

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

ОТЧЕТ

по дисциплине «Проектный практикум по разработке ETL-решений»
Вебинар: Бизнес кейс «Umbrella»
Направление подготовки – 38.03.05 «Бизнес-информатика».
профиль подготовки – «Аналитика данных и эффективное управление»

Выполнила:

студентка группы АДЭУ-211
st92

Руководитель:

Кандидат технических наук, доцент

Москва
2025 год

Цель работы: Научиться прогнозировать продажи по бизнес кейсу

Задачи:

Развернуть ВМ и запустить контейнер с кейсом

Спроектировать верхнеуровневую архитектуру аналитического решения задания Бизнес кейс Umbrella в draw.io

Выполнить индивидуальное задание.

Проверить корректность предсказания продаж по обученной модели.

Ход работы:

1. Клонировем на ПК задание Бизнес кейс Umbrella в домашний каталог ВМ.

```
mgpu@mgpu-VirtualBox:~$ git clone https://github.com/BosenkoTM/workshop-on-ETL.git
Cloning into 'workshop-on-ETL'...
remote: Enumerating objects: 502, done.
remote: Counting objects: 100% (392/392), done.
remote: Compressing objects: 100% (337/337), done.
remote: Total 502 (delta 184), reused 59 (delta 32), pack-reused 110 (from 1)
Receiving objects: 100% (502/502), 5.77 MiB | 5.75 MiB/s, done.
Resolving deltas: 100% (222/222), done.
```

Рисунок 1 – Выполнение команды для клонирования

2. Запустим контейнер с кейсом:

```
mgpu@mgpu-VirtualBox:~/workshop-on-ETL/business_case_umbrella_25$ sudo docker compose down -v -
-remove-orphans
```

Рисунок 2 – Выполнение команды для выключения контейнеров

```
mgpu@mgpu-VirtualBox:~/workshop-on-ETL/business_case_umbrella_25$ sudo docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS        NAMES
```

Рисунок 3 – Выполнение команды для просмотра контейнеров

```
dev@dev-vm:~/Downloads/business_case_umbrella_25$ sudo docker compose up
[+] Running 4/4
 ✓ Container business_case_umbrella_25-postgres-1   Created          0.0s
 ✓ Container business_case_umbrella_25-init-1       Created          0.0s
 ✓ Container business_case_umbrella_25-scheduler-1  Created          0.0s
 ✓ Container business_case_umbrella_25-webserver-1  Created          0.0s
```

Рисунок 4 – Выполнение команды для запуска контейнера

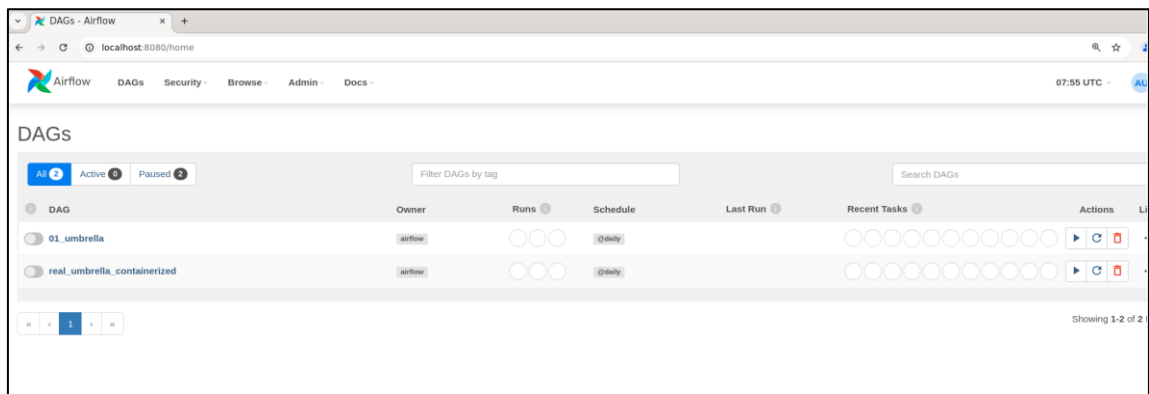


Рисунок 5 – Страница Airflow

Опишем интерфейс:

Airflow DAG — это ориентированный ациклический граф, т.е. граф, у которого отсутствуют циклы, но могут быть параллельные пути, выходящие из одного и того же узла. Простыми словами DAG это сущность, объединяющая ваши задачи в единый data pipeline (или цепочку задач), где явно видны зависимости между узлами.

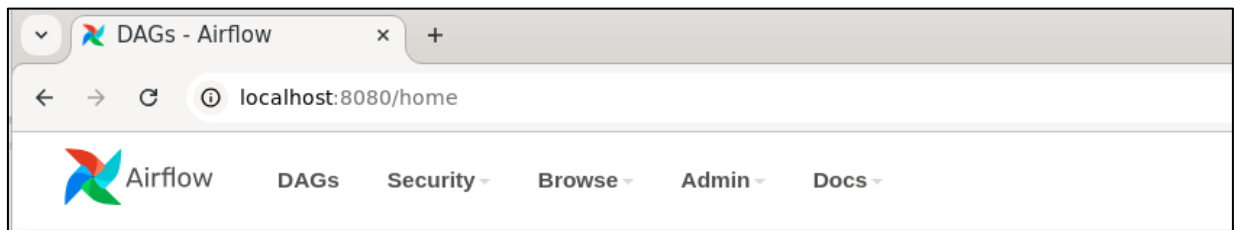


Рисунок 6 – Меню Apache Airflow

На первой вкладке представлена структура DAGa, узловыми точками которой являются задачи, справа от структуры метрика выполнения всего DAGa и его задач отдельно, сгруппированная по датам. Одна колонка метрики – это один запуск DAGa. Цветовая индикация, согласно легенде верху справа описывает статусы выполнения задач. Наведя курсор на задачу, мы можем увидеть ее свойства, пройдя по ней получим более подробное описание, и набор действий над ней.

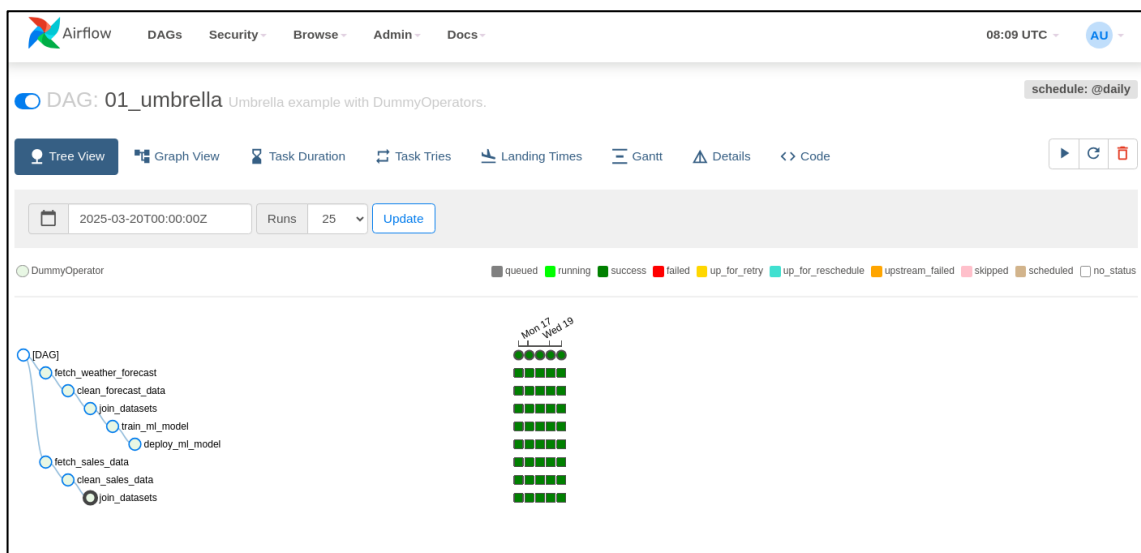


Рисунок 7 – Вкладка Tree views

На второй вкладке отображается графическое представление DAGa, с возможностью пройти в свойства составляющих его задач

