

# CBOE WEB APPLICATION

---

A simple class-based Flask API (flask-restplus) with a Celery beat scheduler, dockerised.

It is made from two parts, a Flask backend and a React Frontend.

The Flask backend can be found in this current repo: (<https://github.com/Markonick/cboe-pitch>)

The React frontend can be found here: (<https://github.com/Markonick/cboe-react>)

## Prerequisites

- Docker (<https://docs.docker.com/install/>)
- docker-compose (<https://docs.docker.com/compose/install/>)
- Ports 5000, 5433, 5555, 6379 free

## Environment Variables

Create an **.env** file in the root directory and add the following:

```
FLASK_APP=run.py
PITCH_ENDPOINT=http://backend:5000/api/v1/pitch
SQLALCHEMY_TRACK_MODIFICATIONS=False
CELERY_BROKER_URL=redis://redis:6379/0
CELERY_RESULT_BACKEND=redis://redis:6379/0
DATABASE_URL=postgresql://postgres:admin@cboe-db:5432/pitch
POSTGRES_USER=postgres
POSTGRES_PASSWORD=admin
DATA_FILE=pitch_data.txt
PER_PAGE=50
```

## Backend Docker Containers

In order to run the celery tasks, we need to run 6 docker containers.

1. Flask backend
2. Postgres
3. Celery beat scheduler
4. Celery worker
5. Redis queue
6. Flower (Web based GUI task monitor)

## Frontend Docker Containers

## 1. React/Nginx

# Instructions

To start the frontend container served on nginx on port 80 (for the purposes of this app we did not use HTTPS), just

go to the folder where you git cloned or copied **cboe-react** and do a

```
docker-compose up --build
```

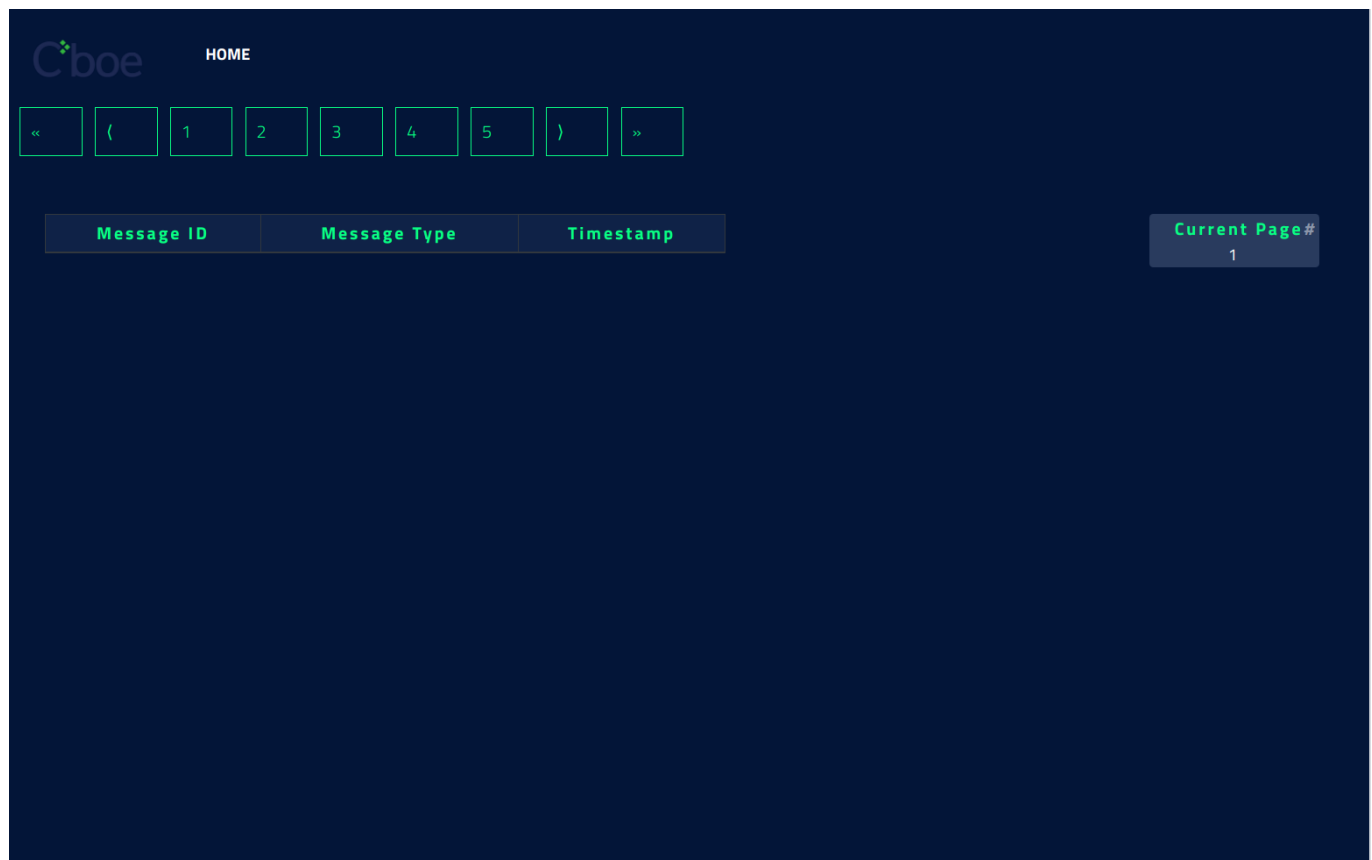
To start the backend flask application, similarly do a

```
docker-compose up --build
```

