



## Como instalar Discourse en Ubuntu 14.04

Alumno : *Federico Lochbaum*

Tema : *Discourse*

Repositorio del proyecto : <https://github.com/discourse/discourse>

## ¿Que es Discourse?

Discourse es una plataforma de discusiones web donde cada usuario puede crear, contestar y calificar vínculos, buscar temas por categoría, marcar temas como favoritos, etc. Su objetivo es crear una comunidad de un genero específico, puede ser una marca, una empresa, entre otros.

Discourse es muy flexible, ya que permite orientar la plataforma de acuerdo al tema, como por ejemplo un foro, o una red social. Además posee una interfaz muy simple y fácil de entender para el usuario, mantiene un sistema de notificaciones.

Discourse consume bastante recursos de hardware, necesita al menos 1GB de RAM, también necesita tener acceso a un servicio de envío de correo para la creacion y verificacion de cuentas en el sistema.

## ¿Que necesitamos para instalarlo?

Antes de instalar **Discourse** vamos a instalar **Docker**, docker es un sistema contenedor de programas, **Docker** no es un motor de virtualización ni de emulación, es más ligero que estas soluciones ya que los contenedores no tienen por qué incluir un sistema operativo (aunque pueden), si no que se apoyan en el sistema operativo anfitrión.

# Instalando Docker

Para instalar docker, vamos a verificar que el sistema cumple con sus requisitos, iremos a la consola y ejecutaremos `uname -r`

Nos debería dar una respuesta así, donde cumple con una arquitectura de 64 bits y kernel 3.10 o superior :

```
Archivo Editar Ver Buscar Terminal Ayuda
trimegisto@FedePC:~$ uname -r
3.16.0-73-generic
trimegisto@FedePC:~$
```

Una vez que verificamos que nuestro sistema cumple con los requisitos de docker, incluiremos docker ejecutando como root el siguiente comando :

```
apt-key adv --keyserver hkp://p80.pool.sks-keyserver.net:80 --recv-keys
58118E89F3A912897C070ADBF76221572C52609D
```

Ya que docker no está incluido en los repositorios de ubuntu debemos incluirlo de esta manera utilizando la clave gpg

Luego creamos el archivo `/etc/apt/sources.list.d/docker.list` para alojar el repositorio y le agregamos el mismo, para no hacerlo a mano, copiamos y pegamos este comando como root:

```
echo "deb https://apt.dockerproject.org/repo debian-jessie main" >
/etc/apt/sources.list.d/docker.list
```

Esto lo que dice es, escribi esta línea

**“deb <https://apt.dockerproject.org/repo> debian-jessie main”**

en el archivo alojado en esta dirección

**“/etc/apt/sources.list.d/docker.list”**

como no existe, lo crea por defecto.

Una vez que tenemos agregado el repositorio de docker , debemos comprobarlo y actualizarlo, entonces hacemos :

**apt-get update**

**apt-cache policy docker-engine**

Si se ejecutan ambos comandos con éxito, pasaremos a instalar y ejecutar nuestro contenedor, para esto ejecutamos por consola como root:

**apt-get install docker-engine**

**service docker start**

para verificar que hemos instalado docker con éxito podemos utilizar el siguiente comando :

**sudo docker run hello-world**

debería aparecerles algo así :

```
trimegisto@FedePC:~$ sudo docker run hello-world
[sudo] password for trimegisto:

Hello from Docker.
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker Hub account:
https://hub.docker.com

For more examples and ideas, visit:
https://docs.docker.com/engine/userguide/
```

Bien! ya estamos un paso más cerca de tener nuestra plataforma de discusiones con Discourse.

