Установка системы без SSL (http)

1. Для разворачивания контейнеров создаем директории для баз данных и обратного прокси

```
mkdir resources
```

2. Переходим в директорию баз данных и создаем директорию для postgres

```
cd resources/
mkdir postgres
```

3. Переходим в директорию postgres, копируем в нее файл docker-compose.yml из предоставленного дистрибутива.

Создаем сеть для контейнеров базами данных и запускаем контейнер с postgres

```
cd postgres/
docker network create resources
docker-compose up -d
```

После запуска, скачается необходимый образ и запустится контейнер postgres.
Логин по умолчанию: postgres
Пароль: postgres
*изменить можно в файле **docker-compose.yml** перед запуском контейнера*

4. Создаем базу данных postgres

С именем fastboard_back. Создать можно разными способами, например подключиться к контейнеру при помощи PgAdmin или psql-16

```
sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt $(lsb_release -cs)-pgdg main" >
/etc/apt/sources.list.d/pgdg.list'
    curl -fsSL https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo gpg --dearmor -o
/etc/apt/trusted.gpg.d/postgresql.gpg
    apt update
    apt install postgresql-client-16
    psql -h 127.0.0.1 -U postgres
            \l
            CREATE DATABASE fastboard_back;
            \l
            \q
```

5. Переходим на директорию выше, т.е. в директорию resources

```
cd ..
```

Создаем директории для Clickhouse переходим в нее

```
mkdir clickhouse
cd  clickhouse/
```

6. Копируем в директорию clickhouse файл docker-compose.yml и директории с конфигами (etc, custom_config) из предоставленного дистрибутива Clickhouse.

Меняем параметры выделенной памяти для контейнера в файле **docker-compose.yml** (по умолчанию выставлено от 2 до 8 Гб ОЗУ) и запускаем контейнер с clickhouse

```
docker-compose up -d
```

После запуска, скачается необходимый образ и запустится контейнер с clickhouse.
Логин: admin
Пароль: Passw0rd
*изменить можно в файле **etc/users.xml** перед запуском контейнера*

7. Переходим на директорию выше, т.е. в директорию resources

```
cd ..
```

8. Создаем директории для Redis и переходим в нее

```
mkdir redis
cd  redis/
```

9. Создаем директории для баз и логов redis

```
mkdir -p data/{bases,log}
```

10. Копируем в директорию redis файл docker-compose.yml и директорию с конфигами etc из предоставленного дистрибутива Redis. Запускаем контейнер с Redis

```
docker-compose up -d
```

После запуска, скачается необходимый образ и запустится контейнер с redis.
Пароль по умолчанию: Passw0rd,
*изменить можно в файле **etc/redis.conf** перед запуском контейнера*

11. Создаем директории для RabbitMQ и переходим в нее

Для доступа к веб-интерфейсу Traefik нужно сгенерировать и подставить пароль в файл docker-compose.yml, для этого выполняем следующие шаги:

```
mkdir rabbitmq
cd rabbitmq/
```

Копируем в директорию RabbitMQ файл **docker-compose.yml** из предоставленного дистрибутива RabbitMQ. Запускаем контейнер с RabbitMQ

```
docker-compose up -d
```

После запуска, скачается необходимый образ и запустится контейнер с RabbitMQ.
Логин и пароль по умолчанию: fb_rabbit,
*изменить можно в файле **docker-compose.yml** перед запуском контейнера*
Теперь нужно создать VHOST для контейнера бэк:

```
docker exec -it rabbitmq_1 bash
    rabbitmqctl add_vhost fb1
    rabbitmqctl list_vhosts
    rabbitmqctl set_permissions -p "fb1" "fb_rabbit" ".*" ".*" ".*"
```

12. Создаем директорию для FastBoard,

Переходим в нее, копируем **docker-compose.yml**, **license_rsa.pub** и образы docker из архива **fastboard.tar.gz** в рабочую директорию проекта

```
cd ..
mkdir fastboard
cd fastboard
```

Проверяем и устанавливаем данные для авторизации в базах данных и параметры подключения к контейнерам с базами в файле **docker-compose.yml**, в данный момент там установлены данные для подключения по умолчанию.

