

**ИНСТРУКЦИЯ**  
**по установке Программы для ЭВМ «Система мониторинга**  
**эффективности «Градиент»**

**Листов: 21**

**Правообладатель: Общество с ограниченной ответственностью «ИТ1»**

Москва, 2025

## Оглавление

1. Общие требования.....	3
2. Подготовка среды.....	4
2.1 Создание ssh-ключа.....	4
2.2 Создание директорий и распаковка установочного комплекта.....	4
2.3 Загрузка Docker-образов.....	4
3. Настройка компонентов .....	5
3.1 Настройка Nginx.....	5
3.2 Настройка S3 Storage (MinIO).....	5
3.3 Gradient + InfluxDB + Keycloak + PostgreSQL.....	6
3.3.1 Развертывание базового набора .....	6
3.3.2 Настройка .env .....	7
3.3.3 Инициализация InfluxDB .....	7
3.3.4 Настройка Keycloak.....	8
3.3.5 Заполнение общих параметров .....	10
3.3.6 Запуск Gradient.....	11
3.4 Apache Airflow .....	11
4. Добавление пользователя в MinIO через CLI.....	14
5. Проверка работоспособности .....	15
6. Завершение установки .....	16
6.1 Apache Airflow – подключение тестовых данных.....	16
7. Контакты .....	21

1. ОБЩИЕ ТРЕБОВАНИЯ

Для работы Программы для ЭВМ «Система мониторинга эффективности «Градиент» (далее – Система) необходимы компоненты, указанные в таблице 1.

Таблица 1

№ п/п	Компонент	Версия
1.	Apache Airflow	2.11
2.	S3 Storage (MinIO)	RELEASE.2025-09-07T16-13-09Z
3.	InfluxDB 3 Core	3.0
4.	Keycloak	24.0.1
5.	PostgreSQL	16
6.	Gradient	—

Все компоненты можно развернуть из установочного комплекта или использовать существующие инсталляции (S3, InfluxDB, PostgreSQL, Keycloak). Однако Apache Airflow должен использовать только DAG из комплекта поставки Системы.

Требования:

- Linux сервер с Docker и Docker Compose.
- Доступ к интернету для загрузки образов.
- Для Airflow необходимо наличие доступов к Jira, Bitbucket и другим корпоративным системам (настраиваются через connection.json).

Схема развертывания Системы представлена на рисунке 1.

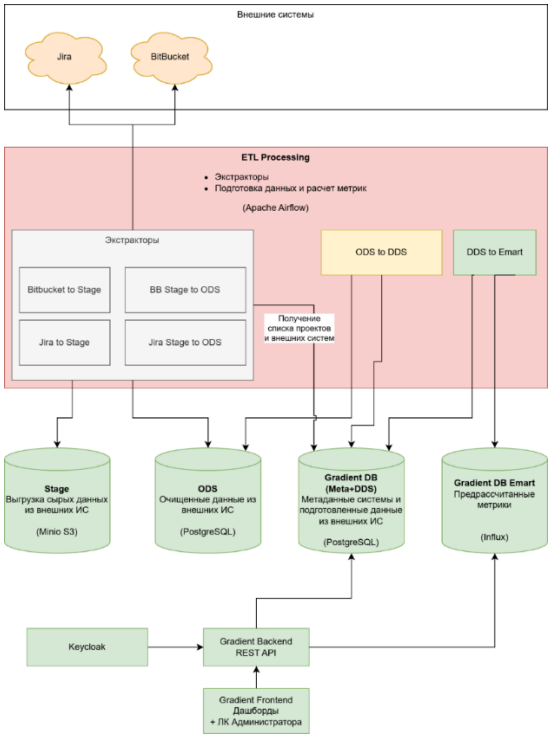


Рисунок 1. Схема развертывания Программы для ЭВМ «Система мониторинга эффективности «Градиент»

## 2. ПОДГОТОВКА СРЕДЫ

### 2.1 Создание ssh-ключа

Перед началом развертывания ПО создайте ssh-ключ:

1. Создайте файл config с помощью powershell используя команду: notepad "\$env:USERPROFILE\.ssh\config"
2. Внесите необходимые для подключения данные в файл config:

```
Host gradient-poco
  user deploy
  port 43522
  hostname 123.456.789.1
  identityfile ~/.ssh/id_ed25519
```

3. Проверьте подключение к удаленной машине с помощью powershell используя команду:

```
ssh gradient-poco
```

В случае возникновения трудностей по созданию ssh-ключа, просьба обратиться к техническому специалисту, контакты которого указаны в разделе 7 настоящей инструкции.

### 2.2 Создание директорий и распаковка установочного комплекта

**Создаем папки для установки:**

```
mkdir -p /opt/services/gradient-ropo
mkdir -p /opt/services/gradient-ropo/images
mkdir -p /opt/services/gradient-ropo/dags
```

**Распаковываем архивы:**

```
tar -xzf /tmp/resources/resources/gradient_dist.tgz -C
/opt/services/gradient-ropo/
tar -xzf /tmp/resources/resources/gradient_images.tgz -C
/opt/services/gradient-ropo/images/
tar -xzf /tmp/resources/resources/airflow-dags-*.tgz -C
/opt/services/gradient-ropo/dags/
```

### 2.3 Загрузка Docker-образов

**Загружаем все образы в Docker:**

```
ls -l /opt/services/gradient-ropo/images/*.img | xargs --no-run-if-empty -L 1
docker load -i
```

### 3. НАСТРОЙКА КОМПОНЕНТОВ

#### 3.1 Настройка Nginx

4. Создаем директорию, необходимую для копирования:

```
mkdir -p /opt/services/gradient-ropo/gradient
```

5. Копируем настройки Nginx в созданную директорию:

```
cp -r /opt/services/gradient-ropo/nginx /opt/services/gradient-ropo/gradient/
```

6. Переходим в директорию nginx:

```
cd /opt/services/gradient-ropo/nginx
```

7. Запускаем Nginx:

```
docker compose up -d
```

📁 Используемый файл: **docker-compose.yaml** (из каталога /opt/services/gradient-ropo/nginx).

При переходе на ip сервера <http://123.456.789.1> мы увидим успешно запущенный nginx

localhost:8080

## Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to [nginx.org](http://nginx.org).  
Commercial support is available at [nginx.com](http://nginx.com).

*Thank you for using nginx.*

Рисунок 2. Окно успешного запуска Nginx Reverse Proxy

#### 3.2 Настройка S3 Storage (MinIO)

1. Копируем настройки MinIO:

```
cp -r /opt/services/gradient-ropo/minio /opt/services/gradient-ropo/gradient/
```

2. Переходим в директорию minio:

```
cd /opt/gradient/minio
```

3. Редактируем файл .env (команда: nano .env), указывая следующие данные:

```
MINIO_ROOT_USER=admin_user
```

```
MINIO_ROOT_PASSWORD=Crjhhtt,s25
```

```
MINIO_BROWSER_REDIRECT_URL=http://123.456.789.1/minio/ui/
```

Сохраняем изменения в файле .env: Ctrl+X, затем Y, затем Enter

4. Запускаем:

```
docker compose up -d
```

📁 Используемые файлы: .env, docker-compose.yml

📷 **Скриншот:** веб-интерфейс MinIO по адресу <http://123.456.789.1/minio/ui/>.

