Тестовое задание на позицию Backend Python

# Задача

Необходимо написать микросервис для асинхронного поиска в различных провайдерах. Сервис должен предоставлять HTTP API и принимать/отдавать запросы/ответы в формате JSON.

1. Реализовать сервис **provider-a** с одним методом POST /search, который возвращает данные из файла [response\_a.json](https://drive.google.com/open?id=1qLWsL570PxC5eTDkMYWBbFYtLTTBU1KE) c задержкой 30 секунд (для задержки нужно использовать sleep(30))
2. Реализовать сервис **provider-b** с одним методом POST /search, который возвращает данные из файла [response\_b.json](https://drive.google.com/open?id=1eGlS-_4gAqY1LkJU0rXgzGxZrq9yfMXZ) c задержкой 60 секунд (для задержки нужно использовать sleep(60)).
3. Реализовать сервис **airflow**, c методом поиска POST /search который отправляет запросы на поиск в сервисы **provider-a** и **provider-b** и в ответе возвращает уникальный search\_id поиска

| {  "search\_id": "d9e0cf5a-6bb8-4dae-8411-6caddcfd52da"  } |
| --- |

1. В сервисе **airflow** реализовать метод GET /results/{search\_id}/{currency}, который возвращает результаты поиска в провайдерах **provider-a** и **provider-b** по уникальному search\_id поиска с указанием валюты currency, например KZT. Результаты поиска должны накапливаться и быть отсортированными по цене, ответ также должен содержать статус поиска **PENDING**, **COMPLETED** в зависимости от стадии поиска. Пример ответа:

| {   "search\_id": "d9e0cf5a-6bb8-4dae-8411-6caddcfd52da",   "status": "PENDING",   "items": [...]  } |
| --- |
|  |

1. Результаты поиска нужно привести к единой валюте **currency,** нужно добавить в каждый результат поиска поле price, и пересчитать по текущему курсу сумму результата. Курс валют должен скачиваться раз в день в 12:00, также нужно скачать курс при первом запуске. Чтобы не нагружать основной метод поиска.

{   
 "price": {

"amount": "10000.00",   
 "currency": "KZT"

}

"pricing":{...}

...

}

Курсы валют можно получить с помощью запроса

GET https://www.nationalbank.kz/rss/get\_rates.cfm?fdate=26.10.2021.

# Требования

1. Сервисы должен быть написан на языке Python с использованием любого из веб фреймворков
2. В качестве хранилища данных можно использовать любую технологию.
3. Сервер должен быть доступен на порту 9000
4. Предоставить инструкцию по запуску приложения. В идеале (но не обязательно) – использовать контейнеризацию с возможностью запустить проект командой docker-compose up

# Будет плюсом

1. Использование асинхронного фреймворка и кода
2. Написать тесты (постарайтесь достичь покрытия в 70% и больше)
3. Если вдруг будет желание, можно сделать простой UI
4. Вместо файлов [response\_a.json](https://drive.google.com/open?id=1qLWsL570PxC5eTDkMYWBbFYtLTTBU1KE) и [response\_b.json](https://drive.google.com/open?id=1eGlS-_4gAqY1LkJU0rXgzGxZrq9yfMXZ), распарсить реальные ответы провайдеров из файлов [response\_a.xml](https://drive.google.com/open?id=1orQYMqY70iAZbAKKa9YnKQAOgofzYKEm) и [response\_b.xml](https://drive.google.com/open?id=1AuvU-QlV2Nv47h2D_tP3BB7Qzp9UaY0Z), результат должен быть идентичным по формату.