T8 - Network & Sys Admin

T-NSA-800

SLA

DevOps

1.3.1

SLA

binary name: SLA_*$AcademicYear*_*$GroupNumber*.zip
delivery method: Moodle

- The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.
- All the bonus files (including a potential specific Makefile) should be in a directory named *bonus*.

A client of yours wants to change outsourcing, and is interested in your profile.

They want to have different metrics on their infrastructure, to be reactive on incidents that could impact them and garantuee a high-disponibility service.

To test your skills, they decide to assign your group an infrastructure mimicking theirs on a Public Cloud service, but with limited components.

This infrastructure will be available on mondays and tuesdays, from 10am to 6pm, and is made of:

- an **application server** which carries a vote page, a result page, and a worker,
- a **database server** which carries a PostgreSQL and Redis database,
- a **Docker server** which will be at your disposal.

Your client wants to test your reactivity but also your capacity to come with improvements and suggestions.

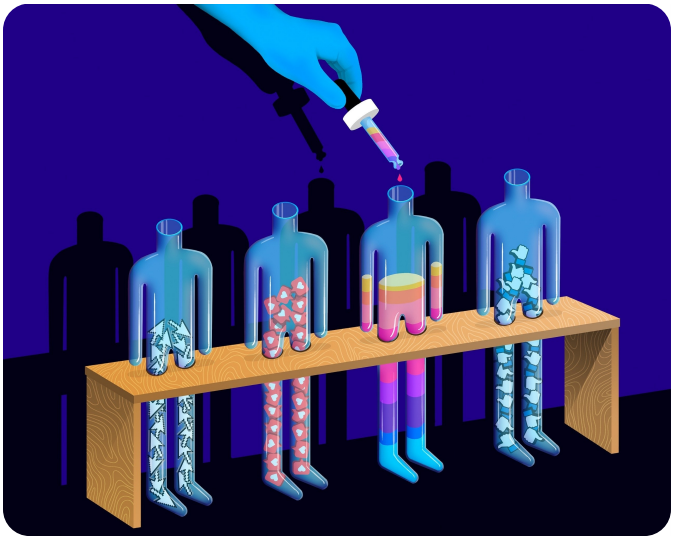
In a first phase, you will have time to discover, get familiar with the infra and add your own tools to it.

Use the first phase to test extensively, prepare your metrics, monitoring, automated deployments, backups and restores,… Feel free to restart and reinstall without taking care of the SLA.

Then, you will be tested on realm-like incidents.

GET READY FOR THE APPLIANCE TEST !!

The test can start at any time after the first 3 weeks.



The client will then start monitoring the two applications in order to measure their availability over several weeks. It will be up to you to keep them accessible and functional.

Be careful, if a database fails, your application will no longer respond correctly to client monitoring.

Obvoiusly, you must be able to restore data from backup in case of database failure.

During these weeks, incidents / events will be regularly triggered, for example :

- cessation of a service
- request spam
- resource saturation (CPU / RAM / FS)
- database crash
- killed container
- and so on…

During each incident, you will need to:

1. notify the client of the start of unavailability,
2. restore the service in the exact same state as it was before the incident,
3. notify the client of the restoration of the service with a brief analysis of the reason / the correctives applied / the solutions so that it does not happen again.

No need here to advice you to count on reliable metrics, monitoring and backup / restore system correctly documented, in order to be as reactive as possible.

At the very end of the exercise, you will be asked to:

- present your different **monitoring tools and dashboards**
- carry out a **REX** on the main incidents detected / treated
- show ways of **improving the resilience** of this infrastructure

If you lose a machine, we will have the possibility to destroy it in order to rebuild it, but it will then be delivered **in the state it was the first time that you connected to it**. Therefore document (and automate) the different things you deploy and how you deploy them.