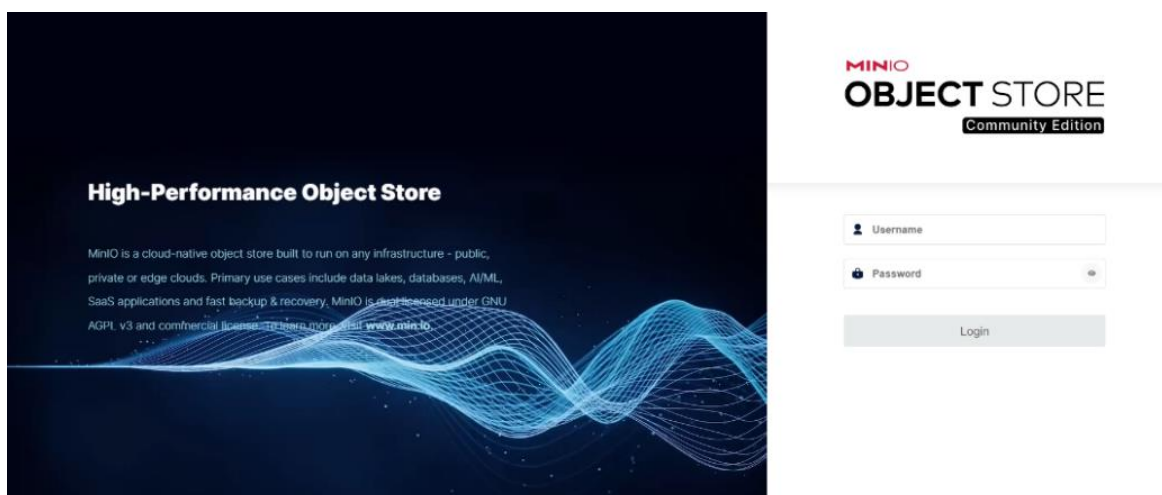


Рисунок 3. Веб-интерфейс MinIO

### 5. Создаем bucket gradient

Запускаем <https://123.456.789.1/minio/ui/>

Вводим ранее указанные данные для входа:

login: admin\_user

password: Crjhhtt\_,s25

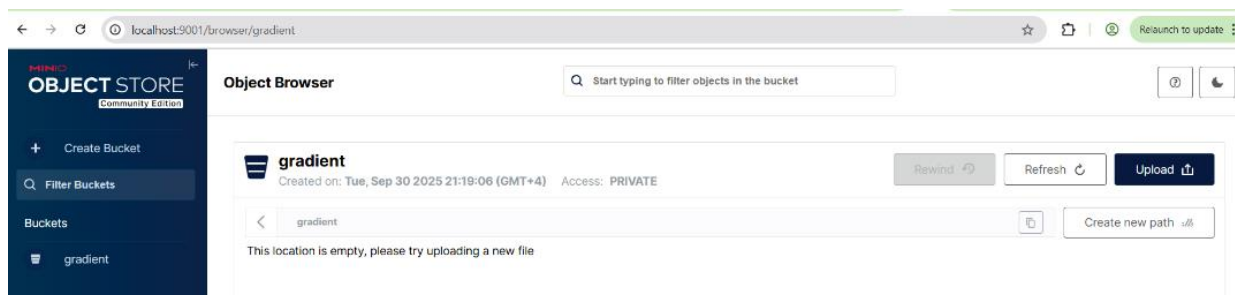


Рисунок 4. Создание bucket gradient

### 6. Создаем пользователя для bucket gradient

Процесс создания пользователя изложен в разделе 4 «Добавление пользователя в MinIO через CLI».

## 3.3 Gradient + InfluxDB + Keycloak + PostgreSQL

Эти компоненты разворачиваются совместно.

Отдельный PostgreSQL используется для Keycloak.

### 3.3.1 Развертывание базового набора

```
cp -r /opt/services/gradient-ropo/gradient /opt/gradient/
cd /opt/services/gradient-ropo/gradient
./init_catalogs.sh
```

Создать схемы в базе данных PostgreSQL для работы с Airflow:

```
docker exec gradient-db-1 psql -U gradient -d gradient-db -c 'CREATE SCHEMA
IF NOT EXISTS dds;'
```

```
docker exec gradient-db-1 psql -U gradient -d gradient-db -c 'CREATE SCHEMA IF NOT EXISTS ods;'
```

Альтернативный вариант (если контейнер запущен через docker compose из каталога gradient):

```
bash
docker compose exec db psql -U gradient -d gradient-db -c 'CREATE SCHEMA IF NOT EXISTS dds; '
docker compose exec db psql -U gradient -d gradient-db -c 'CREATE SCHEMA IF NOT EXISTS ods;'
```

Проверить создание схем:

```
docker exec gradient-db-1 psql -U gradient -d gradient-db -c '\dn'
```

Должны быть созданы схемы: `dds`, `ods`, `public`

### 3.3.2 Настройка .env

Редактируем файл .env (команда: nano .env):

```
KEYCLOAK_ADMIN_USERNAME=gradient-admin
KEYCLOAK_ADMIN_PASSWORD=5#EYl6ZyI4Qh

POSTGRES_DB= gradient-db
POSTGRES_USER=gradient
POSTGRES_PASSWORD=qiipN6EIfS
```

Сохраняем изменения в файле .env: Ctrl+X, затем Y, затем Enter

### 3.3.3 Инициализация InfluxDB

Запустить скрипт init\_influx.sh, в результате в файле env\_back переменная INFLUX\_TOKEN будет заполнена значением административного токена для Influx

Просмотр файла: nano env\_back

```
POSTGRES_DB=gradient-db
POSTGRES_USER=gradient
POSTGRES_PASSWORD=<указать данные>
POSTGRES_HOST=db
POSTGRES_PORT=5432

SECRET_STORE_ENCRYPTION_KEY="8RtKaTl5lFhA"

INFLUX_URL=http://influxdb3-core:8181
INFLUX_DB=gradient_metric
INFLUX_TOKEN=apiv3_3h4BfK1ZO_bGssGiLUL2mfHypZ9bKrB3h3V1HqkxEA-
r6W4IkaXijD0vXZwah3z3pNxq59AbBAZpV4dZK5k4MQ

VERIFY_SSL=False
KEYCLOAK_URL=http://gradient-demo.it-one.ru/keycloak
KEYCLOAK_INTERNAL_URL=http://keycloak:8080/keycloak
KEYCLOAK_EXTERNAL_URL=http://gradient-demo.it-one.ru/keycloak
```

```
KEYCLOAK_REALM=Gradient
KEYCLOAK_CLIENT_ID=gradient_client
KEYCLOAK_CLIENT_SECRET=<указать данные>

BASE_URL=http://gradient-demo.it-one.ru
```

### 3.3.4 Настройка Keycloak

1. Запустить сервер:

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

2. Перейти по адресу <https://gradient-demo.it-one.ru/keycloak> (VPN должен быть отключен)

Вход осуществляется с использованием данных, указанных в файле .env.

**NB** Если пользователь не создался, создать вручную

3. Выполнить первичную настройку:

- Создаем **Realm** для приложения Градиент, указывая в поле «Realm name» значение «gradient-realm» (рисунок 5).

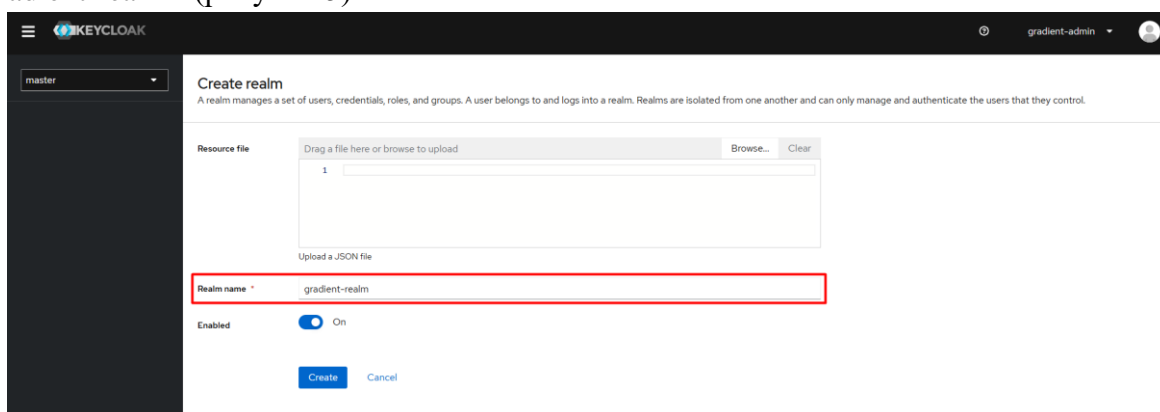


Рисунок 5. Интерфейс создания Realm

- Создаем **Client** для приложения Градиент

- нажимаем кнопку [Create client](#);

- на вкладке «General settings» в поле «Client ID» значения «gradient\_client» и нажимаем «Next» (рисунок 6);

