# FICHEROS

## Para Jenkins

### 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: ClusterIP

  admin:

    username: admin

    password: admin

  persistence:

    enabled: true

    size: 8Gi

    storageClass: do-block-storage-retain

  resources:

    requests:

      cpu: "100m"

      memory: "512Mi"

    limits:

      cpu: "500m"

      memory: "1Gi"

### 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

## Para Pipelines

### Jenkinsfile

properties([

  parameters([

    string(name: 'APP\_VERSION', defaultValue: '1.0', description: 'Introduce la versión a desplegar (ej. 1.0, 2.0, etc.)')

  ])

])

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:${APP\_VERSION}"

  }

  stages {

    stage('⏳ Esperar Docker') {

      steps {

        sh 'echo "⏱ Esperando que Docker esté disponible..."'

        sh 'sleep 20'

      }

    }

    stage('🐳 Build Docker image') {

      steps {

        sh 'echo "🔍 Verificando Docker..." && docker version'

        sh 'echo "🏗️ Construyendo imagen $DOCKER\_IMAGE..." && docker build -t $DOCKER\_IMAGE .'

      }

    }

    stage('📤 Push Docker image') {

      steps {

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

          sh 'echo "🔐 Login en Docker Hub..." && echo $DOCKER\_PASS | docker login -u $DOCKER\_USER --password-stdin'

          sh 'echo "📦 Subiendo imagen a Docker Hub..." && docker push $DOCKER\_IMAGE'

        }

      }

    }

    stage('🚀 Deploy to Kubernetes') {

      steps {

        container('kubectl') {

          sh 'echo "📂 Listando archivos..." && ls -la'

          sh 'echo "🔧 Verificando kubectl..." && kubectl version --client'

          sh 'echo "🚀 Aplicando deployment.yaml..." && kubectl apply -f deployment.yaml'

          sh 'echo "🔁 Actualizando imagen del deployment..." && kubectl set image deployment/web-nginx nginx=$DOCKER\_IMAGE -n jenkins'

          sh 'echo "♻️ Borrando pod antiguo (si existe)..." && kubectl delete pod -l app=web-nginx -n jenkins || true'

          sh 'echo "⌛ Esperando nuevo pod..." && sleep 10'

          sh 'echo "📦 Nuevo pod desplegado:" && kubectl get pods -l app=web-nginx -n jenkins'

          sh 'echo "🌍 Servicio expuesto:" && kubectl get svc web-nginx -n jenkins'

        }

      }

    }

  }

}

### 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", "watch", "create", "update", "patch", "delete"]

- apiGroups: [""]

  resources: ["services", "pods"]   # ✅ Añadimos "pods"

  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

          imagePullPolicy: Always

          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

### Dockerfile

FROM nginx:alpine

# Elimina la página por defecto de nginx que interfiere

RUN rm -rf /usr/share/nginx/html/\*

# Copia todo el contenido del sitio

COPY web/ /usr/share/nginx/html/

# Asegura los tipos MIME y rutas válidas

COPY nginx.conf /etc/nginx/nginx.conf

# REQUISITOS

## Crear clúster Kubernetes

$env:KUBECONFIG = "C:\Users\0020360\.kube\kubeconfig.yaml"

## Cuenta DockerHub

jvicmar95 / America\*\*\*\*\*1@

## Cuenta Github

jvicmar95 / America\*\*\*\*\*1@

# Aplicar ServiceAccount

# 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

# Desplegar Jenkins

Para ello crearemos una imagen custom de base oficial Jenkins pero instalando Docker y Git.

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

Para crear la imagen utilizamos nuestro WSL

C:\Users\0020360>**wsl --list --verbose**

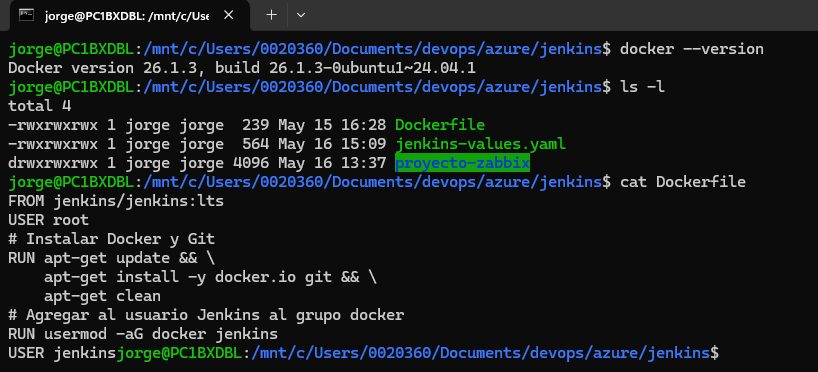
NAME STATE VERSION

\* docker-desktop Stopped 2

Ubuntu Stopped 2

C:\Users\0020360>**wsl -d Ubuntu**

jorge@PC1BXDBL:/mnt/c/Users/0020360$



jorge@PC1BXDBL:/mnt/c/Users/0020360/Documents/devops/azure/jenkins$ **docker --version**

Docker version 26.1.3, build 26.1.3-0ubuntu1~24.04.1

jorge@PC1BXDBL:/mnt/c/Users/0020360/Documents/devops/azure/jenkins$ **ls -l**

total 4

-rwxrwxrwx 1 jorge jorge 239 May 15 16:28 Dockerfile

-rwxrwxrwx 1 jorge jorge 564 May 16 15:09 jenkins-values.yaml

drwxrwxrwx 1 jorge jorge 4096 May 16 13:37 proyecto-zabbix

jorge@PC1BXDBL:/mnt/c/Users/0020360/Documents/devops/azure/jenkins$ **cat Dockerfile**

