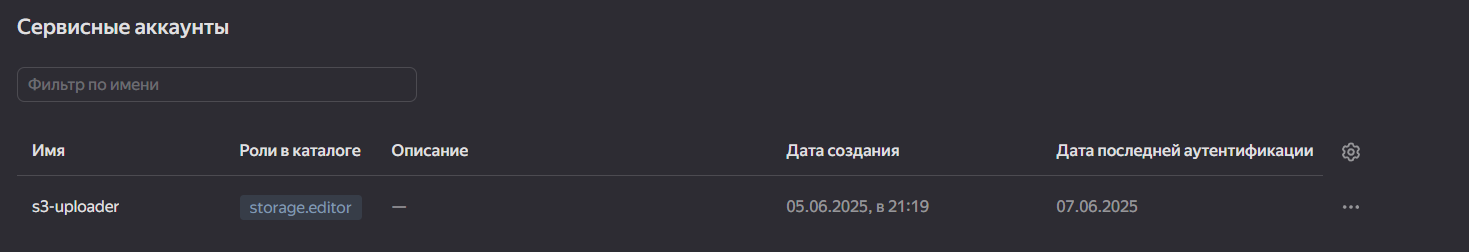
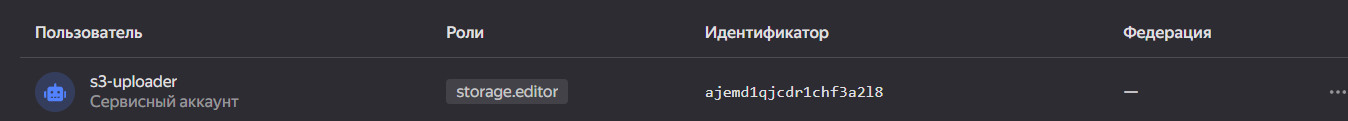
**Отчет Object Storage**

