

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

### **Практическая работа №5**

по дисциплине «Проектный практикум по разработке ETL-решений»

Тема: «Airflow DAG»

Направление подготовки 38.03.05 – бизнес-информатика  
Профиль подготовки «Аналитика данных и эффективное управление»  
(очная форма обучения)

**Выполнила:**

студентка группы АДЭУ-201  
Башкатова Анна Денисовна

**Преподаватель:**

Босенко Т. М., к.т.н., доцент

Москва  
2024

## Содержание

<b>Постановка задачи .....</b>	<b>3</b>
<b>Решение задачи .....</b>	<b>4</b>
<b>Задание 1. Начало работы в VirtualBox.....</b>	<b>4</b>
<b>Задание 2. Изучение задания Бизнес-кейс «Rocket». ....</b>	<b>5</b>
<b>Задание 3. Запуск контейнера с кейсом, работа в Apache Airflow. ....</b>	<b>6</b>
<b>Задание 4. Исполняемый файл. ....</b>	<b>11</b>
<b>Задание 5. Верхнеуровневая архитектура аналитического решения. ....</b>	<b>12</b>
<b>Задание 6. Архитектура DAG Бизнес-кейса «Rocket».....</b>	<b>12</b>
<b>Задание 7. Диаграмма Ганта работы DAG в Apache Airflow. ....</b>	<b>12</b>
<b>Заключение. ....</b>	<b>13</b>

## Постановка задачи

В ходе выполнения лабораторной работы №5 необходимо реализовать следующие задачи:

1. Развернуть ВМ `ubuntu_mgpu.ova` в VirtualBox.
2. Клонировать на ПК задание Бизнес-кейс «Rocket» в домашний каталог ВМ.
3. Запустить контейнер с кейсом, изучить основные элементы DAG в Apache Airflow. Необходимо:
  - создать DAG согласно алгоритму;
  - изучить логи, выполненного DAG и скачать логи из контейнера на основную ОС;
  - выгрузить полученный результат работы DAG в основной каталог ОС.
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`.
7. Построить диаграмму Ганта работы DAG в Apache Airflow.
8. Результаты исследований представить в виде файла `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.

## Решение задачи

### Задание 1. Начало работы в VirtualBox.

Итог развёртывание VM в VirtualBox представлен на рисунке 1.

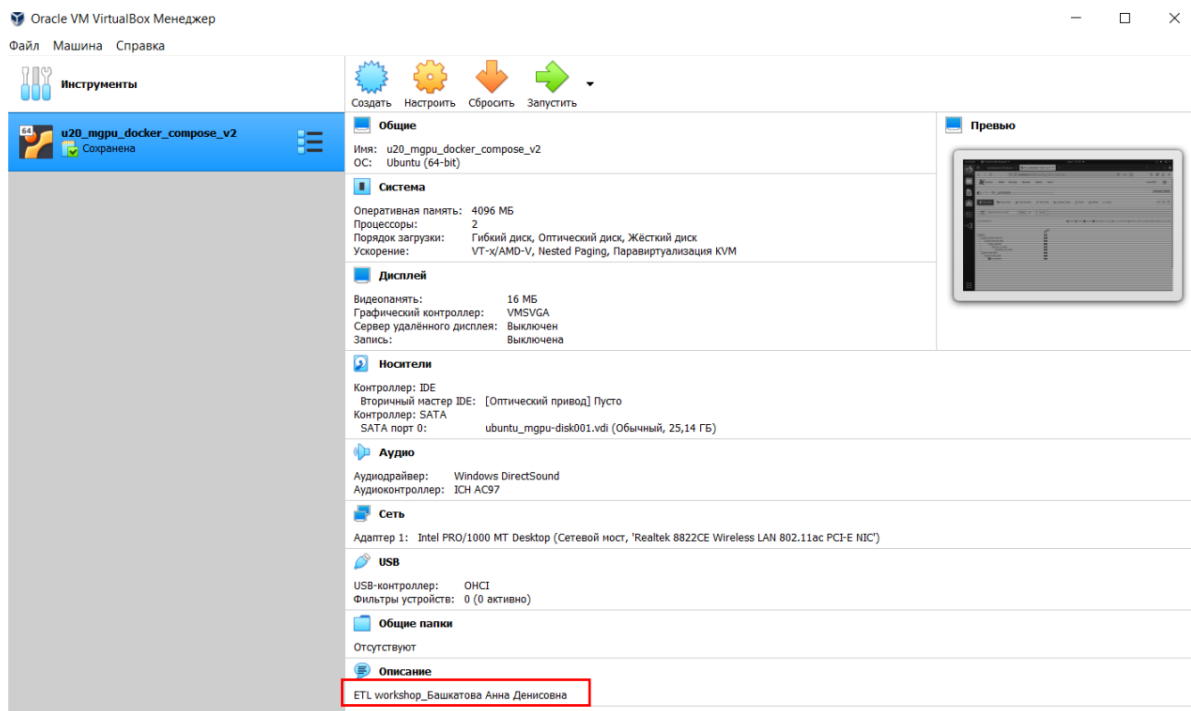


Рисунок 1 – Развёртывание VM

Далее проводится исследование данных – проверка ответа URL-адреса с помощью `curl` из командной строки. Ответ представляет собой документ JSON, где квадратные скобки обозначают список, а все значения в этих фигурных скобках относятся к одному запуску ракеты. Здесь представлена такая информация, как идентификатор ракеты, а также время начала и окончания окна запуска ракеты.

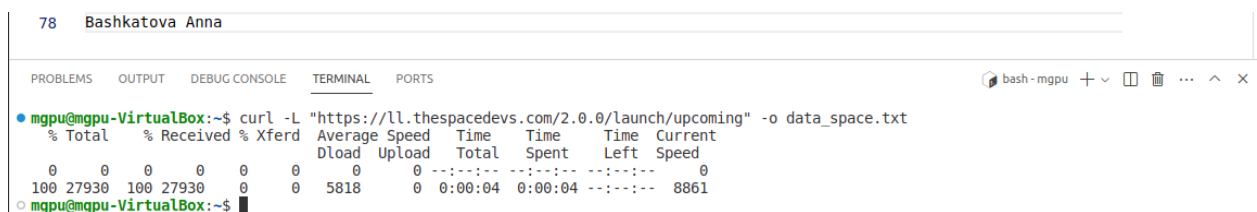


Рисунок 2 – Использование утилиты curl

На рисунке 2 представлено выполнение команды, которая использует утилиту `curl` для загрузки данных с заданного URL-адреса "https://ll.thespacedevs.com/2.0.0/launch/upcoming" с автоматическим

перенаправлением (-L) и сохранение этих данные в файл с именем data\_space.txt, содержимое которого представлено на рисунке 3.

