

Департамент образования и науки города Москвы  
Государственное автономное образовательное учреждение  
высшего образования города Москвы  
«Московский городской педагогический университет»  
Институт цифрового образования  
Департамент информатики, управления и технологий

ДИСЦИПЛИНА:

«Проектный практикум по разработке ETL-решений»

**Практическая работа №5**

**Тема:**

**«Airflow DAG».**

Выполнила: Овсепян Милена, АДЭУ-201

Преподаватель: Босенко Т.М.

Москва

2023

## **СОДЕРЖАНИЕ**

<b>ГЛАВА 1. ПОСТАНОВКА ЗАДАЧИ.....</b>	<b>3</b>
<b>ГЛАВА 2. РЕШЕНИЕ ЗАДАЧИ.....</b>	<b>5</b>
<b>2.1 Исходный код всех DAGs.....</b>	<b>5</b>
<b>2.2 Граф DAG в Apache Airflow.....</b>	<b>5</b>
<b>2.3 Верхнеуровневая архитектура задания Бизнес-кейса «Rocket»</b>	<b>6</b>
<b>2.4 Архитектура DAG Бизнес-кейса «Rocket» .....</b>	<b>6</b>
<b>2.5 Выгрузка лог-файлов результатов работы DAGs в Apache</b>	
<b>Airflow .....</b>	<b>7</b>
<b>2.6 Выгрузка картинок.....</b>	<b>9</b>
<b>2.7 Диаграмма Ганта DAG в Apache Airflow. ....</b>	<b>10</b>
<b>2.8 Автоматизация выгрузки данных из контейнера в основную</b>	
<b>ОС данных.....</b>	<b>10</b>
<b>ЗАКЛЮЧЕНИЕ .....</b>	<b>11</b>

## ГЛАВА 1. ПОСТАНОВКА ЗАДАЧИ

1. Развернуть ВМ [ubuntu\\_mgpu.ova](#) в [VirtualBox](#).
2. Клонировать на ПК задание **Бизнес-кейс «Rocket»** в домашний каталог ВМ.  
`git clone https://github.com/BosenkoTM/workshop-on-ETL.git`
3. Запустить контейнер с кейсом, изучить основные элементы DAG в Apache Airflow.
  - Создать DAG согласно алгоритму, который предоставит преподаватель.
  - Изучить логи, выполненного DAG. Скачать логи из контейнера на основную ОС, используя команду:
  - `docker cp <container_hash>: /path/to/zip/file.zip /path/on/host/new_name.zip`
  - Выгрузить полученный результат работы DAG в основной каталог ОС, используя команду: `docker cp -r <containerId>:/path/to/directory /path/on/host`
4. Создать исполняемый файл с расширением .sh, который автоматизирует выгрузку данных из контейнера в основную ОС данных, полученные в результате работы DAG в Apache Airflow.
5. Спроектировать верхнеуровневую архитектуру аналитического решения задания **Бизнес-кейса «Rocket»** в draw.io.
6. Спроектировать архитектуру DAG **Бизнес-кейса «Rocket»** в draw.io. Необходимо использовать:
7. Построить диаграмму Ганта работы DAG в Apache Airflow.
8. Результаты исследований представить в виде файла ФИО-05.pdf, в котором отражены следующие результаты:
  - постановка задачи;

