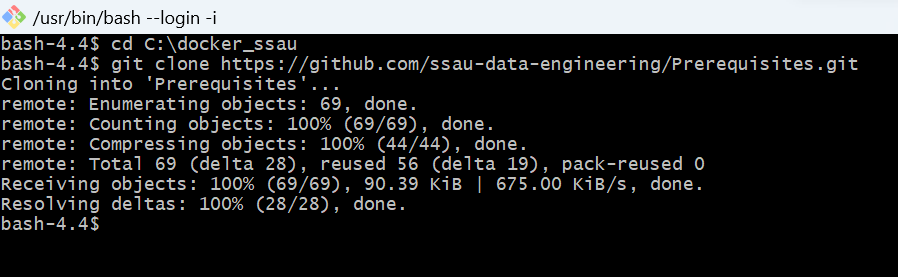
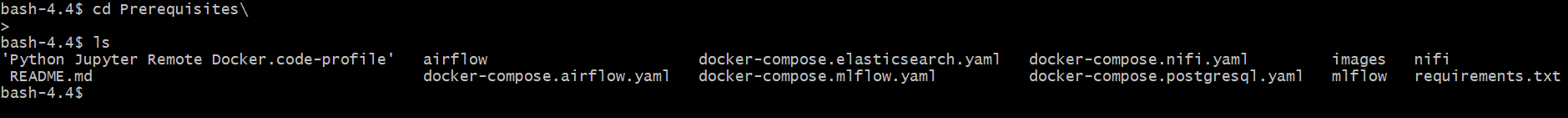
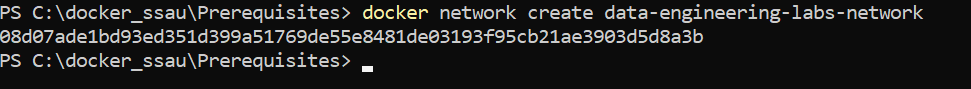
**Файл Nifi-scheme.xml и DAG файл airflow\_dag.py к работе приложены**

# Разворачивание Docker

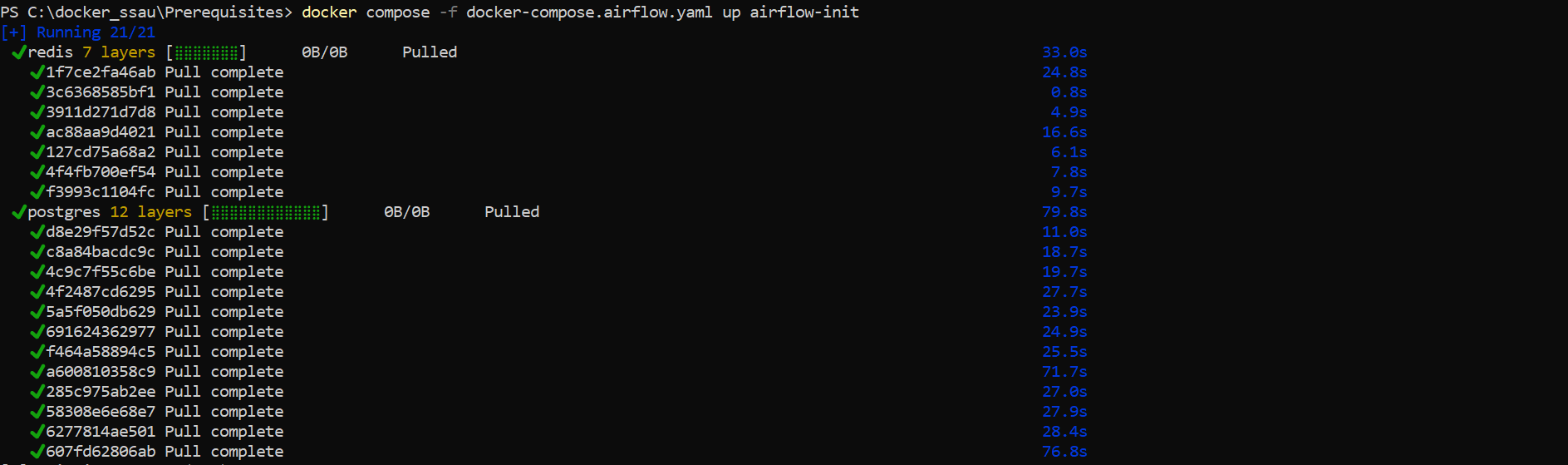




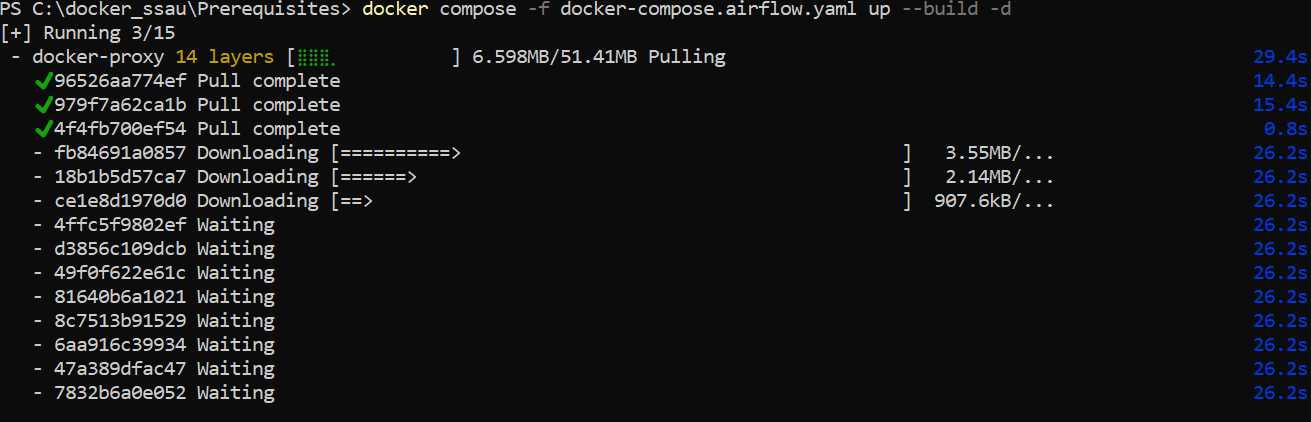
1. docker network create data-engineering-labs-network



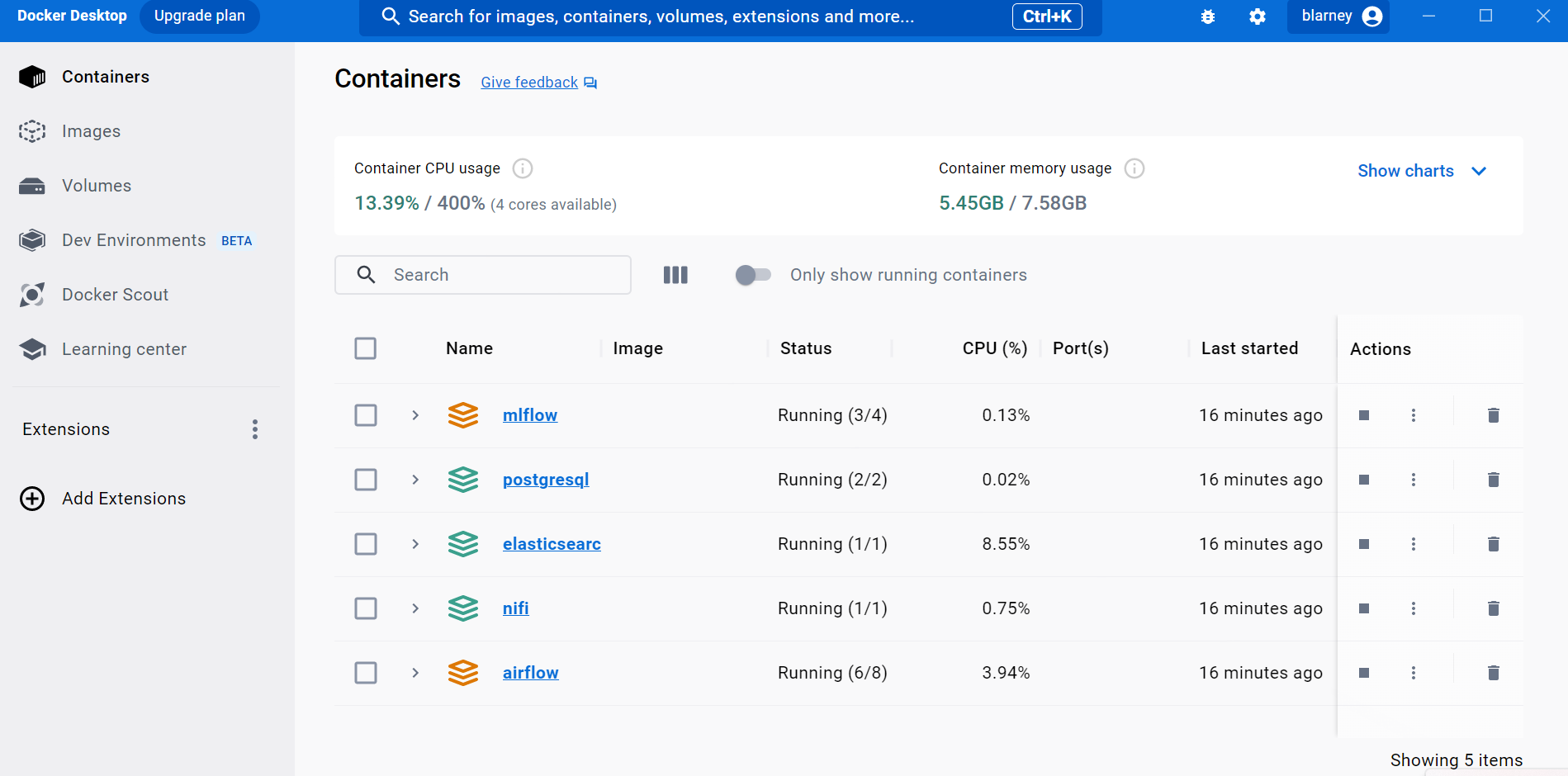
docker compose -f docker-compose.airflow.yaml up airflow-init



