Install Moodle

Moodle is an LMS (Learning Management System). It's a very large and complex system, intended for setting up, creating, hosting, and scheduling learning sessions. These can be either or both online learning, or in class / in person style learning.

it's a powerful, open source system, and for a business in IT or an MSP, a valuable service to be able to rpovide. Particularly when considering you may need to train client users on best practices for better online security, privacy, and safe habits. Perhaps you have setup a new software for a client, and you need to provide training for their users on how to create an account, login, navigate, and perform the basic functions of the software.

Today we'll install moodle, and get you setup to provide e-Learning services.

What You'll Need

- A server or VM / Container to install the application software on.
- Docker and Docker Compose installed on the host server / VM / Container
- (optional) a domain / sub-domain to host your site at (e.g. myclasses.mygreatdomain.com)
- (optional) a Reverse Proxy to help get traffic onto the right service.
- About 20 minutes of your time.

What this tutorial is not:

This is not an in depth overview of how to configure or use the moddle software, either as an admin or student. It is massive, and far too much to cover in a video or online tutorial. Instead, I highly recommend you seek out other videos on more specific areas of Moodle, and definitely check out their extensive user and admin guides.

Installing Docker-CE (Community Edition) and Docker Compose

1. Setup a server with a non-root, sudo user.

`add user <username>`

`usermod -aG sudo <username>` or on CentOS, Fedora, RedHat `usermod -aG wheel <username>`

1. Install Docker-CE and Docker Compose

`curl https://get.docker.com | sh`

1. Add your non-root user to the docker group.

`sudo usermod -aG docker <username>`

1. Log out and back in.

Setup for Our Moodle Install

1. Create a folder structure for our Moodle application:

`mkdir -p docker/moodle`

1. Inside the new folder, we need to create a file called "compose.yaml", and three folders.

`cd docker/moodle`

`mkdir moodle_data`

`mkdir mariadb_data`

`mkdir moodledata_data`

`nano compose.yaml`

1. We need to add the code to our `compose.yaml` file now that it's created and open.

```
---
services:
  mariadb:
    image: docker.io/bitnami/mariadb:latest
    container_name: moodle_mysql
    environment:
      - MARIADB_ROOT_PASSWORD=<a-long-strong-password>
      - MARIADB_PASSWORD=<a-different-long-strong-password>
      - MARIADB_USER=moodle
      - MARIADB_DATABASE=moodle
      - MARIADB_CHARACTER_SET=utf8mb4
      - MARIADB_COLLATE=utf8mb4_unicode_ci
    volumes:
      - ./mariadb_data:/bitnami/mariadb
  moodle:
```

```
    image: docker.io/bitnami/moodle:4.5
    container_name: moodle
    ports:
      - 80:8080
      - 443:8443
    environment:
      - MOODLE_DATABASE_HOST=mariadb
      - MOODLE_DATABASE_PORT_NUMBER=3306
      - MOODLE_DATABASE_USER=moodle
      - MOODLE_DATABASE_NAME=moodle
      - ALLOW_EMPTY_PASSWORD=no
      - MOODLE_DATABASE_PASSWORD=<a-different-long-strong-password> # <-- same as the
MARIADB_PASSWORD from the first section.
      - MOODLE_HOST=<name.mygreatdomain.org>
      - MOODLE_REVERSEPROxY=true # <-- if you're using a reverse proxy make it true, otherwise
false
      - MOODLE_SSLPROXY=true # <-- if you are using LetsEncrypt through your proxy, again make
it true, otherwise false
      - MOODLE_SMTP_HOST=<smtp.somemailprovider.com>
      - MOODLE_SMTP_PORT=587
      - MOODLE_SMTP_USER=<leanring@somemailprovider.com>
      - MOODLE_SMTP_PASSWORD=<some-email-password-for-the-user-above>
      - MOODLE_SMTP_PROTOCOL=tls
      - MOODLE_LANG=en
      - MOODLE_USERNAME=<admin_username>
      - MOODLE_PASSWORD=<admin_password>
    volumes:
      - ./moodle_data:/bitnami/moodle
      - ./moodledata_data:/bitnami/moodledata
    depends_on:
      - mariadb
```

In the above file, there are multiple values you'll need to change, and make unique to your installation and setup. I have marked those fields by surrounding the value in less than and greater than signs ("<" and ">").