En algunos casos hace falta hacer una modificación en la configuración de Docker para asegurar su conexión a internet y la descarga del contenedor Discourse. Para ello se puede añadir la siguiente línea al archivo /etc/default/docker:

**DOCKER\_OPTS="--dns 8.8.8.8 --dns 8.8.4.4"**

Y luego reiniciamos Docker

**sudo service docker restart**

Cómo Discourse es un software que está en un repositorio de github necesitamos clonarlo, para esto es necesario tener instalado GIT

Si no tenes instalado git, es fácil, solo debes utilizar el siguiente comando :

**sudo apt-get install git**

**Ahora por fin pasaremos al proceso de instalación de Discourse**

# Instalando Discourse

Primero debemos crear la carpeta donde alojaremos nuestro Discourse la llamaremos “discourse”, luego usamos git para clonar el proyecto en el directorio creado, para esto haremos :

```
sudo mkdir /var/discourse
```

Y luego

```
sudo git clone https://github.com/discourse/discourse_docker.git  
/var/discourse
```

Una vez descargado el proyecto debemos configurarlo, antes de esto ejecutamos

```
cp /var/discourse/samples/standalone.yml  
/var/discourse/containers/app.yml
```

Vamos con la consola hasta “/var/discourse/containers/app.yml” y lo editamos como root , para esto yo use gedit, pueden editarlo cuando cualquier editor de texto.

Acá debemos configurar cada opción con nuestros respectivos datos, como verán hay muchas opciones, iremos viendo las más importantes y explicando las

*Lo primero que configuraremos es el dominio de nuestro discourse y el correo electrónico del usuario administrador, las opciones son :*

```
COURSE_HOSTNAME: 'example.net' # dominio  
COURSE_DEVELOPER_EMAILS: 'mail@example.net' # email del administrador
```

**OJO! las opciones de configuración están comentadas por defecto, debemos descomentar las y luego editarlas, los comentarios por línea se hacen con “#”**

Como la creacion y verificacion de cuentas dentro de nuestro Discourse se hace con un servicio de correo, debemos tener instalado localmente alguno de estos, actualmente estoy usando ssmtp para este proyecto.

Ahora hay que incluir los datos del mail relay. Por ejemplo, se puede usar GMail:

```
DISCOURSE_SMTP_ADDRESS: smtp.gmail.com
DISCOURSE_SMTP_PORT: 587 # (opcional)
DISCOURSE_SMTP_USER_NAME: usuario@gmail.com # (opcional)
DISCOURSE_SMTP_PASSWORD: gmailContraseña # (opcional, CUIDADO: el carácter '#' en una contraseña puede causar problemas)
DISCOURSE_SMTP_ENABLE_START_TLS: true # (optional, default true)
```

Pese a que los comentarios se explican solos, acá lo que hacemos es configurar el servicio de correo que utilizara discourse cuando esté conectado, recomiendo crearse un nuevo correo ya que debemos agregar usuario y contraseña (cualquier que observe el archivo de configuración podría observar nuestra contraseña) al archivo yml.

Por último configuraremos cuanta memoria RAM puede usar como máximo nuestro Discourse y cuantos usuarios en simultáneos acepta , ya que como dijimos Discourse exige bastante hardware, evitaremos que nos cuelgue la máquina, pero es configurable a gusto.

```
db_shared_buffers: "128MB"
UNICORN_WORKERS: 2
```

Por último debemos compilar el contenedor de Discourse con las opciones que hemos configurado y si todo va bien iniciará el contenedor.

Para acceder a Discourse basta visitar la URL que se haya definido con la variable :

DISCOURSE\_HOSTNAME en app.yml.

En caso de que editemos nuevamente discourse, por cada cambio que hagamos sobre /var/discourse/containers/app.yml tenemos que recompilar el proyecto, para esto usaremos :

```
./launcher rebuild app
```

## Como levantar Discourse con Apache

Para poder hacer esto debemos configurar Discourse como virtualhost de Apache y configurarlo para que escuche al contenedor Docker de Discourse en un puerto diferente al 80 mediante un proxy. Así se evita que Apache y Docker utilicen el mismo puerto.

para cambiar el puerto de Discourse editaremos nuestro archivo de configuración de la siguiente manera :

```
expose:  
- "85:80"
```

