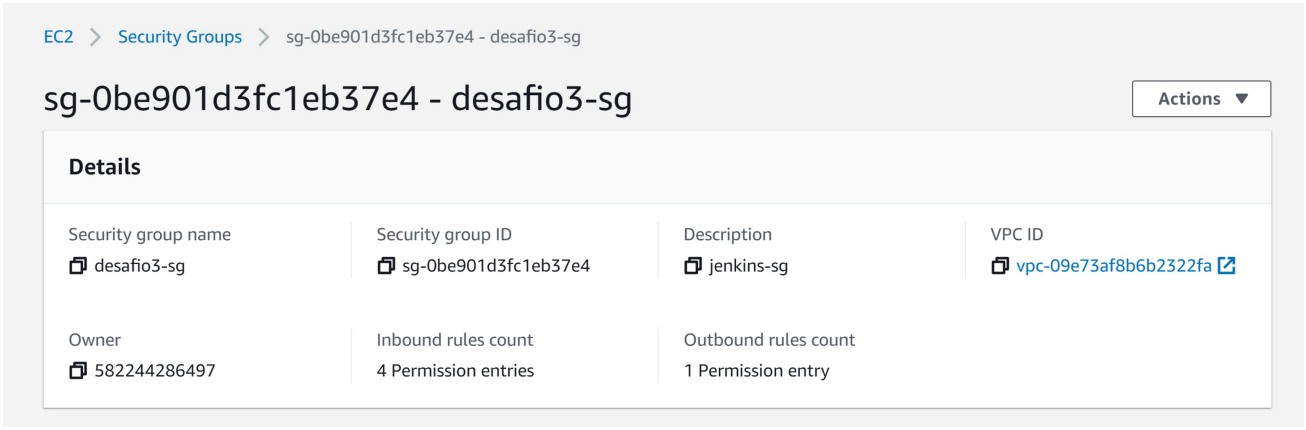
b) Automatizacion del desafio 3 con Jenkins