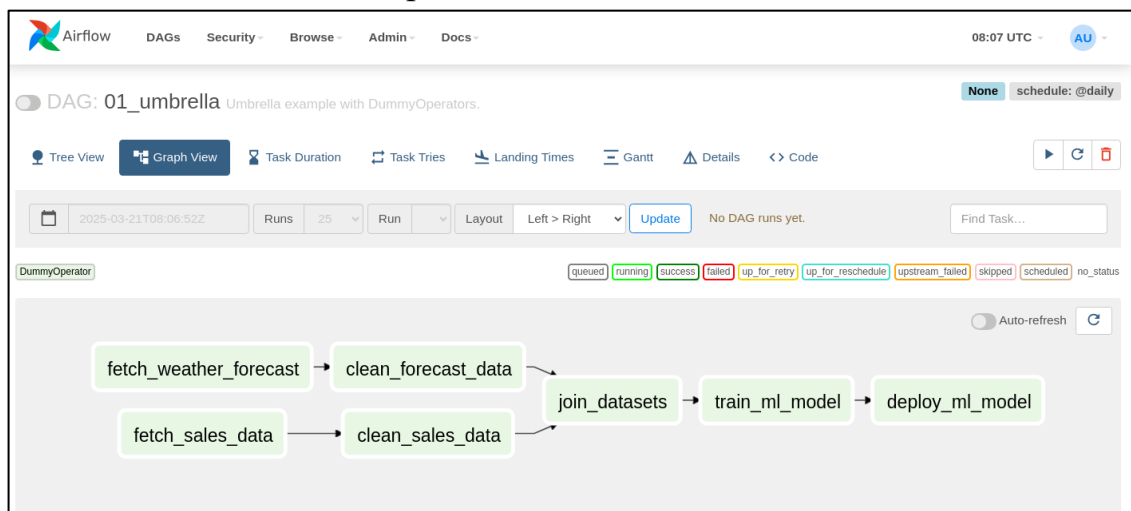


Рисунок 8 – Вкладка Graph View

В каждый блок можно перейти и посмотреть о нем информацию и выполнить операцию.

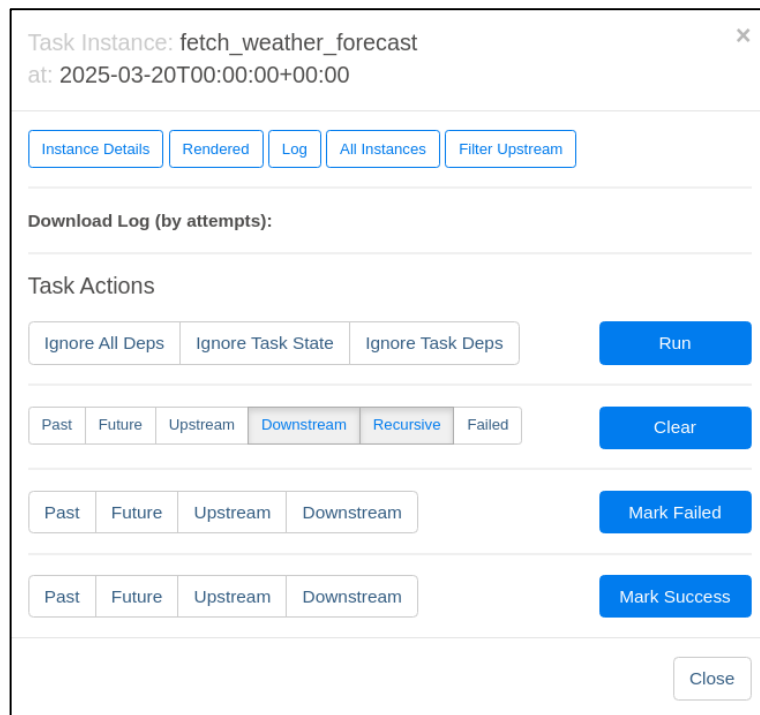


Рисунок 9 – Свойства задания fetch_weather_forecast

Посмотрим логи:

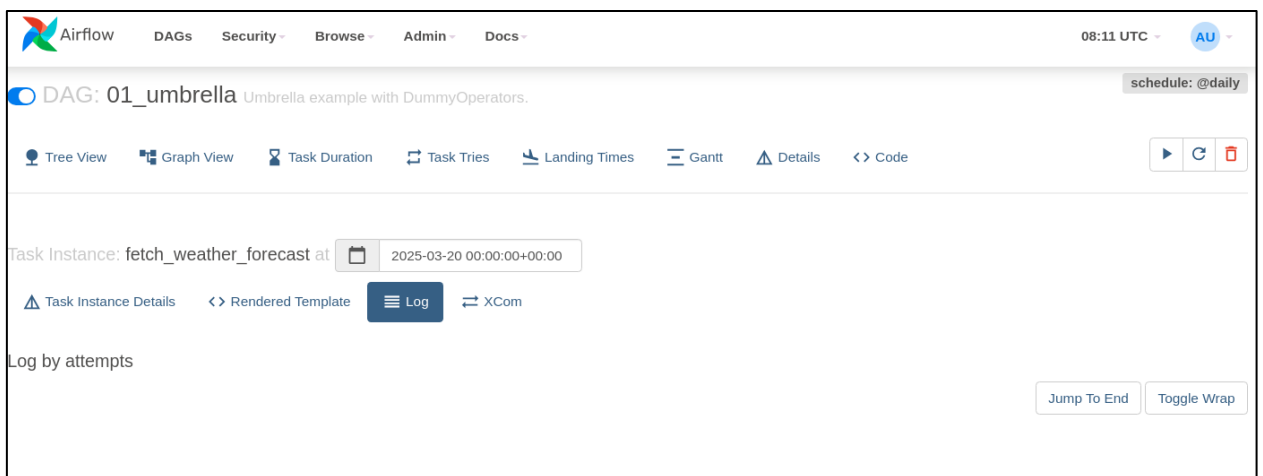


Рисунок 10 – Логи задачи fetch_weather_forecast

3. Спроектируем верхнеуровневую архитектуру аналитического решения задания Бизнес кейс Umbrella в draw.io. Необходимо использовать:

- Source Layer - слой источников данных.
- Storage Layer - слой хранения данных.
- Business Layer - слой для доступа к данным бизнес пользователей.