Esto redirige puerto 85 del host al 80 (http) del contenedor Docker

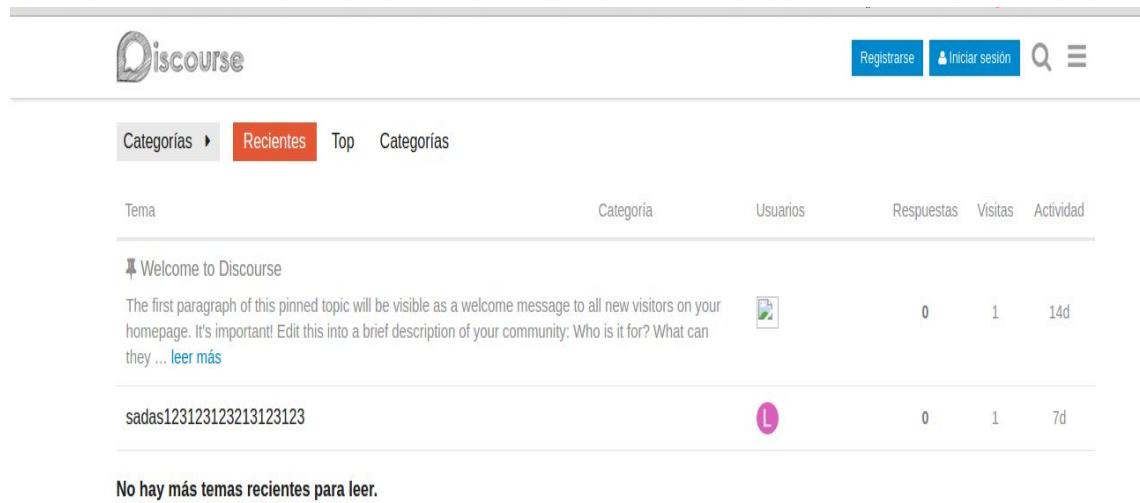
Luego creacion el VirtualHost para Discourse y configuración de apache :

```
<VirtualHost *:80>  
ServerName example.net  
ServerAlias www.example.net  
  
ProxyPreserveHost On  
ProxyRequests Off  
ProxyPass / http://example.net:85/  
ProxyPassReverse / http://example.net:85/  
  
DocumentRoot /var/discourse  
</VirtualHost>
```

Ponemos nuestro `serverName`, su alias y la ruta del documento, en nuestro caso `/var/dicourse`. Luego habilitamos los módulos `proxy` y `proxy_http` y reiniciamos apache :

```
sudo a2enmod proxy proxy_http
sudo service apache2 restart
```

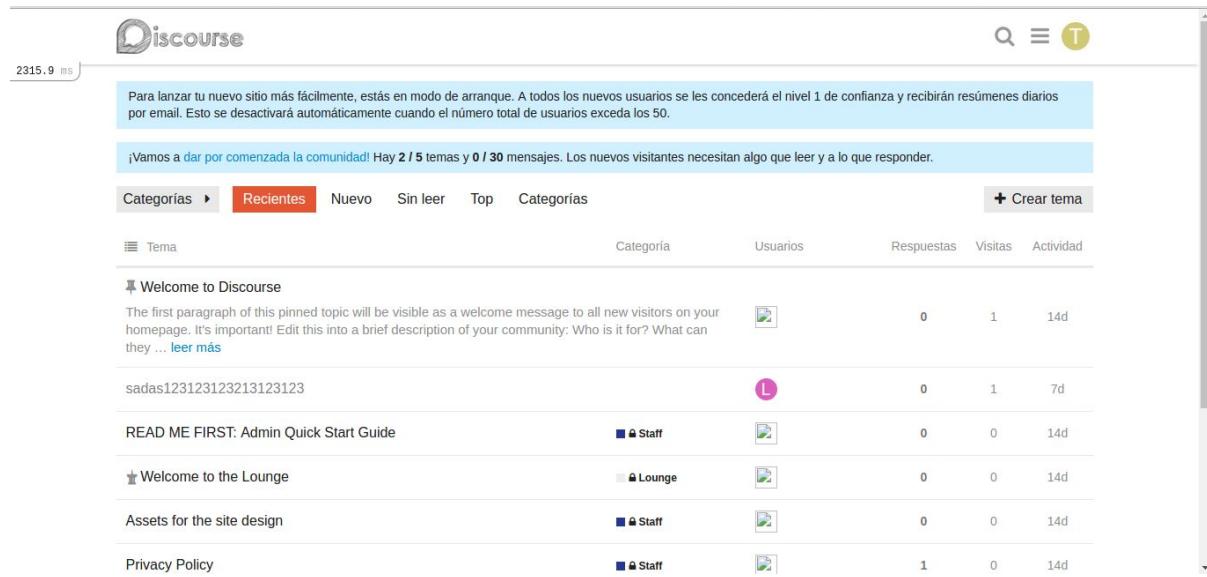
Vamos a la dirección donde configuramos discourse, en mi caso `localhost` en el puerto 85 y :



