

SQA S.A - Innovación

CT - Porvenir

Versión 1.0

Tabla de Contenido

[Versionamiento 5](#_Toc81301050)

[Descripción General 5](#_Toc81301051)

[Evaluación Histórica 6](#_Toc81301052)

[Accesibilidad y flexibilidad 7](#_Toc81301053)

[¿En qué se diferencian las pruebas continuas de la automatización de pruebas? 8](#_Toc81301054)

[Sistemas involucrados en el proceso 10](#_Toc81301055)

[Alojamiento de aplicaciones 11](#_Toc81301056)

[Sitios web 11](#_Toc81301057)

[Backup y almacenamiento 11](#_Toc81301058)

[TI empresarial 11](#_Toc81301059)

[Entrega de contenido 11](#_Toc81301060)

[Bases de datos 11](#_Toc81301061)

[Servicios 11](#_Toc81301062)

[Consola de administración de AWS 13](#_Toc81301063)

[IAM 14](#_Toc81301064)

[Características 14](#_Toc81301065)

[Crear un usuario con permiso en IAM 16](#_Toc81301066)

[Ingresar con usuario creado 28](#_Toc81301067)

[AWS CodeCommit 29](#_Toc81301068)

[Configuración usuario para CodeCommit 31](#_Toc81301069)

[Amazon Elastic Compute Cloud (Amazon EC2) 39](#_Toc81301070)

[Características de Amazon EC2 39](#_Toc81301071)

[Configuración usuario para AWS EC2 40](#_Toc81301072)

[Creación de instancia en EC2 40](#_Toc81301073)

[Configuración de EC2 mediante PuTTY 58](#_Toc81301074)

[Configuración de Java 58](#_Toc81301075)

[Configuración de Gradle 59](#_Toc81301076)

[Configuración de Git 60](#_Toc81301077)

[Configuración de Jenkins 60](#_Toc81301078)

[Configuración de variables de entornos en Jenkins 66](#_Toc81301079)

[Integración Jenkins con CodeCommit 76](#_Toc81301080)

[Jenkins para proyectos Maven 83](#_Toc81301081)

[Jenkins para proyectos Gradle 84](#_Toc81301082)

[Reportes 85](#_Toc81301083)

[Cucumber 85](#_Toc81301084)

[Serenity BDD 88](#_Toc81301085)

[Ejecutar comando para organizar el reporte de SerenityBDD 91](#_Toc81301086)

[Configuración de Agentes en Jenkins 93](#_Toc81301087)

[Descargar Agente en dispositivo local 100](#_Toc81301088)

[Configuración de Agente con Pipeline 103](#_Toc81301089)

[Ejecutar un Pipeline 105](#_Toc81301090)

# Versionamiento

|  |  |  |  |
| --- | --- | --- | --- |
| MODIFICADO POR | VERSIÓN | FECHA | DESCRIPCIÓN |
| Javier Ricardo Charry Barrera | 1.0 | 10/08/2020 | Inicio de documento |

# Descripción General

Implementación de esquema de integración continua y construcción automática en pruebas QA: Revisando la percepción de recibir y generar consistentemente aplicaciones hacia ambientes de QA y producción para evitar errores.

•Automatización de despliegues: Permite eliminar errores en salidas a producción, promueve mayor frecuencia de salidas a producción al encontrarse en un esquema automatizado.

•Generación de Ambientes Dinámicos (dockerización): Dockerización de ambientes para disponer de ambientes paralelos de pruebas o de producción, así como disponer de crecimiento dinámico.

•Gestión del Ciclo de Vida de Desarrollo (Requerimientos → Release): Brindamos prácticas para gestionar el ciclo de vida de software, permitiéndoles aprender sobre empoderamiento y así, tener el control de su software. Herramientas para controlar y utilizar sus códigos fuente.

•Automatización de Pruebas Continuas: Automatización de casos de prueba, generación de datos de pruebas, virtualización de servicios.

• Seguridad en el Proceso de Desarrollo

• Monitoreo de Negocio (Ops)

• Implementación de infraestructura de nube híbrida para Contenedores

• Gestión del Cambio en la Organización

Prerrequisitos

Tener elaborado los scripts de automatización en un proyecto (verificar que se ejecuta de forma correcta localmente).

Tener Java instalado, versión 11 o superior. Asegurarse de que las variables de entorno están correctamente configuradas.

Mucha capacidad de investigación y recursividad.

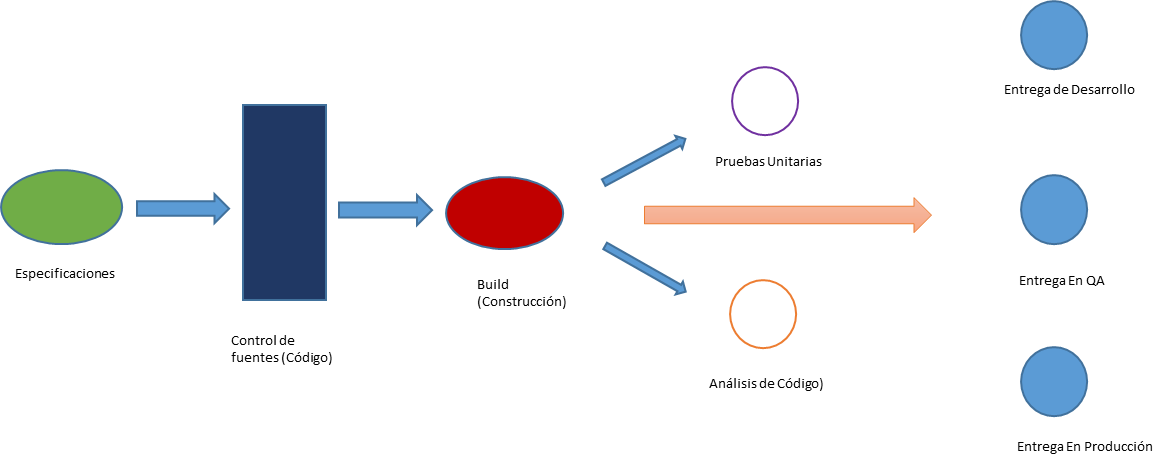
## Evaluación Histórica

La antigua forma de realizar pruebas era centrada en realizar pruebas funcionales de forma manual y convertirlas en transacciones una a una. El software se pasó de un equipo a otro. Un proyecto tendría fases definidas de desarrollo y garantía de calidad. Los equipos de control de calidad siempre querían más tiempo para garantizar la calidad. El objetivo era que la calidad prevaleciera sobre el cronograma del proyecto.

Sin embargo, las empresas o equipos de tecnología quieren una entrega más rápida de software al usuario final. Cuanto más nuevo es el software, mejor se puede distribuir y aumentar el potencial de entradas para el Core del negocio. Por lo tanto, se desarrolló una nueva forma de pruebas.

Continuo significa pruebas ininterrumpidas realizadas de forma continua. En un proceso de DevOps continuo, un cambio de software (candidato de lanzamiento) se mueve continuamente desde el desarrollo hasta las pruebas y la implementación.

## Accesibilidad y flexibilidad



El código se desarrolla, entrega, prueba e implementa continuamente.

Por ejemplo, cada vez que un desarrollador verifica el código en el servidor de código fuente, como Jenkins, se ejecuta un conjunto automatizado de pruebas unitarias en el proceso continuo.

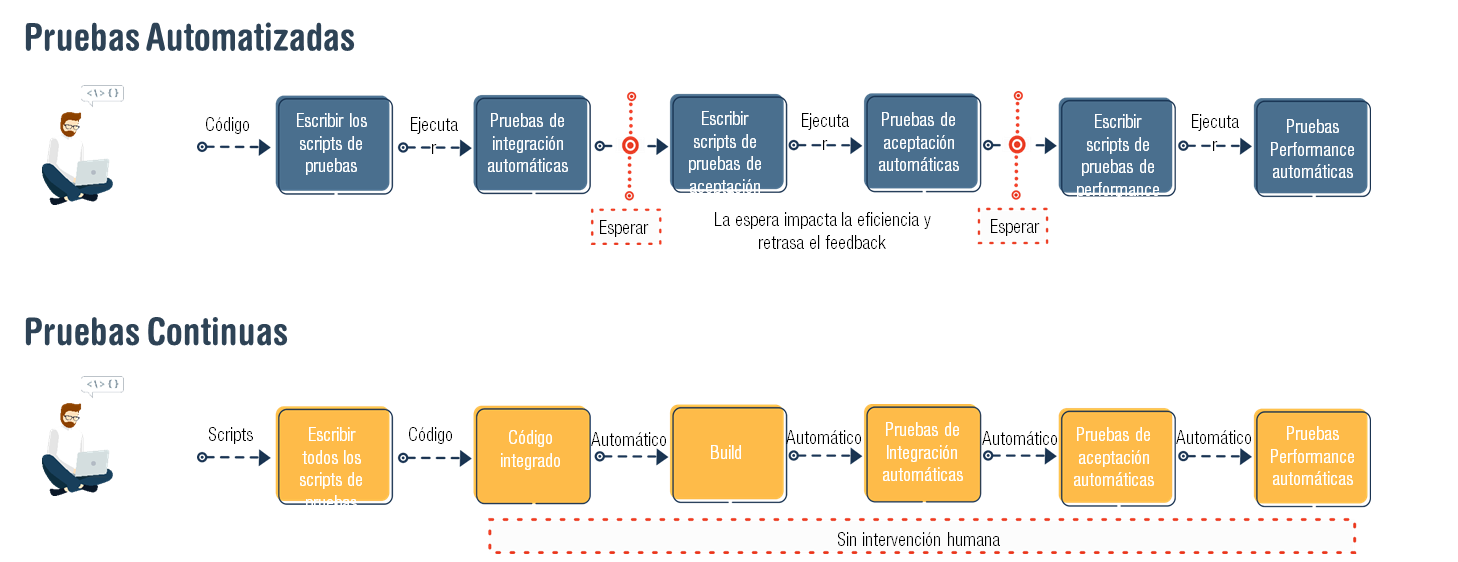
* Si las pruebas fallan, la compilación se rechaza y se notifica al desarrollador.
* Si la compilación pasa la prueba, se implementa en servidores de control de calidad y rendimiento para pruebas funcionales y de carga exhaustivas. Las pruebas se realizan en paralelo.
* Si las pruebas pasan, el software se implementa en producción.

La prueba continua es un pequeño engranaje en el ciclo de desarrollo, integración e implementación continuos.

## ¿En qué se diferencian las pruebas continuas de la automatización de pruebas?

**Automatización de pruebas frente a pruebas continúas**

|  |  |  |
| --- | --- | --- |
| Parámetro | Automatización de pruebas | Prueba continua |
| Definición | La automatización de pruebas es un proceso en el que se utilizan herramientas o software para automatizar tareas. | Es una metodología de prueba de software que se enfoca en lograr una mejora y calidad continuas. |
| Objetivo | Un conjunto de tareas similares o repetitivas que una máquina puede ejecutar, más rápido y con menos errores. | El proceso de prueba continuo ayuda a encontrar el riesgo, abordarlo y mejorar la calidad del producto. |
| Requisito previo | La automatización en las pruebas es posible sin integrar las pruebas continuas. | Las pruebas continuas no se pueden implementar sin la automatización de pruebas. |
| Tiempo | El lanzamiento del software puede tardar de un mes a años. | La versión del software puede publicarse semanalmente o cada hora. |
| Retroalimentación | Comentarios regulares después de probar cada versión. | La retroalimentación en cada etapa debe ser instantánea. |
| Historia | Las pruebas automatizadas se han realizado durante décadas para acelerar el proceso de prueba. | La prueba continua es un concepto relativamente nuevo. |



Cómo hacer pruebas continuas

* Uso de herramientas para generar un conjunto de automatización de pruebas a partir de historias / requisitos de usuario
* Crear entorno de prueba.
* Copie y anonimice los datos de producción para crear un banco de datos de prueba
* Utilice la virtualización de servicios para probar la API
* Pruebas de rendimiento en paralelo

Análisis del proceso actual Porvenir

El objetivo de este esta sección es describir cómo porvenir está manejando el proceso ahora, antes de optimizarlo y automatizarlo. Se busca responder a preguntas como: ¿Cuáles son los pasos que debe seguir alguien dentro de la empresa cada vez para que este proceso funcione?, ¿Qué software tienen que utilizar? ¿Dónde obtienen la información que necesitan? ¿Qué ocurre si algo va mal? Entre otras.

### Sistemas involucrados en el proceso

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Descripción** | **Sistema** | **Se necesita rol de usuario** | **Tecnología usada por el sistema** | **Versión** |
| Qué **Plataforma**está usando: | Azure. |  | Portal Zona Franca | 2020 |
| ¿Qué **servicio**de control de versiones usa? | GitLab | Usuario de auxiliar solo para aprobación y eliminación. | N/A | N/A |
| ¿Qué herramienta de **compilación**de código utiliza? | Maven | N/A | N/A | N/A |
| ¿Qué herramienta utiliza para gestionar los **pipelines**? | Jenkins  y está montado en un servidor de la nube (Azure) | Usuario y Contraseña | N/A | (2.277.1) |
| En qué **tecnologías**de desarrollo están construida la **aplicación web**con la realizará el**piloto de CT**? | **Front**    Java (Freamework Liferay ) | No requiere usuario | N/A | N/A |
| **Back** | Java (EJB’s) | N/A | N/A |
| Base de datos de salida | Oracle | N/A | N/A | N/A |

### Alojamiento de aplicaciones

Utilice infraestructura bajo demanda de confianza para procesar las aplicaciones, desde aplicaciones internas hospedadas hasta ofertas SaaS.

### Sitios web

Satisfaga las necesidades de hospedaje web dinámicas con la plataforma de infraestructura escalable de AWS.

### Backup y almacenamiento

Almacene datos o cree soluciones de backup de confianza mediante los servicios de almacenamiento de datos económicos de AWS.

### TI empresarial

Hospede aplicaciones de TI internas o externas en entornos seguros de AWS.