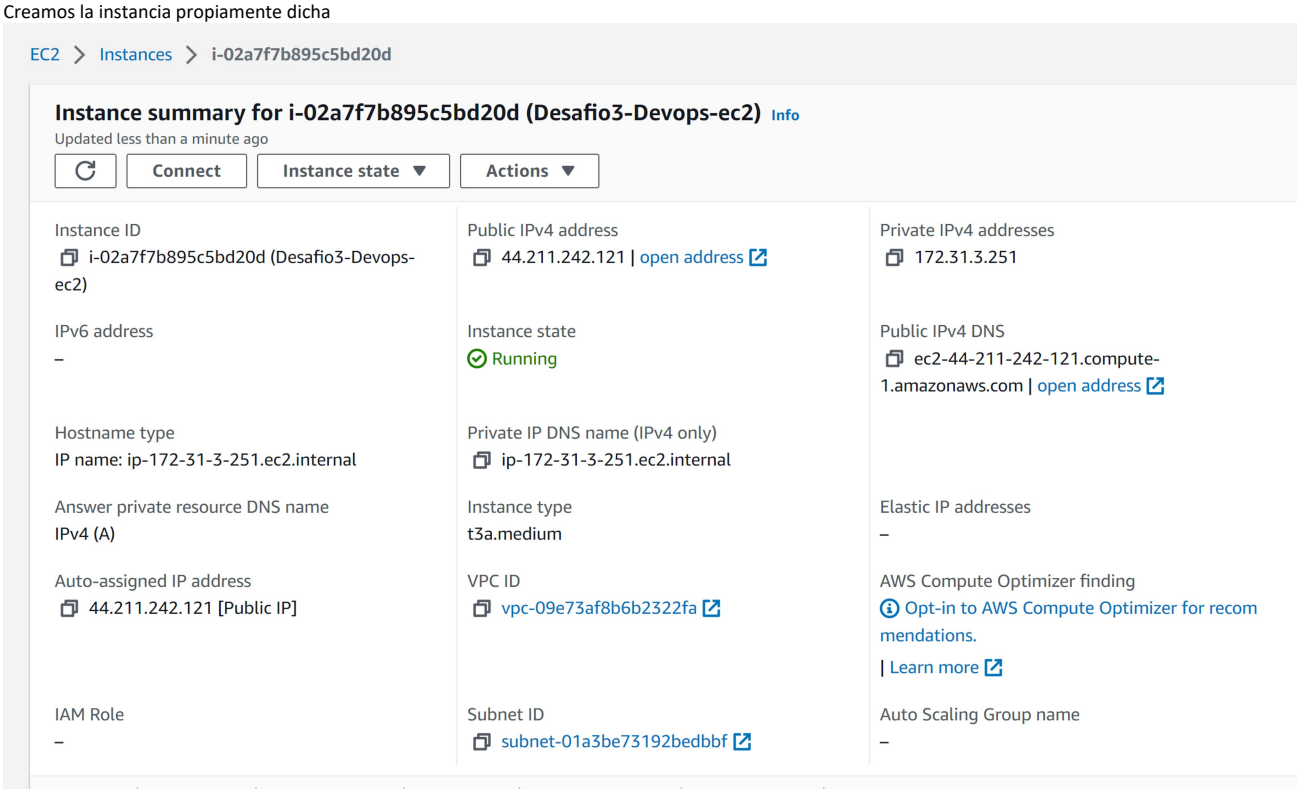
miércoles, 9 de noviembre de 2022 00:50

Para esto necesitamos crear una instancia ec2, donde alojaremos, entre otras cosas, Jenkins.

Creamos el SG:



Creamos la instancia propiamente dicha



Como Userdata agregamos:

```
#!/bin/bash
DOMAIN=$(curl icanhazip.com)
SSLIP="$DOMAIN.sslip.io"
sudo mv jenkins /etc/nginx/sites-available/
sudo apt update -y
sudo apt install -y wget unzip nginx
sudo systemctl start nginx
sudo mkdir -p /var/www/jenkins/html
sudo chown -R $USER:$USER /var/www/jenkins/html
sudo chmod -R 755 /var/www/jenkins
cat > jenkins <<EOF
server {
    listen 80;
    listen [::]:80;
    root /var/www/jenkins/html;
    index index.html index.htm index.nginx-debian.html;
    server_name $SSLIP www.$SSLIP;
    location / {
        #try_files $uri $uri/ =404;
        proxy_pass    http://$SSLIP:8080;
        proxy_read_timeout  90s;
    }
```

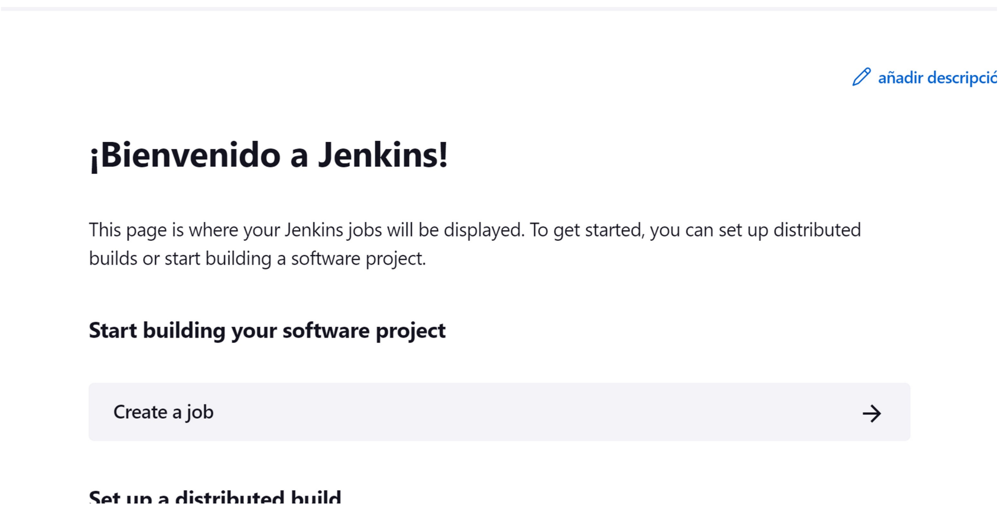
```
}
EOF
sudo mv jenkins /etc/nginx/sites-available/
sudo ln -s /etc/nginx/sites-available/jenkins /etc/nginx/sites-enabled/
sudo systemctl restart nginx
sudo apt install -y certbot python3-certbot-nginx
sudo certbot --nginx --register-unsafely-without-email --agree-tos -d "${SSLIP}" --cert-name
jenkins
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key |sudo gpg --dearmor -o
/usr/share/keyrings/jenkins.gpg
sudo sh -c 'echo deb [signed-by=/usr/share/keyrings/jenkins.gpg] http://pkg.jenkins.io/debian-
stable binary/ > /etc/apt/sources.list.d/jenkins.list'
sudo apt update -y
sudo apt install -y default-jre
sudo apt install -y jenkins
sudo systemctl start jenkins.service
```

