2nd Assignment

Scenario

As DevOps experts, you and your team were hired as a consultant by a company in order to improve their development processes and make them more effective and efficient. Your goal is to set up a CI/CD pipeline to automate the entire development process using the Gitlab CI/CD infrastructure. The pipeline is structured in 7 stages as follows.

Stages:
- *build*: resolve dependencies and compile the code
- *verify*: run static and dynamic analysis (see further details*)
- *test*: run unit/integration/performance tests
- *package*: generate a package ready to be released
- *release*: make the package available (e.g., for download)
- *docs*: generate and publish updated documentation
- *deploy*: deploy the software to the field

You can find below some ***examples*** about how to implement these stages for different programming languages.

	Java	***PHP***	***Python***	***Node.js***
Build	Run Maven *compile* to download dependencies and compile source code	Run Composer *install* to download vendor packages	Run pip *install* with requirements file to download dependencies	Run npm *install* to download vendor packages
Verify	Run SpotBugs and Checkstyle maven plugins, and Valgrind.	Run PHP-CS-Fixer	Run Prospector and Bandit	Run ESLint and Flow
Test	Run maven *test*	Run *phpunit*	Run *pytest*	Run *npm test*
Package	Run maven *package* to produce the artifact	Tag the repository exploiting the Gitlab API	Use *setuptools* and *wheel* to create Source Archive and Built Distribution	Run *npm run build* to run your build script which produce your dist folder
Release	Build a Docker Image with the produced package and	Publish the repository as a Composer package on the	Publish the Build Distribution to PyPI with *twine*	Build a Docker Image with the produced dist folder and push

	push it to the Gitlab Container Registry	Gitlab Composer Package Registry		it to the Gitlab Container Registry
Docs	Run maven *javadoc:javadoc* to generate Javadoc and publish it to GitLab Pages	Run *phpDox* to generate PHP API doc and publish it to GitLab Pages	Write an handwritten documentation with Markdown, generate a static website with MkDocs, and publish it to GitLab PAges	Use documentation.js to generate API doc and publish it to GitLab Pages
Deploy	Connect to a VM via SSH and run a Docker container based on the previously released Docker image	Connect to a VM via SSH and download the new package version (in a web server, or whatever…)	Connect to a VM via SSH and install the Python package and run it	Connect to a VM via SSH and run a Docker container based on the previously released Docker image

PLEASE NOTE:
The development of mobile apps (i.e., Android) is discouraged. Known issues have been experienced in running emulators within Gitlab CI pipelines. No technical support is provided to groups working on mobile apps.

*__VERIFY STAGE__:
This stage **MUST** consist of static and dynamic analysis jobs executed in parallel.

Assignment Goal

Your goal for this assignment is to make this pipeline work for a given application. You are free to pick up an application of your choice.
This can be:

1. An application that you already own (e.g., from a previous course, etc.).
2. An open source application of your choice.

Pay attention that the *focus is on the pipeline*, **NOT** on the application.

This means that a simple application can be perfectly suited in this case.

Your pipeline **must include comments** that explain the individual steps.

Group composition

The work has to be done in groups.

1. Teams of 2 or 3 developers MUST implement a pipeline that includes the **first the 6 stages (from build to docs)**. The implementation of the 7th stage (*deploy*) is optional.
2. Teams of 4 developers MUST address **the first 6 stages (from build to docs)** from the list above and define at least two pipelines for **two different git branches** (each pipeline may have a different number of stages, however all the 6 stages must be covered with the implemented pipelines). The implementation of the 7th stage (*deploy*) is optional.

Single developers are **NOT ALLOWED**. Exceptions to this rule can be considered in case of sound motivations. Please contact leonardo.mariani@unimib.it if this is your case.

Teams larger than 4 developers are NOT allowed.

Instructions

1. Create a GitLab account.
2. Create a repository named "*2024_assignment2_{your_project_name}*".
3. Add the user *"sw_dev_process_course"* to the GitLab repository as soon as the repository is created.
4. Assign to the user *"sw_dev_process_course"* the developers privileges.

Submissions

Report the progress of the work on the README.md file on the Gitlab repository. Submit a **PDF** version of this file as a report. This report must briefly document the work done. The submission must include the address of the Git repository with all the material. The document can be written either in English or Italian. **Only one member of the group must submit the PDF.** Report the names of all members in the PDF.

There are **2 submissions** of this document. The **first** submission simply represents the state of your work at that time. The **second** submission is the updated and **final** version of this document. The dates of the submissions are reported on the corresponding section of the course website. Please check it out.

Evaluation

This assignment is expected to show your professional skills and it will be evaluated considering:

1. The submitted report.
2. The implementation of the pipeline, including its documentation.
3. The inspection of the repository and all the artifacts used to set up the pipeline.

Pipelines will be checked for plagiarism against pipelines available on the Internet, and pipelines produced by other students (both from past years and this year). Students submitting material that is not the result of their original work will not pass the exam. Appropriate actions will also be taken by the teachers.

Troubles with time available to run pipelines

If a group finishes the time available to run the pipelines before completing the assignment, please contact us, we have additional pipeline execution time that we can distribute exploiting the educational account (still, use the available time wisely).

Alternatively, a group can create a new project and copy the content from the old repository to the new repository, obtaining additional time for pipelines. In such a case, the team has to invite our user also to the new repository and indicate the presence of this second repository in the shared spreadsheet.