### Entrega de contenido

Distribuya de manera rápida y sencilla contenido a usuarios finales en todo el mundo, con precios bajos y grandes velocidades de transferencia de datos.

### Bases de datos

Aprovéchese de una variedad de soluciones de bases de datos escalables, desde las alojadas en empresas hasta las no relacionales.

# Servicios

Ofrece potencia informática, almacenamiento, entrega de contenido y otras funciones que las organizaciones pueden utilizar para implementar aplicaciones y servicios de manera rentable, con flexibilidad, escalabilidad y confiabilidad. El autoservicio de AWS significa que puede abordar de manera proactiva sus planes internos y reaccionar a las demandas externas cuando lo desee.

Configuración de Maven

Para configurar Maven ingresaremos los siguientes comandos:

* sudo yum install maven

Nos pedirá ingresar la letra ‘y’ para permitir instalaciones o actualizaciones

* mvn –version

Al final debemos ver el mensaje en pantalla de PuTTY

Apache Maven 3.0.5 (Red Hat 3.0.5-17)

Maven home: /usr/share/maven

Java version: 1.8.0\_282, vendor: Red Hat, Inc.

Java home: /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.282.b08-1.amzn2.0.1.x86\_64/jre

Default locale: en\_US, platform encoding: UTF-8

OS name: "linux", version: "4.14.225-169.362.amzn2.x86\_64", arch: "amd64", family: "unix"

En la respuesta anterior podremos ver el valor del Maven home el cual vamos a copiar y pegar en el siguiente comando

* export MAVEN\_HOME=”/usr/share/maven”
* echo $MAVEN\_HOME

Se nos mostrara la ruta que hemos ingresado

### Configuración de Gradle

Para configurar Gradle ingresaremos los siguientes comandos:

* wget https://services.gradle.org/distributions/gradle-5.0-bin.zip -P /tmp
* sudo unzip -d /opt/gradle /tmp/gradle-5.0-bin.zip
* export GRADLE\_HOME=/opt/gradle/gradle-5.0
* export PATH=${GRADLE\_HOME}/bin:${PATH}
* gradle -v

Al final debemos ver el mensaje en pantalla de PuTTY



### Configuración de Git

Para configurar Git ingresaremos los siguientes comandos:

* sudo yum install git
* git –version

Se mostrara una respuesta similar a la siguiente:

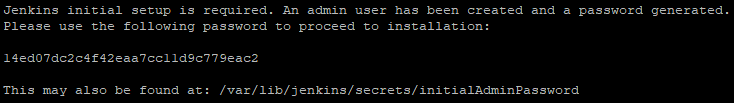
git version 2.23.4

### Configuración de Jenkins

Para configurar Jenkins ingresaremos los siguientes comandos:

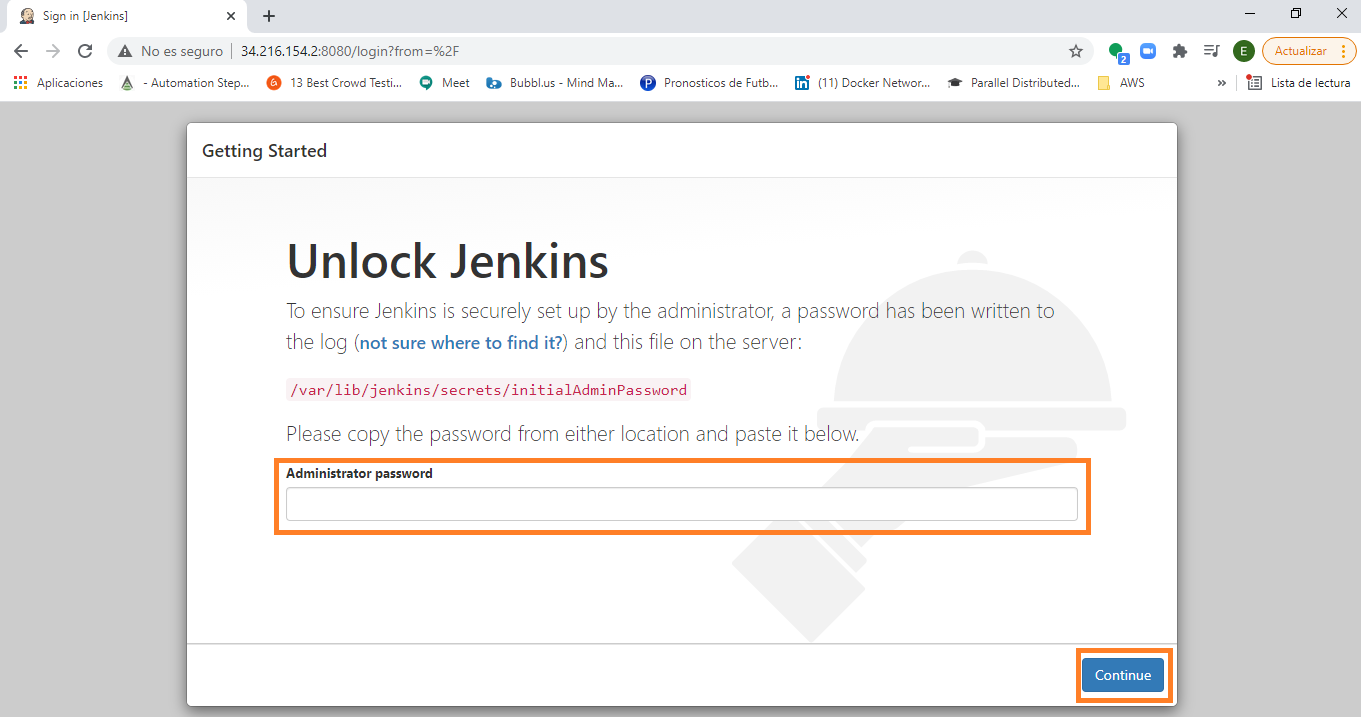
* sudo rpm --import https://pkg.jenkins.io/redhat/jenkins.io.key
* sudo wget -O /etc/yum.repos.d/jenkins.repo <http://pkg.jenkins-ci.org/redhat/jenkins.repo>
* sudo yum install Jenkins
* sudo systemctl start jenkins
* sudo systemctl enable jenkins
* sudo systemctl status Jenkins
* sudo grep -A 5 password /var/log/jenkins/jenkins.log

Se mostrara una respuesta similar a la siguiente:

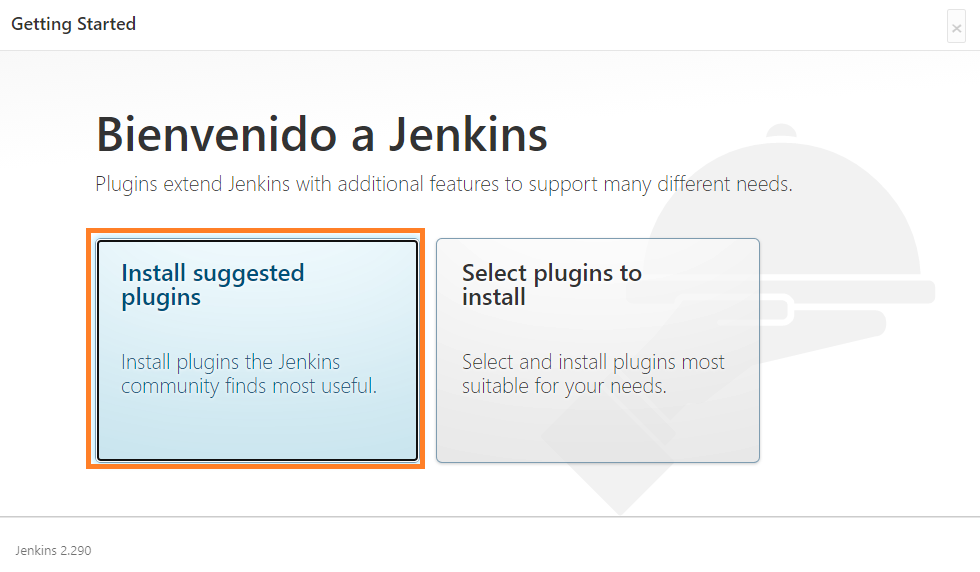


El número 14ed07dc2c4f42eaa7cc11d9c779eac2 es diferente por instancia, ese número se copia.

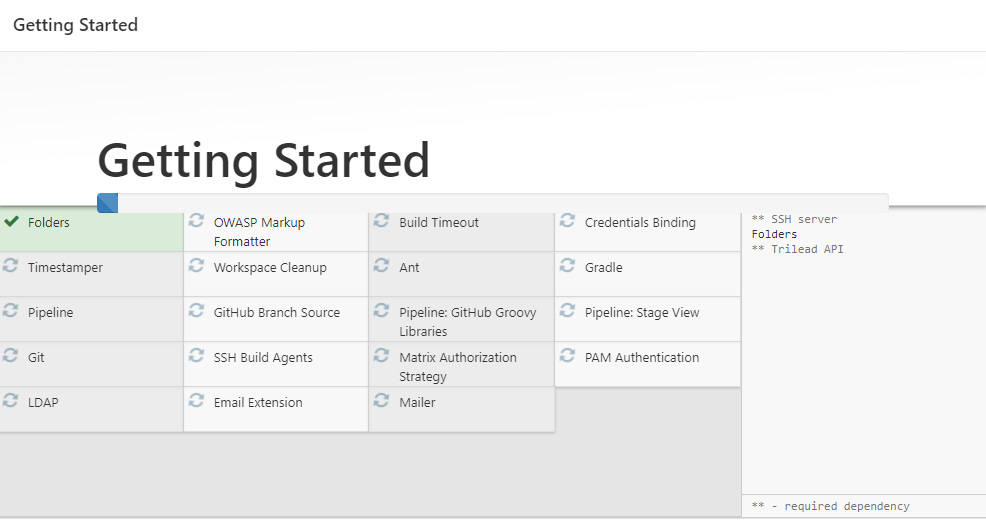
Ingresamos **IP** de nuestra instancia con el puerto 8080 atreves de nuestro navegador de preferencia y pegamos el numero anterior en el campo **Administrator password** y damos clic en el botón **Continue**:



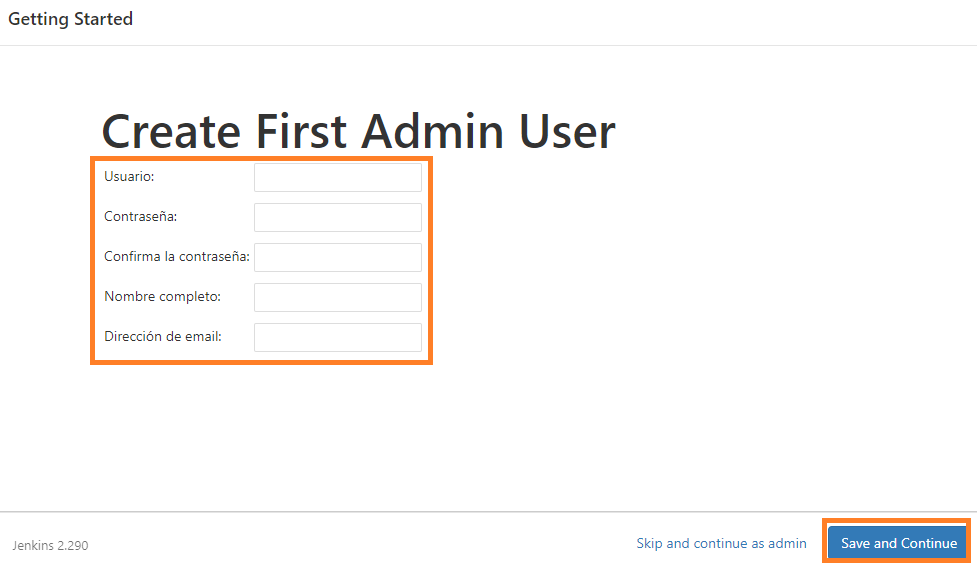
Damos clic en el boton **Install suggested plugins**

****

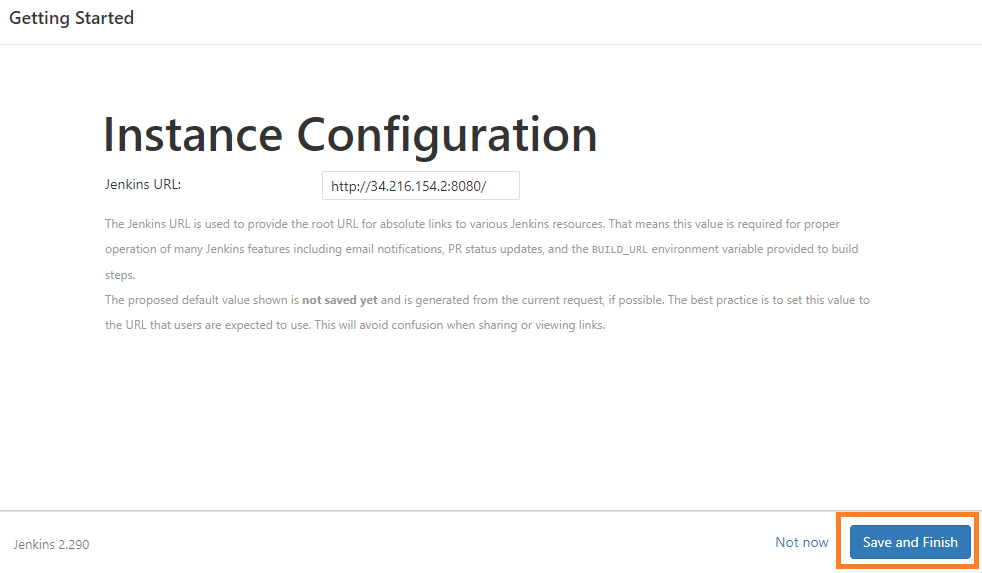
Esperamos que finalice la instalación



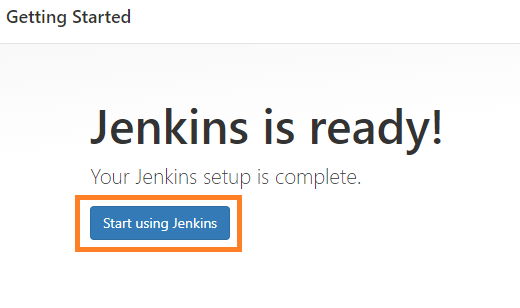
Ingresamos los datos del formulario y damos clic en el botón **Save and Continue**

