# Choix BDD

Mon choix s’est porté sur les données suivantes:

[https ://data.world/datafiniti/electronic-products-and-pricing-data](https://data.world/datafiniti/electronic-products-and-pricing-data)

Après pré-processing détaillé dans le rapport ci-joint, toutes les lignes de ce jeu de données ont pu être intégrées avec succès à l’aide d’un index. A l’aide d’ElasticSearch des requêtes ont pu être testées. La rapidité des réponses et leur pertinence ont contribuées au choix de cette BDD. De plus, la possibilité de faire une API à l’aide de fastApi pour requêter, insérer et peupler la BDD ont conforté ce choix. De plus, Elasticsearch fonctionne à l’aide de docker, le déploiement est donc possible avec docker-compose.

# FastApi

## Lancement

En se plaçant dans le répertoire « fastApi\_SNICOLE » ci-joint , il faut lancer ElasticSearch par la commande : « docker-compose up -d ». Il faut patienter un peu et lancer la commande pour s’assurer que tout fonctionne.   
« curl -X GET "localhost:9200" »

En restant dans le répertoire « fastApi\_SNICOLE » il faut ensuite lancer l’api par la commande :

« uvicorn app.main:api --reload »:

Une image contenant texte

Description générée automatiquement

Ouvrez l’Api grâce à ce lien : <http://localhost:8000/docs> :