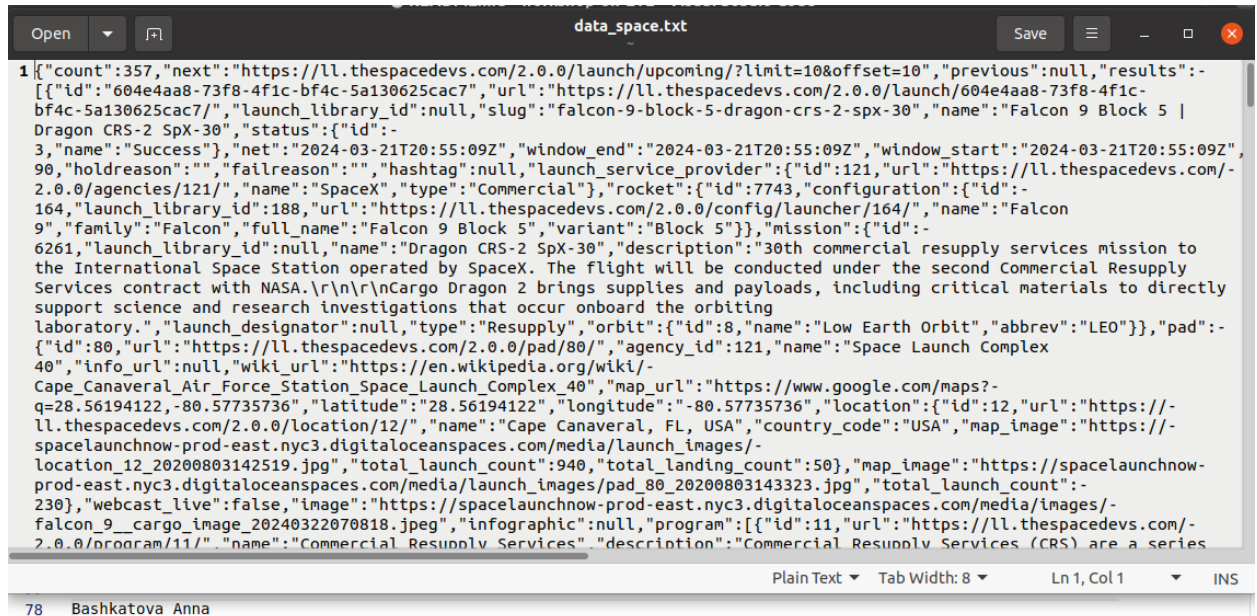


Рисунок 3 – Результат выполнения команды

## Задание 2. Изучение задания Бизнес-кейс «Rocket».

На рисунке 4 показано клонирование репозитория в каталог ВМ и переход в папку business\_case\_rocket.

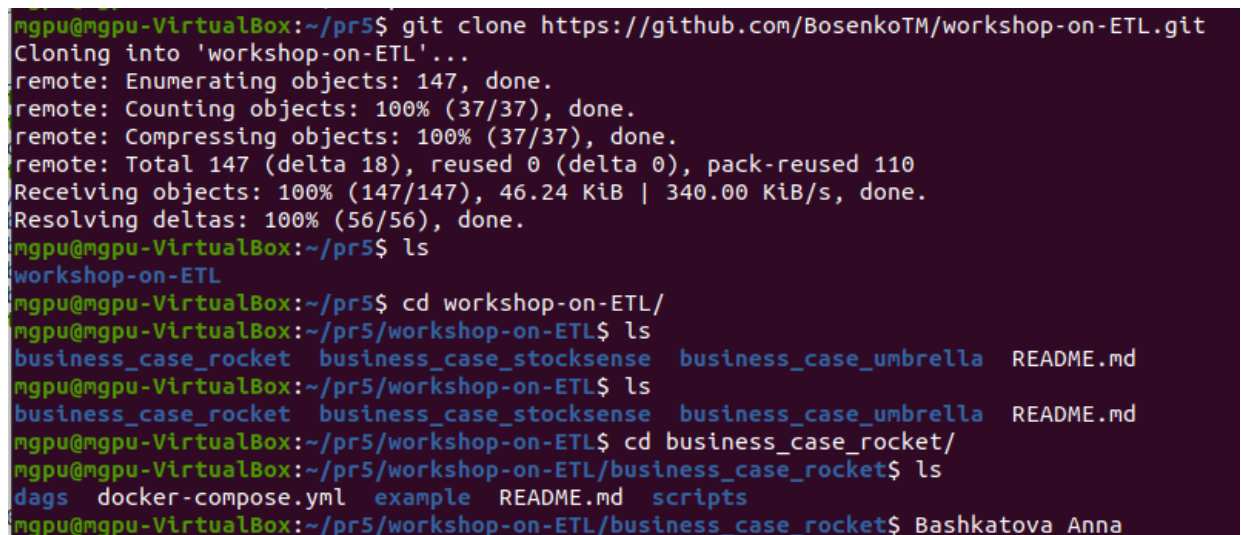


Рисунок 4 – Клонирование репозитория

### Задание 3. Запуск контейнера с кейсом, работа в Apache Airflow.

На рисунках 5 – 6 показан запуск DAG в Airflow и проверка пользовательского интерфейса Airflow, перейдя по ссылке `http://localhost:8080/`.

```
mgpu@mgpu-VirtualBox:~/pr5/workshop-on-ETL/business_case_rocket$ sudo docker compose up -d
[+] Running 4/5
  Network business_case_rocket_default      Created                               2.0s
  Container business_case_rocket-postgres-1 Started                             0.7s
  Container business_case_rocket-webserver-1 Started                           1.7s
  Container business_case_rocket-scheduler-1 Started                           1.6s
  Container business_case_rocket-init-1     Started                             1.8s
```

