

Internship Project

Kubernetes Cluster Setup for Scalable Microservices

Name: Devang Tyagi

Description:

The architecture of microservices has become increasingly famous for building scalable, resilient, and manageable applications. However, successful management of microservices at scale requires orchestration and infrastructure management capability. Kubernetes is one of the best-known open-source container orchestration platforms to deploy, scale, and manage containerized applications. This project consists of establishing a Kubernetes cluster, in which the scalable microservices will run and where Kubernetes can demonstrate its superior features like auto-scaling, service discovery, and load balancing. Students will learn by field experience in deploying microservices, configuring Kubernetes clusters, and assigning resources for optimizing infrastructure for scalability, reliability, and fault tolerance.

Objective:

- 1. Learn about Microservice Architecture:** Understand the fundamentals of microservices, concentrating on how the services are constructed, deployed, and communicated with one another.
- 2. Introduction to Kubernetes Basics:** Become well versed in the architecture of Kubernetes, and its huge components: clusters, pods, services, deployments, and scaling techniques.
- 3. Build Scalable Infrastructure:** Design such Kubernetes architecture that enables the dynamic scaling of microservices depending on demand, while not compromising on high availability and fault tolerance.
- 4. Establish CI/CD Pipelines:** Create a CI/CD pipeline to automate microservice deployments on the Kubernetes cluster.
- 5. Monitor and Maintain the Cluster:** Install monitoring and logging tools that will be utilized to track the health and performance of microservices in a Kubernetes cluster.
- 6. Cost and Performance Optimization:** You'll analyze and optimize resource usage in the Kubernetes cluster to strike the right balance between performance and cost.

Technologies used:

1. Azure (Azure Cloud Portal)
 - Kubernetes Services (creating Kubernetes cluster).
 - Container Registry
 - Prometheus
 - Grafana
2. Azure DevOps (Azure DevOps Portal)
 - Classic Editor (for building and pushing docker image to desired container registry)
 - Pipelines (one for building the project and another for deploying the project.)
3. Azure CLI (in local host)
4. GitHub link (to import files to Azure DevOps)

AKS: Azure Kubernetes Service, a container orchestration service that is a managed Kubernetes service and allows deploying, managing, and scaling containerized applications. We will set up AKS in Azure Cloud as part of the lab exercise.

Container Registry: Utilities and considered being for storage, manageability, and securing deployment services of the container image empowered by Docker.

Prometheus: Open-source instrumentation tool built for capturing and querying metrics in container environments.

Grafana: Visualization platform, an interactive dashboard tool for monitoring and analyzing different metrics.

Azure DevOps: An online service supported by Microsoft with features for source control, CI/CD, and project management.

Classic Editor (for building and pushing Docker images to desired container registry): A GUI-based pipeline editor in Azure DevOps that accepts pipeline definition that, in turn, automates Docker builds and pushes.

Pipelines: CI/CD automation tools for compiling, testing, and deploying applications.

Azure CLI: A command-line tool meant for Azure resource management and going forth automates tasks for backend cloud service users.

GitHub link: A repository URL with which Azure DevOps can fetch source code for builds and deployments.

Create a Registry and Kubernetes Cluster in Azure Portal

Create a Container registry:

Create a container registry where the docker image will be pushed.

Microsoft Azure

Search resources, services, and docs (G+)

Home > Container registries >

Create container registry

Basics Networking Encryption Tags Review + create

Azure Container Registry allows you to build, store, and manage container images and artifacts in a private registry for all types of container deployments. Use Azure container registries with your existing container development and deployment pipelines. Use Azure Container Registry Tasks to build container images in Azure on-demand, or automate builds triggered by source code updates, updates to a container's base image, or timers. [Learn more](#)

Project details

Subscription * Pay-As-You-Go (1a87e9ea-afbe-42e9-a7e5-0a146a6a8015) ▾

Resource group * ▾
[Create new](#)

Instance details

Registry name * Enter the name .azurecr.io

Location * Central India ▾

Use availability zones ⓘ
Availability zones are activated on premium registries and in regions that support availability zones. [Learn more](#)

Pricing plan * ⓘ Standard ▾

[Review + create](#) [< Previous](#) [Next: Networking >](#)

Create a Container registry:

Once the registry is created successfully the ‘Provisioning State’ will show the state as ‘Succeeded’.

The screenshot shows the Azure portal interface for a Container registry named 'mylabacr14'. The left sidebar contains navigation links: Overview, Activity log, Access control (IAM), Tags, Quick start, Events, and Settings. The main content area is titled 'Overview' under 'Essentials'. It displays the following details:

Setting	Value
Resource group (move)	MyLab-RG
Location	Central India
Subscription (move)	Pay-As-You-Go
Subscription ID	1a87e9ea-afbe-42e9-a7e5-0a146a6a8015
Soft delete (Preview)	Disabled
Tags (edit)	Add tags
Login server	mylabacr14.azurecr.io
Creation date	1/31/2025, 2:22 PM GMT+5:30
Provisioning state	Succeeded
Pricing plan	Standard

Create a Kubernetes Cluster:

1. The latest view is a bit different for creating a Kubernetes cluster so to create a cluster choose the desired option.
2. For this experiment, the shadowed option was used.

≡ Microsoft Azure

Home >

Kubernetes services

Default Directory (devangtyagi622@gmail.onmicrosoft.com)

+ Create Manage view Refresh Export

- Deploy application (new)
Deploy your application to a Kubernetes cluster.
- Kubernetes cluster
Customizable setup for added control and flexibility.
- Automatic Kubernetes cluster (preview)
Automated operations for streamlined application deployment.
- Add a Kubernetes cluster with Azure Arc
- Create a Kubernetes cluster with Azure Arc

Create a Kubernetes Cluster:

The options were selected for saving the cost but for desired option change only the following options:

1. Resource Group
2. Cluster preset configuration
3. Kubernetes cluster name

The screenshot shows the 'Create Kubernetes cluster' page in the Microsoft Azure portal. The top navigation bar includes 'Home > Kubernetes services > Create Kubernetes cluster ...'. Below the navigation, there are tabs for 'Basics', 'Node pools', 'Networking', 'Integrations', 'Monitoring', 'Security', 'Advanced', 'Tags', and 'Review + create'. The 'Basics' tab is selected.

Project details

Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * (dropdown): Pay-As-You-Go (1a87e9ea-afbe-42e9-a7e5-0a146a6a8015)

Resource group * (dropdown): (New) mylab-aks group
Create new

Cluster details

Cluster preset configuration * (dropdown): Production Standard

To quickly customize your Kubernetes cluster, choose one of the preset configurations above. You can modify these configurations at any time.
[Compare presets](#)

Kubernetes cluster name * (input field): mylab-aks

Region * (dropdown): (Asia Pacific) Central India

Availability zones (dropdown): None

AKS pricing tier (dropdown): Free

(Info): Maximize cost savings by [switching your Cluster Preset to 'Dev/Test'](#). However, this will greatly impact your cluster's availability.

Kubernetes version * (dropdown): 1.30.7 (default)

Automatic upgrade (dropdown): Disabled

Automatic upgrade scheduler (dropdown): No schedule
[Add schedule](#)

Node security channel type (dropdown): None

Security channel scheduler (dropdown): No schedule
[Add schedule](#)

Choose between local accounts or Microsoft Entra ID for authentication and Azure RBAC or Kubernetes RBAC for your authorization needs.

Authentication and Authorization (dropdown): Local accounts with Kubernetes RBAC

(Info): Once the cluster is deployed, use the Kubernetes CLI to manage RBAC configurations. [Learn more](#)

Create a Kubernetes Cluster:

In the next section for node pool click on the node pool name and make the required changes

The screenshot shows the 'Node pools' tab of the 'Create Kubernetes cluster' wizard in the Microsoft Azure portal. The tab is highlighted with a blue underline. The page includes a navigation bar with 'Home > Kubernetes services >' and a search bar. Below the tabs, there's a section titled 'Node pools' with a note about optional node pools for workloads. A table lists one node pool named 'agentpool' with details like mode (System), size (Standard_D2s_v3), OS (Ubuntu), and count (2). There are 'Add' and 'Delete' buttons above the table. Further down, sections for 'Enable virtual nodes' and 'Node pool OS disk encryption' are shown, each with their own configuration options and links to learn more.

Microsoft Azure

Home > Kubernetes services >

Create Kubernetes cluster

Basics **Node pools** Networking Integrations Monitoring Security Advanced Tags Review + create

Node pools

In addition to the required primary node pool configured on the Basics tab, you can also add optional node pools to handle a variety of workloads [Learn more](#)

<input type="checkbox"/>	Name	Mode	Node size	OS SKU	Node count	Available
<input type="checkbox"/>	agentpool	System	Standard_D2s_v3 (c...)	Ubuntu	2	None

Enable virtual nodes

Virtual nodes allow burstable scaling backed by serverless Azure Container Instances. [Learn more](#)

Enable virtual nodes

Node pool OS disk encryption

By default, all disks in AKS are encrypted at rest with Microsoft-managed keys. For additional control over encryption, you can supply your own keys using a disk encryption set backed by an Azure Key Vault. The disk encryption set will be used to encrypt the OS disks for all node pools in the cluster. [Learn more](#)

Encryption type

Previous Next **Review + create**

Create a Kubernetes Cluster:

For this experiment the desired changes are as follows:

1. The scaling method was ‘Manual’
2. The VM size was D2s_v3.

Microsoft Azure Search resources, se

Home > Kubernetes services > Create Kubernetes cluster >

Update node pool

mylab-aks

Node pool name * ①

Mode * System ①

ⓘ The primary node pool must be a system node pool to support system pods.