****

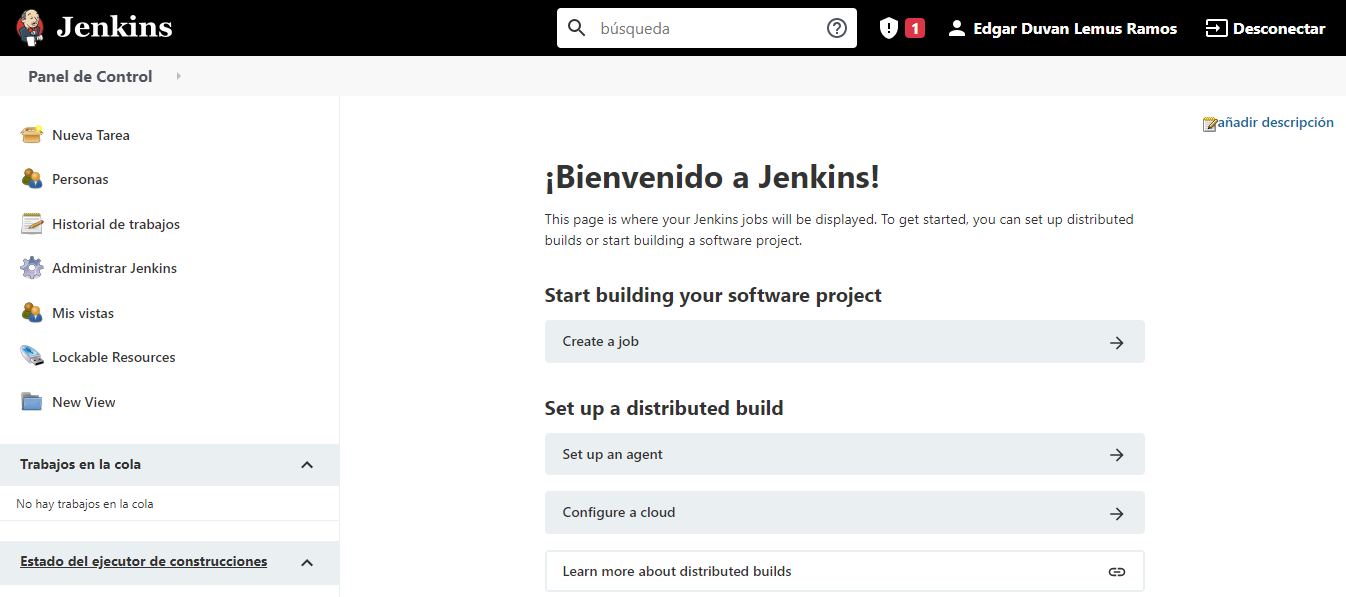
Damos clic en el boton **Save and Finish**

****

Damos Clic en el boton **Start using Jenkins**

****

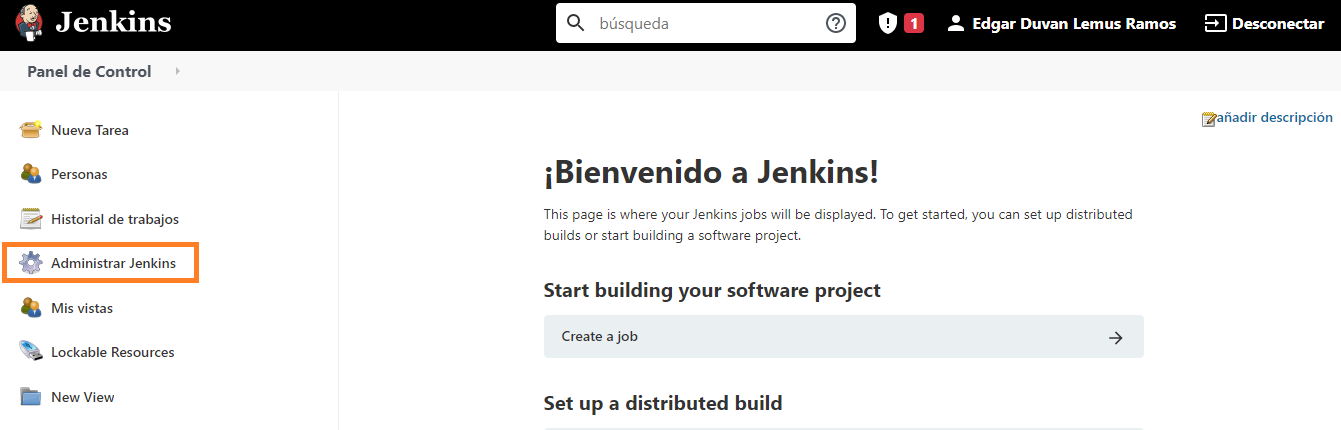
Finalmente veremos la interface de Jenkins en la **IP** de nuestra instancia en EC2



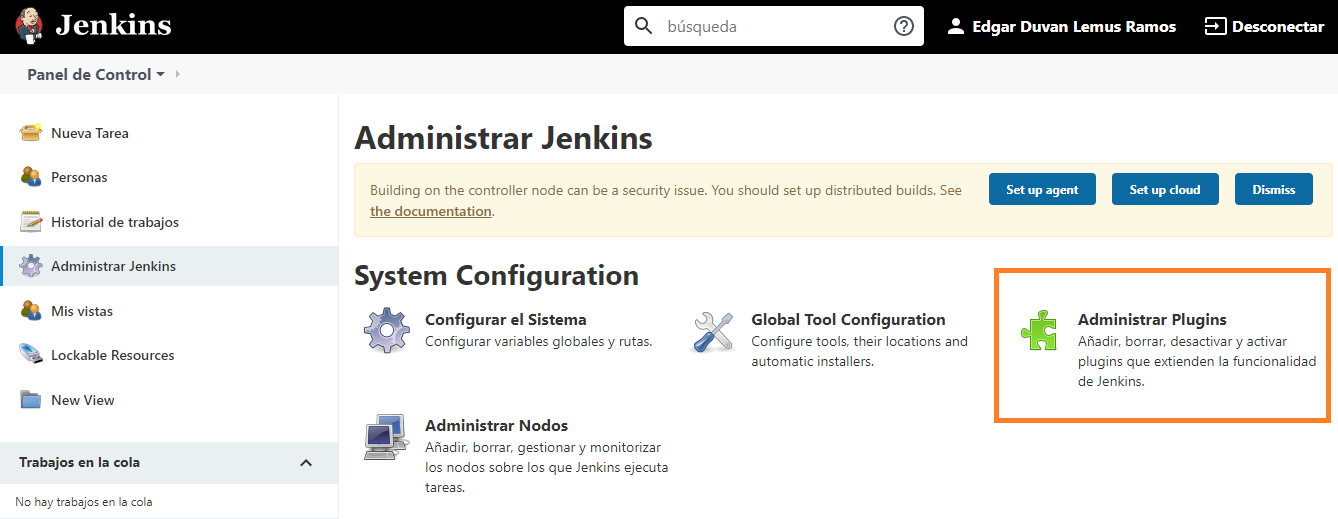
### Configuración de variables de entornos en Jenkins

Para configurar las variables de entornos de nuestra instancia en Jenkins, realizamos los siguientes pasos:

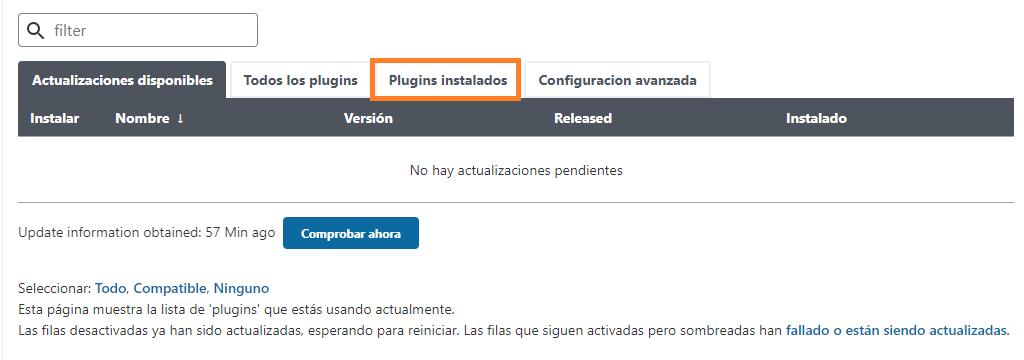
**Paso 1:** Damos clic en el boton **Administrar Jenkins**



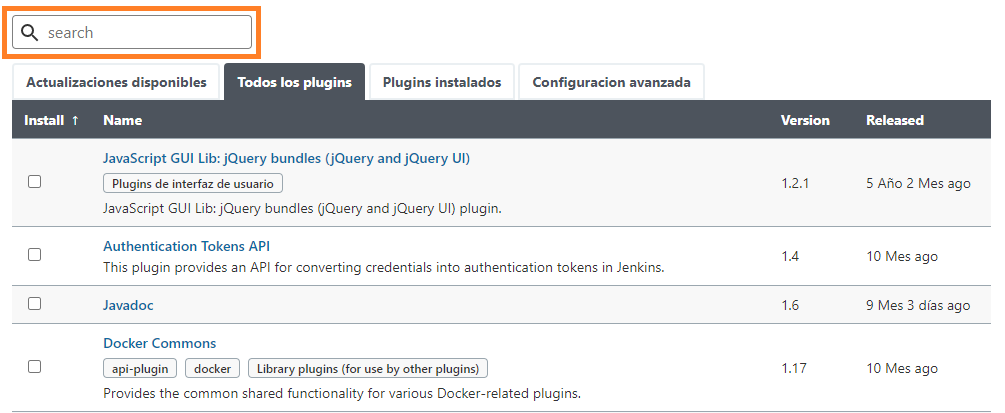
**Paso 2:** Damos clic en el boton **Administrar Plugins**

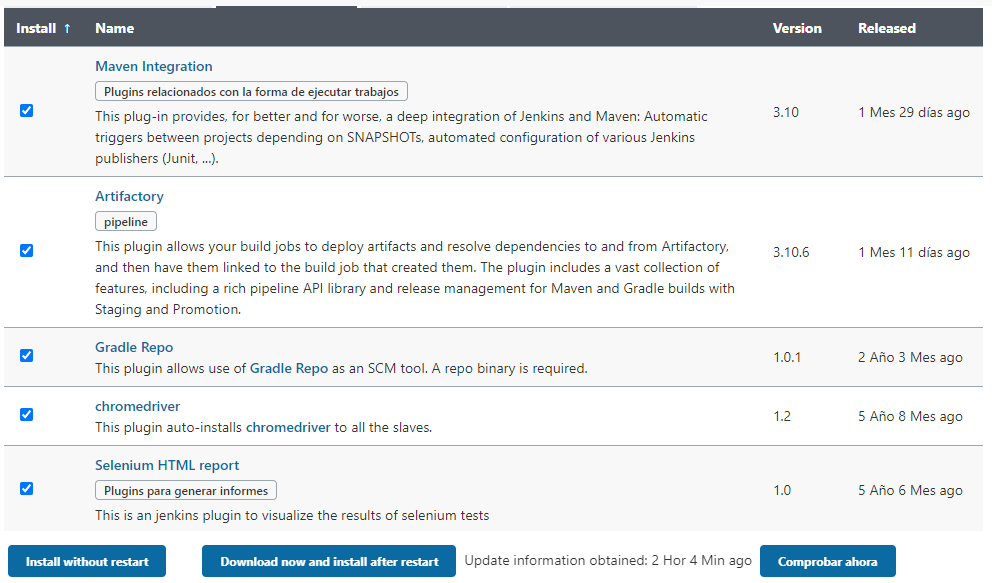
****

**Paso 3:** Damos clic en la pestaña **Todos los plugins**

****

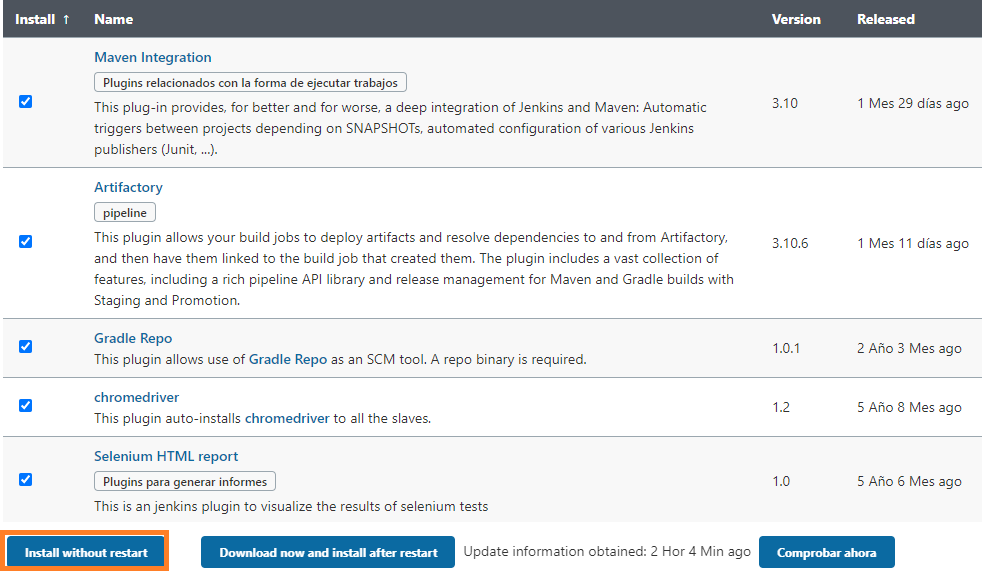
**Paso 4:** Buscamos los plugins **Maven Integration, Artifactory, Gradle Repo, chromedriver** y **Selenium HTML report** en el campo de búsqueda y los seleccionamos



