CENTRE ESPECÍFIC
D´EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

UNIT 5.
ACTIVITY 2: GITLAB

Web Applications Deployment
CFGS DAW

Important: this activity is not mandatory and does not compute for the final grade.

Importante: esta actividad no es obligatoria y no cuenta para la nota final.

Author: Carlos Cacho López

Reviewed by: Lionel Tarazón Alcocer

lionel.tarazon@ceedcv.es

2019/2020

License

Nomenclature

During this unit we are going to use special symbols to distinct some important elements.

This symbols are:

📚 Important

⚡ Attention

📢 Interesting

INDEX

UT05. VERSION CONTROL SYSTEM
ACTIVITY: GITLAB

1. INTRODUCTION

In this activity we are going to explain about the platform GitLab, create an account and explain how to use it.

To do this activity you can use your own physical machine or a virtual machine (Linuxserver for instance). I will do this activity in Ubuntu. Remember that we need Internet connection.

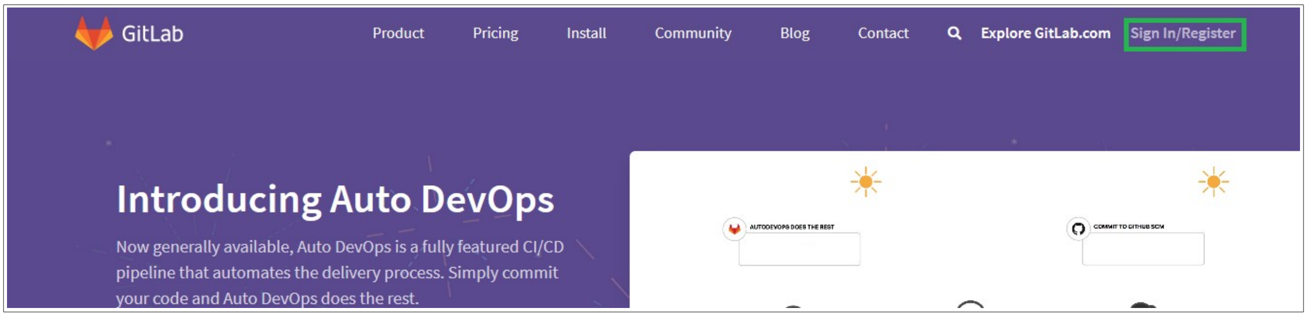
2. GITLAB

GitLab is a web-based Git-repository manager with MIT license which it provides wiki, issue-tracking and CI/CD pipeline.

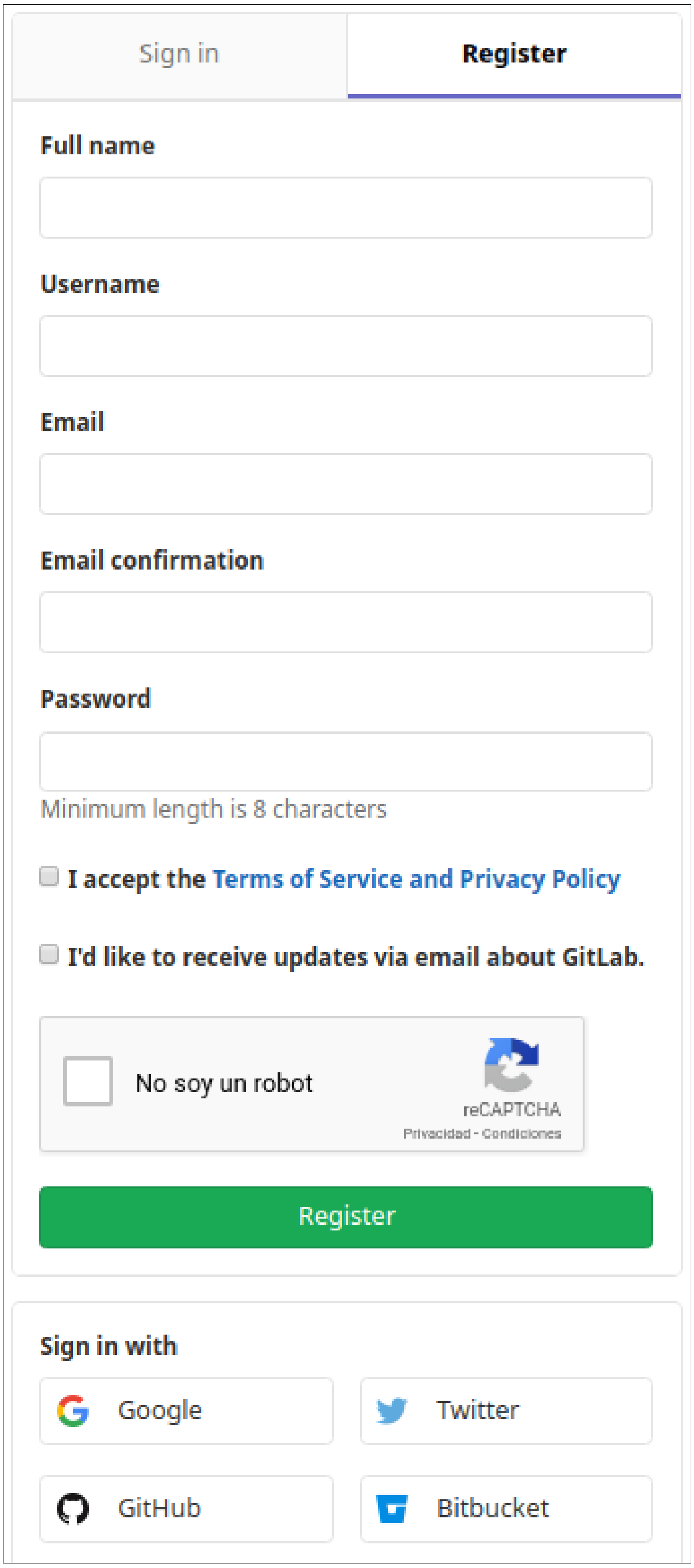
It offers an integrated solution covering the whole DevOps lifecycle. (We talk about it in unit 7). You can find information about it in the GitLab official web site: https://about.gitlab.com/stages-devops-lifecycle/

2.1 Register

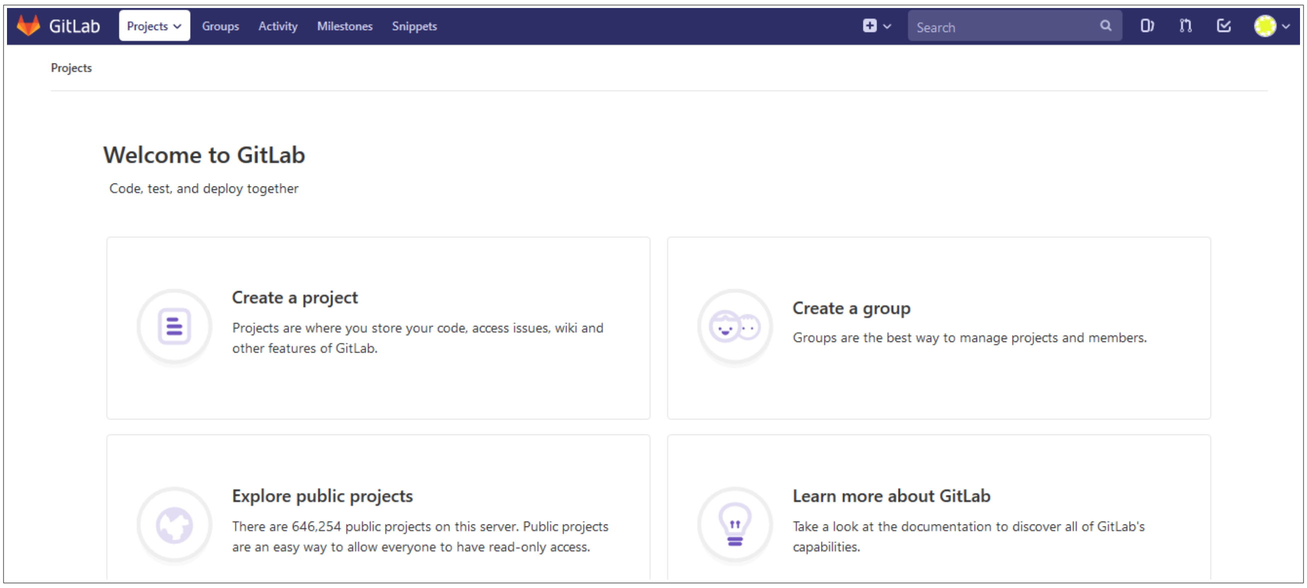
First, we have to register in GitLab so we go to the official site: https://about.gitlab.com/, click on ***Sign In/Register***:


GitLab
Product
Pricing
Install
Community
Blog
Contact
Explore GitLab.com
Sign In/Register
Introducing Auto DevOps
Now generally available, Auto DevOps is a fully featured CI/CD pipeline that automates the delivery process. Simply commit your code and Auto DevOps does the rest.
AUTODEVOPS DOES THE REST
COMMIT TO GITHUB SCM

Fill the form and click on ***Register***:



Check your email to confirm our account, sign in and we will be in:



Welcome to GitLab

Code, test, and deploy together

Create a project

Projects are where you store your code, access issues, wiki and other features of GitLab.

Create a group

Groups are the best way to manage projects and members.

Explore public projects

There are 646,254 public projects on this server. Public projects are an easy way to allow everyone to have read-only access.

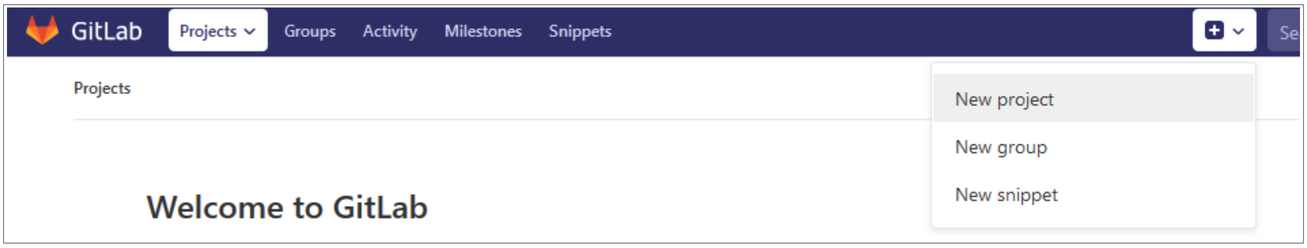
Learn more about GitLab

Take a look at the documentation to discover all of GitLab's capabilities.

Now we are going to work with GitLab following the tutorials in https://docs.gitlab.com.

2.2 Create a project

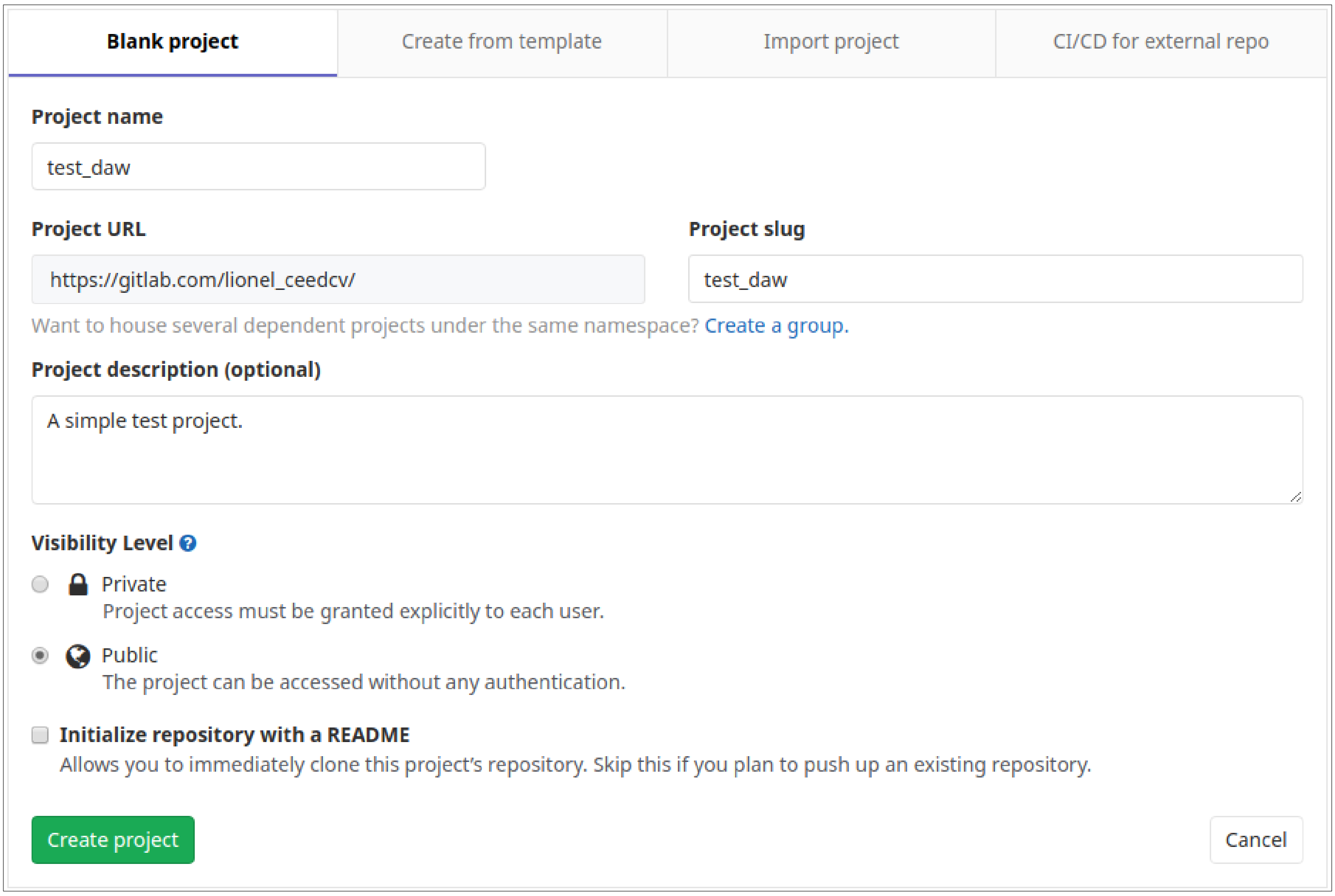
To create a new project from the platform we have to click on the button **+** and then ***New Project***:



Welcome to GitLab

We can create a blank project, from a template or import from GitHub, Bitbucket, etc, even use continuous integration (CI) and continuous delivery (CD) (we will see these concepts in unit 7).

Now, we are going to create a blank project. For that, we have to indicate the project name (*test_daw* in my case), the description and the visibility level, (I have choose public so all of you can see it) and click on the button ***Create project***:



Project name

test_daw

Project URL

https://gitlab.com/lionel_ceedcv/

Project slug

test_daw

Want to house several dependent projects under the same namespace? Create a group.

Project description (optional)

A simple test project.

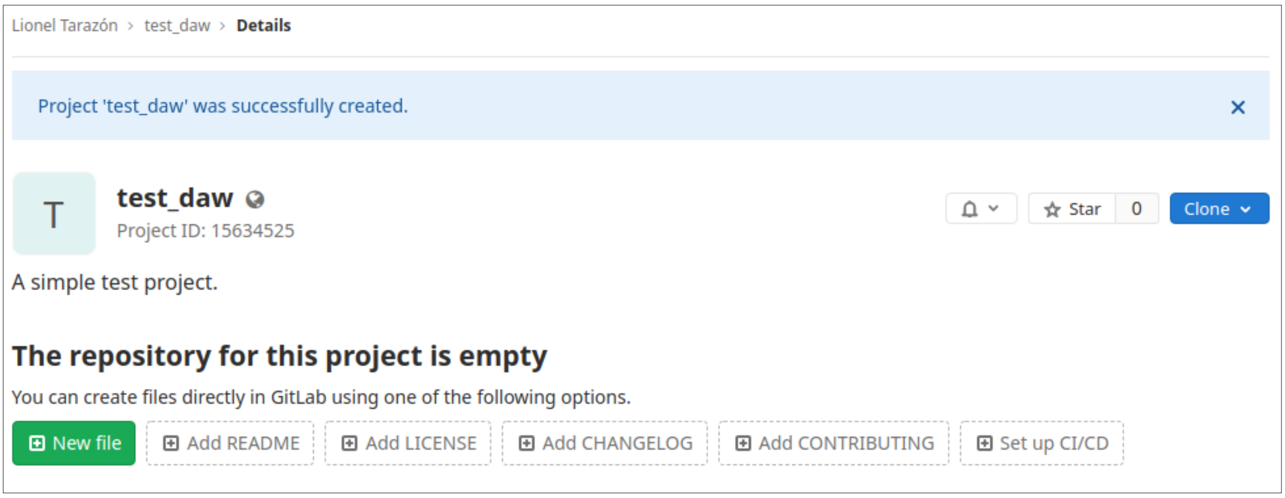
Visibility Level

○ Private
Project access must be granted explicitly to each user.

