Catalogi Importer

Release 0.1.0

Maykin Media

Jun 25, 2021

CONTENTS

1 Getting started **3**

2 Index **5**

2.1 Introduction 5

2.1.1 Open-source 5

2.1.2 Who's behind the Catalogi Importer? 5

2.2 Installation 5

2.2.1 Environment configuration reference 5

2.2.2 Quickstart installation 7

2.2.3 Deployment 7

2.3 Usage 9

2.3.1 Configuration 9

2.3.2 Importing i-Navigator files 11

2.3.3 Open Zaak resources 12

Easily import i-Navigator exports into the Catalogi API, such as the one in Open Zaak.

The Catalogi API is the main place to store your zaaktypes when using the API's voor Zaakgericht Werken, part of the Common Ground landscape. However, many municipalities currently have their zaaktypes stored in i-Navigator.

To keep the manual overhead to a minimum, the Catalogi Importer can load i-Navigator exports into any catalog present in a Catalogi API. All zaaktypes are created as concepts, so you can easily make additional changes afterwards if needed.

CHAPTER

ONE

GETTING STARTED

To get you started, you might find some of these links relevant:

- New to the application? Have a look at the *Introduction*
- Wondering how to use this application? Follow our *Usage manual*
- Want to get started now? Follow the *Quickstart installation*

CHAPTER

TWO

INDEX

2.1 Introduction

2.1.1 Open-source

The Catalogi Importer is open-source and available under the EUPL license.

In addition, this project makes use of various open-source Python libraries and npm packages under the hood.

2.1.2 Who's behind the Catalogi Importer?

The Catalogi Importer was initiated and funded by the municipality of Delft. Maykin Media developed the re-usable product as part of the Open Gemeente Initiatief.

2.2 Installation

2.2.1 Environment configuration reference

The Catalogi Importer can be ran both as a Docker container or directly on a VPS or dedicated server. It relies on other services, such as database and cache backends, which can be configured through environment variables.

Available environment variables

Required settings

- `DJANGO_SETTINGS_MODULE`: which environment settings to use. Available options:
  - `importer.conf.docker`
  - `importer.conf.dev`
  - `importer.conf.ci`
- `SECRET_KEY`: secret key that's used for certain cryptographic utilities. You should generate one via mini-webtool
- `ALLOWED_HOSTS`: A comma separated (without spaces!) list of domains that serve the installation. Used to protect against `Host` header attacks. Defaults to `*` for the `docker` environment and defaults to `127.0.0.1, localhost` for the `dev` environment.

Database settings

- `DB_HOST`: Hostname of the PostgreSQL database. Defaults to `db` for the `docker` environment, otherwise defaults to `localhost`.
- `DB_USER`: Username of the database user. Defaults to `importer`,
- `DB_PASSWORD`: Password of the database user. Defaults to `importer`,
- `DB_NAME`: Name of the PostgreSQL database. Defaults to `importer`,
- `DB_PORT`: Port number of the database. Defaults to `5432`.
- `CELERY_BROKER_URL`: URL for the Redis task broker for Celery. Defaults to `redis://127.0.0.1:6379/1`.
- `CELERY_RESULT_BACKEND`: URL for the Redis result broker for Celery. Defaults to `redis://127.0.0.1:6379/1`.

Other settings

- `ADMINS`: Comma seperated list (without spaces!) of e-mail addresses to sent an email in the case of any errors. Defaults to an empty list.
- `SITE_ID`: The database ID of the site object. Defaults to `1`.
- `DEBUG`: Used for more traceback information on development environment. Various other security settings are derived from this setting! Defaults to `True` for the `dev` environment, otherwise defaults to `False`.
- `IS_HTTPS`: Used to construct absolute URLs and controls a variety of security settings. Defaults to the inverse of `DEBUG`.
- `SUBPATH`: If hosted on a subpath, provide the value here. If you provide `/gateway`, the component assumes its running at the base URL: `https://somedomain/gateway/`. Defaults to an empty string.
- `SENTRY_DSN`: URL of the sentry project to send error reports to. Defaults to an empty string (ie. no monitoring).

Specifying the environment variables

There are two strategies to specify the environment variables:

- provide them in a `.env` file
- start the component processes (with uwsgi/gunicorn/celery) in a process manager that defines the environment variables

Providing a .env file

This is the most simple setup and easiest to debug. The `.env` file must be at the root of the project - i.e. on the same level as the `src` directory (NOT *in* the `src` directory).

The syntax is key-value:

```
SOME_VAR=some_value
OTHER_VAR="quoted_value"
```

Provide the envvars via the process manager

If you use a process manager (such as supervisor/systemd), use their techniques to define the envvars. The component will pick them up out of the box.

