Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencias de la Computación

IIC2413 – Bases de Datos

FlaskDB Deployment to Production

Introducción

La presente guía busca entregarte los pasos necesarios para hacer deploy en production[1] de la aplicación `FlaskDB`, con el fin ser usada en las futuras entregas del curso IIC2413 - Bases de datos. Se te hará entrega de la aplicación FlaskDB, la cual será la que deberas modificar y deployar en producción.
`Flask` es un microframework web, similar a `Django`, sin embargo su diseño es minimalista y su compilación es más liviana. `Flask` es principalmente usado como API. En el contexto del curso, la aplicación `FlaskDB` permite el acceso a `PostgreSQL` y a `MongoDB`, y permite la manipulación de sus datos. El fin de la guia es que comprendas como usar y configurar esta aplicación en un servidor externo.

Funcionamiento de la aplicación

La aplicación se conecta con las credenciales de la base de datos relacional `PostgreSQL` y de documentos `MongoDB`, las cuales se te explicara como configurar más adelante en esta misma guía. El detalle de la estructura interna de la aplicación se encuentra en el documento adjunto `Estructura de FlaskDB`.

Intrucciones

1. Conectate con el servidor

La primero que debes hacer conectarte al servidor para poder configurar la aplicación FlaskDB y las bases de datos. Se te otorgara la dirección de tu servidor (al que llamaermos en la guia como `myserver`), un nombre de usuario (al que nos referiremos como `myuser`) y una contraseña (al que nos referiremos como `mypass`).

MacOS y Unix dist. Puedes conectarte directamente desde la consola con el protocolo `ssh` usando el siguiente comando:

```
ssh myuser@myserver
```

Se te pedira tu contraseña, deberas ingresarla. desde este punto tu terminal representara a la terminal del servidor que se te fue asignado, y los comandos que ejecutes serán reflejados en tu servidor. Para salir debes ingresar el comando `exit`.

Windows Para lograr la conexión `ssh` con windows, debes descargar PuTTY, e ingresar los datos de tu servidor. Se te abrira una consola, que será la que podrás usar para la conexión con tu servidor.

[1] Dejar la aplicación *corriendo* en un servidor externo, no desde tu maquina en localhost. Ver Software deployment.

2. Instala las Bases de Datos

Instalar Postgres

Los siguientes comandos son para poder correr la base de datos `Postgres`

```
sudo apt-get update
sudo apt-get install postgresql postgresql-contrib
```

Instalar Mongo

Los sisguientes comandos son para instalar `MongoDB`

```
sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv EA312927
echo "deb http://repo.mongodb.org/apt/ubuntu trusty/mongodb-org/3.2 multiverse" | sudo tee
/etc/apt/sources.list.d/mongodb-org-3.2.list
sudo apt-get update
sudo apt-get install -y mongodb-org mongodb-clients
```

3. Configura el servidor

Instalar Apache

Una vez has establecido la conexión con el servidor, debes ejecutar el siguiente comando para instalar los paquetes necesarios para correr la aplicación.

```
sudo apt-get update
sudo apt-get install apache2
sudo apt-get install libapache2-mod-wsgi python3-dev python3-pip
```

Si se descargó correctamente, podrás ver en la dirección de tu servidor la pagina por defecto de Apache 2.

Lo siguiente es setear wsgi para flask, con el siguiente comando

```
sudo a2enmod wsgi
```

4. Descarga y configura la aplicación

Ahora procederemos a configurar e instalar la aplicación, para eso debes clonarla desde el repositorio[2] en github donde hayas subido tu aplciación.

```
cd /var/www
sudo apt-get install git
git clone repositorio
mv FlaskDB flaskr
```

Donde repositorio es la dirección de tu github, donde subiste FlaskDB.

Ahora para correr la aplicación se usara un virtual enviroment, para eso debes ejecutar los siguientes comandos

[2] Asumiendo que descargaste la aplicación que se te facilitó, y la subiste a tu propio repositorio

```
sudo pip3 install virtualenv
sudo virtualenv venv
source venv/bin/activate
```

Con el comando anterior creamos y activamos nuestro virtual enviroment. Ahora instalaremos Flask dentro de el.

```
sudo pip3 install Flask Flask-PyMongo psycopg2
sudo apt-get install python3-psycopg2 libpq-dev
```

Luego para comprobar que todo funciona, puedes entrar a la carpeta del repositorio `cd flaskr` y ejecutar

```
cd flaskr
sudo python3 flaskr run
```