● Public
The project can be accessed without any authentication.

□ **Initialize repository with a README**
Allows you to immediately clone this project's repository. Skip this if you plan to push up an existing repository.

And our project is created:



Project 'test_daw' was successfully created.

test_daw

Project ID: 15634525

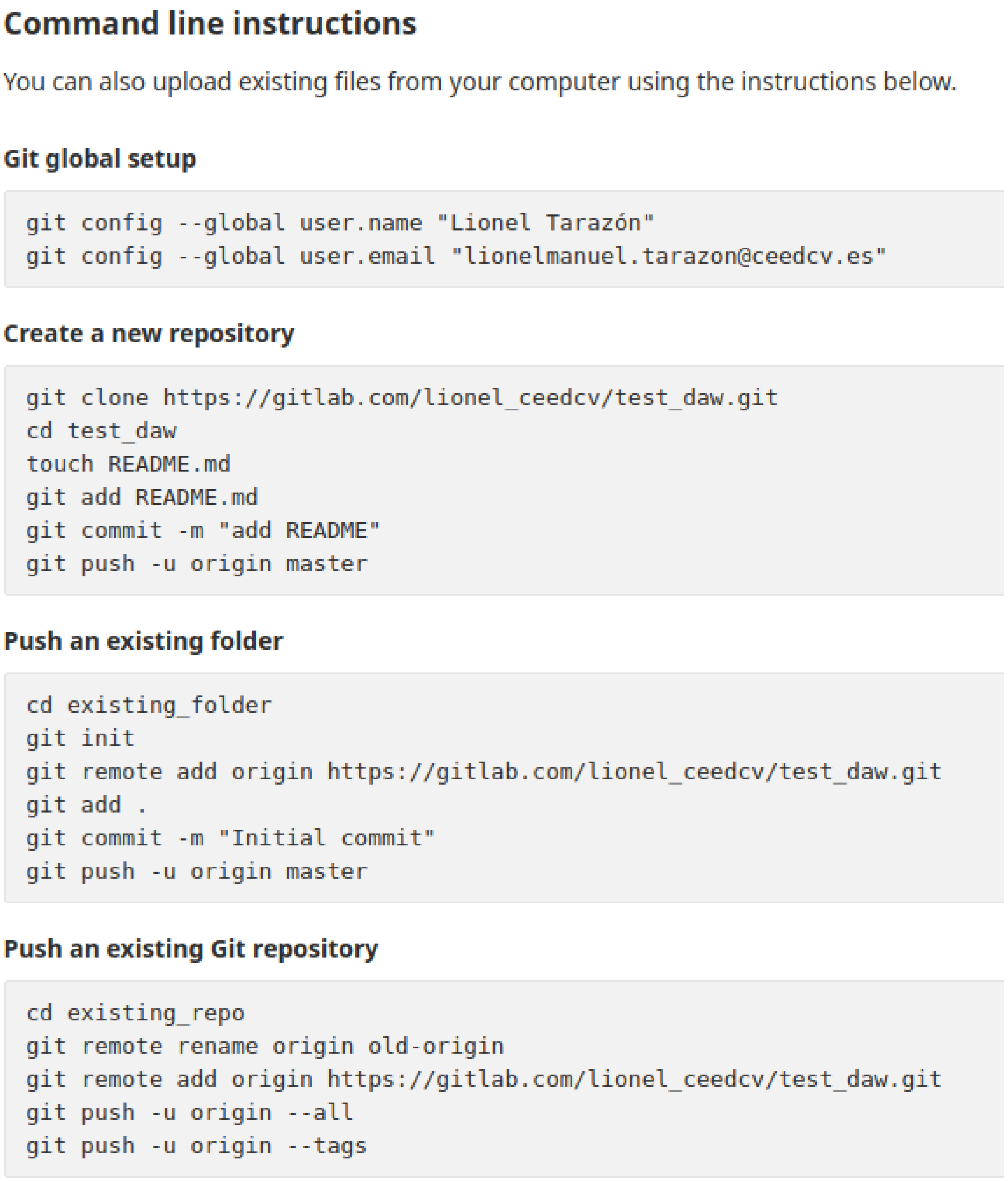
A simple test project.

The repository for this project is empty

You can create files directly in GitLab using one of the following options.

We can see buttons to add files, a Readme file, license, etc.

Also some command line instructions to connect and manage the repository from your computer.



Command line instructions

You can also upload existing files from your computer using the instructions below.

Git global setup

```
git config --global user.name "Lionel Tarazón"
git config --global user.email "lionelmanuel.tarazon@ceedcv.es"
```

Create a new repository

```
git clone https://gitlab.com/lionel_ceedcv/test_daw.git
cd test_daw
touch README.md
git add README.md
git commit -m "add README"
git push -u origin master
```

Push an existing folder

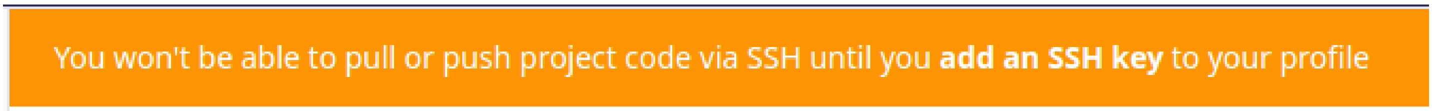
```
cd existing_folder
git init
git remote add origin https://gitlab.com/lionel_ceedcv/test_daw.git
git add .
git commit -m "Initial commit"
git push -u origin master
```

Push an existing Git repository

```
cd existing_repo
git remote rename origin old-origin
git remote add origin https://gitlab.com/lionel_ceedcv/test_daw.git
git push -u origin --all
git push -u origin --tags
```

2.3 Add an SSH Key

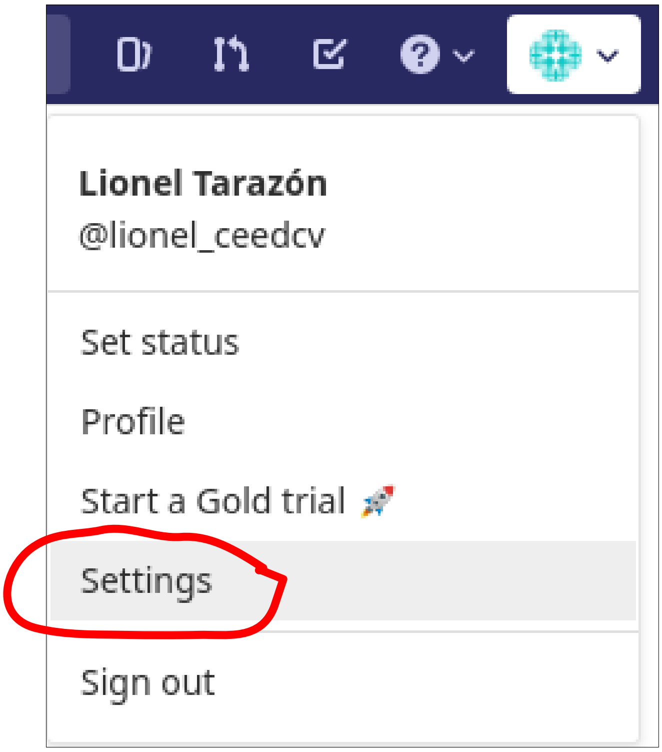
If we take a look at the top of the page we can see this warning:

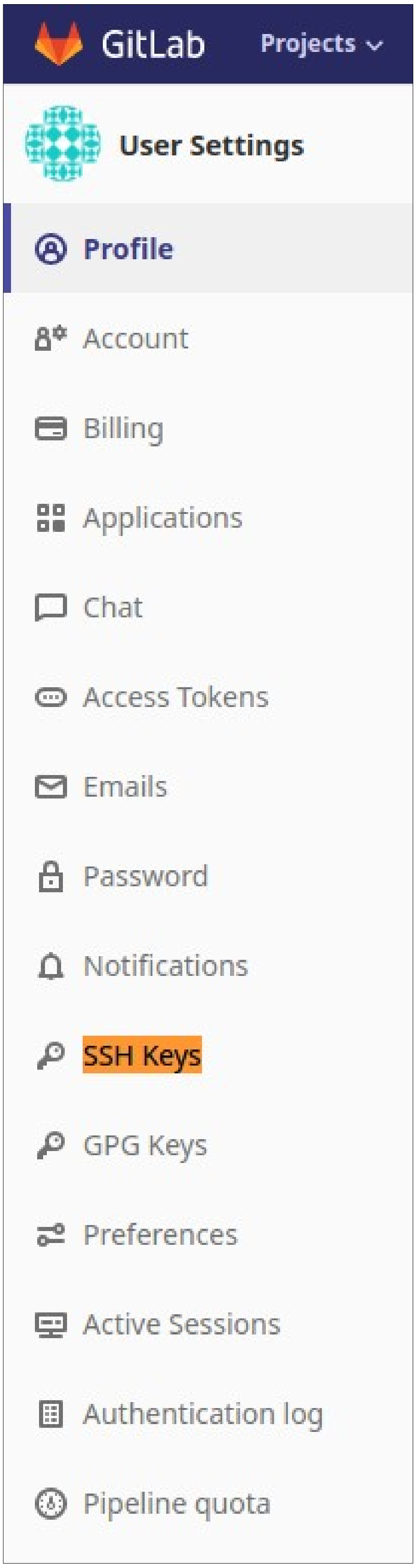


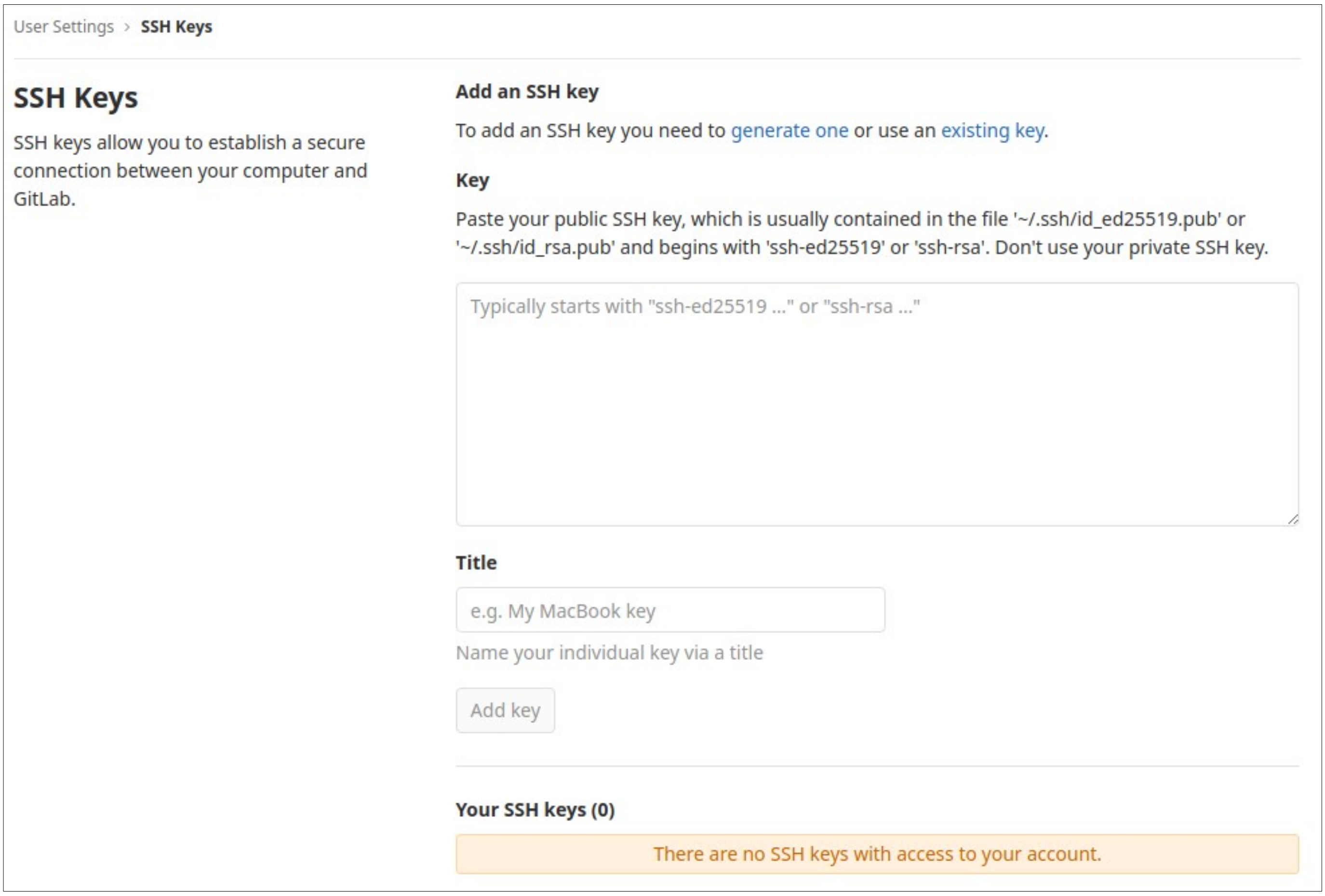
You won't be able to pull or push project code via SSH until you **add an SSH key** to your profile

SSH keys are necessary to stablish a secure encrypted connection between your computer and Gitlab.

Click on ***add an SSH key*** in the warning or click on ***SSH keys*** in ***Settings > User Settings***




GitLab
Projects
User Settings
Profile
Account
Billing
Applications
Chat
Access Tokens
Emails
Password
Notifications
SSH Keys
GPG Keys
Preferences
Active Sessions
Authentication log
Pipeline quota



User Settings > SSH Keys

SSH Keys

SSH keys allow you to establish a secure connection between your computer and GitLab.

Add an SSH key

To add an SSH key you need to generate one or use an existing key.

Key

Paste your public SSH key, which is usually contained in the file '~/.ssh/id_ed25519.pub' or '~/.ssh/id_rsa.pub' and begins with 'ssh-ed25519' or 'ssh-rsa'. Don't use your private SSH key.

Typically starts with "ssh-ed25519 …" or "ssh-rsa …"

Title

e.g. My MacBook key

Name your individual key via a title

Add key

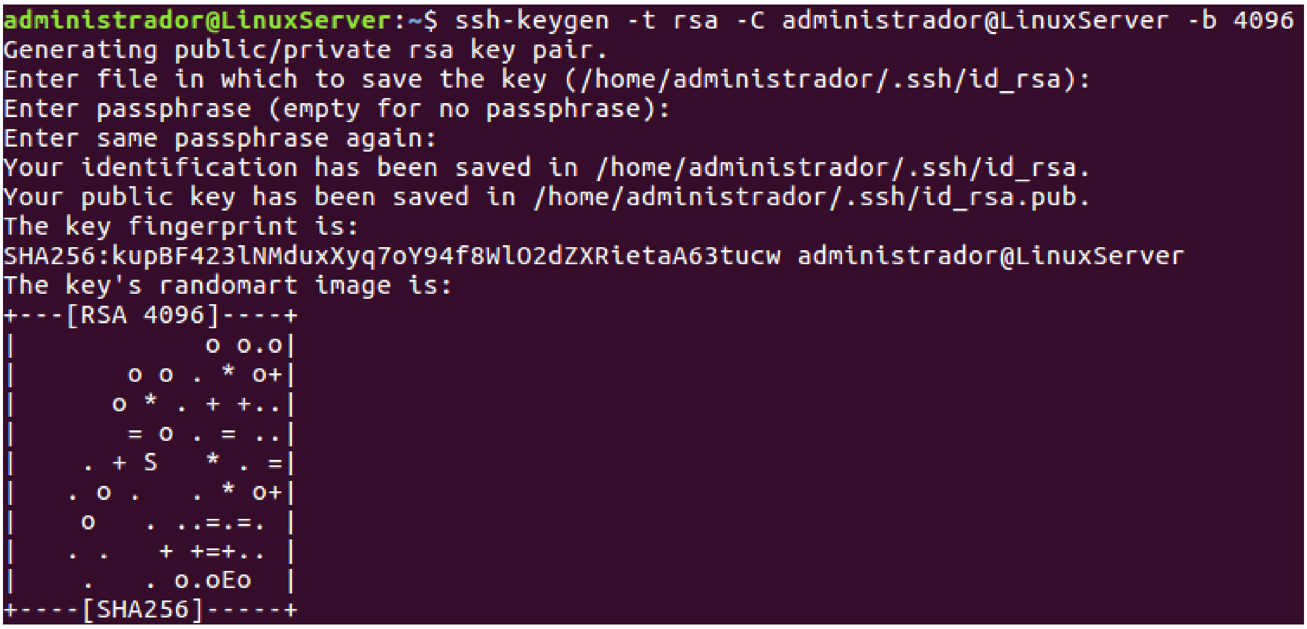
Your SSH keys (0)

There are no SSH keys with access to your account.

Here is where we are going to paste our public SSH key. Remember that we have to use the public key not the private.

