**MANUAL COMPLETO DE INSTALACIÓN DEL AMBIENTE DE CNCFLORA PARA EL INSTITUTO ALEXANDER VON HUMBOLDT**

VERSION 1.0 – 01-12-2016

Elaborado Por: Jaime Alberto Gutiérrez Mejía

# ÍNDICE

[ÍNDICE 2](#_Toc472592334)

[1. OBJETO DEL DOCUMENTO 3](#_Toc472592335)

[2. PARTICIPANTES 3](#_Toc472592336)

[3. OBJETIVOS 3](#_Toc472592337)

[4. INSTALACIÓN DE AMBIENTE VAGRANT, GITSHELL Y LÍNEA BASE DE PYTHON EN LA MÁQUINA VIRTUAL LINUX UBUNTU 14.04 DE LA DISTRIBUCIÓN 4](#_Toc472592338)

[5. ACTUALIZACIÓN DE APACHE PARA LA CONFIGURACIÓN DEL PROXY DEL SERVIDOR 20](#_Toc472592339)

[6. INSTALACIÓN DE DOCKER PARA EL CLÚSTER DE APLICACIONES APP NUVEM CNCFLORA 25](#_Toc472592340)

[7. ARRANQUE DEL AMBIENTE DE FRONT END PORTAL NUVEM CNCFLORA 37](#_Toc472592341)

# OBJETO DEL DOCUMENTO

Este documento tiene como objeto mostrar una visión del panorama tecnológico de la aplicación CNCFLORA PORTAL y del clúster de aplicaciones NUVEM de la solución, con el fin de conocer cómo descomprimir la paquetería, instalarla y poder ponerla a punto.

# PARTICIPANTES

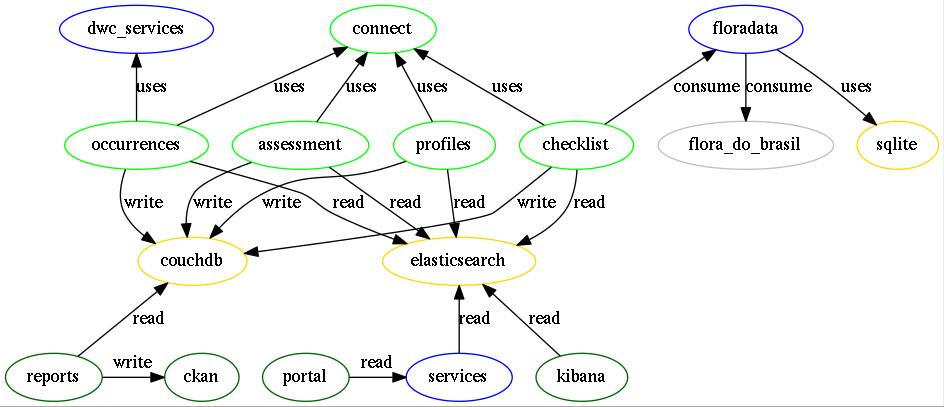
En este proceso tiene principal participación el Administrador Técnico del Sistema, el jefe de gestión de aplicaciones del Instituto Alexander von Humboldt y los participantes del proceso de despliegue y paso a producción.

# OBJETIVOS

1. El objetivo de este proceso, es asegurar el montaje del ambiente de virtualización
2. Configurar la paquetería base Linux del Sistema Operativo Host de la solución
3. Habilitar las extensiones técnicas para el correcto funcionamiento de todos los módulos
4. Poner la solución a punto y ponerla a funcionar en el ambiente virtualizado escogido

# CONTEXTO PARA EL DESARROLLO DE LA INSTALACIÓN

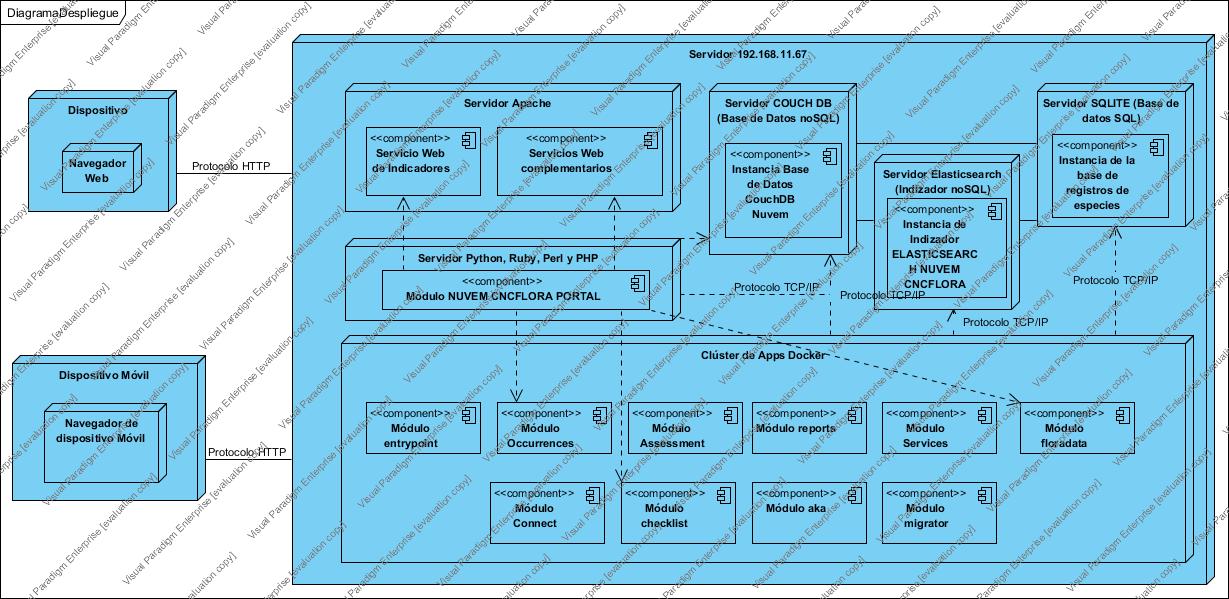
El Sistema de Gestión de Registros Biológicos de NUVEM CNCFLORA es una colección de microservicios y aplicaciones subsidiarias, integradas entre sí, de acuerdo al siguiente modelo arquitectónico:



Básicamente, el Sistema está compuesto por un conjunto de módulos y aplicaciones que interactúan entre sí y alimentan información a bases de datos no relacionales SQL para el proceso de registro, transformación y análisis de registros taxonómicos, fundamentales para el proceso de seguimiento de análisis de riesgo de extinción de especies. Entre los principales módulos, encontramos los siguientes:

1. **Flora do brasil:** Fuente taxonómica autoritativa. Expone un IPT (Kit integrado de herramientas de publicación) con sus datos.
2. **Floradata:** Consume el IPT de Flora do Brasil en una base de datos SQLite más adecuada, y expone los nombres a través de una API de webservices REST en la taxonomía.
3. **DWC-Services:** Biblioteca y API para manejar las operaciones principales en estándar Darwincore: Conversión entre formatos de archivo (xlsx, csv, dwc-a, json, geojson), validación, solución de problemas comunes y ejecución de algunos análisis.
4. **Connect:** Componente de autenticación único (Single Sign On) para todas las aplicaciones.
5. **CouchDB:** Base de datos basada en REST y libre de esquema para manipulación de documentos. Óptima en procesos de persistencia, replicación y reducción del mapa de riesgos para el tratamiento de los datos. Se utiliza como almacenamiento principal.
6. **ElasticSearch:** Servidor de indexación y búsqueda.
7. **Checklist:** Módulo que permitie el agrupamiento de las especies en una lista de control, la creación de la base de datos en couchdb y el índice en elasticsearch.
8. **Occurrences:** Manipula los datos de ocurrencias. Permite cargar, insertar, georeferenciar, buscar por especies con sinónimos (de lista de verificación), validación y análisis.
9. **Profiles:** Perfil de la especie con datos ecológicos, con flujo de trabajo para la creación, validación y revisión.
10. **Assessments:** Flujo de trabajo para la evaluación de riesgos.
11. **Entrypoing:** Módulo único de especificación con enlaces para las diferentes aplicaciones en diferentes listas de verificación.
12. **Services:** API de publicación de Web Services RESTFull.
13. **Reports:** Generar archivos CSV (y un archivo SQLite) con los datos para su posterior publicación en CKAN.
14. **CKAN:** Herramienta de publicación de datos amigable para el usuario.
15. **Portal:** Portal Web Unificado con la capacidad de consulta basada en Web Services de la Lista Roja de Especies Amenazadas.
16. **Dockers:** Base contenedores de base, especialmente: Apache, Ruby, Java8, ElasticSearch, CouchDB, Kibana y Selenio.
17. **Docker Proxy:** Proxy NGINX dinámico para los contenedores del acoplador.
18. **Nuvem:** Proyecto utilizando docker-compose para ejecutar todas las aplicaciones.

