Champlain College - Lennoxville
Final project: An open-source web application

PROGRAM:	420.B0 Computer Science Technology	
COURSE:	Transactional Web Applications 2	
COURSE CODE:	420-530-LE	
WEIGHT:	20% of the final score	
SEMESTER:	Fall 2023	
INSTRUCTOR:	Francis Gauthier fgauthier@crcmail.net	Office C-239

Project overview

The goal of the project will be to use some code from an open-source web application. The students will install and document the application. After a review of the application, they will come up with a CI/CD pipeline with stages to build, prepare, enhance, and deploy the web application.
This project is meant as a complete review and application of the concepts seen in this course so far. It is also an opportunity to showcase one's ability to understand any web application based on our current knowledge.

Working in Teams

This project should be done in teams of two. If a student wants to work alone on the project, they can, but the scope will be only slightly reduced.
Each team must be approved by the teacher to avoid any problematic collaboration.
Team approbation and application validated by Nov. 27th 2023.

Part 1 - Locate, install, and document the application

Intended timeline: Nov. 20th - Nov. 30th

Locate

Your first locate an open-source web application publicly available online. Verify that the license allows public access.

You will be evaluated on:

- The complexity of the web stack (backend only, frontend only or full-stack, with DB or no DB)
- The complexity of the application chosen (number of features, technologies used)

**Think about part 2 of the project. A simpler application can result in losing points for this task but might make it easier to achieve the part 2 of the project. Choose one project within your reach.*

Install

Next, you will be to clone the application on your machine. Then, you must install and run the application locally.
To do so:

- Follow the documentation provided by the application owners (usually in Readme.md)
- Install the required dependencies
- Provide the environment variables needed (if any)
- Provide a database connection (if any)
- Run the application locally

You will be evaluated on:

- The ability to run the application locally

Document

Next, you should document that the application does.
Using a text editor or any other tools, write a summary of:

- What is the general purpose of the application?
- If the application has a back-end:
    - The routes available on the API
    - The purpose of these routes
- If the application has a front-end:
    - The different views (take screenshots to document them)
    - What are the actions possible?
    - Is there authentication required?
- If the application has a database:
    - The different collections/table names
    - What are the models used in the database?

You will be evaluated on:

- The thoroughness of the documentation
- The exactitude of the documentation
- The format/organization of the documentation

Part 2 - CI/CD pipeline

Intended timeline: Nov. 30th - Dec. 11th

Each team will then have to build a CI/CD pipeline on GitLab.
To do so:

- Clone the application
- Create a new GitLab project
- Add the remote to your repository cloned (git remote add …)
- Push on GitLab

Stages required

For a team of two:

- (Required) Install stage (setting up, installing node_modules, etc.)
- At least **two** stages producing artifacts:
    - A Lint stage
    - A documentation stage (produces automatic documentation based on annotations)
    - A test stage (unit tests)
    - A test stage (integration tests)
- (Required) Build stage (preparing resources for the deployment stage)
- (Required) Deploy stage (deploying the web application on AWS)

For students working alone:

- (Required) Install stage (setting up, installing node_modules, etc.)
- At least **one** stage producing artifacts:
    - A Lint stage
    - A documentation stage (produces automatic documentation based on annotations)
    - A test stage (unit tests)
    - A test stage (integration tests)
- (Required) Build stage (preparing resources for the deployment stage)
- (Required) Deploy stage (deploying the web application on AWS)

The stages artifacts

The stage in the middle of your pipeline should produce an artifact.
Here are the files expected as types of artifacts:

A lint stage

For a lint stage, it is possible to get the results of the linting in an output file (JSON for example).
To do so, use the --output-file option: https://eslint.org/docs/latest/use/command-line-interface#output
Objective: to get a file specify the number of warnings, errors left and the files/folders covered.

A documentation stage
A generated documentation file that can be consulted by others to know more about the app.
Format: HTML, PDF, or other formats depending on the documentation tool.
Contents: Comprehensive documentation generated from code comments, README files, or other sources.
Example: documentation/index.html generated by js-doc or user-guide.pdf.