docker compose -f docker-compose.airflow.yaml up --build -d

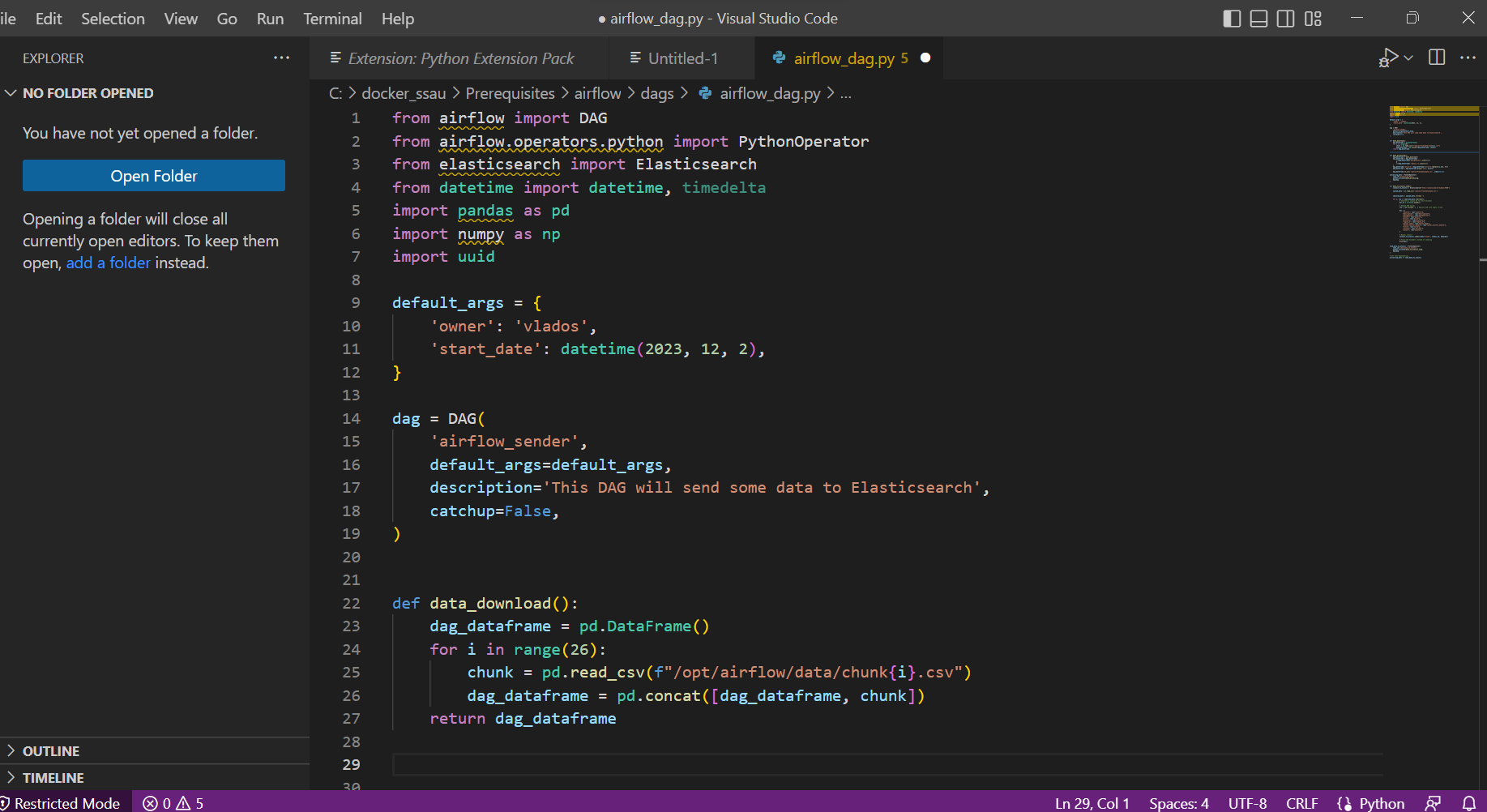


Запускаем остальные контейнеры

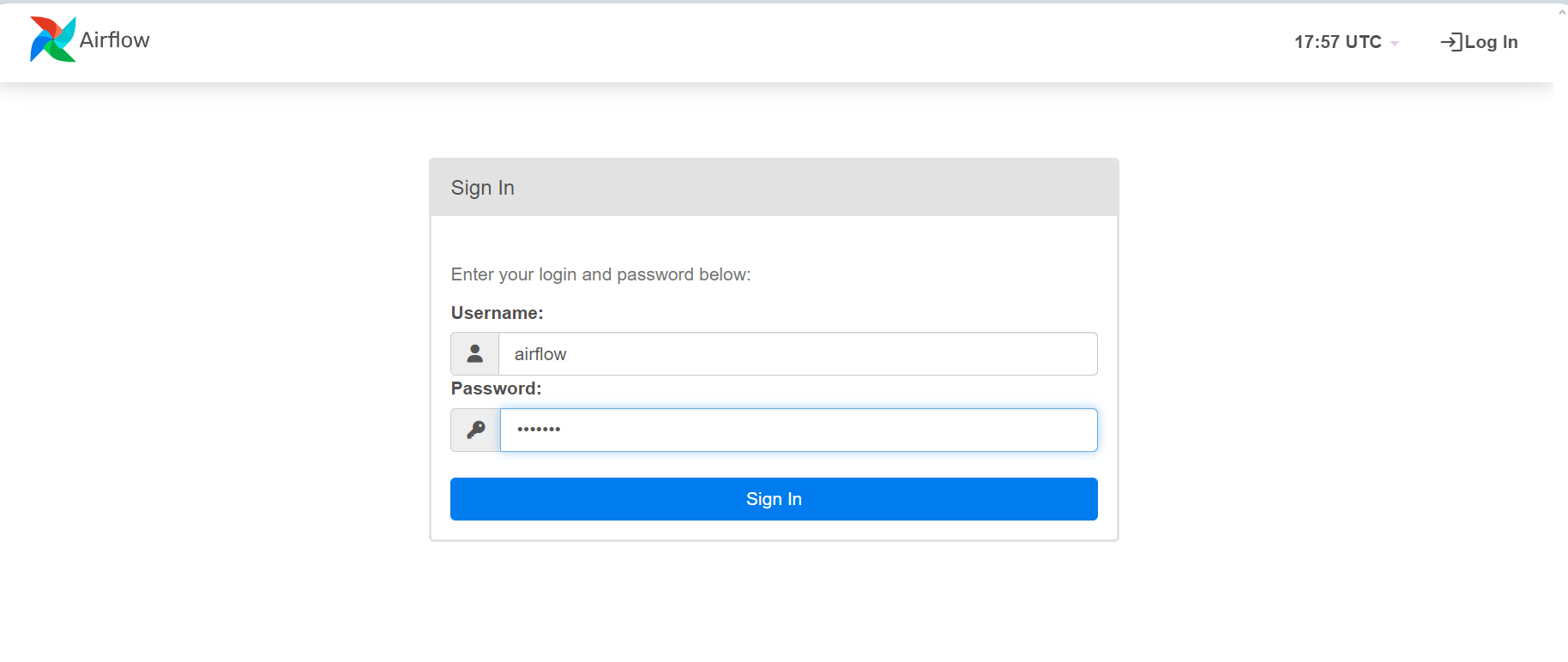


# AIRFLOW

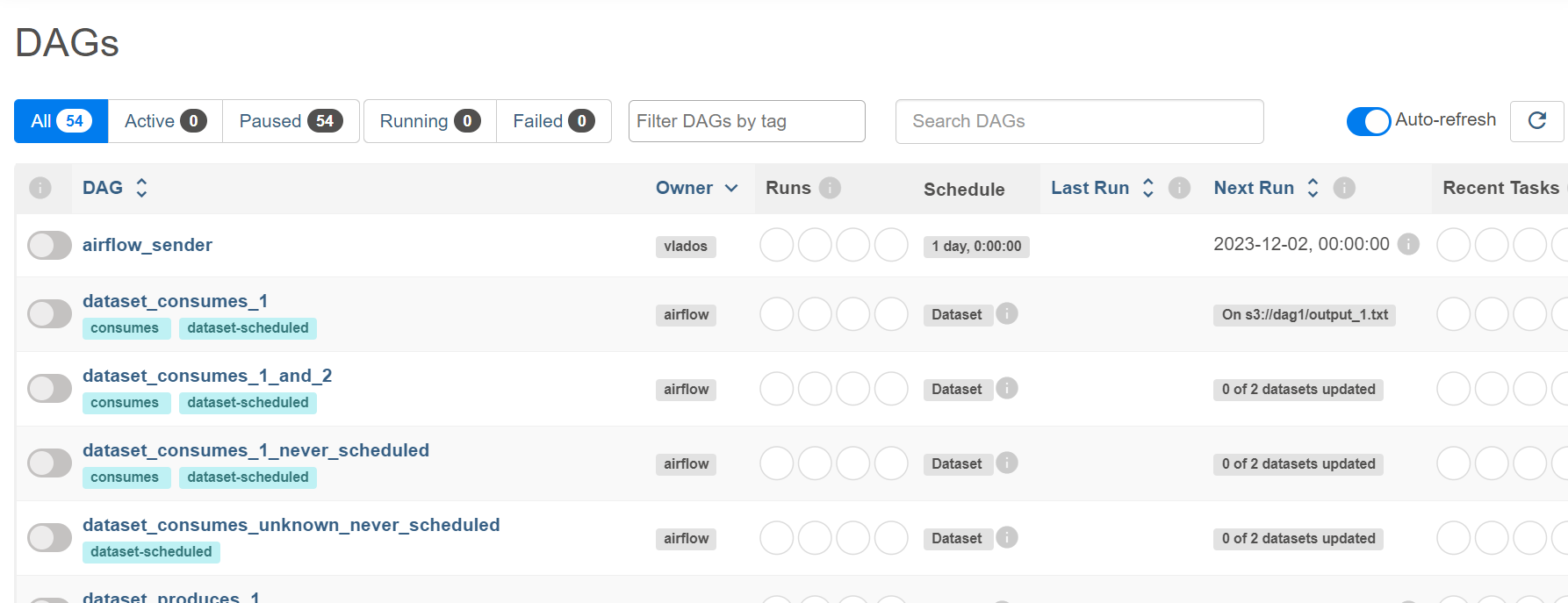
Создаём DAG и сохраняем его по пути C:\docker\_ssau\Prerequisites\airflow\dags



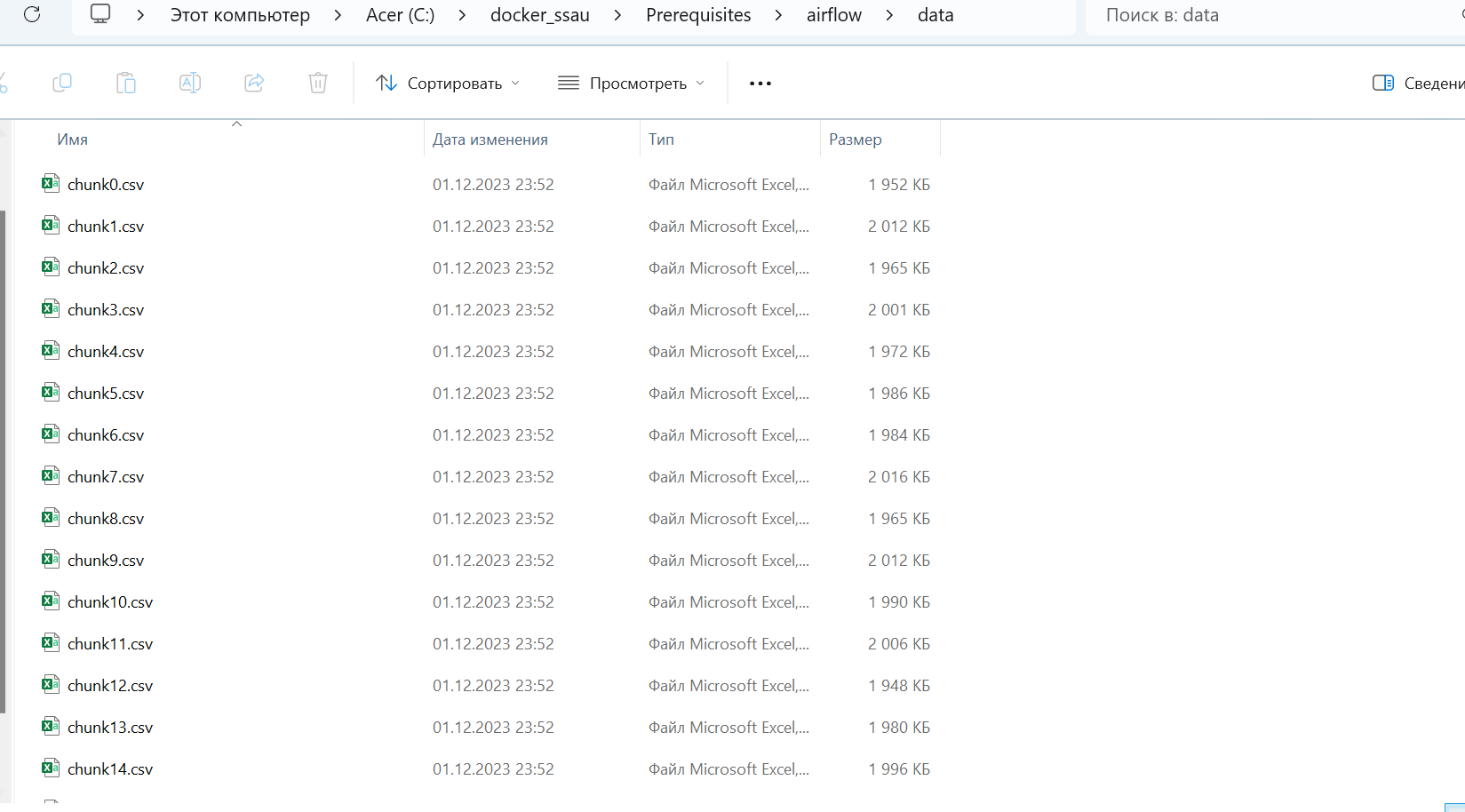
Заходим по ссылке <http://localhost:8080/> на Airflow логин airflow пароль airflow



