Configuration Manual

Research Project
MSc Cloud Computing

Nitu Kumari
Student ID: x21215995

School of Computing
National College of Ireland

Supervisor: Diego Lugones

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Nitu Kumari
Student ID:	x21215995
Programme:	MSc Cloud Computing
Year:	2023
Module:	Research Project
Supervisor:	Diego Lugones
Submission Due Date:	14/8/2023
Project Title:	ANALYSIS OF DYNAMIC APPLICATION LOAD BALANCING IN KUBERNETES USING CDN
Word Count:	546
Page Count:	9

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Nitu Kumari
Date:	13th August 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

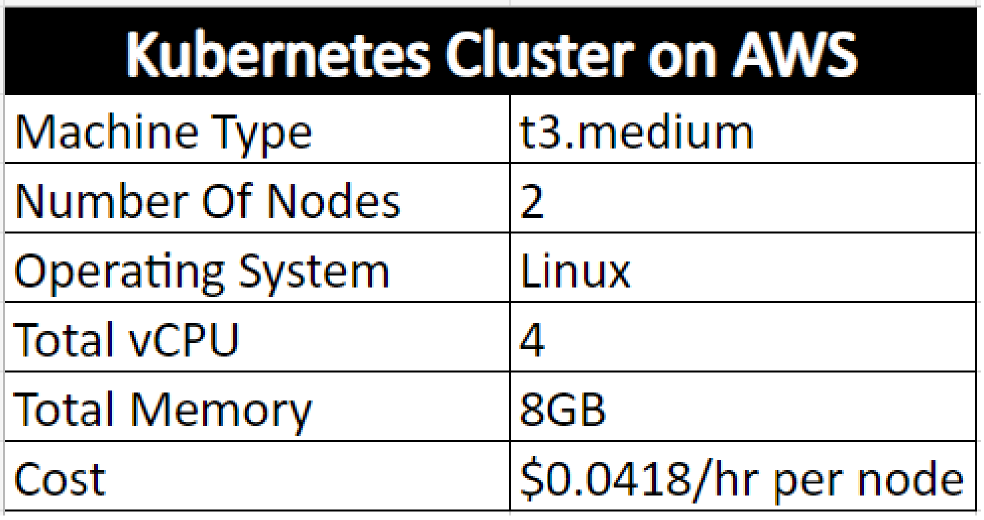
Nitu Kumari
x21215995

1 Introduction

This configuration manual details the installation, setup, and deployment of each piece of software, tool, and file needed to put the suggested system into practice.

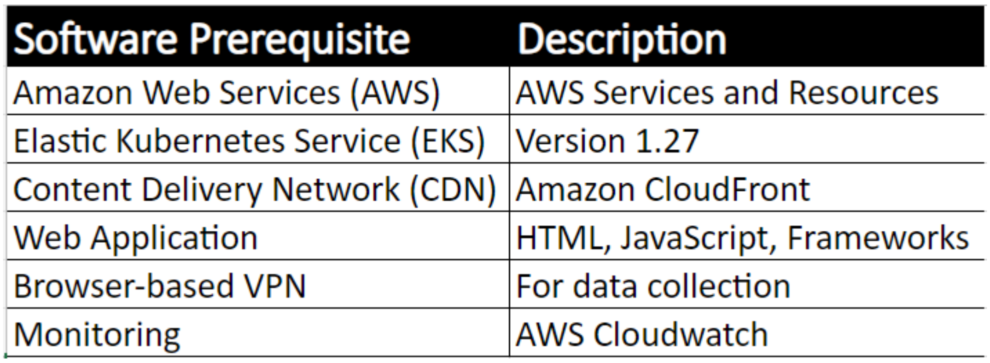
2 Prerequisites

The required software and hardware for implementing this project are listed below.



Kubernetes Cluster on AWS	
Machine Type	t3.medium
Number Of Nodes	2
Operating System	Linux
Total vCPU	4
Total Memory	8GB
Cost	$0.0418/hr per node

Figure 1: AWS EKS cluster details



Software Prerequisite	Description
Amazon Web Services (AWS)	AWS Services and Resources
Elastic Kubernetes Service (EKS)	Version 1.27
Content Delivery Network (CDN)	Amazon CloudFront
Web Application	HTML, JavaScript, Frameworks
Browser-based VPN	For data collection
Monitoring	AWS Cloudwatch

Figure 2: Required Software And Description

3 Implementation

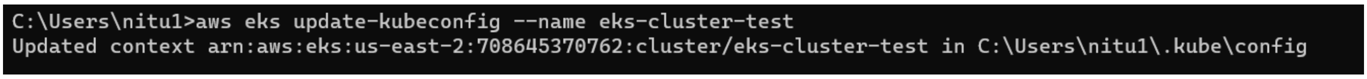
3.1 Deplying Kubernetes Cluster on Amazon Web Services (AWS)

1. **Create an AWS EKS cluster**

The AWS EKS cluster is created by using the AWS Management Console Algarni (2021)and specifying the cluster name, region, and other required settings.