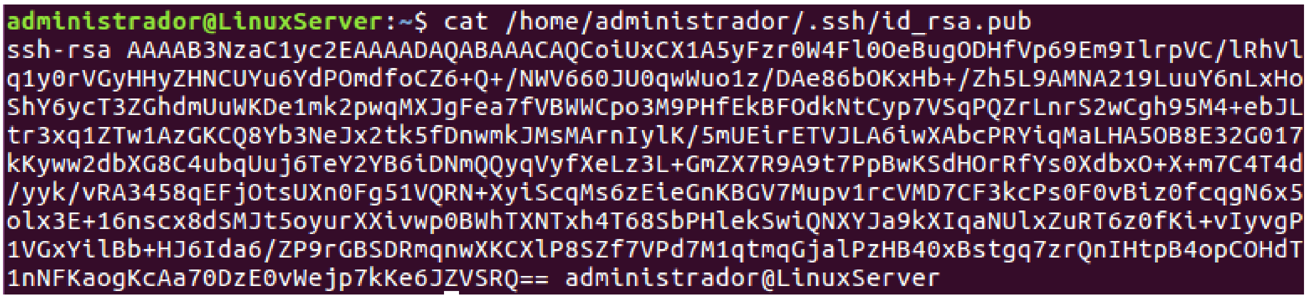
Now we have to generate an SSH key. For that, we click on ***generate one*** and follow the instructions.

First, we write the command *ssh-keygen -t rsa -C "your.email@example.com" -b 4096* in the terminal where you are working: the virtual machine (Linuxserver or other) or your physical one. (In this case we will use the *username@hostname* to generate the SSH key because is the machine where we will connect to GitLab. In my case, *administrador@LinuxServer*)



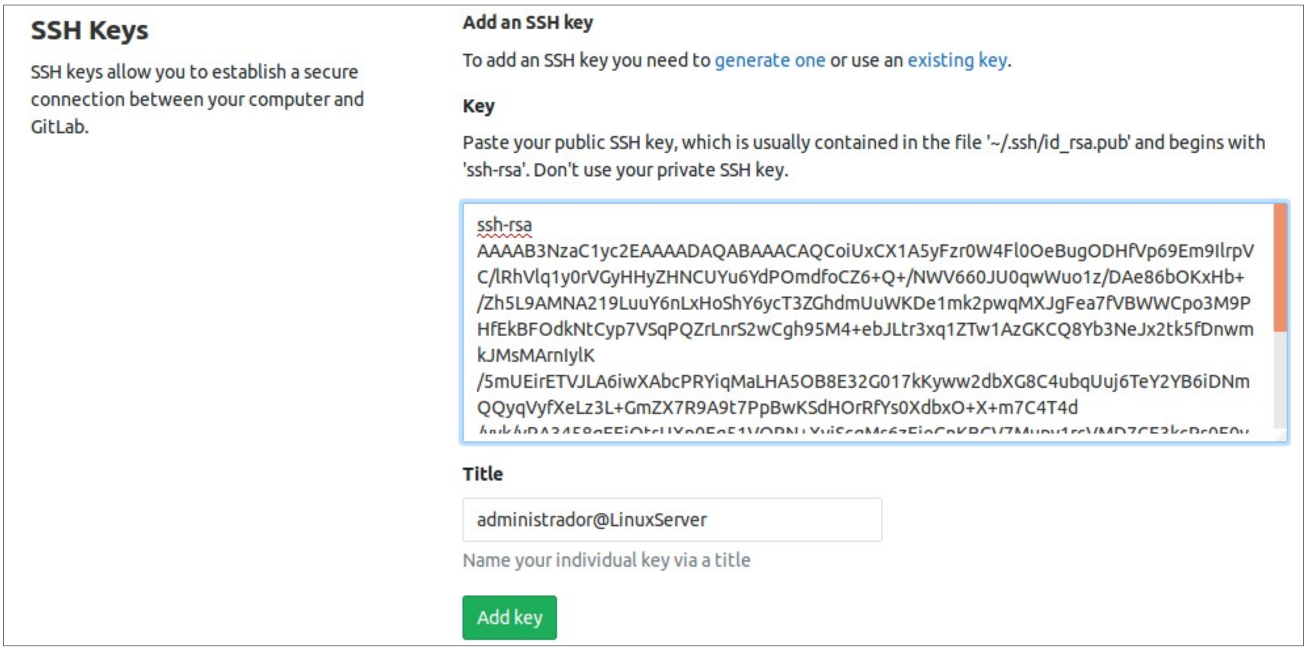
```
administrador@LinuxServer:~$ ssh-keygen -t rsa -C administrador@LinuxServer -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/administrador/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/administrador/.ssh/id_rsa.
Your public key has been saved in /home/administrador/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:kupBF423lNMduxXyq7oY94f8WlO2dZXRietaA63tucw administrador@LinuxServer
The key's randomart image is:
+---[RSA 4096]----+
|            o o.o|
|       o o . * o+|
|      o * . + +..|
|       = o . = ..|
|    . + S   * . =|
|   . o .   . * o+|
|    o   . ..=.=. |
|   . .   + +=+.. |
|    .   . o.oEo  |
+----[SHA256]-----+
```

We can see the content of the file with our public SSH key:



```
administrador@LinuxServer:~$ cat /home/administrador/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQCoiUxCX1A5yFzr0W4Fl0OeBugODHfVp69Em9IlrpVC/lRhVl
q1y0rVGyHHyZHNCUYu6YdPOmdfoCZ6+Q+/NWV660JU0qwWuo1z/DAe86bOKxHb+/Zh5L9AMNA219LuuY6nLxHo
ShY6ycT3ZGhdmUuWKDe1mk2pwqMXJgFea7fVBWWCpo3M9PHfEkBFOdkNtCyp7VSqPQZrLnrS2wCgh95M4+ebJL
tr3xq1ZTw1AzGKCQ8Yb3NeJx2tk5fDnwmkJMsMArnIylK/5mUEirETVJLA6iwXAbcPRYiqMaLHA5OB8E32G017
kKyww2dbXG8C4ubqUuj6TeY2YB6iDNmQQyqVyfXeLz3L+GmZX7R9A9t7PpBwKSdHOrRfYs0XdbxO+X+m7C4T4d
/yyk/vRA3458qEFjOtsUXn0Fg51VQRN+XyiScqMs6zEieGnKBGV7Mupv1rcVMD7CF3kcPs0F0vBiz0fcqgN6x5
olx3E+16nscx8dSMJt5oyurXXivwp0BWhTXNTxh4T68SbPHlekSwiQNXYJa9kXIqaNUlxZuRT6z0fKi+vIyvgP
1VGxYilBb+HJ6Ida6/ZP9rGBSDRmqnwXKCXlP8SZf7VPd7M1qtmqGjalPzHB40xBstgq7zrQnIHtpB4opCOHdT
1nNFKaogKcAa70DzE0vWejp7kKe6JZVSRQ== administrador@LinuxServer
```

This is the key we have to use in GitLab, so we copy it and paste in the GitLab page, write a title and click on ***Add key*** button



SSH Keys

SSH keys allow you to establish a secure connection between your computer and GitLab.

Add an SSH key

To add an SSH key you need to generate one or use an existing key.

Key

Paste your public SSH key, which is usually contained in the file '~/.ssh/id_rsa.pub' and begins with 'ssh-rsa'. Don't use your private SSH key.

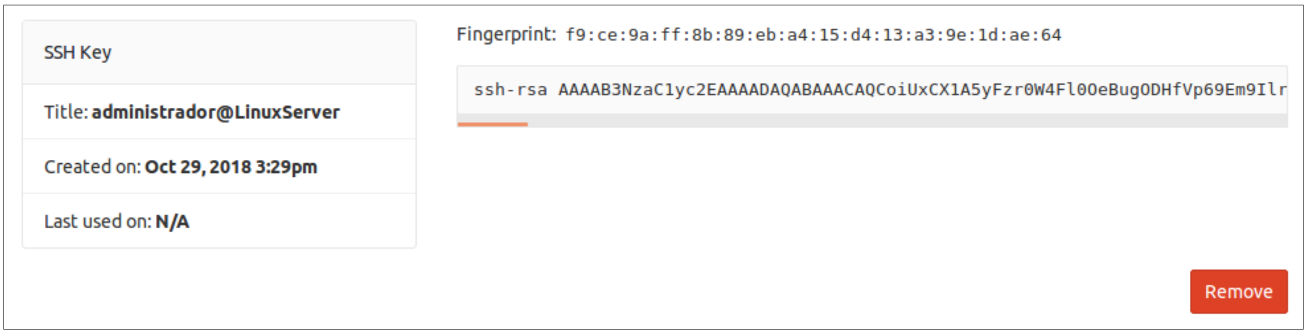
```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAACAQCoiUxCX1A5yFzr0W4Fl0OeBugODHfVp69Em9IlrpV
C/lRhVlq1y0rVGyHHyZHNCUYu6YdPOmdfoCZ6+Q+/NWV660JU0qwWuo1z/DAe86bOKxHb+
/Zh5L9AMNA219LuuY6nLxHoShY6ycT3ZGhdmUuWKDe1mk2pwqMXJgFea7fVBWWCpo3M9P
HfEkBFOdkNtCyp7VSqPQZrLnrS2wCgh95M4+ebJLtr3xq1ZTw1AzGKCQ8Yb3NeJx2tk5fDnwm
kJMsMArnIylK
/5mUEirETVJLA6iwXAbcPRYiqMaLHA5OB8E32G017kKyww2dbXG8C4ubqUuj6TeY2YB6iDNm
QQyqVyfXeLz3L+GmZX7R9A9t7PpBwKSdHOrRfYs0XdbxO+X+m7C4T4d
```

Title

administrador@LinuxServer

Name your individual key via a title

And we have our public SSH key added in Gitlab:



SSH Key

Title: **administrador@LinuxServer**

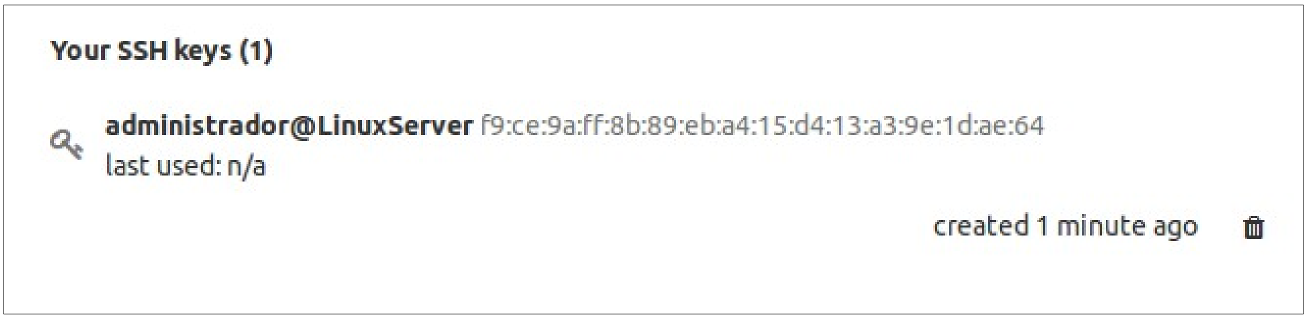
Created on: **Oct 29, 2018 3:29pm**

Last used on: **N/A**

Fingerprint: `f9:ce:9a:ff:8b:89:eb:a4:15:d4:13:a3:9e:1d:ae:64`

```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQCoiUxCX1A5yFzr0W4Fl0OeBugODHfVp69Em9Ilr
```

If we click on ***SSH keys*** again we could see our SSH Keys in the bottom of the page. We can add all keys we want:



Your SSH keys (1)

administrador@LinuxServer f9:ce:9a:ff:8b:89:eb:a4:15:d4:13:a3:9e:1d:ae:64
last used: n/a

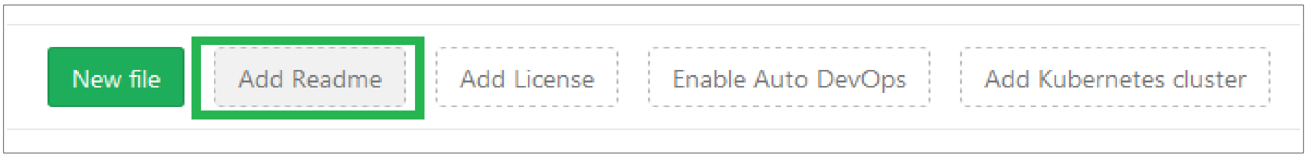
created 1 minute ago

Now that the administrador@LinuxServer SSH key has been added **we will be able to connect to the Gitlab repository using a Linux terminal** from administrador@LinuxServer (or whatever user@host you used). We will do that in **section 2.10** of this activity.

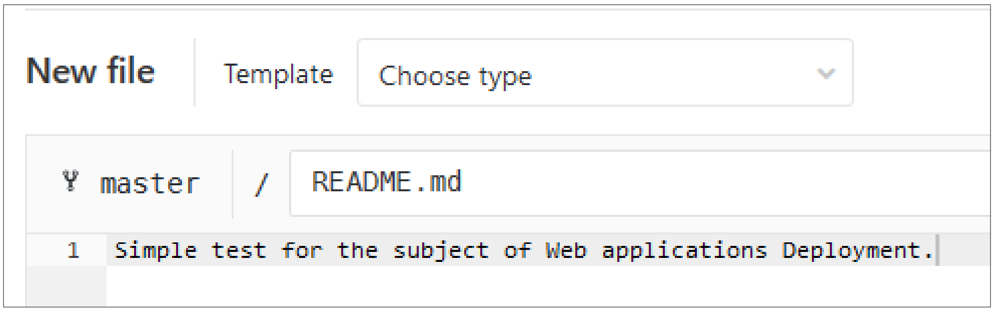
2.4 Add the Readme file

A Readme file contains information about the project. Its extension is *md*.

First of all, we are going to create a Readme file. For that, we have to click on the ***Add Readme*** button:



Now, we write the content of the file. We can see that we are in the *master* branch (created by default).

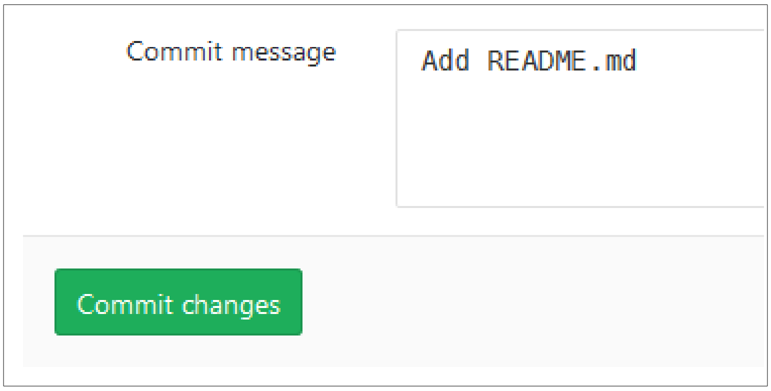


Choose type

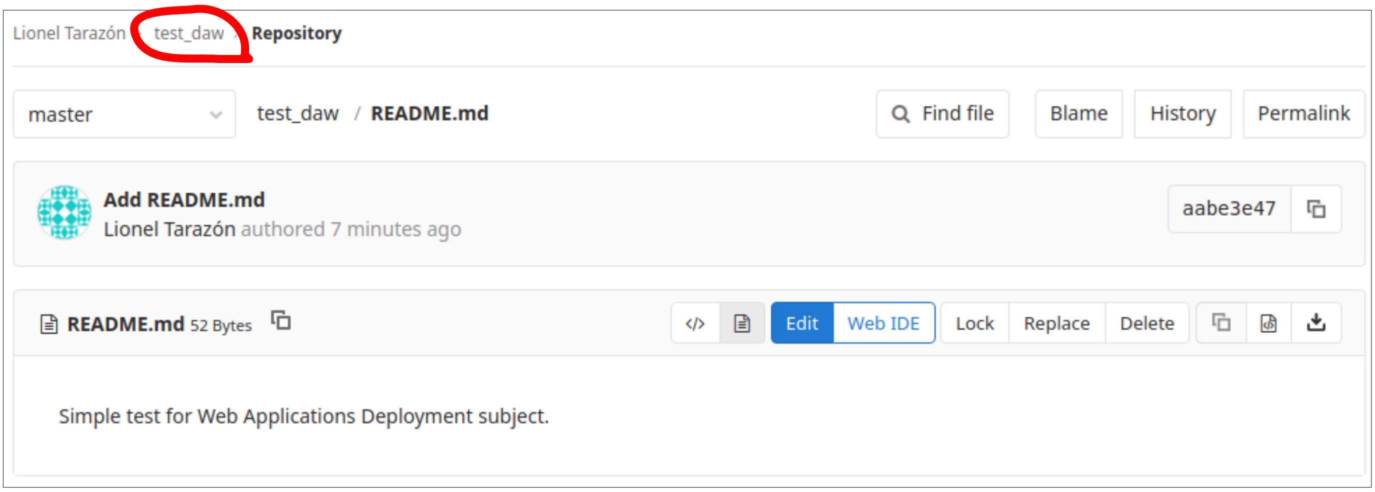
master / README.md

```
Simple test for the subject of Web applications Deployment.
```

