Table of Contents

Harbourmaster Single Sign On (SSO)

With the Harbourmaster Single Sign On (SSO), Valiton has developed a solution that focuses on the media industry. As part of the Thunder initiative of Hubert Burda Media, the Harbourmaster is available as a ‘Freemium’ version.

Whether for registered users, newsletter subscribers or anonymous users – all types of visitors on websites, online shops and apps, Single Sign On is essential for online businesses.

Thunder comes with built in Drupal User Management, which could be used as long you only need to allow users to login on your website. For your Mobile App, you may have considered registering via Facebook or Twitter.

With the rapid growth of your online business, and the use of various apps, websites, shops, and landing pages, the need for a Single Sign On solution has now become essential.

Some of the main strengths of SSO are:

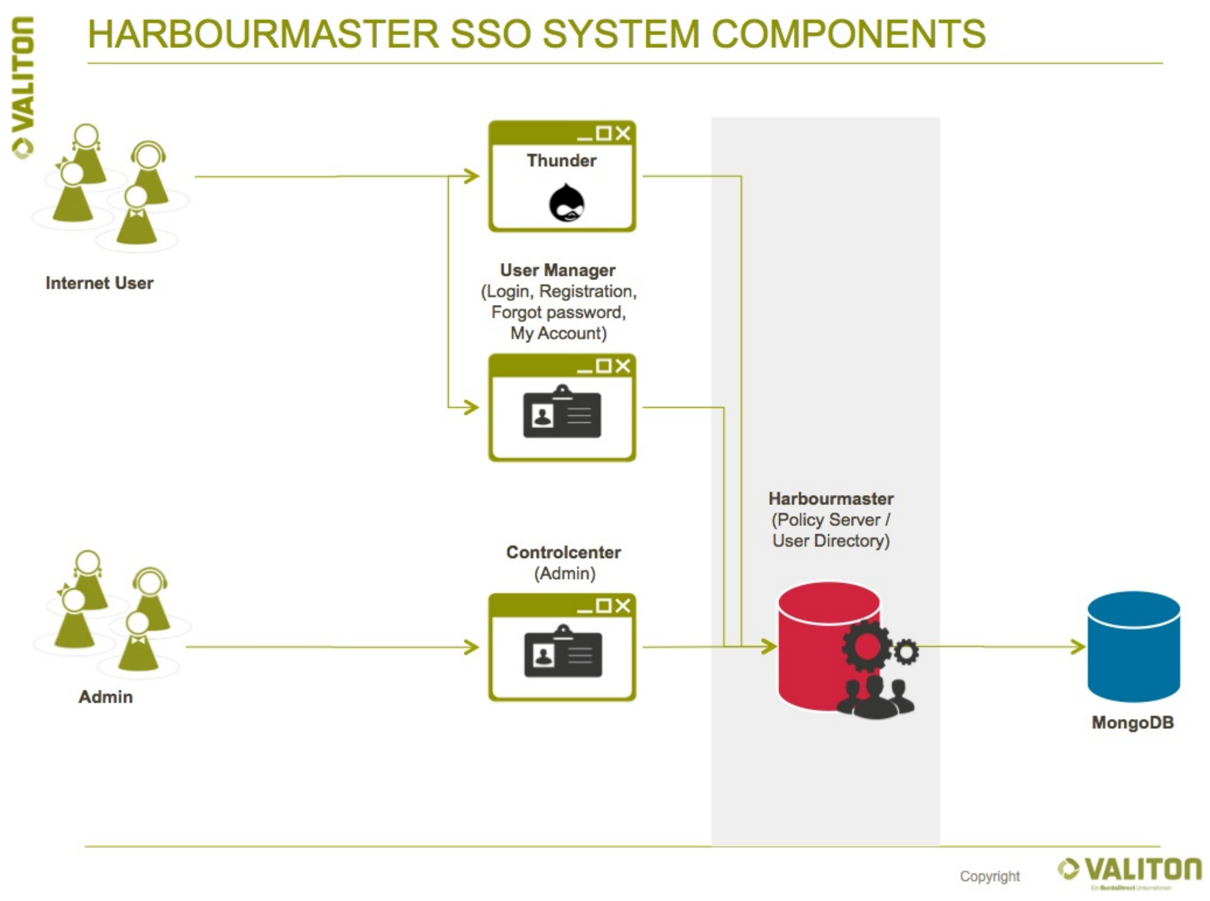
- Private Hosting
  - Cloud scaling compatibility
  - Hosting friendly by using Docker container
- Multi-tenant (two tenants included in Freemium)
- Newsletter Management (not available in Freemium)
  - Subscription status Sync with email service providers (Salesforce Marketing Cloud, Elaine, etc)
  - Integration of subscribers' status
- Management of data protection and advertising agreement
- Paid content access control (billing component requires the Valiton product, Tallymann)
  - CDN/Caching compatibility
- iOS SDK (not available in Freemium)
- Android SDK (not available in Freemium)

- Cross Domain login/logout (not available in Freemium)
- Login with Facebook, Twitter, Google Plus et al
- API Driven – all functions can also be accessed by third-party systems via API
- Customer self-service widget

Valiton's many years of development experience in NodeJS was the base on which it implemented the applications in the Harbourmaster SSO and developed a fast and resource-efficient SSO solution as a loosely coupled architecture. Harbourmaster is also the central component for access management of premium content on publishing sites. We can perfectly apply our know-how in the field of online shops and mobile apps into the integration of third party systems.

For more information about Harbourmaster SSO please feel free to contact us directly. kontakt@valiton.com

System Components


HARBOURMASTER SSO SYSTEM COMPONENTS
VALITON
Internet User
Thunder
User Manager (Login, Registration, Forgot password, My Account)
Admin
Controlcenter (Admin)
Harbourmaster (Policy Server / User Directory)
MongoDB
Copyright
VALITON

The Harbourmaster SSO contains three components in the Freemium version of Thunder. The Freemium version is under the attached LICENSE.

The Core component is the Harbourmaster it fully API driven. It manages User Authentication and Authorisation.

The user facing component is the Usermanager, the main feature of which is a widget that is embedded in the website and backend, and which sends confirmation emails and communicates with the Harbourmaster.

The Controlcenter gives admin/product manager/call center agents access to all necessary information. Groups and policies allow granular access control.

Quick Start Guide

User Manager

Control Center

Harbourmaster

Drupal Module

Quick Start Guide

The purpose of this quick start guide is to provide you with a fast and easy way of getting Harbourmaster set up and running on your local environment. For an advanced production setup, please refer to the more detailed documentation further below.

What is in this Guide

This guide will help you set up a set of Docker containers which will run the following applications:

- MongoDB (used as main database)
- Redis (used for caching)
- Ngnix (used as reverse proxy)
- Harbourmaster
- Usermanager
- Control Center

Create a docker runtime environment

This guide uses the software containerization platform Docker in order to make the installation process as easy as possible. The first step is to install a docker runtime environment on your system. We recommend using the lastest version of the native Docker client. If you're running an older version of your operating system which doesn't meet Docker's requirements, you can alternatively install Docker Toolbox.

For further instructions on how to get Docker or Docker Toolbox running on your system, please refer to the official starting guide: Get Started with Docker

Clone the harbourmaster-docs repo and switch to the quickstart folder

Get the lastest version of the Harbourmaster Quick Start Guide from our GitHub repository over at https://github.com/valiton/harbourmaster-docs.git. This repository contains all the necessary files to get you set up and running. After you've downloaded the files, switch to the quickstart directory to get started:

```
git clone https://github.com/valiton/harbourmaster-docs.git
cd harbourmaster-docs/quickstart/
```

Accept Harbourmaster license

In order to run Harbourmaster, its license has to be accepted via configuration. This quick start guide uses the `shared_config.env` configuration file, which already contains the necessary line `LICENSE=accept` .

For more information on how to view and accept the license for your production setup, please refer to the chapter Harbourmaster.

Start containers with Docker Compose

Docker Compose is a tool which can automatically start up multiple Docker containers in a way which is defined in a `docker-compose.yml` file (included in this quick start guide). While inside the quickstart directory, simply run:

```
docker-compose up
```

The latest version of Harbourmaster will automatically be downloaded and started up using a pre-defined configuration.

If Docker Compose didn't come with your Docker installation, please refer to the official Docker instructions on how to install this tool.

Seed Harbourmaster tenants

This guide comes with a shell script which seeds the database with a few test-users as well as the admin user. During its run, the seed-harbourmaster.sh script will prompt you to input an initial password for the admin user. In the quickstart directory, run:

```
./seed-harbourmaster.sh
```

Add /etc/hosts entries

Create the following entry in your /etc/hosts file. If your Docker containers are running non-locally, replace the IP address in this entry accordingly.

All Harbourmaster applications are running on static TCP ports (Harbourmaster and Usermanager on port 80, Control Center on port 18040). As previously mentioned, the `docker-compose.yml` also starts up an Nginx container, which listens on port 80 and maps our subdomains (e.g. `usermanager.thunder` or `harbourmaster.thunder`) to the individual application containers.

```
127.0.0.1 usermanager.thunder.dev harbourmaster.thunder.dev controlcenter.thunder.dev
```

Accessing the applications

To access the applications, simply point your web browser to the following URLs:

Control Center

http://controlcenter.thunder.dev

(username: "admin", password: <YOUR_INPUT_DURING_SEED>)

User Manager

http://usermanager.thunder.dev/demo

As described in the Usermanager chapter, you can configure an SMTP server to handle user registrations. It is, however, still possible to create users using the usermanager for testing purposes, even without any SMTP configurations.

http://harbourmaster.thunder.dev

Configuring the Drupal Module

For the installation and configuration of the Drupal Module, please refer to the chapter Drupal Module. Once installed, users will be able to register and log in using `<YOUR_DRUPAL_URL>/harbourmaster/login` .

General outline for your production setup

In this Quick Start Guide, we have provided you with pre-configured docker-compose and *.env files in order to provide you with a working setup easily and quickly.

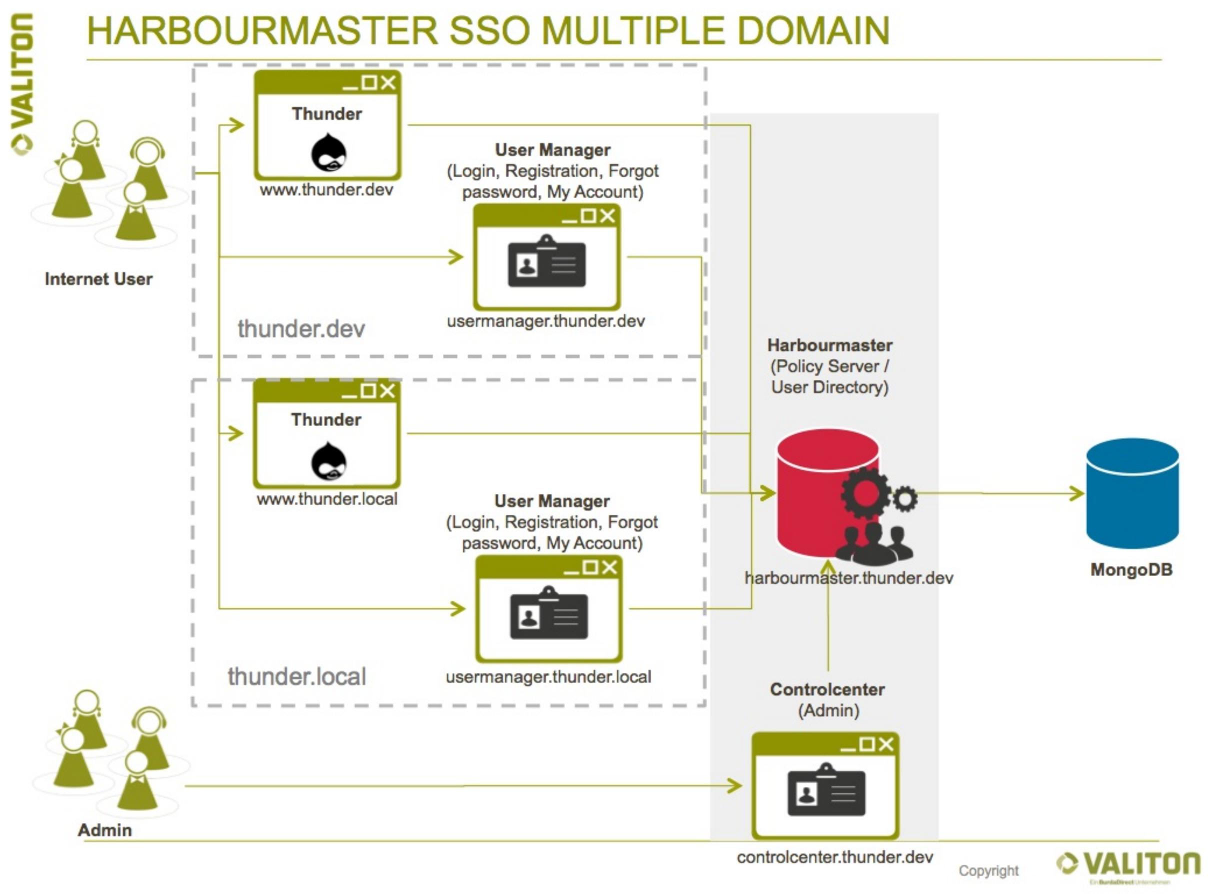
When setting up Harbourmaster in a production environment, you should consider starting up each Harbourmaster application container (Harbourmaster, Usermanager and Control Center) individually and providing your own configurations. Please refer to the configuration section of each application to get an overview of how they can be customized and consider using your pre-configured local setup as a reference point.

When working with Docker, the Harbourmaster applications will accept configuration options passed to them in environment variables. Each chapter provides you with the relevant variables and gives you example Docker run commands. For more information on how to start up applications using Docker, you can refer to the official documentation at https://docs.docker.com/engine/reference/run/.

The Harbourmaster applications are all running on static ports which cannot be changed. You can, however, use Docker's EXPOSE feature (see link above) to determine which host port should be mapped to the application. As demonstrated in this Quick Start Guide, you can then set up a reverse proxy using applications such as Nginx or HAProxy. If you would like to set up an Nginx server similar to the one in this guide (and enable additional features such as SSL termination), you can check out our example nginx.conf file and refer to the official documentation (specifically the Beginner's Guide) for setup instructions.

Quick Start Guide Multi Domain Setup

With Harbourmaster it is possible allow users to login with the same username and password on multiple top level domains e.g. `tunder.dev` and `thunder.local`. This requires to set up multiple usermanagers, each dedicated to their own domain but each communicating with the same Harbourmaster. This setup allows for the most customization since each usermanager can be configured individually (see chapter Usermanager).


HARBOURMASTER SSO MULTIPLE DOMAIN
VALITON
Internet User
Thunder
www.thunder.dev
User Manager
(Login, Registration, Forgot password, My Account)
usermanager.thunder.dev
thunder.dev
Thunder
www.thunder.local
User Manager
(Login, Registration, Forgot password, My Account)
usermanager.thunder.local
thunder.local
Harbourmaster
(Policy Server / User Directory)
harbourmaster.thunder.dev
MongoDB
Controlcenter
(Admin)
controlcenter.thunder.dev
Admin
Copyright
VALITON

It is possible to automatically log in users on all SSO domains with one single login action, also known as **cross domain login**. This feature relies on third-party cookies (setting a cookie on a domain which the user is not currently visiting). Some browser block third-party cookies, which is why this feature is not compatible with all browsers.