in **cboe-pitch** at the folder app root.

This should kick-off all containers.

Navigate to the website: (<http://localhost>)



You should see a website, but no data, since we haven't uploaded the data file yet.

You can observe the supported API endpoints in Swagger at

(http://localhost:5000/)

# CBOE PITCH API

1.0

[ Base URL: / ]  
<http://localhost:5000/swagger.json>

Allows users to upload pitch data

pitch

CBOE pitch data related operations

POST

/api/v1/pitch

GET

/api/v1/pitch

Eg. by clicking on the GET method, we can execute a get list command:

GET

/api/v1/pitch

Parameters

Try it out

No parameters

Responses

Response content type

application/json

Curl

curl -X GET "http://localhost:5000/api/v1/pitch" -H "accept: application/json"

Request URL

http://localhost:5000/api/v1/pitch

Server response

Code

Details

200

Response body

```
{
  "body": {
    "counts": [
      {
        "count": 10364,
        "message_type": "Add Order (Short)"
      },
      {
        "count": 9594,
        "message_type": "Order Cancel"
      },
      {
        "count": 27,
        "message_type": "Trade (Short)"
      },
      {
        "count": 20,
        "message_type": "Order Executed"
      },
      {
        "count": 1,
        "message_type": "Symbol Clear"
      }
    ],
    "messages": [
      {
        "description": "Add Order (Short)",
        "message_id": 20000,
        "timestamp": "2019-10-30T14:00:00.000Z"
      }
    ]
  }
}
```

Download

Response headers

```
access-control-allow-origin: *
content-length: 12806
```

and the Flower monitor at

(http://127.0.0.1:5555/tasks)

Flower

Dashboard

Tasks

Broker

Monitor

Logout

Docs

Code

Show

10

entries

Search:

Name	UUID	State	args	kwargs	Result	Received	Started	Runtime	Worker
tasks.upload_pitch_data	b9d4a197-ef05-4371-b03c-f1057d99a253	SUCCESS	0	{}	None	2019-10-26 17:51:46.266	2019-10-26 17:51:46.276	0.008	celery@788dff8c8ab6
tasks.upload_pitch_data	94bc9603-32c7-4b8d-8075-7899b2b06af4	SUCCESS	0	{}	None	2019-10-26 17:51:56.267	2019-10-26 17:51:56.274	0.001	celery@788dff8c8ab6
tasks.upload_pitch_data	585535e0-be89-493d-b9a7-d828fcbab98f	SUCCESS	0	{}	None	2019-10-26 17:52:06.266	2019-10-26 17:52:06.271	0.000	celery@788dff8c8ab6
tasks.upload_pitch_data	dcef01e5-074f-4cba-9c97-70c62fcad691	SUCCESS	0	{}	None	2019-10-26 17:52:16.266	2019-10-26 17:52:16.272	0.001	celery@788dff8c8ab6
tasks.upload_pitch_data	388dbf9a-d979-421b-92c1-8ea8fe4f8387	SUCCESS	0	{}	None	2019-10-26 17:52:26.266	2019-10-26 17:52:26.272	0.001	celery@788dff8c8ab6
tasks.upload_pitch_data	a096e2b6-ba72-4d4a-a642-c38f6139fb5e	SUCCESS	0	{}	None	2019-10-26 17:52:36.265	2019-10-26 17:52:36.272	0.001	celery@788dff8c8ab6
tasks.upload_pitch_data	4a40ee60-5222-4c0a-8826-e7609770f729	SUCCESS	0	{}	None	2019-10-26 17:52:46.265	2019-10-26 17:52:46.269	0.001	celery@788dff8c8ab6
tasks.upload_pitch_data	6ff0ec73-8cd7-4065-a99c-36389831649e	SUCCESS	0	{}	None	2019-10-26 17:52:56.265	2019-10-26 17:52:56.273	0.006	celery@788dff8c8ab6
tasks.upload_pitch_data	831757b6-3bb0-4482-9088-6df7cdaeef8d1	SUCCESS	0	{}	None	2019-10-26 17:53:06.266	2019-10-26 17:53:06.275	0.007	celery@788dff8c8ab6
tasks.upload_pitch_data	63e1605c-a966-467e-ae1a-0c9b4f290966	SUCCESS	0	{}	None	2019-10-26 17:53:16.266	2019-10-26 17:53:16.276	0.008	celery@788dff8c8ab6

