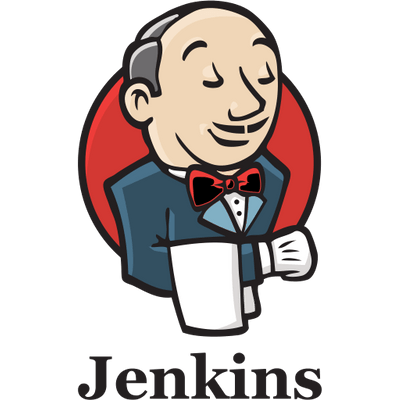


Índice

1. Introducción…………………………………………………………………….......pág 1
2. Jenkins…………………………………………………………………………………..pág 2
   1. Instalación de Jenkins…………………………………………………..pág 4
3. Git………………………………………………………………………………………….pág 12
   1. Github………………………………………………………………………….pág 13
4. Instalación de Git plugin en Jenkins…………………………..………..…pág 14
5. Enlace Jenkins con Git………………………………………………….……..…pág 16
6. Crear trabajo de Jenkins……………………………..………………………...pág 22
7. Crontab………………………………………………………………………….………pág 28
8. Bibliografía……………………………………………………………………..……..pág 32

1. Introducción



En este proyecto vamos a aprender a montar un servidor con Jenkins y a crear trabajos en este.



También usaremos Git y GitHub, creando repositorios locales y remotos, aprendiendo a cómo usarlos.

Al final del proyecto conseguiremos crear un trabajo en el servidor de Jenkins que sincronice el repositorio de GitHub, para hacer un clonado cada vez que se ejecute la tarea a la carpeta de trabajo creada anteriormente por Jenkins y, ejecutar tareas añadidas en un script subido al repositorio.



Después usaremos Crontab para programar una tarea que copie el archivo creado por el script al escritorio para que sea más accesible para el usuario.

Realizaremos el proyecto usando una máquina virtualizada con VirtualBox, que tendrá 2GB de RAM y 1 núcleo de procesador.

El Sistema Operativo que vamos a usar es Ubuntu 18.04

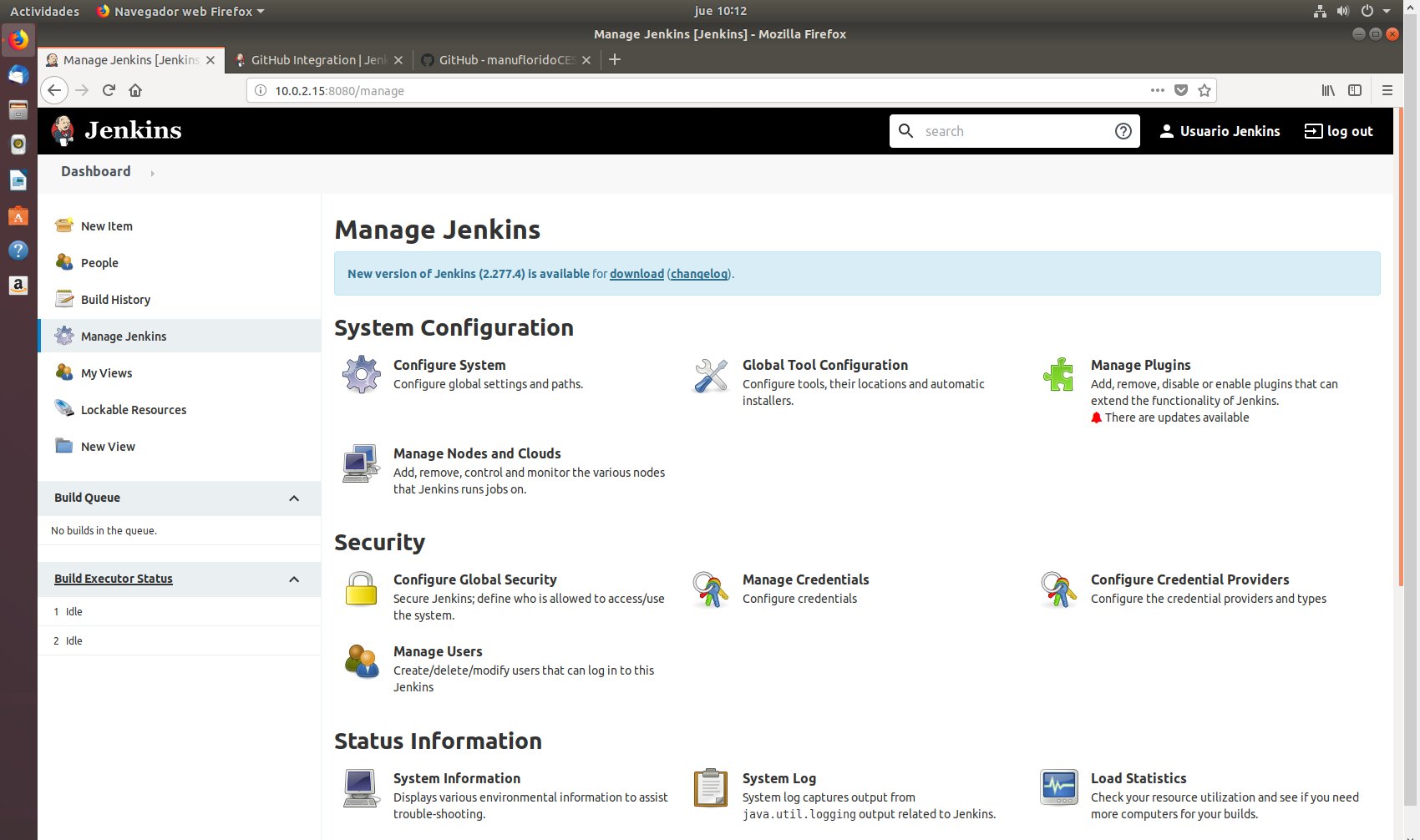
2. Jenkins

Jenkins es un servidor de integración continua, esto significa que es el encargado de actualizar y ejecutar pruebas de código periódicamente.

Sus objetivos son:

* Encontrar y arreglar errores de código.
* Mejorar la calidad del software.
* Reducir el tiempo y el esfuerzo que supone actualizar el software periódicamente.

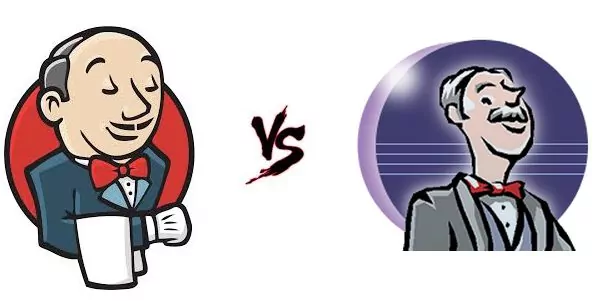
Dispone de una interfaz web con la que configurar todos los ajustes disponibles.



Admite el uso de herramientas de control de versiones como Subversion, GIT, Mercurial… Y además tiene la posibilidad de instalar una infinitud de plugins.

Originalmente fue desarrollado con el nombre de *“Hudson”*, en verano de 2004, pero debido a ciertos desacuerdos con Oracle, la comunidad decidió cambiar el nombre a *“Jenkins”.*

Actualmente Jenkins y Hudson son proyectos independientes, aunque entre ellos no hay casi ninguna diferencia.



V

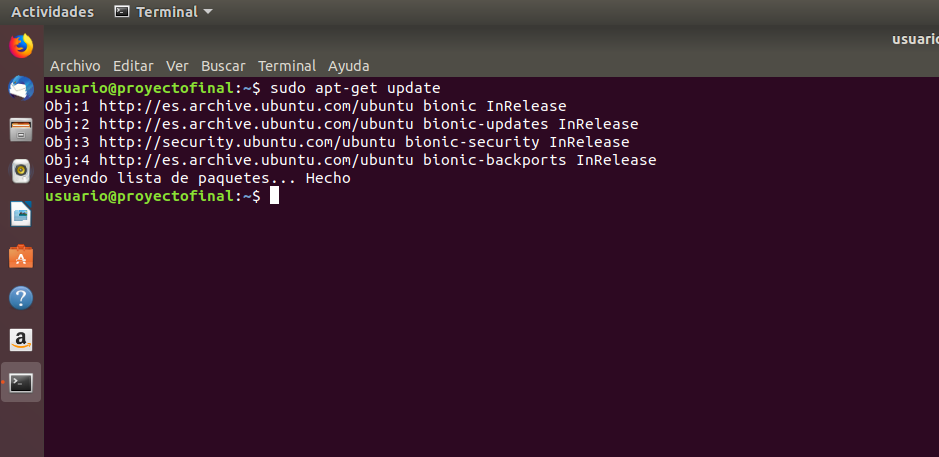
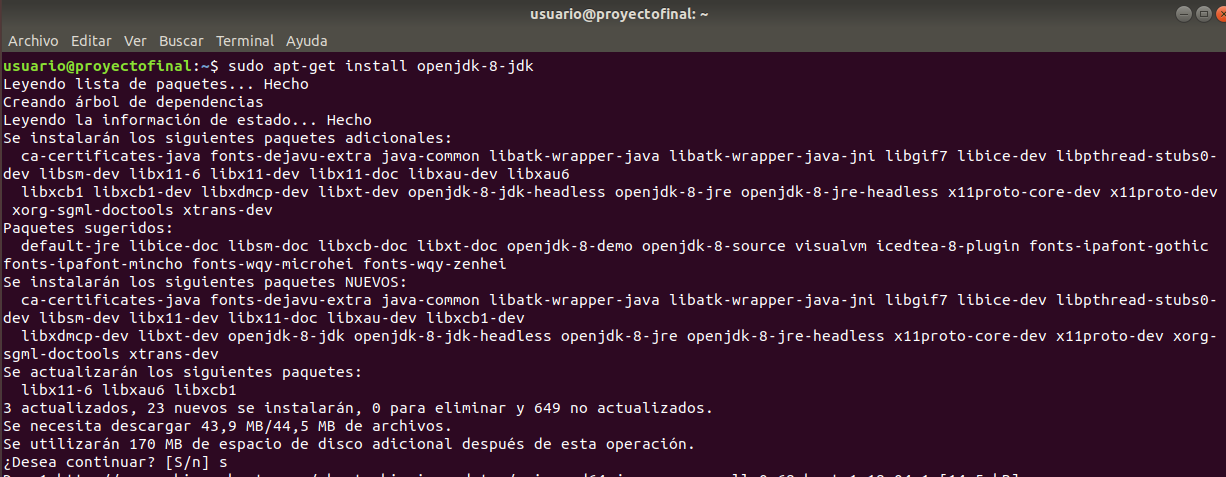
s

