

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	4
II.3. Monitoring de l'application	5
III. Bilan	5
III.1. NGINX et Supervisor	5
III.2. Travis, Sentry et Monitoring	5
III.3. Mergify	5
III.4 Conclusion	6

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).

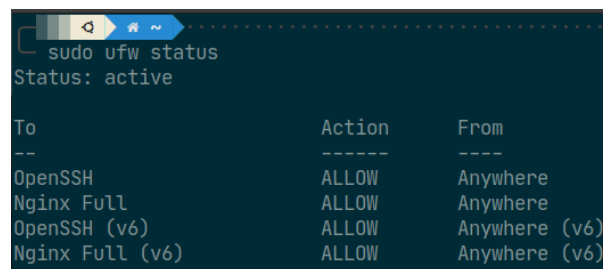
I.1. Liens du projets

- 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 :

```

-- sudo cat /etc/nginx/sites-available/pur-beurre
server{
    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;
        }
    }

    listen 443 ssl; # managed by Certbot
    ssl_certificate /etc/letsencrypt/live/pur-beurre.ojardias.io/fullchain.pem; # managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/pur-beurre.ojardias.io/privkey.pem; # managed by Certbot
    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot
}

server{
    if ($host = projet-8.ojardias.io) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

    if ($host = projet-10.ojardias.io) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

    if ($host = pur-beurre.ojardias.io) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

    listen 80;
    server_name projet-8.ojardias.io;
    server_name projet-10.ojardias.io;
    server_name pur-beurre.ojardias.io;
    return 404; # managed by Certbot
}

```

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 :

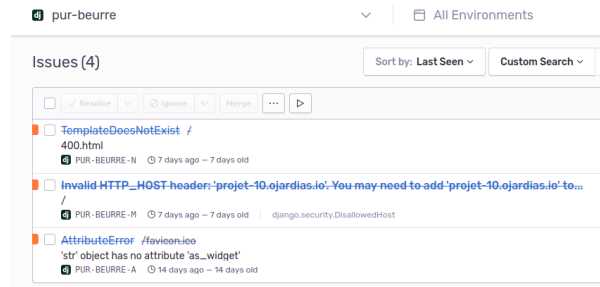


Figure 7: Dashboard de Sentry

- 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-fra1-01
 Disk Utilization is running high Disk Utilization is above 70% for 5 min	 ubuntu-s-1vcpu-1gb-fra1-01
 Memory Utilization is running high Memory Utilization is above 70% for 5 min	 ubuntu-s-1vcpu-1gb-fra1-01

Figure 8: Monitoring Digital Ocean

III. Bilan

III.1. NGINX et Supervisor

La combinaison de ces deux outils permet de mettre en place très rapidement avec une configuration minimaliste un serveur fonctionnel. Cependant, j'ai dû faire face à deux difficultés pour la configuration de ces deux services.

Je souhaitais héberger d'autres applications sur ce serveur, mon CV en ligne (<https://guillaume.ojardias.io>) par exemple. Après quelques tâtonnements et essais infructueux, j'ai fini par comprendre qu'il suffisait simplement de rediriger chaque application sur un port local unique. Raison pour laquelle l'application du projet 8 est redirigée vers le port 8001 (et non 8000 par défaut).

Ensuite, je tenais à mettre en place des certificats SSL. Après un premier essai catastrophique, dans la panique j'ai complètement réinitialisé mon serveur à zéro, j'ai fini par comprendre qu'il me manquait une étape cruciale. Modifier les `Nameservers` de mon fournisseur de nom de domaine vers ceux de Digital Ocean, j'ai fini par réussir à avoir une configuration qui fonctionne.

III.2. Travis, Sentry et Monitoring

Même si ces trois outils n'ont pas forcément de rapport les uns avec les autres, je me suis permis de les regrouper. En effet, je n'ai pas eu de difficultés particulière quand à leur utilisation (dans le cadre de mon stage, nous utilisons CircleCi et Sentry).

Sentry m'a permis toutefois de mettre en avant un bug non géré est embêtant sur mon application du projet 8. Un bug sur les templates des erreurs 4xx et 5xx. Et j'ai pu ainsi apporter un correctif.

III.3. Mergify

Pour finir, un dernier mot sur l'utilisation de Mergify pour compléter le processus de Continuous Integration. J'ai découvert cet outil par le biais de mon stage chez l'éditeur Mergify. Travaillant dessus et l'utilisant au quotidien, je ne pouvais pas mettre un système de pull-request sur mon repository sans au final utilisé ce plugin permettant de merger mes pull-requests une fois les conditions paramétrées remplies.

Dans le cadre de ce projet, la pull-request est mergée dans la branche master une fois que les build Travis ont le statut `success` et sans review à partir du moment où je suis l’auteur de la pull-request.

III.4 Conclusion

Pour conclure, je suis très content des sujets que j’ai pu aborder au cours de ce projet. C’est en quelque sorte la “dernière” brique d’un projet permettant de gérer un projet de sa conception à sa mise en production en passant par sa réalisation.

La dernière chose que j’aimerais faire pour améliorer mon workflow et être dans une dynamique de Continuous Integration / Continuous Deployment, c’est de mettre en place un outil permettant de déployer automatiquement mes applications lorsque la branche master du repository est mise à jour.