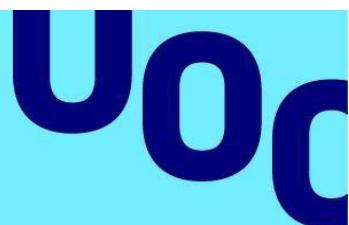


# FP.050 - (P) Devops y cloud computing

Grupo: ALT\_CTRL\_Supr PRODUCTO 2



## Descripción

- Despliegue automatizado en la nube de AWS y creación de servicio de alta disponibilidad a través de Kubernetes.

## Objetivos

- Los objetivos principales del producto son:

Configurar el despliegue automatizado en el servidor de AWS.

Configurar el servicio en alta disponibilidad mediante la utilización de Kubernetes.

- Alvaro Parada De Lucas
- Bernat Diaz Calafat
- Rubén Vicente Gilabert

## Contenido

Caso práctico.....	3
1. Leer detenidamente estas instrucciones e identificar los requerimientos de la actividad.....	3
2. Revisar detenidamente la rúbrica de evaluación.....	3
3. Consultar los recursos necesarios facilitados en el aula.....	3
4. Conectarse al servidor EC2 de AWS.....	3
5. Instalar Jenkins.....	8
6. Configurar Jenkins para que despliegue automáticamente nuestros cambios en el código de la aplicación, publicados en GitHub.....	11
7. Instalar MiniKube en nuestra máquina virtual.....	23
8. Instalar Kubernetes en nuestra máquina virtual.....	24
9. Configurar Kubernetes para que nuestro contenedor se encuentre en alta disponibilidad (balanceo de carga entre mínimo dos instancias). .....	25
10. Comprobar que el balanceo funciona correctamente.....	27
Bibliografía.....	29
Anexo.....	30
Anexo 1 - k8s.yaml .....	30

# Caso práctico

1. Leer detenidamente estas instrucciones e identificar los requerimientos de la actividad.
2. Revisar detenidamente la rúbrica de evaluación.
3. Consultar los recursos necesarios facilitados en el aula.
4. Conectarse al servidor EC2 de AWS.

El primer paso es crear una nueva cuenta en AWS seguido de crear una nueva instancia. Deberemos tener en cuenta si queremos el servicio de pago, o si por lo contrario con el servicio gratuito ya podemos trabajar. En nuestro caso utilizaremos el servicio gratuito. Para registrar nuestra cuenta y tener nuestra instancia seguiremos estos pasos:

Primero deberemos buscar Amazon AWS para ir a la página, y seleccionar “Crear una cuenta para AWS”



O también podemos acceder desde los recursos de aprendizaje del grado:

## Recursos



Una vez aquí introduciremos nuestros datos para crear la cuenta:

## Regístrate en AWS

Dirección de correo electrónico del usuario raíz  
Se utiliza para la recuperación de cuentas y tal como se describe en [Aviso de privacidad de AWS](#)

aparadad@uoc.edu

### Nombre de cuenta de AWS

Elija un nombre para su cuenta. Puede cambiar este nombre en la configuración de la cuenta después de registrarse.

Alt\_Ctrl\_Supr

**Verificar la dirección de correo electrónico**

O

**Iniciar sesión en una cuenta de AWS existente**

Seguiremos los pasos necesarios hasta estar dentro de la cuenta que hemos creado.

The screenshot shows the AWS Billing and Cost Management console. The top navigation bar includes the AWS logo, a search bar, and a 'Buscar' button. The main menu has 'Administración de facturación y costos' selected. On the left, there's a sidebar with 'Inicio', 'Introducción', and 'Páginas'. The main content area is titled 'Cuenta' and shows 'Información'. It displays account details: 'Nombre' (Alt\_Ctrl\_Supr), 'ID' (183408706607), and 'Proveedor de servicios' (Amazon Web Services EMEA SARL). A red box highlights the 'Nombre' field. The top right corner shows the account ID '1834-0870-6607' and the name 'Alt\_Ctrl\_Supr'. A red arrow points from the text 'A continuación, en la página que es la principal entraremos en "Todos los servicios" > Informática > EC2' to the 'Cerrar la cuenta' button.

A continuación, en la página que es la principal entraremos en “Todos los servicios” > Informática > EC2

The screenshot shows the AWS Console Home page. The top navigation bar includes the AWS logo, a search bar, and a 'Buscar' button. The main content area has a 'Recently visited' section with a red box around the 'View all services' button. To the right, there's a 'Applications' section with a red box around the 'Create application' button. The top right corner shows the account ID '1834-0870-6607' and the name 'Alt\_Ctrl\_Supr'. A red arrow points from the text 'Una vez instalado, veremos que no tenemos ningún recurso, y, por lo tanto, ninguna instancia.' to the 'View all services' button.

Una vez instalado, veremos que no tenemos ningún recurso, y, por lo tanto, ninguna instancia.

El siguiente paso es lanzar una instancia para poder crearla. Damos clic derecho al botón en naranja que pone “Lanzar instancias”:

The screenshot shows the AWS EC2 Instances page. At the top, there's a search bar with 'Buscar' and a placeholder '[Alt+S]'. Below it, a filter bar includes 'Estado de la instancia = running' and 'Quitar los filtros'. The main area displays a table header with columns: Name, ID de la instancia, Estado de la i..., Tipo de inst... (with a dropdown arrow), Comprobación de, Estado de la al..., Zona de disponib..., DNS de IPv4 pública, Dirección IP..., and IP elástica. A message at the bottom says 'No se encontraron instancias que coincidan'.

A partir de aquí configuraremos nuestra instancia. Para ello seguiremos los pasos indicados a continuación.

En nuestro caso hemos configurado la instancia de esta manera:

- Nombre de la instancia.

#### Lanzar una instancia Información

Amazon EC2 le permite crear máquinas virtuales, o instancias, que se ejecutan en la nube de AWS. Comience rápidamente siguiendo los sencillos pasos que se indican a continuación.

#### Nombre y etiquetas Información

##### Nombre

Alt\_Ctrl\_Supr\_entorno\_produccion

[Agregar etiquetas adicionales](#)

- Hemos configurado un par de claves.

### Crear par de claves

**Nombre del par de claves**  
Con los pares de claves es posible conectarse a la instancia de forma segura.

Alt\_Ctrl\_Supr

El nombre puede incluir hasta 255 caracteres ASCII. No puede incluir espacios al principio ni al final.

**Tipo de par de claves**

RSA  
Par de claves pública y privada cifradas mediante RSA

ED25519  
Par de claves privadas y públicas cifradas ED25519

**Formato de archivo de clave privada**

.pem  
Para usar con OpenSSH

.ppk  
Para usar con PuTTY

**⚠️** Cuando se le solicite, almacene la clave privada en un lugar seguro y accesible del equipo. Lo necesitará más adelante para conectarse a la instancia. [Más información](#)

[Cancelar](#) [Crear par de claves](#)

- Para el Inicio rápido, hemos elegido Ubuntu.

**Inicio rápido**

Amazon Linux	macOS	Ubuntu	Windows	Red Hat	SUSE Linux	Debian

**Buscar más AMI**  
Inclusión de AMI de AWS, Marketplace y la comunidad

**Imágenes de máquina de Amazon (AMI)**

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type  
ami-0a716d3f3b16d290c (64 bits (x86)) / ami-0cab1941a8a08b817 (64 bits (Arm))  
Virtualización: hvm Activado para ENA: true Tipo de dispositivo raíz: ebs

**Apto para la capa gratuita**

**Descripción**  
Ubuntu Server 24.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).

Canonical, Ubuntu, 24.04, amd64 noble image

Arquitectura	ID de AMI	Fecha de publicación	Nombre de usuario
64 bits (x86)	ami-0a716d3f3b16d290c	2025-08-21	ubuntu <span style="background-color: green; color: white; padding: 2px;">Proveedor verificado</span>

- Seleccionamos el tipo de instancia.