Showing 1 to 10 of 174 entries

Previous

1

2

3

4

5

...

18

Next

This is not all however. The image above shows that indeed there were tasks scheduled but nothing really happened. We first need to make a migration (through alembic / Flask-migrate).

Make sure there are no **migrations** folder already installed in the root app folder. If there is then remove it:

```
sudo rm -rf migrations
```

Now do the migrations:

```
docker exec -it backend flask db init
docker exec -it backend flask db migrate
docker exec -it backend flask db upgrade
```

This creates the relations in our database so that we can now populate them with real data.

The app still doesn't do anything usefull, it will need the data file. The terminal should currently look something like this:

```

beat_1 | [2019-10-26 18:34:06.304: INFO/MainProcess] Scheduler: Sending due task Upload new pitch data file every 10 sec, if there is a file available in the path (tasks.upload_pitch_data)
worker_1 | [2019-10-26 18:34:06.311: INFO/ForkPoolWorker-1] Task tasks.upload_pitch_data[ede17251-5b9f-434b-8a99-9f347f5b81e1] succeeded in 0.003163981993566267s: None
worker_1 | [2019-10-26 18:34:16.306: INFO/MainProcess] Received task: tasks.upload_pitch_data[d551cd8c-2f7b-4952-bcd1-8b4523388548]
beat_1 | [2019-10-26 18:34:16.304: INFO/MainProcess] Scheduler: Sending due task Upload new pitch data file every 10 sec, if there is a file available in the path (tasks.upload_pitch_data)
worker_1 | [2019-10-26 18:34:16.314: INFO/ForkPoolWorker-1] Task tasks.upload_pitch_data[d551cd8c-2f7b-4952-bcd1-8b4523388548] succeeded in 0.007005092003964819s: None
worker_1 | [2019-10-26 18:34:26.306: INFO/MainProcess] Received task: tasks.upload_pitch_data[d3160c63-1bfe-4d4b-bb14-ca3ed65fab54]
beat_1 | [2019-10-26 18:34:26.304: INFO/MainProcess] Scheduler: Sending due task Upload new pitch data file every 10 sec, if there is a file available in the path (tasks.upload_pitch_data)
worker_1 | [2019-10-26 18:34:26.311: INFO/ForkPoolWorker-1] Task tasks.upload_pitch_data[d3160c63-1bfe-4d4b-bb14-ca3ed65fab54] succeeded in 0.002994752998347394s: None
worker_1 | [2019-10-26 18:34:36.304: INFO/MainProcess] Scheduler: Sending due task Upload new pitch data file every 10 sec, if there is a file available in the path (tasks.upload_pitch_data)
worker_1 | [2019-10-26 18:34:36.308: INFO/MainProcess] Received task: tasks.upload_pitch_data[0ec94c9d-5ba2-4859-9914-f0bb33c478c0]
worker_1 | [2019-10-26 18:34:36.314: INFO/ForkPoolWorker-1] Task tasks.upload_pitch_data[0ec94c9d-5ba2-4859-9914-f0bb33c478c0] succeeded in 0.0031246990110957995s: None
beat_1 | [2019-10-26 18:34:46.304: INFO/MainProcess] Scheduler: Sending due task Upload new pitch data file every 10 sec, if there is a file available in the path (tasks.upload_pitch_data)
