
TQ Website Documentation

Release 1.

Renato Bellotti, Kadir Akin

Apr 28, 2020

Contents:

1	Howtos for Non-Programmers	3
1.1	Enrol in a course	3
1.2	I have problems enrolling	5
1.3	Export (payment) information of the teachers	5
1.4	Add a new partner association	7
2	Server setup, configuration and maintenance	13
2.1	Introduction and general architecture	13
2.2	Setup basic tools	13
2.3	Setup local project folder	14
2.4	Let docker install all development dependencies	16
2.5	Load test data into database	16
2.6	Create super user	16
2.7	Test the website locally	16
2.8	Setup on a Mac	17
3	Contributing & Apply code changes	19
3.1	Troubleshooting	19
4	Tips when working on the production server [Admin only]:	21
5	Infrastructure & Architecture	23
5.1	Python packages	23
5.2	Non-Python Libraries & Credits	24
5.3	Translations	24
5.4	Enrolment in a course	25
5.5	Payment system	27
5.6	Unit Tests	28
5.7	Generate Sphinx Documentation	29
5.8	Altering the database: Migrations	29
6	Common Problems	31
6.1	One of the migrations fails	31
6.2	Solving migration errors	33
6.3	CKEditor does not display an option to add links and/or images:	34
6.4	Error during docker-compose build:	34
6.5	I get “INTERNAL SERVER ERROR”, but neither Sentry nor the log files notice it!	35

6.6	Certificate renewal	35
6.7	Enrollment error: No account for email (though the account exists)	36
7	Courses	37
7.1	Models	38
7.2	Services	39
8	Indices and tables	43
	Python Module Index	45
	Index	47

Howtos for Non-Programmers

1.1 Enrol in a course

- If you do not have an account on the website: Create one.
- If you already have an account, but forgot your password, click on “Forgot Password?” in the login form and follow the instructions.
- Log in.
- Go to courses.

Let's dance!

The TQ welcomes you to our website! We are here to let you move your body to many kinds of music. To make this as true as possible for all students all over Zurich, we give each year a big effort to organize awesome dance courses, workshops, trainings and our famous dance events.

[Course enrolment for FS 2018 Q1 is now possible!](#)

Keep in touch

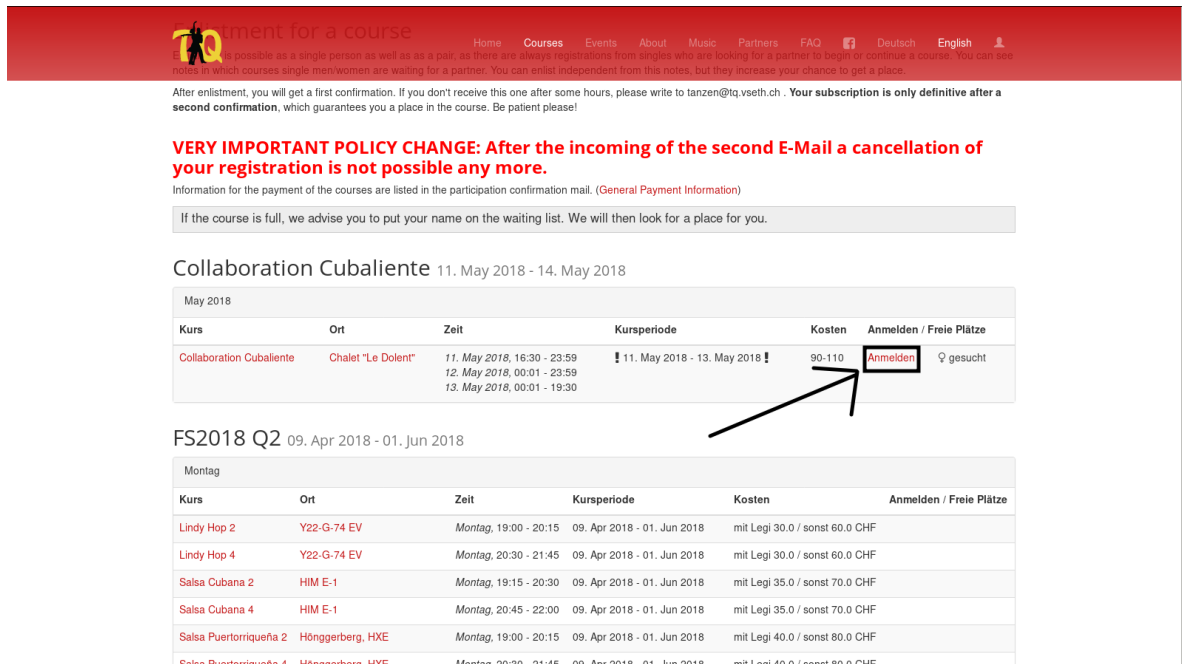
Never miss when course subscriptions open or our awesome events take place: [LIKE US ON FACEBOOK](#) or [integrate our events into your calendar](#). If you prefer emails, we also have a traditional newsletter. Subscriptions / Unsubscriptions can be made [here](#). To unsubscribe, please first [log in](#) and uncheck the box 'Newsletter' on your profile

<https://tanzquotient.org/en/courses/>

Upcoming Events

Freies Tanzen April 9, 2018 20:30-23:15, Zentrum, GEP/Alumni-Pavillon No entry fee
Freies Tanzen April 23, 2018 20:30-23:15, Zentrum, GEP/Alumni-Pavillon No entry fee

- Scroll down and click on the “Anmelden” link of the course you want to enrol in. In the example picture, this is the “Collaboration Cubaliente”.



Enrollment for a course

Home Courses Events About Music Partners FAQ Deutsch English

After enrollment, you will get a first confirmation. If you don't receive this one after some hours, please write to tanzen@tq.vsteth.ch. Your subscription is only definitive after a second confirmation, which guarantees you a place in the course. Be patient please!

VERY IMPORTANT POLICY CHANGE: After the incoming of the second E-Mail a cancellation of your registration is not possible any more.

Information for the payment of the courses are listed in the participation confirmation mail. ([General Payment Information](#))

If the course is full, we advise you to put your name on the waiting list. We will then look for a place for you.

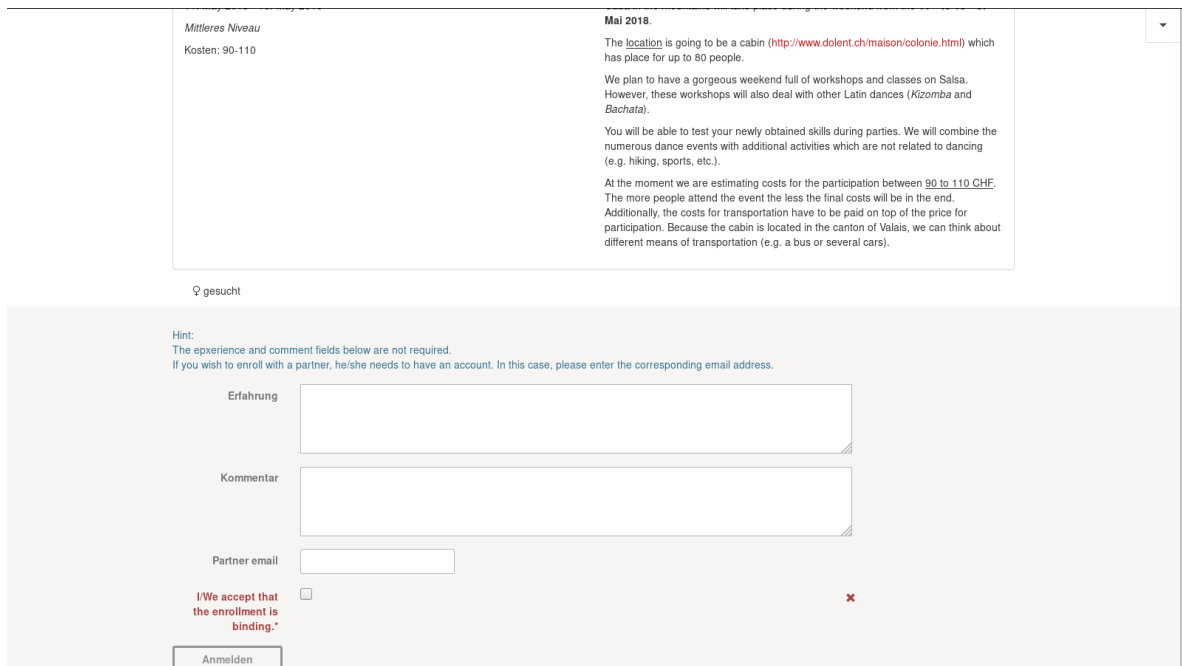
Collaboration Cubaliente 11. May 2018 - 14. May 2018

Kurs	Ort	Zeit	Kursperiode	Kosten	Anmelden / Freie Plätze
Collaboration Cubaliente	Chalet "Le Dolent"	11. May 2018, 16:30 - 23:59 12. May 2018, 00:01 - 23:59 13. May 2018, 00:01 - 19:30	11. May 2018 - 13. May 2018	90-110	Anmelden ♀ gesucht

FS2018 Q2 09. Apr 2018 - 01. Jun 2018

Kurs	Ort	Zeit	Kursperiode	Kosten	Anmelden / Freie Plätze
Lindy Hop 2	Y22-G-74 EV	Montag, 19:00 - 20:15	09. Apr 2018 - 01. Jun 2018	mit Legi 30.0 / sonst 60.0 CHF	
Lindy Hop 4	Y22-G-74 EV	Montag, 20:30 - 21:45	09. Apr 2018 - 01. Jun 2018	mit Legi 30.0 / sonst 60.0 CHF	
Salsa Cubana 2	HIM E-1	Montag, 19:15 - 20:30	09. Apr 2018 - 01. Jun 2018	mit Legi 35.0 / sonst 70.0 CHF	
Salsa Cubana 4	HIM E-1	Montag, 20:45 - 22:00	09. Apr 2018 - 01. Jun 2018	mit Legi 35.0 / sonst 70.0 CHF	
Salsa Puertorriqueña 2	Hönggerberg, HXE	Montag, 19:00 - 20:15	09. Apr 2018 - 01. Jun 2018	mit Legi 40.0 / sonst 80.0 CHF	
Salsa Puertorriqueña 4	Hönggerberg, HXE	Montag, 20:30 - 21:45	09. Apr 2018 - 01. Jun 2018	mit Legi 40.0 / sonst 80.0 CHF	

- You are now at the enrolment page. It should look something like this:



Mittleres Niveau
Kosten: 90-110

Mai 2018.
The location is going to be a cabin (<http://www.dolent.ch/maison/colonie.html>) which has place for up to 80 people.
We plan to have a gorgeous weekend full of workshops and classes on Salsa. However, these workshops will also deal with other Latin dances (Kizomba and Bachata).
You will be able to test your newly obtained skills during parties. We will combine the numerous dance events with additional activities which are not related to dancing (e.g. hiking, sports, etc.).
At the moment we are estimating costs for the participation between 90 to 110 CHF. The more people attend the event the less the final costs will be in the end. Additionally, the costs for transportation have to be paid on top of the price for participation. Because the cabin is located in the canton of Valais, we can think about different means of transportation (e.g. a bus or several cars).

♀ gesucht

Hint:
The experience and comment fields below are not required.
If you wish to enroll with a partner, he/she needs to have an account. In this case, please enter the corresponding email address.

Erfahrung

Kommentar

Partner email

☐ I/We accept that the enrollment is binding.

Anmelden

- If you are enrolling to a couple course, you can enrol together with your partner. For this, enter his/her email address in the field “Partner email”. Note that this must be the email address that your partner uses in his TQ account. You can also enrol alone, and we will try to find a partner for you.
- Not required: Give us some hints about your experience and/or other comments.
- Check the checkbox.