**▼ Tipo de instancia** [Información](#) | [Obtener asesoramiento](#)

**Tipo de instancia**

t3.micro	<a href="#">Apto para la capa gratuita</a>
Familia: t3 2 vCPU 1 GiB Memoria Generación actual: true	
Bajo demanda Ubuntu Pro base precios: 0.0143 USD por hora	Bajo demanda RHEL base precios: 0.0396 USD por hora
Bajo demanda SUSE base precios: 0.0108 USD por hora	Bajo demanda Linux base precios: 0.0108 USD por hora
Bajo demanda Windows base precios: 0.02 USD por hora	

**Todas las generaciones**

**Comparar tipos de instancias**

**Se aplican costos adicionales a las AMI con software preinstalado**

- Configuración de red

**▼ Configuraciones de red** [Información](#)

**Red** | [Información](#)  
vpc-08dedd8427a148273

**Subred** | [Información](#)  
Sin preferencias (subred predeterminada en cualquier zona de disponibilidad)

**Asignar automáticamente la IP pública** | [Información](#)  
Habilitar

**Firewall (grupos de seguridad)** | [Información](#)  
Un grupo de seguridad es un conjunto de reglas de firewall que controlan el tráfico de la instancia. Agregue reglas para permitir que un tráfico específico llegue a la instancia.

**Crear grupo de seguridad** Ayuda a establecer conexión con la instancia

**Seleccionar un grupo de seguridad existente**

Crearemos un nuevo grupo de seguridad denominado "launch-wizard-1" con las siguientes reglas:

**Permitir el tráfico de SSH desde** Cualquier lugar 0.0.0.0/0

**Permitir el tráfico de HTTPS desde Internet** Para configurar un punto de enlace, por ejemplo, al crear un servidor web

**Permitir el tráfico de HTTP desde Internet** Para configurar un punto de enlace, por ejemplo, al crear un servidor web

**Resumen**

Número de instancias: [Información](#) 1

Imagen de software (AMI)  
Canonical, Ubuntu, 24.04, amd64...[más información](#)

ami-0a716d3f3b16d290c

Tipo de servidor virtual (tipo de instancia)  
t3.micro

Firewall (grupo de seguridad)  
Nuevo grupo de seguridad

Almacenamiento (volúmenes)  
Volumenes: 1 (8 GiB)

**Lanzar instancia** [Código de versión preliminar](#)

Y para ver que todo está correcto, lanzamos la instancia.

**Lanzamiento de instancia**

**Inicio del lanzamiento**

80%

Deberíamos ver algo así para saber que se ha lanzado de forma correcta.

The screenshot shows a browser window with the AWS logo and search bar. The URL bar shows [Alt+S]. Below the header, it says "EC2 > Instancias > Lanzar una instancia". A green success message box contains the text: "Correcto" with a checkmark icon, followed by "El lanzamiento de la instancia se inició correctamente (i-0048a1bc41bfa051f)".

Aquí ya podríamos dar uso de nuestro sistema Ubuntu.

The terminal session displays the following text:

```
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1011-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

System information as of Sun Oct 26 10:59:50 UTC 2025

 System load:  0.0          Temperature:      -273.1 °C
 Usage of /:   25.6% of 6.71GB  Processes:         109
 Memory usage: 22%           Users logged in:   0
 Swap usage:   0%            IPv4 address for ens5: 172.31.40.234

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
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-172-31-40-234:~$
```

i-0048a1bc41bfa051f (Alt\_Crtl\_Supr\_entorno\_produccion)

PublicIPs: 16.171.37.34 PrivateIPs: 172.31.40.234

## 5. Instalar Jenkins.

Primero utilizaremos “***sudo apt update***” y “***sudo apt upgrade***” para preparar la terminal para el uso.

```
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-172-31-40-234:~$ 
ubuntu@ip-172-31-40-234:~$ sudo apt update
Hit:1 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB] 
Get:5 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [1260 kB]
Get:7 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [208 kB]
Get:50 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe Translation-en [17.5 kB]
Get:51 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [11.0 kB]
Get:52 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 c-n-f Metadata [1444 B]
Get:53 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [212 B]
Get:54 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 c-n-f Metadata [116 B]
Get:55 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Get:56 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 c-n-f Metadata [116 B]
Fetched 38.3 MB in 6s (6272 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
55 packages can be upgraded. Run 'apt list --upgradable' to see them.
ubuntu@ip-172-31-40-234:~$ 
```

```
ubuntu@ip-172-31-40-234:~$ sudo apt upgrade
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following NEW packages will be installed:
  linux-aws-6.14-headers-6.14.0-1015 linux-aws-6.14-tools-6.14.0-1015 linux-headers-6.14.0-1015-aws linux-i
The following packages will be upgraded:
  bind9-dnsutils bind9-host bind9-libs cloud-init coreutils distro-info-data dpkg fwupd landscape-common li
  libpam-systemd libpam0g libpython3.12-minimal libpython3.12-stdlib libpython3.12t64 libsqlite3-0 libssl3t
  linux-image-aws linux-tools-common locales open-vm-tools openssh-client openssh-server openssh-sftp-serv
  systemd-resolved systemd-sysv tcpdump udev udisks2 vim vim-common vim-runtime vim-tiny xxd
55 upgraded, 6 newly installed, 0 to remove and 0 not upgraded.
32 standard LTS security updates
Need to get 161 MB of archives.
After this operation, 181 MB of additional disk space will be used.
Do you want to continue? [Y/n] 
```

Jenkins requiere un entorno de ejecución de Java (JRE). La versión 17 es la mas recomendada. “***sudo apt install openjdk-17-jre -y***” para instalar Jenkins.

```
ubuntu@ip-172-31-40-234:~$ sudo apt install openjdk-17-jre -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  adwaita-icon-theme alsaa-topology-conf alsua-ucm-conf at-spi2-common at-spi2-core ca-certificates-java dconf-gsettings-backend
  fonts-dejavu-extra fonts-dejavu-mono gsettings-desktop-schemas gtk-update-icon-cache hicolor-icon-theme humanity-icon-theme
  libatk-wrapper-java libatk-wrapper-jni libatk1.0-0t64 libatspi2.0-0t64 libavahi-client3 libavahi-common-data libavahi-libdeflate0 libdrm-amdgpu libdrm-intel libfontconfig1 libgail-common libgail18t64 libgbm libgdk-pixbuf-2.0-0 libgdk-pix
  libglx-mesa0 libglx0 libgraphite2-3 libgtk2.0-0t64 libgtk2.0-bin libgtk2.0-common libharfbuzz0b libice6 libjbig0 libjpeg-t
  libpangoft2-1.0-0 libpaciaccess0 libpccslite1 libpixman-1-0 librsvg2-2 librsvg2-common libsharpuyv0 libsm6 libthai-data lib
  libx11-xcb1 libxcb-wm7 libxcb-dri3-0 libxcb-glx0 libxcb-present0 libxcb-randr0 libxcb-render0 libxcb-shape0 libxcb-shm0 libxc
  libxi6 libxinerama1 libxkbfile1 libxmu6 libxpmp4 libxrandr2 libxrender1 libxshmfence1 libxt6t64 libxtst6 libxv1 libxxf86dga
  session-migration ubuntu-mono x11-common x11-utils
Suggested packages:
  default-jre alsaa-utils libasound2-plugins cups-common gvfs liblcms2-utils pcscd librsvg2-bin libnss-mdns fonts-ipafont-got
  mesa-utils
Recommended packages:
  luit
The following NEW packages will be installed:
  adwaita-icon-theme alsaa-topology-conf alsua-ucm-conf at-spi2-common at-spi2-core ca-certificates-java dconf-gsettings-backend
  fonts-dejavu-extra fonts-dejavu-mono gsettings-desktop-schemas gtk-update-icon-cache hicolor-icon-theme humanity-icon-theme
  libatk-wrapper-java libatk-wrapper-jni libatk1.0-0t64 libatspi2.0-0t64 libavahi-client3 libavahi-common-data libavahi-libdeflate0 libdrm-amdgpu libdrm-intel libfontconfig1 libgail-common libgail18t64 libgbm libgdk-pixbuf-2.0-0 libgdk-pix
  libglx-mesa0 libglx0 libgraphite2-3 libgtk2.0-0t64 libgtk2.0-bin libharfbuzz0b libice6 libjbig0 libjpeg-t
  libpangoft2-1.0-0 libpaciaccess0 libpccslite1 libpixman-1-0 librsvg2-2 librsvg2-common libsharpuyv0 libsm6 libthai-data lib
```