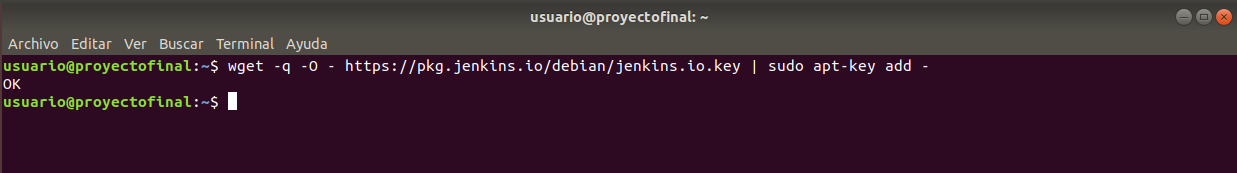
Hudson

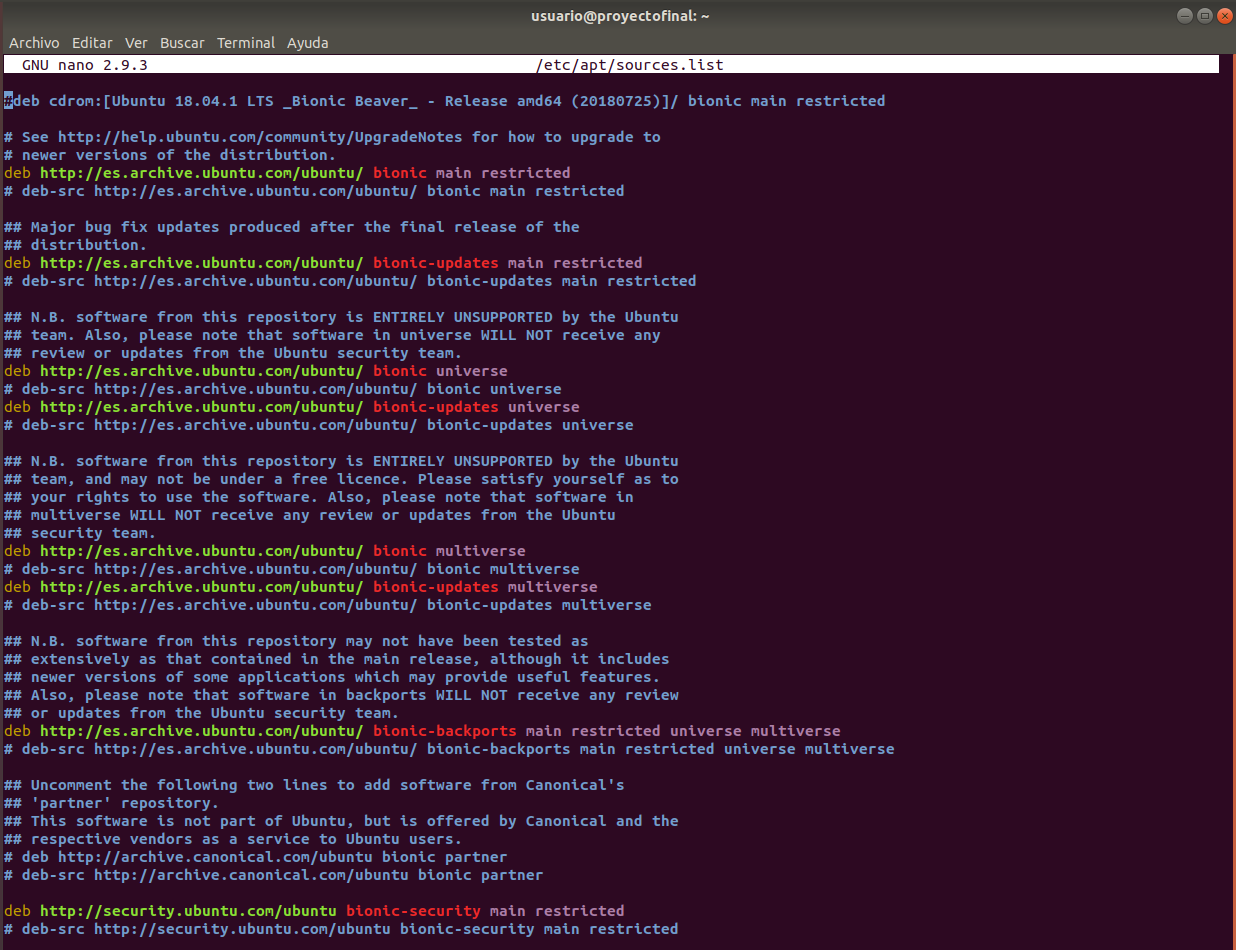
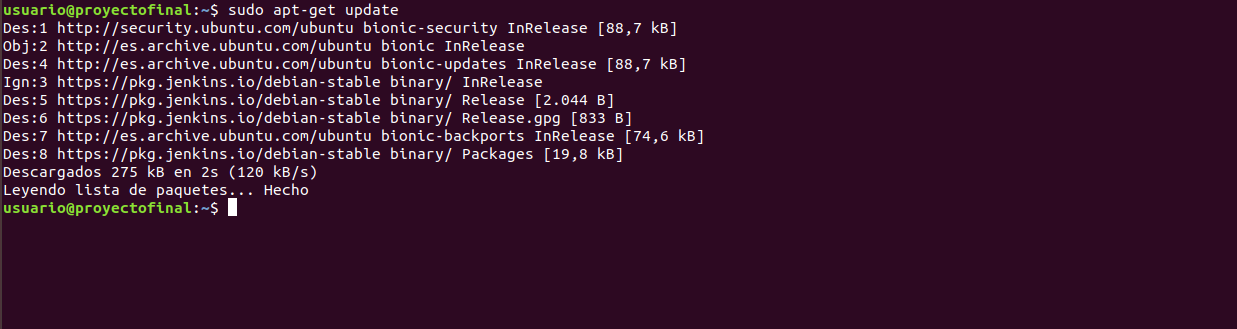
Jenkins

2.1. Instalación de Jenkins

¿Qué pasos debemos de seguir para instalar Jenkins?

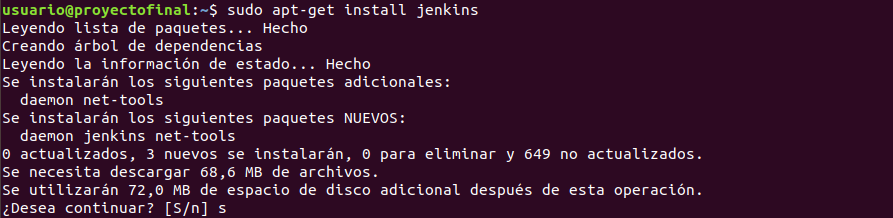
* El primer paso para la instalación de Jenkins es tener instalado Java. El primer comando es:
* ~$ sudo apt-get update.
* Después de actualizar la lista de paquetes, instalamos Java:
* ~$ sudo apt-get install openjdk-8-jdk.
* Esperaremos a que aparezca el mensaje “done.” que confirma la instalación del paquete:
* Añadimos la llave de Jenkins para que nos acepte el repositorio más tarde:
* ~$ wget –q –O - https://pkg.jenkins.io/debian/jenkins.io.key | sudo apt-key add –

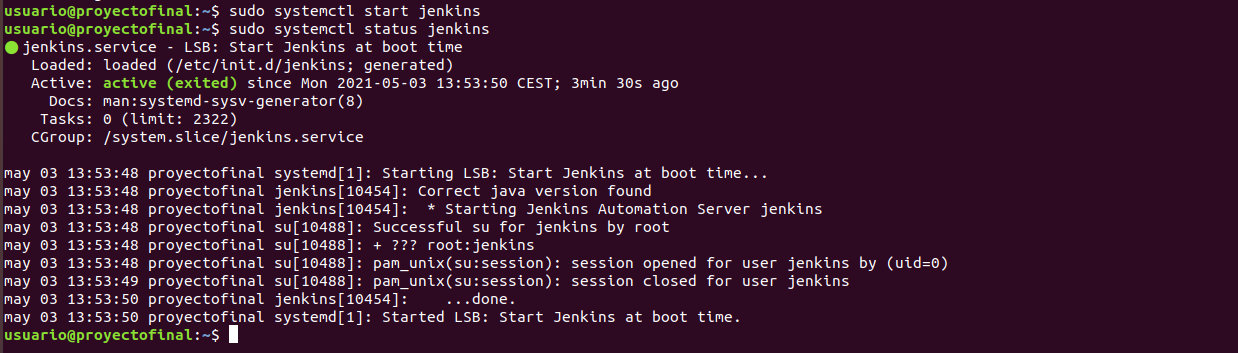


* Accedemos al fichero “sources.list” y añadimos el repositorio de Jenkins:
* ~$ sudo nano /etc/apt/sources.list
* Añadimos “deb [https://pkg.jenkins.io/debian binary/](https://pkg.jenkins.io/debian%20binary/)” y guardamos el archivo (Ctrl+o, Intro, Ctrl+x)
* Volvemos a actualizar la lista de paquetes

Como vemos, se ha actualizado la lista de paquetes disponibles con el repositorio de Jenkins.

* Procedemos a instalar Jenkins:
* ~$ sudo apt-get install jenkins



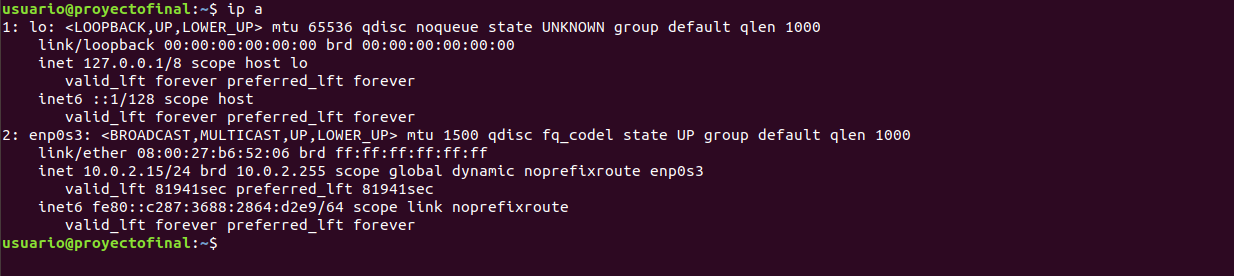
* Iniciamos el servicio de Jenkins y vemos su estado:
* ~$ sudo systemctl start jenkins
* ~$ sudo systemctl status jenkins
* Habilitamos el Firewall y abrimos el puerto 8080:
* ~$ sudo ufw enable
* ~$ sudo ufw allow 8080