- Click on the “Anmelden” button.
- Congratulations, you have successfully enrolled! Check your email account for a confirmation email, click the link in it and wait for a second email from us where we confirm your enrolment definitely.

1.2 I have problems enrolling

When I try to enrol my partner, the system says that nobody with this email is registered. But when I try to create an account for this email address, it says that this address is already used.

- **Solution (student):**

Contact the website admin by email [informatik \[at\] tq.vseth.ch](mailto:informatik[at]tq.vseth.ch), or use the support chat <https://t.me/joinchat/AIE6nw89dgmPvObM8kcKQA>.

- **Solution (TQ responsible person):**

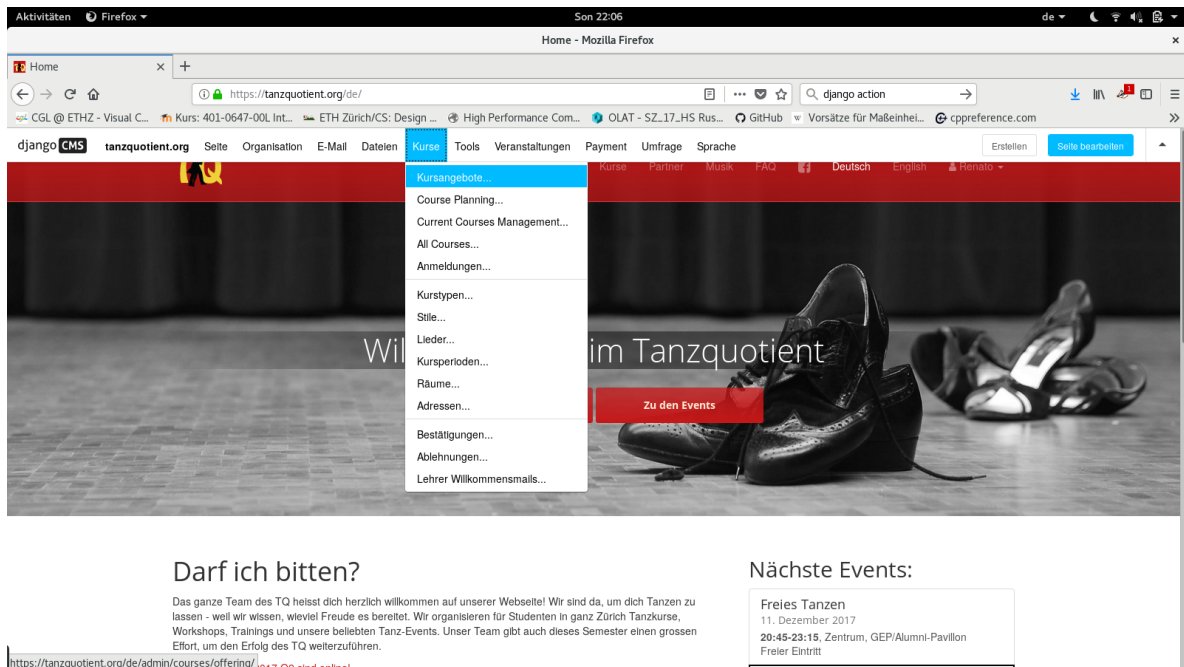
This is a multiaccount issue caused by an old version of the website. It was possible to enrol without an account, but the system internally created an account, which led to the situation that there were multiple accounts associated with the same email address. Fixed on the 1st of April 2018.

- Search for the user profiles.
- Select the one where the last login lies further back. If no “last login” time is set, this means that the last login was very long ago.
- Inactivate this profile by removing the “Active” checkbox.

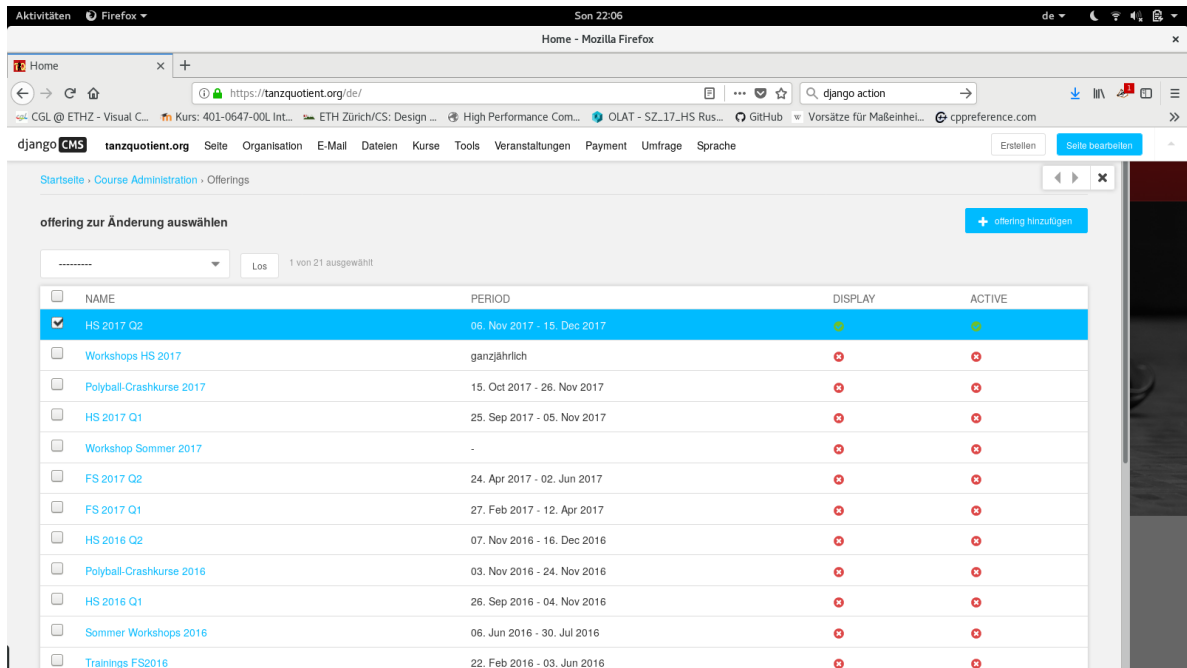
1.3 Export (payment) information of the teachers

The following steps refer to the buttons in the admin panel. If you cannot see them, you probably don’t have enough rights. Please contact the administrator in this case.

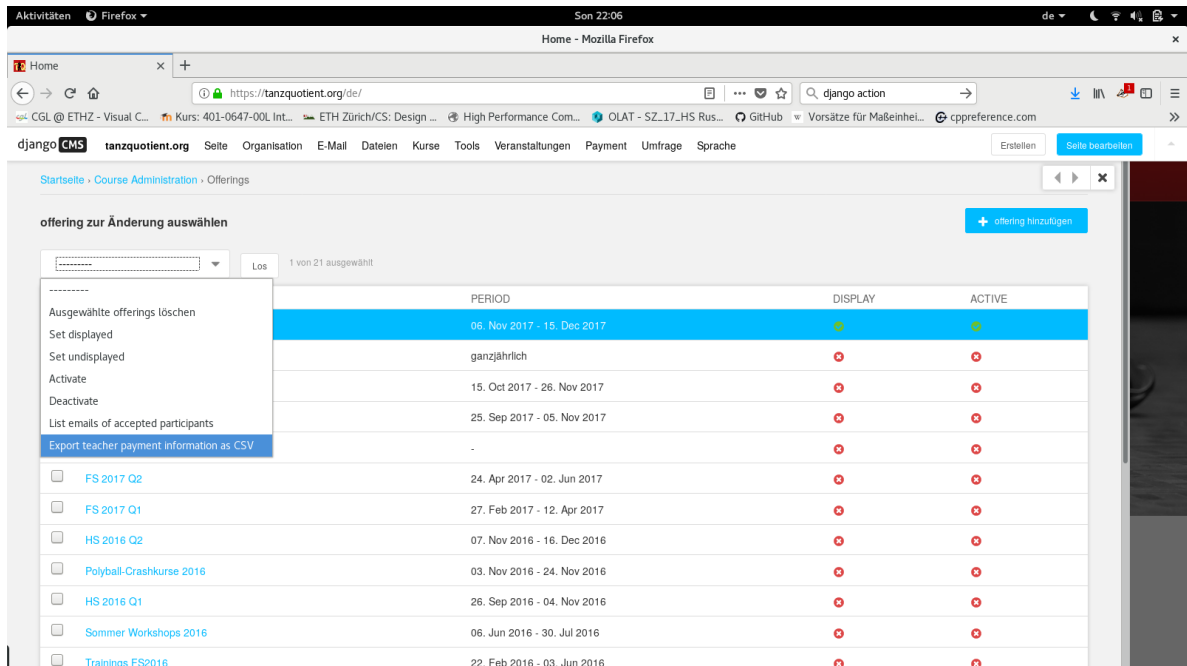
- Go to Courses → Course offerings.



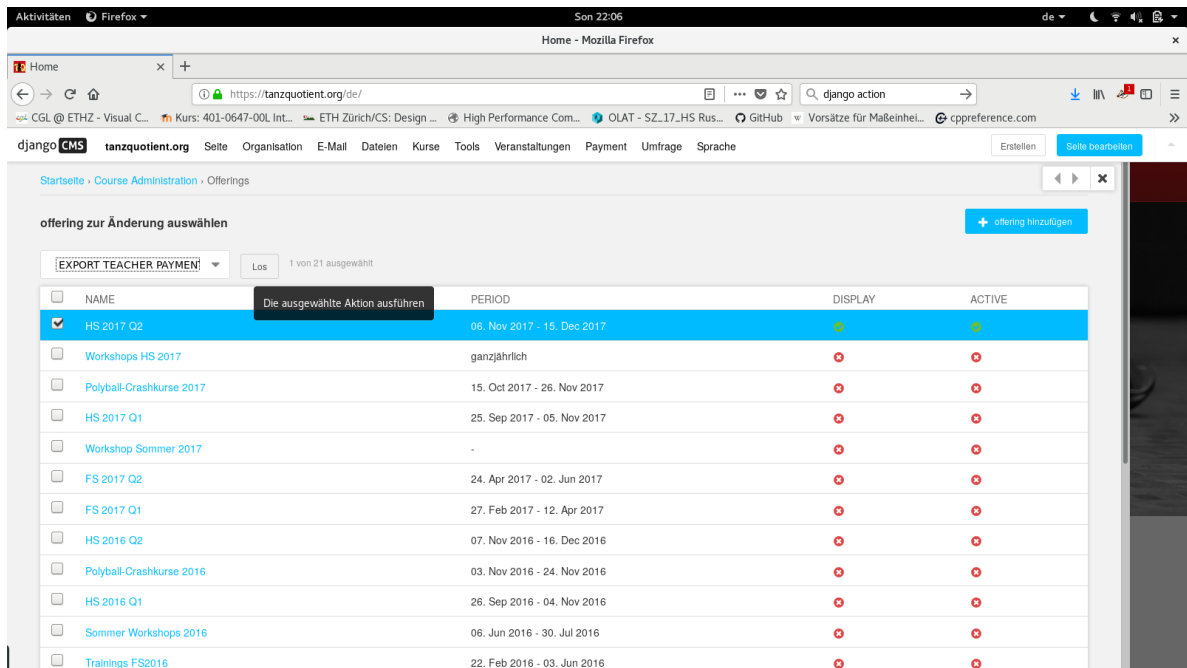
- Select the course period during you're interested in.



- Select “Export teacher payment information as CSV” in the action selection widget.