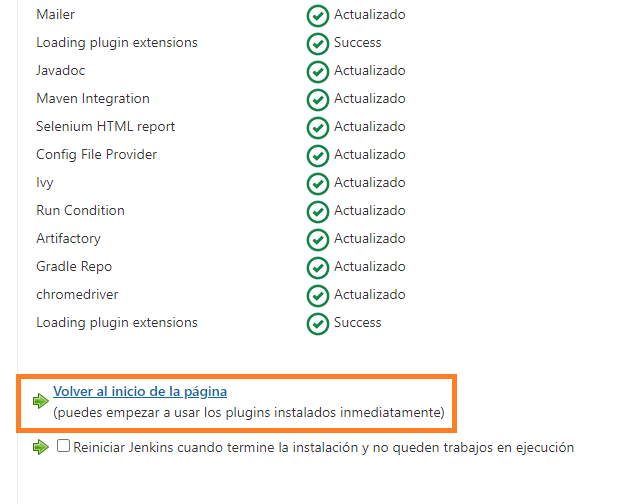


También agregamos los siguientes plugins **Thucydides, Publishing HTML Reports**

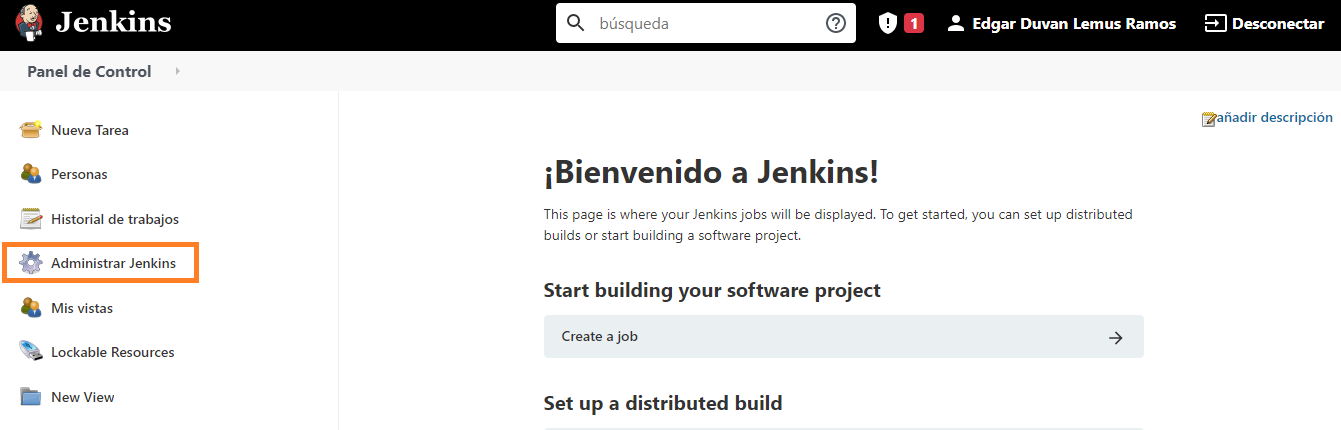
**Paso 5:** Damos clic en el boton **Install without restart**

****

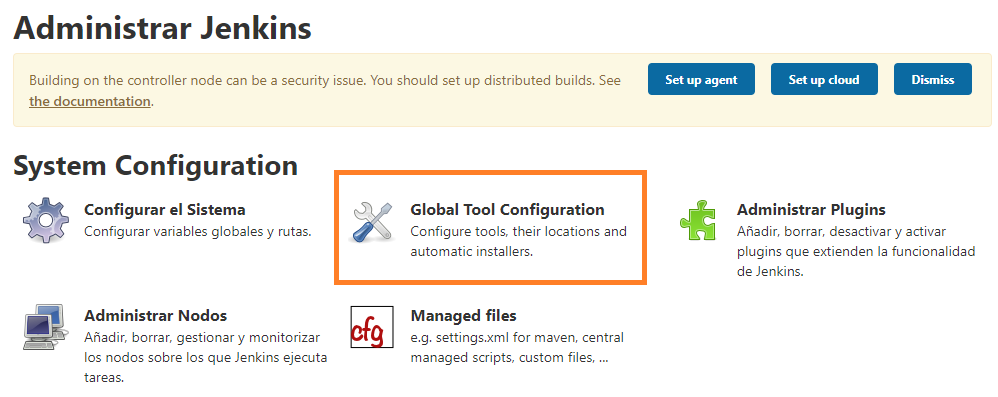
**Paso 6:** Esperamos a que se termine las instalaciones de los plugins y damos clic en la opción **Volver al inicio de la pagina**

****

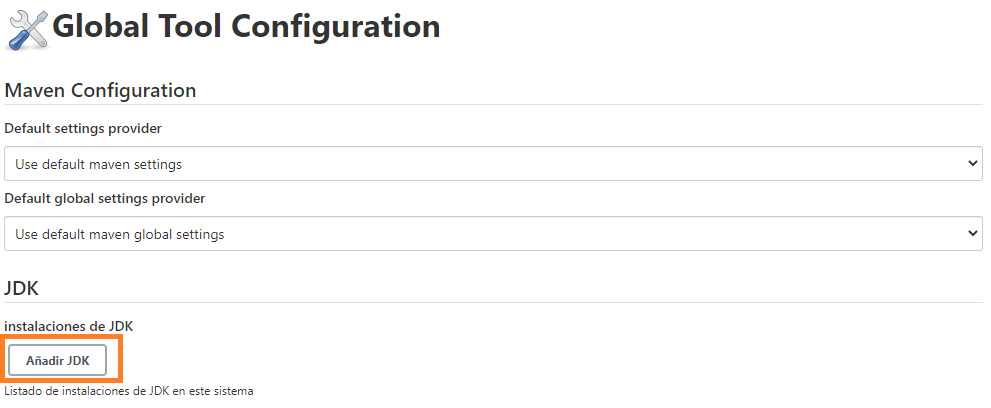
**Paso 7:** Damos clic en el boton **Administrar Jenkins**



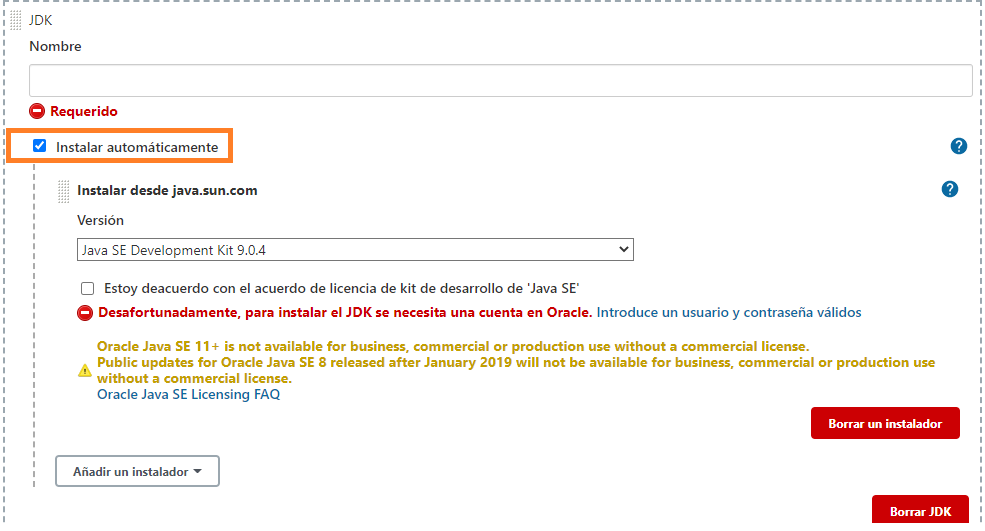
**Paso 8:** Damos clic en el boton **Global Tool Configuration**

****

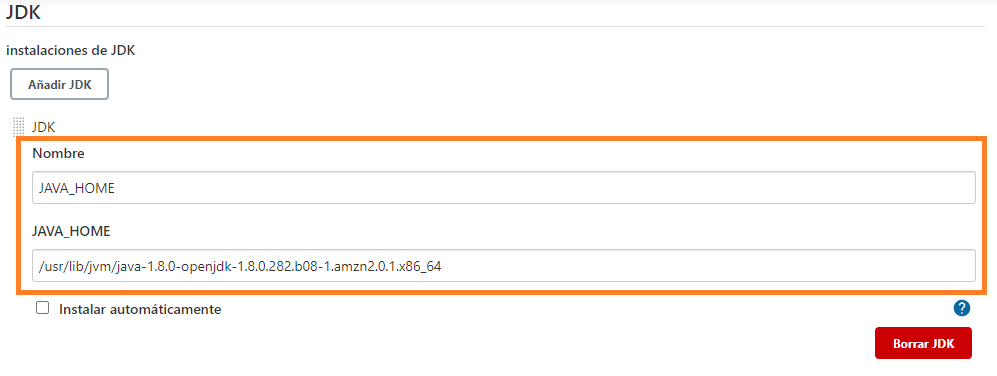
**Paso 9:** Buscamos la opción de **JDK** y damos clic en el boton **Añadir JDK**

****

**Paso 10:** Deseleccionamos la opción **Instalar desde java.sun.com**

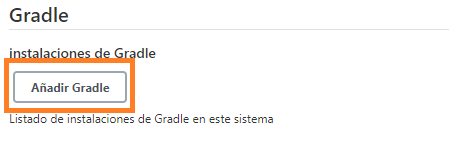
****

**Paso 11:** Ingresamos el nombre de nuestro JDK y la ruta del JAVA\_HOME asignado en PuTTY

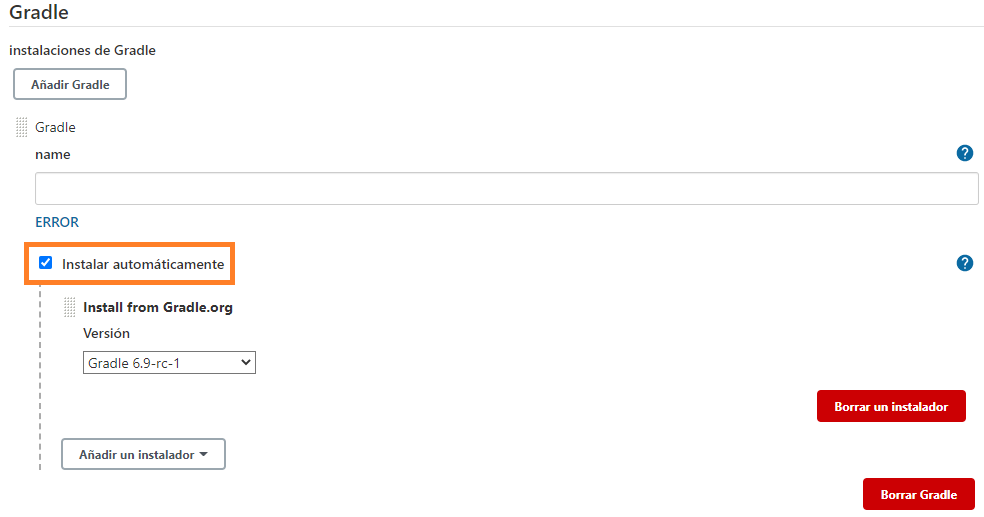


**NOTA:** Eliminar de la ruta del **JAVA\_HOME** la sección **/jre** para evitar error

**Paso 12:** Buscamos la opción de Gradle y damos clic en el boton **Añadir Gradle**



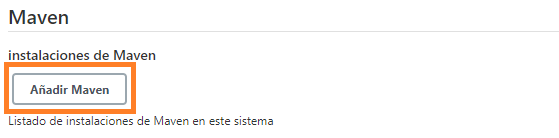
**Paso 13:** Deseleccionamos la opción **Instalar from Gradle.org**

****

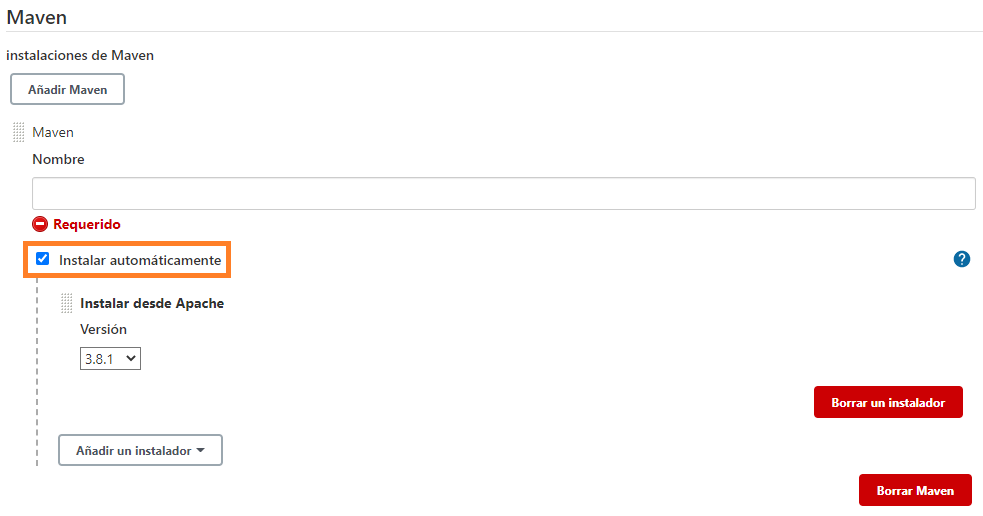
**Paso 14:** Ingresamos el nombre de nuestro Gradle y la ruta del GRADLE\_HOME asignado en PuTTY



