ТЕСТОВОЕ ЗАДАНИЕ BIBINET.RU

Необходимо создать 2 сервиса с отдачей данных в JSON формате, на Django и FastAPI.
Базу используем PostgreSQL.

***Django >= 3.12***

Сторонние библиотеки не используем, такие как Django REST framework. Делаем тем что идет из коробки с Django
Создаем модели базы данных:

- ***mark*** (id, name, producer_country_name, is_visible)
- ***model*** (id, name, mark_id, is_visible)
- ***part*** (id, name, mark_id, model_id, price, json_data, is_visible)

Заполнить таблицы mark model по 5 записей будет достаточно.
Написать скрипт для заполнения таблицы part через модель ORM, рандомно 10_000 записей.
Наименование запчасти можно заполнять из массива, 5 записей хватит. json_data словарь в коротом могут быть ключи (color, is_new_part, count) так же рандомно заполнять от всех ключей до пустого словаря.

Модели оптимизировать и добавить индексы для поиска, варианты запросов для оптимизации:

- Название запчасти, марка автомобиля, цвет автомобиля (color), показывать запчасть (is_visible)
- Страна производителя (producer_country_name), новая запчасть (is_new_part), показывать запчасть (is_visible)

Добавляем пути для вывода марок и моделей /mark/ и /model/

***Создаем поиск по запчастям***
***POST /search/part/***

Параметры поиска передаем через body json, все запросы строим через ORM
Примеры поиска по параметрам:

```
{
   "mark_name": "honda",
   "part_name": "бампер",
   "params": {
      "color": "черный"
   },
   "page": 1
}

{
   "mark_list": [1,3],
   "part_name": "бамп",
   "params": {
      "is_new_part": false,
     "color": "белый"
   },
   "price_gte": 2000,
   "price_lte": 5000,
```

```
    "page": 1
}
```


***Результат поиска***


```
{
    "response": [
        {
            "mark": {
                "id": 1,
                "name": "Honda",
                "producer_country_name": "Япония"
            },
            "model": {
                "id": 2,
                "name": "Accord"
            },
            "name": "Бампер передний",
            "json_data": {
                "is_new_part": true,
                "color": "красный",
                "count": 4
            },
            "price": 2300
        }
    ],
    "count": 1,
    "summ": 2300
}
```


Выводим по 10 записей в результатах (через page выводим следующие страницы)
response - список с результатом поиска (по 10)
count - количество всех найденных запчастей
summ – сумма всех найденных запчастей по параметру price

***FastAPI***

Сделать только поиск запчастей (такой же как на Django), но все запросы пишем на чистом SQL.

***Итог***
Написать команды для запуска в боевом режиме, для каждого из сервисов. Веб сервер используем Nginx. Настроить запуск так, чтобы максимально утилизировать процессор. Показать готовый вариант или описать как можно было бы обновлять проект без потери пользовательских соединений.
Все разворачивать можно по-разному, просто через Python системный, или же через Docker. Так же было бы не лишним проверить по производительности оба поиска запчастей через утилиту wrk или siege (по возможности запускать с другого ПК, в локальной сети) и сделать сравнение что быстрее, и по сколько запросов обрабатывается, через htop можно так же проверить какая нагрузка.

Создаем Git репозиторий, и делаем коммиты почаще для каждой новой логики, все можно в одном репозитории делать.

Результат на gorina@bibinet.ru. Вопросы также на gorina@bibinet.ru
Спасибо!