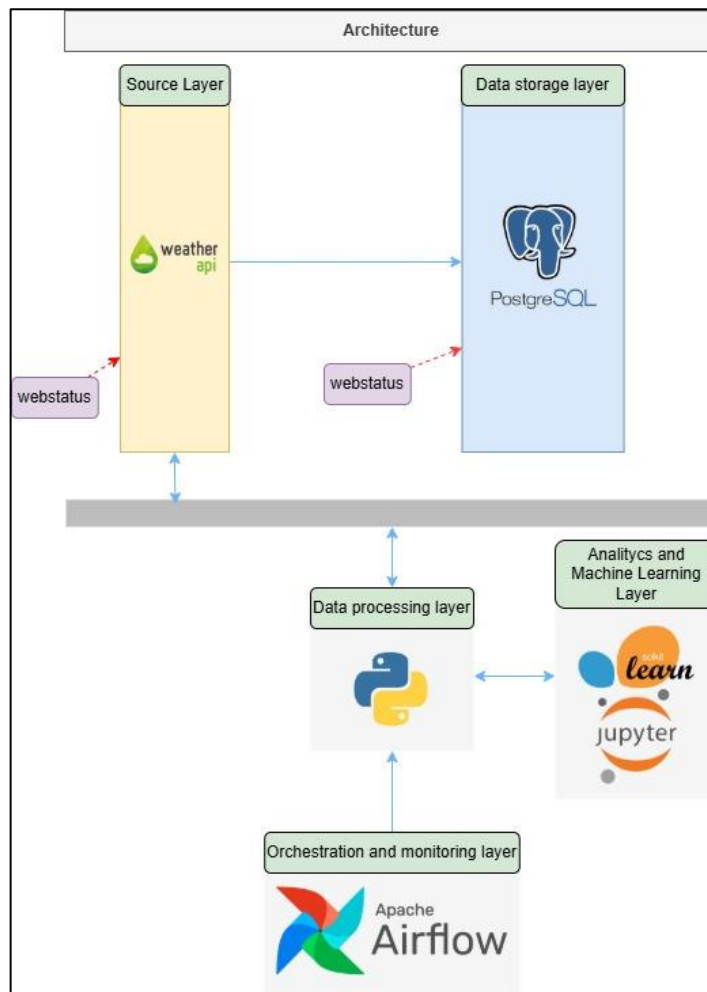


Рисунок 11 –Архитектура аналитического решения задания

4. Выполнение индивидуального задания:

Вариант 5:

1. Получить данные по координатам 48.85, 2.35 (Париж) на 7 дней

Выполняем необходимые изменения в файле `real_umbrella.py`

```
# 1. Получение прогноза погоды
def fetch_weather_forecast():
    api_key = "0ce927d8f8734ccca1e71309252103" # замените на ваш API ключ
    url = f"http://api.weatherapi.com/v1/forecast.json?key={api_key}&q=48.85,2.35&days=7"
    response = requests.get(url)
    data = response.json()
    forecast_data = [(day['date'], day['day']['avgtemp_c']) for day in data['forecast']['forecast']]
    df = pd.DataFrame(forecast_data, columns=['date', 'temperature'])
```

Рисунок 12 – Код для получения прогноза погоды

2. Оставить только столбцы: date, avgtemp_c

```
real_umbrella.py • 01_umbrella.py
dags > real_umbrella.py

24 # 1. Получение прогноза погоды
25 def fetch_weather_forecast():
26     api_key = "0ce927d8f8734ccca1e71309252103" # замените на ваш API ключ
27     url = f"http://api.weatherapi.com/v1/forecast.json?key={api_key}&q=48.85,2.35&days=7"
28     response = requests.get(url)
29     data = response.json()
30     forecast_data = [(day['date'], day['day']['avgtemp_c']) for day in data['forecast']['forecastday']]
31     df = pd.DataFrame(forecast_data, columns=['date', 'avgtemp_c'])
32     data_dir = '/opt/airflow/data'
33     os.makedirs(data_dir, exist_ok=True)
34     df.to_csv(os.path.join(data_dir, 'weather_forecast.csv'), index=False)
35     print("Weather forecast data saved.")
36
```

Рисунок 13 – Код для получения прогноза погоды с необходимыми столбцами

```
real_umbrella.py × 01_umbrella.py
dags > real_umbrella.py

25 def fetch_weather_forecast():
36
37 # 2. Очистка данных погоды
38 def clean_weather_data():
39     data_dir = '/opt/airflow/data'
40     df = pd.read_csv(os.path.join(data_dir, 'weather_forecast.csv'))
41     df['avgtemp_c'] = df['avgtemp_c'].fillna(method='ffill')
42     df.to_csv(os.path.join(data_dir, 'clean_weather.csv'), index=False)
43     print("Cleaned weather data saved.")
44
```

