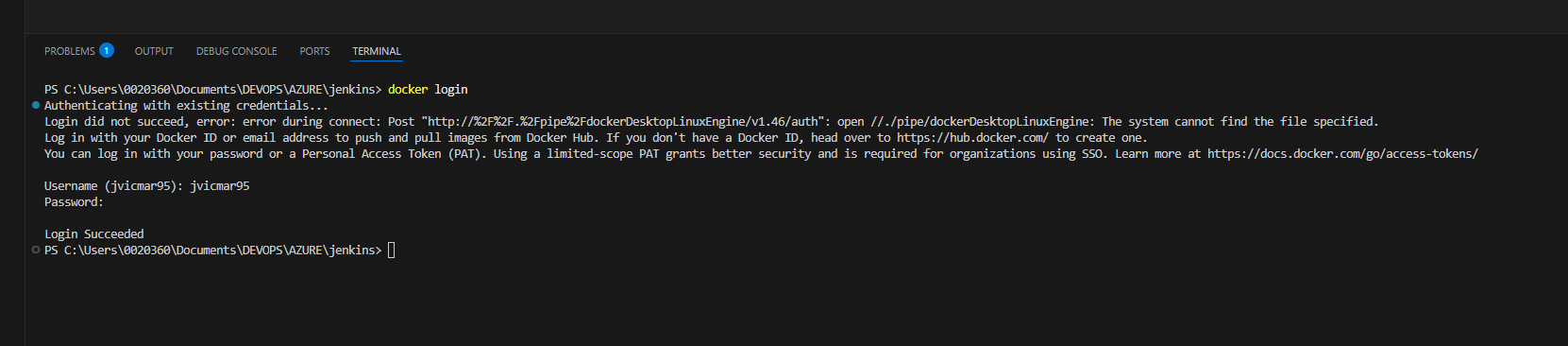
# Dockerhub

[jvicmar95@gmail.com](mailto:jvicmar95@gmail.com)

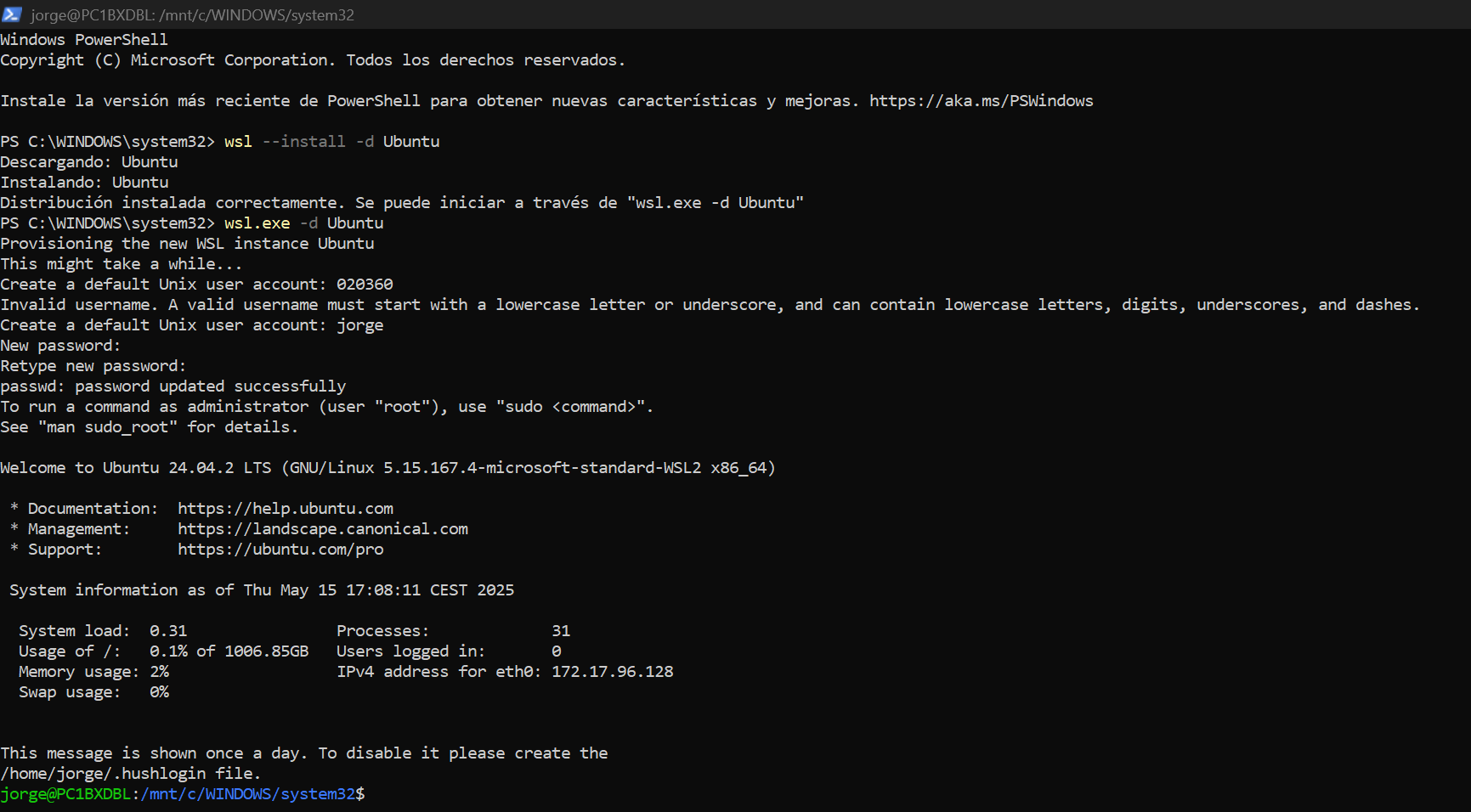
America0000001@

## Login en dockerhub

Docker login



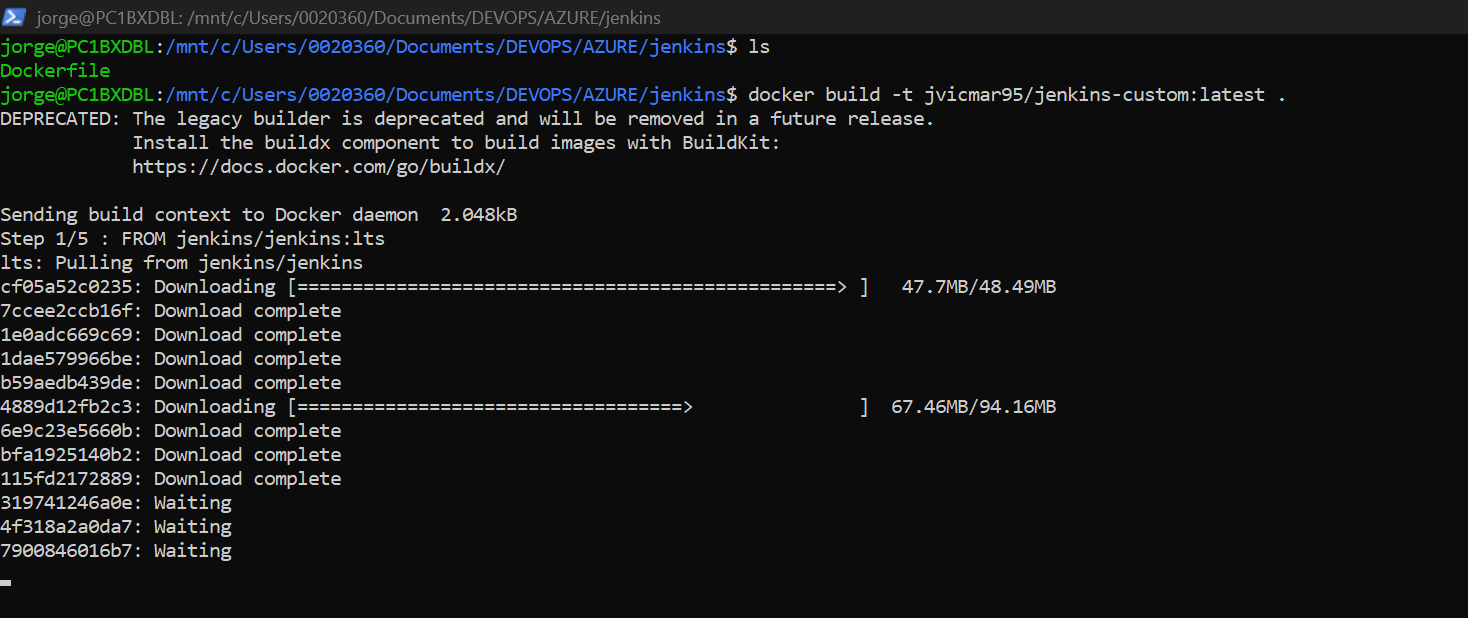
## Tener puesto Docker en nuestro pc



wsl.exe -d Ubuntu

Jorge / America0000001@

Y Aquí crear la imagen



## Build de la imagen con el Dockerfile

FROM jenkins/jenkins:lts

USER root

# Instalar Docker y Git

RUN apt-get update && \

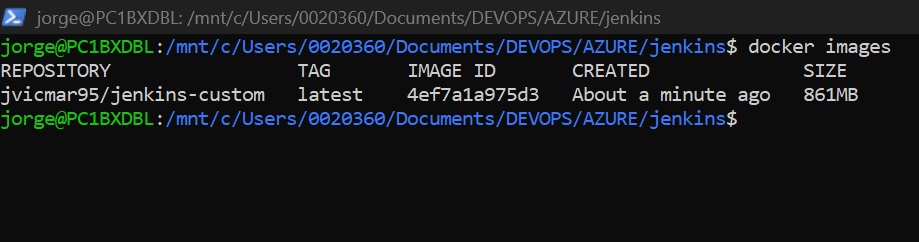
    apt-get install -y docker.io git && \

