Lab 9.1 Create a Pipeline to Build and Push the Container Image

Overview

For this exercise, we'll make use of a **git clone** task to fetch the repository necessary for image creation. Additionally, we'll utilize the **build and push** task, which employs Kaniko to construct the image and then push it to the repository. Detailed information about both of these tasks is provided below.

First Task (Git Clone)

The purpose of this task is to clone a Git repository using the GitHub URL and branch name.
The task will be directly applied via the Tekton Hub, and you can implement it using the provided command:

```
kubectl apply -f
https://raw.githubusercontent.com/tektoncd/catalog/main/task/git-clone/0.9/git-clone.yaml -n tekton-pipelines
```

Second Task (Build and Push Image)

To build and push an image to a repository, we can create a **task** with the manifest provided below and name it `build-push-docker-image-task.yaml`.

```
apiVersion: tekton.dev/v1beta1
kind: Task
metadata:
  name: build-push-docker-image-task
spec:
  workspaces:
    - name: output
  params:
    - name: app_repo
    - name: container_image
    - name: container_tag
    - name: secret-name
  volumes:
    - name: kaniko-secret
      secret:
        secretName: $(params.secret-name)  #name of the docker secret
        items:
          - key: .dockerconfigjson
            path: config.json
  steps:
    - name: build
      image: gcr.io/kaniko-project/executor:debug
      workingDir: "/workspace/output/"
      command: [/kaniko/executor]
      args:
        - --context=./
        - --destination=$(params.container_image):$(params.container_tag)
        - --force
      volumeMounts:
        - name: kaniko-secret
          mountPath: /kaniko/.docker/
```

In this task, we passed the `secret` name, `image` name, and `tag` name as parameters. The task will use the kaniko command to build and push the image to the repository.

To use this task, we need to pre-apply a secret that contains the Docker registry credentials and pass the secret as a volume. For more information on creating secrets, please refer to the Authentication section in the Installation and Configuration chapter.

After creating the secret, apply this task using the following command:

```
kubectl apply -f build-push-docker-image-task.yaml -n tekton-pipelines
```

Pipeline

Now that we have the tasks ready, let's create a `pipeline` with the following manifest and name it `build-and-push-image-pipeline.yaml`:

```
apiVersion: tekton.dev/v1beta1
kind: Pipeline
metadata:
  name: build-and-push-image-pipeline
spec:
  params:
    - name: gitrevision-tag
  workspaces:
    - name: shared-data
```

```
      description: |
        This workspace will receive the cloned git repo and be passed
        to the next Task for the repo's README.md file to be read.
  tasks:
    - name: fetch-repo
      taskRef:
        name: git-clone
      params:
        - name: url
          value: <your git-repo url>
        - name: revision
          value: $(params.gitrevision-tag)
      workspaces:
        - name: output
          workspace: shared-data
    - name: build-container-image
      runAfter: ["fetch-repo"]
      taskRef:
        name: build-push-docker-image-task
      params:
        - name: app_repo
          value: dir:///workspace/output/
        - name: container_image
          value:  # <image name, give a name of your choice for image>
        - name: container_tag
          value: # <image tag>
        - name: secret-name
          value: # <name of the secret that has docker credentials>
      workspaces:
        - name: output
          workspace: shared-data
```

Apply this pipeline with the following command:

```
kubectl apply -f build-and-push-image-pipeline.yaml -n tekton-pipelines
```

PipelineRun

After having created the tasks and the pipeline, the next step is to create a PipelineRun. Use the following manifest and save it as `build-and-push-image-run.yaml` to create a `PipelineRun`:

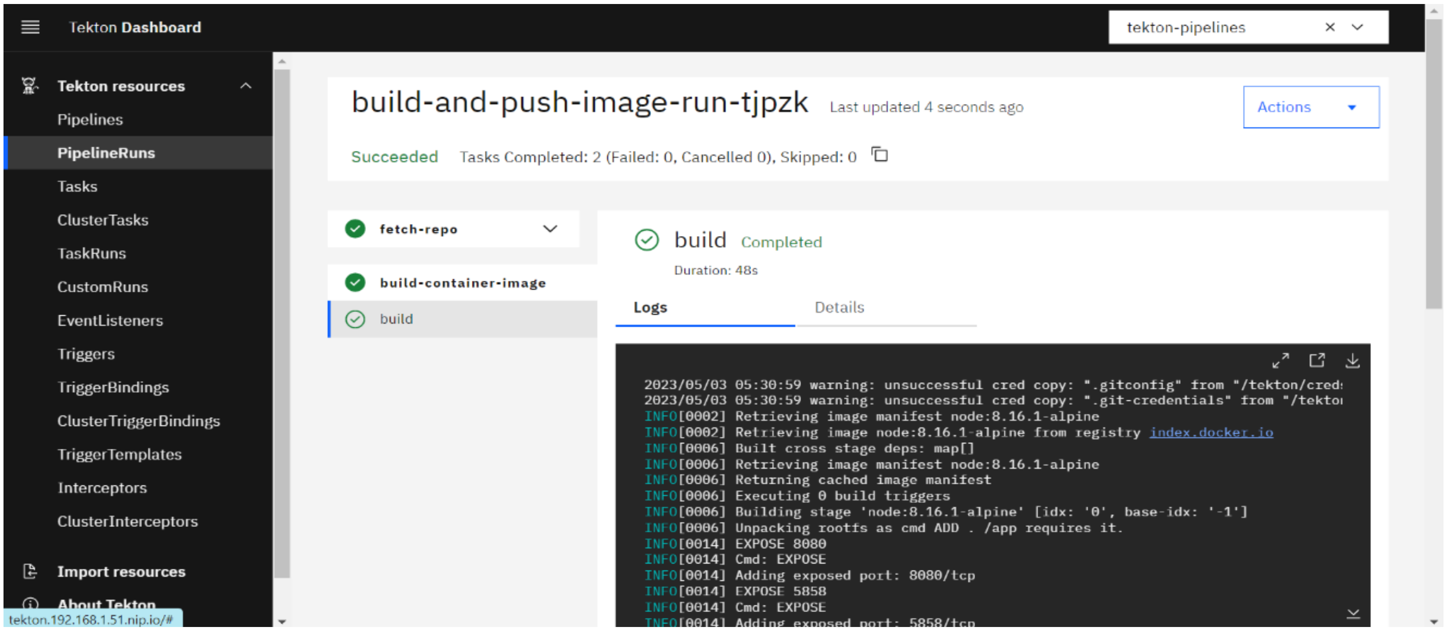
```
apiVersion: tekton.dev/v1beta1
kind: PipelineRun
metadata:
  generateName: build-and-push-image-run-
spec:
```

```
serviceAccountName: <service account name>
pipelineRef:
  name: build-and-push-image-pipeline
podTemplate:
  securityContext:
    fsGroup: 1001
params:
  - name: gitrevision-tag
    value: <branch name>
workspaces:
  - name: shared-data
    volumeClaimTemplate:
      spec:
        accessModes:
          - ReadWriteOnce
        resources:
          requests:
            storage: 500Mi
```

Apply this manifest with the following command:

```
kubectl create -f build-and-push-image-run.yaml -n tekton-pipelines
```

Once you have successfully completed your build and push pipeline, your screen should look like the image below.



build-and-push-image-run-tjpzk Last updated 4 seconds ago

Succeeded Tasks Completed: 2 (Failed: 0, Cancelled 0), Skipped: 0

build Completed

Duration: 48s

```
2023/05/03 05:30:59 warning: unsuccessful cred copy: ".gitconfig" from "/tekton/cred:
2023/05/03 05:30:59 warning: unsuccessful cred copy: ".git-credentials" from "/tekto
INFO[0002] Retrieving image manifest node:8.16.1-alpine
INFO[0002] Retrieving image node:8.16.1-alpine from registry index.docker.io
INFO[0006] Built cross stage deps: map[]
INFO[0006] Retrieving image manifest node:8.16.1-alpine
INFO[0006] Returning cached image manifest
INFO[0006] Executing 0 build triggers
INFO[0006] Building stage 'node:8.16.1-alpine' [idx: '0', base-idx: '-1']
INFO[0006] Unpacking rootfs as cmd ADD . /app requires it.
INFO[0014] EXPOSE 8080
INFO[0014] Cmd: EXPOSE
INFO[0014] Adding exposed port: 8080/tcp
INFO[0014] EXPOSE 5858
INFO[0014] Cmd: EXPOSE
INFO[0014] Adding exposed port: 5858/tcp
```

Completed Build and Push Pipeline