OS SKU * Ubuntu Linux ①

ⓘ Linux is required for system node pools.

Availability zones ①

Enable Azure Spot instances ①

ⓘ Azure Spot instances cannot be used with system node pools.

Node size * Standard D8ds v5 ①

8 vcpus, 32 GiB memory

[Choose a size](#)

ⓘ This option is recommended so that the cluster is automatically sized correctly for the current running workloads.

Scale method Manual ①

Autoscale - **Recommended**

ⓘ This option is recommended so that the cluster is automatically sized correctly for the current running workloads.

Minimum node count * ①

Maximum node count * ①

The maximum node count allowed for an AKS cluster is 1000 per node pool and 5000 nodes across all node pools in this cluster.

Optional settings

Update **Cancel**

Node Size:

VM Size ↑↓	Type ↑↓	vCPUs ↑↓	RAM (GiB) ↑↓	Data disks ↑↓	Max IOPS ↑↓	Local storage (GiB) ↑↓	Premium disk ↑↓	Cost/month ↑↓
ⓘ Most used by Azure users ↗								
D2s_v3 ↗	General purpose	2	8	4	3200	16 (SCSI)	Supported	Loading...
D2as_v4 ↗	General purpose	2	8	4	3200	16 (SCSI)	Supported	Loading...

Create a Kubernetes Cluster:

In the container network select Kubenet

Microsoft Azure Search resources, services, and activity

Home > Kubernetes services >

Create Kubernetes cluster

...

Basics Node pools Networking **Networking** Integrations Monitoring Security Advanced Tags Review + create

Public access

Set authorized IP ranges

Container networking

Network configuration Azure CNI Overlay
Assigns pod IP addresses from a private IP space. Best for scalability

Azure CNI Node Subnet
Previously named Azure CNI. Assigns pod IP addresses from your host VNet. Best for workloads where pods must be reachable by other VNet resources

kubenet
Older, route table-based Overlay with limited scalability. Not recommended for most clusters

Bring your own Azure virtual network

DNS name prefix *

Network policy * None
Allow all ingress and egress traffic to the pods

Calico
Open-source networking solution. Best for large-scale deployments with strict security requirements

Azure
Native networking solution. Best for simpler deployments with basic security and networking requirements

The Azure network policy is not compatible with kubenet networking.

Load balancer Standard

[Previous](#) [Next](#) [Review + create](#)

Create a Kubernetes Cluster:

In Azure container registry select the required container registry

Microsoft Azure

Search resources, services, a

Home > Kubernetes services >

Create Kubernetes cluster

Basics Node pools Networking **Integrations** Monitoring Security Advanced Tags Review + create

Connect your AKS cluster with additional services.

Azure Container Registry
Connect your cluster to an Azure Container Registry to enable seamless deployments from a private image registry.
[Learn more ↗](#)

Container registry
[Create new](#)

Service mesh - Istio
Enable Istio to configure traffic management, maximize observability capabilities and reinforce service-to-service security measures without changing the application code. [Learn more ↗](#)

Enable Istio

Azure Policy
Apply at-scale enforcements and safeguards for AKS clusters in a centralized, consistent manner through Azure Policy. [Learn more ↗](#)

Azure Policy Enabled Disabled

[Previous](#) [Next](#) [Review + create](#)

Create a Kubernetes Cluster:

Under Monitoring select the following options:

1. Alerts
2. Prometheus
3. Grafana

Microsoft Azure

Search resources, services

Home > Kubernetes services >

Create Kubernetes cluster

Basics Node pools Networking Integrations **Monitoring** Security Advanced Tags Review + create

Azure Monitor

In addition to the CPU and memory metrics included in AKS by default, you can enable Container Insights for more comprehensive data on the overall performance and health of your cluster. Billing is based on data ingestion and retention settings.

[Learn more about container performance and health monitoring](#)

[Learn more about pricing](#)

Container Insights

Enable Container Logs Azure monitor is recommended for production standard configuration.

Log Analytics workspace * [Create new](#)

Cost Preset * [Create new](#)

1 min collection frequency
No namespaces
Syslogs Disabled

Managed Prometheus

Managed Prometheus provides a highly available, scalable, and secure metrics platform to monitor your containerized workloads. [Learn more](#)

Enable Prometheus metrics

Azure Monitor workspace * [Create new](#)

Managed Grafana

Selecting a fully managed instance of Grafana to visualize your managed Prometheus data stored in your Azure Monitor workspace. [Learn more about pricing](#)

Previous Next **Review + create**

Create a Kubernetes Cluster:

Under Monitoring select the following options:

1. Alerts
2. Prometheus
3. Grafana

Managed Grafana

Selecting a fully managed instance of Grafana to visualize your managed Prometheus data stored in your Azure Monitor workspace. [Learn more about pricing ↗](#)

Enable Grafana	<input checked="" type="checkbox"/>
Grafana workspace *	<input type="text" value="grafana-20250202102527 (devops)"/> ▼
Create new	

Alerts

To Monitor events on your cluster, enable alerts recommended for your monitoring configuration above (you can select and customize the rules) [Learn more about recommended alert rules ↗](#)

Enable recommended alert rules ⓘ	<input checked="" type="checkbox"/>
Alert rules	<div style="border-left: 2px solid #0078D4; padding-left: 10px;"><p>Alert me if</p><ul style="list-style-type: none">• CPU Usage Percentage is greater than 95%• Memory Working Set Percentage is greater than 100%<p>Notify me by</p><ul style="list-style-type: none">• Email: devangtyagi622@gmail.com</div>

[Previous](#)

[Next](#)

[Review + create](#)

Create a Kubernetes Cluster:

After creating the Kubernetes cluster the operation status will show 'Succeeded'

Note:

1. The screenshot was taken after the experiment was complete and to prevent further increase in cost the cluster was stopped.
2. It is always recommended to stop the cluster when it is used to reduce the cost, as this action reduces the need to delete the cluster and saves money.

The screenshot shows the Azure portal interface for managing a Kubernetes service named 'mylab-aks'. The top navigation bar includes a search bar, Copilot, and user information. The main page displays the 'Overview' tab for the service, which is currently set to 'Properties'. The 'Essentials' section provides key details about the cluster, such as its resource group ('MyLab-RG'), Kubernetes version ('1.30.7'), and API server address ('mylab-aks-dns-umqlnypz.hcp.centralindia.azurek8s.io'). The 'Networking' section shows the cluster's IP range ('10.244.0.0/16') and network policies. The 'Configuration' section lists the Kubernetes version ('1.30.7') and auto-upgrade settings. The 'Integrations' section shows that Container insights and Service Mesh - Istio are enabled. On the left sidebar, there are sections for Activity log, Access control (IAM), Tags, Monitor, Diagnose and solve problems, Microsoft Defender for Cloud (preview), Cost analysis, and various Kubernetes resources like Namespaces, Workloads, Services and ingresses, Storage, Configuration, Custom resources, Events, and Run command. Settings and extensions are also listed at the bottom of the sidebar.

Create a Kubernetes Cluster:

Here comes the use of the CLI. It is optional to use Cloud Shell or the Azure CLI but not in every region the Cloud Shell is available so Azure CLI can be used and all the necessary commands can be accessed using the 'connect' option as shown in the picture below.

The screenshot shows the Azure portal interface for managing Kubernetes services. At the top, there's a navigation bar with 'Home > Kubernetes services >'. Below it, a card for 'mylab-aks' is displayed, which is a 'Kubernetes service'. The card includes a 'Search' bar, a 'Create' button, a 'Connect' button, and other management options like 'Start', 'Stop', 'Delete', 'Refresh', 'Open in mobile', and 'Give it a try'. A 'Filter for any field...' search bar is also present. The main content area shows the 'Overview' tab selected, with a 'Essentials' section below it. The overall theme is blue and white, typical of the Azure branding.

Connect to mylab-aks

1. [Install Azure CLI](#)

2. [Install kubectl](#)

Set cluster context

1. Open terminal

2. Run the following commands

Login to your azure account

```
az login
```



Set the cluster subscription

```
az account set --subscription 1a87e9ea-afbe-42e9-a7e5-0a146a6a8015
```



Download cluster credentials

```
az aks get-credentials --resource-group MyLab-RG --name mylab-aks --overwri...
```



Sample commands

Once you have run the command above to connect to the cluster, you can run any kubectl commands. Here are a few examples of useful commands you can try.

List all deployments in all namespaces

```
kubectl get deployments --all-namespaces=true
```



List all deployments in a specific namespace

```
kubectl get deployments --namespace <namespace-name>
```



List details about a specific deployment

```
kubectl describe deployment <deployment-name> --namespace <namespace-na...
```



Get logs for all pods with a specific label

```
kubectl logs -l <label-key>=<label-value>
```



Create a CI/CD pipeline in Azure DevOps

Create a Repo:

When you open the Azure DevOps Portal, sign-in using the email and password used in the Azure Portal sign-in.