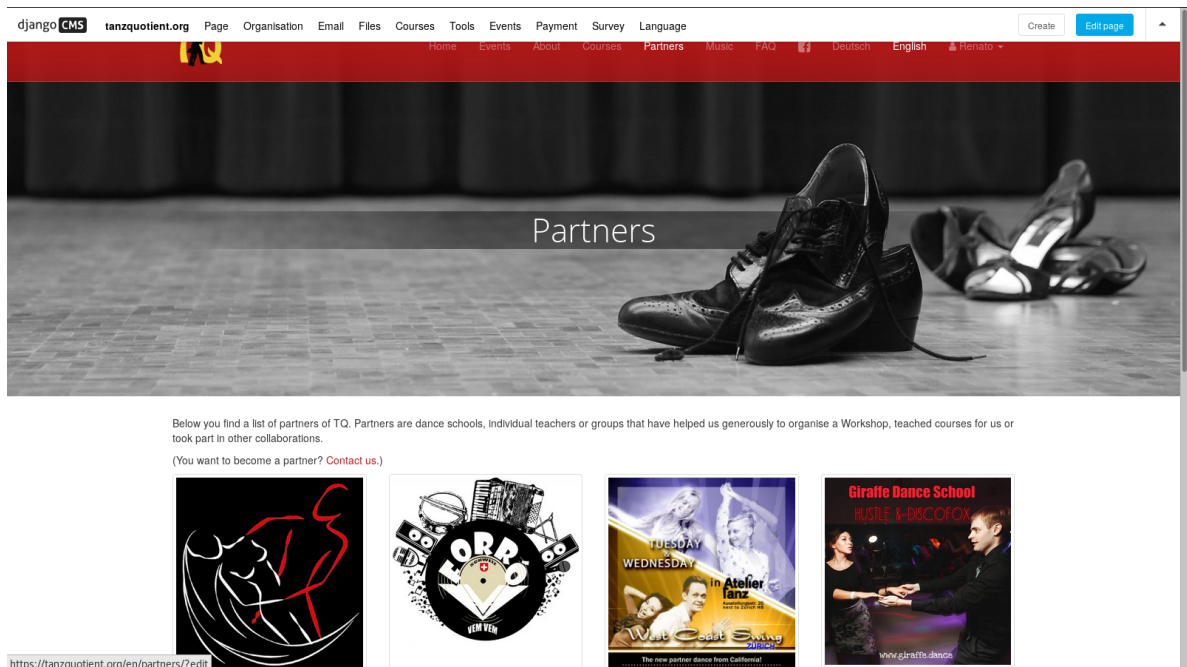
- Click “OK” and download the CSV (comma separated values) file to your computer. You can open it with your favourite spreadsheet processing program (e. g. Microsoft Excel or LibreOffice Calc).



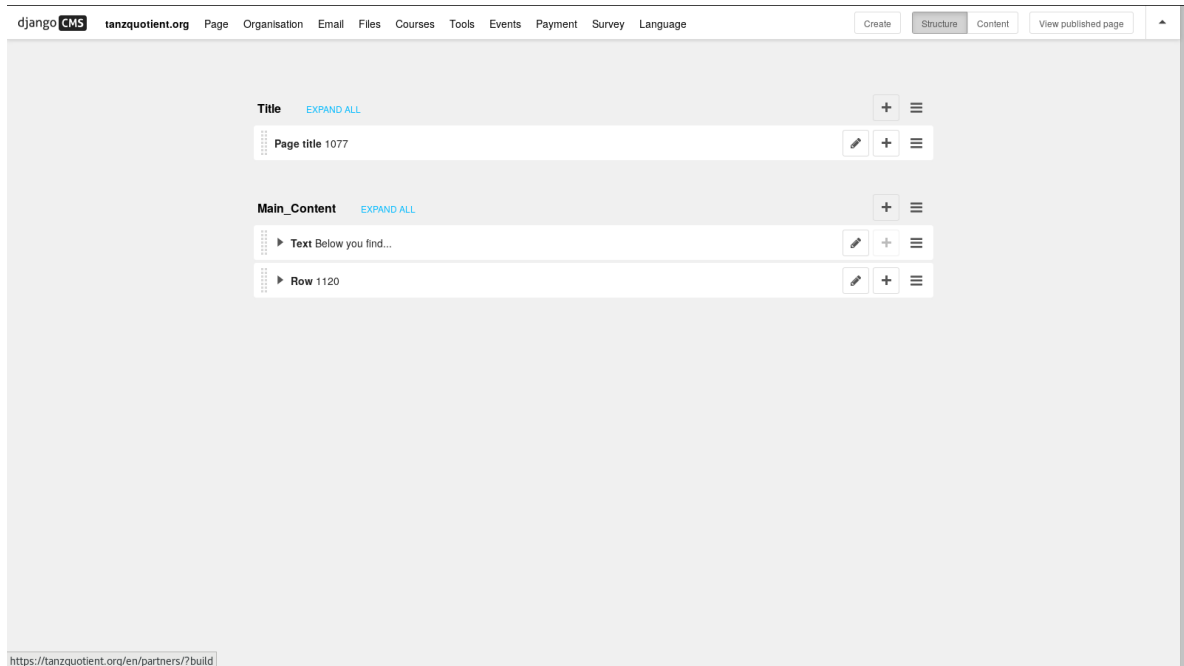
1.4 Add a new partner association

Log in to the website and go to our [partner](#) page. You might have to perform the steps below for all languages.

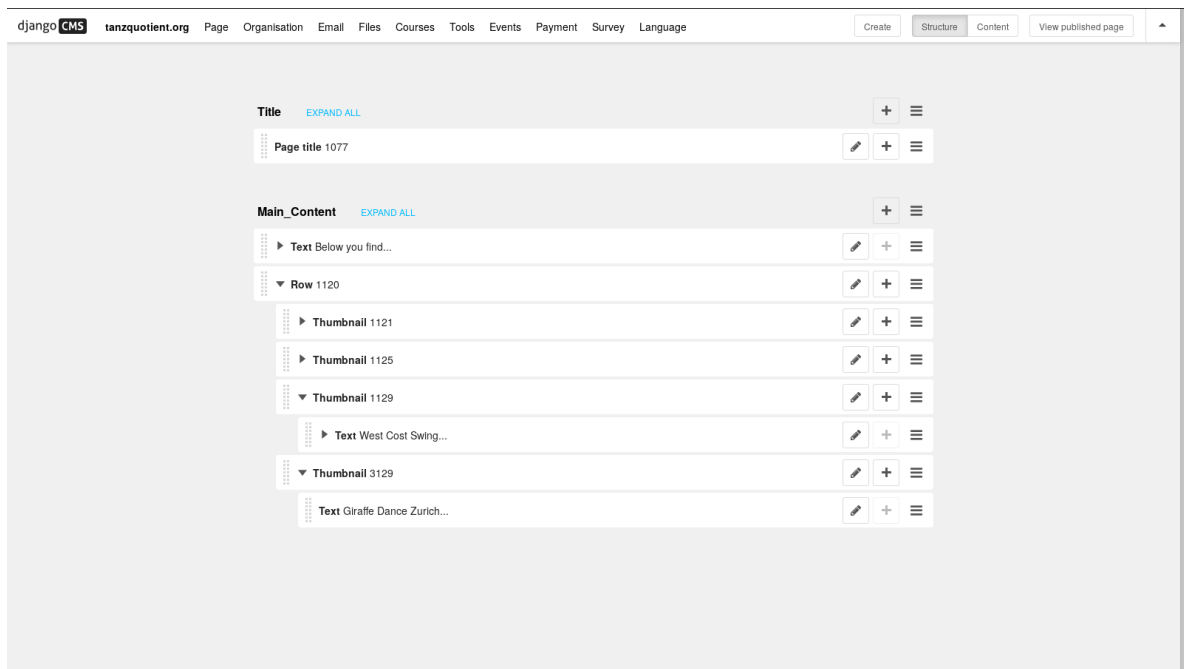
- Display the admin toolbar (the white toolbar).
- Click the “Edit page” button in the top right corner.



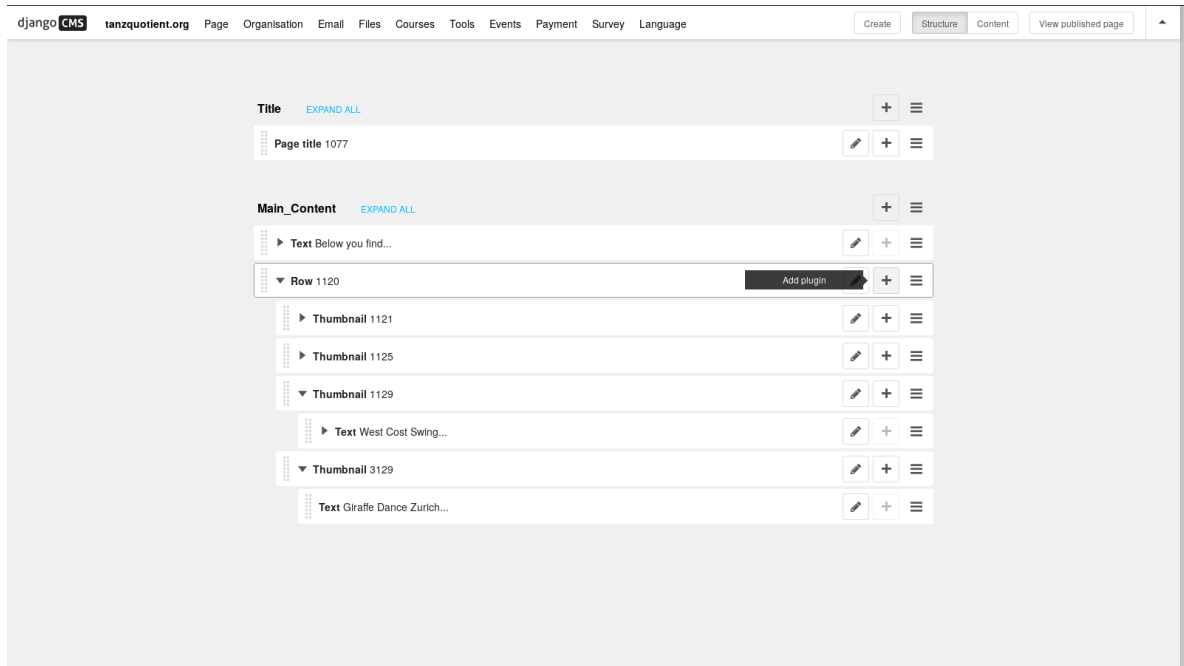
- Click on “Structure” in the top right corner.



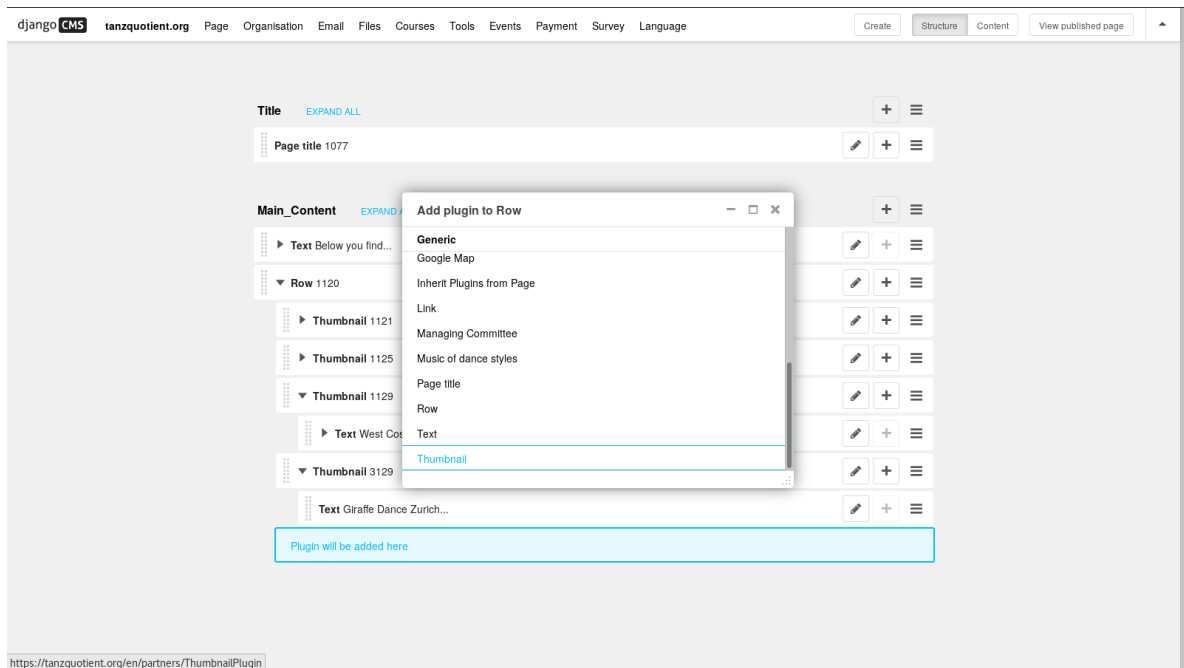
- Expand the row by clicking on it.



