# n8n.io - Workflow Automation

- n8n Workflow Automation

# n8n Workflow Automation

https://www.youtube.com/embed/lpRsqiBc2JQ

n8n is a freely licensed (based on the Apache 2.0, but more permissive) workflow automation tool set.  It enables you to drag and drop nodes for various tools to generate amazing information workflows.

Be sure to check out the end of the article for a discount code for 20%  off of any of the n8n hosted / licensed options.

Say you have a customer interest, or sales lead form, and you want the information entered in that form to be added to a database, and to also send a notification to you  through your chat application (like RocketChat or Slack, etc).  This is exactly what n8n does.  It allows you to use drag and drop nodes for various tools, to create a workflow where you can push data from a web form into n8n, potentially pretty up the data you receive, and then push that data down a pipeline of nodes to go into other applications and perform tasks automatically.  You imagination is really thte limiting factor with something like n8n.

# What You'll Need

- Docker and Docker-Compose installed on a host server / system.
- (optional) NGinX Proxy Manager or some other reverse proxy if you want access from the outside interente.
- About 10 minutes of your time

# Installing Docker-CE and Docker-Compose

If you already have Docker and Docker-Compose installed, feel free to skip down to the next section.

You may want to install some pre-requisite softwre as well:

Debian / Ubuntu

```
sudo apt install git curl wget
```

Fedora / Redhat

`dnf install git curl wget`

Arch

`sudo pacman -Sy git curl wget`

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 permissios 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.

For instance, you may want to answer 'y' to NGinX Proxy Manager, and Portainer-CE if you don't already use these in your system.

At some point, you'll 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.

# Installing n8n

Installing n8n is fairly straight forward.  There are only a couple of files we'll need to setup, and for the most part, it will be copy / paste.

First, let's prepare our folder structure. I like to have everything I run in docker, inside of a folder called "docker".  This way I simply backup the entire "docker" folder, and everything is backed up in case of catastrophic failure.

To create a new "docker" folder, just do the following in the terminal:

`mkdir docker`

Now, let's move into the "docker" folder, and create a new folder called "n8n".

`cd docker`

`mkdir n8n`

Finally, let's move into our "n8n" folder and start creating the files we need.

`cd n8n`

Now, we'll create a folder and two files for our application. First, let's create a hidden folder called ".n8n" where we'll put a configuration file.

NOTICE: there is a period "." in front of our folder name for ".n8n". This is important as it makes this a *hidden* folder in our system.  You can only see it if you do the `ls` command with the `-a` argument afterward like this `ls -a`.

`mkdir .n8n`

Mow move into the ".n8n" folder and create a new file called "config".

`cd .n8n`

`nano config`

Inside this file, paste the following lines of text:

```
{
        "encryptionKey": "abcdefg1234567890ABCdefG0123456789"
}
```

Once pasted into your new file, remove the placeholder key "abcdefg1234567890ABCdefG0123456789", and replace the characters with a key of your onw. The key should be long and strong, so make sure it's got upper and lower case letters and numbers randomly.

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

Next, we'll go back one level in our folder structure with the command:

`cd ..`

You should be back in the "docker/n8n" folder now.  Let's create our dokcer-compose.yml file with

`nano docker-compose.yml`

Now paste the code block below into that file:

```
version: '3.3'
services:
  n8n:
    container_name: n8n
    ports:
      - 5678:5678
    environment:
      - WEBHOOK_URL=https://n8nio.your-great-domain.org/
      - EXECUTIONS_PROCESS=main
    volumes:
      - ./.n8n:/home/node/.n8n
    image: n8nio/n8n
    restart: unless-stopped
```

In the above file, the only two things you may want to change are:

1. the left side of the port mapping.  If your host machine (the machine / server you are running docker on) has port 5678 already in use, then change the left side port to a port number that is not in use on your host. Otherwise, feel free to leave it as 5678.
2. The WEBHOOK_URL value should be changed to either your hosts private IP address on your local network (LAN), or to the domain / subdomain you want to use for yoru n8n instance.  NOTE: You should own the domain name you intend to use.

Now, we can run our docker-compose file with the command

`docker-compose up -d`

If you want to see logging of the output of your container as it starts use the command:

`docker-compose logs -f` after you see the 'done' message from the `docker-compose up -d` command.

I like to run them in a single line like this:

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

This way I get to see the log output immediately after the container is finished starting.

To exit the logs, use the hotkey combination:

`CTRL + C`

# Test your Install

You can now open your favorite modern browser and go to the LAN ip address of your host, and port 5678 to see that the n8n startup screen loads.

In my case I went to `http://192.168.10.42:5678`

If you don't want / need / intend to run n8n from outside your local network, then you can just continue to use your system as it is now.

If, however, you do want to have access to your n8n system, then you'll want to set it to use a domain / sub-domain. For that we need to setup a reverse proxy.

I used NGinX Proxy Manager, but feel free to use any reverse prosy you like.

# Setting Up a Reverse Proxy

If you used my script to install docker and docker-compose, then you also had the option to install NGinX Proxy Manager.  If you didn't do it initially, not to worry, just re-run the script, and this time choose 'y' when asked about NGinX Proxy Manager, and it should be installed and started. You'll want to go in, and change the defaults set in the compose file, then restart it for security.  Make sure you forward ports 80 and 443 to your NGinX Proxy Manager host machine's IP address in your router.

Now, open the UI for NPM, and let's cretae our reverse proxy.  You can call your site url anything you want as long as you own the domain / sub-domain you are using.

I called mine `auto.routemehome.org`.

Type that into your url field, then press tab.  Next, move to the IP field, and enter the IP address of your host machine for n8n.  If you are running n8n on the same docker install as NPM, then you can use the docker0 IP address as well.

If you setup NPM and n8n to run on the same docker network (a little mroe advnced, but definitely do-able), you can refer to n8n by container name instead of IP - note: these cannot be on the default network).

Now in the port field enter 5678 (or whatever you changed the left side of the port mapping to in the compose file).

Tick the options for "Block Common Exploits" and "Websocket Support",t hen click 'Save'.

Test your new entry and make sure the page loads as expected. If not, open it up in edit mode, and check for mistakes.  if you don't find any, ensure you have an A-record pointing to your external IP, that you ISP doesn't blcok access to port 80 and / or 443, and that you ahve setup your port-forwarding properly in your router. At the very least, you should see the generic NGinx Proxy Manager 'Congratulations' page.

If everything worked, let's get your SSL certificate. Open the NPM item in edit mode by clicking the 3-dots at the right, and selecting 'Edit' from the drop-down list.

Move to the 'SSL' tab, and click the drop-down that says 'None'. Select, request a new certificate, then tick the box for 'Force SSL', and the item for 'Accept the LetsEncrpt Terms of Service', and make sure your email is filled in.

Click 'Save' and be patient as LetsEncrypt challenges your domain and issues an SSL certificate. If the Edit pop-up closes with no errors, test your site again, and you should now be accessing your n8n site with SSL.

Congratulations, you can start making workflows on n8n!

Make sure to check out the video at the top of this article for all the cool things you can do with n8n.

# Get a Discount on n8n Hosted Services.

Use the Dicsount Code voucher:

AWESOMEOPENSOURCE

to receive a 20% dicsount on n8n services at checkout.  This is an offer from n8n to my viewers / subscribers, and I truly appreciate them offering you all an opprotunity to save on their services.

# Support my Channel and Content

Support my Channel and ongoing efforts through Patreon:

https://www.patreon.com/bePatron?u=234177