# 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
```

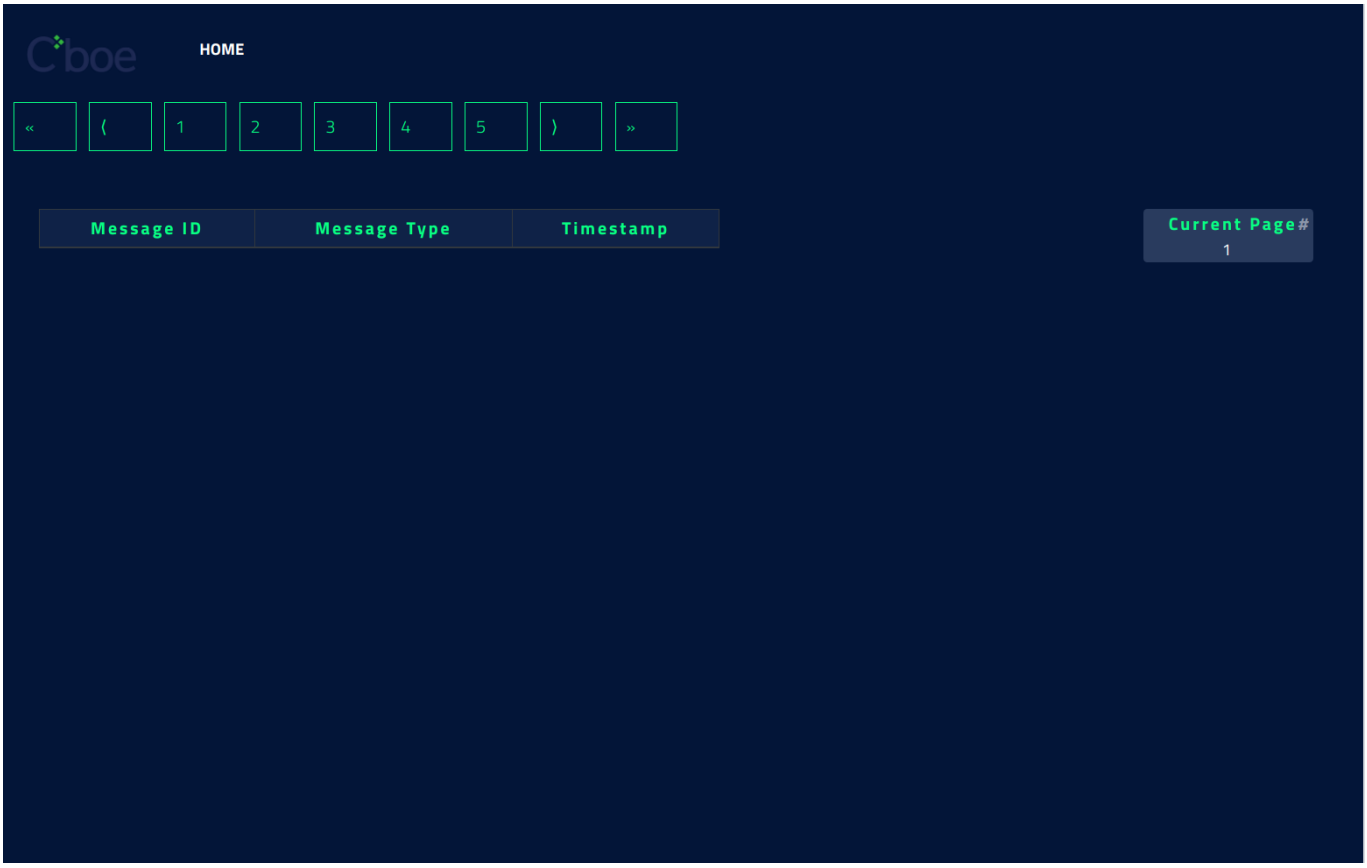
To start the backend flask application, similarly do a

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

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

This should kick-off all containers.