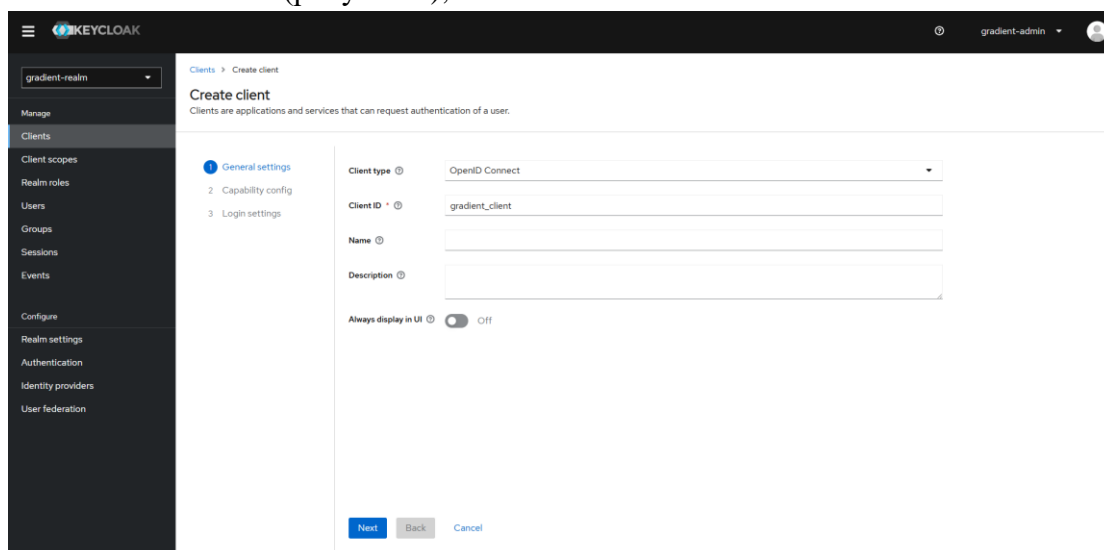


Рисунок 6. Интерфейс создания Client вкладка «General settings»



- на вкладке «Capability config» включаем аутентификацию и нажимаем «Next» (рисунок 7);

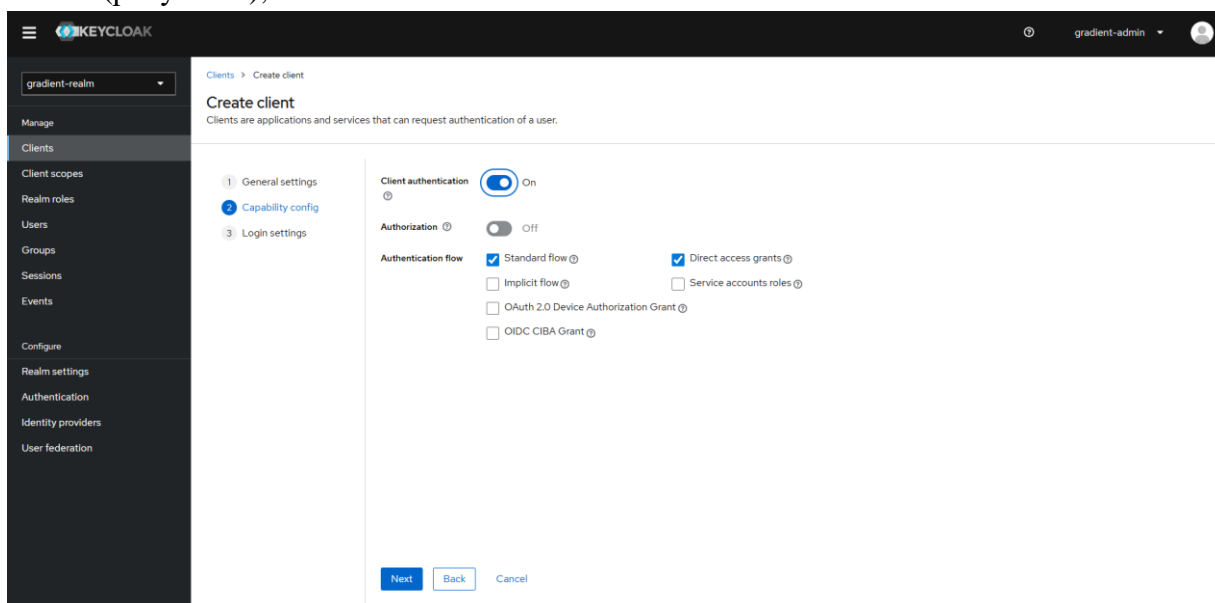


Рисунок 7. Интерфейс создания Client вкладка «Capability config»

- вкладку «Login settings» оставляем незаполненной и нажимаем «Save».
- Создаем пользователей для приложения Градиент
  - с помощью нажатия на кнопку **Create new user** в разделе Users;
  - указываем в поле «Username» значение «gradient-admin», почтовый ящик «admin@example.com» (для основного варианта указать реальный адрес) и с помощью переключателя указываем, что почтовый ящик верифицирован (рисунок 8);

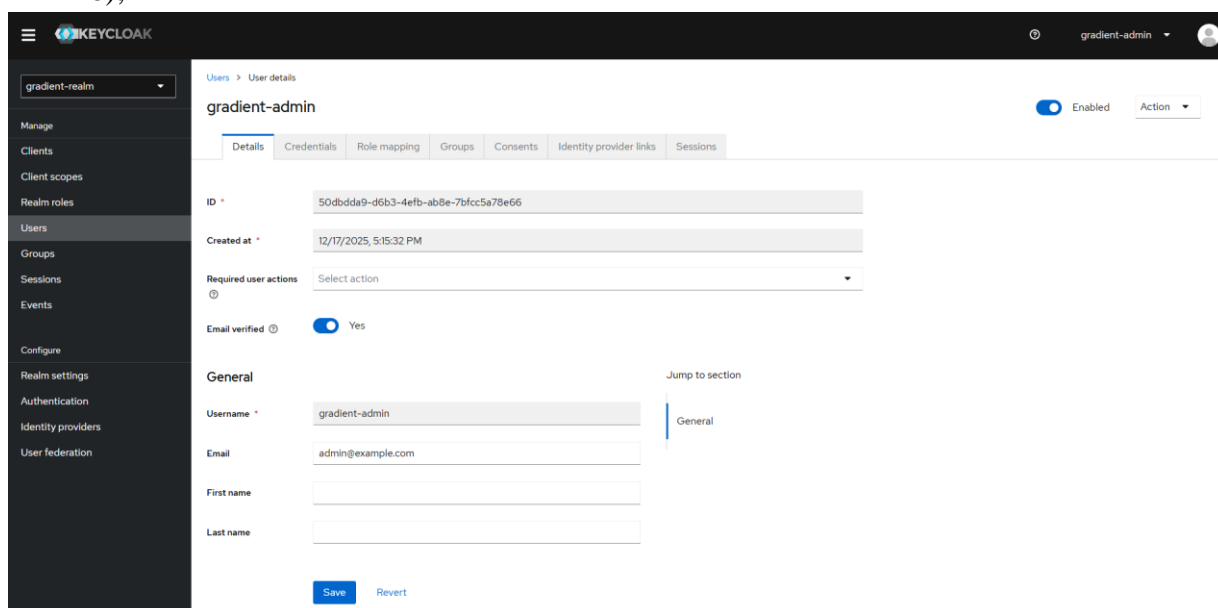


Рисунок 8. Интерфейс создания Пользователя

- на вкладке «Credentials» создаем пароль и с помощью переключателя указываем, что пароль не временный (рисунок 9)