Рисунок 5 – Запуск контейнера с кейсом

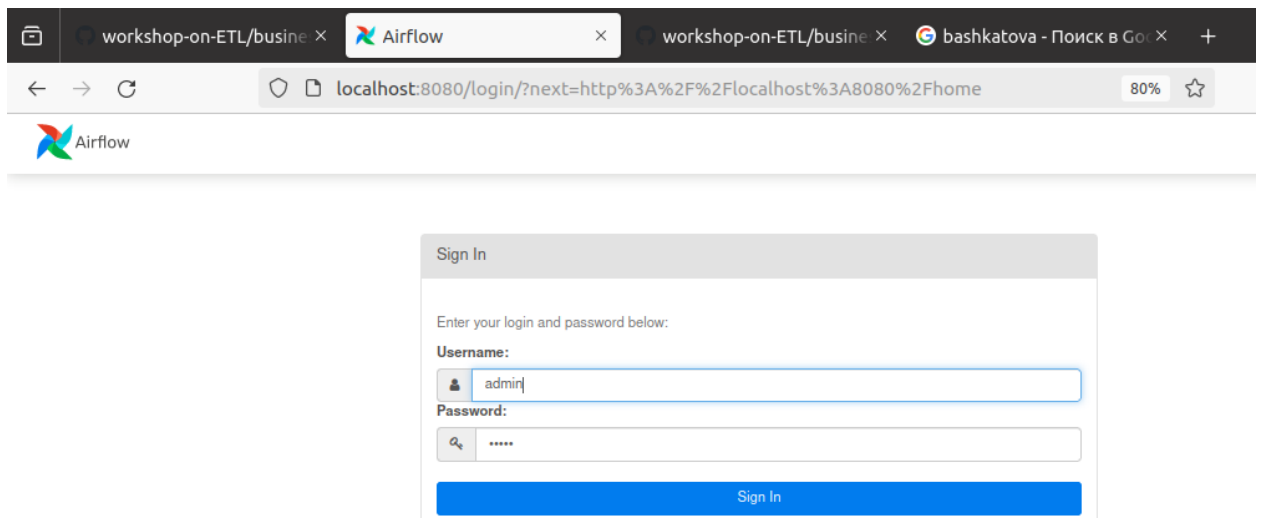


Рисунок 6 – Переход по ссылке

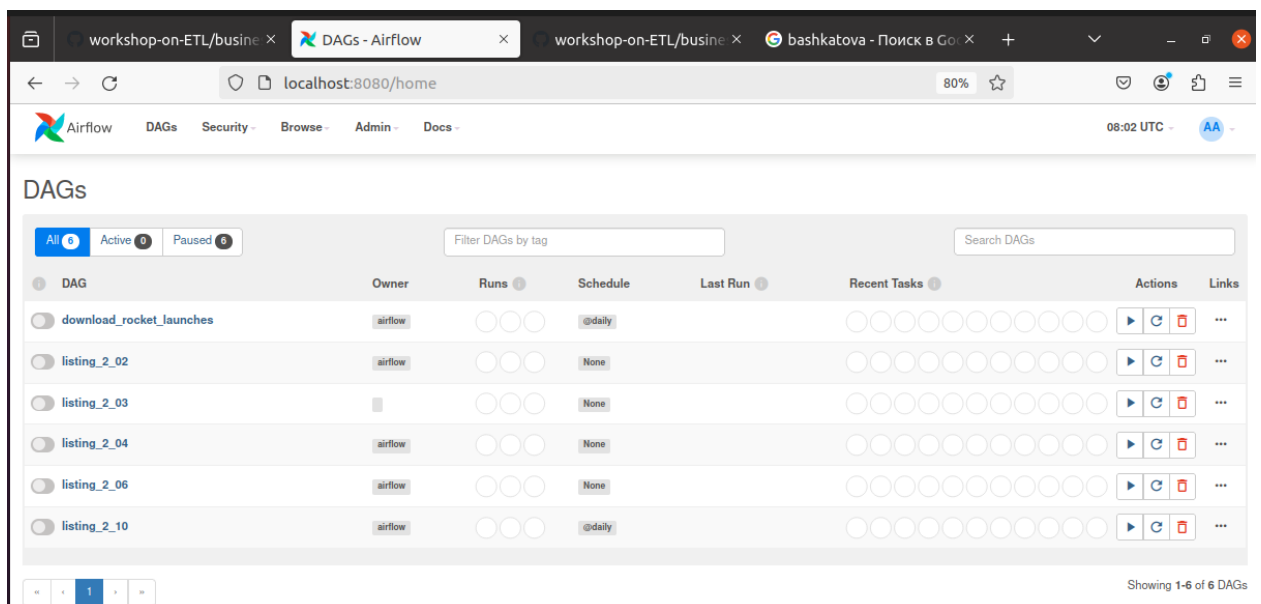


Рисунок 7 – Работа с пользовательским интерфейсом

После того как DAG запущен необходимо посмотреть его граф в Apache Airflow перейдя на вкладку Graph View, он показан на рисунке 8.

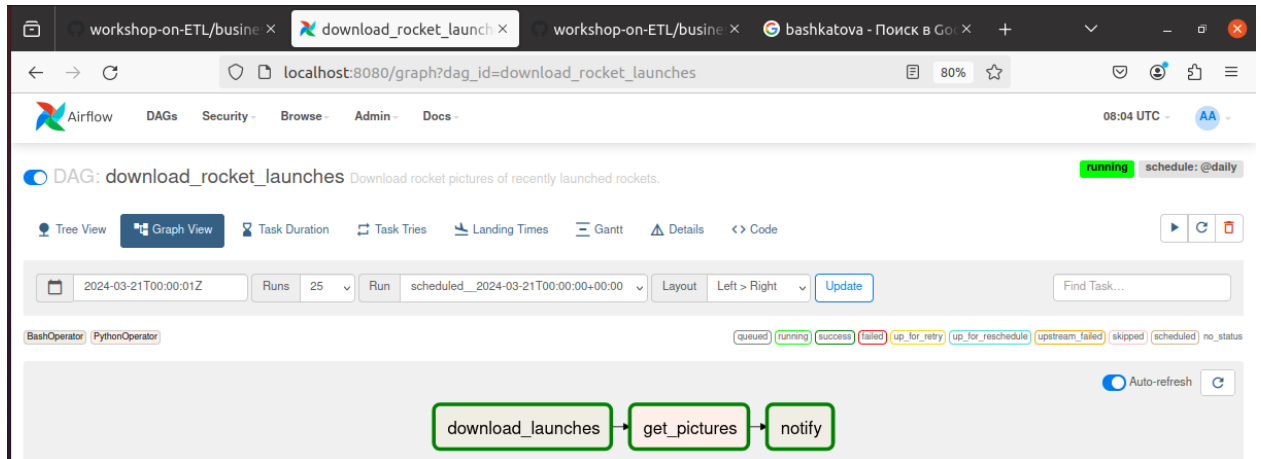


Рисунок 8 – Граф DAG

Дополнительно необходимо посмотреть диаграмму Ганта DAG в Apache Airflow на вкладке Gantt. На рисунке 9 показана история выполнения DAG в виде Диаграммы Ганта.