This guide aims to provide you with a quick way to set up a local multi usermanager installation. It is assumed that you have followed our Quick Start Guide to the point where the Harbourmaster, Usermanager and Controlcenter are all running on your system.

Adding a second Usermanager to the Quick Start Guide

The Harbourmaster quick start repository comes with the docker-compose-second-domain.yml which is pre-configured to add a second Usermanager to the Quick Start Guide. This Usermanager will be running under the `usermanager.thunder.local` domain.

Stop the docker-compose run from the Quick Start Guide and start it again with providing both docker compose files

```
cd harbourmaster-docs/quickstart/
docker-compose -f docker-compose.yml -f docker-compose-second-domain.yml up
```

Add /etc/hosts entries

Create the following entry in your /etc/hosts file. If the second Usermanager's Docker container is running non-locally, replace the IP address in this entry accordingly.

```
127.0.0.1 usermanager.thunder.local www.thunder.local
```

Accessing the applications

To access your second Usermanager, simply point your web browser to http://usermanager.thunder.local/demo/.

You should now have two Usermanagers running under two different domains: `thunder.local` and `thunder.dev`. Both Usermanagers are communicating with the same Harbourmaster, allowing users to log in accross multiple domains.

In your production setup, you will have to customize each Usermanager's configuration. Please refer to the configuration section of the Usermanager chapter, especially regarding the variables `SSO_WIDGET_CORS_HTTP_HOSTS` and `CROSS_DOMAIN_LOGIN_BASE_URL`, as these are required to be set according to your setup.

*Note concerning the Drupal Module installation: In the **URL to usermanager** option, use http://usermanager.thunder.local*

Harbourmaster

About

The Harbourmaster application is the backbone of the Harbourmaster SSO. It provides the API which powers all other applications.
https://hub.docker.com/r/valiton/harbourmaster/

Its main API is the REST API, which contains a full documentation with Swagger:
http://YOUR_HARBOURMASTER_URL/docs

As well as the Rest API, it provides a change notification mechanism that allows notifications to be received for example, when a user changes their profile data (not included in the Freemium version).

The Harbourmaster stores data in a MongoDB.

Modular Architecture

The Harbourmaster comprises a core module and a couple of feature extending modules listed. Not all modules are include in the Freemium version.

Module Name	Description	In Freemium
harbourmaster	Core module with all API required for basic SSO	YES
hms-module-user-confirmable	Adds API for user registration confirmation	YES
hms-module-user-tokens	Adds API to assign tokens to users, used in combination with mobile SDK	YES
hms-module-anonymous-sessions	Adds API for handling anonymous sessions	YES
hms-module-oauth2	Adds OAuth2 API	NO
hms-module-purchases	Adds API to assign purchases (paid one time) to users	NO
hms-module-entitlements	Adds API to assign entitlements (paid subscription) to users	NO
hms-module-entitlements-postentitlement-ciscom-crm	Adds API to check subscription status with a external ciscom CRM	NO
hms-module-notificator-kinesis	Adds change notifications using AWS Kinesis	NO
hms-module-notificator-sqs	Adds change notifications using AWS SQS	NO
hms-module-notificator-redis-sentinel	Adds change notifications using Redis, also Redis Sentinel compatible	NO
hms-module-managed-subscriptions	Adds API to manage newsletter subscription and advertisement permissions, one tenant can have one newsletter account	NO
hms-module-multiclient-subscriptions	Adds API to manage newsletter subscription and advertisement permissions, one tenant can have multiple newsletter accounts	NO

License

The Harbourmaster Freemium is licensed under the attached LICENSE.

To start the Harbourmaster you need to accept the license by passing the LICENSE=accept, to display the license pass LICENSE=view

```
LICENSE=<accept|view>
```

Simple command to view just the license and dont run all the containers

```
docker run -e LICENSE=view valiton/harbourmaster
```

Configuration

The necessary configuration of the Harbourmaster is defined with environment variables.

```
LICENSE=<accept|view>

REDIS_PORT_6379_TCP_ADDR=<ip of redis>  #compatible with docker link redis
REDIS_PORT_6379_TCP_PORT=<prot of redis> #compatible with docker link redis

MONGO_PORT_27017_TCP_ADDR=<ip of mongodb> #compatible with docker link mongo
MONGO_PORT_27017_TCP_PORT=<prot of mongodb> #compatible with docker link mongo


# used for seed script run, shared values with usermanger
HARBOURMASTER_TENANT=<teante name > e.g. thunder
USERMANAGER_HMS_USER_KEY=<usermanager key| 20 char hex format> e.g.  32be04fb9495229f3e4f
USERMANAGER_HMS_USER_SECRET_KEY=<usermanager secret key random string> e.g. 58c94af9f955eebeb
```

Docker run

Start a mongodb and redis as a container:

```
docker run -d --name=mongo mongo:3.2
docker run -d --name=redis redis
```

Start the Harbourmaster server:

```
docker run --name=thunder-harbourmaster -it -e LICENSE=accept --link mongo:mongo --link redis
```

Seed the mongodb with an initial data set; this is necessary because all API endpoints need a logged in user. The seedscript creates the initial user.

```
docker exec -it thunder-harbourmaster bash -c "node /app/scripts/seed-create-thunder-tenant.j
```

Backup

The Quick Start Guide repository contains a backup script which can be used as a starting point to create an automated backup process for the MongoDB. It is recommended to use a Cron job to automate and schedule the backup prodecure.

API

The Harbourmaster provides a HTTP REST API. Its main API is the REST API, which is full documents with swagger. http://YOUR_HARBOURMASTER_URL/docs

The basic API structure example:

```
<server> domain where the harbourmaster is deployed to.
<TENANT> placeholder for the tenant to be used.
<resource> placeholder for the resource to be used.
<nestedResource> placeholder for a nested resource.

http://<server>/v1/<TENANT>/<resource>
http://<server>/v1/<TENANT>/<resource>/<ID>
http://<server>/v1/<TENANT>/<resource>/<ID>/<nestedResource>
```

API responses

Positive response

All positive responses have this general JSON structure:

```
{
 "status": 1,
 "responseCode":  <HTTP_RESPONSE_CODE>,
 "data": { <DATA> }
}
```

Error response

All error responses have this general JSON structure:

```
{
 "status": 0,
 "message": "<MESSAGE>",
 "responseCode": <HTTP_RESPONSE_CODE>
}
```

Policies

Policies in the Harbourmaster are a very flexible way to grant access to resources in the Harbourmaster. The policy system is loosely inspired by the Amazon AWS IAM policies.

All Harbourmaster API endpoints verify access based on the policy of the requesting user.

The API endpoint is represented as action and the resource, depending on the context of the created/updated/shown/deleted data.

The actions and resources involved in the Harbourmaster API are listed in the swagger documentation.

A Policy consists of one or many statements, which are structured as follows:

- **Effect**: can be Allow or Deny
- **Tenant**: the tenant where the policy should apply; gets converted in a regular expression
- **Action**: represents an API endpoint action; gets converted in a regular expression
- **Resource**: the resource which is affected by the action; get converted in a regular expression
  - Special Resource **[currentUser]** access is only granted to the resource owned by the accessing user

Policy Example

Full access to all actions on all resource in all tenants:

```
{"Statement":[
 {"Tenant":"*", "Effect":"Allow","Action":"*","Resource":"*"}
]}
```

currentUser User can read and edit his own data

```
{ "Statement": [
 { "Tenant": "thunder", "Effect": "Allow", "Action": "hms:login", "Resource": "[currentUser]"
 { "Tenant": "thunder", "Effect": "Allow", "Action": "hms:updateUserPassword", "Resource": "[
 { "Tenant": "thunder", "Effect": "Allow", "Action": "hms:getUserServices", "Resource": "[cur
 { "Tenant": "thunder", "Effect": "Allow", "Action": "hms:getUser", "Resource": "[currentUser
 { "Tenant": "thunder", "Effect": "Allow", "Action": "hms:updateUser", "Resource": "[currentU
 { "Tenant": "thunder", "Effect": "Allow", "Action": "hms:listUserEntitlements", "Resource":
 { "Tenant": "thunder", "Effect": "Allow", "Action": "hms:listAllowedTenants", "Resource": "[
]}
```

User Manager

About

The usermanager extends the Harbourmaster SSO. It is a dedicated web application where users of the single sign-on can manage their user account.
https://hub.docker.com/r/valiton/usermanager/

The user facing component is the usermanager; its main features are a frontend widget that is embedded in the website, and a backend that sends confirmation emails and communicates with the Harbourmaster.

Features

- Registration
- Password Recovery
- Login/Social Login
- Logout
- My Account
    - Change User Data
    - Change Avatar
    - Change Password
- Opt-in/double opt-in
- Newsletter subscription/unsubscribe

Backend application

The usermanager backend uses the Harbourmaster API to achieve its main tasks. The key responsibilities of the usermanager backend are:

- User registration
    - Send confirmation email
- User Login
- Show Profile

- Edit Profile
    - User image resize and upload to Amazon S3
- Password Reset
    - Send password rest email

Widgets

The widgets are easy to embed and customise with JavaScript applications. The Drupal Module already comes with built in integration. For all user cases there is a single widget, making placement on the page very easy. The widgets are responsive based on the twitter bootstrap grid.

The widgets are based on Ember.js. Widget Overview:

- Login
- Sign-up
- Request password reset
- Profile
- Confirmation
- Password reset

Compatibility

Ember.js uses JavaScript prototype Extensions for Array, String and Function. To avoid compatibility issues in custom JavaScript, do not use incompatible JavaScript code.

Installation

Configuration

The usermanager uses external Service/APIs, which need to be configured.

The minimal required configurations for the usermanager are done by ENVIROMENT variables.

usermanager.env file for docker run

```
#usermanager.env

HARBOURMASTER_USER_KEY=<enter 20 hex char># e.g. 32be04fb9495229f3e4f
HARBOURMASTER_USER_SECRET_KEY=<enter a long password>
HARBOURMASTER_TENANT=thunder

EMAIL_FROM=<Name name@thunder.dev> #send the emails from this address. Format see https://git
EMAIL_SMTP_HOST=<host> #is the hostname or IP address to connect to (defaults to 'localhost')
EMAIL_SMTP_PORT=<port> #is the port to connect to (defaults to 25 or 465)
EMAIL_SMTP_USER=<user> #smtp server user
EMAIL_SMTP_PASS=<password> #smtp server password
EMAIL_SMTP_SECURE=<true|false> #if true the connection will only use TLS. If false (the defau
EMAIL_SMTP_IGNORE_TLS=<true|false> #if this is true and secure is false, TLS will not be used

SSO_COOKIE_DOMAIN=.thunder.dev #the domain where the sso cookie lives
SSO_COOKIE_NAME=token  #the name of the SSO cookie
SSO_WIDGET_CORS_HTTP_HOSTS=http://www.thunder.dev #add your Drupal domain here, comma separat

CROSS_DOMAIN_LOGIN_BASE_URL=http://www.thunder.local:9080/hms_cross_domain_login  #add the cr


BASE_URL=http://usermanager.thunder.dev #url where the usermanager will be publicly accessibl
FALLBACK_URL=http://www.thunder.dev/ #fallback to this url. It is recommended to use the thun
```

Docker Run

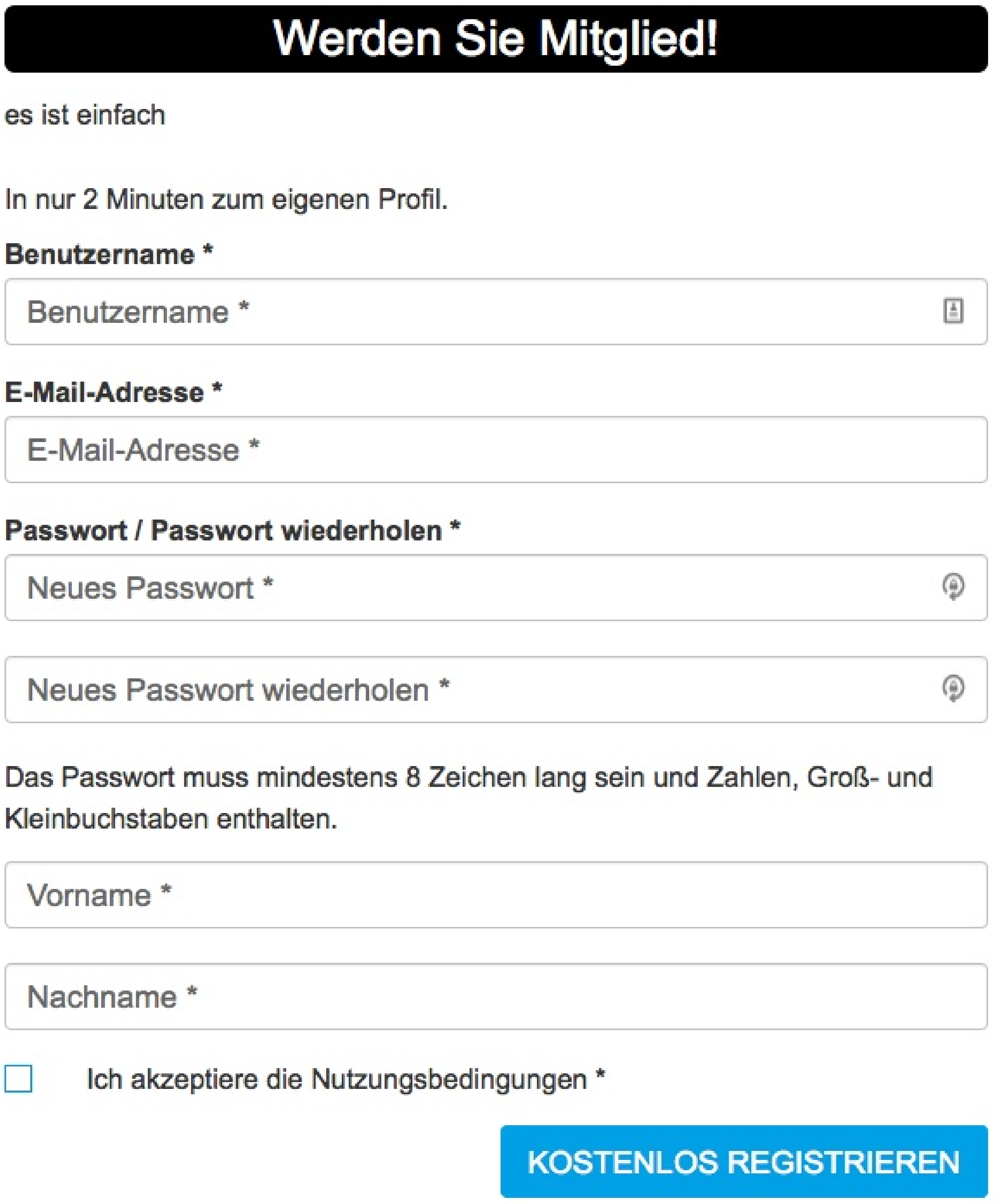
The following command requires a running Harbourmaster and redis container to link to.

```
docker run --name usermanager -p 80:80 --link thunder-harbourmaster:harbourmaster --link redi
```

Changing UI Text

Every Text which get displayed to the user in the frontend widget can be customized. Each UI text is individually addressable in a hierarchical JSON object notation, unlike Drupal, English is not the default text.

```
"signup": {
  "title": {
    "signup": "Werden Sie Mitglied!",
    }
  }
}
```


Werden Sie Mitglied!

es ist einfach

In nur 2 Minuten zum eigenen Profil.