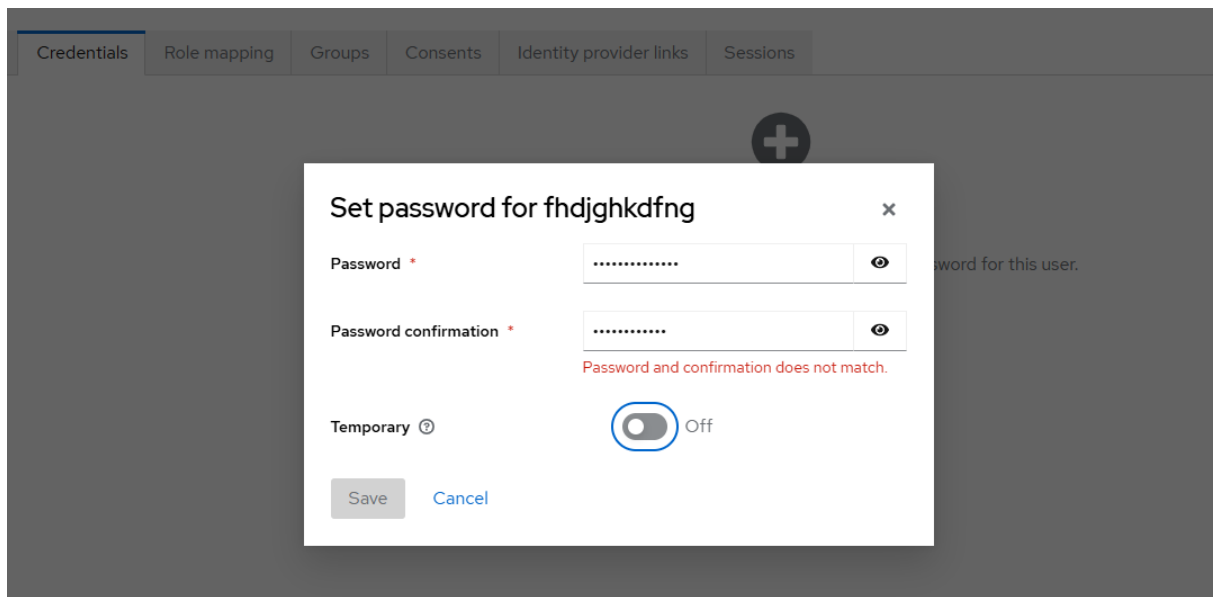


Рисунок 9. Создание пароля для почтового ящика пользователя

### 3.3.5 Заполнение общих параметров

В файле `env_back` заполнить:

```
POSTGRES_DB=gradient-db
POSTGRES_USER=gradient
POSTGRES_PASSWORD= qweRT12345 <указать данные>
POSTGRES_HOST=db
POSTGRES_PORT=5432

SECRET_STORE_ENCRYPTION_KEY="8RtKaTl51FhA"

INFLUX_URL=http://influxdb3-core:8181
INFLUX_DB=gradient_metric
INFLUX_TOKEN=apiv3_3h4BfKlZO_bGssGiLUL2mfHypZ9bKrB3h3V1HqkxEA-
r6W4IkaXijD0vXZwah3z3pN×q59AbBAZpV4dZK5k4MQ

VERIFY_SSL=False
KEYCLOAK_URL=https://gradient-demo.it-one.ru/keycloak
KEYCLOAK_INTERNAL_URL=http://keycloak:8080/keycloak
KEYCLOAK_EXTERNAL_URL=https://gradient-demo.it-one.ru/keycloak
KEYCLOAK_REALM=gradient-realm
KEYCLOAK_CLIENT_ID=gradient_client
KEYCLOAK_CLIENT_SECRET=uv3Apcx1r6GgGmMzGCCQz7nR2kC×w4Rb

BASE_URL=https://gradient-demo.it-one.ru
```

Для заполнения строки `KEYCLOAK_CLIENT_SECRET` в файле `env_back` данные необходимо скопировать со вкладки «Credentials» созданного нами ранее пользователя (рисунок 10)

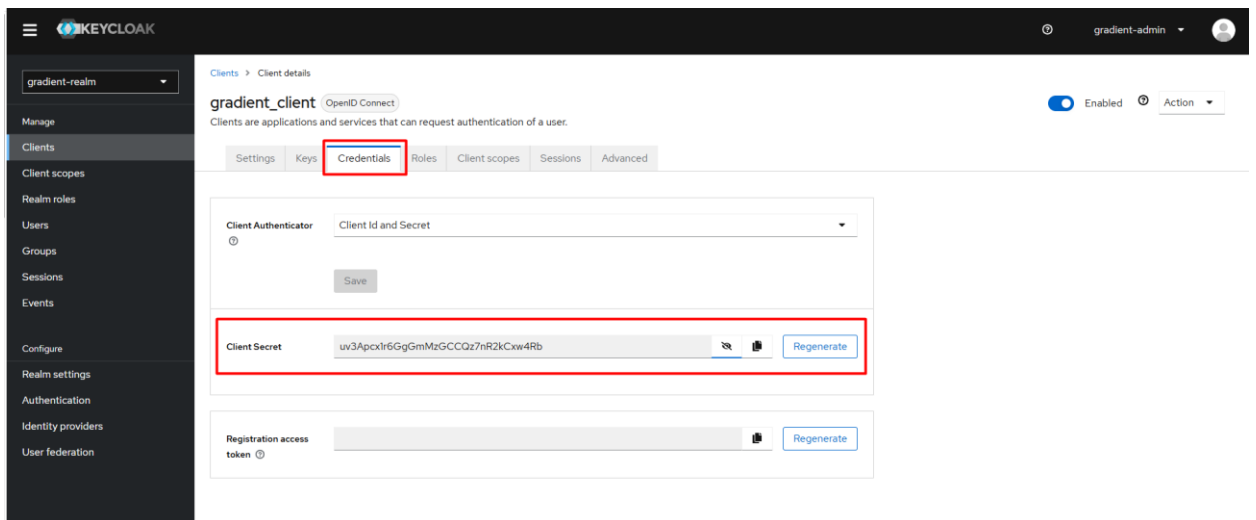


Рисунок 10. Данные Client Secret

### 3.3.6 Запуск Gradient

#### 1. Запускаем скрипт:

```
docker compose up -d
```

```
deploy@codefine-external-vm:/opt/services/gradient-ropo/gradient$ docker compose up -d
[+] Running 7/7
  Container gradient-kc_db-1      Running
  Container influxdb3-core       Running
  Container gradient-db-1        Healthy
  Container gradient-keycloak-1   Running
  Container gradient-backend      Running
  Container gradient-frontend     Running
  Container gradient-backend_init Exited
deploy@codefine-external-vm:/opt/services/gradient-ropo/gradient$
```

Рисунок 11. Вывод docker ps с контейнерами gradient-backend, db и keycloak

#### 2. Создаем администратора (рисунок 12):

```
docker compose exec -it gradient-backend python -m app.cli create-admin -f
Ivan -l Ivanov -e admin@example.com
```

```
deploy@codefine-external-vm:/opt/services/gradient-ropo/gradient$ docker compose exec -it gradient-backend python -m app.cli create-admin -f Ivan -l Ivanov -e admin@example.com
INFO | __main__:create_admin_async:52 - Пользователь Ivan Ivanov (admin@example.com) создан с ролью ADMIN
deploy@codefine-external-vm:/opt/services/gradient-ropo/gradient$
```

Рисунок 12. Создание пользователя

## 3.4 Apache Airflow

#### 1. Скопировать каталоги:

```
cp -r /opt/services/gradient-ropo/airflow /opt/services/gradient-ropo/gradient/
cp -r /opt/services/gradient-ropo/dags/dags_v1.0.9 /opt/services/gradient-ropo/gradient/airflow/
cd /opt/services/gradient-ropo/gradient/airflow
```

#### 2. Отредактировать .env, указав параметры доступа (логин и пароль) от Airflow.

```
_AIRFLOW_WWW_USER_USERNAME=airflow-user
_AIRFLOW_WWW_USER_PASSWORD=<указать данные>
AIRFLOW_IMAGE=gradient-airflow:2.11.0-python3.10-v1.0.9 - взять последнюю версию из дистрибутива
```

```
AIRFLOW__API__BASE_URL=http://server-name/airflow - url - на котором будет
доступен airflow
```

### 3. Инициализация:

```
./init_airflow.sh
ln -snf ./dags_v1.0.9 ./dags
docker compose up -d
```

### 4. Обновить файл /opt/services/gradient-ropo/gradient/airflow/connection.json (данные БД, S3 и токена influx).

Для заполнения данных для БД используем сведения из файла .env, для заполнения S3 – настройки, с которыми мы развернули minio (рисунок 13).

Так как ods и dds размещены в одном месте, то настройки для них будут одинаковые. Необходимость внесения данных токена influx (SECRET\_STORAGE\_ENCRYPTION\_KEY) обусловлена тем, что с его помощью зашифровываются пароли опрашиваемых систем. Это помогает не хранить пароли в открытом виде в БД, так как с помощью этого ключа пароли шифруются и расшифровываются им же. Требования к ключу такие же, как и к обычному паролю. Главное, чтобы ключ совпадал в файле connection.json и в файле env\_back в директории gradient.