2. **Configure kubectl to Access the Cluster**

By running the below command, the kubectl has been configured with the cluster details *Getting started with Amazon EKS* (Year).



```
C:\Users\nitu1>aws eks update-kubeconfig --name eks-cluster-test
Updated context arn:aws:eks:us-east-2:708645370762:cluster/eks-cluster-test in C:\Users\nitu1\.kube\config
```

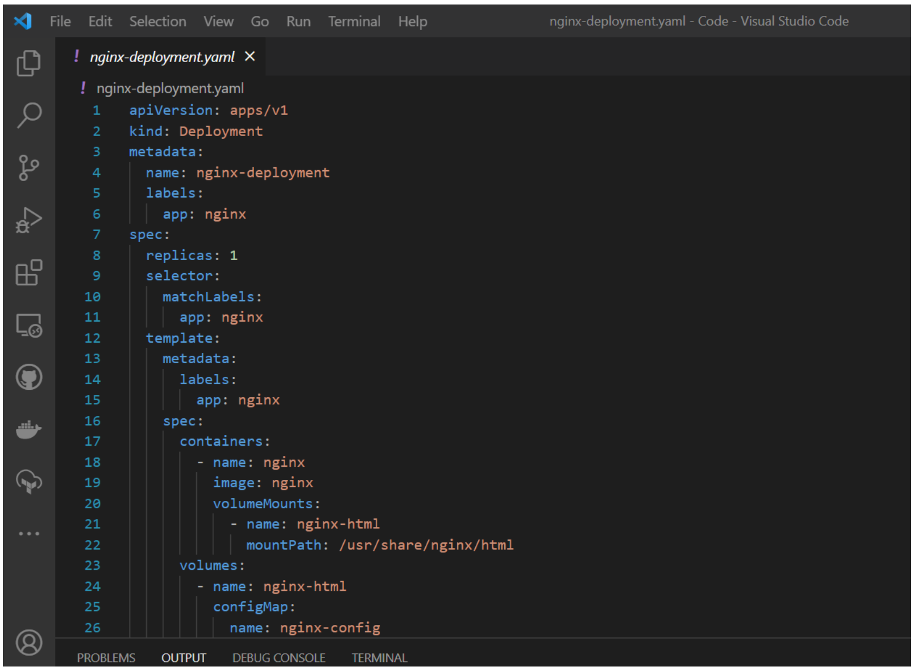
Figure 3: Command to configure Kubectl with cluster details

3.2 Configurations files

While setting up the EKS cluster, three configuration files are used here, named as Deployment.yaml, Service.yaml, and Ingress.yaml.

3.2.1 Deployment.yaml

This file mainly consists of the configurations about the cluster. It specifies the number of replicas, image name of the container, resource requests, and limits. The code snippet is as shown below:



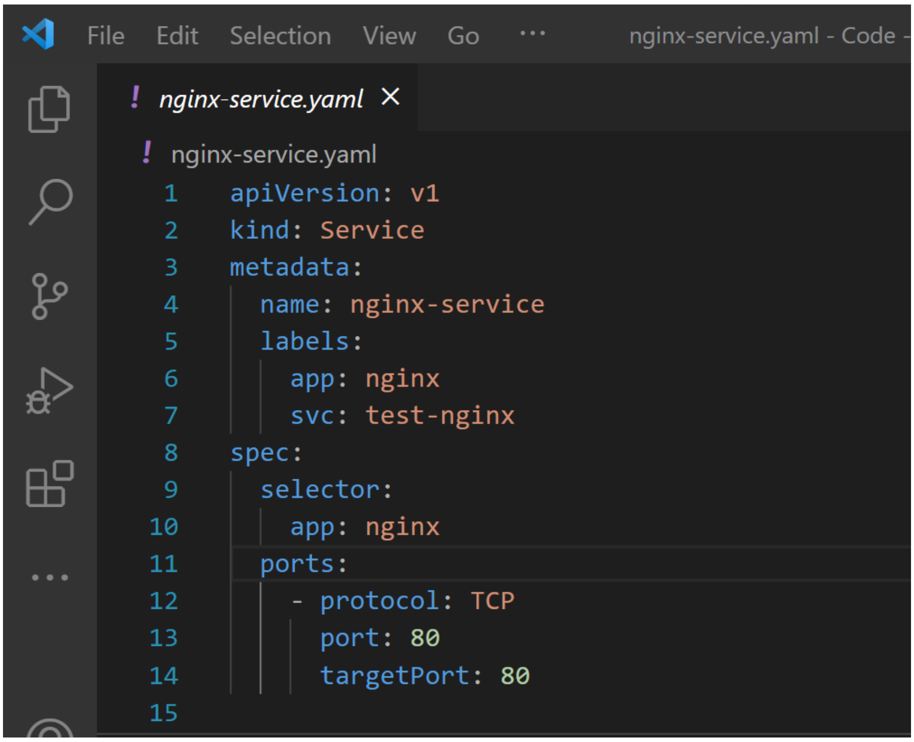
File Edit Selection View Go Run Terminal Help
nginx-deployment.yaml - Code - Visual Studio Code
! nginx-deployment.yaml ×
! nginx-deployment.yaml
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx
          volumeMounts:
            - name: nginx-html
              mountPath: /usr/share/nginx/html
      volumes:
        - name: nginx-html
          configMap:
            name: nginx-config
```

Figure 4: Deployment.yaml configuration file code snippet

3.2.2 Service.yaml

Service resource is defined by Servcie.yaml file. Instead of managing and ,maintaineing each pod's IP ,this file gives an endpoint which can be used by other services and users.

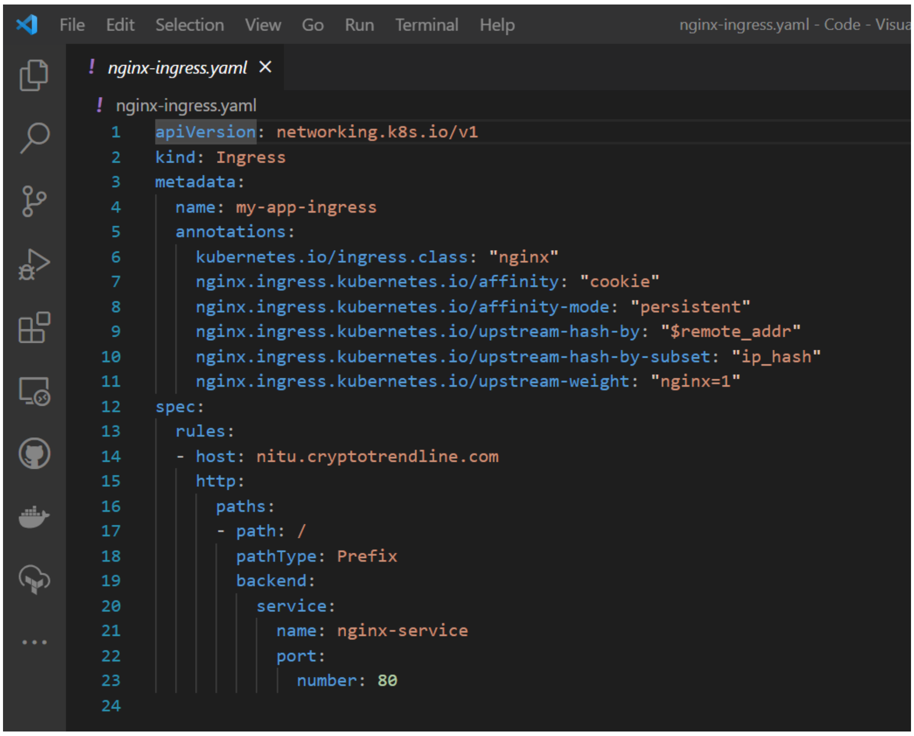


```yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx-service
  labels:
    app: nginx
    svc: test-nginx
spec:
  selector:
    app: nginx
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80

```

Figure 5: Service.yaml configuration file code snippet

3.2.3 Ingress.yaml

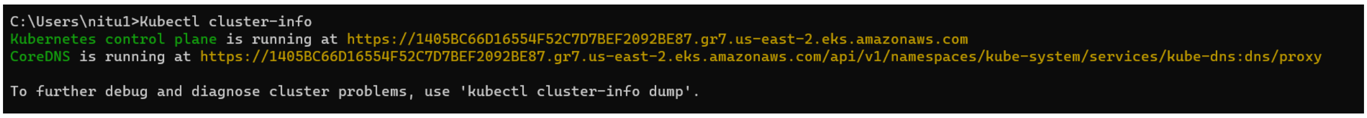
Thus, the configuration file defines the rules that are going to be responsible for routing the traffic and hence balancing the load among the nodes. Annotation is used to specify the two load balancing algorithms - IP Hash and Weighted Round Robin.



```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: my-app-ingress
  annotations:
    kubernetes.io/ingress.class: "nginx"
    nginx.ingress.kubernetes.io/affinity: "cookie"
    nginx.ingress.kubernetes.io/affinity-mode: "persistent"
    nginx.ingress.kubernetes.io/upstream-hash-by: "$remote_addr"
    nginx.ingress.kubernetes.io/upstream-hash-by-subset: "ip_hash"
    nginx.ingress.kubernetes.io/upstream-weight: "nginx=1"
spec:
  rules:
  - host: nitu.cryptotrendline.com
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: nginx-service
            port:
              number: 80

```

Figure 6: Ingress.yaml configuration file code snippet