Benutzername *

Benutzername *

E-Mail-Adresse *

E-Mail-Adresse *

Passwort / Passwort wiederholen *

Neues Passwort *

Neues Passwort wiederholen *

Das Passwort muss mindestens 8 Zeichen lang sein und Zahlen, Groß- und Kleinbuchstaben enthalten.

Vorname *

Nachname *

☐ Ich akzeptiere die Nutzungsbedingungen *

Drupal local Module Interface Translation

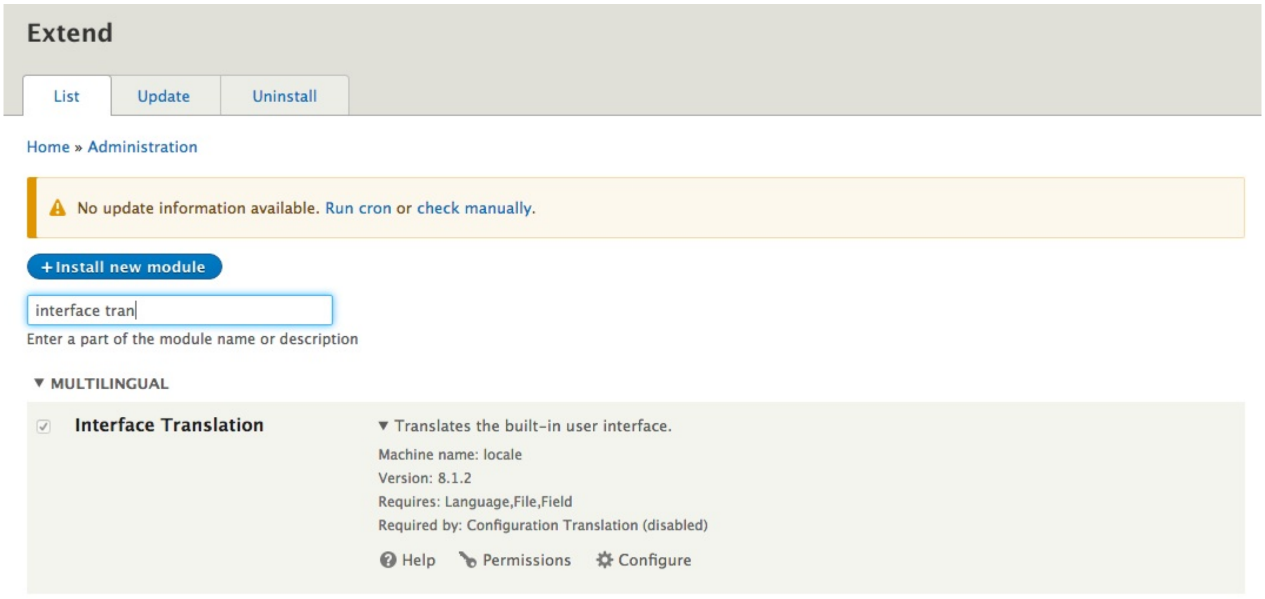
The Harbourmaster Drupal Module utilizes the option to use Drupal translations in JavaScript https://www.drupal.org/node/323109

The Drupal Module contains a de.po and a hms.pot file which can be used to translate all UI text elements.

All translation strings are prefixed with hms-widget e.g. `hms-widget.signup.title.signup`

To be able to use this you need:

1. to install the **Interface Translation** https://www.drupal.org/documentation/modules/locale


Extend

List | Update | Uninstall

Home » Administration

No update information available. Run cron or check manually.

+ Install new module

interface tran

Enter a part of the module name or description

▼ MULTILINGUAL

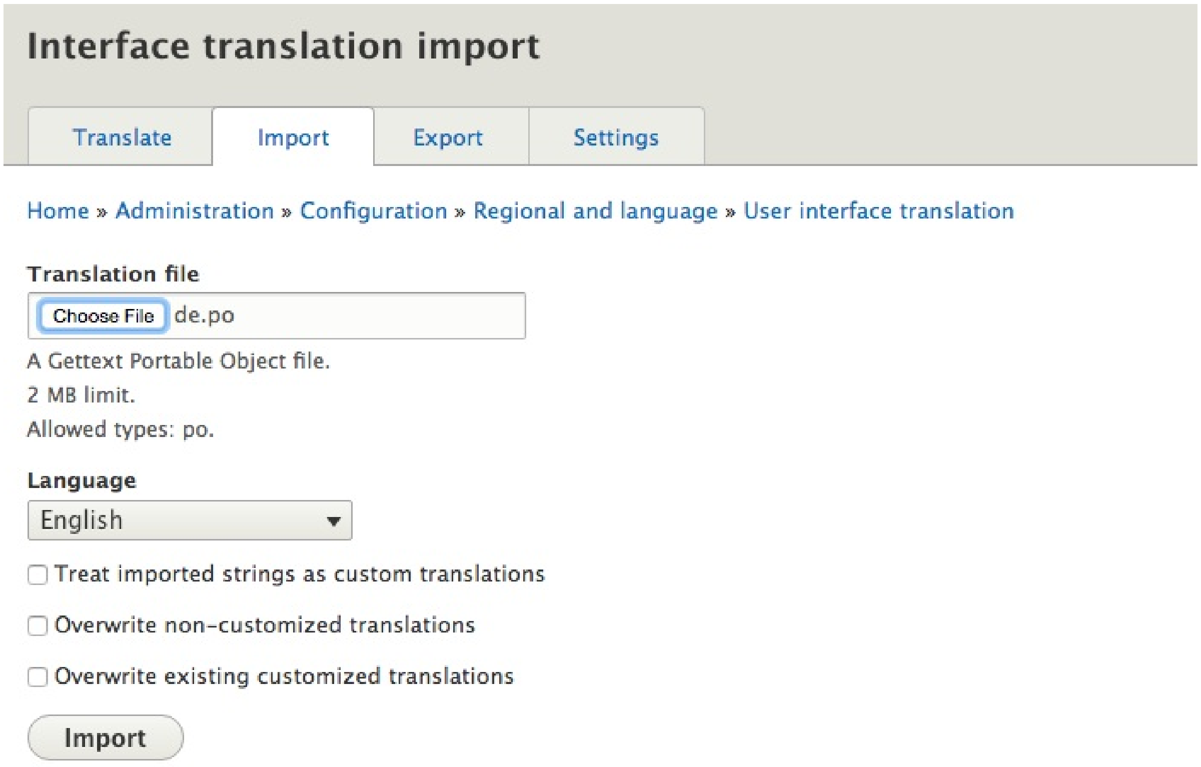
Interface Translation

▼ Translates the built-in user interface.

Machine name: locale
Version: 8.1.2
Requires: Language,File,Field
Required by: Configuration Translation (disabled)

Help | Permissions | Configure

1. Upload the de.po file from the Harbourmaster Drupal Module:


Interface translation import

Translate | Import | Export | Settings

Home » Administration » Configuration » Regional and language » User interface translation

Translation file

Choose File de.po

A Gettext Portable Object file.
2 MB limit.
Allowed types: po.

Language

English

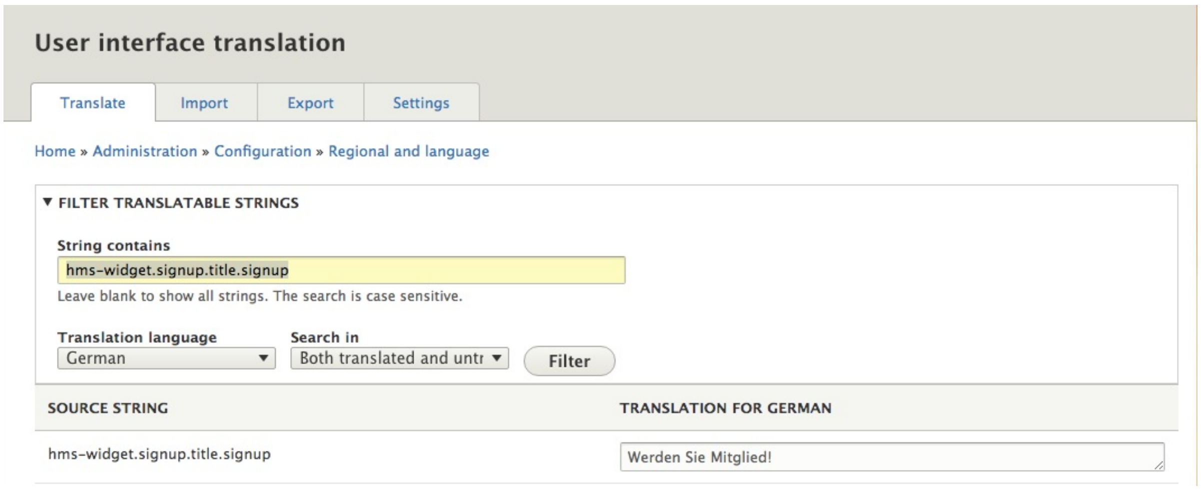
☐ Treat imported strings as custom translations

☐ Overwrite non-customized translations

☐ Overwrite existing customized translations

Import

2. Search UI Text and edit:


User interface translation

Translate | Import | Export | Settings

Home » Administration » Configuration » Regional and language

▼ FILTER TRANSLATABLE STRINGS

String contains: hms-widget.signup.title.signup

Leave blank to show all strings. The search is case sensitive.

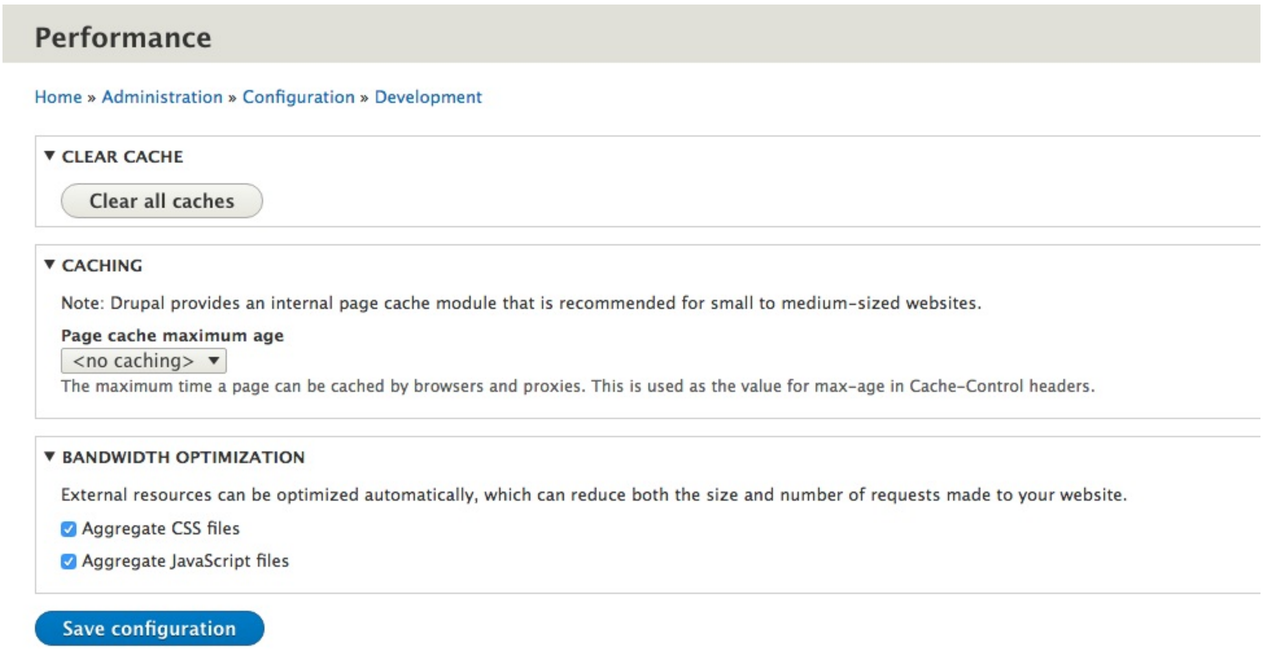
Translation language: German

Search in: Both translated and untr

Filter

SOURCE STRING	TRANSLATION FOR GERMAN
hms-widget.signup.title.signup	Werden Sie Mitglied!

3. Flush Drupal cache and reload UI:


Performance

Home » Administration » Configuration » Development

▼ CLEAR CACHE

Clear all caches

▼ CACHING

Note: Drupal provides an internal page cache module that is recommended for small to medium-sized websites.

Page cache maximum age: <no caching>

The maximum time a page can be cached by browsers and proxies. This is used as the value for max-age in Cache-Control headers.

▼ BANDWIDTH OPTIMIZATION

External resources can be optimized automatically, which can reduce both the size and number of requests made to your website.

- [x] Aggregate CSS files
- [x] Aggregate JavaScript files

Save configuration

Language file injection

An alternative option is to edit the UI text JSON file and inject it into the docker container.

The injection uses the docker volume feature to mount individual host files. Currently the usermanager expects a de.js file, even if it contains English text. Multiple languages are currently not implemented.

```
./de.js:/app/assets/languages/de.js
```

When editing a de.js file remember to restart the docker container, since the files are compiled when the container starts.

```
$ docker stop thunder-usermanager
$ docker start thunder-usermanager
```

Changing email templates

The Usermanager sends a multipart email with a text and an html version of the text. Therefore, two files are required for every email template. The email client will pick one version of the text based on email client capability and user preferences. The html email in this example is a very basic html, but could be expanded to a much richer html. Multiple languages are currently not implemented.

The email templates are created in the ejs template language. Changing the email text requires the injection of the email template files into the docker container.

The `passwordRecoveryHtml.ejs` and `passwordRecoveryText.ejs` are for sending by email if a user requests a password recovery link. The `registrationConfirmationHtml.ejs` and `registrationConfirmationText.ejs` are for the account registration confirmation email.

```
    ./passwordRecoveryHtml.ejs:/app/views/email/forgotPassword/passwordRecoveryHtml.ejs
    ./passwordRecoveryText.ejs:/app/views/email/forgotPassword/passwordRecoveryText.ejs
    ./registrationConfirmationHtml.ejs:/app/views/email/signup/registrationConfirmationHtml.ejs
  ./registrationConfirmationText.ejs:/app/views/email/signup/registrationConfirmationText.ejs
```

Skinning

The widget basic styling and grid rely on the popular bootstrap CSS framework. The `hms-widget.css` widget css is generated from a set of LESS files. Within the generated stylesheet all widget related parts are prefixed with the `.hms-widget` class declaration to avoid overwriting other parts of the page.

That is because all imported files are called inside the parent class definition are called:

```
.hms-widget {
    @import ...
    @import "white-label.less";
}
```

To change the design of the widget you can use one of the following options depending on your knowledge of web technologies:

Drupal based CSS overwrite

This option describes the adaption of the widget design by replacing predefined values in an existing style sheet.

The relevant file is located inside the Harbourmaster Drupal plugin folder.
https://github.com/valiton/harbourmaster-sso-drupal8-plugin

```
harbourmaster-sso-drupal8-plugin/css/white-label-static.css
```

It includes a basic set of selector classes to tackle most of the visible form fields.

Just play around with the property values (colours, sizes, backgrounds) and when finished, simply flush all caches (available from the Drupal admin menu) before reloading your page.

Usermanager LESS injection

Besides the ability to include a pre-compiled CSS file, we also offer a more advanced option to adapt the look and feel of the widget.

