Tiempo para aprender juntos

Dirección de Servicios de
Infraestructura y Operaciones

Mayo 2021

Jenkins

CopaAirlines

Agenda

- ¿Qué es la Orquestación?
- Beneficios de la Orquestación
- ¿Qué es Jenkins?
- ¿Cómo aprender Jenkins?
- ¿Cómo funciona Jenkins?
- ¿Cómo se integra con otras herramientas?
- Laboratorios

CopaAirlines

¿Qué es la Orquestación?

- Es la **configuración**, **gestión** y **coordinación automatizadas** de los *sistemas informáticos*, las *aplicaciones* y los *servicios*, que ayuda a **IT** a gestionar con mayor facilidad las **tareas complejas** y los **flujos de trabajo**.

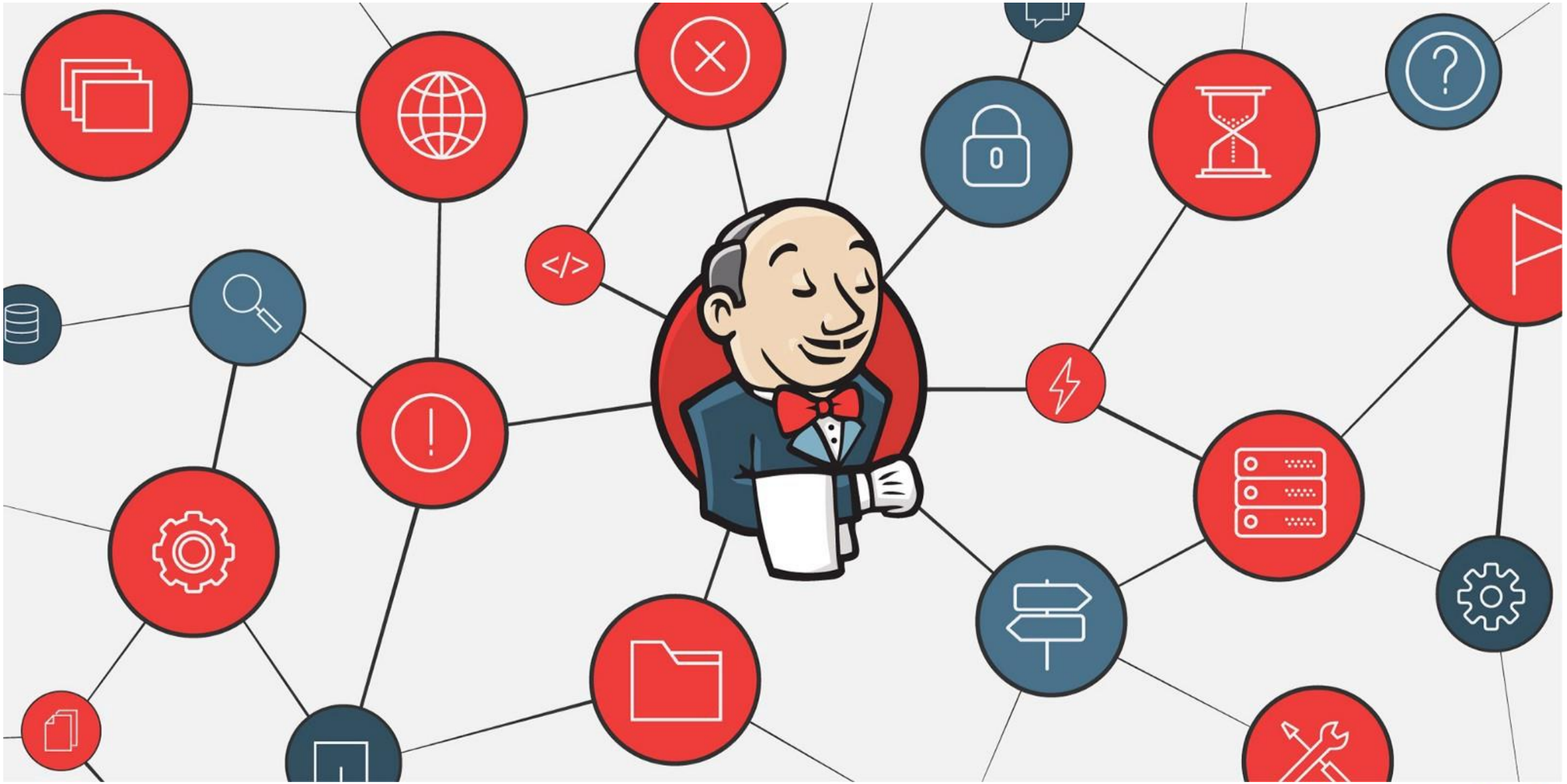
CopaAirlines

Beneficios de la Orquestación

- **Ahorro** de **tiempo**
- **Reducción** de **errores**
- **Reducción** de **Time to Market**
- **Reducción** de **duración** de **incidentes**
- **Mejor atención** al **cliente**
- **Incrementa colaboración** entre los **equipos** de trabajo

CopaAirlines

¿Qué es Jenkins?



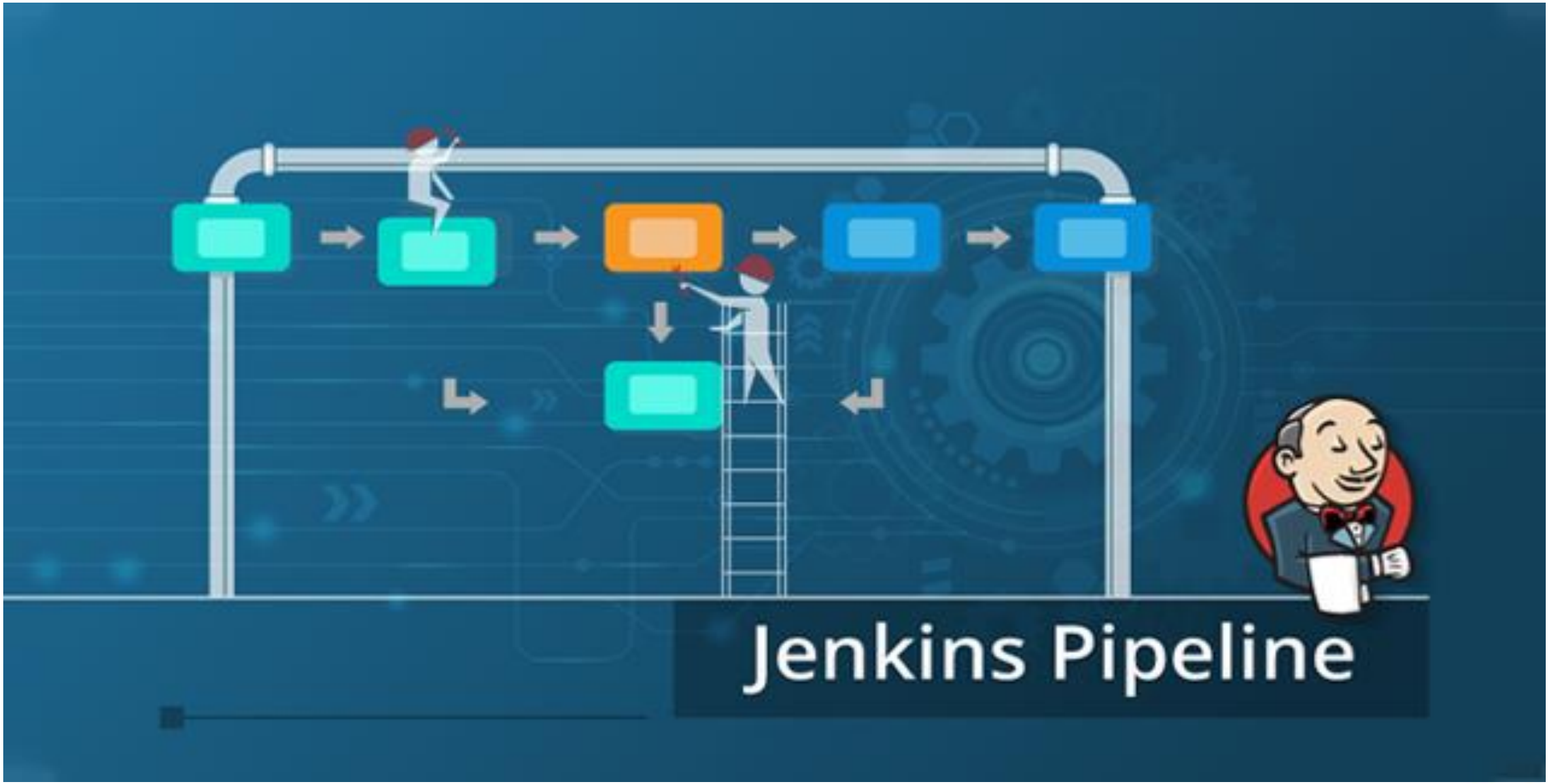
CopaAirlines

¿Qué es Jenkins?

- Es una **herramienta open source** (desarrollada en *Java*) de **integración continua**, usada principalmente para **orquestar procesos** en el **desarrollo** de **software**, sus grandes **capacidades** permiten utilizarlo para mas cosas a través del uso de **pipelines**, como por ejemplo:

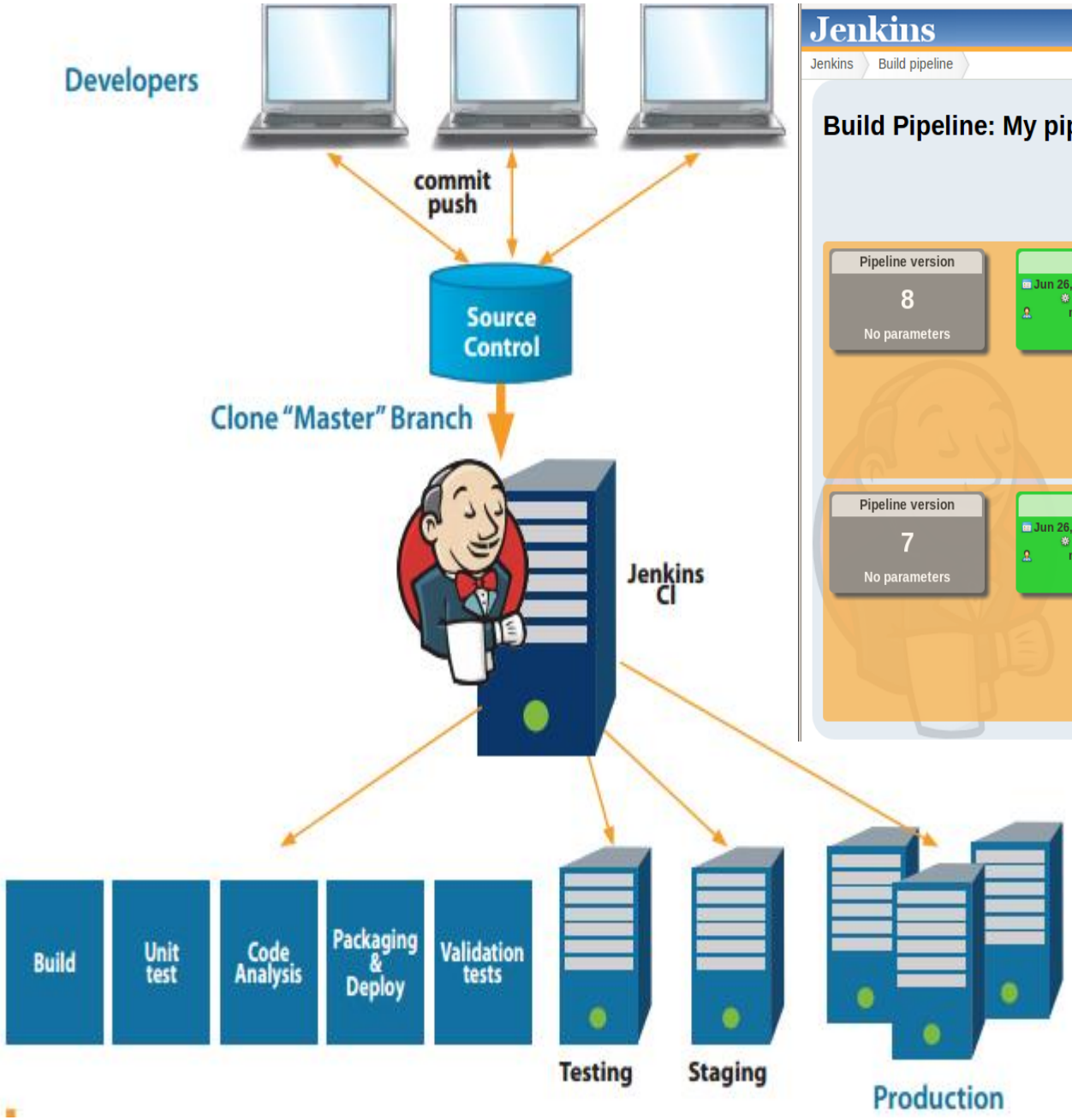
  - **Ejecución** de **procesos secuenciales**
  - **Monitoreo** de procesos
  - **Ejecución** a **horas determinadas**
  - **Ejecución** en base a **eventos**
  - **Ejecución distribuida** en varias **máquinas**
  - **Alertas** ante **problemas** en algún proceso

¿Qué es Jenkins?



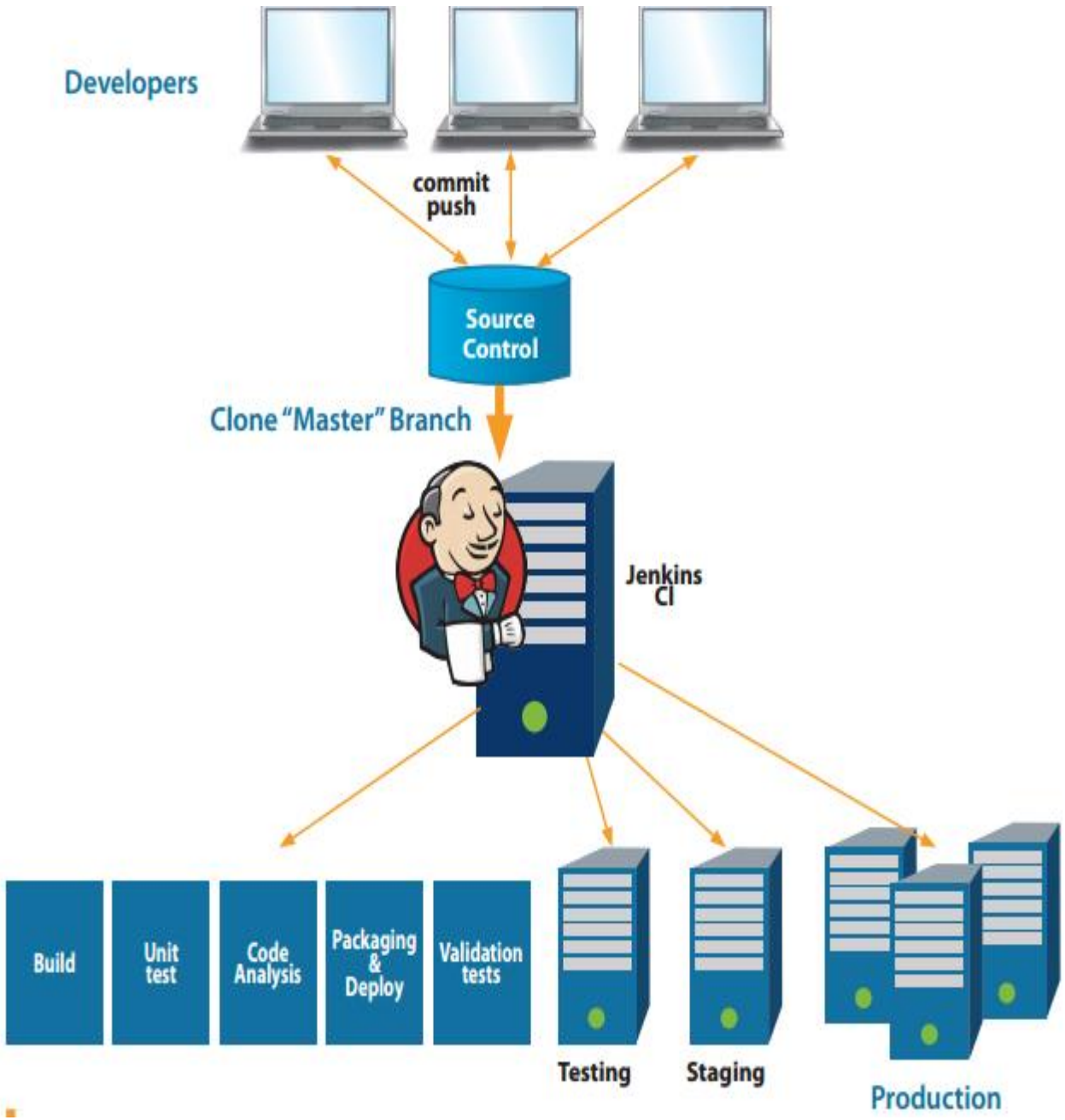
CopaAirlines

¿Qué es Jenkins?



CopaAirlines

¿Qué es Jenkins?



Declarative

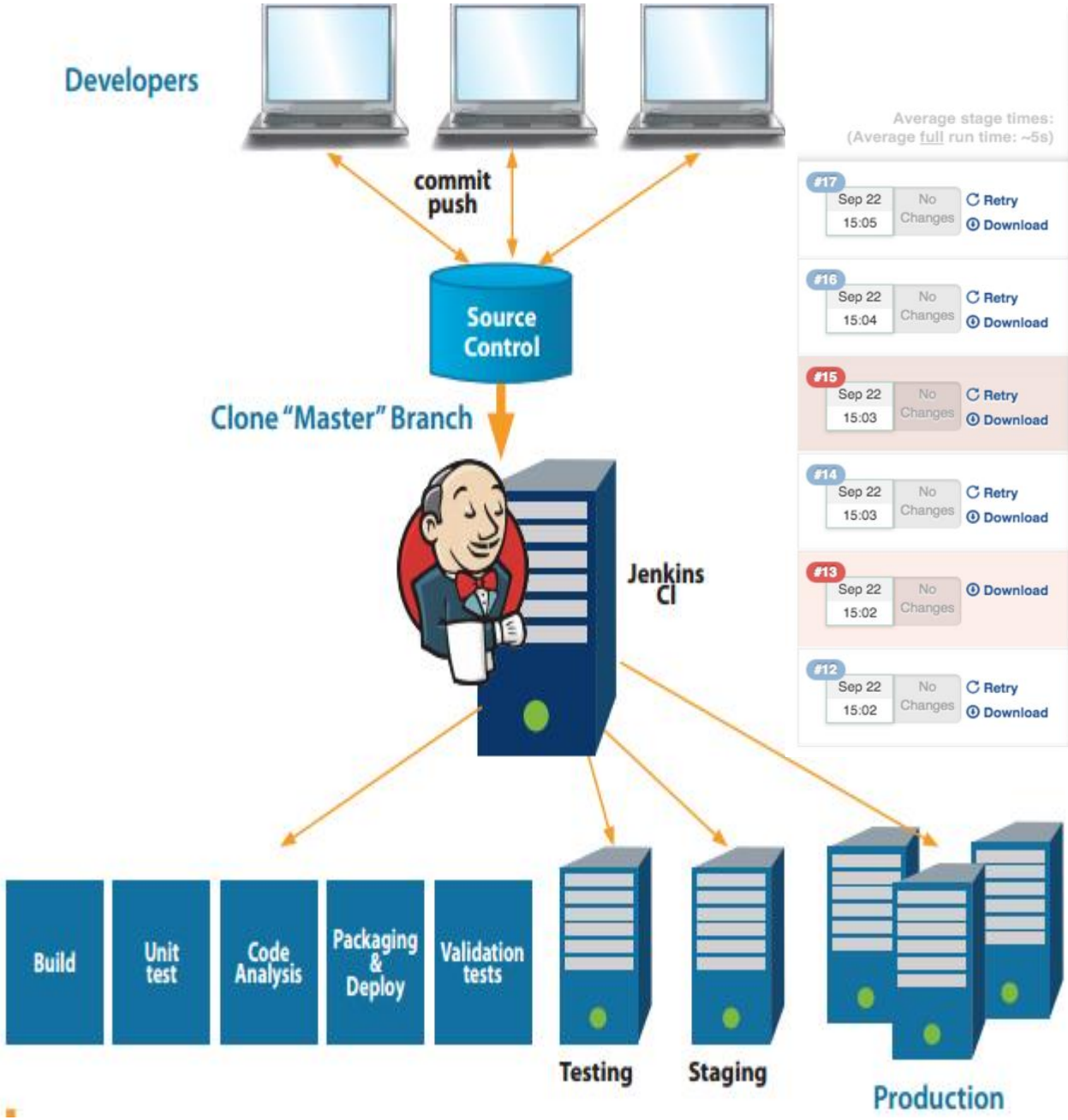
```
pipeline {
    agent any
    stages {
        stage('Build') {
            steps {

            }
        }
        stage('Test') {
            steps {

            }
        }
        stage('Deploy') {
            steps {

            }
        }
    }
}
```

Scripted

```
node {
    stage('Build') {

    }
    stage('Test') {

    }
    stage('Deploy') {

    }
}
```

CopaAirlines

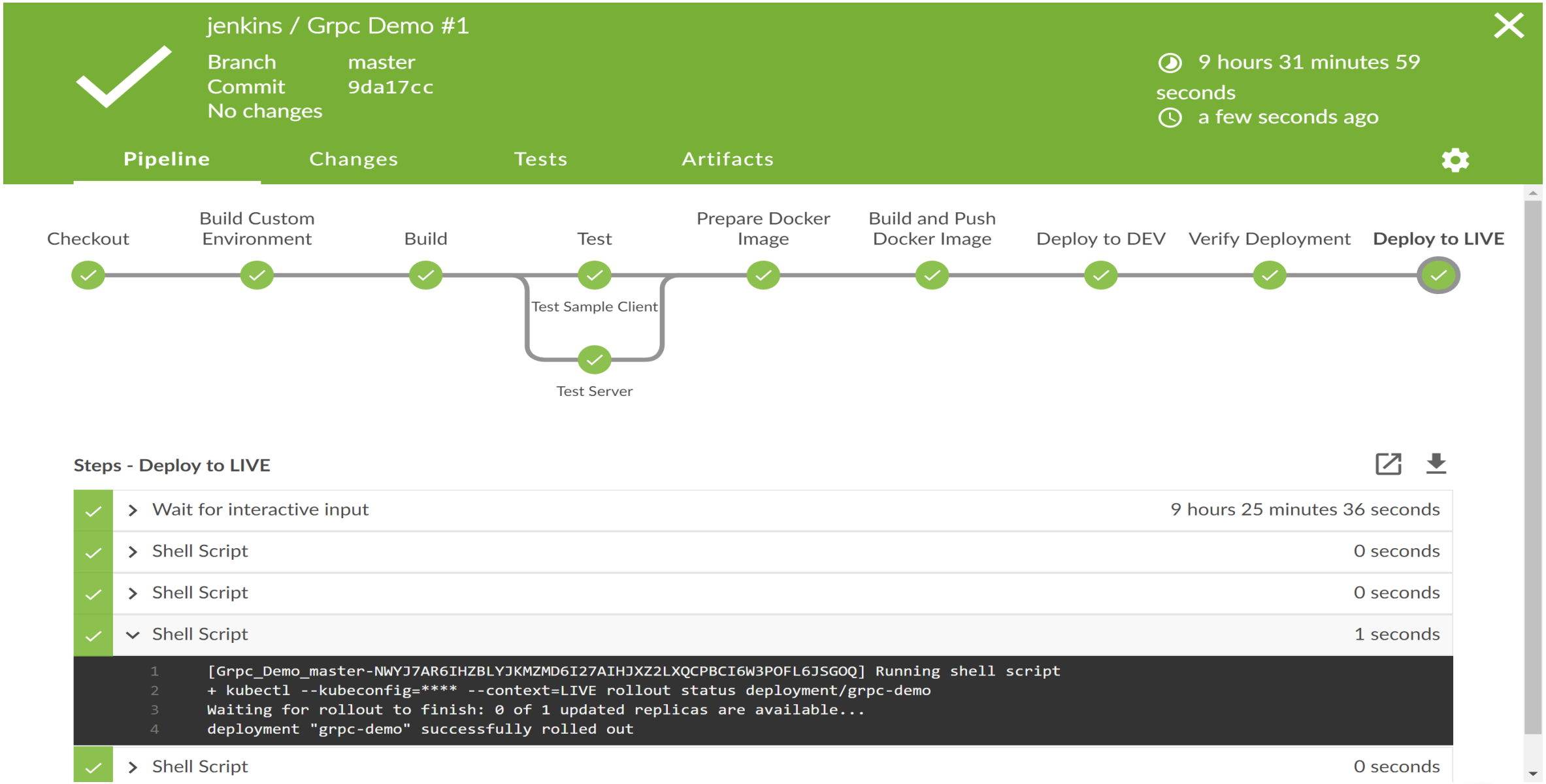
¿Qué es Jenkins?



	build	test: integration-&-quality	test: functional	test: load-&-security	approval	deploy: prod
Average stage times: (Average full run time: ~5s)	836ms	20min 43s	9ms	7ms	89ms	5ms
#17 Sep 22 15:05 No Changes Retry Download	538ms	10s	10ms	8ms	72ms (paused for 7s)	4ms
#16 Sep 22 15:04 No Changes Retry Download	479ms	6s	9ms	9ms	74ms (paused for 6s)	5ms
#15 Sep 22 15:03 No Changes Retry Download	922ms	6s	10ms	9ms failed		
#14 Sep 22 15:03 No Changes Retry Download	1s	8s	12ms	9ms	80ms (paused for 5s)	5ms
#13 Sep 22 15:02 No Changes Download	942ms	9s	13ms failed			
#12 Sep 22 15:02 No Changes Retry Download	1s	6s	13ms	11ms	111ms (paused for 5s) aborted	

CopaAirlines

¿Qué es Jenkins?