Con “**java –version**” sabremos la versión utilizada.

```
ubuntu@ip-172-31-40-234:~$ java --version
openjdk 17.0.16 2025-07-15
OpenJDK Runtime Environment (build 17.0.16+8-Ubuntu-0ubuntu124.04.1)
OpenJDK 64-Bit Server VM (build 17.0.16+8-Ubuntu-0ubuntu124.04.1, mixed mode, sharing)
ubuntu@ip-172-31-40-234:~$
```

Luego de instalar JAVA, debemos añadir el repositorio de Jenkins antes de instalarlo.

Ahora debemos añadir la clave de Jenkins para autenticar el repositorio y luego añadir la URL del repositorio a tus fuentes de apt.

```
sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
```

```
ubuntu@ip-172-31-40-234:~$ sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
  https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
--2025-10-26 11:38:15-- https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
Resolving pkg.jenkins.io (pkg.jenkins.io)... 199.232.174.133, 2a04:4e42:6b::645
Connecting to pkg.jenkins.io (pkg.jenkins.io)|199.232.174.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3175 (3.1K) [application/pgp-keys]
Saving to: '/usr/share/keyrings/jenkins-keyring.asc'

/usr/share/keyrings/jenkins-keyring.asc          100%[=====] 2025-10-26 11:38:15 (46.3 MB/s) - '/usr/share/keyrings/jenkins-keyring.asc' saved [3175/3175]
```

Luego añadimos el repositorio con:

```
echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc]
https://pkg.jenkins.io/debian-stable binary/" | sudo tee /etc/apt/sources.list.d/jenkins.list
> /dev/null
```

```
ubuntu@ip-172-31-40-234:~$ echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc]" \
  https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null
ubuntu@ip-172-31-40-234:~$
```

Actualizamos con “**sudo apt update**”

```
ubuntu@ip-172-31-40-234:~$ sudo apt update
Hit:1 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://eu-north-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Ign:4 https://pkg.jenkins.io/debian-stable binary/ InRelease
Get:5 https://pkg.jenkins.io/debian-stable binary/ Release [2044 B]
Get:6 https://pkg.jenkins.io/debian-stable binary/ Release.gpg [833 B]
Hit:7 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:8 https://pkg.jenkins.io/debian-stable binary/ Packages [30.0 kB]
Fetched 32.9 kB in 0s (93.2 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
ubuntu@ip-172-31-40-234:~$
```

Finalmente instalamos Jenkins con “***sudo apt-get install jenkins***”

```
ubuntu@ip-172-31-40-234:~$ sudo apt-get install jenkins
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  net-tools
The following NEW packages will be installed:
  jenkins net-tools
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 95.2 MB of archives.
After this operation, 96.3 MB of additional disk space will be used.
Do you want to continue? [Y/n]
Fetched 95.2 MB in 2s (50.0 MB/s)
Selecting previously unselected package net-tools.
(Reading database ... 118075 files and directories currently installed.)
Preparing to unpack .../net-tools_2.10-0.1ubuntu4.4_amd64.deb ...
Unpacking net-tools (2.10-0.1ubuntu4.4) ...
Selecting previously unselected package jenkins.
Preparing to unpack .../jenkins_2.528.1_all.deb ...
Unpacking jenkins (2.528.1) ...
Setting up net-tools (2.10-0.1ubuntu4.4) ...
Setting up jenkins (2.528.1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /usr/lib/systemd/system/jenkins.service.
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
  6.14.0-1011-aws
Diagnostics:
  The currently running kernel version is not the expected kernel version 6.14.0-1015-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...

Service restarts being deferred:
/etc/needrestart/restart.d/dbus.service
systemctl restart getty@tty1.service
systemctl restart networkd-dispatcher.service
systemctl restart serial-getty@ttyS0.service
systemctl restart systemd-logind.service
systemctl restart unattended-upgrades.service

No containers need to be restarted.

User sessions running outdated binaries:
  ubuntu 0 session #2: sshd[1088,1541]
  ubuntu 0 user manager service: systemd[1435]

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-40-234:~$
```

## 6. Configurar Jenkins para que despliegue automáticamente nuestros cambios en el código de la aplicación, publicados en GitHub.

Habilitamos e iniciamos Jenkins con “sudo systemctl enable jenkins”

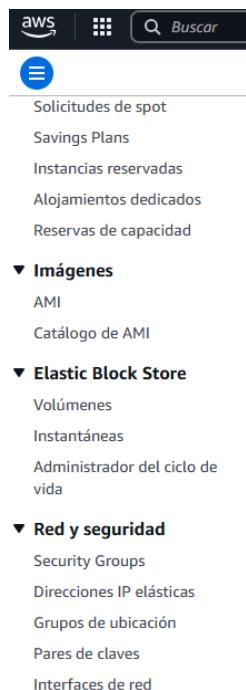
“sudo systemctl start jenkins”

```
ubuntu@ip-172-31-40-234:~$ sudo systemctl enable jenkins
Synchronizing state of jenkins.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable jenkins
ubuntu@ip-172-31-40-234:~$ sudo systemctl start jenkins
ubuntu@ip-172-31-40-234:~$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
   Active: active (running) since Sun 2025-10-26 11:54:52 UTC; 8min ago
     Main PID: 18486 (java)
        Tasks: 38 (limit: 1008)
       Memory: 294.0M (peak: 322.8M)
          CPU: 21.473s
         CGroup: /system.slice/jenkins.service
                  └─18486 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080

Oct 26 11:54:47 ip-172-31-40-234 jenkins[18486]: [LF]> This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Oct 26 11:54:47 ip-172-31-40-234 jenkins[18486]: [LF]>
Oct 26 11:54:47 ip-172-31-40-234 jenkins[18486]: [LF]> ****
Oct 26 11:54:52 ip-172-31-40-234 jenkins[18486]: 2025-10-26 11:54:52.813+0000 [id=32]      INFO    jenkins.InitReactorRunner$1#onReactorThread
Oct 26 11:54:52 ip-172-31-40-234 jenkins[18486]: 2025-10-26 11:54:52.844+0000 [id=23]      INFO    hudson.lifecycle.Lifecycle$onReactorThread
Oct 26 11:54:52 ip-172-31-40-234 systemd[1]: Started Jenkins Continuous Integration Server.
Oct 26 11:54:53 ip-172-31-40-234 jenkins[18486]: 2025-10-26 11:54:53.627+0000 [id=49]      INFO    h.m.DownloadService$DownloadJob
Oct 26 11:54:53 ip-172-31-40-234 jenkins[18486]: 2025-10-26 11:54:53.628+0000 [id=49]      INFO    hudson.util.Retriger#start: Periodic
lines 1-20/20 (END)
```

El siguiente paso será crear un grupo de seguridad en la consola de AWS, esto nos permitirá conectarnos con nuestra IP pública desde el navegador.

