Day - 1

Git:

$ git –version

Git project setup commands: anz\_gh - folder

git config --global user.email "uremaild"

git config --global user.name "urusername"

git add .

git commit -m intiatilcode

git push

1. SimpleWorkflow.yml

—----------------------------------------------------------------------------------------------------------------------------

name: simple workflow

on: push

jobs:

my-first-job:

name: my first job

runs-on: ubuntu-latest

steps:

- run: echo "hello world"

—----------------------------------------------------------------------------------------------------------------------------

1. Label Pull Requests

Label.yml on workflows

—----------------------------------------------------------------------------------------------------------------------------

name: Labeler

on: #[pull\_request\_target]

jobs:

label:

runs-on: ubuntu-latest

permissions:

contents: read

pull-requests: write

steps:

- uses: actions/labeler@v4

with:

repo-token: "${{ secrets.GITHUB\_TOKEN }}"

—----------------------------------------------------------------------------------------------------------------------------

Labeler.yml file in .github

—----------------------------------------------------------------------------------------------------------------------------

Documentation:

- changed-files:

– any-glob-to-any-file:

- Docs/\*

—----------------------------------------------------------------------------------------------------------------------------

5-Actions.yml

—----------------------------------------------------------------------------------------------------------------------------

name: 5-Actions

on: workflow\_dispatch

jobs:

my-first-job:

name: my first job

runs-on: ubuntu-latest

steps:

- uses: actions/checkout@v4

- uses: actions/setup-java@v4

with:

distribution: 'adopt'

java-version: '11'

- run: echo "hello world"

- run: mvn build

—----------------------------------------------------------------------------------------------------------------------------

Github context

—----------------------------------------------------------------------------------------------------------------------------

name: Contexts - github

on:

workflow\_dispatch:

jobs:

build:

runs-on: ubuntu-latest

steps:

- name: Display Event Information

run: |

echo "Event Name: ${{ github.event\_name }}"

echo "Ref: ${{ github.ref }}"

echo "SHA: ${{ github.sha }}"

echo "Actor: ${{ github.actor }}"

echo "Workflow: ${{ github.workflow }}"

echo "Run ID: ${{ github.run\_id }}"

echo "Run number: ${{ github.run\_number }}"

—----------------------------------------------------------------------------------------------------------------------------

expressions.yml

—----------------------------------------------------------------------------------------------------------------------------

name: 7-expresions

on: [workflow\_dispatch, push]

jobs:

build:

runs-on: ubuntu-latest

steps:

- if: github.event\_name == ‘push’

name: Conditional Step Using an Expression

run: echo "This is a push to the master branch."

- if: github.event\_name == ‘workflow\_dispatch’

name: Conditional Step Using an Expression

run: echo "This is a workflow dispatch to the master branch."

—----------------------------------------------------------------------------------------------------------------------------

variable.yml

—----------------------------------------------------------------------------------------------------------------------------

name: Using Variables

on:

workflow\_dispatch:

env:

WORKFLOW\_VAR: 'I am a workflow env var'

OVERWRITTEN: 'I will be overwritten'

UNDEFINED\_VAR\_WITH\_DEFAULT: ${{ vars.UNDEFINED\_VAR || 'default value' }}

jobs:

echo:

runs-on: ubuntu-latest

env:

JOB\_VAR: 'I am a job env var'

OVERWRITTEN: 'I have been overwritten at the job level'

steps:

- name: Print Env Variables

env:

STEP\_VAR: 'I am a step env var'

step\_var2: 'I am another step env var'

run: |

echo "Step env var: ${{ env.STEP\_VAR }}"

echo "Step env var 2: $step\_var2"

echo "Job env var: ${{ env.JOB\_VAR }}"

echo "Workflow env var: ${{ env.WORKFLOW\_VAR }}"

echo "Overwritten: ${{ env.OVERWRITTEN }}"

- name: Overwrite job variable

env:

OVERWRITTEN: 'I have been overwritten at the step level'

run: |

echo "Step env var: ${{ env.OVERWRITTEN }}"

—----------------------------------------------------------------------------------------------------------------------------

9-functions.yml

—----------------------------------------------------------------------------------------------------------------------------

name: 9-Using-Functions

on:

pull\_request:

workflow\_dispatch:

jobs:

echo1:

runs-on: ubuntu-latest

steps:

- name: Print PR title

run: echo "${{ github.event.pull\_request.title }}"

- name: Print PR labels

run: |

cat << EOF

${{ toJSON(github.event.pull\_request.labels) }}

EOF

- name: Bug step

if: ${{ !cancelled() && contains(github.event.pull\_request.title, 'fix') }}

run: echo "I am a bug fix"

- name: Sleep for 20 seconds

run: sleep 20

- name: Failing step

run: exit 1

- name: I will be skipped

if: ${{ success() }}

run: echo "I will print if previous steps succeed."

- name: I will execute

if: ${{ failure() }}

run: echo "I will print if any previous step fails."

- name: I will execute

if: ${{ !cancelled() }}

run: echo "I will always print, except when the workflow is cancelled."

- name: I will execute when cancelled

if: ${{ cancelled() }}

run: echo "I will print if the workflow has been cancelled."

—----------------------------------------------------------------------------------------------------------------------------

Executionflowcontrol.yml

—----------------------------------------------------------------------------------------------------------------------------

name: 10-Controlling the Execution Flow

on:

workflow\_dispatch:

inputs:

pass-unit-tests:

type: boolean

description: Whether unit tests will pass or not

default: true

jobs:

lint-build:

runs-on: ubuntu-latest

steps:

- name: Lint and build

run: echo "Linting and building project"

unit-tests:

runs-on: ubuntu-latest

steps:

- name: Running unit tests

run: echo "Running tests..."

- name: Failing tests

if: ${{ !inputs.pass-unit-tests }}

run: exit 1

deploy-nonprod:

runs-on: ubuntu-latest

needs:

- lint-build

- unit-tests

steps:

- name: Deploying to nonprod

run: echo "Deploying to nonprod..."

e2e-tests:

runs-on: ubuntu-latest

needs:

- deploy-nonprod

steps:

- name: Running E2E tests

run: echo "Running E2E tests"

load-tests:

runs-on: ubuntu-latest

needs:

- deploy-nonprod

steps:

- name: Running load tests

run: echo "Running load tests"

deploy-prod:

runs-on: ubuntu-latest

needs:

- e2e-tests

- load-tests

steps:

- name: Deploying to prod

run: echo "Deploying to prod..."

—----------------------------------------------------------------------------------------------------------------------------

11-outputs.yml

—----------------------------------------------------------------------------------------------------------------------------

name: 11-Outputs

on:

workflow\_dispatch

jobs:

build:

runs-on: ubuntu-latest

outputs:

result: ${{ steps.build.outputs.result }}

steps:

- name: Build

id: build

run: echo "result=success" >> "$GITHUB\_OUTPUT"

deploy:

runs-on: ubuntu-latest

needs: build

steps:

- name: Deployment

env:

BUILD\_STATUS: ${{ needs.build.outputs.result }}

run: echo "Build Status:” $BUILD\_STATUS

—----------------------------------------------------------------------------------------------------------------------------

12-cache.yml

—----------------------------------------------------------------------------------------------------------------------------

name: 15-01-Caching

on:

workflow\_dispatch

jobs:

install-dependencies:

runs-on: ubuntu-latest

steps:

- name: Checkout code

uses: actions/checkout@v4

- name: Setup Java

uses: actions/setup-java@v4

with:

distribution: 'adopt'

java-version: '11'

cache: 'maven'

- name: Install Dependencies

run: mvn dependency:go-offline

test:

runs-on: ubuntu-latest

needs: install-dependencies

steps:

- name: Checkout code

uses: actions/checkout@v4

- name: Setup Java

uses: actions/setup-java@v4

with:

distribution: 'adopt'

java-version: '11'

cache: 'maven'