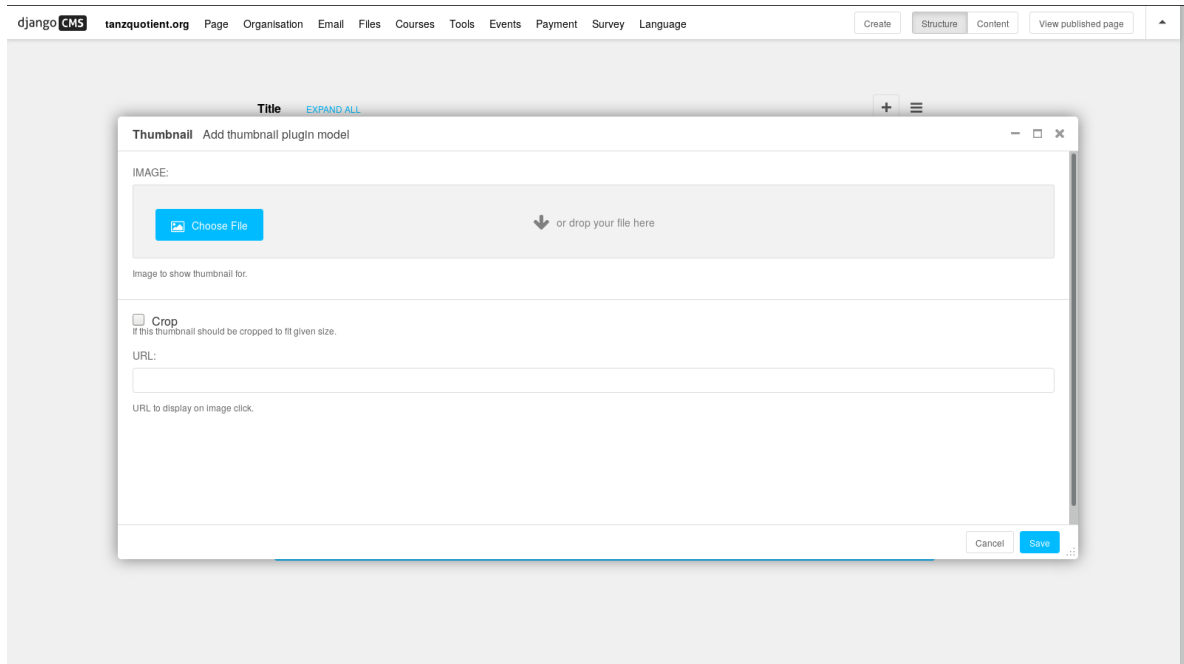
- Click the “+” button in the row entry to add a new thumbnail (i. e. a clickable image).



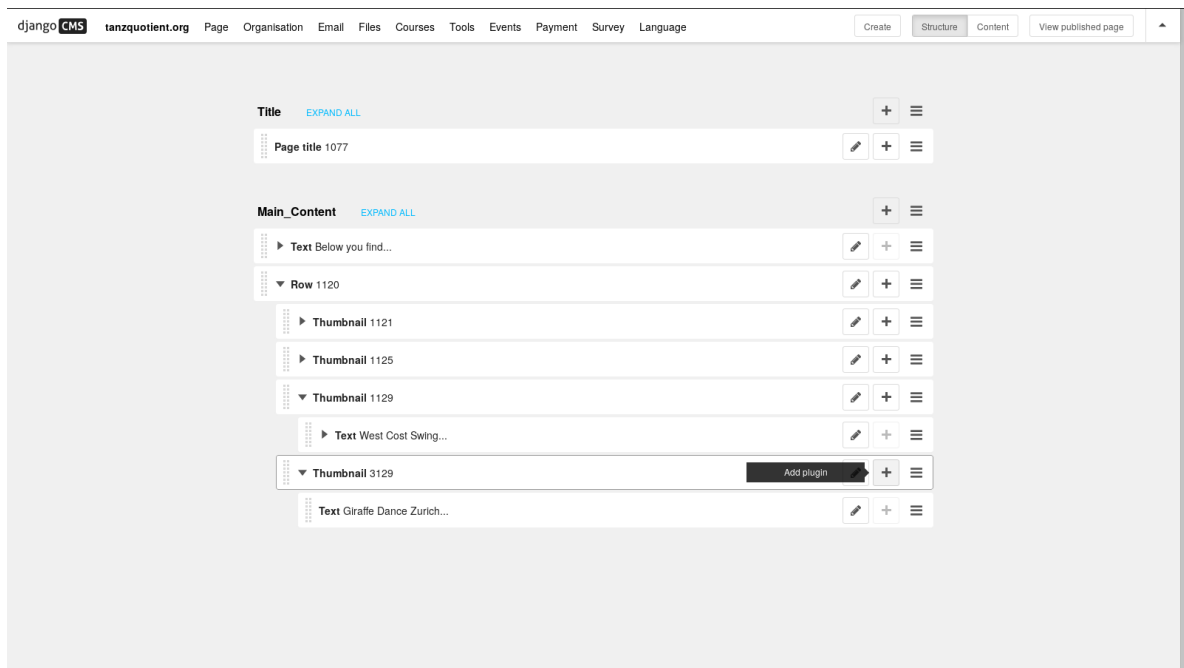
- Select “Thumbnail” in the list.

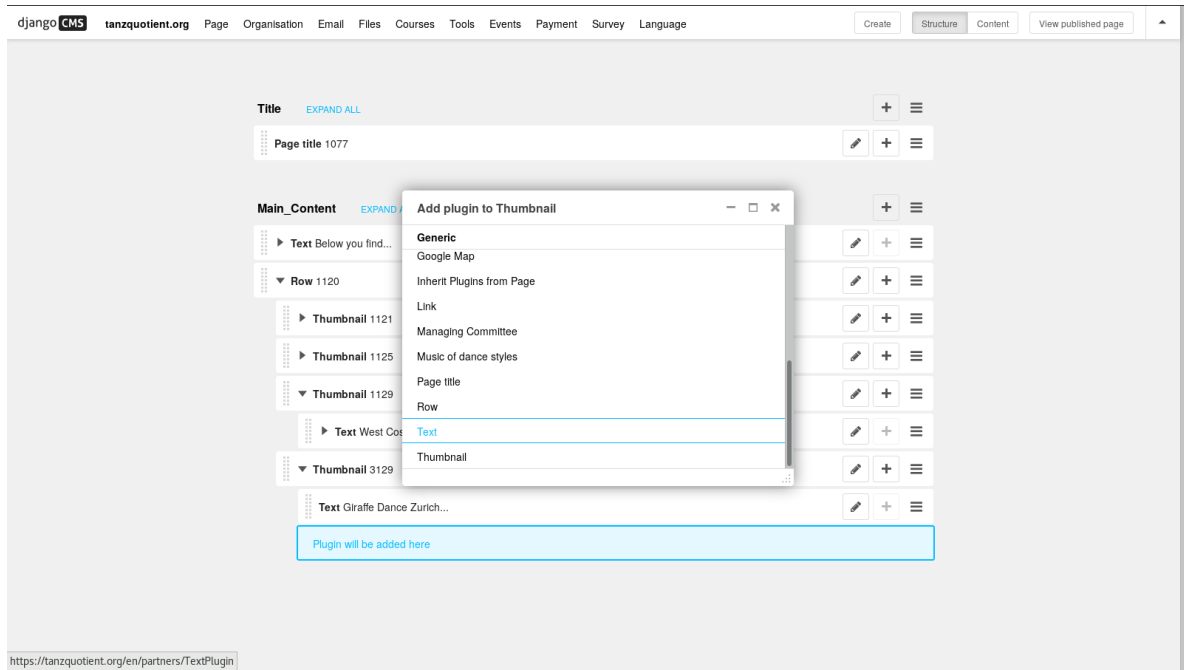


- Now you can select the image to display and a URL. This is the URL that will be opened if the user clicks on the image. When you’re done, click on the “Save” button in the bottom right corner.



- Finally, you can add the description text below the image. To achieve this, do the same steps as before, but add the test box as a subelement of the thumbnail you've just created (NOT of the row)!





Server setup, configuration and maintenance

2.1 Introduction and general architecture

This file contains setup with docker. Many of the steps you may expect to setup a webstack are automatized. You can still look them up in the [Dockerfile](#).

The setup instructions are divided into:

- common steps for all setups
- steps for local development
- steps only necessary on the production server

The following picture sketches the setup. Some notes:

- Deployment is done by logging in to the server via SSH and pulling the (production-)branch from the repository
- The python environment is configured the same locally and on the server.
- On the server a faster and more secure web server (nginx) is used instead of the Django development server
- Some secrets (config file with login information, secret keys) are not synchronized via the repository. This secrets also differ from the one used on development machines.

2.2 Setup basic tools

You must use [git](#) for code management and [Docker](#) for setup automation.

We use a standard *Debian 8* on the server. On development machines, any operating system can be used in principle (we know that many Linux und Mac OS versions works). The instructions here are compiled for a **Debian/Ubuntu** installation.

First update your system:

```
sudo apt-get update
sudo apt-get upgrade
```

NOTE: Package names can deviate depending on your Linux distribution.

```
sudo apt-get install git mysql-client
```

We need **Docker Community Edition (CE)** and **docker-compose**. With prerequisites satisfied, it boils down to

```
sudo apt-get install docker-ce
sudo curl -L "https://github.com/docker/compose/releases/download/1.11.2/docker-
→compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
```

Start the docker daemon:

```
sudo service docker start
```

2.2.1 Setup basic tools with Fedora

The above instructions are basically the same with Fedora, except that you have to use `dnf` instead of `apt`. Altogether, the following instructions install everything one needs:

```
sudo dnf update
sudo dnf install git mysql docker docker-compose
```

2.2.2 IDE

Install a local IDE. We highly recommend to use **PyCharm**. The full version has Django support and is free for educational purposes.

To get the educational version, go to **PyCharm Student** and fill out the form using an official `@ethz.ch` mail address. After completing, you will receive an e-mail from JetBrains with a link to confirm your request. If all works well, you will receive another e-mail with further instructions on how to set up a JetBrains user account.

Finally you can download PyCharm Professional Edition, extract it and place it somewhere you want. There is no installation required. To start the program run `<YourPyCharmFolder>/bin/pycharm.sh`.

Activation is easiest if you download the licence-file from your JetBrains account-page. When asked for activation, simply drag&drop the file into the activation-key textbox.