Any technical savvy developer/designer/site builder can customize the white-label.less file to meet the requirements in regards of CI and existing specifications. The injection uses the docker volume feature to mount individual host files

```
 ./white-label.less:/app/assets/styles/less/white-label.less
```

Some basic selector groups have already been included; for further adjustments one can inspect the required DOM nodes in source code view and amend existing class or ID definitions within the style sheet.

You can change default color, size and font definitions in the variables file located at

```
./vars.less:/app/assets/styles/less/vars.less
```

When editing LESS files, remember to restart the docker container, since the files are compiled when the container starts.

```
$ docker start thunder-usermanager
$ docker start thunder-usermanager
```

If you decide to go with the LESS file option you can optionally decide not to use the `white-label-static.css` from the Drupal Module or comment out all unnecessary declarations from the Drupal Module.

Configure email SMPT server

The usermanager uses a SMPT Server to send email. Email will be sent with the nodemailer npm module. The most common SMTP configurations used are exposed as environment variables.

```
EMAIL_FROM=Name name@thunder.dev     #send the mails form this address. Format see https://gi
EMAIL_SMTP_HOST=<host>                #is the hostname or IP address to connect to (defaults
EMAIL_SMTP_PORT=<port>                #is the port to connect to (defaults to 25 or 465)
EMAIL_SMTP_USER=<user>                #smtp server user
EMAIL_SMTP_PASS=<password>            #smtp server password
EMAIL_SMTP_SECURE=<true|false>        #if true the connection will only use TLS. If false (th
EMAIL_SMTP_IGNORE_TLS=<true|false>    #if this is true and secure is false, TLS will not be u
```

Control Center Installation

About

The Control Center is a web application which provides an administration interface for the Harbourmaster SSO. It is provided as a docker container.
https://hub.docker.com/r/valiton/controlcenter/

Configuration

The necessary configuration of the Control Center is defined with environment variables.

```
COOKIE_SECURE=<true|false> #should set the login cookie only on https

REDIRECT_TO_HTTPS=<true|false> #should redirect to https

HARBOURMASTER_API_URL=<harbourmaster_api_base_url_v1> #Base URL of the Harbourmaster API incl
```

Docker run

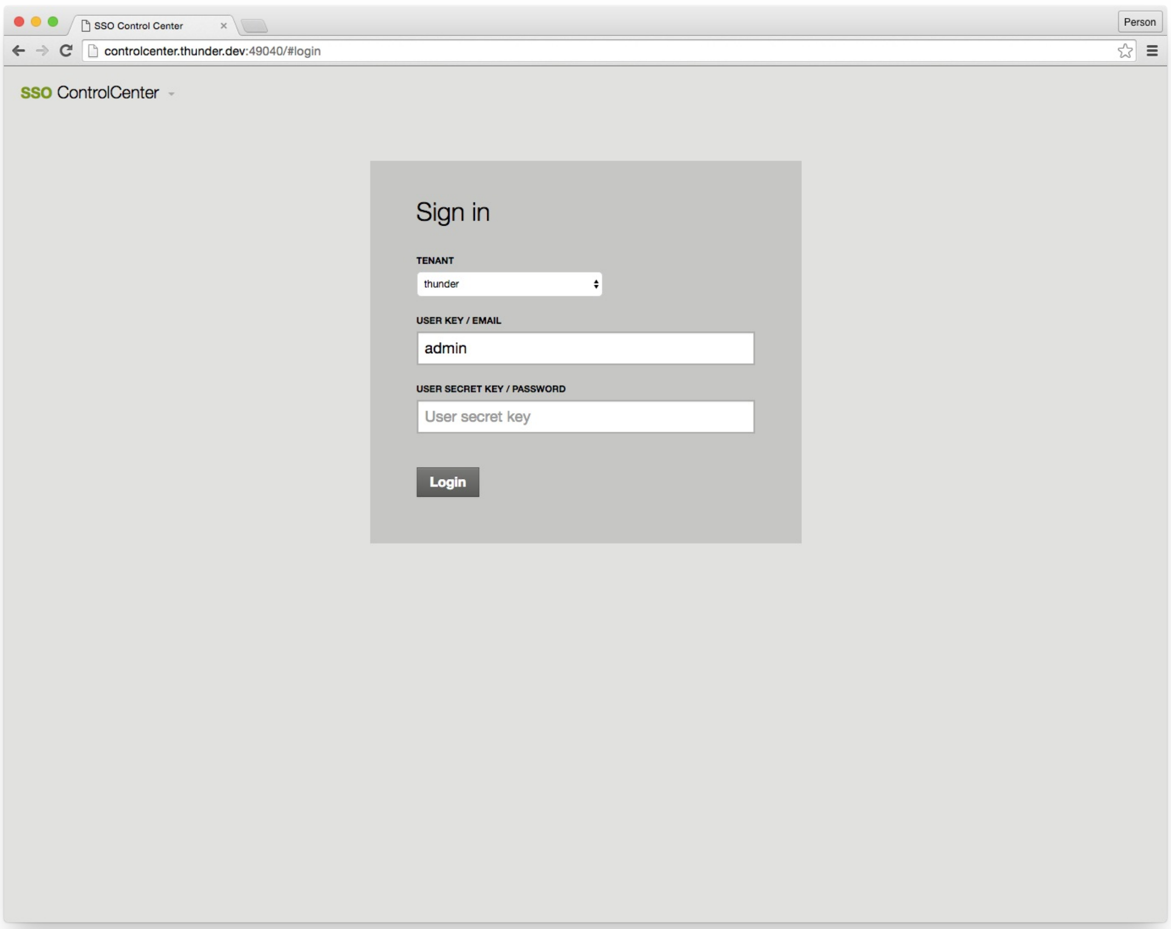
The quick start guide provides a preconfigured docker-compose file. Quick Start Guide

```
docker run --name thunder-controlcenter -e COOKIE_SECURE=false -e REDIRECT_TO_HTTPS=false -e
```

Control Center user documentation

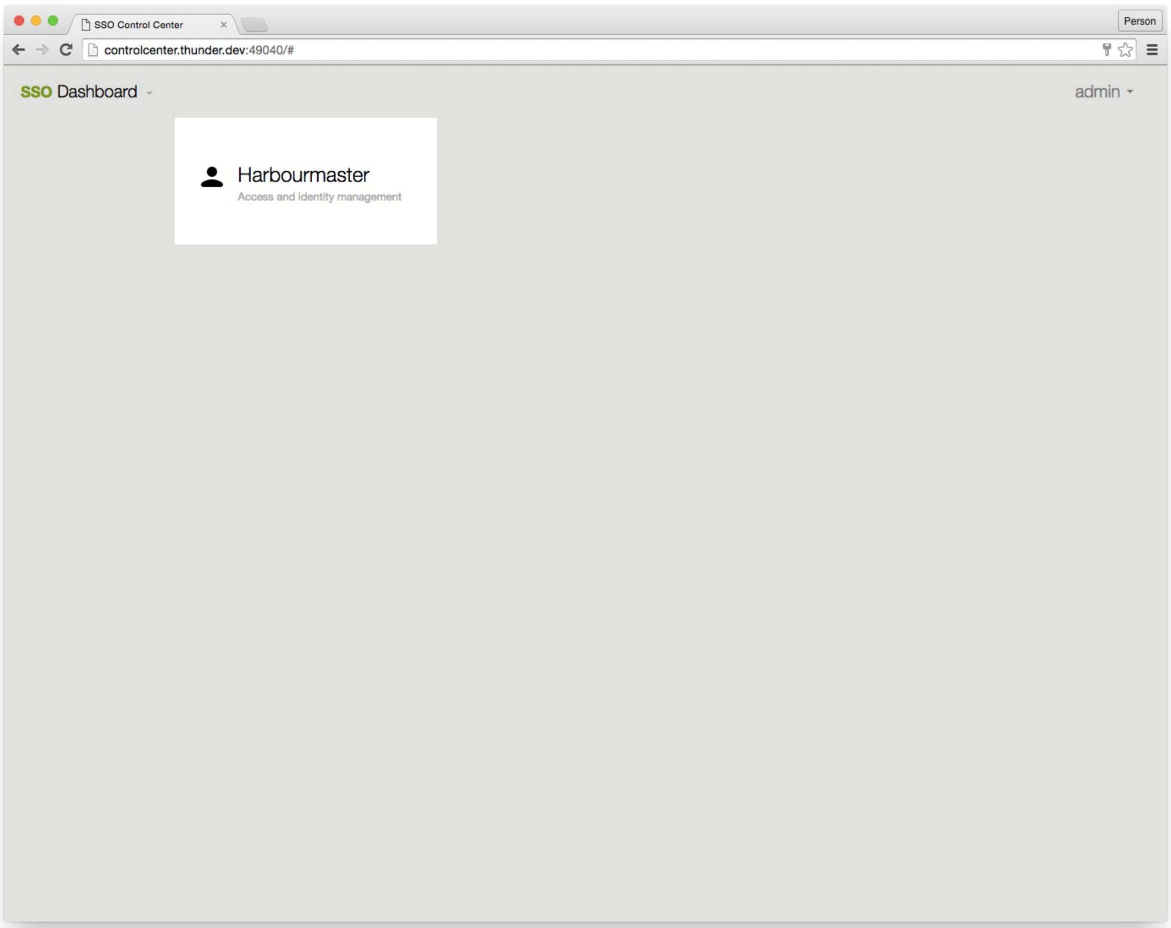
Login

Log in with username/password or with user key and user secret key. The user must have the permission **cc:accessUI**. The admin user created in the seed script has complete permission.



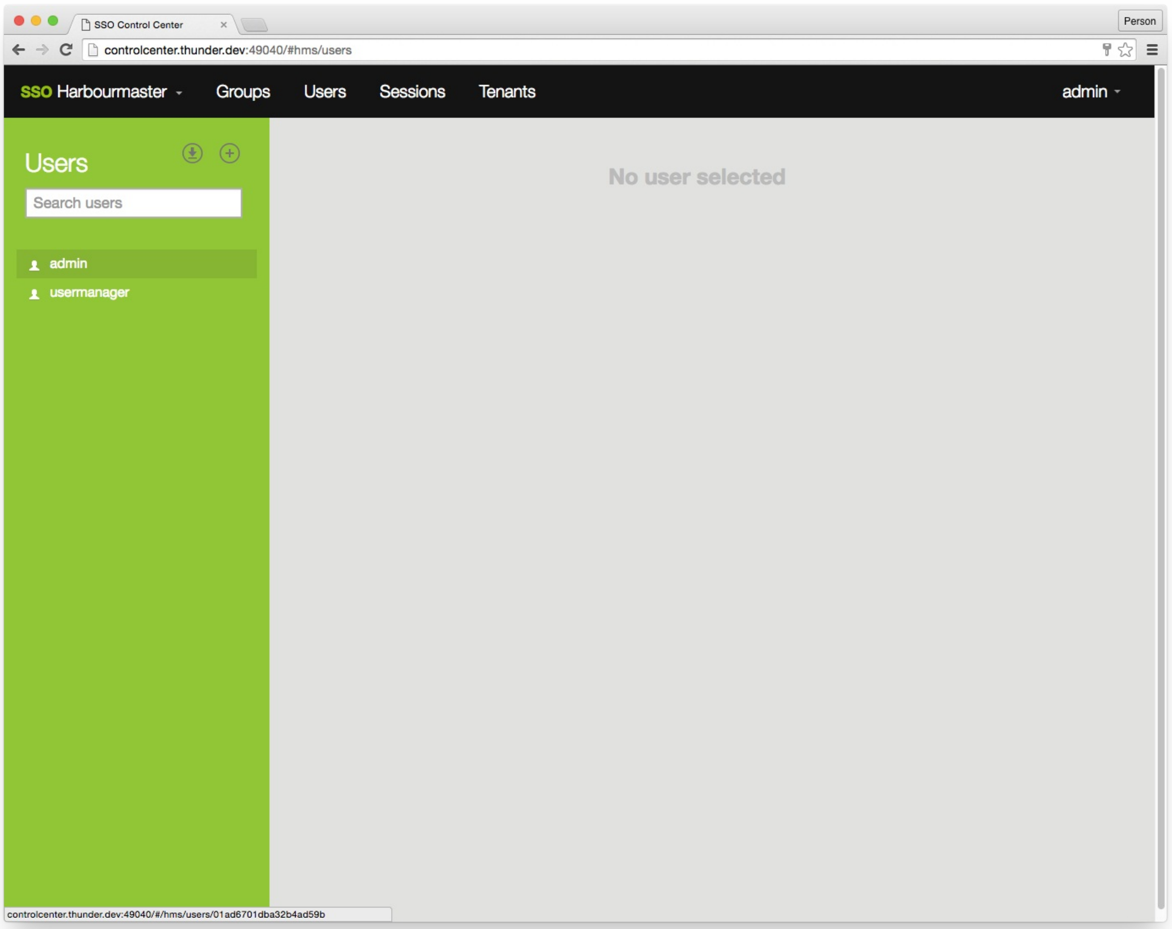
Control Center Dashboard

On the Control Center dashboard, select Harbourmaster to manage users.


SSO Control Center
Person
controlcenter.thunder.dev:49040/#
SSO Dashboard
admin
Harbourmaster
Access and identity management

List Users

The Control Center lists all users function in paginated form on the left hand side. The list is searchable.

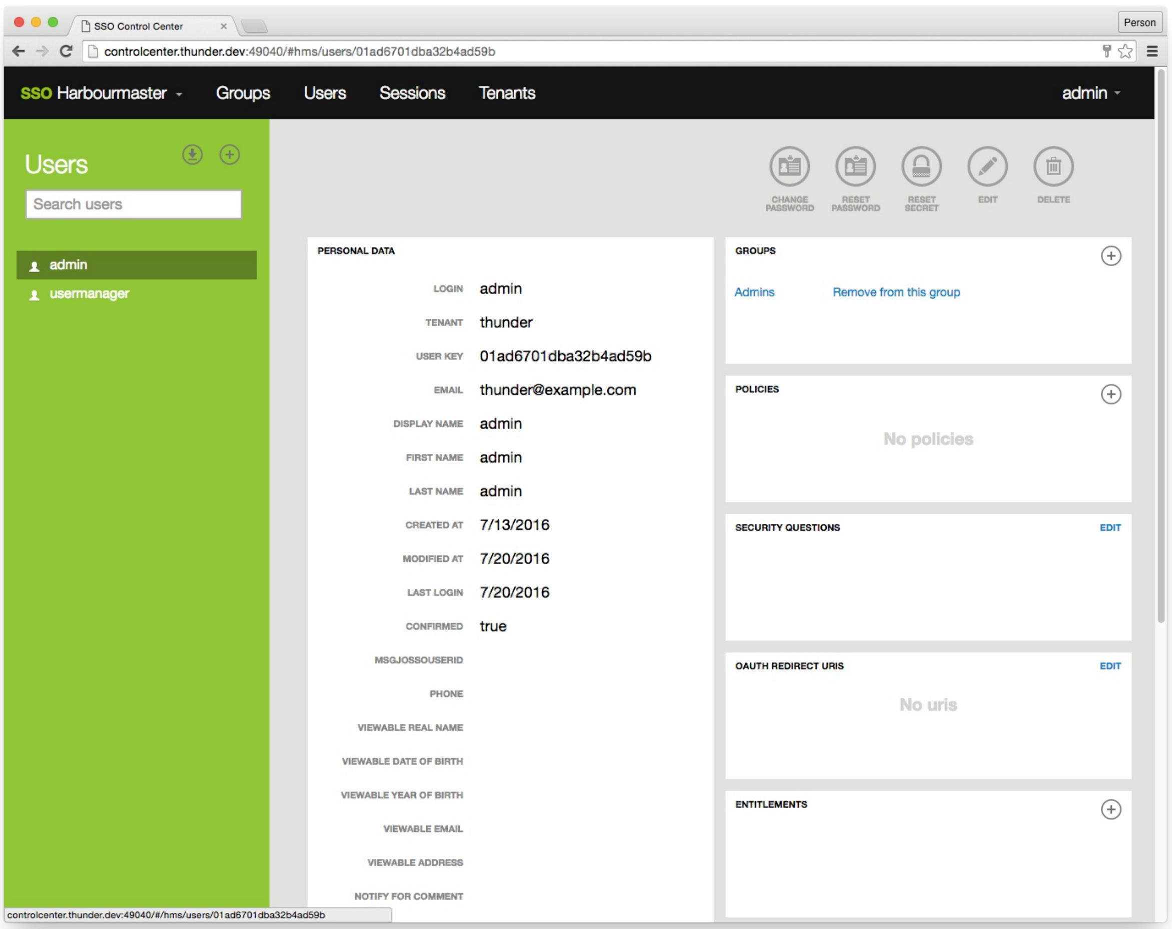


User Details

The Control Center shows all user details when clicking on a user from the list. On the user detail page the following admin actions can be done.