FROM jenkins/jenkins:lts

USER root

# Instalar Docker y Git

RUN apt-get update && \

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

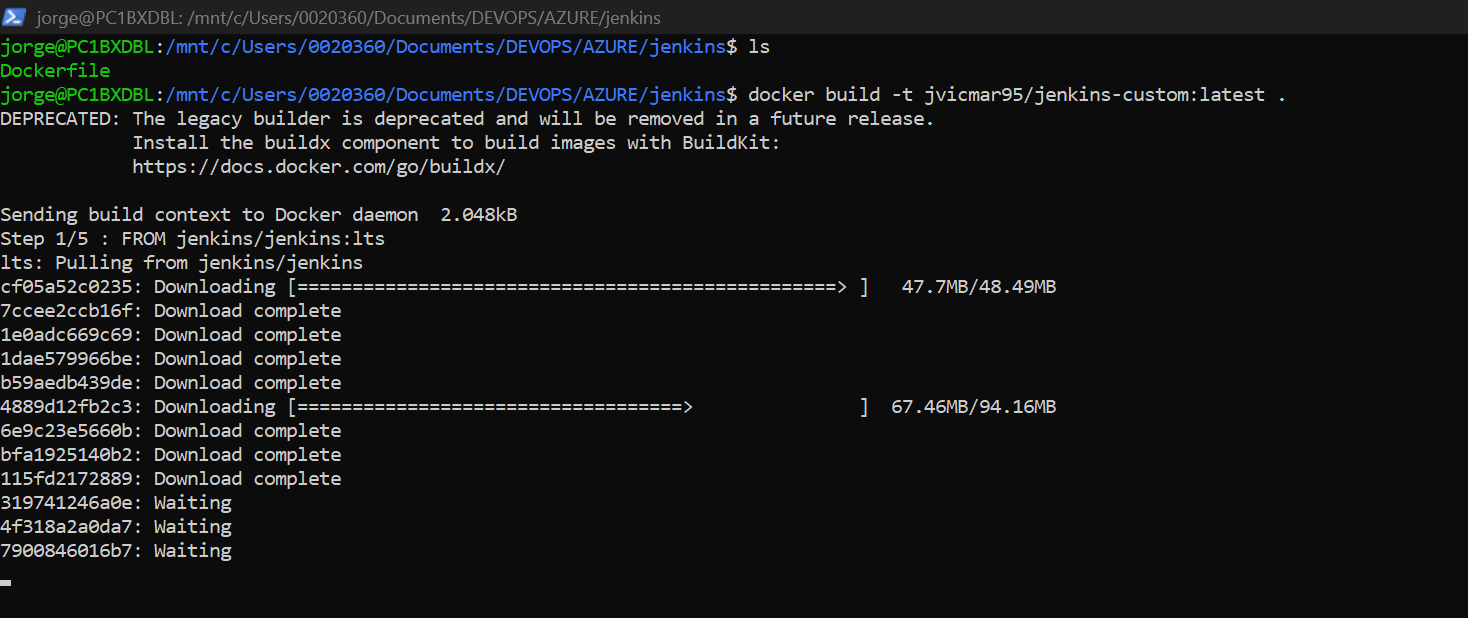
apt-get clean

# Agregar al usuario Jenkins al grupo docker

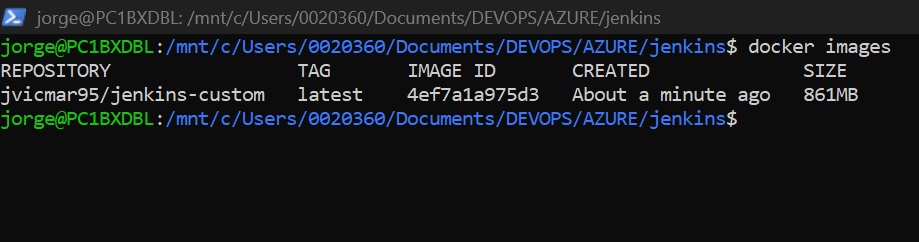
RUN usermod -aG docker jenkins

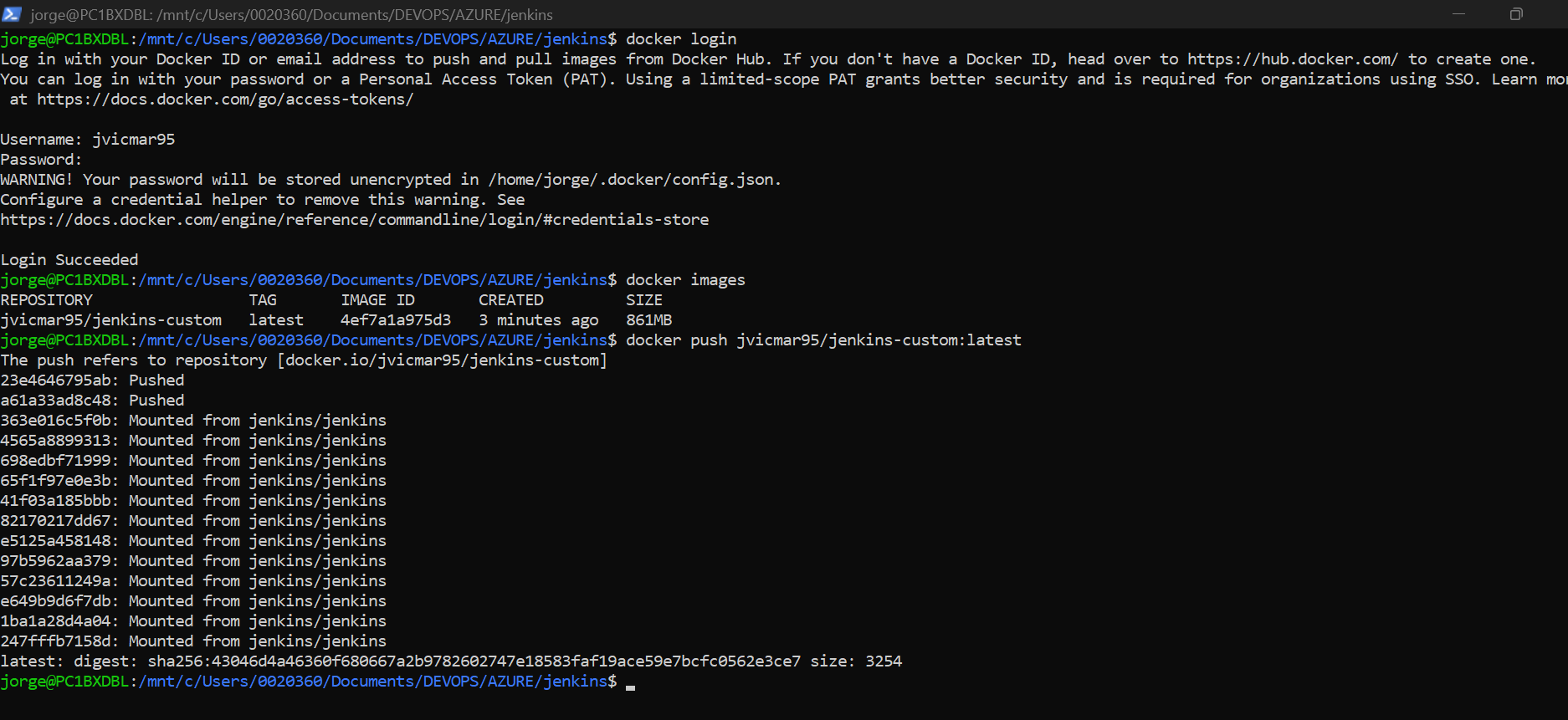
USER jenkinsjorge@PC1BXDBL:/mnt/c/Users/0020360/Documents/devops/azure/jenkins$