2.3 Setup local project folder

2.3.1 Pull files with git

Create a folder on your machine where you want to store the local copy of the repository. This could e.g. be in your home directory.

```
mkdir ~/Projects/<project home>
```

Now `cd` into the newly created folder

```
cd ~/Projects/<project home>
```

and execute the following commands to tell git that your local copy of the repository now lives in this folder.

```
git init
git remote add origin https://github.com/gitsimon/tq_website.git
git fetch
git checkout -t origin/master
```

If you want to work on your own branch, create it and check it out

```
git branch dev-<your name>
git checkout dev-<your name>
```

You can push it to the server and setup push/pull by

```
git push -u origin <branch>
```

It's a good idea to rebase your branch on the master from time to time. While your branch is checked out, run:

```
git rebase master
```

Git is a powerful tool. Have a look at the [official documentation](#), especially on [branching](#).

2.3.2 Make helper scripts executable

Since file permissions are not synchronized with git, you have to make the helper scripts executable:

```
sudo chmod +x scripts/*
```

2.3.3 Initial Configuration

We have to create 2 files, that are not under version control, manually:

Create the maintenance file `<project home>/maintenance.conf`. You can use the provided template file and copy it with

```
cp configurations/maintenance-template.conf maintenance.conf
```

(Whenever doing maintenance on a live server, switch the flag in this file to 1 (and back again), and restart docker-compose to make nginx reload the config and display a maintenance message)

Create the *secret* environment file `<project home>/ .env`. You can use the provided template file and copy it with

```
cp configurations/.env-template .env
```

This files are not under version control because it contains some secrets and machine dependent configurations and secrets.

Attention: The configured mail account is used to - depending on the action - send huge amounts of auto-generated mails. Leave the mail settings empty (as it is in the template) or configure a test mail server before starting a production-like docker configuration (which will actually send out mails!).

2.4 Let docker install all development dependencies

Note: In the current setup the `docker-compose.yml` is customized via environment variables. Due to escaping issues, this works only with the `zsh` shell which may not be standard on some unix based systems.

On development machine:

Run in the `<project home>` directory:

```
docker-compose build
```

It will fetch all required dependencies and install it for you.

Note: This can take some minutes

Note: If you encounter a problem because some ports are already in use, you can choose your preferred development ports in the `.env`-file.

In production environment (or to setup a production-like stack on development machine):

```
docker-compose -f docker-compose-production.yml build
```

Simulated production environment (to setup a production-like stack on development machine):

```
docker-compose -f docker-compose-production-no_ssl.yml build
```

2.5 Load test data into database

Get in touch with admin to get a backup of live database (with removed personal data). The backup can then be applied to the database with (while docker is running the containers)

```
mysql -h 127.0.0.1 --port=3309 -u root -proot -t tq_website < database_dump.sql
```

2.6 Create super user

Create a superuser with your favorite name and password:

```
./scripts/create_superuser.sh
```

Note: This are the credentials to login anywhere on the frontend/backend.

Note: Even if the loaded database dump contains a user representing you you have to repeat that step since the dump has different salted passwords, so your password will be considered invalid.

2.7 Test the website locally

Whenever working on the project, run the following command in the `<project home>` directory primarily:

```
docker-compose up --build
```

While this command is running you should be able to view the local, full-stack website at this addresses:

- `localhost:8000` or `127.0.0.1:8000`

- `localhost:8001` or `127.0.0.1:8001` (if you started with `-f docker-compose-production-no_ssl.yml`)

2.8 Setup on a Mac

If you have a Mac and prefer to use GUIs, the following tutorial gives an alternative way over the command line setup:

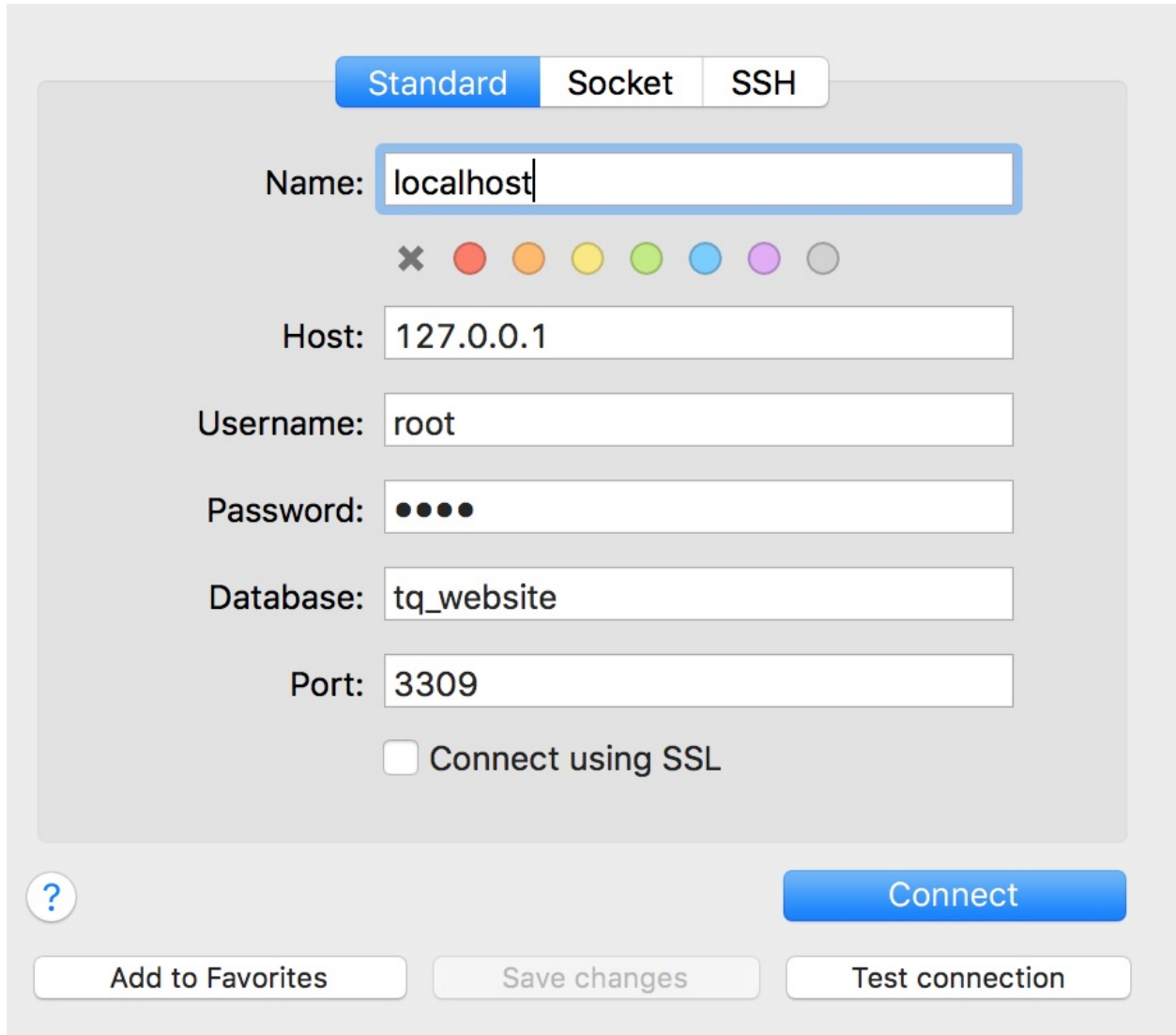
2.8.1 Download the necessary Software:

- [Docker for Mac](#)
- [Download PyCharm](#) - As a student, you can get a professional license for free
- [Sequel Pro](#)

2.8.2 Debugger

1. Open the pycharm settings
2. In the *Settings / Preferences* dialog go to *Build, Execution, Deployment* and then *Docker*
3. Click on + and then *Apply*
4. Go to *Project: tq_website*, then *Project Interpreter* and in the *Drop Down Menu* choose *Show All*
5. Click on the + and then *Add Remote*
6. In the Pop-Up, choose *Docker-Compose*
 - a) Under *Server*, *Docker* should show up (only if you did steps 2. and 3. right)
 - b) *Configuration Files* should show your `docker-compose.yml` file (if not, are you in the `tq_website` project?)
 - c) *Service* should have *Django*
7. Apply and close, go back to the main editor. In the right upper corner click on the combo box reading *tq_website*, and then *Edit Configurations*
8. In the pop-up, with the `tq_website` configuration selected, choose *Host* to be *0.0.0.0*

2.8.3 Sequel Pro



Standard Socket SSH

Name: localhost

Host: 127.0.0.1

Username: root

Password: ●●●●

Database: tq_website

Port: 3309

☐ Connect using SSL

?

Connect

Add to Favorites Save changes Test connection

Password: root

2.8.4 PyCharm

1. On the main menu, choose File | Open.
2. Select the directory that contains the desired source code (pulled from git repository).
3. Click ok.

Contributing & Apply code changes

Note: This method of applying code changes does not destroy your test data, but gradually migrates the database.

Pull the changes from the correct branch (here the master):

```
git pull
```

If you are working on your own developer branch, pull the changes of the master branch explicitly: .. code-block:: bash

```
git pull origin master
```

It's a good idea to rebase your branch on the master from time to time. While your branch is checked out, run: .. code-block:: bash

```
git rebase master
```

Apply migrations to your test database by entering in a shell

```
./scripts/migrate.sh
```

Then cancel (Ctrl + C) and restart the `docker-compose` command to ensure changes in configuration are reflected. Docker will detect configuration changes with that option and rebuild containers if necessary.

```
docker-compose up --build
```

Note: If desired, see the above how to reset the database and reload a database dump. Not however, that the migrate command still has to be run because the dump can be a little outdated compared to the newest code.

3.1 Troubleshooting

3.1.1 What often helps

Docker is complicate to predict. Some config files are not loaded ad-hoc. Whenever there is a problem, try to restart the containers all together with

```
docker-compose restart
```

or

```
docker-compose stop  
docker-compose up
```

(with the second option you will be directly attached to the containers and you see the output)

3.1.2 Page reload

Some assets files are cached by the browser: ensure that you make a full page reload (`Ctrl + F5`) or you even delete all session cookies.

Tips when working on the production server [Admin only]:

- **Always perform a backup before you change something on the server!!!** The scripts responsible for this can be found in the parent directory of the source code directory on the server.
 - `mysql-backup.sh`: Performs a full dump of all databases. Then it encrypts the dump, removes the unencrypted directory and uploads the encrypted backup to the VSETH cloud.
 - `mysql-backup-tq_website.sh`: Performs a backup of the `tq_website` database. **The backup is neither encrypted nor uploaded to the cloud!**
- **Never ever use `docker-compose down` !!!** This will “delete” the database! Use `docker-compose stop` and `docker-compose start` or `docker-compose restart` instead.

Infrastructure & Architecture

5.1 Python packages

This page contains information about all the Python packages that are used somewhere in the TQ website. Please keep the information up to date and extend it as new features are added. This makes it easier for newcomers to understand what is going on in the project.

Keep in mind that if the person who implemented a feature is not available anymore, this page is the only reliable source of information about the dependencies.