Рисунок 14 – Код для очистки данных погоды

```
# 6. Обучение ML модели
def train_ml_model():
    data_dir = '/opt/airflow/data'
    df = pd.read_csv(os.path.join(data_dir, 'joined_data.csv'))
    X = df[['avgtemp_c']]
    y = df['sales']
    model = LinearRegression()
    model.fit(X, y)
    joblib.dump(model, os.path.join(data_dir, 'ml_model.pkl'))
    print("ML model trained and saved.")
```

Рисунок 15 – Код для обучения ML модели

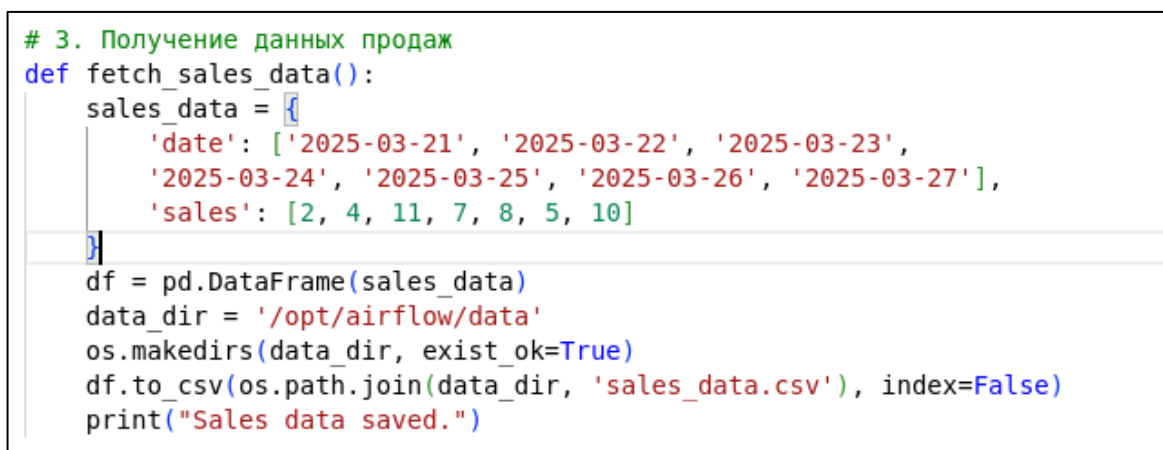
3. Найти минимальную температуру



```
real_umbrella.py x 01_umbrella.py
dags > real_umbrella.py
38 def clean_weather_data():
39     print("cleaned weather data saved.")
40
41
42
43
44
45 def find_min_temperature():
46     data_dir = '/opt/airflow/data'
47     df = pd.read_csv(os.path.join(data_dir, 'clean_weather.csv'))
48     min_temp = df['avgtemp_c'].min()
49     print(f"Минимальная температура за 7 дней: {min_temp}°C")
50
51
```

Рисунок 16 – Код для поиска минимальной температуры

Заведем данные по продаже зонтиков для 7 дней:



```
# 3. Получение данных продаж
def fetch_sales_data():
    sales_data = {
        'date': ['2025-03-21', '2025-03-22', '2025-03-23',
                '2025-03-24', '2025-03-25', '2025-03-26', '2025-03-27'],
        'sales': [2, 4, 11, 7, 8, 5, 10]
    }

    df = pd.DataFrame(sales_data)
    data_dir = '/opt/airflow/data'
    os.makedirs(data_dir, exist_ok=True)
    df.to_csv(os.path.join(data_dir, 'sales_data.csv'), index=False)
    print("Sales data saved.")
```

