

Projet 10 - Déployez votre application sur un serveur comme un pro !

Étudiant : Guillaume OJARDIAS Mentor : Erwan KERIBIN
Mentor évaluateur : Benoît PRIEUR

Sommaire

I. Présentation	2
I.1. Liens du projets	2
II. Démarche de création	2
II.1. Mise en place du serveur	2
II.2. Mise en place du CI	3
II.3. Monitoring de l'application	4
III. Bilan	4

I. Présentation

Ce projet se base sur le projet 8 : *Créez une plateforme pour les amateurs de Nutella.*

L'objectif principal de ce projet est de déployer l'application du projet 8 sur un serveur paramétré par nos soins. Pour cela nous devons mettre en œuvre les éléments suivants : - un outil d'intégration continue type Travis, - déployer l'application sur un serveur du même type que ceux proposés par Digital Ocean, - un outil de monitoring permettant le suivi des performances du serveur, - un système de logging pour surveiller les logs de l'application du type Sentry, - créer un tâche CRON pour la mise à jour de la base de données de l'application à raison d'une fois par semaine, - et enfin utiliser un nom de domaine (optionnel).

I.1. Liens du projets

- Le site est visible en ligne à cette adresse : <http://projet-10.ojardias.io/>.

II. Démarche de création

II.1. Mise en place du serveur

- Création d'un Droplet Digital Ocean à l'adresse <http://167.172.169.38> :
- Mise en place de la connexion SSH, mise à jours des packages du serveur et création d'un utilisateur **guillaume** :
- Installation de **python3**, **postgresql**, **git**, **nginx** et **supervisor**.
- Clone du repository dans le répertoire **/home/guillaume/pur-beurre**.
- Paramétrage du pare-feu UFW :

```
~$ sudo ufw status
Status: active

To Action From
--
OpenSSH ALLOW Anywhere
Nginx Full ALLOW Anywhere
OpenSSH (v6) ALLOW Anywhere (v6)
Nginx Full (v6) ALLOW Anywhere (v6)
```

Figure 1: Configuration de UFW

- Configuration de NGINX et Supervisor :

```
server{
    listen 80;
    server_name projet-8.ojardias.io;
    server_name projet-10.ojardias.io;
    server_name pur-beurre.ojardias.io;

    root /home/guillaume/pur-beurre;

    location /static/ {
        alias /home/guillaume/pur-beurre/staticfiles/;
    }

    location / {
        proxy_set_header Host $http_host;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_redirect off;
        if (!-f $request_filename) {
            proxy_pass http://127.0.0.1:8001;
            break;
        }
    }
}
```

Figure 2: Configuration de NGINX

```
[program:pur-beurre]
command = /home/guillaume/pur-beurre/venv/bin/gunicorn --bind 127.0.0.1:8001 config.wsgi:application
user = guillaume
directory = /home/guillaume/pur-beurre/
autostart = true
autorestart = true
```

Figure 3: Configuration de supervisor

- Création de tâches CRON

```

# crontab -l
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m. every week with:
# 5 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
* * * * * /home/guillaume/update_repos/update_repos.sh >> /home/guillaume/update_repos/update_repos.log
0 12 * * sun /home/guillaume/pur-beurre/update_pur_beurre.sh >> /home/guillaume/update_repos/update_pur_beurre.log
0 12 * * sun rm /home/guillaume/pur-beurre/update_repos.log

```

Figure 4: Mise en place de tâches CRON

II.2. Mise en place du CI

- Configuration de Travis pour la partie Continuous Integration :

```

cat ./pur-beurre/.travis.yml
language: python
python:
  - '3.8'

env: DJANGO_SETTINGS_MODULE="config.settings.local"

services:
  - postgresql

script:
  - ./manage.py test -v 2
  - black . --check
  - isort . --check

```

Figure 5: Configuration de Travis

- Mise en place et configuration de Mergify pour le merge des Pull Requests :

```

cat ./pur-beurre/.mergify.yml
pull_request_rules:
  - name: Merge Guillaume's pull requests
    conditions:
      - "author=GuillaumeOj"
      - "label!=work-in-progress"
      - "label!=manual merge"
      - "status-success=Travis CI - Branch"
    actions:
      merge:
        strict: "smart"
        method: "rebase"
      delete_head_branch:
        force: True

```

Figure 6: Configuration de Mergify

II.3. Monitoring de l'application

- Configuration de Sentry pour le reporting des issues non gérées par l'application :

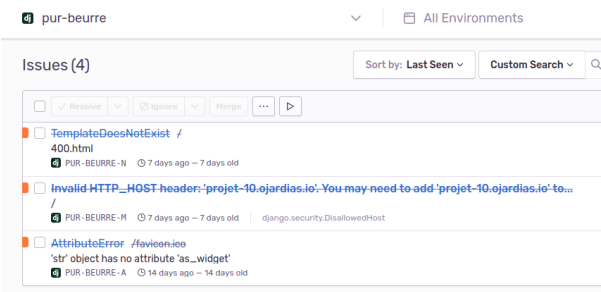


Figure 7: Dashboard de Sentry

- Mise en place d'un monitoring du Droplet en utilisant les outils mis à disposition par Digital Ocean :

Alert Policies [Setup instructions](#)

Name	Applied to
CPU is running high CPU is above 70% for 5 min	ubuntu-s-1vcpu-1gb-frs1-01
Disk Utilization is running high Disk Utilization is above 70% for 5 min	ubuntu-s-1vcpu-1gb-frs1-01
Memory Utilization is running high Memory Utilization is above 70% for 5 min	ubuntu-s-1vcpu-1gb-frs1-01

Figure 8: Monitoring Digital Ocean

III. Bilan