DevOps - LLAMA GEMST

tbab, harw, gusm, edtr, mihr

May 2024

1 I'm a section

1.1 I'm a subsection

2 Tasks

- Miha, Simon: security analysis
- Simon: find & install template for latex file
- Simon: look at Mircea's slides and make us a template for graphs
- Tomas: get gh actions to compile latex
- Gustave, Eduardo: finish Docker swarm
- Eduardo: can you figure out if CodeClimate is installed on repo? if not, install it and give me logins. See session 07 for instructions

3 Introduction

The following report described the work done during the Spring 2024 DevOps, Software Evolution and Software Maintenance taught at the IT University of Copenhagen. Our report includes the description of the final state of our system, the process we followed to develop and deploy it, lessons we learned, and reflections on technologies used and high and low level decisions we made throughout the semester.

4 System's Perspective

A description and illustration of the:

Design and architecture of your ITU-MiniTwit systems All dependencies of your ITU-MiniTwit systems on all levels of abstraction and development stages. That is, list and briefly describe all technologies and tools you applied and depend on. Important interactions of subsystems For example, via an illustrative

UML Sequence diagram that shows the flow of information through your system from user request in the browser, over all subsystems, hitting the database, and a response that is returned to the user. Similarly, another illustrative sequence diagram that shows how requests from the simulator traverse your system. Describe the current state of your systems, for example using results of static analysis and quality assessments. MSc should argue for the choice of technologies and decisions for at least all cases for which we asked you to do so in the tasks at the end of each session.

4.1 How we refactored our app and why we chose GoLang

4.2 How a request is processed when it comes from the front-end/simulator

4.3 Architecture graphs of the system

4.4 List of dependencies (tools and techs) (add some quick description of why we chose a specific dependency if we're not going to write a paragraph about it later in the text)

4.5 Db design and cloud db?

4.6 Static analysis tool results

5 Process' Perspective

This perspective should clarify how code or other artifacts come from idea into the running system and everything that happens on the way.

In particular, the following descriptions should be included:

A complete description of stages and tools included in the CI/CD chains, including deployment and release of your systems. How do you monitor your systems and what precisely do you monitor? What do you log in your systems and how do you aggregate logs? Brief results of the security assessment and brief description of how did you harden the security of your system based on the analysis Applied strategy for scaling and upgrades Also, in case you have used AI-assistants during your project briefly explain which system(s) you used during the project and reflect how it supported/hindered your process.

5.1 Diagram showing our CI/CD chains, how we release and deploy

5.2 How we monitor our systems and what we monitor/how we use this information to improve our system (give the example of indexing the db)

5.3 Logging and all that witchcraft crap. What we log and why we used those technologies, how we make sure that we're not logging too little but also not logging too much

5.4 Security assessment

5.5 Strategy for scaling and upgrading our system (talk about moving to a cloud db, and buying more droplets when moving to swarm

5.6 Our use of AI assistants

6 Lessons Learned Perspective

Describe the biggest issues, how you solved them, and which are major lessons learned with regards to: evolution and refactoring operation, and maintenance of your ITU-MiniTwit systems. Link back to respective commit messages, issues, tickets, etc. to illustrate these.

Also reflect and describe what was the "DevOps" style of your work. For example, what did you do differently to previous development projects and how did it work?

6.1 Issues we ran into:

- Container being killed by the OS on the cloud
- Database cleared for some reason every now and then?
- Slow db querying
- Swarm setup issues

6.2 Maybe talk about how nice it was to have a CD pipeline to work on a project together

6.3 What tools we used during the semester, how we divided work, how we shared knowledge?

6.4 The three DevOps ways and what we used?