

Event Sphere

Mythreya Hardur Madhukeshwara - 121307484

Kunal Gadgil - 121058843

Sai Saketh Bolla - 121393579

Shengjie Liu - 117102260

Baelul Haile - 116451246

Darpan Nagendra Karur - 120481420

Github:

<https://github.com/containerization-team-cems-eks/EventSphere>

Table of Contents

Table of Contents.....	2
Introduction.....	3
Architecture & Design.....	4
Infrastructure-as-Code.....	5
Docker and Kubernetes Configuration.....	8
Load Balancing & Networking.....	23
CI/CD Pipeline.....	26
Collaboration and Version Control.....	27
Observability.....	28
Innovation.....	42
Reflection.....	53

Introduction

EventSphere is a Campus event management system that helps manage campus events like lectures, workshops, club activities, student programs, and more. Users are students and faculty who can use the website for searching and registering for events. Organizers interact with the website to manage events. The application is deployed in the cloud with EKS on AWS, and scales automatically, allowing for high availability.

Architecture & Design

The EventSphere application handles traffic with a microservices architecture, prioritizing decoupled components to ensure scalability and maintainability. By separating the system into distinct domains, i.e, Authentication, Events, Bookings, and Notifications, we enable independent development and deployment cycles for each core feature.

Core Components

Frontend Client: A React-based Single Page Application (SPA) serves as the entry point for users. It provides a responsive interface for browsing events, managing bookings, and administration. It interacts with the backend services via RESTful HTTP APIs.

Backend Services:

- **Auth Service:** Handles user registration and login, issuing JSON Web Tokens (JWT) to secure subsequent requests across the platform. It enforces Role-Based Access Control (RBAC) to distinguish between attendees and administrators.
- **Event Service:** Serves as the catalog source of truth, managing event metadata (locations, dates, prices) and inventory.
- **Booking Service:** Manages the transaction logic for reservations, ensuring that bookings are linked correctly to users and events while validating availability with the Event Service.
- **Notification Service:** A lightweight service designed to decouple email operations from the critical path. Instead of sending emails directly, it publishes messages to an AWS SNS topic, offloading the actual delivery logic to the cloud.

Data Layer: We utilize MongoDB as our persistent storage. While the services share a database cluster for development simplicity, the architecture supports logical separation, where each microservice owns its specific collections (e.g., auth, events, bookings), preventing tight data coupling.

Infrastructure & Orchestration: The application is containerized using Docker and orchestrated via Amazon EKS (Elastic Kubernetes Service). This cloud-native approach allows for features like Horizontal Pod Autoscaling (HPA) and automated health checks. Traffic is routed into the cluster via an AWS Application Load Balancer (ALB).

Infrastructure-as-Code

- eksctl IaC template VPC, subnets, node groups, and the EKS control plane:
[eks-cluster.yaml](#)

```
! eksctl-cluster.yaml X

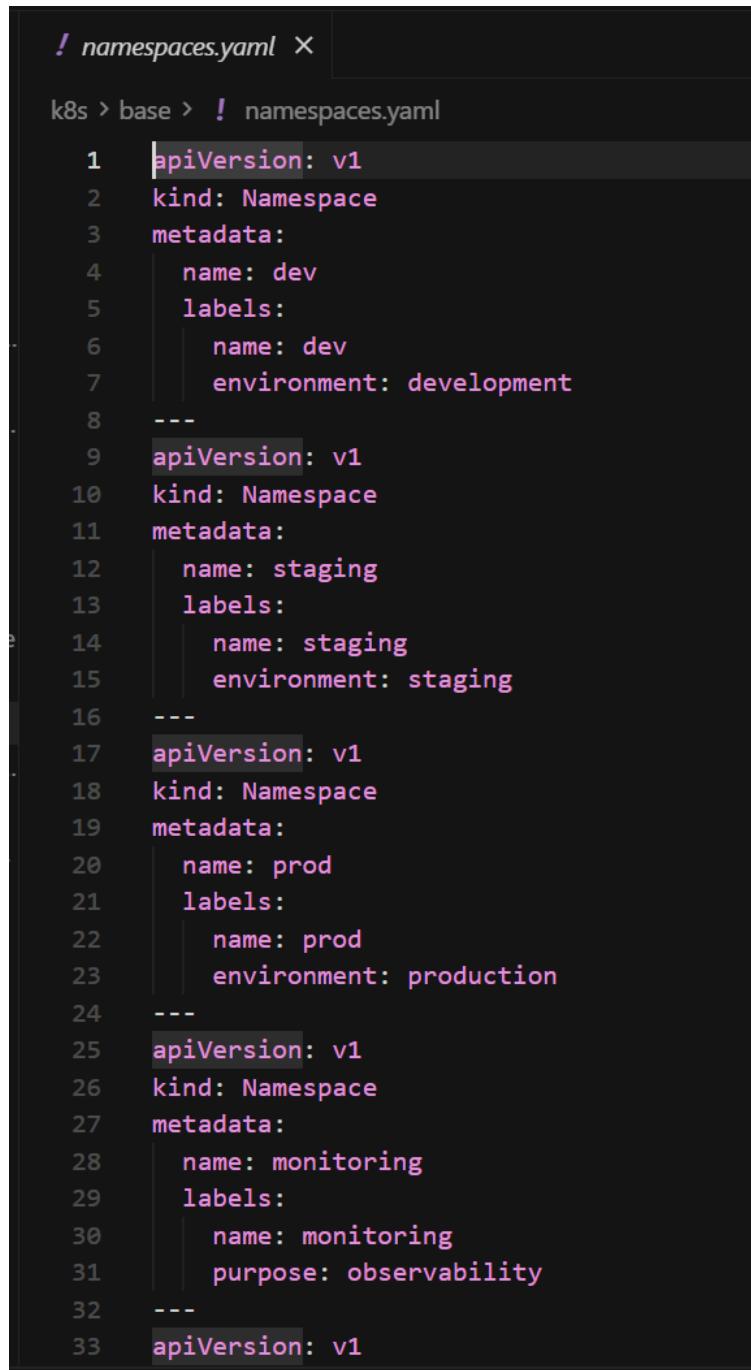
infrastructure > ! eksctl-cluster.yaml

1  apiVersion: eksctl.io/v1alpha5
2  kind: ClusterConfig
3
4  metadata:
5    name: eventsphere-cluster
6    region: us-east-1
7    version: "1.34"
8    tags:
9      Project: EventSphere
10     Environment: Production
11     ManagedBy: eksctl
12
13 # VPC Configuration
14 # eksctl will create a new VPC with public and private subnets across all AZs
15 vpc:
16   cidr: 10.0.0.0/16
17   nat:
18     gateway: HighlyAvailable # One NAT Gateway per AZ for high availability
19   clusterEndpoints:
20     publicAccess: true
21     privateAccess: true
22
23 # IAM Configuration
24 iam:
25   withOIDC: true # Required for IRSA (IAM Roles for Service Accounts)
26   serviceAccounts:
27     # AWS Load Balancer Controller
28     - metadata:
29       name: aws-load-balancer-controller
30       namespace: kube-system
31     wellKnownPolicies:
32       awsLoadBalancerController: true
33   # Cluster Autoscaler
```

- RBAC manifests for namespaces and service accounts (including IRSA):

- Namespaces:

[namespaces.yaml](#)



The screenshot shows a terminal window with the title bar 'namespaces.yaml X'. The window displays a YAML configuration for four Kubernetes namespaces: dev, staging, prod, and monitoring. Each namespace is defined by an 'apiVersion: v1' block, 'kind: Namespace', and 'metadata' section. The 'dev' namespace has a 'name: dev' and 'environment: development' label. The 'staging' namespace has a 'name: staging' and 'environment: staging' label. The 'prod' namespace has a 'name: prod' and 'environment: production' label. The 'monitoring' namespace has a 'name: monitoring' and 'purpose: observability' label.

```
! namespaces.yaml X

k8s > base > ! namespaces.yaml

1  apiVersion: v1
2  kind: Namespace
3  metadata:
4    name: dev
5    labels:
6      name: dev
7      environment: development
8  ---
9  apiVersion: v1
10 kind: Namespace
11 metadata:
12   name: staging
13   labels:
14     name: staging
15     environment: staging
16  ---
17  apiVersion: v1
18  kind: Namespace
19  metadata:
20    name: prod
21    labels:
22      name: prod
23      environment: production
24  ---
25  apiVersion: v1
26  kind: Namespace
27  metadata:
28    name: monitoring
29    labels:
30      name: monitoring
31      purpose: observability
32  ---
33  apiVersion: v1
```

- Service Accounts:

[rbac.yaml](#)

```
! rbac.yaml X

k8s > base > ! rbac.yaml

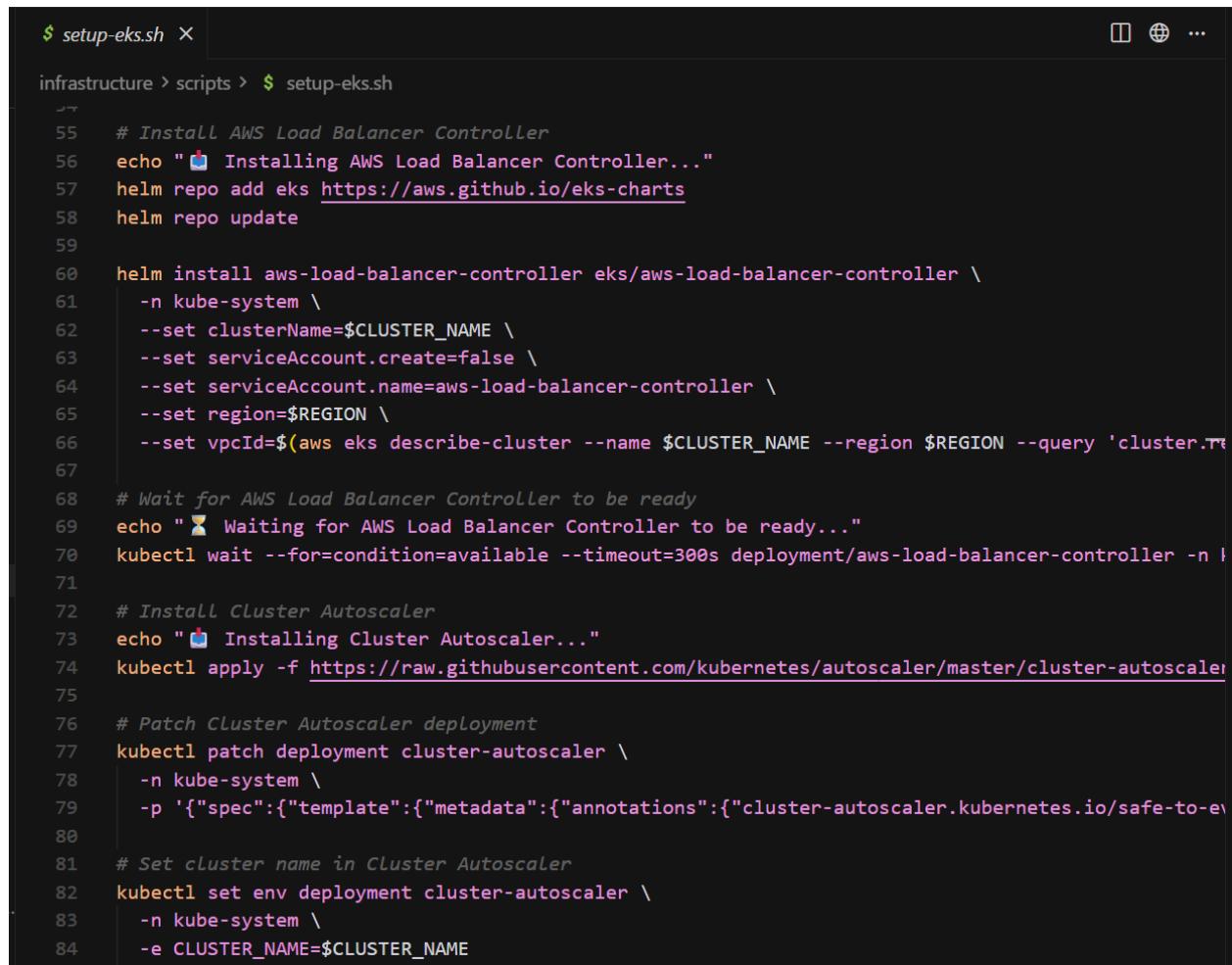
1  # -----
2  # RBAC Configuration for EventSphere
3  # -----
4  # This file contains:
5  # 1. Service Accounts for all microservices (dev, staging, prod)
6  # 2. Service-specific Roles with Least-privilege access
7  # 3. User Roles (Developer, Operator, Admin) for environment access
8  # 4. RoleBindings to connect users/service accounts to roles
9  # -----
10 #
11 # -----
12 # SERVICE ACCOUNTS - Production Namespace
13 # -----
14 ---
15 apiVersion: v1
16 kind: ServiceAccount
17 metadata:
18   name: auth-service-sa
19   namespace: prod
20   labels:
21     app: auth-service
22     environment: production
23 ---
24 apiVersion: v1
25 kind: ServiceAccount
26 metadata:
27   name: event-service-sa
28   namespace: prod
29   labels:
30     app: event-service
31     environment: production
32 ---
33 apiVersion: v1
```

Docker and Kubernetes Configuration

Deploy and Manage a Kubernetes Cluster on AWS EKS:

- Installed add-ons (ALB Controller, Cluster Autoscaler) with configs:

[setup-eks.sh](#)



The screenshot shows a terminal window with the title '\$ setup-eks.sh X'. The window displays a shell script named 'setup-eks.sh' with line numbers from 55 to 84. The script installs the AWS Load Balancer Controller and the Cluster Autoscaler. It uses Helm to install the controller and kubectl to apply a configuration file. The script also patches the deployment and sets the cluster name.

```
$ setup-eks.sh X
infrastructure > scripts > $ setup-eks.sh
55  # Install AWS Load Balancer Controller
56  echo "📦 Installing AWS Load Balancer Controller..."
57  helm repo add eks https://aws.github.io/eks-charts
58  helm repo update
59
60  helm install aws-load-balancer-controller eks/aws-load-balancer-controller \
61      -n kube-system \
62      --set clusterName=$CLUSTER_NAME \
63      --set serviceAccount.create=false \
64      --set serviceAccount.name=aws-load-balancer-controller \
65      --set region=$REGION \
66      --set vpcId=$(aws eks describe-cluster --name $CLUSTER_NAME --region $REGION --query 'cluster.resourcesVpcConfig.vpcId')
67
68  # Wait for AWS Load Balancer Controller to be ready
69  echo "⏳ Waiting for AWS Load Balancer Controller to be ready..."
70  kubectl wait --for=condition=available --timeout=300s deployment/aws-load-balancer-controller -n kube-system
71
72  # Install Cluster Autoscaler
73  echo "📦 Installing Cluster Autoscaler..."
74  kubectl apply -f https://raw.githubusercontent.com/kubernetes/autoscaler/master/cluster-autoscaler/deploy/kube-addon.yaml
75
76  # Patch Cluster Autoscaler deployment
77  kubectl patch deployment cluster-autoscaler \
78      -n kube-system \
79      -p '{"spec": {"template": {"metadata": {"annotations": {"cluster-autoscaler.kubernetes.io/safe-to-eviction": "true"}, "labels": {"k8s-app": "cluster-autoscaler", "beta.kubernetes.io/arch": "amd64", "beta.kubernetes.io/os": "linux", "node-role.kubernetes.io/master": ""}}}, "strategy": {"type": "RollingUpdate", "rollingUpdate": {"maxSurge": "10%", "maxUnavailable": "0%"}, "type": "RollingUpdate"}, "target": {"type": "DeploymentConfig", "name": "cluster-autoscaler", "namespace": "kube-system"}}, "updateLabel": "true"}, "type": "Patch"}'
80
81  # Set cluster name in Cluster Autoscaler
82  kubectl set env deployment cluster-autoscaler \
83      -n kube-system \
84      -e CLUSTER_NAME=$CLUSTER_NAME
```

- kubectl get nodes -o wide shows nodes in multiple AZs:

```
kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/Cyber/Sem 3/ENPM818R - Virtualization/Final project 2/EventSphere$ kubectl get nodes -o wide
NAME           STATUS   ROLES      AGE    VERSION   INTERNAL-IP   EXTERNAL-IP   OS-IMAGE
ip-10-0-39-3.ec2.internal   Ready    <none>    15h   v1.34.2-eks-ecaa3a6   10.0.39.3   184.72.214.172   Amazon Linux 2023
.ip-10-0-4-244.ec2.internal   Ready    <none>    15h   v1.34.2-eks-ecaa3a6   10.0.4.244   34.204.168.21   Amazon Linux 2023
.ip-10-0-40-82.ec2.internal   Ready    <none>    15h   v1.34.2-eks-ecaa3a6   10.0.40.82   107.20.100.184   Amazon Linux 2023
.ip-10-0-41-129.ec2.internal   Ready    <none>    15h   v1.34.2-eks-ecaa3a6   10.0.41.129   54.81.207.250   Amazon Linux 2023
.ip-10-0-6-193.ec2.internal   Ready    <none>    15h   v1.34.2-eks-ecaa3a6   10.0.6.193   98.92.155.102   Amazon Linux 2023
.ip-10-0-39-3.ec2.internal   Ready    <none>    15h   v1.34.2-eks-ecaa3a6   10.0.39.3   184.72.214.172   Amazon Linux 2023
kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/Cyber/Sem 3/ENPM818R - Virtualization/Final project 2/EventSphere$ kubectl get nodes -o custom-columns=NAME:.metadata.name,ZONE:.metadata.labels."topology\kubernetes\.io/zone",INSTANCE-TYPE:.metadata.labels."node\kubernetes\.io/instance-type",STATUS:.status.conditions[-1].type
NAME          ZONE     INSTANCE-TYPE   STATUS
ip-10-0-39-3.ec2.internal   us-east-1b   t3.small       Ready
ip-10-0-4-244.ec2.internal   us-east-1f   t3.small       Ready
ip-10-0-40-82.ec2.internal   us-east-1b   t3.small       Ready
ip-10-0-41-129.ec2.internal   us-east-1b   t3.small       Ready
ip-10-0-6-193.ec2.internal   us-east-1f   t3.small       Ready
```