4 Cluster Information



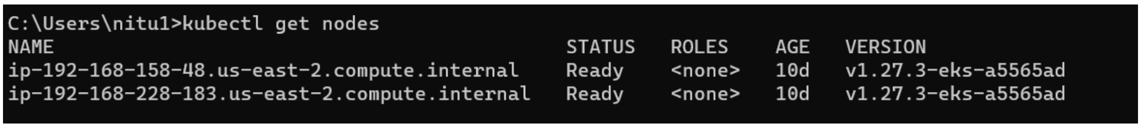
```
C:\Users\nitu1>Kubectl cluster-info
Kubernetes control plane is running at https://1405BC66D16554F52C7D7BEF2092BE87.gr7.us-east-2.eks.amazonaws.com
CoreDNS is running at https://1405BC66D16554F52C7D7BEF2092BE87.gr7.us-east-2.eks.amazonaws.com/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
```

Figure 7: EKS Cluster Information

This command provides information about the Kubernetes control plane, which includes the API server, controller manager, and scheduler. It will also display the cluster's master endpoint's location.

5 EKS Nodes

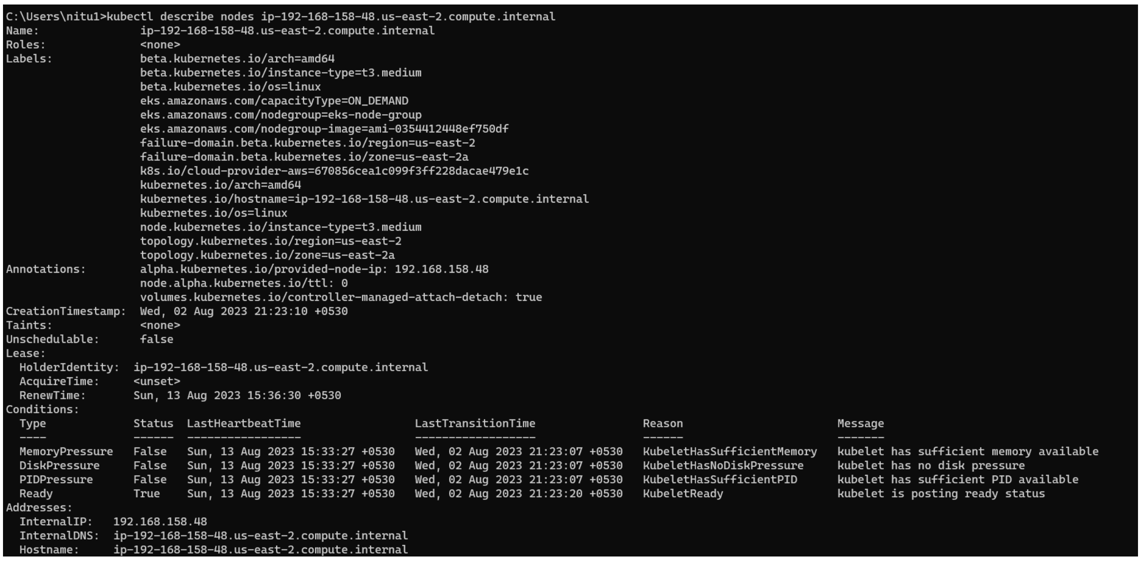
The EKS cluster consists of two nodes. By using the following command, it will display the names and details of the nodes.



```
C:\Users\nitu1>kubectl get nodes
NAME                                           STATUS   ROLES    AGE   VERSION
ip-192-168-158-48.us-east-2.compute.internal   Ready    <none>   10d   v1.27.3-eks-a5565ad
ip-192-168-228-183.us-east-2.compute.internal  Ready    <none>   10d   v1.27.3-eks-a5565ad
```

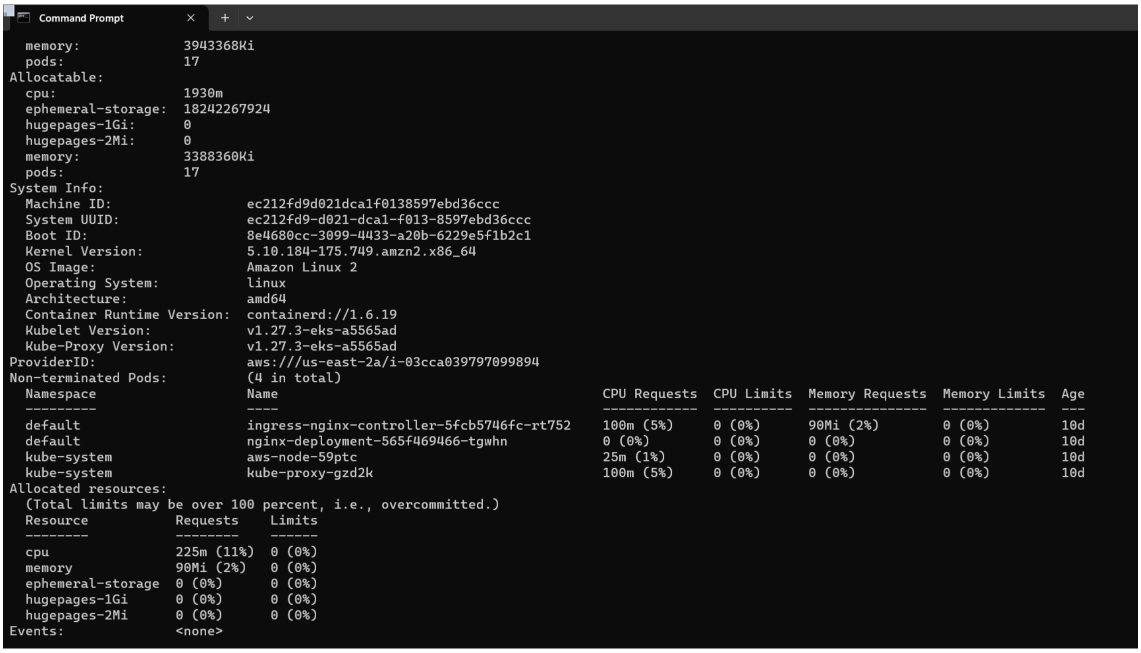
Figure 8: Nodes

And the following command will provide detailed information about the nodes.



