National College of Ireland

Configuration Manual

MSc Industrial Internship
Cyber Security

Rohit Anand Ahuja
Student ID: x21168296

School of Computing
National College of Ireland

Supervisor: Vikas Sahni

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Rohit Anand Ahuja
Student ID:	21168296
Programme:	MSc in Cybersecurity **Year:** 2022-2023
Module:	MSc Industrial Internship
Lecturer:	Vikas Sahni
Submission Due Date:	06/01/2023
Project Title:	Securing the End Points of Microservices using GitLab Client-Based Authentication.
Word Count:	2136 **Page Count:** 28

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	ROHIT ANAND AHUJA
Date:	06/01/2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

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

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

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

Configuration Manual

Rohit Anand Ahuja
Student ID: 21168296

1 Deploying a Kubernetes Cluster on AWS EKS (Elastic Kubernetes Service)

Here are the steps to deploy a Kubernetes cluster on AWS EKS with screenshots:

1. Sign in to the AWS Management Console and open the Amazon EKS console at https://console.aws.amazon.com/eks/ and click on create cluster button.

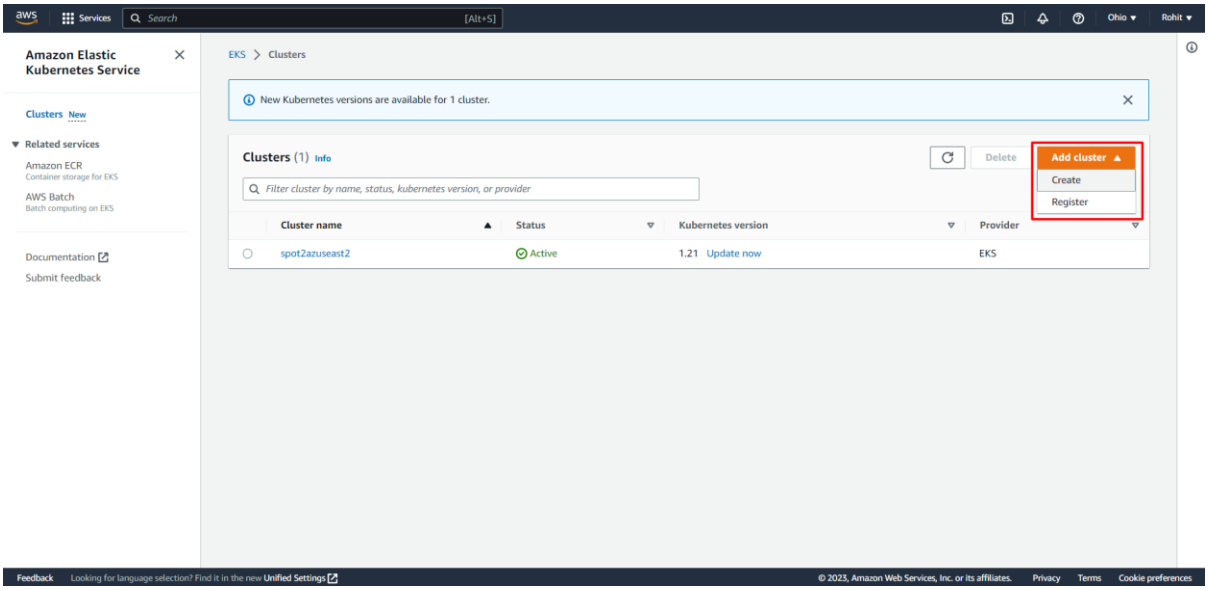

Figure 1: AWS EKS Create Cluster Page

2. On the "Create Cluster" page, enter a name for the cluster and select the desired service role. Then, choose a Kubernetes version and click the "Next" button. Kubernetes v1.21 has been selected for the proposed research.

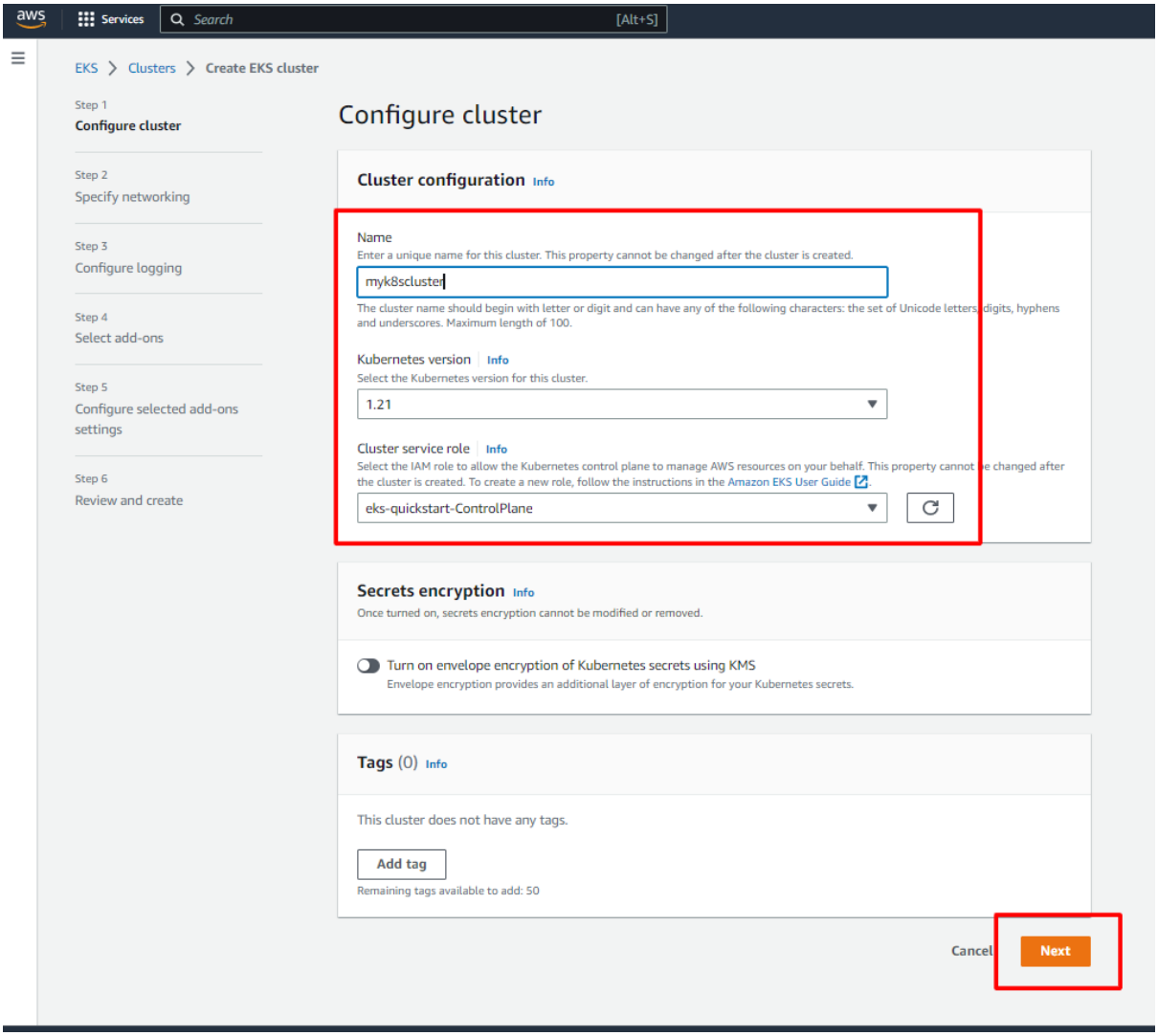

Figure 2: Configure Cluster Page

3. On the "Configure Cluster" page, choose the VPC, subnets, and security group for the cluster. Configure the Kubernetes API server endpoint access as Private. Click the "Next" button.

