

Trabajo Práctico 7 - Servidor de Build (de integración continua).

1- Poniendo en funcionamiento Jenkins

Crear una imagen de Docker que se base en la imagen oficial de Jenkins y que tenga instalado .NET Core.

Crear un archivo llamado Dockerfile.jenkins con el siguiente contenido:

FROM jenkins/jenkins:lts

USER root

Instala dependencias necesarias

**RUN apt-get update && apt-get install -y \
apt-transport-https \
software-properties-common \
wget**

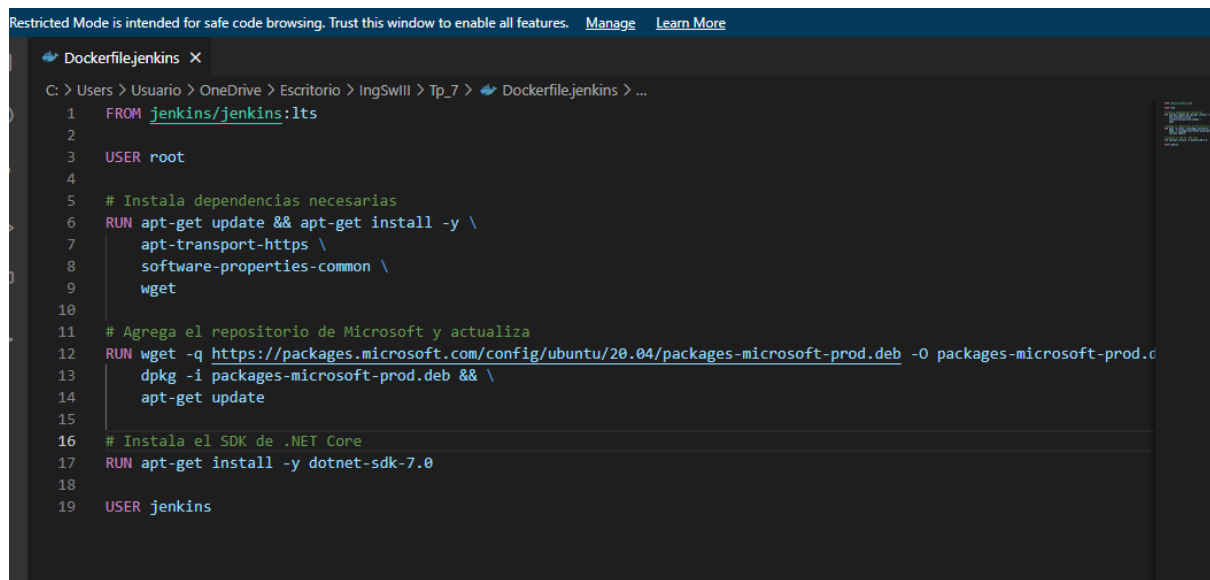
Agrega el repositorio de Microsoft y actualiza

**RUN wget -q
https://packages.microsoft.com/config/ubuntu/20.04/packages-microsoft-prod.deb -O
packages-microsoft-prod.deb && \
dpkg -i packages-microsoft-prod.deb && \
apt-get update**

Instala el SDK de .NET Core

RUN apt-get install -y dotnet-sdk-7.0

USER jenkins

A screenshot of a code editor window titled 'Dockerfile.jenkins'. The editor shows the following content:

```
1 FROM jenkins/jenkins:lts
2
3 USER root
4
5 # Instala dependencias necesarias
6 RUN apt-get update && apt-get install -y \
7     apt-transport-https \
8     software-properties-common \
9     wget
10
11 # Agrega el repositorio de Microsoft y actualiza
12 RUN wget -q https://packages.microsoft.com/config/ubuntu/20.04/packages-microsoft-prod.deb -O packages-microsoft-prod.d
13     dpkg -i packages-microsoft-prod.deb && \
14     apt-get update
15
16 # Instala el SDK de .NET Core
17 RUN apt-get install -y dotnet-sdk-7.0
18
19 USER jenkins
```

The editor has a dark theme and a sidebar on the right showing a file explorer view.

Desde la misma ubicación donde tengas el archivo Dockerfile personalizado, ejecuta el siguiente comando para construir una nueva imagen de Docker con .NET Core y Jenkins. Esto creará una nueva imagen de Docker llamada jenkins-with-dotnetcore:

docker build -t jenkins-with-dotnetcore -f Dockerfile.jenkins .

```
PS C:\Users\Usuario\OneDrive\Escritorio\IngSwIII\Tp_7> docker build -t jenkins-with-dotnetcore -f Dockerfile
.jenkins
[+] Building 104.2s (9/9) FINISHED
=> [internal] load .dockerignore                                docker:default
=> => transferring context: 2B                                  0.0s
=> [internal] load build definition from Dockerfile.jenkins    0.0s
=> => transferring dockerfile: 570B                             0.0s
=> [internal] load metadata for docker.io/jenkins:lts         2.7s
=> [auth] jenkins/jenkins:pull token for registry-1.docker.io 0.0s
=> [1/4] FROM docker.io/jenkins/jenkins:lts@sha256:b705323eaf70a7da4c1eed8b816f33dff2d5c8c3671170a2 29.5s
=> => resolve docker.io/jenkins/jenkins:lts@sha256:b705323eaf70a7da4c1eed8b816f33dff2d5c8c3671170a2c 0.0s
=> => sha256:f6154b0007a4a7dcee550de8b63ddbdf177961a972a33c54b2c4390dc4a873bf 61.32MB / 61.32MB 22.8s
=> => sha256:5e0093fb8d5f36e86b93870a544e50bf21f3968e872f91ad0a2bc9abdc672472 8.68MB / 8.68MB 1.0s
=> => sha256:b705323eaf70a7da4c1eed8b816f33dff2d5c8c3671170a2c17cf77aa4f15432 3.12kB / 3.12kB 0.0s
=> => sha256:2a15645303916751e9fd781fd403051199d94052d5d18be6e1b2398aab1f79c1 2.77kB / 2.77kB 0.0s
=> => sha256:3d0d48f61941fc7cb5253b07ef932d06ee6e335445f6f53ac3d1d49e385a7e 12.62kB / 12.62kB 0.0s
=> => sha256:012c0b3e998c1a0c0bedcf712eaaaf188580529dd026a04aa1ce13fdb39e42b 49.56MB / 49.56MB 7.9s
=> => sha256:34d3735955786e60f8bb8e7bd6fa2dfd7739dc047ba49c637c18a5d54b99d23c 1.23kB / 1.23kB 1.5s
=> => sha256:10cad8b7dbf876c2af6402c97060b77d723bd6838daef8c2fadab362864f768 183B / 183B 1.9s
=> => sha256:2471c7529ac37af126ea6c1afad134f08afa5c684f7e10816e314d286ce7dc38 89.33MB / 89.33MB 22.7s
=> => sha256:655eaf4c27231f280d11bcd10dfa0887a3165d2eb9345ab44ed75063bed17ae7 189B / 189B 8.2s
=> => extracting sha256:012c0b3e998c1a0c0bedcf712eaaaf188580529dd026a04aa1ce13fdb39e42b 5.8s
=> => sha256:3145851d1a67db7eb3dd0d7f6734a26340476b9d1f6db58668088d72b4833847 6.09MB / 6.09MB 9.4s
=> => sha256:614d8a8c6a95f7a9ab30388cf16a55f34eb54e0bcc52d4c3ef5033ac46200cd8 77.15MB / 77.15MB 19.7s
=> => sha256:95f5dcd4e08b1d468bac93b830b9aa4d9c0f302d5be66c478dc70fb48fb6bf62 1.93kB / 1.93kB 20.5s
=> => sha256:7c8f869ed080beb2a95df25c497c699a86746c448fbb3a6f29b9fd42313ec2a49 1.16kB / 1.16kB 20.8s
=> => sha256:3a7be85a26d2887982180264f604ffb596884aa270ed5a80eed3b30711cfd44c 392B / 392B 21.1s
=> => sha256:d754f693a7c2efe59a3b9340ff01604beb630ca1094f9024926d915957cd7fcd 265B / 265B 21.4s
=> => extracting sha256:f6154b0007a4a7dcee550de8b63ddbdf177961a972a33c54b2c4390dc4a873bf 3.5s
=> => extracting sha256:5e0093fb8d5f36e86b93870a544e50bf21f3968e872f91ad0a2bc9abdc672472 0.2s
=> => extracting sha256:314d8a8c6a95f7a9ab30388cf16a55f34eb54e0bcc52d4c3ef5033ac46200cd8 0.0s
=> => extracting sha256:10cad8b7dbf876c2af6402c97060b77d723bd6838daef8c2fadab362864f768 0.0s
=> => extracting sha256:2471c7529ac37af126ea6c1afad134f08afa5c684f7e10816e314d286ce7dc38 0.9s
=> => extracting sha256:655eaf4c27231f280d11bcd10dfa0887a3165d2eb9345ab44ed75063bed17ae7 0.0s
=> => extracting sha256:3145851d1a67db7eb3dd0d7f6734a26340476b9d1f6db58668088d72b4833847 0.1s
=> => extracting sha256:614d8a8c6a95f7a9ab30388cf16a55f34eb54e0bcc52d4c3ef5033ac46200cd8 1.1s
=> => extracting sha256:95f5dcd4e08b1d468bac93b830b9aa4d9c0f302d5be66c478dc70fb48fb6bf62 0.0s
=> => extracting sha256:7c8f869ed080beb2a95df25c497c699a86746c448fbb3a6f29b9fd42313ec2a49 0.0s
=> => extracting sha256:3a7be85a26d2887982180264f604ffb596884aa270ed5a80eed3b30711cfd44c 0.0s
=> => extracting sha256:d754f693a7c2efe59a3b9340ff01604beb630ca1094f9024926d915957cd7fcd 0.0s
=> [2/4] RUN apt-get update && apt-get install -y apt-transport-https software-properties-c 35.8s
=> [3/4] RUN wget -q https://packages.microsoft.com/config/ubuntu/20.04/packages-microsoft-prod.deb 5.7s
=> [4/4] RUN apt-get install -y dotnet-sdk-7.0 26.6s
=> exporting to image 3.9s
```

Ejecuta un Contenedor con la Nueva Imagen. Ahora, puedes ejecutar un contenedor utilizando la imagen que acabas de crear:

windows

mkdir -p c:\jenkins

docker run -d -p 8080:8080 -p 50000:50000 --name jenkins -v c:\jenkins:/var/jenkins_home jenkins-with-dotnetcore

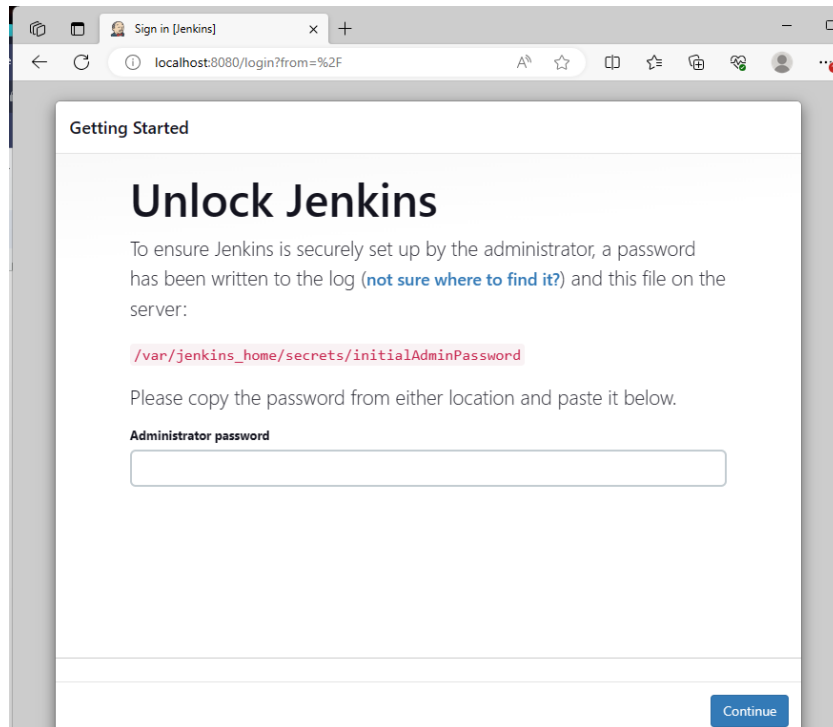
```
PS C:\Users\Usuario\OneDrive\Escritorio\IngSwIII\Tp_7> mkdir -p c:\jenkins

Directorio: C:\

Mode                LastWriteTime         Length Name
----                -
d-----          17/10/2023   10:15             jenkins

PS C:\Users\Usuario\OneDrive\Escritorio\IngSwIII\Tp_7> docker run -d -p 8080:8080 -p 50000:50000 --name jenkins -v c:\jenkins:/var/jenkins_home jenkins-with-dotnetcore
7f27646d95feaf22ac64ba3c653057bf2fafc4222512d9c25b762a4450c358a0
```

