

Подготовка окружения:

1. Выполним инструкции по сборке проекта и созданию докер образов, согласно README.MD.

2. Запустим окружение в режиме standalone - выполним команду из пункта «Запуск в режиме standalone (Только база данных, все остальное не работает)». В результате будет запущено окружение в следующей конфигурации: балансировщик nginx принимает запросы и проксирует их на один из серверов приложения, оба сервера приложения смотрят на инстанс HA-Proxy, который принимает запросы к БД и балансирует их между инстансами Postgres.

3. Используя коллекцию Postman, расположенную в корне проекта - (oleg\_galimov\_social\_network.postman\_collection.json), проверим что балансировка с помощью Nginx работает (балансировка настроена на порт 80, используется Sticky session через ip-hash, так как токен после авторизации живет только на конкретном инстансе приложения):

The screenshot shows the Postman interface for a POST request to `localhost/user/register`. The 'Query Params' tab is active, showing an empty table with columns 'Key' and 'Value'. The 'Body' tab is also visible, showing a JSON object with `"user_id": "692d0e50-cf44-45c1-8167-a693bffb7196"`.

The screenshot shows the REST Client interface with a POST request to `localhost/login`. The request body is in JSON format, containing `"id": "692d0e50-cf44-45c1-8167-a693bffb7196"` and `"password": "password123"`. The response is also in JSON format, showing a successful status of 200 OK and a `"token"` value. The interface includes tabs for Params, Authorization, Headers, Body, Pre-request Script, Tests, and Settings. The Body tab is active, and the response is displayed in the lower section.

### 3. с полученным авторизационным токеном поищем пользователей:

The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** localhost/user/search?first\_name=Антон&last\_name=Абрамов
- Params:** first\_name (checked), last\_name (checked)
- Response Body (JSON):**

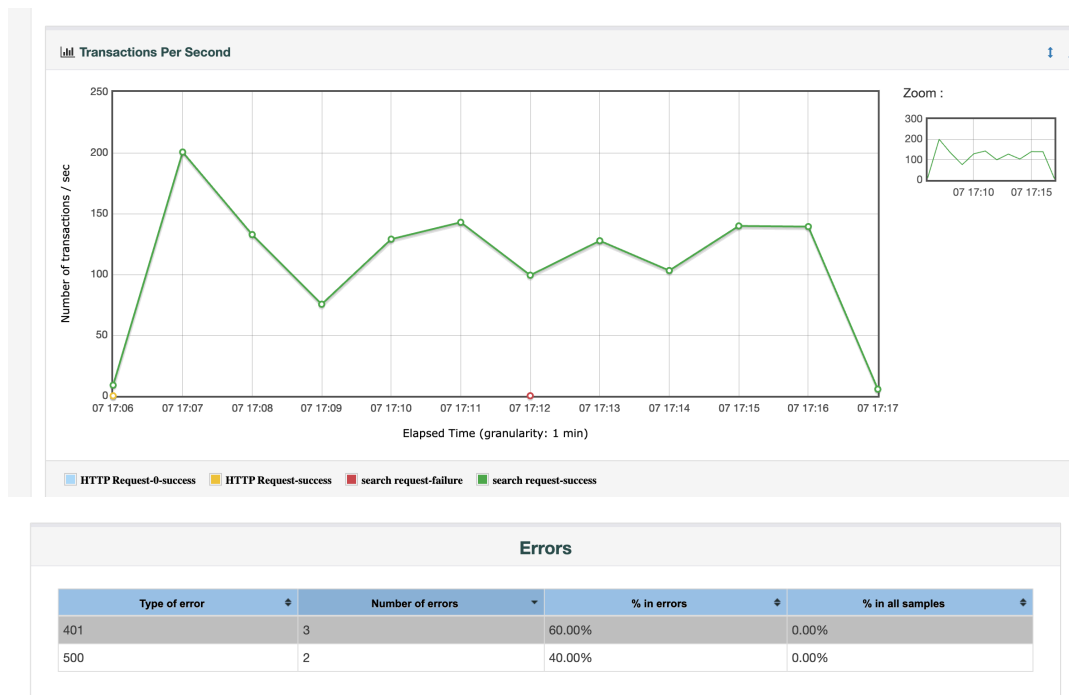
```
[{"id": "33f27b9c-55cf-4354-9909-dfb5f5e2276f", "first_name": "Антон", "second_name": "Абрамов", "age": 50, "sex": "undefined", "interests": "", "city": "Ижевск"}, {"id": "07f62ab9-0386-4b32-95ec-65ebeb5b6ec1", "first_name": "Антон", "second_name": "Абрамов", "age": 50, "sex": "undefined", "interests": "", "city": "Ижевск"}]
```

Видно, что старый функционал работает, причем уже через балансировку, так как обращение идет к localhost по порту Nginx - 80.

#### Нагрузочное тестирование, сценарий:

1. Создание нового пользователя: в один поток отправляется запрос на сервер для регистрации нового пользователя.
2. Авторизация, получение токена: обращение к серверу по полученному в п. 1 идентификатору.
  - Поиск случайного пользователя из заданного набора данных в течении 10 минут: 1000 пользователей параллельно (по 10 на каждый запрос).
  - Старт с 0 до 1000 в течении минуты. Верификация, что каждый ответ получил 200 Ок.
  - Поиск по имени и фамилии на эндпоинте **/user/search**
  - Все ответы приходят с кодом 200 Ок.
  - Спустя 2 минуты после старта останавливаем один инстанс приложения, затем спустя еще 3 минуты - останавливаем один инстанс БД.
  - Ожидаемый результат - стабильная работа в течении оставшихся 5 минут теста с незначительным фоном ошибок.

#### Результат:



Из графиков видно, что производительность снизилась на второй минуте (с 200 grs до ~140 в момент отключения одной ноды приложения. При этом, система оставалась стабильной судя по логам Nginx и отвечала на запросы:

```
2023-08-07 17:12:39 192.168.32.1 - - [07/Aug/2023:14:12:39 +0000] "GET /user/search?first_name=%D0%92%D0%B0%D0%BB%D0%B5%D1%80%D0%B8%D1%8F&last_name=%D0%A4%D0%BE%D0%BA%D0%B8%D0%BD%D0%B0 HTTP/1.1" 200 12 "-" "Apache-HttpClient/4.5.13 (Java/19.0.2)" "-"
```

Далее на 7-й минуте теста отключен инста постгрес. На графиках этот момент заметен - в ответ на запрос получено несколько ошибок с кодом 500, что вполне ожидаемое, так как сервис использует пул коннектов, которые были ранее открыты к инстансу, который неактивен.

Исходя из этого можно сделать вывод: балансировка с помощью NGINX и HAProxy настроена корректно, система остается стабильной при отказе разных узлов.