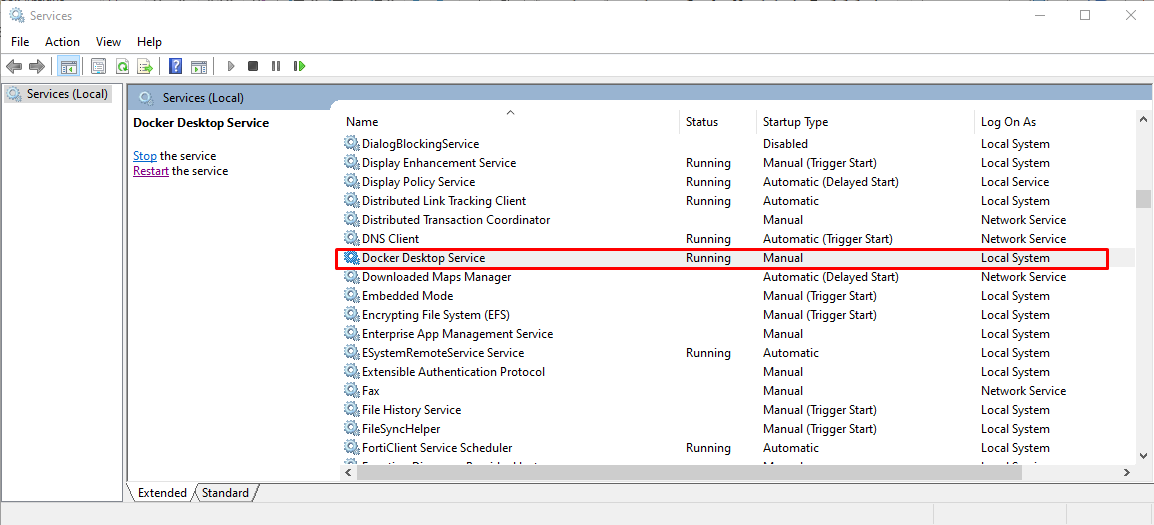
# How to setup Kubernetes for monitoring using Prometheus and grafana

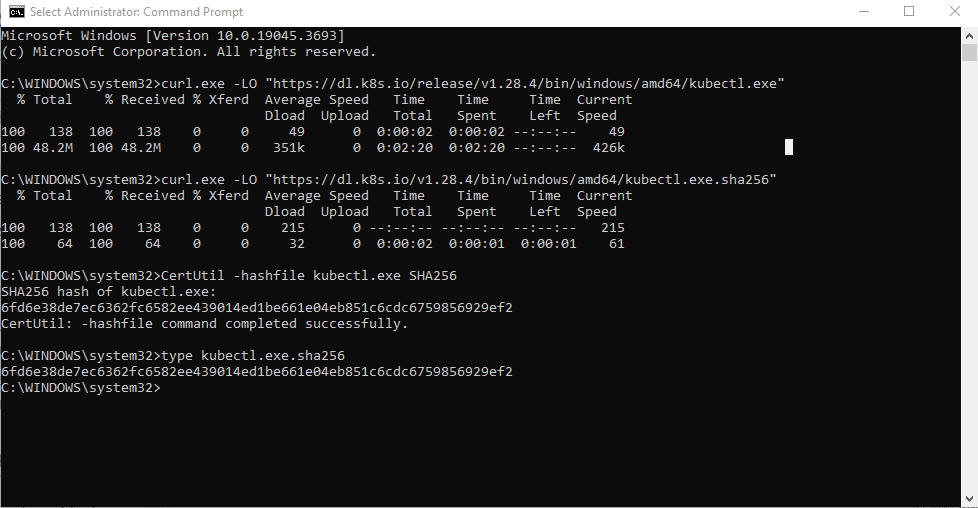
Step 1: Download docker and install it.

Step 2: Run docker under administrative privileges.