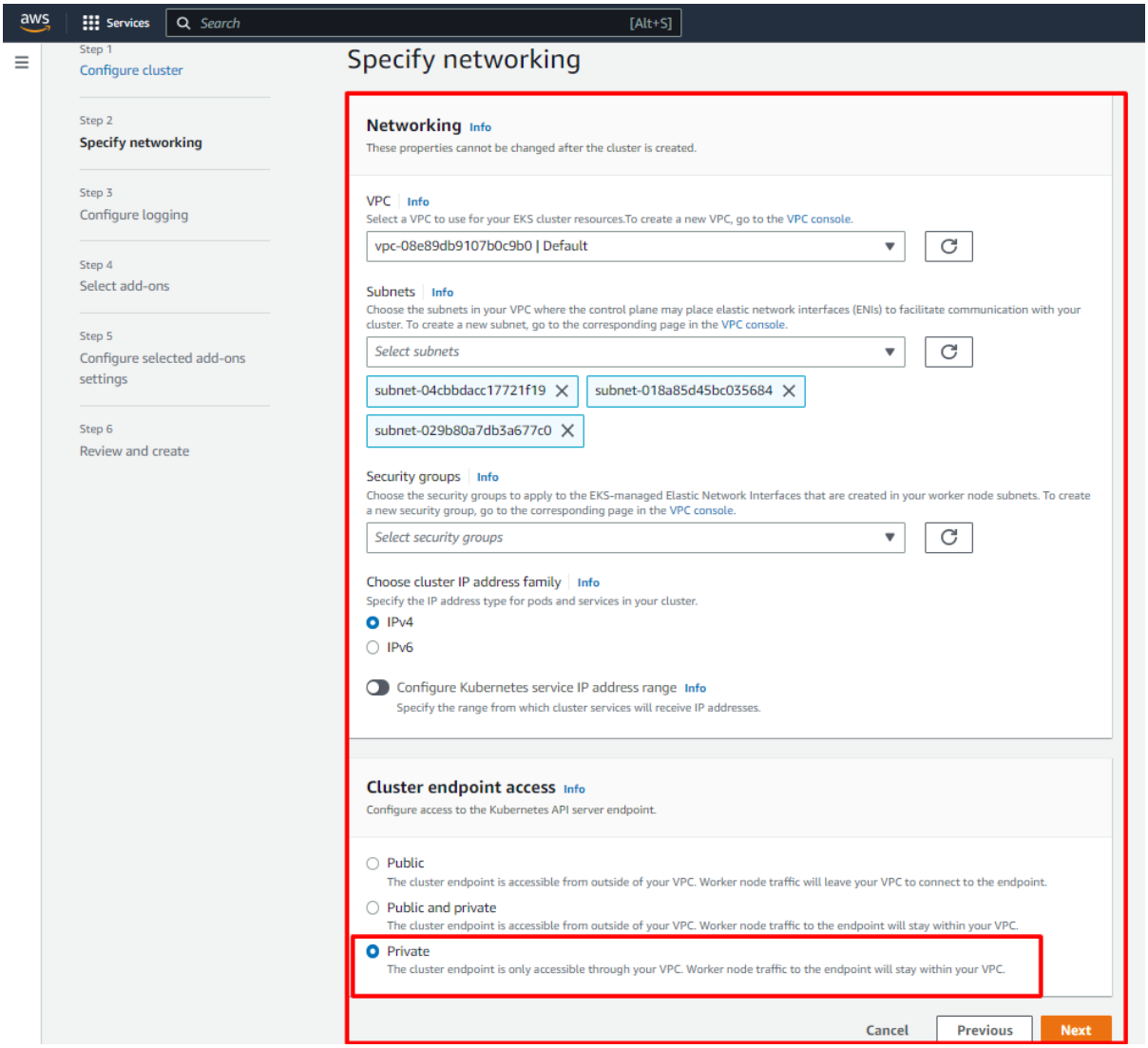

Figure 3: Specify Networking Page

4. Skip Step 3,4,5 of AWS EKS Cluster Creation and at the last step 6 i.e., Review and Create – Click on Create.

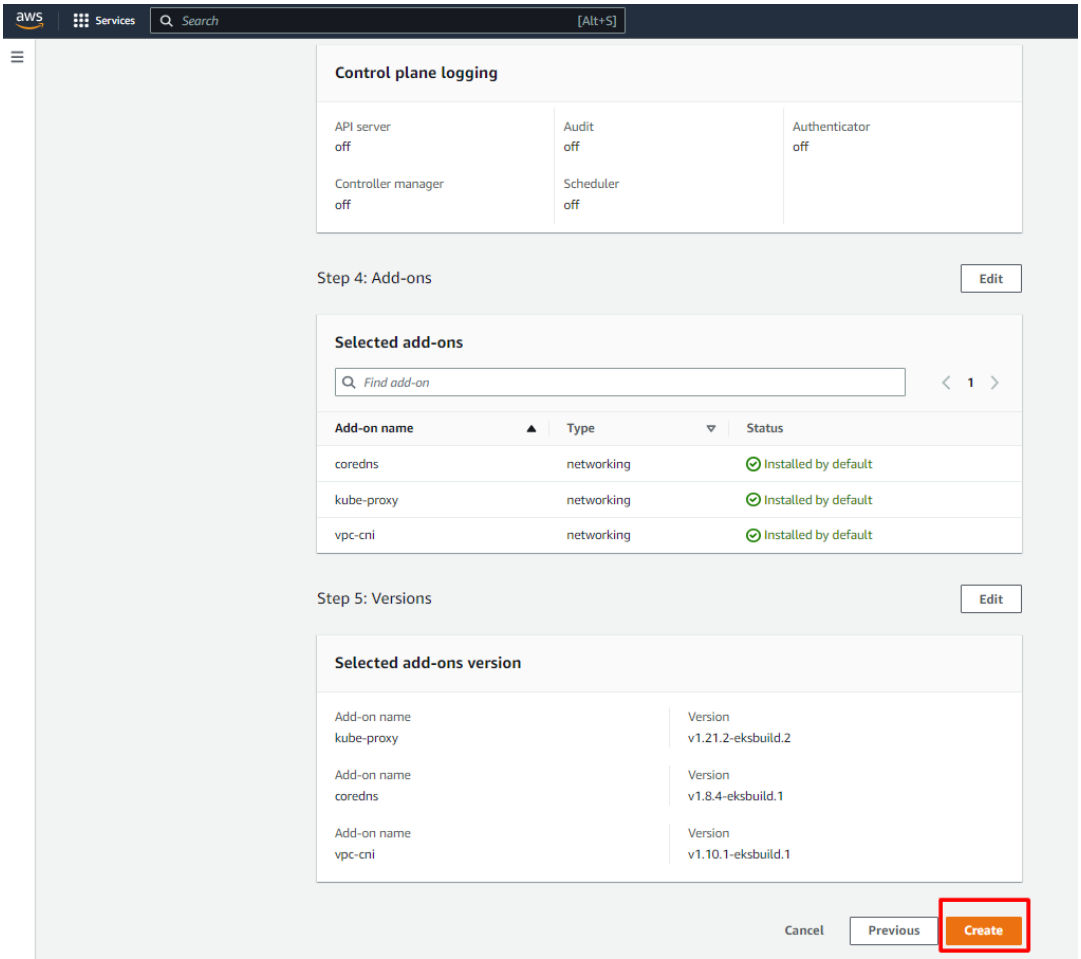

Figure 4: Create and Review Cluster Page

5. This step deploys the Kubernetes cluster on AWS EKS; cluster will be created in 10-15 mins along with one EKS Bastion Host for managing Kubernetes cluster.
6. Once the cluster is created, go to the compute tab of the created cluster and click on add node group to add worker nodes.

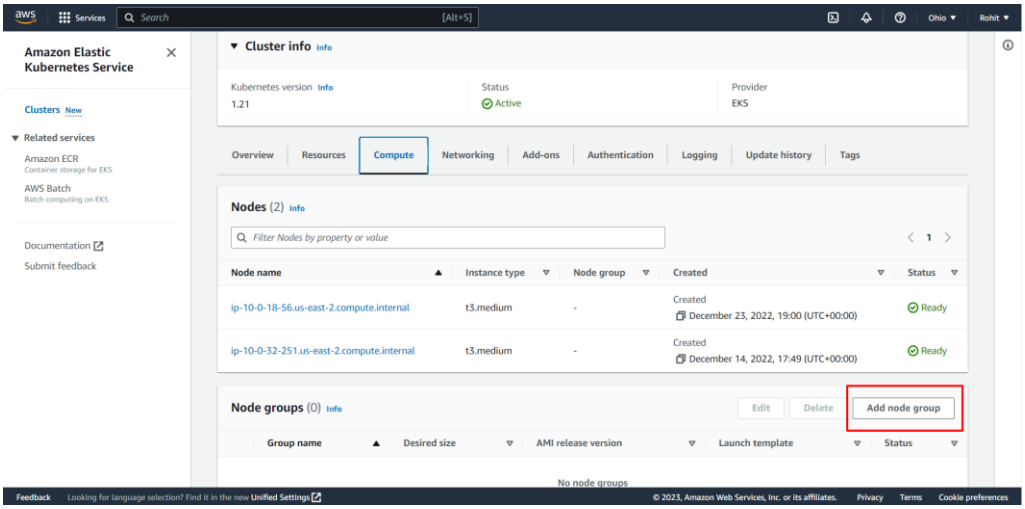

Figure 5: Compute tab of Cluster

7. On the configure node page, write the node name and specify the IAM role and click on next.

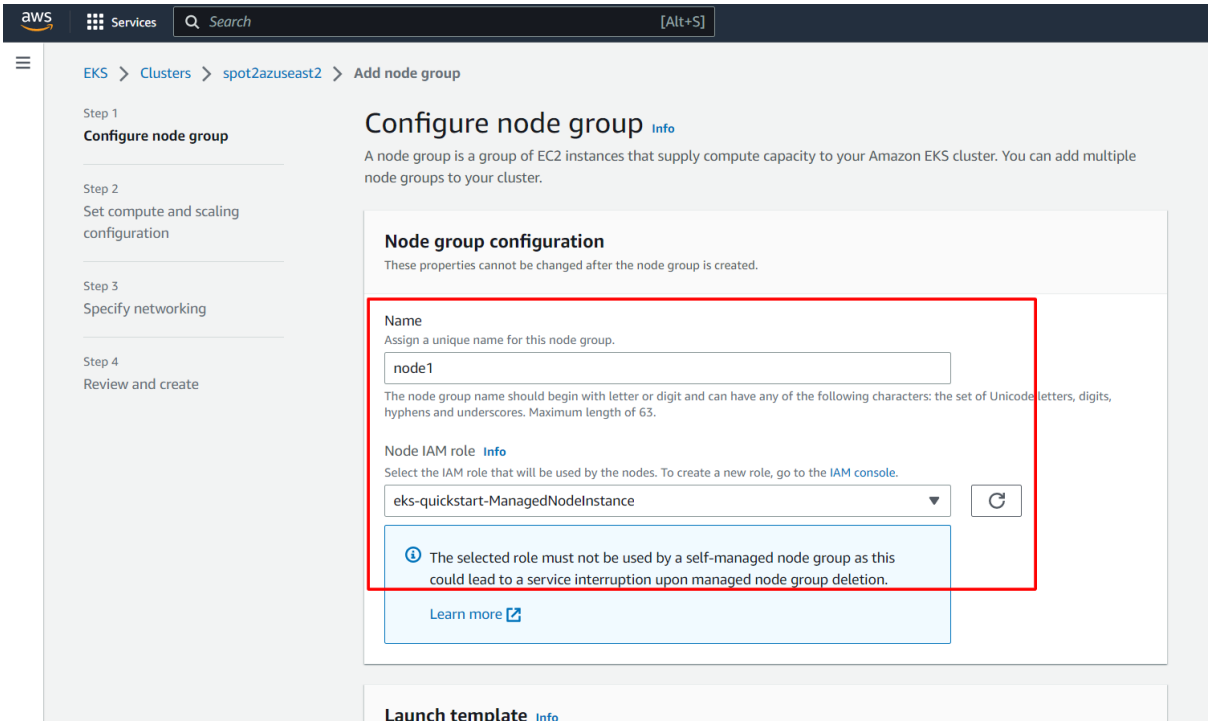

Figure 6:Configure Node Group Page

8. On the "Add Worker Nodes" page, specify the Amazon EC2 instances that will be used as worker nodes for the cluster. Choose the instance type, auto scaling group configuration, and other options. When done, click the "Next" button.