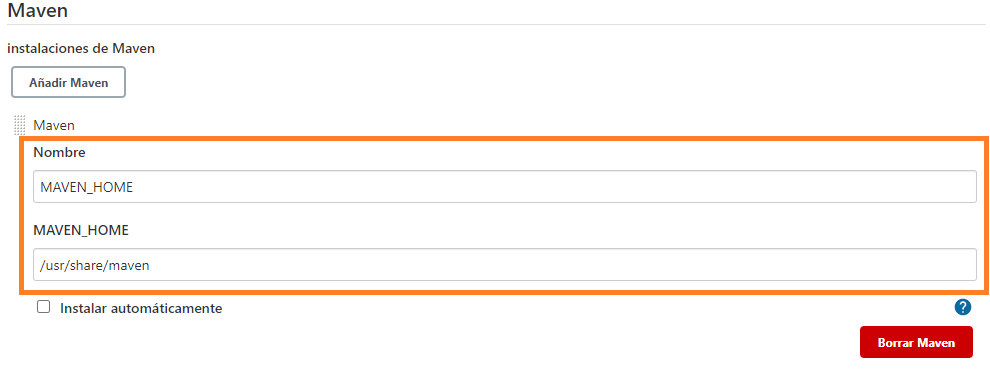
**Paso 15:** Buscamos la opción de Maven y damos clic en el boton **Añadir Maven**



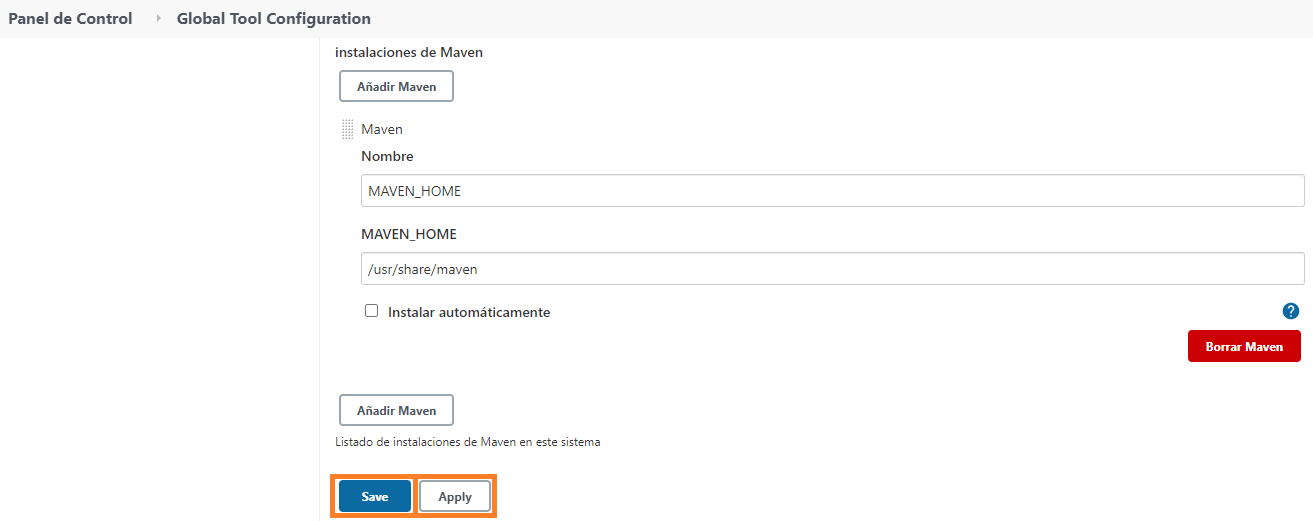
**Paso 16:** Deseleccionamos la opción **Instalar desde Apache**

****

**Paso 17:** Ingresamos el nombre de nuestro Maven y la ruta del MAVEN\_HOME asignado en PuTTY



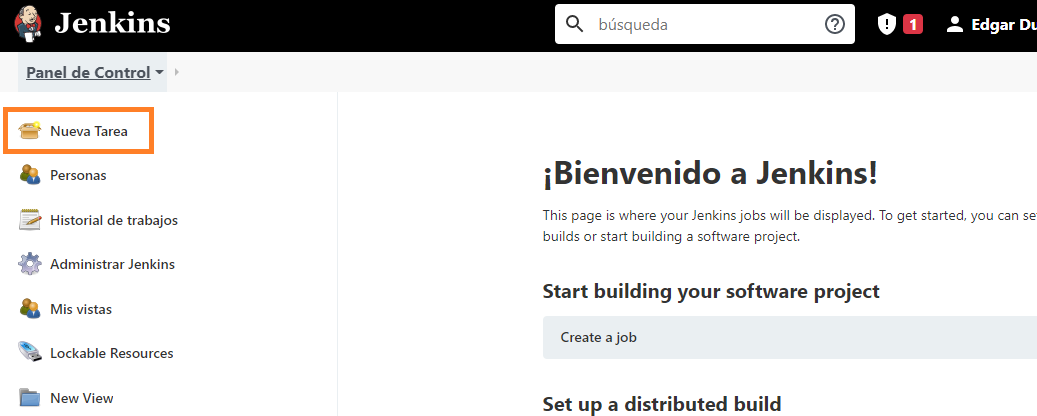
**Paso 18:** Damos clic al botón **Apply** y luego al botón **Save**

****

## Integración Jenkins con Gitlab

Para la integración de Jenkins con CodeCommit vamos a reutilizar mucha información creada anteriormente, seguiremos los siguientes pasos:

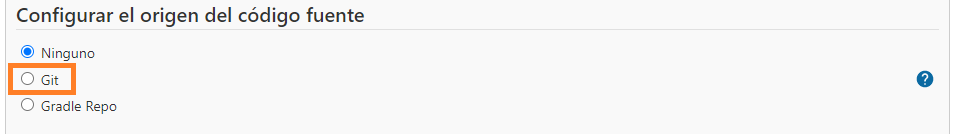
**Paso 1:** Damos clic a la opción **Nueva Tarea**



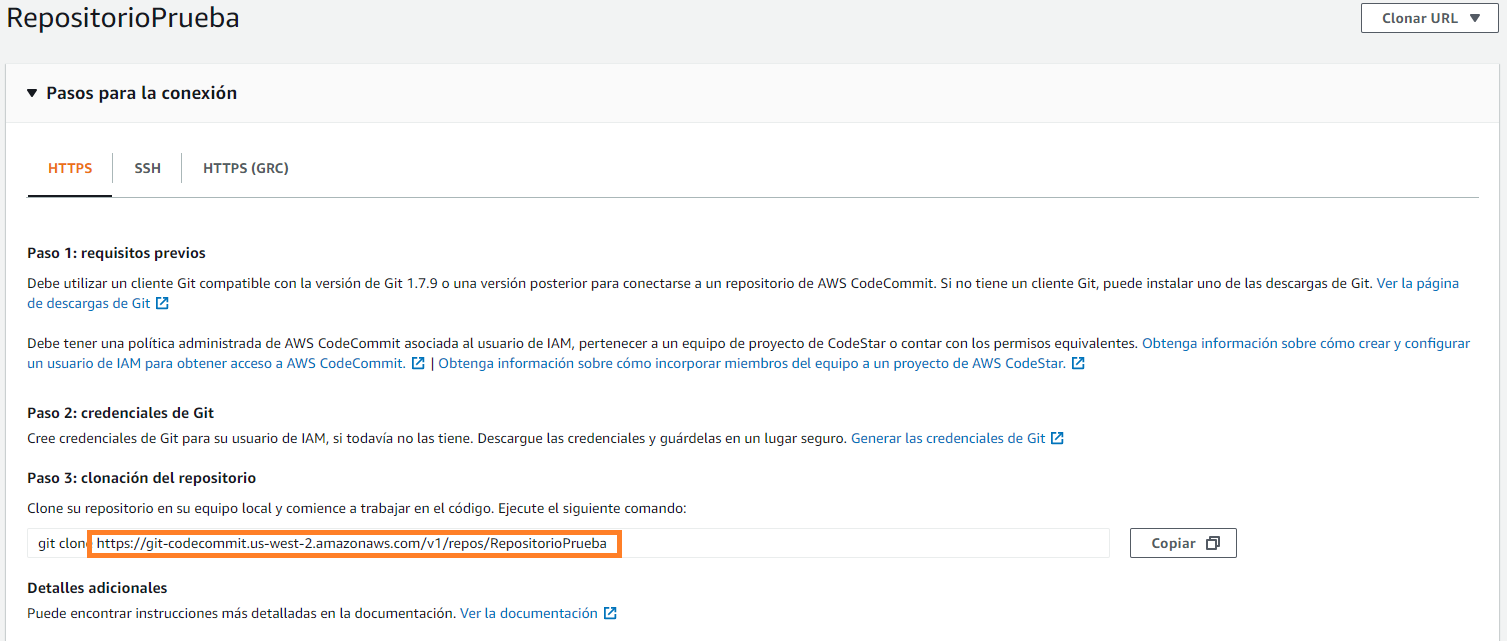
**Paso 2:** Ingresamos el nombre del pipeline, seleccionamos la opción **Crear un proyecto de estilo libre** y damos clic en el boton **OK**

****

**Paso 3:** En la pestaña **Configurar el origen del código fuente** seleccionamos la opción de **Git**

****

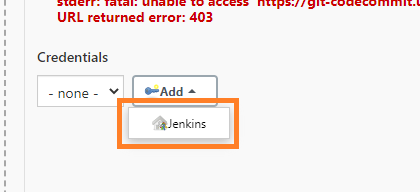
**Paso 4:** Copiamos la **URL Repository** de **CodeCommit**



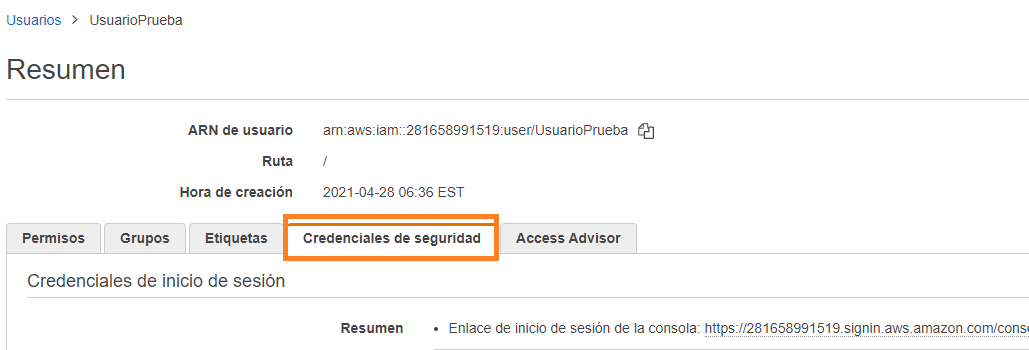
**Paso 5:** Pegamos la URL del repositorio en el campo **Repository URL** y damos clic en el botón **Add**

****

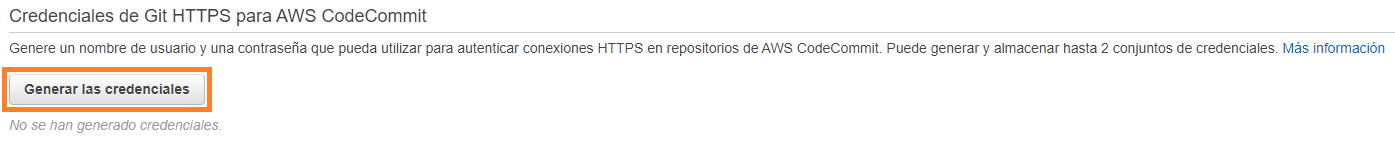
**Paso 6:** Escogemos la opción **Jenkins**

****

**Paso 7:** Ingresamos a la pestaña **Credenciales de seguridad** en el resumen de nuestro usuario en la consola de **IAM**

****

**Paso 8:** En las opciones de la pestaña buscamos la opción **Credenciales de Git HTTPS para AWS CodeCommit** y le damos clic al botón **Generar las credenciales**

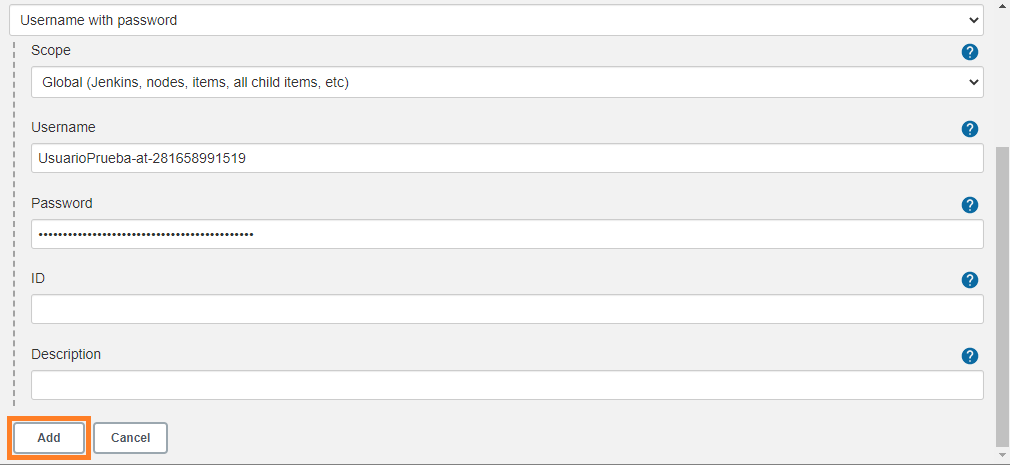
****

**Paso 9:** Nos aparecerá una ventana emergente el cual vamos a darle clic en el botón **Descargar credenciales** y luego clic en el botón **Cerrar**

**Paso 10:** Abrimos el Excel que descargamos y obtenemos el User Name y el Password y volvemos a **Jenkins** y agregamos al campo **Username** y al campo **Password** en las credenciales que nos aparece en **Jenkins**



