Setup Environment untuk Pengujian Mekanisme Kubernates Event-Driven Autoscalling (KEDA)

# Setup Apache Kafka.

* Download Apache Kafka 3.8. Jalankan Kafka dengan konfigurasi Kraft (bukan dengan konfigurasi Zookeeper) <https://kafka.apache.org/quickstart>.
* Konfigurasi untuk Monitoring kafka pada prometheous
  + - Set KAFKA\_JMX\_ADDITIONAL\_OPTS di kafka-server-start.bat, bukan di kafka-class-run.bat karena akan membuat JMX Address dibinding kembali ketika mengeluarkan perintah lain pada kafka/bin/windows/\* lainnya. (bisa gunakan file yang sudah disiapkan di repository github *RnD-XBRL-local/Source/apache-kafka/bin/\**)
    - Copy-kan file lib javaagent yang mengekspose nilai metric untuk promotheus. (bisa gunakan file yang sudah disiapkan di repository github *RnD-XBRL-local/Source/apache-kafka/libs/jmx\_prometheus\_javaagent-0.12.0.jar*)
* Konfigurasi komunikasi antara kafka client pada docker container dengan kafka broker pada host machine.
  + - Set property bernama advertised.listeners pada file kafka/config/kraft/server.properties, gunakan hostname host.docker.internal. (jika tidak disetting entah mengapa terhubung ke localhost). (bisa gunakan file yang sudah disiapkan di repository github *RnD-XBRL-local/Source/apache-kafka/config/kraft/server.properties*)

1. Setup Docker Desktop

* Download dan Install Docker Desktop
* Buka firewall untuk Aplikasi Docker Desktop, sehingga service yang ada di host dapat berkomunikasi dengan para container.

1. Setup Minikube

* Download minikube
* Jalankan minikube dengan driver docker.
  + minikube start -–driver=docker
* Lalu masuk ke dalam konteks docker daemon minikube dengan perintah (untuk perintah yang kedua dalam kasus menggunakan powershell, berbeda untuk cmd).
  + minikube docker-env
  + & minikube -p minikube docker-env --shell powershell | Invoke-Expression

!! Pastikan semua langkah selanjutnya dijalankan pada konteks docker daemon minikube, bukan docker daemon host/ desktop !!

1. Setup Aplikasi Consumer (keda-demo-consumer)
   * Clone <https://github.com/fauzi16/RnD-XBRL-local.git>
   * Build image aplikasi keda-consumer-demo dengan menggunakan maven plugin (pastikan sudah masuk docker daemon minikube dan konfigurasi docker host pada pom.xml sesuai)
     + mvn spring-boot:build-image
   * Setting pengaturan kubernate object yang diperuntukan bagi Aplikasi keda-consumer-demo
     + Kubectl apply -f https://github.com/kedacore/keda/releases/download/v2.10.1/keda-2.10.1.yaml
     + Kubectl apply -f kubernates-configuration/consumer-deployment.yaml
     + Kubectl apply -f kubernates-configuration/consumer-scale-object.yaml
     + Kubectl apply -f kubernates-configuration/consumer-service.yaml
2. Setup Monitoring Tools Kubernates Cluster.
   * Setup helm, download <https://helm.sh/docs/intro/install/>, install dan tambahkan pada PATH environment variable.
   * Setup Prometheus

Installasi Prometheus dengan helm package manager

* + - helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
    - helm repo update
    - helm install prometheus prometheus-community/prometheus

Expose service Prometheous

* + - kubectl expose service prometheus-server --type=NodePort --target-port=9090 --name=prometheus-server-ext
  + Setup Grafana

Installasi Grafana dengan helm package manager

* + - helm repo add grafana https://grafana.github.io/helm-charts
    - helm repo update
    - helm install grafana grafana/grafana

Cek password untuk login ke admin grafana (hasilnya di-decode dari base64)

* + - kubectl get secret --namespace default grafana -o jsonpath="{.data.admin-password}"

Expose service Grafana

* + - kubectl expose service grafana --type=NodePort --target-port=3000 --name=grafana-ext
  + Setup Dashboard kube-state-metrics Grafana
    1. Tonton video tutorial <https://youtu.be/EeiYpnBHnhY?si=beNgCeyxRo9kEJSy> pada bagian *import dashboard* dan *add new datasource*
    2. Tambahkan metric jumlah replica pod aplikasi keda-demo-consumer. PQL = kube\_deployment\_status\_replicas{namespace="default",deployment="keda-demo-consumer-deployment"}