Once the file is written we add a message to do the commit and click on the ***Commit changes*** button:


Commit message: Add README.md

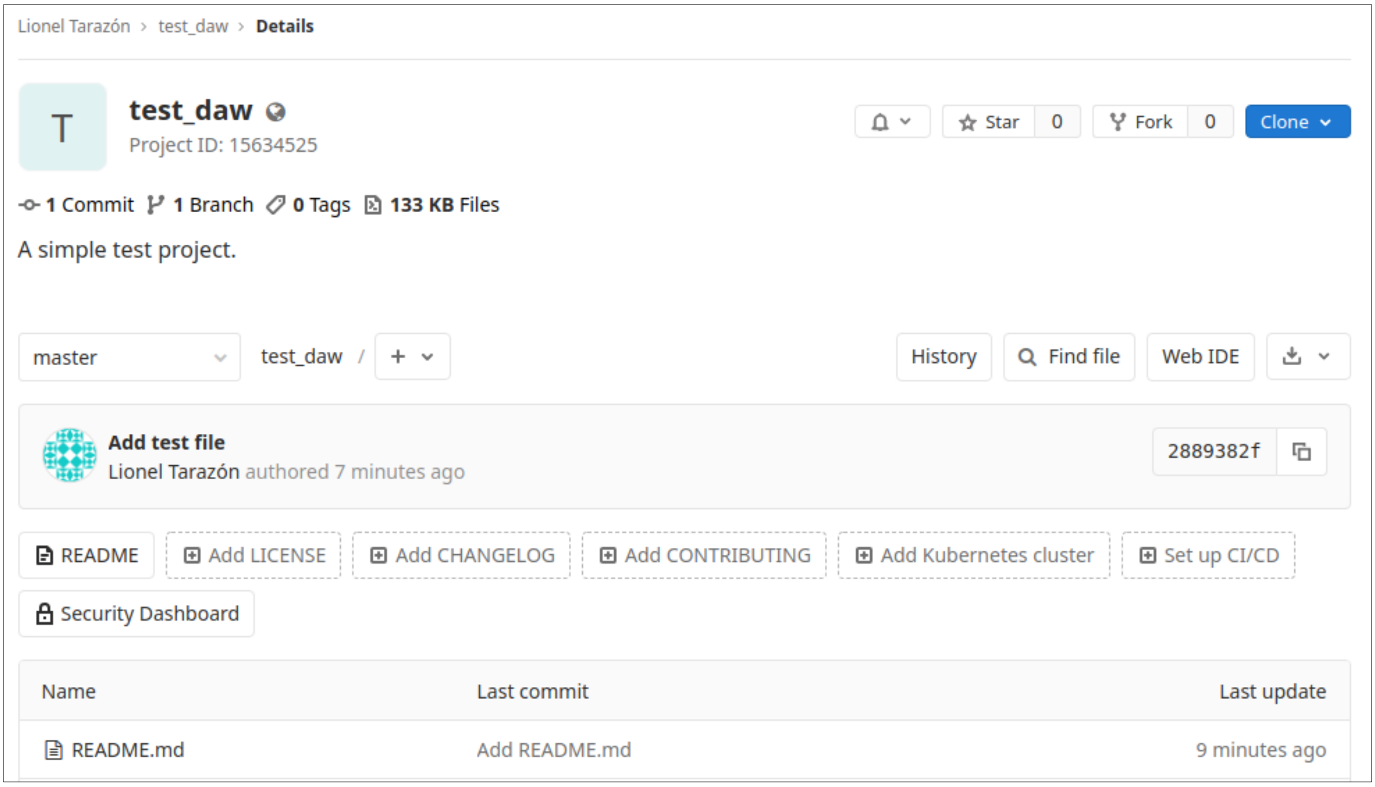
And we can see the file into our repository and if we click on *test_daw* we can see its content:



Add README.md
Lionel Tarazón authored 7 minutes ago
aabe3e47

README.md 52 Bytes

Simple test for Web Applications Deployment subject.



test_daw
Project ID: 15634525

1 Commit 1 Branch 0 Tags 133 KB Files

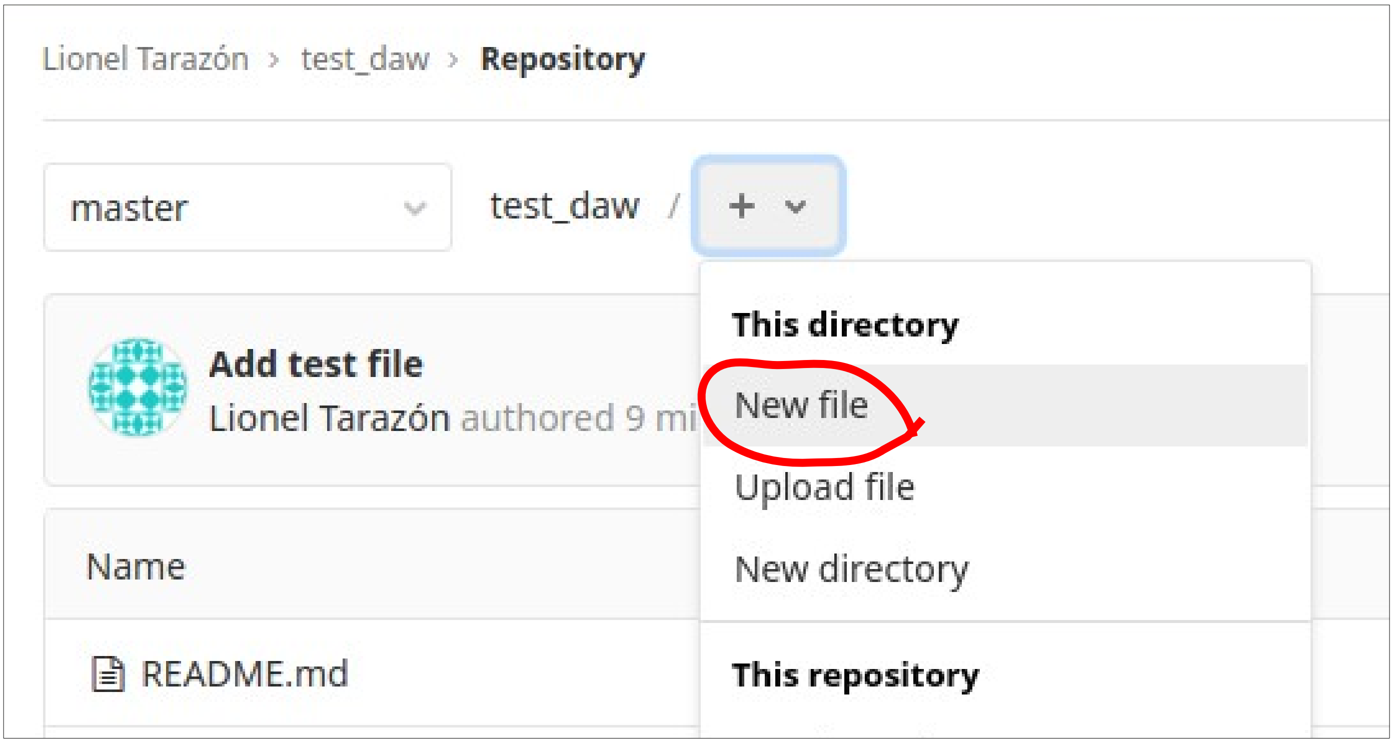
A simple test project.

Add test file
Lionel Tarazón authored 7 minutes ago
2889382f

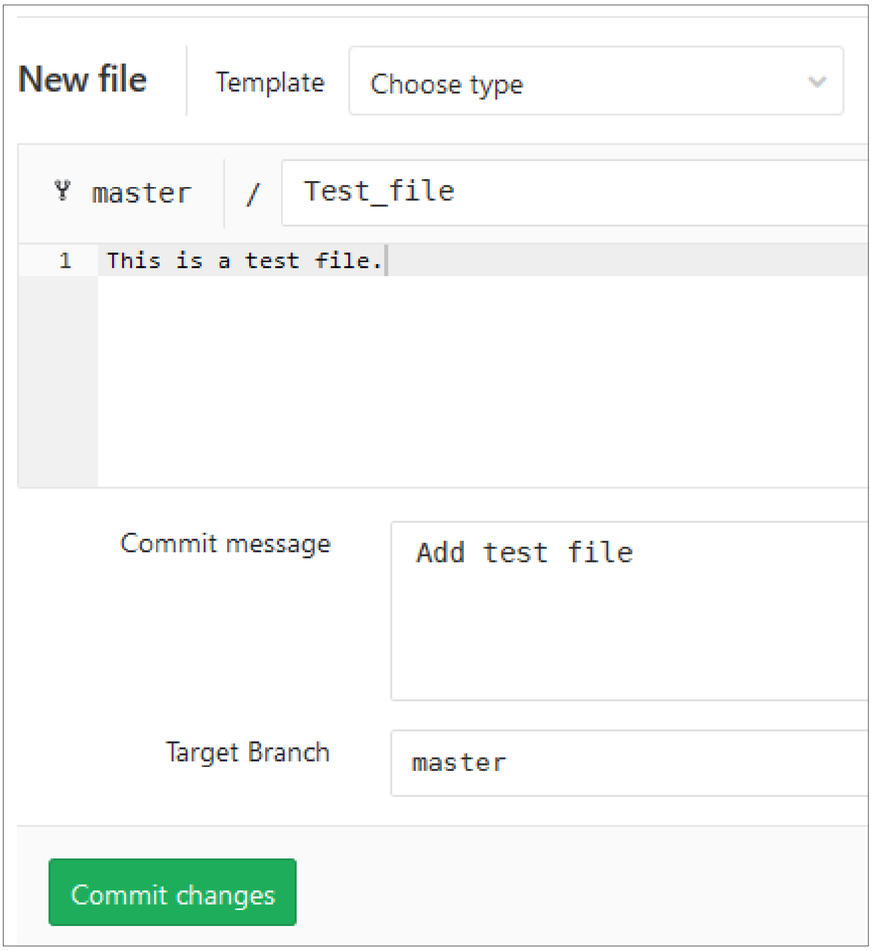
Name	Last commit	Last update
README.md	Add README.md	9 minutes ago

2.5 Add files

Now we are going to add a file to our repository. We can create a new file or upload it. For that, we have to click on the + icon:



If we click on ***New file*** we could create and edit an empty file. We write its name, its content, the commit message and the target branch (*master*, the only we have) and click on ***Commit changes*** button:



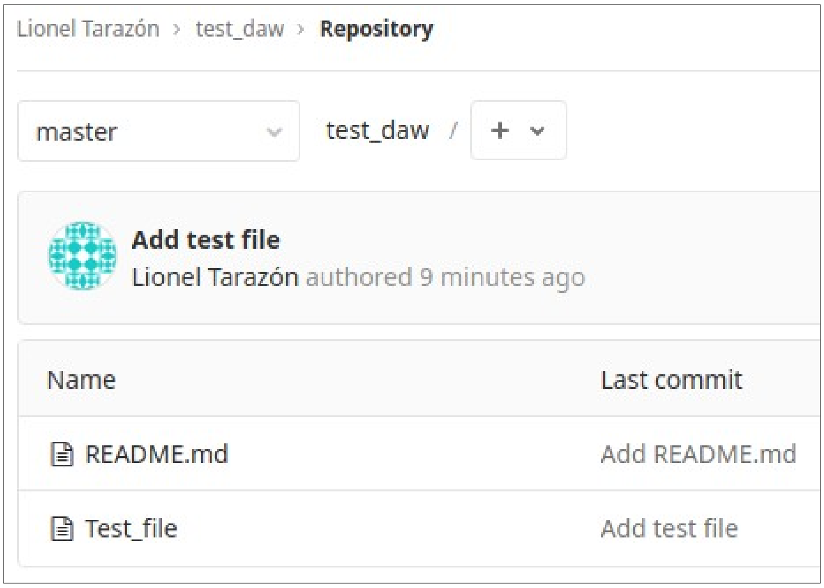
master / Test_file

```
This is a test file.
```

Commit message: Add test file

Target Branch: master

And we will have a new file in our repository:

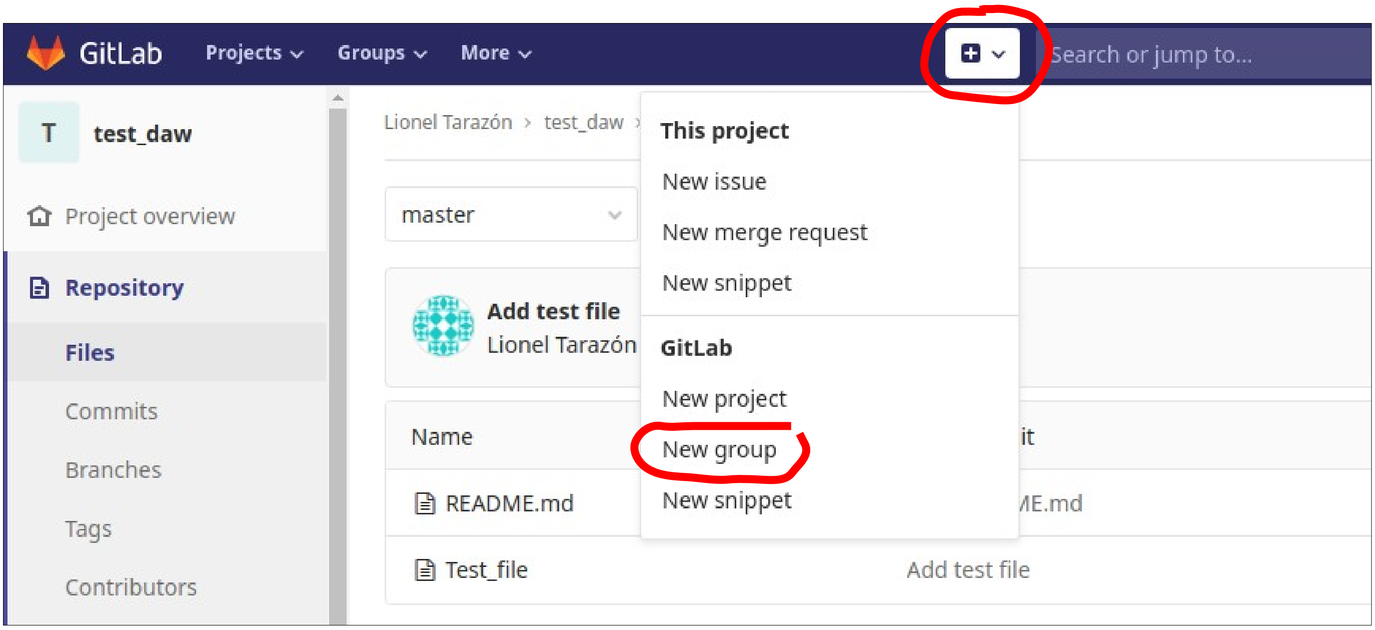

Lionel Tarazón › test_daw › Repository
master
test_daw /
+
Add test file
Lionel Tarazón authored 9 minutes ago

Name	Last commit
README.md	Add README.md
Test_file	Add test file

Also we can upload files. To do so, we have to click on **+** icon and then ***Upload file***, and choose the file we want to upload.