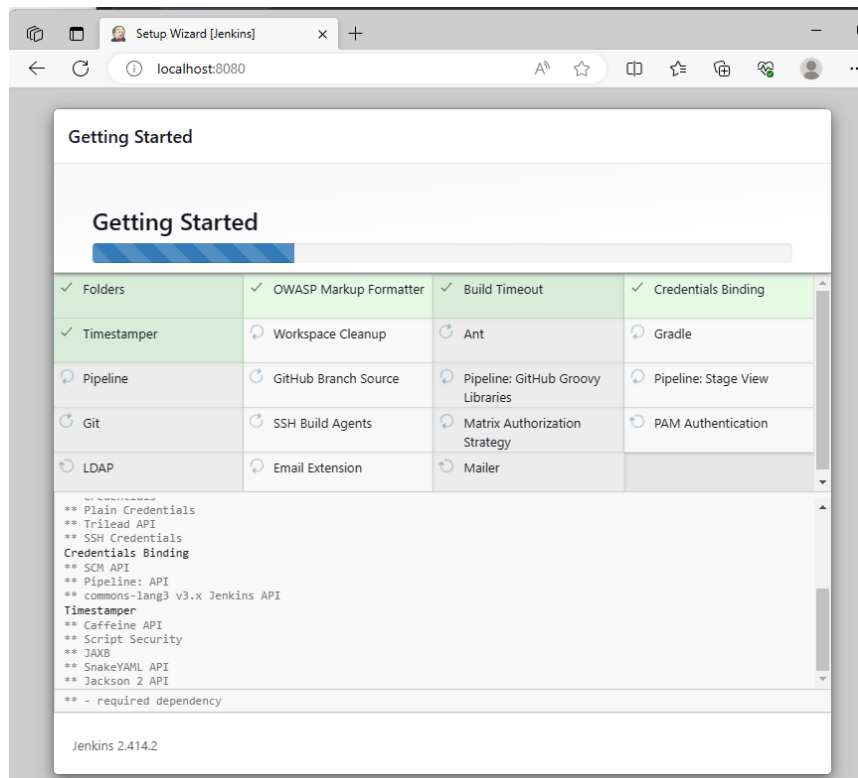
Una vez en ejecución, abrir <http://localhost:8080>



Inicialmente deberá especificar el texto que se encuentra dentro del archivo
`~/.jenkins/secrets/initialAdminPassword` en el contenedor

`cat ~/.jenkins/secrets/initialAdminPassword`

Instalar los plugins por defecto alt text



Crear el usuario admin inicial. Colocar cualquier valor que considere adecuado. alt text

Getting Started

Create First Admin User

Usuario

Contraseña

Confirma la contraseña

Nombre completo

Dirección de email

Jenkins 2.414.2

[Skip and continue as admin](#) [Save and Continue](#)

JENKINS URL: <http://localhost:8080/>

Panel de control [Jenkins]

localhost:8080

Jenkins

búsqueda (CTRL+K)

Panel de Control

- + Nueva Tarea
- Personas
- Historial de trabajos
- Administrar Jenkins
- Mis vistas

Trabajos en la cola

No hay trabajos en la cola

Estado del ejecutor de construcciones

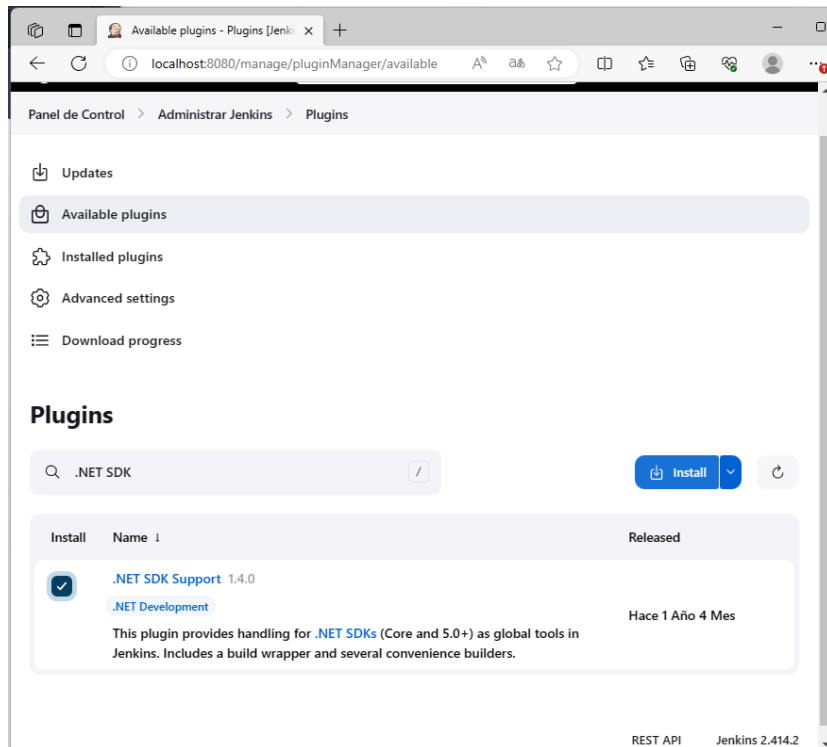
- 1 Inactivo
- 2 Inactivo

[añadir descripción](#)

¡Bienvenido a Jenkins!

