~$
```

Containers are distributed across 2 nodes

```
insightiq@ProjectVM-1:~$ sudo docker node ls
ID            HOSTNAME          STATUS  AVAILABILITY  MANAGER STATUS  ENGINE VERSION
ryb5x0fmqrkqncnoqmuywuvra Jenkins-Master    Ready    Active         Reachable        23.0.4
b66tgkkivhqtkp1zpst0mi37s * ProjectVM-1       Ready    Active         Leader           23.0.3
iuuj7vjm5d6wj4wuo27u1r8dnp workernode-1     Down     Active         Leader           23.0.3
insightiq@ProjectVM-1:~$
```

Details of Nodes participating in the cluster

Docker swarm service: A service defines the tasks to execute on the manager or worker nodes.

Service can run across multiple nodes and support advanced features like replication, high availability, automated container migration, etc.

Infrastructure as Code (Terraform)

- Used Terraform as an Infrastructure as a code tool to deploy our virtual machines.
- Resources are created using the Azure provider
- Following resources are created:
- Resource group, VM, security group, Virtual network
- network interface, Public IP.

```
#Mentioning the provider as Azure
provider "azurerm" {
  #version = "= 2.0.0"
  features {}
}

#Creating an Azure resource group
resource "azurerm_resource_group" "insightiq_resource_group" {
  name="insightiq-resource-group"
  location = "Canada Central"
}
```

Configuration Management (Ansible)

- Used Ansible as a configuration management tool to automate the container lifecycle.
- By using the Ansible playbook, did the following tasks
- Copy the code to the remote docker server
- Destroy the previously running containers before deploying new containers

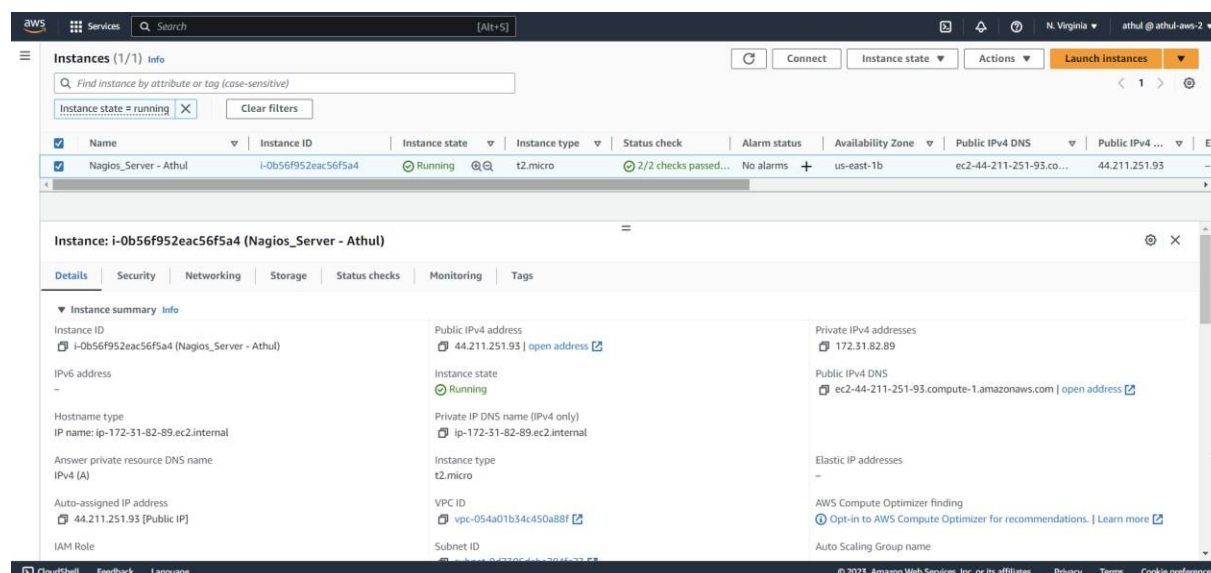
- Build, and push the docker image to the docker hub

```
---  
- hosts: all  
  become: True  
  
  tasks:  
    - name: Copy source code and move the code to Docker node  
      synchronize:  
        src: /var/lib/jenkins/workspace/insightIQ/  
        dest: /home/jenkins/Devops-project/  
        delete: yes
```

Step 6: Monitoring the VMs using Nagios

Overview of Nagios:

Monitoring is an essential aspect of the DevOps pipeline. It helps in ensuring that the application infrastructure and its components are running smoothly and efficiently. One of the most significant advantages of monitoring is its ability to identify issues before they become critical, reducing the downtime of the application.



