Install Music Assistant

https://www.youtube.com/embed/dIk-6wDtTB8

Setup a Server

In most self hosted applications these days, you'll need a machine to act as a server. This can be a machine in your home / business (such as an old laptop or desktop, a Raspberry PI or Single Board Com#puter (SBC), or even the computer you're reading this article from), a VM / Container hosted on one of your machines, or a VPS (Virtual Private Server) hosted by companies like RackNerd, Digital Ocean, Linode, Vultr, and so many more. Regardless of which option you choose, you'll want to do a few things to get the server setup properly.

Install updates to our Server

Ubuntu / Debian

```
sudo apt update && sudo apt upgrade -y
```

RedHat / CentOS / Fedora / Alma / Rocky

```
sudo dnf update -y
```

Add a non-root / sudo user on the server

Generally, when you setup a new server, the VPS (Virtual Private Server) service sets up a default "root" user for you. It's considered unsafe to do everything as "root", so let's setup a non-root user who has super user (sudo) privileges.

`adduser <username>` You'll be prompted to enter and confirm a password for this user. You'll also be asked for some user information like Name, etc, but this is not required information. At the end, confirm the entries, and you'll have your new user.

Next, we need to add the user to the super user group.

Ubuntu / Debian

```
usermod -aG sudo <username>
```

RedHat / CentOS / Fedora / Alma / Rocky

```
usermod -aG wheel <username>
```

Now, you can log out of the system, and log back in as your new non-root super user.

Install Docker and Docker Compose

Fortunately, there is a single line command that will install both Docker and Docker Compose for us on most Linux based distributions.

We need the 'curl' utility to get this to work, so if you don't have it, you'll want to install it first with

Ubuntu / Debian

```
sudo apt install curl -y
```

RedHat / CentOS / Fedora / Alma / Rocky

```
sudo dnf install curl -y
```

Next, we'll run the command to install Docker and Docker Compose:

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

You may be prompted to enter your super user password, so be ready for it. Once you do, the install should proceed.

Once complete. we want to add our user to the 'docker' group so we can do `docker` and `docker compose` commands without having to type in `sudo` each time.

```
sudo usermod -aG docker <username>
```

Now we'll logout / exit the session, and log right back in so the updated group will take effect.

Stand Alone Install of Music Assistant

First, let's create a folder on our server for our install:

```
mkdir -p docker/musicassistant
```

Now we'll move into that folder

```
cd docker/musicassistant
```

Now create a new file with a text editor (I like to use nano):

```
nano compose.yaml
```

Copy the yaml code block below, and paste it into the editor window.

```
---
services:
  music-assistant-server:
    image: ghcr.io/music-assistant/server:latest
    container_name: music-assistant-server
    restart: unless-stopped
    network_mode: host
    volumes:
      - ./music-assistant-server/data:/data/
      - </path/to/your/music/collection>:/media
    cap_add:
      - SYS_ADMIN
      - DAC_READ_SEARCH
    security_opt:
      - apparmor:unconfined
    environment:
      # default=info, possible=(critical, error, warning, info, debug)
      - LOG_LEVEL=info
```

In this code block, you'll want to make a couple of changes to make sure the application works for your music collection (if you are planning to use a local collection).

In this case, if you are wanting to use local music (whether on the same disk as the container you are running, or on a mounted volume such as NFS or SMB), then you should change the volumen mapping on the left side of the colon ':' where it says

`<path/to/your/music/collection>`

and enter the actual proper path to link your music collection to the container's internal folder labeled `/media` .

If you are not including locally hosted music, then you can remove that line completely.

Use CTRL + O to save your changes, press Enter to confirm, and exit the nano editor with CTRL + X.

Start Our Application in Docker Compose

First, let's pull down the images needed to run our container (tiny virtual machine).

`docker compose pull`

Once that pulls down, we can start the application with:

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

This is actually two commands concatenated together with the `&&` symbol. The first command tells docker to bring up the container, and the second command says we want to follow `-f` the logs as the application starts. We are just looking quickly for any obvious error messages. Once the logging slows, we can stop following the logs with CTRL + C.

Now open your browser and type in the IP address of your host server, and the port 8095. For instance I went to:

http://192.168.10.36:8095

You should be greeted by the Music Assistant system, up and running.

!!! A Word About Security

Music Assistant does not use a user model, and there is no login screen for this application. That said, it would be a very bad idea to host your Music Assistant system directly on the open internet with a domain name and no other protections. This type of implementation would open your home music system, any device (player providers), your connected online music accounts (music providers), and your local music collection to the world! This is not a good thing.

If, however you run a reverse proxy like Pangolin, then you can mitigate this by setting up the Pangolin authentication in front of the proxied services / sites. I have not personally set this up, but would be willing to do a video on it if you are interested. Comment on the linked video to let me know.

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 #1
Created 2025-08-19 13:35:57 UTC by Brian McGonagill
Updated 2025-08-19 13:39:17 UTC by Brian McGonagill