2- Instalando Plugins y configurando herramientas

En Administrar Jenkins vamos a la sección de Administrar Plugins
De la lista de plugins disponibles instalamos .NET SDK Support

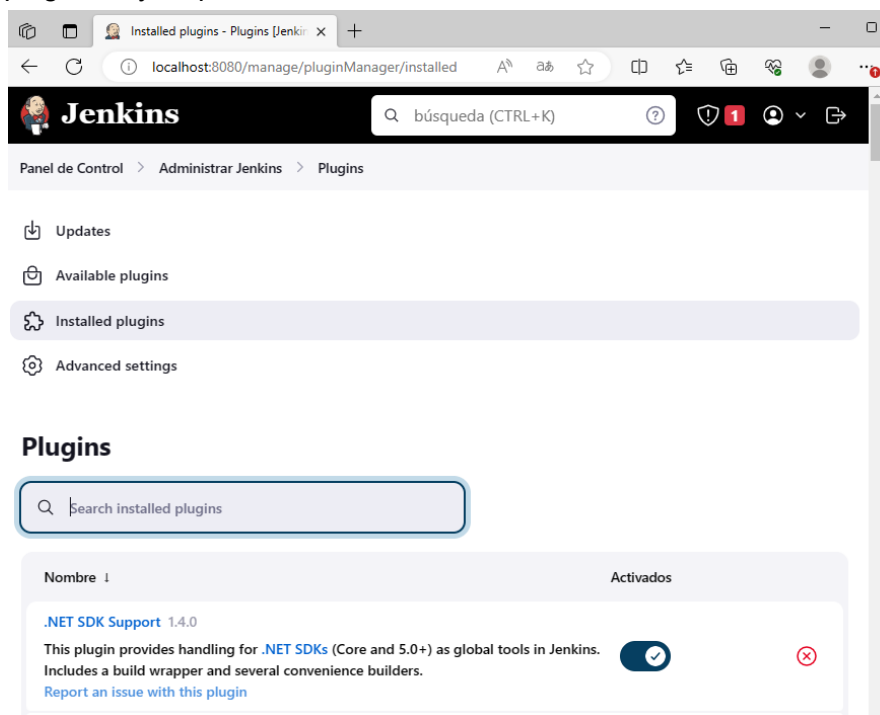


.NET SDK Support



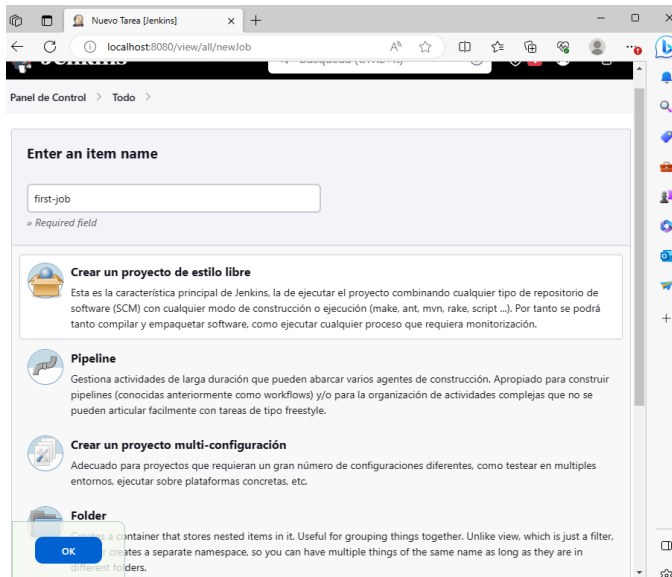
Reiniciamos el servidor

Abrir nuevamente página de Plugins y explorar la lista, para familiarizarse qué tipo de plugins hay disponibles.



3- Creando el primer Job de estilo libre

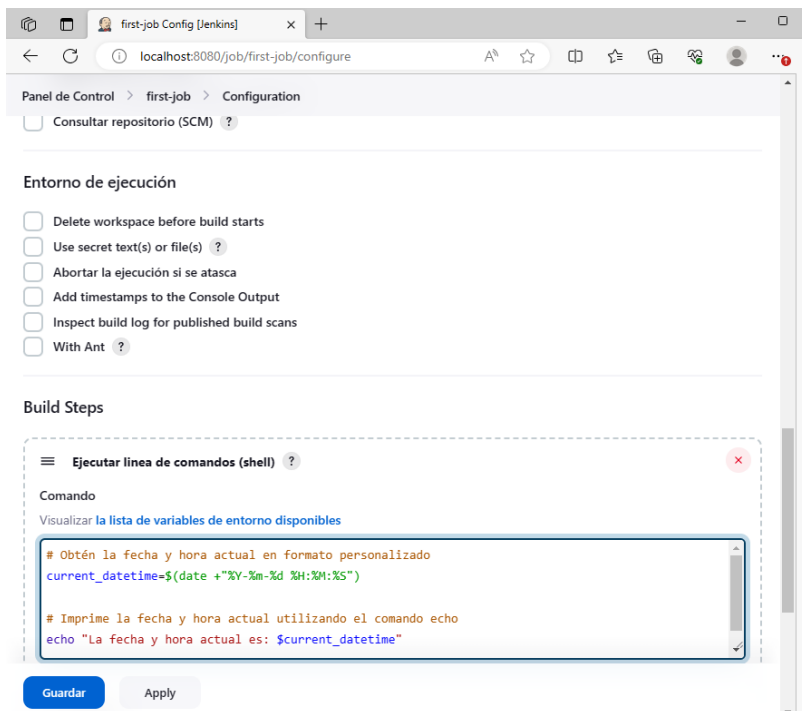
Crear un nuevo item, del tipo estilo libre con nombre first-job image



