T6 - DevOps

T-DOP-600

DevOps

Ansible Deployement

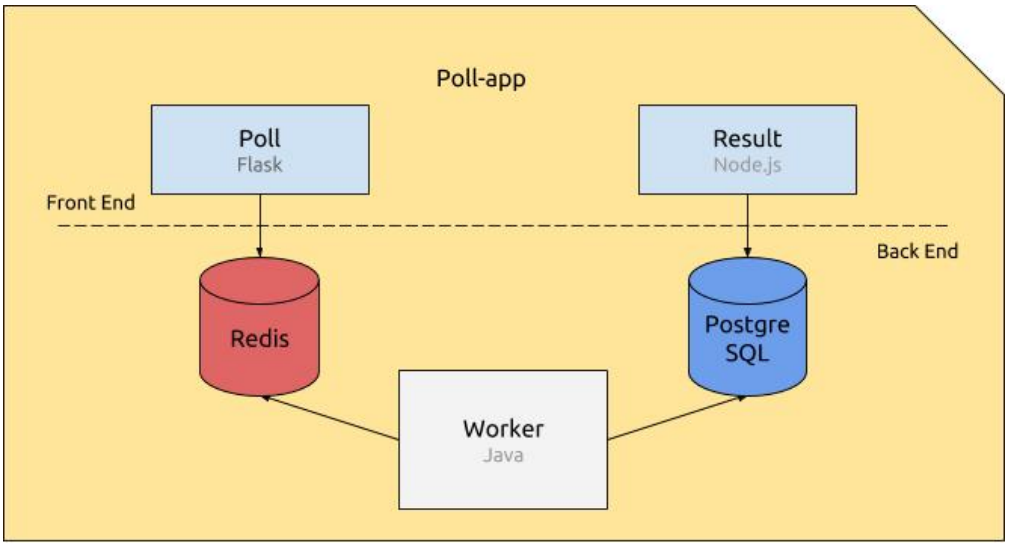
DevOps

repository name: DOP_automation_*$ACADEMICYEAR*
repository rights: ramassage-tek

- The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.

This project aims to teach you configuration management and automated deployment using ansible.

The application you are working on during this project is a simple poll web application. Poll is a Python Flask web application that gathers the votes to push them into a Redis queue. The Java Worker consumes the votes stored in the Redis queue, then pushes it into a PostgreSQL database. Finally, the Node.js Result web application fetches the votes from the DB and displays the result.



For this project, you have to hand in an `playbook.yml`, an inventory and a folder named `roles` that will hold your ansible's roles.

+ Roles

You should write the following roles:

- **base**: installs essential packages using `apt` and configure instance.
- **redis**: installs and setup Redis.
- **postgresql**: Installs PostgreSQL 12 and `psql`. It creates a user `paul` with the password `democracyIsFragile` and limited permissions. The schema of the database `paul` must be created during this step.
- **poll**: uploads `poll` service, installs dependencies and runs.
- **worker**: uploads `worker` service, installs dependencies, builds and runs.
- **result**: uploads `result` service, installs dependencies and runs.

Always use the more adapted ansible module instead of using a `command`. Have a look at `apt_key`, `apt_repository`, `pip`, etc.

All services must be managed by `systemd` and must start automatically on boot.

In order to respect the 12 factor best practices, services must be configured with environment variables: host, port, user, password, database name, port…

Docker and Ansible Galaxy are forbidden in this project.

After applying an Ansible playbook twice, you should have 0 “changed” tasks in your “PLAY RECAP” (see the example bellow)

```
PLAY RECAP *********************************************************************
1.1.1.1    : ok=49 changed=0 unreachable=0 failed=0 skipped=12 rescued=0 ignored=0
2.2.2.2    : ok=25 changed=0 unreachable=0 failed=0 skipped=16 rescued=0 ignored=0
3.3.3.3    : ok=42 changed=1 unreachable=0 failed=0 skipped=22 rescued=0 ignored=0
```

+ Environment

Your inventory will have 5 groups containing 1 instance each: `redis`, `postgres`, `poll`, `result` and `worker`.

You will need 5 virtual machines based on Debian 10 (Buster). You can run it locally but it is highly recommanded to use a cloud platform:

- On Digital Ocean ($50 credit, 30 days)

This is a referral link. Feel free to invite each other !
Don't spend too much credit. You will need 50% of this budget for the next project.

DELIVERY

Your project will be tested using the following command line:

Terminal

```
~/T-DOP-600> export ANSIBLE_VAULT_PASSWORD_FILE=/tmp/.vault_pass
echo verySecretPassword > /tmp/.vault_pass
ansible-playbook -i production playbook.yml
```

In the end, your repository must at least contain the following files:

```
|-- production
|-- playbook.yml
|-- poll.tar
|-- result.tar
|-- worker.tar
|-- group_vars
|   `-- all.yml
`-- roles
    |-- base
    |   `-- tasks
    |       `-- main.yml
    |-- postgresql
    |   |-- files
    |   |   |-- pg_hba.conf
    |   |   `-- schema.sql
    |   `-- tasks
    |       `-- main.yml
    |-- redis
    |   |-- files
    |   |   `-- redis.conf
    |   `-- tasks
    |       `-- main.yml
    |-- poll
    |   |-- files
    |   |   `-- poll.service
    |   `-- tasks
    |       `-- main.yml
    |-- result
    |   |-- files
    |   |   `-- result.service
    |   `-- tasks
    |       `-- main.yml
    `-- worker
        |-- files
        |   `-- worker.service
        `-- tasks
            `-- main.yml
```

Any clear-text password in your git history will cause a failure to the project.

GOING FURTHER

- Replicate web services and add a load balancer in front
- Protect web services with LetsEncrypt
- Add a password to Redis (!)
- Datacenter network is not trusted, can you encrypt Postgresql and Redis connections ?

+ SUPER BONUS

- Create integration tests for your Ansible roles
- Build a VM template with Packer
- Run a benchmark to configure your reverse proxy(-ies) correctly
- Setup a CI/CD pipeline on Gitlab
- Cross platform Ansible roles (Debian/Red Hat/…)