Open Project

- Install Open Project in Docker (2023)

Install Open Project in Docker (2023)

https://www.youtube.com/embed/SF2OvG9l2Ko

Open Project is a business grade tool for keeping track of project goals, objectives, activities, assignments, and completions toward any project that may help benefit your business. The project has been more recently adapted to also work quite well as a product management tool to help make sure software projects can generate a roadmap, and set objectives and goals that are attainable using the scrum and agile processes of the software industry.

If you run a business, you'll likely have several projects throughout the life of your business that are intended to improve your business offering, grow your business and solidify your place in whatever market you serve. Using software to help you organize that effort is one of the best things you can do. Adding your team, or even if it's just you, and setting goals, breaking those down into mmilestones of completion, and even further into attainable deliverables can often mean the difference between the success or failure of a project.

What you'll need:

- A server to run the software on (either a self owned machine, or a VPS in the cloud)
- Docker and Docker Compose installed on the machine
- A Reverse Proxy (NGinX Proxy Manager is the one I use)
- A Domain Name or Subdomain with the ability to set an A-record in DNS to point it to your public IP address
- git installed on your host server
- About 30 minutes of your time.

Installation via a Simple Script

You can easily install Docker-CE, Docker-Compose, Portainer-CE, and NGinX Proxy manager by using this quick install script I created and maintain on Github. Just use the command:

```
wget https://gitlab.com/bmcgonag/docker_installs/-/raw/main/install_docker_nproxyman.sh
```

To download the script to your desired host.

Change the permissions to make the script executable:

`chmod +x ./install_docker_nproxyman.sh`

and then run the script with the command:

`./install_docker_nproxyman.sh`

When run, the script will prompt you to select your host operating system, then will ask you which bits of software you want to install.

Simply enter 'y' for each thing you want to install.

At some point, you may be asked for your super user (sudo) password as well.

Allow the script to complete installation.

At this point, you might want to log out and back in, as this will allow you to use the `docker` and `docker-compose` commands without the need of sudo in front of them.

Install OpenProject

Let's first setup our folder structure. We'll create a folder called "docker". This is where you'll want to put any docker applications you install on this server.

`mkdir -p docker`

Now move into that folder with

`cd docker`

Next, we'll clone the open project repository (always check the official documentation at https://www.openproject.org/docs/installation-and-operations/installation/docker/ to make sure you are getting the most up to date version. As of today it's version 12.

`git clone https://github.com/opf/openproject-deploy --depth=1 --branch=stable/12 openproject`

When this command completes. you'll have a new folder called 'openproject' in your docker folder.

Move into this new folder with

`cd openproject`

If we do an `ls` we can see that there are several files and folders. We want to move into the 'compose' folder to continue.

`cd compose`

`ls -al`

Now, let's copy eht '.env.example' file to a new file called '.env', and let's copy the existing 'docker-compose.yml' file to a backup in case we need the original later.

`cp .env.example .env`

`cp docker-compose.yml docker-compose.backup.yml`

We'll first want to edit the values in our new '.env' file. This is where all of our configuration variables are stored.

`nano .env`

```
##
# All environment variables defined here will only apply if you pass them
# to the OpenProject container in docker-compose.yml under x-op-app -> environment.
# For the examples here this is already the case.
#
# Please refer to our documentation to see all possible variables:
#   https://www.openproject.org/docs/installation-and-operations/configuration/environment/
#
TAG=12
OPENPROJECT_HTTPS=false
OPENPROJECT_HOST__NAME=<proj.my-awesome-domain.org>
PORT=8080 # feel free to change this to a less common port (I used 8085)
# OPENPROJECT_RAILS__RELATIVE__URL__ROOT=
IMAP_ENABLED=false
DATABASE_URL=postgres://postgres:<some-long-strong-password-and-
number5s>@db/openproject?pool=20&encoding=unicode&reconnect=true
RAILS_MIN_THREADS=4
RAILS_MAX_THREADS=16
PGDATA="/var/lib/postgresql/data"
OPDATA="/var/openproject/assets"
POSTGRES_PASSWORD=<some-long-strong-password-and-number5s> # this should match the password
above
```

Using the file above, you can modify your .env file accordingly. The items with a less than '<' and greater than '>' sign around them need to be changed to valid values for your site. The port '8080' can be changed to a less common port if you like.

