**Тестовое задание для backend-программиста на Python**

Уважаемый соискатель, перед Вами тест на вакансию **Python-developer** в компании Saber Interactive.

После выполнения задания в тексте электронного письма укажите, пожалуйста:

*Фамилию, имя, отчество*

*Дату выполнения*

*Примерное количество времени, затраченного на выполнение теста*

**Сроки:** мы не ставим жёстких ограничений по срокам: скорость и качество выполнения задания так же продемонстрируют вашу мотивацию и заинтересованность.

# **Задача “Log Streamer”**

Имеется log-файл в кодировке UTF-8 и формате JSONL:

{"level": "DEBUG", "message": "Blah blah blah"}

{"level": "INFO", "message": "Everything is fine!"}

{"level": "WARN", "message": "Hmmm, wait..."}

{"level": "ERROR", "message": "Holly $@#t!"}

…

Каждая строка лога — это JSON-объектам с двумя полями:

* level — уровень логирования (может принимать следующие значения: "DEBUG", "INFO", "WARN" и "ERROR");
* message – произвольный текст с сообщением.

Нужно написать web-сервис, который позволяет последовательно по частям вычитать данный лог.

Сервис должен корректно работать с приемлемым откликом для любого размера файла логов (от гигабайта и выше).

На все запросы сервер должен возвращать ответ с кодом 200 и телом в виде JSON-объекта.

Тело ответа всегда должно содержать булево поле ok, сигнализирующее об успешном завершении операции, и поле reason с сообщением о причине ошибки – в случае неудачного выполнения операции, например:

200 {"ok": true}

200 {"ok": false, "reason": "file was not found"}

## Запросы в backend

### POST /read\_log

Чтение лога.

Формат запроса:

{"offset": <number>}

* offset – позиция, с которой должно быть начато чтение очередной порции лога.

Пример ответа:

{

"ok": true,

"next\_offset": <number>,

"total\_size": <number>,

"messages": [

{"level": "INFO", "message": "Everything is fine!"}

{"level": "WARN", "message": "Hmmm, wait... It looks like..."}

{"level": "ERROR", "message": "Holly $@#t!"}

]

}

* next\_offset – позиция, с которой должно продолжиться чтение лога.
* total\_size – размер всего лога.
* messages – список очередной порции сообщений из лога. В свою очередь, сообщения являются JSON-объектами, имеющие ту же структуру, что и строки лога.

В чём измеряются поля offset, next\_offset и total\_size предлагается решить самостоятельно.

## Взаимодействия с клиентом

При первом обращении к backend'y клиент задаёт значение поля offset равным 0. При последующих обращениях клиент устанавливает значение offset равным значению next\_offset, которое берётся из тела последнего успешного ответа.

Как только значение поля next\_offset становится равным значению поля total\_size, клиент перестаёт посылать запросы к backend'у, и считается, что все сообщения из лога получены.

## Реализация

Следует реализовать сервер для вышеописанного интерфейса.

Для решения следует использовать *Python* версии 2.7. В качестве фреймворка можно использовать любой из нижеперечисленного:

* *Twisted/Cyclone*;
* *Tornado*;
* *Django framework*.