Todos estos módulos permiten la definición de dos ambientes claramente diferenciados, que a nivel de arquitectura, se presentan en el siguiente diagrama UML de despliegue de componentes:

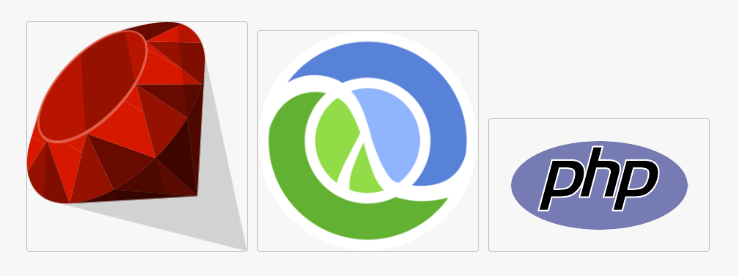


En el cual se integran los siguientes módulos a saber:

1. Un servidor de Sistema Operativo Linux UBUNTU LTS 16.04 (14.04 LTS en su defecto), recomendado para la integración del proceso de virtualización con OVM y vagrant, el cual se recomienda sea una máquina con las siguientes características mínimas:

|  |  |  |
| --- | --- | --- |
| REQUERIMIENTOS DE INFRAESTRUCTURA (Si es requerido) | | |
| Dominio | http://sniare.humboldt.org.co/ | |
| Subdominio | <http://sniare.humboldt.org.co/portal> | |
| Storage | 160 GB | |
| No. Cores | 8 núcleos, dos procesadores | |
| Sistema Operativo | Ubuntu 16.04 LTS | |
| Memoria Ram | 8 GB RAM (MÍNIMA)  16 GB (RECOMENDADA)  32 GB (ÓPTIMA) | |
| Servidor de Aplicaciones | Apache 2.2 con módulo de Ruby y Python 2.7 para LTS 16.04 | |
| Lenguaje de Desarrollo | **PHP** | **PHP 5.5.9-1ubuntu4.19 (cli) (built: Jul 28 2016 19:31:33)** |
| **PYTHON** | **2.7.6 o superior** |
| **RUBY ON RAILS** | **Ruby 1.9.3p484 (2013-11-22 revision 43786) [x86\_64-linux]** |
| Base de Datos: | ELASTICSEARCH | Versión 1.4.4  BuildTimestamp: 2015-02-19T13:05:36Z  Lucene Version: 4.10.3 |
| COUCHDB | 14.04 |
| SQLITE | SQLite version 3.8.2 2013-12-06 14:53:30 |
| Enrutamiento | Por concepto del componente estático de la solución Web contenido en la carpeta static del dominio portal, se requiere que la carpeta sea mapeable y visible por el Servidor Web Apache, con el fin de que las URI del aplicativo sean visibles al contexto /portal/static/estructura.  La carpeta presenta la siguiente estructura de contenido:  static  El enrutamiento debe ser relativo igualmente al contexto, para que cuando se resuelva de forma externa, se pueda acceder al dominio portal, ingresando directamente la URI <http://sniare.humboldt.org.co/> | |
| REQUERIMIENTOS DE SEGURIDAD | | |
| Puertos | |  |  | | --- | --- | | **Servicio Nuvem Clúster** | **Puerto de Resolución** | | proxy | 80 | | 443 | | Base de Datos COUCHDB | 5985 | | Base de Datos ElasticSearch | 9200 | | Servicios Web Floradata | 80 | | 9001 | | Portal (WebApp principal) | 8889 | | Entrypoint | 80 | | Connect (SSO) | 80 | | DWC Services (Servicios Web) | 80 | | CHECKLIST | 80 | | OCURRENCES (Módulo) | 80 | | profiles (Módulo) | 80 | | ASSESSMENTS (Módulo) | 80 | | MIGRATOR (Módulo) | 80 | | REPORTS2 | 80 | | ARQUIVOS | 80 | | ARQUIVOS-SFTP | 2223 | | |
| Re direccionamiento de Puertos | |  |  |  | | --- | --- | --- | | **Servicio Nuvem Clúster** | **Puerto de Resolución** | **Redireccionamiento del Puerto** | | proxy | 80 | 80 | | 443 | 443 | | Base de Datos COUCHDB | 5985 | 5985 | | Base de Datos ElasticSearch | 9200 | 9200 | | Servicios Web Floradata | 80 | 80 | | 9001 | 9001 | | Portal (WebApp principal) | 8889 | 8889 | | Entrypoint | 80 | 80 | | Connect (SSO) | 80 | 80 | | DWC Services (Servicios Web) | 80 | 80 | | CHECKLIST | 80 | 80 | | OCURRENCES (Módulo) | 80 | 80 | | profiles (Módulo) | 80 | 80 | | ASSESSMENTS (Módulo) | 80 | 80 | | MIGRATOR (Módulo) | 80 | 80 | | REPORTS2 | 80 | 80 | | ARQUIVOS | 80 | 80 | | ARQUIVOS-SFTP | 2223 | 22 | | | |
| Permisos Carpetas | Se requiere permisos de full control (lecto/escritura) en las siguientes carpetas por módulo:   |  |  | | --- | --- | | **Servicio Nuvem Clúster** | **Carpeta** | | proxy | ./Caddyfile:/etc/Caddyfile  ./index.html:/var/www/html/index.html  /opt/caddy:/root/.caddy " | | | Base de Datos COUCHDB | /var/data/couchdb:/var/lib/couchdb:rw | | Base de Datos ElasticSearch | /var/data/elasticsearch/data:/usr/share/elasticsearch/data:rw | | Servicios Web Floradata | /var/data/aka:/var/data/aka:rw | | | Portal (WebApp principal) | /portal | | Entrypoint | /entrypoint | | Connect (SSO) | /var/data/connect:/var/floraconnect:rw | | DWC Services (Servicios Web) | /dwc\_services | | CHECKLIST | /checklist | | OCURRENCES (Módulo) | /occurrences | | profiles (Módulo) | /profiles | | ASSESSMENTS (Módulo) | /assessment | | MIGRATOR (Módulo) | /migrator | | REPORTS2 | /reports2 | | ARQUIVOS | /var/data/www:/var/www:rw | | ARQUIVOS-SFTP | /var/data/www:$SFTP\_DATA\_DIR/chroot/www:rw | | | |

1. Un servidor de instancia PYTHON-RUBY/CLOJURE/PHP, que permite la ejecución del ambiente de Portal, necesario para la definición del ambiente de FRONT END de la solución. Este servidor se encuentra soportado por los siguientes paquetes:



* **PYTHON/RUBY:** Ruby es un lenguaje de programación interpretado, reflexivo y orientado a objetos, creado por el programador japonés Yukihiro "Matz" Matsumoto, quien comenzó a trabajar en Ruby en 1993, y lo presentó públicamente en 1995. Combina una sintaxis inspirada en Python y Perl con características de programación orientada a objetos similares a Smalltalk. Comparte también funcionalidad con otros lenguajes de programación como Lisp, Lua, Dylan y CLU. Ruby es un lenguaje de programación interpretado en una sola pasada y su implementación oficial es distribuida bajo una licencia de software libre.
* **CLOJURE:** Clojure es un lenguaje de programación de propósito general dialecto de Lisp. Hace un énfasis especial en el paradigma funcional, con el objetivo (entre otros) de eliminar la complejidad asociada a la programación concurrente. Clojure puede ser ejecutado sobre la Máquina Virtual de Java y la máquina virtual de la plataforma .NET, así como compilado a JavaScript. Asimismo, en principio se rechaza la orientación a objetos, ofreciendo un enfoque en el que los programas se expresan como la aplicación de funciones sobre datos, más que como la interacción entre entidades mutables que mezclan representación de datos, y comportamiento. Por otra parte, características tales como instanciabilidad, polimorfismo e interfaces son efectivamente parte del lenguaje.
* **PHP:** PHP es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página web resultante. PHP ha evolucionado por lo que ahora incluye también una interfaz de línea de comandos que puede ser usada en aplicaciones gráficas independientes. Puede ser usado en la mayoría de los servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún costo. Fue creado originalmente por Rasmus Lerdorf en el año 1995. Actualmente el lenguaje sigue siendo desarrollado con nuevas funciones por el grupo PHP.2 Este lenguaje forma parte del software libre publicado bajo la licencia PHP, que es incompatible con la Licencia Pública General de GNU debido a las restricciones del uso del término PHP.

PHP puede ser desplegado en la mayoría de los servidores web y en casi todos los sistemas operativos y plataformas sin costo alguno. El lenguaje PHP se encuentra instalado en más de 20 millones de sitios web y en un millón de servidores. El número de sitios basados en PHP se ha visto reducido progresivamente en los últimos años, con la aparición de nuevas tecnologías como Node.JS, Golang, ASP.NET, etc. El sitio web de Wikipedia está desarrollado en PHP.5 Es también el módulo Apache más popular entre las computadoras que utilizan Apache como servidor web. El gran parecido que posee PHP con los lenguajes más comunes de programación estructurada, como C y Perl, permiten a la mayoría de los programadores crear aplicaciones complejas con una curva de aprendizaje muy corta. También les permite involucrarse con aplicaciones de contenido dinámico sin tener que aprender todo un nuevo grupo de funciones.

1. Servidor de base de Datos, soportado en los siguientes productos:

* **COUCHDB:**  Apache CouchDB, comúnmente llamada CouchDB, es un gestor de bases de datos de código abierto, cuyo foco está puesto en la facilidad de su uso y en ser "una base de datos que asume la web de manera completa".1 Se trata de una base de datos NoSQL que emplea JSON para almacenar los datos, JavaScript como lenguaje de consulta por medio de MapReduce y HTTP como API.1 Una de sus características más peculiares es la facilidad con la que permite hacer replicaciones. CouchDB fue liberada por primera vez en 2005, transformándose en un proyecto Apache en 2008.

A diferencia de una base de datos relacional, CouchDB no almacena los datos y sus relaciones en tablas. En cambio, cada base de datos es una colección de documentos independientes. Cada documento mantiene sus propios datos y su esquema autocontenido. Una aplicación puede acceder a múltiples bases de datos, por ejemplo una residente en el teléfono móvil del usuario y otra residente en un servidor. Los metadatos del documento contienen información acerca de la versión del mismo, permitiendo refundir cualesquiera diferencias que puedan haberse producido mientras las bases de datos estaban desconectadas.