Package name	version (date of last check for updates)	Purpose of the package
django-ical	1.4 (26.08.2017)	iCal calendar synchronisation
pylint	1.7.2 (26.08.2017)	Rate quality of code and check PEP8 compliance
graphviz	0.8 (26.08.2017)	Write UML diagrams in good graphics formats, e
python-dotenv	0.8.2 (04.06.2018)	Needed for ReadTheDocs.org
sphinx-autodocapi	0.7 (04.09.2018)	Generate summary of entire modules
celery	4.2.0 (11.06.2018)	task queue
django-absolute	0.3 (11.06.2018)	“provides context processors and template tags to
django-analytical	2.4.0 (11.06.2018)	provides an easy way to use different analytics ser
django-appconf	1.0.2 (11.06.2018)	“A helper class for handling configuration default
django-celery-email	2.0.0 (11.06.2018)	“A Django email backend that uses a Celery queu
django-celery-beat	1.1.1 (11.06.2018)	extends Celery so that periodic tasks can be mana
django-celery-results	1.0.1 (11.06.2018)	extends Celery so that task results are stored in D
django-classy-tags	0.8.0 (11.06.2018)	extends Django’s template system
django-filer	1.3.2 (11.06.2018)	file upload manager for Django
django-filter	1.1.0 (11.06.2018)	“allows users to filter down a queryset based on a
django-guardian	1.4.9 (11.06.2018)	per-object permissions for Django (see paper)
django_polymorphic	2.0.2 (20.08.2018)	“Django-polymorphic simplifies using inherited m
django-formtools	2.1 (20.08.2018)	“Django’s “formtools” is a set of high-level abstra
djangoCMS-googlemap	1.1.1 (20.08.2018)	plugins for Django CMS to embed Google Maps
djangoCMS-admin-style	1.2.8 (20.08.2018)	pretty stylesheets for the Django CMS admin inte

Package name	version (date of last check for updates)	Purpose of the package
djangoCMS-link	2.1.2 (20.08.2018)	allows to embed links on the website (?)
django-bootstrap3	10.0.1 (20.08.2018)	“Write Django as usual, and let django-bootstrap3
cmsplugin-filer	DEPRECATED!!!	
django	1.11 (30.08.2018)	The web framework → most important package
django-parler	1.9.2 (30.08.2018)	Translations
django-parler-rest	1.4.2 (30.08.2018)	Translations for django-rest-framework
django-mptt	0.9.1 (30.08.2018) NEEDS UPDATE!	“Utilities for implementing a modified pre-order t
django-post_office	3.1.0 (30.08.2018)	Sends emails asynchronously (also from admin in
mysqlclient	1.3.13 (30.08.2018)	“MySQL database connector for Python (with Py
django-countries	5.3.2 (04.09.2018)	“A Django application that provides country choi
django-sekizai	0.10.0 (05.09.2018)	Makes it possible to load all JS & CSS files at the
base36	0.1.1 (05.09.2018)	represents strings in base36 (needed for our Uniq
PyPDF2	1.26.0 (05.09.2018) NOT NECESSARY ANYMORE!	needed for PDF export of vouchers (CURRENTL

5.2 Non-Python Libraries & Credits

The TQ web site uses the following libraries (only non-Python libs; for Python libs see *Python dependencies*):

- daleharvey/pacman: To customize the 404 error page.

5.3 Translations

5.3.1 Related documentation

Use the `trans` tag in HTML sites for django: <https://docs.djangoproject.com/en/1.10/topics/i18n/translation/#trans-template-tag>

5.3.2 What to do

Essentially, you have to run two scripts:

```
scripts/makemessages.sh
scripts/compilemessages.sh
```

5.3.3 Mac: install gettext

1. `brew install gettext` in Terminal and then
2. `brew link gettext --force` to make it work ;)

5.3.4 Manual way (deprecated for our project)

Run `django-admin makemessages -a` to create the `.po` files. *Note:* Works only with `gettext` installed, see below for installation

Then after you’ve updated the German text in the `.po` file you can run

```
django-admin compilemessages
```

to compile the language files.

Check in the language files to git as well, we don't compile a second time on the live server.

Please use English as the default language and then add German translations.

Generate new binary files (.mo) after changing ASCII file (.po)

```
docker-compose run --rm django python3 manage.py compilemessage
```

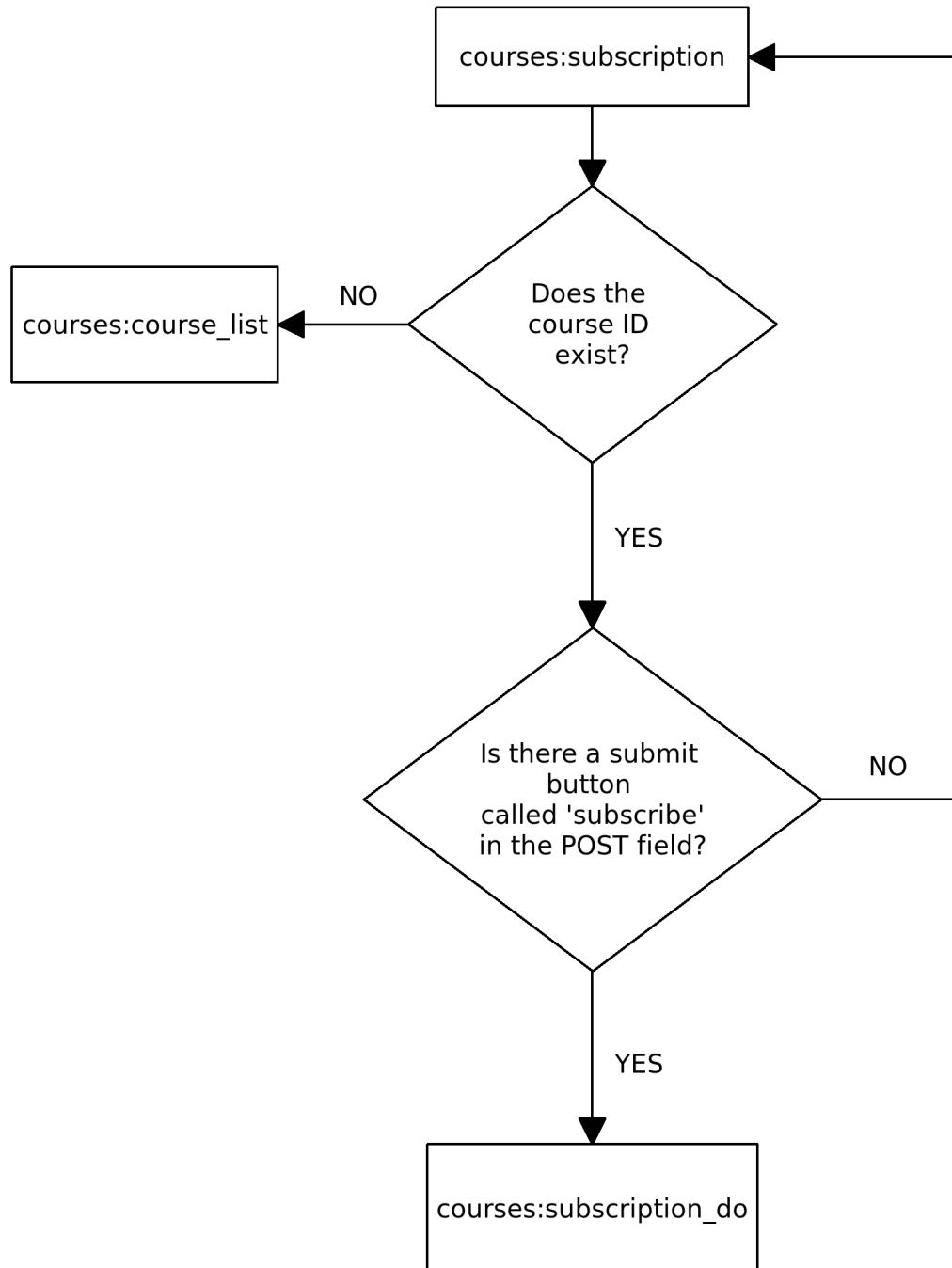
5.4 Enrolment in a course

Enrolment in a course is not a difficult process. But since it is one of the core functionalities of this project, there needs to be documentation.

5.4.1 High level procedure

The following flowchart gives an overview of the involved views and how they interact with each other:

Enrolment process: Views



5.4.2 Which files are involved?

- `courses/views.py`: The views that handle the enrolment and present success/failure to the user

- `courses/services.py`: The function `subscribe` is responsible for the actual enrolment

5.5 Payment system

This page gives an overview of how the payments are processed.

5.5.1 How is the processing triggered?

We use [Celery](#) to download and process payment files in regular time intervals. The Celery tasks are defined in `tq_website/tasks.py`. The following Celery tasks are responsible for the payment system: - Task to download the new transactions from PostFinance. - Task to parse the downloaded transactions and insert them into the database. - Task to process payments.

5.5.2 How and where are the transactions stored?

- The information about the transactions are stored on an SFTP server maintained by PostFinance.
- The data format is called [ISO 20022](#). It is XML based. The pages 37-43 describe the XML attributes: Use this e. g. to see what information is stored and how it can be accessed.
- The downloaded files are saved in `BASE_DIR/FDS_DATA_PATH`, where `BASE_DIR` and `FDS_DATA_PATH` are stored inside `tq_website/settings.py`. (`BASE_DIR` is the project root directory, `FDS_DATA_PATH = 'fds_data'`)
- Processed files are marked by appending `.processed` to the filename.

5.5.3 What does “process payments” mean?

When a payment is said to be “processed”, it means that the system tries to relate (“match”) a payment to a course subscription. If the subscription cannot be successfully matched and/or the paid amount is not correct, human intervention is needed. This is indicated by setting the payment status to `MANUAL`.

5.5.4 What does it mean if a payment is marked as `MANUAL`?

A payment can be marked for manual processing if in one of the following cases:

- The payment cannot be related (“matched”) to a course subscription.
- The paid amount is not enough to pay the subscription.
- The paid amount is higher than the subscription cost.

5.5.5 High level procedure

First, the payments are fetched and saved. Involved files:

- The payments to our bank account are fetched from an SFTP server from PostFinance.
- The XML files are parsed and for each transaction, a `Payment` object is inserted into the database. (The `Payments` have state `NEW` at this point.) Afterwards, the XML file is marked as processed.

- The following operations are applied to new payments: Debit transactions (money that goes out of our account) are marked as `IRRELEVANT`. Process the credit transactions (money comes into our account) in the following way: If something goes wrong while processing a payment, set the payment state to `MANUAL`. If the payment can be successfully matched to a course subscription, a `SubscriptionPayment` is created and stored in the database. Then, the payment status is changed: If the paid amount is equal to the amount that is to be paid, the state of the `Payment` object is set to `MATCHED`. If the paid amount is not enough or too much, the state of the `Payment` object is set to `MANUAL`.

Finally, all `MATCHED` payments are marked as paid and `PROCESSED`.

- In all cases where a subscription is marked as paid, an “online payment successful” email is sent to the user.

5.5.6 Which files are involved?

- `tq_website/tasks.py`: contains the Celery tasks
- `payments/postfinance_connector.py`: Contains code to download XML files from a PostFinance SFTP server and extract payments from them
- `payments/payment_processor.py`: Contains code to process the new payments.
- `payments/models.py`: Contains the classes related to payments (especially `Payment` and `SubscriptionPayment`)

5.6 Unit Tests

This project uses unit tests to guarantee correctness of the code and make it easier to update dependencies without breaking anything.

5.6.1 General rule

Whenever a new feature is implemented, the corresponding unit tests should also be written. The list below should be updated.

5.6.2 How to run tests

The tests can be run by invoking `scripts/run_tests.sh` from the project root directory. The test database is generated the first time you run the script and kept in order to speed up the test execution.

This means that you have to call `scripts/delete_test_database.sh` whenever a model changes because the test database is not synchronized with the code!

5.6.3 Parts of the code for which tests exist

Currently, there are no tests.

5.6.4 Test environment

Some testing data is stored in so called fixtures. The most important data is:

Root user:

name: root

password: root

Test users:

- name: Max Mustermann
password: testtest
email: `example@host.com`
- name: Maxima Musterfrau
password: testtest
email: `example_2@host.com`

5.7 Generate Sphinx Documentation

Run the following command to generate the Sphinx documentation:

```
scripts/build_doc.sh
```

5.8 Altering the database: Migrations

5.8.1 What are migrations for?

Every time changes are made to a **model**, migrations have to be created and applied to the database in order to tell it that its structure has changed. Migrations also determine what happens to entries that already exist in the database. A migrations file contains SQL code that is executed in the database.