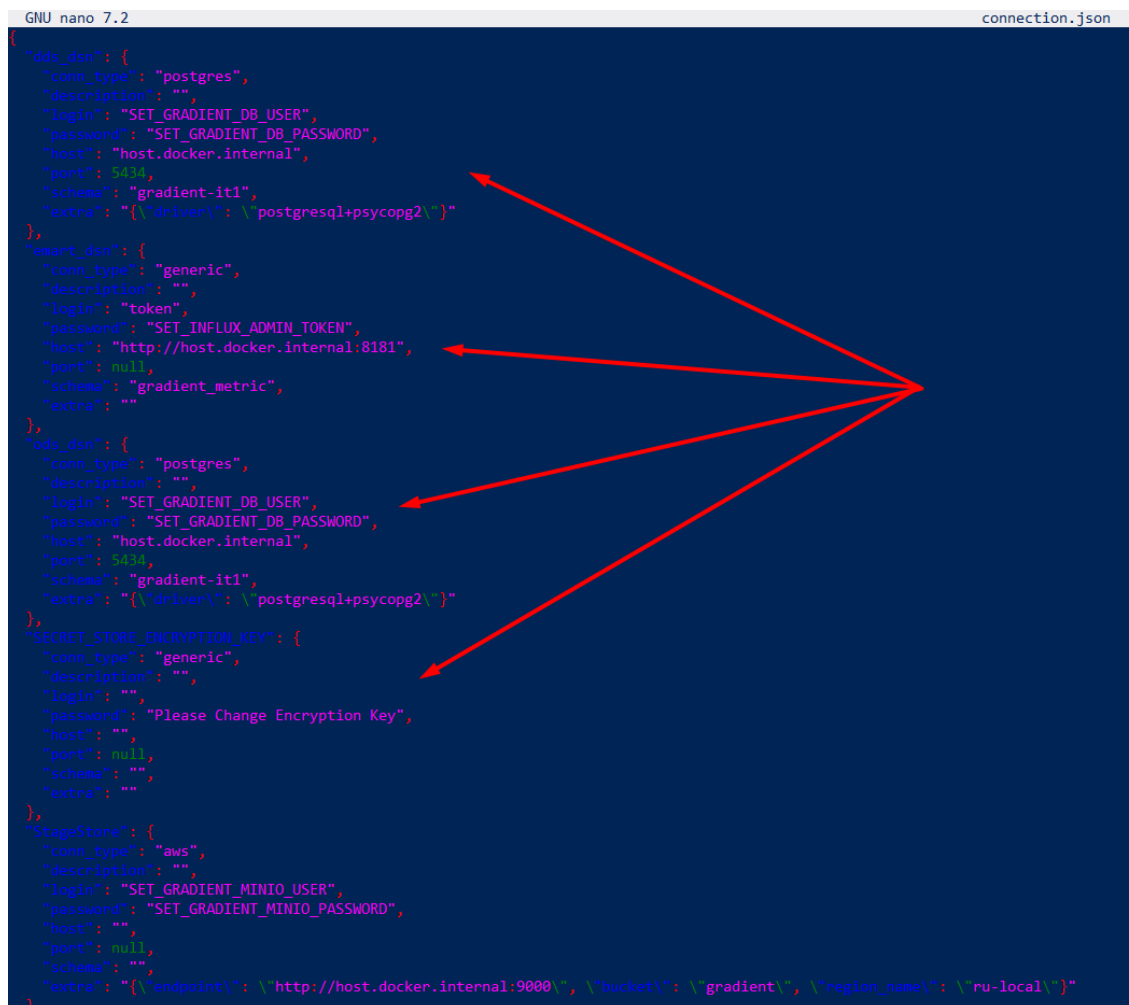


Рисунок 13. Обновление данных файла connection.json

### 5. Импортировать соединения:

Перед запуском команды необходимо убедиться, что вы находитесь в директории `/opt/services/gradient-ropo/airflow/`

```
docker compose exec -it airflow-webserver airflow connections import
./connection.json -overwrite
```

### 6. Применить миграции:

```
docker compose exec -it airflow-webserver airflow dags trigger
Gradient_DB_Migration
```

После установки становится доступным Apache Airflow по адресу: <https://gradient-demo.it-one.ru/airflow/> (рисунок 14).

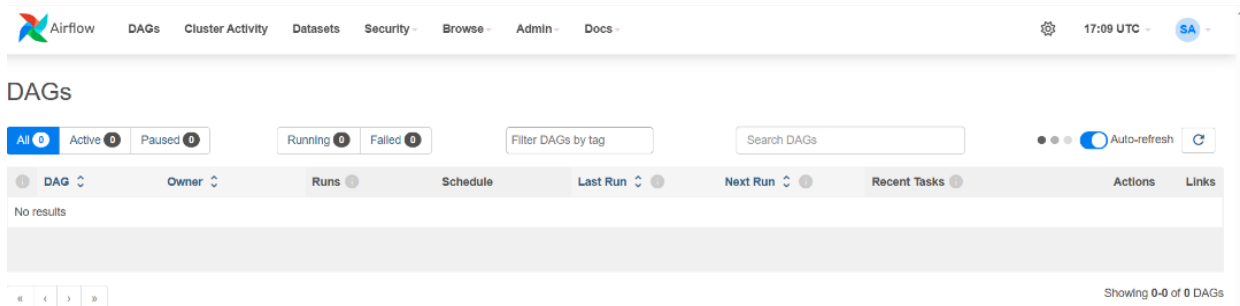


Рисунок 14. Внешний вид Apache Airflow

## 4. ДОБАВЛЕНИЕ ПОЛЬЗОВАТЕЛЯ В MINIO ЧЕРЕЗ CLI

### 1. Добавляем алиас:

```
docker exec minio_server mc alias set myminio http://localhost:9000 admin  
ivX-ss7-TBF-jkQ
```

### 2. Добавляем пользователя для работы с хранилищем:

```
docker exec minio_server mc admin user add myminio gradient-user gradient-  
password
```

### 3. Создаем файл политики доступа «gradient-policy.json»:

```
cat > gradient-policy.json << EOF  
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": [  
        "s3:GetObject",  
        "s3:PutObject",  
        "s3:DeleteObject",  
        "s3:ListBucket"  
      ],  
      "Resource": [  
        "arn:aws:s3:::gradient",  
        "arn:aws:s3:::gradient/*"  
      ]  
    }  
  ]  
}
```

### 4. Копируем файл политики в контейнер

```
docker cp gradient-policy.json minio_server:/tmp/gradient-policy.json
```

### 5. Создаем политику

```
docker exec minio_server mc admin policy create myminio gradient-access  
/tmp/gradient-policy.json
```

### 6. Назначаем политику пользователю

```
docker exec minio_server mc admin policy create myminio gradient-access  
/tmp/gradient-policy.json
```

### 7. Проверяем информацию о пользователе

```
docker exec minio_server mc admin user info myminio gradient-user
```

## 5. ПРОВЕРКА РАБОТОСПОСОБНОСТИ

- Перейти в интерфейс **Gradient** по адресу <https://gradient-demo.it-one.ru/keycloak/> (рисунок 15)
- Проверить вход через **Keycloak**
- Проверить подключение к **S3** и **InfluxDB**
- Убедиться, что в **Airflow** DAGs отображаются корректно