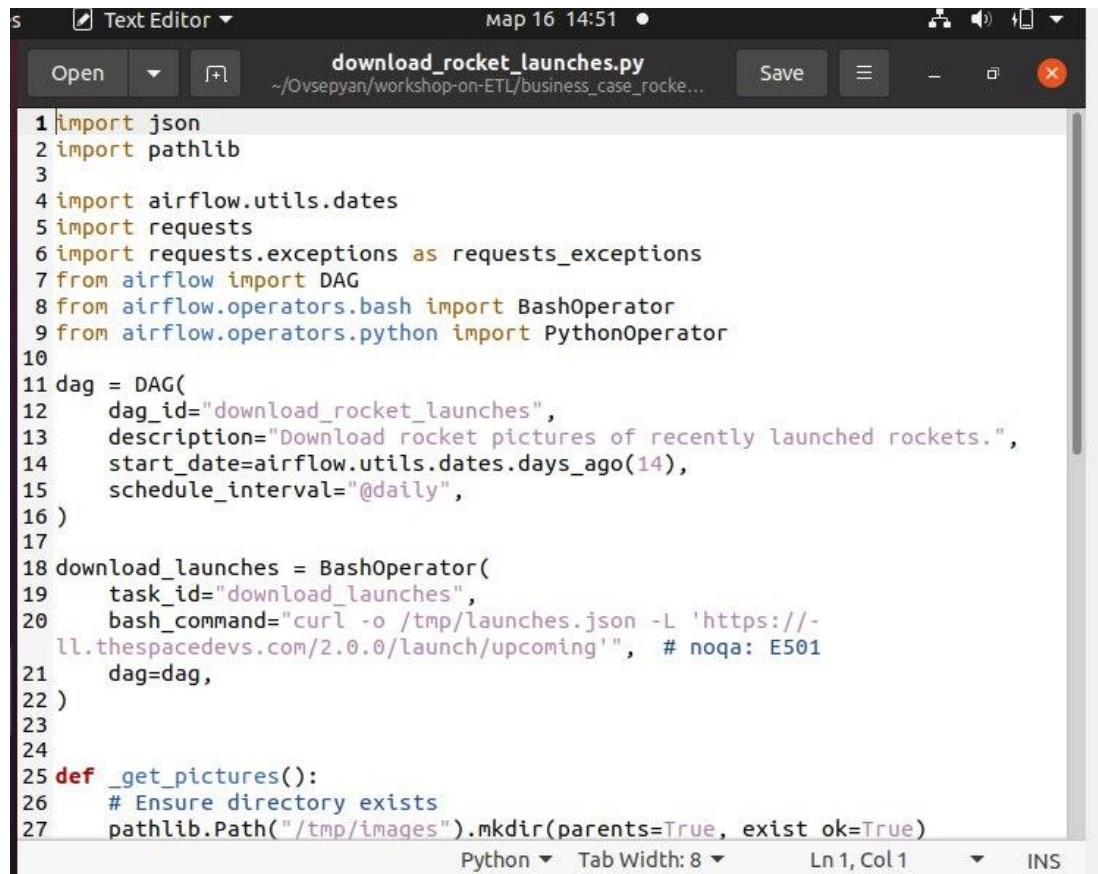
- исходный код всех DAGs, которые требовались для решения задачи, а также представить граф DAG в Apache Airflow;
- верхнеуровневая архитектура задания **Бизнес-кейса «Rocket»**, выполненная в draw.io;
- архитектура DAG **Бизнес-кейса «Rocket»**, выполненная в draw.io;
- скрин лог-файла результатов работы DAGs в Apache Airflow;
- диаграмма Ганта DAG в Apache Airflow.

После проверки преподавателем работоспособности DAG, выгрузить отчет на портал [moodle](#).

## ГЛАВА 2. РЕШЕНИЕ ЗАДАЧИ

### 2.1 Исходный код всех DAGs

Исходный код всех DAGs, которые требовались для решения задачи представлен на рисунке 1.



```
1 import json
2 import pathlib
3
4 import airflow.utils.dates
5 import requests
6 import requests.exceptions as requests_exceptions
7 from airflow import DAG
8 from airflow.operators.bash import BashOperator
9 from airflow.operators.python import PythonOperator
10
11 dag = DAG(
12     dag_id="download_rocket_launches",
13     description="Download rocket pictures of recently launched rockets.",
14     start_date=airflow.utils.dates.days_ago(14),
15     schedule_interval="@daily",
16 )
17
18 download_launches = BashOperator(
19     task_id="download_launches",
20     bash_command="curl -o /tmp/launches.json -L 'https://-
21     ll.thespacedevs.com/2.0.0/launch/upcoming'", # noqa: E501
22     dag=dag,
23 )
24
25 def _get_pictures():
26     # Ensure directory exists
27     pathlib.Path("/tmp/images").mkdir(parents=True, exist_ok=True)
```

Рисунок 1. Исходный код всех DAGs

### 2.2 Граф DAG в Apache Airflow

Вкладка Graph View представлен на рисунке 2.

