Noticiero

Proyecto Infraestructura Virtual

2018-2019

Universidad de Granada

Índice

Introducción

Mi proyecto creado para la asignatura de infraestructura virtual, se basa en un micro-servicio en la nube.

Su funcionalidad es ser un “noticiero” como su nombre índica. Sera como una aplicación en la que el usuario, puede introducir una serie de noticias, consultarlas, por usuario, o fecha.

Esta versión de la aplicación está en pruebas, pero en un futuro, será puesta en desarrollo y mejoraremos tanto su funcionalidad, como su base de datos…

Versión anterior ..

Anteriormente, en un inicio, cuando se le propuso la idea al profesor, nuestra aplicación iba a ser un Bot de Telegram..

Por una serie de circunstancias decidí dejar esta idea, y centrarme simplemente en hacer un noticiero más simple..

En un futuro, cabe destacar que podríamos implementar la versión inicial, de tener un canal de noticias en Telegram o otra plataforma similar.

Herramientas utilizadas

1. **Lenguaje : Python3**
2. **Entorno de desarrollo virtual : virtualenv**
3. **Base de datos: MariaDB (Actualmente para el almacenamiento estático usaremos ficheros JSON)**
4. **Test: Travis-CI.**
5. **Para realizar el testeo de la aplicación: unittest**
6. **Micro-Framework : Flask**

Integración continua

En nuestro proyecto, hemos introducido la integración continua para poder realizar los test, con Travis-CI

Hemos creado una clase que se llama *Noticias* en la que podremos realizar una serie de funciones como :

- Obtener todas las noticias de un determinado día.
- Obtener las noticias que un usuario público en un determinado momento.
- Añadir una noticia a la lista de noticias.

Para detectar los fallos usaremos una serie de test, basándonos en Travis-CI

Hemos tenido que instalar Travis, seleccionar nuestro proyecto para que puedan pasar los test, y tendremos que añadir un fichero de configuración llamado:

.travis.yml

```
1   language: python
2
3   python:
4     - "3.6.6"
5
6   # command to install dependencies
7   install:
8     - pip install -r requirements.txt
9
10  # command to run tests
11  script:
12  - make test
```

También tendremos que instalar los requerimientos que nuestra aplicación necesita.

Test

Los test que hemos usado son:

- **testFecha :** comprueba que la fecha tenga el formato aaaa-mm-dd por lo que no puede ser ni una cadena, ni un número suelto.
- **testUsuario** : nosotros hemos prefijado que el usuario sea un número, por lo que no pude ser una cadena
- **testAniadirNoticia:** para poder añadir una noticia, la fecha que ponemos se pone dentro del set como la fecha actual de día, ya que nosotros no hemos puesto la fecha como la fecha del suceso si no como la fecha de públicación de la misma. Esta tiene que cumplir que el usuario sea un número y la noticia una cadena.

Travis CI

Despliegue PaaS : Heroku

Para que nuestro micro-servicio empiece a funcionar se ha elegido como PaaS (Platform as a Service) : **Heroku**

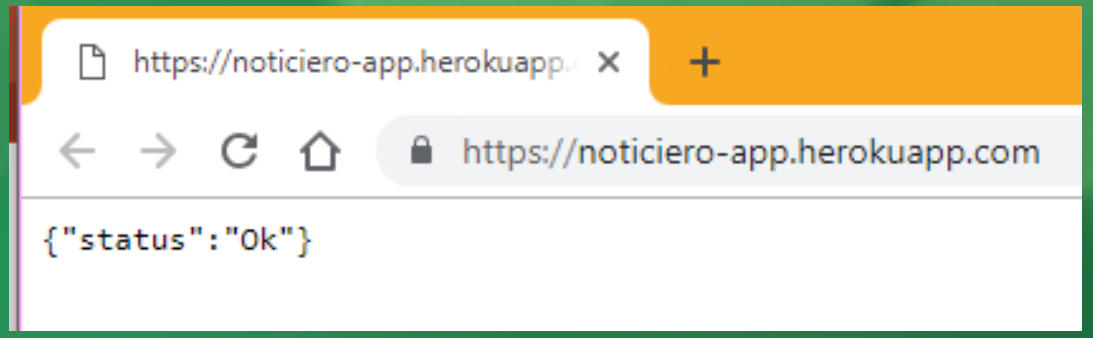
Heroku es una plataforma como servicio de computación en la Nube que soporta distintos lenguajes de programación.

