**Задача**

Необходимо создать сервис для вызова некоторого расчета (kernel.py) и хранения результатов. Результаты расчетов должны храниться в базе данных. Сервис должен предоставлять API, работающее поверх HTTP в формате JSON.

В расчет передаются параметры date\_start (формат "ГГГГ-ММ-ДД"), date\_fin (формат "ГГГГ-ММ-ДД"), lag (целое число).

Результатом расчета является DataFrame с полями 'date' (формат "ГГГГ-ММ-ДД"), 'liquid' (вещественное число), 'oil' (вещественное число), 'water' (вещественное число), 'wct' (вещественное число).

**Требования**

• Язык программирования python — очень желательно, так как это основной язык для нас;

• Предусмотреть очередь запусков, чтобы можно было запустить несколько расчетов друг за другом;

• Финальную версию сделать с возможностью запустить через Docker и передать через архив с инструкцией по установке;

• Простая инструкция для запуска;

• 3 метода: получение списка последних 10 запусков расчетов, получение одного расчета, создание расчета;

• Валидация полей в передаваемых параметрах

Если есть сомнения по деталям — решение принять самостоятельно, но в своём README.md рекомендуем выписать вопросы и принятые решения по ним. Разрешается менять модуль расчета kernel.py, без изменения структуры выходного результата (5 столбцов в таблице).

**Детали**

Метод получения списка последних 10 запусков расчетов

• Пагинация: на одной странице должно присутствовать 10 расчетов;

• Сортировки: по дате запуска (возрастание/убывание);

• Поля в ответе: название расчета, дата запуска, статус (завершен, идет расчет, в очереди).

Метод получения конкретного расчета

• Обязательные поля в ответе: все поля возвращаемого датафрейма, дата запуска, статус. Если расчёт еще не окончен, то ответ = None. Результат должен быть сериализован в JSON, для дальнейшей возможности передачи на фронт.

• Опциональные поля (можно запросить, передав параметр fields): название расчета, время затраченное на расчет.

Метод создания расчета:

• Принимает все вышеперечисленные поля: date\_start, date\_fin, lag;

• Возвращает ID созданного расчета.

**Усложнения**

Не обязательно, но задание может быть выполнено с любым числом усложнений:

• Юнит тесты: постарайтесь достичь покрытия в 70% и больше;

• Архитектура сервиса описана в виде текста и/или диаграмм

• Документация: есть структурированное описание методов сервиса.