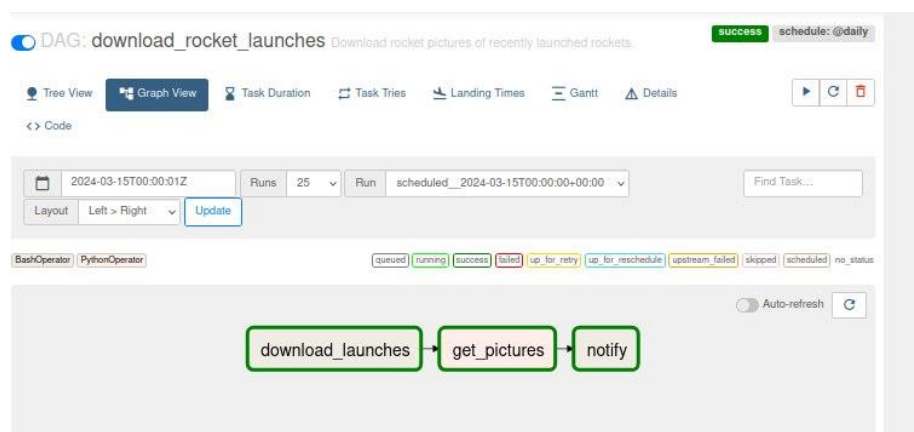


Рисунок 2. Граф DAG

## 2.3 Верхнеуровневая архитектура задания Бизнес-кейса «Rocket»

Верхнеуровневая архитектура задания Бизнес-кейса «Rocket», выполненная в draw.io представлена на рисунке 3.