- Change password for the user (requires knowledge of the user password)
- Reset password (assign a new randomly generated password to the user, password will be displayed once)
- Reset secret key (assign a new randomly generated secret key to the user, secret key will be displayed once)
- Edit the user details
- Delete the user
- Add group to User
- Remove group from user

- Add policy to user
- Remove policy from user
- Edit user security questions
- Add entitlement to user
- Remove entitlement from user
- OAuth Callback Whitelist (relevant for OAuth API user)



PERSONAL DATA

LOGIN	admin
TENANT	thunder
USER KEY	01ad6701dba32b4ad59b
EMAIL	thunder@example.com
DISPLAY NAME	admin
FIRST NAME	admin
LAST NAME	admin
CREATED AT	7/13/2016
MODIFIED AT	7/20/2016
LAST LOGIN	7/20/2016
CONFIRMED	true
MSGJOSSOUSERID	
PHONE	
VIEWABLE REAL NAME	
VIEWABLE DATE OF BIRTH	
VIEWABLE YEAR OF BIRTH	
VIEWABLE EMAIL	
VIEWABLE ADDRESS	
NOTIFY FOR COMMENT	

GROUPS

Admins — Remove from this group

POLICIES

No policies

SECURITY QUESTIONS — EDIT

OAUTH REDIRECT URIS — EDIT

No uris

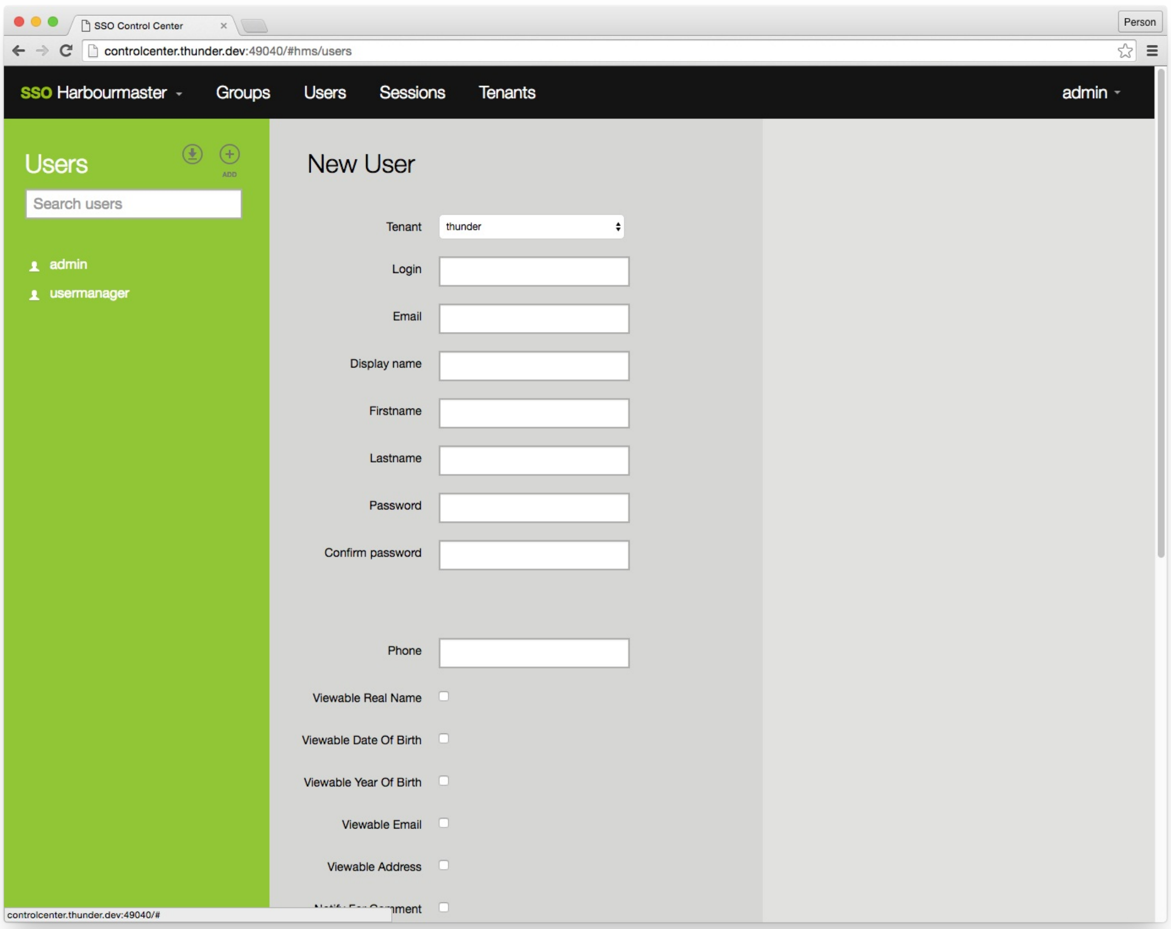
ENTITLEMENTS

controlcenter.thunder.dev:49040/#/hms/users/01ad6701dba32b4ad59b

Create new user

The Control Center create user function has less validation than the usermanager widget used by internet users. e.g. the usermanager validates that the login name is not a email address, the Control Center cannot perform this validation. After a new user has been created, the interface will display a pop-up, which will show that user's secret key which is required for certain API calls.

Important: A new user has no default policies and therefore cannot login; you have to add a group to the new user.

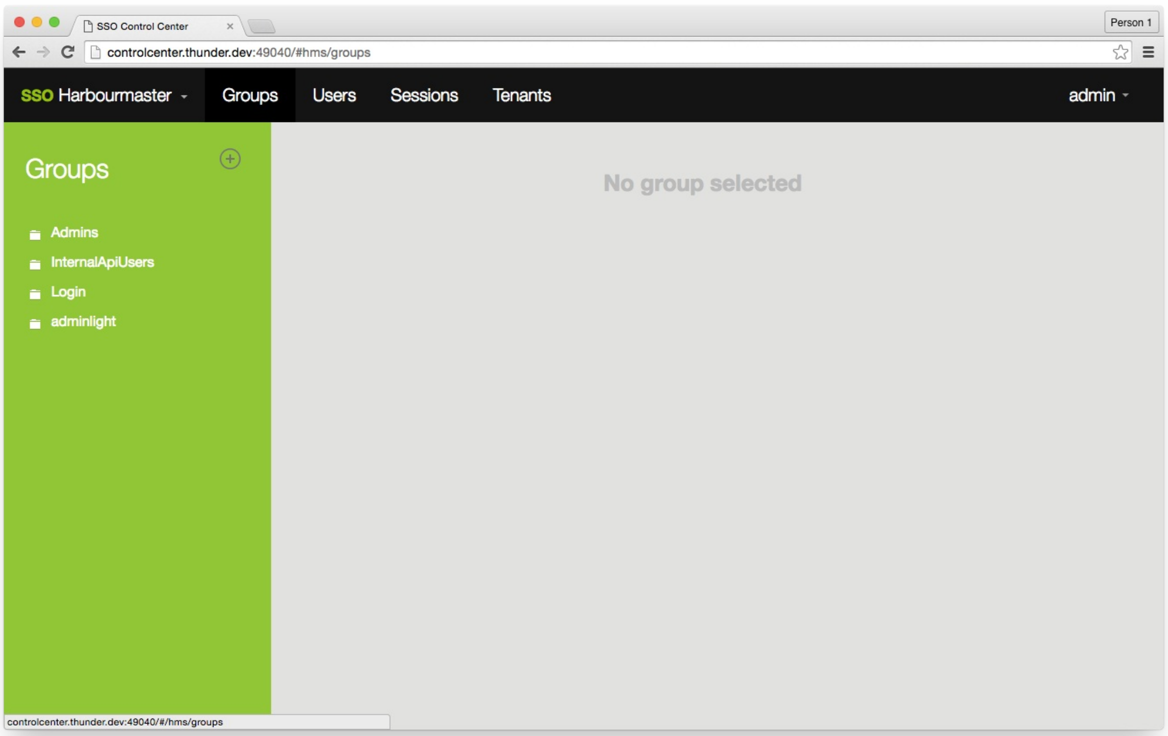


New User

Tenant	thunder
Login	
Email	
Display name	
Firstname	
Lastname	
Password	
Confirm password	
Phone	
Viewable Real Name	☐
Viewable Date Of Birth	☐
Viewable Year Of Birth	☐
Viewable Email	☐
Viewable Address	☐

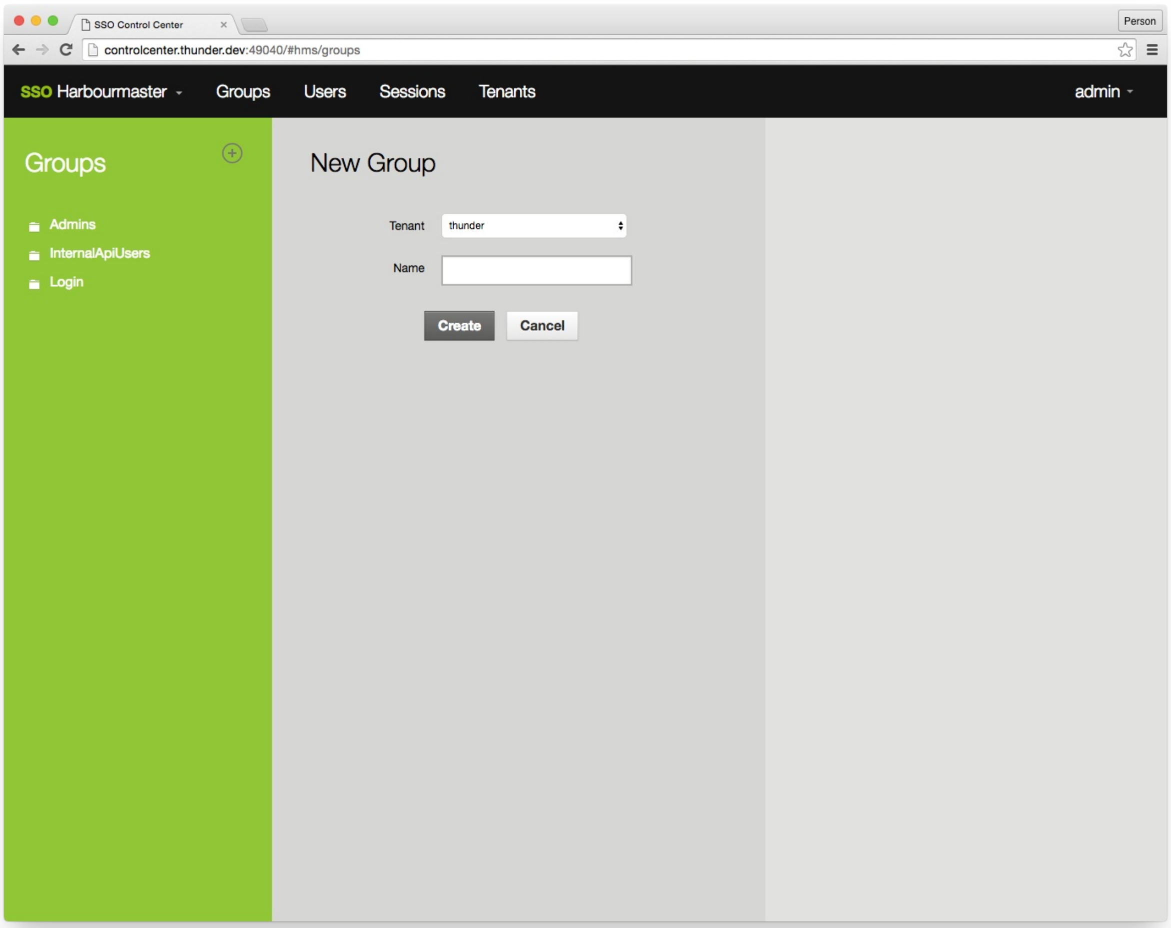
List Groups

The Control Center list all groups function.



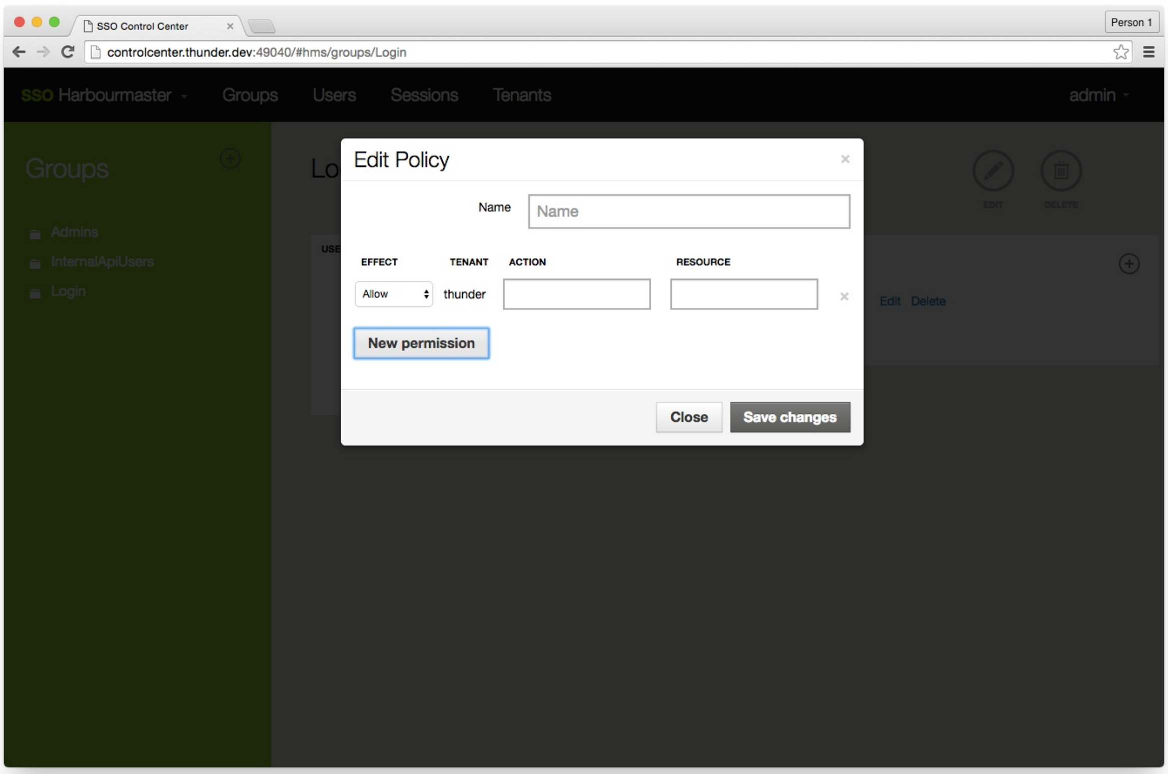
Create new Group

Create a new group for a tenant. Group names should not contain any whitespaces.