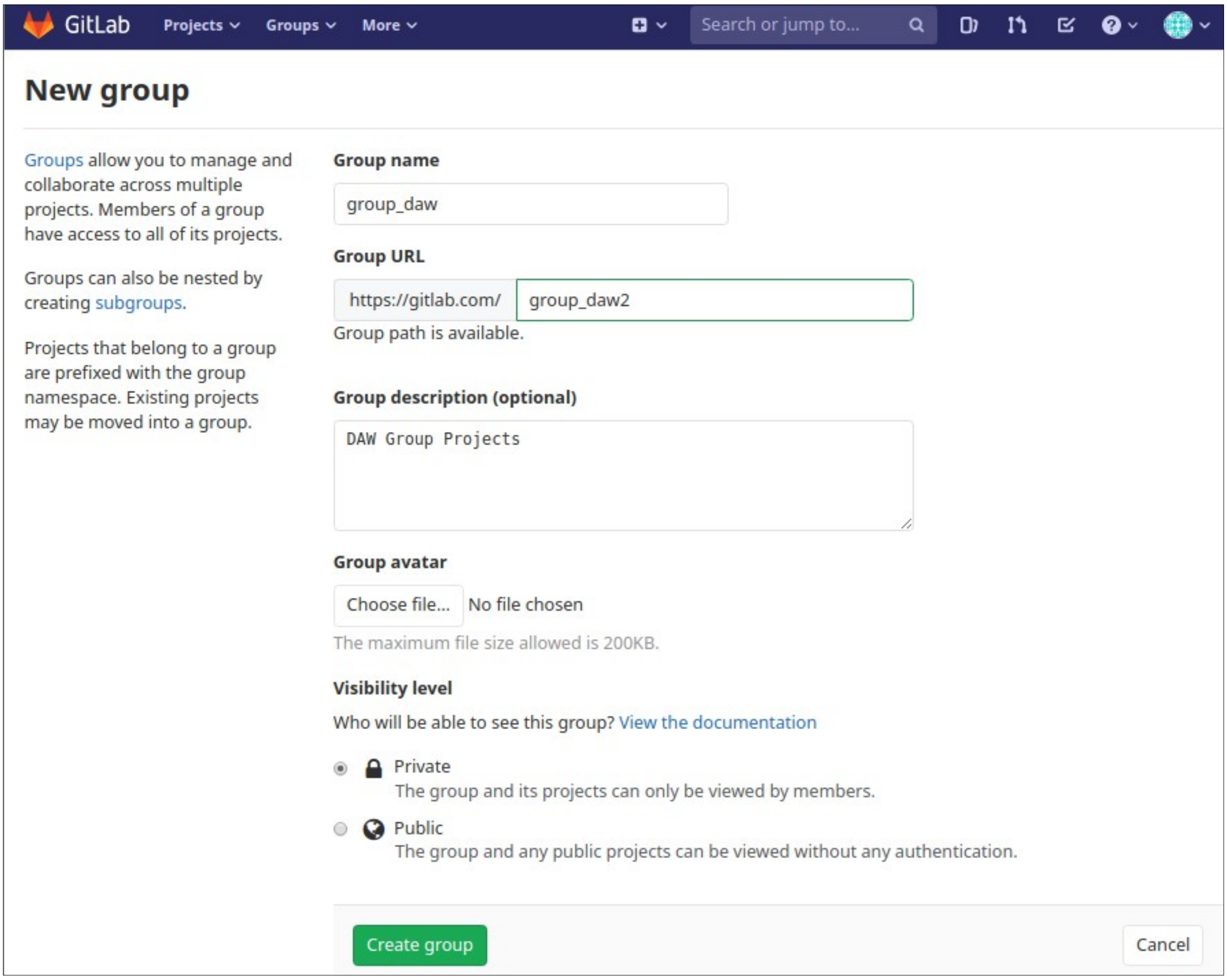
2.6 Create groups

A group is a collection of projects. To create a new group we have to click on **+** icon and then ***New group***:



Name	
README.md	...ME.md
Test_file	Add test file

Now we have to give a name to the group, a description, choose the visibility and click on the ***Create group*** button. We are going to create a group for our projects:



New group

Groups allow you to manage and collaborate across multiple projects. Members of a group have access to all of its projects.

Groups can also be nested by creating subgroups.

Projects that belong to a group are prefixed with the group namespace. Existing projects may be moved into a group.

Group name: group_daw

Group URL: https://gitlab.com/ group_daw2

Group path is available.

Group description (optional): DAW Group Projects

Group avatar: Choose file... No file chosen

The maximum file size allowed is 200KB.

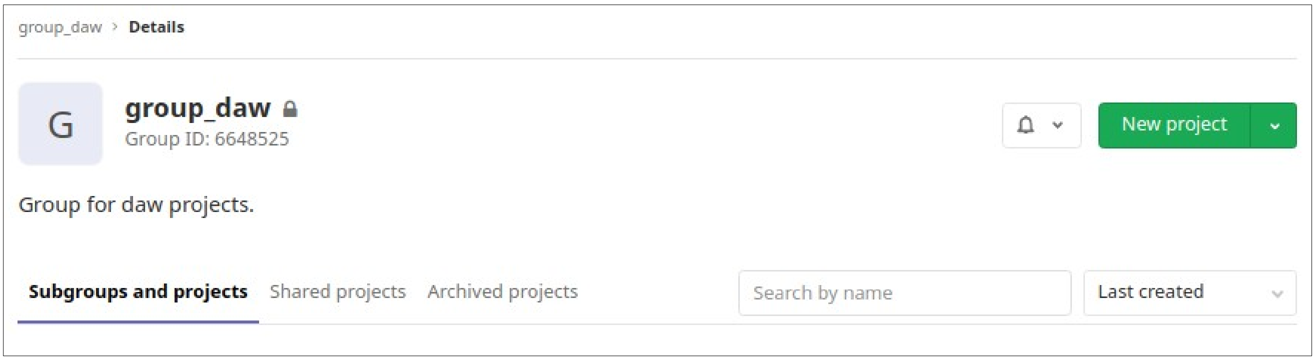
Visibility level

Who will be able to see this group? View the documentation

● Private
The group and its projects can only be viewed by members.

○ Public
The group and any public projects can be viewed without any authentication.

And we have our group:

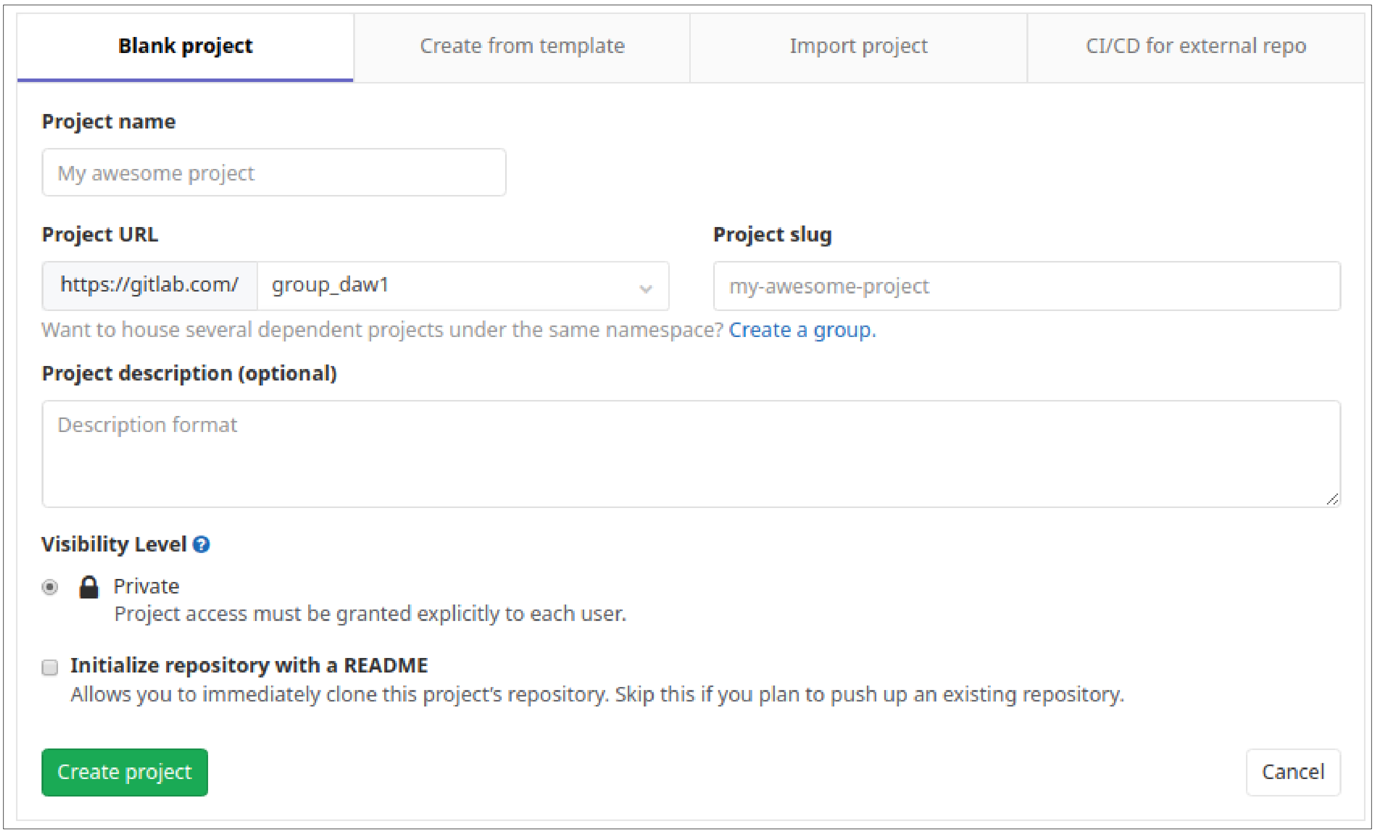


group_daw

Group ID: 6648525

Group for daw projects.

If we click on the ***New project*** button we can create new project inside this group:



Project name

My awesome project

Project URL

https://gitlab.com/ group_daw1

Project slug

my-awesome-project

Want to house several dependent projects under the same namespace? Create a group.

Project description (optional)

Description format

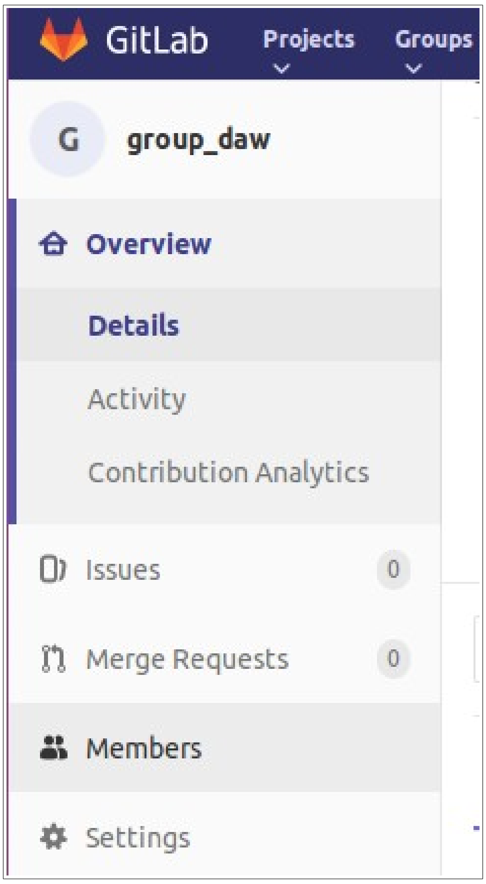
Visibility Level

● Private
Project access must be granted explicitly to each user.

□ Initialize repository with a README
Allows you to immediately clone this project's repository. Skip this if you plan to push up an existing repository.

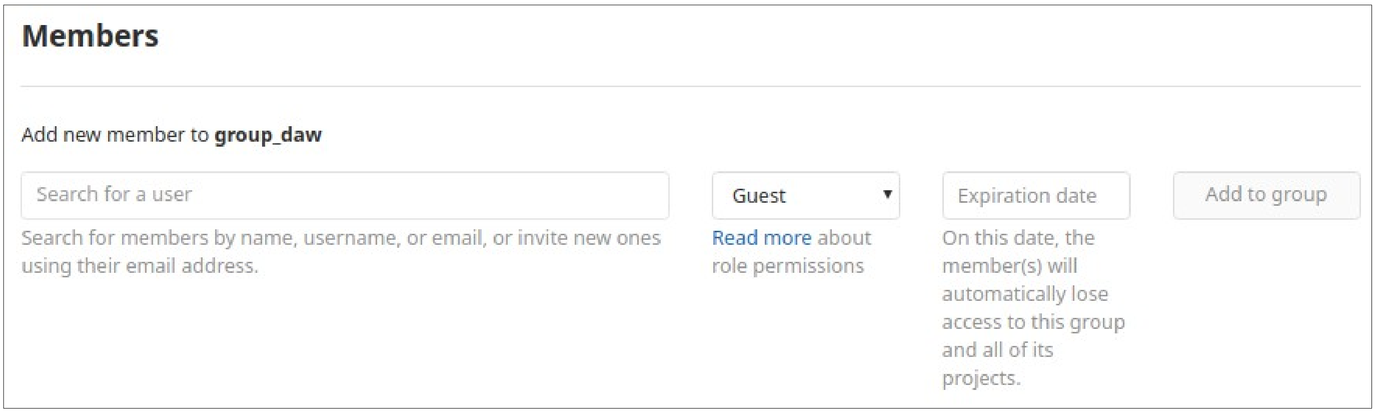
It's often useful to add members to a group so that different people can add projects and collaborate.

To add members you first need to go to the Group main page and click on ***Members*** (on the left side menu).



You can search members by their name, username or email. Then, set their role permission (in https://gitlab.com/help/user/permissions.html you could see a table with the user permission levels), the expiration date and click on ***Add to group*** button.

If you wish you can add other classmates to your group. You can also add me using my e-mail lionelmanuel.tarazon@ceedcv.es and I will tell you if it worked.



Members

Add new member to **group_daw**

Search for a user

Search for members by name, username, or email, or invite new ones using their email address.

Guest ▾

Read more about role permissions

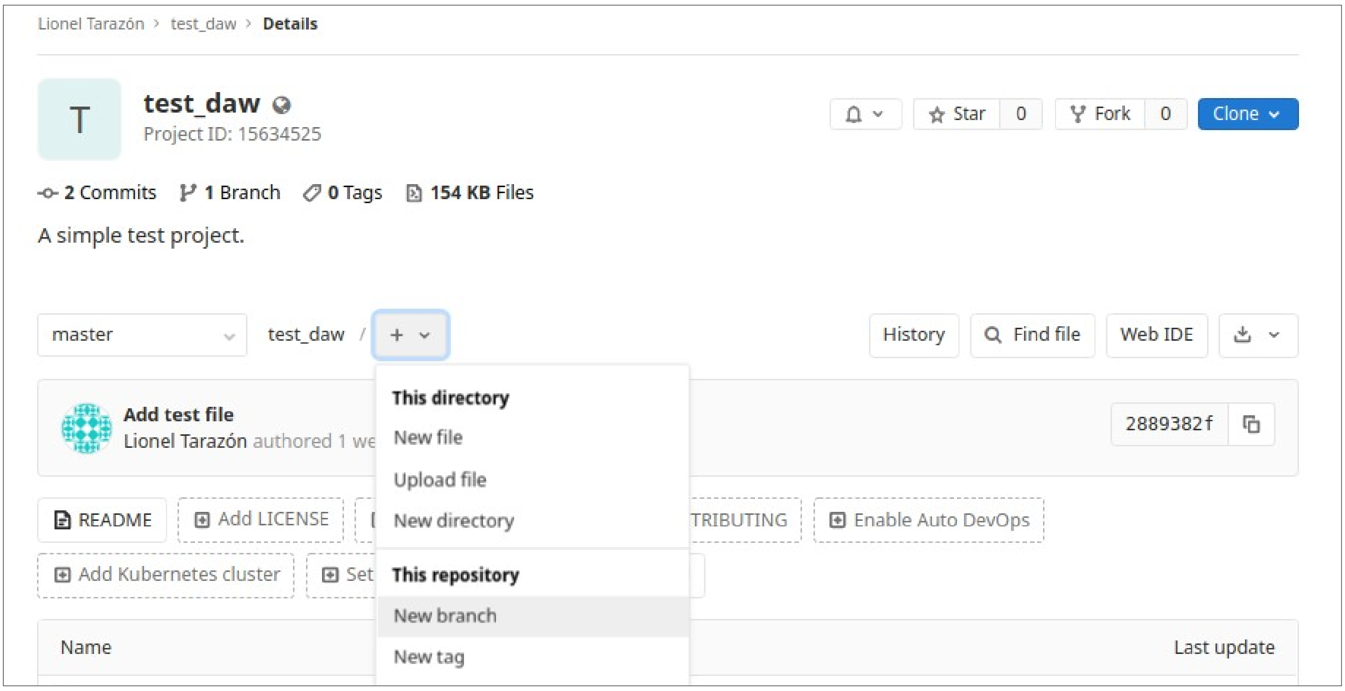
Expiration date

On this date, the member(s) will automatically lose access to this group and all of its projects.