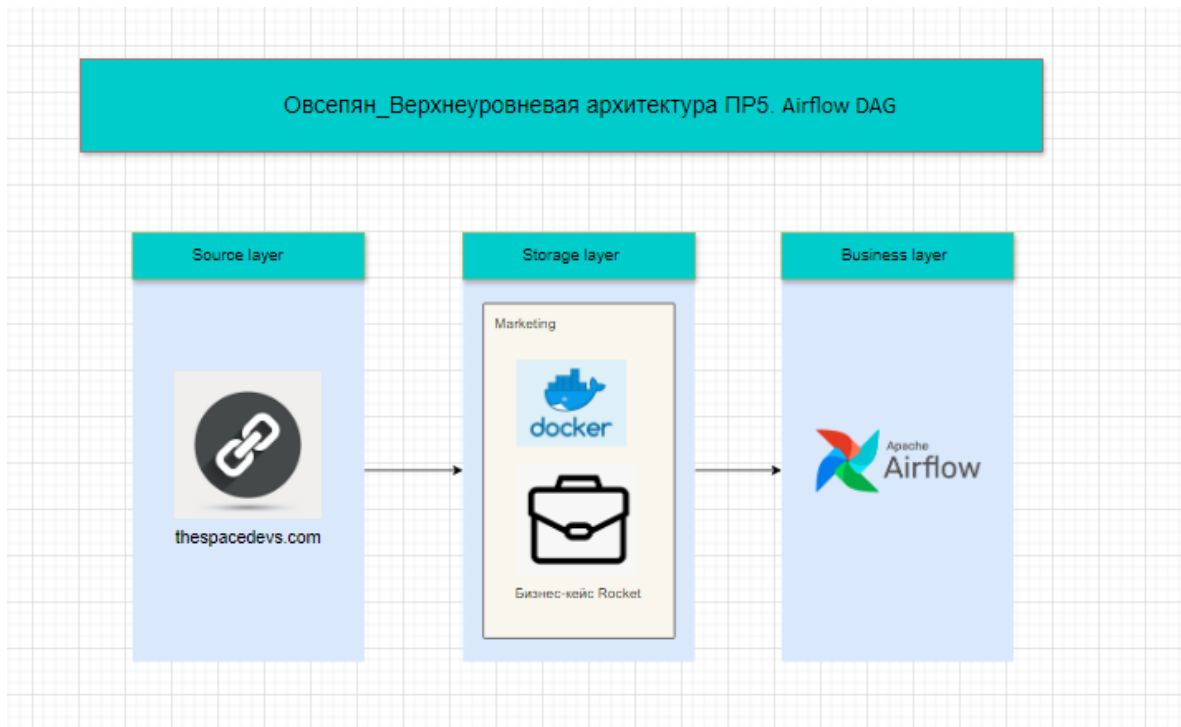


Рисунок 4. Верхнеуровневая архитектура

Более подробно с верхнеуровневой архитектурой можно ознакомиться по ссылке: [https://viewer.diagrams.net/?tags=%7B%7D&highlight=0000ff&edit=\\_blank&layers=1&nav=1&title=%D0%9E%D0%B2%D1%81%D0%B5%D0%BF%D1%8F%D0%BD\\_%D0%B2%D0%B5%D1%80%D1%85%D0%BD%D0%B5%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D0%B5%D0%B2%D0%B0%D1%8F.drawio#Uhttps%3A%2F%2Fdrive.google.com%2Fuc%3Fid%3D1dGZx4J4izbU2SQkA20P4lUohzxeIk5Md%26export%3Ddownload](https://viewer.diagrams.net/?tags=%7B%7D&highlight=0000ff&edit=_blank&layers=1&nav=1&title=%D0%9E%D0%B2%D1%81%D0%B5%D0%BF%D1%8F%D0%BD_%D0%B2%D0%B5%D1%80%D1%85%D0%BD%D0%B5%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D0%B5%D0%B2%D0%B0%D1%8F.drawio#Uhttps%3A%2F%2Fdrive.google.com%2Fuc%3Fid%3D1dGZx4J4izbU2SQkA20P4lUohzxeIk5Md%26export%3Ddownload)

## 2.4 Архитектура DAG Бизнес-кейса «Rocket»

Архитектура DAG Бизнес-кейса «Rocket» выполненная в draw.io представлена на рисунке 4.

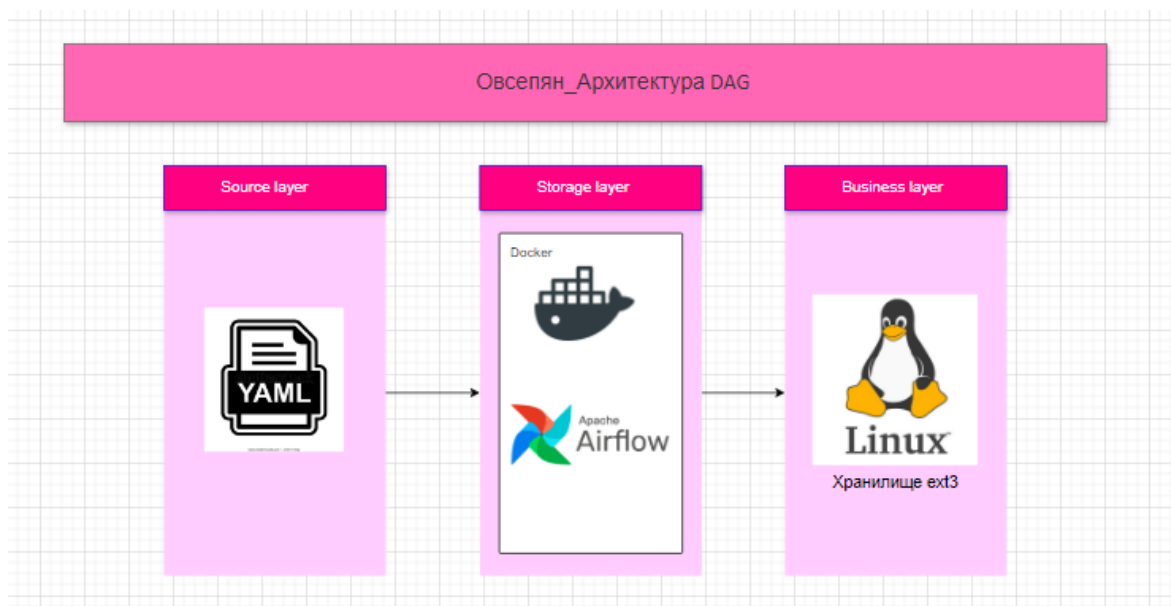


Рисунок 4. Архитектура DAG Бизнес-кейса «Rocket»

Более подробно с архитектурой можно ознакомиться по ссылке: <https://viewer.diagrams.net/?tags=%7B%7D&highlight=0000ff&edit=blank&layers=1&nav=1&title=%D0%9E%D0%B2%D1%81%D0%B5%D0%BF%D1%8F%D0%BD%D1%81%D1%85%D0%B5%D0%BC%D0%B0%20DAG.drawio#Uhttps%3A%2F%2Fdrive.google.com%2Fuc%3Fid%3D1tfFMt2SbUgXDB3QeIOIS245CmqBXqMrI%26export%3Ddownload>