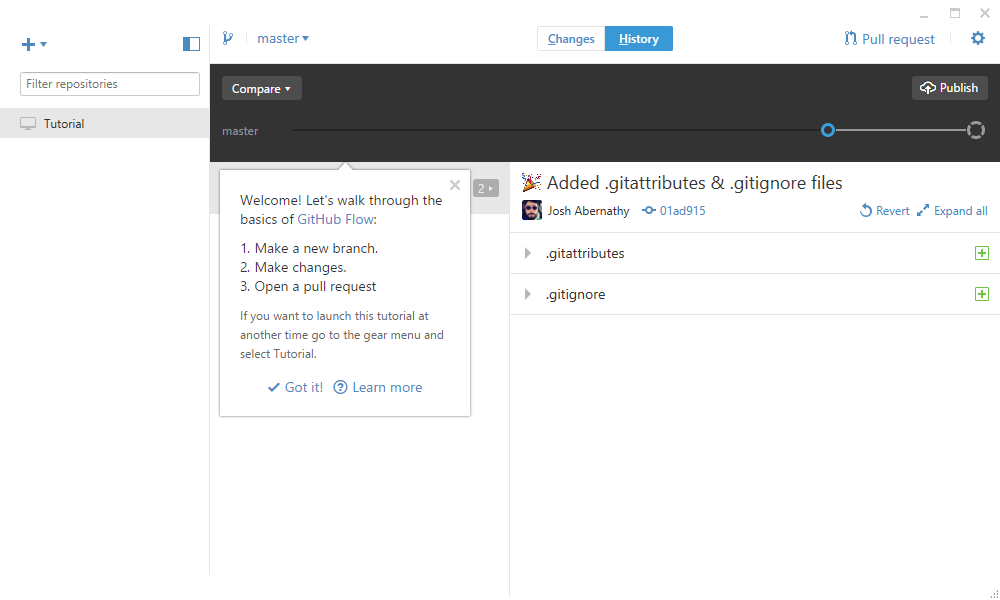
CouchDB implementa una forma de control de concurrencia multiversión (MVCC) a fin de evitar la necesidad de bloquear el archivo de base de datos durante las escrituras. La resolución de conflictos se delega a la aplicación. Para resolver un conflicto generalmente se requiere primero refundir los datos en uno de los documentos para luego eliminar el otro. Otras características son una semántica ACID con consistencia eventual, MapReduce, replicación incremental y tolerancia a los fallos. Las labores de administración se facilitan por medio de una aplicación web incorporada, llamada Futon.

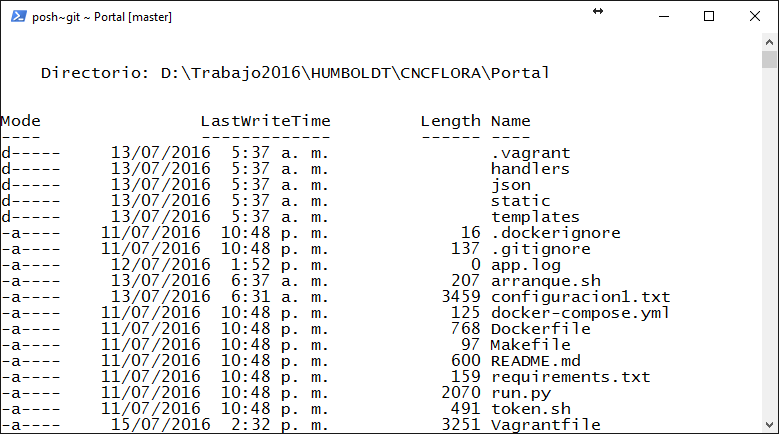
* **ELASTICSEARCH:**  Elasticsearch es un servidor de búsqueda basado en Lucene. Provee un motor de búsqueda de texto completo, distribuido y con capacidad de multi-tenencia con una interfaz web RESTful y con documentos JSON. Elasticsearch está desarrollado en Java y está publicado como código abierto bajo las condiciones de la licencia Apache.
* **SQLITE:** SQLite es un sistema de gestión de bases de datos relacional compatible con ACID, contenida en una relativamente pequeña (~275 kiB)2 biblioteca escrita en C. SQLite es un proyecto de dominio público1 creado por D. Richard Hipp. A diferencia de los sistema de gestión de bases de datos cliente-servidor, el motor de SQLite no es un proceso independiente con el que el programa principal se comunica. En lugar de eso, la biblioteca SQLite se enlaza con el programa pasando a ser parte integral del mismo. El programa utiliza la funcionalidad de SQLite a través de llamadas simples a subrutinas y funciones. Esto reduce la latencia en el acceso a la base de datos, debido a que las llamadas a funciones son más eficientes que la comunicación entre procesos. El conjunto de la base de datos (definiciones, tablas, índices, y los propios datos), son guardados como un sólo fichero estándar en la máquina host. Este diseño simple se logra bloqueando todo el fichero de base de datos al principio de cada transacción.

En su versión 3, SQLite permite bases de datos de hasta 2 Terabytes de tamaño, y también permite la inclusión de campos tipo BLOB. El autor de SQLite ofrece formación, contratos de soporte técnico y características adicionales como compresión y cifrado.

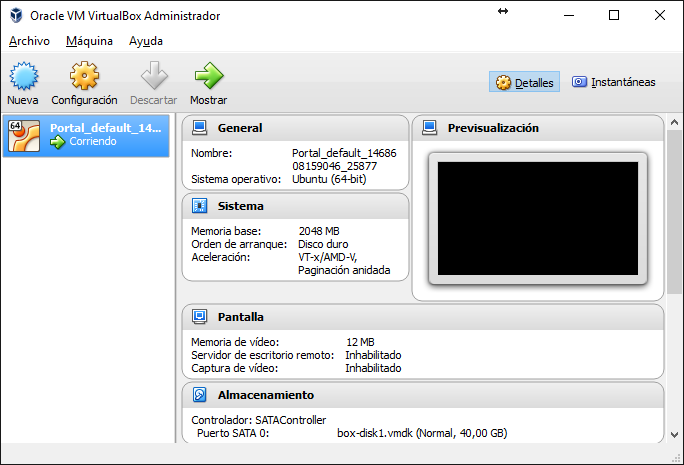
De acuerdo a las recomendaciones dadas por CNCFLORA BRASIL, el proceso de configuración y puesta en marcha de microservicios del ambiente de CLUSTER NUVEM CNCFLORA, requiere del uso de las siguientes herramientas:

1. **GitHubSetup.exe:** GitHub es una forja (plataforma de desarrollo colaborativo) para alojar proyectos utilizando el sistema de control de versiones Git. Utiliza el framework Ruby on Rails por GitHub, Inc. (anteriormente conocida como Logical Awesome). Desde enero de 2010, GitHub opera bajo el nombre de GitHub, Inc. El código se almacena de forma pública, aunque también se puede hacer de forma privada, creando una cuenta de pago.



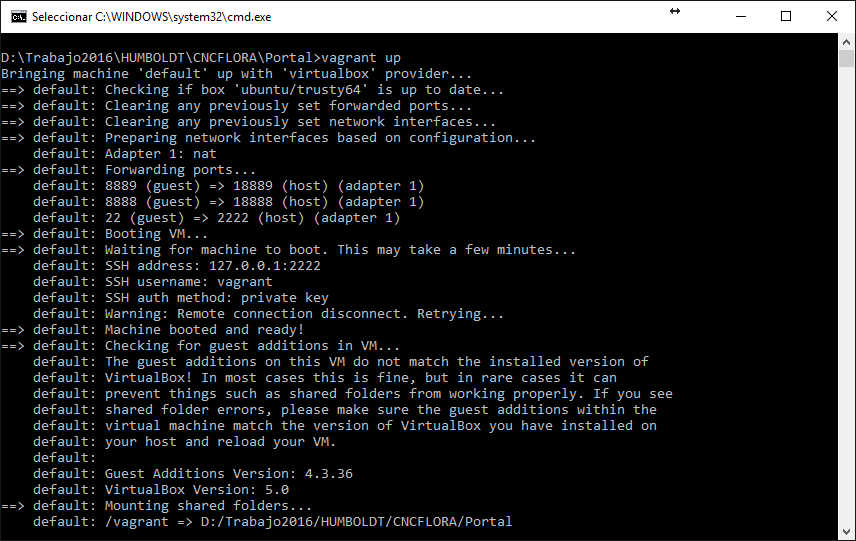


1. **VirtualBox-5.0.24-108355-Win.exe:**  Oracle VM VirtualBox es un software de virtualización para arquitecturas x86/amd64, creado originalmente por la empresa alemana innotek GmbH. Actualmente es desarrollado por Oracle Corporation como parte de su familia de productos de virtualización. Por medio de esta aplicación es posible instalar sistemas operativos adicionales, conocidos como «sistemas invitados», dentro de otro sistema operativo «anfitrión», cada uno con su propio ambiente virtual. Entre los sistemas operativos soportados (en modo anfitrión) se encuentran GNU/Linux, Mac OS X, OS/2 Warp , Microsoft Windows, y Solaris/OpenSolaris, y dentro de ellos es posible virtualizar los sistemas operativos FreeBSD, GNU/Linux, OpenBSD, OS/2 Warp, Windows, Solaris, MS-DOS y muchos otros.



La aplicación fue inicialmente ofrecida bajo una licencia de software privativo, pero en enero de 2007, después de años de desarrollo, surgió VirtualBox OSE (Open Source Edition) bajo la licencia GPL 2. Actualmente existe la versión privativa Oracle VM VirtualBox, que es gratuita únicamente bajo uso personal o de evaluación, y está sujeta a la licencia de "Uso Personal y de Evaluación VirtualBox" (VirtualBox Personal Use and Evaluation License o PUEL) y la versión Open Source, VirtualBox OSE, que es software libre, sujeta a la licencia GPL. VirtualBox ofrece algunas funcionalidades interesantes, como la ejecución de máquinas virtuales de forma remota, por medio del Remote Desktop Protocol (RDP), soporte iSCSI, aunque estas opciones no están disponibles en la versión OSE.