The screenshot shows the Discourse homepage. At the top, there is a navigation bar with the Discourse logo, registration and login buttons, a search icon, and a menu icon. Below the navigation bar, there is a header with buttons for 'Categorías', 'Recientes' (which is highlighted in red), 'Top', and 'Categorías'. The main content area displays a pinned topic with the title 'Welcome to Discourse'. The post content reads: 'The first paragraph of this pinned topic will be visible as a welcome message to all new visitors on your homepage. It's important! Edit this into a brief description of your community: Who is it for? What can they ... [leer más](#)'. Below this, there is a recent post by a user named 'sadas123123123123123123'. The post content is a long string of characters. At the bottom of the page, there is a message: 'No hay más temas recientes para leer.'

# ¿Como se usa discourse?

Primero que nada, debemos crearnos nuestra cuenta, con el email que colocamos como admin en la configuración del mismo, al registrarnos nos enviará un mail de confirmación utilizando ssmtp y el mail asignado en el .yml, validamos la cuenta copiando y pegando la url que nos llegó al correo. y accedemos con nuestra cuenta :



The screenshot shows the homepage of a Discourse forum. At the top, there is a banner with the text: "Para lanzar tu nuevo sitio más fácilmente, estás en modo de arranque. A todos los nuevos usuarios se les concederá el nivel 1 de confianza y recibirán resúmenes diarios por email. Esto se desactivará automáticamente cuando el número total de usuarios exceda los 50." Below the banner, a message says: "¡Vamos a dar por comenzada la comunidad! Hay 2 / 5 temas y 0 / 30 mensajes. Los nuevos visitantes necesitan algo que leer y a lo que responder." The navigation bar includes "Categorías", "Recientes" (which is highlighted in red), "Nuevo", "Sin leer", "Top", "Categorías", and a "Crear tema" button. The main content area displays a list of topics:

Tema	Categoría	Usuarios	Respuestas	Visitas	Actividad
Welcome to Discourse			0	1	14d
The first paragraph of this pinned topic will be visible as a welcome message to all new visitors on your homepage. It's important! Edit this into a brief description of your community: Who is it for? What can they ... <a href="#">leer más</a>					
sadas123123123213123123	L		0	1	7d
READ ME FIRST: Admin Quick Start Guide	Staff		0	0	14d
Welcome to the Lounge	Lounge		0	0	14d
Assets for the site design	Staff		0	0	14d
Privacy Policy	Staff		1	0	14d