**Paso 11:** Damos clic en el boton **Add**

****

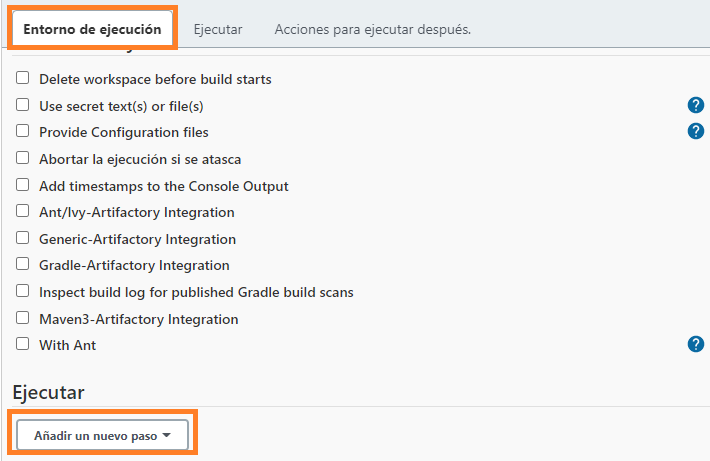
**Paso 12:** Damos clic en **Credenciales** y elegimos la credencial que acabamos de crear



**Paso 13:** Una vez escojamos la credencial nos debe quedar como la siguiente imagen

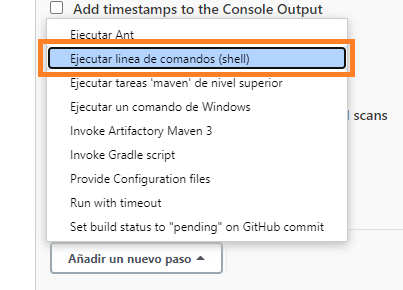


**Paso 14:** Vamos a la pestaña de **Entorno de ejecución** y en la sección de **Ejecutar** damos clic en el boton **Añadir un nuevo paso**

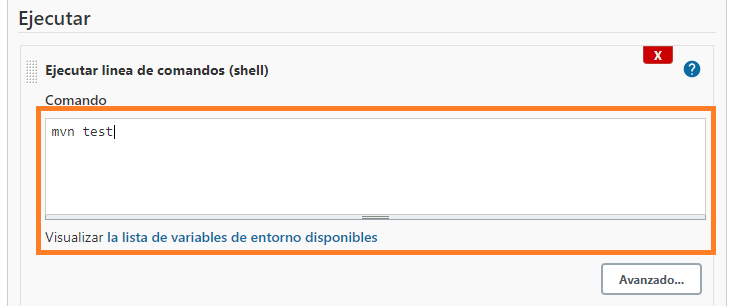


## Jenkins para proyectos Maven

**Paso 14.1:** Escogemos la opción **Ejecutar línea de comandos (shell)**

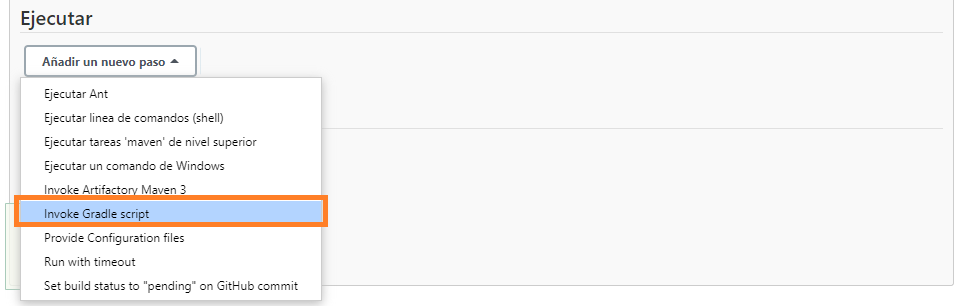


**Paso 14.2:** Escribimos el comando **mvn test** en la caja de texto

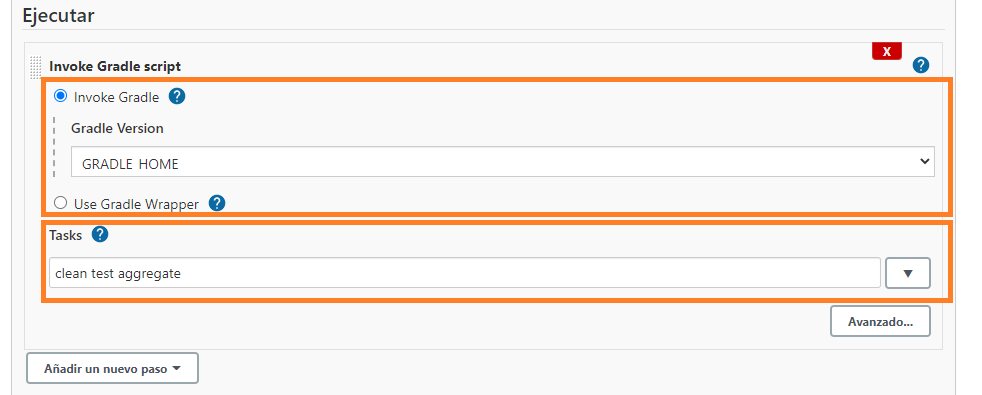


## Jenkins para proyectos Gradle

**Paso 14.1**: Escogemos la opción **Invoke Gradle script**



**Paso 14.2**: Seleccionamos la opción **Invoke Gradle** y escogemos nuestro **GRADLE\_HOME** en la opción **Gradle Version** y finalizamos agregando el comando **clean test aggregate**



## Reportes

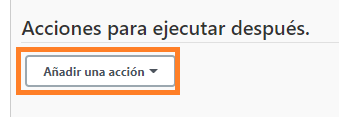
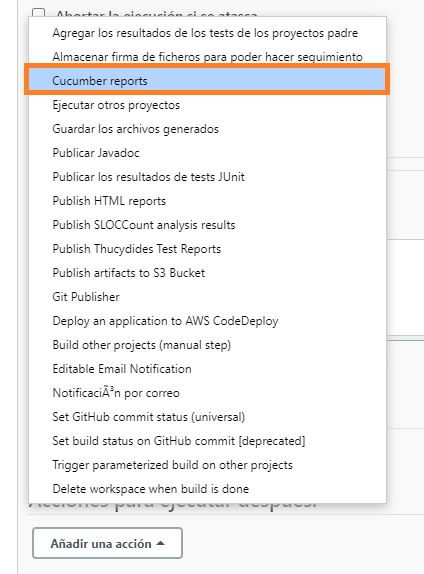
Los reportes es el resultado en estadísticas al final de nuestras ejecuciones, donde podemos observar el paso a paso realizado por nuestras pruebas automatizadas y tomas de evidencias.

### Cucumber

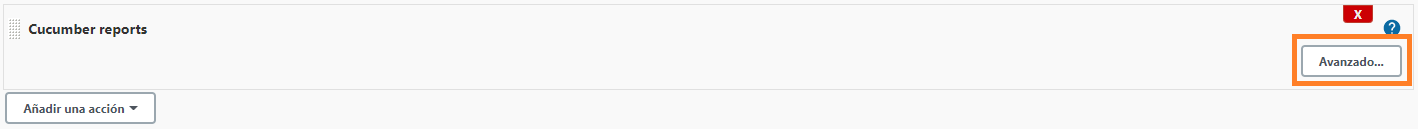
**Paso 15.1:** Inicial debeos modificar el **Runner** de nuestro proyecto insertando en los **CucumberOptions** la siguiente línea de código.

format = {"json:target/cucumber.json"}

**Paso 15.2:** Volviendo a Jenkins en la configuración de nuestro Pipeline en la sección de **Acciones para ejecutar después** damos clic en **Añadir una acción** y escogemos la opción de **Cucumber Report**

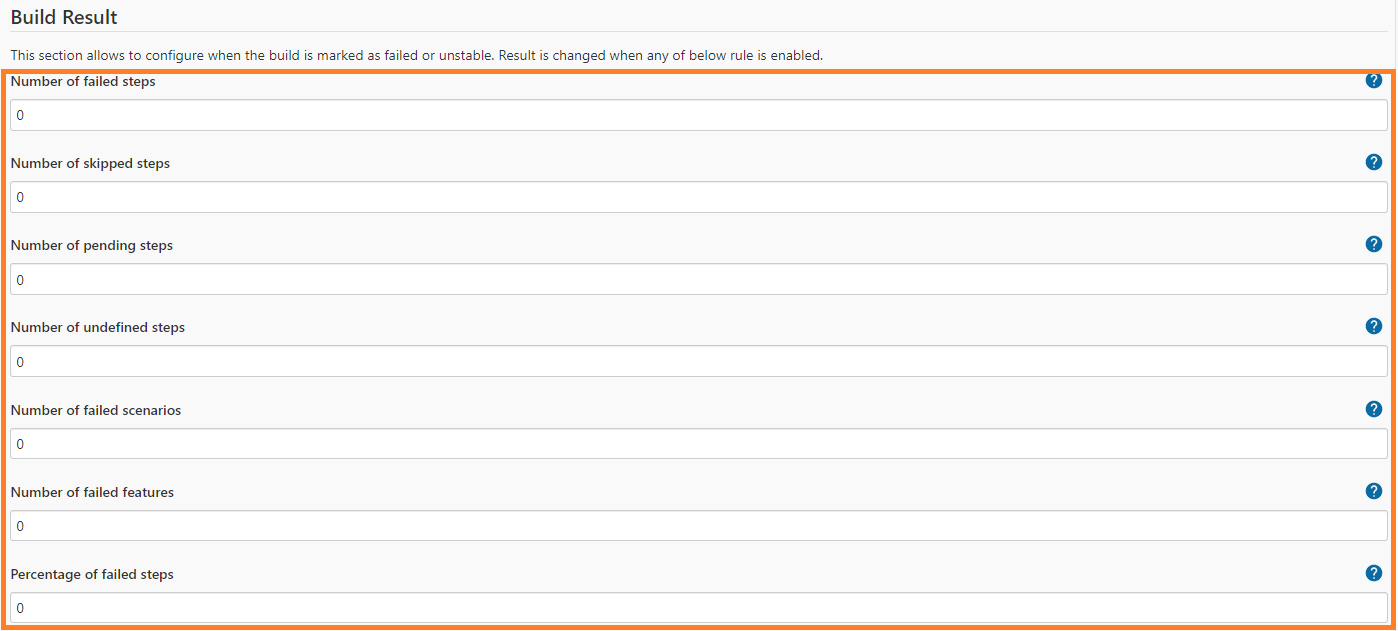
****

**Paso 15.3:** Damos clic en **Avanzado**



**Paso 15.4:** En el campo **Report title**  damos el nombre a nuestro reporte y en los campos de **Build Result** donde aparece el valor de **-1** lo reemplazamos por el valor de **0**



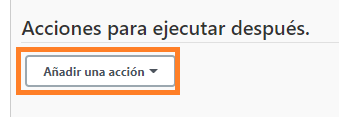


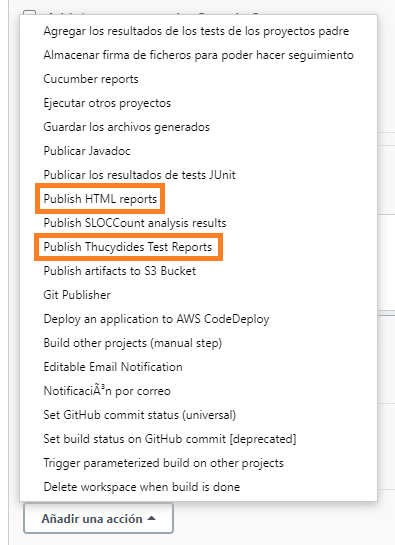
**Nota:** Para conocer más sobre el reporte de Cucumber con Jenkins, lo pueden hacer viendo los siguientes links.

* <https://www.lambdatest.com/blog/cucumber-with-jenkins-integration/>
* <https://www.jenkins.io/doc/pipeline/steps/cucumber-reports/>
* <https://plugins.jenkins.io/cucumber-reports/>

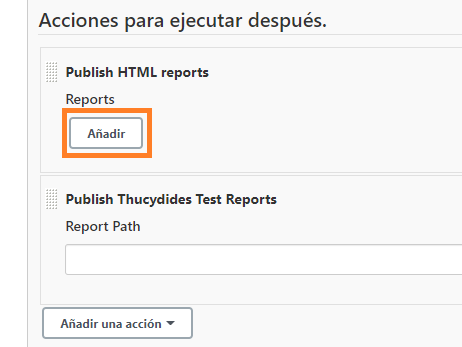
### Serenity BDD

**Paso 15.1**: En la sección de **Acciones para ejecutar después** damos clic en **Añadir una acción** y escogemos la opción de **Publish HTML reports** y **Publish Thucydides Test Reports**

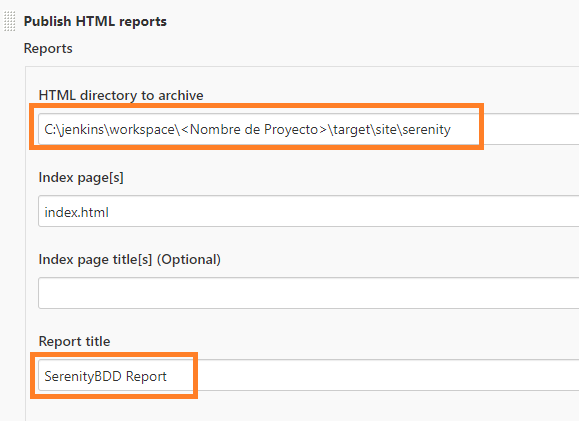
****



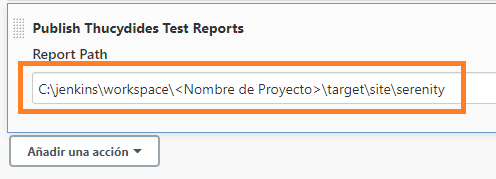
**Paso 15.2:** En **Publish HTML reports** damos clic en el botón **Añadir**