## 2.5 Выгрузка лог-файлов результатов работы DAGs в Apache Airflow

Выгружаем файлы с логами. Команды представлены на рисунке 5.

```
mgpu@mgpu-VirtualBox:~$ sudo docker cp 2a9:/opt/airflow/logs/download_rocket_launches/download_launches/2024-03-02T00:00:00+00:00/1.log /home/mgpu/Downloads/logs.log
Successfully copied 5.12kB to /home/mgpu/Downloads/logs.log
mgpu@mgpu-VirtualBox:~$ sudo docker cp 2a9:/opt/airflow/logs/download_rocket_launches/get_pictures/2024-03-02T00:00:00+00:00/1.log /home/mgpu/Downloads/1logs.log
Successfully copied 6.66kB to /home/mgpu/Downloads/1logs.log
mgpu@mgpu-VirtualBox:~$ sudo docker cp 2a9:/opt/airflow/logs/download_rocket_launches/notify/2024-03-02T00:00:00+00:00/1.log /home/mgpu/Downloads/2logs.log
Successfully copied 5.12kB to /home/mgpu/Downloads/2logs.log
```

Рисунок 5. Выгрузка 3 файлов с логами.

Результаты выгрузки представлены на рисунках 6-9.



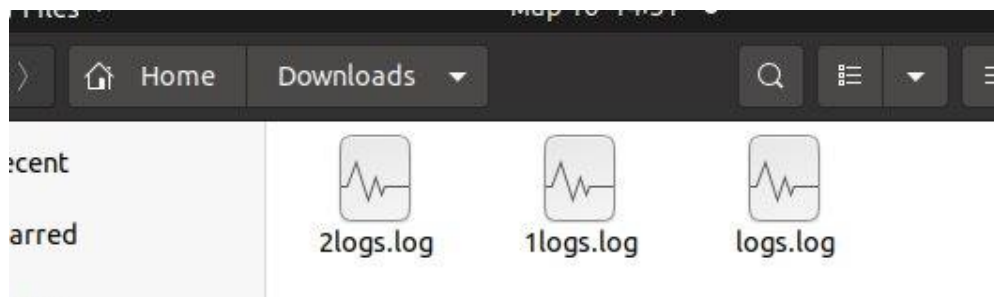


Рисунок 6. Результат выгрузки 3 файлов с логами.

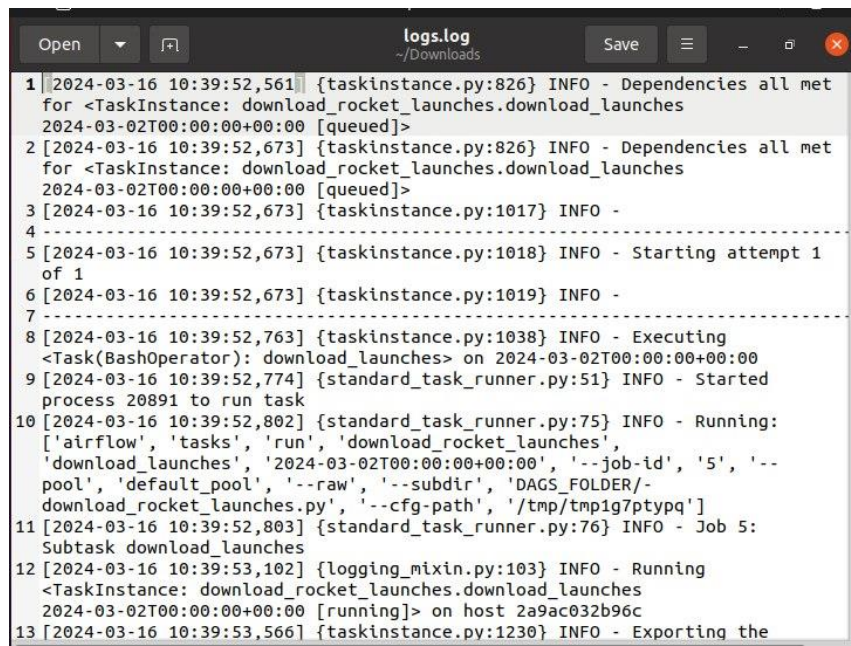


Рисунок 7. Файл 1 «logs.log».