```
C:\Users\nitu1>kubectl describe nodes ip-192-168-158-48.us-east-2.compute.internal
Name:               ip-192-168-158-48.us-east-2.compute.internal
Roles:              <none>
Labels:             beta.kubernetes.io/arch=amd64
                    beta.kubernetes.io/instance-type=t3.medium
                    beta.kubernetes.io/os=linux
                    eks.amazonaws.com/capacityType=ON_DEMAND
                    eks.amazonaws.com/nodegroup=eks-node-group
                    eks.amazonaws.com/nodegroup-image=ami-0354412448ef750df
                    failure-domain.beta.kubernetes.io/region=us-east-2
                    failure-domain.beta.kubernetes.io/zone=us-east-2a
                    k8s.io/cloud-provider-aws=670856cea1c099f3ff228dacae479e1c
                    kubernetes.io/arch=amd64
                    kubernetes.io/hostname=ip-192-168-158-48.us-east-2.compute.internal
                    kubernetes.io/os=linux
                    node.kubernetes.io/instance-type=t3.medium
                    topology.kubernetes.io/region=us-east-2
                    topology.kubernetes.io/zone=us-east-2a
Annotations:        alpha.kubernetes.io/provided-node-ip: 192.168.158.48
                    node.alpha.kubernetes.io/ttl: 0
                    volumes.kubernetes.io/controller-managed-attach-detach: true
CreationTimestamp:  Wed, 02 Aug 2023 21:23:10 +0530
Taints:             <none>
Unschedulable:      false
Lease:
  HolderIdentity:  ip-192-168-158-48.us-east-2.compute.internal
  AcquireTime:     <unset>
  RenewTime:       Sun, 13 Aug 2023 15:36:30 +0530
Conditions:
  Type             Status  LastHeartbeatTime                 LastTransitionTime                Reason                       Message
  ----             ------  -----------------                 ------------------                ------                       -------
  MemoryPressure   False   Sun, 13 Aug 2023 15:33:27 +0530   Wed, 02 Aug 2023 21:23:07 +0530   KubeletHasSufficientMemory   kubelet has sufficient memory available
  DiskPressure     False   Sun, 13 Aug 2023 15:33:27 +0530   Wed, 02 Aug 2023 21:23:07 +0530   KubeletHasNoDiskPressure     kubelet has no disk pressure
  PIDPressure      False   Sun, 13 Aug 2023 15:33:27 +0530   Wed, 02 Aug 2023 21:23:07 +0530   KubeletHasSufficientPID      kubelet has sufficient PID available
  Ready            True    Sun, 13 Aug 2023 15:33:27 +0530   Wed, 02 Aug 2023 21:23:20 +0530   KubeletReady                 kubelet is posting ready status
Addresses:
  InternalIP:   192.168.158.48
  InternalDNS:  ip-192-168-158-48.us-east-2.compute.internal
  Hostname:     ip-192-168-158-48.us-east-2.compute.internal
```

Figure 9:

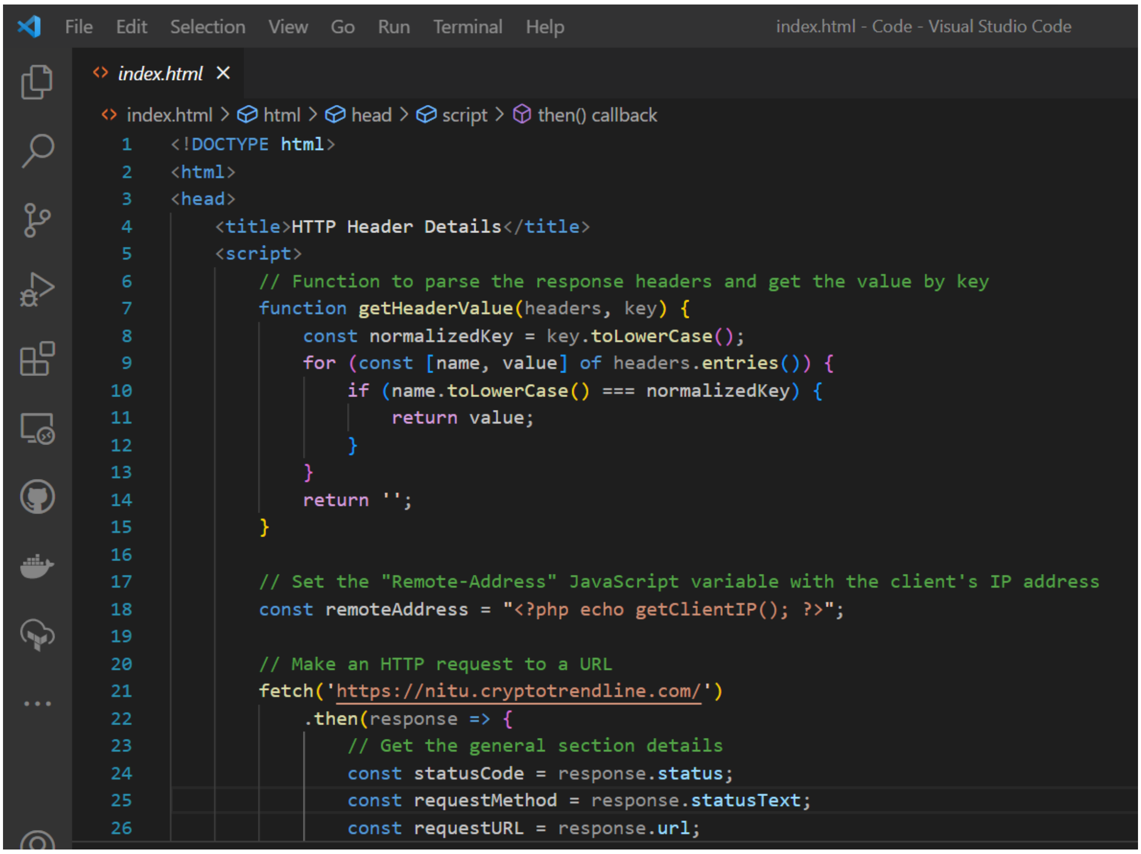