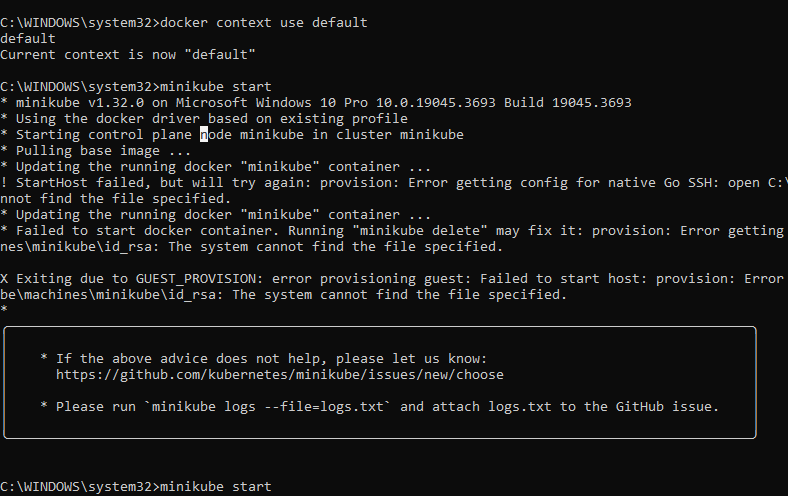
Step 3: Start Docker desktop service.

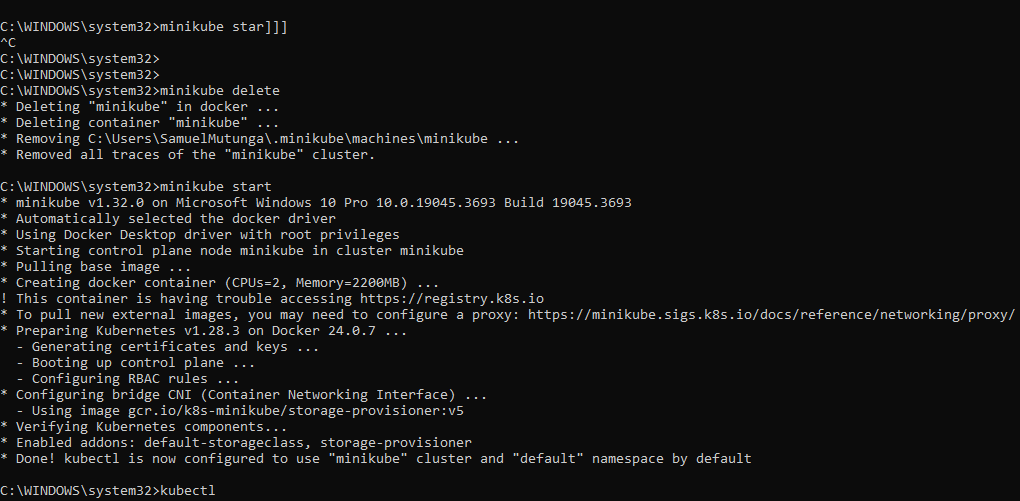


Step 4: Install kubectl



**Try to update your active context of docker which Minikube will pick up.**





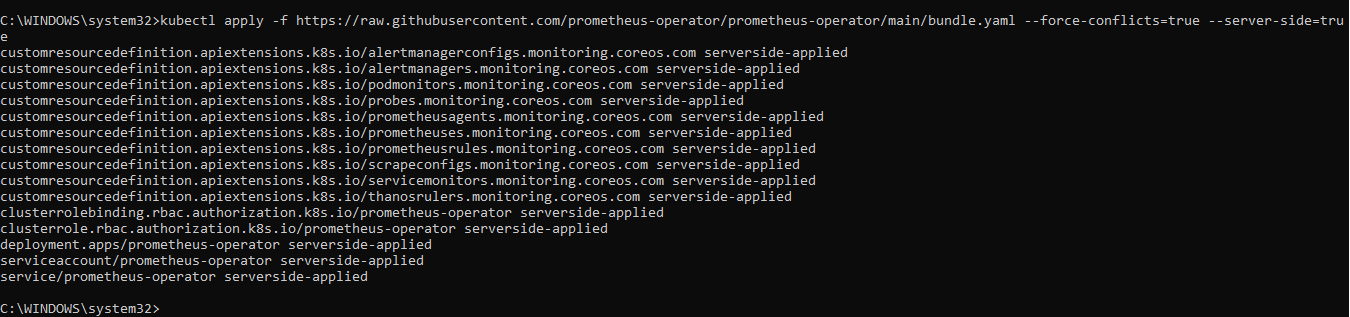
Creating a Kubernetes deployment – This is used to check the health of pods (A group of one or more container)

