Installer, configurer et gérer le sous-domaine "app-monitor"

Notes :

- Ce domaine hébergera l'outil *GlitchTip* permettant de surveiller les erreurs générées par les applications.
- Il est embarqué dans un container *Docker* et fait partie de la *stack app-monitor*.
- Le serveur *bkp-srv* devrait utiliser un container *nginx-proxy* comme serveur web pour tous les outils hébergés dessus.
- *GlitchTip* est un outil plus simple que *Sentry* et demandant moins de ressource. Il se base sur le SDK de Sentry. Il est donc parfaitement compatible avec la méthode d'installation de *Sentry* pour *GeoNature*.

Ressources :

- Missing librairy sentry_sdk

Installer la stack Docker Compose "app-monitor"

- Se connecter au serveur *bkp-srv* sur l'utilisateur *admin*
- Assurez vous que le serveur *bkp-srv* possède un container *nginx-proxy* utilisé comme serveur Nginx global de ce serveur. Pour cela, vérifier la présence du dossier `~/proxy/`. S'il n'existe pas, mettre en place cette "stack" en vous basant sur l'installation du SINP PACA.
- Mettre à jour à l'aide Git ou rsync le dossier `~/docker/app-monitor/`
- Se placer dans le dossier le dossier `~/docker/app-monitor/` avec : `cd ~/docker/app-monitor/`
- Créer un fichier .env à l'aide du fichier .env.sample.
  - Créer des mots de passes (le stocker dans Keepass) et mettre à jour les paramètres (<*domaine-sinp*>)
- Créer une nouvelle entrée dans la zone DNS via l'interface d'OVH :

```
app-monitor              IN CNAME  bkp-srv
```

  - Assurez vous d'avoir une entrée du type :

```
bkp-srv                  IN A      <ipv4-de-bkp-srv>
```

- Lancer le container :

```
docker compose up
```

  - Vérifier que vous accédez à l'interface de connexion : https://app-monitor.<domaine-sino>
  - Si tout semble fonctionnel, l'arrêter et le relancer en tant que service : CTRL+C puis

```
docker compose up -d
```

- Créer un super utilisateur :

```
docker compose run --rm app-monitor-migrate ./manage.py createsuperuser
```

    - Utiliser l'email de l'admin système
    - Stocker le mot de passe dans Keepass

Mise à jour de Glitchtip

Mise à jour de version majeure

- Toujours passer par la dernière version de la branche précédente avant de migrer vers la dernière version de la nouvelle branche majeure.
- Vérifier les changements majeurs indiqués dans les notes de *release*.
- Sinon, les étapes sont les mêmes que pour les mises à jour de version mineure.

Mise à jour de version mineure

- Faire un dump de la base : `docker exec -t app-monitor-postgres pg_dump -U postgres glitchtip -F c > backups/2026-09-16_glitchtip_backup.dump`
- Mettre à jour les versions de l'outil dans le fichier `docker-compose.yml` dans les différents services où l'image est utilisée. Ex. : `image: glitchtip/glitchtip:v4.2.10`
    - Puis lancer :

```
docker compose pull
docker compose down
docker compose run --rm app-monitor-migrate ./manage.py migrate
docker compose up -d
```

- Lors de l'application des migrations de l'outil avec :

```
docker compose run --rm app-monitor-migrate ./manage.py migrate
```

    - En cas d'erreur

```
Error response from daemon: manifest for
glitchtip/glitchtip:v6.2.6 not found: manifest unknown: manifest
unknown
```

      , le problème vient surement du numéro de version inexistant ou ayant changé de format (ici `v6.2.6` a changé pour `6.2.6`). Vous pouvez aussi utiliser `latest`.