- Autoscaler responds to load changes and logs scale events:

1. Total node count 5 before load test:

```
kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/Cyber/Sem 3/ENPM818R - Virtualization/Final project 2/EventSphere$ kubectl get nodes -o wide
NAME           STATUS   ROLES      AGE    VERSION   INTERNAL-IP   EXTERNAL-IP   OS-IMAGE
ip-10-0-39-3.ec2.internal   Ready    <none>    15h   v1.34.2-eks-ecaa3a6   10.0.39.3   184.72.214.172   Amazon Linux 2023
.ip-10-0-4-244.ec2.internal   Ready    <none>    15h   v1.34.2-eks-ecaa3a6   10.0.4.244   34.204.168.21   Amazon Linux 2023
.ip-10-0-40-82.ec2.internal   Ready    <none>    15h   v1.34.2-eks-ecaa3a6   10.0.40.82   107.20.100.184   Amazon Linux 2023
.ip-10-0-41-129.ec2.internal   Ready    <none>    15h   v1.34.2-eks-ecaa3a6   10.0.41.129   54.81.207.250   Amazon Linux 2023
.ip-10-0-6-193.ec2.internal   Ready    <none>    15h   v1.34.2-eks-ecaa3a6   10.0.6.193   98.92.155.102   Amazon Linux 2023
.ip-10-0-39-3.ec2.internal   Ready    <none>    15h   v1.34.2-eks-ecaa3a6   10.0.39.3   184.72.214.172   Amazon Linux 2023
```

2. Node group 1 node count before load test (desiredSize=2):

```
kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/Cyber/Sem 3/ENPM818R - Virtualization/Final project 2/EventSphere$ aws eks describe-nodegroup --cluster-name eventsphere-cluster --nodegroup-name eventsphere-mng-1 --region us-east-1 --query 'nodegroup.scalingConfig'
{
    "minSize": 2,
    "maxSize": 5,
    "desiredSize": 2
}
```

3. Starting load test:

```
kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/Cyber/Sem 3/ENPM818R - Virtualization/Final project 2/EventSphere$ kubectl apply -f ./k8s/load-test.yaml
deployment.apps/load-test created
```

4. Node group 1 node count during load test (desiredSize=5):

```
kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/Cyber/Sem 3/ENPM818R - Virtualization/Final project 2/EventSphere$ aws eks describe-nodegroup --cluster-name eventsphere-cluster --nodegroup-name eventsphere-mng-1 --region us-east-1 --query 'nodegroup.scalingConfig'
{
    "minSize": 2,
    "maxSize": 5,
    "desiredSize": 5
}
```

5. New nodes created and total node count is now 8:

NAME	STATUS	ROLES	AGE	VERSION CONTAINER-RUNTIME	INTERNAL-IP	EXTERNAL-IP	OS-IMAGE
				KERNEL-VERSION			
ip-10-0-11-113.ec2.internal	Ready	<none>	20h	v1.34.2-eks-ecaa3a6 containerd://2.1.4	10.0.11.113	44.203.89.35	Amazon Linux 20
ip-10-0-31-26.ec2.internal	Ready	<none>	21h	v1.34.2-eks-ecaa3a6 containerd://2.1.4	10.0.31.26	44.203.197.239	Amazon Linux 20
ip-10-0-4-42.ec2.internal	Ready	<none>	3m22s	v1.34.2-eks-ecaa3a6 containerd://2.1.4	10.0.4.42	18.205.189.23	Amazon Linux 20
ip-10-0-40-129.ec2.internal	Ready	<none>	21h	v1.34.2-eks-ecaa3a6 containerd://2.1.4	10.0.40.129	18.207.240.232	Amazon Linux 20
ip-10-0-43-2.ec2.internal	Ready	<none>	9h	v1.34.2-eks-ecaa3a6 containerd://2.1.4	10.0.43.2	44.192.19.69	Amazon Linux 20
ip-10-0-50-151.ec2.internal	Ready	<none>	3m25s	v1.34.2-eks-ecaa3a6 containerd://2.1.4	10.0.50.151	3.230.158.223	Amazon Linux 20
ip-10-0-55-2.ec2.internal	Ready	<none>	21h	v1.34.2-eks-ecaa3a6 containerd://2.1.4	10.0.55.2	3.238.116.104	Amazon Linux 20
ip-10-0-60-236.ec2.internal	Ready	<none>	3m24s	v1.34.2-eks-ecaa3a6 containerd://2.1.4	10.0.60.236	98.93.42.178	Amazon Linux 20
ip-10-0-74-119.amzn2023.x86_64							

6. Autoscaler logs showing scale up event:

```
kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/Cyber/Sem 3/ENPM818R - Virtualization/Final project 2/EventSphere$ kubectl get events -n kube-system --sort-by='.lastTimestamp' | grep -i scale-up
9m8s      Normal    ScaledUpGroup    configmap/cluster-autoscaler-status  Scale-up: setting group eks-eventsphere-mng-1-dacd7a87-de3e-36a2-dbd2-c697cda37bfe size to 5 instead of 2 (max: 5)
9m7s      Normal    ScaledUpGroup    configmap/cluster-autoscaler-status  Scale-up: group eks-eventsphere-mng-1-dacd7a87-de3e-36a2-dbd2-c697cda37bfe size set to 5 instead of 2 (max: 5)
```

7. Deleting load-test deployment to trigger scale-down:

```
kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/Cyber/Sem 3/ENPM818R - Virtualization/Final project 2/EventSphere$ kubectl delete deployment load-test -n prod
deployment.apps "load-test" deleted from prod namespace
```

8. Scale-down logs:

```
kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/  
Cyber/Sem 3/ENPM818R - Virtualization/Final project 2/EventSpher  
e$ kubectl logs -n kube-system -l app=cluster-autoscaler --tail=100 | grep -i "scale.*down\|removed\|drain"  
I1207 16:22:40.716653      1 eligibility.go:163] Node ip-10-0-1  
1-113.ec2.internal unremovable: memory requested (52.0856% of all  
allocatable) is above the scale-down utilization threshold  
I1207 16:22:40.716695      1 eligibility.go:163] Node ip-10-0-3  
1-26.ec2.internal unremovable: memory requested (55.9954% of all  
allocatable) is above the scale-down utilization threshold  
I1207 16:22:40.716728      1 eligibility.go:163] Node ip-10-0-4  
0-129.ec2.internal unremovable: memory requested (77.7791% of all  
allocatable) is above the scale-down utilization threshold  
I1207 16:22:40.716759      1 eligibility.go:163] Node ip-10-0-5  
5-2.ec2.internal unremovable: memory requested (86.437% of alloc  
atable) is above the scale-down utilization threshold  
I1207 16:22:40.716771      1 eligibility.go:104] Scale-down cal  
culation: ignoring 1 nodes unremovable in the last 5m0s  
I1207 16:22:40.716819      1 cluster.go:173] node ip-10-0-50-15  
1.ec2.internal may be removed  
I1207 16:22:40.716856      1 cluster.go:173] node ip-10-0-60-23  
6.ec2.internal may be removed  
I1207 16:22:40.716907      1 cluster.go:173] node ip-10-0-4-42.  
ec2.internal may be removed  
I1207 16:22:40.717160      1 static_autoscaler.go:598] Scale do  
wn status: lastScaleUpTime=2025-12-07 16:10:35.936363273 +0000 U  
TC m=+75397.498723648 lastScaleDownDeleteTime=2025-12-06 18:14:2  
0.939505455 +0000 UTC m=-3577.498134162 lastScaleDownFailTime=20  
25-12-06 18:14:20.939505455 +0000 UTC m=-3577.498134162 scaleDow  
nForbidden=false scaleDownInCooldown=false  
I1207 16:22:40.717211      1 static_autoscaler.go:619] Starting  
scale down
```

9. Node group 1 node count after scale-down (desiredSize=2):

```
kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/Cyber/Sem 3/ENPM818R - Virtualization/Final project 2/EventSphere$ kubectl delete deployment load-test -n prod
deployment.apps "load-test" deleted from prod namespace
kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/Cyber/Sem 3/ENPM818R - Virtualization/Final project 2/EventSphere$ aws eks describe-nodegroup --cluster-name eventsphere-cluster --nodegroup-name eventsphere-mng-1 --region us-east-1 --query 'nodegroup.scalingConfig'
{
    "minSize": 2,
    "maxSize": 5,
    "desiredSize": 2
}
```

10. Total node count 5 after scale-down:

NAME	KERNEL-VERSION	STATUS	ROLES	AGE	VERSION	CONTAINER-RUNTIME	INTERNAL-IP	EXTERNAL-IP	OS-IMAGE
ip-10-0-39-3.ec2.internal	6.12.55-74.119.amzn2023.x86_64	Ready	<none>	15h	v1.34.2-eks-ecaa3a6	containerd://2.1.4	10.0.39.3	184.72.214.172	Amazon Linux 2023
ip-10-0-4-244.ec2.internal	6.12.55-74.119.amzn2023.x86_64	Ready	<none>	15h	v1.34.2-eks-ecaa3a6	containerd://2.1.4	10.0.4.244	34.204.168.21	Amazon Linux 2023
ip-10-0-40-82.ec2.internal	6.12.55-74.119.amzn2023.x86_64	Ready	<none>	15h	v1.34.2-eks-ecaa3a6	containerd://2.1.4	10.0.40.82	107.20.100.184	Amazon Linux 2023
ip-10-0-41-129.ec2.internal	6.12.55-74.119.amzn2023.x86_64	Ready	<none>	15h	v1.34.2-eks-ecaa3a6	containerd://2.1.4	10.0.41.129	54.81.207.250	Amazon Linux 2023
ip-10-0-6-193.ec2.internal	6.12.55-74.119.amzn2023.x86_64	Ready	<none>	15h	v1.34.2-eks-ecaa3a6	containerd://2.1.4	10.0.6.193	98.92.155.102	Amazon Linux 2023

- At least one workload uses IRSA for AWS access:

- External Secrets Operator Uses IRSA:

```
kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/Cyber/Sem 3/ENPM818R - Virtualization/Final project 2/EventSphere$ kubectl get serviceaccount external-secrets -n external-secrets-system -o yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  annotations:
    eks.amazonaws.com/role-arn: arn:aws:iam::277707118728:role/external-secrets-role
    kubernetes.io/last-applied-configuration: |
      {"apiVersion":"v1","kind":"ServiceAccount","metadata":{"annotations":{},"creationTimestamp":null,"name":"external-secrets","namespace":"external-secrets-system"}}
    creationTimestamp: "2025-12-06T19:05:19Z"
    labels:
      app.kubernetes.io/managed-by: eksctl
      name: external-secrets
      namespace: external-secrets-system
      resourceVersion: "6037"
      uid: d13b643b-fdbc-4f35-8086-30b780237d6c
```

- ALB or NGINX Ingress Controller routes public traffic correctly:
 1. Curl to ALB DNS responds with 301 status and location as <https://www.enpm818rgroup7.work.gd:443/> to enforce HTTPS:

```
kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/Cyber/Sem 3/ENPM818R - Virtualization/Final project 2/Eventsphere$ curl -v -H "Host: www.enpm818rgroup7.work.gd" http://$ALB_DNS/
* Host k8s-prod-eventsph-395841f6f1-1344915046.us-east-1.elb.amazonaws.com:80 was resolved.
* IPv6: (none)
* IPv4: 52.6.253.22, 3.221.240.165
* Trying 52.6.253.22:80...
* Connected to k8s-prod-eventsph-395841f6f1-1344915046.us-east-1.elb.amazonaws.com (52.6.253.22) port 80
> GET / HTTP/1.1
> Host: www.enpm818rgroup7.work.gd
> User-Agent: curl/8.5.0
> Accept: */*
>
< HTTP/1.1 301 Moved Permanently
< Server: awselb/2.0
< Date: Sun, 07 Dec 2025 17:08:04 GMT
< Content-Type: text/html
< Content-Length: 134
< Connection: keep-alive
< Location: https://www.enpm818rgroup7.work.gd:443/
<
<html>
<head><title>301 Moved Permanently</title></head>
<body>
<center><h1>301 Moved Permanently</h1></center>
</body>
</html>
* Connection #0 to host k8s-prod-eventsph-395841f6f1-1344915046.us-east-1.elb.amazonaws.com left intact
```

2. Curl to <https://www.enpm818rgroup7.work.gd:443/>

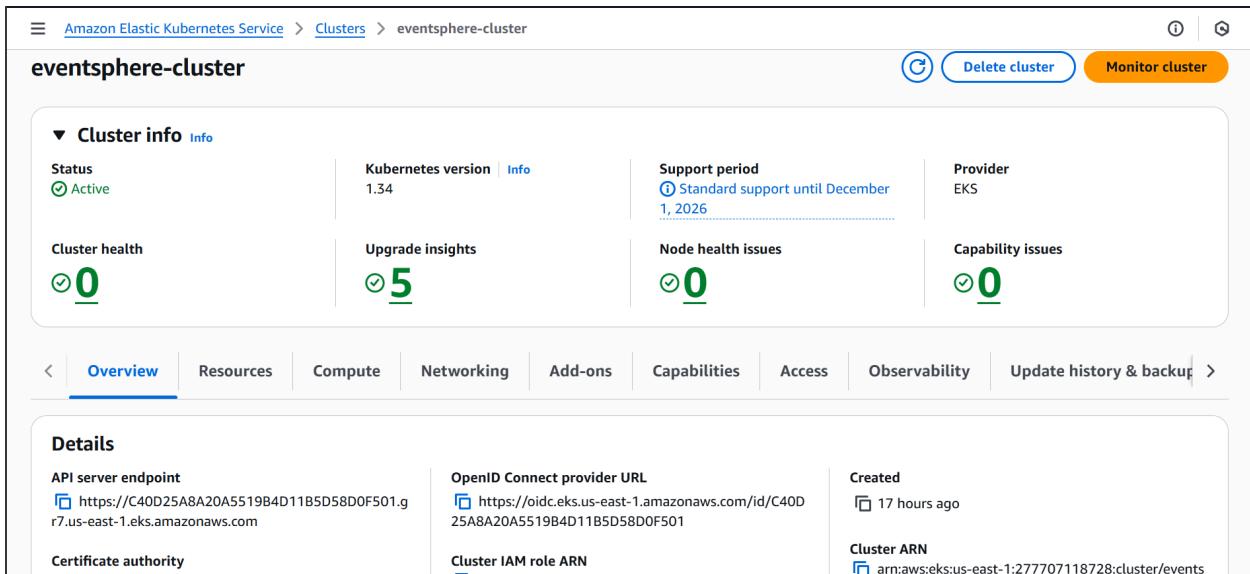
```

kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/Cyber/Sem 3/ENPM818R - Virtualization/Final project 2/Events
phere$ curl -v -H "Host: www.enpm818rgroup7.work.gd" https://www.enpm818rgroup7.work.gd:443/
* Host www.enpm818rgroup7.work.gd:443 was resolved.
* IPv6: (none)
* IPv4: 3.221.240.165, 52.6.253.22
* Trying 3.221.240.165:443...
* Connected to www.enpm818rgroup7.work.gd (3.221.240.165) port 443
* ALPN: curl offers h2,http/1.1
* TLSv1.3 (OUT), TLS handshake, Client hello (1):
* CAfile: /etc/ssl/certs/ca-certificates.crt
* CApth: /etc/ssl/certs
* TLSv1.3 (IN), TLS handshake, Server hello (2):
* TLSv1.2 (IN), TLS handshake, Certificate (11):
* TLSv1.2 (IN), TLS handshake, Server key exchange (12):
* TLSv1.2 (IN), TLS handshake, Server finished (14):
* TLSv1.2 (OUT), TLS handshake, Client key exchange (16):
* TLSv1.2 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.2 (OUT), TLS handshake, Finished (20):
* TLSv1.2 (IN), TLS handshake, Finished (20):
* SSL connection using TLSv1.2 / ECDHE-RSA-AES128-GCM-SHA256 / prime256v1 / rsaEncryption
* ALPN: server accepted h2
* Server certificate:
*   subject: CN=enpm818rgroup7.work.gd
*   start date: Nov 22 00:00:00 2025 GMT
*   expire date: Dec 21 23:59:59 2026 GMT
*   subjectAltName: host "www.enpm818rgroup7.work.gd" matched cert's "www.enpm818rgroup7.work.gd"
*   issuer: C=US; O=Amazon; CN=Amazon RSA 2048 M04
*   SSL certificate verify ok.
*   Certificate level 0: Public key type RSA (2048/112 Bits/secBits), signed using sha256WithRSAEncryption
*   Certificate level 1: Public key type RSA (2048/112 Bits/secBits), signed using sha256WithRSAEncryption
*   Certificate level 2: Public key type RSA (2048/112 Bits/secBits), signed using sha256WithRSAEncryption
* using HTTP/2
* [HTTP/2] [1] OPENED stream for https://www.enpm818rgroup7.work.gd:443/
* [HTTP/2] [1] [:method: GET]
* [HTTP/2] [1] [:scheme: https]

* issuer: C=US; O=Amazon; CN=Amazon RSA 2048 M04
* SSL certificate verify ok.
*   Certificate level 0: Public key type RSA (2048/112 Bits/secBits), signed using sha256WithRSAEncryption
*   Certificate level 1: Public key type RSA (2048/112 Bits/secBits), signed using sha256WithRSAEncryption
*   Certificate level 2: Public key type RSA (2048/112 Bits/secBits), signed using sha256WithRSAEncryption
* using HTTP/2
* [HTTP/2] [1] OPENED stream for https://www.enpm818rgroup7.work.gd:443/
* [HTTP/2] [1] [:method: GET]
* [HTTP/2] [1] [:scheme: https]
* [HTTP/2] [1] [:authority: www.enpm818rgroup7.work.gd]
* [HTTP/2] [1] [:path: /]
* [HTTP/2] [1] [:user-agent: curl/8.5.0]
* [HTTP/2] [1] [:accept: */*]
> GET / HTTP/2
> Host: www.enpm818rgroup7.work.gd
> User-Agent: curl/8.5.0
> Accept: */*
>
< HTTP/2 200
< date: Sun, 07 Dec 2025 17:11:51 GMT
< content-type: text/html
< content-length: 936
< server: nginx/1.29.2
< last-modified: Thu, 04 Dec 2025 07:32:27 GMT
< etag: "6931390b-3a8"
< accept-ranges: bytes
<
* Connection #0 to host www.enpm818rgroup7.work.gd left intact
<!doctype html><html lang="en"><head><meta charset="utf-8"/><link rel="icon" href="/favicon.ico"/><meta name="viewport" content="width=device-width,initial-scale=1"/><meta name="theme-color" content="#000000"/><meta name="description" content="Web site created using create-react-app"/><link rel="apple-touch-icon" href="/logo192.png"/><link rel="manifest" href="/manifest.json"/><title>Campus Event Management</title><link rel="preconnect" href="https://fonts.googleapis.com"><link rel="preconnect" href="https://fonts.gstatic.com" crossorigin><link href="https://fonts.googleapis.com/css2?family=Roboto:wght@300;400;500;700&family=Montserrat:wght@400;500;600;700&display=swap" rel="stylesheet"><script defer="defer" src="/static/js/main.81c7b
kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/Cyber/Sem 3/ENPM818R - Virtualization/Final project 2/Events

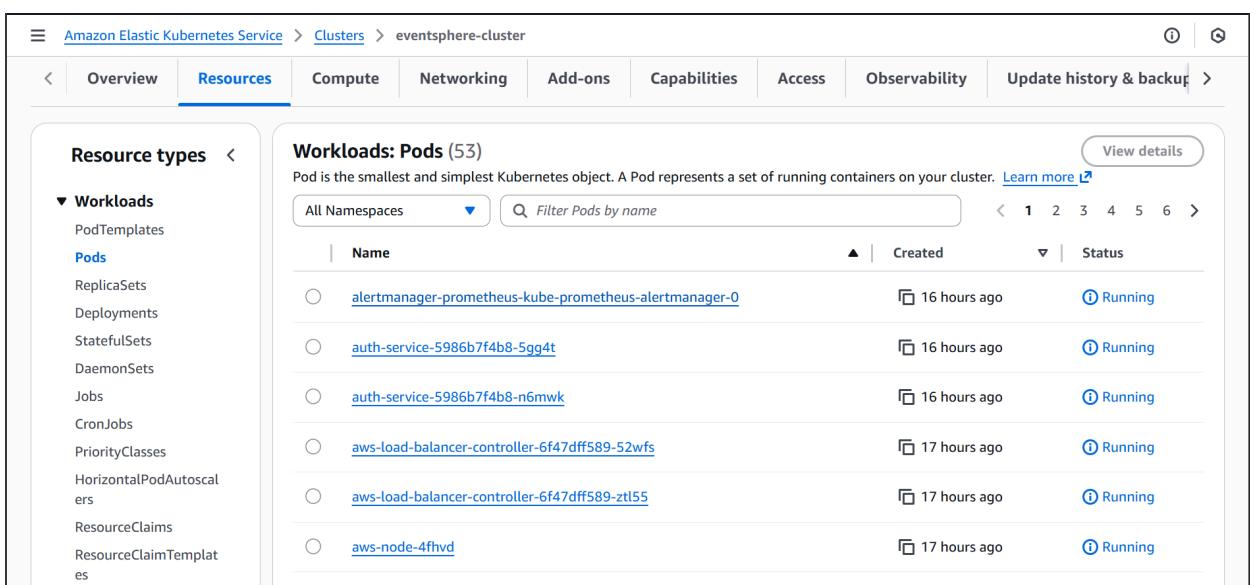
```