2.2.2 Quickstart installation

A `docker-compose-quickstart.yml` file is available to get the app up and running in minutes. It contains 'convenience' settings, which means that no additional configuration is needed to run the app. Therefore, it should *not* be used for anything else than testing. For example, it includes:

- A default `SECRET_KEY` environment variable
- A predefined database with the environment variable `POSTGRES_HOST_AUTH_METHOD=trust`. This lets us connect to the database without using a password.
- Debug mode is enabled.

Getting started with Docker

1. Download the `docker-compose` file:

   Linux

   Windows (Powershell 3)

```
$ wget https://raw.githubusercontent.com/maykinmedia/catalogi-importer/master/
↪docker-compose-quickstart.yml -O docker-compose.yml
```

```
PS> wget https://raw.githubusercontent.com/maykinmedia/catalogi-importer/master/
↪docker-compose-quickstart.yml -Odocker-compose.yml
```

2. Start the docker containers with `docker-compose`. If you want to run the containers in the background, add the `-d` flag to the command below.

```
$ docker-compose up
```

3. Create a super-user.

```
$ docker-compose exec web src/manage.py createsuperuser
```

4. Navigate to `http://127.0.0.1:8000` and use the credentials created above to log in.

2.2.3 Deployment

Deployment is done via Ansible. Currently, only single server deployments are described but you can just as easily deploy the application in a Kubernetes environment.

Warning: The deployment configuration (called a "playbook") is very simplistic and also contains sensitive values. This makes the playbook more readable but is not following good practices!

Server preparation

You can configure the Ansible playbook to install relevant services, do it manually, or have these pre-installed. You will need:

- PostgreSQL
- Nginx
- Docker
- Python3
- Python3 PIP

Apart from Docker, you can install all these with something like:

```
$ sudo apt-get install postgresql nginx python3 python3-pip
```

For Docker, follow the instructions here: https://docs.docker.com/engine/install/

You will also need access to, or create, a database. You can create a database with something like:

```
$  sudo su postgres --command="createuser <db-username> -P"
Enter password for new role:
Enter it again:
$ sudo su postgres --command="createdb <db-name> --owner=<db-username>"
```

Installation

1. Download the project from Github or just the deployment files.

   ```
   $ git clone git@github.com:maykinmedia/catalogi-importer.git
   ```

2. Setup virtual environment:

   ```
   $ python3 -m venv env/
   $ source env/bin/activate
   $ pip install ansible
   ```

   Note: Sometimes, additional or updates packages are needed if they are not installed by the Ansible setup installation. You can do so like this:

   ```
   $ python -m pip install -U pip
   $ pip install ordered_set packaging appdirs six
   ```

3. Install Ansible collections:

   ```
   $ ansible-galaxy collection install community.docker
   $ ansible-galaxy collection install git+https://github.com/maykinmedia/
   ↪commonground-ansible.git
   ```

   Note: The last collection might require explicit access.

4. Edit the playbook `app.yml` to match your setup. Take special note of all **TODO** settings and **read through all the comments and variables**.
5. Run the playbook:

```
$ ansible-playbook app.yml --become --ask-become-pass
```

2.3 Usage

2.3.1 Configuration

Before the Catalogi Importer can be used we need to setup credentials for the backend services. If you just need to add a new catalog in a pre-configured Open Zaak installation, skip to *Add a catalog*.

Configure Open Zaak

To be able to connect from Catalogi Importer to Open Zaak we need to create API credentials in Open Zaak:

a. Navigate to **API Autorisaties > Applicaties**
b. Click **Applicatie toevoegen**.
c. Fill out the form:
   - **Label**: *For example:* `Catalogi Importer`
   - **Client ID**: *For example:* `catalogi-importer`
   - **Secret**: *Some random string. You will need this later on!*
d. Click **Opslaan en opnieuw bewerken**.
e. Click **Beheer autorisaties**.
f. Select component **Catalogi API** and scopes **catalogi.lezen** en **catalogi.schrijven**.
g. Click **Opslaan**

Configure Catalogi Importer

Next, we need to add these credentials to the Catalogi Importer:

a. Navigate to **Importer > Services**
b. Click **Add Service**.
c. Fill out the form:
   - **Label**: *For example:* `Open Zaak Productie`
   - **Type**: `ZTC (Zaaktypen)`
   - **API Root URL**: *For example:* `https://openzaak.gemeente.nl/catalogi/api/v1/`
   - **Client id:**: *Same as above*
   - **Secret:**: *Same as above*
   - **Authorization type:**: `ZGW Client id and secret`
   - **OAS**: `https://openzaak.gemeente.nl/catalogi/api/v1/schema/openapi.yaml`

d. Click **Save**.