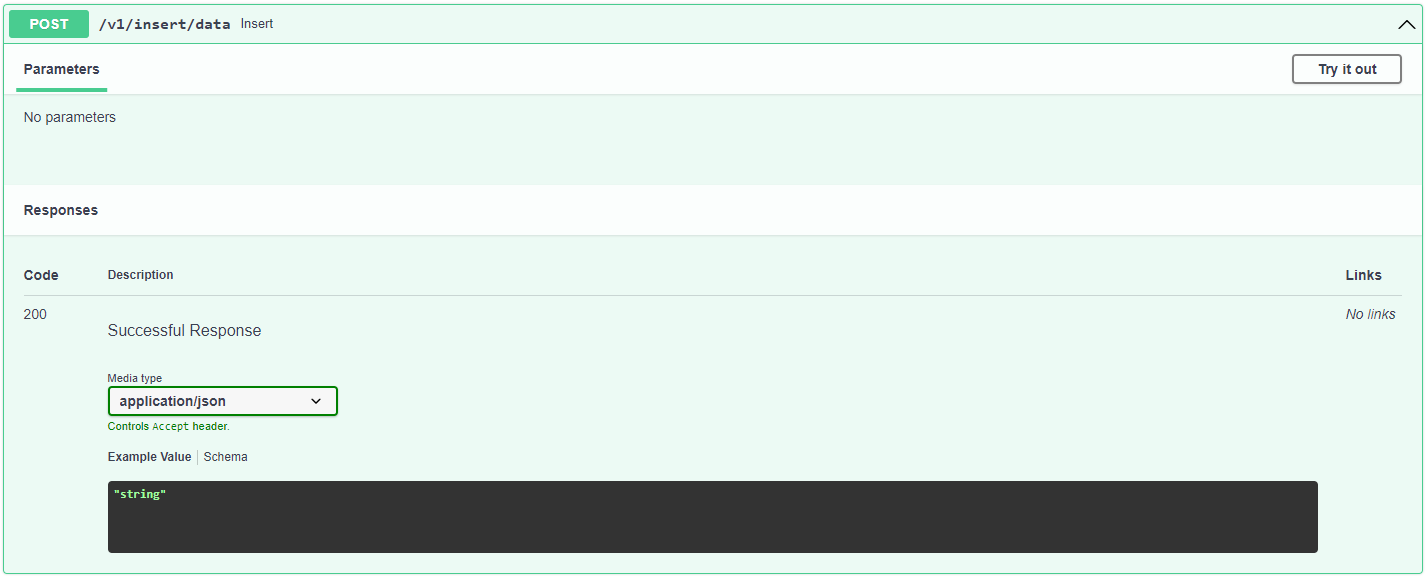
Une image contenant texte

Description générée automatiquement

## Présentation des routes

### “/v1/insert/data

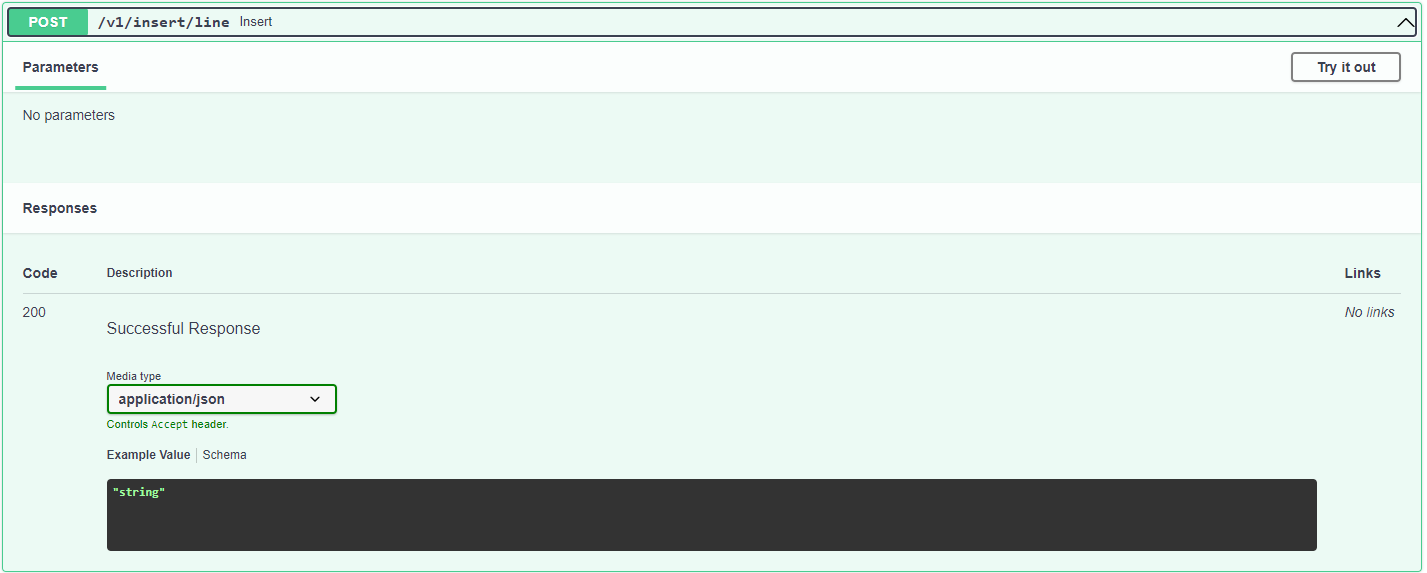
Cette route permet de peupler la base de données à l’aide du jeu de données.



### « /v1/insert/line

Cette route permet d’insérer une ligne où les champs sont séparés par une virgule (voir dossier ElasticSearch ci-joint pour plus de détail sur l’index à respecter) à l’aide du script modifiable « test.sh » , un exemple ci-dessous :





### « /v1/insert/brand »

Cette route permet de faire une recherche par marque. La marque « Boytone » est rentrée par défaut mais il suffit de modifier le fichier « main.py » dans le répertoire « app » ci-joint pour requête suivant la marque de votre choix :

