Manual de Instalación y Configuración para Producción OT N°4

Detalles de la Documentación

Tipo de Documentación:	Documentación Técnica
Cliente:	MITIC
Interesados:	OEE
Autores:	Freelancers del Paraguay
Fecha de Entrega:	08/05/2024

Versiones y Modificaciones

Versión	Fecha	Autor	Modificaciones
1.0	13/03/2024	Freelancers del Paraguay	Inicio
2.0	03/04/2024	Freelancers el Paraguay	Actualización de documentación de instalación y configuración
3.0	08/05/2024	Freelancers el Paraguay	Actualización de documentación de instalación y configuración

Contenido

Detalles del Sistema 2
Implementar GeoNode en un servidor de producción 3
Limitaciones en la Traducción de GeoNode al Español 10
Acceso al sistema 10

Detalles del Sistema

Stack tecnológico

- Python 3.10
- Java 11
- Django 3.2
- PostgreSQL 15.3
- PostGIS 3.3
- GeoServer 2.23
- El servidor webproxy nginx 1.25
- El servidor web HTTP Apache Tomcat 9

Requerimientos mínimos de hardware

- 8 GB de RAM (se prefieren 16 GB o más para una implementación de producción).
- Procesador de 2,2GHz con 4 núcleos. (Es posible que se requiera potencia de procesamiento adicional para múltiples renderizados de estilos simultáneos)
- Uso de disco de software de 30 GB (reservado solo para el sistema operativo y el código fuente).
- 50 a 100 GB en disco adicional para cualquier dato alojado con GeoNode, datos almacenados en la base de datos y mosaicos almacenados en caché con GeoWebCache
- Hardware de 64 bits.

Implementar GeoNode en un servidor de producción

Paso 1 - Configuración de Docker para Ubuntu (22.04)

Instale los siguientes paquetes a nivel de sistema operativo:

```
$ sudo add-apt-repository universe
```

Debe ver una salida similar a la siguiente:

```
root@preprod:/# sudo add-apt-repository universe
Adding component(s) 'universe' to all repositories.
Press [ENTER] to continue or Ctrl-c to cancel.
Obj:1 http://py.archive.ubuntu.com/ubuntu jammy InRelease
Obj:2 http://py.archive.ubuntu.com/ubuntu jammy-updates InRelease
Obj:3 http://py.archive.ubuntu.com/ubuntu jammy-backports InRelease
Obj:4 https://ppa.launchpadcontent.net/ondrej/php/ubuntu jammy InRelease
Obj:5 http://py.archive.ubuntu.com/ubuntu jammy-security InRelease
Obj:6 https://apt.postgresql.org/pub/repos/apt jammy-pgdg InRelease
Leyendo lista de paquetes... Hecho
W: https://ppa.launchpadcontent.net/ondrej/php/ubuntu/dists/jammy/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
```

```
$ sudo apt-get update -y
```

Debe ver una salida similar a la siguiente:

```
root@preprod:/# sudo apt-get update -y
Obj:1 http://py.archive.ubuntu.com/ubuntu jammy InRelease
Obj:2 http://py.archive.ubuntu.com/ubuntu jammy-updates InRelease
Obj:3 https://ppa.launchpadcontent.net/ondrej/php/ubuntu jammy InRelease
Obj:4 http://py.archive.ubuntu.com/ubuntu jammy-backports InRelease
Obj:5 http://py.archive.ubuntu.com/ubuntu jammy-security InRelease
Obj:6 https://apt.postgresql.org/pub/repos/apt jammy-pgdg InRelease
Leyendo lista de paquetes... Hecho
W: https://ppa.launchpadcontent.net/ondrej/php/ubuntu/dists/jammy/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
```

```
$ sudo apt-get install -y git-core git-buildpackage debhelper devscripts python3.10-dev
python3.10-venv virtualenvwrapper
```

Debe ver una salida similar a la siguiente:

```
Running kernel seems to be up-to-date.

The processor microcode seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

```
$ sudo apt-get install -y apt-transport-https ca-certificates curl lsb-release gnupg
gnupg-agent software-properties-common vim
```

Debe ver una salida similar a la siguiente:

```
Running kernel seems to be up-to-date.

The processor microcode seems to be up-to-date.