The Catalogi Importer uses the VNG Selectielijst (such as the one provided by Open Zaak):

a. Navigate to **Importer > Services**

b. Click **Add Service**.

c. Fill out the form:

- **Label**: *For example:* `Selectielijst`
- **Type**: `ORC (overige)`
- **API Root URL**: `https://selectielijst.openzaak.nl/api/v1/`
- **Authorization type:**: `No authorisation`
- **OAS**: `https://selectielijst.openzaak.nl/api/v1/schema/openapi.yaml`

d. Click **Save**.

e. Navigate to **Importer > Selectielijst configuration**

f. In the pull-down menu select the record with the label we just created.

g. Click **Save**.

Add a catalog

We need to retrieve the UUID of the Open Zaak Catalog we want to import to from the Open Zaak admin:

a. Navigate to **Gegevens > Catalogi**

b. Select the Catalogus we want to import to, or create a new one for testing purposes.

c. Copy the **UUID** value from the form.

Then we configure this Catalog in Catalogi Importer:

a. Navigate to **Importer > Catalog configurations**

b. Click **Add Catalog configuration**.

c. Fill out the form:

- Select the **Service** that with the label we configured earlier.
- Paste the **UUID** we copied from Open Zaak.
- Enter a descriptive **Label**, ideally matching the Catalog in Open Zaak.

d. Click **Save**.

e. The system will validate the **UUID** at the selected **Service**.

- If the details are correct the RSIN and domain will be retrieved and printed

This catalog is now configured in Catalogi Importer to accept one or more **Import Jobs**.

This is a one-time configuration that is valid for all catalogs in this Open Zaak installation.

Additional Open Zaak installations like a testing or acceptation environment can be added following the same pattern.

2.3.2 Importing i-Navigator files

Make sure you have an XML export of i-Navigator ready and *configured* the Catalog Importer correctly.

Create an Import Job

Every import operation is represented as an **Import Job** and sets the Catalog, the Selectielijst-year and the iNavigator XML-file to use, plus some parameters for the import.

a. Navigate to **Importer > Import jobs**

b. Click **Add Import job**.

c. Fill out the form:

- Select the **Catalog** we configured earlier.
- Select the *Selectielijst year* to use.
- Click **Browse** and select the iNavigator **XML file** to import.
- Optionally override **Start date** to set the begin date of the new records (eg: *beginGeldigheid* in Open Zaak).
- Select "Close published" to close currently published Zaaktypen or InformatieObjecttypen on the above date. Note this means there won't be active records after this date until you publish the newly imported records.

d. Click **Continue**.

e. The system runs a pre-check on the XML and reports potential issues.

- Carefully take note of the reported issues.
- Some of these need to be solved in iNavigator and exported again and run in a new Import Job, and some can be fixed later in Open Zaak.

f. If the report is acceptable click **Continue** and a long running background task is started to run the import.

- The screen will periodically update the counters but the job will continue even if you close the browser tab.

g. When the import is done the report will display with the results of the actual import.

- This report will be saved with the ** Import Job** and can be accessed later for review.

h. Open this Catalog in your Open Zaak admin or other client software and review the new records.

To monitor progress or review an earlier precheck or import report:

a. Navigate to **Importer > Import jobs**

b. Select the **Import job** you're interested in from the list.

2.3.3 Open Zaak resources

Catalogi Importer improteerd de volgende resources in Open Zaak

Zaaktype

Matched with existing records on their **indentificatie/id** field (B-Nummer).

Additional note for importing: If the i-Navigator record does not have a 'resultaat nummer' for the Selectielijst to determine the Zaaktype's 'selectielijstProcestype'; the 'toelichting' field can be used to add a value so the import can complete.

Use the format `nummer, toelichting`, eg `123, voorbeeld tekst`.

InformatieObjecttype / documenttype

Matched with existing records on their **omschrijving** field.

Roltype

Matched with existing records in this Zaaktype on their **omschrijving** field.

Statustype

Matched with existing records in this Zaaktype on their **volgnummer** field.

Resultaatype

Matched with existing records in this Zaaktype on their **omschrijving** field.

Additional note for importing: If the i-Navigator record does not have a 'resultaat nummer' for the Selectielijstto determine the Resultaattype's 'resultaattypeomschrijving and 'selectielijstklasse'; the 'toelichting' field can be used to add a value so the import can complete.

Use the format `nummer, toelichting`, eg `123, voorbeeld tekst`.

Zaaktypeinformatieobjecttype

(connection between Zaaktype en InformatieObjecttype)

Matched with existing records in this Zaaktype and InformatieObjecttype on their **omschrijving** field.