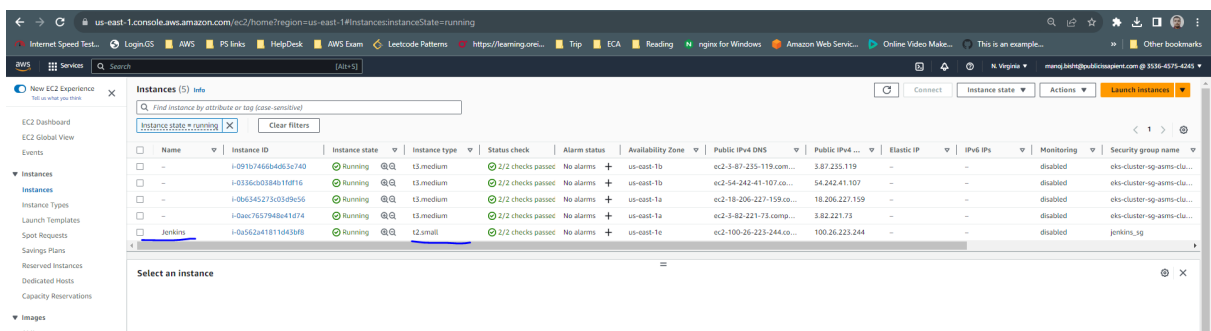
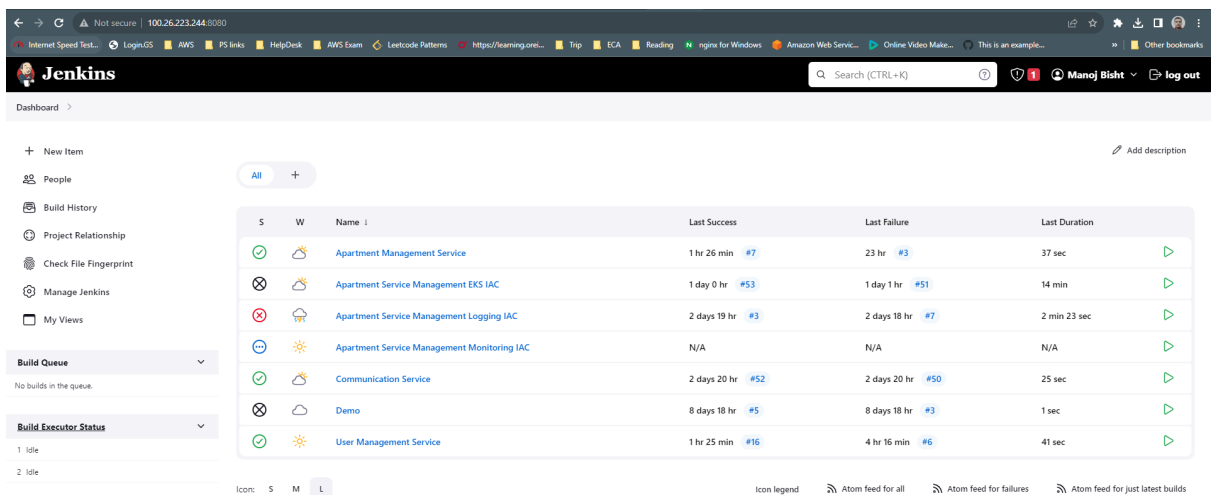
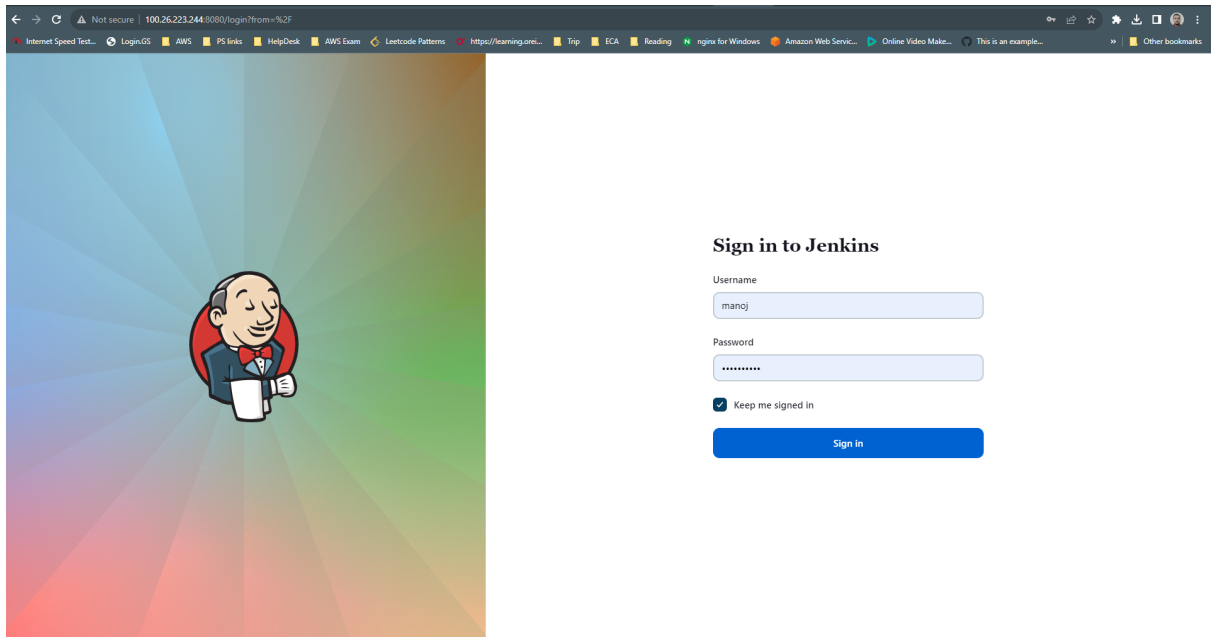