1. **Oracle\_VM\_VirtualBox\_Extension\_Pack-4.3.38-106717.vbox-extpack:** Paquete de extensiones (configuraciones adicionales), que permiten el mejoramiento de las prestaciones de las máquinas virtuales instaladas bajo OVM Virtual Box, permitiendo acceder a las funcionalidades de red, carpetas compartidas y similares. Es fundamental instalar dicho paquete con el fin de garantizar un correcto funcionamiento del ambiente de virtualización.
2. **vagrant\_1.8.X.msi:** Vagrant es una herramienta para la creación y configuración de entornos de desarrollo virtualizados. Originalmente se desarrolló para VirtualBox y sistemas de configuración tales como Chef, Salt y Puppet. Sin embargo desde la versión 1.1 Vagrant es capaz de trabajar con múltiples proveedores, como VMware, Amazon EC2, LXC, DigitalOcean, etc.2 Aunque Vagrant se ha desarrollado en Ruby se puede usar en multitud de proyectos escritos en otros lenguajes, tales como PHP, Python, Java, C# y JavaScript.



1. **Microsoft Visual C++ (vcredist\_x86.exe):** Visual C++ (también conocido como MSVC++, Microsoft Visual C++) es un entorno de desarrollo integrado (IDE) para lenguajes de programación C, C++ y C++/CLI. Visual C++ engloba el desarrollo de aplicaciones hechas en C, C++ y C++/CLI en el entorno Windows. Visual C++ incluye además las bibliotecas de Windows (WinApi), las bibliotecas MFC y el entorno de desarrollo para .NET Framework. Visual C++ cuenta con su propio compilador (de igual nombre) y otras herramientas como IntelliSense, TeamFoundation Server, Debug,... Además provee de bibliotecas propias de cada versión del sistema operativo y sockets. Como otros compiladores, se le pueden añadir nuevas bibliotecas como DirectX, wxWidgets o SDL.

Cuenta con una versión Express, llamada Microsoft Visual C++ Express Edition, la cual es gratuita y se puede descargar desde el sitio de Microsoft. El lenguaje de programación utilizado por esta herramienta, de igual nombre, está basado en C++ y es compatible en la mayor parte de su código con este lenguaje, a la vez que su sintaxis es exactamente igual. En algunas ocasiones esta incompatibilidad impide que otros compiladores, sobre todo en otros sistemas operativos, funcionen bien con código desarrollado en este lenguaje. Este conjunto de paquetes es fundamental para el correcto funcionamiento de la herramienta de configuración de ambientes de desarrollo **vagrant.**

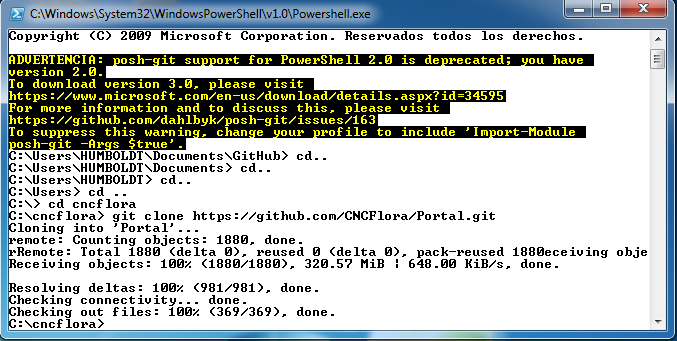
**Nota de versiones de los productos a instalar:**

Considerando el avance permanente de actualización de los productos en cuestión para el desarrollo de las actividades de instalación, no es conveniente establecer en este documento una versión definitiva del producto específico de cada componente, por lo que se establecerá la fuente oficial de descarga de los productos, con el fin de que el responsable del proceso de instalación, acceda a las versiones más recientes de cada uno de ellos:

|  |  |
| --- | --- |
| **PRODUCTO** | **URI DE DESCARGA** |
| ORACLE OVM VIRTUAL BOX | <https://www.virtualbox.org/> |
| LIBRERÍAS DE MICROSOFT VISUAL C++ | <https://www.microsoft.com/es-co/download/details.aspx?id=14632> |
| AMBIENTE DE VIRTUALIZACIÓN VAGRANT | <https://www.vagrantup.com/> |
| AMBIENTE DE GITHUB | <https://desktop.github.com/> |
| PACK DE EXTENSIONES DE ORACLE VIRTUALBOX | <https://www.virtualbox.org/wiki/Downloads> |

# INSTALACIÓN DE AMBIENTE VAGRANT, GITSHELL Y LÍNEA BASE DE PYTHON EN LA MÁQUINA VIRTUAL LINUX UBUNTU 14.04 DE LA DISTRIBUCIÓN

1. Clonar repositorio con el comando git clone <https://github.com/CNCFlora/Portal.git>

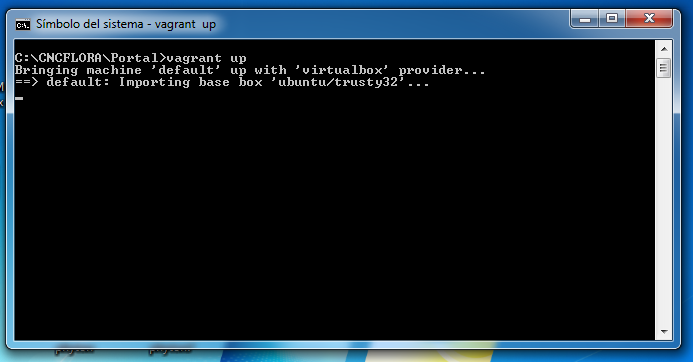
0

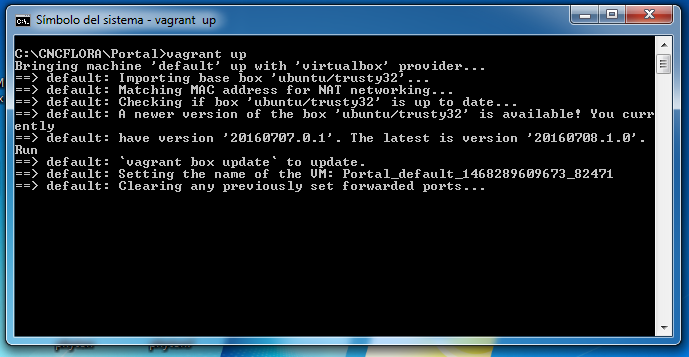
1. Editar el archivo de configuración vagrantfile, con los siguientes parámetros:

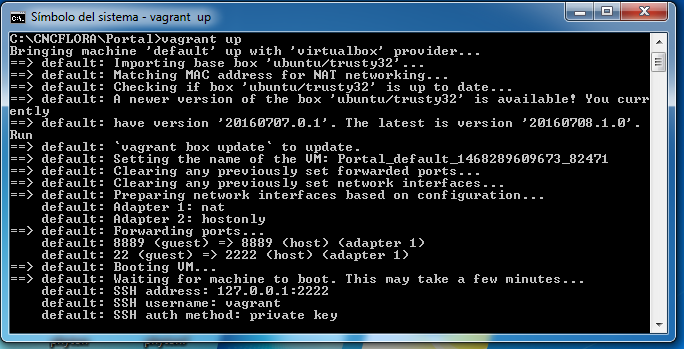
|  |
| --- |
| VAGRANTFILE\_API\_VERSION = "2"  $script = <<SCRIPT  apt-get update  apt-get install -y python-dev  apt-get install -y python-virtualenv  apt-get install -y libfontconfig  SCRIPT  Vagrant.configure(VAGRANTFILE\_API\_VERSION) do |config|  config.vm.box = "ubuntu/trusty64"  config.vm.provider "virtualbox" do |v|  v.memory = 4124  v.cpus = 2  end  config.vm.provision "shell", inline: $script  config.vm.network "forwarded\_port", guest: 80, host: 8080, protocol: "tcp"  config.vm.network "forwarded\_port", guest: 8888, host: 18888, protocol: "tcp"  config.vm.network "forwarded\_port", guest: 8889, host: 18889, protocol: "tcp"  config.vm.network "forwarded\_port", guest: 5984, host: 15984, protocol: "tcp"  config.vm.network "forwarded\_port", guest: 9200, host: 19200, protocol: "tcp"  config.vm.network "forwarded\_port", guest: 9292, host: 19292, protocol: "tcp"  config.vm.network "forwarded\_port", guest: 443, host: 1443, protocol: "tcp"  config.vm.network "private\_network", ip: "192.168.50.200"  config.vm.hostname = "cncfloralocal"  end |

1. Elevar la máquina virtual luego de haber instalado
2. Virtual Box
3. Pack de actualizaciones
4. Microsoft Visual C++ 2010 Redistributable