Restarting services...
 systemctl restart php7.4-fpm.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

A continuación, agregue el repositorio y los paquetes de Docker:

```
$ sudo mkdir -p /etc/apt/keyrings
```

```
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/etc/apt/keyrings/docker.gpg
```

```
$ sudo chmod a+r /etc/apt/keyrings/docker.gpg
```

```
$ echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee
/etc/apt/sources.list.d/docker.list > /dev/null
```

```
$ sudo apt-get update -y
```

Debe ver una salida similar a la siguiente:

```
root@preprod:/# sudo apt-get update -y
Des:1 https://download.docker.com/linux/ubuntu jammy InRelease [48,8 kB]
Des:2 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages [29,1 kB]
Obj:3 http://py.archive.ubuntu.com/ubuntu jammy InRelease
Obj:4 http://py.archive.ubuntu.com/ubuntu jammy-updates InRelease
Obj:5 https://apt.postgresql.org/pub/repos/apt jammy-pgdg InRelease
Obj:6 http://py.archive.ubuntu.com/ubuntu jammy-backports InRelease
Obj:7 https://ppa.launchpadcontent.net/ondrej/php/ubuntu jammy InRelease
Obj:8 http://py.archive.ubuntu.com/ubuntu jammy-security InRelease
Descargados 77,9 kB en 2s (46,2 kB/s)
Leyendo lista de paquetes... Hecho
W: https://ppa.launchpadcontent.net/ondrej/php/ubuntu/dists/jammy/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
```

```
$ sudo apt-get install -y docker-ce docker-ce-cli containerd.io
```

Debe ver una salida similar a la siguiente:

```
Running kernel seems to be up-to-date.

The processor microcode seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

Para finalizar agregue su usuario al grupo de Docker:

```
$ sudo usermod -aG docker ${USER}
```

```
$ su ${USER}
```

Paso 2 – Crear una instancia de geonode-project

Para la siguiente configuración se utilizará el nombre de **my_geonode**, puede modificarlo por el que crea conveniente. A continuación, realice los siguientes pasos:

Cree y acceda al directorio **/opt/docker**:

```
$ sudo mkdir -p /opt/docker
```

```
$ cd /opt/docker
```

Clone el repositorio de **GeoNode**:

```
$ sudo git clone https://github.com/GeoNode/geonode-project.git
```

A continuación, cambie a la rama deseada. Por ejemplo, 4.2.2 para generar un proyecto para GeoNode 4.2.2 (Ultima versión estable)

```
$ cd /opt/docker/geonode-project
```

```
$ sudo git checkout -b 4.2.3
```

Luego prepare y active un entorno virtual **Python** e instale **Django**.

Acceda al siguiente directorio:

```
$ cd /opt/docker
```

A continuación, cree y active el entorno virtual:

```
$ sudo su
```

```
$ python3 -m venv .venvs/my_geonode
```

```
$ source .venvs/my_geonode/bin/activate
```

A continuación debe visualizar el nombre del entorno virtual:

```
(my_geonode) root@preprod:/opt/docker#
```

A continuacion instale Django:

```
$ pip install Django==3.2.13
```

Debe ver una salida similar a la siguiente:

```
(my_geonode) root@preprod:/opt/docker# pip install Django==3.2.13
Collecting Django==3.2.13
  Downloading Django-3.2.13-py3-none-any.whl (7.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7.9/7.9 MB 8.9 MB/s eta 0:00:00
Collecting pytz
  Downloading pytz-2024.1-py2.py3-none-any.whl (505 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 505.5/505.5 KB 10.5 MB/s eta 0:00:00
Collecting sqlparse>=0.2.2
  Downloading sqlparse-0.4.4-py3-none-any.whl (41 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 41.2/41.2 KB 12.8 MB/s eta 0:00:00
Collecting asgiref<4,>=3.3.2
  Downloading asgiref-3.8.1-py3-none-any.whl (23 kB)
Collecting typing-extensions>=4
  Downloading typing_extensions-4.10.0-py3-none-any.whl (33 kB)
Installing collected packages: pytz, typing-extensions, sqlparse, asgiref, Django
Successfully installed Django-3.2.13 asgiref-3.8.1 pytz-2024.1 sqlparse-0.4.4 typing-extensions-4.10.0
```

Por último, cree un nuevo proyecto Django utilizando como plantilla el repositorio de **geonode-project**.