En el panel situado a la izquierda de la pantalla, bajamos hasta “Red y Seguridad” y clicamos en “Security Groups”



Grupos de seguridad (3) <a href="#">Información</a>						
<input type="checkbox"/>	Name	ID de grupo de seguridad	Nombre del grupo de seguridad	ID de la VPC	Descripción	Propietario
<input type="checkbox"/>	-	sg-085feac0508925919	conexion jenkins	vpc-08dedd8427a148273	Permite acceso al puerto 8080 (Jenkins)...	183408706607
<input type="checkbox"/>	-	sg-0b7ee3b5540e5118	default	vpc-08dedd8427a148273	default VPC security group	183408706607
<input type="checkbox"/>	-	sg-0009b6d8f5007787f	launch-wizard-1	vpc-08dedd8427a148273	launch-wizard-1 created 2025-10-26T1...	183408706607

Creamos un nuevo grupo y editamos las reglas de entrada, agregando una nueva:

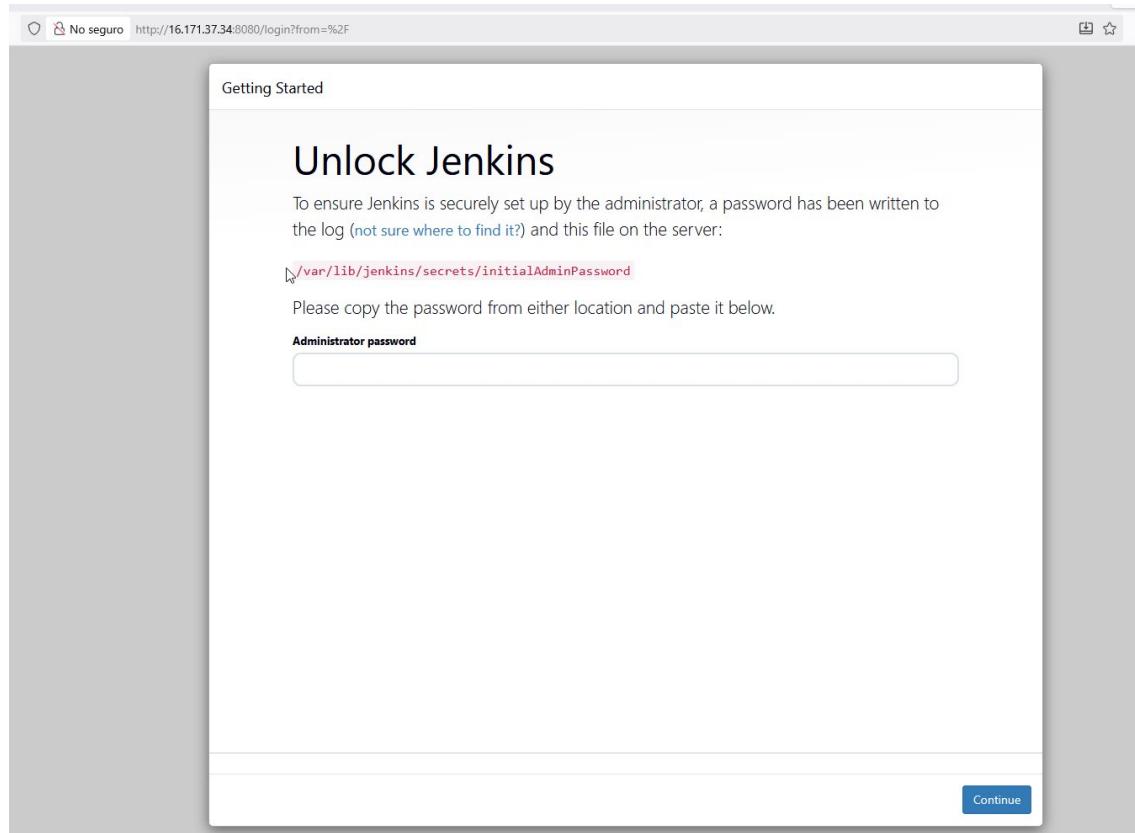
Tipo: TCP personalizado

Intervalo de puertos:8080

Origen: 0.0.0.0/0 si queremos que tenga acceso abierto.

Reglas de entrada (2)						
<input type="checkbox"/>	Name	ID de la regla del grupo	Versión de IP	Tipo	Protocolo	Intervalo de puertos
<input type="checkbox"/>	-	sgr-0d54a5a06f27f25a	IPv4	TCP personalizado	TCP	8080
<input type="checkbox"/>	-	sgr-0be57bbe10072617	IPv4	SSH	TCP	22

Una vez seguidos estos pasos, iremos a nuestro navegador y nos conectaremos a Jenkins  
<http://16.171.37.34:8080/>



La contraseña de administrador por defecto la primera vez que nos conectemos la podemos conseguir con "sudo cat /var/lib/jenkins/secrets/initialAdminPassword"

Una vez iniciada la sesión instalaremos los plugins necesarios para el despliegue automático cuando se realicen cambios en el código de la aplicación.

The screenshot shows the Jenkins dashboard at <http://16.171.37.34:8080>. The pipeline 'ALT-CTRL-SUPR-CD-Pipeline' is highlighted with a red box in the 'Trabajos en la cola' (Jobs in the queue) section. The pipeline has a status icon (blue circle with a white dot), a waiting icon (yellow sun), and the name 'ALT-CTRL-SUPR-CD-Pipeline'. Below the queue, it says '(0 of 2 ejecutores busy)'.

En el menú izquierdo - Administrar Jenkins y se desplegarán las opciones de configuración del sistema

Panel de Control > Administrar Jenkins

#### System Configuration

- System  
Configurar variables globales y rutas.
- Tools  
Configure tools, their locations and automatic installers.
- Plugins  
Añadir, borrar, desactivar y activar plugins que extienden la funcionalidad de Jenkins.
- Nodes  
Añadir, borrar, gestionar y monitorizar los nodos sobre los que Jenkins ejecuta tareas.
- Docker  
Plugin for launching build Agents as Docker containers
- Clouds  
Add, remove, and configure cloud instances to provision agents on-demand.
- Appearance  
Configure the look and feel of Jenkins

Iremos a Plugins y en la pestaña de "Available plugins" buscamos e instalamos las siguientes extensiones

## Plugins

- Updates
- Available plugins
- Installed plugins
- Advanced settings

## Git plugin, GitHub plugin, Pipeline, GitHub Integration Plugin y Docker Pipeline

The screenshot shows the Jenkins 'Available plugins' page. On the left, there are tabs for 'Updates' (with 2 updates), 'Available plugins' (highlighted with a red box), 'Installed plugins', 'Advanced settings', and 'Download progress'. On the right, the 'Download progress' section is titled 'Preparación' and lists several GitHub-related plugins: JavaMail API, SSH server, Git Push, Loading plugin extensions, Script Security, SnakeYAML API, GitHub Integration, Pipeline: GitHub, Loading plugin extensions, Authentication Tokens API, Docker Commons, Docker Pipeline, and Loading plugin extensions. Most are marked as 'Actualizado' (Updated) with green checkmarks, except for 'Script Security' and 'SnakeYAML API' which have orange circles with '1' and '2' respectively, indicating they will be updated at the next Jenkins restart.

Si no teníamos anteriormente, creamos una cuenta en GitHub y un repositorio.

The screenshot shows the GitHub 'Create a new repository' interface. At the top, there's a 'New repository' button and a toolbar with search, filter, and other icons. Below is a 'Create a new repository' section with a note about importing existing projects. The main form has two tabs: 'General' and 'Configuration'. In 'General', the owner is set to 'aparadad' and the repository name is 'ALT\_CTRL\_SUPR'. A note says 'ALT\_CTRL\_SUPR is available.' Below that, there's a 'Description' field containing 'Producto2'. In 'Configuration', there are sections for 'Choose visibility' (set to 'Public'), 'Add README' (set to 'Off'), 'Add .gitignore' (set to 'No .gitignore'), and 'Add license' (set to 'No license'). At the bottom is a large green 'Create repository' button.