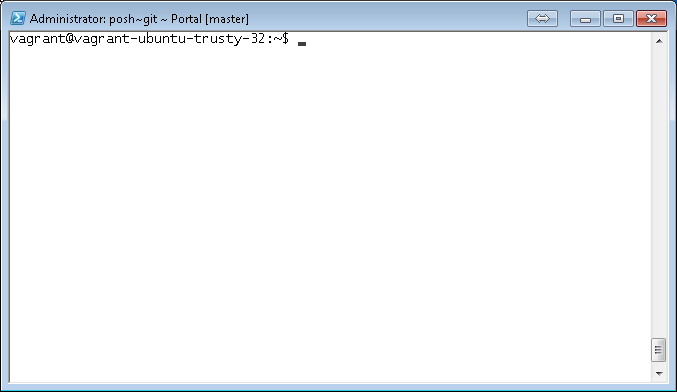
**Comando: vagrant up**

****

****

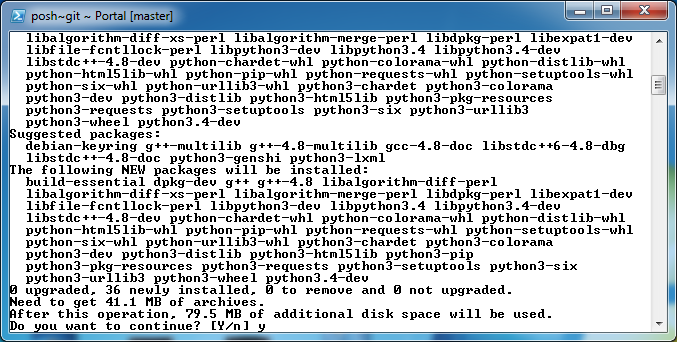
****

1. Una vez la máquina esté arriba, conectarse desde la consola de GitHub con vagrant ssh



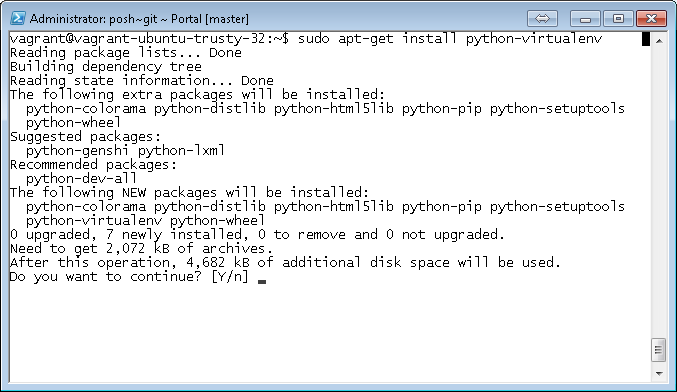
1. Instalar el paquete de PYTHON 3, con el comando

**sudo apt-get install python3-pip**



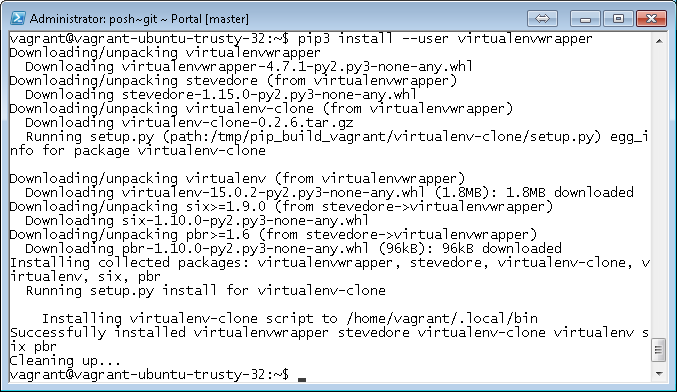
1. Instalar Python virtual env

**sudo apt-get install python-virtualenv**



1. Bajar el virtualenwrapper

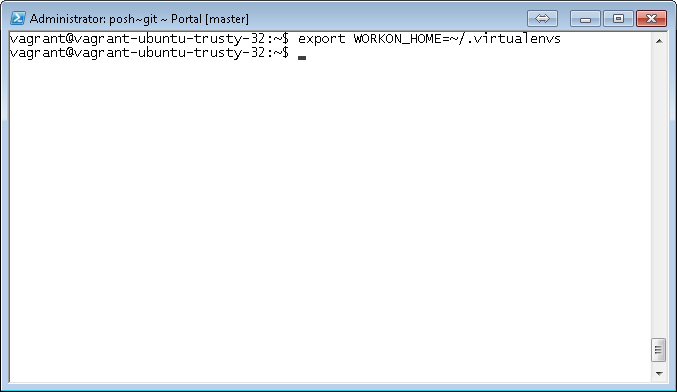
pip3 install --user virtualenvwrapper



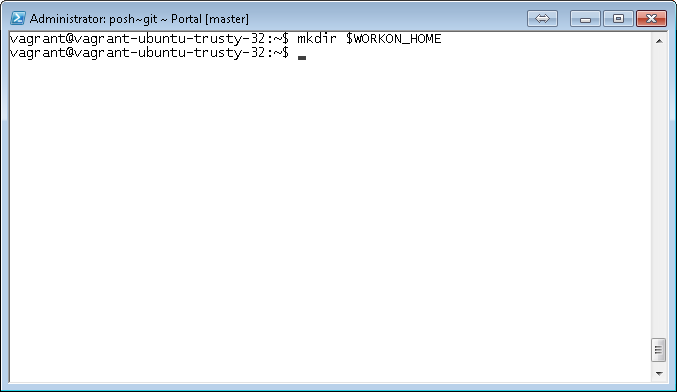
1. Exportar la variable de entorno con

echo "export VIRTUALENVWRAPPER\_PYTHON=/usr/bin/python3" >> ~/.bashrc

1. Exportar la variable export WORKON\_HOME=~/.virtualenvs

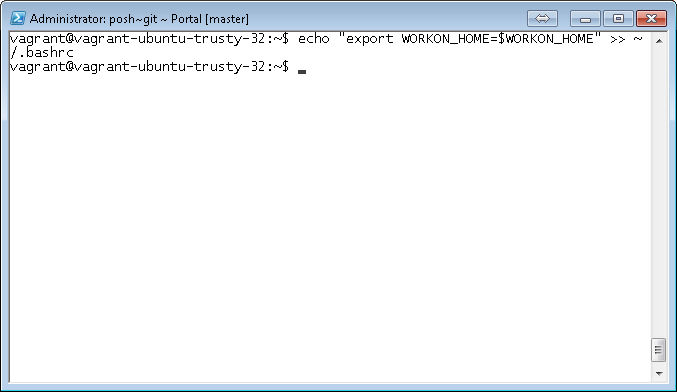


1. Crear el directorio de WORKON\_HOME para la distribución

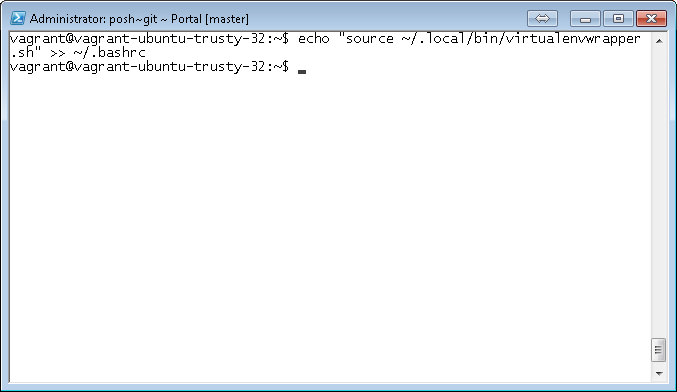


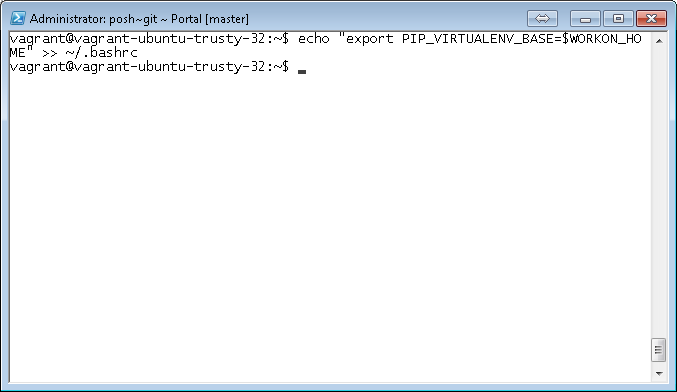
1. Acto seguido se exporta la variable de entorno al Shell, con el comando