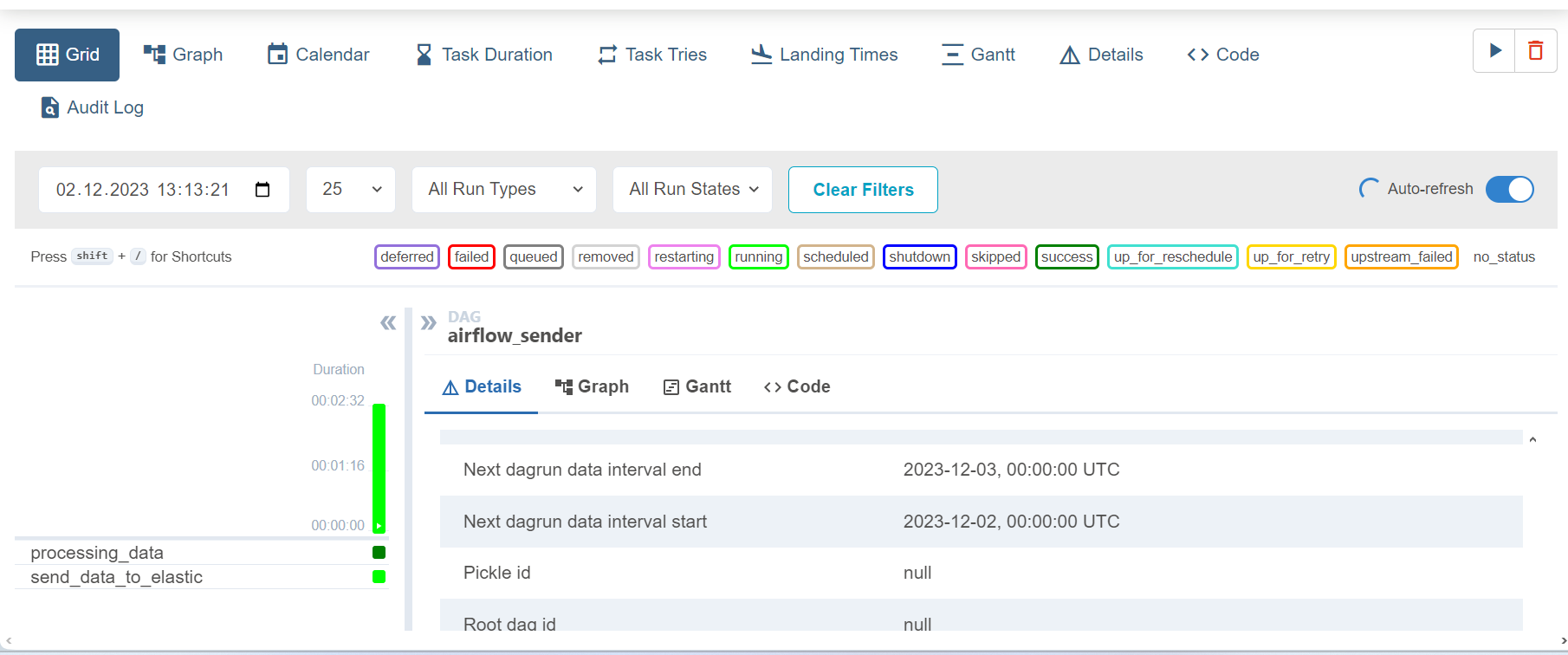
Вот мы и на главной странице



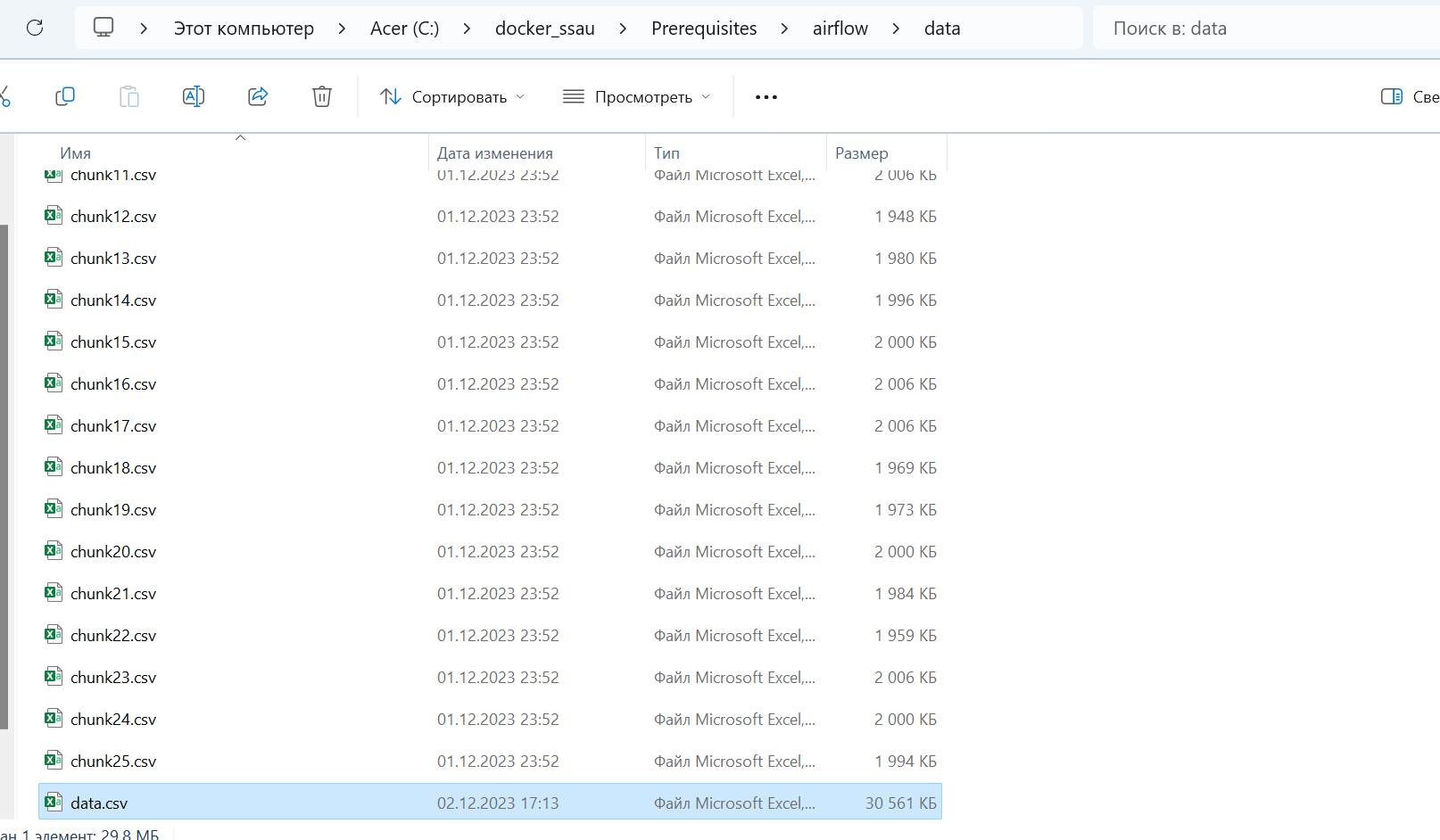
Загружаем доки



Заходим в DAG и запускаем



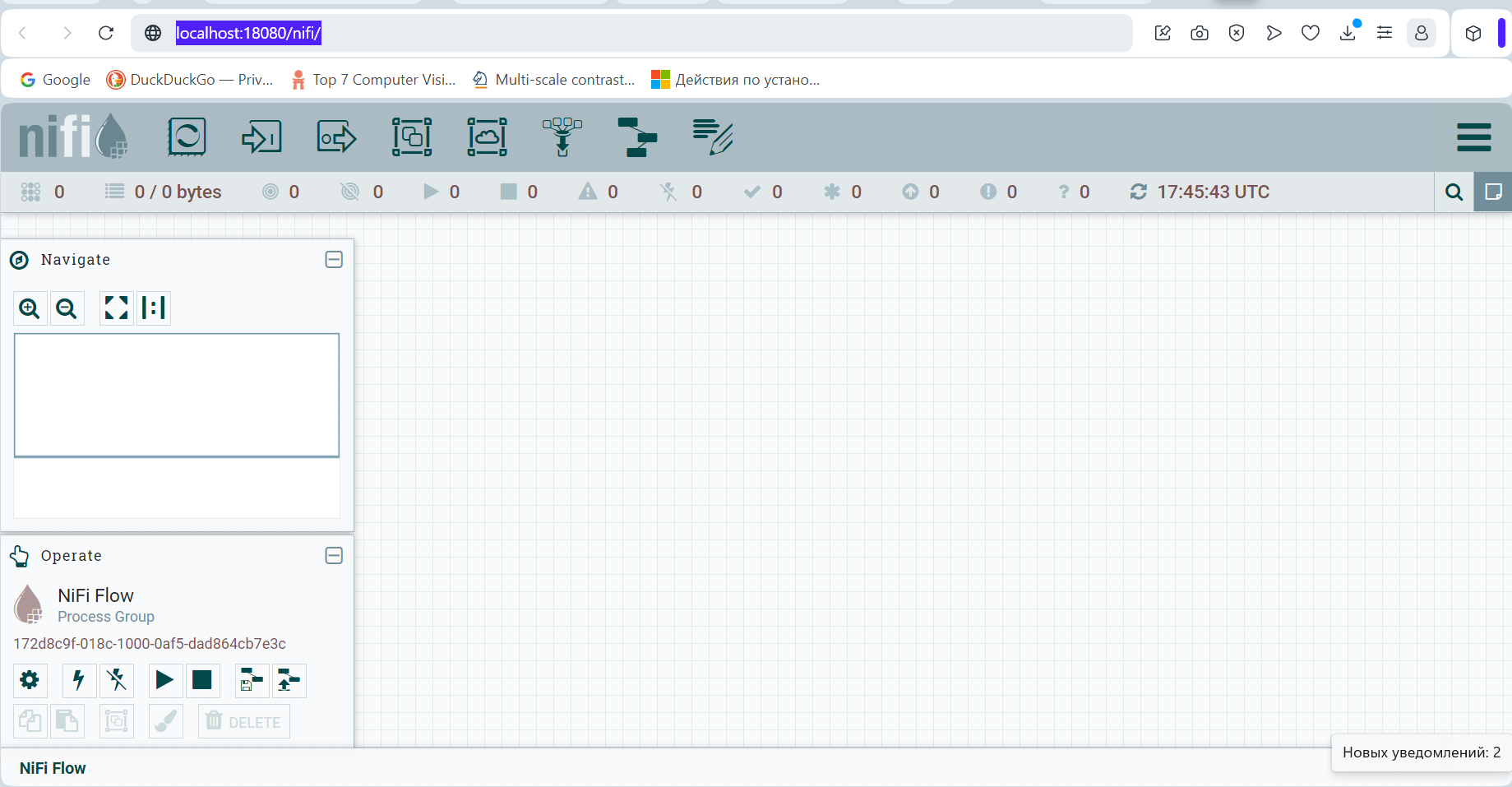
Как видно, файл создался



# НИФИ

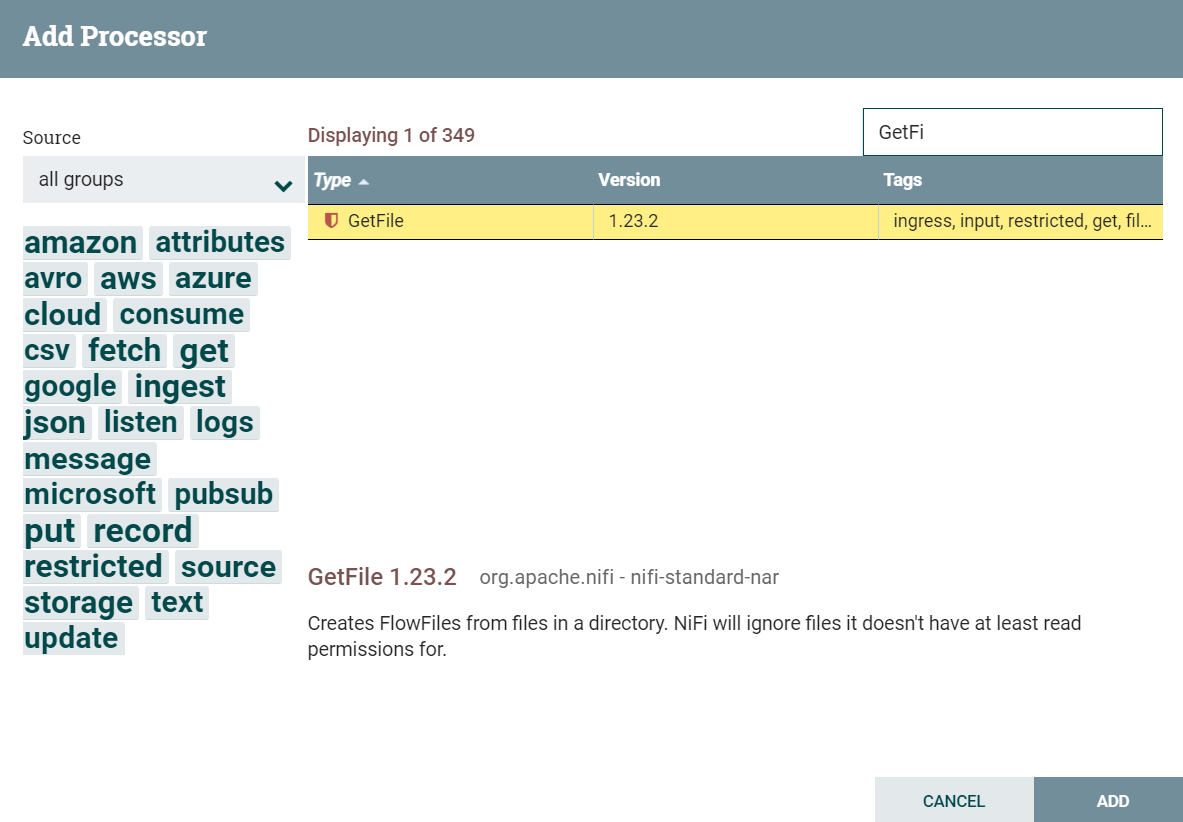
1. Заходим на Nifi

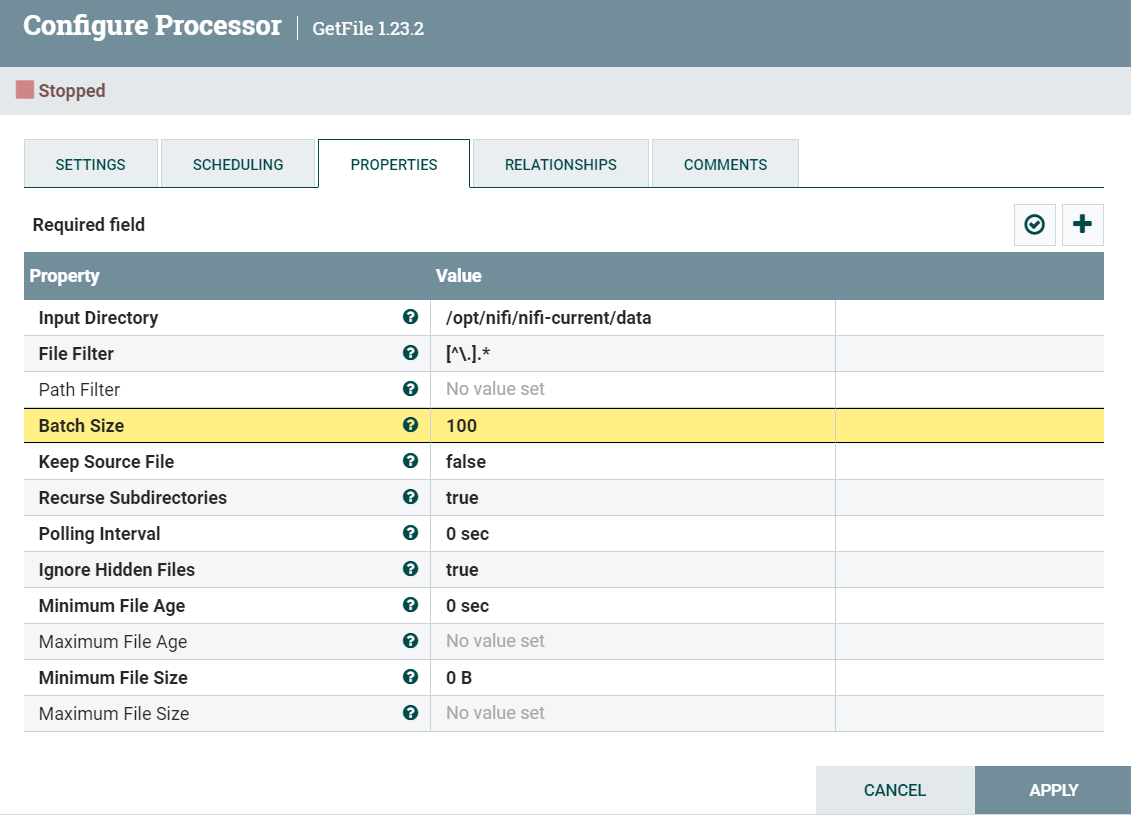
<http://localhost:18080/nifi/>



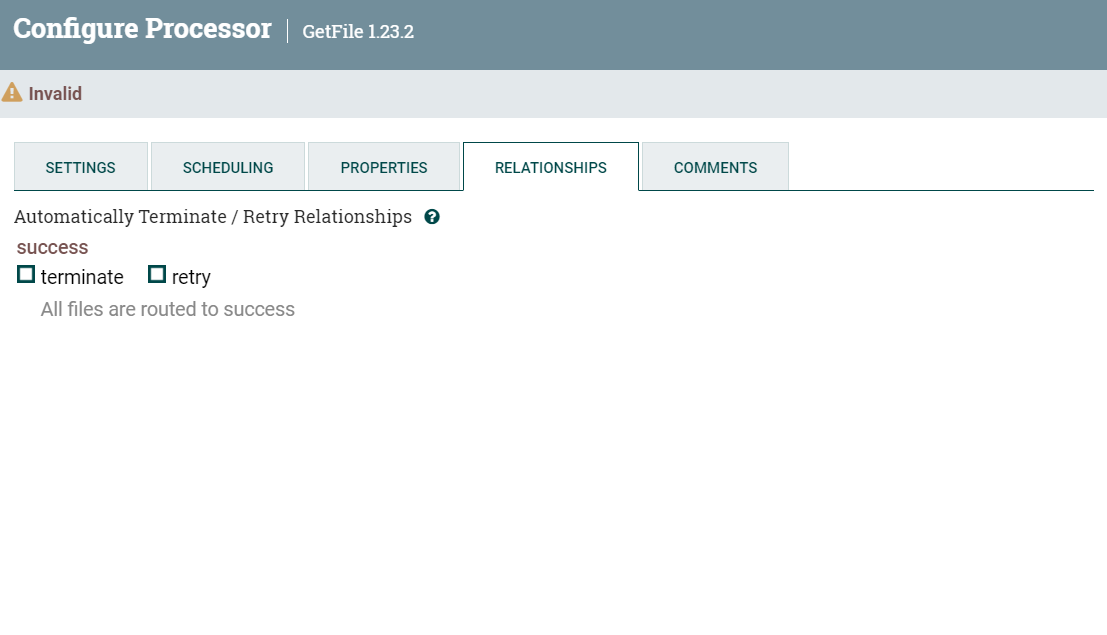
1. **Настраиваем GetFile**

Выбираем тип процессора. Делать это придётся при создании каждого процессора, но покажу единожды

****

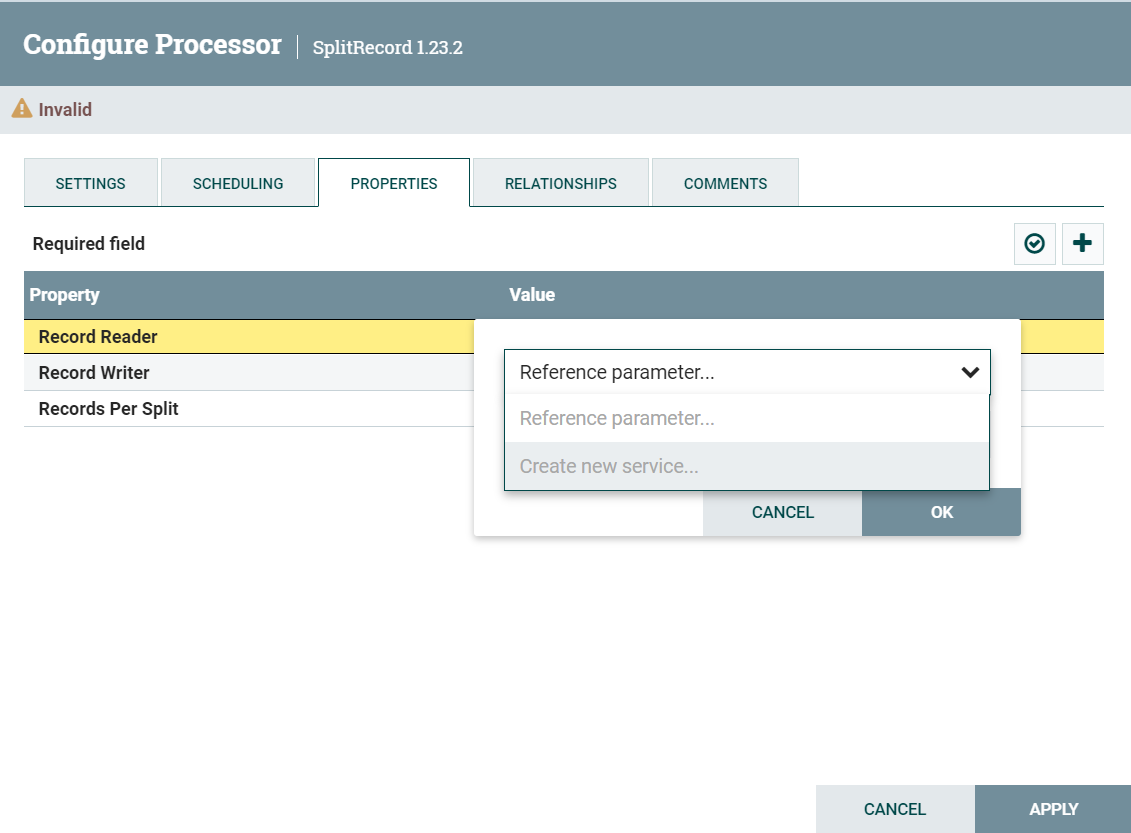


Тут ничего не трогаем

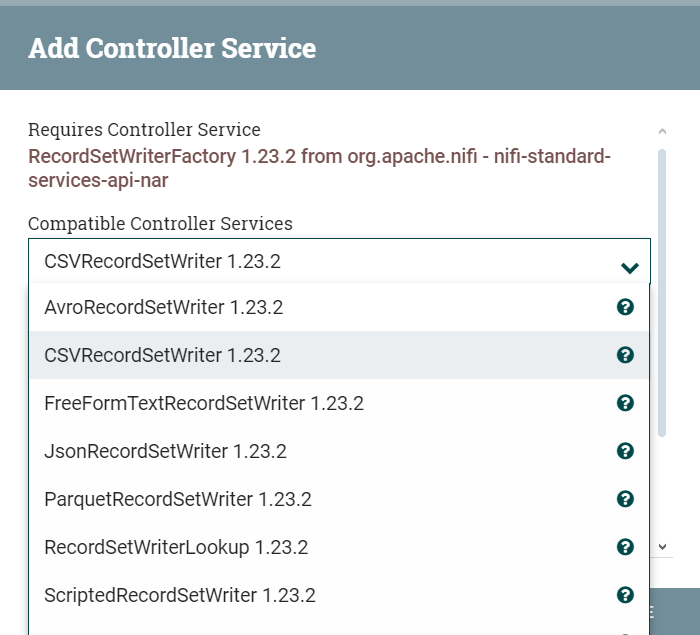


1. **Заполняем splitRecord**

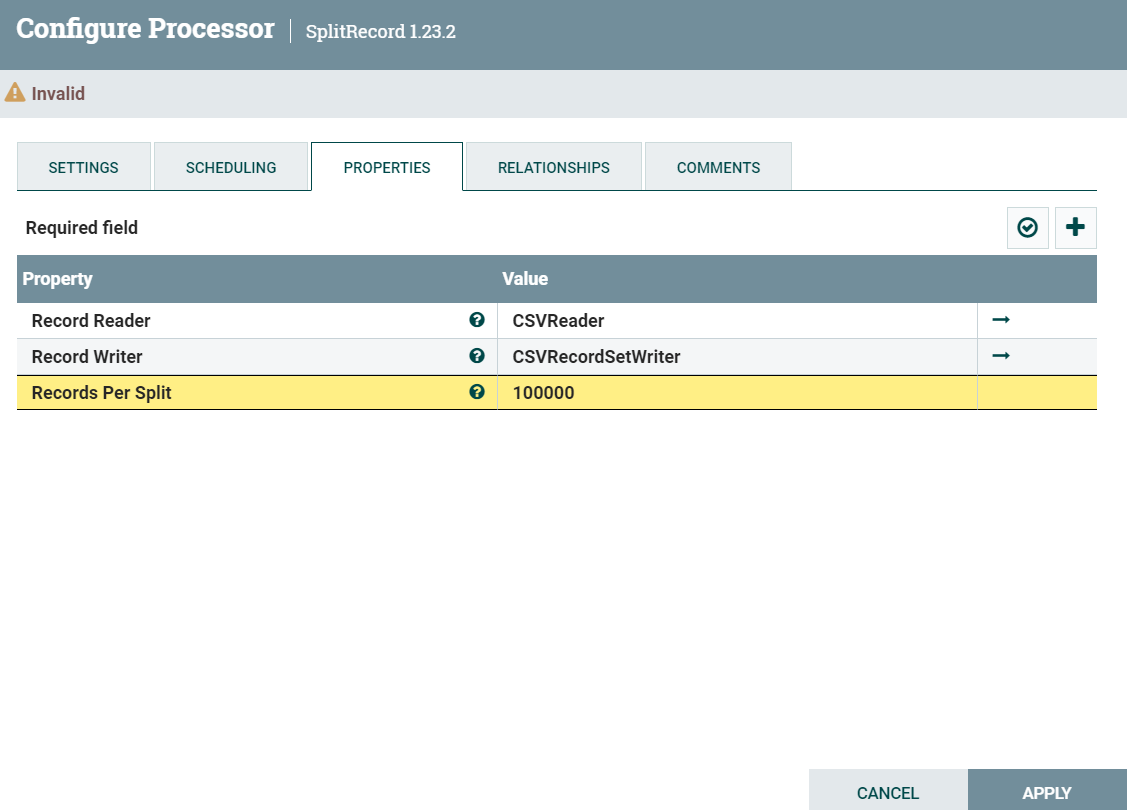
Для этого в Properties заполняем выбираем Create new service