1. Create the project for this experiment 'first_proj' was used

The screenshot shows the Azure DevOps Portal's organization dashboard for 'devangtyagi622'. On the left, there is a sidebar with a profile icon and the organization name. The main area displays two projects: 'first_proj' (green F icon) and 'devops-project' (purple D icon). Each project card has a '...' button at the bottom right.

2. Then click on the project name and navigate to 'repos' to 'Files'

The screenshot shows the 'first_proj' project overview page. The left sidebar includes links for Overview, Summary, Dashboards, Wiki, Boards, Repos, Pipelines, Test Plans, and Artifacts. The 'Repos' link is currently selected, and a dropdown menu is open, showing options: Files, Commits, Pushes, Branches, Tags, Pull requests, and Advanced Security. The 'Files' option is highlighted.

Create a Repo:

3. Then click on the project name and go to 'import registry'

The screenshot shows the Azure DevOps interface. On the left, there's a sidebar with icons for Overview, Boards, and Repos. The 'Repos' icon is highlighted. In the center, there's a detailed view of the 'first_proj' repository, showing its manifest files ('manifests' and 'password_generator') and an 'azure-pipelines.yml' file. At the top right, there's a dropdown menu for the repository named 'first_proj'. Below it is a search bar labeled 'Filter repositories'. The menu includes options like 'New repository', 'Import repository', and 'Manage repositories'. The 'Import repository' option is currently selected.

4. Their 'Repository type' is 'Git' by default.

5. Copy and paste the URL of the GitHub where the source is published. Then click Import.

The screenshot shows the 'Import a Git repository' dialog box. It has fields for 'Repository type' (set to 'Git'), 'Clone URL *' (containing 'https://github.com/dev23-extremis/Passwd-Generator'), 'Requires Authentication' (unchecked), and 'Name *' (containing 'Passwd-Generator'). At the bottom are 'Cancel' and 'Import' buttons.

Import a Git repository

Repository type

Git

Clone URL *

https://github.com/dev23-extremis/Passwd-Generator

Requires Authentication

Name *

Passwd-Generator

Cancel Import

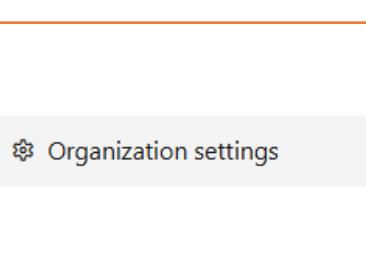
Create a Repo:

6. On the first landing page of Azure DevOps go to ‘Organization settings’.



A screenshot of the Azure DevOps landing page. At the top, there's a blue header bar with a white 'D' icon and the text 'devangtyagi622'. Below the header, a blue button labeled 'New organization' is visible. A large orange rectangular box highlights the 'Organization settings' link in the main content area.

[New organization](#)



7. Navigate to ‘Parallel Jobs’ and change ‘Microsoft-hosted’ to 1. (If your project is private this will allow you to run the pipeline otherwise, the pipeline will not run).



A screenshot of the 'Parallel jobs' section within the 'Organization settings'. The left sidebar has sections for General, Security, and Boards. The 'General' section is expanded, showing links for Overview, Projects, Users, Billing, Global notifications, Usage, Extensions, and Microsoft Entra. The 'Parallel jobs' section is shown under 'General'. It contains two main tables: 'Private projects' and 'Public projects'. Both tables show 'Microsoft-hosted' with 1 parallel job, 'Free parallel jobs' (disabled for private projects), and 'Monthly purchases' (0 for private projects). In the 'Public projects' table, the 'Self-hosted' row shows 'Unlimited' parallel jobs.

Create a Repo:

7. Navigate to Settings under Pipeline and turn-off the marked options since they are turned-on by default this will allow us to access the 'Classic Editor'.

The screenshot shows the 'Organization Settings' page in Azure DevOps. The left sidebar lists various settings categories like General, Security, Boards, Pipelines, etc. The main area displays pipeline-related settings. A specific section is highlighted with an orange border, containing three toggle switches labeled 'Off':

- Protect access to repositories in YAML pipelines**: Describes applying checks and approvals when accessing repositories from YAML pipelines.
- Disable stage chooser**: Describes users not being able to select stages to skip from the Queue Pipeline panel.
- Disable creation of classic build pipelines**: Describes no classic build pipelines can be created/imported; existing ones will continue to work.

Below this section, another toggle switch is shown as 'On':

- Trigger**
 - Limit building pull requests from forked GitHub repositories**: Describes configuring how to build pull requests from forked repositories. It includes three radio button options:
 - Disable building pull requests from forked repositories**: No pipeline will build pull requests from forked repositories.
 - Securely build pull requests from forked repositories**: Following security best practices, builds of pull requests from forked repositories do not have access to secrets or have the same permissions as regular builds. All pull requests require a team member's comment before building.
 - Customize rules for building pull requests from forked repositories**: Allows customization of rules for building pull requests from forked repositories.

At the bottom, another toggle switch is shown as 'Off':

- Disable implied YAML CI trigger**: Describes triggering a pipeline only for code changes that match its trigger section. It notes that the trigger section is missing.

Create CI Pipeline:

1. Click on the project name, then click on the option ‘ Use the classic Editor’.

The screenshot shows the Azure DevOps interface for creating a new pipeline. On the left, there's a sidebar with icons for Overview, Boards, Repos, Pipelines (which is selected), Pipelines, Environments, Releases, Library, Task groups, Deployment groups, Test Plans, and Artifacts. The main area has a header with 'Azure DevOps devangtyagi622 / first_proj / Pipelines'. Below the header, tabs for Connect, Select, Configure, and Review are visible, with 'Connect' being the active tab. A large heading 'Where is your code?' is centered. Below it, a list of source providers is shown: Azure Repos Git (YAML), Bitbucket Cloud (YAML), GitHub (YAML), GitHub Enterprise Server (YAML), Other Git, and Subversion. At the bottom, a callout box with an orange border contains the text 'Use the classic editor to create a pipeline without YAML.'

Create CI Pipeline:

2. Select the source, in this case, 'Azure Repo Git' and make the 'Default branch' to 'master'.
3. Then click continue.

The screenshot shows the 'Select a source' step in the pipeline creation wizard. At the top, there's a search bar and a toolbar with icons for filter, lock, help, and settings. Below the toolbar, the heading 'Select a source' is displayed. A row of source provider icons is shown: Azure Repos Git (selected), GitHub, GitHub Enterprise Server, Subversion, Bitbucket Cloud, and Other Git. Underneath these, there are three dropdown menus: 'Team project' set to 'first_proj', 'Repository' set to 'first_proj', and 'Default branch for manual and scheduled builds' set to 'master'. At the bottom left is a 'Continue' button.

4. Search for a template and type 'docker' then apply.

The screenshot shows the 'Select a template' step in the pipeline creation wizard. At the top, there's a search bar and a toolbar with icons for filter, lock, help, and settings. Below the toolbar, the heading 'Select a template' is displayed. A search bar contains the text 'docker'. To the right of the search bar, there are three suggested templates: 'dockers', 'docker-stable', and 'decker'. Below the search bar, there's a link 'Or start with an Empty job'. Under the heading 'Configuration as code', there's a YAML icon with the text 'YAML' and a note: 'Looking for a better experience to configure your pipelines using YAML files? Try the new YAML pipeline creation experience.' with a 'Learn more' link. In the 'Featured' section, there's a card for 'Docker container' with the subtext 'Build a Docker image and push it to a container registry.' and an 'Apply' button. In the 'Others' section, there's a card for 'Azure Service Fabric application with Docker support' with the subtext 'Build and package an Azure Service Fabric application that contains Docker images to be pushed to a container registry.'

Create CI Pipeline:

5. Name the pipeline
6. In the 'Agent pool' select 'Azure Pipelines'
7. In the 'Agent Configuration' select the desired version.

The screenshot shows the 'Build pipeline' configuration for a project named 'first_proj'. The pipeline consists of three steps: 'Get sources' (using 'first_proj' repository and 'master' branch), 'Agent job 1' (run on agent), and 'Build an image' (using Docker). The 'Build an image' step is currently selected. On the right side, the pipeline is named 'Python-app-pipeline'. The 'Agent pool' is set to 'Azure Pipelines' and the 'Agent Specification' is 'ubuntu-20.04'. Parameters are listed below.

8. Under 'Build an Image' do the following:

- Select the container registry that you created.
- Select the respective Subscription using which all the resources were built.
- Under the Docker file: leave it with the default name.
- Tick the 'Use Default Build Context'.

The screenshot shows the configuration for the 'Build an image' task under 'Agent job 1'. The task is named 'Build an image'. The 'Container Registry Type' is set to 'Azure Container Registry' and the 'Azure subscription' is 'Pay-As-You-Go (1a87e9ea-afbe-42e9-a7e5-0a146a6a8015)'. The 'Azure Container Registry' is 'mylabacr14'. The 'Action' is 'Build an image', 'Docker File' is '**/Dockerfile', and 'Build Arguments' is an empty field. The 'Use Default Build Context' checkbox is checked. The 'Image Name' field is empty.

Create CI Pipeline:

9. Under 'Push an Image' do the following:

- Select the container registry that you created.
- Select the respective Subscription using which all the resources were built.
- Under Image Name hover over (i) icon and copy-paste the text
- Tick 'Qualify Image name'.