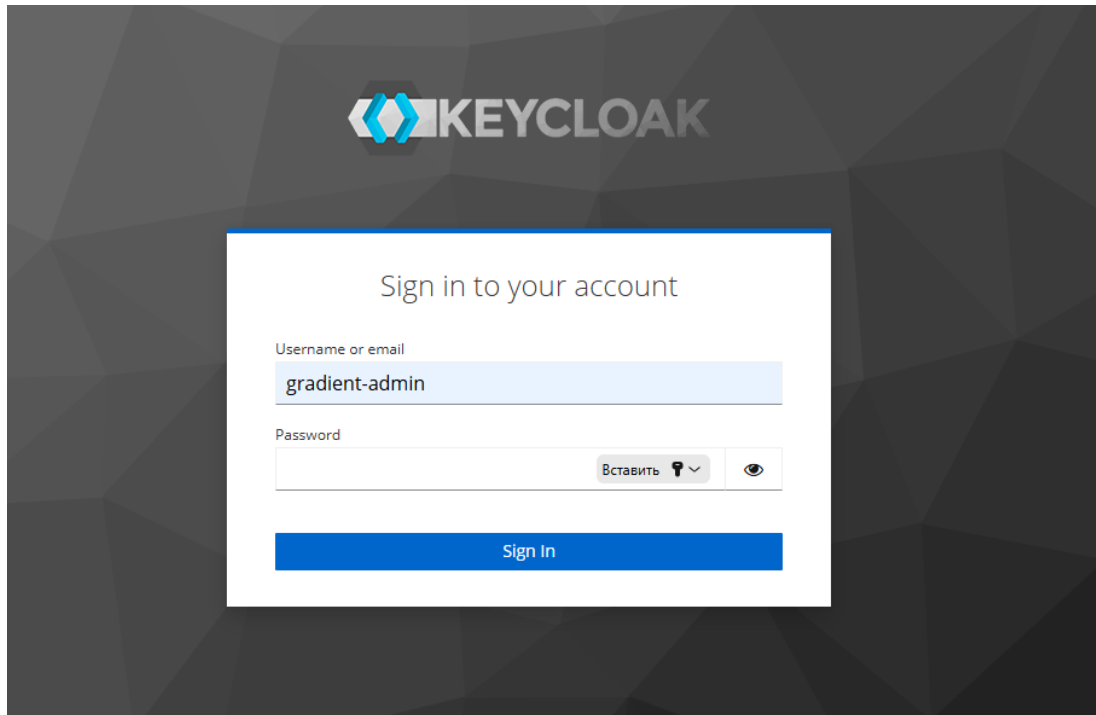


Рисунок 15. Экран входа в Gradient

- успешная авторизация
- список DAG в Airflow

## 6. ЗАВЕРШЕНИЕ УСТАНОВКИ

После успешного развертывания:

- Сохранить резервные копии файлов .env и env\_back.
- Настроить регулярное резервное копирование каталогов /opt/gradient и /opt/gradient\_setup (см. ниже)
- При необходимости настроить мониторинг контейнеров (docker ps, docker logs) и обновление образов.

### 6.1 Apache Airflow – подключение тестовых данных

Для дальнейшей работы необходимо выполнить загрузку данных из источников и настройку Airflow DAG

Для того, чтобы перенести тестовые обезличенные данные из контура компании в демонстрационный контур нужно провести этапы выгрузки обезличенных данных в stage внутри контура (рисунок 16).

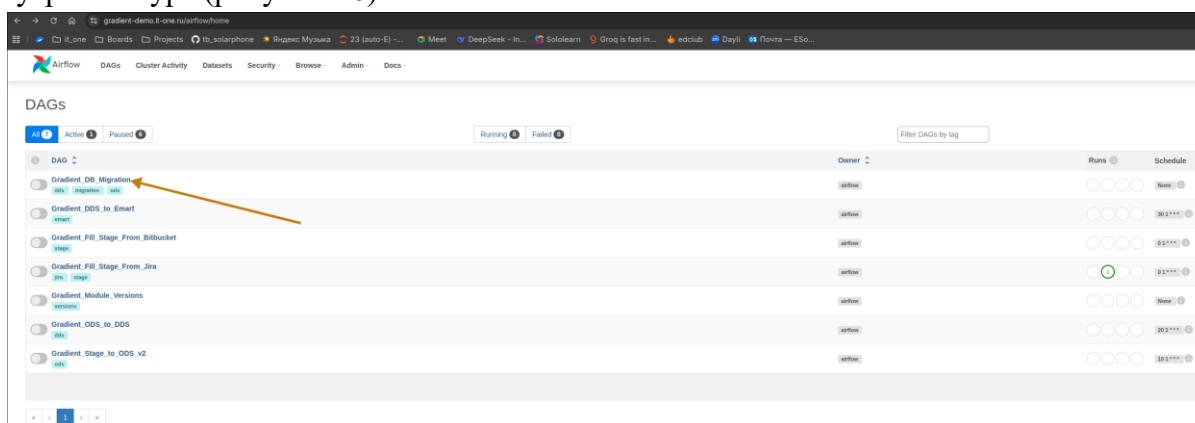


Рисунок 16. Выгрузки обезличенных данных в stage внутри контура

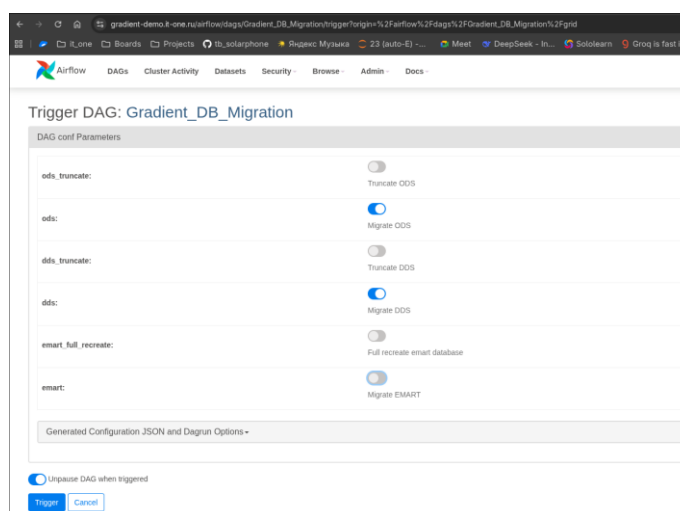


Рисунок 17. Выбор соответствующего DAG

Если в логах миграций есть ошибки, то необходимо проверить параметры подключения в Admin → Connections интерфейса Airflow.



Далее пошагово выполняем загрузки данных (рисунок 18):

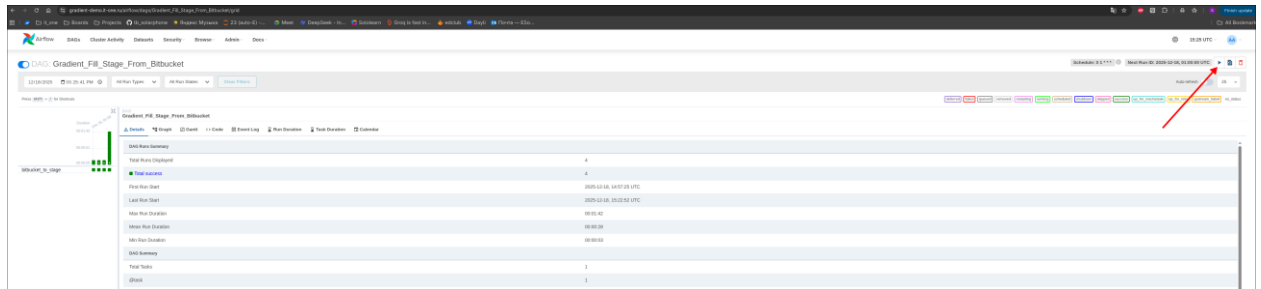


Рисунок 18. Загрузка данных

- загружаем данные из BitBucket (рисунок 19).