Hemos elegido heroku, simplemente por que es gratuito, es fácil de usar, permite la integración con Github y Travis-CI

Pasos para el despliegue

1. **Nos creamos una cuenta en Heroku**
2. **Creamos una nueva aplicación**
3. **Creamos un documento con los Dynos, que se llama *Procfile (Su función es indicar el comando que se tiene que ejecutar, nos indica el fichero en Python donde se produce el despliegue)***
4. ***Asociamos Github con Heroku, y le indicamos que realice el despliegue automático.***

Despliegue : https://noticiero-app.herokuapp.com/

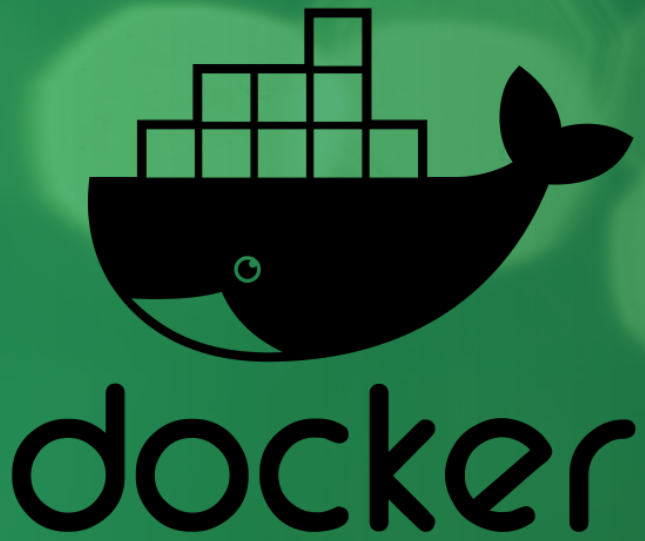


Configuramos el entorno de pruebas usando contenedores
Virtualización ligera : Docker

Hemos usado Docker como contenedor, ya que nos permite aislar la aplicación para que podamos hacer el despliegue de forma sencilla, garantizando que nuestro despliegue sea escalable de forma eficiente.

El despliegue del contenedor, lo hemos realizado en *Heroku*

La publicación de la imagen ha sido en *DockerHub*



Hemos tenido que crear un documento llamado DockerFile:

```
FROM python:3

MAINTAINER Natalia <nataliamartir@correo.ugr.es>
WORKDIR src/
COPY . .
RUN pip install --no-cache-dir -r requirements.txt

EXPOSE 80
CMD cd src && gunicorn noticiero-app:app --log-file=- --bind 0.0.0.0:80
```

Despliegue y creación de la imagen

Para el despliegue en heroku, hemos creado el archivo, heroku.yml que nos permitirá crear la imagen a partir del DockerFile.

Hemos indicado a Heroku, que nuestra aplicación es un contenedor.