    apt-get clean

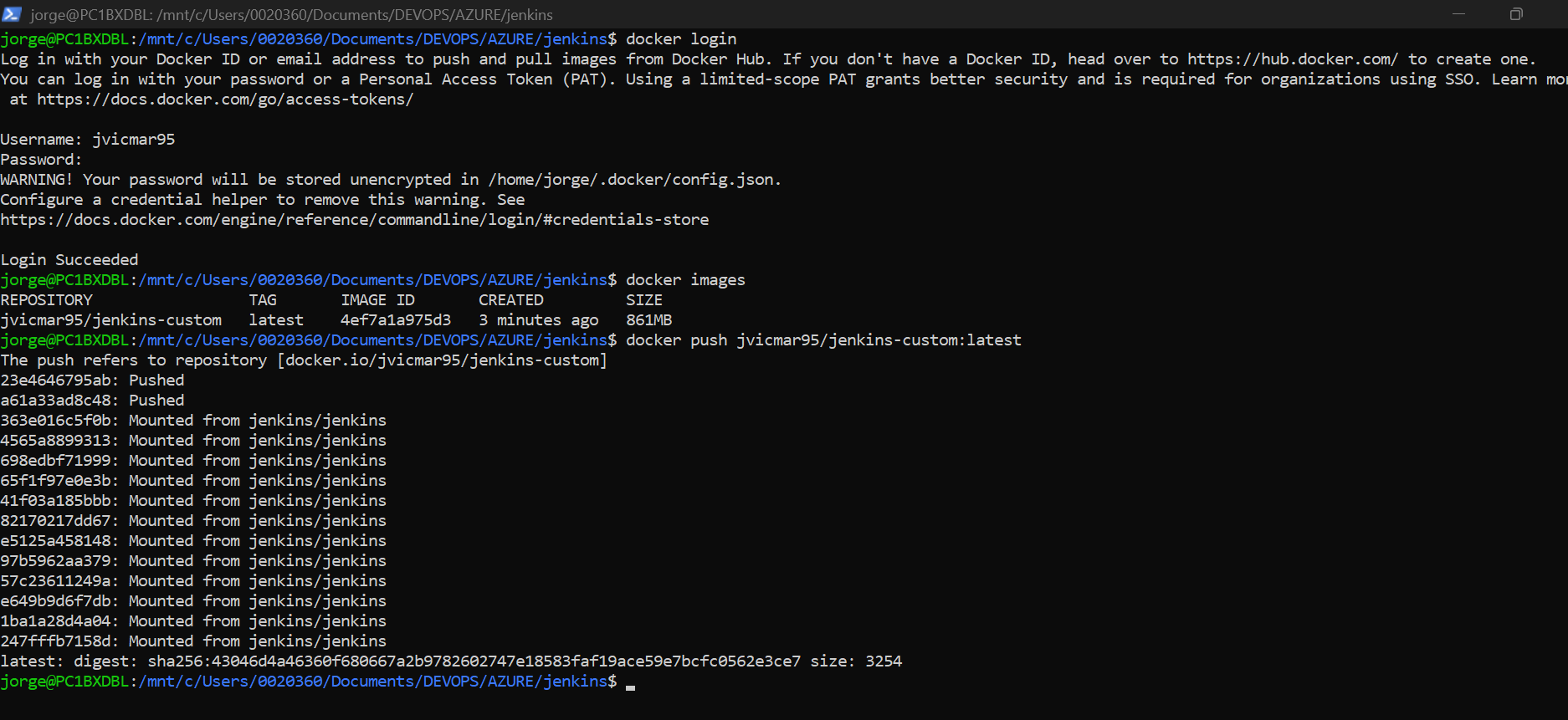
# Agregar al usuario Jenkins al grupo docker

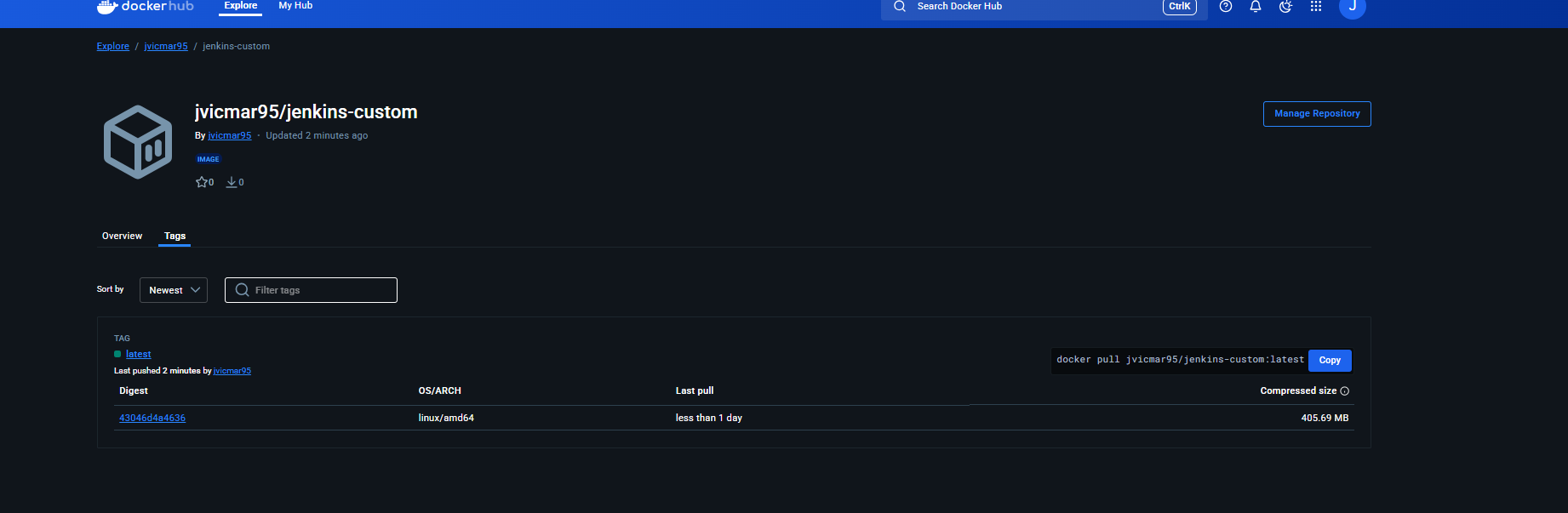
RUN usermod -aG docker jenkins

USER jenkins



## Hacer push





jvicmar95/jenkins-custom

Tags for jvicmar95/jenkins-custom

docker pull jvicmar95/jenkins-custom:latest

# Crear cluster y HELM instalado

PS C:\Users\0020360\Documents\DEVOPS\AZURE\jenkins> kubectl get nodes

NAME STATUS ROLES AGE VERSION

pool-9fr8f85c4-tqsob Ready <none> 3m6s v1.32.2

PS C:\Users\0020360\Documents\DEVOPS\AZURE\jenkins> helm version

version.BuildInfo{Version:"v3.11.1", GitCommit:"293b50c65d4d56187cd4e2f390f0ada46b4c4737", GitTreeState:"clean", GoVersion:"go1.18.10"}

PS C:\Users\0020360\Documents\DEVOPS\AZURE\jenkins>

# PROCEDIMIENTO

AÑADIR REPO JENKINS

helm repo add jenkins <https://charts.jenkins.io>

añadir Jenkins-values.yaml

controller:

  image:

    repository: jvicmar95/jenkins-custom

    tag: latest

    pullPolicy: IfNotPresent

  installPlugins:

    - kubernetes:latest

    - workflow-aggregator:latest

    - git:latest

    - docker-workflow:latest

  serviceType: LoadBalancer

  admin:

    username: admin

    password: admin

  resources:

    requests:

      cpu: "500m"

      memory: "1Gi"

    limits:

      cpu: "1000m"

      memory: "2Gi"

  persistence:

    enabled: true

    size: 8Gi

    storageClass: do-block-storage

helm repo update

PS C:\Users\0020360\Documents\DEVOPS\AZURE\jenkins> helm install jenkins jenkins/jenkins -n jenkins --create-namespace -f jenkins-values.yaml

NAME: jenkins

LAST DEPLOYED: Thu May 15 17:32:22 2025

NAMESPACE: jenkins

STATUS: deployed

REVISION: 1

NOTES:

1. Get your 'admin' user password by running:

kubectl exec --namespace jenkins -it svc/jenkins -c jenkins -- /bin/cat /run/secrets/additional/chart-admin-password && echo

2. Get the Jenkins URL to visit by running these commands in the same shell:

NOTE: It may take a few minutes for the LoadBalancer IP to be available.