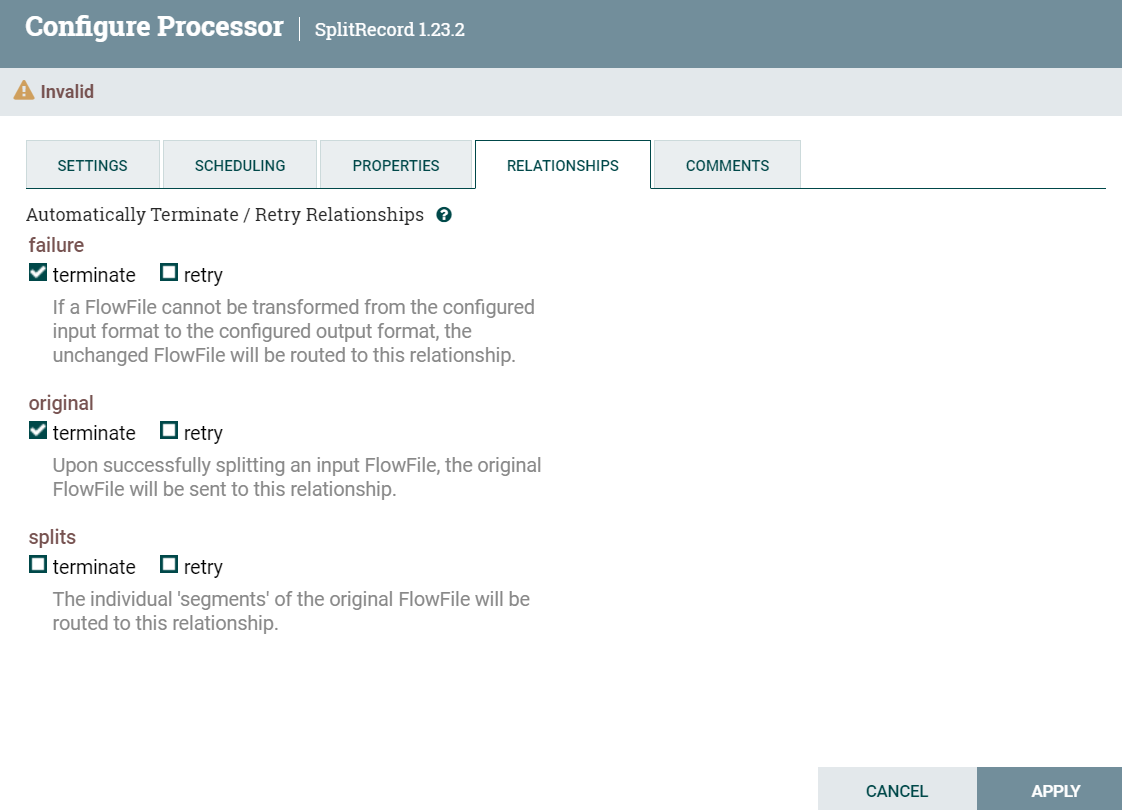
Тут выбираем Controller Service



Итак, вкладка Properties выглядит вот так

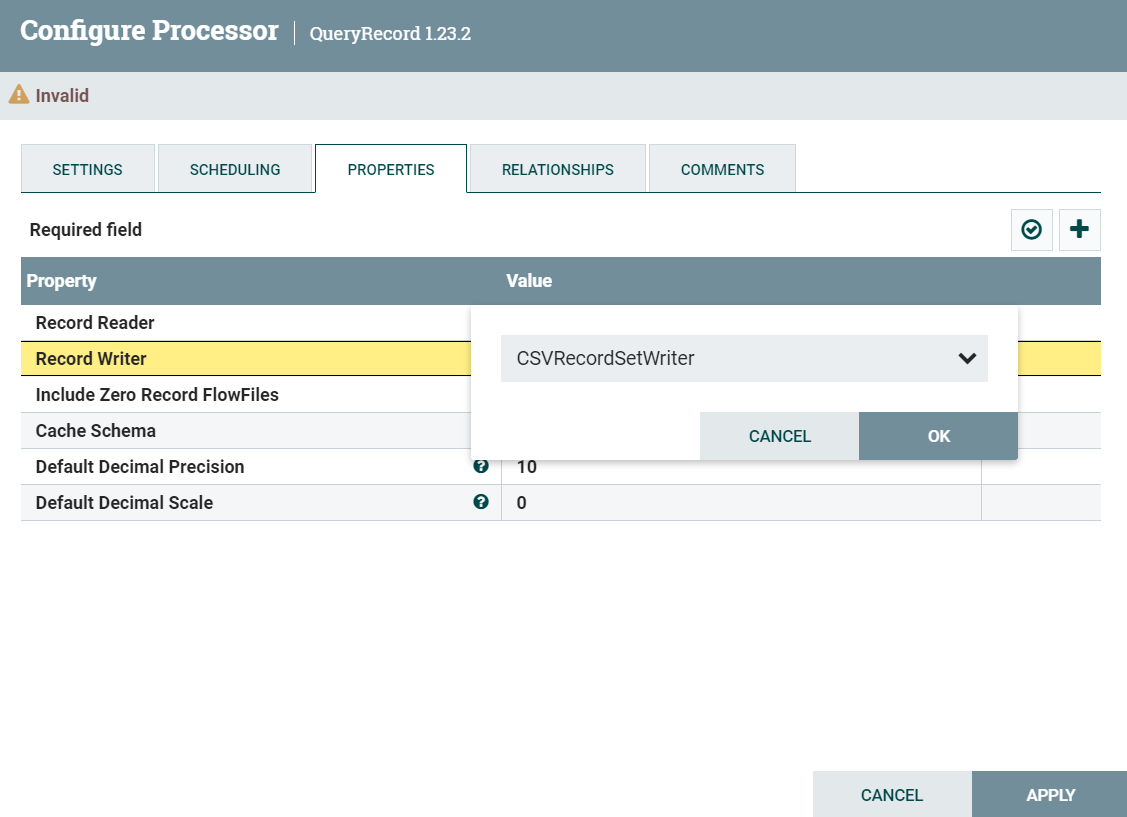


Задаём Relationship

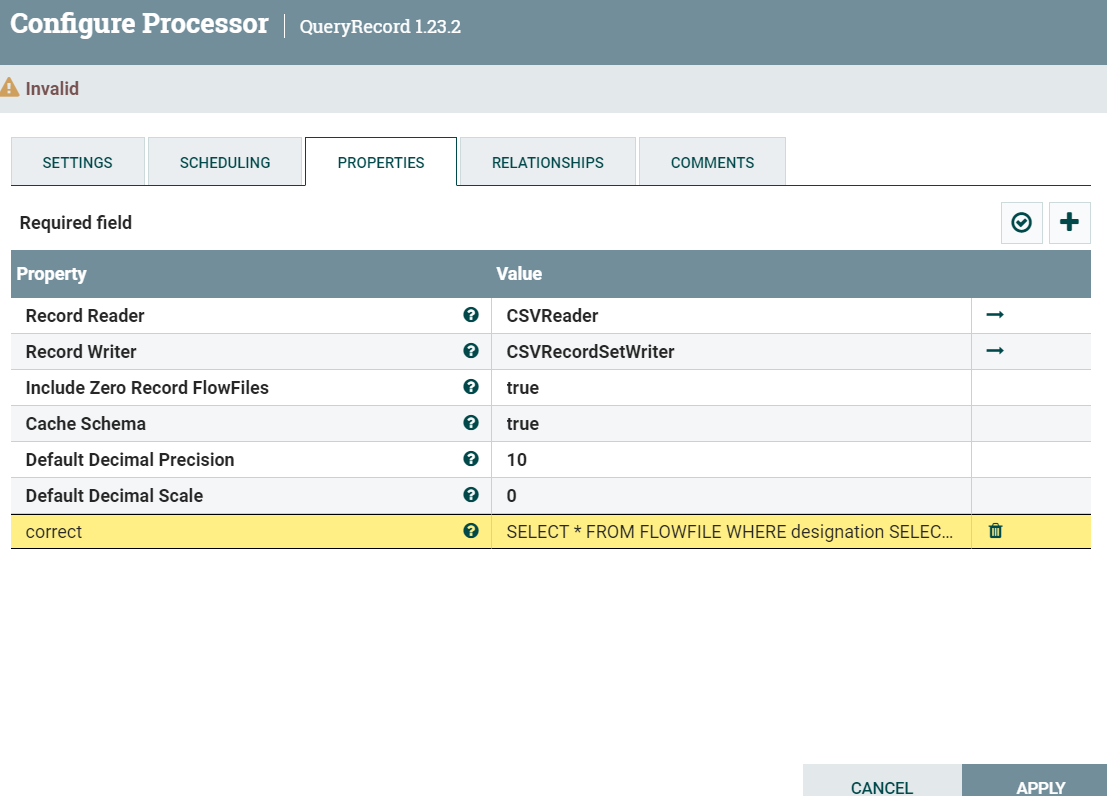


1. **QueryRecord (первый)**

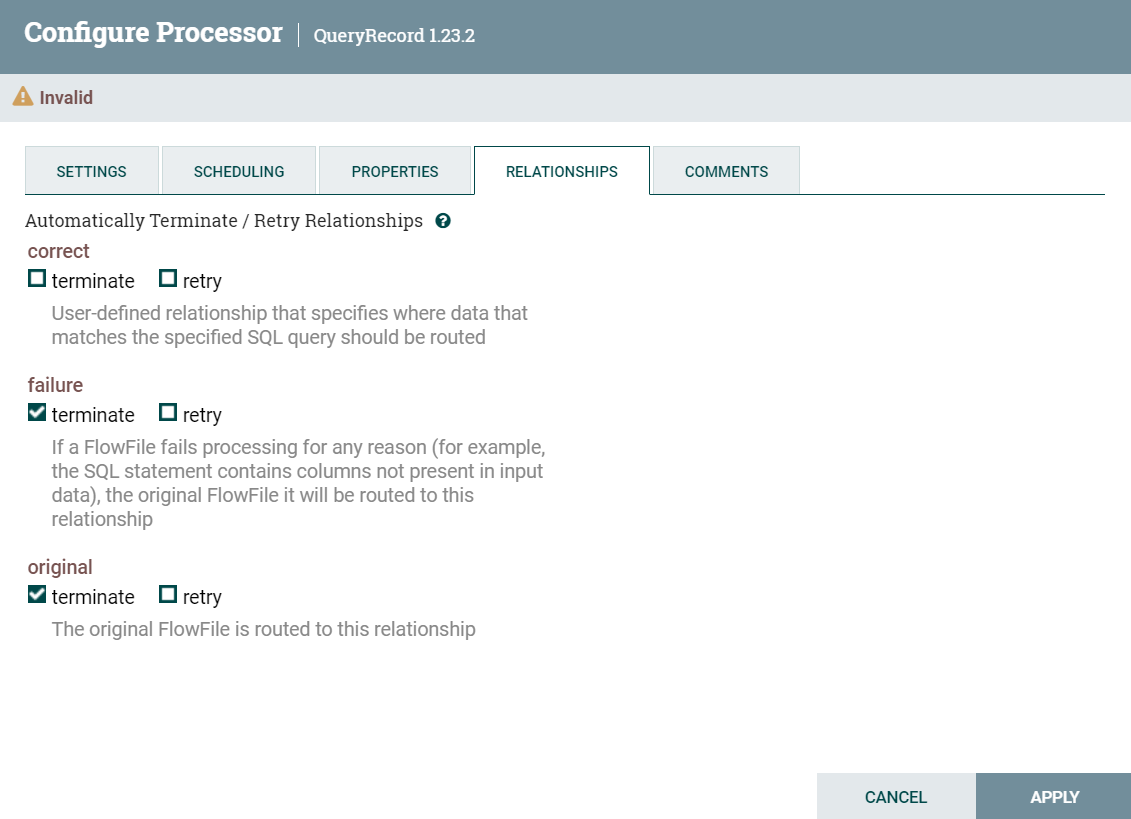
Поскольку мы прежде уже выбрали это значение Value в SplitRecord, нам предлагается выбрать его снова. Так и поступим



Добавим пользовательское значение correct и имеем вид

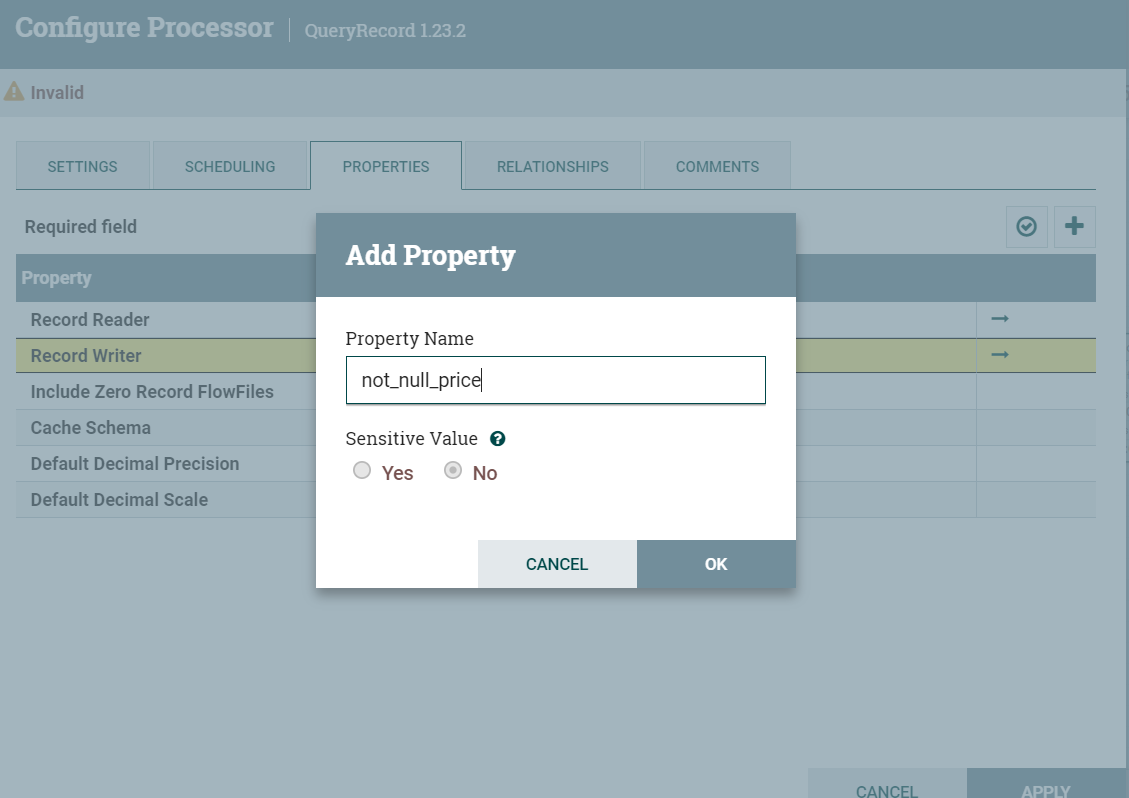


correct = SELECT \* FROM FLOWFILE WHERE designation <> '' AND region\_1 <>''

