$ docker network create todoapp

$ docker run --name some-postgres -e POSTGRES\_PASSWORD=p@ssw0rd -e POSTGRES\_USER=root -e POSTGRES\_DB=todoapi -p 5432:5432 --network=todoapp -d postgres

$ docker run -p 8081:8081 --network=todoapp -e PG\_HOST=some-postgres -d vijaynvb/mytodoappanz:1.0

in browser -> http://localhost:8081/swagger-ui/index.html#/

Push image to docker hub:

# More examples of Codefresh YAML can be found at

# https://codefresh.io/docs/docs/yaml-examples/examples/

version: "1.0"

# Stages can help you organize your steps in stages

stages:

- "clone"

- "build"

- "test"

- "package"

- "push"

steps:

clone:

title: "Cloning repository"

type: "git-clone"

repo: "[yourrepo]/todoapi"

# CF\_BRANCH value is auto set when pipeline is triggered

# Learn more at codefresh.io/docs/docs/codefresh-yaml/variables/

git: "github-anz"

revision: "master"

stage: "clone"

build\_maven:

title: Maven Build

type: freestyle

stage: build

image: maven:3.8.1-jdk-11

working\_directory: '${{clone}}'

commands:

- mvn clean install

test\_maven:

title: Maven Test

type: freestyle

stage: test

image: maven:3.8.1-jdk-11

working\_directory: '${{clone}}'

commands:

- mvn test

copy\_to\_shared\_volume:

title: "list of shared volume"

type: "freestyle"

image: alpine:3.9

working\_directory: "${{clone}}"

commands:

- ls -l /codefresh/volume/

stage: "test"

build\_my\_app:

title: Building Docker Image

type: build

stage: package

image\_name: [youraccount]/todoapipganz

working\_directory: '${{clone}}'

tag: 'master'

dockerfile: Dockerfile

push\_to\_hub:

type: push

title: Push

description: Push to docker hub

candidate: ${{build\_my\_app}}

tag: latest

image\_name: [youraccount]/todoapipganz

registry: dockerhub

stage: push

Conditional steps execution Pipeline:

version: "1.0"

steps:

step1:

title: "Running if variable exists"

type: "freestyle"

image: "alpine:3.9"

commands:

- echo "Step 1 is running"

when:

condition:

all:

whenVarExists: 'includes("${{MY\_VAR}}", "{{MY\_VAR}}") == false'

step2:

title: "Running if variable does not exist"

type: "freestyle"

image: "alpine:3.9"

commands:

- echo "Step 2 is running"

when:

condition:

all:

whenVarIsMissing: 'includes("${{MY\_VAR}}", "{{MY\_VAR}}") == true'

Conditional execution with notification configurations:

version: '1.0'

stages:

- checkout

- build

- test

- notify

steps:

clone\_repository:

title: Cloning Repository

type: git-clone

repo: "vijaynvb/todoapi"

revision: "master"

stage: checkout

build\_maven:

title: Maven Build

type: freestyle

stage: build

image: maven:3.8.1-jdk-11

working\_directory: '${{clone\_repository}}'

commands:

- mvn clean install

when:

steps:

- name: clone\_repository

on:

- success

test\_maven:

title: Maven Test

type: freestyle

stage: test

image: maven:3.8.1-jdk-11

working\_directory: '${{clone\_repository}}'

commands:

- mvn test

when:

steps:

- name: build\_maven

on:

- success

notify\_failure:

title: Notify Build Failure

type: freestyle

stage: notify

image: alpine:3.12

commands:

- echo "Build failed. Notifying user..."

- echo "Sending failure notification..."

when:

steps:

- name: build\_maven

on:

- failure

Kubernetes cluster application deployment yaml k8s.yml

apiVersion: v1

kind: Namespace

metadata:

name: todoapp-prod

---

apiVersion: v1

kind: ConfigMap

metadata:

name: dbconfig

namespace: todoapp-prod

data:

MYSQL\_HOST: svctododb.todoapp.svc.cluster.local

MYSQL\_USER: root

MYSQL\_PASSWORD: P@ssword

MYSQL\_DATABASE: tododb

---

apiVersion: v1

kind: Service

metadata:

name: svctododb

namespace: todoapp-prod

spec:

type: ClusterIP

selector:

app: mysqldb

ports:

- port: 3306

targetPort: 3306

---

apiVersion: v1

kind: Service

metadata:

name: svctodoapp

namespace: todoapp-prod

spec:

type: LoadBalancer

selector:

app: todoapi

ports:

- port: 80

targetPort: 80

---

apiVersion: apps/v1

kind: Deployment

metadata:

name: mysqldb

namespace: todoapp-prod

spec:

selector:

matchLabels:

app: mysqldb

template:

metadata:

labels:

app: mysqldb

spec:

containers:

- name: mysqldb

image: mysql

ports:

- containerPort: 3306

env:

- name: MYSQL\_ROOT\_PASSWORD

valueFrom:

configMapKeyRef:

name: dbconfig

key: MYSQL\_PASSWORD

- name: MYSQL\_DATABASE

valueFrom:

configMapKeyRef:

name: dbconfig

key: MYSQL\_DATABASE

---

apiVersion: apps/v1

kind: Deployment

metadata:

name: todoapp

namespace: todoapp-prod

spec:

selector:

matchLabels:

app: todoapi

template:

metadata:

labels:

app: todoapi

spec:

containers:

- name: todoapi

image: vijaynvb/todoapimysql:1.0

ports:

- containerPort: 80

env:

- name: MYSQL\_HOST

valueFrom:

configMapKeyRef:

name: dbconfig

key: MYSQL\_HOST

- name: MYSQL\_USER

valueFrom:

configMapKeyRef:

name: dbconfig

key: MYSQL\_USER

- name: MYSQL\_PASSWORD

valueFrom:

configMapKeyRef:

name: dbconfig

key: MYSQL\_PASSWORD

- name: MYSQL\_DATABASE

valueFrom:

configMapKeyRef:

name: dbconfig

key: MYSQL\_DATABASE

Deploy codefresh pipeline:

# More examples of Codefresh YAML can be found at

# https://codefresh.io/docs/docs/yaml-examples/examples/

version: "1.0"

# Stages can help you organize your steps in stages

stages:

- "clone"

- "deploy"

steps:

clone:

title: "Cloning repository"

type: "git-clone"

repo: "vijaynvb/todoapi"

# CF\_BRANCH value is auto set when pipeline is triggered

# Learn more at codefresh.io/docs/docs/codefresh-yaml/variables/

revision: "master"

git: "github-anz"

stage: "clone"

deploy\_to\_k8:

title: deploying to cluster

type: deploy

kind: kubernetes

cluster: autopilot-cluster-1

namespace: todoapp-prod

file\_path: ./k8s.yml

stage: deploy

working\_directory: "${{clone}}"