**echo "export WORKON\_HOME=$WORKON\_HOME" >> ~/.bashrc**

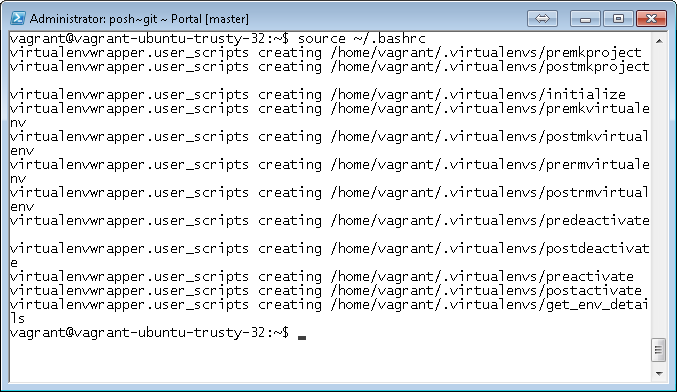


1. Se integra el ambiente de vagrant con virtualenv para crear el ambiente de virtualización para NUVEM CNCFLORA



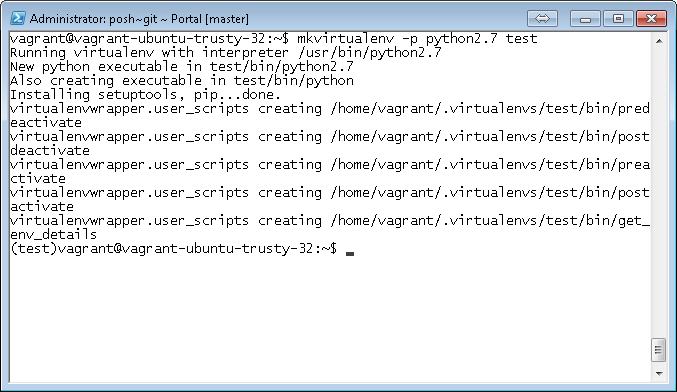


1. Se configura el script de virtualenv para arranque con el Sistema Operativo en los próximos reinicios.



1. Acto seguido, se procede a validar la instalación con

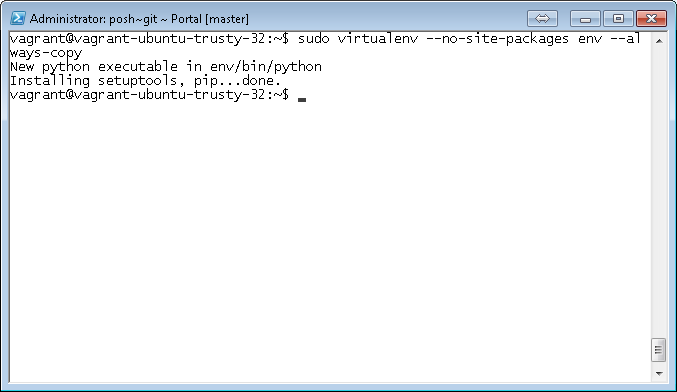
**mkvirtualenv -p python2.7 test**

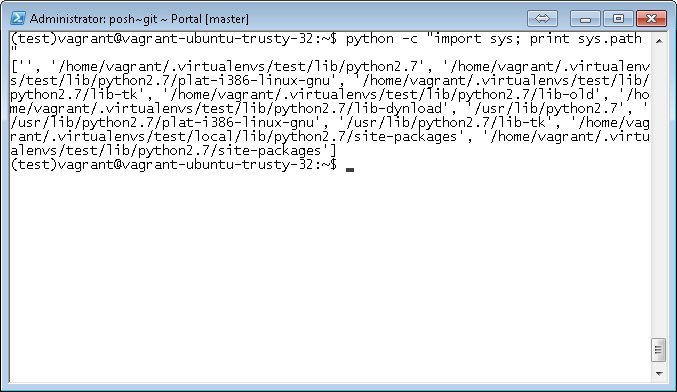


1. Se ejecuta una impresión por consola de PYTHON para validar que el motor (runtime de ejecución), está trabajando correctamente.

Probar con

**python -c "import sys; print sys.path"**



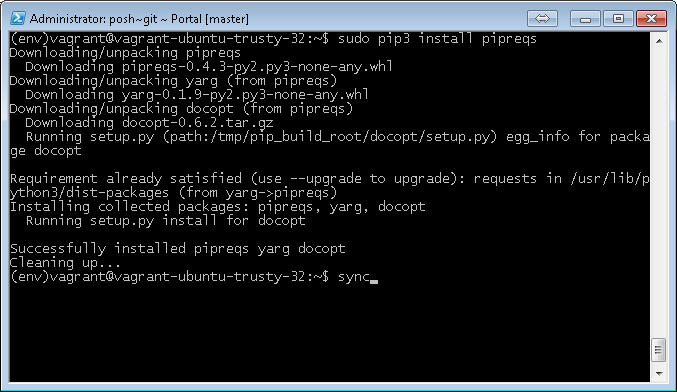


**CONFIGURACIÓN DE LIBRERÍAS DE LÍNEA BASE PARA PYTHON**

Como recurso del correcto funcionamiento de PYTHON, requerido para el clúster de apps CNCFLORA, se necesitan instalar las librerías adicionales, para lo cual, deben ejecutarse una vez terminado el montaje de línea base Python en la máquina virtual vagrant, los siguientes comandos y para los siguientes propósitos:

1. Realizar el montaje de pipreqs para Python con el comando:

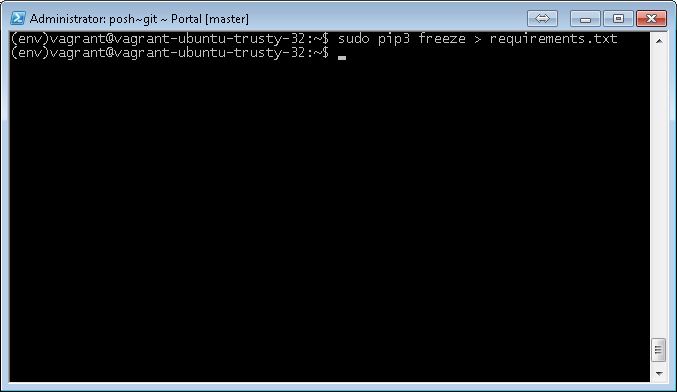
**sudo pip install pipreqs**

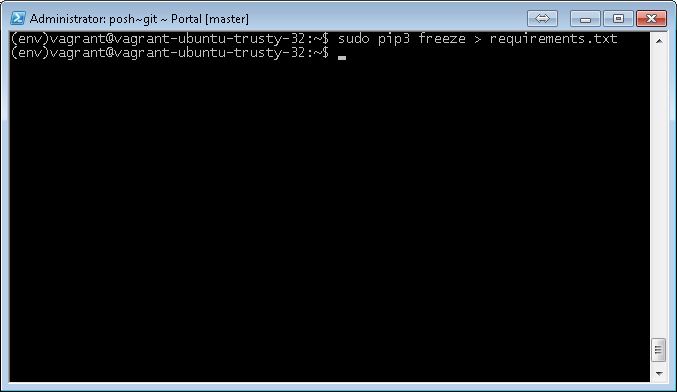


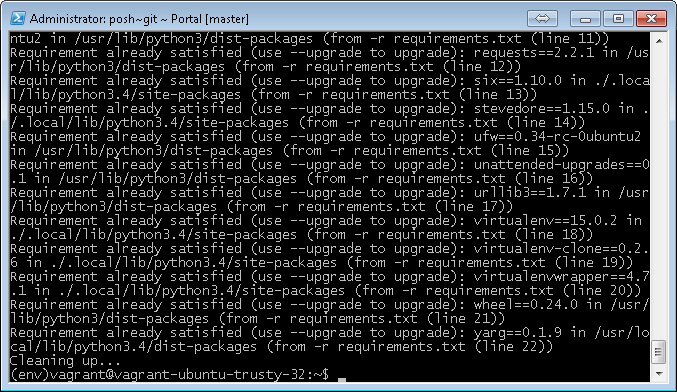
1. Generar backup de los requerimientos de línea base de instalación con el comando:

**sudo pip freeze > requirements.txt**

**sudo pip install -r requirements.txt**

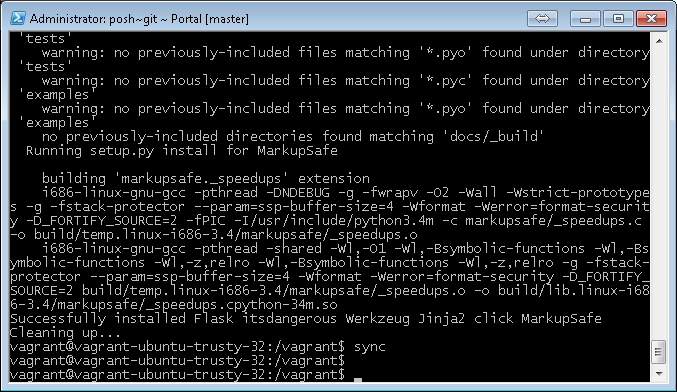






1. Instalar la librería de Flask con el comando

**sudo pip3 install Flask**

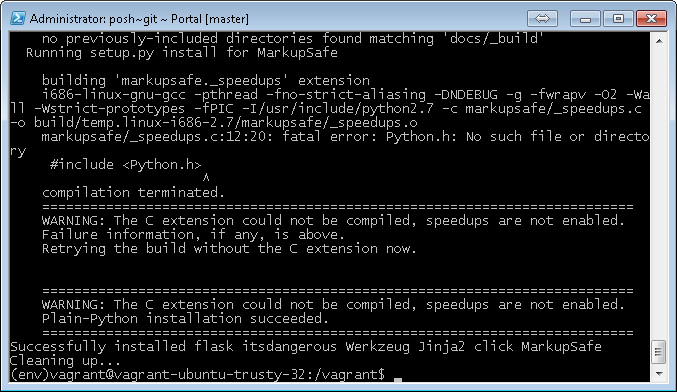


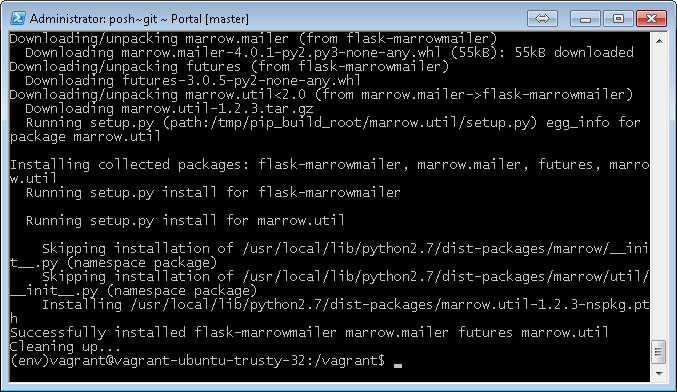
1. Instalar la librería flask (con minúscula) con el comando:

**sudo pip install flask**

1. Instalar posteriormente la librería de correo flask\_marrowmailer, con el comando

**sudo pip install flask\_marrowmailer**





1. Posteriormente se debe instalar la librería netifaces, requerida para la resolución en Python de la dirección local de la máquina (como obtención de la dirección IP del servidor de resolución) y de esta manera, poder configurar rutas de acceso al contexto o configurar hipervínculos de forma dinámico en código PYTHON/HTML. Se debe proceder con el conjunto de comandos