Далее импортируем образы контейнеров **fastboard:back** и **fastboard:front**, для этого выполняем команды:

```
docker load -i fastboard-back.tar
docker load -i fastboard-front.tar
```

После загрузки образов, запускаем контейнеры бэк и фронт, а также проверяем логи из запуска

```
docker-compose up -d
docker-compose logs -f
```

или

```
docker-compose up -d
docker-compose ps
docker logs -f container_name
```

В момент запуска конейнера с бэкенд приложения, пройдут миграции и создастся база данных с логином и паролем по умолчанию. Логин admin пароль YYYY-MM-DD (дата первого запуска контейнеров)
Если контейнеры запустились без проблем, то проверяем работу пройдя по ссылке указанной в **docker-compose.yml** для контейнера фронт: https://example.com

13. Превый вход в систему и пользователи

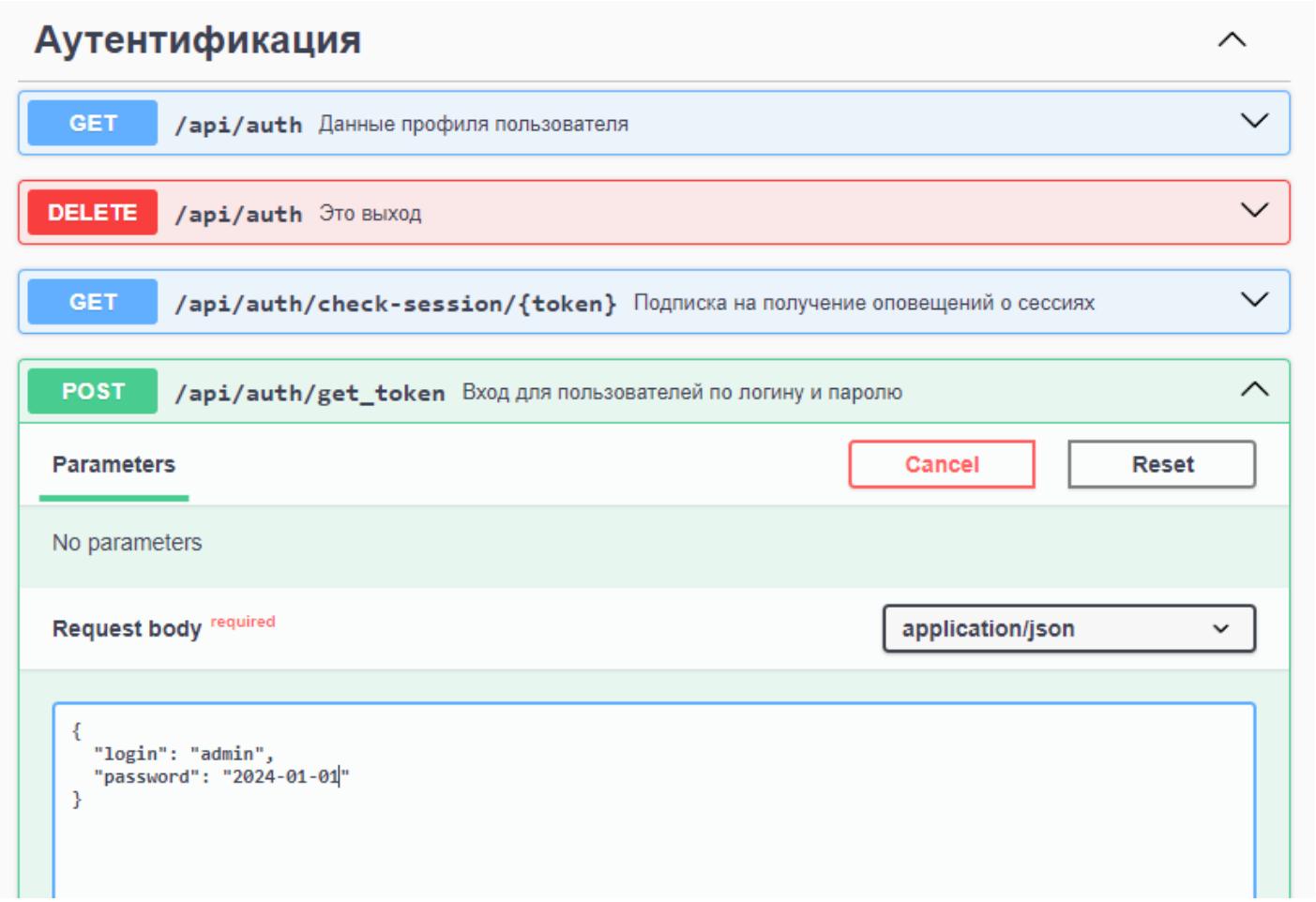
После успешного развертывания нужно войти в API системы под техническим пользователем для активации первого лицензионного ключа. Этот пользователь имеет права администратора и остается в системе.