Une image contenant texte

Description générée automatiquement

Une image contenant texte

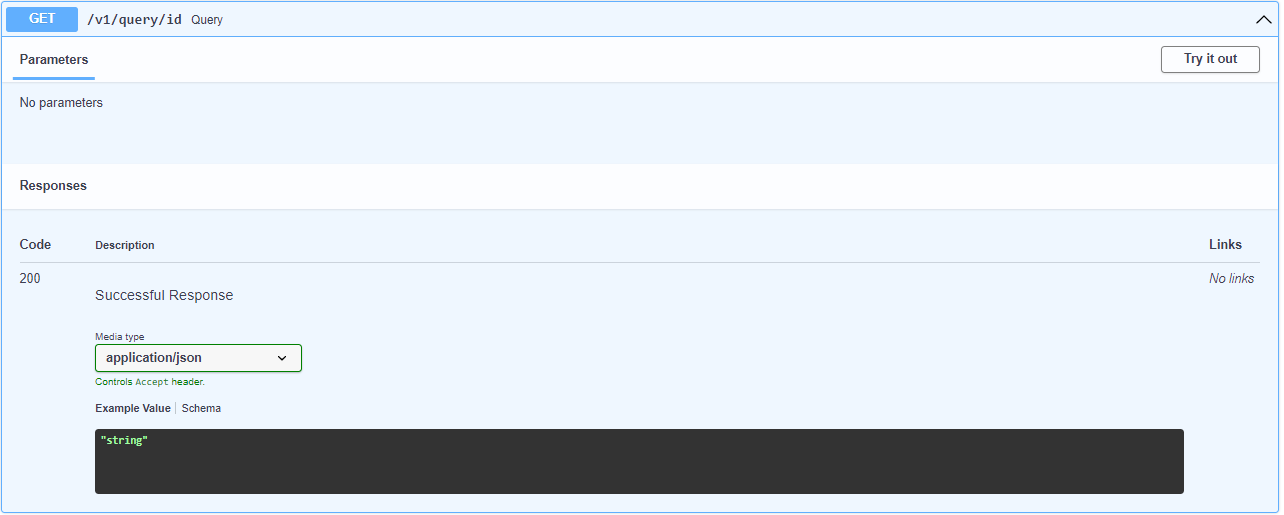
Description générée automatiquement

### « /v1/query/id »

Cette route permet de faire des rechercher par « id ». Un id est rentré par défaut mais peut être modifié dans le fichier « main.py ».

Une image contenant texte

Description générée automatiquement

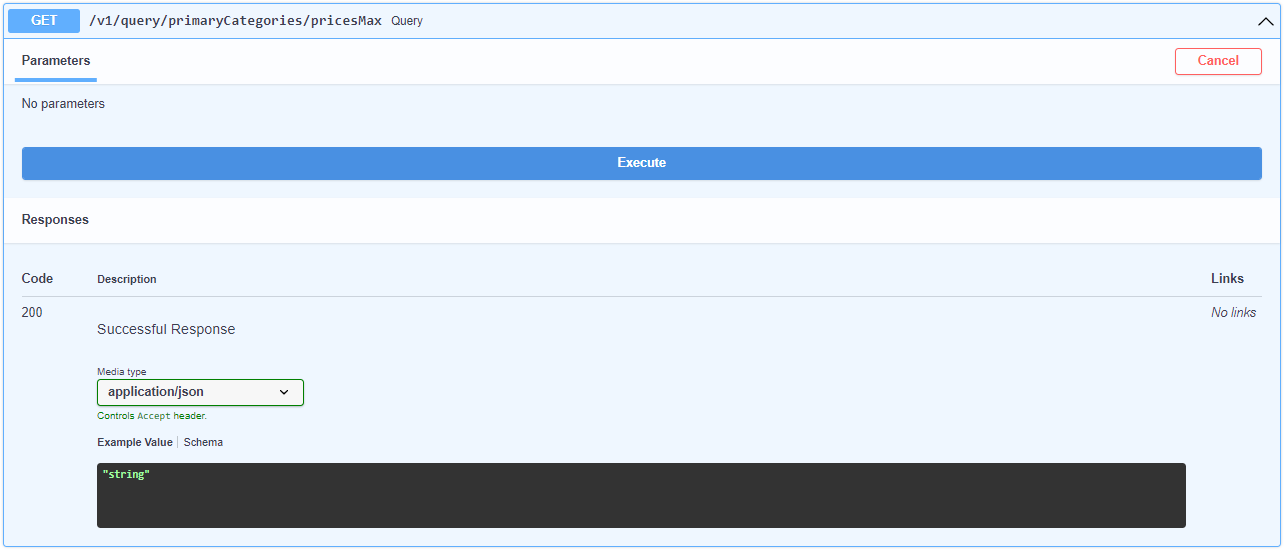


### « /v1/query/primaryCategories/pricesMax »

Cette route permet de faire des recherches par catégorie dans une fourchette de prix. Une recherche est rentrée par défaut, il est possible de la modifier dans le fichier main.py :

Une image contenant texte, capture d’écran, écran

Description générée automatiquement



### « /v1/query/brand/pricesMax »

Cette route permet de faire des recherches par marque dans une fourchette prix. Une recherche est rentrée par défaut, il est possible de la modifier dans le fichier main.py :

Une image contenant texte, capture d’écran, écran

Description générée automatiquement