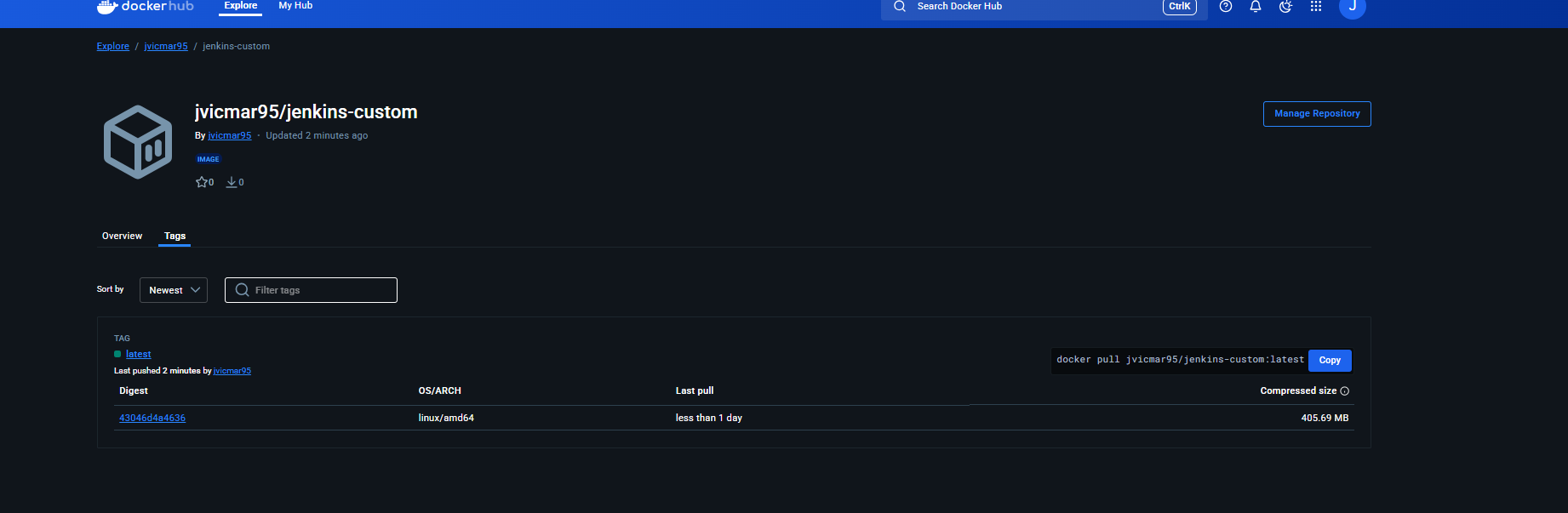
Hacemos Build de la imagen con **docker build -t jvicmar95/jenkins-custom:latest .**



Y hacemos un Push al repositorio haciendo primero un Docker Login.







Ahora ya podemos desplegar esta imagen custom de nuestro Jenkins.

* Añadimos repositorio

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

* Creamos nuestro 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: ClusterIP

  admin:

    username: admin

    password: admin

  persistence:

    enabled: false

  resources:

    requests:

      cpu: "100m"

      memory: "512Mi"

    limits:

      cpu: "500m"

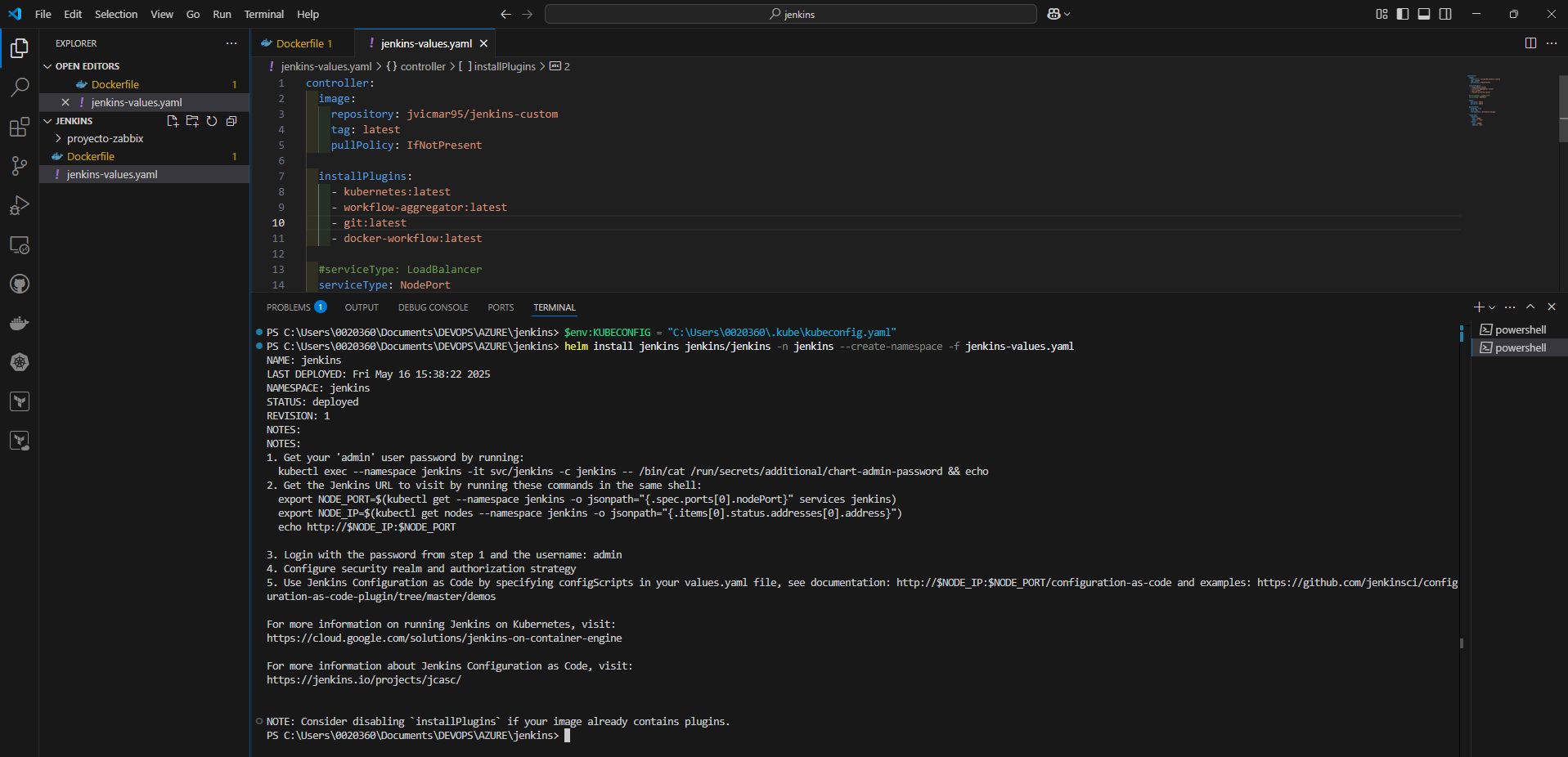
      memory: "1Gi"