10. Once done with the configuration Click on 'Save and queue'.

The screenshot shows the Azure DevOps Pipelines interface for a project named 'first_proj'. A pipeline named 'Python-app-pipeline' is being edited. The pipeline has three tasks: 'Get sources', 'Agent job 1', and 'Push an image'. The 'Push an image' task is currently selected. On the right, its configuration pane is displayed, showing the following settings:

- Container Registry Type: Azure Container Registry
- Azure subscription: Pay-As-You-Go (1a87e9ea-afbe-42e9-a7e5-0a146a6a801)
- Azure Container Registry: mylabacr14
- Action: Push an image
- Image Name: \$(Build.Repository.Name):\$(Build.BuildId)
- Qualify Image Name: checked

The 'Save & queue' button at the top of the pipeline editor is highlighted with a red box.

Create CI Pipeline:

9. Under 'Copy to', under 'Contents' you have to copy-paste the .yaml file name but sometimes it does not appear in the drop. So type '**/*' this will copy all the files in drop.

The screenshot shows the Azure DevOps Pipelines interface for a project named 'first_proj'. A CI pipeline named 'Python-app-pipeline' is selected. The pipeline consists of several tasks: 'Get sources', 'Agent job 1' (which contains 'Build an image' and 'Push an image'), 'Copy Files to: \${Build.ArtifactStagingDirectory}' (selected), and 'Publish Artifact: drop'. The 'Copy Files' task is expanded, showing its configuration. The 'Contents' field is set to '**/*', indicating that all files in the source folder should be copied to the target folder.

10. Under 'Publish Artifact', the default settings should not be changed.

The screenshot shows the Azure DevOps Pipelines interface for the same project and pipeline. The 'Publish Artifact: drop' task is selected. Its configuration shows the following settings: 'Path to publish' is set to '\${Build.ArtifactStagingDirectory}', 'Artifact name' is 'drop', and 'Artifact publish location' is 'Azure Pipelines'. The 'Max Artifact Size' is set to 0. The 'Advanced', 'Control Options', and 'Output Variables' sections are collapsed.

Create CI Pipeline:

9. Under 'Push an Image' do the following:

- Select the container registry that you created.
- Select the respective Subscription using which all the resources were built.
- Under Image Name hover over (i) icon and copy-paste the text
- Tick 'Qualify Image name'.

10. Once done with the configuration Click on 'Save and queue'.

The screenshot shows the Azure DevOps Pipelines interface for a project named 'first_proj'. A pipeline named 'Python-app-pipeline' is being configured. The pipeline structure is as follows:

- Get sources**: Fetches code from the 'first_proj' repository's 'master' branch.
- Agent job 1**: Runs on an agent.
 - Build an image**: Uses Docker to build an image.
 - Push an image**: Uses Docker to push the built image to a registry.
- Publish build artifacts**: Publishes build artifacts.
 - Task version**: 1.*
 - Display name**: Publish Artifact: drop
 - Path to publish**: \$(Build.ArtifactStagingDirectory)
 - Artifact name**: drop
 - Artifact publish location**: Azure Pipelines
 - Max Artifact Size**: 0
 - Advanced**, **Control Options**, and **Output Variables** sections are also present.

The 'Push an image' task is currently selected, indicated by a blue selection bar at the bottom of the pipeline stages.

Create CI Pipeline:

Once done it will be shown as below made sure to check under Publish job in drop if the files were saved.

If not Under 'Copy to', under 'Contents' you have to copy-paste the .yaml file name but sometimes it does not appear in the drop. So type '**/*' this will copy all the files in drop. Then run again.

The screenshot shows the Azure DevOps interface for a project named 'first_proj'. The left sidebar is visible with various navigation options like Overview, Boards, Repos, Pipelines, Environments, Releases, Library, Task groups, Deployment groups, Test Plans, and Artifacts. The 'Pipelines' option is selected. In the center, there's a list of 'Jobs in run #20250131.9' for a 'Python-app-pipeline'. One job, 'Publish Artifact: drop', is expanded to show its logs. The logs are displayed in a monospaced font and include the following text:

```
1 Starting: Publish Artifact: drop
2 -----
3 Task      : Publish build artifacts
4 Description : Publish build artifacts to Azure Pipelines or a Windows file share
5 Version   : 1.247.1
6 Author    : Microsoft Corporation
7 Help      : https://docs.microsoft.com/azure/devops/pipelines/tasks/utility/publish-build-artifacts
8 -----
9 Async Command Start: Upload Artifact
10 Uploading 6 files
11 File upload succeed.
12 Upload '/home/vsts/work/1/a' to file container: '#/14894694/drop' ↵
13 Associated artifact 1 with build 30
14 Async Command End: Upload Artifact
15 Finishing: Publish Artifact: drop
```

A 'Copy permalink' button is located at the bottom right of the log area. The top of the browser window shows several tabs and the URL https://dev.azure.com/devangtyagi622/first_proj/_build/results?buildId=30&view=logs&j=275f1d19-1bd8-5591-b06b-....

Create a CD or release Pipeline:

Here the CI pipeline will be exposed.

1. Click 'New' and choose 'New release Pipeline'.

The screenshot shows the Azure DevOps interface for a project named 'first_proj'. The left sidebar is visible with various navigation options like Overview, Boards, Repos, Pipelines, Releases, Library, Task groups, Deployment groups, Test Plans, and Artifacts. The 'Releases' option is currently selected. In the main area, there is a search bar at the top labeled 'Search all pipelines'. Below it, a button labeled '+ New' is highlighted with an orange box. A list of existing pipelines is shown: 'New release pipeline (2)' (highlighted with an orange box), 'New release pipeline (1)', 'Release2', and 'New release pipeline'. The 'New release pipeline (2)' card shows details: 'Release-1', '20250202....', 'master', '2/2/2025, 2:08:14 PM', and 'Stage 1'. The URL at the bottom of the page is https://dev.azure.com/devangtyagi622/first_proj/_release.

Create a CD or release Pipeline:

Here the CI pipeline will be exposed.

2 . Click On ‘Stage 1’ and then select the template ‘Deploy to a Kubernetes cluster’.

The screenshot shows the Azure DevOps interface for creating a new release pipeline. On the left, the sidebar lists various project management and development tools: Overview, Boards, Repos, Pipelines, Environments, Releases, Library, Task groups, Deployment groups, Test Plans, Artifacts, and Project settings. The main area displays the 'All pipelines > New release pipeline' screen. It features two main sections: 'Artifacts' (with options to add artifacts and set a schedule) and 'Stages' (with an 'Add' button). A callout box highlights 'Stage 1 Select a template'. To the right, a list of available templates is shown, with 'Deploy to a Kubernetes cluster' being selected. Other visible templates include 'Deploy a Node.js app to Azure App Service', 'Deploy a PHP app to Azure App Service and Azure Database for MySQL', 'Deploy a Python app to Azure App Service and Azure database for MySQL', and 'IIS website and SQL database deployment'. An 'Apply' button is located at the bottom right of the template list.

Create a CD or release Pipeline:

2 . Click On ‘Add an artifact’ and then select the source and click ‘Add’

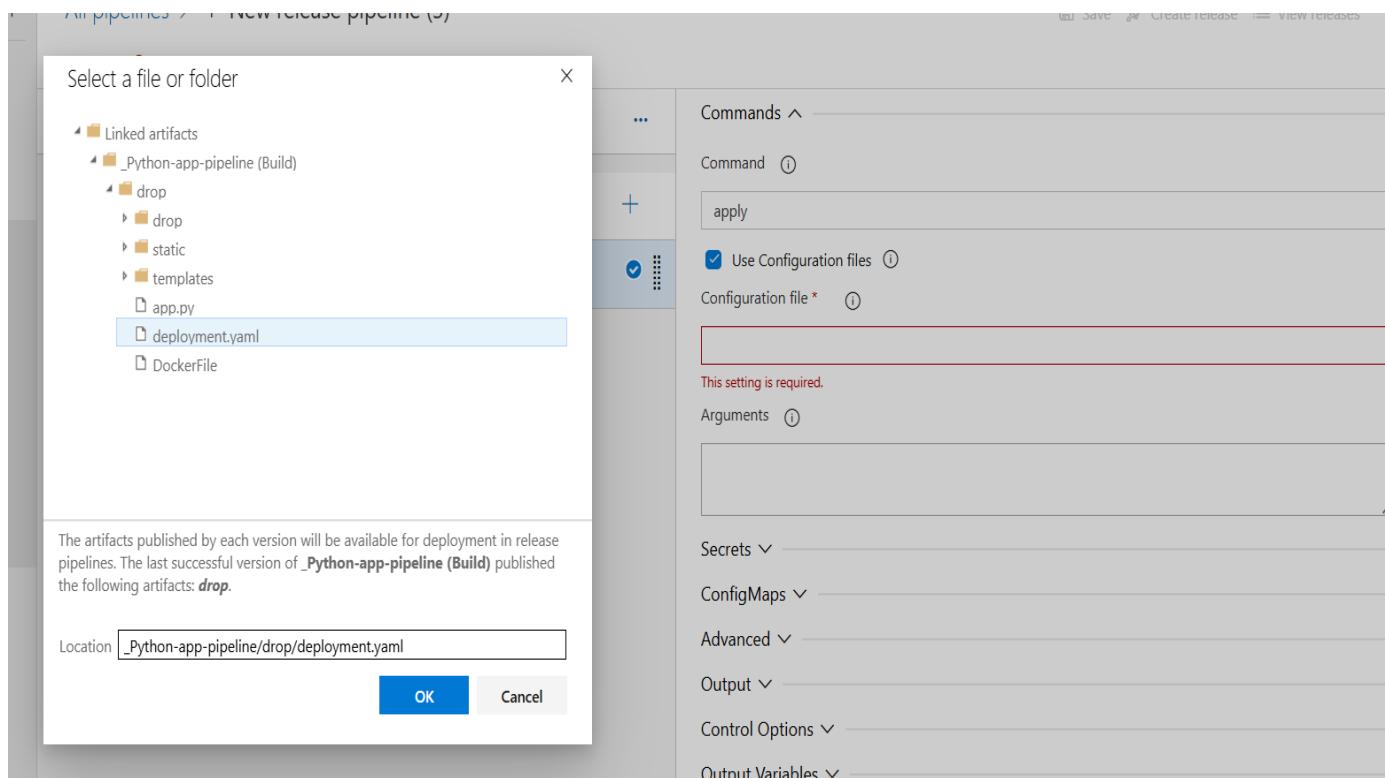
The screenshot shows the 'New release pipeline' configuration screen. In the 'Artifacts' section, there is a button labeled 'Add an artifact'. To the right, under 'Stages', there is a stage named 'Deploy to AKS' which contains one job and one task. The 'Source type' dropdown is set to 'Build', and the 'Project' dropdown is set to 'first_proj'. The 'Source (build pipeline)' dropdown is set to 'Python-app-pipeline'. The 'Default version' dropdown is set to 'Latest'. The 'Source alias' dropdown is set to '_Python-app-pipeline'. A note at the bottom says 'No version is available for Python-app-pipeline or the latest version has no artifacts to publish. Please check the source pipeline.' A blue 'Add' button is located at the bottom right of the artifact configuration area.