```
$ django-admin startproject --template=./geonode-project -e
py,sh,md,rst,json,yml,ini,env,sample,properties -n monitoring-cron -n Dockerfile
```

```
my_geonode
```

Listando el contenido del directorio debe ver una salida similar a la siguiente:

```
$ ls
```

```
(my_geonode) root@preprod:/opt/docker# ls
geonode-project  my_geonode
```

Paso 3 – Preparar el archivo .env

GeoNode utiliza un archivo .env donde se configuran las variables de entorno requeridas por **Docker** y el propio **GeoNode**. Una forma sencilla de generar un archivo .env básico es utilizar el comando de utilidad **create-envfile.py**. Si se ejecuta sin argumentos, generará un archivo .env válido para una implementación local.

Como hostname se utilizará **https://geoportal.namandu.com/**, de contraseña para los servicios **admin** y como email **soporte@freelancer.com.py**. Recuerde modificar esta información según sus datos de su instalación.

Acceda al directorio **my_geonode** creado con el comando anterior y genere el archivo **.env**:

```
$ cd /opt/docker/my_geonode
```

```
$ python create-envfile.py --https --hostname geoportal.namandu.com --env_type prod
--email soporte@freelancer.com.py --geonodepwd admin --geoserverpwd admin --pgpwd admin
--dbpwd admin --geodbpwd admin
```

Si desea modificar alguna de las variables, el comando acepta los siguientes argumentos:

- **https**: Habilitar SSL. Está deshabilitado por defecto.
- **env_type**:
  - Cuando se establece en prod, DEBUG está deshabilitado y se solicita la creación de un SSL válido al servidor ACME de Letsencrypt.
  - Cuando se establece en test, DEBUG está deshabilitado y se genera un certificado SSL de prueba para realizar pruebas locales.
  - Cuando se establece en dev, DEBUG está habilitado y no se genera ningún certificado SSL.
- **hostname**: La URL que servirá a GeoNode (localhost por defecto).
- **email**: El correo electrónico del administrador. Tenga en cuenta que se requiere un correo electrónico real y configuraciones SMPT válidas si env_type está configurado en prod. Letsencrypt utiliza el correo electrónico para emitir el certificado SSL.
- **geonodepwd**: Contraseña de administrador de GeoNode. Se establece un valor aleatorio si se deja vacío.
- **geoserverpwd**: Contraseña de administrador de Geoserver. Se establece un valor aleatorio si se deja vacío.
- **pgpwd**: Contraseña de PostgreSQL. Se establece un valor aleatorio si se deja vacío.

- **dbpwd**: Contraseña de Usuario de GeoNode DB. Se establece un valor aleatorio si se deja vacío.
- **geodbpwd**: Contraseña de Usuario de Geo DB. Se establece un valor aleatorio si se deja vacío.
- **clientid**: ID de cliente del cliente GeoNode Oauth2 de Geoserver. Se establece un valor aleatorio si se deja vacío.
- **clientsecret**: clave secreta del cliente GeoNode Oauth2 de Geoserver. Se establece un valor aleatorio si se deja vacío.

Obs: La variable **geonodepwd** define la contraseña del administrador de Geonode, sin embargo, hasta la última versión a la fecha esta variable no es tenida en cuenta debido a un error del sistema, por lo cual se utiliza el valor **admin** como contraseña por defecto.

Para configurar su servidor de correo, edite el archivo .env y modifique las siguientes variables:

```
$ nano .env
```

```
# EMAIL Notifications
EMAIL_ENABLE=True
DJANGO_EMAIL_BACKEND=django.core.mail.backends.smtp.EmailBackend
DJANGO_EMAIL_HOST=localhost
DJANGO_EMAIL_PORT=25
DJANGO_EMAIL_HOST_USER=my_user
DJANGO_EMAIL_HOST_PASSWORD=my_password
DJANGO_EMAIL_USE_TLS=False
DJANGO_EMAIL_USE_SSL=True
DEFAULT_FROM_EMAIL='Geoportal <geoportal@localhost.com>'
```

A continuación, se detallan las variables que debe modificar para el correcto funcionamiento:

EMAIL_ENABLE: Desactiva o activa el sistema de envío de correos electrónicos.
DJANGO_EMAIL_HOST: Dirección del servidor de correo electrónico SMTP.
DJANGO_EMAIL_PORT: Puerto utilizado por el servidor de correo electrónico SMTP.
DJANGO_EMAIL_HOST_USER: Nombre de usuario para la autenticación en el servidor de correo electrónico SMTP.
DJANGO_EMAIL_HOST_PASSWORD: Contraseña para la autenticación en el servidor de correo electrónico SMTP.
DJANGO_EMAIL_USE_TLS: Indica si se debe usar TLS (Transport Layer Security) para la conexión de correo electrónico.
DJANGO_EMAIL_USE_SSL: Indica si se debe usar SSL (Secure Sockets Layer) para la conexión de correo electrónico.
DEFAULT_FROM_EMAIL: Dirección de correo electrónico predeterminada para usar como remitente en los correos electrónicos enviados.

Tenga en cuenta que para las variables **DJANGO_EMAIL_USE_TLS** y **DJANGO_EMAIL_USE_SSL** son mutuamente excluyentes, por lo que no puede configurarlas como verdaderas al mismo tiempo.

A continuación, modifique la siguiente variable para que el administrador del sistema apruebe el acceso de los usuarios registrados:

```
# Users Registration
ACCOUNT_OPEN_SIGNUP=True
ACCOUNT_EMAIL_REQUIRED=True
ACCOUNT_APPROVAL_REQUIRED=True
ACCOUNT_CONFIRM_EMAIL_ON_GET=False
ACCOUNT_EMAIL_VERIFICATION=none
ACCOUNT_EMAIL_CONFIRMATION_EMAIL=False
ACCOUNT_EMAIL_CONFIRMATION_REQUIRED=False
ACCOUNT_AUTHENTICATION_METHOD=username_email
AUTO_ASSIGN_REGISTERED_MEMBERS_TO_REGISTERED_MEMBERS_GROUP_NAME=True
```

Para modificar la dirección de correo electrónico que se agrega al final de la página de restablecimiento de contraseña, agregue la siguiente línea especificando su correo electrónico de contacto:

```
THEME_ACCOUNT_CONTACT_EMAIL=soporte@ejemplo.com
```

Paso 4 – Construir y ejecutar contenedores con Docker

Los siguientes comandos compilan las imágenes de los servicios, y luego, lanzan los contenedores en un modo desacoplado, permitiendo que los servicios corran en segundo plano.

```
$ sudo docker compose build
```

```
$ sudo docker compose up -d
```

Debe ver una salida similar a la siguiente:

```
[+] Running 9/9
 ✔Container rabbitmq4my_geonode     Running
 ✔Container gsconf4my_geonode       Healthy
 ✔Container db4my_geonode           Healthy
 ✔Container django4my_geonode       Healthy
 ✔Container celery4my_geonode       Running
 ✔Container geoserver4my_geonode    Running
 ✔Container nginx4my_geonode        Running
 ✔Container memcached4my_geonode    Running
 ✔Container letsencrypt4my_geonode  Started
```

Obs.: La siguiente advertencia se debe a que el proyecto mantiene soporte para **docker-compose**, el cual sí utiliza el campo **version**

```
(my_geonode) root@preprod:/opt/docker/my_geonode# sudo docker compose up -d
WARN[0000] /opt/docker/my_geonode/docker-compose.yml: `version` is obsolete
```

En caso de que se presente algún error, reinicie su sesión e intente nuevamente con los siguientes comandos:

```
$ sudo docker compose down
```

```
$ sudo docker compose build
```

```
$ sudo docker compose up -d
```

Limitaciones en la Traducción de GeoNode al Español

Actualmente, la traducción a idioma español y mantenimiento depende del aporte voluntario de una comunidad global y debido a las actualizaciones constantes del sistema que introducen nuevos elementos que necesitan traducción, resulta en un desfase entre el contenido en inglés y sus versiones en español. Por lo que es necesario realizar de forma particular las traducciones necesarias en el diccionario del sistema y su funcionamiento en la ejecución del mismo.

Traducción de las notificaciones por correo

Como alternativa para que las notificaciones enviadas por el sistema estén español, puede realizar las traducciones de forma manual accediendo al contenedor de **geonode** y modificando las plantillas de correo.

Ejecute el siguiente comando para listar los contenedores e identifique el que pertenece a **django**

