

**UNIVERSIDAD NACIONAL DE INGENIERÍA**  
**FACULTAD DE CIENCIAS**



**ÁREA DE CIENCIA DE LA COMPUTACIÓN**  
**6° LABORATORIO - CC312**  
**2 PARTE**

- **TÍTULO:** Construya una canalización CI/CD usando Jenkins
- **ALUMNO:**  
JHONATAN POMA MARTINEZ 20182729F
- **PROFESORES:** YURI JAVIER.,CCOICCA PACASI

**2023**

## **OBJETIVOS:**

**Parte 1: Iniciar la Máquina Virtual (VM) DEVASC**

**Parte 2: Confirmar la aplicación de muestra a Git**

**Parte 3: Modificar la aplicación de muestra y enviar cambios a Git**

**Parte 4: Descargar y ejecutar la imagen de Jenkins Docker**

**Parte 5: Configurar Jenkins**

**Parte 6: Usar Jenkins para ejecutar una compilación de su aplicación**

**Parte 7: Usar Jenkins para probar una compilación**

**Parte 8: Crear una tubería o canalización en Jenkins**

## **Aspectos básicos/Situación**

En este laboratorio, se confirmará el código de la aplicación de muestra en un repositorio de GitHub, modificará el código localmente y, a continuación, confirmará los cambios. Igualmente, se instalará un contenedor Docker que incluye la última versión de Jenkins. Configure Jenkins y luego use Jenkins para descargar y ejecutar su programa de aplicación de muestra. Además, se creará un trabajo de prueba dentro de Jenkins que verificará que el programa de aplicación de muestra se ejecuta correctamente cada vez que lo compile. Finalmente, integrará su aplicación de ejemplo y el trabajo de prueba en una canalización de integración continua/desarrollo constante que verificará que su aplicación de muestra está lista para implementarse cada vez que cambie el código.

## **JENKINS:**

Jenkins es una herramienta de automatización de código abierto que se utiliza para la construcción, prueba y despliegue de software. Proporciona un entorno de integración continua (CI) y entrega continua (CD) para ayudar a los desarrolladores a detectar y solucionar problemas de manera temprana en el ciclo de vida del desarrollo de software. Jenkins permite a los equipos de desarrollo automatizar tareas repetitivas y crear flujos de trabajo personalizados para sus proyectos. Además, Jenkins tiene una gran cantidad de plugins que permiten integrar y extender su funcionalidad. Es una herramienta muy popular y utilizada por muchos equipos de desarrollo en todo el mundo.

## **CI/CD:**

Son las siglas de **Continuous Integration / Continuous Delivery** (Entrega Continua), que son prácticas de desarrollo de software para automatizar el proceso de construcción, prueba y entrega de software. La Integración Continua (CI) se enfoca en asegurar que el código se integre y construya correctamente, mientras que la Entrega Continua (CD) se enfoca en automatizar la entrega del software en un ambiente productivo, lo que implica pruebas, revisiones de seguridad y despliegues en producción. La idea es que el desarrollo de software sea más eficiente, con menos errores y con entregas más rápidas y confiables.

## Parte 1: Iniciar la Máquina Virtual (VM) DEVASC

Maquina virtual iniciada.

## Parte 2: Confirmar la aplicación de muestra a Git

- Configuramos nuestras credenciales de Git localmente en la máquina virtual (VM).

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  bash + - [ ] [X] ... ^
devasc@labvm:~$ git config -global user.name "JPomaMartinez"
error: did you mean '--global' (with two dashes)?
devasc@labvm:~$ git config --global user.name "JPomaMartinez"
devasc@labvm:~$ git config --global user.email "jhonatan.poma.m@uni.pe"
devasc@labvm:~$
devasc@labvm:~$
```

- Inicializamos el git.
- **git remote add origin.....** es un comando que se usara en Git para agregar un nuevo repositorio remoto como un alias con nombre a un repositorio local de Git.
- **git add \*** se usara para añadir todos los archivos modificados en el directorio actual y sus subdirectorios al área de preparación de Git.
- **git status** nos ayudara a mostrar el estado actual del repositorio local. Cuando lo ejecutamos, Git nos mostrara información sobre los cambios realizados en el repositorio local desde el último commit.
- **git commit** lo usaremos para confirmar los archivos almacenados y comenzar a rastrear los cambios. Agregaremos un mensaje de nuestra elección.

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  bash + - [ ] [X] ... ^ X
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git init
Initialized empty Git repository in /home/devasc/labs/devnet-src/jenkins/sample-app/.git/
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git remote add origin https://github.com/JPomaMartinez/sample-app.git
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git add *
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   sample-app.sh
    new file:   sample_app.py
    new file:   static/style.css
    new file:   templates/index.html

devasc@labvm:~/labs/devnet-src/jenkins/sample-app$
```