- name: Test with Maven

run: mvn -B test

build:

runs-on: ubuntu-latest

needs: install-dependencies

steps:

- name: Checkout code

uses: actions/checkout@v4

- name: Setup Java

uses: actions/setup-java@v4

with:

distribution: 'adopt'

java-version: '11'

cache: 'maven'

- name: Build with Maven

run: mvn clean install -Dmaven.test.skip=true

—----------------------------------------------------------------------------------------------------------------------------

artifacts.yml

—----------------------------------------------------------------------------------------------------------------------------

name: 13-Artifacts

on:

workflow\_dispatch

jobs:

upload-artifact:

runs-on: ubuntu-latest

steps:

- name: Checkout code

uses: actions/checkout@v4

- name: Setup Java

uses: actions/setup-java@v4

with:

distribution: 'adopt'

java-version: '11'

cache: 'maven'

- name: Build with Maven

run: mvn clean install

- run: mkdir staging && cp target/\*.jar staging

- uses: actions/upload-artifact@v4

with:

name: Package

path: staging

download-artifact:

runs-on: ubuntu-latest

needs: upload-artifact

steps:

- name: Download web-app content

uses: actions/download-artifact@v4

with:

name: Package

- name: View content

run: ls -R

—----------------------------------------------------------------------------------------------------------------------------

14-releases.yml

—----------------------------------------------------------------------------------------------------------------------------

name: 14-Release

on:

workflow\_dispatch

jobs:

build:

runs-on: ubuntu-latest

steps:

- name: Checkout code

uses: actions/checkout@v4

- name: Setup Java

uses: actions/setup-java@v4

with:

distribution: 'adopt'

java-version: '11'

cache: 'maven'

- name: Build with Maven

run: mvn clean install -Dmaven.test.skip=true

- run: mkdir staging && cp target/\*.jar staging

- uses: actions/upload-artifact@v4

with:

name: Package

path: staging

release:

runs-on: ubuntu-latest

needs: build

steps:

- name: Download web-app content

uses: actions/download-artifact@v4

with:

name: Package

- name: View content

run: ls -R

- name: Archive site content

uses: thedoctor0/zip-release@master

with:

filename: app.zip

- name: Create GitHub release

id: create-new-release

uses: actions/create-release@v1

env:

GITHUB\_TOKEN: ${{ secrets.GITHUB\_TOKEN }}

with:

tag\_name: ${{ github.run\_number }}

release\_name: Release ${{ github.run\_number }}

- name: Upload release asset

uses: actions/upload-release-asset@v1

env:

GITHUB\_TOKEN: ${{ secrets.GITHUB\_TOKEN }}

with:

upload\_url: ${{ steps.create-new-release.outputs.upload\_url }}

asset\_path: ./app.zip

asset\_name: app-v${{ github.ref\_type }}.zip

asset\_content\_type: application/zip

—----------------------------------------------------------------------------------------------------------------------------

15-Matricies

—----------------------------------------------------------------------------------------------------------------------------

name: 15-Matrices

on:

workflow\_dispatch

jobs:

backwards-compatibility:

name: ${{ matrix.os }}-${{ matrix.java-version }}

runs-on: ${{ matrix.os }}

strategy:

matrix:

java-version: [11,14,17]

os:

- ubuntu-latest

- windows-latest

steps:

- name: Setup Java

uses: actions/setup-java@v4

with:

distribution: 'adopt'

java-version: ${{ matrix.java-version }}

- name: Perform some tests

run: echo "Running tests on OS ${{ matrix.os }} and Java ${{ matrix.java-version }}"

—----------------------------------------------------------------------------------------------------------------------------

16-environments.yml

—----------------------------------------------------------------------------------------------------------------------------

name: 16-Environments

on:

workflow\_dispatch:

inputs:

target-env:

type: environment

default: staging

jobs:

echo:

runs-on: ubuntu-latest

environment: ${{ inputs.target-env }}

env:

my-env-value: ${{ vars.MY\_ENV\_VALUE || 'default value' }}

steps:

- name: Echo vars

run: |

echo "Env variable: ${{ env.my-env-value }}"

—----------------------------------------------------------------------------------------------------------------------------

17-shellscripts.yml

—----------------------------------------------------------------------------------------------------------------------------

name: 17-Shell-Script

on:

workflow\_dispatch

jobs:

build:

runs-on: ubuntu-latest

steps:

- name: Checkout code

uses: actions/checkout@v4

- name: Make script executable and run

run: |

chmod +x ./init.sh

./init.sh

shell: bash

—----------------------------------------------------------------------------------------------------------------------------

18-APIAccess.yml

—----------------------------------------------------------------------------------------------------------------------------

name: 18-Fetch-Data-API

on:

workflow\_dispatch:

jobs:

fetch-todo:

runs-on: ubuntu-latest

steps:

- name: Fetch Todo Data

run: |

# Use curl to fetch the data from the API endpoint

# Integrate external systems within the Actions workflow

# prepare API request with the required inputs like token payload etc

todo\_data=$(curl -s https://jsonplaceholder.typicode.com/todos/1)

# Print the fetched data

echo "Fetched Todo Data: $todo\_data"

—----------------------------------------------------------------------------------------------------------------------------

19-CustomActions

.github/

└── actions/

└── composite-cache-deps/

└── action.yaml

—----------------------------------------------------------------------------------------------------------------------------

name: ANZ Setup Java

description: This action allows caching both Java and Maven Dependencies based on the pom.xml file

inputs:

java-version:

description: 'Java version to use'

default: '11'

required: true

working-dir:

description: The working directory of the application

default: .

required: false

runs:

using: 'composite'

steps:

# Step 1: Setup Java version

- name: Setup Java version ${{ inputs.java-version }}

uses: actions/setup-java@v4

with:

distribution: 'adopt'

java-version: ${{ inputs.java-version }}

cache: 'maven' # Cache Maven dependencies

# Step 2: Install Dependencies

- name: Install Dependencies

run: mvn dependency:go-offline

shell: bash

working-directory: ${{ inputs.working-dir }} # Set working directory for Maven command

—----------------------------------------------------------------------------------------------------------------------------

19-CustomActions-Composite.yml

—----------------------------------------------------------------------------------------------------------------------------

name: 19-CustomActions-Composite

run-name: 19-CustomActions-Composite

on:

workflow\_dispatch:

env:

working-directory: ./ # Default working directory

jobs:

build:

runs-on: ubuntu-latest

steps:

# Step 1: Checkout repository

- name: checkout repository

uses: actions/checkout@v4

# Step 2: Setup Java & Maven dependencies

- name: Setup Java & Maven dependencies

uses: ./.github/actions/composite-cache-deps

with:

java-version: 11

working-dir: ${{ env.working-directory }} # Use specified working directory

# Step 3: Run tests

- name: Run tests

run: mvn test

# Step 4: Build with Maven

- name: Build with Maven

run: mvn clean install

—----------------------------------------------------------------------------------------------------------------------------

Docker

[Creating a Docker container action - GitHub Docs](https://docs.github.com/en/actions/sharing-automations/creating-actions/creating-a-docker-container-action)

—----------------------------------------------------------------------------------------------------------------------------

20-Reuseable workflow

20-ReusableWorkflows-Defnition.yml

—----------------------------------------------------------------------------------------------------------------------------

name: 20-Reusable-Workflows - Reusable Definition

on:

workflow\_call:

inputs:

target-directory:

type: string

required: true

outputs:

build-status:

description: The status of the build process

value: ${{ jobs.deploy.outputs.build-status }}

url:

description: The url of the deployed version

value: ${{ jobs.deploy.outputs.url }}

jobs:

deploy:

runs-on: ubuntu-latest

outputs:

build-status: ${{ steps.build.outputs.build-status }}

url: ${{ steps.deploy.outputs.url }}

steps:

- name: Checkout repo

uses: actions/checkout@v4

- name: Build

id: build

run: |

echo "Building using directory ${{ inputs.target-directory }}"

echo "build-status=success" >> "$GITHUB\_OUTPUT"

- name: Deploy

id: deploy

run: |

echo "Deploying build artifacts"

echo "url=https://www.google.com" >> "$GITHUB\_OUTPUT"

—----------------------------------------------------------------------------------------------------------------------------

20--ReusableWorkflows-Caller.yml

—----------------------------------------------------------------------------------------------------------------------------

name: 20-Reusable Workflows

on:

workflow\_dispatch:

jobs:

deploy:

uses: ./.github/workflows/21-01-reusable-workflows.yaml

with:

target-directory: dummmy-dir

print-outputs:

runs-on: ubuntu-latest

needs: deploy

steps:

- name: Print outputs

run: |

echo "Build status: ${{ needs.deploy.outputs.build-status }}"

echo "URL: ${{ needs.deploy.outputs.url }}"

—----------------------------------------------------------------------------------------------------------------------------

21-Docker.yml

—----------------------------------------------------------------------------------------------------------------------------

name: 21-Docker

on:

workflow\_dispatch:

jobs:

docker-build:

runs-on: ubuntu-latest

steps:

- name: Checkout repository

uses: actions/checkout@v2

- name: Set up JDK

uses: actions/setup-java@v2

with:

java-version: '11'

distribution: 'adopt'

cache: maven

- name: Build with Maven

run: mvn clean install

- name: Build Docker image

run: docker build -t username/app .

- name: Log in to Docker Hub

run: echo "${{ secrets.DOCKER\_PASSWORD }}" | docker login -u "${{ secrets.DOCKER\_USERNAME }}" --password-stdin

- name: Push image to Docker Hub

run: docker push username/app

—----------------------------------------------------------------------------------------------------------------------------

21-CI.yml

—----------------------------------------------------------------------------------------------------------------------------

name: 21-CI Pipeline

on:

workflow\_dispatch:

inputs:

java-version:

type: choice

options:

- 11

- 14

- 17

jobs:

intialize:

runs-on: ubuntu-latest

outputs:

java-version: ${{ steps.set-java-version.outputs.java-version }}

steps:

- name: Intialize project

id: set-java-version

run: echo "::set-output name=java-version::${{ github.event.inputs.java-version }}"

build:

runs-on: ubuntu-latest

needs: intialize

steps:

- name: Checkout repository

uses: actions/checkout@v4

- name: Setup Java

uses: actions/setup-java@v4

with:

distribution: 'adopt'

java-version: ${{ needs.intialize.outputs.java-version }}

cache: 'maven'

- name: Build with Maven

run: mvn clean install -Dmaven.test.skip=true

- name: Upload jar to folder

uses: actions/upload-artifact@v4

with:

name: my-artifact

path: target/

test:

runs-on: ubuntu-latest

needs: intialize

steps:

- name: Checkout repository

uses: actions/checkout@v4

- name: Setup Java

uses: actions/setup-java@v4

with:

distribution: 'adopt'

java-version: ${{ needs.intialize.outputs.java-version }}

cache: 'maven'

- name: Test with Maven

run: mvn -B test

release:

runs-on: ubuntu-latest

needs: build

steps:

- name: Checkout repository

uses: actions/checkout@v4

- name: Download web-app content

uses: actions/download-artifact@v4

with:

name: my-artifact

path: target/

- name: Archive site content

uses: thedoctor0/zip-release@master

with:

filename: site.zip

- name: Create GitHub release

id: create-new-release

uses: actions/create-release@v1

env:

GITHUB\_TOKEN: ${{ secrets.GITHUB\_TOKEN }}

with:

tag\_name: ${{ github.run\_number }}

release\_name: Release ${{ github.run\_number }}

- name: Upload release asset

uses: actions/upload-release-asset@v1

env:

GITHUB\_TOKEN: ${{ secrets.GITHUB\_TOKEN }}

with:

upload\_url: ${{ steps.create-new-release.outputs.upload\_url }}

asset\_path: ./site.zip

asset\_name: site-v${{ github.run\_number }}.zip

asset\_content\_type: application/zip

image:

runs-on: ubuntu-latest

needs: build

steps:

- name: Checkout repository

uses: actions/checkout@v4

- name: Download web-app content

uses: actions/download-artifact@v4

with:

name: my-artifact

path: target/

- name: Build Docker image

run: docker build -t username/todoapp .

- name: Log in to Docker Hub

run: echo "${{ secrets.DOCKER\_PASSWORD }}" | docker login -u "${{ secrets.DOCKER\_USERNAME }}" --password-stdin

- name: Push image to Docker Hub

run: docker push username/todoapp

—----------------------------------------------------------------------------------------------------------------------------

22-Runners.yml

—----------------------------------------------------------------------------------------------------------------------------

name: 05-01-Workflow Runners

on: workflow\_dispatch

jobs:

ubuntu-echo:

runs-on: ubuntu-latest

steps:

- name: Show OS

run: |

echo "This job is running on an Ubuntu runner."

echo "Runner OS: $RUNNER\_OS"

windows-echo:

runs-on: windows-latest

steps:

- name: Show OS

shell: bash

run: |

echo "This job is running on a Windows runner."

echo "Runner OS: $RUNNER\_OS"

mac-echo:

runs-on: macos-latest

steps:

- name: Show OS

run: |

echo "This job is running on a MacOS runner."

echo "Runner OS: $RUNNER\_OS"

self-echo:

runs-on: self-hosted

steps:

- name: Show OS

run: |

echo "This job is running on a Self-Hosted runner."

echo "Runner OS: $RUNNER\_OS"

—----------------------------------------------------------------------------------------------------------------------------

Execute in powershell:

set-executionpolicy remotesigned

—----------------------------------------------------------------------------------------------------------------------------

provider "google" {

project = var.project\_id

region = var.region

}

resource "google\_container\_cluster" "primary" {

name = var.cluster\_name

location = var.location

initial\_node\_count = 3

node\_config {

machine\_type = "e2-medium"

oauth\_scopes = [

"https://www.googleapis.com/auth/cloud-platform",

]

service\_account = var.service\_account

}

deletion\_protection = false

}

variable "project\_id" {

description = "The ID of the GCP project"

}

variable "region" {

description = "The GCP region"

}

variable "location" {

description = "The GCP zone or location"

}

variable "cluster\_name" {

description = "The name of the GKE cluster"

}

variable "service\_account" {

description = "The service account for the GKE cluster"

}

—----------------------------------------------------------------------------------------------------------------------------

CD-Google Cluster.yml

—----------------------------------------------------------------------------------------------------------------------------

name: Continuous Deployment

on:

workflow\_run:

workflows: ["Continuous Integration"]

types:

- completed

workflow\_dispatch:

env:

CREDENTIALS: ${{ secrets.GCP\_SA\_KEY }}

PROJECT\_ID: ${{ secrets.GCP\_PROJECT\_ID }}

GKE\_CLUSTER: gke-todoapp-cluster # cluster name

GKE\_REGION: us-east1

GKE\_LOCATION: us-east1-b # cluster location

SERVICE\_ACCOUNT: ${{ secrets.GCP\_SERVICE\_ACCOUNT }}

TFSTATE\_CACHE\_KEY: terraform-state-${{ github.sha }}

jobs:

create-cluster:

if: ${{ github.event.workflow\_run.conclusion == 'success' }}

runs-on: ubuntu-latest

steps:

- name: Checkout repository

uses: actions/checkout@v4

- name: Setup Terraform

uses: hashicorp/setup-terraform@v1

- name: Terraform Init

run: terraform init

env:

GOOGLE\_CREDENTIALS: ${{ env.CREDENTIALS }}

- name: Terraform Plan

run: terraform plan -input=false

env:

GOOGLE\_CREDENTIALS: ${{ env.CREDENTIALS }}

TF\_VAR\_project\_id: ${{ env.PROJECT\_ID }}

TF\_VAR\_region: ${{ env.GKE\_REGION }}

TF\_VAR\_location: ${{ env.GKE\_LOCATION }}

TF\_VAR\_cluster\_name: ${{ env.GKE\_CLUSTER }}

TF\_VAR\_service\_account: ${{ env.SERVICE\_ACCOUNT }}

- name: Terraform Apply

id: terraform-apply

run: terraform apply -auto-approve -input=false

env:

GOOGLE\_CREDENTIALS: ${{ env.CREDENTIALS }}

TF\_VAR\_project\_id: ${{ env.PROJECT\_ID }}

TF\_VAR\_region: ${{ env.GKE\_REGION }}

TF\_VAR\_location: ${{ env.GKE\_LOCATION }}

TF\_VAR\_cluster\_name: ${{ env.GKE\_CLUSTER }}

TF\_VAR\_service\_account: ${{ env.SERVICE\_ACCOUNT }}

- name: Cache Terraform State

uses: actions/cache@v3

with:

path: |

.terraform/

terraform.tfstate

terraform.tfstate.backup

key: ${{ env.TFSTATE\_CACHE\_KEY }}

deploy-to-cluster:

runs-on: ubuntu-latest

needs: create-cluster

outputs:

baseurl: ${{ steps.extract-url.outputs.baseurl }}

steps:

- name: Checkout repository

uses: actions/checkout@v4

- name: Install Google Cloud SDK

uses: 'google-github-actions/auth@v2'

with:

credentials\_json: ${{ env.CREDENTIALS }}

- name: Set up Cloud SDK

uses: 'google-github-actions/setup-gcloud@v2'

- name: Configure kubectl to use gke-gcloud-auth-plugin

run: gcloud components install kubectl

- name: Get GKE credentials

run: gcloud container clusters get-credentials ${{ env.GKE\_CLUSTER }} --zone ${{ env.GKE\_LOCATION }} --project ${{ secrets.GCP\_PROJECT\_ID }}

- name: Deploy application to GKE

run: kubectl apply -f deployment.yaml

- name: Wait for the service to be ready

run: |

echo "Waiting for 60 seconds..."

sleep 60

- name: Extract baseurl from GKE service

id: extract-url

run: |

EXTERNAL\_IP=$(kubectl get svc svctodoh2api -o jsonpath='{.status.loadBalancer.ingress[0].ip}')

PORT=$(kubectl get svc svctodoh2api -o jsonpath='{.spec.ports[0].port}')

echo "EXTERNAL\_IP is: $EXTERNAL\_IP"

echo "PORT is: $PORT"

echo "baseurl=http://$EXTERNAL\_IP:$PORT" >> $GITHUB\_OUTPUT

run-postman-tests:

runs-on: ubuntu-latest

needs: deploy-to-cluster

steps:

- name: Checkout repository

uses: actions/checkout@v4

- name: Install Newman

run: npm install -g newman

- name: Run Postman Collection

env:

BASEURL: ${{ needs.deploy-to-cluster.outputs.baseurl }}

run: |

newman run postman\_collection.json --env-var "base\_url=${{ env.BASEURL }}"

destroy-cluster:

runs-on: ubuntu-latest

needs: [deploy-to-cluster, run-postman-tests]

steps:

- name: Checkout repository

uses: actions/checkout@v4

- name: Restore Terraform State Cache

uses: actions/cache@v3

with:

path: |

.terraform/

terraform.tfstate

terraform.tfstate.backup

key: ${{ env.TFSTATE\_CACHE\_KEY }}

- name: Setup Terraform

uses: hashicorp/setup-terraform@v1

- name: Terraform Init

run: terraform init

env:

GOOGLE\_CREDENTIALS: ${{ env.CREDENTIALS }}

- name: Terraform Destroy

run: terraform destroy -auto-approve -input=false

env:

GOOGLE\_CREDENTIALS: ${{ env.CREDENTIALS }}

TF\_VAR\_project\_id: ${{ env.PROJECT\_ID }}

TF\_VAR\_region: ${{ env.GKE\_REGION }}

TF\_VAR\_location: ${{ env.GKE\_LOCATION }}

TF\_VAR\_cluster\_name: ${{ env.GKE\_CLUSTER }}

TF\_VAR\_service\_account: ${{ env.SERVICE\_ACCOUNT }}

—----------------------------------------------------------------------------------------------------------------------------

1. Install Github CLI ( gh ) - https://cli.github.com/

2. Install the GitHub Actions Importer CLI extension - gh extension install github/gh-actions-importer

3. Installing Bamboo Docker and PostgreSQL

-> docker network create bamboo

-> docker run --name bamboo-server --rm --detach --network bamboo --publish 8085:8085 atlassian/bamboo-server

-> docker run --name postgres-db --rm --detach --network bamboo --env POSTGRES\_PASSWORD=mysecretpassword --env POSTGRES\_USER=bamboo --env POSTGRES\_DB=bamboo --publish 5432:5432 postgres

4. Access Bamboo 'http://localhost:8085'

-> it will be redirected to Welcome to Bamboo Data Center

-> Add the License key where you generated from Atlassin or click the link ( Generate a Bamboo Data Center license ), Select Continue

-> Redirected to Configure Instance and continue

-> Configure database -> PostgreSQL -> Continue

-> Configure how Bamboo will connect to your database -> database url 'jdbc:postgresql://postgres-db:5432/bamboo' , provide Bamboo username password provide docker command

-> Import Data -> Create a new Bamboo home -> Continue

-> Create Admin -> Finish

4. Access Bamboo 'http://localhost:8085'

-> it will be redirected to Welcome to Bamboo Data Center

-> Add the License key where you generated from Atlassin or click the link ( Generate a Bamboo Data Center license ), Select Continue

-> Redirected to Configure Instance and continue

-> Configure database -> PostgreSQL -> Continue

-> Configure how Bamboo will connect to your database -> database url 'jdbc:postgresql://postgres-db:5432/bamboo' , provide Bamboo username password provide docker command

-> Import Data -> Create a new Bamboo home -> Continue

-> Create Admin -> Finish

5. Configure Agent -> Bamboo Settings -> Add Local Agent -> name the agent -> this will create local agent

6. Simple Maven project pipeline to run

7. Create Bamboo Personal Access Token:

Log in to your Bamboo instance.

Go to Profile

Access the "Personal Access Tokens" section.

Create token with the required permissions

Copy the token and save it securely.

8. Create GitHub Personal Access Token

-> Navigate to your GitHub account settings.

-> Go to the "Developer settings" or "Developer options."

-> Select "Personal access tokens."

-> Generate a new token with the "workflow" scope.

-> Copy the token and save it securely.

9. Configure GitHub Actions Importer -> gh actions-importer configure

V Which CI providers are you configuring?: Bamboo

Enter the following values (leave empty to omit):

V Personal access token for GitHub: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

V Base url of the GitHub instance: https://github.com

V Personal access token for Bamboo: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

V Base url of the Bamboo instance: http://192.168.90.174:8085

Environment variables successfully updated.

10. Perform an audit of Bamboo -> gh actions-importer audit bamboo --output-dir tmp/audit

11. Forecasting usage -> gh actions-importer forecast bamboo --output-dir tmp/forecast\_reports

12. Perform a dry-run migration of a Bamboo pipeline -> gh actions-importer dry-run bamboo build --plan-slug :cip-cip --output-dir tmp/dry-run

13. Perform a production migration of a Bamboo pipeline -> gh actions-importer migrate bamboo build --plan-slug :cip-cip --target-url :target\_url --output-dir tmp/migrate