Nagios is an open-source monitoring tool that provides real-time monitoring of the application infrastructure and its components. In our project, have opted to use the AWS cloud service to provide High Availability (HA) for the monitoring part. This ensures that the monitoring system is always available, even in the case of failure of Azure service where the remote hosts are deployed. AWS is a reliable cloud service provider offering various tools and services to ensure high availability, scalability, and security. To deploy Nagios, have used the Apache web server. The Apache web server is a widely used web server known for its stability and security. Nagios can be deployed on the Apache web server to provide a robust and scalable monitoring system.

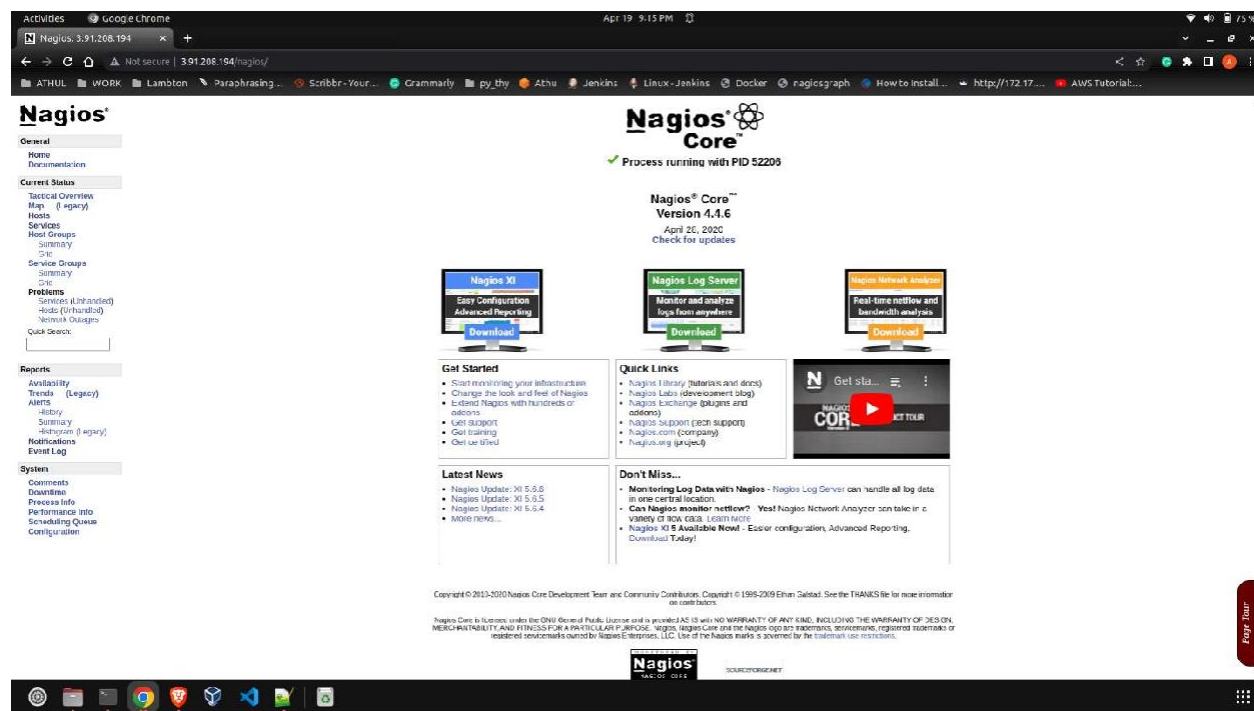
Nagios Configuration:

Nagios configuration is a crucial step in setting up an effective monitoring solution for any application infrastructure. Nagios will be used to monitor the virtual machine running the applications. This will include defining checks, notifications, and escalations to ensure the smooth operation of the application. As part of the Nagios configuration, device information such as hostname, IP address, and other details will be marked. This will help in identifying the

virtual machine quickly and taking appropriate action in case of any issues. In addition, the configuration of Nagios will include setting up a dashboard for real-time monitoring and alerts for critical issues. The dashboard will provide a comprehensive view of the health and status of the virtual machine and the services it is running. This will help in identifying issues early on and taking corrective action before they impact the application's performance or availability.