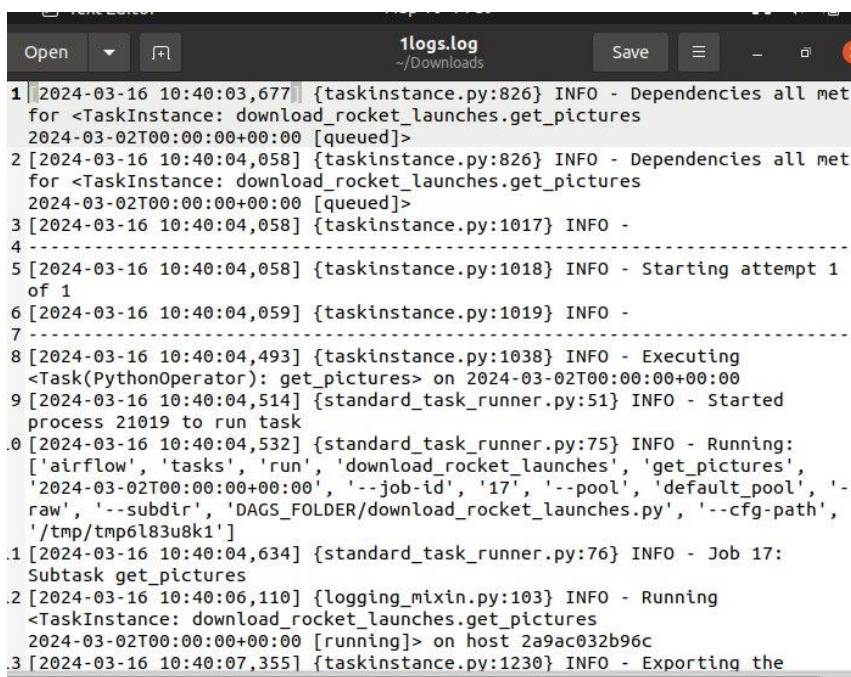


Рисунок 8. Файл 2 «1logs.log».



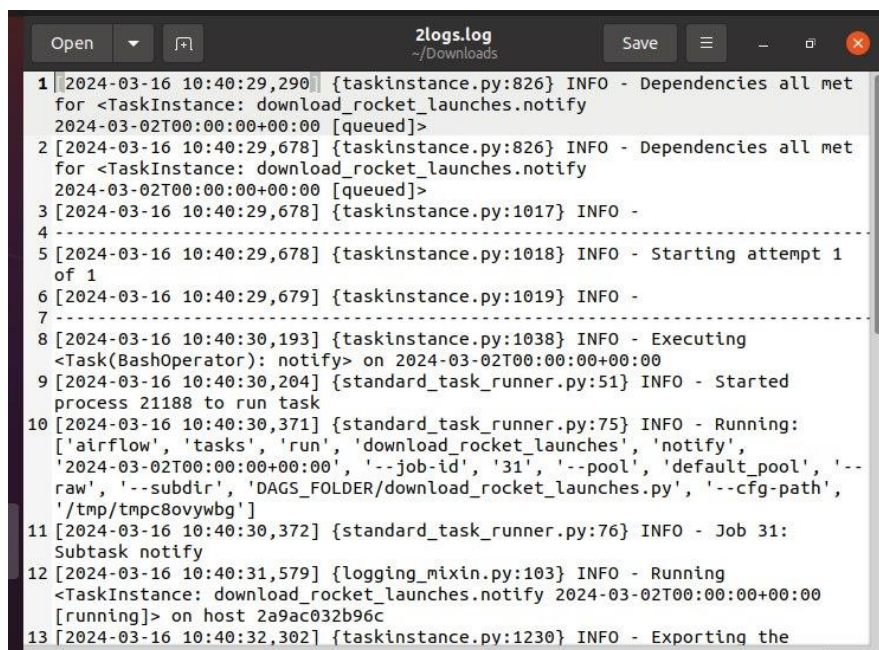


Рисунок 9. Файл 3 «2logs.log».

## 2.6 Выгрузка картинок

Результат выгрузки картинок представлен на рисунках 10-11.

