THREEPOINTS
THE SCHOOL FOR DIGITAL BUSINESS

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Master en FullStack Developer

CI/CD y DevOps
Practica 2 - Construcción de un Pipeline declarativo

MIA 2011

Objetivo

El objetivo de esta práctica es aprender los fundamentos de uso de los pipelines declarativos, altamente usados en cualquier entorno dev ops profesional.

Durante estos ejercicios se reutilizará la máquina virtual presentada en la primera práctica, recordemos cuales eran los servicios montados:

- Gitlab: http://localhost (o http://localhost:8000 si se ha cambiado el puerto)
  - Usuario: threepoints
  - Password: threepoints
- Jenkins: http://localhost:8080
  - Usuario:threepoints
  - Password: threepoints
- ~~SonarQube: http://localhost:8090~~
  - ~~Usuario: admin~~
  - ~~Password: threepoints~~
  - ~~(Está apagada por defecto, hay que seguir las instrucciones del tema 3 para desplegarlo).~~

Para entrar a esta máquina virtual, se usarán las credenciales:
- Usuario: threepoints
- Password: threepoints

1. Creación de un Pipeline Declarativo básico de CI (6 puntos).

Durante este primer ejercicio, versionaremos el pipeline que creamos en la primera práctica. Para ello, se usará el proyecto de gitlab, que consistía en una aplicación web dockerizada: http://localhost/threepoints/docker-vulnerable-dvwa/

Habrá que crear un nuevo Job de tipo Pipeline, que consista en un pipeline básico que realice ciertas comprobaciones antes de construir y lanzar el Docker con el proyecto que tenemos.

El pipeline debe de tener los siguientes stages:

1. **Descarga de código**. Debe de descargarse el código del gitlab para su integración. Este stage se debe de llamar “Checkout”.

2. **Ejecución de pruebas de calidad de código**. ~~Para esto, se hará un step con dos ejecuciones, en la primera usando el plugin de SonarQubeEnv, y en la segunda esperando en un timeout los resultados de la ejecución del qualityGate, fijando la variable de abortPipeline a false.~~ Debido a los problemas con SonarQube, este stage se dejará en estado de Mock. Habrá que asignarle un valor de True a la variable de entorno PASS_QUALITY_GATE e imprimir por

pantalla, si el valor está a True, la frase “Pasa las pruebas de SonarQube”, y si está a False, la frase “No pasa la prueba de SonarQube”. Este stage se debe de llamar “Pruebas de SAST”.

3. **Construcción del container de Docker**. Para ejecutar esto, hace falta ejecutar el comando ‘**`docker build -t dvwa .`**’ Este stage se debe de llamar “Build”.

El Job tiene que llamarse “**practica_2_2022_*NOMBRE_ALUMNO***” y se debe de entregar:

- El código del pipeline.
- Capturas del pipeline ejecutado correctamente mediante la visión de BlueOcean.
- Captura de pantalla con los resultados de SonarQube.

2. Ejecutar dos steps de manera paralela. (2 puntos).

Durante el siguiente punto, basándose en el pipeline creado en el anterior ejercicio, hay que añadir un nuevo stage, hecho de forma **paralelizada** (pararells) al stage de “Pruebas de SAST” del anterior ejercicio, que imprima por pantalla, mediante un comando “echo”, el path del WORKSPACE donde se está ejecutando el Jenkins. Este Stage se debe de llamar “Imprimir Env”.

Este path se puede obtener mediante una de las variables globales de entorno que se ha explicando durante la clase. Como entregable de este ejercicio, será necesario:

- Código del pipeline que ejecuta esto.
- Captura de la pantalla de la ejecución exitosa de BlueOcean.
- Captar de pantalla del debug de la ejecución del pipeline donde muestra la impresión realizada.

3. Usar credenciales en un pipeline declarativo (2 puntos).

Para este ejercicio vamos a crear un nuevo parámetro en el Job de tipo Credenciales, de Usuario y Contraseña, y serán el usuario “admin” y la contraseña, “password”. Este parámetro deberá de recibir el nombre y el ID de “Credentials_Threepoints”.

Al Job del ejercicio 1 (o del 2 si se ha realizado), hay que añadir un nuevo Stage llamado “Configurar archivo”, ejecutado antes del “Build”, que, usando las credenciales anteriormente definidas, genere un archivo llamado ‘credentials.ini’ con el siguiente formato:

```
[credentials]
user=${user}
password=${password}
```

Siendo ${user} y ${password}, respectivamente, los valores que se obtiene de los credenciales configurados anteriormente.

Como entregables de este ejercicio, será necesario:

- Código del pipeline que ejecuta esto.
- Captura de la pantalla de la ejecución exitosa de BlueOcean.
- Captura de pantalla del debug de la ejecución del pipeline donde muestra la ejecución de los comandos necesarios.