worker_1 | [2019-10-26 18:34:46.308: INFO/MainProcess] Received task: tasks.upload_pitch_data[fd4c7b1-2486-430c-bc18-5b7eb86b0d61]
worker_1 | [2019-10-26 18:34:46.315: INFO/ForkPoolWorker-1] Task tasks.upload_pitch_data[fd4c7b1-2486-430c-bc18-5b7eb86b0d61] succeeded in 0.003720633001648821s: None
beat_1 | [2019-10-26 18:34:56.304: INFO/MainProcess] Scheduler: Sending due task Upload new pitch data file every 10 sec, if there is a file available in the path (tasks.upload_pitch_data)
worker_1 | [2019-10-26 18:34:56.307: INFO/MainProcess] Received task: tasks.upload_pitch_data[6944f88c-ea8f-480b-967e-caa80e14f17b]
worker_1 | [2019-10-26 18:34:56.314: INFO/ForkPoolWorker-1] Task tasks.upload_pitch_data[6944f88c-ea8f-480b-967e-caa80e14f17b] succeeded in 0.003917807000107132s: None
beat_1 | [2019-10-26 18:35:06.304: INFO/MainProcess] Scheduler: Sending due task Upload new pitch data file every 10 sec, if there is a file available in the path (tasks.upload_pitch_data)
worker_1 | [2019-10-26 18:35:06.308: INFO/MainProcess] Received task: tasks.upload_pitch_data[5ba2ed11-def6-47d1-acf3-3ec99976d94a]
worker_1 | [2019-10-26 18:35:06.316: INFO/ForkPoolWorker-1] Task tasks.upload_pitch_data[5ba2ed11-def6-47d1-acf3-3ec99976d94a] succeeded in 0.0034360880026090145s: None
beat_1 | [2019-10-26 18:35:16.304: INFO/MainProcess] Scheduler: Sending due task Upload new pitch data file every 10 sec, if there is a file available in the path (tasks.upload_pitch_data)
worker_1 | [2019-10-26 18:35:16.309: INFO/MainProcess] Received task: tasks.upload_pitch_data[5581fae4-4805-4e40-8286-f9a3f42e30ea]
worker_1 | [2019-10-26 18:35:16.312: INFO/ForkPoolWorker-1] Task tasks.upload_pitch_data[5581fae4-4805-4e40-8286-f9a3f42e30ea] succeeded in 0.0008147229964379221s: None
worker_1 | [2019-10-26 18:35:26.307: INFO/MainProcess] Received task: tasks.upload_pitch_data[09623e8f-ef89-458e-8ce8-778d5e3e108b]
beat_1 | [2019-10-26 18:35:26.304: INFO/MainProcess] Scheduler: Sending due task Upload new pitch data file every 10 sec, if there is a file available in the path (tasks.upload_pitch_data)
worker_1 | [2019-10-26 18:35:26.311: INFO/ForkPoolWorker-1] Task tasks.upload_pitch_data[09623e8f-ef89-458e-8ce8-778d5e3e108b] succeeded in 0.003144225003779866s: None

```

To this end we will copy the **pitch data file** containing the stock orders in the celery folder, for the celery task to pick up and parse it. First of all, make sure no **pitch\_data.txt** files already exist in the celery folder.

If none, then let's copy the file, which will effectively kick off a task at the next 10 second scheduler beat:

```
cp pitch_data.txt celery
```