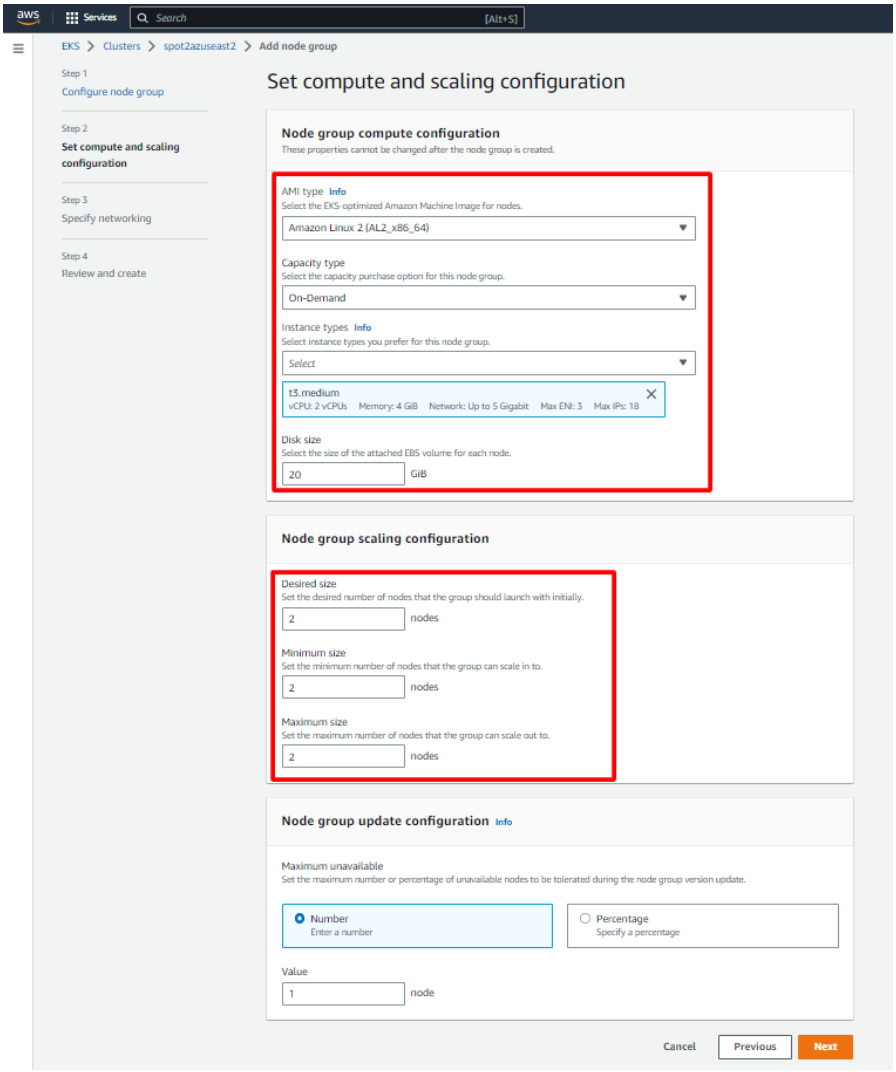

Figure 7: Node Configuration Page

9. Click on create on the last page of the node group configuration.

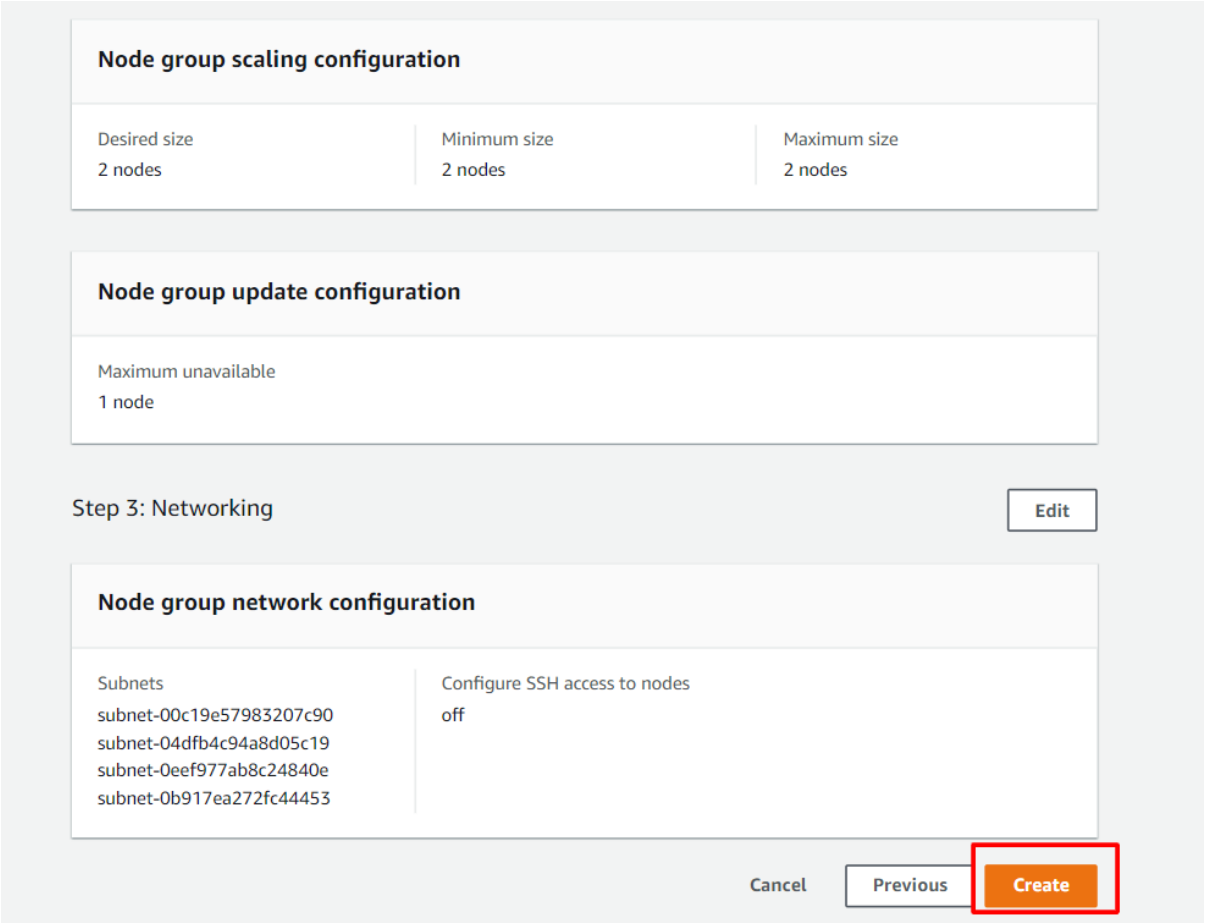

Figure 8: Create Node Group

10. Once the cluster is ready, the cluster can be viewed under EKS Bastion Host, by firing the below command as shown in the screenshots:

```
sh-4.2$ kubectl get svc
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   172.20.0.1   <none>        443/TCP   19d
sh-4.2$ kubectl get nodes -A
NAME                                          STATUS   ROLES    AGE   VERSION
ip-10-0-18-56.us-east-2.compute.internal      Ready    <none>   10d   v1.21.14-eks-fb459a0
ip-10-0-32-251.us-east-2.compute.internal     Ready    <none>   19d   v1.21.14-eks-fb459a0
sh-4.2$
```

Figure 9: Kubectl

2 Create Cluster Management Project on Gitlab

Follow these steps:

1. Create a group in GitLab.
2. Click on the "New project" or "New repository" button.
3. Click on the "Import" button.
4. Select the "From a template" option.
5. Select the "GitLab Cluster Management" template and click "Import".
6. Enter "Cluster Management" as the project name.

7. Click the "Create project" button.

After completing these steps, a new project in GitLab that is based on the GitLab Cluster Management template will be created. Use this project to manage to deploy GitLab agent and applications to a Kubernetes cluster.

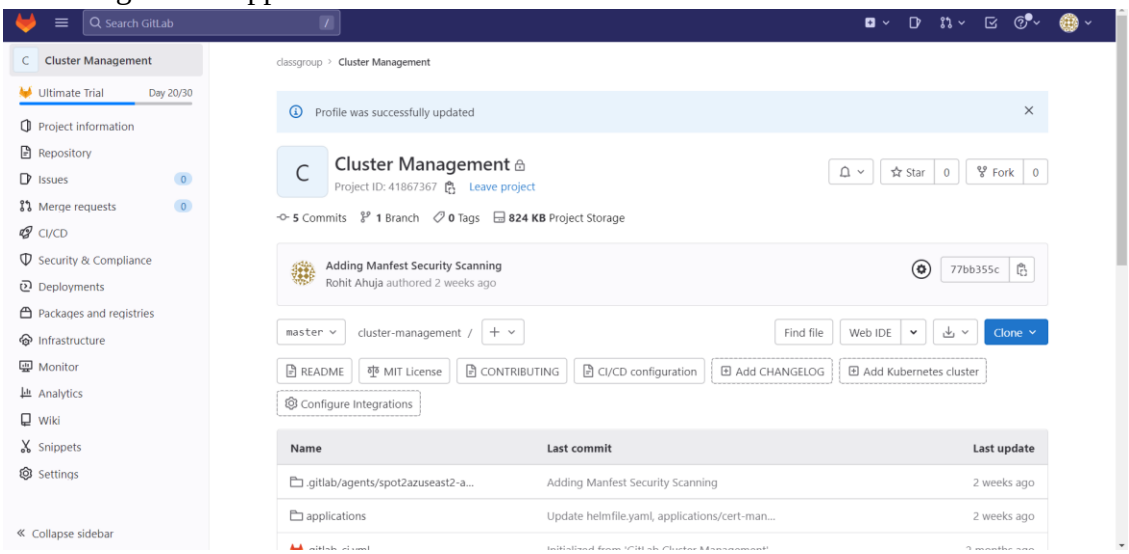

Figure 10: Cluster Management Gitlab Project

3 Register the Gitlab Agent for Kubernetes.

Connect to the EKS Bastion Host, which was deployed earlier, and fire the below commands to register the GitLab agent.

```
helm repo add gitlab https://charts.gitlab.io
helm repo update
helm upgrade --install spot2azuseast2-agent1 gitlab/gitlab-agent \
    --namespace gitlab-agent-spot2azuseast2-agent1 \
    --create-namespace \
    --set image.tag=v15.7.0 \
    --set config.token=6ym1kUJYtiAyWnkNWmkoZXFudKMymsm3DCN1SRziuNfmAPxBkg \
    --set config.kasAddress=wss://kas.gitlab.com
```

Figure 11:Gitlab Agent Setup

Config token can be acquired by following the steps in:
https://docs.gitlab.com/ee/user/clusters/agent/install/

Once the agent has been installed, the agent will reflect in GitLab Kubernetes cluster and also on AWS EKS under Cluster Management Project as shown below:

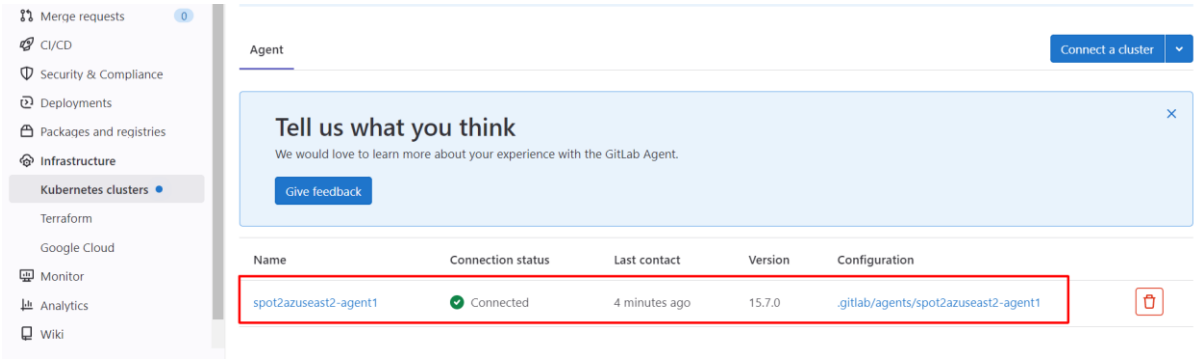

Figure 12:Gitlab Agent in Gitlab Kubernetes Cluster

```
sh-4.2$ kubectl get pods -A | grep "spot2azuseast2-agent1"
gitlab-agent-spot2azuseast2-agent1            scan-vulnerabilityreport-54d958d484-kxjrv                  0/1     Completed   0          10d
gitlab-agent-spot2azuseast2-agent1            scan-vulnerabilityreport-55cbd67ff7-j4prp                  0/1     Completed   0          10d
gitlab-agent-spot2azuseast2-agent1            scan-vulnerabilityreport-6458d55fb8-8kfg2                  0/1     Completed   0          10d
gitlab-agent-spot2azuseast2-agent1            scan-vulnerabilityreport-6677f5d548-qdskl                  0/1     Completed   0          10d
gitlab-agent-spot2azuseast2-agent1            scan-vulnerabilityreport-674b77f559-fxlnk                  0/1     Completed   0          10d
gitlab-agent-spot2azuseast2-agent1            scan-vulnerabilityreport-786b77889f-2rpbx                  0/1     Completed   0          14d
gitlab-agent-spot2azuseast2-agent1            scan-vulnerabilityreport-797b76d9cb-6rkcf                  0/1     Completed   0          10d
gitlab-agent-spot2azuseast2-agent1            scan-vulnerabilityreport-7b4bd8c44-9cffj                   0/1     Completed   0          10d
gitlab-agent-spot2azuseast2-agent1            scan-vulnerabilityreport-7fc557c567-4ndd8                  0/1     Completed   0          10d
gitlab-agent-spot2azuseast2-agent1            scan-vulnerabilityreport-845cf48f7d-c9n15                  0/1     Completed   0          14d
gitlab-agent-spot2azuseast2-agent1            spot2azuseast2-agent1-gitlab-agent-bb86cff8c-lbkl7         1/1     Running     0          10d
sh-4.2$
```

Figure 13: Gitlab Agent deployed in Kubernetes Cluster

4 Deploy Isolated Application Build and Application Environment Projects.

Application Build and Application Environment can be deployed using the below steps:

1. Log in to GitLab account.
2. Click the "Import project" button in the top-right corner of the dashboard.
3. Enter the URL of the Git repository that want to import, and click the "Create project" button.
4. GitLab will then import the project and create a new repository for it in the registered account.
5. Once the import is complete, GitLab will redirect to the project page for the newly-imported repository.