Nagios Dashboard:

The Nagios dashboard is a critical component of the Nagios monitoring system that provides a comprehensive view of the health and status of the virtual machine, network, and services of the application. It provides a centralized location for monitoring various components and services, making it easier to manage and troubleshoot issues. The right side of the Nagios dashboard interface provides various analysis tools and services that can be used to judge the availability and performance of virtual machines. These tools and services include graphs, charts, and tables that display metrics such as CPU usage, memory usage, and network bandwidth.



The screenshot displays the Nagios web interface in a Google Chrome browser window. The interface is divided into several sections:

- Current Network Status:** Shows the last update time (Apr 20 01:17:16 UTC 2023) and the Nagios version (4.4.0).
- Host Status Totals:** A summary table showing the number of hosts in different states: Up (1), Down (0), Unreachable (0), and Pending (0).
- Service Status Totals:** A summary table showing the number of services in different states: OK (11), Warning (0), Unknown (0), Critical (1), and Pending (0).
- Service Status Details For All Hosts:** A detailed table listing the status of various services across different hosts. The table includes columns for Host, Service, Status, Last Check, Duration, Attempt, and Status Information.

Host	Service	Status	Last Check	Duration	Attempt	Status Information
insight1	HTTP	OK	04-20-2023 01:14:09	0d 0h 33m 7s	1/0	HTTP OK - HTTP/1.1 200 OK - 7263 bytes in 0.036 second response time
insight1	PING	OK	04-20-2023 01:14:29	0d 0h 30m 47s	1/0	PING OK - Packet loss = 0%, RTT = 17.08 ms
insight1	SSH	OK	04-20-2023 01:11:07	0d 0h 48m 7s	1/0	SSH OK - OpenSSH_8.2p1 Ubuntu-Autumn0.5 (protocol 2.0)
insight2	HTTP	OK	04-20-2023 01:15:41	0d 0h 35m 35s	1/0	HTTP OK - HTTP/1.1 200 OK - 7263 bytes in 0.041 second response time
insight2	PING	OK	04-20-2023 01:07:13	0d 0h 47m 27s	1/0	PING OK - Packet loss = 0%, RTT = 18.39 ms
insight2	SSH	OK	04-20-2023 01:12:39	0d 0h 45m 47s	1/0	SSH OK - OpenSSH_8.2p1 Ubuntu-Autumn0.5 (protocol 2.0)
localhost	Current Load	OK	04-20-2023 01:12:38	0d 1h 4m 48s	1/4	OK - load average: 0.06, 0.01, 0.00
localhost	Current Users	OK	04-20-2023 01:14:11	0d 1h 4m 10s	1/4	USERS OK - 1 users currently logged in
localhost	HTTP	OK	04-20-2023 01:14:40	0d 1h 3m 33s	1/4	HTTP OK - HTTP/1.1 200 OK - 10945 bytes in 0.000 second response time
localhost	PING	OK	04-20-2023 01:16:12	0d 1h 2m 55s	1/4	PING OK - Packet loss = 0%, RTT = 0.05 ms
localhost	Root Partition	OK	04-20-2023 01:12:38	0d 1h 2m 10s	1/4	DISK OK - free space / 5177 MB (64.99% inode=89%)
localhost	SSH	OK	04-20-2023 01:12:38	0d 1h 2m 40s	1/4	SSH OK - OpenSSH_8.2p1 Ubuntu-3ubuntu0.2 (protocol 2.0)
localhost	Swap Usage	CRITICAL	04-20-2023 01:12:38	0d 0h 58m 3s	4/4	SWAP CRITICAL - 0% free (0 MB out of 0 MB) - Swap is either disabled, not present, or of zero size.