*# Run a test container image that includes a webserver*

kubectl create deployment hello-node --image=registry.k8s.io/e2e-test-images/agnhost:2.39 -- /agnhost netexec --http-port=8080

**Install Prometheus using kubectl**

kubectl apply -f https://raw.githubusercontent.com/prometheus-operator/prometheus-operator/main/bundle.yaml --force-conflicts=true --server-side=true



This procedure deploys Prometheus Operator in your cluster, then configure permission for the deployed operator to scrape targets in your cluster.

You can deploy by applying the bundle.yaml file from the Prometheus Operator GitHub repository:

A black screen with white text

Description automatically generated

Create a service account with RBAC through **Prometheus\_rbac.yaml**

A computer screen shot of white text

Description automatically generated

Deploy Prometheus instance through **Prometheus\_instance.yaml**

A black screen with white text

Description automatically generated

A screenshot of a computer program

Description automatically generated

Now, access the server by forwarding a local port to the Prometheus service:

A computer screen with white text

Description automatically generated

<http://localhost:9090>

A screenshot of a computer

Description automatically generated

**Configure Prometheus to monitor services and applications**

Create a ServiceMonitor CRD – This enables one to customize how you want Prometheus to operate in terms of scraping metrics.

This is done through **service\_monitor.yaml**

>> The code tells Prometheus to scrape services with the operated-prometheus: "true" label every thirty seconds, which you can access on the /metrics endpoint.

Apply it to your Kubernetes cluster by running the following command:

A screenshot of a computer screen

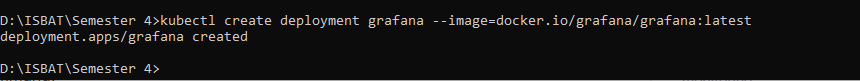
Description automatically generated

A screenshot of a computer

Description automatically generated

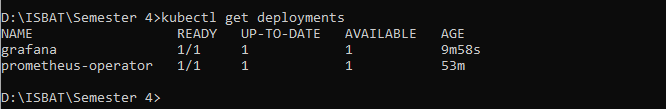
Deploying Grafana in your Kubernetes cluster

In your terminal, run the following command:



A black screen with white text

Description automatically generated

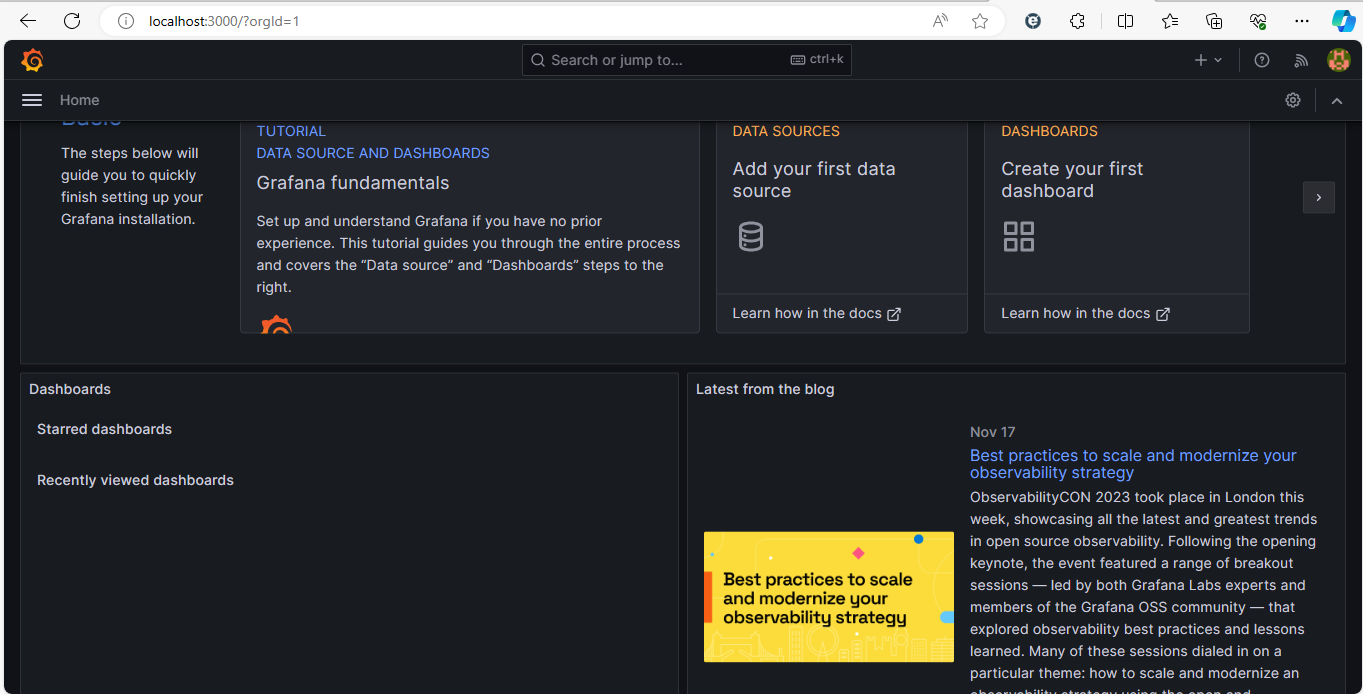


Next create a service for the grafana deployment

A black screen with white text

Description automatically generated

Forward port 3000 to the service



Click Data Sources and select Prometheus

>> You can’t use http://localhost:9090 as your HTTP URL because Grafana won’t have access to it. You must expose prometheus using a NodePort or LoadBalancer.

Create a file named expose\_prometheus.yaml and paste the following:

Run **kubectl apply -f expose\_prometheus.yaml**. Grafana will be able to pull the metrics from http://<node\_ip>:30900. To view the <node\_ip>, run kubectl get nodes -o wide.

A black screen with white text

Description automatically generated

A black background with white text

Description automatically generated

Enter http://<node\_ip>:30900 in the **URL** box, then click **Save & Test**:

http://192.168.49.2:30900

A screenshot of a computer

Description automatically generated

A screenshot of a computer

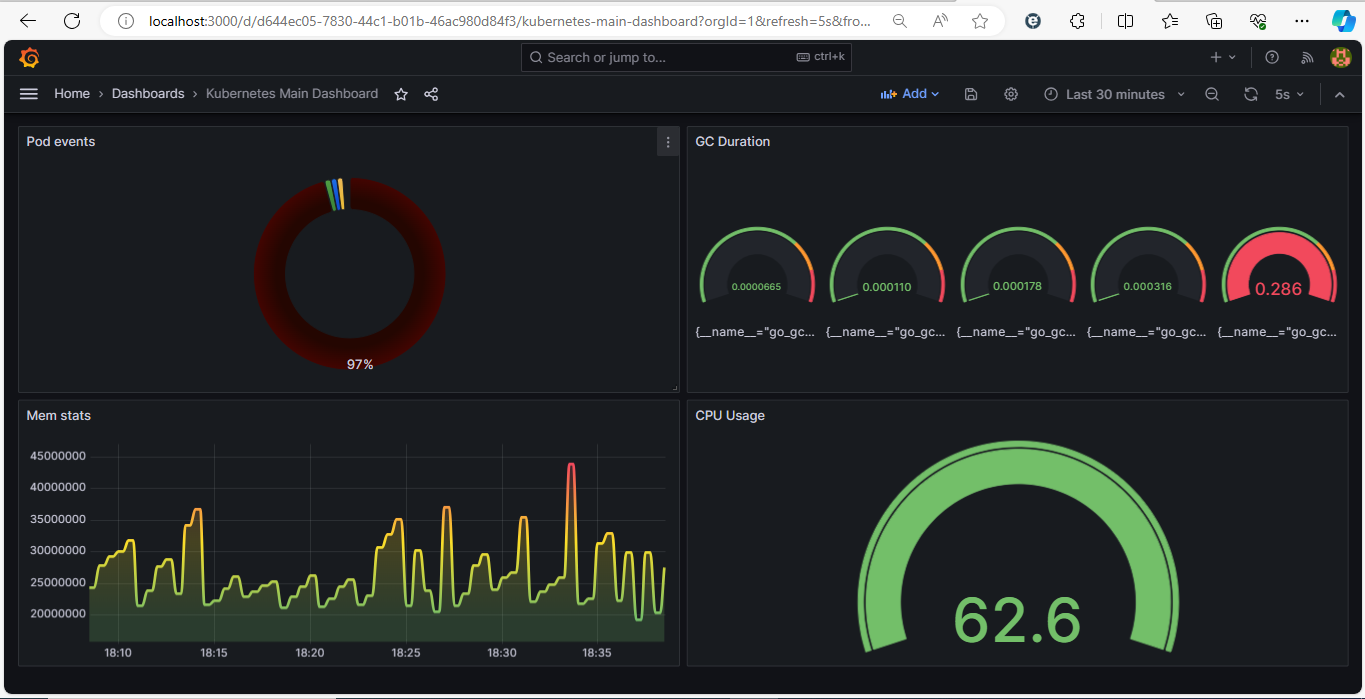
Description automatically generated

A screenshot of a computer

Description automatically generated

Creating a Grafana dashboard to monitor Kubernetes events.

Select **Prometheus** as your **Data source**, choose prometheus\_sd\_kubernetes\_events\_total for the metric and prometheus-prometheus-0 in the labels input or you can choose any metric and labels you want to monitor. Then, click **Run queries**:



1.How can Grafana be effectively integrated with Kubernetes for monitoring and observability in a DevOps environment?

***Run an agent such as Prometheus node exporter on each node.***

***The agent periodically scrapes the nodes current metrics and sends data to grafana through Prometheus.***

***Alternatively, when you have Metrics API, installed in the cluster, Kubernetes can collect this data itself.***

2.What are the best practices for setting up monitoring dashboards in Grafana to monitor Kubernetes clusters and applications?

***Understand recommended rules***

***Define aggregation rules***

***Troubleshoot your aggregated metric query***

***Adaptive metrics plugin***

***Adaptive metrics API.***

3.How does Grafana support metrics collection and visualization for Kubernetes-specific components such as nodes, pods, and services?

***Through prebuilt dashboards***

4.What are the recommended data sources and plugins in Grafana for monitoring and visualizing Kubernetes metrics?

***Data sources:***

***Prometheus, Graphite etc.***

***These should provide Time series data***

***Plugins***

***Grafana-worldmap-panel***

***Grafana-clock-panel***

***Grafana-clickhouse-datasource***

***Alertlist***

5.How can Grafana and Kubernetes be utilized together to achieve effective log monitoring and analysis in a DevOps workflow?

6.What are the key challenges and solutions for scaling Grafana in large Kubernetes clusters with a high number of applications?

***Ever -growing maintenance burden.***

***Risk of errors during day-to-day administration.***

***Unintentional downtime and regressions***

***Using same tools to achieve same goal.***

***Take a holistic and intentional approach***

***Be deliberate in strategy and document tools used, how to integrate, steps to be followed to achieve common tasks.***

***Acknowledge complexities of scale early on. Involve different stakeholders e.g., dev, opera and prj managers.***

***Focus on fundamentals and must-haves.***

7.How does Grafana's alerting system integrate with Kubernetes to provide timely notifications and alerts for critical issues?

***Create them in grafana UI and export them in a YAML file.***

8.What are the security considerations and best practices when using Grafana and Kubernetes in a DevOps environment?

***Security hardening enables one to apply additional security which might stop certain vulnerabilities from being exploited by a malicious attacker.***

9.How can Grafana and Kubernetes facilitate automated deployments and rolling updates of applications in a continuous delivery pipeline?

10.What are the recommended strategies for monitoring multi-cloud or hybrid cloud environments using Grafana and Kubernetes?