- EKS Console:



The screenshot shows the 'Overview' tab of the EKS Cluster details page. It displays cluster status, Kubernetes version, support period, and provider information. It also shows cluster health, upgrade insights, node health issues, and capability issues. Below this, the 'Details' section provides API server endpoint, OpenID Connect provider URL, and Cluster IAM role ARN.

API server endpoint	OpenID Connect provider URL	Created
https://C40D25A8A20A5519B4D11B5D58D0F501.g7.us-east-1.eks.amazonaws.com	https://oidc.eks.us-east-1.amazonaws.com/id/C40D25A8A20A5519B4D11B5D58D0F501	17 hours ago
Certificate authority	Cluster IAM role ARN	Cluster ARN
	arn:aws:iam::277707119728:role/eksctl-eventsphere-role	arn:aws:eks:us-east-1:277707118728:cluster/eventsphere



The screenshot shows the 'Resources' tab of the EKS Cluster details page. It displays a list of workloads under the 'Pods' category. Each row shows the pod name, creation time, and status.

Name	Created	Status
alertmanager-prometheus-kube-prometheus-alertmanager-0	16 hours ago	Running
auth-service-5986b7f4b8-5gg4t	16 hours ago	Running
auth-service-5986b7f4b8-n6mwk	16 hours ago	Running
aws-load-balancer-controller-6f47dff589-52wfs	17 hours ago	Running
aws-load-balancer-controller-6f47dff589-ztl55	17 hours ago	Running
aws-node-4fhvd	17 hours ago	Running

Amazon Elastic Kubernetes Service > Clusters > eventsphere-cluster

Resource types

- Workloads
- PodTemplates
- Pods
- ReplicaSets**
- Deployments
- StatefulSets
- DaemonSets
- Jobs
- CronJobs
- PriorityClasses
- HorizontalPodAutoscalers
- ResourceClaims
- ResourceClaimTemplates
- ResourceSlices

Workloads: ReplicaSets (26)

ReplicaSet aims to maintain a set of replica Pods running at any given time. [Learn more ↗](#)

Name	Namespace	Type	Created	Pod count	Status
auth-service-54b59c4d5c	prod	replicasets	17 hours ago	0	0 Ready 0 Failed 0 Desired
auth-service-5986b7f4b8	prod	replicasets	16 hours ago	2	2 Ready 0 Failed 2 Desired
aws-load-balancer-controller-6f47dff589	kube-system	replicasets	17 hours ago	2	2 Ready 0 Failed 2 Desired
booking-service-7c6d6c87f7	prod	replicasets	16 hours ago	2	2 Ready 0 Failed 2 Desired

Booking service 17 hours

[View details](#)

Amazon Elastic Kubernetes Service > Clusters > eventsphere-cluster

Resource types

- Workloads
- PodTemplates
- Pods
- ReplicaSets
- Deployments**
- StatefulSets
- DaemonSets
- Jobs
- CronJobs
- PriorityClasses
- HorizontalPodAutoscalers
- ResourceClaims
- ResourceClaimTemplates
- ResourceSlices

Workloads: Deployments (16)

Deployment is an API object that manages a replicated application, typically by running Pods with no local state. [Learn more ↗](#)

Name	Namespace	Type	Created	Pod count	Status
auth-service	prod	deployments	17 hours ago	2	2 Ready 0 Failed 2 Desired
aws-load-balancer-controller	kube-system	deployments	17 hours ago	2	2 Ready 0 Failed 2 Desired
booking-service	prod	deployments	17 hours ago	2	2 Ready 0 Failed 2 Desired
cluster-autoscaler	kube-system	deployments	17 hours ago	1	1 Ready 0 Failed 1 Desired

17 hours

[View details](#)

Amazon Elastic Kubernetes Service > Clusters > eventsphere-cluster

Resource types <

▼ Workloads

- PodTemplates
- Pods
- ReplicaSets
- Deployments
- StatefulSets
- DaemonSets**
- Jobs
- CronJobs
- PriorityClasses
- HorizontalPodAutoscalers
- ResourceClaims
- ResourceClaimTemplates
- ResourceSlices

Workloads: DaemonSets (6)

Daemonset ensures a copy of a Pod is running across a set of nodes in a cluster. [Learn more ↗](#)

Name	Namespace	Type	Created	Pod count	Status
aws-node	kube-system	daemonsets	17 hours ago	5	5 Ready 0 Failed 5 Desired
ebs-csi-node	kube-system	daemonsets	17 hours ago	5	5 Ready 0 Failed 5 Desired
ebs-csi-node-windows	kube-system	daemonsets	17 hours ago	0	0 Ready 0 Failed 0 Desired
fluent-bit	amazon-cloudwatch	daemonsets	16 hours ago	0	0 Ready 0 Failed 5 Desired
			17 hours ago		

[View details](#)

< 1 >

Amazon Elastic Kubernetes Service > Clusters > eventsphere-cluster

Resource types <

▼ Workloads

- PodTemplates
- Pods
- ReplicaSets
- Deployments
- StatefulSets
- DaemonSets
- Jobs
- CronJobs
- PriorityClasses
- HorizontalPodAutoscalers**
- ResourceClaims
- ResourceClaimTemplates
- ResourceSlices

Workloads: HorizontalPodAutoscalers (5)

The horizontal pod autoscaler automatically scales Pods depending on CPU demand. [Learn more ↗](#)

Name	Created
auth-service-hpa	17 hours ago
booking-service-hpa	17 hours ago
event-service-hpa	17 hours ago
frontend-hpa	17 hours ago
notification-service-hpa	17 hours ago

[View details](#)

< 1 >

Amazon Elastic Kubernetes Service > Clusters > eventsphere-cluster

Resource types

- ▶ Workloads
- ▼ Cluster
 - Nodes**
 - Namespaces
 - APIServices
 - Leases
 - RuntimeClasses
 - FlowSchemas
 - PriorityLevelConfigurations
- ▶ Service and networking
- ▶ Config and secrets
- ▶ Storage
- ▶ Authentication
- ▶ Authorization

Cluster: Nodes (5)
A node is a worker machine in Kubernetes. [Learn more ↗](#)

Filter Nodes by name

Node name	Instance type	Compute	Managed by	Created	Status
ip-10-0-39-3.ec2.internal	t3.small	Node group	eventsphere-mng-2	16 hours ago	Read
ip-10-0-4-244.ec2.internal	t3.small	Node group	eventsphere-mng-2	17 hours ago	Read
ip-10-0-40-82.ec2.internal	t3.small	Node group	eventsphere-mng-1	17 hours ago	Read
ip-10-0-41-129.ec2.internal	t3.small	Node group	eventsphere-mng-2	17 hours ago	Read
ip-10-0-6-193.ec2.internal	t3.small	Node group	eventsphere-mng-1	17 hours ago	Read

Amazon Elastic Kubernetes Service > Clusters > eventsphere-cluster

Resource types

- ▶ Workloads
- ▼ Cluster
 - Nodes
 - Namespaces**
 - Namespaces
 - APIServices
 - Leases
 - RuntimeClasses
 - FlowSchemas
 - PriorityLevelConfigurations
- ▶ Service and networking
- ▶ Config and secrets
- ▶ Storage
- ▶ Authentication
- ▶ Authorization

Cluster: Namespaces (10)
Namespace is an abstraction used by Kubernetes to support isolation of groups of resources within a single cluster. [Learn more ↗](#)

Filter Namespaces by name

Name	Created
amazon-cloudwatch	16 hours ago
default	17 hours ago
dev	17 hours ago
external-secrets-system	17 hours ago
kube-node-lease	17 hours ago
kube-public	17 hours ago
kube-system	17 hours ago
monitoring	17 hours ago

kubectl get pvc,pv outputs showing Bound state:

```
kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/Cyber/Sem 3/ENPM818R - Virtualization/Final project 2/Events
phere$ kubectl get pvc -n prod
NAME           STATUS    VOLUME                                     CAPACITY   ACCESS MODES  STORAGECLASS  VOLUME
ATTRIBUTESCLASS AGE
mongodb-data-mongodb-0 Bound    pvc-4273ef6d-40aa-4766-839a-e2d128954985  20Gi       RWO          mongodb-ebs  <unset>
>              2d4h
```

```
kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/Cyber/Sem 3/ENPM818R - Virtualization/Final project 2/Events
phere$ kubectl get pv -n prod
NAME           CAPACITY   ACCESS MODES  RECLAIM POLICY  STATUS    CLAIM
STORAGECLASS  VOLUMEATRIBUTESCLASS  REASON  AGE
pvc-4273ef6d-40aa-4766-839a-e2d128954985  20Gi     RWO        Retain      Bound    prod/mongodb-data-mongodb-0
mongodb-ebs  <unset>      2d4h
```

Design and Build Dockerized Microservices

Screenshot of containers running locally:

The screenshot shows the Docker Desktop interface. On the left, there's a sidebar with options like Ask Gordon, Containers (which is selected), Images, Volumes, Kubernetes, Builds, Models, MCP Toolkit, Docker Hub, Docker Scout, and Extensions. The main area is titled 'Containers' and shows various metrics: Container CPU usage (3.21% / 1600%), Container memory usage (342.07MB / 7.28GB), and a 'Show charts' button. Below these are search and filter fields ('Search' and 'Only show running containers'). A table lists seven containers:

Name	Container ID	Image	Port(s)	CPU	Actions
eventsphere	-	-	-	3	[...]
event-platform-mongodb	a39f82d3337e	mongo:7	27017:27017	0	[...]
notification-service	d033e94d7558	eventspher_4004:4004	4004:4004	0	[...]
auth-service	e00c4fb66be3	eventspher_4001:4001	4001:4001	0	[...]
event-service	1266afbea28a	eventspher_4002:4002	4002:4002	0	[...]
booking-service	336346264a32	eventspher_4003:4003	4003:4003	0	[...]
frontend	2ddc3c83f1f0	eventspher_3000:80	3000:80	0	[...]

```
>> => unpacking to docker.io/library/eventsphere-frontend:latest
=> [notification-service] resolving provenance for metadata file
=> [auth-service] resolving provenance for metadata file
=> [event-service] resolving provenance for metadata file
=> [booking-service] resolving provenance for metadata file
=> [frontend] resolving provenance for metadata file
[+] Running 13/13
✓eventsphere-frontend          Built
✓eventsphere-notification-service Built
✓eventsphere-auth-service        Built
✓eventsphere-event-service       Built
✓eventsphere-booking-service    Built
✓Network eventsphere_event-platform-network Created
✓Volume eventsphere_mongo-data   Created
✓Container event-platform-mongodb Healthy
✓Container notification-service  Started
✓Container auth-service          Started
✓Container event-service         Started
✓Container booking-service       Started
kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/Cyber/Sem 3/ENPM818R - Virtualization/Final project 2/Eventsphere$ docker ps
CONTAINER ID IMAGE                                COMMAND                  CREATED                 STATUS                  PORTS
2ddc3c83f1f0  eventsphere-frontend              "/docker-entrypoint...." 2 minutes ago   Up 2 minutes (unhealthy)  0.0.0.
0:3000->80/tcp, [::]:3000->80/tcp
336346264a32  eventsphere-booking-service     "docker-entrypoint.s..." 2 minutes ago   Up 2 minutes (healthy)   0.0.0.
0:4003->4003/tcp, [::]:4003->4003/tcp
1266afbea28a  eventsphere-event-service       "docker-entrypoint.s..." 2 minutes ago   Up 2 minutes (healthy)   0.0.0.
0:4002->4002/tcp, [::]:4002->4002/tcp
e00c4fb66be3  eventsphere-auth-service        "docker-entrypoint.s..." 2 minutes ago   Up 2 minutes (healthy)   0.0.0.
0:4001->4001/tcp, [::]:4001->4001/tcp
d033e94d7558  eventsphere-notification-service "docker-entrypoint.s..." 2 minutes ago   Up 2 minutes (healthy)   0.0.0.
0:4004->4004/tcp, [::]:4004->4004/tcp
a39f82d3337e  mongo:7                         "docker-entrypoint.s..." 2 minutes ago   Up 2 minutes (healthy)   0.0.0.
0:27017->27017/tcp, [::]:27017->27017/tcp  event-platform-mongodb
```

ECR listing with tagged versions:

Private repositories (5)				
	Repository name	URI	Created at	Tag immutability
<input type="radio"/>	auth-service	277707118728.dkr.ecr.us-east-1.amazonaws.com/auth-service	December 06, 2025, 14:19:13 (UTC-05)	Mutable
<input type="radio"/>	booking-service	277707118728.dkr.ecr.us-east-1.amazonaws.com/booking-service	December 06, 2025, 14:19:19 (UTC-05)	Mutable
<input type="radio"/>	event-service	277707118728.dkr.ecr.us-east-1.amazonaws.com/event-service	December 06, 2025, 14:19:17 (UTC-05)	Mutable
<input type="radio"/>	frontend	277707118728.dkr.ecr.us-east-1.amazonaws.com/frontend	December 06, 2025, 14:19:25 (UTC-05)	Mutable
<input type="radio"/>	notification-service	277707118728.dkr.ecr.us-east-1.amazonaws.com/notification-service	December 06, 2025, 14:19:22 (UTC-05)	Mutable

auth-service						
	Summary	Images				
Images (24)						
	Filter active images		Delete	Copy URI	Details	Scan
	Image tags	Type	Created at	Image size	Image digest	Last pulled at
<input type="checkbox"/>	latest	Image Index	December 06, 2025, 14:19:34 (UTC-05)	68.56	sha256:f7...	December 07, 2025, 15:01:22 (UTC-05)
<input type="checkbox"/>	v1	Image Index	December 08, 2025, 12:32:34 (UTC-05)	68.56	sha256:1...	-
<input type="checkbox"/>	v1.1	Image Index	December 08, 2025, 12:33:55 (UTC-05)	68.56	sha256:7...	-
<input type="checkbox"/>	v1.2	Image Index	December 08, 2025, 12:35:56 (UTC-05)	68.56	sha256:6...	-