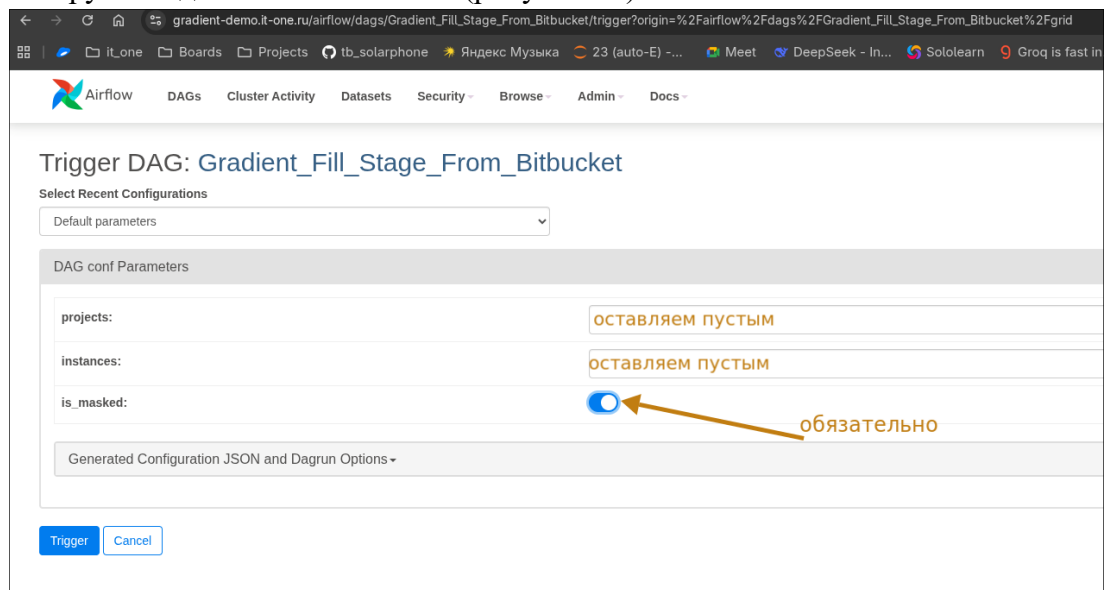


Рисунок 18. Загрузка данных из BitBucket

- загружаем данные в ODS слой (рисунок 19):

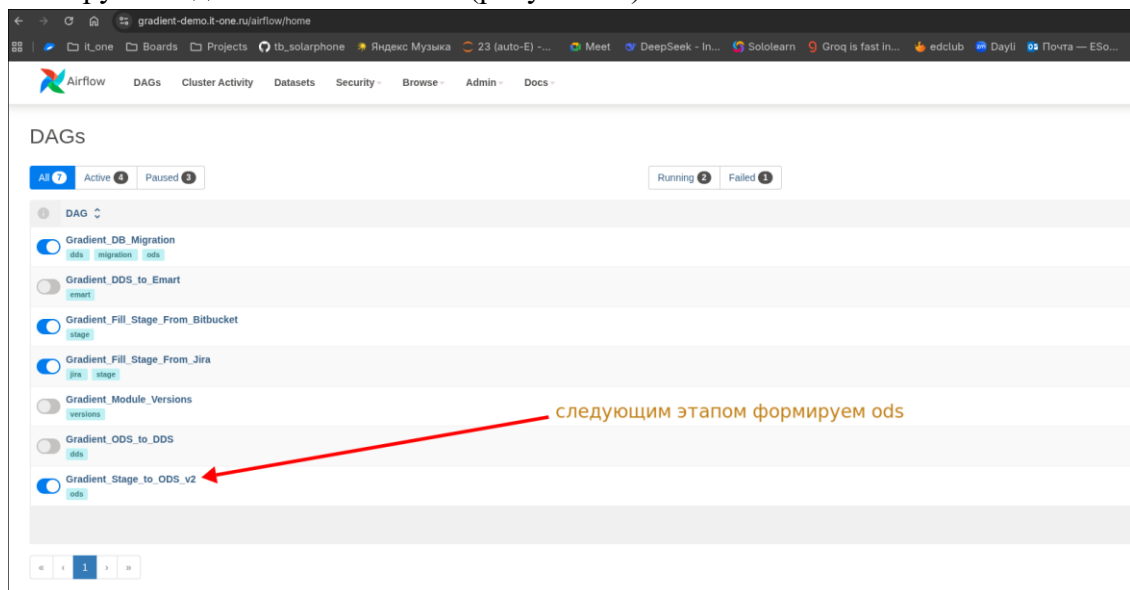


Рисунок 19. Загрузка данных в ODS слой

- указываем дату для сбора отчетности – ориентир полтора месяца назад (рисунок 20):

Trigger DAG: Gradient\_Stage\_to\_ODS\_v2

Select Recent Configurations

Default parameters

DAG conf Parameters

from\_date:

main\_destination\_save\_threshold: 1500

max\_concurrent\_storage\_reads: 16

Generated Configuration JSON and Dagrun Options

Trigger Cancel

указать дату начала сбора отчетности

Рисунок 20. Указание даты для сбора отчетности

- отмечаем состояния (рисунок 21):

Triggered Gradient\_Stage\_to\_ODS\_v2 with new Run ID manual\_\_2025-12-18T15:43:20+00:00, it should start any moment now.

DAG: Gradient\_Stage\_to\_ODS\_v2

12/18/2025 03:46:43 PM All Run Types All Run States Clear Filters

Press **shift** + **z** for Shortcuts

Duration 00:00:33 00:00:16 00:00:00 00:00:00 00:00:00

wait\_stage\_bitbucket wait\_stage\_jira stage\_to\_ods\_jira stage\_to\_ods\_bitbucket

DAG Run Details

Run ID manual\_\_2025-12-18T15:43:20+00:00

Run type manual

Run duration 00:00:11

Last scheduling decision 2025-12-18, 15:46:52 UTC

Queried at 2025-12-18, 15:46:44 UTC

Started 2025-12-18, 15:46:44 UTC

Data interval start 2025-12-17, 01:10:00 UTC

Data interval end 2025-12-18, 01:10:00 UTC

Externally triggered True

нажать на эти этапы ожидания левой кнопкой мыши и нажать на кнопку Mark state as... Success

Рисунок 21. Отметка состояния

- загружаем данные в DDS слой (рисунок 22):

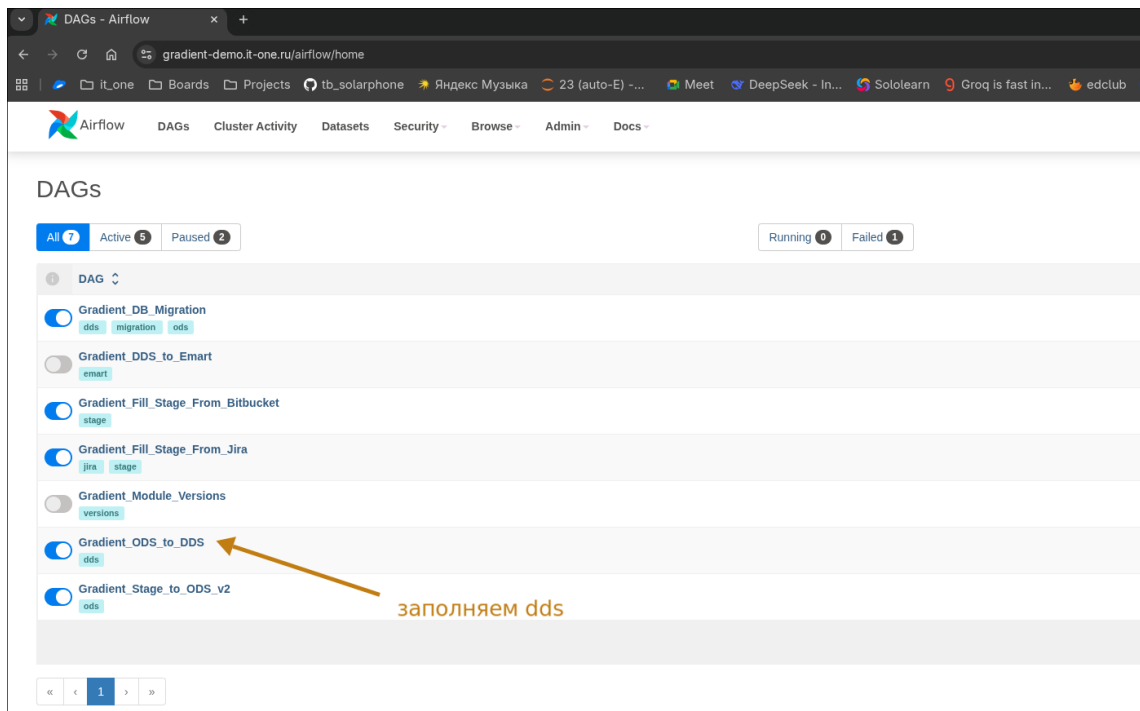


Рисунок 22. Загрузка данных в DDS слой

- указываем дату начала периода (рисунок 23):