* Upgradeamos los repositorios

helm repo update

* Instalamos Jenkins

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



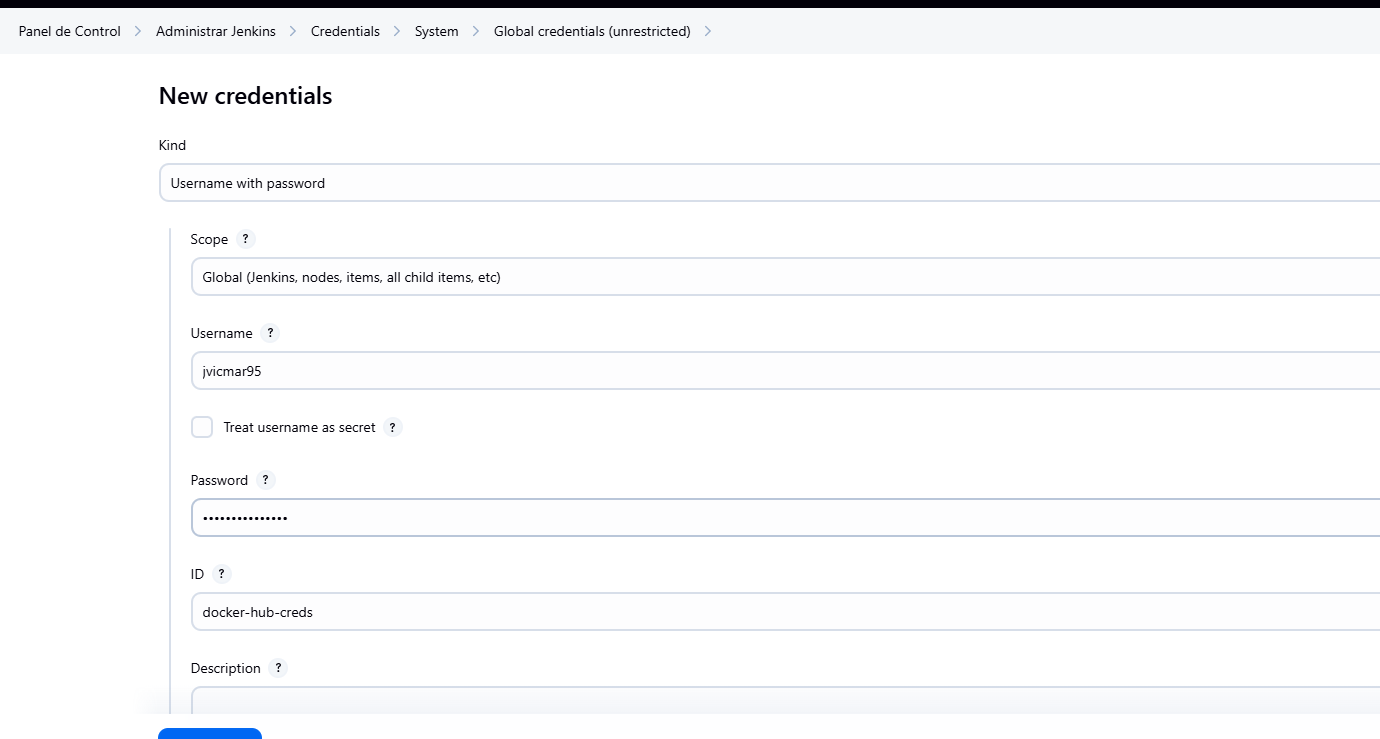
* **Conectarnos al frontal de Jenkins**

**kubectl port-forward svc/jenkins 8080:8080 -n jenkins**

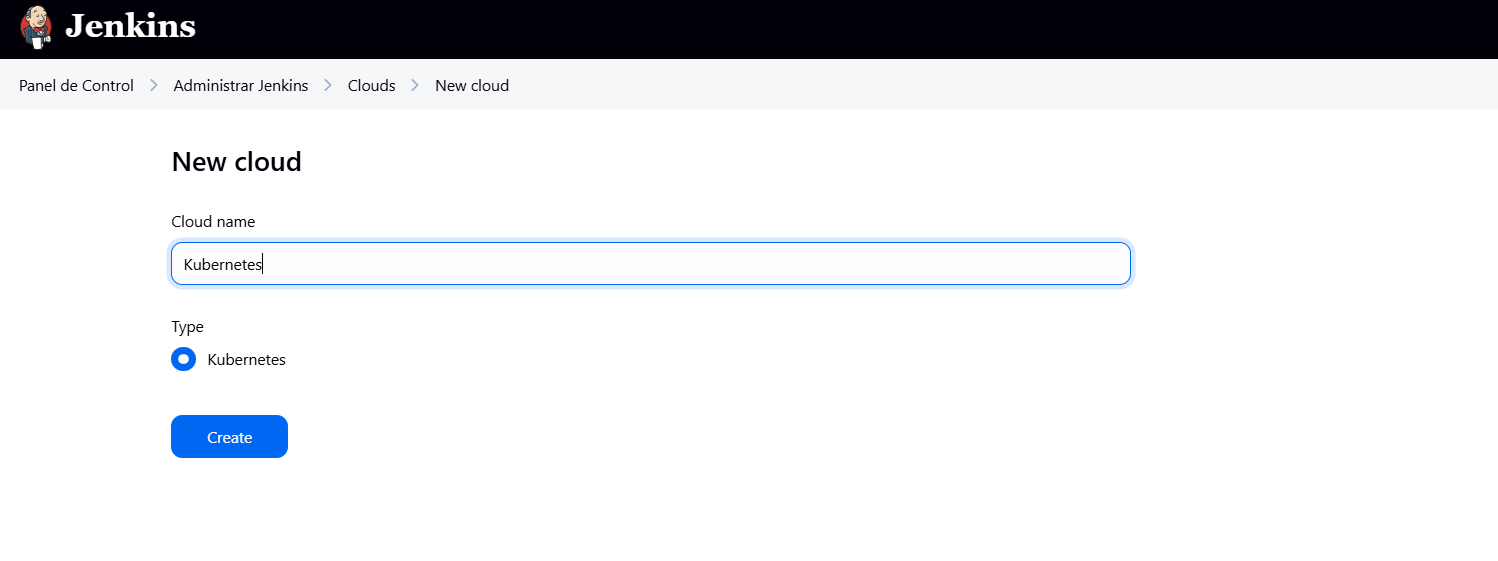
* **Accederemos con** [**http://localhost:8080/**](http://localhost:8080/)

