# Welcome to the contributing guide for PeerTube

Interested in contributing? Awesome!

\*\*This guide will present you the following contribution topics:\*\*

<!-- START doctoc generated TOC please keep comment here to allow auto update -->

<!-- DON'T EDIT THIS SECTION, INSTEAD RE-RUN doctoc TO UPDATE -->

- [Translate](#translate)

- [Give your feedback](#give-your-feedback)

- [Write documentation](#write-documentation)

- [Improve the website](#improve-the-website)

- [Develop](#develop)

- [Prerequisites](#prerequisites)

- [Online development](#online-development)

- [Server side](#server-side)

- [Client side](#client-side)

- [Client and server side](#client-and-server-side)

- [Testing the federation of PeerTube servers](#testing-the-federation-of-peertube-servers)

- [Unit tests](#unit-tests)

- [Emails](#emails)

- [Plugins & Themes](#plugins--themes)

<!-- END doctoc generated TOC please keep comment here to allow auto update -->

## Translate

You can help us to translate the PeerTube interface to many languages! See [the documentation](/support/doc/translation.md) to know how.

## Give your feedback

You don't need to know how to code to start contributing to PeerTube! Other

contributions are very valuable too, among which: you can test the software and

report bugs, you can give feedback on potential bugs, features that you are

interested in, user interface, design, decentralized architecture...

## Write documentation

You can help to write the documentation of the REST API, code, architecture,

demonstrations.

For the REST API you can see the documentation in [/support/doc/api](https://github.com/Chocobozzz/PeerTube/tree/develop/support/doc/api) directory.

Then, you can just open the `openapi.yaml` file in a special editor like [http://editor.swagger.io/](http://editor.swagger.io/) to easily see and edit the documentation. You can also use [redoc-cli](https://github.com/Redocly/redoc/blob/master/cli/README.md) and run `redoc-cli serve --watch support/doc/api/openapi.yaml` to see the final result.

Some hints:

\* Routes are defined in [/server/controllers/](https://github.com/Chocobozzz/PeerTube/tree/develop/server/controllers) directory

\* Parameters validators are defined in [/server/middlewares/validators](https://github.com/Chocobozzz/PeerTube/tree/develop/server/middlewares/validators) directory

\* Models sent/received by the controllers are defined in [/shared/models](https://github.com/Chocobozzz/PeerTube/tree/develop/shared/models) directory

## Improve the website

PeerTube's website is [joinpeertube.org](https://joinpeertube.org), where people can learn about the project and how it works – note that it is not a PeerTube instance, but rather the project's homepage.

You can help us improve it too!

It is not hosted on GitHub but on [Framasoft](https://framasoft.org/)'s own [GitLab](https://about.gitlab.com/) instance, [FramaGit](https://framagit.org): https://framagit.org/framasoft/peertube/joinpeertube

## Develop

Don't hesitate to talk about features you want to develop by creating/commenting an issue

before you start working on them :).

### Prerequisites

First, you should use a server or PC with at least 4GB of RAM. Less RAM may lead to crashes.

Make sure that you have followed

[the steps](/support/doc/dependencies.md)

to install the dependencies.

Fork the github repository,

and then clone the sources and install node modules:

```

$ git clone https://github.com/Chocobozzz/PeerTube

$ git remote add me git@github.com:YOUR\_GITHUB\_USERNAME/PeerTube.git

$ cd PeerTube

$ yarn install --pure-lockfile

```

Note that development is done on the `develop` branch. If you want to hack on

Peertube, you should switch to that branch. Also note that you have to repeat

the `yarn install --pure-lockfile` command.

When you create a new branch you should also tell to use your repo for upload

not default one. To do just do:

```

$ git push --set-upstream me <your branch name>

```

Then, create a postgres database and user with the values set in the

`config/default.yaml` file. For instance, if you do not change the values

there, the following commands would create a new database called `peertube\_dev`

and a postgres user called `peertube` with password `peertube`:

```

# sudo -u postgres createuser -P peertube

Enter password for new role: peertube

# sudo -u postgres createdb -O peertube peertube\_dev

```

Then enable extensions PeerTube needs:

```

$ sudo -u postgres psql -c "CREATE EXTENSION pg\_trgm;" peertube\_dev

$ sudo -u postgres psql -c "CREATE EXTENSION unaccent;" peertube\_dev

```

In dev mode, administrator username is \*\*root\*\* and password is \*\*test\*\*.

### Online development

You can get a complete PeerTube development setup with Gitpod, a free one-click online IDE for GitHub:

[![Open in Gitpod](https://gitpod.io/button/open-in-gitpod.svg)](https://gitpod.io/#https://github.com/Chocobozzz/PeerTube)

### Server side

You can find a documentation of the server code/architecture [here](https://docs.joinpeertube.org/#/contribute-architecture?id=server-code).

To develop on the server-side:

```

$ npm run dev:server

```

Then, the server will listen on `localhost:9000`. When server source files

change, these are automatically recompiled and the server will automatically

restart.

### Client side

You can find a documentation of the client code/architecture

[here](https://docs.joinpeertube.org/#/contribute-architecture?id=client-code).

To develop on the client side:

```

$ npm run dev:client

```

The API will listen on `localhost:9000` and the frontend on `localhost:3000`.

Client files are automatically compiled on change, and the web browser will

reload them automatically thanks to hot module replacement.

### Client and server side

The API will listen on `localhost:9000` and the frontend on `localhost:3000`.

File changes are automatically recompiled, injected in the web browser (no need to refresh manually)

and the web server is automatically restarted.

```

$ npm run dev

```

### Testing the federation of PeerTube servers

Create a PostgreSQL user \*\*with the same name as your username\*\* in order to avoid using the \*postgres\* user.

Then, we can create the databases (if they don't already exist):

```

$ sudo -u postgres createuser you\_username --createdb

$ createdb -O peertube peertube\_test{1,2,3}

```

Build the application and flush the old tests data:

```

$ npm run build -- --light

$ npm run clean:server:test

```

This will run 3 nodes:

```

$ npm run play

```

Then you will get access to the three nodes at `http://localhost:900{1,2,3}`

with the `root` as username and `test{1,2,3}` for the password.

Instance configurations are in `config/test-{1,2,3}.yaml`.

### Unit tests

Create a PostgreSQL user \*\*with the same name as your username\*\* in order to avoid using the \*postgres\* user.

Then, we can create the databases (if they don't already exist):

```

$ sudo -u postgres createuser you\_username --createdb --superuser

$ npm run clean:server:test

```

Build the application and run the unit/integration tests:

```

$ npm run build -- --light

$ npm test

```

If you just want to run 1 test:

```

$ npm run mocha -- --exit -r ts-node/register -r tsconfig-paths/register --bail server/tests/api/index.ts

```

Instance configurations are in `config/test-{1,2,3,4,5,6}.yaml`.

Note that only instance 2 has transcoding enabled.

### Emails

To test emails with PeerTube:

\* Run [mailslurper](http://mailslurper.com/)

\* Run PeerTube using mailslurper SMTP port: `NODE\_CONFIG='{ "smtp": { "hostname": "localhost", "port": 2500, "tls": false } }' NODE\_ENV=test npm start`

## Plugins & Themes

See the dedicated documentation: https://docs.joinpeertube.org/#/contribute-plugins