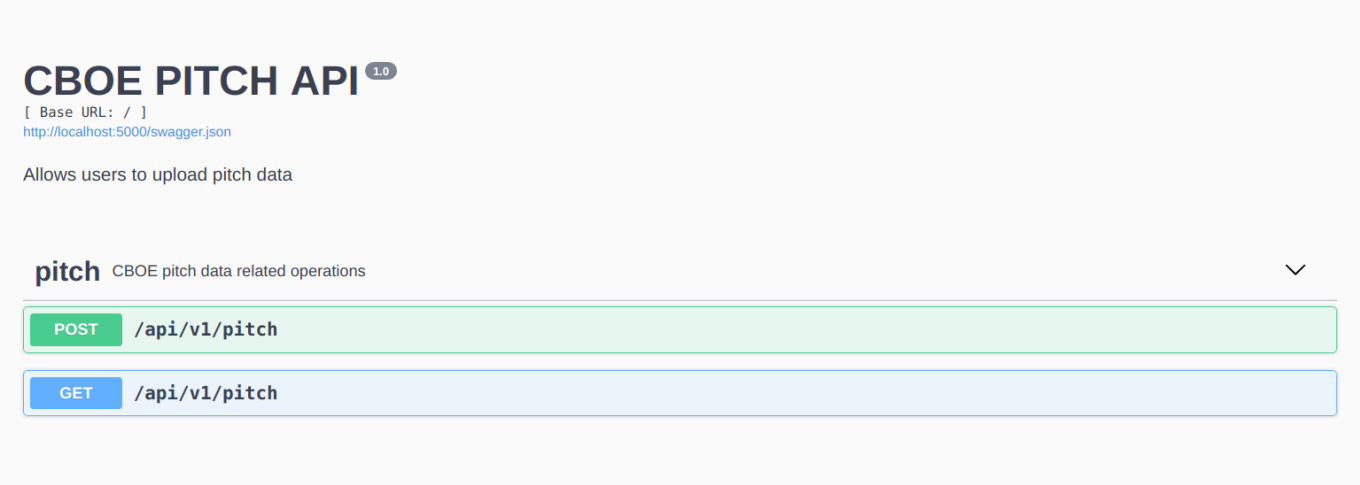
Navigate to the website: (http://localhost)



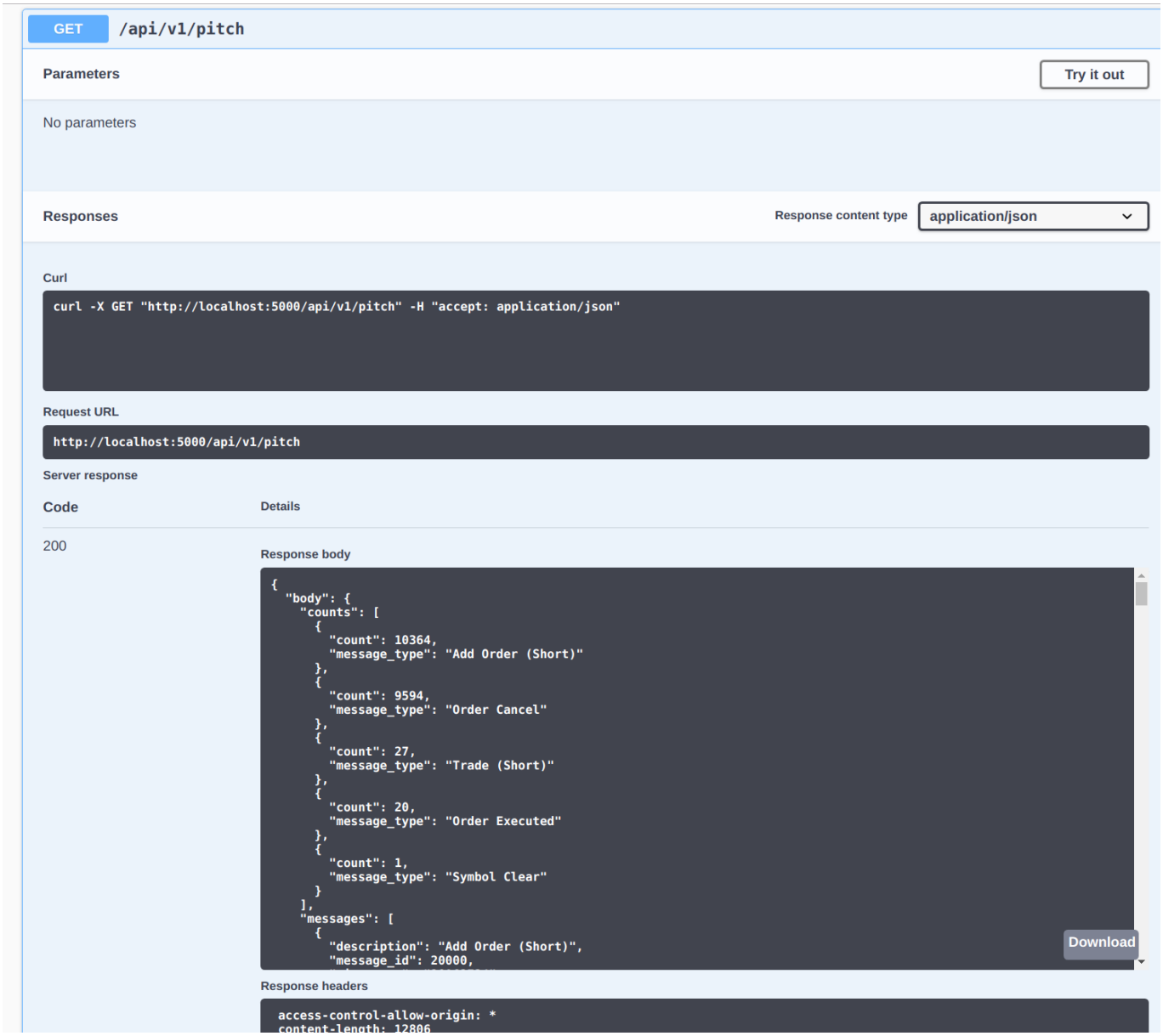

You should see a website, but no data, since we haven't uploaded the data file yet.