Application Build URL: https://gitlab.com/classgroup6/rohit_ahuja/hello-world.git

Application Environment URL: https://gitlab.com/classgroup6/rohit_ahuja/world-greetings-env-1.git

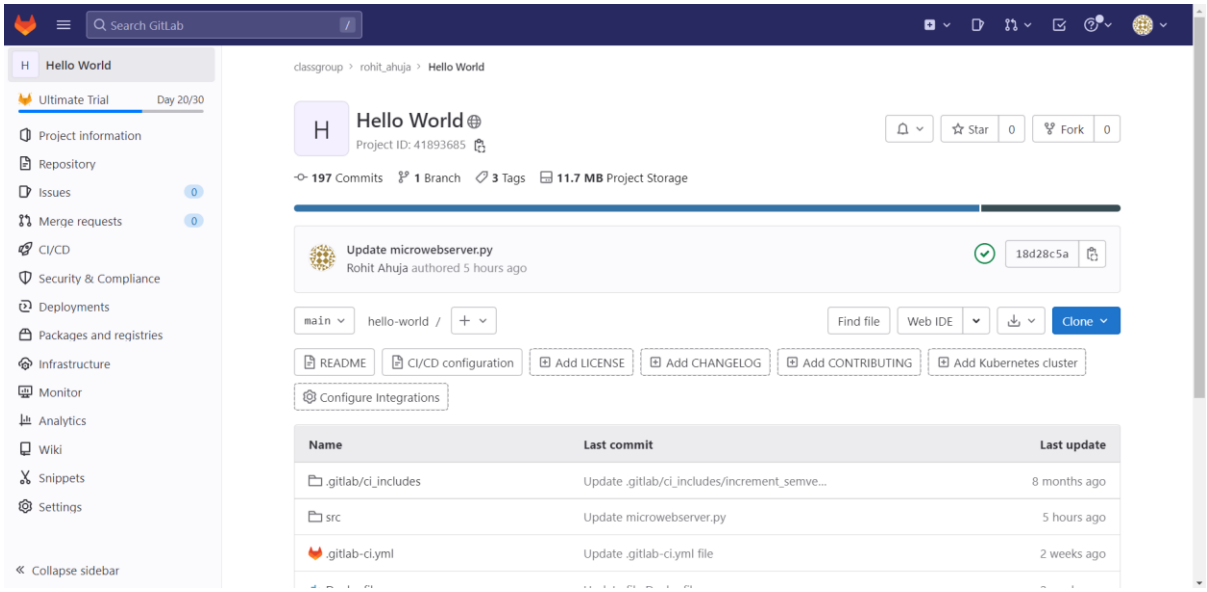

Figure 14: Application Build

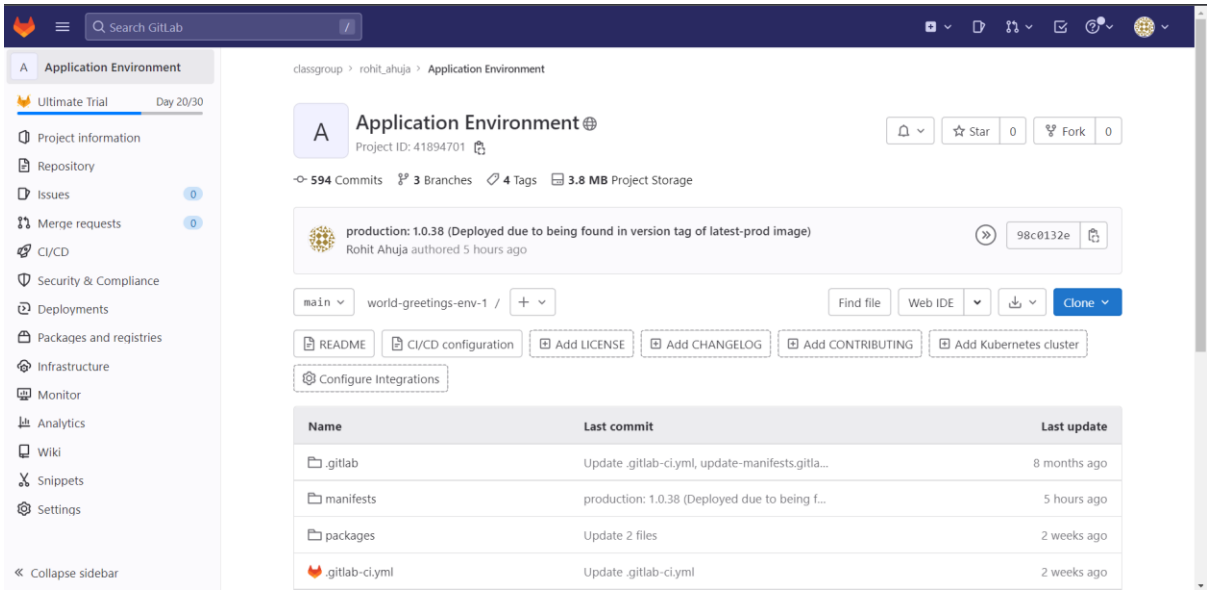

Figure 15: Application Environment

5 Running the Application Build and Saving the Container Image

To create a deploy token for a container registry in GitLab:

1. Start a pipeline run for the project that i.e., Application Build. This should create a container image.

2. After the pipeline has completed, go to the repository settings for the project.

3. In the "Repository" section, click on "Deploy Tokens".
4. Click the "Create Deploy Token" button.
5. Give the token a name, such as "ReadRegistry".
6. Under "Scopes", select "read_registry".
7. Click the "Create Deploy Token" button to create the token.

This deploy token can be used to access the container registry for this project.

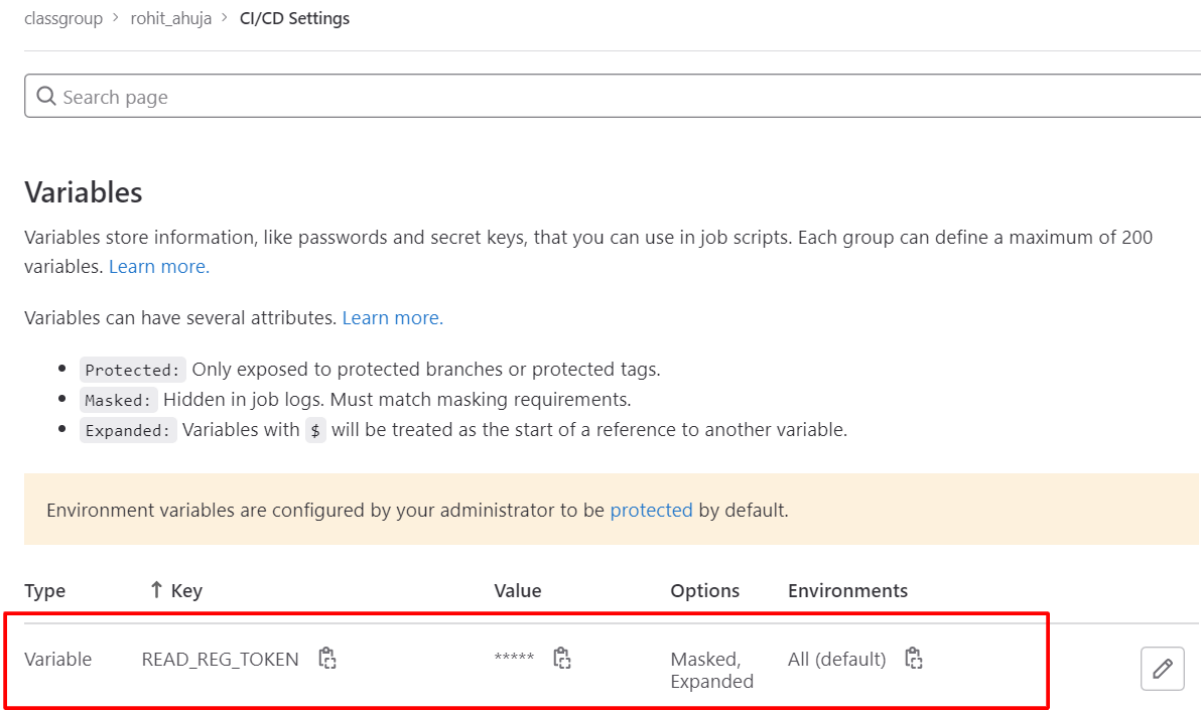

Figure 16: Read Container Registry Image Token

6 Setting up Application Environment Manifests to Monitor Container Image

Steps:

1. Edit the manifests of production and staging under the application environment by going under the application environment manifests

2. Add the registry link of the application build project under the image tag.

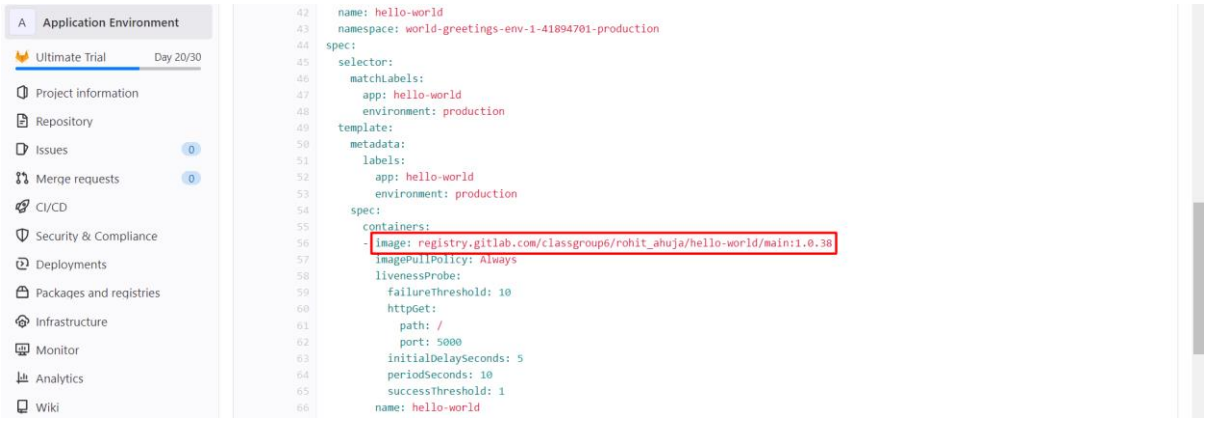

Figure 17: Application Environment Manifests - YAML

7 Setting up agent to monitor the changes in Application Environment.

If the agent detects a change to the container image version, it will automatically deploy the updated version to both the staging and production environments. This helps to ensure that the application is always running the most up-to-date version, which can help to improve performance and stability. The agent makes it easy to manage and maintain the application, as it handles the deployment process automatically, freeing up time and resources for other tasks.