Trivy scan reports (no Critical vulnerabilities):

```
kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/Cyber/Sem 3/ENPM818R - Virtualization/Final project 2/EventSphere$ ./bin/trivy image auth-service:latest
2025-12-08T13:30:22-05:00    INFO  [vuln] Vulnerability scanning is enabled
2025-12-08T13:30:22-05:00    INFO  [secret] Secret scanning is enabled
2025-12-08T13:30:22-05:00    INFO  [secret] If your scanning is slow, please try '--scanners vuln' to disable secret scanning
2025-12-08T13:30:22-05:00    INFO  [secret] Please see https://trivy.dev/docs/v0.68/guide/scanner/secret#recommendation for faster secret detection
2025-12-08T13:30:22-05:00    INFO  Detected OS   family="alpine" version="3.23.0"
2025-12-08T13:30:22-05:00    WARN  This OS version is not on the EOL list  family="alpine" version="3.23"
2025-12-08T13:30:22-05:00    INFO  [alpine] Detecting vulnerabilities...  os_version="3.23" repository="3.23" pkg_num=18
2025-12-08T13:30:22-05:00    INFO  Number of language-specific files      num=1
2025-12-08T13:30:22-05:00    INFO  [node-pkg] Detecting vulnerabilities...
2025-12-08T13:30:22-05:00    INFO  Table result includes only package filenames. Use '--format json' option to get the full path to the package file.

Report Summary
```

Target	Type	Vulnerabilities	Secrets
auth-service:latest (alpine 3.23.0)	alpine	0	-
app/node_modules/@isaacs/clui/package.json	node-pkg	0	-
app/node_modules/@isaacs/fs-minipass/package.json	node-pkg	0	-
app/node_modules/@mongodb-js/saslprep/package.json	node-pkg	0	-
app/node_modules/@opentelemetry/api/package.json	node-pkg	0	-
app/node_modules/@pkjjs/parseargs/package.json	node-pkg	0	-
app/node_modules/@types/webidl-conversions/package.json	node-pkg	0	-
app/node_modules/@types/whatwg-url/package.json	node-pkg	0	-
app/node_modules/accepts/package.json	node-pkg	0	-
app/node_modules/ansi-regex/package.json	node-pkg	0	-
app/node_modules/ansi-styles/package.json	node-pkg	0	-
app/node_modules/asynckit/package.json	node-pkg	0	-
app/node_modules/axios/package.json	node-pkg	0	-
app/node_modules/balanced-match/package.json	node-pkg	0	-

usr/local/lib/node_modules/npm/node_modules/yallist/package.json	node-pkg	0	-
usr/local/lib/node_modules/npm/package.json	node-pkg	0	-

Legend:
- '-' Not scanned
- '0': Clean (no security findings detected)

```
Node.js (node-pkg)
Total: 3 (UNKNOWN: 0, LOW: 0, MEDIUM: 1, HIGH: 2, CRITICAL: 0)


```

Library	Vulnerability	Severity	Status	Installed Version	Fixed Version	Title
glob (package.json)	CVE-2025-64756	HIGH	fixed	10.4.5	11.1.0, 10.5.0	glob: glob: Command Injection Vulnerability via Malicious Filenames https://avd.aquasec.com/nvd/cve-2025-64756
tar (package.json)	CVE-2025-64118	MEDIUM		7.5.1	7.5.2	node-tar has a race condition leading to uninitialized memory exposure https://avd.aquasec.com/nvd/cve-2025-64118

Load Balancing & Networking

ALB:

The screenshot shows the AWS CloudFormation Load Balancer configuration for the stack `k8s-prod-eventsph-395841f6f1`. The main details are:

- Scheme:** Internet-facing
- Hosted zone:** Z355XDOTRQ7X7K
- Availability Zones:** subnet-0e4331bf6cf4a9bff (us-east-1f (use1-az5)), subnet-0ecc6b14bc81ab99f (us-east-1a (use1-az2))
- Date created:** December 6, 2025, 14:29 (UTC-05:00)

Load balancer ARN: arn:aws:elasticloadbalancing:us-east-1:277707118728:loadbalancer/app/k8s-prod-eventsph-395841f6f1/c74fc405daabc92e

DNS name Info: k8s-prod-eventsph-395841f6f1-1344915046.us-east-1.elb.amazonaws.com (A Record)

Listeners and rules: (2) [Info](#)

A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules.

Protocol:Port	Default action	Rules	ARN	Security policy	Default SSL/TLS certificate	mTLS
HTTP:80	Redirect to HTTPS://#{host}:443/#{path}?	1 rule	ARN	Not applicable	Not applicable	Not applicable
HTTP:301	Status code: HTTP_301					
HTTPS:443	Return fixed response	7 rules	ARN	ELBSecurityPolicy-TLS-1-2-20...	enpm818rgroup7.work.gd (Cer...	Off
	Response code: 404					
	Response body					
	Response content type: text/plain					

ALB Network Mapping:

The screenshot shows the AWS CloudFormation Network mapping configuration for the stack `k8s-prod-eventsph-395841f6f1`. The main details are:

- Internet-facing:** Z355XDOTRQ7X7K
- Subnets:** subnet-0e4331bf6cf4a9bff (us-east-1f (use1-az5)), subnet-0ecc6b14bc81ab99f (us-east-1a (use1-az2))
- Date created:** December 6, 2025, 14:29 (UTC-05:00)

Load balancer ARN: arn:aws:elasticloadbalancing:us-east-1:277707118728:loadbalancer/app/k8s-prod-eventsph-395841f6f1/c74fc405daabc92e

DNS name Info: k8s-prod-eventsph-395841f6f1-1344915046.us-east-1.elb.amazonaws.com (A Record)

Network mapping: [Info](#)

Targets in the listed zones and subnets are available for traffic from the load balancer using the IP addresses shown.

VPC	Load balancer IP address type	IP pools
vpc-0fa3cec13b872361a	IPv4	-
IPv4 VPC CIDR: 10.0.0.0/16		
IPv6 VPC CIDR: -		

Availability Zones and subnets: [Info](#)

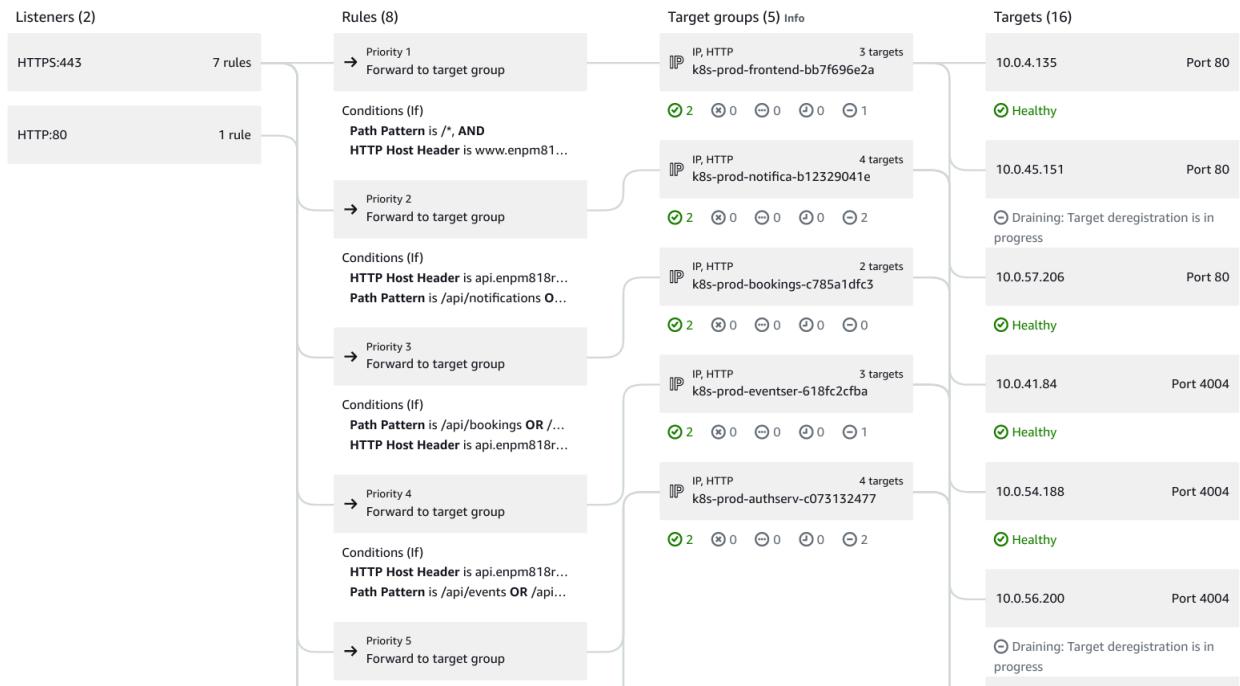
Including two or more Availability Zones, and corresponding subnets, increases the fault tolerance of your applications.

Zone	Subnet	Private IPv4 address	IPv6 address
us-east-1f (use1-az5)	subnet-0e4331bf6cf4a9bff	Assigned from CIDR 10.0.32.0/19	Not applicable
us-east-1a (use1-az2)	subnet-0ecc6b14bc81ab99f	Assigned from CIDR 10.0.0.0/19	Not applicable

ALB Resource Map (target health OK):

Load balancer: k8s-prod-eventsph-395841f6f1

us-east-1 | 277707118728



Output of kubectl describe ingress:

```
kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/Cyber/Sem 3/ENPM818R - Virtualization/Final project 2/EventSphere$ kubectl describe ingress -n prod
Name:           eventsphere-ingress
Labels:         app=eventsphere
                environment=production
Namespace:      prod
Address:        k8s-prod-eventsph-395841f6f1-1344915046.us-east-1.elb.amazonaws.com
Ingress Class:  alb
Default backend: <default>
Rules:
  Host          Path  Backends
  ----          ----
  www.enpm818rgroup7.work.gd
    /   frontend:80 (10.0.4.135:80,10.0.57.206:80)
  api.enpm818rgroup7.work.gd
    /api/auth       auth-service:4001 (10.0.55.26:4001,10.0.39.74:4001)
    /api/events     event-service:4002 (10.0.23.181:4002,10.0.27.88:4002)
    /api/bookings   booking-service:4003 (10.0.47.216:4003,10.0.57.233:4003)
    /api/notifications notification-service:4004 (10.0.54.188:4004,10.0.41.84:4004)
    /health         auth-service:4001 (10.0.55.26:4001,10.0.39.74:4001)
Annotations:
  alb.ingress.kubernetes.io/certificate-arn: arn:aws:acm:us-east-1:277707118728:certificate/2ab5f4c8-842c-447b-842f-b741fdbdb2126
  alb.ingress.kubernetes.io/healthcheck-interval-seconds: 30
  alb.ingress.kubernetes.io/healthcheck-path: /health
  alb.ingress.kubernetes.io/healthcheck-timeout-seconds: 5
  alb.ingress.kubernetes.io/healthy-threshold-count: 2
  alb.ingress.kubernetes.io/listen-ports: [{"HTTP": 80, "HTTPS": 443}]
  alb.ingress.kubernetes.io/load-balancer-attributes: idle_timeout.timeout_seconds=60
  alb.ingress.kubernetes.io/scheme: internet-facing
  alb.ingress.kubernetes.io/ssl-policy: ELBSecurityPolicy-TLS-1-2-2017-01
  alb.ingress.kubernetes.io/ssl-redirect: 443
  alb.ingress.kubernetes.io/tags: Environment=production,Project=EventSphere
  alb.ingress.kubernetes.io/target-type: ip
  alb.ingress.kubernetes.io/unhealthy-threshold-count: 3
Events:
```

Curl test results of API health endpoint confirm that the ALB is responding:

```
kunal@ASUS:/mnt/c/Users/kunal/OneDrive - University of Maryland/Cloudsphere$ curl -I https://www.enpm818rgroup7.work.gd/api/health
HTTP/2 200
date: Mon, 08 Dec 2025 15:52:55 GMT
content-type: text/html
content-length: 936
server: nginx/1.29.2
last-modified: Thu, 04 Dec 2025 07:32:27 GMT
etag: "6931390b-3a8"
accept-ranges: bytes
```

CI/CD Pipeline and Version Control

To manage the development across our team, we set up a dedicated GitHub Organization to centralize our code repositories and infrastructure files. We enforced a strict workflow where no developer was allowed to push code directly to the main branch.

Instead, every update, whether it was a fix for a microservice or a change to a Helm chart, required a Pull Request (PR). This allowed us to review each other's code for logic errors and potential security issues before integrating it.

We connected these PRs directly to our CI/CD pipeline. Whenever a pull request was created, our automated testing suite (GitHub Actions) would run immediately. We configured the repository settings so that a PR could only be merged into main after all the pipeline tests passed successfully. This "quality gate" ensured that we never accidentally broke the production build with faulty code.

Observability

Observability Implementation

The system implements a complete observability stack using **Prometheus**, **Grafana**, and **AWS CloudWatch**, deployed in an **Amazon EKS cluster using Helm**.

Metrics and Visualization

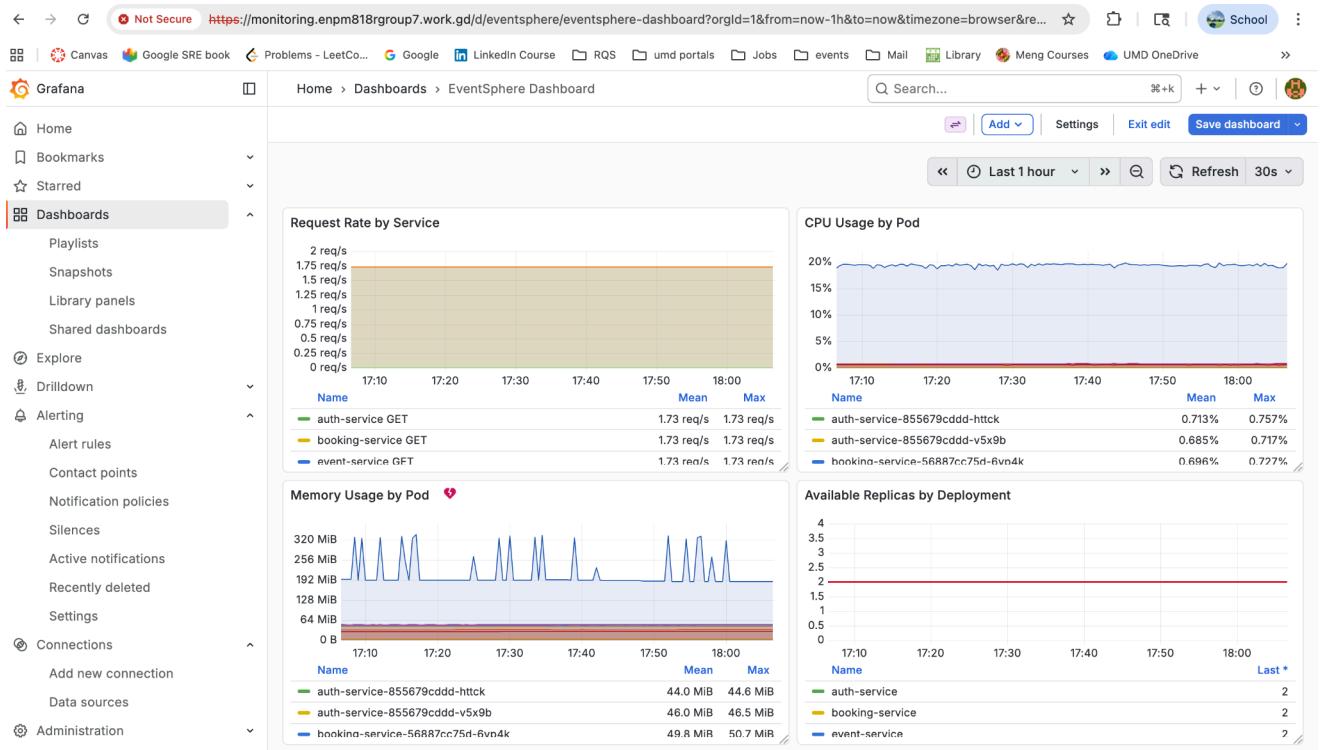
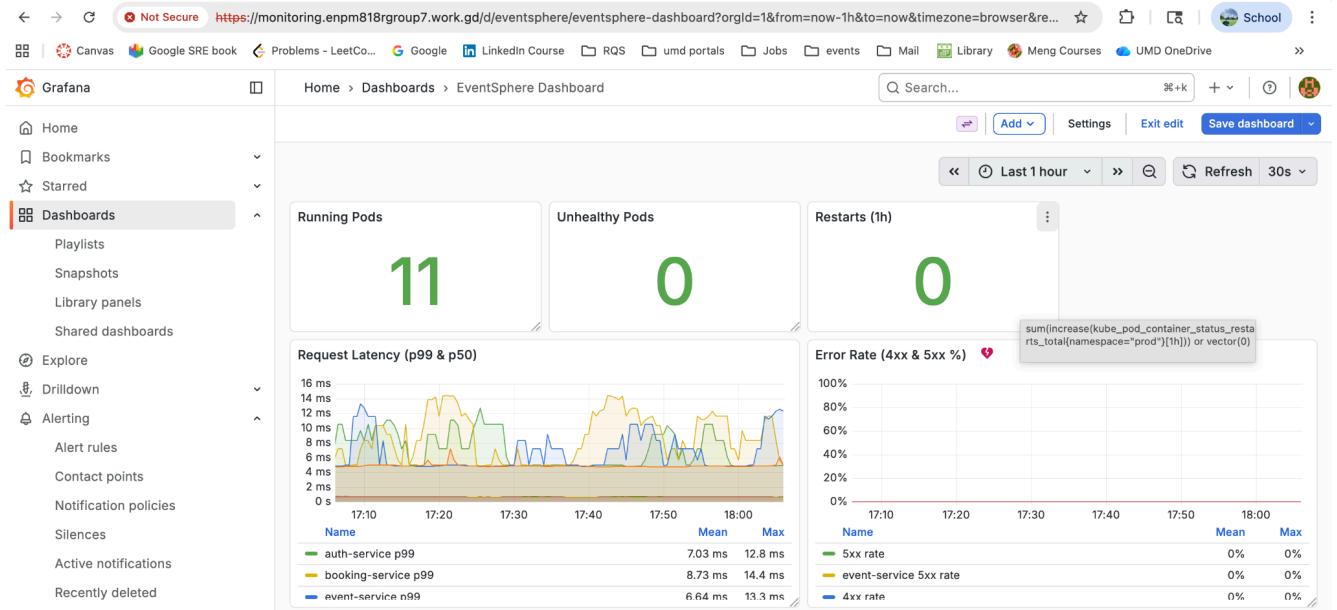
Prometheus scrapes Kubernetes and application-level metrics from all microservices running in the production namespace. Each service exposes a /metrics endpoint instrumented using Prometheus client libraries.