You can observe the supported API endpoints in Swagger at

```
(http://localhost:5000/)
```

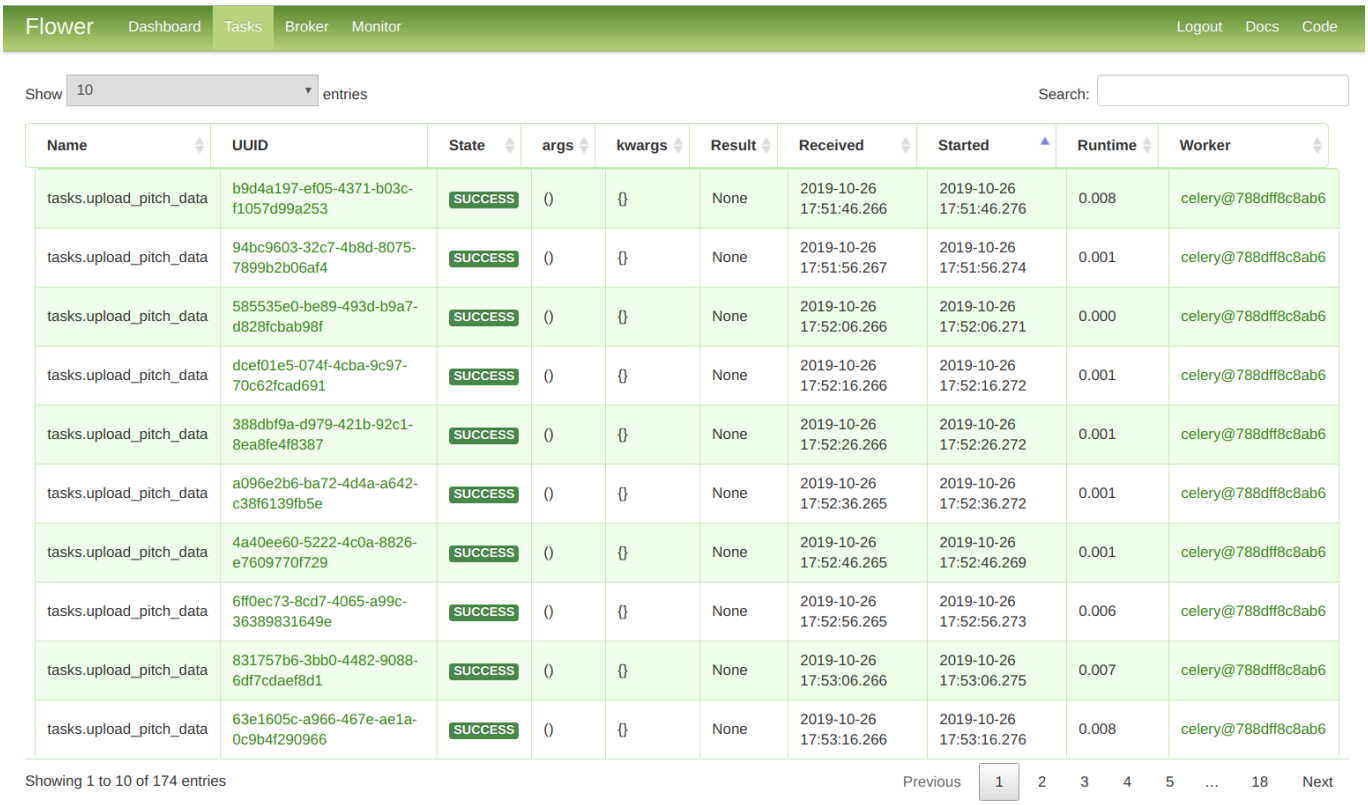



Eg. by clicking on the GET method, we can execute a get list command:




and the Flower monitor at

```
(http://127.0.0.1:5555/tasks)
```




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:

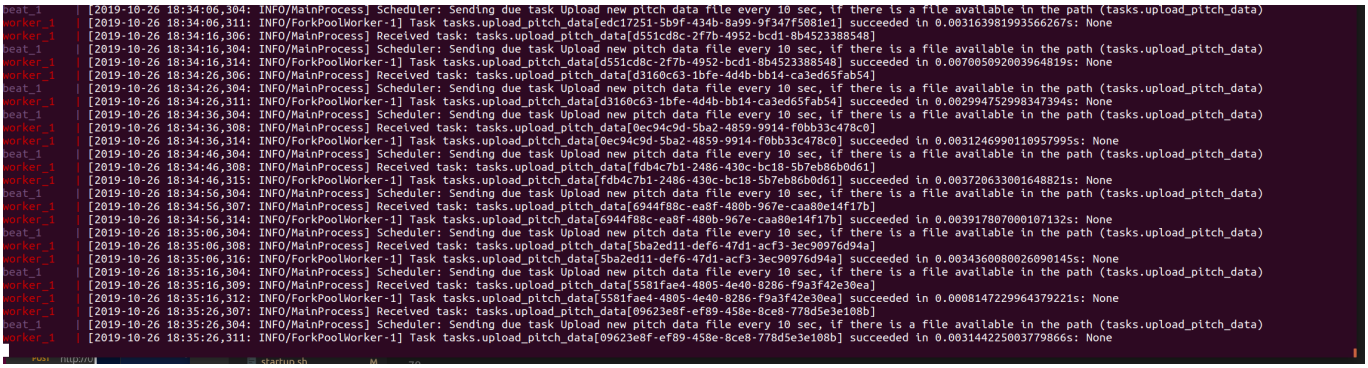