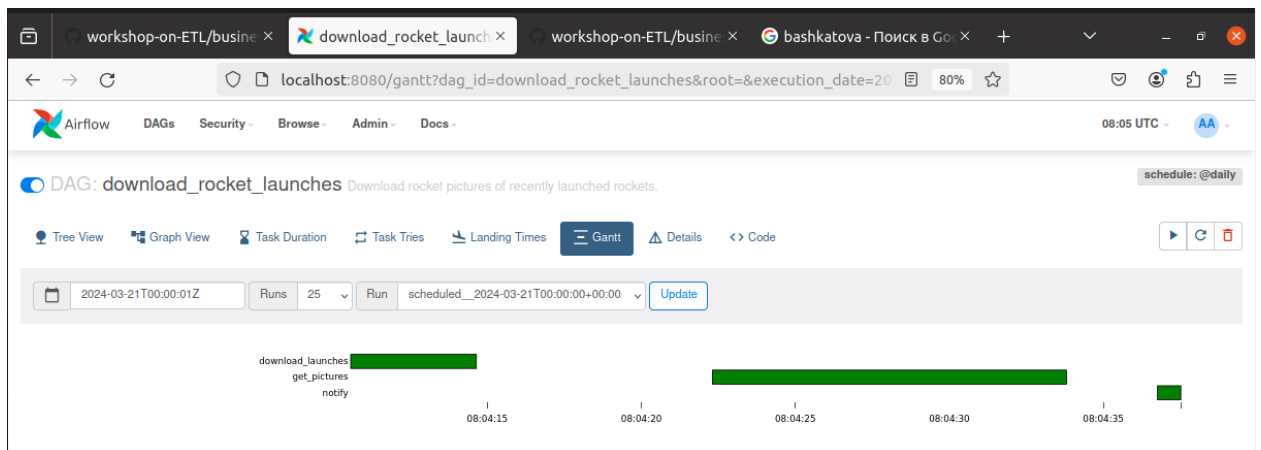
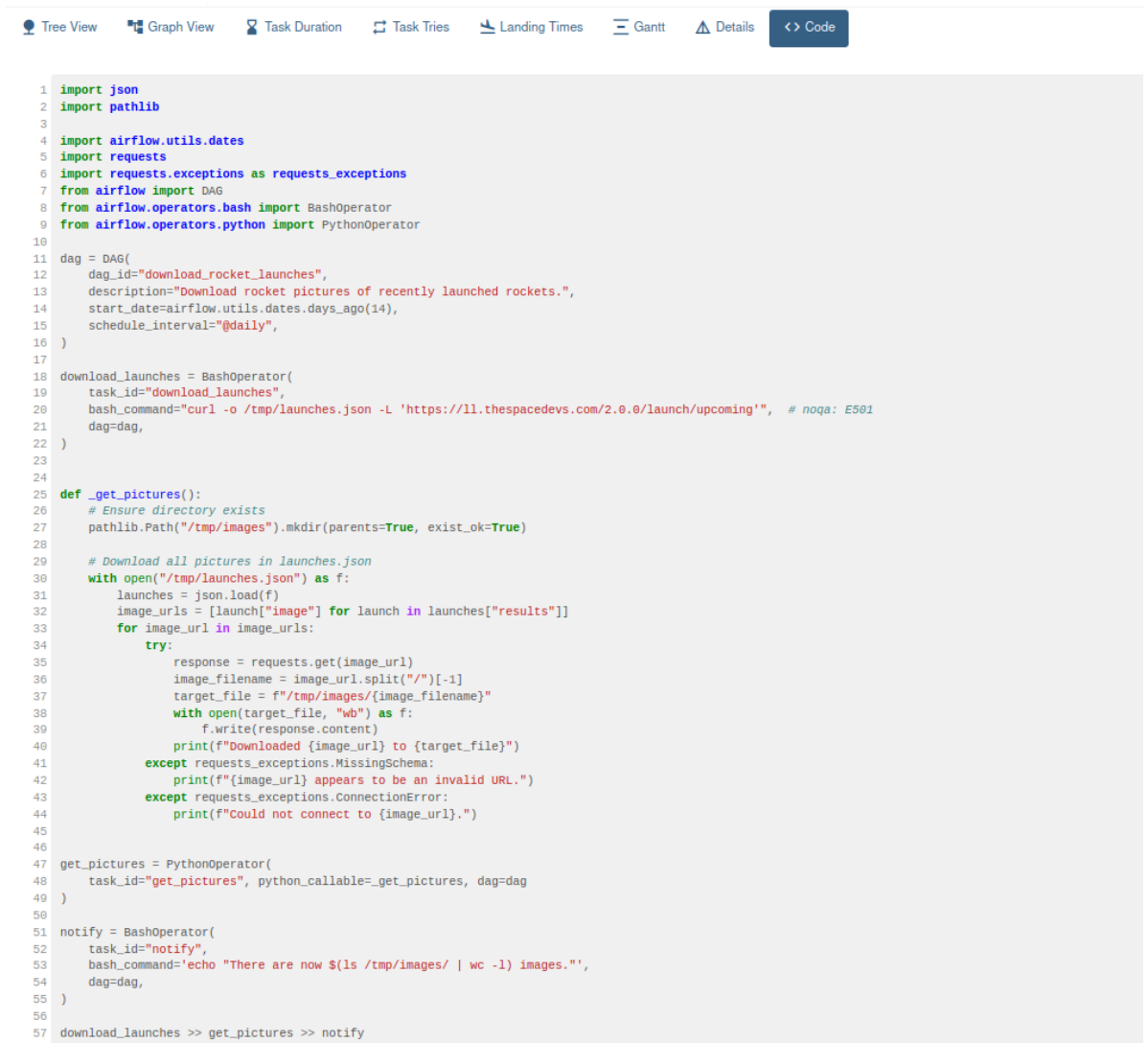


Рисунок 9 – Диаграмму Ганта

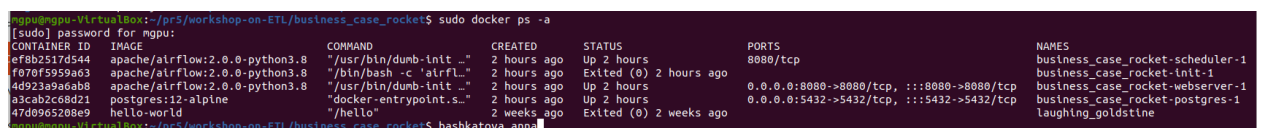
На вкладке Code необходимо просмотреть исходный код DAG, который представлен на рисунке 10.



```
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://11.thespacedevs.com/2.0.0/launch/upcoming'", # noqa: E501
21     dag=dag,
22 )
23
24
25 def _get_pictures():
26     # Ensure directory exists
27     pathlib.Path("/tmp/images").mkdir(parents=True, exist_ok=True)
28
29     # Download all pictures in launches.json
30     with open("/tmp/launches.json") as f:
31         launches = json.load(f)
32         image_urls = [launch["image"] for launch in launches["results"]]
33         for image_url in image_urls:
34             try:
35                 response = requests.get(image_url)
36                 image_filename = image_url.split("/")[-1]
37                 target_file = f"/tmp/images/{image_filename}"
38                 with open(target_file, "wb") as f:
39                     f.write(response.content)
40                 print(f"Downloaded {image_url} to {target_file}")
41             except requests_exceptions.MissingSchema:
42                 print(f"{image_url} appears to be an invalid URL.")
43             except requests_exceptions.ConnectionError:
44                 print(f"Could not connect to {image_url}.")
45
46
47 get_pictures = PythonOperator(
48     task_id="get_pictures", python_callable=_get_pictures, dag=dag
49 )
50
51 notify = BashOperator(
52     task_id="notify",
53     bash_command="echo 'There are now $(ls /tmp/images/ | wc -l) images.'",
54     dag=dag,
55 )
56
57 download_launches >> get_pictures >> notify
```