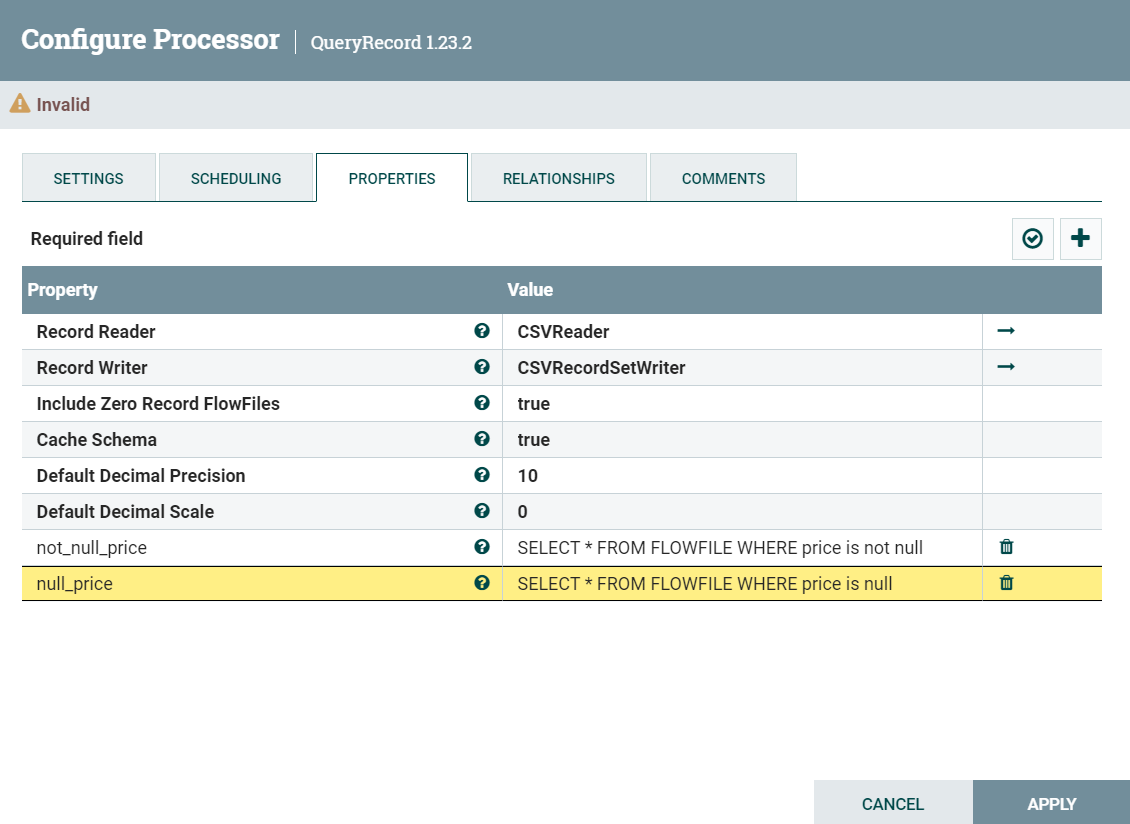


**4 Теперь настраиваем QueryRecord (второй)**

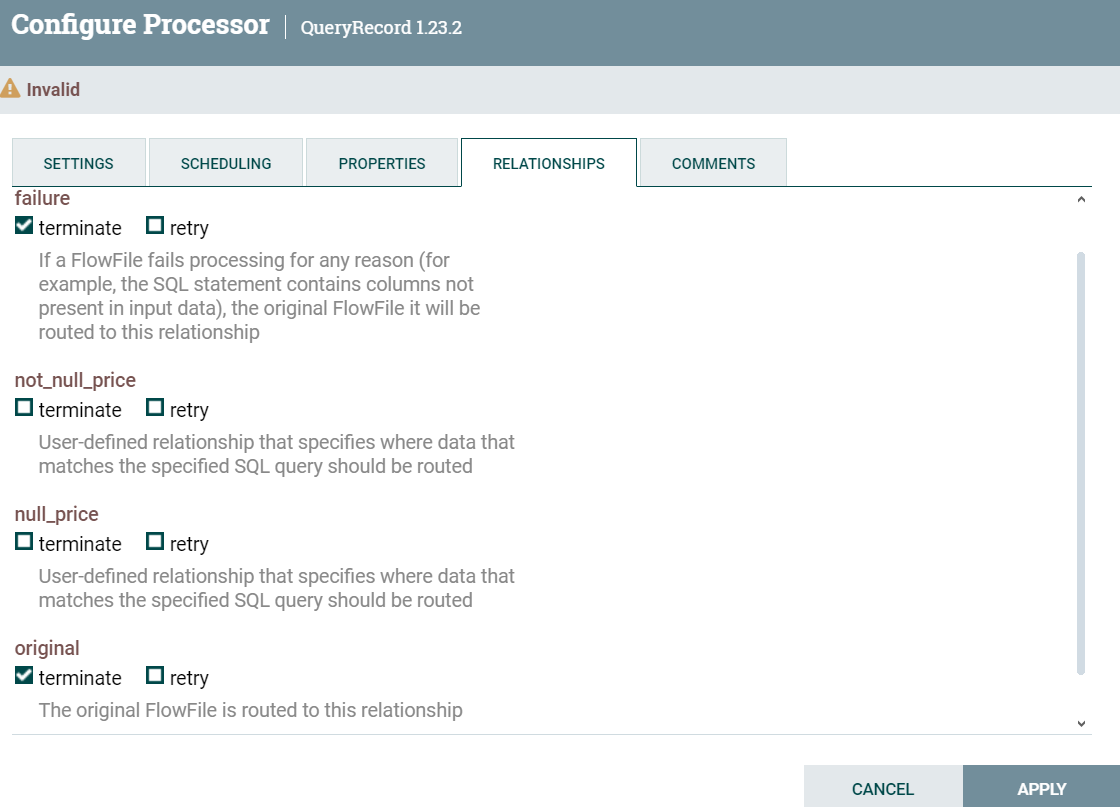
Добавим пару пользовательских значений



Итого у нас получилось

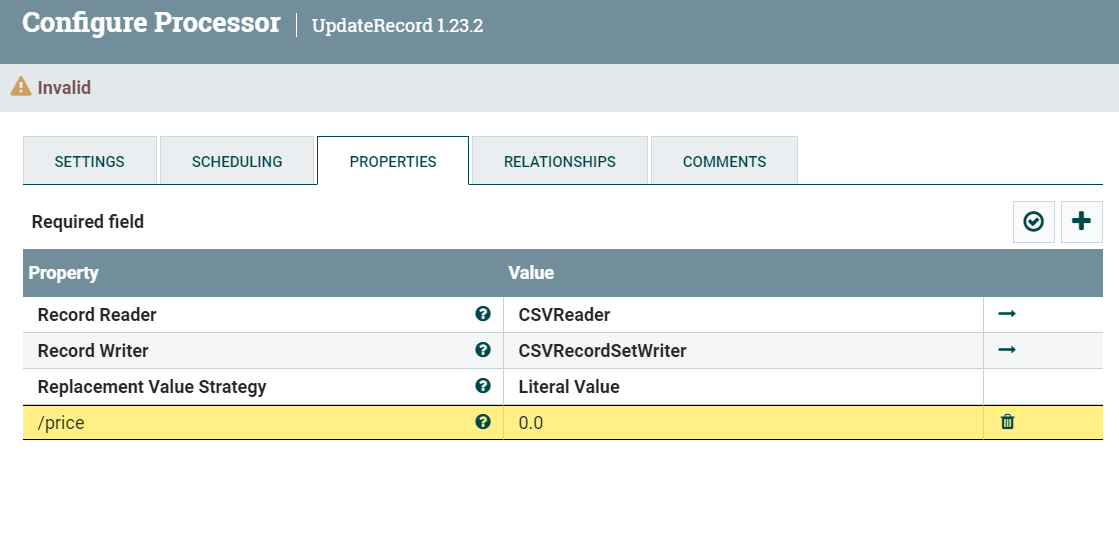


Перезаходим в настройку процессора и понимаем, что в Relationship появилось два новых значения, но трогать мы их не будем

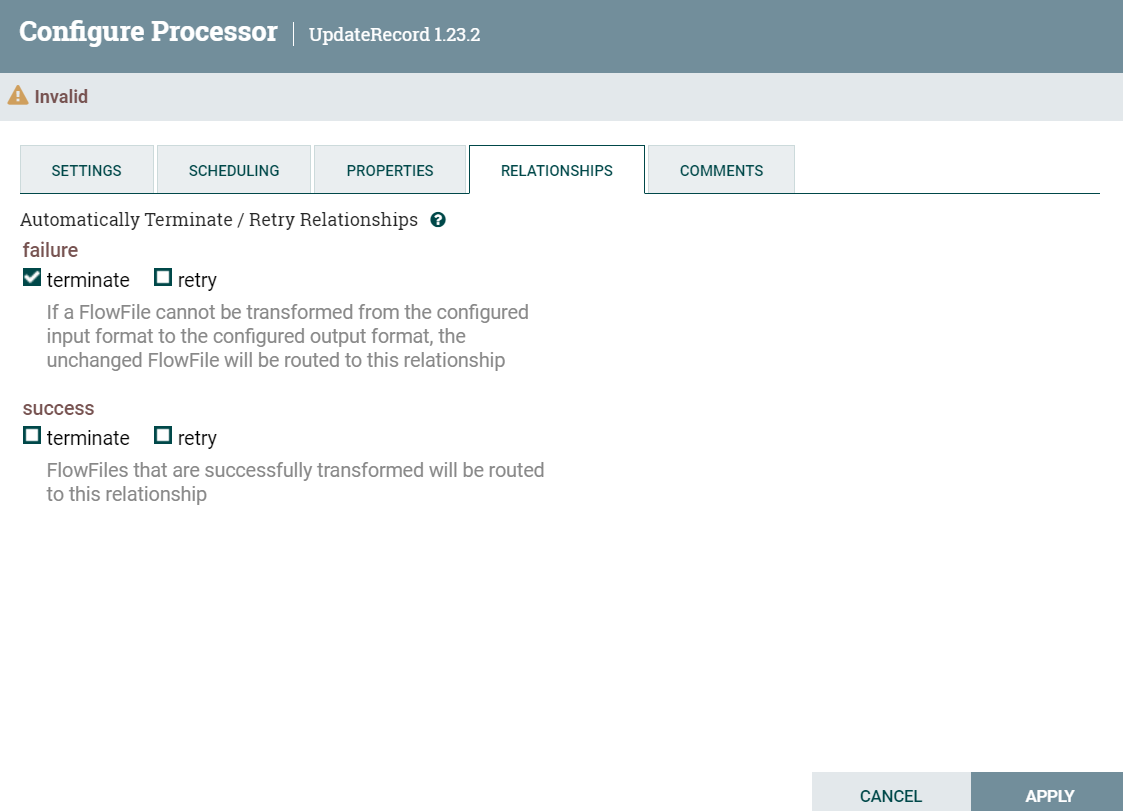


1. **Настраиваем UpdateRecord**

Properties

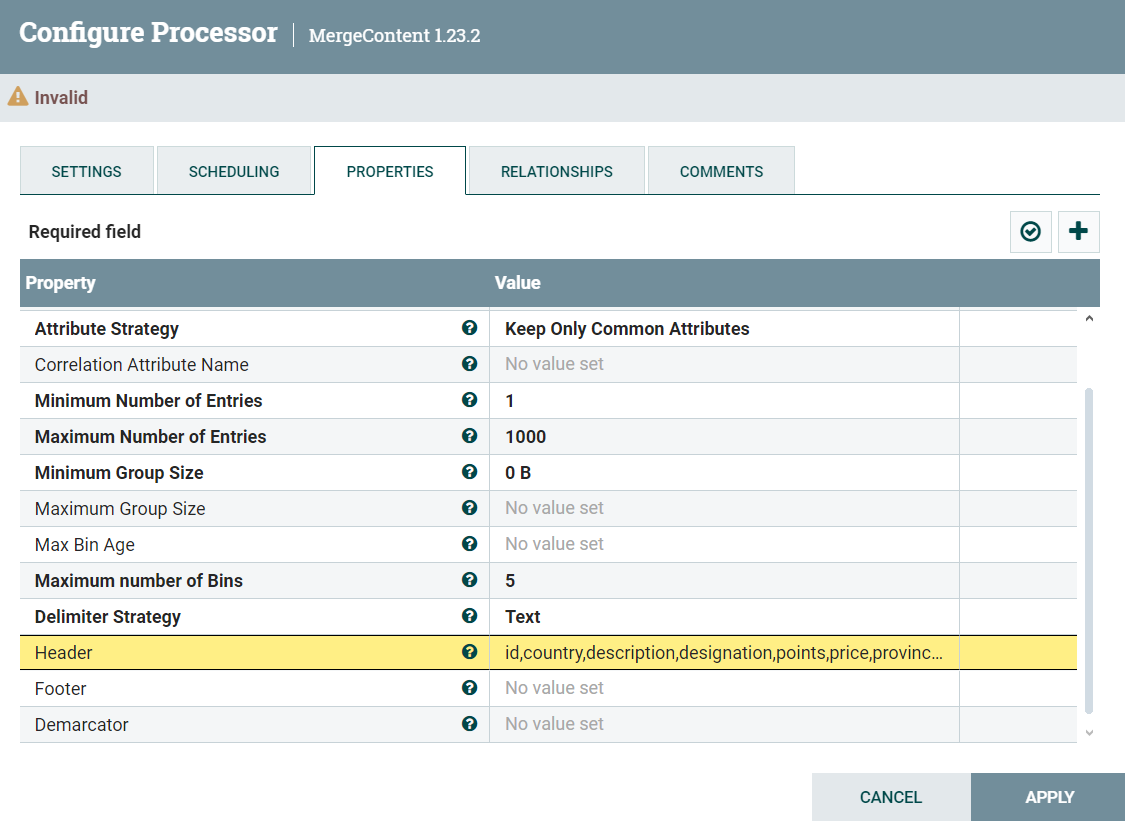
****

Relationships (На этот раз дополнительная строчка по имени пользовательского Property не создалась)

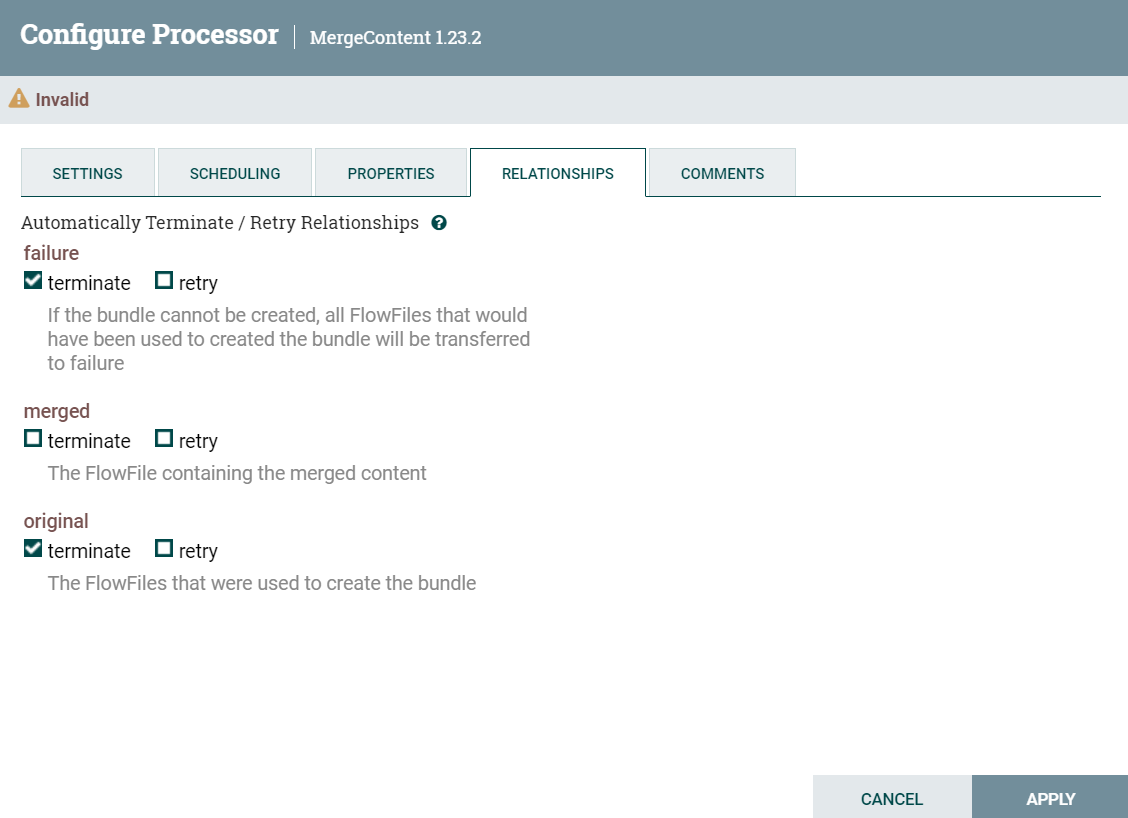
****

1. **MergeContent**

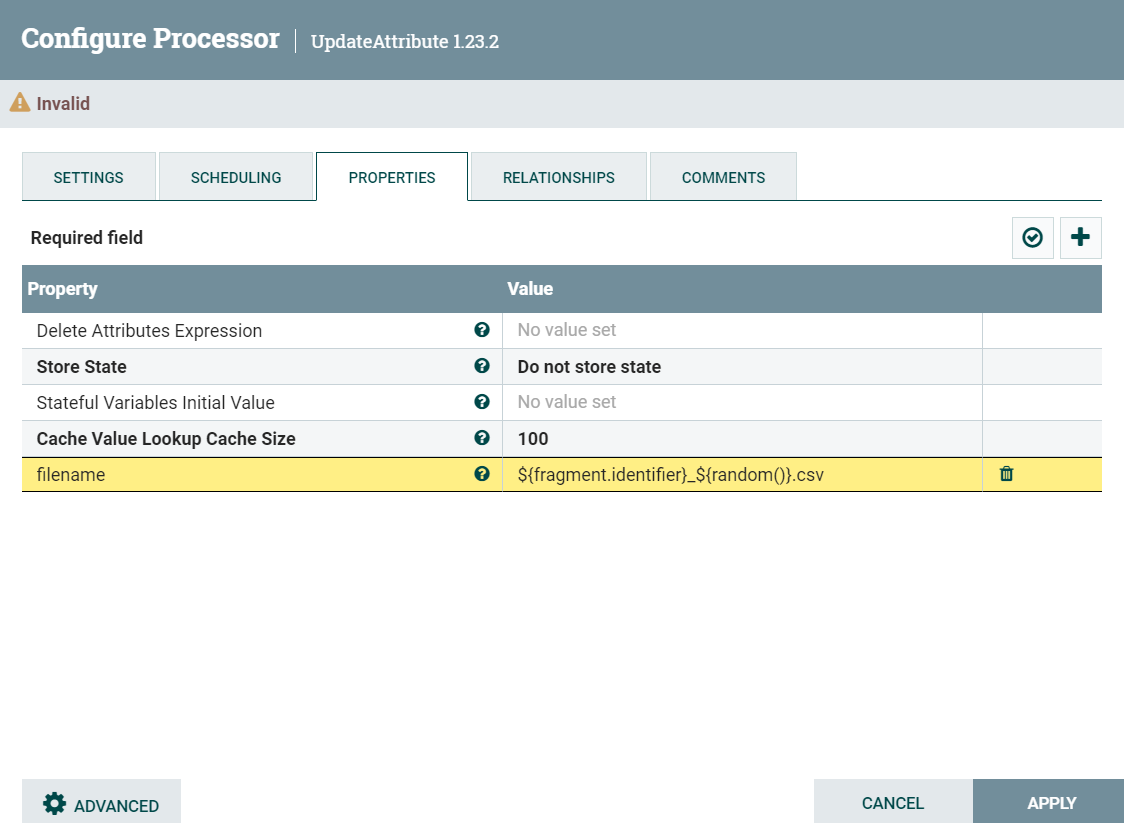
В Properties настраиваем Delimiter Strategy