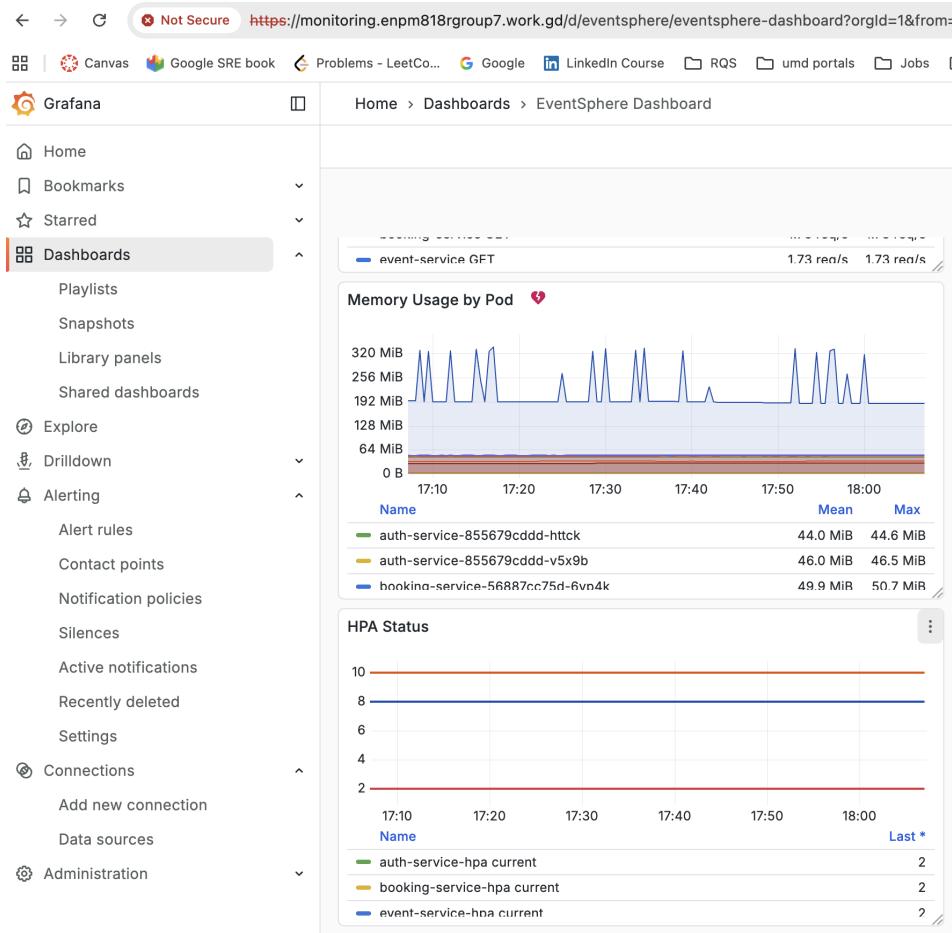
Grafana is used for metric visualization and alert evaluation through **Grafana Unified Alerting**. Unlike a port-forward-only setup, Grafana is exposed externally through an **AWS Application Load Balancer (ALB)** and accessed via DNS.

- **Grafana URL:** <https://monitoring.enpm818rgroup7.work.gd/>
- **Username:** admin
- **Password:** EventSphere2024
- **Dashboard:** *EventSphere Dashboard*

Grafana dashboards visualize:

1. Request latency (p99 and p50)
2. HTTP error rate (4xx & 5xx)
3. CPU and memory utilization by pod
4. Request rate per service
5. Pod status, restarts, and HPA state





Prometheus Access

The Prometheus server itself is not publicly exposed and is accessed securely using port forwarding when needed for inspection and validation.

kubectl port-forward -n monitoring svc/prometheus-kube-prometheus-prometheus 9090:9090

Prometheus UI:

<http://localhost:9090/alerts>

localhost:9090/alerts

Prometheus

Alerts

Status

Filter by rule group state

Filter by rule name or labels

1 2 >

eventsphere.deployment /etc/prometheus/rules/prometheus-prometheus-kube-prometheus-prometheus-rulefiles-0/monitoring-eventsphere-alerts-a7f7ae9d-d47a-4b12-8887-56b0158e2d57.yaml

INACTIVE (2)

- DeploymentReplicasMismatch
- HPAAAtMaxReplicas

eventsphere.pods /etc/prometheus/rules/prometheus-prometheus-kube-prometheus-prometheus-rulefiles-0/monitoring-eventsphere-alerts-a7f7ae9d-d47a-4b12-8887-56b0158e2d57.yaml

INACTIVE (2)

- PodCrashLooping
- PodNotReady

eventsphere.resources /etc/prometheus/rules/prometheus-prometheus-kube-prometheus-prometheus-rulefiles-0/monitoring-eventsphere-alerts-a7f7ae9d-d47a-4b12-8887-56b0158e2d57.yaml

INACTIVE (2)

- HighMemoryUsage

localhost:9090/alerts

Prometheus

Alerts

Status

alertmanager.rules

/etc/prometheus/rules/prometheus-prometheus-kube-prometheus-prometheus-rulefiles-0/monitoring-prometheus-kube-prometheus-alertmanager.rules-f281270a-0d5d-4fb-a654-a3d9cccd955.yaml

INACTIVE (8)

- AlertmanagerFailedReload
- AlertmanagerMembersInconsistent
- AlertmanagerFailedToSendAlerts
- AlertmanagerClusterFailedToSendAlerts
- AlertmanagerClusterFailedToSendAlerts
- AlertmanagerConfigInconsistent
- AlertmanagerClusterDown
- AlertmanagerClusterCrashlooping

config-reloaders

/etc/prometheus/rules/prometheus-prometheus-kube-prometheus-prometheus-rulefiles-0/monitoring-prometheus-kube-prometheus-config-reloaders-3106b860-ef4d-4d51-94a6-f93341fe85f9.yaml

INACTIVE (1)

- ConfigReloaderSidecarError

The screenshot shows the Prometheus Alerts interface at localhost:9090/alerts. The top navigation bar includes links like Canvas, Google SRE book, Problems - LeetCode, Google, LinkedIn Course, RQS, umd portals, Jobs, events, Mail, Library, Meng Courses, and UMD OneDrive. The main area displays two sections of alerts:

- kubernetes-storage**: Contains five inactive alerts: KubePersistentVolumeFillingUp, KubePersistentVolumeFillingUp, KubePersistentVolumeNodesFillingUp, KubePersistentVolumeNodesFillingUp, and KubePersistentVolumeErrors.
- INACTIVE (5)**: Contains five inactive alerts: KubePersistentVolumeFillingUp, KubePersistentVolumeFillingUp, KubePersistentVolumeNodesFillingUp, KubePersistentVolumeNodesFillingUp, and KubePersistentVolumeErrors.

Logging Pipeline

CloudWatch is used for centralized log ingestion and log analytics.

Application logs, Kubernetes events, and alerting-related logs are ingested into **CloudWatch** and stored in **structured JSON format**, enabling filtering and correlation by pod, service, and timestamp.

Role of CloudWatch in the Observability Stack

CloudWatch is used to:

- Store and query **application and Kubernetes control-plane logs**
- Enable **on-demand log analysis** using CloudWatch Logs and Logs Insights
- Provide visibility into Kubernetes audit logs and request-level events

This complements, but does not duplicate, the Prometheus–Grafana metrics pipeline.

Relevant log groups include:

- /aws/eks/eventsphere-cluster/cluster
- /aws/lambda/eventsphere-email-sender
- /aws/lambda/grafana-alerting

In this architecture:

- Metrics and alerting are handled by Prometheus and Grafana
- Logs and log analytics are handled by CloudWatch

This separation follows recommended cloud-native observability practices.

The screenshot shows the AWS CloudWatch Log management interface. On the left, there's a navigation sidebar with sections like Application Signals (APM), Infrastructure Monitoring, Logs (Log Management, Log Anomalies, Live Tail, Logs Insights, Contributor Insights), Metrics, Network Monitoring, Setup (Ingestion, Getting Started), and CloudWatch Metrics. The main content area is titled "Log groups (4)" and shows a table with four entries:

Log group	Log class	Anomaly d...	Deletio...	Data ...	Sens...	Retenti...
/aws/eks/eventsphere-cluster/application	Standard	Configure	Off	-	-	1 month
/aws/eks/eventsphere-cluster/cluster	Standard	Configure	Off	-	-	Never exp...
/aws/lambda/eventsphere-email-sender	Standard	Configure	Off	-	-	Never exp...
/aws/lambda/grafana-alerting	Standard	Configure	Off	-	-	Never exp...

The screenshot shows the AWS CloudWatch Log management interface, specifically the Log streams page for the log group `/aws/eks/eventsphere-cluster/cluster`. The left sidebar is identical to the previous screenshot. The main content area is titled "Log streams (12)" and shows a table with twelve entries:

Log stream	Last event time
kube-apiserver-a8390d0aa74a4e88356d7cb4183f297e	2025-12-07 23:16:25 (UTC)
kube-scheduler-de3d1856b0abd2c9a1d91d76b36c17b0	2025-12-07 23:05:09 (UTC)
kube-apiserver-audit-de3d1856b0abd2c9a1d91d76b36c17b0	2025-12-07 23:01:50 (UTC)
kube-apiserver-audit-a8390d0aa74a4e88356d7cb4183f297e	2025-12-07 23:01:45 (UTC)
authenticator-a8390d0aa74a4e88356d7cb4183f297e	2025-12-07 22:52:41 (UTC)
kube-apiserver-de3d1856b0abd2c9a1d91d76b36c17b0	2025-12-07 22:50:25 (UTC)
authenticator-de3d1856b0abd2c9a1d91d76b36c17b0	2025-12-07 22:49:40 (UTC)
kube-controller-manager-de3d1856b0abd2c9a1d91d76b36c17b0	2025-12-07 22:32:28 (UTC)
cloud-controller-manager-de3d1856b0abd2c9a1d91d76b36c17b0	2025-12-07 22:29:50 (UTC)
kube-scheduler-a8390d0aa74a4e88356d7cb4183f297e	2025-12-07 20:13:49 (UTC)
kube-controller-manager-a8390d0aa74a4e88356d7cb4183f297e	2025-12-06 19:04:57 (UTC)
cloud-controller-manager-a8390d0aa74a4e88356d7cb4183f297e	2025-12-06 19:04:51 (UTC)

AWS CloudWatch Logs Insights interface showing a query for log group arm:awslogsus-east-1:277707118728:log-group:/aws/eks/eventsphere-cluster/cluster. The query filters for @message like /error|Error|ERROR|exception|Exception|failed|Failed/ and sorts by @timestamp desc, limit 100.

Log class: Standard

Queries

Saved queries

Sample queries

- Common queries
- Lambda
- VPC Flow Logs
- CloudTrail
- NetworkFirewall
- Route53
- AWS AppSync
- NAT Gateway
- IoT
- Elemental MediaPackage V2 Access Logs
- SES Mail Manager
- Amazon Q Business Conversation Log
- CloudFront Standard Logs

CloudWatch Dashboards interface showing the eventsphere dashboard. The main view displays a log stream with 231 records matched, 59,477 records scanned in 1.8s at 33,376 records/s (45.8 MB/s). The log stream shows audit events from kube-apiserver and audit-de3d18560abd29a1d91d7636c17b0.

Logs (100)

Requests by service

Legend:

1. kube-apiserver-audit-de3d18560abd29a1d91d7636c17b0
2. kube-apiserver-audit-a83900daa74a4e88356d7db183f297e
3. kube-controller-manager-d034...
4. kube-apiserver-audit-03900da...
5. kube-apiserver-audit-03900da...
6. authenticator-a3d7800da...
7. authenticator-de3d18560abd...
8. cloud-controller-manager-d0...
9. kube-scheduler-d3d18560ab...

Alerting and Notifications

Alert rules are evaluated using **Grafana Unified Alerting**, and notifications are delivered through an AWS-native pipeline:

Grafana → Webhook → AWS Lambda → Amazon SNS → Email

The screenshot shows the Grafana Alerting interface. On the left, the sidebar is open with the 'Alert rules' option selected. The main area displays a table of 185 rules, with 8 firing, 128 normal, and 49 recording. The first three rows in the table represent the configured alerts:

State	Name	Health	Summary	Next evaluation	Actions
Firing	Error Rate (4xx & 5xx %)	ok	p99 request latency exceeded 1 second for one or more services.	in a few seconds	View Edit More
Normal	Pod Restarts	ok	One or more containers restarted unexpectedly in the production namespace.	in a few seconds	View Edit More
Firing	Memory Usage by Pod	ok	Memory usage exceeded the defined threshold for one or more pods.	in a few seconds	View Edit More

Alerting and Runbooks Implementation

We implemented **three production-grade alert rules** using **Grafana Unified Alerting (Grafana-managed Alert manager)**. Alert notifications are delivered through an **AWS-integrated pipeline**, consisting of **Grafana webhook notifications routed through AWS Lambda and delivered via Amazon SNS (email)**.

The alerts are actively evaluated and observed in a **firing state**, as shown in the Grafana Alert Rules view.

Configured Alerts

1. **Error Rate (4xx & 5xx)**
 - Monitors elevated HTTP error rates across application services
 - Triggers when the error percentage exceeds the defined SLO threshold
2. **Memory Usage by Pod**
 - Tracks pod-level memory consumption using Kubernetes container metrics
 - Triggers when memory usage exceeds the configured threshold for a sustained period

3. Pod Restart Spike

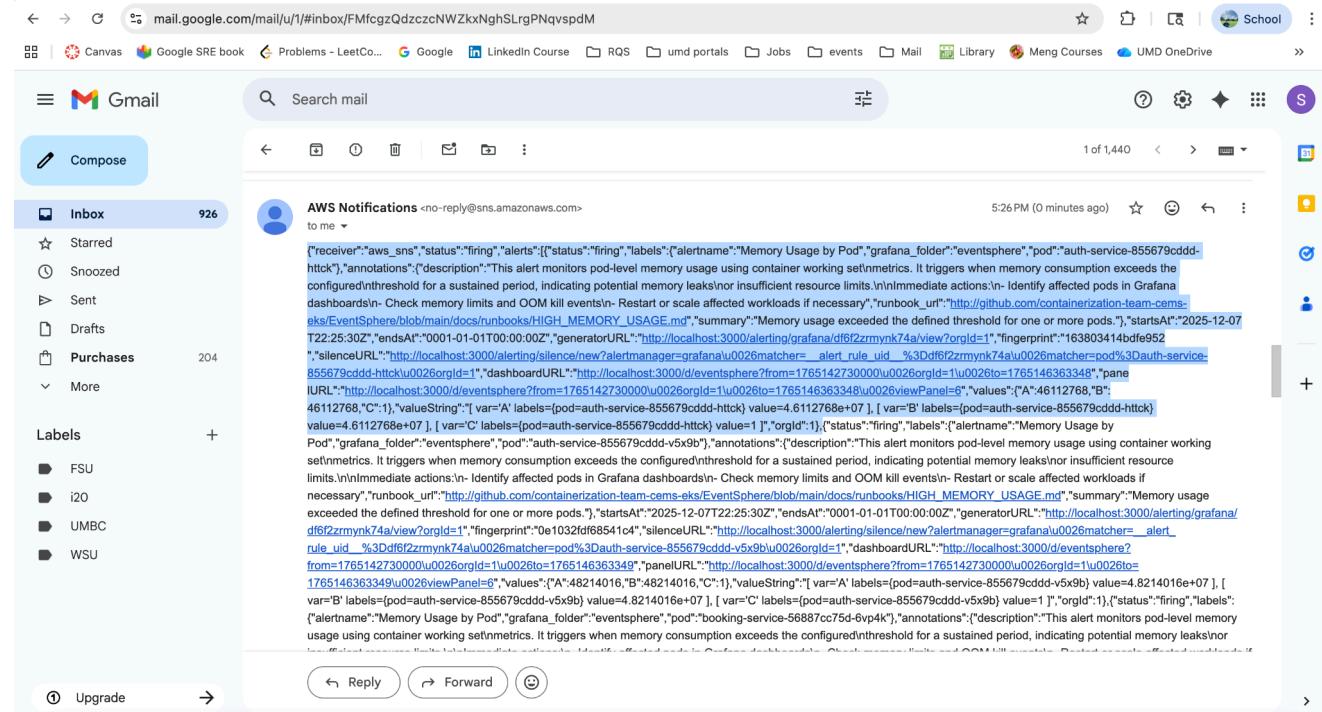
- Detects unexpected container restarts within the production namespace
 - Indicates potential crash loops, failed deployments, or resource misconfigurations

All alert rules are evaluated by **Grafana Alert manager**, while notifications are delivered using **Amazon SNS**, invoked via an **AWS Lambda webhook integration**.