```
  memory:             3943368Ki
  pods:               17
Allocatable:
  cpu:                1930m
  ephemeral-storage:  18242267924
  hugepages-1Gi:      0
  hugepages-2Mi:      0
  memory:             3388360Ki
  pods:               17
System Info:
  Machine ID:                 ec212fd9d021dca1f0138597ebd36ccc
  System UUID:                ec212fd9-d021-dca1-f013-8597ebd36ccc
  Boot ID:                    8e4680cc-3099-4433-a20b-6229e5f1b2c1
  Kernel Version:             5.10.184-175.749.amzn2.x86_64
  OS Image:                   Amazon Linux 2
  Operating System:           linux
  Architecture:               amd64
  Container Runtime Version:  containerd://1.6.19
  Kubelet Version:            v1.27.3-eks-a5565ad
  Kube-Proxy Version:         v1.27.3-eks-a5565ad
ProviderID:                   aws:///us-east-2a/i-03cca039797099894
Non-terminated Pods:          (4 in total)
  Namespace                   Name                                         CPU Requests  CPU Limits  Memory Requests  Memory Limits  Age
  ---------                   ----                                         ------------  ----------  ---------------  -------------  ---
  default                     ingress-nginx-controller-5fcb5746fc-rt752    100m (5%)     0 (0%)      90Mi (2%)        0 (0%)         10d
  default                     nginx-deployment-565f469466-tgwhn            0 (0%)        0 (0%)      0 (0%)           0 (0%)         10d
  kube-system                 aws-node-59ptc                               25m (1%)      0 (0%)      0 (0%)           0 (0%)         10d
  kube-system                 kube-proxy-gzd2k                             100m (5%)     0 (0%)      0 (0%)           0 (0%)         10d
Allocated resources:
  (Total limits may be over 100 percent, i.e., overcommitted.)
  Resource           Requests    Limits
  --------           --------    ------
  cpu                225m (11%)  0 (0%)
  memory             90Mi (2%)   0 (0%)
  ephemeral-storage  0 (0%)      0 (0%)
  hugepages-1Gi      0 (0%)      0 (0%)
  hugepages-2Mi      0 (0%)      0 (0%)
Events:              <none>
```

Figure 10:

6 WebApplication

The web application is written in HTML and JavaScript. The frontend of the web application displays the user's IP address when attempting to access the webpage.



File Edit Selection View Go Run Terminal Help
index.html - Code - Visual Studio Code
index.html
index.html > html > head > script > then() callback

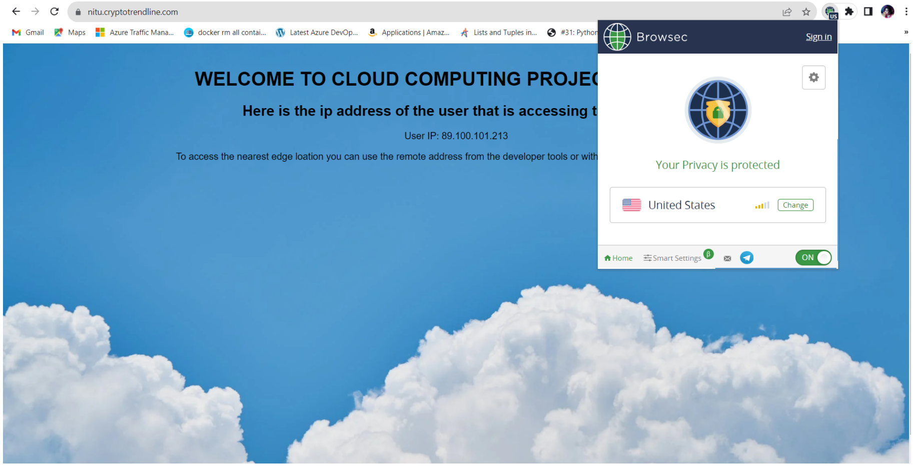
```
<!DOCTYPE html>
<html>
<head>
    <title>HTTP Header Details</title>
    <script>
        // Function to parse the response headers and get the value by key
        function getHeaderValue(headers, key) {
            const normalizedKey = key.toLowerCase();
            for (const [name, value] of headers.entries()) {
                if (name.toLowerCase() === normalizedKey) {
                    return value;
                }
            }
            return '';
        }