We should see **20 HTTP POSTS** corresponding to 20 bulk uploads (**20000 rows/1000 per bulk upload**) and hopefully, their corresponding **200 HTTP responses**:

```

backend | 2019-10-26 18:36:58.654 INFO werkzeug Thread-49 : 172.24.0.4 - - [26/Oct/2019 18:36:58] "POST /api/v1/pitch HTTP/1.1" 200 -
backend | 2019-10-26 18:36:58.998 DEBUG REPO Thread-50 : CREATE PITCH REPO: COMMITTING TO DB
backend | 2019-10-26 18:36:59.000 DEBUG API Thread-50 : True
backend | 2019-10-26 18:36:59.003 INFO werkzeug Thread-50 : 172.24.0.4 - - [26/Oct/2019 18:36:59] "POST /api/v1/pitch HTTP/1.1" 200 -
backend | 2019-10-26 18:36:59.362 DEBUG REPO Thread-51 : CREATE PITCH REPO: COMMITTING TO DB
backend | 2019-10-26 18:36:59.366 DEBUG API Thread-51 : True
backend | 2019-10-26 18:36:59.367 INFO werkzeug Thread-51 : 172.24.0.4 - - [26/Oct/2019 18:36:59] "POST /api/v1/pitch HTTP/1.1" 200 -
backend | 2019-10-26 18:36:59.824 DEBUG REPO Thread-52 : CREATE PITCH REPO: COMMITTING TO DB
backend | 2019-10-26 18:36:59.826 DEBUG API Thread-52 : True
backend | 2019-10-26 18:36:59.827 INFO werkzeug Thread-52 : 172.24.0.4 - - [26/Oct/2019 18:36:59] "POST /api/v1/pitch HTTP/1.1" 200 -
backend | 2019-10-26 18:37:00.207 DEBUG REPO Thread-53 : CREATE PITCH REPO: COMMITTING TO DB
backend | 2019-10-26 18:37:00.211 DEBUG API Thread-53 : True
backend | 2019-10-26 18:37:00.212 INFO werkzeug Thread-53 : 172.24.0.4 - - [26/Oct/2019 18:37:00] "POST /api/v1/pitch HTTP/1.1" 200 -
backend | 2019-10-26 18:37:00.561 DEBUG REPO Thread-54 : CREATE PITCH REPO: COMMITTING TO DB
backend | 2019-10-26 18:37:00.564 DEBUG API Thread-54 : True
backend | 2019-10-26 18:37:00.567 INFO werkzeug Thread-54 : 172.24.0.4 - - [26/Oct/2019 18:37:00] "POST /api/v1/pitch HTTP/1.1" 200 -
backend | 2019-10-26 18:37:00.966 DEBUG REPO Thread-55 : CREATE PITCH REPO: COMMITTING TO DB
backend | 2019-10-26 18:37:00.973 DEBUG API Thread-55 : True
backend | 2019-10-26 18:37:00.977 INFO werkzeug Thread-55 : 172.24.0.4 - - [26/Oct/2019 18:37:00] "POST /api/v1/pitch HTTP/1.1" 200 -
backend | 2019-10-26 18:37:01.433 DEBUG REPO Thread-56 : CREATE PITCH REPO: COMMITTING TO DB
backend | 2019-10-26 18:37:01.437 DEBUG API Thread-56 : True
backend | 2019-10-26 18:37:01.440 INFO werkzeug Thread-56 : 172.24.0.4 - - [26/Oct/2019 18:37:01] "POST /api/v1/pitch HTTP/1.1" 200 -
