SCALE FOR PROJECT CLOUD-1 (/PROJECTS/42CURSUS-CLOUD-1)

You should evaluate 2 students in this team

★

Git repository

```
git@vogsphere-v2.1337.ma:vogsphere/intra-uuid-cb47d8c1-84c9-4dc
```

Introduction

For the smooth running of this evaluation, we ask you to
respect the following rules :

- Remain courteous, polite, respectful and constructive in all your dealings.
situations during this exchange. The bond of trust between the
community 42 and you depend on it.

- Highlight to the person (or group) being graded
possible malfunctions of the work rendered, and take the
time for discussion and debate.

- Accept that sometimes there may be differences.
of interpretation on the subject's demands or the scope of the
functionalities. Stay open-minded to the vision of
the other (is he or she right or wrong?), and note the more
honestly possible. The pedagogy of 42 makes sense only if the
peer-evaluation is done seriously.

Guidelines

- This project requires correcting external resources to 42, in
plus the student's usual GIT deposit. The student must therefore
connect as root access as possible to the service he will have
chosen, his root account must be his email address.
student or his login. Otherwise, indicate a case of
cheats, and the defense stops.

- Make sure to check that the GIT deposit is indeed the one
corresponding to the student or group, and the project.

- Verify meticulously that no malicious aliases have been used.
used to mislead you and get you evaluated differently
thing than the contents of the official repository.

- Any script that is supposed to facilitate the evaluation provided by one of the
two parties must be rigorously checked by the other party
to avoid unpleasant surprises.

- If the student corrector has not yet done this
project, it is mandatory for this student to read the project's
subject in its entirety before beginning this presentation.

- Use the flags available on this scale to signal a
rendered empty, non-functional, a standard error, a case of cheating,
etc. In this case, the evaluation is completed and the final grade is
is 0 (or -42 in the special case of cheating). However, out of
cases of cheating, you are encouraged to continue to exchange
around the work done (or not done precisely) for
identify the problems that have led to this situation and the
Avoid for the next render.

- The defense stops at the first wrong answer.
met. But you can of course continue to exchange
around the work done.

Attachments

subject.pdf (https://cdn.intra.42.fr/pdf/pdf/148046/en.subject.pdf)

Overview

This section aims to verify that the website is indeed working as intended.

Website is working

Go to the student's website:
-Is this site running under WordPress? If you
Refresh the page multiple times, does the content of the page
remains the same (except for the visual identification
of the server we are on)?
-Take a walk in the pages, refresh the content, does it
works?

Yes No

Web security basics

What arrangements have been made by the user
to secure its content?
-Only the ports 80/443 (HTTP/HTTPS) should be accessible from
the outside of the instances.
-The database must be accessible only on the port
necessary for its operation and only in local.

If other security mechanisms are
present, it's cool, talk about it.

Yes No

Infrastructure as code

This section will assess the "infrastructure as code" aspect of the project. The goal here is to asses a student understood what infrastructure as code is about. What is idempotency? What are the diffe provision something? Does the deployment works?

Deployment

Verify that the Ansible deployment script works by deploying the
the student code on another machine than the one they own.
It must be possible to get the same site by executing the same script.

Yes No

Idempotency of deployment

Check that you can apply the same script several times and get
a site that works.

Yes No

Modular architecture

Is the student's code broken down into several relevant roles?
Typically the web server in one role, the database in another role, etc.
The goal is to be able to deploy the parts independently of each other.

Yes No

Secrets managements

The code does NOT contain any hard-coded secrets. Like a password
to connect to the database hard-coded in the code.
Be ruthless about it. People get fired for this kind of neglect.

Yes No

Containers

In this part, we will review how well the containers are configured and managed

Containers best practices

Are the docker-compose organized to benefit from the cache?
ie. Having the part that often changes at the END of the Dockerfile and the part
that does not change a lot (system dependencies) at the beginning?

Are the containers as small as possible?

Yes No

Modular containers

Are the containers well separated from each other?
Is there one container for the database, one for the PHP run time, one for the web proxy etc.?
It must be possible for these containers to communicate with each other.

Yes No

Container and storage

Is the site data stored in mounted volumes?
Is it possible to view these volumes for an administrator?
using the docker commands?

Yes No

Orchestration

Is it possible to launch all containers in one go? is it possible to restart them individually?

Yes No

Ratings

Don’t forget to check the flag corresponding to the defense

Ok Outstanding project

Empty work No author file Invalid compilation Norme Cheat

Incomplete group Forbidden function Can’t support / expl

Conclusion

Leave a comment on this evaluation (2048 chars max)

Finish evaluation