****

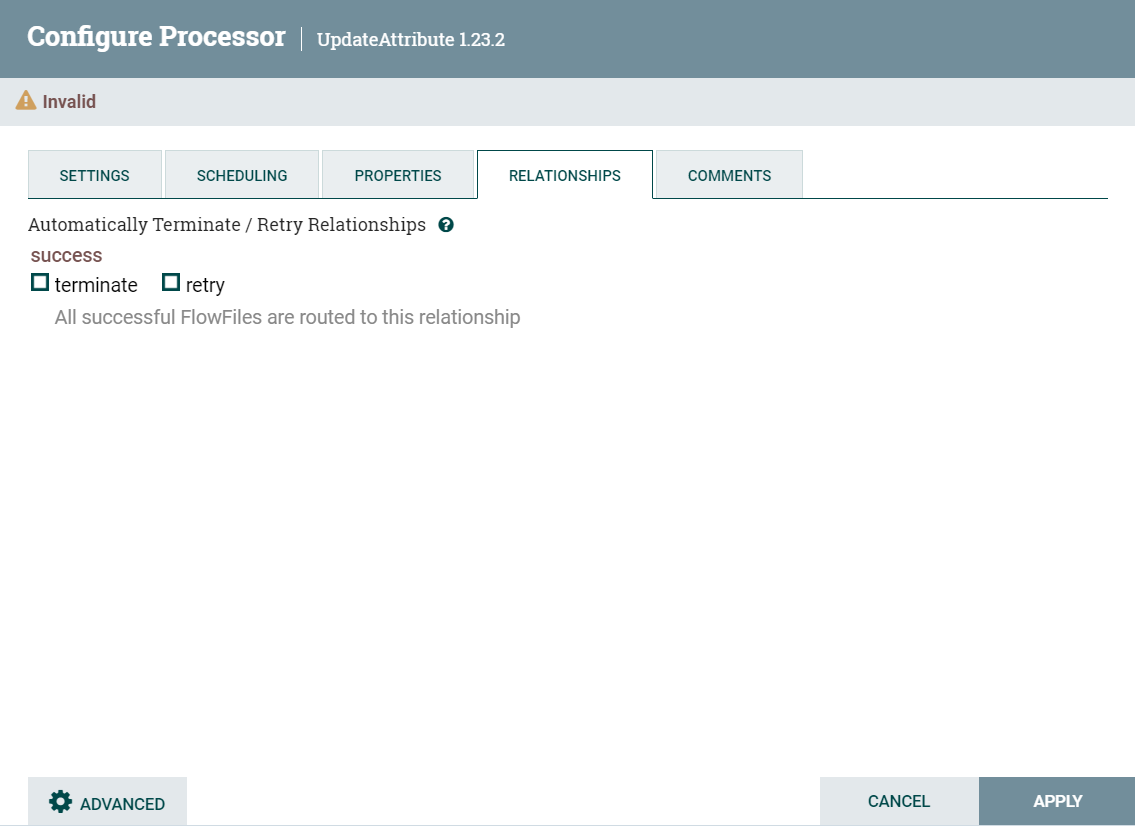
В Header значение:id,country,description,designation,points,price,province,region\_1,region\_2,taster\_name,taster\_twitter\_handle,title,variety,winery

****

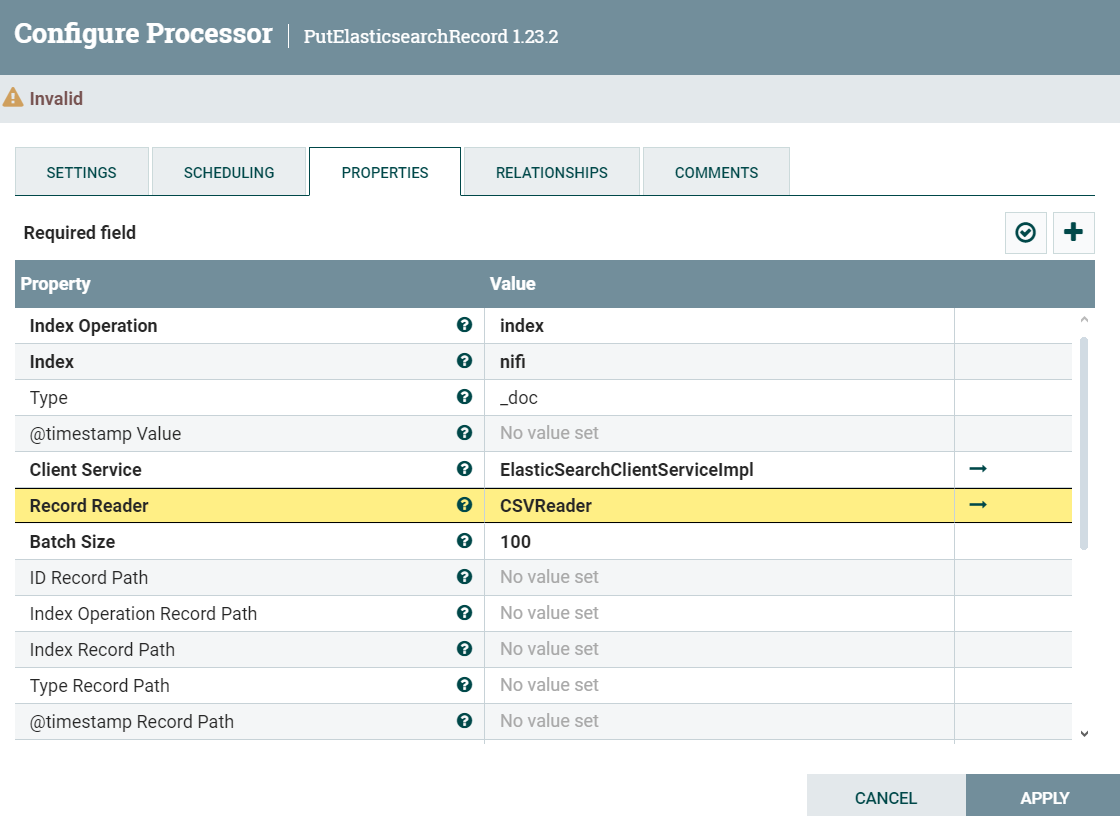
**7 UpdateAttribute**

****

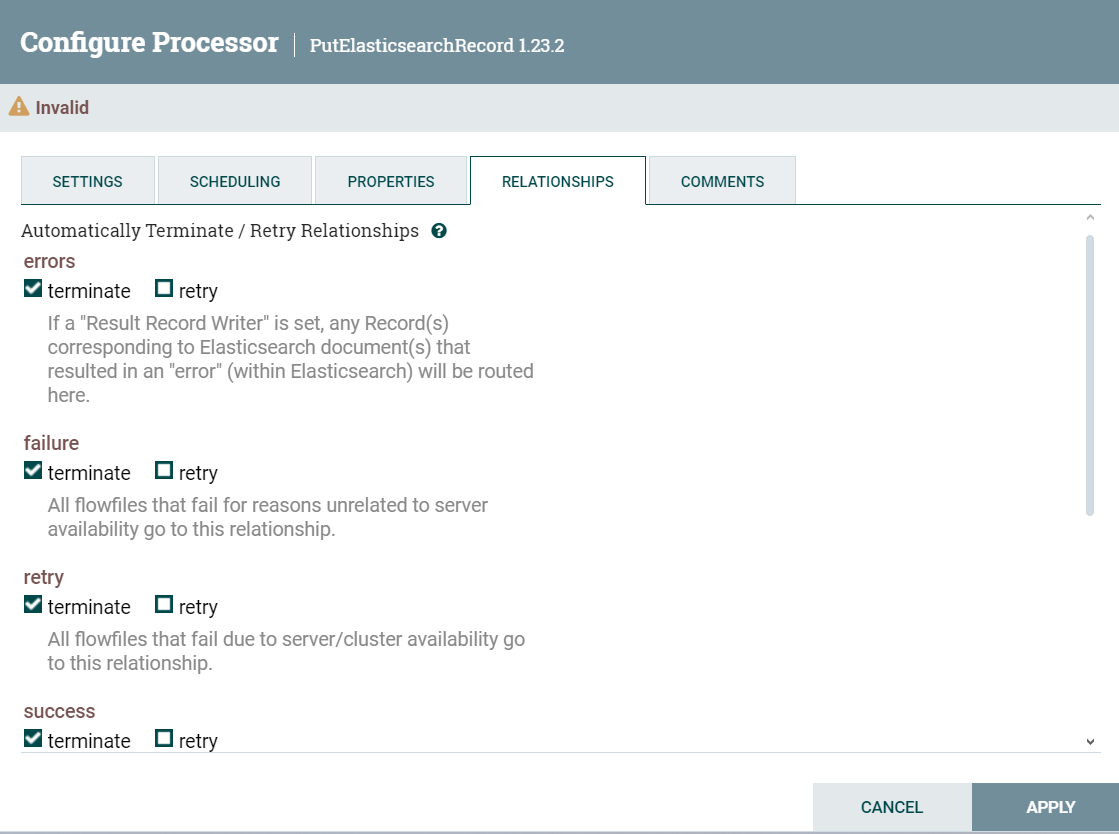
filename -= ${fragment.identifier}\_${random()}.csv