Рисунок 17 – Код для получения данных продаж

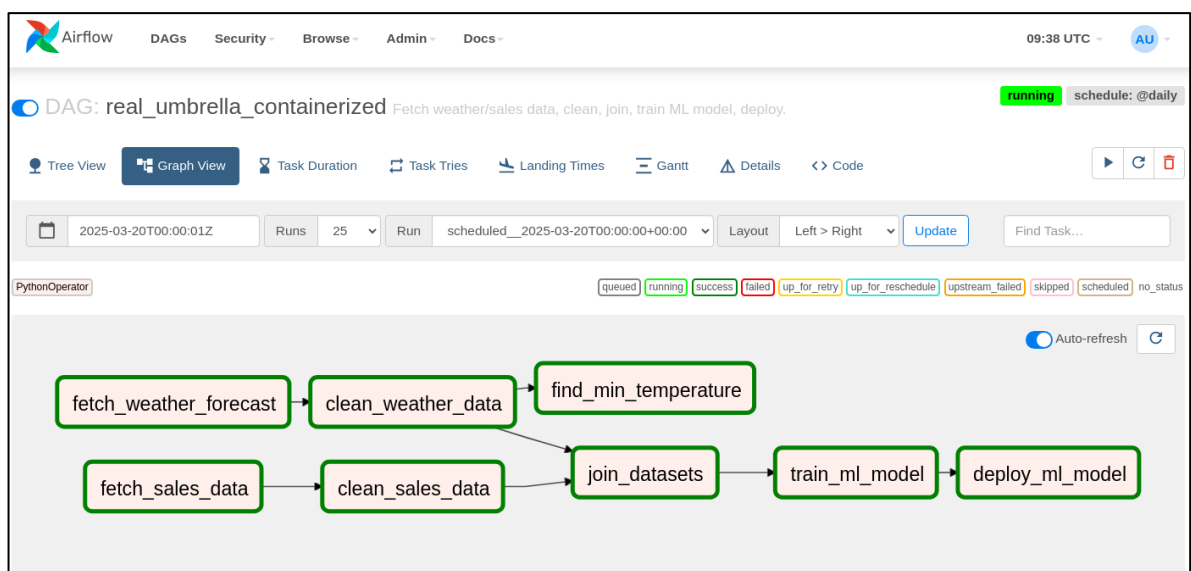


Рисунок 18 – Граф в Apache Airflow

Посмотрим лог задачи find_min_temperature и убедимся в том, что минимальная температура вычисляется


```
*** Reading local file: /opt/airflow/logs/real_umbrella_containerized/find_min_temperature/2025-03-20T00:00:00+00:00/1.log
[2025-03-21 09:35:54,201] {taskinstance.py:826} INFO - Dependencies all met for <TaskInstance: real_umbrella_containerized.find_min_temperature 2025-03-20T00:00:00+00:00 [queued]>
[2025-03-21 09:35:54,273] {taskinstance.py:826} INFO - Dependencies all met for <TaskInstance: real_umbrella_containerized.find_min_temperature 2025-03-20T00:00:00+00:00 [queued]>
[2025-03-21 09:35:54,273] {taskinstance.py:1017} INFO -
-----
[2025-03-21 09:35:54,273] {taskinstance.py:1018} INFO - Starting attempt 1 of 1
[2025-03-21 09:35:54,273] {taskinstance.py:1019} INFO -
-----
[2025-03-21 09:35:54,305] {taskinstance.py:1038} INFO - Executing <Task(PythonOperator): find_min_temperature> on 2025-03-20T00:00:00+00:00
[2025-03-21 09:35:54,333] {standard_task_runner.py:51} INFO - Started process 1609 to run task
[2025-03-21 09:35:54,333] {standard_task_runner.py:75} INFO - Running: ['airflow', 'tasks', 'run', 'real_umbrella_containerized', 'find_min_temperature', '2025-03-20T00:00:00+00:00']
[2025-03-21 09:35:54,334] {standard_task_runner.py:76} INFO - Job 7: Subtask find_min_temperature
[2025-03-21 09:35:54,459] {logging_mixin.py:103} INFO - Running <TaskInstance: real_umbrella_containerized.find_min_temperature 2025-03-20T00:00:00+00:00 [running]> on host 5581
[2025-03-21 09:35:54,640] {taskinstance.py:1230} INFO - Exporting the following env vars:
AIRFLOW_CTX_DAG_OWNER=airflow
AIRFLOW_CTX_DAG_ID=real_umbrella_containerized
AIRFLOW_CTX_TASK_ID=find_min_temperature
AIRFLOW_CTX_EXECUTION_DATE=2025-03-20T00:00:00+00:00
AIRFLOW_CTX_DAG_RUN_ID=scheduled_2025-03-20T00:00:00+00:00
[2025-03-21 09:35:54,666] {logging_mixin.py:103} INFO - Минимальная температура за 7 дней: 9.4°C
[2025-03-21 09:35:54,666] {python.py:118} INFO - Done. Returned value was: None
[2025-03-21 09:35:54,699] {taskinstance.py:1135} INFO - Marking task as SUCCESS. dag_id=real_umbrella_containerized, task_id=find_min_temperature, execution_date=20250320T000000
[2025-03-21 09:35:54,798] {taskinstance.py:1195} INFO - 0 downstream tasks scheduled from follow-on schedule check
[2025-03-21 09:35:54,821] {local_task_job.py:118} INFO - Task exited with return code 0
```