Nos conectamos a la instancia para chequear si se hizo correctamente la instalacion.

```
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-3-251:~$ systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset:>
     Active: active (running) since Sun 2022-11-13 00:30:54 UTC; 3min 6s ago
       Docs: man:nginx(8)
   Main PID: 2211 (nginx)
      Tasks: 3 (limit: 4635)
     Memory: 5.5M
        CPU: 210ms
     CGroup: /system.slice/nginx.service
             ├─2211 "nginx: master process /usr/sbin/nginx -g daemon on; master>
             ├─2699 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" "">
             └─2700 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" "">

Nov 13 00:30:54 ip-172-31-3-251 systemd[1]: Starting A high performance web ser>
Nov 13 00:30:54 ip-172-31-3-251 systemd[1]: Started A high performance web serv>
lines 1-15/15 (END)...skipping...
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
     Active: active (running) since Sun 2022-11-13 00:30:54 UTC; 3min 6s ago
       Docs: man:nginx(8)
   Main PID: 2211 (nginx)
      Tasks: 3 (limit: 4635)
     Memory: 5.5M
        CPU: 210ms
     CGroup: /system.slice/nginx.service
             ├─2211 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
             ├─2699 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" ""
             └─2700 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" ""

Nov 13 00:30:54 ip-172-31-3-251 systemd[1]: Starting A high performance web server and a reverse proxy server...
Nov 13 00:30:54 ip-172-31-3-251 systemd[1]: Started A high performance web server and a reverse proxy server.
~
~
```

Vemos que el Userdata funciono correctamente.