Jenkins :-

On AWS EC2 instance



Terraform Code to Deploy Jenkins on EC2 from my local machine

<https://tools.publicis.sapient.com/bitbucket/projects/EOB/repos/manoj-bisht/browse/apartment-service-management-jenkins-iac>

```
2 sudo yum update -y
3 |
4 #Download Jenkins
5 sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo
6 sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
7
8 #Install Java
9 sudo yum upgrade -y
10 sudo dnf install java-11-amazon-corretto
11
12 #Install Git
13 yum install git -y
14 # which git
15
16 #Install Maven
17 sudo yum install wget maven -y
18 # mvn -version
19
20 #Install Jenkins
21 sudo yum install jenkins -y
22 sudo systemctl enable jenkins
23 sudo systemctl start jenkins
24 # sudo systemctl status jenkins
25
26 #Install Docker
27 sudo yum install -y docker
28 sudo systemctl enable docker
29 sudo systemctl start docker
30 # sudo systemctl status docker
31
32 #Add user with docker
33 sudo usermod -a -G docker jenkins
34 sudo chmod 777 /var/run/docker.sock
35
36 #kubect1
37 sudo curl -LO "https://dl.k8s.io/release/${curl -L -s https://dl.k8s.io/release/stable.txt}/bin/linux/amd64/kubect1"
38 sudo install -o root -g root -m 0755 kubect1 /usr/local/bin/kubect1
39 # kubect1 version --client --output=yaml
40
41 #Helm
42 sudo curl -fsSL -o get_helm.sh https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3
43 sudo chmod 700 get_helm.sh
44 sudo ./get_helm.sh
45 # helm version
46
47 #Terraform install
48 sudo yum install -y yum-utils
49 sudo yum-config-manager --add-repo https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo
50 sudo yum -y install terraform
51 # terraform -v
```

```
1 terraform {
2   required_providers {
3     aws = {
4       source = "hashicorp/aws"
5       version = ">= 5.0"
6     }
7   }
8 }
9
10 provider "aws" {
11   region = "us-east-1"
12 }
13
14 resource "aws_security_group" "jenkins_sg" {
15   name = "jenkins_sg"
16   description = "Allow ssh and HTTP traffic"
17   vpc_id = "vpc-066f579aa552f1bb3"
18
19   ingress {
20     from_port = 22
21     to_port = 22
22     protocol = "tcp"
23     cidr_blocks = ["0.0.0.0/0"]
24   }
25
26   ingress {
27     from_port = 8080
28     to_port = 8080
29     protocol = "tcp"
30     cidr_blocks = ["0.0.0.0/0"]
31   }
32
33   ingress {
34     from_port = 443
35     to_port = 443
36     protocol = "tcp"
37     cidr_blocks = ["0.0.0.0/0"]
38   }
39
40   egress {
41     from_port = 0
42     to_port = 0
43     protocol = "-1"
44     cidr_blocks = ["0.0.0.0/0"]
45   }
46 }
47
```

EKS Cluster

EKS cluster deploy on AWS using terraform and Jenkins Pipeline

Jenkins Pipeline for EKS

The screenshot shows the Jenkins Pipeline for EKS. The pipeline is titled "Pipeline Apartment Service Management EKS IAC". The stage view shows the following stages and their durations:

Stage	Declarative: Checkout SCM	Checkout	Terraform init	Terraform format	Terraform validate	Terraform plan	Terraform action
Stage 1	435ms	363ms	2s	333ms	30s	48s	17min 47s
Stage 2	401ms	345ms	3s	354ms	32s	39s	117ms
Stage 3	477ms	310ms	2s	331ms	26s	38s	13min 23s
Stage 4	393ms	388ms	2s	321ms	32s	46s	4min 32s
Stage 5	452ms	319ms	3s	312ms	32s	38s	30min 36s

The build history shows the following builds:

Build	Build Time	Status
514	02-Sep-2023 11:47 am	Success
513	02-Sep-2023 11:20 am	Success
512	02-Sep-2023 10:47 am	Success
511	01-Sep-2023 9:41 am	Success
510		Success

EKS Cluster

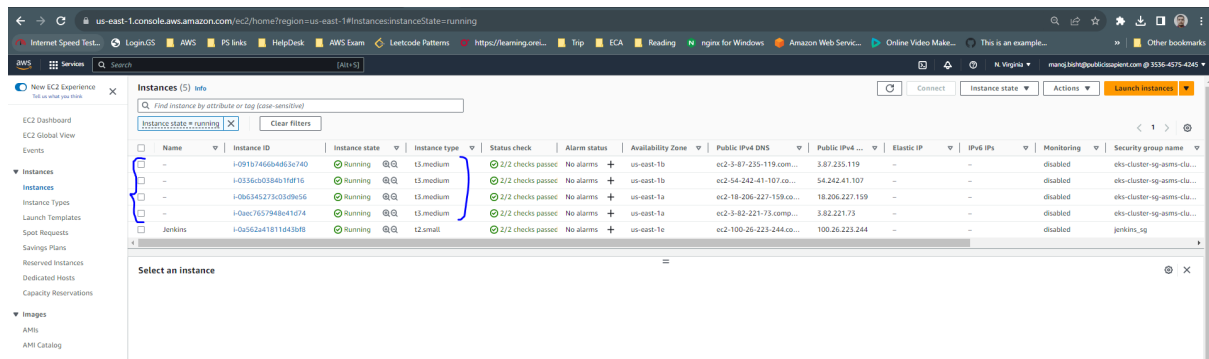
The screenshot shows the Amazon Elastic Kubernetes Service console for the "asms-cluster". The cluster is in the "Active" state. The console displays the following information:

- Cluster info:** Kubernetes version 1.27, Provider EKS.
- Nodes (4) info:** A table showing 4 nodes, all in "ready" status.
- Node groups (1) info:** A table showing 1 node group, "asms-cluster-node-group", with a desired size of 4.