localhost	Total Processes	OK	04-20-2023 01:15:11	0d 1h 0m 25s	1/4	PROCS OK - 42 processes with STATE = R5ZDT

Nagios Plugins:

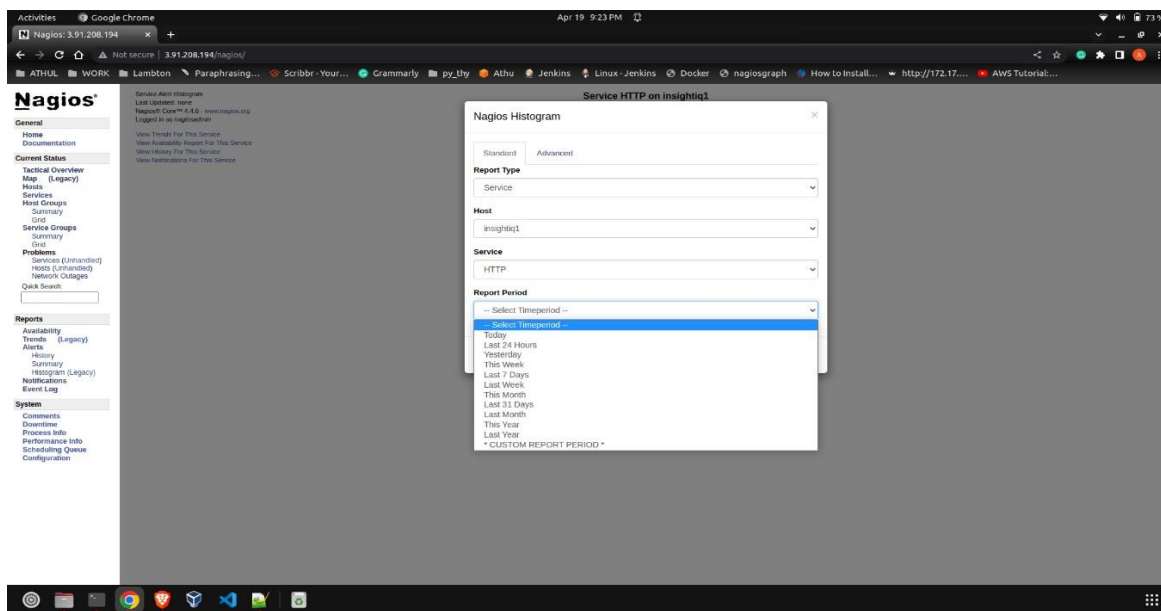
Nagios plugins are essential components of the Nagios monitoring system that allow you to monitor various components and services of remote host machines. One of the most commonly used Nagios plugins is the NRPE (Nagios Remote Plugin Executor) plugin, which enables you to execute plugins on remote hosts. This plugin allows you to monitor various aspects of the application, such as CPU usage, disk space, and memory usage. By monitoring these critical resources, you can identify potential issues before they cause significant downtime or failure of the application.