Рисунок 10 – Код DAG

На следующем шаге изучаются логи, выполненного DAG. Для этого необходимо скачать логи из контейнера на основную ОС. На рисунке 11 представлено определение номера контейнера, в котором выполнен DAG. Таким образом, хеш контейнера ef8b2517d544, а имя контейнера business\_case\_rocket-scheduler-1.



```
mpu@mpu-VirtualBox:~/pr5/workshop-on-ETL/business_case_rocket$ sudo docker ps -a
[sudo] password for mpu:
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS                    NAMES
ef8b2517d544   apache/airflow:2.0.0-python3.8     "/usr/bin/dumb-init _"  2 hours ago   Up 2 hours   8080/tcp                 business_case_rocket-scheduler-1
f070f5959a63   apache/airflow:2.0.0-python3.8     "/bin/bash -c 'airfl"  2 hours ago   Exited (0)   2 hours ago             business_case_rocket-init-1
44923a9a0a88   apache/airflow:2.0.0-python3.8     "/usr/bin/dumb-init _"  2 hours ago   Up 2 hours   8080/tcp                 business_case_rocket-webserver-1
a3cab2c8dd21   postgres:12-alpine                 "docker-entrypoint.s"   2 hours ago   Up 2 hours   5432/tcp                 business_case_rocket-postgres-1
47d8965208e9   hello-world                          "/hello"                 2 weeks ago   Exited (0)   2 weeks ago             laughing_goldstine
```

Рисунок 11 – Определение контейнера



Далее, как показано на рисунке 12, производится проверка наличия логов в контейнере, предварительно войдя в контейнер.

```
mgpu@mgpu-VirtualBox:~/prj/workshop-on-ETL/business_case_rocket$ sudo docker exec -it business_case_rocket-scheduler-1 /bin/bash
airflow@ef8b2517d544:/opt/airflow$ ls
airflow.cfg dags logs unittests.cfg webserver_config.py
airflow@ef8b2517d544:/opt/airflow$ cd logs
airflow@ef8b2517d544:/opt/airflow/logs$ ls
dag_processor_manager download_rocket_launches scheduler
airflow@ef8b2517d544:/opt/airflow/logs$ cd download_rocket_launches
airflow@ef8b2517d544:/opt/airflow/logs/download_rocket_launches$ ls
download_launches get_pictures notify
airflow@ef8b2517d544:/opt/airflow/logs/download_rocket_launches$ cd notify
airflow@ef8b2517d544:/opt/airflow/logs/download_rocket_launches/notify$ ls
2024-03-05T00:00:00+00:00 2024-03-08T00:00:00+00:00 2024-03-11T00:00:00+00:00 2024-03-14T00:00:00+00:00 2024-03-17T00:00:00+00:00 2024-03-20T00:00:00+00:00
2024-03-06T00:00:00+00:00 2024-03-09T00:00:00+00:00 2024-03-12T00:00:00+00:00 2024-03-15T00:00:00+00:00 2024-03-18T00:00:00+00:00 2024-03-21T00:00:00+00:00
2024-03-07T00:00:00+00:00 2024-03-10T00:00:00+00:00 2024-03-13T00:00:00+00:00 2024-03-16T00:00:00+00:00 2024-03-19T00:00:00+00:00
airflow@ef8b2517d544:/opt/airflow/logs/download_rocket_launches/notify$ cd 2024-03-21T00:00:00+00:00
airflow@ef8b2517d544:/opt/airflow/logs/download_rocket_launches/notify/2024-03-21T00:00:00+00:00$ ls
1.log
airflow@ef8b2517d544:/opt/airflow/logs/download_rocket_launches/notify/2024-03-21T00:00:00+00:00$ bashkatova anna
```

Рисунок 12 – Проверка контейнера

Далее происходит выгрузка логов из контейнера в основную ОС (рисунок 13).

```
mgpu@mgpu-VirtualBox:~$ sudo docker cp ef8:/opt/airflow/logs/download_rocket_launches/notify/2024-03-21T00:00:00+00:00/1.log /home/mgpu/Downloads/logs_anna.log
Successfully copied 7.17kB to /home/mgpu/Downloads/logs_anna.log
```

Рисунок 13 – Выгрузка логов

После выгрузки проверяются в каталоге основной ОС файлы логов, которые были сохранены. Как показано на рисунках 14 – 16, файлы логов успешно сохранены на основной ОС.

```
home > mgpu > Downloads > logs > notify_latest.log
1 [2024-03-22 15:01:04,606] {taskinstance.py:826} INFO - Dependencies all met for <TaskInstance: download_rocket_launches.notify 2024-03-22T15:00:20.658537+00:00 [queued]>
2 [2024-03-22 15:01:04,641] {taskinstance.py:826} INFO - Dependencies all met for <TaskInstance: download_rocket_launches.notify 2024-03-22T15:00:20.658537+00:00 [queued]>
3 [2024-03-22 15:01:04,641] {taskinstance.py:1017} INFO -
4 .....
5 [2024-03-22 15:01:04,641] {taskinstance.py:1018} INFO - Starting attempt 1 of 1
6 [2024-03-22 15:01:04,641] {taskinstance.py:1019} INFO -
7 .....
8 [2024-03-22 15:01:04,658] {taskinstance.py:1038} INFO - Executing <Task(BashOperator): notify> on 2024-03-22T15:00:20.658537+00:00
9 [2024-03-22 15:01:04,661] {standard_task_runner.py:51} INFO - Started process 972338 to run task
10 [2024-03-22 15:01:04,677] {standard_task_runner.py:75} INFO - Running: ['airflow', 'tasks', 'run', 'download_rocket_launches', 'notify', '2024-03-22T15:00:20.658537+00:00', '--job-id', '46']
11 [2024-03-22 15:01:04,678] {logging_mixin.py:103} INFO - Job 46: Subtask notify
12 [2024-03-22 15:01:04,744] {logging_mixin.py:103} INFO - Running <TaskInstance: download_rocket_launches.notify 2024-03-22T15:00:20.658537+00:00 [running]> on host ef8b2517d544
13 [2024-03-22 15:01:04,866] {taskinstance.py:1230} INFO - Exporting the following env vars:
14 AIRFLOW_CTX_DAG_OWNER=airflow
15 AIRFLOW_CTX_DAG_ID=download_rocket_launches
16 AIRFLOW_CTX_TASK_ID=notify
17 AIRFLOW_CTX_EXECUTION_DATE=2024-03-22T15:00:20.658537+00:00
18 AIRFLOW_CTX_DAG_RUN_ID=manual_2024-03-22T15:00:20.658537+00:00
19 [2024-03-22 15:01:04,873] {bash.py:135} INFO - Tmp dir root location:
20 /tmp
21 [2024-03-22 15:01:04,875] {bash.py:158} INFO - Running command: echo "There are now $(ls /tmp/images/ | wc -l) images."
22 [2024-03-22 15:01:04,915] {bash.py:169} INFO - Output:
23 [2024-03-22 15:01:05,021] {bash.py:173} INFO - There are now 7 images.
24 [2024-03-22 15:01:05,021] {bash.py:177} INFO - Command exited with return code 0
25 [2024-03-22 15:01:05,129] {taskinstance.py:1135} INFO - Marking task as SUCCESS. dag_id=download_rocket_launches, task_id=notify, execution_date=20240322T150020, start_date=20240322T150104,
26 [2024-03-22 15:01:05,268] {taskinstance.py:1195} INFO - 0 downstream tasks scheduled from follow-on schedule check
27 [2024-03-22 15:01:05,312] {local_task_job.py:118} INFO - Task exited with return code 0
28 Bashkatova
```