You can watch the status of by running 'kubectl get svc --namespace jenkins -w jenkins'

export SERVICE\_IP=$(kubectl get svc --namespace jenkins jenkins --template "{{ range (index .status.loadBalancer.ingress 0) }}{{ . }}{{ end }}")

echo http://$SERVICE\_IP:8080

3. Login with the password from step 1 and the username: admin

4. Configure security realm and authorization strategy

5. Use Jenkins Configuration as Code by specifying configScripts in your values.yaml file, see documentation: http://$SERVICE\_IP:8080/configuration-as-code and examples: https://github.com/jenkinsci/configuration-as-code-plugin/tree/master/demos

For more information on running Jenkins on Kubernetes, visit:

https://cloud.google.com/solutions/jenkins-on-container-engine

For more information about Jenkins Configuration as Code, visit:

https://jenkins.io/projects/jcasc/

NOTE: Consider disabling `installPlugins` if your image already contains plugins.

PS C:\Users\0020360\Documents\DEVOPS\AZURE\jenkins>

ACCEDER JENKINS

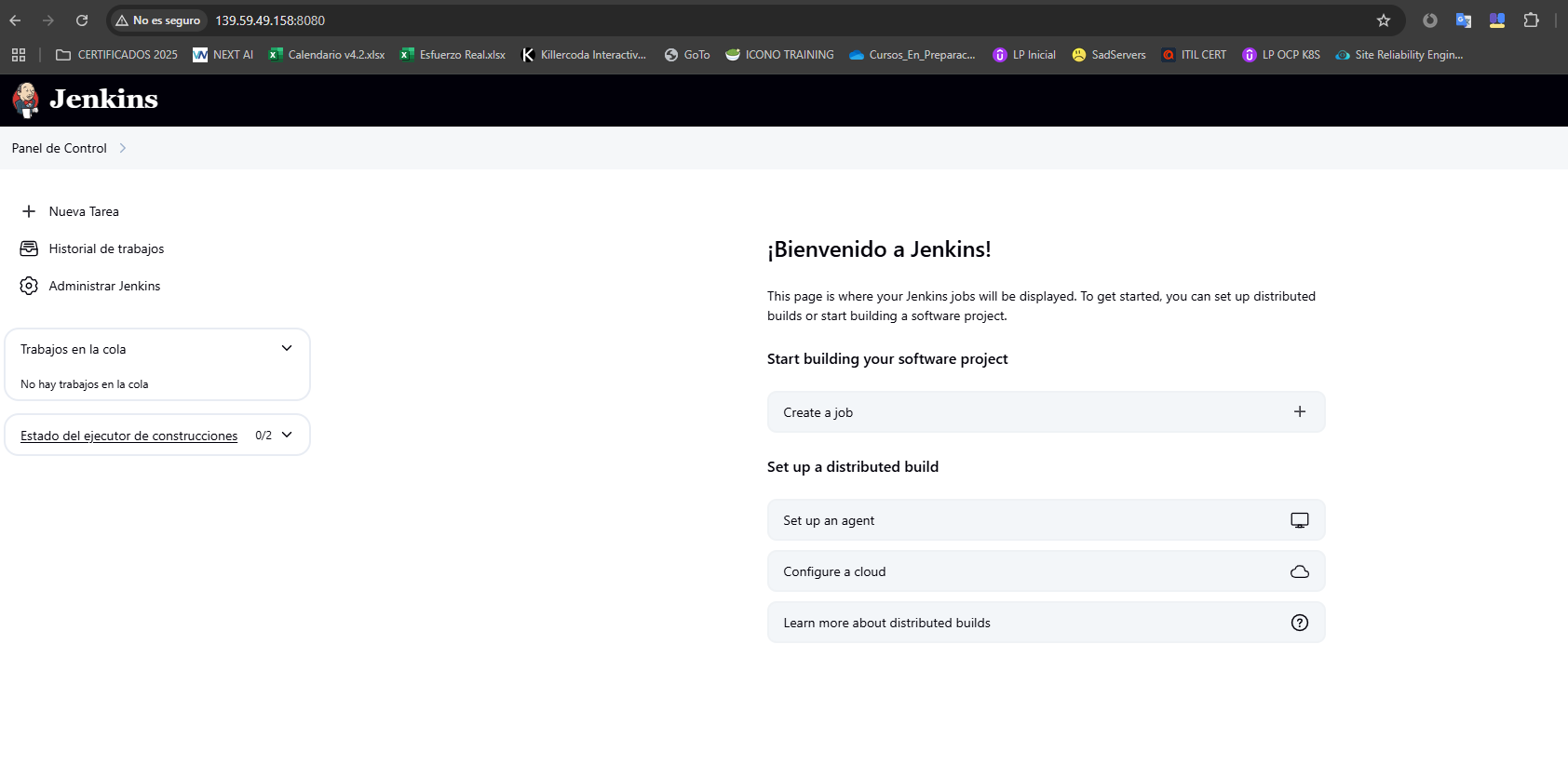
PS C:\Users\0020360\Documents\DEVOPS\AZURE\jenkins> kubectl get svc -n jenkins

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

jenkins LoadBalancer 10.109.11.69 139.59.49.158,2400:6180:100:d0::7dce:7001 8080:30173/TCP 2m37s

jenkins-agent ClusterIP 10.109.3.47 <none> 50000/TCP 2m37s

PS C:\Users\0020360\Documents\DEVOPS\AZURE\jenkins>



Verificar imagen

PS C:\Users\0020360\Documents\DEVOPS\AZURE\jenkins> kubectl get pods -n jenkins -o jsonpath="{.items[0].spec.containers[0].image}"

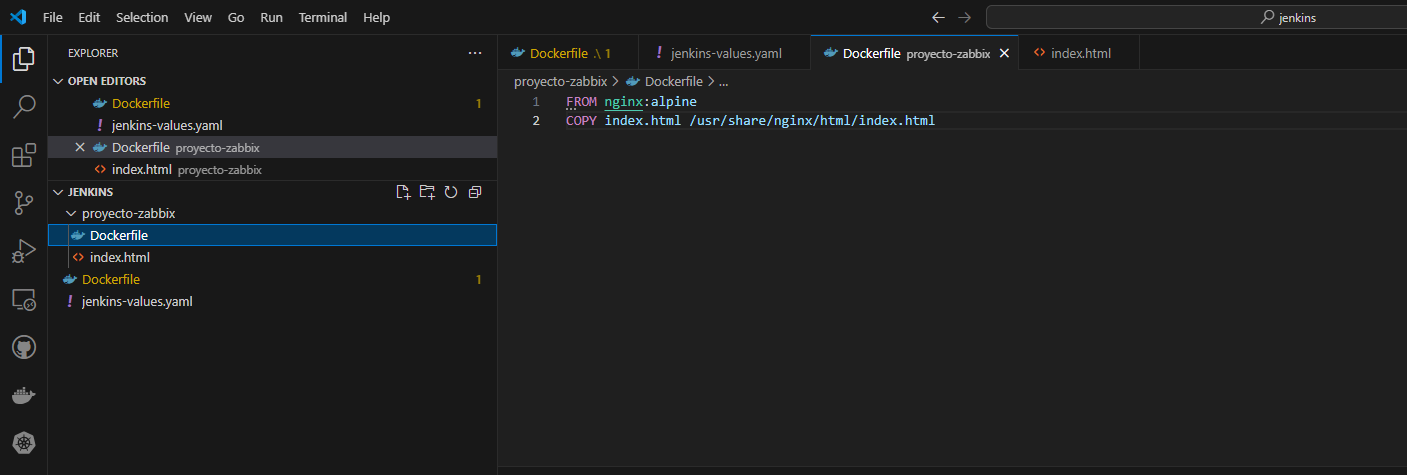
docker.io/jvicmar95/jenkins-custom:latest