3. Click on ‘kubectl’, then select the ‘Azure subscription’ give ‘namespace’ name ‘default’ and in the service name give any name.

The screenshot shows the 'Tasks' tab of the 'New release pipeline' configuration. There is a task named 'kubectl' selected, indicated by a checkmark icon. To the right, a 'New service connection' dialog is open. Under 'Authentication method', 'Azure Subscription' is selected. Under 'Azure Subscription', 'Pay-As-You-Go (1a87e9ea-abfe-42e9-a7e5-0a146a6a8015)' is selected. Under 'Cluster', 'mylab-aks (MyLab-RG)' is selected. Under 'Details', there is a 'Service connection name' input field and a 'Description (optional)' input field. Under 'Security', 'Grant access permission to all pipelines' is checked. At the bottom right of the dialog, there is a 'Save' button.

Create a CD or release Pipeline:

- 4 . Under 'command' select 'apply'
5. Tick the configuration file.
6. Browse the .yaml file in drop as shown in the picture.



7. Click 'Save' and then click 'create release'



Create a CD or release Pipeline:

Once it is created successfully it will show the status succeeded and you check the logs also.

The screenshot shows the Azure DevOps interface for a 'New release pipeline > Release-1'. The left sidebar displays a 'Release' section with a 'Manually triggered' entry by 'devangtyagi622' on '1/31/2025, 10:17 PM' and an 'Artifacts' section for '_Python-app-pipeline 20250131.9' on 'master'. The main area shows the 'Stages' tab with 'Stage 1' listed as 'Succeeded' with '2 warnings' on '1/31/2025, 10:25 PM'. The 'Stage 1' details page is open, showing a summary of the deployment. It includes a 'Now at Release-1' section with a link to 'View all deployments', a 'Deployment succeeded' section with a timestamp of 'on 1/31/2025, 10:25 PM' and a note that it 'Ran for 17s', and a 'Manual trigger' section indicating it was triggered by 'devangtyagi622' on '1/31/2025, 10:24 PM'. The 'Associated changes' section links to 'View commits and work items' for the artifact.

Once the deployment is successful in the Kubernetes cluster under 'workloads' the namespaces should look like the below.

The screenshot shows the Azure portal interface for a Kubernetes service named 'mylab-aks'. The left sidebar has a 'Workloads' section selected. The main area is titled 'Deployments' and lists the following entries:

Name	Namespace	Ready	Age
coredns	kube-system	✓ 2/2	1 day
coredns-autoscaler	kube-system	✓ 1/1	1 day
konnectivity-agent	kube-system	✓ 2/2	1 day
metrics-server	kube-system	✓ 2/2	1 day
password-generator	default	✓ 1/1	1 day
ama-metrics	kube-system	✓ 2/2	11 minutes
ama-metrics-ksm	kube-system	✓ 1/1	11 minutes
ama-metrics-operator-targets	kube-system	✓ 1/1	11 minutes
ama-logs-rs	kube-system	✓ 1/1	7 minutes

Using the Azure CLI the IP of the cluster is exposed or using Service and ingress just under workloads.

```
C:\Windows\System32\cmd.exe
aks-userpool-40297926-vms000001 Ready <none> 13m v1.30.7
E:\edvedha\password_generator>kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
password-generator-7f94d96785-njvm4  1/1     Running   0          21h

E:\edvedha\password_generator>kubectl get deployments
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
password-generator   1/1      1          1          26h

E:\edvedha\password_generator>kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
password-generator-7f94d96785-njvm4  1/1     Running   0          21h

E:\edvedha\password_generator>kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
password-generator-7f94d96785-njvm4  1/1     Running   0          21h

E:\edvedha\password_generator>kubectl get deployments --all-namespaces=true
NAMESPACE   NAME           READY   UP-TO-DATE   AVAILABLE   AGE
default     password-generator  1/1     1           1           26h
kube-system coredns         2/2     2           2           27h
kube-system coredns-autoscaler 1/1     1           1           27h
kube-system konnectivity-agent 2/2     2           2           27h
kube-system metrics-server   2/2     2           2           27h

E:\edvedha\password_generator>kubectl get deployments
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
password-generator   1/1      1          1          26h

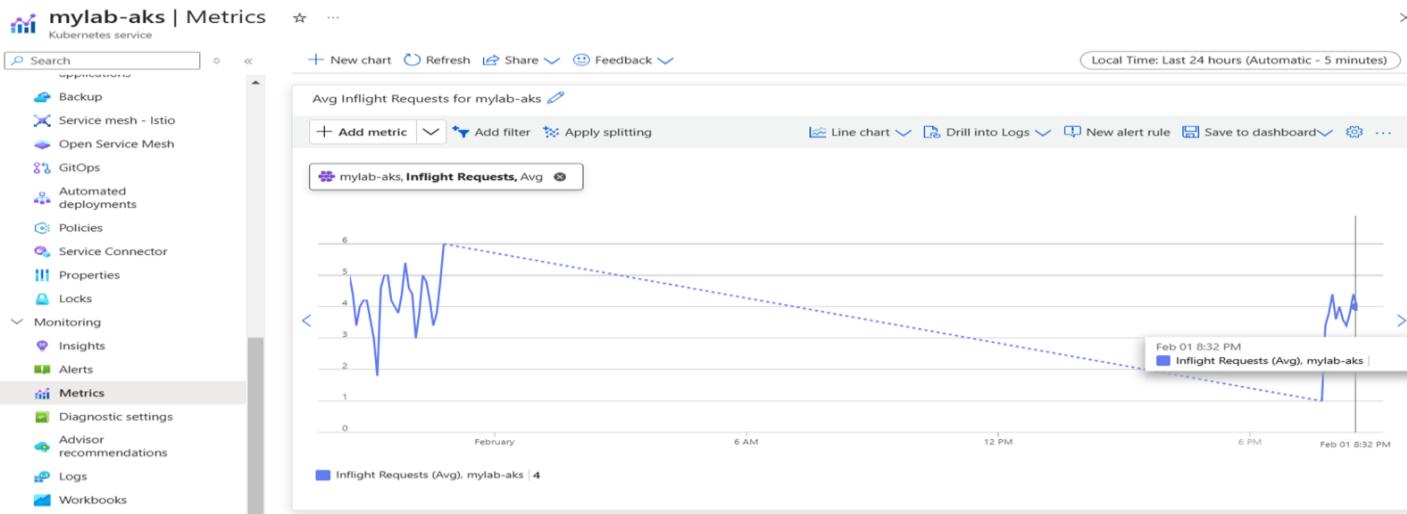
E:\edvedha\password_generator>kubectl get deployments
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
password-generator   1/1      1          1          26h

E:\edvedha\password_generator>kubectl get svc -n default
NAME            TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes      ClusterIP  10.0.0.1    <none>       443/TCP   27h
password-generator-service LoadBalancer  10.0.23.132  52.140.90.143  80:32689/TCP  26h

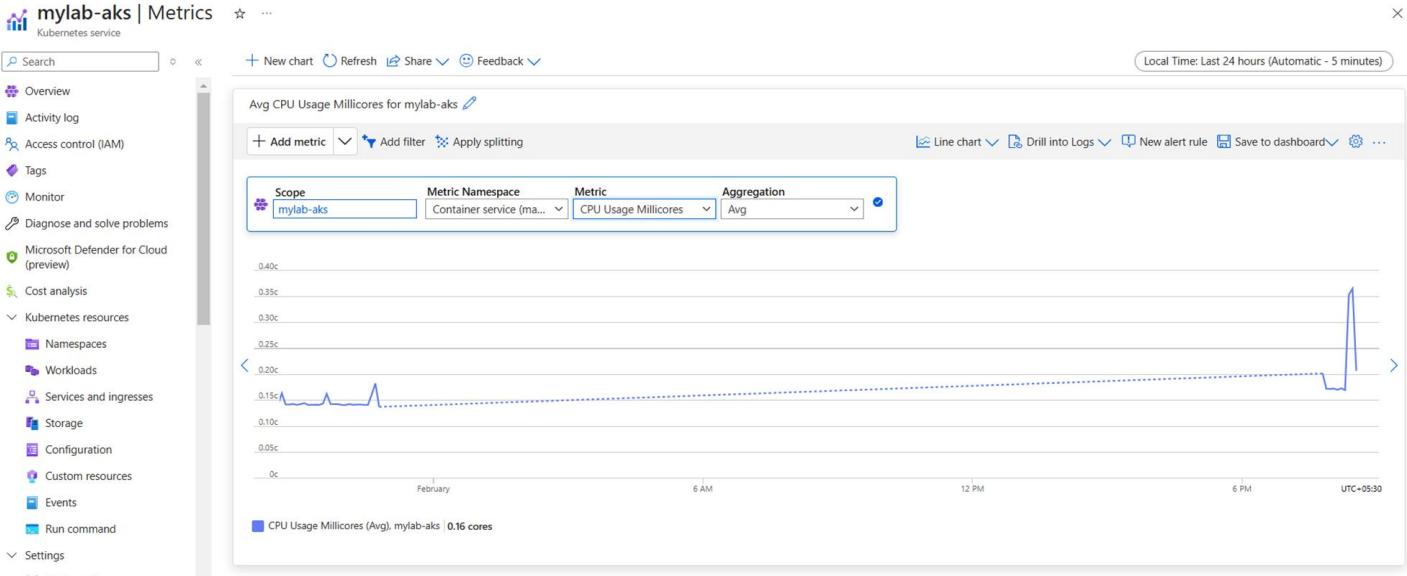
E:\edvedha\password_generator>kubectl logs password-generator-7f94d96785-njvm4
 * Serving Flask app 'app'
 * Debug mode: on
[31m[1mWARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.[0m
 * Running on http://127.0.0.1:15000
[33mPress CTRL+C to quit![0m
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 207-087-728
E:\edvedha\password_generator>
```