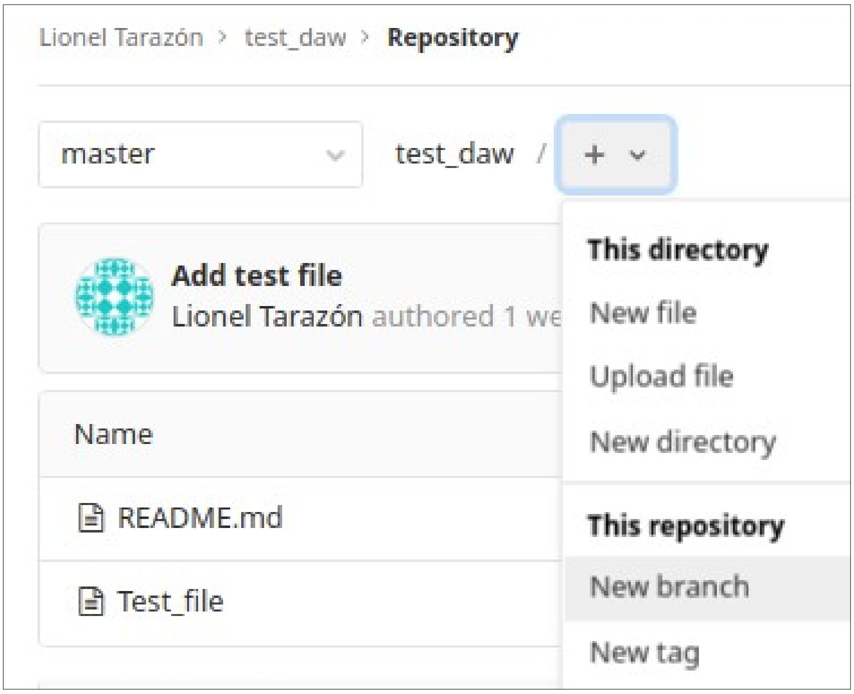
Add to group

2.7 Create a new branch

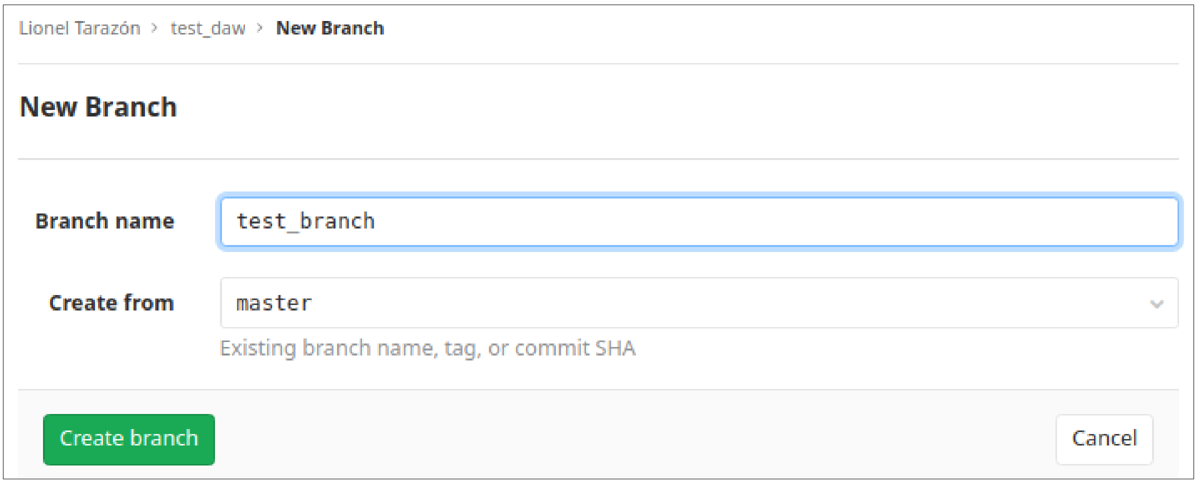
Now we are going to create a new branch in our project *test_daw*. To do so we have to click on the **+** icon and then click on ***New branch***:



Also you can do it using the **+** icon in the repository view:



Now we have to write the branch name and select from what branch it will be created (in this case we only have one, the *master* branch) and click on ***Create branch***:



Lionel Tarazón > test_daw > New Branch

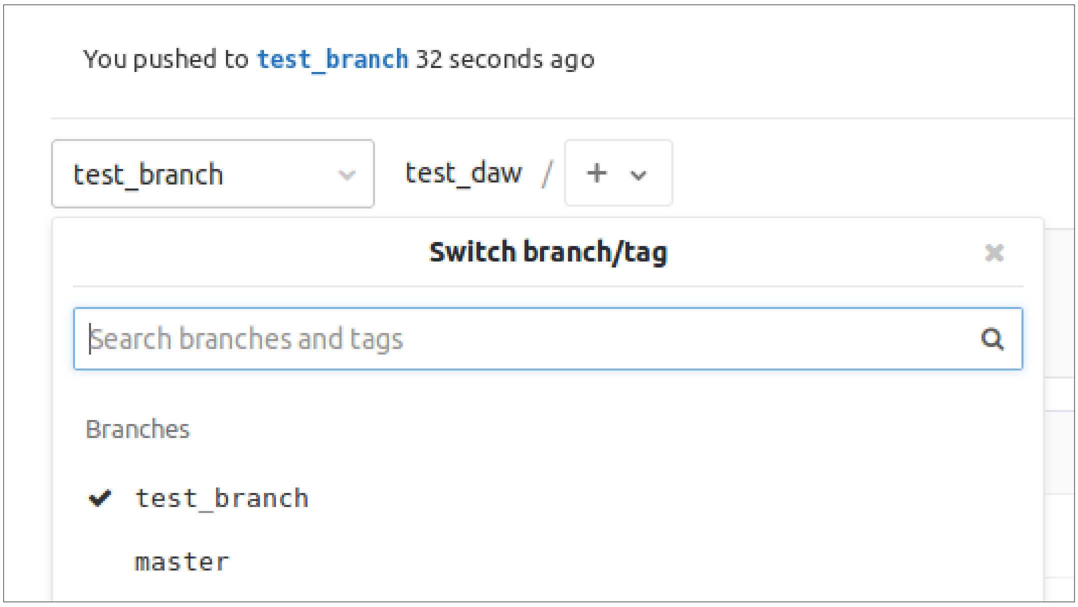
New Branch

Branch name: test_branch

Create from: master

Existing branch name, tag, or commit SHA

Now we can switch the branch clicking in its name:



You pushed to test_branch 32 seconds ago

test_branch test_daw / +

Switch branch/tag

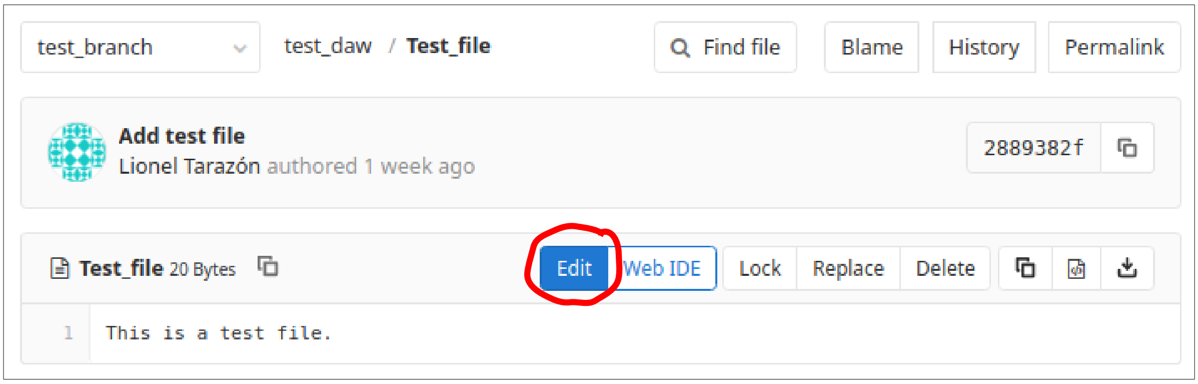
Search branches and tags

Branches

✓ test_branch

master

Now we are going to modify the *Test_file* file in the *test_branch* branch and do a commit, then we will merge the changes to the *master* branch. To do so first we choose the *test_branch*, click on the *Test_file* file and click on ***Edit*** (we will edit it using the GitLab Web Editor):

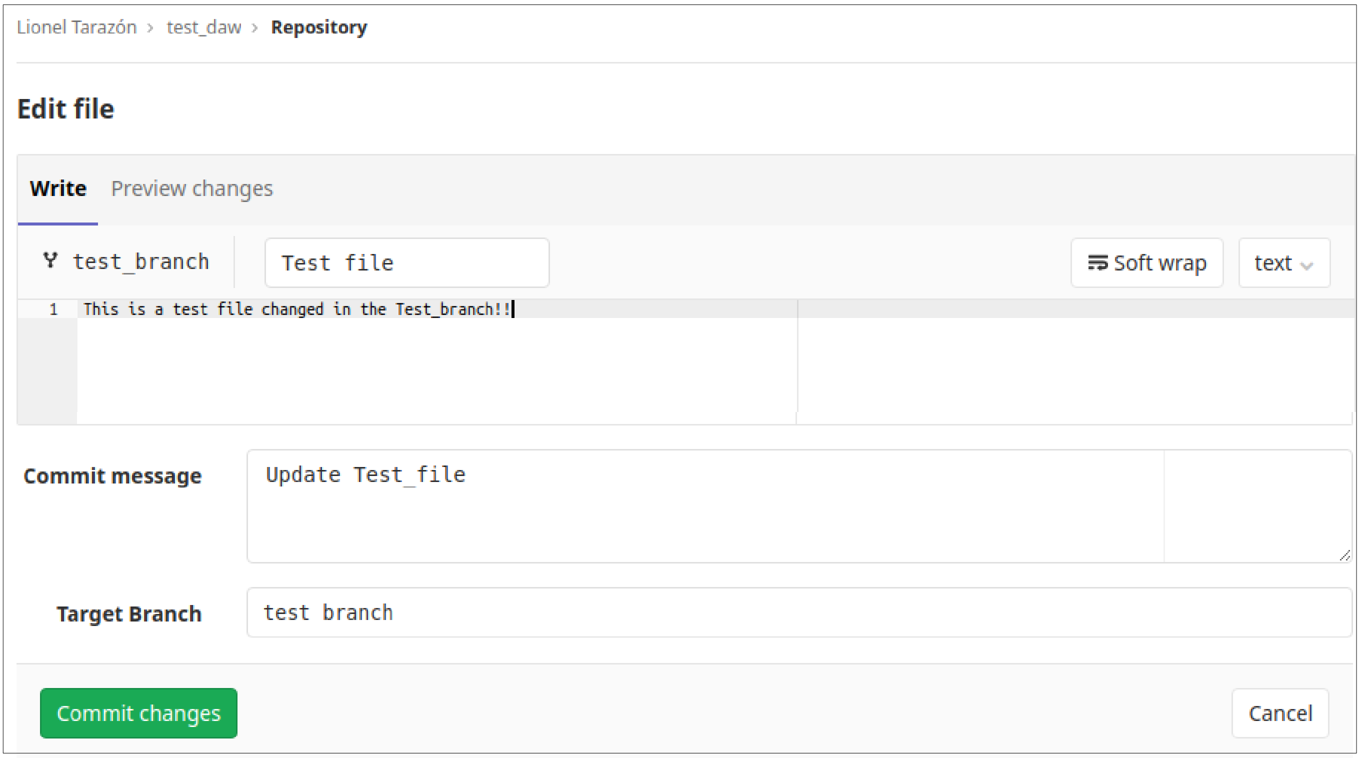


Add test file
Lionel Tarazón authored 1 week ago
2889382f

Test_file 20 Bytes

```
This is a test file.
```

We change the content, write a message for the commit and click on the ***Commit changes*** button (make sure the target branch is *test_branch*):



Edit file

test_branch — Test file

```
This is a test file changed in the Test_branch!!
```

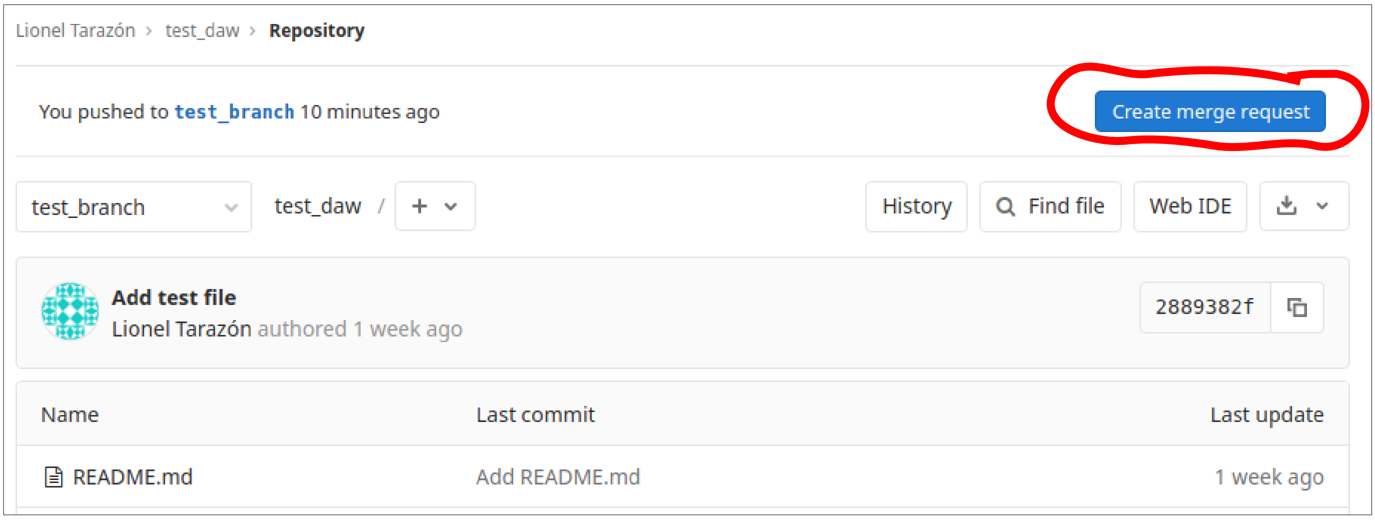
Commit message: Update Test_file

Target Branch: test branch

2.8 Create a merge request

Having different branches is useful to keep different working versions of your software and to colaborate with others in the same project. After making changes to a branch it can be useful to merge those change into the master branch. Lets do it.

Click on the ***Create merge request*** button.



Lionel Tarazón › test_daw › **Repository**

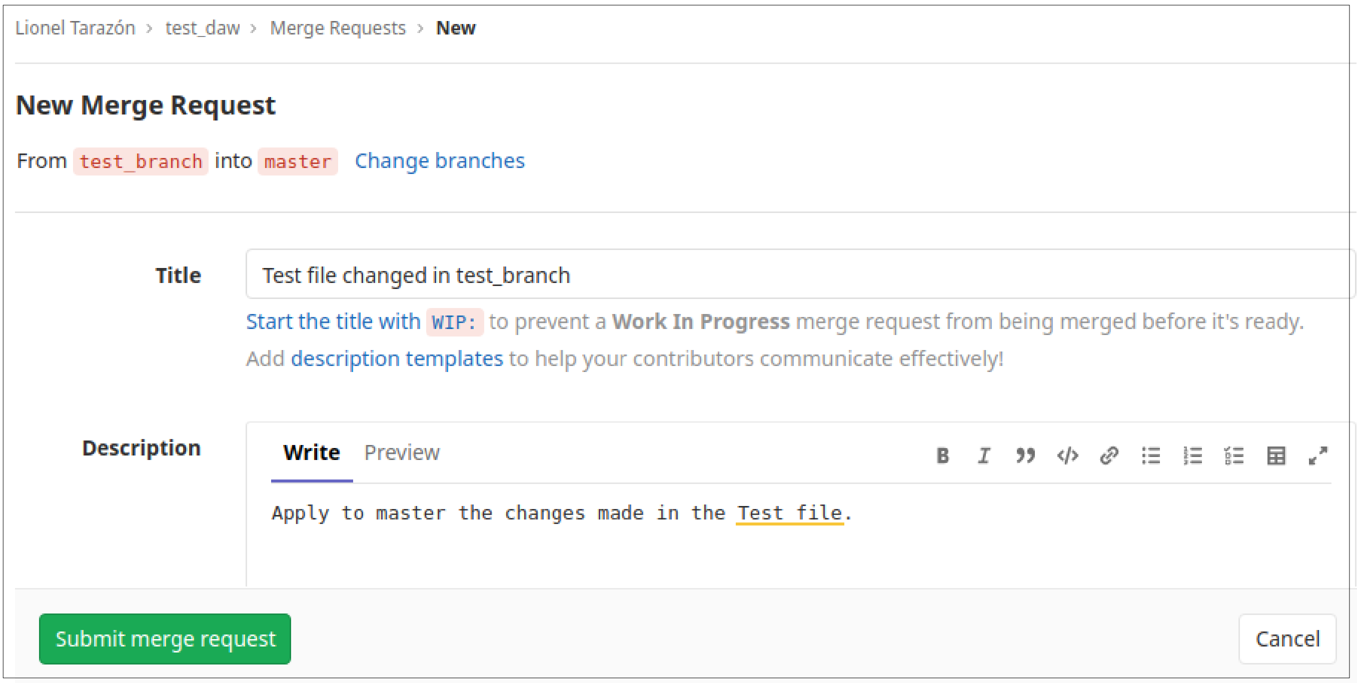
You pushed to `test_branch` 10 minutes ago

Add test file
Lionel Tarazón authored 1 week ago

2889382f

Name	Last commit	Last update
README.md	Add README.md	1 week ago

Now we write a title, a description and do the request merge from *test_branch* to *master* clicking on the ***Submit merge request*** button:



Lionel Tarazón › test_daw › Merge Requests › **New**

New Merge Request

From `test_branch` into `master` Change branches

Title Test file changed in test_branch

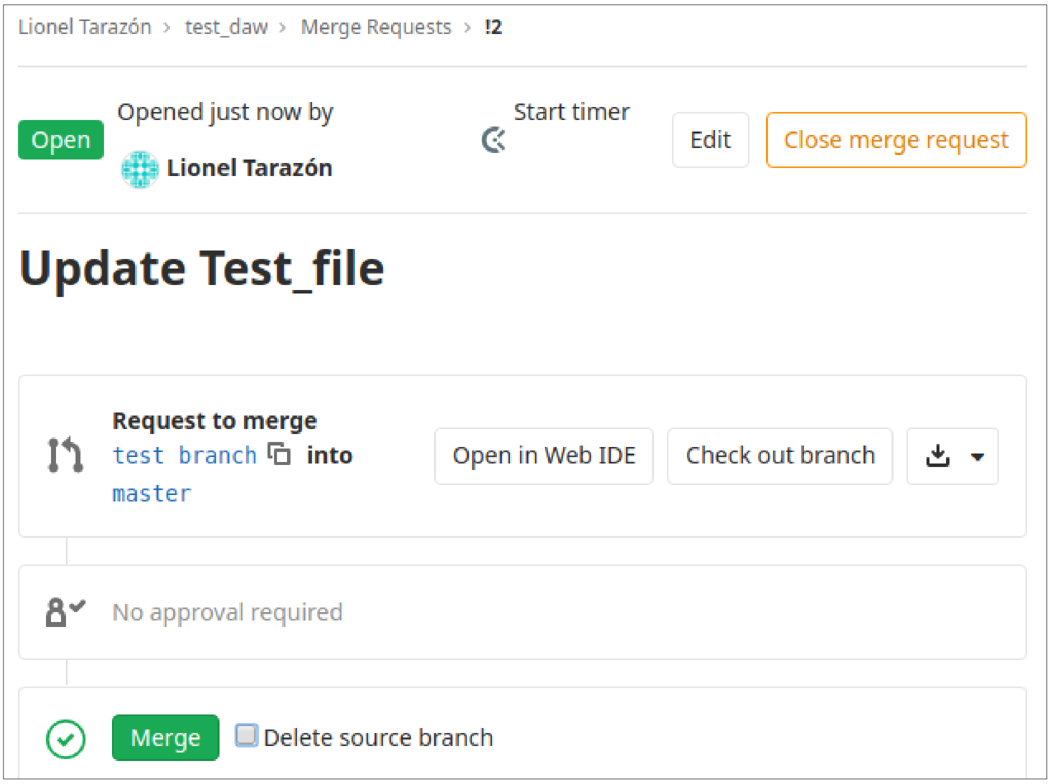
Start the title with `WIP:` to prevent a **Work In Progress** merge request from being merged before it's ready.
Add description templates to help your contributors communicate effectively!

