ACIT 4850 – Lab 8 – GitLab CI

Instructor	Mike Mulder (mmulder10@bcit.ca)
Total Marks	10
Due Dates	Screenshots due by end of next class: • End of class on March 18th for Set C • End of class on March 19th for Set B • End of class on March 21st for Set A

Group Work

You will be working on the same Azure cloud environment as the previous lab, shared with your Lab partner. This lab will be done together with your partner.

You will need the following application(s) running for this lab:

- GitLab
- *Apache and Jenkins (if you need to compare you Jenkins pipeline with your GitLab CI pipeline)*

This is an experimental lab so you will have to use the GitLab documentation and your skills in Docker and Shell script commands.

You are going to re-produce your Python pipeline for the Point project in GitLab CI. Note that it will not be a shared library for this lab (or the equivalent in *GitLab CI).*

Note: Assignment 2 will build upon this lab

Part 1 – Install a GitLab CI Runner

Use the documentation on your GitLab installation to create a Project Runner for your Point project in GitLab. The runner should be a 'Runner with a Shell executor' and installed on the GitLab VM.

You create a runner you should be able to follow the instructions on your GitLab installation. Go to the Point project -> Settings -> CI/CD -> Runners.

Part 2 – Setup Your Dependencies

If you are using a Shell Runner install your dependencies on the VM where the Runner is running. This include those dependencies required for the pipeline. For Jenkins, these are defined in the Jenkins Dockerfile and includes Python, Pylint, Zip, Coverage, etc.

Part 3 – Re-Create Your Python Pipeline

Implement the following stages in an identical manner to your Jenkins pipeline in the gitlab-ci.yml file:

- Build – Setup your Virtual Environment and install the requirements.txt
- Lint – Run pylint on your code

- Test and Coverage – Clear out the XML test results, run coverage on each test file (using a loop), generate the coverage report and junit your XML test results
- Zip – Zip up all the Python files and make the Zipfile an artifact

A few caveats to look out for:

- Loops for the Test and Coverage stage need to be done in bash
- GitLab CI automatically zips up artifacts. So the "Zip" in the Zip stage may not be necessary

<u>**Questions**</u>

1. What are two advantages of GitLab CI over Jenkins?
2. What are two disadvantages of GitLab CI over Jenkins?
3. Research GitLab CI Runners and describe the difference(s) between a Docker executor and Shell executor for a GitLab CI runner? What would be the advantage of one over the other?

Demo, Grading and Submission

Demo – Show your running GitLab CI pipeline and gitlab-ci.yml file for the Point project to your instructor, proving that it is functionally equivalent to your Python pipeline in Jenkins.

Submit to the Lab 8 dropbox on the Learning Hub:

- Answers to the questions above
- Your gitlab-ci.yml file
- A screenshot of your successful pipeline with the four required stages