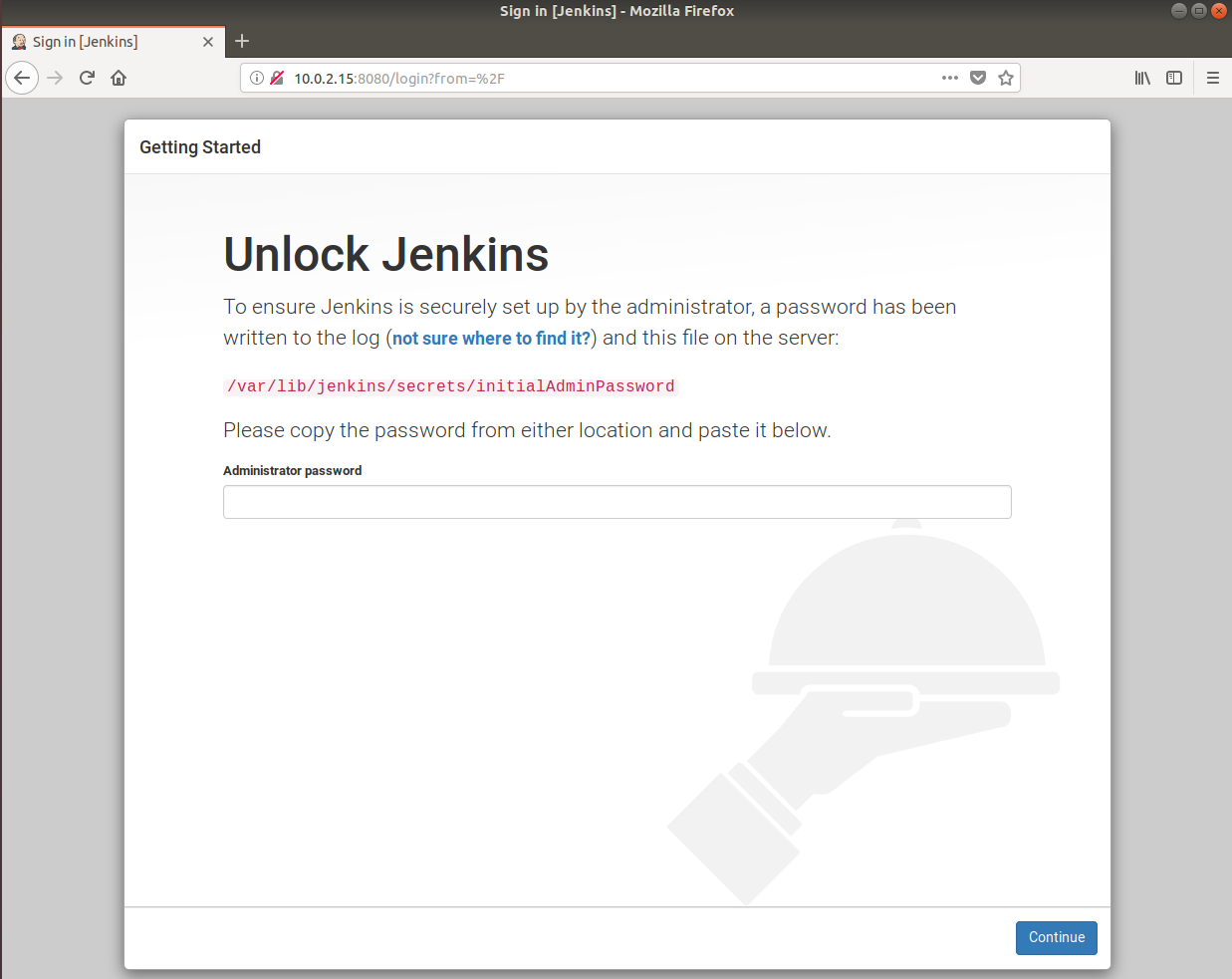


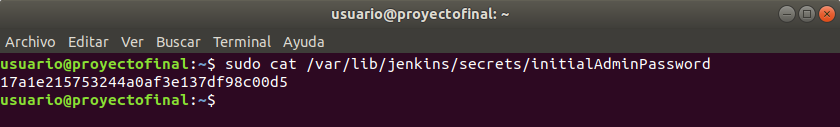
* ~$ sudo ufw status



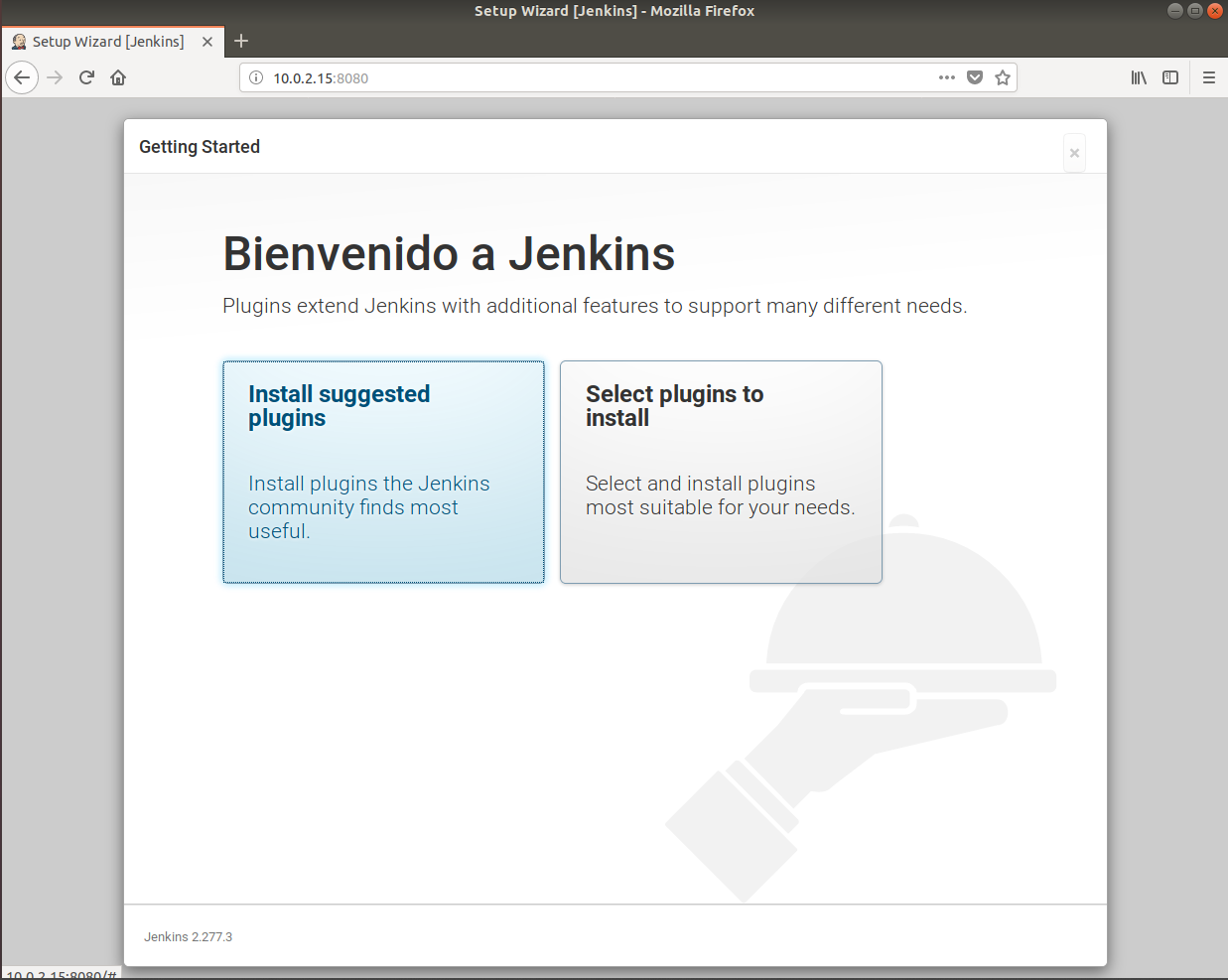
* Miramos la IP del equipo para acceder al panel de control de Jenkins:
* ~$ ip a



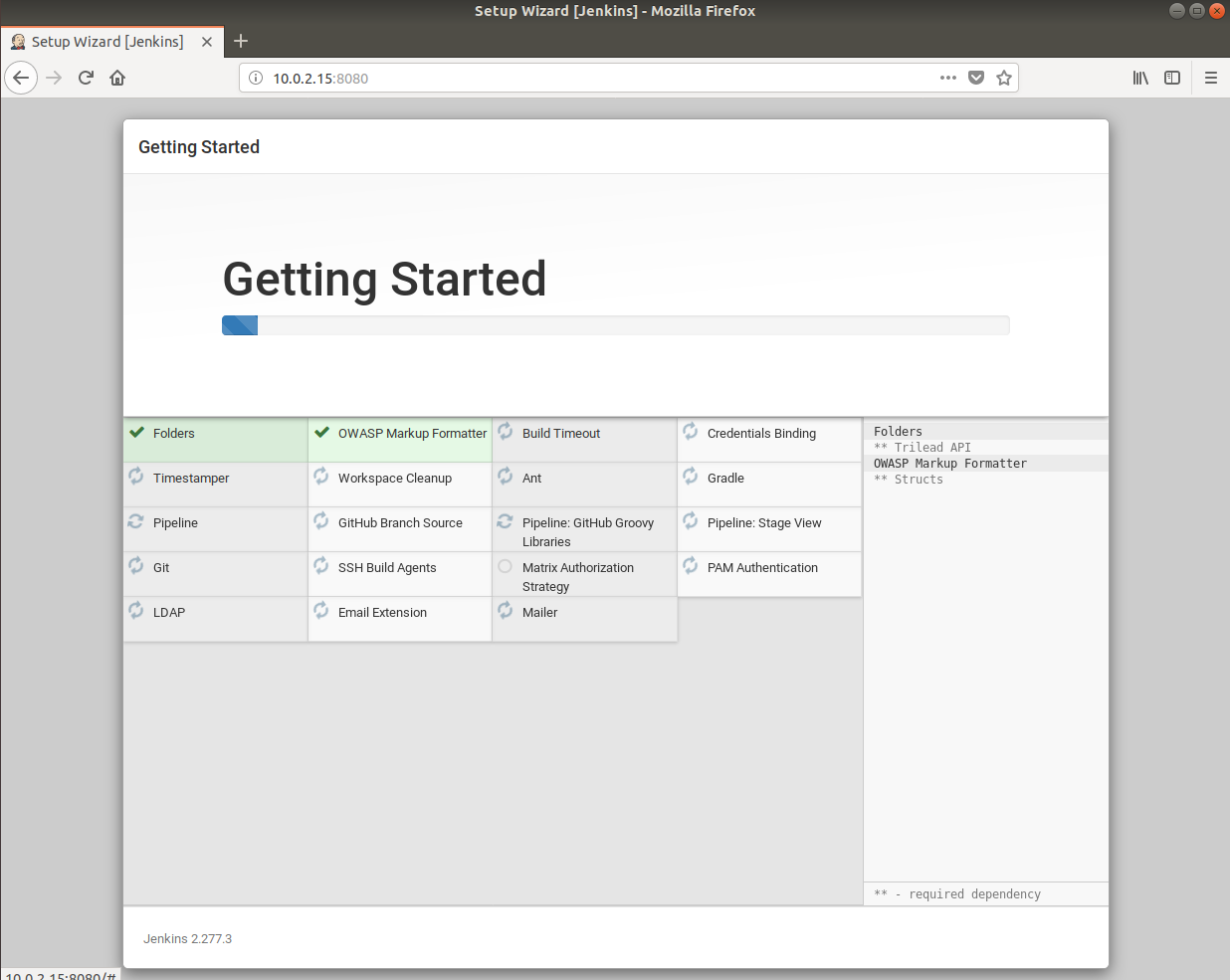
* Accedemos al navegador y ponemos la IP del equipo añadiendo el puerto:
* ~$ http://ip:8080
* Debemos buscar la contraseña en la ruta que nos indica Jenkins y pegarla en el navegador:
* ~$ sudo cat /var/lib/Jenkins/secrets/initialAdminPassword



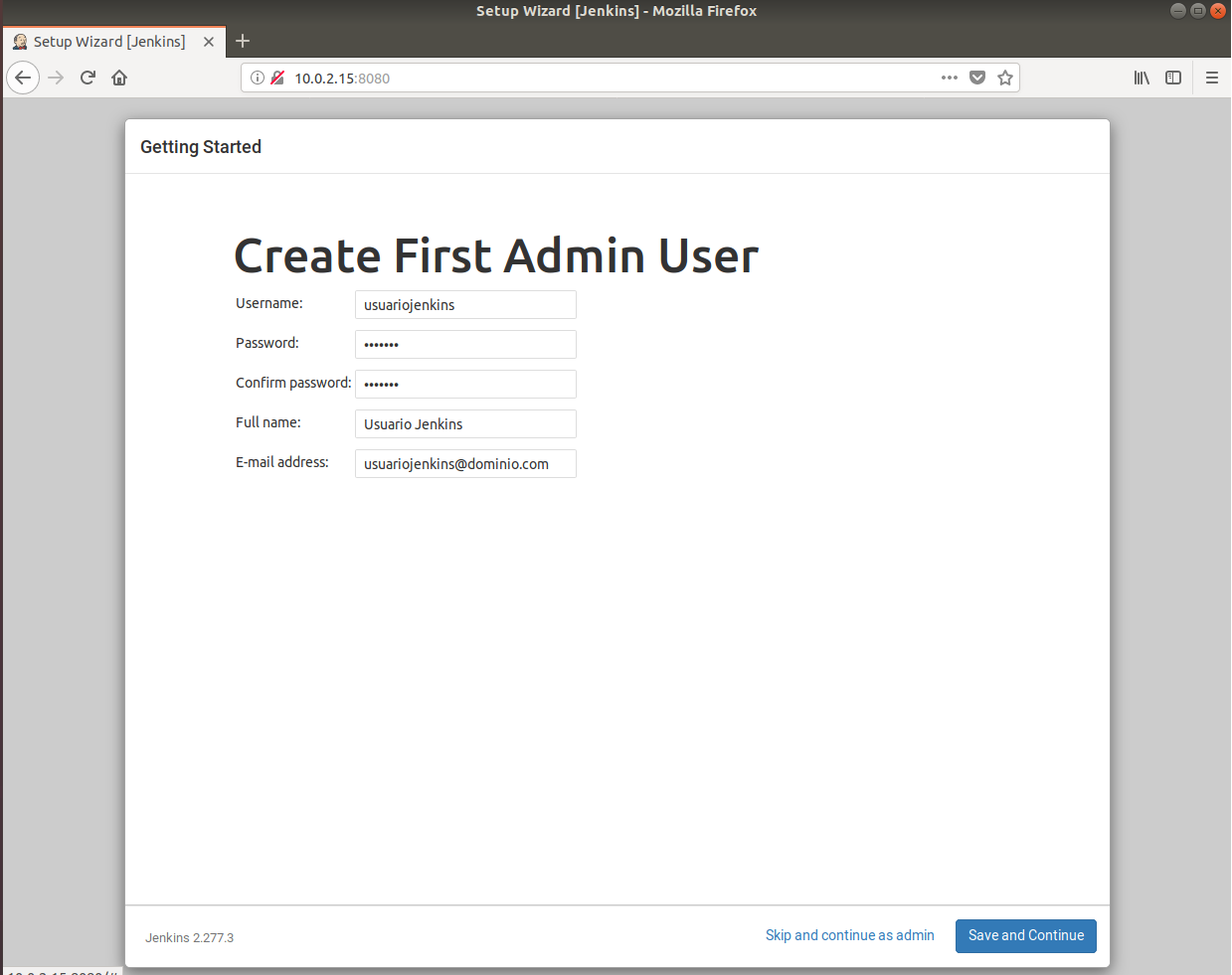
* Una vez que ponemos la contraseña, nos deja elegir que plugins queremos instalar:



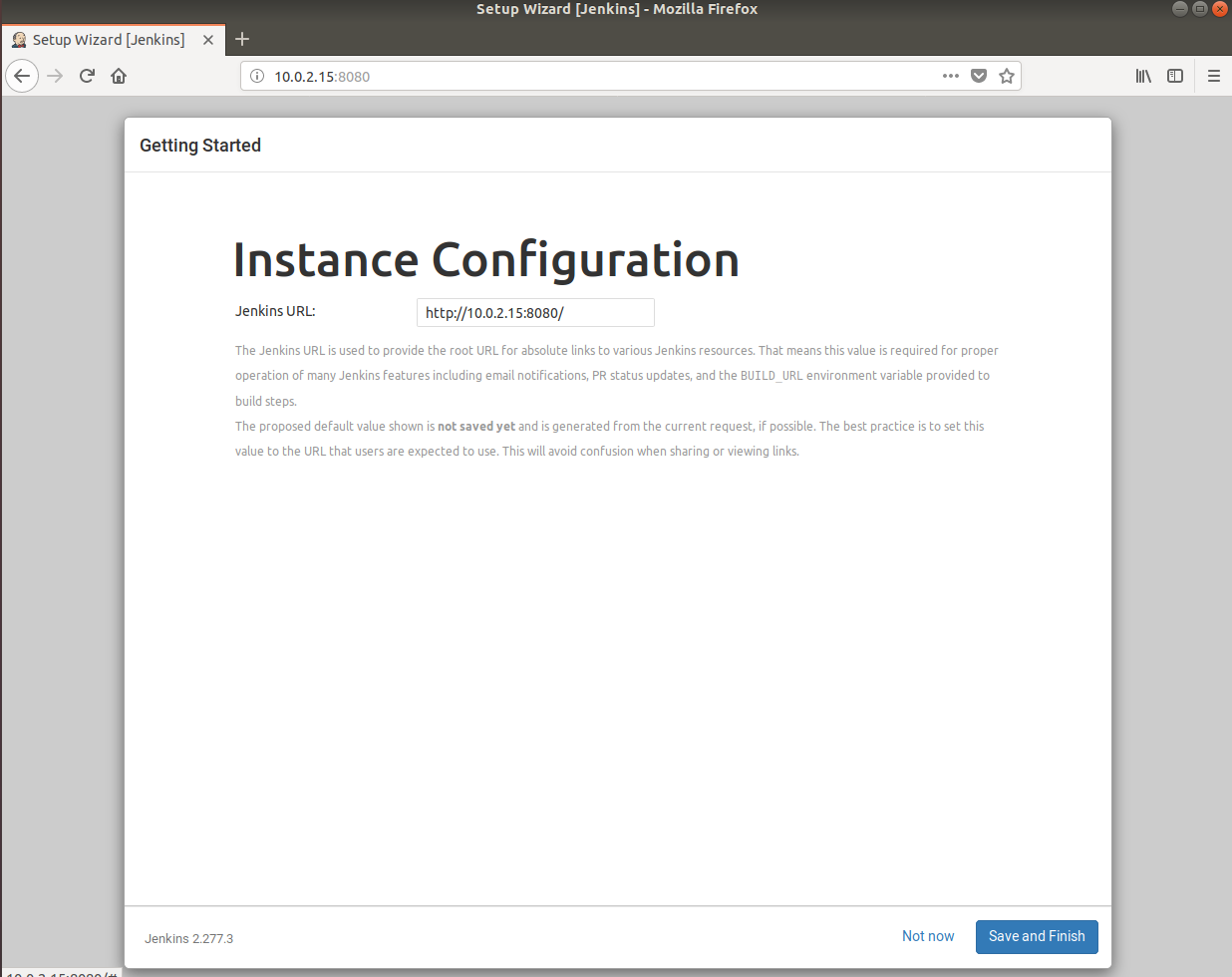
* Elegimos los plugins recomendados y dejamos que se instalen:



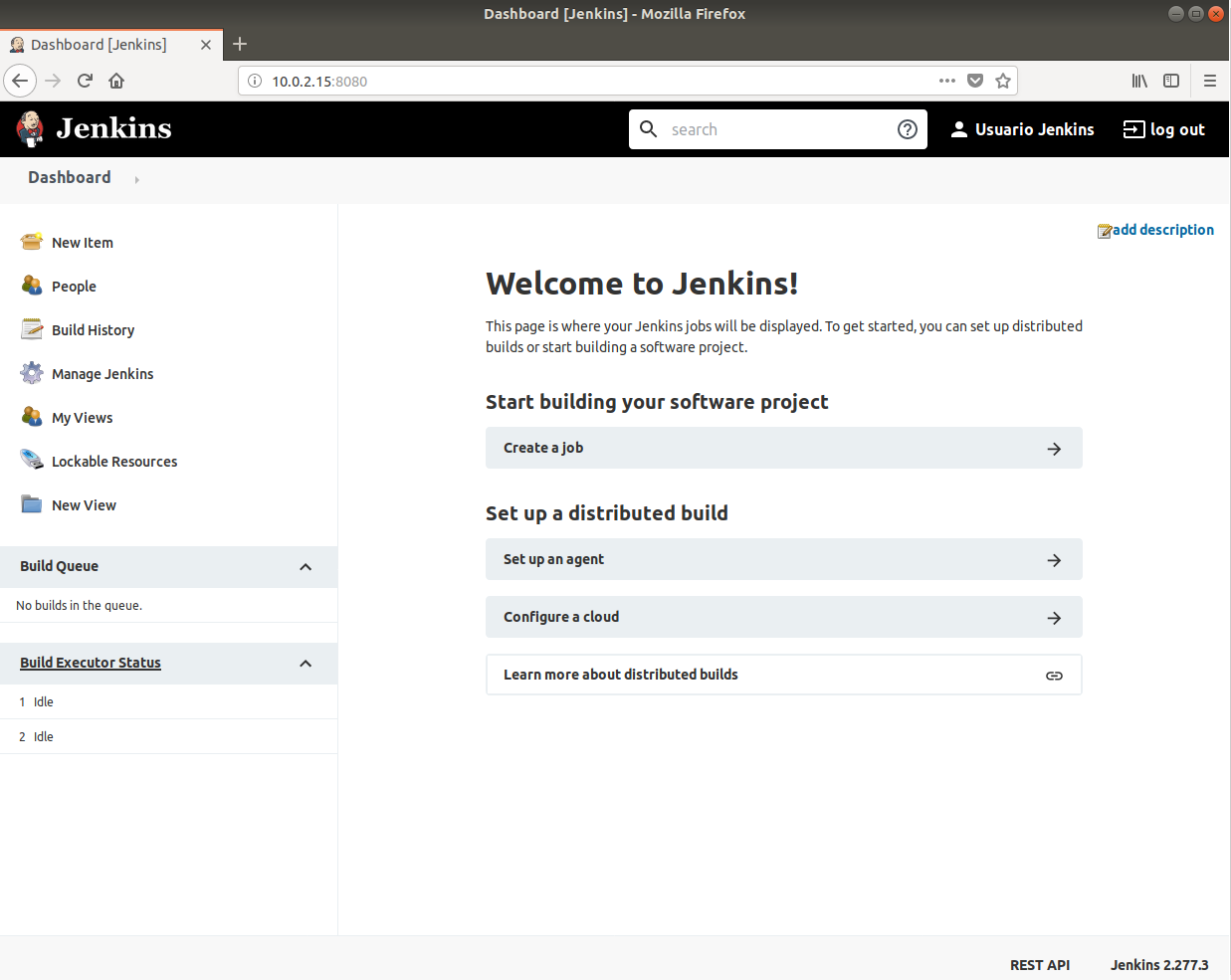
* Cuando se instalen, nos pedirá que creemos un usuario administrador:



* Nos permite elegir la URL que queremos asignar a Jenkins, en este caso dejaremos la que está por defecto:



* Una vez aceptado todo, podremos acceder al panel de control de Jenkins:



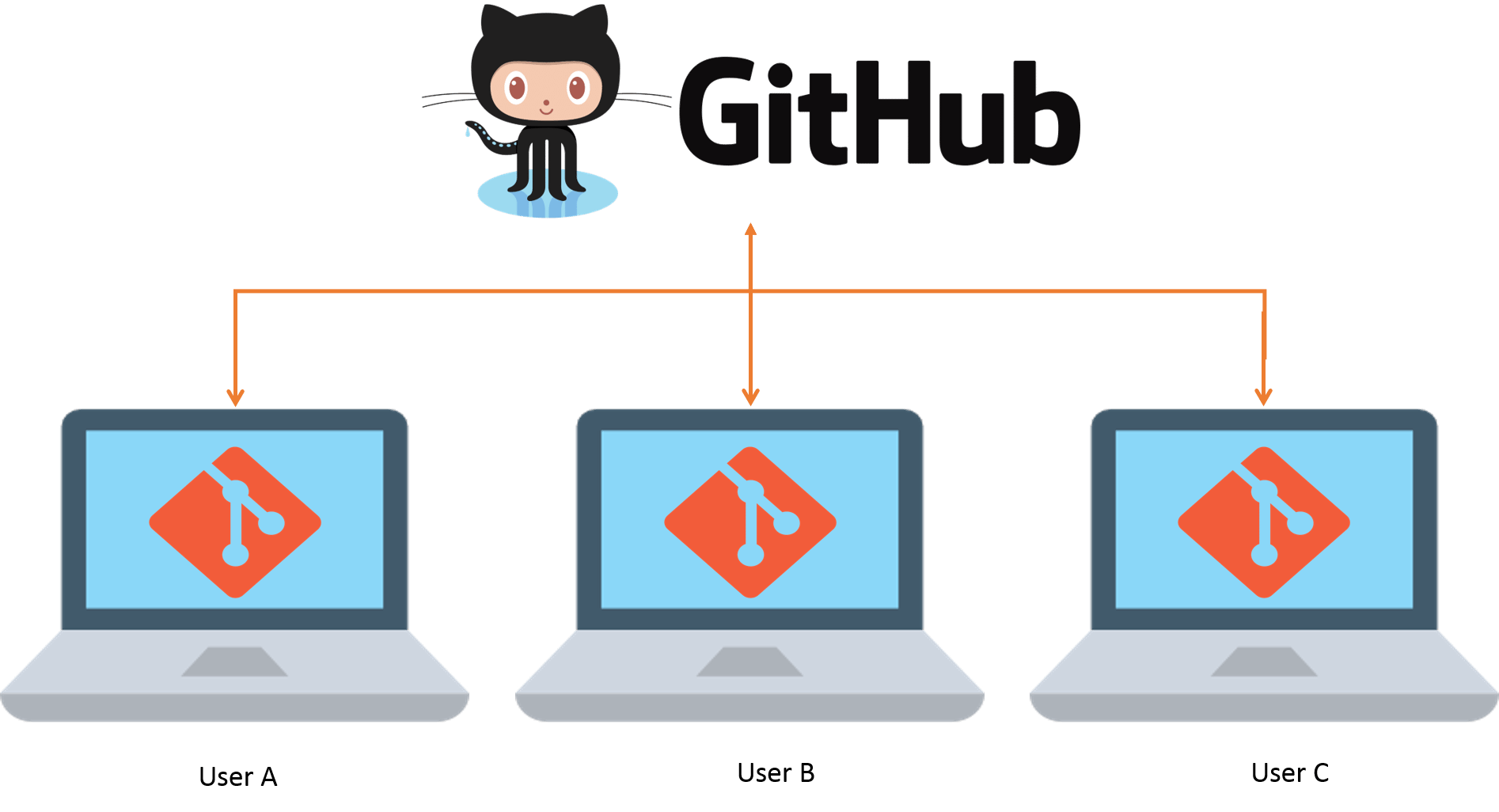
3. Git

Git es una herramienta de software libre que se encarga del control de versiones de forma distribuida.

Fue diseñada por Linus Torvalds y su lanzamiento fue en 2005.

Actualmente la supervisa Junio Hamano.





Su propósito es llevar un registro de todos los cambios en los archivos locales del ordenador, añadirlos al servidor remoto y poder coordinar el trabajo con varias personas, incluso trabajar a la vez.

3.1. Github



Github es una plataforma de gestión y organización basada en la nube que integra todas las funciones de Git.

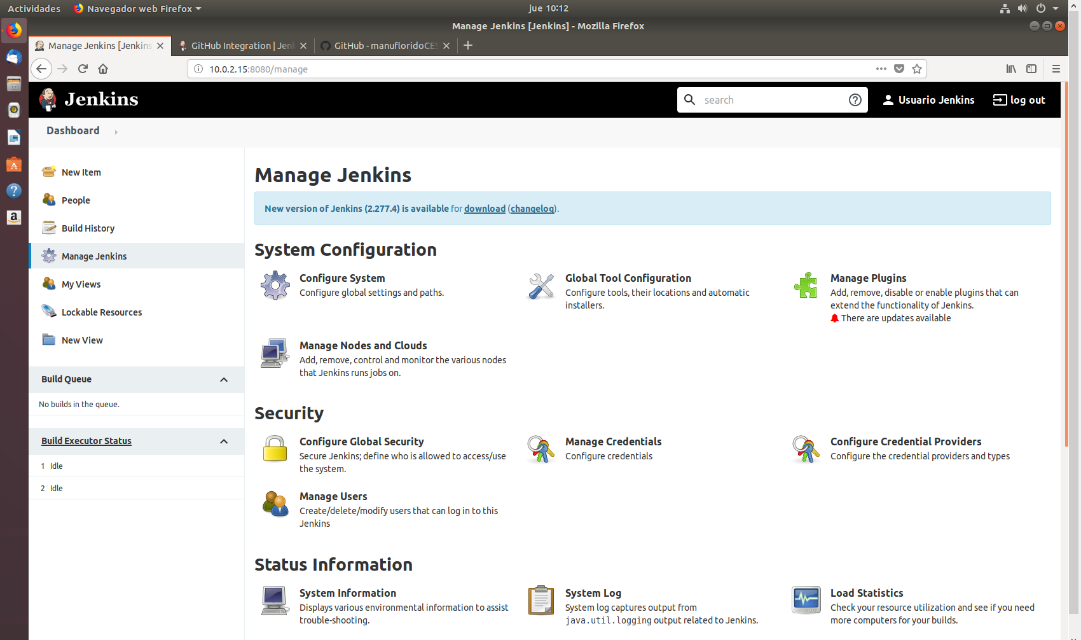
Todos los usuarios pueden ver y gestionar cambios desde la interfaz gráfica que nos proporciona.

Esta interfaz facilita su uso para personas con pocos conocimientos.