Creamos dos ramas la main que será la de producción y la preprod que es la de preproducción.

The screenshot shows the GitHub repository page for 'ALT\_CTRL\_SUPR'. At the top, there are navigation links: Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. Below the header, the repository name 'ALT\_CTRL\_SUPR' is displayed with a green icon. To the right is a search bar with placeholder text 'Type to search'. Underneath the repository name, there are buttons for Pin, Watch, Fork, and Star. The repository has 3 commits and 0 stars. On the left, a sidebar shows the current branch is 'main' (indicated by a red box). A dropdown menu titled 'Switch branches/tags' shows '2 Branches' (also indicated by a red box) and '0 Tags'. The branches listed are 'main' (selected) and 'preprod' (also indicated by a red box). The main content area lists files: 'entorno de producción.pdf' (a9e6918 - 1 minute ago), 'Create README.md' (8 minutes ago), and a file with a question mark icon. On the right, sections include 'About' (Product2, Readme, Activity, 0 stars, 0 watching, 0 forks), 'Releases' (No releases published, Create a new release), and 'Packages' (No packages published, Publish your first package).

Subimos los ficheros necesarios a github a main y a preprod. Subimos nuestra aplicación de go (go.mod, main.go y la carpeta static con nuestra imagen), también subimos el dockerfile y el jenkinsfile que es el script que define el pipeline de Jenkins (Etapas Build, Push, Deploy.)

The screenshot shows the GitHub repository page for 'ALT\_CTRL\_SUPR'. At the top, there are navigation links: Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. Below the header, the repository name 'ALT\_CTRL\_SUPR' is displayed with a green icon. To the right is a search bar with placeholder text 'Type to search'. Underneath the repository name, there are buttons for Pin, Watch, Fork, and Star. The repository has 8 commits and 0 stars. On the left, a sidebar shows the current branch is 'main' (indicated by a red box). A yellow banner at the top says 'preprod had recent pushes 10 seconds ago'. A dropdown menu titled 'Switch branches/tags' shows 'main' (selected) and '2 Branches' (indicated by a red box). The branches listed are 'main' and 'preprod'. The main content area lists files: 'aparadad Delete README.md' (ec6e775 - 24 minutes ago), 'static' (Add files via upload - 34 minutes ago), 'Dockerfile' (Add files via upload - 34 minutes ago), 'Jenkinsfile' (Add files via upload - 24 minutes ago), 'go.mod' (Add files via upload - 34 minutes ago), and 'main.go' (Add files via upload - 34 minutes ago). On the right, sections include 'About' (Product2, Activity, 0 stars, 0 watching, 0 forks), 'Releases' (No releases published, Create a new release), and 'Packages' (No packages published, Publish your first package).

**Screenshot 2:**

This screenshot is identical to the one above, showing the GitHub repository page for 'ALT\_CTRL\_SUPR'. It displays recent pushes to the 'preprod' branch 25 minutes ago. The main content area lists files: 'aparadad Delete README.md' (abf721f - 25 minutes ago), 'static' (Add files via upload - 35 minutes ago), 'Dockerfile' (Add files via upload - 35 minutes ago), 'Jenkinsfile' (Add files via upload - 26 minutes ago), 'go.mod' (Add files via upload - 35 minutes ago), and 'main.go' (Add files via upload - 35 minutes ago). The right sidebar shows standard repository metrics and sections for Releases and Packages.

Para autorizar el acceso desde Jenkins iremos a la configuración de nuestra cuenta de GitHub y entramos en “developer settings” - “personal access tokens” - “tokens”

The screenshot shows the GitHub Developer Settings page. The 'Personal access tokens' section is selected. A red box highlights the 'Tokens (classic)' link under 'Personal access tokens'. Below it, a message says 'No GitHub Apps'.

Generamos el token y lo guardamos en una carpeta segura y a la vista en nuestro PC

The screenshot shows the GitHub Personal access tokens (classic) page. A red box highlights the generated token 'Jenkins-ALT\_CTRL\_SUPR'. The token details are: 'admin:repo\_hook, repo', 'Never used', and 'Expires on Mon, Jan 26 2026'. A note at the bottom explains that these tokens function like OAuth access tokens.

Token: `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`

Volvemos a Jenkins para añadir nuestras credenciales de GitHub, panel de control - administrar jenkins y nos dirigimos al menú “Credentials”

The screenshot shows the Jenkins Security page. The 'Credentials' link is highlighted with a red box. Other options shown are 'Security', 'Credential Providers', and 'Users'.

New credentials

Kind: Secret text

Scope: Global (Jenkins, nodes, items, all child items, etc)

Secret: \*\*\*\*\*

ID: github-pat-alt-ctrl-supr

Description: Token de GitHub para ALT\_CTRL\_SUPR

**Create**

Donde añadiremos un tipo: usuario (usuario en GitHub) con contraseña (el token recientemente descargado)

ID	Name	Kind	Description
github-pat-alt-ctrl-supr	Token de GitHub para ALT_CTRL_SUPR	Secret text	Token de GitHub para ALT_CTRL_SUPR
github-user-pat-token	aparadad/***** (Token de GitHub (PAT) usando formato Username/Password)	Username with password	Token de GitHub (PAT) usando formato Username/Password

Creamos el fichero Jenkinsfile que solo confirma la recepción del código y lista los archivos en el workspace.

```

@file
@tree
@diff
@diff-tree
@diff-tree-0
@diff-tree-1
@diff-tree-2
@diff-tree-3
@diff-tree-4
@diff-tree-5
@diff-tree-6
@diff-tree-7
@diff-tree-8
@diff-tree-9
@diff-tree-10
@diff-tree-11
@diff-tree-12
@diff-tree-13
@diff-tree-14
@diff-tree-15
@diff-tree-16
@diff-tree-17
@diff-tree-18
@diff-tree-19
@diff-tree-20
@diff-tree-21
@diff-tree-22
@diff-tree-23
@diff-tree-24
@diff-tree-25
@diff-tree-26
@diff-tree-27
@diff-tree-28
@diff-tree-29
@diff-tree-30
@diff-tree-31
@diff-tree-32
@diff-tree-33
@diff-tree-34
@diff-tree-35

```

The Jenkinsfile contains the following code:

```

// Jenkinsfile (Pipeline de Integración Básica)
pipeline {
    agent any
    stages {
        stage('1. Clonar y Copiar Ficheros') {
            steps {
                // La variable BRANCH_NAME es detectada automáticamente por el Pipeline Multibranch.
                echo "Pipeline activado por webhook. Rama detectada: ${env.BRANCH_NAME}"
            }
        }
        // ***** Lógica Condicional para Distinguir Ramas *****
        script {
            if (env.BRANCH_NAME == 'main') {
                echo 'Ejecutando proceso de copia para el entorno de PRODUCCIÓN (main).'
                sh 'ls -l'
            } else if (env.BRANCH_NAME == 'preprod') {
                echo 'Ejecutando proceso de copia para el entorno de PREPRODUCCIÓN (preprod).'
                sh 'ls -l'
            } else {
                // Opcional: Ignorar otras ramas
                echo "Rama ${env.BRANCH_NAME} detectada. Solo se procesan 'main' y 'preprod'."
            }
        }
    }
    post {
        always {
            echo "Proceso de CI básico finalizado para ${env.BRANCH_NAME}."
        }
    }
}

```

Como queremos manejar múltiples ramas automáticamente, creamos un **Pipeline Multibranch** en lugar de un *Pipeline* simple.

En el panel de control de Jenkins, seleccionamos nueva tarea. Ponemos el nombre ALT\_CTRL\_SUPR\_Repo\_CI.