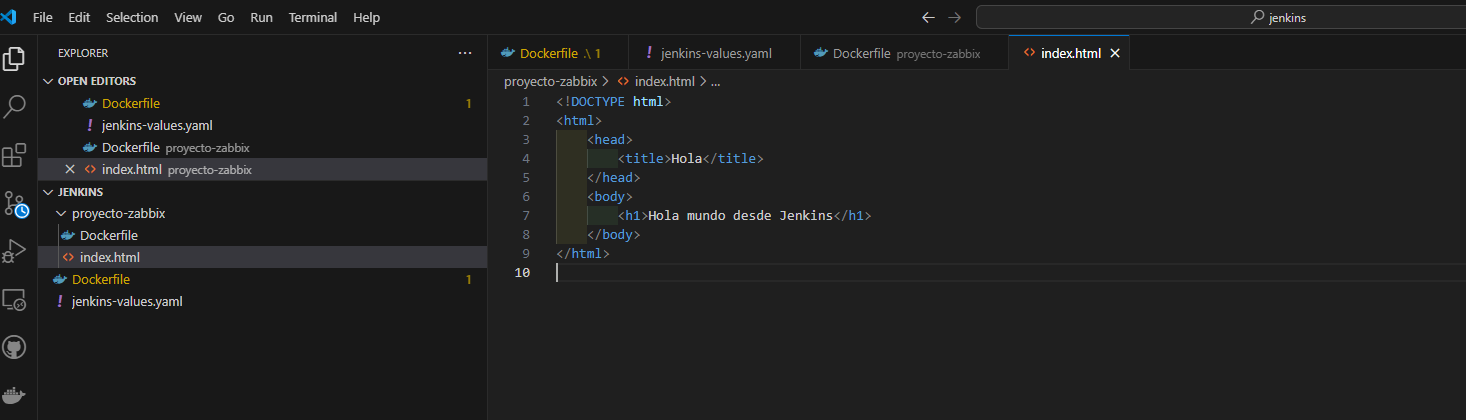
# GITHUB

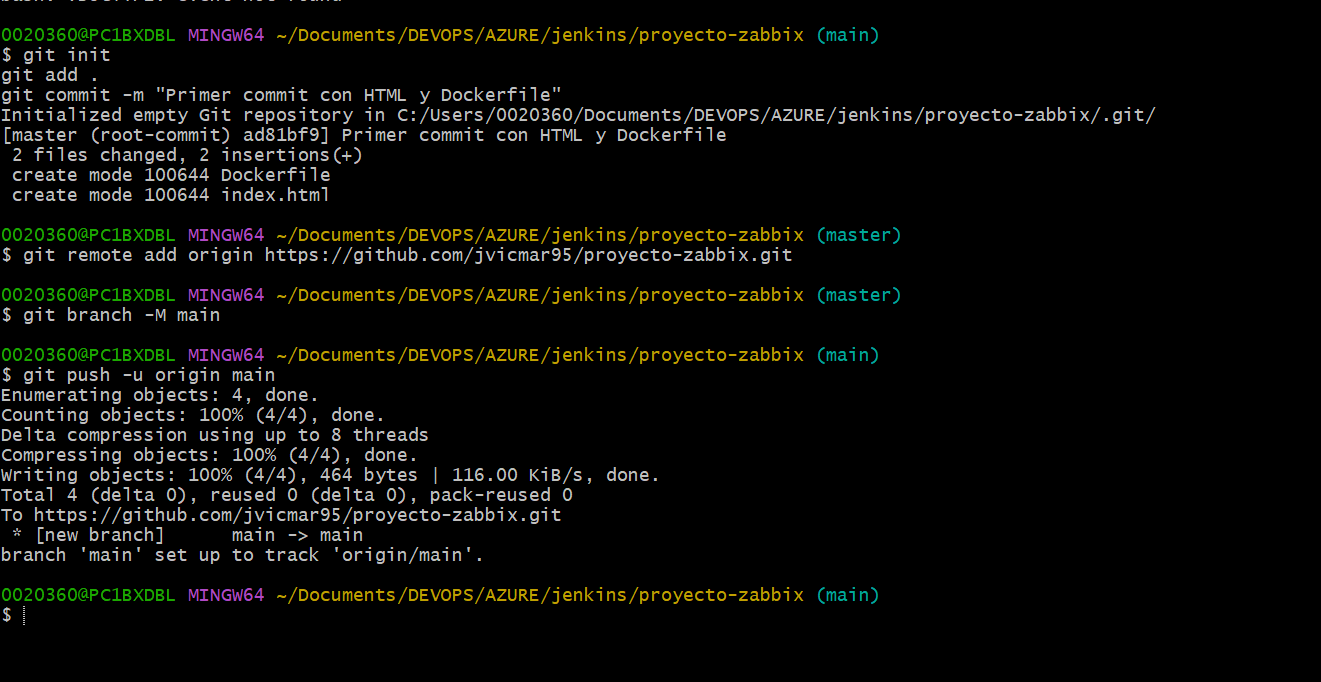
jvicmar95

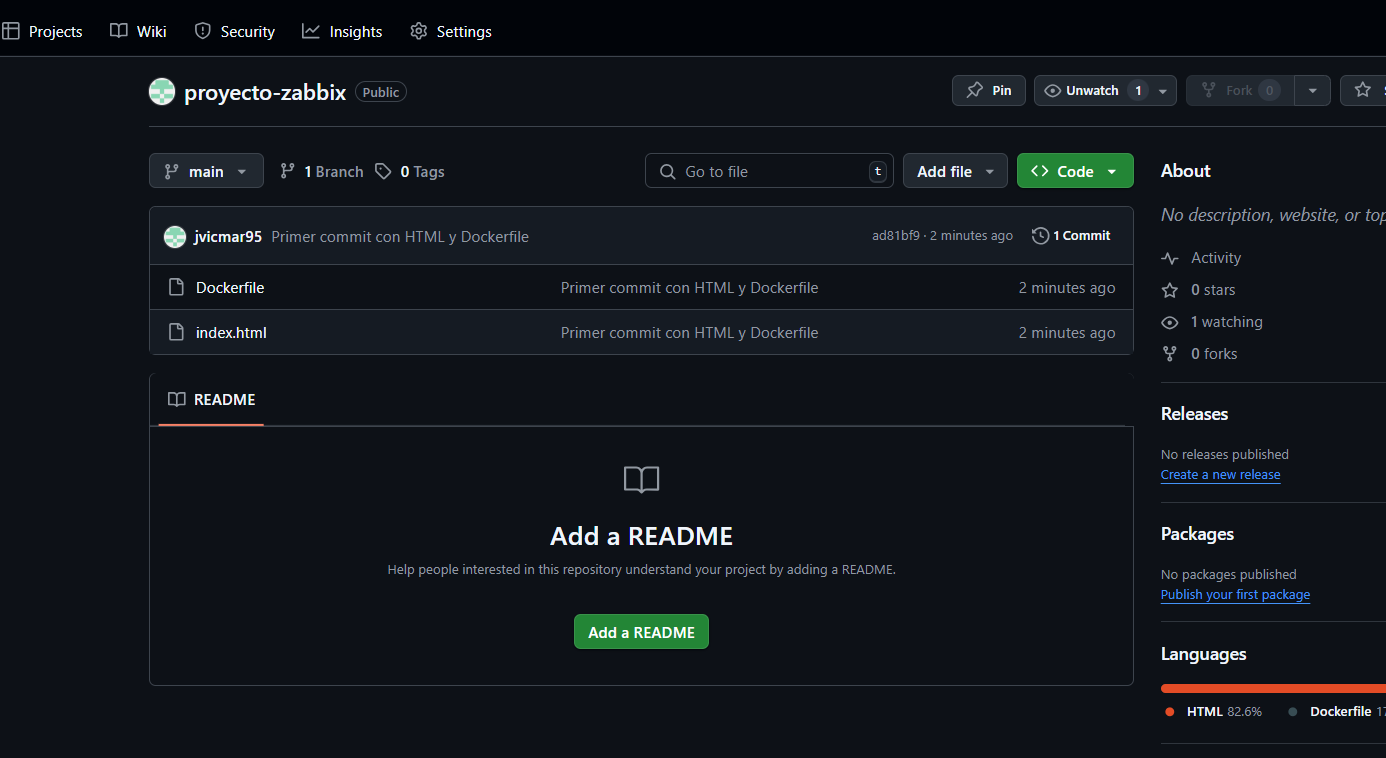
America0000001@

Crear una carpeta con un dockerfile y un html y subirlo a un repo









Cramos jenkinsfile y lo subimos

pipeline {

  agent any

  environment {

    DOCKER\_IMAGE = "jvicmar95/proyecto-zabbix:latest"

  }

  stages {

    stage('Clonar repo') {

      steps {

        checkout scm

      }

    }

    stage('Build Docker image') {

      steps {

        sh 'docker build -t $DOCKER\_IMAGE .'