Una vez creado el job, en la sección Build Steps seleccionamos Ejecutar linea de comandos (shell) y escribimos:

Obtén la fecha y hora actual en formato personalizado
current_datetime=\$(date +"%Y-%m-%d %H:%M:%S")

Imprime la fecha y hora actual utilizando el comando echo
echo "La fecha y hora actual es: \$current_datetime"



Guardamos y ejecutamos el Job

The screenshot shows the Jenkins web interface in a browser. The address bar indicates the URL is `localhost:8080/job/first-job/1/console`. The breadcrumb navigation shows 'Panel de Control > first-job > #1 > Salida de consola'. Below the navigation, there are several icons and links: a code icon, 'Cambios', a terminal icon, 'Console Output', a document icon, 'View as plain text', a checkmark icon, 'Editar información de la ejecución', a trash icon, 'Delete build '#1'', and a right arrow icon, 'Ejecución siguiente'.

✓ Salida de consola

```
Started by user AGOSTINA
Running as SYSTEM
Building in workspace /var/jenkins_home/jobs/first-job/workspace
[workspace] $ /bin/sh -xe /tmp/jenkins10917129180032070158.sh
+ date +%Y-%m-%d %H:%M:%S
+ current_datetime=2023-10-17 14:09:31
+ echo La fecha y hora actual es: 2023-10-17 14:09:31
La fecha y hora actual es: 2023-10-17 14:09:31
Finished: SUCCESS
```

- Analizar la salida del mismo

La salida de consola indica que el **first-job #1** fue iniciado por el usuario previamente creado AGOSTINA. Este first-job se está construyendo en un espacio de trabajo ubicado en `/var/jenkins_home/jobs/first-job/workspace`.

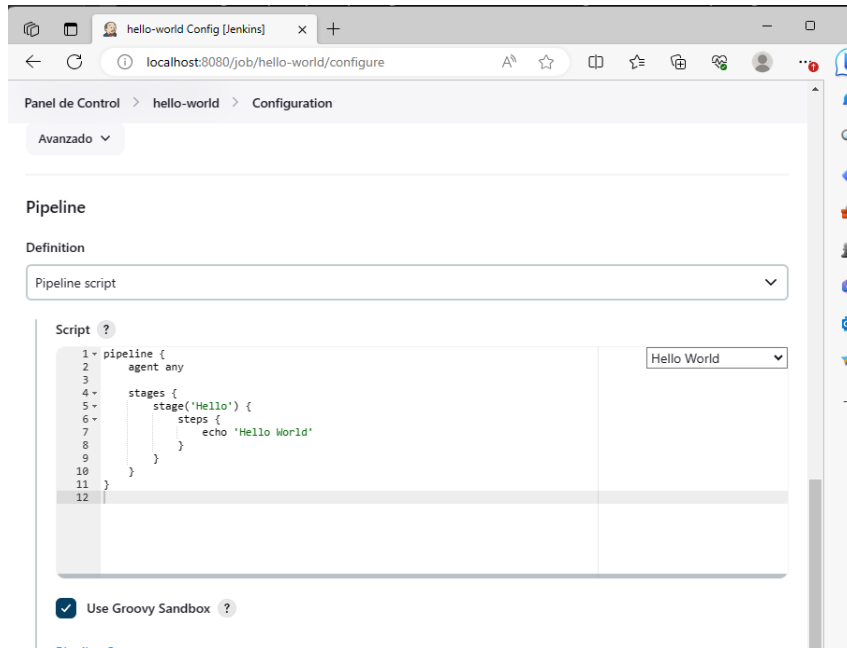
Captura la fecha y hora actual utilizando el comando **date** y la almacena en la variable **current_datetime**. Luego, muestra la fecha y hora actual en la consola. Se completó con éxito, como lo indica el mensaje "Finished: SUCCESS". Se ejecutó sin errores y la fecha y hora actual se mostraron en la consola.

4- Creando el primer Pipeline Job

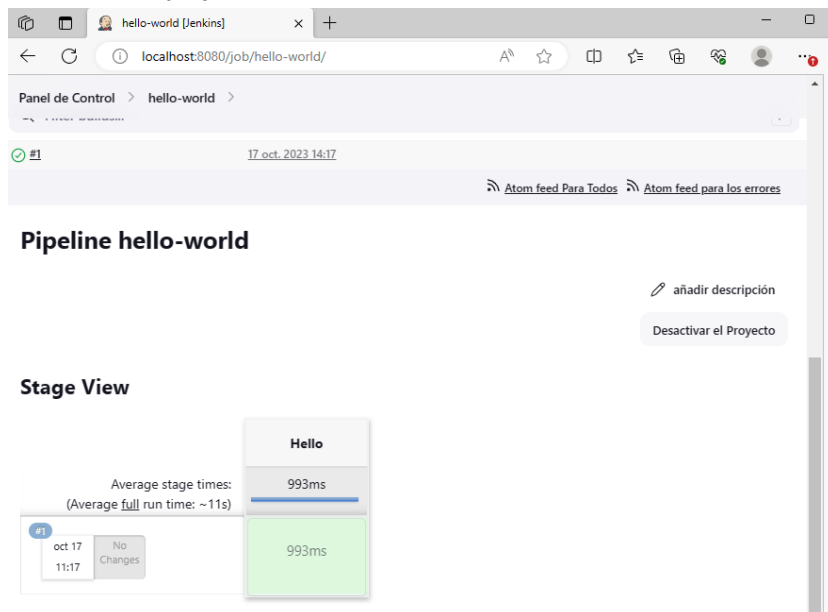
Crear un nuevo item, del tipo Pipeline con nombre hello-world