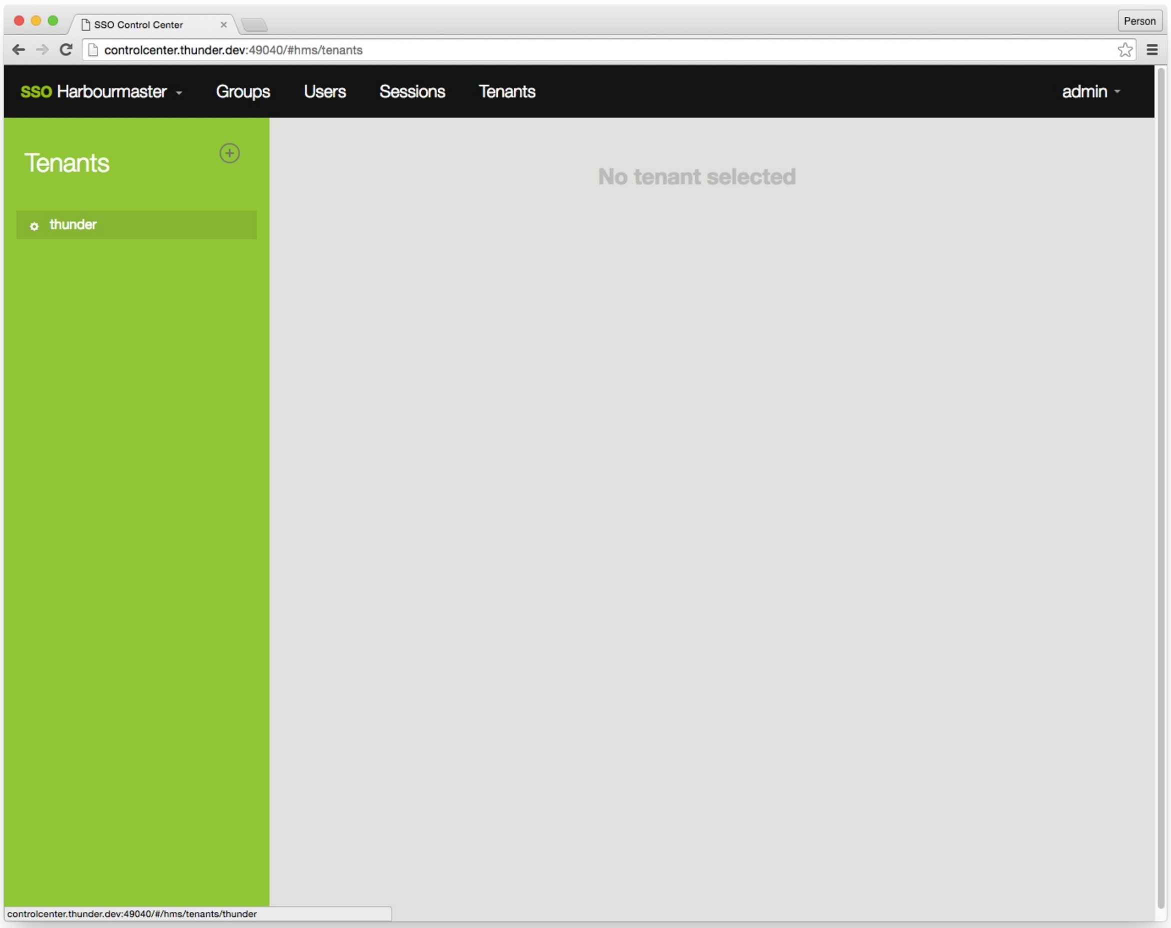
Create new Policy in Group OR User

Create a new policy. The UI represents the policy statements JSON structure as described in the Harbourmaster Policies.



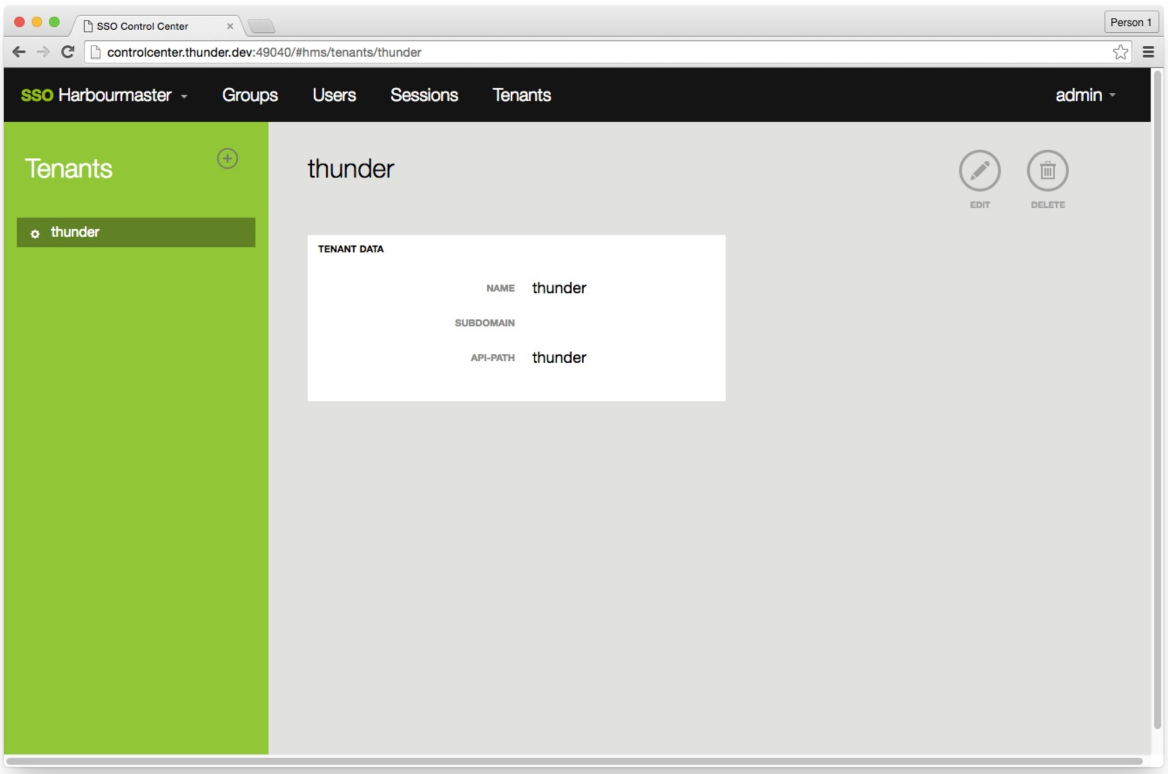
List tenants

List the tenants. All Harbourmaster entities such as users and groups are attached to one and only one tenant. A move from one tenant on other list not supported. In other words, a tenant represents one set of users.



Show tenant

Show the details of a tenant.



thunder

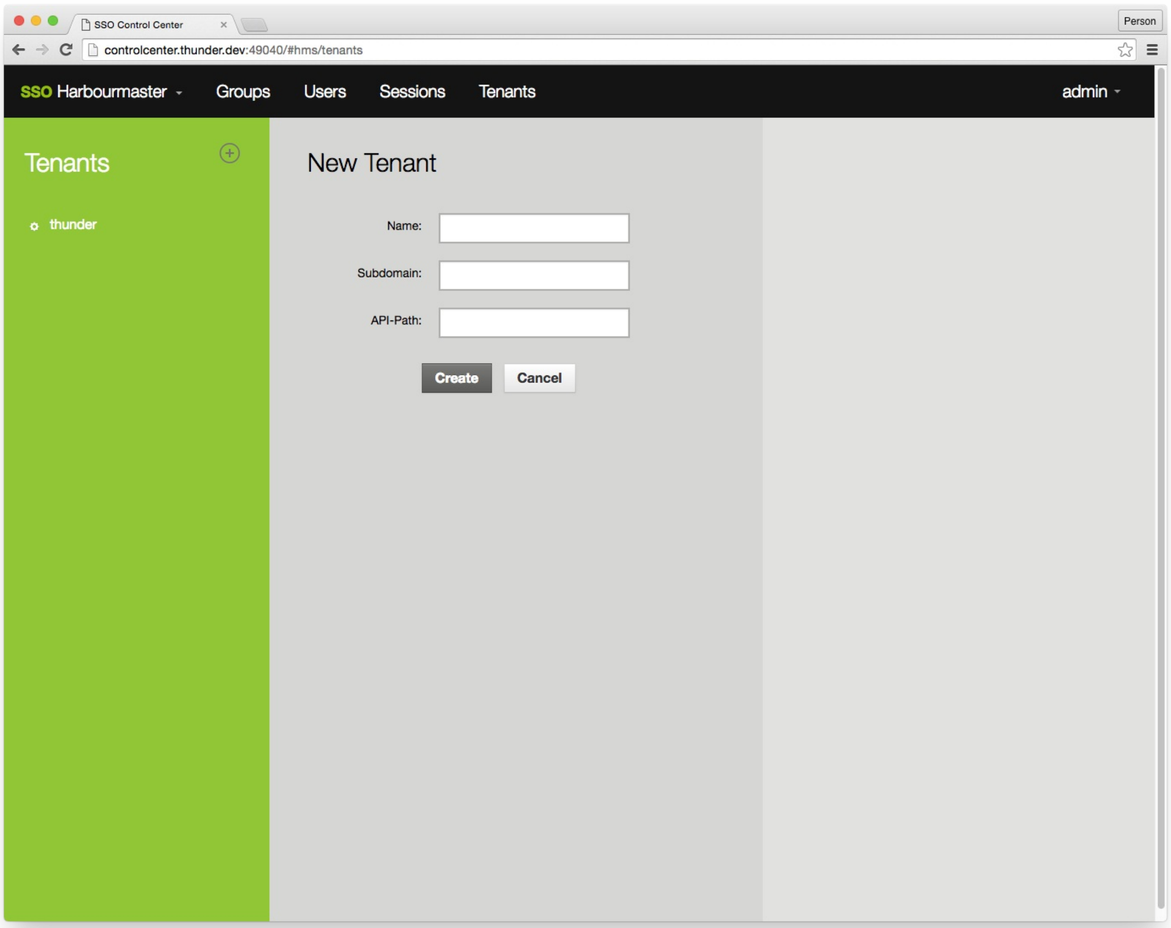
TENANT DATA

NAME	thunder
SUBDOMAIN	
API-PATH	thunder

Create new tenant

Create a new tenant and define the API path, which will be used in the Harbourmaster API, as well as a name. Sub-domain configuration gets ignored at this stage.

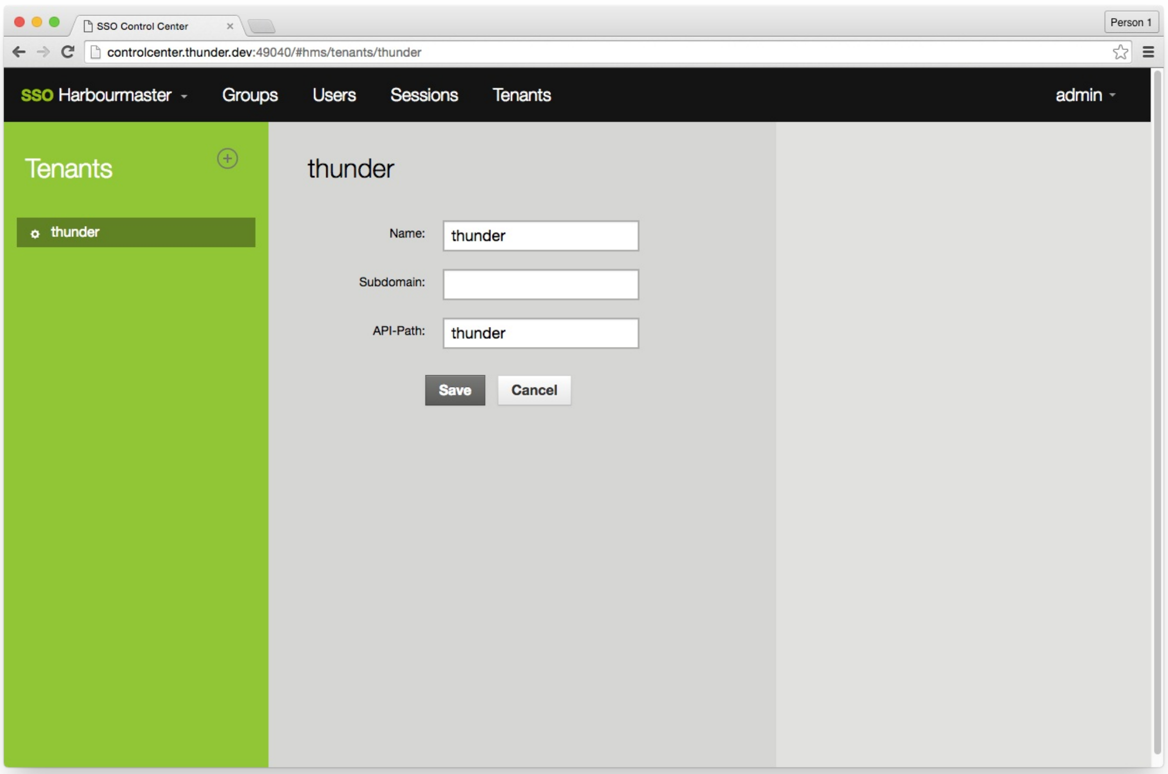
Please note: a new tenant comes with no groups, no policies, no admin user. As a admin user with cross tenant access you have to create all necessary information. The initial tenant is created by the seed script which also creates a basic set of groups and policies.



Edit Tenant

The Control Center allows the editing of a tenant; **please use with caution** as this can break access policies, and API access.