Рисунок 14 – Файл notify

```
home > mguu > Downloads > logs > download_latest.log
1 [2024-03-22 15:00:24,662] {taskinstance.py:826} INFO - Dependencies all met for <TaskInstance: download_rocket_launches.download_launches 2024-03-22T15:00:20.658537+00:00 [queued]>
2 [2024-03-22 15:00:24,746] {taskinstance.py:826} INFO - Dependencies all met for <TaskInstance: download_rocket_launches.download_launches 2024-03-22T15:00:20.658537+00:00 [queued]>
3 [2024-03-22 15:00:24,746] {taskinstance.py:1017} INFO -
4 .....
5 [2024-03-22 15:00:24,747] {taskinstance.py:1018} INFO - Starting attempt 1 of 1
6 [2024-03-22 15:00:24,747] {taskinstance.py:1019} INFO -
7 .....
8 [2024-03-22 15:00:24,819] {taskinstance.py:1038} INFO - Executing <Task(BashOperator): download_launches> on 2024-03-22T15:00:20.658537+00:00
9 [2024-03-22 15:00:24,839] {standard_task_runner.py:51} INFO - Started process 970491 to run task
10 [2024-03-22 15:00:24,850] {standard_task_runner.py:75} INFO - Running: ['airflow', 'tasks', 'run', 'download_rocket_launches', 'download_launches', '2024-03-22T15:00:20.658537+00:00', '--job-id']
11 [2024-03-22 15:00:24,871] {standard_task_runner.py:76} INFO - Job 44: Subtask download_launches
12 [2024-03-22 15:00:25,099] {logging_mixin.py:103} INFO - Running <TaskInstance: download_rocket_launches.download_launches 2024-03-22T15:00:20.658537+00:00 [running]> on host ef8b2517d544
13 [2024-03-22 15:00:25,371] {taskinstance.py:1230} INFO - Exporting the following env vars:
14 AIRFLOW_CTX_DAG_OWNER=airflow
15 AIRFLOW_CTX_DAG_ID=download_rocket_launches
16 AIRFLOW_CTX_TASK_ID=download_launches
17 AIRFLOW_CTX_EXECUTION_DATE=2024-03-22T15:00:20.658537+00:00
18 AIRFLOW_CTX_DAG_RUN_ID=manual_2024-03-22T15:00:20.658537+00:00
19 [2024-03-22 15:00:25,379] {bash.py:135} INFO - Tmp dir root location:
20 /tmp
21 [2024-03-22 15:00:25,384] {bash.py:158} INFO - Running command: curl -o /tmp/launches.json -L 'https://ll.thespacedevs.com/2.0.0/launch/upcoming'
22 [2024-03-22 15:00:25,404] {bash.py:169} INFO - Output:
23 [2024-03-22 15:00:25,667] {bash.py:173} INFO - % Total % Received % Xferd Average Speed Time Time Current
24 [2024-03-22 15:00:25,667] {bash.py:173} INFO - % Total % Received % Xferd Average Speed Time Time Current
25 [2024-03-22 15:00:38,715] {bash.py:173} INFO -
26 0 0 0 0 0 0 0 0 0 0:00:00 0:00:00 0 0
27 0 0 0 0 0 0 0 0 0 0:00:00 0:00:00 0 0
28 0 0 0 0 0 0 0 0 0 0:00:01 0:00:00 0 0
29 0 0 0 0 0 0 0 0 0 0:00:02 0:00:00 0 0
30 0 0 0 0 0 0 0 0 0 0:00:03 0:00:00 0 0
31 0 0 0 0 0 0 0 0 0 0:00:04 0:00:00 0 0
32 0 0 0 0 0 0 0 0 0 0:00:05 0:00:00 0 0
33 0 0 0 0 0 0 0 0 0 0:00:06 0:00:00 0 0
34 0 0 0 0 0 0 0 0 0 0:00:07 0:00:00 0 0
35 0 0 0 0 0 0 0 0 0 0:00:08 0:00:00 0 0
36 0 0 0 0 0 0 0 0 0 0:00:09 0:00:00 0 0
37 0 0 0 0 0 0 0 0 0 0:00:10 0:00:00 0 0
38 0 0 0 0 0 0 0 0 0 0:00:11 0:00:00 0 0
39 0 0 0 0 0 0 0 0 0 0:00:12 0:00:00 0 0
40 0 0 0 0 0 0 0 0 0 0:00:13 0:00:00 0 0
41 [2024-03-22 15:00:41,947] {bash.py:173} INFO -
42 0 0 0 0 0 0 0 0 0 0:00:14 0:00:00 0 0
43 0 0 0 0 0 0 0 0 0 0:00:15 0:00:00 0 0
44 0 0 0 0 0 0 0 0 0 0:00:16 0:00:00 0 0
45 39 27930 39 10921 0 0 675 0:00:41 0:00:16 0:00:25 5178
46 100 27930 100 27930 0 0 1715 0:00:16 0:00:16 0:00:16 12524
47 [2024-03-22 15:00:41,953] {bash.py:177} INFO - Command exited with return code 0
48 [2024-03-22 15:00:42,037] {taskinstance.py:1135} INFO - Marking task as SUCCESS; dag_id=download_rocket_launches, task_id=download_launches, execution_date=20240322T150020, start_date=20240322T150020
```