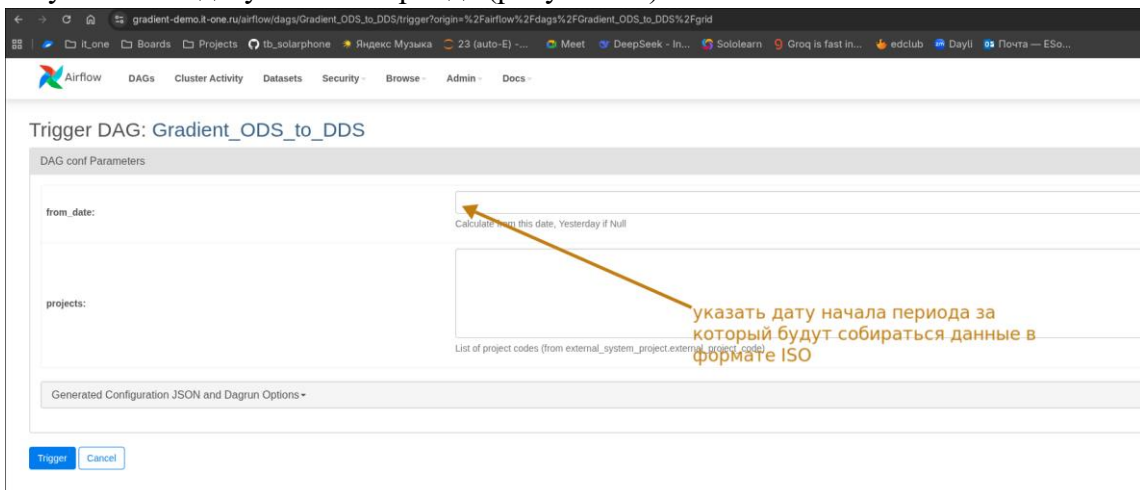


Рисунок 23. Указание даты начала периода

- пропускаем этап ожидания предыдущего DAG (рисунок 24):

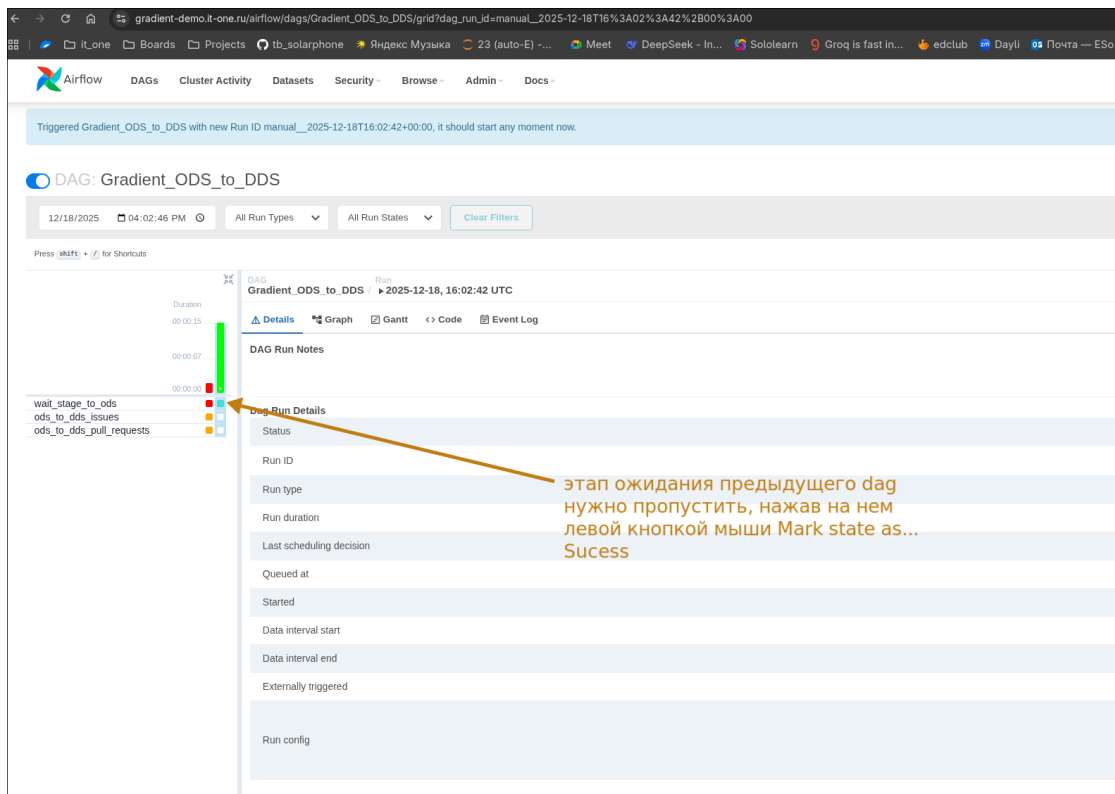


Рисунок 24. Пропуск этапа ожидания предыдущего DAG

- выгружаем данные в витрину (рисунок 25):

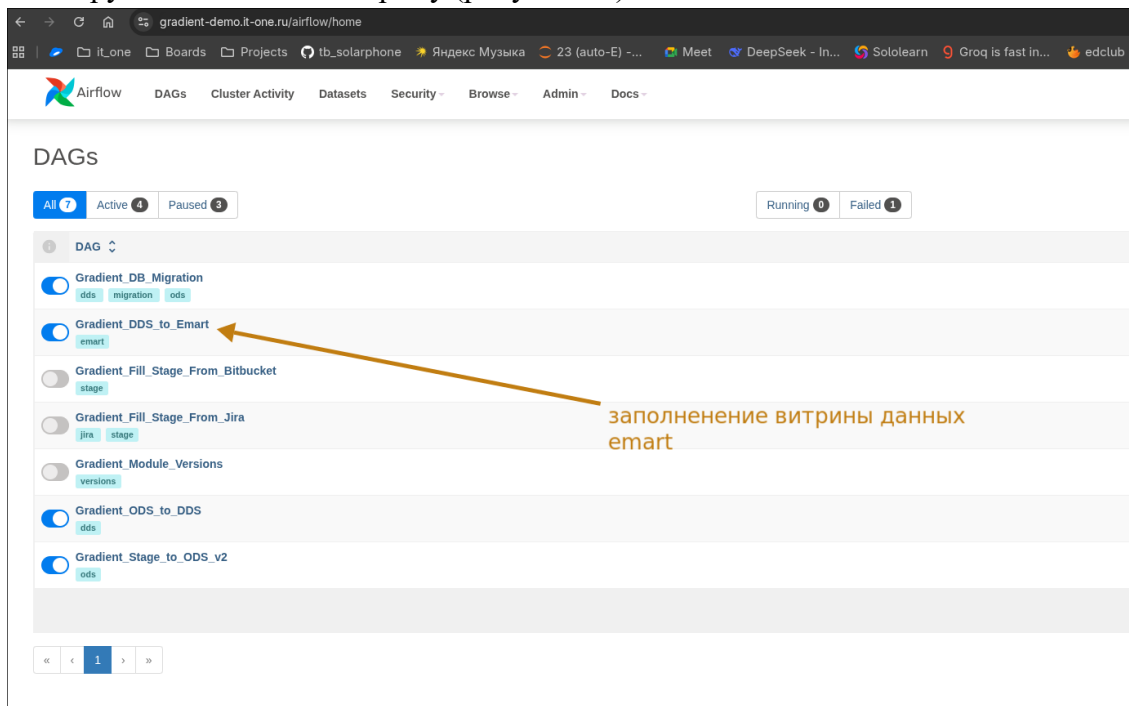


Рисунок 25. Выгрузка данных в витрину

## 7. КОНТАКТЫ

**Наименование организации:** ООО «ИТ1»

**ИНН:** 5010028861

**ОГРН:** 1065010021284

**Служба технической поддержки:** тел. 8 (495) 274-06-77

**По общим вопросам обращайтесь по адресу:** 129085, г. Москва, ул. Годовикова, 9, 17,  
Этаж 6 Помещ. 1

**Телефон:** 8 (495) 274-06-77

По техническим вопросам, связанным с проведением экспертной проверки программного обеспечения, можно обращаться к сотруднику ООО «ИТ1»:

**ФИО:** Левшин Дмитрий Кириллович

**почта:** [DLevshin@it-one.ru](mailto:DLevshin@it-one.ru)

**тел.:** 8 (495) 274-06-77