CopaAirlines

¿Cómo aprender Jenkins?

- En el sitio de Jenkins (https://www.jenkins.io/)

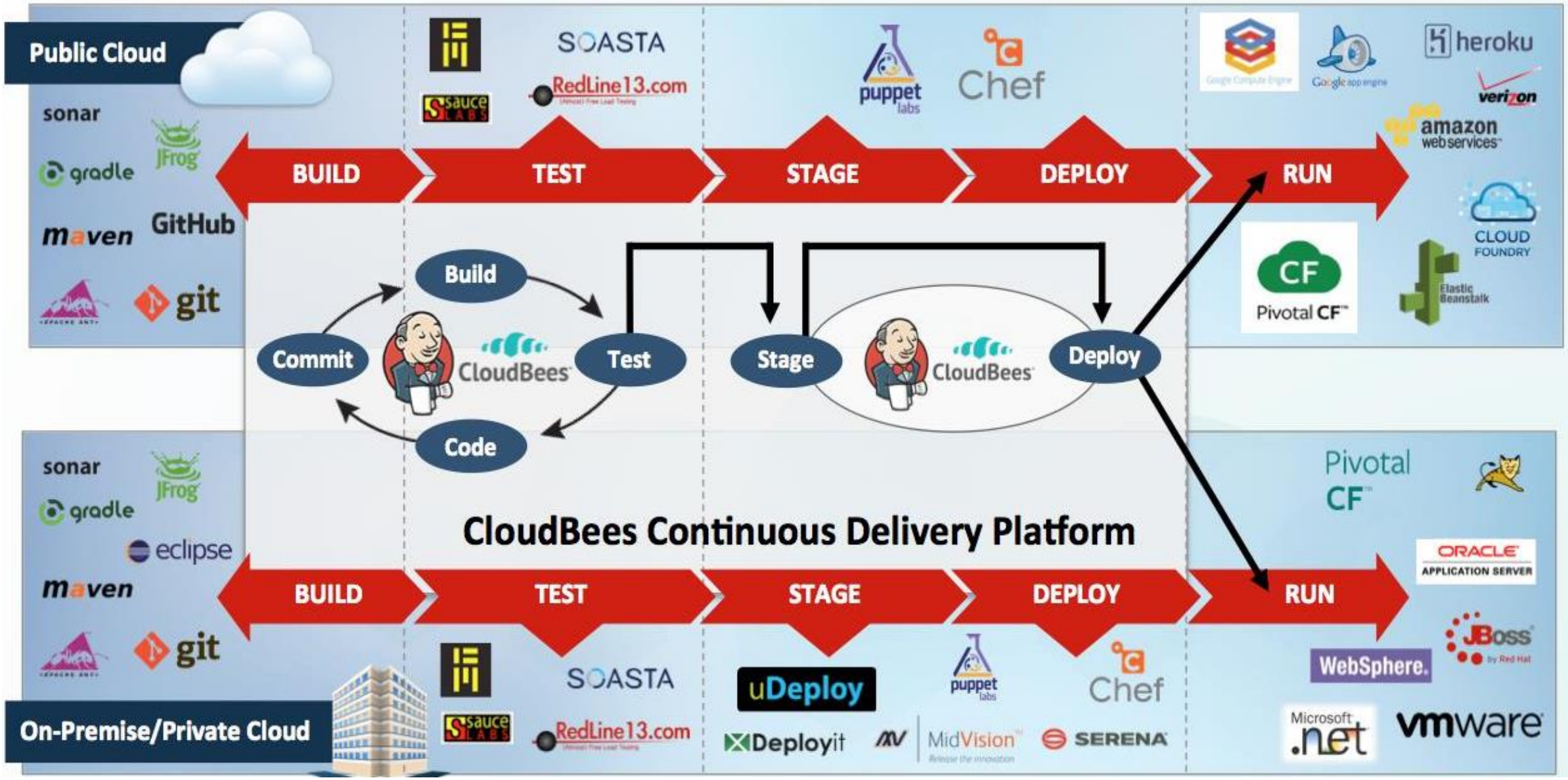
- Sobre los pipelines:
  - https://www.jenkins.io/doc/book/pipeline/

- **Buscando** en foros (https://stackoverflow.com/) o sitios (https://medium.com/).

- **Aplicarlo** a una **situación** en **2021** que consideras puedas resolver a través de la **orquestación**.

CopaAirlines

¿Cómo se integra con otras herramientas?



CopaAirlines

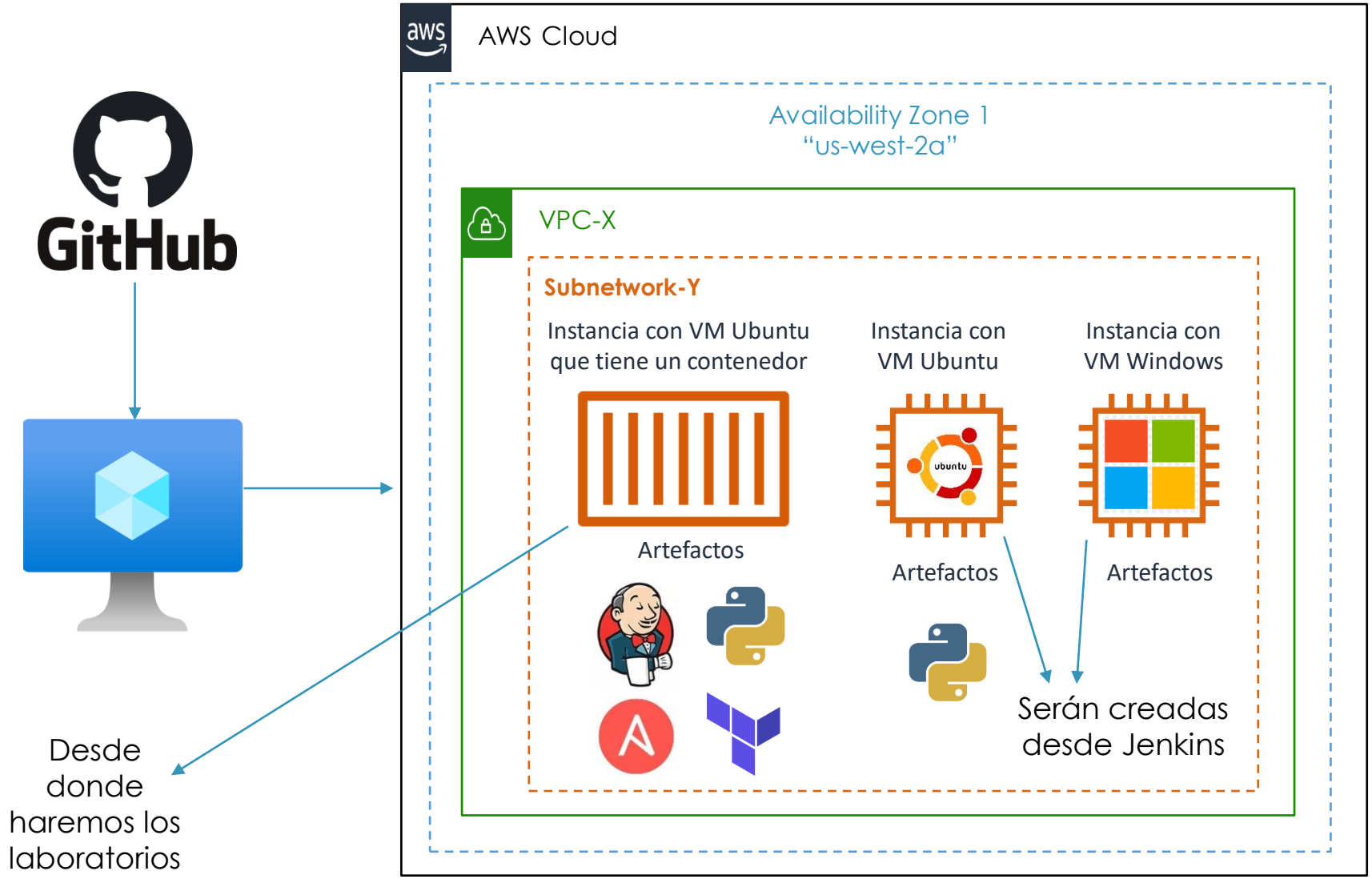
Laboratorios – Comienza la diversión

- En estos laboratorios vamos a aprender a:
  - Crear **pipelines** usando la interfaz gráfica de Jenkins:
  - Integrarlo con **GitHub**
    - Usar **Webhooks**
  - Crear **Jenkinfiles**
  - Ejecutarlos desde la línea de comando usando **curl**:

- **Nota: reutilizaremos códigos de los repositorios vistos en los cursos anteriores**

CopaAirlines

Laboratorios – Comienza la diversión



Laboratorios

- En **Sharepoint** se encuentra el archivo con todo el inventario de VMs creadas, selecciona **1 VM** (registra en el Excel tu **nombre**).

- La VM ya tiene instalado **Jenkins 2.285**, el cual corre usando **Docker** y dentro del contenedor donde se ejecuta Jenkins se instaló **Docker**.

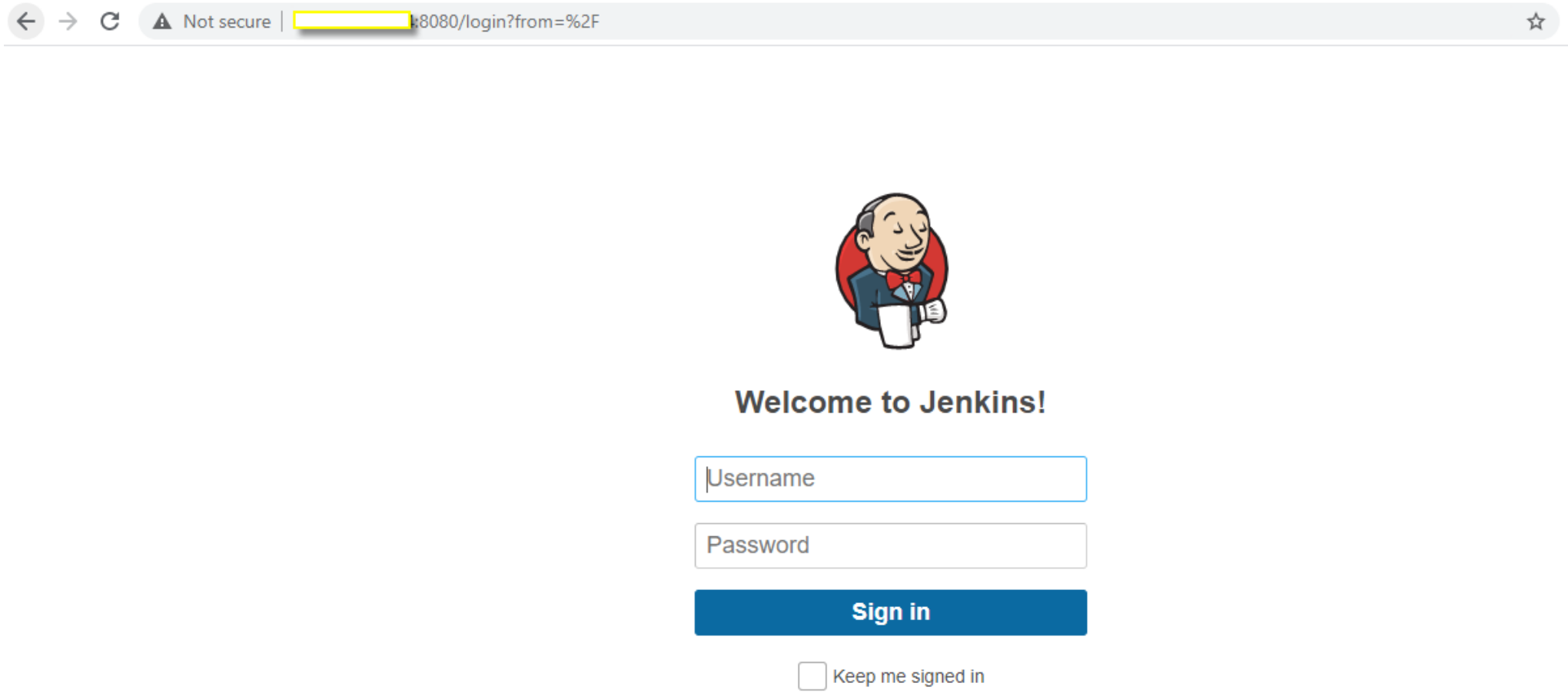
- Conéctate por SSH a la VM y ejecuta el siguiente comando para subir Jenkins:
  - ssh -v -l ubuntu -i <llave> <ip_publica_vm>
  - docker start jenkins

- Verifica que Jenkins esté corriendo:
  - docker ps

```
CONTAINER ID   IMAGE                          COMMAND                  CREATED       STATUS          PORTS
                                NAMES
[redacted]     jenkins/jenkins:2.285-slim     "/sbin/tini -- /usr/…"   11 days ago   Up 18 seconds   0.0.0.0:8080->8080/tcp, :::8080->8080/tcp, 0.0.0.0:49153
->50000/tcp, :::49153->50000/tcp   jenkins
```

