EvaP Documentation

Release 1.0

Michael Grünewald, Stefan Richter

February 28, 2015

Contents

CHAPTER 1

Contents

1.1 Installation

1.1.1 Vagrant

Usually it's easiest to just use the Vagrant VM. Since we use puppet in git submodules for easier configuration of the VM, you'll need to clone with `--recurse-submodules` or run `git submodule update --init` after checking out EvaP, and of course you need to have Vagrant installed. Then you can run `vagrant up` in your EvaP root directory; this builds you a fully working EvaP installation with PostgreSQL as database and Apache as webserver.

You can use EvaP at `http://localhost:8000/` on your host, write code in your git repo, and access the VM with `vagrant ssh`. Your git checkout is mounted in `/vagrant` there.

Use `vagrant halt` to shutdown the VM, and `vagrant destroy` to delete it.

If you don't want to use Vagrant, you can use the following instructions to manually set up EvaP.

1.1.2 Vagrant and Hyper-V

To use the Vagrant VM with Hyper-V as provider, do the following:

1. Enable Hyper-V (note the warning on this page).

2. Configure Networking

   You can either connect the VM to your external network directly by creating an *External Virtual Switch* in Hyper-V Manager->Virtual Switch Manager, or use "NAT-Mode" by creating an *Internal Virtual Switch* and setting up Windows Internet Connection Sharing (recommended).

3. `vagrant up --provider hyperv`

   Select the virtual switch created in step 2 and enter your Windows host credentials when prompted, in order to give the virtual machine access to the working directory via SMB.

4. Because port forwarding doesn't work with Hyper-V, you have to access the EvaP website using the VM's IP address.

1.1.3 Filesystem Structure

We recommend that you install the application into the directory `/opt/evap` according to the filesystem hierarchy standard. In the remaining steps, we will assume that this is your installation directory.

Clone the repository or copy the files into that directory. The installation should be correct if the settings file has the path `/opt/evap/evap/settings.py`. Make sure that all files and directories are readable by the Apache web server. Additionally please make sure that the directory `/opt/evap/evap/upload` is writable by the web server.

1.1.4 Dependencies

EvaP is written in Python using the Django framework and you need at least Python 3.4 to run it. Apart from Python and Django there are some other dependencies that are listed in the file `requirements.txt`. Use pip to install these with the following command: `pip install -r requirements.txt`.

For Windows users: As `lxml` has native code and compiling it with your installed compiler usually fails, you need to install a precompiled package from lfd.uci.edu.

For Windows users: For the compilation of translation files gettext is used. This can be downloaded as a precompiled package from the GnuWin project.

1.1.5 Database Initialization

To initialize the database during the installation you have to perform the following command in `/opt/evap` directory:

- `python manage.py migrate`

1.1.6 File Refresh

You have to run some additional commands during the installation and whenever you update EvaP. Perform these steps in the `/opt/evap` directory after you have upgraded the files:

- `python manage.py migrate` to perform any potential database updates.
- `python manage.py collectstatic` to collect all files that the front-end webserver should serve.
- `python manage.py compilemessages` to update the binary translation catalog.

Finally, restart the Apache web server.

1.1.7 Load test data

The Vagrant VM automatically loads test data from a fixture on it's creation. If you want to load this data in your manual installation, run the following commands (*Note: This will remove all data previously stored in the database*):

```
python manage.py flush --no-initial-data
python manage.py loaddata test_data.json
```

1.1.8 Development Environment: Setup and Server Running

The manual creation of a superuser is just recommended for development environments. In productive environments a Kerberos authentification system or similar should be used.

To create a superuser perform in the `/opt/evap` directory the command `python manage.py createsuperuser`.

A login in a development environment can be simulated by setting the variable `REMOTE_USER` to your previously created user.

The server for this environment can be started from within the `/opt/evap` directory with `python manage.py runserver`.

1.1.9 Productive Environment: Settings

The configuration of the application is done by creating a `localsettings.py` in the `evap` folder and overwriting the defaults from `settings.py` there. The defaults should be OK for most development purposes. For a production environment you should change the following settings:

- Choose an appropriate database and modify the `default` entry in the `DATABASES` settings. Please make sure that you use a database that supports transactions.
- Change the `DEFAULT_FROM_EMAIL` to an administrative mail address that is used as the sender of the mails generated by the system.
- Change `MEDIA_ROOT` to a directory that is writable by the web application. This directory will hold user-uploaded files.
- You should change the `SECRET_KEY`.
- Finally, set `DEBUG` to `False`.

1.1.10 Productive Environment: Apache 2 Configuration

We recommend the following Apache configuration:

:: WSGIScriptAlias / /opt/evap/evap/wsgi.py

> Alias /static /opt/evap/evap/static_collected <Location /static>
>
> > ExpiresActive On ExpiresDefault “access plus 1 month”
>
> </Location>
>
> Alias /media /opt/evap/evap/upload

1.1.11 Productive Environment: Kerberos Authentication

To use Kerberos as an authentication backend, do the following:

- run `pip install django_auth_kerberos`
- copy the following to your `localsettings.py` and edit `KRB5_REALM` and `KRB5_SERVICE` according to your setup:

```
KRB5_REALM = 'EXAMPLE.COM'
KRB5_SERVICE = 'krbtgt@AS.EXAMPLE.COM'
INSTALLED_APPS += ('django_auth_kerberos',)
MIDDLEWARE_CLASSES += ('django_auth_kerberos.backends.KrbBackend',)
```

1.1.12 Productive Environment: Cron Configuration

EvaP has components which need to react to timed events. This behavior is implemented by running two cronjobs, which in turn trigger a management command. One should be run hourly and the other one daily.

For example you could use a /etc/cron.hourly/evap like

```
#!/bin/sh

pushd  /opt/evap
/usr/bin/python manage.py run_tasks
popd
```

And a /etc/cron.daily/evap like

```
#!/bin/sh

pushd  /opt/evap
/usr/bin/python manage.py run_tasks daily
popd
```

1.2 Source Code

EvaP is a standard Django project, divided up into several applications:

`evaluation` contains all the data models relevant for the evaluation, but no views. It also contains some generic code used by other applications.

`staff` provides views for the student representatives

`contributor` provides views for lecturers

`results` provides views to show results and produce exports

`student` provides views relevant for students (voting etc.)

1.2.1 Data Import

The key module for data imports is `staff.importers`.

1.3 License

The software is licensed under the MIT license. The source code includes other components in whole or in part; namely jQuery, jQuery UI, jQuery UI Multiselect and jQuery Formset. These components are used under the MIT resp. BSD licenses. It also uses symbols of the Silk icon set from famfamfam.com, which is licensed under a Creative Commons Attribution 2.5 License.

The source repository may include logos, names or other trademarks of the Hasso Plattner Institute or other entities; potential usage restrictions for these elements still apply and are not touched by the software license.

```
EvaP - Evaluation Platform
Copyright (C) 2011-2015 by Michael Grünewald, Stefan Richter, Johannes Linke and Johannes Wolf

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
```

```
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in
all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
THE SOFTWARE.
```

CHAPTER 2

Indices and tables

- *genindex*
- *modindex*
- *search*