Runbooks

For each alert, a corresponding operational runbook has been created. Each runbook documents possible root causes, investigation steps, and remediation actions.

- **Error Rate Alert Runbook**
https://github.com/containerization-team-cems-eks/EventSphere/blob/main/docs/runbooks/ERROR_RATE.md
 - **Pod Restart Spike Runbook**
https://github.com/containerization-team-cems-eks/EventSphere/blob/main/docs/runbooks/POD_RESTART_SPIKE.md
 - **High Memory Usage Runbook**
https://github.com/containerization-team-cems-eks/EventSphere/blob/main/docs/runbooks/HIGH_MEMORY_USAGE.md



This email is an automated **AWS SNS notification triggered by a Grafana alert** indicating **high memory usage at the pod level**. The alert fires when one or more Kubernetes pods

exceed the configured memory threshold for a sustained period, signaling potential memory leaks, misconfigured resource limits, or abnormal traffic patterns. Along with detailed metadata such as the affected pod, timestamps, and alert labels, the notification **includes links to Grafana dashboards and an attached runbook**, providing clear, predefined steps for investigation and remediation to reduce response time and operational guesswork.

Three Pillars of Observability

1. Metrics (Prometheus + Grafana)

Component	Description
prom-client	Each service exposes /metrics with latency histograms and request counters
middleware	Scrapes metrics from all pods and stores time-series data
Prometheus	Custom dashboard (eventsphere-dashboard.json) for visualization
Grafana	Evaluates alert rules and triggers notifications

Key metrics collected:

- http_request_duration_seconds – latency histogram
- http_requests_total – request count by method/route/status
- Default Node.js metrics (CPU, memory, event loop)

2. Logs (Fluent Bit → CloudWatch)

Component	Description
Fluent Bit	DaemonSet tailing /var/log/containers/*_prod_*.log
CloudWatch	Centralized log storage and search

3. Alerting

Alert rules are defined and managed using Helm configuration and evaluated by Grafana Alertmanager.

Configured Alerts:

Alert	Trigger	Severity

PodCrashLooping	Restart rate > 0 for 5m	Critical
PodNotReady	Pending/Failed > 10m	Warning
HighMemoryUsage	> 90% limit for 5m	Warning
HighCPUUsage	> 80% limit for 10m	Warning
DeploymentReplicasMismatc h	Desired ≠ available for 10m	Warning
HPAAtMaxReplicas	Max replicas for 15m	Warning

Deployment Using Helm

All observability components in the EventSphere platform are deployed to the Amazon EKS cluster using **Helm**, ensuring consistency, repeatability, and version-controlled configuration.

Helm charts are used to deploy and manage:

- Prometheus (metrics scraping and storage)
- Grafana (dashboards and alerting)
- Alert rules (Prometheus Rule and Grafana alert definitions)
- Fluent Bit (log forwarding to AWS CloudWatch)

Using Helm allows centralized configuration management, simplified upgrades, and reproducible deployments across environments.

The Helm-based deployment confirms that the observability stack follows cloud-native and production-aligned deployment practices rather than ad-hoc manual configuration.

```
sakethbolla@sakeths-MacBook-Air-2 ~ % kubectl get pods -n monitoring --show-labels | grep helm
prometheus-grafana-5965f805c-1d908      3/3   Running   0    3h54m   app.kubernetes.io/instance=prometheus,app.kubernetes.io/name=grafana,app.kubernetes.io/version=2.1.3,app.helm.sh/chart=grafana-18.3.0,app.kubernetes.io/pod-template-hash=5965f805c
prometheus-state-metrics-exporter-4cfb1b- 1/1   Running   0    3h54s   app.kubernetes.io/component=metrics,app.kubernetes.io/instance=prometheus,app.kubernetes.io/name=prometheus-state-metrics-exporter,app.kubernetes.io/part-of=prometheus-node-exporter,app.kubernetes.io/released-prometheus-node-exporter-v4cfb1b,app.kubernetes.io/version=1.10.2,controller-revision-hash=5d7db7286,release=prometheus
prometheus-prometheus-node-exporter-4cfb6- 1/1   Running   0    3h54m   app.kubernetes.io/component=metrics,app.kubernetes.io/instance=prometheus,app.kubernetes.io/name=prometheus-node-exporter,app.kubernetes.io/part-of=prometheus-node-exporter,app.kubernetes.io/released-prometheus-node-exporter-v4cfb6,app.kubernetes.io/version=1.10.2,controller-revision-hash=66df454fc,helm.sh/chart=prometheus-node-exporter-4.49.2
prometheus-prometheus-node-exporter-4cfb7- 1/1   Running   0    3h54m   app.kubernetes.io/component=metrics,app.kubernetes.io/instance=prometheus,app.kubernetes.io/name=prometheus-node-exporter,app.kubernetes.io/part-of=prometheus-node-exporter,app.kubernetes.io/released-prometheus-node-exporter-v4cfb7,app.kubernetes.io/version=1.10.2,controller-revision-hash=7a5e0454fc,helm.sh/chart=prometheus-node-exporter-4.49.2
prometheus-prometheus-node-exporter-4cfb8- 1/1   Running   0    3h54m   app.kubernetes.io/component=metrics,app.kubernetes.io/instance=prometheus,app.kubernetes.io/name=prometheus-node-exporter,app.kubernetes.io/part-of=prometheus-node-exporter,app.kubernetes.io/released-prometheus-node-exporter-v4cfb8,app.kubernetes.io/version=1.10.2,controller-revision-hash=868f454fc,helm.sh/chart=prometheus-node-exporter-4.49.2
prometheus-prometheus-node-exporter-4cfb9- 1/1   Running   0    3h54m   app.kubernetes.io/component=metrics,app.kubernetes.io/instance=prometheus,app.kubernetes.io/name=prometheus-node-exporter,app.kubernetes.io/part-of=prometheus-node-exporter,app.kubernetes.io/released-prometheus-node-exporter-v4cfb9,app.kubernetes.io/version=1.10.2,controller-revision-hash=968f454fc,helm.sh/chart=prometheus-node-exporter-4.49.2
prometheus-prometheus-node-exporter-4cfba- 0/1   Pending   0    17h    app.kubernetes.io/component=metrics,app.kubernetes.io/instance=prometheus,app.kubernetes.io/name=prometheus-node-exporter,app.kubernetes.io/part-of=prometheus-node-exporter,app.kubernetes.io/released-prometheus-node-exporter-v4cfba,app.kubernetes.io/version=1.10.2,controller-revision-hash=a68f454fc,helm.sh/chart=prometheus-node-exporter-4.49.2
prometheus-prometheus-node-exporter-4cfbb- 0/1   Pending   0    17h    app.kubernetes.io/component=metrics,app.kubernetes.io/instance=prometheus,app.kubernetes.io/name=prometheus-node-exporter,app.kubernetes.io/part-of=prometheus-node-exporter,app.kubernetes.io/released-prometheus-node-exporter-v4cfbb,app.kubernetes.io/version=1.10.2,controller-revision-hash=b68f454fc,helm.sh/chart=prometheus-node-exporter-4.49.2
sakethbolla@sakeths-MacBook-Air-2 ~ %
```

```
sakethbolla@sakeths-MacBook-Air-2 ~ % helm list -A
NAME          NAMESPACE      REVISION      UPDATED           VERSION
aws-load-balancer-controller  kube-system   1            2025-12-06 14:13:51.2220488 -0500 EST
external-secrets      external-secrets-system 1            2025-12-06 14:14:42.9649706 -0500 EST
prometheus          monitoring     1            2025-12-06 14:30:17.1225593 -0500 EST
```

```
sakethbolla@Sakeths-MacBook-Air-2 ~ % kubectl get pods -n monitoring
NAME                                     READY   STATUS    RESTARTS   AGE
alertmanager-prometheus-kube-prometheus-alertmanager-0   2/2     Running   0          3h57m
prometheus-grafana-5965ff89c5-ld998      3/3     Running   0          3h57m
prometheus-kube-prometheus-operator-6fc45cb74f-h7t6g      1/1     Running   0          3h57m
prometheus-kube-state-metrics-5d7bdb7c86-7fhbl      1/1     Running   0          3h57m
prometheus-prometheus-kube-prometheus-prometheus-0      2/2     Running   0          3h57m
prometheus-prometheus-node-exporter-4cj6       1/1     Running   0          3h57m
prometheus-prometheus-node-exporter-lhxgf      1/1     Running   0          3h57m
prometheus-prometheus-node-exporter-svfb6      1/1     Running   0          3h56m
prometheus-prometheus-node-exporter-x4jbp      1/1     Running   0          17h
prometheus-prometheus-node-exporter-xj5zr       0/1     Pending   0          3h57m
```

```
sakethbolla@Sakeths-MacBook-Air-2 ~ % kubectl get svc -n monitoring
NAME           TYPE        CLUSTER-IP   EXTERNAL-IP  PORT(S)   AGE
alertmanager-operated  ClusterIP  None         <none>      9093/TCP, 9094/TCP, 9094/UDP  28h
prometheus-grafana  ClusterIP  172.20.65.33  <none>      80/TCP     28h
prometheus-kube-prometheus-alertmanager  ClusterIP  172.20.35.219 <none>      9093/TCP, 8080/TCP  28h
prometheus-kube-prometheus-operator  ClusterIP  172.20.31.130 <none>      443/TCP    28h
prometheus-kube-prometheus-prometheus  ClusterIP  172.20.51.12  <none>      9090/TCP, 8080/TCP  28h
prometheus-kube-state-metrics  ClusterIP  172.20.235.236 <none>      8080/TCP    28h
prometheus-operated  ClusterIP  None         <none>      9090/TCP    28h
prometheus-prometheus-node-exporter  ClusterIP  172.20.139.39 <none>      9100/TCP    28h
sakethbolla@Sakeths-MacBook-Air-2 ~ %
```

Deliverable

Prometheus/Grafana Helm
Values

PrometheusRule (Alerts)

Grafana Dashboard JSON

Fluent Bit Config

File Path

[monitoring/prometheus/values.yaml](#)

[monitoring/prometheus/alertrules.yaml](#)

[monitoring/grafana/dashboards/eventsphere-dashboard.json](#)

[monitoring/cloudwatch/fluent-bit-config.yaml](#)

```
EventSphere / monitoring / prometheus / values.yaml
Code Blame 92 lines (86 loc) · 1.9 KB ↑ Top
4 # Reduce resource usage for smaller clusters
5 prometheus:
6   prometheusSpec:
7     retention: 7d
8     resources:
9       requests:
10      memory: 256Mi
11      cpu: 100m
12     limits:
13      memory: 1Gi
14      cpu: 500m
15 # Use emptyDir instead of PVC for simplicity (data lost on restart, but works everywhere)
16   storageSpec: {}
17
18 alertmanager:
19   enabled: true
20   alertmanagerSpec:
21     resources:
22       requests:
23         memory: 64Mi
24         cpu: 25m
25       limits:
26         memory: 128Mi
27         cpu: 100m
28   storage: {}
29
30 grafana:
31   enabled: true
32   adminUser: admin
33   adminPassword: "EventSphere2024"
34   resources:
35     requests:
36       memory: 128Mi
37       cpu: 50m
38     limits:
```

Deployment Methodology

All observability components, including **Prometheus**, **Grafana**, **Alerting rules**, and **Fluent Bit**, are deployed into the EKS cluster **using Helm charts**, ensuring reproducibility and consistency.

Amazon Elastic Kubernetes Service > Clusters > eventsphere-cluster

Resource types

- Workloads
- PodTemplates
- Pods
- ReplicaSets
- Deployments
- StatefulSets**
- DaemonSets
- Jobs
- CronJobs
- PriorityClasses
- HorizontalPodAutoscalers
- ResourceClaims
- ResourceClaimTemplates
- ResourceSlices

Workloads: StatefulSets (3)

Statefulset manages the deployment and scaling of a set of Pods, and provides guarantees about the ordering and uniqueness of these Pods.

Learn more [↗](#)

Name	Namespace	Type	Created	Pod count	Status
alertmanager-prometheus-kube-prometheus-alertmanager	monitoring	statefulsets	16 hours ago	1	<div style="width: 100%; background-color: green;"></div> 1 Ready 0 Failed 1 Desired
mongodb	prod	statefulsets	17 hours ago	1	<div style="width: 100%; background-color: green;"></div> 1 Ready 0 Failed 1 Desired
prometheus-prometheus-kube-prometheus-prometheus	monitoring	statefulsets	16 hours ago	1	<div style="width: 100%; background-color: green;"></div> 1 Ready 0 Failed 1 Desired

[View details](#)

Security and Compliance

All Images are taken from trusted sources. Images are scanned for vulnerabilities with trivy, and rejected if there are critical vulnerabilities. Database is encrypted, and TLS is enforced for all data in transit. Trivy Scan reports:

eventsphere-auth-service:latest (alpine 3.23.0) - Trivy Report - 2025-12-09 12:03:33.33226948 -0500 EST m=+0.794909274

alpine					
No Vulnerabilities found					
No Misconfigurations found					
node-pkg					
Package	Vulnerability ID	Severity	Installed Version	Fixed Version	Links
glob	CVE-2025-64756	HIGH	10.4.5	11.1.0, 10.5.0	https://access.redhat.com/security/cve/CVE-2025-64756 https://github.com/isaacs/node-glob https://github.com/isaacs/node-glob/commit/1e4e297342a09f2aa0ced87fc4a70ddc325d75f Toggle more links
glob	CVE-2025-64756	HIGH	11.0.3	11.1.0, 10.5.0	https://access.redhat.com/security/cve/CVE-2025-64756 https://github.com/isaacs/node-glob https://github.com/isaacs/node-glob/commit/1e4e297342a09f2aa0ced87fc4a70ddc325d75f Toggle more links
tar	CVE-2025-64118	MEDIUM	7.5.1	7.5.2	https://github.com/isaacs/node-tar https://github.com/isaacs/node-tar/commit/5330eb04bc43014f216e5c271b40d5c00d45224d https://github.com/isaacs/node-tar/commit/5e1a8e638600d3c3a2969b4de6a6e44fa8d74c9 Toggle more links
No Misconfigurations found					

eventsphere-booking-service:latest (alpine 3.23.0) - Trivy Report - 2025-12-09 12:08:07.252320747 -0500 EST m=+1.756425588

alpine					
No Vulnerabilities found					
No Misconfigurations found					
node-pkg					
Package	Vulnerability ID	Severity	Installed Version	Fixed Version	Links
glob	CVE-2025-64756	HIGH	10.4.5	11.1.0, 10.5.0	https://access.redhat.com/security/cve/CVE-2025-64756 https://github.com/isaacs/node-glob https://github.com/isaacs/node-glob/commit/1e4e297342a09f2aa0ced87fc4a70ddc325d75f Toggle more links
glob	CVE-2025-64756	HIGH	11.0.3	11.1.0, 10.5.0	https://access.redhat.com/security/cve/CVE-2025-64756 https://github.com/isaacs/node-glob https://github.com/isaacs/node-glob/commit/1e4e297342a09f2aa0ced87fc4a70ddc325d75f Toggle more links
tar	CVE-2025-64118	MEDIUM	7.5.1	7.5.2	https://github.com/isaacs/node-tar https://github.com/isaacs/node-tar/commit/5330eb04bc43014f216e5c271b40d5c00d45224d https://github.com/isaacs/node-tar/commit/5e1a8e638600d3c3a2969b4de6a6e44fa8d74c9 Toggle more links
No Misconfigurations found					

eventsphere-event-service:latest (alpine 3.23.0) - Trivy Report - 2025-12-09 12:08:01.892374016 -0500 EST m=+2.317336817

alpine					
No Vulnerabilities found					
No Misconfigurations found					
node-pkg					
Package	Vulnerability ID	Severity	Installed Version	Fixed Version	Links
glob	CVE-2025-64756	HIGH	10.4.5	11.1.0, 10.5.0	https://access.redhat.com/security/cve/CVE-2025-64756 https://github.com/isaacs/node-glob https://github.com/isaacs/node-glob/commit/1e4e297342a09f2aa0ced87fc4a70ddc325d75f Toggle more links
glob	CVE-2025-64756	HIGH	11.0.3	11.1.0, 10.5.0	https://access.redhat.com/security/cve/CVE-2025-64756 https://github.com/isaacs/node-glob https://github.com/isaacs/node-glob/commit/1e4e297342a09f2aa0ced87fc4a70ddc325d75f Toggle more links
tar	CVE-2025-64118	MEDIUM	7.5.1	7.5.2	https://github.com/isaacs/node-tar https://github.com/isaacs/node-tar/commit/5330eb04bc43014f216e5c271b40d5c00d45224d https://github.com/isaacs/node-tar/commit/5e1a8e638600d3c3a2969b4de6a6e44fa8d74c9 Toggle more links
No Misconfigurations found					

eventsphere-notification-service:latest (alpine 3.23.0) - Trivy Report - 2025-12-09 12:03:50.955954529 -0500 EST m=+0.496271750