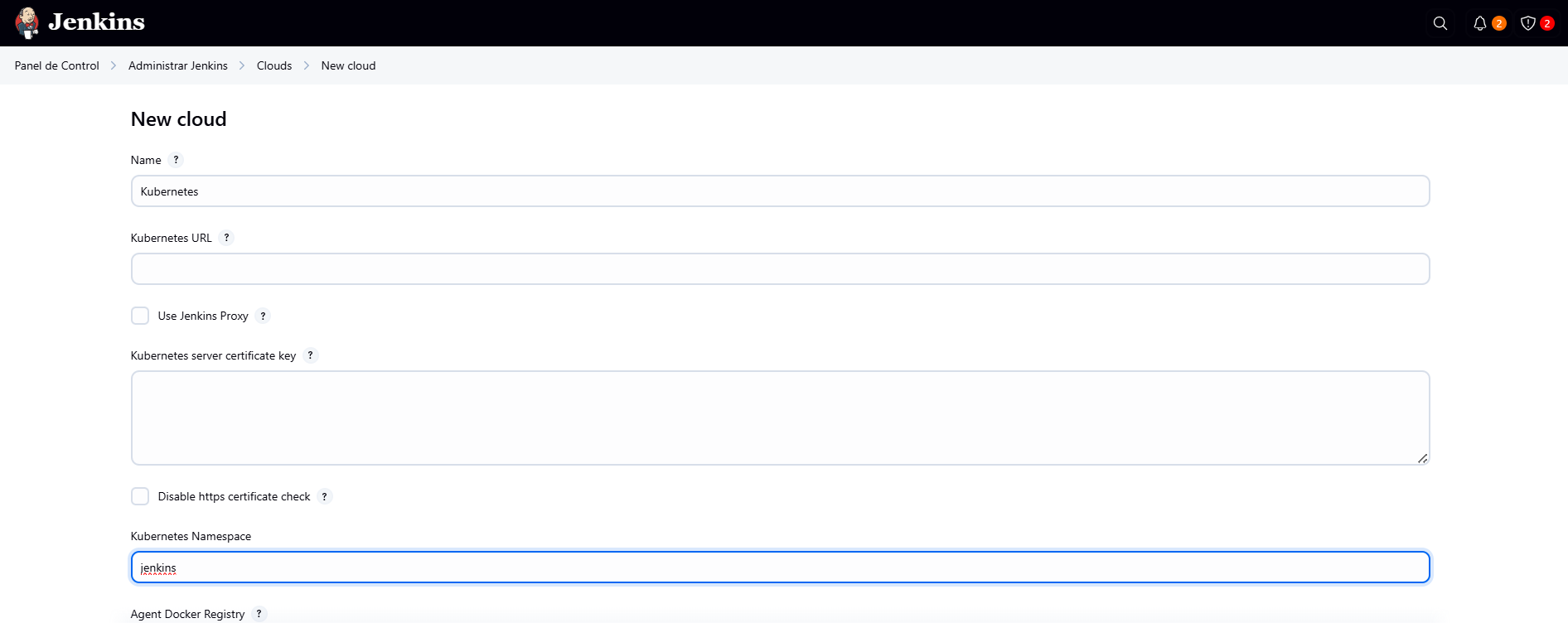
# Configurar Jenkins

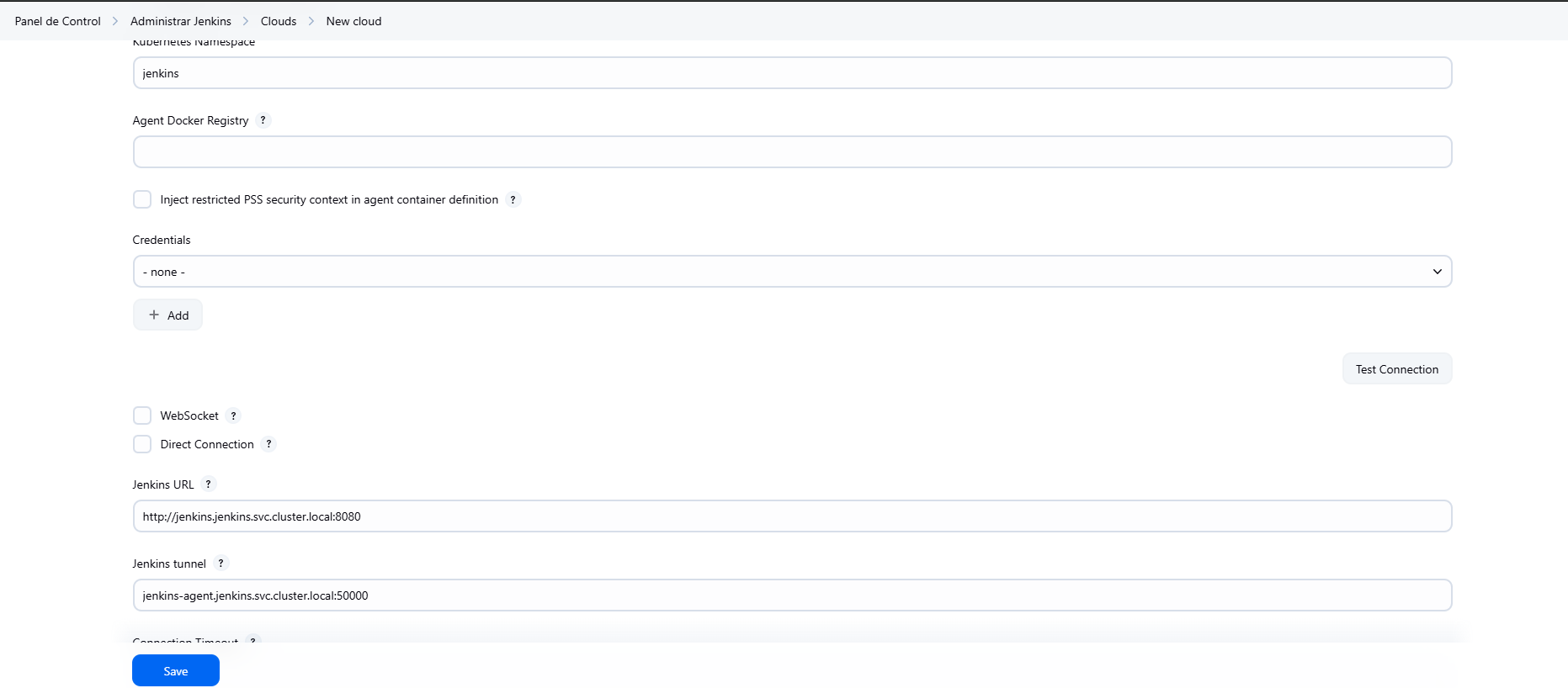
## Credenciales DockerHub



## Cloud

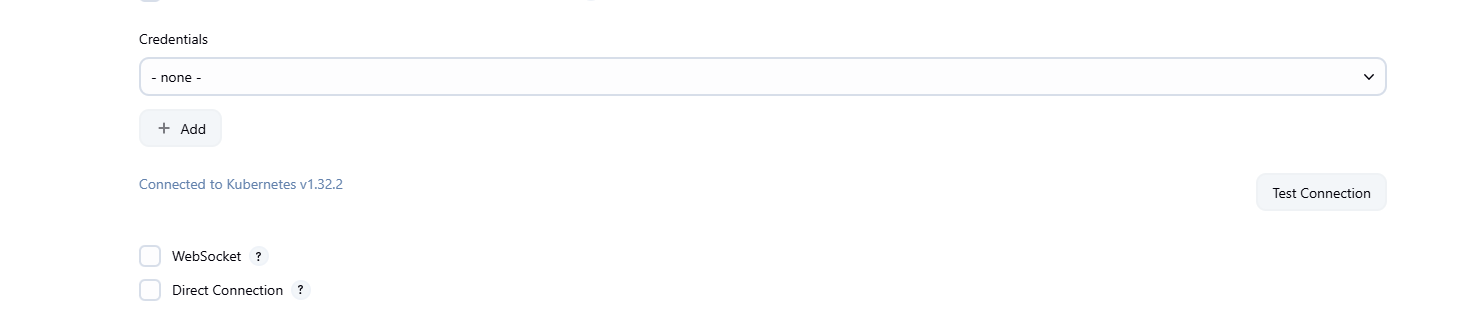




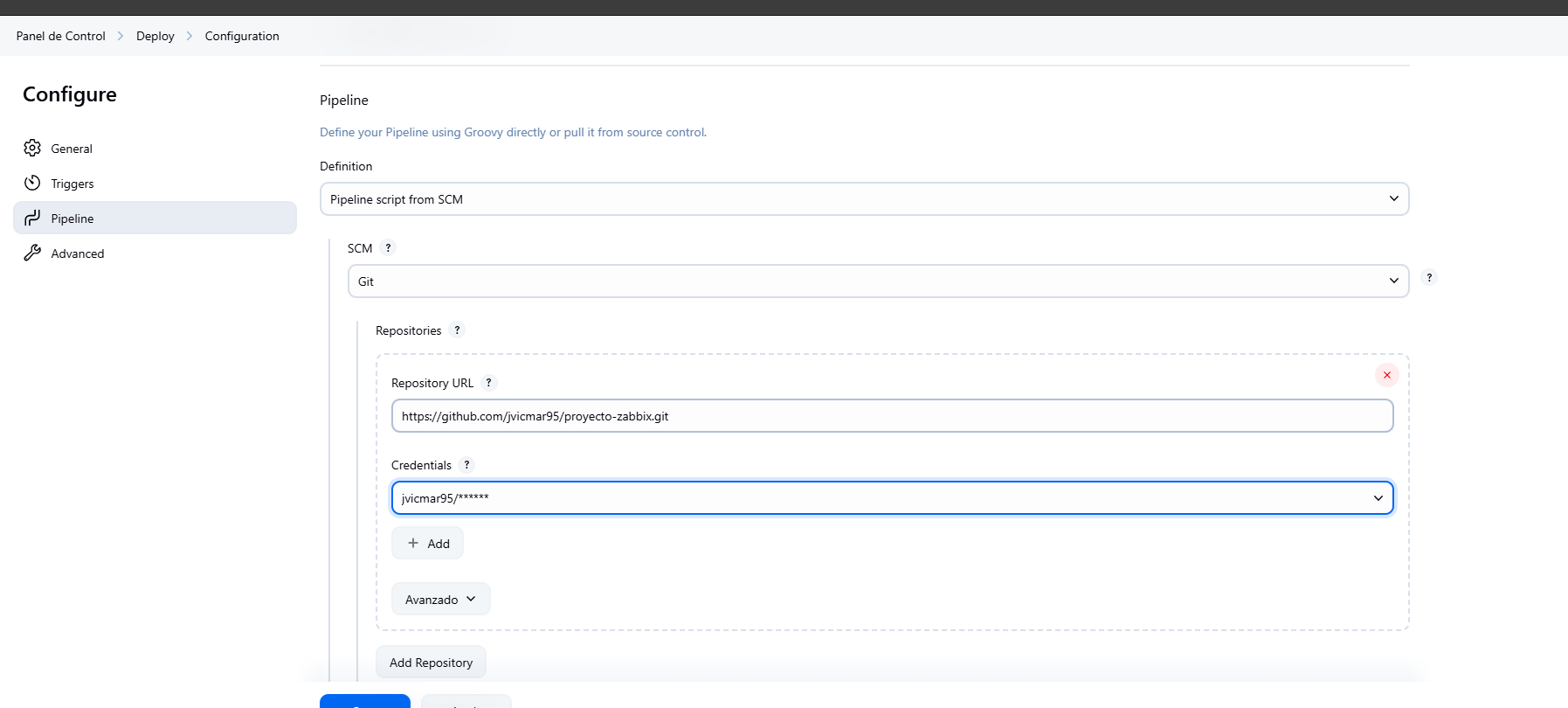


Jenkins URL: http://jenkins.jenkins.svc.cluster.local:8080