        // Set the "Remote-Address" JavaScript variable with the client's IP address
        const remoteAddress = "<?php echo getClientIP(); ?>";

        // Make an HTTP request to a URL
        fetch('https://nitu.cryptotrendline.com/')
            .then(response => {
                // Get the general section details
                const statusCode = response.status;
                const requestMethod = response.statusText;
                const requestURL = response.url;
```

Figure 11: WebApplication code Snippet

By using the below link , webapplication can be accessed.
https://nitu.cryptotrendline.com/

6.1 Setting Up WebApplication

A browser-based VPN was utilized to generate IP addresses from different locations. Subsequently, the web page will showcase the IP address of the user endeavoring to access the web application.



WELCOME TO CLOUD COMPUTING PROJEC

Here is the ip address of the user that is accessing t

User IP: 89.100.101.213

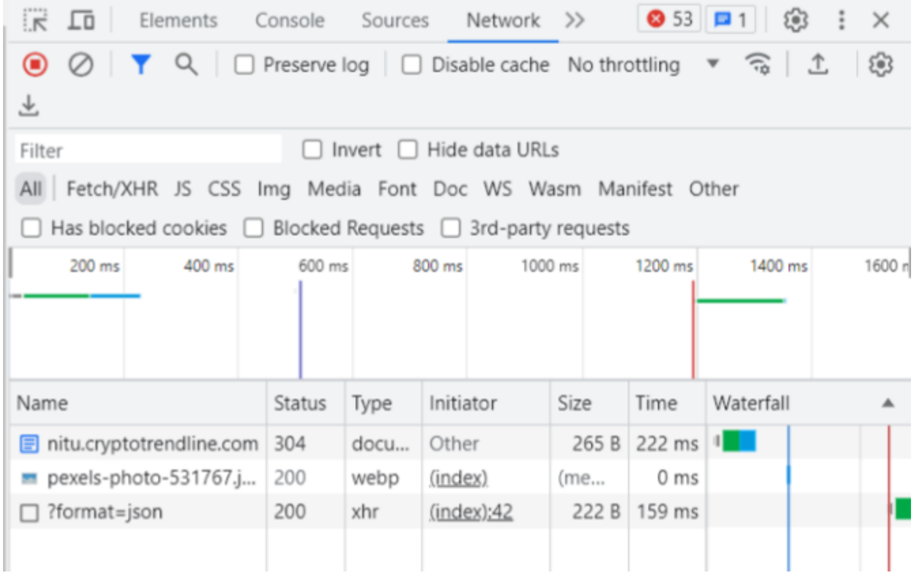
To access the nearest edge loation you can use the remote address from the developer tools or with

Figure 12: WebApplication

7 Observation

7.1 Initial Observation

The first experiment generated from East Region of the United States. The response time 222 milliseconds during this initial access was recorded.

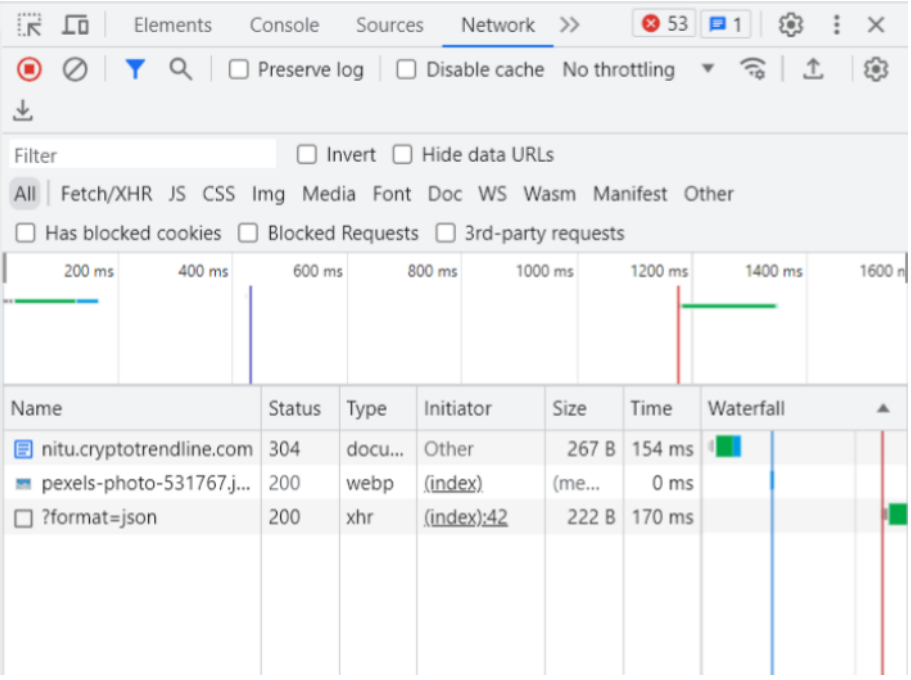


Name	Status	Type	Initiator	Size	Time	Waterfall
nitu.cryptotrendline.com	304	docu...	Other	265 B	222 ms	
pexels-photo-531767.j...	200	webp	(index)	(me...	0 ms	
?format=json	200	xhr	(index):42	222 B	159 ms	

Figure 13: Response Time when WebApplication Accessed for the first time from US-East-2 Region

7.2 Second Observation

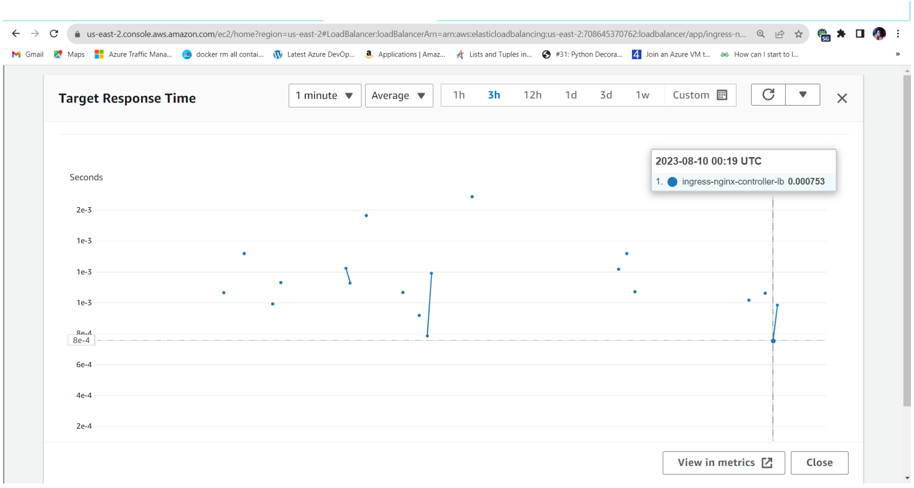