****

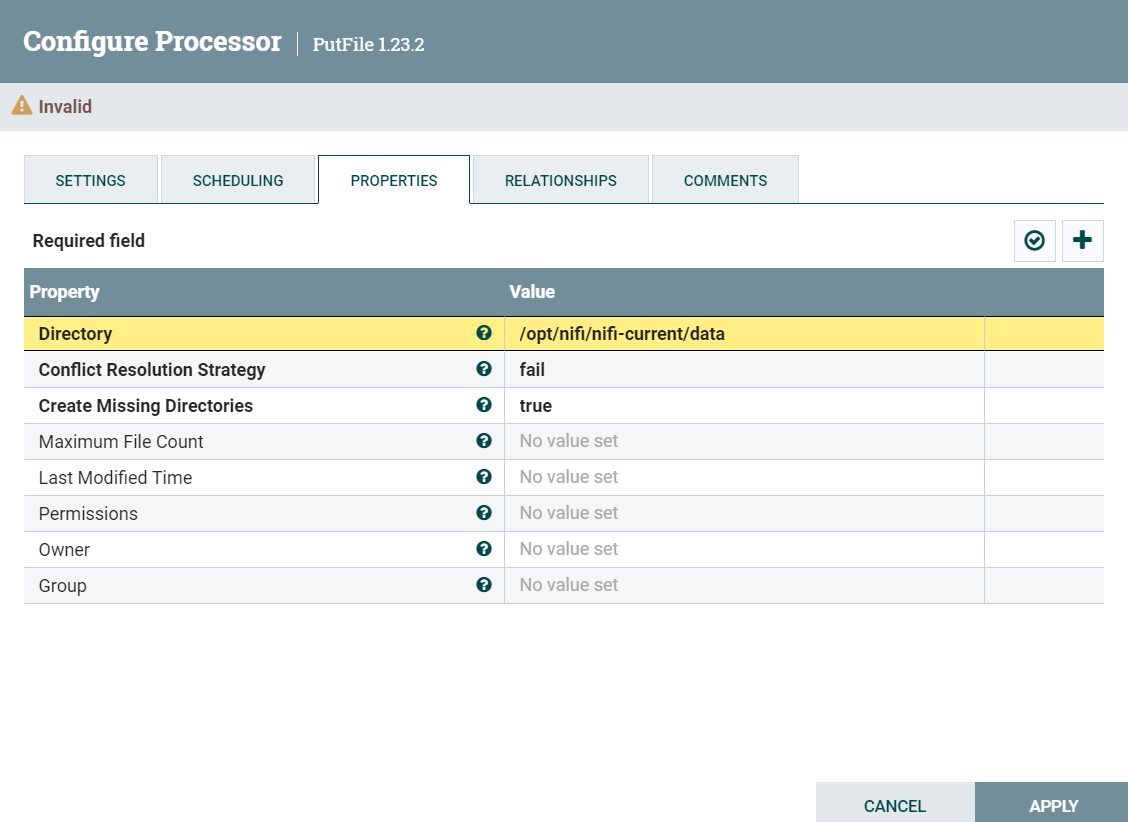
**8 PutElasticsearchRecord**

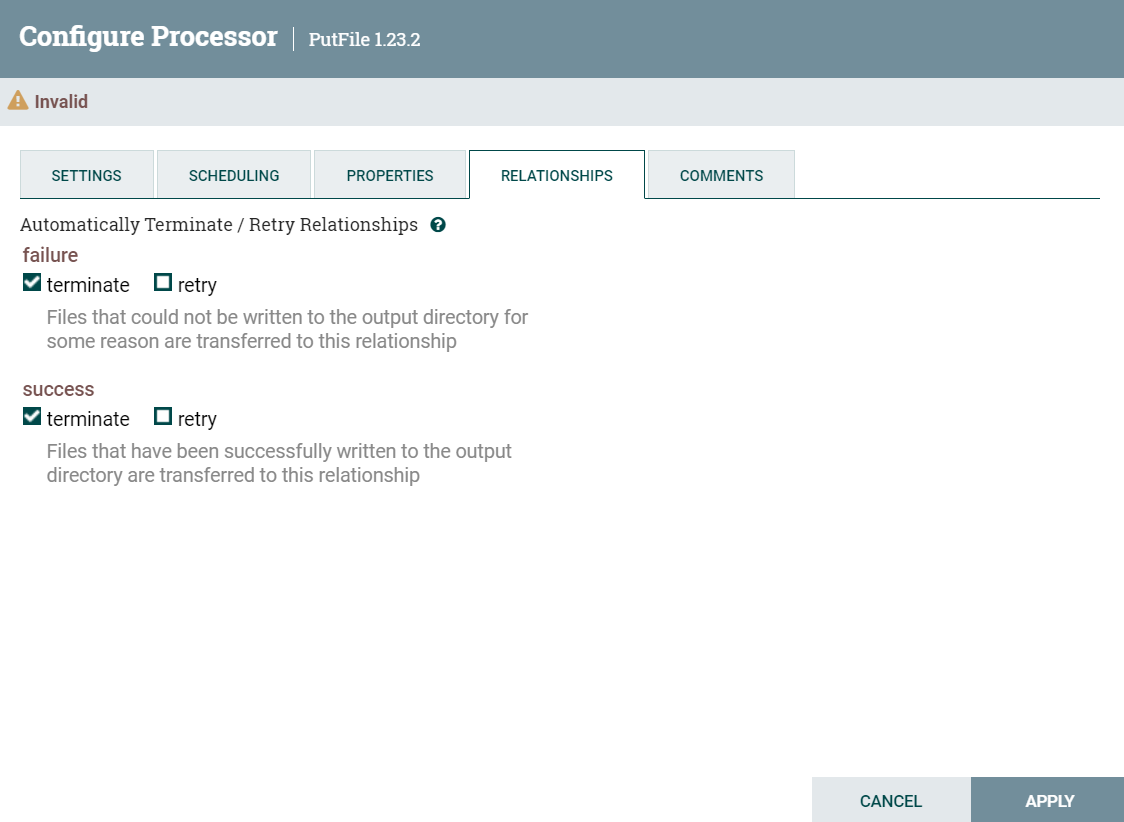
****

Client Service = ElasticSearchClientServiceImpl

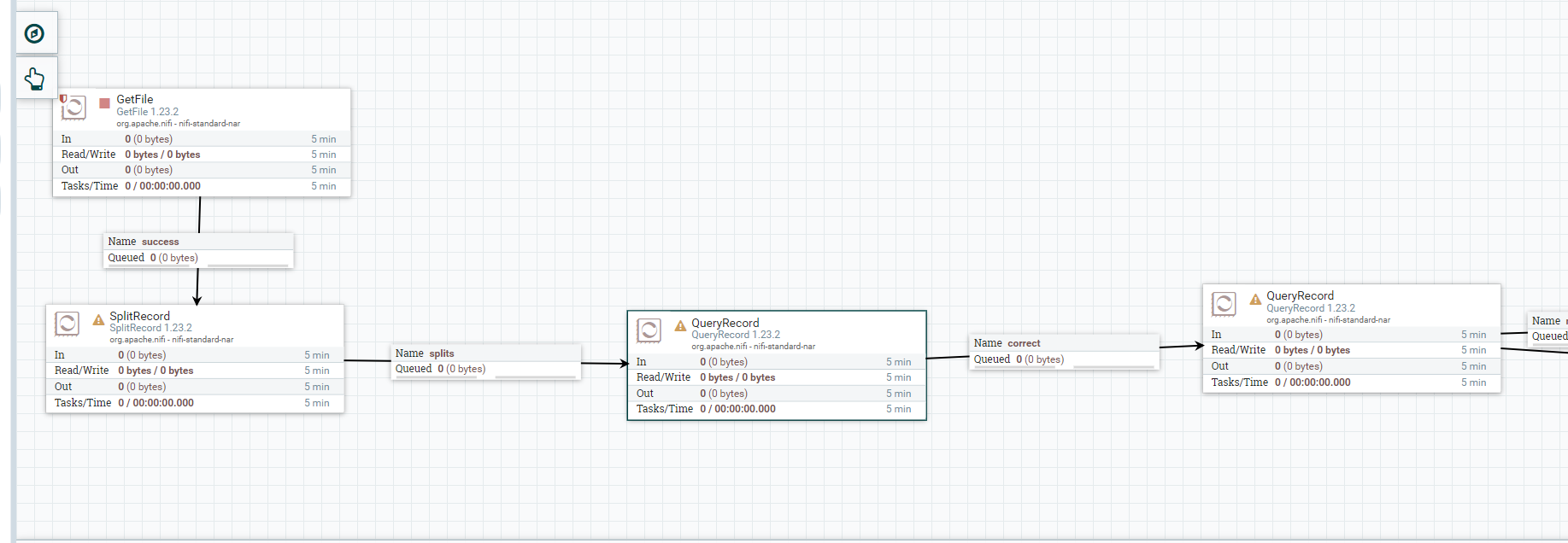


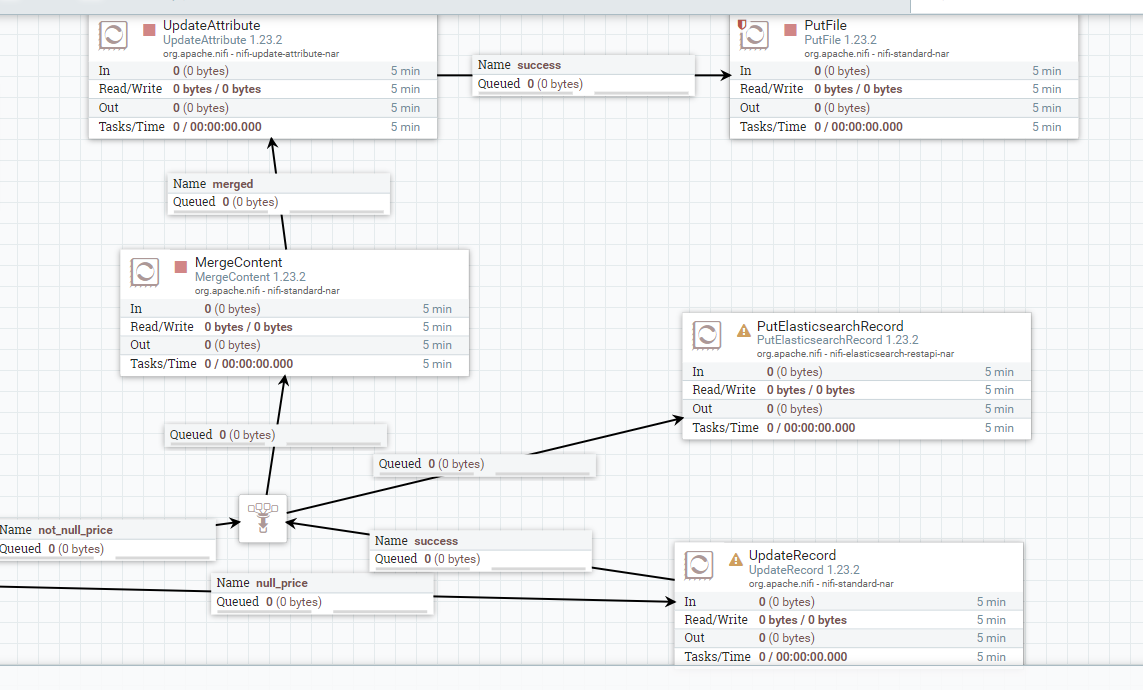
**9 PutFile**

****

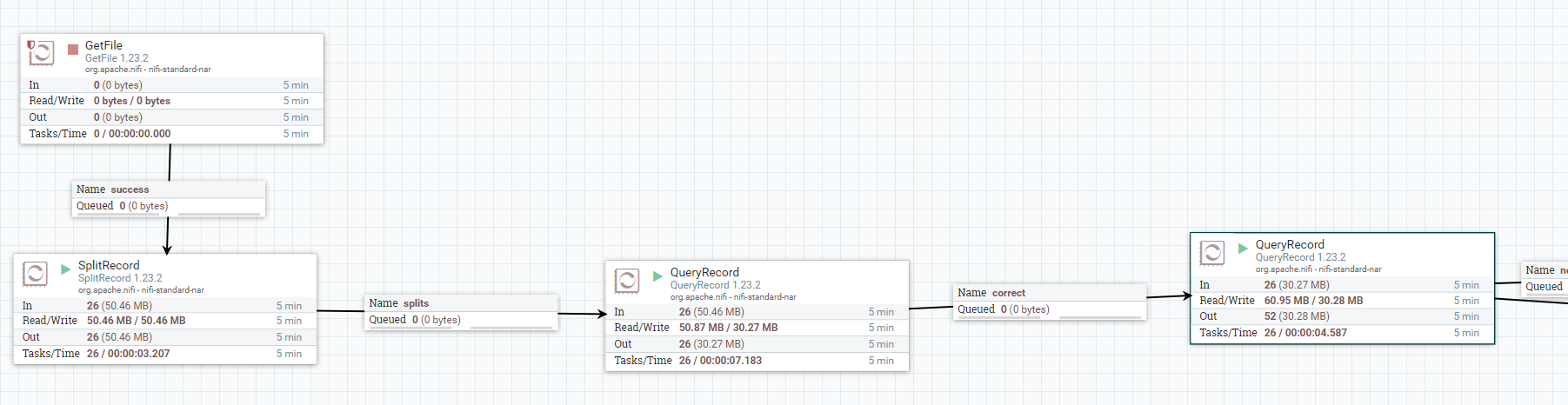
****

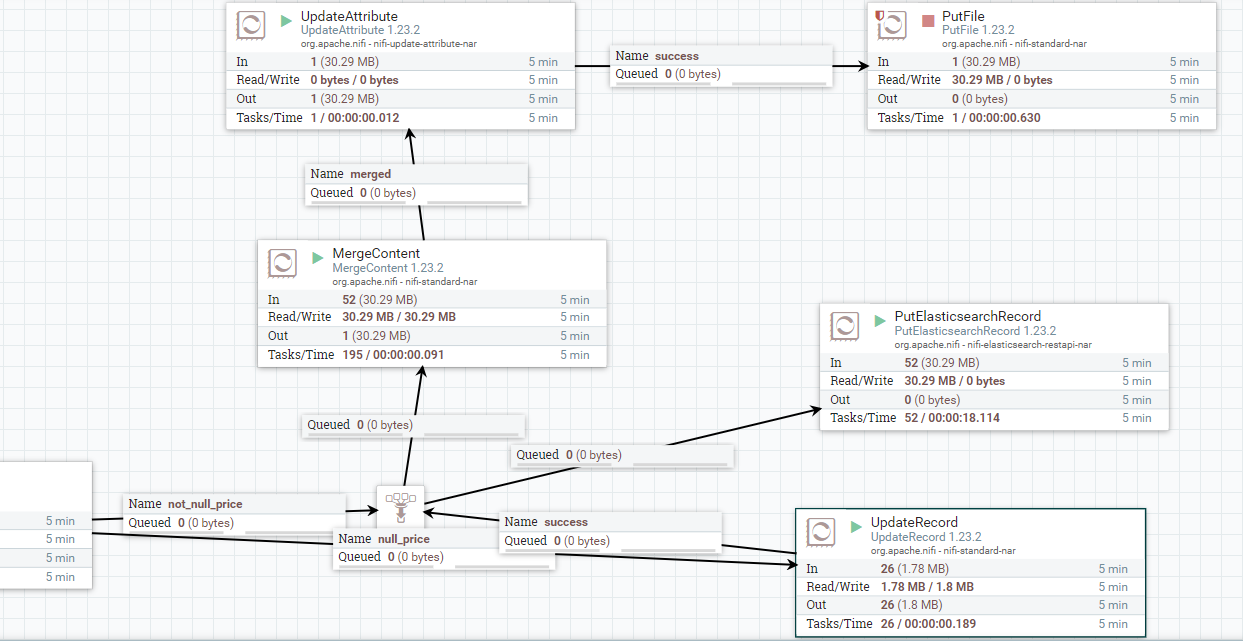
**ИТОГО ПОЛУЧИЛАСЬ СЛЕДУЮЩАЯ СХЕМА (НА ДВУХ СКРИНШОТАХ ВВИДУ ОСОБЕННОСТЕЙ МАСШТАБИРОВАНИЯ AIRFLOW)**