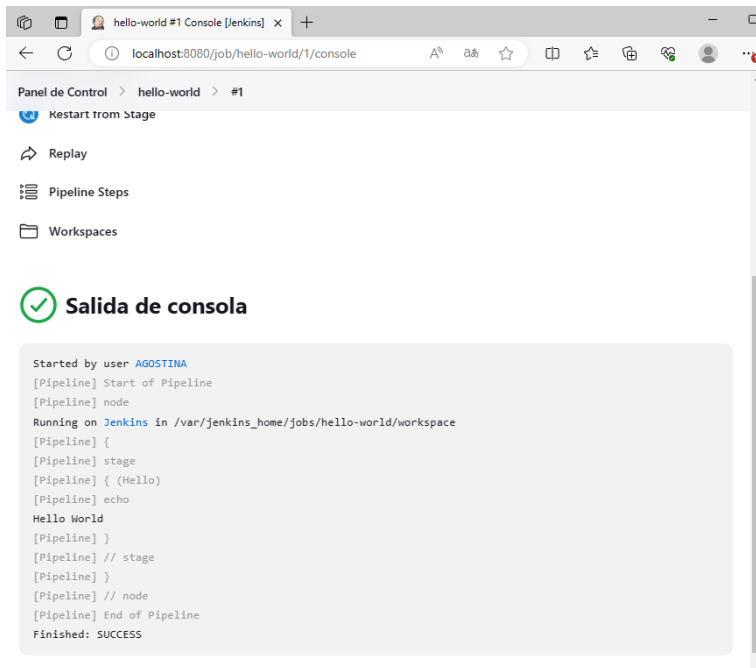
The screenshot shows the Jenkins 'New Item' page in a browser. The address bar indicates the URL is `localhost:8080/view/all/newJob`. The breadcrumb navigation shows 'Panel de Control > Todo >'. The main content area has a section titled 'Enter an item name' with a text input field containing 'hello-world' and a red underline. Below the input field is a link that says '» Required field'. There are two options for creating a new item: 'Crear un proyecto de estilo libre' (Create a free-style project) and 'Pipeline'. The 'Pipeline' option is selected. The description for 'Pipeline' states: 'Gestiona actividades de larga duración que pueden abarcar varios agentes de construcción. Apropiado para construir pipelines (conocidas anteriormente como workflows) y/o para la organización de actividades complejas que no se pueden articular fácilmente con tareas de tipo freestyle.'

Una vez creado el job, en la sección Pipeline seleccionamos try sample Pipeline y luego Hello World



Guardamos y ejecutamos el Job





The screenshot shows the Jenkins web interface in a browser. The address bar indicates the URL is `localhost:8080/job/hello-world/1/console`. The page title is "hello-world #1 Console [Jenkins]". The breadcrumb navigation shows "Panel de Control > hello-world > #1". On the left sidebar, there are links for "Restart from Stage", "Replay", "Pipeline Steps", and "Workspaces". The main content area is titled "Salida de consola" with a green checkmark icon. It displays the following console output:

```
Started by user AGOSTINA
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/jenkins_home/jobs/hello-world/workspace
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Hello)
[Pipeline] echo
Hello World
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Analizar la salida del mismo:

Esta salida de consola se refiere a la ejecución del trabajo en Jenkins utilizando un pipeline, un flujo de trabajo automatizado. Indica que el pipeline fue iniciado por el usuario llamado "AGOSTINA".

Start of Pipeline y End of Pipeline: indica el inicio del pipeline.

node y // node: marca el inicio de un nodo en el pipeline. Un nodo es un agente de Jenkins donde se ejecutarán los pasos del pipeline.

Se muestra que el nodo actual en el pipeline se está ejecutando en un entorno Jenkins, y el directorio de trabajo actual es `/var/jenkins_home/jobs/hello-world/workspace`.

Las llaves (`{}`) indican el inicio y fin de un bloque de pipeline y se utilizan para agrupar conjuntos de pasos relacionados.

stage y // stage marca el inicio y fin de una etapa dentro del pipeline, son partes del pipeline donde se realizan tareas específicas, en este caso es "Hello" el nombre de la etapa. El comando `echo` se utiliza para imprimir texto en la salida de la consola, imprime "Hello World" en la salida de la consola.

Finished: SUCCESS: El pipeline se completó con éxito.

5- Creando un Pipeline Job con Git

Similar al paso anterior creamos un ítem con el nombre github-job

En script escribir:

```
pipeline {  
  agent any  
  stages {  
    stage('Build') {  
      steps {  
        // Get some code from a GitHub repository  
        git branch: 'main', url: 'https://github.com/ingsoft3ucc/SimpleWebAPI'  
      }  
    }  
  }  
}
```

Guardar y ejecutar el Job

github-job #1 [Jenkins]

localhost:8080/job/github-job/1/

Panel de Control > github-job > #1

Console Output

Edit Build Information

Delete build '#1'

Git Build Data

Restart from Stage

Replay

Pipeline Steps

Workspaces

✓

Build #1 (17 oct. 2023 14:30:28)

Conservar esta ejecución para siempre

añadir descripción

Started 20 Seg ago
Took 7.2 Seg

Iniciado por el usuario AGOSTINA

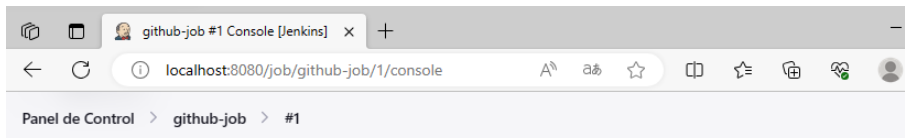
git

Revision: 8726ce58a1894af134cd22fcfc7ecb38ad9444
Repository: https://github.com/ingsoft3ucc/SimpleWebAPI
• refs/remotes/origin/main

Pipeline Steps

Workspaces

Step	Arguments	Status
Start of Pipeline - (7 Seg in block)		✓
node - (6.7 Seg in block)		<div></div> ✓
node block - (5.9 Seg in block)		✓
stage - (5.6 Seg in block)	Build	<div></div> ✓
stage block (Build) - (5 Seg in block)		✓
git - (4.7 Seg in self)		<div></div> ✓



Salida de consola