    - Si cela ne fonctionne pas, il est aussi possible de se connecter sur le service `app-monitor-web` avec :

```
docker exec -it app-monitor-web /bin/bash
```

        - Lancer la migration avec :

```
./manage.py migrate
```

- En cas d'erreur

```
django.db.utils.ProgrammingError: relation
"performance_transactionevent" already exists
```

:

- Lancer la commande :

```
./manage.py migrate --fake-initial
```

- Tenter la migration en tant que root :

```
docker exec --user root -it app-monitor-web /bin/bash
```

- Lancer la commande :

```
./manage.py migrate
```

- Mettre à jour les fichiers statics (**en tant que root**) :

```
docker exec --user root -it app-monitor-web /bin/bash
```

- Lancer la commande :

```
./manage.py collectstatic
```

Migration de Glitchtip v4 à v5

- Commencer par mettre à jour le docker compose vers la dernière version de Glitchtip de la branche 4, soit : `4.2.10`
- Effectuer les commandes classiques de migration indiquées ci-dessus et vérifier que l'outil fonctionne.
- Mettre à nouveau à jour le docker compose vers la dernière version de Glitchtip de la branche 5.0.x, soit : `5.0.9`
- Effectuer à nouveau les commandes de migrations mais remplacer la commande de migration par : `docker compose run –rm app-monitor-migrate ./manage.py migrate –fake-initial`
  - Les migrations de la base de données ayant été "squashées", il est obligatoire d'ajouter le paramètre : `--fake-initial`
- Vérifier que l'outil fonctionne.
- Enfin, mettre à nouveau à jour le docker compose vers la dernière version de Glitchtip `latest` ou `6.2.6` (en septembre 2026).

Créer un nouveau projet de surveillance sur GlitchTip

- Se connecter sur le site : https://app-monitor.<domaine-sinp>

    - Utiliser le login et le mot de passe du super utilisateur créé précédemment
- Créer une nouvelle organisation :
    - pour SINP PACA : Silene
    - pour SINP AURA : Biodiv'AURA
- Créer un nouveau projet :
    - Plateforme : Flask
    - Nom : GeoNature
    - Team : Equipe-SINP
- Copier le DSN qui s'affiche sur la droite de l'interface du nouveau projet.
- Se rendre sur le serveur *web-srv* en tant que *geonat* :

```
ssh geonat@web-<region>-sinp
```

- Activer le venv de GeoNature : `source ~/geonature/backend/venv/bin/activate`
- Installer le paquet Python Sentry SDK pour Flask : `pip install --upgrade sentry-sdk[flask]`
- Désactiver le venv de GeoNature : `deactivate`
- Éditer le fichier de configuration de GeoNature :

```
vi ~/geonature/config/geonature_config.toml
```

    - Ajouter une entrée au début du fichier :

```
# Set Sentry DSN
SENTRY_DSN = "<coller-ici-le-DSN-du-projet-GeoNature-copier-dans-GlitchTip>"
```

- Modifier le fichier des routes *commons* : `vi /home/geonat/geonature/backend/geonature/core/gn_commons/routes.py` ajouter

```
@routes.route('/debug-glitchtip', methods=["GET"])
def trigger_error():
    division_by_zero = 1 / 0
```

- Redémarrer GeoNature :

```
sudo systemctl restart geonature
```

- Générer une erreur pour tester le fonction en vous rendant dans un navigateur sur l'URL suivante : `https://geonature.<domaine-sinp>/api/gn_commons/debug-glitchtip`
    - Si tout se passe bien, vous devriez voir apparaître l'erreur dans l'interface de *GlitchTip* au niveau du projet créé précédemment.
        - Pensez à supprimer le code ajouté à GeoNature ayant servi à tester la gestion d'erreur avec Glitchtip
    - En cas de problème :
        - Redémarrer GeoNature :

```
systemctl restart geonature
```

        - Redémarrer la stack *app-monitor* :

```
docker compose down ; docker compose up -d
```

- A priori, un second redémarrage résoud l'erreur `[SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: self signed certificate`
- Vérifier les log des containers de la stack *app-monitor* sur : `https://manager.<domaine-sinp>/`

From:
https://sinp-wiki.cbn-alpin.fr/ - **CBNA SINP**

Permanent link:
https://sinp-wiki.cbn-alpin.fr/serveurs/installation/web-srv/docker-glitchtip

Last update: **2026/09/16 10:19**