5.8.2 How can I create migrations?

In the root directory of our git directory, enter the following command in the terminal:

```
scripts/makemigrations.sh
```

5.8.3 How can I apply migrations?

In the root directory of our git directory, enter the following command in the terminal:

```
scripts/migrate.sh
```

5.8.4 WARNING

Before applying migrations to the production database, **always make a backup!!!**

6.1 One of the migrations fails

6.1.1 Problem

During migrations, an error message similar to the one below is printed:

```
Traceback (most recent call last):
  File "manage.py", line 10, in <module>
    execute_from_command_line(sys.argv)
  File "/usr/local/lib/python3.5/site-packages/django/core/management/__init__.py", line 367, in execute_from_command_line
    utility.execute()
  File "/usr/local/lib/python3.5/site-packages/django/core/management/__init__.py", line 359, in execute
    self.fetch_command(subcommand).run_from_argv(self.argv)
  File "/usr/local/lib/python3.5/site-packages/django/core/management/base.py", line 294, in run_from_argv
    self.execute(*args, **cmd_options)
  File "/usr/local/lib/python3.5/site-packages/django/core/management/base.py", line 345, in execute
    output = self.handle(*args, **options)
  File "/usr/local/lib/python3.5/site-packages/django/core/management/commands/migrate.py", line 204, in handle
    fake_initial=fake_initial,
  File "/usr/local/lib/python3.5/site-packages/django/db/migrations/executor.py", line 115, in migrate
    state = self._migrate_all_forwards(state, plan, full_plan, fake=fake, fake_initial=fake_initial)
  File "/usr/local/lib/python3.5/site-packages/django/db/migrations/executor.py", line 145, in _migrate_all_forwards
    state = self.apply_migration(state, migration, fake=fake, fake_initial=fake_initial)
  File "/usr/local/lib/python3.5/site-packages/django/db/migrations/executor.py", line 244, in apply_migration
    (continues on next page)
```

(continued from previous page)

```

state = migration.apply(state, schema_editor)
File "/usr/local/lib/python3.5/site-packages/django/db/migrations/migration.py",
↳line 129, in apply
    operation.database_forwards(self.app_label, schema_editor, old_state, project_
↳state)
File "/usr/local/lib/python3.5/site-packages/django/db/migrations/operations/models.
↳py", line 96, in database_forwards
    schema_editor.create_model(model)
File "/usr/local/lib/python3.5/site-packages/django/db/backends/base/schema.py",
↳line 295, in create_model
    self.execute(sql, params or None)
File "/usr/local/lib/python3.5/site-packages/django/db/backends/base/schema.py",
↳line 112, in execute
    cursor.execute(sql, params)
File "/usr/local/lib/python3.5/site-packages/django/db/backends/utils.py", line 79,
↳in execute
    return super(CursorDebugWrapper, self).execute(sql, params)
File "/usr/local/lib/python3.5/site-packages/django/db/backends/utils.py", line 64,
↳in execute
    return self.cursor.execute(sql, params)
File "/usr/local/lib/python3.5/site-packages/django/db/utils.py", line 94, in __
↳exit__
    six.reraise(dj_exc_type, dj_exc_value, traceback)
File "/usr/local/lib/python3.5/site-packages/django/utils/six.py", line 685, in
↳reraise
    raise value.with_traceback(tb)
File "/usr/local/lib/python3.5/site-packages/django/db/backends/utils.py", line 62,
↳in execute
    return self.cursor.execute(sql)
File "/usr/local/lib/python3.5/site-packages/django/db/backends/mysql/base.py",
↳line 110, in execute
    return self.cursor.execute(query, args)
File "/usr/local/lib/python3.5/site-packages/MySQLdb/cursors.py", line 226, in
↳execute
    self.errorhandler(self, exc, value)
File "/usr/local/lib/python3.5/site-packages/MySQLdb/connections.py", line 36, in
↳defaulterrorhandler
    raise errorvalue
File "/usr/local/lib/python3.5/site-packages/MySQLdb/cursors.py", line 217, in
↳execute
    res = self._query(query)
File "/usr/local/lib/python3.5/site-packages/MySQLdb/cursors.py", line 378, in _
↳query
    rowcount = self._do_query(q)
File "/usr/local/lib/python3.5/site-packages/MySQLdb/cursors.py", line 341, in _do_
↳query
    db.query(q)
File "/usr/local/lib/python3.5/site-packages/MySQLdb/connections.py", line 280, in
↳query
    _mysql.connection.query(self, query)
django.db.utils.OperationalError: (1050, "Table 'courses_coursesuccession' already_
↳exists")

```

6.1.2 Solution

- Shut down all containers: `docker-compose down`

- Delete all docker containers and volumes. Delete the containers first in order to be able to delete the associated volumes. **WARNING:** This erases the entire database on your system. Useful commands: - `docker container ls` to get the ID of the containers, `docker container rm [ID]` to remove the container whose ID is [ID] - `docker volume ls` to get the ID of the volumes, `docker volume rm [ID]` to remove the volume whose ID is [ID]
- Start the server: `docker-compose up`
- In another terminal (while the server is still running): `mysql -h 127.0.0.1 --port=3309 -u root -p[password] -t tq_website < path/to/dump_now.sql`, where you have to set the correct path to the database dump that your IT board member gave you. Wait until this command has completed.
- Kill the server.
- Run `scripts/migrate.sh` from within the folder where the TQ website code is located.
- Start the server using `docker-compose up`. Your server should work fine now.

6.2 Solving migration errors

The steps described in this article were developed for [this commit](#). It took a long time to arrive at this solution. So now that you're here: it's not because you're stupid!

6.2.1 Scenario:

You try to migrate using our `scripts/makemigrations.sh` and `scripts/migrate.sh` scripts. When you run `makemigrations.sh`, everything is fine and you get a list of (external) apps for which updates exist:

```
[renato@linux-1 tq]$ sudo scripts/makemigrations.sh
Starting tq_db_1 ... done.. done

Migrations for 'cmsplugin_filer_link':
/usr/local/lib/python3.5/site-packages/cmsplugin_filer_link/migrations/0006_auto_20171014_1947.py:
- Alter field cmsplugin_ptr on filerlinkplugin
Migrations for 'cmsplugin_filer_image':
/usr/local/lib/python3.5/site-packages/cmsplugin_filer_image/migrations/0008_auto_20171014_1947.py:
- Alter field cmsplugin_ptr on filerimage
Migrations for 'djangoCMS_inherit':
/usr/local/lib/python3.5/site-packages/djangoCMS_inherit/migrations/0003_auto_20171014_1947.py:
- Alter field cmsplugin_ptr on inheritpageplaceholder
```

But when you run `migrate.sh`, you get the following error message:

```
[renato@linux-1 tq]$ sudo scripts/migrate.sh
Starting tq_db_1 ... done
Starting tq_memcached_1 ... done
Operations to perform:
  Apply all migrations: account, admin, auth, cms, cms_plugins, cmsplugin_filer_image, cmsplugin_filer_link, contenttypes, courses, django_celery_beat, django_celery_results, d
jangoCMS_googlemap, djangoCMS_inherit, djangoCMS_link, djangoCMS_textckeditor, easy_thumbnails, events, faq, filer, guardian, menus, organisation, payment, post_office, revers
ion, sessions, sites, socialaccount, survey
Running migrations:
  No migrations to apply.
Your models have changes that are not yet reflected in a migration, and so won't be applied.
Run 'python manage.py makemigrations' to make new migrations, and then re-run 'python manage.py migrate' to apply them.
```

6.2.2 Solution:

- Locate the Python packages corresponding to the Django apps that are listed by `makemigrations.sh`.
- Find the entries (and therefore the versions) of the packages in our `configurations/python-requirements`. In the images above, the affected packages are `cmsplugin-filer`, `django-filer` (not sure if both are needed... if in doubt, just upgrade all candidates) `djangoCMS-googlemap`, `djangoCMS-inherit` and `djangoCMS-link`.

- Search for the packages on [PyPI](#) and get the newest version.
- Update `python-requirements.txt` to the new values.

Always use `==` to indicate versions, never use `>=` !

Reason: Package upgrades might break our images. So if external package A is updated and the new version would break our build, we don't notice this. But when somebody tries to build our docker image at some point in the future, the build fails, although it has always worked before! Then it's difficult to find the correct package version that were in use at the time when the dependency was added/updated.

- Rebuild the image; the following code should be run in the parent directory that contains all code:

```
docker-compose build
```

- Actually perform the migrations:

```
scripts/makemigrations.sh
scripts/migrate.sh
```

- Don't forget to *update the table of our dependencies*!

6.3 CKEditor does not display an option to add links and/or images:

Run the following lines on the server:

```
docker-compose run --rm django python3 manage.py cms check
docker-compose run --rm django python3 manage.py cms delete-orphaned-plugins
```

You have to confirm the last command manually.

6.4 Error during docker-compose build:

Error:

```
Rolling back uninstall of cffi
Command "/usr/local/bin/python -u -c "import setuptools, tokenize;__file__='/tmp/pip-build-0k_ja8a5/cffi/setup.py';f=getattr(tokenize, 'open', open)(__file__);code=f.read().replace('\r\n', '\n');f.close();exec(compile(code, __file__, 'exec'))" install --record /tmp/pip-1ohgg37n-record/install-record.txt --single-version-externally-managed --compile" failed with error code 1 in /tmp/pip-build-0k_ja8a5/cffi/
ERROR: Service 'django' failed to build: The command '/bin/sh -c apk add --no-cache --virtual .build-deps build-base && pip3 install --upgrade -r /webapps/tq_website/configurations/python-requirements.txt && apk del .build-deps' returned a non-zero code: 1
```

Run the following lines on the server:

```
docker-compose build --no-cache
```

6.5 I get “INTERNAL SERVER ERROR”, but neither Sentry nor the log files notice it!

6.5.1 Background

This has already occurred once. In fact, the bug was reported by a teacher on 2018-04-08. The corresponding issue was #184. It was fixed on 2018-04-17 after hours of debugging (commit 945a8839f679951277c5d53c2311267a699c48c3).

6.5.2 Debugging process

Since no error message was generated, debugging had to be done manually:

- The error could not be reproduced on a development server (it only occurred for a few courses), so code modifications were performed on the production server **AFTER PERFORMING A BACKUP**. The changes were copied to a developer machine and committed, then deleted on the server to get in a consistent state. Then, the solution was pulled from the repo.
- Debugging was done in the following way:

Different messages were printed to the browser by return them as the message to an `HttpResponse` object.

6.5.3 Solution

Debugging made it clear that the error did not come from the view. Commenting out different lines of code indicated that the part where the filename was set lead to the error. The reason was the following:

In that course period, the people from dance administration used German Umlaut characters (ü, ä, ö) in the course name.

Example: Social 2 (Axel) früh (FS2018 Q2)

Since the filename of the file to be downloaded should not contain such characters, an error was thrown.

It was then easy to fix the error by simply converting the Umlauts to their corresponding ASCII form (Ex.: ü → ue)

6.6 Certificate renewal

The live server uses <https://letsencrypt.org> to provide secure, HTTPS connections. Every 3 months, the certificates expire and must be renewed. Ideally this is done by a cronjob, however due to Issue #114 this is not working at the moment. The procedure to renew the certificate is:

- Stop the live server:

```
cd tq_website
docker-compose stop
```

- Renew certificates:

```
certbot renew
```

- Start the live server again:

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

- Check the logs and make sure everything is running as expected.
- Enjoy a cup of coffee.

6.7 Enrollment error: No account for email (though the account exists)

This issue stems from an old version of the website: There, it was possible to enrol to a course without having an account. The website then internally created an account for the subscription. That lead to multiple acounts per email address, which seems to break allauth, the authentication plugin we're using.