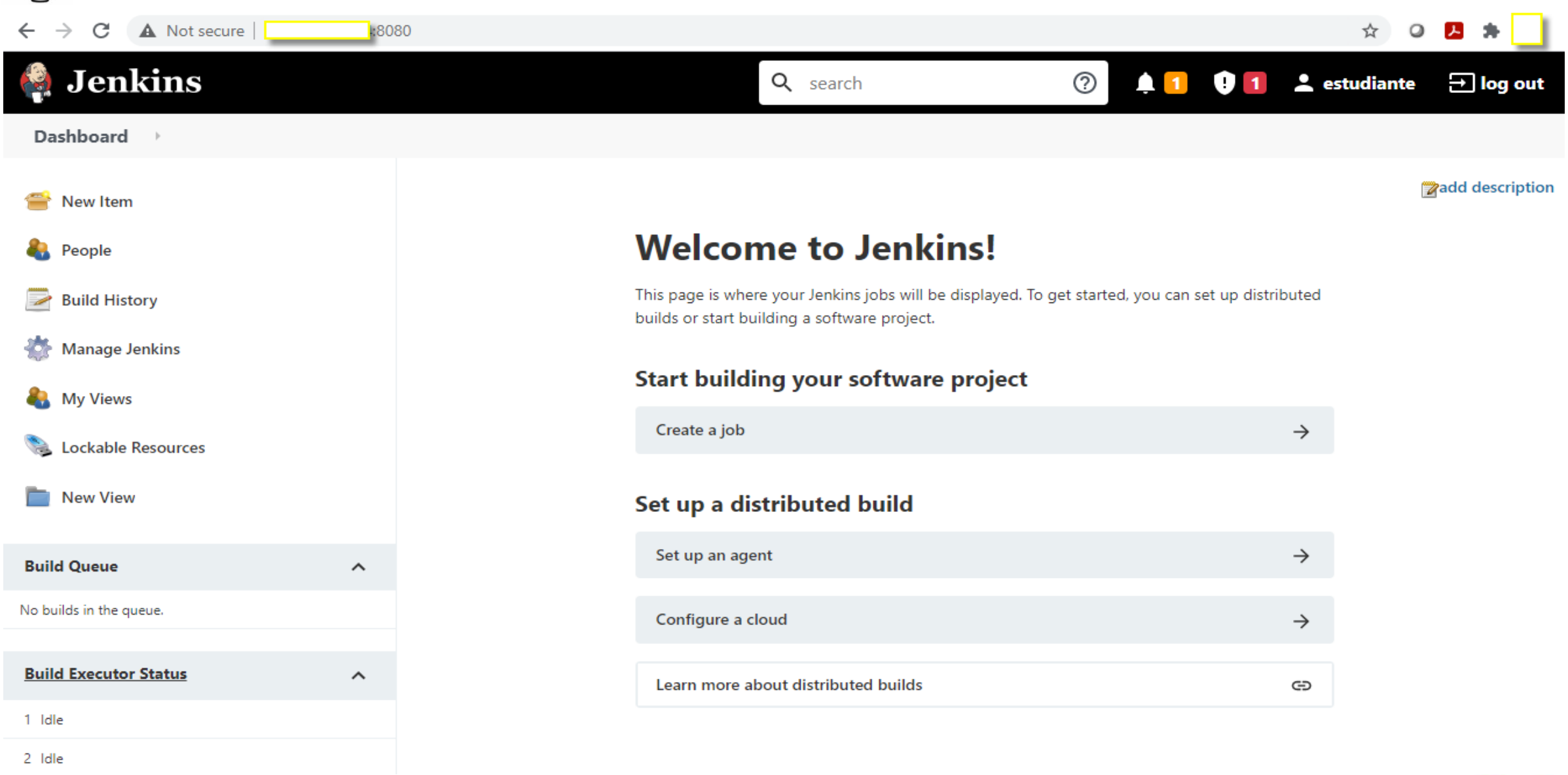
Laboratorios – Comienza la diversión

- Conéctate usando el navegador a Jenkins:
  - http://<ip_publica_vm>:8080



- En la sesión se facilitará la información para accesar a Jenkins

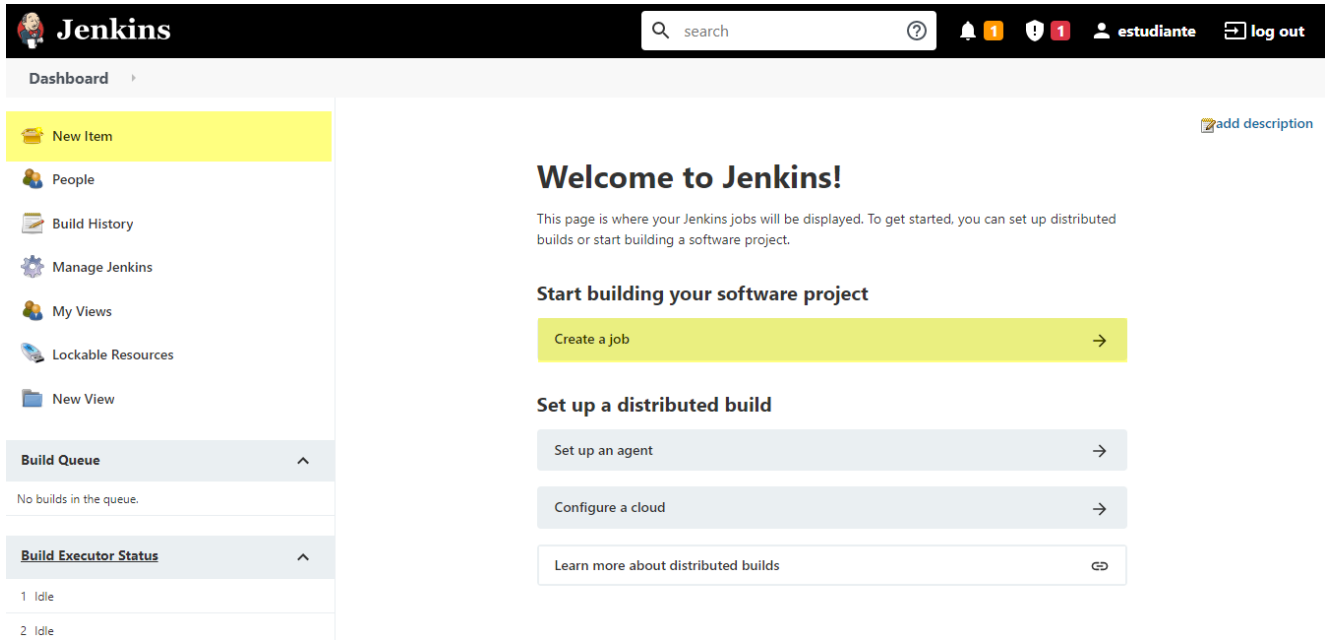
Laboratorios – Comienza la diversión



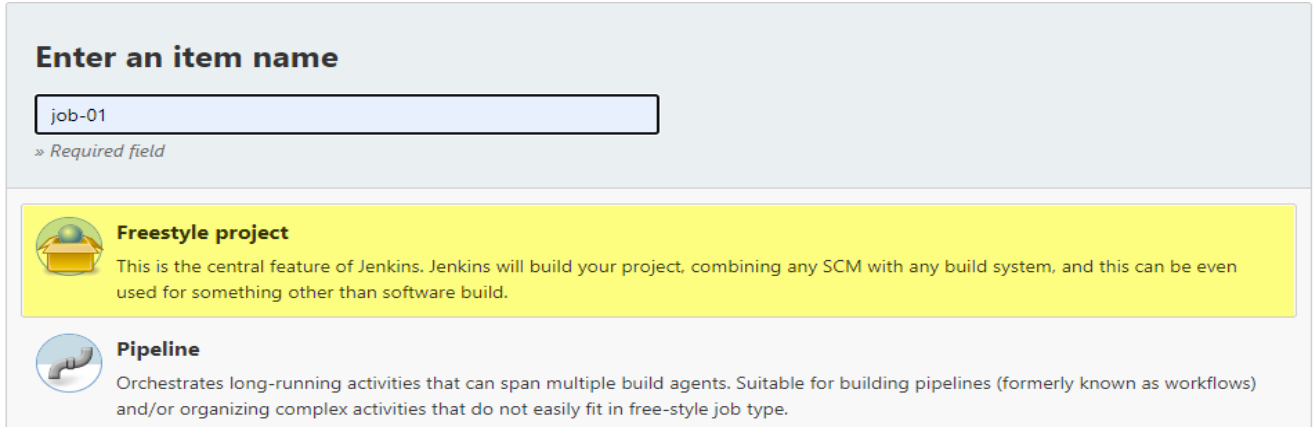
CopaAirlines

Laboratorios – Comienza la diversión

- Vamos a crear nuestro **primer job**

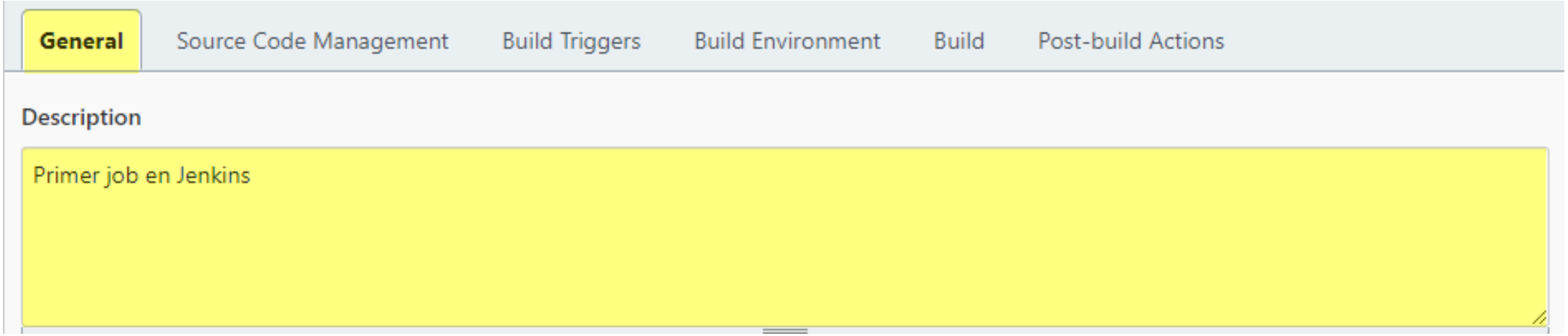


- El cual se llamará "**job-01**" y será del tipo "**Freestyle project**"

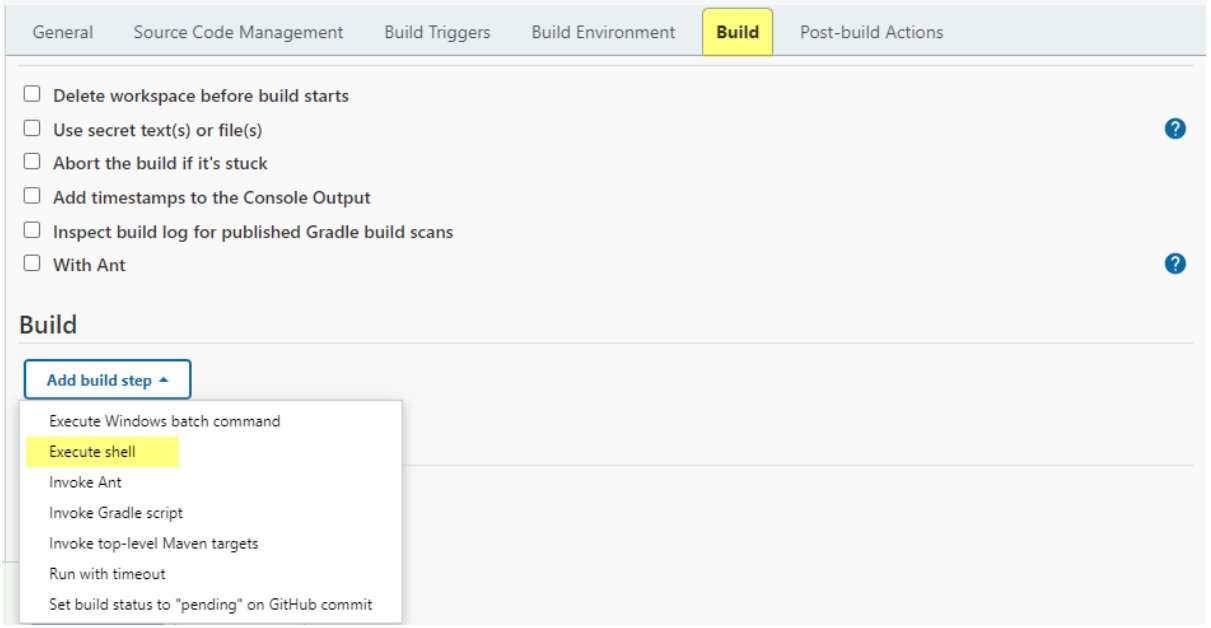


Laboratorios – Comienza la diversión

- Agrégale un comentario en el campo "**Description**" del tab "**General**"

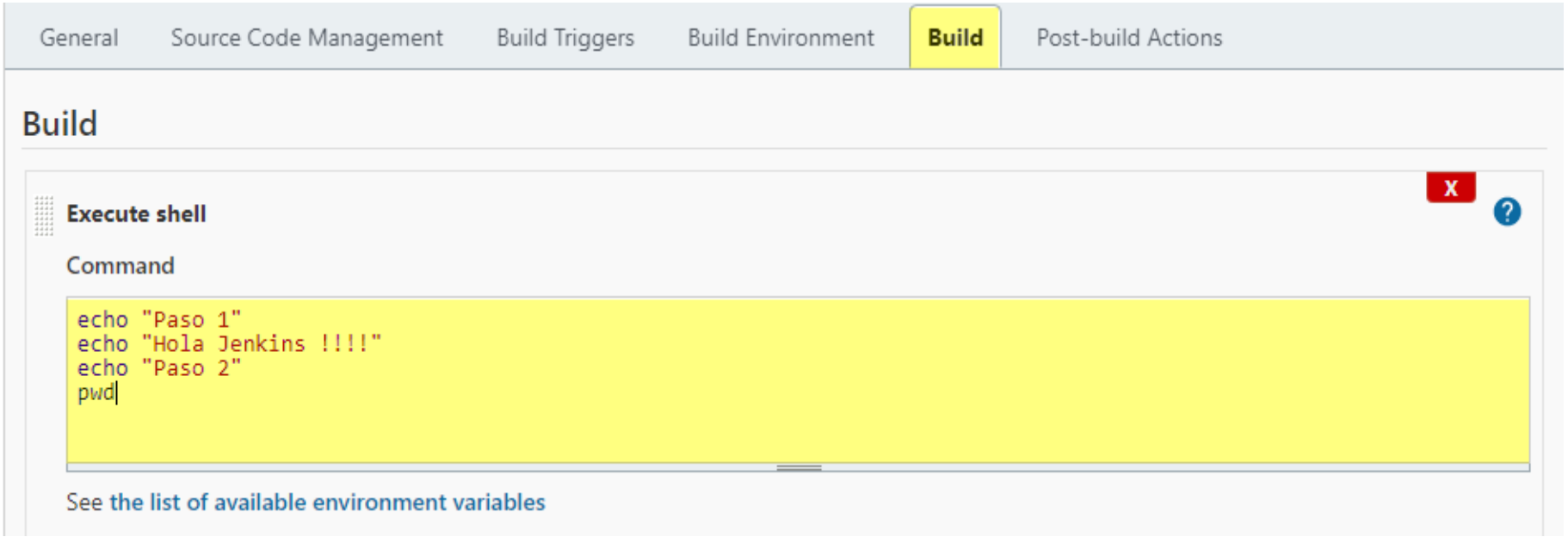


- En el tab "**Build**" selecciona "**Execute shell**"

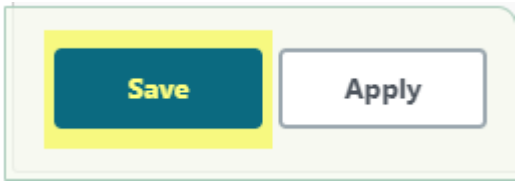


Laboratorios – Comienza la diversión

- El job ejecutará estos comandos:
  - echo "Paso 1"
  - echo "Hola Jenkins !!!!"
  - echo "Paso 2"
  - pwd



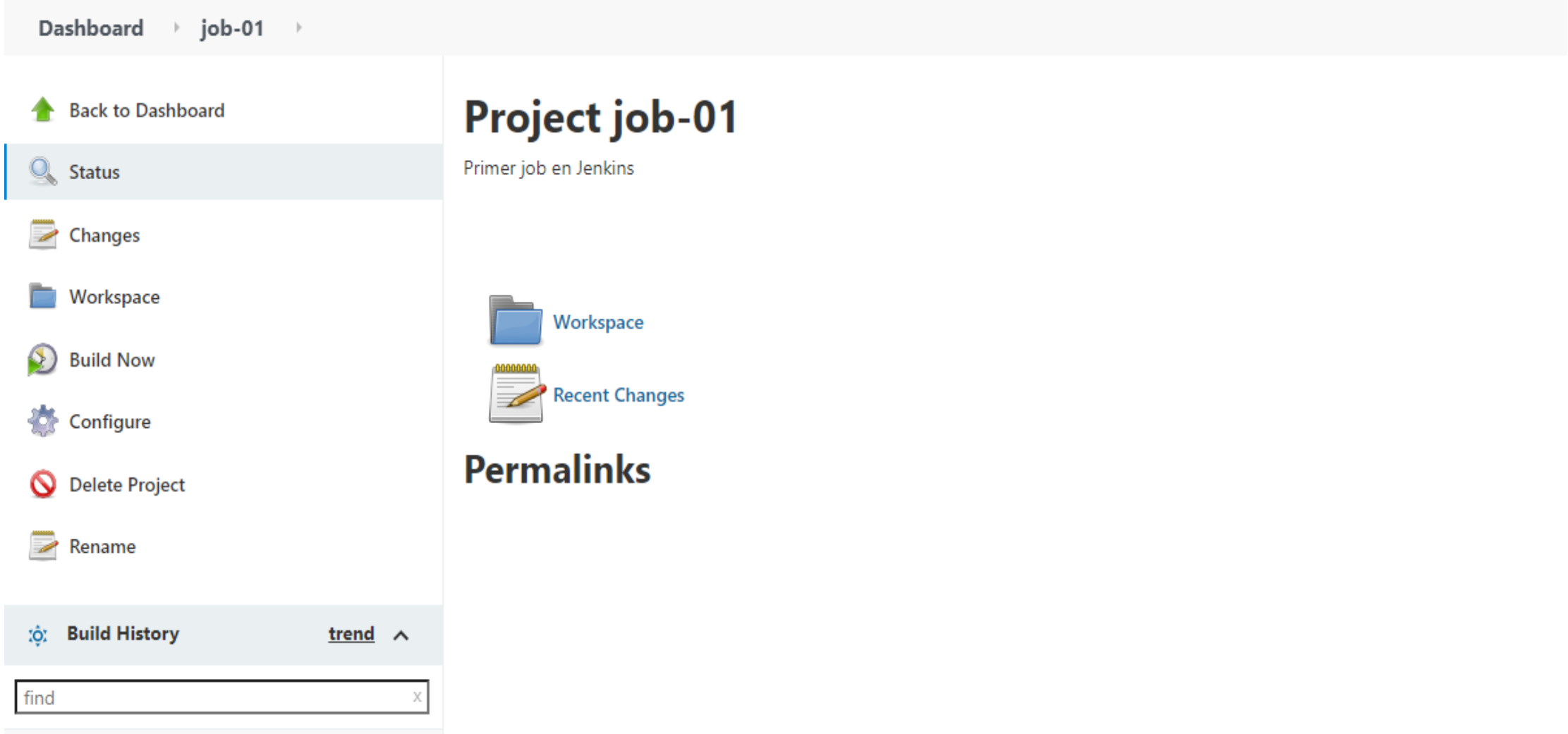
- Salva para seguir con la ejecución



CopaAirlines

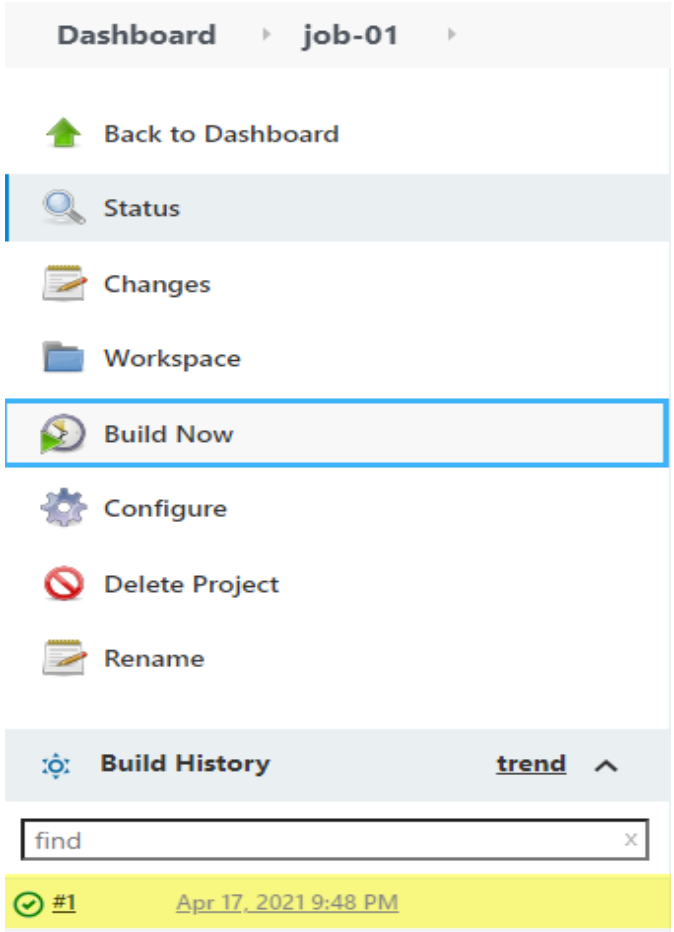
Laboratorios – Comienza la diversión

- Así aparece el job creado



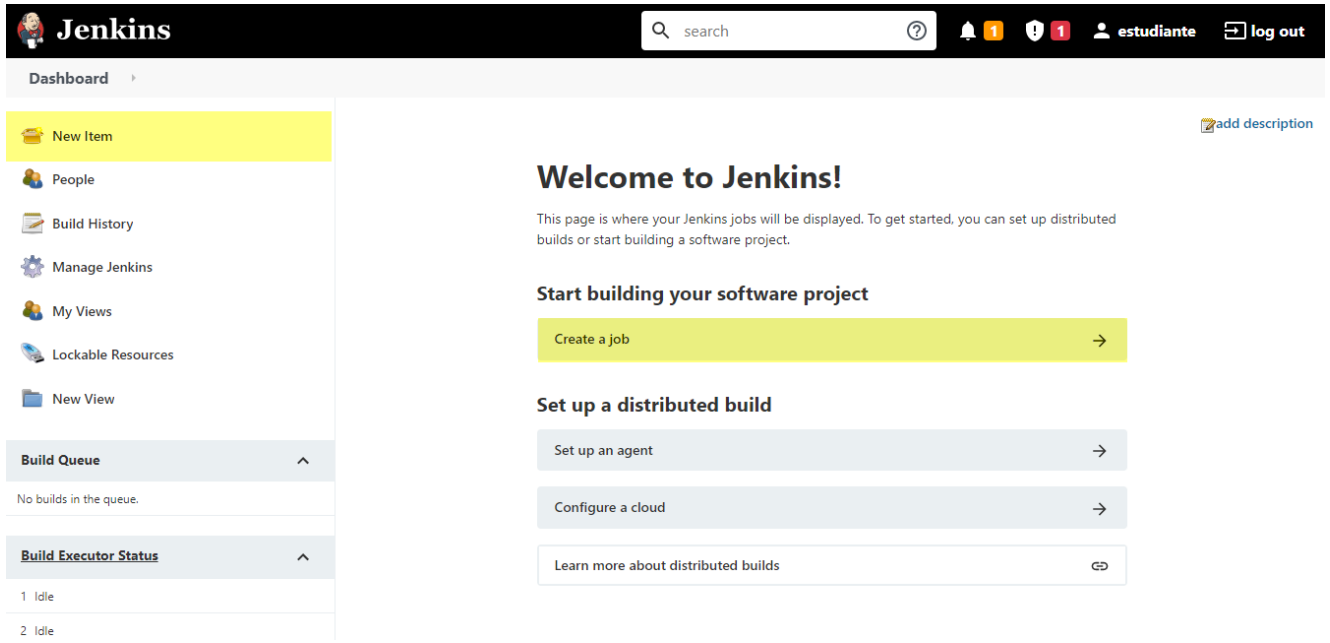
Laboratorios – Comienza la diversión

- Da clic en "**Build Now**" para ejecutarlo, en la sección "**Build History**" aparece un número que identifica la ejecución del job (en este caso **#1**), da clic en ese número para ver el resultado, después da clic en "**Console Output**"

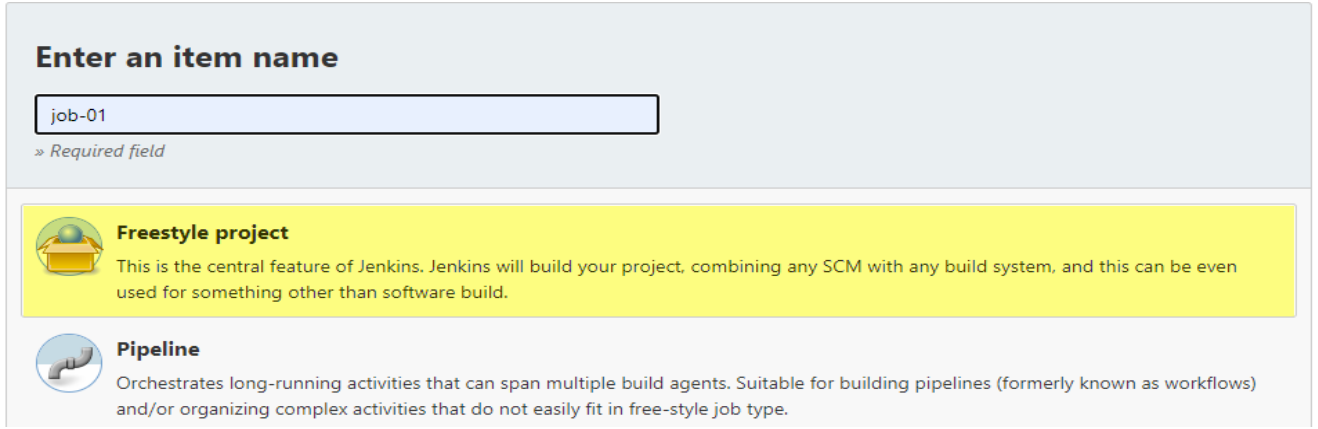


Laboratorios – Comienza la diversión

- Vamos a crear nuestro **segundo job**

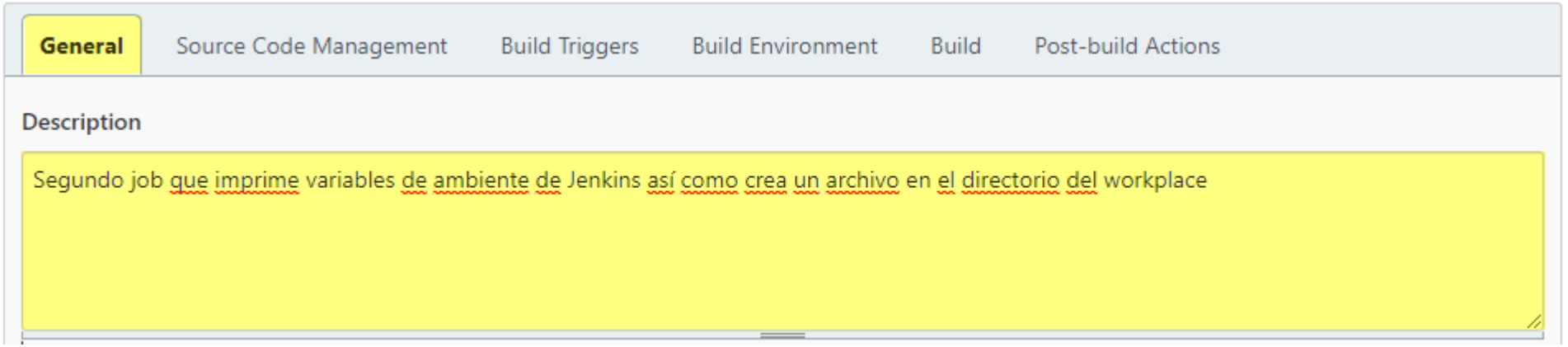


- El cual se llamará "**job-02**" y será del tipo "**Freestyle project**"