A test stage
You can output test results:
Format: JUnit XML, HTML, or other structured formats.
Contents: Detailed information about test cases, their outcomes, and any failures.
Example: test-results.xml or test-results.html.
or
You can output code coverage reports:
Format: HTML, XML, or other structured formats.
Contents: Percentage of code covered by tests, along with details on which lines were covered.
Example: LCOV coverage-report/index.html or coverage-report/coverage.xml.

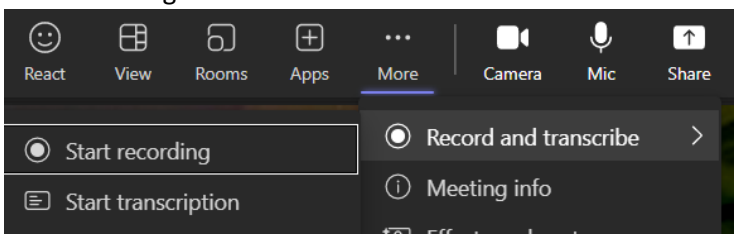
Part 3 - Video/Oral presentations

Each team will do a small **video** presentation to present the result of the application that they worked on.
The video should be a screen share of somebody navigating your application and commenting it.
To include in the video(s):
It can be one video or a series of three smaller videos.

1. **An overview of the application.**
    a. Explaining what the purpose of the application
    b. Explain the features of the application briefly
    c. Show an example of what a user could do with the application
2. **Showcasing the deployment of the application**
    a. Showing that the application is on a deployed public domain
    b. Showing the cloud services currently supporting the deployment. For example, going over the AWS console or MongoDB Atlas and showing the active services in place.
    c. Showing how each part of the application are configured to access one another. For example, where environment variables had to be set.
    d. Talking about the difficulties faced during the deployment process.
3. **Showcasing the CI/CD pipeline**
    a. Showing the latest successful pipeline run.
    b. Showing the different jobs and stages of the pipeline.
    c. Showing and downloading a generated artifact of the pipeline.
    d. Talking about the difficulties faced during the pipeline process.

Recording the video

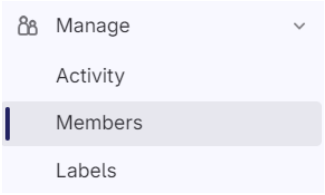
To record your screen, a great way to do so is:

- Create a Teams meeting in your agenda.
- Invite your teammate
- Join the meeting as a videoconference call
- Start recording:
- 

- Share your screen
- Share your camera
- Record your presentation
- End the recording and screen share. The video recording will be sent out in the chat of the meeting. Download it and send it to your teacher.

Submission

The code must be pushed in a GitLab repository.
First, give access to your teammate, then your teacher. To add members, locate the “Manage” section or the project and click “members”.
Click “Invite members”. Add fgauthier@crc-lennox.qc.ca as one of the invited members and select **owner** as role.



In the repository, make sure that these files/folders are present:

- The source code of the app
- The .gitlab-ci.yml file of the pipeline
- A **documentation/** folder with inside
    - your application documentation of part 1
    - your recorded videos for part 3
- A readme.md which gives the link to your deployment

The submission deadline is Thursday December 14th, 2023, End of Day. The repositories will be cloned on December 15th.

Keep the deployments on until your project is corrected. (Do not terminate any cloud environment before receiving your grade)

Grading

Criteria	Weight
Part 1 - Install and document	
- Complexity of the application and technology stack chosen	10%
- The thoroughness and exactitude of the documentation	10%
- The format and organization of the documentation	5%
Part 2 - CI/CD and deployment	
- Proper CI/CD pipeline syntax with minimum required stages	10%
- Proper install stage commands	5%
- Commands for the 1 or 2 extra stages	
o Relevance of the stage actions	5%
o Proper commands to execute the stage(s)	5%
o Production of artifacts	5%
- Proper build stage commands	5%
- Proper deployment commands (automated)	5%
- Proper deployment of the different components of the application in a cloud environment	10%
Part 3 - Video presentation	
- Overview of the application (business logic, features)	5%
- Showcasing the deployment	10%
- Showcasing the pipelines stages	10%