      }

    }

    stage('Login Docker Hub') {

      steps {

        withCredentials([usernamePassword(credentialsId: 'docker-hub-creds', usernameVariable: 'DOCKER\_USER', passwordVariable: 'DOCKER\_PASS')]) {

          sh 'echo $DOCKER\_PASS | docker login -u $DOCKER\_USER --password-stdin'

        }

      }

    }

    stage('Push Docker image') {

      steps {

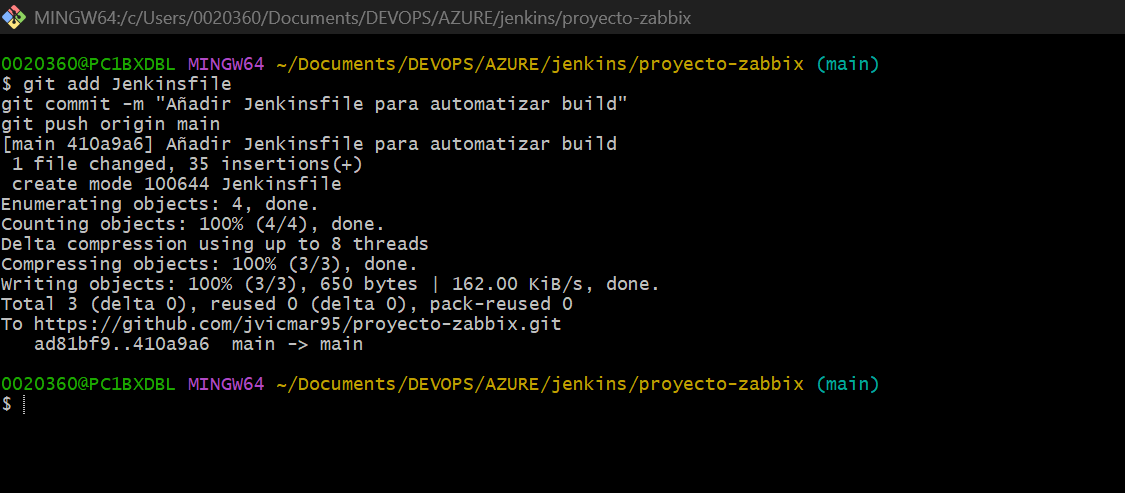
        sh 'docker push $DOCKER\_IMAGE'

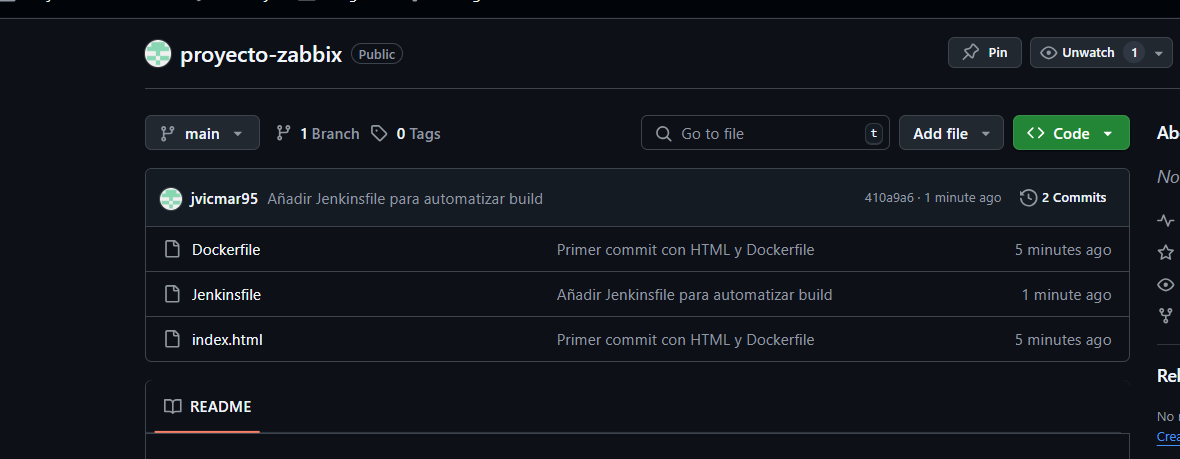
      }

    }

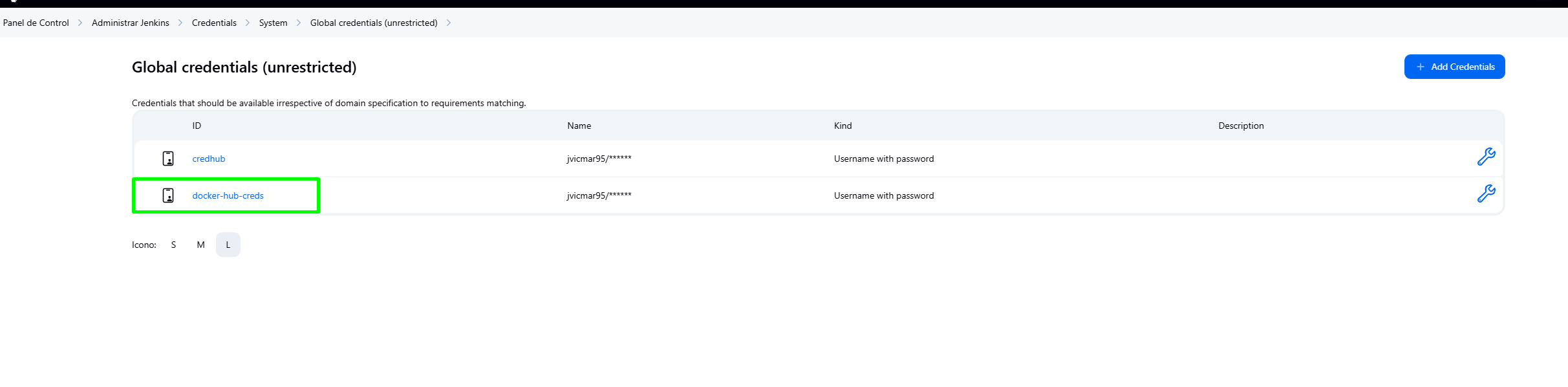
  }

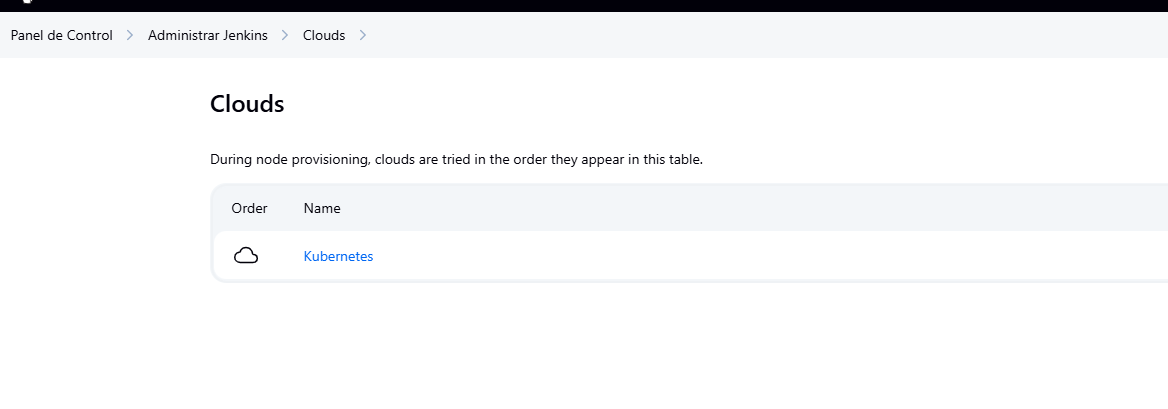
}





AHORA QUE TENGO





[Skip to content](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure#skip2content)

[2](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure)

[2](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure)

1. [Panel de Control](http://209.38.125.71:8080/)
2. [Administrar Jenkins](http://209.38.125.71:8080/manage/)
3. [Clouds](http://209.38.125.71:8080/manage/cloud/)
4. [Kubernetes](http://209.38.125.71:8080/manage/cloud/Kubernetes/)
5. Configure

[Status](http://209.38.125.71:8080/manage/cloud/Kubernetes/)

