Process Cubes Developer Guide

Requirements

Python

You can also find all python requirements in the ***requirements.txt***, which is in the root folder of the Django webservice. This file is always up to date. The requirements can be installed with:

```
pip install -r requirements.txt
```

Currently the requirements are:

- django
- django-tables2
- django-crispy-forms
- djongo # Django-MongoDB Connector
- sqlparse==0.2.4 #required by djongo
- pm4py # Process Mining framework
- dnspython

Database

There needs to be a MongoDB (**version >= 3.6**) instance running. The database settings are stored in ***process_cubes/settings.py*** in the **DATABASES** variable.

Getting started

Run the webservice

- Install the requirements defined in the requirements.txt
    - e.g. by running: `pip install -r requirements.txt`
- Install MongoDB
- run: `python manage.py migrate --run-syncdb` to create the document collections
- run: `python manage.py runserver` to start the webservice

Django

To learn more about the Django framework, have a look at the gettings started guide: https://docs.djangoproject.com/en/2.2/intro/tutorial01/

Django Apps

Every page/functionality is in its own Django App. You can create an app with:

`python manage.py startapp`

After this, the app must be added to the **INSTALLED_APPS** in the ***settings.py*** in the ***process_cubes*** app.

Data

Have a look at the design specification document to read more about the data design. Additionally, have a look at the *import_xes* function in the *import_xes* module to see how the log is stored in the database.

Currently, the models for **EventLog, ProcessCube, Dimension and Attribute**, are defined in the ***models.py*** of the ***import_xes*** app. These might be moved to the ***process_cubes*** app, to keep the ***import_xes*** app clean and have them at a central place, because they are required by every app. Slice and Dice objects are in the ***slice-dice*** app.

Events

Events and Traces are not modeled as Django models. These are stored directly in the database with PyMongo.

Events are stored in the **`events`** collection. One document (or object) in this collection stores for each attribute a value. Each event has the attribute names as keys and the values as values. Attributes of traces start with 'trace:'. The ID of the corresponding trace is stored under **`trace:_id`**.

Example:

```
{
    "_id":"5ce43529aca227ddfe30a572",
    "org:group":"A",
    "lifecycle:transition":"complete",
    "concept:name":"ER Sepsis Triage",
    "time:timestamp":"2014-12-21T11:15:45.000Z",
    "trace:concept:name":"B",
    "trace:_id":"5ce43528aca227ddfe30a13e",
    "log":8
}
```

Dictionary attributes

If the event log has attributes that have dictionaries as values, every entry in the dictionary is handled as an attribute, with the name:
`<name of the dictionary>:<key of the entry>`.

The values are stored like this:

```
{
    …
    "trace:OrderLine_Details": {
        "value": "EMPTY",
        "children": {
            "TotalAmount": "99.9800",
            "Description": "John Denver's life Album (15 CD premium pack)",
            "Quantity": "1",
            "Price": "99.9800"
        }
    },
    …
}
```

To filter for example for the quantity in the dictionary use `{trace:OrderLine_Details.children.Quantity': 1.0}` as a query

The way these attributes are stored in an event might get improved in the future.

Traces

Traces are stored in the **traces** collection. Like events, each trace object has its attributes as keys and a value as value. Each trace stores the ID of its trace. The trace itself doesn't store which events belong to the trace.

Example:

```
{
    "_id":"5ce43528aca227ddfe30a13e",
    "concept:name":"B",
    "log":8
}
```

MongoDB

How do I access the database?

For everything that is implemented with Django Models, i.e. exists as a class in the corresponding ***models.py***. To learn more about querying Django objects have a look at: https://docs.djangoproject.com/en/2.2/ref/models/querysets/

Events and Traces are not modeled as Django models. These are stored directly in the database with PyMongo.

For example, to get all events for a given event log:

```
client = MongoClient(host=DATABASES['default']['HOST'])
db = client[DATABASES['default']['NAME']]
event_collection = db['events']

t1 = time.time()
events = event_collection.find({'log': log.id})
```

Have a look at the PyMongo tutorial to learn more about querying the database: http://api.mongodb.com/python/current/tutorial.html

FAQ

`ImportError: cannot import name 'create_prompt_application'`

If you get an error like this:

```
from prompt_toolkit.shortcuts import create_prompt_application, create_eventloop,
create_prompt_layout, create_output

ImportError: cannot import name 'create_prompt_application' File
"/usr/local/lib/python3.6/site-packages/django/db/models/base.py", line 111, in
_new_ "INSTALLED_APPS." % (module, name)

RuntimeError: Model class import_xes.models.EventLog doesn't declare an explicit
app_label and isn't in an application in INSTALLED_APPS.
```

You probably have an incompatible version of *promp_toolkit*. Install a compatible version with:

```
pip install 'prompt-toolkit<2.0.0,>=1.0.15' --force-reinstall
```

Future Improvements

Step Size

Currently step size means how many elements are combined. This might be improved in the future such that it is possible to set a value range or e.g. for timestamps a step size as days/years/months etc.

Performance

There are some serious performance issues for large event logs (>100mb). One limiting factor here is PM4Py. Another problem is the “Show Values” button. It takes a quite long time to construct the table that shows the values and the page is frezzed until it’s loaded.

GitHub

You can find the sources on GitHub: https://github.com/Moo-State/PCubes-PADS2019. Feel free to contribute by opening issues or making pull request.