Metrics:

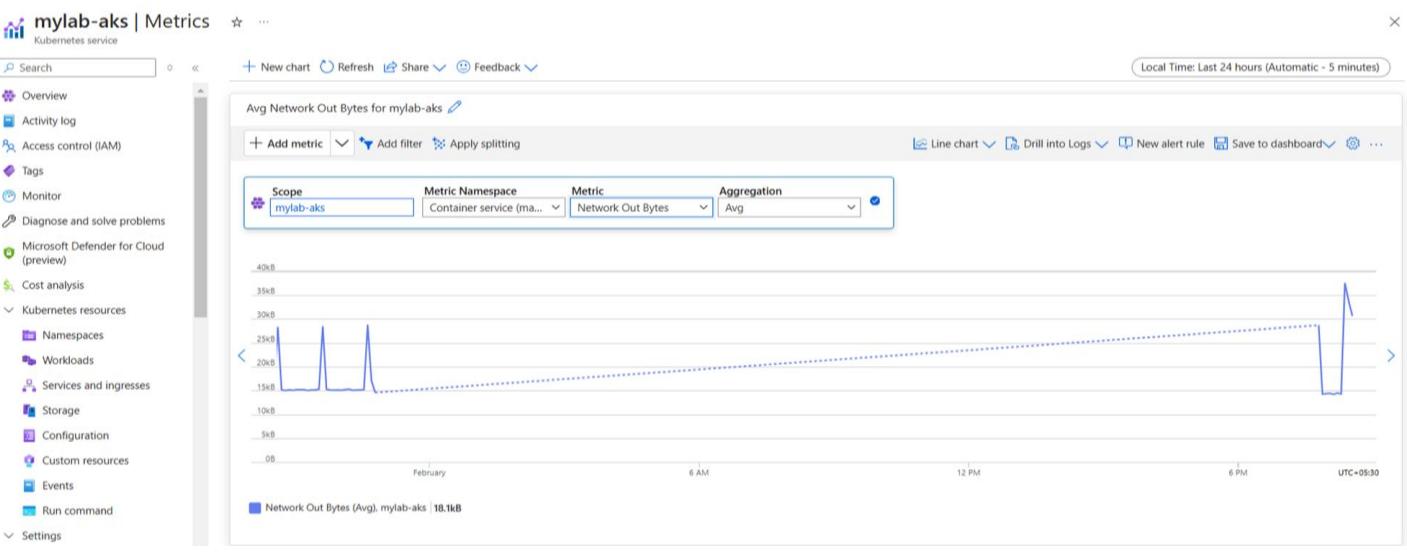
Avg. Inflight Requests:



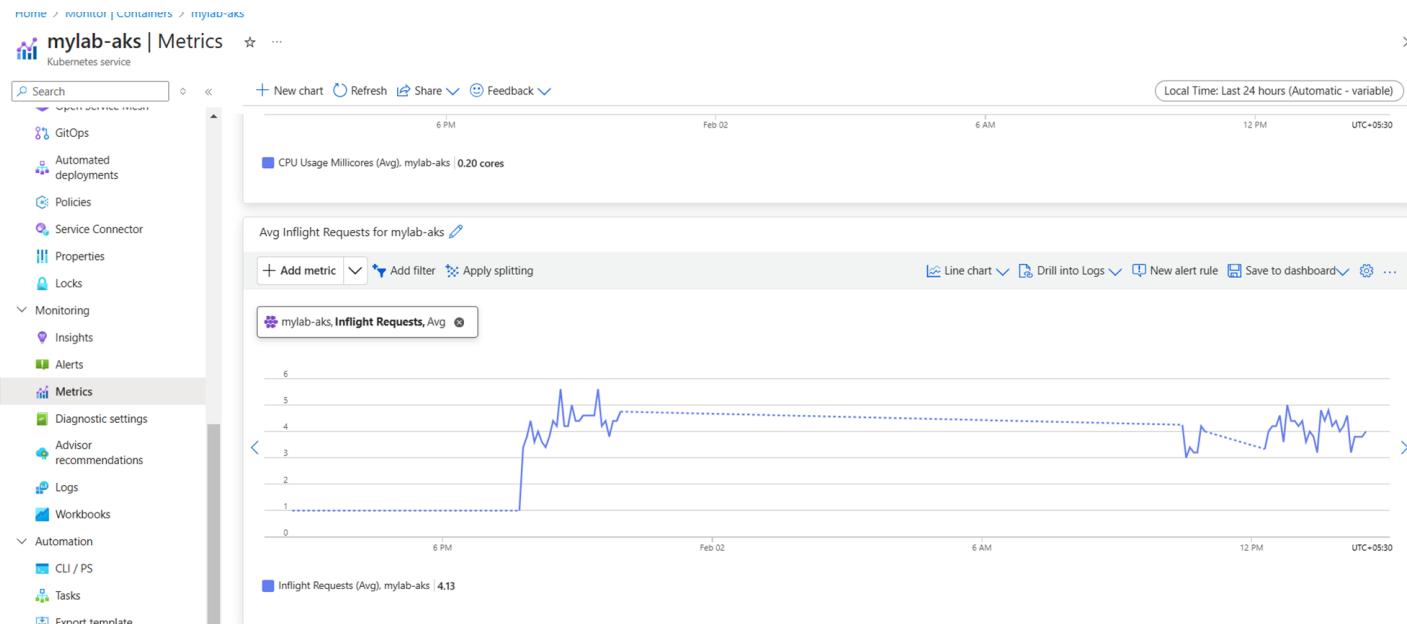
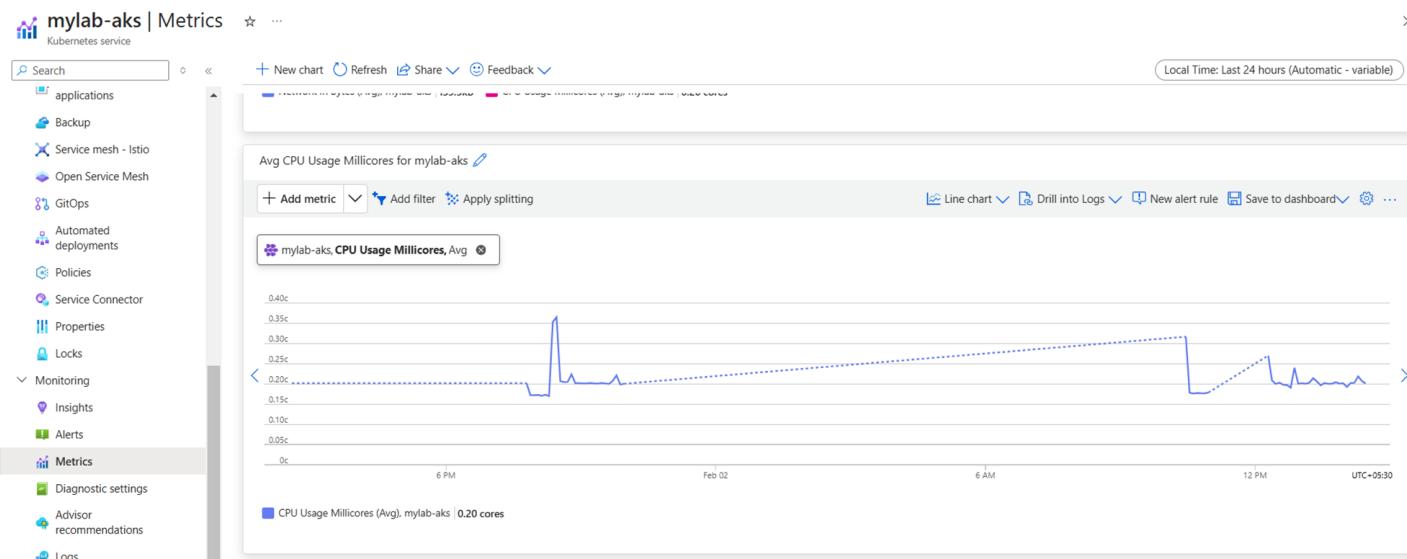
Avg. CPU utilization Millicores:



Avg. Network Out Bytes:



Metrics:



Self Diagnostics Result: (Done by Azure itself)

Self-diagnostics

Running the following self-diagnostics tasks will help you find the root cause and resolution for your error.



