# DOCUMENTACIÓN DE APLICACIONES Y CONTROL DE VERSIONES



## ÍNDICE

TΑ	REA 2	. 2
	INSTALACIÓN DE LA HERRAMIENTA GIT	. 2
	COMANDOS BÁSICOS PARA SUBIR UN PROYECTO A GITHUB	

En esta práctica veremos cómo podemos instalar GitLab en nuestro equipo y como podemos subir un proyecto como repositorio a GitHub.

### TAREA 2

### INSTALACIÓN DE LA HERRAMIENTA GIT

En la terminal de Ubuntu haremos uso de este comando para instalar "openssh", lo necesitamos para el funcionamiento de GitLab. *Fig. 1.* 

```
root@cristianpopc:/home/cristianpo# apt-get install -y curl openssh-server ca-certificates tzdata
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
```

Fig. 1. Comando de instalación de OpenSSH

Usaremos este comando para instalar "postfix", necesario para GitLab. Fig. 2.

```
root@cristianpopc:/home/cristianpo# sudo apt-get install -y postfix
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
```

Fig. 2. Comando de instalación de Postfix

Mediante este comando obtenemos el paquete donde están los archivos con lo que poder instalar GitLab. Fig. 3.

```
root@cristianpopc:/home/cristianpo# curl -sS https://packages.gitlab.com/install/repositories/git lab/gitlab-ce/script.deb.sh | bash
Detected operating system as Ubuntu/jammy.
Checking for curl...
Detected curl...
Checking for gpg...
Detected gpg...
Running apt-get update... done.
Installing apt-transport-https... done.
Installing /etc/apt/sources.list.d/gitlab_gitlab-ce.list...done.
Importing packagecloud gpg key... done.
Running apt-get update... done.
The repository is setup! You can now install packages.
```

Fig. 3. Comando para obtención de paquete GitLab

Una vez realizados los comandos anteriores ya podemos realizar el comando de instalación de GitLab. Fig. 4. Comando de instalación de GitLab

```
root@cristlanpopc:/home/cristlanpo# apt-get install gitlab-ce
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
linux-headers-6.2.0-34-generic linux-hwe-6.2-headers-6.2.0-34 linux-inage-6.2.0-34-generic
linux-modules-6.2.0-34-generic linux-modules-extra-6.2.0-34-generic
Utilice «apt autoremove» para eliminarlos.

Se instalarán los siguientes paquetes NUEVOS:
gitlab-ce
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 5 no actualizados.

Se necesita descargar 1.102 MB de archivos.

Se utilizarán 3.121 MB de espacio de disco adicional después de esta operación.
Des:1 https://packages.gitlab.com/gitlab/gitlab-ce/ubuntu jammy/main amd64 gitlab-ce amd64 16.8.2-ce
0.9 [1.102 MB]
Descargados 1.102 MB en 19min 7s (961 kB/s)
Seleccionando el paquete gitlab-ce previamente no seleccionado.
(Leyendo la base de datos ... 247437 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../gitlab-ce_16.8.2-ce.0_amd64.deb ...
Desempaquetando gitlab-ce (16.8.2-ce.0) ...
Configurando gitlab-ce (16.8.2-ce.0) ...
It looks like Gitlab has not been configured yet; skipping the upgrade script.
```

Fig. 4.

En el directorio "/etc/gitlab" editamos el fichero "gitlab.rb" (Fig. 5.) En el que editamos el atributo "external\_url" para que apunte a nuestra dirección IP y al puerto que nosotros deseemos (Fig. 6.). Esta será la dirección URL que deberemos introducir en el buscador para acceder a nuestro GitLab.

```
root@cristianpopc:/etc/gitlab# nano gitlab.rb
external_url 'http://10.0.2.15:3005'
```

Fig. 5 y 6. Archivo y atributo de configuración de GitLab

Usamos este comando para actualizar las configuraciones de GitLab y que se apliquen los cambios realizados. *Fig.* 7.

```
root@cristianpopc:/etc/gitlab# gitlab-ctl reconfigure
```

Fig. 7. Reinicio de la configuración de GitLab

Por último comprobamos que se ha realizado la instalación y configuración correctamente accediendo a GitLab desde el navegador. Fig. 8.

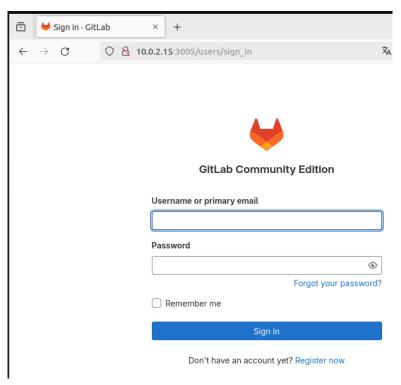


Fig. 8. Correcto funcionamiento de GitLab

# COMANDOS BÁSICOS PARA SUBIR UN PROYECTO A GITHUB

Desde la página de inicio de nuestro perfil de GitHub, seleccionamos la opción de "New" en la sección de nuestros repositorios. *Fig. 9.* 

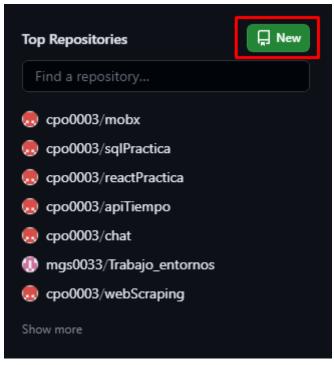


Fig. 9. Botón de creación del repositorio

En la pantalla de creación le damos un nombre al repositorio y lo creamos. Fig. 10.