```
Started by user AGOSTINA
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/jenkins_home/jobs/github-job/workspace
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Build)
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/ingsoft3ucc/SimpleWebAPI
> git init /var/jenkins_home/jobs/github-job/workspace # timeout=10
Fetching upstream changes from https://github.com/ingsoft3ucc/SimpleWebAPI
> git --version # timeout=10
> git --version # 'git version 2.39.2'
> git fetch --tags --force --progress -- https://github.com/ingsoft3ucc/SimpleWebAPI
+refs/heads/*:refs/remotes/origin/* # timeout=10
> git config remote.origin.url https://github.com/ingsoft3ucc/SimpleWebAPI # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 8726ce58a1894af134cd22fcfb7eccb38ad9444 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 8726ce58a1894af134cd22fcfb7eccb38ad9444 # timeout=10
> git branch -a -v --no-abbrev # timeout=10
> git checkout -b main 8726ce58a1894af134cd22fcfb7eccb38ad9444 # timeout=10
Commit message: "Update WeatherForecastController.cs"
First time build. Skipping changelog.
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

6- Utilizando nuestros proyectos.

6.1- .NET Core

Crear un Job de estilo libre llamado "git-netcore-job" que construya el proyecto en .NET Core del ejercicio 1 del TP 05.

Configurar el Job para obtener el código desde el repositorio de cada alumno:

Indicar la rama desde la cual se construya:

Configurar el origen del código fuente

☐ Ninguno

☒ Git ?

Repositories ?

Repository URL ?

https://github.com/AgostinaMorellato/SimpleWebAPI

Credentials ?

- none -

Add ▾

Avanzado ▾

Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ?

*/master

Add Branch

Navegador del repositorio ?

Agregar las credenciales correspondientes:
Configurar las etapas de Build y Publish del proyecto
image

git-netcore-job Config [Jenkins] x +

localhost:8080/job/git-netcore-job/configure

Panel de Control > git-netcore-job > Configuration

☐ Ninguno

☒ Git ?

Repositories ?

Repository URL ?

https://github.com/AgostinaMorellato/SimpleWebAPI

Credentials ?

AGOSTINA00/*****

Add ▾

Avanzado ▾

Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ?

*/main

Add Branch



Salida de consola

```
Started by user AGOSTINA
Running as SYSTEM
Building in workspace /var/jenkins_home/jobs/git-netcore-job/workspace
The recommended git tool is: NONE
using credential 58a6a2f2-9168-496b-90eb-0514bb8e780f
> git rev-parse --resolve-git-dir /var/jenkins_home/jobs/git-netcore-job/workspace/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/AgostinaMorellato/SimpleWebAPI # timeout=10
Fetching upstream changes from https://github.com/AgostinaMorellato/SimpleWebAPI
> git --version # timeout=10
> git --version # 'git version 2.39.2'
using GIT_ASKPASS to set credentials
> git fetch --tags --force --progress -- https://github.com/AgostinaMorellato/SimpleWebAPI
+refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 8726ce58a1894af134cd22fcfbc7ecb38ad9444 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 8726ce58a1894af134cd22fcfbc7ecb38ad9444 # timeout=10
Commit message: "Update WeatherForecastController.cs"
> git rev-list --no-walk 8726ce58a1894af134cd22fcfbc7ecb38ad9444 # timeout=10
[workspace] $ dotnet build SimpleWebAPI
MSBuild version 17.7.3+8ec440e68 for .NET
    Determining projects to restore...
    Restored /var/jenkins_home/jobs/git-netcore-job/workspace/SimpleWebAPI/SimpleWebAPI.csproj (in 12.02
sec).
    SimpleWebAPI -> /var/jenkins_home/jobs/git-netcore-
job/workspace/SimpleWebAPI/bin/Debug/net7.0/SimpleWebAPI.dll

Build succeeded.
    0 Warning(s)
    0 Error(s)

Time Elapsed 00:00:18.90
.NET Command Completed - Exit Code: 0

[workspace] $ dotnet publish SimpleWebAPI --self-contained false
MSBuild version 17.7.3+8ec440e68 for .NET
    Determining projects to restore...
    All projects are up-to-date for restore.
    SimpleWebAPI -> /var/jenkins_home/jobs/git-netcore-
job/workspace/SimpleWebAPI/bin/Debug/net7.0/SimpleWebAPI.dll
    SimpleWebAPI -> /var/jenkins_home/jobs/git-netcore-job/workspace/SimpleWebAPI/bin/Debug/net7.0/publish/
.NET Command Completed - Exit Code: 0

Finished: SUCCESS
```

Corremos el Job y revisamos la salida.

La salida de consola es el registro de la ejecución del trabajo en Jenkins. Fue iniciado por mi usuario llamado "AGOSTINA". El trabajo se está ejecutando con permisos del sistema, para asegurar que tenga acceso a los recursos necesarios. No está utilizando una herramienta de Git específica.

using credential 58a6a....: Jenkins está utilizando una credencial con el ID

"58a6a2f2-9168-496b-90eb-0514bb8e780f" para autenticarse en el repositorio remoto.

git rev-parse --resolve-git-dir /var/jenkins_home/jobs/git-netcore-job/workspace/.git:

verifica el directorio Git para el proyecto en el espacio de trabajo.

Fetching changes from the remote Git repository: trata de obtener cambios desde el repositorio Git remoto.

git config remote.origin.url https://github.com/AgostinaMorellato/SimpleWebAPI:

Establece la URL del repositorio Git remoto.

Fetching upstream changes from

https://github.com/AgostinaMorellato/SimpleWebAPI: obtiene cambios (pull) desde el repositorio remoto en GitHub.

git fetch --tags --force --progress --

https://github.com/AgostinaMorellato/SimpleWebAPI

+refs/heads/*:refs/remotes/origin/*: Realiza una operación de git fetch para obtener todas las ramas y etiquetas del repositorio remoto.