Входим в API

Переходим по адресу: **ваш_бэкенд/docs/swagger,** находим блок аутентификации и метод **get_token.**
Жмем кнопку **TRY OUT**, вводим учетные данные:

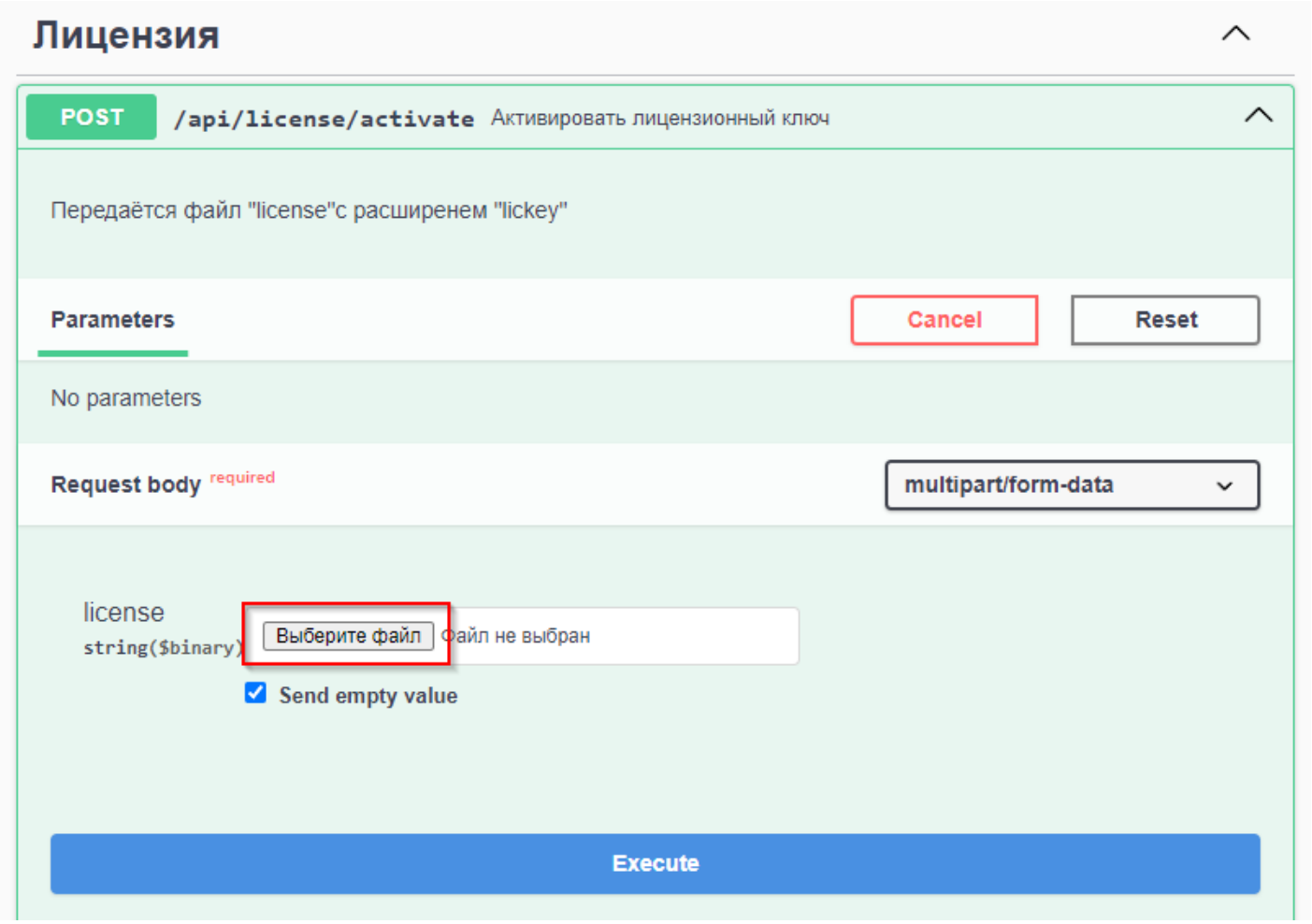
Логин: admin
Пароль: дата первого запуска контейнеров ГГГГ-ММ-ДД

Далее нажимаем кнопку **EXECUTE**



Далее активируем лицензионный ключ

В блоке «Лицензия» находим метод Acivate, жмем кнопку **TRY OUT**, выбираем файл лицензии и нажимаем кнопку **EXECUTE**



Система готова к работе

Теперь можно войти спомощью интерфейса и создать пользователей через панель администратора и выдавать им лицензии

Revision #9
Created 10 September 2024 08:59:44 by Станислав
Updated 29 May 2025 11:18:54 by Марина