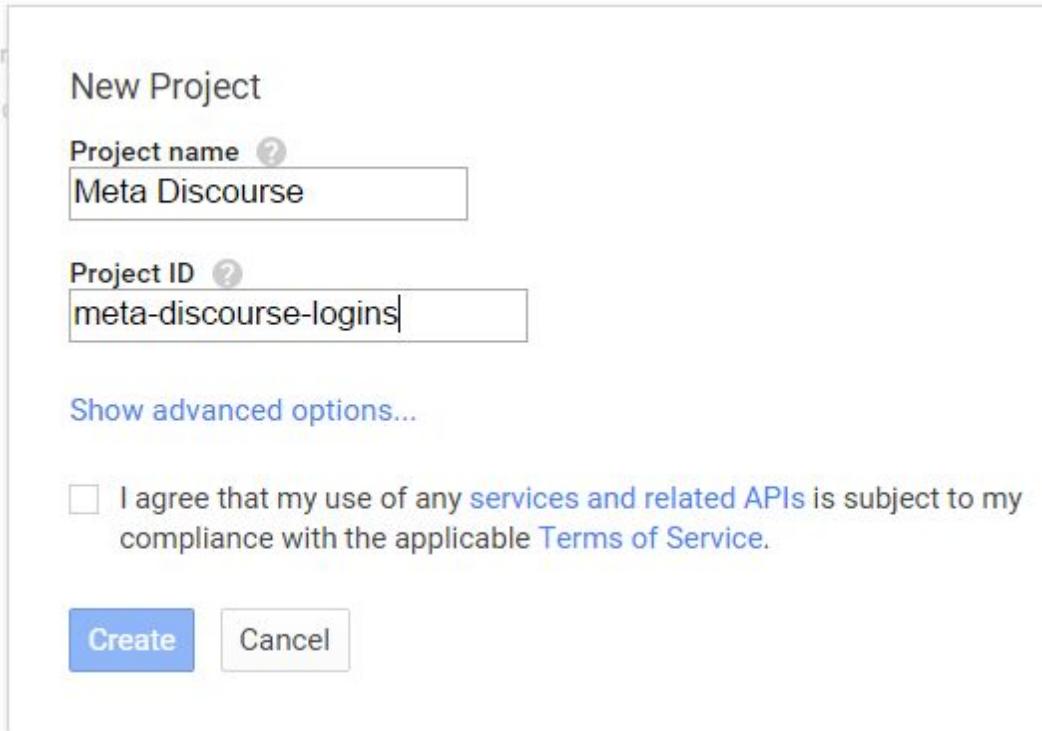
Se podría decir que ya tenemos instalado, funcionando el discourse, además estamos logueado como admin, pero ¿ qué podemos hacer como administrador ?

Como administrador podemos desde cambiar el idioma de la aplicación a instalar plugins sobre el discourse y hay muchos tipos de este último.

Como último paso pasare a explicar como usar OAuth2 y poder loguearse con google

## Discourse + OAuth2

Para esto, lo primero que debemos hacer es entrar [aca](#) donde crearemos un nuevo proyecto con el nombre que queramos.



The screenshot shows the 'New Project' dialog box. It has fields for 'Project name' (Meta Discourse) and 'Project ID' (meta-discourse-logins). There is a link to 'Show advanced options...' and a checkbox for agreeing to terms of service. At the bottom are 'Create' and 'Cancel' buttons.

New Project

Project name [?](#)  
Meta Discourse

Project ID [?](#)  
meta-discourse-logins

Show advanced options...

I agree that my use of any [services and related APIs](#) is subject to my compliance with the applicable [Terms of Service](#).

**Create** **Cancel**

Una vez dentro de nuestro proyecto iremos a "APIs & auth" y luego a *Credentials* ahí seleccionaremos *addCredential* o *agregar Credencial* , dependiendo el lenguaje de tu browser.

Seleccionaremos Web application y lo configuraremos :

#### Create client ID

##### Application type

- Web application
- Android [Learn more](#)
- Chrome App [Learn more](#)
- iOS [Learn more](#)
- PlayStation 4
- Other

##### Name

##### Authorized JavaScript origins

Enter JavaScript origins here or redirect URIs below (or both) [?](#)

Cannot contain a wildcard (`http://*.example.com`) or a path (`http://example.com/subdir`).