- **git push origin master** es un comando en Git para enviar los cambios realizados en la rama master del repositorio local al repositorio remoto origin. Ingresamos nuestro username y nuestro password, en este caso me pidio hacer un token en mi github.

```

● devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git push origin master
Username for 'https://github.com': JPomaMartinez
Password for 'https://JPomaMartinez@github.com':
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 2 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (8/8), 1.04 KiB | 76.00 KiB/s, done.
Total 8 (delta 0), reused 0 (delta 0)
To https://github.com/JPomaMartinez/sample-app.git
 * [new branch]      master -> master
○ devasc@labvm:~/labs/devnet-src/jenkins/sample-app$

```

### Parte 3: Modificar la aplicación de muestra y enviar cambios a Git

- Cambiamos de puerto en los archivos sample\_app.py y sample-app.sh

```

File Edit View Search Terminal Help
GNU nano 4.8 sample_app.py
# Add to this file for the sample app lab
from flask import Flask
from flask import request
from flask import render_template

sample = Flask(__name__)

@sample.route("/")
def main():
    return render_template("index.html")

if __name__ == "__main__":
    sample.run(host="0.0.0.0", port=5050)

```

```

GNU nano 4.8 sample-app.sh
mkdir tempdir/templates
mkdir tempdir/static

cp sample_app.py tempdir/.
cp -r templates/* tempdir/templates/.
cp -r static/* tempdir/static/.

echo "FROM python" >> tempdir/Dockerfile
echo "RUN pip install flask" >> tempdir/Dockerfile
echo "COPY ./static /home/myapp/static/" >> tempdir/Dockerfile
echo "COPY ./templates /home/myapp/templates/" >> tempdir/Dockerfile
echo "COPY sample_app.py /home/myapp/" >> tempdir/Dockerfile
echo "EXPOSE 5050" >> tempdir/Dockerfile
echo "CMD python /home/myapp/sample_app.py" >> tempdir/Dockerfile

cd tempdir
docker build -t sampleapp .
docker run -t -d -p 5050:5050 --name samplerunning sampleapp
docker ps -a

```

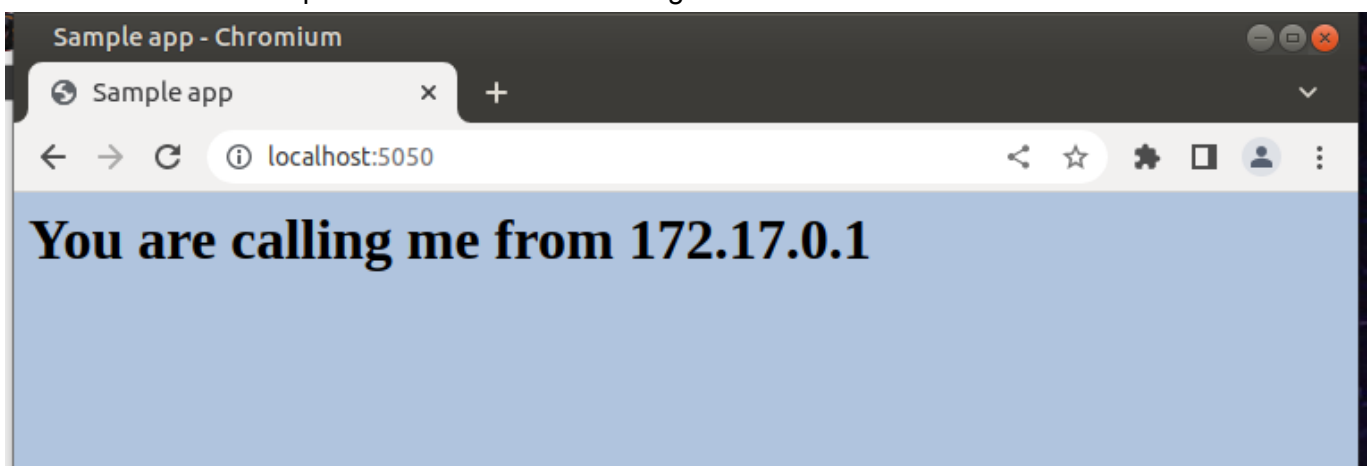
- Ejecutamos el sample-app.sh

```

devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ bash ./sample-app.sh
Sending build context to Docker daemon 6.144kB
Step 1/7 : FROM python
---> 815c8c75dfc0
Step 2/7 : RUN pip install flask
---> Using cache
---> 4b46416de15b
Step 3/7 : COPY ./static /home/myapp/static/
---> 048df4e67ab8
Step 4/7 : COPY ./templates /home/myapp/templates/
---> 6f67803cdecd
Step 5/7 : COPY sample_app.py /home/myapp/
---> cba134922767
Step 6/7 : EXPOSE 5050
---> Running in 6bba289a4988
Removing intermediate container 6bba289a4988
---> 0736dfbbc0fb
Step 7/7 : CMD python /home/myapp/sample_app.py
---> Running in cc1de4fd171a
Removing intermediate container cc1de4fd171a
---> 774d62af76ed
Successfully built 774d62af76ed
Successfully tagged sampleapp:latest
ba17e25efb82c24442106ebf9ff423e8b9f7d241f345a0ea0d6c08496fbe8244
CONTAINER ID          IMAGE          COMMAND          CREATED
STATUS                PORTS         NAMES
ba17e25efb82          sampleapp     "/bin/sh -c 'python ..." 4 seconds ago
Up Less than a second 0.0.0.0:5050->5050/tcp    samplerunning
9672ddaa46bd          2365ec36406a  "/bin/sh -c 'python3..." 4 days ago
Created                                tender_keldysh
03f6c04a2c27          2365ec36406a  "/bin/sh -c 'python3..." 4 days ago
Exited (255) 4 days ago 0.0.0.0:8080->8080/tcp    dazzling_lehmann
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$