In a few cases I also added a comment about the values, in case you need to adjust them for some reason. Finally, and most importantly, make sure the values for MARIADB_PASSWORD in the first section and MOODLE_DATABASE_PASSWORD in the second section match exactly. This is important for the application to function.

When done setting your values, save the file with CTRL + O, then press Enter to confirm, and exit the file with CTRL + X.

Now let's pull our images down from dockerhub or github.

```
docker compose pull
```

Once pulled, we'll start our virtual machines (containers) with

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

The above is a set of two commands concatenated with two ampersands (&&). The first command tells docker compose to start the containers from the images we pulled down, and run those containers in detached mode (-d). The second command tells docker compose to show us the logs when the containers start, and follow (-f) the output as it's generated.

When we are done viewing the logs, we can use CTRL + C, to stop seeing them being output to the screen.

Our system should now be up and running, but if we want our users to be able to access it easily, we need to setup a reverse proxy entry so they can use a URL instead of an IP Address and port number.

Reverse Proxy Setup

I use NginX Proxy Manager, but feel free to use any reverse proxy you are comfortable with.

Pre-setup Requirements.

1. You must own the domain that you are setting up for this site, and have the ability to
2. Set A-Records for subdomains and / or this main domain.
3. Add new Subdomains to the main domain.

I own "opensourceisawesome.com", and I created an A-record that points `*.opensourceisawesome.com` to my public IP address where NGinX Proxy Manager is running.

```
A  -->  *.opensourceisawesome.com --> 76.24.31.143
```

Now< I can create any subdomain I need inside NGinX Proxy Manager, and tell it where the service for that subdomain is running.

In NGinX Proxy Manager, I created `learn.opensourceisawesome.com` and told it to point ot the private IP address of my VM running the Moodle install. In the "Port" field, I entered `80`, as this is what's mapped on the left side of the port mapping in our "compose.yaml" file.

I ticked the two options for "Block Common Exploits" and "Websockets Support", then moved to the SSL tab.

In the SSL Tab, I changed "None" to "Request a new certificate", then ticked the boxes for "Force SSL", "HTTP/2 Support", and both options with `HSTS` in them. Finally, I ticked the box to accept the LetsEncrypt terms of service, and made sure my email was entered.

Click 'Save'. This will take anywhere form 20 to 45 seconds, but if the pop-up windown in NGinX Proxy Manager just closes with no errors, you should now be able to access your domain / subdomain in your favorite browser.

You will see a fairly blank starting page, as there is nothing configured so far. In order to configure your site, you'll want to go to your URL and add `/admin` to the end of it. My domain is

`https://leanr.opensourceisawesome.com/admin`

Once, there, use the username (MOODLE_USERNAME) and password (MOODLE_PASSWORD) you setup in your "compose.yaml" file to login as an admin user.

From here, the settings and configuration are available for you to begin configuring your Moodle learning site.

Moodle is a very massive, powerful, flexible site. I highly recommend you read their documentation, set aside time to learn how to configure the site, and even take your own notes about configurations you are making. It's far too much to cover in a video, or this article, but it's there and ready for you to jump into.

Support My Channel and Content

Support my Channel and ongoing efforts through Patreon:

https://www.patreon.com/awesomeopensource

Buy me a Beer / Coffee: https://paypal.me/BrianMcGonagill

Revision #4
Created 2025-04-15 11:31:45 UTC by Brian McGonagill
Updated 2025-04-15 11:53:19 UTC by Brian McGonagill