The metrics monitored by Nagios plugins are customizable, allowing you to configure them to meet your specific monitoring requirements. This provides a significant advantage in terms of providing a comprehensive view of the application's health and performance, which is

vital for identifying and resolving potential issues. The NRPE daemon runs on the remote host, and the check_nrpe plugin runs on the Nagios server. The check_nrpe plugin sends requests to the NRPE daemon on the remote host to execute the desired plugin and return the results. This communication between the Nagios server and the NRPE daemon enables the monitoring of a wide range of metrics, which is essential for ensuring the application's health and performance.

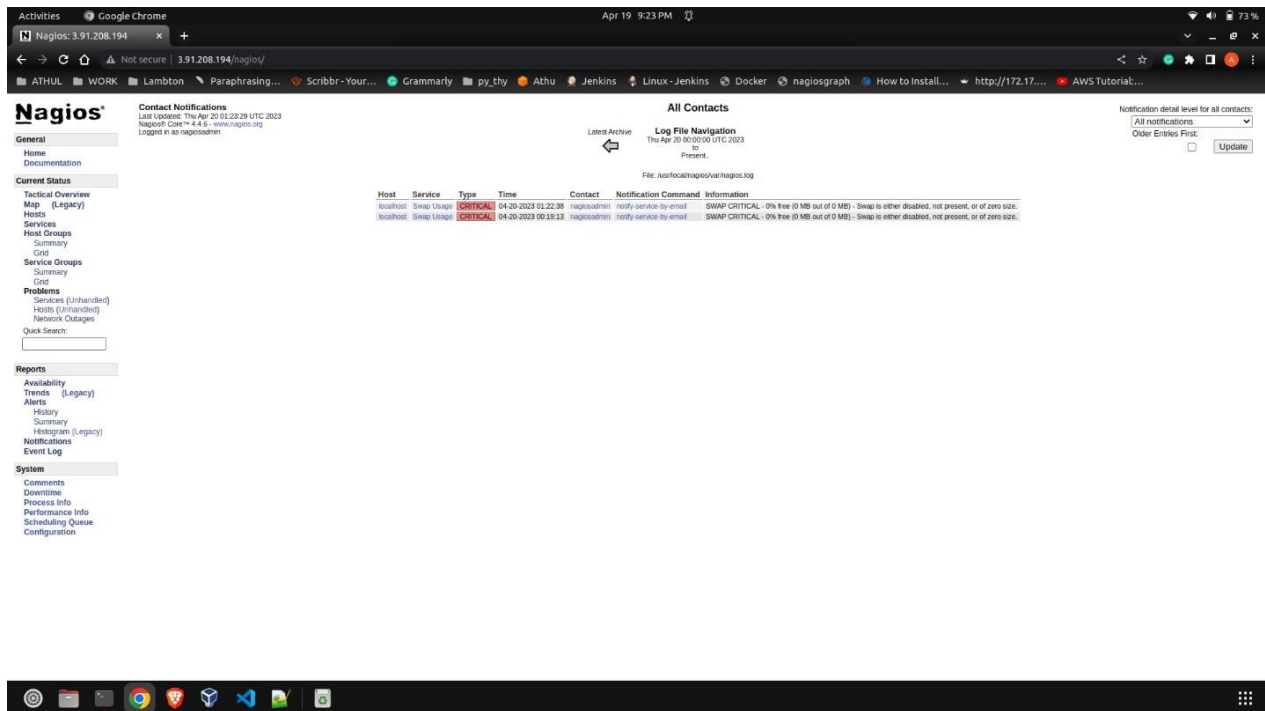
Nagios Reporting:

Nagios reporting is a powerful feature that provides comprehensive insights into the performance and trends of the monitored infrastructure. Reports can be generated for each remote host as well as the local host. These reports provide detailed information about the performance metrics, including the uptime, availability, and response time of the monitored services. The reports also help in identifying issues and improving the efficiency and reliability of the application by highlighting the areas that require attention. The data is presented in an easy-to-understand format, making it easier for the DevOps team to take corrective actions.