```
sudo rm -rf migrations
```

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:

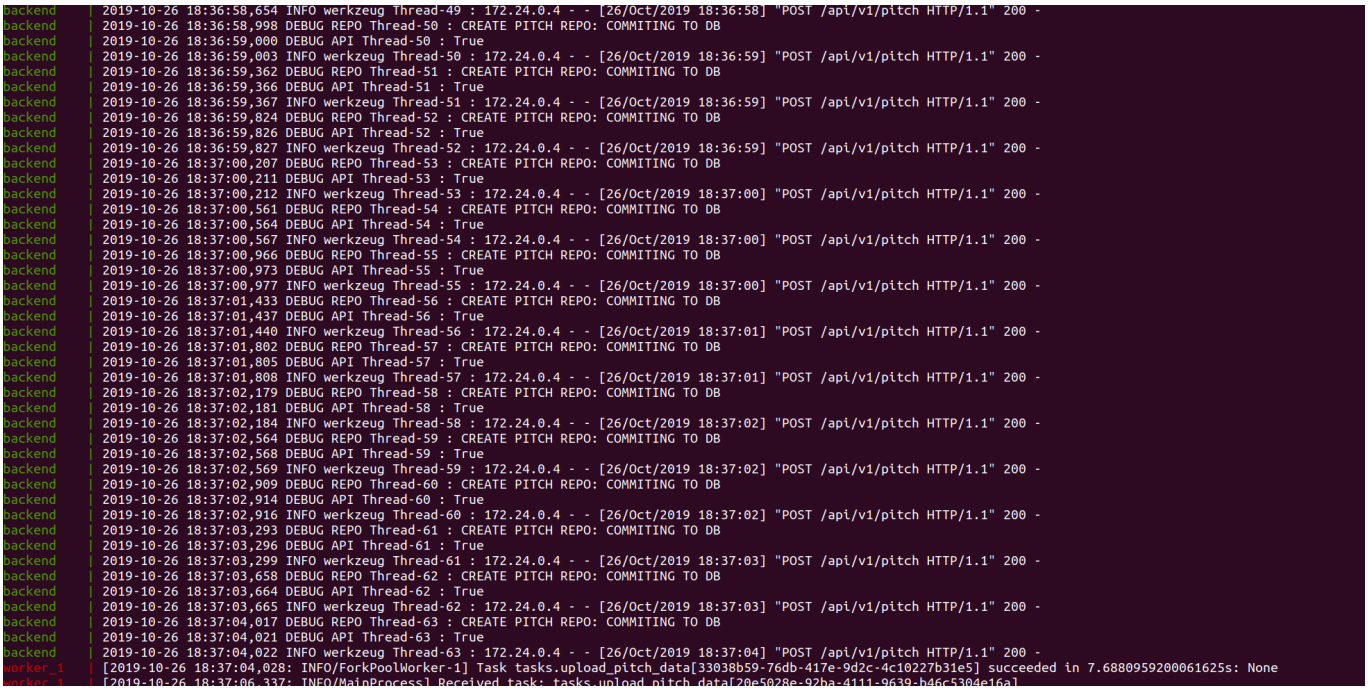



To this end we will copy the **pitch data file** containg the stock orders in the celery folder, for the celery task to pick up and parse it. First of all, make sure no **pitch_data.txt** files already exist in the celery folder.