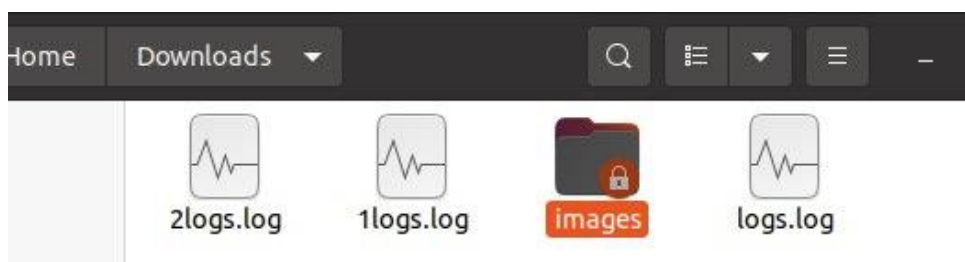


Рисунок 10. Выгруженная папка images.

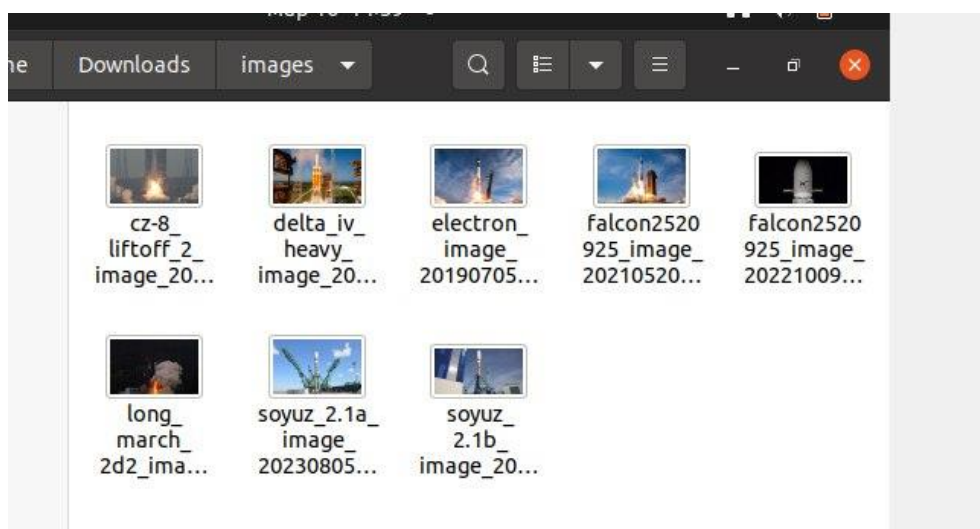


Рисунок 11. Выгруженные картинки.

## 2.7 Диаграмма Ганта DAG в Apache Airflow.

Диаграмма Ганта DAG в Apache Airflow представлена на рисунке

12.