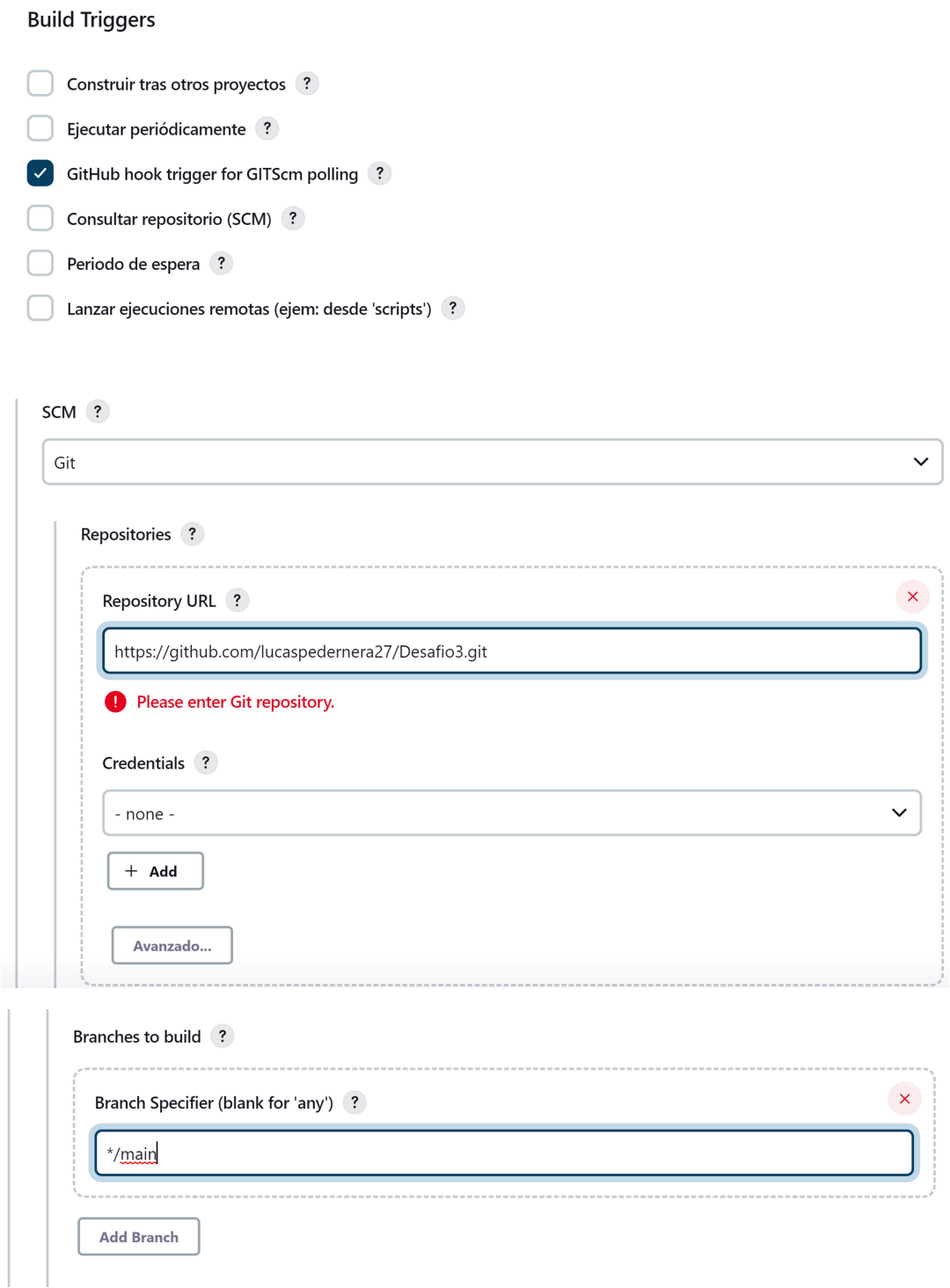
Para hacerla completa podemos utilizar
https://44.211.242.121.sslip.io/



Una vez configurado Jenkins, instalamos los pluggins sugeridos e instalamos los pluggins: S3

Publisher y AWS Global configuration.

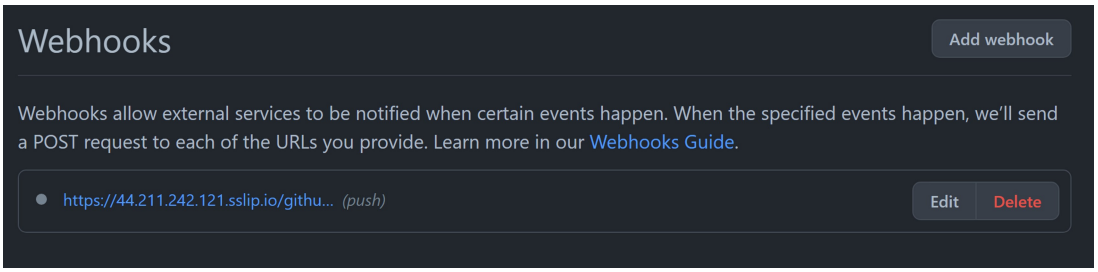
Configuramos Jenkins, agregamos un nuevo Pipeline con la configuracion para Git



A la par cargamos el JenkinsFile en el repo de GitHub con la configuracion:

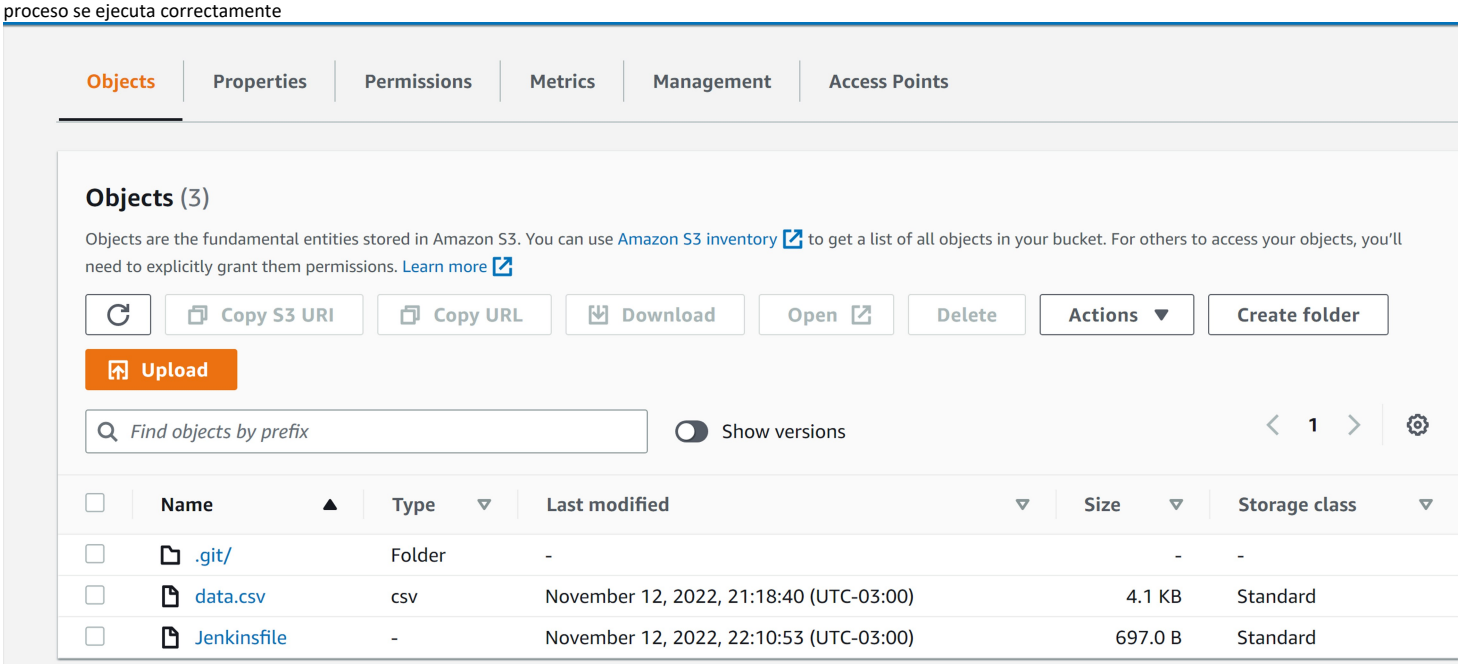
```
pipeline {
  agent any

  stages {
    stage('AWS STS') {
      steps {
        echo 'AWS STS'
        sh 'aws sts get-caller-identity'
      }
    }
    stage('AWS S3 listar') {
      steps {
        sh 'aws s3 ls'
      }
    }
    stage('Git Clone') {
```

```
            steps {
                sh 'rm -rf Desafio3/'
                sh 'git clone https://github.com/lucaspedernera27/Desafio3.git'
                sh 'ls -lrt Desafio3/'
            }
        }
        stage('Upload to S3') {
            steps {
                sh 'aws s3 cp Desafio3 s3://desafio3-csv-loader-bucket --recursive'
            }
        }
    }
}
```



Una vex hecha la configuracion del lado del repo...

Sumamos al repositorio, volvemos a Jenkins y le damos a Build with parameters para chequear si el proceso se ejecuta correctamente



Se ejecuto correctamente.

Ahora voy a hacer un pequeño cambio en la tabla y agregarla al repositorio para ver si la automatizacion con webhook funciona

El cambio funciono y el proceso quedo automatizado con Jenkins.