```

- Corroboramos que esta llamando cuando ingresamos localhost:5050



- Guardamos los cambios y hacemos un commit , despues un push a nuestro repositorio.

```








devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ docker stop samplerunning
samplerunning
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git add *
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   sample-app.sh
    modified:   sample_app.py
    new file:   tempdir/Dockerfile
    new file:   tempdir/sample_app.py
    new file:   tempdir/static/style.css
    new file:   tempdir/templates/index.html

devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git commit -m "Se cambio el p
uerto de 8080 a 5050, Administracion de REDES"
[master 1f530ad] Se cambio el puerto de 8080 a 5050, Administracion de REDES
6 files changed, 33 insertions(+), 3 deletions(-)
create mode 100644 tempdir/Dockerfile
create mode 100644 tempdir/sample_app.py
create mode 100644 tempdir/static/style.css
create mode 100644 tempdir/templates/index.html
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git push origin master

devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git push origin master
Username for 'https://github.com': JPomaMartinez
Password for 'https://JPomaMartinez@github.com':
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 2 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 780 bytes | 70.00 KiB/s, done.
Total 6 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/JPomaMartinez/sample-app.git
    20068c..1f530ad  master -> master
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$

```

- Verificando el commit y el push que hicimos en nuestro repositorio.

	<b>JPomaMartinez</b>	Se cambio el puerto...	...	5 minutes ago	 2
	static	Confirmando archivos de aplic...		yesterday	
	tempdir	Se cambio el puerto de 8080 a...		5 minutes ago	
	templates	Confirmando archivos de aplic...		yesterday	
	sample-app.sh	Se cambio el puerto de 8080 a...		5 minutes ago	
	sample_app...	Se cambio el puerto de 8080 a...		5 minutes ago	



#### Parte 4: Descargar y ejecutar la imagen de Jenkins Docker

- Descargamos la imagen de JenkinsDocker

```
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ docker pull jenkins/jenkins
Using default tag: latest
latest: Pulling from jenkins/jenkins
918547b94326: Already exists
c0be97cc2282: Pull complete
140f56184639: Pull complete
c22aadcc6d63: Pull complete
79559a9f5d3d: Pull complete
7838486c1e96: Pull complete
5dc223965d32: Pull complete
1ced8ab6ace5: Pull complete
e6019e9b8ce7: Pull complete
da1a9c4a7ebd: Pull complete
ac9b408013e6: Pull complete
18ad4727f85f: Pull complete
c8a36db3fbbb: Pull complete
Digest: sha256:37b42880c4046fab6000e2db3308a1d2c2fb2fb0f6b034bf6cc1685e9de4d3b7
Status: Downloaded newer image for jenkins/jenkins:latest
docker.io/jenkins/jenkins:latest
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$
```

- Arrancamos el contenedor de JenkinsDocker con el comando :  
**docker run —rm -u root -p 8080:8080 -v jenkins-data: /var/jenkins\_home -v \$ (que docker):/usr/bin/docker -v /var/run/docker.sock:/sovar/run/docker.ck -v «\$HOME»:/home —name jenkins\_server jenkins/jenkins:its**
- Vemos que nos dan una contraseña y la ubicación en donde esta se guarda:  
/var/jenkins\_home/secrets/initialAdminPassword

```
*****
*****
*****

Jenkins initial setup is required. An admin user has been created and a password
generated.
Please use the following password to proceed to installation:

79e76b50329d4851bab4d080abdf592e

This may also be found at: /var/jenkins_home/secrets/initialAdminPassword

*****
*****
*****
```