Y seleccionamos **Multibranch Pipeline** (Pipeline Multibranch). Tenemos nuestro repositorio [https://github.com/aparadad/ALT\\_CTRL\\_SUPR.git](https://github.com/aparadad/ALT_CTRL_SUPR.git) y seleccionamos nuestras credenciales que configuramos anteriormente.

En **Build Configuration** (Configuración de Construcción). Dejamos la opción "**Discover branches**" y marcamos "**Periodically if not otherwise run**" dejandolo en un dia (es útil como *fallback*).

The image contains two screenshots of the Jenkins Pipeline Multibranch configuration interface. The top screenshot shows the 'General' configuration page for the pipeline. It includes fields for 'Display Name' (set to 'ALT\_CTRL\_SUPR\_Repo\_CI'), 'Description' (empty), and a 'Branch Sources' section for a Git repository at 'https://github.com/aparadad/ALT\_CTRL\_SUPR.git' using credentials 'aparadad/\*\*\*\*\*\*\*\* (Token de GitHub (PAT) usando formato Username/Password)'. A 'Behaviours' section contains a 'Discover branches' item. The bottom screenshot shows the full 'Configuration' page for the pipeline, including sections for 'Branch Sources' (selected), 'Build Configuration' (mode 'by Jenkinsfile', script path 'Jenkinsfile'), and 'Scan Multibranch Pipeline Triggers' (checkbox 'Periodically if not otherwise run' checked, interval '1 day').

Como vemos nos ha creado una tarea por cada rama. Ya que seleccionamos **Multibranch Pipeline** (Pipeline Multibranch).

## Configuración Final del Webhook en GitHub

El URL del **webhook** que debes configurar en GitHub para que el *Pipeline* Multibranch se active es ligeramente diferente:

Vamos a GitHub (**Settings > Webhooks**). Y añadimos un Webhooks ponemos el **Payload URL** apuntando a la URL de escaneo de nuestro controlador

<http://16.171.37.34:8080/github-webhook/> y la opción **Eventos**: Solo eventos de push.

Con esta configuración (Multibranch Pipeline), Jenkins detectará que hay actividad en la rama main o preprod y ejecutará el Jenkinsfile de esa rama automáticamente.

Una vez creado, GitHub enviará un pequeño "ping" de prueba a Jenkins.

## Webhooks / Manage webhook

Settings   Recent Deliveries

✓ 71c0adba-b707-11f0-8999-8ee3d842347a ping 2025-11-01 10:45:06 ...

Webhooks Add webhook

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our [Webhooks Guide](#).

✓ http://16.171.37.34:8080/github-we... (push)  
Last delivery was successful.

Edit   Delete

Ahora vamos a realizar un cambio en nuestras ramas para ver que se realiza ese cambio automáticamente. Vamos a Github y creamos un fichero `readme_prueba.txt` en la rama `preprod`.

ALT\_CTRL\_SUPR Public

preprod had recent pushes 8 minutes ago Compare & pull request

preprod 2 Branches 0 Tags Go to file Add file Code

This branch is 7 commits ahead of, 11 commits behind main Contribute

aparadad Create readme\_prueba.txt 7e1103f · 8 minutes ago 9 Commits

static	Add files via upload	3 days ago
Dockerfile	Add files via upload	3 days ago
Jenkinsfile	Update Jenkinsfile	1 hour ago
go.mod	Add files via upload	3 days ago
main.go	Add files via upload	3 days ago
readme_prueba.txt	Create readme_prueba.txt	8 minutes ago

README

General Webhooks / Manage webhook

Access   Recent Deliveries

Collaborators   Moderation options

Code and automation   Branches

✓ 832859ec-b70a-11f0-847e-f6c532dddec63 push 2025-11-01 11:07:03 ...

✓ 71c0adba-b707-11f0-8999-8ee3d842347a ping 2025-11-01 10:45:06 ...

[Webhooks](#) / Manage webhook[Settings](#)[Recent Deliveries](#)

✓ 832859ec-b70a-11f0-847e-f6c532ddec63 push 2025-11-01 11:07:03 ...  
Request Response 200 Redeliver Completed in 0.29 seconds.

## Headers

```
Request URL: http://16.171.37.34:8080/github-webhook/  
Request method: POST  
Accept: */*  
Content-Type: application/json  
User-Agent: GitHub-Hookshot/c012cf0  
X-GitHub-Delivery: 832859ec-b70a-11f0-847e-f6c532ddec63  
X-GitHub-Event: push  
X-GitHub-Hook-ID: 578552773  
X-GitHub-Hook-Installation-Target-ID: 1085230339  
X-GitHub-Hook-Installation-Target-Type: repository
```

## Payload

```
{  
  "ref": "refs/heads/preprod",  
  "before": "25f285ede0e0b53dccebbc230cd2fe20a781df9a",  
  "after": "7e1103ffd3fa23222456a4b09acf48d6f47a8d82",  
  "repository": {  
    "id": 1085230339,  
    "node_id": "R_kgDOQK9NAw",  
    "name": "ALT_CTRL_SUPR",  
    "full_name": "aparadad/ALT_CTRL_SUPR",  
    "private": false,  
    "owner": {  
      "name": "aparadad",  
      "email": "aparadad@uoc.edu",  
      "login": "aparadad",  
      "id": 240777356,
```

```

},
  "created": false,
  "deleted": false,
  "forced": false,
  "base_ref": null,
  "compare": "https://github.com/aparadad/ALT_CTRL_SUPR/compare/25f285ede0e0...7e1103ffd3fa",
  "commits": [
    {
      "id": "7e1103ffd3fa23222456a4b09acf48d6f47a8d82",
      "tree_id": "3b68d963ed2bc165cd35f9ff22843690257b83a",
      "distinct": true,
      "message": "Create readme_prueba.txt",
      "timestamp": "2025-11-01T11:07:02+01:00",
      "url": "https://github.com/aparadad/ALT_CTRL_SUPR/commit/7e1103ffd3fa23222456a4b09acf48d6f47a8d82",
      "author": {
        "name": "aparadad",
        "email": "aparadad@uoc.edu",
        "username": "aparadad"
      },
      "committer": {
        "name": "GitHub",
        "email": "noreply@github.com",
        "username": "web-flow"
      },
      "added": [
        "readme_prueba.txt"
      ],
      "removed": [
      ],
      "modified": [
      ]
    }
  ],
  "head_commit": {
    "id": "7e1103ffd3fa23222456a4b09acf48d6f47a8d82",
    "tree_id": "3b68d963ed2bc165cd35f9ff22843690257b83a",
    "distinct": true,
    "message": "Create readme_prueba.txt",
    "timestamp": "2025-11-01T11:07:02+01:00"
  }
}

```

En jenkins lo podemos ver en el Historial de trabajos, pinchamos en el último, en cambios y vemos:

S	Ejecución	Tiempo desde	Estado
●	ALT_CTRL_SUPR_Repo_CI » preprod #2	1 Seg	estable
●	ALT_CTRL_SUPR_Repo_CI » preprod #1	54 Min	estable
●	ALT_CTRL_SUPR_Repo_CI » main #1	54 Min	estable

**Changes**

#2 (1 nov 2025 10:18:03)

1. Create readme\_prueba.txt — noreply / githubweb

En la maquina Ubuntu lo podemos ver en nuestro workspace.

```
ubuntu@ip-172-31-40-234:/var/lib/jenkins/workspace/ALT_CTRL_SUPR_Repo_CI_prepod$ ls
Dockerfile Jenkinsfile go.mod main.go readme_prueba.txt static
ubuntu@ip-172-31-40-234:/var/lib/jenkins/workspace/ALT_CTRL_SUPR_Repo_CI_prepod$ cd /var/lib/jenkins/workspace/ALT-CTRL-
-SUPR-CD-Pipeline
ubuntu@ip-172-31-40-234:/var/lib/jenkins/workspace/ALT-CTRL-SUPR-CD-Pipeline$ ls
Dockerfile Jenkinsfile go.mod main.go static
ubuntu@ip-172-31-40-234:/var/lib/jenkins/workspace/ALT-CTRL-SUPR-CD-Pipeline$ |
```

## 7. Instalar MiniKube en nuestra máquina virtual.

Seguimos las indicaciones de la página oficial

### 1 Installation

Click on the buttons that describe your target platform. For other architectures, see [the release page](#) for a complete list of minikube binaries.