The Content Delivery Network (CDN) is utilized in Experiment 2. To optimize content delivery and reduce latency, Kubernetes was integrated with a content delivery network (CDN) with a Load Balancing algorithm. IPhash and Weighted Round Robin are used alongside AWS CloudFront. The impact of CDN integration on response times and system performance as a whole was investigated. Observed response time is 154 ms.



Name	Status	Type	Initiator	Size	Time	Waterfall
nitu.cryptotrendline.com	304	docu...	Other	267 B	154 ms	
pexels-photo-531767.j...	200	webp	(index)	(me...	0 ms	
?format=json	200	xhr	(index):42	222 B	170 ms	

Figure 14: Response Time WebApplication Accessed for the second time from US-East-2 Region

7.3 Observation on AWS Cloudwatch

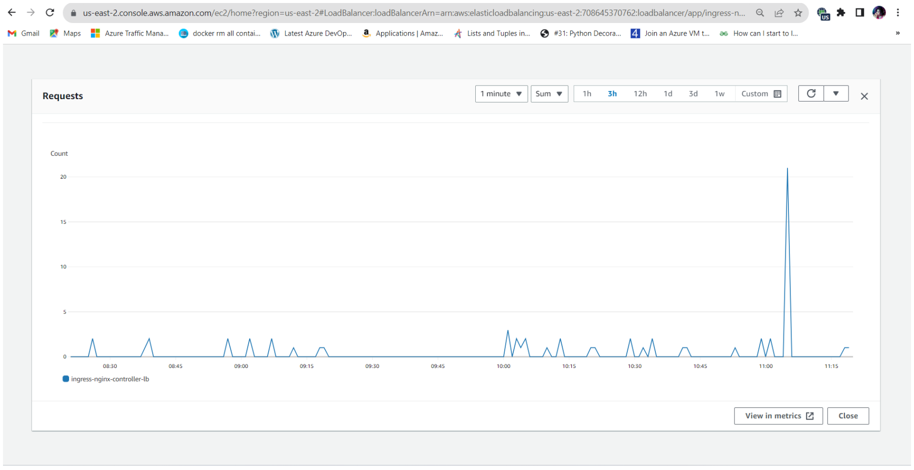

us-east-2.console.aws.amazon.com/ec2/home?region=us-east-2#LoadBalancer:loadBalancerArn=arn:aws:elasticloadbalancing:us-east-2:708645370762:loadbalancer/app/ingress-n...

Target Response Time

1 minute · Average · 3h

2023-08-10 00:19 UTC
1. ingress-nginx-controller-lb 0.000753

Figure 15: Target Response Time of Application Load Balancer Captured on CloudWatch



us-east-2.console.aws.amazon.com/ec2/home?region=us-east-2#LoadBalancer:loadBalancerArn=arn:aws:elasticloadbalancing:us-east-2:708645370762:loadbalancer/app/ingress-n...

Requests

1 minute ▼ Sum ▼ 1h 3h 12h 1d 3d 1w Custom

Figure 16: Number of Requests Captured on Application Load Balancer in CloudWatch

References

Algarni, B. A. M. (2021). Managing deployed containerized web application on aws using eks on aws fargate. `https://www.rit.edu/ischoolprojects/sites/rit.edu.ischoolprojects/files/document_library/Bashair_Algarni_Capstone_finalReport.pdf`.

Getting started with Amazon EKS (Year).
URL: *https://docs.aws.amazon.com/eks/latest/userguide/getting-started.html*