Para crear la imagen, hemos entrado en la pagina: https://hub.docker.com/

Contenedor : https://contenedornoticias.herokuapp.com/

Despliegue de la imagen : https://hub.docker.com/r/natalia2911/proyectoiv-bot/

Despliegue en Azure :
Virtualización completa

- **Para ello hemos tenido que crearnos una cuenta en Azure, con una subscripción.**
- **Con vagrant, hemos creado una máquina virtual en azure, con el fichero Vagrantfile.**
- **Hemos creado un archivo de aprovisionamiento con ansible, llamado *playbook.yml* en el que queremos preparar la máquina para su correcto funcionamiento.**
- **Por último hemos creado un fichero con Fabric, llamado *fabfile.py* en el que se ejecutaran una serie de funciones de manera remota.**

Vagrantfile

```
        config.vm.box_url = "https://github.com/azure/vagrant-azure/raw/v2.0/dummy.box"

        # use local ssh key to connect to remote vagrant box
        config.ssh.private_key_path = "~/.ssh/id_rsa"


        config.vm.provider :azure do |azure, override|

                azure.tenant_id = ENV['AZURE_TENANT_ID']
                azure.client_id = ENV['AZURE_CLIENT_ID']
                azure.client_secret = ENV['AZURE_CLIENT_SECRET']
                azure.subscription_id = ENV['AZURE_SUBSCRIPTION_ID']

                azure.vm_size = "Basic_A0"
                azure.location = "westeurope"
                azure.tcp_endpoints = "80"
                azure.vm_name = "noticieroapp"
                azure.resource_group_name= "recursosiv"
        end

        config.vm.provision :ansible do |ansible|
                ansible.playbook = "./provision/playbook.yml"
        end
end
```

Playbook.yml

```
- hosts: all
  sudo: yes
  remote_user: vagrant

  tasks:
  - name: Actualizacion
    command: sudo apt-get update

  - name: Instalar Python
    command: sudo apt-get install python3

  - name: Instalar Pip
    command: sudo apt-get install python3-pip

  - name: Instalar Git
    command: sudo apt-get install git

  - name: Instalar Requerimientos
    command: pip3 install -r ProyectoIV-BOT/requirements.txt
```

Fabfile.py

```python
from fabric.api import *
from fabric.contrib.console import confirm
import os


# Definimos una variable de entorno con el host al que nos vamos a conectar
# y el nombre de usuario
env.user = "vagrant"
env.host = ['noticieroapp.westeurope.cloudapp.azure.com']


def Desinstalar():
        #Borramos el codigo antiguo
        run("sudo rm -rf ./ProyectoIV-BOT")

def Instalar():
        #Instalamos el servicio clonando eliminando el codigo anterior e instalando los requerimientos
        Desinstalar()
        run("git clone https://github.com/natalia2911/ProyectoIV-BOT")
        run('cd ProyectoIV-BOT/  && sudo pip3 install -r requirements.txt')


def Iniciar():
        #Iniciamos el servicio
        with cd("ProyectoIV-BOT/src/"):
                sudo("gunicorn noticiero-app:app --log-file - &")
```

Despliege final : 13.80.251.123

DNS : noticieroapp.westeurope.cloudapp.azure.com

Enlace proyecto:

https://github.com/natalia2911/ProyectoIV-BOT

Bibliografía

https://www.mysql.com/

https://mariadb.org/

https://docs.python.org/3/library/unittest.html

https://rukbottoland.com/blog/tutorial-de-python-virtualenv/

https://travis-ci.com/

https://www.heroku.com/

https://dashboard.heroku.com/

https://hub.docker.com/

https://devcenter.heroku.com/articles/build-docker-images-heroku-yml

https://moduslaborandi.net/post/introduccion-a-fabric-ii/

https://axiacore.com/blog/sudo-y-fabric-para-administrar-y-desplegar-como-devop/

https://blocknitive.com/blog/workshop-hyperledger-fabric-despliegue-de-una-red-fabric-para-universidades/

https://docs.fabfile.org/en/1.4.3/

http://docs.fabfile.org/en/1.14/tutorial.html

https://www.rubydoc.info/gems/vagrant-azure/1.3.0

https://blog.deiser.com/es/primeros-pasos-con-ansible

https://stackoverrun.com/es/q/7574013

¿Preguntas?

Natalia María Mártir Moreno

Realizado por