Node name	Instance type	Node group	Created	Status
ip-172-31-21-118.ec2.internal	t3.medium	asms-cluster-node-group	Created 19 hours ago	ready
ip-172-31-26-50.ec2.internal	t3.medium	asms-cluster-node-group	Created 19 hours ago	ready
ip-172-31-81-6.ec2.internal	t3.medium	asms-cluster-node-group	Created 19 hours ago	ready
ip-172-31-85-87.ec2.internal	t3.medium	asms-cluster-node-group	Created 19 hours ago	ready

Group name	Desired size	AMI release version	Launch template	Status
asms-cluster-node-group	4	1.27.4-20230825	-	Active

Managed EC2 Instances



Terraform Code to deploy EKS

<https://tools.publicis.sapien.com/bitbucket/projects/EOB/repos/manoj-bisht/browse/apartment-service-management-eks-iac>

Jenkins Pipeline Code for EKS

<https://tools.publicis.sapien.com/bitbucket/projects/EOB/repos/manoj-bisht/browse/apartment-service-management-eks-iac/Jenkinsfile>

```
1 pipeline {
2   agent any
3
4   stages {
5     stage('Checkout') {
6       steps {
7         checkout changelog: false, poll: false, scm: scmGit(branches: [[name: 'main']], extensions: [], userRemoteConfigs: [[credentialId: 'Token', url: 'https://pscode.lioncloud.net/apartment-service-management-solution/apartment-service-management-eks-iac'])
8       }
9     }
10    stage('Terraform init') {
11      steps {
12        sh 'terraform init'
13      }
14    }
15    stage('Terraform format') {
16      steps {
17        sh 'terraform fmt'
18      }
19    }
20    stage('Terraform validate') {
21      steps {
22        sh 'terraform validate'
23      }
24    }
25    stage('Terraform plan') {
26      steps {
27        sh 'terraform plan'
28      }
29    }
30    stage('Terraform action') {
31      input {
32        message 'Please select terraform action'
33        id 'actionId'
34        ok 'Submit'
35        submitterParameter 'approverId'
36        parameters {
37          choice choices: ['apply', 'destroy'], name: 'action'
38        }
39      }
40      steps {
41        sh 'terraform $(action) --auto-approve'
42      }
43    }
44  }
45 }
```

ELK Stack for logging

I use Elastic Search, Kibana and File beat for this.

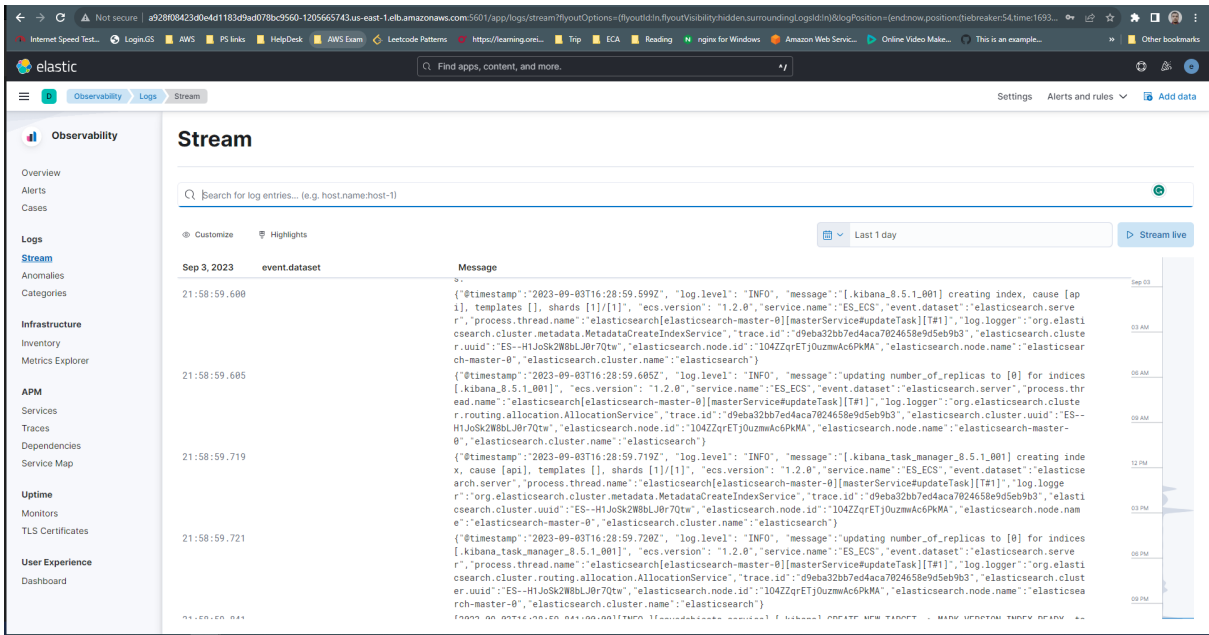
Code Repo:

<https://tools.publicis.sapient.com/bitbucket/projects/EOB/repos/manoj-bisht/browse/apartment-service-management-logging-and-monitoring-iac>