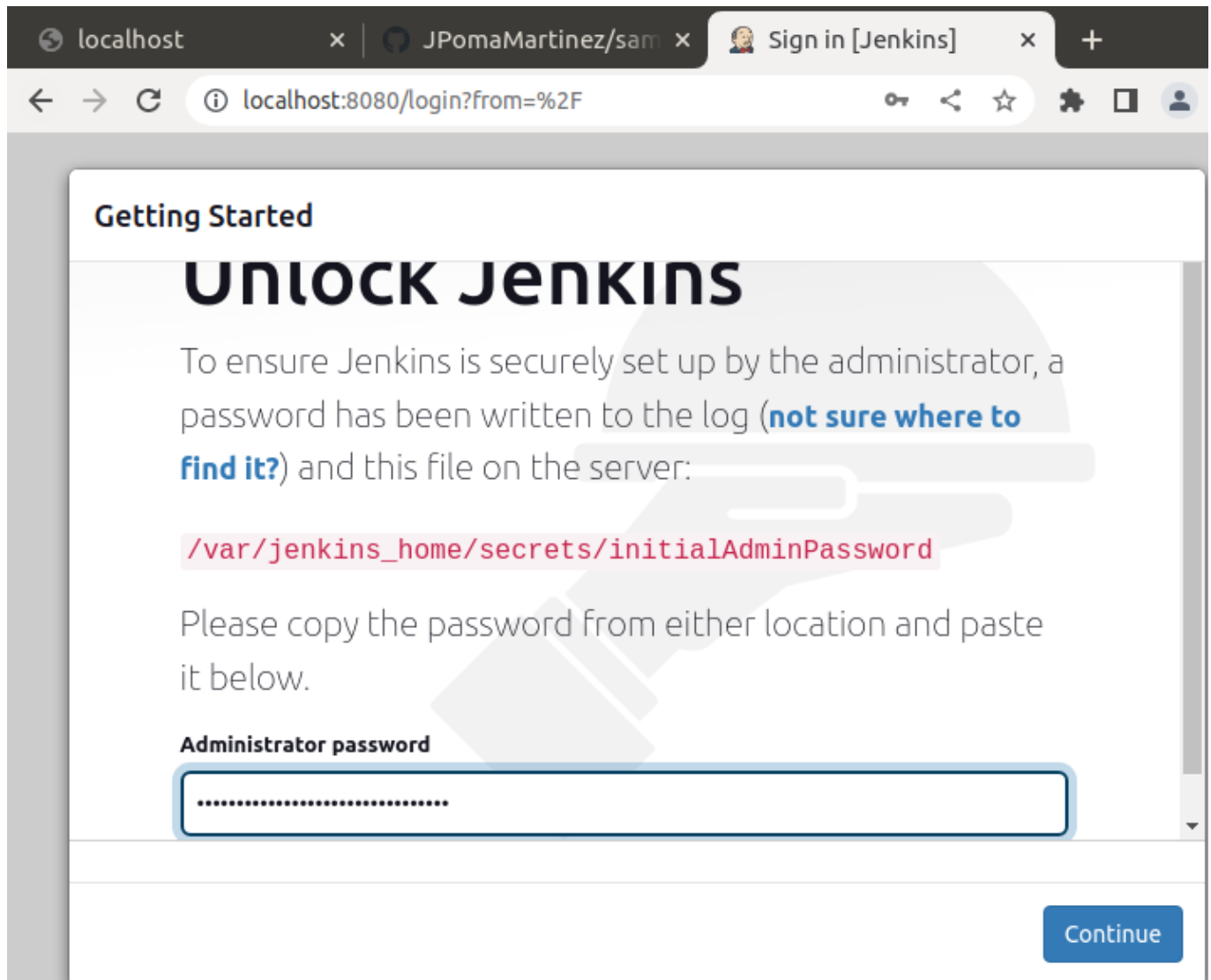
En la guía nos dice que si perdemos la contraseña o no se muestra como se muestra arriba, o necesita reiniciar el servidor Jenkins,  
siempre puede recuperar la contraseña accediendo a la línea de comandos del contenedor Jenkins Docker. En una ventana de terminal en VS Code e ingrese los siguientes comandos para que no detenga el servidor Jenkins.:  
Y ahí también nos saldrá el password.

```
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ docker exec -it jenkins_server /bin/bash
root@cb0652c113b9:/# cat /var/jenkins_home/secrets/initialAdminPassword
79e76b50329d4851bab4d080abdf592e
root@cb0652c113b9:/#
```

- El servidor Jenkins debería estar ahora en ejecución. Copiamos la contraseña de administrador que aparece en la salida, como se mostro en la ventana anterior.

