# Install and Setup Anchor Notes

https://pt.opensourceisawesome.com/videos/embed/jryCGvmMbpM5VMTHZ3hNLA

## Link to the Project

https://github.com/zhfahim/anchor

# Install OS and Update

Update and upgrade your server's packages using the following commands:

- For Ubuntu/Debian:

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

- For RedHat/CentOS/Fedora/Alma/Rocky:

```
sudo dnf update -y
```

## Create a Non-Root User

Create a non-root user with superuser (sudo) privileges:

1. Add a new user using

```
adduser <username>
```

2. Set the password for this user.

3. Enter the relevant information (optional)

4. Enter 'Y', then press Enter.

5. Add the user to the "sudo" group:

   - For Ubuntu/Debian:

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

- For 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

Install Docker and Docker Compose on your server:

1. Install the curl utility:

- For Ubuntu/Debian:

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

- For RedHat/CentOS/Fedora/Alma/Rocky:

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

2. Run the command to install Docker and Docker Compose:

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

# Add Your User to the Docker Group

Add your non-root user to the docker group so you can use Docker commands without sudo:

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

# Install Anchor in Docker

We'll make a directory for our installation with

`mkdir -p docker/anchor`

Let's move into the folder with

`cd docker/anchor`

We'll make our 'compose.yaml' file which will hold our application setup configuration.

```
---
services:
  anchor:
    image: ghcr.io/zhfahim/anchor:latest
    container_name: anchor
    restart: unless-stopped
    ports:
      - "3000:3000"
    volumes:
      - ./anchor_data:/data
```

You can edit the left side port number on the port mapping if your host happens to be using port 3000 for some other service.

For instance, if you need to use port 8212, then your port mapping should look like

```
    ports:
      - "8212:3000"
```

With this setup, you can save the file with CTRL + O, then press Enter to confirm, and exit the nano editor with CTRL + X.

We can pull our images with

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

This is two commands concatenated into one line. The first command `docker compose up -d` tells docker compose to bring up the containers based on any images pulled down for the application or service. The second part, `docker compose logs -f` tells docker compose to show me the log output live `follow` so that we can look for any errors or issues. You don't need to run the second part of the command every time, just the first time, or if you make changes to your 'compose.yaml'.

Generally you can just do `docker compose up -d`.

If you don't see any errors in the logs, you can stop watching the logs with CTRL + C.

Before we get on to checking out the User Interface, let's talk about a few other options you can set in the 'compose.yaml' file.

First, you can setup your own generated JWT_SECRET. To do this, run this command in your terminal:

`openssl rand -base64 32`

This will generate a 32 character, random, base 64 key for you to use. NOTE: If you don't set this up, the system autogenerates one for you.

Next, we have the option to use your own external PostgreSQL database. Again, if you don't set these up, the system will generate a PostgreSQL database for you and use it. I prefer the included DB as this makes it easier to back everything up all together.

| `PG_HOST` | **No** | **(empty)** | **External Postgres host (leave empty for embedded)** |
|---|---|---|---|
| `PG_PORT` | No | `5432` | Postgres port |
| `PG_USER` | No | `anchor` | Postgres username |
| `PG_PASSWORD` | No | `password` | Postgres password |
| `PG_DATABASE` | No | `anchor` | Database name |

Finally, we have the option to setup our own authentication provider such as Authentik, Authelia, Keycloak, etc. The system supports OIDC (Open ID Connect) for authentication, which means you get the option for SSO with a bunch of your other applications.

| `OIDC_ENABLED` | **No** | **—** | **Enable OIDC authentication** |
|---|---|---|---|
| `OIDC_PROVIDER_NAME` | No | `"OIDC Provider"` | Display name for the login button |
| `OIDC_ISSUER_URL` | When OIDC enabled | — | Base URL of your OIDC provider |
| `OIDC_CLIENT_ID` | When OIDC enabled | — | OIDC client ID |
| `OIDC_CLIENT_SECRET` | No | — | OIDC client secret. Omit for public client (PKCE) |
| `DISABLE_INTERNAL_AUTH` | | | |

To add these items, or the PostgreSQL variables to your 'compose.yaml' file, just edit it with

`nano compose.yaml`

Then add the `environment` section, and add each item under as shown below. Fill in the values that are correct for your setup, of course.

```
---
services:
  anchor:
    image: ghcr.io/zhfahim/anchor:latest
```

```
    container_name: anchor
    restart: unless-stopped
    ports:
      - "3000:3000"
    volumes:
      - ./anchor_data:/data
    environment:
      - JWT_SECRET=<some long string of chars and numbs>
      - PG_HOST=10.21.215.121
      - PG_PORT=24314
      - PG_USER=anchor
      - PG_PASSWORD=<some long strong password>
      - PG_DATABASE=anchor
      - OIDC_ENABLED=true
      - OIDC_PROVIDER_NAME=Authentik
      - OIDC_ISSUER_URL=https://your-oidcprovider.net/o/auth/somename
      - OIDC_CLIEND_ID=5omelonGCLientId
      - OIDC_CLIENT_SECRET=someRalLyL0ngS3cre7Th4atI5N07PubLiC
      - DISABLE_INTERNAL_AUTH=true # if you are using OIDC
```

Once you've added the environment variables you need, save the file with CTRL + O, then press Enter to confirm, and exit the nano editor with CTRL + X.

Again, run `docker compose up -d` to restart your application, and take in the configuration updates.

# Test the User Interface

Once you've setup the app and started it running, open your favorite modern browser, and navigate to the IP address of your host machine, and the port on which you setup the application.

Something like `https://192.168.1.44:3000`

If everything has gone properly, you'll be presented with the login screen (or int he case of SSO, the option to get to your SSO login).

You should now be able to setup an account and / or login to an account and start using Anchor for all of your note taking needs.

# Setup a Reverse Proxy

We use a reverse proxy when we want to access our web applications from outside of the local network where it's hosted. IF you setup Anchor on a VPS with a Public IP address, you don't

specifically need a reverse proxy to route to it, but reverse proxies can help with keeping traffic controlled through ACLs (Access Control Lists), obfuscate your applications public IP, and get freely available LetsEncrypt SSL certificates to ensure the encryption of your data across the internet.

I use the Pangolin Reverse Proxy, but there are many out there to choose from. If you prefer a different one, i assume you know how to set it up. If you are looking for a tutorial to help you get a Reverse Proxy setup, then let me point you to one of mine.

Install and Configure Pangolin Reverse Proxy

# Support this Channel and Content

## Become a Patron at Patreon

Check me out on Patreon

## Be me a Coffee or Beer at Paypal

Paypal Support for Awesome Open Source

## Get the Awesome Open Source Merch

Great Merch for Open Source Advocates!

---

Revision #4
Created 2026-02-28 12:08:16 UTC by Brian McGonagill
Updated 2026-04-15 12:54:07 UTC by Brian McGonagill