**Шаг 1**

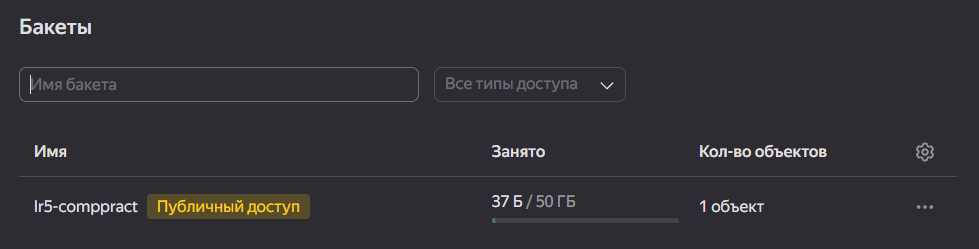
Созадние сервисного аккаунта



Установка прав доступа + ключ



Создание бакета



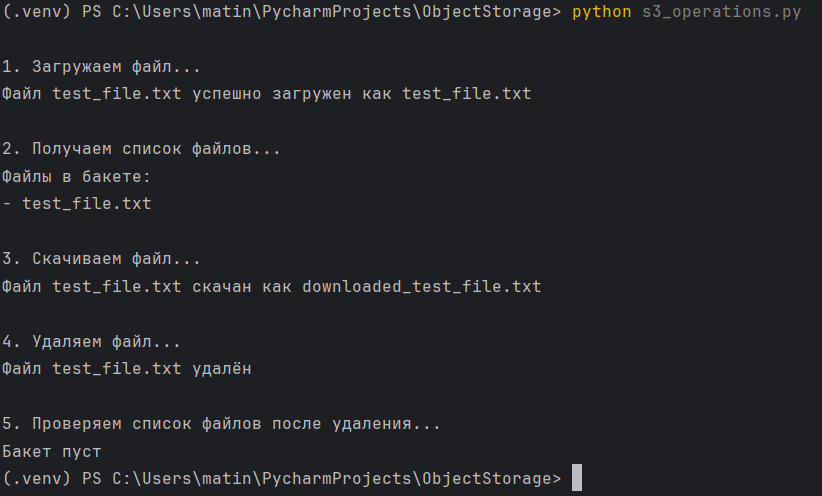
**Шаг 2**

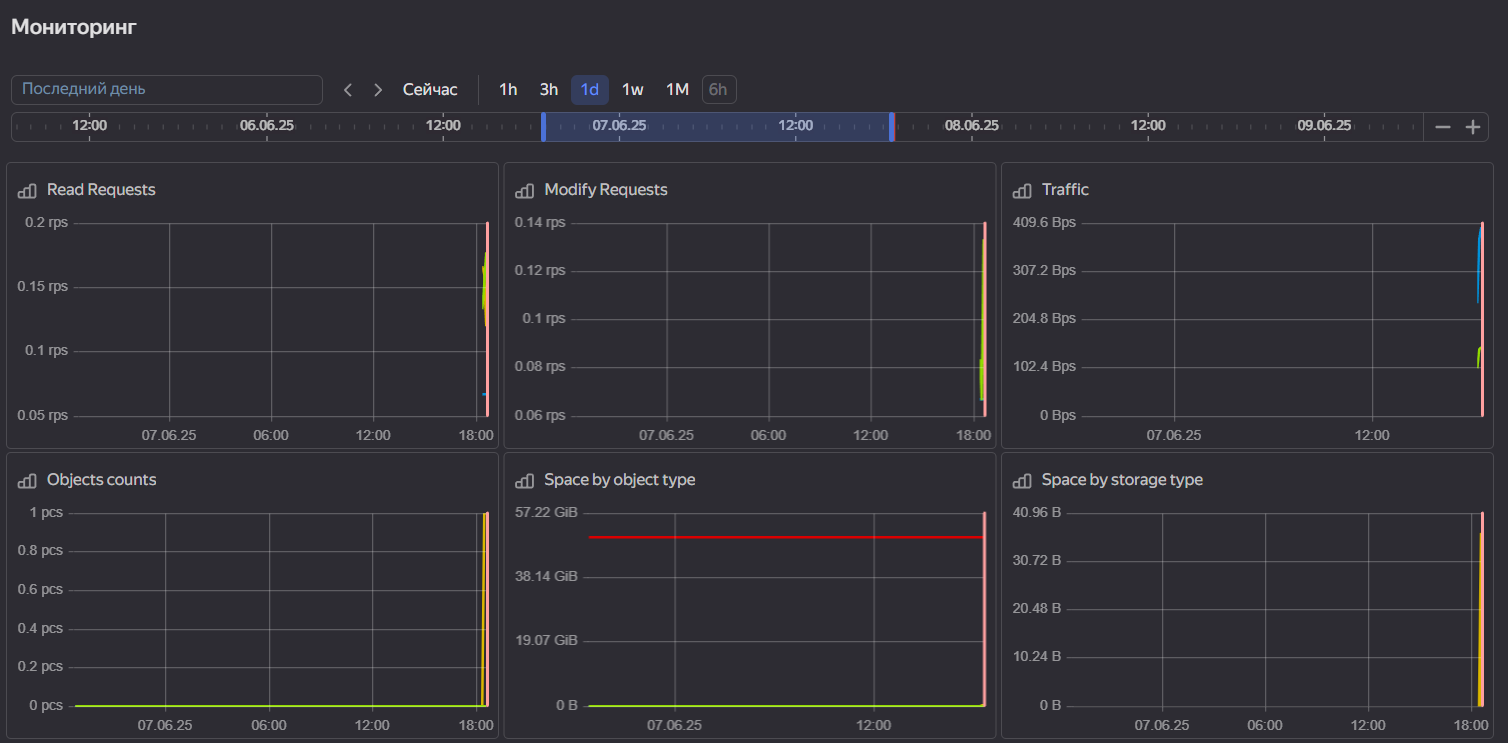
Реализация функции (локально) для выполнения основных операций с объектами в хранилище: получение списка загруженных файлов, загрузка файла в бакет, получение файла, удаление файла.