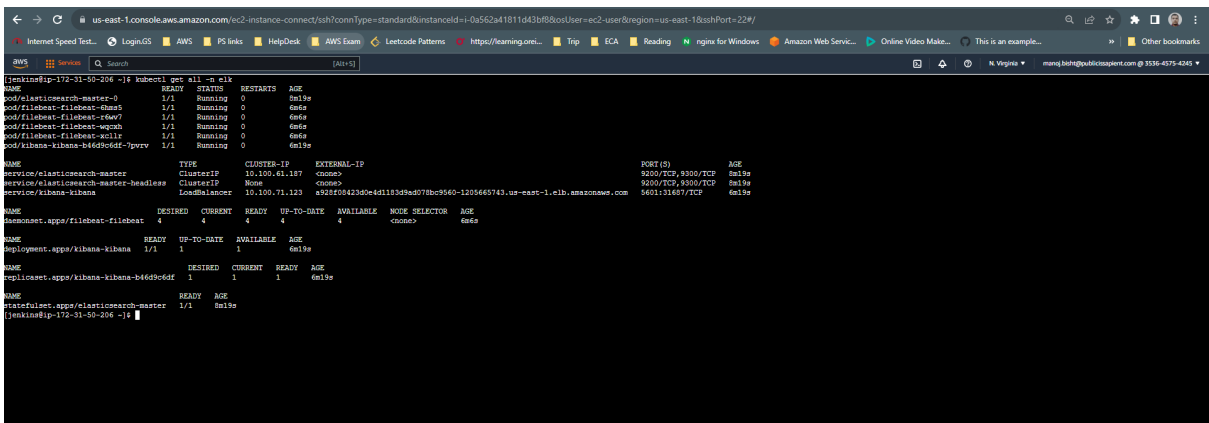
Jenkins Pipeline:

<https://tools.publicis.sapient.com/bitbucket/projects/EOB/repos/manoj-bisht/browse/apartment-service-management-logging-and-monitoring-iac/Jenkinsfile>

Kibana:



ELK Nodes:



Prometheus and Grafana:

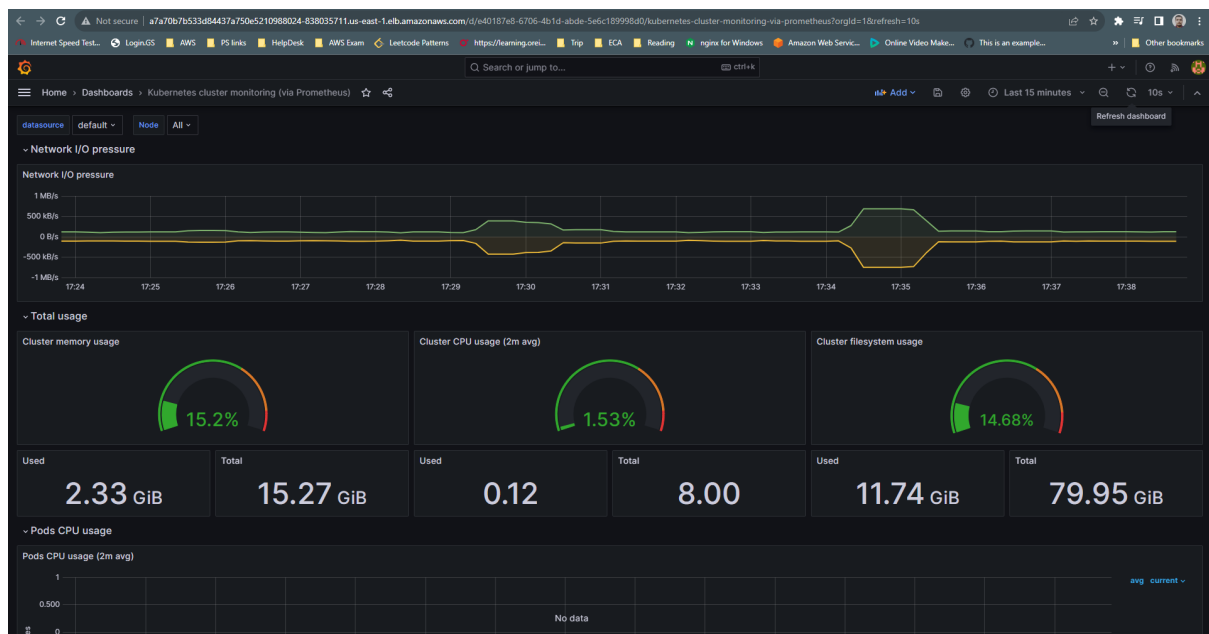
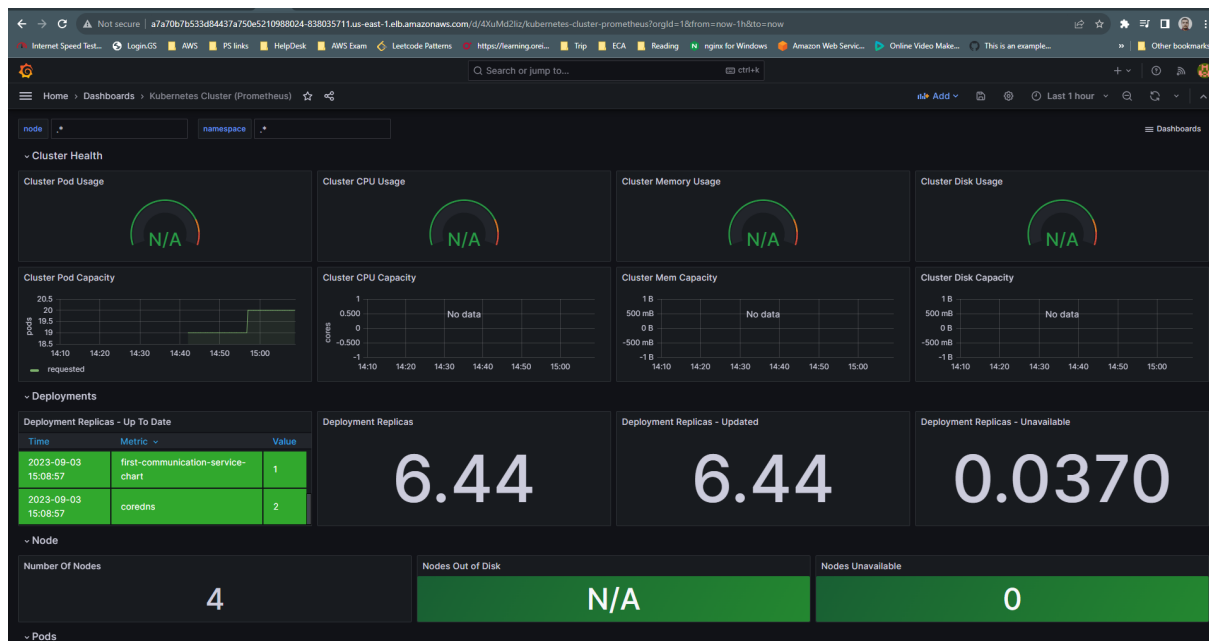
Code Repo:

<https://tools.publicis.sapien.com/bitbucket/projects/EOB/repos/manoj-bisht/browse/apartment-service-management-logging-and-monitoring-iac/monitoring>

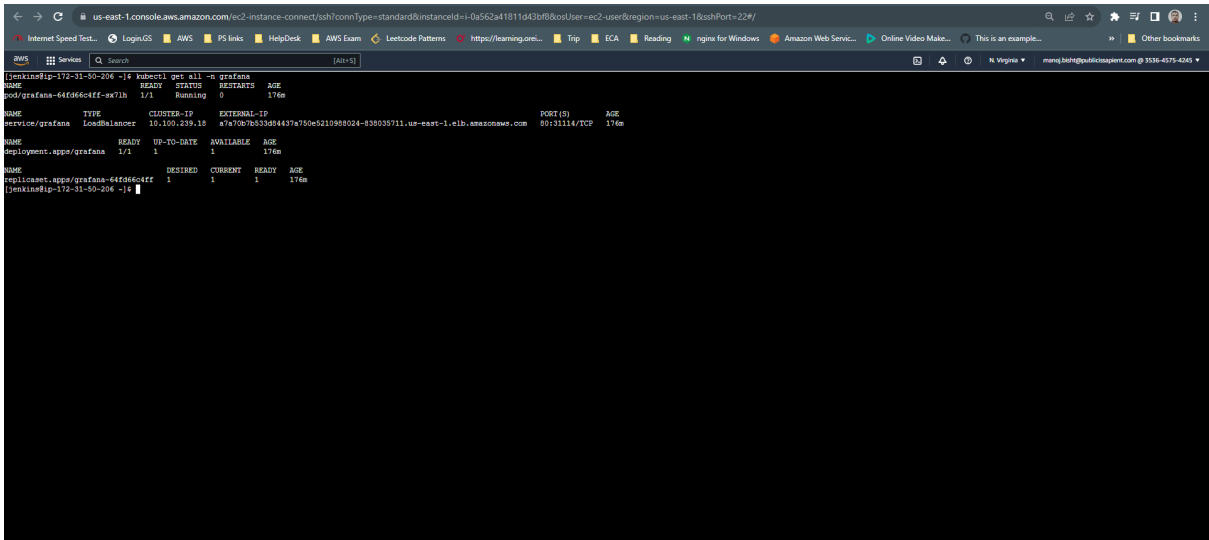
Jenkins Pipeline:

<https://tools.publicis.sapien.com/bitbucket/projects/EOB/repos/manoj-bisht/browse/apartment-service-management-logging-and-monitoring-iac/monitoring/Jenkinsfile>

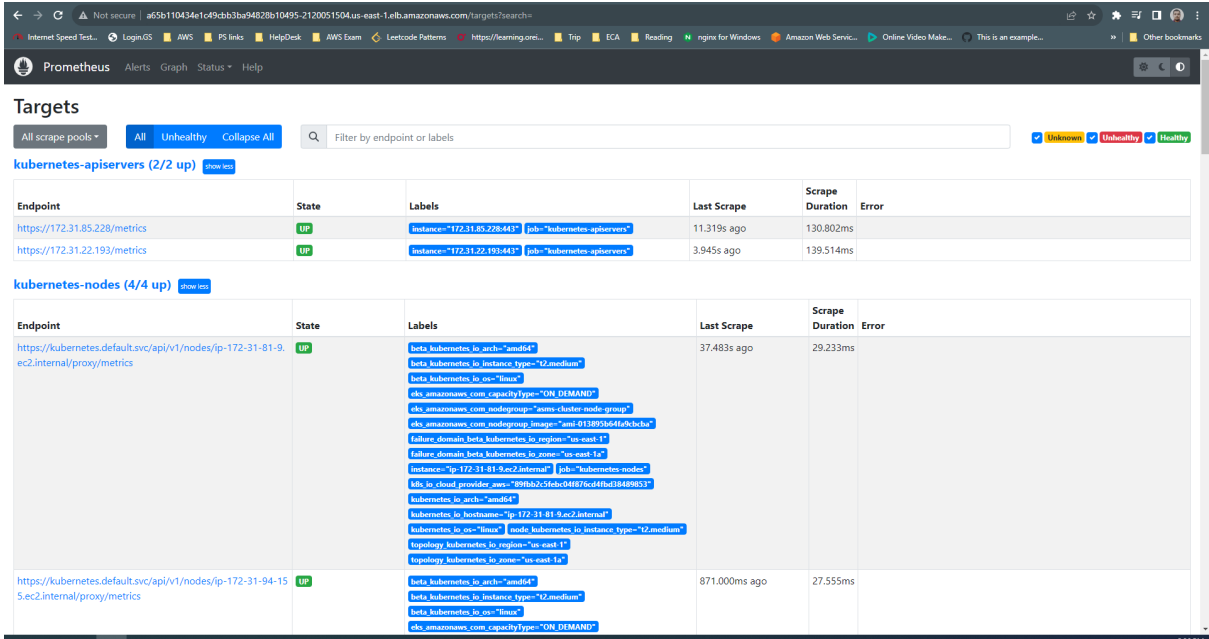
Grafana GUI:



Grafana Status:



Prometheus GUI:



Prometheus Status:

```
us-east-1.console.aws.amazon.com/ec2-instance-connect?sshConnType=standard&instanceId=i-0a562a41811d43b8&osUser=ec2-user&region=us-east-1&sshPort=22&

Internet Speed... LoginGifs AWS PS Keys HelpDesk AWS Exam LeetCode Patterns https://fearning.com/ Tip ECA Reading ngining for Windows Amazon Web Services... Online Video Make... This is an example... Other bookmarks

AWS Services Search [Alt+F]

Amazon Linux 2023

https://aws.amazon.com/linux/amazon-linux-2023

Last login: Sun Sep 3 08:15:34 2023 from 18.204.107.28
[ec2-user@ip-172-31-50-206 ~]$ sudo su - jenkins
Last login: Sun Sep 3 08:15:48 UTC 2023 on pts/0
[jenkins@ip-172-31-50-206 ~]$ kubectl get all -n prometheus

NAME                                READY    STATUS    RESTARTS   AGE
pod/prometheus-alertmanager-0       0/1      Pending   0           167m
pod/prometheus-kube-state-metrics-b4557d966-4460n  1/1      Running   0           167m
pod/prometheus-prometheus-node-exporter-8lrvr     1/1      Running   0           167m
pod/prometheus-prometheus-node-exporter-9p1a9     1/1      Running   0           167m
pod/prometheus-prometheus-node-exporter-47fba     1/1      Running   0           167m
pod/prometheus-prometheus-node-exporter-muab      1/1      Running   0           167m
pod/prometheus-prometheus-pushgateway-79fz799469-ctwt  1/1      Running   0           167m
pod/prometheus-server-695c7bc47-vj5c2             2/2      Running   0           167m

NAME                                TYPE                CLUSTER-IP          EXTERNAL-IP          PORT(S)          AGE
service/prometheus-alertmanager     ClusterIP            10.100.145.155      <none>                9093/TCP          167m
service/prometheus-alertmanager-headless ClusterIP            None                <none>                <none>            167m
service/prometheus-kube-state-metrics ClusterIP            10.100.123.15     <none>                8080/TCP          167m
service/prometheus-prometheus-node-exporter ClusterIP            10.100.61.7       <none>                9100/TCP          167m
service/prometheus-prometheus-pushgateway ClusterIP            10.100.193.111    <none>                9091/TCP          167m
service/prometheus-server            LoadBalancer        10.100.250.93     8080310461c49dc8ba9420b10495-2120051504.us-east-1.elb.amazonaws.com 80 31311/TCP 167m

NAME                                DESIRED    CURRENT    UP-TO-DATE    AVAILABLE    AGE
deployment.apps/prometheus-prometheus-node-exporter 4          4          4             4            167m
pod-template-hash: 167m
kubelet: 167m
kubernetes.io/os: linux 167m

NAME                                READY    UP-TO-DATE    AVAILABLE    AGE
deployment.apps/prometheus-kube-state-metrics       1/1      1             1            167m
deployment.apps/prometheus-prometheus-pushgateway  1/1      1             1            167m
deployment.apps/prometheus-server                   1/1      1             1            167m

NAME                                DESIRED    CURRENT    READY    AGE
replicaset.apps/prometheus-kube-state-metrics-b4557d966 1          1          1        167m
replicaset.apps/prometheus-prometheus-pushgateway-79fz799469 1          1          1        167m
replicaset.apps/prometheus-server-695c7bc47           1          1          1        167m

NAME                                READY    AGE
statefulset.apps/prometheus-alertmanager              0/1      167m
statefulset.apps/prometheus-kube-state-metrics-b4557d966 1/1      167m
statefulset.apps/prometheus-prometheus-pushgateway-79fz799469 1/1      167m
statefulset.apps/prometheus-server-695c7bc47           1/1      167m
statefulset.apps/prometheus-server-695c7bc47-vj5c2     2/2      167m
```


Application Services:

← → ↺ us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?connType=standard&instanceId=i-0a562a41811d43bf8&osUser=ec2-user®ion=us-east-1&sshPort=22#/Internet Speed Test... Login.GS AWS PS links HelpDesk AWS Exam Leetcode Patterns https://learning.orei... Trip ECA Reading nginx for Windows Amazon Web Servic...