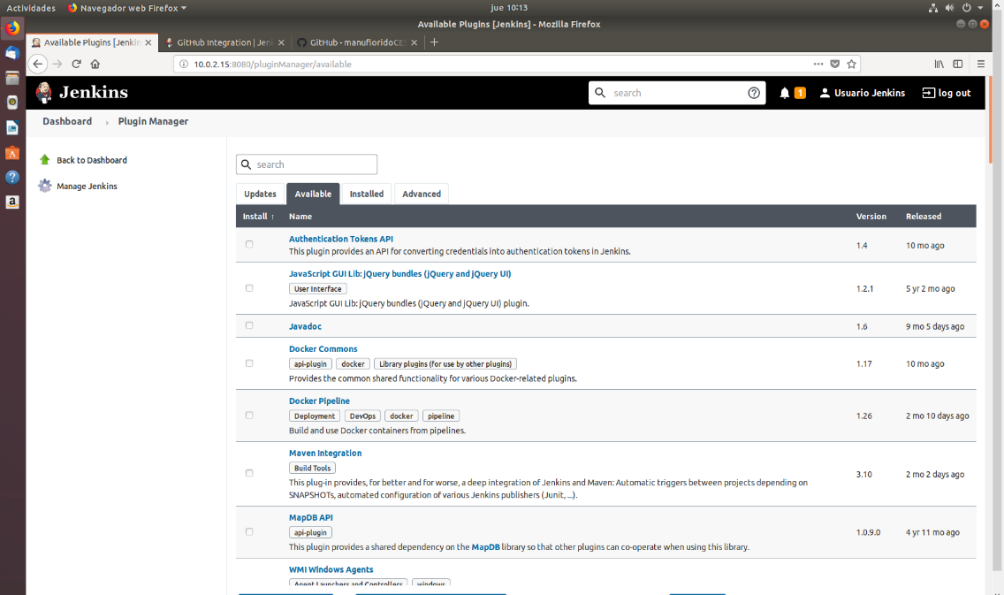


4. Instalación de Git plugin en Jenkins

* Pulsamos en Manage Jenkins y después en Manage Plugins para instalar el plugin de Git:



* Pulsamos en Available (Disponible) y buscamos el plugin llamado Git plugin:

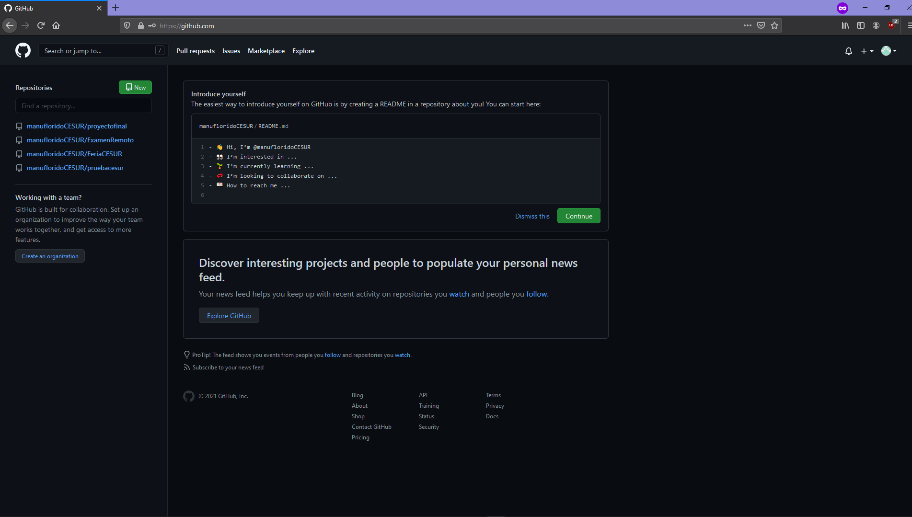




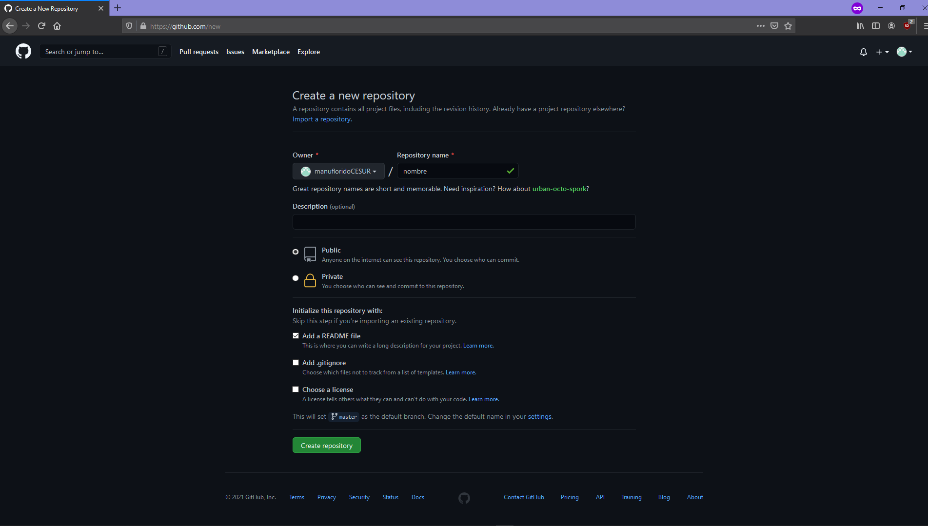
* A continuación, abriremos un terminal e instalaremos Git en nuestro equipo:
* ~$ sudo apt-get install git
* Una vez realizados estos pasos, podremos enlazar Jenkins y Git.

5. Enlace Jenkins con Git

* Una vez realizados estos pasos, pasamos a GitHub. Debemos iniciar sesión y crear un repositorio:

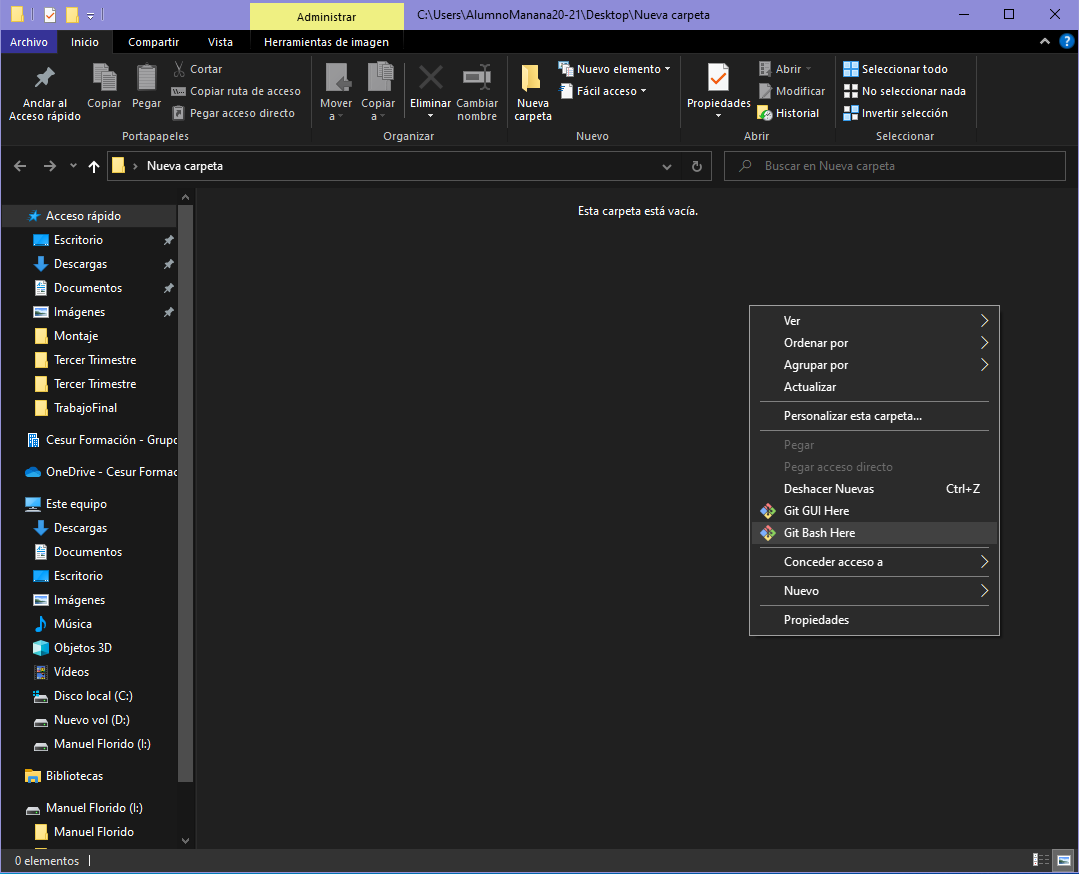


* Pulsamos en New para crear el repositorio, le ponemos un nombre y lo ponemos en público:

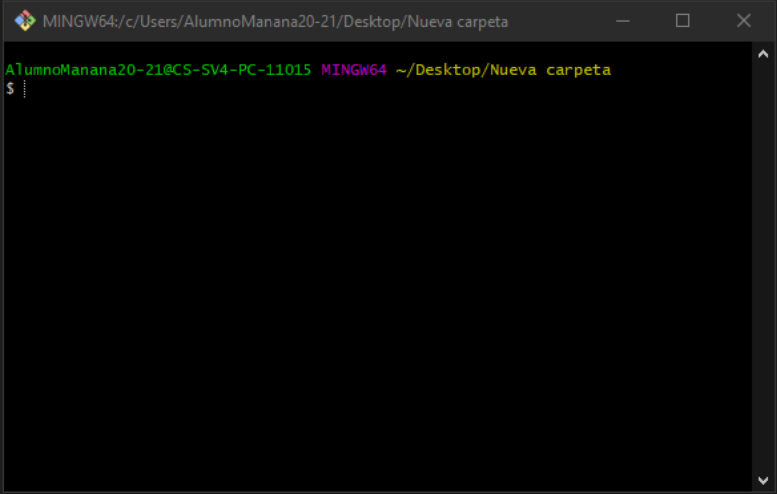


* Cuando lo hayamos creado, pasaremos a subir archivos al repositorio para usarlo luego con Jenkins.

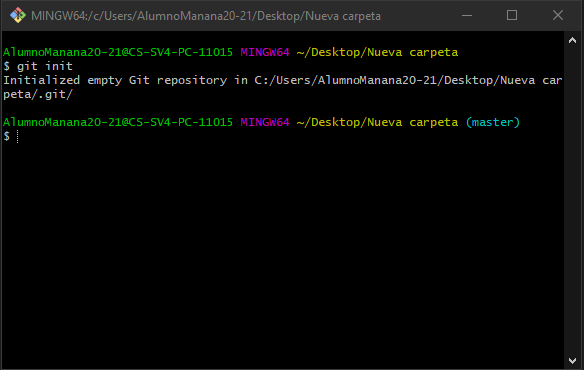
Crearemos una carpeta desde el equipo Windows (Git instalado), dentro de la carpeta haremos click derecho y pulsamos “Git Bash Here”:



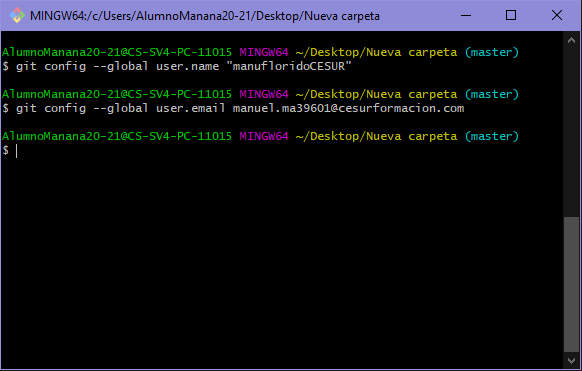
* Se abrirá una ventana como esta:

