Getting started with TEKTON

Prerequisites:-

1. Install minikube

```
  curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64

sudo install minikube-linux-amd64 /usr/local/bin/minikube

minikube start
```

2. Install Kubectl

```
 sudo curl -o kubectl https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.7/2022-06-29/bin/linux/amd64/kubectl

chmod +x ./kubectl

mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$PATH:$HOME/bin

echo 'export PATH=$PATH:$HOME/bin' >> ~/.bashrc
```

Verify if kubectl got installed

```
kubectl version --short –client
```

STEPS:-

Step 1: Create a Kubernetes cluster:

```
minikube start --kubernetes-version v1.24.4
```

Check cluster info:

```
kubectl cluster-info
```

Step 2: Install Tekton pipelines:

```
kubectl apply --filename \
https://storage.googleapis.com/tekton-releases/pipeline/latest/release.yaml
```

Step 3: Monitor the installation:

```
kubectl get pods --namespace tekton-pipelines –watch
```

Create and run basic task in TEKTON

A **Task**, represented in the API as an object of kind `Task`, defines a series of **Steps** that run sequentially to perform logic that the Task requires. Every Task runs as a pod on the Kubernetes cluster, with each step running in its own container.

1. To create a Task, open your favorite editor and create a file named `hello-world.yaml` with the following content:

```
apiVersion: tekton.dev/v1beta1
kind: Task
metadata:
  name: hello
spec:
  steps:
    - name: echo
      image: alpine
      script: |
        #!/bin/sh
        echo "Hello World"
```

2. Apply the changes to your cluster:

```
kubectl apply --filename hello-world.yaml
```

3. A `TaskRun` object instantiates and executes this Task. Create another file named `hello-world-run.yaml` with the following content:

```
apiVersion: tekton.dev/v1beta1
kind: TaskRun
metadata:
  name: hello-task-run
spec:
  taskRef:
    name: hello
```

4. Apply the changes to your cluster to launch the Task:

```
  kubectl apply --filename hello-world-run.yaml
```

5. Verify that everything worked correctly:

```
    kubectl get taskrun hello-task-run
```

6. Take a look at the logs:

```
   kubectl get taskrun hello-task-run
```

Getting started with TEKTON Pipelines

Prerequisites:

1. Install tkn ,the TEKTON CLI (Ubuntu or Debian)

```
    sudo apt update;

    sudo apt install -y gnupg

    sudo mkdir -p /etc/apt/keyrings/

    sudo gpg --no-default-keyring --keyring /etc/apt/keyrings/tektoncd.gpg --keyserver keyserver.ubuntu.com --recv-keys 3EFE0E0A2F2F60AA

   echo "deb [signed-by=/etc/apt/keyrings/tektoncd.gpg] http://ppa.launchpad.net/tektoncd/cli/ubuntu eoan main"|sudo tee /etc/apt/sources.list.d/tektoncd-ubuntu-cli.list

   sudo apt update && sudo apt install -y tektoncd-cli
```

STEPS:

- Create two Tasks.
- Create a Pipeline containing your Tasks.
- Use `PipelineRun` to instantiate and run the Pipeline containing your Tasks.

1. Create 1st task using above commands.

2.Create and run second task. Create a new file named `goodbye-world.yaml` and add the following content:

```
apiVersion: tekton.dev/v1beta1

kind: Task

metadata:

  name: goodbye

spec:

  params:

  - name: username

    type: string

  steps:

    - name: goodbye

      image: ubuntu

      script: |

        #!/bin/bash

        echo "Goodbye $(params.username)!"
```

NOTE: This Task takes one parameter, `username`. Whenever this Task is used a value for that parameter must be passed to the Task.

3. Apply the task file

```
kubectl apply --filename goodbye-world.yaml
```

Note: When a Task is part of a Pipeline, Tekton creates a `TaskRun` object for every task in the Pipeline.

4. Create and run a pipeline

A **Pipeline** defines an ordered series of Tasks arranged in a specific execution order as part of the CI/CD workflow.

In this section you are going to create your first Pipeline, that will include both the “Hello World!” and “Goodbye!” Tasks.

1. Create a new file named `hello-goodbye-pipeline.yaml` and add the following content:

```
apiVersion: tekton.dev/v1beta1

kind: Pipeline

metadata:

  name: hello-goodbye

spec:

  params:

  - name: username

    type: string

  tasks:

    - name: hello

      taskRef:
        name: hello
    - name: goodbye
      runAfter:
        - hello
      taskRef:
        name: goodbye
      params:
      - name: username
        value: $(params.username)
```

Note: The Pipeline defines the parameter `username`, which is then passed to the `goodbye` Task.

2. Apply the pipeline configuration to your cluster

```
kubectl apply --filename hello-goodbye-pipeline.yaml
```

3. A **PipelineRun**, represented in the API as an object of kind `PipelineRun`, sets the value for the parameters and executes a Pipeline. To create PipelineRun, create a new file named `hello-goodbye-pipeline-run.yaml` with the following:

```
 apiVersion: tekton.dev/v1beta1
kind: PipelineRun
metadata:
  name: hello-goodbye-run

spec:

  pipelineRef:

    name: hello-goodbye

  params:

  - name: username

    value: "Tekton"
```

Note: This sets the actual value for the `username` parameter: `"Tekton"`.

4. Start the Pipeline by applying the `PipelineRun` configuration to your cluster:

```
kubectl apply --filename hello-goodbye-pipeline-run.yaml
```

5. To see the logs of PipelineRun

```
tkn pipelinerun logs hello-goodbye-run -f -n default
```

OUTPUT

```
[hello : hello] Hello World!

[goodbye : goodbye] Goodbye Tekton!
```