AWS Services Search [Alt+S]

Every 2.0s: kubectl get all -n asms-deployment

NAME	READY	STATUS	RESTARTS	AGE
pod/first-apartment-management-service-chart-7cf5b4cb84-q4w7w	1/1	Running	0	57m
pod/first-asm-apartment-management-service-chart-64c947bd9c-r925f	1/1	Running	0	53m
pod/first-ums-user-management-service-chart-68fd49fd4-vs5ol	1/1	Running	0	52m

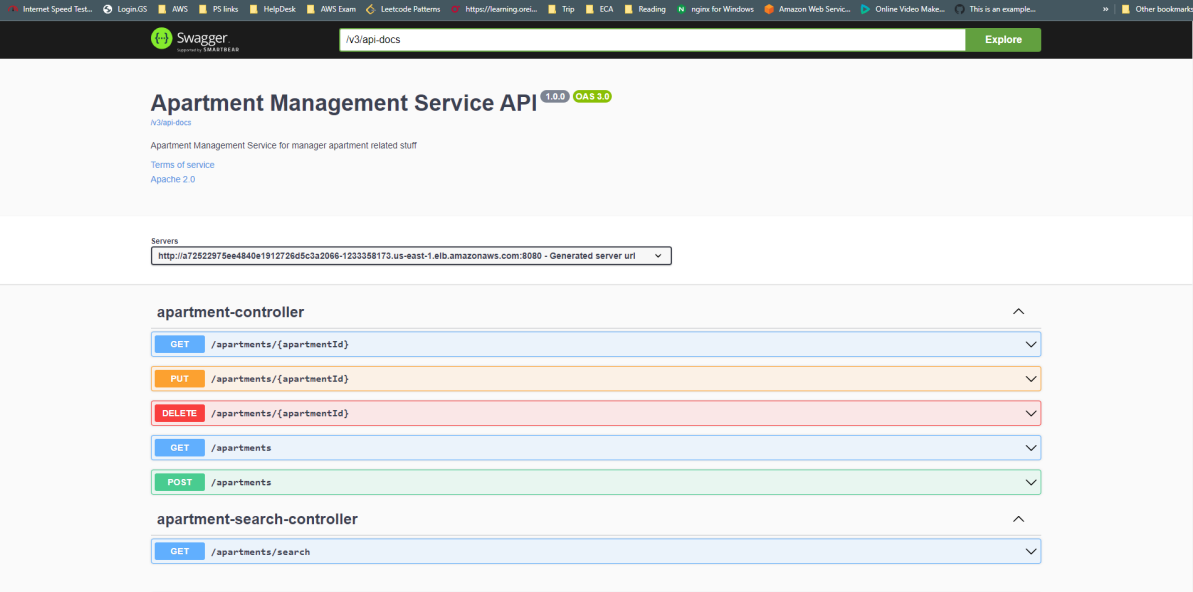
NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
service/first-apartment-management-service-chart	LoadBalancer	10.100.245.134	a0db0d0f23f834a4db67ddb162e4b0bf-1179289413.us-east-1.elb.amazonaws.com	8080:32626/TCP	57m
service/first-asm-apartment-management-service-chart	LoadBalancer	10.100.198.255	a72522975ee4840e1912726d5c3a2066-1233358173.us-east-1.elb.amazonaws.com	8080:30171/TCP	53m
service/first-ums-user-management-service-chart	LoadBalancer	10.100.57.29	a61f2853a21e64909a4a20e1b9d27e74-1746013164.us-east-1.elb.amazonaws.com	8081:31132/TCP	52m

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
deployment.apps/first-apartment-management-service-chart	1/1	1	1	57m
deployment.apps/first-asm-apartment-management-service-chart	1/1	1	1	53m
deployment.apps/first-ums-user-management-service-chart	1/1	1	1	52m

NAME	DESIRED	CURRENT	READY	AGE
replicaset.apps/first-apartment-management-service-chart-7cf5b4cb84	1	1	1	57m
replicaset.apps/first-asm-apartment-management-service-chart-64c947bd9c	1	1	1	53m
replicaset.apps/first-ums-user-management-service-chart-68fd49fd4	1	1	1	52m

Apartment Management Service

<https://tools.publicis.sapient.com/bitbucket/projects/EOB/repos/manoj-bisht/browse/apartment-management-service>



The image shows the Swagger UI for the Apartment Management Service API. The interface is clean and modern, with a dark header bar containing the Swagger logo and a search bar. The main content area is white and displays the API's title, version (1.0.0), and a description. Below this, there's a section for servers, followed by a list of endpoints grouped under two controllers: 'apartment-controller' and 'apartment-search-controller'. Each endpoint is represented by a colored bar indicating its HTTP method (GET, PUT, DELETE, POST) and the corresponding URL path. The 'apartment-controller' has five endpoints, and the 'apartment-search-controller' has one. A 'Schemas' section is partially visible at the bottom.

Swagger
v3/api-docs

Apartment Management Service API

1.0.0 OAS 3.0

Apartment Management Service for manager apartment related stuff

[Terms of service](#)

[Apache 2.0](#)

Servers

<http://a72522975ee4840e1912726d5c3a2066-1233358173.us-east-1.elb.amazonaws.com:8080> - Generated server url