In order to instruct the Kubernetes Gitlab Agent to monitor the application environment, will need to edit the config.yaml file. This file is located in the cluster management repository, and can be edited using a text editor or other suitable software. The process of editing the config.yaml file is shown in the accompanying screenshots. To edit the file, simply open using online IDE and make changes. Once the changes have been done, save the file and the agent will begin monitoring the application environment according to the updated configuration. It is important to carefully review and understand the contents of the config.yaml file before making any changes, as incorrect configurations can impact the operation of the agent and the overall performance of the application.

master cluster-management / .gitlab / agents / spot2azuseast2-agent1 / config.yaml

config.yaml 816 bytes

```yaml
#id: = Full group path without instance url
# and without leading or trailing slashes.
# for https://gitlab.com/this/is/an/example, id would be:
# - id: this/is/an/example

ci_access:
  groups:
  - id: classgroup6

observability:
  logging:
    level: debug

starboard:
  cadence: '45 * * * *' #Every hour at 55 minutes past the hour

gitops:
  manifest_projects:
  - id: classgroup6/rohit_ahuja/world-greetings-env-1
    default_namespace: default
    paths:
    - glob: '/manifests/**/*.yaml'
    reconcile_timeout: 3600s # 1 hour by default
    dry_run_strategy: none # 'none' by default
    prune: true # enabled by default
    prune_timeout: 360s # 1 hour by default
    prune_propagation_policy: foreground # 'foreground' by default
    inventory_policy: must_match # 'must_match' by default
```

Figure 18: Gitlab Agent Configuration

8 Deploying the Application using Pull-Based Approach by Gitlab Agent.

After making any changes to the application build, the application build pipeline will create a registry image and save it in GitLab registry.

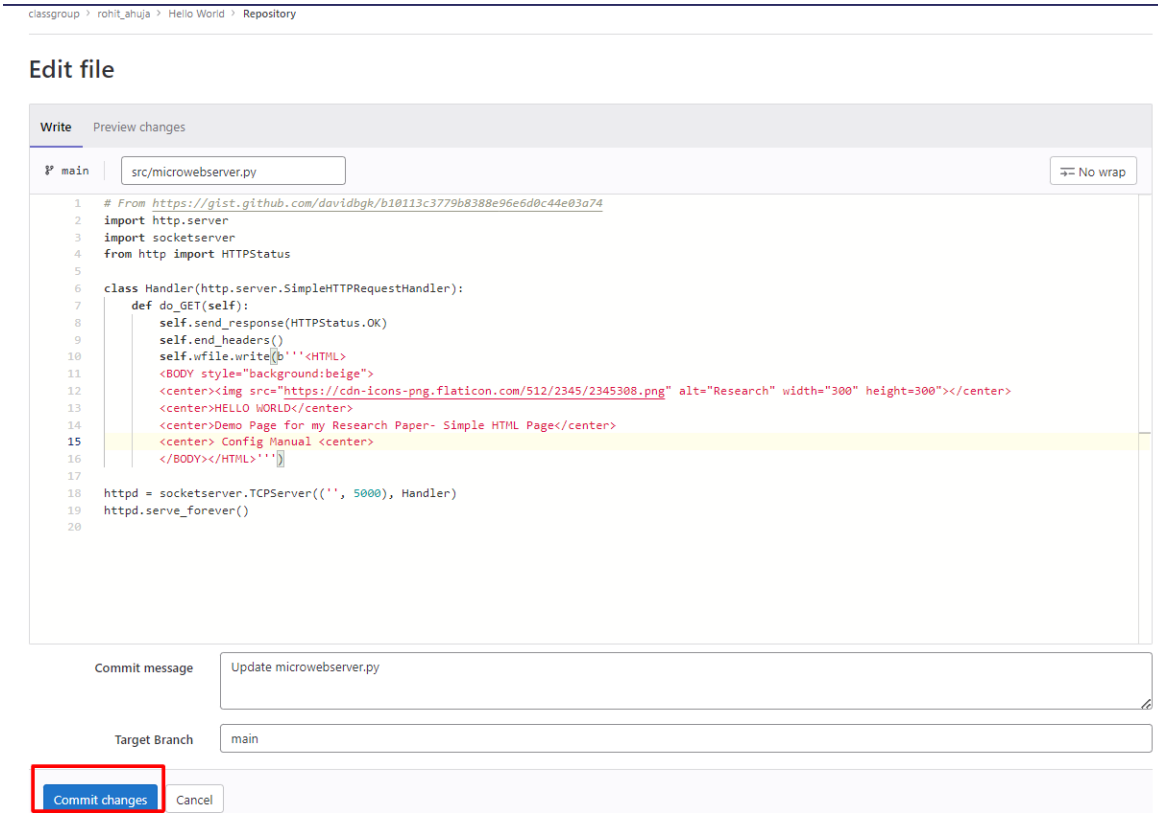

Figure 19:Committing change to Application Build

Figure 20: Running the Application Build Pipeline

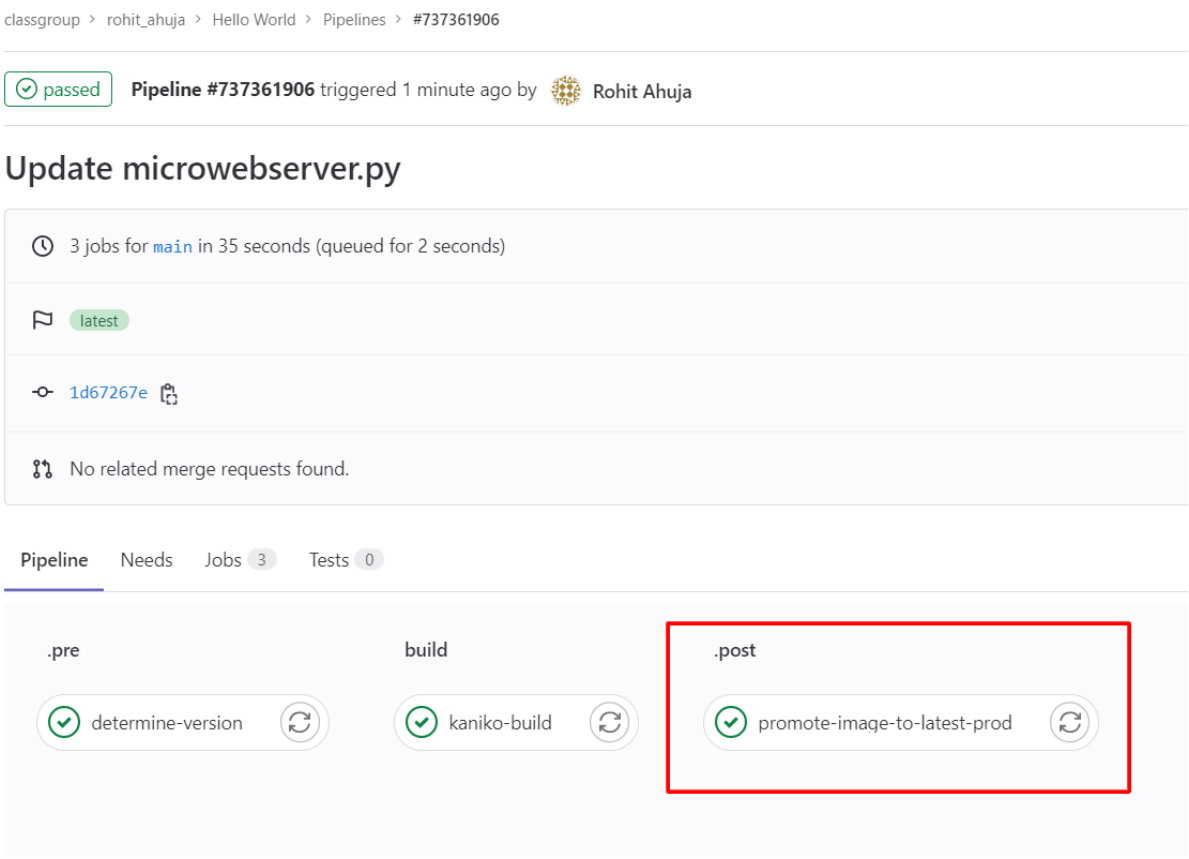

Figure 21: Updating the latest container image in registry

After the image is stored in the registry, the application environment manifests file will be updated and application will be deployed in the staging and production environment after running the application environment pipeline.

Figure 22:Running the Application Environment Pipeline

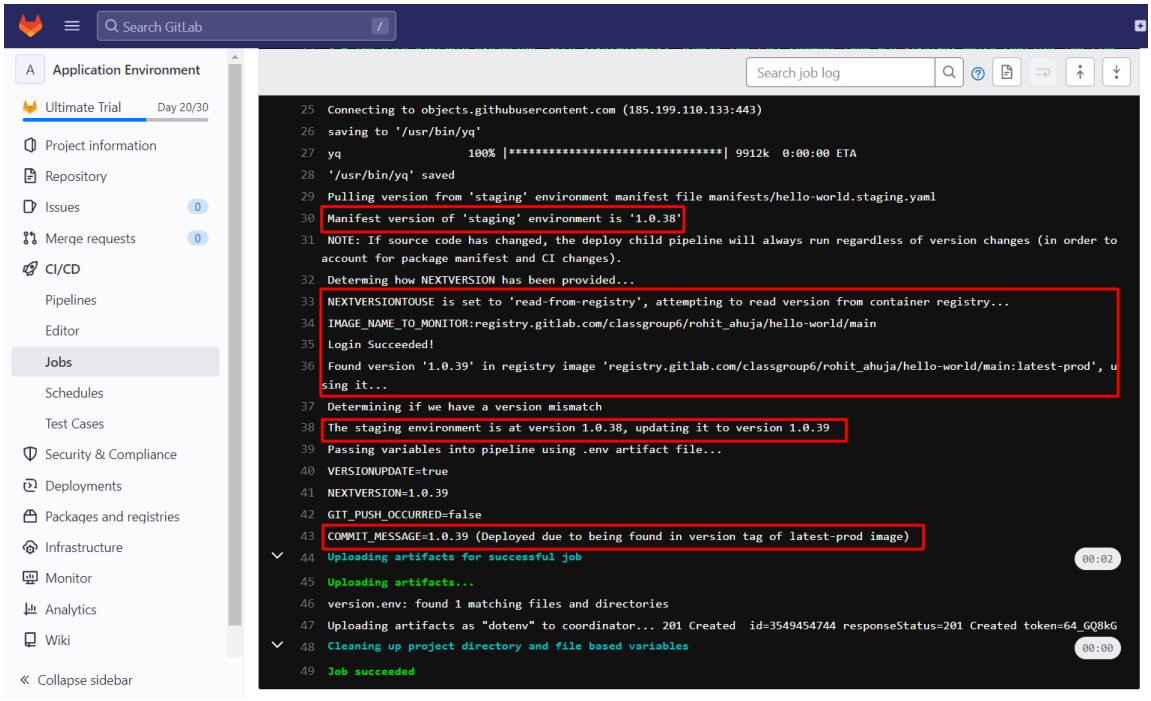

Figure 23: Reading image from the registry

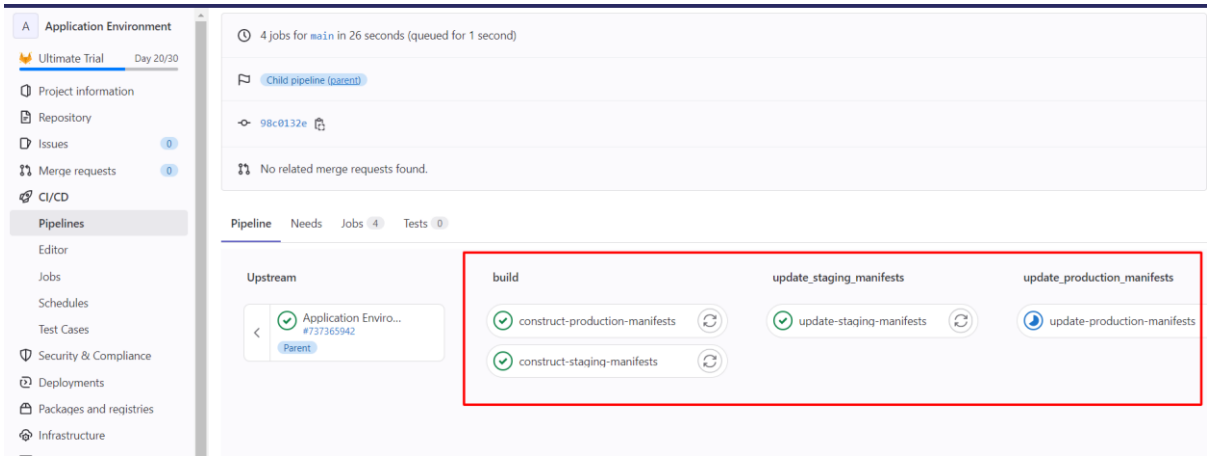

Figure 24: Updating the Staging and Production Manifests

The gitlab agent automatically pulls the changes in the application enivronment manifests and deploy the application to the staging and production environment.