We do not recommend editing a tenant.



thunder

Name:	thunder
Subdomain:	
API-Path:	thunder

Drupal8 / Thunder

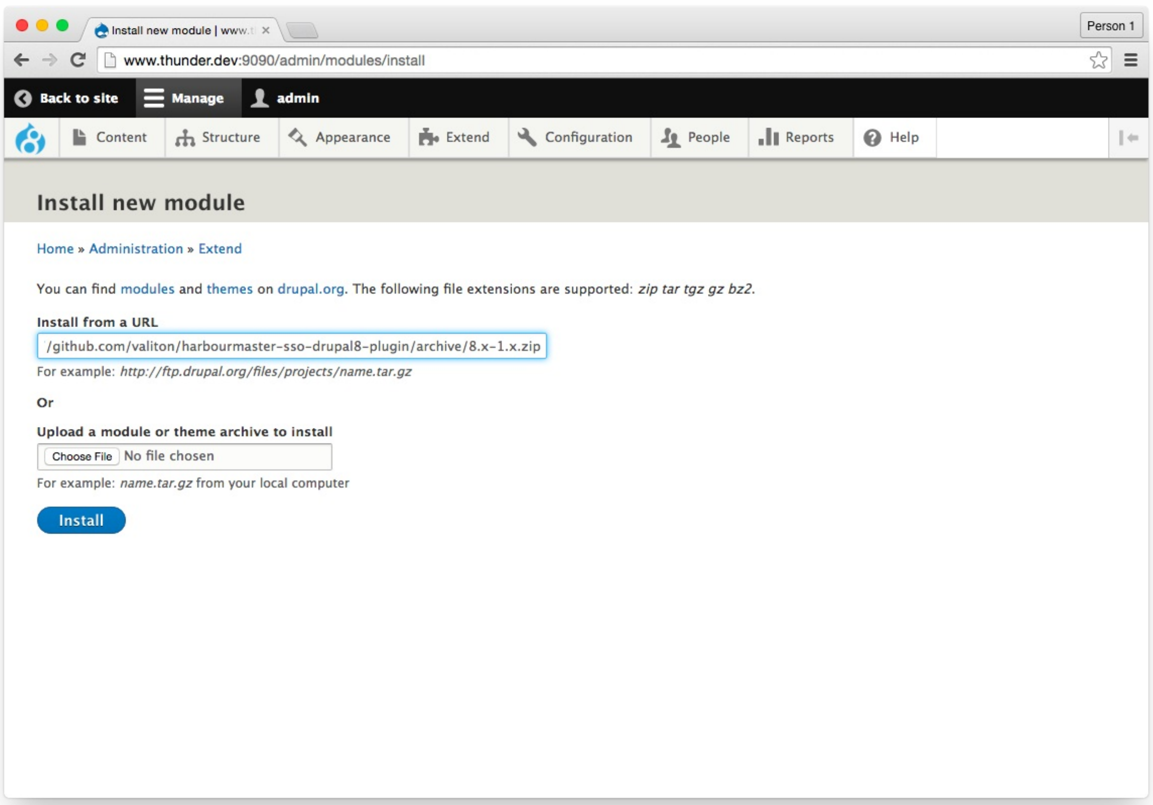
Install Harbourmaster SSO plugin

Github Link https://github.com/valiton/harbourmaster-sso-drupal8-plugin

zip to install: https://github.com/valiton/harbourmaster-sso-drupal8-plugin/archive/8.x-1.x.zip

The Module is also on Packagist https://packagist.org/packages/valiton/harbourmaster

Install the module



Install new module

Home » Administration » Extend

You can find modules and themes on drupal.org. The following file extensions are supported: *zip tar tgz gz bz2*.

Install from a URL

/github.com/valiton/harbourmaster-sso-drupal8-plugin/archive/8.x-1.x.zip

For example: *http://ftp.drupal.org/files/projects/name.tar.gz*

Or

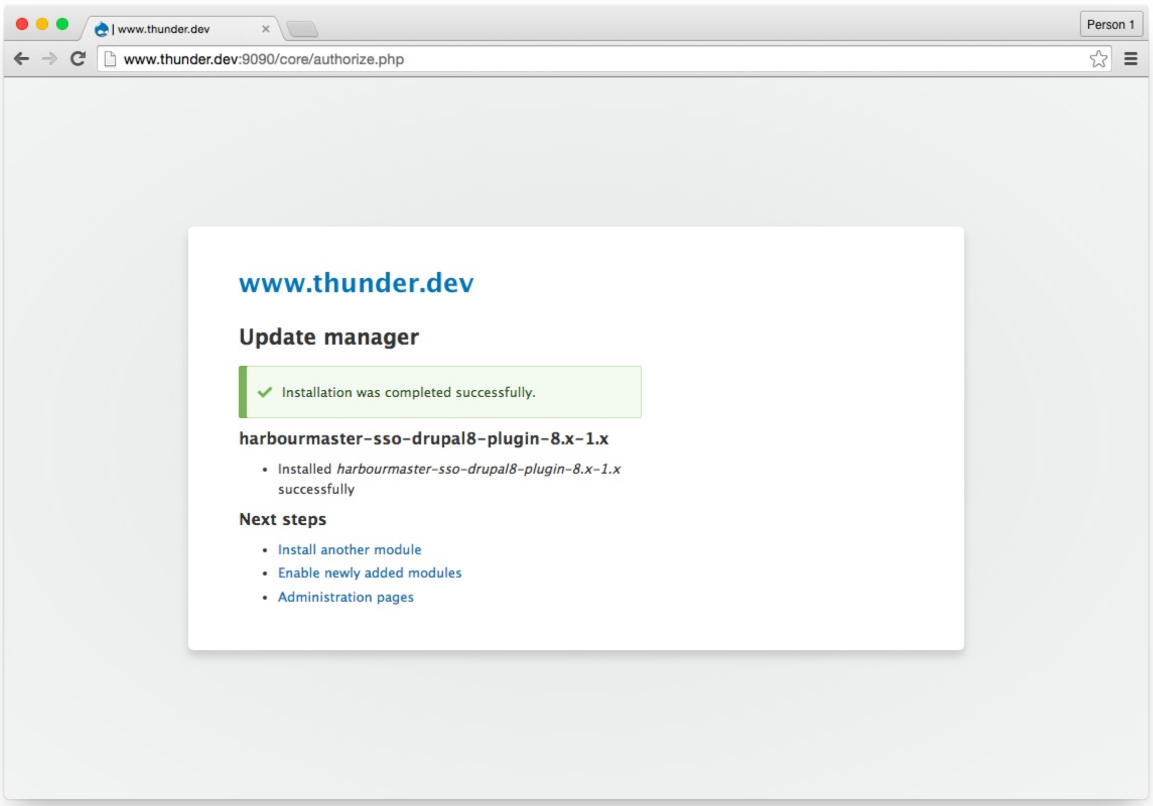
Upload a module or theme archive to install

Choose File No file chosen

For example: *name.tar.gz* from your local computer

Install

Successfully installed



www.thunder.dev

Update manager

✓ Installation was completed successfully.

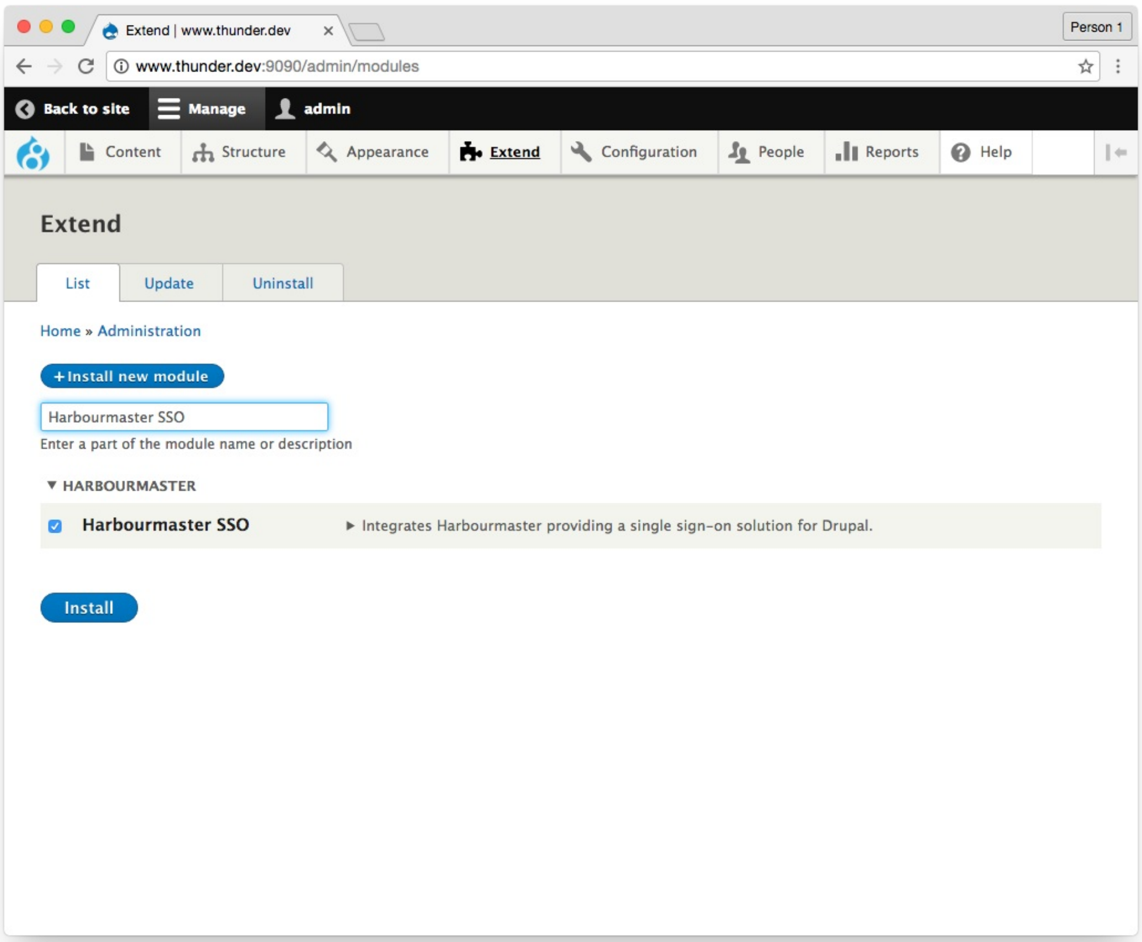
harbourmaster-sso-drupal8-plugin-8.x-1.x

- Installed *harbourmaster-sso-drupal8-plugin-8.x-1.x* successfully

Next steps

- Install another module
- Enable newly added modules
- Administration pages

Activate the module



Extend

List | Update | Uninstall

Home » Administration

+Install new module

Harbourmaster SSO

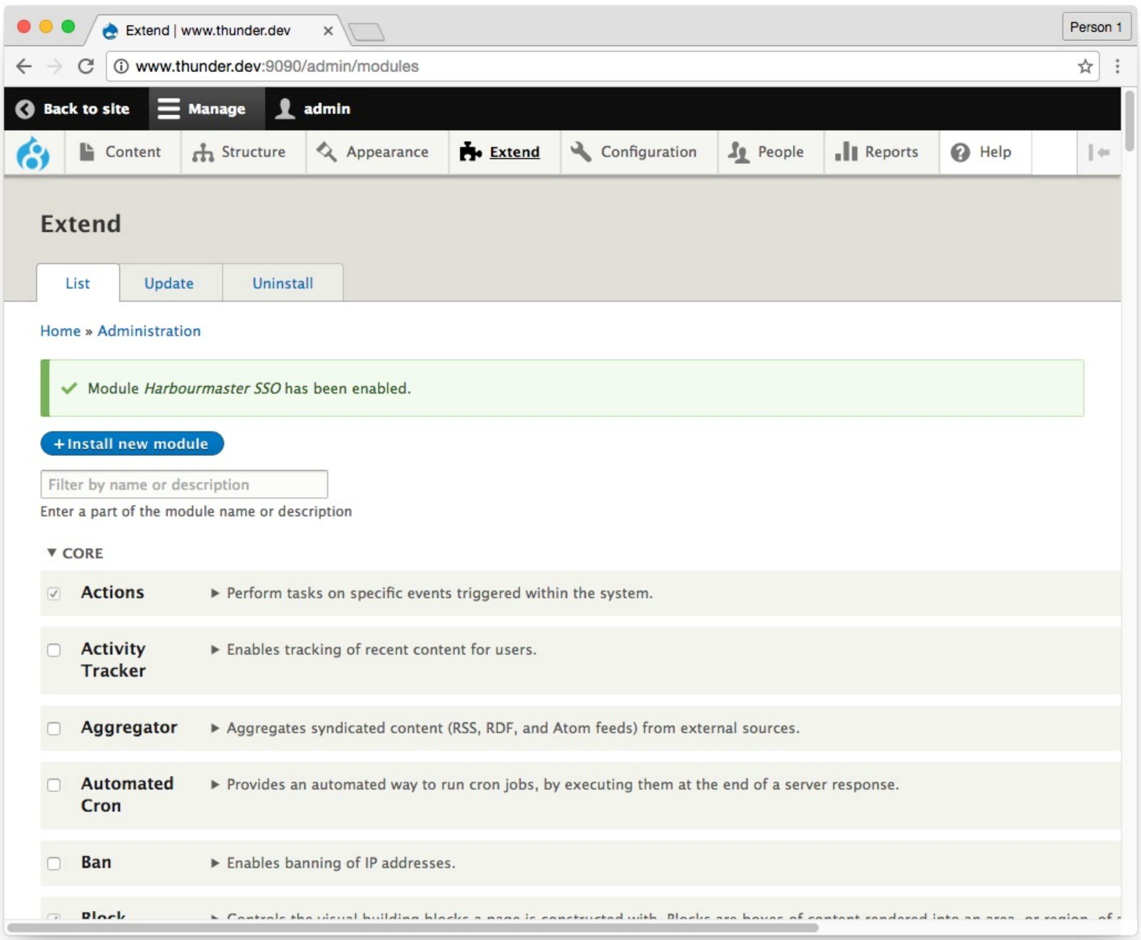
Enter a part of the module name or description

▼ HARBOURMASTER

✓	Harbourmaster SSO	▶ Integrates Harbourmaster providing a single sign-on solution for Drupal.

Install

Successfully activated



Extend

Module *Harbourmaster SSO* has been enabled.