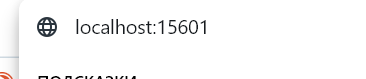
****

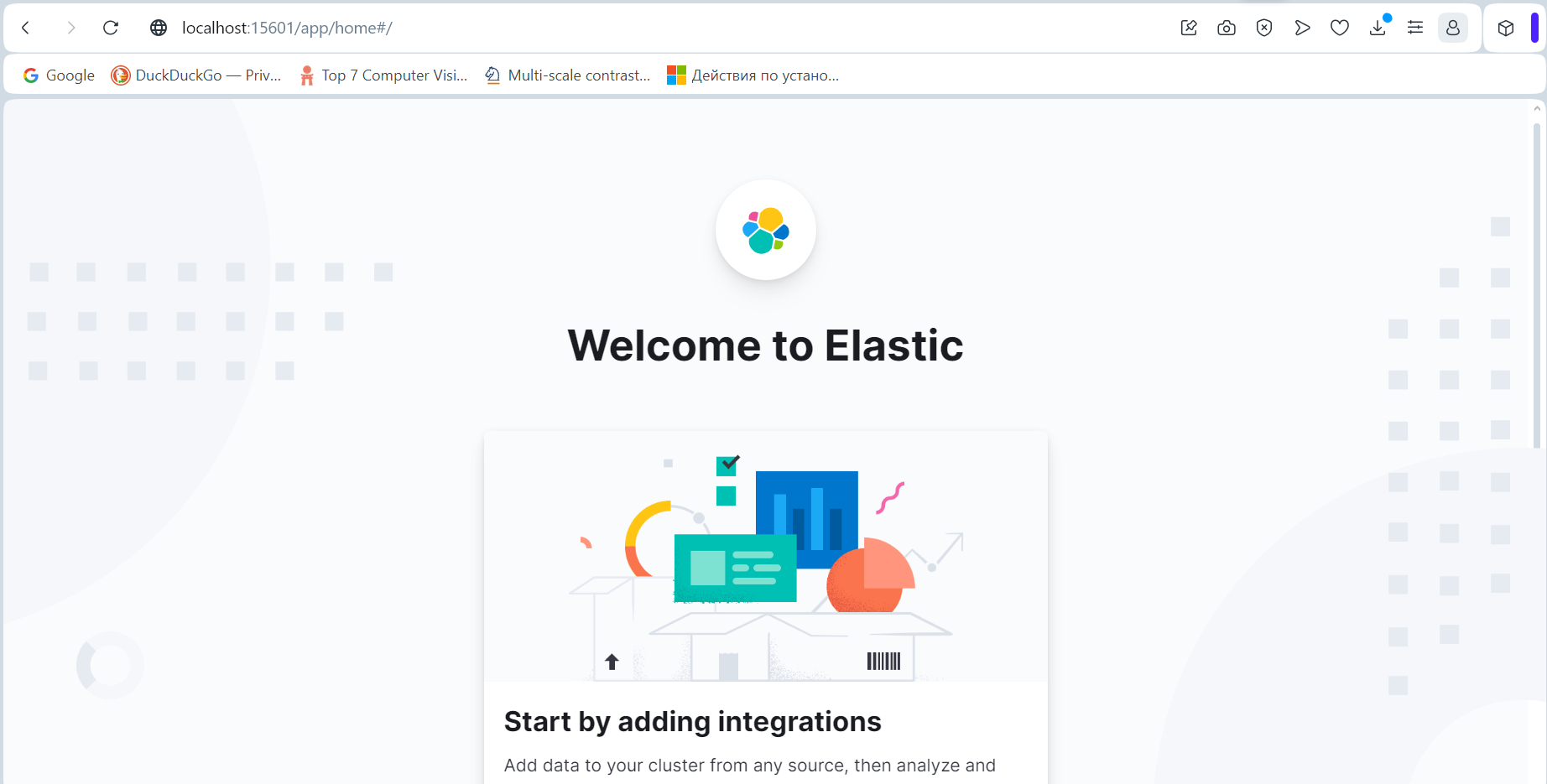
****

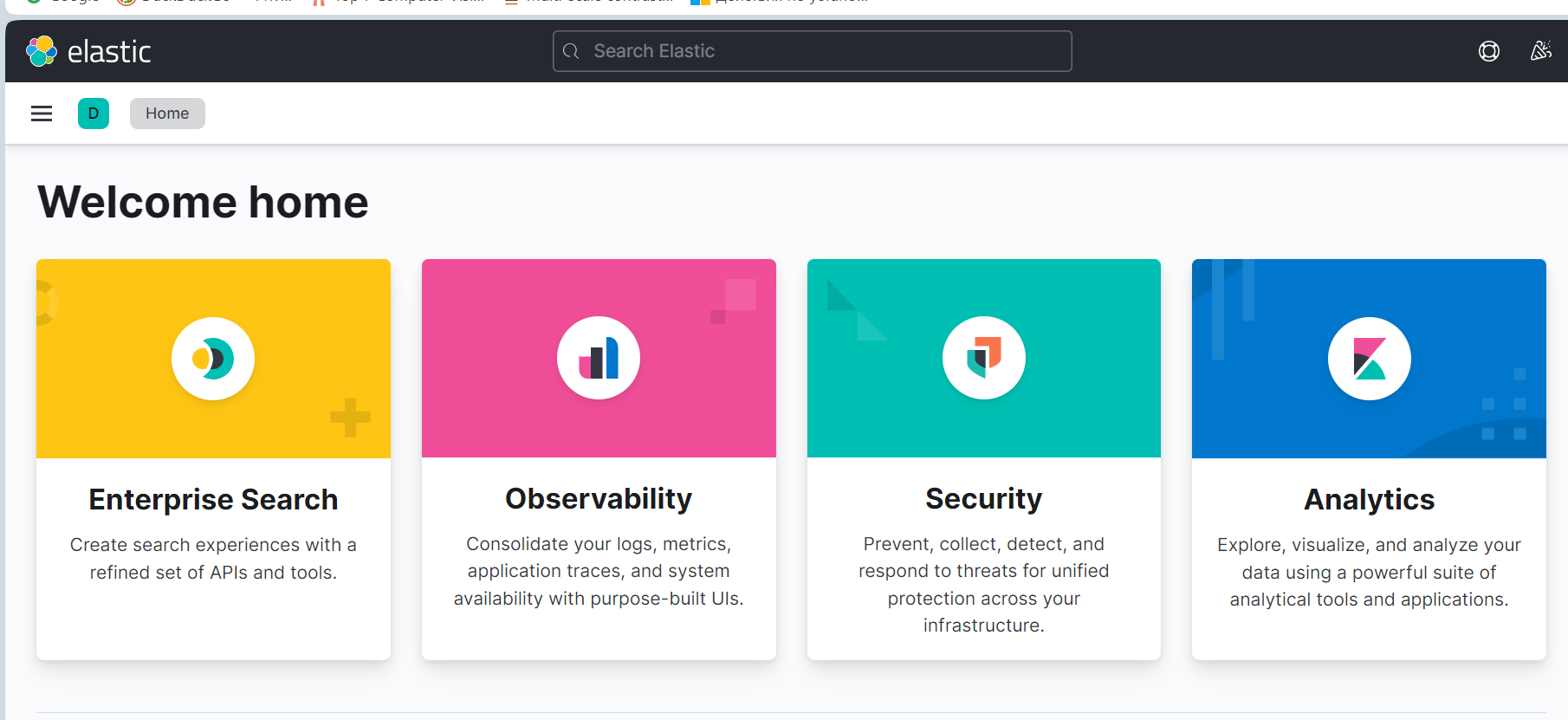
**Произведён запуск**

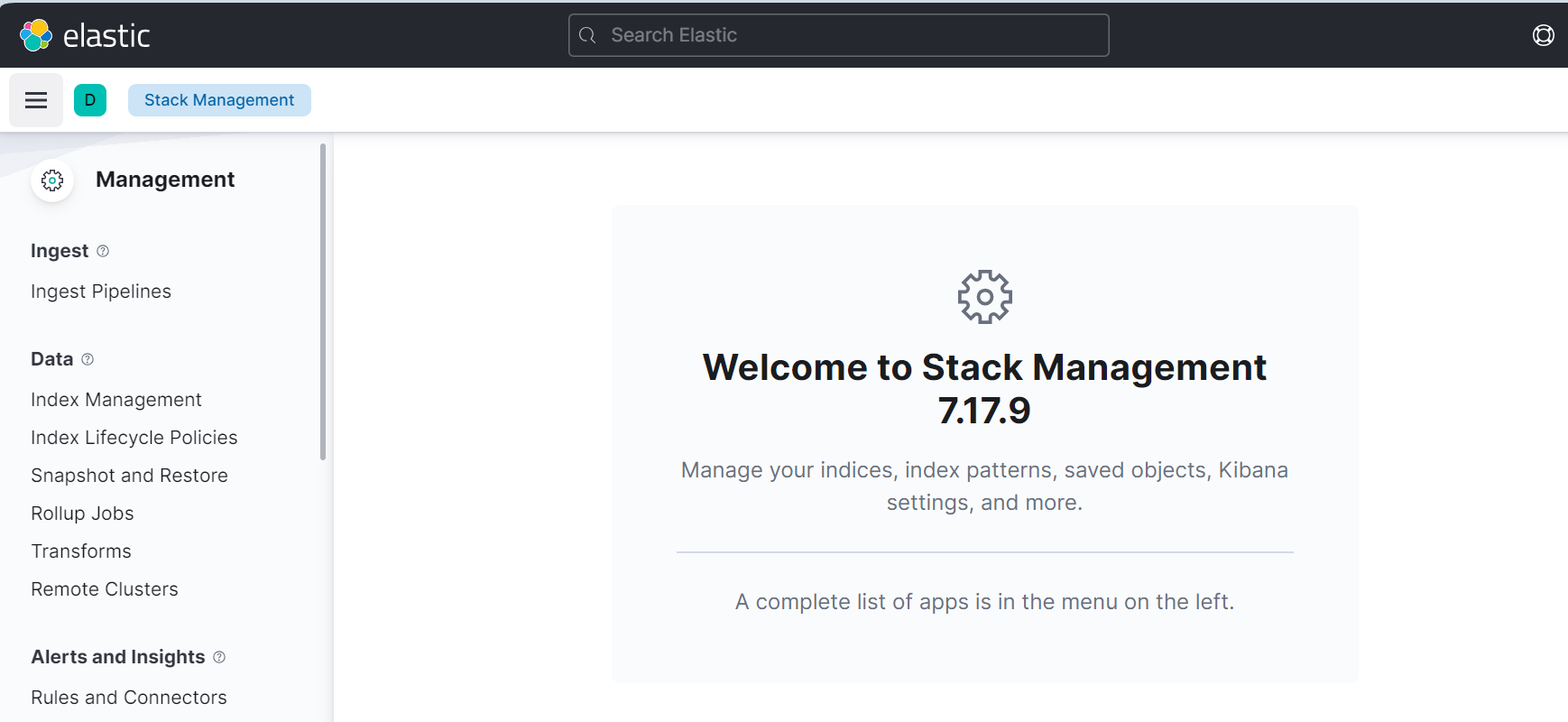
****

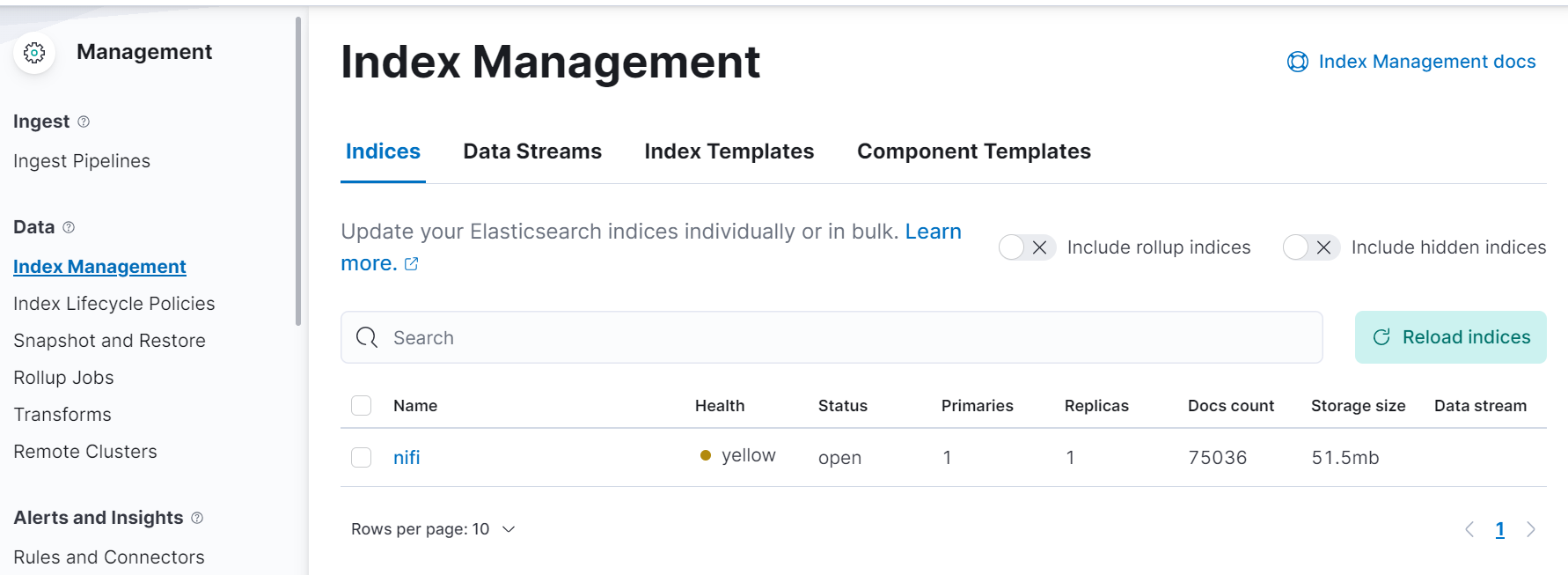
****

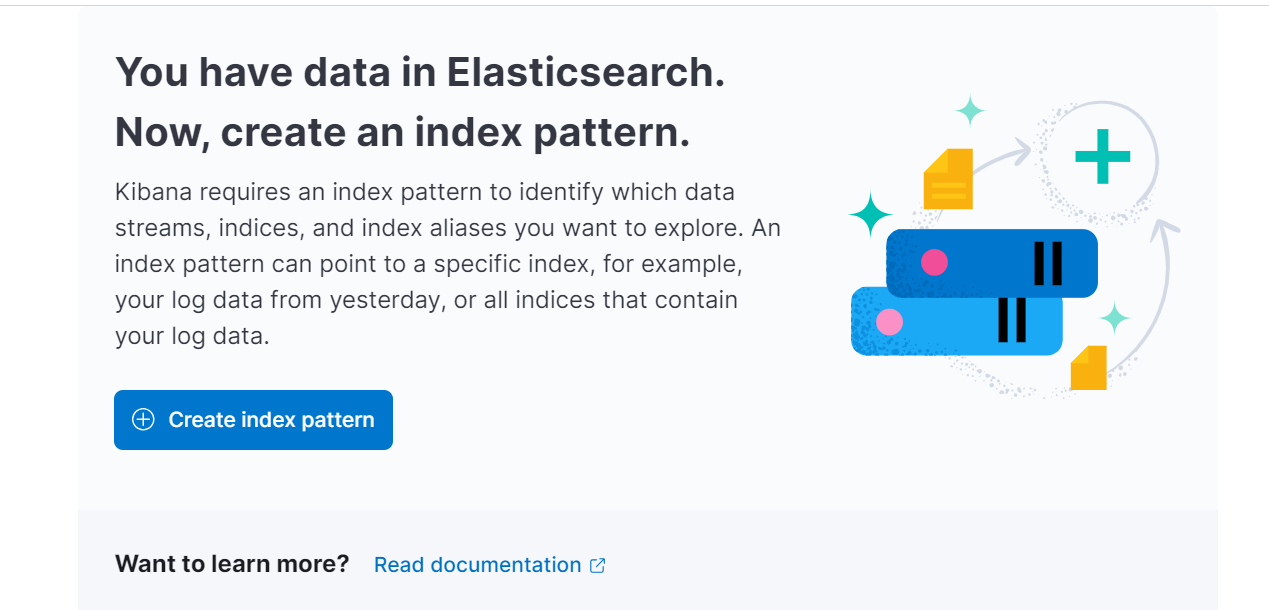
****

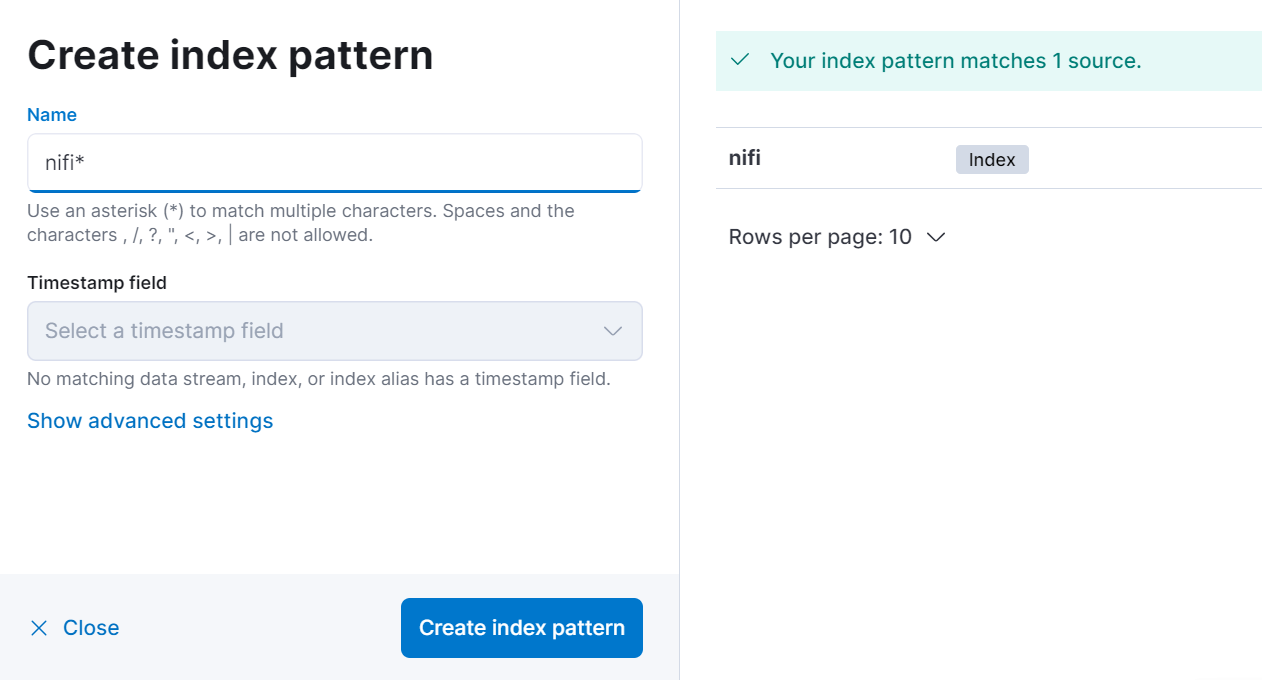
****

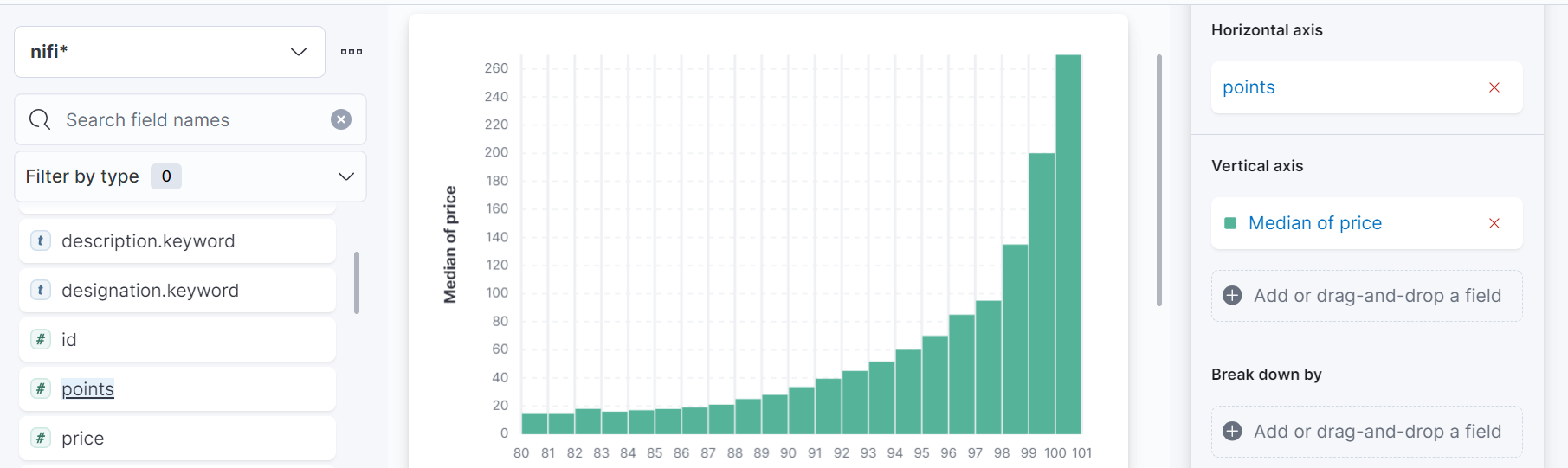
****

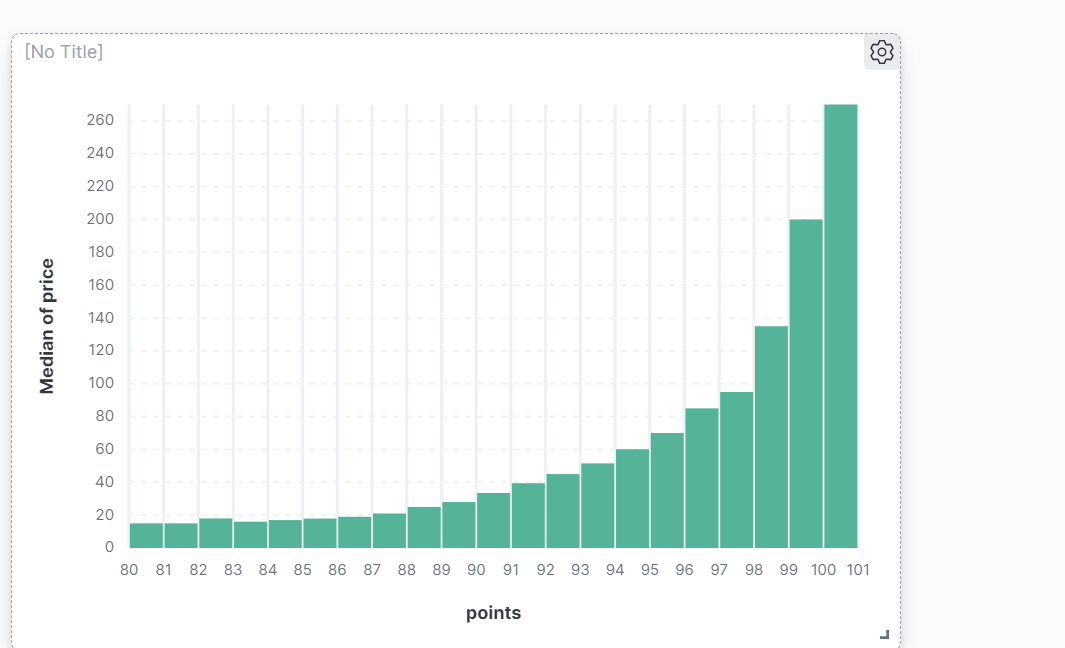
****

****

****

****

****

****