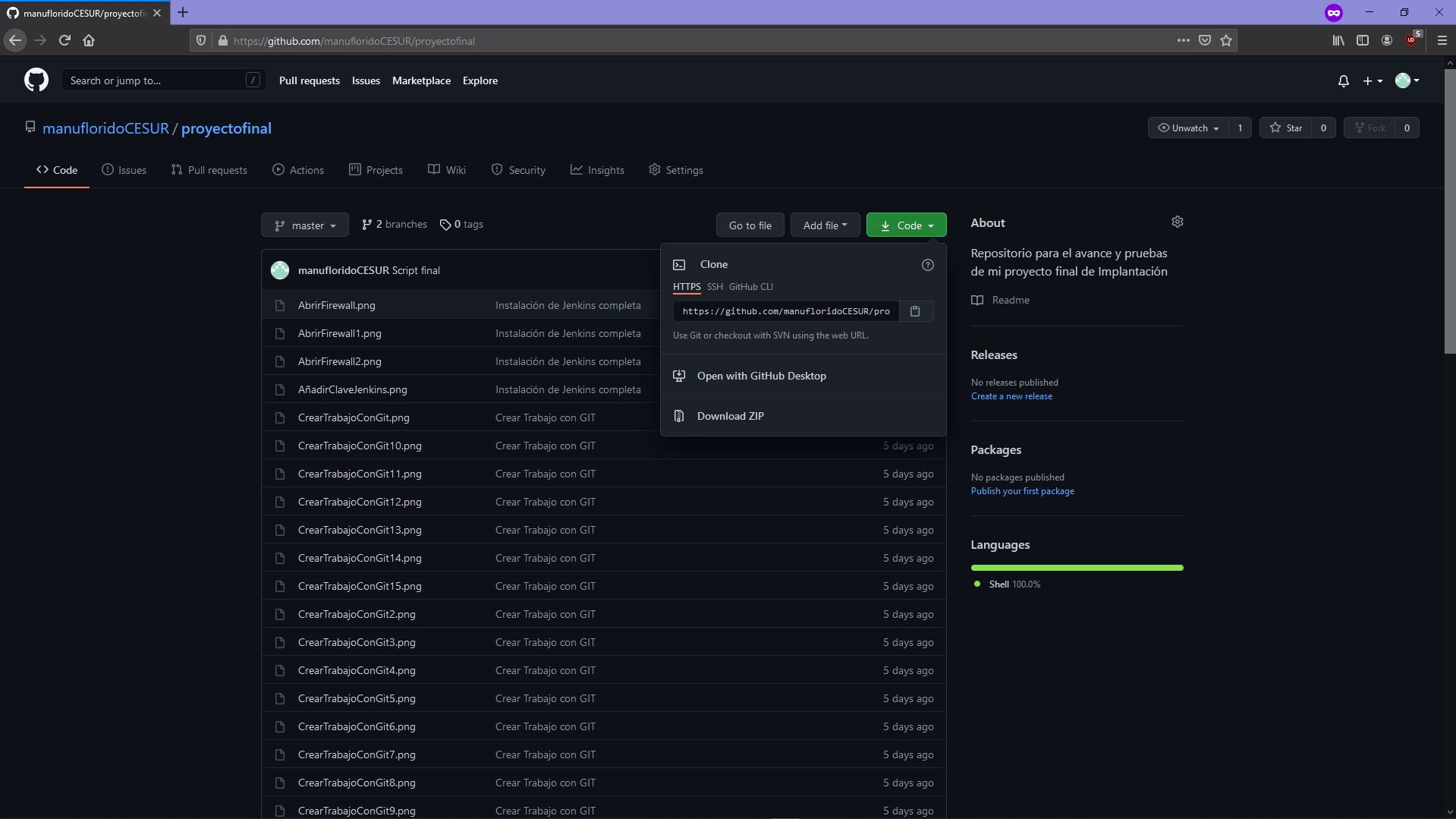


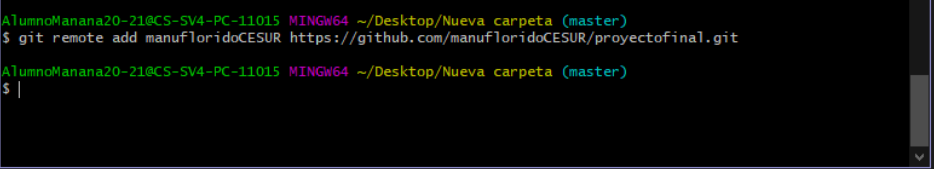
* Inicializamos el repositorio:
* ~$ git init

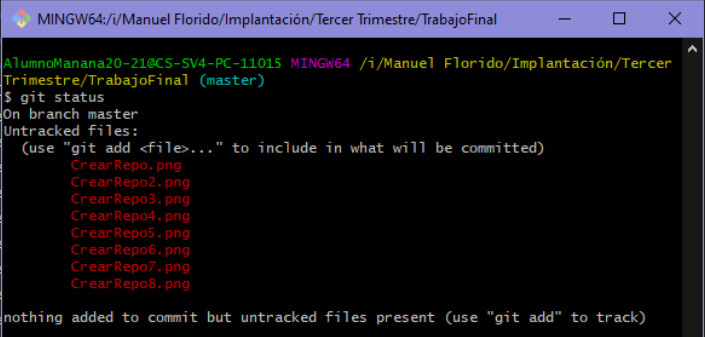


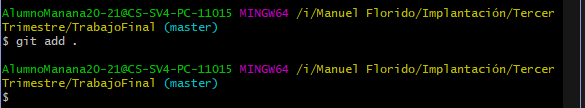
* Después configuraremos el usuario y el email de Git:
* ~$ git config –global user.name “nombre”
* ~$ git config –global user.email [correo@correo.com](mailto:correo@correo.com)



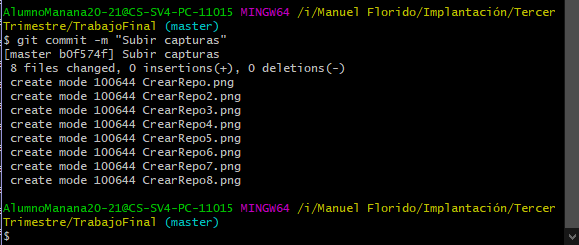
* Añadimos el repositorio de GitHub, cogiendo la URL del repositorio:
* ~$ git remote add usuario URL



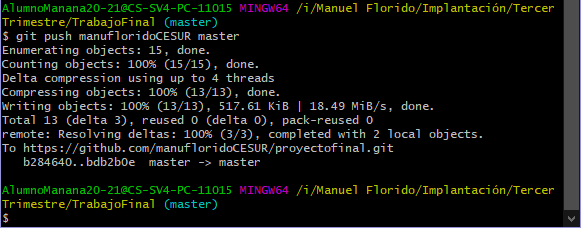
* Miramos el estado del repositorio para saber si se pueden añadir archivos:
* ~$ git status
* Añadimos los archivos para posteriormente confirmarlos:
* ~$ git add . *(El punto añade todos los archivos que tienen la posibilidad de añadirse)*

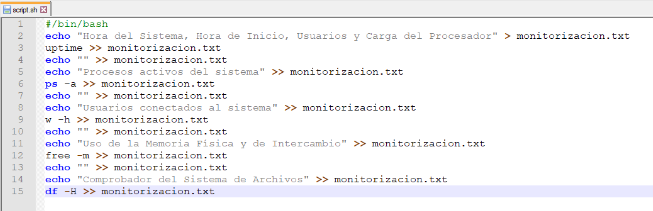


* Confirmamos los archivos que hemos añadido:
* ~$ git commit -m “Comentario”



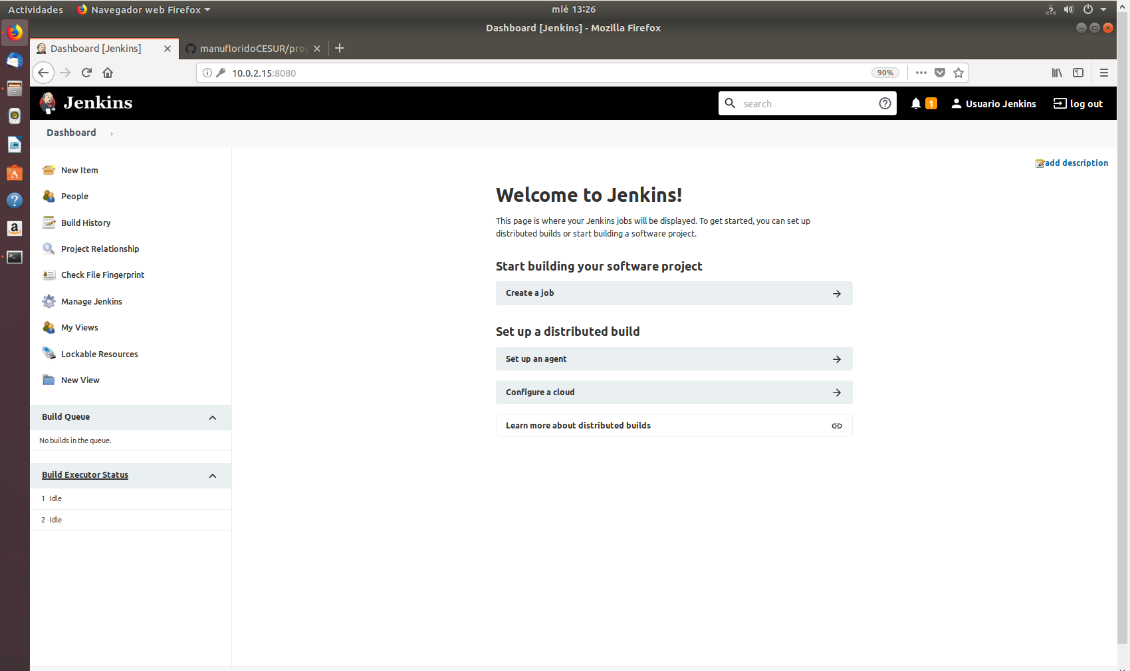
* Para subir los archivos al repositorio de GitHub habrá que hacer un push:
* ~$ git push usuario rama



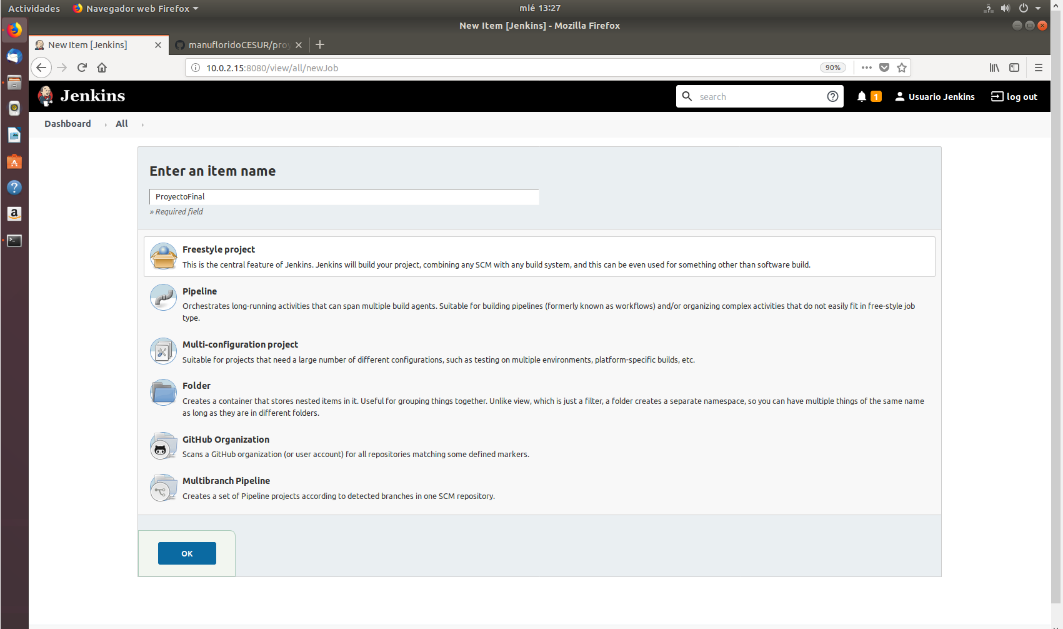
* Una vez subido el script al repositorio de GitHub, podremos crear el trabajo de Jenkins:
* Este script ejecuta varias tareas de monitorización y pega su resultado en un archivo llamado monitorización.txt

6. Crear trabajo de Jenkins

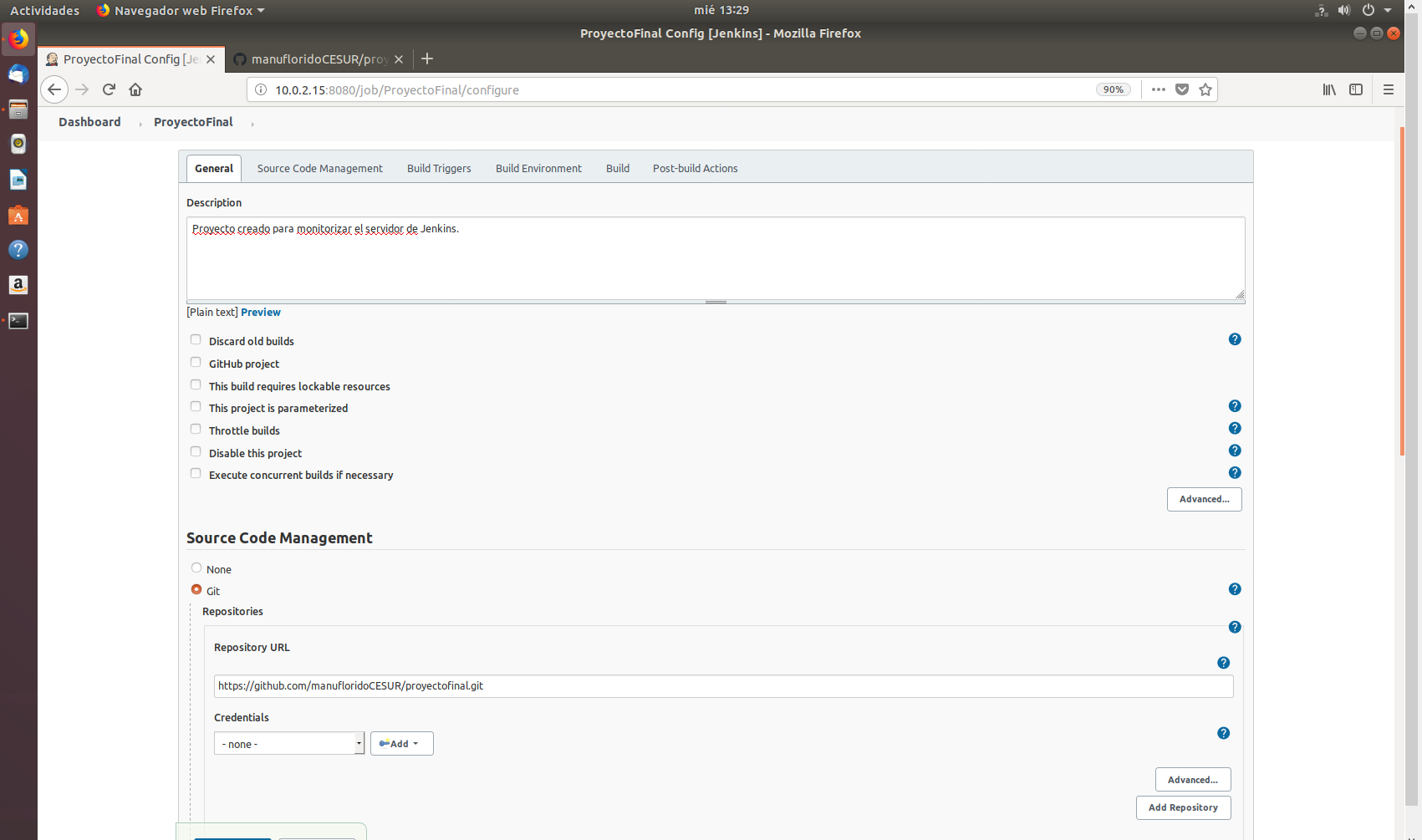
* Para crear el trabajo de Jenkins iremos al panel de control y pulsaremos “New Item”:



* Escribimos el nombre del proyecto, elegimos “Freestyle project” y le damos a “Ok”:



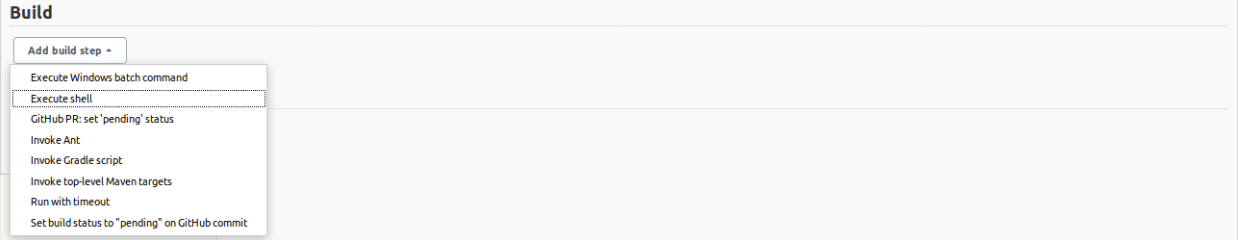
* Pulsamos en Git y escribimos la URL del repositorio de Git, también tenemos la posibilidad de escribir un comentario sobre nuestro proyecto:

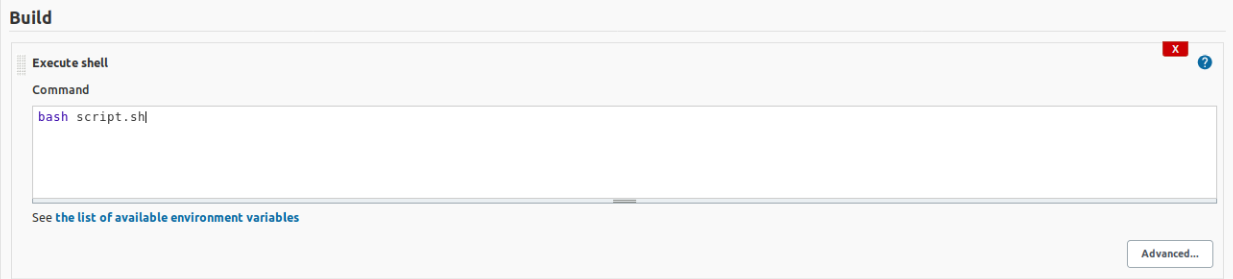


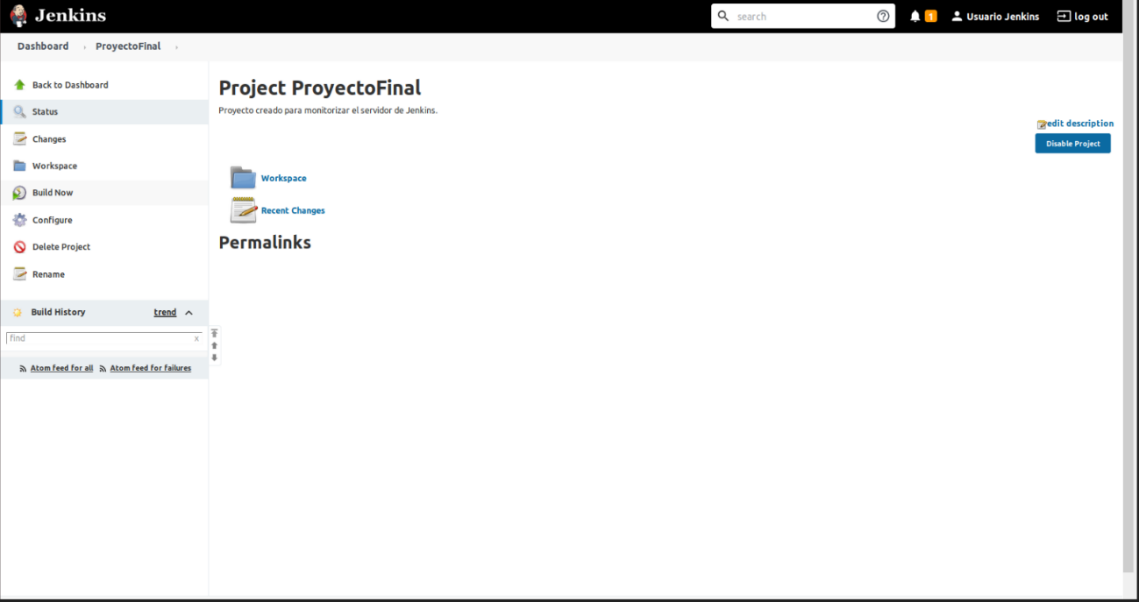
* Para que el trabajo se ejecute periódicamente tenemos que marcar la opción de “Build periodically” y elegir cada cuanto tiempo va a ejecutarse. En este caso hemos puesto que se haga cada 15 minutos.



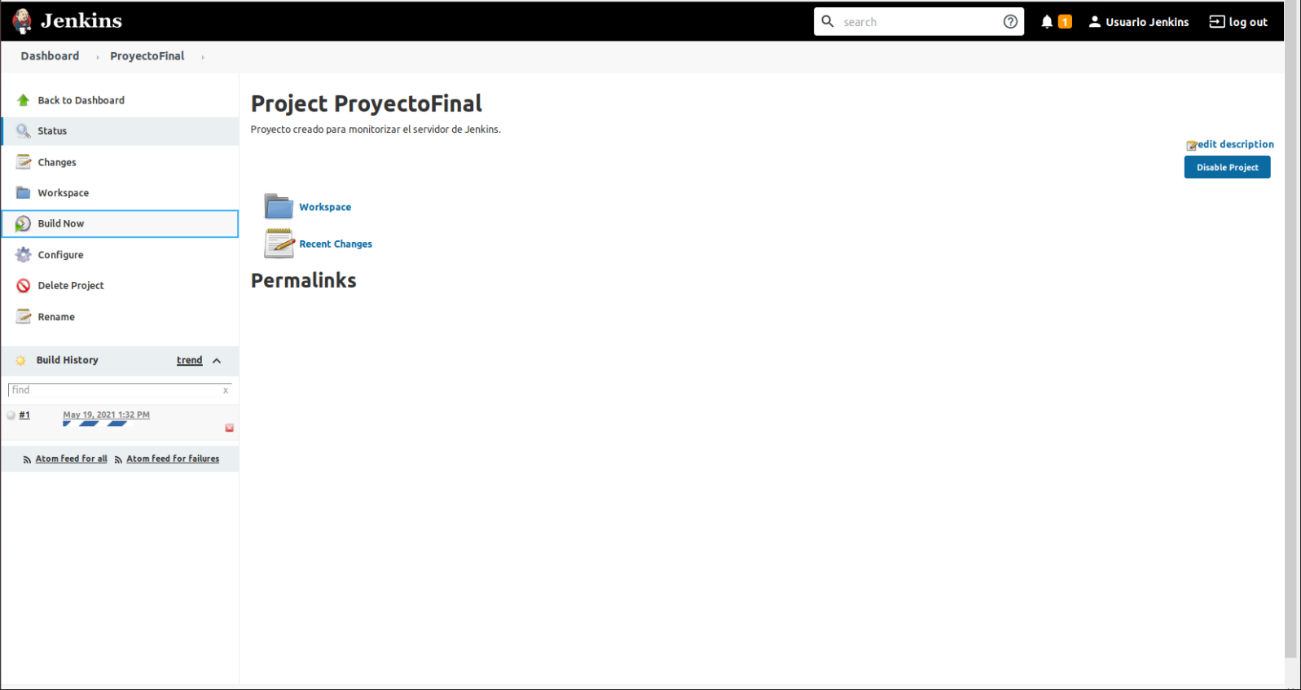
* Después especificamos la acción que queremos que se realice. Para ejecutar el script necesitaremos un terminal:



* Se abrirá un cuadro de texto donde podremos escribir los comandos que necesitemos. Nosotros pondremos que ejecute el script que tenemos en el repositorio de GitHub:
* Luego pulsaremos en “Apply” y en “Save” para que nuestro proyecto se quede guardado y aparezca en el Panel de Control de Jenkins.
* Aparecerá el menú de nuestro proyecto:



* Para que se ejecute el proyecto sin que pase el tiempo establecido, pulsaremos en “Build Now” y veremos como en el historial se añade el número #1:

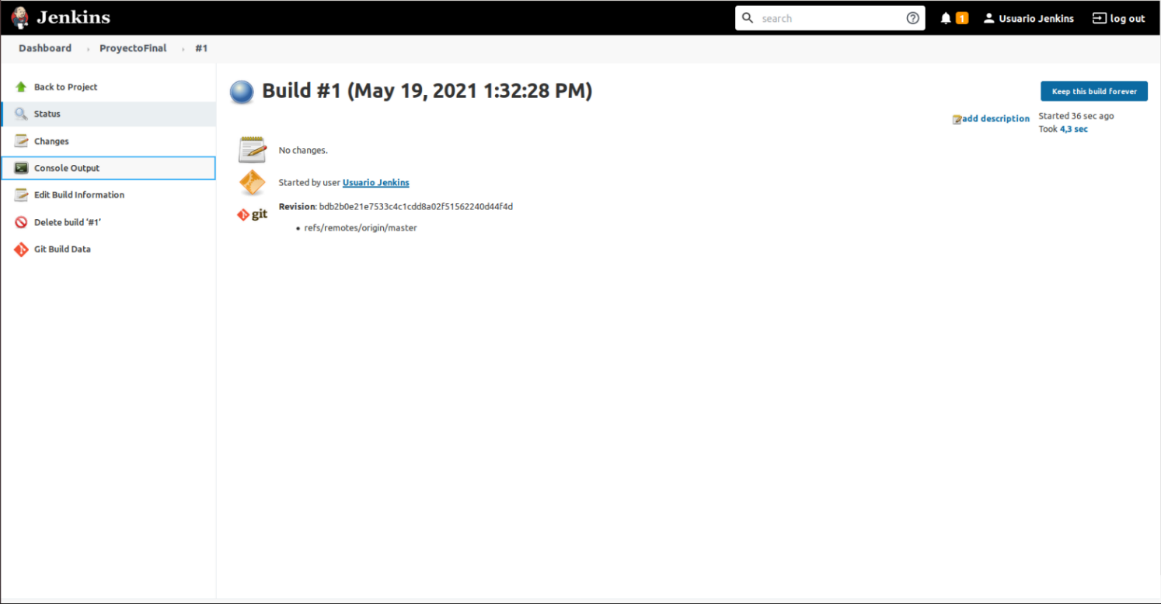


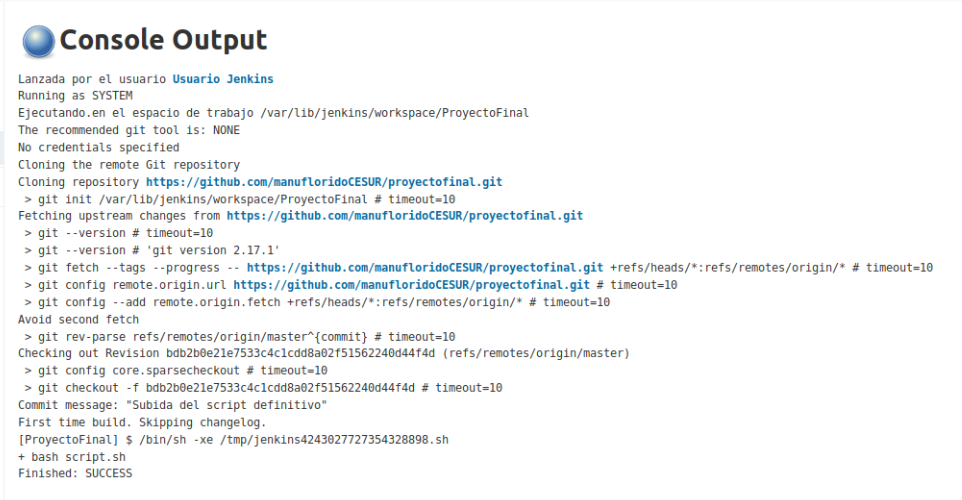
* En el caso de que no haya habido fallos en el proceso, saldrá al lado del #1 un círculo. Si el círculo sale azul significa que está bien ejecutado y si sale rojo significa que ha habido fallos:



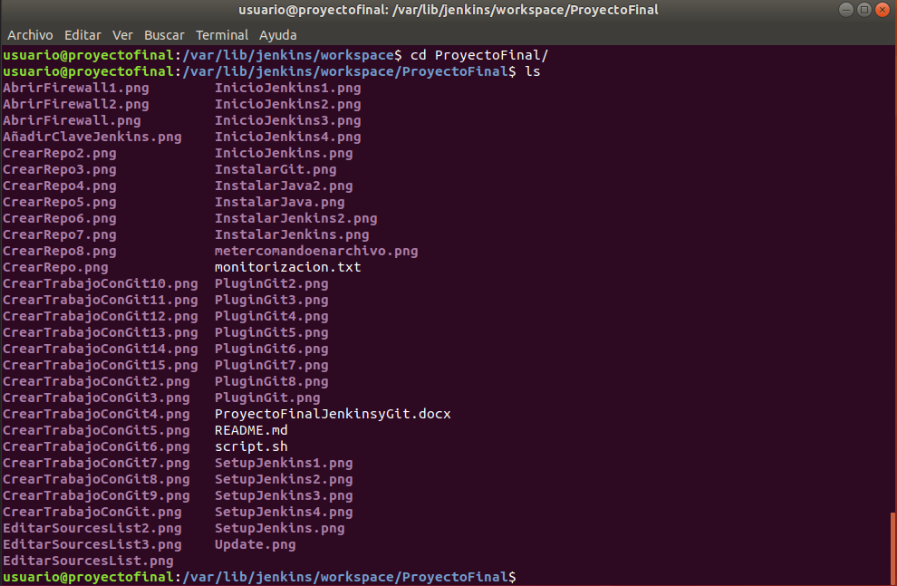


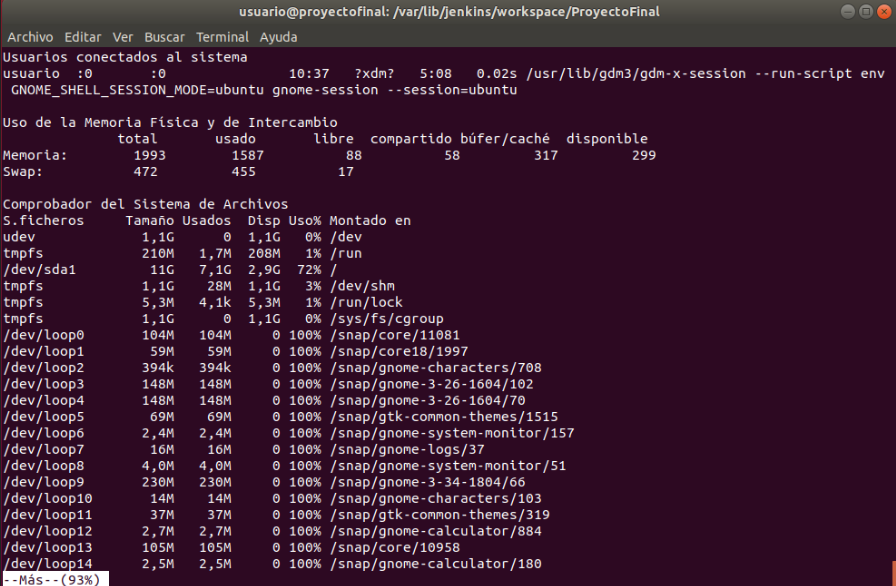
En este caso, como se puede observar, ha salido azul, por lo que significa que está bien ejecutado.

* Podemos pulsar en el número y nos mostrará las características de nuestra Build. En el apartado “Console Output” veremos todos los pasos que ha seguido Jenkins para ejecutar nuestro script:
* Veremos el usuario que ha ejecutado la Build, los comandos que ha ejecutado el plugin de Git (ha clonado el repositorio a su Workspace) y al final ha ejecutado el script:



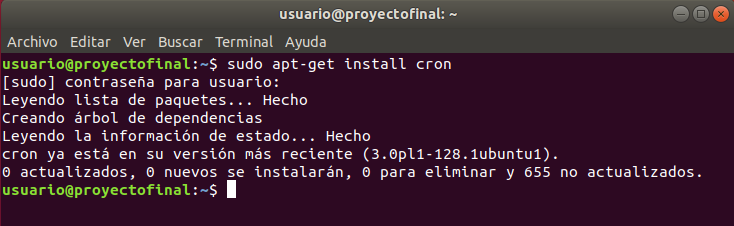
* Ahora accederemos a la ruta del Workspace para ver si se ha realizado de todo el clonado del repositorio:



* Como podemos ver, está todo el repositorio clonado y a parte el script ha creado el archivo monitorizacion.txt:

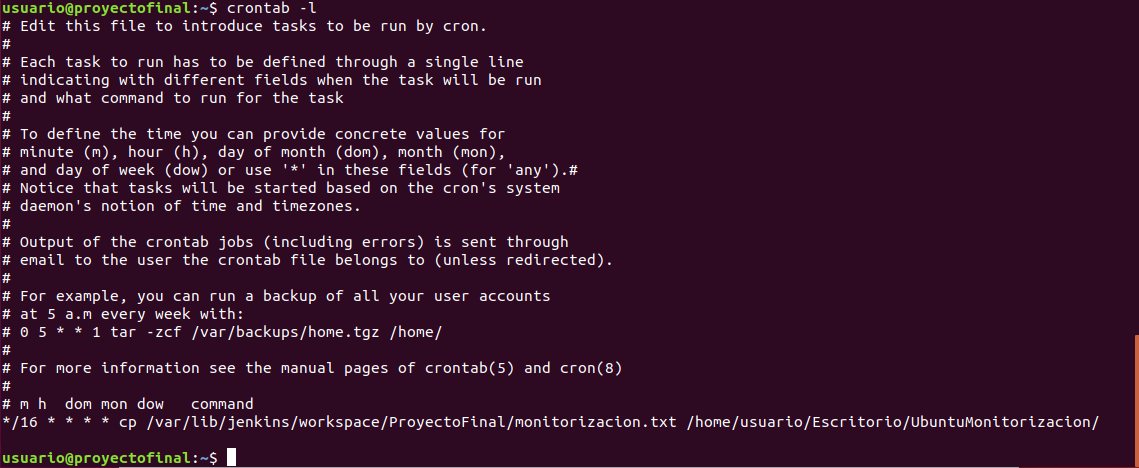
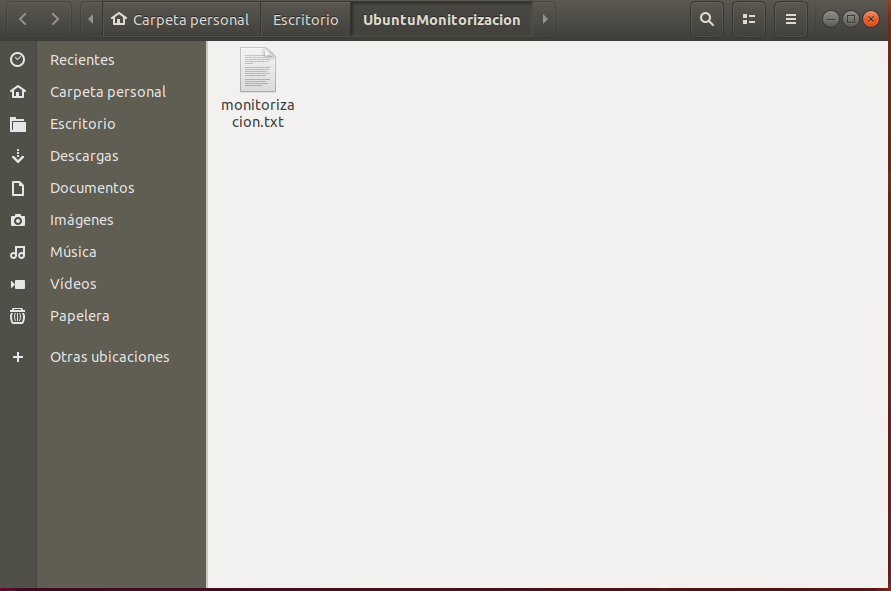
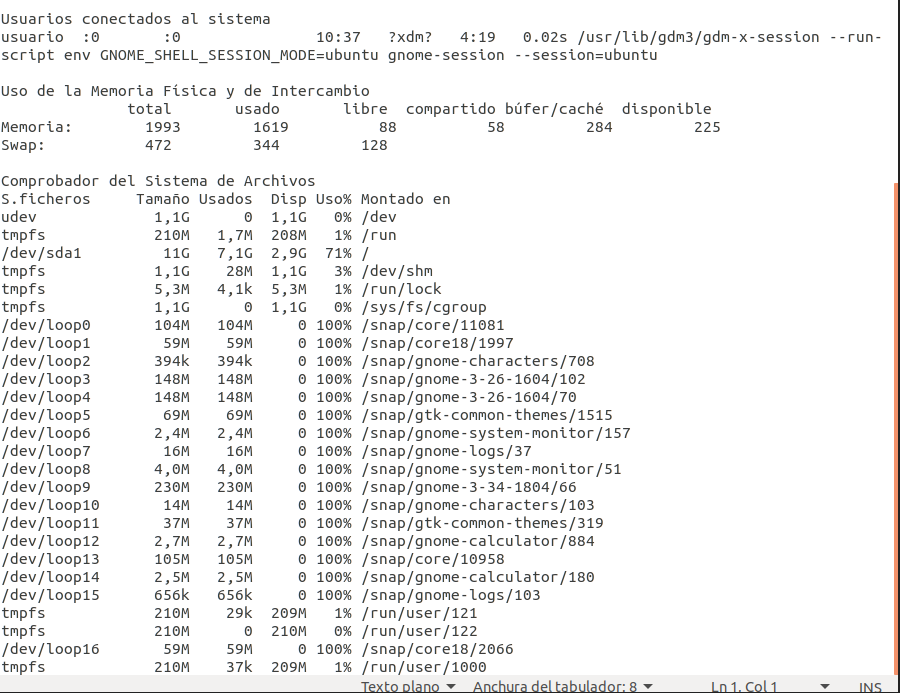
7. Crontab

* Una vez comprobada la creación del archivo, crearemos una tarea programada en Crontab, para que copie el archivo a una carpeta del escritorio y sea más fácil acceder al archivo.
* El primer paso es comprobar si tenemos instalado el paquete “cron”:
* ~$ sudo apt-get install cron



* Ejecutamos el comando para iniciar el Crontab:
* ~$ crontab -e
* Veremos el archivo donde añadir una línea con el comando que queremos ejecutar y el tiempo que pasará antes de volver a ejecutar la orden. Añadiremos esta línea:



* Cada 16 minutos se ejecutará la copia del archivo “monitorizacion.txt” a la carpeta “UbuntuMonitorizacion” ubicada en el escritorio, recordemos que cada 15 minutos se ejecuta el script en el servidor Jenkins. Si borramos del comando el /16, se ejecutará cada minuto.
* Podemos ver los procesos que ejecutará Crontab con:
* ~$ crontab -l
* Pasado el tiempo que hayamos especificado procedemos a ir al escritorio y mirar en la carpeta a ver si se ha copiado el archivo:
* Pulsamos en el archivo y comprobamos que tenga la misma información que el archivo original:



8. Bibliografía

<https://www.guru99.com/jenkins-github-integration.html>

<https://www.youtube.com/watch?v=bGqS0f4Utn4>

<https://www.youtube.com/watch?v=rwbN6JjDCXo>

<https://www.hostinger.com/tutorials/how-to-install-jenkins-on-ubuntu/>

<https://linuxize.com/post/how-to-install-jenkins-on-ubuntu-18-04/>

<https://www.jenkins.io/doc/book/installing/>

<https://www.blazemeter.com/blog/how-to-integrate-your-github-repository-to-your-jenkins-project>

<https://www.digitalocean.com/community/tutorials/how-to-install-jenkins-on-ubuntu-18-04-es>

<https://www.guru99.com/download-install-jenkins.html>

<https://phoenixnap.com/kb/install-jenkins-ubuntu>

<https://www.techrepublic.com/article/how-to-install-jenkins-on-ubuntu-server-18-04/>

<https://www.youtube.com/watch?v=1jNXEWTBgDo>

<https://www.youtube.com/watch?v=63HEKFh8T2c>

<https://www.redeszone.net/2017/01/09/utilizar-cron-crontab-linux-programar-tareas/>

<https://geekytheory.com/programar-tareas-en-linux-usando-crontab>

<https://www.linuxtotal.com.mx/?cont=info_admon_006>

<https://www.youtube.com/watch?v=s4wEmAFwhsg>

<https://www.youtube.com/watch?v=bGqS0f4Utn4>

<https://www.youtube.com/watch?v=h8gbk7coq5g>

<https://www.blazemeter.com/blog/how-to-integrate-your-github-repository-to-your-jenkins-project>

<https://huongdanjava.com/install-git-plugin-in-jenkins.html>

<https://www.thegeekstuff.com/2016/10/jenkins-git-setup/>