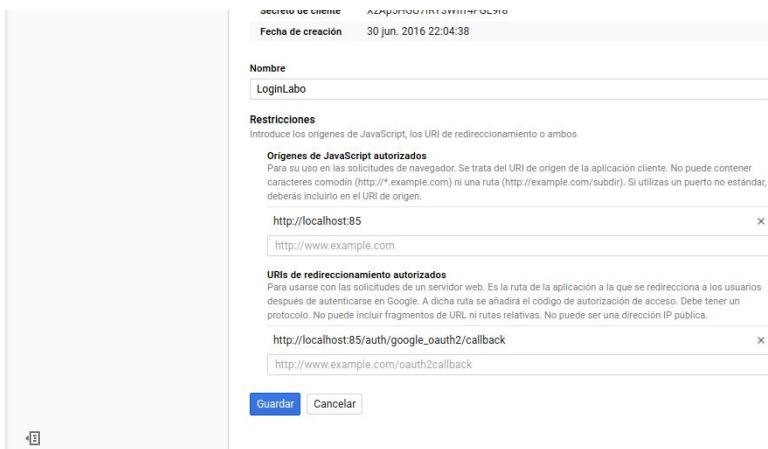
[X](#)

##### Authorized redirect URIs

Must have a protocol. Cannot contain URL fragments or relative paths. Cannot be a public IP address.

[X](#)[Create](#)[Cancel](#)

Un problema muy común en este paso es no asignar bien las direcciones URL en la configuración de la credencial, utilizando localhost quedaria así :



The screenshot shows the 'Client ID' configuration page in the Google Cloud Platform. The 'Nombre' field is set to 'LoginLabo'. The 'Orígenes de JavaScript autorizados' field contains 'http://localhost:85'. The 'URIs de redireccionamiento autorizados' field contains 'http://localhost:85/auth/google\_oauth2/callback' and 'http://www.example.com/oauth2callback'. The 'Guardar' button is visible at the bottom.

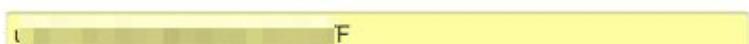
Una vez creada y configurada la credencial, Google nos dará un Client ID el cual será público y un Secret ID el cual será privado, los guardaremos y habilitaremos las apis que vamos a utilizar en nuestro proyecto , en este caso serán dos "Contacts API" y "Google+ API".

Para esto, iremos a "APIs & auth" luego a APIs buscamos, seleccionamos y habilitamos estas. Hecho esto ya configuramos todo lo necesario desde google para poder utilizarlo como login, solo falta configurar nuestro Discourse.

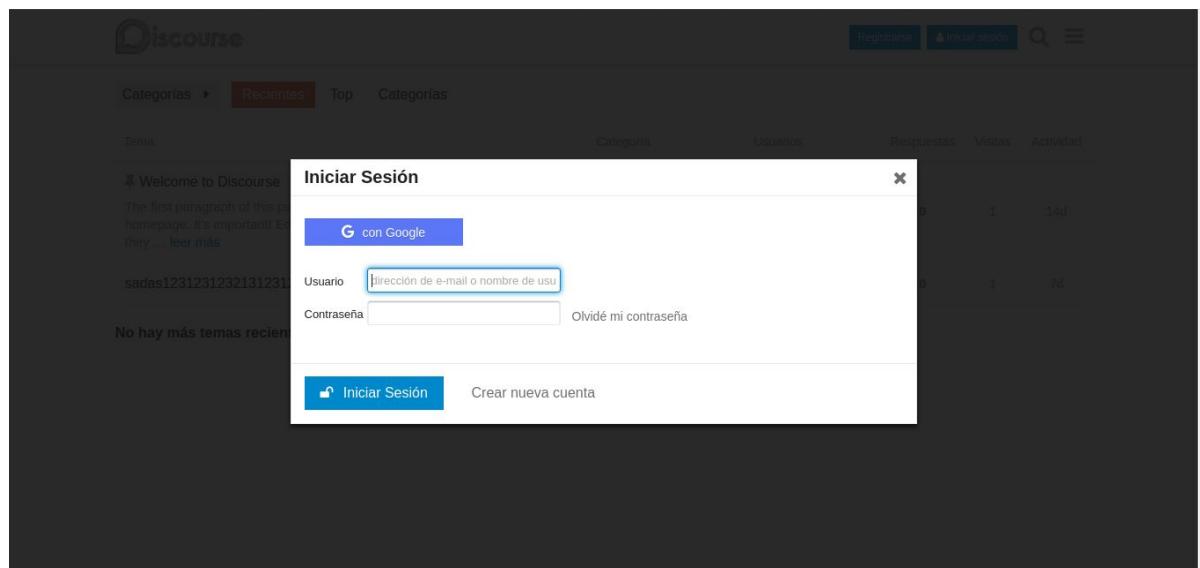
Ahora vamos a nuestro discourse, logueamos como administrador , seleccionaremos la pestaña “Ajustes del sitio” , en “login” y habilitaremos la opción “enable google oauth2 logins” e ingresamos nuestro Client ID y Secret ID en las siguientes dos entradas :