Description

```
Apply to master the changes made in the Test file.
```

Finally, we can do the merge clicking on the ***Merge*** button.

Important: Careful with the “delete source branch” tick box. If you select it, the test_branch you are merging will be deleted after the merge.


Lionel Tarazón > test_daw > Merge Requests > !2

Open Opened just now by Lionel Tarazón

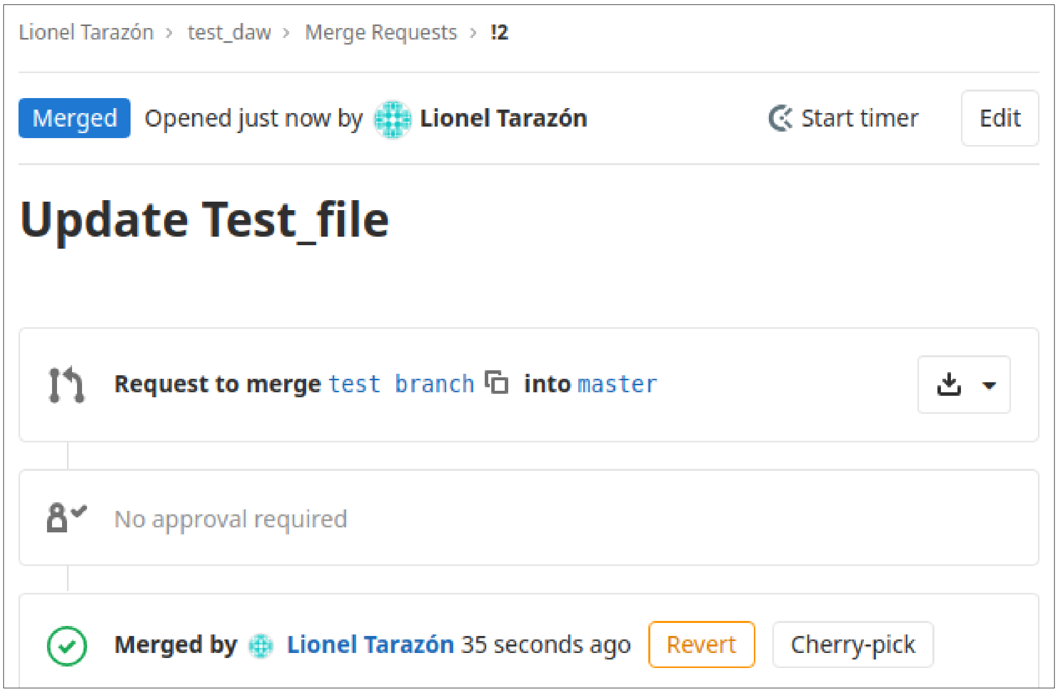
Update Test_file

Request to merge test branch **into** master

No approval required

□ Delete source branch

And the merge is done:


Lionel Tarazón > test_daw > Merge Requests > !2

Merged Opened just now by Lionel Tarazón

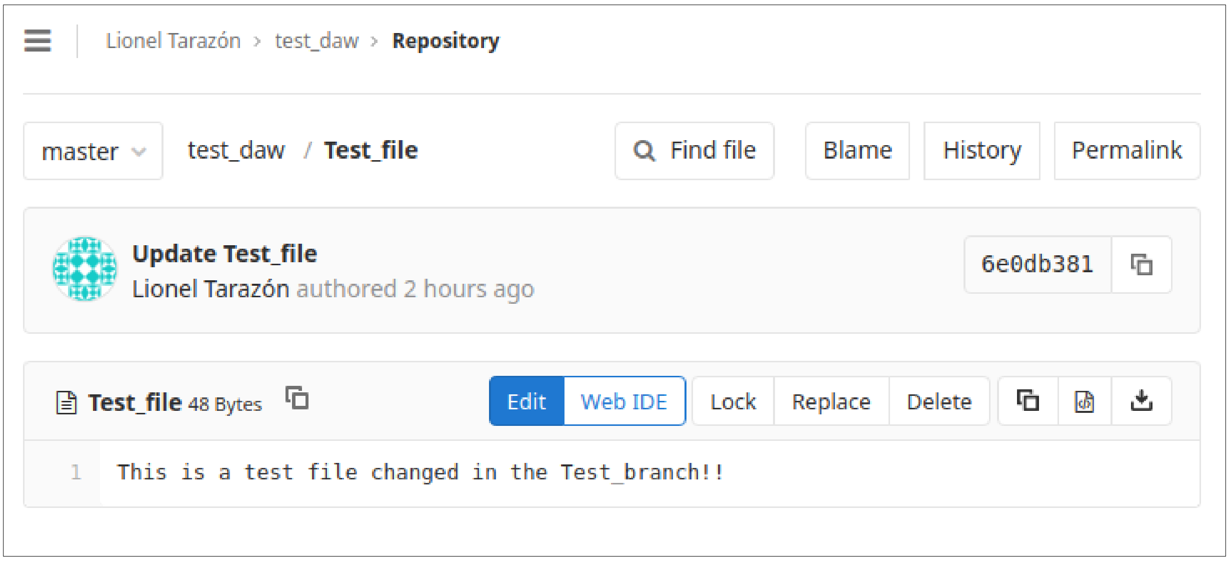
Update Test_file

Request to merge test branch **into** master

No approval required

Merged by Lionel Tarazón 35 seconds ago

Now if we take a look at the *Test_file* file from *master* we can see the changes:



Update Test_file
Lionel Tarazón authored 2 hours ago
6e0db381

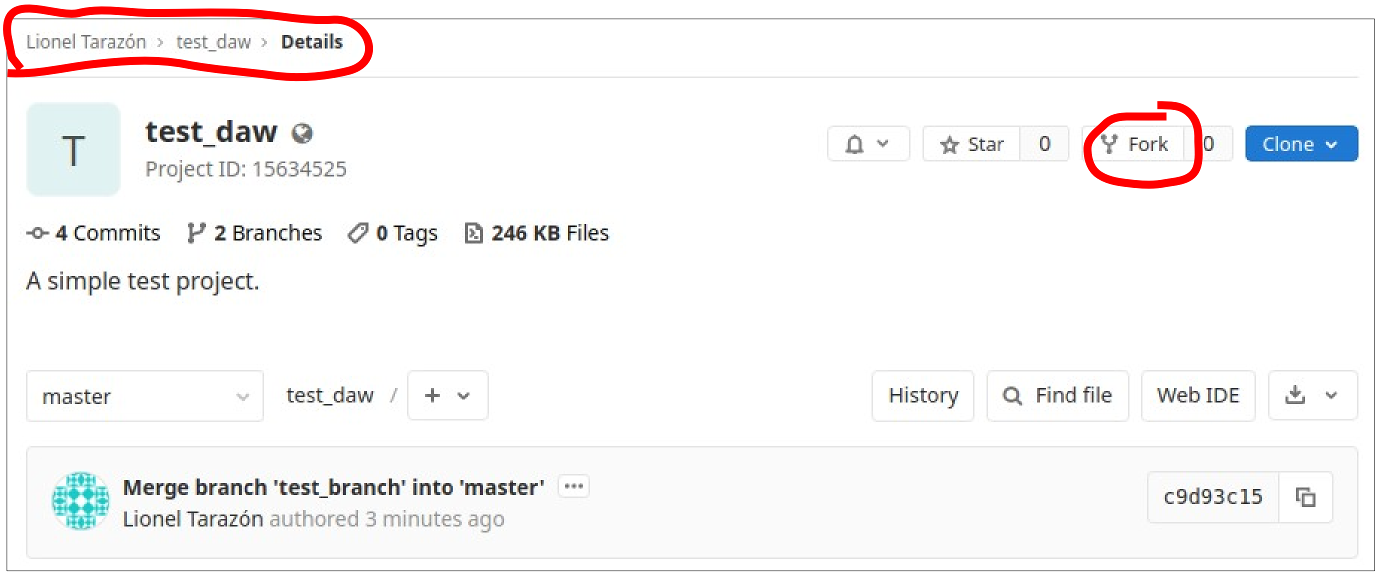
Test_file 48 Bytes

```
This is a test file changed in the Test_branch!!
```

2.9 Fork a project

A fork is a copy of a project. Forking a repository allows you to make changes without affecting the original project.

Now we are going to fork our project clicking on the ***Fork*** button in our project dashboard:



Lionel Tarazón › test_daw › **Details**

test_daw

Project ID: 15634525

4 Commits **2** Branches **0** Tags **246 KB** Files

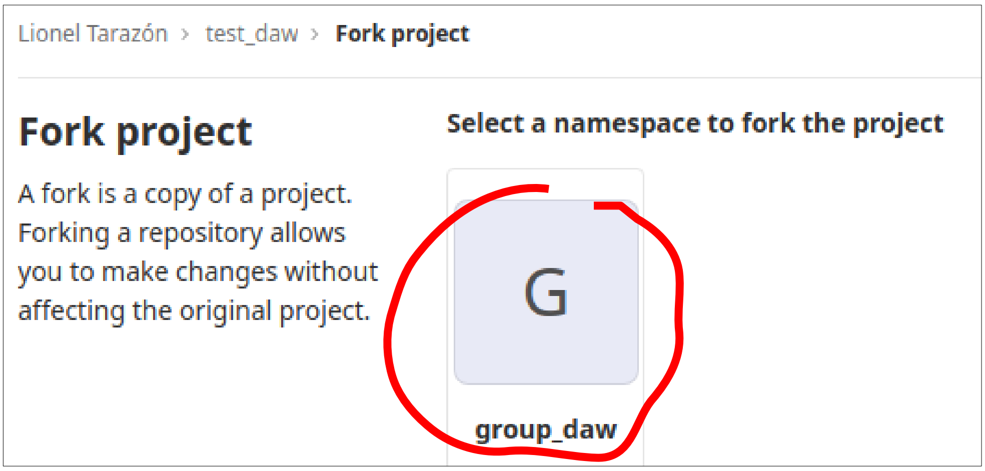
A simple test project.

Merge branch 'test_branch' into 'master'

Lionel Tarazón authored 3 minutes ago

c9d93c15

Now we have to choose where the fork will be done. In our case, we only can choose our group *group_daw*:



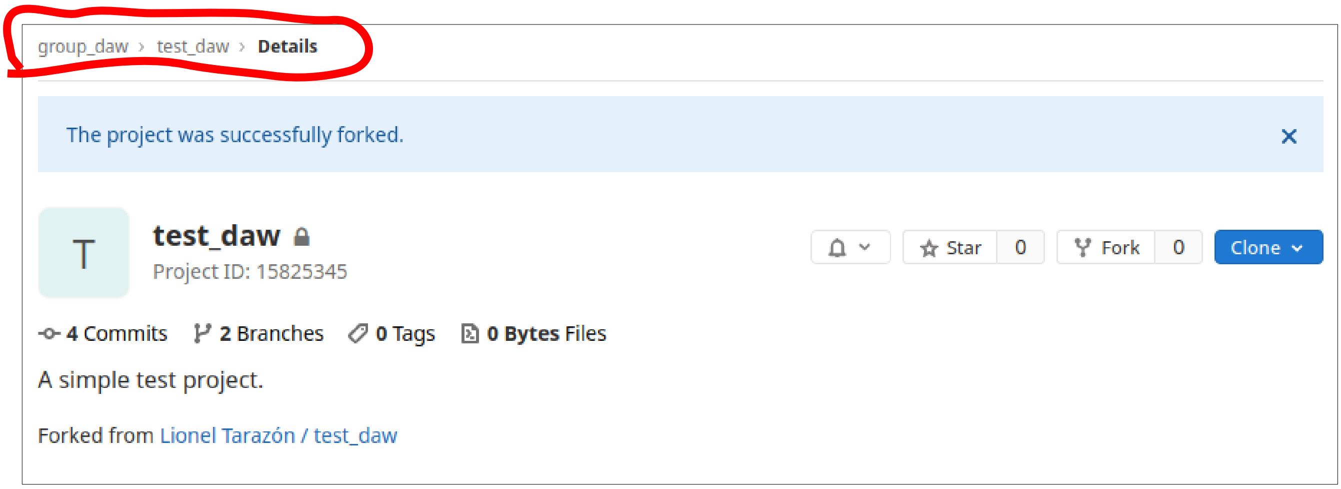
Lionel Tarazón › test_daw › **Fork project**

Fork project

A fork is a copy of a project. Forking a repository allows you to make changes without affecting the original project.

Select a namespace to fork the project

And the project will be forked. Now there are **two different copies of the test_daw repository**, one in **our personal repositories** and another one in **the group_daw group**.



group_daw › test_daw › Details

The project was successfully forked.

test_daw

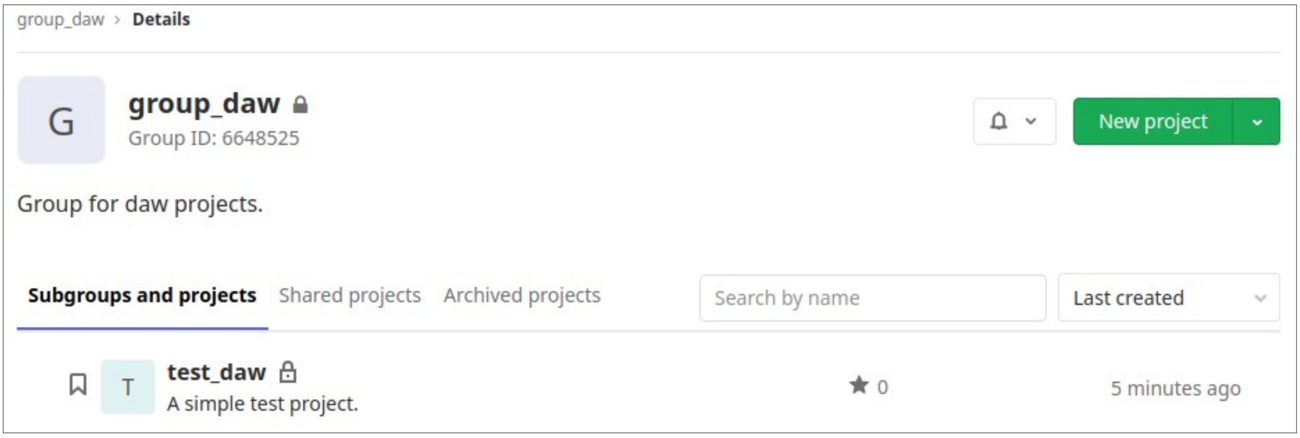
Project ID: 15825345

4 Commits 2 Branches 0 Tags 0 Bytes Files

A simple test project.