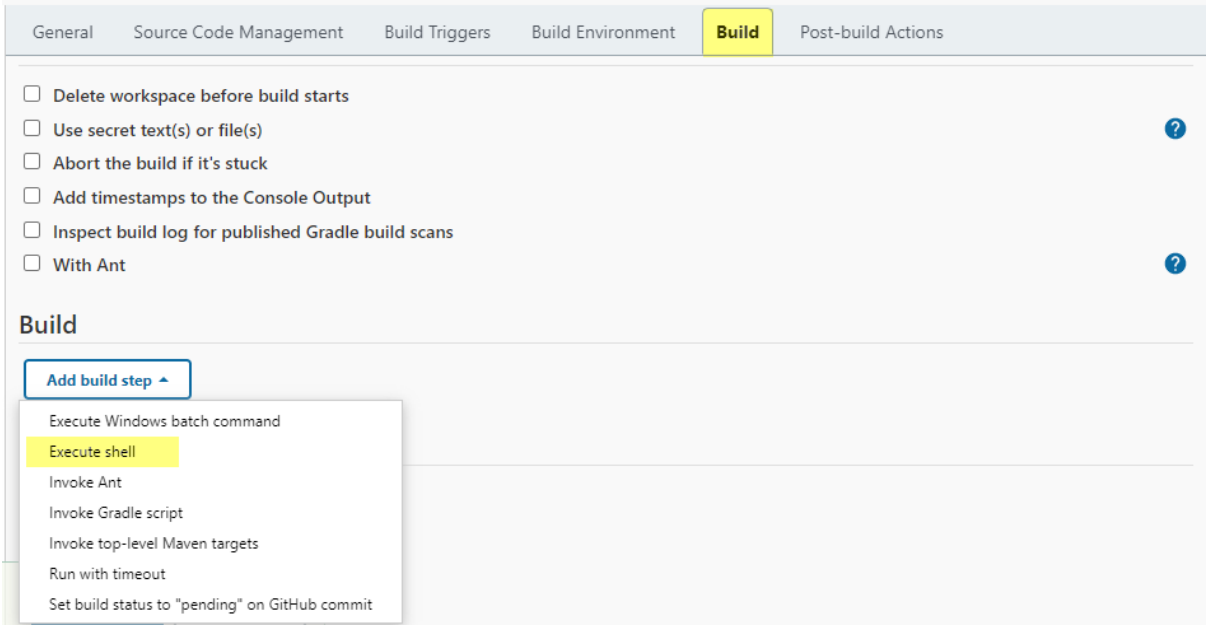


Laboratorios – Comienza la diversión

- Agrégale un comentario en el campo "**Description**" del tab "**General**"



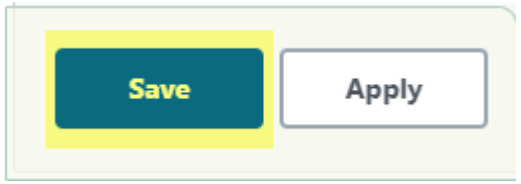
- En el tab "**Build**" selecciona "**Execute shell**"



Laboratorios – Comienza la diversión

- El job ejecutará estos comandos:
  - echo Paso 1 del job-02
  - echo $BUILD_ID
  - echo Paso 2 del job-02
  - echo $WORKSPACE
  - echo Paso 3 del job-02
  - ls -lR $WORKSPACE
  - echo Paso 4 del job-02
  - echo "Que ondas aprendiendo Jenkins !!!" > $WORKSPACE/file.txt
  - echo Paso 5 del job-02
  - ls -lR $WORKSPACE
  - echo Paso 6 del job-02
  - cat $WORKSPACE/file.txt

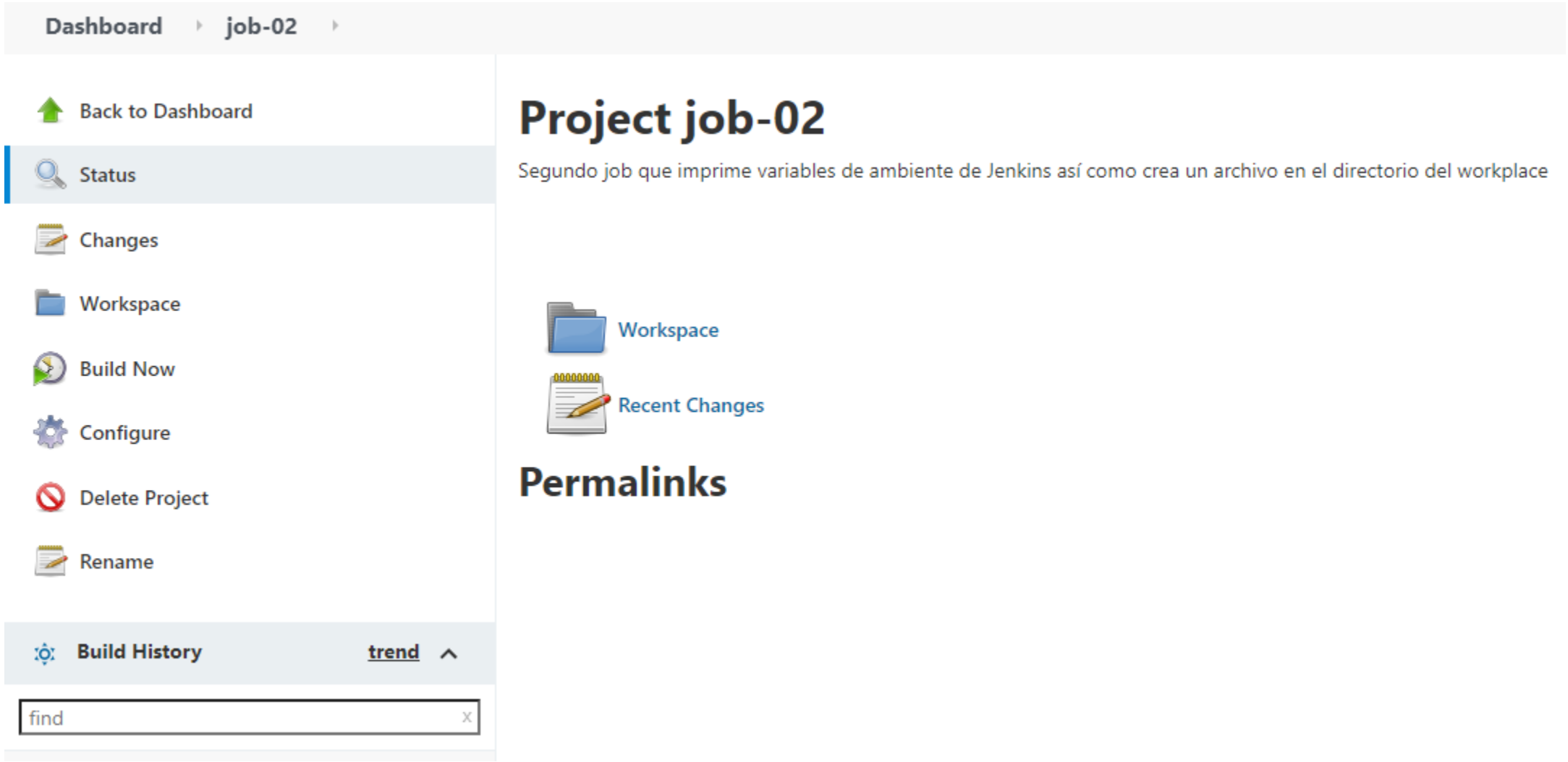
- Salva para seguir con la ejecución



CopaAirlines

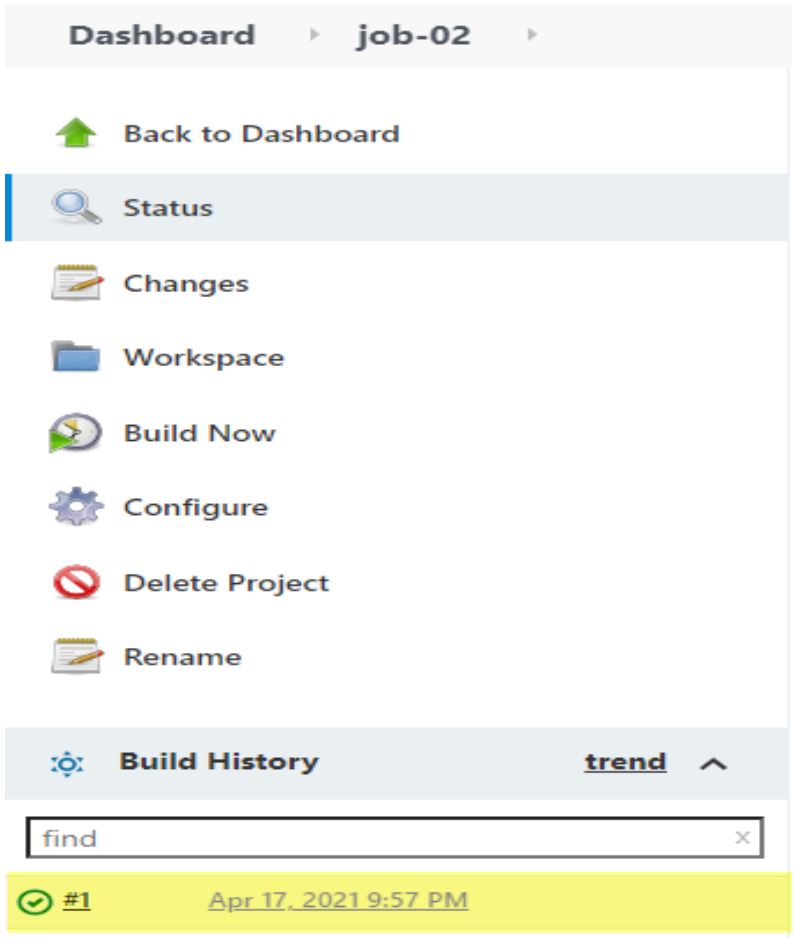
Laboratorios – Comienza la diversión

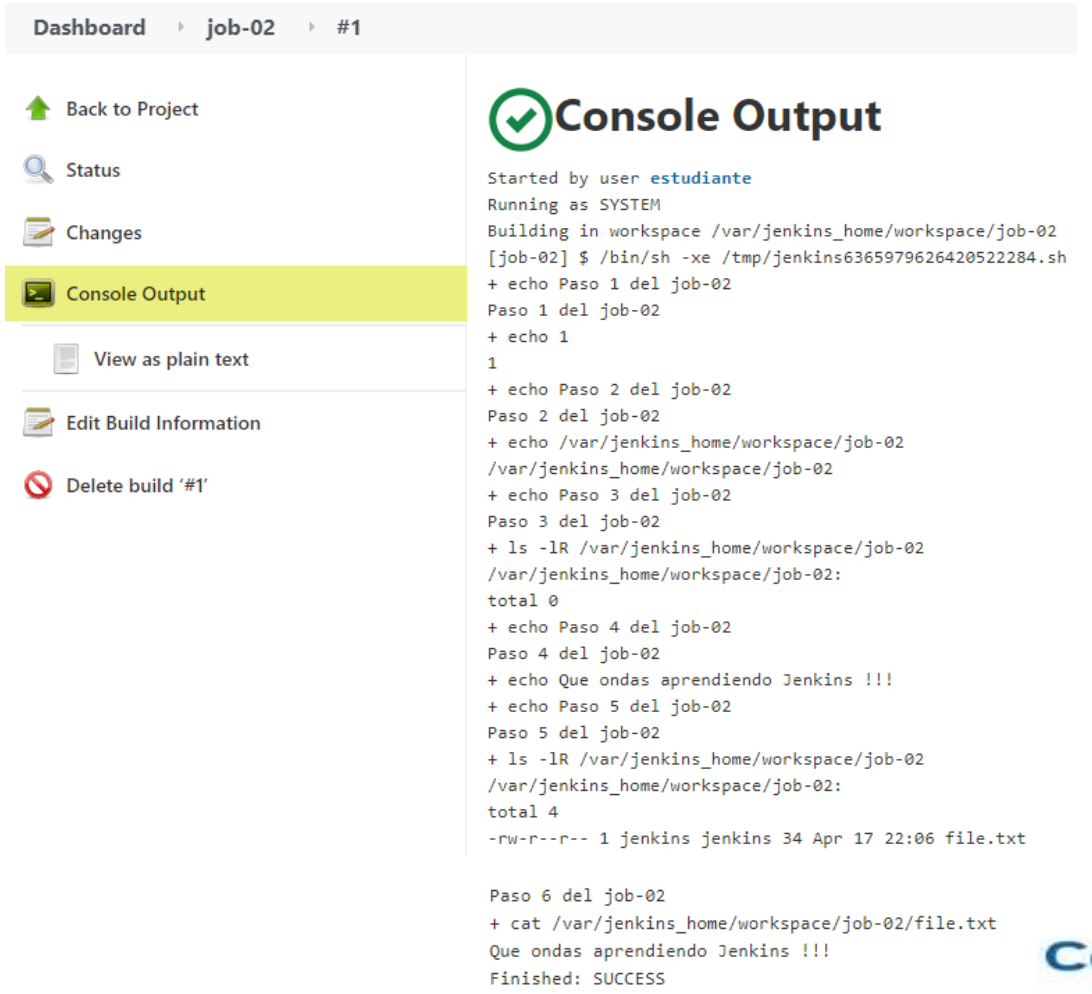
- Así aparece el job creado



Laboratorios – Comienza la diversión

- Da clic en "**Build Now**" para ejecutarlo, en la sección "**Build History**" aparece un número que identifica la ejecución del job (en este caso **#1**), da clic en ese número para ver el resultado, después da clic en "**Console Output**"





Laboratorios – Comienza la diversión

- Vamos a crear nuestro **tercer job**
  - Nombre: "**job-03**"
  - Descripción: "**Job para obtener información de docker, ansible, python y terraform así como los contenedores corriendo**"
  - Tipo "**Freestyle project**"
  - Ejecutará:
  - docker version
  - docker ps
  - docker image ls
  - ansible --version
  - python3 --version
  - terraform --version

CopaAirlines

- **Reto 1 – Crea un job**
  - Nombre "**reto_1**"
  - Tipo: Free Style
  - Que imprima
    - El día del año en curso
    - El nombre del día de la semana
    - El nombre completo del mes
    - El año

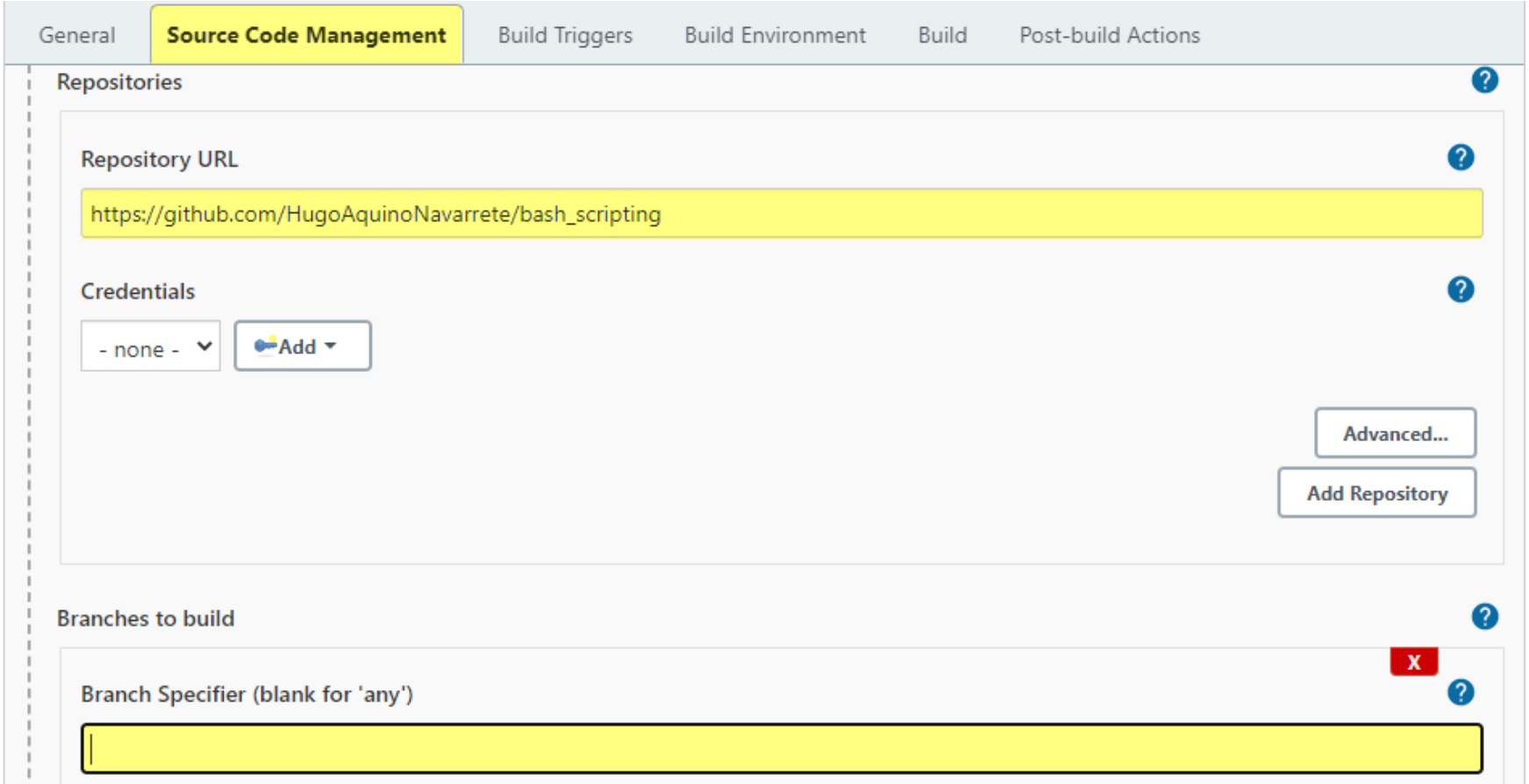
Laboratorios – Comienza la diversión

- Vamos a crear un **cuarto job**
  - Nombre: "**job-04**"
  - Descripción: "**Job para clonar el repositorio de GitHub del curso de bash y posteriormente ejecuta 3 scripts**"
  - Tipo "**Freestyle project**"

CopaAirlines

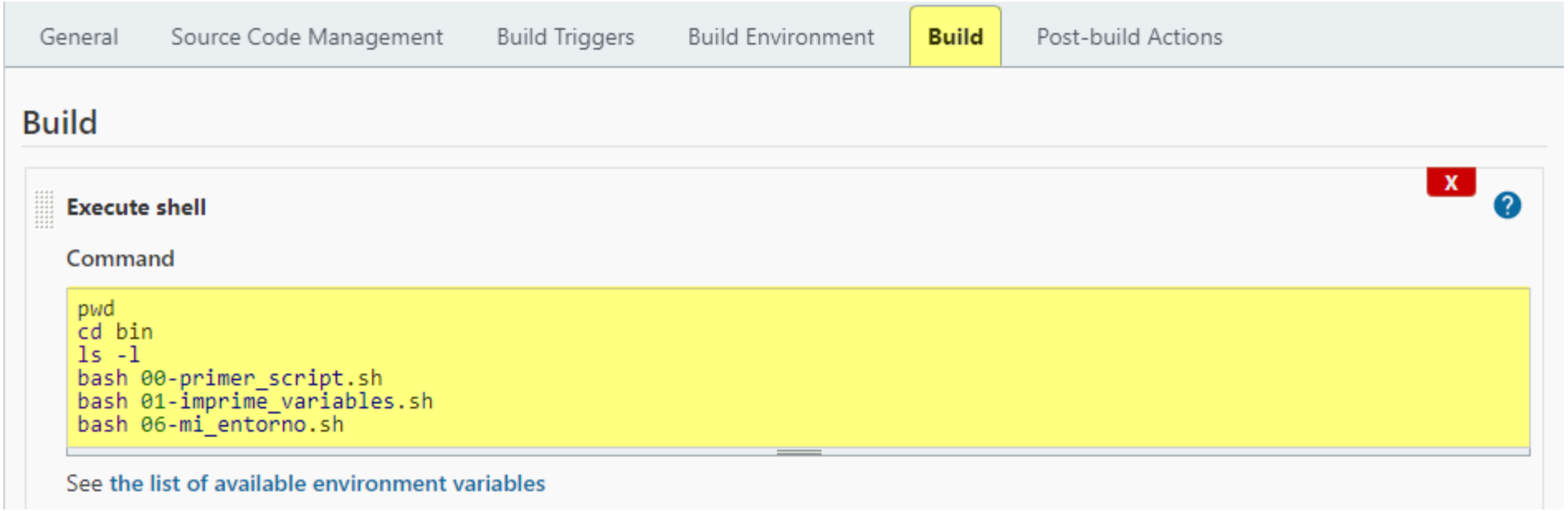
Laboratorios – Comienza la diversión

- En la sección de "**Source Code Management**" en "Repository URL" escribe "**https://github.com/HugoAquinoNavarrete/bash_scripting**" y en "**Branch Specifier**" déjalo en blanco.