**sudo apt-get install python-dev**

**sudo pip install netifaces**

# ACTUALIZACIÓN DE APACHE PARA LA CONFIGURACIÓN DEL PROXY DEL SERVIDOR

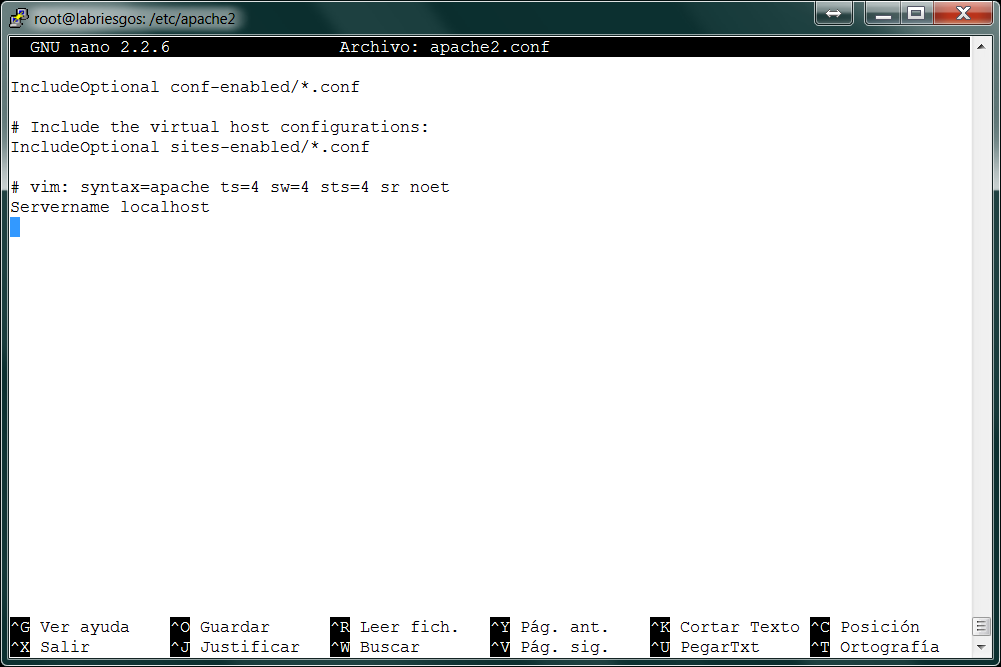
Con el fin de evitar el error BAD GATEWAY 502, se despliega luego de colocar a correr el clúster de aplicaciones APP NUVEM CNCFLORA, se debe instalar Apache con las siguientes instrucciones:

1. sudo apt-get update
2. sudo apt-get install apache2
3. sudo apt-get install libapache2-mod-php5 php5 php5-mcrypt

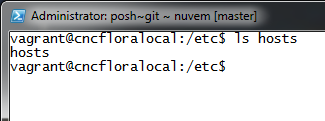
Acto seguido, se debe configurar el archivo apache2.conf, con la siguiente línea

ServerName localhost, tal como se muestra a continuación:

**/etc/apache2/apache2.conf**



Acto seguido, debe editarse la configuración de archivo de hosts en la ruta **/etc**



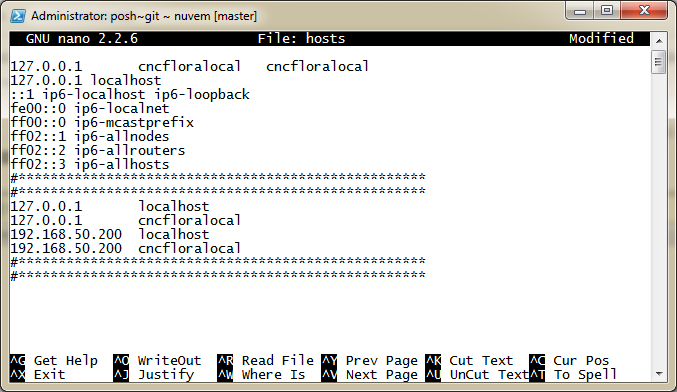
Editar el archivo con la dirección IP estática o dinámica asignada al Servidor

127.0.0.1 localhost

127.0.0.1 cncfloralocal

192.168.50.200 localhost

192.168.50.200 cncfloralocal

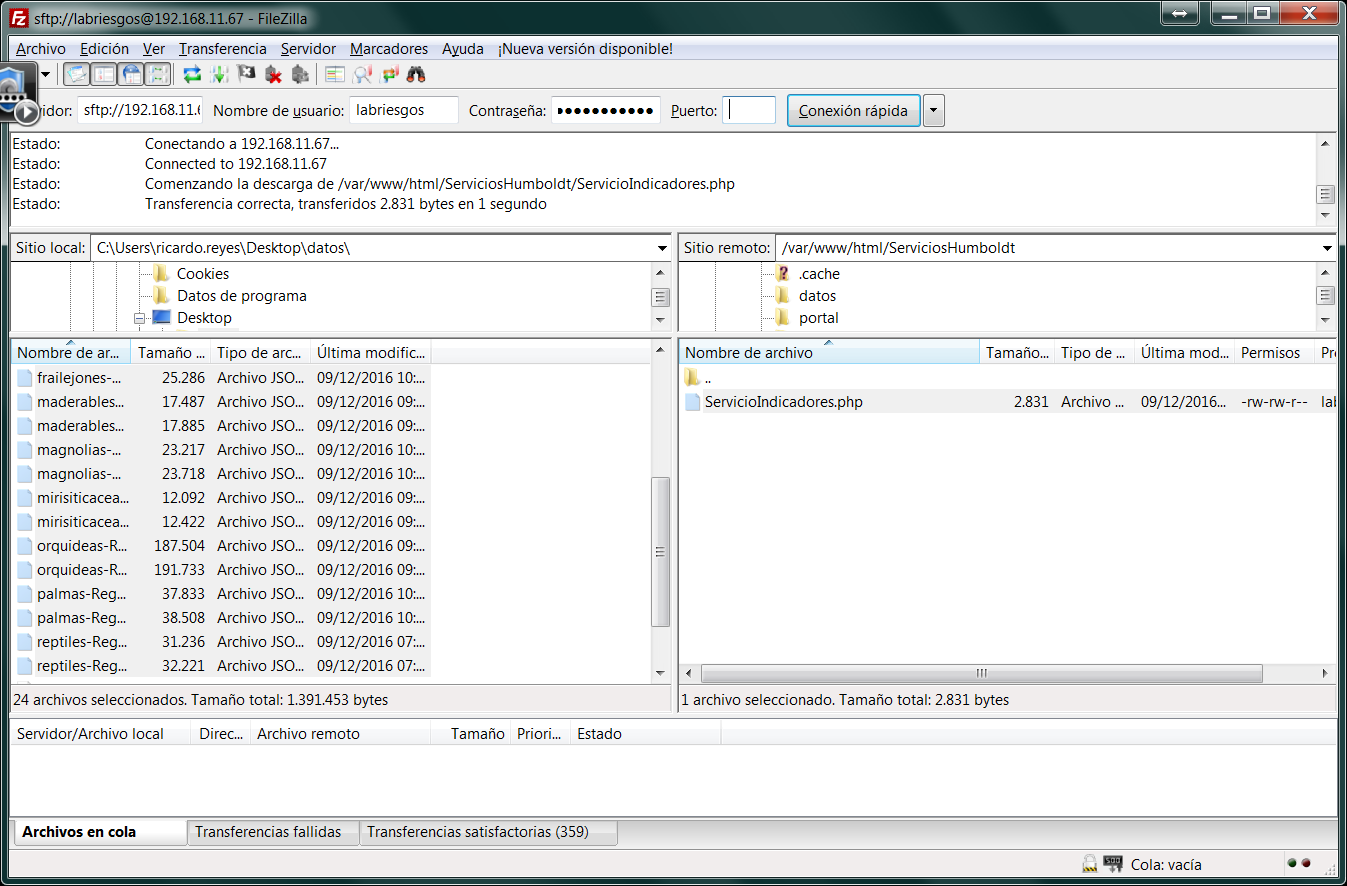


Y reiniciar el servicio de apache con

**service apache2 restart**

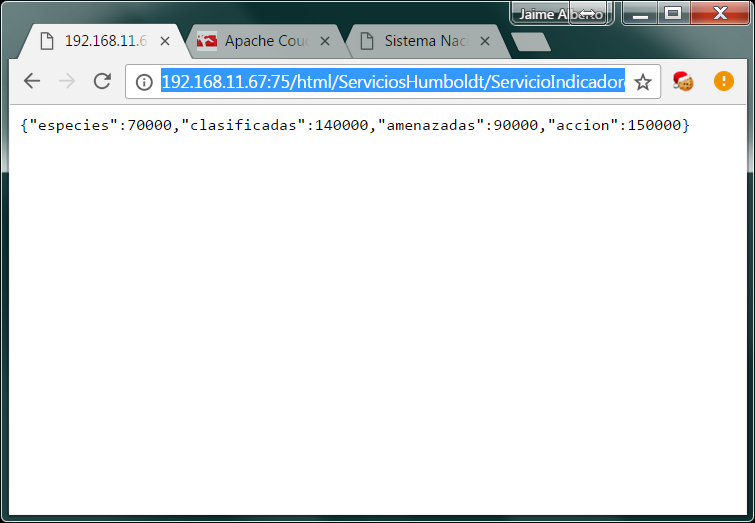


Posteriormente validar la instalación **/var/www/html** en la raíz del disco y desplegar los Servicios Web PHP desarrollados que hablarán con los handlers Python del App de Portal de CNCFLORA, tal como se ilustra a continuación:

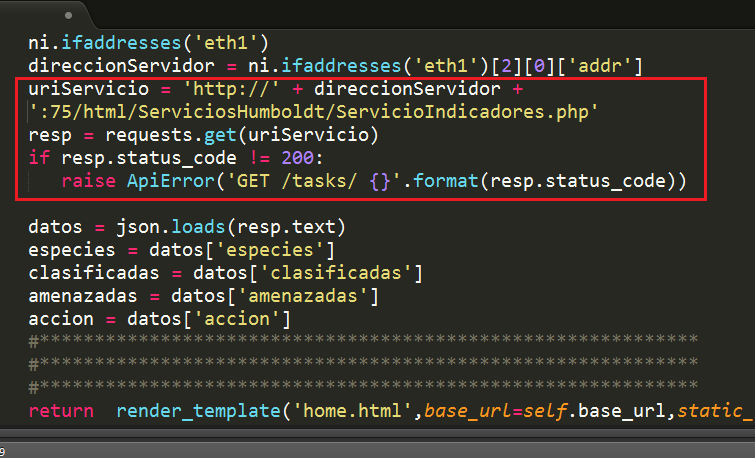


Resolver la URI con

<http://192.168.11.67:75/html/ServiciosHumboldt/ServicioIndicadores.php>



Validar la salida JSON y configurar la URI en el consumo del componente Python como se muestra a continuación:

****

# **INSTALACIÓN DE DOCKER PARA EL CLÚSTER DE APLICACIONES APP NUVEM CNCFLORA**

Para instalar el clúster se deben ejecutar los siguientes comandos en la MV instalada:

1. sudo apt-get update
2. sudo apt-get install apt-transport-https ca-certificates
3. Montar el repositorio con:

**sudo apt-key adv --keyserver hkp://pgp.mit.edu:80 --recv-keys 58118E89F3A912897C070ADBF76221572C52609D**

1. Actualizar el archivo del repo con:

**sudo echo 'deb https://apt.dockerproject.org/repo ubuntu-trusty main' > /etc/apt/sources.list.d/docker.list**

1. sudo apt-get update
2. sudo apt-get install docker-engine –y
3. sudo service docker start
4. sudo curl -L https://github.com/docker/compose/releases/download/1.6.2/docker-compose-`uname -s`-`uname -m` >$sudo chmod +x /usr/local/bin/docker-compose
5. sudo usermod -aG docker vagrant
6. sudo apt-get install docker
7. sudo pip install docker-compose
8. sudo docker-compose up –d (Descargar todos los paquetes y esperar que inicien los servicios)
9. Una vez se validen que los servicios suben, editar el docker-compose.yml, con la siguiente estructura:

|  |
| --- |
| version: "2"  services:  proxy:  image: diogok/caddy  restart: "always"  ports:  - 80:80  - 443:443  networks:  - nuvem  volumes:  - ./Caddyfile:/etc/Caddyfile  - ./index.html:/var/www/html/index.html  - /opt/caddy:/root/.caddy  couchdb:  image: cncflora/couchdb  networks:  - nuvem  restart: "always"  ports:  - "5984:5984"  volumes:  - "/var/data/couchdb:/var/lib/couchdb:rw"  elasticsearch:  image: cncflora/elasticsearch  networks:  - nuvem  restart: "always"  ports:  - "9200:9200"  volumes:  - "/var/data/elasticsearch/data:/usr/share/elasticsearch/data:rw"  kibana:  image: cncflora/kibana  networks:  - nuvem  restart: "always"  floradata:  image: cncflora/floradata  networks:  - nuvem  restart: "always"  ports:  - 80  - 9001  aka:  image: cncflora/aka  networks:  - nuvem  restart: "always"  volumes:  - "/var/data/aka:/var/data/aka:rw"  environment:  PROXY: "/aka"  services:  image: cncflora/services  networks:  - nuvem  restart: "always"  environment:  DB: "public"  BASE: "/services"  **#portal:**  **# image: cncflora/portal**  **# networks:**  **# - nuvem**  **# restart: "always"**  **# environment:**  **# BASE: "/portal"**  entrypoint:  image: cncflora/entrypoint  networks:  - nuvem  restart: "always"  connect:  image: cncflora/connect  networks:  - nuvem  restart: "always"  volumes:  - "/var/data/connect:/var/floraconnect:rw"  environment:  PROXY: "/connect"  dwc\_services:  image: cncflora/dwc-services  networks:  - nuvem  restart: "always"  environment:  PROXY: "/dwc\_services"  checklist:  image: cncflora/checklist  networks:  - nuvem  restart: "always"  environment:  BASE: "/checklist"  HOST: "${HOST}"  occurrences:  image: cncflora/occurrences  networks:  - nuvem  restart: "always"  environment:  BASE: "/occurrences"  HOST: "${HOST}"  profiles:  image: cncflora/profiles  networks:  - nuvem  restart: "always"  environment:  BASE: "/profiles"  HOST: "${HOST}"  assessments:  image: cncflora/assessments  networks:  - nuvem  restart: "always"  environment:  BASE: "/assessment"  HOST: "${HOST}"  migrator:  image: cncflora/migrator  networks:  - nuvem  restart: "always"  environment:  HOST: "${HOST}"  reports2:  image: cncflora/reports2  networks:  - nuvem  restart: "always"  arquivos:  image: cncflora/apache  networks:  - nuvem  restart: "always"  volumes:  - "/var/data/www:/var/www:rw"  arquivos-sftp:  image: asavartzeth/sftp  networks:  - nuvem  restart: "always"  ports:  - "2223:22"  environment:  SFTP\_USER: "cncflora"  SFTP\_PASS: "1cncflora2"  volumes:  - "/var/data/www:$SFTP\_DATA\_DIR/chroot/www:rw"  networks:  nuvem: |

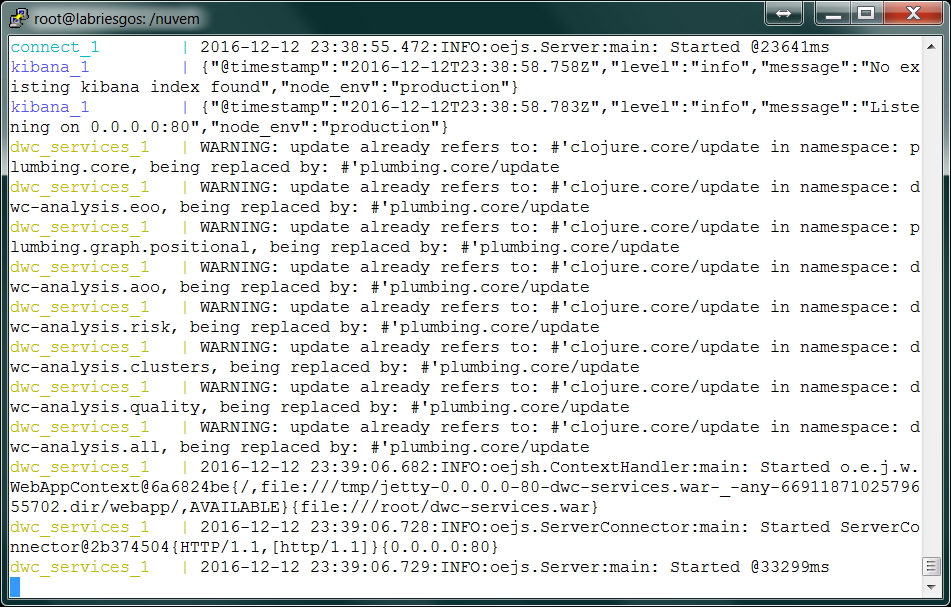
Solo comentariar la sesión de Portal y arrancar el comando con:

export HOST=IP\_SERVIDOR\_AMBIENTE

ejemplo

**export HOST=192.168.11.67**, y luego

**sudo docker-compose up** en la carpeta **/vagrant**

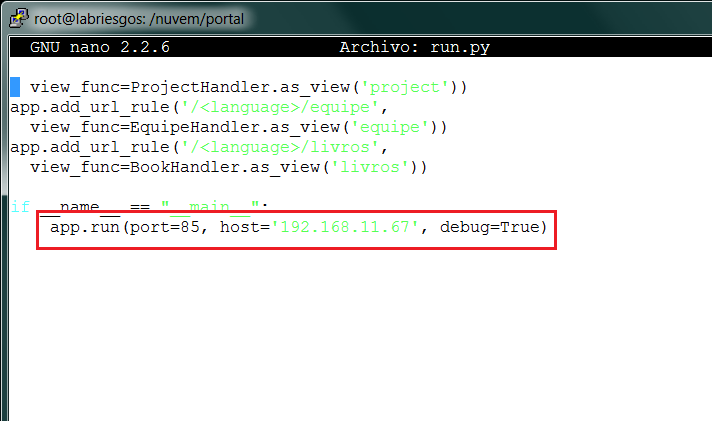


Validar el acceso a aplicaciones cluster, en las siguientes URI:

|  |  |  |
| --- | --- | --- |
| **Aplicación** | **URI de acceso** | **APLICACIÓN** |
| **Occurrences** | **http://192.168.11.67/occurrences/** |  |
| **Assessment** | [**http://192.168.11.67/assessments**](http://192.168.50.100/assessments) |  |
| **reports** | [**http://192.168.11.67/reports2/index.html**](http://192.168.50.100/reports2/index.html) |  |
| **Services** | **http://192.168.11.67/services/index.html** |  |
| **entrypoint** | [http://192.168.11.67/entrypoint/](http://192.168.50.100/entrypoint/) |  |
| **floradata** | [**http://192.168.11.67/floradata/index.html**](http://192.168.50.100/floradata/index.html) |  |
| **Connect** | [**http://192.168.11.67/connect/index**](http://192.168.50.100/connect/index) |  |
| **checklist** | **http://192.168.11.67/checklist** |  |
| **Aka** | **http://192.168.11.67/aka** |  |
| **migrator** | **http://192.168.11.67/migrator/** |  |

# ARRANQUE DEL AMBIENTE DE FRONT END PORTAL NUVEM CNCFLORA

1. Ingresar en la consola luego de sesionar a la carpeta /vagrant/portal
2. Editar el archivo run.py y establecer la configuración de depuración y la dirección IP del servidor



1. Crear el comando arranque.sh, con el siguiente contenido:

|  |
| --- |
| echo "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"  echo "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"  **sudo python run.py**  echo "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"  echo "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" |

1. Asignar permiso de ejecución al script con

**chmod +777 arranque.sh**

1. Ejecutar y validar el arranque del servidor



1. Resolver el Portal en la URI

<http://192.168.11.67:85/>





**Elaboró:**

**Ingeniero**

**JAIME ALBERTO GUTIÉRREZ MEJÍA**

**CONSTRATISTA INSTITUTO ALEXANDER VON HUMBOLDT**

**DICIEMBRE 2016**