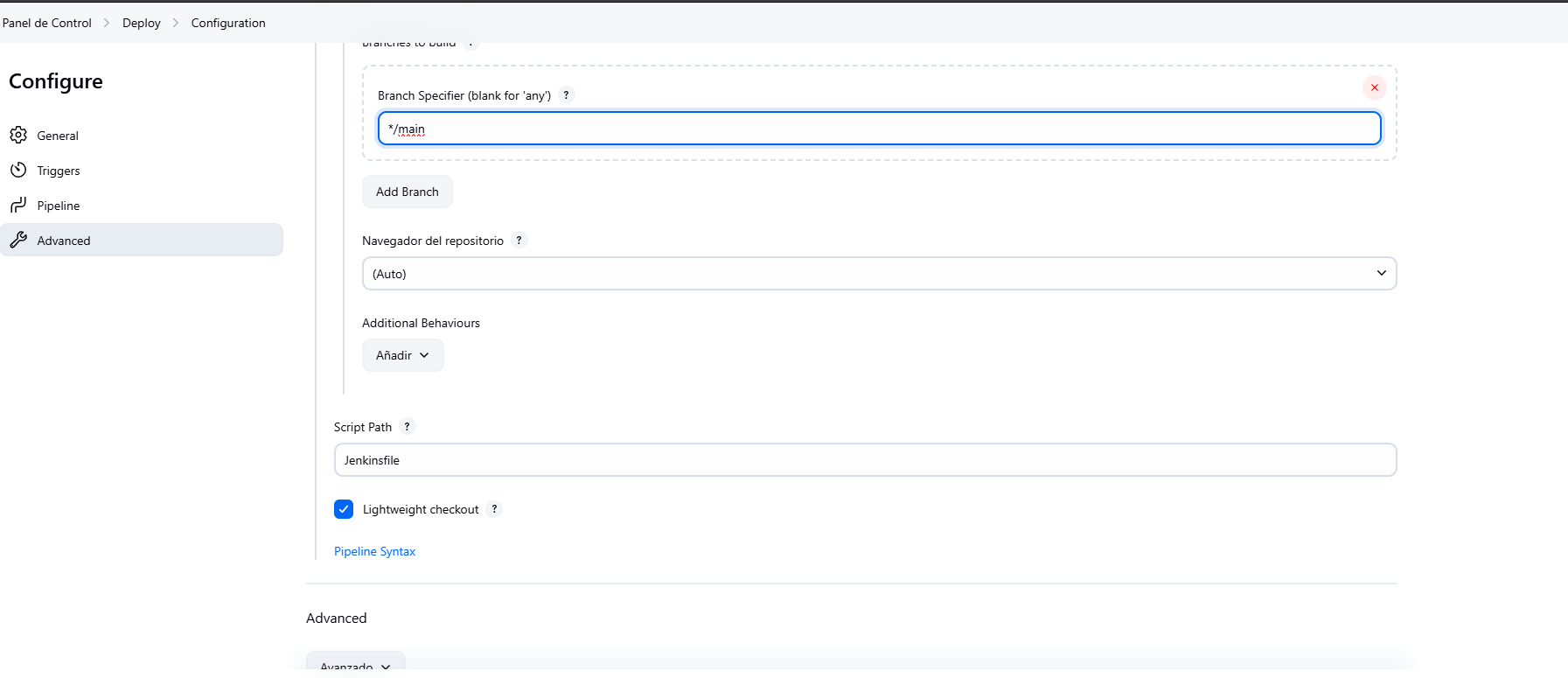
Jenkins tunnel: jenkins-agent.jenkins.svc.cluster.local:50000



# Crear Pipeline



https://github.com/jvicmar95/proyecto-zabbix.git



# Acceder

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

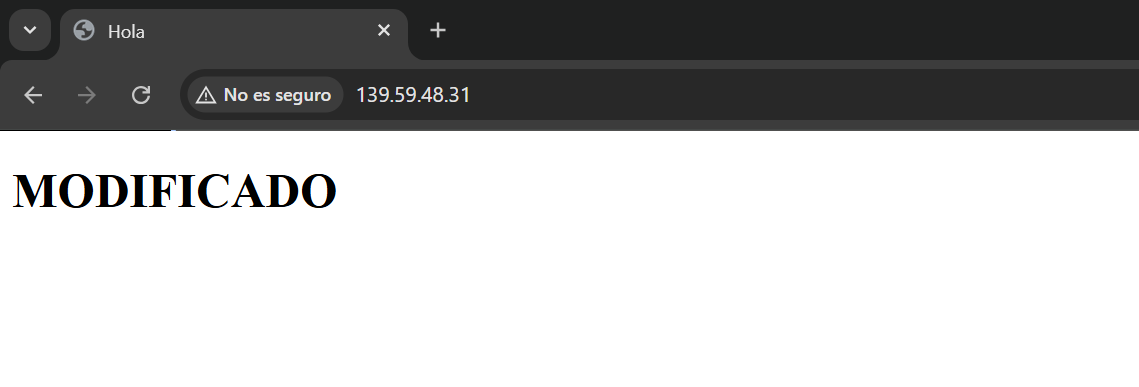
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

jenkins ClusterIP 10.108.74.142 <none> 8080/TCP 13m

jenkins-agent ClusterIP 10.108.71.53 <none> 50000/TCP 13m

web-nginx LoadBalancer 10.108.91.169 139.59.48.31,2400:6180:100:d0::7e72:a001 80:30403/TCP 4m42s