Рисунок 19 – Лог find_min_temperature

Убедимся, что это так – перейдем по соответствующему адресу:

```
{
  "location": {
    "name": "Paris",
    "region": "Ile-de-France",
    "country": "France",
    "lat": 48.867,
    "lon": 2.333,
    "tz_id": "Europe/Paris",
    "localtime_epoch": 1742547551,
    "localtime": "2025-03-21 09:59"
  },
  "current": {
  },
  "forecast": {
    "forecastday": [
      {
        "date": "2025-03-21",
        "day": {
          "avgtemp_c": 14.5
        }
      },
      {
        "date": "2025-03-22",
        "day": {
          "avgtemp_c": 13
        }
      },
      {
        "date": "2025-03-23",
        "day": {
          "avgtemp_c": 9.4
        }
      },
      {
        "date": "2025-03-24",
        "day": {
          "avgtemp_c": 11.4
        }
      },
      {
        "date": "2025-03-25",
        "day": {
          "avgtemp_c": 11.1
        }
      },
      {
        "date": "2025-03-26",
        "day": {
          "avgtemp_c": 11.5
        }
      },
      {
        "date": "2025-03-27",
        "day": {
          "avgtemp_c": 10.5
        }
      }
    ]
  }
}
```

Рисунок 20 – Выведенные данные при обращении по API

Выгрузим обученную модель:

```
dev@dev-vm:~/Downloads/business_case_umbrella_25$ sudo docker cp business_case_umbrella_25-webserver-1:/opt/airflow/data/ml_model.pkl ./ml_model.pkl
Successfully copied 2.56kB to /home/dev/Downloads/business_case_umbrella_25/ml_model.pkl
```

Рисунок 21 – Выполнение команды для выгрузки обученной модели

Загрузим обученную модель и попытаемся вычислить количество продаж зонтиков для разных средних температур:

```
import joblib
model = joblib.load("ml_model.pkl")
import pandas as pd
print(model.predict(pd.DataFrame({'avgtemp_c': [11]}))) # Например, прогноз продаж при 15°C
```

[7.8817094]

Рисунок 22 – Количество зонтиков для средней температуры 11 градусов

```
import joblib
model = joblib.load("ml_model.pkl")
import pandas as pd
print(model.predict(pd.DataFrame({'avgtemp_c': [9]})))
```

[11.59623932]

Рисунок 23 – Количество зонтиков для средней температуры 9 градусов

Выводы:

В ходе работы мы изучили работу с weather api, контейнерами и apache airflow. Нам удалось получить данные о погоде по необходимой геопозиции на несколько дней, выделить необходимые показатели и найти минимальную среднюю температуру за 7 дней. Также были предсказаны продажи зонтиков по средней дневной температуре.