**Lab 5: Reusable CI/CD Pipeline with GitHub Action**

**🔧 Objective:**

**Build a secure and reusable CI/CD pipeline** with:

* Automated linting
* SBOM generation
* SAST scanning (Bandit)
* Docker build & push (to DockerHub or GHCR)
* Optional Helm lint & deployment job
* Add the .github/workflows/ci.yml file in the cookicuter template :

(venv) root@ip-172-31-87-197:~/cookiecutter-golden-path-test/{{cookiecutter.project\_slug}}# cat .github/workflows/ci.yml

name: {{ cookiecutter.ci\_pipeline\_name }}

on:

workflow\_dispatch:

push:

branches: [ "{{ cookiecutter.default\_branch }}" ]

pull\_request:

branches: [ "{{ cookiecutter.default\_branch }}" ]

jobs:

lint-test-scan:

runs-on: ubuntu-latest

steps:

- name: Checkout Code

uses: actions/checkout@v4

- name: Set up Python

uses: actions/setup-python@v5

with:

python-version: '{{ cookiecutter.python\_version }}'

- name: Install Dependencies

run: |

python -m pip install --upgrade pip

pip install -r requirements.txt

pip install bandit cyclonedx-bom

- name: Run Unit Tests

run: |

pytest tests/

{% if cookiecutter.enable\_security == "yes" %}

- name: Run Bandit Security Scan

run: bandit -r app -f json -o bandit-report.json || true

- name: Generate SBOM

run: cyclonedx-py requirements -i requirements.txt -o sbom.json

{% endif %}

docker-build-push:

needs: lint-test-scan

runs-on: ubuntu-latest

steps:

- name: Checkout Code

uses: actions/checkout@v4

- name: Set up Docker Buildx

uses: docker/setup-buildx-action@v3

- name: Login to DockerHub

uses: docker/login-action@v3

with:

username: ${{ '{{ secrets.DOCKERHUB\_USERNAME }}' }}

password: ${{ '{{ secrets.DOCKERHUB\_TOKEN }}' }}

- name: Build & Push Docker Image

uses: docker/build-push-action@v5

with:

context: .

push: true

tags: ${{ '{{ secrets.DOCKERHUB\_USERNAME }}' }}/{{ cookiecutter.docker\_image\_name }}:{{ cookiecutter.docker\_image\_tag }}

* **Update cookicutter.json file :**

(venv) root@ip-172-31-87-197:~/cookiecutter-golden-path-test# cat cookiecutter.json

{

"ci\_pipeline\_name": "CI Pipeline",

"project\_name": "My Microservice",

"project\_slug": "my\_microsvc",

"author\_name": "Raman",

"description": "A sample microservice scaffolded using Cookiecutter.",

"port": "8080",

"python\_version": "3.11",

"docker\_image\_name": "raman-micro",

"docker\_image\_tag": "latest",

"enable\_security": "yes",

"default\_branch": "main"

}

* **Run the cookicuter template to create raman-micro service with cicd .**

**cookiecutter cookiecutter-golden-path-test/**

**📁 Project Structure Recap (required)**

* **cookiecutter ./cookiecutter-golden-path/**

Ensure your scaffolded project (e.g., raman-micro) looks like this:

raman-micro/

├── .github/

│ └── workflows/

│ └── ci.yaml

├── app/

│ └── main.py

├── Dockerfile

├── requirements.txt

├── tests/

│ └── test\_app.py

├── prometheus\_exporter.py

├── helm/

│ └── Chart.yaml

Note : In general ; cicd ci.yml template resides inside cookiecutter template and renders as per the template configuration.

**🔐 Step 2: Add GitHub Secrets**

Go to your GitHub repo → **Settings → repo Secrets → Actions** and add:

* DOCKERHUB\_USERNAME
* DOCKERHUB\_TOKEN (generate from DockerHub > Account Settings > Security > Access Tokens)

**✅ Step 3: Commit & Push to github as a separate repository:**

cd raman-micro/

403 ls

404 ls -a

405 history

406 git init

407 git commit -m "first commit"

408 git commit -m .

409 git add .

410 git commit -m "first commit"

411 git branch -M main

412 git remote add origin https://github.com/ramannkhanna2/raman-micro.git

413 git push -u origin main

**🔎 Step 4: Watch It Run**

Go to **GitHub → Actions**, you’ll see the pipeline executing:

* 🧪 Unit tests pass
* 🛡️ Bandit scan runs
* 📦 SBOM is generated
* 🐳 Docker image gets pushed to DockerHub

**✅ ✅ ✅ Checkpoints Completion Checklist Golden Path**

**1. Project Bootstrapped from Cookiecutter Template**

* Generated project structure via cookiecutter-golden-path
* Ensured proper layout with:
  + app/main.py
  + requirements.txt
  + Dockerfile
  + tests/test\_app.py
  + prometheus\_exporter.py

✅ *Golden Path project structure created successfully*

**2. Application Verified Locally**

* Ran app via python app/main.py
* Verified / endpoint with:

bash

CopyEdit

curl http://localhost:5000/

* Verified Prometheus /metrics on port 8080

✅ *App works as expected in local development*

**3. Docker Containerization**

* Built Docker image from Dockerfile
* Included Prometheus exporter
* Verified container runs with:

bash

CopyEdit

docker run -p 5000:5000 -p 8080:8080 <your-image>

✅ *Containerized version of app runs successfully*

**4. CI/CD with GitHub Actions**

* Ran multi-step CI pipeline:
  + ✅ Install dependencies
  + ✅ Run Unit Tests via pytest
  + ✅ Run Security Scan with bandit
  + ✅ Generate SBOM with cyclonedx-py
  + ✅ Docker Build & Push to Docker Hub
* ✅ Image successfully tagged and pushed:

bash

CopyEdit

docker.io/<your-username>/raman-micro:latest

✅ *End-to-end CI pipeline executed successfully with SBOM + Security*

**🎯 Summary**

| **Component** | **Status** |
| --- | --- |
| Cookiecutter Setup | ✅ Done |
| Flask App Endpoints | ✅ Tested |
| Prometheus Exporter | ✅ Working |
| Unit Tests | ✅ Passed |
| Bandit Scan | ✅ Clean |
| SBOM Generation | ✅ Done w/ warning (unpinned deps) |
| Docker Build | ✅ Success |
| DockerHub Push | ✅ Done |