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

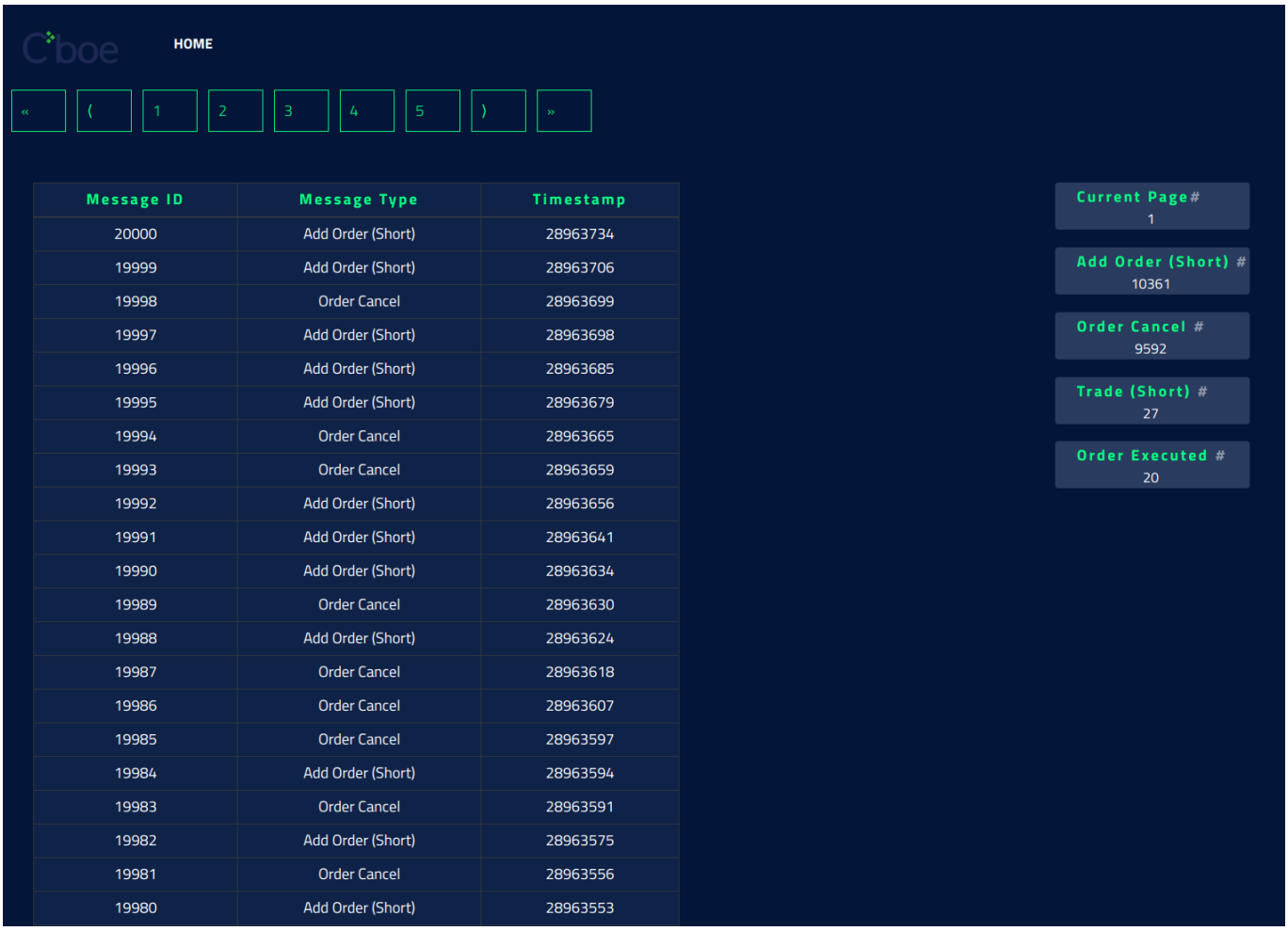
```
cp pitch_data.txt celery
```

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




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 **cboe-react** repo.