Checking out Revision 8726ce58a1894af134cd22fcfb7ecb38ad9444

(refs/remotes/origin/main): está cambiando al commit específico con el hash

"8726ce58a1894af134cd22fcfb7ecb38ad9444" en la rama "main".

Luego, se ejecutan comandos **dotnet** para construir y publicar el proyecto .NET llamado "SimpleWebAPI". La construcción se realiza con éxito y no se generan advertencias ni errores.

Finalmente, se muestra **"Finished: SUCCESS"**, lo que indica que el trabajo de Jenkins se completó con éxito.

6.2- Monitorear Cambios en el repo de GitHub

Configurar Poll cada 1 minuto

Panel de Control > git-netcore-job > Configuration

(Auto) ▼

Additional Behaviours

Añadir ▼

Disparadores de ejecuciones

- ☐ Lanzar ejecuciones remotas (ejem: desde 'scripts') ?
- ☐ Construir tras otros proyectos ?
- ☐ Ejecutar periódicamente ?
- ☐ GitHub hook trigger for GITScm polling ?
- ☒ Consultar repositorio (SCM) ?

Programador ?

⚠ Do you really mean "every minute" when you say "*****"? Perhaps you meant "H*****" to poll once per hour
Would last have run at Tuesday, October 17, 2023 at 3:47:34 PM Coordinated Universal Time; would next run at Tuesday, October 17, 2023 at 3:47:34 PM Coordinated Universal Time.

☐ Ignore post-commit hooks ?

Verificar su funcionamiento al hacer un commit y cuando no se hizo commit

Panel de Control > git-netcore-job > #8 > Cambios

- Estatus
- </> Cambios**
- Console Output
- Editar información de la ejecución
- Delete build '#8'
- Git Build Data
- Ejecución previa
- Ejecución siguiente

✓ Cambios

Summary

Panel de Control > git-netcore-job > #9 > Cambios

- Estatus
- </> Cambios**
- Console Output
- Editar información de la ejecución
- Delete build '#9'
- Logs de polling
- Git Build Data
- Ejecución previa

✓ Cambios

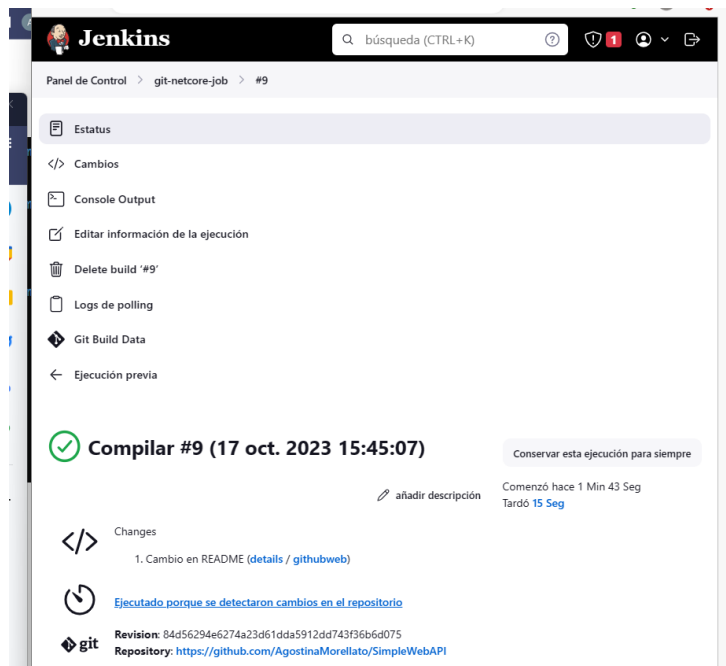
Summary

1. Cambio en README (details)

Commit 84d56294e6274a23d61dda5912dd743f36b6d075 by 2004827
Cambio en README

[README.md \(diff\)](#)

```
PS C:\Users\Usuario\OneDrive\Escritorio\IngSwIII\SimpleWebAPI> git commit -m "Cambio en README"
[main 84d5629] Cambio en README
 1 file changed, 1 insertion(+), 2 deletions(-)
PS C:\Users\Usuario\OneDrive\Escritorio\IngSwIII\SimpleWebAPI> git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 332 bytes | 332.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/AgostinaMorellato/SimpleWebAPI.git
 8726ce5..84d5629  main -> main
```

Explicar por qué no es posible tener un WebHook en nuestro TP

Un webhook es una forma de que una aplicación o servicio en línea proporcione información en tiempo real a otra aplicación o servicio. Es una solicitud HTTP (POST) que se envía automáticamente desde un servidor cuando ocurre un evento específico en el servidor fuente.

No es posible tener uno en nuestro TP, por que el servidor que recibe el webhook debe ser accesible públicamente desde el servidor emisor. Esto puede plantear problemas de seguridad o configuración de firewall. Debes tomar medidas adicionales para garantizar la seguridad de los webhooks, como la autenticación y la validación de las solicitudes entrantes.

Explicar Diferencia con ejecutar periódicamente:

Si ejecutamos a partir de la consulta de cambios en el repositorio, este va a consultar si hay cambios y en el caso de que lo haga va a ejecutar nuevamente, por otro lado si no lo hay no ejecuta, solo se hacen ejecuciones si existen cambios.

Al ejecutar periódicamente, ejecutará con el intervalo de tiempo que le configuremos, haciendo ejecuciones siempre, por más de que este ejecutando lo mismo varias veces, en este caso cada un minuto.

Historia de tareas		Tendencia ▾
Filter builds...		
✓ #13	17 oct. 2023 15:53	
✓ #12	17 oct. 2023 15:52	
✓ #11	17 oct. 2023 15:51	
✓ #10	17 oct. 2023 15:50	

6.3- Node.js

Repetir el ejercicio 6.1 para construir el proyecto del ejercicio 2 del TP 05.