Laboratorios – Comienza la diversión

- En la sección de "**Build**" copia estos comandos:
  - pwd
  - cd bin
  - ls -l
  - bash 00-primer_script.sh
  - bash 01-imprime_variables.sh
  - bash 06-mi_entorno.sh



- Sálvalo y córrelo

CopaAirlines

Laboratorios – Comienza la diversión

Dashboard › job-04 › #1

- Back to Project
- Status
- Changes
- Console Output
  - View as plain text
- Edit Build Information
- Delete build '#1'
- Git Build Data

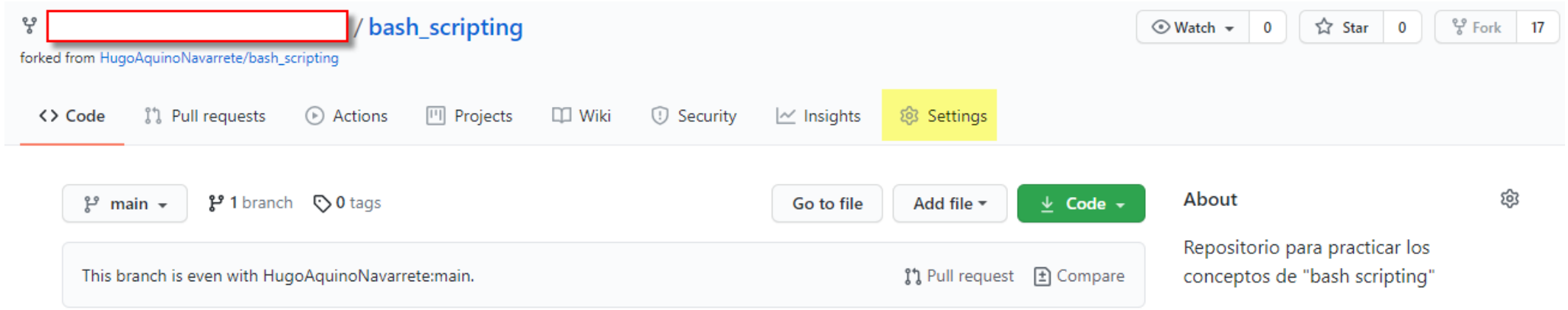
Console Output

```
Started by user estudiante
Running as SYSTEM
Building in workspace /var/jenkins_home/workspace/job-04
The recommended git tool is: NONE
No credentials specified
 > git rev-parse --resolve-git-dir /var/jenkins_home/workspace/job-04/.git # timeout=10
Fetching changes from the remote Git repository
 > git config remote.origin.url https://github.com/HugoAquinoNavarrete/bash_scripting # timeout=10
Fetching upstream changes from https://github.com/HugoAquinoNavarrete/bash_scripting
 > git --version # timeout=10
 > git --version # 'git version 2.20.1'
 > git fetch --tags --force --progress -- https://github.com/HugoAquinoNavarrete/bash_scripting +refs/heads/*:refs/remotes/origin/* #
timeout=10
Seen branch in repository origin/main
Seen 1 remote branch
 > git show-ref --tags -d # timeout=10
Checking out Revision 025e07bfb4b0cf4b4edd69f6789825c549643ab8 (origin/main)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 025e07bfb4b0cf4b4edd69f6789825c549643ab8 # timeout=10
Commit message: "Ajuste a presentación pare reflejar cambio en sitio para hacer fork al repositorio"
First time build. Skipping changelog.
[job-04] $ /bin/sh -xe /tmp/jenkins2143680279553034611.sh
+ pwd
/var/jenkins_home/workspace/job-04
+ cd bin
+ ls -l

+ bash 06-mi_entorno.sh
El nombre del usuario es:
El nombre del equipo es: d7201c6c9a88
El shell que usamos es: /bin/bash
El directorio donde nos encontramos es: /var/jenkins_home/workspace/job-04/bin
La ruta de búsqueda de comandos es: /opt/java/openjdk/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
Finished: SUCCESS
```

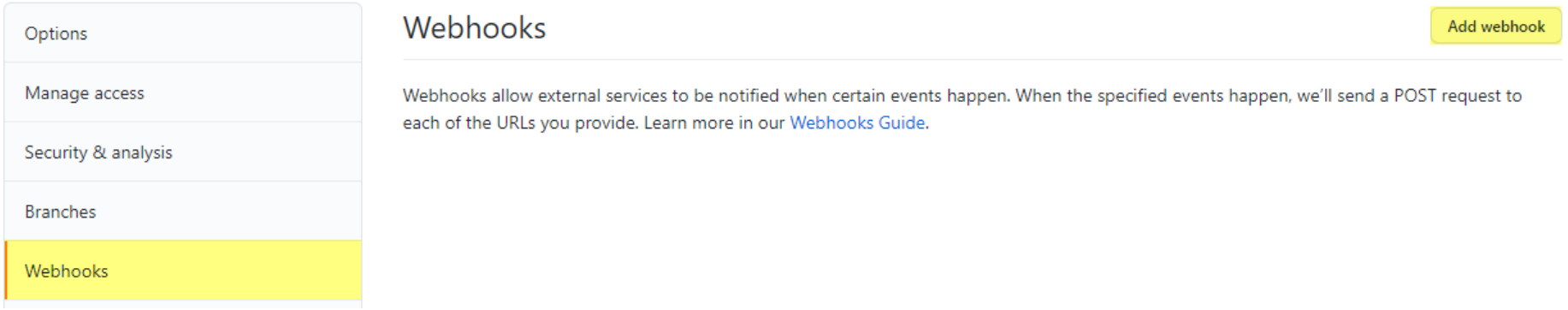
Laboratorios – Comienza la diversión

- En tu cuenta de **GitHub**, haz un **fork** del siguiente repositorio:
  - https://github.com/HugoAquinoNavarrete/bash_scripting

- Dentro del repositorio, da clic en **Settings**:

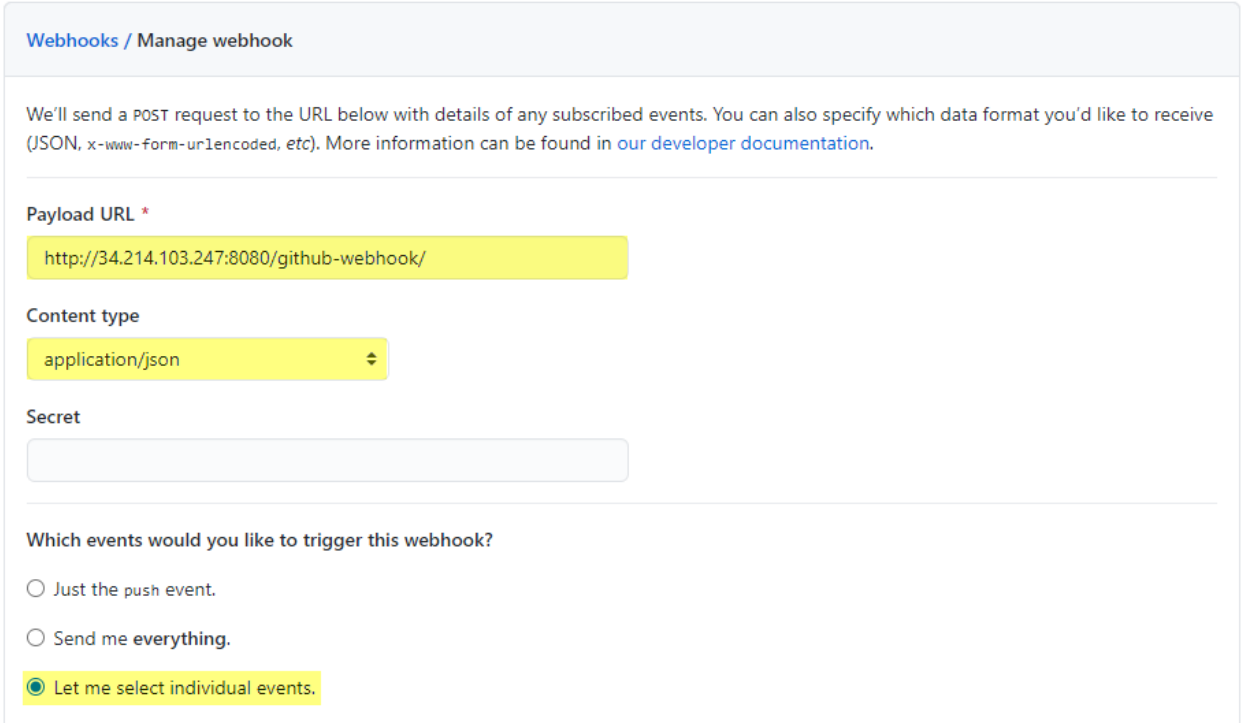


- Da clic en **Webhooks** y después en "**Add webhook**"

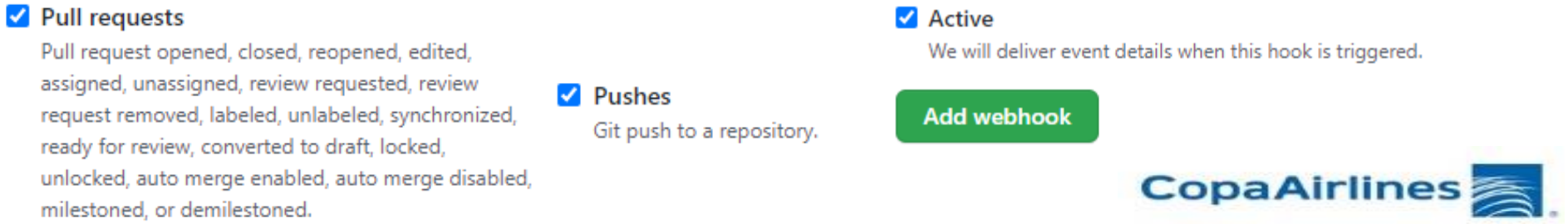


Laboratorios – Comienza la diversión

- En "**Payload URL**" completa esta información:
  - http://<ip_publica_vm>:8080/github-webhook/



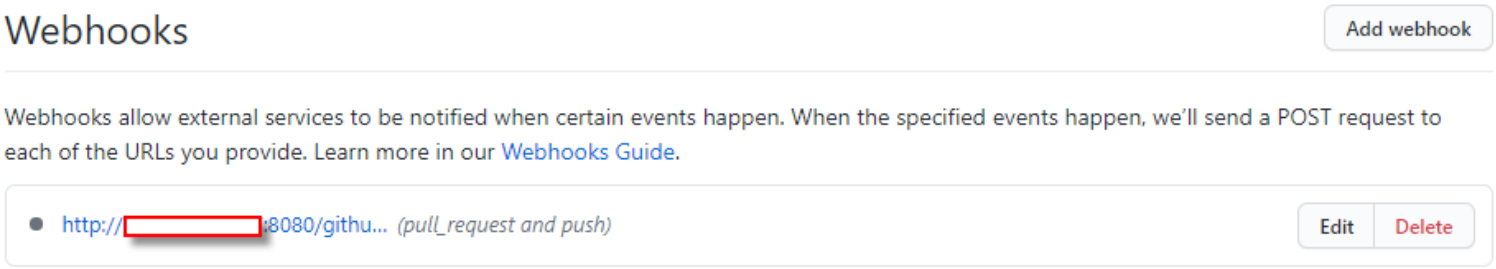
- Selecciona "**Pull requests**" y "**Pushes**", finalmente da clic en "**Add webhook**"



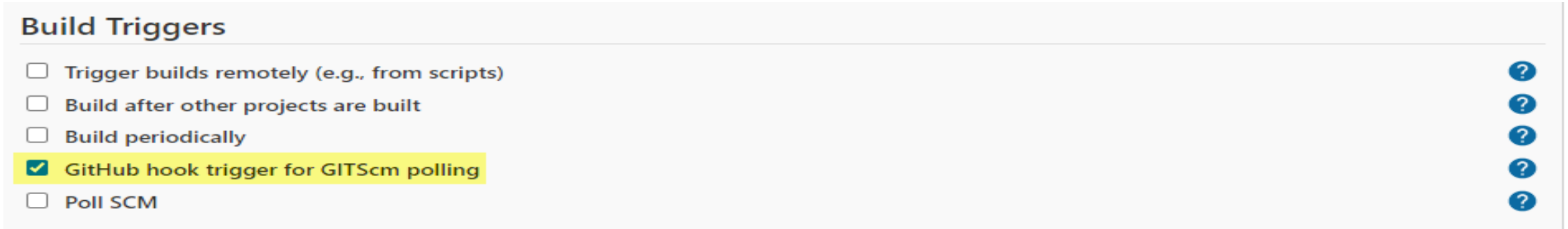
CopaAirlines

Laboratorios – Comienza la diversión

- En la pantalla aparecerá el **webhook** creado:



- En **Jenkins** vamos a crear un **quinto job**
  - Nombre: "**job-05**"
  - Tipo "**Freestyle project**"
  - Descripción: "**Job para verificar el uso de un webhook**"
  - Source Code Management: "**https://github.com/<usuario>/bash_scripting**"
  - Branch Specifier: dejarlo vacío
  - En Build Triggers selecciona "**GitHub hook trigger for GITScm polling**"



Laboratorios – Comienza la diversión

- En **Build**, selecciona **"Execute a script"** que contenga
  - echo "Aprendiendo el uso de los webhooks"
  - echo "Al hacer un commit al repositorio"
  - echo "Se ejecutará este job"
  - echo "Este es el ID:" $BUILD_ID
  - date

- En tu laptop clona el repositorio
  - git clone https://github.com/<usuario_git_hub>/bash_scripting

- Edita el archivo "**README.md**", agrega al final del archivo una línea en blanco y posteriormente haz un commit al reposotorio:
  - nano README.md
  - git add README.md
  - git commit –m "Ajuste en README.md para ver funcionamiento de un webhook"
  - git push

CopaAirlines

Laboratorios – Comienza la diversión

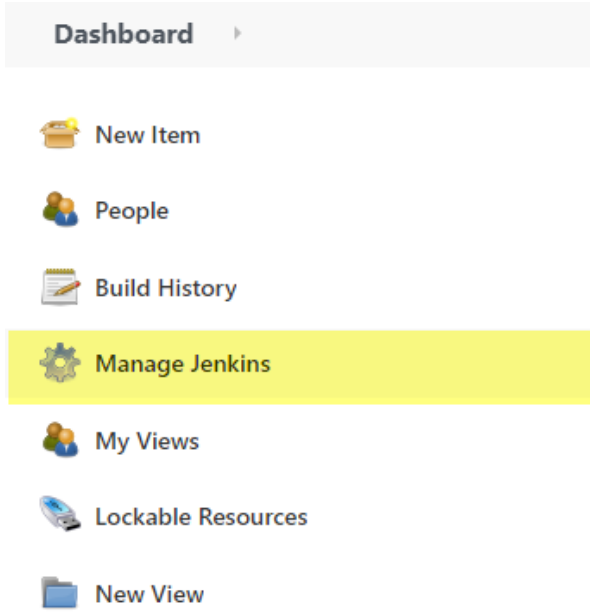
- En **Jenkins** vamos a crear un **sexto job**
  - Nombre: "**job-06**"
  - Tipo "**Freestyle project**"
  - Copy from "**job-05**"
  - Descripción: "**Job para verificar el uso de un webhook**"
- En **Build**, dentro de **"Execute a script"** borra lo anterior y agrega
  - cd bin
  - bash 11-mlb_equipos.sh
- En el repositorio clonado en tu laptop edita el archivo **year.txt** y cámbiale el valor y una vez que hagas **commit** mira lo que aparece en Jenkins
  - nano input/year.txt
  - git add input/year.txt
  - git commit –m "Ajuste en year.txt para ver funcionamiento de un webhook"
  - git push

CopaAirlines

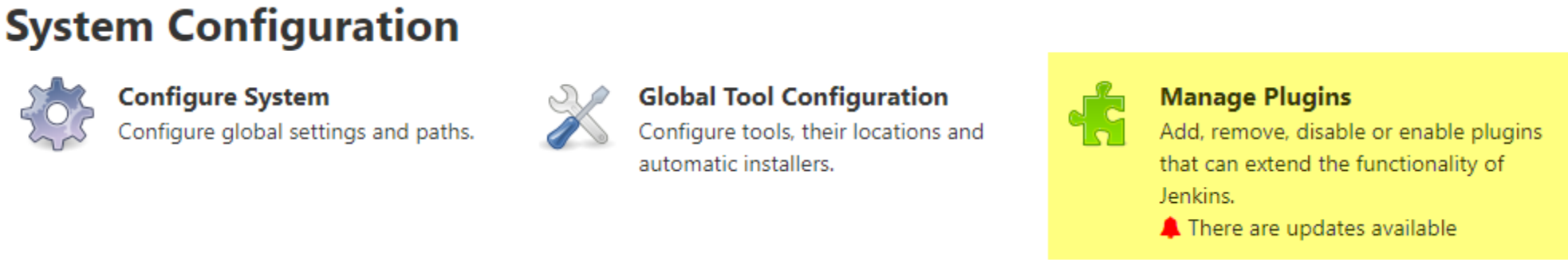
- **Reto 2 – Crea un job**
  - Nombre "reto_2"
  - Tipo: Free Style
  - Haz "**Fork**" al siguiente repositorio:
    - https://github.com/HugoAquinoNavarrete/python_scripting
  - Haz lo siguiente:
    - Configura un "**webhook**" (reutiliza el que ya creaste)
    - En "**Build**" se tienen que ejecutar 2 scripts del directorio "**bin**"
    - Clona el repositorio en tu laptop, edita el archivo "**README.md**" y haz un commit en git para lograr una ejecución automática del job.

Laboratorios – Comienza la diversión

- Vamos a instalar un **plugin**. Da clic en "**Dashboard**" -> "**Manage Jenkins**"



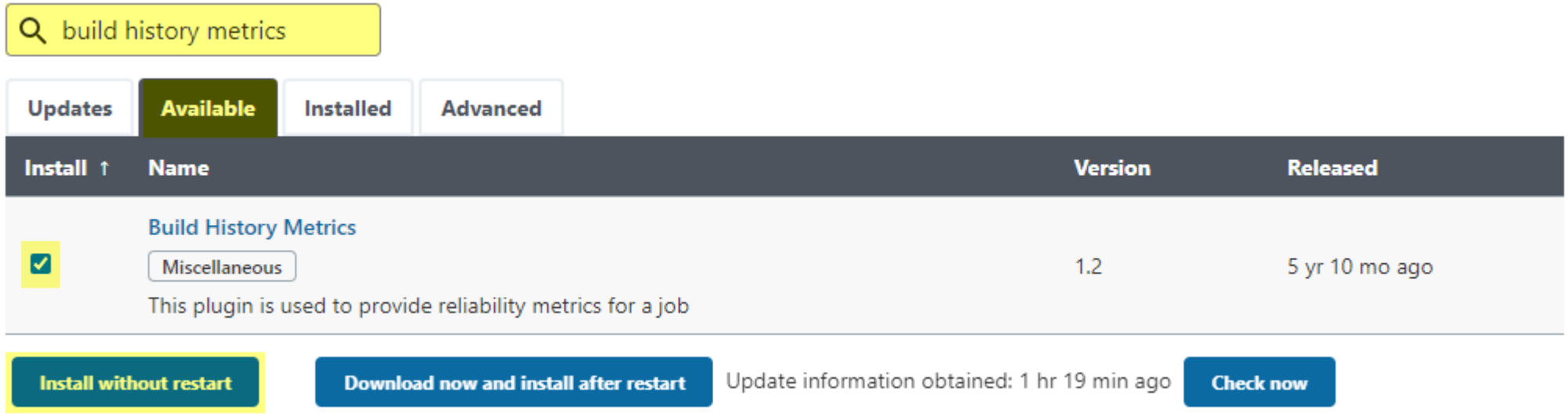
- En "**System Configuration**" da clic en "**Manage Plugins**"



Laboratorios – Comienza la diversión

- Da clic en "**Available**"

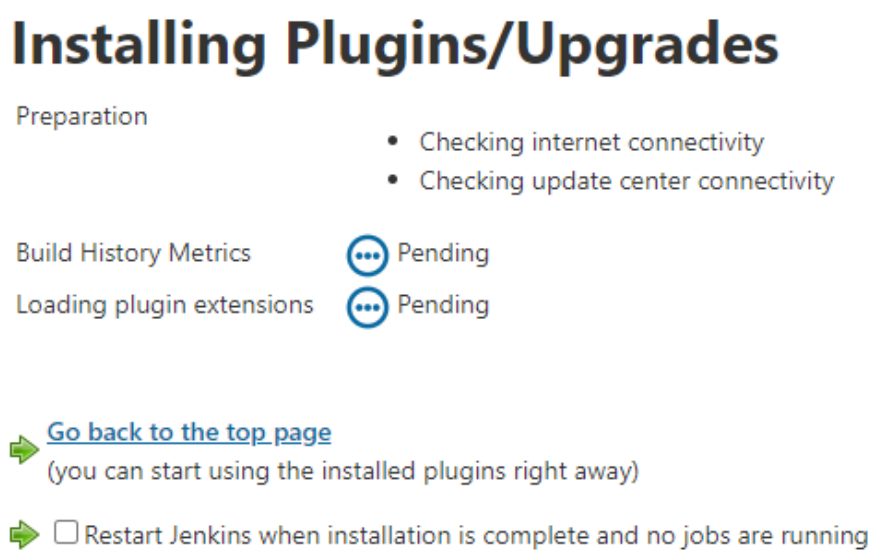
- En "**Search**" escribe "**build history metrics**", "**build with parameters**", "**build pipeline**", "**ansi color**" y "**build monitor view**" selecciónalo y da clic en "**Install without restart**".

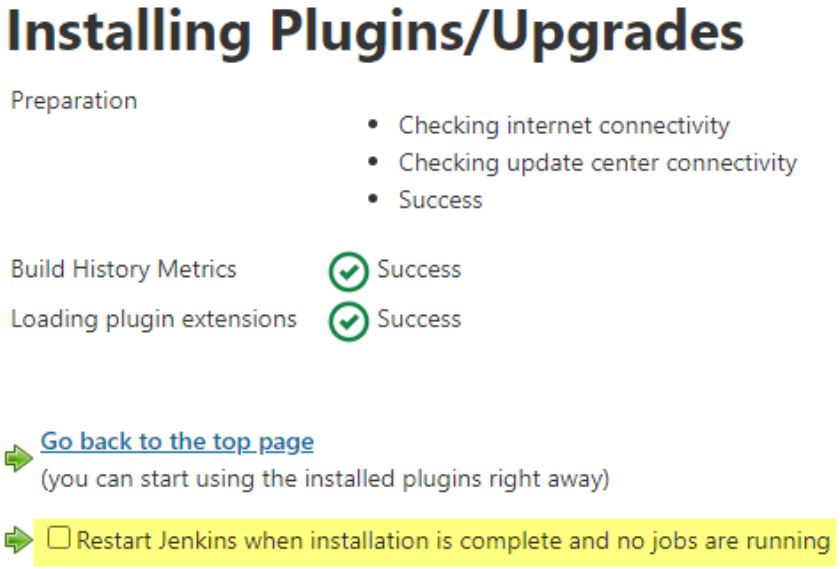


CopaAirlines

Laboratorios – Comienza la diversión

- Después de unos segundos el estado pasará de "**Pending**" a "**Success**" y da clic en "**Restart Jenkins when installation is complete and no Jobs are running**", espera unos segundos mientras carga Jenkins para que te conectes nuevamente.





Laboratorios – Comienza la diversión

- Ve a alguno de los Jobs que se han creado para que veas las estadísticas.