```
$ docker ps
```

Debe ver una salida similar a la siguiente

```
(my_geonode) root@preprod:/opt/docker/my_geonode# docker ps
CONTAINER ID   IMAGE                                   COMMAND                  CREATED       STATUS                 PORTS                                                                                   NAMES
b5b482e40B3B   my_geonode/geoserver:2.24.3-latest      "/usr/local/tomcat/t…"   5 hours ago   Up 5 hours (healthy)   0.0.0.0:8080->8080/tcp, :::8080->8080/tcp                                               geoserver4my_ge
onode
27cdb3ea4816   my_geonode/geonode:master               "/usr/src/my_geonode…"   5 hours ago   Up 5 hours             8000/tcp                                                                                celery4my_geono
de
fd26b86f5ebb   my_geonode/geonode:master               "/usr/src/my_geonode…"   5 hours ago   Up 5 hours (healthy)   8000/tcp                                                                                django4my_geono
de
84865c32b2a7   memcached:alpine                        "docker-entrypoint.s…"   5 hours ago   Up 5 hours (healthy)   11211/tcp                                                                               memcached4my_ge
onode
06577f43133f   my_geonode/letsencrypt:2.6.0-latest     "./docker-entrypoint…"   5 hours ago   Up 4 hours                                                                                                     letsencrypt4my_
geonode
00602c0b7d78   my_geonode/nginx:1.25.3-latest          "/docker-entrypoint.…"   5 hours ago   Up 5 hours             0.0.0.0:80->80/tcp, :::80->80/tcp, 0.0.0.0:443->443/tcp, :::443->443/tcp   nginx4my_geonod
e
a4f97d9df888   my_geonode/geoserver_data:2.24.3-latest "sleep infinity"         5 hours ago   Up 5 hours (healthy)                                                                                           gsconf4my_geono
de
20f1711e5546   my_geonode/postgis:15.3-latest          "docker-entrypoint.s…"   5 hours ago   Up 5 hours (healthy)   5432/tcp                                                                                db4my_geonode
6844c1689b60   rabbitmq:3-alpine                       "docker-entrypoint.s…"   5 hours ago   Up 5 hours             4369/tcp, 5671-5672/tcp, 15691-15692/tcp, 25672/tcp                                     rabbitmq4my_geo
node
(my_geonode) root@preprod:/opt/docker/my_geonode#
```

En este ejemplo el **CONTAINER_ID** de django es **fd26b86f5ebb** con el cual se puede acceder al contenedor

```
$ docker exec -it fd26b86f5ebb /bin/bash
```

Debe visualizar su terminal de forma similar al siguiente ejemplo

```
(my_geonode) root@preprod:/opt/docker/my_geonode# docker exec -it fd26b86f5ebb /bin/bash
root@fd26b86f5ebb:/usr/src/my_geonode#
```

A continuación acceda al siguiente directorio, en el cual encontrara los archivos que debe modificar para realizar la traducción

```
$ cd /usr/src/geonode/geonode/templates/account/email
```

```
$ ls
```

A continuación aparecerán varios archivos .txt que corresponden a los distintos mensajes que son enviados a través del correo.

```
root@fd26b86f5ebb:/usr/src/my_geonode# cd /usr/src/geonode/geonode/templates/account/email
root@fd26b86f5ebb:/usr/src/geonode/geonode/templates/account/email# ls
email_confirmation_message.txt  invite_user.txt          password_change.txt          password_reset_key.txt
email_confirmation_subject.txt  invite_user_subject.txt  password_change_subject.txt  password_reset_key_subject.txt
```

Estos archivos deben traducirse al español, guardar y automáticamente aparecerá la traducción sin necesidad de reiniciar el sistema.

Acceso al sistema

Una vez finalizado al acceder al **hostname** configurado anteriormente, para este ejemplo **https://geoportal.namandu.com/**, debe visualizar la página de inicio de la aplicación.

FREELANCERS DEL PARAGUAY S.A.

Firmado digitalmente por JUAN CARLOS LOPEZ AGUAYO
Fecha: 2024.05.08 17:34:02 -04'00'