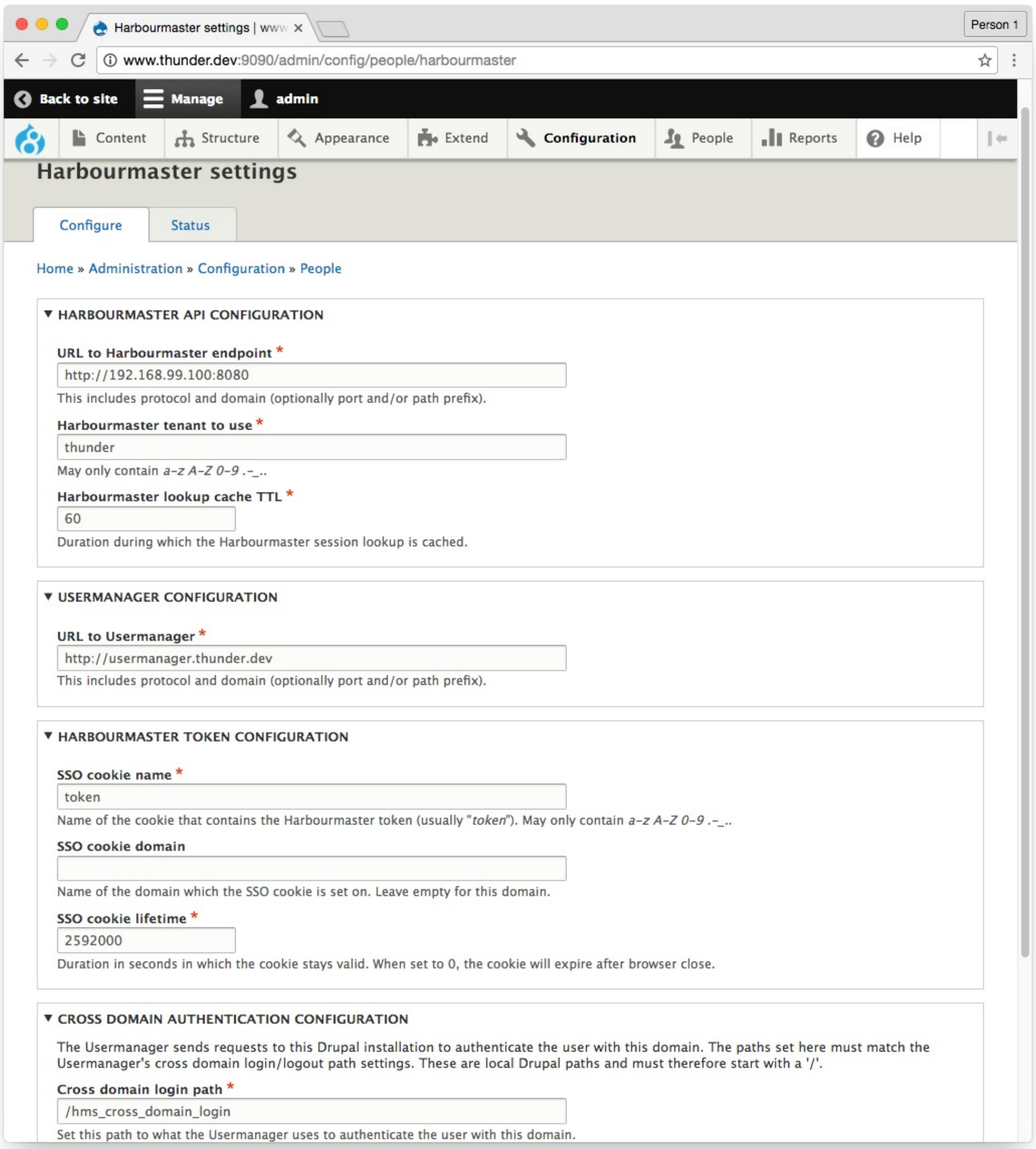
Enter a part of the module name or description

▼ CORE

✓	Actions	▸ Perform tasks on specific events triggered within the system.
☐	Activity Tracker	▸ Enables tracking of recent content for users.
☐	Aggregator	▸ Aggregates syndicated content (RSS, RDF, and Atom feeds) from external sources.
☐	Automated Cron	▸ Provides an automated way to run cron jobs, by executing them at the end of a server response.
☐	Ban	▸ Enables banning of IP addresses.

Configuration

- Configure the module



Harbourmaster settings

Configure | Status

Home » Administration » Configuration » People

▼ HARBOURMASTER API CONFIGURATION

URL to Harbourmaster endpoint *
http://192.168.99.100:8080
This includes protocol and domain (optionally port and/or path prefix).

Harbourmaster tenant to use *
thunder
May only contain *a-z A-Z 0-9 .-_.*.

Harbourmaster lookup cache TTL *
60
Duration during which the Harbourmaster session lookup is cached.

▼ USERMANAGER CONFIGURATION

URL to Usermanager *
http://usermanager.thunder.dev
This includes protocol and domain (optionally port and/or path prefix).

▼ HARBOURMASTER TOKEN CONFIGURATION

SSO cookie name *
token
Name of the cookie that contains the Harbourmaster token (usually "*token*"). May only contain *a-z A-Z 0-9 .-_.*.

SSO cookie domain

Name of the domain which the SSO cookie is set on. Leave empty for this domain.

SSO cookie lifetime *
2592000
Duration in seconds in which the cookie stays valid. When set to 0, the cookie will expire after browser close.

▼ CROSS DOMAIN AUTHENTICATION CONFIGURATION

The Usermanager sends requests to this Drupal installation to authenticate the user with this domain. The paths set here must match the Usermanager's cross domain login/logout path settings. These are local Drupal paths and must therefore start with a '/'.

Cross domain login path *
/hms_cross_domain_login
Set this path to what the Usermanager uses to authenticate the user with this domain.

The Drupal Module needs to be configured to communicate to Harbourmaster the API and includes the widget js from the usermanager. Once the Drupal module has been correctly configured, users will be able to register and log in using `<YOUR_DRUPAL_URL>/harbourmaster/login`, e.g. `www.thunder.dev/harbourmaster/login`.

HMS API CONFIGURATION

- URL to Harbourmaster endpoint *Includes protocol and domain (optionally port and/or path prefix). This Url is used for server to server communication, it is not used by the browser.* e.g. "http://harbourmaster.thunder.dev:8080" Required for the multi domain setup. Note if you user Docker Native for Mac use the IP address of your computer e.g. 192.x.x.x

- Harbourmaster tenant to use *May only contain a-z A-Z 0-9 .* e.g. "thunder"
- HMS lookup cache TTL *Duration during which the HMS session lookup is cached.* e.g. "60"

USERMANAGER CONFIGURATION

- URL to usermanager *Includes protocol and domain (optionally port and/or path prefix). This Url will be used to load the Usermanger Widget, it is only used by the web browser, not for server to server communication.* e.g. "http://usermanager.thunder.dev"

HMS TOKEN CONFIGURATION

- SSO cookie name *Name of the cookie that contains the HMS token (usually "token"). May only contain a-z A-Z 0-9 .-\..SSO _* e.g. "token"
- Cookie domain

  Name of the domain which the SSO cookie is set on. e.g. ".thunder.dev"

- SSO cookie lifetime *Duration in seconds in which the cookie stays valid. When set to 0, the cookie will expire after browser close.*

  e.g. 2592000 (30 days)

CROSS DOMAIN AUTHENTICATION CONFIGURATION

- Cross domain login

  Set this path to what the Usermanager uses to authenticate the user with this domain. e.g. /hms_cross_domain_login

- Cross domain logout

  Set this path to what the Usermanager uses to invalidate the user's session with this domain. e.g. /hms_cross_domain_logout

Harbourmaster Freemium License Terms & Conditions

1. License

1.1 Subject to the terms and conditions hereof, Valiton GmbH, Arabellastr. 21, D-81925 München ("Valiton") hereby grants to Licensee a royalty-free, limited, non-transferable, non-sublicensable, non-exclusive license to use the computer program acquired under this agreement (including all versions, features, applications and modules thereto the "Harbourmaster Freemium Software") only for Licensee' own business purposes (but not for third party services). Limitations of the Harbourmaster Freemium Software are specified in Annex 1.

1.2 The software is provided to Licensee in the form of an image file. In order to use the image, it is necessary to accept the terms of this Harbourmaster Freemium License. This is achieved by specifying the environment variable LICENSE equal to "accept" when running the image. Licensee can also view the license terms by setting this variable to "view". Failure to set the variable will result in the termination of the container with a usage statement. By setting the variable to "accept" and running the image, Licensee accepts and agrees to the terms of this Harbourmaster Freemium License (the "Agreement").

1.3 Licensee accepts and agrees that, under this Agreement, Valiton has no obligation to provide any maintenance service or support with regard to the Harbourmaster Freemium Software.

2. Proprietary Rights

2.1 The Harbourmaster Program furnished under this Agreement is licensed, not sold, to Licensee. Valiton possesses all right, title and interest in and to the Harbourmaster Program and Documentation, and Licensee acknowledges that it receives no such right, title or interest under this Agreement except for the specifically described limited right of use.

2.2 Licensee acknowledges that Valiton is the sole and exclusive owner of the trademarks relating to the Harbourmaster Freemium Software. Licensee shall not attack and\/or challenge Valitons intellectual property rights or raise corresponding claims against Valiton.

2.3 Licensee acknowledges that the Programs embody Valiton's proprietary and confidential information, and Licensee agrees not to disclose or disseminate any Program or Documentation, in whole or in part, or copies in any form, to any person other than Licensee’s own employees, consultants and contractors, provided such persons have a need to know to perform their assigned tasks.

2.4 Except as expressly permitted hereunder, Licensee shall not and shall not permit or authorize any third party to: (i) reverse engineer, decompile, disassemble or otherwise attempt to discover the source code, object code or underlying structure, ideas or algorithms of the Harbourmaster Freemium Software; (ii) circumvent or provide a method to circumvent any technological protection measures in the Harbourmaster Freemium Software, (iii) modify, translate or create derivative works based on the Harbourmaster Freemium Software; (iv) copy (except for archival purposes), rent, lease, distribute, pledge, assign or otherwise transfer or allow any lien, security interest or other encumbrance on the Harbourmaster Freemium Software; (v) use the Harbourmaster Freemium Software for timesharing or otherwise for the benefit of a third party; (vi) remove or obscure any proprietary notices or labels of Valiton or its suppliers on any of the Harbourmaster Freemium Software.

3. System Requirements and other limitations

3.1 Licensee may use the Harbourmaster Freemium Software only in accordance with the system requirements set forth in Annex 2 (the „System Requirements").

3.2 Under this Agreement, Licensee’s use of the Harbourmaster Freemium Software is limited as follows:

- Licensee may not use the Harbourmaster Freemium Software in the following industries: Banking, Insurance or any other financial services; Medical. In case of doubt, Licensee is obliged to consult with Valiton in order to ascertain whether the Harbourmaster Freemium Software may be used.
- Licensee may not operate the Harbourmaster Freemium Software unless the Harbourmaster API is protected against unauthorized access applying accepted industry standards.
- Licensee may not operate the Harbourmaster Freemium Software unless the server, CPU and RAM sizing are appropriate in relation to Licensee’s usage pattern.

Any use of the Harbourmaster Freemium Software in violation of these limitations without prior written approval of Valiton is prohibited.

4. No Warranty

The Harbourmaster Freemium Software is provided on an „as is" or „with all faults" basis. Unless a defect was maliciously concealed, Valiton disclaims any and all warranties AS FOLLOWS:

Except as otherwise expressly set forth in this Agreement, Valiton makes no representation or warranty of any kind, whether express or implied. In particular, Valiton does not warrant THAT THE SPECIFICATIONS OR FUNCTIONS CONTAINED IN THE HARBOURMASTER FREEMIUM SOFTWARE WILL MEET LICENSEE's REQUIREMENTS, THAT DATA INTEGRITY IS MAINTAINED, that the Harbourmaster Freemium Software is error free, that operation of the Harbourmaster Freemium Software OR THE UNDERLYING TECHNOLOGY STACK will be secure or uninterrupted OR THAT DEFECTS IN THE HARBOURMASTER FREEMIUM SOFTWARE WILL BE CORRECTED. VALITON DOES NOT GUARANTEE THAT THE PRODUCT WILL WORK WITH THIRD-PARTY SOFTWARE OR OTHER PRODUCTS. Valiton expressly disclaims all implied warranties of merchantability, fitness for a particular purpose, quality, accuracy and\/or title. Valiton does not warrant against interference with the enjoyment of the Harbourmaster Freemium Software.

5. Liability

5.1 Valiton shall be liable for damages that (i) result from an intentional breach of its contractual obligations by Valiton, its legal representatives or its agents, (ii) occur as a result of the lack of a guaranteed specification of the Harbourmaster Freemium Software, (iii) are a result of a culpable injury to life, limb or health and\/or (iv) that are subject to product liability under the German Product Liability Act.

5.2 Valiton's liability shall be further limited to damages that are typically foreseeable in the context of an agreement such as the present one.

5.3 Further, Valiton's liability shall in each case be limited to maximum amounts of EUR 50,000 per case and EUR 100,000 overall.

5.4 Any liability other or beyond the liability provided for in this Section 5 shall be excluded.

5.5 Licensee shall be responsible for regular data backups. If Licensee suffers damages that result from the loss of data, Valiton shall only be liable for such damages insofar as the damages could not have been avoided by carrying out data backups of all relevant data in regular intervals according to industry best practice.

6. Licensee's use of the Program, Indemnification

6.1 Licensee represents and warrants that it will use the Harbourmaster Freemium Software in compliance with all applicable laws (including data protection law) and the rights of third parties.

6.2 Licensee shall indemnify, defend and hold harmless Valiton, its officers, directors, partners, employees, affiliates, subsidiaries, agents, successors and assigns from and against all third party losses, claims, damages (compensatory and punitive), liabilities, expenses (including reasonable costs of investigation and reasonable legal counsel fees), fines, penalties, judgments and settlement amounts (collectively, the "Claims") arising out of any use of the Harbourmaster Freemium Software by Licensee in violation of the provisions of this Agreement.

7. Term, Termination

7.1 This Agreement may be terminated by either party for good cause without observing a notice period. Good cause shall exist in particular if circumstances occur which, taking into consideration the substance and purpose of this Agreement, would make it unreasonable for one or both of the Parties to continue the contractual relationship. Good cause for Valiton shall include, but is not limited to:

7.1.1 Licensee’s attempt to transfer or sublicense the Harbourmaster Freemium Software;

7.1.2 Licensee‘s violation of the System Requirements and other limitations as described in Section 3

7.2 Any notice of termination must be in writing.

8. Audit Right

Valiton may at any time require assurances of compliance with the terms of this Agreement. Valiton may audit Licensee’s use of the Harbourmaster Program upon reasonable notice, either on-site or by reviewing Program use by telephone. If the audit reveals that Licensee has used the Harbourmaster Freemium Software beyond the scope of its license, Licensee shall be in material breach of the Agreement and Valiton may terminate this Agreement and all licenses immediately.

9. Export Controls

Licensee agrees that the Harbourmaster Freemium Software will not be used, shipped, transferred or exported into any country or to anyone in violation of EU or US export control regulations or in any manner prohibited by the EU Common Foreign and Security Policy or the United States Export Administration Act. Using the Harbourmaster

Freemium Software is acknowledgement that the Licensee is not located in, a resident of or under the control of any such country. Furthermore, Licensee takes complete responsibility for use of the Harbourmaster Freemium Software.

10. Miscellaneous

10.1 Neither party may assign this Agreement or any of its rights or obligations hereunder without the other party's prior written consent, except by operation of law, merger, reorganization, or as a result of an acquisition or change of control.

10.2 Valiton is authorized, but not obligated, to include Licensee in Valiton's list of reference customers.

10.3 This Agreement is the entire agreement relating to the Harbourmaster Freemium Software; it supersedes all prior or contemporaneous oral or written communications, proposals and representations with respect to the Harbourmaster Freemium Software. Any amendments of, or supplements to, this Agreement (including this Clause) must be made in writing.

10.4 Should one or more provisions of this Agreement be, become, or be deemed by a competent court to be, invalid or unenforceable, the remainder of this Agreement shall remain unaffected. Any such provision shall be replaced by the provision that best reflects the economic and legal purpose of the original provision.

10.5 This Agreement is exclusively governed by the laws of the Federal Republic of Germany, excluding, however, the application of the German and European conflicts of laws provisions and principles.

10.6 The Parties irrevocably submit to the jurisdiction of the courts of Munich, Germany, for any disputes that arise out of or in connection with this Agreement.

Annex 1 (Limitations of Harbourmaster Freemium)

- Definitions:
  - A "User" is a record in the user database table (either a unique email address or a unique social login).
  - A "Tenant" is the equivalent of a domain. Tenants are used to host multiple distinct SSO use cases on one Harbourmaster Server, which eliminates the need to set up multiple Harbourmaster Servers. Users are not shared between Tenants. All Users are owned by one Tenant.
- Free version does not support a detailed audit trail of user activity

- Free version is limited to 2 Tenants.
- Free version is limited to 20.000 registered users in total.

Annex 2 (System Requirements).

Docker 1.9 or higher, get it from http://docker.com

Sufficient CPU and RAM capacity depending on usage

Requirement ist using HTTPS for all Harbourmaster Services