Nagios Notifications:

Nagios has a powerful notification system that can be used to alert DevOps teams about any issues detected during monitoring. This feature plays a crucial role in reducing the downtime of the application by notifying the responsible team as soon as an issue is detected.

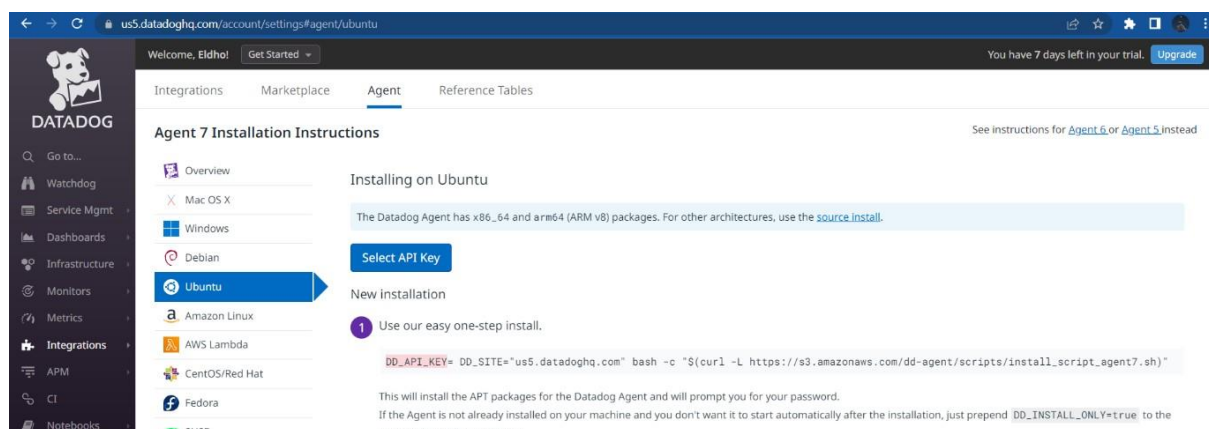


The customizable email templates of Nagios can be configured to include detailed information about the problem, its severity, and its potential impact on the application or infrastructure. This helps DevOps teams quickly identify the issue and take necessary action to resolve it. Nagios allows you to configure email notifications to be sent to specific individuals or groups based on their roles or responsibility. This feature ensures that the right person or team is notified about the issue, reducing the response time and improving the overall efficiency of the monitoring system.

Step 7: Elaborated visualization of the VMs performance using DataDog.

Datadog is a cloud monitoring tool that provides real-time insights into the performance of applications, infrastructure, and networks. In this technical report, I will document the steps I have taken to install and configure the Datadog agents and create a dashboard for monitoring a web application.