[Run tasks](#) [Download results](#)

Show All Details

All checks have succeeded. If you are still experiencing issues please contact Microsoft support.

Service Health insights

Confirms the availability of portal services such as Azure Resource Manager, Microsoft Entra ID, and Microsoft Graph.

[Run task again](#)

6 of 6 details

Microsoft Graph
[https://graph.microsoft.com/v1.0/\\$metadata](https://graph.microsoft.com/v1.0/$metadata)
Status: 200

Azure Resource Manager
<https://management.azure.com/healthcheck?api-version=2014-04-01>
Status: 200

Graph
[https://graph.windows.net/v1.0/\\$metadata](https://graph.windows.net/v1.0/$metadata)
Status: 200

Azure Active Directory
<https://login.microsoftonline.com/>
Status: 200

Insights
<https://insights1.exp.azure.com/>
Status: 200

Status: 200

Graph
[https://graph.windows.net/v1.0/\\$metadata](https://graph.windows.net/v1.0/$metadata)
Status: 200

Azure Active Directory
<https://login.microsoftonline.com/>
Status: 200

Insights
<https://insights1.exp.azure.com/>
Status: 200

Azure Email Orchestration
<https://ham.comms.azure.net/>
Status: 200

Note: For more information about Azure Service availability, you can visit: <https://status.azure.com/>

Verify connectivity to Azure domains

Checks your connection to Azure Resource Manager and other portal dependencies.

[Run task again](#)

8 of 8 details

Azure Resource Manager
<https://management.azure.com/healthcheck?api-version=2014-04-01>
Status: 200

React View
<https://reactblade.portal.azure.net/api/ping>
Status: 204

Hosting Service: https://hosting.portal.azure.net/
<https://hosting.portal.azure.net/api/ping>
Status: 200

Hosting Service: https://hosting.portal.azure.net/
<https://hosting.portal.azure.net/api/ping>
Status: 200

Hosting Service: https://ms.hosting.portal.azure.net/
<https://ms.hosting.portal.azure.net/api/ping>
Status: 200

Cloud Services: https://cloudservices.portal.azure.net/
<https://cloudservices.portal.azure.net/api/ping>
Status: 200

Hosting Service: https://hosting.portal.azure.net/
<https://hosting.portal.azure.net/api/ping>
Status: 200

Hosting Service: https://hosting.portal.azure.net/
<https://hosting.portal.azure.net/api/ping>
Status: 200

Hosting Service: https://ms.hosting.portal.azure.net/
<https://ms.hosting.portal.azure.net/api/ping>
Status: 200

Hosting Service: https://hosting-ms.portal.azure.net/
<https://hosting-ms.portal.azure.net/api/ping>
Status: 200

Hosting Service: https://hosting-ms.portal.azure.net/
<https://hosting-ms.portal.azure.net/api/ping>
Status: 200

Third Party Hosting Service
<https://hosting.partners.azure.net/api/ping>
Status: 200

Load an iframe

Ensures that iframes are available within the portal.

[Run task again](#)

1 of 1 details

Loading an iframe was successful.

Create web workers

Ensures that web workers are functional and communicating as designed.

[Run task again](#)

1 of 1 details

All web workers responded successfully.

Verify the manifest

Ensures the availability of the portal manifest and verifies we are able to connect to Cosmos DB.

[Run task again](#)

1 of 1 details

Manifest was successfully retrieved with a status of 200. Cosmos DB is available.

Verify persistence of cookies

Checks that cookies are able to be retrieved from browser and are not stripped on a POST request.

[Run task again](#)

1 of 1 details

Cookies were received by server and not stripped during request.

Verify the persistence of request body

Checks that the request body is not stripped on a POST request.

[Run task again](#)

1 of 1 details

Request body was not stripped while making a POST request to the server.

Verify read/write functionality to session storage

Checks whether session storage is supported and functioning properly.

[Run task again](#)

1 of 1 details

Read/write to storage was successful.

Apply Prometheus and Grafana to the Kubernetes cluster

Home > Monitor

The screenshot shows the Azure Monitor interface for Managed Prometheus. At the top, there's a search bar and navigation links for Overview, Activity log, Alerts, Metrics, Logs, Change Analysis, and Service health. Below the navigation is a banner with the text "Set up Prometheus metrics on additional AKS clusters. Learn more". A filter bar includes "Subscription == all" and "Metrics collection == All". The main table lists one Prometheus instance:

Cluster Name	Metrics collection	Workspace	Subscription	Grafana workspace
mylab-aks	Enabled	defaultazuremonitorworkspace-cin	Pay-As-You-Go	grafana-20250202102527 (+2)

The name of the Grafana attached to Kubernetes cluster is 'metrics' .

The screenshot shows the Azure portal page for a Managed Grafana resource named "metrics". The left sidebar has sections for Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Settings, Integrations, Monitoring, Alerts, and Metrics. The "Overview" section is selected. The main content area displays resource details under the "Essentials" tab:

Resource group	Endpoint
LAG-RG	https://metrics-crcsa8dvhje6epdy.cin.grafana.azure.com
Location	Provisioning State
Central India	Succeeded
Subscription	Grafana Version
Pay-As-You-Go	10.4.11
Subscription ID	Pricing Plan
1a87e9ea-afbe-42e9-a7e5-0a146a6a8015	Standard

At the bottom, there are "Tags (edit)" and "Add tags" buttons.

Grafana dashboards for the Kubernetes cluster

Home
Starred
Dashboards
Playlists
Snapshots
Library panels
Reporting
Explore
Alerting
Connections
Add new connection
Data sources
Administration