## Parte 5: Configurar Jenkins

- Vaya a <http://localhost:8080/> e inicie sesión con su contraseña de administrador copiada.



localhost x | JPomaMartinez/sam x Sign in [Jenkins] x +

localhost:8080/login?from=%2F

### Getting Started

# UNLOCK JENKINS

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (**not sure where to find it?**) and this file on the server:

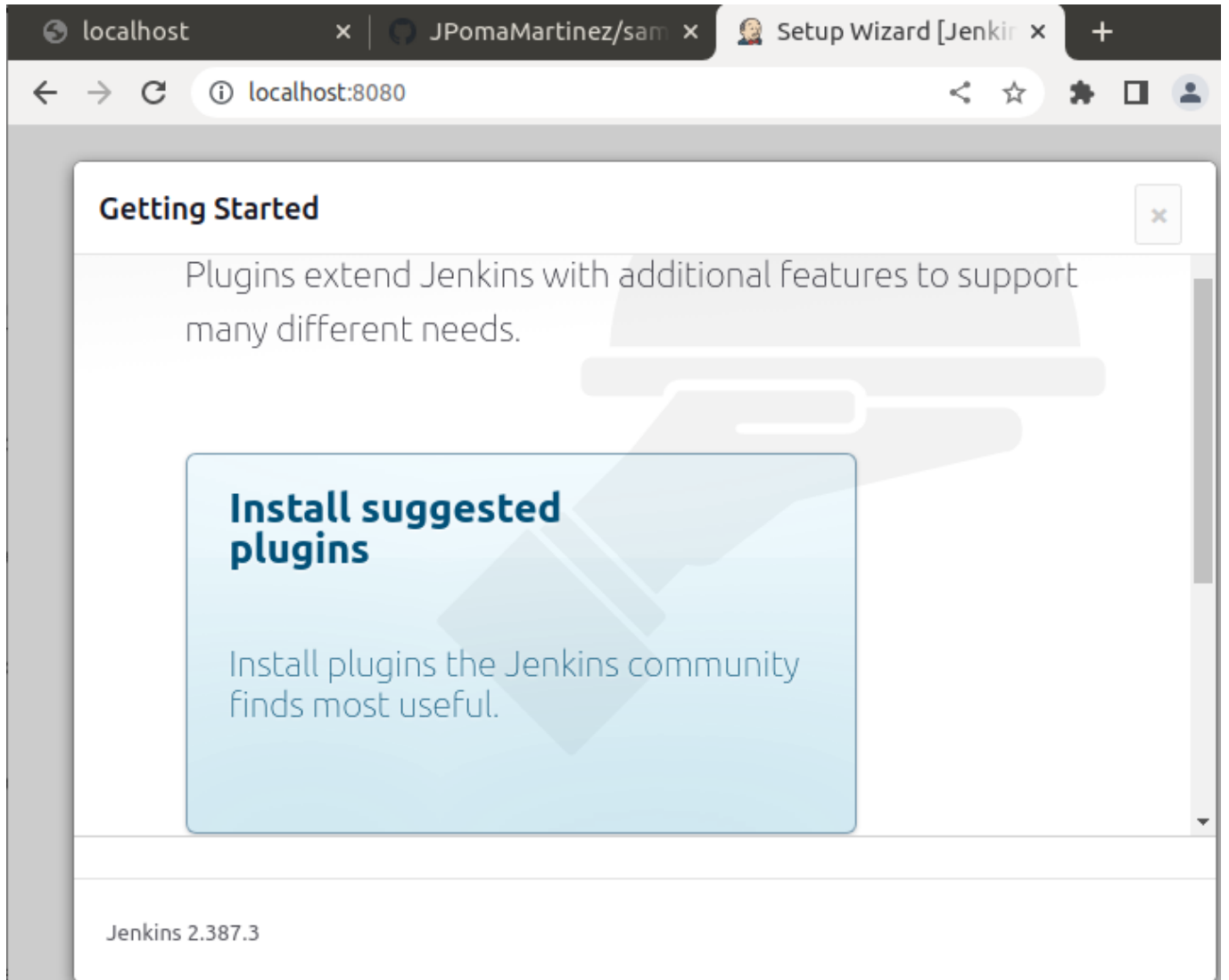
```
/var/jenkins_home/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

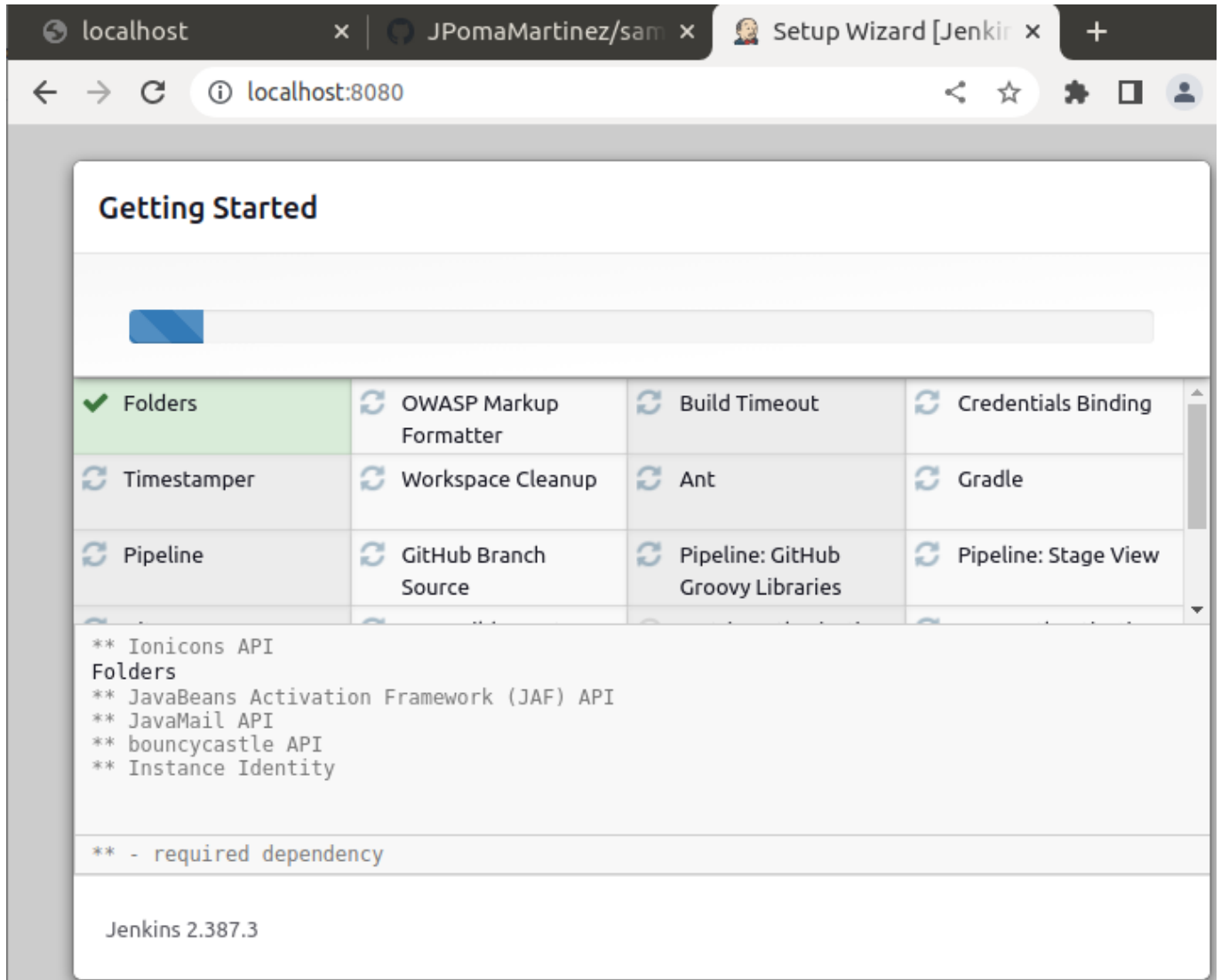
**Administrator password**

Continue





- Instalamos los complementos sugeridos.



- Omitimos la creación de un nuevo usuario administrador le damos a “Skip and continue as admin”:

**Getting Started**

## Create First Admin User

Username

Password

Confirm password

Jenkins 2.387.3

[Skip and continue as admin](#)

- Omitimos la creación de una instancia, y hacemos click en “not now”

# Configuration

Jenkins URL:

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the BUILD\_URL environment variable provided to build steps.

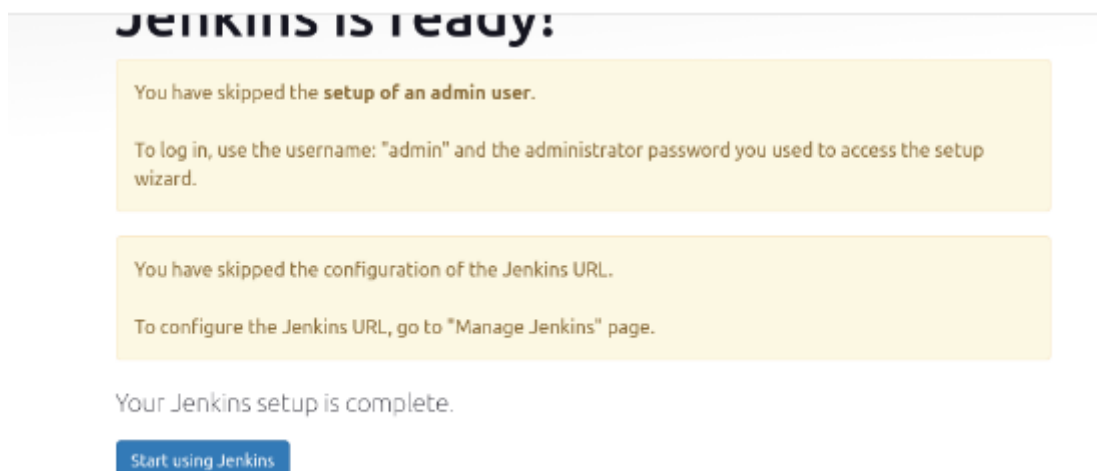
The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

Jenkins 2.387.3

[Not now](#)

[Save and Finish](#)

