CompleteSearch Documentation

Release 1.0.0

Evgeny Anatskiy

Jun 26, 2017

Contents:

1 Installation **3**
- 1.1 Requirements . . . 3
- 1.2 Installation steps . . . 3
- 1.3 Usage . . . 4
- 1.4 Troubleshooting . . . 4

2 API Documentation **5**
- 2.1 Upload API . . . 5
- 2.2 Search API . . . 6
- 2.3 Settings API . . . 7

3 Indices and tables **9**

Python Module Index **11**

This is a Master Project at the University of Freiburg.

CHAPTER 1

Installation

The easiest way to install the app is to use CompleteSearch the Docker image (below). Otherwise, all installation steps can be seen here.

Requirements

- Docker for Linux/macOS/Windows

Installation steps

Step 0

Install Docker.

Step 1

Clone the repository:

```
git clone https://github.com/anatskiy/docker-completesearch.git
cd docker-completesearch
```

Build the image:

```
docker build \
  -t completesearch \
  --build-arg SVN_USERNAME="username" \
  --build-arg SVN_PASSWORD="password" \
  .
```

Where SVN_USERNAME and SVN_PASSWORD are your credentials for the CompleteSearch svn repository. More information.

Note: You need to escape the username and the password with double quotes "".

Step 2

Run the container:

```
docker run -d \
  --name completesearch \
  -p 8000:8000 \
  -p 8888:8888 \
  completesearch \
  python3 manage.py runserver -h 0.0.0.0 -p 8000
```

Usage

Open CompleteSearch at http://localhost:8000/

Enter the container:

```
docker exec -it completesearch /bin/bash
```

Run a command inside of the container:

```
docker exec -d completesearch <command>
```

See the app logs:

```
docker exec -t completesearch cat app.log
```

or:

```
docker logs --tail 100 completesearch
```

Troubleshooting

If you see the error `Cannot get facets for ... CompleteSearch server is not responding.`, try to restart the server:

```
docker exec -d completesearch make stop start
```

CHAPTER 2

API Documentation

Upload API

API operations on uploading files.

upload.views.**allowed_file**(*filename*)
Check if the uploading file's type is allowed.

Parameters **filename** – filename

Returns result of the check

Return type bool

upload.views.**define_facets**(*data*)
Define facets by their occurrence in the dataset.

Parameters **data** – DataFrame with the uploaded dataset

Returns field names which will be used as facets

Return type list

upload.views.**save_uploaded_dataset**()

POST /save_uploaded_dataset/ Save the ploaded dataset's settings and start the CompleteSearch server

Parameters **data** – a dictionary with all dataset's settings, which have been generated by the upload_file function.

Returns dictionary with the success property and an error message

Return type JSON response

upload.views.**upload_file**()

POST /upload_file/ Upload, validate and process a file, and send it to CompleteSearch.

Parameters

- **use_first_row** – use the first data row as a header. If the parameter is `False`, the column names will be generated automatically (i.e. Column1, Column2, etc.).
- **file** – the uploading file

Returns dictionary with dataset settings, e.g. facet/filter fields, which fields to use for the full-text search, etc.

Return type JSON response

Search API

API operations searching information.

`search.views.`**`get_facets`**`()`

GET /get_facets/?name={name}&q={query} Return all facets for a given field name and a search query.

Parameters

- **name** – facet field name
- **q** – search query

Returns list of facet items for the given field

Return type JSON response

`search.views.`**`get_facets_list`**`()`

GET /get_facets_list/ Get the list of facet fields.

Returns list of all facet fields

Return type JSON response

`search.views.`**`process_user_input`**`(`*query*`)`

Process the user input by stripping extra whitespaces, ensuring prefix search by appending asterisks to each search term, and escaping commas and semicolons.

Parameters **query** – search query

Returns processed query

Return type string

Doctests:

```
>>> process_user_input('')
''
```

```
>>> process_user_input('antonio vivaldi')
'antonio* vivaldi*'
```

```
>>> process_user_input('vivaldi, antonio')
'vivaldi","* antonio*'
```

```
>>> process_user_input('The . Beatles$')
'The*.Beatles$'
```

```
>>> process_user_input(':facet:Preis:* :facet:Preis:2.40')
':facet:Preis:* :facet:Preis:"2.40"'
```

```
>>> process_user_input('magic :facet:Preis:* :facet:Preis:2.40')
':facet:Preis:* :facet:Preis:"2.40" magic*'
```

```
>>> process_user_input('ge :facet:Autor:* mo*   vi$ | fr | neur.netw')
':facet:Autor:* ge* mo* vi$|fr*|neur*.netw*'
```

```
>>> process_user_input('mo| fr | ge .tu')
'mo*|fr*|ge*.tu*'
```

search.views.**search**()

GET /search/?q={query}&start={start}&hits_per_page={hits_per_page} Perform search using CompleteSearch.

Parameters

- **q** – search query
- **start** – send hits starting from this one
- **hits_per_page** – how many hits return per page

Returns list of search hits

Return type JSON response

Settings API

API operations on configuring the uploaded dataset.

settings.views.**configure_dataset**()

POST /configure_dataset/ Change dataset parameters and regenerate CompleteSearch's indices.

Parameters

- **title_field** – which field to use as a hit's title
- **allow_multiple_items** – fields containing multiple terms per column, e.g. several authors per one document
- **within_field_separator** – a delimiter which is used in `allow_multiple_items`
- **full_text** – which columns should be searched
- **show** – which fields should be returned on a hit
- **filter** – search in a specific column
- **facets** – restrict the search to specific columns and phrases

Returns dictionary with the `success` property and an `error` message

Return type JSON response

settings.views.**delete_dataset**()

POST /delete_dataset/ Reset app's settings, delete the uploaded dataset and stop the CompleteSearch server.

Returns dictionary with the `success` property

Return type JSON response

`settings.views.`**`get_settings`**`()`

GET /get_settings/ Get a dictionary with all dataset settings.

Returns dictionary with dataset settings, e.g. facet/filter fields, which fields to use for the full-text search, etc.

Return type JSON response

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

s

search.views, 6
settings.views, 7

u

upload.views, 5

Index

A

allowed_file() (in module upload.views), 5

C

configure_dataset() (in module settings.views), 7

D

define_facets() (in module upload.views), 5
delete_dataset() (in module settings.views), 7

G

get_facets() (in module search.views), 6
get_facets_list() (in module search.views), 6
get_settings() (in module settings.views), 8

P

process_user_input() (in module search.views), 6

S

save_uploaded_dataset() (in module upload.views), 5
search() (in module search.views), 7
search.views (module), 6
settings.views (module), 7

U

upload.views (module), 5
upload_file() (in module upload.views), 5