alpine					
No Vulnerabilities found					
No Misconfigurations found					
node-pkg					
Package	Vulnerability ID	Severity	Installed Version	Fixed Version	Links
glob	CVE-2025-64756	HIGH	10.4.5	11.1.0, 10.5.0	https://access.redhat.com/security/cve/CVE-2025-64756 https://github.com/isaacs/node-glob https://github.com/isaacs/node-glob/commit/1e4a207342a09f2aa0ced87fd4a70ddc325d75f Toggle more links
glob	CVE-2025-64756	HIGH	11.0.3	11.1.0, 10.5.0	https://access.redhat.com/security/cve/CVE-2025-64756 https://github.com/isaacs/node-glob https://github.com/isaacs/node-glob/commit/1e4a207342a09f2aa0ced87fd4a70ddc325d75f Toggle more links
tar	CVE-2025-64118	MEDIUM	7.5.1	7.5.2	https://github.com/isaacs/node-tar https://github.com/isaacs/node-tar/commit/5330eb04bc43014f218e5c271b40d5c00d45224d https://github.com/isaacs/node-tar/commit/5e1a8ed38900d3c3a29c9b4de9a0ec44fb8d74c9 Toggle more links
No Misconfigurations found					

eventsphere-frontend:latest (alpine 3.22.2) - Trivy Report - 2025-12-09 11:59:09.832655202 -0500 EST m=+0.854720586

alpine					
Package	Vulnerability ID	Severity	Installed Version	Fixed Version	Links
busybox	CVE-2024-58251	MEDIUM	1.37.0-r19	1.37.0-r20	http://www.openwall.com/lists/oss-security/2025/04/23/6 https://bugs.busybox.net/show_bug.cgi?id=15922 https://www.busybox.net Toggle more links
busybox	CVE-2025-46394	LOW	1.37.0-r19	1.37.0-r20	http://www.openwall.com/lists/oss-security/2025/04/23/5 http://www.openwall.com/lists/oss-security/2025/04/24/3 https://bugs.busybox.net/show_bug.cgi?id=16018 Toggle more links
busybox-binsh	CVE-2024-58251	MEDIUM	1.37.0-r19	1.37.0-r20	http://www.openwall.com/lists/oss-security/2025/04/23/6 https://bugs.busybox.net/show_bug.cgi?id=15922 https://www.busybox.net Toggle more links
busybox-binsh	CVE-2025-46394	LOW	1.37.0-r19	1.37.0-r20	http://www.openwall.com/lists/oss-security/2025/04/23/5 http://www.openwall.com/lists/oss-security/2025/04/24/3 https://bugs.busybox.net/show_bug.cgi?id=16018 Toggle more links
libpng	CVE-2025-64720	HIGH	1.6.47-r0	1.6.51-r0	https://access.redhat.com/security/cve/CVE-2025-64720 https://github.com/pnggroup/libpng/commit/08da3394c88cfcd36e5a706558a8d7e0e4773643 https://github.com/pnggroup/libpng/issues/686 Toggle more links
libpng	CVE-2025-65018	HIGH	1.6.47-r0	1.6.51-r0	https://access.redhat.com/security/cve/CVE-2025-65018 https://github.com/pnggroup/libpng/commit/106e3823918840aae65c0a6da57c78a5a496a4d https://github.com/pnggroup/libpng/commit/218612dd9b17944e21eda56cafb4bf7779d1ea Toggle more links
libpng	CVE-2025-66293	HIGH	1.6.47-r0	1.6.53-r0	http://www.openwall.com/lists/oss-security/2025/12/03/8 http://www.openwall.com/lists/oss-security/2025/12/03/7 http://www.openwall.com/lists/oss-security/2025/12/03/8 Toggle more links
libpng	CVE-2025-64505	MEDIUM	1.6.47-r0	1.6.51-r0	https://access.redhat.com/security/cve/CVE-2025-64505 https://github.com/pnggroup/libpng/commit/6a528eb51d0dd7f6de1c39d30de0e41473431c37 https://github.com/pnggroup/libpng/commit/6a528eb51d0dd7f6de1c39d30de0e41473431c37.(v1.6.51) Toggle more links
libpng	CVE-2025-64506	MEDIUM	1.6.47-r0	1.6.51-r0	https://access.redhat.com/security/cve/CVE-2025-64506 https://github.com/pnggroup/libpng/commit/2bd84c019c300b78e811743fbcd67c9dbf821 https://github.com/pnggroup/libpng/pull/749 Toggle more links
ssl_client	CVE-2024-58251	MEDIUM	1.37.0-r19	1.37.0-r20	http://www.openwall.com/lists/oss-security/2025/04/23/6 https://bugs.busybox.net/show_bug.cgi?id=15922 https://www.busybox.net Toggle more links
ssl_client	CVE-2025-46394	LOW	1.37.0-r19	1.37.0-r20	http://www.openwall.com/lists/oss-security/2025/04/23/5 http://www.openwall.com/lists/oss-security/2025/04/24/3 https://bugs.busybox.net/show_bug.cgi?id=16018 Toggle more links
No Misconfigurations found					

Once we analyze the trivy reports, we mitigate each of the vulnerabilities found and do a report once again. If a fix isn't available, we can setup alerts and monitor those images closely with our monitoring stack, while waiting for the vendor to publish a fix.

Innovation

In developing EventSphere, our team moved beyond simple containerization to implement an operations-ready platform. The primary innovation lies in our automated resiliency and proactive security capabilities.

Dynamic Auto-Scaling instead of static provisioning: We implemented a dual-layer scaling strategy. We configured the Cluster Autoscaler to watch for pending pods and automatically provision new EC2 nodes when resources run low. We validated this by running a load test that pushed the cluster from 2 nodes up to 5 nodes automatically when traffic spiked, and then scaled it back down when the load was removed. We also implemented Horizontal Pod Autoscalers (HPA) for individual services like the auth-service and booking-service to scale pod replicas based on CPU demand.

DevSecOps Integration: We integrated security directly into our pipeline. We used Trivy to scan our container images for vulnerabilities before deployment. For example, our scans of the auth-service identified specific package vulnerabilities, allowing us to address risks before they hit production.

Production-Grade Alerting & Runbooks: Rather than just staring at dashboards, we set up an automated alerting pipeline. If error rates spike (4xx/5xx errors) or memory usage gets too high, Grafana triggers a webhook that routes through AWS Lambda to Amazon SNS, sending us an email notification immediately. Crucially, each alert is paired with a specific Runbook (hosted on GitHub) that tells the on-call engineer exactly how to diagnose the crash loop or memory leak, significantly reducing time-to-recovery.

Notification Service

1. Introduction:

The Notification Service is a cloud-native microservice responsible for delivering transactional emails—including welcome emails and booking confirmations—in the EventSphere system. It leverages Amazon SNS for message brokering, AWS Lambda for serverless processing, and Amazon SES for authenticated email delivery. The service runs on AWS EKS and uses IRSA (IAM Roles for Service Accounts) for secure AWS integration.

2. Architecture Overview

The notification pipeline follows an event-driven architecture:

User registers / books an event

Auth or Booking Service calls Notification Service REST endpoint

Notification Service publishes a structured message to SNS

SNS invokes Lambda

Lambda sends email via SES

SES delivers to end-user inbox

3. Technology Stack

Node.js + Express backend service
AWS SNS – asynchronous messaging
AWS Lambda – email processing
AWS SES – email delivery
Amazon EKS – container orchestration
Amazon ECR – container registry
Kubernetes HPA – auto-scaling
IAM + IRSA – identity and permissions

4.1 Notification Service (Microservice)

Handles REST API requests and publishes messages to SNS.

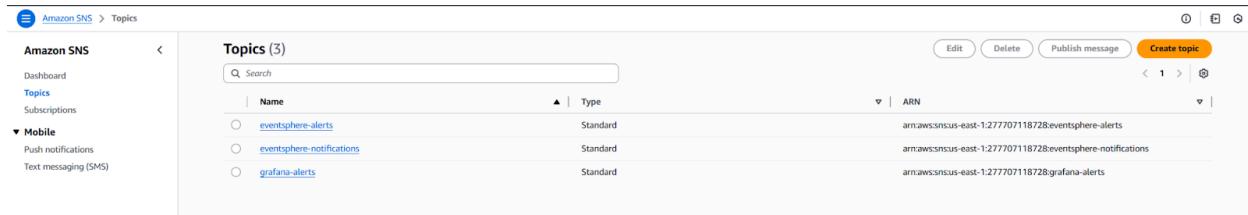
Endpoints:

POST /api/notifications/welcome
POST /api/notifications/booking-confirmation
GET /health

4.2 Amazon SNS (Message Broker)

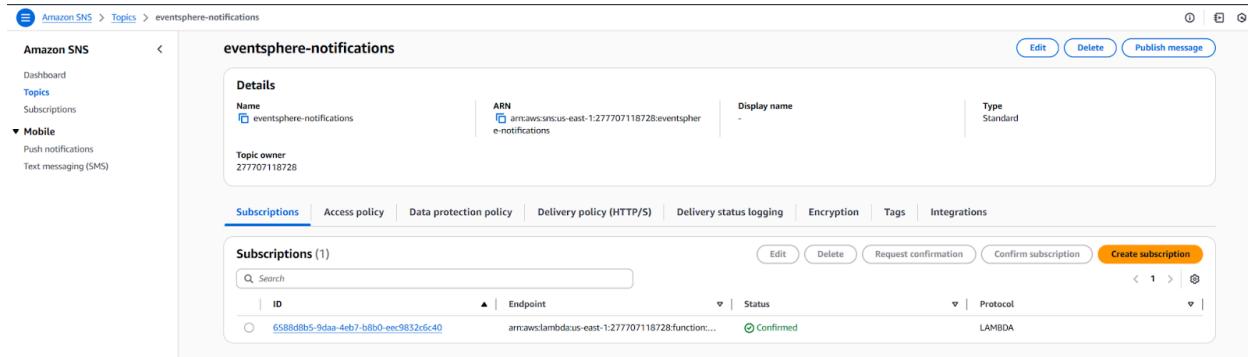
SNS topic receives messages from the Notification Service and triggers Lambda.

Displays the SNS Topic List page, confirming that the topic *eventsphere-notifications* exists in the correct region.



The screenshot shows the 'Topics' section of the Amazon SNS console. It lists three topics: 'eventsphere-alerts', 'eventsphere-notifications', and 'grafana-alerts'. Each topic entry includes its name, type (Standard), ARN, and a small preview icon. The 'eventsphere-notifications' topic is highlighted.

Shows the SNS Topic Details page, including the Topic ARN and the number of connected subscriptions, verifying correct configuration and availability.



The screenshot shows the 'Details' page for the 'eventsphere-notifications' topic. It displays the topic's name, ARN, display name (empty), and type (Standard). Below this, the 'Subscriptions' tab is selected, showing one subscription. The subscription details include the ID (6588dbb5-9daa-4eb7-bb0-eec9832c6c40), endpoint (arn:aws:lambda:us-east-1:277707118728:function:...), status (Confirmed), and protocol (LAMBDA).

Provides the SNS Subscriptions page, confirming that the Notification Service's SNS topic is successfully subscribed to the Lambda function and that the subscription status is *Confirmed*.

Amazon SNS > Topics > eventsphere-notifications > Subscription: 6588d8b5-9daa-4eb7-b8b0-eec9832c6c40

Amazon SNS

Dashboard
Topics
Subscriptions

▼ Mobile
Push notifications
Text messaging (SMS)

Subscription: 6588d8b5-9daa-4eb7-b8b0-eec9832c6c40

Details

ARN
arn:aws:sns:us-east-1:277707118728:eventsphere-notifications:6588d8b5-9daa-4eb7-b8b0-eec9832c6c40

Endpoint
arn:aws:lambda:us-east-1:277707118728:function:eventsphere-email-sender

Topic
[eventsphere-notifications](#)

Subscription Principal
arn:aws:iam::277707118728:user/admin

Status
Confirmed

Protocol
LAMBOA

[Edit](#) [Delete](#)

Subscription filter policy [Info](#) [Redrive policy \(dead-letter queue\)](#)

Subscription filter policy [Info](#)

This policy filters the messages that a subscriber receives.

No filter policy configured for this subscription.
To apply a filter policy, edit this subscription.

[Edit](#)

4.3 AWS Lambda (Email Processor)

Lambda receives SNS messages and sends emails using SES.

Shows the Lambda Code interface, index.js file implements SNS event handling and SES email sending.

The screenshot shows the AWS Lambda Code source interface. The left sidebar displays the project structure under 'EXPLORER' and 'TEST EVENTS'. The main area shows the 'index.js' file content:

```
index.js
6 * Flow:
7 * 1. SNS receives message from notification service
8 * 2. SNS triggers this Lambda function
9 * 3. Lambda extracts email and message from SNS event
10 * 4. Lambda sends email via SES to specific user
11 */
12
13 const { SESClient, SendEmailCommand } = require('@aws-sdk/client-ses');
14
15 // AWS_REGION is automatically available in Lambda environment
16 const sesClient = new SESClient({ region: process.env.AWS_REGION || 'us-east-1' });
17 const FROM_EMAIL = process.env.FROM_EMAIL || 'noreply@example.com';
18
19 exports.handler = async (event) => {
20     console.log('Received SNS event:', JSON.stringify(event, null, 2));
21
22     try {
23         // Process each SNS record
24         for (const record of event.Records) {
25             if (record.EventSource === 'aws:sns') {
26                 const snsMessage = record.sns;
27
28                 // Extract email from message attributes
29                 const emailAttribute = snsMessage.MessageAttributes?.email;
30                 const toEmail = emailAttribute?.Value;
31
32                 if (!toEmail) {
33                     console.error('No email found in message attributes');
34                     continue;
35                 }
36             }
37         }
38     } catch (error) {
39         console.error(error);
40     }
41 }
42
```

Displays the Lambda Trigger configuration, confirming that the Lambda function is correctly triggered by the SNS topic defined earlier.

The screenshot shows the AWS Lambda Function Overview page for the function 'eventsphere-email-sender'. The top navigation bar includes 'Lambda > Functions > eventsphere-email-sender'. On the right, there are buttons for 'Throttle', 'Copy ARN', and 'Actions'. Below the navigation, the function name 'eventsphere-email-sender' is displayed with a 'Function overview' section and an 'Info' link. A 'Diagram' button is highlighted, showing a visual representation of the function's architecture. The diagram includes a central box labeled 'eventsphere-email-sender' with an orange icon, a 'Layers' section with '(0)', and an 'SNS' trigger represented by a pink icon. Buttons for '+ Add destination' and '+ Add trigger' are visible. To the right, there is a 'Description' section with a link to 'Last modified 2 days ago', a 'Function ARN' link (arn:aws:lambda:us-east-1:277707118728:function:eventsphere-email-sender), and a 'Function URL' link. At the bottom, tabs for 'Code', 'Test', 'Monitor', 'Configuration' (which is selected), 'Aliases', and 'Versions' are present. The 'Configuration' tab contains sections for 'General configuration', 'Triggers (1)', 'Permissions', 'Destinations', 'Function URL', 'Environment variables', and 'Tags'. The 'Triggers' section shows one trigger named 'SNS: eventsphere-notifications' with a detailed view of its configuration.

Shows the CloudWatch Lambda log stream, containing a log entry such as “*Email sent successfully. MessageId: ...*”, proving that the Lambda processor executed successfully and SES accepted the outgoing email.

Email received evidences

Welcome to EventSphere! Inbox x

 **noreply@enpm818rgroup7.work.gd**
to me ▾

7:18 PM (4 minutes ago) ☆ ⓘ ↗ ⋮

Hello s,

Welcome to EventSphere - Your Gateway to Amazing Events!

We're thrilled to have you join our community. EventSphere makes it easy to discover, book, and manage tickets for exciting events.

Here's what you can do:

- Browse upcoming events across various categories
- Book tickets instantly with our seamless booking system
- Manage your bookings and view your event history
- Get real-time updates about your events

Start exploring events now at <https://enpm818rgroup7.work.gd>

If you have any questions, feel free to reach out to our support team.

Happy event hunting!

Best regards,
The EventSphere Team

This is an automated message. Please do not reply to this email.

Booking Confirmed - MAGE Christmas Inbox x

 **noreply@enpm818rgroup7.work.gd**
to me ▾

Hello s,

Your booking has been confirmed!

Event Details:

- Event: MAGE Christmas
- Date: 12/25/2025
- Time: 10:00 AM
- Venue: JMP
- Category: festival

Booking Details:

- Booking ID: BKG1765239534560526
- Number of Tickets: 1
- Total Amount: \$0
- Booking Date: 12/9/2025, 12:18:54 AM

Important Information:

- Please arrive at least 15 minutes before the event starts
- Bring a valid ID for verification
- Your booking reference: BKG1765239534560526

View your booking details at: <https://enpm818rgroup7.work.gd/my-bookings>

We look forward to seeing you at the event!

Best regards,
The EventSphere Team

This is an automated message. Please do not reply to this email.

4.4 Amazon SES (Email Delivery)

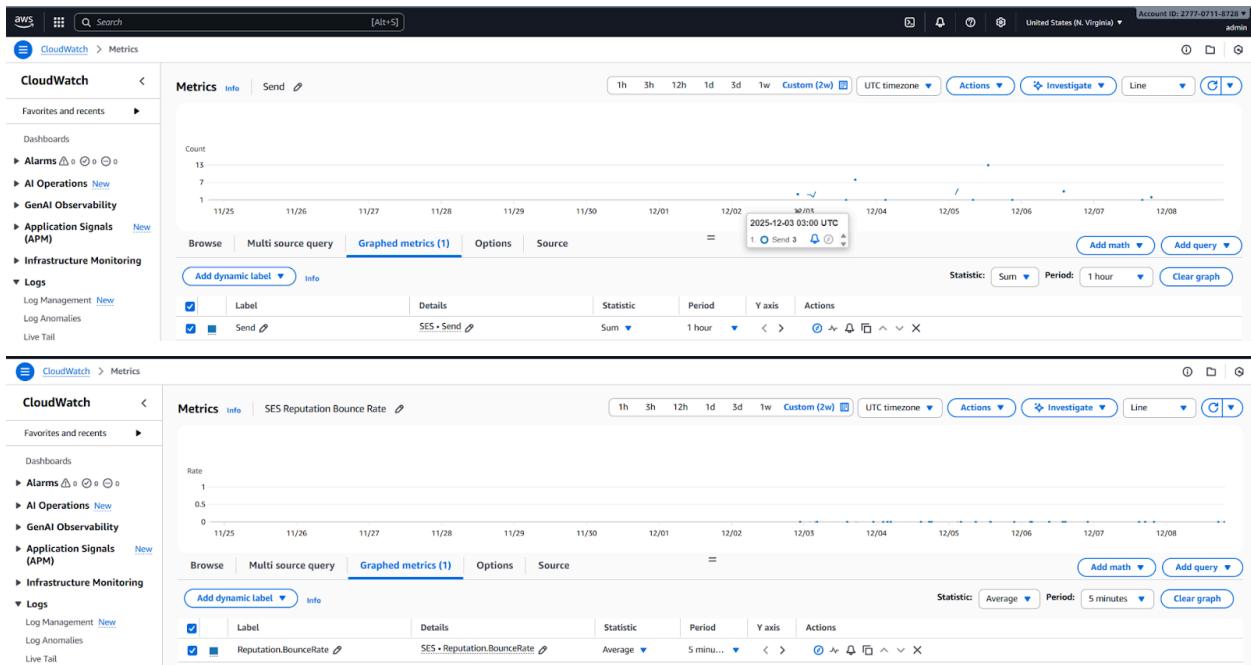
Displays the SES Sending Statistics dashboard, showing metrics such as total messages sent, delivery success rate, and bounce rates. This evidence confirms that SES is actively processing and delivering outgoing emails.