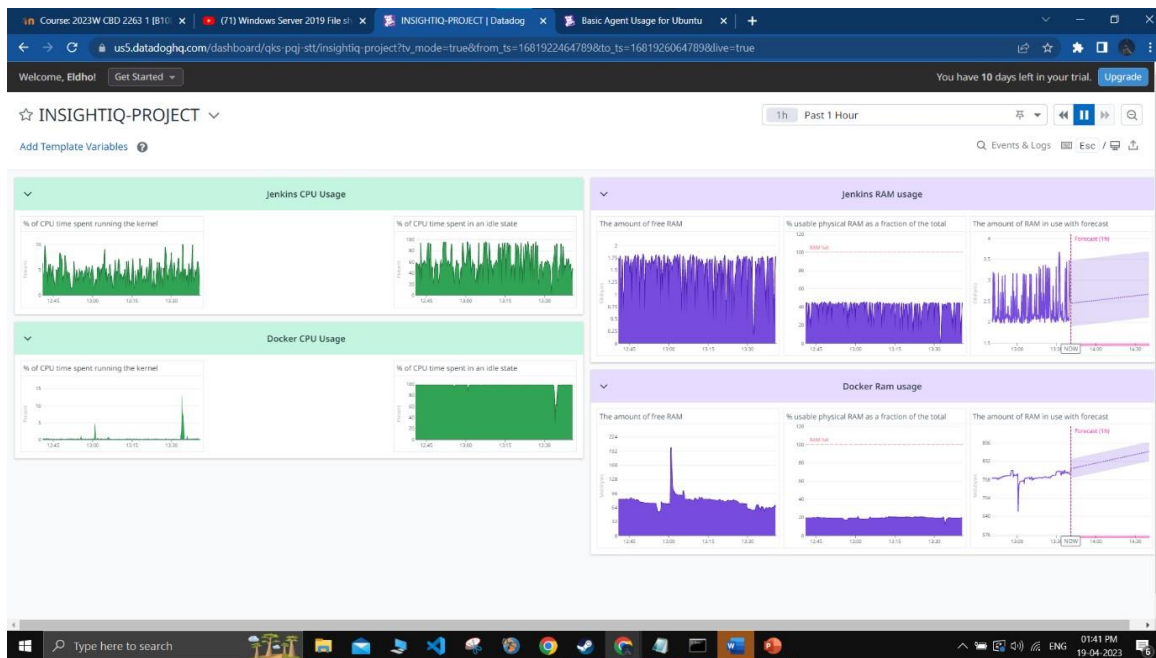
- Installation and Configuration of Datadog Agents:
- Sign up for a Datadog account: First, have to sign up for a Datadog account by visiting their website and creating an account.
- Install the Datadog Agent on the server: have installed the Datadog Agent on the server



Start the Datadog Agent: After configuring the Datadog Agent, start it using the following command: *“sudo systemctl start datadog-agent”*

- Log in to Datadog: After starting the Datadog Agent.
- Create a new dashboard: In the Datadog dashboard, create a new dashboard by clicking on the “New Dashboard” button.

Add widgets to the dashboard: I added various widgets to the dashboard to monitor my web application's performance. For example, I added a widget to monitor the system CPU and system RAM and SWAP.



Conclusion

In conclusion, the Continuous Integration/Continuous Deployment (CI/CD) approach is an essential element of modern software development that helps streamline the software development process by automating various stages of the software development lifecycle. This project demonstrates the practical application of CI/CD by creating a pipeline for a QR code generator and scanner application using Python and Docker containers managed by Docker Swarm. The project leverages a variety of DevOps tools such as Azure, AWS, Terraform, Ansible, Nagios, and Jenkins to enable continuous integration and delivery. With Terraform, the required infrastructure in Azure, including virtual networks, subnets, security groups, and virtual machines, can be created seamlessly. This project provides a clear example of how DevOps tools can be used to simplify the software development process and improve the efficiency and reliability of software delivery. For monitoring, the Nagios is an essential tool in the DevOps pipeline that helps in identifying issues, reducing downtime, and improving the efficiency and reliability of the remote hosts. Nagios configuration, dashboard, plugins, reporting, and notifications provide a comprehensive view of the VM's health and performance.

References

1. *PyWebIO - Build full stack web app with Python*. (n.d.).

<https://www.pyweb.io/>

2. *Swarm mode overview*. (2023, February 9). Docker Documentation.

<https://docs.docker.com/engine/swarm/>

3. *Table Of Contents · Nagios Core Documentation*. (n.d.).

<https://as-sets.nagios.com/downloads/nagioscore/docs/nagioscore/4/en/toc.html>

4. *Getting Started with the Agent*. (n.d.). Datadog Infrastructure and Application Monitoring.

https://docs.datadoghq.com/getting_started/agent/

5. Trapani, K. (2023). How to Integrate Jenkins with GitHub. *Cprime*.

<https://www.cprime.com/resources/blog/how-to-integrate-jenkins-github/>