PS C:\Users\0020360\Documents\DEVOPS\AZURE\jenkins\proyecto-zabbix>

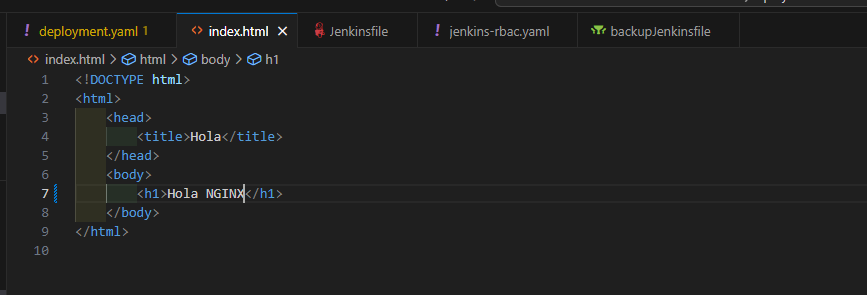


Para ver que imagen esta desplegada

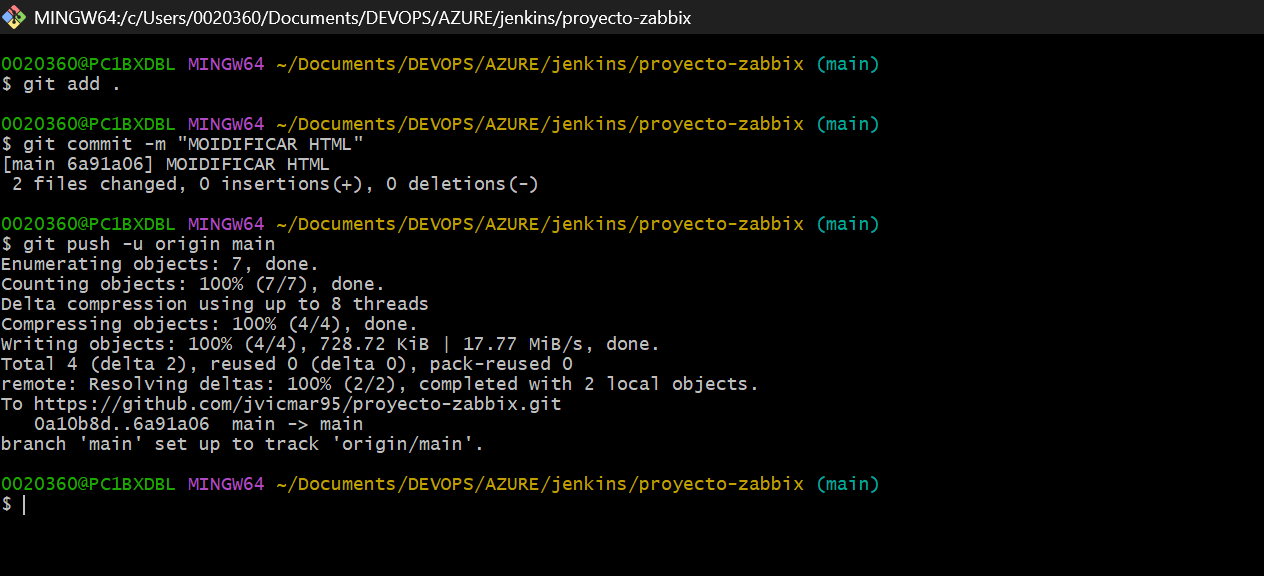
kubectl get deployment web-nginx -n jenkins -o=jsonpath="{.spec.template.spec.containers[\*].image}"

# Modificar código y desplegar

Modificamos nuestros ficheros



Lo subimos a Dockerhub



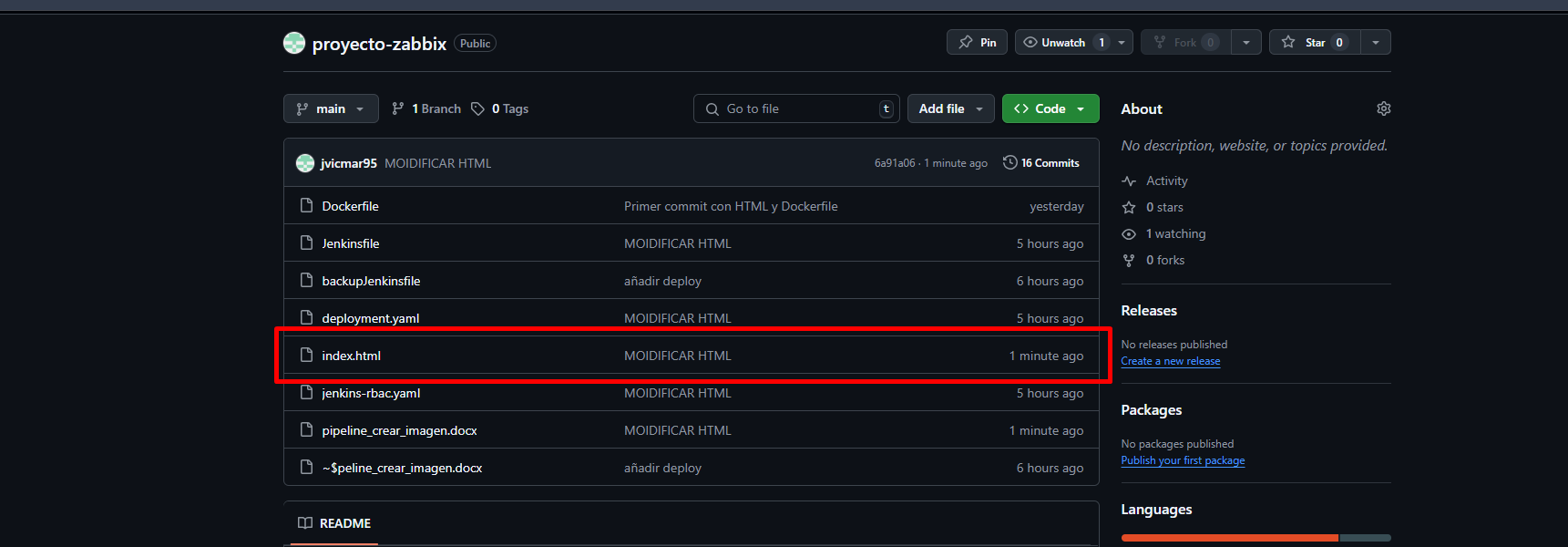
0020360@PC1BXDBL MINGW64 ~/Documents/DEVOPS/AZURE/jenkins/proyecto-zabbix (main)

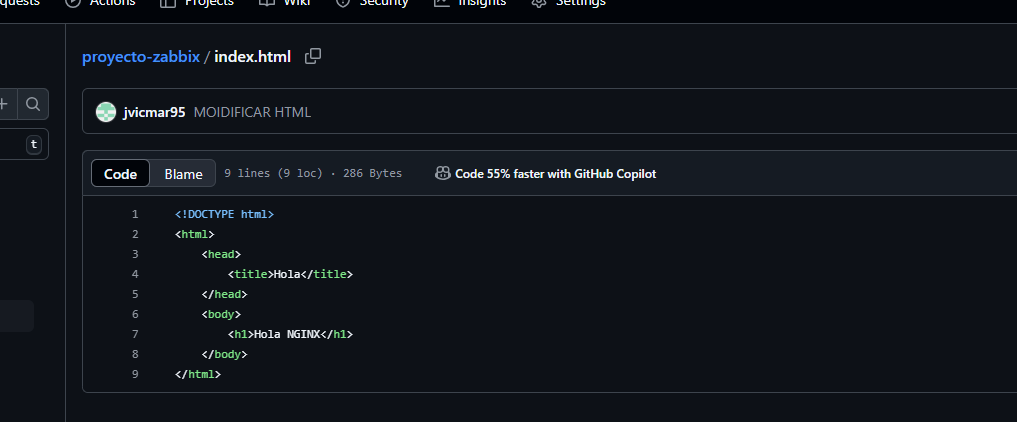
git add .

git commit -m "MOIDIFICAR HTML"

git push -u origin main

Ver que ha subido el código





Desplegamos con la Pipeline y comprobar

