# BigData – Project „MTG Trading Card – Crawler”

This document contains information about the creation process of individual containers and some used software components. Also shown is an example of use.

There are 4 containers described in more detail below, which were used to solve the task.

## Container – airflow

This container is based on the Docker image "marcelmittelstaedt/airflow" but has been modified. It contains all scripts and applications important for the airflow DAG. Among them is the file of the DAG and some self-programmed libs that provide functionalities for crawling, information extraction and for connecting to Hive as well as MySQL. The detailed description of the used DAG operators can be found on the GitHub wiki.

Procedure (to start Docker container and get into it):

* docker run -dit --name airflow -p 8080:8080 --net bigdatanet adrianw424/airflow:latest
* docker exec -it airflow bash

Access the container innards:

* sudo su airflow
* cd
* cd airflow/dags

In this folder is the file "mtgcrawler.py" and the folder "libs". "mtgcrawler.py" contains the Airflow DAG, which can be edited. In the folder "libs" are the files "crawler.py" and "hiveMySQLLink.py". These are two scripts that are used by the DAG. "crawler.py" is the web crawler that extracts the desired data from the website and converts it into a CSV file. "hiveMySQLLink.py" contains a class with different functionalities that are used to transfer the data stored in Hive to a MySQL database.

* cd

The folder "mtg" will be created in here if it does not already exist. The CSV file created by "crawler.py" with the extracted data is stored in this folder.

What’s new in the container:

* Installed Python3.6 modules:
  + mysql-connector-python
  + pyhive
* Installed Python3.9:

Due to version problems of the Python library "asyncio" with Python3.6, Python3.9 had to be installed additionally. “asyncio” is required by “crawler.py” to work properly. The following modules were added to Python3.9:

* + aiohttp
  + asyncio
  + bs4
  + urllib3
* The previously mentioned folders and files

## Container – hadoop

This container is completely based on the Docker image "marcelmittelstaedt/spark\_base". The crawled data is copied from the "airflow" container and then imported into Hive.

Procedure (to start Docker container and get it work):

* docker run -dit --name hadoop -p 8088:8088 -p 9870:9870 -p 9864:9864 -p 10000:10000 -p 8032:8032 -p 8030:8030 -p 8031:8031 -p 9000:9000 -p 8888:8888 --net bigdatanet adrianw424/hadoop:latest
* docker exec -it hadoop bash
* sudo su hadoop
* cd
* start-all.sh
* hiveserver2

(Note that hiveserver2 should not be terminated. Otherwise, access to hive is not possible.)

## Container – mysql

This container is completely based on the official Docker image "mysql". To make the data accessible to users, it is imported into a database of RDMS MySQL. MySQL is in this container.

Procedure (to start Docker container and get it work):

* docker run --name mysql -p 3306:3306 --net bigdatanet -e MYSQL\_ROOT\_PASSWORD=mysql -d adrianw424/mysql:latest

## Container – frontend

This container is based on the Docker image "ubuntu:18.04", but it has been modified. It contains scripts for the front end of the application.

Procedure (to start Docker container and get it work):

* docker run -dit --name frontend -p 5000:5000 --net bigdatanet adrianw424/frontend:latest
* docker exec -it frontend bash
* cd
* cd webfrontend
* python3.8 frontend.py

(Note that the development server should not be terminated. Otherwise, access to it is not possible.)

Now you should be able to access the frontend, by typing “[IP-Address/External-IP]:5000” into your browser.

Access the container innards:

* cd
* cd webfrontend

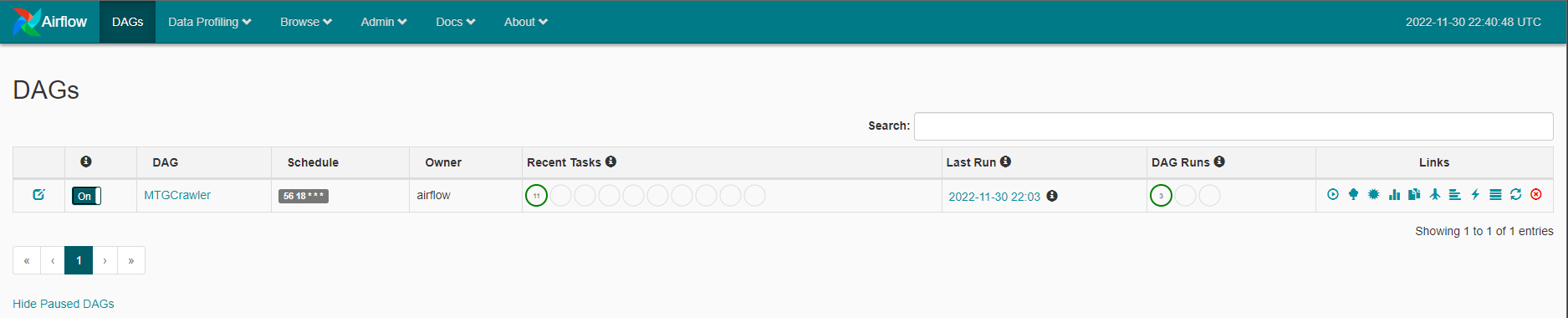
In this folder are the files "frontend.py", "MySQLHandler.py" and the folders "static" and "templates". "frontend.py" is the main script and contains the starting point for running the web server. "MySQLHandler.py" contains the class "MySQLHandler" which can be used to implement MySQL database accesses. The folder "static" contains static elements, in this case it contains the folder "styles" and in it again the file "style001.css". In "style001.css" design data for the displayed window are stored. The "templates" folder contains the "index.html" file. This contains information about the structure of the website and is called by "frontend.py".

What’s new in the container:

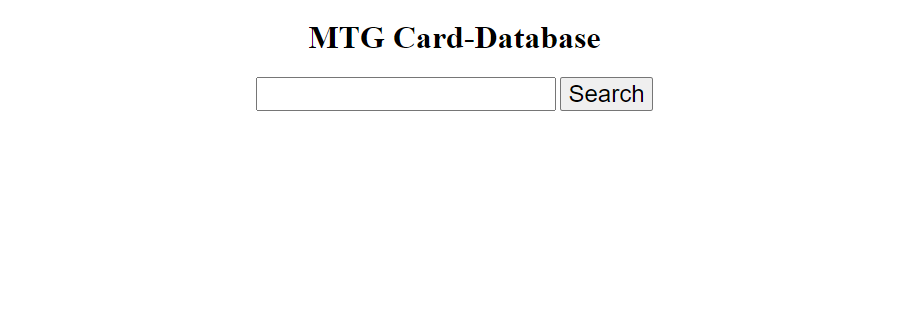
* Installed Python3.8 and some modules:
  + flask
  + mysql-connector-python

## Example

After all containers have been started, Airflow can be accessed via the web frontend. This is done via the URL “[IP-Address/External-IP]:8080/admin”.



Subsequently, the DAG must be switched on. After some time the work is completed and the data from the MySQL database can be accessed using the frontend “[IP-Address/External-IP]:5000”.



In the search bar you can now enter terms to be searched for in the database. For example, "Abandon" can be entered. The result of this search is shown in the following image.