Forked from Lionel Tarazón / test_daw

Now if we access to our group *group_daw* we can see that it contains our project forked:



group_daw › Details

group_daw

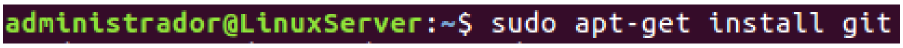
Group ID: 6648525

Group for daw projects.

test_daw A simple test project.	0	5 minutes ago

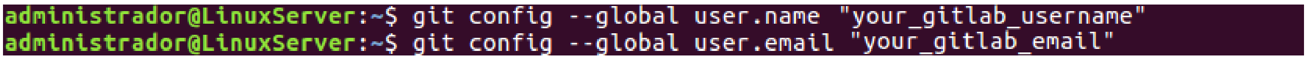
2.10 Using the command line

Now we are going to use the command line to work with our GitLab repository from our Linux machine. Make sure to use the same computer from which you created the SSH Key in section 2.3 of this activity. First of all we have to install Git in our machine:



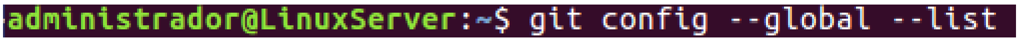
```
administrador@LinuxServer:~$ sudo apt-get install git
```

Now we have to add our GitLab username and email:



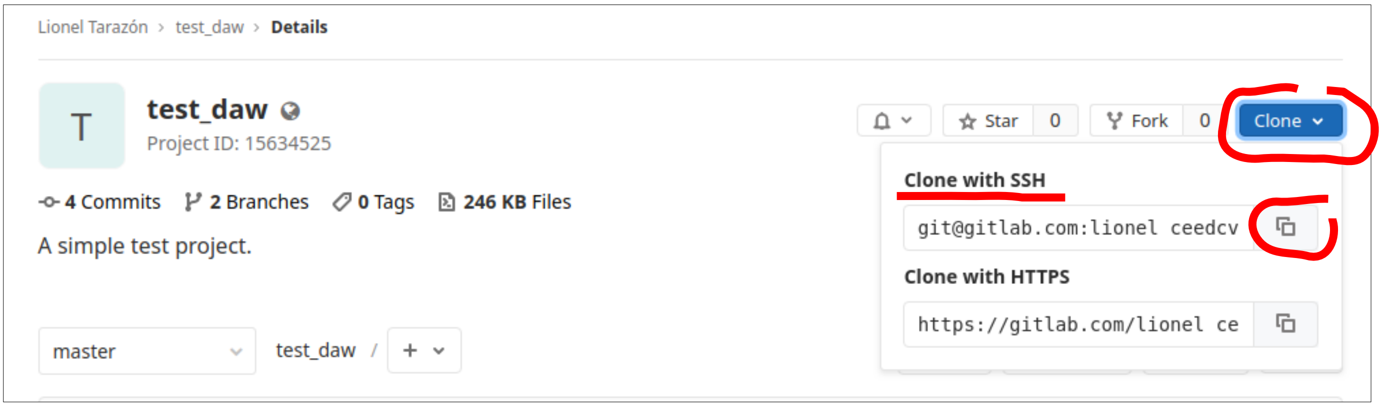
```
administrador@LinuxServer:~$ git config --global user.name "your_gitlab_username"
administrador@LinuxServer:~$ git config --global user.email "your_gitlab_email"
```

You can check the information added using:



```
administrador@LinuxServer:~$ git config --global --list
```

Now we are going to connect via SSH to our GitLab repository and clone it to our computer. Frist we need the project address (we can copy it from the project details).



Lionel Tarazón › test_daw › **Details**

test_daw

Project ID: 15634525

4 Commits 2 Branches 0 Tags 246 KB Files

A simple test project.

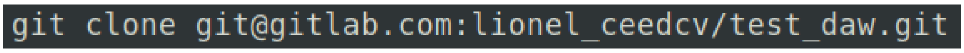
Clone with SSH

```
git@gitlab.com:lionel_ceedcv
```

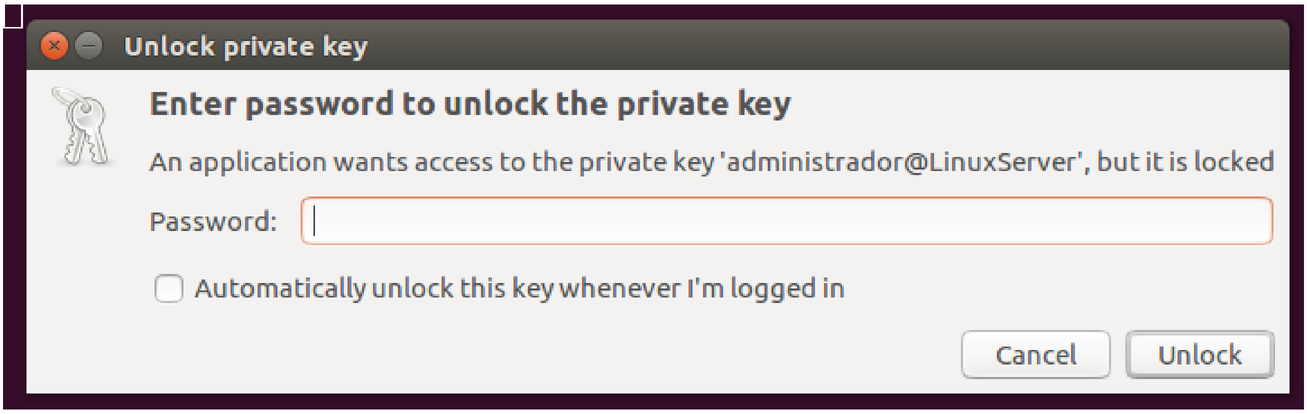
Clone with HTTPS

```
https://gitlab.com/lionel_ce
```

Now lets clone the repository. We will need to write the password of the private key we created In section 2.3 of this activity.



```
git clone git@gitlab.com:lionel_ceedcv/test_daw.git
```



Unlock private key

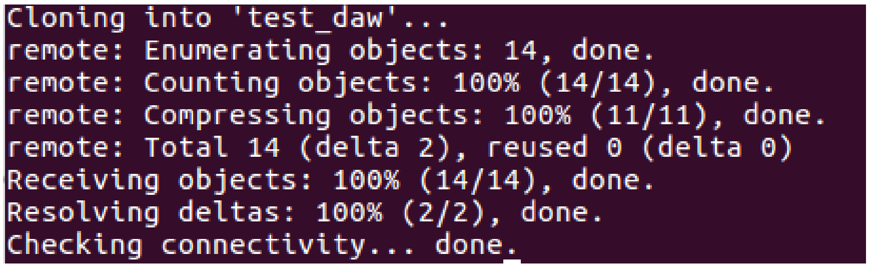
Enter password to unlock the private key

An application wants access to the private key 'administrador@LinuxServer', but it is locked

Password:

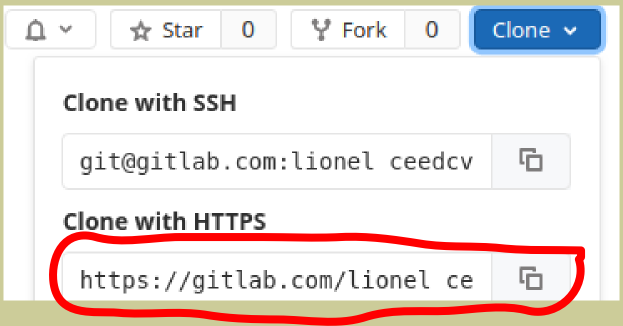
☐ Automatically unlock this key whenever I'm logged in

And the repository will be cloned to you computer:



```
Cloning into 'test_daw'...
remote: Enumerating objects: 14, done.
remote: Counting objects: 100% (14/14), done.
remote: Compressing objects: 100% (11/11), done.
remote: Total 14 (delta 2), reused 0 (delta 0)
Receiving objects: 100% (14/14), done.
Resolving deltas: 100% (2/2), done.
Checking connectivity... done.
```

📢 We can also clone via HTTPS using the GitLab HTTPS address of our project.



Clone with SSH

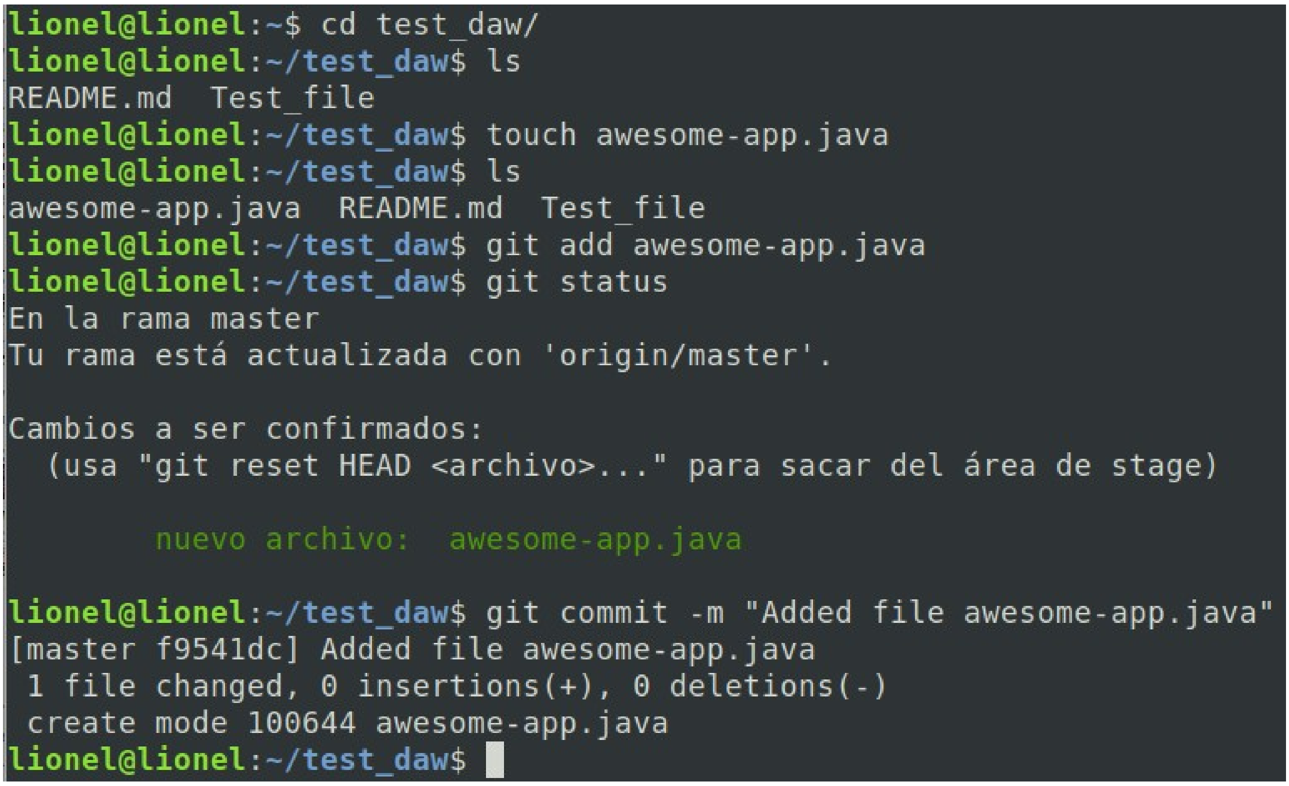
git@gitlab.com:lionel ceedcv

Clone with HTTPS

https://gitlab.com/lionel ce

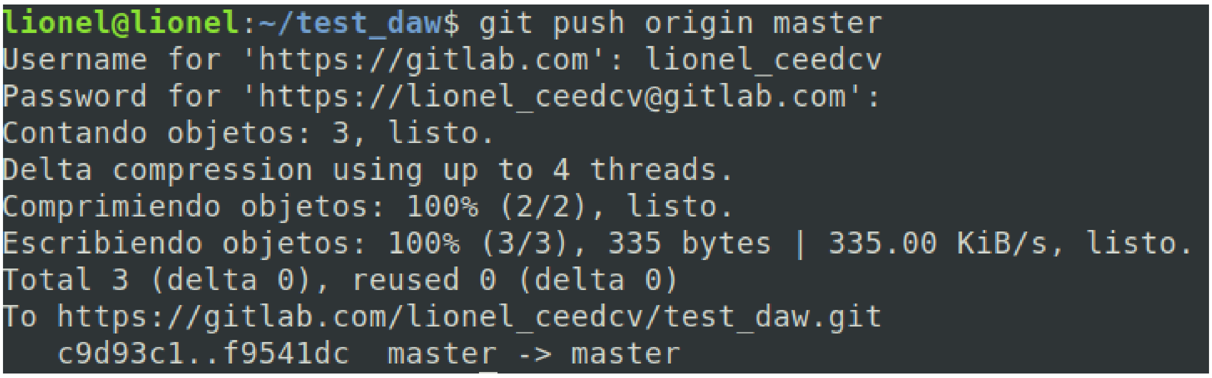
```
git clone https://gitlab.com/lionel_ceedcv/test_daw.git
```

Now we can work with the repository in our local computer. For instance, lets create a new file, and add it to the repository doing using git add and git commit.



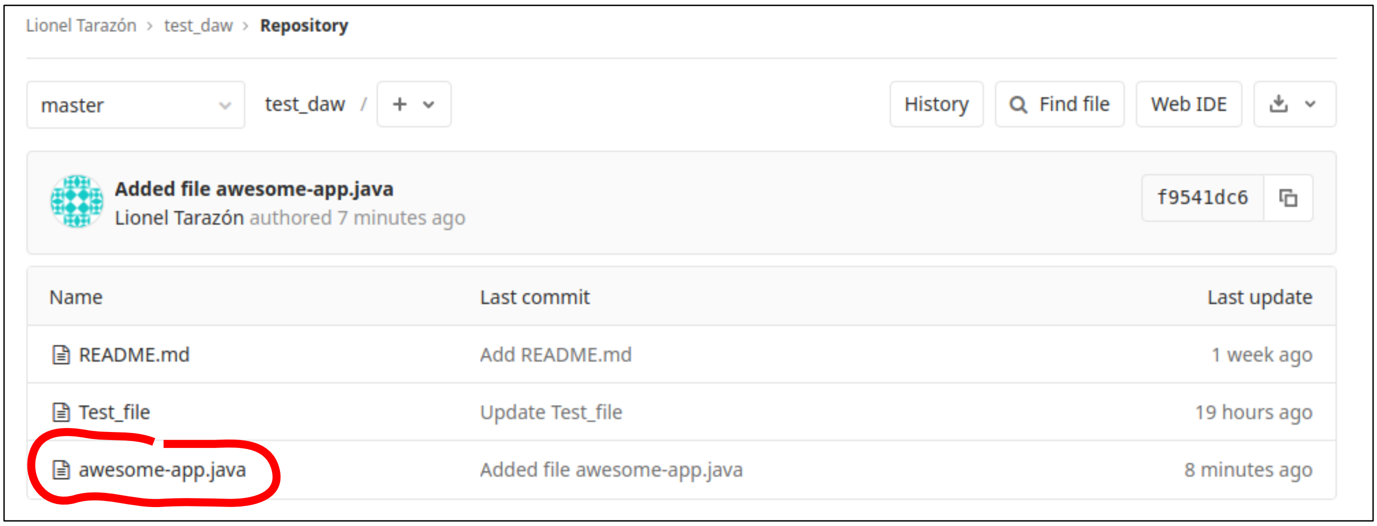
```
lionel@lionel:~$ cd test_daw/
lionel@lionel:~/test_daw$ ls
README.md  Test_file
lionel@lionel:~/test_daw$ touch awesome-app.java
lionel@lionel:~/test_daw$ ls
awesome-app.java  README.md  Test_file
lionel@lionel:~/test_daw$ git add awesome-app.java
lionel@lionel:~/test_daw$ git status
En la rama master
Tu rama está actualizada con 'origin/master'.

Cambios a ser confirmados:
  (usa "git reset HEAD <archivo>..." para sacar del área de stage)

        nuevo archivo:  awesome-app.java

lionel@lionel:~/test_daw$ git commit -m "Added file awesome-app.java"
[master f9541dc] Added file awesome-app.java
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 awesome-app.java
lionel@lionel:~/test_daw$ 
```

And finally we push it to GitLab:



```
lionel@lionel:~/test_daw$ git push origin master
Username for 'https://gitlab.com': lionel_ceedcv
Password for 'https://lionel_ceedcv@gitlab.com':
Contando objetos: 3, listo.
Delta compression using up to 4 threads.
Comprimiendo objetos: 100% (2/2), listo.
Escribiendo objetos: 100% (3/3), 335 bytes | 335.00 KiB/s, listo.
Total 3 (delta 0), reused 0 (delta 0)
To https://gitlab.com/lionel_ceedcv/test_daw.git
   c9d93c1..f9541dc  master -> master
```

If we take a look at our GitLab repository we can see the new file has been added.



Added file awesome-app.java
Lionel Tarazón authored 7 minutes ago
f9541dc6

Name	Last commit	Last update
README.md	Add README.md	1 week ago
Test_file	Update Test_file	19 hours ago
awesome-app.java	Added file awesome-app.java	8 minutes ago

When other collaborators working on this project do a git pull, their local repository will be updated.