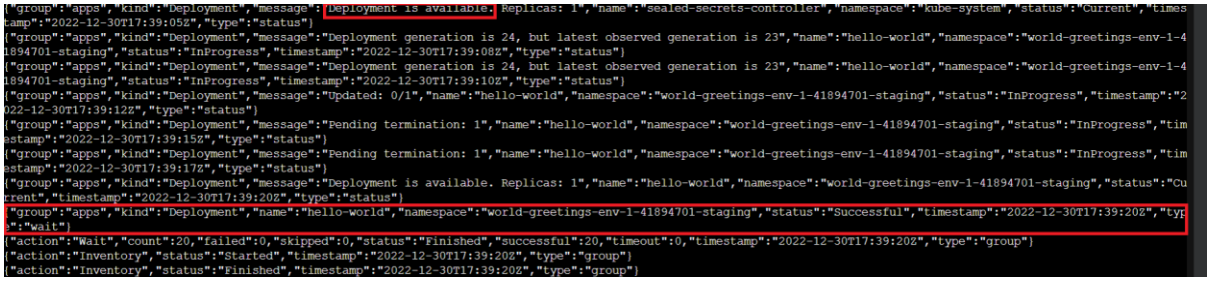

Figure 25:Gitlab Agent pulling changes into the cluster and deploying the application

Staging Environment:

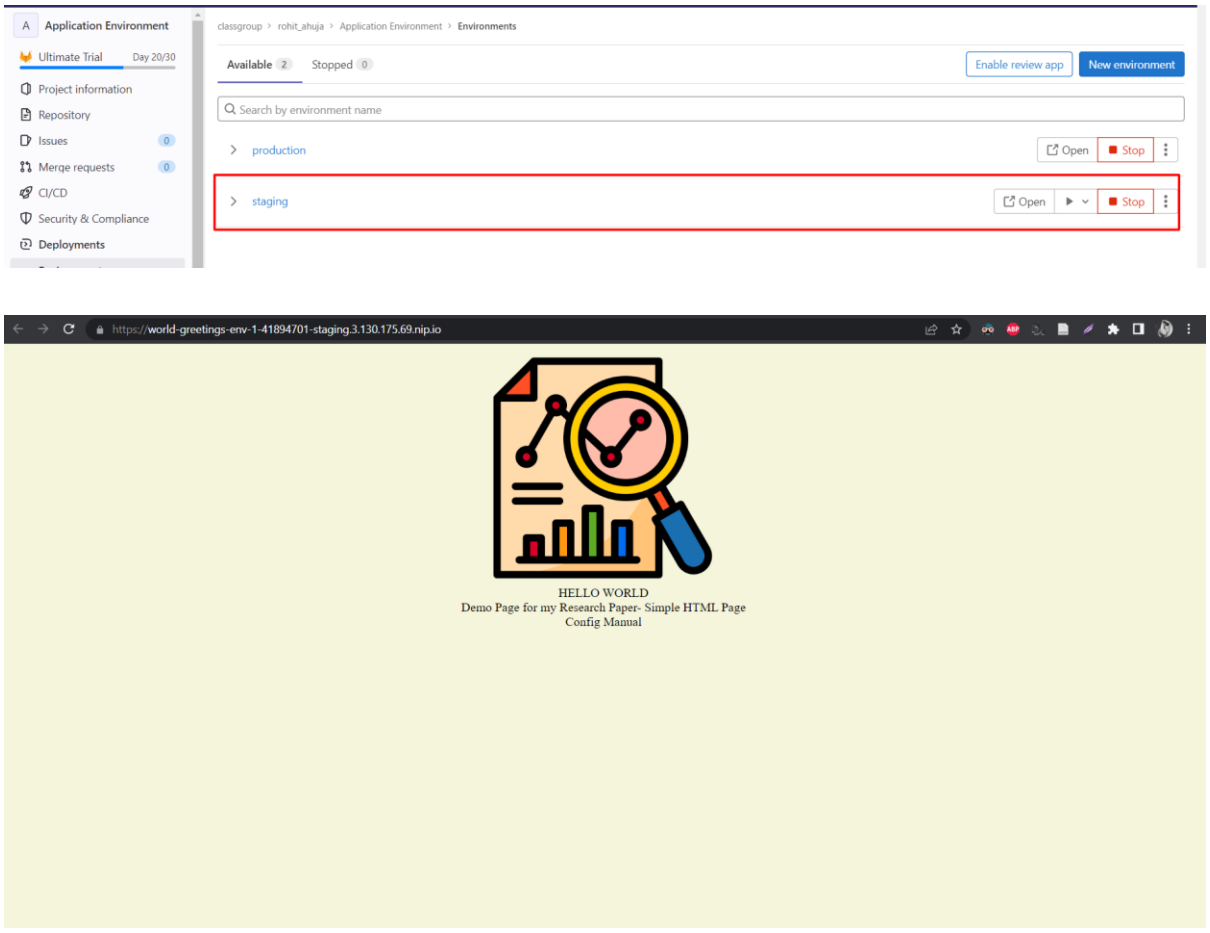

Figure 26: Application Deployed to Staging

Production Environment:

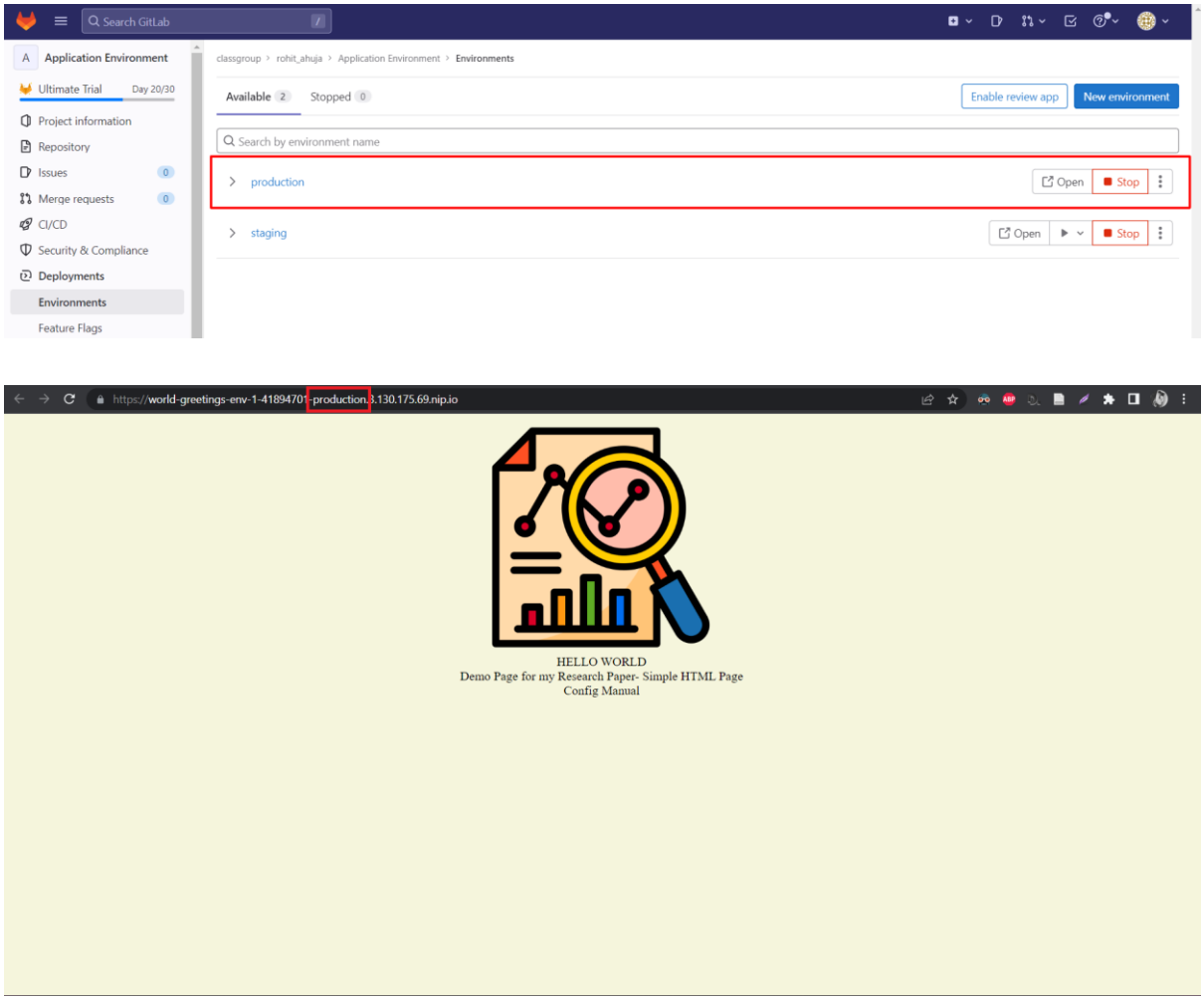

Figure 27: Application Deployed to Production

9 Adding Security Scanning, Kubernetes Manifests Scanning and Container Scanning.

To add security scanning to application by editing the pipeline in Gitlab, the built-in static application security testing (SAST) feature is used. Here are the steps to do this:

1. In the Gitlab web interface, navigate to the project containing Application Build
2. In the project's "Settings" menu, select "Pipelines" and then click the "Expand" button next to the "Jobs" heading.
3. Click the "Edit" button next to the job that builds and test the application. This will open the job's configuration file in the Gitlab code editor.
4. To enable SAST add the following:

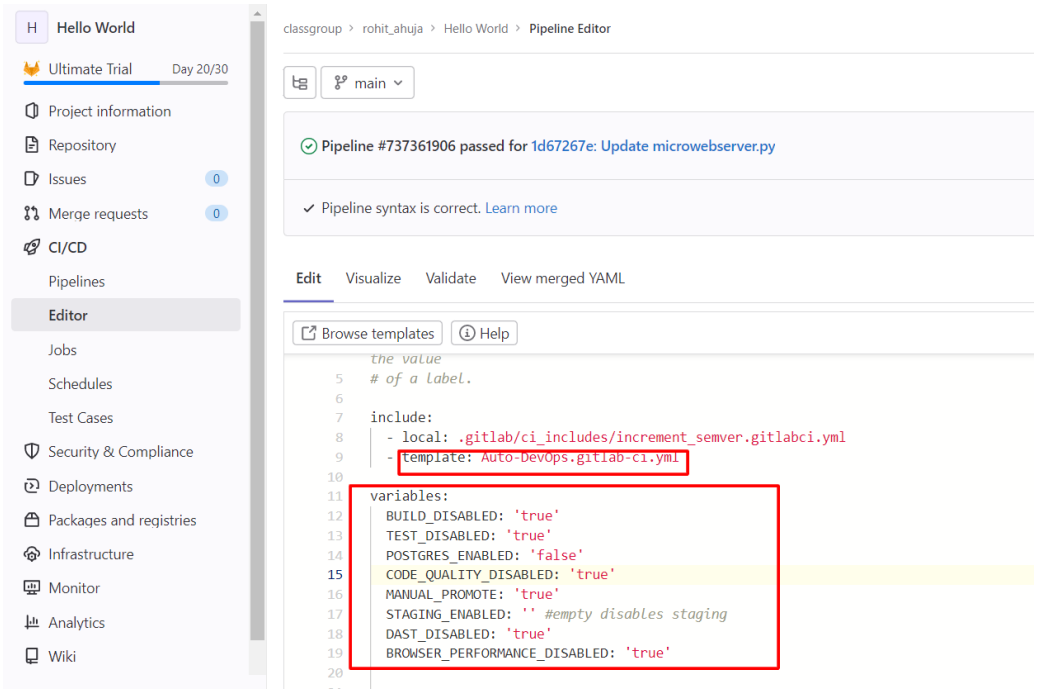

Figure 28:Adding Security Scanning to Application

5. Save the changes to the configuration file by committing and pushing the changes to the repository.

From now on, Gitlab will automatically run SAST on the application every time the pipeline is triggered. User can view the results of the scan in the "Pipelines" section of the project. If any vulnerabilities are found, Gitlab will provide recommendations on how to fix them.