[Pod Templates](http://209.38.125.71:8080/manage/cloud/Kubernetes/templates)

[Configure](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure)

[Delete Cloud](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure)

# Cloud Kubernetes Configuration

Principio del formulario

Name[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure" \o "Help for feature: Name)



Kubernetes URL[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure)



Use Jenkins Proxy[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure)

Kubernetes server certificate key[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure" \o "Help for feature: Kubernetes server certificate key)



Disable https certificate check[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure" \o "Help for feature: Disable https certificate check)

Kubernetes Namespace



Agent Docker Registry[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure" \o "Help for feature: Agent Docker Registry)



Inject restricted PSS security context in agent container definition[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure" \o "Help for feature: Inject restricted PSS security context in agent container definition)

Credentials



Add

Test Connection

WebSocket[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure" \o "Help for feature: WebSocket)

Direct Connection[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure" \o "Help for feature: Direct Connection)

Jenkins URL[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure)



Jenkins tunnel[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure" \o "Help for feature: Jenkins tunnel)



Connection Timeout[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure" \o "Help for feature: Connection Timeout )

Read Timeout[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure" \o "Help for feature: Read Timeout )

Concurrency Limit[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure" \o "Help for feature: Concurrency Limit)

Pod Labels[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure" \o "Help for feature: Pod Labels)

**Pod Label**

Key[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure)



Value[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure" \o "Help for feature: Value)



Add Pod Label

Pod Retention

Max connections to Kubernetes API[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure)

Seconds to wait for pod to be running[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure)

Container Cleanup Timeout[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure" \o "Help for feature: Container Cleanup Timeout )

Transfer proxy related environment variables from controller to agent[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure" \o "Help for feature: Transfer proxy related environment variables from controller to agent)

Restrict pipeline support to authorized folders[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure)

Defaults Provider Template Name[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure" \o "Help for feature: Defaults Provider Template Name)



Enable garbage collection[?](http://209.38.125.71:8080/manage/cloud/Kubernetes/configure" \o "Help for feature: Enable garbage collection)

SaveApply

Final del formulario

Jenkins 2.504.1

Y LA PIPELINE

[Skip to content](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure#skip2content)

[2](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure)

[2](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure)

1. [Panel de Control](http://209.38.125.71:8080/)
2. [pipeline-proyecto-zabbix](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/)
3. Configuration

# Configure

General

Triggers

Pipeline

Advanced

Principio del formulario

## General

DisabledEnabled

Descripción



Plain text[Visualizar](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure)

Desechar ejecuciones antiguas[?](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure)

Do not allow concurrent builds

Do not allow the pipeline to resume if the controller restarts

Esta ejecución debe parametrizarse[?](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure)

Pipeline speed/durability override[?](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure" \o "Help for feature: Pipeline speed/durability override)

Preserve stashes from completed builds[?](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure" \o "Help for feature: Preserve stashes from completed builds)

Throttle builds[?](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure" \o "Help for feature: Throttle builds)

Triggers

Set up automated actions that start your build based on specific events, like code changes or scheduled times.

Construir tras otros proyectos[?](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure)

Consultar repositorio (SCM)[?](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure)

Ejecutar periódicamente[?](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure)

Pipeline

Define your Pipeline using Groovy directly or pull it from source control.

Definition



SCM[?](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure)



[?](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure)

Repositories[?](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure" \o "Help for feature: Repositories)

Repository URL[?](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure)



Credentials[?](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure" \o "Help for feature: Credentials)



Add

Avanzado

Add Repository

Branches to build[?](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure)

Branch Specifier (blank for 'any')[?](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure)



Add Branch

Navegador del repositorio[?](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure)



Additional Behaviours

Añadir

Script Path[?](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure)



Lightweight checkout[?](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/configure" \o "Help for feature: Lightweight checkout)

[Pipeline Syntax](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/pipeline-syntax)

Advanced

Avanzado

SaveApply

Final del formulario

[REST API](http://209.38.125.71:8080/job/pipeline-proyecto-zabbix/api/)Jenkins 2.504.1

# RESUMEN CREACION CREACION Y SUBIDA IMAGEN

## 🧩 Procedimiento completo: Jenkins en Kubernetes con pipeline CI/CD

### 🔐 Docker Hub

* **Usuario:** jvicmar95@gmail.com
* **Contraseña (actual):** America0000001@
* **Login desde terminal:**

bash

CopiarEditar

docker login

### 🏗️ Crear imagen personalizada de Jenkins

1. En VS Code, crea un archivo llamado Dockerfile:

Dockerfile

CopiarEditar

FROM jenkins/jenkins:lts

USER root

RUN apt-get update && \

apt-get install -y docker.io git && \

apt-get clean

RUN usermod -aG docker jenkins

USER jenkins

1. Abre terminal y ejecuta:

bash

CopiarEditar

docker build -t jvicmar95/jenkins-custom:latest .

docker push jvicmar95/jenkins-custom:latest

### ☸️ Verificar entorno Kubernetes y Helm

* Verifica clúster:

bash

CopiarEditar

kubectl get nodes

* Verifica Helm:

bash

CopiarEditar

helm version

### 📦 Instalar Jenkins con Helm

1. Añadir repositorio:

bash

CopiarEditar

helm repo add jenkins https://charts.jenkins.io

helm repo update

1. Crear archivo jenkins-values.yaml en VS Code:

yaml

CopiarEditar

controller:

image:

repository: jvicmar95/jenkins-custom

tag: latest

pullPolicy: IfNotPresent

installPlugins:

- kubernetes:latest

- workflow-aggregator:latest

- git:latest

- docker-workflow:latest

serviceType: LoadBalancer

admin:

username: admin

password: admin

resources:

requests:

cpu: "500m"

memory: "1Gi"

limits:

cpu: "1000m"

memory: "2Gi"

persistence:

enabled: true

size: 8Gi

storageClass: do-block-storage

1. Ejecutar instalación:

bash

CopiarEditar

helm install jenkins jenkins/jenkins -n jenkins --create-namespace -f jenkins-values.yaml

1. Ver IP pública:

bash

CopiarEditar

kubectl get svc -n jenkins

Ejemplo:

pgsql

CopiarEditar

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

jenkins LoadBalancer 10.109.11.69 139.59.49.158 8080:30173/TCP 2m37s

### 🧪 Verificar imagen desplegada en Pod

bash

CopiarEditar

kubectl get pods -n jenkins -o jsonpath="{.items[0].spec.containers[0].image}"

Resultado esperado:  
docker.io/jvicmar95/jenkins-custom:latest

### 🐙 Repositorio GitHub

* Usuario: jvicmar95
* Crear un repositorio y añadir:
  + Dockerfile
  + index.html
  + Jenkinsfile

### 📜 Jenkinsfile (para pipeline)

Crear archivo Jenkinsfile en VS Code:

groovy

CopiarEditar

pipeline {

agent any

environment {

DOCKER\_IMAGE = "jvicmar95/proyecto-zabbix:latest"

}

stages {

stage('Clonar repo') {

steps {

checkout scm

}

}

stage('Build Docker image') {

steps {

sh 'docker build -t $DOCKER\_IMAGE .'

}

}

stage('Login Docker Hub') {

steps {

withCredentials([usernamePassword(credentialsId: 'docker-hub-creds', usernameVariable: 'DOCKER\_USER', passwordVariable: 'DOCKER\_PASS')]) {

sh 'echo $DOCKER\_PASS | docker login -u $DOCKER\_USER --password-stdin'

}

}

}

stage('Push Docker image') {

steps {

sh 'docker push $DOCKER\_IMAGE'

}

}

}

}

### ⚙️ Configuración en Jenkins UI

Una vez dentro de Jenkins:

1. Ir a: Administrar Jenkins → Clouds → Kubernetes
2. Configura:
   * **Kubernetes URL**
   * **Namespace**: jenkins
   * **Docker registry** (si usas Docker-in-Docker o sidecar)
   * **Credenciales**: Añade docker-hub-creds en Jenkins
3. Crea nuevo pipeline pipeline-proyecto-zabbix con:
   * Definición: Pipeline desde SCM
   * Repositorio: GitHub
   * Ramas: main
   * Script path: Jenkinsfile

### ✅ Resultado final

Al ejecutar el pipeline:

* Jenkins clona el repo de GitHub
* Construye una imagen Docker con el contenido del repositorio
* Hace login en Docker Hub
* Sube la imagen como jvicmar95/proyecto-zabbix:latest

# EMPIEZO DEPLOY

## 🚀 Procedimiento: Despliegue automático en Kubernetes desde Jenkins

### 🎯 Objetivo:

Desplegar una imagen Docker ya subida a Docker Hub (jvicmar95/proyecto-zabbix:latest) a tu clúster Kubernetes automáticamente desde una pipeline en Jenkins.