## Testing

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

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

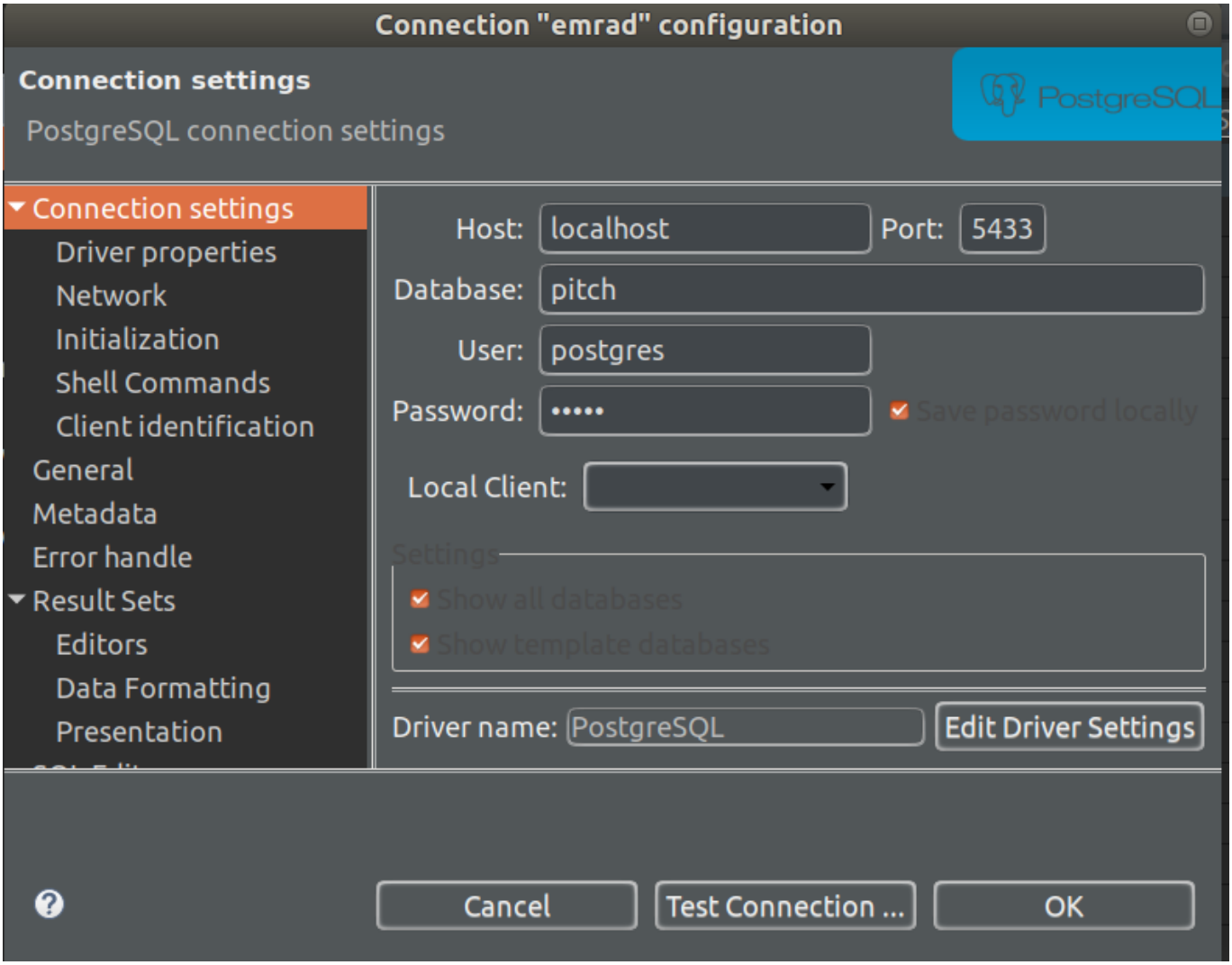
## Pitch Message Types

| 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 {} \;
```