Operating system	<input checked="" type="button"/> Linux	<input type="button"/> macOS	<input type="button"/> Windows		
Architecture	<input checked="" type="button"/> x86-64	<input type="button"/> ARM64	<input type="button"/> ARMv7	<input type="button"/> ppc64	<input type="button"/> S390x
Release type	<input checked="" type="button"/> Stable				
Installer type	<input checked="" type="button"/> Binary download				
	<input type="button"/> Debian package				
	<input type="button"/> RPM package				

To install the latest minikube **stable** release on **x86-64 Linux** using **binary download**:

```
curl -LO https://github.com/kubernetes/minikube/releases/latest/download/minikube-linux-amd64
sudo install minikube-linux-amd64 /usr/local/bin/minikube && rm minikube-linux-amd64
```

Copy

```
root@ip-172-31-39-237:/home/ubuntu# curl -LO https://github.com/kubernetes/minikube/releases/latest/download/minikube-linux-amd64
% Total    % Received % Xferd  Average Speed   Time   Time     Current
          Dload  Upload Total Spent   Left Speed
0       0      0      0      0      0      0 ---:--- ---:--- ---:--- 0
0       0      0      0      0      0      0 ---:--- ---:--- ---:--- 0
100  133M  100  133M  0      0  138M  0 ---:--- ---:--- ---:--- 138M
root@ip-172-31-39-237:/home/ubuntu# ls
minikube-linux-amd64
root@ip-172-31-39-237:/home/ubuntu# install minikube-linux-amd64 /usr/local/bin/minikube && rm minikube-linux-amd64
```

```
root@ip-172-31-39-237:/home/ubuntu# minikube version
minikube version: v1.37.0
commit: 65318f4cff9c12cc87ec9eb8f4cdd57b25047f3
```

Al ejecutar **minikube start**, nos hemos topado con un error que no está instalado docker.

### Procedemos a instalarlo

```
ubuntu@ip-172-31-39-237:~$ sudo apt install -y curl apt-transport-https docker.io conntrack
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
curl is already the newest version (8.5.0-2ubuntu10.6).
curl set to manually installed.
The following packages were automatically installed and are no longer required:
  linux-aws-6.14-headers-6.14.0-1011 linux-aws-6.14-tools-6.14.0-1011 linux-headers-6.14.0-1011-aws linux-image-6.14.0-1011-aws linux-modules-6.14.0-1011-aws
  linux-tools-6.14.0-1011-aws
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  docker
ubuntu@ip-172-31-39-237:~$ minikube start --driver=docker
* minikube v1.37.0 on Ubuntu 24.04
* Using the docker driver based on user configuration
X Exiting due to PROVIDER_DOCKER_NEWRGP: "docker version --format <no value>:<no value>" exit status 1: permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://127.0.0.1:2375/v1.50/version": dial unix /var/run/docker.sock: connect: permission denied
* Suggestion: Add your user to the 'docker' group: 'sudo usermod -aG docker $USER && newgrp docker'
* Documentation: https://docs.docker.com/engine/install/linux-postinstall/
```

```
ubuntu@ip-172-31-39-237:~$ minikube start --driver=docker
* minikube v1.37.0 on Ubuntu 24.04
* Using the docker driver based on user configuration

X Exiting due to RSRC_INSUFFICIENT_CONTAINER_MEMORY: docker only has 914MiB available, less than the required 1800MiB for Kubernetes
```

Hemos cambiado la instancia de t3.micro a t3.medium. Sigue dandonos error así que hemos ampliado el volumen de 8GB a 20GB

Utilizamos **minikube start --driver=docker --memory=2048 --cpu=2 --preload=false** para restringir el uso de memory y de cpus que usará minikube.

```
Last login: Fri Nov  7 11:42:41 2025 from 3.120.181.44
ubuntu@ip-172-31-47-108:~$ df -h /
Filesystem      Size  Used Avail Use% Mounted on
/dev/root       19G   5.2G  14G  29% /
ubuntu@ip-172-31-47-108:~$ minikube start --driver=docker --memory=2048 --cpus=2 --preload=false
* minikube v1.37.0 on Ubuntu 24.04
* Using the docker driver based on user configuration
* Using Docker driver with root privileges
* Starting "minikube" primary control-plane node in "minikube" cluster
* Pulling base image v0.0.48 ...
* Creating docker container (CPUs=2, Memory=2048MB) ...
    > kubectl.sha256: 64 B / 64 B [=====] 100.00% ? p/s 0s
    > kubeadm.sha256: 64 B / 64 B [=====] 100.00% ? p/s 0s
    > kubelet.sha256: 64 B / 64 B [=====] 100.00% ? p/s 0s
    > kubectl: 57.75 MiB / 57.75 MiB [=====] 100.00% 252.01 MiB p/s 400ms
    > kubelet: 56.45 MiB / 56.45 MiB [=====] 100.00% 208.42 MiB p/s 500ms
    > kubeadm: 70.60 MiB / 70.60 MiB [=====] 100.00% 141.70 MiB p/s 700ms
* Preparing Kubernetes v1.34.0 on Docker 28.4.0 ...
* Configuring bridge CNI (Container Networking Interface) ...
* Verifying Kubernetes components...
- Using image gcr.io/k8s-minikube/storage-provisioner:v5
* Enabled addons: storage-provisioner, default-storageclass
* kubectl not found. If you need it, try: 'minikube kubectl -- get pods -A'
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
ubuntu@ip-172-31-47-108:~$ █
```

## 8. Instalar Kubernetes en nuestra máquina virtual.

[Documentación](#) / [Tareas](#) / [Instalar herramientas](#) / [Herramientas incluidas](#) / [Instalar y configurar kubectl en Linux](#)

### Instalar y configurar kubectl en Linux

#### Antes de empezar

Debes usar una versión de kubectl que esté dentro de una diferencia de versión menor de tu clúster. Por ejemplo, un cliente v1.34 puede comunicarse con v1.33, v1.34, y v1.35 del plano de control. El uso de la última versión de kubectl ayuda a evitar problemas inesperados.

#### Instalar kubectl en Linux

Seguimos las instrucciones de la página oficial.

```
curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
```

Este comando descarga la versión estable más reciente de kubectl directamente desde el repositorio oficial de Kubernetes.

```
ubuntu@ip-172-31-47-108:~$ curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
% Total    % Received % Xferd  Average Speed   Time   Time     Current
          Dload  Upload Total   Spent    Left Speed
100 138 100 138  0    0  895  0 --:--:-- --:--:-- --:--:-- 896
100 57.7M 100 57.7M  0    0 155M  0 --:--:-- --:--:-- --:--:-- 155M
ubuntu@ip-172-31-47-108:~$ █
```

Una vez descargado el archivo, es importante **comprobar que no ha sido modificado o dañado durante la descarga**.

Para ello, realizamos los siguientes pasos:

- Descargar el archivo de comprobación de kubectl:  
`curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl.sh  
256"`
- Validar el binario kubectl con el archivo de comprobación:  
`echo "$(cat kubectl.sha256) kubectl" | sha256sum  
--check`

Si es válido, la salida será “Kubectl: OK” como vemos en la siguiente captura.

```
ubuntu@ip-172-31-47-108:~$ echo "$(cat kubectl.sha256) kubectl" | sha256sum --check  
kubectl: OK  
ubuntu@ip-172-31-47-108:~$ █
```

Una vez verificada la integridad del archivo, procedemos a instalar kubectl en el sistema.