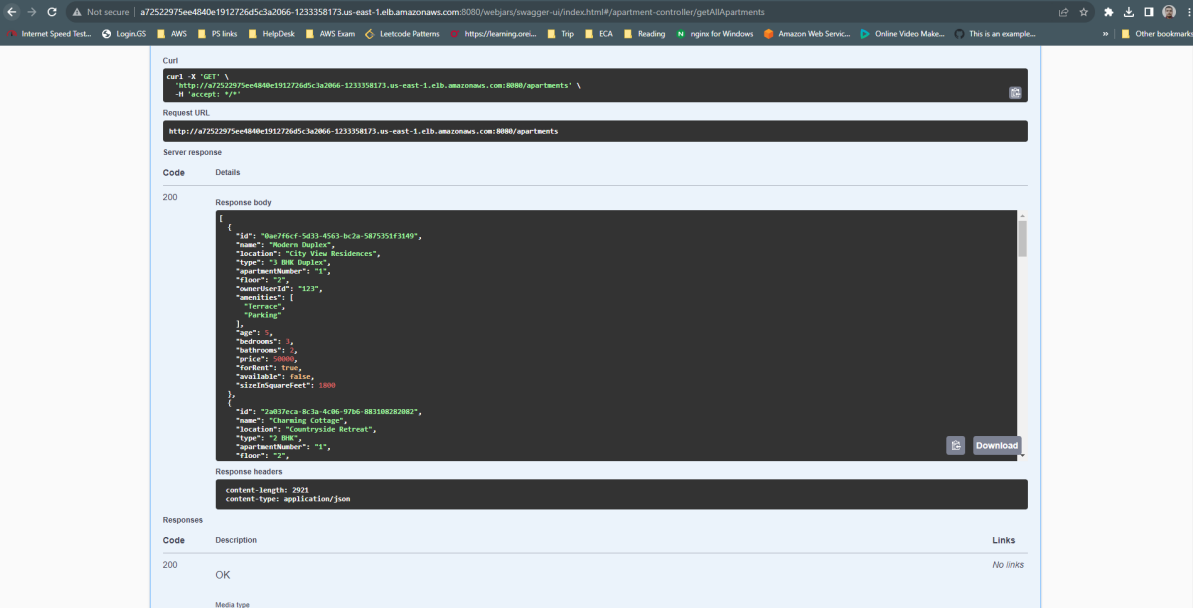
apartment-controller

- GET /apartments/{apartmentId}
- PUT /apartments/{apartmentId}
- DELETE /apartments/{apartmentId}
- GET /apartments
- POST /apartments

apartment-search-controller

- GET /apartments/search

Schemas



The image shows the Swagger UI displaying the details of a successful GET request to the /apartments endpoint. The 'Curl' section shows the command used to make the request. The 'Request URL' is the full URL. The 'Server response' section shows the status code 200. The 'Response body' is a JSON array containing two apartment objects. The 'Response headers' section shows the content-length and content-type. The 'Responses' section shows the status code 200 and the description 'OK'.

curl -X GET \n http://a72522975ee4840e1912726d5c3a2066-1233358173.us-east-1.elb.amazonaws.com:8080/apartments \n -H 'accept: */*' \n

Request URL

<http://a72522975ee4840e1912726d5c3a2066-1233358173.us-east-1.elb.amazonaws.com:8080/apartments>

Server response

Code Details

200

Response body

```
{\n  "id": "b8e7f6c9-5d33-4563-bc2a-5875351f3149",\n  "name": "Modern Duplex",\n  "location": "City View Residences",\n  "type": "BHK Duplex",\n  "apartmentNumber": "1",\n  "floor": 2,\n  "ownerId": "123",\n  "amenities": {\n    "furniture": true,\n    "parking": true\n  },\n  "age": 5,\n  "bedrooms": 3,\n  "bathrooms": 2,\n  "price": 50000,\n  "forRent": true,\n  "available": false,\n  "sizeInSquarefeet": 1800\n},\n{\n  "id": "2ab0feca-8c3a-4c0b-97b6-88108d2d2082",\n  "name": "Charming Cottage",\n  "location": "Countryside Retreat",\n  "type": "BHK",\n  "apartmentNumber": "1",\n  "floor": 2\n}
```

Response headers

content-length: 2928
content-type: application/json

Responses

Code	Description	Links
200	OK	No links

Media type

/

User Management Service:

<https://tools.publicis.sapient.com/bitbucket/projects/EOB/repos/manoj-bisht/browse/user-management-service>

The screenshot shows the Swagger UI for the 'User Management Service API'. The browser address bar displays a URL from an AWS S3 bucket. The Swagger header includes the 'Swagger' logo and a search bar containing '/v3/api-docs'. The main content area features the API title 'User Management Service API' with version '1.0.0' and 'OAS 3.0' tags. Below the title, there are links for 'v3/api-docs', 'User Management Service for user creation and authentication', 'Terms of service', and 'Apache 2.0'. A 'Servers' section shows a dropdown menu with the selected server URL: 'http://a61f2853a21e64909a4a20e1b9d27e74-1746013164.us-east-1.elb.amazonaws.com:8081 - Generated server url'. The API endpoints are organized into two sections: 'user-controller' and 'authentication-controller'. The 'user-controller' section lists five endpoints: GET /user/{email}, PUT /user/{email}, DELETE /user/{email}, POST /user, and GET /user/id/{id}. The 'authentication-controller' section lists two endpoints: POST /auth/validate-token and POST /auth/otp. Each endpoint is represented by a colored bar with its HTTP method and path, and a dropdown arrow on the right.

Swagger
v3/api-docs Explore

User Management Service API 1.0.0 OAS 3.0

[v3/api-docs](#)
User Management Service for user creation and authentication
[Terms of service](#)
[Apache 2.0](#)

Servers
http://a61f2853a21e64909a4a20e1b9d27e74-1746013164.us-east-1.elb.amazonaws.com:8081 - Generated server url

user-controller

- GET /user/{email}
- PUT /user/{email}
- DELETE /user/{email}
- POST /user
- GET /user/id/{id}

authentication-controller

- POST /auth/validate-token
- POST /auth/otp

Docker:

<https://tools.publicis.sapient.com/bitbucket/projects/EQB/repos/manoj-bisht/browse/user-management-service/Dockerfile>

Source view Diff to previous History 155 B

```
1 FROM openjdk:17
2 VOLUME /tmp
3 EXPOSE 8080
4 ARG JAR_FILE=target/user-management-service-1.0.0.jar
5 ADD ${JAR_FILE} app.jar
6 ENTRYPOINT ["java","-jar","/app.jar"]
```