Debería salir un mensaje como el siguiente

```
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

En caso de que no aparezca, puede que te haya faltado instalar algun paquete. Si aparece estamos bien. Aprieta CTRL+C para salir de ese modo, y continuar con el tutorial. Escribe `deactivate` en la terminal para salir del virtual enviroment.

5. Enlaza la aplicación con Apache

Hasta ahora tu servidor no sabe que la aplicación FlaskDB es la que debe ser mostrada cuando se accede a la ruta desde el navegador. Ahora vamos a decirle que debe apuntar a FlaskBD. Ahora ejecuta en la terminal el siguiente comando

```
sudo nano /etc/apache2/sites-available/flaskr.conf
```

Con lo anterior, podras editar un archivo desde la terminal. Deberas copiar y pegar el siguiente texto en la terminal [3].

```
<VirtualHost *:80>
 ServerName direccionserver.com
 ServerAdmin usuario@direccionserver.com
 WSGIScriptAlias / /var/www/flaskr/flaskr.wsgi
 <Directory /var/www/flaskr/flaskr/>
 Order allow,deny
 Allow from all
 </Directory>
 Alias /static /var/www/flaskr/flaskr/static
 <Directory /var/www/flask/flaskr/static/>
 Order allow,deny
 Allow from all
 </Directory>
 ErrorLog ${APACHE_LOG_DIR}/error.log
 LogLevel warn
 CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

[3] Se te facilitara el contenido de este archivo para que puedas copiarlo facilmente

Guarda presionando `CTRL + X` y `ENTER`.

Lo siguiente es habilitar el host con el siguiente comando

```
sudo a2ensite flaskr
```

Ahora vas a crear el archivo wsgi que expone como funciona FlaskDB

```
sudo nano /var/www/flaskr/flaskr.wsgi
```

Debe decir lo siguiente[4]

```
#!/usr/bin/python3
activate_this = '/var/www/flaskr/venv/bin/activate_this.py'
with open(activate_this) as file_:
    exec(file_.read(), dict(__file__=activate_this))
import sys
import logging

logging.basicConfig(stream=sys.stderr)
sys.path.insert(0,"/var/www/flaskr/")

from flaskr import app as application
application.secret_key = 'Add your secret key'
```

Finalmente,

```
sudo service apache2 restart
```

Ahora puedes acceder a la dirección de tu servidor, desde tu navegador y ver la aplicación corriendo. Si aparece ¡Felicitaciones! Has terminado el *deployment to production* de tu aplicación. Algunas cosas de tu aplicación aun no deberían estar listas. Las queries del archivo `queries.json` no tinen sentido[5], así mismo no se conecta con tu base de datos mongo (pues aun no has creado alguna base de datos en ella[6]). Por otra parte, tu aplicación podría necesitar cambios dependiendo de lo que quieras mostrar[7].

Recuerda guardar los cambios creados en tu repositorio, con los siguientes comandos

```
git add --all
git commit -m ":rocket: Deployed app"
git push origin master
```

Ahora tu servidor y aplicación en el repositorio se encuentran sincronizadas. Desde aqui solo debes jugar con los `push` y `pull` de lo que tienes en tu computador y lo que esta en el servidor, y evitar eliminar algun archivo de configuración.

[4] Se te facilitara el contenido de este archivo para que puedas copiarlo facilmente

[5] Es tu misión que estas queries terminen teniendo algun sentido

[6] Puedes ver la documentacion de mongo, para ver como crear una database y una collection a partir de un dataset en json

[7] Puedes lograr esto facilmente desde el repositorio github, sincronizando tu maquina local con `git push ...`, y luego sincronizar en el servidor con `git pull ...` en la carpeta `/var/www/flaskr/` del servidor

Recomendaciones

1. **Manten tu aplicación en el servidor con tu github sincronizados** Esto te ahorrara muchos problemas, pues te da la posibilidad de probar las cosas de forma local, clonando el repositorio en tu computador. Así mismo, si realizas cambios directamente en el servidor, podrás guardarlos en el tu repositorio para no perderlos en caso de modificaciones futuras. Sin embargo no es recomendable modificar el codigo directamente desde tu servidor, pues no puedes ver directamente que es lo que podría estar fallando.
2. **Dedicale tiempo a subir tu aplicación** En general el proceso de subir tu aplicación al servidor es un proceso facilmente mecanizable, sin embargo es común que salgan problemas al ponerlo en práctica, como por ejemplo, problemas con paquetes o versiones. Es por eso que se te recomienda hacerlo de forma calmada y con tiempo.