backend | 2019-10-26 18:37:01.802 DEBUG REPO Thread-57 : CREATE PITCH REPO: COMMITTING TO DB
backend | 2019-10-26 18:37:01.805 DEBUG API Thread-57 : True
backend | 2019-10-26 18:37:01.808 INFO werkzeug Thread-57 : 172.24.0.4 - - [26/Oct/2019 18:37:01] "POST /api/v1/pitch HTTP/1.1" 200 -
backend | 2019-10-26 18:37:02.179 DEBUG REPO Thread-58 : CREATE PITCH REPO: COMMITTING TO DB
backend | 2019-10-26 18:37:02.181 DEBUG API Thread-58 : True
backend | 2019-10-26 18:37:02.184 INFO werkzeug Thread-58 : 172.24.0.4 - - [26/Oct/2019 18:37:02] "POST /api/v1/pitch HTTP/1.1" 200 -
backend | 2019-10-26 18:37:02.564 DEBUG REPO Thread-59 : CREATE PITCH REPO: COMMITTING TO DB
backend | 2019-10-26 18:37:02.568 DEBUG API Thread-59 : True
backend | 2019-10-26 18:37:02.569 INFO werkzeug Thread-59 : 172.24.0.4 - - [26/Oct/2019 18:37:02] "POST /api/v1/pitch HTTP/1.1" 200 -
backend | 2019-10-26 18:37:02.909 DEBUG REPO Thread-60 : CREATE PITCH REPO: COMMITTING TO DB
backend | 2019-10-26 18:37:02.914 DEBUG API Thread-60 : True
backend | 2019-10-26 18:37:02.916 INFO werkzeug Thread-60 : 172.24.0.4 - - [26/Oct/2019 18:37:02] "POST /api/v1/pitch HTTP/1.1" 200 -
backend | 2019-10-26 18:37:03.293 DEBUG REPO Thread-61 : CREATE PITCH REPO: COMMITTING TO DB
backend | 2019-10-26 18:37:03.296 DEBUG API Thread-61 : True
backend | 2019-10-26 18:37:03.299 INFO werkzeug Thread-61 : 172.24.0.4 - - [26/Oct/2019 18:37:03] "POST /api/v1/pitch HTTP/1.1" 200 -
backend | 2019-10-26 18:37:03.658 DEBUG REPO Thread-62 : CREATE PITCH REPO: COMMITTING TO DB
backend | 2019-10-26 18:37:03.664 DEBUG API Thread-62 : True
backend | 2019-10-26 18:37:03.665 INFO werkzeug Thread-62 : 172.24.0.4 - - [26/Oct/2019 18:37:03] "POST /api/v1/pitch HTTP/1.1" 200 -
backend | 2019-10-26 18:37:04.017 DEBUG REPO Thread-63 : CREATE PITCH REPO: COMMITTING TO DB
backend | 2019-10-26 18:37:04.021 DEBUG API Thread-63 : True
backend | 2019-10-26 18:37:04.022 INFO werkzeug Thread-63 : 172.24.0.4 - - [26/Oct/2019 18:37:04] "POST /api/v1/pitch HTTP/1.1" 200 -
worker_1 | [2019-10-26 18:37:04.028: INFO/ForkPoolWorker-1] Task tasks.upload_pitch_data[33038b59-76db-417e-9d2c-4c10227b31e5] succeeded in 7.6880959200061625s: None
worker_1 | [2019-10-26 18:37:06.337: INFO/MainProcess] Received task: tasks.upload_pitch_data[20e5028e-92ba-4111-9639-b46c530de16a]