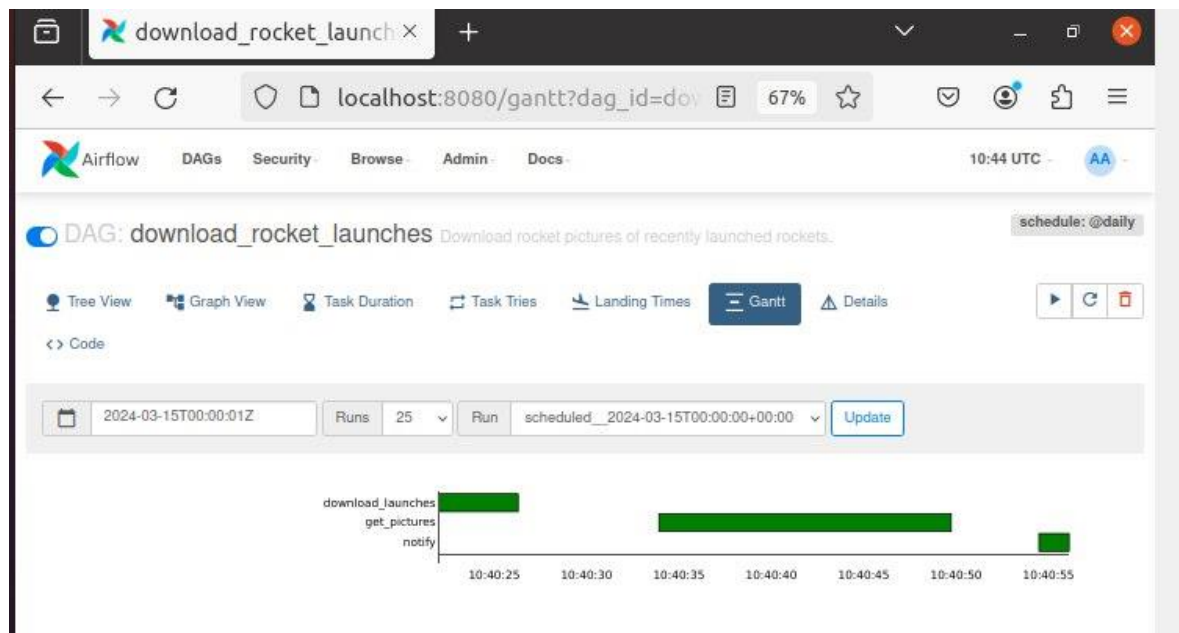


Рисунок 12. Диаграмма Ганта DAG.

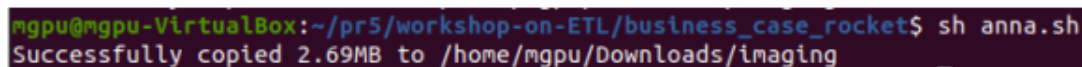
## 2.8 Автоматизация выгрузки данных из контейнера в основную ОС данных

На рисунке 13 представлен скрипт исполняемого файла с расширением .sh, который автоматизирует выгрузку данных из контейнера в основную ОС данных, полученные в результате работы DAG в Apache Airflow.

```
1 #!/bin/bash
2
3 #определить номер контейнера, в котором выполнялся DAG
4 CONTAINER_ID=$(sudo docker ps --filter "name=business_case_rocket-scheduler-1" -q)
5
6 #копировать файлы из каталога /tmp/images в основную ОС
7 sudo docker cp --archive $CONTAINER_ID:/tmp/images /home/mgpu/Downloads/imaging
```

Рисунок 13 – Исполняемый файл

По итогам запуска исполняемого файла `anna.sh` выгрузка данных, полученных в результате работы DAG, из контейнера в основную ОС выполнена успешно, это отражено на рисунке 14.



```
mgpu@mgpu-VirtualBox:~/pr5/workshop-on-ETL/business_case_rocket$ sh anna.sh
Successfully copied 2.69MB to /home/mgpu/Downloads/imaging
```

Рисунок 14 – Выполнение исполняемого файла

## ЗАКЛЮЧЕНИЕ

В рамках выполнения поставленной задачи по разворачиванию и анализу Бизнес-кейса "Rocket" был проведен комплексный процесс, включающий различные этапы работы с виртуальной машиной, Apache Airflow, и проектирование архитектурных решений.

1. Была успешно развернута виртуальная машина `ubuntu_mgpu.ova` в VirtualBox, а затем клонирован задание Бизнес-кейса "Rocket" на ПК для дальнейшей работы.
2. С использованием Apache Airflow был запущен контейнер с кейсом, создан DAG согласно предоставленному алгоритму, и изучены основные элементы DAG. Логи выполненного DAG были изучены и успешно скачаны на основную операционную систему.
3. Создан исполняемый файл `.sh` для автоматизации выгрузки данных из контейнера в основную операционную систему, полученные в результате работы DAG в Apache Airflow.
4. Спроектирована верхнеуровневая архитектура аналитического решения задания "Rocket" с использованием `draw.io`.
5. Архитектура DAG для Бизнес-кейса "Rocket" также была спроектирована в `draw.io`, включая диаграмму Ганта работы DAG в Apache Airflow.
6. Все результаты исследований были подробно описаны и представлены в файле `ФИО-05.pdf`, включая постановку задачи, исходный код всех DAGs, верхнеуровневую архитектуру и

архитектуру DAG для Бизнес-кейса "Rocket", скриншоты лог-файлов и диаграмму Ганта DAG в Apache Airflow.

В результате проделанной работы были достигнуты поставленные цели, и представленная документация содержит все необходимые основания и результаты исследований для успешного завершения проекта по анализу Бизнес-кейса "Rocket".