Dashboards

Create and manage dashboards to visualize your data

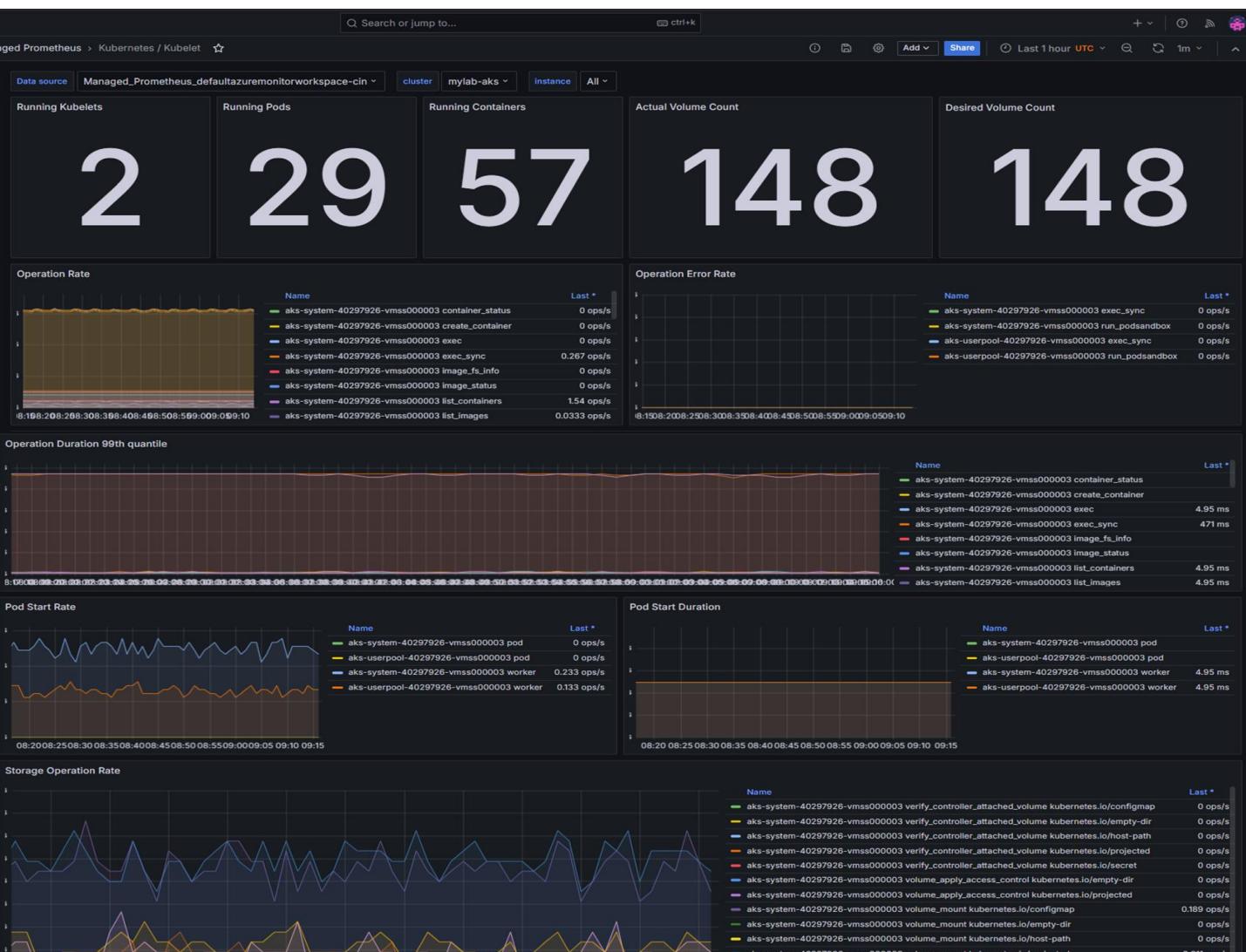
Q Search for dashboards and folders

Filter by tag Starred

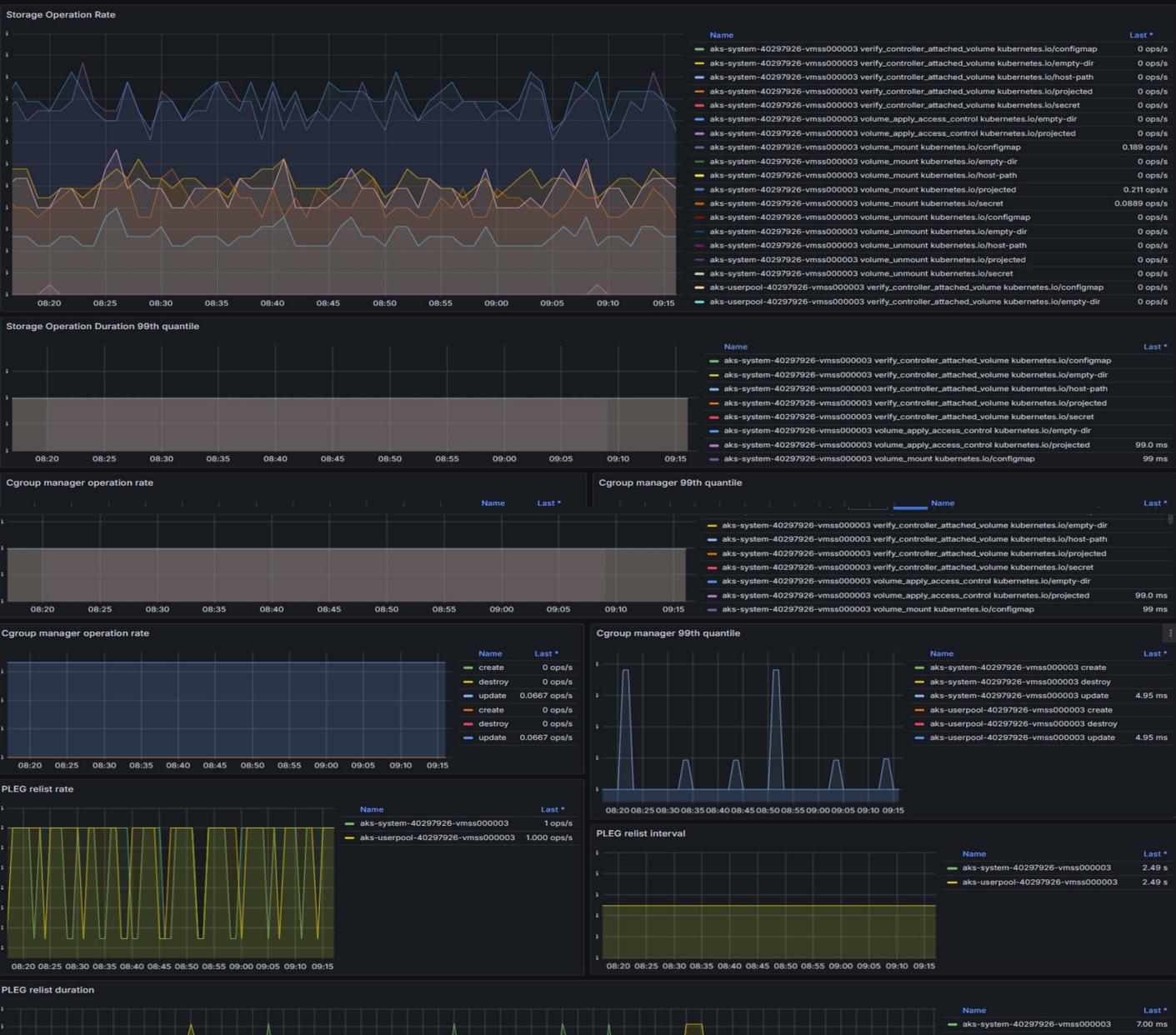
Tags

- Name
- Azure Managed Prometheus
- Azure Monitor
- Microsoft Defender for Cloud

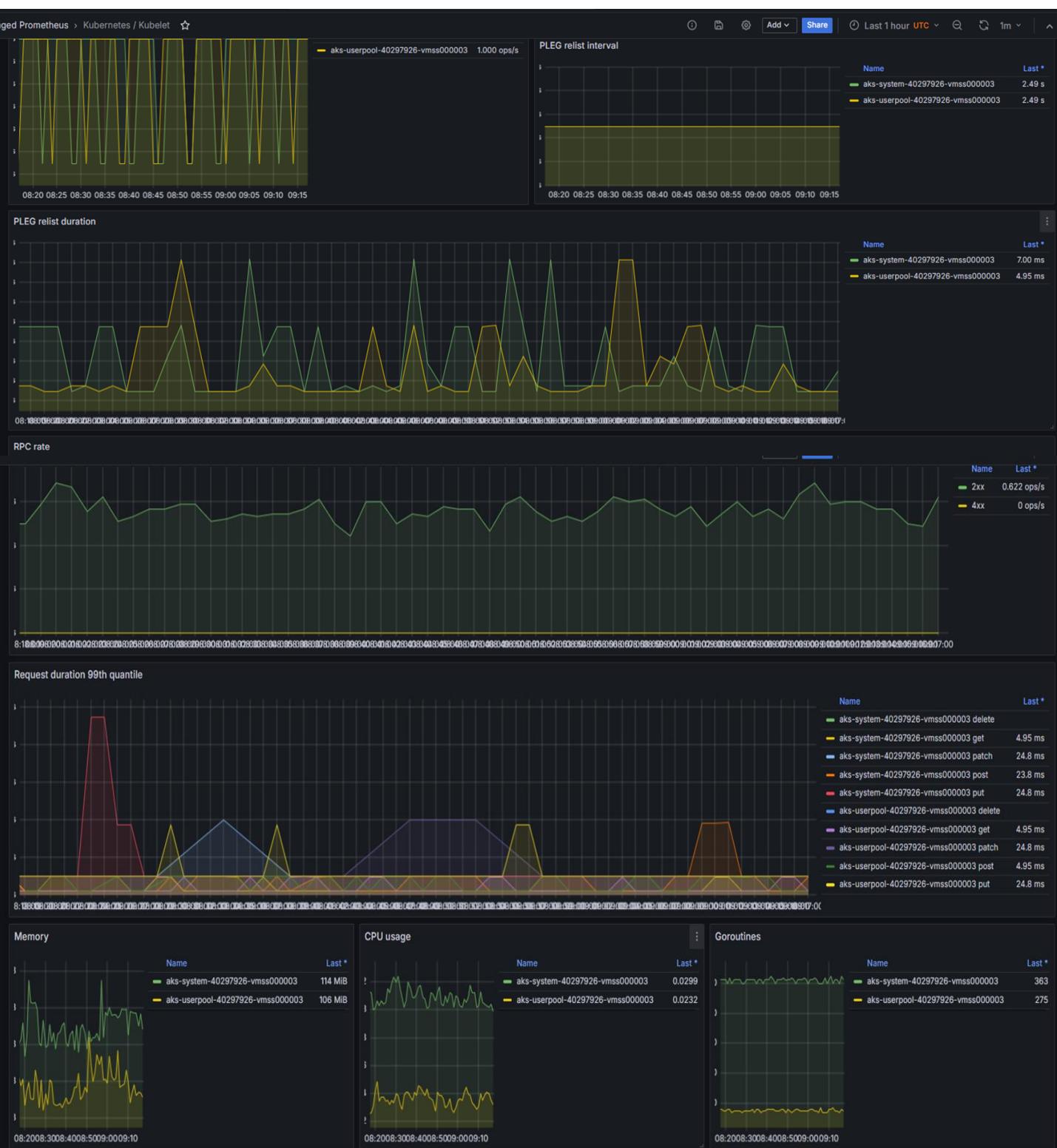
Azure Merged Prometheus> Kubernetes Dashboard



Azure Merged Prometheus> Kubernetes Dashboard



Azure Merged Prometheus> Kubernetes Dashboard



CPU Utilization Dashboard (for instance aks-system-40297926-vmss000003)



CPU Utilization Dashboard (for instance aks-userpool-40297926-vmss000003)