****

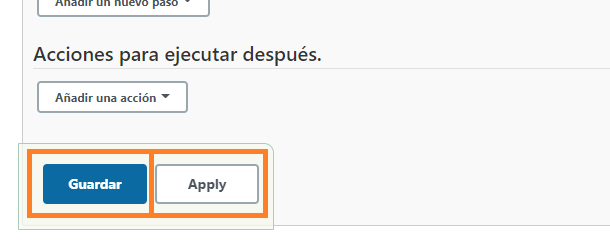
**Paso 15.3:** En **Publish HTML reports** ingresamos la dirección de donde se encuentra nuestro proyecto en nuestra maquina local y le agregamos al final **\target\site\serenity** en caso tal que sea en un servidor en la nube seria **/var/lib/jenkins/workspace/<Nombre de Proyecto> /target/site/serenity** y también agregamos en la opción **de Report title** el título que deseemos para nuestro reporte



**Paso 15.4**: En **Publish Thucydides Test Reports** solo ingresamos la ruta de nuestro proyecto que ingresamos en **Publish HTML reports**



**Paso 15:** Damos clic al botón **Apply** y luego al botón **Guardar**

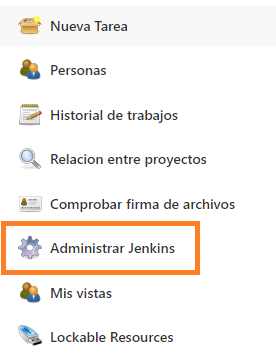
****

### Ejecutar comando para organizar el reporte de SerenityBDD

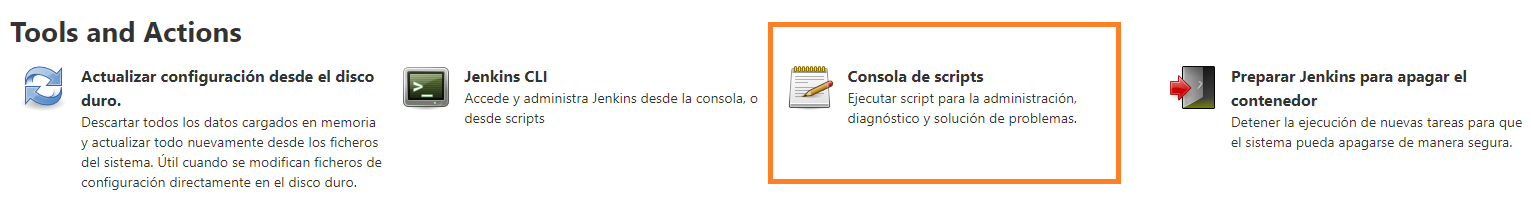
**Paso 1:** Damos clic en el logo de **Jenkins** en la parte superior izquierda



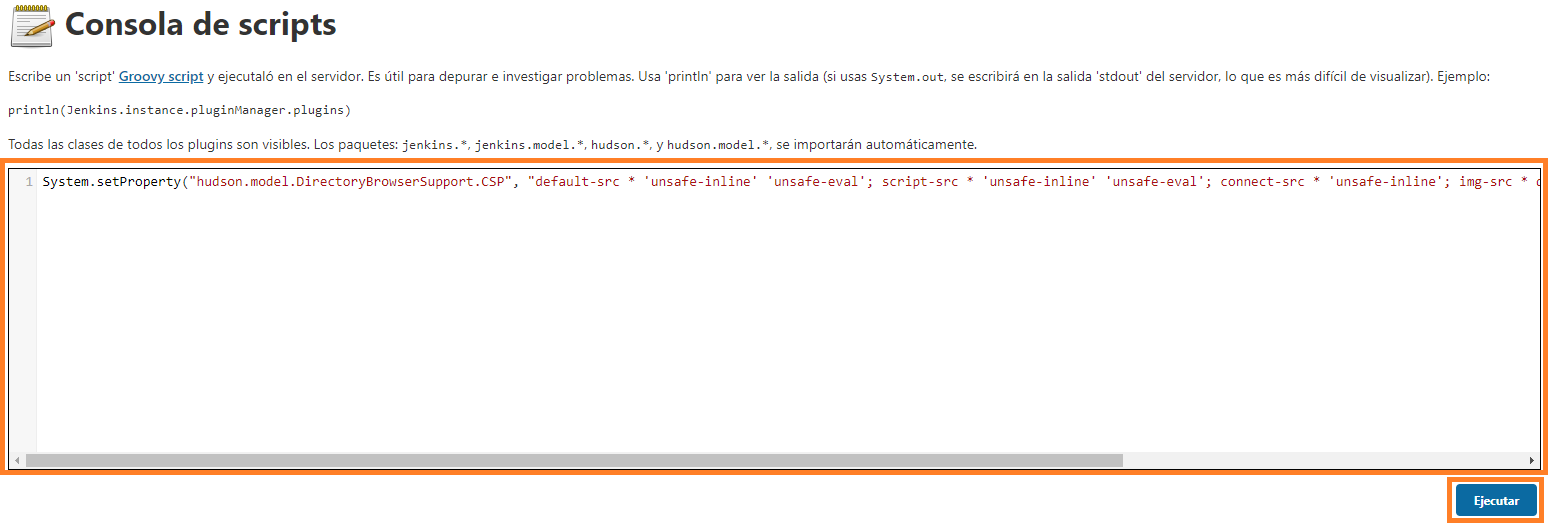
**Paso 2:** Damos clic en **Administrar Jenkins**



**Paso 3:** Bajamos hasta **Tools and Actions** y damos clic en **Consola de scripts**



**Paso 4:** Ingresamos en la **Consola de scripts** el siguiente comando y damos clic en **Ejecutar** System.setProperty("hudson.model.DirectoryBrowserSupport.CSP", "default-src \* 'unsafe-inline' 'unsafe-eval'; script-src \* 'unsafe-inline' 'unsafe-eval'; connect-src \* 'unsafe-inline'; img-src \* data: blob: 'unsafe-inline'; frame-src \*; style-src \* 'unsafe-inline';")

****

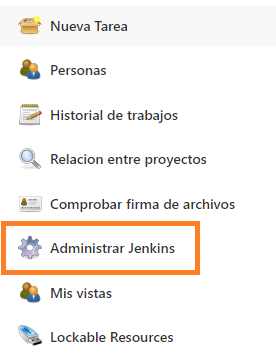
**Nota:** Este comando nos organiza el reporte de **SerenityBDD** de nuestro pipeline cuando termine la ejecución de la prueba

## Configuración de Agentes en Jenkins

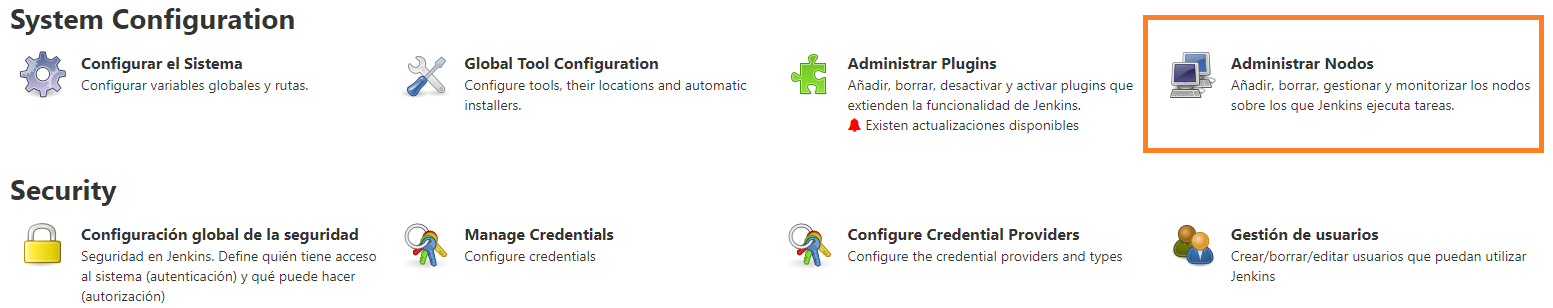
Los agentes de Jenkins son conectores que descargamos en nuestros computadores locales o servidores en la nube para conectar Jenkins con el dispositivo y realizar las pruebas en este.

Para configurar un agente debemos realizar los siguientes pasos:

**Paso 1:** Damos clic en **Administrar Jenkins**



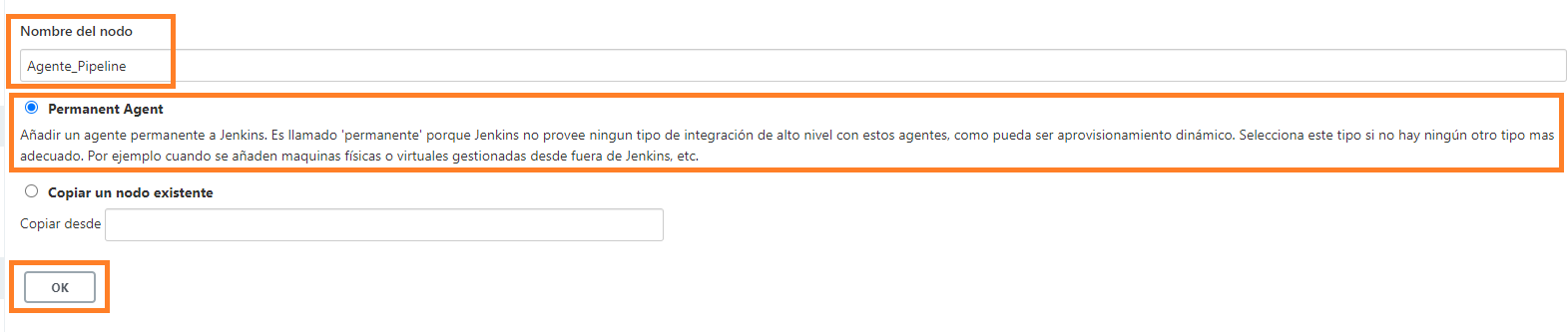
**Paso 2:** Nos ubicamos en **System Configuration** y damos clic en **Administrar Nodos**



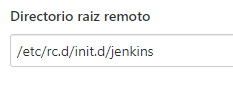
**Paso3:** Damos clic en **Nuevo Nodo**



**Paso 4:** Ingresamos el nombre a nuestro agente, damos clic en **Permanent Agent**  y damos clic en el botón **OK**

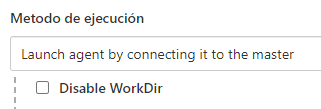
****

**Paso 5**: En el campo **Directorio raíz remoto** ingresamos la dirección de donde tenemos instalado **Jenkins** en este caso como es en un **servidor de AWS** ponemos la ruta donde se encuentra instalado Jenkins mediante **Putty** en caso de ser instalado en un computador local la ruta es donde se indique la instalación de Jenkins, normalmente queda en la ruta **C:\Program Files\Jenkins**

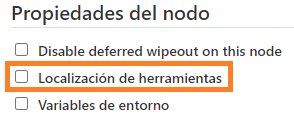


**Nota:** Normalmente cuando hacemos la instalación de Jenkins en un servidor en la nube la ruta donde queda instalado es la de la imagen sin importar el servidor

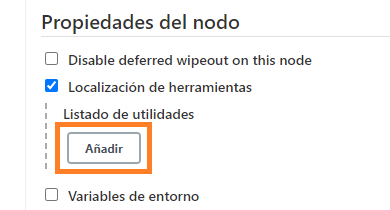
**Paso 6:** En el campo **Metodo de ejecución** escogemos la opción **Launxh agent by connecting it to the master**



**Paso 7:** En **Propiedades del nodo** seleccionamos la casilla **Localización de herramientas**

****

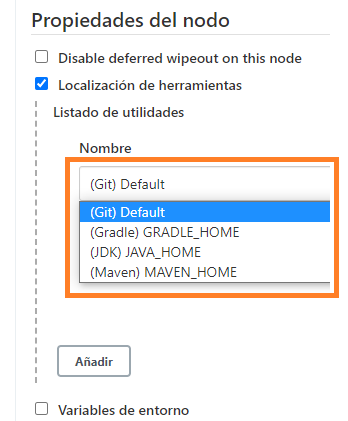
**Paso 8:** Damos clic en el botón **Añadir**

****

**Paso 9:** Nos aparecerá dos campos **Nombre** el cual contiene una lista de utilidades y **Directorio** un campo donde pondremos las direcciones de donde se encuentran las utilidades donde se ejecutara el agente



**Paso 10:** Al darle clic en el campo **Nombre** nos aparecerá una lista con las utilidades necesaria para una ejecución, al escoger una debemos ingresar la dirección de donde se encuentra instalada o ubicada la utilidad seleccionada. Para añadir cada utilidad debemos darle clic en el botón **Añadir** para que nos vuelva aparecer los campos para ingresarlas