Project job-05

Job para verificar el uso de un webhook


Workspace

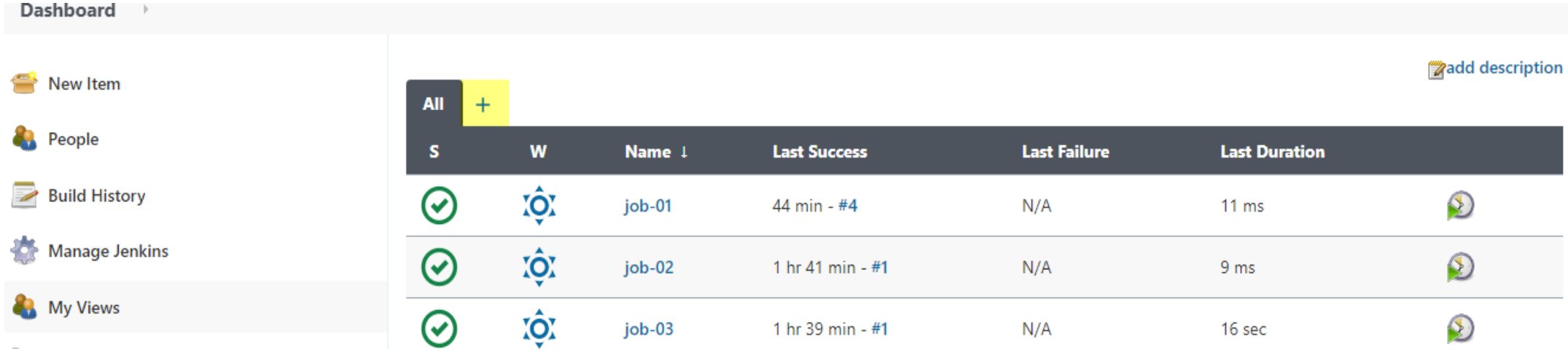

Recent Changes

MTTR	Last 7 Days	0 ms
	Last 30 Days	0 ms
	All Time	0 ms
MTTF	Last 7 Days	0 ms
	Last 30 Days	0 ms
	All Time	0 ms
Standard Deviation	Last 7 Days	5.1 sec
	Last 30 Days	5.1 sec
	All Time	5.1 sec

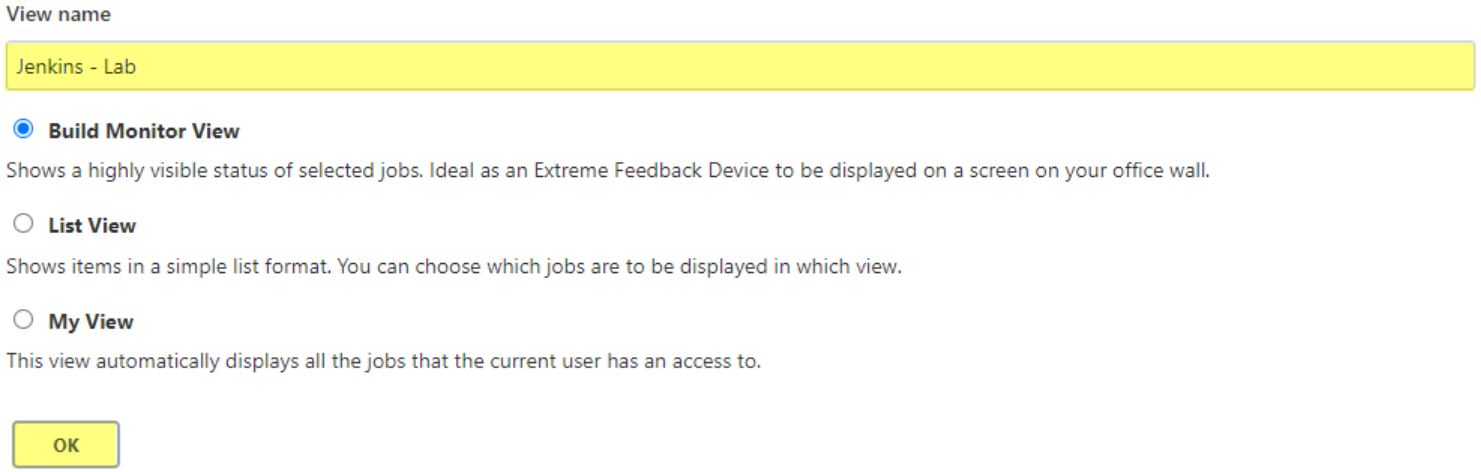
CopaAirlines

Laboratorios – Comienza la diversión

- Da clic en el signo "**+**" para agregar una nueva vista.

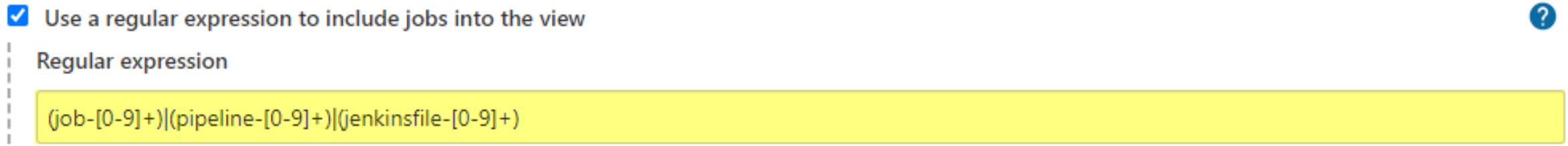


- En "**View name**" agrega "**Jenkins – Lab**" y selecciona "**Build Monitor View**"



Laboratorios – Comienza la diversión

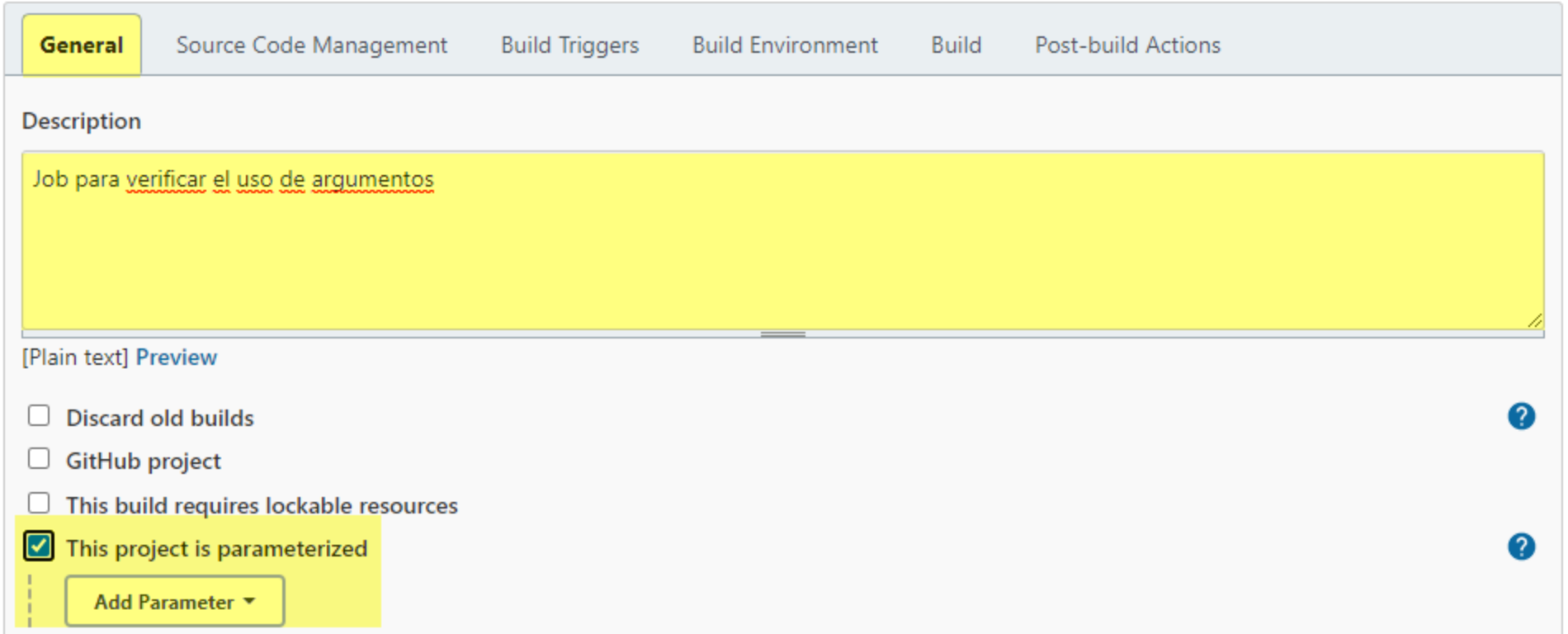
- Da clic en "**Use a regular expresión to include jobs into the view**" y escribe "**(job-[0-9]+)|(pipeline-[0-9]+)|(jenkinsfile-[0-9]+)**"



- Obsérvalos en la vista

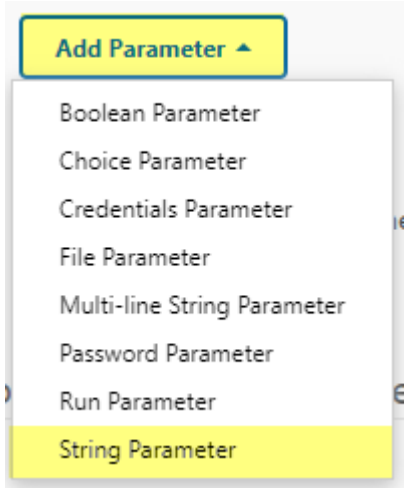
Laboratorios – Comienza la diversión

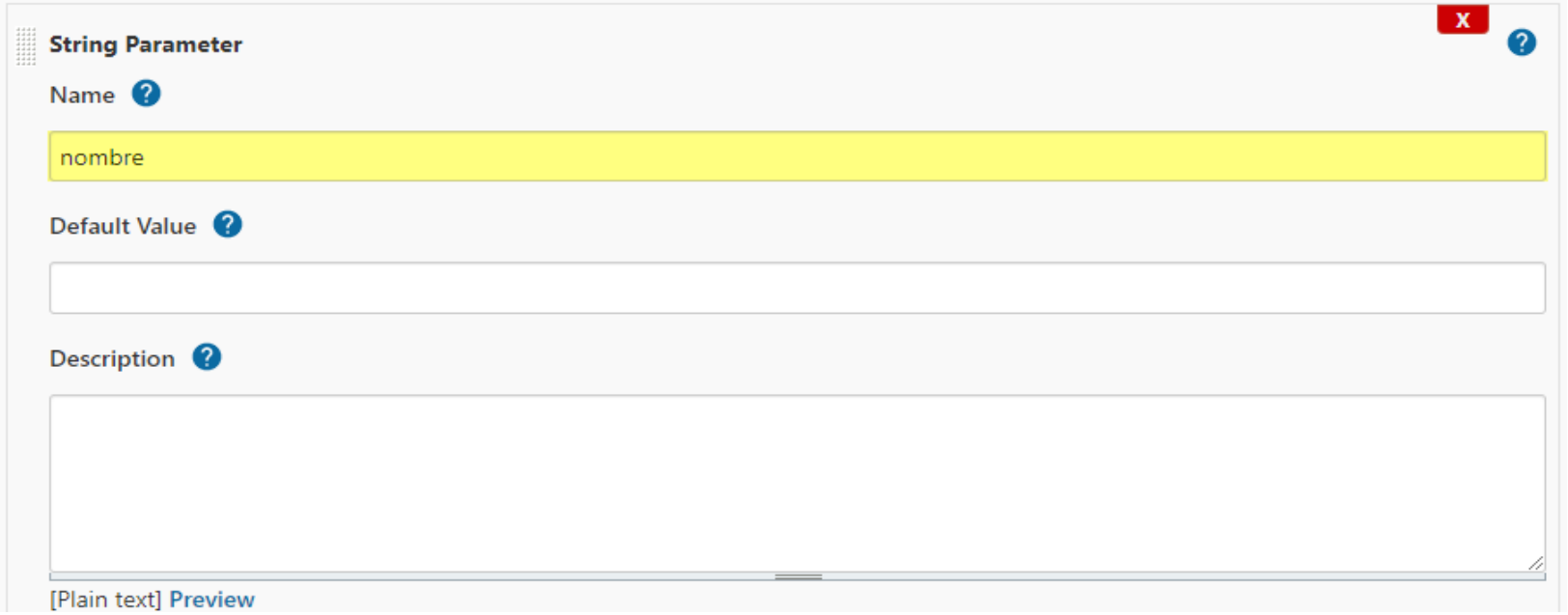
- En **Jenkins** vamos a crear un **séptimo job**
  - Nombre: "**job-07**"
  - Tipo "**Freestyle project**"
  - Descripción: "**Job para verificar el uso de argumentos**"
  - Da clic en "**This Project is parametrized**"



Laboratorios – Comienza la diversión

- En "**Add Parameter**" selecciona "**String Parameter**" en **Name** ponle **nombre**





- En "**Source Code Management**" deja **None**
- En "**Build Environment**" selecciona "**Color ANSI Console Output**", dejando "**xterm**"
- En "**Build**" selecciona "**Execute shell**" escribe:
  - echo "Imprimiendo el valor de la variable \"nombre\":" ${nombre}
- Ejecútalo ("**Build with Parameters**")

CopaAirlines

Laboratorios – Comienza la diversión

- En **Jenkins** vamos a crear un **octavo job**
  - Nombre: "**job-08**"
  - Tipo "**Freestyle project**"
  - Copy from "**job-07**"
  - En "**This project is parameterized**"
    - "String Parameter" -> Name: "palabra"
    - "String Parameter" -> Name: "veces"
  - En "**Source Code Management**" seleccionar **Git** con el siguiente URL:
  - https://github.com/HugoAquinoNavarrete/bash_scripting
  - En "**Branch Specifier**" dejar vacío
  - En "**Build**"
    - bash bin/07-imprime_palabras_n_veces.sh ${palabra} ${veces}
  - Ejecútalo

CopaAirlines

Laboratorios – Comienza la diversión

- En **Jenkins** vamos a crear un **noveno job**
  - Nombre: "**job-09**"
  - Tipo "**Freestyle project**"
  - Copy from "**job-08**"
  - En "**This project is parameterized**"
    - "String Parameter" -> Name: "year"
  - En "**Source Code Management**" deberá aparecer **Git** con el siguiente URL:
  - https://github.com/HugoAquinoNavarrete/bash_scripting
  - En "**Branch Specifier**" dejar vacío
  - En "**Build**"
    - cd bin
    - bash 10-mlb_equipos.sh ${year}
  - Ejecútalo

CopaAirlines

Laboratorios – Comienza la diversión

- En **GitHub** hagan un **fork** del siguiente repositorio
  - https://github.com/HugoAquinoNavarrete/terraform_jenkins_aws
- En **Jenkins** vamos a crear un **décimo job**
  - Nombre: "**job-10**"
  - Tipo "**Freestyle project**"
  - Copy from "**job-09**"
  - En "**This project is parameterized**"
    - "String Parameter" -> Name: "AWS_ACCESS_KEY_ID"
    - "String Parameter" -> Name: "AWS_SECRET_ACCESS_KEY"
    - "String Parameter" -> Name: "nombre_instancia"
    - "String Parameter" -> Name: "cantidad_vm_ubuntu"
    - "String Parameter" -> Name: "cantidad_vm_windows"
    - "String Parameter" -> Name: "subred_id"
    - "String Parameter" -> Name: "sg_id"
    - "String Parameter" -> Name: "nombre_llave"
    - "String Parameter" -> Name: "usuario_github"
    - "Choice Parameter" -> Name: "accion" -> Choices: "crea" "destruye"

Laboratorios – Comienza la diversión

- En **GitHub** hagan un **fork** en su laptop del siguiente repositorio:
  - https://github.com/HugoAquinoNavarrete/terraform_jenkins_aws

- Edita el archivo "**main.tf**" y ajusta las siguientes líneas
  - **Línea 55**: aparece default = "<nombre_llave>", cambia <nombre_llave> con el nombre (sin espacios) que deseas tenga la llave

- Haz un "**commit**" al repositorio
  - git add main.tf
  - git commit -m "Ajustes en main.tf para reflejar cambios en el nombre de la llave privada"
  - git push

CopaAirlines

Laboratorios – Comienza la diversión

- En **Jenkins** vamos a crear un **décimo job**
  - En "**Source Code Management**" deberá aparecer **Git** con el siguiente URL:
    - https://github.com/${usuario_github}/terraform_jenkins_aws
  - En "**Branch Specifier**" dejar vacío, continua en el siguiente slide

CopaAirlines

Laboratorios – Comienza la diversión

- En **Jenkins** vamos a crear un **décimo job**
  - En "**Build**"
    - #!/bin/bash
    - terraform init
    - export AWS_ACCESS_KEY_ID=${AWS_ACCESS_KEY_ID}
    - export AWS_SECRET_ACCESS_KEY=${AWS_SECRET_ACCESS_KEY}
    - if [["${accion}" == "crea"]]; then
    - terraform apply -var nombre_instancia=${nombre_instancia} -var cantidad_instancias_ubuntu=${cantidad_vm_ubuntu} -var cantidad_instancias_windows=${cantidad_vm_windows} -var subred_id=${subred_id} -var sg_id=${sg_id} -var key_name=${nombre_llave} -auto-approve
    - else
    - terraform destroy -var subred_id=${subred_id} -var sg_id=${sg_id} -auto-approve
    - fi

CopaAirlines

Laboratorios – Comienza la diversión

- En **Jenkins** vamos a crear un **décimo job**
  - Ejecútalo con la opción "**crea**" habilitando **1 servidor Ubuntu** y **1 servidor Windows**. El resto de la información aparece en el **archivo de Excel**.

CopaAirlines

Laboratorios – Comienza la diversión

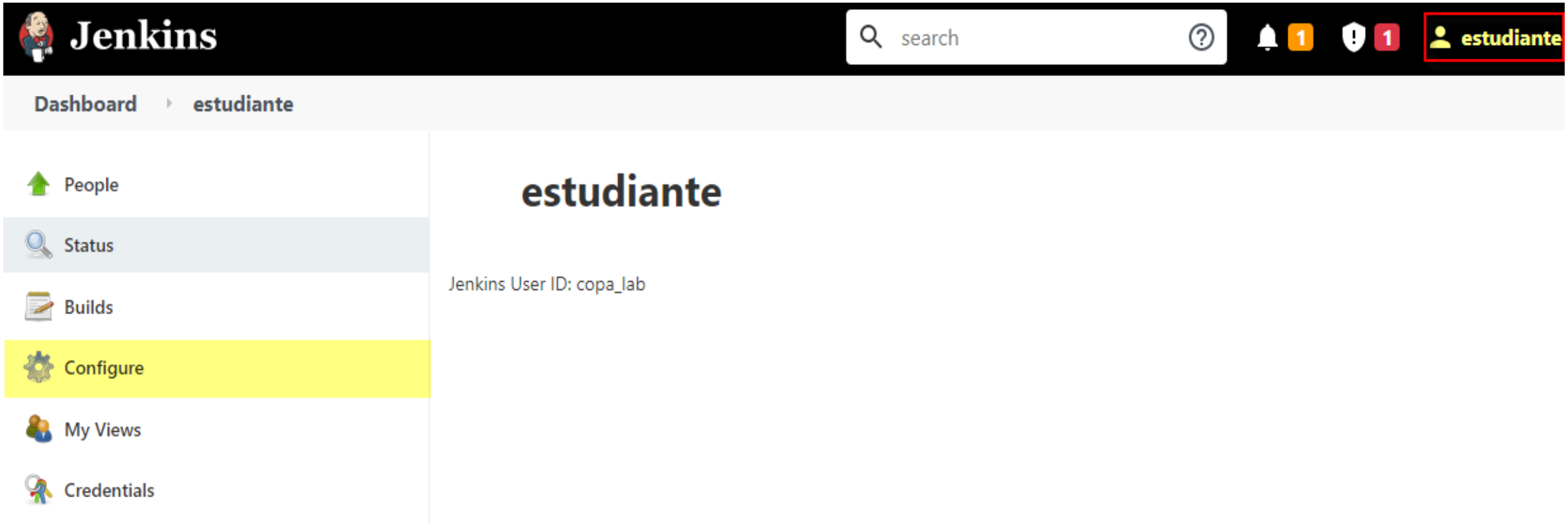
- En **Jenkins** vamos a crear un **décimo job**
  - Ahora vuelve a ejecutar el **job-10** solamente facilitando la información de "**AWS_ACCESS_KEY_ID**", "**AWS_SECRET_ACCESS_KEY**", "**subred_id**", "**sd_id**" y "**usuario_github**" seleccionando la opción "**destruye**"

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a aprender a usar otra manera de ejecutar un job desde tu laptop usando **curl** y un **token**.

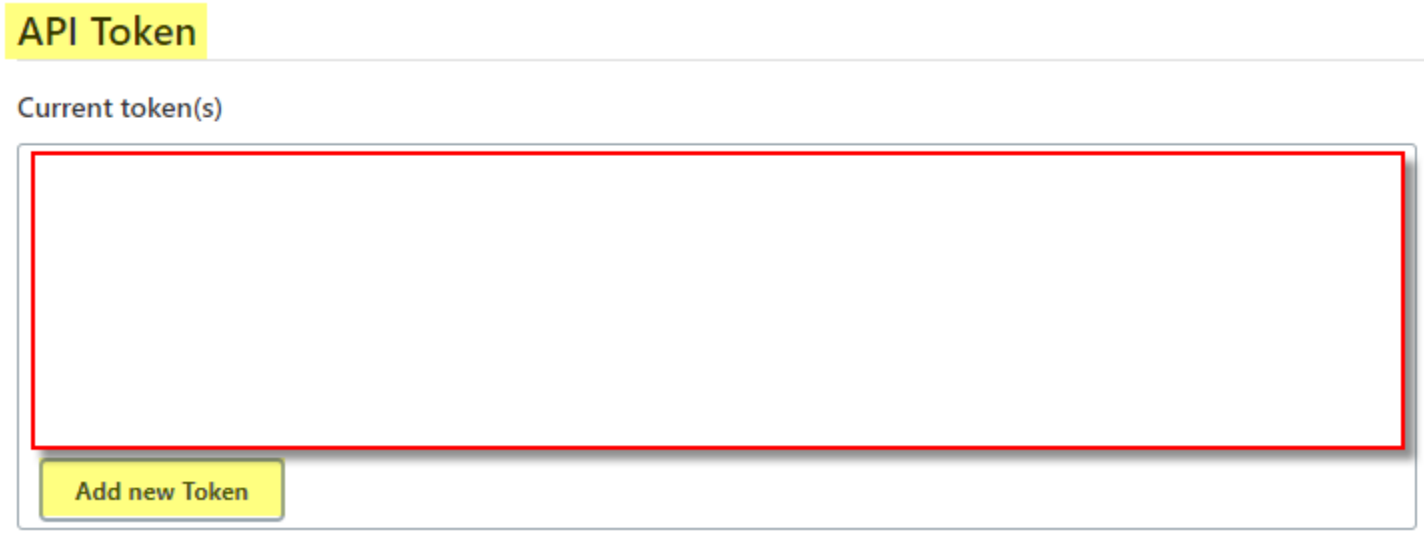
- Da clic en "**estudiante**" y posteriormente en "**Configure**"



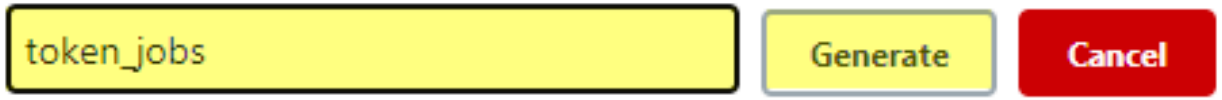
CopaAirlines

Laboratorios – Comienza la diversión

- En "**API Token**" da clic en "**Add new Token**"



- Ponle como nombre "**token_jobs**" y da clic en "**Generate**"



- Copia el valor en un lugar seguro y has clic en "**Save**"

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a usarlo ejecutando el "**job-01**". Desde tu laptop ejecuta esto:
  - curl -X POST http://<ip_publica_vm>:8080/job/job-01/build --user copa_lab:<token_recien_creado>

- Para el "**job-02**". Desde tu laptop ejecuta esto:
  - curl -X POST http://<ip_publica_vm>:8080/job/job-02/build --user copa_lab:<token_recien_creado>

- Para el "**job-03**". Desde tu laptop ejecuta esto:
  - curl -X POST http://<ip_publica_vm>:8080/job/job-03/build --user copa_lab:<token_recien_creado>

- Para el "**job-04**". Desde tu laptop ejecuta esto:
  - curl -X POST http://<ip_publica_vm>:8080/job/job-04/build --user copa_lab:<token_recien_creado>

CopaAirlines

Laboratorios – Comienza la diversión

- Para el "**job-05**". Desde tu laptop ejecuta esto:
  - curl -X POST http://<ip_publica_vm>:8080/job/job-05/build --user copa_lab:<token_recien_creado>

