

# Docker Chatwoot Production deployment guide

Link: <https://www.chatwoot.com/docs/self-hosted/deployment/docker/>

## Pre-requisites

Before proceeding, make sure you have the latest version of `docker` and `docker-compose` installed.

As of now[at the time of writing this doc], we recommend a version equal to or higher than the following.

```
$ docker --version
Docker version 20.10.10, build b485636
$ docker compose version
Docker Compose version v2.14.1
```

■ **Note** Container name uses dashes instead of underscores by default with new docker/compose versions. If you are using an older version of docker/compose, replace `-` with `_`. Also, use `docker-compose` instead of `docker compose`.

## Steps to deploy Chatwoot using docker-compose

```
# Download the env file template
wget -O .env https://raw.githubusercontent.com/chatwoot/chatwoot/develop/.env.example

# Download the Docker compose template
wget -O docker-compose.yaml https://raw.githubusercontent.com/chatwoot/chatwoot/develop/docker-compose.production.yaml
```

### 1. Install Docker on your VM

```
# example in ubuntu
apt-get update
```

```
apt-get upgrade
curl -fsSL https://get.docker.com -o get-docker.sh
sudo sh get-docker.sh
apt install docker-compose-plugin
```

## 2. Download the required files

3. Tweak the `.env` and `docker-compose.yaml` according to your preferences. Refer to the available environment variables. You could also remove the dependant services like `Postgres`, `Redis` etc., in favor of managed services configured via environment variables.

```
# update redis and postgres passwords
nano .env
# update docker-compose.yaml same postgres pass
nano docker-compose.yaml
```

## 4. Prepare the database by running the migrations.

```
docker compose run --rm rails bundle exec rails db:chatwoot_prepare
```

## 5. Get the service up and running.

```
docker compose up -d
```

6. Your Chatwoot installation is complete. Please note that the containers are not exposed to the internet and they only bind to the localhost. Setup something like Nginx or any other proxy server to proxy the requests to the container.

If you want to verify whether the installation is working, try `curl -I localhost:3000/api` to see if it returns `200`. Also, you could temporarily drop the `127.0.0.1:3000:3000` for rails to `3000:3000` in the compose file to access your instance at `http://<your-external-ip>:3000`. It's recommended to revert this change back and use Nginx or some proxy server in the front.

# Additional Steps

1. Have an `Nginx` web server acting as a reverse proxy for Chatwoot installation. So that you can access Chatwoot from `https://chat.yourdomain.com`

2. Run `docker compose run --rm rails bundle exec rails db:chatwoot_prepare` whenever you decide to update the Chatwoot images to handle the migrations.

## Configure Nginx and **Let's Encrypt**

1. Configure Nginx to serve as a frontend proxy.

```
sudo apt-get install nginx
cd /etc/nginx/sites-enabled
nano yourdomain.com.conf
```



2. Use the following Nginx config after replacing the `yourdomain.com` in `server_name` .

```
server {
    server_name <yourdomain.com>;

    # Point upstream to Chatwoot App Server
    set $upstream 127.0.0.1:3000;

    # Nginx strips out underscore in headers by default
    # Chatwoot relies on underscore in headers for API
    # Make sure that the config is set to on.
    underscores_in_headers on;
    location /.well-known {
        alias /var/www/ssl-proof/chatwoot/.well-known;
    }

    location / {
        proxy_pass_header Authorization;
        proxy_pass http://$upstream;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_set_header X-Forwarded-Ssl on; # Optional

        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
```

```
proxy_http_version 1.1;
proxy_set_header Connection "";
proxy_buffering off;

client_max_body_size 0;
proxy_read_timeout 36000s;
proxy_redirect off;
}
listen 80;
}
```

- 
3. Verify and reload your Nginx config by running the following command.

```
nginx -t
systemctl reload nginx
```

- 
4. Run **Let's Encrypt** to configure **SSL certificate**.

```
apt install certbot
apt-get install python3-certbot-nginx
mkdir -p /var/www/ssl-proof/chatwoot/.well-known
certbot --webroot -w /var/www/ssl-proof/chatwoot/ -d yourdomain.com -i nginx
```

- 
5. Your Chatwoot installation should be accessible from the `https://yourdomain.com` now.

# Steps to build images yourself

We publish our base images to the Docker hub. You should be able to build your Chatwoot web/worker images from these base images.

## Web

```
FROM chatwoot/chatwoot:latest
RUN chmod +x docker/entrypoints/rails.sh
ENTRYPOINT ["docker/entrypoints/rails.sh"]
```

```
CMD bundle exec bundle exec rails s -b 0.0.0.0 -p 3000
```

■

## worker

```
FROM chatwoot/chatwoot:latest
RUN chmod +x docker/entrypoints/rails.sh
ENTRYPOINT ["docker/entrypoints/rails.sh"]
CMD bundle exec sidekiq -C config/sidekiq.yml
```

■

The app servers will run available on port `3000`. Ensure the images connect to the same database and Redis servers. Provide the configuration for these services via [environment variables](#).

## Initial database setup

To set up the database for the first time, you must run `rails db:chatwoot_prepare`. You may get errors if you try to run `rails db:migrate` at this point.

## Upgrading

If you're not using the `latest` or `latest-ce` tag, you first need to change the desired tag in your docker-compose file

After that you can pull the new image and start using them:

```
docker compose pull
docker compose up -d
```

■

Finally you may need to update the database: `docker compose run --rm rails bundle exec rails db:chatwoot_prepare`

## Running Rails Console

```
docker exec -it $(basename $(pwd))-rails-1 sh -c 'RAILS_ENV=production bundle exec rails c'
```



# Chatwoot CE edition docker images

If you want to run Chatwoot CE edition, replace the docker image tag with equivalent foss version tag. Docker tag for current `master` would be `latest-ce`. Version specific tags would follow the pattern `v*-ce`. For example the docker ce edition tag for Chatwoot `v2.3.2` would be `v2.3.2-ce`.

---

Revision #1

Created 13 September 2024 14:49:22 by Administrador

Updated 13 September 2024 14:52:11 by Administrador