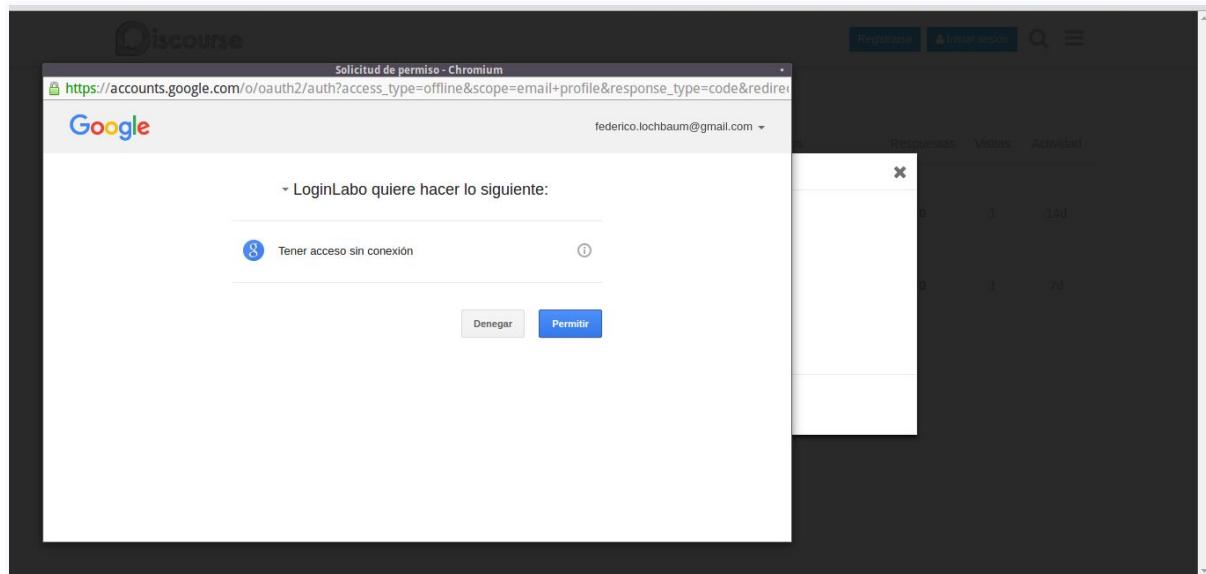
check → **enable google oauth2 logins**  Enable Google Oauth2 authentication. This is the method of authentication that Google currently supports. Requires key and secret.

**google oauth2 client id**  Client ID of your Google application.

**google oauth2 client secret**  Client secret of your Google application.

Acto seguido, guardamos la configuración, deslogueamos y verificamos si podemos loguearnos con google, de haber seguido al pie de la letra todos los pasos deberia abrirte una pestaña donde habilitas a la aplicación Discourse a obtener tus datos de usuario de google para poder loguearse con el mismo.





## Fuentes

[Para la instalación de Discourse](#)

[Para la instalación de Docker](#)

[Configuración de Apache + Docker](#)

[Configuración de OAuth0 + Discourse](#)