- Para el "**job-06**". Desde tu laptop ejecuta esto:
  - curl -X POST http://<ip_publica_vm>:8080/job/job-06/build --user copa_lab:<token_recien_creado>

- Para el "**job-07**". Desde tu laptop ejecuta esto:
  - curl -X POST http://<ip_publica_vm>:8080/job/job-07/build --user copa_lab:<token_recien_creado> --data-urlencode json='{"parameter": [{"name":"nombre", "value":"mensaje_prueba"}]}'

CopaAirlines

Laboratorios – Comienza la diversión

- Para el "**job-08**". Desde tu laptop ejecuta esto:
  - curl -X POST http://<ip_publica_vm>:8080/job/job-08/build --user copa_lab:<token_recien_creado> --data-urlencode json='{"parameter": [{"name":"palabra", "value":"mensaje_prueba"},{"name":"veces", "value":"10"}]}'

- Para el "**job-09**". Desde tu laptop ejecuta esto:
  - curl -X POST http://<ip_publica_vm>:8080/job/job-09/build --user copa_lab:<token_recien_creado> --data-urlencode json='{"parameter": [{"name":"year", "value":"1921"}]}'

CopaAirlines

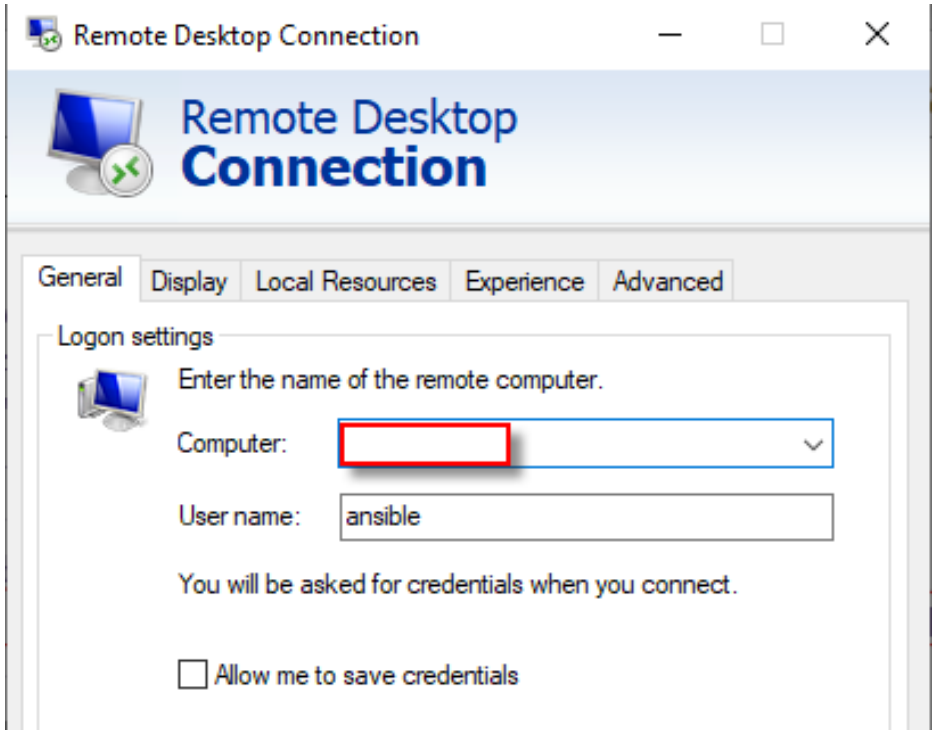
Laboratorios – Comienza la diversión

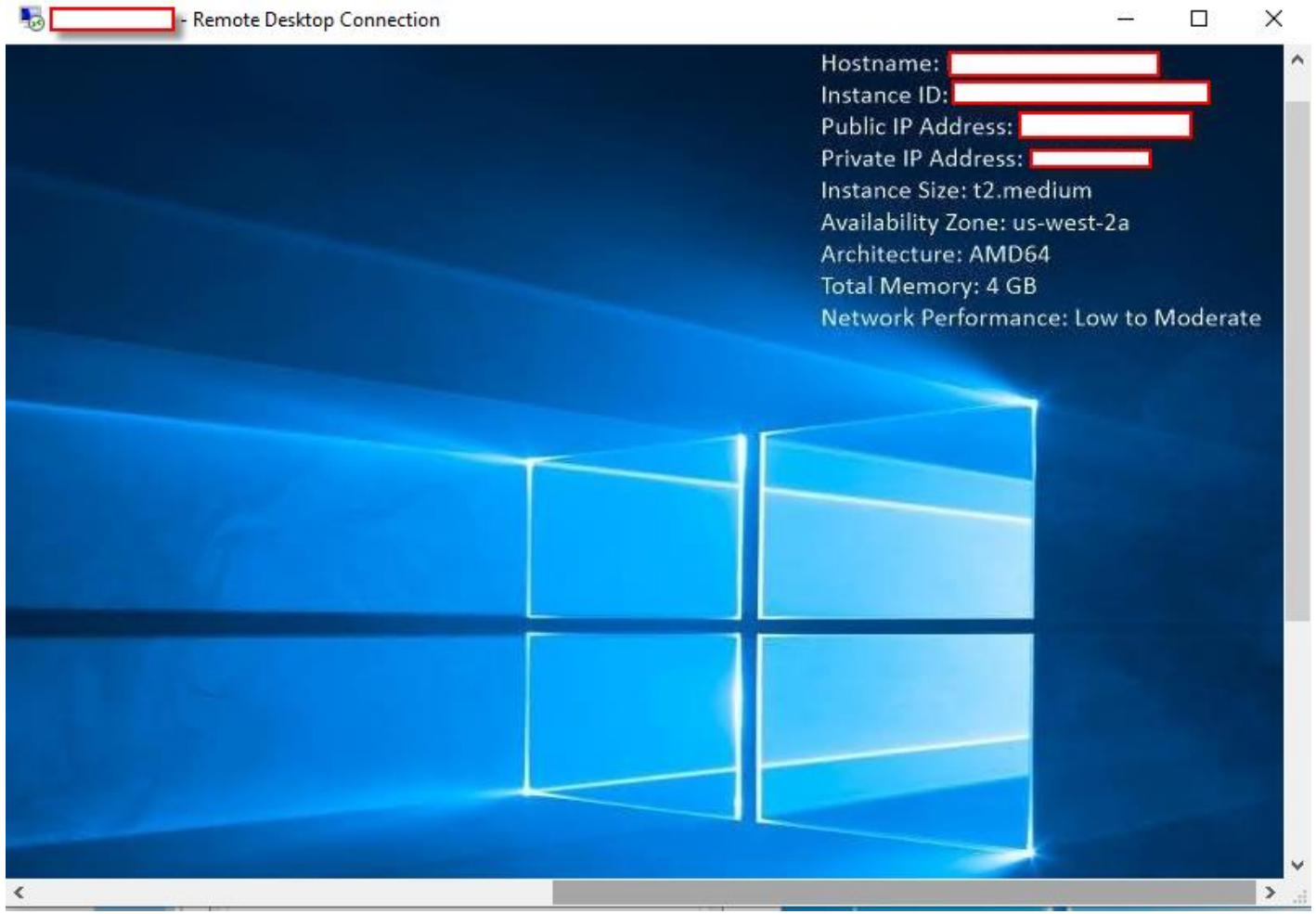
- Para el "**job-10**". Desde tu laptop ejecuta esto:
  - Para **crear** las VMs
  - curl -X POST http://<ip_publica_vm>:8080/job/job-10/build --user copa_lab:<valor_dado_en_clase> --data-urlencode json='{"parameter": [{"name":"AWS_ACCESS_KEY_ID", "value":"<valor_dado_en_clase>"},{"name":"AWS_SECRET_ACCESS_KEY", "value":"<valor_dado_en_clase>"},{"name":"nombre_instancia", "value":"lab"},{"name":"cantidad_vm_ubuntu", "value":"2"},{"name":"cantidad_vm_windows", "value":"2"},{"name":"subred_id", "value":"<valor_dado_en_clase>"},{"name":"sg_id", "value":"<valor_dado_en_clase>"}, {"name":"nombre_llave", "value":"<nombre_llave>"},{"name":"usuario_github", "value":"<usuario_github>"},{"name":"accion", "value":"crea"}]}'

CopaAirlines

Laboratorios – Comienza la diversión

- Para el "**job-10**". Desde tu laptop ejecuta esto:
  - Intenta **conectarte** a una de las **VM Windows** usando **RDP** (**IP pública** que aparece en la ejecución del job, usuario "**ansible**" y contraseña el campo "**password_data**")





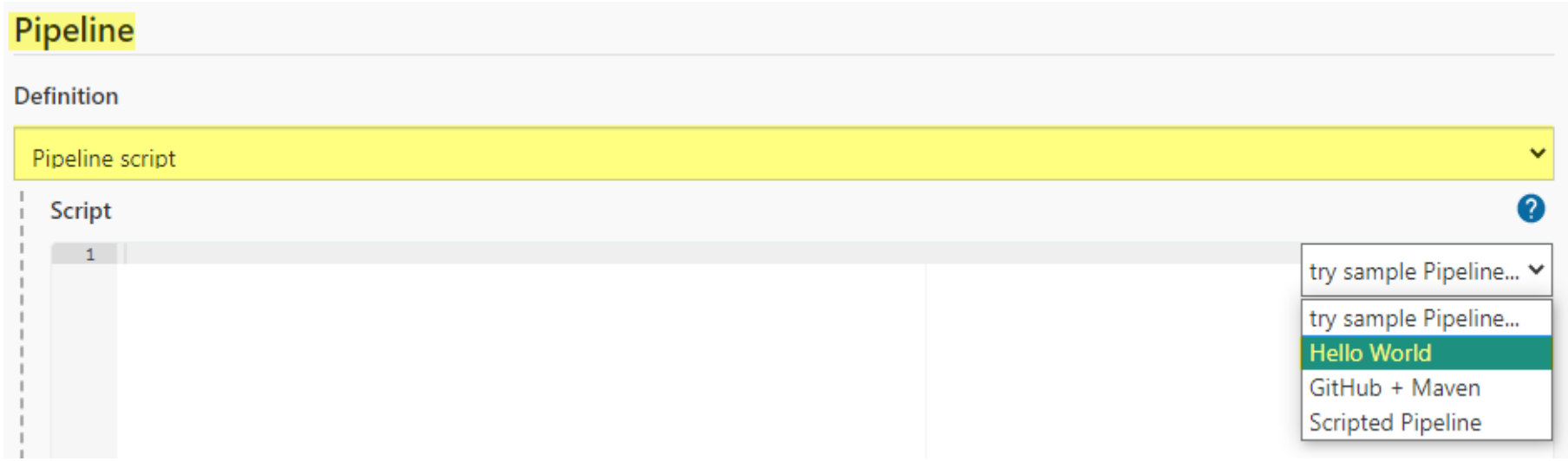
Laboratorios – Comienza la diversión

- Para el "**job-10**". Desde tu laptop ejecuta esto:
  - Para **destruir** las VMs
  - curl -X POST http://<ip_publica_vm>:8080/job/job-10/build --user copa_lab:<valor_dado_en_clase> --data-urlencode json='{"parameter": [{"name":"AWS_ACCESS_KEY_ID", "value":"<valor_dado_en_clase>"},{"name":"AWS_SECRET_ACCESS_KEY", "value":"<valor_dado_en_clase>"}, {"name":"subred_id", "value":"<valor_dado_en_clase>"},{"name":"sg_id", "value":"<valor_dado_en_clase>"},{"name":"accion", "value":"destruye"}]}'

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **primer pipeline**.
  - Nombre: "**pipeline-01**"
  - Tipo "**Pipeline**"
  - En "**Description**" pon "**Pipeline como código**"
  - En "**Pipeline**" escoge "**Pipeline script**" da clic en "**try simple Pipeline**" y selecciona "**Hello World**"



  - Haz clic en "**Save**" y ejecútalo desde tu laptop de la siguiente manera:
    - curl -X POST http://<ip_publica_vm>:8080/job/pipeline-01/build --user copa_lab:<token_recien_creado>

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **segundo pipeline**.
  - Nombre: "**pipeline-02**"
  - Copy from "**pipeline-01**"
  - Tipo "**Pipeline**"
  - En "**Description**" pon "**Pipeline como código**"
  - En "**Pipeline**" hay que editar las líneas de código que aparecen con las que aparecen en el siguiente slide
  - Haz clic en **Save** y ejecútalo desde tu laptop de la siguiente manera:
    - curl -X POST http://<ip_publica_vm>:8080/job/pipeline-02/build --user copa_lab:<token_recien_creado>

CopaAirlines

Laboratorios – Comienza la diversión

```
pipeline {
    agent any
    stages {
        stage('# BUILD_ID') {
            steps {
                sh 'echo $BUILD_ID'
            }
        }
        stage('WORKSPACE') {
            steps {
                sh 'echo $WORKSPACE'
            }
        }
        stage('Contenido inicial del WORKSPACE') {
            steps {
                sh 'ls -lR $WORKSPACE'
            }
        }
        stage('Crea archivo file.txt') {
            steps {
                sh 'echo "Que ondas aprendiendo Jenkins !!!" > $WORKSPACE/file.txt'
            }
        }
        stage('Contenido final del WORKSPACE') {
            steps {
                sh 'ls -lR $WORKSPACE'
            }
        }
        stage('Contenido archivo file.txt') {
            steps {
                sh 'cat $WORKSPACE/file.txt'
            }
        }
    }
}
```

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **tercer pipeline**.
  - Nombre: "**pipeline-03**"
  - Copy from "**pipeline-02**"
  - Tipo "**Pipeline**"
  - En "**Description**" pon "**Pipeline como código**"
  - En "**Pipeline**" hay que editar las líneas de código que aparecen con las que aparecen en el siguiente slide
  - Haz clic en **Save** y ejecútalo desde tu laptop de la siguiente manera:
    - curl -X POST http://<ip_publica_vm>:8080/job/pipeline-03/build --user copa_lab:<token_recien_creado>

CopaAirlines

Laboratorios – Comienza la diversión

```
pipeline {
    agent any

    stages {
        stage('Clona repositorio') {
            steps {
                git branch: 'main', url: 'https://github.com/HugoAquinoNavarrete/bash_scripting'
            }
        }

        stage('Imprime directorios') {
            steps {
                sh 'ls -lR'
            }
        }

        stage('Imprime variables') {
            steps {
                sh '/bin/bash ./bin/01-imprime_variables.sh'
            }
        }

        stage('Imprime variables de ambiente') {
            steps {
                sh '/bin/bash ./bin/06-mi_entorno.sh'
            }
        }

    }
}
```

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **cuarto pipeline**.
  - Nombre: "**pipeline-04**"
  - Copy from "**pipeline-03**"
  - Tipo "**Pipeline**"
  - En "**Description**" pon "**Pipeline como código**"
  - En "**This project is parameterized**"
    - "String Parameter" -> Name: "palabra"
    - "String Parameter" -> Name: "veces"
  - En "**Pipeline**" hay que editar las líneas de código que aparecen con las que aparecen en el siguiente slide
  - Haz clic en **Save** y ejecútalo desde tu laptop de la siguiente manera:
    - curl -X POST http://<ip_publica_vm>:8080/job/pipeline-04/build --user copa_lab:<token_recien_creado> --data-urlencode json='{"parameter": [{"name":"palabra", "value":"ejecutando_pipeline_desde_curl"},{"name":"veces", "value":"25"}]}'

CopaAirlines

Laboratorios – Comienza la diversión

```
pipeline {
    agent any

    stages {
        stage('Clona repositorio') {
            steps {
                git branch: 'main', url: 'https://github.com/HugoAquinoNavarrete/bash_scripting'
            }
        }

        stage('Imprime un mensaje n veces') {
            steps {
                sh '/bin/bash ./bin/07-imprime_palabras_n_veces.sh ${palabra} ${veces}'
            }
        }

    }
}
```

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **quinto pipeline**.
  - Nombre: "**pipeline-05**"
  - Copy from "**pipeline-04**"
  - Tipo "**Pipeline**"
  - En "**Description**" pon "**Pipeline como código**"
  - En "**This project is parameterized**"
    - "String Parameter" -> Name: "AWS_ACCESS_KEY_ID"
    - "String Parameter" -> Name: "AWS_SECRET_ACCESS_KEY"
    - "String Parameter" -> Name: "nombre_instancia"
    - "String Parameter" -> Name: "cantidad_vm_ubuntu"
    - "String Parameter" -> Name: "cantidad_vm_windows"
    - "String Parameter" -> Name: "subred_id"
    - "String Parameter" -> Name: "sg_id"
    - "String Parameter" -> Name: "nombre_llave"
    - "String Parameter" -> Name: "usuario_github"
    - "Choice Parameter" -> Name: "accion" -> Choices: "crea" "destruye"
  - En "**Pipeline**" hay que editar las líneas de código que aparecen con las que aparecen en el siguiente slide, después haz clic en **Save**

CopaAirlines

Laboratorios – Comienza la diversión

```
pipeline {
    agent any
    options {
      ansiColor('xterm')
    }
    stages {
        stage('Clona repositorio') {
            steps {
                git branch: 'main', url: 'https://github.com/${usuario_github}/terraform_jenkins_aws'
            }
        }
        stage('Exporta variables de entorno') {
            steps {
                sh 'export AWS_ACCESS_KEY_ID=${AWS_ACCESS_KEY_ID}'
                sh 'export AWS_SECRET_ACCESS_KEY=${AWS_SECRET_ACCESS_KEY}'
            }
        }
        stage('Inicializa terraform') {
            steps {
                sh 'terraform init'
            }
        }
        stage('Crea infraestructura') {
            when {
                equals expected: "crea", actual: "${accion}"
            }
            steps {
                sh 'terraform apply -var nombre_instancia=${nombre_instancia} -var cantidad_instancias_ubuntu=${cantidad_vm_ubuntu} -var cantidad_instancias_windows=${cantidad_vm_windows} -var subred_id=${subred_id} -var sg_id=${sg_id} -var key_name=${nombre_llave} -var usuario_github=${usuario_github} -auto-approve'
                sh 'echo "Inventario VM Ubuntu"'
                sh 'cat ansible_inventario.txt'
                sh 'echo "Inventario VM Windows"'
                sh 'cat ansible_inventario_win.txt'
            }
        }
        stage('Destruye infraestructura') {
            when {
                equals expected: "destruye", actual: "${accion}"
            }
            steps {
                sh 'terraform destroy -var subred_id=${subred_id} -var sg_id=${sg_id} -auto-approve'
            }
        }
    }
}
```

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **quinto pipeline**.
  - Crea **1 VM Ubuntu** y **1VMs Windows**
  - Para **crear** las VMs
  - curl -X POST http://<ip_publica_vm>:8080/job/pipeline-05/build --user copa_lab:<valor_dado_en_clase> --data-urlencode json='{"parameter": [{"name":"AWS_ACCESS_KEY_ID", "value":“<valor_dado_en_clase>"},{"name":"AWS_SECRET_ACCESS_KEY", "value":"<valor_dado_en_clase>"},{"name":"nombre_instancia", "value":"lab"},{"name":"cantidad_vm_ubuntu", "value":"2"},{"name":"cantidad_vm_windows", "value":"2"},{"name":"subred_id", "value":"<valor_dado_en_clase>"},{"name":"sg_id", "value":"<valor_dado_en_clase>"}, {"name":"nombre_llave", "value":“<nombre_llave>"}, {"name":"usuario_github", "value":“<usuario_github>"},{"name":"accion", "value":“crea"}]}’

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **quinto pipeline**.
  - Para **destruir** las VMs
  - curl -X POST http://<ip_publica_vm>:8080/job/pipeline-05/build --user copa_lab:<valor_dado_en_clase> --data-urlencode json='{"parameter": [{"name":"AWS_ACCESS_KEY_ID", "value":“<valor_dado_en_clase>"},{"name":"AWS_SECRET_ACCESS_KEY", "value":"<valor_dado_en_clase>"}, {"name":"usuario_github", "value":“<usuario_github>"}, {"name":"accion", "value":“destruye"}]}’

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **sexto pipeline**.
  - Nombre: "**pipeline-06**"
  - Copy from "**pipeline-05**"
  - Tipo "**Pipeline**"
  - En "**Description**" pon "**Pipeline como código**"
  - En "**This project is parameterized**"
    - "String Parameter" -> Name: "nombre_llave"
    - "Choice Parameter" -> Name: "accion" -> Choices: "lista_archivos" "haz_ping" "imprime_mensaje_ubuntu" "imprime_mensaje_windows"
  - En "**Pipeline**" hay que editar las líneas de código que aparecen con las que aparecen en el siguiente slide, después haz clic en **Save**

CopaAirlines

Laboratorios – Comienza la diversión

```
pipeline {
    agent any
    options {
      ansiColor('xterm')
    }
    stages {
        stage('Clona repositorio') {
            steps {
                git branch: 'main', url: 'https://github.com/HugoAquinoNavarrete/ansible_scripting'
            }
        }
        stage('Copia llave e inventarios del pipeline-05') {
            steps {
                sh 'cp ../pipeline-05/${nombre_llave} .'
                sh 'cp ../pipeline-05/ansible_inventario.txt .'
                sh 'cp ../pipeline-05/ansible_inventario_win.txt .'
            }
        }
        stage('Lista archivos del repositorio clonado') {
            when {
                equals expected: "lista_archivos", actual: "${accion}"
            }
            steps {
                sh 'ls -l'
                sh 'cat ansible_inventario.txt'
                sh 'cat ansible_inventario_win.txt'
            }
        }
        stage('Haz ping con Ansible a las VMs creadas') {
            when {
                equals expected: "haz_ping", actual: "${accion}"
            }
            steps {
                sh '/bin/bash ./bin/01-ping.sh ${nombre_llave}'
                sh '/bin/bash ./bin/01-ping_win.sh ${nombre_llave}'
            }
        }
        stage('Imprime mensaje con Ansible a las VMs Ubuntu creadas') {
            when {
                equals expected: "imprime_mensaje_ubuntu", actual: "${accion}"
            }

            steps {
                sh '/bin/bash ./bin/02-mensaje.sh ${nombre_llave}'
            }
        }
        stage('Imprime mensaje con Ansible a las VMs Windows creadas') {
            when {
                equals expected: "imprime_mensaje_windows", actual: "${accion}"
            }
            steps {
                sh '/bin/bash ./bin/02-mensaje_win.sh ${nombre_llave}'
            }
        }
    }
}
```