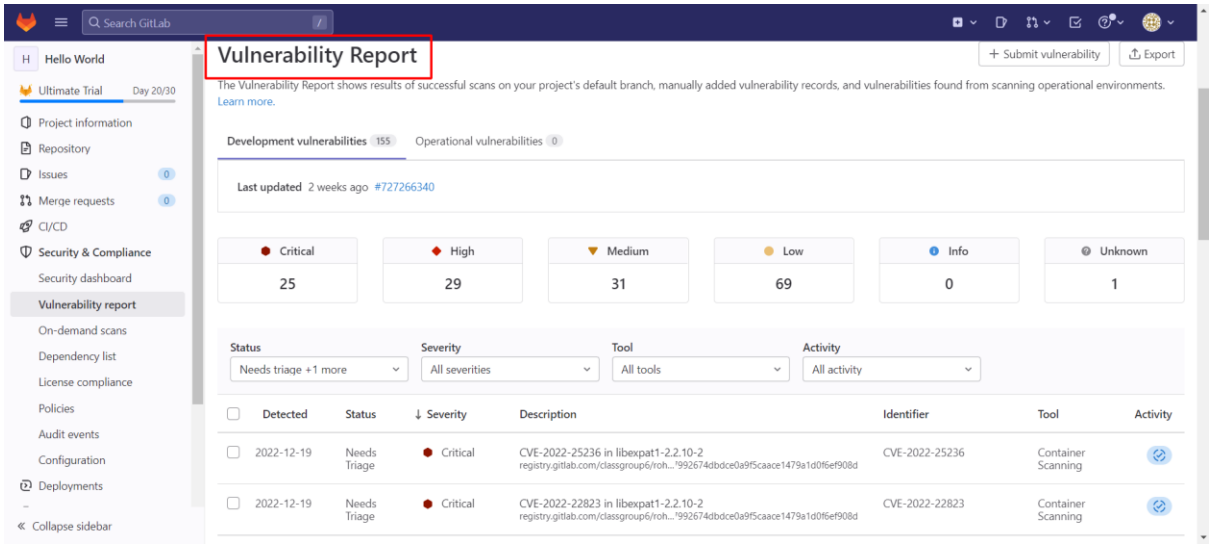

Figure 29: Application Security Testing

To add Kubernetes Manifests Scanning :
Edit the update manifests file in the application environment repository and add the below code. This will enable kubernetes manifests scanning for the application.

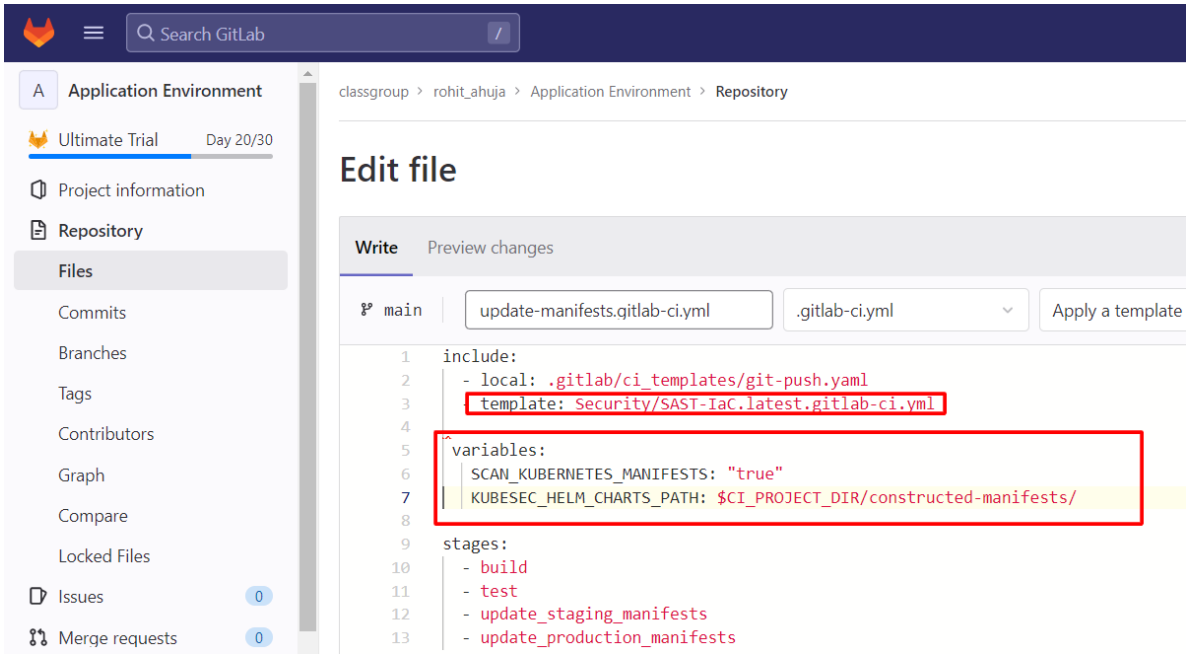

Figure 30:Adding Kubernetes Manifests Scanning

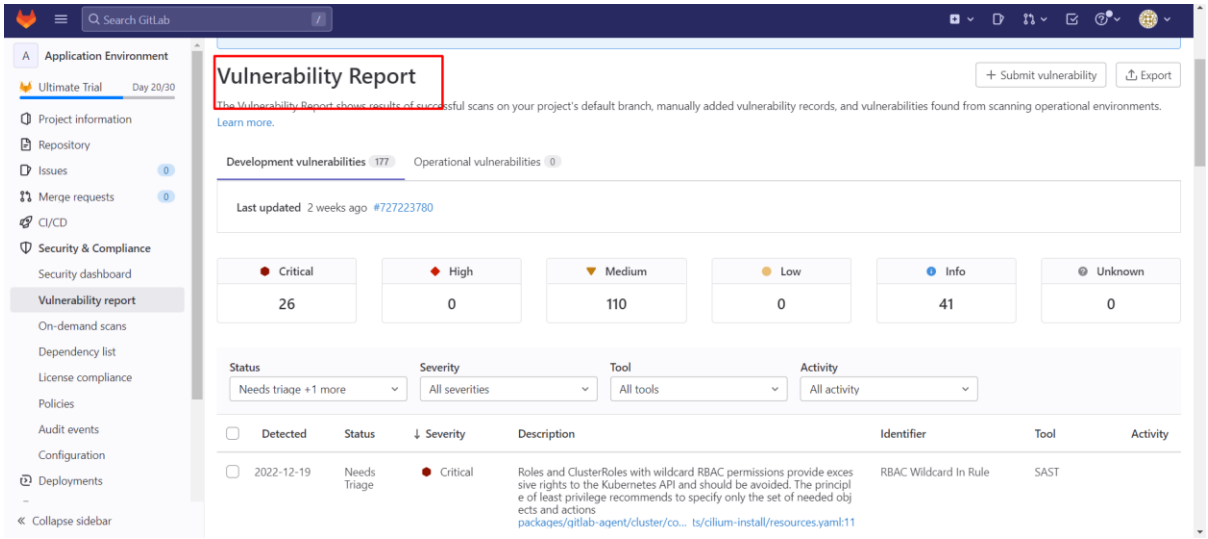

Figure 31: Kubernetes Manifests Scanning Report

To add container scanning using Gitlab Agent for Kubernetes:

Edit the following file and add the mentioned lines:

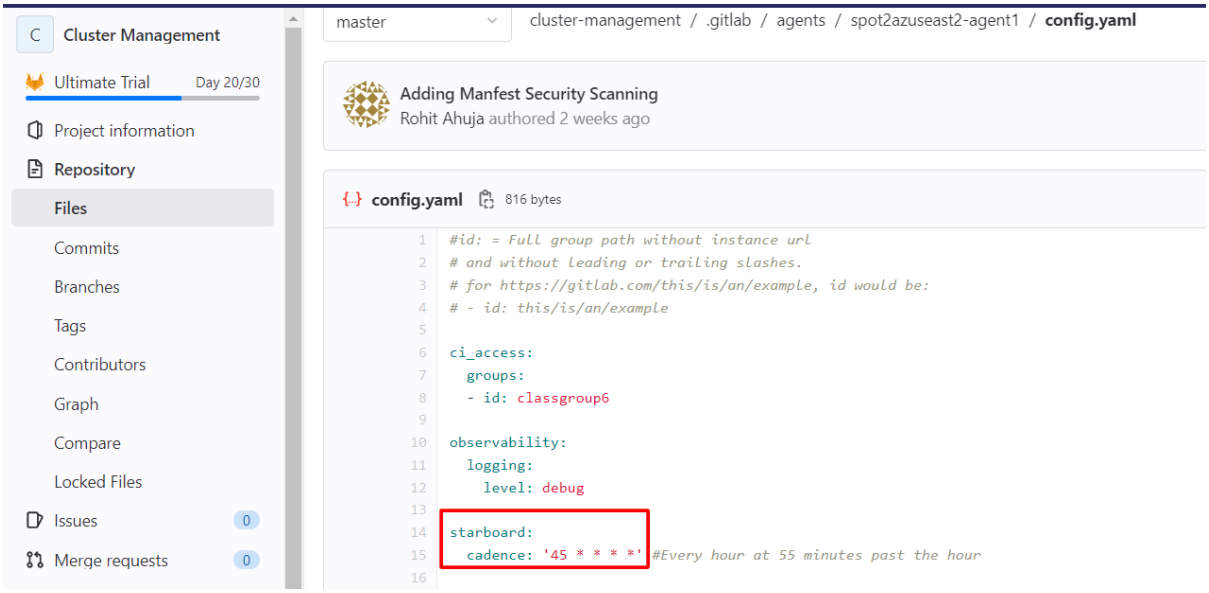

Figure 32:Adding Container Scanning

Starboard tool developed by aqua security has been used which automatically scans Kubernetes containers.