We will come back after the initial setup and change the value for OPENPROJECT_HTTPS=false to be 'true' as well. But for now, leave it as 'false'.

Save the file with CTRL + O, then press Enter to confirm, and use CTRL + X to exit the nano editor.

Inside the docker-compose.yml file we just need to make a single adjustment. We want to set the volumes to be kept inside this foldeer where we have the docker-compose.yml file. To do this we'll adjust the lines that say the following:

`nano docker-compose.yml`

`- "${OPDATA:-opdata}:/var/openproject/assets"`

and

`- "${PGDATA:-pgdata}:/var/lib/postgresql/data"`

to say

`- "./opdata:/var/openproject/assets"`

and

`- "./pgdata:/var/lib/postresql/data"`

You'll find each line under the `volumes:` section in the `app` and `db` sections respectively.

Save the file with CTRL + O, then press Enter to confirm, and use CTRL + X to exit the nano editor.

It's time to run our server. We'll use two commands concatenated with the double ampersand `&&` for this:

`docker compose up -d && docker compose logs -f`

You'll see the images being pulled down and started, then once started you'll see log output of the images as they start. What we want to look for, is the lines that say "200 OK" in them somewhere. This is our indicator that the site is up and running.

You can also just wait about 5 or so minutes afterward to try the site.

While you're waiting, it's a perfect time to setup your reverse proxy.

Setting Up the Reverse Proxy

In NGinX Proxy Manager, you want to add a new proxy host. For the domain name, you'll enter the name you put for the OPENPROJECT_HOST_NAME in the .env file earlier.

Next, in the IP / Host field, enter the private IP of your host machine (this is if you're running inside of a local area netowrk). If you are running NPM on the same host as Open Project, you can also enter 'localhost'. In the port field, enter the port you entered for the PORT variable in the .env file earlier.

Tick the options that say 'Block Common Exploits', and 'Websocket Support', an dclick 'Save'.

After several minutes of the docker server starting have passed, if all has gone according to plan, you should be able to go to your domian for open project and be greeted with the main open project page.

Setup SSL

If you are able to see your open project site, let's make sure you're accessing it securely using SSL and HTTPS with a free CA Certified LetsEncrypt certificate.

first, in NPM, click on the 3-dot icon to the right end of the open project row in your proxy hosts page. In the menu that opens select 'Edit'. In the pop-up window, click the 'SSL' tab, and then from the first drop menu change the selection from 'none' to 'Request a new Certificate'.

Now tick the options for 'Force SSL', 'HTTP/2 Support', ensure your email is entered, and tick the option to accept the LetsEncrypt Terms of Service. Then click 'Save'. Allow it to work, and it should issue you a new SSL certificate. The pop-up window should close without any errors being displayed. This is a good sign that the certificate was issues. In the row on the proxy hosts page, you should see a column that now says 'LetsEncrypt'.

Click the domain in the row, or go to your open project domain again in your browser, and you should be automatically re-routed to the HTTPS page.

Awesome! Almost done.

Finally, go back to your terminal, and edit the .env file.

```
nano .env
```

Change the value of OPENPROJECT_HTTPS=false to say OPENPROJECT_HTTPS=true, then save with CTRL + O, press Enter to confirm, and exit then nano editor with CTRL + X.

Now we'll restart our containers to pick up these changes with

```
docker compose up -d
```

Give the containers time to restart. You can optionally follow the logs again with

```
docker compose logs -f
```

or simply be patient for about 5 to 10 minutes for everything to get up and running again, then re-visit your site. You are now accessing your own Open Project site with full HTTPS encryption.

You can login with the default user credentials

username: admin

password: admin

Make sure to change these immediately to something much more secure. You can start looking around the site, go through the initial walk-through, an dbegin learning how to run your projects with professional level tools!

Support My Channel and Content

Support my Channel and ongoing efforts through Patreon:
https://www.patreon.com/awesomeopensource