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **sexto pipeline**.
  - Ejecútalo una vez seleccionando la **acción** "**lista_archivos**"
  - curl -X POST http://<ip_publica_vm>:8080/job/pipeline-06/build --user copa_lab:<token_recien_creado> --data-urlencode json='{"parameter": [{"name":"accion", "value":"lista_archivos"}, {"name":"nombre_llave", "value":"<nombre_llave>"}]}'
  - Para poder ejecutar las acciones "**haz_ping**" e "**imprime_mensajes**" hay que pasar la llave "**key**" y el inventario "**ansible_inventario.txt**" creados en el "**pipeline-05**" al workspace del "**pipeline-06**", para hacerlo vamos a conectarnos desde tu laptop usando la llave que se facilitó al inicio del curso a la VM en la cual está corriendo **Jenkins**:
  - ssh -v -l ubuntu -i <llave> <ip_pública_vm>

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **sexto pipeline**.
  - Para **VM Ubuntu**, ejecuta el script "**./bin/01-ping.sh**" para habilitar el "**fingerprint**",
  - ./bin/01-ping.sh <nombre_llave>
  - A la pregunta "**Are you sure you want to continue connecting (yes/no)?**", contesta "**yes**", cancela la ejecución del script "**Control-C**" y vuélvelo a ejecutar tantas veces como VMs hayas creado o teclea "**Enter**" para ir avanzando en la ejecución e ir contestando "**yes**"
  - Para **VM Windows**, ejecuta el script "**./bin/01-ping_win.sh**" para habilitar el "**fingerprint**"
  - ./bin/01-ping_win.sh <nombre_llave>
  - Instala el siguiente componente de **ansible-galaxy** para los siguientes laboratorios asociados a windows
  - ansible-galaxy collection install ansible.windows

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **sexto pipeline**.
  - La ejecución final debería verse así:

```
jenkins@[redacted]:~/workspace/pipeline-06$ ./bin/01-ping.sh
Using /etc/ansible/ansible.cfg as config file
/var/jenkins_home/workspace/pipeline-06/ansible_inventario.txt did not meet host_list requirements, check plugin documentation if this is unexpected
/var/jenkins_home/workspace/pipeline-06/ansible_inventario.txt did not meet script requirements, check plugin documentation if this is unexpected
/var/jenkins_home/workspace/pipeline-06/ansible_inventario.txt did not meet yaml requirements, check plugin documentation if this is unexpected

PLAY [servers] *****************************************************************************************************************************************

TASK [ping] ********************************************************************************************************************************************
ok: [[redacted]] => {"changed": false, "ping": "pong"}
ok: [[redacted]] => {"changed": false, "ping": "pong"}

PLAY RECAP *********************************************************************************************************************************************
[redacted]                 : ok=1    changed=0    unreachable=0    failed=0
                           : ok=1    changed=0    unreachable=0    failed=0

jenkins@d7201c6c9a88:~/workspace/pipeline-06$ ./bin/01-ping_win.sh
Using /etc/ansible/ansible.cfg as config file
/var/jenkins_home/workspace/pipeline-06/ansible_inventario_win.txt did not meet host_list requirements, check plugin documentation if this is unexpected
/var/jenkins_home/workspace/pipeline-06/ansible_inventario_win.txt did not meet script requirements, check plugin documentation if this is unexpected
/var/jenkins_home/workspace/pipeline-06/ansible_inventario_win.txt did not meet yaml requirements, check plugin documentation if this is unexpected

PLAY [servers_win] *************************************************************************************************************************************

TASK [win_ping] ****************************************************************************************************************************************
ok: [[redacted]] => {"changed": false, "ping": "pong"}
ok: [[redacted]] => {"changed": false, "ping": "pong"}

PLAY RECAP *********************************************************************************************************************************************
[redacted]                 : ok=1    changed=0    unreachable=0    failed=0
                           : ok=1    changed=0    unreachable=0    failed=0
```

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **sexto pipeline**.
  - Ejecútalo una vez seleccionando la **acción** "**haz_ping**"
  - curl -X POST http://<ip_pública_vm>:8080/job/pipeline-06/build --user copa_lab:<token_recien_creado> --data-urlencode json='{"parameter": [{"name":"accion", "value":"haz_ping"}, {"name":"nombre_llave", "value":"<nombre_llave>"}]}'
  - Ejecútalo una vez seleccionando la **acción** "**imprime_mensaje_ubuntu**"
  - curl -X POST http://<ip_pública_vm>:8080/job/pipeline-06/build --user copa_lab:<token_recien_creado> --data-urlencode json='{"parameter": [{"name":"accion", "value":"imprime_mensaje_ubuntu"}, {"name":"nombre_llave", "value":"<nombre_llave>"}]}'
  - Ejecútalo una vez seleccionando la **acción** "**imprime_mensaje_windows**"
  - curl -X POST http://<ip_pública_vm>:8080/job/pipeline-06/build --user copa_lab:<token_recien_creado> --data-urlencode json='{"parameter": [{"name":"accion", "value":"imprime_mensaje_windows"}, {"name":"nombre_llave", "value":"<nombre_llave>"}]}'

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **séptimo pipeline**.
  - Nombre: "**pipeline-07**"
  - Copy from "**pipeline-06**"
  - Tipo "**Pipeline**"
  - En "**Description**" pon "**Pipeline como código**"
  - En "**This project is parameterized**"
    - "String Parameter" -> Name: "nombre_llave"
    - "Choice Parameter" -> Name: "accion" -> Choices: "crea_directorios" "borra_directorios" "clona_repositorio" "copia_archivos" "borra_archivos"
  - En "**Pipeline**" hay que editar las líneas de código que aparecen con las que aparecen en el siguiente slide, después haz clic en **Save**

CopaAirlines

Laboratorios – Comienza la diversión

```
pipeline {
    agent any
    options {
      ansiColor('xterm')
    }
    stages {
        stage('Clona repositorio') {
            steps {
                git branch: 'main', url: 'https://github.com/HugoAquinoNavarrete/ansible_scripting'
            }
        }
        stage('Copia llave e inventario del pipeline-05') {
            steps {
                sh 'cp ../pipeline-05/${nombre_llave} .'
                sh 'cp ../pipeline-05/ansible_inventario.txt .'
                sh 'cp ../pipeline-05/ansible_inventario_win.txt .'
            }
        }
        stage('Crea directorios') {
            when {
                equals expected: "crea_directorios", actual: "${accion}"
            }
            steps {
                sh '/bin/bash ./bin/08-directorios_crea.sh ${nombre_llave}'
                sh '/bin/bash ./bin/08-directorios_crea_win.sh ${nombre_llave}'
            }
        }
        stage('Borra directorios') {
            when {
                equals expected: "borra_directorios", actual: "${accion}"
            }
            steps {
                sh '/bin/bash ./bin/09-directorios_borra.sh ${nombre_llave}'
                sh '/bin/bash ./bin/09-directorios_borra_win.sh ${nombre_llave}'
            }
        }
        stage('Clona repositorio git') {
            when {
                equals expected: "clona_repositorio", actual: "${accion}"
            }
            steps {
                sh '/bin/bash ./bin/10-git_clona.sh ${nombre_llave}'
            }
        }
        stage('Copia archivos') {
            when {
                equals expected: "copia_archivos", actual: "${accion}"
            }
            steps {
                sh '/bin/bash ./bin/13-archivos_copia.sh ${nombre_llave}'
                sh '/bin/bash ./bin/13-archivos_copia_win.sh ${nombre_llave}'
            }
        }
        stage('Borra archivos') {
            when {
                equals expected: "borra_archivos", actual: "${accion}"
            }
            steps {
                sh '/bin/bash ./bin/22-archivos_borra.sh ${nombre_llave}'
                sh '/bin/bash ./bin/22-archivos_borra_win.sh ${nombre_llave}'
            }
        }
    }
}
```

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **séptimo pipeline**.
  - Ejecútalo una vez seleccionando la **acción** "**crea_directorios**"
  - curl -X POST http://<ip_publica_vm>:8080/job/pipeline-07/build --user copa_lab:<token_recien_creado> --data-urlencode json='{"parameter": [{"name":"nombre_llave", "value":"<nombre_llave>"}, {"name":"accion", "value":"crea_directorios"}]}'
  - Ejecútalo una vez seleccionando la **acción** "**borra_directorios**"
  - curl -X POST http://<ip_publica_vm>:8080/job/pipeline-07/build --user copa_lab:<token_recien_creado> --data-urlencode json='{"parameter": [{"name":"nombre_llave", "value":"<nombre_llave>"}, {"name":"accion", "value":"borra_directorios"}]}'
  - Ejecútalo una vez seleccionando la **acción** "**clona_repositorio**"
  - curl -X POST http://<ip_publica_vm>:8080/job/pipeline-07/build --user copa_lab:<token_recien_creado> --data-urlencode json='{"parameter": [{"name":"nombre_llave", "value":"<nombre_llave>"}, {"name":"accion", "value":"clona_repositorio"}]}'

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **séptimo pipeline**.
  - Ejecútalo una vez seleccionando la **acción** "**copia_archivos**"
  - curl -X POST http://<ip_publica_vm>:8080/job/pipeline-07/build --user copa_lab:<token_recien_creado> --data-urlencode json='{"parameter": [{"name":"nombre_llave", "value":"<nombre_llave>"}, {"name":"accion", "value":"copia_archivos"}]}'

  - Ejecútalo una vez seleccionando la **acción** "**borra_archivos**"
  - curl -X POST http://<ip_publica_vm>:8080/job/pipeline-07/build --user copa_lab:<token_recien_creado> --data-urlencode json='{"parameter": [{"name":"nombre_llave", "value":"<nombre_llave>"}, {"name":"accion", "value":"borra_archivos"}]}'

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **primer jenkinsfile**.
  - En tu cuenta de GitHub haz un **fork** del siguiente repositorio:
    - https://github.com/HugoAquinoNavarrete/jenkins_scripting
  - Habilita un **webhook** para el repositorio al cual le hiciste **fork** (slides 36 y 37)
  - Nombre: "**jenkinsfile-01**"
  - Tipo "**Pipeline**"
  - En "**Description**" pon "**Jenkinsfile como código**"
  - En "**Build Triggers**" selecciona "**GitHub hook trigger for GITScm polling**"

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **primer jenkinsfile**.
  - En "**Pipeline**" selecciona "**Pipeline script from SCM**", en "**SCM**" escoge "**Git**" y en "**Repository URL**" pon el URL del sitio al cual le hiciste fork:

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **primer jenkinsfile**.
  - En "**Branches to build**" escribe "***/main**"

  - En "**Script Path**" escribe "**jenkinsfiles/Jenkinsfile-01.txt**" y después da clic en "**Save**"

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **primer jenkinsfile**.
  - Ejecútalo una vez
  - curl -X POST http://<ip_publica_vm>:8080/job/jenkinsfile-01/build --user copa_lab:<token_recien_creado>

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **segundo jenkinsfile**.
  - Nombre: "**jenkinsfile-02**"
  - Copy from "**jenkinsfile-01**"
  - En "**Script Path**" escribe "**jenkinsfiles/Jenkinsfile-02.txt**" y después da clic en "**Save**"

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **segundo jenkinsfile**.
  - Ejecútalo una vez seleccionando las **acciones** en **directorios** "**Crear**" y en **archivos** "**Copia**"
  - curl -X POST http://<ip_publica_vm>:8080/job/jenkinsfile-02/build --user copa_lab:<token_recien_creado> --data-urlencode json='{"parameter": [{"name":"directorios", "value":"Crear"},{"name":"archivos", "value":"Copia"}, {"name":"nombre_llave", "value":"<nombre_llave>"}, {"name":"nombre_llave", "value":"<nombre_llave>"}]}'
  - Ejecútalo una vez seleccionando las **acciones** en **directorios** "**Borrar**" y en **archivos** "**Nada**"
  - curl -X POST http://<ip_publica_vm>:8080/job/jenkinsfile-02/build --user copa_lab:<token_recien_creado> --data-urlencode json='{"parameter": [{"name":"directorios", "value":"Borrar"},{"name":"archivos", "value":"Nada"}, {"name":"nombre_llave", "value":"<nombre_llave>"}]}'

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **tercer jenkinsfile**.
  - Nombre: "**jenkinsfile-03**"
  - Copy from "**jenkinsfile-02**"
  - En "**Script Path**" escribe "**jenkinsfiles/Jenkinsfile-03.txt**" y después da clic en "**Save**"
  - Ejecútalo una vez seleccionando la **acción** en **nginx** "**Instalar**"
  - curl -X POST http://<ip_publica_vm>:8080/job/jenkinsfile-03/build --user copa_lab:<token_recien_creado> --data-urlencode json='{"parameter": [{"name":"nombre_llave", "value":"<nombre_llave>"}, {"name":"nginx", "value":"Instala"}]}'
  - Intenta cargar a través de la IP pública de una de las VMs que creaste
  - http://<ip_vm_creada_Jenkins_pipeline>

CopaAirlines

Laboratorios – Comienza la diversión

- Ahora vamos a crear el **tercer jenkinsfile**.
  - Ejecútalo una vez seleccionando la **acción** en **nginx** "**Sube**"
  - curl -X POST http://<ip_publica_vm>:8080/job/jenkinsfile-03/build --user copa_lab:<token_recien_creado> --data-urlencode json='{"parameter": [{"name":"nombre_llave", "value":"<nombre_llave>"}, {"name":"nginx", "value":"Sube"}]}'
  - Intenta cargar a través de la IP pública de una de las VMs que creaste
  - http://<ip_vm_creada_Jenkins_pipeline>
  - Ejecútalo una vez seleccionando la **acción** en **nginx** "**Baja**"
  - curl -X POST http://<ip_publica_vm>:8080/job/jenkinsfile-03/build --user copa_lab:<token_recien_creado> --data-urlencode json='{"parameter": [{"name":"nombre_llave", "value":"<nombre_llave>"}, {"name":"nginx", "value":"Baja"}]}'
  - Intenta cargar a través de la IP pública de una de las VMs que creaste
  - http://<ip_vm_creada_Jenkins_pipeline>

CopaAirlines

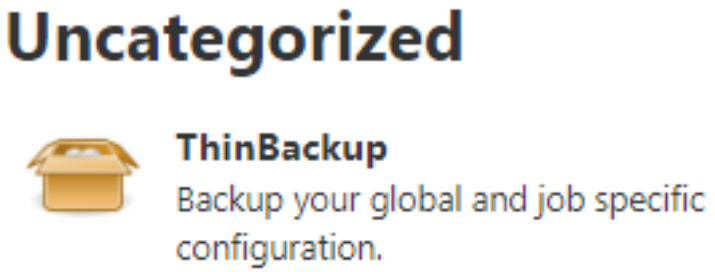
Laboratorios – Comienza la diversión

- Ahora vamos a crear el **tercer jenkinsfile**.
  - Ejecútalo una vez seleccionando la **acción** en **nginx** "**Despliega**"
  - curl -X POST http://<ip_publica_vm>:8080/job/jenkinsfile-03/build --user copa_lab:<token_recien_creado> --data-urlencode json='{"parameter": [{"name":"nombre_llave", "value":"<nombre_llave>"}, {"name":"nginx", "value":"Despliega"}]}'
  - Intenta cargar a través de la IP pública de una de las VMs que creaste
  - http://<ip_vm_creada_Jenkins_pipeline>

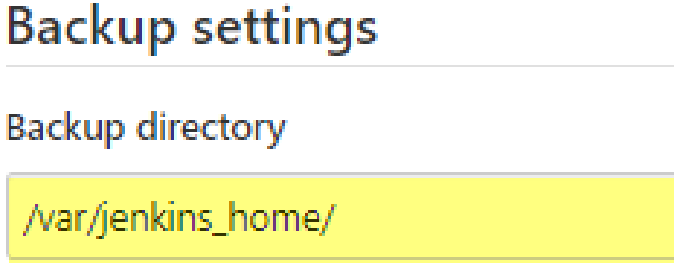
CopaAirlines

Respaldar información

- Para respaldar la información, ve a "**Manage Jenkins**" -> "**Manage Plugins**" -> "**Available**" y escribe "**thinbackup**", instálalo sin reiniciar Jenkins
- Ve a "**Manage Jenkins**", ve al final en "**ThinBackup**" da clic:



- Da clic en "**Settings**" y en "**Backup directory**" escribe "**/var/jenkins_home/**" y da clic en "**Save**"



Respaldar información

- Da clic en "**Backup Now**"

- Conéctate a la VM que está corriendo Jenkins
  - ssh -v -l ubuntu -i <llave> <ip_pública_vm>

- Conéctate al contenedor donde está corriendo Jenkins
  - docker exec -it jenkins /bin/bash

Respaldar información

- Ve al directorio "/var/jenkins_home" y lista el contenido
  - cd /var/jenkins_home
  - ls
- Los respaldos tienen este formato "**FULL-aaaa-mm-dd-hh-mm**"
- Comprime el directorio que tiene el respaldo
  - tar -zcvf jenkins.tar.gz <directorio>
- Sal del contenedor
  - exit

CopaAirlines

Respaldar información

- Identifica el ID del contenedor
  - docker ps

```
CONTAINER ID   IMAGE                          COMMAND                  CREATED        STATUS       PORTS
                                                                                  NAMES
d7201c6c9a88   jenkins/jenkins:2.285-slim   "/sbin/tini -- /usr/…"   5 weeks ago   Up 12 days   0.0.0.0:8080->
8080/tcp, :::8080->8080/tcp, 0.0.0.0:49154->50000/tcp, :::49154->50000/tcp   jenkins
```

- Copia el archivo del contenedor a la VM
  - docker cp <id_contenedor>:/var/jenkins_home/jenkins.tar.gz .

- Desde tu laptop y teniendo la llave de la VM en la que corre Jenkins, copia el respaldo a un directorio de la laptop
  - scp -i <llave> ubuntu@<ip_pública_vm>:/home/ubuntu/jenkins.tar.gz <directorio_laptop>

CopaAirlines

Restaurar información

- Desde tu laptop y teniendo la llave de la VM en la que corre Jenkins, copia el respaldo a la VM en la que corre Jenkins
  - scp -i <llave> <directorio_laptop>/jenkins.tar.gz ubuntu@<ip_publica_vm>:/home/ubuntu/jenkins.tar.gz
- Conéctate a la VM que está corriendo Jenkins
  - ssh -v -l ubuntu -i <llave> <ip_pública_vm>
- Copia el archivo del respaldo al contenedor
  - docker cp /home/ubuntu/jenkins.tar.gz <id_contenedor>:/var/jenkins_home
- Conéctate al contenedor donde está corriendo Jenkins
  - docker exec -it jenkins /bin/bash

CopaAirlines

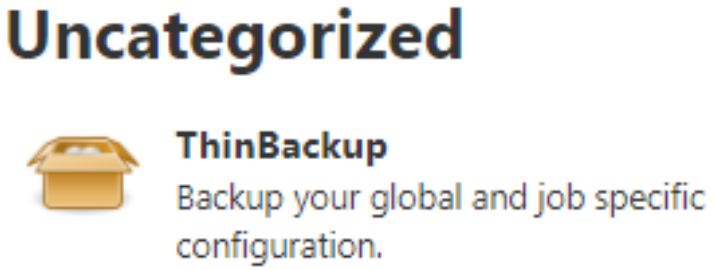
Restaurar información

- Ve al directorio "/var/jenkins_home" y lista el contenido
  - cd /var/jenkins_home
  - ls
- El respaldo aparecerá con el nombre "**jenkins.tar.gz**"
- Descomprime el directorio que tiene el respaldo
  - tar -xvf jenkins.tar.gz
- Sal del contenedor
  - exit
- Abre Jenkins desde el navegador
  - http://<ip_pública_vm>:8080

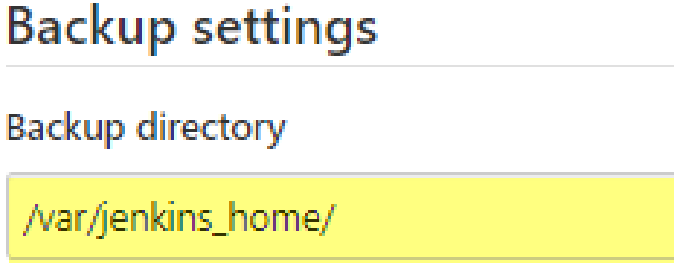
CopaAirlines

Restaurar información

- Para restaurar la información, ve a "**Manage Jenkins**" -> "**Manage Plugins**" -> "**Available**" y escribe "**thinbackup**", instálalo sin reiniciar Jenkins
- Ve a "**Manage Jenkins**", ve al final en "**ThinBackup**" da clic:

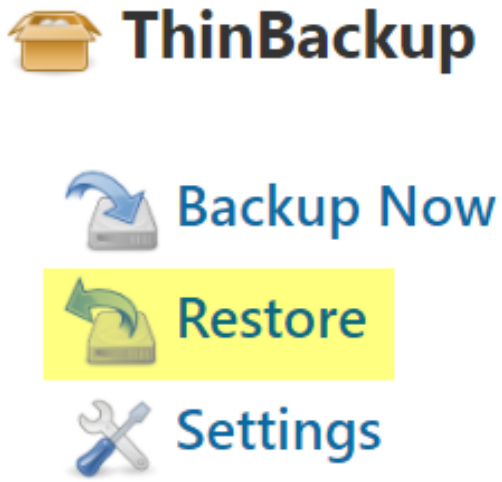


- Da clic en "**Settings**" y en "**Backup directory**" escribe "**/var/jenkins_home/**" y da clic en "**Save**"

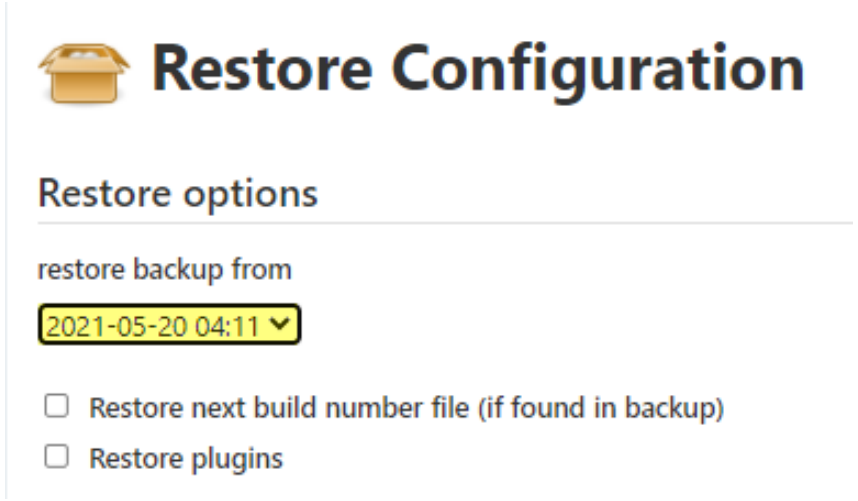


Restaurar información

- Da clic en "**Restore**"



- Selecciona el respaldo que deseas restaurar y da clic en "**Restore**"



Restaurar información

- En la VM baja y sube el contenedor Jenkins
  - docker stop jenkins
  - docker stop jenkins

- Abre Jenkins desde el navegador y deberá aparecerte los jobs, pipelines y demás información respaldada

CopaAirlines

Quiz

- Tiempo para repasar jugando

Gracias

CopaAirlines