### 1️⃣ Añadir manifiesto de Kubernetes al repo

**Archivo:** deployment.yaml

yaml

CopiarEditar

apiVersion: apps/v1

kind: Deployment

metadata:

name: web-nginx

labels:

app: web-nginx

spec:

replicas: 1

selector:

matchLabels:

app: web-nginx

template:

metadata:

labels:

app: web-nginx

spec:

containers:

- name: nginx

image: jvicmar95/proyecto-zabbix:latest

ports:

- containerPort: 80

---

apiVersion: v1

kind: Service

metadata:

name: web-nginx

spec:

selector:

app: web-nginx

ports:

- protocol: TCP

port: 80

targetPort: 80

type: LoadBalancer

🗂 Este archivo debe estar en el **mismo repositorio** que el Jenkinsfile.

### 2️⃣ Añadir contenedor kubectl en el Jenkinsfile

Dentro del bloque yaml """ del Jenkinsfile, añade:

yaml

CopiarEditar

- name: kubectl

image: lachlanevenson/k8s-kubectl:latest

command:

- cat

tty: true

volumeMounts:

- mountPath: /home/jenkins/agent

name: workspace-volume

### 3️⃣ Crear stage de despliegue en el Jenkinsfile

Después del push de la imagen, añade:

groovy

CopiarEditar

stage('Deploy to Kubernetes') {

steps {

container('kubectl') {

sh 'kubectl apply -f deployment.yaml'

}

}

}

### 4️⃣ Dar permisos al ServiceAccount de Jenkins

**Una sola vez desde terminal**:

yaml

CopiarEditar

# jenkins-rbac.yaml

apiVersion: rbac.authorization.k8s.io/v1

kind: Role

metadata:

name: jenkins-role

namespace: jenkins

rules:

- apiGroups: ["apps"]

resources: ["deployments"]

verbs: ["get", "list", "create", "update", "delete"]

- apiGroups: [""]

resources: ["services"]

verbs: ["get", "list", "create", "update", "delete"]

---

apiVersion: rbac.authorization.k8s.io/v1

kind: RoleBinding

metadata:

name: jenkins-rolebinding

namespace: jenkins

subjects:

- kind: ServiceAccount

name: default

namespace: jenkins

roleRef:

kind: Role

name: jenkins-role

apiGroup: rbac.authorization.k8s.io

bash

CopiarEditar

kubectl apply -f jenkins-rbac.yaml

### 5️⃣ Acceder a la aplicación desplegada

bash

CopiarEditar

kubectl get svc web-nginx -n jenkins

* Si tiene EXTERNAL-IP, accede en navegador:  
  http://<external-ip>
* Si está en <pending>:

bash

CopiarEditar

kubectl port-forward svc/web-nginx 8080:80 -n jenkins

Accede en navegador a:  
http://localhost:8080

JENKINSFILE FINAL

pipeline {

  agent {

    kubernetes {

      label 'dind-agent'

      defaultContainer 'dind'

      yaml """

apiVersion: v1

kind: Pod

spec:

  containers:

  - name: dind

    image: docker:20.10-dind

    securityContext:

      privileged: true

    args:

    - dockerd

    - --host=tcp://127.0.0.1:2375

    - --host=unix:///var/run/docker.sock

    - --tls=false

    ports:

    - containerPort: 2375

    volumeMounts:

    - mountPath: /var/lib/docker

      name: docker-graph-storage

    - mountPath: /home/jenkins/agent

      name: workspace-volume

  - name: kubectl

    image: lachlanevenson/k8s-kubectl:latest

    command:

    - cat

    tty: true

    volumeMounts:

    - mountPath: /home/jenkins/agent

      name: workspace-volume

  - name: jnlp

    image: jenkins/inbound-agent:3309.v27b\_9314fd1a\_4-1

    resources:

      requests:

        memory: "256Mi"

        cpu: "100m"

    env:

    - name: JENKINS\_AGENT\_WORKDIR

      value: /home/jenkins/agent

    volumeMounts:

    - mountPath: /home/jenkins/agent

      name: workspace-volume

  nodeSelector:

    kubernetes.io/os: linux

  restartPolicy: Never

  volumes:

  - name: docker-graph-storage

    emptyDir: {}

  - name: workspace-volume

    emptyDir: {}

"""

    }

  }

  environment {

    DOCKER\_HOST = "tcp://127.0.0.1:2375"

    DOCKER\_IMAGE = "jvicmar95/proyecto-zabbix:latest"

  }

  stages {

    stage('Esperar Docker') {

      steps {

        sh 'sleep 20'

      }

    }

    stage('Build Docker image') {

      steps {

        sh 'docker version'

        sh 'docker build -t $DOCKER\_IMAGE .'

      }

    }

    stage('Push Docker image') {

      steps {

        withCredentials([usernamePassword(credentialsId: 'docker-hub-creds', usernameVariable: 'DOCKER\_USER', passwordVariable: 'DOCKER\_PASS')]) {

          sh 'echo $DOCKER\_PASS | docker login -u $DOCKER\_USER --password-stdin'

          sh 'docker push $DOCKER\_IMAGE'

        }

      }

    }

    stage('Deploy to Kubernetes') {

      steps {

        container('kubectl') {

          sh 'ls -la'

          sh 'kubectl version --client'

          sh 'kubectl apply -f deployment.yaml'

        }

      }

    }

  }

}

JENKINS-RBAC.YAML (OSLO EJECUTARLO CON UN KUBECTL APPLY)

apiVersion: rbac.authorization.k8s.io/v1

kind: Role

metadata:

  name: jenkins-role

  namespace: jenkins

rules:

- apiGroups: ["apps"]

  resources: ["deployments"]

  verbs: ["get", "list", "watch", "create", "update", "patch", "delete"]

- apiGroups: [""]

  resources: ["services"]

  verbs: ["get", "list", "watch", "create", "update", "patch", "delete"]

---

apiVersion: rbac.authorization.k8s.io/v1

kind: RoleBinding

metadata:

  name: jenkins-rolebinding

  namespace: jenkins

subjects:

- kind: ServiceAccount

  name: default

  namespace: jenkins

roleRef:

  kind: Role

  name: jenkins-role

  apiGroup: rbac.authorization.k8s.io

DEPLOYMENT.YAML

apiVersion: apps/v1

kind: Deployment

metadata:

  name: web-nginx

  labels:

    app: web-nginx

spec:

  replicas: 1

  selector:

    matchLabels:

      app: web-nginx

  template:

    metadata:

      labels:

        app: web-nginx

    spec:

      containers:

        - name: nginx

          image: jvicmar95/proyecto-zabbix:latest

          ports:

            - containerPort: 80

---

apiVersion: v1

kind: Service

metadata:

  name: web-nginx

spec:

  selector:

    app: web-nginx

  ports:

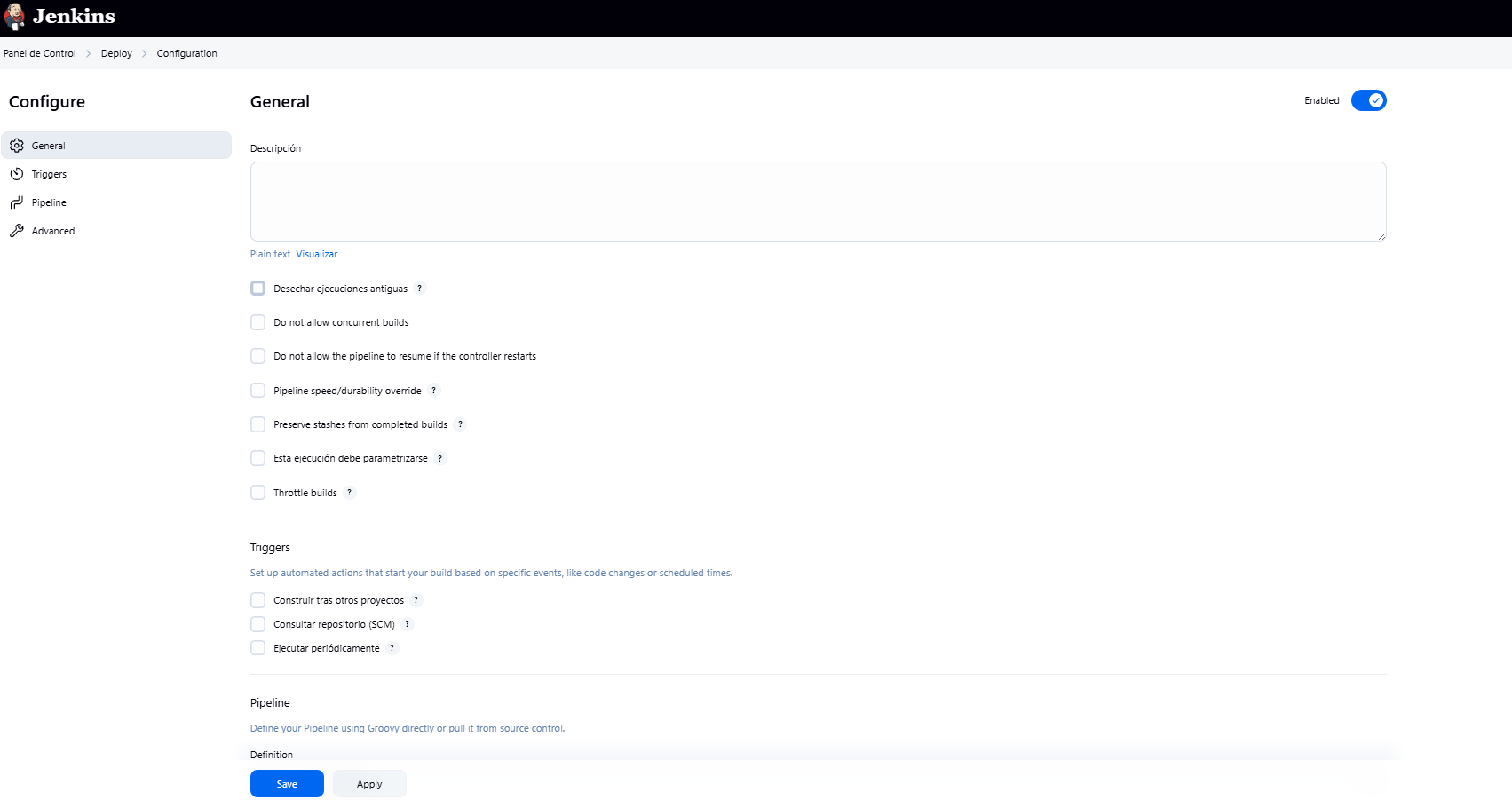
    - protocol: TCP

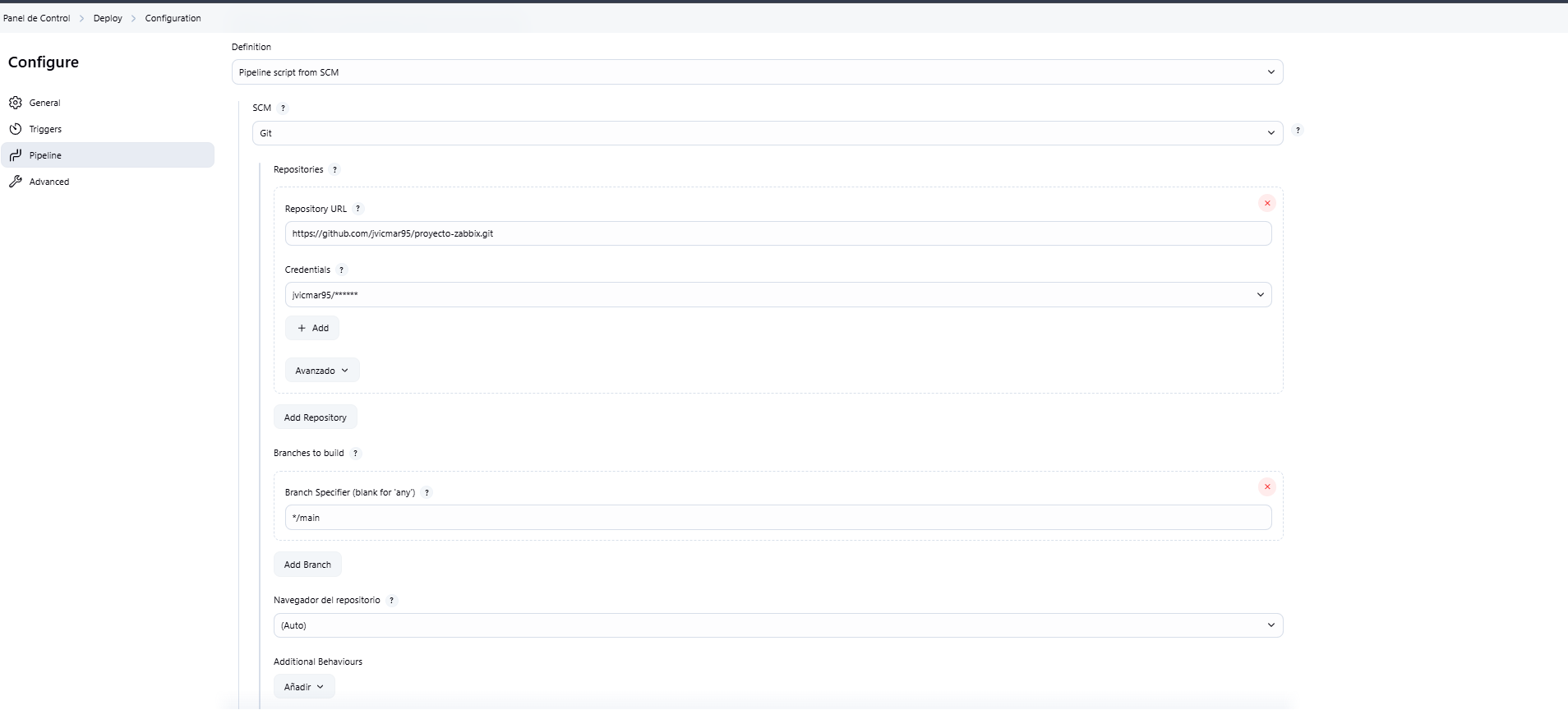
      port: 80

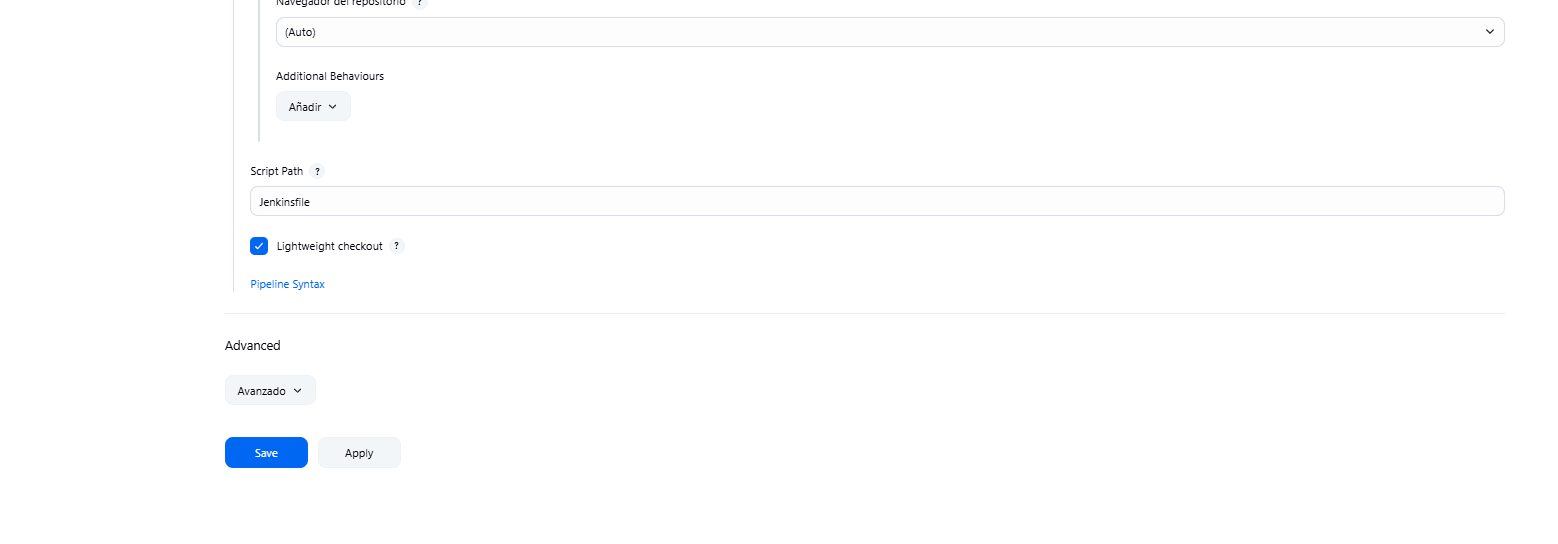
      targetPort: 80

  type: LoadBalancer # o ClusterIP si harás port-forward

pipeline







Cloud