- En la siguiente ventana, hacemos click en “Empezar a usar Jenkins”. Ahora debería estar en el panel principal con un mensaje similar a bienvenido a Jenkins! .



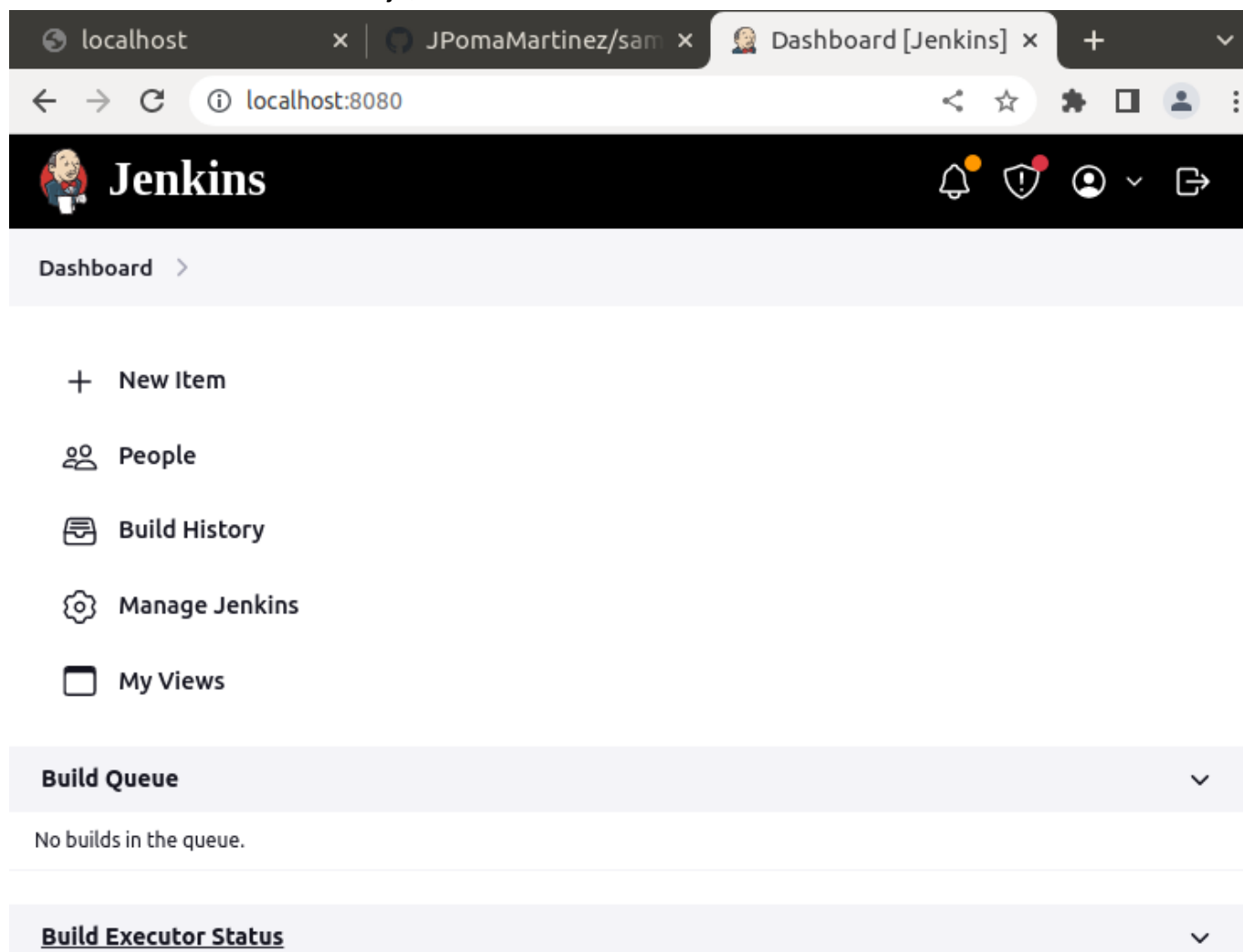
## Parte 6: Usar Jenkins para ejecutar una compilación de su aplicación

Puede crear trabajos que realicen una variedad de tareas, entre las que se incluyen las siguientes:

1. Recuperar código de un repositorio de administración de código fuente como GitHub.
2. Compilar una aplicación utilizando una secuencia de comandos (script) o una herramienta de compilación.
3. Empaquetar una aplicación y ejecutarla en un servidor

En esta parte, creará un trabajo de Jenkins simple que recupera la última versión de su aplicación de muestra de GitHub y ejecuta el script de compilación.






- Creamos un nuevo trabajo, hacemos click en "NEW ITEM"



The screenshot shows the Jenkins web interface in a browser. The browser's address bar displays 'localhost:8080'. The Jenkins logo and name are at the top left, with navigation icons on the right. A sidebar on the left contains links: '+ New Item', 'People', 'Build History', 'Manage Jenkins', and 'My Views'. The main content area features two expandable sections: 'Build Queue' and 'Build Executor Status'. The 'Build Queue' section is currently expanded, showing the message 'No builds in the queue.'.