Рисунок 15 – Файл download

```
home > mguu > Downloads > logs > pictures_latest.log
1 [2024-03-22 15:00:43,583] {taskinstance.py:826} INFO - Dependencies all met for <TaskInstance: download_rocket_launches.get_pictures 2024-03-22T15:00:20.658537+00:00 [queued]>
2 [2024-03-22 15:00:43,614] {taskinstance.py:826} INFO - Dependencies all met for <TaskInstance: download_rocket_launches.get_pictures 2024-03-22T15:00:20.658537+00:00 [queued]>
3 [2024-03-22 15:00:43,615] {taskinstance.py:1017} INFO -
4 .....
5 [2024-03-22 15:00:43,615] {taskinstance.py:1018} INFO - Starting attempt 1 of 1
6 [2024-03-22 15:00:43,615] {taskinstance.py:1019} INFO -
7 .....
8 [2024-03-22 15:00:43,664] {taskinstance.py:1038} INFO - Executing <Task(PythonOperator): get_pictures> on 2024-03-22T15:00:20.658537+00:00
9 [2024-03-22 15:00:43,667] {standard_task_runner.py:51} INFO - Started process 971413 to run task
10 [2024-03-22 15:00:43,697] {standard_task_runner.py:75} INFO - Running: ['airflow', 'tasks', 'run', 'download_rocket_launches', 'get_pictures', '2024-03-22T15:00:20.658537+00:00', '--job-id']
11 [2024-03-22 15:00:43,700] {standard_task_runner.py:76} INFO - Job 45: Subtask get_pictures
12 [2024-03-22 15:00:44,003] {logging_mixin.py:103} INFO - Running <TaskInstance: download_rocket_launches.get_pictures 2024-03-22T15:00:20.658537+00:00 [running]> on host ef8b2517d544
13 [2024-03-22 15:00:44,280] {taskinstance.py:1230} INFO - Exporting the following env vars:
14 AIRFLOW_CTX_DAG_OWNER=airflow
15 AIRFLOW_CTX_DAG_ID=download_rocket_launches
16 AIRFLOW_CTX_TASK_ID=get_pictures
17 AIRFLOW_CTX_EXECUTION_DATE=2024-03-22T15:00:20.658537+00:00
18 AIRFLOW_CTX_DAG_RUN_ID=manual_2024-03-22T15:00:20.658537+00:00
19 [2024-03-22 15:00:47,075] {logging_mixin.py:103} INFO - Downloaded https://spacelaunchnow-prod-east.nyc3.digitaloceanspaces.com/media/images/falcon_9_cargo_image_20240322070818.jpeg to /tmp/
20 [2024-03-22 15:00:50,813] {logging_mixin.py:103} INFO - Downloaded https://spacelaunchnow-prod-east.nyc3.digitaloceanspaces.com/media/images/falcon2520925_image_20221009234147.png to /tmp/
21 [2024-03-22 15:00:53,267] {logging_mixin.py:103} INFO - Downloaded https://spacelaunchnow-prod-east.nyc3.digitaloceanspaces.com/media/images/soyuz_2.1a_on_3_image_20240320172211.jpg to /tmp/
22 [2024-03-22 15:00:56,987] {logging_mixin.py:103} INFO - Downloaded https://spacelaunchnow-prod-east.nyc3.digitaloceanspaces.com/media/images/falcon2520925_image_20221009234147.png to /tmp/
23 [2024-03-22 15:00:57,904] {logging_mixin.py:103} INFO - Downloaded https://spacelaunchnow-prod-east.nyc3.digitaloceanspaces.com/media/images/falcon2520925_image_20221009234147.png to /tmp/
24 [2024-03-22 15:00:58,977] {logging_mixin.py:103} INFO - Downloaded https://spacelaunchnow-prod-east.nyc3.digitaloceanspaces.com/media/images/delta_iv_heavy_image_20210426103838.jpg to /tmp/
25 [2024-03-22 15:00:59,795] {logging_mixin.py:103} INFO - Downloaded https://spacelaunchnow-prod-east.nyc3.digitaloceanspaces.com/media/images/soyuz_2.1b_image_20230802085331.jpg to /tmp/
26 [2024-03-22 15:01:00,808] {logging_mixin.py:103} INFO - Downloaded https://spacelaunchnow-prod-east.nyc3.digitaloceanspaces.com/media/images/falcon_9_image_20230807133459.jpeg to /tmp/
27 [2024-03-22 15:01:01,692] {logging_mixin.py:103} INFO - Downloaded https://spacelaunchnow-prod-east.nyc3.digitaloceanspaces.com/media/images/gaganyaan_abort_image_20231021132156.jpeg to /tmp/
28 [2024-03-22 15:01:02,635] {logging_mixin.py:103} INFO - Downloaded https://spacelaunchnow-prod-east.nyc3.digitaloceanspaces.com/media/images/falcon2520925_image_20221009234147.png to /tmp/
29 [2024-03-22 15:01:02,945] {python.py:118} INFO - Returned value was: None
30 [2024-03-22 15:01:02,705] {taskinstance.py:1135} INFO - Marking task as SUCCESS; dag_id=download_rocket_launches, task_id=get_pictures, execution_date=20240322T150020, start_date=20240322T150020
31 [2024-03-22 15:01:02,891] {taskinstance.py:1195} INFO - 1 downstream tasks scheduled from follow-on schedule check
32 [2024-03-22 15:01:02,932] {local_task_job.py:118} INFO - Task exited with return code 0
```

Рисунок 16 – Файл pictures

Также необходимо выгрузить полученный результат работы DAG в основной каталог ОС. Для начала проверяем наличие результатов в контейнере, предварительно войти в контейнер как показано на рисунке 17.

```
mguu@mguu-VirtualBox:~$ sudo docker exec -it business_case_rocket-scheduler-1 /bin/bash
airflow@ef8b2517d544:/opt/airflow$ cd /tmp/images
airflow@ef8b2517d544:/tmp/images$ ls
delta_iv_heavy_image_20210426103838.jpg falcon_9_cargo_image_20240322070818.jpeg gaganyaan_abort_image_20231021132156.jpeg soyuz_2.1b_image_20230802085331.jpg
falcon2520925_image_20221009234147.png falcon_9_image_20230807133459.jpeg soyuz_2.1a_on_3_image_20240320172211.jpg
airflow@ef8b2517d544:/tmp/images$ bashkatova anna
```

Рисунок 17 – Проверка в контейнере

Далее необходимо скопировать все файлы в основную ОС (рисунок 18).

```
mgpu@mgpu-VirtualBox:~$ sudo docker cp ef8:/tmp/images /home/mgpu/Downloads/images  
Successfully copied 1.68MB to /home/mgpu/Downloads/images  
mgpu@mgpu-VirtualBox:~$ bashkatova anna
```

Рисунок 18 – Копирование файлов

После этого необходимо проверить их в каталоге основной ОС. Как видно из рисунка 19, файлы успешно скопированы.