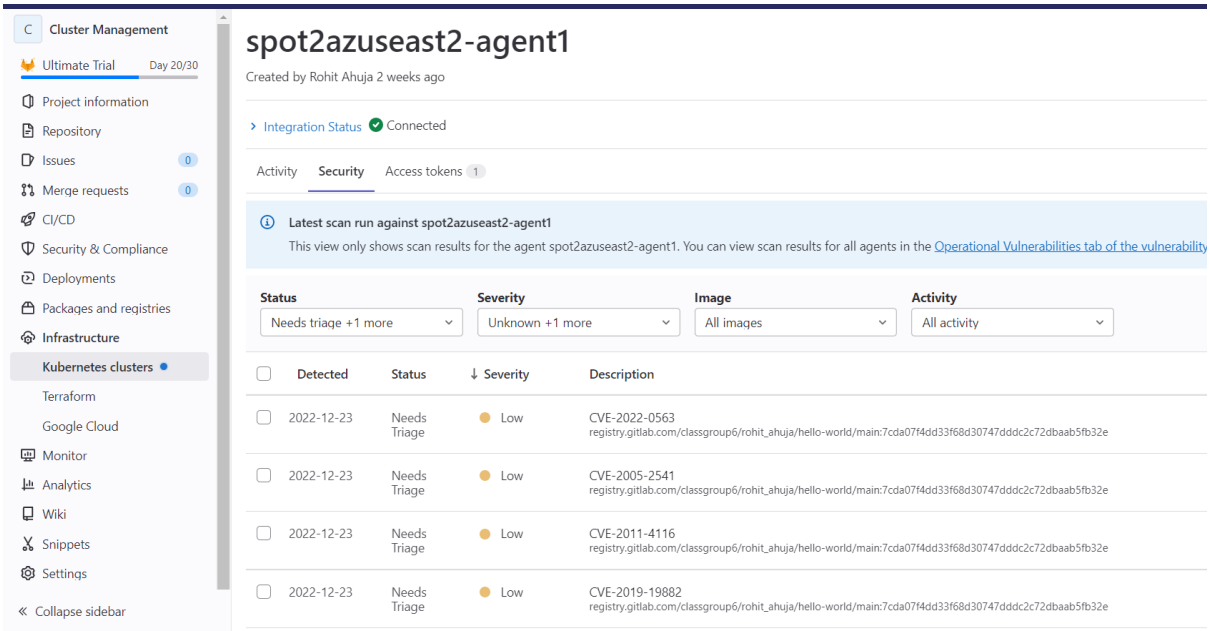

Figure 33: Container Scanning Report View

10 Scanning Kubernetes Cluster using Kube-Tools

1. Using Kube-Hunter

   Kube-Hunter was deployed in one of the Kubernetes pods using the following steps:

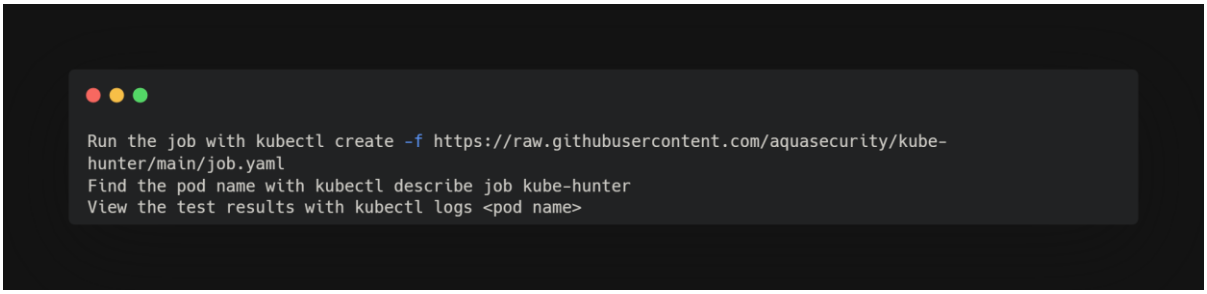

Figure 34: Code to install kube-hunter

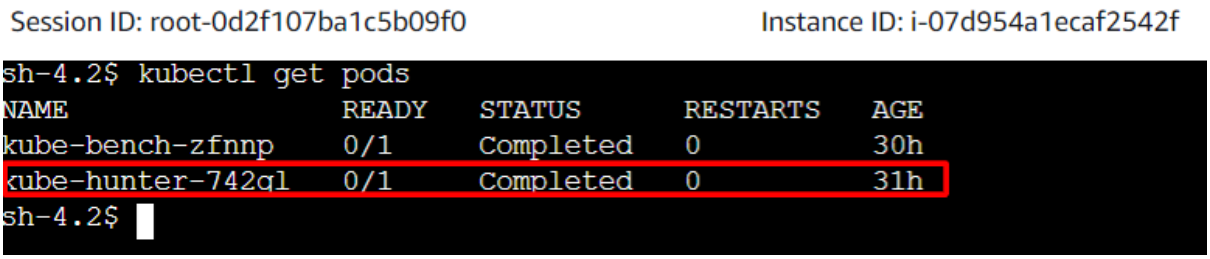

Figure 35: Kube-hunter in pod

2. Scanning using Kube-Scape

   Scanning using Kube-Scape was done by following the below steps:

   User need to create an account first on kubescape cloud portal and extract the account id and fire the below commands in kubernetes cluster:

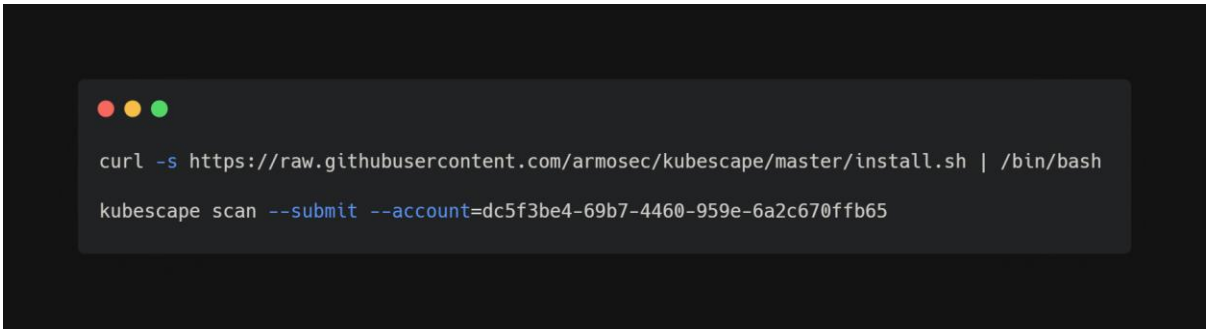

Figure 36: Code to perform Kube-Scape scan

The kubescape will scan the cluster and save the results in the cloud dashboard.

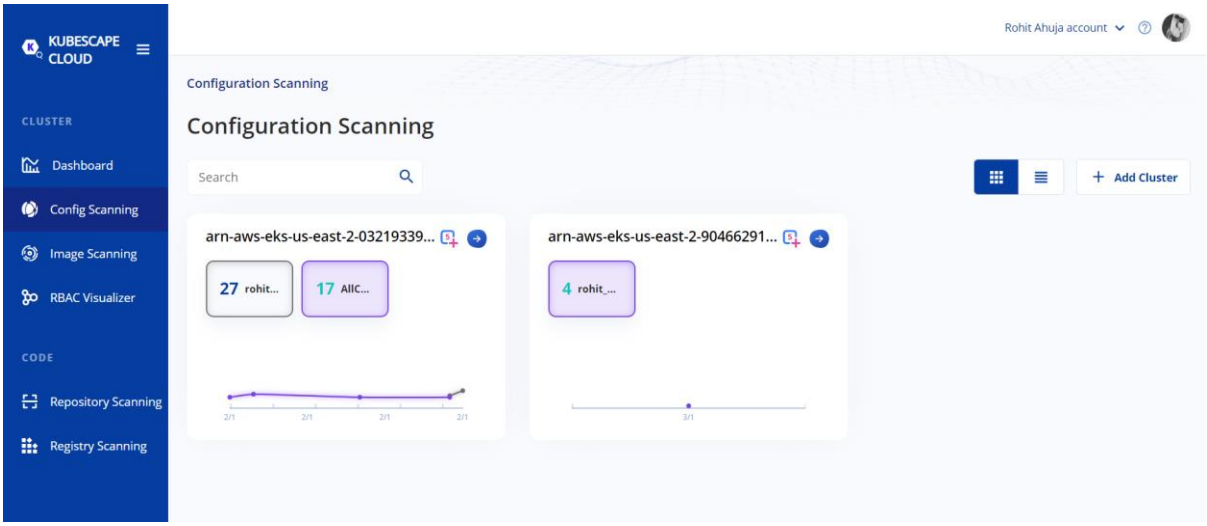

Figure 37:Kube Scape Cloud Portal

References

aquasecurity/kube-hunter: Hunt for security weaknesses in Kubernetes clusters Available at: https://github.com/aquasecurity/kube-hunter (Accessed: 3 January 2023).

Connecting a Kubernetes cluster with GitLab | GitLab (no date). Available at: https://docs.gitlab.com/ee/user/clusters/agent/ (Accessed: 3 January 2023).

Kubescape. Available at: https://cloud.armosec.io/config-scanning (Accessed: 3 January 2023).

Starboard. Available at: https://aquasecurity.github.io/starboard/v0.15.6/ (Accessed: 3 January 2023).

Appendix H – Monthly Internship Activity Report

The Internship Activity Report is a 1-page monthly summary of the activities performed by you andwhat you have learned during that month. The Internship Activity Report must be signed off by your Company and included in the configuration manual as part of the portfolio submission.

Student Name: Rohit Anand Ahuja Student number: 21168296

Company: RiskSek Private Limited Month Commencing: October 2022

- Perform Network and Application Vulnerability Assessments and Penetration Testing activities.
- Involved in other Cyber Security Services offered by the company as required.
- Worked on researching topics and started understanding the working of Kubernetes and its architecture and its literature review.

Employer comments

Student Signature: Rohit Anand Ahuja Date: 03/01/2023

Industry Supervisor Signature: Sarat Lingamallu Date: 03/01/2023

Appendix H – Monthly Internship Activity Report

The Internship Activity Report is a 1-page monthly summary of the activities performed by you andwhat you have learned during that month. The Internship Activity Report must be signed off by your Company and included in the configuration manual as part of the portfolio submission.

Student Name: Rohit Anand Ahuja Student number: 21168296

Company: RiskSek Private Limited Month Commencing: November 2022

- Worked on documenting data breaches that commenced in November month.
- Started working on the Design Specification of the proposed research regarding securing the endpoints of the microservices using client-based authentication.
- Worked on learning about AWS EKS services and Gitlab Agent.

Employer comments

Student Signature: Rohit Anand Ahuja Date: 03/01/2023

Industry Supervisor Signature: Sarat Lingamallu Date: 03/01/2023

Appendix H – Monthly Internship Activity Report

The Internship Activity Report is a 1-page monthly summary of the activities performed by you andwhat you have learned during that month. The Internship Activity Report must be signed off by your Company and included in the configuration manual as part of the portfolio submission.

Student Name: Rohit Anand Ahuja Student number: 21168296

Company: RiskSek Private Limited Month Commencing: December 2022

- Worked on documenting data breaches commenced in December month.
- Started working on implementation of integration of AWS EKS and Gitlab. Deployed Kubernetes Cluster on AWS EKS and installed agent in one of the pods.
- Deployed projects and isolated the application build and application environment, added security scanning.
- Evaluated the proposed architecture using tools.

Employer comments

Student Signature: Rohit Anand Ahuja Date: 03/01/2023

Industry Supervisor Signature: Sarat Lingamallu Date: 03/01/2023