localhost × | JPomaMartinez/sam × | Dashboard [Jenkins] × +


← → ↻ ⓘ localhost:8080


 **Jenkins**    ▾ 


Dashboard >

+ New Item

 People

 Build History

 Manage Jenkins

 My Views

**Build Queue** ▾

No builds in the queue.

**Build Executor Status** ▾

- Le ponemos como nombre BuildAppJob

localhost x JPomaMartinez/sam x New Item [Jenkins] x +

localhost:8080/view/all/newJob


# Jenkins

Dashboard > All >

## Enter an item name


BuildAppJob

» Required field



### Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.



### Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities

- Y lo creamos como Proyecto de estilo libre (Freestyle Project).
- Configuramos el **Jenkins BuildAppJob**, al configurar en la parte de credenciales nos saldra este error que indica que la conexión ha fallado. Esto se debe a que aún no ha seleccionado las credenciales.

JPomaMartinez/sample-a x BuildAppJob Config [Jenl x +

localhost:8080/job/BuildAppJob/configure

Dashboard > BuildAppJob > Configuration

Git ?

Repositories ?

Repository URL ?

https://github.com/JPomaMartinez/sample-app.git

Failed to connect to repository : Command "git ls-remote -h -- https://github.com/JPomaMartinez/sample-app.git HEAD" returned status code 128:  
stdout:  
stderr: remote: Support for password authentication was removed on August 13, 2021.  
remote: Please see https://docs.github.com/en/get-started/getting-started-with-git/about-remote-repositories#cloning-with-https-urls for information on currently recommended modes of authentication.  
fatal: Authentication failed for 'https://github.com/JPomaMartinez/sample-app.git/'

Credentials ?

- Ahora en credenciales insertamos:

Credentials ?



JPomaMartinez/\*\*\*\*\*


Add ▾

- Hacemos que Jenkins compile la ejecución



## Build Steps

 **Execute shell** 



**Command**

See [the list of available environment variables](#)

```
bash ./sample-app.sh
```

**Parte 7: Usar Jenkins para probar una compilación**

**Parte 8: Crear una tubería o canalización en Jenkins**

### **RESUMEN:**

**PARTE2:** Aquí hemos aprendido o recordado de como crear una repositorio en Github y realizar algunos comandos para guardar o actualizar nuestro repositorio.

PARTE3: En este apartado realizamos un cambio de puerto de 8080 al 5050 y hicimos un commit y push en nuestro repositorio github. como en la PARTE4 instalaremos una imagen de JENKINS DOCKER y usaremos el puerto 8080 , por eso hicimos el cambio de puerto.