```

The website runs on (<http://localhost>) and to get it up and running you will need to run a simple docker-compose up

on the **cboc-react** repo.

CboeHOME

«

{

1

2

3

4

5

}

»

Message ID	Message Type	Timestamp
20000	Add Order (Short)	28963734
19999	Add Order (Short)	28963706
19998	Order Cancel	28963699
19997	Add Order (Short)	28963698
19996	Add Order (Short)	28963685
19995	Add Order (Short)	28963679
19994	Order Cancel	28963665
19993	Order Cancel	28963659
19992	Add Order (Short)	28963656
19991	Add Order (Short)	28963641
19990	Add Order (Short)	28963634
19989	Order Cancel	28963630
19988	Add Order (Short)	28963624
19987	Order Cancel	28963618
19986	Order Cancel	28963607
19985	Order Cancel	28963597
19984	Add Order (Short)	28963594
19983	Order Cancel	28963591
19982	Add Order (Short)	28963575
19981	Order Cancel	28963556
19980	Add Order (Short)	28963553

Current Page #1

Add Order (Short) #10361

Order Cancel #9592

Trade (Short) #27

Order Executed #20

Testing

To run the functional tests open a new terminal at the app root folder and run

```
docker exec -it backend pytest -v
```

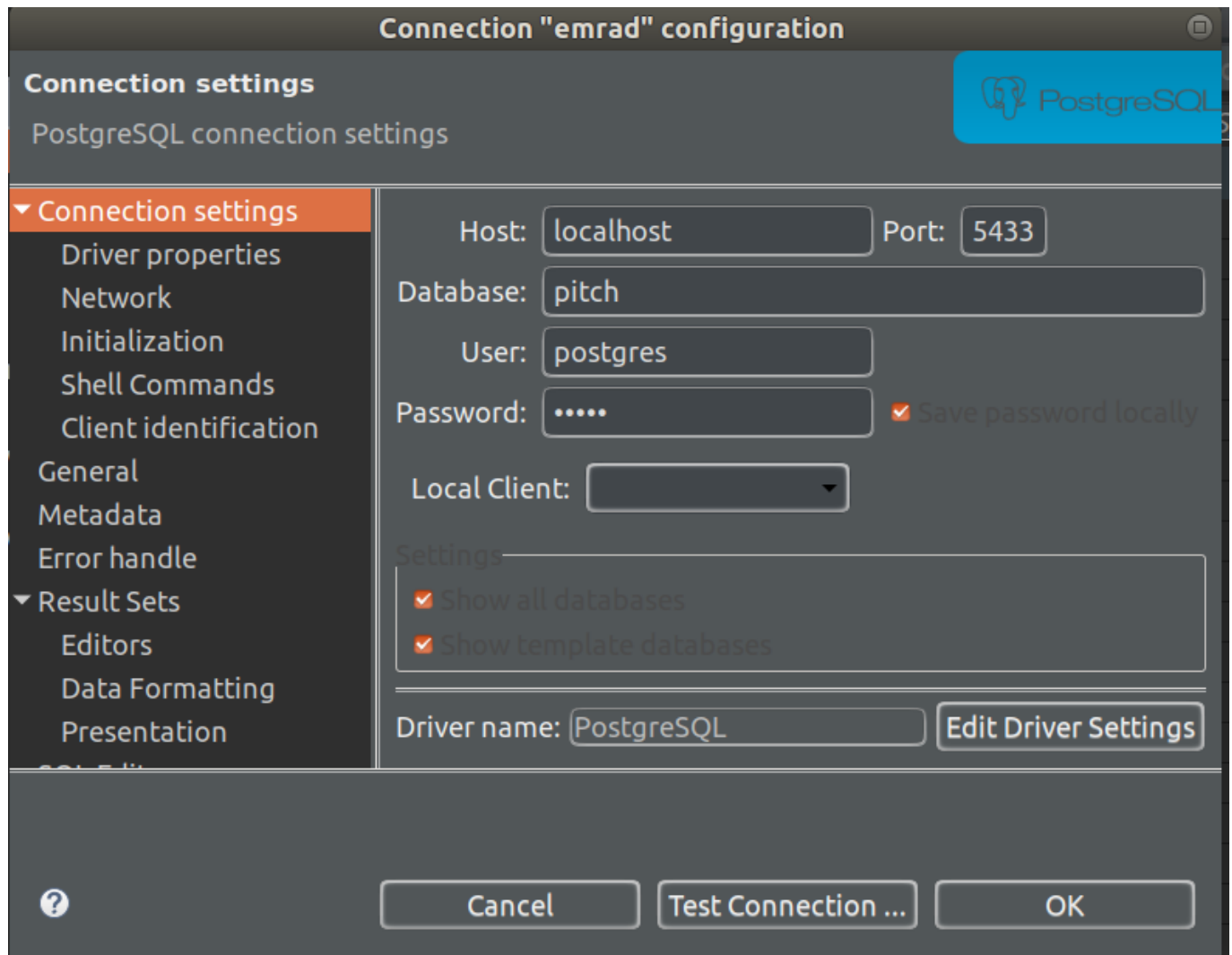
Pitch Message Types

ID	DESCRIPTION
0	Symbol Clear
1	Add Order (Short)
2	Add Order (Long)
3	Order Executed
4	Order Cancel
5	Trade (Short)
6	Trade (Long)
7	Trade Break

ID	DESCRIPTION
8	Trading Status
9	Auction Update
10	Auction Summary

## DBeaver

You can easily inspect this simple database by setting up a db connection on DBeaver:



or you can login to the postgres docker container via `docker exec -it cboe-pitch_cboe-db_1 bash`

then to enter postgres just type:

```
psql -U postgres
```

and you can now connect to pitch:

```
\c pitch
```

---

and investigate tables and write normal db queries.

## Miscellaneous

If for any reason we are in a unsure state, its always better to just build everything from scratch.

If you want to really build everything from scratch that means even deleting docker image caches and dangling images.

Stop the containers:

```
docker-compose stop
```

Stop containers not already stopped and remove containers, networks, volumes, and images created by up:

```
docker-compose down
```

Prunes images, containers, and networks:

```
docker system prune
```

Use --volumes flag to remove dangling volumes (especially to avoid running out of disk space)

If for any reason you get a permission error like this in between builds and test runs:

```
PermissionError: [Errno 13] Permission denied:  
'/home/markonick/Projects/cboe-pitch/tests/__pycache__/conftest.cpython-37-  
pytest-5.2.2.pyc'
```

just run a

```
sudo find . -name "*.pyc" -exec rm -f {} \;
```