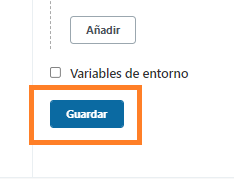


**Paso 11:** Al final debe verse como la siguiente imagen

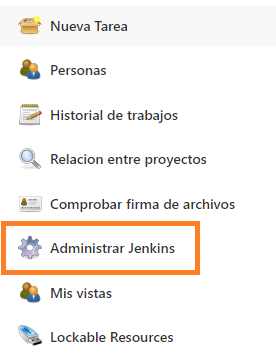


**Nota:** En este caso se está manejando para que el agente sea ejecutado en un computador local.

**Paso 12:** Damos clic en el botón **Guardar**



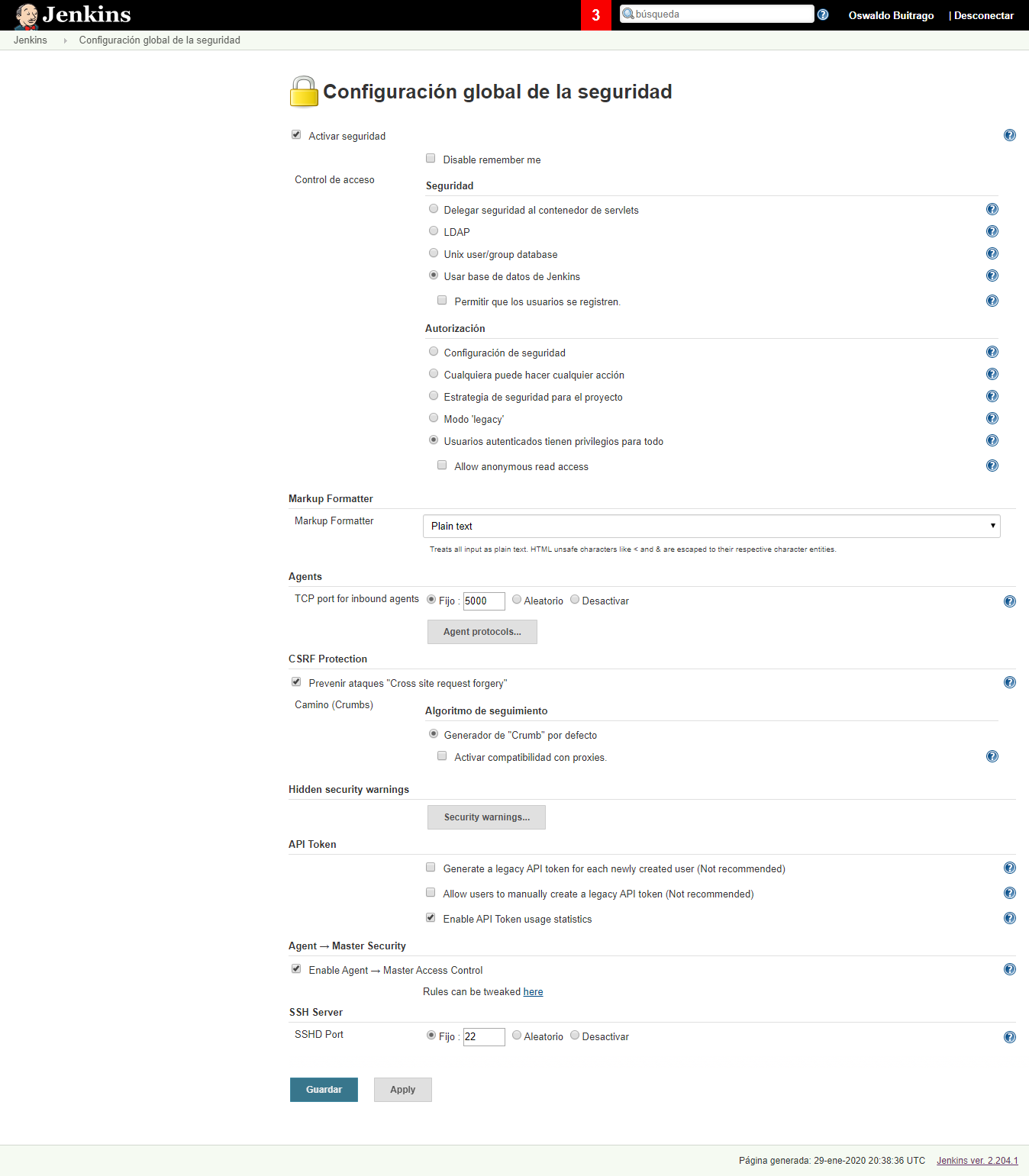
**Paso 13:** Damos clic en **Administrar Jenkins**



**Paso 14:** Damos clic en la opción **Configuración global de la seguridad**

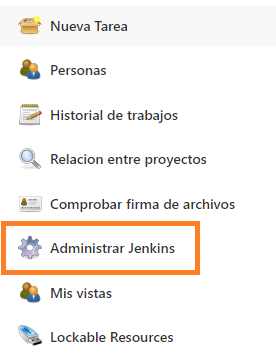


**Paso 15:** Cuando estemos en la **Configuración global de la seguridad** debemos configurar los campos tal y como se muestra en la siguiente imagen y damos clic al botón **Guardar** al final

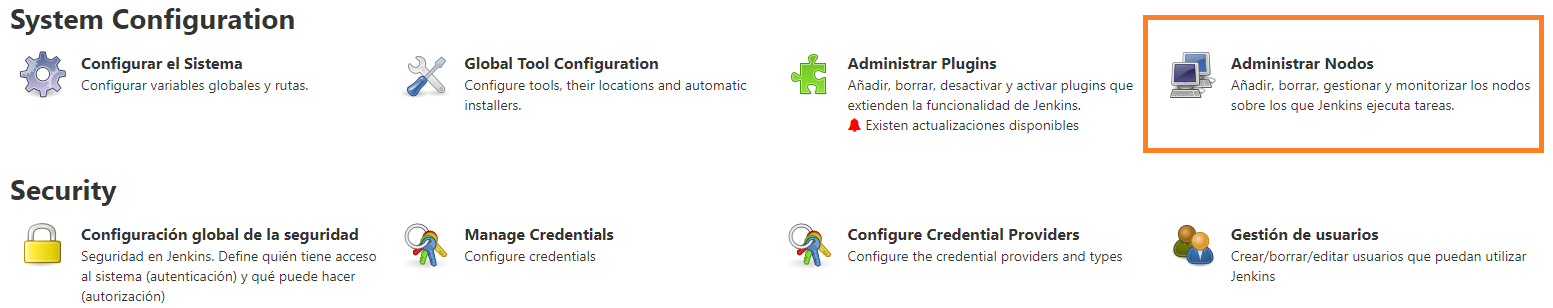


## Descargar Agente en dispositivo local

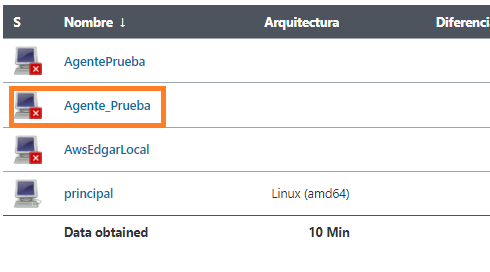
**Paso 1:** Damos clic en **Administrar Jenkins**



**Paso 2:** Nos ubicamos en **System Configuration** y damos clic en **Administrar Nodos**



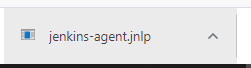
**Paso 3:** Damos clic al agente que creamos.



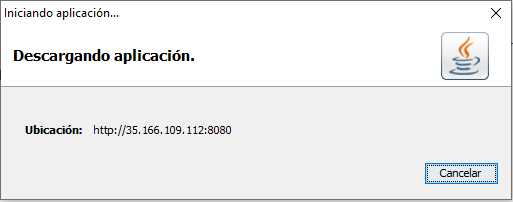
**Paso 4:** Damos clic al botón **Launch** para descargar el agente

****

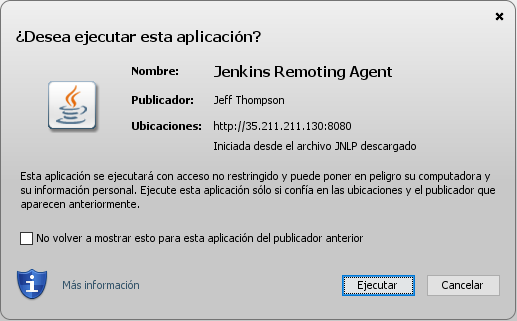
**Paso 5:** Damos clic en el ejecutable que descargamos



**Paso 6:** Dejamos que se inicialice



**Paso 7:** Damos clic en **Ejecutar**



**Paso 8:** Nos debe aparecer esta ventana que nos indica que ya está conectado nuestro agente



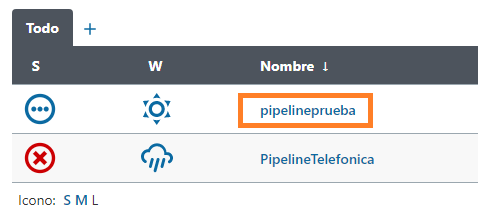
## Configuración de Agente con Pipeline

Una vez tenemos el Agente creado debemos asignarlo a un Pipeline para su ejecución. Para asociar un agente debemos realizar los siguientes pasos:

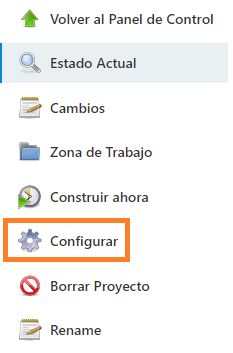
**Paso 1:** Damos clic en el logo de **Jenkins** en la parte superior izquierda



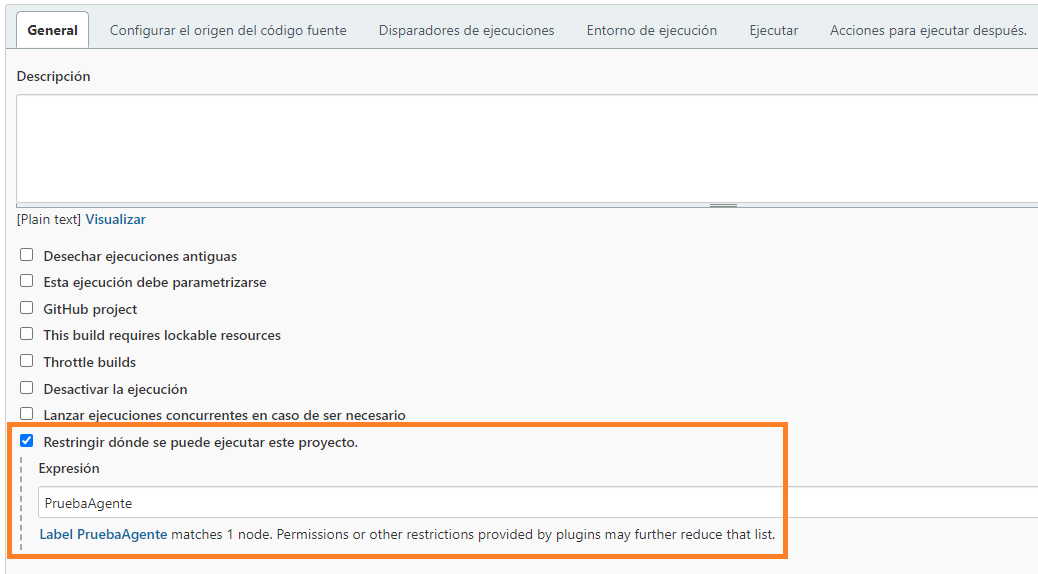
**Paso 2:** Damos clic al pipeline que queremos asignarle el agente. (No es necesario escoger el Pipeline que aparece en la imagen si no el que desee)



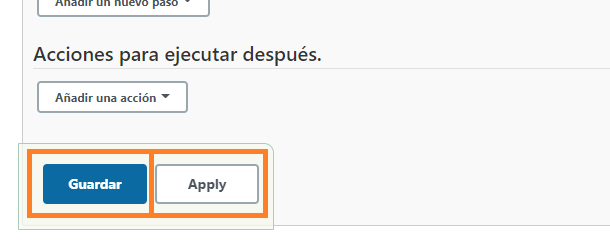
**Paso 3:** Damos clic en **Configurar**

****

**Paso 4:** En la pestaña **General** seleccionamos la opción **Restringir donde se puede ejecutar este proyecto** y en el campo **Expresión** ingresamos el nombre que le hemos dado al agente

****

**Paso 5:** Damos clic al botón **Apply** y luego al botón **Guardar**

****

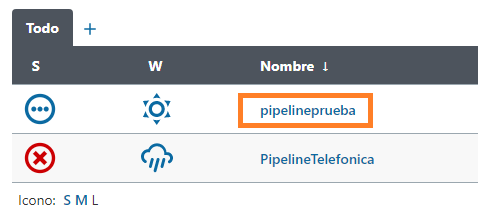
## Ejecutar un Pipeline

Para ejecutar un Pipeline ya sea en un servidor en la nube, en el mismo Jenkins o un computador local mediante un agente sin importar el modo se ejecuta de la misma forma. Para ejecutar un Pipeline debemos realizar los siguientes pasos:

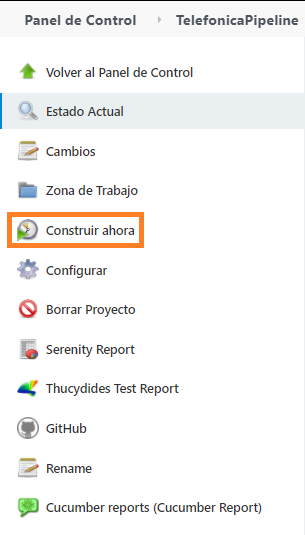
**Paso 1:** Damos clic en el logo de **Jenkins** en la parte superior izquierda



**Paso 2:** Damos clic al pipeline que queremos ejecutar. (No es necesario escoger el Pipeline que aparece en la imagen si no el que desee)



**Paso 3:** Damos clic en Construir ahora

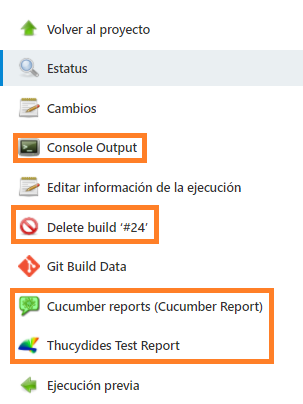
****

En este momento el pipeline está ejecutando pero si quisiéramos ver el log de la ejecución debemos hacerlos los siguientes pasos:

**Paso 1:** En la parte inferior izquierda nos encontraremos con el listado de ejecuciones que hemos realizado y al darle clic a cualquiera podremos ver el resultado de las ejecuciones.



**Paso 2:** Una vez le demos clic a alguna ejecución podremos ver nuevas opciones al lado izquierdo, la cuales el **Console Output** permite ver en consola de Jenkins la ejecución de nuestro Pipeline, **Delete build** eliminar la ejecución el cual hemos ingresado y **Cucumber Report** y **Thucydides Test Report** permiten ver los reportes de Cucumber y SerenityBDD

****