The screenshot shows the AWS Management Console for Amazon SES. The left sidebar navigation includes 'Amazon SES' (selected), 'Get set up', 'Account dashboard', 'Reputation metrics' (selected), 'SMTP settings', 'What's new', 'Configuration' (expanded), 'Identities' (selected), 'Configuration sets', 'Tenants' (New), 'Dedicated IPs', 'Global endpoints', 'Email templates', 'Suppression list', 'Email receiving', 'WorkMail' (Overview, Get set up), 'Virtual Deliverability Manager' (Advisor, Dashboard, Settings), and 'Mail Manager' (Overview, Get set up, Dashboard). The main content area is titled 'enpm818rgroup7.work.gd'. It displays a 'Summary' section with 'Identity status' (Verified) and 'Amazon Resource Name (ARN)' (arn:aws:ses:us-east-1:277707118728:identity/enpm818rgroup7.work.gd). The 'AWS Region' is 'US East (N. Virginia)'. Below this is a 'Recommendations' table with one entry: 'Impact: High, Age: 5 days, Recommendation: DMARC configuration was not found.' A 'Check for recommendations' button is available. The 'Authentication' tab is selected, showing 'DomainKeys Identified Mail (DKIM)' status (Successful) and 'DKIM configuration' (Enabled). The 'DomainKeys next signing length' is RSA_2048_BIT. The 'Last generated time' is December 3, 2025 at 04:03 (UTC-05:00). Other tabs include 'Notifications', 'Authorization', 'Configuration set', 'Tenants', and 'Tags'.

Shows the SES Reputation Metrics dashboard, confirming that SES is active and healthy.

The screenshot shows the AWS Management Console for Amazon SES. The left sidebar navigation is identical to the previous screenshot. The main content area is titled 'Reputation metrics'. It displays a 'Summary' section with 'Status' (Healthy), 'Emails sent (last 24 hours)' (4), 'Remaining sends' (49,996), and 'Sending quota used' (0.01%). Below this is a 'Monitoring historic bounce and complaint rates' section. The 'Bounce rate' chart shows a 'Historic bounce rate' of 0.00% with a 'Status' of 'Healthy'. The 'Complaint rate' chart shows a 'Historic complaint rate' of 0.00% with a 'Status' of 'Healthy'. Both charts have a legend: blue square for 'Historic [metric] rate', yellow dashed line for 'Warning', and red dashed line for 'Account at risk'. The 'Account-level' tab is selected, and the 'Configuration set' tab is available.

Metrics showing email sent and bounce rates in Cloudwatch.



5. Infrastructure & Deployment (Kubernetes + AWS)

This screenshot shows the **EKS worker nodes** successfully registered with the cluster. All nodes report a Ready status, confirming that the EKS node group is functioning correctly and able to run workloads.

```
Jerem@NOVO MINGW64 ~ (master)
$ aws eks update-kubeconfig --name eventsphere-cluster --region us-east-1
Added new context arn:aws:eks:us-east-1:27707118728:cluster/eventsphere-cluster to C:\Users\Jerem\.kube\config

Jerem@NOVO MINGW64 ~ (master)
$ kubectl get nodes -o wide
NAME           STATUS  ROLES   AGE    VERSION      INTERNAL-IP     EXTERNAL-IP   OS-IMAGE        KERNEL-VERSION   CONTAINER-RUNTIME
ip-10-0-31-26.ec2.internal  Ready   <none>  2d5h  v1.34.2-eks-ecaa3a6  10.0.31.26  44.203.197.239  Amazon Linux 2023.9.20251117  6.12.55-74.119.amzn2023.x86_64  containerd://2.1.4
ip-10-0-37-166.ec2.internal  Ready   <none>  28h   v1.34.2-eks-ecaa3a6  10.0.37.166  3.235.41.204   Amazon Linux 2023.9.20251117  6.12.55-74.119.amzn2023.x86_64  containerd://2.1.4
ip-10-0-43-2.ec2.internal   Ready   <none>  42h   v1.34.2-eks-ecaa3a6  10.0.43.2   44.192.19.69   Amazon Linux 2023.9.20251117  6.12.55-74.119.amzn2023.x86_64  containerd://2.1.4
ip-10-0-50-15.ec2.internal  Ready   <none>  28h   v1.34.2-eks-ecaa3a6  10.0.50.15  44.200.139.230  Amazon Linux 2023.9.20251117  6.12.55-74.119.amzn2023.x86_64  containerd://2.1.4
ip-10-0-6-229.ec2.internal  Ready   <none>  28h   v1.34.2-eks-ecaa3a6  10.0.6.229  54.164.143.16   Amazon Linux 2023.9.20251117  6.12.55-74.119.amzn2023.x86_64  containerd://2.1.4
```

This screenshot confirms that the Notification Service has **two running pods** in the prod namespace and displays the deployment configuration, including image version, replicas, environment variables, and health probes.

```
Jerem@NOVO MINGW64 ~ (master)
$ kubectl get pods -n prod -l app=notification-service -o wide
NAME          READY   STATUS    RESTARTS   AGE   IP           NODE      NOMINATED-NODE   READINESS   GATES
notification-service-69dd79f78c-29rns  1/1    Running   0          28h   10.0.41.84   ip-10-0-50-15.ec2.internal   <none>       <none>
notification-service-69dd79f78c-c169w  1/1    Running   0          28h   10.0.54.188  ip-10-0-37-166.ec2.internal  <none>       <none>

Jerem@NOVO MINGW64 ~ (master)
$ kubectl describe deployment notification-service -n prod
Name:           notification-service
Namespace:      prod
CreationTimestamp: Sat, 06 Dec 2025 14:22:29 -0500
Labels:         app=notification-service
                app.kubernetes.io/part-of=eventsphere
                tier-backend
Annotations:   deployment.kubernetes.io/revision: 1
Selector:       app=notification-service
Replicas:       2 desired | 2 updated | 2 total | 2 available | 0 unavailable
StrategyType:  RollingUpdate
MinReadySeconds: 0
RollingUpdateStrategy: 25% max unavailable, 25% max surge
Pod Template:
  Labels:      app=notification-service
                app.kubernetes.io/part-of=eventsphere
                tier-backend
  Service Account: notification-service-sa
  Containers:
    notification-service:
      Image:      27707118728.dkr.ecr.us-east-1.amazonaws.com/notification-service:latest
      Port:       4004/TCP (http)
      Host Port: 0/TCP (http)
      Limits:
        cpu: 200m
        memory: 256Mi
      Requests:
        cpu: 100m
        memory: 128Mi
      Liveness:   http-get http://:4004/health delay=30s timeout=5s period=10s #success=1 #failure=3
      Readiness:  http-get http://:4004/health delay=10s timeout=3s period=5s #success=1 #failure=3
      Environment:
        PORT:        4004
        NODE_ENV:    production
        AWS_REGION: <set to the key 'AWS_REGION' of config map 'app-config'>
        SNS_TOPIC_ARN: <set to the key 'SNS_TOPIC_ARN' of config map 'notification-service-config'> Optional: false
        AUTH_SERVICE_URL: http://auth-service:4001
      Mounts:
        /tmp from tmp (rw)
  Volumes:
    tmp:
      Type:      EmptyDir (a temporary directory that shares a pod's lifetime)
      Medium:
      SizeLimit: <unset>
  Node-Selectors: <none>
  Tolerations:  <none>
Conditions:
  Type        Status  Reason
  ----        ----   -----
  Progressing True   NewReplicaSetAvailable
  Available   True   MinimumReplicasAvailable
OldReplicaSets: <none>
NewReplicaSet:  notification-service-69dd79f78c (2/2 replicas created)
Events:        <none>
```

This screenshot shows the Notification Service's **IRSA-enabled ServiceAccount**, its ConfigMap with environment settings, and the Horizontal Pod Autoscaler, confirming proper AWS integration, externalized configuration, and autoscaling.

```

└── [root@MINIG64 ~ (master)] $ kubectl get serviceaccount notification-service-sa -n prod -o yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  annotations:
    eks.amazonaws.com/role-arn: arn:aws:iam::277707118728:role/eventsphere-notification-service-role
    kubernetes.io/taints: no-taint
  creationTimestamp: "2025-12-06T19:22:22Z"
  labels:
    app: notification-service
    environment: production
    name: notification-service-sa
    namespace: prod
    role: notification-service
    uid: df93c9f-2ab4-4008-8f10-bd8317fe3d7
  resourceVersion: "648"
  selfLink: /apis/serviceaccounts/v1/namespaces/prod/serviceaccounts/notification-service-sa
  uid: 5173630b-cadd-4068-b1e-1dfff7b5ec9
└── [root@MINIG64 ~ (master)] $ kubectl get configmap notification-service-config -n prod -o yaml
apiVersion: v1
data:
  AUTH_SERVICE_URL: http://auth-service.prod.svc.cluster.local:4001
  CORS_ORIGINS: http://localhost:3000,https://www.eventsp818group7.work.gd,https://emp818group7.work.gd
  NODE_ENV: production
  PORT: 4004
  SNS_TOPIC_ARN: arn:aws:sns:us-east-1:277707118728:eventsphere-notifications
kind: ConfigMap
metadata:
  annotations:
    kubernetes.io/last-updated-config: |-
      {"version": "v1", "data": {"AUTH_SERVICE_URL": "http://auth-service.prod.svc.cluster.local:4001", "CORS_ORIGINS": "http://localhost:3000,https://www.eventsp818group7.work.gd,https://emp818group7.work.gd", "NODE_ENV": "production", "PORT": "4004", "SNS_TOPIC_ARN": "arn:aws:sns:us-east-1:277707118728:eventsphere-notifications"}, "lastUpdated": "2025-12-06T19:22:20Z"}
  labels:
    name: notification-service-config
  name: notification-service-config
  namespace: prod
  resourceVersion: "648"
  uid: 5173630b-cadd-4068-b1e-1dfff7b5ec9
└── [root@MINIG64 ~ (master)] $ kubectl get hpa -n prod
NAME          REFERENCE          TARGETS          MINPODS   MAXPODS   REPLICAS   AGE
auth-service-hpa Deployment/auth-service   cpu: 7%/10%, memory: 37%/80%   2          10        2          2d5h
auth-service-hpa Deployment/auth-service   cpu: 7%/10%, memory: 37%/80%   2          10        2          2d5h
event-service-hpa Deployment/event-service  cpu: 7%/10%, memory: 37%/80%   2          10        2          2d5h
event-service-hpa Deployment/event-service  cpu: 2%/10%, memory: 37%/80%   2          5         2          2d5h
notification-service-hpa Deployment/notification-service  cpu: 0%/10%, memory: 25%/80%   2          10        2          2d5h
└── [root@MINIG64 ~ (master)] $ 

```

5.1 IAM Role

IAM Role trust relationship for eventsphere-notification-service-role. This policy shows that only the EKS OIDC provider and the Kubernetes ServiceAccount prod:notification-service-sa are allowed to assume this role via sts:AssumeRoleWithWebIdentity. This confirms that the Notification Service uses IRSA for secure, credential-free access to AWS services.

Entity	Action	Condition
oidc-provider/eks.us-east-1.amazonaws.com	sts:AssumeRoleWithWebIdentity	
sts.amazonaws.com	sts:AssumeRoleWithWebIdentity	Condition: system:serviceaccount:prod:notification-service-sa

```

arn:aws:iam::277707118728:role/eventsphere-notification-service-role
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": "oidc-provider/eks.us-east-1.amazonaws.com",
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "oidc-provider/eks.us-east-1.amazonaws.com/id": "7340932E960B1230204D7B53329E3F55"
        }
      }
    },
    {
      "Effect": "Allow",
      "Principal": "sts.amazonaws.com",
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "sts.amazonaws.com/id": "7340932E960B1230204D7B53329E3F55;sub": "system:serviceaccount:prod:notification-service-sa"
        }
      }
    }
  ]
}

```

IAM policy eventsphere-notification-sns-policy attached to the Notification Service role. The policy grants the minimum required permission—sns:Publish—allowing the service to publish notification messages to the SNS topic eventsphere-notifications.

The screenshot shows the AWS IAM Policies page. The left sidebar shows navigation options like Identity and Access Management (IAM), Dashboard, Access management, and Policies. The main content area displays the details of the 'eventsphere-notification-sns-policy'. It includes a 'Policy details' section with creation and edited times, and an 'ARN' field. Below this is a 'Permissions' tab showing the JSON code for the policy:

```

1  [ "Version": "2012-10-17",
2   "Statement": [
3     {
4       "Effect": "Allow",
5       "Action": "sns:Publish"
6     },
7     {
8       "Resource": "arn:aws:sns:us-east-1:277707118728:eventsphere-notifications"
9     }
10    ]
11  ]
12 ]

```

6.CI/CD Pipeline (Notification Service)

The Notification Service is integrated into the EventSphere GitHub Actions **CI/CD pipeline**. Any change in the services/notification-service/ directory triggers the CI workflow, which runs security scans, validates Kubernetes manifests, and builds the container image. After changes are merged into main, the CD pipeline builds and pushes the final Docker image to ECR and deploys it to EKS using Helm with atomic rollouts and automated health checks.

Below screenshot shows the full set of **CI/CD workflows running** in the EventSphere repository, including the Notification Service integration pipeline, the “Build and Push Docker Images” job, and the “Deploy to Production (EKS)” workflow. The presence of successful and failed runs demonstrates that automated pipelines enforce quality control and ensure reliable deployments to EKS. too long

The screenshot shows the GitHub Actions pipeline for the EventSphere repository. The left sidebar lists actions like CD, CI, Management, Caches, Deployments, Attestations, Usage metrics, and Performance metrics. The main area shows the 'All workflows' page with 147 workflow runs. The runs are listed in a table with columns for Event, Status, Branch, Actor, and Duration. The runs include various steps such as 'Merge pull request #12 from Darpankarur/main', 'Email Notification System Integration - SNS + SES + Lambda', 'Deploy to Production (EKS)', 'Deploy to Staging (kind)', 'Build and Push Docker Images', 'Merge pull request #11 from sakethbolla/kugadgil/autoscaler-and-eks...', and 'fix: cluster autoscaler autodiscovery, eks control plane logging and some run...'. Most runs are successful, indicated by green icons.

This screenshot displays the **complete GitHub Actions CD pipeline** for the EventSphere project. The workflow includes environment selection, image build and push to ECR, Helm deployment to EKS, and post-deployment smoke tests. All steps succeeded, confirming reliable automated deployment of the Notification Service to the cluster.

containerization-team-cems-eks / EventSphere

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

← CD Fix Frontend.js Test and CI Pipeline (#22) #11

Re-run all jobs ...

Summary

Triggered via push 1 hour ago Status Success Total duration 6m 29s Artifacts

baeful pushed → 8c83035 main

Jobs

- Determine Environment
- Build and Push Images
- Deploy to EKS
- Post-Deployment Smoke Tests

cd.yml on: push

```
graph LR; A[Determine Environment] --> B[Build and Push Images]; B --> C[Deploy to EKS]; C --> D[Post-Deployment Smoke Tests]
```

Run details Usage Workflow file

The screenshot shows a GitHub Actions pipeline summary for a pull request. The pipeline is named "Fix Frontend.js Test and CI Pipeline" and has been triggered via a push 1 hour ago. The status is "Success" with a total duration of 6m 29s. The pipeline consists of four jobs: "Determine Environment", "Build and Push Images", "Deploy to EKS", and "Post-Deployment Smoke Tests". All jobs have passed. The pipeline is defined in a cd.yml file, which triggers on pushes. The workflow file is available for download.

Reflection

The successful deployment of EventSphere was a huge learning process for the team. We learnt how to build a scalable, micro-service architecture platform, with monitoring, CI/CD pipeline, and version control system. We learnt how to use Kubernetes in the cloud with AWS EKS, and how to build and use it securely. Our CI/CD pipeline allowed us to update our production servers/pods live, with no downtime, which gave us insights into how real world applications are built and maintained. We deployed an auto-scaling cluster that scaled up and down based on traffic and demand, which meant low costs during low traffic, and high availability during peak traffic times. We learnt how to build and integrate microservices together, and how we could use pods that scale automatically based on each specific microservice's demands. With SNS and SES, we developed a robust system that could both send emails and notifications to users regarding events, and developers regarding operational triggers, reducing response time to errors/problems that occur during live production.

Overall, this was a very challenging, yet rewarding project, that taught each one of us how to deploy and maintain a microservice, autoscaling application in the cloud.