**Код программы**

import os  
import boto3  
from botocore.client import Config  
from botocore.exceptions import ClientError  
from dotenv import load\_dotenv  
# Загрузка переменных окружения  
load\_dotenv()  
# Конфигурация клиента S3  
s3 = boto3.client(  
 's3',  
 endpoint\_url=os.getenv('ENDPOINT\_URL'),  
 aws\_access\_key\_id=os.getenv('AWS\_ACCESS\_KEY\_ID'),  
 aws\_secret\_access\_key=os.getenv('AWS\_SECRET\_ACCESS\_KEY'),  
 region\_name=os.getenv('REGION\_NAME'),  
 config=Config(  
 signature\_version='s3v4',  
 s3={'addressing\_style': 'virtual'}  
 )  
)  
bucket\_name = os.getenv('BUCKET\_NAME')  
def upload\_file(file\_path, object\_name=None):  
 *"""Загрузка файла в бакет"""* if object\_name is None:  
 object\_name = os.path.basename(file\_path)  
 try:  
 s3.upload\_file(  
 Filename=file\_path,  
 Bucket=bucket\_name,  
 Key=object\_name,  
 ExtraArgs={  
 'ACL': 'private',  
 'ContentType': 'text/plain'  
 }  
 )  
 print(f"Файл {file\_path} успешно загружен как {object\_name}")  
 return True  
 except ClientError as e:  
 print(f"Ошибка загрузки: {e.response['Error']['Message']}")  
 return False  
def list\_files():  
 *"""Получение списка файлов в бакете"""* try:  
 response = s3.list\_objects\_v2(Bucket=bucket\_name)  
 if 'Contents' in response:  
 files = [obj['Key'] for obj in response['Contents']]  
 print("Файлы в бакете:")  
 for file in files:  
 print(f"- {file}")  
 return files  
 print("Бакет пуст")  
 return []  
 except ClientError as e:  
 print(f"Ошибка при получении списка: {e.response['Error']['Message']}")  
 return None  
def download\_file(object\_name, local\_path=None):  
 *"""Скачивание файла из бакета"""* if local\_path is None:  
 local\_path = object\_name  
 try:  
 s3.download\_file(bucket\_name, object\_name, local\_path)  
 print(f"Файл {object\_name} скачан как {local\_path}")  
 return True  
 except ClientError as e:  
 print(f"Ошибка скачивания: {e.response['Error']['Message']}")  
 return False  
def delete\_file(object\_name):  
 *"""Удаление файла из бакета"""* try:  
 s3.delete\_object(Bucket=bucket\_name, Key=object\_name)  
 print(f"Файл {object\_name} удалён")  
 return True  
 except ClientError as e:  
 print(f"Ошибка удаления: {e.response['Error']['Message']}")  
 return False  
if \_\_name\_\_ == "\_\_main\_\_":  
 # Тестовый файл  
 test\_file = "test\_file.txt"  
 with open(test\_file, 'w', encoding='utf-8') as f:  
 f.write("Тестовое содержимое для Yandex Object Storage")  
 # 1. Загрузка файла  
 print("\n1. Загружаем файл...")  
 upload\_file(test\_file)  
 # 2. Список файлов  
 print("\n2. Получаем список файлов...")  
 list\_files()  
 # 3. Скачивание файла  
 print("\n3. Скачиваем файл...")  
 downloaded\_file = "downloaded\_" + test\_file  
 download\_file(test\_file, downloaded\_file)  
 # 4. Удаление файла  
 print("\n4. Удаляем файл...")  
 delete\_file(test\_file)  
 # 5. Проверка после удаления  
 print("\n5. Проверяем список файлов после удаления...")  
 list\_files()  
 # Удаляем локальные файлы  
 os.remove(test\_file)  
 os.remove(downloaded\_file)

**Результат**





**Комментарий**

Секретный ключ и идентификатор лежат в отдельном файле