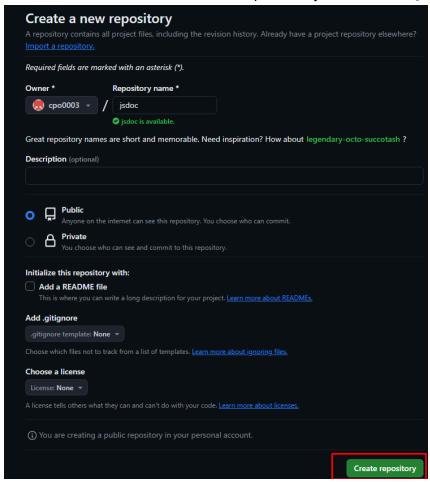


Fig. 10. Creación del repositorio

Una vez que tenemos el repositorio creado en nuestro proyecto creamos el archivo ".gitignore" en el que indicaremos todos aquellos archivos y directorios que no queremos que se suban a GitHub, en nuestro caso la carpeta "node\_modules". Fig. 8.



Fig. 11. Archivo .gitignore

Ahora desde la terminal estando posicionados en el directorio del proyecto realizamos este comando para iniciar Git en el proyecto. *Fig. 12.* 

```
PS C:\Users\Cristian\Desktop\jsdoc-main> git init
Initialized empty Git repository in C:/Users/Cristian/Desktop/jsdoc-main/.git/
```

Fig. 12. Comando git init

Mediante este comando añadiremos los archivos del proyecto a el "push" que realizaremos. *Fig. 13.* 

```
PS C:\Users\Cristian\Desktop\jsdoc-main> git add .
warning: in the working copy of 'about.html', LF will be replaced by CRLF the next time
warning: in the working copy of 'contact.html', LF will be replaced by CRLF the next tim
e Git touches it
warning: in the working copy of 'css/styles.css', LF will be replaced by CRLF the next t
ime Git touches it
warning: in the working copy of 'docs/Task.html', LF will be replaced by CRLF the next t
ime Git touches it
warning: in the working copy of 'docs/TaskManager.html', LF will be replaced by CRLF the
 next time Git touches it
warning: in the working copy of 'docs/about.js.html', LF will be replaced by CRLF the ne
xt time Git touches it
warning: in the working copy of 'docs/contact.js.html', LF will be replaced by CRLF the
next time Git touches it
warning: in the working copy of 'docs/fonts/OpenSans-Bold-webfont.svg', LF will be repla
ced by CRLF the next time Git touches it
warning: in the working copy of 'docs/fonts/OpenSans-BoldItalic-webfont.svg', LF will be
replaced by CRLF the next time Git touches it
```

Fig. 13. Comando git add

Con este comando le damos un nombre al "push" que vamos a realizar. Fig. 14.

```
PS C:\Users\Cristian\Desktop\jsdoc-main> git commit -m "Primer commit"
[master (root-commit) 0e54237] Primer commit
 46 files changed, 15452 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 about.html
 create mode 100644 contact.html
 create mode 100644 css/styles.css
 create mode 100644 docs/Task.html
 create mode 100644 docs/TaskManager.html
 create mode 100644 docs/about.js.html
 create mode 100644 docs/contact.js.html
 create mode 100644 docs/fonts/OpenSans-Bold-webfont.eot
 create mode 100644 docs/fonts/OpenSans-Bold-webfont.svg
 create mode 100644 docs/fonts/OpenSans-Bold-webfont.woff
 create mode 100644 docs/fonts/OpenSans-BoldItalic-webfont.eot
 create mode 100644 docs/fonts/OpenSans-BoldItalic-webfont.svg
 create mode 100644 docs/fonts/OpenSans-BoldItalic-webfont.woff
 create mode 100644 docs/fonts/OpenSans-Italic-webfont.eot
```

Fig. 14. Commit del git

Ahora con este comando le indicamos en qué repositorio creado de nuestro GitHub queremos que se realice el "push". *Fig. 15.* 

```
PS C:\Users\Cristian\Desktop\jsdoc-main> git remote add origin https://github.com/cpo000

3/jsdoc.git
```

Fig. 15. Comando git remote add origin

Por último utilizamos este comando con el que ya realizamos el "push" del proyecto al repositorio de GitHub. *Fig. 16*.

```
PS C:\Users\Cristian\Desktop\jsdoc-main> git push -u origin master
Enumerating objects: 56, done.
Counting objects: 100% (56/56), done.
Delta compression using up to 12 threads
Compressing objects: 100% (54/54), done.
Writing objects: 100% (56/56), 3.81 MiB | 1.80 MiB/s, done.
Total 56 (delta 7), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (7/7), done.
To https://github.com/cpo0003/jsdoc.git
* [new branch] master -> master
branch 'master' set up to track 'origin/master'.
```

Fig. 16. Comando git push