GitLab CI/CD Pipeline Overview This project implemented a six-stage GitLab CI/CD pipeline for a simple Java “AverageServer” web service. The pipeline consisted of the following stages:

build – compile Java code

test – verify basic functionality

package – build Docker image

smoketest – run the image and test response

scan – scan the Docker image for vulnerabilities

deliver – push final image to Harbor registry

The pipeline was implemented in .gitlab-ci.yml and tested using multiple iterations to ensure all stages executed successfully.

Initially I created just four stages and it failed a few times and then I changed the docker images and the pipeline. Following are the errors I got after I created a correct pipeline:

GitLab CI log of a build error

$ docker build -t average-app . ERROR: Cannot connect to the Docker daemon at unix:///var/run/docker.sock. Is the docker daemon running? Cleaning up project directory and file based variables ERROR: Job failed: exit code 1

Explanation: Initially, the Docker daemon was not accessible in the GitLab CI environment because the DOCKER_HOST and docker:dind service were not properly configured. GitLab CI log of a successful build $ javac -d build src/AverageServer.java $ ls build/ AverageServer.class

Explanation: After correcting the Docker-in-Docker setup with: variables: DOCKER_HOST: “tcp://docker:2375” DOCKER_TLS_CERTDIR: “” services: - docker:dind

The build successfully compiled the Java code and saved the output in the build/ directory.

2. GitLab CI log of a test error $ ./tests/test_smoke.sh /bin/sh: eval: line 169: curl: not found ERROR: Job failed: exit code 127

Explanation: The curl command was missing in the Docker image used for the smoketest stage (docker:26 image has no curl preinstalled).

GitLab CI log of a successful test running the container $ docker build -t average-app . $ docker run -d -p 8199:8199 –name average-test average-app $ sleep 5 $ response=$(curl - s - XPOSThttp : //localhost : 8199 - H'Content - Type : application/json' - d'"numbers" : [2, 4, 6]')Response : 4$ docker stop average-test $ docker rm average-test

Explanation: After switching to an image with curl support or installing curl manually, the test successfully launched the container and validated that the API correctly computed the average of the input numbers.

3. GitLab CI Log of Successful Test Running the Container $ docker run -d -p 8199:8199 –name average-test average-app c14757eb6dd54ebe6bb7fd5706058dd120c8a4730dc7ad5810a515e7 $ sleep 5 $ response=($curl - s - XPOSThttp : //localhost : 8199 - H"Content{-}Type : application/json"{-}d'"numbers" : [2, 4, 6]')Response :$ 4 docker stop average-test $ docker rm average-test

Explanation: This output shows a successful smoketest run. The container was started, responded correctly with average 4, and then was stopped and removed cleanly.

3. GitLab CI Log of Scan $ trivy image –skip-version-check –severity HIGH,CRITICAL -f json -o trivy-report.json harbor.treok.eu/devops_compcs140/compse140-test-service1:latest 2025-10-27T22:02:43Z FATAL Fatal error run error: image scan error: scan error: unable to initialize a scan service: unable to find the specified image "average-app:latest" in ["docker" "containerd" "podman" "remote"]: 4 errors occurred:
    - docker error: unable to inspect the image (average-app:latest): Error response from daemon: No such image: average-app:latest
    - containerd error: containerd socket not found: /run/containerd/containerd.sock
    - podman error: unable to initialize Podman client: no podman socket found: stat podman/podman.sock: no such file or directory
    - remote error: GET https://index.docker.io/v2/library/average-app/manifests/latest: UNAUTHORIZED: authentication required Cleaning up project directory and file based variables ERROR: Job failed: exit code 1

Explanation: Trivy couldn't find the image locally because GitLab runners don't persist images between jobs. The solution was to scan the image from Harbor instead of local Docker by changing: trivy image harbor.treok.eu/devops_compcs140/compse140-test-service1:latest

4. GitLab CI Log of Delivery $ echo $HARBOR_PASSWORD | docker login harbor.treok.eu -u $HARBOR_USERNAME - -password - stdinLoginSucceeded$ docker push harbor.treok.eu/devops_compcs140/compse140-test-service1:latest The push refers to repository [harbor.treok.eu/devops_compcs140/compse140-test-service1] Successfully pushed image Cleaning up project directory and file based variables Job succeeded

Explanation: The delivery stage successfully logged into Harbor using credentials stored in GitLab CI variables and pushed the final image. This confirmed full pipeline functionality from build → test → package → scan → deliver.

Summary What was difficult?

-Getting Docker-in-Docker to work correctly inside GitLab runner.

-Understanding the difference between local Docker context and the one inside GitLab jobs.

-Debugging curl: not found and connection refused errors during smoke tests.

-Getting Trivy to scan the image from Harbor rather than local Docker.

-Managing Harbor authentication for image pushes.

What were the main problems? -Cannot connect to the Docker daemon → fixed by setting DOCKER_HOST: “unix:///var/run/docker.sock”

-curl: not found → fixed by using an image with curl installed (curlimages/curl or alpine/curl).

-Image not found for Trivy → fixed by scanning the image from Harbor.

-Push unauthorized → fixed by adding docker login before push and using correct $HARBOR_USERNAME