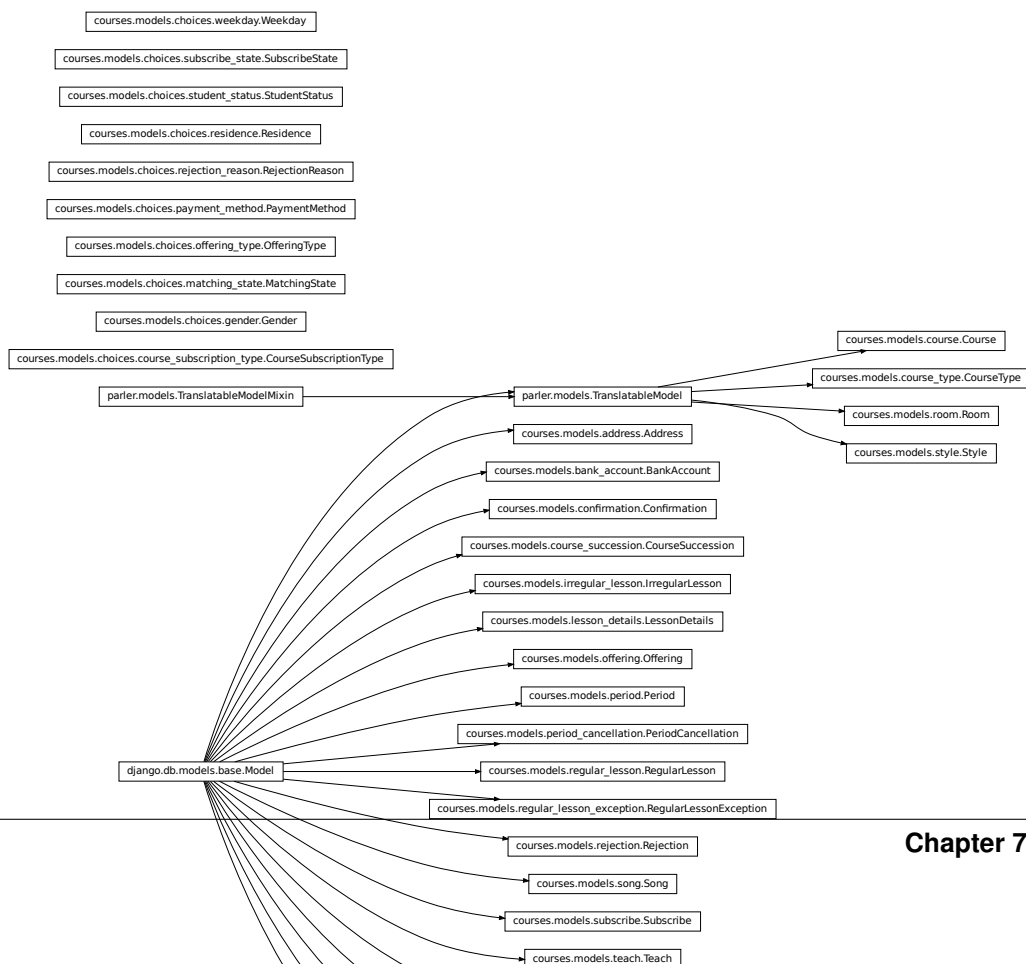
Solution: Make all users that have the same email address inactive but one. Do this by clocking on “tanzquotient.org” in the admin bar, then “users...” and search for the email address. Then mark the accounts you want to inactivate (policy: the surviving account should be the one with the most recent “last logged in” field), then select “Make inactive” in the dropdown actions list.

CHAPTER 7

Courses

7.1 Models

7.1.1 Diagram



7.1.2 Summary

Address(id, street, plz, city, country)	
BankAccount(id, iban, bank_name, ...)	
Confirmation(id, subscription, date, mail)	
Course(id, name, type, subscription_type, ...)	
CourseSubscriptionType	
CourseSuccession(id, predecessor, successor)	
CourseType(id, name, level, couple_course)	
Gender	
IrregularLesson(id, course, date, time_from, ...)	
LessonDetails(id, room)	
MatchingState	
Offering(*args, **kwargs)	An offering is a list of courses to be offered in the given period
OfferingType	
PaymentMethod	
Period(id, name, date_from, date_to)	
PeriodCancellation(id, name, period, date)	
RegularLesson(id, course, weekday, ...)	
RegularLessonException(id, regular_lesson, ...)	
Rejection(id, subscription, date, reason, ...)	
RejectionReason	
Residence	
Room(id, name, address, url, contact_info)	
Song(id, title, artist, length, speed, ...)	
StudentStatus	
Style(id, name, parent_style, ...)	
Subscribe(id, user, course, date, partner, ...)	
SubscribeState	
Teach(id, teacher, course, welcomed, hourly_wage)	
TeachLesson(id, teacher, lesson, hourly_wage)	
TeacherWelcome(id, teach, date, mail)	
UserProfile(user, language, legi, gender, ...)	
Voucher(id, purpose, percentage, key, ...)	
VoucherPurpose(id, name, description)	
Weekday	

7.1.3 Details

7.2 Services

7.2.1 Summary

Course(id, name, type, subscription_type, ...)	
CourseManager()	
DEFAULT_BODY_HEIGHT	int(x=0) -> integer int(x, base=10) -> integer

Continued on next page

Table 2 – continued from previous page

Http404	
HttpResponseServerError([content])	
INVALID_TITLE_CHARS	Compiled regular expression objects
IrregularLesson(id, course, date, time_from, ...)	
MESSAGE_NO_PARTNER_SET	str(object='') -> str str(bytes_or_buffer[, encoding[, errors]]) -> str
MatchingState	
<i>NoPartnerException</i>	
ObjectDoesNotExist	The requested object does not exist
Offering(*args, **kwargs)	An offering is a list of courses to be offered in the given period
OfferingType	
Prefetch(lookup[, queryset, to_attr])	
Q(*args[, _connector, _negated])	Encapsulate filters as objects that can then be combined logically (using & and).
RegularLesson(id, course, weekday, ...)	
RegularLessonException(id, regular_lesson, ...)	
Subscribe(id, user, course, date, partner, ...)	
SurveyInstance(id, survey, email_template, ...)	
TranslationUtils	
User(*args, **kwargs)	Users within the Django authentication system are represented by this model.
UserProfile(user, language, legi, gender, ...)	
Voucher(id, purpose, percentage, key, ...)	
Weekday	
breakup_couple(subscriptions, request)	
<i>calculate_relevant_experience</i> (user, course)	finds a list of courses the “user” did already and that are somehow relevant for “course”
<i>clean_username</i> (name)	first try to find ascii similar character, then strip away disallowed characters still left
<i>confirm_subscription</i> (subscription[, ...])	sends a confirmation mail if subscription is confirmed (by some other method) and no confirmation mail was sent before
confirm_subscriptions(subscriptions[, ...])	
copy_course(course[, to, set_preceding_course])	
correct_matching_state_to_couple(subscriptions)	
create_course_info(course)	
create_user_info(user)	
date	date(year, month, day) -> date object
detect_rejection_reason(subscription)	detect the reason why the subscription is rejected :return: the reason as constant from Rejection.Reason
export(export_format, title, data[, multiple])	
export_subscriptions(course_ids, export_format)	ex-
<i>export_summary</i>	exports a summary of all offerings with room usage, course/subscription numbers
<i>export_teacher_payment_information</i>	Exports a summary of the given offerings concerning payment of teachers.
find_unused_username_variant(name[, ignore])	

Continued on next page

Table 2 – continued from previous page

<code>format_prices(price_with_legi, ...[, ...])</code>	
<code>generate_voucher_pdf(vouchers)</code>	
<code>get_all_offerings()</code>	
<code>get_current_active_offering()</code>	
<code>get_historic_offerings([offering_type])</code>	
<code>get_offerings_to_display([request, ...])</code>	return offerings that have display flag on and order them by start date in ascending order
<code>get_or_create_userprofile(user)</code>	
<code>get_sections(offering[, course_filter])</code>	
<code>get_subsequent_offering()</code>	
<code>get_upcoming_courses_without_offering()</code>	
<code>log</code>	Instances of the Logger class represent a single logging channel.
<code>match_partners(subscriptions[, request])</code>	
<code>model_attribute_language_fallback(model, ...)</code>	
<code>reject_subscription(subscription[, reason, ...])</code>	sends a rejection mail if subscription is rejected (by some other method) and no rejection mail was sent before
<code>reject_subscriptions(subscriptions[, ...])</code>	same as reject_subscription, but for multiple subscriptions at once
<code>reverse(viewname[, urlconf, args, kwargs, ...])</code>	
<code>send_course_email(data, courses)</code>	
<code>send_online_payment_successful(subscription)</code>	
<code>send_participation_confirmation(subscription)</code>	
<code>send_payment_reminder(subscription)</code>	
<code>send_rejection(subscription, reason)</code>	
<code>send_sorry_for_incorrect_reminder(subscription)</code>	
<code>send_subscription_confirmation(subscription)</code>	
<code>send_teacher_welcome(teach)</code>	
<code>send_vouchers(data, recipients)</code>	
<code>settings</code>	A lazy proxy for either global Django settings or a custom settings object.
<code>subscribe(course_id, data)</code>	Actually enrolls a user or a pair of users in a course
<code>unconfirm_subscriptions(subscriptions[, request])</code>	
<code>unmatch_partners(subscriptions, request)</code>	
<code>unreject_subscriptions(subscriptions[, request])</code>	
<code>update_user(user, user_data)</code>	
<code>welcome_teacher(teach)</code>	
<code>welcome_teachers(courses, request)</code>	
<code>welcome_teachers_reset_flag(courses, request)</code>	

7.2.2 Details

exception `courses.services.NoPartnerException`

`courses.services.calculate_relevant_experience(user, course)`

finds a list of courses the “user” did already and that are somehow relevant for “course”

`courses.services.clean_username(name)`

first try to find ascii similar character, then strip away disallowed characters still left

`courses.services.confirm_subscription(subscription, request=None, al-
low_single_in_couple_course=False)`

sends a confirmation mail if subscription is confirmed (by some other method) and no confirmation mail was sent before

`courses.services.export_summary()`

exports a summary of all offerings with room usage, course/subscription numbers

`courses.services.export_teacher_payment_information()`

Exports a summary of the given offerings concerning payment of teachers.

Contains profile data relevant for payment of teachers and how many lesson at what rate to be paid.

Parameters

- **export_format** – export format
- **offerings** – offerings to include in summary

Returns response or None if format not supported

`courses.services.get_offerings_to_display(request=None, force_preview=False,
only_regular_offerings=False)`

return offerings that have display flag on and order them by start date in ascending order

`courses.services.reject_subscription(subscription, reason=None, send_email=True)`

sends a rejection mail if subscription is rejected (by some other method) and no rejection mail was sent before

`courses.services.reject_subscriptions(subscriptions, reason=None, send_email=True)`

same as reject_subscription, but for multiple subscriptions at once

`courses.services.subscribe(course_id, data)`

Actually enrolls a user or a pair of users in a course

CHAPTER 8

Indices and tables

- `genindex`
- `modindex`
- `search`

C

`courses.models`, [39](#)

`courses.services`, [41](#)

C

`calculate_relevant_experience()` (*in module* `courses.services`), 41
`clean_username()` (*in module* `courses.services`), 41
`confirm_subscription()` (*in module* `courses.services`), 42
`courses.models` (*module*), 39
`courses.services` (*module*), 41

E

`export_summary()` (*in module* `courses.services`), 42
`export_teacher_payment_information()` (*in module* `courses.services`), 42

G

`get_offerings_to_display()` (*in module* `courses.services`), 42

N

`NoPartnerException`, 41

R

`reject_subscription()` (*in module* `courses.services`), 42
`reject_subscriptions()` (*in module* `courses.services`), 42

S

`subscribe()` (*in module* `courses.services`), 42