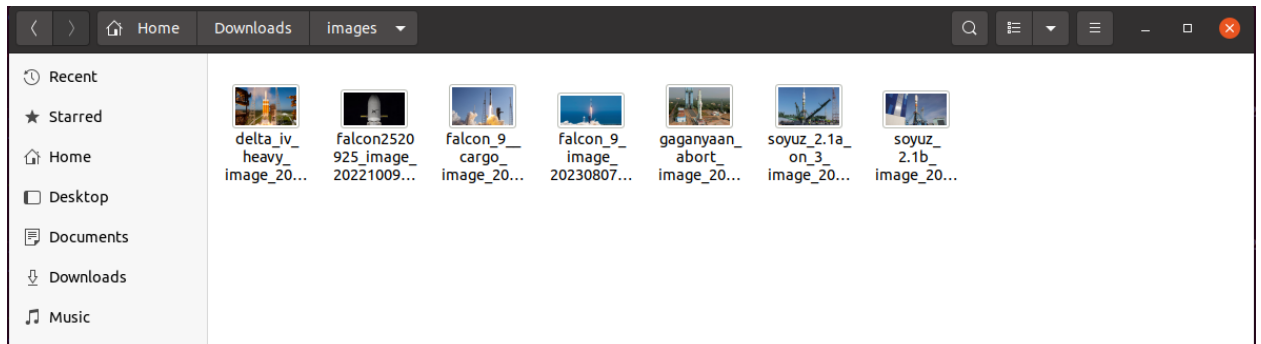


Рисунок 19 – Скопированные файлы

#### Задание 4. Исполняемый файл.

На рисунке 20 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.

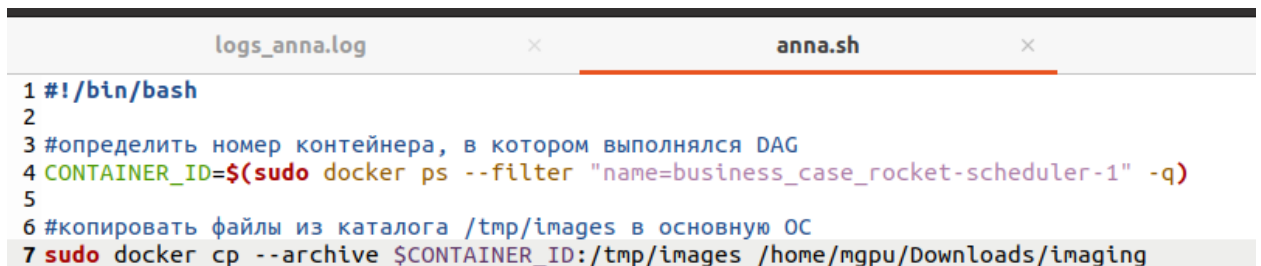


Рисунок 20 – Исполняемый файл

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 21.

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

Рисунок 21 – Выполнение исполняемого файла

### Задание 5. Верхнеуровневая архитектура аналитического решения.

Верхнеуровневая архитектура задания Бизнес-кейса «Rocket» представлена на рисунке 22.

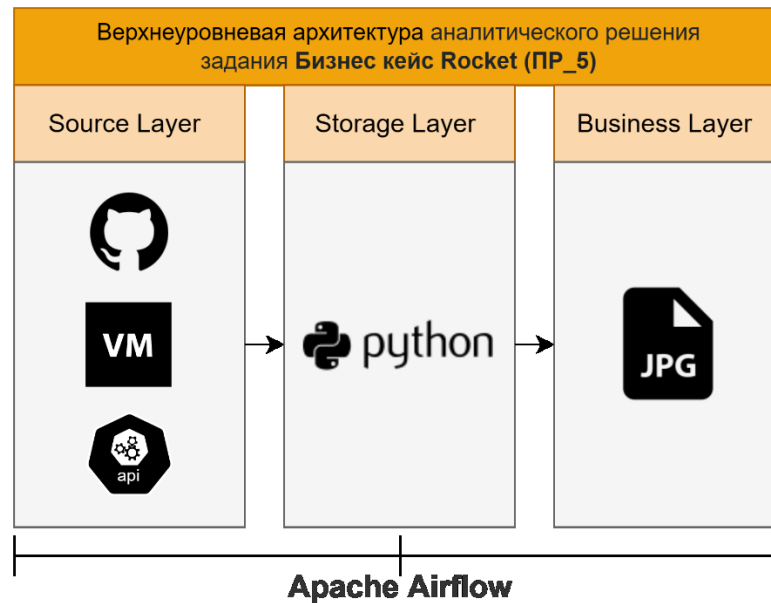


Рисунок 22 – Верхнеуровневая архитектура

### Задание 6. Архитектура DAG Бизнес-кейса «Rocket».

Архитектуру DAG Бизнес-кейса «Rocket» представлена на рисунке 23.

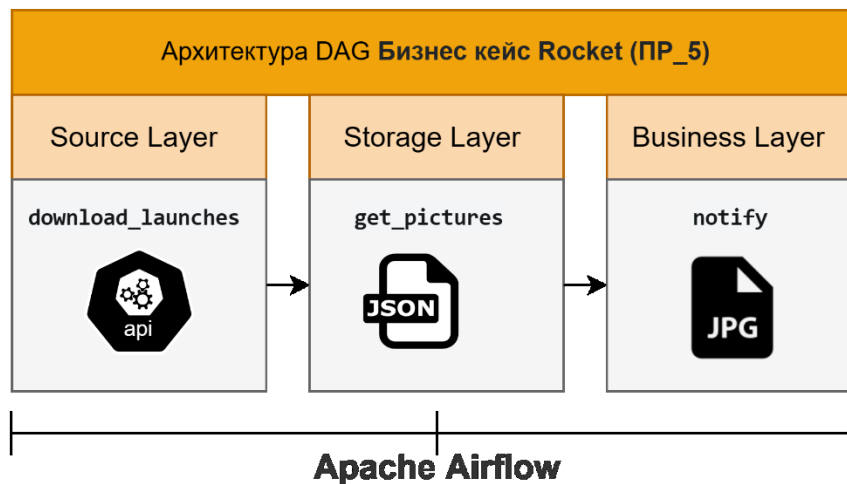


Рисунок 23 – Архитектура DAG

### Задание 7. Диаграмма Ганта работы DAG в Apache Airflow.

Диаграмма Ганта работы DAG представлена на рисунке 9.

### **Заключение.**

По итогам выполнения лабораторной работы №5 было выполнено развёртывание ВМ в VirtualBox, клонировано задание "Rocket" из репозитория, запущен контейнер с кейсом в Apache Airflow, создан и изучен DAG, создан исполняемый файл для автоматизации выгрузки данных, спроектированы архитектуры в draw.io для задания и DAG, построена диаграмма Ганта работы DAG, и подготовлен отчёт. К данному отчету прилагаются файлы логов, а также исполняемый файл.