Para ello, ejecutamos el siguiente comando:

```
sudo install -o root -g root -m 0755 kubectl  
/usr/local/bin/kubectl
```

Una vez completada la instalación, verificamos que la herramienta kubectl se ha instalado correctamente.

Para ello, ejecutamos el siguiente comando en la terminal:

```
kubectl version --client
```

El resultado muestra la versión del cliente de Kubernetes instalada en la máquina.

```
ubuntu@ip-172-31-47-108:~$ kubectl version --client  
Client Version: v1.34.1  
Kustomize Version: v5.7.1  
ubuntu@ip-172-31-47-108:~$ █
```

## 9. Configurar Kubernetes para que nuestro contenedor se encuentre en alta disponibilidad (balanceo de carga entre mínimo dos instancias).

En este paso configuramos Kubernetes para que nuestro contenedor se ejecute con **alta disponibilidad**, garantizando que el servicio siga activo incluso si alguna instancia falla.

Esto se logra mediante la creación de un **deployment** con varias réplicas y la exposición del servicio a través de un **LoadBalancer**.

Primero, la imagen Docker de la aplicación utilizando el Dockerfile del proyecto y la etiqueta como **altctrlsupr/app:dev**.

```
docker build -t altctrlsupr/app:dev .
```

```
ubuntu@ip-172-31-47-108:~/Producto2$ ls
Dockerfile  k8s.yaml  main.go  static
ubuntu@ip-172-31-47-108:~/Producto2$ docker build -t altctrlsupr/app:dev .
```

El listado confirma que la imagen **altctrlsupr/app:dev** está almacenada dentro del Docker de Minikube.

```
ubuntu@ip-172-31-47-108:~/Producto2$ docker images
REPOSITORY          TAG      IMAGE ID   CREATED        SIZE
altctrlsupr/app     dev      d11322f8cc3d  About a minute ago  7.03MB
```

A continuación, creamos el archivo **k8s.yaml**, contiene la configuración necesaria para desplegar nuestra aplicación en Kubernetes. Definimos un **Deployment** con varias réplicas del contenedor y un **Service** tipo LoadBalancer que permite acceder a la aplicación desde fuera del clúster. (anexo 1)

```
kubectl apply -f k8s.yaml
```

```
ubuntu@ip-172-31-47-108:~/Producto2$ kubectl apply -f k8s.yaml
deployment.apps/uoc created
service/uoc created
```

Este comando crea tanto el Deployment como el Service definidos en el archivo.

La salida confirma que Kubernetes ha creado correctamente los recursos:

- deployment.apps/uoc created
- service/uoc created

Finalmente, verificamos que las tres réplicas del contenedor están funcionando correctamente con el siguiente comando:

```
kubectl get pods
```

```
ubuntu@ip-172-31-47-108:~/Producto2$ kubectl get pods
NAME                  READY   STATUS    RESTARTS   AGE
uoc-74c8985855-9vrnx  1/1     Running   0          24s
uoc-74c8985855-fsmrd  1/1     Running   0          24s
uoc-74c8985855-msktz  1/1     Running   0          24s
```

El resultado muestra tres pods en estado Running, confirmando que el balanceo de carga está activo y que el servicio se encuentra en **alta disponibilidad**.

## 10. Comprobar que el balanceo funciona correctamente.

Una vez configurado el *deployment* con tres réplicas, comprobamos que el **balanceador de carga** funciona correctamente, distribuyendo las peticiones entre los diferentes pods del clúster.

Para ello, primero iniciamos un túnel en **Minikube** que permite acceder a los servicios tipo *LoadBalancer* desde el exterior:

```
ubuntu@ip-172-31-47-108:~$ minikube tunnel
Status:
  machine: minikube
  pid: 8771
  route: 10.96.0.0/12 -> 192.168.49.2
  minikube: Running
  services: [uoc]
errors:
  minikube: no errors
  router: no errors
  loadbalancer emulator: no errors
```

El comando crea una ruta entre la red interna de Minikube y la máquina anfitriona. En la salida podemos observar que el servicio uoc está activo y sin errores.

A continuación, listamos los servicios disponibles con el siguiente comando:

```
kubectl get services
```

El resultado muestra el servicio uoc con una dirección IP externa asignada por el balanceador de carga.

```
ubuntu@ip-172-31-47-108:~/Producto2$ kubectl get services
NAME      TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
kubernetes  ClusterIP  10.96.0.1      <none>        443/TCP      7d22h
uoc       LoadBalancer  10.111.131.241  10.111.131.241  80:32279/TCP  2m8s
```

Para verificar que las peticiones se reparten correctamente entre los pods, realizamos múltiples solicitudes HTTP al servicio utilizando un bucle:

```
for i in {1..10}; do curl http://10.111.131.241; done
```

Cada solicitud devuelve la página creada en el Producto 1.

```
ubuntu@ip-172-31-47-108:~/Producto2$ for i in {1..10}; do curl http://10.111.131.241; done
<!doctype html>
<html lang="es">
<head><meta charset="utf-8"><title>UOC Go Web</title>
<style>
  body { font-family:sans-serif; margin:2rem }
  img { max-width:360px; height:auto; border-radius:12px }
</style>
</head>
<body>
  <h1>Soy alumno de la UOC</h1>
  
</body></html><!doctype html>
<html lang="es">
<head><meta charset="utf-8"><title>UOC Go Web</title>
<style>
  body { font-family:sans-serif; margin:2rem }
  img { max-width:360px; height:auto; border-radius:12px }
</style>
</head>
```

Finalmente, verificamos en los **logs de cada pod** cómo se distribuyen las peticiones de forma equilibrada:

```
kubectl logs uoc-54d7b5b8f-lsvck --follow
```

```
kubectl logs uoc-54d7b5b8f-w241k --follow
```

```
kubectl logs uoc-54d7b5b8f-wqf8x --follow
```

En los registros se puede observar que las peticiones GET se reparten entre los tres pods en diferentes momentos:

Esto confirma que el balanceo de carga está funcionando correctamente, ya que las solicitudes se distribuyen entre las tres réplicas del contenedor `altctrlsupr/app:dev`, garantizando una alta disponibilidad y tolerancia a fallos.

## Bibliografía

- Jenkins. (s. f.). *Installing Jenkins*. Recuperado el 16 de noviembre de 2025, de <https://www.jenkins.io/doc/book/installing/>
- kubectl. (s. f.). *Instalar y configurar kubectl en Linux*. En Kubernetes Documentation. Recuperado el 16 de noviembre de 2025 de <https://kubernetes.io/es/docs/tasks/tools/include/install-kubectl-linux/>
- Minikube. (s. f.). *Get Started – Windows / x86-64 stable (.exe download)*. Recuperado el 16 de noviembre de 2025, de <https://minikube.sigs.k8s.io/docs/start/?arch=%2Fwindows%2Fx86-64%2Fstable%2F.exe+download>

## Anexo

### Anexo 1 - k8s.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: uoc
spec:
  replicas: 3
  selector:
    matchLabels:
      app: uoc
  template:
    metadata:
      labels:
        app: uoc
    spec:
      containers:
        - name: app
          image: altctrlsupr/app:dev
          imagePullPolicy: Never # <- importante para usar la
          imagen local de Minikube
          ports:
            - containerPort: 8080 # <- cambia si tu app usa otro
            puerto
          readinessProbe:
            httpGet: { path: "/", port: 8080 }
            initialDelaySeconds: 5
          livenessProbe:
            httpGet: { path: "/", port: 8080 }
            initialDelaySeconds: 10
---
apiVersion: v1
kind: Service
metadata:
  name: uoc
spec:
  type: LoadBalancer
  selector:
    app: uoc
  ports:
    - port: 80
      targetPort: 8080
```